1. Make some initial data analysis

Step 1:
Found out label distribution- Highly imbalanced

{'C': 46882, 'D': 9279, 'B': 6602, 'E': 2507, 'A': 867}

Step 2:

Found spurious columns - a columns which doesn't not add any value to the model

Following set of columns have same value for all data points

[59, 179, 268, 269, 270, 271, 272, 273, 274, 275, 276]

Step 3:

Build a covariance matrix - compared with one vs one feature at a time.

Following set of features/columns pairs are highly correlated (more than 70%)

{4: [44], 6: [45], 64: [294], 78: [285], 177: [257]}

Step 4:

Removed the spurious columns(i.e found on step 2)

Step 5:

Sometimes it's better to remove the highly correlated features, in order to decide which column to remove have used **CHI2 metric**

Step 6:

Normalized the data to between 0 and 1 Values. Used sklearn Min_max_scalar

Selected top 100 features/columns

Step7:

Neural Network
Have used 2 layer perceptron : with first hidden units of 60 and second layer of 15.
Last layer using Softmax classifier

Used **weighted cross entropy loss** as loss function Adam optimizer for optimizing the loss function
Used **F1 score** as performance measure - because of imbalance dataset

Results:

Performance measure (On 25% of unseen data):

Accuracy :
0.72125793770789237

F1-Score:
Macro : 0.167611819683

Neural Network always converged to head class i.e C (which has max number of data points)

Conclusion

Even though accuracy is good , but the learnt model is of no use since it's always predicting the same label.

Future Work:

1) Threshold tuning may give better model
2) Handling imbalanced data either by oversampling or undersampling (can use SMOTHE)
3) Better feature selection
4) Even after all this, if the model performance measure is not improving, the labels may be randomly assigned