KAUNAS TECHNOLOGYS UNIVERSITY
INFORMATICS FACULTY

Module P160B114 „Machine Learning Methods“

Laboratory work report
Second laboratory work

Instructor
Doc. Iešmantas Tomas

Student
Rokas Puzonas

KAUNAS, 2023

Table of Contents

Task 1, Decision trees

Problem 1, Spotify dataset

1.1.1.

From the CP plot, we can see that picking an alpha value between 0.005 and 0.0075 would be the best. There exists a higher accuracy in the range 0 to 0.005, but that can be attributed to noise from the random seed by which the decision tree is split.

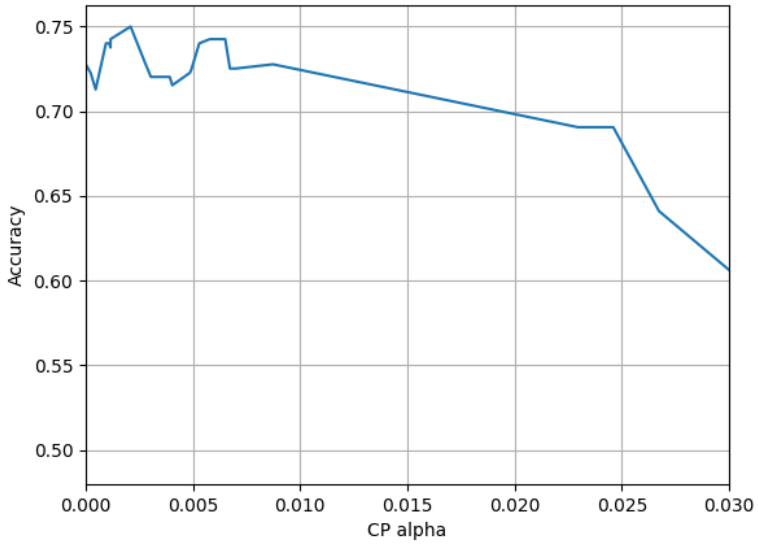

Figure 1. Accuracy vs. CP alpha

1.1.2.

I am using a CP alpha value of 0.006

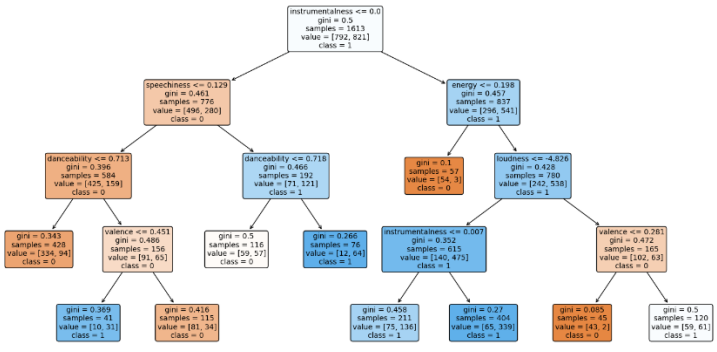

Figure 2. Prunned decision tree visualization

1.1.3.

The tree has 19 leaves. The 10-fold cross-validation error rate is 0.649007

1.1.4.

if instrumentalness <= 0.0 and speechiness <= 0.129 and danceability <= 0.713

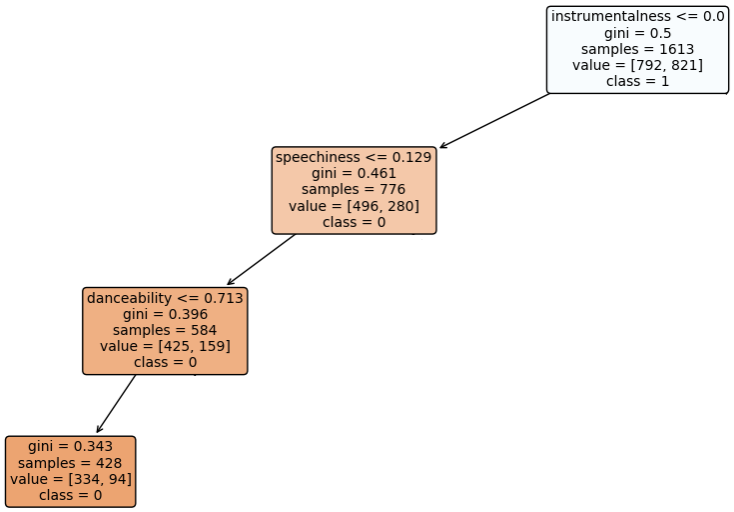

Figure 3. Selected classification rule

I believe that this rule is the strongest, because it correcly classifies the largest portion of testing data, that being 20.7%. Another factor that it has a relatively low gini impurity of 0.343 compared to all of the other nodes.

1.1.5.

The accuracy difference between the pruned and unprunned is ~4%, which is not neglible but also not high. But the more important difference is that the decision tree is not overfitted. This can be determined looking at the decision tree visualy.

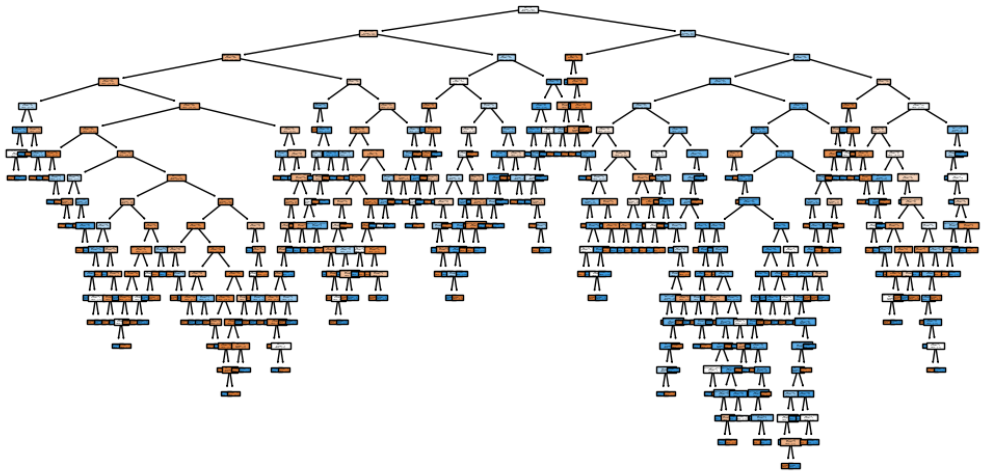

Figure 4. Overfitted decision tree

1.1.6.

My decision tree is a better classifier compared to the logistic regresion, because the accuracy of the decision tree is 74%, compared to the 60% in logistic regression. The reason for this is because all of the features are indepentant of each other.

1.1.7.

The accuracy does not suffer, but it had the opposite effect by raising the accuracy. The reason why is that most of the time for the unimportant features the noise is more likely to be picked rather then the correlation between that feature and the target.

By comparing the prunned and unprunned feature importance plots we can deduce that duration is not that important compared to the other features.

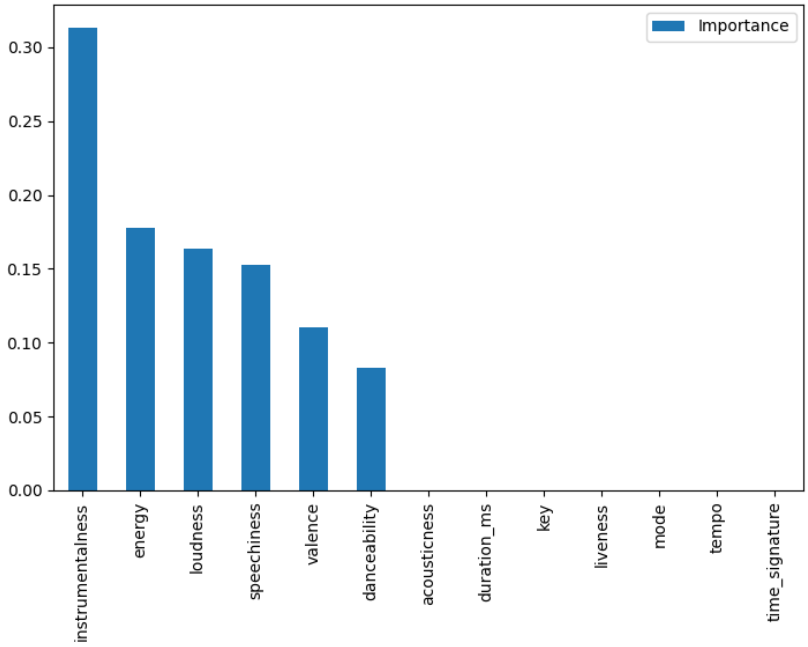

Figure 5. Feature importance (prunned)

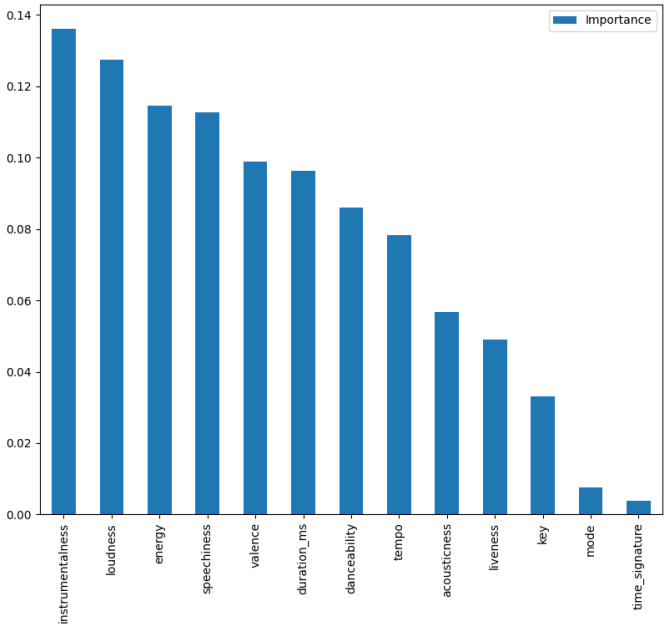

Figure 6. Feature importance (unprunned)

Problem 2, MNIST Sign language dataset

1.2.1.

A CP plot could not be generated in python, because it does not have an equivalent to „plotcp“ from R. Because for each CP alpha value, the decision tree needs to be retrained. Given

that it took ~22.5s to train a fully grown decision tree and with 2185 possible CP alpha values, it would take at most 13h.

1.2.2.

Because I wasn‘t able to create a CP plot in python, by trail and error I found a siutable CP alpha value which still can get a decent accuracy. The chosen CP alpha is 0.0009, this results in a 36.38% accuracy and a 10-fold cross-validation error rate of 0.836591.

The tree is most likely a bit on overfitted, but is difficult to say, because the dataset has a large number of features, so it should be expected that the decision tree will be large.

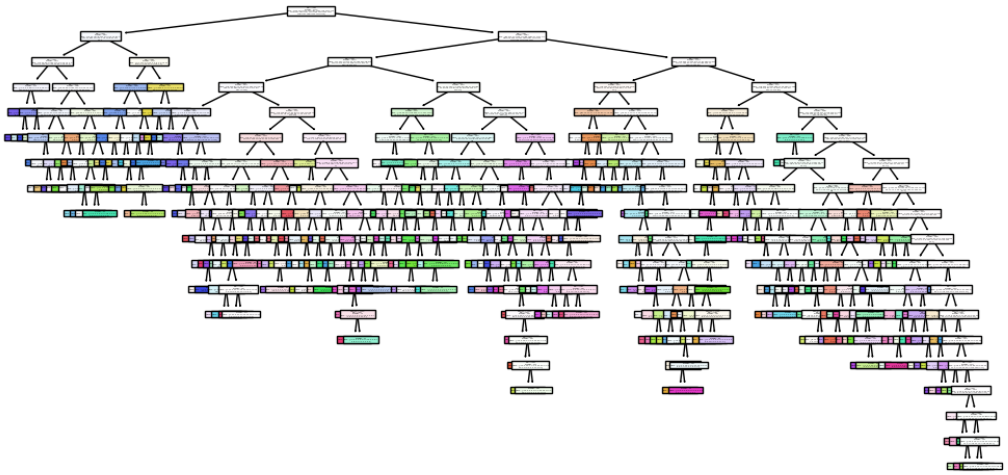

Figure 7. Likely overfitted decision tree

Another good CP alpha value that I found was 0.0012, with a lower accuracy of 33.73% and 10-fold cross-validation error rate of 0.813028. Looking at a visual representation of it, we can guess that it should be less overfitted with a more lenient decision boundry.

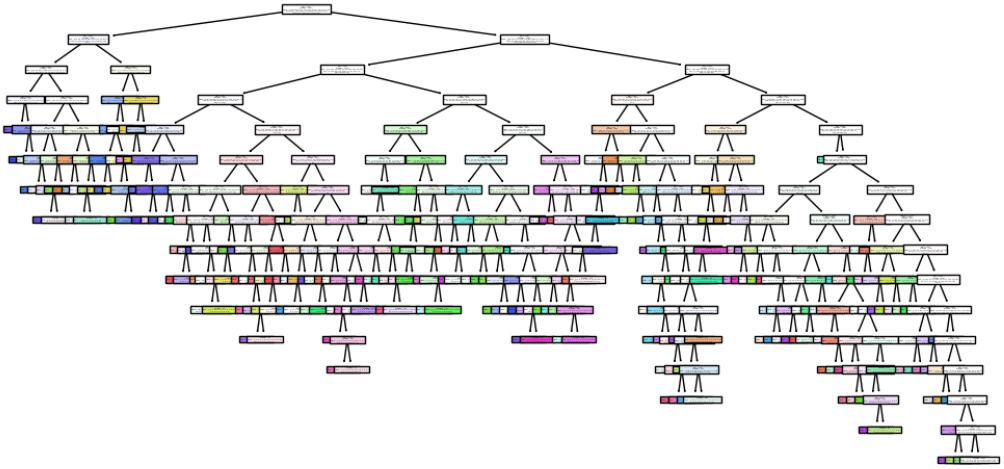

Figure 8. A bit less overfitted decision tree

1.2.3.

I can confidently say that overfitting has occured. Visually taking a look at the unprunned decision tree it is evident that it is a complete mess. The reason for this, because there is a large number of non-indepent features. Another reason is that in the dataset a small difference in a pixels values should not matter much, but given that deicision boundaries in decision trees are very strict, this results in a very deep ovefitted tree.

The accuracy of the unprunned decision tree is 42.93%, 10-fold cross-validation error rate of 0.90644.

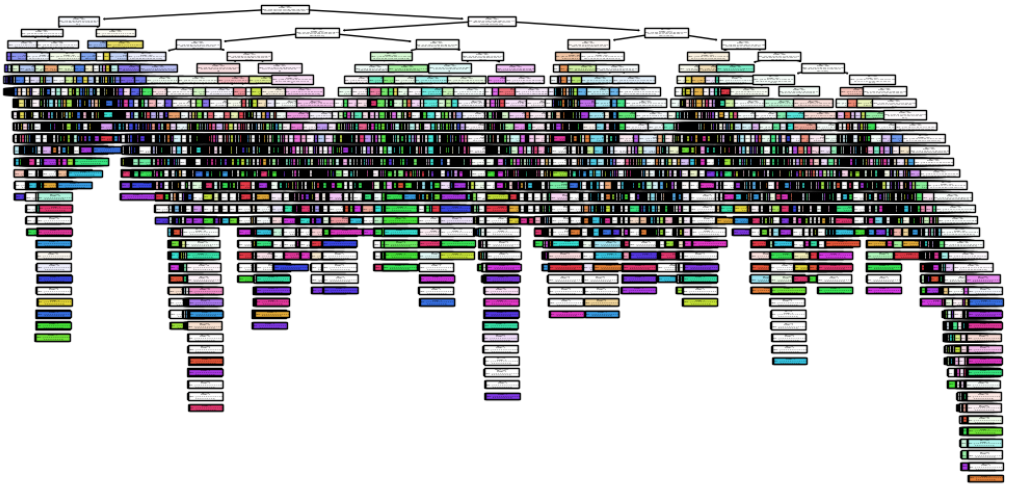

Figure 9. Unpruneed decision tree

1.2.4.

Deleting the same pixels as in the LDA/QDA lab assignment does not significantly increase the accuracy. But it does simplify the decision tree by limiting the splitting options, which makes the tree shallower and less likely to be overfitted.

1.2.5.

The decision tree is a worse classifer for this problem than the LDA/QDA method. I think the cause of this is:

1. Large number of features.
2. Features are related, neighbouring features most of the time have similar values.
3. A lot of noise, changing a single pixels values should not greatly affect the prediction.
4. Because the decision tree is looking at specific pixels, sliding all of the pixels to the right or slightly brightening the whole image can drastically change the trees decision.

1.2.6.

The if the importance of the unprunned tree is plotted like a bar graph, no useful can be learned from it. Because there are so many features and all of them have a similar importance.

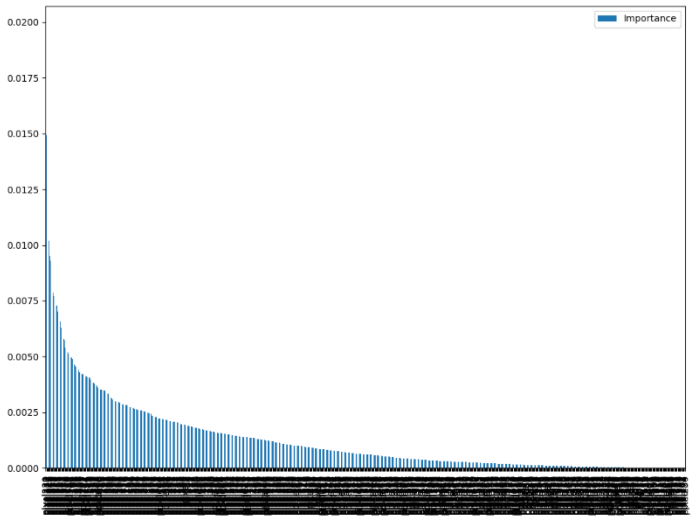

Figure 10. Useless importance graph

A better way to visualise the importance is to lay them out in a grid an assign a color to them dependign on their importance. This gives us a much better picture we can discuss. From this we can see that the tree only learned what the average position of the hand is in the image, but not what distinguishes each hand sign. The read area around the outside of the middle yellow are is the background.

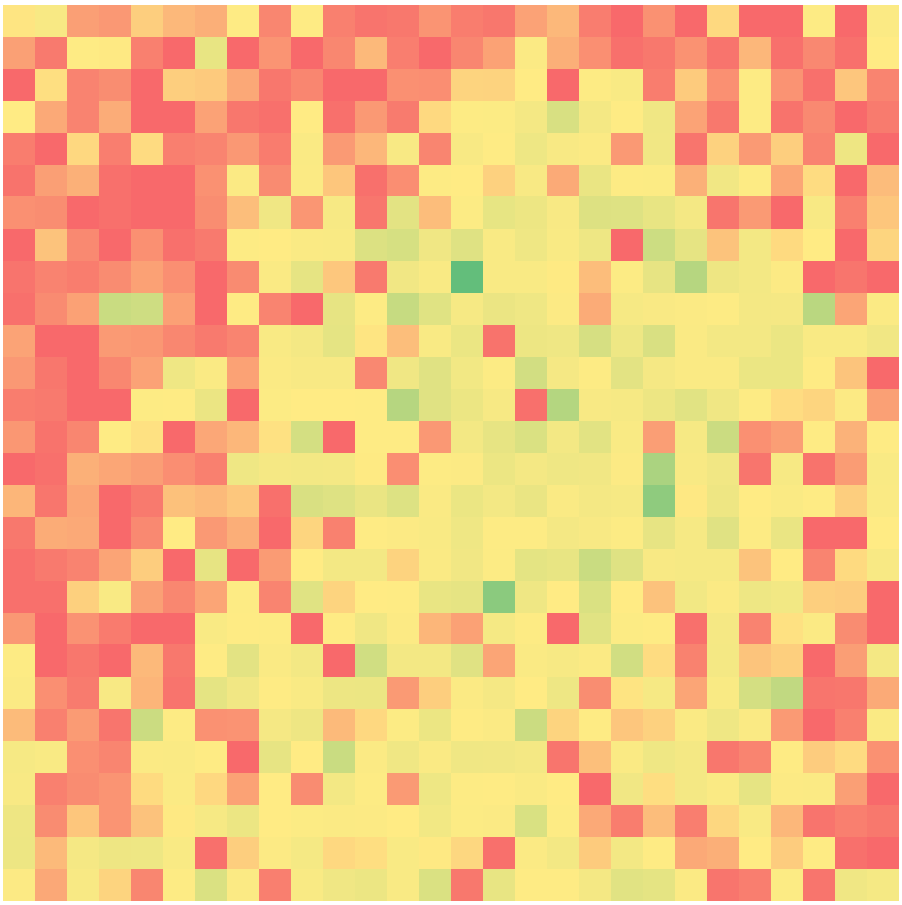

Figure 11. Importance visualised as a grid (unprunned)

Prunning does not help much, we can still see that it is very noisy and the importance is homogenous.

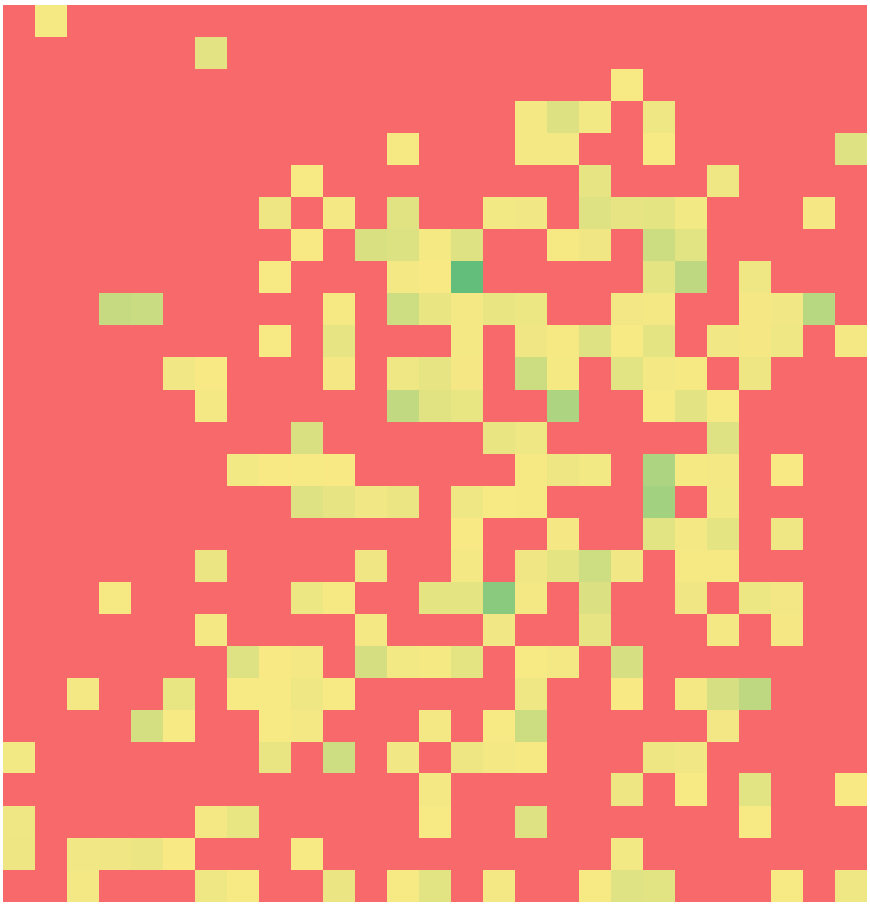

Figure 12. Importance visualised as grid (prunned)

Task 2, Support vector classifier

2.1.

From the frequencies plot we can make the observation that the most common soil types are „red soil“, „grey soil“ and „very damp grey soil“.

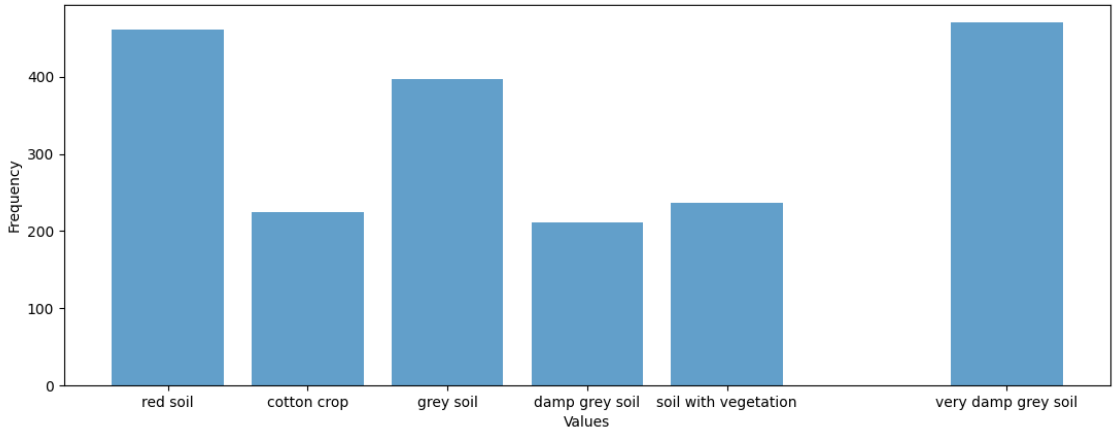

Figure 13. Frequencies of each class

2.2.

In my plot I used feature V1 for the x axis and V2 for the y axis. By looking at I expect good performance of the SVM by these columns, because visually a clear line can be drawn between the different classes, there are not a lot of cases were 2 classess overlap.

I suspect that using a kernel trick will get performance, but not drastically different from linear. My guess is that linear will be just as good or even better, because not needing a complex kernel with result in faster calculations.

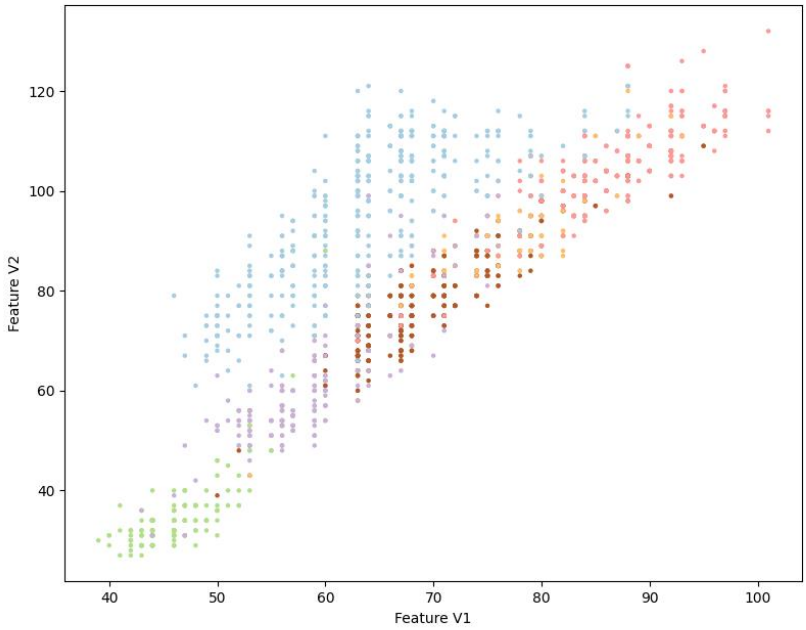

Figure 14. Plot of classess by V1 and V2

2.3.

The best 10-fold cross-validation error is 0.7635 when using a cost value of 0.5. The overall accuracy si 76.05%.

Cost values from 0.1 to 2.1 in 0.2 intervals were checked.

		Predicted					
	Class	1	2	3	4	5	7
True	1	431	2	10	7	15	11
	2	0	185	0	1	8	2
	3	17	5	362	75	6	45
	4	1	0	1	9	2	4
	5	3	15	2	1	135	9
	7	9	17	22	118	71	399

Figure 15. Confusion matrix of SVM

Class	Accuracy
1	90.54%
2	94.38%
3	70.98%
4	52.94%
5	81.81%
7	62.73%

Figure 16. Accuracies for each class

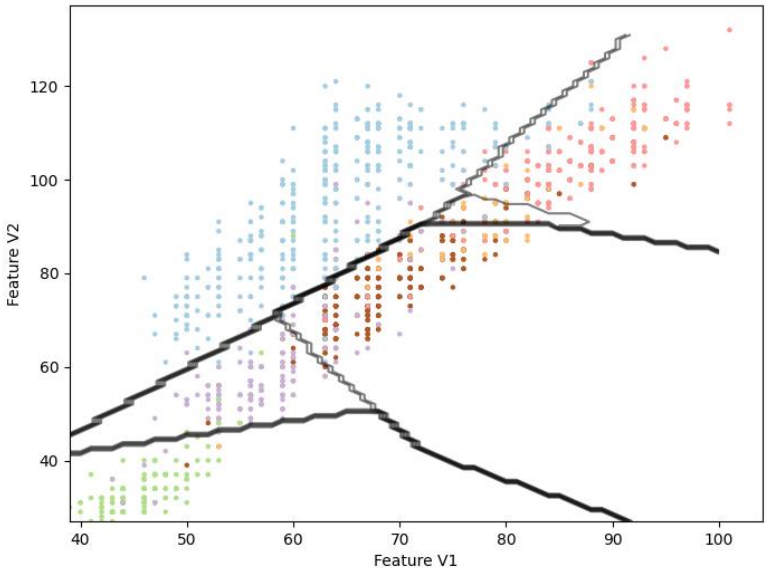

Figure 17. SVM with linear decision boundry

2.4.

The non-linear kernel function that I decided to use is the radial basis funciton. The best 10-fold cross-validation error is 0.7635 when using a cost value of 0.9 and a gamma of 0.1. The overall accuracy si 77.40%.

Cost values from 0.1 to 1.6 in 0.2 intervals were checked.
Cost values from 0.1 to 3.1 in 0.1 intervals were checked.

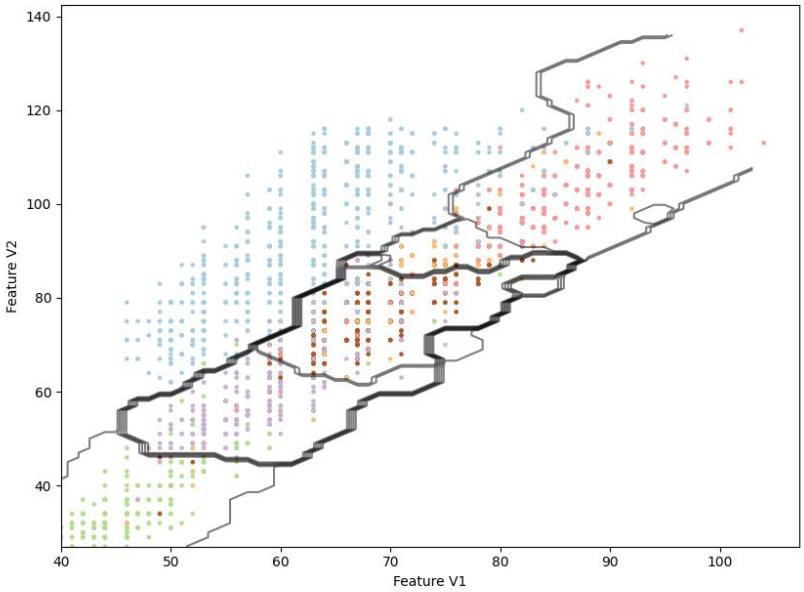

Figure 18. SVM with non-linear decision boundry

2.5.

Using a linear model with all of the feaures, the best cost hyperparameter found was 0.1. The accuracy of this model is 85.20%. The 10-fold cross-validation error rate is 0.8255.
Confusion matrix:

		Predicted					
	Class	1	2	3	4	5	7
True	1	453	1	4	0	8	0
	2	1	210	1	2	10	1
	3	0	0	367	43	0	13
	4	1	1	22	90	6	44
	5	6	12	1	1	184	12
	7	0	0	2	75	29	400

Figure 19. Confusion matrix of a fully trained linear SVM

Using a radial model with all of the feaures, the best cost hyperparameter found was 0.18 and a gamma of 0.00024.
The accuracy of this model is 89.10%. The 10-fold cross-validation error rate is 0.8515.
Confusion matrix:

		Predicted					
	Class	1	2	3	4	5	7
True	1	458	0	2	0	13	0
	2	2	221	1	4	6	0
	3	1	0	383	44	0	16
	4	0	0	6	116	2	36
	5	0	1	0	3	196	10
	7	0	2	5	44	20	408

Figure 20. Confusion matrix of fully trained radial SVM

Task 3, Artificial neural networks

3.1.

By look at the 10 hearbeats, we can deduce a couple of things that could affect the training process:

- Variable length hearbeat
- Baseline ECG strength not during a heartbeat.
- Variable shape of ECG between heartbeats
- They are not categorized by person or order between heartbeats. So we don‘t know how multiple normal heartbeats in a row of a single person looks like.

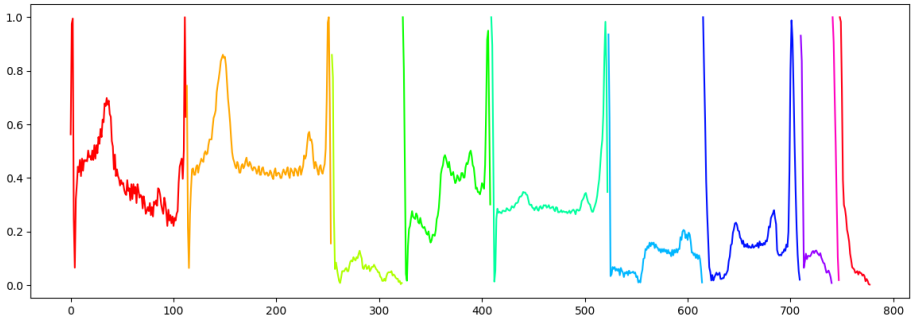

Figure 21. 10 "normal" heartbeats

3.2.

From the scatter plot, we can say that it would be difficult to draw a separating line between all of the classess. The reason for this is, because of all the class have a lot of overlapping area and outliers.

But the distribution of each class is not random, we can clearly see a concentration of different classess in different areas.

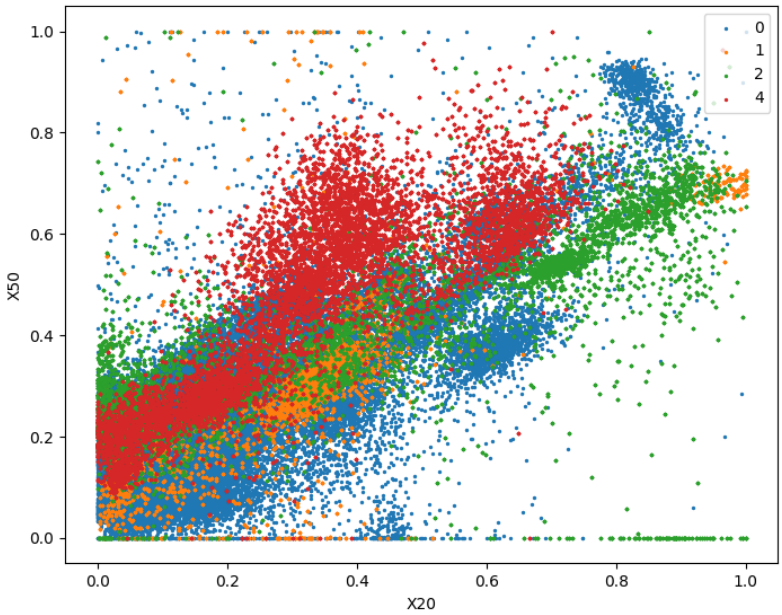

Figure 22. Scatter plot of X20 vs X50

3.3.

In the training set, there is a low number „supraventricular“ samples comared to the other types and there suprisingly there is a significant number of „unknown“ samples.

The test dataset doesn‘t seem very balanced, there are a lot more „normal“ samples than anything else, this in conjuction with the high number of „normal“ samples in the training dataset

it will likely result in a high overall accuracy, but not necessarily good on every class. It will be very important to see how the distribution of predictions looks in the confusion matrix.

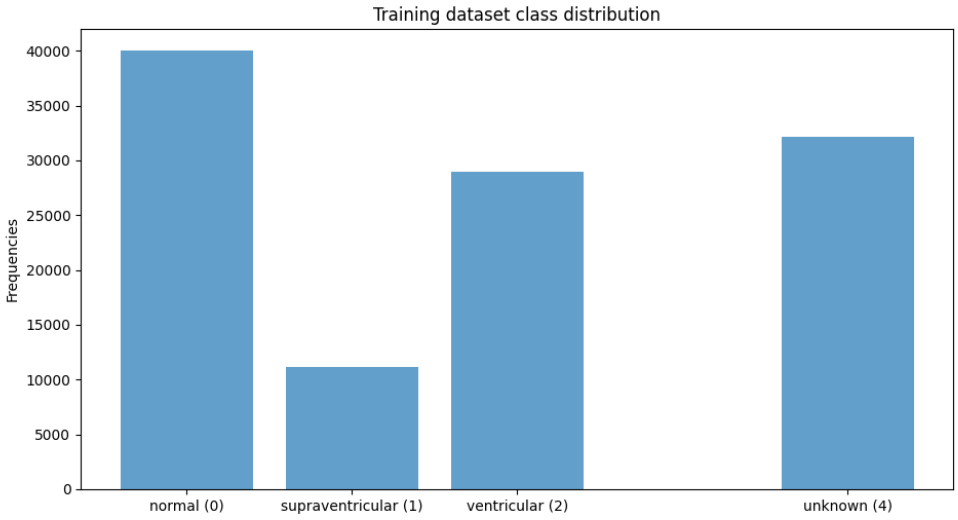

Figure 23. Distribution of classess for training dataset

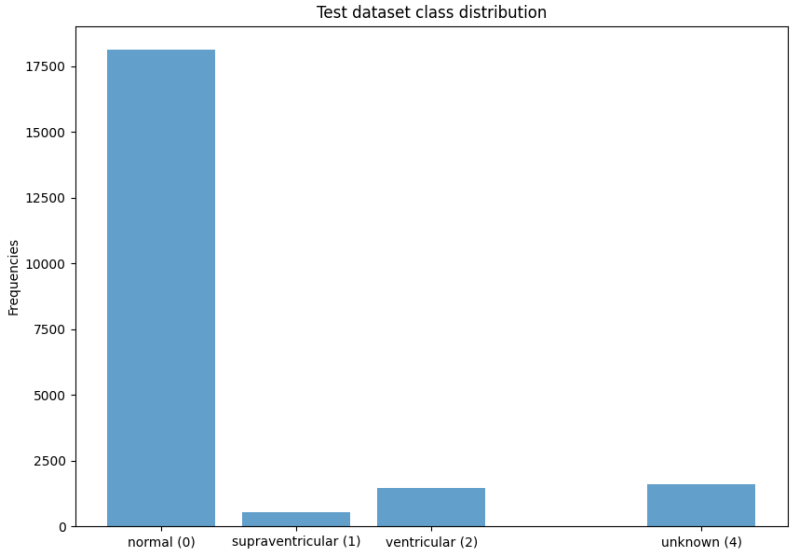

Figure 24. Distribution of classess for test dataset

3.4.

I was successful in training a neural network which has a high overall accuracy, but not for all classess. The main reason for this is the poor distribution of classess in training dataset and especially test dataset.

Observations about how changing parameters affect the neural network:

- Increasing epochs does not necesarrily increase accuracy. If it is too large, the number of false positives increase.
- Increasing the number of hidden layers improves performance. At large numbers of hidden layers most of the nodes don‘t contribute significantly and only increase training time.
- Changing the activation function from rectifier to tanh, I didin‘t notice a significant difference in overall accuracy. But it did increase the accuracy of class 1, 2 and 4, but decreased in class 0.

Parameters:

- Hidden layers: [50, 20]
- Activation function: rectifier
- Epochs: 20
- L1 regularization: 0.00001

Overall accuracy: 87.95%

- Class 0 accuracy: 94.62%
- Class 1 accuracy: 61.62%
- Class 2 accuracy: 48.89%
- Class 4 accuracy: 60.88%

Confusion matrix:

		Predicted				
	Class	0	1	2	4	other
True	0	17143	262	27	2	683
	1	173	281	1	0	1
	2	93	645	708	0	2
	4	38	51	57	979	483

Figure 25. Confusion maitrx of trained NN