A Project report on

ENHANCEMENT OF DECISION TREE FOR SOFTWARE COST ESTIMATION

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

K. Dhathriguptha
(20H51A05J9)
B.Manichandana
(20H51A05N0)
M.Laya
(20H51A05P4)

Under the esteemed guidance of
Dr.V. Venkataiah
(Associate Professor of CSE
Additional controller of Examination)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
(UGC Autonomous)
*Approved by AICTE *Affiliated to JNTUH *NAAC Accredited with A^+ Grade
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2020- 2024

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project Phase I report entitled **"ENHANCEMENT OF DECISION TREE FOR SOFTWARE COST ESTIMATION"** being submitted by K. Dhathri Guptha (20H51A05J9), B. Manichandana (20H51A05N0), M. Laya (20H51A05P4) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Dr. V. Venkataiah
Associate Professor of CSE &
Additional controller of examination
Dept. Of CSE

Dr. Siva Skandha Sanagala
Associate Professor & hod
Dept. Of CSE

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Dr. V Venkataiah,** Professor & Additional controller of examination, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

K. Dhathri Guptha 20H51A05J9
B. Manichandana 20H51A05N0
M. Laya 20H51A05P4

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Effort estimation of the software is one of the biggest problems faced by software industry. It includes planning, supervising projects, analysing viability of the system development, preparing proposals and proposals presentation to clients and among this estimation of the effort for planning is one of the most critical responsibilities. It is necessary to have good effort estimation in order to conduct well budget. The accuracy of the effort estimation of software projects is vital for the competitiveness of software companies. For the forecasting of software effort, it is important to select the correct software effort estimation techniques. Numbers of have been proposed like algorithm, non-algorithm, Machine Learning algorithms and so on. Generally, Decision Trees have drawn the attention of the researchers and changed the direction of Effort Estimation towards computational intelligence.in this project we are using decision tree concept which is a simple yet powerful approach and it can be taken as the base for Random Forest, which is known as a group of decision trees. Decision Trees facing data over fitting problem. To overcome this problem enhancement of Decision Tree with ensemble method proposed in this project and also for evaluate performance of this proposed method using **COCOMO 81 dataset**. **(**or) International Software Benchmarking Standard Groups (ISBSG) data set and to evaluation metrics like Mean Magnitude Relative Error (MMRE) and Root Mean Square Error (RMSE). Comparative study shows that proposed technique, achieves better results than previous techniques

CHAPTER 1
INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1. Problem Statement

In this project, the main aim is to improve software effort estimation by constructing a decision tree that makes use of several machine learning methods on software effort estimation datasets. In this project using COCOMO 81 dataset, we are estimating and calculate the accurate data as compare with other approaches. Software quality is driven by many aspects in software development. In software development organizations, the main aim is to start a project and finish it within acceptable schedule and budget. The main drivers affecting the budget are the number of employees and time. These two come together to form a measure called effort. Effort is the most important factor in a software project determining the cost. Estimating effort is a very important factor affecting the quality of software. At early stages of software development, effort must be estimated to come up with a planned schedule and budget. Traditional software cost estimation methods lack accuracy and reliability. The need for more effective and data-driven approaches is evident. That the reason we come up with this idea

1.2. Research Objective

The research objective for this study is to develop a machine learning-based software cost estimation model using Decision Trees. This model aims to enhance the accuracy of cost predictions for software development projects while maintaining transparency and interpretability. The primary goal is to leverage the data-driven capabilities of Decision Trees to improve cost estimation, ultimately contributing to more informed decision-making in project planning, resource allocation, and budgeting. The research objective for Software cost estimation using decision tree in machine learning are follows:

-Model Development -Data-Driven Insights

-Accuracy Enhancement -integration of COCOMO parameters

-Comparative Analysis

By pursuing these research objectives, this study aims to advance the field of software cost estimation by providing a modern, data-driven approach that can complement and enhance existing methodologies, ultimately leading to more informed and efficient software project management practices.

CHAPTER 2
BACKGROUND WORK

CHAPTER 2
BACKGROUND WORK

2.1 COCOMO 81:

2.1.1 Introduction

The COCOMO (Constructive Cost Estimation Model) is proposed by DR. Berry Boehm in 1981 and that's why it is also known as COCOMO'81. It is a method for evaluating the cost of a software package.

According to him software cost estimation should be done through three stages:

- Basic COCOMO model
- Intermediate COCOMO model
- Complete Detailed COCOMO model
- COCOMO1.gif
- COCOMO 81 models depend upon the two main equations
- Development Effort: MM=a*KDS1 b
- Which is based on MM - man-month / person month / staff-month is one month of effort by one person

NOTE: In COCOMO 81, there are 512 hours per person month. According to organization this Values may differ from the standard by 10% to 20% Efforts and development time: TDEV =2.5*s0MMc

NOTE: The coefficients a, b, c depends on the mode of the development.

2.1.2 Advantages & Limitation of COCOMO 81

Advantages: COCOMO is transparent one can see how it works unlike other models such as SLIM. DRIVERS are particularly helpful to the estimator to understand the impact of different factors that affect project costs.

Limitation of COCOMO 81: It is hard to accurately estimate KDSI early on in the project when effort estimates are requires.

KDSI actually is not a size measure it is a length measure. Extremely vulnerable to mis-classification of the development mode. Success depends largely on tuning the model to the needs of the organization, using historical data which is not always available.

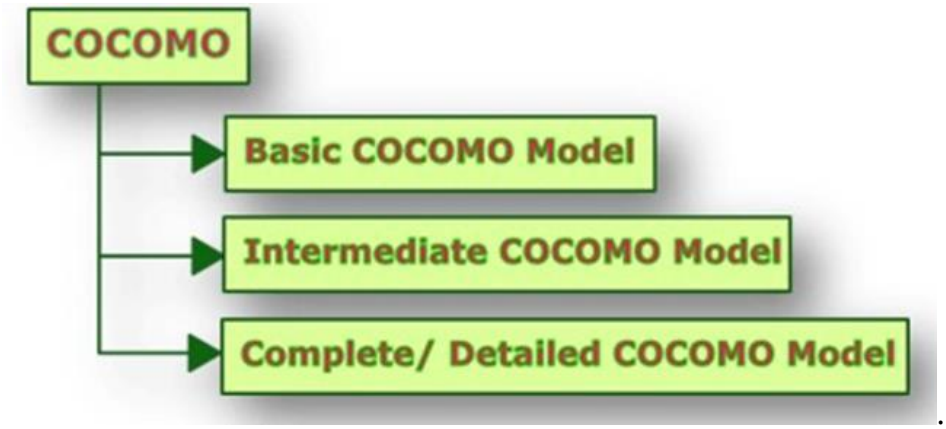


Figure.1 COCOMO MODEL

2.1.3 COCOMO 81 Dataset

rely	data	cplx	time	stor	virt	turn	acap	aexp	pcap	vexp	lexp	modp	tool	sced	loc	actual
0.88	1.16	0.7	1	1.06	1.15	1.07	1.19	1.13	1.17	1.1	1	1.24	1.1	1.04	113	2040
0.88	1.16	0.85	1	1.06	1	1.07	1	0.91	1	0.9	0.95	1.1	1	1	293	1600
1	1.16	0.85	1	1	0.87	0.94	0.86	0.82	0.86	0.9	0.95	0.91	0.91	1	132	243
0.75	1.16	0.7	1	1	0.87	1	1.19	0.91	1.42	1	0.95	1.24	1	1.04	60	240
0.88	0.94	1	1	1	0.87	1	1	1	0.86	0.9	0.95	1.24	1	1	16	33
0.75	1	0.85	1	1.21	1	1	1.46	1	1.42	0.9	0.95	1.24	1.1	1	4	43
0.75	1	1	1	1	0.87	0.87	1	1	1	0.9	0.95	0.91	0.91	1	6.9	8
1.15	0.94	1.3	1.66	1.56	1.3	1	0.71	0.91	1	1.21	1.14	1.1	1.1	1.08	22	1075
1.15	0.94	1.3	1.3	1.21	1.15	1	0.86	1	0.86	1.1	1.07	0.91	1	1	30	423
1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	29	321
1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	32	218
1.15	0.94	1.3	1.11	1.06	1	1	0.86	0.82	0.86	1	0.95	0.91	1	1.08	37	201
1.15	0.94	1.3	1.11	1.06	1.15	1	0.71	1	0.7	1.1	1	0.82	1	1	25	79
1.15	0.94	1.65	1.3	1.56	1.15	1	0.86	1	0.7	1.1	1.07	1.1	1.24	1.23	3	60
1.4	0.94	1.3	1.3	1.06	1.15	0.87	0.86	1.13	0.86	1.21	1.14	0.91	1	1.23	3.9	61
1.4	1	1.3	1.3	1.56	1	0.87	0.86	1	0.86	1	1	1	1	1	6.1	40
1.4	1	1.3	1.3	1.56	1	0.87	0.86	0.82	0.86	1	1	1	1	1	3.6	9
1.15	1.16	1.15	1.3	1.21	1	1.07	0.86	1	1	1	1	1.24	1.1	1.08	320	11400
1.15	1.08	1	1.11	1.21	0.87	0.94	0.71	0.91	1	1	1	0.91	0.91	1	1150	6600
1.4	1.08	1.3	1.11	1.21	1.15	1.07	0.71	0.82	1.08	1.1	1.07	1.24	1	1.08	299	6400
1	1.16	1.15	1.06	1.14	0.87	0.87	0.86	1	1	1	1	0.91	0.91	1	252	2455
1.15	1	1	1.27	1.06	1	1	0.86	0.82	0.86	0.9	1	0.91	1	1.23	118	724
1.15	1	1	1.08	1.06	1	1	0.86	0.82	0.86	0.9	1	1	1	1.23	77	539
0.88	1	0.85	1.06	1.06	1	0.87	1	1.29	1	1.1	0.95	0.82	0.83	1	90	453
1.15	1.16	1.3	1.15	1.06	1	0.87	0.86	1	0.86	1.1	1	0.82	0.91	1.08	38	523
0.94	1	0.85	1.07	1.06	1.15	1.07	0.86	1	0.86	1.1	1	0.91	1.1	1.08	48	387
1.15	0.94	1.15	1.35	1.21	1	0.87	1	1	1	1	1	0.82	1.1	1.08	9.4	88
1.15	1.08	1.3	1.11	1.21	1.15	1.07	0.86	1	0.86	1.1	1.07	1.1	1.1	1	13	98
0.88	1	1	1	1	1	1	1.1	1.29	0.86	1	1	0.91	0.91	1.23	2.14	7.3
0.88	1	1	1	1	1	1	1	1.29	0.86	1	1	0.91	0.91	1.23	1.98	5.9
1.4	1.08	1	1.48	1.56	1.15	1.07	0.86	0.82	0.86	1.1	1.07	1	1	1	62	1063
0.88	1.08	0.85	1	1	1	1	0.71	0.82	1	1	1	1.1	1.1	1	390	702
1.4	1.08	1.3	1.48	1.56	1.15	0.94	0.86	0.82	0.86	0.9	1	0.91	0.91	1	42	605
1.15	1.08	1	1.06	1	1	0.87	1	1	1	1	1	0.91	1.1	1.23	23	230
0.75	0.94	1.3	1.06	1.21	1.15	1	1	0.91	1	1.1	1	1.24	1.24	1	13	82
0.88	1.05	0.81	1	1	0.87	0.87	1.19	1	1.17	0.9	0.95	1	0.91	1.04	15	55
0.88	0.94	0.7	1	1.06	1	1	0.86	0.82	0.86	1	1	1	1	1	60	47
1	1	1.15	1	1	0.87	0.87	0.71	0.91	1	0.9	0.95	0.82	0.91	1	15	12
1	1	1.15	1	1	0.87	1	0.71	0.82	0.7	1	0.95	0.91	1.1	1	6.2	8
1	0.94	1.3	1	1	1	0.87	0.86	0.82	1.17	1	1	1.1	1	1	3	8
0.88	0.94	1	1	1	0.87	0.87	1	0.82	0.7	0.9	0.95	0.91	0.91	1	5.3	6
0.88	1.04	1.07	1	1.06	0.87	1.07	0.86	1	0.93	0.9	0.95	0.95	0.95	1.04	45.5	45
1	1.04	1.07	1	1.21	0.87	1.07	0.86	1	1	0.9	0.95	1	1	1.04	28.6	83
0.88	1.04	1.07	1.06	1.21	0.87	1.07	1	1	1	0.9	0.95	1.1	1	1.04	30.6	87
0.88	1.04	1.07	1	1.06	0.87	1.07	1	1	1	0.9	0.95	1	0.95	1.04	35	106
0.88	1.04	1.07	1	1.06	0.87	1.07	1	1	0.86	0.9	0.95	1	1	1.04	73	126
0.75	0.94	1.3	1	1	0.87	0.87	0.71	0.82	0.7	1.1	1.07	1.1	1	1.04	23	36
0.88	0.94	0.85	1	1	0.87	1	1.19	0.91	1.17	0.9	0.95	1.1	1	1.04	464	1272
1	1	0.85	1	1	1	0.87	0.71	1	0.7	1.1	1	0.82	0.91	1	91	156
1.15	1	1	1.3	1.21	1	0.87	0.86	1	0.86	1.1	1	1	1	1	24	176
0.88	1	1	1	1	1	1.15	1.19	1	1.42	1	0.95	1.24	1.1	1.04	10	122
0.88	0.94	0.85	1	1.06	1.15	1	1	1	1	1.1	1.07	1.24	1.1	1	8.2	41
0.88	0.94	1.15	1.11	1.21	1.3	1	0.71	1	0.7	1.1	1.07	1	1.1	1.08	5.3	14
1	0.94	1	1	1.06	1.15	0.87	1	0.82	1	1	0.95	0.91	1.1	1	4.4	20
0.88	0.94	0.7	1	1	0.87	0.87	0.86	0.82	1.17	0.9	0.95	1.1	1	1	6.3	18
1.15	0.94	1.3	1.3	1.21	1	1	0.86	0.91	1	1.1	1.07	1.1	1.1	1.08	27	958
1	0.94	1.15	1.11	1.21	1.3	1	1	1	1	1.1	1.07	1.1	1.1	1.23	17	237
1.4	0.94	1.3	1.66	1.21	1	1	0.71	0.82	0.7	0.9	0.95	0.91	1	1	25	130
1	0.94	1.15	1.06	1.06	1	0.87	1	1	1	1	1	0.91	1	1	23	70
1.15	0.94	1.3	1.11	1.06	1	1	0.86	1.13	0.86	1.1	1.07	1.1	1.1	1.08	6.7	57
1	0.94	1.15	1	1	0.87	0.87	0.86	1	0.86	0.9	1	0.82	1	1	28	50
0.88	0.94	1.3	1.11	1.21	1.15	1	0.78	0.82	0.7	1.21	1.14	0.91	1.24	1	9.1	38
1	0.94	1.15	1	1	1	0.87	0.71	0.82	0.86	1	1	0.82	1	1	10	15

2.2 DECISION TREE

2.2.1 Introduction

A decision tree is a flowchart-like tree structure where each internal node denotes the feature, branches denote the rules and the leaf nodes denote the result of the algorithm. It is a versatile supervised machine-learning algorithm, which is used for both classification and regression problems. It is one of the very powerful algorithms. And it is also used in Random Forest to train on different subsets of training data, which makes random forest one of the most powerful algorithms in machine learning.

Root Node: It is the topmost node in the tree, which represents the complete dataset. It is the starting point of the decision-making process.

Decision/Internal Node: A node that symbolizes a choice regarding an input feature. Branching off of internal nodes connects them to leaf nodes or other internal nodes.

Leaf/Terminal Node: A node without any child nodes that indicates a class label or a numerical value.

Splitting: The process of splitting a node into two or more sub-nodes using a split criterion and a selected feature.

Branch/Sub-Tree: A subsection of the decision tree starts at an internal node and ends at the leaf nodes.

Parent Node: The node that divides into one or more child nodes.

Child Node: The nodes that emerge when a parent node is split.

Impurity: A measurement of the target variable's homogeneity in a subset of data. It refers to the degree of randomness or uncertainty in a set of examples. The **Gini index** and **entropy** are two commonly used impurity measurements in decision trees for classifications task

Variance: Variance measures how much the predicted and the target variables vary in different samples of a dataset. It is used for regression problems in decision trees. **Mean squared error, Mean Absolute Error, Friedmans, or Half Poisson deviance** are used to measure the variance for the regression tasks in the decision tree.

Information Gain: Information gain is a measure of the reduction in impurity achieved by splitting a dataset on a particular feature in a decision tree. The splitting criterion is determined by the feature that offers the greatest information gain, It is used to determine the most informative feature to split on at each node of the tree, with the goal of creating pure subsets

Pruning: The process of removing branches from the tree that do not provide any additional information or lead to overfitting.

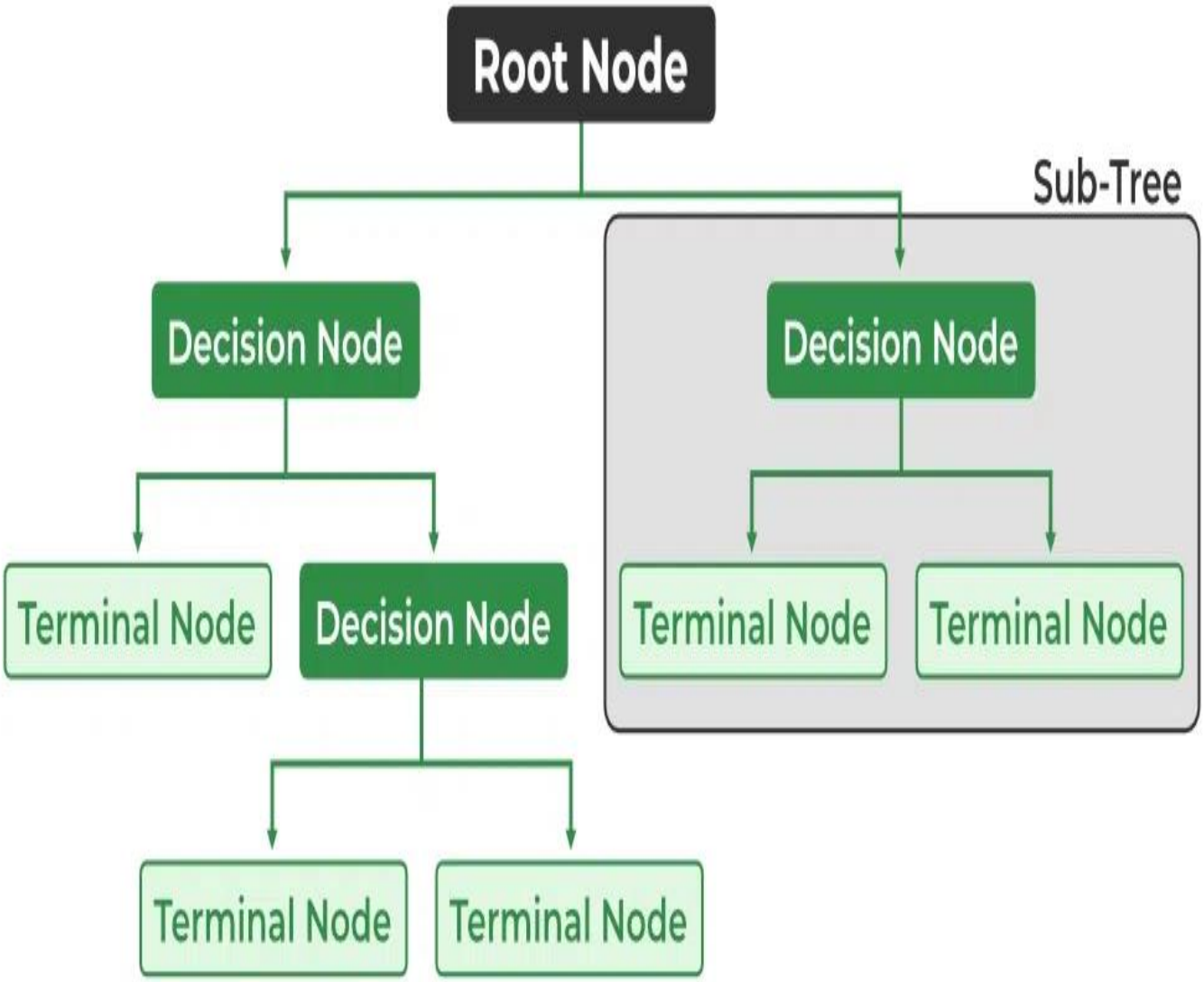


Fig.2 Decision Tree

2.2.2 Advantages and Disadvantages of the Decision Tree

Advantages:

1. It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
2. It can be very useful for solving decision-related problems.
3. It helps to think about all the possible outcomes for a problem.
4. There is less requirement of data cleaning compared to other algorithms

Disadvantages:

1. The decision tree contains lots of layers, which makes it complex.
2. It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
3. For more class labels, the computational complexity of the decision tree may increase.

2.2.3Evaluation of Metrics

Root Mean Square Error (RMSE)

Root Mean Square Error (RMSE) is a commonly used metric for evaluating the accuracy of a predictive model, particularly in the context of regression analysis. It measures the average magnitude of the errors between predicted values and actual values in a dataset. RMSE is a popular choice because it penalizes larger errors more heavily than smaller ones, making it sensitive to outliers.

The formula for calculating RMSE is as follows:

$RMSE = \sqrt{(\Sigma (yi - \hat{y}i)^2 / n)}$

Where:

- yi represents the actual values in the dataset.
- $\hat{y}i$ represents the predicted values generated by your model.
- n is the total number of data points in the dataset.

Here's how you can calculate RMSE step by step:

1. For each data point in your dataset, calculate the squared difference between the actual value (yi) and the predicted value (ŷi).
2. Sum up all these squared differences.
3. Divide the sum by the total number of data points (n) to calculate the mean squared error.
4. Take the square root of the mean squared error to obtain the RMSE.

A lower RMSE value indicates that the model's predictions are closer to the actual values, suggesting better predictive performance. Conversely, a higher RMSE value implies that the model's predictions are further from the actual values, indicating poorer predictive performance.

RMSE is often used in the fields of machine learning, statistics, and data analysis to assess the quality of regression models, such as linear regression, decision trees, and neural networks. It provides a single value that summarizes how well a model fits the data, making it a useful tool for model selection and evaluation.

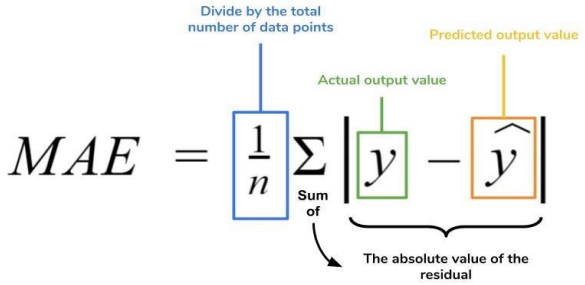
Mean Square Error (MSE)

The mean square error (MSE) is just like the MAE but squares the difference before summing them all instead of using the absolute value. We can see this difference in the equation below. MSE Equation

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

Mean Absolute Error (MAE)

The mean absolute error (MAE) is the simplest regression error metric to understand. We'll calculate the residual for every data point, taking only the absolute value of each so that negative and positive residuals do not cancel out. We then take the average of all these residuals. Effectively, MAE describes the typical magnitude of the residuals. The formal equation is shown below:



2.2.4 Process for software estimation

Software estimation is a crucial step in software development, as it helps to determine the cost and effort required to complete a project. There are several methods for software estimation, but one of the most widely used models is the COCOMO model. This model estimates the cost and effort required for software development based on the size of the project, the complexity of the software, and other factors.

Here are some general steps to follow when using the COCOMO model for software estimation:

- **Define the scope of the project:** This includes identifying the features and functionalities that need to be included in the software.
- **Determine the size of the project:** This can be done by measuring the lines of code or function points required for the project.
- **Identify the complexity of the software**: This includes determining factors such as database complexity, user interface complexity, and algorithmic complexity.
- **Calculate the effort required:** This can be done using a formula that takes into account the size and complexity of the project.
- **Estimate the cost:** The cost can be estimated by multiplying the effort required by an hourly rate.

2.3 Implementation of Existing System

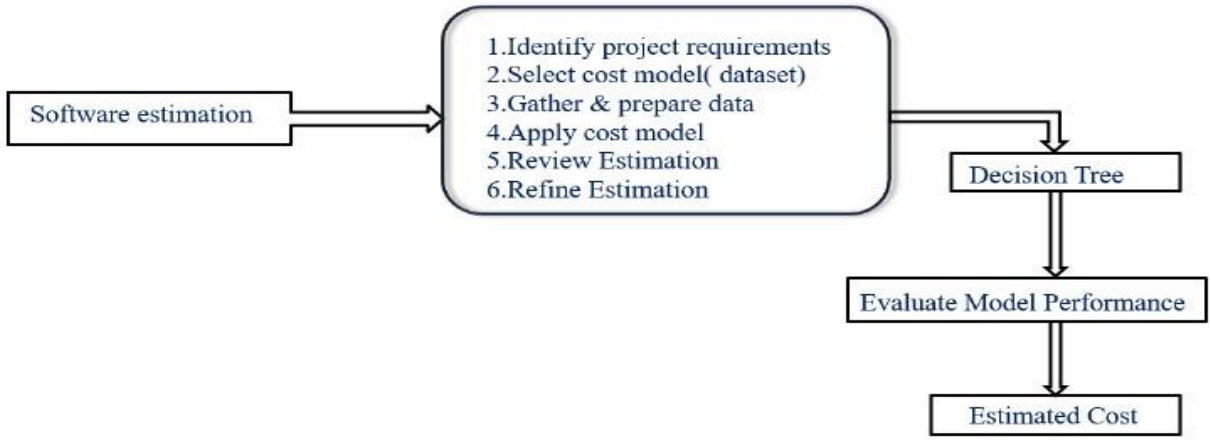


Fig.3 Block Diagram of Existing System

2.3.2 Executed Code

```
[1]: import pandas as pd

[2]: df=pd.read_csv("cocomo 81.csv")

[3]: df.head()
```

[3]:

	Rely	Data	Cplx	Time	Stor	Virt	Turn	Acap	Aexp	Pcap	Vexp	Lexp	\
0	0.88	1.16	0.70	1.0	1.06	1.15	1.07	1.19	1.13	1.17	1.1	1.00	
1	0.88	1.16	0.85	1.0	1.06	1.00	1.07	1.00	0.91	1.00	0.9	0.95	
2	1.00	1.16	0.85	1.0	1.00	0.87	0.94	0.86	0.82	0.86	0.9	0.95	
3	0.75	1.16	0.70	1.0	1.00	0.87	1.00	1.19	0.91	1.42	1.0	0.95	
4	0.88	0.94	1.00	1.0	1.00	0.87	1.00	1.00	1.00	0.86	0.9	0.95	

	Modp	Tool	Sced	Size	Effort1
0	1.24	1.10	1.04	113.0	2040.0
1	1.10	1.00	1.00	293.0	1600.0
2	0.91	0.91	1.00	132.0	243.0
3	1.24	1.00	1.04	60.0	240.0
4	1.24	1.00	1.00	16.0	33.0

```
[4]: X = df.drop(columns=['Effort1'])
     Y = df['Effort1']

[5]: from sklearn.model_selection import train_test_split
     # Split the data into features (X) and the target variable (Y)
     # Assuming you've already split the data as shown in the previous response
     # Split the data into training and testing sets
     X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,␣
      ↪random_state=42)

[6]: from sklearn.tree import DecisionTreeRegressor

[7]: from sklearn.model_selection import train_test_split

[12]: regressor = DecisionTreeRegressor(random_state=7)

[9]: regressor.fit(X_train, Y_train)

[9]: DecisionTreeRegressor(random_state=42)

[10]: Y_pred = regressor.predict(X_test)

[11]: from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

      # Calculate regression metrics
      mae = mean_absolute_error(Y_test, Y_pred)
      mse = mean_squared_error(Y_test, Y_pred)
      r2 = r2_score(Y_test, Y_pred)

      print("Mean Absolute Error: ", mae)
      print("Mean Squared Error: ", mse)
      print("R-squared (R^2): ", r2)

Mean Absolute Error:  304.7692307692308
Mean Squared Error:  351599.53846153844
R-squared (R^2):  -0.13777512821729432
```

Fig.4 Implementation of Decision Tree

CHAPTER 3
RESULTS AND DISCUSSION

CHAPTER 3

RESULTS AND DISCUSSION

- We present the outcomes of Existing machine learning model.
- We are tried to evaluate its accuracy, efficiency and relative errors.
- We compare its performance with traditional methods and existing other methods, showcasing how it outperforms them.
- We share key findings and insights from our evaluation.
- From this, we are saying that we can do more enhancement of decision tree implementation to get more accuracy and efficiency for software estimation.

CHAPTER 4
CONCLUSION

CHAPTER 4

CONCLUSION

We summarize the main findings from our presentation. We underscore the transformative potential of machine learning, specifically decision trees, in elevating software cost estimation. We highlight how this innovation can lead to more effective project management and resource allocation.

This project can estimate the effort and cost using decision tree in machine learning. The estimating is done using enhancement of decision tree in machine learning technique using COCOMO 81 dataset. From this , we conclude that the decision tree in machine learning model can be used with promising results to predict software effort.

CHAPTER 5
REFERENCES

REFERENCES

1. Software Engineering | COCOMO Model – GeeksforGeeks
2. ChatGPT (openai.com)
3. https://gamma.app/
4. Machine Learning Models for Software Cost Estimation | IEEE Conference Publication | IEEE Xplore
5. IEEE Xplore
6. Wikipedia
7. https://sci-hub.se/
8. Bilge Baseless, Burak Turhan, Ayşe Bener Department of Computer Engineering, Bouazizi University bilge.baskeles@boun.edu.tr, turhanb@boun.edu.tr, bener@boun.edu.tr
9. ICCIT-2013: Special Session-Computational Intelligence Applications in Software Engineering (CIASE), Beirut