Tomas Bata University in Zlín
Faculty of Applied Informatics

Doctoral Thesis

Optimisation of Software Effort Estimation by Improving Functional Points Analysis

Optimalizace odhadu úsilí vývoje softwaru zlepšením analýzy funkčních bodů

Author:	**Vo Van Hai**
Degree programme:	Engineering Informatics
Degree course:	Software Engineering
Supervisor:	Assoc. Prof. Ing. Zdenka Prokopová, CSc.
Consulting Supervisor:	Assoc. Prof. Ing. Radek Šilhavý, Ph.D.

Zlín, October 2022

ACKNOWLEDGEMENT

Completing my Ph.D. degree would not have been possible without the encouragement and support of my supervisors, family, friends, and colleagues. The journey to a foreign country for me until today has been very long and challenging. Your advice, support and beliefs have helped me immensely in my journey to where I am today.

I would like to express my deep gratitude to the Faculty of Applied Informatics of Tomas Bata University for allowing me to participate in this doctoral course.

I would like to express my deepest gratitude and sincere respect to my supervisor, doc. Ing. Zdenka Prokopova, CSc. and doc. Ing. Radek Šilhavý, PhD. I sincerely thank you for your expert guidance, patience, constructive criticism, careful supervision, and unwavering encouragement throughout my years of study. Through this, I would also like to express my deep gratitude to doc. Ing. Petr Šilhavý, Ph.D., who has been of great assistance to me in the publication of my research and other scholarly contributions. It is a privilege and an honour to work with you.

I would like to thank my colleagues for sharing knowledge and life experiences during my time studying this course.

I would like to thank all the staff and faculty members of the Faculty Applied Informatic, UTB, for assisting me in various ways.

Finally, I wish to thank my family for their endless love, tolerance, encouragement, and support. I cannot describe the sacrifices and efforts that my wife, Nguyen Thi Thanh Binh, made during my time at school in raising children and ensuring family life. I wouldn't have been able to study without her caring for my family. Thank you, mom, for the kindness of giving birth and nurturing. Thank my children for your precious encouragement and efforts during my absence.

ABSTRAKT

Předkládaná disertační práce představuje novou metodu odhadu vývojového úsilí softwaru pomocí technik strojového učení. Hlavní myšlenkou práce bylo představit nový systém vah, které se využívají v metodě Function Points Analysis pro kalibrování odhadu rozsahu softwaru. Dále byl navržen nový optimalizační rámec pro výpočet výsledků odhadu úsilí. Pro realizaci tohoto návrhu bylo nezbytné provést výběr vhodného algoritmu strojového učení a zkoumat vliv rozsahu dat na přesnost odhadu. Ukázalo se, že shlukování dat má velký vliv na přesnost výsledků odhadů. Z toho důvodu byly provedeny experimenty na vyhodnocení nejvhodnějšího shlukování.

Výsledky získané v této disertační práci byly hodnoceny podle několika hodnotících kritérií a dosáhly mnohem lepšího výsledku než původní metoda FPA nebo další srovnávané metody.

Key words in Czech: *odhad úsilí vývoje softwaru, analýza funkčních bodů, váhy funkční složitosti, kategoriální proměnné, klastrování dat, strojové učení*

ABSTRACT

The doctoral thesis proposes a new method of software effort estimation using machine learning techniques. The main idea of the work was to present a new weighting system of calibration complexity applied in the Function Points Analysis (FPA) method and to propose an optimization framework for the calculation of effort estimation results. The selection of a suitable machine learning algorithm is necessary for the implementation of this proposal. In addition, other attributes were investigated. Data clustering has been shown to have a large effect on the accuracy of estimation results. For that reason, experiments were made to find the most suitable clustering mechanism.

The results obtained in this dissertation were evaluated according to unbiased evaluation criteria and achieved a much better result than the original FPA method and other compared methods.

Key words: *software effort estimation, function point analysis, calibration complexity weight, categorical variables, data clustering, machine learning*

Contents Of the Thesis

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

Abbreviation	Definition
FPA	Function Points Analysis
ML	Machine Learning
IFPUG	International Function Point User Group
ISBSG	International Software Benchmarking Standards Group
WBS	Work Breakdown Structure
COCOMO	Constructive Cost Model
UCP	Use Case Points
OOM	Object-Oriented Modeling
UML	Unified Modeling Language
UAW	Unadjusted Actor Weight
UUCW	Unadjusted Use Case Weight
UUCP	Unadjusted Use Case Point
ILF	Internal Logic Files
EIF	External Interface Files
EI	External Input
EO	External Output
EQ	External Inquiry
DET	Data Element Type
RET	Record Element Type
FTR	File Type Reference
UFP	Unadjusted Function Point
VAF	Value Adjustment Factor
GSC	General System Characteristic
AFP	Adjusted Function Points
MPCV	Metrics Practices Committee
UKSMA	UK Software Metrics Association
COSMIC	Common Software Measurement International Consortium
BFC	Base Functional Component
FiSMA	Finnish Software Measurement Association
FSM	Function Size Measurement
NESMA	Netherlands Software Metrics Association
FPM	Function Point Modeler
CPM	Counting Practice Manual

BIRCH/BIR	Balanced Iterative Reducing and Clustering Hierarchies
CF	Clustering Feature
FCM	Fuzzy C-Means
GMM	Gaussian Mixture Model
SC	Spectral clustering
MS	MeanShift
MLR	Multiple Linear Regression
ANN	Artificial Neural Networks
MLP	Multiple Layer Perception
SVM	Support Vector Machine
SVR	Support Vector Regression
BRR	Bayesian Ridge Regression
LASSO/LAS	Least Absolute Shrinkage and Selection Operator
ABC	Artificial Bee Colony
MCS	Modified Cuckoo Search
FLANN	Functional Link Artificial Neural Network
CFCW	Calibration of Functional Complexity Weight
CFCWO	CFCW Optimization
IQR	Interquartile Range
DT	Development Type
DP	Development Platform
LT	Language Type
IS	Industry Sector
OT	Organization Type
RS	Relative Size
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
MBRE	Mean Balance Relative Error
MIBRE	Mean Inverted Balance Relative Error
MAPE	Mean Absolute Percentage Error
PRED(x)	Prediction at level x
BAN	Banking
COM	Communication
FIN	Financial
INS	Insurance
MAN	Manufacturing

SI	Service Industry
3GL	3rd Generation Programming Language
4GL	4th 3rd Generation Programming Language
MR	Mid-Range
MF	Main Frame
Multi	Multi-Platform

1. INTRODUCTION

Nowadays, software plays a huge role in every aspect of modern society. Our daily works are directly or indirectly related to software products: looking up information, searching for buses, working with banks, entertaining and many other activities. From the beginning of the software age, the software was relatively simple, and it has become increasingly complex due to increasing human needs today.

The development of a software system that meets the real needs of modern society is a complex and challenging process to control. These systems require high adaptability, intelligence, and flexibility. Therefore, the software development team needs to be highly qualified and proficient. In addition, the control and management of the development process also require professionalism, ease of management, and predictability of arising cases. Over the past decades, broad research has been done, and many propositions have been made to make the software project management process more flexible and efficient.

In the process of developing a software system, a significant problem is software estimation. Software estimation relates to estimating the size of the development product, the effort in-person/months or person/hours, the project cost in the agreed currency, and the schedule in calendar months. It is considered a critical activity in broad aspects of software project management, which is defined as planning and controlling the development of a system at the optimal cost of meeting requirements. It is a recognized fact that a lot of software fails due to faulty project management practices. Billions of dollars are wasted on entirely preventable mistakes every year.

As we know, when starting a project, we need to study its feasibility and many other vital issues to implement the project. According to Standish Group [2], up to 83.9% of software projects fail partially or entirely. There are many reasons for this failure. According to Charette [3], the various common factors behind the failure of a software project are: a)Unrealistic or unarticulated project goals, b) inaccurate estimates of needed resources, c) badly defined system requirements, d) poor reporting of the project's status, e) unmanaged risks, f) poor communication among customers, developers, and users, g) use of immature technology, h) inability to handle the project's complexity, j) sloppy development practices, k) poor project management, l) stakeholder politics, and m) commercial pressures.

There are three aspects of the outcome of the estimation process: under-estimation, relative estimation accuracy, and over-estimation. Overestimating will lead to wasted resources and even customer rejection or giving up bids. Conversely, underestimating leads to overbudgeting, understaffing, and delayed deliveries. Both of these issues cause significant financial loss and potential loss

of contract [4]. But how to predict an acceptable outcome has never been an easy problem.

1.1 Motivation

Many previous studies have related to software estimation, from making predictive models to applying artificial intelligence. We can categorize these approaches according to 3 categories [5] [6]: 1) non-algorithmic approaches, 2) Algorithmic approaches, and 3) Machine Learning (ML)techniques.

In the first group, experts and historical sample projects play a decisive role in making the final prediction. The estimation depends on the team of experts and the company's development history. It is the strength as well as the weakness of this approach. Some typical methods are introduced in section 2.1.

In the second group, the estimation is based on mathematical processes and formulas. This approach is based on procedures that use a mathematical equation in the form of a multivariable function to represent the relationship between the parameters of the software project to be estimated. That means this approach is not subjectively dependent on a particular object. Some typical methods of this approach will be presented in section 2.2. Among the methods following this approach, the International Function Point User Group (IFPUG) Function Points Analysis (FPA) is considered the model widely applied in the software industry. This method aims to calculate the software's size based on the Function Points (FPs). From the estimated software size, the required effort is determined. From the effort estimation, product completion time and cost are found.

With the machine learning approach, the estimation will be based on machine learning algorithms to build the model. The model will be trained and tested based on datasets (which can be well-known datasets or company datasets). When it is necessary to estimate a new project, parameters will have to be provided, and the target value will be derived from the previously trained model. This model has been and is being used a lot in recent years [7]. It is even seen as an alternative to the two other approaches.

It can be seen that each approach has different strengths and weaknesses [8] [9]. Depending on the specular situation of each software development organization, the appropriate approach should be chosen.

One question arises about whether we can combine the strengths of the different approaches to reduce the error in the estimation process. That is to say, at some stage of this approach, we take another approach to solving the problem. In other words, we combine approaches to solve problems. That is also the driving force of this study.

1.2 Problem statement

Among the methods that follow the algorithmic approach, FPA is considered the most commonly used method in the software industry. It has the advantage of estimating the software size from which other values, such as effort, schedule, and cost can be calculated. However, this method also has many limitations [10]. One of its most significant limitations is the complexity weight constant values built on the IBM dataset in the 1970s [11]. Therefore, according to the technology and related issues updated to the present time (2022), it has become out-of-date. Besides, due to each company's specific nature, the accuracy will obviously be low when applying this set of constants.

Therefore, this study will propose a calibration complexity weight constant system based on the machine learning technique with the International Software Benchmarking Standards Group (ISBSG) [] as the training/testing dataset. ISBSG is a dataset contributed by companies worldwide, so the use of it with the expectation of reducing the locality of a particular company (IBM). Besides, with the August 2020 R1 version, the out-of-date is also excluded. In addition, the proposed model can also be applied to the software development company's dataset to optimize the estimate.

This study's first problem was using the machine learning technique to calibrate the complexity weight. Out of the many machine learning techniques, which one would be the best fit for this process? Obviously, we can hardly test on all known algorithms. Therefore, selecting some experimental algorithms is also a matter of concern.

Besides, many studies also show that applying data segmentation will increase the accuracy of the estimation process [12], [13]. However, the criterion for clustering is a matter of consideration because they have many methods. Choosing a suitable clustering method is also a significant issue in applying data segmentation to the estimation process.

In addition, with the estimation results based on machine learning techniques on specific clusters, can this result be again optimized to improve the accuracy of the estimate? Many studies have determined that the results will be better with ensemble techniques than with using an individual algorithm [14], [15]. Therefore, in this study, we also propose an optimization framework that uses an ensemble technique to improve the accuracy of the estimated results.

As mentioned, we can infer the effort from the software size and then determine the cost and schedule. Thus, the effort is the most critical intermediate value after determining size. However, a complex issue incredibly involved in the effort prediction process is the value of "productivity". This value depends a lot on the software development and management team. Human resource is a sensitive issue in the software development process (and other fields). The project completion

rate is high with a good team understanding of the problem to be implemented, mastering development techniques, and effective teamwork. However, such a team is desirable, and it isn't easy to have such a comprehensive team for every software development case of an organization. Therefore, this study pays the most significant attention to software effort estimation.

1.3 Objectives of the thesis

With the aim of increasing the accuracy in software effort estimation, this study uses the IFPUG FPA method combined with machine learning techniques to allow effort estimation on groups segmented by Industry Sector categorical variable.

It is necessary to determine the appropriate clustering criteria and algorithm to solve this target. With the determination of the proper clustering criteria, the clustering algorithms and categorical variables are assessed to determine the most suitable clustering criteria. With the determination of the most suitable algorithm, a survey was conducted to identify some of the most commonly used algorithms recently and then performed an evaluation on these algorithms to find the most appropriate algorithm. In addition, the results of the FPA counting process will also be optimized again to improve the accuracy of the final estimate.

Hence, with the above problems, the aim of this thesis is:

1) A calibration complexity weight system for the IFPUG FPA method is proposed.
2) Create an effort estimation optimization framework based on regression models, machine learning, clustering, and the use of categorical variables.
3) Based on experiments, compare the proposed approaches with the reference method FPA and the tested approaches with each other.
4) Assess the contribution of the proposed optimization procedures to the refinement of the FPA method used for software estimation.

1.4 Dissertation layout

Chapter 1, "Introduction", identifies the motivation, the problem to be studied, and the research goal.

Chapter 2, "Software Estimation Overview", introduces current approaches to software estimation. This chapter presents three common approaches: non-algorithmic, algorithmic and machine learning. Each approach is introduced, along with some common methods. In addition, several software tools used in the software industry are also presented.

Chapter 3, "Machine Learning Algorithms used", introduces the machine learning algorithms used in this thesis. It includes clustering algorithms, software estimation algorithms, and ensemble algorithms for the optimization of estimation results.

Chapter 4, "Current State of Effort Estimation", introduces the current state of approaches in effort estimation. This gives us an overview of related studies.

Chapter 5, "Proposed Method," describes the proposed approach of this thesis. The main content here is to propose a calibration complexity weight system according to the Industry Sector categorical variable. To be able to generate this calibration complexity weight system, it is necessary to define an algorithm that fits the applied dataset and the appropriate clustering criteria. The results obtained from the estimation using the IFPUG FPA method with the new complexity weight system will be optimized again to get the final result.

Chapter 6, "Experiment Part", presents the experiment based on which to achieve the complexity weight system and optimization framework as described in section 4. In addition, this chapter also demonstrates the evaluation criteria.

Chapter 7, "Results and Discussion," presents the experimental results obtained from the experiment. The calibration complexity weight system will be introduced in this section, and the evaluation results based on the criteria are also presented.

Chapter 8, "Threat of Validity," covers the risks that affect test results and how to deal with them to ensure the results are as accurate and objective as possible.

Chapter 9, "Contribution of the thesis to science and practice", presents the main contributions of the thesis to science as well as practical application.

Chapter 10 presents the conclusions as well as future work.

Chapter 11 contains all references used in the doctoral theses.

2. SOFTWARE ESTIMATION OVERVIEW

Previous researchers have proposed many software estimation methods. Classifying them is not easy because of the properties that can be difficult to distinguish. As mentioned in the motivation section, studies [5] [6] indicate that these approaches can be categorized into the following three categories:

1) Non-algorithmic approaches
2) Algorithmic approaches
3) Using machine learning

However, this category has a relative meaning. At a particular stage of this approach, it is possible to apply a part or the whole of the other approaches. Following are the approaches and some representation methods.

2.1 Non-algorithmic approaches

2.1.1 Analogy Technique

In a software development organization, information about past projects is stored as a database for reference when a new project is available. When starting a new project, it was evident that project information and many other problems were missing. As such, whether the new project is worthwhile to develop or likely to succeed. In case it is feasible, how much human resources, how much money, and how long will it take to develop? These issues are the core issues. Thus, based on "experience" from existing projects, managers will look for similar projects or experiences from other projects to be able to estimate these key issues.

Analogous estimation is a technique used to estimate the parameters for a developing project based on the parameters of previous projects. These parameters are size, weight, complexity, scope, cost, and duration. This estimation process is seen as a combination of historical information of completed projects and expert judgment. The steps involved in this technique are shown in Fig. 2-1.

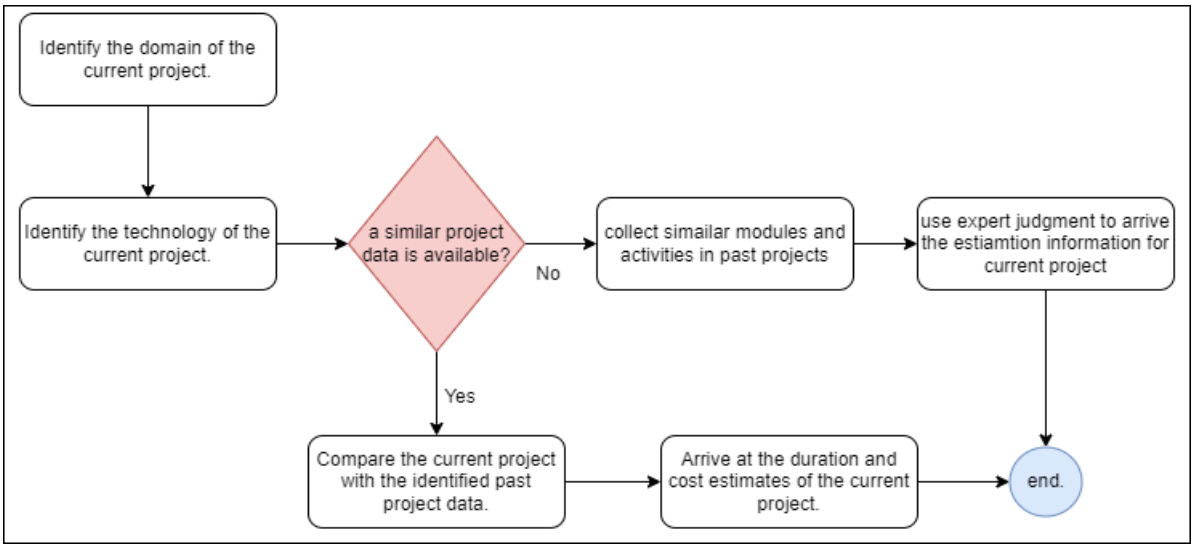


Fig. 2-1. Analogous estimation steps

This technique is suitable for the very early stages of a project when the information the project will develop is very limited. This technique requires very little time to perform the estimation. However, this technique depends a lot on the information from previously implemented projects that not all companies are diverse and rich. In addition, expert judgment is a very subjective assessment and depends entirely on the experience and level of the expert.

2.1.2 Wideband Delphi

Delphi is a structured interaction technique derived from symmetric prediction and interactive forecasting based on expert questionnaires [16] [17].

The method is described as follows: A questionnaire is given, and experts (anonymously) have to answer these questions in different rounds (two or more rounds). After each round, the coordinator will summarize the experts' predictions from the previous round. Experts must state the reasons for their choice. They (experts) are also encouraged to see other experts' questions and answers and to review their answers. After a certain number of rounds, the experts will agree on the parameters (the stability of results, consensus of experts). Through this process, the answer area will be reduced, and the group of experts will be closer to the correct problem [18].

Boehm [19] proposed a variant of the Delphi method and called it Wideband Delphi. The term "wideband" refers to the more significant interaction and communication between the experts involved than in the original Delphi method. The steps of this process can be depicted as shown in Fig. 2-2.

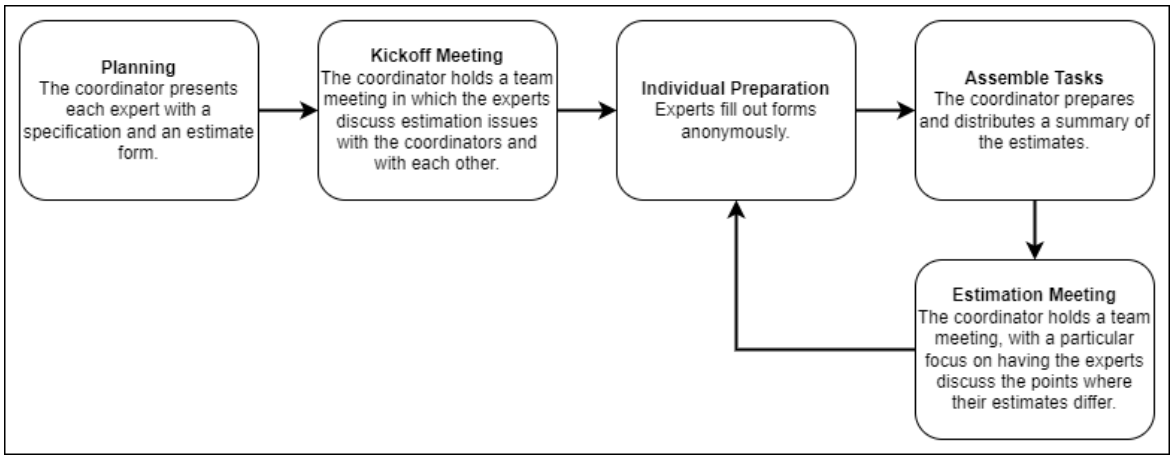


Fig. 2-2. Wideband Delphi process

The estimation process is completed when specified exit criteria are satisfied. Typical Wideband Delphi exit criteria are that:

- The overall task list has been assembled.

- You have a summarized list of estimating assumptions.

- The estimators have agreed on the process used to combine their estimates into a single set with a tolerable range.

2.1.3 Work Breakdown Structure

Work Breakdown Structure (WBS) is a concept used to describe the splitting and breaking of work objects (projects) into smaller components in project management and systems engineering. This procedure aims to make project deployment and management simpler and more practical. Small tasks must be guaranteed to fall within the scope of the task's target resolution. These small jobs can have a binding or independent relationship with each other. A job is considered complete when all its minor jobs are completed.

The smallest unit of work that is separated is called a work package. A work package can be used for group activities. Each activity will be scheduled, estimated, monitored and controlled. That means the project deliverables will be decomposed into work packages (managed by the WBS structure). Then the work packages will continue to be decomposed to a lower level called activities (managed by operational teams).

In the process of building WBS, when we separate the work content, we should set the priority of the work package. Fig. 2-3 shows an example of the WBS structure of a project.

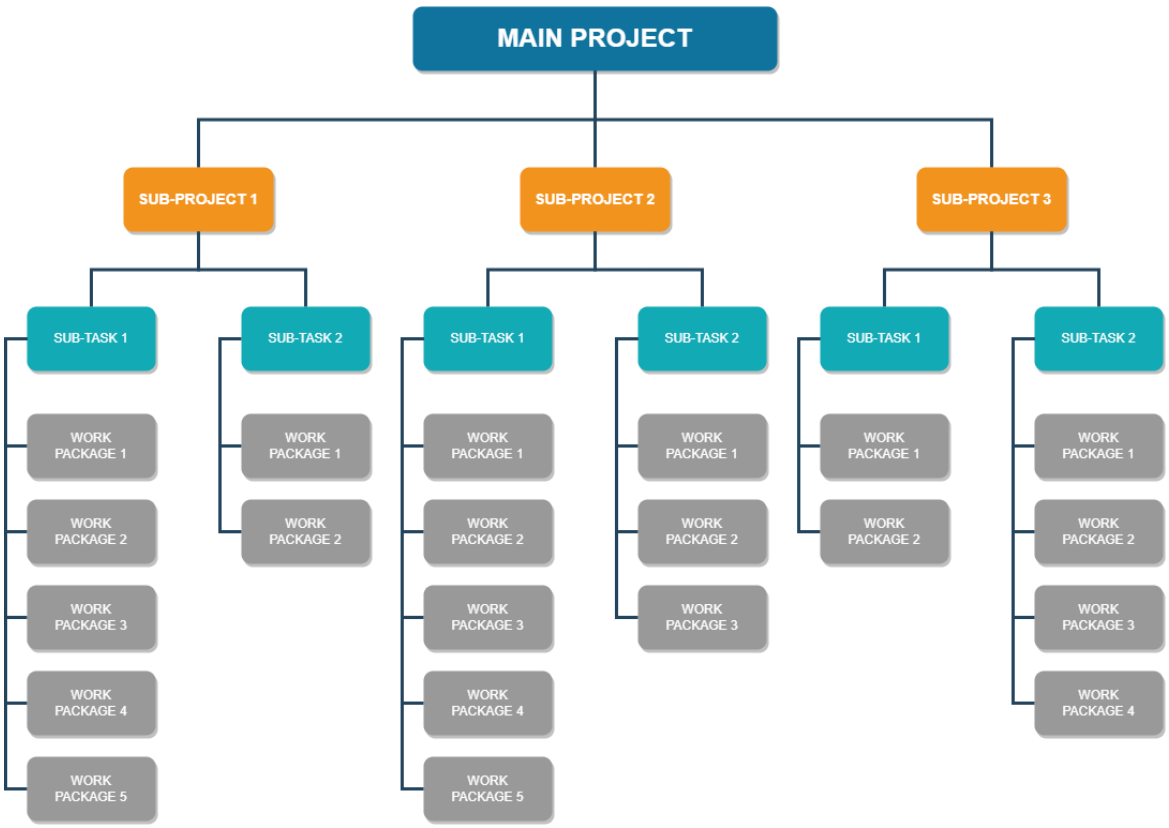


Fig. 2-3. A sample WBS

2.2 Algorithmic approaches

2.2.1 Constructive Cost Model

The Constructive Cost Model (COCOMO) was first proposed by Boehm [19]. It is a lines-of-code-based regression model. It serves as a procedural cost estimate model for software projects and is frequently applied as a method for accurately forecasting the different project-related characteristics, including size, effort, cost, time, and quality.

We can apply different COCOMO models to estimate costs depending on different levels. These levels are based on the desired level of precision and correctness. For each level, different constants will be applied in the calculation formula. Projects using the COCOMO method are categorized into three categories: organic, semi-detached, and embedded.

If the team size necessary is suitably small, the problem is well understood and has already been solved, and the team members have an amount of expertise with the problem, the software project is said to be of the organic kind.

A software project is referred to as Semi-detached if key elements like team size, expertise, and familiarity with several programming environments fall between Organic and Embedded. Semi-detached projects demand more expertise,

better supervision, and creative thinking than organic projects because they are less familiar and more challenging to create.

The embedded category includes software projects with the most significant levels of complexity, inventiveness, and experience requirements. In comparison to the other two models, this software needs a larger team, and the programmers must have the necessary expertise and imagination to create such intricate models.

The COCOMO model includes three increasingly detailed and accurate forms: Basic, Intermediate, and Detailed. We should choose the correct form based on the requirements.

Basic COCOMO model

$$Effort = a * (KLOC)^b \quad (2.1)$$

$$Time = c * (Effort)^d \quad (2.2)$$

$$Person\ Required = Effort/Time \quad (2.3)$$

where,

$a, b, c, and\ d$ are constants (see Table 2-1)

KLOC is the size of code in Kilo lines-of-code

Table 2-1. The constant values for the Basic COCOMO model

	Organic	**Semi-Detached**	**Embedded**
a	2.4	3.0	3.6
b	1.05	1.12	1.2
c	2.5	2.5	2.5
d	0.38	0.35	0.32

Intermediate COCOMO Model

The Basic COCOMO model is expanded upon in the Intermediate COCOMO model, which includes a set of cost drivers to enhance the cost estimation model's accuracy. The estimation model uses a set of "cost driver attributes" to compute the cost of the software. The relationship gives the estimated effort and scheduled time:

$$Effort = a * (KLOC)^b * EAF \quad (2.4)$$

$$Time = c * (Effort)^d \quad (2.5)$$

where,

a, b, c, and d are constant values that can be found in Table 2-2.

EAF is the Effort Adjustment Factor. This value can be found in Table 2-3

Table 2-2. The constant values for the Intermediate COCOMO model

Constant	**Organic**	**Semi-Detached**	**Embedded**
a	3.2	3.0	2.8
b	1.05	1.12	1.2
c	2.5	2.5	2.5
d	0.38	0.35	0.32

Table 2-3. The cost drivers of the Intermediate COCOMO model

Abbr.	Attributes	Very Low	Low	Nominal	High	Very High	Extra High
Product Attributes							
RELY	Required software reliability extent	0.75	0.88	1.00	1.15	1.40	
DATA	Size of the application database		0.94	1.00	1.08	1.16	
CPLX	The complexity of the product	0.70	0.85	1.00	1.15	1.30	1.65
Hardware Attributes							
TIME	Run-time performance constraints			1.00	1.11	1.30	1.66
STOR	Memory constraints			1.00	1.06	1.21	1.56
VIRT	The volatility of the virtual machine environment		0.87	1.00	1.15	1.30	
TURN	Required turnabout time		0.87	1.00	1.07	1.15	
Personnel Attributes							
ACAP	Analyst capability	1.46	1.19	1.00	0.86	0.71	
AEXP	Software engineering capability	1.29	1.13	1.00	0.91	0.82	
PCAP	Applications experience	1.42	1.17	1.00	0.86	0.70	
VEXP	Virtual machine experience	1.21	1.10	1.00	0.90		
LEXP	Programming language experience	1.14	1.07	1.00	0.95		

Project Attributes							
TOOL	Use of software tools	1.24	1.10	1.00	0.91	0.83	
MOD P	Application of software engineering methods	1.24	1.10	1.00	0.91	0.82	
SCED	Required development schedule	1.23	1.08	1.00	1.04	1.10	

Detailed COCOMO model

The detailed COCOMO model incorporates all qualities of both the Basic and Intermediate COCOMO models for each step in the software engineering process.

In each stage of the software development procedure, the Basic or Intermediate model is applied depending on the context of the problem definition to be solved. In other words, in the software development process, the project will be divided into different modules. Each module will be applied to the Basic or Intermedia model, depending on the case. This is a relatively complicated process.

There are six phases in the Detailed COCOMO model as follows

1. Planning and requirements
2. System structure
3. Complete structure
4. Module code and test
5. Integration and test
6. Cost Constructive model

2.2.2 Use Case Points

Object-Oriented Modeling (OOM) gradually became popular since the release of Unified Modeling Language (UML) version 1.1 in 1997 [20]. Since the release of UML 2.0 in 2004 [21], OOM has become a became a near-dominant model in the software industry. Basically, there are two main categories, structure diagrams and behavioural diagrams. In the first category, there are seven diagrams: Class, Component, Deployment, Object, Package, Profile, and Composite Structure diagrams. The second category includes: Use Case, Activity, State Machine, Sequence, Communication, Interaction Overview, and Timing diagrams

Use Case Points (UCP) is a software estimation method based on use-case diagrams. This method is often applied to object-oriented programming and is implemented at a very early stage of software development. Gustav Karner proposed the UCP model in 1993 [22]. This initial model focuses on predicting the total resources required to develop an object-oriented software system in the early stages of software development. The UCP method can be seen as an

extension of the Function Point Analysis method applied to the object-oriented development model.

Basically, the UCP method is illustrated as shown in Fig. 2-4.

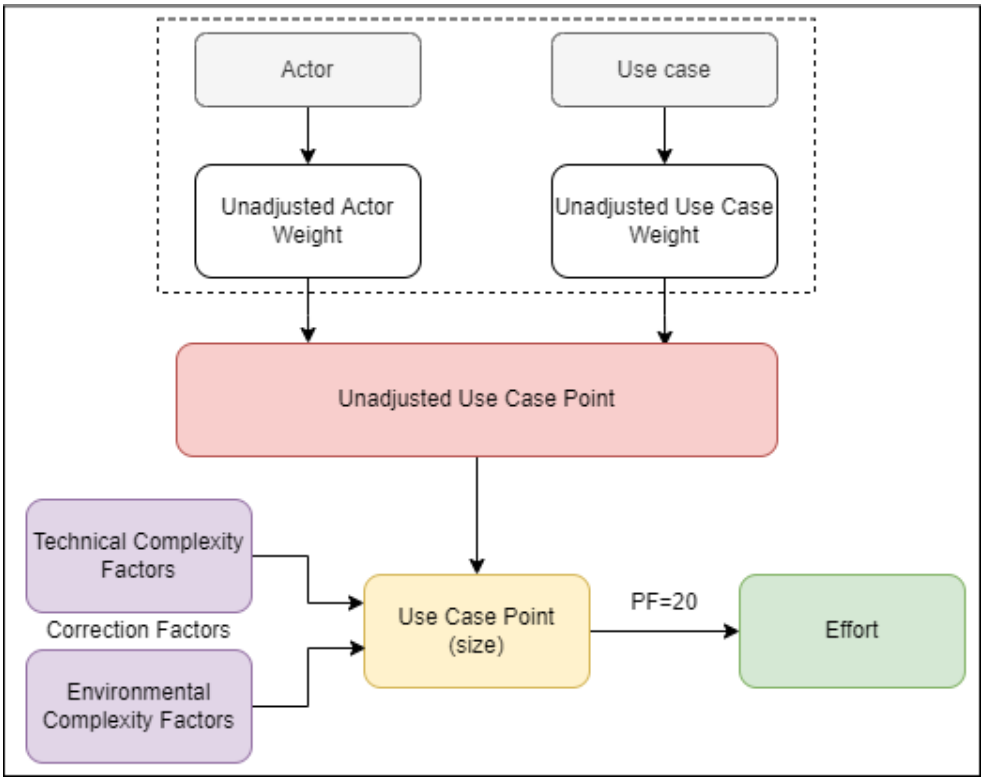


Fig. 2-4. The Use Case Points method's process

Each use case is classified into three categories simple, average, and complex, based on the number of transactions. The corresponding complexity weights for these types are shown in Table 2-4.

Table 2-4. Use case classification and their complexity weights

Use case classification	**Number of transactions**	**Weight**
Simple	(0, 4)	1
Average	[4, 7]	2
Complex	(7, ∞)	3

Similarly, actors are also classified into simple, average, and complex based on the type of actor.

Table 2-5. Actor classifications and their complexity weights

Actor classification	Type	Weight
Simple	The system through an Application Program Interface	1
Average	The system through a protocol	2
Complex	The system through a Graphic User Interface	3

The Unadjusted Actor Weight (UAW) is the sum of all complexity weight of actors. The sum of all complexity weight of use cases creates the Unadjusted Use Case Weight (UUCW). The sum of UAW and UUCW computes the Unadjusted Use Case Point (UUCP).

The correction factors, Technical Complexity Factors (TCFs) and Environment Complexity Factors (ECFs) are used to depict the experience level of the software development team. Each factor has a particular influence on its weight, which we can find in Table 2-6 and Table 2-7.

Table 2-6. Technical Complexity Factors

T_i	Description	Weight (Wt_i)
T_1	Distributed System	2
T_2	Response Adjectives	2
T_3	End-Use Efficiency	1
T_4	Complex Processing	1
T_5	Reusable Code	1
T_6	Easy to install	0.5
T_7	Easy to Use	0.5
T_8	Portability	2
T_9	Easy to Change	1
T_{10}	Concurrency	1
T_{11}	Security Features	1
T_{12}	Access for Third Parties	1
T_{13}	Special Training Facilities	1

Table 2-7. Environmental Complexity Factors

E_i	Description	Weight (We_i)
E_1	Family with RUP	1.5
E_2	Application Experience	0.5
E_3	Object-oriented Experience	1
E_4	Lead Analyst Capability	0.5
E_5	Motivation	1
E_6	Stable Requirements	2
E_7	Part-time Workers	-1
E_8	Difficult Programming Language	2

The correction factors can be computed using Eq. 2.6 and Eq. 2.7.

$$TCF = 0.6 + 0.01 \sum_{i=1}^{13} T_i \times Wt_i \quad (2.6)$$

$$ECF = 1.4 - 0.03 \sum_{i=1}^{8} E_i \times We_i \quad (2.7)$$

where,

- T_i is the value of TCF i,
- Wt_i is the complexity weight of technical factor i,
- E_i is the value of ECF i,
- and We_j is the complexity weight of environmental factor i.

The Use Case Point (UCP) can be calculated using Eq. 2.8.

$$UCP = (UAW + UUCW) \times TCF \times ECF \quad (2.8)$$

For estimating the development effort, the productivity factor (PF) value can be used as the value of 20, as the suggestion by Karner [22].

$$Effort = UCP * PF \quad (2.9)$$

2.2.3 Function Points Analysis

The Function Points Analysis (FPA) method was first introduced by Albrecht in the late 1970s [11]. FPA was proposed as a metric to measure the functionality of the project. The International Function Point User Group (IFPUG) [23] has been the governing body for FPA since 1986, responsible for improving and developing counting rules and other related matters. Since IFPUG was created, the original FPA method has been known as IFPUG's FPA. In this study, the standard FPA method refers to the IFPUG FPA method, simply called FPA. FPA is currently standardized by ISO/IEC 20926:2010 [24]. This standard defines a set of definitions, rules, and steps to apply this standard [25].

The IFPUG counter method can be summarized in the following chart as Fig. 2-5.

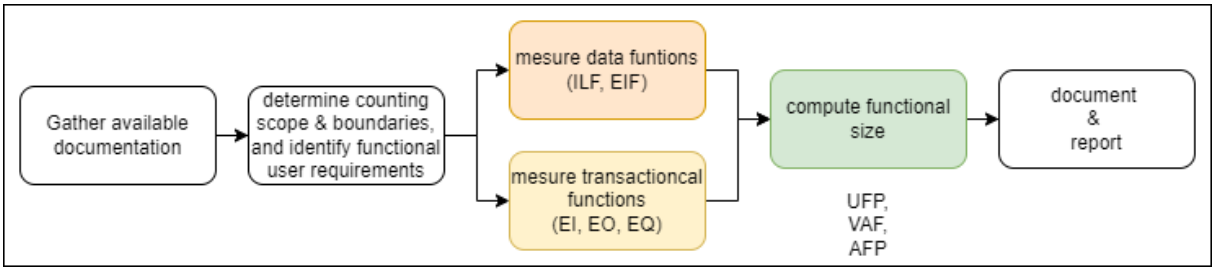


Fig. 2-5. Graphical overview of the FPA counting process

This process determines all standard steps. But in this section, we focus on measuring data functions and transactional functions. After that, we calculate the functional size.

There are five primary types of components, and they're grouped into data functions and transactional functions. Data functions are divided into Internal Logic Files (ILF) and External Interface Files (EIF). The transactional functions are classified each transactional function as an External Input (EI), External Output (EO), or an External Inquiry (EQ), as Fig. 2-6.

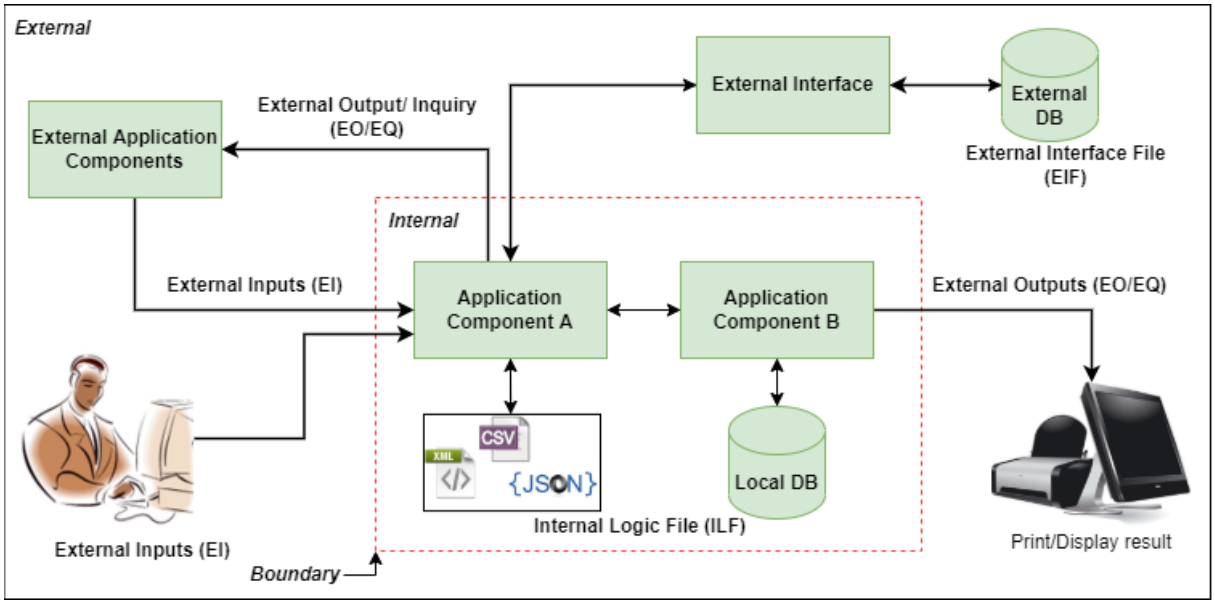


Fig. 2-6. Component types in IFPUG FPA

The External Input, these components are responsible for changes in internal system data. These include, for example, screens, forms, dialogues, or control signal that allows the user or other program to conduct data operations on the system. EI contains all inputs that have a unique format or processing logic.

The External Output is an elementary process that transmits data or control information from the outside of the boundary of the application and processes additional tasks of an external inquiry. The primary purpose of an external output is to display information to a user through processing logic other than the retrieval of data or control information. The processing logic must involve at least one mathematical formula or calculation, create derived data, maintain one or more ILFs, and alter the behaviour of the system.

The External Inquiry is an elementary process that transmits data or control information outside the boundary of the application. The primary purpose of an external inquiry is to present information to a user through the retrieval of data or control information. The processing logic includes no mathematical formula or calculation and does not produce derived data. Neither ILF is maintained during the processing, nor is the behaviour of the system altered.

The Internal Logical File is a recognizable user group of logically involved data or control information preserved within the boundary of the application being measured. The primary objective of an ILF is to maintain data maintained through one or more elementary processes of the application being measured.

The External Interface Files is a user-identifiable of logically related data or control information that the application being measured can be referenced. This data or control must be preserved within the boundary of another application. The primary goal of an EIF is to hold referenced data between elementary processes in the application being measured. It means that the ILF of another application could be the EIF of the application being measured.

Measure data functions

We count the DET (Data Element Type) and Record Element Type (RET) for each data. A RET is a unique, user-recognizable, non-repeated attribute; A RET is a recognizable user sub-group of data element types within a data function. To count DET, we usually count for each unique attribute maintained in/or regained from the data function in the execution of all elementary processes within the counting scope. To count RET, we count for each data function and additional logical sub-groups of DETs such as associative entity, sub-type, and attributive entity.

After counting DETs and RETs, we determine the functional complexity of each data function. Depending on the number of DETs and RETs, we can determine the specified value using the following matrix.

Table 2-8. Data function complexity

		DETs		
		1 - 19	**20 - 50**	**>50**
RETs	**1**	Low	Low	Average
	2 – 5	Low	Average	High
	>5	Average	High	High

Finally, we determine the functional size of each data function. Each data function's function size shall be determined using the type and functional complexity of the tables below.

Table 2-9. Data function weight

		Type	
		ILF	**EIF**
Functional Complexity	**Low**	7	5
	Average	10	7
	High	15	10

Measure transactional functions

Firstly, to identify each elementary process, we compose and decompose the Functional User Requirements into the smallest unit of activity, compose a complete transaction, are self-contained, and make the coating application's business a consistent state. Secondly, to determine unique elementary processes, we usually analyse if the elementary process requires the same set of DETs, File Type References (FTRs), and processing logic. Thirdly, to classify each transactional function as an EI, EO, or an EQ, the majority is its primary intent. The primary intent can be identified based on altering the application's behaviour, maintaining one or more ILFs, or presenting information to the user.

Finally, we determine the function complexity and contribution. In this phase, we identify and count the FTRs and DETs, and then we refer to the following tables to determine each transaction's functional complexity.

Table 2-10. The EI functional complexity

		DETs		
		1 – 4	**5 – 15**	**>15**
FTRs	**0 – 1**	Low	Low	Average
	2	Low	Average	High
	>2	Average	High	High

Table 2-11. The EO and EQ functional complexity

		DETs		
		1 - 5	**6 - 19**	**>19**
FTRs	**0 - 1**	Low	Low	Average
	2 - 3	Low	Average	High
	>3	Average	High	High

Each transactional function's functional size should be determined using the type and functional complexity of the table below.

Table 2-12. Transactional function weight

		Type		
		EI	**EO**	**EQ**
Functional Complexity	**Low**	3	4	3
	Average	4	5	4
	High	6	7	6

For the calculation of Unadjusted Function Point (UFP), set the number of types in groups (multiplier), multiply them by complexity sizes and make the sum of all fields in the table.

$$UFP = \sum_{i=1}^{n} \sum_{j-1}^{m} (multiplier_{ij} \times size_{ij}) \qquad (2.10)$$

where *n* is the number of types, and *m* is the number of complexity groups.

In the next phase, we calculate the Value Adjustment Factor (VAF). This value is based on 14 General System Characteristics (GSCs) that rate the general functionality of the application being measured (see Table 2-13).

Table 2-13. General System Characteristics

Factor	Content
F_1	Data Communications
F_2	Distributed Data Processing
F_3	Performance
F_4	Heavily Used Configuration
F_5	Transaction Rate
F_6	Online Data Entry
F_7	End-User Efficiency
F_8	On-line Update
F_9	Complex Processing
F_{10}	Reusability
F_{11}	Installation Ease
F_{12}	Operational Ease
F_{13}	Multiple Sites
F_{14}	Facilitate Change

Based on the stated user requirements, each factor is rated in terms of its degree of influence on the 0 – 5 scale. Table 2-14 represents the significance of the influence factors rating.

Table 2-14. Influence Factor weights

	System Influence	Rating
1	Not present or no influence	0
2	Incidental influence	1
3	Moderate influence	2
4	Average influence	3
5	Significant influence	4
6	Strong influence throughout	5

Then we can calculate the VAF as the formula below:

$$VAF = 0.65 + 0.01 \times \sum_{i=1}^{14} (F_i \times rating) \qquad (2.11)$$

The VAF adjusts the unadjusted functional size +/-35 % to produce the adjusted functional size. It can vary in range from 0.65 (when all GSCs are low) to 1.35 (when all GSCs are high). In a normal case, it can be chosen in the lowest case.

The number of Adjusted Function Points (AFP) is calculated as:

$$AFP = UFP \times VAF \quad (2.12)$$

2.2.4 MarkII FPA

MarkII FPA [26] was introduced by Symons in 1988 to improve the original FPA method. This method proposed some suggestions to reflect the internal complexity of an application system. During this period, the Metrics Practices Committee of the UK Software Metrics Association (UKSMA) is the design authority of the method [27]. It was also principally designed to measure business information systems. MarkII FPA has been accepted as ISO/IEC 14143 and became an international ISO standard in 2002 [28].

In the MarkII size measurement, the application size is counted as a collection of logical transactions. Each transaction consists of an input, a process, and an output component, as shown in Fig. 2-7.

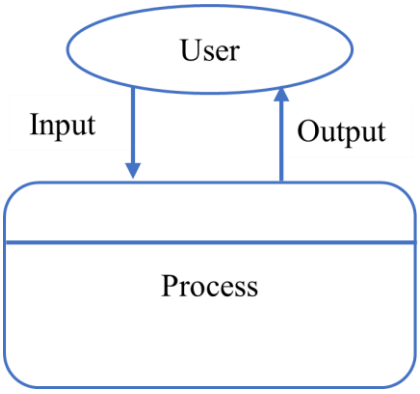


Fig. 2-7. Mark II Functional Size Measurement

The size of the application is the sum of the sizes of logical transactions. Two critical definitions for this method exist Entity (or data entity type) and Data Element Type (DET). The entity is a fundamental thing of relevance to the user about which information was kept [26]. Data Element Type is a distinctive, user recognizable, non-recursive attribute. The number of data element types is used to determine the input and output size of each logical transaction. An Input DET (In-DET) comes from outside the system boundary and changes the state of the system. An Output DET (Out-DET) goes back across the system boundary so a user can see/use it. Both of them are concerned with the formatting and presentation of data.

Essentially, the function points (function points index or FPI) can be obtained by computing each logical transaction size and summing this up for all identified logical transactions. Size can be expressed as follows:

$$Size = W_i * \sum N_i + W_e * \sum N_e + W_o * \sum N_o \quad (2.13)$$

where:

- W_i is the weights for input data elements, the recommended value is 0.58,
- N_i is the number of input data elements,
- W_e is the weight for entities referenced, the recommended value is 1.66,
- N_e is the number of entities referred,
- W_o is the weight for output elements, the recommended value is 0.26,
- N_o is the number of output data elements.

2.2.5 COSMIC

COSMIC, the project was commenced in November 1998, adopted in 2011 as ISO 19761 [29], and has been proposed as a 2nd generation FSM method. COSMIC stands for Common Software Measurement International Consortium. The aim of the COSMIC is to propose a new rule of functional size measures that extract the best features of IFPUG, Mark II, etc. These characteristics are (a) the size of the application should be measured from user requirements; (b) the method should be academically sound and compatible with the modern ways of stating requirements but independent of specific purposes; (c) the measurement should be consistent with ISO 14143 standards.

It proposed a series of innovations: a better fit with both real-time and Management Information Systems environments; it not only applied in identification systems and measurement of multiple software system layers but also in different viewpoints from which the software can be observed and measured. The special problem was it could be estimated the absence of a weighting system.

2.2.6 FiSMA

Finnish Software Measurement Association (FiSMA) Function Size Measurement (FSM) is service-oriented instead of a process-oriented method. So, as Base Functional Components (BFCs) of measurement, services are defined. It defines seven BFC classes. Each BFC class of FiSMA 1.1 further decomposes into several BFC types. After identifying each service, the size of each service was found by applying the rules of the method. Finally, the total functional size was estimated by adding up the sizes of all services.

FiSMA FSM was accepted as an international FSM standard in 2008 [30] and was developed by a working group of FiSMA. It is a general parameterized size measurement method that is proposed to be applied to all types of software. The main difference between FiSMA FSM from other methods is that it's service-oriented rather than process-oriented.

2.2.7 NESMA

Netherlands Software Metrics Association (NESMA) FPA [31] has the same rules as the IFPUG FPA method. ISO accepted it as an international standard in 2005 [32]. NESMA, the user group for function points in the Netherlands, suggests three types of function point counts depending on the degree of detail possible - detailed, estimative, and indicative. The detailed function point count is the IFPUG count. In the estimated function point count, the steps are: (1) determine all functions of all five types ILF, EIF, EI, EO, and EQ; (2) calculate the total unadjusted function point assuming that every data function point is of complexity low, every transaction point is of average complexity.

2.3 Machine-Learning approaches

In recent decades, ML techniques have overgrown and are present in almost all aspects of life. Software estimation is no exception. Many algorithms have been presented and applied in software effort estimation. It can even be seen as an alternative to the other two approaches [7]. These algorithms participate in the estimation process either as a part or as an expert predictor for the entire process.

Some of them are Artificial Neural Networks, Support Vector Machines, Fuzzy Logic, Neuro-Fuzzy, Bayesian Networks, Regression Tree, and Genetic Algorithms. In this thesis, some algorithms will be used and described in section 3.

2.4 Software Estimation Tools in the Software Industry

In the industrial field, software estimation is a matter of great interest. From the proposed studies, software has been developed to carry out these studies. This section introduces some tools that are used in the software industry.

2.4.1 COCOMO II - Constructive Cost Model

This is an implementation tool for Boehm's COCOMO model [19]. It is implemented as a web application and can be accessed from the link: http://softwarecost.org/tools/COCOMO. The Constructive Cost Model using is depicted in Fig. 2-8 and Fig. 2-9.

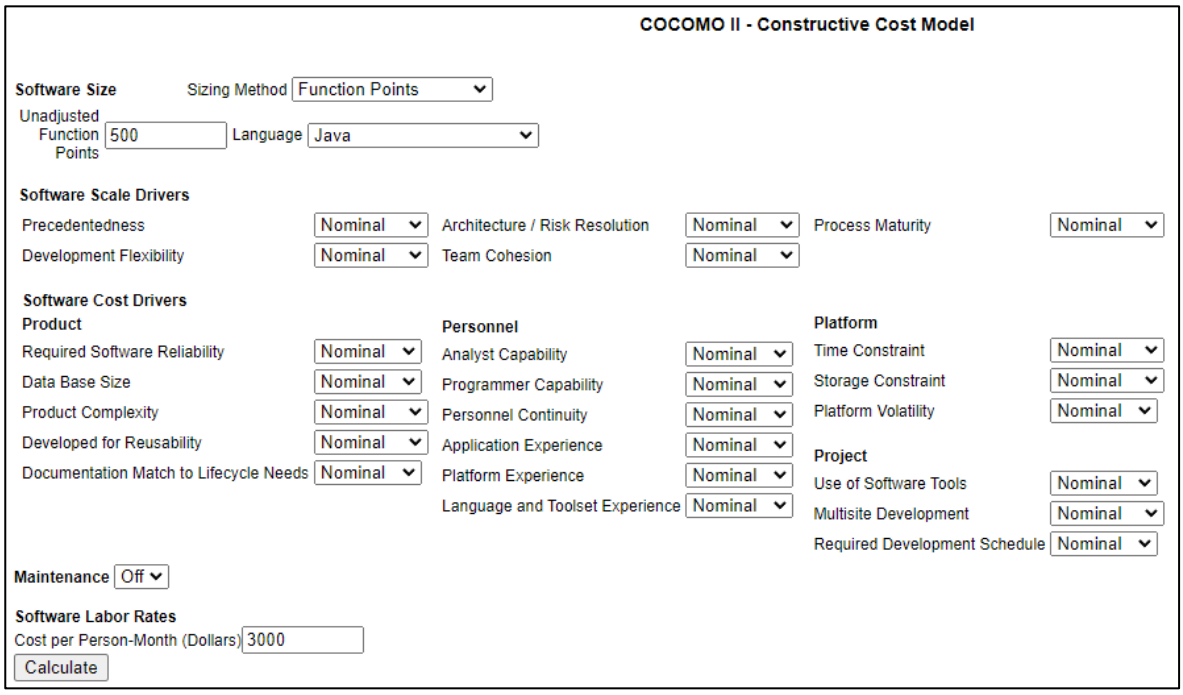


Fig. 2-8. COCOMO II - Constructive Cost Model user interface

Results with 500 UFP, Java programming language, and cost per Person - Month = 3000.

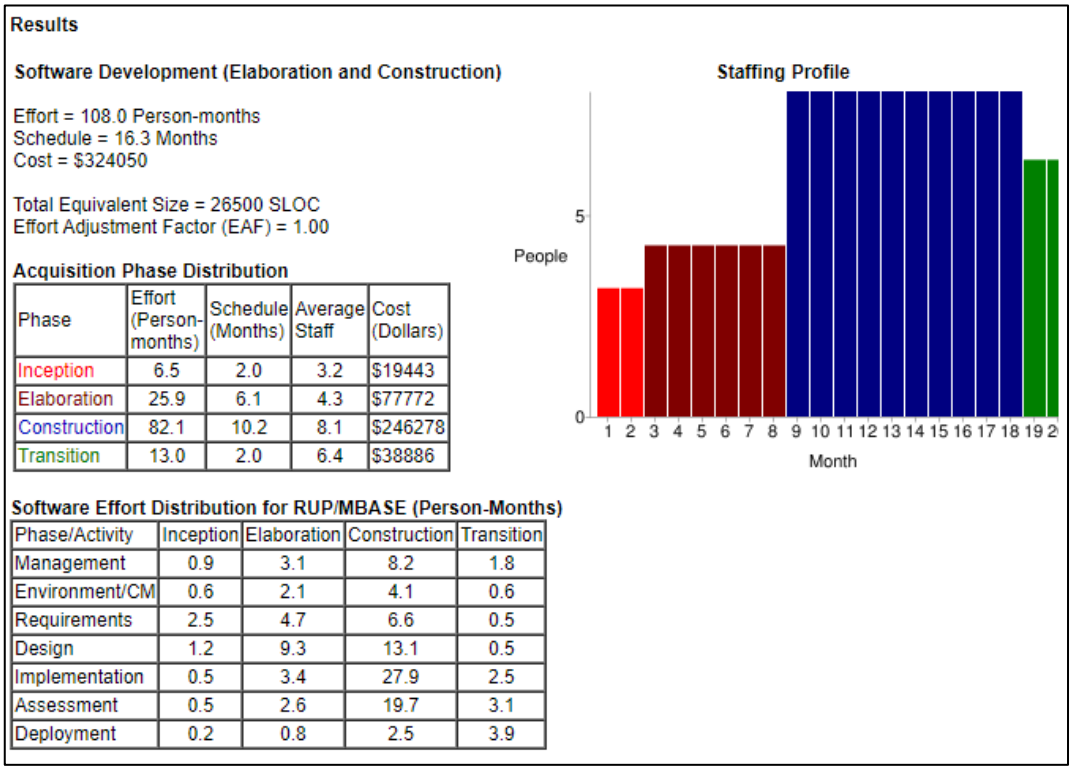


Fig. 2-9. COCOMO II report sample

2.4.2 Construx Estimate Tool

It is an old tool for software estimation and can be found at

http://www.construx.com/Resources/Construx_Estimate_Download.

When using it, the size of the project must be known. No reports for distribution phases. We can estimate in three ways: 1) industry data, 2) cost factor, and 3) historical data. The wizard is easy to use.

The kinds of units and units in this tool should be specified in the following images Fig. 2-10 and Fig. 2-11.

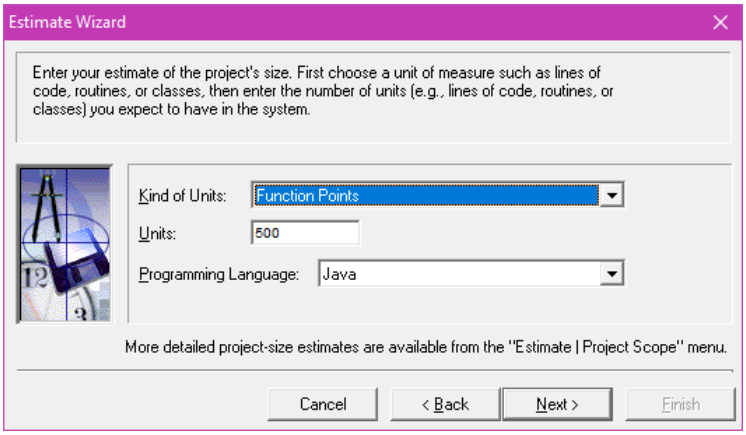


Fig. 2-10. Construx estimate wizard – select features

The result can be displayed as follow

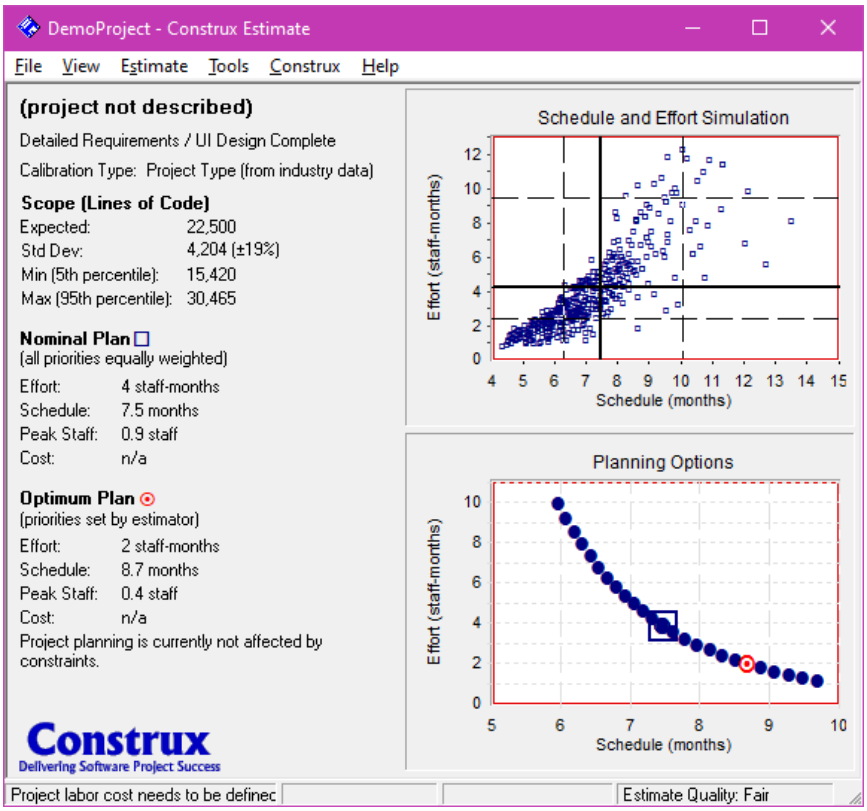


Fig. 2-11. Construx Demo - Results

We can find the report in the report manager.

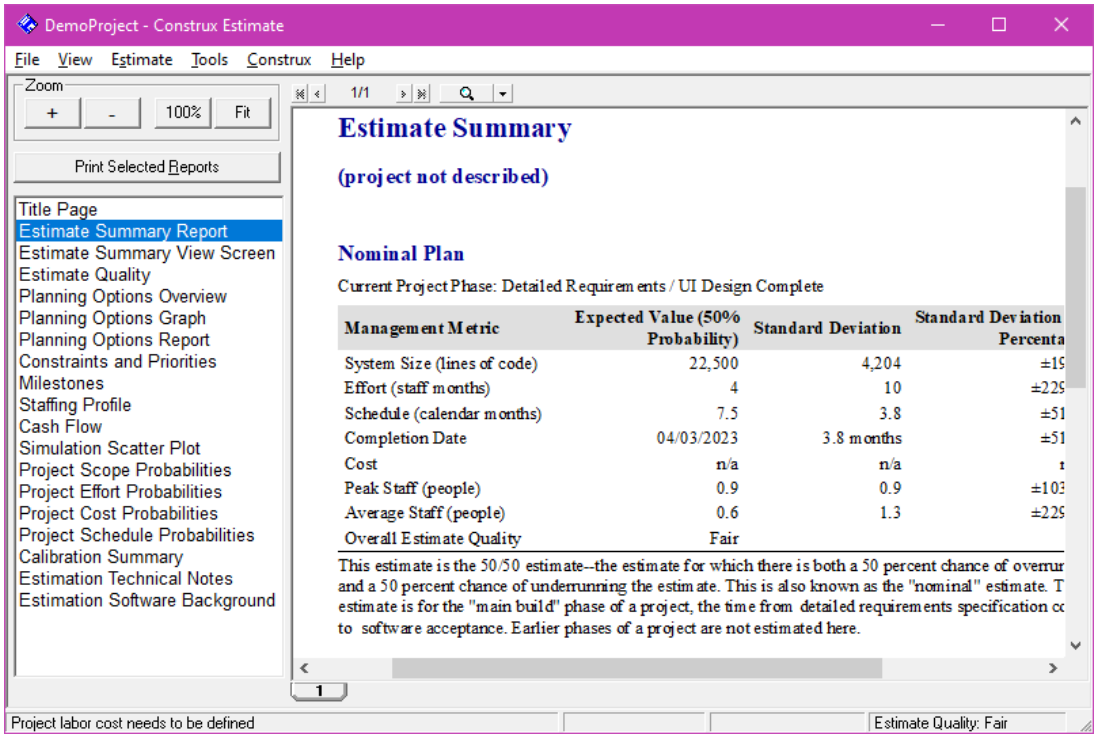


Fig. 2-12. Construx Estimate - Report

2.4.3 SystemStar for COCOMO and COSYSMO Estimation Tools

It is the implementation version of the COCOMO [19] and COSYSMO [33] models. It can be found at http://www.softstarsystems.com/. It is commercial software, and we should pay for use. The version used in this study is a demo version. It gives us a sample of the application of this tool. The use of this tool is shown in the images Fig. 2-13 to Fig. 2-17.

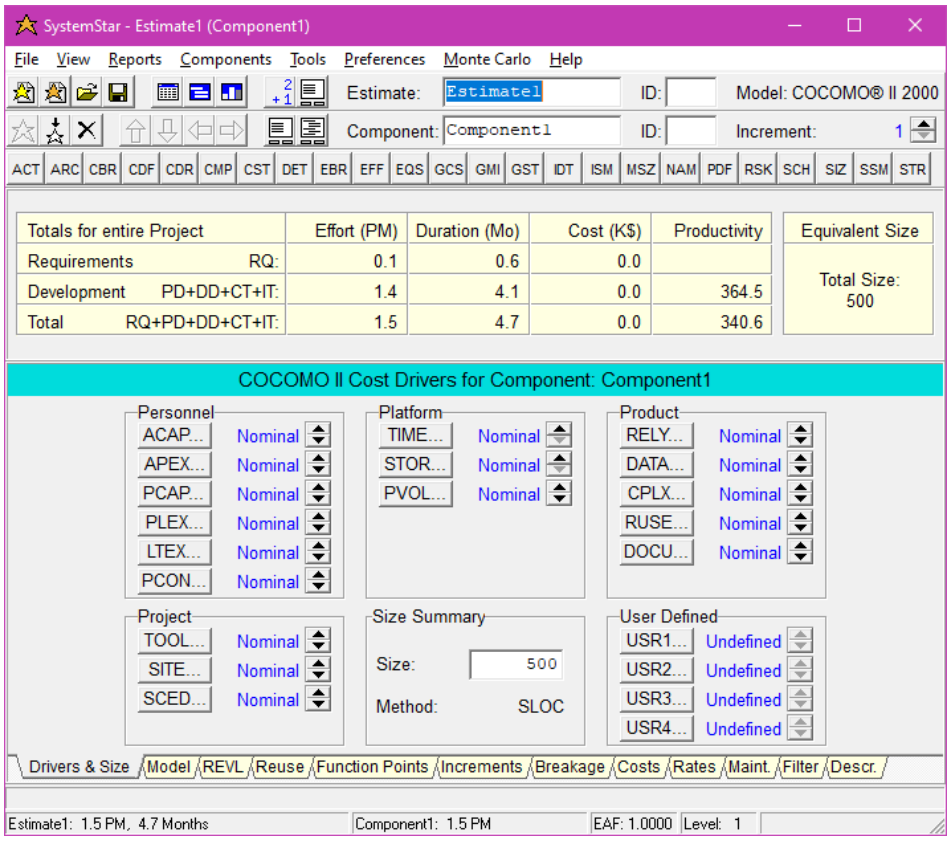


Fig. 2-13. SystemStar sample project - Drive and Size tab

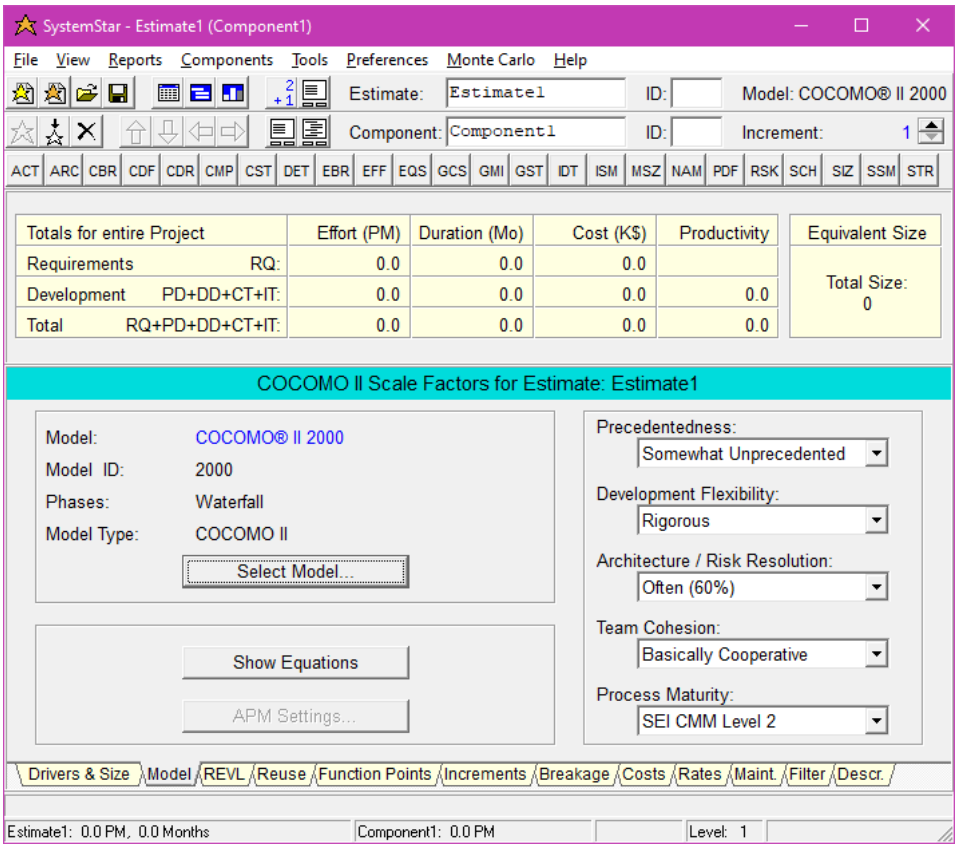
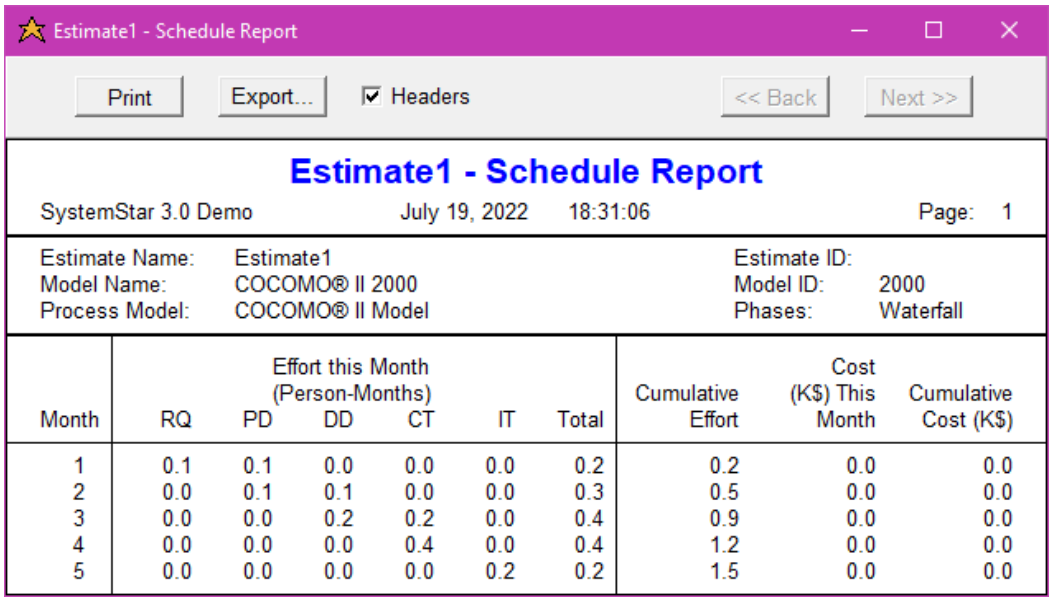


Fig. 2-14. SystemStar sample project - Model Tab



Estimate1 - Schedule Report

Print | Export... | Headers | << Back | Next >>

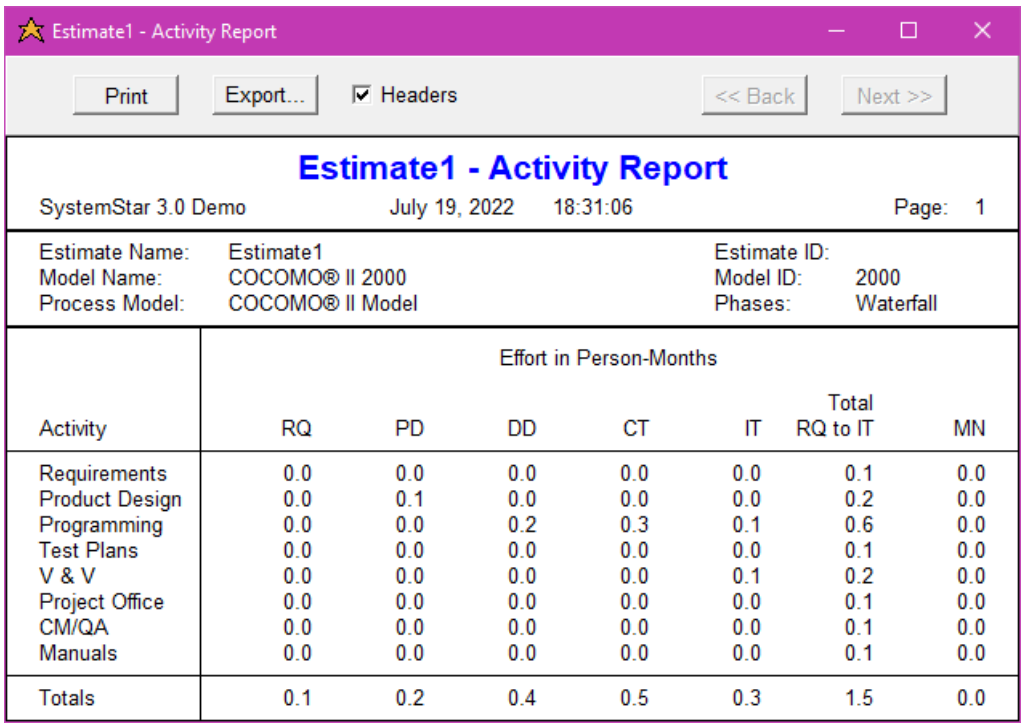
Estimate1 - Schedule Report

SystemStar 3.0 Demo July 19, 2022 18:31:06 Page: 1

Estimate Name: Estimate1
Model Name: COCOMO® II 2000
Process Model: COCOMO® II Model
Estimate ID:
Model ID: 2000
Phases: Waterfall

Month	Effort this Month (Person-Months) RQ	PD	DD	CT	IT	Total	Cumulative Effort	Cost (K$) This Month	Cumulative Cost (K$)
1	0.1	0.1	0.0	0.0	0.0	0.2	0.2	0.0	0.0
2	0.0	0.1	0.1	0.0	0.0	0.3	0.5	0.0	0.0
3	0.0	0.0	0.2	0.2	0.0	0.4	0.9	0.0	0.0
4	0.0	0.0	0.0	0.4	0.0	0.4	1.2	0.0	0.0
5	0.0	0.0	0.0	0.0	0.2	0.2	1.5	0.0	0.0

Fig. 2-15. SystemStar sample project - Schedule Report



Estimate1 - Activity Report

Print | Export... | Headers | << Back | Next >>

Estimate1 - Activity Report

SystemStar 3.0 Demo July 19, 2022 18:31:06 Page: 1

Estimate Name: Estimate1
Model Name: COCOMO® II 2000
Process Model: COCOMO® II Model
Estimate ID:
Model ID: 2000
Phases: Waterfall

Activity	Effort in Person-Months RQ	PD	DD	CT	IT	Total RQ to IT	MN
Requirements	0.0	0.0	0.0	0.0	0.0	0.1	0.0
Product Design	0.0	0.1	0.0	0.0	0.0	0.2	0.0
Programming	0.0	0.0	0.2	0.3	0.1	0.6	0.0
Test Plans	0.0	0.0	0.0	0.0	0.0	0.1	0.0
V & V	0.0	0.0	0.0	0.0	0.1	0.2	0.0
Project Office	0.0	0.0	0.0	0.0	0.0	0.1	0.0
CM/QA	0.0	0.0	0.0	0.0	0.0	0.1	0.0
Manuals	0.0	0.0	0.0	0.0	0.0	0.1	0.0
Totals	0.1	0.2	0.4	0.5	0.3	1.5	0.0

Fig. 2-16. SystemStar sample project - Activity Report

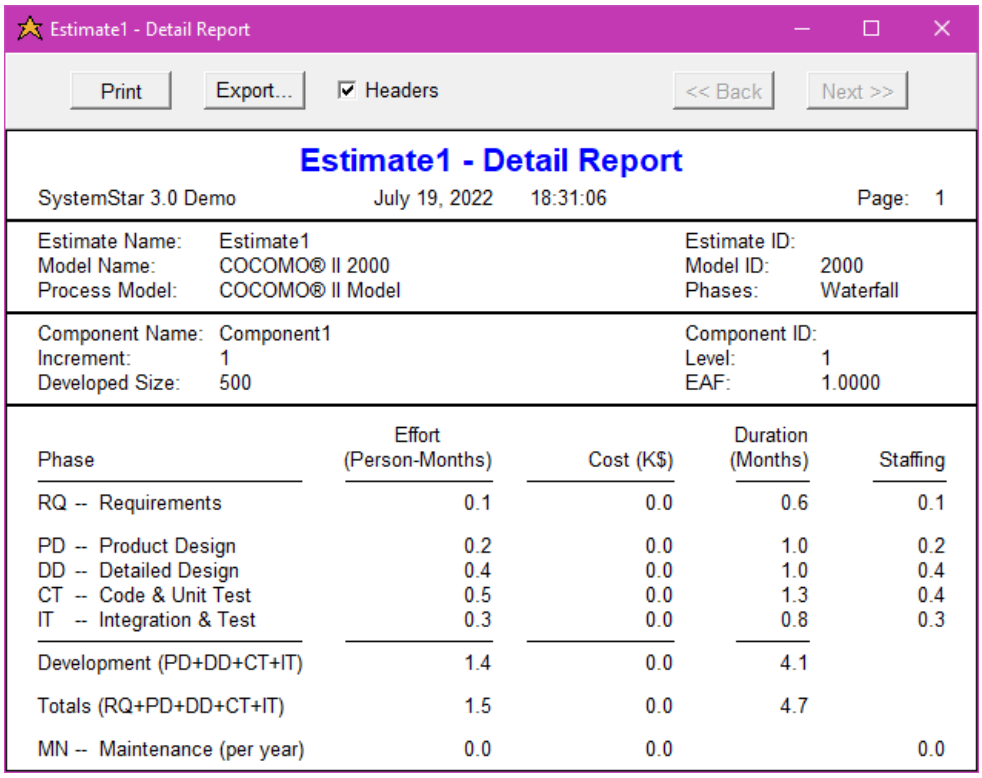


Fig. 2-17. SystemStar sample project – Detail report

2.4.4 Function Point Modeler

Function Point Modeler (FPM) is a tool for Function Point Analysis to measure Software. It is an IFPUG Counting Practice Manual (CPM) conform tool (CPM 4.2 and 4.x). It was designed and implemented by Certified Function Point Specialists to satisfy all requirements of FPA counting practice specialists [34].

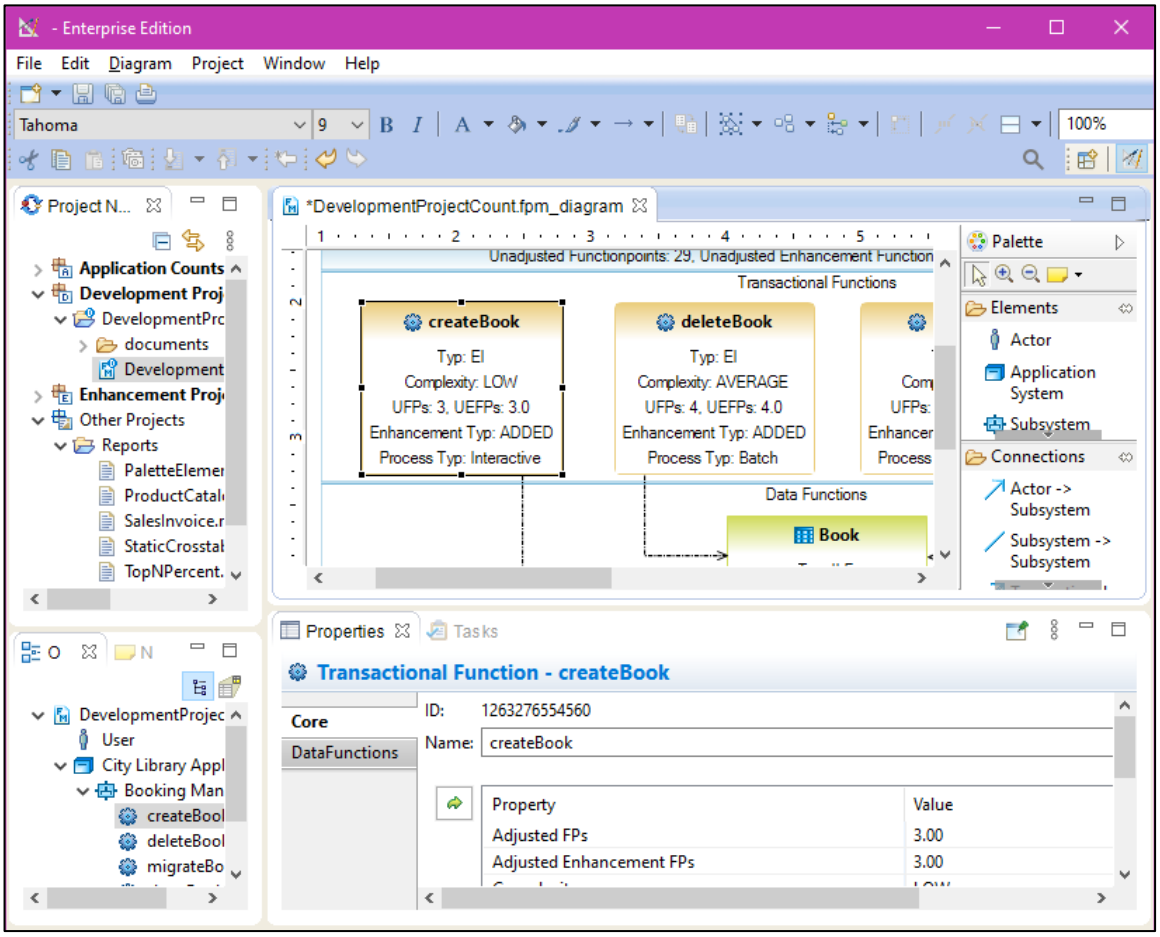


Fig. 2-18. FPM user interface

FPM was built based on the Eclipse Graphical Modeling framework, an open-source project. Basically, FPM has a very easy-to-use graphic user interface. The left with project navigation and the outline panel, the right with the palette panel for drag-n-drop operations. At the bottom is the properties panel that can display all characteristics of the selected object. We can imagine this tool in Fig. 2-18.

FPM is a model-driven architecture tool that uses XMI and other formats for exchange with other applications (Fig. 2-20). FPA can be used for development, enhancement, and application project count (Fig. 2-19).

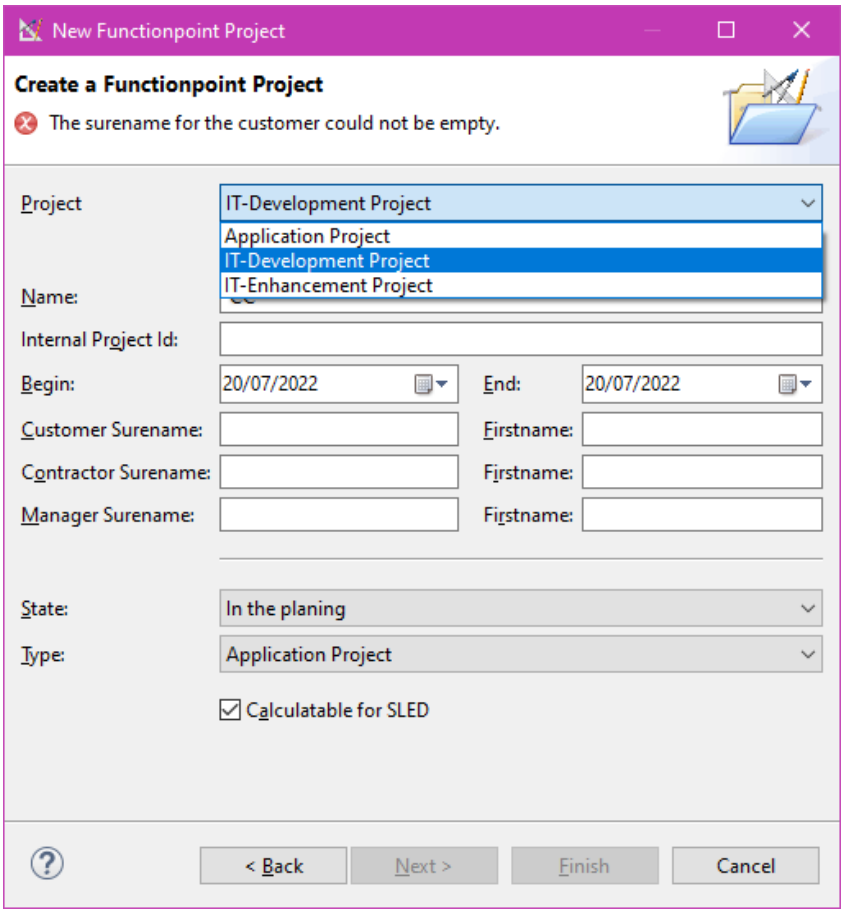


Fig. 2-19. FPM project types

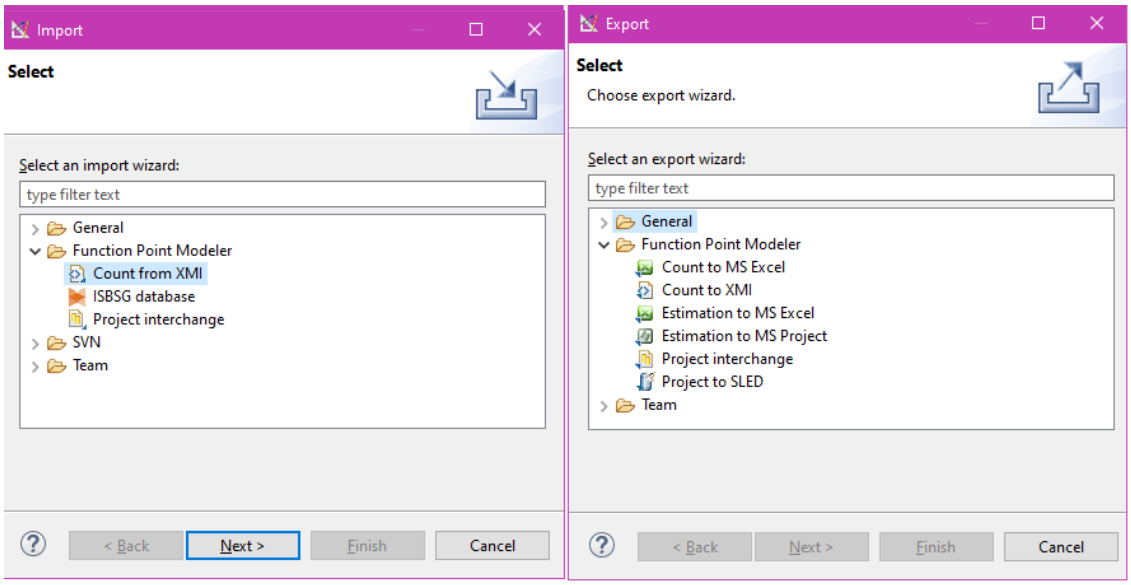


Fig. 2-20. FPM as a tool that uses model-driven architecture

An example of this tool can be found when we create a new Function point project. The Development project count in this example can be seen in Fig. 2-21.

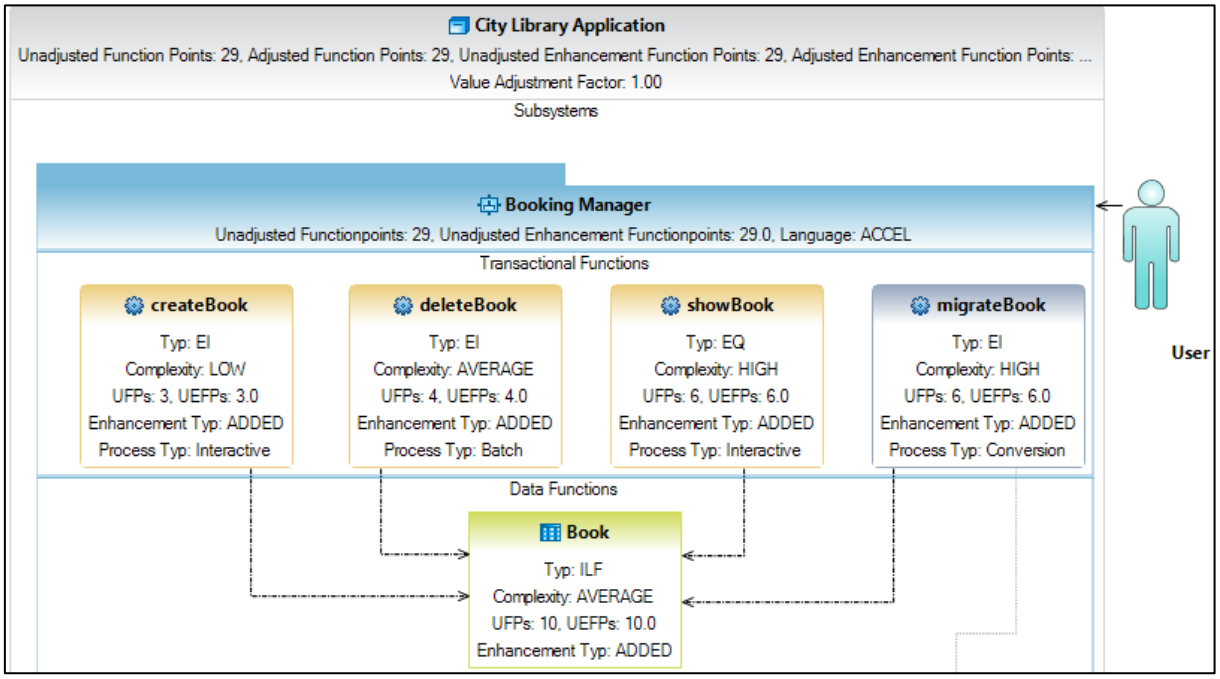


Fig. 2-21. FPM - Development project count example

An ILF component was known as Book with the average complexity. Four other components related to a component are createBook (an EI with low complexity), deleteBook (EI with average complexity), showBook (EQ with high complexity), and migrateBook with high complexity).

We can change the property of these components easily by using the properties palette in Fig. 2-22.

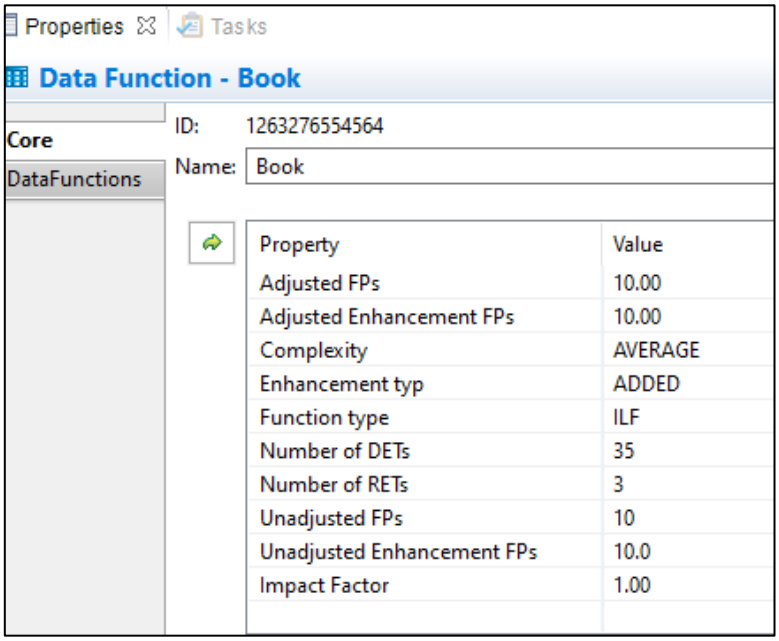


Fig. 2-22. FPM - Property palette

2.5 Summary

The above is an introduction to some techniques for software estimation. In general, these techniques are divided into three main groups as presented. However, this categorization is only relatively. In fact, within a method belonging to an approach, it is possible to use part or whole of a method belonging to another approach. Besides, each approach has its points suitable for different contexts in the software development process. Choosing the suitable approach is the job of the software management team.

In addition, in the process of conducting the estimation of a development software project, we can use software tools for this process. Section 2.4 introduced some typical software estimation tools. Selecting a software estimation tool or another common tool such as a spreadsheet or manual is also the project manager's decision.

3. MACHINE LEARNING ALGORITHMS USED

Nowadays, there are many machine learning algorithms developed and applied. Each algorithm has specific strengths and application areas. This section will briefly describe some algorithms used in this study. These algorithms are divided into two groups: 1) group of algorithms related to clustering and 2) group of machine learning algorithms related to the calibration of the complexity weight system and ensemble algorithms.

3.1 Clustering algorithms

3.1.1 Balanced Iterative Reducing and Clustering using Hierarchies

The Balanced Iterative Reducing and Clustering Hierarchies (BIRCH) Clustering algorithm [35] is a distance-based hierarchical clustering method. There are two main concepts of this algorithm, Clustering Feature (CF) and Clustering Feature Tree (CF-Tree).

Given N d-dimensional data samples in a specific cluster. These data samples can be represented as $\{x_i\}, i = 1,2 \dots N$. The centroid x_0 and the radius R of this cluster can be defined as

$$x_0 = \frac{\sum_{i=1}^{N} x_i}{N} \qquad R = \sqrt{\frac{\sum_{i=1}^{N}(x_i - x_0)^2}{N}} \qquad (3.1)$$

R can be seen as the mean distance from member samples to the centroid. The CF can be defined as a three-dimensional vector as $CF = (N, LS, SS)$, where N is the number of data samples in the cluster, LS is the linear sum of N samples, and SS is the square sum of N data samples.

CF Tree is a height-balanced tree with two parameters, B and T, where B is the branching factor, and T is the cluster radius threshold. The ranching factor can be B for the non-leaf node or L for the leaf node. The tree is created sequentially by following the closest CF down to the leaf nodes. When a sample adding to CF, it must satisfy T; otherwise, it should be a new leaf node.

3.1.2 Fuzzy C-Mean

The Fuzzy C-Means (FCM method) is a fuzzy clustering algorithm based on the cluster's distance measure of data objects. It was introduced by Dunn [36] and improved by Bezdek [37].

Given a finite set of data samples $X = \{x_i, i \in \{1, \dots, N\}\}$, and a set of clusters $C = \{c_k, k \in \{1, \dots, C\}\}$. Call w_{ij} is the degree of membership of the x_i sample in the c_j cluster, we have:

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{2/(m-1)}} \tag{3.2}$$

This feature represents the probability that the sample x_i belongs to cluster c_j or not (closer to 1 means more confidence). The FCM aims to minimize the cost function given by

$$P = \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij}^{m} \|x_i - c_j\|^2 \tag{3.3}$$

where n is the number of data samples, c is the number of clusters, x_i is the i^{th} sample, c_j is the j^{th} cluster centre, and m is the degree of fuzziness $m \in (1, \infty)$.

Starting with a specific number of clusters c and an initial prediction for each cluster centre $c_{j, j = 1, \dots c}$, FCM will converge to a solution for c_j that represents either a local minimum or a saddle point cost function [37].

3.1.3 Gaussian Mixture Model

Gaussian Mixture Model (abbreviated GMM) is a clustering model belonging to the class of unsupervised mathematical problems where the probability distribution of each cluster is assumed to be a multidimensional Gaussian distribution. The model is called a Mixture because the probability of each data point depends not only on a single Gaussian distribution but on a combination of many different Gaussian distributions from each cluster.

The Gaussian Mixture Model [38], the most popular data clustering method as a linear combination of distinct Gaussian components, is a parametric probability density function represented as a weighted sum of Gaussian component densities [39]. GMM models are used to represent Normally Distributed subpopulations within an overall population.

Give a set $\{\mu_i, i = 1..N\}$ where $\mu_i$is the value of i^{th}given set, N is the number of the given set. GMM assumes a mixture model consisting of m Gaussian density components with the parameters $\theta_k = \{u_k, \Sigma_k\}$ in the k^{th} component. The probability density of μ_i is formulated by

$$p(\mu_i|\pi, \theta) = \sum_{k=1}^{m} \pi_k p(\mu_i|\theta_k) \tag{3.4}$$

where $\theta = \theta_1 .. \theta_m$ is all component's parameters, and π_k is the k^{th} component's missing weight, with $\pi_k \geq 0$ and $\sum_{k=1}^{m} \pi_k = 1$(sum of probabilities normalized to 1). The k^{th}Gaussian is denoted by

$$p(\mu_i|\theta_k) = \frac{1}{\sqrt{(2\pi)|\Sigma_k|}} exp\left(-\frac{(\mu_i - u_k)^T \sum_k^{-1} (\mu_i - u_k)}{2}\right) \quad (3.5)$$

where u_k is the mean and Σ_k is the covariance matrix.

The parameters $\{\theta, \pi\}$ can be estimated by utilizing the Expectation-Maximization technique to maximize the likelihood function in terms of the following:

$$\omega_i^{k(t)} = \frac{\pi_i^{(t)} p\left(\mu_i \middle| \theta_k^{(t)}\right)}{\sum_{j=1}^{m} \pi_j^{(t)} p\left(\mu_i \middle| \theta_j^{(t)}\right)}, \quad (3.6)$$

$$\pi_k^{(t+1)} = \frac{\sum_{i=1}^{N} \omega_i^{k(t)}}{N}, \quad (3.7)$$

$$u_k^{(t+1)} = \frac{\sum_{i=1}^{N} \omega_i^{k(t)} \mu_i}{\sum_{i=1}^{N} \omega_i^{k(t)}}, \quad (3.8)$$

$$\sum_k^{(t+1)} = \frac{\sum_{i=1}^{N} \omega_i^{k(t)} \left(\mu_i - u_k^{(t+1)}\right)\left(\mu_i - u_k^{(t+1)}\right)^T}{\sum_{i=1}^{N} \omega_i^{k(t)}}. \quad (3.9)$$

In this study, GMM is the essential clustering model for clustering the ISBSG dataset and then performing the cluster's effort estimation process.

3.1.4 K-means

Clustering is a popular machine learning technique for analysing data. Clustering is the process of classifying data points into specific groups. In which data points in the same group must have similar properties and vice versa, points in different groups must have dis-similar features. Distance measurement to evaluate the similarity between data points.

K-means clustering [40] is an unsupervised machine learning algorithm used to cluster given objects into k clusters, where k is pre-specified. In clustering K-means, each cluster is represented by its centre (centroid) corresponding to the mean of the points assigned to the cluster [41]. We can summarize the k-means algorithm as follows:

1. Specify the number of clusters k.
2. Randomly select k points from the central data set (centroids) for the k clusters.

3. Calculate the distance between the points to the centre.
4. Group the objects into the nearest group.
5. Redefine the new centre for the groups by calculating the mean for the data points in the respective clusters.
6. Repeat step 3 until there is no group change of data points.

In this study, the k-means algorithm was implemented by *sklearn.cluster.KMeans* package with the number of clusters is obtained from the Elbow method (next section), and other parameters are the default. The distance between the points and the centre was calculated using the Euclidean Distance algorithm. When a new project requires effort estimation, the Euclid distance will be calculated, and the cluster to which the new project belongs will be determined. For effort estimate, the selected cluster's correspondence model will be applied.

3.1.5 Mean-Shift

The Mean-Shift algorithm is a widely used algorithm in image processing and computer vision [42, 43]. It is a density-based clustering algorithm. The Mean-Shift algorithm is a clustering algorithm based on density. The main idea is to compute the mean distance between a point and a vector within a certain radius (the Mean-Shift vector) and compute the direction of the moving of the point in the subsequent step. When the point is no longer moving, it forms a class cluster with the surrounding points and then calculates the distance between the class cluster and the previous class cluster. If the distance is less than the threshold, it will be merged into the same class cluster; otherwise, it will form a class cluster. This process happens Until all data points are selected [44].

For n samples in a given space $x_i, i = 1 \dots n$, k is the number of points in the range S_h, the Mean-Shift vector was defined as:

$$M_h(x) = \frac{1}{k} \sum x_i \in S_h \, (x_i - x) \tag{3.10}$$

S_h is the high-dimensional sphere region with centre x and radius h.

$$S_h = (x_i | (x_i - x)(x_i - x)^T < h^2) \tag{3.11}$$

In this study, the Mean-Shift algorithm is implemented by *sklearn.cluster*. The bandwidth in this case with quantile value is 0.2, n-sample value is 300, and random state is 0.

3.1.6 Spectral clustering

Spectral clustering (SC) is a well-known unsupervised clustering algorithm with an ancestor from graph theory [45] [46]. In this context, data points are nodes, and the similarity of two nodes is the edge between them. This means the

SC will group objects belonging to irregular form clusters based on connectivity. The non-graph data can be clustered using the SC technique as well.

Given n objects, we can build a graph representation of the objects G (V, E, S), where V represents the graph vertices (nodes), E presents the link (edge) between vertices, and S presents the weights of edges (Similarity Matrix or Affinity Matrix). In the similarity matrix S, with two objects i and j, the element S_{ij} represents their similarity. S_{ij} close to 1 means that i and j are similar. In contrast, if S_{ij} is close to 0, it means that I and j are different.

The spectral clustering algorithm's essential idea is to use the similarity matrix to achieve dimensionality reduction and then cluster data objects.

First, the Similarity matrix was computed with each value of the matrix specified by ()

$$S_{ij} = \begin{cases} exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right) & x_i \; and \; x_j \; are \; neighbors \\ 0 & otherwise \end{cases} \quad (3.12)$$

where σ is the scaling parameter (control the spread of neighbours).

Next, the Laplacian Matrix is computed

$$L = D - S \quad (3.13)$$

where D is the diagonal matrix, $D_{ii} = \sum_j^n S_{ij}$, $S = (S_{ij})$

The next step is finding the k largest eigenvector of L presented by

$$X = \{x_1, \dots, x_k\} \quad (3.14)$$

Then calculate the normalized matrix Y_{ij} with

$$Y_{ij} = \frac{X_{ij}}{\left(\sum_j {X_{ij}}^2\right)^{1/2}} \quad (3.15)$$

The final step is finding the k clusters. SC uses these eigenvectors as a feature. Any known algorithm can perform the clustering of features. In this paper, the k-means algorithm is used.

3.2 Other Machine Learning algorithms

3.2.1 Linear Regression

Regression is a statistical technique for determining the relationship between two or more variables. The dependent and independent variables have a relationship that can be recognized. Probability distribution functions, as shown in Eq. 1, can be used to represent it.

$$Y = f(X, \beta) \tag{3.16}$$

Where Y is a dependent variable, X is an independent variable, and β is the coefficient. The variable dependency can be either univariate or multivariate regression. Univariate regression identifies the dependency among a single variable, while multivariate regression identifies the dependence among several variables simultaneously [47].

$$\mathrm{Y} = \beta_0 + \beta_1 \mathrm{X}_1 + .. + \beta_\mathrm{n} \mathrm{X}_\mathrm{n} + \varepsilon \tag{3.17}$$

where Y is the predicted dependent variable, n is the number of variables, β_0 is the intercept, X_i are the independent variables and $\beta_i \; i \in 1, n$ are called Partial Regression Coefficients, and ε is the error residual. In this study, multivariate regression was used.

3.2.2 Multilayer Perceptron

Artificial Neural Networks (ANN) are computing system that simulates the biological neural networks that constitute human brains. One of the most abilities of this system is "learning." It means that we can train it by providing data, and it can be performed tasks by considering these data [48].

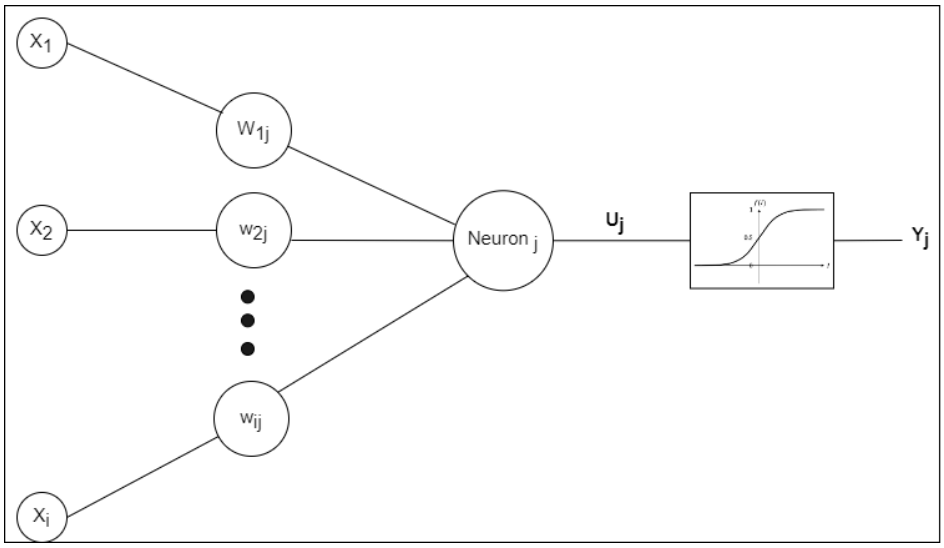


Fig. 3-1. ANN model

Each input that is put into a neural was modified by multiplying each input by the weight for that connection. These weighted inputs are then summed by the neuron and, with reference to a threshold value, will determine the neuron output. In Fig. 3-1, these are marked as w_{1j} through to w_{ij} for all the inputs. These weighted inputs are then summed by the neuron and, with reference to a threshold value, will determine the neuron output.

The output is described by two sets of equations. The first one is the combined operation of the neuron yielding U_j for the jth neuron, is

$$U_j = \sum (x_i \times w_{ij}) - t_j \quad (3.18)$$

where U_j is biased and adjusted by a previously established threshold value, t_j, and is then subjected to the activation or threshold function. Where the activation function is sigmoidal, the equation is as follows:

$$Y_j = (1 + e^{-U_j})^{-1} \quad (3.19)$$

This equation implements the firing of the neuron. Numerous activation functions have been used, such as gaussian, linear, stepwise, or the most commonly used one, which is the sigmoid function. This output, Y_j, is the input to the successive layer or the response of the network if this neuron is in the last layer.

A Multilayer Perceptron (MLP) is a fully connected feedforward ANN class. An MLP has an input layer, a hidden layer, and an output layer, which together make up at least three layers of nodes. Neurons serve as input points and represent input variables in the first layer. The data processing and output are handled by other layers. As a result, a neural network may be thought of as a sophisticated computation function that expands the solution by passing data through the network to the output layer [49]. Except for the input nodes, each node is a neuron that uses a nonlinear activation function.

ANNs have the ability to model any complex non-linear associations and are capable of approximating any measurable function. Many network architectures have been developed for applying various applications. Finnie et al. [50] used the back-propagation networks for software effort estimation. The networks are built with the inputs are function points, GSCs, and the programming environment; the output is the estimated development effort, as Fig. 3-2.

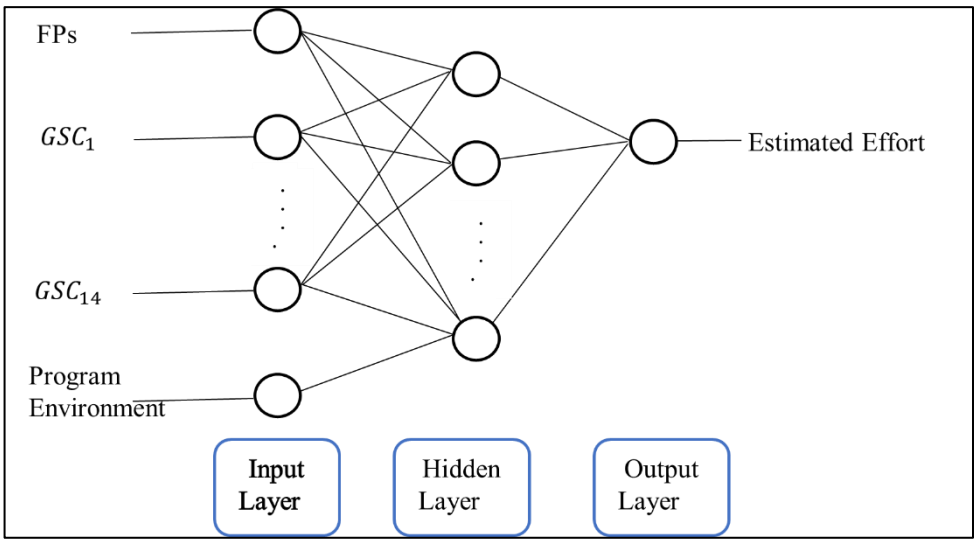


Fig. 3-2. ANN Architecture for Software Development Effort Estimation

3.2.3 Support Vector Machine

The Support Vector Machine (SVM) is a robust learning algorithm based on advances in statistical learning theory [51]. SVMs are learning systems that employ a hypothesis space of linear functions in a high-dimensional space and are trained using an optimization theory-based learning algorithm that incorporates a learning bias [52]. SVMs have recently evolved into one of the most popular tools for machine learning and data mining and can perform both classification and regression. SVM employs a linear model to implement nonlinear class boundaries by nonlinearly mapping input vectors into a high-dimensional feature space using kernels. The training examples closest to the maximum margin hyperplane are called support vectors. All other training examples are irrelevant for defining the binary class boundaries. The support vectors are then used to construct an optimal linear separation hyperplane (in the case of pattern recognition) or a linear regression function (in the case of regression) in this feature space. The support vectors are conventionally determined by solving a quadratic programming problem.

For regression tasks, Vapnik proposed an SVM called ε-support vector regression (ε-SVR), which performs prediction tasks from the ε-insensitive loss function. A maximum tolerance parameter for errors in the training phase is used so that the errors under the ε-insensitive loss function are not penalized [53]. The ε-parameter can also be considered as a tolerance measure with respect to the actual values, i.e., the SVR tolerates an absolute value of at most ε for the difference between the actual value and the predicted value.

In practice, the SVR algorithm can be linear or non-linear using the respective kernel function. In this study, the linear kernel was used.

3.2.4 Bayesian Ridge Regression

The full Bayesian regression inference using the Markov Chain Monte Carlo algorithm was used to construct the models [54]. The Bayesian modelling framework has been celebrated for its capability to deal with the hierarchical data structure. Bayesian regression techniques can be used to comprise regularization parameters in the prediction procedure. The regularization parameter is tailored to the data at hand rather than being set in complex meaning. This can be accomplished by introducing uninformative priors over the model's hyperparameters. The l_2 regularization used in Ridge regression and classification is equivalent to finding a maximum a posteriori estimation under a Gaussian primary over the coefficients with precision. Instead of manually specifying lambda, it is possible to treat it as a random variable to be estimated from the data [55]. To attain a fully probabilistic model, the output y is assumed to be Gaussian distributed around $X\omega$:

$$p(y|X,\omega,\alpha) = \aleph(y|X\omega,\alpha) \tag{3.20}$$

where $\propto$ is likewise treated as a random variable to be estimated from the data.

Bayesian Ridge Regression (BRR) is a probabilistic approach that employs Bayesian inference to construct a regression model. It combines prior knowledge about the parameters (the coefficient of software features) with the observed training data to obtain the posterior distribution of the parameters [55]. The initial for the coefficient ω is specified by a spherical Gaussian:

$$p(\omega|\lambda) = \aleph\left(\omega\middle|0,\lambda^{-1}\mathbf{I}_p\right) \tag{3.21}$$

The initials over α and λ are chosen to be gamma distributions [56], the conjugate prior to the precision of the Gaussian.

3.2.5 LASSO

The Least Absolute Shrinkage and Selection Operator (LASSO – for short, LAS) regression model was proposed by Tibshirani [57] and is an innovative variable selection method for regression by minimizing the residual sum of squares under the condition that the sum of absolute values of coefficients is less than a constant, and it is a well-known sparse regression method that regulates the parameter beta under the sparse assumption. It was first explained in terms of least squares. The basic framework is summarized as follows: Considering a sample consisting of N cases, each of which consists of p covariates and a single outcome. Supposing yi is the response variable and $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$ is the covariate vector for the i^{th} case, $\boldsymbol{\beta} = (\beta_1, \beta_1, \ldots, \beta_p)^T$, so the objective of LASSO is to solve the optimization problem:

$$\arg \min_{\beta_0, \boldsymbol{\beta} \in \mathcal{R}^p} \frac{1}{N} \sum_{i=1}^{N} (y_i - \beta_0 - x_i^T \boldsymbol{\beta})^2 \tag{3.22}$$

$$s.t. \sum_{j=1}^{p} |\beta_j| \leqslant t \tag{3.23}$$

where $t \geq 0$ is a pre-specified free parameter that determines the amount of regularization. If t is large, all the coefficients are almost zero. For smaller values of t, the LASSO shrinks some of the estimated coefficients equal to zero.

Suppose that X represents the $N \times p$ covariates matrix, N is the number of samples, p is the number of covariates, and y represents a response vector as excepted output. Formula (1) can be written more compactly as:

$$\arg\min_{\beta_0,\boldsymbol{\beta}\in\mathcal{R}^p} \frac{1}{N}\|y-\beta_0 I - X\boldsymbol{\beta}^T\|_2^2 \tag{3.24}$$

$$s.t.\ \|\boldsymbol{\beta}\|_1 \leqslant t \tag{3.25}$$

Where $\|Z\|_p = (\sum_{i=1}^{N}|Z_i|^p)^{1/p}$ is the standard l^p norm. Since $\hat{\beta} = \bar{y} - \bar{x}^T\beta$, so that

$$y_i - \hat{\beta}_0 - x_i^T\boldsymbol{\beta} = y_i - (\bar{y} - \bar{x}^T\boldsymbol{\beta}) - x_i^T\boldsymbol{\beta} \tag{3.26}$$

$$= (y_i - \bar{y}) - (x_i - \bar{x})^T\boldsymbol{\beta},$$

The formula (2) can be rewritten as

$$\arg\min_{\boldsymbol{\beta}\in\mathcal{R}^p}\left\{\frac{1}{N}\|y - X\boldsymbol{\beta}^T\|_2^2\right\} \tag{3.27}$$

$$s.t.\ \|\boldsymbol{\beta}\|_1 \leqslant t$$

Lagrangian form of the estimator $\hat{\beta}$ [58]can be represented as the following:

$$L(\boldsymbol{\beta},\lambda) = \min_{\boldsymbol{\beta}\in\mathcal{R}^p}\left\{\frac{1}{N}\|y - X\boldsymbol{\beta}^T\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1\right\} \tag{3.28}$$

where the tuning parameter $\lambda \geqslant 0$ is used to balance the empirical error and the sparsity of the model parameter, and where the exact relationship between t and λ is data dependent.

3.2.6 Voting Regressor

The voting regressor [59] is based on integrating various machine learning approaches to produce uniform average projected values. A voting regressor is a method that fits each base regressor to the entire dataset. A regressor like this can help a group of similar performance-level estimators balance out their flaws. Ensemble methods operate best when the predictors are as independent as feasible. Generally, each regressor is trained using a distinct technique to make each prediction more independent of the others. This raises the likelihood that they will make a variety of blunders, which will improve the ensemble's performance. The voting regressor can be applied for classification or regression. Each label's predictions are put together when it comes to classification, and the label with the most votes is chosen. This pertains to calculating the mean of the predictions made by the models in the case of regression. A voting ensemble

should be used when all models that apply to it should perform well on a predictive modelling assignment, according to Witten et al. [60]. In other words, the ensemble's models must largely concur with one another.

3.3 Summary

In this chapter, the machine learning techniques used in this dissertation have been introduced. These techniques diverge into two groups as represented. In particular, the group related to clustering algorithms introduces some clustering algorithms that have been used in the thesis to find the most suitable clustering algorithm in this research context. The second group is the algorithms used to find an appropriate algorithm and recommend a new calibration complexity weight system. In optimizing the estimated result after applying the proposed complexity weight system, an ensemble algorithm (Voting Regressor) was also used and introduced in this section.

4. CURRENT STATE OF EFFORT ESTIMATION

As mentioned in section 2, software estimation techniques today involve three approaches: non-algorithmic, algorithmic, and machine learning. Each approach has its advantages and optimal domain of application. In this study, an algorithmic approach (in particular, IFPUG FPA) is used as the fundamental technique in which the size estimation stage is customized based on machine learning techniques to calibrate the system complexity weight. This section will examine the state of the current effort estimation studies. We will look at the workaround of these approaches.

The FPA method has made certain contributions to the software industry. Albrecht [11] first introduced FPA in 1979 and presented the Functional Point (FP) metric to measure the functionality of a project. It was proposed in response to a number of problems with other system size measures, such as lines of code. Effective software development and maintenance management with FPA was advocated and made more widely known in 1986 by the International Function Point User Group (IFPUG) [23]

However, the FPA method has encountered some disputes from different researchers - in terms of advantages and limitations. Sheetz et al. [61] studied FPA from a manager´s and developer´s perspective-based on 13 attributes with three key findings: SLOC count is less complicated than FP; developers are better able to comprehend the benefits of FP than managers; the difference between managers and developers is in the Values Block Communication necessary to propound informed decisions. Some studies reported that the FPA method does not create consistent results when applied to different metrics [62, 63]. Meli [64]

pointed out a mismatch between the complexities established for the Base Functional Components and the possible productivity estimates.

Many studies showed that the FPA scored the BFC incorrectly. For example, the same data function and / or the same transactional function with different combinations of DET and RET/FTR can be categorized with the same complexity. This leads to the same number of function points for the data functions and / or transactional functions. They also notice that - in some situations, functionalities that have very similar DET and RET / FTR can be categorized with different complexities; and thus, receive different FPA weightings.

Xia et al. [65] doubted that the Unadjusted Function Points weight values, which were raised based on a study of the IBM data processing systems - (locally), could not reflect the software globally. In [66], they continually point out the existence of ambiguous classification, and the original method may not fully reflect the reality of the software complexity under the specific software application. In [67], they proved that there is no clear boundary between two classifications in FP counting. To resolve these problems, the authors suggested the merging of three techniques - (Fuzzy Logic, Artificial Neural Networks, and Statistical Regression) in a neuro-fuzzy function point calibration model.

Ahmed et al. [68] showed that many factors could be affected by the complexity of FP weight metrics values, like methodologies to develop software, support tools, and other factors. The authors proposed that new FPA weights were measured based on an adapted genetic algorithm. The proposed algorithm is based on a set of initial solutions - using biologically inspired evolution mechanisms to derive new - and possibly better, solutions.

According to Hajri et al. [69], the classification of function types into simple, average, and complex does not reflect the entire complexity necessary to develop user systems. The main improvement idea of this research is to establish a new weighting system for FP measurement using Artificial Neural Networks (ANN), that is to say, (the back-propagation technique). In the first step, they use the original weights system as baselines in order to establish the new weights. Next, they train one of the most popular Neural Networks techniques to predict the values of the new weights. And then, they apply the new weights and the original weights in the FP model. Finally, they calculate the FP count, depending on the original and new weights.

Ya-Fang et al. [70] studied that the BFCs weights - which were set by IFPUG, are said to reflect the functional size of the software, but actually - today's software differs drastically from the past; so it is no longer suitable. Authors also discovered that this inconsistency in a large number of BFCs, which lies on the specified intervals´ boundary areas, becomes even worse. The cause is due to the

inaccurate classification of various system functionalities – which would distort its functional size.

The Function Points Measurement Process is not accurate in some specific cases, as demonstrated by Rao and Raju [71], and the number of referenced items, which establishes the lower limits of the high-complexity range, can lead to the same measurement accuracy issues, particularly in systems that reference a variety of data element types (DETs).

To learn about various machine learning (ML) strategies, their estimate accuracy, and the comparison between multiple models and estimation contexts, Wen et al. [72] looked through 84 original studies of ML techniques in SEE. This study discovered that eight different ML techniques have been used in SEE and concluded that ML models offer more accurate estimates than non-ML models. Case-Based Reasoning, Artificial Neural Networks, Decision Trees, Bayesian Networks, Support Vector Regression, Genetic Algorithms, Genetic Programming, and Association Rules are the eight ML subtypes mentioned above. They also discovered that DT, ANN, and CBR are utilized the most frequently.

Phannachitta [73] analyzed 13 different datasets using 14 machine learning algorithms frequently utilized in data science. According to the results, two algorithms, random forest and bagging, outperform the other algorithms. Additionally, the author suggests merging algorithms to get better outcomes. The author revisits another study's comparison of software effort adaptors based on heuristics and machine learning methods [74]. The authors integrated the seven separate methods. Ordinary least squares regression, classification and regression trees, SVR, ANN, deep random forests, and Gradient boosting machines are the algorithms employed in this work. The findings of this research suggest that a combination model is required to get more accurate estimates. The study's top performer was the analogy-based model, which adjusts to the effort by combining the Gradient boosting machine algorithm and a conventional adaptation method based on productivity adjustment.

S. Shukla and S. Khumar [75] use LR, SVM, KNN, and ANN algorithms to find a more feasible model for estimating software effort. The authors prove experimentally that the ANN algorithm is the best in this case. In another study [76], the authors used ANN with its ensembles (Ridge-MLPNN, Lasso-MLPNN, Bagging-MLPNN, and AdaBoost-MLPNN) to improve the software performance estimation process. The result signified that this model improves the performance compared to just ANN, and the combination of AdaBoost-MLPNN produced the highest accuracy. Priya Varshini et al. [77] used the ensemble technique to find the best suitable method with the same idea about using ensemble approaches for software estimation. Ensemble techniques studied for assessment were averaging, weighted averaging, stacking, boosting, and bagging. Single models considered for comparison were SVM, decision tree, random forest, neural net, ridge,

LASSO, elastic net, and deep net algorithms. The proposed stacking using random forest provided the best result. This result was compared with the singles model and also got outperformed.

Additionally, Hammad et al. [78] examined four different algorithms (MLR, SVM, ANN, and K-Star) in estimating the real effort from the software features at the early phases of the software development life cycle to find a good ML technique for predicting software effort. The outcomes demonstrate the viability of using ML for software effort estimation. The results produced by SVM are the best of the four suggested algorithms.

Another issue related to this study is the application of data clustering techniques to improve the accuracy of the software effort estimation process.

Using parametric models with a mathematical foundation has some drawbacks, as demonstrated by Aroba et al. [79]. These restrictions can be overcome by combining segmentation models with the participation of other models to produce one model. Due to the fuzziness, it is crucial to take into account the fact that a project only fits within one segment. In order to estimate software cost, a segmentation model based on fuzzy logic is proposed. According to experimental findings, accuracy has dramatically increased.

Using three categorical variables—relative size, industrial sector, and organization type area - Silhavy et al. [80] created a novel categorical variable segmentation model based on dataset segmentation. The category variable of relative size serves as the segmentation parameter for the suggested approach. The proposed approach beats the IFPUG FPA model, spectral clustering-based models, and regression models in terms of estimation accuracy.

A soft computing approach to estimating software effort was suggested by Azath et al. [81]. The dataset clustered by the fuzzy-c-means clustering algorithm will be used to produce the rules. These learned rules will serve as the input for another neural network-based operation. This study's neural network model is an amalgam of optimization algorithms. This optimization algorithm will use the algorithms Artificial Bee Colony (ABC), Modified Cuckoo Search (MCS), and hybrid ABC-MCS. The experiment made use of the NASA 60, NASA 90, and Desharnais databases. The outcomes obtained using this suggested paradigm are excellent.

Prokopova et al. [12] analyze three different distance metrics using k-means, hierarchical, and density-based clustering techniques. The outcomes emphasize the significance of choosing the proper clustering type and distance metric. The authors demonstrate that hierarchical clustering results in inaccurate cluster distributions and can thus not be used. K-means clustering appears to be the segmentation technique that performs best.

According to Bardisiri et al. [82], clustering as a method of dataset segmentation significantly impacts the accuracy of development effort estimation

since it enables the removal of insignificant projects from historical data points. In that study, the authors put up a hybrid model that incorporates fuzzy clustering, artificial neural networks, and analogy-based estimation. The test made use of the Desharnais and Maxwell datasets. The experimental findings show promise, reaching up to 127% for the PRED (0.25) evaluation criterion.

Benala et al. [83] combined functional link artificial neural networks with unsupervised learning approaches in a study to forecast the software effort (clustering algorithms). To thoroughly examine the performance, the Functional Link Artificial Neural Networks (FLANNs) technique was employed in this instance. Chebyshev polynomials were chosen as the functional expansion method. The empirical evaluation of this proposed method took into account three real-world datasets related to software cost estimation. The experimental results demonstrate that the proposed method performs well for software cost estimates and can significantly increase the prediction accuracy of standard FLANN.

The above are some of the studies related to this study. In addition, there are many other studies with proposed solutions to increase the accuracy of the estimation process. However, with certain limitations, this study cannot fully address these contributions.

5. PROPOSED METHOD

The essential idea of this study is based on the combination of the IFPUG FPA method and the machine learning techniques. The FPA takes the basement role, and the machine learning techniques play an inference role. First, two following phases should be done as the premise for the whole process: 1) find the best suitable machine learning algorithm and 2) find the best suitable clustering criterion. After selecting the best suitable algorithm and clustering criterion, the calibration phase calibrates and proposes the functional complexity weight system.

The new project that needs the effort estimation uses the FPA for counting function points with the default complexity weight (of the FPA method) was replaced by the new complexity weight system. This phase's result (effort) will be optimized using the effort optimization framework.

The effort estimation framework uses the Voting ensemble model with four base estimators (Random Forest Regressor, Bayesian Ridge Regressor, MLP Regressor, and LASSO). The result after this phase is the final result. All these processes can be illustrated in Fig. 5-1.

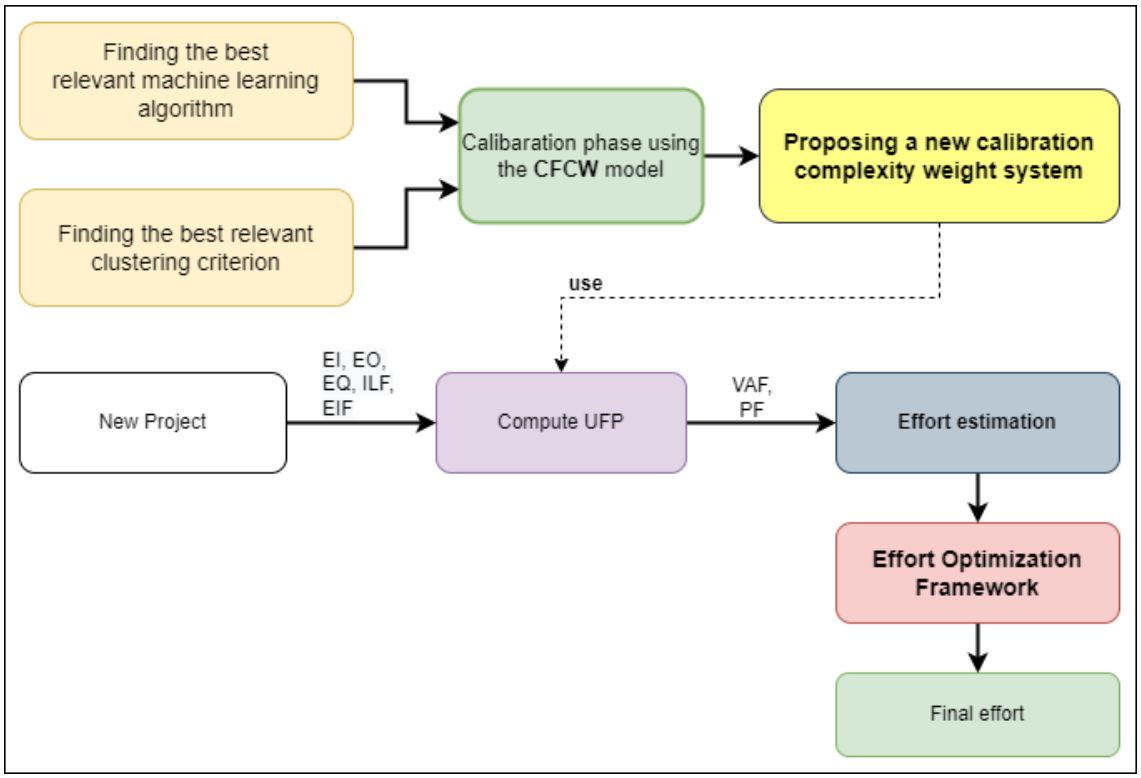


Fig. 5-1.Theoretical Framework

In the main step of proposing the new calibration complexity weight system, the Calibration of Functional Complexity Weight (CFCW) algorithm is proposed. This algorithm (CFCW) elicits the complexity weights from the EI, EO, EQ, EIF, and EIF variables using Bayesian Ridge Regression.

Another vital proposition is the effort optimization framework (named CFCW Optimization - CFCWO). This framework is constructed based on an ensemble algorithm with some base estimators. This framework takes a final role in the estimation of the effort.

6. EXPERIMENT PART

This section presents the experiment. There are four processes in this part. The first is data processing. The second and the third processes can be processed parallelly: Finding the best suitable clustering criterion and algorithm. The final step is to propose the functional complexity weight system and the optimization framework. Each of them will be introduced below. Moreover, with the purpose of minimizing the bias associated with the random sampling of the training and holdout data samples in comparing the estimative accuracy of two or more methods, the k-fold cross-validation (k=5) method was used in all experiments.

6.1 Data processing

The dataset we used in our experiment is the ISBSG repository August 2020 R1 [84]. The basic criteria for data filtering will be presented as follows.

1. The selected data quality of records is A and B.
2. We only select records with the counting approach of IFPUG (including IFPUG Old and IFPUG 4+).
3. The development Type should be New Development.
4. The rows with an empty value of BSCs should be eliminated.
5. Rows with empty values in Normalized Productivity Delivery Rate (PDR) and Summary Work Effort were also erased.
6. Fill the blank cell of VAF by the value that got from the formula $AFP = UFP \times VAF$.

In the ISBSG dataset, we considered the correlation between SWE and AFP variables. The Pearson correlation coefficient was used to summarize the linear relationship's strength. Fig. 6-1 describes this correlation. The Pearson's correlation value of 0.58 means a positive relationship exists between these variables.

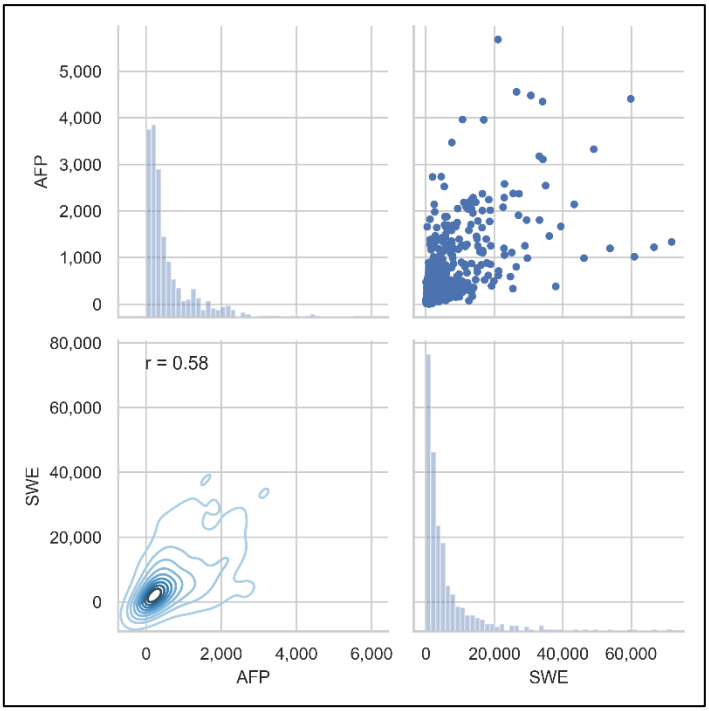


Fig. 6-1. The correlation between AFP and SWE in the ISBSG dataset

Additionally, we observe that several AFP and SWE values are too far from the mean group, implying that the data could be noisy. We determined and removed outliers using the Interquartile Range (IQR) method [85, 86] on these features in this study, with the subordinate bound being 0.15 and the upper bound

is 0.85. This technique was utilized in all of the studies in this study to eliminate outliers.

6.2 Applying machine learning algorithms in effort estimation

Choosing the suitable algorithm for the estimator is a matter of first concern. Selecting a lousy algorithm can lead to estimating results that are not as expected. The development project can end up failing because of this selection. Moreover, for each different dataset, the suitable algorithm for it is also different. A practical algorithm for a dataset could not necessarily be adequate for another.

Many ML algorithms have been proposed and applied in recent years. Each algorithm proves its superiority in certain areas. Hence, between these algorithms, whether any algorithm will be suitable for the proposal in the research of this thesis. Another important thing is that we can hardly be using all ML algorithms. So, choosing the applied algorithms in the experiment is also challenging. In this study, we carry out a survey to find out the most used algorithms today and then select some algorithms to test.

Table 6-1. Machine learning algorithms used in the survey

No.	Algorithms	References
1	Linear Regression	[87], [74], [78], [75], [88], [89], [90], [91], [92], [93], [94], [95], [80], [96]
2	Support Vector Machine	[72] [74], [78], [75], [77], [88], [90], [97], [98], [99], [100]
3	Artificial Neural Networks	[72], [74], [78], [75], [76], [89], [90], [97], [91], [98], [92]
4	Ridge Regression	[87], [74], [73], [76], [77], [88], [95], [100], [101]
5	Least Absolute Shrinkage and Selection Operator	[87], [74], [76], [77], [88], [93], [96]
6	Random Forests	[74], [77], [88], [102], [97]
7	K-Nearest Neighbor	[75], [91], [98], [99]
8	Decision Trees	[72], [77], [88], [98]
9	Elastic Net	[74], [77], [88]
10	Bagging	[74], [76]
11	Adaptive Boosting	[74], [76]
12	Naïve Bayes	[102], [99]
13	Logistic Regression	[102], [95]
14	Neural Net	[77], [88]
15	Bayesian Networks	[72], [74]
16	Genetic Algorithms	[72], [103]
18	Association Rules	[72]

19	Least-angle regression	[74]
20	CART	[74]
21	Analogy based estimation	[74]
22	Gradient boosting machine	[74]
23	k-star	[78]
24	Case-Based Reasoning	[72]

Based on Table 6-1, the survey's five most-used ML algorithms are Linear Regression, SVM, ANN, Ridge Regression, and LASSO. They are the algorithms used in the CFCW model to discover the most suitable algorithm.

We perform an experiment to find the most suitable algorithm for proposing the new calibration functional complexity weight system based on these selected algorithms. This experimental process can be illustrated in Fig. 6-2.

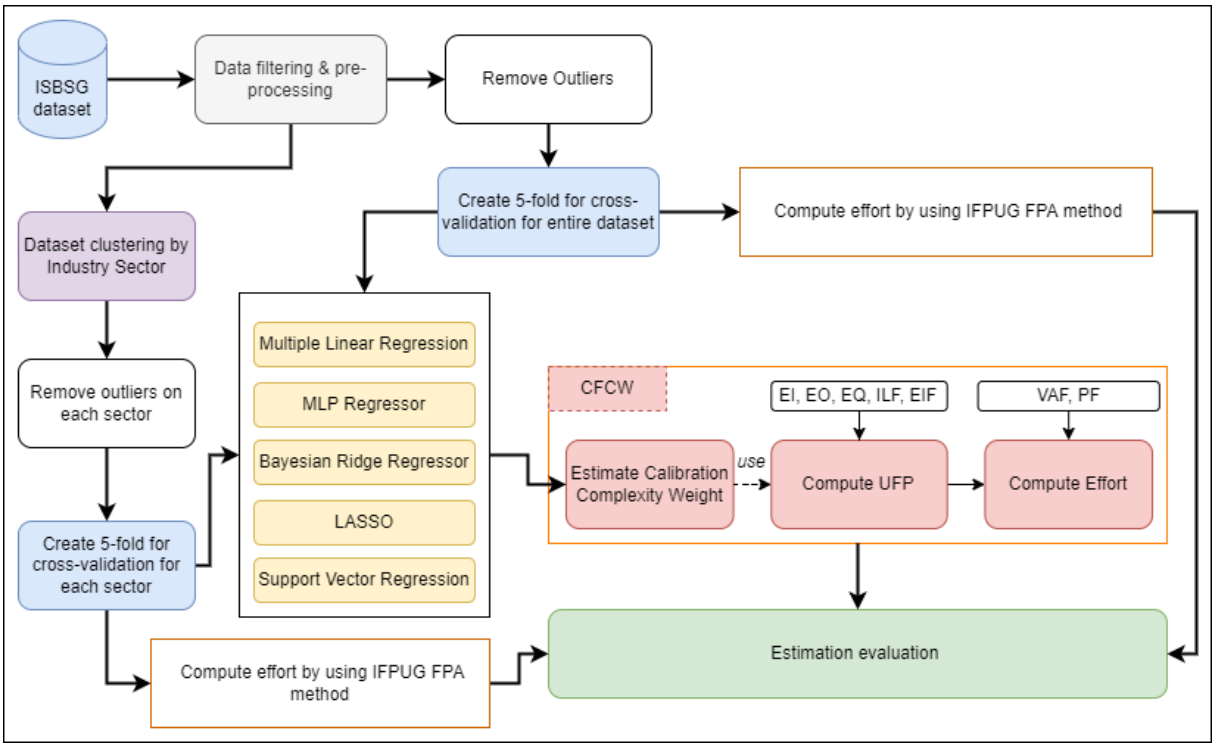


Fig. 6-2. Finding the best relevant algorithm

First, the dataset will be filtered and preprocessed, as described in section 6.1. After this step, the dataset is now called the tested dataset. Four cases need to be evaluated in this test. We can group them into two groups: the first group includes the first and second cases; the second group includes the third and fourth cases.

In the first case, the experiment will be conducted using the FPA method on the entire non-clustered dataset. This case is considered the baseline model for comparison with other models.

The second case will still be performed on the entire non-clustered dataset but now apply the CFCW algorithm in the effort estimation process. Specifically, this

case will use five machine learning algorithms, MLR, MLP, BRR, LASSO, and SVR, respectively, to estimate complexity weight.

In the third case, the tested dataset will be segmented according to the Industry Sector categorical variable, then use the FPA method on the segments to estimate the effort.

The last case of this experiment is to use the CFCW algorithm on five machine learning algorithms on the clusters identified by the Industry Sector categorical variable segmentation.

These cases will be evaluated through the evaluation criteria mentioned in the section and compared. The ultimate goal is to find the algorithm with the lowest estimation error.

In short, the tested models used in this experiment are:

1. IFPUG FPA method on the entire non-clustered dataset.
2. CFCW method with the specific algorithm on the entire non-clustered dataset.
3. IFPUG FPA method on the clusters formed by clustering the ISBSG dataset using the Industry Sector categorical variable.
4. CFCW method with the specific algorithm on the clusters formed by clustering the ISBSG dataset using the Industry Sector categorical variable.

6.3 Applying segmentation techniques in effort estimation

This section considers two segmentation approaches: 1) segmentation based on categorical variables; 2) segmentation based on clustering algorithms. This process aims to find the best suitable segmentation criterion for proposing the new calibration functional complexity weight system.

In general, the terms clustering and segmentation may have different meanings, but in this study, these two concepts are the same meaning and are interchangeable.

6.3.1 Using categorical variables

The same problem for algorithm selection; in the ISBSG dataset, many categorical variables can be used for segmentation. We can hardly test for all these variables. Therefore, we will choose the representative variables for this study through a survey to investigate the recent studies that used categorical variables. Table 6-2 lists categorical variables in this survey.

Table 6-2. Categorical variables used in the survey

No.	Variables	References
1	Development Type (DT)	[104], [105], [106], [107], [108], [109], [110], [111]
2	Development Platform (DP)	[104], [112], [108], [113], [109], [110], [111]
3	Language Type (LT)	[112], [114], [108], [109], [110], [115], [111]
4	Industry Sector (IS)	[116], [117], [80], [118], [106], [108], [110]
5	Organization Type (OT)	[104], [105], [113], [111]
6	Relative Size (RS)	[118], [80], [110], [115]
7	Application Type (AT)	[104], [105], [106], [110]
8	Business Area Type (BAT)	[104], [118], [80]
9	Primary Programming Language (PPL)	[107], [108]
10	Application Group (AG)	[108]
11	1st Database System (1DB)	[108], [109]
12	Used Methodology (UM)	[108], [110]
13	Count Approach (CA)	[118], [115]
14	Project Type (PT)	[113]
15	Resources Level (RL)	[110]
16	1st Operation System (1OS)	[115]

Based on the survey, the categorical variables used in this study are the most used categorical variables. They are Development Platform (DP), Industry Sector (IS), Language Type (LT), Organization Type (OT), and Project Relative Size (RS). The experiment model is visualized in Fig. 6-3.

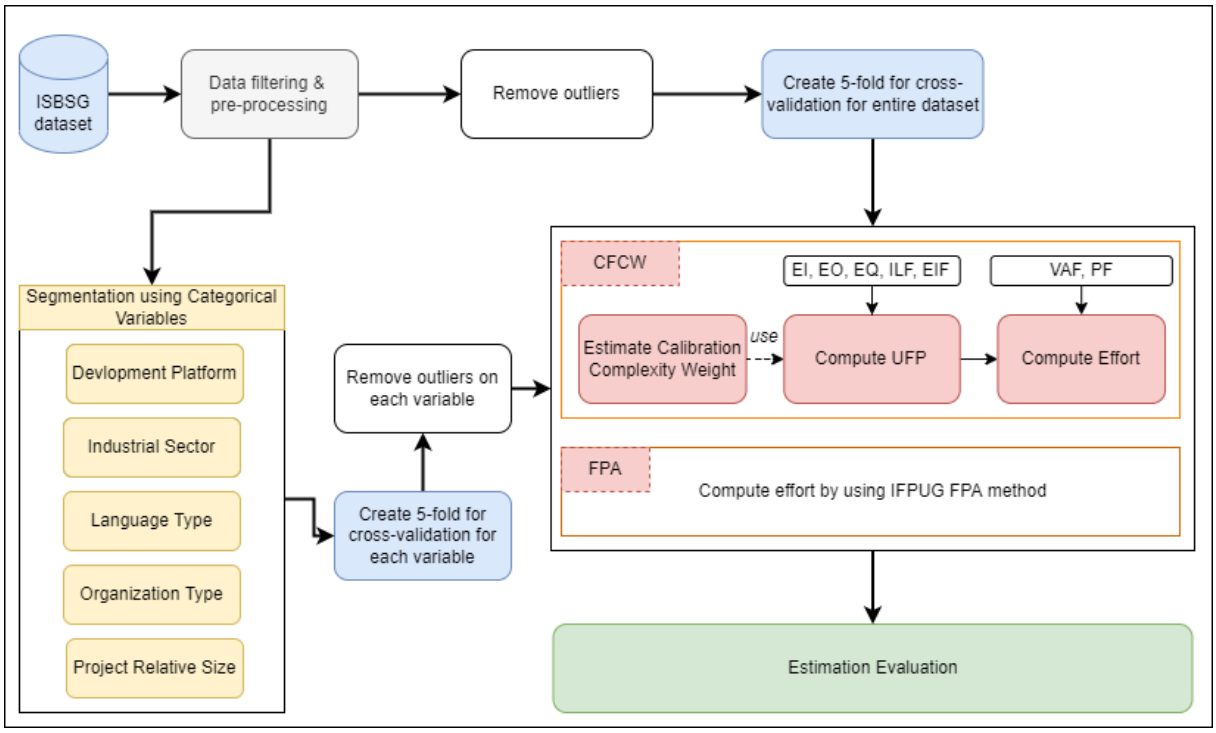


Fig. 6-3. Finding the best suitable categorical variables

There are four cases to evaluate in this model. The first case is still testing the IFPUG FPA method on an entire non-clustered dataset as a baseline model for comparison.

With the best suitable algorithm found in section 6.2, the second case is tested on the entire dataset with the CFCW algorithm built on this best-suitable machine learning algorithm.

The third and fourth cases start with segmenting the tested dataset according to the categorical variables mentioned. For each of these categorical variables, specific criteria are defined then the tested dataset will be segmented according to these criteria. The third case will be performed on the IFPUG FPA method, and the fourth case will be conducted on the CFCW method with the algorithm selected for the Complexity weight correction defined in section 6.4.

Tested models:

1. IFPUG FPA method on the entire non-clustered dataset.
2. CFCW method on the entire non-clustered dataset with suitable ML algorithm.
3. IFPUG FPA method on the clusters formed by clustering the ISBSG dataset using the specific categorical variable.
4. CFCW method on the clusters formed by clustering the ISBSG dataset using the specific categorical variable.

The segmentation criteria of the six categorical variables are described as follows:

Development Platform

The Development Platform defines the primary development platform (as determined by the operating system used [119]). Each project is classified as either a PC, Mid-Range (MR), Main Frame (MF), or Multi-platform (Multi). In our case, records without the development platform were categorized into a cluster named "Others." Fig. 6-4 is the histogram of the Development Platform variable, and Fig. 6-5 is the boxplot of the dataset before and after removing outliers.

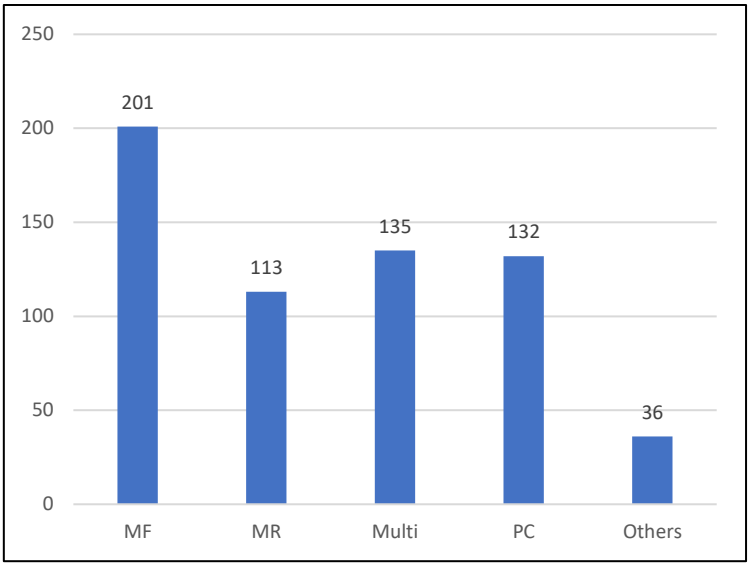


Fig. 6-4. Histogram of the Development Platform

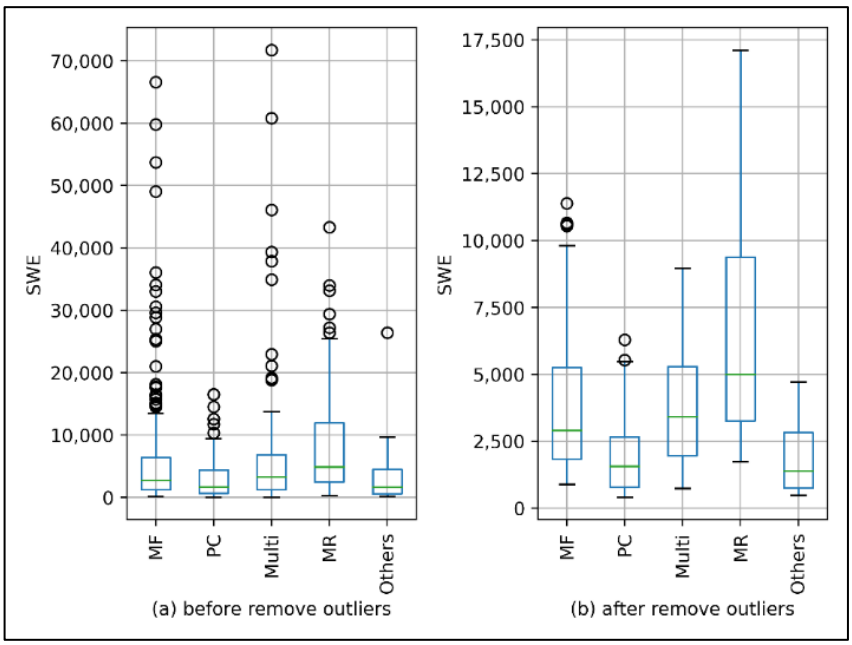


Fig. 6-5. Boxplot of the dataset clustered by DP

Industry Sector

The Industry Sector categorical variable is the organization's industry sector where software is maintained and supported. This study combined the industry sector with less than 30 records into a group named "Other". Fig. 6-6 is the histogram of the Industry Sector variable, and Fig. 6-7 is the boxplot of the dataset before and after removing outliers.

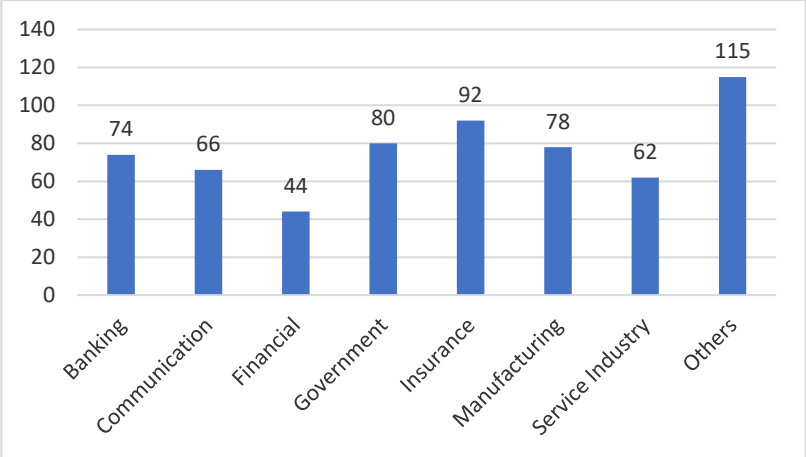


Fig. 6-6. Histogram of the Industry Sector

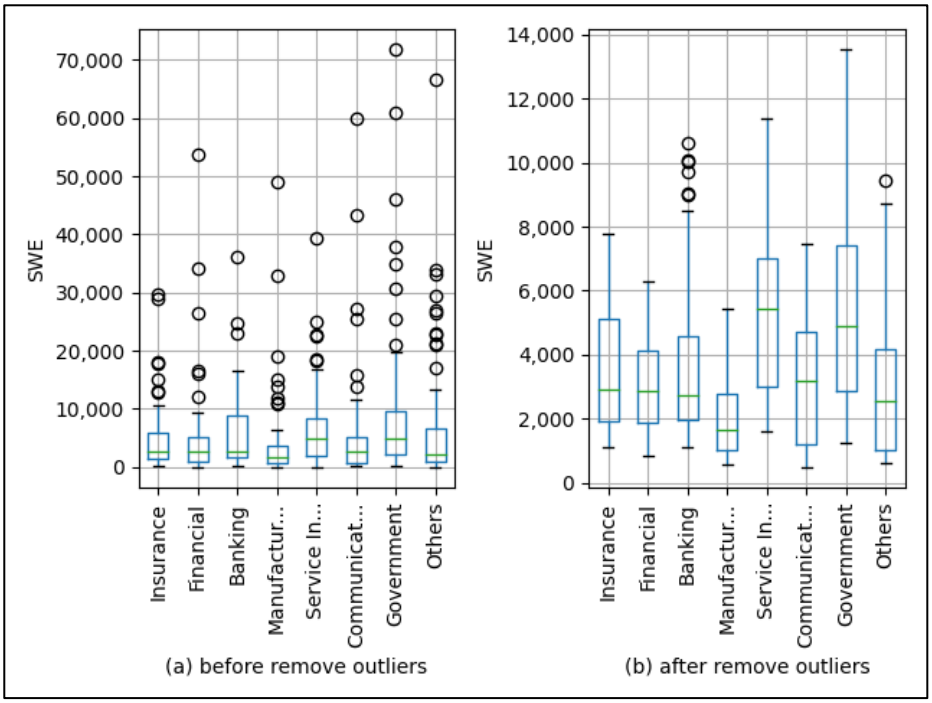


Fig. 6-7. Boxplot of the dataset clustered by IS

After processing, the remaining industry sectors are Banking (BAN), Communication (COM), Financial (FIN), Government (GOV), Insurance (INS), Manufacturing (MAN), Service Industry (SI), and Others.

Language Type

Language Type defines the type of programming language used for the project. In our study, there are three types of language: 3rd generation programming language (3GL), 4GL, and Others (projects with an empty Language Type field or the number of records less than 20). Fig. 6-8 is the histogram of the Language Type variable, and Fig. 6-9 is the boxplot of the dataset before and after removing outliers.

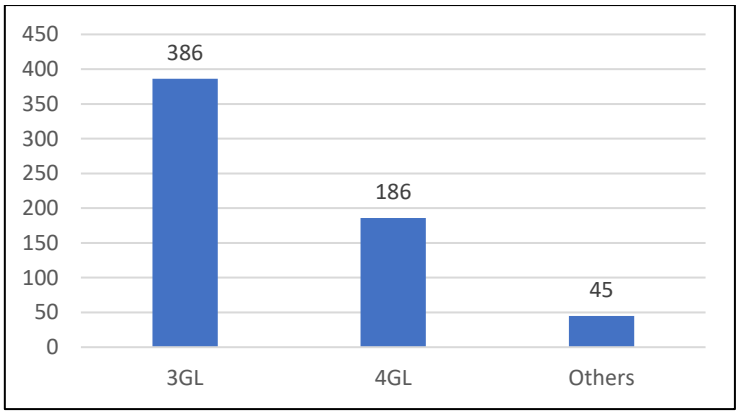


Fig. 6-8. Histogram of Language Type

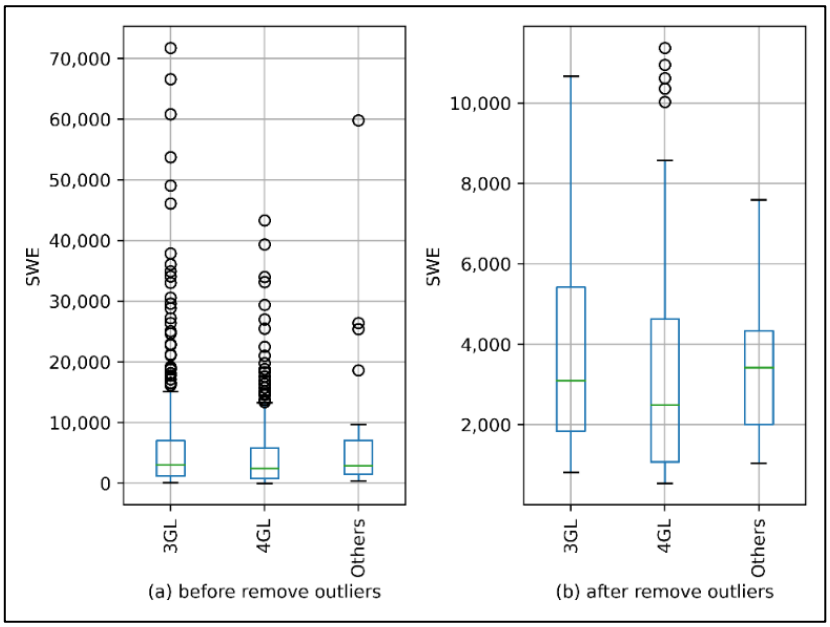


Fig. 6-9. Boxplot of the dataset clustered by LT

Organization Type

Organization Type identifies the type of organization that submitted the project. Fig. 6-10 is the histogram of the Organization Type variable, and Fig. 6-11 is the boxplot of the dataset before and after removing outliers.

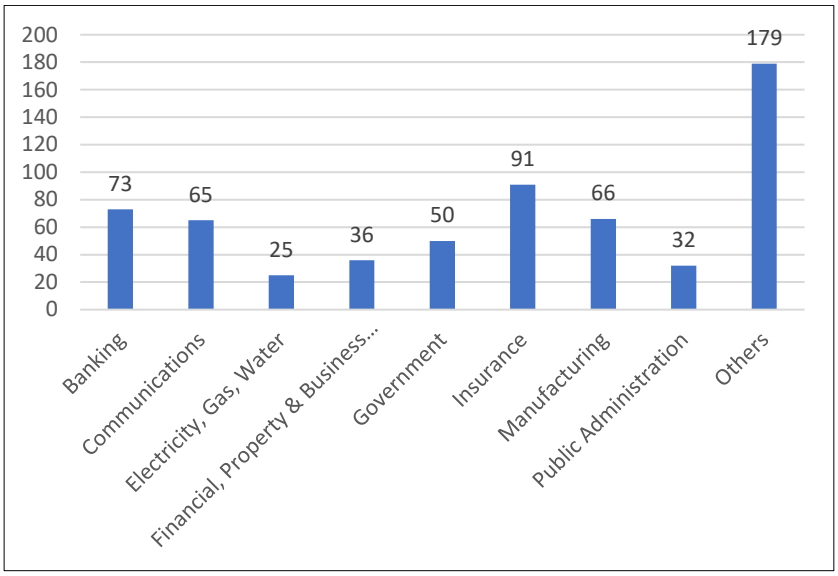


Fig. 6-10. Histogram of the Organization Type

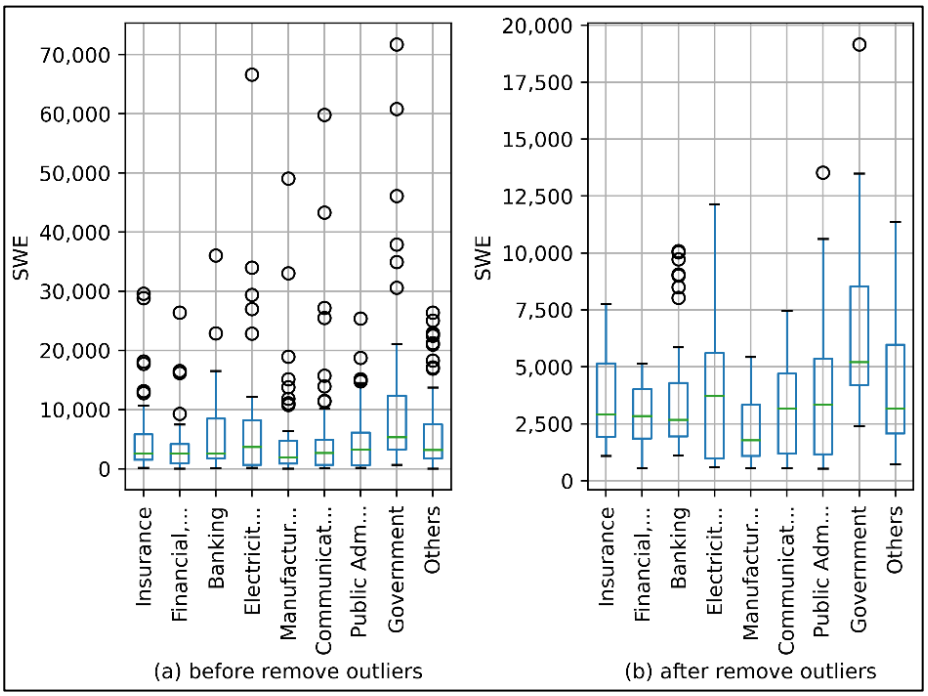


Fig. 6-11. Boxplot of the dataset clustered by OT

Relative Size

In the ISBSG dataset, there are nine types of relative size, as shown in Table 6-3. In our study, after filtering data, the relative size records number and their types are shown in Fig. 6-12. In the working dataset, the number of records in the XXS, XS, and XL groups is too slight (XXS=1, XS=8, XL=9); the XXL and XXXL are even zero. It cannot be used to represent the group in this study. So, we re-define the S group as the group with a functional size from 0 to 100; the L group has a functional size greater than 1000. According to this re-group, the S group includes XXS, XS, and S, and the L group includes L, XL, XXL, and XXXL.

Table 6-3. Relative size

No.	Abbr.	Relative Size	Functional Size
1	XXS	Extra-extra-small	>= 0 and <10
2	XS	Extra-small	>= 10 and <30
3	S	Small	>= 30 and <100
4	M1	Medium1	>= 100 and <300
5	M2	Medium2	>= 300 and <1000
6	L	Large	>= 1000 and < 3000
7	XL	Extra-large	>= 3000 and < 9000
8	XXL	Extra-extra-large	>= 9000 and < 18000
9	XXXL	Extra-extra-extra-large	>= 18,000

Fig. 6-12 is the histogram of the Relative Size variable, and Fig. 6-13 is the boxplot of the dataset before and after removing outliers.

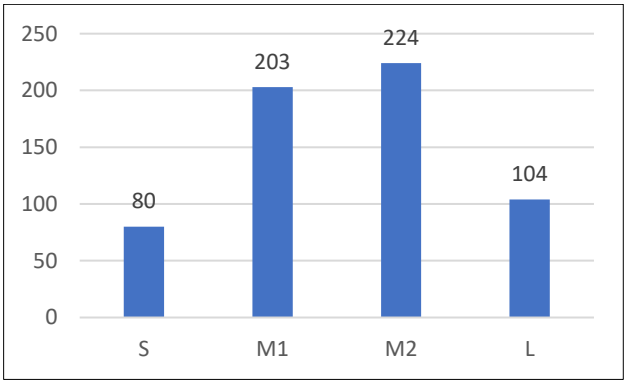


Fig. 6-12. Histogram of the Relative Size

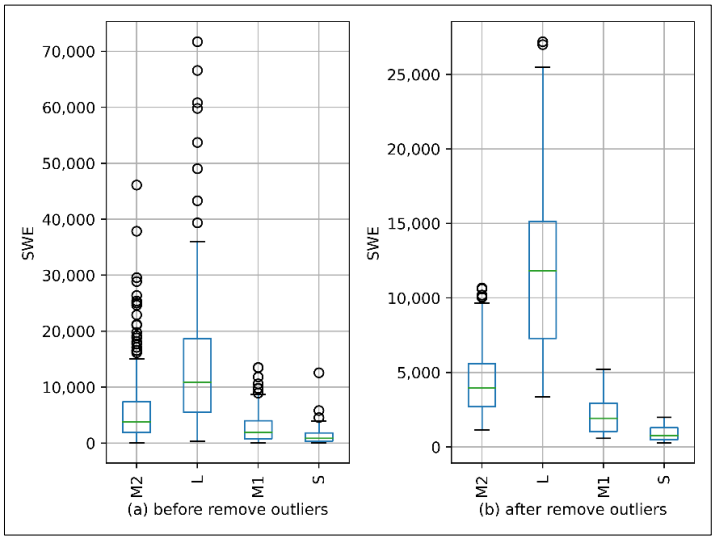


Fig. 6-13. Boxplot of the dataset clustered by RS

6.3.2 Using segmentation algorithms

Previous researchers have proposed many clustering algorithms. People have tried to organize them into groups of relative similarity for easy access and application. However, this categorization is just relatively because there are algorithms that can belong to different groups.

Regarding the problem of categorizing clustering algorithms, there have been many studies looking at the classification of these clustering algorithms. According to Xu [120], traditional clustering techniques are divided into nine categories with some illustrative algorithms. Table 6-4 presents this classification.

Table 6-4. Classification of clustering algorithms by Xu

No.	Categories (Clustering algorithm based on)	Typical algorithms
1	Partition	k-means, k-medoids, PAM, CLARA, CLARANS
2	Hierarchy	BIRCH, CURE, ROCK, CHAMELEON
3	Fuzzy theory	Fuzzy C-Means (FCM), FCS, MM
4	Distribution	GMM, DBCLASD
5	Density	DBSCAN, OPTICS, Mean-shift
6	Graph theory	CLICK, MST, Spectral Clustering
7	Grid	STING, CLIQUE, GRIDCLUS
8	Fractal theory	FC
9	Model	COBWEB, GMM, SOM, ART

According to another categorisation, Andreopoulos et al. [121], clustering techniques are divided into five categories, and the authors also listed some illustrative algorithms for each type, as shown in Table 6-5.

Table 6-5. Classification of clustering algorithms by Andreopoulos

No.	Categories	Typical algorithms
1	Partitioning	k-Means, k-Medois, CLARA, CLARANS
2	Hierarchical	Spectral, BIRCH, STING, CHAMELEON
3	Density-based	DBSCAN, OPTICS, DENCLUE, CLIQUE, Wave-Cluster
4	Model-based	COBWEB, Auto-Class, SVM Clustering
5	Graph-based	MCODE, SPC, RNSC, MCL

Gupta et al. [122] have statistics and classified clustering techniques into five groups and listed some represented algorithms for each group as described in Table 6-6. In addition, the author also mentions a classification called Modern

Clustering Methods that divides clustering techniques into the following categories: (1) Kernel-based algorithms, (2) Ensemble-based algorithms, (3) nature-inspired algorithms, (4) graph-based algorithms, (5) model-based algorithms, (6) Quantum Theory-based algorithms, etc.

Table 6-6. Classification of clustering algorithms by Gupta

No.	**Method**	**Typical algorithms**
1	Hierarchical Method	SLINK, CLINK, BIRCH, CURE, DIANA
2	Partition-based Method	k-Means, k-Medoids, PAM, CLARA, CLARANS
3	Density-based Method	DBSCAN, OPTICS, DENCLUE, RDBC
4	Grid-based Method	STING, CLIQUE, OPTIGRID, GRIDCLUS, GDILC, WaveCluster
5	Fuzzy-based Method	Fuzzy k-means, FCM, FCS, MM, MEC

In this study, according to the above classifications, we try to choose algorithms so that they belong to different groups relatively. This choice is difficult because we cannot evaluate all existing clustering algorithms. A representative group is possible within the realm of possibility. Therefore, the following algorithms are selected in the experiment of this study.

Table 6-7. Selected clustering algorithms

No.	**Clustering Algorithms**	**Abbr.**
1	Balanced Iterative Reducing and Clustering Hierarchies (BIRCH)	BIR
2	Fuzzy C-Means	FCM
3	Gaussian Mixture Model	GMM
4	k-means	KM
5	MeanShift	MS
6	Spectral	SC

The process of this experiment can be illustrated in Fig. 6-14. The features of all clustering algorithms in this study are EI, EO, EQ, ILF, EIF, and VAF.

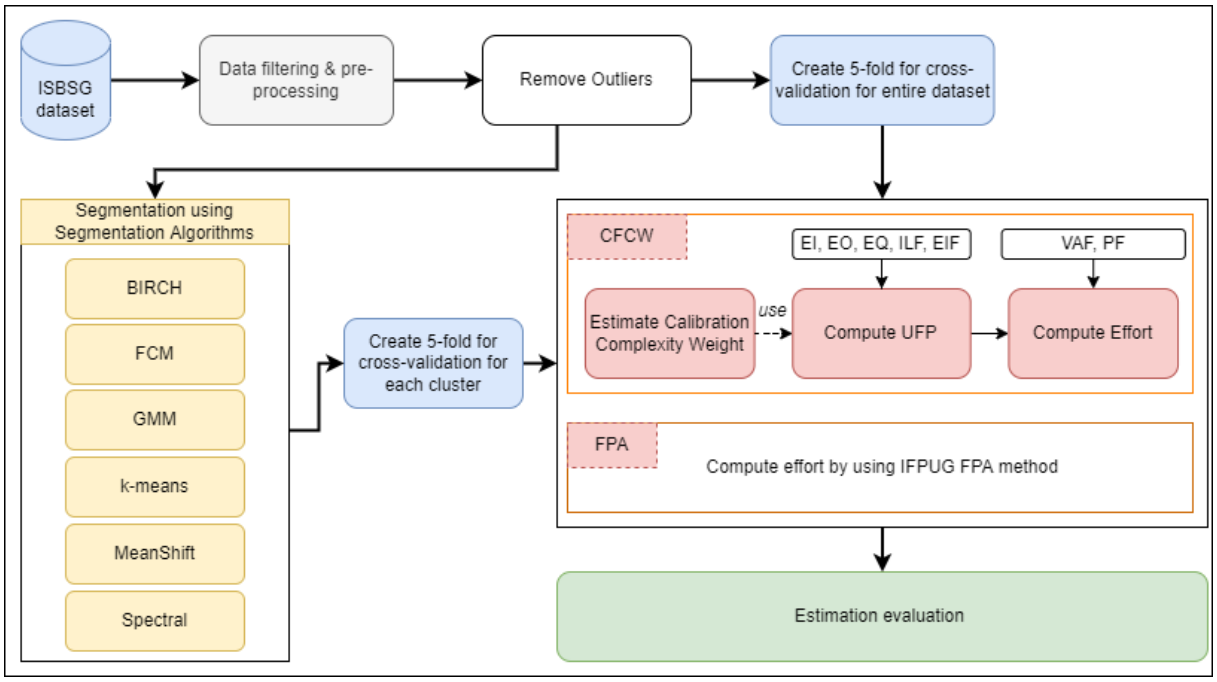


Fig. 6-14. Finding the best suitable segmentation algorithms

Firstly, the filtering and pre-processing of the ISBSG dataset should be done; Then, the outliers of the entire dataset should be detected and removed. After this step, the original dataset was called the tested dataset. Before performing the estimation step, the 5-fold cross-validation procedure was used for the full tested dataset in the baseline case. In the remaining cases, clustering algorithms were used to cluster the dataset. The tested dataset will be clustered into clusters, and the 5-fold cross-validation is also used on these clusters. The IFPUG FPA and CFCW estimate approaches were used in the estimation process. Finally, the results are compared using the estimation assessment with evaluation criteria. The CFCW algorithm, in this case also built on this best-suitable machine learning algorithm in section **6.2**.

Tested models in this experiment are:

1. IFPUG FPA method on the entire non-clustered dataset.
2. CFCW method on the entire non-clustered dataset.
3. IFPUG FPA method on clusters formed by clustering the ISBSG dataset using the specific clustering algorithm.
4. CFCW method on the clusters formed by clustering the ISBSG dataset using the specific clustering algorithm.

Some clustering algorithms don't need to determine the number of clusters as a parameter, while some are required. In selected clustering algorithms, BIRCH and MeanShift do not need to specify the number of clusters; Remains algorithms need this parameter. We will use the Silhouette methodology [123] in this study to determine the number of eligible clusters for each clustering methodology. The

following subsections discuss each clustering algorithm in more detail. Following are the results of determining the number of clusters with each algorithm.

Finding the number of clusters for the GMM clustering algorithm

Fig. 6-15 shows that the Silhouette score peaked at k=3. It means that the best number of clusters, in this case, is 3.

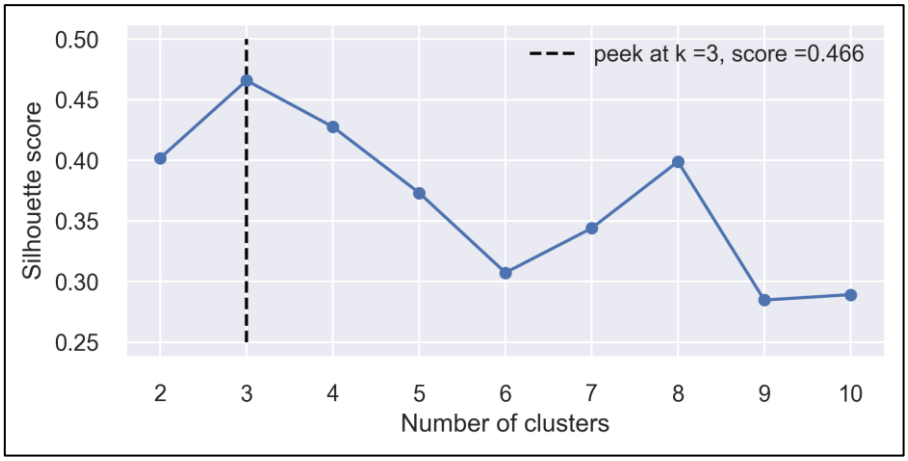


Fig. 6-15. Silhouette scores of datasets

Finding the number of clusters for the FCM clustering algorithm

Fig. 6-16 shows the silhouette scores for the clusters. We can see that the Silhouette score reaches its maximum value at k=3. Thus, 3 is the optimal cluster number in this case.

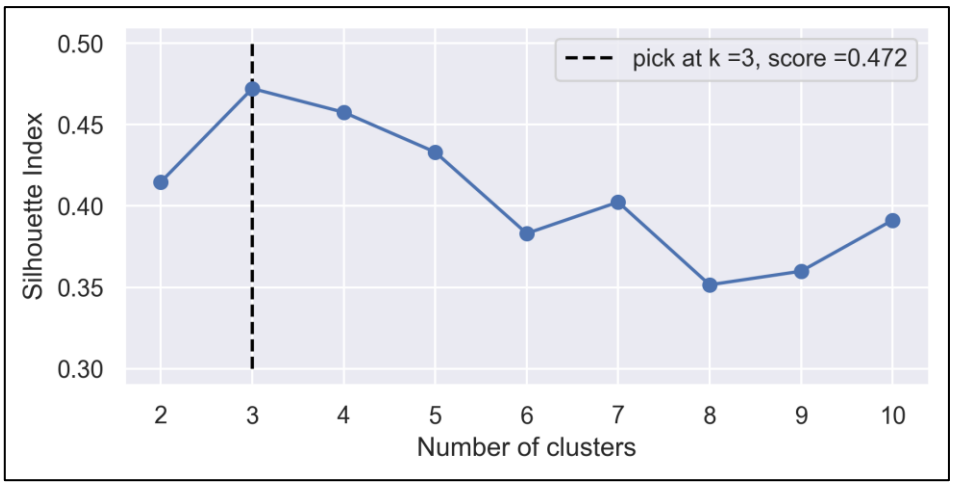


Fig. 6-16. Determine k-optimal for the FCM algorithm using the Silhouette score

Finding the number of clusters for the GMM clustering algorithm

Fig. 6-17 shows the silhouette scores for the clusters. Similarly, the number of optimal clusters for this clustering algorithm is 3.

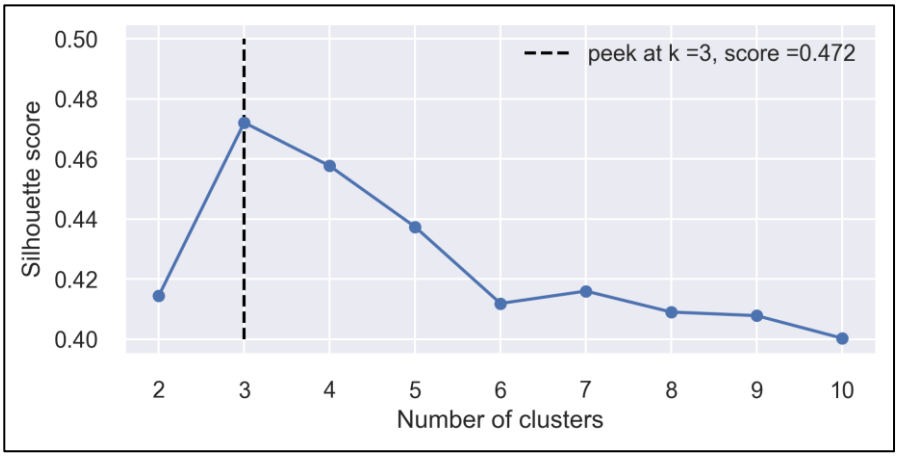


Fig. 6-17. Determine k-optimal for k-means algorithm using the Silhouette score

Finding the number of clusters for the Spectral clustering algorithm

With the Spectral clustering algorithm, k=3 is also the optimal number of clusters, as shown in Fig. 6-18.

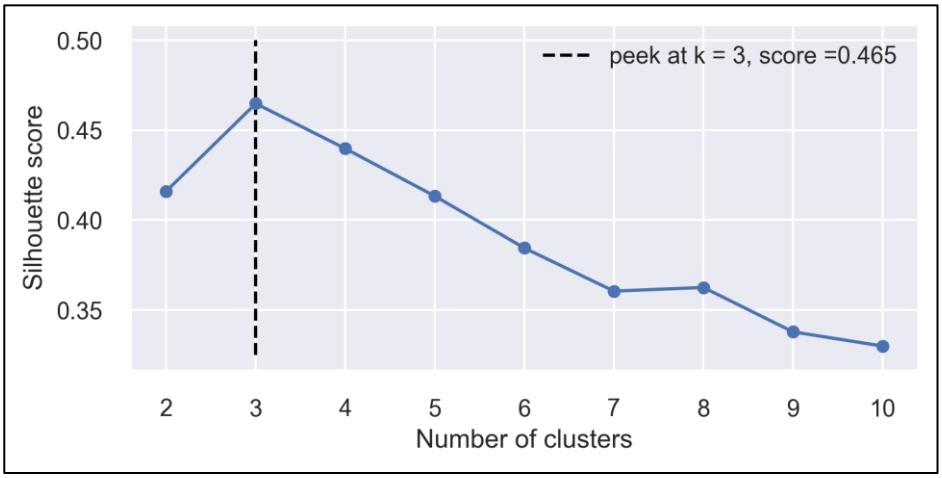


Fig. 6-18. Determine k-optimal for Spectral clustering

In actual implementation, the following Python libraries are selected for the mentioned clustering algorithms as well as the corresponding parameters are given in Table 6-8:

Table 6-8. Clustering algorithms and parameters

	Algorithm	**Implementation library**	**Parameters**
1	BIRCH	sklearn.cluster.Birch	n_clusters=None, threshold=0.45, branching_factor=500
2	FCM	fcmeans.FCM	n_clusters=3
3	GMM	sklearn.mixture.GaussianMixture	n_clusters=3

4	k-means	sklearn.cluster.KMeans	n_clusters=3
5	MeanShift	sklearn.cluster.MeanShift	bandwidth=estimate_bandwidth(X, quantile=0.2, n_samples=300, random_state=0)
6	Spectral	sklearn.cluster.SpectralClustering	n_clusters=3, random_state=0

6.4 Propose a New Calibration System and the Optimization Framework

After finding the best suitable algorithm and segmentation criterion, the next step will be to propose a new calibration complexity weight system and optimization framework. Fig. 6-19 presents the experiment phases for this process.

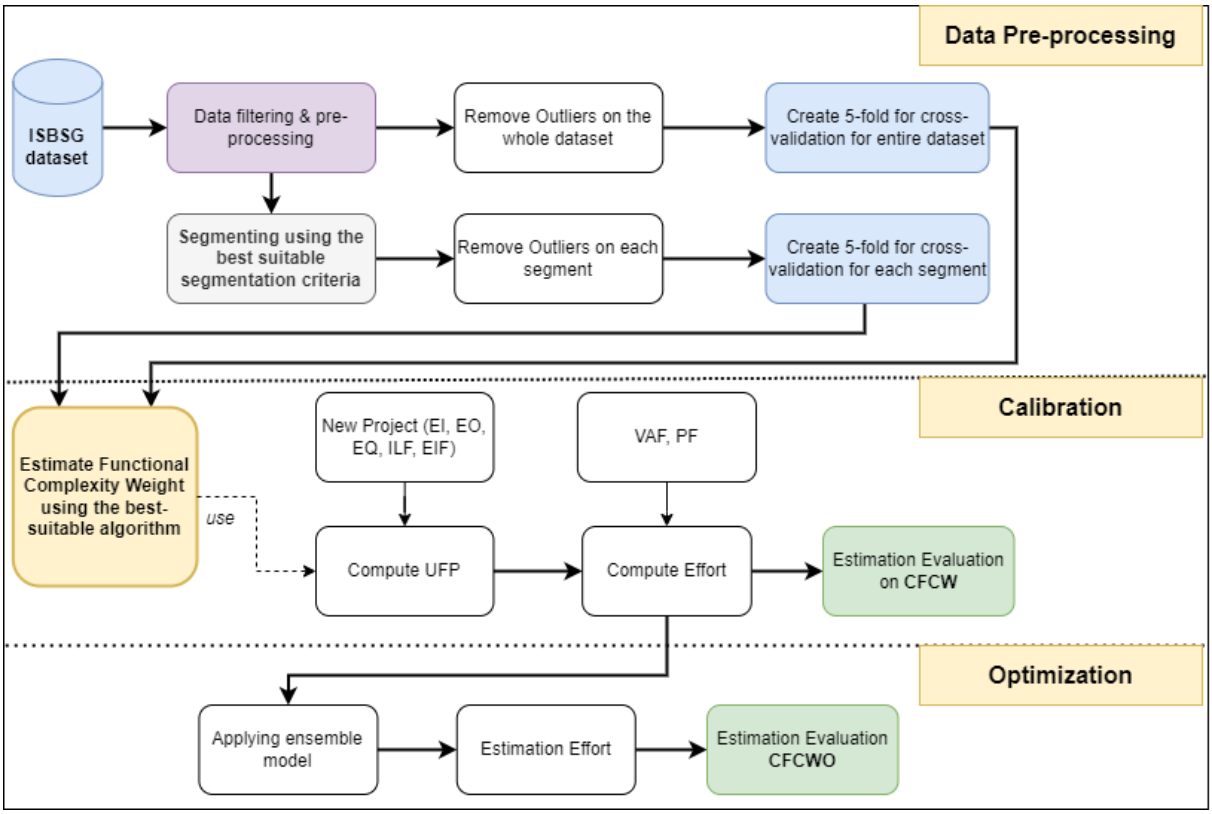


Fig. 6-19. Proposing new calibration complexity weight system and the optimization algorithms

Basically, this process has three phases: 1) Data pre-processing, 2) Calibration, and 3) Optimization.

In the first phase, the dataset will be filtered and preprocessed according to section 6.1. The data is now called the tested dataset. At this point, two experimental branches are deployed: one on the entire unsegmented tested dataset and the other on the tested dataset segmented according to the best-suitable clustering criteria found in section 6.2. Both branches are performed in the

subsequent calibration phase. In this calibration phase, the CFCW algorithm will be applied with the best-suitable machine learning algorithm according to section 6.3 to propose a new calibration functional complexity weight system. This calibration functional complexity weight system will be used to calculate the UFP value, followed by the effort estimation. After that, the process transitions to the third phase, optimization. In this optimization phase, an ensemble model (Voting Regressor with base-estimators Random Forest Regressor, Bayesian Ridge, MLP Regressor, and LASSO) will be trained with the input of effort calculated in the second phase and then output the final result.

Four tested models are grouped into two groups in this process. The first group includes tested models 1 and 2, and the second group contains tested models 3 and 4.

1. Applying the FPA method to the entire unsegmented dataset.
2. Applying the CFCW method and then the CFCWO method to the entire unsegmented dataset.
3. Applying the FPA method to the segments yielded by the best-suitable segmentation criteria.
4. Applying the CFCW method and then the CFCWO method to the segments yielded by the best-suitable segmentation criteria.

6.5 Evaluation Criteria

It is also essential to consider the criteria used to assess the accuracy of predictive models. This topic has generated many debates. According to some previous studies [19] - [124], [125], the most common metrics used are the Mean Magnitude of Relative Error Relative (MMER) and Mean Magnitude of Relative Error (MMRE) criteria. However, Shepper [126] and Azzeh [127, 128] have indicated that these methods are biased, so they are not exemplary at estimating estimates. Instead, these authors also recommend using unbiased evaluation criteria.

The presented thesis uses the evaluation criteria Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Balance Relative Error (MBRE), Mean Inverted Balance Relative Error (MIBRE), Prediction at level x (PRED(x)) and Mean Absolute Percentage Error (MAPE). These evaluation criteria produce an unbiased and symmetric distribution [129]. Following are the formulas of these evaluation criteria.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{6.1}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{6.2}$$

$$MAPE = \frac{1}{N}\sum_{i=0}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| * 100 \tag{6.3}$$

$$MBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|(y_i - \hat{y}_{i)}|}{min(y_i - \hat{y}_i)} \tag{6.4}$$

$$MIBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|(y_i - \hat{y}_{i)}|}{max(y_i - \hat{y}_i)} \tag{6.5}$$

$$PRED(x) = \frac{1}{n}\sum\nolimits_{i=1}^{n}\begin{cases}1 \;\; if \;\; \dfrac{|y_i - \hat{y}_i|}{y_i} \leq x \\ 0 \; otherwise\end{cases} \tag{6.6}$$

where y_i it the actual value, $\hat{y}_i$ is the estimated value, and n is the number of projects.

7. RESULTS AND DISCUSSION

This section presents the results archive from the experiment. Also, reiterate the goal of the investigation in this study is to find the most suitable algorithm and data segmentation criteria and then apply them to our proposal for a new calibration complexity weight system.

There are three subsections in this section: The first is finding the most suitable machine learning algorithm results, the second is the experimental result of determining the most appropriate segmentation criteria, and the third of proposing a new calibration complexity weight system.

7.1 Finding the best suitable machine learning algorithm

This section presents the results with the experimental model presented in section 6.2. In this section, there is a notation that we should mention: the CFCW model with the suffix is the notation for which algorithm means the CFCW model that uses that algorithm. For example, CFCW-MLP means the CFCW model uses the algorithm MLP.

7.1.1 On the entire non-clustered dataset

In the first group (including 2 experimental tested models 1 and 2 in which the IFPUG FPA and CFCW methods are applied to the entire dataset without clustering). Table 7-1 is the results of the first tested model, and tables from Table 7-2 to Table 7-6 are the results of the second test for the selected algorithms. In each of these tables, the first five rows are the average of each fold, and the last line is the mean of these folds.

Table 7-1. Evaluation results of the FPA method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	293.14	10.737	473.55	0.912	0.108	0.086
Ex 2	294.53	8.223	528.11	0.897	0.083	0.070
Ex 3	463.59	11.250	752.55	0.824	0.113	0.093
Ex 4	437.16	10.762	681.11	0.897	0.108	0.091
Ex 5	411.04	11.648	681.59	0.868	0.118	0.095
mean	*379.89*	*10.524*	*623.38*	*0.879*	*0.106*	*0.087*

Table 7-2. Evaluation results of the CFCW-MLR method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	264.51	9.221	386.65	0.956	0.096	0.079
Ex 2	267.42	7.571	425.88	0.956	0.079	0.069
Ex 3	429.89	10.245	694.97	0.853	0.104	0.087
Ex 4	336.36	8.665	519.45	0.956	0.090	0.079
Ex 5	331.20	9.776	520.58	0.927	0.101	0.083
mean	*325.88*	*9.096*	*509.50*	*0.929*	*0.094*	*0.079*

Table 7-3. Evaluation results of the CFCW-MLP method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	289.84	10.491	430.88	0.912	0.106	0.085
Ex 2	254.35	7.412	404.74	0.971	0.077	0.068
Ex 3	480.07	11.555	736.47	0.897	0.117	0.098
Ex 4	386.28	9.669	605.11	0.897	0.098	0.084
Ex 5	412.88	11.886	663.19	0.853	0.121	0.098
mean	*364.69*	*10.203*	*568.08*	*0.906*	*0.104*	*0.087*

Table 7-4. Evaluation results of the CFCW-BRR method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	258.91	8.965	375.96	0.956	0.092	0.076

Ex 2	256.55	7.144	404.90	0.985	0.074	0.065
Ex 3	417.37	9.809	680.76	0.882	0.100	0.084
Ex 4	325.61	8.151	493.74	0.985	0.084	0.074
Ex 5	330.57	9.603	512.09	0.956	0.099	0.082
mean	*317.80*	*8.734*	*493.49*	*0.953*	*0.090*	*0.076*

Table 7-5. Evaluation results of the CFCW-LAS method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	264.487	9.220	386.576	0.956	0.096	0.079
Ex 2	267.344	7.568	425.780	0.956	0.079	0.069
Ex 3	429.885	10.245	694.925	0.853	0.104	0.087
Ex 4	336.254	8.663	519.279	0.956	0.090	0.079
Ex 5	331.193	9.775	520.473	0.927	0.101	0.083
mean	*325.83*	*9.094*	*509.41*	*0.929*	*0.094*	*0.079*

Table 7-6. Evaluation results of the CFCW-SVR method on the unsegmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	274.08	9.602	420.79	0.927	0.098	0.080
Ex 2	270.56	7.528	446.26	0.941	0.077	0.067
Ex 3	443.56	10.557	718.24	0.838	0.107	0.089
Ex 4	406.06	9.855	627.45	0.912	0.099	0.085
Ex 5	362.04	10.501	581.87	0.897	0.107	0.088
mean	*351.26*	*9.609*	*558.92*	*0.903*	*0.098*	*0.082*

Table 7-7 contains the evaluation results compiled from the above tables. It was constructed by all the mean rows of each algorithm.

Table 7-7. First experiment group mean values of all algorithms results

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
FPA	379.89	10.524	623.38	0.879	0.106	0.087
CFCW-MLR	325.88	9.096	509.50	0.929	0.094	0.079
CFCW-MLP	364.69	10.203	568.08	0.906	0.104	0.087
CFCW-BRR	317.80	8.734	493.49	0.953	0.090	0.076
CFCW-LAS	325.83	9.094	509.41	0.929	0.094	0.079
CFCW-SVR	351.26	9.609	558.92	0.903	0.098	0.082

From Table 7-7, we can observe that with five evaluation criteria, MAE, MAPE, RMSE, MBRE, and MIBRE, the value of the CFCW model using the BRR algorithm has the minimum value. With the PRED (0.25) evaluation criteria,

the maximum value belonged to the BRR algorithm in the CFCW model. These results indicate that the CFCW model with the BRR algorithm gets the most accuracy. Fig. 7-1 shows us the visualization of this experiment's results.

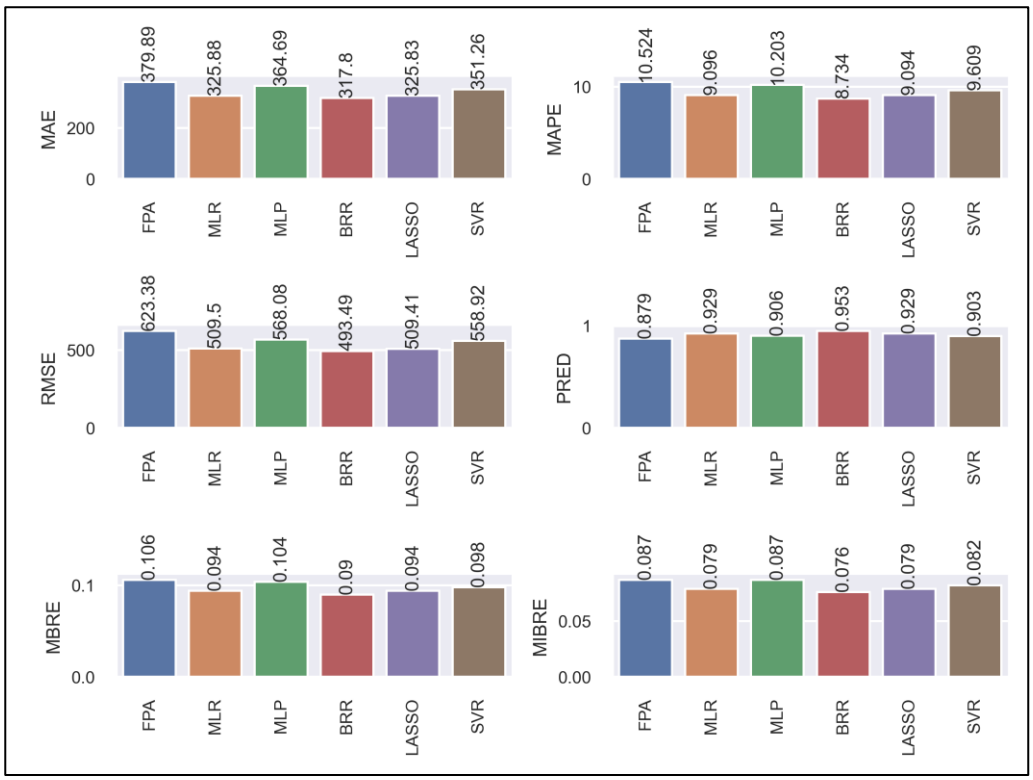


Fig. 7-1. evaluation results

7.1.2 On the clustered dataset

In the second group, the experiment includes two tested models (3 and 4). The IFPUG FPA and CFCW methods are applied to the segmenting dataset by Industry Sector categorical variables.

Table 7-8 is the results of the third tested model (using the FPA method), and tables from Table 7-9 to Table 7-13 are the results of the fourth test for the selected algorithms. In each of these tables, the first five rows are the average of each fold, and the last line is the mean of these folds.

Table 7-8. Evaluation results of FPA on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	388.91	12.827	564.86	0.920	0.128	0.107
Communication	603.38	17.491	732.54	0.743	0.175	0.144
Financial	240.16	8.221	400.77	1.000	0.086	0.075
Government	474.22	7.106	977.49	1.000	0.076	0.065
Insurance	320.24	9.532	452.39	0.855	0.095	0.080
Manufacturing	240.62	17.313	349.26	0.778	0.176	0.124
Service Industry	956.21	20.232	1678.47	0.686	0.203	0.140

Others	452.51	11.959	623.16	0.967	0.121	0.099
mean	*459.53*	*13.085*	*722.37*	*0.868*	*0.132*	*0.104*

Table 7-9. Evaluation results of CFCW-MLR on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	215.58	6.809	309.79	1.00	0.070	0.063
Communication	319.16	11.603	406.55	0.86	0.121	0.102
Financial	199.40	6.596	315.68	1.00	0.066	0.060
Government	384.57	6.167	724.40	1.00	0.067	0.059
Insurance	231.88	6.894	342.94	0.95	0.071	0.063
Manufacturing	208.37	15.425	293.81	0.84	0.160	0.118
Service Industry	881.37	18.686	1519.62	0.74	0.190	0.134
Others	296.85	9.788	421.35	0.97	0.102	0.084
mean	*342.15*	*10.246*	*541.77*	*0.92*	*0.106*	*0.085*

Table 7-10. Evaluation results of CFCW-MLP on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	322.24	10.779	451.53	0.940	0.109	0.093
Communication	332.76	10.237	410.76	0.971	0.107	0.094
Financial	259.56	8.084	404.80	1.000	0.083	0.073
Government	550.89	8.162	1069.11	0.956	0.086	0.074
Insurance	302.25	8.728	449.25	0.855	0.088	0.074
Manufacturing	238.12	17.128	349.79	0.800	0.175	0.124
Service Industry	946.47	20.318	1670.77	0.686	0.205	0.140
Others	369.01	10.521	530.25	0.967	0.108	0.088
mean	*415.16*	*11.745*	*667.03*	*0.897*	*0.120*	*0.095*

Table 7-11. Evaluation results of CFCW-BRR on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	179.18	5.694	262.19	1.000	0.059	0.054
Communication	234.48	8.268	296.61	1.000	0.087	0.077
Financial	150.56	5.331	231.50	1.000	0.055	0.050
Government	207.89	4.685	278.03	1.000	0.052	0.046
Insurance	191.06	5.875	283.11	0.964	0.060	0.053
Manufacturing	207.15	15.961	314.13	0.822	0.162	0.114
Service Industry	878.57	18.831	1546.76	0.743	0.191	0.134
Others	318.32	9.314	421.70	0.967	0.097	0.083
Mean	*295.90*	*9.245*	*454.25*	*0.937*	*0.095*	*0.076*

Table 7-12. Evaluation results of CFCW-LAS on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	215.21	6.798	309.20	1.000	0.070	0.063
Communication	318.97	11.597	406.35	0.857	0.121	0.102
Financial	197.72	6.532	312.50	1.000	0.066	0.059
Government	382.73	6.158	719.18	1.000	0.067	0.059
Insurance	231.26	6.881	342.16	0.946	0.071	0.063
Manufacturing	208.30	15.421	293.78	0.844	0.160	0.118
Service Industry	881.15	18.683	1519.54	0.743	0.190	0.134
Others	296.46	9.784	419.87	0.967	0.101	0.084
mean	*341.47*	*10.232*	*540.32*	*0.920*	*0.106*	*0.085*

Table 7-13. Evaluation results of CFCW-SVR on all sectors of the segmented dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	213.22	6.615	322.23	1.000	0.067	0.061
Communication	336.60	11.492	424.14	0.914	0.127	0.105
Financial	174.44	6.041	272.79	1.000	0.062	0.056
Government	361.63	6.297	630.42	1.000	0.068	0.060
Insurance	269.23	7.549	371.74	0.946	0.077	0.068
Manufacturing	224.18	15.859	317.29	0.800	0.162	0.118
Service Industry	893.87	18.685	1557.15	0.743	0.188	0.133
Others	324.18	9.781	418.84	0.967	0.103	0.090
mean	*349.67*	*10.290*	*539.33*	*0.921*	*0.107*	*0.086*

Table 7-14 presents the results compiled from the above tables.

Table 7-14. Second experiment group mean values of all algorithms results

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
FPA	459.53	13.085	722.37	0.868	0.132	0.104
CFCW-MLR	342.15	10.246	541.77	0.920	0.106	0.085
CFCW-MLP	415.16	11.745	667.03	0.897	0.120	0.095
CFCW-BRR	295.90	9.245	454.25	0.937	0.095	0.076
CFCW-LAS	341.47	10.232	540.32	0.920	0.106	0.085
CFCW-SVR	349.67	10.290	539.33	0.921	0.107	0.086

As we can see, the CFCW model with the BRR algorithm gets the most accurate because it has the minimum estimation error (the evaluation result is maximum in the PRED (0.25) and minimum in other evaluation criteria). Fig. 7-2 shows us the visualization of the experiment results.

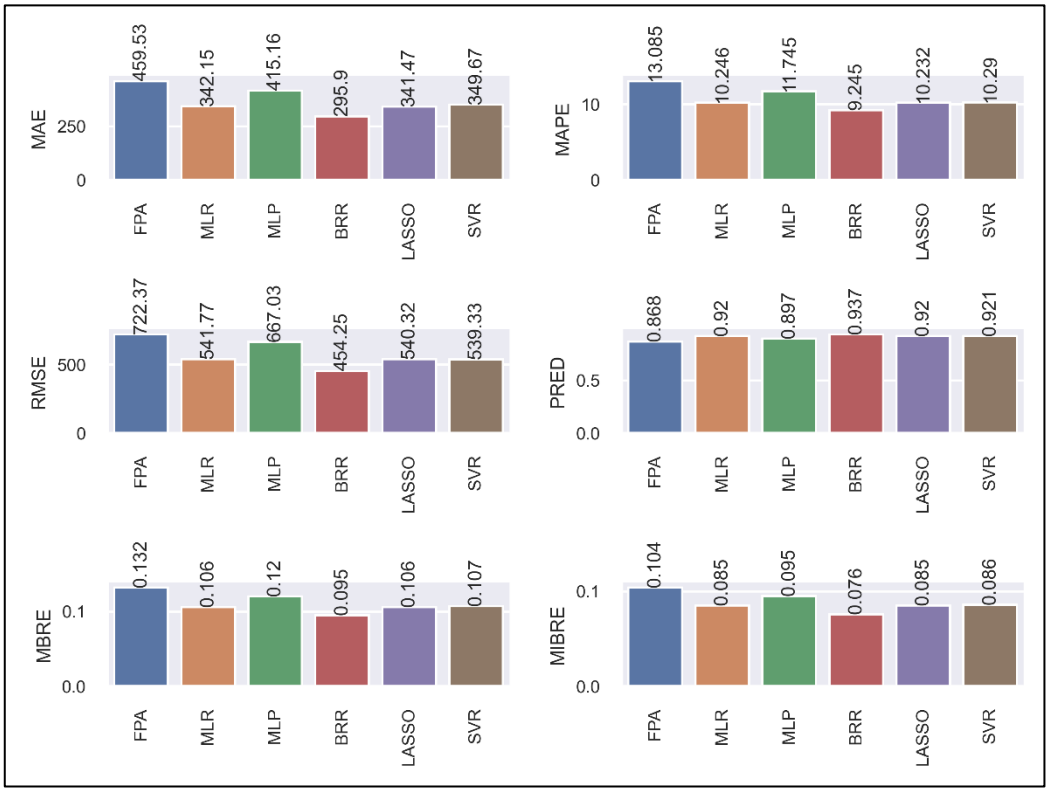


Fig. 7-2. Summary evaluation results

The pair-wise t-test technique is applied to ensure that the experiment results are statistically correct. The comparisons are concentrated on 1) Estimated accuracy using the FPA and CFCW methods on the entire unsegmented dataset and the segmented dataset based on the IS categorical variable, and 2) Estimated accuracy using six selected algorithms in the CFWC method on the whole unsegmented dataset and the segmented dataset based on the IS categorical variable. Two hypotheses of this situation are:

- H1: There is a significant difference in estimation capability between using the CFCW and the FPA methods on the entire unsegmented dataset and the segmented dataset based on the IS categorical variable. This statement implies that the estimation accuracy when using the CFCW approach differs significantly from that of the FPA manner on the whole unsegmented dataset and the segmented dataset based on the IS categorical variable.
- H2: There is a significant difference in estimation capability between algorithms used in the CFCW method on the entire unsegmented dataset and the segmented dataset based on the IS categorical variable. This implies that one algorithm will outperform the others when used in the CFCW method on the whole unsegmented dataset and the segmented dataset based on the IS categorical variable.

Table 7-15 and Table 7-16 show the pairwise t-test results of the first and the second groups. The notation ≫ reflects the statistical superiority of the CFCW

approach compared to the FPA method on the unsegmented dataset. The findings show that the CFCW approach, which uses both unsegmented and segmented datasets, is statistically significant compared to other methods at the 95% confidence level. As a result, we adopt hypothesis H1, which is compatible with the abovementioned results. This means that the CFCW model outperforms the FPA method on the dataset both with and without segmentation.

Table 7-15. The statistical t-test results on evaluation results of the CFCW with the FPA method on the unsegmented dataset

Pairs of methods		CFCW-MLR vs FPA	CFCW-MLP vs FPA	CFCW-BRR vs FPA	CFCW-LAS vs FPA	CFCW-SVR vs FPA
MAE results	Avg. MAE	325.876 vs 379.892	354.993 vs 379.892	317.8 vs 379.892	325.832 vs 379.892	351.261 vs 379.892
	Avg. p-value	1.19E-02	6.32E-03	6.91E-03	1.19E-02	3.28E-03
	Statistical conclusion	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	9.096 vs 10.524	9.917 vs 10.524	8.734 vs 10.524	9.094 vs 10.524	9.609 vs 10.524
	Avg. p-value	2.97E-03	4.38E-03	1.19E-03	2.96E-03	3.94E-04
	Statistical conclusion	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	509.504 vs 623.382	568.1 vs 623.382	493.49 vs 623.382	509.407 vs 623.382	558.924 vs 623.382
	Avg. p-value	2.64E-03	6.18E-05	1.93E-03	2.64E-03	2.60E-03
	Statistical conclusion	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.094 vs 0.106	0.101 vs 0.106	0.09 vs 0.106	0.094 vs 0.106	0.097 vs 0.106
	Avg. p-value	4.77E-03	6.74E-03	1.64E-03	4.75E-03	4.75E-04
	Statistical conclusion	>>	>>	>>	>>	>>

MIBRE results	Avg. MIBRE	0.079 vs 0.087	0.083 vs 0.087	0.076 vs 0.087	0.079 vs 0.087	0.082 vs 0.087
	Avg. p-value	1.17E-02	2.16E-02	3.80E-03	1.17E-02	1.48E-03
	Statistical conclusion	>>	>>	>>	>>	>>

Table 7-16. The statistical t-test result on evaluation results of the CFCW with the FPA method on the segmented dataset using the IS categorical variable

Pairs of methods		CFCW-MLR vs FPA	CFCW-MLP vs FPA	CFCW-BRR vs FPA	CFCW-LAS vs FPA	CFCW-SVR vs FPA
MAE results	Avg. MAE	342.15 vs 459.53	416.57 vs 459.53	295.90 vs 459.53	341.48 vs 459.53	349.67 vs 459.53
	Avg. p-value	2.18E-09	9.48E-03	2.05E-10	1.86E-09	1.84E-08
	Statistical conclusion	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	10.246 vs 13.085	11.946 vs 13.085	9.245 vs 13.085	10.232 vs 13.085	10.29 vs 13.085
	Avg. p-value	3.53E-09	5.61E-03	6.80E-10	3.02E-09	2.94E-09
	Statistical conclusion	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	541.77 vs 722.37	670.85 vs 722.37	454.25 vs 722.37	540.32 vs 722.37	539.33 vs 722.37
	Avg. p-value	2.81E-10	5.00E-03	7.13E-09	2.49E-10	1.35E-08
	Statistical conclusion	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.106 vs 0.132	0.123 vs 0.132	0.095 vs 0.132	0.106 vs 0.132	0.107 vs 0.132
	Avg. p-value	5.40E-09	1.46E-02	8.10E-10	4.65E-09	1.16E-09
	Statistical conclusion	>>	>>	>>	>>	>>
MIBRE results	Avg. MIBRE	0.085 vs 0.104	0.097 vs 0.104	0.076 vs 0.104	0.085 vs 0.104	0.086 vs 0.104

	Avg. p-value	3.72E-08	1.27E-02	1.62E-09	3.21E-08	5.08E-08
	Statistical conclusion	>>	>>	>>	>>	>>

Table 7-17 and Table 7-18 show the comparison results between the BRR and other algorithms. The notation ≫ reflects the statistical superiority of the BRR algorithm compared to other algorithms on both the unsegmented and segmented datasets. This confirms statistically that the BRR algorithm is the most suitable statistical significance algorithm with 95% confidence (hypothesis H2 is also accepted).

Table 7-17. The statistical t-test on evaluation results of the BRR algorithm with other algorithms on the CFCW method on the unsegmented dataset

Pairs of methods		CFCW-BRR vs CFCW-MLR	CFCW-BRR vs CFCW-MLP	CFCW-BRR vs CFCW-LAS	CFCW-BRR vs CFCW-SVR
MAE results	Avg. MAE	317.8 vs 325.876	317.8 vs 354.993	317.8 vs 325.832	317.8 vs 351.261
	Avg. p-value	1.06E-02	1.52E-02	1.06E-02	2.59E-02
	Statistical conclusion	>>	>>	>>	>>
MAPE results	Avg. MAPE	8.734 vs 9.096	8.734 vs 9.917	8.734 vs 9.094	8.734 vs 9.609
	Avg. p-value	2.24E-04	2.29E-04	2.24E-04	2.24E-04
	Statistical conclusion	>>	>>	>>	>>
RMSE results	Avg. RMSE	493.49 vs 509.50	493.49 vs 568.1	493.49 vs 509.40	493.49 vs 558.92
	Avg. p-value	7.17E-04	2.63E-04	7.19E-04	5.37E-04
	Statistical conclusion	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.09 vs 0.094	0.09 vs 0.101	0.09 vs 0.094	0.09 vs 0.097
	Avg. p-value	2.10E-04	2.10E-04	2.10E-04	2.10E-04
	Statistical conclusion	>>	>>	>>	>>
MIBRE results	Avg. MIBRE	0.076 vs 0.079	0.076 vs 0.083	0.076 vs 0.079	0.076 vs 0.082
	Avg. p-value	2.10E-04	2.10E-04	2.10E-04	2.09E-04
	Statistical conclusion	>>	>>	>>	>>

Table 7-18. The statistical t-test on evaluation results of the BRR algorithm with other algorithms on the CFCW method on the segmented dataset using the IS categorical variable

Pairs of methods		CFCW-BRR vs CFCW-MLR	CFCW-BRR vs CFCW-MLP	CFCW-BRR vs CFCW-LAS	CFCW-BRR vs CFCW-SVR
MAE results	Avg. MAE	295.90 vs 342.15	295.90 vs 416.57	295.90 vs 341.48	295.90 vs 349.67
	Avg. p-value	2.90E-04	2.67E-07	3.27E-04	1.51E-05
	Statistical conclusion	>>	>>	>>	>>
MAPE results	Avg. MAPE	9.245 vs 10.246	9.245 vs 11.946	9.245 vs 10.232	9.245 vs 10.29
	Avg. p-value	7.76E-05	3.22E-11	9.09E-05	1.10E-05
	Statistical conclusion	>>	>>	>>	>>
RMSE results	Avg. RMSE	454.25 vs 541.77	454.25 vs 670.85	454.25 vs 540.32	454.25 vs 539.32
	Avg. p-value	2.90E-03	2.15E-06	3.14E-03	1.02E-03
	Statistical conclusion	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.095 vs 0.106	0.095 vs 0.123	0.095 vs 0.106	0.095 vs 0.107
	Avg. p-value	3.15E-05	1.92E-10	3.77E-05	2.98E-05
	Statistical conclusion	>>	>>	>>	>>
MIBRE results	Avg. MIBRE	0.076 vs 0.085	0.076 vs 0.097	0.076 vs 0.085	0.076 vs 0.086
	Avg. p-value	5.15E-06	1.43E-09	6.24E-06	3.49E-07
	Statistical conclusion	>>	>>	>>	>>

7.1.3 Summary

We can easily see that the BRR algorithm achieves the most optimal results with four tests in this group. This assertion is made based on the evaluation criteria in section 6.5. In all tests, the BRR algorithm consistently achieves the lowest estimation error. To conclude this section, we can confirm that the BRR algorithm is the most suitable algorithm for this study.

7.2 Finding the best suitable clustering criterion

This section presents the results of finding the best segmentation criteria. The finding for the best suitable segmentation criterion is based on two main directions: segmentation by categorical variables and segmentation by clustering algorithms.

7.2.1 Using categorical variables

With the first tested model, applying the FPA method to the entire dataset without segmentation, the experimental results are shown in Table 7-19. The first five lines are the results of runs based on the folds of k-fold cross-validation. The last line is the mean of five runs over five folds.

Table 7-19. FPA with the entire non-clustered dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	579.46	14.38	1294.07	0.816	0.146	0.103
Ex 2	588.86	14.339	1305.18	0.805	0.145	0.102
Ex 3	572.42	13.281	1308.24	0.839	0.134	0.094
Ex 4	579.38	12.815	1322.90	0.851	0.13	0.101
Ex 5	624.73	15.308	1356.72	0.816	0.155	0.106
mean	*588.97*	*14.025*	*1317.42*	*0.825*	*0.142*	*0.101*

The second tested model is similar to the first tested model with the difference that the CFCW method is used instead of the IFPUG FPA. The results of this experiment are shown in Table 7-20.

Table 7-20. CFCW with the entire non-clustered dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	518.02	13.225	1136.31	0.897	0.138	0.102
Ex 2	518.27	13.271	1132.10	0.885	0.137	0.101
Ex 3	547.42	12.947	1205.92	0.874	0.133	0.098
Ex 4	525.43	11.354	1179.33	0.92	0.118	0.095
Ex 5	565.27	14.301	1197.26	0.874	0.147	0.105
mean	*534.88*	*13.02*	*1170.19*	*0.89*	*0.135*	*0.1*

For categorical variables, DP, IS, LT, OT, and RS, the experimental results of the 3rd and 4th models are presented in 2 tables, respectively. The first table shows the results of the FPA method, and the second table shows the results of the CFCW method applied to the same categorical variable. In each table, the first

lines refer to the subgroups within the categorical variable (these subgroups can be seen in section 6.3.1), and the last line indicates the mean of the rows. In each of these subgroups, the value shown in the table is the average of the five runs corresponding to a 5-fold cross-validation.

Table 7-21. The estimation results of the FPA method on the DP categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
MF	557.05	11.167	918.75	0.843	0.113	0.093
MR	487.58	8.687	880.23	0.892	0.087	0.073
Multi	275.65	10.150	444.24	0.929	0.101	0.073
PC	272.62	10.816	360.84	0.907	0.109	0.091
Others	217.23	11.331	315.53	0.960	0.113	0.097
mean	*362.02*	*10.430*	*583.92*	*0.906*	*0.105*	*0.085*

Table 7-22. The estimation results of the CFCW method on the DP categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
MF	464.39	10.026	734.801	0.913	0.105	0.089
MR	399.75	7.188	572.052	0.969	0.075	0.066
Multi	234.00	9.105	388.645	0.929	0.091	0.065
PC	185.21	7.620	246.666	0.973	0.079	0.069
Others	88.17	6.717	110.04	1.000	0.077	0.065
mean	*274.30*	*8.131*	*410.441*	*0.957*	*0.085*	*0.071*

Table 7-23. The estimation results of the FPA method on the IS categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	235.19	7.901	364.04	1.000	0.079	0.067
Communication	345.85	12.194	503.12	0.880	0.122	0.101
Financial	169.44	7.481	260.83	1.000	0.079	0.067
Government	582.35	8.343	1364.58	0.917	0.089	0.069
Insurance	355.01	9.592	594.03	0.892	0.096	0.081
Manufacturing	168.95	9.690	293.06	0.873	0.098	0.076
Service Industry	255.23	5.437	380.36	0.978	0.055	0.049
Others	254.78	7.882	384.98	0.914	0.079	0.068
mean	*288.37*	*8.489*	*478.53*	*0.950*	*0.086*	*0.073*

Table 7-24. The estimation results of the CFCW method on the IS categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	194.45	5.974	269.17	0.980	0.063	0.057
Communication	193.87	8.306	269.36	0.960	0.093	0.080
Financial	136.00	6.844	190.63	1.000	0.071	0.062
Government	492.71	7.587	1108.54	0.917	0.083	0.066
Insurance	277.85	8.347	403.01	1.000	0.089	0.078
Manufacturing	138.01	8.450	244.49	0.927	0.087	0.068
Service Industry	209.20	4.605	278.04	1.000	0.048	0.044
Utilities	224.73	7.353	332.07	1.000	0.079	0.070
mean	*204.63*	*6.768*	*327.40*	*0.980*	*0.072*	*0.062*

Table 7-25. The estimation results of the FPA method on the LT categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
3GL	452.58	10.636	755.78	0.837	0.107	0.088
4GL	251.04	7.970	427.18	0.973	0.081	0.068
Others	395.76	10.682	556.77	0.880	0.108	0.092
mean	366.46	9.763	579.91	0.897	0.099	0.083

Table 7-26. The estimation results of the CFCW method on the LT categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
3GL	380.36	9.160	592.44	0.940	0.094	0.082
4GL	229.39	7.194	372.67	0.973	0.075	0.064
Others	291.33	9.189	361.69	1.000	0.096	0.083
mean	*300.36*	*8.514*	*442.27*	*0.971*	*0.088*	*0.076*

Table 7-27. The estimation results of the FPA method on the OT categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	456.74	8.759	857.81	0.909	0.088	0.074
Communications	352.83	11.518	517.52	0.911	0.115	0.098
Electricity, Gas, Water	357.28	8.708	513.26	1.000	0.092	0.081
Financial, Property & Business Services	260.48	8.605	377.86	1.000	0.093	0.079
Government	438.16	7.023	683.77	1.000	0.072	0.064
Insurance	388.03	10.989	569.74	0.846	0.110	0.092
Manufacturing	146.16	8.449	230.09	0.900	0.086	0.065

Public Administration	250.65	8.379	369.29	1.000	0.088	0.079
Others	316.36	7.310	672.91	0.924	0.073	0.064
mean	*329.63*	*8.860*	*532.47*	*0.943*	*0.091*	*0.077*

Table 7-28. The estimation results of the CFCW method on the OT categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Banking	326.01	6.546	541.24	1.000	0.068	0.061
Communications	199.08	8.243	267.23	1.000	0.093	0.081
Electricity, Gas, Water	158.93	5.514	220.59	1.000	0.060	0.054
Financial, Property & Business Services	233.91	8.008	304.42	1.000	0.091	0.078
Government	417.66	6.674	640.48	1.000	0.068	0.060
Insurance	323.91	9.738	478.06	0.923	0.103	0.087
Manufacturing	113.27	7.326	172.98	0.900	0.075	0.059
Public Administration	177.82	6.776	232.12	1.000	0.070	0.064
Others	258.01	6.173	434.86	1.000	0.063	0.057
mean	*245.40*	*7.222*	*365.78*	*0.980*	*0.077*	*0.067*

Table 7-29. The estimation results of the FPA method on the RS categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
S	362.02	10.430	583.92	0.906	0.105	0.085
M1	288.37	8.489	478.53	0.950	0.086	0.073
M2	366.46	9.763	579.91	0.897	0.099	0.083
L	329.63	8.860	532.47	0.943	0.091	0.077
mean	*350.75*	*9.168*	*536.74*	*0.917*	*0.094*	*0.078*

Table 7-30. The estimation results of the CFCW method on the RS categorical variable

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
S	274.30	8.131	410.44	0.957	0.085	0.071
M1	204.63	6.768	327.40	0.980	0.072	0.062
M2	300.36	8.514	442.27	0.971	0.088	0.076
L	245.40	7.222	365.78	0.980	0.077	0.067
mean	*290.95*	*7.970*	*415.98*	*0.934*	*0.084*	*0.072*

For an overview, Table 7-31 and Table 7-32 are used to represent the combined results of FPA and CFCW methods on the categorical variables applied in this study. Each row in this table is the mean of all subgroups in each segment variable.

Table 7-31. The estimation results of the FPA method on all categorical variables

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
DP	362.02	10.430	583.92	0.906	0.105	0.085
IS	288.37	8.489	478.53	0.950	0.086	0.073
LT	366.46	9.763	579.91	0.897	0.099	0.083
OT	329.63	8.860	532.47	0.943	0.091	0.077
RS	350.75	9.168	536.74	0.917	0.094	0.078

Table 7-32. The estimation results of the CFCW method on all categorical variables

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
DP	274.30	8.131	410.44	0.957	0.085	0.071
IS	204.63	6.768	327.40	0.980	0.072	0.062
LT	300.36	8.514	442.27	0.971	0.088	0.076
OT	245.40	7.222	365.78	0.980	0.077	0.067
RS	290.95	7.970	415.98	0.934	0.084	0.072

Accordingly, we can easily see that the IS categorical variable criterion achieves the slightest estimation error on the FPA and CFCW methods. From this point, we can conclude that the categorical variable "IS" is the most suitable segmentation criterion among all the assessed categorical variables.

The pair-wise t-test is performed with the two following hypotheses to make sure the experiment's statistics are correct.

- H1: There is a significant difference in estimation capability between using the FPA method on the dataset with and without clustering. This statement implies that the FPA method estimation accuracy on the dataset with clustering differs significantly from that without clustering.
- H2: There is a significant difference in estimation capability between using the CFCW and FPA methods on the dataset with and without clustering. This statement implies that the CFCW method estimation accuracy differs significantly from the FPA method on the same dataset with and without clustering.

Table 7-33 is the result of this t-test based on evaluation results for the FPA method on the whole dataset without clustering with each cluster. The notation ≫reflects the statistical superiority of the FPA approach on clustered datasets compared to the same method on the dataset without clustering. The results

confirm that the FPA method based on clustering is statistically significant at the 95% confidence level compared to without clustering. Therefore, the H1 hypothesis is accepted, consistent with the results presented above. It means that there is a significant improvement when using the FPA method with clustering data compared to without clustering.

Table 7-33. The statistical t-test based on evaluation results for the FPA method on the whole dataset without clustering with each cluster

Pairs of methods		DP vs WC	IS vs WC	LT vs WC	OT vs WC	RS vs WC	k-means vs WC
MAE results	Avg. MAE	362.022 vs 588.971	288.367 vs 588.971	366.46 vs 588.971	329.631 vs 588.971	350.747 vs 588.971	315.691 vs 588.971
	Avg. p-value	0.00000	0.00000	0.00047	0.00002	0.00002	0.00001
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	10.43 vs 14.025	8.489 vs 14.025	9.763 vs 14.025	8.86 vs 14.025	9.168 vs 14.025	9.138 vs 14.025
	Avg. p-value	0.00089	0.00035	0.00086	0.00007	0.00149	0.00045
	Statistical conclusion	>>	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	583.917 vs 1317.422	478.53 vs 1317.422	579.911 vs 1317.422	532.472 vs 1317.422	536.738 vs 1317.422	493.161 vs 1317.422
	Avg. p-value	0.00000	0.00000	0.00001	0.00001	0.00000	0.00000
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.105 vs 0.142	0.086 vs 0.142	0.099 vs 0.142	0.091 vs 0.142	0.093 vs 0.142	0.092 vs 0.142
	Avg. p-value	0.00085	0.00039	0.00078	0.00006	0.00164	0.00041
	Statistical conclusion	>>	>>	>>	>>	>>	>>

MIBRE results	Avg. MIBRE	0.00289	0.073 vs 0.101	0.083 vs 0.101	0.077 vs 0.101	0.078 vs 0.101	0.076 vs 0.101
	Avg. p-value	0.00289	0.00033	0.00158	0.00015	0.00214	0.00025
	Statistical conclusion	>>	>>	>>	>>	>>	>>

Table 7-34 shows the results of the statistical pairwise t-test of FPA and CFCW methods on each cluster. The results confirm that the CFCW method based on the cluster is statistically significant at the 95% confidence level compared to the FPA method. Therefore, we accept hypothesis H2, which is consistent with the results presented above. It means that the CFCW model outperforms the FPA model on the dataset with clustering.

Table 7-34. The statistical t-test based on evaluation results for the CFCW and FPA methods on the dataset clustering with each cluster

Pairs of methods		DP	IS	LT	OT	RS
		CFCW vs FPA	CFCW vs FPA	CFCW vs FPA	CFCW vs FPA	CFCW vs FPA
MAE results	Avg. MAE	274.303 vs 362.022	204.631 vs 288.367	300.357 vs 366.46	245.401 vs 329.631	290.95 vs 350.747
	Avg. p-value	0.00000	0.00000	0.00006	0.00000	0.00001
	Statistical conclusion	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	8.131 vs 10.43	6.768 vs 8.489	8.514 vs 9.763	7.222 vs 8.86	7.97 vs 9.168
	Avg. p-value	0.00000	0.00000	0.00001	0.00000	0.00000
	Statistical conclusion	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	410.441 vs 583.917	327.403 vs 478.53	442.266 vs 579.911	365.776 vs 532.472	415.983 vs 536.738
	Avg. p-value	0.00000	0.00000	0.00002	0.00000	0.00001
	Conclusion	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.085 vs 0.105	0.072 vs 0.086	0.088 vs 0.099	0.077 vs 0.091	0.083 vs 0.093
	Avg. p-value	0.00000	0.00000	0.00005	0.00000	0.00000

	Statistical conclusion	>>	>>	>>	>>	>>
MIBRE results	Avg. MIBRE	0.071 vs 0.085	0.062 vs 0.073	0.076 vs 0.083	0.067 vs 0.077	0.072 vs 0.078
	Avg. p-value	0.00001	0.00000	0.00035	0.00000	0.00000
	Statistical conclusion	>>	>>	>>	>>	>>

7.2.2 Using segmentation algorithms

The experiment's segmentation algorithm findings are presented in this section. There are four tested models corresponding to each segmentation algorithm. With the first test model (IFPUG FPA on the entire non-clustered dataset). The results obtained corresponding to this model are shown in Table 7-35.

Table 7-35. FPA method on the whole dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	579.46	14.380	1294.07	0.816	0.146	0.103
Ex 2	588.86	14.339	1305.18	0.805	0.145	0.102
Ex 3	572.42	13.281	1308.24	0.839	0.134	0.094
Ex 4	579.38	12.815	1322.90	0.851	0.130	0.101
Ex 5	624.73	15.308	1356.72	0.816	0.155	0.106
mean	*588.97*	*14.025*	*1317.42*	*0.825*	*0.142*	*0.101*

With the second tested model (CFCW over the entire dataset), the obtained results corresponding to this model are shown in Table 7-36.

Table 7-36. CFCW method on the whole dataset

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Ex 1	518.02	13.225	1136.31	0.897	0.138	0.102
Ex 2	518.27	13.271	1132.10	0.885	0.137	0.101
Ex 3	547.42	12.947	1205.92	0.874	0.133	0.098
Ex 4	525.43	11.354	1179.33	0.920	0.118	0.095
Ex 5	565.27	14.301	1197.26	0.874	0.147	0.105
mean	*534.88*	*13.020*	*1170.19*	*0.890*	*0.135*	*0.100*

Fig. 7-3 shows the evaluation result of the first and second tested models' evaluation results in visual form. As we can see, the orange column (CFCW) always gets a better result than the blue (FPA) in all evaluation criteria.

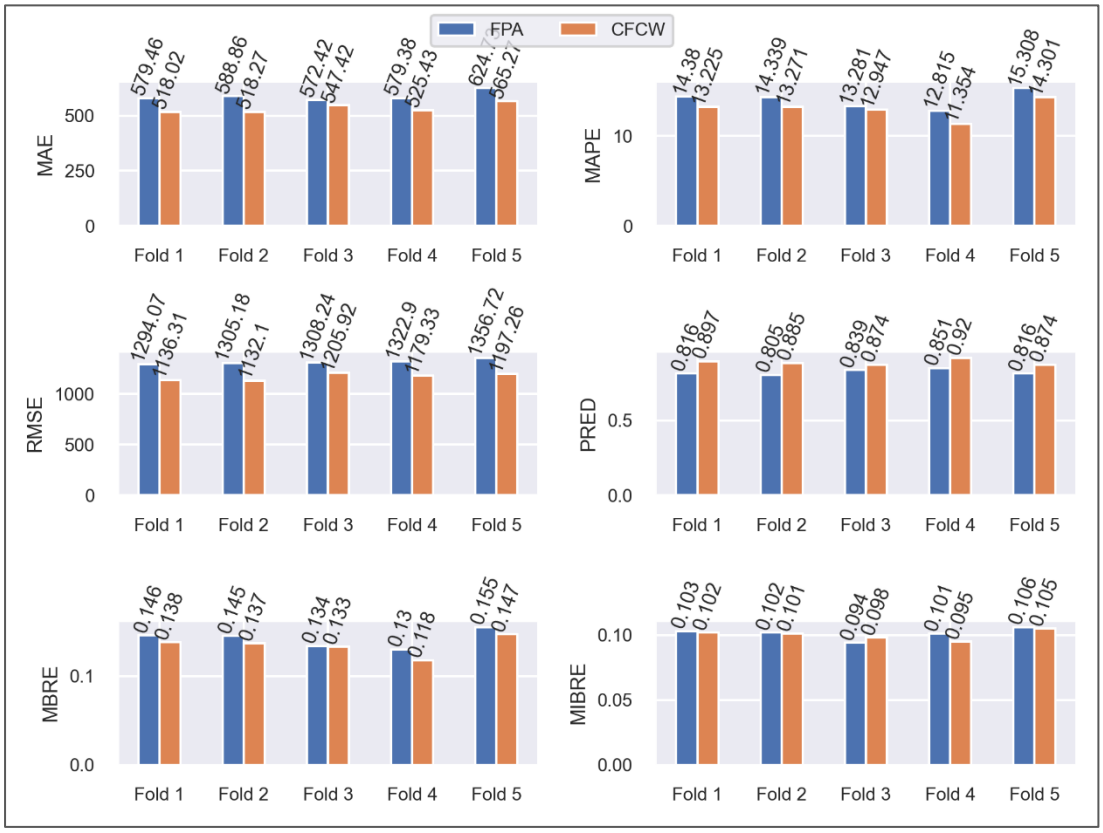


Fig. 7-3. The evaluation result of the first tested model

The results of the third tested model (FPA method on clusters formed by specific clustering algorithm) and the fourth experimental model (CFCW method on clusters formed by specific clustering algorithm) are listed in tables Table 7-37 - Table 7-48. Tables Table 7-37, Table 7-39, Table 7-41, Table 7-43, Table 7-45, and Table 7-47 are the results of the 3rd model, and the remaining tables are the results of the 4th model for all selected segmentation algorithms.

Table 7-37. FPA method on clusters formed by BIRCH clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	358.64	10.658	662.13	0.879	0.107	0.088
Cluster 2	355.69	6.928	664.74	0.929	0.069	0.059
Cluster 3	441.75	8.691	754.89	0.914	0.088	0.076
Cluster 4	337.32	10.990	545.78	0.831	0.112	0.093
mean	*373.35*	*9.317*	*656.88*	*0.888*	*0.094*	*0.079*

Table 7-38. CFCW method on clusters formed by BIRCH clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	319.80	9.546	543.59	0.950	0.097	0.083
Cluster 2	300.21	5.953	479.76	1.000	0.061	0.054
Cluster 3	388.03	7.595	663.70	0.986	0.079	0.068
Cluster 4	288.92	9.364	420.82	0.969	0.099	0.085
mean	*324.24*	*8.115*	*526.97*	*0.976*	*0.084*	*0.073*

Table 7-39. FPA method on clusters formed by FCM clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	338.43	7.653	583.93	0.917	0.077	0.066
Cluster 2	370.80	10.772	637.33	0.860	0.108	0.089
Cluster 3	425.12	8.219	825.96	0.900	0.082	0.069
mean	*378.12*	*8.881*	*682.41*	*0.892*	*0.089*	*0.075*

Table 7-40. CFCW method on clusters formed by FCM clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	306.65	6.975	482.41	0.975	0.072	0.063
Cluster 2	310.86	8.993	503.27	0.960	0.092	0.079
Cluster 3	321.04	6.613	597.81	0.962	0.068	0.059
mean	*312.85*	*7.527*	*527.83*	*0.966*	*0.077*	*0.067*

Table 7-41. FPA method on clusters formed by GMM clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	412.28	7.639	777.16	0.907	0.076	0.065
Cluster 2	316.88	9.715	559.06	0.897	0.098	0.082
Cluster 3	378.27	8.591	634.82	0.887	0.086	0.073
mean	*369.14*	*8.648*	*657.01*	*0.897*	*0.087*	*0.073*

Table 7-42. CFCW method on clusters formed by GMM clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	330.81	6.097	580.44	0.973	0.062	0.055
Cluster 2	295.93	8.783	485.26	0.981	0.090	0.078
Cluster 3	338.94	7.530	539.01	0.974	0.077	0.067
mean	*321.90*	*7.470*	*534.90*	*0.976*	*0.076*	*0.067*

Table 7-43. FPA method on clusters formed by the k-means clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	435.527	7.706	1029.986	0.911	0.077	0.066
Cluster 2	289.419	11.363	551.85	0.82	0.114	0.092
Cluster 3	389.263	9.049	732.182	0.831	0.091	0.071
mean	*371.403*	*9.373*	*771.339*	*0.854*	*0.094*	*0.076*

Table 7-44. CFCW method on clusters formed by the k-means clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	368.154	6.454	747.31	1	0.067	0.06
Cluster 2	253.881	10.351	455.637	0.92	0.106	0.088
Cluster 3	329.455	7.802	609.307	0.923	0.08	0.067
mean	*317.163*	*8.202*	*604.085*	*0.948*	*0.084*	*0.072*

Table 7-45. FPA method on clusters formed by MeanShift clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	364.00	11.367	609.53	0.867	0.114	0.092
Cluster 2	361.83	8.020	628.72	0.912	0.081	0.069
Cluster 3	421.16	12.395	604.49	0.840	0.127	0.106
Cluster 4	362.24	8.741	692.05	0.873	0.088	0.073
Cluster 5	374.05	7.977	599.79	0.889	0.080	0.066
mean	*376.66*	*9.700*	*626.92*	*0.876*	*0.098*	*0.081*

Table 7-46. CFCW method on clusters formed by MeanShift clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	310.05	9.888	486.63	0.967	0.101	0.085
Cluster 2	306.46	7.029	504.35	0.975	0.072	0.063
Cluster 3	362.66	10.850	490.32	0.960	0.113	0.096
Cluster 4	299.32	7.479	510.57	0.964	0.076	0.066
Cluster 5	328.04	6.552	478.99	1.000	0.067	0.059
mean	*321.30*	*8.360*	*494.17*	*0.973*	*0.086*	*0.074*

Table 7-47. FPA method on clusters formed by Spectral clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	372.58	10.635	658.87	0.885	0.108	0.090
Cluster 2	390.75	9.001	637.08	0.918	0.091	0.077
Cluster 3	425.04	8.423	778.09	0.886	0.084	0.070
mean	*396.12*	*9.353*	*691.35*	*0.896*	*0.094*	*0.079*

Table 7-48. CFCW method on clusters formed by Spectral clustering algorithm

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Cluster 1	347.38	9.460	601.77	0.945	0.098	0.085
Cluster 2	342.18	8.136	535.68	0.973	0.083	0.073
Cluster 3	371.78	7.406	602.02	0.957	0.075	0.066
mean	*353.78*	*8.334*	*579.83*	*0.958*	*0.085*	*0.075*

The detailed results above were summed up by taking the mean row across all algorithms. Then the results of FPA and CFCW methods on the algorithms are shown in Table 7-49 and Table 7-50, respectively.

Table 7-49. FPA method on clusters formed by clustering algorithms

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Non-clustered	588.97	14.025	1317.42	0.825	0.142	0.101
BIRCH	373.35	9.317	656.88	0.888	0.094	0.079
FCM	378.12	8.881	682.41	0.892	0.089	0.075
GMM	369.14	8.648	657.01	0.897	0.087	0.073
k-means	371.40	9.373	771.34	0.854	0.094	0.076
MeanShift	376.66	9.700	626.92	0.876	0.098	0.081
Spectral	396.12	9.353	691.35	0.896	0.094	0.079

Table 7-50. CFCW method on clusters formed by clustering algorithms

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Non-clustered	534.88	13.020	1170.19	0.890	0.135	0.100
BIRCH	324.24	8.115	526.97	0.976	0.084	0.073
FCM	312.85	7.527	527.83	0.966	0.077	0.067
GMM	321.90	7.470	534.90	0.976	0.076	0.067
k-means	317.16	8.202	604.09	0.948	0.084	0.072
MeanShift	321.30	8.360	494.17	0.973	0.086	0.074
Spectral	353.78	8.334	579.83	0.958	0.085	0.075

Fig. 7-4 and Fig. 7-5 give us a visual overview of the presented results. It is easy to see that whether using FPA or CFCW method, the estimated error value using segmentation algorithms is always smaller than the without-segmented (WS) on all evaluation criteria.

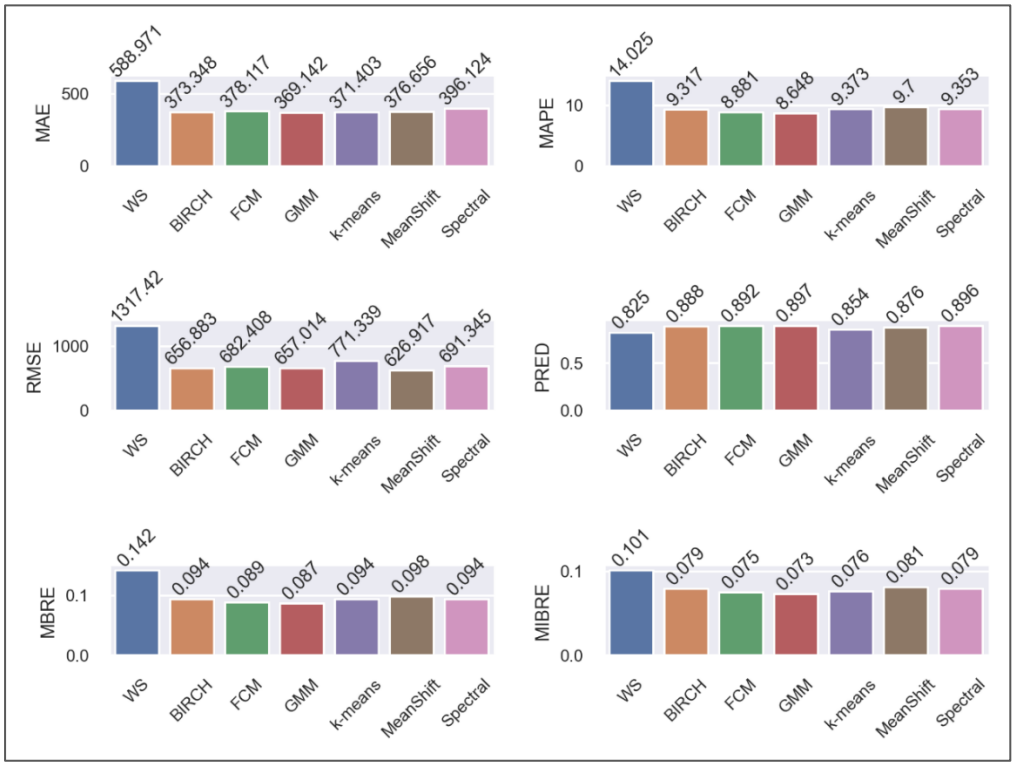


Fig. 7-4. FPA method results in clusters formed by clustering algorithms

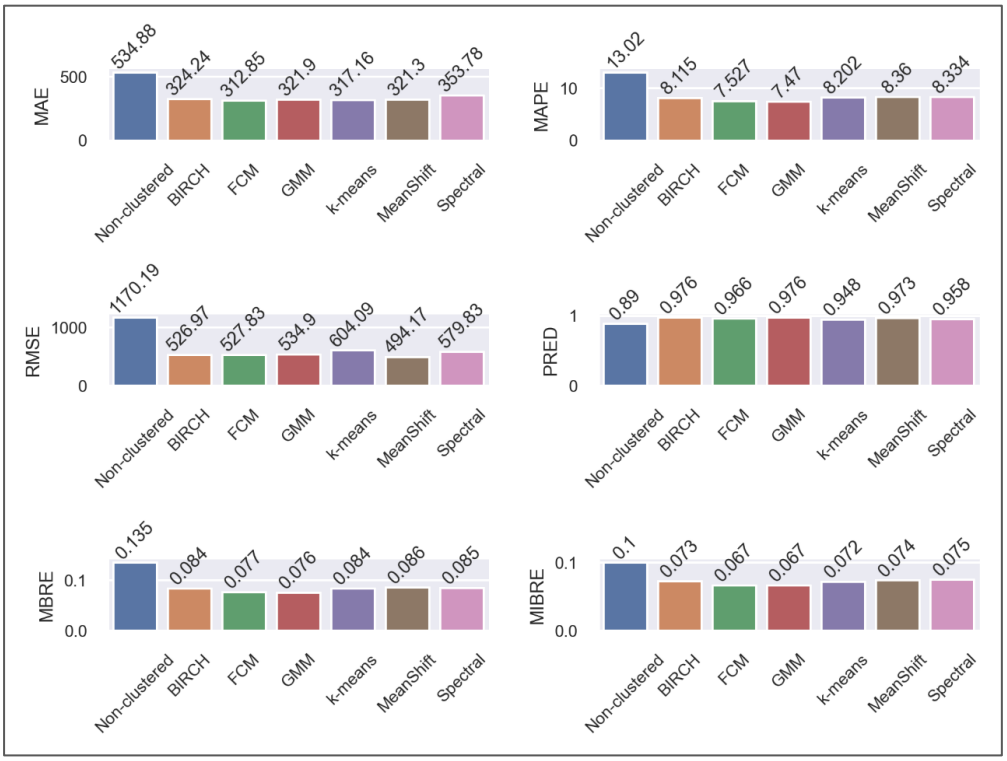


Fig. 7-5. CFCW method results in clusters formed by clustering algorithms

Fig. 7-6 gives an overall view of all comparisons on all evaluation criteria: 1) between FPA and CFCW methods and 2) between all selected algorithms.

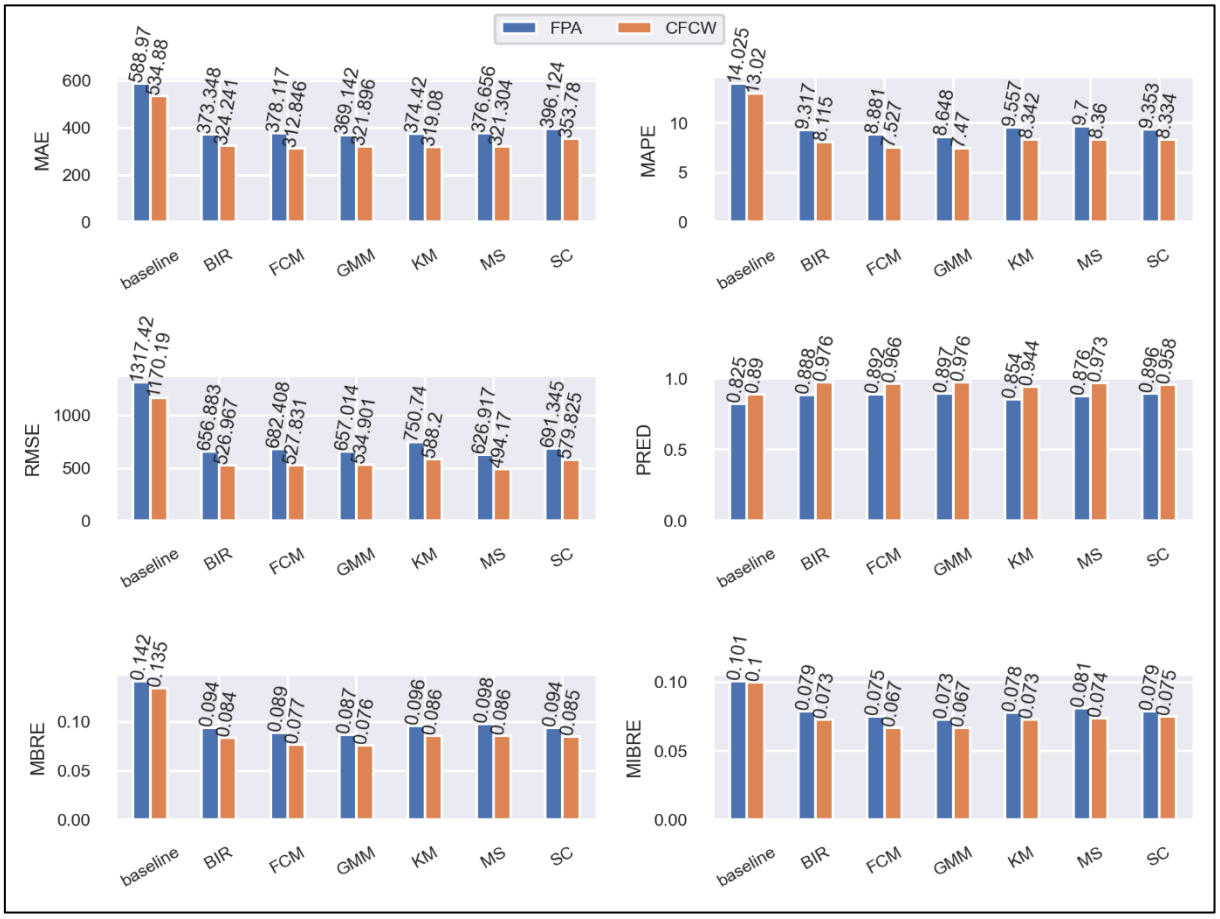


Fig. 7-6. Evaluation results of the FPA and CFCW-CA methods on clusters using the clustering algorithms

Corresponding to all algorithms, we consistently found that 1) applying clustering algorithms will give estimation accuracy better results than not applying, and 2) applying the CFCW method gives estimation accuracy better results than the FPA method.

To ensure the statistical correctness of the statement that applying clustering algorithms will give estimation accuracy better results than not applying, we performed a pairwise t-test with the following hypothesis:

- There is a significant difference in estimation capability between using the FPA and CFCW methods on non-clustering and clustering datasets using clustering algorithms. This statement means that the estimation accuracy of the FPA and CFCW methods on the clustered dataset is significantly different from that on the non-clustered dataset.

The notation FPA with the abbreviation of clustering algorithm refers to the result using the FPA method on the dataset that clustered using the specific clustering algorithm. For example, FPA-FCM means applying the FPA method to the dataset that is clustered using the FCM clustering algorithm. Similarly, the same meaning for the notation CFCW with the algorithm‘s abbreviation postfix is defined.

Table 7-51 and Table 7-52 show the pairwise t-test statistical result for FPA and CFCW methods. Accordingly, the p-values are always less than 0.05 on all evaluation criteria and the algorithms used in this study. It means that on datasets clustered using selected clustering algorithms, the FPA and CFWC methods get better-estimated accuracy than applied to the non-clustered dataset. According to this result, the hypothesis is accepted.

Table 7-51. The statistical t-test between the FPA method on the entire unsegmented dataset and the segmented dataset

Pairs of methods		FPA-BIR vs FPA	FPA-FCM vs FPA	FPA-GMM vs FPA	FPA-KM vs FPA	FPA-MS vs FPA	FPA-SC vs FPA
MAE results	Avg. MAE	373.348 vs 588.971	378.117 vs 588.971	369.142 vs 588.971	371.403 vs 588.971	376.656 vs. 588.971	396.125 vs. 588.971
	Avg. p-value	0.00001	0.00025	0.00110	0.00541	0.00002	0.00176
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	9.317 vs 14.025	8.881 vs 14.025	8.648 vs 14.025	9.373 vs 14.025	9.7 vs. 14.025	9.353 vs. 14.025
	Avg. p-value	0.00030	0.00024	0.00063	0.00134	0.00030	0.00005
	Statistical conclusion	>>	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	656.883 vs 1317.422	682.408 vs 1317.422	657.015 vs 1317.422	771.339 vs 1317.422	626.917 vs. 1317.422	691.345 vs. 1317.422
	Avg. p-value	0.00000	0.00002	0.00025	0.00198	0.00002	0.00038
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.094 vs 0.142	0.089 vs 0.142	0.087 vs 0.142	0.076 vs 0.142	0.098 vs. 0.142	0.094 vs. 0.142
	Avg. p-value	0.00031	0.00024	0.00058	0.00026	0.00027	0.00004
	Statistical conclusion	>>	>>	>>	>>	>>	>>

MIBRE results	Avg. MIBRE	0.079 vs 0.101	0.075 vs 0.101	0.073 vs 0.101	0.094 vs 0.101	0.081 vs. 0.101	0.079 vs. 0.101
	Avg. p-value	0.00097	0.00051	0.00063	0.08483	0.00357	0.00081
	Statistical conclusion	>>	>>	>>	<<	>>	>>

Table 7-52. The statistical t-test between the CFCWCA method on the entire unsegmented dataset and the segmented dataset

Pairs of methods		CFCW-BIR vs CFCW	CFCW-FCM vs CFCW	CFCW-GMM vs CFCW	CFCW-KM vs CFCW	CFCW-MS vs CFCW	CFCW-SC vs CFCW
MAE results	Avg. MAE	324.241 vs 534.882	312.846 vs 534.882	321.896 vs 534.882	317.163 vs 534.882	275.542 vs. 534.882	353.78 vs. 534.882
	Avg. p-value	0.00002	0.00003	0.00128	0.00473	0.00012	0.00283
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	8.114 vs 13.02	7.527 vs 13.02	7.47 vs 13.02	8.202 vs 13.02	6.493 vs. 13.02	8.334 vs. 13.02
	Avg. p-value	0.00033	0.00018	0.00034	0.00141	0.00059	0.00009
	Statistical conclusion	>>	>>	>>	>>	>>	>>
RMSE results	Avg. RMSE	526.967 vs 1170.185	527.831 vs 1170.185	534.901 vs 1170.185	604.085 vs 1170.185	459.835 vs. 1170.185	579.825 vs. 1170.185
	Avg. p-value	0.00000	0.00001	0.00031	0.00171	0.00002	0.00063
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MBRE results	Avg. MBRE	0.084 vs 0.135	0.077 vs 0.135	0.076 vs 0.135	0.072 vs 0.135	0.066 vs. 0.135	0.086 vs. 0.135
	Avg. p-value	0.00037	0.00015	0.00027	0.00041	0.00047	0.00006

	Statistical conclusion	>>	>>	>>	>>	>>	>>
MIBRE results	Avg. MIBRE	0.073 vs 0.1	0.067 vs 0.1	0.067 vs 0.1	0.082 vs 0.1	0.058 vs. 0.1	0.074 vs. 0.1
	Avg. p-value	0.00041	0.00007	0.00020	0.01213	0.00058	0.00014
	Statistical conclusion	>>	>>	>>	>>	>>	>>

To confine which algorithm is the best suitable for the evaluated dataset, a ranking table is created with the rating of each algorithm according to each evaluation criterion. A mean value of the evaluation criteria (EC) will also be determined, then considering the ranking position of each algorithm.

Table 7-53. The rank of algorithms with the FPA method

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE	mean of all EC	rank
WS	7	7	7	7	7	7	7.00	7
BIRCH	3	3	2	4	3	4	3.17	3
FCM	5	2	4	3	2	2	3.00	2
GMM	1	1	3	1	1	1	1.33	1
k-means	2	5	6	6	3	3	4.17	5
MeanShift	4	6	1	5	6	6	4.67	6
Spectral	6	4	5	2	3	4	4.00	4

Table 7-54. The rank of algorithms with the CFCW method

	MAE	MAPE	RMSE	PRED	MBRE	MIBRE	mean of all EC	rank
WS	7	7	7	7	7	7	7.00	7
BIRCH	5	3	2	1.5	3	4	3.08	3
FCM	1	2	3	4	2	1	2.17	2
GMM	4	1	4	1.5	1	1	2.08	1
k-means	2	4	6	6	3	3	4.00	4
MeanShift	3	6	1	3	6	5	4.00	4
Spectral	6	5	5	5	5	6	5.33	6

The results of the ranking process are presented in Table 7-53 and Table 7-54. Accordingly, with both the FPA and CFCW methods, the GMM clustering algorithm has the highest accuracy.

The pair-wise t-test is performed with the following hypothesis to make sure that among selected algorithms, the GMM is the most suitable algorithm statistically:

- There is a significant difference in estimation capability between segmentation algorithms used in the FPA and CFCWCA methods. This implies that one segmentation algorithm will outperform the others when used in the FPA and CFCWCA methods.

This pairwise t-test compares the estimation results using FPA and CFWCCA methods on the GMM clustering algorithm and the estimation results obtained on the remaining clustering algorithms. Table 7-55 and Table 7-56 show these results, respectively. Accordingly, the remaining algorithms' p-values, when compared to the GMM algorithm, consistently exceed 0.05 in terms of the evaluation criteria. With a 95% confidence level, it can be concluded that the other algorithms are not superior to the GMM method.

Table 7-55. The statistical t-test between the FPA method on clusters using the GMM clustering algorithm and the FPA method on clusters using other algorithms

Pairs of methods		FPA-BIR vs. FPA-GMM	FPA-FCM vs. FPA-GMM	FPA-KM vs. FPA-GMM	FPA-MS vs. FPA-GMM	FPA-SC vs. FPA-GMM
MAE results	Avg. MAE	373.348 vs. 369.142	378.117 vs. 369.142	374.419 vs. 369.142	376.656 vs. 369.142	396.125 vs. 369.142
	Avg. p-value	0.45307	0.41275	0.45620	0.41281	0.32251
	Statistical conclusion	<<	<<	<<	<<	<<
MAPE results	Avg. MAPE	9.317 vs. 8.648	8.881 vs. 8.648	9.557 vs. 8.648	9.7 vs. 8.648	9.353 vs. 8.648
	Avg. p-value	0.11150	0.32615	0.08600	0.11882	0.21837
	Statistical conclusion	<<	<<	<<	<<	<<
RMSE results	Avg. RMSE	656.883 vs. 657.015	682.408 vs. 657.015	750.744 vs. 657.015	626.917 vs. 657.015	691.345 vs. 657.015
	Avg. p-value	0.49927	0.32368	0.21156	0.31739	0.39141
	Statistical conclusion	<<	<<	<<	<<	<<

MBRE results	Avg. MBRE	0.094 vs. 0.087	0.089 vs. 0.087	0.096 vs. 0.087	0.098 vs. 0.087	0.094 vs. 0.087
	Avg. p-value	0.09979	0.31117	0.08446	0.10782	0.20255
	Statistical conclusion	<<	<<	<<	<<	<<
MIBRE results	Avg. MIBRE	0.079 vs. 0.073	0.075 vs. 0.073	0.078 vs. 0.073	0.081 vs. 0.073	0.079 vs. 0.073
	Avg. p-value	0.10439	0.36438	0.10328	0.09975	0.18438
	Statistical conclusion	<<	<<	<<	<<	<<

Table 7-56. The statistical t-test between the CFCWCA method on clusters using the GMM clustering algorithm and the CFCWCA method on clusters using other algorithms

Pairs of methods		CFCWCA-BIR vs. CFCWCA-GMM	CFCWCA-FCM vs. CFCWCA-GMM	CFCWCA-KM vs. CFCWCA-GMM	CFCWCA-MS vs. CFCWCA-GMM	CFCWCA-SC vs. CFCWCA-GMM
MAE results	Avg. MAE	324.241 vs. 321.896	312.846 vs. 321.896	319.081 vs. 321.896	321.304 vs. 321.896	353.78 vs. 321.896
	Avg. p-value	0.47188	0.39631	0.47308	0.49235	0.29180
	Statistical conclusion	<<	<<	<<	<<	<<
MAPE results	Avg. MAPE	8.114 vs. 7.47	7.527 vs. 7.47	8.342 vs. 7.47	8.36 vs. 7.47	8.334 vs. 7.47
	Avg. p-value	0.10488	0.44407	0.06959	0.11137	0.13666
	Statistical conclusion	<<	<<	<<	<<	<<
RMSE results	Avg. RMSE	526.967 vs. 534.901	s	588.201 vs. 534.901	494.17 vs. 534.901	579.825 vs. 534.901
	Avg. p-value	0.45334	0.44875	0.28808	0.23307	0.35854
	Statistical conclusion	<<	<<	<<	<<	<<
MBRE results	Avg. MBRE	0.084 vs. 0.076	0.077 vs. 0.076	0.086 vs. 0.076	0.086 vs. 0.076	0.086 vs. 0.076

	Avg. p-value	0.08384	0.39120	0.05661	0.09318	0.11773
	Statistical conclusion	<<	<<	<<	<<	<<
MIBRE results	Avg. MIBRE	0.073 vs. 0.067	0.067 vs. 0.067	0.073 vs. 0.067	0.074 vs. 0.067	0.074 vs. 0.067
	Avg. p-value	0.08928	0.48412	0.05622	0.08816	0.09743
	Statistical conclusion	<<	<<	<<	<<	<<

7.2.3 Summary

The experiment's goal in this section is to find which segmentation criterion is best suitable for the analysed dataset. We examine two aspects of segmentation: 1) segmentation based on segmentation variables and 2) segmentation based on clustering algorithms. There are four tested models for this experiment. The final composite results are shown in the following tables:

Table 7-57. The most-suitable results when applying the FPA method to categorical variables and segmentation algorithms

Types	Criteria	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Categorical Variables	IS	288.37	8.489	478.53	0.950	0.086	0.073
Clustering Algorithms	GMM	369.14	8.648	657.01	0.897	0.087	0.073

Table 7-57 is built based on selecting the results of applying the CFCW method on the best suitable segmentation criteria (IS) and the best suitable algorithm (GMM) based on six evaluation criteria. We can easily see that the FPA method applied to the clusters segmented by the IS categorical variable has a more miniature estimation error than the FPA method on the segments based on the GMM clustering algorithm.

Table 7-58 is built based on the selection of the results of applying the CFCW method on the best suitable segmentation criteria (IS) and the best suitable algorithm (GMM) based on six evaluation criteria. Accordingly, we can see that the estimation results using the CFCW method on the IS categorical variable have a smaller estimation error than the result achieved on clusters formed by the GMM algorithm.

Table 7-58. The most-suitable results when applying the CFCW method to categorical variables and segmentation algorithms

Types	Criteria	MAE	MAPE	RMSE	PRED	MBRE	MIBRE
Categorical Variables	IS	204.63	6.768	327.40	0.980	0.072	0.062
Clustering Algorithms	GMM	321.90	7.470	534.90	0.976	0.076	0.067

According to this result, with the segmentation criterion according to the IS categorical variable, the estimation accuracy consistently achieves better results (smaller estimation error) than the remaining criteria on both FPA and CFCW methods. Thus, the categorical variable "IS" is the most suitable segmentation criterion in this study.

7.3 New Calibration System and Optimization Framework

After determining the best-suitable machine learning algorithm (the BRR algorithm) and the best-suitable segmentation criterion (the IS categorical variable), this section aims to propose a new calibration complexity weight system. After applying this new system to software effort estimation, an optimization step will be applied to optimize the obtained results to give a better new result. Following are the results of this process.

7.3.1 Design Process

In the case of the first tested group, the CFCW model and CFCWO model are sequentially applied to the unsegmented dataset. For comparison purposes, a baseline model is also applied in this case. The base model is generated by applying the FPA model to this unsegmented dataset.

The evaluation results based on six evaluation criteria of this group are listed in Table 7-59.

Table 7-59. Evaluation results of the first group experiment

Criteria	Methods	Ex 1	Ex 2	Ex 3	Ex 4	Ex 5	**mean**
MAE	FPA	695.25	732.93	639.30	671.35	654.79	**678.72**
	CFCW	654.14	667.36	607.66	633.22	646.01	**641.68**
	CFCWO	565.39	571.50	555.00	570.61	563.25	**565.15**
MAPE	FPA	12.692	14.947	13.721	15.099	14.445	**14.181**
	CFCW	12.425	14.033	13.267	14.174	14.168	**13.613**
	CFCWO	11.402	12.755	12.677	13.589	13.128	**12.710**
RMSE	FPA	1580.32	1633.22	1591.47	1537.69	1551.89	**1578.92**
	CFCW	1420.73	1447.52	1427.09	1383.34	1395.47	**1414.83**
	CFCWO	1089.72	1087.42	1077.51	1042.25	1027.52	**1064.88**
PRED	FPA	0.826	0.767	0.849	0.791	0.826	**0.812**
	CFCW	0.872	0.860	0.884	0.872	0.872	**0.872**
	CFCWO	0.872	0.860	0.884	0.872	0.872	**0.872**
MBRE	FPA	0.129	0.153	0.138	0.153	0.149	**0.144**

	CFCW	0.134	0.147	0.136	0.149	0.153	**0.144**
	CFCWO	0.120	0.132	0.131	0.144	0.139	**0.133**
MIBRE	FPA	0.099	0.114	0.096	0.107	0.105	**0.104**
	CFCW	0.103	0.113	0.099	0.110	0.112	**0.107**
	CFCWO	0.097	0.105	0.099	0.108	0.106	**0.103**

As we can see, after applying the CFCW and then CFCWO methods, the estimation error is consistently more minor using the FPA method. It means that the proposed methods get more accurate than the FPA method. The calibration complexity weight system of this case is proposed in Table 7-60. The columns Ex 1 to Ex 5 are the five experiments on 5-fold cross-validation. The "mean" column is the mean value of five experiments.

Table 7-60. The calibration complexity weight system on the unsegmented dataset

Components	Complexity Level	Ex 1	Ex 2	Ex 3	Ex 4	Ex 5	**mean**
EI	Low	3.63	2.85	3.27	4.02	3.63	**3.48**
	Avg.	0.6	0.68	1.72	0.44	2.4	**1.17**
	High	10.5	9.66	6.3	7.38	7.92	**8.35**
EO	Low	3.08	3.28	3.64	4.2	3.24	**3.49**
	Avg.	4.1	4.5	4.2	4.55	4.75	**4.42**
	High	4.27	6.37	7.21	3.08	3.78	**4.94**
EQ	Low	4.08	4.44	3	4.05	3.36	**3.79**
	Avg.	4.6	6.04	7.84	6.04	5.16	**5.94**
	High	3.3	0.66	1.38	1.26	4.8	**2.28**
ILF	Low	4.8	5.2	5.6	6.1	3.95	**5.13**
	Avg.	5.74	7.56	7.91	7.35	7.56	**7.22**
	High	12.9	9.4	7.7	8.9	9	**9.58**
EIF	Low	8.89	7.28	7.07	6.79	5.6	**7.13**
	Avg.	4.3	8.5	6.4	6.6	9.4	**7.04**
	High	15.6	13.05	15.45	17.25	17.4	**15.75**

In the case of the second group, the tested dataset was segmented by the IS categorical variable (including Banking, Communication, Financial, Government, Insurance, Manufacturing, Service Industry, and Others). After the segmentation phase, each segment was applied to the CFCW method to calculate the effort. These results will be applied to the CWCFO framework for the optimization phase.

Like the first group, a baseline model is also created for comparison purposes. This baseline model is based on applying the FPA method to each segment. Based on six evaluation criteria in Section 6.5, the results of this phase are shown as follows.

Table 7-61. The evaluation result of the second group experiment

Criteria	Methods	BAN	COM	FIN	GOV	INS	MAN	SI	Others
MAE	FPA	463.96	391.92	219.86	567.01	516.91	207.29	319.78	354.73
	CFCW	301.02	213.70	195.12	508.22	376.64	192.84	294.48	268.76
	CFCWO	244.56	195.21	153.93	490.69	350.50	168.38	256.66	234.33
MAPE	FPA	10.625	14.416	8.152	8.413	13.369	11.191	7.413	10.151
	CFCW	7.359	10.062	7.722	7.708	11.473	10.422	6.638	8.161
	CFCWO	6.289	8.616	6.573	7.308	10.749	9.597	5.928	7.609
RMSE	FPA	828.43	574.89	315.90	1210.29	813.23	362.54	461.49	611.07
	CFCW	441.71	276.41	279.20	1095.56	548.40	311.59	370.71	381.67
	CFCWO	322.69	257.52	250.32	1039.66	523.46	233.31	340.40	329.97
PRED	FPA	0.909	0.820	1.000	0.933	0.738	0.782	0.867	0.875
	CFCW	1.000	0.940	1.000	0.933	0.862	0.855	0.956	0.975
	CFCWO	1.000	0.940	1.000	0.933	0.862	0.855	0.956	0.975
MBRE	FPA	0.106	0.144	0.089	0.087	0.134	0.113	0.074	0.102
	CFCW	0.076	0.114	0.087	0.081	0.126	0.107	0.068	0.085
	CFCWO	0.065	0.094	0.071	0.076	0.115	0.099	0.060	0.080
MIBRE	FPA	0.089	0.120	0.077	0.070	0.107	0.085	0.063	0.086
	CFCW	0.068	0.097	0.074	0.066	0.102	0.084	0.060	0.075
	CFCWO	0.060	0.081	0.061	0.063	0.094	0.080	0.052	0.071

As we can observe, the evaluation results always decrease from FPA to CFCW and then to CFCWO in all segments for each evaluation criterion. That means that the CFCW method achieves higher accuracy than the FPA method, and the CFCWO method is consistently more accurate than the CFCW. Fig. 7-7 presents the evaluation results graphically.

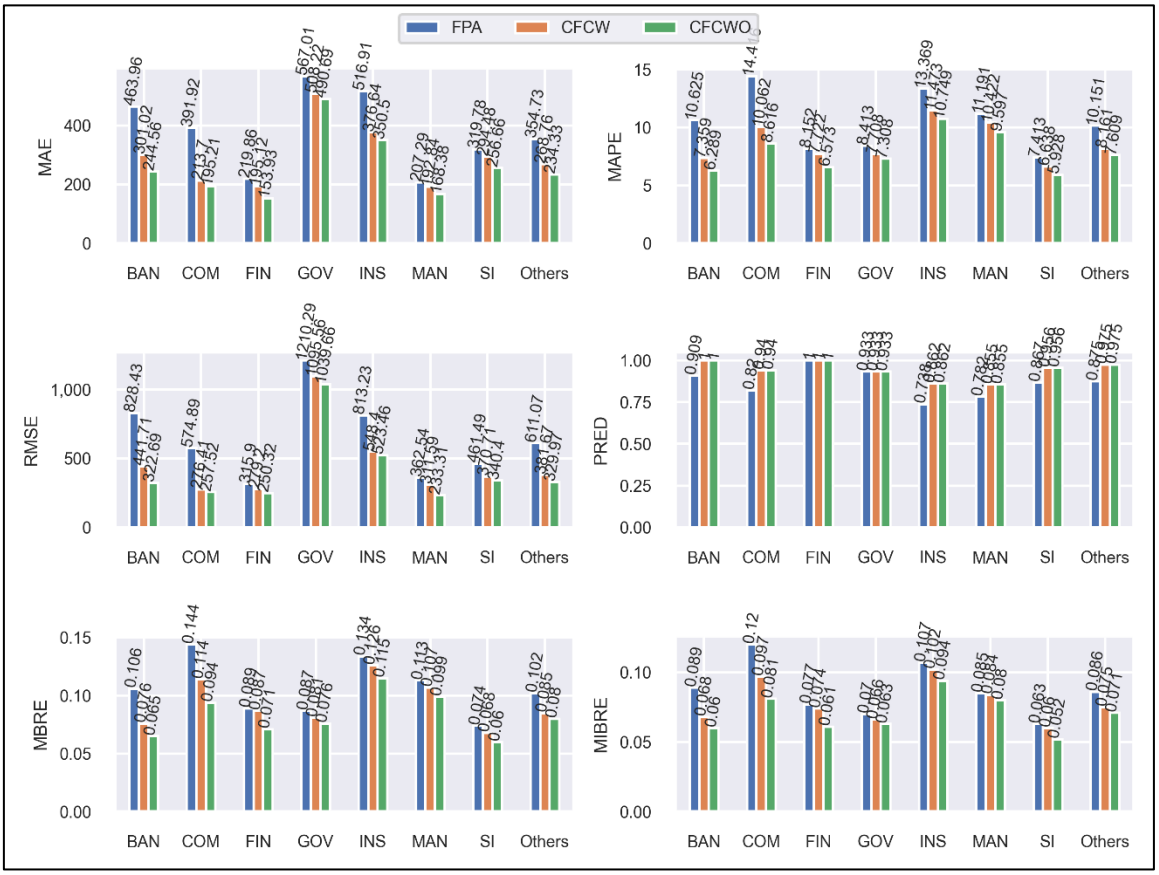


Fig. 7-7. The evaluation results

The proposed calibration complexity weight system from this experiment is shown in Table 7-62. The first column is the name of the components (EI, EO, EQ, ILF, and EIF), and the second column is the complexity level (including Low (L), Average (A), and High (H)). The third column contains the complexity weight values from the IFPUG FPA method. The unsegmented column is the complexity weight corresponding to the unsegmented dataset. The remains columns are the complexity weight of the segments in IS categorical variable.

Table 7-62. Proposed Calibration Complexity Weight system on segments of the IS categorical variable

Components	Level	FPA	Unsegmented	BAN	COM	FIN	GOV	INS	MAN	SI	Others
EI	L	3	3.48	1.73	1.10	3.38	0.76	1.81	3.49	3.95	3.33
	A	4	1.17	1.21	4.04	4.15	7.13	4.48	0.99	5.50	1.15
	H	6	8.35	7.90	4.07	4.68	7.33	3.60	8.54	6.72	9.53
EO	L	4	3.49	3.06	3.42	5.24	2.87	4.19	3.58	3.83	3.07
	A	5	4.42	5.37	2.49	1.99	5.40	5.68	5.28	4.74	4.31
	H	7	4.94	8.09	7.60	6.37	5.50	5.42	6.89	6.37	7.20
EQ	L	3	3.79	0.71	1.51	3.41	2.90	3.55	2.00	2.42	2.83
	A	4	5.94	4.39	3.45	3.71	4.18	6.40	4.81	4.26	6.11
	H	6	2.28	9.48	4.37	5.90	3.32	6.10	5.78	4.81	3.38
ILF	L	5	5.13	6.26	4.92	4.14	7.91	4.73	5.52	3.07	3.07
	A	7	7.22	2.24	10.65	9.58	5.84	4.65	7.99	3.53	10.15
	H	10	9.58	15.96	9.82	7.24	7.20	9.08	6.38	5.82	11.30
EIF	L	7	7.13	2.23	6.93	8.64	6.12	10.26	8.55	5.82	6.34
	A	10	7.04	10.16	10.82	13.04	11.96	2.58	1.88	17.32	4.94
	H	15	15.75	24.90	8.19	10.23	19.53	9.42	22.41	16.59	12.99

To ensure that the experiment results are statistically correct, the pairwise t-test technique is performed. The comparisons are based on the results of the CFCW versus FPA methods and the CFCWO versus CFCW methods on the sectors of the IS categorical variable. The hypotheses for this situation are:

- H1: There is a significant difference in estimation capability between using the CFCW and the FPA methods on the segmented dataset based on categorical variables. This statement implies that the estimation accuracy when using the CFCW approach differs significantly from that of the FPA manner on the segmented dataset based on categorical variables.
- H2: There is a significant difference in estimation capability between using the CFCWO and the CFCW methods on the segmented dataset based on categorical variables. This statement implies that the estimation accuracy when using the CFCWO approach differs significantly from that of the CFCW manner on the segmented dataset based on categorical variables.

Table 7-63,Table 7-64, and Table 7-65 display the pairwise t-test results. There are two comparisons on each column (a sector of the IS categorical variable): CFCW vs FPA and CFCWO vs CFCW. The notation ≫ reflects the statistical superiority of the CFCW approach compared to the FPA method (and CFCWO vs CFCW) on sectors of the IS categorical variable.

The findings show that the CFCW approach is statistically significant compared to the FPA method at the 95% confidence level. The CFCWO is also statistically significant compared to the CFCW method at the 95% confidence level.

As a result, we adopt the proposed hypotheses (H1 and H2), which are compatible with the abovementioned results. This means that the CFCW method outperforms the FPA method and the CFCWO method outperforms the CFCW method on the dataset with segmentation using the IS categorical variable.

Table 7-63. The statistical t-test result based on the evaluation results of the segmented dataset using the categorical variables

Pairs of methods		Banking		Communication		Financial	
		CFCW vs FPA	CFCWO vs CFCW	CFCW vs FPA	CFCWO vs CFCW	CFCW vs FPA	CFCWO vs CFCW
MAE results	Avg. MAE	301.021vs 463.962	244.869 vs 301.021	213.703vs 391.924	194.783 vs 213.703	195.121vs 219.855	153.836 vs 195.121
	Avg. p-value	0.00695	0.01064	0.00404	0.0211	0.01648	0.01143
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	7.359vs 10.625	6.296 vs 7.359	10.062vs 14.416	8.599 vs 10.062	7.722vs 8.152	6.573 vs 7.722
	Avg. p-value	0.02279	0.00646	0.0295	0.00757	0.00272	0.0052
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MSE results	Avg. MSE	2.11E+05 vs 7.58E+05	1.13E+05 vs 2.11E+05	8.06E+04 vs 3.37E+05	7.05E+04 vs 8.06E+04	8.49E+04 vs 1.05E+05	7.10E+04 vs 8.49E+04
	Avg. p-value	0.00593	0.00916	0.0049	0.04882	0.00829	0.02331
	Statistical conclusion	>>	>>	>>	>>	>>	>>

RMSE results	Avg. RMSE	441.706vs 828.429	323.04 vs 441.706	276.413vs 574.89	257.375 vs 276.413	279.2vs 315.902	250.388 vs 279.2
	Avg. p-value	0.00285	0.00914	0.0041	0.04341	0.02201	0.02909
	Statistical conclusion	>>	>>	>>	>>	>>	>>

Table 7-64. The statistical t-test result based on the evaluation results of the segmented dataset using the categorical variables (cont.)

Pairs of methods		Government		Insurance		Manufacturing	
		CFCW vs FPA	CFCWO vs CFCW	CFCW vs FPA	CFCWO vs CFCW	CFCW vs FPA	CFCWO vs CFCW
MAE results	Avg. MAE	508.218vs 567.009	490.656 vs 508.218	376.642 vs 516.907	350.725 vs 376.642	192.842vs 207.285	168.118 vs 192.842
	Avg. p-value	0.00221	0.00341	0.00032	0.00501	0.00018	0.01121
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MAPE results	Avg. MAPE	7.708vs 8.413	7.301 vs 7.708	11.473vs 13.369	10.752 vs 11.473	10.422vs 11.191	9.585 vs 10.422
	Avg. p-value	0.0153	0.03273	0.01373	0.00915	0.02223	0.01524
	Statistical conclusion	>>	>>	>>	>>	>>	>>
MSE results	Avg. MSE	1.25E+06 vs 1.53E+06	1.12E+06 vs 1.25E+06	3.02E+05 vs 6.69E+05	2.74E+05 vs 3.02E+05	9.95E+04 vs 1.34E+05	6.10E+04 vs 9.95E+04
	Avg. p-value	0.00321	0.00503	0.00114	0.0133	0.00067	0.00061
	Statistical conclusion	>>	>>	>>	>>	>>	>>

RMSE results	Avg. RMSE	1095.559 vs 1210.293	1038.723 vs 1095.559	548.4vs 813.234	522.889 vs 548.4	311.594vs 362.54	233.02 vs 311.594
	Avg. p-value	0.00106	0.00212	0.00039	0.01267	0.00039	0.00572
	Statistical conclusion	>>	>>	>>	>>	>>	>>

Table 7-65. The statistical t-test result based on the evaluation results of the segmented dataset using the categorical variables (cont.)

Pairs of methods		Service Industry		Others	
		CFCW vs FPA	CFCWO vs CFCW	CFCW vs FPA	CFCWO vs CFCW
MAE results	Avg. MAE	294.475vs 319.784	256.416 vs 294.475	6.638vs 7.413	5.912 vs 6.638
	Avg. p-value	0.00439	0.00388	0.00805	0.00047
	Statistical conclusion	>>	>>	>>	>>
MAPE results	Avg. MAPE	6.638vs 7.413	5.912 vs 6.638	8.161vs 10.151	7.595 vs 8.161
	Avg. p-value	0.00805	0.00047	0.01801	0.00756
	Statistical conclusion	>>	>>	>>	>>
MSE results	Avg. MSE	1.41E+05 vs 2.17E+05	1.21E+05 vs 1.41E+05	1.56E+05 vs 3.76E+05	1.17E+05 vs 1.56E+05
	Avg. p-value	0.00005	0.00105	0.00093	0.01837
	Statistical conclusion	>>	>>	>>	>>
RMSE results	Avg. RMSE	370.708vs 461.492	339.92 vs 370.708	381.667vs 611.073	330.435 vs 381.667
	Avg. p-value	0	0.00208	0.00281	0.00848
	Statistical conclusion	>>	>>	>>	>>

7.3.2 Summary

This experiment has given a new calibration complexity weight system to be applied with the data segmented according to the IS categorical variable. When a new project needs to estimate the effort, it uses this new calibration complexity weight system to estimate the effort. The result of this phase then applies the optimization framework CFCWO to get more accurate results. Experimental results also confirmed that the proposed method accomplishes higher accuracy than the FPA method.

8. THREAT OF VALIDITY

Internal validity is an incorrect/inaccurate evaluation approach to analysing the proposed method in this study, which can affect the validity of conclusions produced from experimental research. It refers to the method of statistical sample validation in particular. The k-fold cross-validation method was used to mitigate the threat to this validity, ensuring that the suggested method is appropriately appraised. Another internal hazard that may affect the validity of the generated findings is the parameter selection in the machine learning technique. We employ the BRR technique's default parameter configuration for the proposed method in this work.

The external validity of the data produced in this study is concerned with whether the results gained might be utilized in a different environment. The proposed method's prediction ability was tested using the ISBSG repository August 2020 R1 dataset. The dataset covers a variety of software projects from various organizations throughout the world, each with its own set of features, fields, size, and number of features.

This study uses the evaluation criteria MAE, MAPE, RMSE, PRED (0.25), MBRE, and MIBRE to assess the experiment's accuracy. According to published studies [128], [129], the above evaluation criteria are classified as unbiased evaluation criteria. Therefore, we can conclude that this study's experimental results are highly generalizable.

9. CONTRIBUTION OF THE THESIS TO SCIENCE AND PRACTICE

The main contribution of the thesis is the proposal of procedures for more accurate software effort estimation by improving the Functional Point Analysis method. The FPA method was born and is widely applied to the software industry. However, many reasons lead to this method being inadequate, as mentioned in

section 4. That leads to the method needing to be updated to meet the evolving trends of the modern software industry.

Overall, it is possible to summarize the contributions as follows:

- The results of the performed experiments clearly showed that the estimates of the development effort using the new calibration complexity weight algorithm are more accurate than the estimates using the IFPUG FPA reference method.
- The effect of clustering has been demonstrated, allowing new algorithms to be applied to clustered data with the benefit of increasing the accuracy of effort estimation.
- The most suitable clustering algorithm and categorical variable were determined in the context of the study.
- A new framework has been created to optimize effort estimation based on improved FPA using regression models, machine learning, and clustering.
- Based on experimental results, the Bayesian Ridge Regressor (BRR) algorithm is the most appropriate approach to a new framework for optimizing software effort estimation.

10. CONCLUSION AND FUTURE WORK

FPA was proposed and has made significant contributions to the software industry. Besides, machine learning has also brought a big revolution in the field of computer science; Software development effort estimation is no exception. The application of machine learning in software effort estimation has been achieving many remarkable achievements. This study combines FPA (traditional) and machine learning (modern) methods to create a new method. In which the effort estimate in principle is still based on FPA but with the complexity weight system on the basis of machine learning (CFCW). In addition, the results of the FPA-based estimation process are once again optimized to achieve higher accuracy (CFCWO). It has been demonstrated experimentally that with the proposal of this study, the accuracy can be improved markedly.

Because software engineering is a continually changing field, today's actual values may not correctly reflect software values tomorrow. As a result, the weights proposed in this work must be revised to reflect the new trend. The ISBSG dataset is a current database of companies from all over the world. It represents the dynamic nature of today's software industry. As a result, when project data is updated in the future, the IFPUG FPA weighting values should be recalculated to reflect the most recent software industry trends. Cause the coefficients are coherently related to data, the calibration process should be re-performed when using another dataset differ ISBSG.

In this thesis, the main work is focused on improving the effort estimation accuracy based on the functional complexity weight calibration. In fact, two other factors in effort estimation need to be considered, VAF and productivity factor. Future work will focus on these factors. With VAF, 14 GSCs are assessed as potentially obsolete, or some of these properties are not suitable for the current situation. It is necessary to find the proper criteria for modern software industry trends and their influence. With the productivity factor, new development technologies help significantly improve productivity. The determination of this factor is also another work that needs attention. In addition, other estimation methods such as COSMIC, FiSMA, and NESMA will also be studied as alternatives to the IFPUG FPA method.

11. REFERENCES

[1] P. Naur and B. Randell, "Software engineering: Report of a conference sponsored by the NATO science committee, Garmisch, Germany, 7th-11th October 1968," 1969.

[2] Standish Group, "CHAOS report," Standish Group 2015 Chaos Report, 2019.

[3] R.N. Charette, "Why software fails [software failure]," IEEE Spectrum, vol. 42, pp. 42-49, 2005.

[4] F.J. Heemstra, "Software cost estimation," Information and Software Technology, vol. 34, pp. 627-639, 1992.

[5] T. Vera, S.F. Ochoa and D. Perovich, "Survey of software development effort estimation taxonomies," Computer Science Department, University of Chile: Santiago, Chile, 2017.

[6] B. Khan, W. Khan, M. Arshad and N. Jan, "Software cost estimation: Algorithmic and non-algorithmic approaches," International Journal of Data Science and Advanced Analytics (ISSN 2563-4429), vol. 2, pp. 1-5, 2020.

[7] P. Sharma and J. Singh, "Systematic literature review on software effort estimation using machine learning approaches," in 2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS), pp. 43-47, 2017.

[8] Y. Mahmood, N. Kama, A. Azmi, A.S. Khan and M. Ali, "Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation," Software: Practice and Experience, vol. 52, pp. 39-65, 2022.

[9] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," IEEE Trans.Software Eng., vol. 21, pp. 126-137, 1995.

[10] V.V. Hai, Thi Kim Nhung, Ho Le and H.T. Hoc, "A review of software effort estimation by using functional points analysis," Proceedings of the Computational Methods in Systems and Software, pp. 408-422, 2019.

[11] A.J. Albrecht, "Measuring application development productivity," in Proc. Joint Share, Guide, and IBM Application Development Symposium, 1979, 1979.

[12] Z. Prokopová, R. Silhavy and P. Silhavy, "The effects of clustering to software size estimation for the use case points methods," in Computer Science On-line Conference, pp. 479-490, 2017.

[13] U. Gupta and M. Kumar, "Software effort estimation through clustering techniques of RBFN network," IOSR Journal of Computer Engineering (IOSR-JCE), vol. 14, pp. 58-62, 2013.

[14] A. Idri, M. Hosni and A. Abran, "Systematic literature review of ensemble effort estimation," J.Syst.Software, vol. 118, pp. 151-175, 2016.

[15] S. Goyal, "Effective Software Effort Estimation using Heterogenous Stacked Ensemble," in 2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), pp. 584-588, 2022.

[16] V. Mahajan, "The Delphi method: Techniques and applications," JMR, Journal of Marketing Research (Pre-1986), vol. 13, pp. 317, 1976.

[17] N. Dalkey and O. Helmer, "An experimental application of the Delphi method to the use of experts," Management Science, vol. 9, pp. 458-467, 1963.

[18] G. Rowe and G. Wright, "The Delphi technique as a forecasting tool: issues and analysis," Int.J.Forecast., vol. 15, pp. 353-375, 1999.

[19] B. Barry, "Software engineering economics," New York, vol. 197, 1981.

[20] G. Booch, I. Jacobson and J. Rumbaugh, "Unified Modeling Language Specifications-version 1.1," UML Consortium–Object Management Group, 1997.

[21] Object Management Group, "Unified modeling language (UML) specification: Superstructure, version 2.0," 2004.

[22] G. Karner, "Resource estimation for objectory projects," Objective Systems SF AB, vol. 17, pp. 9, 1993.

[23] IFPUG, "International Function Point User Group," vol. 2022, .

[24] I. ISO, "IEC 20926: 2009 Software and systems engineering-Software measurement-IFPUG functional size measurement method 2009. Standard," International Organization for Standardization, Geneva, .

[25] IFPUG, Function Point Counting Practices Manual, Ohio ,USA: Westerville, 2010, .

[26] C.R. Symons, "Function point analysis: difficulties and improvements," IEEE Trans.Software Eng., vol. 14, pp. 2-11, 1988.

[27] UKSMA Metrics Practices Committee, "Function Point Analysis Counting Practice Manual–Version 1.3.1," 1998, .

[28] I. ISO, "IEC 20968: 2002, Software engineering-Mk II Function Point Analysis-Counting Practices Manual," International Organization for Standardization, Geneva, 2002.

[29] I. ISO, "IEC 19761: 2003 Software Engineering-COSMIC-FFP-A functional size measurement method," International Organization for Standardization-ISO, Geneva, 2003.

[30] ISO/IEC 29881: 2010, "Information technology, Systems and software engineering, FiSMA 1.1 functional size measurement method," 2010.

[31] D. NESMA, "Counting Guidelines for the Application of Function Point Analysis," 1997.

[32] AS/NZS, ISO/IEC 24570: 2007 Software Engineering - NESMA Functional Size Measurement Method Version 2.1 - Definitions and Counting Guidelines for the Application of Function Point Analysis, 2005, .

[33] R. Valerdi, The constructive systems engineering cost model (COSYSMO), University of Southern California, 2005, .

[34] Function Point Modeler Inc., "Function Point Modeler," vol. 2022, .

[35] T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: an efficient data clustering method for very large databases," ACM Sigmod Record, vol. 25, pp. 103-114, 1996.

[36] J.C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," 1973.

[37] J.C. Bezdek, R. Ehrlich and W. Full, "FCM: The fuzzy c-means clustering algorithm," Comput.Geosci., vol. 10, pp. 191-203, 1984.

[38] J. Liu, D. Cai and X. He, "Gaussian mixture model with local consistency," in Proceedings of the AAAI conference on artificial intelligence, pp. 512-517, 2010.

[39] M. Nilashi, O. bin Ibrahim, N. Ithnin and N.H. Sarmin, "A multi-criteria collaborative filtering recommender system for the tourism domain using Expectation Maximization (EM) and PCA–ANFIS," Electronic Commerce Research and Applications, vol. 14, pp. 542-562, 2015.

[40] J. McQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281-297, 1967.

[41] S.S. Khan and A. Ahmad, "Cluster center initialization algorithm for K-means clustering," Pattern Recog.Lett., vol. 25, pp. 1293-1302, 2004.

[42] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," IEEE Trans.Pattern Anal.Mach.Intell., vol. 24, pp. 603-619, 2002.

[43] D. Comaniciu and P. Meer, "Mean shift analysis and applications," in Proceedings of the seventh IEEE international conference on computer vision, pp. 1197-1203, 1999.

[44] Y. Dong, Y. Dong, J. Shi, Y. Deng and S. Pang, "MSK: A Grade Mining Method for Second-Class Based on MeanShift and K-means," in 2021 IEEE 3rd International Conference on Computer Science and Educational Informatization (CSEI), pp. 109-114, 2021.

[45] A. Ng, M. Jordan and Y. Weiss, "On spectral clustering: Analysis and an algorithm," Advances in Neural Information Processing Systems, vol. 14, 2001.

[46] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Trans.Pattern Anal.Mach.Intell., vol. 22, pp. 888-905, 2000.

[47] J.F. Hair, "Multivariate data analysis," 2009.

[48] F. Rosenblatt, "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms," Principles of Neurodynamics.Perceptrons and the Theory of Brain Mechanisms, 1961.

[49] S. Haykin and N. Network, "A comprehensive foundation," Neural Networks, vol. 2, pp. 41, 2004.

[50] G.R. Finnie and G.E. Wittig, "AI tools for software development effort estimation," in Proceedings 1996 International Conference Software Engineering: Education and Practice, pp. 346-353, 1996.

[51] V.N. Vapnik, "An overview of statistical learning theory," IEEE Trans.Neural Networks, vol. 10, pp. 988-999, 1999.

[52] N. Cristianini and J. Shawe-Taylor, An introduction to support vector machines and other kernel-based learning methods, Cambridge university press, 2000, .

[53] M. Awad and R. Khanna, Efficient learning machines: theories, concepts, and applications for engineers and system designers, Springer nature, 2015, .

[54] D.J. MacKay, "Bayesian interpolation," Neural Comput., vol. 4, pp. 415-447, 1992.

[55] A.E. Hoerl and R.W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," Technometrics, vol. 12, pp. 55-67, 1970.

[56] M.E. Tipping, "Sparse Bayesian learning and the relevance vector machine," Journal of Machine Learning Research, vol. 1, pp. 211-244, 2001.

[57] R. Tibshirani, "Regression shrinkage and selection via the lasso," Journal of the Royal Statistical Society: Series B (Methodological), vol. 58, pp. 267-288, 1996.

[58] B. Efron, T. Hastie, I. Johnstone and R. Tibshirani, "Least angle regression," The Annals of Statistics, vol. 32, pp. 407-499, 2004.

[59] K. An and J. Meng, "Voting-averaged combination method for regressor ensemble," in International Conference on Intelligent Computing, pp. 540-546, 2010.

[60] I.H. Witten, E. Frank, M.A. Hall, C.J. Pal and M. DATA, "Practical machine learning tools and techniques," in Data Mining, 2005.

[61] S.D. Sheetz, D. Henderson and L. Wallace, "Understanding developer and manager perceptions of function points and source lines of code," J.Syst.Software, vol. 82, pp. 1540-1549, 2009.

[62] P. Kampstra and C. Verhoef, "Reliability of function point counts," Department of Computer Science, VU University Amsterdam, Amsterdam, the Netherlands, 2010.

[63] C.F. Kemerer, "Reliability of function points measurement: a field experiment," Commun ACM, vol. 36, pp. 85-97, 1993.

[64] R. Meli, "Functional metrics: problems and possible solutions," Proc.of the FESMA, Brussels, pp. 503-514, 1998.

[65] W. Xia, L.F. Capretz and D. Ho, "Neuro-fuzzy approach to calibrate function points," in Proc. 8th WSEAS Int. Conf. Fuzzy Syst, pp. 116-119, 2007.

[66] W. Xia, L.F. Capretz, D. Ho and F. Ahmed, "A new calibration for Function Point complexity weights," Information and Software Technology, vol. 50, pp. 670-683, 2008.

[67] W. Xia, D. Ho and L.F. Capretz, "A neuro-fuzzy model for function point calibration," arXiv Preprint arXiv:1507.06934, 2015.

[68] F. Ahmed, S. Bouktif, A. Serhani and I. Khalil, "Integrating function point project information for improving the accuracy of effort estimation," in 2008 The Second International Conference on Advanced Engineering Computing and Applications in Sciences, pp. 193-198, 2008.

[69] M.A. Al-Hajri, A.A.A. Ghani, M.N. Sulaiman and M.H. Selamat, "Modification of standard function point complexity weights system," J.Syst.Software, vol. 74, pp. 195-206, 2005.

[70] Y. Fu, X. Liu, R. Yang, Y. Du and Y. Li, "A software size estimation method based on improved FPA," in 2010 Second World Congress on Software Engineering, pp. 228-233, 2010.

[71] K.K. Rao and G.S. Raju, "Error correction in function point estimation using soft computing technique," in Proceedings of the International Conference on Advances in Computing and Artificial Intelligence, pp. 194-198, 2011.

[72] J. Wen, S. Li, Z. Lin, Y. Hu and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," Information and Software Technology, vol. 54, pp. 41-59, 2012.

[73] P. Phannachitta, "On an optimal analogy-based software effort estimation," Information and Software Technology, vol. 125, pp. 106330, 2020.

[74] P. Phannachitta and K. Matsumoto, "Model-based software effort estimation—a robust comparison of 14 algorithms widely used in the data science community," Int.J.Innov.Comput.Inf.Control, vol. 15, pp. 569-589, 2019.

[75] S. Shukla and S. Kumar, "Applicability of neural network-based models for software effort estimation," in 2019 IEEE World Congress on Services (SERVICES), pp. 339-342, 2019.

[76] S. Shukla, S. Kumar and P.R. Bal, "Analyzing effect of ensemble models on multi-layer perceptron network for software effort estimation," in 2019 IEEE World Congress on Services (SERVICES), pp. 386-387, 2019.

[77] P.V. AG and V. Varadarajan, "Estimating software development efforts using a random forest-based stacked ensemble approach," Electronics, vol. 10, pp. 1195, 2021.

[78] M. Hammad and A. Alqaddoumi, "Features-level software effort estimation using machine learning algorithms," in 2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), pp. 1-3, 2018.

[79] J. Aroba, J.J. Cuadrado-Gallego, M. Sicilia, I. Ramos and E. Garcia-Barriocanal, "Segmented software cost estimation models based on fuzzy clustering," J.Syst.Software, vol. 81, pp. 1944-1950, 2008.

[80] P. Silhavy, R. Silhavy and Z. Prokopova, "Categorical variable segmentation model for software development effort estimation," IEEE Access, vol. 7, pp. 9618-9626, 2019.

[81] H. Azath, M. Mohanapriya and S. Rajalakshmi, "Software Effort Estimation Using Modified Fuzzy C Means Clustering and Hybrid ABC-MCS Optimization in Neural Network," J.Intell.Syst., vol. 29, pp. 251-263, 2020.

[82] V.K. Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim and E. Khatibi, "Increasing the accuracy of software development effort estimation using projects clustering," IET Software, vol. 6, pp. 461-473, 2012.

[83] T.R. Benala, R. Mall, S. Dehuri and K.C. Babu, "Software effort prediction using unsupervised learning (clustering) and functional link artificial neural networks," in 2012 World Congress on Information and Communication Technologies, pp. 115-120, 2012.

[84] International Software Benchmarking Standards Group, "ISGSB dataset," .

[85] G. Upton and I. Cook, Understanding statistics, Oxford University Press, 1996, .

[86] D. Zwillinger and S. Kokoska, CRC standard probability and statistics tables and formulae, Crc Press, 1999, .

[87] X. Yang and W. Wen, "Ridge and lasso regression models for cross-version defect prediction," IEEE Trans.Reliab., vol. 67, pp. 885-896, 2018.

[88] A.G. Priya Varshini and K. Anitha Kumari, "Predictive analytics approaches for software effort estimation: A review," Indian J Sci Technol, vol. 13, pp. 2094-2103, 2020.

[89] S. Goyal and P.K. Bhatia, "A non-linear technique for effective software effort estimation using multi-layer perceptrons," in 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), pp. 1-4, 2019.

[90] T.M. Abdellatif, "A comparison study between soft computing and statistical regression techniques for software effort estimation," in 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), pp. 1-5, 2018.

[91] A.B. Nassif, M. AlaaEddin and A.A. Sahib, "Machine learning models for stock price prediction," in 2020 Seventh International Conference on Information Technology Trends (ITT), pp. 67-71, 2020.

[92] A.B. Nassif, L.F. Capretz and D. Ho, "Estimating software effort using an ANN model based on use case points," in 2012 11th International Conference on machine learning and applications, pp. 42-47, 2012.

[93] Le Thi Kim Nhung, Ho, H.T. Hoc and V.V. Hai, "An evaluation of technical and environmental complexity factors for improving use case points estimation," in Proceedings of the Computational Methods in Systems and Software, pp. 757-768, 2020.

[94] V.V. Hai, Nhung, Ho Le Thi Kim and H.T. Hoc, "A Productivity Optimising Model for Improving Software Effort Estimation," in Proceedings of the Computational Methods in Systems and Software, pp. 735-746, 2020.

[95] M.V.N. Hemrajani, "PREDICTING EFFORT OF AGILE SOFTWARE PROJECTS USING LINEAR REGRESSION, RIDGE REGRESSION AND LOGISTIC REGRESSION," .

[96] Nhung, Ho Le Thi Kim, V. Van Hai, R. Silhavy, Z. Prokopova and P. Silhavy, "Parametric software effort estimation based on optimizing correction factors and multiple linear regression," IEEE Access, vol. 10, pp. 2963-2986, 2021.

[97] P. Sharma and J. Singh, "Machine learning based effort estimation using standardization," in 2018 International Conference on Computing, Power and Communication Technologies (GUCON), pp. 716-720, 2018.

[98] M. Hosni, A. Idri, A.B. Nassif and A. Abran, "Heterogeneous ensembles for software development effort estimation," in 2016 3rd international conference on soft computing & machine intelligence (ISCMI), pp. 174-178, 2016.

[99] A.B. Nassif, O. Mahdi, Q. Nasir, M.A. Talib and M. Azzeh, "Machine learning classifications of coronary artery disease," in 2018 International Joint

Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP), pp. 1-6, 2018.

[100] H. Luo and Y. Liu, "A prediction method based on improved ridge regression," in 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), pp. 596-599, 2017.

[101] M. Suhail and S. Chand, "Performance of some new ridge regression estimators," in 2019 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS), pp. 1-4, 2019.

[102] A. BaniMustafa, "Predicting software effort estimation using machine learning techniques," in 2018 8th International Conference on Computer Science and Information Technology (CSIT), pp. 249-256, 2018.

[103] B.K. Singh and A.K. Misra, "Software effort estimation by genetic algorithm tuned parameters of modified constructive cost model for nasa software projects," International Journal of Computer Applications, vol. 59, 2012.

[104] J. Huang, Y. Li, J.W. Keung, Y.T. Yu and W.K. Chan, "An empirical analysis of three-stage data-preprocessing for analogy-based software effort estimation on the ISBSG data," in 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS), pp. 442-449, 2017.

[105] K. Meridji, K.T. Al-Sarayreh, M. Abu-Arqoub and W.M. Hadi, "Exploration of development projects of renewable energy applications in the ISBSG dataset: Empirical study," in 2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes & Systems (IT-DREPS), pp. 1-6, 2017.

[106] K. Kaewbanjong and S. Intakosum, "Statistical analysis with prediction models of user satisfaction in software project factors," in 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 637-643, 2020.

[107] S.P. Pillai, S.D. Madhukumar and T. Radharamanan, "Consolidating evidence-based studies in software cost/effort estimation—A tertiary study," in TENCON 2017-2017 IEEE Region 10 Conference, pp. 833-838, 2017.

[108] M. Fernández-Diego and F. González-Ladrón-de-Guevara, "Application of mutual information-based sequential feature selection to ISBSG mixed data," Software Quality Journal, vol. 26, pp. 1299-1325, 2018.

[109] J. Liu, Q. Du and J. Xu, "A learning-based adjustment model with genetic algorithm of function point estimation," in 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), pp. 51-58, 2018.

[110] P. Pospieszny, B. Czarnacka-Chrobot and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," J.Syst.Software, vol. 137, pp. 184-196, 2018.

[111] L. Song, L.L. Minku and X. Yao, "Software effort interval prediction via Bayesian inference and synthetic bootstrap resampling," ACM Transactions on Software Engineering and Methodology (TOSEM), vol. 28, pp. 1-46, 2019.

[112] C. López-Martín, "Feedforward Neural Networks for Predicting the Duration of Maintained Software Projects," in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 528-533, 2016.

[113] K. Usharani, V.V. Ananth and D. Velmurugan, "A survey on software effort estimation," in 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp. 505-509, 2016.

[114] Y. Li, L. Shi, J. Hu, Q. Wang and J. Zhai, "An empirical study to revisit productivity across different programming languages," in 2017 24th Asia-Pacific Software Engineering Conference (APSEC), pp. 526-533, 2017.

[115] J.I.S. Martínez, F.V. Souto and M.R. Monje, "Analysis of automated estimation models using machine learning," in 2020 8th International Conference in Software Engineering Research and Innovation (CONISOFT), pp. 110-116, 2020.

[116] V.V. Hai, Nhung, Ho Le Thi Kim, Z. Prokopova, R. Silhavy and P. Silhavy, "A New Approach to Calibrating Functional Complexity Weight in Software Development Effort Estimation," Computers, vol. 11, pp. 15, 2022.

[117] F. González-Ladrón-de-Guevara, M. Fernández-Diego and C. Lokan, "The usage of ISBSG data fields in software effort estimation: A systematic mapping study," J.Syst.Software, vol. 113, pp. 188-215, 2016.

[118] Z. Prokopova, P. Silhavy and R. Silhavy, "Influence analysis of selected factors in the function point work effort estimation," in Proceedings of the Computational Methods in Systems and Software, pp. 112-124, 2018.

[119] R. ISBSG, "R1 Field Description," vol. R1, 2020.

[120] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," Annals of Data Science, vol. 2, pp. 165-193, 2015.

[121] B. Andreopoulos, A. An, X. Wang and M. Schroeder, "A roadmap of clustering algorithms: finding a match for a biomedical application," Briefings in Bioinformatics, vol. 10, pp. 297-314, 2009.

[122] M.K. Gupta and P. Chandra, "A comparative study of clustering algorithms," in 2019 6th International Conference on Computing for Sustainable Global Development (INDIACom), pp. 801-805, 2019.

[123] P.J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," J.Comput.Appl.Math., vol. 20, pp. 53-65, 1987.

[124] A.B. Nassif, L.F. Capretz and D. Ho, "Estimating software effort based on use case point model using sugeno fuzzy inference system," in 2011 IEEE 23rd International Conference on Tools with Artificial Intelligence, pp. 393-398, 2011.

[125] L.C. Briand, K. El Emam, D. Surmann, I. Wieczorek and K.D. Maxwell, "An assessment and comparison of common software cost estimation modeling techniques," in Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No. 99CB37002), pp. 313-323, 1999.

[126] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," Information and Software Technology, vol. 54, pp. 820-827, 2012.

[127] M. Azzeh, A.B. Nassif, S. Banitaan and F. Almasalha, "Pareto efficient multi-objective optimization for local tuning of analogy-based estimation," Neural Computing and Applications, vol. 27, pp. 2241-2265, 2016.

[128] M. Azzeh, A.B. Nassif and I.B. Attili, "Predicting software effort from use case points: A systematic review," Science of Computer Programming, vol. 204, pp. 102596, 2021.

[129] A. De Myttenaere, B. Golden, B. Le Grand and F. Rossi, "Mean absolute percentage error for regression models," Neurocomputing, vol. 192, pp. 38-48, 2016.

LIST OF PUBLICATIONS

Journal:

1. **Hai V.V**, H.L.T.K. Nhung, Z. Prokopova, R. Silhavy, and P. Silhavy, "A New Approach to Calibrating Functional Complexity Weight in Software Development Effort Estimation," Computers 11, no. 2: 15, DOI: 10.3390/computers11020015, 2022.
2. **Hai V.V.**, H.L.T.K. Nhung, Z. Prokopova, R. Silhavy, and P. Silhavy, "Towards improving the efficiency of software development effort estimation via clustering analysis," IEEE Access, vol. 10, pp. 83249-83264, DOI: 10.1109/ACCESS.2022.3185393, 2022.
3. H.LT.K. Nhung, **V.V. Hai**, R. Silhavy, Z. Prokopova, and P. Silhavy, "Parametric Software Effort Estimation Based on Optimizing Correction Factors and Multiple Linear Regression, " IEEE Access, vol. 10, pp. 2963-2986, DOI: 10.1109/ACCESS.2021.3139183, 2022.

Conference:

4. **V.V. Hai**, H.L.T.K. Nhung, Z. Prokopova, R. Silhavy, and P. Silhavy, “Analysing the effectiveness of the Gaussian Mixture Model clustering algorithm in Software Enhancement Effort Estimation,” ACIIDS 2022.
5. **V.V. Hai**, Mohsin Javed, Zuhair Abbas, Meryem Ari, and Michaela Bílá, "On the Software Projects' Duration Estimation using Support Vector Regression," Software Engineering Perspectives in Intelligent Systems, CoMeSySo 2022, Advances in Intelligent Systems and Computing, Springer, Cham.
6. **V.V. Hai**, H.L.T.K. Nhung, and R. Jasek, "Toward applying agglomerative hierarchical clustering in improving the software development effort estimation," Lecture Notes in Networks and Systems, vol. 501 LNNS, pp. 353-371, DOI: 10.1007/978-3-031-09070-7_30, 2022.
7. **V.V. Hai**, H.L.T.K. Nhung, H.T. Hoc, "Calibrating Function Complexity in Enhancement Project for Improving Function Points Analysis Estimation," Lecture Notes in Networks and Systems, 232 LNNS, pp. 857-869, DOI: 10.1007/978-3-030-90318-3_67, 2021.
8. **V.V. Hai**, H.L.T.K. Nhung, H.T. Hoc, "Empirical Evidence in Early-Stage Software Effort Estimation Using Data Flow Diagram," Lecture Notes in Networks and Systems, 230, pp. 632-644, DOI: 10.1007/978-3-030-77442-4_53, 2021.
9. **V.V. Hai**, H.L.T.K. Nhung, H.T. Hoc, "A Productivity Optimising Model for Improving Software Effort Estimation," Advances in

Intelligent Systems and Computing, 1294, pp. 735-746, DOI: 10.1007/978-3-030-63322-6_62, 2020.

10. **V.V. Hai**, H.L.T.K. Nhung, H.T. Hoc, "A Review of Software Effort Estimation by Using Functional Points Analysis," Advances in Intelligent Systems and Computing, 1047, pp. 408-422, DOI: 10.1007/978-3-030-31362-3_40, 2019.
11. H.LT.K. Nhung, **V.V. Hai**, and R. Jasek, "Towards a Correction Factors-based Software Productivity using Ensemble approach for Early Software Development Effort Estimation," Lecture Notes in Networks ans Systems, vol. 501 LNNS, pp. 413-425, DOI: 10.1007/978-3-031-09070-7_35, 2022.
12. H.LT.K. Nhung, **V.V. Hai**, and H.T. Hoc, "Analyzing Correlation of the relationship between Technical Complexity Factors and Environmental Complexity Factors for Software Development Effort Estimation", Lecture Notes in Networks and Systems, 232 LNNS, pp. 835-848, DOI: 10.1007/978-3-030-90318-3_65, 2021.
13. H.LT.K. Nhung, **V.V. Hai**, and H.T. Hoc, "Evaluation of Technical and Environmental Complexity Factors for Improving Use Case Points Estimation," Advances in Intelligent Systems and Computing Springer, 1294, pp. 757–768, DOI: 10.1007/978-3-030-63322-6_64, 2020.
14. H.L.T.K. Nhung, H.T. Hoc, and **V.V. Hai**, "A Review of Use Case-Based Development Effort Estimation Methods in the System Development Context," Advances in Intelligent Systems and Computing, 1046, pp. 484-499, DOI: 10.1007/978-3-030-30329-7_44, 2019.
15. H.T. Hoc, **V.V. Hai**, H.L.T.K. Nhung, "An Approach to Adjust Effort Estimation of Function Point Analysis," Lecture Notes in Networks and Systems, 230, pp. 522-537, DOI: 10.1007/978-3-030-77442-4_45, 2021.
16. H.T. Hoc, **V.V. Hai**, H.L.T.K. Nhung, "AdamOptimizer for the Optimisation of Use Case Points Estimation," Advances in Intelligent Systems and Computing, 1294, pp. 747-756, 2020.
17. H.T. Hoc, **V.V. Hai**, H.L.T.K. Nhung, " A Review of the Regression Models Applicable to Software Project Effort Estimation," Advances in Intelligent Systems and Computing, 1047, pp. 399-407, 10.1007/978-3-030-31362-3_39, 2019.

CURRICULUM VITAE

Name and tile:	Vo Van Hai, M.Sc.
Date and place of birth:	23/06/1977, Quang Nam, Vietnam.
Address:	Nam T.G. Masaryk 1281, 76001, Zlín, Czechia.
Email:	vo_van@utb.cz, vovanhai@iuh.edu.vn.

Education:

2018-Recent	Tomas Bata University in Zlín, Czech Republic. Doctoral studies, Engineering Informatics, Software Engineering.
2011-2013	Master of Computer Science. Information Technology Faculty, Hue University of Science, Vietnam.
1996-2000	Bachelor of Information System. Information Technology Faculty, University of Dalat, Vietnam.

Work experience

Lecturer, Faculty of Software Engineering, University of Industrial, Ho Chi Minh City, Vietnam, 2001-2014.

Visited lecturer, 2004-2017.

- Faculty of Information Technology, Tran Dai Nghia University, Ho Chi Minh City, Vietnam.
- VTC Academy, Vietnam.
- Faculty of Information Technology, Gia Dinh University, Ho Chi Minh City, Vietnam.
- FPT University and FPT Aptech, Vietnam

Software Developer, Monet+, Zlín, Czech Republic 2019-2021.

Consultant, Manulife Vietnam, 2014-2017.

Isolation Developer, 2008-2014

Manager, Nhan Tri Hop company, Vietnam, 2004-2008

Language skills

English – proficient in spoken and written

French – basic communication skills.