Supervised learning – report

Description

In this lab we were asked to use 2 different data sets and apply vaious alghorithms to solve classification problems.

Python 3.11.6 was used for all scenarios below

Spotify the most streamed songs of 2023

I am not only a software engineer, but also a musician. I listen to Spotify for around 4+ hours a day. I decided to use Spotify report about the most popular songs in 2023 and try to predict number of song streams per month (with ML approach).

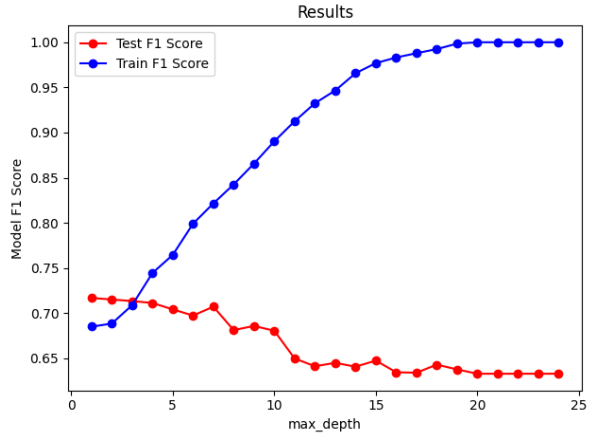
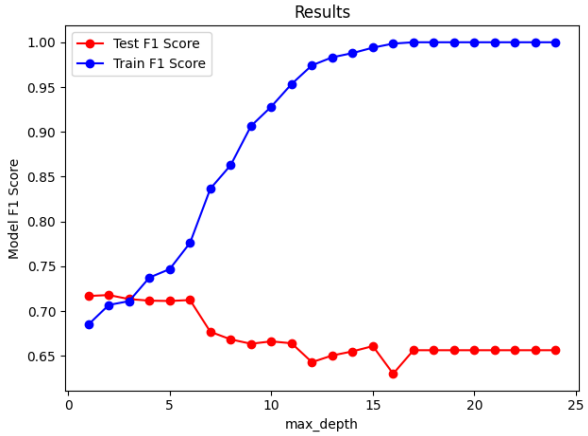
To make it a classification problem, I bucketized number of streams per month and tried to predict the bucket

Data was quite dirty and required preprocessing and adaptation to be used as classification problem.

At the end of the day these classes have been chosen based on monthly listens:
"'<1kk'","'1kk-25kk'","'25kk-50kk'","'50kk-100kk'","'100kk-500kk'", "'500kk+'"

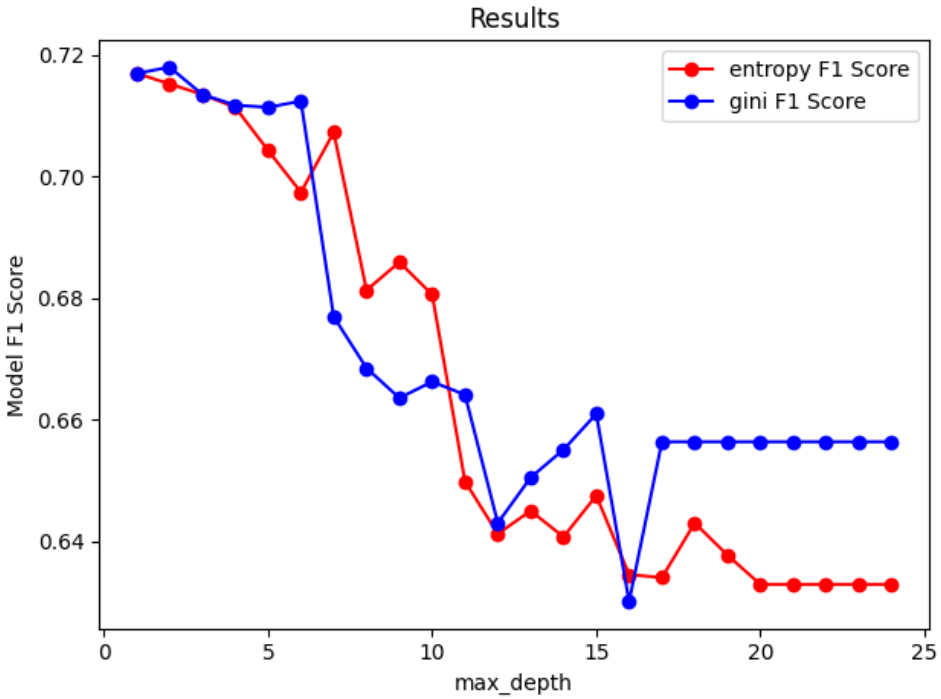
Decision tree classifier:

I decided not to prune anything and check what is the best max_depth would be for this problem using 2 different functions to measure the quality of a split. You can check the results below:

Gini	Entropy
	

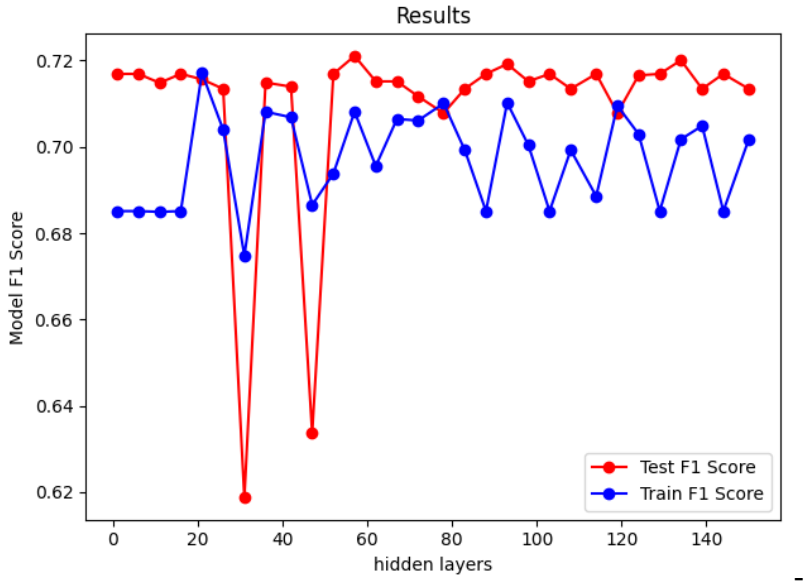
As expected, longer trees show better results on training data, but not on test set.

On that dataset test scores are better for gini, because according to results we will not use max_depth larger than 4-5



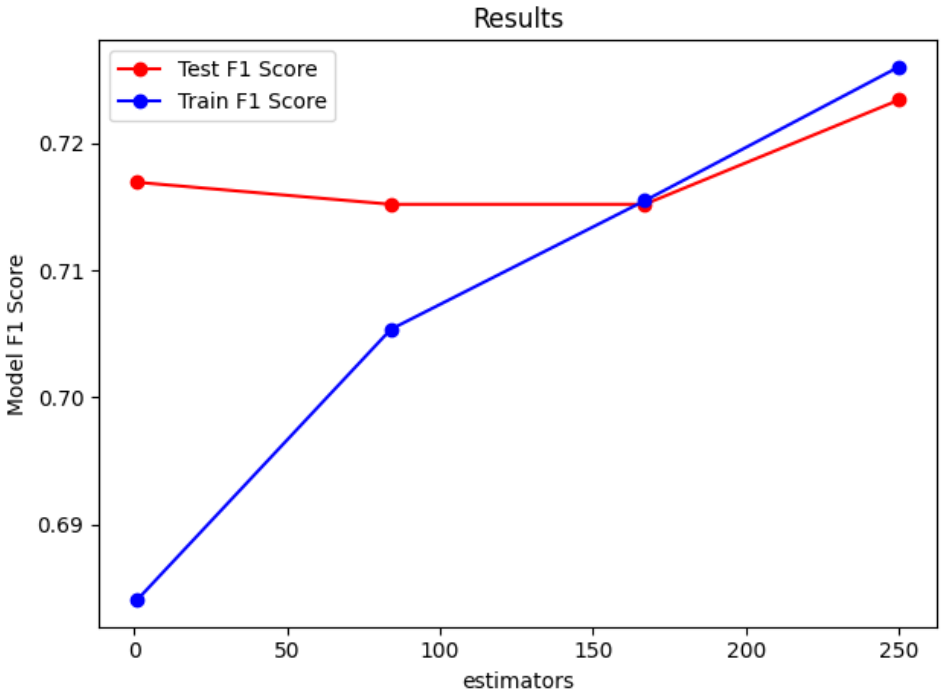
Neural network classifier

I used MLPClassifier from sklearn package. I tried to play around with a “number of hidden layers” parameter. But it seems like that dataset accuracy did not improve along with the number of layers.

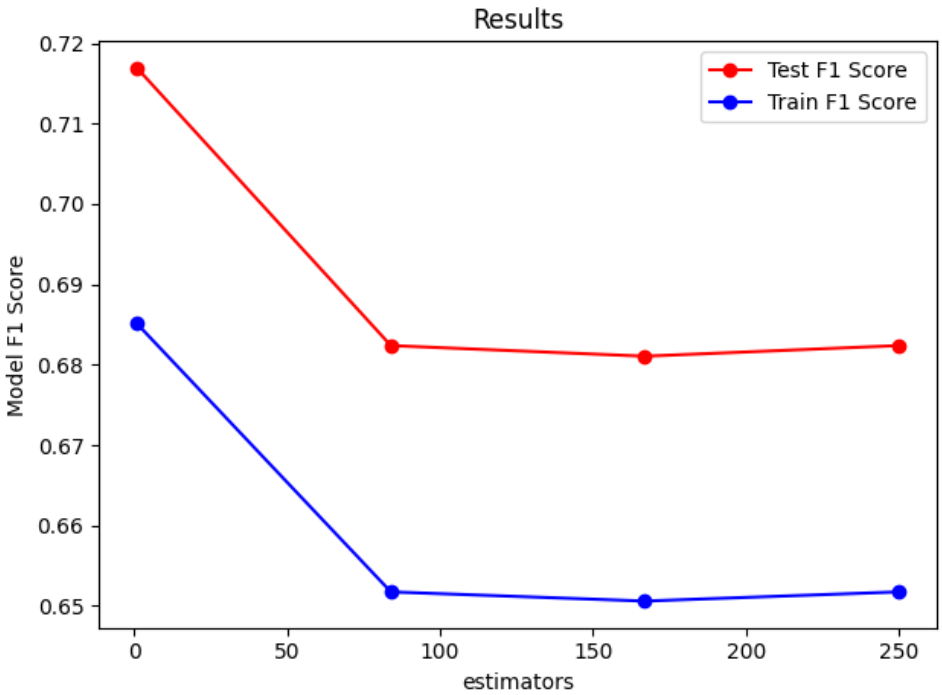


Boosted decision tree

I used GradientBoostingClassifier and changed number of estimators



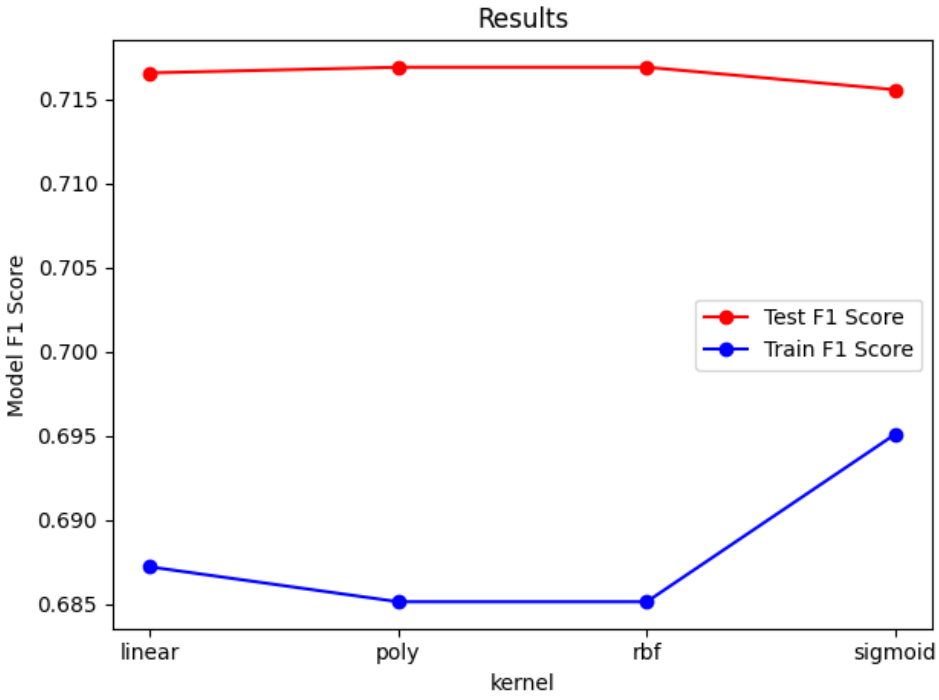
I also tried AdaBoost, but it showed worse results



SVM classifier

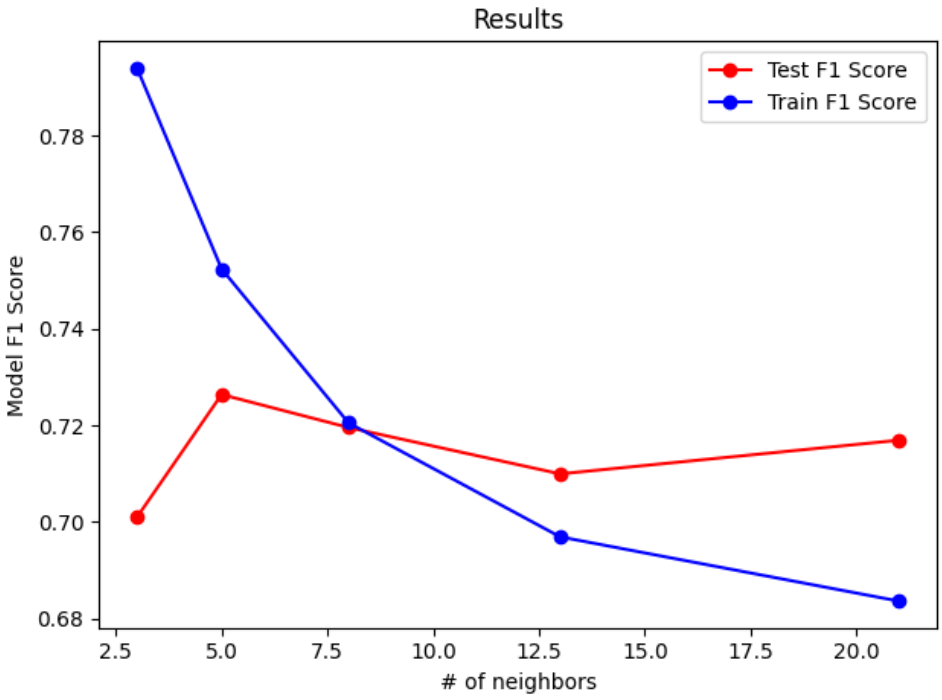
I used “support vector machines” classifier from sklearn package.

I tried different kernels, and it seemed like polynomial and RBF kernels showed the best results on the test set.



KNN

I used Fibonacci sequence to play with number of neighbors for this classifier. You can check results below:



Seems like on test set the best result is when I used small number (5 or large one 21)

Results:

Why did you get the results you did?

It seems like relying on music attributes is not enough. Artist name and popularity play a significant role in the popularity of the song. But to be honest I am quite surprised that it is possible to predict song popularity relying only on music characteristics. For that case I am quite happy with 0.72% accuracy.

What sort of changes might you make to each of those algorithms to improve performance?

I am sure it might be improved with more data and by diving deeper into hyperparameters

How fast were they in terms of wall clock time?

Because the dataset is not large, all classifiers worked in less than a minute. Although SVM appeared to be the slowest.

Would cross validation help?

I expect that cross validation will provide more accurate predictions and increase accuracy and f score. But for SVM it may take too much time to finish

How much performance was due to the problems you chose?

I chose 6 class classification problem with dataset of 953 rows. I expected error to be quite high

Which algorithm performed best?

The winner is… knn with n = 5. It showed the highest score of 0.7263827082008899. But to be honest, all algorithms performed with more or less the same value

How do you define best?

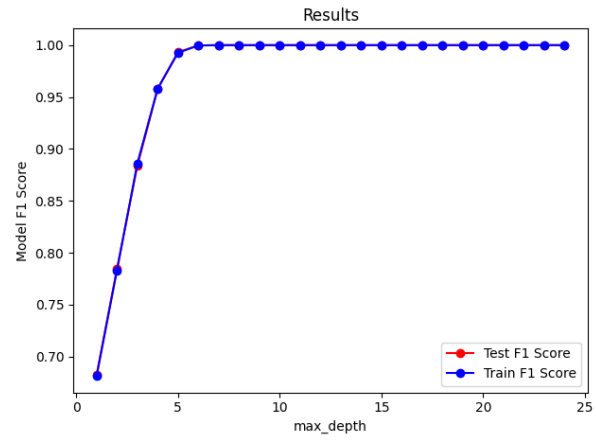
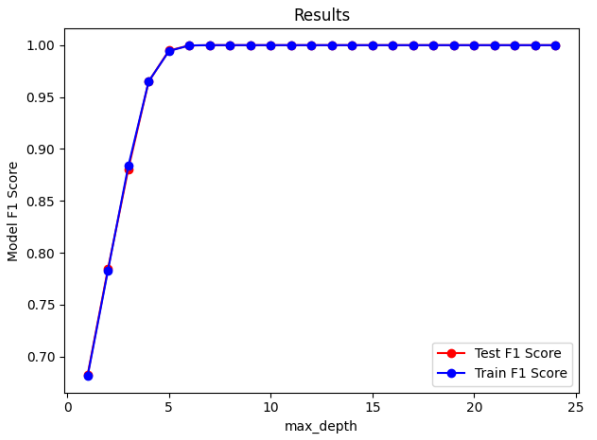
I used f1 score as key a metric

Credit card fraud

This is one of the most useful problems that engineers solve in commercial organizations. In my company another department tried to identify if the transaction was a fraud or not. I will try to do the same for this learning project. Column “fraud” represents a binray classification

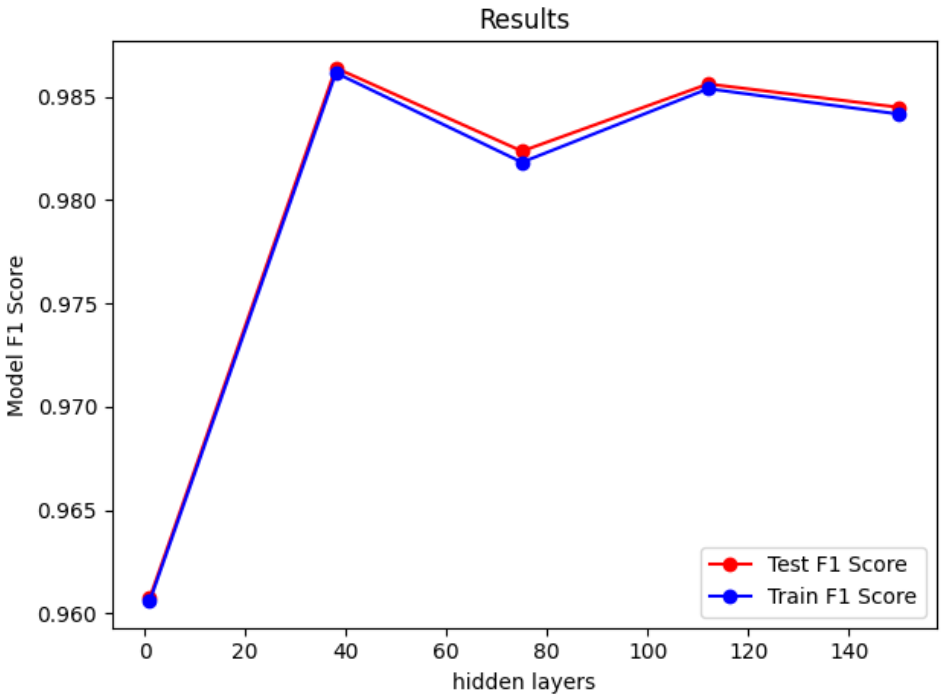
Decision tree classifier:

Dataset is larger and much cleaner than the previous one. As we can see results for train set are comparable with the test set for both gini and entropy

Gini	Entropy
	

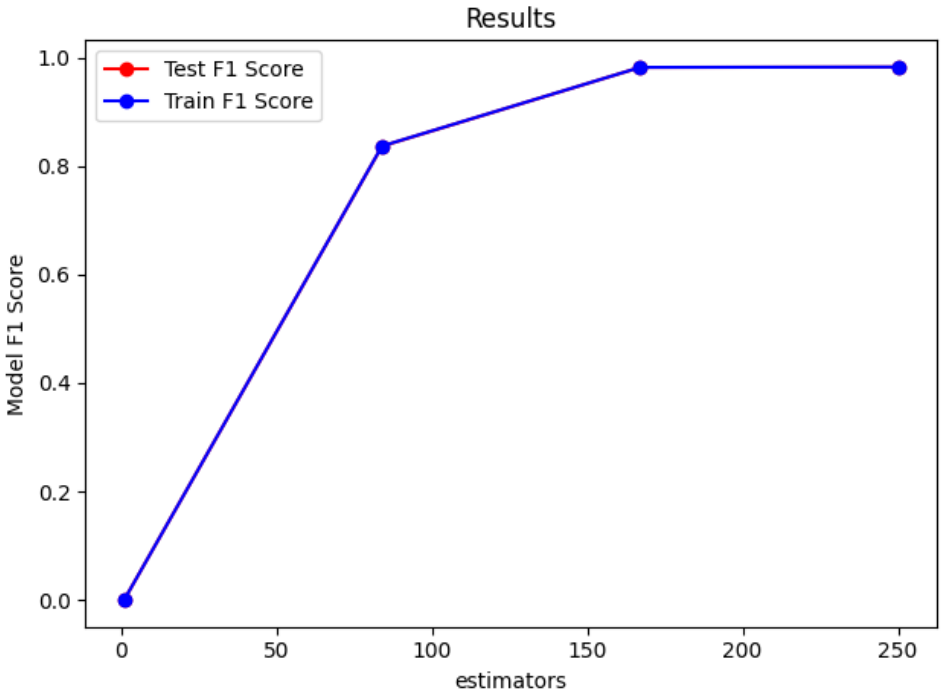
Neural network classifier

I used MLPClassifier from sklearn package. Number of hidden layers does not affect much after 40



Boosted decision tree

As we can see in the graph below after around 170 estimators f1 score stopped raising

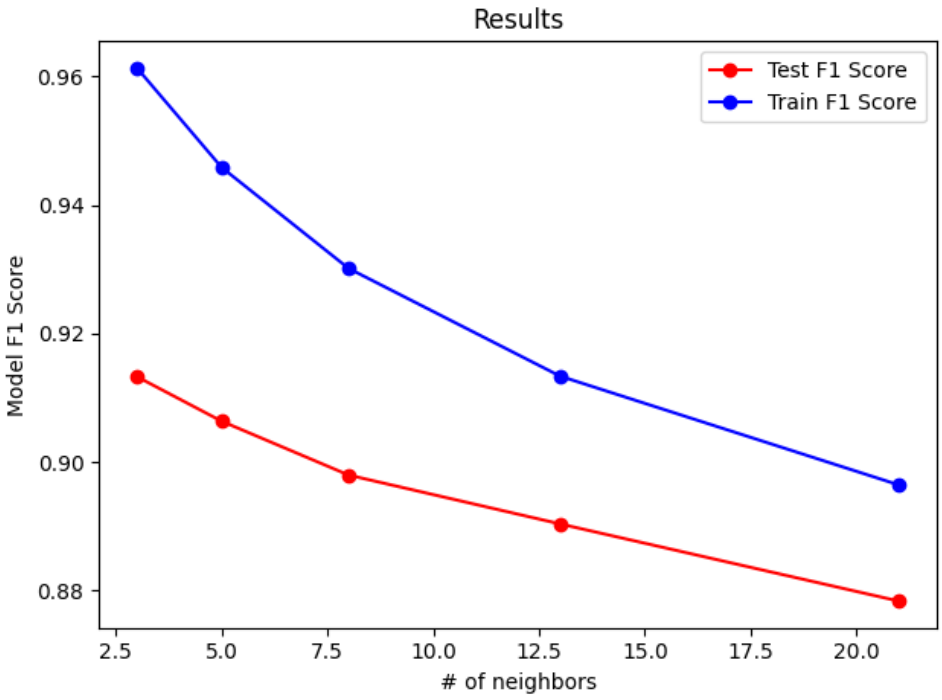


SVM classifier

I left it for the couple of hours but still wasn't able to get results in that time frame. I guess something more powerful is required or external graphic card will help, but for now, I was not able to get results for that dataset

KNN

I used Fibonacci sequence to play with a number of neighbors for this classifier. You can check results below:



Seems like the best results were achieved when I used 3 as a hyperparameter

Results:

Why did you get the results you did?

Dataset was clean and accurate. Also it contained attributes (like distance_from_last_transaction, and distance_hrom_home) that allowed classifier to make accurate predictions

What sort of changes might you make to each of those algorithms to improve performance?

Hyperparameters plays quite significant role for this dataset

How fast were they in terms of wall clock time?

Because this dataset is larger than the previous one, all classifiers worked much slower especially SVM that I could not get results from.

Would cross validation help?

I expect that cross validation will provide more accurate predictions and increase accuracy and f score. But for SVM it may take too much time to finish

How much performance was due to the problems you chose?

I chose binary classification for that problem and taking into count number of rows it worked quite well

Which algorithm performed best?

Decision tree and Boosted tree was able to achieve .9999 f1 score

How do you define best?

I used f1 score as key a metric