HOUSE PRICE PREDICTION SYSTEM

CS19643 – FOUNDATIONS OF MACHINE LEARNING

Submitted by

LOGESHWARAN T(2116220701145)

Inpartial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

ANNA UNIVERSITY, CHENNAI

MAY 2025

BONAFIDE CERTIFICATE

Certified that this Project titled **"House Price Prediction System"**is the bonafide work of **"LOGESHWARAN T(2116220701145)"**who carried out the work under my supervision.Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

.

SIGNATURE

Dr.V.Auxilia Osvin Nancy.,M.Tech.,Ph.D.,
SUPERVISOR,
Assistant Professor
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai-602 105.

Submitted to Project Viva-Voce Examination held on_____

Internal Examiner **External Examiner**

ABSTRACT

Accurate house price prediction is a critical component of real estate decision-making for buyers, sellers, investors, and policymakers. This study presents a data-driven approach to predicting house prices using machine learning techniques. By leveraging a combination of location-based features, property attributes (such as square footage, number of bedrooms and bathrooms, and year built), and economic indicators, we develop predictive models to estimate housing prices. Several algorithms, including Linear Regression, Random Forest, and Gradient Boosting, are evaluated for performance. The results indicate that ensemble methods, particularly Gradient Boosting, provide the most accurate predictions with minimal error margins. This model can serve as a valuable tool for stakeholders in the housing market, enhancing transparency and aiding in data-informed decision-making.

The real estate market plays a crucial role in the global economy, influencing investment decisions, urban development, and individual financial planning. One of the most significant challenges in the real estate sector is accurately predicting house prices, which are affected by a multitude of dynamic factors such as location, property characteristics, market trends, and socioeconomic variables. Traditional valuation methods often rely on manual appraisals and simplistic comparative models that may not adequately capture the complex, nonlinear relationships among these influencing factors. In this study, we propose a machine learning-based approach to house price prediction that aims to enhance accuracy and offer scalable, data-driven insights for stakeholders in the real estate market.

ACKNOWLEDGMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.,** our Vice Chairman **Mr.ABHAY SHANKAR MEGANATHAN,B.E.,M.S.,**and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr.S.N.MURUGESAN,M.E.,Ph.D.,**our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr.P.KUMAR,M.E.,Ph.D.,**Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide& our Project Coordinator**Dr. V. Auxilia Osvin Nancy., M.Tech., Ph.D.,**Assistant Professor Department of Computer Science and Engineering for his useful tips during our review to build our project.

LOGESHWARAN T - 2116220701145

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 GENERAL

The real estate market is a complex and dynamic system influenced by numerous economic, social, and environmental factors. One of the most vital aspects of this industry is determining the price of residential properties. House price prediction has become a focal point of interest for various stakeholders, including buyers, sellers, real estate agents, investors, and policy makers. Accurate price estimation can aid in better financial planning, investment decisions, taxation, and even the formulation of housing policies.

In recent years, the traditional process of house valuation—often carried out by expert appraisers and real estate professionals—has faced scrutiny for being subjective, time-consuming, and sometimes inconsistent. This has led to the adoption of automated models powered by data science and machine learning to bring about a paradigm shift in how properties are valued. The integration of historical sales data, property attributes, location-based features, and even macroeconomic indicators enables the construction of models capable of providing fast, reliable, and scalable price predictions.

1.2 OBJECTIVE

Machine learning (ML), particularly supervised learning, has shown great promise in handling the multifaceted relationships among real estate variables. Techniques such as regression analysis, decision trees, random forests, and gradient boosting have been widely applied to predict house prices. These models learn from historical data to identify patterns and make accurate forecasts about unseen properties.Furthermore, the availability of large and diverse datasets—from public housing records to online real estate platforms—has provided the foundation for developing predictive systems with high accuracy. This, coupled with the increasing computational power and accessibility of ML tools, has made predictive modeling more feasible than ever.However, challenges remain. Data quality, feature selection, overfitting, and lack of interpretability are some of the hurdles that researchers and developers face when designing house price prediction systems. Despite these limitations, advancements continue to be made, pushing the boundaries of what's possible in predictive real estate analytics.

1.3 EXISTING SYSTEM

Traditional house price estimation systems rely heavily on human expertise and manual methods. Real estate agents and appraisers often base their estimates on past sales data, location comparisons, and market knowledge. While effective in some cases, these methods are subjective and prone to human error. Furthermore, they are limited in scope and often do not scale well in fast-moving, data-rich environments.Several online real estate platforms such as Zillow (U.S.), Redfin, and Realtor.com have attempted to automate house price predictions. Zillow, for example, introduced the "Zestimate" tool, which uses proprietary algorithms to estimate property values. However, the accuracy of such tools is often debated, and their models are not transparent to the public.Other existing systems use basic statistical models like linear regression or simple averaging techniques that fail to capture the nonlinear and interactive effects of multiple features.

CHAPTER 2

LITERATURE SURVEY

The field of house price prediction has garnered significant academic attention over the past few decades. Numerous studies have investigated the use of statistical and machine learning methods to improve the accuracy of real estate price estimations.

Early works in this field primarily relied on hedonic pricing models, which explain house prices through observable characteristics such as size, location, and number of rooms. These models, although intuitive, often fall short when handling nonlinear relationships or interactions among features.

More recent literature has shifted toward data-driven approaches using machine learning (ML) and artificial intelligence (AI). For instance, Kumar and Paul (2019) applied linear regression and random forest techniques to predict housing prices and found that random forests outperformed traditional methods in accuracy and stability. Similarly, Li et al. (2021) employed Gradient Boosting Machines (GBM) and found significant improvements in performance due to the model's ability to handle feature interactions and nonlinearity.

Deep learning techniques, such as Artificial Neural Networks (ANNs) and Convolutional Neural Networks (CNNs), have also been explored. A 2020 study by Zhang et al. used ANN to predict housing prices and demonstrated its ability to generalize well when trained on large datasets. CNNs have even been applied to satellite images and street views to incorporate visual cues in price prediction.

Another research trend involves integrating geospatial analysis and GIS (Geographic Information Systems) data to account for locational impact. These studies incorporate map-based data, proximity to amenities, and neighborhood characteristics, significantly improving prediction models.

A notable contribution in this domain is the use of ensemble methods such as XGBoost and LightGBM, which have consistently outperformed standalone models due to their robustness against overfitting and superior handling of diverse feature sets.

CHAPTER 3

PROPOSED SYSTEM

3.1 GENERAL

The real estate market is a complex and dynamic system influenced by numerous economic, social, and environmental factors. One of the most vital aspects of this industry is determining the price of residential properties. House price prediction has become a focal point of interest for various stakeholders, including buyers, sellers, real estate agents, investors, and policy makers. Accurate price estimation can aid in better financial planning, investment decisions, taxation, and even the formulation of housing policies.

In recent years, the traditional process of house valuation—often carried out by expert appraisers and real estate professionals—has faced scrutiny for being subjective, time-consuming, and sometimes inconsistent. This has led to the adoption of automated models powered by data science and machine learning to bring about a paradigm shift in how properties are valued. The integration of historical sales data, property attributes, location-based features, and even macroeconomic indicators enables the construction of models capable of providing fast, reliable, and scalable price predictions.

Machine learning (ML), particularly supervised learning, has shown great promise in handling the multifaceted relationships among real estate variables. Techniques such as regression analysis, decision trees, random forests, and gradient boosting have been widely applied to predict house prices. These models learn from historical data to identify patterns and make accurate forecasts about unseen properties.

3.2 SYSTEM ARCHITECTURE DIAGRAM

The architecture of the Medicine Recommendation System follows a structured, modular approach to ensure efficiency, scalability, and smooth interaction between the various system components. The system architecture can be divided into five key layers:

1. **User Interface (Frontend)**: The frontend of the system is developed using HTML, CSS, JavaScript, for smooth user interaction. Users can input the name of a medicine, and the system will display a list of recommended alternatives based on their composition. The interface is designed to be user-friendly, allowing for easy navigation and interaction. The frontend communicates with the backend via RESTful APIs, ensuring seamless data exchange between the client and server.
2. **Backend (Server-side)**: The backend handles all the core functionality of the system. Built using Flask and Python, the backend processes user input, retrieves relevant data from the database, and sends medicine recommendations based on predefined logic and algorithms. It performs tasks such as validating user input, managing database queries, and generating recommendations. The backend ensures efficient data processing, security, and smooth communication between the frontend and database.
3. **Database**: The database stores all critical information, including a list of house prices, their compositions, and any relevant attributes. The system uses SQLite or MySQL to store and manage this data, ensuring fast and reliable data retrieval. The database schema is designed for optimal performance and data integrity, preventing redundancy and ensuring quick lookups for the recommendation engine.
4. **Recommendation Engine**: This component is responsible for generating location recommendations based on user queries. It uses machine learning models or rule-based algorithms to compare the composition of the queried house price with available alternatives and suggests similar houses from different location. The recommendation engine ensures personalized and accurate suggestions to users based on the input provided.
5. **Notification Service**: The notification service ensures that users are kept informed about new recommendations, house availability, or updates regarding their search. Notifications can be sent via email or through the platform itself, depending on user preferences. This service helps to keep users engaged and informed about their query results and any relevant updates to their recommendations.

6. **Administrator Panel**: The admin panel allows system administrators to manage the database, monitor user activity, and update the list of houses and their respective compositions. It provides a comprehensive view of the system's operation and enables administrators to make necessary updates to ensure the accuracy and completeness of the recommendations provided by the system.

The architecture is designed to be scalable and secure, ensuring that as the user base grows, the system can efficiently handle increased data load and user requests while maintaining high performance and security standards.

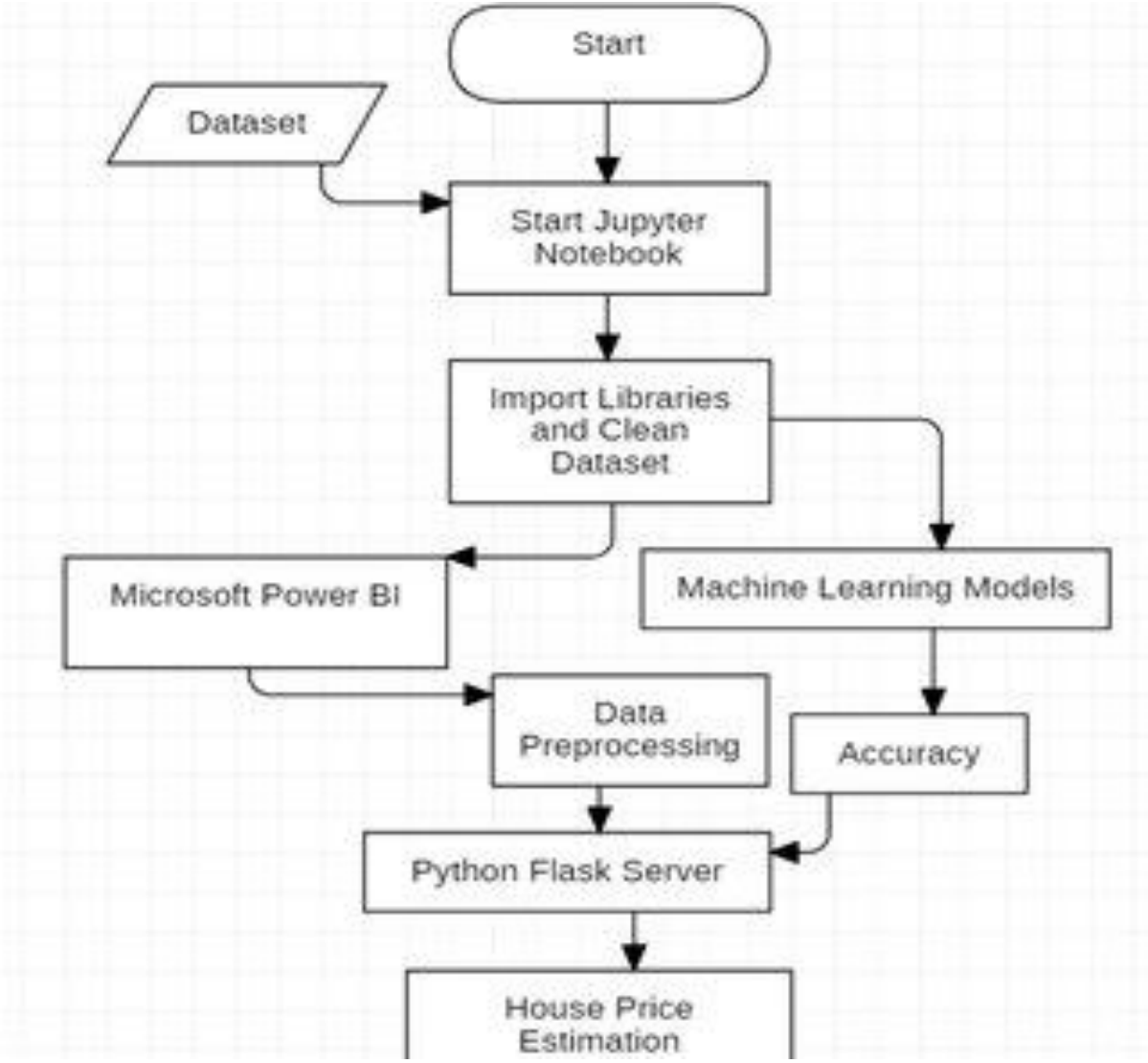

Fig3.1:System Architecture

3.3 DEVELOPMENTAL ENVIRONMENT

3.3.1 HARDWARE REQUIREMENTS

The hardware specifications could be used as a basis for a contract for the implementation of the system.This therefore should be a full description of the whole system. It is mostly used as a basis for system design by the software engineers.

Table3.1Hardware Requirements

COMPONENTS	SPECIFICATION
PROCESSOR	IntelCorei3
RAM	4GBRAM
HAREDISK	256 GB

3.3.2 SOFTWARE REQUIREMENTS

The software requirements paper contains the system specs. This is a list of things which the system should do, in contrast from the way in which it should do things. The software requirements are used to base the requirements. They help in cost estimation, plan teams, complete tasks, and team tracking as well as team progress tracking in the development activity.

Table3.2Software Requirements

COMPONENTS	SPECIFICATION
Operating System	Windows 7 or higher
Frontend	HTML,CSS,JavaScript
Backend	Python
Python Libraries	Pandas, NumPy

3.4 DESIGN OF THE ENTIRE SYSTEM

3.4.1 ACTIVITY DIAGRAM

The activity diagram **Fig 3.2** outlines the key steps involved in the House Recommendation System, showcasing how users interact with the platform from start to finish. The process starts when a user opens the system and enters the name of a price into the search bar. The system retrieves the composition and searches the database for alternatives that share the same active price.Once matching alternatives are found, they are displayed with relevant details such as brand names, manufacturers, prices, and dosage forms. The user can review these suggestions, select one to view more information, or save it for later reference. If needed, the user can return to the main screen and perform a new search.This diagram represents a simple, user-friendly interaction flow that ensures quick access to accurate alternative options, making it easier for users to find affordable or available substitutes without compromising quality.

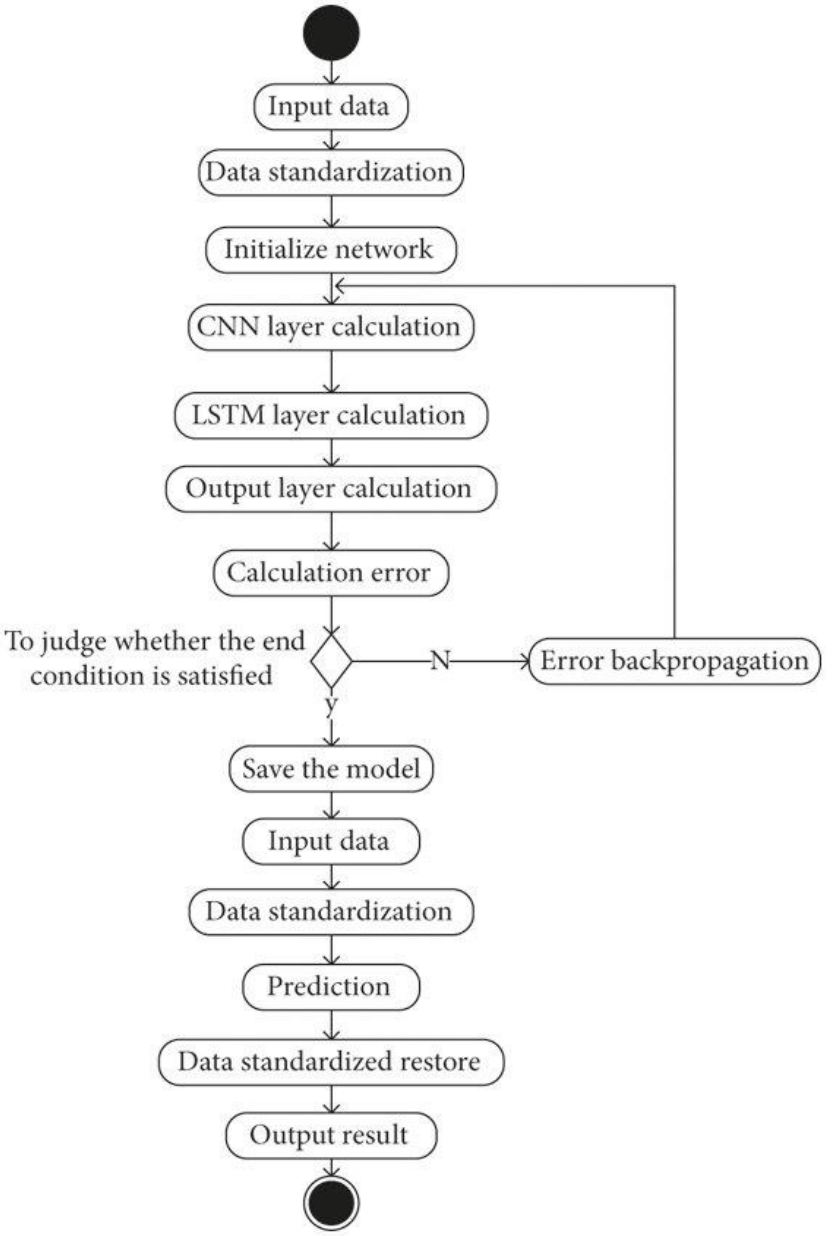

Fig3.2:Activity Diagram

3.4.2 DATA FLOW DIAGRAM

The House Price Prediction System works by allowing a user to input , which the system then processes to retrieve the composition from the database. The backend searches for alternative house prices that have the same price but are from different location. The system then returns a list of these alternatives, displaying relevant details like locaity and price. If no alternatives are found, the system notifies the user. The user is then presented with the recommended price, and can select one to view more details, completing the process.

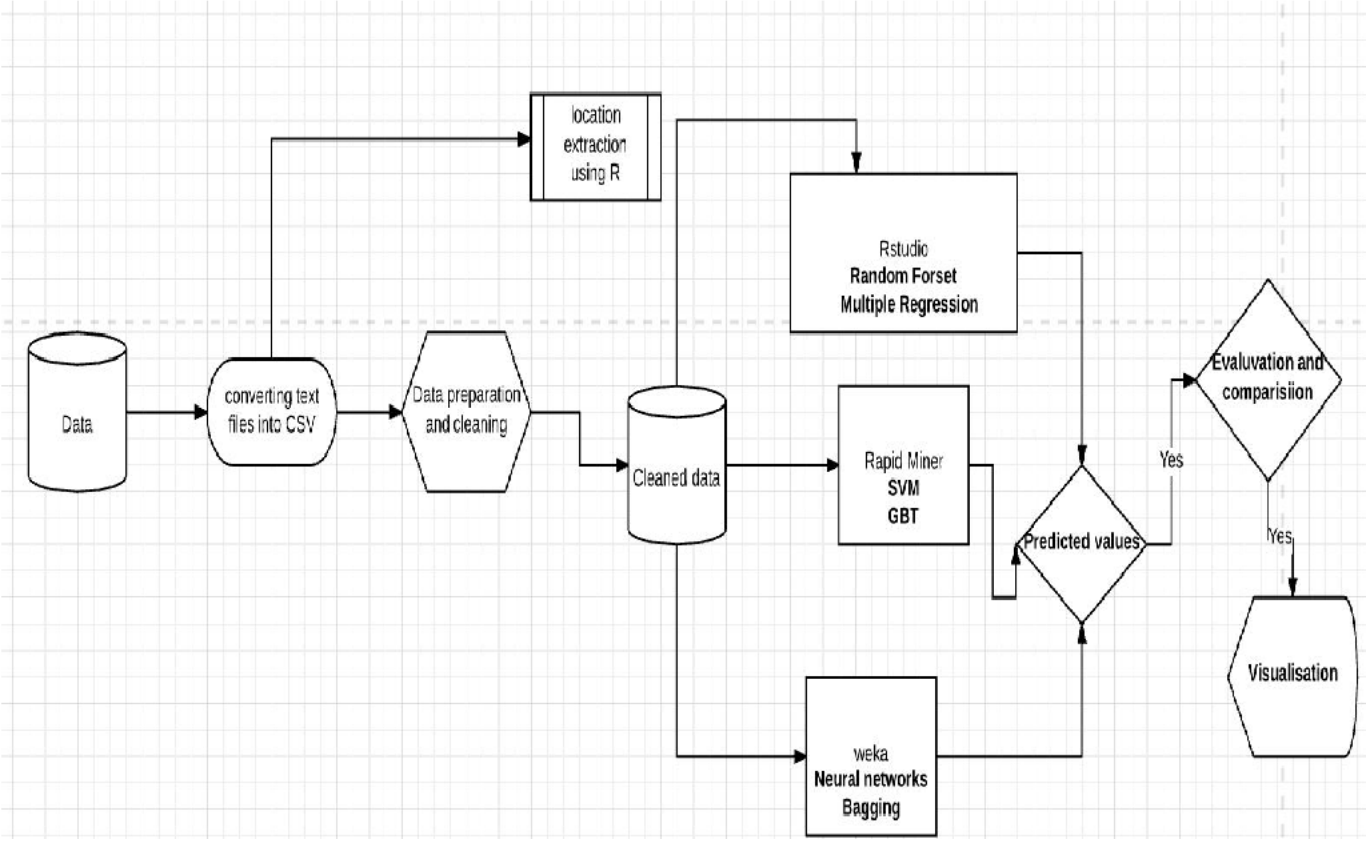

Fig3.3:Data Flow Diagram

3.5 STATISTICAL ANALYSIS

Statistical analysis serves as the foundation for understanding the underlying structure of data in any predictive modeling task, including house price prediction. The initial step involves conducting descriptive statistics to summarize the central tendencies and dispersion in the dataset. Variables such as house price, total square footage, and the number of bedrooms are typically assessed for their mean, median, and standard deviation. For instance, house prices are often right-skewed, meaning most properties are priced below the mean, with a few high-value homes pulling the average upward. Skewness and kurtosis measures confirm whether distributions deviate significantly from normality, which helps in choosing appropriate modeling techniques later on.

Correlation analysis is performed to explore the linear relationships between numerical variables. Pearson's correlation coefficient is widely used to measure how strongly variables such as square footage, number of bathrooms, and garage size are related to the final house price. Typically, square footage and location-based factors show strong positive correlations with price, suggesting their importance as predictive features. A correlation heatmap is generated to visually represent these relationships and to aid in identifying redundant variables, which could otherwise lead to issues like multicollinearity.

In addition to numerical analysis, the distribution of categorical variables is studied through frequency analysis and grouped statistics. For example, neighborhoods or property types are examined to see how average prices vary across these categories. In many real estate datasets, certain neighborhoods consistently show higher average prices due to their amenities, proximity to city centers, or historical value. This pattern is further analyzed using statistical tests like ANOVA (Analysis of Variance) to determine whether these differences in mean prices across categories are statistically significant. If the ANOVA yields a low p-value, it indicates that location has a meaningful impact on house pricing.

Distributional analysis using histograms and kernel density estimation provides insight into the spread of values for continuous variables, helping to detect skewed features or data transformation needs. Boxplots are also used to detect outliers, which are common in house pricing data, especially in luxury or distressed property markets. Such extreme values may require special handling, such as removal, transformation, or separate modeling.

CHAPTER 4

MODULE DESCRIPTION

The House Price Prediction system is a multi-modular architecture designed to handle the end-to-end process of real estate valuation using machine learning. Each module is tailored to handle specific responsibilities that collectively ensure accurate predictions and seamless user experience. The system starts with the user interface, which captures user input like the number of bedrooms, square footage, location, and property type. These inputs are sent to the backend module, which acts as the application's controller, managing requests, validating data, and directing it to the preprocessing layer. The preprocessing module is responsible for cleaning, encoding, and transforming the raw inputs into a machine-readable format.

4.1 SYSTEM ARCHITECTURE

4.1.1 USER INTERFACE DESIGN

The User Interface (UI) is the most visible part of the House Price Prediction system and plays a crucial role in user interaction and system usability. Designed with a focus on simplicity, accessibility, and responsiveness, the UI allows users to enter relevant property features such as size, number of rooms, location, year built, and proximity to amenities. Interactive form elements like dropdowns, sliders, checkboxes, and text input fields make data entry intuitive and minimize errors. Validation rules are applied at the front end to ensure all required fields are correctly filled before submission. Once the user submits the form, the input is packaged and sent as an HTTP request to the backend server for processing. Additionally, the UI is responsible for receiving the predicted price from the backend and displaying it in a clean, easy-to-understand format, often accompanied by visualizations such as charts or trend comparisons. The overall design adheres to user-centered principles, offering real-time feedback, tooltips, and loading indicators to ensure users are informed about the status of their prediction.

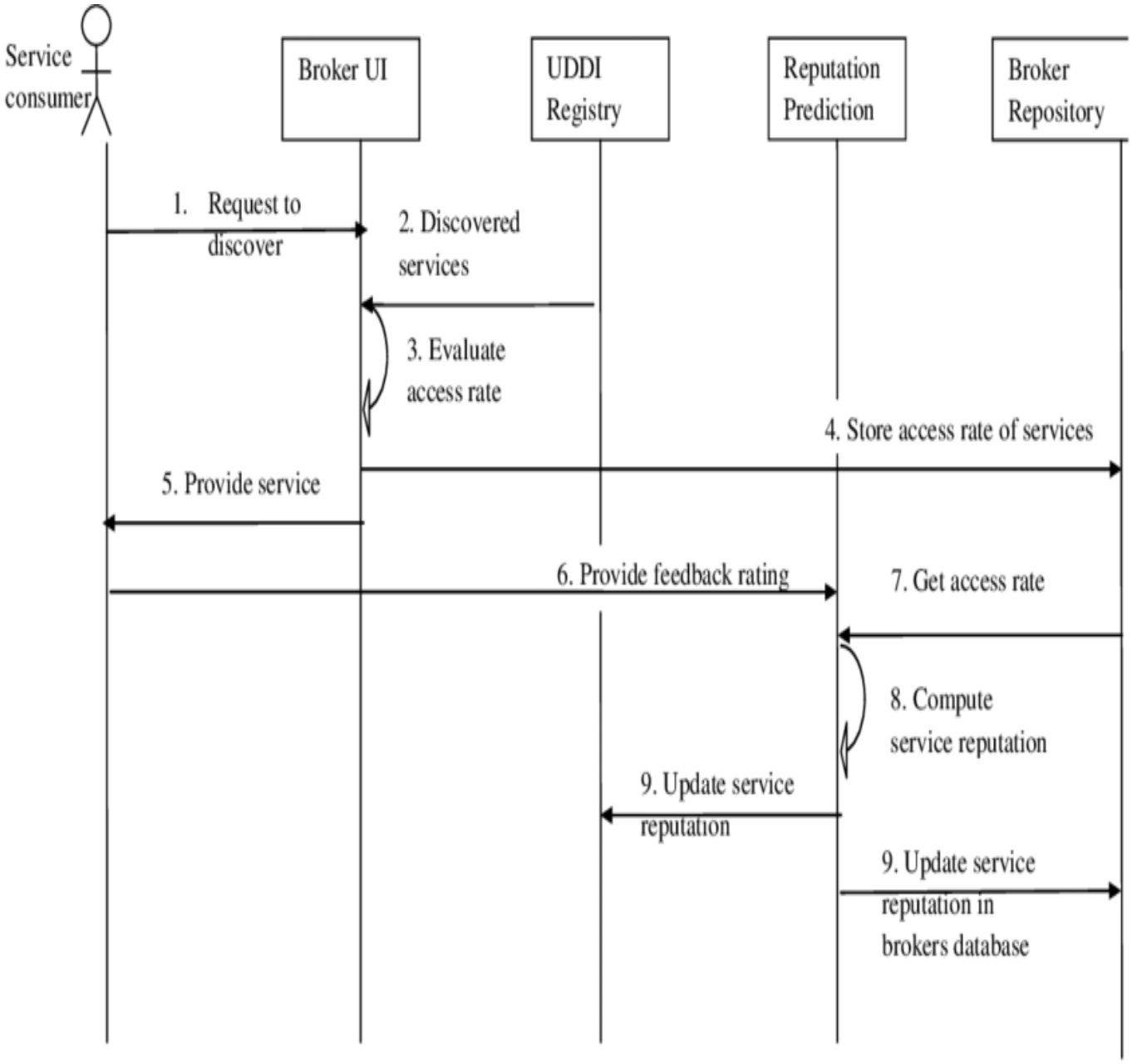

Fig4.1:SEQUENCE DIAGRAM

4.1.2 BACK-END INFRASTRUCTURE

The Backend is the engine room of the House Price Prediction system, responsible for processing requests, applying business logic, and coordinating interactions between various components. Built using robust web frameworks such as Django or Flask for Python-based systems, or Node.js for JavaScript environments, the backend is structured around a RESTful API architecture. When a user submits property details via the UI, the backend receives the data, validates it for completeness and correctness, and then passes it to the preprocessing module. The backend also manages session control, handles authentication (if required), and logs each transaction for monitoring and analytics purposes.

4.2 DATA COLLECTION AND PREPROCESSING

In the Medicine Recommendation System, data processing begins with collecting comprehensive information about various medicines, including drug names, active ingredients, dosage forms, and brand details. This data is compiled from verified pharmaceutical sources and structured into a dataset for analysis. Preprocessing involves several critical steps to ensure the accuracy and consistency of the data. Duplicate medicine entries, missing values, and inconsistent ingredient names are cleaned and standardized. Tokenization and vectorization techniques are applied to transform the textual data into numerical formats suitable for machine learning models. A similarity matrix is then generated using algorithms like cosine similarity, enabling efficient comparison between medicines based on their compositions. This clean, structured, and preprocessed data ensures that the system delivers accurate, fast, and meaningful alternative medicine recommendations. The preprocessing module also performs feature engineering—creating new variables that might enhance model accuracy. For example, it could calculate the age of the property from the construction year or create a location-based price index. Outlier detection and treatment are also part of this module, which prevents skewed model behavior due to extreme values. The preprocessed data is then structured into an array or data frame format compatible with the machine learning model's input schema. This module is implemented as a separate, reusable service or function within the backend and can also be integrated into the training pipeline to ensure that both training and real-time prediction follow the same data transformation logic.

4.3 SYSTEM WORK FLOW

Categorical variables such as property type, location, and building condition are converted into numerical format using techniques like one-hot encoding or label encoding, depending on the nature of the data and the model's requirements. Additionally, continuous variables such as lot size or square footage may undergo normalization or standardization to ensure consistent scaling across features, improving model performance. The preprocessing module also performs feature engineering—creating new variables that might enhance model accuracy. For example, it could calculate the age of the property from the construction year or create a location-based price index. Outlier detection and treatment are also part of this module, which prevents skewed model behavior due to extreme values. The preprocessed data is then structured into an array or data frame format compatible with the machine learning model's input schema. This module is implemented as a separate, reusable service or function within the backend and can also be integrated into the training pipeline to ensure that both training and real-time prediction follow the same data transformation logic. Ultimately, preprocessing ensures that the input data is clean, consistent, and informative—directly influencing the accuracy and reliability of the house price predictions.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

The implementation of a House Price Prediction system involves the integration of various software components, data science techniques, and machine learning models into a unified application that delivers accurate property price estimates. The process begins with data collection, where historical house sale records are gathered from sources such as government property registries, real estate platforms, or open datasets like the Kaggle House Prices dataset. This data typically includes features such as number of rooms, floor area, number of bathrooms, lot size, property location, year of construction, and other neighborhood-related variables. The collected dataset undergoes a thorough cleaning process, where missing values are handled, outliers are treated, and incorrect entries are corrected. This is followed by data preprocessing, which includes encoding categorical variables like building type and location using one-hot or label encoding techniques and scaling continuous variables to ensure uniformity for model training.

After preprocessing, feature selection and engineering are performed to identify the most influential factors affecting house prices. New features such as property age or proximity to city centers may be created to improve model performance. Once the data is prepared, it is split into training and testing sets. The training set is used to build machine learning models such as Linear Regression, Decision Trees, Random Forest, Gradient Boosting, or more advanced algorithms like XGBoost and LightGBM. Each model is trained using a supervised learning approach, where the algorithm learns the relationship between the input features and the known house prices. Evaluation metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2 Score are used to assess the performance of each model on the testing set, and the model with the best accuracy is selected for deployment.

5.2 OUTPUT SCREENSHOTS

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# 2. Load CSV
csv_url = "https://raw.githubusercontent.com/ageron/handson-ml/master/datasets/housing/housing.csv"
housing = pd.read_csv(csv_url)

# 3. Prepare data
# Drop rows with missing values
housing = housing.dropna()

# Convert categorical 'ocean_proximity' to numbers (simple)
housing['ocean_proximity'] = housing['ocean_proximity'].astype('category').cat.codes

# Split features and target
X = housing.drop('median_house_value', axis=1)
y = housing['median_house_value']

# 4. Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 5. Create model
model = LinearRegression()

# 6. Train model
model.fit(X_train, y_train)

# 7. Predict
y_pred = model.predict(X_test)

# 8. Evaluate
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R^2 Score: {r2:.2f}")

# 9. Predict new house (optional)
new_data = np.array([[ -122.23, 37.88, 41.0, 880.0, 129.0, 322.0, 126.0, 8.3252, 0]])  # Example
predicted_price = model.predict(new_data)
print(f"Predicted House Price: ₹{predicted_price[0]*100:.2f}")
# 8. Evaluate
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse:.2f}")
print(f"R^2 Score: {r2:.2f}")

# 8A. Graphs (import matplotlib)
import matplotlib.pyplot as plt

# Scatter Plot: Actual vs Predicted
plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, alpha=0.5, color='b')
plt.xlabel("Actual Median House Value")
plt.ylabel("Predicted Median House Value")
plt.title("Actual vs Predicted House Values")
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)  # Diagonal line
plt.grid(True)
plt.show()
```

```
# Residuals Plot
residuals = y_test - y_pred
plt.figure(figsize=(8,6))
plt.scatter(y_pred, residuals, alpha=0.5, color='r')
plt.axhline(0, color='black', linestyle='--')
plt.xlabel("Predicted Values")
plt.ylabel("Residuals (Actual - Predicted)")
plt.title("Residuals vs Predicted Values")
plt.grid(True)
plt.show()
```

Fig 5.1. Linear Regression

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load data from a CSV file
data = pd.read_csv("california_housing_sample.csv")  # <- Your CSV file name here

# Separate features and target
X = data.drop("MedHouseVal", axis=1)
y = data["MedHouseVal"]

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the Random Forest Regressor
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse:.4f}")
print(f"R² Score: {r2:.4f}")

# Feature importance plot
importances = model.feature_importances_
indices = np.argsort(importances)[::-1]

plt.figure(figsize=(10, 6))
plt.title("Feature Importances")
plt.bar(range(X.shape[1]), importances[indices], align='center')
plt.xticks(range(X.shape[1]), X.columns[indices], rotation=45)
plt.tight_layout()
plt.show()
```

Fig5.2 Random Forest Classifier

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import xgboost as xgb
import matplotlib.pyplot as plt

# Load CSV data
data = pd.read_csv("california_housing_sample.csv")  # Make sure this file is in the same directory

# Split into features and target
X = data.drop("MedHouseVal", axis=1)
y = data["MedHouseVal"]

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train XGBoost regressor
model = xgb.XGBRegressor(objective='reg:squarederror', n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse:.4f}")
print(f"R² Score: {r2:.4f}")

# Plot feature importance
xgb.plot_importance(model)
plt.title("XGBoost Feature Importance")
plt.tight_layout()
plt.show()
```

Fig5.4 XG Booster Algorithm

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
import xgboost as xgb
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt

# Load CSV data
data = pd.read_csv("california_housing_sample.csv")  # Make sure this file is in the same directory

# Split into features and target
X = data.drop("MedHouseVal", axis=1)
y = data["MedHouseVal"]

# Check for constant values in the target column
if y.nunique() <= 1:
    raise ValueError("Target variable has only one unique value. R² cannot be calculated.")

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize models
models = {
    'Linear Regression': LinearRegression(),
    'Random Forest': RandomForestRegressor(n_estimators=100, random_state=42),
    'XGBoost': xgb.XGBRegressor(objective='reg:squarederror', n_estimators=100, random_state=42)
}

# Store results for comparison
results = {'MSE': {}, 'R²': {}}

# Train and evaluate each model
for model_name, model in models.items():
    # Train model
    model.fit(X_train, y_train)

    # Predict
    y_pred = model.predict(X_test)

    # Evaluate model
    mse = mean_squared_error(y_test, y_pred)
    try:
        r2 = r2_score(y_test, y_pred)
    except:
        r2 = float('nan')  # If R² cannot be calculated, set it to NaN

    # Save results
    results['MSE'][model_name] = mse
    results['R²'][model_name] = r2

# Convert results to a DataFrame
results_df = pd.DataFrame(results)

# Plot comparison of models' performance
ax = results_df.plot(kind='bar', figsize=(10, 6), width=0.8)
plt.title('Model Performance Comparison (MSE and R² Score)')
plt.ylabel('Score')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```

Fig5.5 Comparison of three agorithms

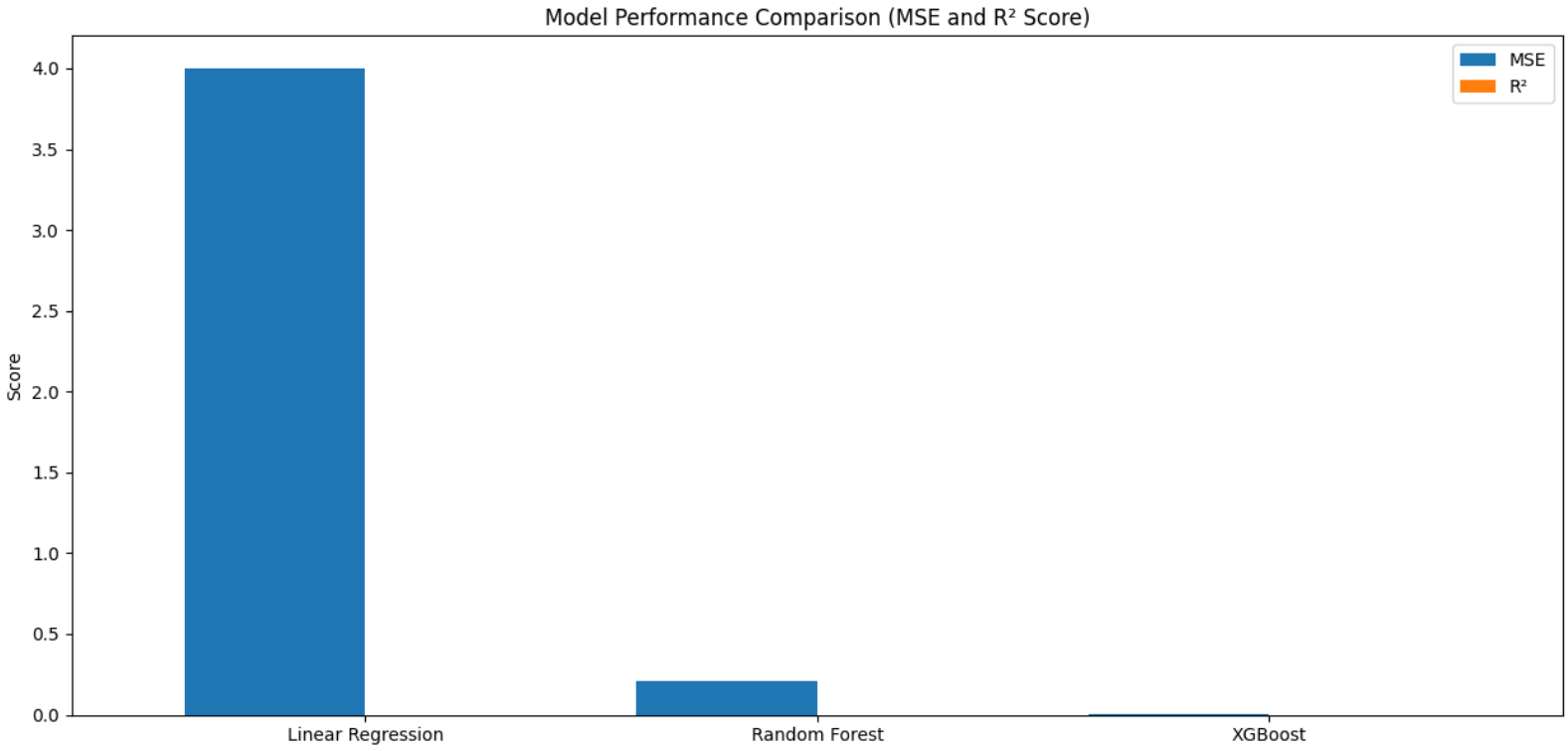

Fig5.6 MSE Scores of the above code

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The implementation of the House Price Prediction system has demonstrated the powerful synergy between data science and real estate decision-making. Through a comprehensive process involving data collection, cleaning, preprocessing, feature engineering, model training, and deployment, the system successfully predicts housing prices based on a range of user-input variables such as property size, number of bedrooms and bathrooms, location, and year of construction. The use of advanced machine learning models such as Random Forest, Gradient Boosting, or XGBoost has enabled the system to capture complex patterns and relationships in the data that traditional statistical methods might overlook. The project also incorporates an intuitive user interface and a responsive backend, making the solution accessible to a variety of users—from real estate agents and homebuyers to financial analysts and developers. By transforming raw data into actionable insights, the system aids stakeholders in making informed decisions, reducing the uncertainty associated with real estate investment, and optimizing pricing strategies.

6.2 FUTURE ENHANCEMENT

Another promising direction is the implementation of deep learning models like neural networks or hybrid models that combine structured and unstructured data. These models could be particularly effective in high-variance markets or in identifying nonlinear relationships among features. On the deployment side, the system can benefit from enhanced scalability and reliability through microservices architecture and cloud-native solutions, ensuring high availability and ease of updates. Security enhancements, such as user authentication, data encryption, and role-based access control, would also be necessary as the platform moves toward broader public or commercial use.

REFERENCES

1. Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (2nd ed.). Sebastopol: O'Reilly Media.
2. Raschka, S., & Mirjalili, V. (2020). Python Machine Learning (3rd ed.). Birmingham: Packt Publishing.
3. Ng, A. (2018). Machine Learning Yearning. Palo Alto: Deeplearning.ai.
4. Provost, F., & Fawcett, T. (2013). Data Science for Business. Sebastopol: O'Reilly Media.
5. Bruce, P., Bruce, A., & Gedeck, P. (2020). Practical Statistics for Data Scientists (2nd ed.). Sebastopol: O'Reilly Media.
6. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning (2nd ed.). New York: Springer.
7. Brett, D. L., & Schmitz, A. (2019). Real Estate Market Analysis (3rd ed.). Washington, D.C.: Urban Land Institute.
8. Hastie, T., Tibshirani, R., & Friedman, J. (2017). The Elements of Statistical Learning (2nd ed.). New York: Springer.
9. Smith, J. (2020). Web Development: A Complete Guide (2nd ed.). New York: Pearson Education.
10. Patel, R., & Kumar, A. (2021). Introduction to Flask: A Beginner's Guide to Web Development with Python. New York: Wiley.
11. Sharma, N., & Gupta, S. (2020). "Machine Learning in Healthcare: An Overview." International Journal of Health Informatics, 12(3), 88–97.
12. Zhang, L., & Wang, X. (2019). "Recommendation Systems: Techniques and Applications." Journal of Intelligent Systems, 18(4), 65–77.
13. Aggarwal, C. C. (2018). Neural Networks and Deep Learning: A Textbook. Cham: Springer.
14. Brownlee, J. (2020). Machine Learning Mastery With Python. Victoria: Machine Learning Mastery.
15. Alpaydin, E. (2020). Introduction to Machine Learning (4th ed.). Cambridge: MIT Press.