Predicting the onset of diabetes based on diagnostic measures

FYS-STK4155: Project 3

Kim-Roger Grønmo
http://github.com/kimgronmo/fys-stk4155-p3

December 18, 2022

Abstract

In recent years the changes to our work and leisure activities causes a transition into a more sedentary lifestyle. In combination with easier access to high calorie foods this causes an increase in lifestyle diseases. In this paper we use data from the Pima Indians Diabetes Database to predict the onset of diabetes based on diagnostic measures. This is a binary classification problem which we investigate and aim to solve using methods of classification as provided by the standard libraries in Scikit-learn. These include tree based methods, neural networks, support vector machines and logistic regression. We find that the Decision Tree Classifier is best suited with an accuracy score of 0.89. Although the Gradient Boosting Classifier had an accuracy score of 0.90, the Decision Tree Classifier had a better accuracy in predicting true positives. It is worth noting that the dataset had some issues with missing values. How this is resolved by either replacing or removing samples can impact the performance of the evauated models and is worthy of further investigations.

1 Introduction

Our modern lifestyle comes with certain costs. A more sedentary lifestyle with less excercise and easier access to high calorie content foods, causes increases in lifestyle diseases. A common one that is increasing in recent years is diabetes. This disease has complications which include health problems such as kidney disease, nerve damage, retinal disease, heart disease and stroke.

This provides a strong motivation for quickly identifying and preventing this disease. Although doctors have methods available for identifying diabetes in a patient, a big part of the problem is for the patients themselves to be aware of risk factors and what they themselves can do to prevent the onset of the disease. Machine learning algorithms might in the future enable doctors to clearly show patient how changes in a certain predictor variable, such as weight loss or lower blood sugar levels, can change the probability of either getting the disease or recovering from it.

Although the use of machine learning algorithms in practical treatment and disease preventions has its share of controversities, it is an important topic for future research.

Our goal in this project is to build and evaluate machine learning models in order to accurately predict whether or not the patients in the dataset have diabetes or not. As there are a lot of models to choose from we will not go into great depth of how each of them functions, but instead give a quick overview of most of them. For further information we refer the reader to the standard Scikit-learn documentation and the references provided in our reference section.

We begin this report by looking at some theory behind the models and methods we will use, including a brief overview of a decision tree. We then investigate our data set and do some data cleaning and feature engineering. We present some of our findings and discuss the results. The report is then concluded with a summary of our findings.

An interested reader can download both this report and supporting code from the following github address:

http://github.com/kimgronmo/fys-stk4155-p3

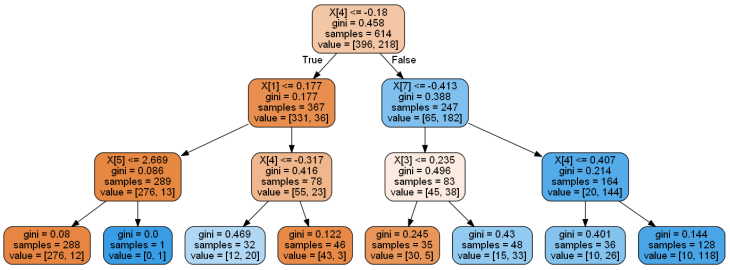


Figure 1: Example tree from the diabetes database with depth 3

2 Theory and Methods

The general references for this section is reference [1], [2] and [3]. For discussions about how each of these methods are implemented in Scikit-learn we refer the reader to the tool-kits homepage at [4]. We give a brief overview of some of the methods.

2.1 Decision Trees

Decision trees are supervised learning algorithms used for both classification and regression tasks. A decision tree is usually divided into a root node, interior nodes, and leaf nodes. These nodes are connected by branches. Each node specifies a test of some attribute. The branches corresponds to a possible value of an attribute, and the leaf node provide the classification of an instance. An instance is classified by starting at the root node of the tree, then the attribude of this node is tested. The instance is then moved down the tree branch corresponding to the value of the attribute tested. This process is repeated at the subtree of the next node.

The decision tree is then trained by giving it a number of training instances characterized by descriptive feature and a target feature. In our case of analyzing diabetes data, the target feature is the outcome represented by 0 for non diabetic and 1 for diabetic. The descriptive features are given by the medical measurements to be evaluated by the models.

The decision tree model is trained by continuously splitting the target feature along the values of the descriptive feature using a measure of information gain. This tree is grown until we accomplish a stopping criteria where we create leaf nodes. These leaf nodes represent the predictions that we make for each instance. In order to test the model we query instances to the tree

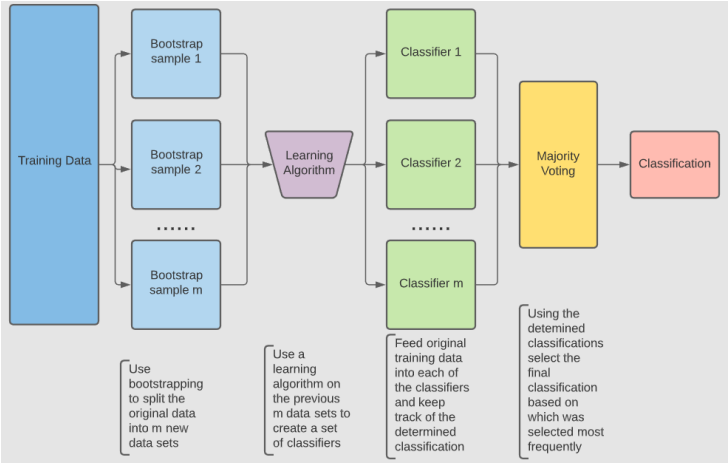


Figure 2: Flow chart of the bagging algorithm when used for classification source: [6]

and run down the tree until we reach a leaf node.

Our targets are the outcome of a classification process that takes $k = 1, 2, \ldots, K$ values. Let p_{mk} define a PDF that represents the number of observations of a class k in a region R_m with N_m observations. The most common ways of splitting a node are given by:

Gini index g

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

Information entropy s

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

2.2 Bagging, Voting and Random Forest

The aim 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.

Bootstrap aggregation, or bagging, is a procedure for reducing the variance of a statistical learning method. A Bagging classifier is an ensemble estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions, by voting or by averaging,

to form a final prediction. Bagging typically results in improved accuracy over prediction using a single tree. Figure 2 gives an overview of the process when used for classification.

The idea behind the VotingClassifier is to combine different machine learning classifiers and use a majority vote (hard vote) or the average predicted probabilities (soft vote) to predict the class labels. In majority voting, the predicted class label for a particular sample is the class label that represents the majority of the class labels predicted by each individual classifier. In contrast to majority voting (hard voting), soft voting returns the class label as argmax of the sum of predicted probabilities.

Random forests provide an improvement over bagged trees by way of a small tweak that decorrelates the trees. In random forests each tree in the ensemble is built from a sample drawn with replacement (a bootstrap sample) from the training set. Each time a split in a tree is considered fresh sample of m predictors is taken. The split is allowed to use only one of those m predictors.

Boosting methods combine weak classifiers into a strong classifier. A weak classifier is applied iteratively to modify the data. Each new classifier is trying to learn from the errors of the preceding classifier by emphasizing those observation which are misclassified and weighting them with a factor. In this project we used the ADAboost, GradientBoost and XGBoost classifiers included in the Scikit-learn package. We refer the reader to the relevant documentation at [4] for more information about how these are implemented.

3 The Pima Indians Diabetes Dataset

This dataset is from the National Institute of Diabetes and Digestive and Kidney Diseases. All patients in this set are females at least 21 years old of Pima Indian heritage. The objective of the set is to diagnostically predict whether or not a patient has diabetes.

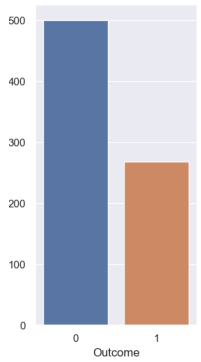
The dataset is available at [5]. A copy of it is also in the DataFiles directory in the github project folder.

3.1 Data Exploration and feature engineering

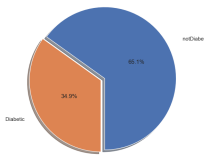
The datasets consist of several medical predictor variables and one target variable, Outcome. Outcome 1 represents the patient having diabetes, while 0 is non diabetic. The feature variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on. Note the diabetes pedigree function is a function which scores likelihood of diabetes based on family history.

Figure 3 shows the distribution of patients among the two outcomes. We can see that the data set is not very imbalanced. No single outcome greatly outnumbers the other. Figure 4 shows the correlation betweeen different features. We can see that Glucose levels have the biggest influence on the outcome, followed by insulin levels and BMI. From further exploration of the data set some obvious problems became apparent.

Although there were no missing values, certain factors were set to 0 indicating missing values. Features such as skin thickness, bmi etc are not possible to be at 0 for a living person. This was dealt with by splitting the data set in two (diabetic vs non diabetic), and replacing the 0 values with the average of the respective column for the given group. The two parts were then merged back into a complete set. Since the data set was rather small this seemed a better compromise than dropping the samples with the missing values.



(a) Number of patients with diabetes (1) and non diabetic (0)



(b) Distribution of patients

Figure 3: Overview of the patients in the data set

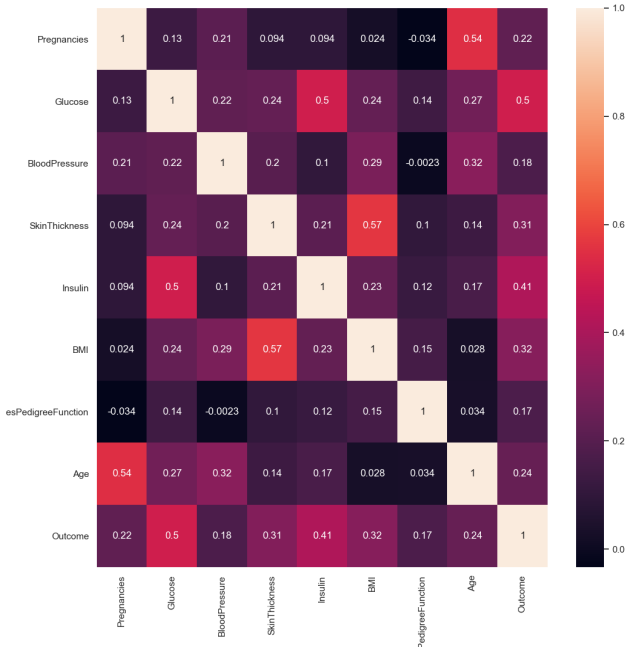


Figure 4: Correlation of different features

4 Results and Discussion

Further results for running the code are are included in the github folder. Figure 5 and 6 give the accuracy scores for the training data and test data.

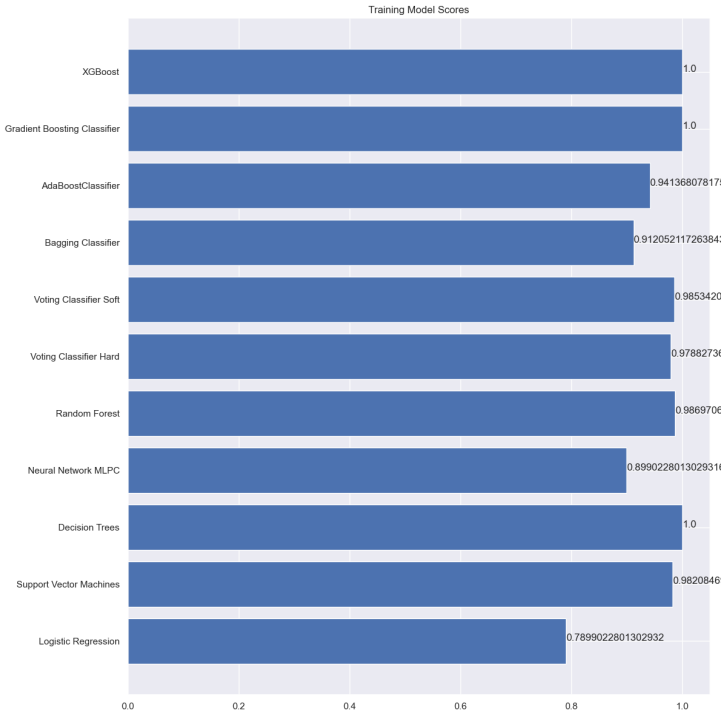


Figure 5: Training data model accuracy scores

We can see that several models reach 100 percent accuracy with the training data. The accuracy for the test data is not as high so the models are overfitting on the training data. This is apparent for Gradient Boosting, XGBoost and Decision Trees.

We can see that the best accuracy score for the test set is given by Gradient Boosting at 0.902, while Decision Trees, XGBoost and Voting Classifier Soft are right behind with 0.89. In order to differentiate these models we will look closer at the confusion matrices to see how well they can predict the actual disease and not just the cases where there are no disease outcome.

From Figure 7 and 8 we can see there are clear differences in how well the models perform in identifying the critial outcome (outcome 1: diabetes). The Decision Tree Classifier had the best performance with 86% clearly outperforming the other 3 methods.

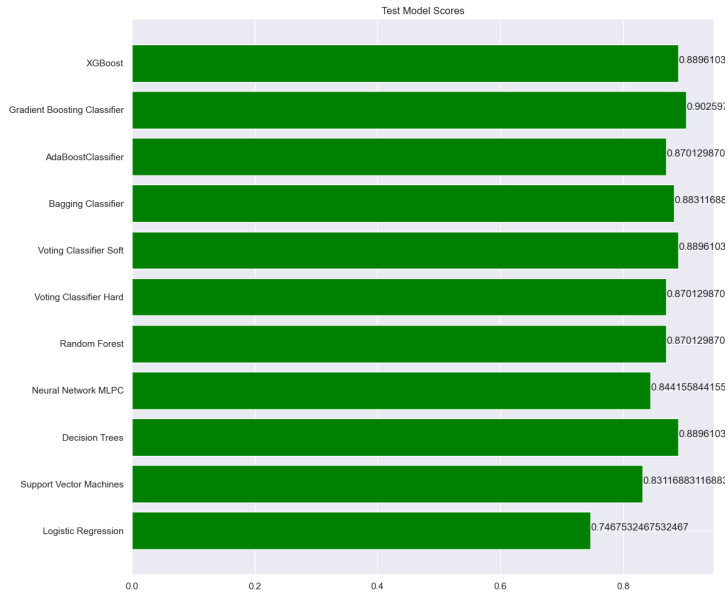
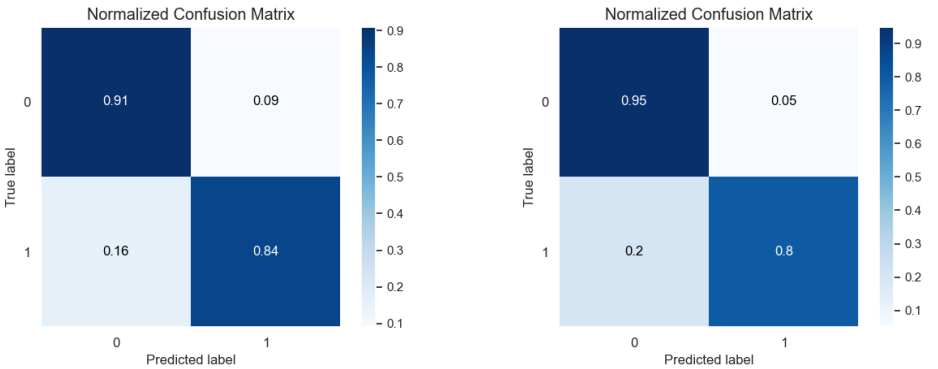
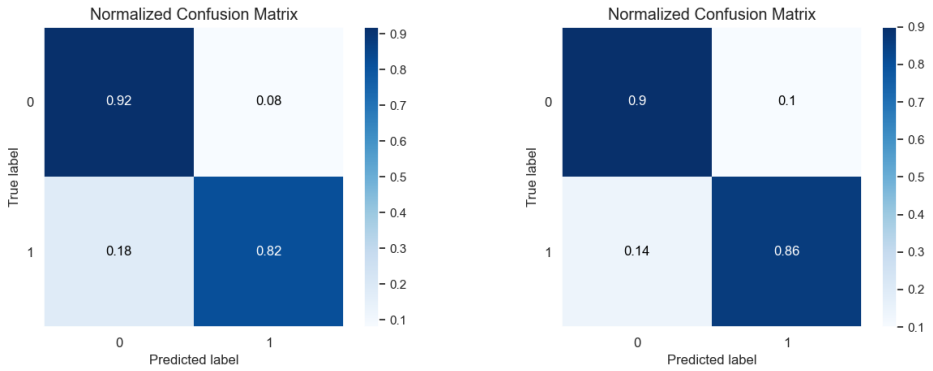


Figure 6: Test data model accuracy scores



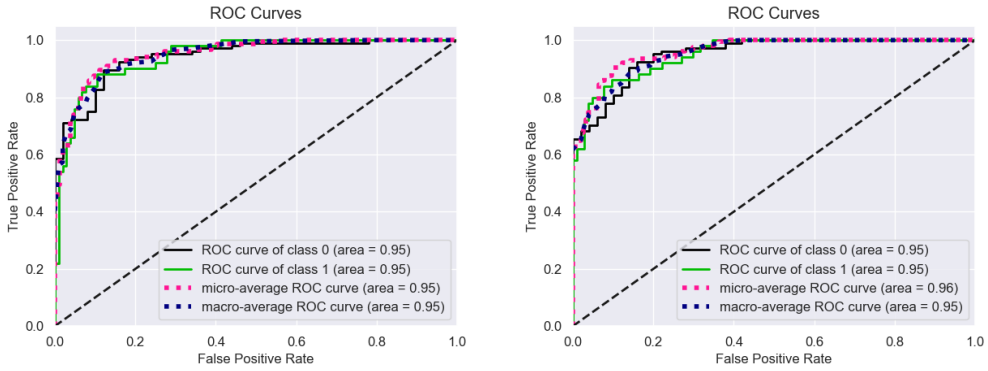
(a) Voting Classifier Soft (b) Gradient Boosting Classifier

Figure 7: Confusion Matrices



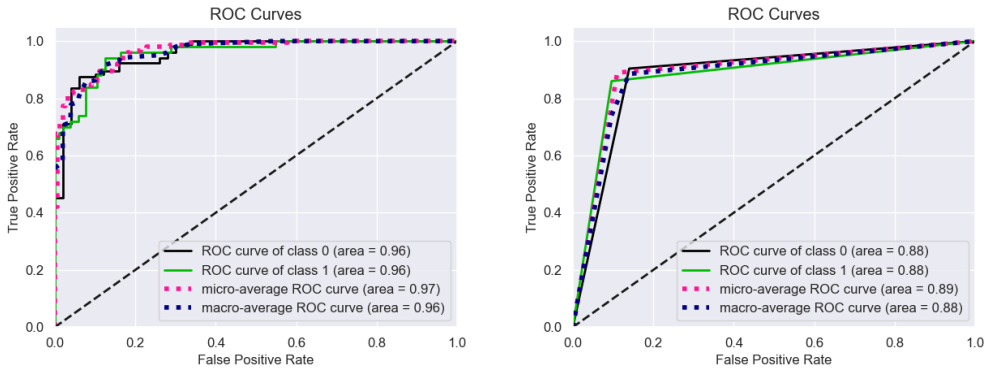
(a) XGBoost Classifier (b) Decision Trees Classifier

Figure 8: Confusion Matrices



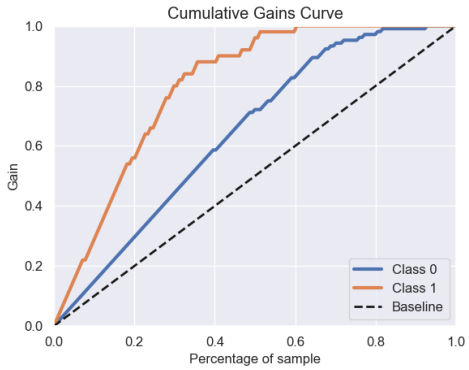
(a) Voting Classifier Soft (b) Gradient Boosting Classifier

Figure 9: Receiver Operating Characteristic Curve

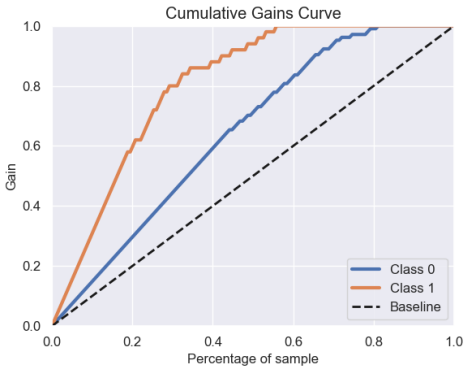


(a) XGBoost Classifier (b) Decision Trees Classifier

Figure 10: Receiver Operating Characteristic Curve

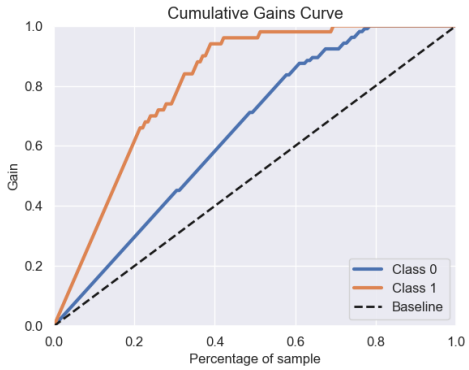


(a) Voting Classifier Soft

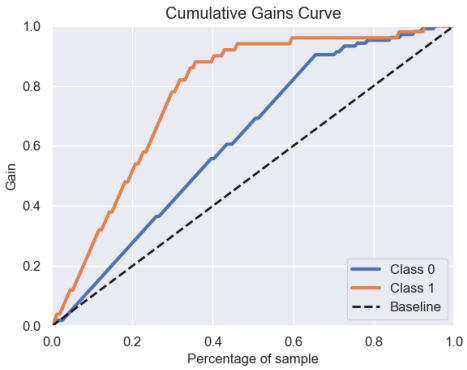


(b) Gradient Boosting Classifier

Figure 11: Cumulative Gains Curve



(a) XGBoost Classifier



(b) Decision Trees Classifier

Figure 12: Cumulative Gains Curve

5 Conclusion

In this report we have evaluated different models for predicting the onset of diabetes. We found that although several of these models had almost the same accuracy score, the DecisionTree Classifier was slightly better at correctly identifying diabetes outcome. In cases of illness it is often important to quickly administer treatments. It is better to be misclassified as a diabetic case and not have the disease, than it is to be classified as non diabetic and have the disease.

The dataset available had some issues with missing data. Combined with a somewhat small sample size this can effect the accuracy of our models. In order to improve our models we might investigate further how to split the data differently and also how to replace the missing values by for instance using an average of their nearest neighbours, rather than sample averages. Further investigation into this matter is necessary to draw a proper conclusion.

6 Bibliography

[1] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: data mining, inference, and prediction. 2009.

[2] Aurelien Geron. Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. O'Reilly Media, Inc.", 2019.

[3] Lecture Notes for FYS-STK4155 at:
https://compphysics.github.io/MachineLearning/doc/web/course.html
access date: 18.12.2022

[4] Scikit-Learn homepage at: https://scikit-learn.org/ access date: 18.12.2022

[5] The Pima Indians Diabetes Database at:
https://www.kaggle.com/uciml/pima-indians-diabetes-database
access date: 18.12.2022

[6] Wikipedia article: https://en.wikipedia.org/wiki/Bootstrap_aggregating
access date: 18.12.2022