Car Dheko-Used Car Price Prediction

1. Introduction

This project focuses on predicting the prices of used cars using various machine learning techniques. The prediction model was developed and deployed using Stream lit, allowing users to input car details and obtain real-time price predictions. The project involves data cleaning, exploratory data analysis (EDA), model development, and deployment.

2. Data Collection and Pre processing

2.1 Data Sources

- The dataset contains information on used cars, including details like kilo meters driven (`km`), year of manufacture (`year`), and other relevant features.

2.2 Data Cleaning

- **Missing Values**: Missing or erroneous values were handled through imputation or removal.
- **Data Formatting**: Columns like `km` were converted from strings with commas to numeric values, and the `year` column was ensured to be numeric.
- **Feature Engineering**: An additional `age` feature was created by subtracting the year of manufacture from the current year.

2.3 Sample Code for Data Pre processing

```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

data = {'km': ['1,20,000', '30,000', '50,000'], 'year': ['2015', '2016', '2017']}
df_all_cities = pd.DataFrame(data)

# Clean and convert columns
df_all_cities['km'] = df_all_cities['km'].str.replace(',', '').astype(float)
df_all_cities['year'] = pd.to_numeric(df_all_cities['year'], errors='coerce')

# Feature engineering: Adding 'age' column
df_all_cities['age'] = 2024 - df_all_cities['year']
```

3. Exploratory Data Analysis (EDA)

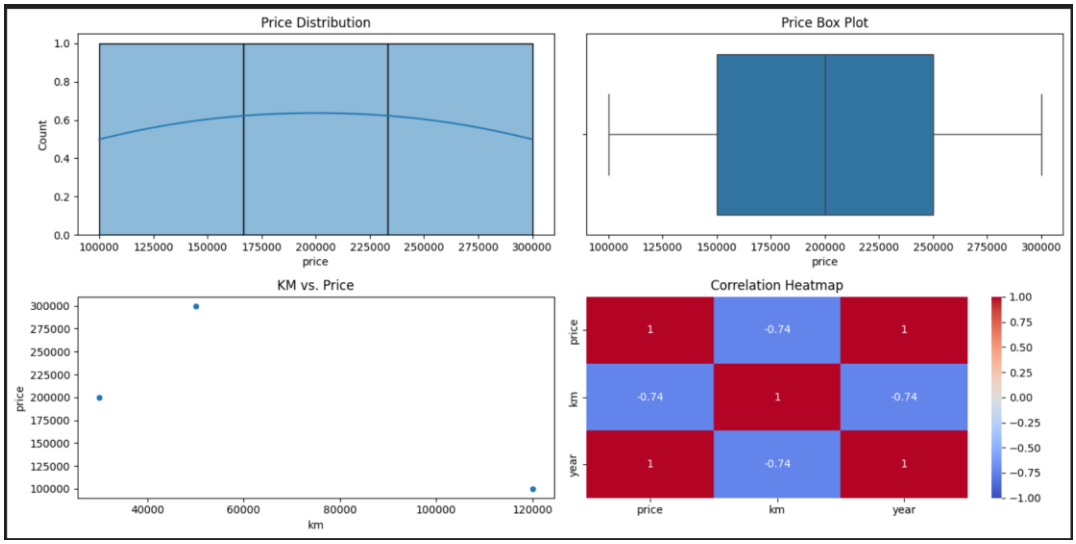
3.1 Descriptive Statistics

- Calculated summary statistics (mean, median, mode, standard deviation) for key numerical features like km, year, and price.
- Identified patterns and outliers within the dataset.

3.2 Data Visualization

- **Scatter Plots**: Used to identify relationships between car prices and features like km and year.
- **Histograms**: Plotted to visualize the distribution of numerical features.
- **Box Plots**: Utilized to detect outliers in the dataset.
- **Correlation Heat maps**: Generated to understand the relationships between different features.

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Create visualizations
plt.figure(figsize=(14, 7))

# Histogram of price
plt.subplot(2, 2, 1)
sns.histplot(df_all_cities['price'], kde=True)
plt.title('Price Distribution')
# plt.show()
# Box plot of price
plt.subplot(2, 2, 2)
sns.boxplot(x=df_all_cities['price'])
plt.title('Price Box Plot')

# Scatter plot of km vs. price
plt.subplot(2, 2, 3)
sns.scatterplot(x=df_all_cities['km'], y=df_all_cities['price'])
plt.title('KM vs. Price')

# Correlation heatmap
plt.subplot(2, 2, 4)
corr = df_all_cities.corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap')

plt.tight_layout()
plt.show()
```



3.3 Sample Code for EDA

```
import matplotlib.pyplot as plt
import seaborn as sns

# Scatter plot
sns.scatterplot(x='km', y='price', data=df_all_cities)
plt.title('Scatter Plot of KM vs Price')
plt.show()

# Correlation heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(df_all_cities.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

4. Model Development

4.1 Train-Test Split

- The dataset was split into training and testing sets using an 80-20 ratio.

4.2 Model Selection

- Various machine learning models were considered, including Linear Regression, Decision Trees, and Random Forests.
- The final model was selected based on performance metrics like Mean Absolute Error (MAE) and R-squared.

4.3 Model Training

- Models were trained on the training dataset using cross-validation to ensure robust performance.

4.4 Hyper parameter Tuning

- Grid Search was used to optimize model parameters and improve accuracy.

4.5 Sample Code for Model Training

```
import streamlit as st
import joblib
import pandas as pd

model = joblib.load('car_price_model.pkl')

st.title('Car Price Prediction')

km = st.number_input('Enter KM:', min_value=0, step=1000)
year = st.number_input('Enter Year:', min_value=2000, max_value=2024, step=1)
age = 2024 - year

input_features = pd.DataFrame([[km, year, age]], columns=['km', 'year', 'age'])

if st.button('Predict Price'):
    predicted_price = model.predict(input_features)
    st.write(f"Predicted Price: ₹ {predicted_price[0]:,.2f}")
```

5. Model Evaluation

5.1 Performance Metrics

- **Mean Absolute Error (MAE)**: Measures the average magnitude of errors in predictions.
- **R-squared**: Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

5.2 Model Comparison

- Different models were compared based on MAE and R-squared values to select the best-performing model.

6. Deployment

6.1 Stream lit Application

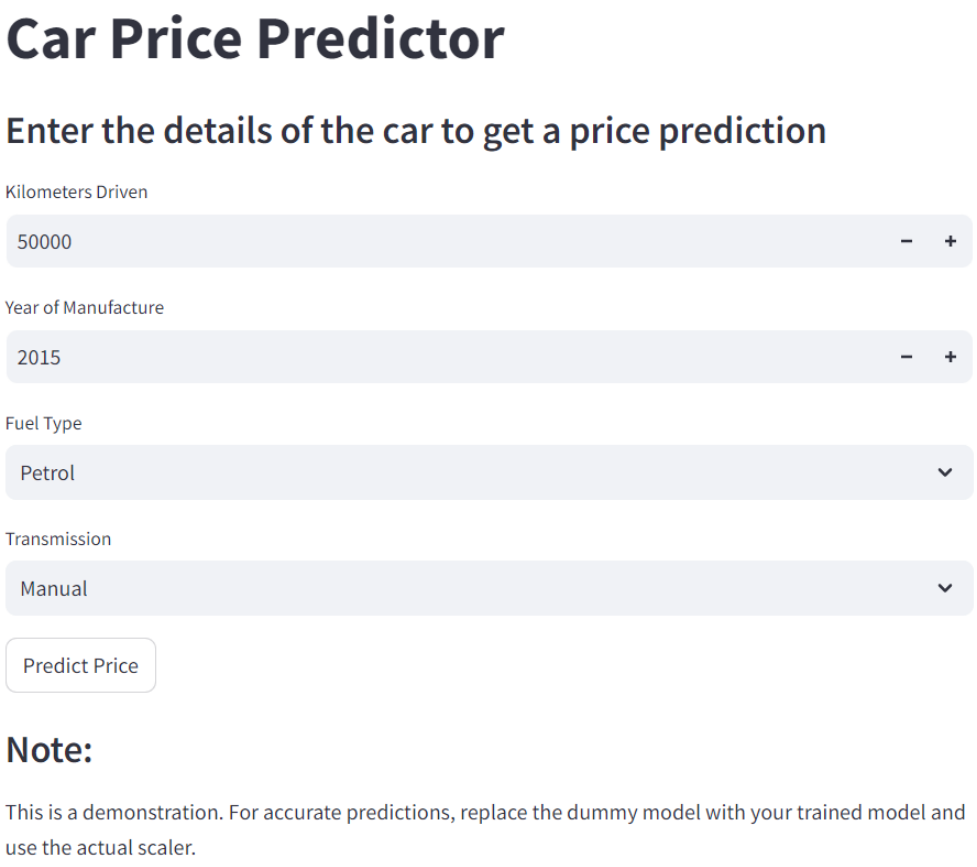
- The final model was deployed using Stream lit to create an interactive web application where users can input car details and get real-time price predictions.

6.2 User Interface Design

- The application was designed to be user-friendly, with clear instructions and error handling mechanisms.

6.3 Sample Code for Streamlit Deployment

```
import streamlit as st
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler
class DummyModel:
    def predict(self, X):
        return np.random.rand(len(X)) * 100000
model = DummyModel()
scaler = StandardScaler()
st.title('Car Price Predictor')
st.write("""
### Enter the details of the car to get a price prediction
""")
km = st.number_input('Kilometers Driven', min_value=0, max_value=500000, value=50000)
year = st.number_input('Year of Manufacture', min_value=1990, max_value=2023, value=2015)
fuel_type = st.selectbox('Fuel Type', ['Petrol', 'Diesel', 'CNG'])
transmission = st.selectbox('Transmission', ['Manual', 'Automatic'])
input_data = pd.DataFrame({
    'km': [km],
    'year': [year],
    'fuel_type': [fuel_type],
    'transmission': [transmission]
})
input_data = pd.get_dummies(input_data, columns=['fuel_type', 'transmission'])
expected_columns = ['km', 'year', 'fuel_type_Petrol', 'fuel_type_Diesel', 'fuel_type_CNG',
                    'transmission_Manual', 'transmission_Automatic']
for col in expected_columns:
    if col not in input_data.columns:
        input_data[col] = 0
input_data = input_data[expected_columns]
if st.button('Predict Price'):
    prediction = model.predict(input_data)
    st.success(f'The predicted price is ₹{prediction[0]:,.2f}')
st.write("""
### Note:
This is a demonstration. For accurate predictions, replace the dummy model with your trained model and use the actual scaler.
""")
```



7. Conclusion

This project successfully developed a machine learning model to predict used car prices based on various features like kilometers driven and year of manufacture. The model was deployed using Streamlit, providing an interactive platform for real-time predictions. Future work could involve adding more features, refining the model, and enhancing the user interface for better usability.