Project : Car Poland Price Prediction Using SVM/Decision Tree

Introduction and Overview:

1 - Project idea and overview.

The aim of this project is to develop a machine learning model that predicts the prices of used cars in Poland. Car prices are influenced by various factors such as brand, model, age, mileage, fuel type, and technical condition. By leveraging historical data on car sales, along with relevant features, we can build a predictive model to estimate the price of a car accurately.

Proposed Solution & Dataset:

1 - Main Functionalities

- The main functionality of our project , Cleaning and preprocessing the dataset to handle missing values, outliers, and inconsistencies, ensuring data quality and reliability for model training.

2 - Dataset

- The dataset contains information about the make, model, generation, year of production, mileage, engine type and volume, localization and price

Dataset Name: Car Prices Poland

Contents

- This dataset is split into two sets:

1. train (80%)
2. test (20%)

Dataset Source:

https://www.kaggle.com/datasets/aleksandrglotov/car-prices-poland

Applied Algorithms:

1-Support Vector Machines (SVMs):

Support Vector Machines are powerful supervised learning algorithms used for classification and regression tasks. SVMs are particularly effective in high-dimensional spaces and are capable of capturing complex relationships in the data.

Key Components of SVM:

1. **Support Vectors:** These are data points that lie closest to the decision boundary (hyperplane) between different classes. Support vectors play a

crucial role in defining the decision boundary and determining the margin, which is the distance between the hyperplane and the closest data points.

2. **Hyperplane:** In SVM, the decision boundary that separates different classes is called a hyperplane. For binary classification, the hyperplane aims to maximize the margin, which is the distance between the hyperplane and the nearest data points of each class. SVM finds the optimal hyperplane that best separates the classes while maximizing the margin.
3. **Kernel Trick:** SVMs can efficiently handle non-linear classification tasks by implicitly mapping the input features into a higher-dimensional space using kernel functions. The kernel trick allows SVM to operate in a higher-dimensional space without explicitly computing the transformed feature vectors, thereby avoiding the computational burden.

Types of SVM Kernels:

SVMs support various kernel functions, each of which maps the input features into a higher-dimensional space, enabling SVM to capture complex relationships in the data. The choice of kernel function significantly impacts the model's ability to separate classes effectively. Here are the common types of SVM kernels:

1. **Linear Kernel:**
    - The linear kernel is the simplest form of kernel function.
    - It represents the dot product between the feature vectors in the original feature space.
    - The linear kernel is suitable for linearly separable data, where classes can be separated by a straight line or hyperplane.
2. **Radial Basis Function (RBF) Kernel:**
    - The RBF kernel, also known as the Gaussian kernel, is widely used in SVMs for handling non-linear data.
    - It maps the input features into an infinite-dimensional space using a Gaussian radial basis function.

- The RBF kernel is flexible and capable of capturing complex decision boundaries, making it suitable for a wide range of classification tasks.

3. **Polynomial Kernel:**
    - The polynomial kernel maps the input features into a higher-dimensional space using polynomial functions.
    - It is suitable for data that exhibits non-linear relationships and can capture more complex decision boundaries compared to the linear kernel.
    - The degree parameter of the polynomial kernel controls the degree of the polynomial transformation.

SVM models aim to find the optimal hyperplane that best separates the classes in the input feature space. By maximizing the margin between different classes, SVMs achieve robust generalization performance and resistance to overfitting. SVM models effectively handle both linearly separable and non-linearly separable data by leveraging kernel functions to map the input features into higher-dimensional spaces. SVMs are widely used for classification, regression, and outlier detection tasks in various domains, including finance, healthcare, text classification, and image recognition.

Support Vector Regression (SVR)

It is an **extension of Support Vector Machines (SVMs)** that are typically used for classification. SVR is designed to predict continuous outcomes, making it suitable for tasks where the target variable is a continuous value rather than a class label.

the key components

1. **Objective:**
    - The primary objective of SVR is to build a regression model that predicts the value of a continuous target variable based on input features.
    - SVR aims to find the optimal hyperplane (or regression function) that best fits the training data while minimizing the prediction error.
2. **Margin and Support Vectors:**

- Similar to SVMs for classification, SVR involves the concept of a margin, which represents the distance between the regression function and the data points.
- Support vectors are the data points that lie within a certain margin of the regression function. These support vectors play a crucial role in determining the optimal regression function.

3. **Kernel Trick:**
    - SVR leverages the kernel trick to map the input features into a higher-dimensional space, where non-linear relationships between the features and the target variable can be captured.
    - Common kernel functions used in SVR include linear, radial basis function (RBF), and polynomial kernels.
4. **Loss Function:**
    - SVR uses a loss function to measure the error between the predicted and actual target values.
    - The loss function typically includes a term that penalizes deviations from the regression function while allowing for a certain margin of error.
5. **Regularization:**
    - Regularization techniques, such as L1 or L2 regularization, may be applied to SVR to prevent overfitting and improve generalization performance.
    - Regularization helps control the complexity of the regression function and ensures that the model generalizes well to unseen data.
6. **Hyperparameters:**
    - SVR includes hyperparameters such as the regularization parameter (C), the kernel type, and kernel-specific parameters (1-gamma for RBF kernel,2-degree for polynomial kernel).
    1. **Gamma (γ) Parameter:**
        - Gamma is a parameter used in kernel functions, particularly in the Radial Basis Function (RBF) kernel.

- In the context of the RBF kernel, gamma defines the influence of a single training example, with low values meaning 'far' and high values meaning 'close'.
- A small gamma value implies a large similarity radius, leading to a smooth decision boundary, whereas a large gamma value results in a tighter decision boundary, potentially leading to overfitting.
- High gamma values can cause the model to focus more on individual data points, potentially leading to overfitting if not appropriately tuned.
- Tuning the gamma parameter is crucial to balancing the trade-off between model complexity and generalization performance.

> **"auto":**
>
> - When "auto" is specified for the gamma parameter, the algorithm automatically calculates the gamma value based on a heuristic that takes into account the inverse of the number of features in the input data.
> - The "auto" option is useful when you want the algorithm to automatically adjust the gamma value based on the characteristics of the dataset without explicitly specifying it.
> - This option is often used as a default setting, especially when the dataset has a large number of features, as it helps prevent overfitting by adjusting the scale of the RBF kernel appropriately.

> **"scale":**
>
> - When "scale" is specified for the gamma parameter, the algorithm scales the gamma value inversely

proportional to the standard deviation of the input features.

- This option is particularly useful when the input features have varying scales, as it ensures that the gamma value is adjusted accordingly to maintain consistency in the kernel computation across features.
- By scaling the gamma value based on the standard deviation of the input features, the "scale" option helps prevent certain features from dominating the kernel calculation, thus improving the overall stability and performance of the SVM model.

2. **Degree Parameter:**
    - Degree is a parameter used in kernel functions, particularly in the Polynomial kernel.
    - In the context of the Polynomial kernel, the degree parameter determines the degree of the polynomial used in the kernel function.
    - For example, a degree of 2 indicates a quadratic kernel, while a degree of 3 indicates a cubic kernel, and so on.
    - Higher degree values allow the model to capture more complex relationships between the input features and the target variable.
    - However, increasing the degree can also lead to increased model complexity and potentially overfitting, especially with high-degree polynomials.
    - Like other hyperparameters, tuning the degree parameter is essential to find the optimal balance between model complexity and generalization performance.

- Tuning these hyperparameters is essential for optimizing the performance of the SVR model.

7. **Training and Prediction:**
   - During the training phase, SVR learns the optimal regression function from the training data by minimizing the loss function.
   - Once trained, the SVR model can make predictions on new data by applying the learned regression function to the input features.
   - we used Cross-validation scores (5 Folds) are calculated for each type of kernel in SVM model.

1. **Applications:**
   - SVR is commonly used in various domains, including finance, engineering, healthcare, and environmental science, for tasks such as stock price prediction, time series forecasting, and modeling physical phenomena.

Decision Tree Regressor (DT):

1. **Objective:**
   - Decision Tree Regressor is a non-parametric supervised learning algorithm used for regression tasks.
   - It aims to build a regression model that predicts the target variable based on the input features by recursively partitioning the feature space into disjoint regions.
2. **Model Structure:**
   - The decision tree consists of a tree-like structure where each internal node represents a feature/attribute, each branch represents a decision rule based on that feature, and each leaf node represents the predicted value.
   - During training, the decision tree algorithm recursively splits the feature space into smaller regions based on the feature values,

minimizing the variance of the target variable within each region, The cross-validation score (5 folds) is calculated.

3. **Splitting Criteria:**
    - Decision trees use various criteria to decide how to split the data at each node, such as mean squared error (MSE), mean absolute error (MAE), or variance reduction.
    - The splitting criteria aim to maximize the homogeneity (or minimize the impurity) of the target variable within each split.
4. **Advantages:**
    - Decision trees are interpretable and easy to visualize, making them suitable for understanding the underlying decision-making process.
    - They can handle both numerical and categorical features without requiring feature scaling.
    - Decision trees implicitly perform feature selection by selecting the most informative features at each split.
5. **Disadvantages:**
    - Decision trees are prone to overfitting, especially when the tree depth is not controlled.
    - They may produce biased predictions if the training data is imbalanced or noisy.
    - Decision trees are sensitive to small variations in the data, leading to high variance models.

Gradient Boosting Regressor (GB):

Objective:

- Gradient Boosting Regressor is an ensemble learning technique that combines multiple weak learners (typically decision trees) sequentially to improve predictive performance.

- It aims to build a strong regression model by iteratively fitting new models to the residuals (errors) of the previous models.

Model Structure:

- Gradient Boosting Regressor builds an additive model, where each new model (weak learner) is trained to correct the errors made by the existing ensemble.
- The final prediction is obtained by summing the predictions of all individual models in the ensemble.
- The cross-validation score (5 folds) is calculated.

Learning Process:

- Gradient Boosting Regressor employs a gradient descent optimization algorithm to minimize a loss function (e.g., mean squared error) with respect to the residuals.
- At each iteration, a new weak learner is trained to minimize the loss function by fitting the negative gradient of the loss function (i.e., the residual).

Advantages:

- Gradient Boosting Regressor typically yields higher predictive performance compared to individual decision trees.
- It can capture complex relationships in the data and handle non-linearities effectively.
- Gradient Boosting Regressor is less prone to overfitting compared to single decision trees, especially when appropriate regularization techniques are applied.

Disadvantages:

- Gradient Boosting Regressor may be computationally expensive and require tuning of hyperparameters such as learning rate, tree depth,

and number of estimators.

- It may suffer from longer training times compared to simpler models due to the sequential nature of the learning process.

Model Training

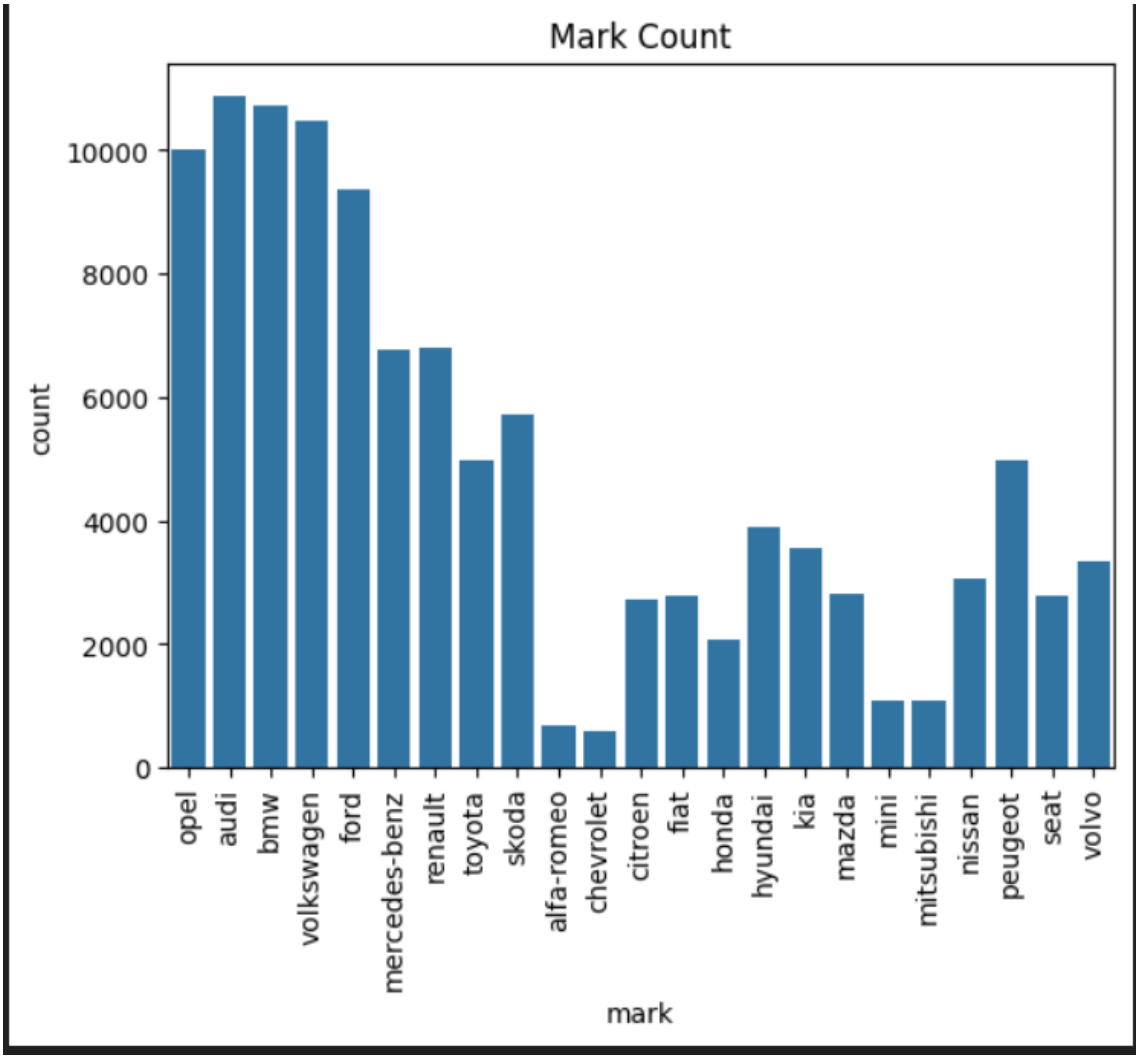
- We fit the model with our the training set to start the training phase .

Experiments & Results:

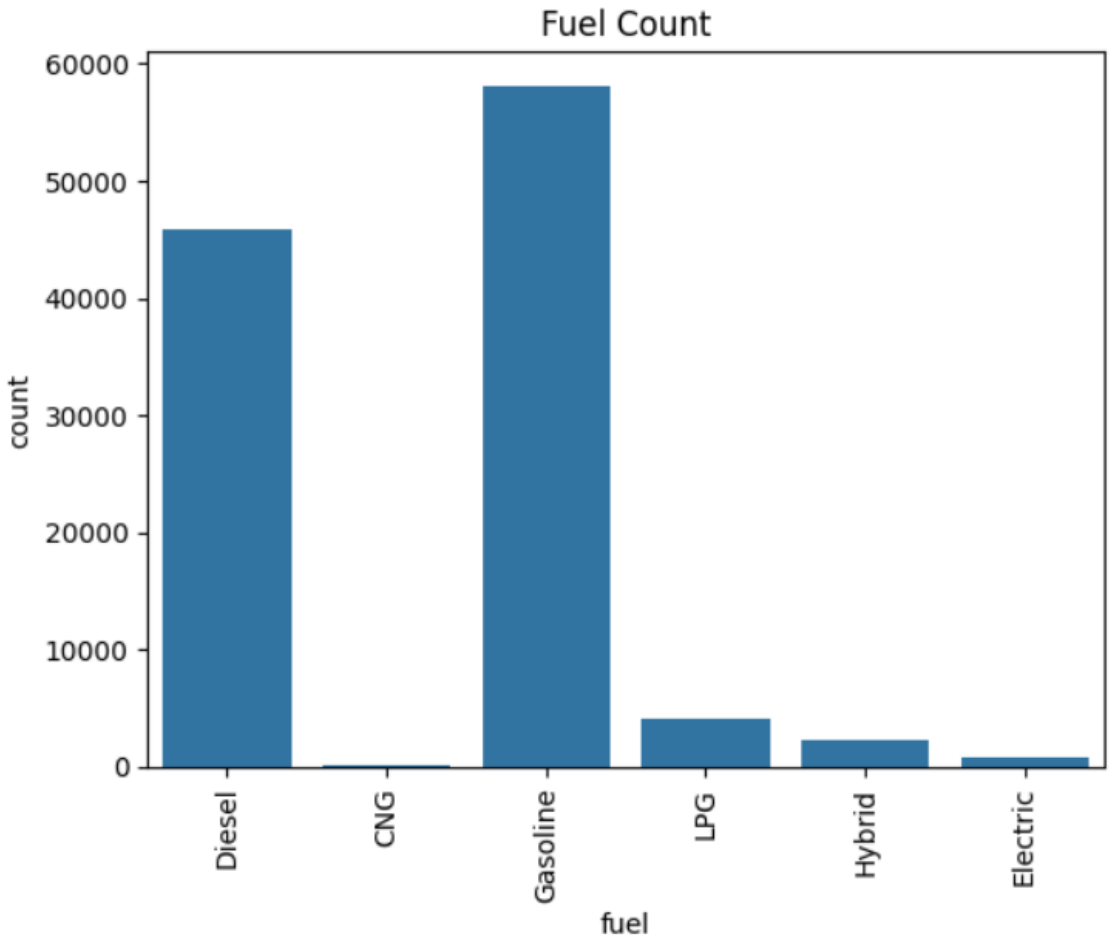
Experiments

Analysis Plots

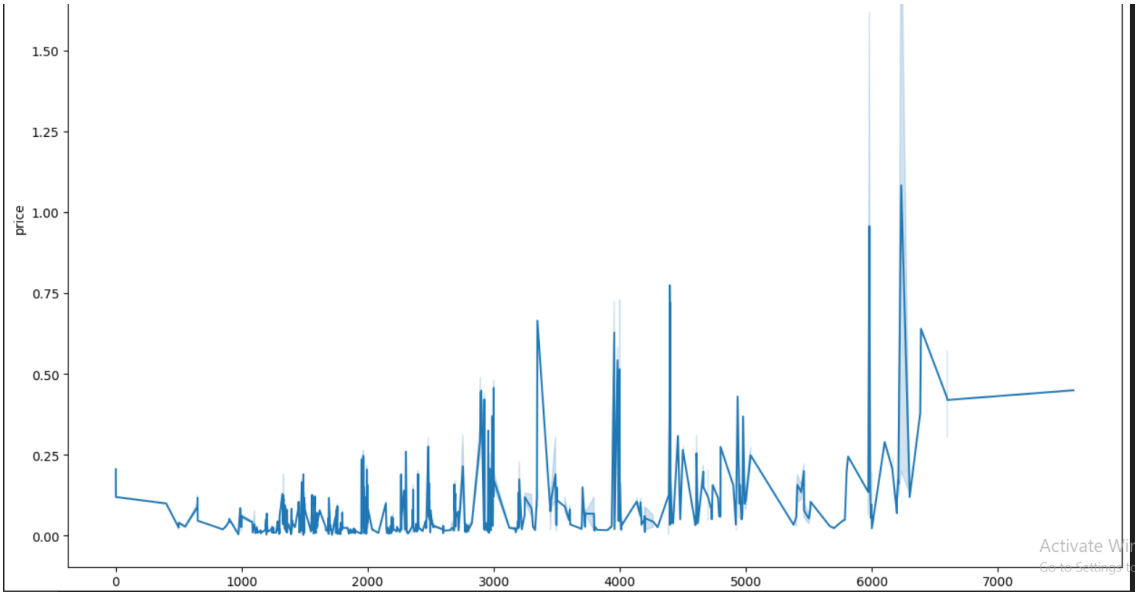
1-Value count plot for each mark



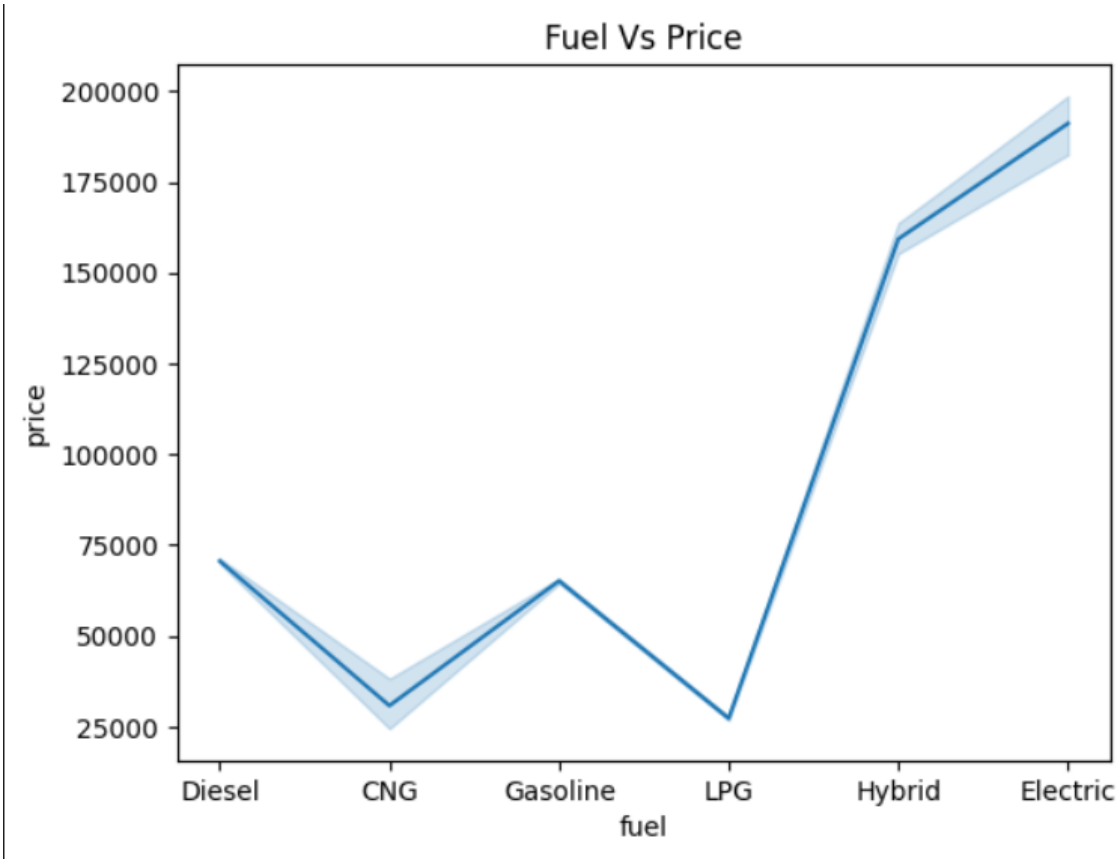
2-Value count of each fuel



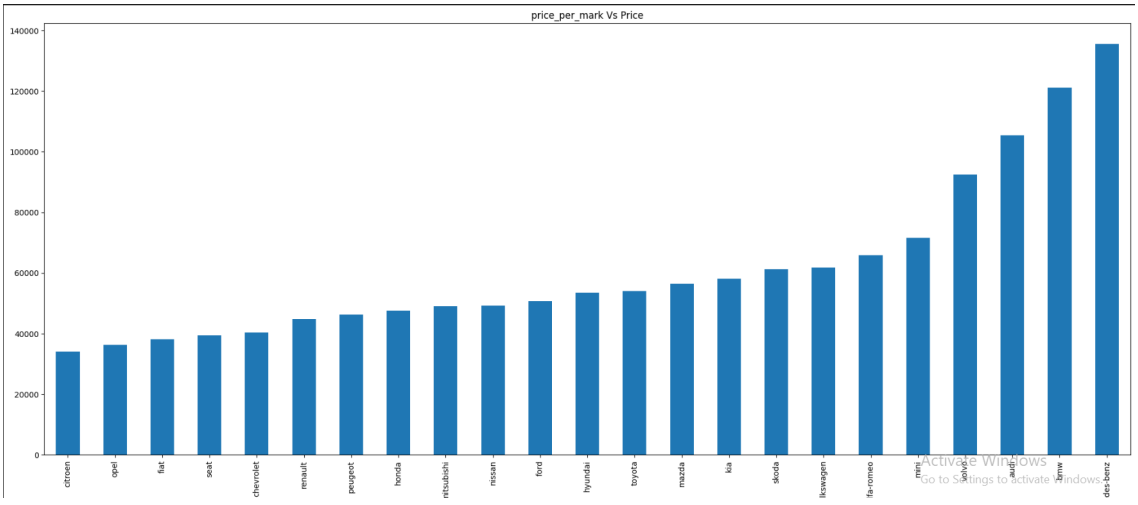
3-Lineplot for volume_engine VS price



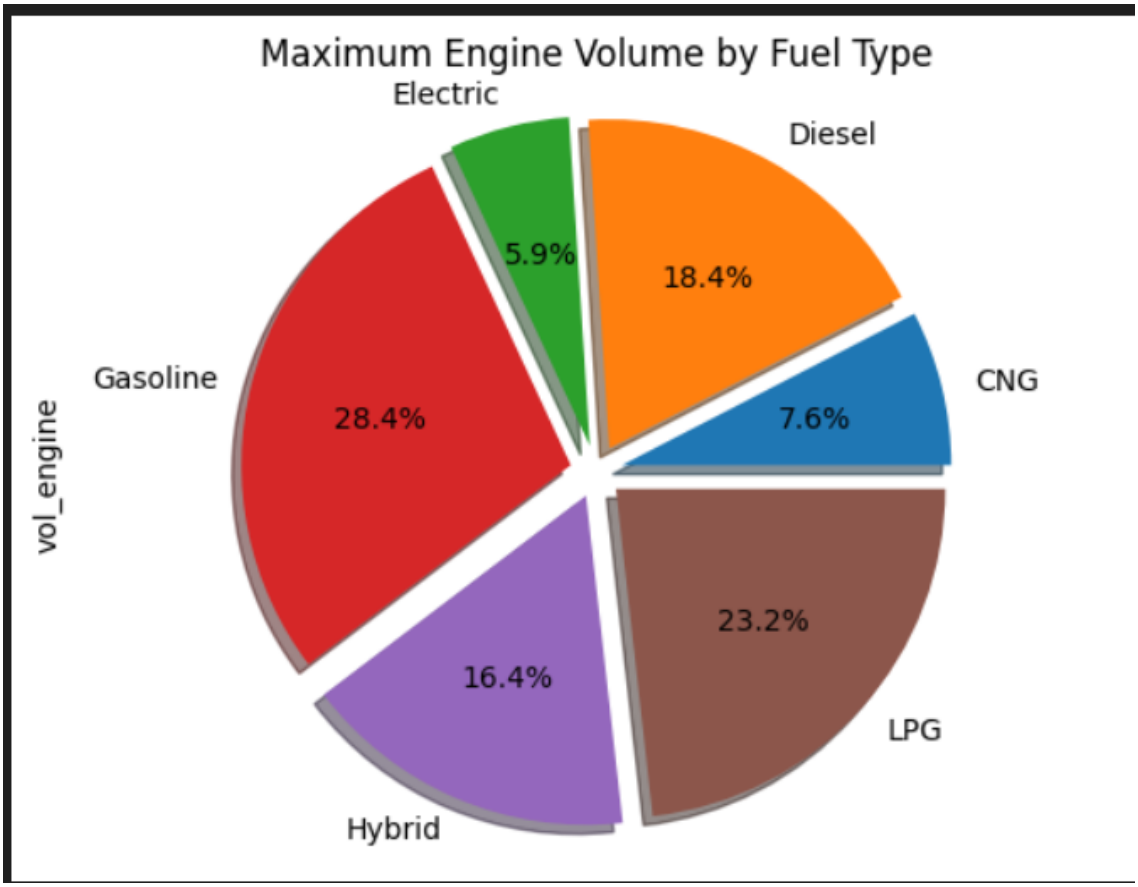
4-lineplot for fuel VS price



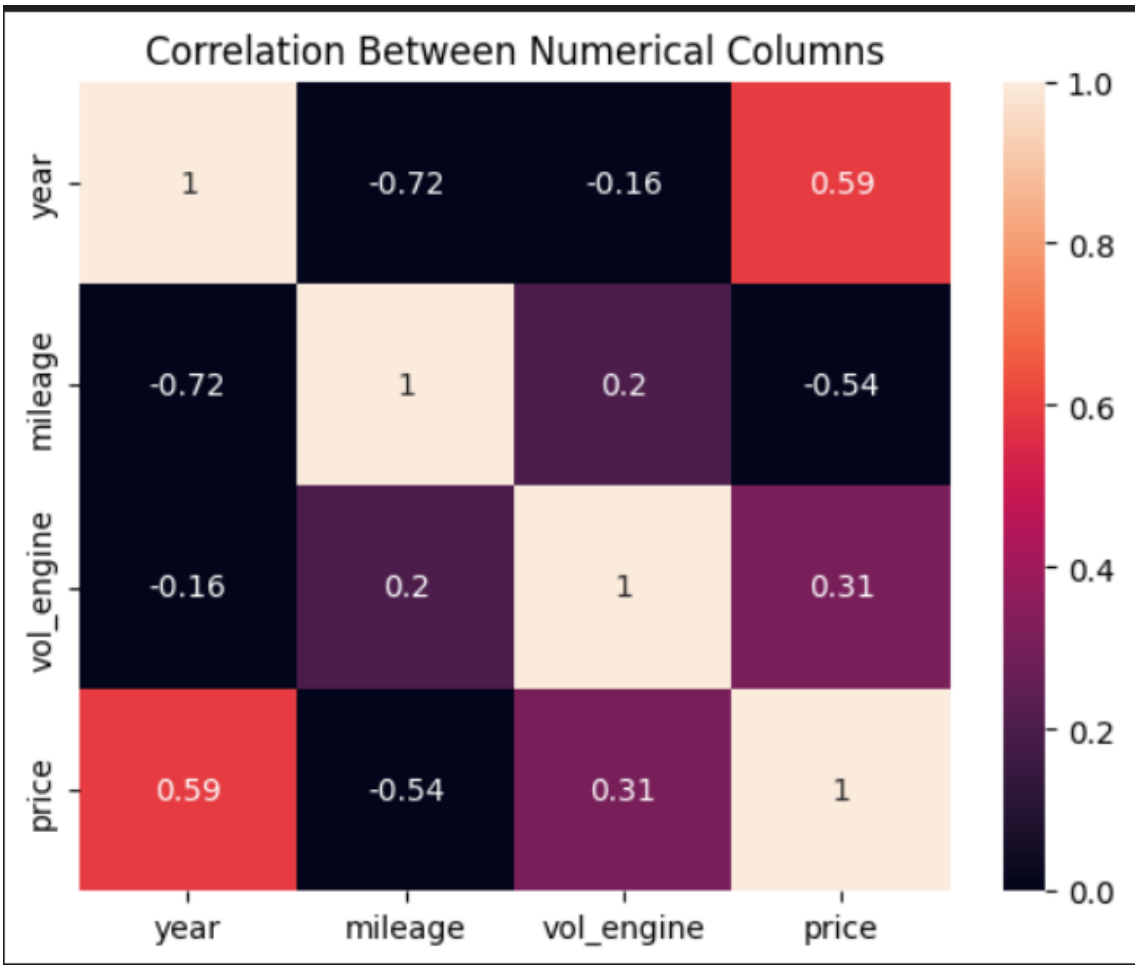
5-barplot for average price of each mark VS price



6-Maximum Engine Volume by Fuel Type

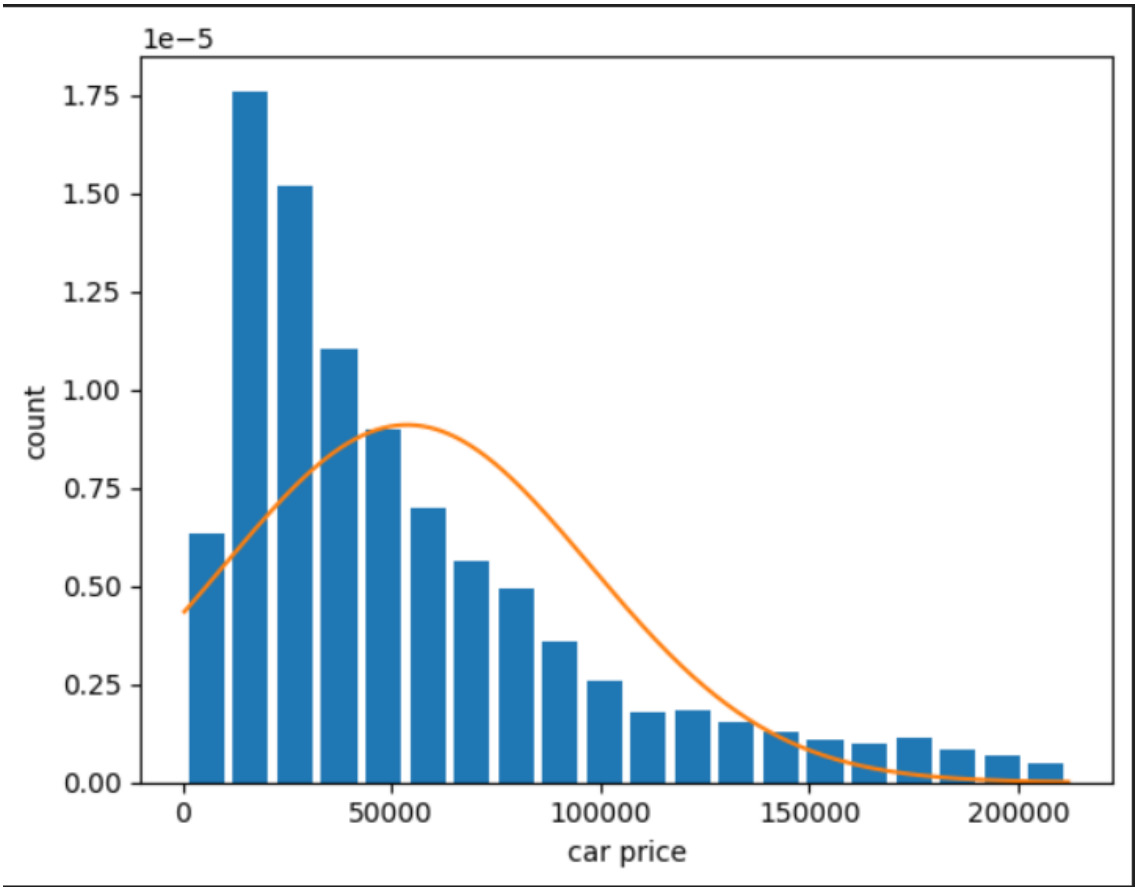


7-Correlation Between Numerical Columns

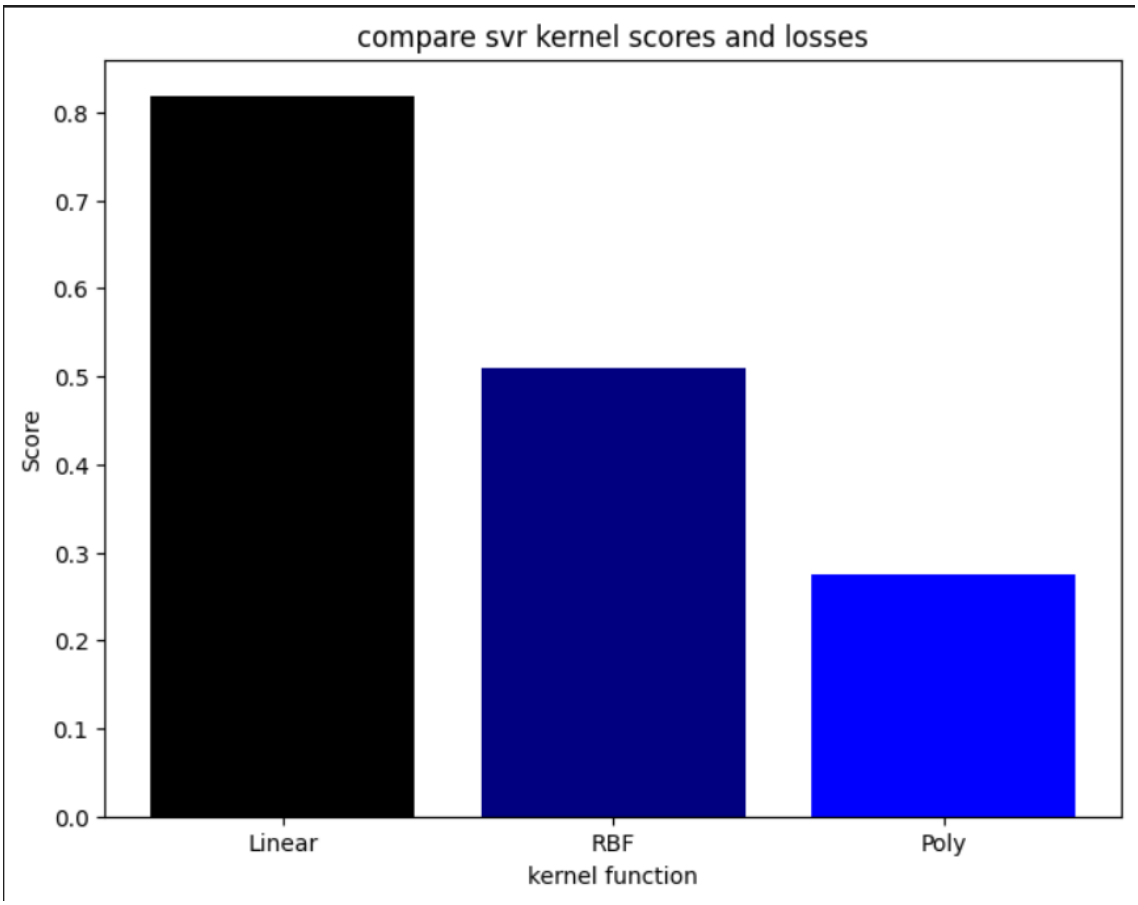


Results:

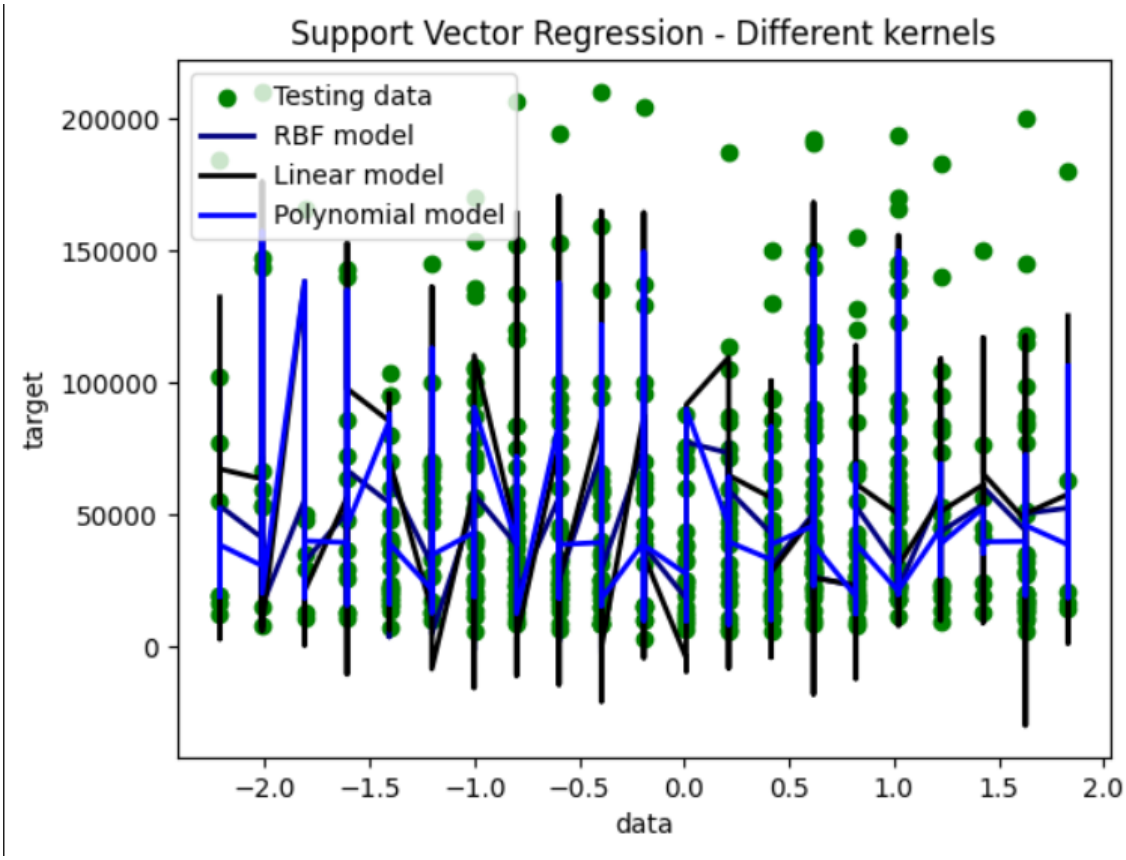
1- Distribution of Car Prices After Removing Outliers



2-Compare svr kernel scores and losses



3-Support Vector Regression - Different kernels



Analysis of the results

- **Accuracy and Performance:**

  model accuracy (SVM):

  a)Linear_svr → 82%

  b)rbf_svr → 52%

  c)ploy_svr → 27%

- model accuracy (Decision Tree): 85.4%
- model accuracy (gradient Boosting): 90%

Shared Folder

https://rb.gy/stuvws

https://drive.google.com/drive/folders/1a6d1k2N-aU-jHbP3vM2m7ULdOij51Wpe

https://github.com/marwanali213/Automated-object-detection-using-artificial-neural-network-/tree/main