Applied Machine Learning (COMP 551)
Mini-Project 1

# Machine Learning 101

*Authors:*
Younggue Kim
Nikita Letov
Negin Moslemi

*Course Instructor:*
Dr. Reihaneh Rabbany
Dr. Mohsen Ravanbakhsh

**McGill University**
Montreal, QC

February, 2020

## Abstract

The objective of this project is to apply some basic techniques which are broadly used in machine learning. Important outcome of this project is obtaining a skill to choose the most effective method and its parameters. The methods we consider in this paper are Logistic Regression and Naive Bayes. They are widely used for comparing performances of different models. We address the performance of the two methods on four different datasets. The following steps were followed to evaluate the performance. First, the datasets had to be pre-processed. The datasets were cleaned and the statistics were analyzed. The data was split into test and train datasets and the occurrence of classes was derived. Moreover, some meaningful data has been plotted to see potential dependencies. Secondly, the models for both Logistic Regression and Naive Bayes were trained for all four train datasets. Then the predicted outcome of the models was obtained from the test datasets and compared to the actual test target values. The Naive Bayes and the Logistic Regression were compared in several parameters such as their accuracy and performance speed.

## 1. Introduction

The datasets that are used in this work are the Ionosphere dataset, the Adult dataset, the Chess (King-Rook vs. King-Pawn) dataset, and the Tic-Tac-Toe Endgame dataset, all taken from the UCI Machine Learning Repository [1]. The Logistic Regression was applied to all four datasets. However, the datasets have different nature and types of features and thus the Naive Bayes were applied differently. The Ionosphere dataset is continuous and was modelled with the Gaussian Naive Bayes, while the Chess (King-Rook vs. King-Pawn) dataset and the Tic-Tac-Toe Endgame dataset are categorical and were modelled with the Categorical Naive Bayes. Finally, the Adult dataset has both continuous and categorical features and thus the mixture of the both Naive Bayes methods. All four datasets have binary target features. In order to see whether the implemented approaches produce correct results, the outputs of the learning methods for the Ionosphere dataset and the Adult dataset were compared to the research previously done by Ng and Jordan [2]. Similarly, for the Chess (King-Rook vs. King-Pawn) dataset the work of Cheng and Greiner and the work of Greiner et al. are referred [3, 4], and for the Tic-Tac-Toe Endgame dataset the work of Aha is referred [5].

## 2. Datasets

In this section of the work the acquiring, preprocessing, and analyzing the data are described.

### 2.1. Ionosphere dataset

As was mentioned, this dataset comes as a table of continuous data with 34 feature columns and 1 target column. In the original dataset the columns are not named, so it was decided to name them appropriately for convenience. The dataset description mentions that every two columns are two readings from the same antenna. With this in mind, the feature columns are called *Pulse 1.1*, *Pulse 1.2*, *Pulse 2.1*, *Pulse 2.2*, ..., *Pulse 17.1*, and *Pulse 17.2*. The target column is called *Structure detected* as the readings should indicate whether there is some structure in the air or not, and has values *good* or *bad*. The dataset consists of 351 instances with no missing values. Moreover, the dataset was tested for data type inconsistency, e.g. cases when one column has attributes of more than one data type. It was shown that the dataset has no data type inconsistency or missing values. For the first train, the dataset was split into train and test datasets with 80% belonging to the train dataset. From the train dataset it was obtained that approximately 65% of the instances have *good* target values.

It was noted that all features corresponding to *Pulse 1.2* are the same and equal to 1. Therefore, this feature was dropped.

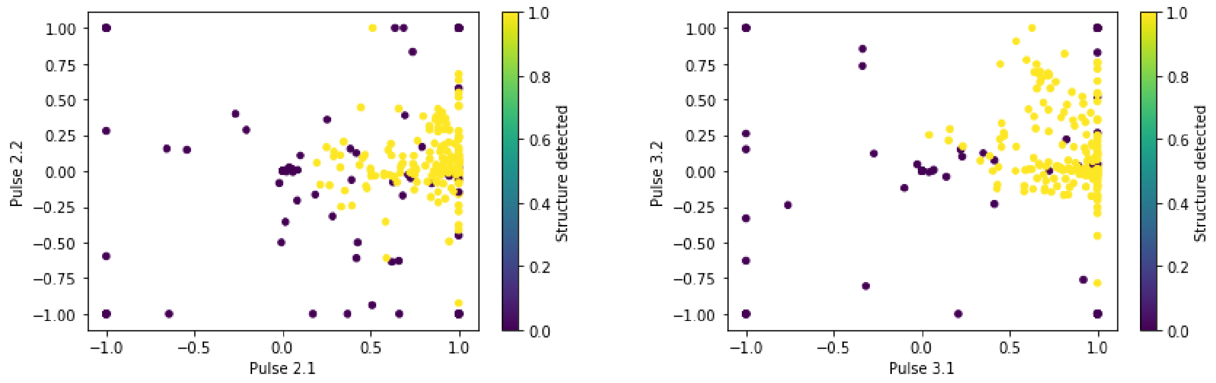

**Figure 2.1.** Some of the scatter plots of some of the pulses from the Ionosphere dataset.

Some scatter plots of Pulses i.1 and i.2 for all i except 1 (as *Pulse 1.2* was dropped) have been plotted to see potential dependencies and are illustrated in Figure 2.1. Note that the *good* class tends to gather around the right edge of some of the scatter plots, i.e. higher values of *Pulse i.1* seem to normally correspond to the *good* class.

## 2.2. Adult dataset

This dataset comes as a table of integer( e.g age, hours per week) and categorical data( e.g marital-status or sex) with 14 features column and 1 target column which is shown as class. Prediction task is to determine whether a person makes over 50k a year or not. There are some missing values which are shown as "?". We replace them with Nan values and then drop them. To realize the distribution of each feature in a better way, we actually plot them . Below are some of the plots. Actually we suspect 3 features. The first feature is 'native-country' which we realize approximately %99 of the instances are from the US. Those two features are 'capital-gain' and 'capital-loss' which has 0 value for the %99.91 and %98.92 of the instances respectively. As we mentioned, we have some categorical data here that we convert them to binary features by One-hot Encoding, so as a result our dataset consists of 54 features. We also change the class from '>=50k' to '1' and from '<50k' to '0'. We recognize that the data in each column are not in the same range of other ones so we use Feature scaling. We already have seperated train data and test data. So we do all of these actions for both training and test data.

## 2.3. Chess (King-Rook vs. King-Pawn) dataset

This dataset comes as a table of categorical data with 36 feature columns and 1 target column. The feature columns are called *f*, *f.1*, *f.2*, ..., *f.28*, *l*, *n*, *n.1*, *t*, *t.1*, *t.2*, *t.3*. The target column is called *won* and indicates whether the game has a chance to be won or not, and has values *won* or *nowin*. The dataset consists of 3195 instances with no missing values. Moreover, the dataset was tested for data type inconsistency, e.g. cases when one column has attributes of more than one data type. It was shown that the dataset has no data type inconsistency or missing values. For the first train, the dataset was split into train and test datasets with 80% belonging to the train dataset. From the train dataset it was obtained that approximately 52% of the instances have *won* target values.

It was noted that all features corresponding to *f.1*, *f.2*, *f.12*, *f.13*, *f.14*, *f.15*, *f.19*, *f.21*, *f.22*, *f.23*, *f.24*, *f.25* have a significant amount (over 90%) of same features. Therefore, these features were dropped.

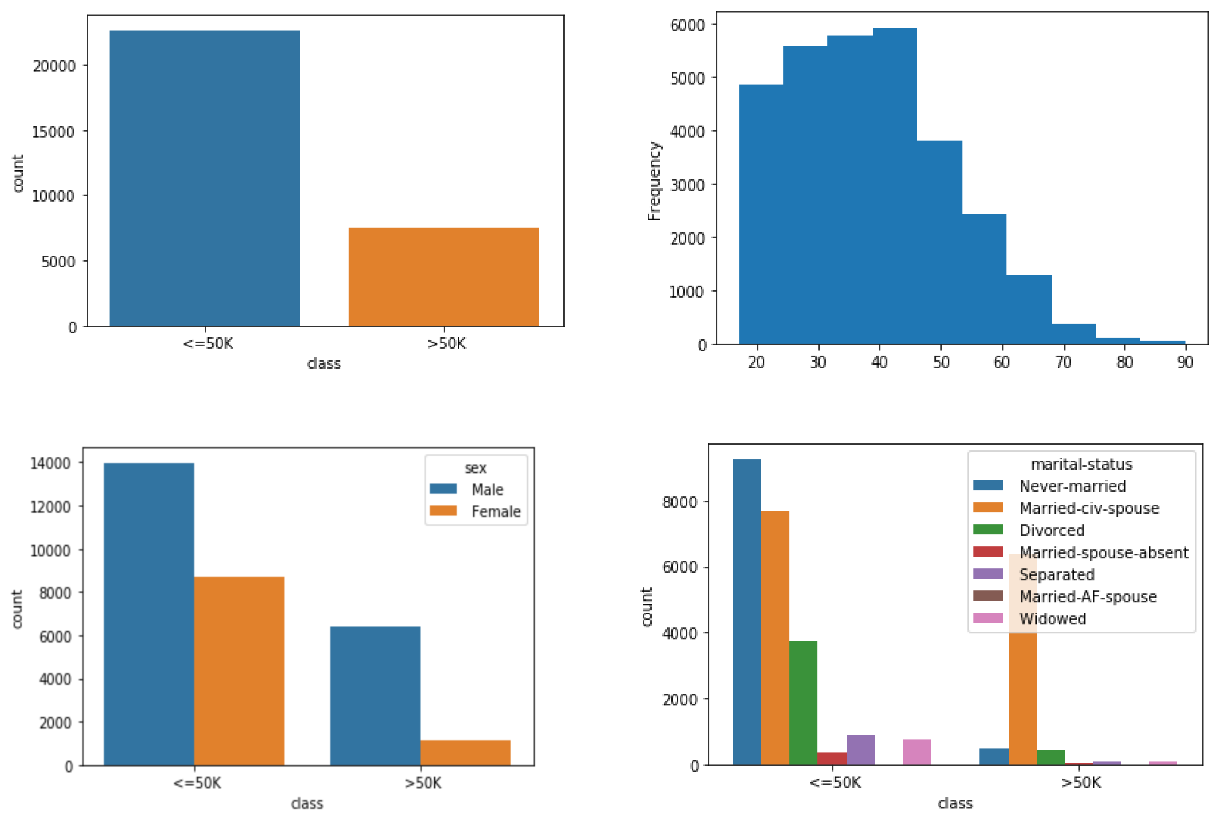

**Figure 2.2.** Plots of the features from the Adult dataset.

Some plots of features in the dataset have been plotted to see potential dependencies and are illustrated in Figure 2.3. Note that the chance to win seems to lower down with each new turn, so it is better to win fast.

## 2.4. Tic-Tac-Toe dataset

This dataset comes as a table of categorical data with 9 feature columns and 1 target column. The feature columns are called *x*, *x.1*, *x.2*, *x.3*, *o*, *o.1*, *x.4*, *o.2*, *o.3*. The target column is called *positive* and indicates whether the game is won or not, and has values *positive* or *negative*. The dataset consists of 957 instances with no missing values. Moreover, the dataset was tested for data type inconsistency, e.g. cases when one column has attributes of more than one data type. It was shown that the dataset has no data type inconsistency or missing values. For the first train, the dataset was split into train and test datasets with 80% belonging to the train dataset. From the train dataset it was obtained that approximately 65% of the instances have *won* target values.

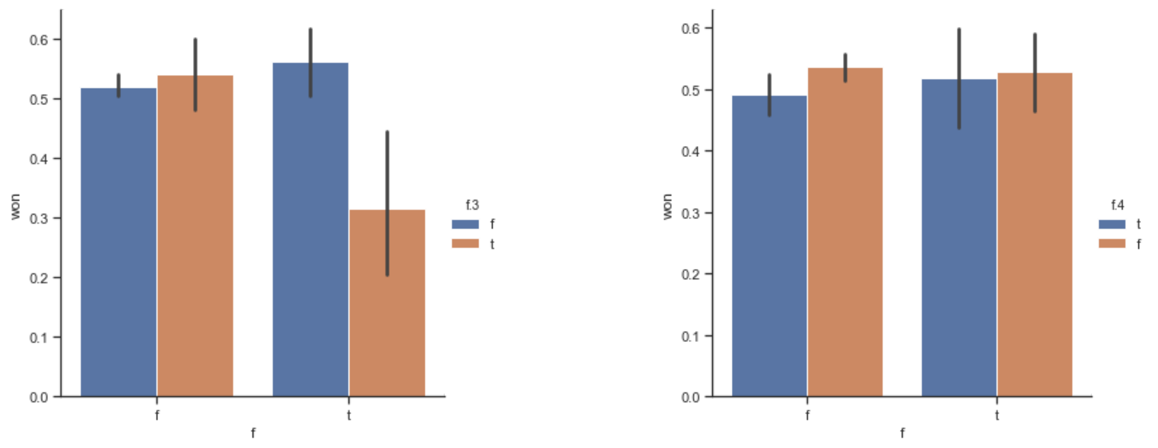

**Figure 2.3.** Some of the scatter plots of some of the features from the Chess dataset.

No feature has a significant amount (over 90%) of the same values. Therefore, no feature was dropped.

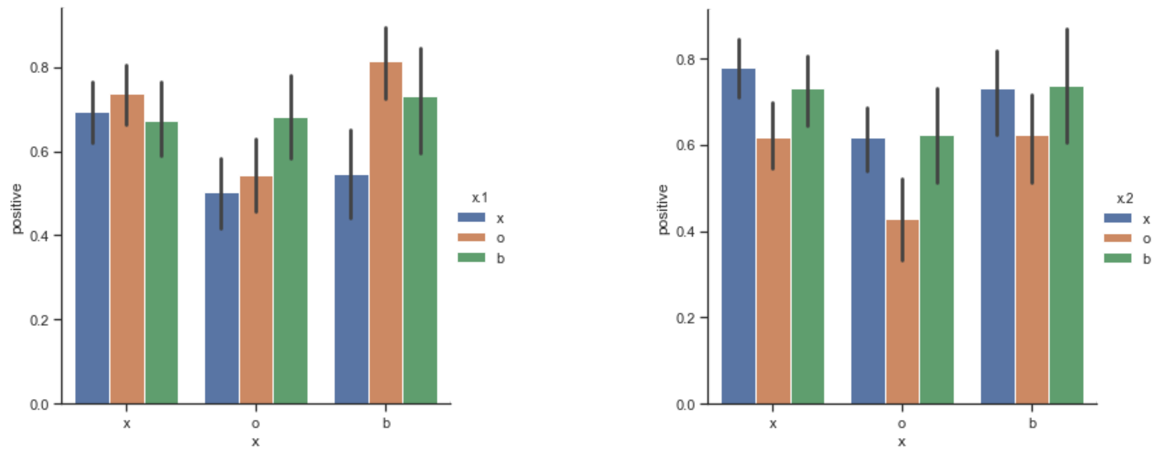

**Figure 2.4.** Plots of the features from the Tic-Tac-Toe dataset.

The features in the dataset have been plotted to see potential dependencies and are illustrated in Figure 2.4. Note that whoever goes first in a Tic-Tac-Toe game and puts his first mark in the very middle cell usually wins.

## 3. Results

In this section of the work the implementation of both models for each dataset are described, the implementation of the k-fold cross validation, and the results of the carried experiments. For all cases the number of folds for the k-fold cross validation is 5.

### 3.1. Ionosphere dataset

The implementation of the Logistic Regression is relatively straightforward for this dataset. As all features are continuous, the Gaussian Naive Bayes was implemented for this dataset.

#### 3.1.1. k-fold cross-validation test

The k-fold cross-validation of the Logistic Regression and the Gaussian Naive Bayes showed the accuracy of 85% and 70%, respectively. The learning rate of 0.01 and the termination condition of 0.01 was used for the gradient descent of the Logistic Regression. Note that the accuracy of the Logistic Regression is higher than of Naive Bayes, which comes at a cost as the Gaussian Naive Bayes performs significantly faster than the Logistic Regression (0.5 s vs 74 s).

**3.1.2. Test of different learning rates for the gradient descent**

For this test the values of learning rates $10^{-3}$, $10^{-2}$, $10^{-1}$, $10^{0}$ were taken while keeping the termination criteria 0.01 the same. Note that for the learning rate greater than 1 the gradient descent overshoots and does not converge and for the learning rate lower than $10^{-3}$ the computation is too slow and seems like an overkill. The accuracy of the method is not greatly affected by the lowering of the learning rate, but the computations become significantly slow.

The accuracy of the Logistic Regression depending on the number of iterations of the gradient descent method used in logistic regression was analyzed and the results are illustrated in Figure 3.1.1. The maximum number of iterations is set to 1000. Time expenses are close to linear when increasing the learning rates except for some cases where the algorithm seems to overshoot.

The dependence of the error rate of both learning methods on the test/train split ratio is analyzed for both Logistic Regression and Gaussian Naive Bayes and shown in Figure 3.1.2 and Figure 3.1.3, respectively.

For the Logistic Regression the error asymptotically goes down when we use more data for training until the very end where the test data becomes insufficient. Similarly, for the Naive Bayes the error goes down when we use more data for training until the very end Naive Bayes seems to arrive at the asymptote at ~0.2 faster than the Logistic Regression.

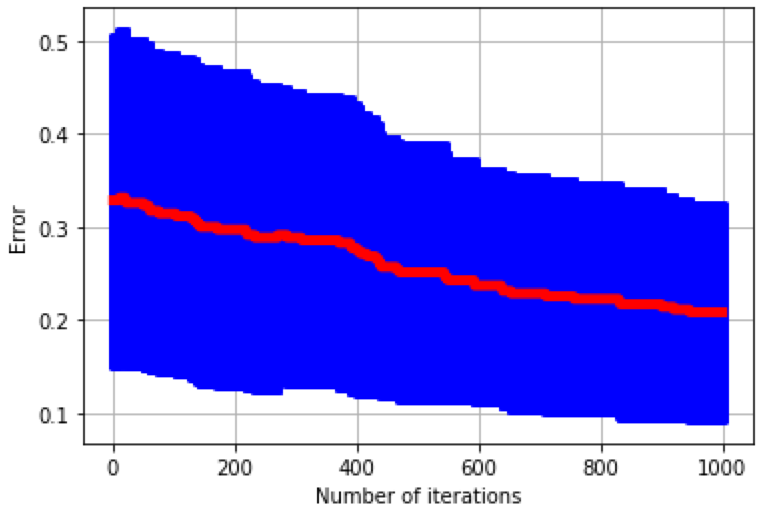

**Figure 3.1.1.** Accuracy of the Logistic Regression on the Ionosphere dataset depending on the number of iterations in the gradient descent with the mean in red and the standard deviation in blue.

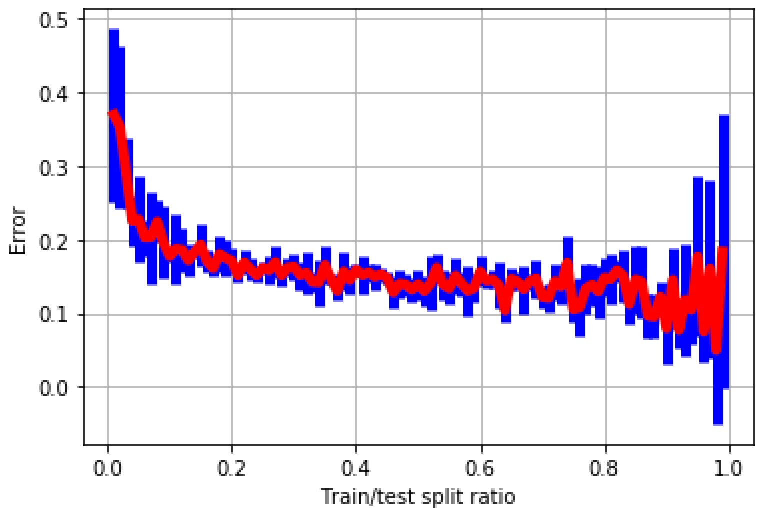

**Figure 3.1.2.** Accuracy of the Logistic Regression on the Ionosphere dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

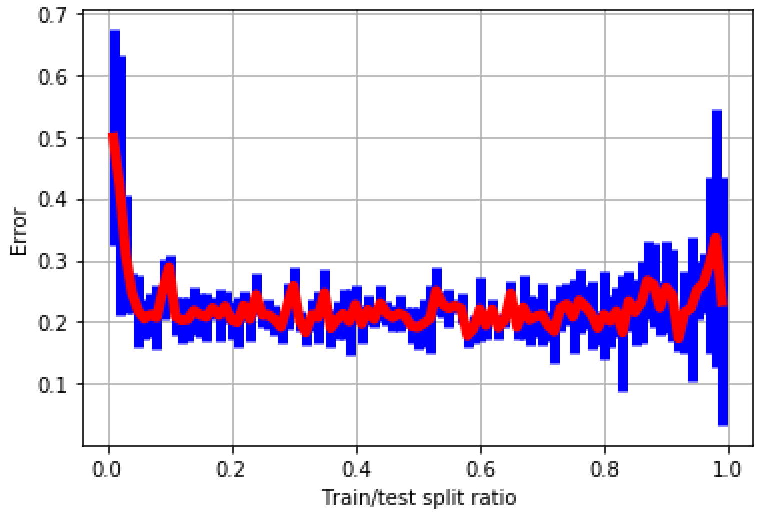

**Figure 3.1.3.** Accuracy of the Gaussian Naive Bayes on the Ionosphere dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

## 3.2. Adult dataset

The implementation of the Logistic Regression is relatively straightforward for this dataset. As the features are mixed with Continuous and Categorical features, the Mixed Naive Bayes was implemented for this dataset.

### 3.2.1. k-fold cross-validation test

The k-fold cross-validation of the Logistic Regression and the Mixed Naive Bayes showed the accuracy of 82% and 81%, respectively. The learning rate of 0.01 and the termination condition of 0.01 was used for the gradient descent of the Logistic Regression. Note that the accuracy of the Logistic Regression is higher than of Naive Bayes, which comes at a cost as the Mixed Naive Bayes performs significantly faster than the Logistic Regression .

**3.1.2. Test of different learning rates for the gradient descent**

For this test the values of learning rates $10^{-3}$, $10^{-2}$, $10^{-1}$, $10^{0}$ were taken while keeping the termination criteria 0.01 the same. Note that for the learning rate greater than 1 the gradient descent overshoots and does not converge and for the learning rate lower than $10^{-3}$ the computation is too slow and seems like an overkill. The accuracy of the method is not greatly affected by the lowering of the learning rate, but the computations become significantly slow.

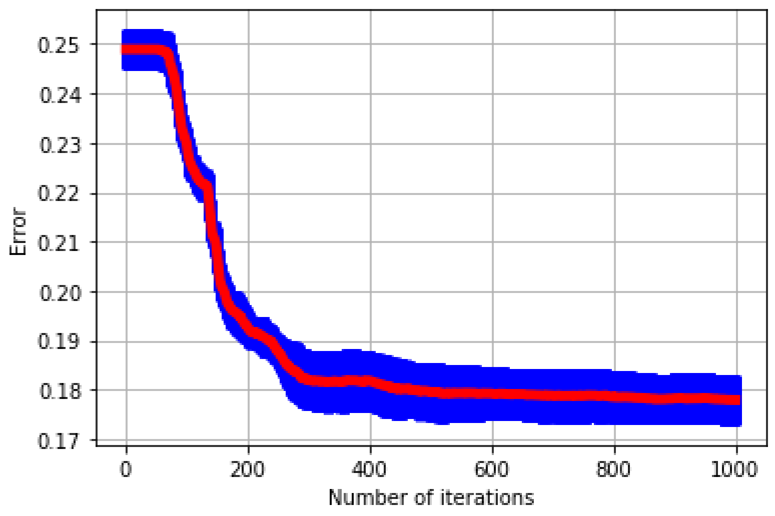

**Figure 3.2.1.** Accuracy of the Logistic Regression on the adult dataset depending on the number of iterations in the gradient descent with the mean in red and the standard deviation in blue.

The accuracy of the Logistic Regression depending on the number of iterations of the gradient descent method used in logistic regression was analyzed and the results are illustrated in Figure 3.2.1. The maximum number of iterations is set to 1000. Time expenses are close to linear when increasing the learning rates except for some cases where the algorithm seems to overshoot.

The dependence of the error rate of both learning methods on the test/train split ratio is analyzed for Mixed Naive Bayes and shown in Figure Figure 3.2.2.

For the Logistic Regression the error asymptotically goes down when we use more data for training until the very end where the test data becomes insufficient. Similarly, for the Naive Bayes the error goes down when we use more data for training until the very end Naive Bayes seems to arrive at the asymptote at ~0.2 faster than the Logistic Regression.

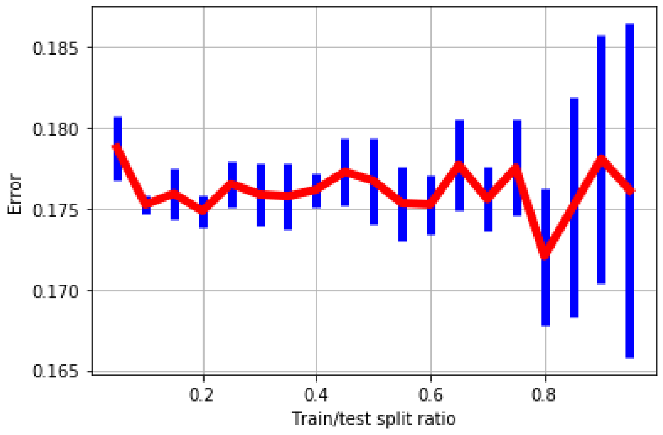

**Figure 3.2.2.** Accuracy of the Mixed Naive Bayes on the adult dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

## 3.3. Chess (King-Rook vs. King-Pawn) dataset

The implementation of the Logistic Regression is relatively straightforward for this dataset. As all features are categorical, the Categorical Naive Bayes was implemented for this dataset.

### 3.3.1. k-fold cross-validation test

The k-fold cross-validation of the Logistic Regression and the Categorical Naive Bayes showed the accuracy of 76% and 53%, respectively. The learning rate of 0.01 and the termination condition of 0.01 was used for the gradient descent of the Logistic Regression. Note that the accuracy of the Logistic Regression is higher than of Naive Bayes, which comes at a cost as the Gaussian Naive Bayes performs significantly faster than the Logistic Regression (0.3 s vs 187 s).

### 3.3.2. Test of different learning rates for the gradient descent

For this test the values of learning rates $10^{-3}$, $10^{-2}$, $10^{-1}$, $10^{0}$, $10^{1}$ were taken while keeping the termination criteria 0.01 the same. Note that for the learning rate greater than 10 the gradient descent overshoots and does not converge and for the learning rate lower than $10^{-3}$ the computation is too slow and seems like an overkill. The accuracy of the method is somewhat affected by the lowering of the learning rate, but the computations become significantly slow.

The accuracy of the Logistic Regression depending on the number of iterations of the gradient descent method used in Logistic Regression was analyzed and the results are illustrated in Figure 3.3.1. The maximum number of iterations is set to 1000. Note that the mean of accuracy grows with the number of iterations its standard deviation shrinks. Also note that for this dataset a significant amount of iterations of the gradient descent algorithm is required to perform well. Time expenses are close to linear when increasing the learning rates except for some cases where the algorithm seems to overshoot.

The dependence of the error rate of both learning methods on the test/train split ratio is analyzed for both Logistic Regression and Categorical Naive Bayes and shown in Figure 3.3.2 and Figure 3.3.3, respectively.

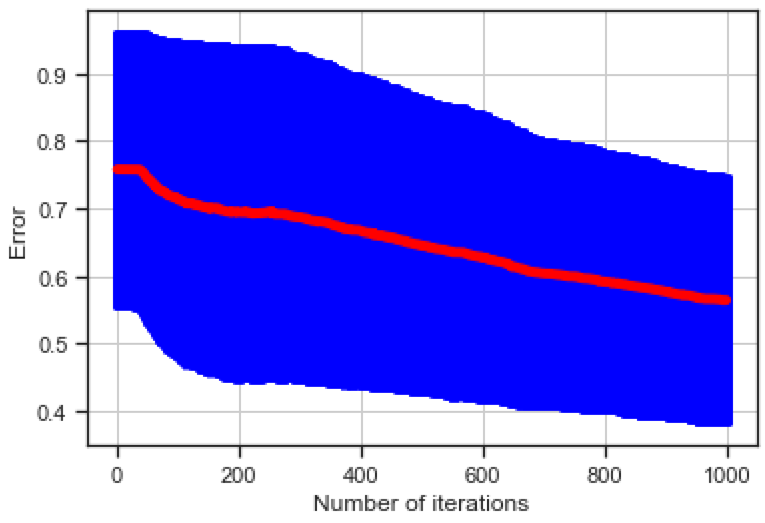

**Figure 3.3.1.** Accuracy of the Logistic Regression on the Chess dataset depending on the number of iterations in the gradient descent with the mean in red and the standard deviation in blue.

For the Logistic Regression the error asymptotically goes down to approximately 5% when we use more data for training until the very end where the test data becomes insufficient. Similarly, for the Naive Bayes the error goes down when we use more data for training until the very end Naive Bayes seems to arrive at the asymptote at ~14% faster than the Logistic Regression but its accuracy is lower.

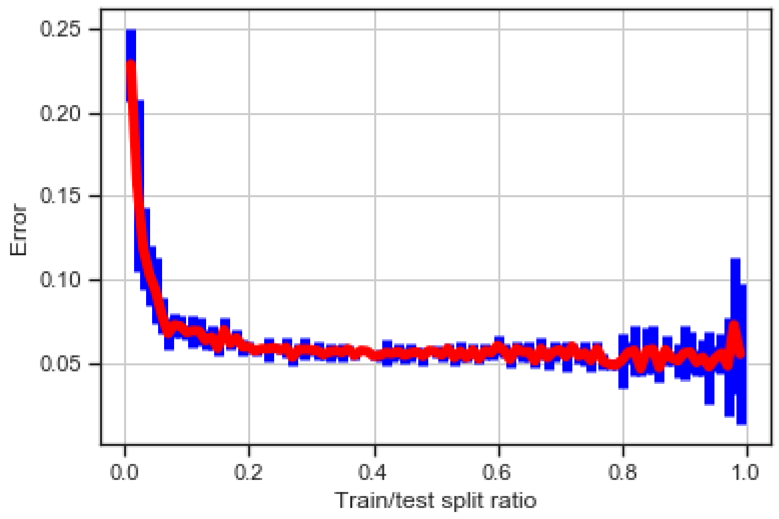

**Figure 3.3.2.** Accuracy of the Logistic Regression on the Chess dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

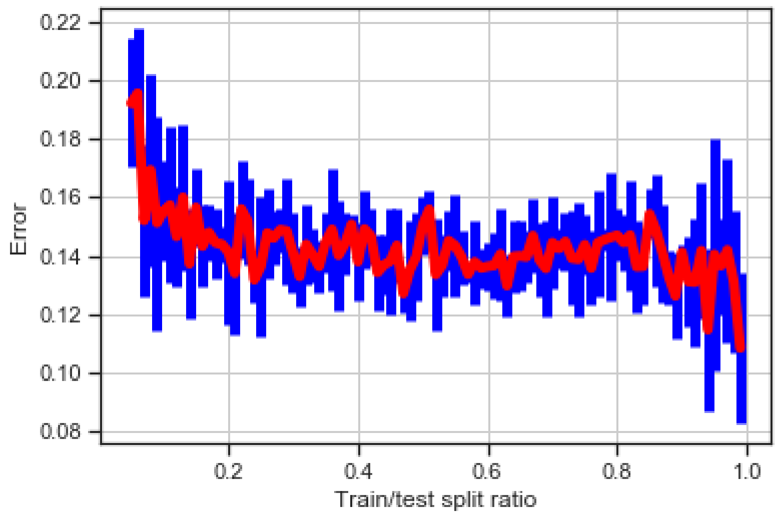

**Figure 3.3.3.** Accuracy of the Categorical Naive Bayes on the Chess dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

## 3.4. Tic-Tac-Toe dataset

The implementation of the Logistic Regression is relatively straightforward for this dataset. As all features are categorical, the Categorical Naive Bayes was implemented for this dataset.

### 3.4.1. k-fold cross-validation test

The k-fold cross-validation of the Logistic Regression and the Categorical Naive Bayes both showed the accuracy of 70%. The learning rate of 0.01 and the termination condition of 0.001 were used for the gradient descent of the Logistic Regression. Note that the Categorical Naive Bayes performs significantly faster than the Logistic Regression (0.1 s vs 43 s).

### 3.3.2. Test of different learning rates for the gradient descent

For this test the values of learning rates $10^{-3}$, $10^{-2}$, $10^{-1}$ were taken while keeping the termination criteria 0.01 the same. Note that for the learning rate greater than 0.1 the gradient descent overshoots and does not converge and for the learning rate lower than $10^{-3}$ the computation is too slow and seems like an overkill. The mean of the accuracy of the method is not significantly affected by the lowering of the learning rate but affects the standard deviation, but the computations become significantly slow.

The accuracy of the Logistic Regression depending on the number of iterations of the gradient descent method used in Logistic Regression was analyzed and the results are illustrated in Figure 3.4.1. The maximum number of iterations is set to $10^5$. Note that the mean of accuracy grows with the number of iterations its standard deviation shrinks. Time expenses are close to linear when increasing the learning rates except for some cases where the algorithm seems to overshoot.

The dependence of the error rate of both learning methods on the test/train split ratio is analyzed for both Logistic Regression and Categorical Naive Bayes and shown in Figure 3.4.2 and Figure 3.4.3, respectively.

For the Logistic Regression the error goes down to approximately 30% when we use more data for training until the very end where the test data becomes insufficient. For the Naive Bayes the error is almost the same and stays at ~30%, but the Naive Bayes is faster than the Logistic Regression.

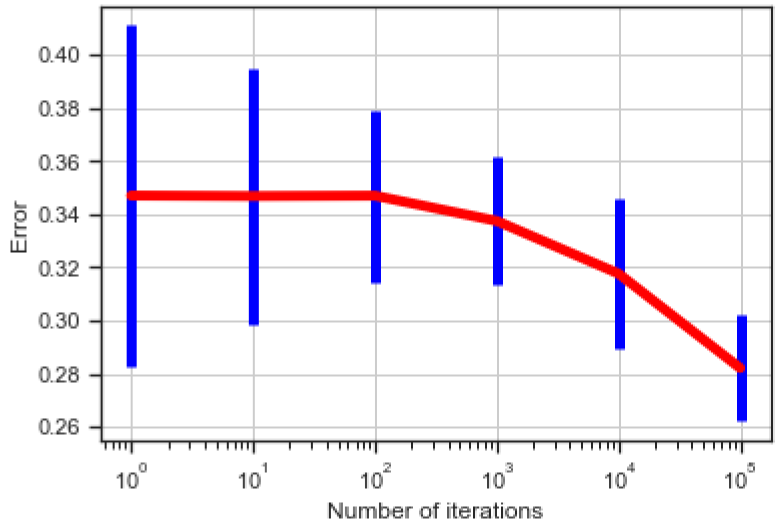

**Figure 3.4.1.** Accuracy of the Logistic Regression on the Tic-Tac-Toe dataset depending on the number of iterations in the gradient descent with the mean in red and the standard deviation in blue.

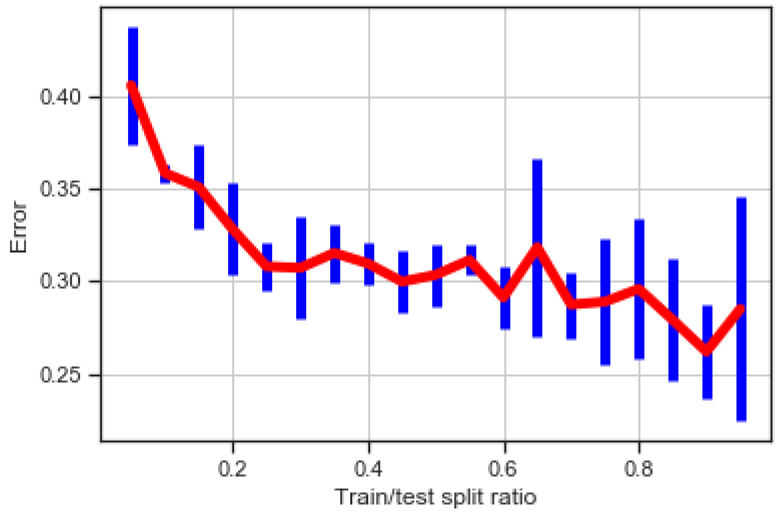

**Figure 3.4.2.** Accuracy of the Logistic Regression on the Tic-Tac-Toe dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

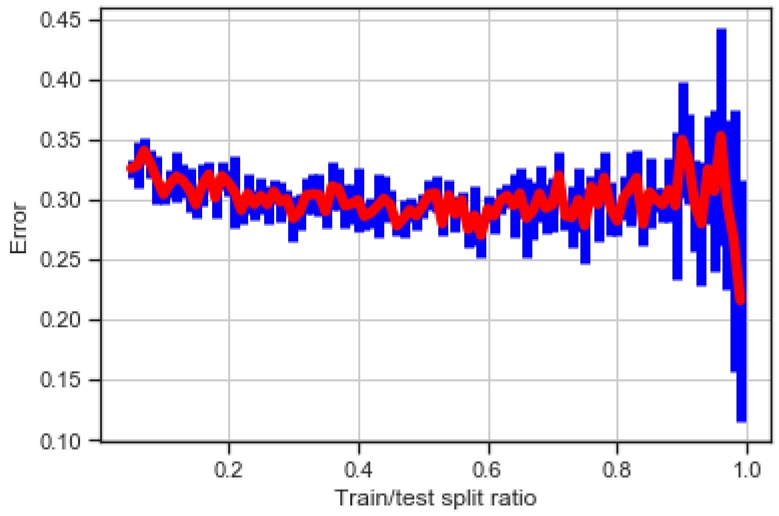

**Figure 3.4.3.** Accuracy of the Categorical Naive Bayes on the Tic-Tac-Toe dataset depending on the test/train split ratio with the mean in red and the standard deviation in blue.

## 4. Discussion and conclusion

The tests performed for the datasets showed differences in two learning approaches: the Logistic Regression and the Naive Bayes. While the Logistic regression tries to converge to an optimal set of weights that describe the model in the best possible way, the Naive Bayes is a purely probabilistic approach that estimates posterior probability from prior probability, likelihoods, and the marginal probability. We see that the Naive Bayes predicts target values reasonably while doing it fast. The Logistic Regression is significantly much slower but provides more reliable results in general. The resulting plots of the error ratios for the considered dataset appear in high consistency with the results obtained in relevant research [2-5].

## 5. Statement of contributions

It was decided to follow the agile principles as the amount of coding is significant and had to be developed by a team of three. Thus, the code is stored in a private GitHub repository. Every team member was able to contribute to the repository independently from others and committed equally.

## References

[1] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[2] Ng, A. Y., and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In Advances in neural information processing systems (pp. 841-848).

[3] Cheng, J., and Greiner, R. (2001). *Learning Bayesian Belief Network Classifiers: Algorithms and System*. https://doi.org/10.1007/3-540-45153-6_14

[4] Greiner, R., Su, X., Shen, B., and Zhou, W. (2005). Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. *Machine Learning*, *59*(3), 297–322.

https://doi.org/10.1007/s10994-005-0469-0

[5] Aha, D. W. (1991). Incremental Constructive Induction: An Instance-Based Approach. In *Machine Learning Proceedings 1991* (pp. 117–121). https://doi.org/10.1016/B978-1-55860-200-7.50027-1