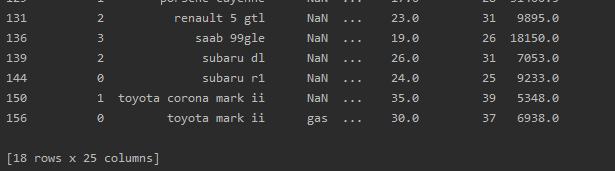
Milestone1 Machine Learning Project Report

Preprocessing techniques :

The dataset was having a missing values so we have two ways of preprocessing techniques

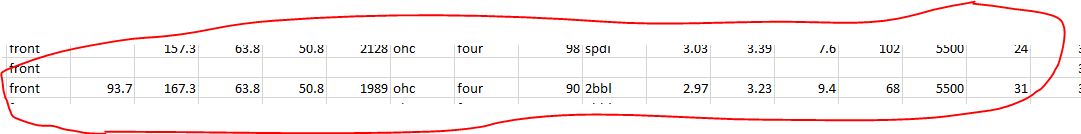
1. Removing all the rows which Have missing values
2. Trying to Fill It with some values and removing rows which has a lot of missing fields
3. Normalizing Ranges Between Data To Reach Global Minimum Faster

But the data set is very small(159 Rows) If we have removed all the rows that will affect the training and testing samples which will decrease Their sizes.



As we see in the dataframe representing the dataset where null values exist in 18 rows which is relatively high as compared to the size of dataset of 159 rows

We removed rows which have a large amount of null values as this row having 15 empty fields



The missing values can be filled manually by giving it a meaning value for example we can fill it with the average of the column values



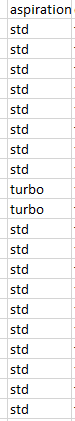
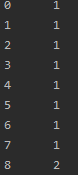
Some values are filled with label encoder as well

After making sure we haven’t any null values we applied the Encoder Which converts the categorical values to numerical values for regression we have used two encoders

1. Label Encoder( all columns except column named fueltype)
2. Hot Encoder (fuel type column)

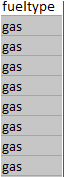
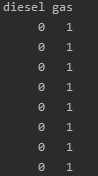
Label Encoder :

Label Encoder is giving each class in the column a random number which is specific for that class

Hot Encoder :

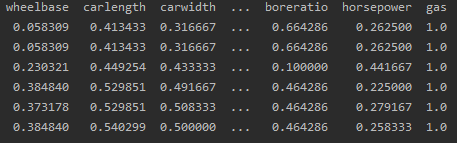
Hot encoder is giving each class a specific column specific for that class and the rest will be zero so if the value of the field is gas so it will have 1 at gas column and it will have 0 value in diesel its advantage is that the result is a binary value better so its good for high computations but it uses a large space for increasing number of features

Features Scaling:

We know that feature scaling affect the accuracy of the model to reach global minimum faster so we have used normalization of each feature to assure that data ranges form each other we have used range(0,1) and here is the data after using normalization equation (Xnew = Xold-min(X))/(max(X)-min(X))\*(b-a)+a

Where range is[a,b] and here is the data after using feature scaling



Data Analysis:

We Have Applied Data Analysis on Features Because it’s a very important step which can cause overfitting in the models and also to understand the relation between the independent variables and the Label Value which is the price of the car also we took the absolute value of the correlation so we can get the +ve and -ve correlation as we see in the next figure the top features of our model are

1)horse power

2)bore ratio

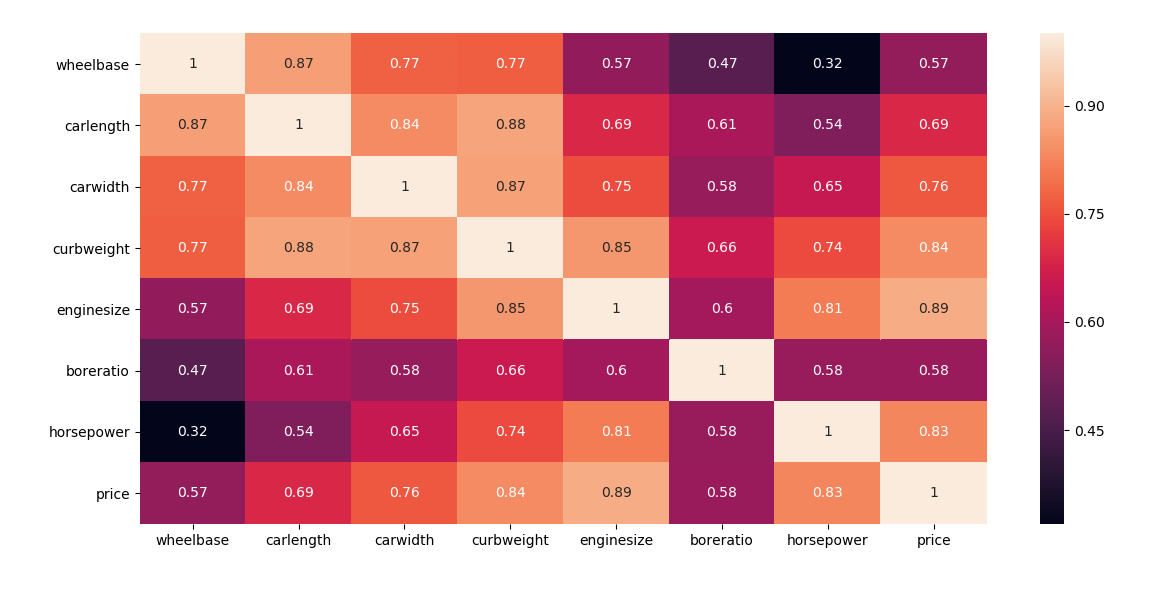
3) car length

4)car width

5)curb weight

6)engine size

7)wheel base



So After we have seen this correlation matrix for example the horsepower feature affect the price by 83% , enginesize affect by 89% and so on

Regression Techniques Used :

1)Multiple Linear Regression

2)Polynomial Regression

1) for Multiple Linear Regression :

The multilinear regression using gradient descent tries to get the global minimum using gradient descent trying to get the best slope for each feature and interception and its dimension = number of features + 1 to make best plane or shape fitting the data points

Accuracy :



Accuracy Of Training = 80%

Accuracy Of Testing = 87%

MeanSquaredError :



Mean Squared Error = 12170401.33

Training Time :



Training Time = 0.003995 seconds

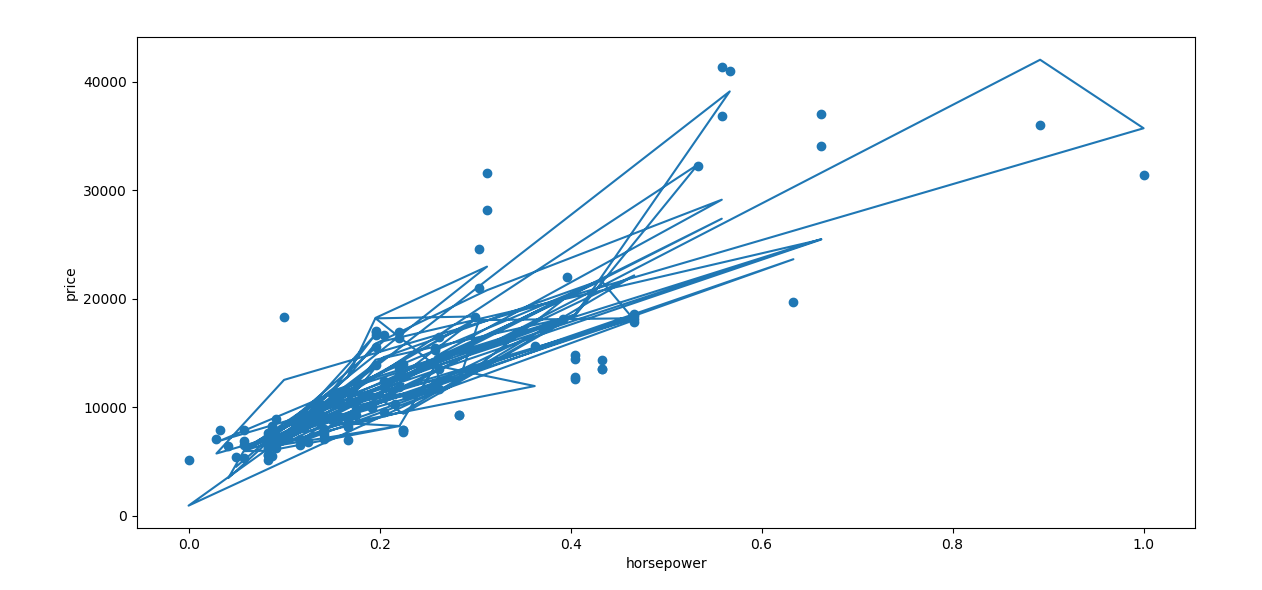
Prediction For a Sample :



True value for car = 7259

Predicted value for car = 7508.49

Plotted Curve :



We have taken one feature (HorsePower) to represent some of it in 2d to be able to visualize it with price

For Polynomial Regression :

The Polynomial Regression Is pretty much closer to multiple linear regression but it has more features than the multiple linear as its having for example the square of some feature such as (X1)2 which is called polynomial features and it depends on the degree of feature and also number of features to fit the best curve to fit data points

Accuracy :





Accuracy Of Training = 94%

Accuracy Of Testing = 71%

MeanSquaredError :



Mean Squared Error = 169469066.13

Training Time :



Training Time = 0.004997 seconds

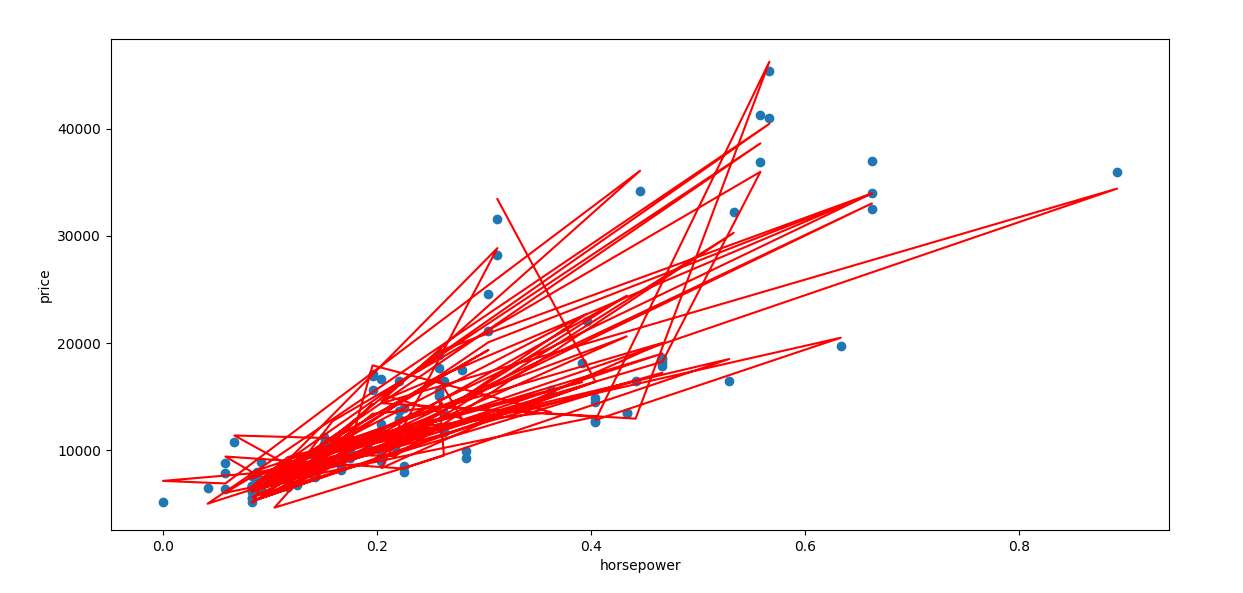
Prediction For Sample :



True Value of car = 7053

Predicted value = 1776.0

Plotted Curve:



Features Selected :

The Features We used depending on correlation matrix 8 features 7 from them which affect the price directly and another categorical feature for fueltype which was hot encoded to get only one feature

So they are

1)horse power

2)bore ratio

3) car length

4)car width

5)curb weight

6)engine size

7)wheel base

8)gas or not gas used in the car which is hot encoded

Training And Testing Datasets :

We have used 70% of the data as training samples and 30% as testing samples of the size of dataset which is (158 rows x 8 columns)

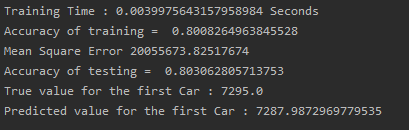
70% training = 100 rows

30% testing = 48 rows

Techniques Can improve answer :

We can improve answers by using Polynomial Lasso Regression Which is using Regularization Factor can prevent overfitting which normal polynomial regression suffered from and its results made a big difference and accuracy much better

Lasso Polynomial Regression Results :



Training Time = 0.0039 Seconds

Accuracy Of Training = 80%

Accuracy Of Testing = 80%

MeanSquaredError = 20055673

True price of car = 7259

Predicted value for the same car =7287

Conclusion :

Before Project we was thinking that polynomial regression is best for everything but it doesn’t at many different samples it may give a very low accuracy and that because its features is increased because of polynomial features while by using lasso polynomial regression it has improved that through increasing number of iterations and preventing coefficients from being too large while multiple linear regression is more stable for many cases and we have concluded that each problem has its own way of dealing with it and there is nothing which is best for every problem at first we used all features and we have got 99% Training accuracy but a very low accuracy in testing because of over fitting and also on trying to increase degree of polynomial overfitting took place so we tried to decrease the features to avoid it and increasing number of samples so model can learn more

Milestone2 Machine Learning Project Report

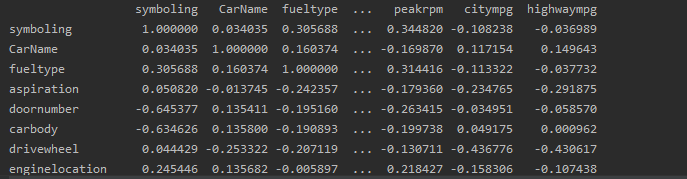
Preprocessing Techiniques:

We have changed some things in the previous code of milestone1 and applying it in milestone 2 ie. there was a row which was dropped at all for specific threshold we tried to fill each field by knowing its column average and filling these null values

We Also Have Used Label Encoder to fill Categorized Data

Feature Engineering And Data Analysis:

in Regression Models We Have Used Only The Top Features As We Used Correlation Technique For The Target But This Time We Decided To Use Another Technique Which Is Correlation But In Pairs Which Means Correlation Between Features To Really Know The Importance Of Each Feature From Correlation Matrix Values Which Are Higher Than Specific Percent By Negative Correlation Or Positive One



After Applying This Technique In The Correlation Matrix Getting Each Indices Of The Names Of Features Which Affect Our Model We Have Found That There is 18 Features Out Of 25 Ones Which We Will Use As They Are Much Important:

Features Selected :

1)enginesize

2)horsepower

3) highwaympg

4) symboling

5) fueltype

6) drivewheel

7) fuelsystem

8) boreratio

9) carheight

10) doornumber

11) carbody

12) carlength

13) wheelbase

14) cylindernumber

15) curbweight

16) citympg

17) compressionratio

18) carwidth

Models Used :

We have used Logistic Lasso Regression , One vs One Support Vector Machine and Decision Trees In our classification

1. Accuracy Graph Bar
2. Training Time Graph Bar
3. Testing Time Graph Bar

Hypertunning Parameters For Models:

1. Logistic Regression Parameters Changing

# 1)Penalty Parameter

we have many parameters in the logistic regression function first we will use is the penalty parameter

we have two types of penalties l1 and l2 which are terms of the regularization parameter to prevent the overfitting in our model giving Additional sum of weights(Lasso Regression) or sum of weights squared(Ridge Regression)

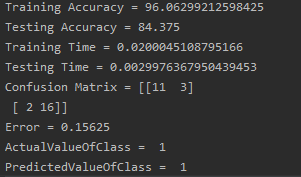
l1 => Lasso Regression

l2 => Ridge Regression

none => No Regularization In The Model

we will show results without regularization first by Adding Term none

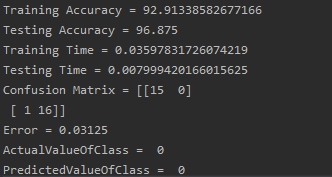




As we see the accuracy at testing is much lower than training which may cause overfitting

The solver parameter must be used as lbfgs while we has no regularization because by default its liblinear So we will se results after adding regularization parameter and if accuracy increased or not?

We will use l1 which is lasso regression parameter



As we see here the accuracy became better and error became much lower than before and Testing accuracy became better than training one

### 2)Constant C Value

We now will try to fix the penalty to (l1) and will try to change the C

value which is considered the measurement or controller of

regularization meaning it make the Regularization term whether

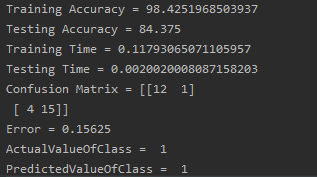
useful or useless by increasing value of it so we by that making the loss

function part more important than the Regularization part



As we see C by default have value =1 when we will increase it to 10

And then see results



As we see the accuracy became much lower in testing because we have

made our model to make more importance only to the loss function part

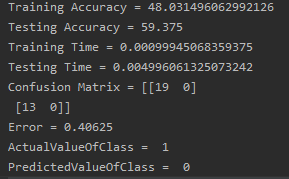
more than the regularization part what about making C value having a

very low value



Making C = 0.00001 means we want the value of regularization

parameter to takeover the loss function part lets see the results



As we see here the training accuracy is very low than testing accuracy

which means that the model hasn’t learnt anything and it even predicted

wrong class for a specific input because of underfitting

2)Support Vector Machine

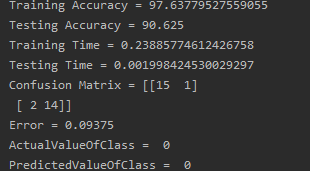
1)Degree

Lets try to change the kernel in the svm function which makes the model

More complex by moving it to another dimension by increasing in its

features Firstly we used kernel as linear lets see results



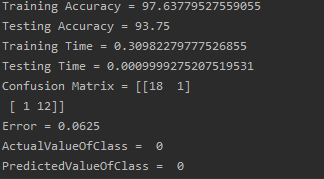


As we have seen before the linear kernel of svm by default is (ovr) which

means one versus rest classifier lets try to make it (ovo) which means one

versus one classifier and see results



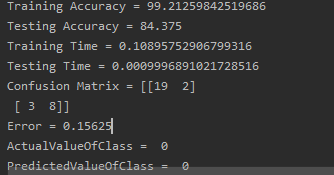


As We have seen the ovo svm is better than the ovr because we have two

classes and not more than two so it have better performance

We will try now to use kenel of Poly now and of degree = 2





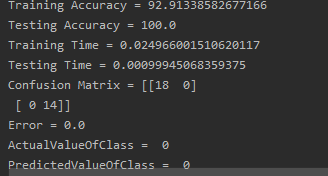
So when we made the model of degree too the model became predicting

true but its coming to overfit as training accuracy is so much higher than

testing accuracy by far

lets try poly of degree = 1





As we see its pretty much better ! as when the degree is lowered the

model performance have became much better

3)Decision Trees:

1) Criteria:

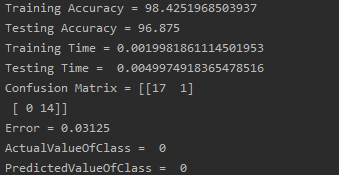
Lets Try See The Effect Of Changing Criteria Used on the model firstly

We will use Gini Index Method which uses an algorithm to measure

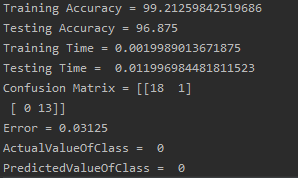
How often a randomly chosen element would be incorrectly identified

Which means an attribute with lower gini index would be preferred









As we have seen in the two outputs the accuracy is pretty much the

Same but it seems that the entropy method which uses the log function

To get information gain is more computational so it take much time in

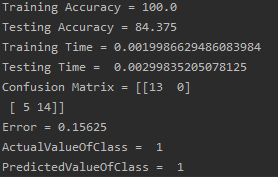
Testing than gini method

We will fix the criteria to entropy for now and change others parameters

2)Max Depth :

Lets try a very large depth of tree levels lets say max depth =100





As we see results the model is overfitting the training accuracy is known

Very well which make the model unable to generalize Data in Testing so

It gives lower testing accuracy

What about if we removed the value at all as we have seen when we

removed the max depth meaning not setting it with value it means ‘none’

value for max depth it will expand the nodes until nodes are pure and

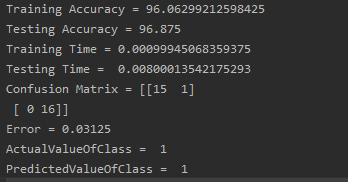
pure means higher information gain which means the randomness of data

decrease meaning that all the samples of specific feature have the same

label (according the question asked) at that node as well

now lets see if we have made it not low not high lets say max depth=3



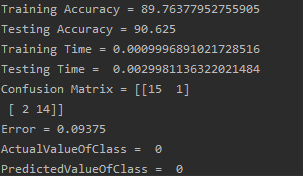


As we see its far better testing and training accuracy are nearly the same

And the error is low as well

What if we changed the depth =1 what it will affect





It seems that the very low depth affect the learning of the model which

May cause underfitting so we must be moderate in choosing complexity

Of our tree and here is plotting of the very low depth tree of 1 level only

