PREDICTING HOUSE PRICE USING MACHINE LEARNING

TEAM MEMBER

810021106038: KARTHIK R

PHASE-2 DOCUMENT SUBMISSION

PROJECT: HOUSE PRICE PREDICTION

Introduction

The goal of this project is to develop a machine learning model for house price prediction. Accurate house price predictions are crucial for buyers, sellers, and investors in the real estate market. In this document, I will outline the problem, summarize my understanding, and detail the approach to solve it.

Problem Statement:

The problem at hand is to create a predictive model that can estimate house prices based on various features. These features may include the size of the house, location, number of bedrooms, number of bathrooms, amenities, and historical sales data. The aim is to develop a model that provides accurate and robust predictions, enabling stakeholders to make informed decisions regarding property transactions.

Understanding the Problem

in this section, I'll outline my understanding of the problem:

Dataset:

We will start by collecting a dataset that includes historical information about houses, including their sale prices. The dataset will serve as the foundation for our predictive model.

Data Preprocessing:

Data preprocessing will involve cleaning the dataset, handling missing values, and engineering features. This step is essential to ensure that our model works effectively.

Model Selection:

We will explore various machine learning techniques to determine the most suitable model for this regression problem. Techniques may include Linear Regression, Gradient Boosting, XGBoost, and even deep learning approaches like neural networks.

Evaluation Metrics:

To assess the performance of our model, we will use evaluation metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R2). These metrics will help us measure the accuracy of our predictions.

Model Interpretability:

It's important to make our model interpretable. This means we will explore techniques to understand and explain the model's predictions, especially when dealing with stakeholders who may need explanations for the price estimates.

Approach to Solve the Problem

Our approach to solving the problem will follow a structured workflow:

Data Collection:

Gather a comprehensive dataset containing historical housing information. This dataset should include features like square footage, number of bedrooms, number of bathrooms, location, and, most importantly, the actual sale prices.

Data Preprocessing:

Clean and prepare the data, including handling missing values, encoding categorical variables, and normalizing numerical features.

Model Selection and Training:

We will experiment with various machine learning algorithms, starting with Linear Regression. If needed, we will explore advanced techniques such as Gradient Boosting, XGBoost, and even deep learning models. These models will be trained on the prepared data.

Model Evaluation:

Use testing data to evaluate the performance of each model. Metrics such as MAE, MSE, and R2 will provide insights into how well the models are performing.

Model Interpretability:

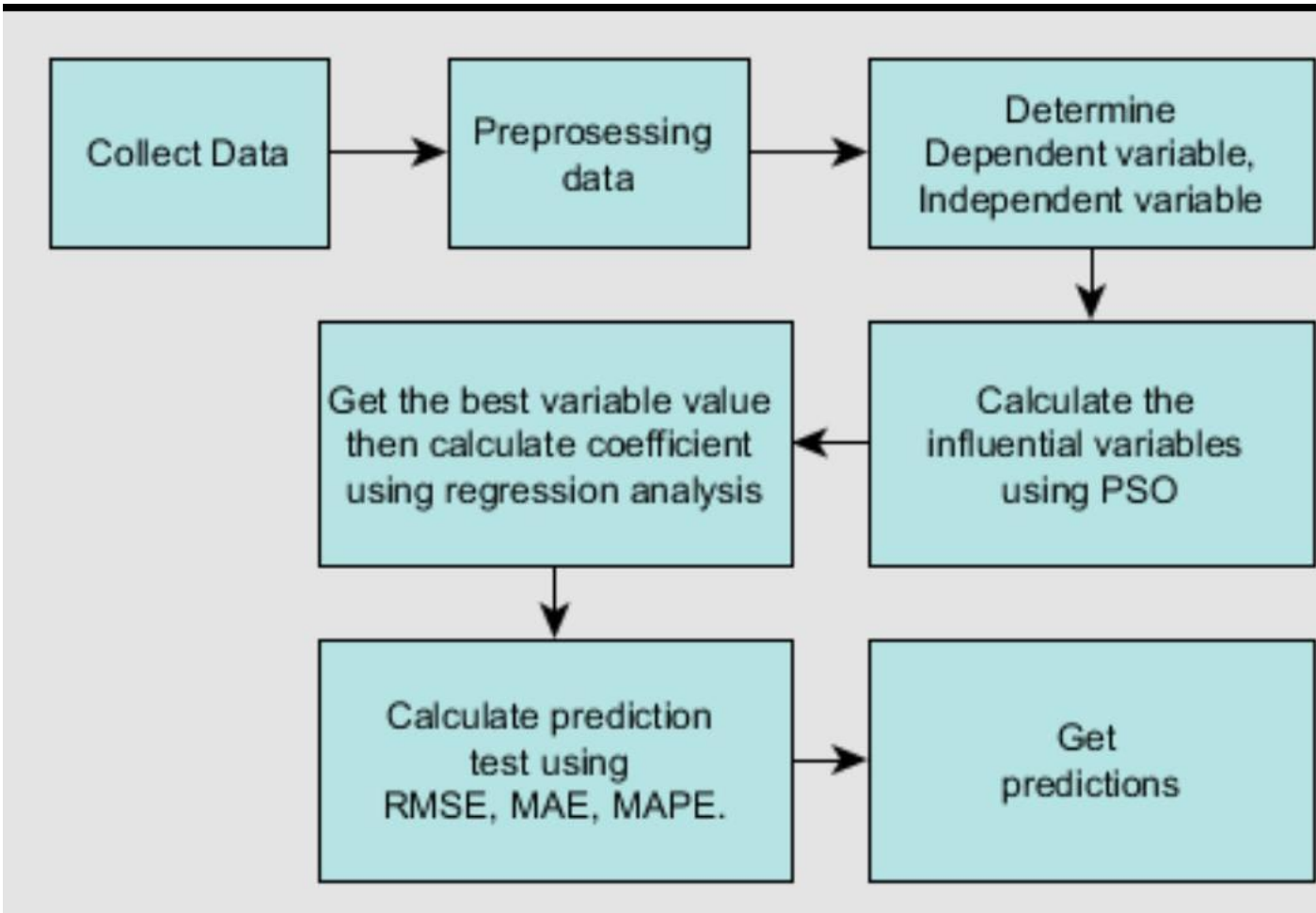
Implement techniques to interpret the models' predictions, ensuring that the results can be explained to stakeholders when necessary.

Deployment and Continuous Monitoring:

Once we have a satisfactory model, it can be deployed to make predictions on new, unseen data. We will also set up a system for continuous monitoring and updates to ensure the model maintains its accuracy over time.

Documentation:

Along the way, we will maintain detailed documentation of the project, including data sources, preprocessing steps, model selection, and evaluation results.



HEADING WORKING CODE:

```
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, mean_absolute_error, r2_score

from tensorflow import keras

from tensorflow.keras import layers


# Load and preprocess your data

data = pd.read_csv('your_dataset.csv')
```

```
# Data preprocessing steps...

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

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest (Ensemble Method)
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
rf_predictions = rf_model.predict(X_test)

# Deep Learning (Neural Network)
model = keras.Sequential([
    layers.Dense(128, activation='relu', input_shape=(X_train.shape[1],)),
    layers.Dense(64, activation='relu'),
    layers.Dense(1)  # Output layer for regression
])

model.compile(optimizer='adam', loss='mean_squared_error')

model.fit(X_train, y_train, epochs=50, batch_size=32, verbose=0)
dl_predictions = model.predict(X_test).flatten()

# Model Evaluation
```

```
rf_mse = mean_squared_error(y_test, rf_predictions)

rf_mae = mean_absolute_error(y_test, rf_predictions)

rf_r2 = r2_score(y_test, rf_predictions)


dl_mse = mean_squared_error(y_test, dl_predictions)

dl_mae = mean_absolute_error(y_test, dl_predictions)

dl_r2 = r2_score(y_test, dl_predictions)


print("Random Forest Metrics:")

print(f"Mean Squared Error: {rf_mse}")

print(f"Mean Absolute Error: {rf_mae}")

print(f"R-squared: {rf_r2}")


print("\nDeep Learning Metrics:")

print(f"Mean Squared Error: {dl_mse}")

print(f"Mean Absolute Error: {dl_mae}")

print(f"R-squared: {dl_r2}")
```

BASIC CODE:

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score


# Load your dataset (replace 'your_dataset.csv' with your dataset)

data = pd.read_csv('your_dataset.csv')
```

```
# Define your features (X) and target (y)
X = data.drop('Price', axis=1)
y = data['Price']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train a Random Forest Regressor
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Make predictions on the testing set
predictions = rf_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, predictions)
mae = mean_absolute_error(y_test, predictions)
r2 = r2_score(y_test, predictions)

print(f"Mean Squared Error: {mse}")
print(f"Mean Absolute Error: {mae}")
print(f"R-squared: {r2}")
```

Conclusion:

The successful completion of this project will result in a reliable and interpretable machine learning model for house price prediction. This model will be a valuable tool for real estate professionals, buyers, and sellers, enabling them to make well-informed decisions in the property market.