MAJOR PROJECT

TITLE:

CAR SELLING PRICE AND ANALYSIS WEB APPLICATION

PROJECT CATEGORY:

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

REGISTER ID:

 # 122306207

STUDENT NAME:

SARATHY.S

CONTACT DETAILS:

EMAIL: sarusarathy0401@gmail.com

PH-NO: +91 **8807770831**

CAR SELLING PRICE PREDICTION AND ANALYSIS WEB APPLICATION

Objective:

The objective of this project is to create an interactive web application that predicts the selling price of a used car using a machine learning model and provides detailed insights into the factors influencing the price. The application utilizes **Linear Regression** to predict car prices based on features such as the car's year of manufacture, mileage, fuel type, and seller type. The project is developed using **Streamlit** for the web interface, **scikit-learn** for machine learning, and **Plotly** for data visualization.

Project Overview:

This major project is organized into three key parts: **data preprocessing**, **model training**, and **user interface (UI) development**. The web application allows users to interactively input car details and receive predicted selling prices. It also provides insightful analysis of the data and the machine learning model's performance, including feature importance and prediction accuracy.

1. Modules and Packages Used:

- **Streamlit**:
  - Used for creating the interactive web interface. Streamlit enables the easy deployment of machine learning models and data visualizations in a user-friendly environment.
- **Pandas**:
  - Used for data manipulation and cleaning. The library facilitates loading the dataset, encoding categorical variables, and performing statistical operations on the data.
- **NumPy**:
  - Used for numerical operations, especially when dealing with large arrays or matrices of data.

- **Scikit-learn**:
  - Provides tools for splitting the dataset, training the machine learning model (Linear Regression), and evaluating model performance through metrics like **Mean Absolute Error (MAE)** and **R^2 Score**.
- **Plotly**:
  - Used to create interactive visualizations like scatter plots, bar charts, and trendlines. This library enhances the exploratory data analysis (EDA) process and helps visualize the relationship between features and selling prices.
- **LabelEncoder (from scikit-learn)**:
  - Used for encoding categorical features (`Fuel Type` and `Seller Type`) into numeric values, enabling them to be used as inputs for the machine learning model.

2. Dataset Description:

The dataset used for this project is a CSV file, which contains information on used cars listed for sale. The dataset includes the following key columns:

- **Year**: The manufacturing year of the car.
- **Mileage**: The total mileage (in kilometers) driven by the car.
- **Fuel Type**: The type of fuel the car uses (e.g., Petrol, Diesel).
- **Seller Type**: The type of seller (e.g., Individual, Dealer).
- **Selling Price**: The price at which the car is being sold (target variable for the model).

The dataset is used to train and evaluate a linear regression model that predicts the **Selling Price** based on the other attributes.

3. Data Preprocessing and Cleaning:

- **Categorical Encoding**: The `Fuel` and `Seller_Type` columns are categorical in nature. These columns are encoded into numeric values using **LabelEncoder** from **scikit-learn**. This step is essential because machine learning models cannot process string values directly.

- **Data Splitting**: The data is split into training and testing sets using **train_test_split** from **scikit-learn**. This ensures the model is trained on one subset of the data and evaluated on another, ensuring a more accurate evaluation of its performance.

4. Model Training:

- **Linear Regression -** The model used for price prediction is **Linear Regression** from **scikit-learn**. Linear regression is a suitable algorithm for this type of problem since the goal is tocontinuous numerical value (the selling price) based on other numerical and categorical features.
- **Model Evaluation**:

The model's performance is evaluated using two key metrics:

  - **Mean Absolute Error (MAE)**: Measures the average magnitude of the errors in the predictions.
  - **R^2 Score**: Indicates how well the model explains the variance in the target variable, with higher values indicating better performance.

5. User Interface (UI):

The project utilizes **Streamlit** to build an interactive UI with the following features:

- **Car Selling Price Prediction**:

Users can input the year, mileage, fuel type, and seller type of a car through a form. After submitting the form, the model predicts the car's selling price, which is then displayed to the user.

- **Dataset Overview**:

The dataset is displayed in an interactive table, where users can view the first 100 rows. Additionally, basic statistical metrics (such as mean, median, and standard deviation) are shown to help users understand the data distribution.

- **Model Insights**:

This section provides insights into the model's performance, including the **Mean Absolute Error (MAE)** and **R^2 Score**. It also includes visualizations of the relationship between features (e.g., `Year`, `Mileage`) and the selling price, as well as a bar chart showing the importance of each feature in predicting the price.

- **Data Visualizations**:
  - **Scatter Plots**: Display the relationship between car features (e.g., `Year` or `Mileage`) and the selling price, along with a trendline to show the correlation.
  - **Feature Importance Bar Chart**: A horizontal bar chart visualizing the importance of each feature (such as `Mileage`, `Year`, etc.) based on the model's coefficients.

6. Technologies Used:

- **Streamlit**: For building the interactive web application.
- **Pandas**: For handling data, cleaning, and preprocessing.
- **Scikit-learn**: For building the machine learning model and evaluating its performance.
- **NumPy**: For numerical operations.
- **Plotly**: For creating interactive charts and visualizations.
- **LabelEncoder**: For encoding categorical data into numerical values.

7. Summary:

This project provides an interactive web application that predicts the selling price of a used car and offers in-depth analysis of car features, the model's performance, and feature importance. By combining **Streamlit**, **scikit-learn**, and **Plotly**, the application offers both predictive capabilities and a clear visualization of how different car attributes influence the selling price. The machine learning model, based on **Linear Regression**, ensures accurate predictions, while the visualizations allow users to gain valuable insights into the used car market.

Potential Applications:

- **Used Car Dealers**: Quickly assess car prices and improve pricing strategies.
- **Car Buyers and Sellers**: Estimate the price of a car based on its features.
- **Data Analysts**: Gain insights into car market trends through data analysis and visualization.

This project is an effective tool for anyone involved in buying, selling, or analyzing used cars, providing both practical predictions and useful data-driven insights.

Code:

```
 import streamlit as st

import pandas as pd

import numpy as np

import plotly.express as px

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import mean_absolute_error, r2_score



# Load dataset

@st.cache_data

def load_data():

    data = pd.read_csv('C:/Users/sarus/Downloads/car_data.csv')



    # Encode categorical variables

    label_encoders = {}

    for col in ['Fuel', 'Seller_Type']:

        le = LabelEncoder()

        data[col] = le.fit_transform(data[col])

        label_encoders[col] = le  # Store encoders for later use



    return data, label_encoders
```

```
# Train model
@st.cache_resource
def train_model(data):
    X = data[['Year', 'Mileage', 'Fuel', 'Seller_Type']]
    y = data['Selling_Price']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    model = LinearRegression()
    model.fit(X_train, y_train)

    # Calculate metrics
    y_pred = model.predict(X_test)
    mae = mean_absolute_error(y_test, y_pred)
    r2 = r2_score(y_test, y_pred)

    return model, mae, r2, X_train, y_train

# Main application
def main():
    st.title('□ Car Selling Price Prediction & Analysis')

    # Load data
    data, label_encoders = load_data()
```

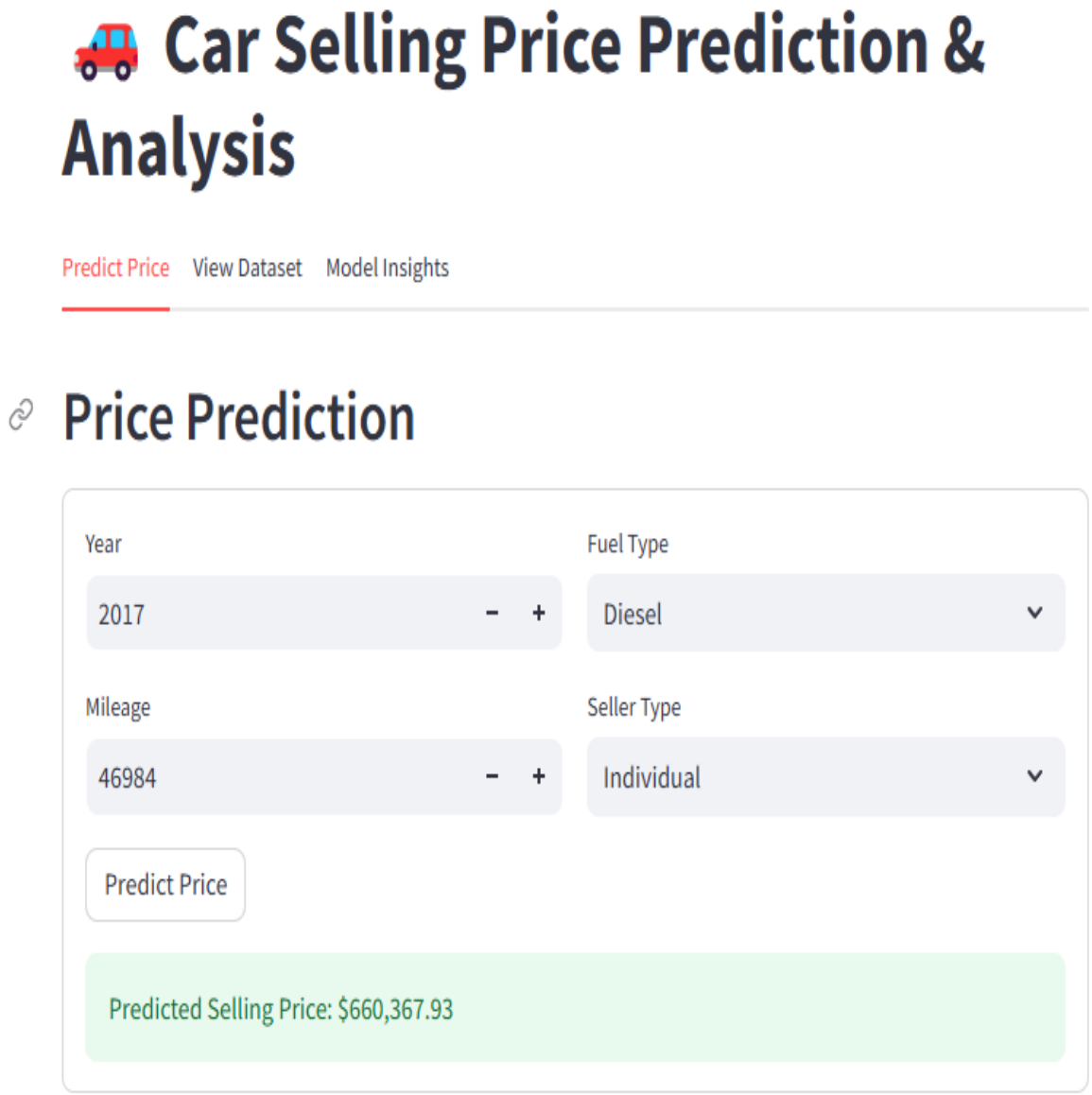
```
# Train model
model, mae, r2, X_train, y_train = train_model(data)

# Create tabs
tab1, tab2, tab3 = st.tabs(["Predict Price", "View Dataset", "Model Insights"])

with tab1:
    st.header("Price Prediction")
    with st.form("prediction_form"):
        col1, col2 = st.columns(2)
        with col1:
            year = st.number_input('Year', min_value=data['Year'].min(), max_value=data['Year'].max(), value=2020)
            mileage = st.number_input('Mileage', min_value=int(data['Mileage'].min()), value=int(data['Mileage'].median()))
        with col2:
            fuel = st.selectbox('Fuel Type', label_encoders['Fuel'].classes_)
            seller_type = st.selectbox('Seller Type', label_encoders['Seller_Type'].classes_)

        submitted = st.form_submit_button("Predict Price")
        if submitted:
            # Encode categorical inputs
            fuel_encoded = label_encoders['Fuel'].transform([fuel])[0] seller_encoded = label_encoders['Seller_Type'].transform([seller_type])[0]
```

```
            input_data = [[year, mileage, fuel_encoded, seller_encoded]]
            prediction = model.predict(input_data)[0]
            st.success(f'Predicted Selling Price: ${prediction:,.2f}')

with tab2:
    st.header("Dataset Overview")
    st.dataframe(data.head(100), height=400)

    st.subheader("Basic Statistics")
    st.write(data.describe())

with tab3:
    st.header("Model Performance")
    col1, col2 = st.columns(2)
    col1.metric("Mean Absolute Error", f"${mae:,.2f}")
    col2.metric("R² Score", f"{r2:.2%}")

    st.subheader("Feature Relationships")
    feature = st.selectbox('Select feature to plot', ['Year', 'Mileage'])
    fig = px.scatter(data, x=feature, y='Selling_Price', trendline="ols")
    st.plotly_chart(fig)

    st.subheader("Feature Importance")
```

```
        coefficients = pd.DataFrame({
            'Feature': X_train.columns,
            'Importance': model.coef_
        }).sort_values('Importance', ascending=False)
        fig2 = px.bar(coefficients, x='Importance', y='Feature', orientation='h')
        st.plotly_chart(fig2)


if __name__ == '__main__':
  main()
```

Output:

THE WEB PAGE : TAB (1)



TAB (2):

🚗 Car Selling Price Prediction & Analysis

Predict Price | View Dataset | Model Insights

Dataset Overview

	Year	Mileage	Fuel	Seller_Type	Selling_Price
2	2,018	22,000	3	0	800,000
3	2,016	55,000	1	0	650,000
4	2,019	18,000	0	0	900,000
5	2,013	68,000	3	1	400,000
6	2,017	35,000	1	0	720,000
7	2,014	74,000	2	1	300,000
8	2,020	12,000	3	0	1,100,000
9	2,011	88,000	1	1	280,000
10	2,016	49,000	0	0	600,000
11	2,018	21,000	3	0	780,000

Basic Statistics

	Year	Mileage	Fuel	Seller_Type	Selling_Price
count	12	12	12	12	12
mean	2,015.75	47,083.3333	1.75	0.3333	612,500
std	2.8324	25,928.0765	1.2154	0.4924	260,807.3131
min	2,011	12,000	0	0	280,000
25%	2,013.75	21,750	1	0	380,000
50%	2,016	47,000	1.5	0	625,000
75%	2,018	69,500	3	1	785,000
max	2,020	88,000	3	1	1,100,000

TAB (3):

🚗 Car Selling Price Prediction & Analysis

Predict Price | View Dataset | Model Insights

Model Performance

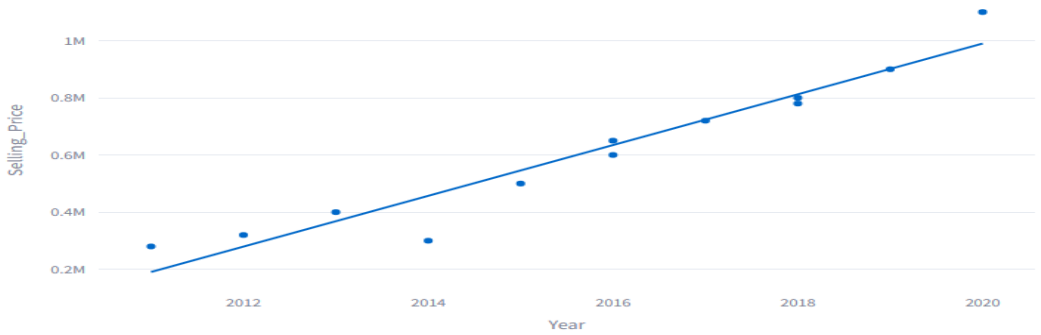
Mean Absolute Error
$76,906.68

R^2 Score
61.64%

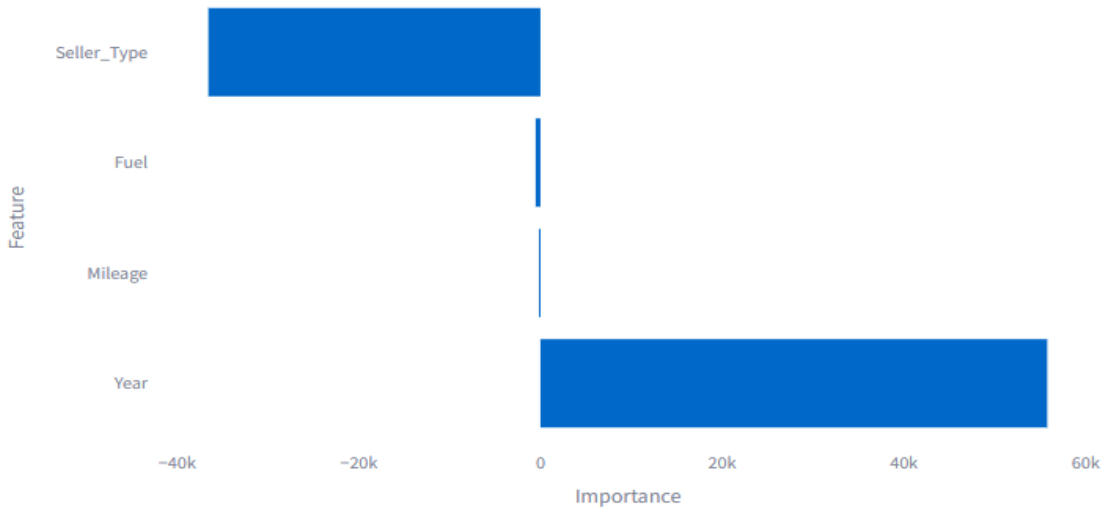
Feature Relationships

Select feature to plot

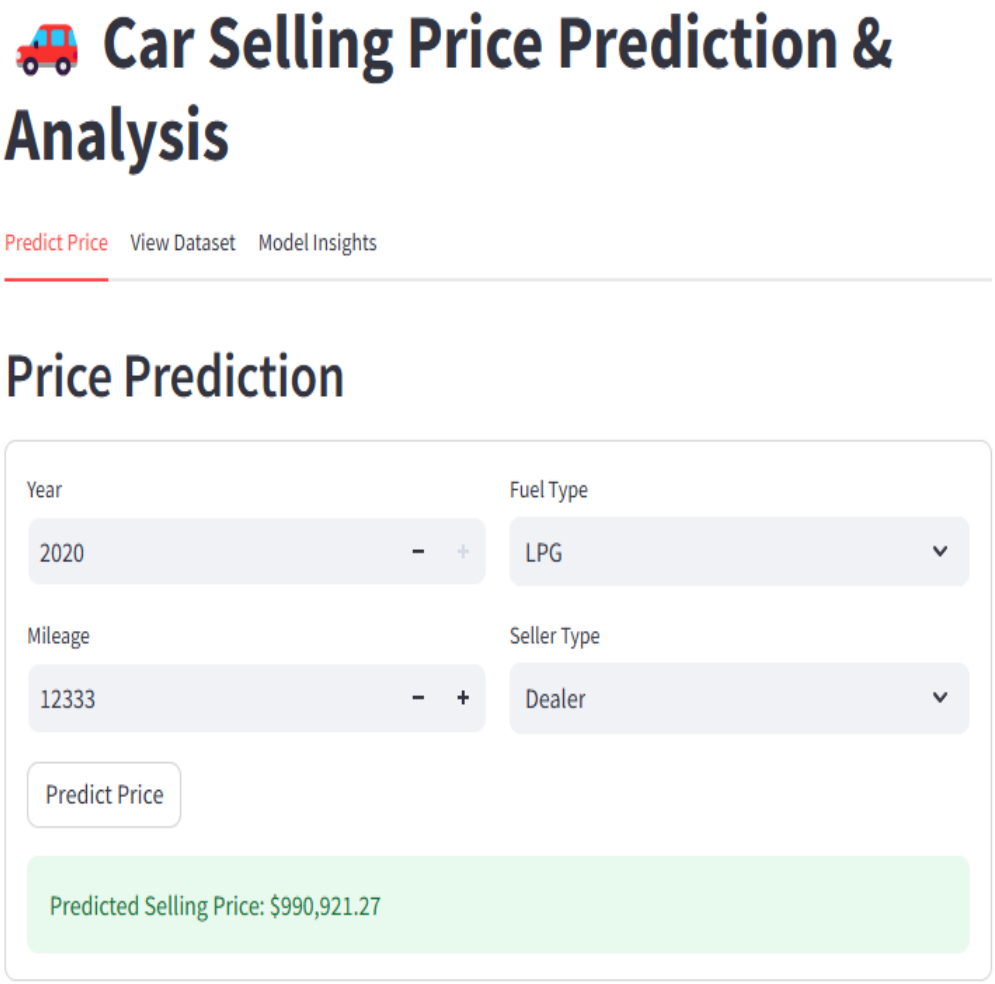
Year



Feature Importance



TAB (1) : WE CAN CHANGE SOME DETAILS SO THAT MODEL WILL PREDICT THE CAR PRICE WITH THE HELP DATASET.



DATASET:

car_data

	A	B	C	D	E
1	Year	Mileage	Fuel	Seller_Type	Selling_Price
2	2015	45000	Petrol	Dealer	500000
3	2012	78000	Diesel	Individual	320000
4	2018	22000	Petrol	Dealer	800000
5	2016	55000	Diesel	Dealer	650000
6	2019	18000	CNG	Dealer	900000
7	2013	68000	Petrol	Individual	400000
8	2017	35000	Diesel	Dealer	720000
9	2014	74000	LPG	Individual	300000
10	2020	12000	Petrol	Dealer	1100000
11	2011	88000	Diesel	Individual	280000
12	2016	49000	CNG	Dealer	600000
13	2018	21000	Petrol	Dealer	780000
14					
15					