APPLICATIONS OF BINARY CLASSIFICATION IN MACHINE LEARNING

SHAFAQ NAZ SHEIKH
HTTPS://GITHUB.UIO.NO/SHAFAQNS/FYS-STK4155/TREE/MASTER/PROJECT3

Abstract. In this article we will be looking at the binary classification task in machine learning. The learning methods we will be using are logistic regression, decision trees and ensemble methods. The ensemble methods that we will be using are voting classifiers, bagging and random forests. Instead of developing our own code, we will be using scikit's built in functions in python. We will be using two data sets for analyzing these methods. The first one is the Wisconsin breast cancer data set, where we want to classify a tumour as malignant or benign. The second one is the South African heart disease data set, where we want to predict if a person has coronary heart disease or not. We find that the best method for the breast cancer data is the random forest, with 253 decision trees, which gives an accuracy of 0.98 on the test set, while the best method for the heart disease data is the decision tree with the *max_depth* parameter equal to 9, this gives an accuracy of 0.77 on the test set.

Date: 11th November 2020.

1. Introduction

Binary classification is the task of classifying the elements of a data set into two categories, based on whether they have some property or not. A typical binary classification task could be diagnosis of medical testing to determine if a patient has a certain disease or not. In this article I will be looking at two scenarios of binary classification. The first data set is the breast cancer Wisconsin (diagnostic) data set[1], where we want to predict if a breast cancer tumour is malignant or benign. A malignant tumour is cancerous, while a benign tumour is non-cancerous. This data set contains 569 samples, and 30 features that describe the tumours. The second data set is the South African heart disease data set[2]. This data is a subset of the coronary risk-factor study(CORIS) baseline survey, carried out in three rural areas of the Western Cape in South Africa. The data represents white males between the ages of 15 and 64. We want to predict if a person has coronary heart disease, or not. This data set contains 462 samples, and 9 features that describe the risk factors for coronary heart disease.

The machine learning algorithms that I will be using are logistic regression, decision trees, and ensemble learning methods. The ensemble learning methods I will be looking at are voting classifiers, bagging and random forests. To implement these methods, I will be using scikit's built in functions in python. We start by presenting the theoretical background behind these algorithms, after which we talk about the performance measures we will use. We briefly mention ways to explore the features in the data set. Then we present our results for the breast cancer data set, followed by the heart disease data set. Finally, we end with a discussion of our findings of the various models, before giving a conclusion.

2. Theoretical background of the methods

2.1. **Logistic Regression.** The logistic regression model arises from the desire to model the posterior probabilities of the K classes via linear functions in t, while at the same time ensuring that they sum to one and remain in $[0, 1]$. When $K = 2$, this model is especially simple, since there is only a single linear function. The probability that a data point x_i belongs to a class $y_i = \{0, 1\}$, is given by the *logistic*, also called the *logit* or *sigmoid* function.

$$sigmoid(t) = \sigma(t) = \frac{1}{1 + e^{-t}} \tag{1}$$

If we have p features, then $\hat{x} \in \mathbb{R}^p$, with $p+1$ features we introduce $\hat{\beta} \in \mathbb{R}^{p+1}$, where we have a bias β_0. The variable t is of the form,

$$t = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p \tag{2}$$

[1]sklearn.datasets.load_breast_cancer — scikit-learn 0.23.2 documentation, 2020
[2]Hastie, Tibshirani and Friedman, 2020

We define the probability of each class as,

$$P(y = 1|\hat{x}, \hat{\beta}) = \frac{1}{1 + exp(-\hat{\beta}^T \hat{x})} = \sigma(\hat{\beta}^T \hat{x}) \tag{3}$$

$$P(y = 0|\hat{x}, \hat{\beta}) = 1 - P(y = 1|\hat{x}, \hat{\beta}) \tag{4}$$

To define the total likelihood for all possible outcomes from a data set $\mathcal{D} = \{(x_i, y_i)\}$, we use the *maximum likelihood estimation (MLE)* principle. We can approximate the likelihood in terms of the product of the individual probabilities of a specific outcome y, that is,

$$\mathbf{P}(\mathcal{D}, \hat{\beta}) = \prod_{i=1}^{n} [P(y = 1|\hat{x}_i, \hat{\beta})]^{y_i} [1 - P(y = 1|\hat{x}_i, \hat{\beta})]^{1-y_i} \tag{5}$$

We obtain the log likelihood and our cost/loss function as,

$$C(\hat{\beta}) = \sum_{i=1}^{n} (y_i \hat{\beta}^T \hat{x}_i - log(1 + exp(\hat{\beta}^T \hat{x}_i)) \tag{6}$$

The maximum likelihood estimator is defined as the set of parameters that maximize the log likelihood, we maximize with respect to $\hat{\beta}$. For logistic regression, the cost function is given by,

$$C(\hat{\beta}) = -\sum_{i=1}^{n} (y_i \hat{\beta}^T \hat{x}_i - log(1 + exp(\hat{\beta}^T \hat{x}_i)) \tag{7}$$

In statistics, equation (7) is known as the *cross entropy.* The cross entropy is a convex function of the weights $\hat{\beta}$, so a local minimum is a global minimum. We define a vector $\hat{y}$ with n elements y_i, an $n \times p$ matrix $\hat{\mathbf{X}}$ which contains the x_i values, and a vector $\hat{p}$ which contains the probabilities $P(y_i|\hat{x}, \hat{\beta})$. We also define the diagonal matrix $\hat{\mathbf{W}}$ where the elements are $P(y_i|\hat{x}, \hat{\beta})(1 - P(y_i|\hat{x}, \hat{\beta}))$. We can write the partial derivatives of the cost function as,

$$\frac{\partial C(\hat{\beta})}{\partial \hat{\beta}} = -\hat{\mathbf{X}}^T (\hat{y} - \hat{p}) \tag{8}$$

$$\frac{\partial^2 C(\hat{\beta})}{\partial \hat{\beta} \partial \hat{\beta}^T} = \hat{\mathbf{X}}^T \hat{\mathbf{W}} \hat{\mathbf{X}} \quad \text{(Hessian matrix)} \tag{9}$$

If we can compute these matrices, then we can solve these equations using Newton-Raphson's iterative method. The iterative scheme is given by,

$$\beta^{new} = \beta^{old} - (\hat{\mathbf{X}}^T \hat{\mathbf{W}} \hat{\mathbf{X}})^{-1} (-\hat{\mathbf{X}}^T (\hat{y} - \hat{p}))_{\beta^{old}} \tag{10}$$

Here the right hand side is computed with the old values of β.

2.2. **Decision trees.**

Decision trees are supervised learning algorithms that can perform both regression and classification. Here we will focus on the classification problem. A tree can be described as a nonlinear hierarchical data structure, that consists of nodes connected by edges. A decision tree is typically divided into a root node, the interior nodes, and the final leaf nodes or just leaves. These nodes are connected by branches, which are the edges.

The main idea is to find those descriptive features which contain the most *information* regarding the target feature, and then split the data set along the values of these features, such that the target feature values for the resulting underlying data sets are as pure as possible. The descriptive features which reproduce best the output features are normally said to be the most informative ones.The process of finding the most informative feature is done until we accomplish a stopping criteria, where we finally end up in leaf nodes.

2.2.1. *General features.* A decision tree takes a top-down approach,

- A leaf provides the classification of a given instance.
- A node specifies a test of some attribute of the instance.
- A branch corresponds to a possible value of an attribute.
- An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node.

2.2.2. *Classification tree features.* For a classification tree, we predict that each observation belongs to the most commonly occurring class of training observations in the region to which it belongs. We use recursive binary splitting to grow the tree. When building a classification tree, either the *Gini index* or the *entropy* are generally used to evaluate the quality of a particular split. In the classification problem, our targets take for example $k = 1, 2, ..., K$ values, then the only thing we need to think of is to set up the splitting criteria for each node.

We define a PDF p_{mk}, that represents the number of observations of a class k, in a region R_m with N_m observations. We represent this likelihood function in terms of the proportion $I(y_i = k)$ of observations of this class in the region R_m as,

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{11}$$

p_{mk} represents the majority class of observations in region m. The most common ways of splitting a node are given by,

- Misclassification error

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i \neq k) = 1 - p_{mk}$$

- Gini index, G

$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

- Entropy, s

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

2.2.3. *The CART training algorithm.* To set up the decision tree, we can typically use the CART algorithm. The popular library *Scikit-Learn* uses the *Classification And Regression Tree(CART)* algorithm to train the decision trees. For classification, the CART algorithm first splits the training set in two subsets using a single feature k and a threshold t_k. How does it choose k and t_k? It searches for the pair (k, t_k) that produces the purest subset using for example the gini factor. The cost function it tries to minimize is given by,

$$C(k, t_k) = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right} \tag{12}$$

- here $G_{left/right}$ measures the impurity of the left/right subset, and
- $m_{left/right}$ is the number of instances in the left/right subset.

Once it has successfully split the training set in two, it splits the subsets using the same logic, then the subsubsets and so on, recursively. It stops recursing once it reaches the maximum depth, or if it cannot find a split that will reduce impurity.

2.3. **Ensemble methods.** Ensemble methods are based on the idea that if we aggregate the predictions of a group of predictors, we will often end up with better predictions than with the best individual predictor. A group of predictors is called an *ensemble*, therefore this technique is called *ensemble learning*. The ensemble methods used in this article are discussed below.

2.3.1. *Voting Classifiers.* The aim of voting is to create a good classifier by combining several weak classifiers. A weak classifier is a classifier which is able to produce results that are only slightly better than guessing at random. To understand the idea, suppose we have trained a few classifiers, each one achieving about 80% accuracy. We may have a logistic regression classifier, a support vector machine(SVM) classifier, a random forest classifier, and a few more.

- A simple way to create an even better classifier is to aggregate the predictions of each classifier, and predict the class that gets the most votes. This majority vote classifier is called a **hard voting** classifier.
- If all classifiers are able to estimate class probabilities, then we can predict the class with the highest class probability, averaged over all the individual classifiers. This is called **soft voting**. This can give better results than hard voting because more weight is given to highly confident votes.

For the classification problem, the predictions are the majority vote of the diverse predictors in the ensemble.

2.3.2. *Bagging.* Bootstrap aggregation, or just bagging, is a general purpose procedure for reducing the variance of a statistical learning method. The main approach is to use the same training algorithm for every predictor, but to train them on different random subsets of the training data. When this sampling is performed with replacement, this method is called bagging. This allows training instances to be sampled several times for the same predictor. Once all the predictors are trained, the ensemble can make a prediction for a new observation by simply aggregating the predictions of all predictors. For classification tasks, the aggregation function is typically the statistical mode (the most frequent prediction).

The plain decision tree generally suffers from a high variance. This means that if we split the training data into two parts at random, and fit a decision tree to both halves, the resulting accuracy would be better than if we were to use a single tree.

2.3.3. *Random Forests.* Random forest is an extension over bagging. Random forests provide an improvement over bagged trees by implementing a small tweak that decorrelates the trees. As in bagging, we build a number of decision trees on bootstrapped training samples. But when growing these decision trees, each time a split in a tree is considered, a random sample of m features is chosen as split candidates from the full set of p features. The split is allowed to use only one of those m features. A fresh sample of m features is taken at each split, typically m is chosen to be $m \approx \sqrt{p}$. The algorithm for building random forests can be described as,

(1) Suppose we want to grow a random forest of say, B trees, then for b = 1 : B
    - Draw a bootstrap sample from the training data, which is organized in our $\mathbf{X}$ matrix.
    - We then grow a random forest tree T_b based on the bootstrapped data, by repeated the steps outlined until the maximum node size is reached.
(2) Select $m \leq p$ variables at random from the p features.
(3) Pick the best split point among the m features using for eg. the CART algorithm, and create a new node.
(4) Split the node into daughter nodes.
(5) Output the ensemble of trees $\{T_b\}_1^B$, and make predictions for the classification task.

3. Performance measures

To analyse the results of the classification task, we can use the following measures,

- Accuracy Score

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} I(t_i = y_i)}{n}$$

The accuracy is the number of correctly guessed targets t_i divided by the total number of targets. Here I is the indicator function, 1 if $t_i = y_i$, and 0 otherwise. Here t_i represents the target, y_i is the prediction from the model, and n is the total number of targets.

- Confusion Matrix

The rows of the confusion matrix represent an actual class, while the columns represent a predicted class. The elements in the the confusion matrix are the the number of true negatives(TN), false positives(FP), false negatives(FN), and true positives(TP). Figure 1 illustrates the matrix. A perfect model would have only true positives and true negatives.

- The Receiver Operating Characteristic(ROC) Curve

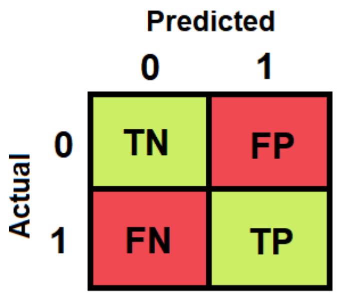


FIGURE 1. Elements in the confusion matrix for a binary classifier.

The ROC curve is a plot of the *true positive rate(TPR)* against the *false positive rate(FPR)*. The TPR(also called recall or sensitivity) is the ratio of positive instances that are correctly classified by the model, it is defined as

$$TPR = \frac{TP}{TP + FN}$$

The FPR is the ratio of negative instances that incorrectly classified as positive. The FPR is defined as,

$$FPR = \frac{FP}{FP + TN}$$

The higher the TPR, the lower the FPR. The dotted line represents the ROC curve of a purely random classifier(see plots in the results below), a good classifier stays as far away from this line as possible, towards the top left corner. We wish to have the ROC curves as close to the top left corner as possible, this would represent the best classifier. Another feature we want to measure is the *area under the curve(AUC)*. We want to have a ROC AUC as close to 1 as possible.

- The Cumulative gains curve

The cumulative gains curve shows the percentage of the total number of samples in a given class, ”gained” by targeting a percentage of the total number of samples. The diagonal line (see plots in the section below) in the plot is called the baseline curve, which shows the results of a random classifier. The greater the area between the baseline and the curve, the better the model. This means we want the gains curve to go up towards 1, as close to the top left corner of the chart, and then flatten out. There should also be a clear distinction between the two classes.

4. Exploring the features in the data set

4.1. **The correlation matrix.** The elements of the correlation matrix is the *correlation coefficient* between the features represented by the corresponding row and column. The correlation coefficient is a number between -1 and 1, that indicates the strength of the relationship between two variables. A value close to 1.0 indicates a strong positive correlation, this means that as the value of one variable increases, the value of the other variable increases as well. A value close to -1.0 indicates a strong negative correlation, this means the value of one variable decreases as the other increases, and vice versa. A value close to 0 indicates no correlation, this means that the two variable are independent of each other.

4.2. **The principle component analysis(PCA).** The PCA is a dimensionality reduction algorithm, and by dimensionality, we mean the number of features. The goal is to reduce the number of variables in the data set, while preserving the maximum amount of information. PCA identifies the axis that accounts for the largest amount of variance in the training set. It also finds a second axis, orthogonal to the first one, that accounts for the largest amount of remaining variance. It finds a third axis, orthogonal to both previous axes, and a fourth, a fifth, and so on, as many axes as the number of dimensions in the data set. The unit vector that defines the i^{th} axis is called the i^{th} principle component(PC). Before performing the PCA, the data set must be scaled.

One way to find the right number of dimension,is to use the *explained variance ratio* of each principle component. This indicates the proportion of the data set's variance that lies along the axis of each principle component. Plotting the explained variance as a function of the number of dimensions d, will allow us to see how much variance is contained in the first d dimensions.

5. Results of the Wisconsin Breast Cancer data analysis

Now we will present the results from the analysis of the Wisconsin breast cancer(diagnostic) data set. We want to classify a breast cancer tumour as malignant or benign. This data set consists of 569 samples, and 30 features that describe the tumours. There are 212 cases of malignant tumours, and 357 cases of benign tumours. All results presented for logistic regression, decision trees and ensemble methods, are produced from using 80% of the data for training, and 20% of the data for testing.

We will start by taking a look at the data set, and looking what kind of features are used to classify a tumour. First we will look at the correlation matrix, which is given in figure 2. I can see that the correlation between mean area and worst area is 1, and that there is no correlation between mean radius and worst fractal dimension.

I also want to look at the PCA of the data set. I make a plot of the features vs the explained variance, this is shown in figure 3. I can see that with just 2 dimensions, I can represent 99% of the data set, and with 4 dimensions, I can pretty much approximate the whole data set. This shows that many features are redundant, and approximately 4 features contain the most variance in this data set. However I will be using the whole data set in my further analysis, this is because there is no need for dimensionality reduction in this case. The data set has 30 dimensions, and 569 samples, it is quite a small data set. Reducing dimensionality does lead to loss of information and should be used if the training is too slow, but with this data set, I do not get slow training, so I will use the whole data set to get as accurate results as possible.

5.1. **Logistic regression.** With logistic regression I get an accuracy of 0.96 on the test data, and an accuracy of 0.98 on the training data. I also tried logistic regression with my own code, this is the same function that I used in

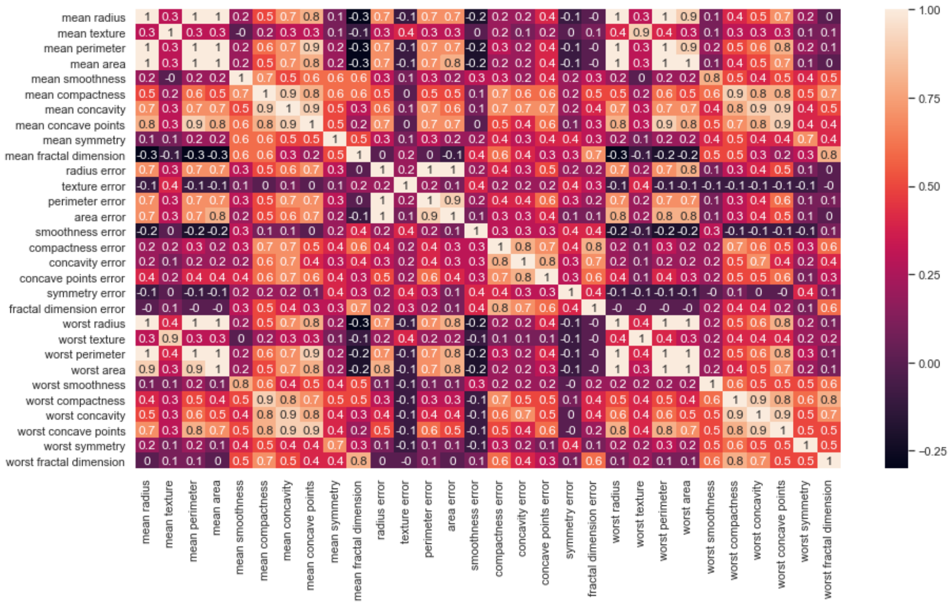


FIGURE 2. Correlation matrix for the Wisconsin breast cancer data.

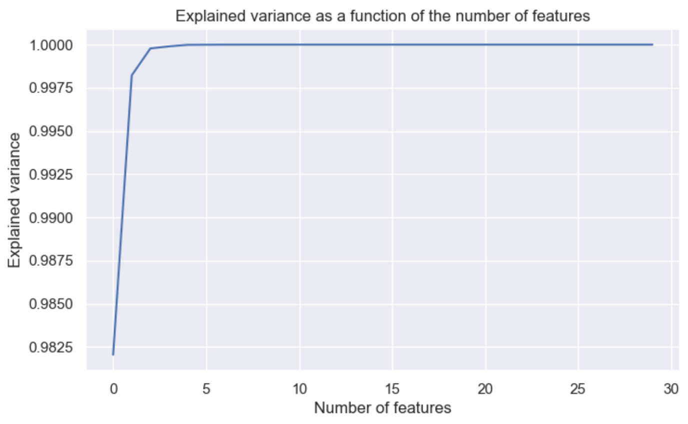


FIGURE 3. Explained variance as a function of the number of dimensions(features) on the entire data set for the breast cancer data. This data set has 30 features.

project two. With my own function I got an accuracy of 0.412 on the test set. This may be because the function is specialized for logistic regression with more than two classes, even when I tried to change the hyperparameters I could not get a better accuracy. I tried logistic regression with the training set using only the first 5 principle components, I got an accuracy of 0.978 on the training set, this is very close to 0.98, which is what I got using the whole training set. This shows that with just 5 PC's, I get an almost perfect approximation to the actual training set. Figure 4 shows the confusion matrix for the model, it gives TP=0.97,TN=0.96, FP=0.04 and FN=0.03.

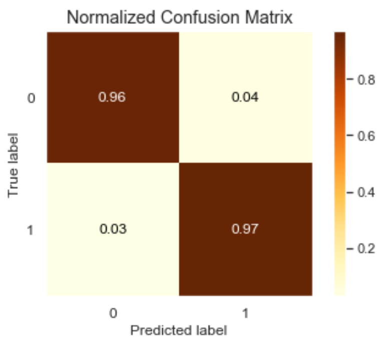
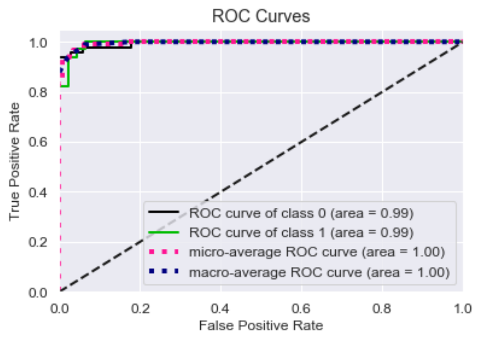
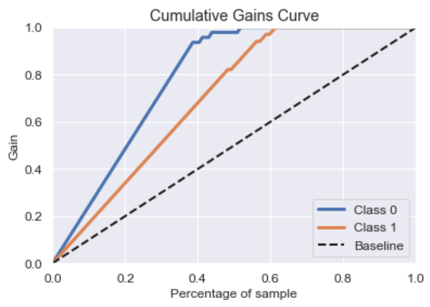


FIGURE 4. Normalized confusion matrix from logistic regression on the breast cancer data. This method gives TP=0.97, TN=0.96, FP=0.04 and FN=0.03.



(A) ROC curves for the model



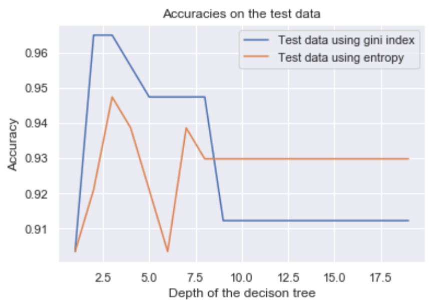
(B) Cumulative gains curves for the model

FIGURE 5. Looking at the performance of logistic regression on the breast cancer data.

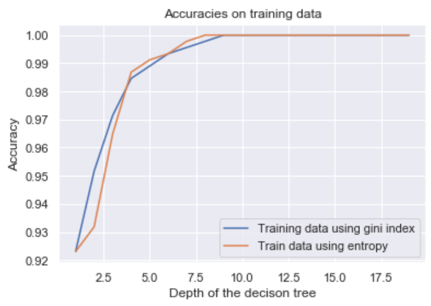
This model is almost perfect, as I can see that the ROC curves are very close to the top left corner, and have an area of 0.99. The gains curve show a clear distinction between the two classes, with 40% of the sample, I can get a gain of 1.0.

5.2. **Decision trees.** I have looked at the hyperparamters max_depth, which is the maximum depth of the tree, and $min\_samples\_leaf$, which is the minimum number of samples required to be at the leaf node.

When the maximum depth is set to none, I get an accuracy of 0.91 on the test set with the gini index, and an accuracy of 0.93 with the entropy. Here the entropy was slightly better than the gini index. To find the optimum tree depth, I have made a plot of the accuracies at various depths. This is shown in figure 6. I can see that the gini index gives a higher accuracy than the entropy. The best results are obtained at a tree depth of $2-5$, using the gini index. After that the accuracies begin to decrease, this is most likely due to overfitting of the model, which leads to poor performance on the test set. Varying the minimum samples at the leaf node, I get odd results. Figure 7a shows that the highest accuracy is achieved when the $min\_samples\_leaf$ is 3, or between $10-15$. However, the accuracies on

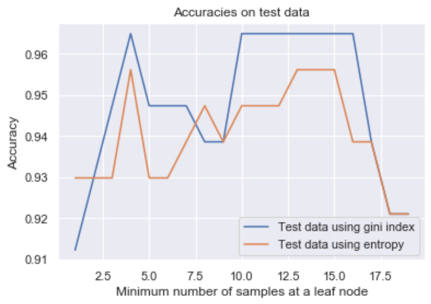


(A) Accuracies of the model on the test set.

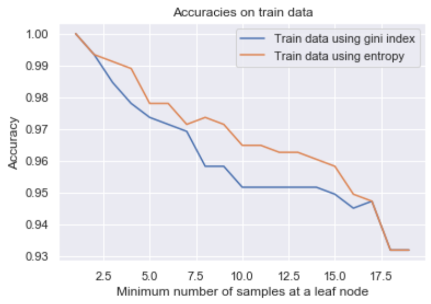


(B) Accuracies of the model on the training set.

FIGURE 6. Accuracies on the breast cancer data as the maximum depth of the decision tree is increased for the decision tree method.



(A) Accuracies of the model on the test set.



(B) Accuracies of the model on the training set.

FIGURE 7. Accuracies on the breast cancer data as the minimum number of samples required to be at a leaf node is varied in the decision tree method.

the training data begin to decrease as the $min\_samples\_leaf$ parameter is increased. This is strange, as I would expect the training set to have an increase in accuracy as $min\_samples\_leaf$ increases. Perhaps this parameter should remain unchanged.

The best results are gotten when the maximum depth of the tree is 3, using the gini index. This gives me an accuracy of 0.96 on the test set, and 0.97 on the training set. From the normalized confusion matrix I get TP=0.99, TN=0.94, FP=0.06 and FN=0.01. The ROC and cumulative gains curves were similar to the results obtained from logistic regression. The area under the ROC curve was 0.95.

5.3. **Ensemble methods.**

5.3.1. *Voting classifier.* From the voting classifer, I got pretty good results. The predictors I used in the hard classifier are logistic regression, random forest, SVM classifier, SGD classifier, and the voting classifier. The accuracies on the test set that I get from the hard classifer are 0.964 from the

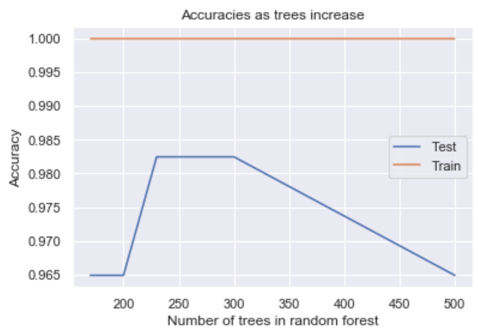


FIGURE 8. Accuracies on the breast cancer data as the number of trees in the random forests increases. The highest accuracy for the test set is between $240 - 300$ trees.

logistic regression, random forest, and the SVC. I got an accuracy of 0.947 from the SGD classifier, and finally 0.964 from the voting classifier.

The predictors I used in the soft classifier are logistic regression, randomm forests, SVM classifier and the voting classifier. I get an accuracy of 0.964, on the test set, from all of the predictors, including the voting classifier.

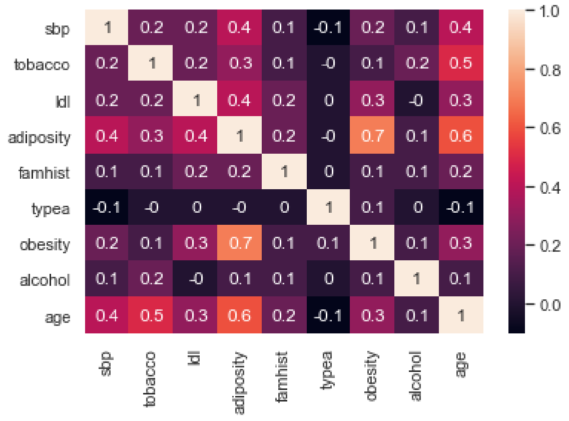
5.3.2. *Bagging.* For the bagging method, the best results I got was from using $n\_estimators = 253$, where $n\_estimators$ is the number of decision trees in the bagging classifier. I got an accuracy of 0.973 on the test set, and 1.0 on the training set. The confusion matrix for the model gave TP=0.99, TN=0.96, FP=0.04 and FN=0.01.

5.3.3. *Random forests.* Since I got the best results from bagging by using 253 decision trees, I tried the same number of trees in the random forest method, this gave me the best results with random forests as well. I also made a plot to see how the number of trees affects the accuracies of the model, this is shown in figure 8. The figure also shows that the best results are obtained with 253 trees. With 253 trees in the random forest, I get an accuracy of 0.982 on the test set, and 1.0 on the training set. The confusion matrix gives TP=0.99, TN=0.98, FP=0.02, and FN=0.01.

6. Results of the South African heart disease data analysis

Now we will present the results from the analysis of the South African heart disease data set. We want to predict if a person has coronary heart disease, or not. The data set consists of 462 samples, and 9 features that describe the risk factors for coronary heart disease. There are 160 cases of coronary heart disease, and 302 cases of no coronary heart disease. All results presented for logistic regression, decision trees and ensemble methods, are produced from using 80% of the data for training, and 20% of the data for testing.

We will start by looking at the data set, and what kind of features are present. This data set has only 9 features, so we can easily visualize them in the correlation matrix. This is shown in figure 9. I can see that there is a strong positive correlation between obesity and adiposity. The correlation



	sbp	tobacco	ldl	adiposity	famhist	typea	obesity	alcohol	age
sbp	1	0.2	0.2	0.4	0.1	-0.1	0.2	0.1	0.4
tobacco	0.2	1	0.2	0.3	0.1	-0	0.1	0.2	0.5
ldl	0.2	0.2	1	0.4	0.2	0	0.3	-0	0.3
adiposity	0.4	0.3	0.4	1	0.2	-0	0.7	0.1	0.6
famhist	0.1	0.1	0.2	0.2	1	0	0.1	0.1	0.2
typea	-0.1	-0	0	-0	0	1	0.1	0	-0.1
obesity	0.2	0.1	0.3	0.7	0.1	0.1	1	0.1	0.3
alcohol	0.1	0.2	-0	0.1	0.1	0	0.1	1	0.1
age	0.4	0.5	0.3	0.6	0.2	-0.1	0.3	0.1	1

FIGURE 9. Correlation matrix for the South African heart disease data. Here *sbp* is systolic blood pressure, *famhist* is family history of heart disease, and *ldl* is low densiity lipoprotein cholesterol.

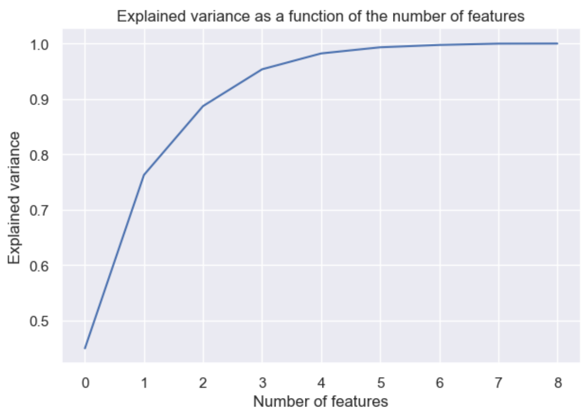
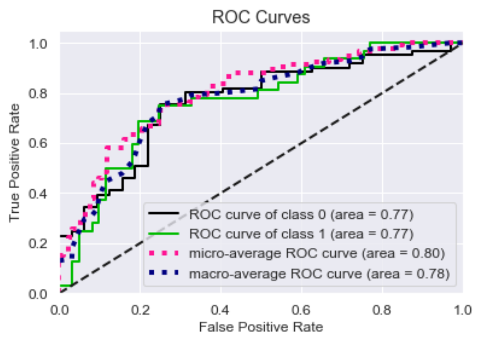


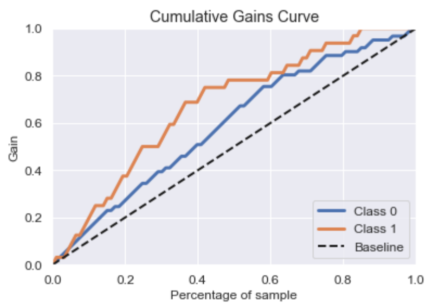
FIGURE 10. Explained variance as a function of the number of dimensions(features) on the entire data set, for the heart disease data set. This data set has 9 features.

between typea and alcohol is zero, so the two factors are independent of each other. The plot of the features vs explained variance is shown in figure 10. I can see that using just 2 principle components, I can represent 90% of the data set, and with 4 principle components, I can almost approximate the whole data set. This is very interesting to look at. I will be using the whole data set in my further analysis, since the data set is quite small, I do not need to reduce the dimensionality. Since the training of my models does not take a lot of time, I will keep the whole data set, to get the most accurate results.

6.1. **Logistic Regression.** With logistic regression I get an accuracy of 0.742 on the test set, and an accuracy of 0.732 on the training set. It is pretty odd that the test set has a slightly better accuracy than the training

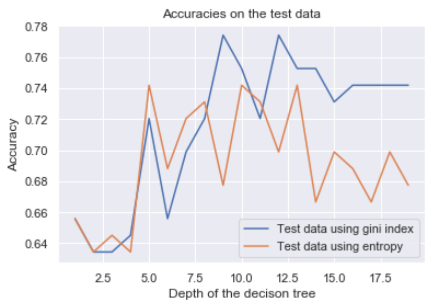


(A) ROC curves for the model

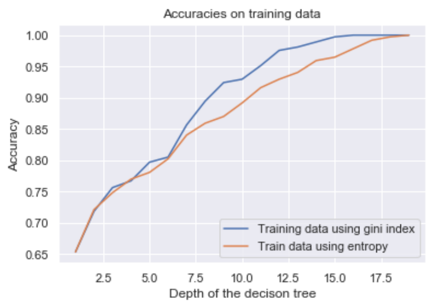


(B) Cumulative gains curves for the model

FIGURE 11. Looking at the performance of logistic regression on the heart disease data.



(A) Accuracies of the model on the test set.

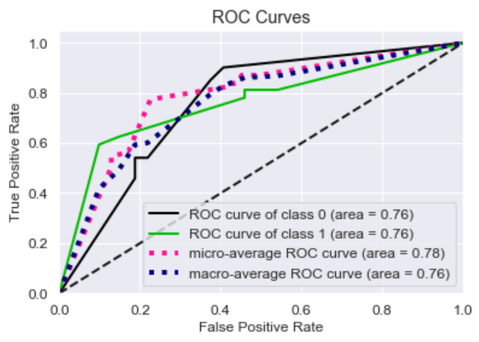


(B) Accuracies of the model on the training set.

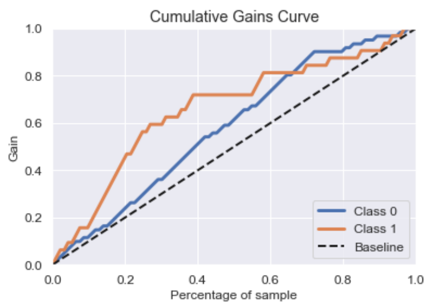
FIGURE 12. Accuracies on the heart disease data as the maximum depth of the decision tree is increased for the decision tree method.

set. From the confusion matrix, I get TP=0.50, TN=0.87, FP=0.13, and FN=0.50. The ROC curves for this model are going towards the top left corner, they do have the correct shape, so the model is good. This is shown in figure 11a. The cumulative gains curve does not have a clear distinction between the two classes in the beginning, although there is a separation between them. This is shown in figure 11b.

6.2. **Decision trees.** I have looked at the hyperparameter *max_depth*, which is the maximum depth of the decision tree. When the maximimum depth is set to none, I get an accuracy of 0.741 on the test set, and 1.0 on the training set. To improve this, I have find the optimum tree depth. I have plotted the accuracies at various depths. This is shown in figure 12. I can see from figure 12a that the best result is obtained for $max\_depth = 9$, with the gini index. Here also I can see that the gini index gives better results than the entropy. With a $max\_depth = 9$, on the decision tree, I get an accuracy of 0.774 on the test set, and 0.924 on the training set. From the confusion matrix I get TP=0.62 TN=0.85, FP=0.15, and FN=0.38. The ROC curves and the cumulative gains curves are shown in figure 13. The



(A) ROC curves for the model



(B) Cumulative gains curves for the model

FIGURE 13. Looking at the performance of decision trees with $max\_depth = 9$, with the gini index, on the heart disease data.

ROC curves seem to have the right shape, the ROC AUC is 0.76, the gains curve for the two classes seem to intercept and overlap each other as they go towards 1.

6.3. **Ensemble methods.**

6.3.1. *Voting classifier.* From the voting classifier, I got similar results to the logistic regression, and the decision tree. For the hard classifier I have used logistic regression, random forest, SVM classifier, SGD classifier, and the voting classifier. The accuracy that I got on the test set was 0.74 from the logistic regression, 0.72 from the random forest, 0.64 from the SVM, 0.72 from the SGDclassifier, and 0.698 from the voting classifier. For the soft classifier I have used logistic regression, random forest, SVM classifier, and the voting classifier. The accuracy that I got on the test set was 0.74 from the logistic regression, 0.72 from the random forest, 0.64 from the SVM, and 0.73 from the voting classifier. These results are good, but they are not any better than the ones from logistic regression or the decision tree.

6.3.2. *Bagging.* For the bagging method, the best results I got was from using $max\_samples = 300$, where $max\_samples$ is the number of samples drawn from the data set to train each decision tree, and with $n\_estimators = 290$, where $n\_estimators$ is the number of decision trees in the ensemble. With these parameters I get an accuracy of 0.742 on the test set, and 1.0 on the training set. From the confusion matrix I get TP=0.50 TN=0.87, FP=0.13, and FN=0.5.

6.3.3. *Random forests.* The best results that I got from random forest was with using $n\_estimators = 195$, where $n\_estimators$ is the number of decision trees in the forest. With this parameter and by using the gini index, I got an accuracy of 0.752 on the test set, and a1.0 on the training set. This is a pretty good result. From the confusion matrix, I get the values TP=0.53, TN=0.87, FP=0.13, and FN=0.47.

7. Discussion of the results

7.1. **Looking at the features.** Before performing any machine learning algorithm on our data set, it is helpful to take a look at the features that are present in the data set. The correlation matrix helps to see how the features are related to each other. In addition, the PCA helps us to see which features are most important for the data set, allowing us to reduce the dimensionality of the data. For example, for the breast cancer data, I saw that 5 PC's contain the most information about the data, where the total number of components was 30. Both of my data sets, where quite small, thus I did not need to reduce the dimensionality of my data sets.

7.2. **Comparing the different methods.** I have used three main methods for the binary classification task. Logistic regression is one if the simplest machine learning algorithms, and is easy to implement, it makes no assumptions about distributions of classes in feature space. Training a model with this method does not require high computation power. It gives good accuracy when the data set is linearly separable. I got pretty good results with logistic regression on both of my data sets.

Decision trees are very powerful algorithms, capable of fitting complex data sets. Decision trees closely mirror human decision making, thus making them easy to interpret, especially for small data sets. They do not require scaling or feature normalization during pre-processing of data. If there are missing values in the data set, it will not affect the decision tree, however in this case, I did not have any missing values.

If the decision tree is left unconstrained, it can adapt itself to the training data, fitting it very closely, and most likely overfitting it. This is something we have experienced in this article. For the breast cancer data, we can see that as the depth of the tree increases, the training set gets an accuracy of 1 after a tree depth of 5, this is when the model begins to overfit and the test accuracyies begin to decrease from 0.96 to 0.92. This can be seen in figure 6. This is why I have tried to find the optimum value for the tree depth, and the minimum samples in the leaf node, so that the model can generalize well to the test set. There are other hyperparamters that can be fine tuned, like the minimum samples required to split an internal node or the number of features to consider when looking for the best split. We must also decide on the splitting criteria, for the tree. In this case, I can see that the gini index gives better results, compared to entropy. This is because entropy lies between 0 and 1, while gini index lies between 0 and 0.5, hence it is better compared to entropy for selecting the best features.

For the voting classifier I did not get any good results, as compared to the other methods. The ensemble methods work best when the predictors involved, are as independent from one another as possible. This increases the chance of them making very different types of errors, which will improve the ensemble's accuracy. I tried to use different predictors in the voting classifier, but I got similar results from each predictor. Bagging did give me good results, this is because bagging does use bootstrap, so it generally results in better models. Random forests also gave me good results. Random forests are able to handle high dimensionality data very well, as well as

Wisconsin Breast cancer data						
Method	Accuracy	ROC AUC	TP	TN	FP	FN
Logistic regression	0.96	0.99	0.97	0.96	0.04	0.03
Decision tree	0.96	0.95	0.99	0.94	0.06	0.01
Bagging	0.97	0.99	0.99	0.96	0.04	0.01
Random forest	0.98	1.0	0.99	0.98	0.02	0.01

Table 1. Summary of the results obtained from the different models for the breast cancer data set. These are all obtained from the test set.

South African heart disease data						
Method	Accuracy	ROC AUC	TP	TN	FP	FN
Logistic regression	0.74	0.77	0.50	0.87	0.13	0.50
Decision tree	0.77	0.76	0.62	0.85	0.15	0.38
Bagging	0.74	0.73	0.50	0.87	0.13	0.50
Random forest	0.75	0.75	0.53	0.87	0.13	0.47

Table 2. Summary of the results obtained from the different models for the heart disease data set. These are all obtained from the test set.

missing values in the data set. Random forests and bagging also have some hyperparameters that need to be chosen. For the heart disease data set, I could have gotten better results with the bagging and random forests methods if I did a grid search to find the right parameters. I looked at the *n_estimators* parameter, to find the best results. There are other parameters such as *max_depth*, *min_samples_split*, *max_features*, *min_samples_leaf*, that can be chosen to find the best fit. This again leads to many parameters to tweak, which can make the training process slow, and computationally expensive.

7.3. **Finding the optimum model for the data sets.** Now we are in a good position to find the best model for the two data sets studied in this article. The results obtained from each method, are given in table 1, for the breast cancer data, and in table 2 for the heart disease data. I have not included the voting classifier because it did not give any better results, compared to the others.

For the breast cancer data, I got an accuracy of 0.96 from logistic regression and decision tree, however the ROC AUC for logistic regression is 0.99, while decision tree has 0.95. The closer the ROC AUC is to 1.0, the better the model. The decision tree classifies more TP's, but logistic regression classifies more TN's. However random forest gets the highest accuracy of 0.98, as well as a ROC AUC of 1.0. This method also classifies the most TP's and TN's. Therefore the best method for the breast cancer data is the random forest method.

For the heart disease data, I pretty much get an accuracy of 75%. Firstly I notice that I get an accuracy of 0.74 from logistic regression, and bagging. The values for TP, TN, FP and FN are also the same. However it seems like logistic regression is a better model compared to bagging, as it has a ROC AUC of 0.77, while bagging has 0.73. Random forest also gives similar results to logistic regression and bagging, the values for TP, TN, FP and FN are also very similar. Decision tree gets the highest accuracy of 0.77, with TP=0.62, TN=0.85, and ROC AUC of 0.76. The best method would be the decision tree, as it is complex enough to give decent results, bagging and random forest are computationally heavy, and dont lead to a dramatic improvement in results, and logistic regression is too simple. Logistic regression just classifies 0.2 more TN's compared to decision tree. Therefore the best method for the heart disease data is the decision tree.

8. Conclusion

In conclusion, we can say that machine learning methods are closely dependent on the data set being used. For the Wisconsin breast cancer data, the optimal method is the random forest method, with 253 decision trees. This gives an accuracy of 0.98 on the test set, with TP=0.99, TN=0.98, FP=0.02, and FN=0.01. For the South African heart disease data, the optimal method is the decision tree, with $max\_depth = 9$. This gives an accuracy of 0.77 on the test set, with TP=0.62, TN=0.85, FP=0.15, and FN=0.38. When it comes to the different methods, we can say that logistic regression is easy to implement, and very efficient for linearly separable data. Decision trees can model complex data sets, and ensemble methods generally result in better models. Although decision trees are prone to overfitting, we can easily find the right fit by fine tuning the hyperparameters. Bagging and random forests tend to reduce overfitting as they are an ensemble learning technique. However they have a much longer training time, as they have to generate a lot of trees. They also require some fine tuning of the parameters to get good results, but the gini index gives better results, as compared to entropy. This shows that the optimum parameters are important for any method to give the best results. To improve my results I would explore other parameters in bagging and random forests, to find the optimum parameters, as ensemble methods are supposed to give better results than a single predictor. I would also look at other predictors to get better results from the voting classifier.

9. References

(1) Scikit-learn.org. 2020. Sklearn.Datasets.Load_Breast_Cancer — Scikit-Learn 0.23.2 Documentation. [online] Available at: `https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html` [Accessed 15 December 2020].

(2) Hastie T., Tibshirani, R. and Friedman, J., 2020. South African Heart Disease. [online] Web.stanford.edu. Available at: `https://web.stanford.edu/~hastie/ElemStatLearn/datasets/SAheart.data`

[Accessed 15 December 2020].

(3) Hjorth-Jensen, M., 2020. Data Analysis And Machine Learning: Logistic Regression. [online] Compphysics.github.io. Available at: `https://compphysics.github.io/MachineLearning/doc/pub/week38/html/week38-reveal.html` [Accessed 15 December 2020].

(4) Hjorth-Jensen, M., 2020. Week 44: From Decision Trees To Bagging Methods. [online] Compphysics.github.io. Available at: `https://compphysics.github.io/MachineLearning/doc/pub/week44/html/week44-reveal.html` [Accessed 15 December 2020].

(5) Hjorth-Jensen, M., 2020. Week 45: Random Forests And Boosting. [online] Compphysics.github.io. Available at: `https://compphysics.github.io/MachineLearning/doc/pub/week45/html/week45-reveal.html` [Accessed 15 December 2020].

(6) Geron, A., 2017. Hands-On Machine Learning With Scikit-Learn And Tensorflow. pp.181-186, 211-215,.

(7) All codes can be found in the github repository `https://github.uio.no/shafaqns/FYS-STK4155/tree/master/Project3`