CENG499 - Homework 3

Utku Güngör - 2237477

1 Decision Trees

1.1 Info Gain

Test accuracy for this case is 0.9333333333333333. Related diagram can be checked from info-gain.pdf .

1.2 Average Gini Index

Test accuracy for this case is 0.9. Related diagram can be seen checked info-gain-pre-pruned.pdf .

1.3 Info Gain with Pre-pruning

Test accuracy for this case is 0.9666666666666667. Related diagram can be checked from avg-gini.pdf .

1.4 Average Gini Index with Pre-pruning

Test accuracy for this case is 0.9. Related diagram can be seen checked avg-gini-pre-pruned.pdf .

2 Support Vector Machines

2.1 Task 1

The first effect we can observe as C changes is the margin width. We have large margins with small C, and as C increases, margins get narrower. Hence, different support vectors are formed. Chance of misclassifying an example decreases as well as C increases and hyperplane does a better job on the training data. Against this, ignoring some data and misclassifying them is more possible with lower C values. However, this might depend on the dataset and C value so that some outliers can be ignored with large C value as well.

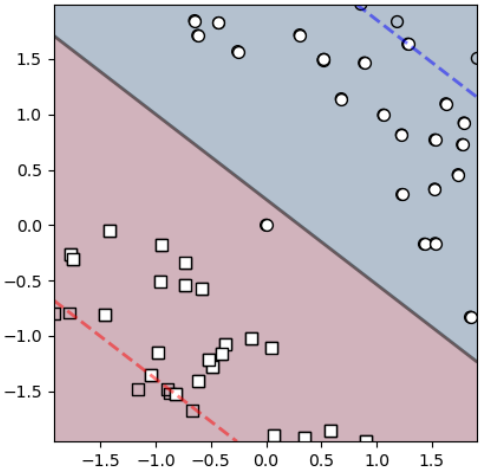

Figure 1: C = 0.01

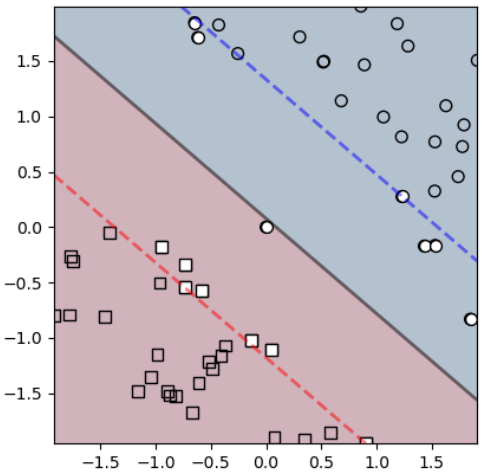

Figure 2: C = 0.1

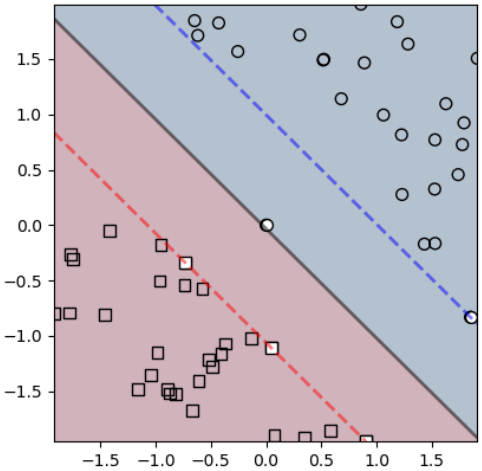

Figure 3: C = 1

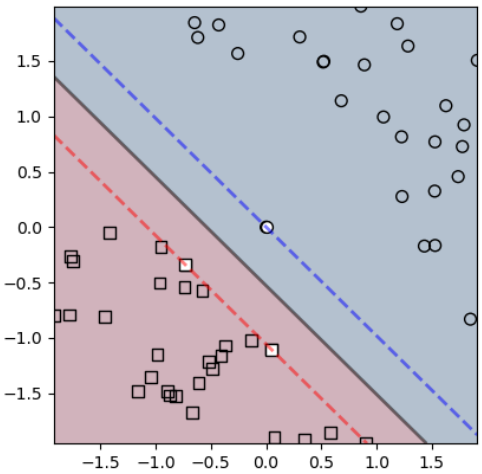

Figure 4: C = 10

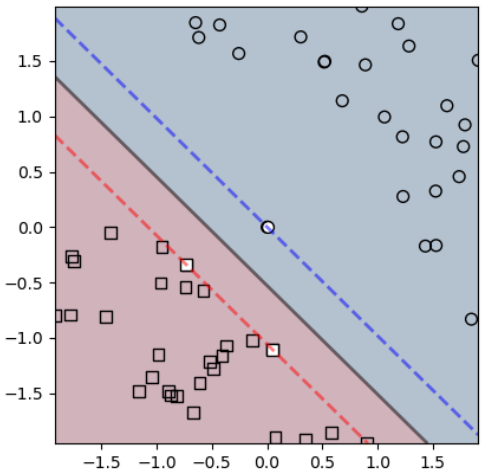

Figure 5: C = 100

2.2 Task 2

- **Linear Kernel:** Performs good on linearly separable data, but when it comes to linearly not separable data which is our case in this homework, this kernel performs terrible classifications.
- **Polynomial Kernel:** Performs better than linear kernel, it was able to separate left part of the data as seen in Figure 7. But since the classification needs to be made in kind of a circular shape, this kernel is not a right choice.
- **RBF Kernel:** This kernel gives the best results since our data has kind of inner and outer groups and needs to be separated with a circular shape classifier, this is the perfect choice.
- **Sigmoid Kernel:** This kernel does not perform as good as RBF in our case as seen in Figure 9. Our data set is not suitable for this type of kernel.

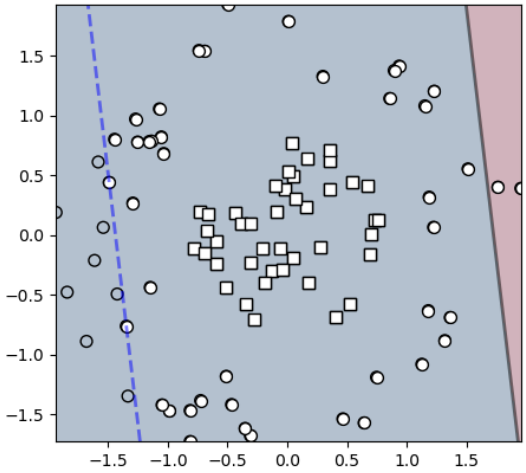

Figure 6: Linear Kernel

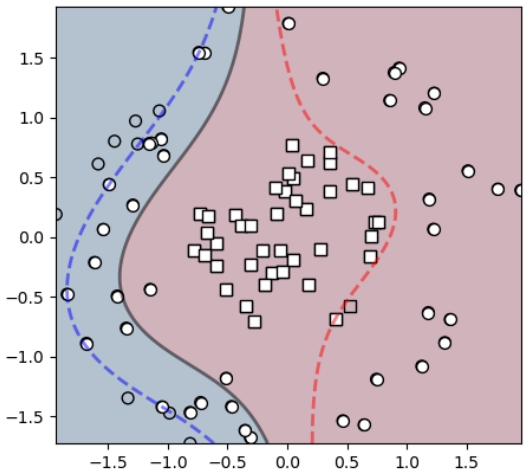

Figure 7: Polynomial Kernel

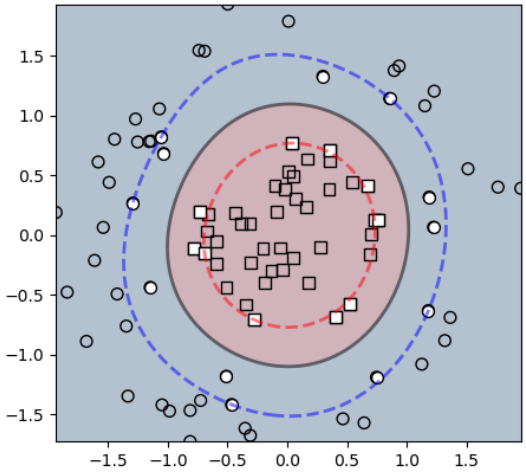

Figure 8: RBF Kernel

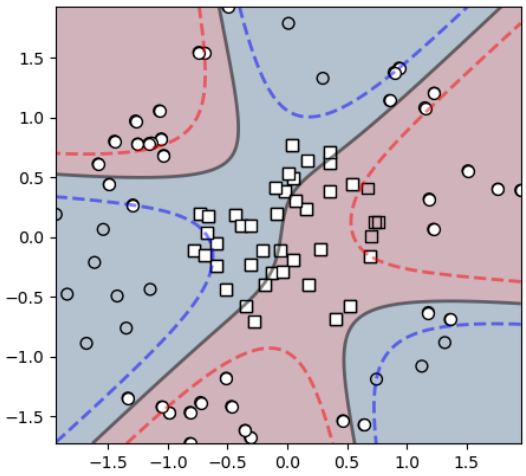

Figure 9: Sigmoid Kernel

2.3 Task 3

For this task, the model has been tested with 4 kernels, 3 C values and 3 gamma values, so 36 hyperparameter configurations in total.

Table 1: Linear Kernel

Gamma \ C Value	0.1	1	10
Does not effect	0.734	0.692	0.664

Table 2: RBF Kernel

Gamma \ C Value	0.1	1	10
0.001	0.706	0.761	0.794
0.01	0.774	0.813	**0.831**
0.1	0.532	0.733	0.737

Table 3: Polynomial Kernel

Gamma \ C Value	0.1	1	10
0.001	0.572	0.689	0.736
0.01	0.770	0.787	0.766
0.1	0.751	0.751	0.751

Table 4: Sigmoid Kernel

Gamma \ C Value	0.1	1	10
0.001	0.693	0.743	0.737
0.01	0.408	0.676	0.673
0.1	0.503	0.503	0.504

The best hyperparameter configuration is RBF kernel, C = 10, gamma = 0.01, and its test accuracy with the whole training data is 0.815.

2.4 Task 4

I have interpreted 0 as positive, and 1 as negative for the following parts.

2.4.1 Without setting class_weight

The initial data is pretty imbalanced, there are 1000 examples and 50 of them belong to one class while 950 belong to another. Given the data is very imbalanced, accuracy is not a good performance metric since there will be a bias towards the majority class.

$\frac{True}{Predicted}$	0	1
0	0	0
1	50	950

As seen in the confusion matrix, predicted values are from the majority class which shows the bias. 50 of the 1000 examples are false negatives because of the tendency towards the majority but there is no false positive example. So, the accuracy of the positive(0) class is 0.0 whereas the accuracy of the negative(1) class is 1.0. The test accuracy is 0.950.

2.4.2 Oversampled data

To oversample the data, I basically added the minority class' elements to itself according to the imbalance ratio so that they have same number of examples. Below is the confusion matrix for the oversampled data set:

$\frac{True}{Predicted}$	0	1
0	12	11
1	38	939

There are 38 false negatives and 11 false positives according to the table. So, the accuracy of the positive(0) class is 0.24 whereas the accuracy of the negative(1) class is 0.988. Therefore, this model performs well at classifying negatives but not positives. Overall test accuracy is better than the previous case, and it is 0.951.

2.4.3 Undersampled data

To undersample the data, I basically ignored some part of the majority class according to the imbalance ratio so that they have same number of examples. Below is the confusion matrix for the undersampled data set:

$\frac{True}{Predicted}$	0	1
0	36	222
1	14	728

There are 14 false negatives and 222 false positives according to the table. So, the accuracy of the positive(0) class is 0.72(so much better than the previous two cases) whereas the accuracy of the negative(1) class is 0.766(so much worse than the previous two cases). So the model performs close for both classes and the performance is not as good as other cases. Overall test accuracy is 0.764. This accuracy might change according to the undersampling technique. As mentioned before, I basically ignored some examples of the majority class. Test accuracy would be higher for some other algorithms.

2.4.4 With setting class_weight to balanced

Below is the confusion matrix for the data set with class_weight set to balanced:

$\frac{True}{Predicted}$	0	1
0	16	30
1	34	920

There are 34 false negatives and 30 false positives according to the table. So, the accuracy of the positive(0) class is 0.32 whereas the accuracy of the negative(1) class is 0.968. Therefore, this model also performs well at classifying negatives but not positives. Overall test accuracy is 0.936, which is better than the accuracy of the undersampled data, but less successful than the oversampled data.