FYS-STK4155 - Applied data analysis and machine learning
Project 3 - Classification of breast cancer data and wine quality data

Mäntysalo A. Visa, Nyberg Fredrik and Zariouh Osama

Desember 2019

The material for project 3 can all be found at: `https://github.com/OsamaZa/Project3`

Abstract

Five data mining algorithms are used for two widely different classification problems in this project: decision trees with and without bootstrap aggregation, random forests, neural networks and support vector machines. For each algorithm the degrees of freedom in the model building were explored to find the most successful model. The first classification problem was done on the Wisconsin breast cancer data where predictions on whether a tumor is benign or malignant were based on tumor characteristics. The most successful model for the breast cancer was a support vector machine with an accuracy score of 0.97661. The second classification problem was done on a much more unbalanced dataset with low correlation between features and targets relative to the breast cancer data. Predictions on wine quality were based on physicochemical characteristics, and here the most successful models were found to be random forest algorithm for both white and red wine dataset, with accuracy scores of 0.70090 and 070858 for white and red wine respectively. This project demonstrates the importance of correlation and class balance within a dataset when setting out to build predictive models of high quality.

Introduction

Historical data is of great importance when attempting to obtain knowledge for the future. In order to extract meaningful information from historical data, methods and tools of analysis are needed. The building of such methods are central in disciplines such as statistics and data science. Machine learning methods are extremely useful when classifying large amounts of data where a set of classes are paired with respective features. This bypasses the need of any prior knowledge about the underlying interplay between features because the algorithms are seemingly self-learning. These methods can be utilized in making forecasts that can be of great assistance in all arenas where large amounts of data is available.

A fundamental part in building good models that are based on self-learning is in the size and general quality of the dataset. It is important to have large amounts of data such that an algorithm gets influenced by as many occurances in the scenario as possible. Whether or not a dataset is balanced, as in near equal frequency of all targets, may also affect the model. It is also often the case that each feature present in a set of data is of varying importance. Some measurements might even be incorrect or simply be outliers. Such problems are often treated by optimization of the dataset by statistical analysis before initiating training. However, the

choice of method is not a one size fits all when evaluating different scenarios, and each method contains a large degree of freedom for different parameters for fine-tuning a model.

In this project, three different datasets will be used for both binary and multi classification problems. The datasets will differ in terms of class balance and correlation between features and targets. Analysis will be made with decision trees with and without bootstrap aggregation, random forests, neural networks and support vector machines by using algorithms from scikit-learn. Each method will be explored by varying the available input parameters such as regularization parameters, kernel, kernel coefficient, solver, activation function etc. The best method will be evaluated based on accuracy score and precision score. Results from previous research will be used for benchmarking. Finally, since both datasets are very different in terms of class balance and correlation, the effect of the quality of dataset on the models will also be discussed.

The Data

Breast Cancer Data

The first dataset is the "Breast Cancer Wisconsin (Original) Dataset" and consists of 699 samples and 9 descriptive features of tumors such as clump thickness, mitoses etc. Each sample has the class benign or malignant and the total dataset contains 65.5% and 34.5% of each class respectively. The feature correlation to the output for the data is high. The lowest, which is "Mitoses", has a correlation of around 0.45 and the highest "Bare Nuclei" with a correlation of around 0.83. The rest of the features are somewhere in between towards the high end. Which means that all the features have a high influence the on output result. This is illustrated in Figure 1.

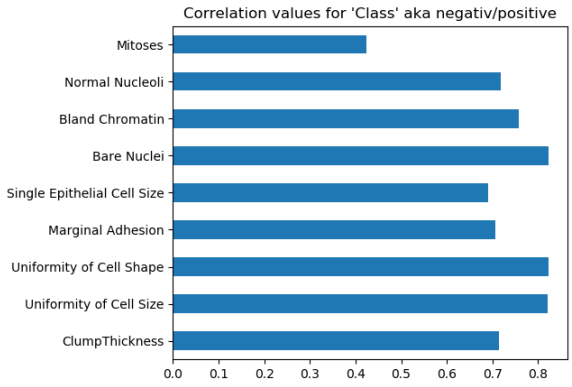

Figure 1: The correlation of the features to the targets in breast cancer dataset

Wine Quality Data

The second and third dataset is the "Wine Quality Dataset" which contains 4898 and 1599 samples of white and red wine respectively, and are described by 11 physicochemical features such as citric acid content, pH etc. The data output is based on median of at least 3 evaluations made by wine experts. Each expert graded the wine quality between 0 (very bad) and 10 (very excellent). However, the scores are distributed quite unevenly with a large majority of wines being classified with the scores 5, 6 and 7 as seen in Figure 2 and 3.

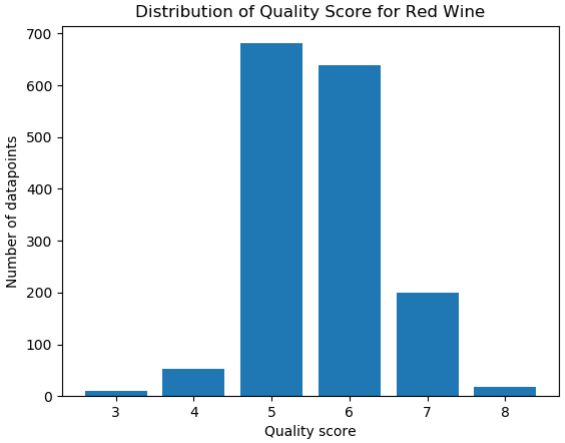

Figure 2: Distribution of quality score for the red wine dataset.

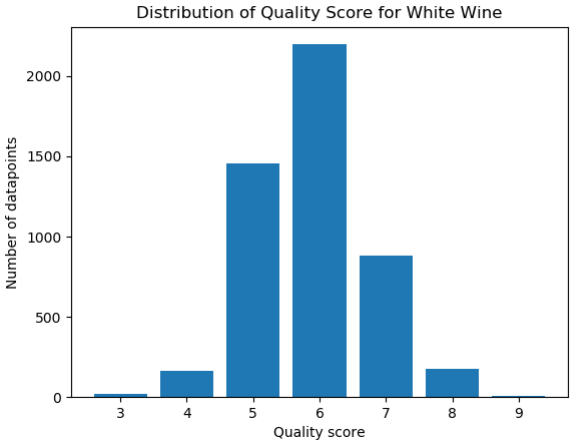

Figure 3: Distribution of quality score for the white wine dataset.

An important difference from the breast cancer data apart from amount of data is how the correlation to the target class is for the wine data. The correlation for the features vary way more and are lower in value. As seen in Figure 4 and 5.

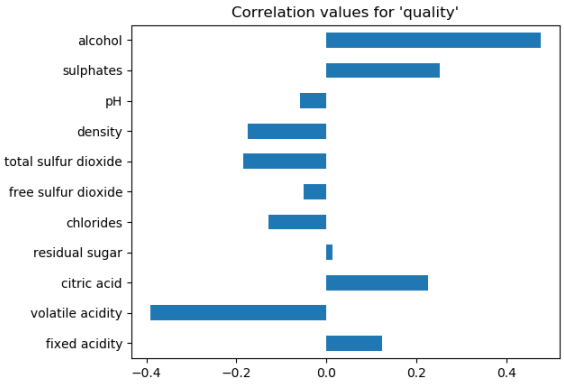

Figure 4: The correlation of the features to the target in red wine dataset

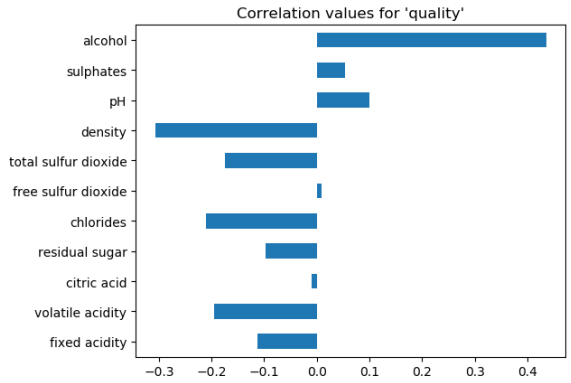

Figure 5: The correlation of the features to the target in white wine dataset

Theory

Accuracy score

In a classification problem, the performance of the predictions made by a given algorithm can be measured with a accuracy score.

$$Accuracy = \frac{\Sigma_{i=0}^{n} I(t_i = y_i)}{n} \tag{1}$$

where I is an indicator function. I yields 1 if the target t_i is equal to the prediction y_i and 0 otherwise. The sum is normalized by dividing with the total number of targets n, hence if all the predictions equal their corresponding target, $Accuracy = 1$, the algorithm will be a perfect classifier for the given set of data.

One hot encoder

scikit-learn's OneHotEncoder takes features which have discrete values as input, and makes a binary column for each discrete value. That means that if a feature has discrete values from 1 to 6, OneHotEncoder will split this column into 6 columns with values where the elements have values 0 or 1. And each column correspond to one of the discrete values. For each element in the columns, the column that corresponds to the original value in the original column, will be 1. The others will be 0.[15]

Confusion matrix

A confusion matrix is a table that sums up how the trained model performs on test data where the true values are known. This gives a better indication of how well the model predicts the outcome. The output of a binary confusion matrix are how many 1's and 0's that are correctly predicted, and how many 1's and 0's that are wrongly predicted. This is only shown for the best models of the three dataset in this study.[11]

K-Fold Cross-Validation

When it comes to making reliable predictions with a finite amount of available measurements it is common practice to separate the data into training and test data. When the data is split in this manner we artificially create a working environment where some of the data is used to train the model, and the remaining data is used to evaluate how well it predicts.[12]

The k-fold CV algorithm randomly rearranges the set of data and then splits the data into k-folds. Then the data from (k-1) of the folds are selected to train the model and the remaining fold is then used to benchmark the prediction performance of the model. This is repeated until all possible combinations of training and test data of grouped (k-1) and k-folds have been used for training. The initial randomisation ensures that each group represents data from the whole span of data. This way of building a model is done in order to obtain the most reliable model which consistently can make decent predictions for novel data, based only on the available data.[12]

Decision trees

Decision trees (DT) has the structure of a tree. It starts with a root node, that later splits into branches. The branches will then lead to new nodes that can be split further, until the leaf nodes are reached. Each node corresponds to a feature and the dataset is split into new nodes for the values of the feature. The most informative features are the features that will be split first. So the most informative feature is at the root node, then the next important feature and so on. The importance of a feature is decided by how good they can split the data into the correct target values. The splitting of features will happen until a certain stopping criteria is reached or the leaf nodes is reached. The leaf nodes decides the value of the target.[5]

Gini index

A way to split nodes is to use the gini index. It is defined as

$$g = \sum_{k=1}^{K} p_{mk}(1 - p_m k) \tag{2}$$

$$p_m k = \frac{1}{N_m} \sum_{x_i \epsilon R_m} I(y_i = k) \tag{3}$$

where k is a class, N_m is the number of observations in R_m, K is the number of classes, x_i is the i-th observation in region R_m and I is 1 when the i-th output y_i is equal to k and 0 else. In the CART algorithm, the gini factor can be used in the cost function to decide the split. The cost function that will be minimized is then

$$C = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right} \tag{4}$$

where m_{left} is the number points in the left split, m_{right} the number of points in the right split, m the total number of points and $G_{left/right}$ the gini index in the left and right split.[5]

Information entropy

Another way to split a node is to use the information entropy defined as [5]

$$s = -\sum_{k=1}^{K} p_{mk} * log(p_{mk}). \tag{5}$$

This is used by calculating the loss of entropy for each of the attributes that splits the node. To calculate the loss of entropy, the entropy loss for each of the splits gets subtracted from the entropy before the split.[7]

Bootstrap

The bootstrap method is to create multiple sub-samples which will randomly draw samples from the full sample. The samples drawn can be selected multiple times for each sub sample. Then the calculations wanted, will be done for each of the sub-samples. The final result will be the mean of the results from all of the sub-samples.[3]

Bootstrap aggregation

Bootstrap aggregation (BA) is a general ensemble method that combines several machine learning algorithms and uses the average of them as the final prediction. It creates several sub-samples as in the bootstrap method and makes a prediction for each sub-sample. Then the final prediction is the average of all the sub-sample predictions. This is often used for machine learning methods with high variance because it can reduce the variance. Decision trees is an algorithm with high variance. BA on decision trees would be to create a decision tree for each of the sub-sample and take the average prediction as the final prediction to reduce the variance.[3]

Random forest

Random forest (RF) also uses the BA algorithm. The difference is that for each split of a node, only a selected number of features are considered. The features considered are chosen randomly for each node. This is to make the decision trees more uncorrelated. Because if there is a feature that is dominating in informativeness, it will be chosen as the root node in almost every tree. It will not be the case for random forests because the most dominating feature will not always be selected, and therefore the trees will be more unlike.[5]

Support Vector Machines

Support vector machines (SVM) has obtained success and good reputation in the machine learning community due to high performance in tackling challenging problems. The idea behind SVM is quite simple, but the complexity comes from the mathematical implementation of the idea. The idea is trying to separate groups of data points on a two dimensional plane. Let's say that the groups are easily separable and that you have achieved a good result. Now the model might be very good at solving the problem, based on that the dataset is naturally easy to separate. So when new unseen data is tested the model might classify some points wrong. Therefore SVM tries to find the best separating line between the groups. Figure 6 shows how the data groups can be separated. Infinite amount of lines can separate the easily separable groups, but which line is the most optimal? If the separating line is poor in quality (not optimized) new data can be wrongly classified.

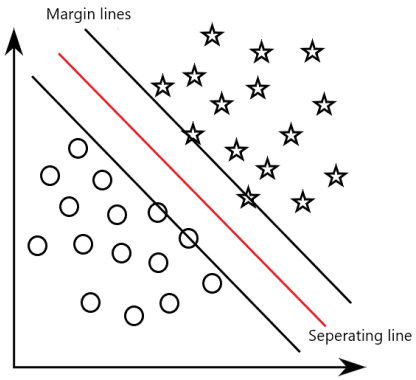

Figure 6: How data groups are separated by SVM.

SVM minimizes the possibility of choosing the wrong line, by finding the line that has the largest distance to the bordering points of the two groups. Having as much space between the groups as possible will lower the chances of wrongly classifying data points. This distance is called the margin. The margin is determined by the points that are on the limit of the margin. The points that are on the limit of the margin are the support vectors, which gives the machine learning method it's name. This can be seen in Figure 7. The separating line is called for the hyperplane. SVM's task is to find a hyperplane in a p-dimensional space, where p is the number of features that distinctly classifies the data points.

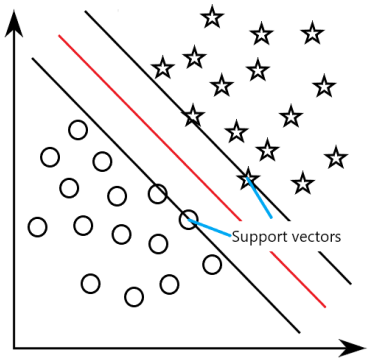

Figure 7: The points that are on the limit of the margin are the support vectors.

To achieve maximum space separation the following equation has to be minimized

$$\mathcal{L} = \sum_i \lambda_i - \frac{1}{2} \sum_{ij}^{n} \lambda_i \lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \tag{6}$$

where $\boldsymbol{x}$ are the support vectors, λ is the Lagrange multiplier and y is the class indicator. Separation of the groups isn't limited by a straight line. The line can be in any shape or form so that separation is as optimized as it can be. In reality the data points are harder to separate due to group overlap. The idea is to map the data points in higher dimension space so that a multi dimensional hyperplane can be used for separation of data groups. SVM can therefore use a quite complex multi dimensional space for separation of data points, this technique is called the kernel trick. A kernel is a function that is defined as the mapping function for the data points multiplied with each other. If the mapping function is defined as ϕ then the kernel function is

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j) \tag{7}$$

This changes equation 5 to

$$\begin{aligned} \mathcal{L} = \sum_i \lambda_i - \frac{1}{2} \sum_{ij}^{n} \lambda_i \lambda_j y_i y_j \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j) = \\ \sum_i \lambda_i - \frac{1}{2} \sum_{ij}^{n} \lambda_i \lambda_j y_i y_j * K(\boldsymbol{x}_i, \boldsymbol{x}_j) \end{aligned} \tag{8}$$

The major point here is that the kernel function is used instead of the complex mapping function, but there are some important factors that determines if a function can be used as a kernel function. These conditions are expressed by Mercer's theorem. The theorem states that if a kernel function K is symmetric, continuous and gives a positive "semi-definite" matrix P then there exists a function ϕ that maps x_i and x_j. If a kernel fulfils the conditions then ϕ exists even nothing is known about ϕ.[5]

Neural Network

Artificial neural networks (NN) attempt to mimic the human brain. A human brain is composed of billions of neurons that communicate with each other via electric signals. It is believed that repeated patterns of signals between neurons result in stronger connections between the said neurons, and hence promotes learning or memory associated to what induced the signals in the first place. Artificial neural networks are therefore able to learn without being programmed explicitly to solve specific problems. The first and the simplest artificial network devised was the feed-forward neural network.[14]

Feed-forward neural networks have a layered design, input layer and output layer separated with one or more hidden layers, and information moves only in the forward direction through the layers. The hidden layers consist of nodes which act as artificial neurons, and are connected to the nodes in the subsequent layers. These artificial neurons activate and output accordingly to what is fed into them, the most basic model for these is called the perceptron [14]

$$Output = \begin{cases} 1, & \text{if } w \cdot x + b > 0 \\ 0, & \text{if } w \cdot x + b \leq 0 \end{cases}$$

Here w can describe the importance of the connections to the preceding layer and is often referred to as the weight, and b is the bias and is a measure on how easy it is for the perceptron to activate. As mentioned above, a neural network often consists of several layers with several neurons, which means that there are a set of weights and biases between each layer in the network. Initially all the weights and biases are set to some random numbers.[14]

The training of a neural network is a form of supervised learning and can be done with a backpropagation algorithm. When feeding in training data, the networks performance is evaluated by computing the error between the outputs from the output layer and the actual output. This error is the cost function $C(w, b, x)$ of the network, which is to be minimized with respect to the preceding weights and biases. This minimization problem makes suggestions, with gradient descent methods, to what the weights and biases of the preceding layers should be changed to in order to minimize the output error. Hence, updating the weights and biases repeatedly with different data will produce better and better output, and the network will appear to be learning.[14]

With the perceptron model a small change to certain weights or biases can totally flip whether or not a neuron is activated, and therefore change the ripple through the network in some complicated way, and hence also have

a large impact on the final output. It is therefore desirable to have the activation of neurons be in the span of 0 and 1 rather than strictly being 0 or 1. This way small changes in the weights and biases only cause small changes to the output.[14]

This is done with so-called activation functions which squish the input of a weighted sum to some value between 0 and 1. The Sigmoid function is one such function. Very positive values of the weighted sum are transformed to be close to 1 and very negative weighted sums are transformed to be close to 0. The threshold for what should be considered a positive and negative weighted sum is controlled by the bias.[14]

This type of transformations between the neuron inputs and outputs makes the interplay between layers more nuanced and subtle changes to the parameters w and b cause subtle changes to the output layer. When the network is exposed to sufficient amounts of training data for a given task, it will have optimized the weights and biases between the layers. The network will therefore appear to be learning, or rather memorising, how to treat given input data.[11]

Method

Datasets

A specific random seed(10) was used throughout the analysis. There are three datasets that are used in this study. Two wine quality dataset, one for red wine and one white wine, and a dataset for breast cancer. The sizes of the wine quality data and breast cancer data is different where the wine quality data is larger. And since the amount of data has a connection to computational time, the wine quality data was randomly split with scikit-learn's traintestsplit package. The red and white wine data was respectively reduced to 719 data points and 734 data points. Note that the data reduction was only done for SVM and NN since the calculation was very time consuming. This is because of the grid search of the model parameters. The data was read in and the wine data was split and reduced in size. The breast cancer data was then one-hot-encoded and the wine quality data was scaled. A heatmap of the correlation matrix and a bar plot of the correlation to the target result was made for all the datasets.

Decision Tree

For decision trees the DecisionTreeClassifier from scikit-learn's tree package was used. To measure the accuracy score and confusion matrix a 5-fold cross-validation was used. The mean accuracy and confusion matrix was used as the final result. This was done for both entropy and the gini index as the splitting criterion. For each fold the tree was expanded until all leaves were pure.

Bootstrap Aggregation

For BA the BaggingClassifier from scikit-learn's ensemble package was used. It was specified that the DecisionTreeClassifier was the base estimator to be combined, and was combined in two ways. One with the gini index as the splitting criterion and one with the entropy as the splitting criterion. The number of decision trees to be combined was varied as well as the maximal samples drawn to train each decision tree. For each combination of these three parameters the a mean accuracy score and confusion matrix were computed from a 5-fold cross-validation. The accuracy scores were stored in a matrix and plotted to show different accuracy scores for different combinations of the number of trees, and maximal samples used to train each tree for both entropy and the gini index as splitting criterion. For each fold the trees were expanded until all leaves were pure during training.

Random Forest

For random forest the RandomForestClassifier from scikit-learn's ensemble package was used. It was run with the criterion for splitting as the gini index and as entropy. For each of these the number of decision trees and features considered for each split were varied. For each combination of these three parameters a mean accuracy and confusion matrix were computed from a 5-fold cross-validation. The accuracy scores were stored in a matrix and plotted to show it as a function of number of trees and number of features considered for each split for both entropy and the gini index as splitting criterion. For each fold the trees were expanded until all leaves were pure during training.

Support Vector Machines

The scikit-learn's SVM package was used for the analyzation of the breast cancer data and the wine quality data. 5-fold cross-validation was implemented for all models. In total four model parameters was investigated; the regularization parameter C, the kernel function, the degrees for polynomial kernel function "poly" and the kernel coefficient Gamma. The regularization parameter determines the miss classifications of the model trained. A high C will make the optimization to choose a smaller margin and therefore have a lower chance of misclassification. But a higher C might not give a better result since C is not only correlated to the margin. This means that there is a optimal C for different models. The choice of the right kernel is critical. The most important difference between the kernels in scikit-learn is that the RBF is more complex and the complexity increases with the amount of data. Gamma influences the importance of a single training data. A high Gamma will only assess data points close to the hyperplane and low Gamma will use data points further from the hyperplane. So having a low Gamma will result in the model finding a more correct hyperplane. Then all the models trained, made with all the combinations of model parameters, are saved along with their respectively results; confusion matrix and accuracy. Then results from the model or models with the best accuracy are extracted.

Neural Network

Also the neural network used is this study was made with scikit-learn's MLPClassifier package for all datasets. Five model parameters were varied; the amount of nodes in the hidden layer (all models had only one hidden layer), the activation function, the solver for weight optimization, the L2 penalty parameter and the learning rate. 5-fold cross-validation was implemented on all models. The models were then trained and saved. The models with the best accuracy score was extracted along with their corresponding results.

Results

The accuracy scores for the classification analysis for all methods can be found in tables 1 - 2, and confusion matrices for the best models for each dataset can be found as matrices labelled 9, 10 and 11. The results can also be found on `https://github.com/OsamaZa/Project3`.

Table 1: Accuracy score of the different models for the breast cancer dataset. The best result is highlighted in bold text.

Accuracy for the Breast Cancer dataset

	DT	BA	RF	NN	SVM
Accuracy Score	0.93279	0.96631	0.97657	0.97659	**0.97661**

Table 2: Accuracy score of the different models for the wine datasets. The best results are highlighted in bold text.

Accuracy for the Wine datasets

	DT	BA	RF	NN	SVM
White Wine Accuracy Score	0.60351	0.69518	**0.70090**	0.58318	0.57088
Red Wine Accuracy Score	0.6097 5	0.70797	**0.70858**	0.62730	0.61332

Breast Cancer

Highest accuracy, for the breast cancer classification, was the SVM model, barely outperforming RF, NN and BA as shown in Table 1. DT on the other hand was considerably outperformed by SVM which had a difference in

accuracy of approximately 0.04. The precision for malignant and benign was 0.9738 and 0.9775 respectively, which indicates that the model was best at predicting if a tumor was malignant, which is illustrated in the confusion matrix 9.

The parameter grid search resulted in the sigmoid-kernel with the kernel coefficient set to auto and a cost set to 2.6827 for the best SVM model. NN had 100 hidden neurons in the hidden layer, logistic activation function and used the adam solver with a constant learning rate and a penalty of $5.6234 * 10^{-7}$. The best criterion for both DT and RF was found to be gini whereas entropy was best for BA. In RF 100 trees with 4 features were considered for each split and for BA 100 trees were trained with 100 samples.

The average confusion matrix for breast cancer dataset made from the SVM model with the regularization parameter $C = 2.6827$, rbf kernel function and the kernel coefficient $Gamma = auto$. The first and the second row in the confusion matrix respectively represent benign and malignant. The diagonal is the number of correct predictions for the given class.

$$\begin{bmatrix} 86.2 & 2.6 \\ 0.6 & 47.2 \end{bmatrix} \tag{9}$$

White Wine

For white wine classification the highest accuracy was found by the RF model, and considerably outperformed DT, SVM and NN as shown in Table 2. BA came the closest to RF in terms of accuracy with a difference in accuracy of approximately 0.005. The less successful models had approximately 0.10 - 0.13 lower accuracy than RF and BA. All models did a poor job at identifying the highest and lowest quality (quality = 3, 4, 8 and 9) wines as shown in the confusion matrix 10.

The best RF model had 500 trees with 3 features being considered for each split. Both DT and BA used the gini criterion, and 1000 trees were trained with 2500 samples for BA. The SVM model used the rbf-kernel with the kernel coefficient set to auto and a cost set to 2.6827. NN had 25 hidden neurons in the hidden layer, relu activation function and used the adam solver, with inverse scaled learning rate and a penalty of 0.1.

Matrix 10 shows the confusion matrix for the best RF model. The first row represent the quality output for 3 and the remaining is in increasing order to 9. The diagonal is the number of correct predictions for the given class.

$$\begin{bmatrix} 0 & 0 & 1.2 & 2.8 & 0 & 0 & 0 \\ 0 & 8.4 & 14.2 & 9.8 & 0.2 & 0 & 0 \\ 0 & 2.6 & 204.8 & 81.6 & 2.4 & 0 & 0 \\ 0 & 0.6 & 52.6 & 361 & 25 & 0.4 & 0 \\ 0 & 0 & 2.4 & 75.2 & 97.6 & 0.8 & 0 \\ 0 & 0 & 0.2 & 11.8 & 8.2 & 14.8 & 0 \\ 0 & 0 & 0 & 0.2 & 0.8 & 0 & 0 \end{bmatrix} \tag{10}$$

Red Wine

The RF model was also found to be the best for red wine classification, with a near negligible outperformance of BA as shown in Table 2. The less successful models had approximately 0.08 - 0.10 lower accuracy than RF and BA. Interestingly the SVM and NN models had improved results for red wine compared to white wine, while RF, BA and DT had similar success for both datasets. Also here, all models did a poor job at identifying the highest and lowest quality wines. Matrix 11 shows the confusion matrix for the best RF model. The first row represent the quality output for 3 and the remaining is in increasing order to 8. The diagonal is the number of correct predictions for the given class.

$$\begin{bmatrix} 0 & 0.2 & 1.4 & 0.4 & 0 & 0 \\ 0 & 0 & 7.4 & 3.0 & 0.2 & 0 \\ 0 & 0.2 & 111.6 & 23.4 & 1.0 & 0 \\ 0 & 0.2 & 27.8 & 92.2 & 7.2 & 0.2 \\ 0 & 0 & 1.8 & 15.2 & 22.8 & 0 \\ 0 & 0 & 0 & 2.2 & 1.4 & 0 \end{bmatrix} \tag{11}$$

RF and BA used entropy criterion whereas DT used gini. RF used 500 trees with 7 features for each split and in BA 1000 trees were trained with 1200 samples. The best SVM model used the rbf-kernel with the kernel coefficient set to auto and a cost set to 37.276. NN had 25 hidden neurons in the hidden layer, relu activation function and used the adam solver, with adaptive learning rate and a penalty of 0.1.

Discussion

All benchmark accuracies can be found in Table 3 . Note that benchmarks for BA are not included in this study for model comparisons.

Table 3: Accuracy scores from previous studies by; Bharat, Pooja, and Reddy [2], Karabatak and Ince [8], Lee, Park, and Kang [9] and Wang et al. [16]. No benchmark accuracies was found for BA. For NN, only benchmark accuracy for the breast cancer data was found.

Accuracy Scores From Previous Studies

	Decision Tree	Random Forest	Neural Network	Support Vector Machines
Breast Cancer	0.9513 [2]	0.9701 [16]	0.952 [8]	0.9713 [2]
Red Wine	0.607 [9]	0.6839 [10]	—	0.6372 [10]
White Wine	0.587 [9]	0.6874 [10]	—	0.6460 [10]

Breast Cancer Data

DT was the poorest method for this dataset. It is not surprising, as it is a simple algorithm which often suffers from overfitting. That is why it can be improved by BA and RF.

Our SVM model outperformed the SVM benchmark by Asri et.al. in terms of accuracy. Our results for precision scores show that our SVM model was more successful at predicting tumors that were malignant rather than benign. The benchmark precisions shows the opposite trend with a precision of 0.98 and 0.95 for benign and malignant respectively. These different results may be explained by their use of 10-fold cross-validation whereas we only used 5-fold. This can have affected the results from the grid search since they used the rbf kernel whereas we used the sigmoid kernel for the best model.

Our NN was one of the models that outperformed its respective benchmark model by Karabatak & Ince with an increase in accuracy of 0.02459. It should be mentioned that our NN did not fully converge, which introduces uncertainty to our numerical results and ultimately affects their reliability. On the other hand, the results can also be explained through the differences in NN architecture resulting from the parameter grid search. The only equal parameter between our model and the benchmark model which was the use of one hidden layer. Where our models differ is in the number of neurons in the hidden layer, the solver and the activation function. We used 100 hidden neurons, the adam solver and a logistic activation function, whereas, they used 25 hidden neurons, a Levenberg-Marquardt solver and the identity (linear) activation function. It should also be mentioned that they used 3-fold cross-validation.

Only our decision tree model was unable to outperform its benchmark accuracy of 0.9513. Our decision tree was the scikit-learn tree's DecisionTreeClassi-

fier which uses an optimized version of the CART algorithm. The benchmark decision tree uses the C4.5 algorithm which is quite similar to the CART algorithm but converts trained trees from the ID3 algorithm into sets of if-then rules. This may explain the difference in our results.[1]

The random forest used in this project got a better accuracy compared to Sutong Wang et.al.[16] The benchmark result obtained is an average of 25 runs where each run does a stratified 10-fold cross-validation. It does not say anything about what kind of parameters it uses or algorithm for the decision trees. Another reason the benchmark gets a lower accuracy is probably because the data has been treated differently, than in the method used in this project. The benchmark has treated the features as continuous values and the missing values have been replaced with the median of the corresponding features, where as we one-hot-encoded and removed the incomplete data.

Wine Data

BA and RF did a lot better than SVM, NN and DT. It is not surprising that it did better than DT, as DT is often overfitted whereas BA and RF reduces this overfitting. The best models for classifying the wine data was the RF algorithm. Looking at the confusion matrix one clearly sees that the classes 3, 4, 8 and 9 are rarely guessed, this is because of the low occurrence of these classes.

RF and BA was very close in accuracy. The difference between them is that RF only consider a number of random features for each split, while BA consider all. RF's way of splitting is supposed to make more uncorrelated trees and is supposed to be more effective if there are some features that are very dominating. In this case, the features correlation with the targets are rather small, and there is no feature that is clearly dominating as seen in Figure 4 and 5. Alcohol has the biggest correlation, but it is not that much larger than some of the closest features in correlation such as density, chlorides and volatile acidity. Another factor, is that even though the correlation of alcohol is more than twice the lowest correlated features, it is quite low at only 0.4. So the target does not depend so much on this feature in the first place. This explains the overall lower accuracy for wine quality data compared to the breast cancer data.

Our RF models outperformed the benchmark models. The benchmark RF uses 5-fold cross-validation in 5-runs, which differ from this 1-run 5-fold cross-validation. For white wine it uses 1000 decision trees and considers only one feature for each split, as that gave the best result in that experiment. This study does not try one feature per split, but still gets an higher accuracy. For red wine the benchmark also uses 1000 decision trees and considers 2 features per split, as that gave the best accuracy. It does not say what kind of decision tree it uses, or the size of each bootstrap.

The DT in this study gives better accuracy than the benchmark for both red and white wine. The benchmark uses the C4.5 algorithm and the entropy to split nodes, while our approach uses a CART algorithm and gets that the gini index gives better accuracy. It should also be mentioned that our entropy model outperform the benchmark. Both algorithms uses 5-fold cross-validation, so the CART algorithm performs better for the wine data.

SVM was originally created for binary classification, and that is presumably the reason for the underperformance in classifying the wine data. The features in the wine data are in general not dominating in determining the output result, which will make it more difficult for the SVM to actually separate the data points in higher dimensional space. Comparing with the benchmark result from [10], the benchmark achieved a better result for both the red and white wine dataset. The major differences between our model and the benchmark model is the gamma value and that the benchmark value didn't limit the model to a specific random seed but made an average of 25 experiments. A plausible reason for our models underperformance is that it was limited to a single random seed which might have split the data so that the worst data was used in the prediction. What is meant with worst data is that it would be difficult to split and categorize the data due to much underlying similarity. A future improvement could be to let our model compute with different random seeds and find the average of the experiments like Malloy [10].

The possible reasons for NN to have so low accuracy in classifying the wine data are that the data used was reduced to save computational time, and that NN's are very dependent on how much data is used for training and how the data is distributed. The distribution of the wine data was not spread out. So the NN had less training on classifying the outermost classes. As well as the NN didn't converge completely after 200 intera-

tions. The cost function would should have also been investigated to ensure good convergence. Adding more layers to the NN could also have improved the model. All these factors could be improved in future work and maybe NN would perform better.

The aim for this part of the study was mainly to point out the importance of correlation and balance within the dataset, hence the wine data was treated in the same way as the breast cancer data. As implied by the correlation matrices 9,10 and 11, the breast cancer dataset contains both highly and more correlated features than the wine data. This resulted in a lower accuracy overall for the wine data models than for breast cancer models. This also makes sense because the malignant nature of a tumor is more tangible than the sensory perception of wine quality. One may therefore assume that making measurements for more impacting features surrounding something as subjective as wine quality is more challenging. Which ultimately affects the reliability of our wine models.

Previous studies on the wine data have used different methodology with each machine learning method, which made a direct comparison with existing results a challenge. If a deeper analysis for the wine data were to be conducted in the future, many approaches could be considered. Alternative statistical metrics beyond accuracy score could have been more appropriate when working with imbalanced classes to evaluate the models. A study from 2016 conducted by Atasoy and Er classified the wine data with an array of machine learning methods with metrics such as the F1-score (the harmonic mean of precision and recall).[4] Another approach to bypass the imbalanced nature of the wine data was done in a study from 2016 by Hu et.al. by transforming the imbalanced data into balanced data with a SMOTE algorithm (Synthetic Minority Over-Sampling Technique) which increased the accuracy for their random forest model by 0.7%. However, this analysis treated the wine data as a tertiary classification problem by grouping the 7 targets into low (scores 3 and 4), normal (scores 5, 6 and 7) and high (scores 8 and 9) quality.[13] Another interesting way of analysing the wine data would have been to utilize more regression-like approaches by evaluating the models by quantifying how close the model is in predicting the correct quality scores. This was done in in 2009 by Cortez et.al. by comparing the mean absolute deviance for each model when evaluating each machine learning method.[6]

Conclusion

Our models performed better for the breast cancer data than the wine data overall. This was to be expected due to the correlation of the features to the outputs in the datasets, where the breast cancer data had better correlation. Many of our models also performed better than the benchmarks for many possible reasons such as difference in methodology. SVM turned out to be the best model for the binary breast cancer analysis, while RF was the best for the multi class classification of the wine data. Possible improvements to the wine data analysis could have been done by treating the dataset differently, as discussed. For the analysis of all datasets, multiple experiments could have been run without a fixed random seed, rather than only relying on 5-fold cross-validation. It would also have been interesting to include the xtreme gradient boost algorithm with decision trees as the base estimator. The reason it would be interesting to implement this is that it wins all machine learning contests nowadays, and maybe better results would have been achieved.

Appendix

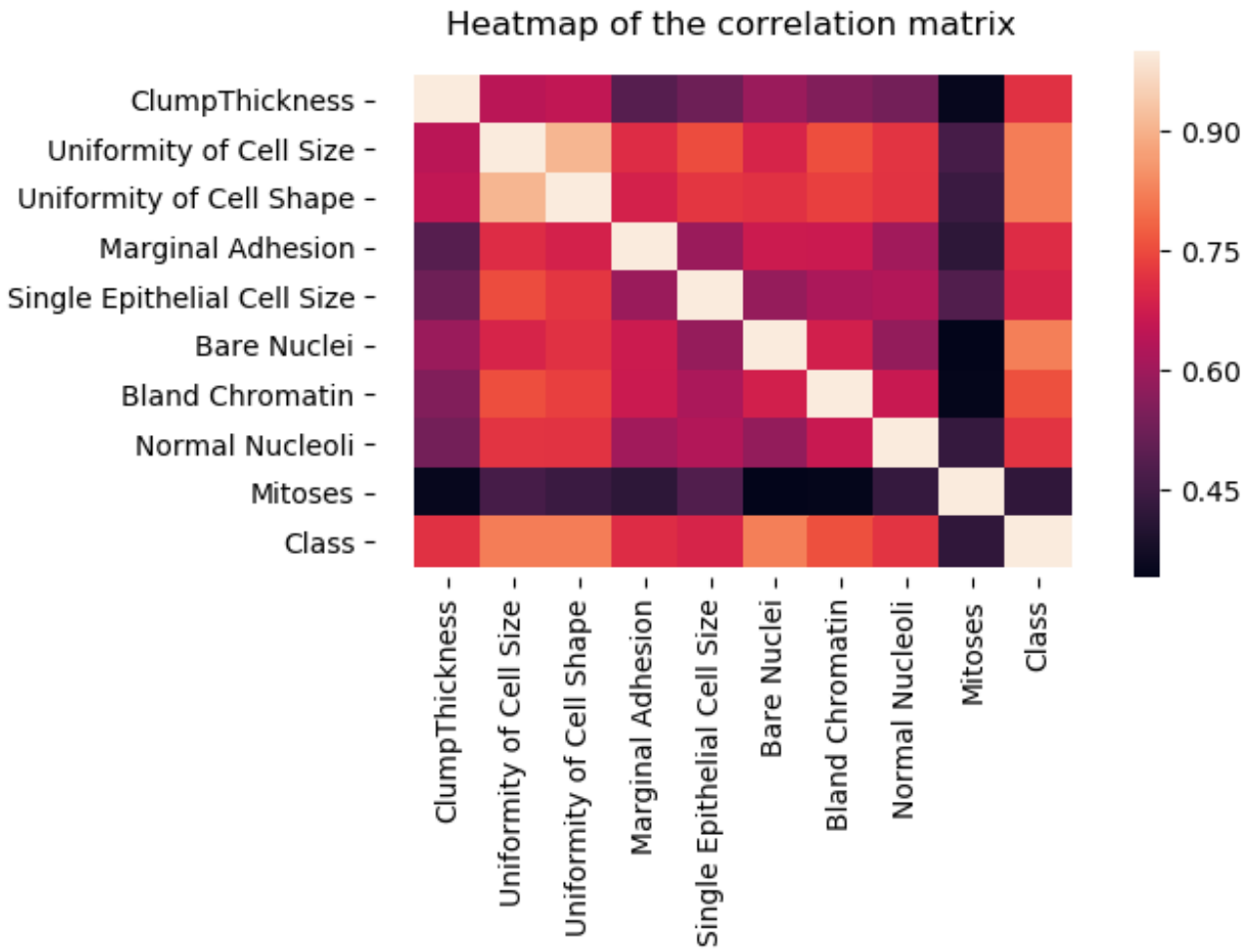

Figure 8: Heat map of the correlation matrix for breast cancer dataset.

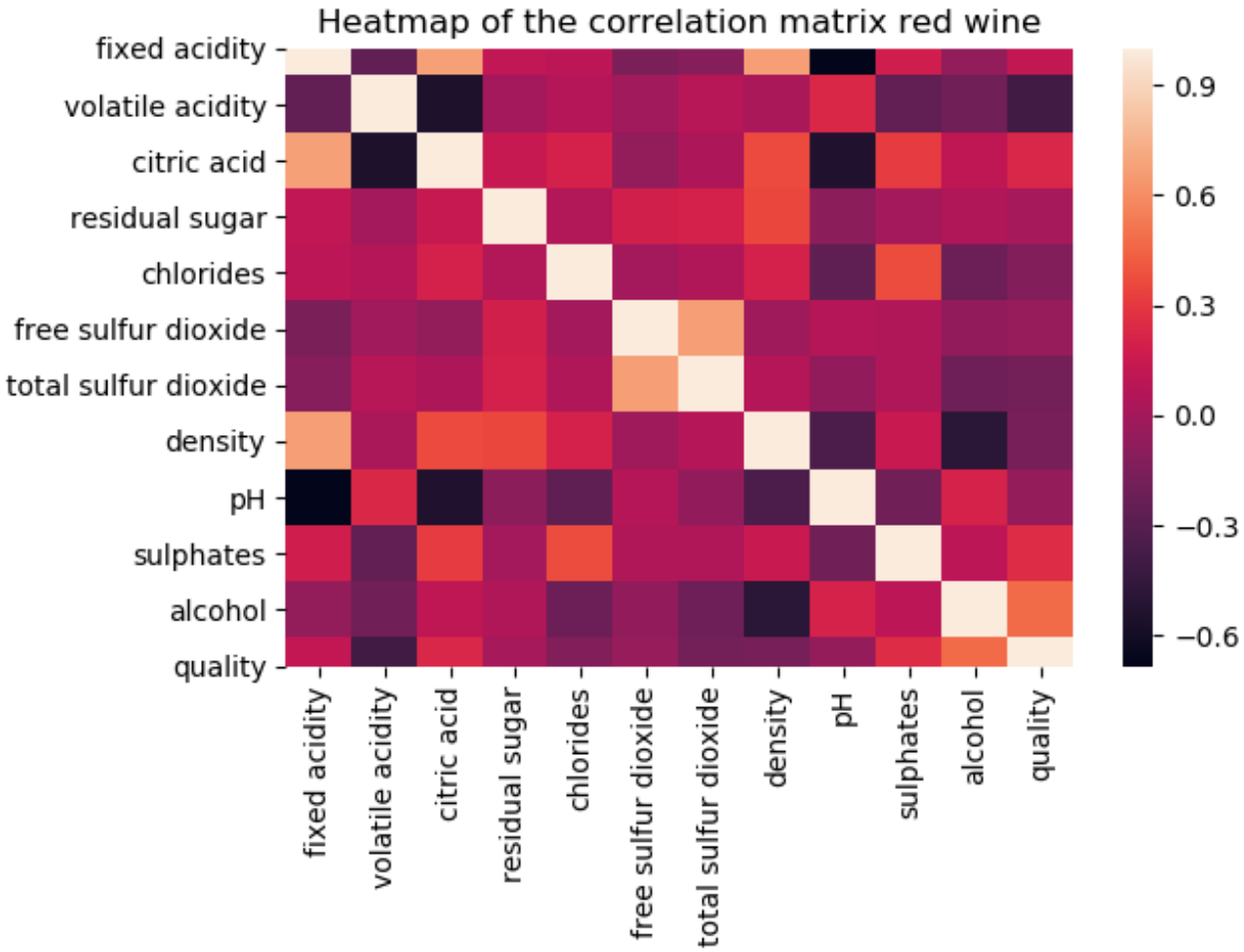

Figure 9: Heat map of the correlation matrix for red wine dataset.

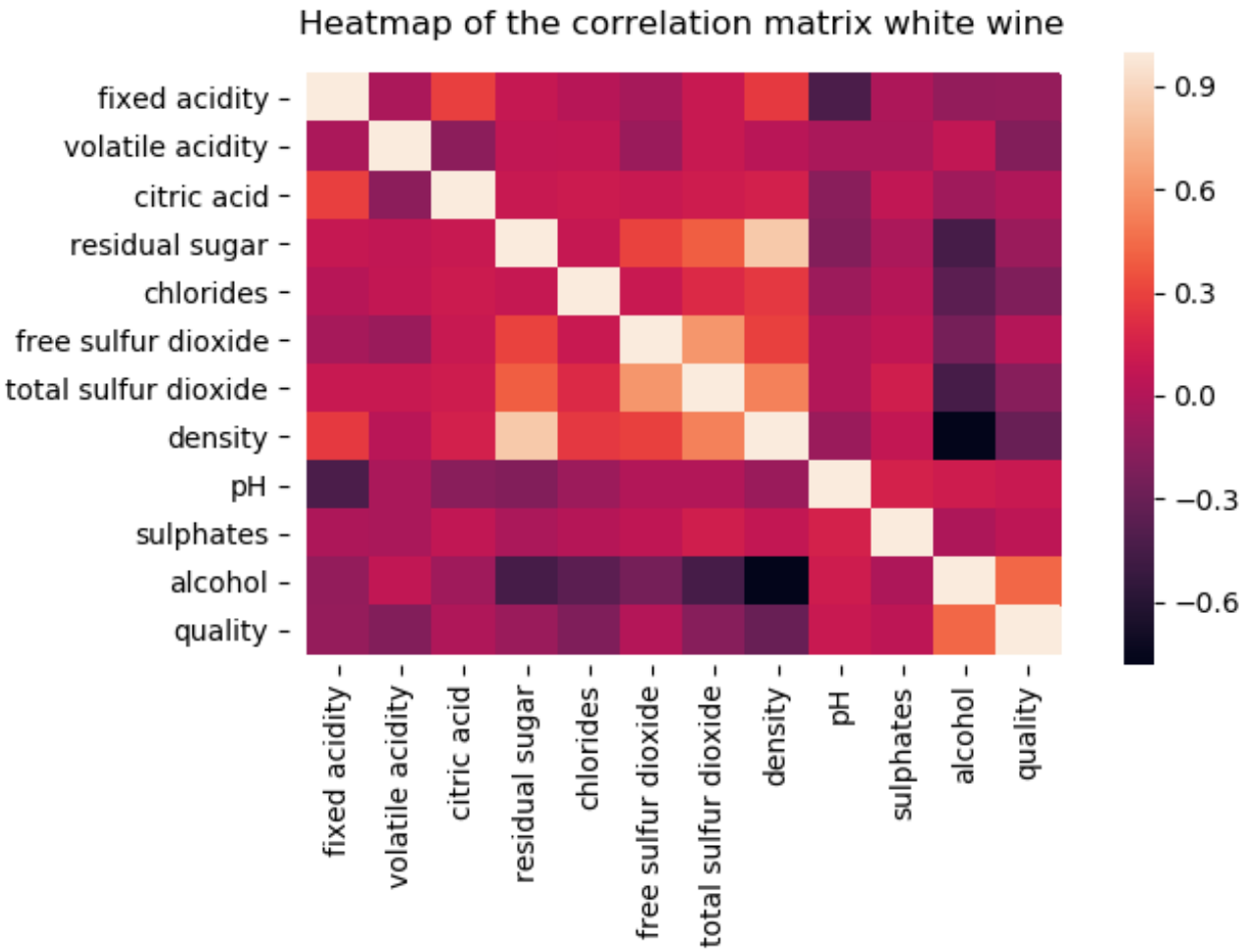

Figure 10: Heat map of the correlation matrix for white wine dataset.

Table 4: Precision scores for breast cancer data from the machine learning models

Precision Scores

Model	Precision for benign	Precision for Malignant
SVM	0.9707	0.9874
NN	0.97747	0.97489

Table 5: The SVM models that gave the highest accuracy with their respective model parameters

Support Vector Machine				
Dataset	C	Kernel	Gamma	Accuracy
Breast Cancer	2.6827	rbf	auto	0.97661
White Wine	2.6827	rbf	auto	0.5310
Red Wine	0.1931	sigmoid	scale	0.5528

Table 6: The NN models that gave the highest accuracy with their respective model parameters

Neural Network						
Dataset	Nodes	Activation Function	Solver	L2	Learningrate	Accuracy
Breast Cancer	75	logistic	adam	3.1623e-05	adaptive	0.97659
White Wine	100	relu	adam	1e-08	constant	0.5473
Red Wine	25	identity	lbfgs	0.1	adaptive	0.5705

References

[1] *1.10. Decision Trees*. URL: https://scikit-learn.org/stable/modules/tree.html#tree.

[2] A. Bharat, N. Pooja, and R. A. Reddy. "Using Machine Learning algorithms for breast cancer risk prediction and diagnosis". In: *2018 3rd International Conference on Circuits, Control, Communication and Computing (I4C)*. Oct. 2018, pp. 1–4. DOI: 10.1109/CIMCA.2018.8739696.

[3] Jason Brownlee. *Bagging and Random Forest Ensemble Algorithms for Machine Learning*. 2016. URL: https://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/.

[4] Yeşim Er and Ayten Atasoy. "The Classification of White Wine and Red Wine According to Their Physicochemical Qualities". In: *International Journal of Intelligent Systems and Applications in Engineering* (Dec. 2016), pp. 23–26. DOI: 10.18201/ijisae.265954. URL: https://www.ijisae.org/IJISAE/article/view/914.

[5] Morten Hjorth-Jensen. *Lectures Notes in FYS-STK4155. Data Analysis and Machine Learning: Linear Regression and more Advanced Regression Analysis*. Sept. 2019. URL: https://compphysics.github.io/MachineLearning/doc/pub/Regression/html/Regression.html.

[6] Cortez P.; Teixeira J.; Cerdeira A.; Almeida F.; Matos T.; Reis J. *Using Data Mining for Wine Quality Assessment*. 2009.

[7] Rishabh Jain. *Decision Tree. It begins here*. 2017. URL: https://medium.com/@rishabhjain_22692/decision-trees-it-begins-here-93ff54ef134.

[8] Murat Karabatak and M. Cevdet Ince. "An Expert System for Detection of Breast Cancer Based on Association Rules and Neural Network". In: *Expert Syst. Appl.* 36.2 (Mar. 2009), pp. 3465–3469. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2008.02.064. URL: http://dx.doi.org/10.1016/j.eswa.2008.02.064.

[9] S. Lee, J. Park, and K. Kang. "Assessing wine quality using a decision tree". In: *2015 IEEE International Symposium on Systems Engineering (ISSE)*. Sept. 2015, pp. 176–178. DOI: 10.1109/SysEng.2015.7302752.

[10] Garry Malloy. *Machine Learning with the UCI Wine Quality Dataset*. URL: https://rstudio-pubs-static.s3.amazonaws.com/175762_83cf2d7b322c4c63bf9ba2487b79e77e.html?fbclid=IwAR0bhxIUxn4KrdCyHaa5K20TF6Zt_w0Vdb4.

[11] Nyberg Fredrik Mäntysalo A. Visa and Zariouh Osama. *Project 2 - Classification and Regression, from linear and logisticregression to neural networks*. Nov. 2019. URL: https://github.com/OsamaZa/Project2.

[12] Zariouh Osama Mäntysalo A. Visa Nyberg Fredrik. *Project - Regression analysis and resampling methods*. Oct. 2019. URL: https://github.com/OsamaZa/Project1.

[13] G. Hu; T. Xi; F. Mohammed; H. Miao. "Classification of wine quality with imbalanced data". In: *2016 IEEE International Conference on Industrial Technology (ICIT), pp. 1712-1217* (2016).

[14] Michael A. Nielsen. *Neural Networks and Deep Learning*. 2015. URL: http://neuralnetworksanddeeplearning.com/chap1.html.

[15] *OneHotEncoder*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html#sklearn-preprocessing-onehotencoder.

[16] Sutong Wang et al. "An improved random forest-based rule extraction method for breast cancer diagnosis". In: (). DOI: 10.1016/j.asoc.2019.105941.