Title :

Feature Finder

Subtitle:

Feature Finder is a cutting-edge machine learning tool engineered to pinpoint and prioritize key features from extensive datasets. Utilizing sophisticated algorithms, it refines the feature selection process, boosting both the efficiency and clarity of predictive models.

Enroll. No :

202418011 , 202418009 , 202418042 , 202418056

Date : **26/08/2004**

Introduction

Objective:

Feature Finder is a machine learning tool designed to optimize the estimation of maintenance costs. By leveraging advanced algorithms to identify and prioritize crucial features from extensive datasets, it enhances the accuracy of cost predictions.

Project Overview

Dataset:

Total Instances : 10,000

Attributes :

1. **Car Make** – The manufacturer of the car (object)
2. **Car Model** – The specific model of the car (object)
3. **Year of Manufacture** – The year the car was made (int64)
4. **Engine Size** – The size of the car's engine (int64)
5. **Mileage** – The total distance the car has traveled (int64)
6. **Maintenance History** – Record of past maintenance activities (object)
7. **Maintenance Cost** – The cost incurred for maintenance (int64)
8. **Repair Costs** – Expenses related to repairs (int64)
9. **Insurance Premium** – Cost of insurance for the car (int64)
10. **Coverage Type** – Type of insurance coverage (object)
11. **Location** – Geographical location of the vehicle (object)
12. **Driver Age** – Age of the car's driver (int64)
13. **Driver Experience** – Years of experience of the driver (int64)
14. **Safety Feature** – Features related to vehicle safety (object)
15. **Vehicle Condition** – Current condition of the vehicle (object)

Data Types:

- **Numeric (int64)**: 8 attributes
- **Categorical (object)**: 7 attributes

Model:

Feature Finder utilizes three powerful regression models to estimate maintenance costs:

1. **Linear Regression:** An ensemble method that combines multiple decision trees to improve accuracy and handle complex data patterns.

2. **Ridge Regression:** A technique that applies L2 regularization to address multicollinearity and enhance model robustness.

3. **Lasso Regression:** A method incorporating L1 regularization to perform feature selection and promote sparse, interpretable models.

Metrics :

- MSE provides an idea of the average squared error, useful for understanding the variance of errors.
- R^2 reflects how well the model explains the variability of the outcome.
- MAE offers a simple interpretation of average prediction error.
- RMSE quantifies the average prediction error by taking the square root of the mean squared differences between predicted and actual values.

Data Preparation

Preprocessing Steps:

Outline key preprocessing tasks with brief descriptions.

1. **Cleaning:** Removed missing values.
2. **Normalization:** convert objective data to categorical and then apply transformation which is scaling the data into [0,1].
3. **Data Transformation**: Raw data is converted into a format suitable for analysis or modeling.
4. **Standardization :** Standardizes features by transforming them to have zero mean and unit variance.

Model Training and Evaluation

❖ Linear Regression Model

Model Training :

1. **Data Splitting:**
   - Training Set: 80%
   - Temporary Set: 20% (further split into Validation Set: 10% and Test Set: 10%)
2. **Feature Selection:**
   - Use Recursive Feature Elimination (RFE) to select key features.
3. **Standardization:**
   - Normalize numeric features to mean 0 and standard deviation 1.
4. **Training the Model:**
   - Fit a Linear Regression model using the training set.

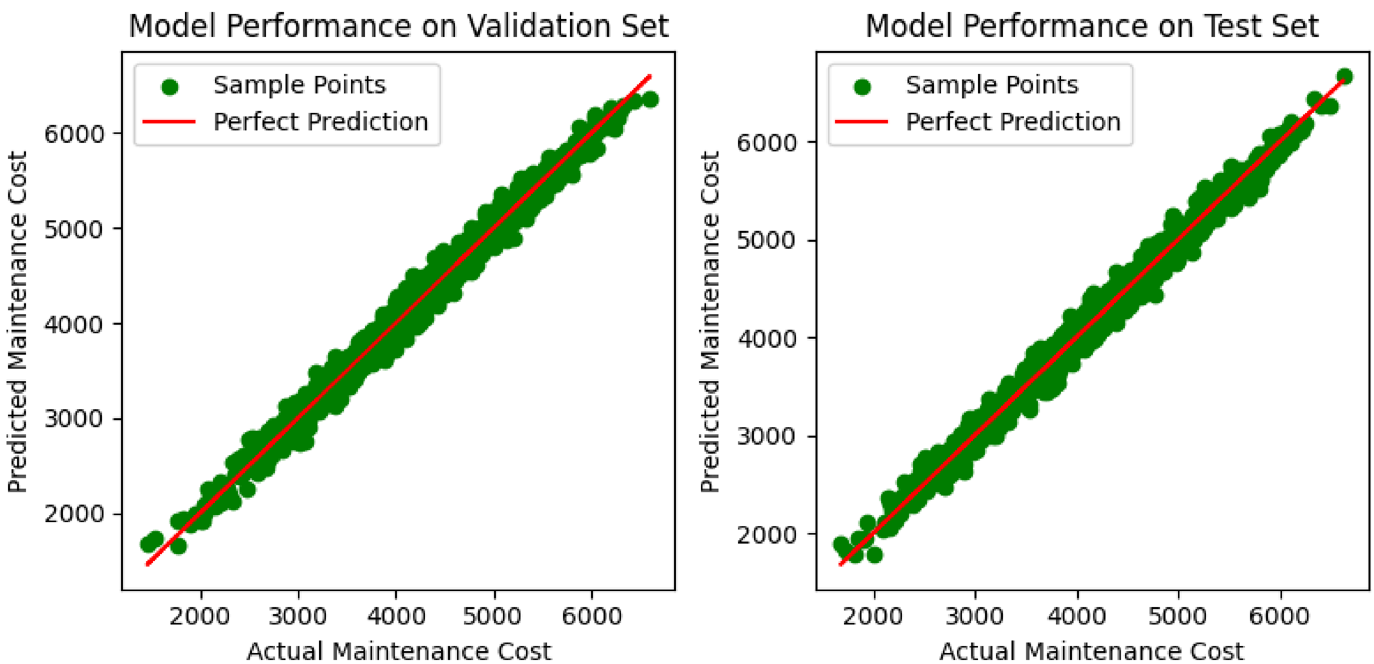
Model Evaluation

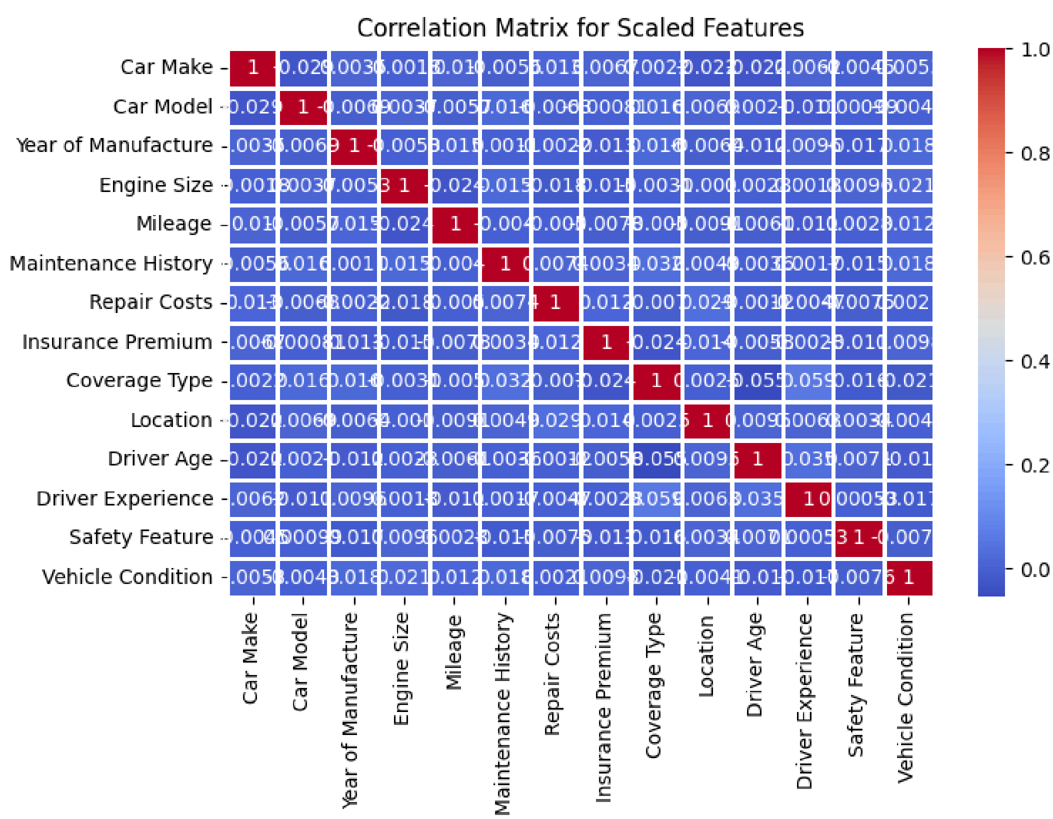
Performance Metrics

Metric	Validation Set	Test Set
Cross MSE	0.01	0.01
MSE	0.01	0.01
R^2	0.99	0.99
MAE	0.09	0.09
RMSE	0.011	0.011
Explained Variance Score	0.99	0.99

Evaluation Insights

- **Cross MSE:** Indicates the consistency of the model across different subsets of data.
- **R^2 Score:** Demonstrates the high explanatory power of the model, nearing perfection.
- **MAE and RMSE:** Provide insights into average prediction errors, with low values indicating excellent performance.


Model Performance on Validation Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost
Model Performance on Test Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost


Correlation Matrix for Scaled Features
Car Make
Car Model
Year of Manufacture
Engine Size
Mileage
Maintenance History
Repair Costs
Insurance Premium
Coverage Type
Location
Driver Age
Driver Experience
Safety Feature
Vehicle Condition

Pseudocode:

START

```
# IMPORT libraries
IMPORT pandas, numpy
IMPORT LinearRegression, Ridge, Lasso, RandomForestRegressor FROM sklearn.linear_model
IMPORT train_test_split, GridSearchCV FROM sklearn.model_selection
IMPORT PolynomialFeatures, StandardScaler, LabelEncoder FROM sklearn.preprocessing
IMPORT mean_squared_error, r2_score, mean_absolute_error FROM sklearn.metrics
IMPORT Pipeline FROM sklearn.pipeline
IMPORT matplotlib.pyplot AS plt
IMPORT seaborn AS sns

# LOAD data
df = pd.read_excel("path/to/car_dataset.xlsx")
X, y = df EXCEPT "Maintenance Cost", df["Maintenance Cost"]

# ENCODE and SCALE
FOR EACH cat_col IN X.select_dtypes('object'):
    X[cat_col] = LabelEncoder().fit_transform(X[cat_col])
X_scaled = StandardScaler().fit_transform(X)
y_scaled = StandardScaler().fit_transform(y.values.reshape(-1, 1))

# SPLIT data
X_train, X_temp, y_train, y_temp = train_test_split(X_scaled, y_scaled, 0.2)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, 0.5)

# SET UP and TRAIN model
pipeline = Pipeline([('poly', PolynomialFeatures()), ('model', LinearRegression())])
grid_search = GridSearchCV(pipeline, {'model': [LinearRegression()]}, cv=5)
grid_search.fit(X_train, y_train)

# PREDICT and EVALUATE
y_pred_val = grid_search.predict(X_val)
y_pred_test = grid_search.predict(X_test)
PRINT 'Validation MSE:', -cross_val_score(grid_search, X_val, y_val, cv=5).mean()
PRINT 'Test MSE:', -cross_val_score(grid_search, X_test, y_test, cv=5).mean()

# PREDICT for new car
FUNCTION car_make(car):
    new_data = DataFrame({...})
    FOR EACH cat_col IN new_data:
        new_data[cat_col] = LabelEncoder().fit_transform(new_data[cat_col])
    cost = StandardScaler().inverse_transform(grid_search.best_estimator_.predict(new_data))
    PRINT "Cost for", car, ":", int(cost[0].item())

# VISUALIZE
PLOT actual vs predicted costs
SHOW correlation heatmap
```

END

❖ Ridge Model

Model Training :

1. **Data Splitting:**
    - Training Set: 80%
    - Temporary Set: 20% (further split into Validation Set: 10% and Test Set: 10%)
2. **Feature Selection:**
    - Use Recursive Feature Elimination (RFE) to select key features.
3. **Standardization:**
    - Normalize numeric features to mean 0 and standard deviation 1.
4. **Training the Model:**
    - Fit a Linear Regression model using the training set.

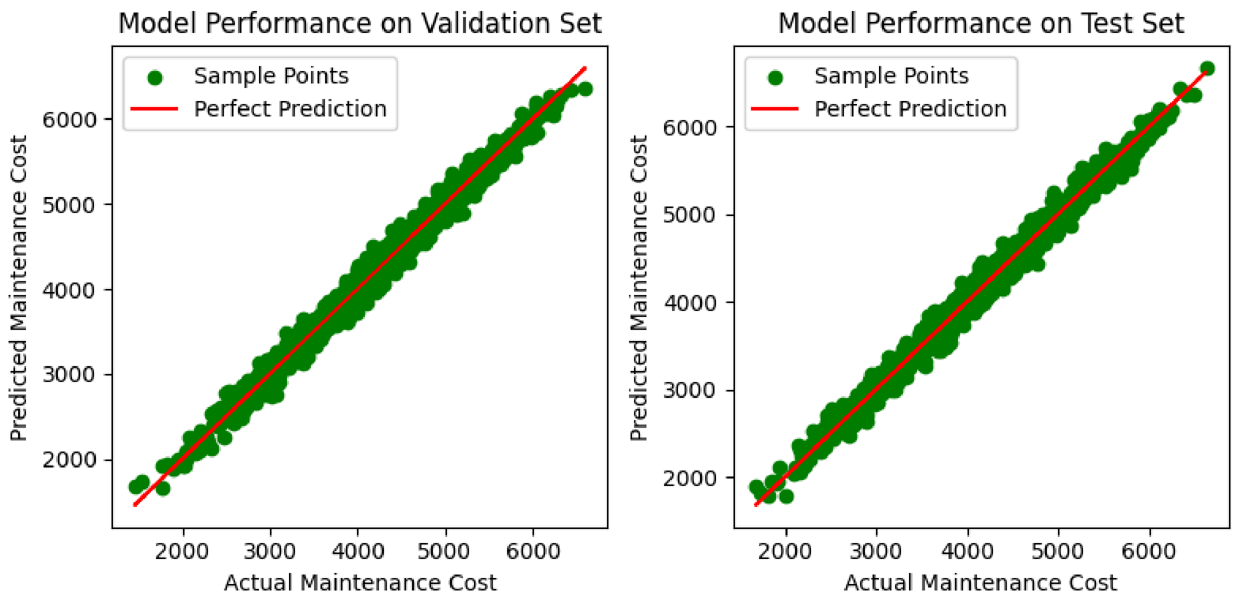
Model Evaluation :

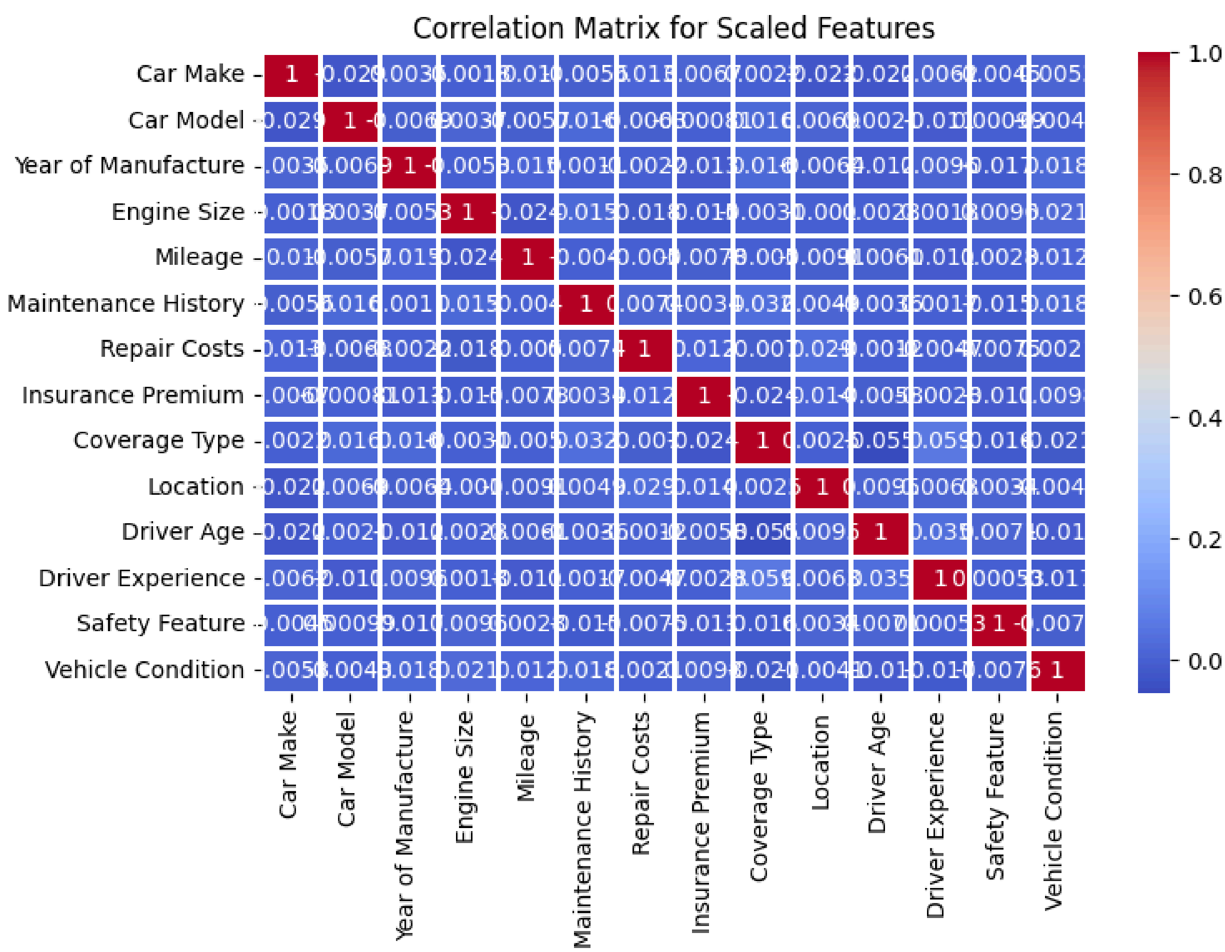
Performance Metrics

Metric	Validation Set	Test Set
Cross MSE	0.01	0.01
MSE	0.01	0.01
R^2	0.99	0.99
MAE	0.09	0.09
RMSE	0.011	0.011
Explained Variance Score	0.99	0.99

Evaluation Insights

- **Cross MSE:** Indicates the consistency of the model across different subsets of data.
- **R^2 Score:** Demonstrates the high explanatory power of the model, nearing perfection.
- **MAE and RMSE:** Provide insights into average prediction errors, with low values indicating excellent performance.


Model Performance on Validation Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost
Model Performance on Test Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost


Correlation Matrix for Scaled Features
Car Make
Car Model
Year of Manufacture
Engine Size
Mileage
Maintenance History
Repair Costs
Insurance Premium
Coverage Type
Location
Driver Age
Driver Experience
Safety Feature
Vehicle Condition

Pseudocode :

```
START

  # IMPORT
IMPORT pandas, numpy
IMPORT train_test_split, GridSearchCV FROM sklearn.model_selection
IMPORT PolynomialFeatures, StandardScaler, LabelEncoder FROM sklearn.preprocessing
IMPORT mean_squared_error, r2_score FROM sklearn.metrics
IMPORT Ridge FROM sklearn.linear_model
IMPORT Pipeline FROM sklearn.pipeline
IMPORT matplotlib.pyplot, seaborn

# LOAD data
df = pd.read_excel("path/to/car_dataset.xlsx")

# DEFINE X, y
X = df EXCEPT "Maintenance Cost"
y = df["Maintenance Cost"]

# ENCODE cat features
FOR col IN categorical columns OF X:
   X[col] = LabelEncoder().fit_transform(X[col])

# STANDARDIZE
X_scaled = StandardScaler().fit_transform(X)
y_scaled = StandardScaler().fit_transform(y.reshape(-1, 1))

# SPLIT
X_train, X_temp, y_train, y_temp = train_test_split(X_scaled, y_scaled, 0.2)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, 0.5)

# PIPELINE
pipeline = Pipeline([('poly', PolynomialFeatures(2)), ('model', Ridge())])

# GRID SEARCH
grid_search = GridSearchCV(pipeline, {'model__alpha': [0.1, 0.5, 1, 5]}, cv=5)
grid_search.fit(X_train, y_train)

# BEST MODEL
model = grid_search.best_estimator_

# PREDICT
y_pred_val = model.predict(X_val)
y_pred_test = model.predict(X_test)

# EVALUATE
PRINT "Validation:", metrics FOR y_val, y_pred_val
PRINT "Test:", metrics FOR y_test, y_pred_test

# car_make
FUNCTION car_make(car):
   new_car = DataFrame WITH car features
   ENCODE new_car
   COST = model.predict(SCALE(new_car))
   PRINT cost

# PREDICT
FOR car IN unique car makes:
   car_make(car)

# VISUALIZE
PLOT actual vs predicted
SHOW heatmap of correlation

END
```

❖ Lasso Model

Model Training :

1. **Data Splitting:**
    - Training Set: 80%
    - Temporary Set: 20% (further split into Validation Set: 10% and Test Set: 10%)
2. **Feature Selection:**
    - Use Recursive Feature Elimination (RFE) to select key features.
3. **Standardization:**
    - Normalize numeric features to mean 0 and standard deviation 1.
4. **Training the Model:**
    - Fit a Linear Regression model using the training set.

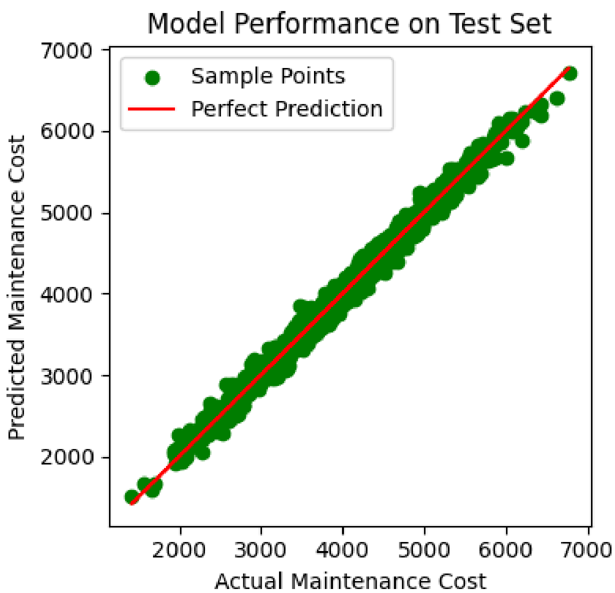
Model Evaluation :

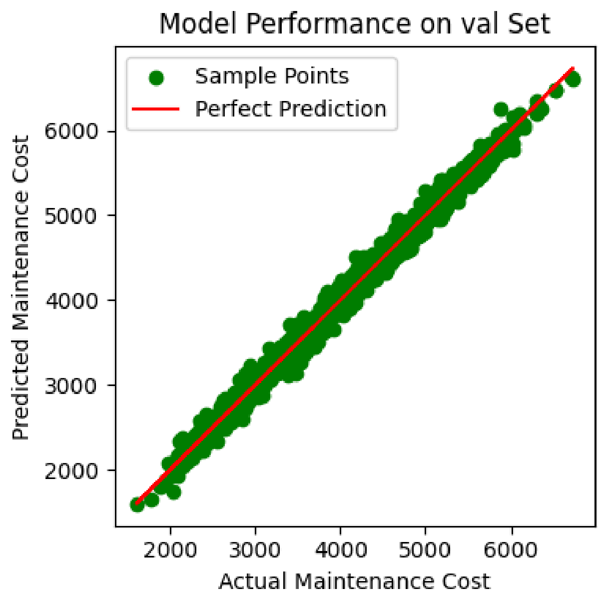
Performance Metrics

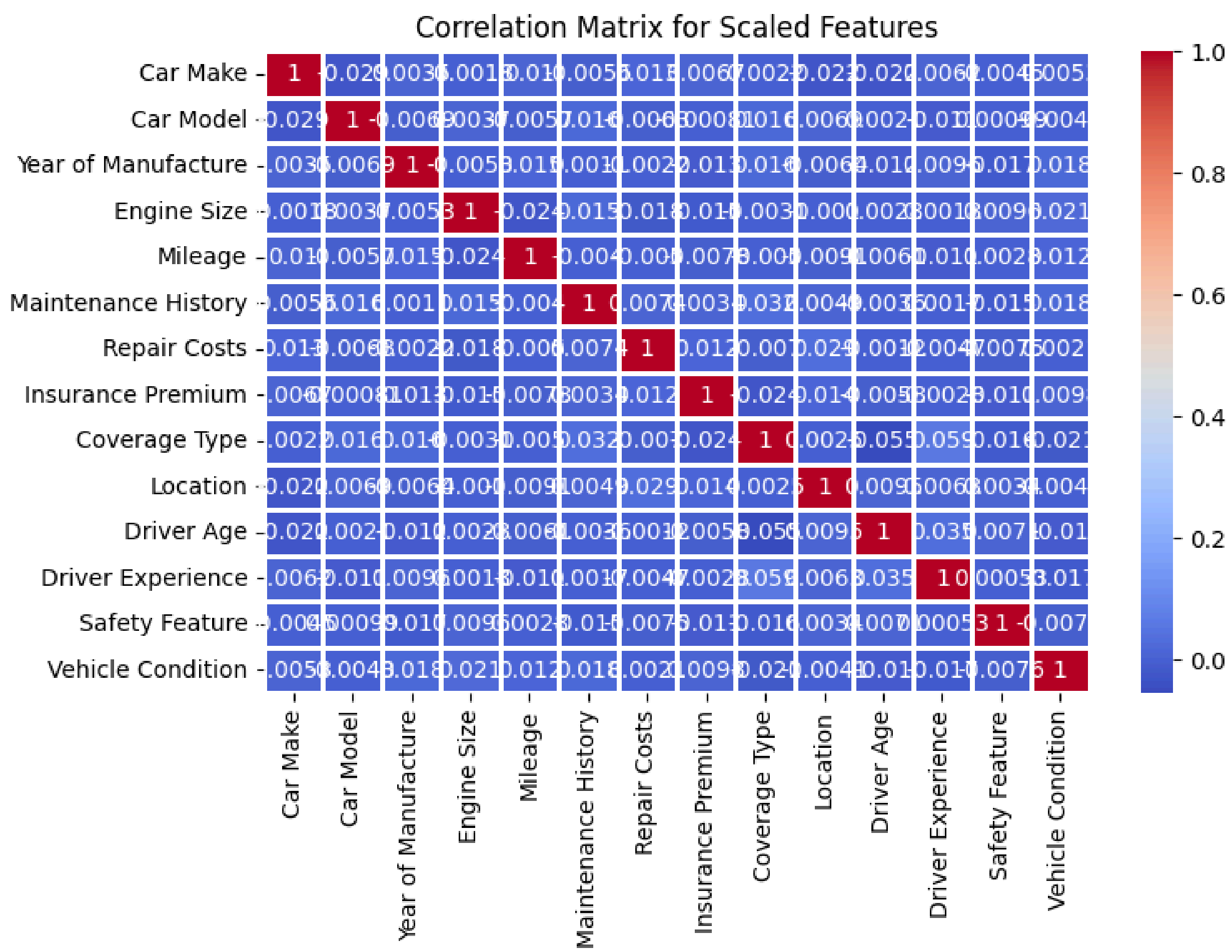
Metric	Validation Set	Test Set
Cross MSE	0.01	0.01
MSE	0.01	0.01
R^2	0.99	0.99
MAE	0.09	0.09
RMSE	0.011	0.011
Explained Variance Score	0.99	0.99

Evaluation Insights

- **Cross MSE:** Indicates the consistency of the model across different subsets of data.
- **R^2 Score:** Demonstrates the high explanatory power of the model, nearing perfection.
- **MAE and RMSE:** Provide insights into average prediction errors, with low values indicating excellent performance.


Model Performance on Test Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost
2000
3000
4000
5000
6000
7000


Model Performance on val Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost
2000
3000
4000
5000
6000


Correlation Matrix for Scaled Features
Car Make
Car Model
Year of Manufacture
Engine Size
Mileage
Maintenance History
Repair Costs
Insurance Premium
Coverage Type
Location
Driver Age
Driver Experience
Safety Feature
Vehicle Condition
1.0
0.8
0.6
0.4
0.2
0.0

Pseudocode :

```
START

   IMPORT pandas, numpy, sklearn, matplotlib, seaborn

   LOAD dataset INTO df
   DEFINE X, y FROM df

   ENCODE categorical columns IN X
   SCALE X AND y

   SPLIT data INTO train, val, test sets

   DEFINE pipeline WITH PolynomialFeatures AND Lasso
   RUN GridSearchCV ON pipeline
   FIT grid_search TO training data

   PREDICT val AND test sets USING best pipeline

   PRINT metrics FOR val AND test sets

   DEFINE car_make(car):
      CREATE new car data
      ENCODE, SCALE, PREDICT cost
      PRINT cost

   FOR EACH unique car IN df['Car Make']:
      CALL car_make(car)

   PRINT car costs

   PLOT validation AND test performance
   PLOT correlation matrix HEATMAP

END
```

- **Elastic Net**

Model Training :

5. **Data Splitting:**
    - Training Set: 80%
    - Temporary Set: 20% (further split into Validation Set: 10% and Test Set: 10%)
6. **Feature Selection:**
    - Use Recursive Feature Elimination (RFE) to select key features.
7. **Standardization:**
    - Normalize numeric features to mean 0 and standard deviation 1.
8. **Training the Model:**
    - Fit a Linear Regression model using the training set.

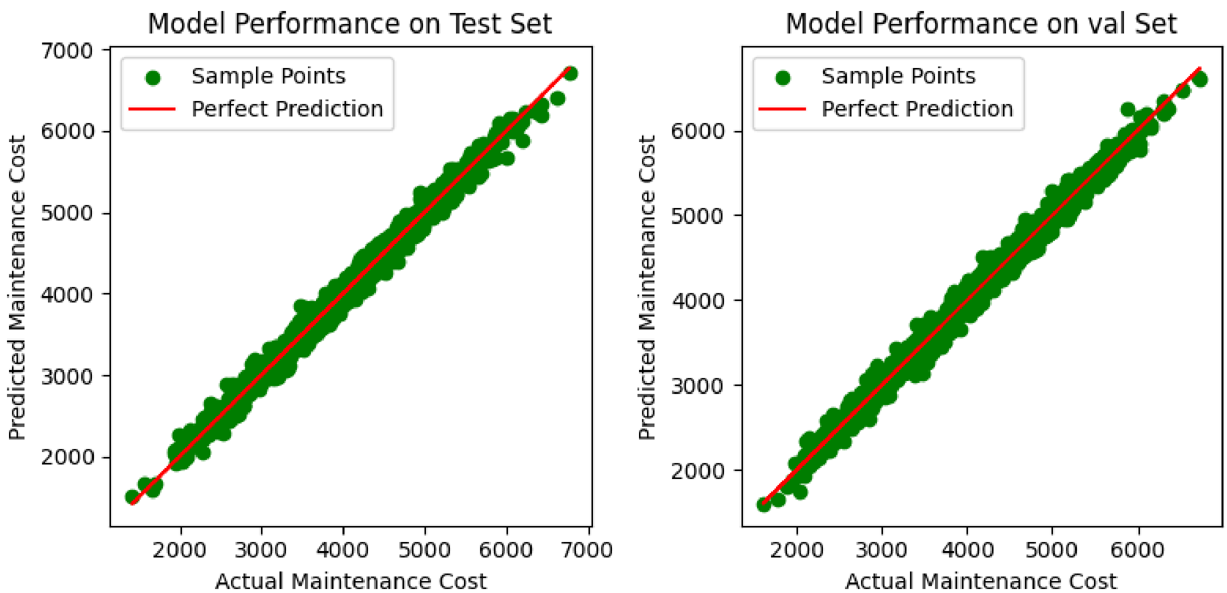
Model Evaluation :

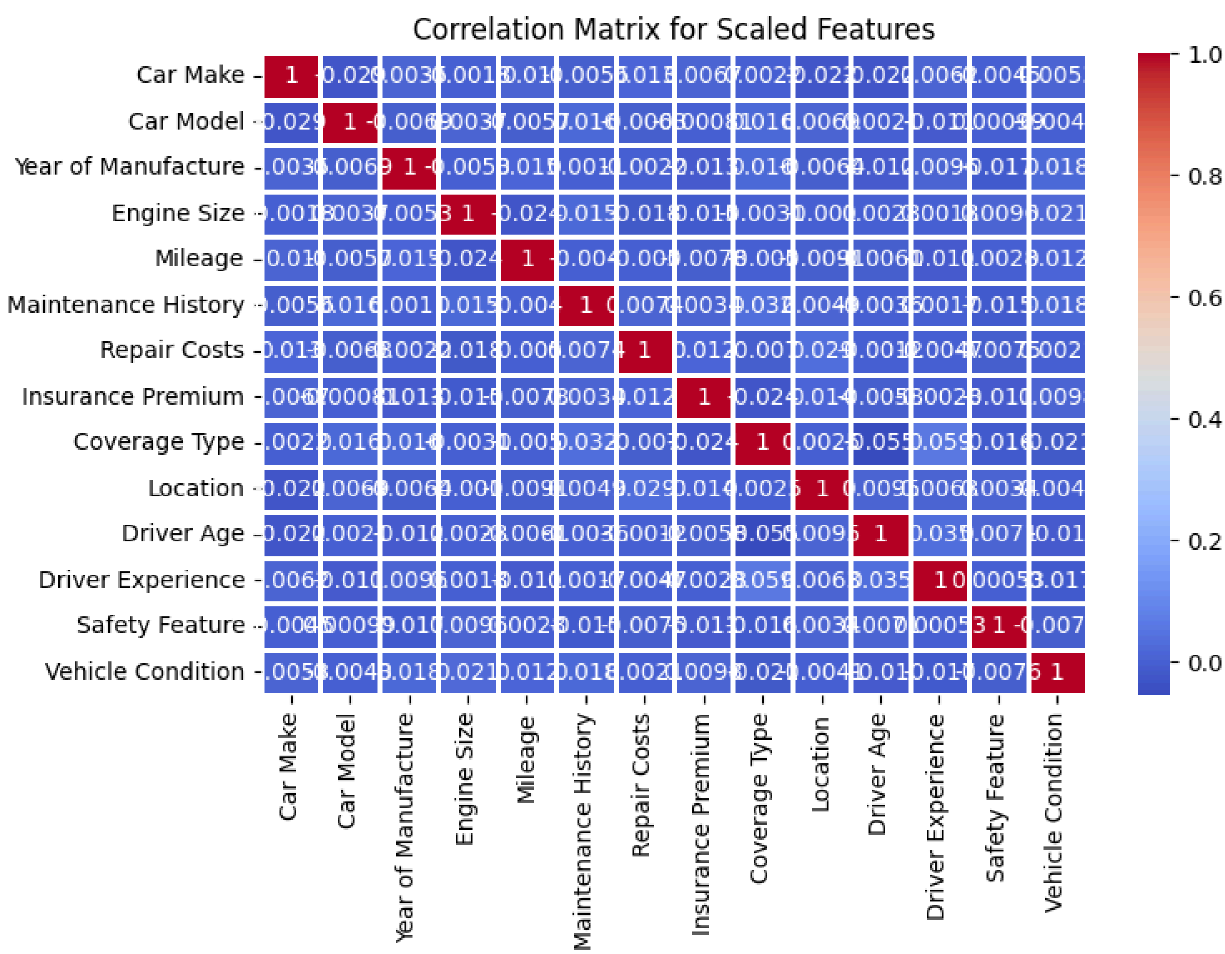
Performance Metrics

Metric	Validation Set	Test Set
Cross MSE	0.01	0.01
MSE	0.01	0.01
R^2	0.99	0.99
MAE	0.09	0.09
RMSE	0.011	0.011
Explained Variance Score	0.99	0.99

Evaluation Insights

- **Cross MSE:** Indicates the consistency of the model across different subsets of data.
- **R^2 Score:** Demonstrates the high explanatory power of the model, nearing perfection.
- **MAE and RMSE:** Provide insights into average prediction errors, with low values indicating excellent performance.


Model Performance on Test Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost
Model Performance on val Set
Sample Points
Perfect Prediction
Predicted Maintenance Cost
Actual Maintenance Cost


Correlation Matrix for Scaled Features
Car Make
Car Model
Year of Manufacture
Engine Size
Mileage
Maintenance History
Repair Costs
Insurance Premium
Coverage Type
Location
Driver Age
Driver Experience
Safety Feature
Vehicle Condition

Pseudocode :

```
START

    IMPORT necessary libraries (pandas, numpy, sklearn, matplotlib, seaborn)

    LOAD dataset FROM path
    DEFINE features (X) AND target (y)

    CONVERT categorical columns USING LabelEncoder
    SCALE features (X) AND target (y)

    SPLIT data INTO train, validation, and test sets

    CREATE pipeline WITH PolynomialFeatures AND ElasticNet
    DEFINE parameter grid FOR ElasticNet
    INITIALIZE GridSearchCV WITH pipeline AND parameter grid
    FIT GridSearchCV TO training data

    SET pipeline TO best estimator
    MAKE predictions ON validation AND test sets
    COMPUTE evaluation metrics (MSE, R², MAE, RMSE)

    PRINT validation AND test set performance

    DEFINE function FOR predicting maintenance cost OF new car
        CREATE new car data
        CONVERT categorical features
        SCALE features
        PREDICT cost AND PRINT

    FOR EACH unique car MAKE PREDICT AND STORE results

    CREATE subplots FOR validation AND test set performance
    PLOT actual vs. predicted costs

    PLOT correlation matrix HEATMAP

END
```