Machine Learning - Assignment 1

Adam Domoslawski (12241331), Dmytro Kondrashov (01641210),
Michael Wagner (00926215)

November 19, 2024

The first exercise required the training and testing of three classifiers for four different datasets, two of which were self selected, while two were selected by the lecturers. In Section 1 we will first describe the details about the four selected datasets as well as all required/executed pre-processing steps. Afterwards, Section 2 introduces our experimental setup as well as the three selected classifiers and their respective parameter choices, while Section 3 describes and analyzes the results of all classifiers for each of the datasets. Finally, Section 4 provides a short conclusion.

1 Data exploration

For each dataset, we identified which attributes are binary, categorical, ordinal or numeric and provide a short analysis for each dataset as well as for the distribution of the classification targets below.

1.1 Porto Seguro

The first self-selected dataset is the *Porto-seguero* dataset[1]. It is a large dataset for binary classification, consisting of 37 different attributes (25 nominal, 5 ordinal and 7 interval attributes), more than half a million samples as well as many attributes/samples with missing values. For a more detailed analysis and discussion on how we treated the missing values we refer the interested reader to Exercise 0 and Subsections 1.5 and 1.6.

1.2 HPC

For the second dataset, we selected the HPC dataset[2]. Data consist of information on 4331 jobs in a high-performance computing environment. Seven attributes were recorded for each job along with a discrete class describing the execution time. In those 8 features, there was only one class (target). Furthermore, it does not contain any missing values that need to be treated. There are 5 numeric, 2 nominal and 1 ordinal classes. 'class' is a categorical attribute meaning how fast the job was done: 'VF' (Very Fast), 'F' (Fast), 'M' (Medium), 'L' (Long). There are two features (day and hour), which are related to each other. We therefore decided to test two versions of this dataset: The original as well as one where we merged these two columns into on column representing a timestamp. The intuition behind this was to create some cyclic ordering (e.g. Tuesday morning follows after Monday night). Evaluations of both dataset encodings are presented in the final evaluation.

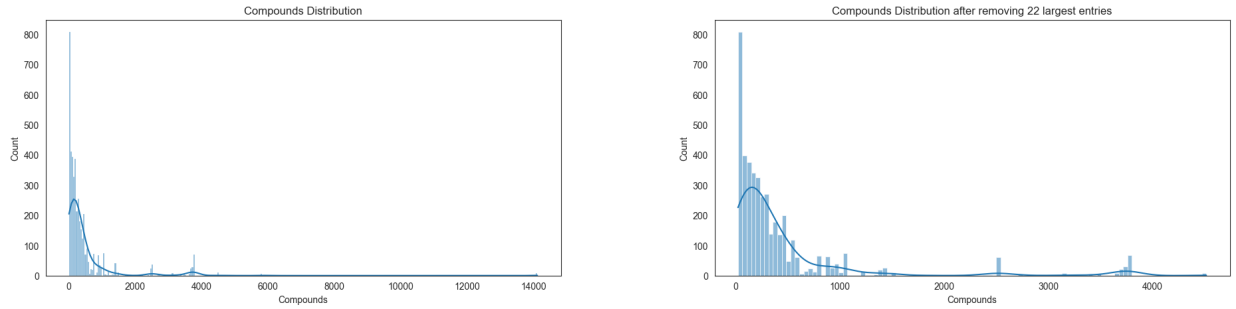
Additionally, we observed various outliers towards the end of the x-axis in the charts. To better visualize the data distribution, we removed the 22 largest entries from Figure 1. However, there still remains a number of entries with significantly larger values.

1.3 Breast Cancer

The breast cancer dataset is a realtively small binary classification dataset for predicting whether breast tissue is malignant (true) or benign (false), containing 30 attributes and 285 samples. Therefore, all attributes besides the target are of numeric (ratio) datatype, describing the mean, stderr and worst of various metrics. About a third of the labels in the training set is true, the rest being false. Since there were no missing values and all values were of ratio type, the only pre-processing necessary was normalization.

[1] `https://www.openml.org/search?type=data&sort=version&status=any&order=asc&exact_name=porto-seguro&id=42206`

[2] `https://www.openml.org/search?type=data&sort=version&status=any&order=asc&exact_name=hpc-job-scheduling&id=41212`



(a) Original Compounds

(b) Truncated Compounds

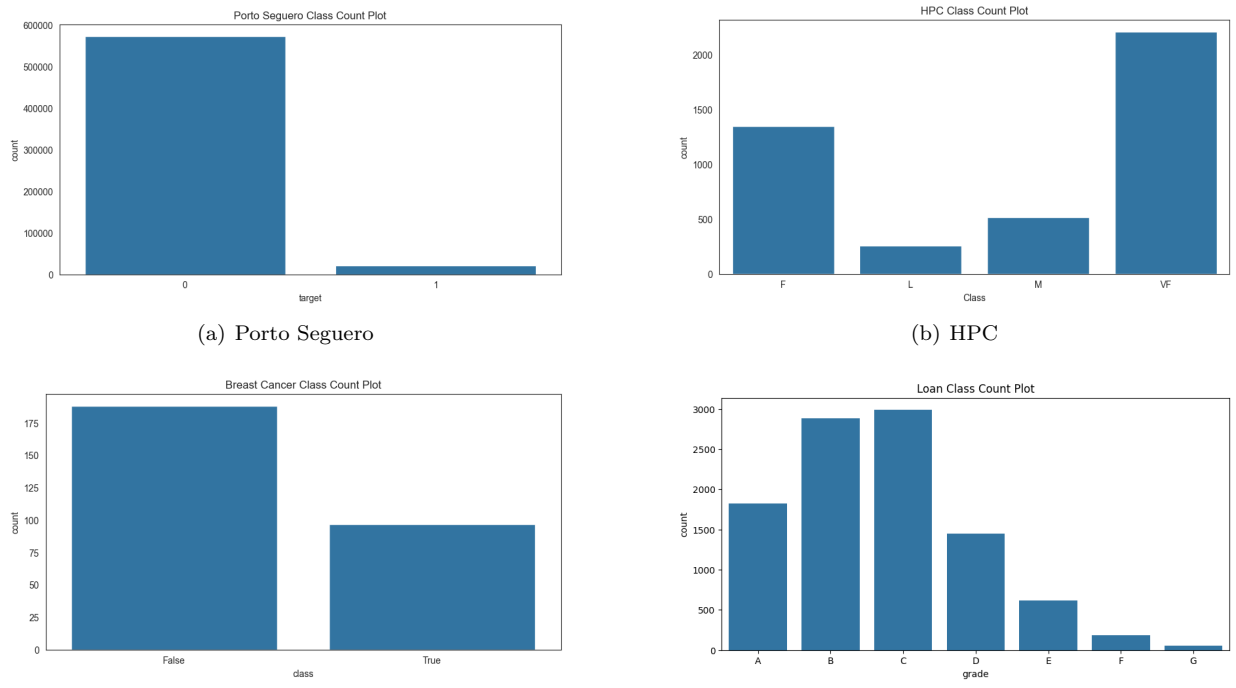
Figure 1: Compound highest value truncation

1.4 Loan

The loan dataset is significantly larger, containing 90 different attributes and 10000 samples. The goal is to classify for different grades (A,B,C,D,E,F,G) as target. While it does not contain any missing values, it consists of many different attribute types. In total we identified 13 categorical variables (5 of which binary) and 27 ordinal variables, these include for example metrics such as months or years. We left any columns with more than 30 unique values as numeric, as these might otherwise increase the model complexity too much, resulting in a total of 49. Finally, we removed the attribute *policy_code*, since it had only one value.

1.5 Target distributions

In this subsection, we briefly discuss the distributions of the classification targets. Porto Seguro and Breast Cancer are datasets for binary classification, while HPC and Loan require multi-class classification. We can observe that the data distribution in the Porto Seguro dataset is very unbalanced. Further, the distribution for loan looks reasonably close a gaussian distribution around grades B and C.



(a) Porto Seguero

(b) HPC

(c) Breast Cancer

(d) Loan

Figure 2: Target distributions of the four different datasets

1.6 Encoding + Normalization

To encode our non-binary categorical data we used the 1-hot encoding, as implemented in sklearn's OneHotEncoder. For normalization we use the z-score normalization provided by StandardScaler, while missing values are treated via imputation with the mean, using SimpleImputer.

2 Classifiers and experimental setup

In order to facilitate testing of many different parameter configurations and ensure that our results are comparable we designed an experimental setup based on sklearn's Pipeline. The pipeline takes as input the feature structure of the dataset and then sets the different encoders as discussed in Section 1.6. Afterwards, for each classifier and parameter setting, the pipeline is used to train models using the holdout method (with a random 80/20 split between training set and test set with fixed seed) as well as cross-validation (using 5-fold validation) based on the data set. Finally, the model with the best performance (we used accuracy as metric) is returned for each classifier, parameter setting and splitting method (holdout/cross validation) and calculate the following four metrics, where TP, TN, FP and FN are the number of true positive, true negatives, false positives and false negatives respectively:

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \qquad \textbf{Precision} = \frac{TP}{TP+FP}$$

$$\textbf{Recall} = \frac{TP}{TP+FN} \qquad \textbf{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

We selected the following three classifiers and parameter settings for our experiments:

- **k-NN:** The first selected classifier is the k-Nearest Neighbors heuristic as implemented in sklearn. Considering that there is real training time needed for k-NNs we expected this classifier to be the fastest out of the selected. As parameters, we simply tested all different values of k between 2 and 12.
- **Decision Trees:** Next, we consider decision trees, as implemented in sklearn. Our experiments consider decision tress of various depths between 3 and 14. This can lead to some understanding how deep a search tree can be before overfitting.
- **SVMs:** Last we selected the scalable linear support vector machines (SVC) as implemented in sklearn. By using different kernels we can draw conclusions about the data, such as if the dependencies within the data appear to be mostly linear or not. We consider the following kernels: linear, radial basis function (rbf), polynomial (poly) and sigmoid.

3 Experimental Results

In the following we will present our classification results for each selected datatype and classifier. Note that all results reported with cross validation describe the average over all cross validation runs, therefore the ratio between holdout and cross validation can give a bit of insight about the stability of the models. Additionally, all metrics and the confusion matrices are calculated only on the test sets, not on the training sets.

3.1 Porto Seguro

The experimental results for two of the classifiers on the Porto Seguro dataset are reported in Table 1 and 2 respectively. Due to the immense size of the dataset with over half a million samples and the consequent large run time required for training and classification, we only have one models for k-NNs and SVC. Decision trees were significantly faster to train for this dataset, so we provide results for all considered maximum depths. Further, we would like to mention that the results for the timing variables between the different classifiers cannot be directly compared since we had to run them on different infrastructures.

k-NN: We only managed to train a k-NN with $k = 1$, which already took almost an hour with 30 jobs in parallel. The model still achieved a reasonable accuracy, precision and recall with an f1-score of 0.931. Since k-NNs scale bad in terms of running time with an increased number of samples, this result was to be somewhat expected.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
k=1	0.931	0.932	0.929	0.930	0.931	0.932	0.930	0.931	647.665	3254.685

Table 1: Results Porto k-NN

Decision Trees: Training the decision trees was surprisingly fast for when compared to the other two classifiers. Even with a maximum depth of 14 our experiments did not take much longer than 10 minutes, even with cross validation. Furthermore, the classification results were significantly better than for the 1-NN classifier. The best model was actually the decision tree with a maximal depth of 5, achieving an accuracy and recall of 0.964, a precision of 0.944 and therefore an f1-score of 0.946, while needing less than a minute to train using cross validation.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
depth=3	0.963	0.964	0.927	0.931	0.963	0.964	0.945	0.946	7.385	34.609
depth=4	0.963	0.964	0.927	0.939	0.963	0.964	0.945	0.946	7.708	55.772
depth=5	0.963	0.964	0.939	0.944	0.963	0.964	0.945	0.946	11.777	59.769
depth=6	0.963	0.964	0.938	0.939	0.963	0.964	0.945	0.946	13.832	65.472
depth=7	0.963	0.963	0.936	0.937	0.963	0.963	0.945	0.946	17.317	87.361
depth=8	0.963	0.963	0.935	0.935	0.963	0.963	0.945	0.946	19.547	101.412
depth=9	0.963	0.963	0.933	0.933	0.963	0.963	0.945	0.946	24.671	127.355
depth=10	0.963	0.963	0.934	0.932	0.963	0.963	0.945	0.946	29.995	157.281
depth=11	0.962	0.963	0.932	0.932	0.962	0.963	0.945	0.945	36.979	188.276
depth=12	0.962	0.962	0.932	0.932	0.962	0.962	0.945	0.945	43.250	231.091
depth=13	0.962	0.962	0.932	0.931	0.962	0.962	0.945	0.945	52.519	278.062
depth=14	0.961	0.961	0.931	0.931	0.961	0.961	0.944	0.945	62.872	325.279

Table 2: Results Porto Decision Trees

SVC: Finally, due to the time consumption, we also only opted to train one model with the SVC classifier using a holdout method. The resulting model achieved an accuracy of 0.963, precision of 0.927, recall of 0.963 and f1-score of 0.945, while taking 4190 seconds to train with the same setup as used for the 1-NN. So, while the performance was comparable to decision trees, the immense training time required does not make usage of SVC an attractive classifier for the Porto Seguro dataset.

Overall, decision trees were the only real viable classifiers for the Porto Seguro dataset, performing as good as SVC in terms of f1-score and outperforming k-NNs, while taking many orders of magnitude less in terms of time to train the models. Lastly, decision trees with depth 5 already performed best. We omit any visual representations, such as confusion matrices, for this dataset for space reasons and due to the limited availability of models for each classifier.

3.2 HPC

The experimental results for the different classifiers on the modified HPC dataset are reported in Table 3, 5 and 7 respectively; on the original HPC dataset are reported in Table 4, 6 and 8 respectively. The HPC dataset consists of 4331 entries and 8 features. Therefore, the model training was very fast (up to a couple of seconds). There are 2 tables for the same classfier: one uses the original dataset, and the other one uses an updated feature: timestamp. Timestamp is a combination of the day and hour, which was afterwards normalized. Day and hour were removed respectively.

k-NN: For the HPC dataset, the holdout method is better than cross-validation across all metrics. The best performance was achieved with $k = 2$, where the accuracy, precision, recall, and f1-score: 0.994. In contrast, the best cross-validation f1-score was 0.941 for $k = 7$. While the holdout method showed minimal variation across k, the cross-validation results were less stable, with performance peaking between $k = 7$ and $k = 9$. Timing results indicate that the k-NN algorithm was relatively fast for the holdout method, with runtimes ranging from 0.064s to 0.081s. Cross-validation, however, required significantly more time, especially for larger values of k.

The results are slightly (2-3%) more precise on the updated HPC dataset. Additionally, the timing is faster on the updated dataset by 60%. However, the timing does not play a vital role since all results are subsecond fast.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
k=2	0.994	0.932	0.994	0.943	0.994	0.932	0.994	0.931	0.041	0.286
k=3	0.993	0.949	0.993	0.952	0.993	0.949	0.993	0.949	0.050	0.303
k=4	0.993	0.950	0.993	0.954	0.993	0.950	0.993	0.950	0.041	0.285
k=5	0.990	0.955	0.990	0.956	0.990	0.955	0.990	0.954	0.037	0.306
k=6	0.990	0.954	0.990	0.956	0.990	0.954	0.990	0.953	0.039	0.282
k=7	0.988	0.955	0.989	0.957	0.988	0.955	0.989	0.955	0.043	0.290
k=8	0.988	0.952	0.989	0.953	0.988	0.952	0.989	0.951	0.041	0.293
k=9	0.985	0.956	0.985	0.956	0.985	0.956	0.985	0.955	0.042	0.287
k=10	0.988	0.955	0.989	0.955	0.988	0.955	0.989	0.954	0.045	0.301
k=11	0.988	0.955	0.989	0.956	0.988	0.955	0.989	0.955	0.044	0.274
k=12	0.988	0.956	0.989	0.956	0.988	0.956	0.988	0.955	0.040	0.281

Table 3: Results of updated HPC k-NN

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
k=2	0.994	0.922	0.994	0.934	0.994	0.922	0.994	0.923	0.067	0.402
k=3	0.992	0.934	0.992	0.940	0.992	0.934	0.992	0.934	0.067	0.466
k=4	0.993	0.933	0.993	0.940	0.993	0.933	0.993	0.933	0.066	0.401
k=5	0.990	0.936	0.990	0.940	0.990	0.936	0.990	0.936	0.064	0.393
k=6	0.988	0.938	0.988	0.941	0.988	0.938	0.988	0.938	0.065	0.397
k=7	0.987	0.942	0.987	0.944	0.987	0.942	0.987	0.941	0.065	0.400
k=8	0.991	0.939	0.991	0.942	0.991	0.939	0.991	0.939	0.066	0.398
k=9	0.987	0.941	0.987	0.943	0.987	0.941	0.987	0.941	0.065	0.399
k=10	0.983	0.941	0.983	0.943	0.983	0.941	0.983	0.941	0.065	0.410
k=11	0.987	0.940	0.987	0.941	0.987	0.940	0.987	0.939	0.078	0.398
k=12	0.985	0.939	0.985	0.940	0.985	0.939	0.985	0.939	0.065	0.413

Table 4: Results of original HPC k-NN

Decision Trees: Decision Trees achieved perfect classification for all tested depths using both the holdout and cross-validation methods. Depths of 3 and above yielded an accuracy, precision, recall, and f1-score of 1.0. The timing was consistently under 0.031s for holdout and slightly longer for cross-validation, with the highest timing of 0.391s for depth=6. This classification suits best for this dataset.

The results between the updated and original are completely identical since we reached the maximum score (1.000). The only difference is the timing. It is still not important, since we work in the 1ms range for holdout and subsecond for the cross-validation.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
depth=3	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.014	0.127
depth=4	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.013	0.123
depth=5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.012	0.130
depth=6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.014	0.124
depth=7	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.014	0.124
depth=8	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.013	0.129
depth=9	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.013	0.129
depth=10	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.013	0.129
depth=11	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.012	0.135
depth=12	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.012	0.127
depth=13	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.013	0.127
depth=14	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.012	0.126

Table 5: Results of updated HPC Decision Trees

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
depth=3	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.017	0.144
depth=4	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.142
depth=5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.014	0.141
depth=6	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.139
depth=7	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.136
depth=8	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.133
depth=9	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.136
depth=10	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.142
depth=11	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.015	0.144
depth=12	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.016	0.148
depth=13	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.016	0.140
depth=14	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.014	0.130

Table 6: Results of original HPC Decision Trees

SVC: SVC achieved the best results with the linear kernel, reaching perfect accuracy, precision, recall, and f1-score of 1.0 for both holdout and cross-validation methods. The Rbf and Poly worked for holdout well as well, but Sigmoid showed the worst result.

The original dataset performed slightly better than the updated dataset. The biggest difference is in svc sigmoid: around 2%.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
SVC linear	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.026	0.184
SVC rbf	0.999	0.987	0.999	0.988	0.999	0.987	0.999	0.987	0.049	0.301
SVC poly	0.998	0.984	0.998	0.984	0.998	0.984	0.998	0.984	0.048	0.295
SVC sigmoid	0.889	0.868	0.918	0.900	0.889	0.868	0.898	0.879	0.036	0.251

Table 7: Results of updated HPC SVC

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
SVC linear	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.036	0.202
SVC rbf	0.998	0.988	0.998	0.988	0.998	0.988	0.998	0.987	0.061	0.333
SVC poly	0.999	0.987	0.999	0.987	0.999	0.987	0.999	0.987	0.074	0.365
SVC sigmoid	0.894	0.911	0.926	0.928	0.894	0.911	0.904	0.913	0.042	0.250

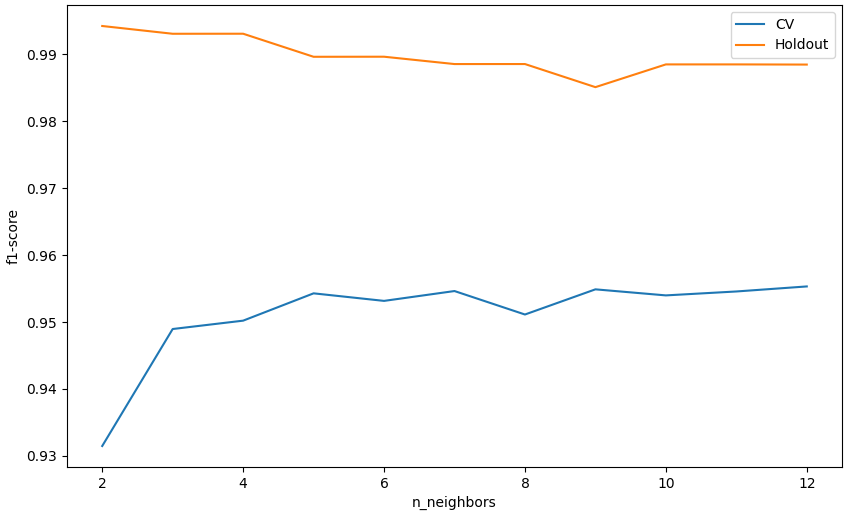
Table 8: Results of original HPC SVC

Overall, the SVC classifier with linear kernel as well as decision trees performed the best out of the tested classifiers. Grouping the day and hour into one timestamp resulted in improved performance for the k-NNs, while slightly decreasing performance of SVCs. Figure 3 shows how the performance in terms of f1-score changes for different parameters settings for the three classifiers for both cross validation and holdout, while Figure 4 shows the confusion matrix of the best performing model of each classifier.

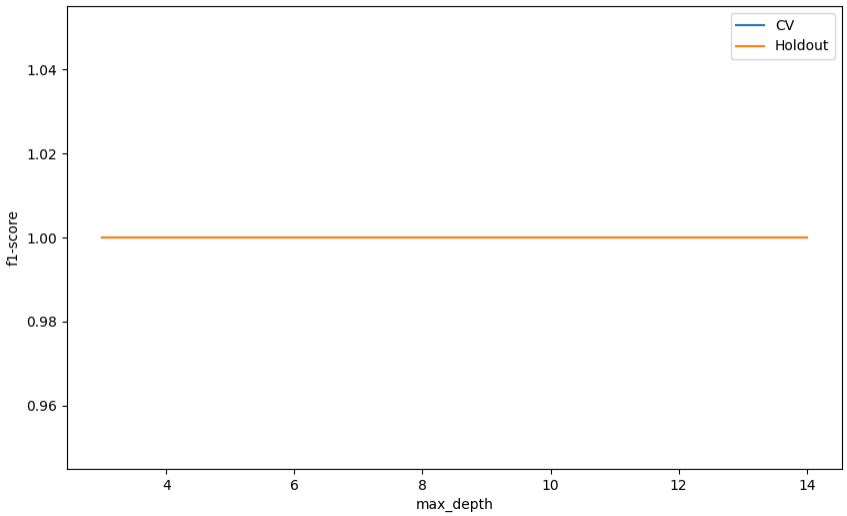
3.3 Breast Cancer

The experimental results for the different classifiers on the breast cancer dataset are reported in Table 9, 10 and 11 respectively. Since breast cancer was a relatively small dataset the required training and classification times are quite small, with all classifiers taking way below one second of run time.

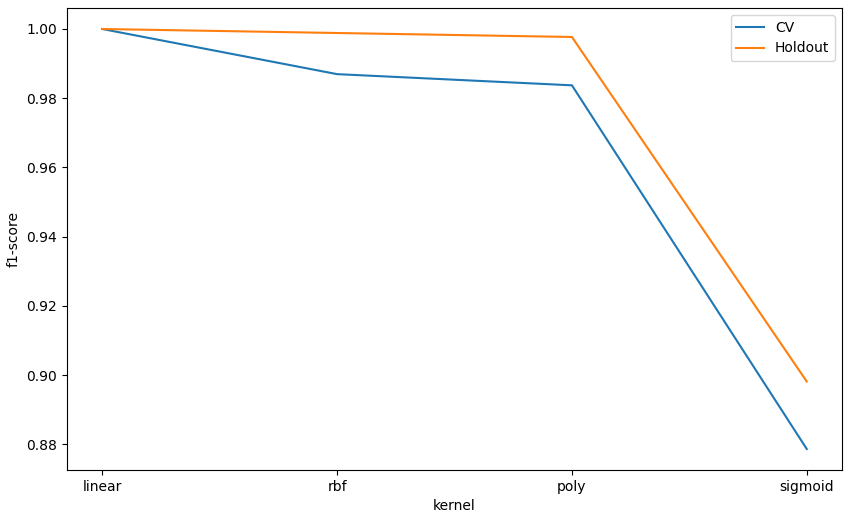
k-NN: For this dataset, the accuracy, precision, recall, and f1-score of the experiments using the holdout method were superior to the cross-validation. The model with the best f1-score had an accuracy and recall of 0.982, precision of 0.983 and therefore also an f1-score of 0.982. The best f1-score using cross-validation was 0.972 for $k = 9$. The performance of all k-NNs with $k \geq 5$ was pretty much identical, while k-NNs with fewer neighbors (aside from $k = 3$) performed worse.



(a) k-NN

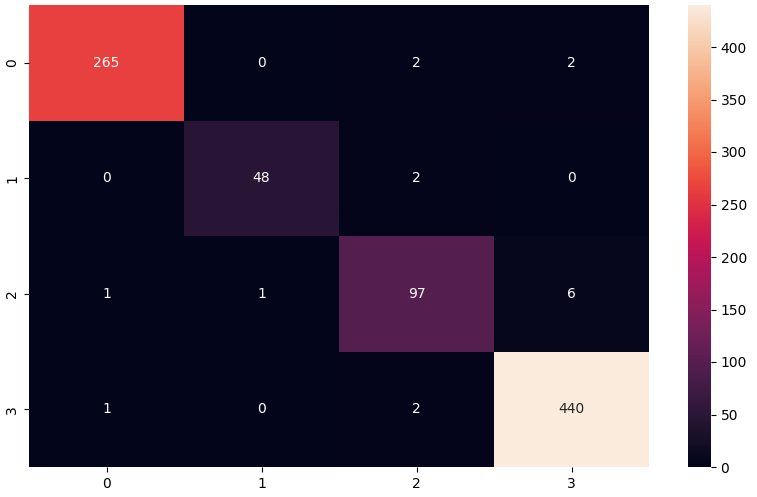


(b) Trees

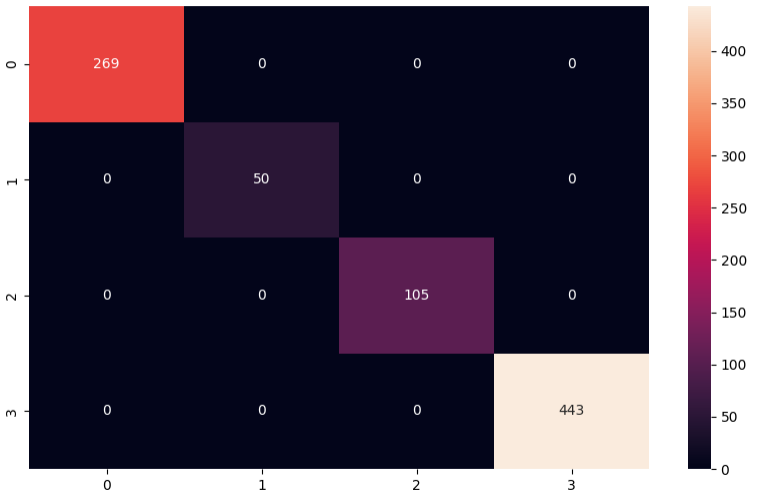


(c) SVC

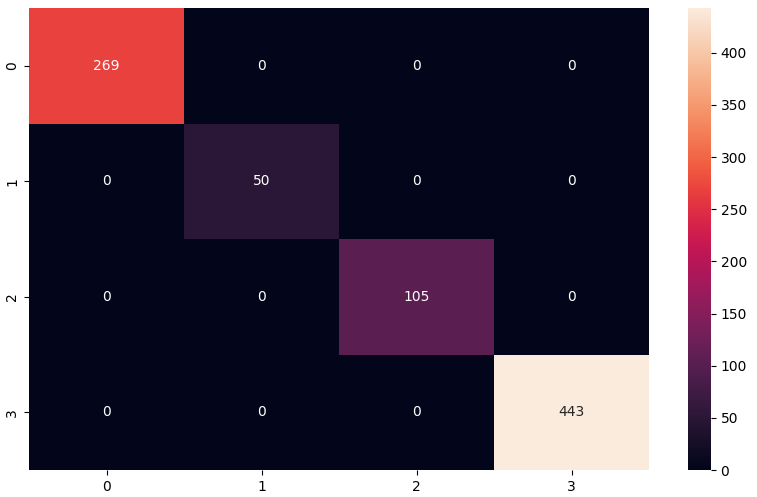
Figure 3: F1-score for different parameter settings for the three classifiers on the HPC dataset



(a) k-NN



(b) Decision Trees



(c) SVC

Figure 4: Confusion Matrix of the best model for each of the three classifiers on the HPC dataset

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
k=2	0.965	0.947	0.967	0.950	0.965	0.947	0.965	0.946	0.012	0.084
k=3	0.982	0.965	0.983	0.966	0.982	0.965	0.982	0.964	0.010	0.073
k=4	0.947	0.954	0.952	0.956	0.947	0.954	0.947	0.954	0.010	0.074
k=5	0.982	0.965	0.983	0.966	0.982	0.965	0.982	0.965	0.010	0.072
k=6	0.982	0.965	0.983	0.966	0.982	0.965	0.982	0.964	0.010	0.073
k=7	0.982	0.965	0.983	0.966	0.982	0.965	0.982	0.965	0.009	0.072
k=8	0.982	0.968	0.983	0.969	0.982	0.968	0.982	0.968	0.010	0.070
k=9	0.982	0.972	0.983	0.973	0.982	0.972	0.982	0.972	0.010	0.071
k=10	0.982	0.968	0.983	0.969	0.982	0.968	0.982	0.968	0.010	0.070
k=11	0.982	0.968	0.983	0.969	0.982	0.968	0.982	0.968	0.010	0.069
k=12	0.982	0.968	0.983	0.969	0.982	0.968	0.982	0.968	0.009	0.071

Table 9: Results Breast Cancer k-NN

Decision Trees: For this dataset decision trees were actually even faster with the training and classification than k-NNs. For, the holdout method performed significantly better than cross-validation, suggesting that the random split with our used seed led to significantly above-average classification performance and also hinting that the decision tree model might be less stable than the k-NN. All decision trees with a depth of at least 7 resulted in models of the same quality. Each of them had an accuracy, precision, recall and f1-score of 1 for holdout, however the results for cross validation were worse than for k-NNs with a best f1-score of 0.933 at a depth of only 3.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
depth=3	0.982	0.933	0.983	0.936	0.982	0.933	0.982	0.933	0.007	0.060
depth=4	0.982	0.926	0.983	0.930	0.982	0.926	0.982	0.926	0.006	0.053
depth=5	1.000	0.919	1.000	0.921	1.000	0.919	1.000	0.918	0.007	0.053
depth=6	0.982	0.916	0.983	0.917	0.982	0.916	0.982	0.914	0.006	0.055
depth=7	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.007	0.052
depth=8	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.007	0.051
depth=9	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.007	0.050
depth=10	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.006	0.050
depth=11	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.007	0.050
depth=12	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.006	0.051
depth=13	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.008	0.059
depth=14	1.000	0.923	1.000	0.924	1.000	0.923	1.000	0.922	0.007	0.056

Table 10: Results Breast Cancer Decision Trees

SVC: Surprisingly, SVC was the were just as fast as decision trees for this dataset. The best-performing configuration was the linear kernel, achieving an f1-score of 1 for holdout and 0.979 for cross-validation. rbf and sigmoid performed only slightly worse, while the polynomial kernel performed poorly. This suggests that the dependencies in the data in breast cancer are somewhat linear.

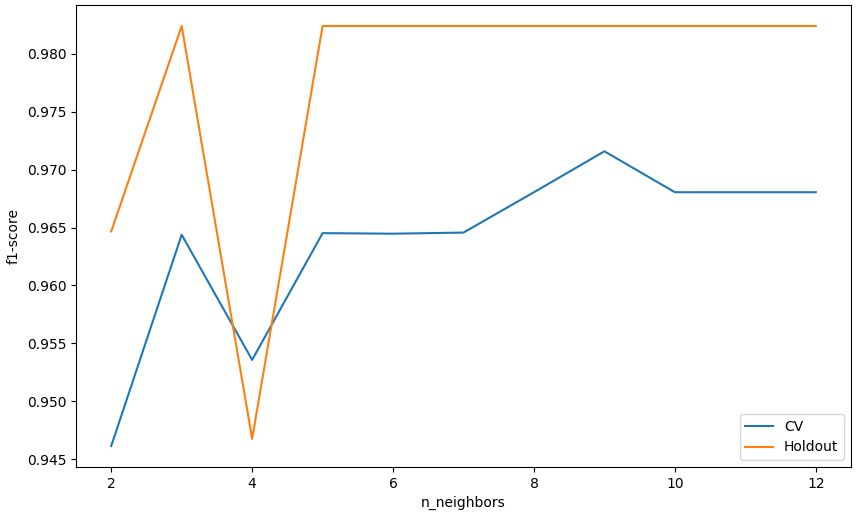
	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
SVC linear	1.000	0.979	1.000	0.979	1.000	0.979	1.000	0.979	0.007	0.055
SVC rbf	1.000	0.975	1.000	0.976	1.000	0.975	1.000	0.975	0.006	0.048
SVC poly	0.930	0.926	0.937	0.934	0.930	0.926	0.929	0.923	0.005	0.044
SVC sigmoid	1.000	0.947	1.000	0.949	1.000	0.947	1.000	0.947	0.004	0.046

Table 11: Results Breast Cancer SVC

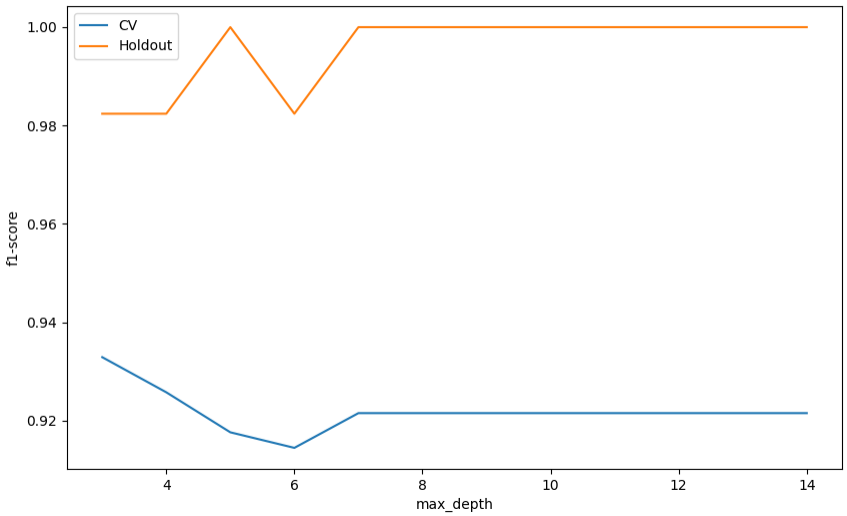
Overall, while all classifiers achieved good results, the SVC with a linear kernel produced the best results out of the three classifiers. Figure 5 shows how the performance in terms of f1-score changes for different parameters settings for the three classifiers for both cross-validation and holdout, while Figure 6 shows the confusion matrix of the best-performing model of each classifier.

3.4 Loan

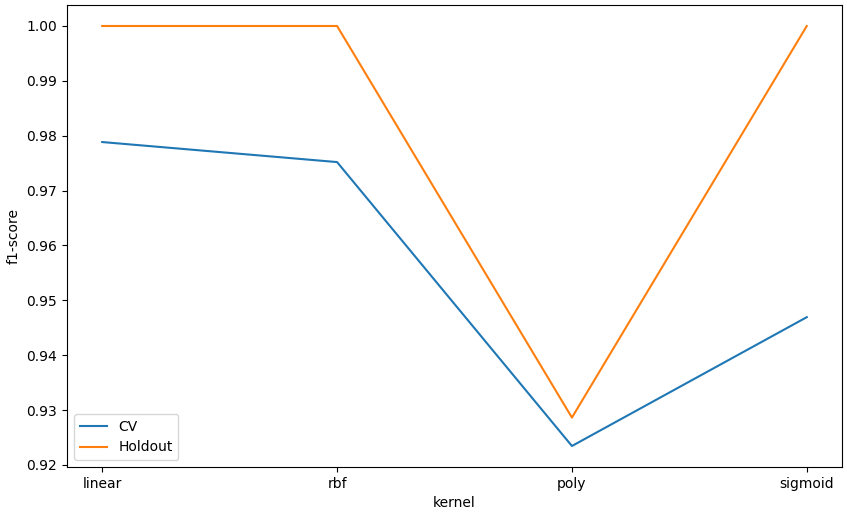
The experimental results for the different classifiers on the loan dataset are reported in Table 12, 13 and 14 respectively. While the loan dataset was nowhere near as large as porto-seguro, it was nonetheless significantly larger than HPC and breast-cancer. Therefore, while training and classification still took only



(a) k-NN

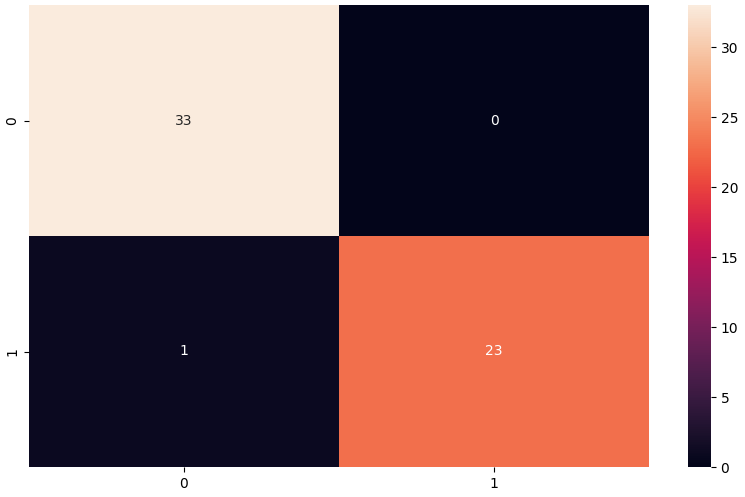


(b) Decision Trees

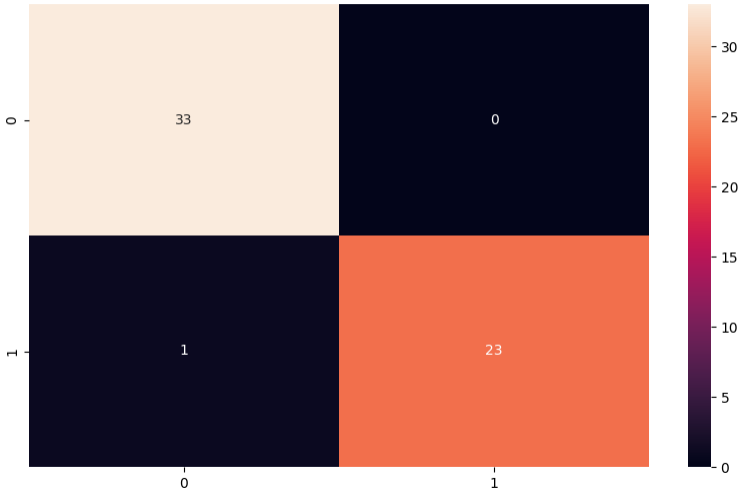


(c) SVC

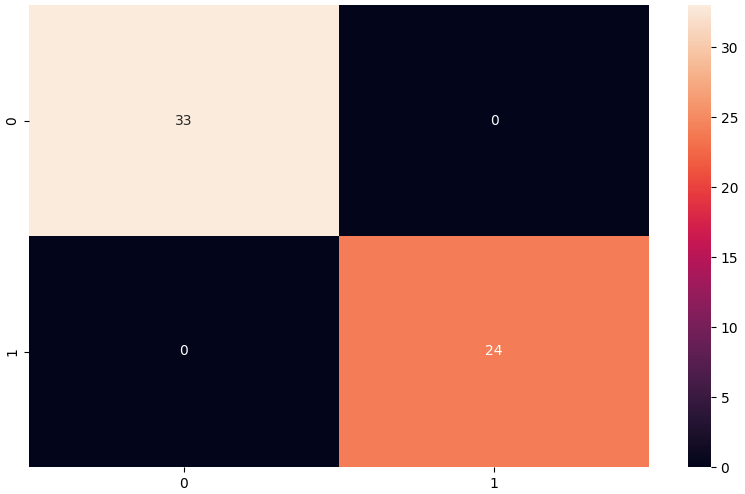
Figure 5: F1-score for different parameter settings for the three classifiers on the Breast Cancer dataset



(a) k-NN



(b) Decision Trees



(c) SVC

Figure 6: Confusion Matrix of the best model for each of the three classifiers on the Breast Cancer dataset

about a second or less for k-NNs or Decision Trees, SVC took around 8 to 17 seconds. Further, in this case, the prediction metrics for cross-validation and holdout are again somewhat similar, with cross-validation generally performing slightly better, suggesting that either the resulting seed was relatively average or that the models were more stable for this dataset.

k-NN: The k-NNs performed very poorly in this multi-class classification task. The model with the best f1-score (using cross-validation) in our experiments was the model with $k = 11$ and only had an accuracy of 0.469, precision of 0.445, recall of 0.469, and an f1-score of 0.447, meaning no even half of the samples were classified correctly. Therefore, while training time was the lowest of the three methods, k-NNs seemed to be ineffective for this multi-classification task.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
k=2	0.423	0.424	0.406	0.407	0.423	0.424	0.396	0.397	0.066	0.422
k=3	0.415	0.414	0.399	0.403	0.415	0.414	0.398	0.398	0.061	0.431
k=4	0.442	0.451	0.422	0.434	0.442	0.451	0.424	0.435	0.061	0.438
k=5	0.448	0.449	0.424	0.437	0.448	0.449	0.427	0.433	0.065	0.436
k=6	0.452	0.450	0.431	0.439	0.452	0.450	0.432	0.433	0.062	0.449
k=7	0.456	0.458	0.435	0.439	0.456	0.458	0.437	0.440	0.066	0.454
k=8	0.468	0.464	0.449	0.450	0.468	0.464	0.447	0.445	0.065	0.456
k=9	0.460	0.465	0.439	0.445	0.460	0.465	0.439	0.445	0.065	0.466
k=10	0.465	0.466	0.446	0.444	0.465	0.466	0.445	0.445	0.060	0.451
k=11	0.462	0.469	0.441	0.445	0.462	0.469	0.441	0.447	0.065	0.465
k=12	0.467	0.467	0.451	0.443	0.467	0.467	0.446	0.445	0.063	0.461

Table 12: Results Loan k-NN

Decision Trees: Decision trees performed much better than k-NNs on the loan dataset. The model with the best performance in our experiments was the model with a depth of 9 and had an accuracy of 0.916, precision of 0.926, recall of 0.926 and an f1-score of 0.925. While the training and classification took longer than for k-NNs, it still generally finished in around a second.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
depth=3	0.854	0.868	0.838	0.851	0.854	0.868	0.843	0.858	0.110	0.590
depth=4	0.879	0.886	0.871	0.882	0.879	0.886	0.873	0.881	0.113	0.680
depth=5	0.891	0.895	0.894	0.898	0.891	0.895	0.888	0.890	0.127	0.759
depth=6	0.914	0.910	0.917	0.914	0.914	0.910	0.912	0.909	0.139	0.825
depth=7	0.922	0.920	0.922	0.920	0.922	0.920	0.921	0.919	0.150	0.871
depth=8	0.915	0.919	0.914	0.920	0.915	0.919	0.915	0.918	0.169	0.915
depth=9	0.916	0.926	0.916	0.926	0.916	0.926	0.915	0.925	0.167	0.963
depth=10	0.914	0.925	0.914	0.925	0.914	0.925	0.914	0.925	0.174	0.999
depth=11	0.910	0.922	0.909	0.922	0.910	0.922	0.910	0.922	0.178	1.022
depth=12	0.907	0.921	0.907	0.921	0.907	0.921	0.907	0.921	0.182	1.025
depth=13	0.910	0.919	0.911	0.919	0.910	0.919	0.910	0.919	0.182	1.043
depth=14	0.913	0.918	0.913	0.919	0.913	0.918	0.913	0.918	0.185	1.046

Table 13: Results Loan Decision Trees

SVC: Finally, SVCs took significanly longer for training and classification, but did not produce results that were as good as the decision trees. The model that performed best was again the linear kernel with cross validation with an accuracy of 0.868, precision of 0.866, recall of 0.868 and f1-score of 0.866. While all kernels performed better than the k-NNs, some still performed rather poorly, e.g. the simoid kernel getting an f1-score below 0.6 for cross validation.

	accuracy		precision		recall		f1-score		timing	
Parameters	holdout	cv	holdout	cv	holdout	cv	holdout	cv	holdout	cv
SVC linear	0.862	0.868	0.860	0.866	0.862	0.868	0.860	0.866	1.538	7.928
SVC rbf	0.806	0.821	0.795	0.806	0.806	0.821	0.798	0.812	3.374	16.773
SVC poly	0.705	0.703	0.704	0.702	0.705	0.703	0.692	0.690	3.123	16.187
SVC sigmoid	0.608	0.601	0.603	0.596	0.608	0.601	0.605	0.596	1.783	9.171

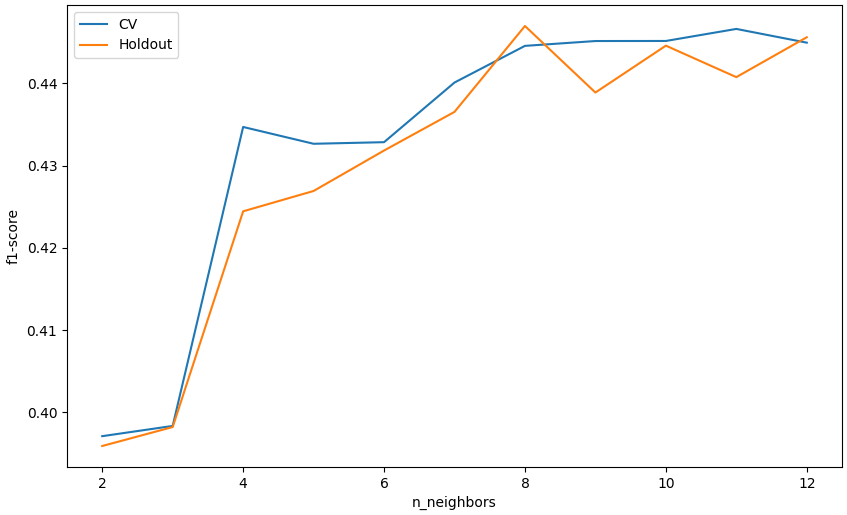
Table 14: Results Loan SVC

Overall, the best-performing model for the loan dataset was the Decision Tree with a maximal depth of 9, while k-NNs and SVCs achieved significantly worse results than decision trees. Figure 7 shows how the

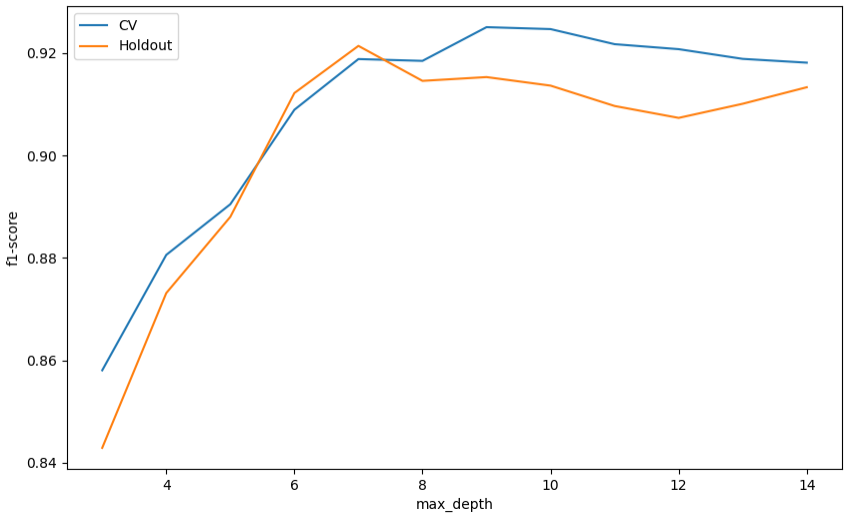
performance in terms of f1-score changes for different parameters settings for the three classifiers for both cross-validation and holdout, while Figure 8 shows the confusion matrix of the best performing model of each classifier.

4 Conclusion

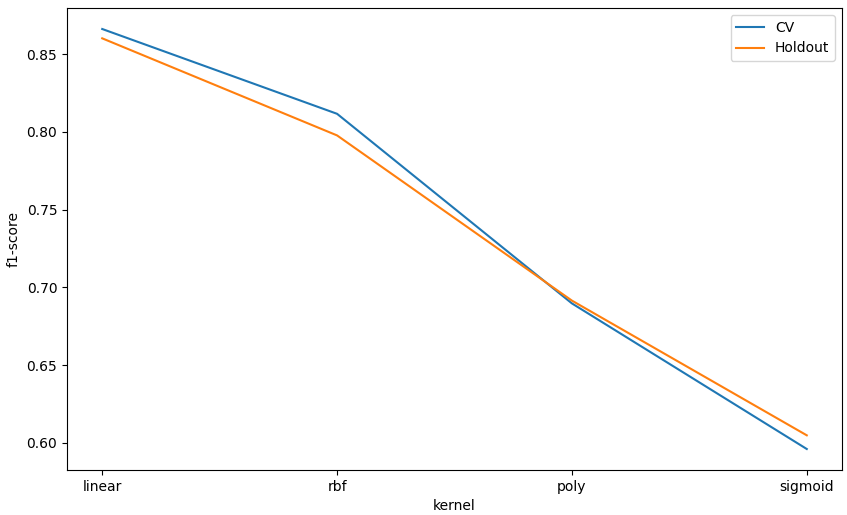
Our experiments nicely showed that for different datasets different classifiers can achieve the best performance. SVCs performed the best for Breast Cancer and HPC, while Decision trees performed the best for Porto Seguro, HPC, and Loan. k-NNs were the second best performing in various data sets, fastest to train/classify for loan and often more stable than decision trees. Further, we noticed that encoding two separate time-related parameters into a single timestamp parameter can be beneficial to performance. Finally, various of our experiments showed that increasing the depth of decision trees or the value of k for k-NNs to the maximum does not always lead to the best performance.



(a) k-NN

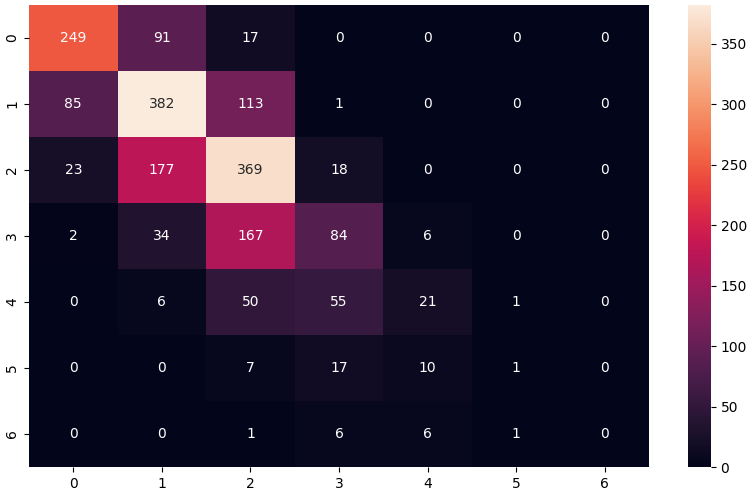


(b) Trees

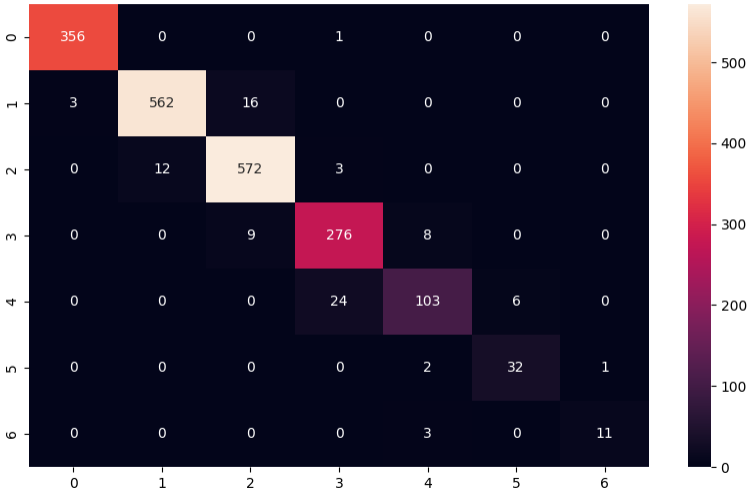


(c) SVC

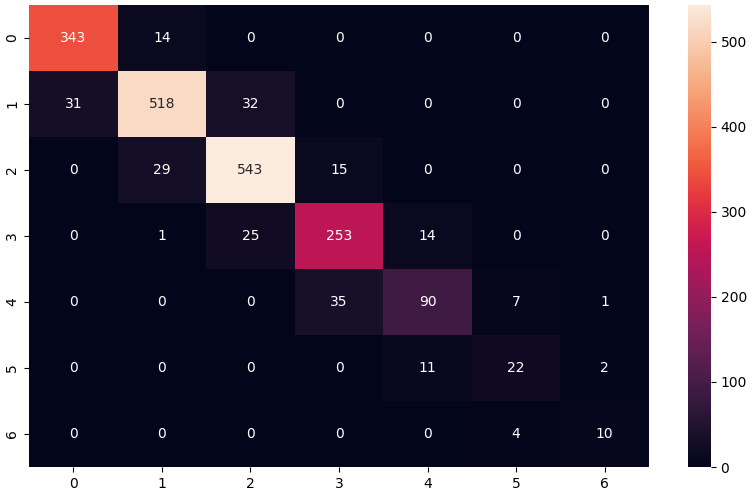
Figure 7: F1-score for different parameter settings for the three classifiers on the Loan dataset



(a) k-NN



(b) Decision Trees



(c) SVC

Figure 8: Confusion Matrix of the best model for each of the three classifiers on the Loan dataset