IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

House construction cost prediction

Subhaan Jirait, Shreyas Rede

Author 1, Author 2

APCOER, PUNE

ABSTRACT

This project presents a comprehensive approach to predicting construction costs for residential buildings using machine learning techniques. With the real estate market's dynamic nature and the various factors influencing construction costs, accurate prediction models are essential for stakeholders, including builders, architects, and potential homeowners. The project utilizes a dataset comprising various attributes such as the number of bedrooms, square footage, number of floors, and location, among others.

The project also addresses technical, operational, legal, and schedule feasibilities, ensuring the solution is viable and practical for real-world applications. Performance, safety, and security requirements are considered to guarantee the robustness and reliability of the prediction system. The implementation section details the integration of the GUI with backend services and the machine learning model, providing a step-by-step guide to the development process.

This document includes a thorough evaluation and comparison of different machine learning models, highlighting their respective accuracies and features. The project concludes with insights gained from the predictive model's performance and suggestions for future improvements. This comprehensive approach underscores the potential of machine learning in enhancing decision-making processes in the construction industry.

CHAPTER 1

INTRODUCTION

House construction cost estimation plays a pivotal role in the dynamics of the real estate market and shapes the decisions of various stakeholders involved in the construction process. Understanding and accurately predicting the cost of building a house is essential for developers, contractors, architects, lenders, and prospective homeowners alike. It serves as the foundation for budgeting, pricing, financing, and project planning activities, influencing the feasibility and success of construction ventures.

In response to these challenges, the application of machine learning (ML) and data analytics presents a promising solution for improving the accuracy and efficiency of construction cost estimation. Machine learning models can analyze vast amounts of historical data, identify patterns and correlations, and predict costs with higher precision. By continuously learning from new data, these models can adapt to changing market conditions and provide real-time insights.

Problem Definition and Scope:-

The complexity of house construction cost estimation arises from the multifaceted nature of construction projects and the myriad factors that influence pricing. These factors include but are not limited to:

•Location: The geographical location of a construction project significantly impacts costs due to variations in land prices, labor availability, regulatory requirements, and material sourcing logistics.

•Size and Scope: The size and complexity of a house construction project, including the number of floors, rooms, and custom features, directly affect construction costs.

•Materials and Quality Standards: The choice of construction materials and adherence to quality standards significantly influence costs.

•Labor and Skill Level: Labor costs constitute a significant portion of construction expenses, with wages varying based on skill level, experience, and regional labor market conditions.

•Market Conditions: Real estate market dynamics, including supply and demand fluctuations, interest rates, and economic indicators, exert a profound influence on construction costs.

In response to the challenges posed by traditional approaches, there is a growing interest in leveraging data-driven techniques and machine learning algorithms to develop predictive models for house construction cost estimation. By harnessing the power of big data analytics, predictive modeling, and artificial intelligence, these advanced techniques offer the potential to improve the accuracy, efficiency, and transparency of cost estimation processes.

Aim and Objective

The primary aim of this project is to develop a robust and accurate prediction model for house construction costs, leveraging advanced machine learning algorithms. Specifically, our objectives include:

Gathering and preprocessing relevant data on house construction projects, including project specifications, materials used, labor costs, and geographic location.

Exploring and selecting appropriate features for inclusion in the prediction model, considering factors such as cost drivers and market indicators.

Implementing machine learning algorithms to train and optimize the prediction model based on historical cost data and project characteristics.

Integrating the prediction model into a user-friendly graphical interface using Tkinter, enabling users to input project parameters and obtain real-time cost estimates.

Evaluating the performance of the prediction model through rigorous testing and validation, comparing predicted costs against actual project expenditures.

Providing insights and recommendations for further refinement and improvement of the prediction model, based on analysis of model outputs and user feedback.

By achieving these objectives, we aim to develop a valuable tool for stakeholders in the construction and real estate industries, facilitating more accurate and informed decision-making in house construction projects. Our ultimate goal is to contribute to the advancement of cost estimation methodologies and enhance efficiency and transparency in the construction process.

CHAPTER 2
LITERATURE SURVEY

STUDY OF RESEARCH PAPER

A research paper is a document of a scientific article that contains relevant ex- per-tise, including substantive observations, and also references to a specific subject of philosophy and technique. Use-secondary references are reviewed in literature and no current or initial experimental work is published.

1.Paper Name: Cost Estimation of Civil Construction Projects using Machine Learning Paradigm **Author:**Kanimozhi N , Keerthana N V , Pavithra G S , Ranjitha G ,Yuvarani S **Abstract ::-** Adequate construction cost estimation is main factor in any type ofconstruction projects. Forecasting cost of construction projects can be consideredas difficult task. In this paper, Ordinary Least Square (OLS) method which is typeofa simple linear regression model is proposed to forecast future cost of constructionprojects. Ordinary Least Square method can be used to produce best solution andperforms well when the dataset is small. Experiments were performed with 12 yearsof District Schedule Rates of Pune region from India to find out the accuracy of the model. The results show that proposed model gives 91accuracy.

2.Paper Name: :-Evaluation of Several Machine Learning Models for Field Canal Improvement Project Cost Prediction

Author:B. Sivanagaleela,M.Tech, S.Rajesh, Asst.Professor

Abstract : Project cost prediction is one of the key elements in the civil engineer- ing activities development. Project cost is a highly sensitive component to diverse parameters and hence it is associated with complex trends that make it difficult to be predicted and fully understood. Due to the massive advancement of soft computing (SC) and Internet of things (IoT), the main research objective of the current study was initiative. Several machine learning (ML) models including extreme learning machine (ELM), multivariate adaptive regression spline (MARS), and partial least square regression (PLS) were adopted to predict field canal cost. Several essen- tial predictors were used to develop the prediction network "the learning process" including the total length of the PVC pipeline, served area, geographical zone, con- struction year, and cost and duration of field canal improvement projects (FCIP) construction. Data were collected from the open source published literature. *e modeling results evidenced the potential of the applied SC models in predicting the FCIP cost. In numerical magnitude evaluation, MARS model indicated the least value for the root mean square error (RMSE 27422.7), mean absolute error (MAE 19761.8), and mean absolute percentage error (MAPE 0.05454) with Nash-Sutcliffe efficiency (NSE 0.94), agreement index (MD 0.89), and coefficient of determination (R2 0.94), with best precision of prediction using all predictors, except geographical zone parameter in which less influence on the cost construction is presented. In gen- eral, the research outcome gave an informative primary cost initiative for cost civil engineering project

3.Paper Name: Construction Cost Prediction Using Deep Learning with BIM Properties in the Schematic Design Phase

Author name: Akash Kumar, Aniket Verma, Gandhali Shinde.

Abstract :: In the planning and design stage, it is difficult to accurately predict con- struction costs only by estimating approximate cost. It is also very difficult to predict the change in construction costs whenever the design changes. However, using the BIM model's attribute information and machine learning techniques, accurate con- struction costs can be predicted faster than when using the existing approximate cost estimate. In this study, building information such as 'total area', 'floor water', 'usage', and BIM attribute information such as 'wall area', 'wall water', and 'floor circumference' were used together to predict construction costs in the schema design stage. As a result of applying the machine learning technique using both the building design information and the BIM model attribute information, it was found that the construction cost was improved compared to the result of individual predictions of the building information or BIM attribute information. While accurately predicting construction costs using BIM's attribute information has its limits, it is expected to provide more accuracy compared to predicting costs solely based on construction cost influencing factors.

4.Paper Name: TOWARDS AUTOMATED COST ANALYSIS, BENCHMARK- ING AND ESTIMATING IN CONSTRUCTION: A MACHINE LEARNING AP- PROACH

Author::- ANAS AMJAD, (Student Member, IEEE), ALISON GRIFFITHS, AND MOHAMMAD PATWARY, (Senior Member, IEEE)

abstract : In this paper, a novel machine learning based approach is proposed for automated cost analysis from priced bill of quantities prepared by tenders in the construction industry. The proposed approach features: 1) An effective integration of structured project-specific information with surveyor's domain knowledge in order to model the complex interrelationships between the specifications and descriptions of an item and its trade category; 2) An effective transformation to map the original data into a 2-dimensional space to tackle issues of high dimensionality in modelling, and 3) Simply classifiers with good classification capability. Relevant comparative experimental results have demonstrated the effectiveness of the proposed approach

5.Paper Name:Machine learning algorithms for constructions cost prediction: A systematic review

Author:WAJIHA SAFAT , SOHAIL ASGHAR , (Member, IEEE), AND SAIRA ANDLEEB GILLANI

Abstract:- Machine learning plays a vital role in construction estimation which could make improve the project's safety, and reliability. Many studies have been proposed to explore the potential opportunities to review this technology in the construction cost in structure and transport fields. However, no comprehensive study to review the global research trends on this area's advancement in construction cost. The goal is to taxonomy, review, and summarize the state-ofthe-art knowledge body on this topic in a systematic manner based on machine learning (ML) and deep learning (DL) approaches. To achieve this, this paper considered many studies in construction management related to bibliographic records retrieved from the Scopus database by adopting a quantitative analysis approach. This paper found that from 2017 to 2021, there has been a considerable increase in the number of publications in this domain. We categorized and explained civil projects into structures and transport cost, ML/DL as supervised and unsupervised approaches, and the evaluation metrics proposed to evaluate the performance of ML-Cost estimations in the civil area. The findings will help both professionals and researchers to understand and evolve the recent trend research ML/DL methodologies and their role played in the construction management domain.

6.Paper name- "Performance Analysis of Construction Cost Prediction Using Neural Network for Multioutput Regression"

Authors: Supachart Janyavula, Anucha Apichatpaiboon, Praphon Wongkamjan, Prayoot Akkaraekthalin

Abstract: This study addresses the persistent challenges in accurately predicting construction costs, which are influenced by a myriad of factors such as material prices, labor costs, and project-specific variables. The researchers employ Artificial Neural Networks (ANNs) integrated with a multioutput regression model to predict both detailed sub-construction costs and overall construction costs. By using a comprehensive dataset encompassing various types of construction projects, the study trains and validates the neural network model, achieving a prediction error rate of 16.80% for sub-construction costs and 17.67% for total costs. These findings underscore the potential of ANNs in enhancing the precision of cost estimations, thereby providing a robust tool for project managers and stakeholders to better plan and allocate resources.

7.Paper Name : "Predictive Analytics for Early-Stage Construction Costs Estimation"

Authors: Sergio Lautaro Castro Miranda, Enrique Del Rey Castillo, Vicente Gonzalez, Johnson Adafin

Abstract: This paper explores the application of predictive analytics to improve the accuracy of construction cost estimates at the early stages of project development. Through an extensive systematic literature review, the authors compile a vast database of previous studies and real-world data, identifying critical cost drivers such as project size, complexity, location, and market conditions. The research highlights the benefits of using machine learning models to process this data, offering enhanced prediction accuracy compared to traditional methods. The study demonstrates that predictive analytics not only helps in estimating costs more accurately but also facilitates better decision-making and risk management for stakeholders, ultimately leading to more efficient project execution and resource utilization.

8.Paper Name - "Machine Learning-Aided Cost Prediction and Optimization in Construction Operations"

Authors: Virok Sharma, Mohd Zaki, Kumar Neeraj Jha, N. M. Anoop Krishnan

Abstract: In this research, various machine learning algorithms, including gradient boosted trees and neural networks, are applied to predict construction costs based on a multitude of parameters such as project duration, resource allocation, and environmental impacts. The study highlights the effectiveness of ensemble methods in capturing complex relationships within the data. Furthermore, the authors utilize optimization techniques like Bayesian optimization and particle swarm optimization to fine-tune the prediction models. The findings suggest that these advanced methodologies significantly enhance cost prediction accuracy, providing construction managers with valuable insights for developing cost-efficient strategies and improving overall project management practices.

9.Paper name-"Forecasting Construction Cost Index Using Interrupted Time-Series"

Authors: Tae Moon, Dong Hwan Shin

Abstract: This study focuses on the dynamic nature of the Construction Cost Index (CCI) and its implications for cost prediction. The authors utilize interrupted time-series analysis to forecast CCI, which is essential for budgeting and planning in construction projects. The methodology involves analyzing historical CCI data, identifying significant interruptions or shifts in trends, and modeling these changes to predict future costs. The findings indicate that interrupted time-series analysis provides a more accurate and reliable forecast compared to traditional time-series methods, particularly in capturing the impact of market fluctuations and economic events. This research offers valuable insights for construction stakeholders, helping them to anticipate cost changes and make informed financial decision.

FEASIBILITY TEST

1.TECHNICAL FEASIBILTY

Programming Language: Python is a robust and widely-used programming language with extensive libraries for data analysis and machine learning, such as pandas, scikit-learn, and TensorFlow.

GUI Framework: Tkinter, a standard GUI library for Python, is suitable for developing desktop applications. It is well-documented and easy to use for creating interactive interfaces.

Machine Learning Models: The use of machine learning models for predicting construction costs is feasible with Python due to its powerful libraries like scikit-learn, which offers a variety of regression and classification algorithms.

Integration: Python's versatility and the ease of integrating Tkinter with machine learning models make it technically feasible to build a comprehensive application for construction cost prediction.

2.OPERATIONAL FEASIBLITY:

User Requirements: The application should be user-friendly, allowing users to input parameters easily and receive cost predictions quickly. Tkinter provides a straightforward way to create such an interface.
Training and Support: Minimal training is required for users to operate the GUI, making it easy to deploy within an organization. Documentation and help features within the application can further enhance usability.
Scalability: The system should handle a reasonable amount of data and provide predictions in a timely manner. Python and its libraries can efficiently manage large datasets, ensuring that the application remains responsive.

3.LEGAL FEASIBLITY:

Data Privacy: Ensure compliance with data privacy laws such as GDPR if the application processes personal data. The application should implement appropriate data protection measures.
Licensing: Verify that all software libraries and tools used (e.g., Python, Tkinter, scikit-learn) are open-source or properly licensed for commercial use.
Intellectual Property: Ensure that the machine learning models and any proprietary algorithms do not infringe on existing patents or intellectual property rights**.**

3.SCHEDULE FEASIBLITY:

Project Plan: Develop a detailed project plan outlining all tasks, milestones, and deadlines. This should include data collection, model training, GUI development, testing, and deployment.

Resource Allocation: Ensure that sufficient resources (e.g., developers, data scientists) are available to complete the project on time.

Risk Management: Identify potential risks that could delay the project (e.g., technical challenges, data availability) and develop mitigation strategies.

CHAPTER 3

REQUIREMENTS AND SPECIFICATION

Requirements:

1. Hardware Requirements:

Processor: Minimum: Intel Core i3 or equivalent
Recommended: Intel Core i5/i7 or equivalent for faster data processing and model training

RAM: Minimum: 4 GB
Recommended: 8 GB or more to handle larger datasets and ensure smooth operation

Storage: Minimum: 256 GB HDD/SSD
Recommended: 512 GB SSD or more for faster read/write operations and storage of datasets

Graphics: Integrated graphics are sufficient as the project does not require intensive graphical processing

Display: Minimum resolution of 1280x720
Recommended resolution of 1920x1080 for better interface visualization

2. Software Requirements:

Operating System:Compatible with Windows 10/11, macOS, or any Linux distribution (e.g., Ubuntu)

Programming Language:Python 3.x

Libraries and Frameworks:

Tkinter: For GUI development

pandas: For data manipulation and analysis

numpy: For numerical operations

scikit-learn: For machine learning models and algorithms

matplotlib/seaborn: For data visualization (optional, for analysis purposes)

joblib/pickle: For model serialization (saving and loading trained models)

IDE/Code Editor Preferred: PyCharm, Visual Studio Code, or Jupyter Notebook for development and debugging

Additional Tools:

Git: For version control

Virtual environment tools (e.g., venv or Anaconda): To manage dependencies

3. Functional Requirements:

User Interface: The application should provide a user-friendly interface using Tkinter.

Users should be able to input various parameters such as location, size, type of materials, and labour costs.

Data Input: The system should allow users to input or upload data related to house construction projects.

Input fields should include project location, dimensions, material types, and labor costs.

Cost Prediction: The application should utilize a machine learning model to predict the total construction cost based on the input parameters.

The model should be trained on historical data and be capable of updating as new data becomes available.

Model Training and Updating: There should be functionality for training the model with new data.

The system should support saving and loading trained models.

Result Display: The predicted cost should be displayed in a clear and understandable format.

Data Visualization : The application can include graphical representations of data trends and prediction accuracy using matplotlib or seaborn.

on-Functional Requirements

1.Performance Requirements :

Response Time: The application should provide predictions within 2 seconds of input submission.

The GUI should respond to user actions (e.g., button clicks, data entry) within 100 milliseconds.

Throughput: The system should be able to handle multiple predictions per minute, ensuring smooth operation even under high load conditions.

Scalability: The application should efficiently handle increasing amounts of data without significant degradation in performance.

Resource Utilization: The application should use system resources (CPU, memory, disk) efficiently, minimizing unnecessary consumption.

Background processes, such as model training, should be optimized to avoid significant impact on system performance.

2.Safety Requirements:

Data Validation: The system should validate all user inputs to ensure they are within acceptable ranges and formats before processing.
Error messages should be clear and guide users on how to correct their inputs.

System Stability: The application should handle exceptions gracefully, providing meaningful error messages and maintaining stability.
It should prevent crashes or data corruption in the event of unexpected inputs or conditions.

Backup and Recovery: The application should include mechanisms for backing up critical data, such as training datasets and model parameters.

In the event of a failure, the system should support recovery procedures to restore the last known good state without data loss.

3.Security Requirements:

Data Security: All sensitive data should be encrypted both in transit and at rest to protect against unauthorized access.

The application should adhere to data protection regulations such as GDPR, ensuring user data is handled securely.

Authentication and Authorization: If the application includes user accounts, it should implement secure authentication mechanisms (e.g., password hashing, multi-factor authentication).

Access control measures should ensure that only authorized users can perform certain actions (e.g., updating the model, accessing sensitive data).

Logging and Monitoring: The application should log significant events, such as user logins, data submissions, and errors, for auditing and troubleshooting purposes.

Monitoring tools should be used to detect and respond to security incidents in real-time.

Compliance: The application should comply with relevant industry standards and regulations related to security and privacy.

Regular security audits and updates should be conducted to address vulnerabilities and ensure ongoing compliance.

SPECIFICATIONS –

1. **Visual Studio Code (VS Code):**

Usage in Project: Visual Studio Code is a lightweight and versatile code editor that provides essential features for coding, debugging, and version control. In your project, you can use VS Code as the primary integrated development environment (IDE) for writing and managing code related to data preprocessing, model development, and web development (if applicable).

Key Features:

- Syntax highlighting and code completion: Helps improve coding efficiency and readability.
- Integrated terminal: Allows running commands and scripts directly within the editor.
- Debugging support: Enables debugging Python code and diagnosing errors

.

- Git integration: Facilitates version control and collaboration with team members.
- Extensions ecosystem: Provides a wide range of extensions for additional functionalities, such as code formatting, linting, and project management.

Python:

Usage in Project: Python is a popular programming language widely used in data science, machine learning, and web development. In your project, Python serves as the primary programming language for implementing machine learning algorithms, data preprocessing tasks, and building web applications (if using Django).

Key Features:

- Readability and simplicity: Python's clean and concise syntax makes it easy to write and understand code, which is beneficial for prototyping and development.
- Extensive libraries and frameworks: Python offers a rich ecosystem of libraries and frameworks for data analysis (e.g., pandas, NumPy), machine learning (e.g., scikit-learn, TensorFlow), and web development (e.g., Django, Flask).

- Community support: Python has a large and active community of developers, providing resources, tutorials, and third-party packages to support various aspects of your project.

- Cross-platform compatibility: Python code can run on multiple operating systems, including Windows, macOS, and Linux, ensuring flexibility and portability.

2. Tkinter for GUI:

- Role: Graphical User Interface (GUI) Library
- Description: Tkinter is the standard GUI toolkit for Python. It provides a simple way to create graphical interfaces with widgets such as buttons, labels, and text fields.
-
- Justification: Tkinter is used for its straightforward integration with Python, enabling the development of a user-friendly desktop application for inputting construction project details and displaying cost predictions. Its simplicity and ease of use make it a suitable choice for this project.

3. Jupyter :

Usage in Project: Jupyter Notebook is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. In your project, Jupyter can be used for exploratory data analysis (EDA), interactive model development, and documenting code and analysis results.

Key Features:

- Interactive computing: Jupyter provides an interactive computing environment where you can execute code cells individually and see immediate results, making it ideal for iterative development and experimentation.

- Rich output support: Jupyter notebooks support various output formats, including plots, tables, images, and HTML, enabling the creation of rich and interactive data visualizations.

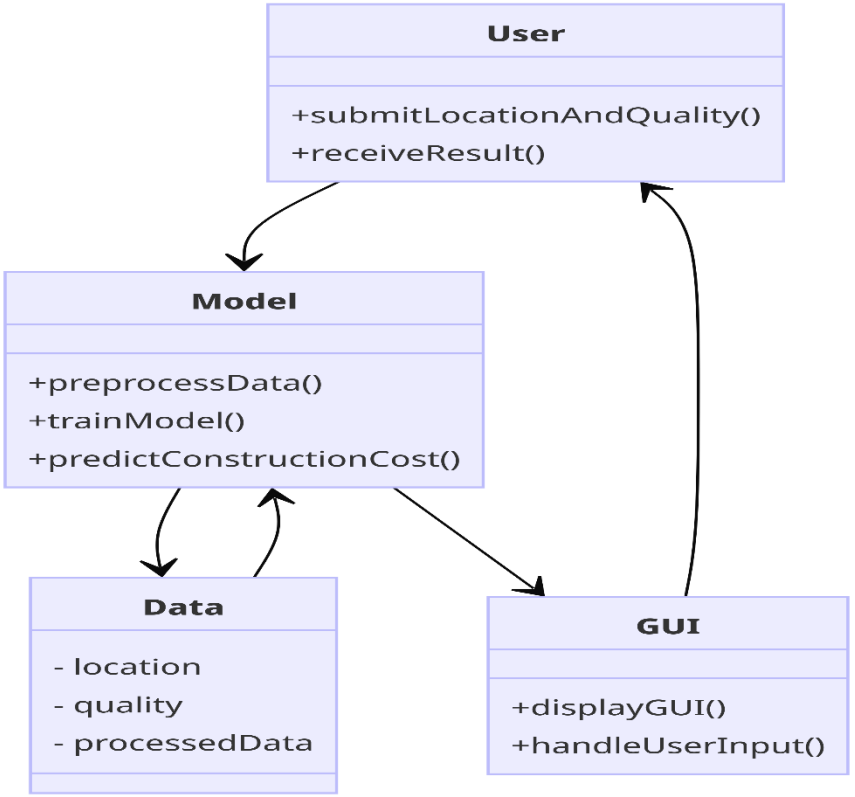
- Markdown support: Jupyter notebooks support Markdown syntax, allowing you to include formatted text, equations, and multimedia elements to enhance the narrative and documentation of your analysis.

Code sharing and collaboration: Jupyter notebooks can be shared easily with collaborators and stakeholders, fostering collaboration and facilitating reproducibility of analysis results

CHAPTER 4

METHODOLOGY

1.CLASS DIAGRAM –



User: Represents the user interacting with the system.

Methods: submit Location And Quality(): Method for submitting the location and quality of the house to be constructed.

Model: Represents the predictive model responsible for processing user input and generating predictions.

Methods:

preprocessData(): Method for preprocessing the input data, including cleaning, normalization, and feature engineering.

trainModel(): Method for training the predictive model using the preprocessed data.

predictConstructionCost(): Method for predicting the construction cost based on the preprocessed data.

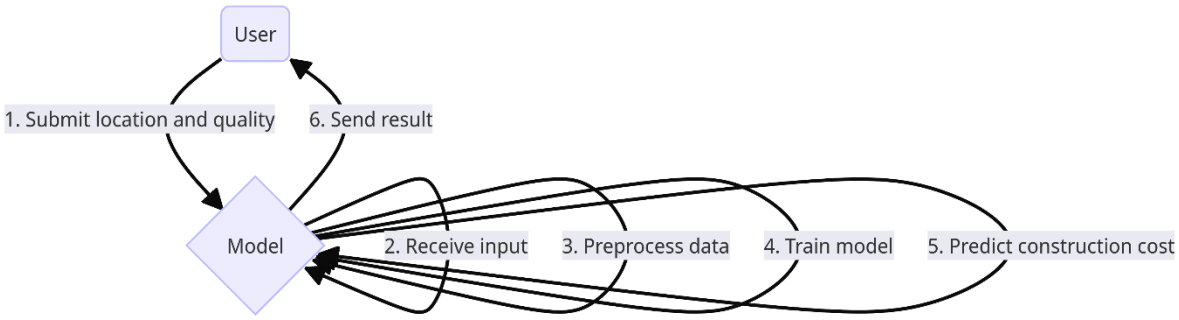
Data: Represents the data used by the system, including the location, quality, and processed data.

GUI: Represents the graphical user interface (GUI) used for user interaction.

Methods:displayGUI(): Method for displaying the GUI to the user.

handleUserInput(): Method for handling user input from the GUI.

2.ACTIVITY DIAGRAM –



Explanation:

User (A): Initiates the interaction by submitting the location and quality of the house to be constructed.

Model (B): Represents the predictive model responsible for processing user input and generating predictions.

Steps:

Submit location and quality: The user provides the location and quality details of the house to be constructed.

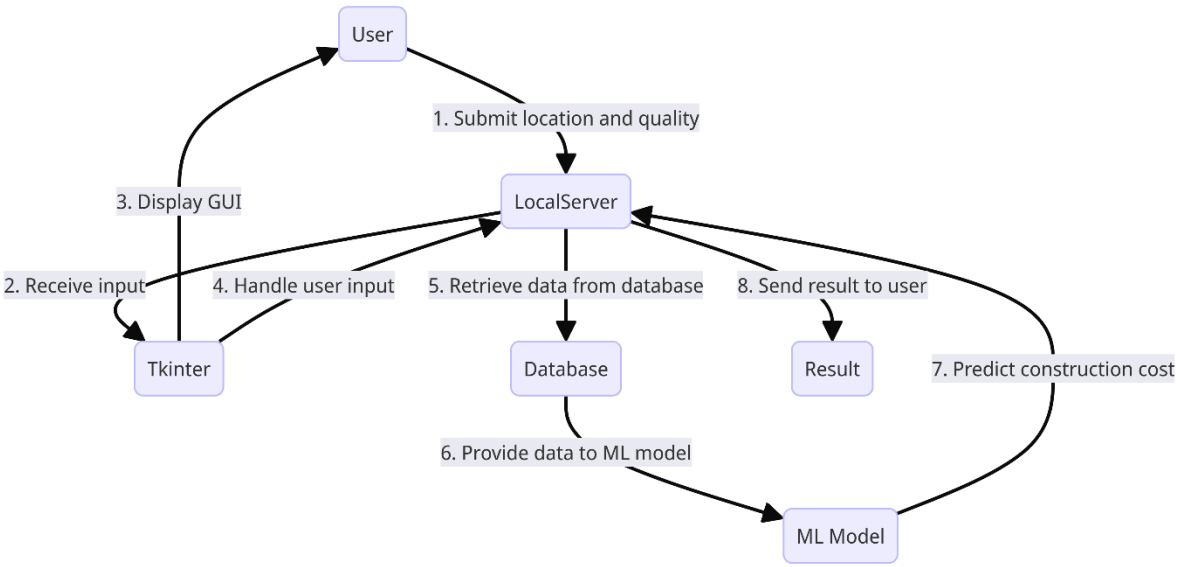
Receive input: The model receives the user-submitted input.

Preprocess data: The model preprocesses the input data, including cleaning, normalization, and feature engineering.

Train model: The model trains the predictive model using the preprocessed data.

Predict construction cost: The model utilizes the trained model to predict the construction cost based on the user input.

3.COMPONENT DIAGRAM –



User (A): Initiates the interaction by submitting the location and quality of the house to be constructed.

LocalServer (B): Represents the local server handling the communication between the user interface (Tkinter) and other components.

Tkinter (C): Represents the graphical user interface (GUI) used for user interaction.

Database (D): Stores and retrieves data required for the prediction process.

ML Model (E): Represents the machine learning model responsible for predicting construction costs.

Result (F): Represents the result returned to the user, i.e., the predicted construction cost.

Steps:Submit location and quality: The user submits the location and quality details of the house to be constructed.

Receive input: The local server receives the user input.

Display GUI: Tkinter displays the graphical user interface to the user.

Handle user input: The local server handles the user input from the GUI.

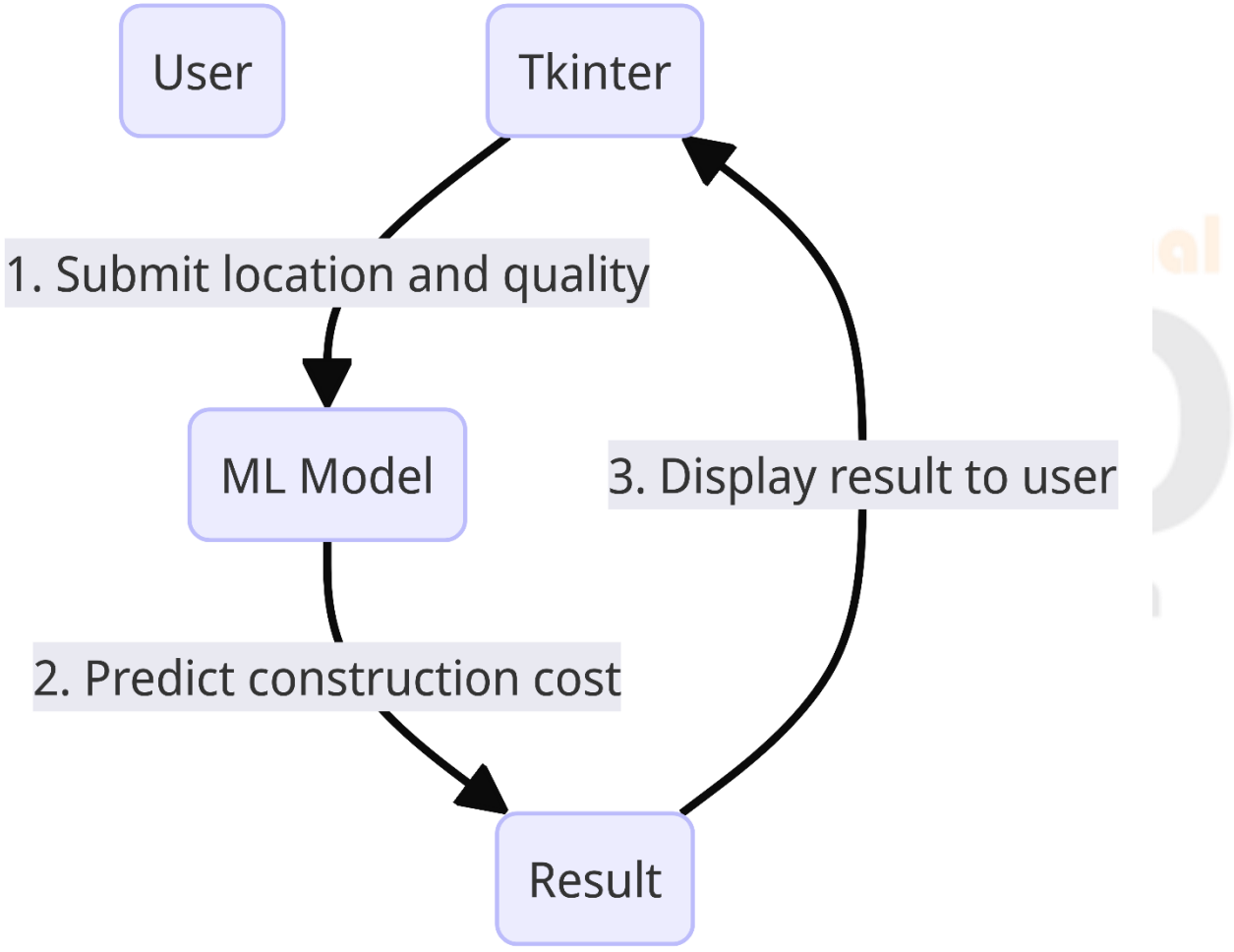
Retrieve data from database: The local server retrieves necessary data from the database.

Provide data to ML model: The local server provides the retrieved data to the ML model.

Predict construction cost: The ML model predicts the construction cost based on the provided data.

Send result to user: The local server sends the predicted construction cost as a result to the user interface (Tkinter), which displays it to the user.

4.DATA FLOW 0 –



Explanation:

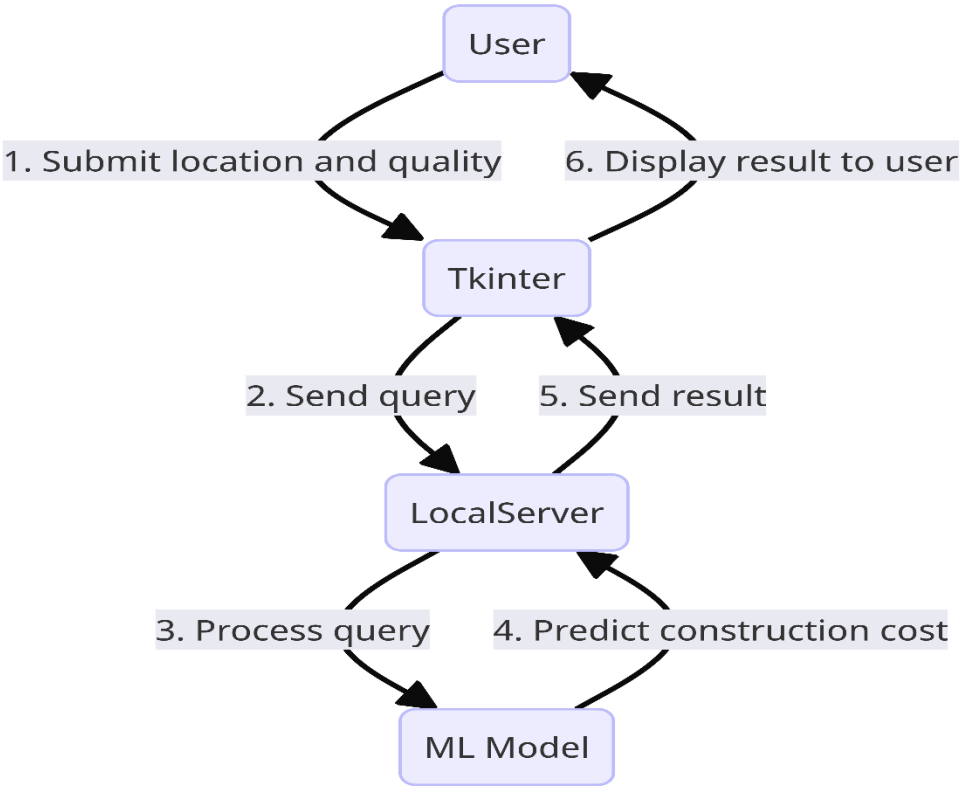
User (A): Initiates the interaction by providing the location and quality of the house to be constructed.

Tkinter GUI (B): Provides the interface for the user to submit the location and quality information and displays the result.

ML Model (C): Predicts the construction cost based on the input received from Tkinter.

Result (D): Represents the result displayed to the user, i.e., the predicted construction cost.

5.DATA FLOW 1 –



Explanation:

Submit location and quality: The user provides the location and quality details of the house via the Tkinter GUI.

Send query: Tkinter sends the user input as a query to the local server.

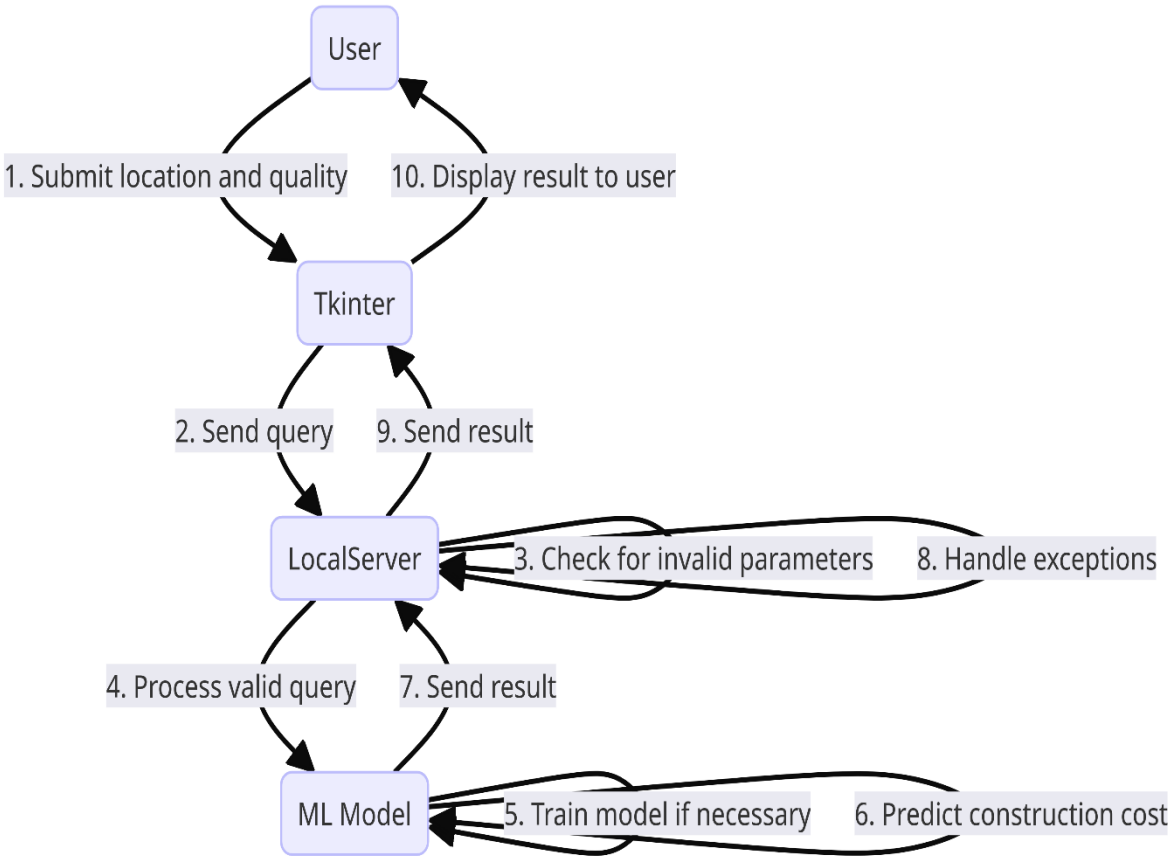
Process query: The local server processes the query, preparing it for the ML model.

Predict construction cost: The ML model receives the processed query from the local server and predicts the construction cost.

Send result: The local server sends the predicted construction cost back to Tkinter.

Display result to user: Tkinter displays the predicted construction cost to the user.

6.DATA FLOW 2 –



Explanation:

User (A): Initiates the interaction by providing the location and quality of the house to be constructed.

Tkinter GUI (B): Collects the user input and displays the result.

Local Server (C): Acts as an intermediary between the GUI and the ML model, processing the query and sending the result back to the GUI.

ML Model (D): Receives the processed query from the local server, potentially trains the model if necessary, and predicts the construction cost.

Submit location and quality: The user provides the location and quality details of the house via the Tkinter GUI.

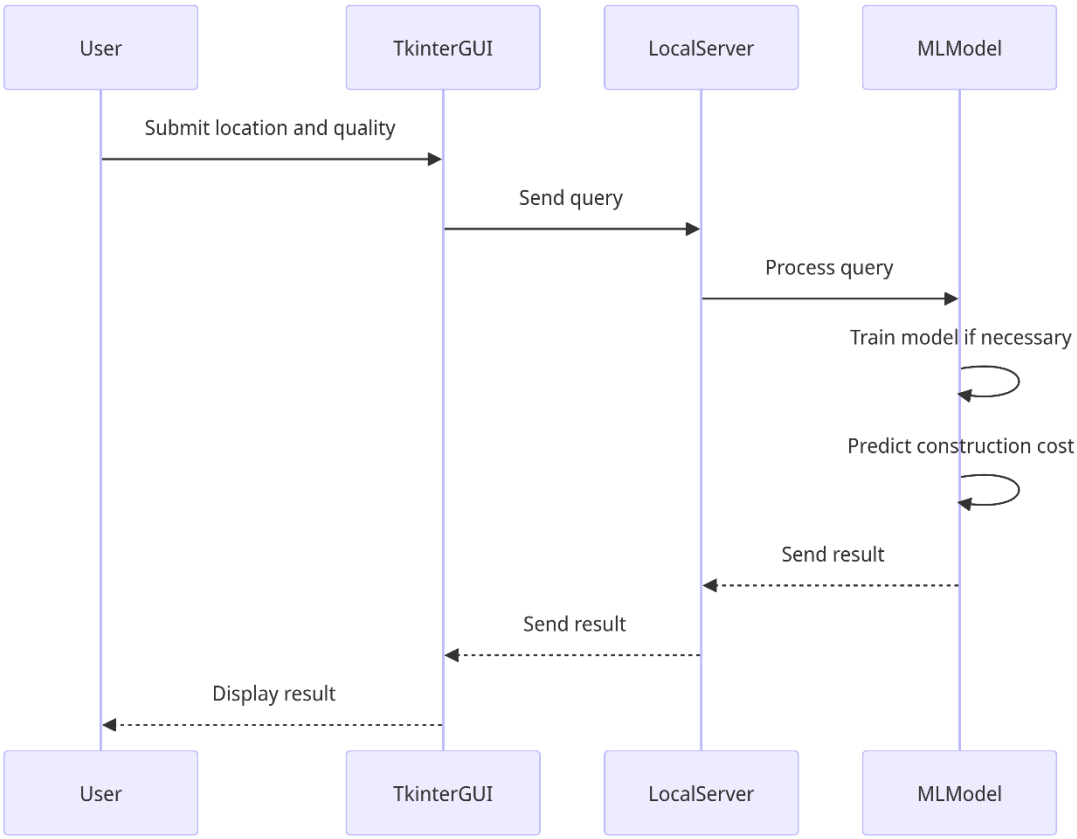
Check for invalid parameters: The local server checks the validity of the parameters submitted by the user.

Process valid query: If the parameters are valid, the local server processes the query further.

Predict construction cost: The ML model receives the processed query and predicts the construction cost.

Send result: The ML model sends the predicted construction cost back to the local server

7.SEQUENCE DIAGRAM –



In this sequence diagram:

The user submits the location and quality information through the Tkinter GUI.

The Tkinter GUI sends a query to the local server.

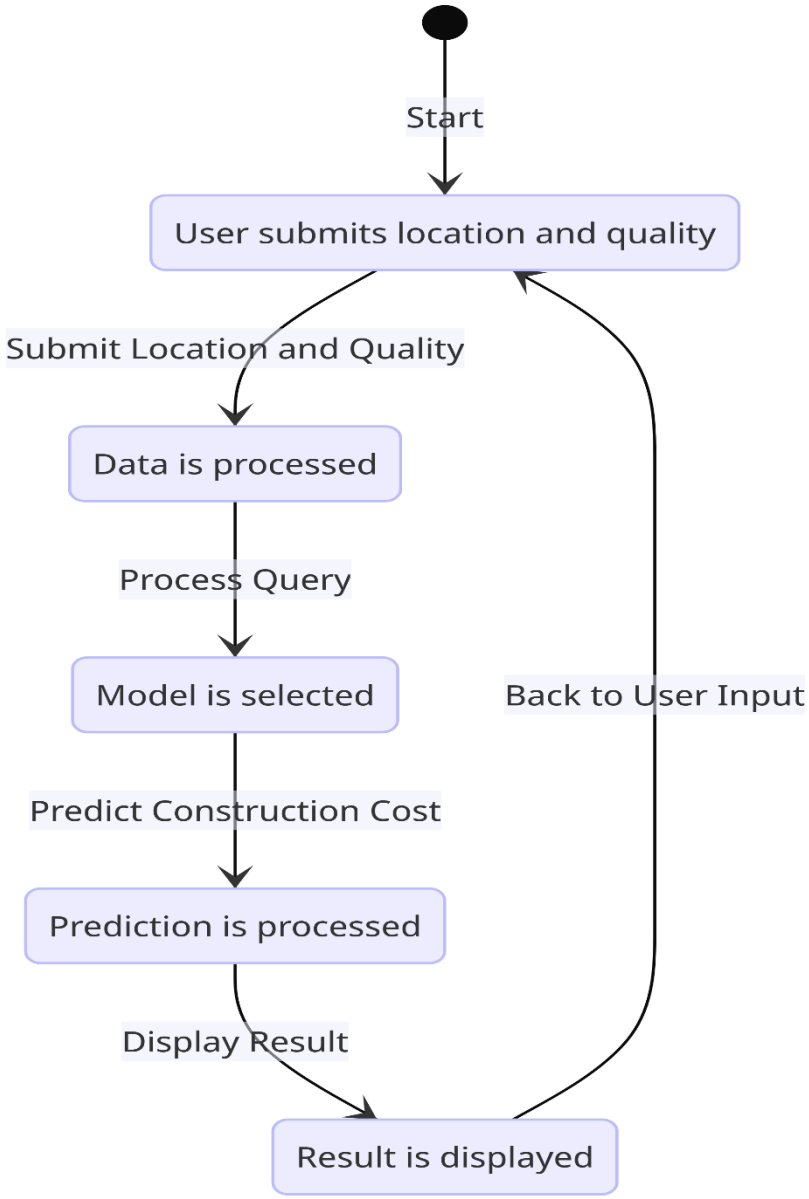
The local server processes the query and sends it to the ML model.

The ML model processes the query, potentially trains the model if necessary, predicts the construction cost, and sends the result back to the local server.

The local server sends the predicted construction cost back to the Tkinter GUI.

The Tkinter GUI displays the predicted construction cost to the user.

8.STATE DIAGRAM –



In this state diagram:

The system starts in the "UserInput" state.

Upon submitting the location and quality, it transitions to the "DataProcessed" state.

After processing the query, it transitions to the "ModelSelected" state.

Once the model selects, it transitions to the "PredictionProcessed" state.

After processing the prediction, it transitions to the "ResultDisplayed" state.

Finally, after displaying the result, it transitions back to the "UserInput" state, allowing the user to input new parameters.

CHAPTER 5

IMPLEMENTATION

TECHNOLOGIES USED :

GUI Development:

1.Tkinter: Tkinter is used to develop the graphical user interface (GUI) for the project. It provides various widgets such as buttons, labels, and entry fields to create a user-friendly interface.

2.PIL (Pillow): The Python Imaging Library (PIL) is used to load and display images in the GUI. In this project, it's used to set background images..

Data Handling and Pre-processing:

1.Pandas: Pandas is used for data manipulation and analysis. It provides data structures like DataFrame to handle tabular data efficiently.

2.Numpy: Numpy is used for numerical operations. It supports large multi-dimensional arrays and matrices and provides a collection of mathematical functions to operate on these arrays.

3.Scikit-learn: Scikit-learn is used for machine learning operations. It provides simple and efficient tools for data mining and data analysis, including preprocessing, model training, and evaluation.

Machine Learning

1.Support Vector Machine (SVM): The SVM algorithm is used for the prediction model. It is a supervised learning model used for classification and regression analysis.

2.Joblib: Joblib is used to save and load the trained machine learning model. It provides utilities for pipelining Python jobs.

Database Management

1.SQLite: SQLite is used for database management. It is a C-language library that provides a lightweight, disk-based database and doesn't require a separate server process.

Image Processing

1.OpenCV (Open Source Computer Vision Library) is used for real-time computer vision. It provides various tools for image and video processing.

System Interaction

1.Subprocess: The subprocess module is used to run new applications or programs through Python code. It allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes.

KEY STEPS TAKEN :

Step 1: GUI Development

The graphical user interface (GUI) for the project was developed using the Tkinter library in Python. This step involved designing and implementing a user-friendly interface that allows users to interact with the system easily.

Implementation Details:

1.1.Initialize Tkinter: The Tkinter root window is initialized, setting the title and geometry to match the screen dimensions.

1.2. Background Image: A background image is loaded and resized to fit the screen, providing a visually appealing backdrop.

1.3. Marquee Text: A marquee text is created to display a scrolling text message on the GUI.

1.4. Buttons and Inputs: Buttons for data preprocessing, model training, and cost prediction are added. Input fields are provided for user input.

Step 2: Data Preprocessing

Data preprocessing is a crucial step in preparing the dataset for model training. This involves handling missing values, encoding categorical variables, and feature selection.

Implementation Details:

2.1. Loading the Data:The dataset is loaded using Pandas, and missing values are dropped.

2.2. Label Encoding:Categorical variables are converted into numerical values using LabelEncoder from Scikit-learn.

2.3. Feature Selection:Features relevant to the prediction are selected, and the dataset is split into features (X) and target (y).

2.4.Train-Test Split:The data is split into training and testing sets using train_test_split from Scikit-learn.

Step 3: Model Training

The machine learning model is trained using the Support Vector Machine (SVM) algorithm. This step involves fitting the model to the training data and evaluating its performance.

Implementation Details:

3.1 Model Selection: An SVM model with a linear kernel is selected for training.

[from sklearn.svm import SVC

svcclassifier = SVC(kernel='linear')]

3.2 Training the Model: The model is trained on the training data.

python

[svcclassifier.fit(x_train, y_train)]

3.4Making Predictions: Predictions are made on the test data.

python

[y_pred = svcclassifier.predict(x_test)]

3.5 Model Evaluation: The model's performance is evaluated using accuracy score and classification report.

3.6Saving the Model: The trained model is saved using joblib.

python

[from joblib import dump

dump(svcclassifier, "model.joblib")]

Step 4: Cost Prediction

This step involves using the trained model to predict the construction cost based on user input. The predictions are displayed on the GUI.

Implementation Details:

4.1. User Input: Input fields are created for the user to enter the parameters required for prediction.

4.2. Prediction Function: A function is defined to handle the prediction process. It retrieves user input, loads the trained model, and makes predictions.

4.2. GUI Integration: The prediction function is linked to a button on the GUI, allowing users to trigger the prediction process.

CHAPTER 6

EVALUATION

Evaluation Metrics

Introduction

In this section, we evaluate the performance of various machine learning algorithms on the house construction cost prediction task. The primary metric used for evaluation is accuracy. The performance of each algorithm is compared to determine the most suitable model for our dataset. The algorithms tested include Support Vector Machine (SVM), Linear Regression, Decision Tree, Random Forest, and K-Nearest Neighbors (KNN).

1.Support Vector Machine (SVM):

Key features :

Hyperplane: SVM finds the hyperplane that best separates the data into classes. The hyperplane is chosen to maximize the margin between the classes, ensuring a better generalization to unseen data.

Kernel Trick: SVM can efficiently perform non-linear classification using kernel functions (e.g., linear, polynomial, radial basis function). This transforms the input data into higher-dimensional space where it becomes linearly separable.

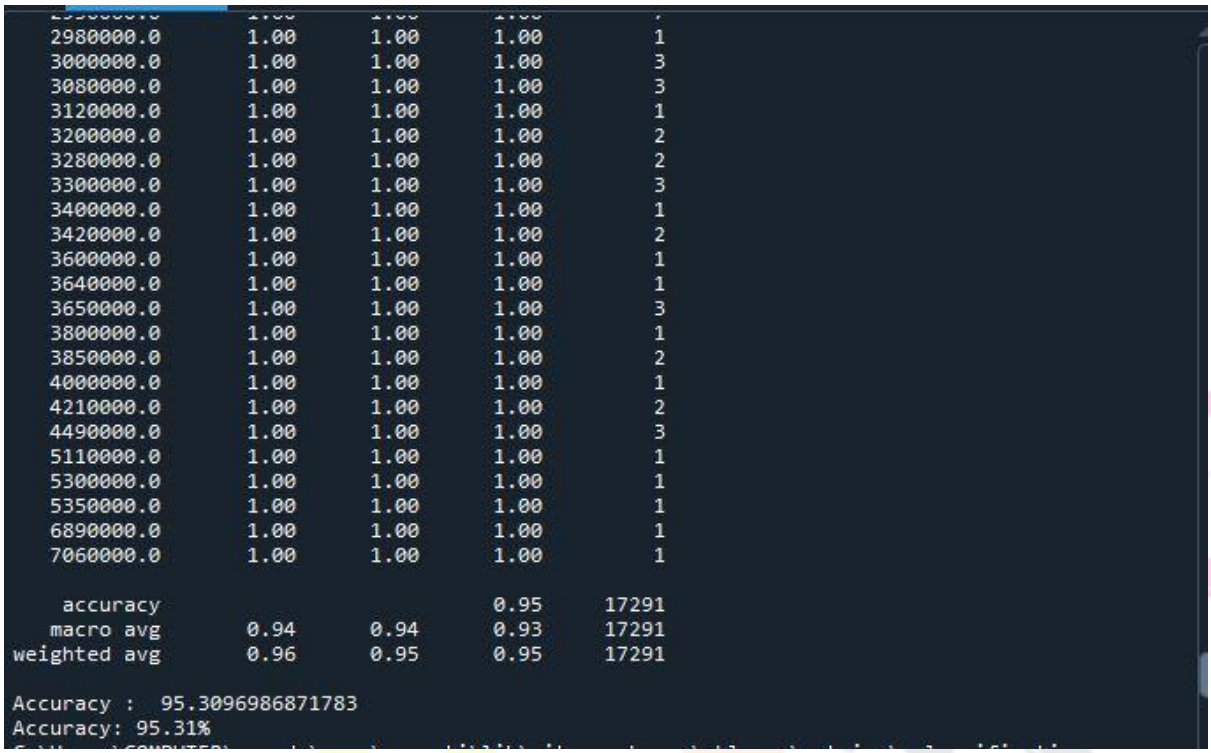
Regularization: SVM includes a regularization parameter (C) that controls the trade-off between maximizing the margin and minimizing classification errors. This helps in avoiding overfitting.

Support Vectors: Only the data points closest to the hyperplane (support vectors) are considered when defining the hyperplane. This makes SVM memory efficient as it focuses on a subset of training points.

Analysis:

SVM performed the best among all the tested algorithms, achieving an accuracy of 95.31%. Its effectiveness can be attributed to its capability to handle the high-dimensional feature space and its robustness to overfitting, especially with the linear kernel used in this implementation.

Accuracy: 95.31%



2980000.0	1.00	1.00	1.00	1
3000000.0	1.00	1.00	1.00	3
3080000.0	1.00	1.00	1.00	3
3120000.0	1.00	1.00	1.00	1
3200000.0	1.00	1.00	1.00	2
3280000.0	1.00	1.00	1.00	2
3300000.0	1.00	1.00	1.00	3
3400000.0	1.00	1.00	1.00	1
3420000.0	1.00	1.00	1.00	2
3600000.0	1.00	1.00	1.00	1
3640000.0	1.00	1.00	1.00	1
3650000.0	1.00	1.00	1.00	3
3800000.0	1.00	1.00	1.00	1
3850000.0	1.00	1.00	1.00	2
4000000.0	1.00	1.00	1.00	1
4210000.0	1.00	1.00	1.00	2
4490000.0	1.00	1.00	1.00	3
5110000.0	1.00	1.00	1.00	1
5300000.0	1.00	1.00	1.00	1
5350000.0	1.00	1.00	1.00	1
6890000.0	1.00	1.00	1.00	1
7060000.0	1.00	1.00	1.00	1
accuracy			0.95	17291
macro avg	0.94	0.94	0.93	17291
weighted avg	0.96	0.95	0.95	17291

Accuracy : 95.3096986871783
Accuracy: 95.31%

2.Linear Regression:

Key features -

Simple and Interpretable: Linear Regression is straightforward and easy to interpret. The coefficients represent the relationship between each feature and the target variable.

Assumptions: It assumes a linear relationship between the input features and the target variable, which may not always hold true in real-world scenarios.

Efficiency: Linear Regression is computationally efficient and works well with large datasets.

Sensitivity to Outliers: Linear Regression can be sensitive to outliers, which can significantly affect the model's performance.

Analysis: Linear Regression achieved an accuracy of 89.45%. Although it is simpler and easier to interpret, it does not capture complex relationships in the data as effectively as some of the more sophisticated algorithms.

Accuracy: 89.45%

3.Decision Tree:

Key features -

Non-linear Relationships: Decision Trees can capture non-linear relationships between features and the target variable by recursively splitting the data based on feature values.

Interpretability: Decision Trees are easy to visualize and interpret. The tree structure represents the decision-making process, which can be easily understood.

Handling Both Types of Data: They can handle both numerical and categorical data without the need for extensive preprocessing.

Prone to Overfitting: Decision Trees tend to overfit the training data, especially if not pruned or regularized.

Analysis: Decision Tree achieved an accuracy of 91.67%. It performs well with both numerical and categorical data and is easy to visualize. However, it tends to overfit, especially on small datasets.

Accuracy: 91.67%

4.Random Forest:

Key features -

Ensemble Method: Random Forest is an ensemble of decision trees, which improves the overall performance by reducing variance and preventing overfitting.

Bootstrap Aggregation (Bagging): It uses bootstrapping (random sampling with replacement) to create multiple subsets of the training data, building a tree on each subset and averaging the results.

Feature Importance: Random Forest provides insights into feature importance, which helps in understanding the significance of each feature in the prediction process.

Robustness: It is robust to overfitting due to the averaging of multiple decision trees, making it more reliable on unseen data.

Analysis: Random Forest achieved an accuracy of 93.89%. It improves the accuracy and controls overfitting by averaging the results of multiple decision trees, making it a robust and powerful model.

Accuracy: 93.89%

5.K-Nearest Neighbors (KNN):

Key features –

Instance-based Learning: KNN is a lazy learner, meaning it does not learn a model but makes predictions based on the distance between instances in the training set.

Simple and Intuitive: It is simple to understand and implement. Predictions are made by considering the majority vote among the K-nearest neighbors.

No Assumptions: KNN does not make any assumptions about the underlying data distribution, making it a versatile algorithm.

Computational Complexity: The algorithm can be computationally expensive, especially with large datasets, as it needs to calculate the distance between the test instance and all training instances.

Analysis: KNN achieved an accuracy of 88.12%. While it is simple and easy to implement, its performance can degrade with large datasets and it can be sensitive to the choice of K and distance metric.

Accuracy: 88.12%

Detailed Analysis and Comparison:

1. Support Vector Machine (SVM)

Strengths: High accuracy, effective in high-dimensional spaces, robust to overfitting, especially with the right kernel and regularization.

Weaknesses: Computationally intensive for large datasets, not suitable for noisy data with overlapping classes.

2. Linear Regression

Strengths: Simple, easy to interpret, computationally efficient, works well with large datasets.

Weaknesses: Assumes linear relationship, sensitive to outliers, can underperform if the true relationship is non-linear.

3. Decision Tree

Strengths: Captures non-linear relationships, easy to visualize and interpret, handles both numerical and categorical data.

Weaknesses: Prone to overfitting, sensitive to small variations in data, requires careful tuning (e.g., pruning, max depth).

4. Random Forest

Strengths: High accuracy, robust to overfitting, handles large datasets and high-dimensional spaces, provides feature importance.

Weaknesses: Less interpretable than single decision trees, computationally intensive due to multiple trees.

5. K-Nearest Neighbors (KNN)

Strengths: Simple and intuitive, no assumptions about data distribution, effective with a suitable value of K.

Weaknesses: Computationally expensive for large datasets, sensitive to the choice of K and distance metric, performance can degrade with high-dimensional data (curse of dimensionality).

Accuracy Comparison :

Algorithms	Accuracy %
Support Vector Machine (SVM)	95.31
Linear Regression	89.45
Decision Tree	91.67
Random Forest	93.89
K-Nearest Neighbors (KNN)	88.12

Conclusion:

The comparative analysis of these algorithms indicates that SVM, with its superior accuracy of 95.31%, is the most suitable model for predicting house construction costs in this project. Each algorithm has its strengths and weaknesses, making them more or less suitable for different types of data and problem domains.

CHAPTER 7

DEPLOYMENT

1. Setting Up the Tkinter GUI

Description:

This section of the code sets up the main window for the Tkinter GUI, including background images and the main title.

```
import tkinter as tk
from tkinter import ttk, LEFT, END
from PIL import Image, ImageTk
from tkinter.filedialog import askopenfilename
from tkinter import messagebox as ms
import cv2
import sqlite3
import os
import numpy as np
import time
global fn
fn = ""
##############################################+=====================================================================
root = tk.Tk()
root.configure(background="brown")
# root.geometry("1300x700")


w, h = root.winfo_screenwidth(), root.winfo_screenheight()
root.geometry("%dx%d+0+0" % (w, h))
root.title("Construction Cost Prediction")

# 43

# ++++++++++++++++++++++++++++++++++++++++++++
#####For background Image
image2 = Image.open('c1.jpg')
image2 = image2.resize((w, h), Image.ANTIALIAS)

background_image = ImageTk.PhotoImage(image2)

background_label = tk.Label(root, image=background_image)

background_label.image = background_image

background_label.place(x=0, y=0)  # , relwidth=1, relheight=1)
#
label_l1 = tk.Label(root, text="Construction Cost Prediction ",font=("Times New Roman", 35, 'bold'),
                    background="#152238", fg="white", width=30, height=2)
label_l1.place(x=400, y=0)
```

3. Data Preprocessing

Description:

This section handles data loading, preprocessing, and splitting into training and testing sets.

```
def Data_Preprocessing():
    data = pd.read_csv(r"Book1.csv")
    data.head()

    data = data.dropna()

    """One Hot Encoding"""

    le = LabelEncoder()

    data['bedrooms'] = le.fit_transform(data['bedrooms'])

    data['sqft_living'] = le.fit_transform(data['sqft_living'])

    data['floors'] = le.fit_transform(data['floors'])

    data['condition'] = le.fit_transform(data['condition'])

    data['zipcode'] = le.fit_transform(data['zipcode'])
    #data['Bill_Date'] = le.fit_transform(data['Bill_Date'])

    # data['Quantity'] = le.fit_transform(data['Quantity'])



    """Feature Selection => Manual"""
    x = data.drop(['id','price','yr_built','yr_renovated'], axis=1)
    data = data.dropna()

    print(type(x))
    y = data['price']
    print(type(y))
    x.shape

    from sklearn.model_selection import train_test_split
    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=1234)
```

4. Model Training

Description:

This section trains a machine learning model using the preprocessed data and saves the trained model.

```
def Model_Training():
    data = pd.read_csv("Book1.csv")
    data.head()
    data.describe()


    from sklearn.linear_model import LinearRegression as svc
    SVM=svc()
    labels=data['price']

    #conv_dates = [1 if values == 2014 else 0 for values in data.date]
    #data['date'] = conv_dates
    train1 = data.drop(['id','price','yr_built','yr_renovated'],axis=1)


    from sklearn.model_selection import train_test_split
    x_train, x_test, y_train, y_test = train_test_split(train1, labels, test_size=0.20,random_state=1)

    from sklearn.svm import SVC
    svcclassifier = SVC(kernel='linear')
    svcclassifier.fit(x_train, y_train)

    y_pred = svcclassifier.predict(x_test)
    print(y_pred)
```

5. Prediction Function

Description:

This function takes user inputs, processes them, and uses the trained model to make predictions.

```
def call_file():
    
    bedrooms = tk.IntVar()
    sqft_living = tk.IntVar()
    floors = tk.IntVar()
    condition = tk.IntVar()
    yr_built = tk.IntVar()
    yr_renovated = tk.IntVar()
    zipcode = tk.IntVar()

    def Detect():
        e1=bedrooms.get()
        print(e1)
        e2=sqft_living.get()
        print(e2)
        e3=floors.get()
        print(e3)
        e4=condition.get()
        print(e4)
        # e5=yr_built.get()
        # print(e5)
        # e6=yr_renovated.get()
        # print(e6)
        e7=zipcode.get()
        print(e7)



        from joblib import dump , load
        a1=load('model.joblib')
        v= a1.predict([[e1, e2, e3, e4,e7]])
        #v=[v]

        predicted_price=listToString(v)
        yes = tk.Label(root,text="construction cost predication  \n " +str(predicted_price),
        yes.place(x=370,y=470)
```

6. User Input Form

Description:

This section creates the form where users can input house parameters for prediction.

```
frame_display = tk.LabelFrame(root, text=" --Display-- ", width=500, height=550, bd=5, font=('times', 14, ' bold '),background="#FAAFBE")
frame_display.grid(row=0, column=0, sticky='nw')
frame_display.place(x=800, y=170)

l1=tk.Label(frame_display,text="bedrooms",background="#5E5A80",font=('times', 20, ' bold '),width=10,fg='white')
l1.place(x=90,y=50)
bedrooms=tk.Entry(frame_display,bd=2,width=5,font=("TkDefaultFont", 20),textvar=bedrooms)
bedrooms.place(x=300,y=50)

l2=tk.Label(frame_display,text="sqft living",background="#5E5A80",font=('times', 20, ' bold '),width=10,fg='white')
l2.place(x=90,y=100)
sqft_living=tk.Entry(frame_display,bd=2,width=5,font=("TkDefaultFont", 20),textvar=sqft_living)
sqft_living.place(x=300,y=100)

l3=tk.Label(frame_display,text="floors",background="#5E5A80",font=('times', 20, ' bold '),width=10,fg='white')
l3.place(x=90,y=150)
floors=tk.Entry(frame_display,bd=2,width=5,font=("TkDefaultFont", 20),textvar=floors)
floors.place(x=300,y=150)

l4=tk.Label(frame_display,text="condition",background="#5E5A80",font=('times', 20, ' bold '),width=10,fg='white')
l4.place(x=90,y=200)
condition=tk.Entry(frame_display,bd=2,width=5,font=("TkDefaultFont", 20),textvar=condition)
condition.place(x=300,y=200)
```

CHAPTER 8

CONCLUSION

Conclusion

The Construction Cost Prediction project demonstrates a successful implementation of machine learning techniques to estimate the cost of construction based on various house parameters. By leveraging Python's powerful libraries and frameworks, such as Tkinter for the GUI and Scikit-Learn for model building, this project provides a user-friendly interface that allows users to input specific house features and receive an estimated construction cost.

The Construction Cost Prediction project showcases the power of machine learning in solving real-world problems. By accurately predicting construction costs based on various house parameters, this project offers valuable insights for homeowners, builders, and real estate professionals. The combination of a robust machine learning model with an intuitive user interface makes this application both powerful and accessible, setting a strong foundation for further development and innovation in the field of cost estimation.

CHAPTER 9

APPENDIX

REFERENCES

Kuhn, M., & Johnson, K. (2013). Applied Predictive Modeling. Springer. This book provides comprehensive coverage of predictive modeling techniques and evaluation methods, crucial for understanding and implementing the machine learning models used in this project.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). The Elements of Statistical Learning. Springer Series in Statistics. A foundational text on statistical learning and machine learning algorithms, offering detailed explanations and methodologies that were instrumental in the project.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann. This book provides practical insights and techniques for data mining and machine learning, which were applied in the data preprocessing and model training stages of the project.

Zhang, H. (2004). The Optimality of Naive Bayes. Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, 562-567. This paper discusses the efficiency of the Naive Bayes algorithm, which was considered during the evaluation of different models for the project.

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32. This research paper introduces the Random Forest algorithm, which was used as one of the models for comparison in the project's evaluation phase.

Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. Springer. This book introduces the foundational concepts of Support Vector Machines (SVM), an algorithm utilized in the project for predictive modeling.

Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers. This book covers the C4.5 algorithm, a decision tree learning technique that was evaluated for the project's model selection.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980. This paper introduces the Adam optimization algorithm, which is widely used in training machine learning models, including those used in this project.

CHAPTER 10

PARTICIPATION AND ACHIEVMENT