Shourya Jindal
2020336

ML Assignment 1

SECTION A

SHOURYA JINDAL
2020336
ML ASSIGNMENT 1

SECTION A

(Q1) (a) No, a strong correlation b/w 2 variables with a 3rd variable, it does not necessarily apply that they will also display high degree of correlation with each other.

Explaination:

Correlation measures the statistical relation b/w 2 variables. But the presence of 3rd variable that is related to both the other variables can affect the observed correlation b/w them. When 3rd variable is not taken into account it can create a misleading impression of direct relⁿ b/w the 1st 2 variables.

Example:
X: Exercise (Variable representing how much a person is exercising)
Y: Diet (Variable representing how much healthy diet a person is taking)
Z: weightloss (Variable representing how much weight is lost by a person).

We, can see that generally

$X \propto Z$ (more a person exercises/workout more he/she will lose weight)

$Y \propto Z$ (healthy diet $\Rightarrow$ more weight loss)

but we cannot say that X and Y are related. a person may Exercise more but still may not have good diet or vice-versa. Although, it may look like they are related since both leads to weight loss, but we have to consider other factors like "metabolism".

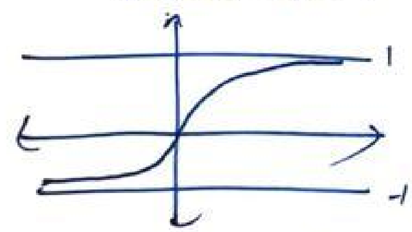
(Q2) Criterias for a mathematical func^n to be categorized as a logistic func^n are:

- Should have S-shaped curve and must be bounded b/w a minima and maxima
- Domain should be $\mathbb{R}$ and Range must be $[0,1]$ or bounded
- Should be continuous, differentiable
- We should be able to draw a decision boundary

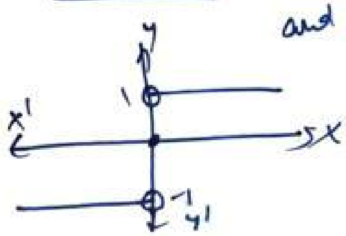
- Sinh(x) and Cosh(x): They are not valid logistic func^n as they are not bounded

$$\left.\begin{array}{l} \text{Range of } \sinh x = (-\infty, \infty) \\ \text{'' of } \cosh x = [1, \infty) \end{array}\right\} \rightarrow \text{not bounded.}$$

- tanh(x): Yes it is valid logistic func^n as it satisfied all criterias.
  - It is sigmoidal i.e S-shaped
  - Is bounded, continuous, differentiable
  - Domain $= \mathbb{R}$, Range $= [-1, 1] \} \rightarrow$ bounded.



- Signum(x): Not valid logistic func^n as it is not continuous and does not have S-shaped curve



(Q3) For very sparse datasets, leave one out ~~cross~~ cross validation is beneficial.
This is because of the following reasons:

- It utilizes and maximises data ~~uses~~ usage for both training and validation
- It generally gives low ~~variance~~ bias estimate of the model. This is useful is sparse datasets as there is high degree of variability due to limited no. of samples

How it is diff^n:

- In this technique, we train the model on all the data pts except one, which is then used for testing. This process is repeated till all pts have been used for testing.
  Average of performance is calculated for all the iterations.
- While in K-fold. we divide dataset into K-subsets and then uses 'K-1' folds for training and '1' fold for testing. This is repeated 'K' times.

K-fold	Leave one out
• Requires K-iterations so, faster	• Req. n-iterations, hence requires more time and computes.
• balance b/w bias and variance depending on K.	• Generally ~~to~~ low bias and high variance model, as it trains on all the data except one

(Q4) Find coeff^n of least square regression in slope - intercept form.

Let reg func^n be : $y = mx + c$

where m = slope and c = intercept are the unknown co~~ee~~efficients.

Cost func^n ~~co~~ for least square regression would be :

$$J(y,x) = \frac{1}{n}\sum_{i=1}^{n}(y_i - y)^2 \qquad \text{(for } n\text{-data points)}$$

$y_i \rightarrow$ actual value

$y \rightarrow$ predicted value

$x_i \rightarrow$ ~~ac~~ input

~~o~~ we need to minimize $J(y,x)$.

So, minimize $J(y,x) = \frac{1}{n}\sum_{i=1}^{n}(y_i - mx_i + c)^2$

So,

$$\frac{\delta J}{\delta m} = 0 \quad \& \quad \frac{\delta J}{\delta c} = 0$$

$$\Rightarrow \frac{2}{n}\Sigma(y_i - mx_i + c)(-x_i) = 0 \quad \& \quad \frac{2}{n}\Sigma(y_i - mx_i + c) = 0$$

$$\Rightarrow \boxed{m = \frac{\Sigma x_i y_i - c\Sigma x_i}{\Sigma x_i^2}} \quad \& \quad \boxed{m = \frac{\Sigma y_i - nc}{\Sigma x_i}}$$

on solving we get

$\cancel{m = n\Sigma x_i}$ $\cancel{c = \Sigma\Sigma y_i}$

$$\boxed{m = \frac{n\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2}}$$

and

$$\boxed{c = \frac{\Sigma y_i - m\Sigma x_i}{n} = \frac{\Sigma y_i}{n} - \left(\frac{\Sigma x_i}{n}\right)\left(\frac{n\Sigma x_i y_i - \Sigma x_i \Sigma y_i}{n\Sigma x_i^2 - (\Sigma x_i)^2}\right)}$$

(Q5) Ans: (a) α, β, σ

$y = \alpha + \beta x + \epsilon \quad \epsilon N(0, \sigma)$

Here 'α, β' are the coefficien~~ient~~ts/weights/parameter of simple linear regression model and 'σ' is the Standard deviation of the ϵ variable with which ~~fell~~ is the Noise parameter which follows $N(0, \sigma)$ dist^n.

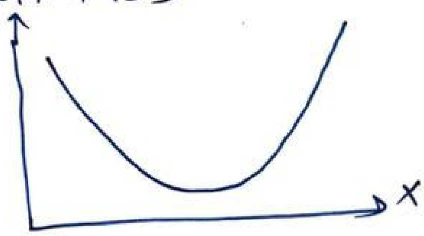
(Q6) Ans: (d) $y = \alpha + \beta_1 x + \beta_2 x^2 + \epsilon \quad \beta_2 > 0$

Reason: $X = [20, 30, 50, 60, 80, 90]$

$Y = [125, 110, 95, 90, 110, 130]$

If we plot this graph:

Rough graph ←

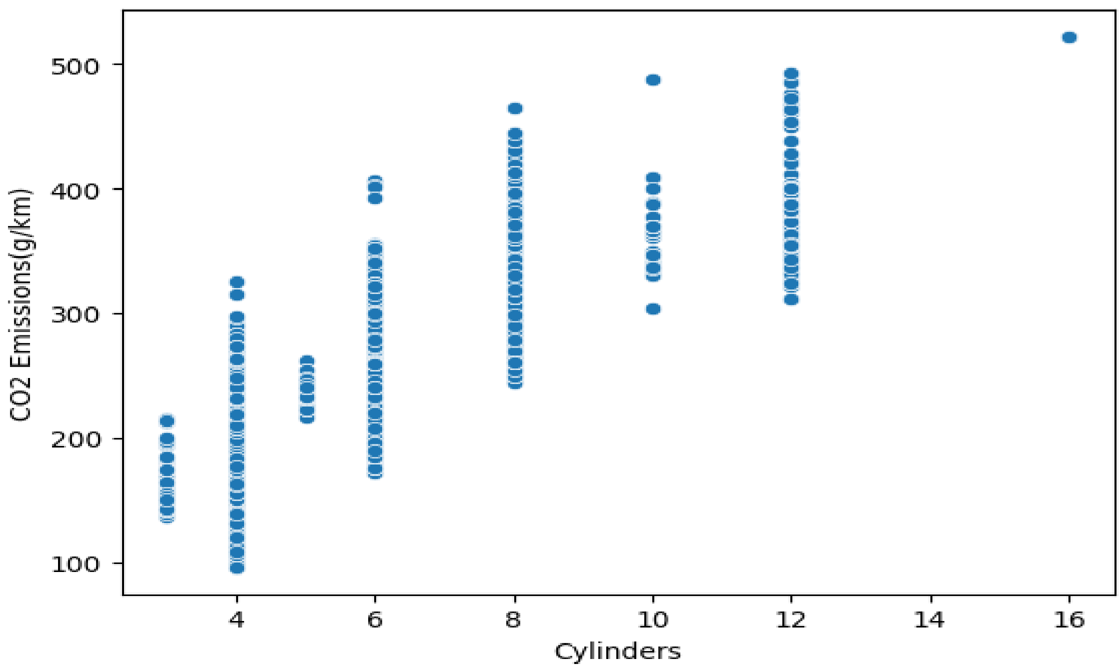


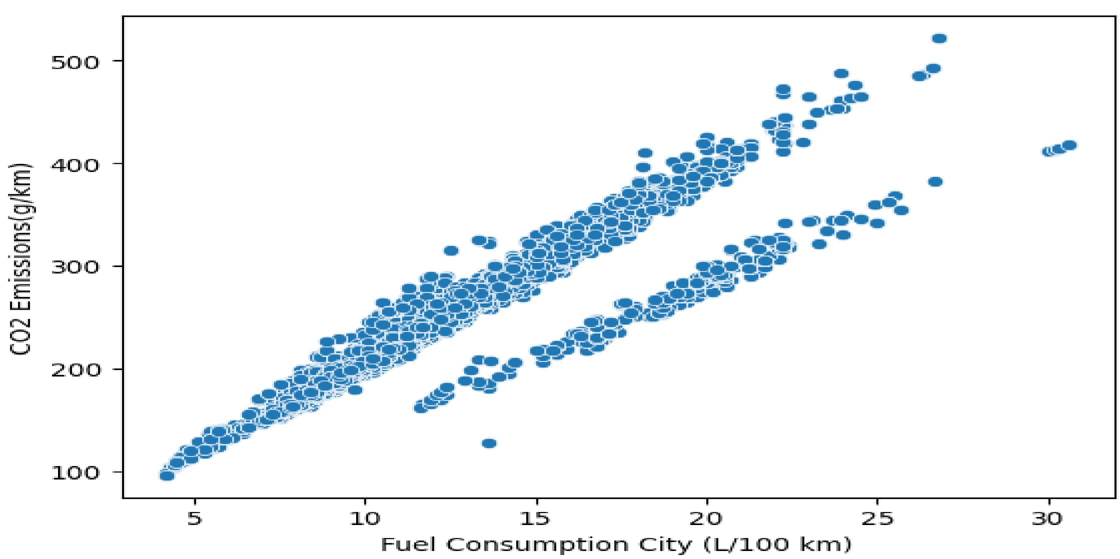
as we can see it is not linear but quadratic hence (a)(b) are ruled out

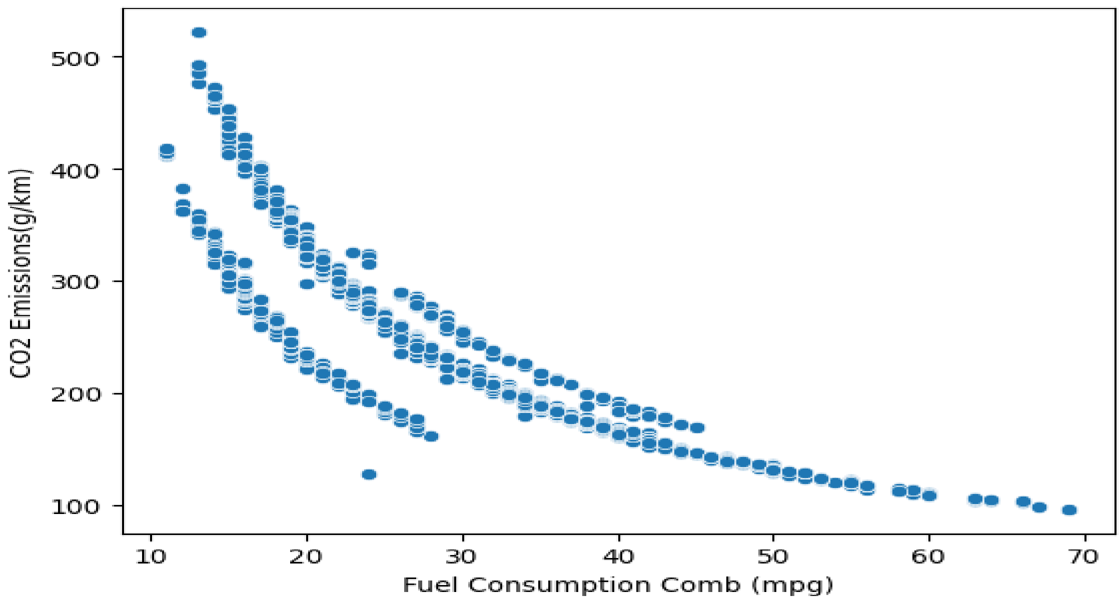
also, this parabolic/quadratic graph reaches minima and is upward ~~facing~~ facing thus. $\beta_2 > 0$.

SECTION C

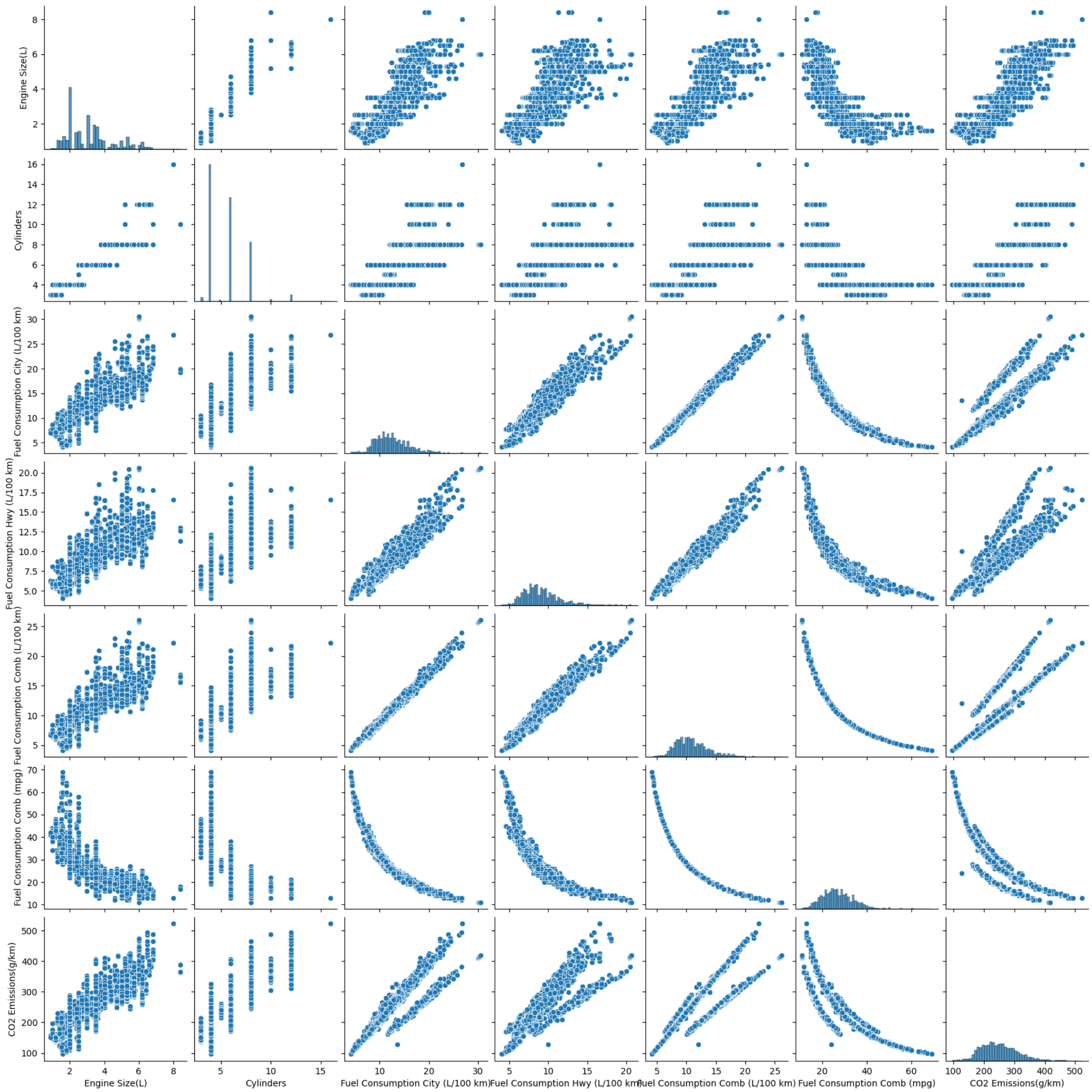
PART A: DATA VISUALIZTION
I visualized the data by creating several graphs and plots.
Some of which are:

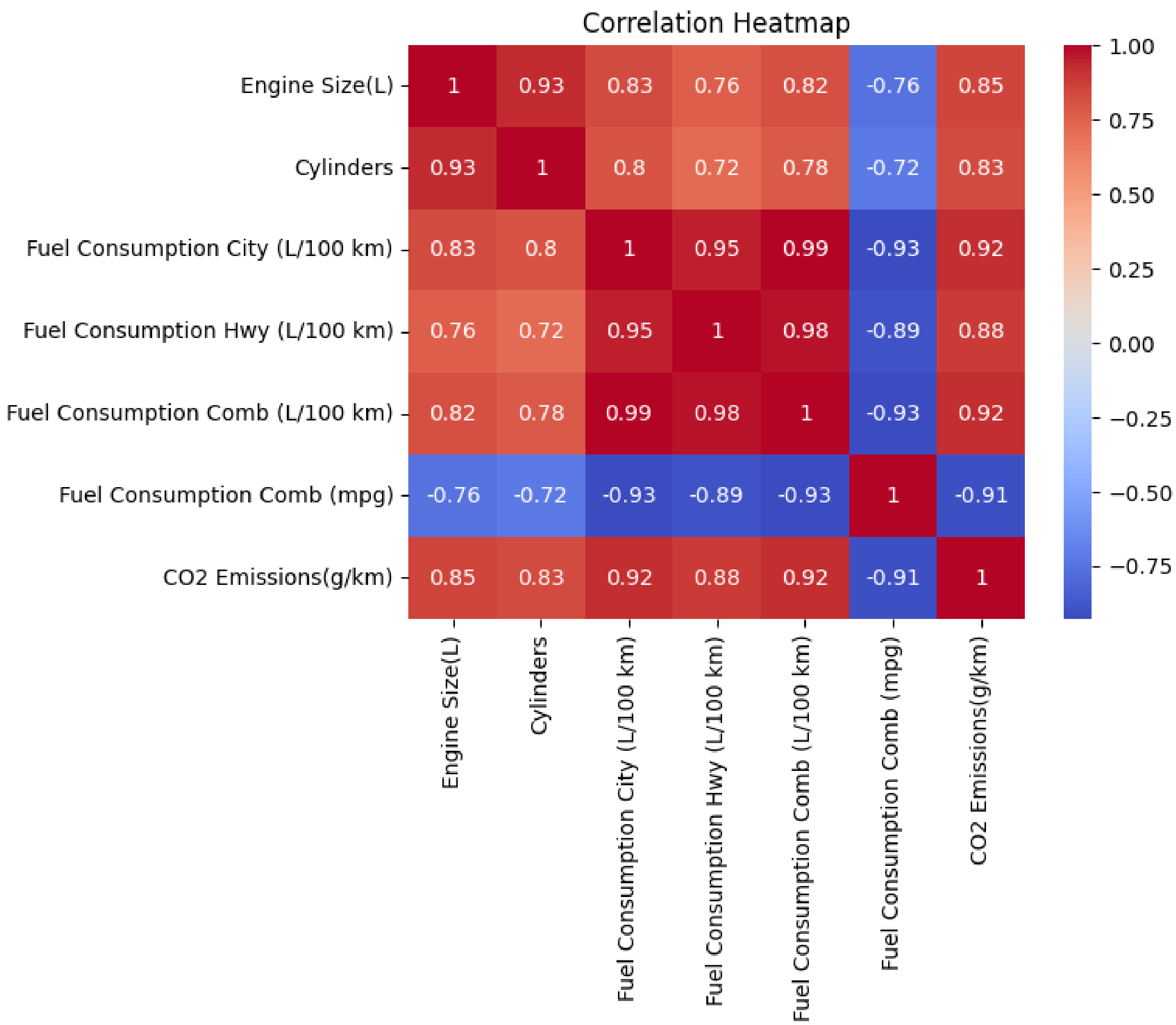


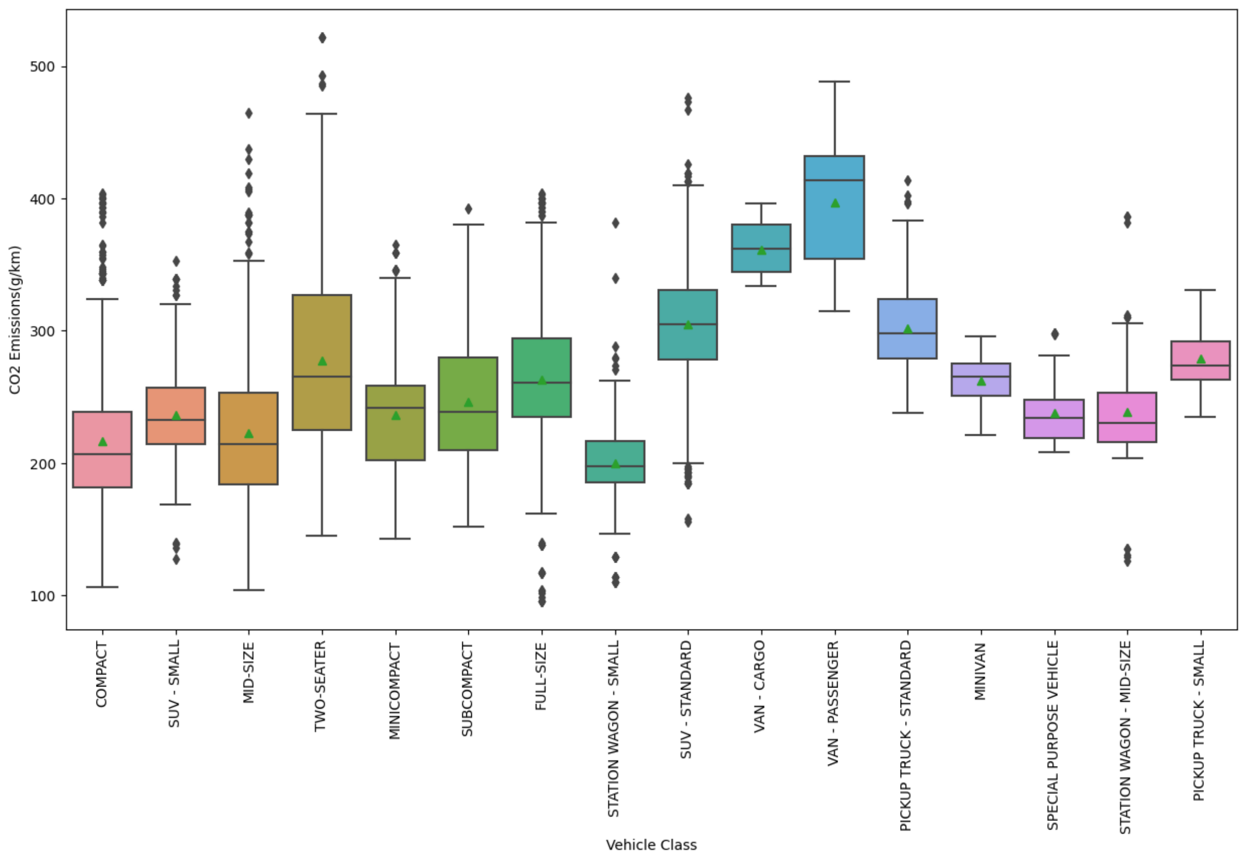




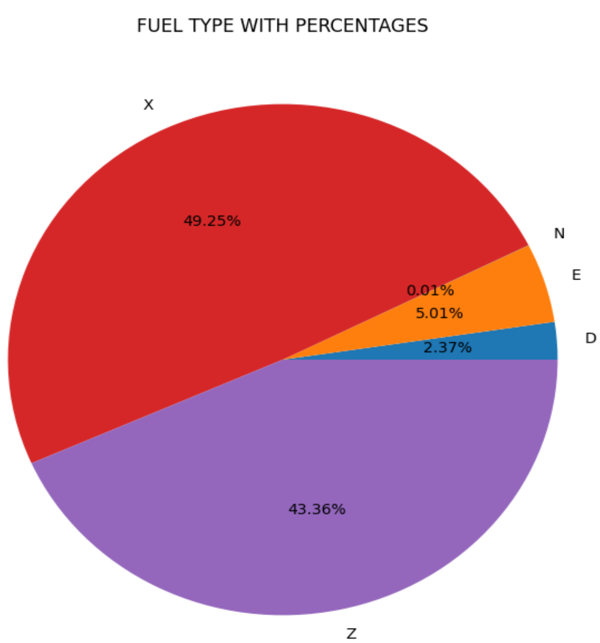
CO2 Emissions(g/km)
500
400
300
200
100
10
20
30
40
50
60
70
Fuel Consumption Comb (mpg)


Engine Size(L)
Cylinders
Fuel Consumption City (L/100 km)
Fuel Consumption Hwy (L/100 km)
Fuel Consumption Comb (L/100 km)
Fuel Consumption Comb (mpg)
CO2 Emissions(g/km)

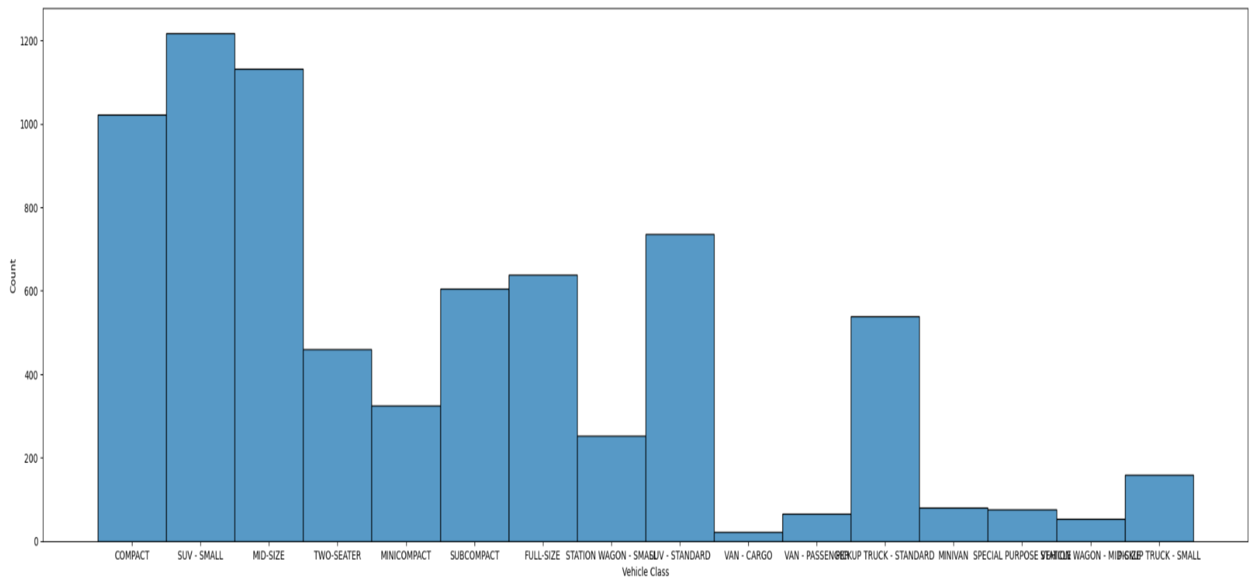

Correlation Heatmap
Engine Size(L)
Cylinders
Fuel Consumption City (L/100 km)
Fuel Consumption Hwy (L/100 km)
Fuel Consumption Comb (L/100 km)
Fuel Consumption Comb (mpg)
CO2 Emissions(g/km)
1 0.93 0.83 0.76 0.82 -0.76 0.85
0.93 1 0.8 0.72 0.78 -0.72 0.83
0.83 0.8 1 0.95 0.99 -0.93 0.92
0.76 0.72 0.95 1 0.98 -0.89 0.88
0.82 0.78 0.99 0.98 1 -0.93 0.92
-0.76 -0.72 -0.93 -0.89 -0.93 1 -0.91
0.85 0.83 0.92 0.88 0.92 -0.91 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75



CO2 Emissions(g/km)
100
200
300
400
500
COMPACT
SUV - SMALL
MID-SIZE
TWO-SEATER
MINICOMPACT
SUBCOMPACT
FULL-SIZE
STATION WAGON - SMALL
SUV - STANDARD
VAN - CARGO
VAN - PASSENGER
PICKUP TRUCK - STANDARD
MINIVAN
SPECIAL PURPOSE VEHICLE
STATION WAGON - MID-SIZE
PICKUP TRUCK - SMALL
Vehicle Class



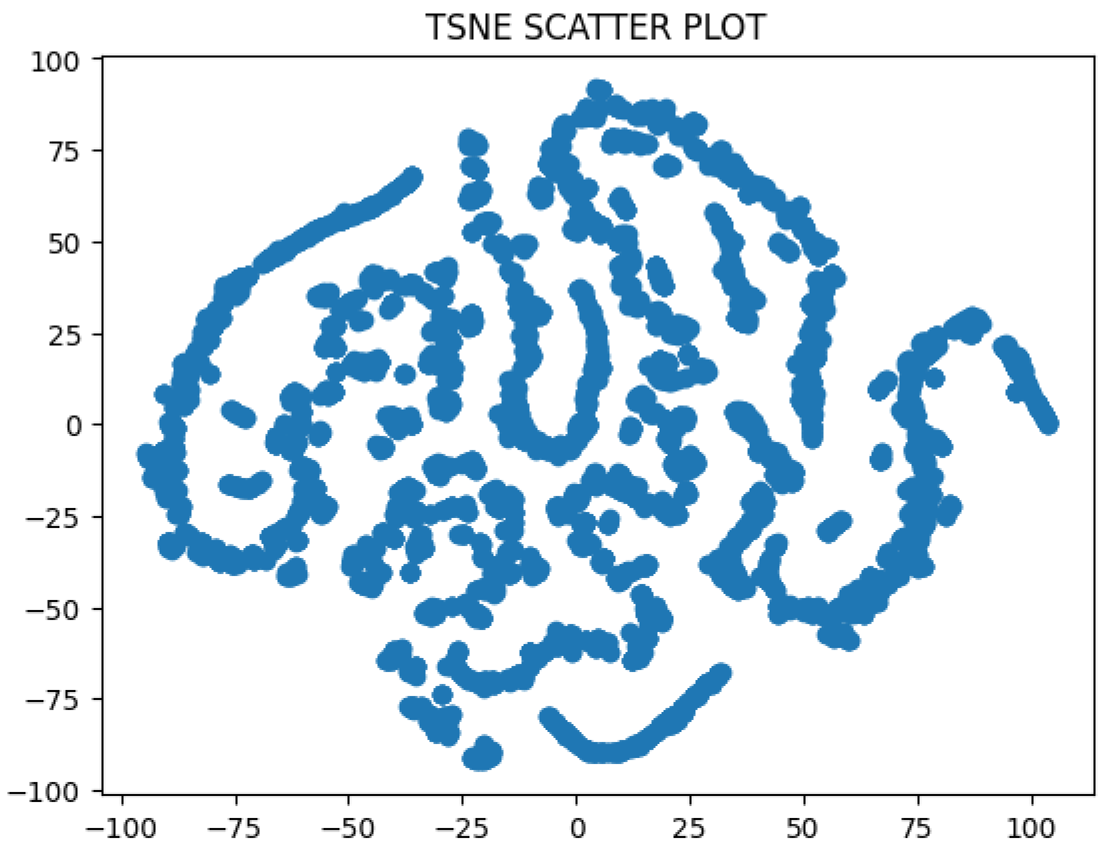
FUEL TYPE WITH PERCENTAGES
X
49.25%
N
0.01%
E
5.01%
D
2.37%
43.36%
Z



Rest are in the other folder and in the code itself.
Some insights for the data are:

- Cylinders are directly proportional to CO2 Emissions
- Engine Size is also directly proportional to CO2 Emissions.
- From heatmap we can see that cylinders and engine size has a very high correlation.
- Also Fuel Consumption (city, hwy etc.) are also directly proportional to CO2 emissions
- Almost 50% of the vehicles are of Fuel Type - ‘X’, and 43% of type ‘Z’ and rest have very few numbers.
- From the box plot of Vehicle class vs CO2 emissions, we infer that vehicle class do not affects CO2 emissions very much.
- Also ‘make’, ‘model’ have very low correlation with Co2 emissions.

Part B: TSNE
I used TSNE algorithm (from sklearn library) to reduce data dimensions to 2 and plotted the resulting data as a scatter plot.



The scatter plot shows that most of the clusters are not independent. The features forms clusters which intermixing on TSNE scatter plot and thus, are not seperable and are most of them are interrelated.

PART C: LABEL ENCODING

Used LabelEncoding from sklearn to LabelEncode the categorical data : `'Make', 'Model', 'Vehicle Class', 'Transmission', 'Fuel Type']`

Then after preprocessing, I split the data into train:test = 80:20, then applied Linear Regression on it, also using sklearn.

The Metric and Performance Results are as follows:

	TRAINING DATA	TESTING DATA
MSE	295.177056	258.138270
RMSE	17.180718	16.066682
R2	0.913177	0.926511
Adjusted R2	0.913015	0.926373
MAE	11.238194	10.470382

We get a good R2 score of 0.91 and very less rmse error.

PART D: PCA on label encoded

Results are as follows:

PCA WITH NO OF COMPONENTS = 2

	TRAINING DATA	TESTING DATA
MSE	2776.707846	2554.442701
RMSE	52.694476	50.541495
R2	0.200487	0.207222
Adjusted R2	0.200216	0.206954
MAE	41.293940	39.795481

PCA WITH NO OF COMPONENTS = 4

	TRAINING DATA	TESTING DATA
MSE	457.237671	436.144308
RMSE	21.383116	20.884068
R2	0.867724	0.867347
Adjusted R2	0.867634	0.867257
MAE	13.783732	13.475276

PCA WITH NO OF COMPONENTS = 6

	TRAINING DATA	TESTING DATA
MSE	369.059370	393.693058
RMSE	19.210918	19.841700
R2	0.891220	0.888947
Adjusted R2	0.891109	0.888834
MAE	11.030404	11.427482

PCA WITH NO OF COMPONENTS = 8

	TRAINING DATA	TESTING DATA
MSE	287.753851	302.565302
RMSE	16.963309	17.394404
R2	0.916848	0.907566
Adjusted R2	0.916735	0.907441
MAE	11.119796	11.333621

PCA WITH NO OF COMPONENTS = 10

	TRAINING DATA	TESTING DATA
MSE	291.276442	274.122151
RMSE	17.066823	16.556635
R2	0.915465	0.917736
Adjusted R2	0.915322	0.917597
MAE	11.155951	11.018914

Used PCA from sklearn to implement this. We observe that as no of components increases error decreases and accuracy increases. The R2 score increased from 0.20 (when no of components = 2) to 0.91(when no of components = 10), which is very significant improvement.

PART E: One-Hot Encoding

I one hot coded the original data on categorical data using pd.get_dummies.
Since the no. of distinct values of categorical data was very large, no of columns increases to almost 2150 thus greatly increasing the size of the data.
Then we did Linear Regression using sklearn.

Performance analysis:

	TRAINING DATA	TESTING DATA
MSE	8.570359	2.888609e+20
RMSE	2.927518	1.699591e+10
R2	0.997464	-8.030857e+16
Adjusted R2	0.996014	-1.262328e+17
MAE	1.893353	3.508334e+09

On comparing the these results with part c, we observe that one hot encoding performs much better than label encoding on training data but performs very poorly on testing data as compared to part c.
This is because one hot encoding led the model to get overfit. As a result bais decreased but variance increased very much.

PART F: PCA on One-hot Encoded:

Performance analysis:

```
PCA WITH NO OF COMPONENTS = 2
             TRAINING DATA  TESTING DATA
MSE             376.272097    383.239369
RMSE             19.397734     19.576500
R2                0.889951      0.888564
Adjusted R2       0.889914      0.888527
MAE              11.006305     11.030149

PCA WITH NO OF COMPONENTS = 4
             TRAINING DATA  TESTING DATA
MSE             336.824469    318.892616
RMSE             18.352778     17.857565
R2                0.902115      0.904769
Adjusted R2       0.902049      0.904704
MAE              11.591412     11.440505

PCA WITH NO OF COMPONENTS = 6
             TRAINING DATA  TESTING DATA
MSE             325.036415    320.624306
RMSE             18.028766     17.905985
R2                0.905821      0.903123
Adjusted R2       0.905726      0.903025
MAE              11.423221     11.482435

PCA WITH NO OF COMPONENTS = 8
             TRAINING DATA  TESTING DATA
MSE             323.992192    322.473531
RMSE             17.999783     17.957548
R2                0.904944      0.907380
Adjusted R2       0.904815      0.907254
MAE              11.387942     11.514365

PCA WITH NO OF COMPONENTS = 10
             TRAINING DATA  TESTING DATA
MSE             324.861023    318.542335
RMSE             18.023901     17.847754
R2                0.905793      0.904132
Adjusted R2       0.905634      0.903969
MAE              11.405065     11.274745
```

As the no of components increases performance slightly get improved.

When we compare training and testing metrics, we can see that they are almost the same and there is not much difference. This is because the data is quite large, also complexity of the model is accurate. Also, shows that methods, models and metrics and evaluations used are good enough.

PART G: L1 and L2 Regularization

L1 - Lasso and L2 - Ridge

Results For L1:

	TESTING DATA
MSE	263.356256
RMSE	16.228255
R2	0.923112
Adjusted R2	0.922969
MAE	10.519545

Results For L2:

	TESTING DATA
MSE	266.076842
RMSE	16.311862
R2	0.922318
Adjusted R2	0.922173
MAE	10.640248

Used label encoded data from part c for this part. Then used Lasso() and Ridge() from sklearn to perform this.
On comparing, we see that performance for both the methods are almost similar and also does not improve the results from part c.
This shows that none of this is useful and data is not overfitted.

PART H: SGDRegressor:

Results:

	TRAINING DATA	TESTING DATA
MSE	4.209948e+28	4.184357e+28
RMSE	2.051816e+14	2.045570e+14
R2	-1.236362e+25	-1.198132e+25
Adjusted R2	-1.238669e+25	-1.200367e+25
MAE	1.782327e+14	1.764734e+14

Used SGDRegressor from sklearn on label encoded data from part c.
We see that performance is very very poor. Error is very high and R2 scores and very less. This is because SGD is not optimal many times. To save time and computations it does not reaches optimal performance.