**Analysis**

**About dataset**

The dataset had 2041 columns and 400 rows in which last column was classes

Got brief insight into the data set using describe() function.

The number of classes are 2.

201 samples belongs to class 1 and 109 classes belonged to class 0. The dataset is almost balanced.

**Data Preprocessing**

1. Counted the number of Nan Values = 3 rows, 1 belonged to class 0, rest 2 belonged to class

2. Next step I did was to remove Nan values as there was only 3 rows.

3. Split the dataset to features and labels.

4. There are 2040 features. So Next was to select best features.

5. For this, I used SelectPercentile from sklearn where I used ANOVA F-value for scoring as it is better suited for classification problems

6. I selected top 1% features. Now we only have top 21 features.

7. Features and labels are split into train and test. We have 20% of features and labels for testing. I used stratify = labels to maintain the proportion both classes equal in train and test set.

**I chose conventional machine learning techniques over deep learning as the dataset was small and balanced. Handcrafted rules would be able to perform better in these cases rather than neural networks.**

**Find Suitable estimator selection for classification**

I used 3 estimators, logistic regression, support vector machine and Gaussian naïve bayes as these estimators tend to perform very well on classification task.

My 1st choice was logistic regression with liblinear solver. I used Ridge regularization(which is present as default) to reduce overfitting.

My 2nd choice was an svm with ‘rbf’ kernel as it could be used for nonlinear cases.

My 3rd choice was naive bayes.

**For both logistic regression and svm I used standard scaling as both each one is based on gradient descent and distance between data points respectively. For naïve bayes, I did not use any scaling.**

**Next step was to use cross validation with size k = 10 on training set, get the f1 scores and learning curve for all three models.**

From cross validation step, the svm was found to have slightly better f1 score and lower standard deviation than logistic regression and naïve bayes.

As next step I ran learning curves for all these models, and found that svm model had better learning curve for this dataset. The naïve bayes had worst learning curve.

So I further fine-tuned svm model using hyperparameter tuning with the help of GridSearchCVSVMclf. The parameters tuned were kernel, gamma, and C.

I ran cross validation and learning curve for the tuned and optimized svm model. This time the f1 score have increased significantly. From the learning curve, the f1 score seems to improve with increase in training size.

**To get a better picture of the model, I used box plots on each model. From the box plot on f1 scores, It seems that tuned svm model and logistic regression are best suited.**

**Testing and Limitations of model**

From testing on all the models, I found that the tuned svm model had slightly better f1 scores compared to all other models. But the model would improve with increase in training data.

The model did not achieve 0.9 f1 score on test set. The reasons are:

Few training samples and large number of features.

All the models seemed to had better training score compared with validation score in learning curve. This shows models might be over fitting.