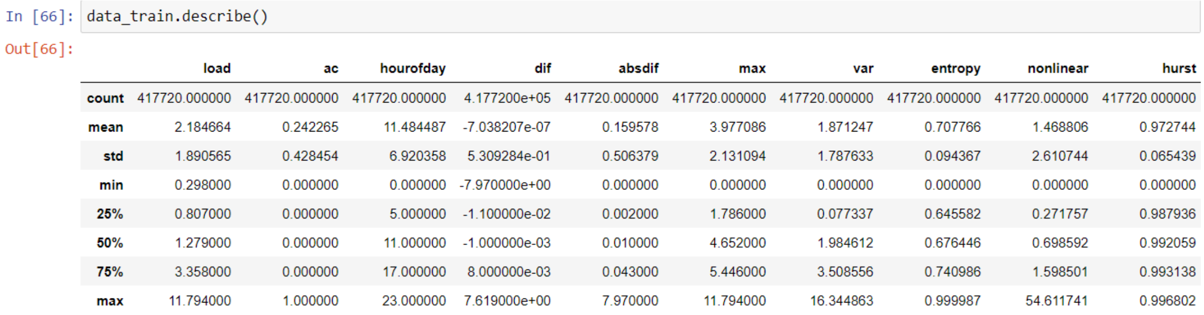
1. I have usedpandas read csv to read the train and test file. I have explored the datatype of the features and the targeted column.
   Here is the summary of the statistical values of the dataset-



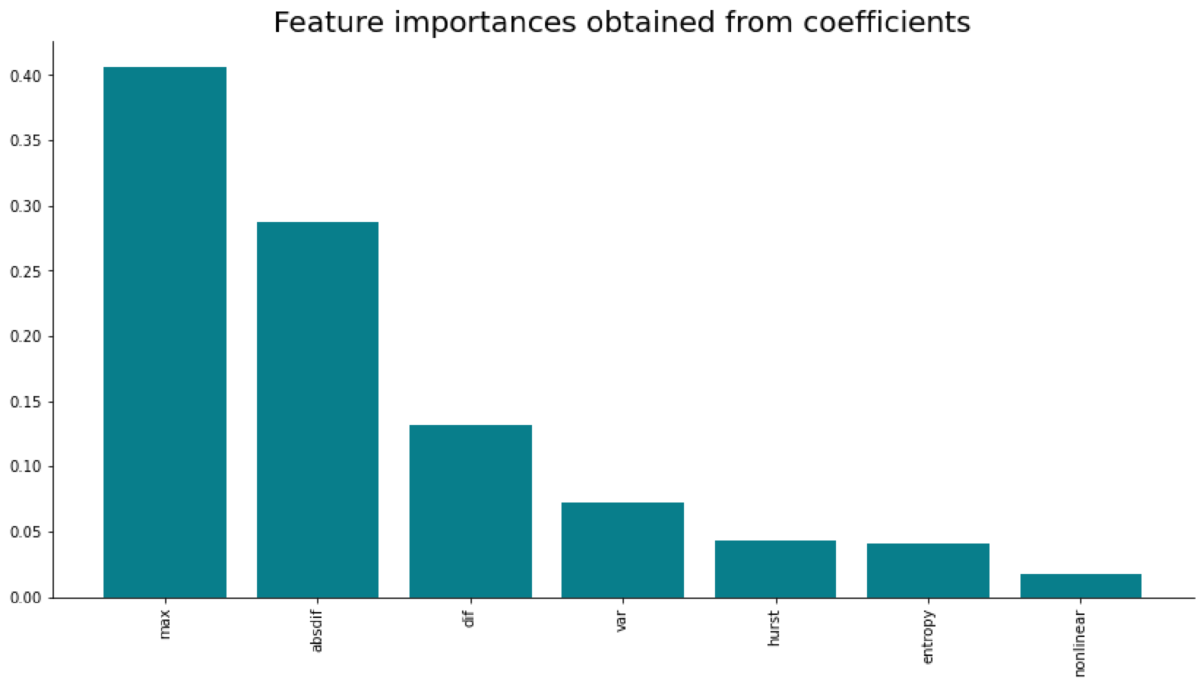
```
In [66]: data_train.describe()
```

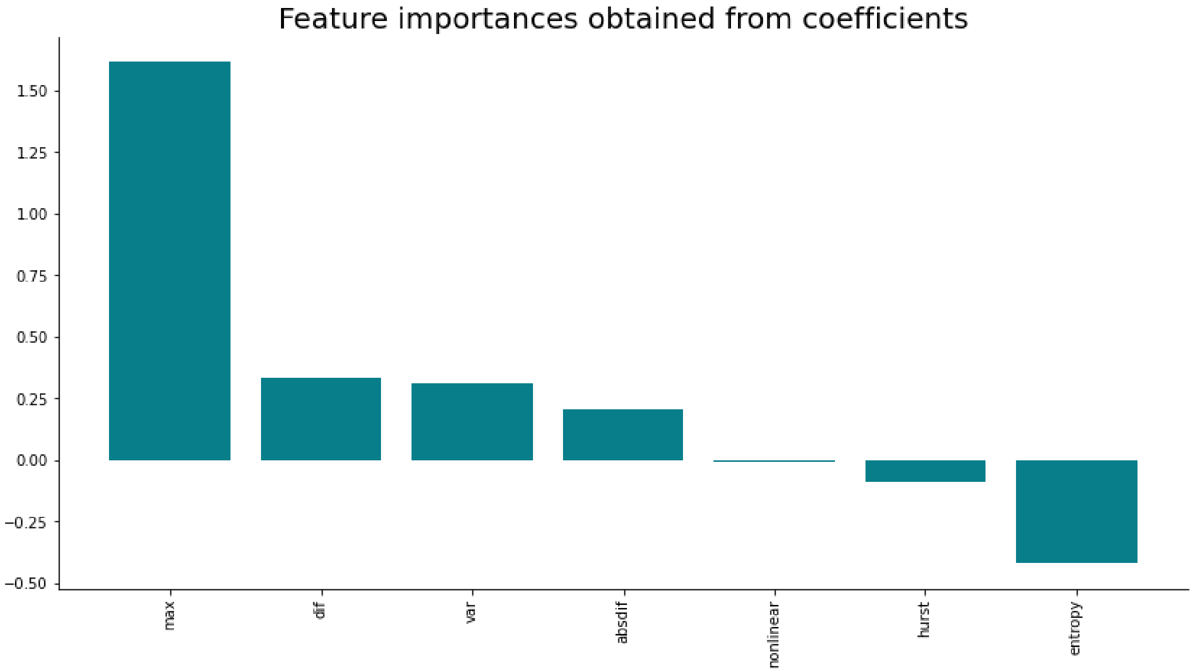
Out[66]:

	load	ac	hourofday	dif	absdif	max	var	entropy	nonlinear	hurst
count	417720.000000	417720.000000	417720.000000	4.177200e+05	417720.000000	417720.000000	417720.000000	417720.000000	417720.000000	417720.000000
mean	2.184664	0.242265	11.484487	-7.038207e-07	0.159578	3.977086	1.871247	0.707766	1.468806	0.972744
std	1.890565	0.428454	6.920358	5.309284e-01	0.506379	2.131094	1.787633	0.094367	2.610744	0.065439
min	0.298000	0.000000	0.000000	-7.970000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.807000	0.000000	5.000000	-1.100000e-02	0.002000	1.786000	0.077337	0.645582	0.271757	0.987936
50%	1.279000	0.000000	11.000000	-1.000000e-03	0.010000	4.652000	1.984612	0.676446	0.698592	0.992059
75%	3.358000	0.000000	17.000000	8.000000e-03	0.043000	5.446000	3.508556	0.740986	1.598501	0.993138
max	11.794000	1.000000	23.000000	7.619000e+00	7.970000	11.794000	16.344863	0.999987	54.611741	0.996802

   Then some columns were unnecessary. I have dropped those column. I have dropped-hourofday,load,dayofweek. I have checked whether there is any null value ini the dataset and I got false.

2. I have analyzed the features using two techniques. One is coefficient based and another is tree based method. Here is the results I have got. From both the graph we can see, max has higher feature importance.





3. I have used KNN,Decissiontree,Logisticregression for this task. Here are the performance metrics of all the three models.
   KNN-

```
precision    recall  f1-score   support

        0       0.94      0.93      0.94     96221
        1       0.36      0.38      0.37      9319
    accuracy                         0.88    105540
   macro avg       0.65      0.66      0.65    105540
weighted avg       0.89      0.88      0.89    105540

[[89851  6370] [ 5769  3550]]
```

```
Decision Tree-
precision    recall  f1-score   support

        0       0.95      0.93      0.94     96221
        1       0.38      0.46      0.42      9319
  accuracy                          0.89    105540
 macro avg       0.66      0.69      0.68    105540
weighted avg       0.90      0.89      0.89    105540
[[89214  7007] [ 5035  4284]]
```

```
LogisticRegression-
precision    recall  f1-score   support

        0       0.93      0.95      0.94     96221
        1       0.39      0.31      0.34      9319
  accuracy                          0.90    105540
 macro avg       0.66      0.63      0.64    105540
weighted avg       0.89      0.90      0.89    105540
[[91728  4493] [ 6453  2866]]
```

For decision tree I have used maxdepth=2,criterion="gini" and splitter best.
For knn, I have used num_neighbour=9, weights=uniform
For logisticregression, I have used solver lblgf.
I have choosed this metrics after some experiment and hyperparameter tuning. I did not use the complex parameters as that will introduce overfitt and as there is good ,amount of data so It was ok to avoid overfit and there is less chance of underfit.

4. I have used Adaboost, Gradientboost and extra tree classifier for this portion. Here is the performance metrics

```
Adaboost-
precision    recall  f1-score   support

        0       0.95      0.93      0.94     96221
        1       0.41      0.48      0.44      9319
  accuracy                          0.89    105540
  macro avg       0.68      0.71      0.69    105540
 weighted avg       0.90      0.89      0.90    105540
[[89714  6507][ 4814  4505]]
 GradientBoost-
precision    recall  f1-score   support

 0       0.95      0.93      0.94     96221
 1       0.41      0.49      0.44      9319
 accuracy                          0.89    105540
   macro avg       0.68      0.71      0.69    105540
weighted avg       0.90      0.89      0.90    105540
[[89609  6612] [ 4786  4533]]
```

Extra Tree classifier-

```
precision    recall  f1-score   support

           0       0.94      0.96      0.95     96221
           1       0.45      0.36      0.40      9319
 accuracy                           0.90    105540
   macro avg       0.69      0.66      0.67    105540
weighted avg       0.90      0.90      0.90    105540
[[92021  4200][ 5939  3380]]
```

Adaboost is not flexible for loss functions. It has a fix loss function. On the other hand, Grdaientboost is adaptive to various loss functions and it can use any differential function as loss function. Adaboost is flexible to outliers but gradient boost is flexible to dense samples. From the result, we can see that for 1 sample, boosting techniques perform much better. As this was an unbalanced dataset so to detect the lesse sample, boosting techniques work much better. Definitely it is possible to use other tree algorithms like extratree classifier. Ext is an excellent ensemble method and I have used it. It gradually works in the weak learner and try to improve the performance.