CS204.2 – Practical Assessment

Nipun Goyal

(270249563)

I have chosen to work upon a scenario requiring reduction in decision-making time such as house price prediction based upon several factors such as the area, number of bedrooms, bathrooms, number of stories, number of parking spots, etc. This model will help us to quickly evaluate or predict the price of houses in the location depending on these important factors. For this project, I used linear regression approach as it would give a streamlined answer and easily help in the prediction.

There are several other factors though which we can manually evaluate without involvement of machine learning but by incorporating machine learning-based solutions into the manual evaluation process, analysts can streamline workflow, improve accuracy, and gain deeper insights into the performance of linear regression models for predicting house prices.

I tracked the outcome from the AI model to ensure its fairness and accuracy. In conclusion, while evaluation metrics like MSE and accuracy provide quantitative measures of model performance, addressing bias, fairness, and ethical concerns is equally crucial. By integrating fairness assessments, bias detection and mitigation strategies, transparency, and continuous monitoring into the model development lifecycle, organizations can uphold high standards of model creation, promote fairness and accuracy, and mitigate potential ethical risks.

For this scenario of developing a project predicting house prices accurately, I used a dataset from Kaggle that was uploaded in 2022. (URL- https://www.kaggle.com/datasets/yasserh/housing-prices-dataset?resource=download). This dataset was chosen as it fulfilled my requirements needed for accurate prediction of house prices. It contained various factors on which the house price depends such as the area of the house, the number of bathrooms, bedrooms and many more.

Splitting the dataset into training and testing sets is crucial for assessing the performance and generalization ability of a linear regression model. This process involves randomly or stratified partitioning the data into subsets for training and evaluation. In Python, libraries like scikit-learn provide functions such as 'train_test_split' to accomplish this. The training set is used to train the model, while the testing set is reserved for evaluating its performance on unseen data. This helps prevent overfitting and ensures the model's ability to make accurate predictions in real-world scenarios. To split the dataset into training and testing sets using linear regression:

1. Use scikit-learn's ‘train_test_split’ function to split the data.
2. Instantiate the linear regression model.
3. Fit the model to the training data.
4. Evaluate the model's performance using metrics like Mean Squared Error on the testing set.

The approach for managing missing values, addressing outliers, performing feature engineering, and optimizing hyperparameters involves a combination of data exploration, decision-making based on domain knowledge, and implementation using appropriate tools and techniques. This iterative process helps improve the quality and predictive performance of the linear regression model for house price prediction.

Integrating deep learning into the house price prediction model can enhance results by leveraging its capabilities in feature learning, handling non-linear relationships, accommodating high-dimensional data, offering flexibility and adaptability, enabling end-to-end learning, and leveraging transfer learning. However, it's essential to consider factors such as data availability, computational resources, and interpretability when deciding whether to incorporate deep learning into the modeling approach.

Precision and recall scores are crucial for evaluating model performance, particularly in scenarios where false positives or false negatives have significant implications like in ours of whether the price will go up or come down depending upon the varied factors. Strategies for enhancing these scores include improving feature selection, fine-tuning model parameters, addressing class imbalance, utilizing feature engineering techniques, experimenting with different algorithms, and balancing precision and recall based on application requirements.

Visualization techniques play a vital role in every stage of the machine learning pipeline, from data exploration and model interpretation to evaluation and improvement. The confusion matrix, in particular, serves as a cornerstone for assessing classification model performance and guiding model refinement in machine learning tasks. Therefore in this project as well, I have plotted graphs depicting the outcome in a very simple and easy manner.

The simple steps followed during this project were –

1. **Data collection** – The suitable dataset was found which in my case was from Kaggle.
2. **Reading the data** – The dataset is gone through to find if there are any anomalies.
3. **Data normalization** – The data is cleaned and used to rescale the features of a dataset to a similar scale without distorting differences in the ranges of values.

4. **Split data into training and test data set** in ratio of 80/20 or 70/30 – This approach helps assess the model's ability to generalize to unseen data and provides an estimate of its performance in real-world scenarios.
5. **Model training** – The model learns the patterns and relationships present in the training data, allowing it to make predictions on new, unseen data.
6. **Plotting results/Visualization** – Visualization techniques can be applied for residual analysis and feature importance visualization to gain further insights into the model's performance and behavior
7. **Evaluation error or Performance stats** like MSE/RMSE – Printing the MSE and RMSE provides insight into the model's accuracy and performance.

CODE SNIPPET –

```
# Importing necessary libraries
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
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
house_data = pd.read_csv('C:/Users/user1/Desktop/204.2 - Practical/Housing.csv')

# Assuming 'X' contains features (e.g., square footage, number of bedrooms, etc.)
# and 'y' contains the target variable (house prices)
X = house_data[['area', 'bedrooms', 'bathrooms', 'stories', 'parking']]
y = house_data['price']

# Splitting the dataset 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)

# Creating a Linear Regression model
model = LinearRegression()

# Training the model
model.fit(X_train, y_train)

# Making predictions on the testing set
y_pred = model.predict(X_test)
```

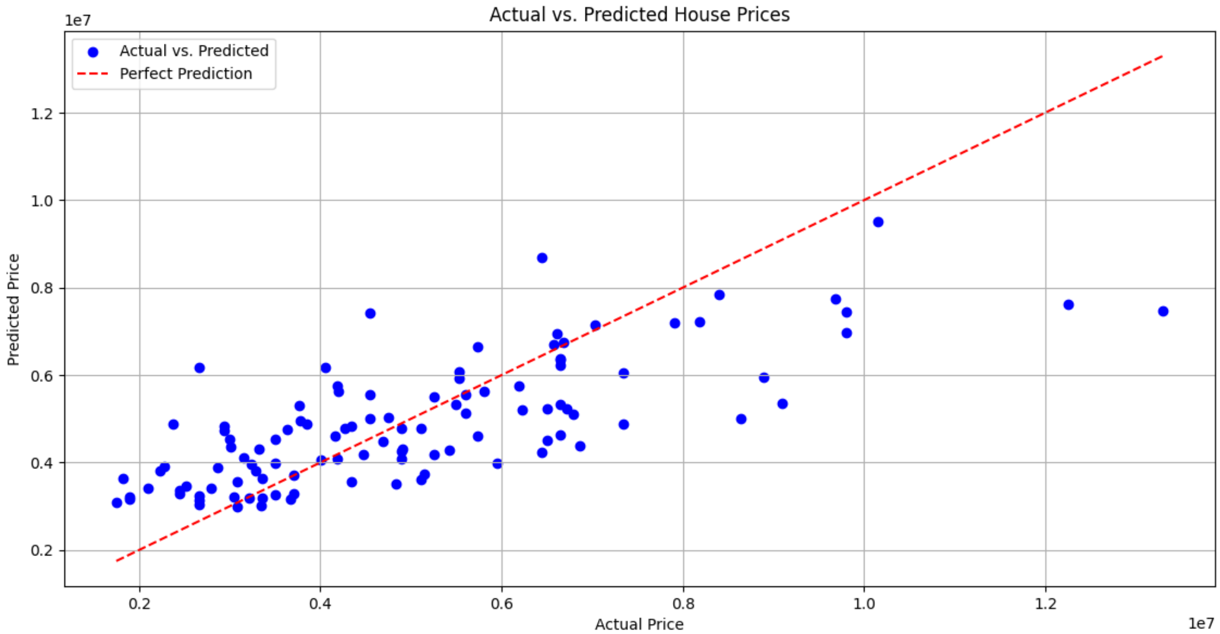
```python
# Evaluating the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error:", mse)
print("R-squared Score:", r2)

# Printing the coefficients and intercept of the linear regression model
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)

# Plotting the actual vs. predicted house prices
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', label='Actual vs. Predicted')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red',
linestyle='--', label='Perfect Prediction')
plt.title('Actual vs. Predicted House Prices')
plt.xlabel('Actual Price')
plt.ylabel('Predicted Price')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT SNIPPET –

```
# Splitting the dataset into training and testing sets

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   SEARCH ERROR                Python

Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Users\user1\Desktop\204.2 - Practical> & C:/Users/user1/AppData/Local/Programs/Python/Python310/python.exe "c:/
Users/user1/Desktop/204.2 - Practical/main.py"
Mean Squared Error: 2292721545725.3623
R-squared Score: 0.5464062355495871
Coefficients: [3.08866956e+02 1.51246751e+05 1.18573171e+06 4.95100763e+05
 3.37660830e+05]
Intercept: 51999.67680883873
```

This snippet depicts the performance stats in the form of Mean Square Error and R-squared Score.

GRAPH SNIPPET –



This graph depicts the relationship between the dependent and independent variables showcasing the prediction for house prices.