CO2 EMISSIONS LINEAR REGRESSION PROJECT

This README file provides an overview of the tasks performed and the results obtained in the implementation of linear regression using various techniques on the CO2 Emissions Dataset in python using libraries like Scikit-Learn, Matplotlib, Pandas, Numpy and Seaborn

Table of Contents

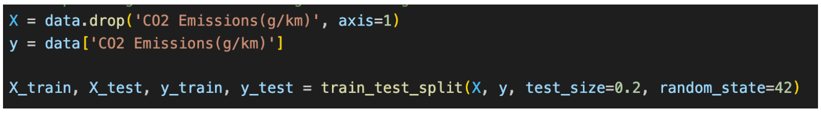
- Task Summary
- Data Split
- Data Visualization
- Dimensionality Reduction with TSNE
- Preprocessing and Linear Regression
- PCA and Model Training
- One-Hot Encoding and Linear Regression
- PCA on One-Hot Encoded Data
- Regularization Techniques
- Linear Regression with SGDRegressor
- Conclusion

Task Summary

In this project, we implemented linear regression on the CO2 Emissions Dataset using various techniques and libraries. The project was divided into several tasks, each focusing on different aspects of data preprocessing, dimensionality reduction, encoding methods, regularization, and model evaluation.

Data Split

Before proceeding with any analysis, we split the dataset into training and testing sets in an 80:20 ratio. This ensured that we had distinct datasets for training and evaluating our models.



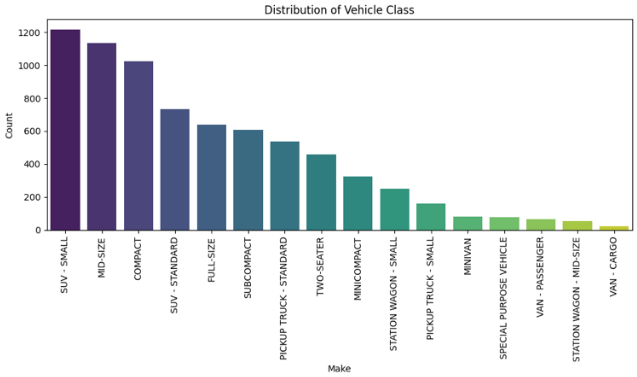
```
X = data.drop('CO2 Emissions(g/km)', axis=1)
y = data['CO2 Emissions(g/km)']

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

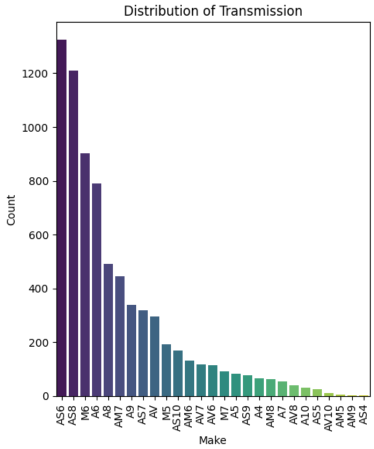
Data Visualization

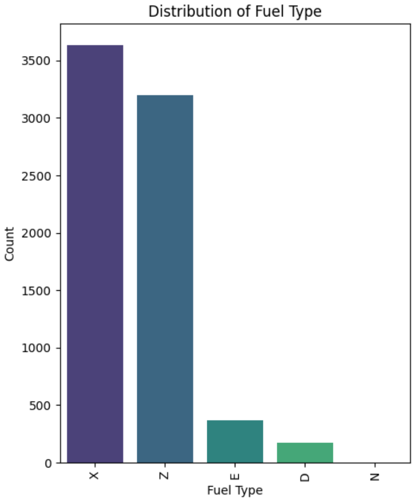
In this section, we visualized the dataset using scatter plots, pair plots, box plots, and a correlation heatmap. We also used distribution plots for categorical features and provided key insights into the dataset. Link to graphs: https://shorturl.at/krCEG

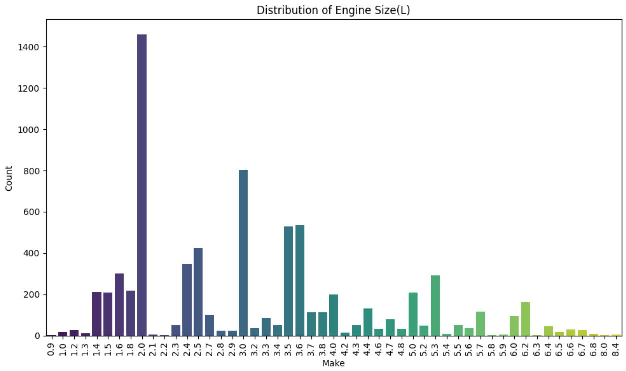
Some Insight on the data:

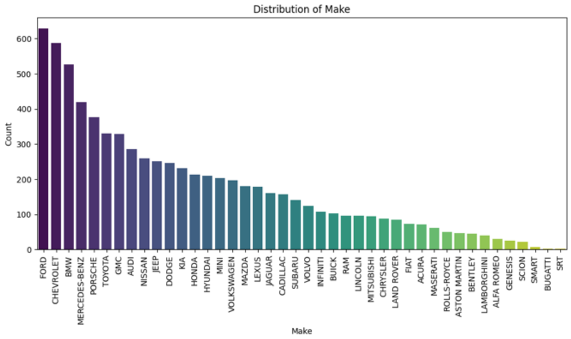
- Vehicles that have larger CO2 emissions generally have larger engines
- Most Vehicles have 4, 6 or 8 cylinders
- A large number of vehicles have a 2L or a 3L engine
- Almost all vehicles have either X or Z fuel type
- Fuel consumption has a largely linear relationship with CO2 emmissions
- Fuel Consumption City is also linearly related to Fuel Consumption Hwy
- The heatmap unveils robust positive associations among engine size, cylinder count, fuel consumption metrics, and CO2 emissions.
- Box plots show that some outliers are present in the dataset.

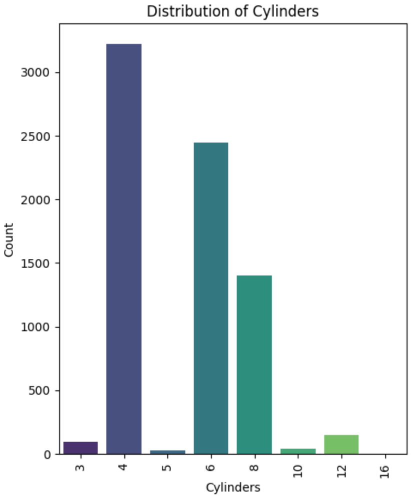

Distribution of Vehicle Class

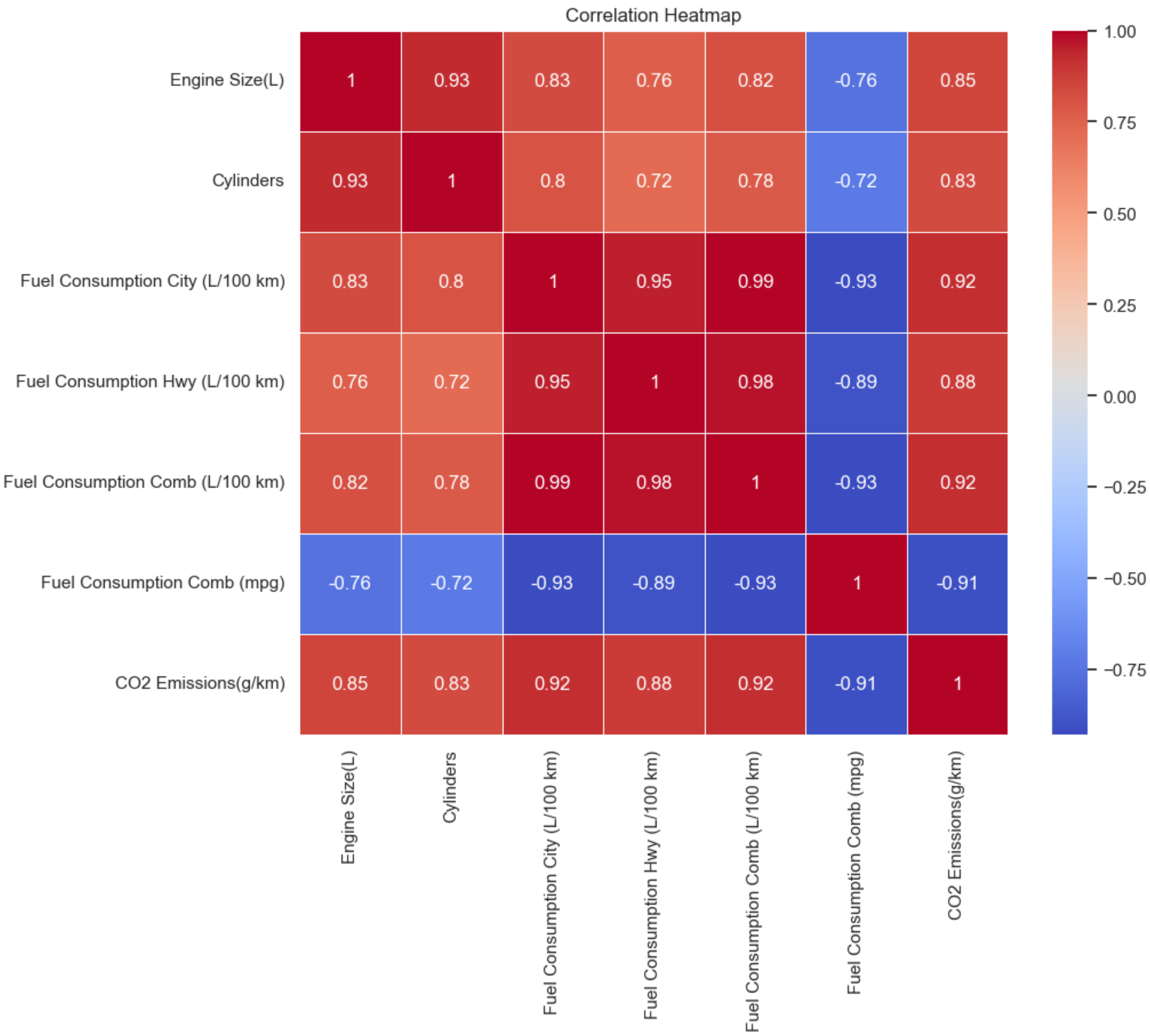

Distribution of Transmission

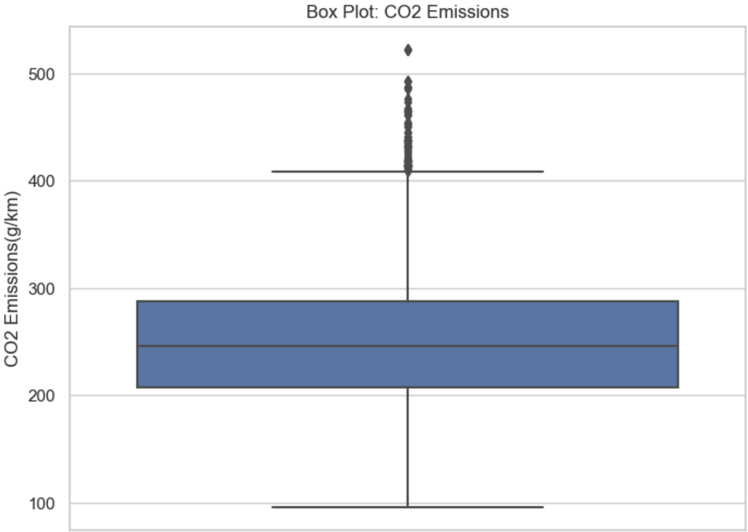

Distribution of Fuel Type


Distribution of Engine Size(L)


Distribution of Make

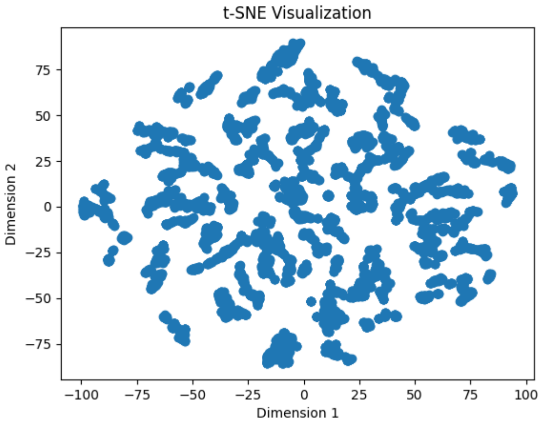

Distribution of Cylinders





Dimensionality Reduction with TSNE

We applied the TSNE algorithm to reduce the data dimensions to 2 and created scatter plots to assess the separability of the data.



After applying the t-SNE algorithm to our dataset, the resulting 2D visualization indicates the presence of roughly 6 to 7 distinct groupings within our high-dimensional data. However, there is some intersection between a few of these clusters, implying potential challenges in achieving complete separability for those particular groups.

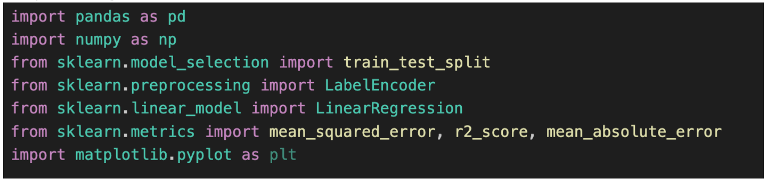
Additionally, we can see a handful of isolated data points, which could represent outliers or unique data instances.

Preprocessing and Linear Regression

We performed preprocessing steps on the data, including label-based encoding for categorical features. Linear regression was applied to the preprocessed data, and we reported metrics such as MSE, RMSE, R2 score, Adjusted R2 score, and MAE on both the train and test datasets.

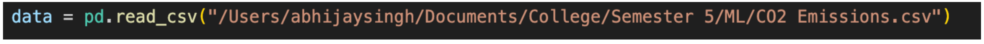
PREPROCESSING STEPS

- Importing Libraries



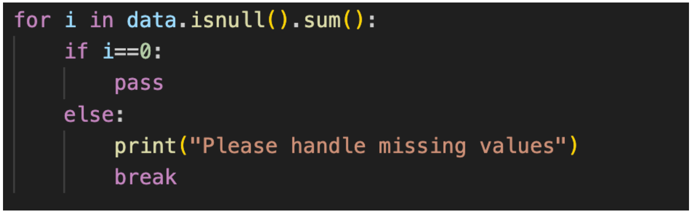
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
import matplotlib.pyplot as plt
```

- Loading Data



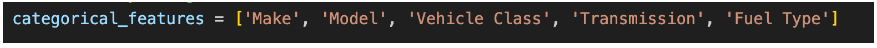
```
data = pd.read_csv("/Users/abhijaysingh/Documents/College/Semester 5/ML/CO2 Emissions.csv")
```

- Checking for missing values



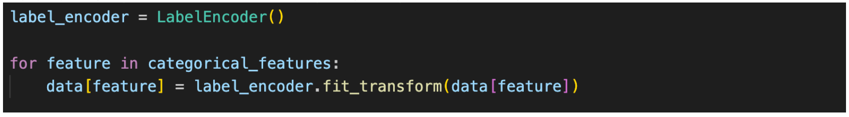
```
for i in data.isnull().sum():
    if i==0:
        pass
    else:
        print("Please handle missing values")
        break
```

- Identifying Categorical Features



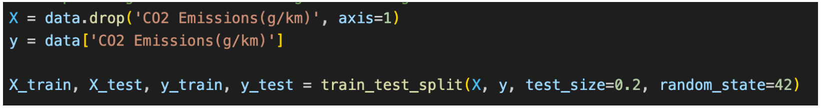
```
categorical_features = ['Make', 'Model', 'Vehicle Class', 'Transmission', 'Fuel Type']
```

- Applying label-based encoding to categorical features



```
label_encoder = LabelEncoder()

for feature in categorical_features:
    data[feature] = label_encoder.fit_transform(data[feature])
```

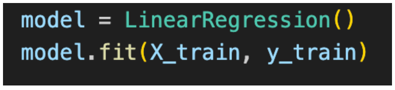
- Splitting data into Training and Testing Sets



```
X = data.drop('CO2 Emissions(g/km)', axis=1)
y = data['CO2 Emissions(g/km)']

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

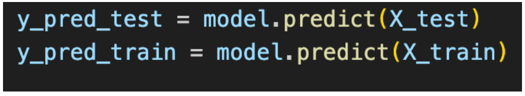
PERFORMING LINEAR REGRESSION ON THE PREPROCESSED DATA

- Creating and training the linear regression model



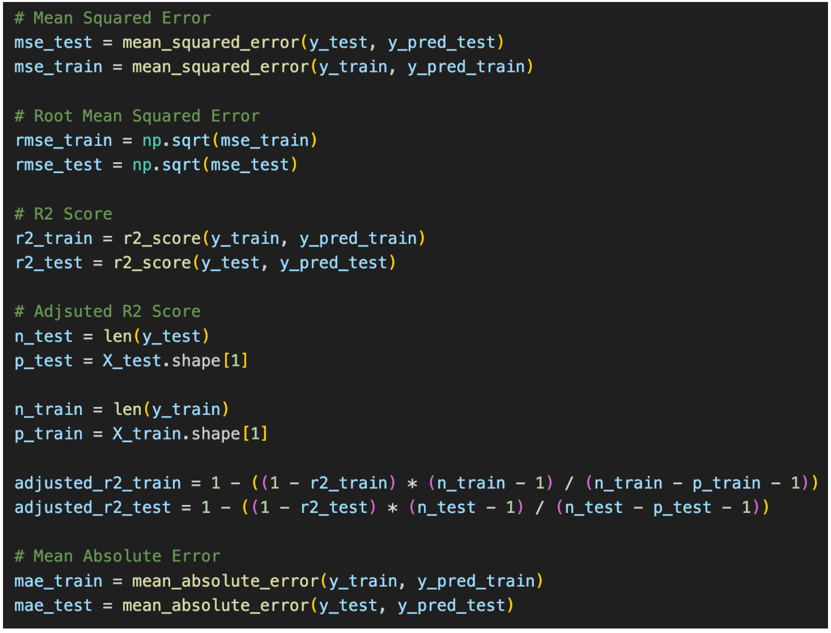
```
model = LinearRegression()
model.fit(X_train, y_train)
```

- Making predictions



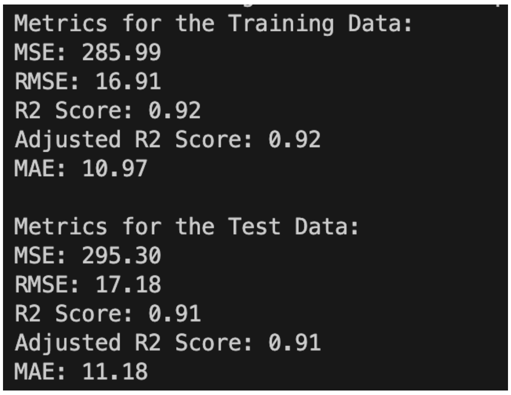
```
y_pred_test = model.predict(X_test)
y_pred_train = model.predict(X_train)
```

- Evaluating the Model



```
# Mean Squared Error
mse_test = mean_squared_error(y_test, y_pred_test)
mse_train = mean_squared_error(y_train, y_pred_train)

# Root Mean Squared Error
rmse_train = np.sqrt(mse_train)
rmse_test = np.sqrt(mse_test)

# R2 Score
r2_train = r2_score(y_train, y_pred_train)
r2_test = r2_score(y_test, y_pred_test)

# Adjsuted R2 Score
n_test = len(y_test)
p_test = X_test.shape[1]

n_train = len(y_train)
p_train = X_train.shape[1]

adjusted_r2_train = 1 - ((1 - r2_train) * (n_train - 1) / (n_train - p_train - 1))
adjusted_r2_test = 1 - ((1 - r2_test) * (n_test - 1) / (n_test - p_test - 1))

# Mean Absolute Error
mae_train = mean_absolute_error(y_train, y_pred_train)
mae_test = mean_absolute_error(y_test, y_pred_test)
```

The model gave the following predictions

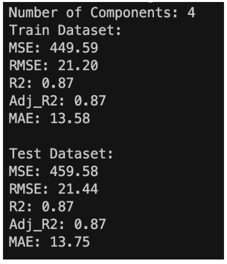


```
Metrics for the Training Data:
MSE: 285.99
RMSE: 16.91
R2 Score: 0.92
Adjusted R2 Score: 0.92
MAE: 10.97

Metrics for the Test Data:
MSE: 295.30
RMSE: 17.18
R2 Score: 0.91
Adjusted R2 Score: 0.91
MAE: 11.18
```

PCA and Model Training

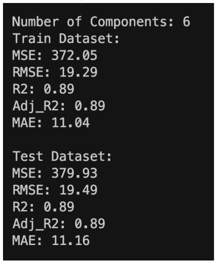
Principal Component Analysis (PCA) was used to reduce the number of features in the original dataset. We experimented with different numbers of components (4, 6, 8, 10) and trained the model with the reduced feature dataset. Model performance metrics were compared on both the train and test datasets.

- 4 Components



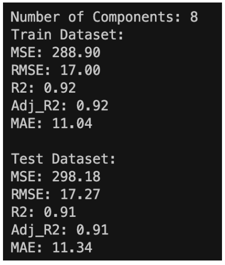
```
Number of Components: 4
Train Dataset:
MSE: 449.59
RMSE: 21.20
R2: 0.87
Adj_R2: 0.87
MAE: 13.58

Test Dataset:
MSE: 459.58
RMSE: 21.44
R2: 0.87
Adj_R2: 0.87
MAE: 13.75
```

- 6 Components



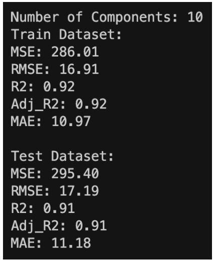
```
Number of Components: 6
Train Dataset:
MSE: 372.05
RMSE: 19.29
R2: 0.89
Adj_R2: 0.89
MAE: 11.04

Test Dataset:
MSE: 379.93
RMSE: 19.49
R2: 0.89
Adj_R2: 0.89
MAE: 11.16
```

- 8 Components



```
Number of Components: 8
Train Dataset:
MSE: 288.90
RMSE: 17.00
R2: 0.92
Adj_R2: 0.92
MAE: 11.04

Test Dataset:
MSE: 298.18
RMSE: 17.27
R2: 0.91
Adj_R2: 0.91
MAE: 11.34
```

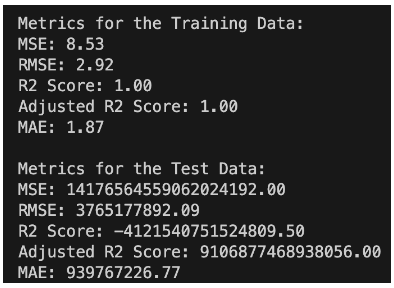
- 10 Components



```
Number of Components: 10
Train Dataset:
MSE: 286.01
RMSE: 16.91
R2: 0.92
Adj_R2: 0.92
MAE: 10.97

Test Dataset:
MSE: 295.40
RMSE: 17.19
R2: 0.91
Adj_R2: 0.91
MAE: 11.18
```

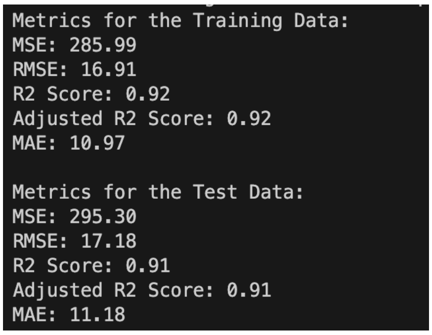
One-Hot Encoding and Linear Regression

Categorical features were encoded using one-hot encoding, and linear regression was applied again. We compared the results obtained with label-based encoding in terms of performance metrics.



```
Metrics for the Training Data:
MSE: 8.53
RMSE: 2.92
R2 Score: 1.00
Adjusted R2 Score: 1.00
MAE: 1.87

Metrics for the Test Data:
MSE: 14176564559062024192.00
RMSE: 3765177892.09
R2 Score: -4121540751524809.50
Adjusted R2 Score: 9106877468938056.00
MAE: 939767226.77
```

Comparing the results obtained earlier



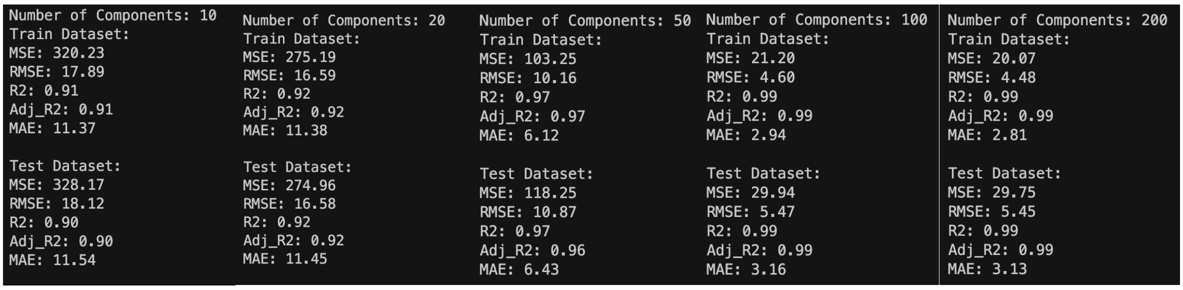
```
Metrics for the Training Data:
MSE: 285.99
RMSE: 16.91
R2 Score: 0.92
Adjusted R2 Score: 0.92
MAE: 10.97

Metrics for the Test Data:
MSE: 295.30
RMSE: 17.18
R2 Score: 0.91
Adjusted R2 Score: 0.91
MAE: 11.18
```

In contrast to the findings earlier, the model trained on the one-hot encoded dataset showcased outstanding performance during training but experienced a significant drop in its performance when tested on new data. This discrepancy between its ability to fit the training data and its inability to perform well on previously unseen data points highlights a pronounced issue of overfitting.

The extensive augmentation of features resulting from one-hot encoding, appears to have been a key factor contributing to the overfitting problem. The model's complexity was heightened to such an extent that it began capturing nuances and noise inherent to the training dataset, ultimately hindering its capacity to effectively generalize to new unseen data.

PCA on One-Hot Encoded Data

PCA was performed on the one-hot encoded dataset, trying different numbers of components (5 different values) to assess how reducing dimensionality impacts the predictive performance of a linear regression model. This analysis was carried out with varying numbers of components: 10, 20, 50, 100, and 200. Model performance metrics were compared on the train and test datasets.



```
Number of Components: 10
Train Dataset:
MSE: 320.23
RMSE: 17.89
R2: 0.91
Adj_R2: 0.91
MAE: 11.37

Test Dataset:
MSE: 328.17
RMSE: 18.12
R2: 0.90
Adj_R2: 0.90
MAE: 11.54
```

```
Number of Components: 20
Train Dataset:
MSE: 275.19
RMSE: 16.59
R2: 0.92
Adj_R2: 0.92
MAE: 11.38

Test Dataset:
MSE: 274.96
RMSE: 16.58
R2: 0.92
Adj_R2: 0.92
MAE: 11.45
```

```
Number of Components: 50
Train Dataset:
MSE: 103.25
RMSE: 10.16
R2: 0.97
Adj_R2: 0.97
MAE: 6.12

Test Dataset:
MSE: 118.25
RMSE: 10.87
R2: 0.97
Adj_R2: 0.96
MAE: 6.43
```

```
Number of Components: 100
Train Dataset:
MSE: 21.20
RMSE: 4.60
R2: 0.99
Adj_R2: 0.99
MAE: 2.94

Test Dataset:
MSE: 29.94
RMSE: 5.47
R2: 0.99
Adj_R2: 0.99
MAE: 3.16
```

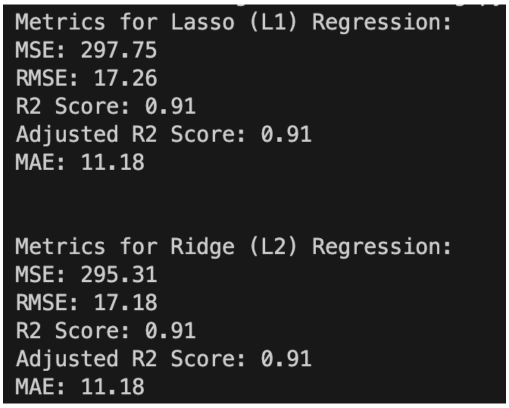
```
Number of Components: 200
Train Dataset:
MSE: 20.07
RMSE: 4.48
R2: 0.99
Adj_R2: 0.99
MAE: 2.81

Test Dataset:
MSE: 29.75
RMSE: 5.45
R2: 0.99
Adj_R2: 0.99
MAE: 3.13
```

As we analyze the table, a notable trend emerges: as the number of components utilized increases, there is a visible enhancement in the model's performance, reflected in improved metrics such as MSE, RMSE, R2, Adjusted R2, and MAE. However, it's important to recognize that this improvement rate begins to level off beyond a specific point, indicating diminishing returns.

These findings underscore the effectiveness of PCA (Principal Component Analysis) in reducing the dimensionality of the dataset. This reduction in dimensionality is achieved while still retaining a substantial portion of the dataset's variance intact, ultimately leading to notable improvements in model performance.

Regularization Techniques

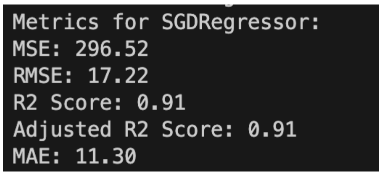
We applied L1 and L2 regularization while training the linear model on the preprocessed dataset. The performance metrics (MSE, RMSE, R2 score, Adjusted R2 score, MAE) were compared for both regularization techniques on the test dataset.



```
Metrics for Lasso (L1) Regression:
MSE: 297.75
RMSE: 17.26
R2 Score: 0.91
Adjusted R2 Score: 0.91
MAE: 11.18


Metrics for Ridge (L2) Regression:
MSE: 295.31
RMSE: 17.18
R2 Score: 0.91
Adjusted R2 Score: 0.91
MAE: 11.18
```

Linear Regression with SGDRegressor

The SGDRegressor library was used to perform linear regression on the preprocessed dataset. Evaluation metrics were reported, and the results were compared with previous approaches.



```
Metrics for SGDRegressor:
MSE: 296.52
RMSE: 17.22
R2 Score: 0.91
Adjusted R2 Score: 0.91
MAE: 11.30
```

Conclusion

This project provided a comprehensive analysis of linear regression techniques, dimensionality reduction, encoding methods, and regularization techniques on the CO2 Emissions Dataset. The results and insights obtained from each task contribute to a better understanding of the dataset and the effectiveness of different approaches in predicting CO2 emissions.