Group No. - 15
Abdul Azeem (21122028)
Ankit Patel (21122008)
Arunima Sood (21115025)
Aryan Raj (21122011)
Vaishnavi Kapse (21115153)

IBM 322 Group Assignment

12th May 2024

PROBLEM STATEMENT

In today's dynamic transportation scene, determining the rental price of a car has become indispensable. People often seek precise information on rental rates within a specific area to make informed choices. Hence, there's a growing need for a straightforward method to predict car rental prices. The projected rental cost acts as a vital resource for prospective renters, enabling them to compare rates based on factors like car model, location, city, and more. Providing an accurate rental price estimation empowers customers to navigate the rental market wisely, facilitating their decision-making and reservation process.

PROBLEM PARAMETERS

A dataset with following parameters is used in order to make predictions based on the requirements of the user.

1. Location (city, state, country)
2. Vehicle (make, model, type, year)
3. Rate on per day basis (rate.daily)
4. Fuel type
5. Rating and review count
6. Renter trips taken

DATA ANALYSIS

We have used the car rental dataset whose csv file is attached. The data set contains two types of data values:

1. Numerical data: The parameters such as rate per day, rating and renter trips taken are numerical and these are needed for mapping the user’s preferences.

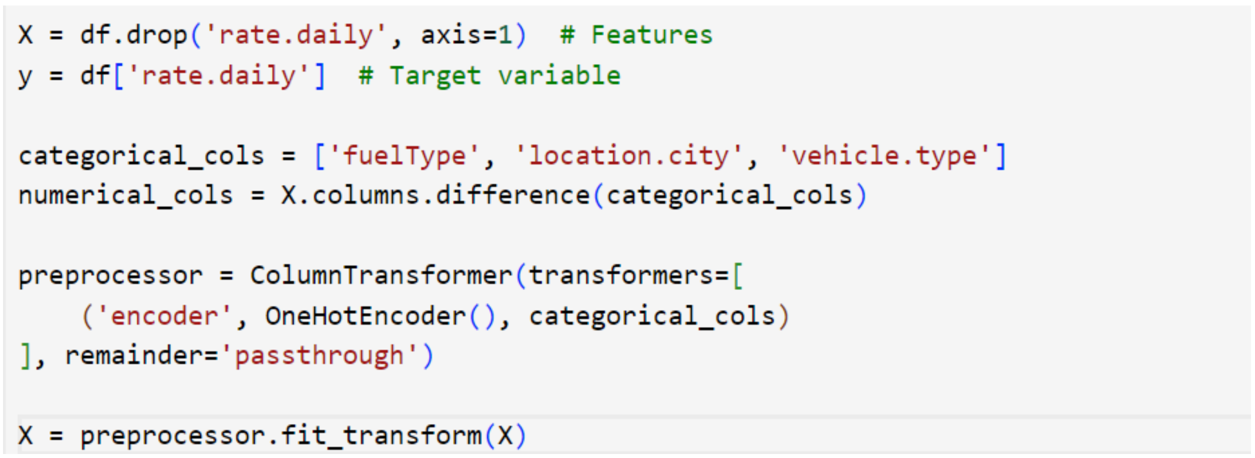
2. Categorical data: The parameters such as vehicle, location and fuel type are categorical and need some preprocessing before it can be used in our model. Apart from these there are several irrelevant fields which need to be dropped before working with the data.

METHODOLOGY

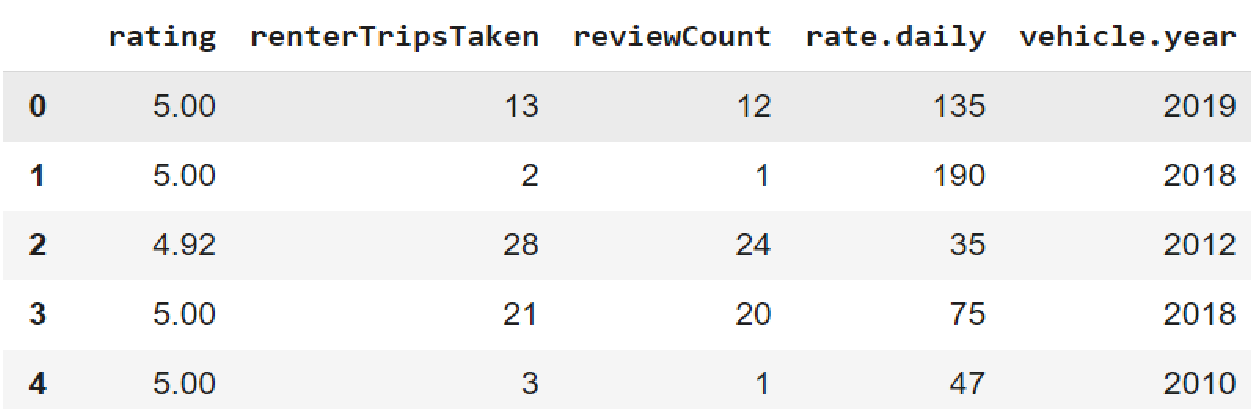
Pre processing

Handling Categorical Variables:

The dataset comprises various types of variables, but the internal models require numerical data rather than categorical. So, we must convert categorical variables into numerical values to facilitate classification. For this purpose, the model employs One Hot Encoding to convert the fuel type field into numerical values.



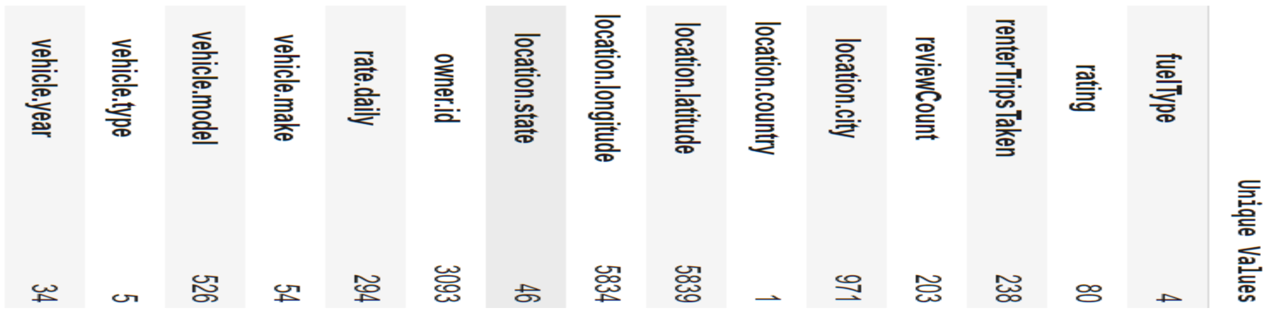
```
X = df.drop('rate.daily', axis=1)  # Features
y = df['rate.daily']  # Target variable

categorical_cols = ['fuelType', 'location.city', 'vehicle.type']
numerical_cols = X.columns.difference(categorical_cols)

preprocessor = ColumnTransformer(transformers=[
    ('encoder', OneHotEncoder(), categorical_cols)
], remainder='passthrough')

X = preprocessor.fit_transform(X)
```



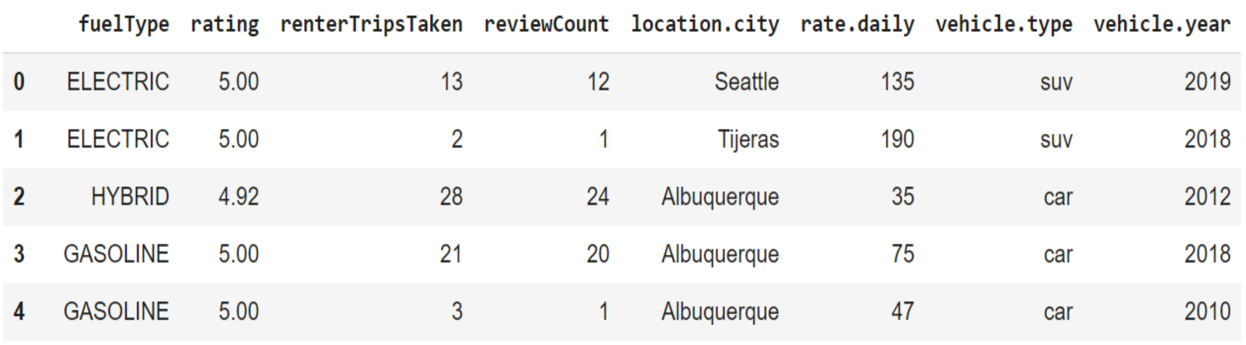
	rating	renterTripsTaken	reviewCount	rate.daily	vehicle.year
0	5.00	13	12	135	2019
1	5.00	2	1	190	2018
2	4.92	28	24	35	2012
3	5.00	21	20	75	2018
4	5.00	3	1	47	2010

Removing irrelevant fields:

Certain fields, such as owner ID, have minimal impact on the model's performance and should be excluded to prevent issues like overfitting. Similarly, the location field contains numerous categories, making it impractical to employ one-hot encoding directly. Therefore, we opt to remove these values from the dataset.



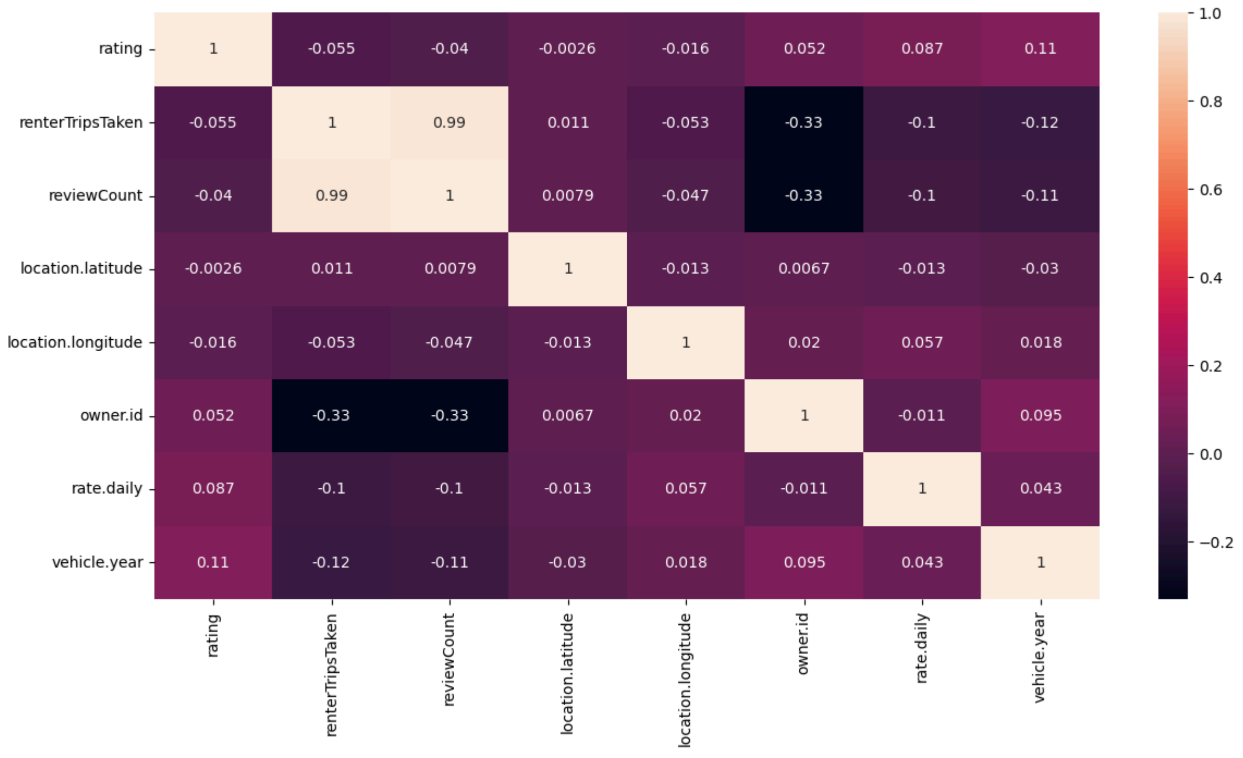
	vehicle.year	vehicle.type	vehicle.model	vehicle.make	rate.daily	owner.id	location.state	location.longitude	location.latitude	location.country	location.city	reviewCount	renterTripsTaken	rating	fuelType
Unique Values	34	5	526	54	294	3093	46	5834	5839	1	971	203	238	80	4



	fuelType	rating	renterTripsTaken	reviewCount	location.city	rate.daily	vehicle.type	vehicle.year
0	ELECTRIC	5.00	13	12	Seattle	135	suv	2019
1	ELECTRIC	5.00	2	1	Tijeras	190	suv	2018
2	HYBRID	4.92	28	24	Albuquerque	35	car	2012
3	GASOLINE	5.00	21	20	Albuquerque	75	car	2018
4	GASOLINE	5.00	3	1	Albuquerque	47	car	2010

Visualizing Correlation Among Numerical Parameters:

To comprehend the relationships between numerical parameters within the dataset, a heatmap is generated to depict the correlation matrix. Each cell in the heatmap represents the correlation coefficient between two numerical parameters. The intensity of the color and the annotation within each cell provide insights into the strength and direction of the correlation. Positive correlations are depicted in lighter shades, while negative correlations are indicated in darker hues. The magnitude of the correlation is reflected in the numerical annotations within the cells. This visualization aids in identifying patterns and dependencies among numerical parameters, guiding further analysis and modeling endeavors.



Clustering

The dataset comprises information across various states and cities, acknowledging that rental prices may differ based on city characteristics. To address this variability, we opt to train the dataset separately for each city, recognizing the distinct price dynamics in different urban environments. Additionally, clustering the data based on localities allows us to group nearby areas into the same cluster, potentially improving prediction accuracy by capturing localized car rental patterns. Moreover, we conduct training without clustering to facilitate a comparative analysis of results.

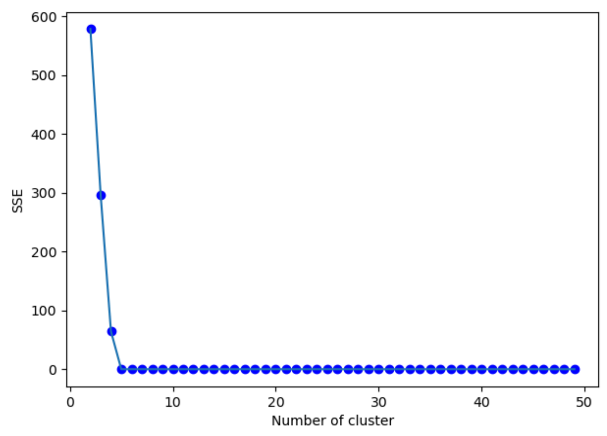
ALGORITHMS

K-Means Clustering

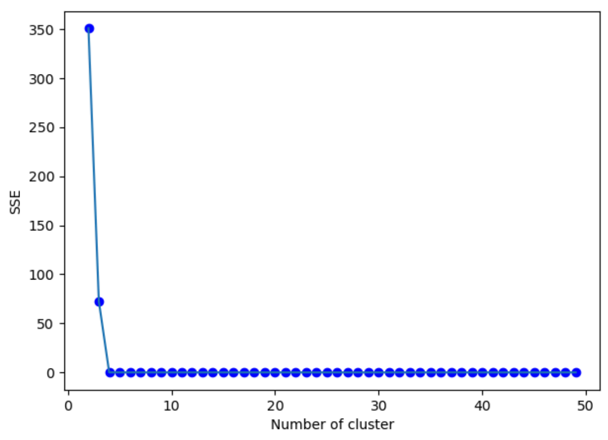
After data preprocessing is done, the first step is to cluster the cars via K-Means clustering algorithm. The optimal number of clusters are found using Davies Bouldin score and the Elbow method. The algorithm uses the K-Means clustering algorithm to group cars into different vehicle and fuel types.

Clustering on the basis of vehicle type feature: The optimal number of clusters on the basis of vehicle type is found to be 5 for our dataset.

Clustering on the basis of fuel type feature: The optimal number of clusters on the basis of fuel type is found to be 4 for our dataset.



Clustering on vehicle type feature



Clustering on fuel type feature

Linear Regression

Linear Regression is a statistical method utilized for modeling the relationship between a target (subordinate) variable and one or more independent factors. In the setting of the car rental cost pricing model, Linear regression points to predict the rental cost (Y) based on different factors such as car make, model, rating, rental duration, and other significant variables (X). Mathematically speaking, linear regression assumes a linear relationship between the independent factors and the target variable, spoken to as:

Y is the predicted rental price;

β_0 is the intercept term (value of target variable Y when all independent factors are zero);

$\beta_1, \beta_2, ..., \beta_n$ are the coefficients of the independent variables, representing the change in Y for a one-unit change in each respective X;

$X_1, X_2, ..., X_n$ are the independent variables;

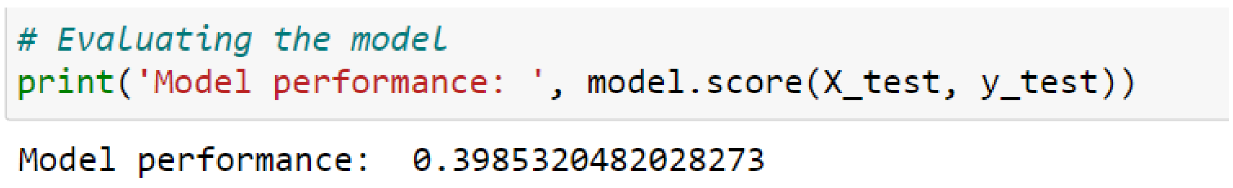
ϵ is the error term, representing the difference between the observed and predicted values.

The objective of the algorithm is to assess the coefficients β1, β2,...,βn that minimize the whole of squared residuals between the observed and the predicted values. This is ordinarily done utilizing the strategy of least squares, where the coefficients are chosen to minimize the entirety of the squared differences between the observed Y values and the values predicted by the model. Once the model is trained on the dataset, it can be utilized to predict rental costs for new occurrences of cars by plugging in the values of their features into the regression equation. The performance of the model can be assessed utilizing measurements such as mean squared error (MSE) and R^2 score, which evaluate how well the model fits the data.

For each cluster representing a fuel or a vehicle type, a separate linear regression model is trained using the features specific to that type. This is achieved by iterating over each type, extracting the relevant data, and training a linear regression model. When predicting the price of the rental car, we first predict its type from K-Means clustering algorithm, then predict the price using the fuel type specific linear regression model.

TRAINING AND RESULTS

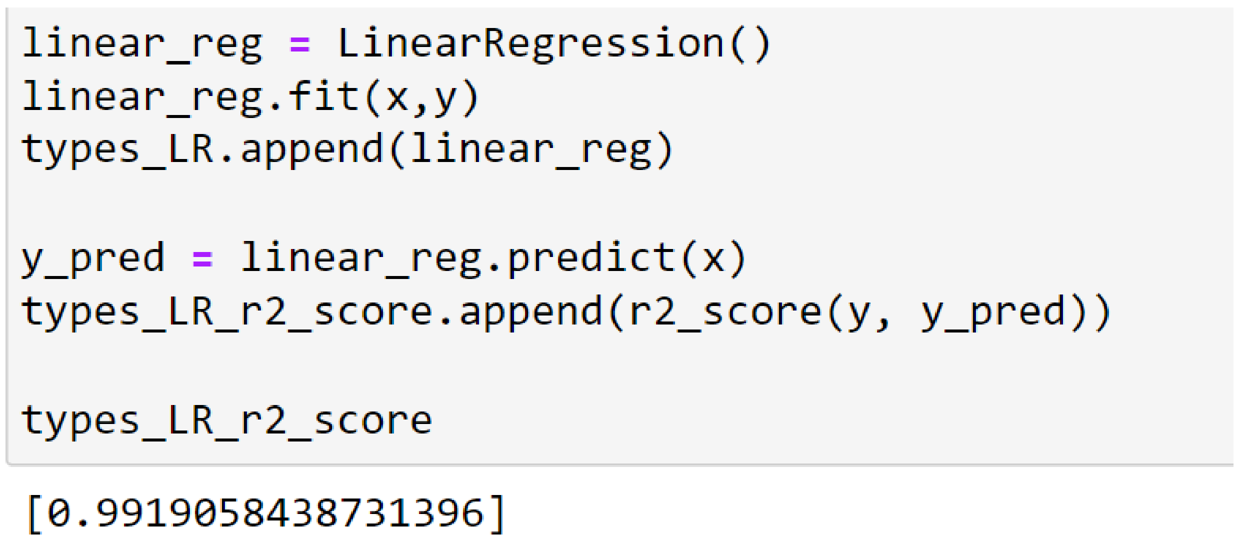
Model performances without clustering



```
# Evaluating the model
print('Model performance: ', model.score(X_test, y_test))
```

```
Model performance:  0.3985320482028273
```

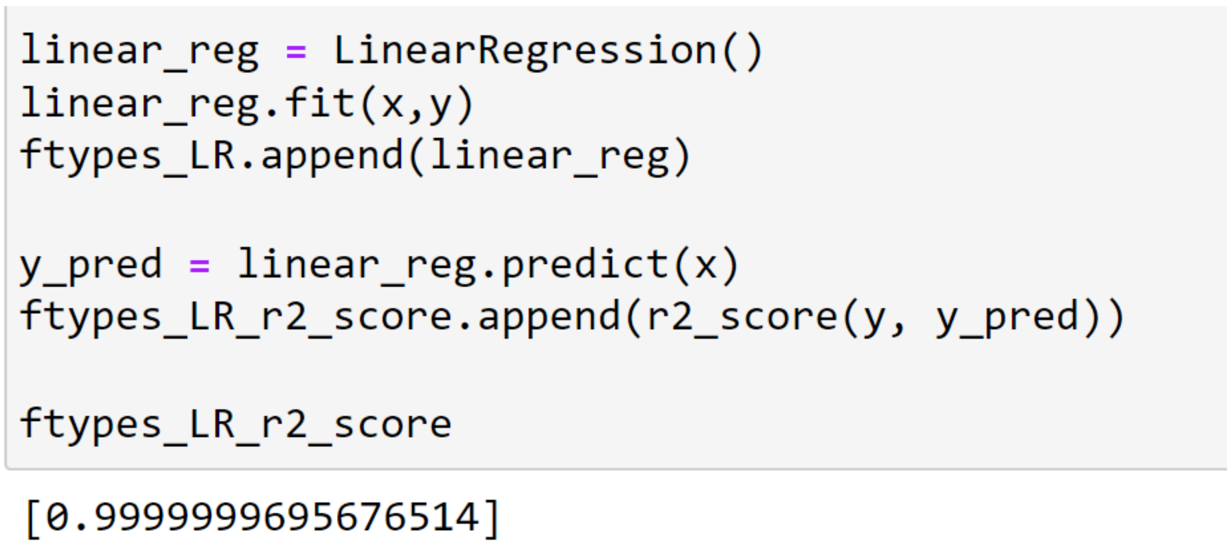
R2-Scores with clustering on basis of vehicle type feature



```
linear_reg = LinearRegression()
linear_reg.fit(x,y)
types_LR.append(linear_reg)

y_pred = linear_reg.predict(x)
types_LR_r2_score.append(r2_score(y, y_pred))

types_LR_r2_score
```

```
[0.9919058438731396]
```

R2-Scores with clustering on basis of fuel type feature



```
linear_reg = LinearRegression()
linear_reg.fit(x,y)
ftypes_LR.append(linear_reg)

y_pred = linear_reg.predict(x)
ftypes_LR_r2_score.append(r2_score(y, y_pred))

ftypes_LR_r2_score
```

```
[0.9999999695676514]
```