Project 1 Veronika Ebenal, Raghavendra Tikare, Stanislav Klein

Scientific background of the project

This project is centered around training and analyzing different classifiers. Classification is a subset of supervised learning wherein a classifier is first given training data to learn how input attributes correlate with different classes. Then, the classifier can use this knowledge to assign new input to a class based on its attributes.

Goal of the project

The goal of this project was to train five different classifiers to determine the presence or absence of heart disease based on fourteen different attributes. The five different classifiers which were trained and tested were: random forest, decision tree, gradient boost, ada boost, and k neighbours classifier.

Description of the data and the data-preprocessing steps

The data used was taken from the Cleveland database wherein 303 patients were described using 14 attributes. The fourteenth attribute is the presence or absence of heart disease. We slightly edited this data in two ways before inputting it to our program. First, we added column names to each attribute. Second, we removed the six rows where data was missing. We then had our program perform pre-processing on the data before classification. Namely, one-hot encoding was done on the columns “restecg”, “thal”, “slope”, and “cp”. With one-hot encoding, each possible type of the attribute is represented with a binary array rather than an integer. The length of the array is the number of possible types. A 1 will be in the position in the array that represents that type, with a 0 in all other positions. This is done to emphasize that no ordinal relationship exists between the attribute types. For example, with attribute “cp” (chest pain), there are four types: 1 = typical angina, 2 = atypical angina, 3 = non-anginal pain, 4 = asymptomatic. There is no intuitive reason for the number ordering of the types, and we want to emphasize to the program that it is completely categorical.

Description and Comparison of the used methods

Random Forest

In a random forest classifier, multiple decision trees will vote on classification. Each decision tree decides its vote based on a random subset of attributes and samples. The final classification is the one voted by the maximum number of trees.

Decision Tree

In a decision tree classifier, a classification model is built in the form of a tree structure. It breaks down the dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes for each attribute, branching off according to possible types, and finally ending in leaf nodes for possible classifications.

Gradient Boost

In a gradient boost classifier, mistakes made in training are recorded and accounted for in each subsequent round of training. Therefore, decision trees are expanded in each iteration to account for possible bias in predictions and prediction accuracy is continually improved.

Ada Boost

An ada boost classifier works very similarly to gradient boosting in the sense that a weak learner is continually improved upon. In ada boosting, the weak learner will focus more on its "problem areas" (where it made incorrect predictions). As it improves, it will contribute to the final classification with a weight correlating to its accuracy.

K Neighbours

The k neighbours classifier works quite similarly to the random forest. However, instead of one vote which is decided from the majority for the final result, the k neighbour's result is decided upon the majority vote of its neighbours with the object being assigned to the class of most common among its k nearest neighbours.

Comparison of Classifiers

All of the used methods resulted in highly overlapping accuracy ranges. We have two rough categories of classifiers: tree-based and boosting-based classifiers. The tree based (random forest, decision tree, k neighbours) classifiers have the advantage of being completely unambiguous in its methods and results, as well as being resilient to a wide variety of classification types and inputs. Due to their transparent nature, they are also very easy to validate as well. The boosting classifiers (ada boost and gradient boosting) are advantageous when thinking about real-world application, as the learning process of the classifier makes it very effective against anomalies in inputs, which should always be a consideration. In the end, the gradient boost seemed the most accurate. This is likely because it takes the previously mentioned advantage of the random forest/k neighbours (that is, having multiple trees decide on the results) and builds on it by having an ensemble of models learn from one another in order to eventually form a very powerful committee. The worst classifier seemed to be decision tree. This makes sense because, while other classifiers rely on multiple votes from multiple trees, decision tree only uses one. This is a much less robust approach which we could expect would be prone to mistakes.

Results

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 14 columns):
age         297 non-null int64
sex         297 non-null int64
cp          297 non-null int64
trestbps    297 non-null int64
chol        297 non-null int64
fbs         297 non-null int64
restecg     297 non-null int64
thalach     297 non-null int64
exang       297 non-null int64
oldpeak     297 non-null float64
slope       297 non-null int64
ca          297 non-null int64
thal        297 non-null int64
target      297 non-null int64
dtypes: float64(1), int64(13)
memory usage: 32.6 KB
Random Forest Accuracy: 53.33%
Decision Tree Accuracy: 40.00%
Gradient Boost Accuracy: 70.00%
Adaboost Accuracy: 48.33%
KNeighborsClassifier Accuracy: 58.33%
```

All of the classifiers averaged at roughly the same accuracy, in a range from 40%-70%. We believe that the reason for the relatively low accuracy (compared to 80%-90% in the example code) is due to having five categories of output, instead of just two, leading to more potential for errors in classification.

Discussion: why is this a typical project for a data-scientist? (Or why not?)

This could definitely be considered a typical project for a data scientist. All of the characteristics of data science discussed in lecture were used in one way or another in this project.

Statistics and mathematics: accuracies of the different classifiers were calculated by training and testing the classifiers on multiple inputs. The accurate/inaccurate results were used to calculate the accuracy of the classifier.

Machine learning: all five classifiers used fall under the category of machine learning.

Programming: the project was actualized using programming in Python.

Data processing: one-hot encoding of the data is an example of data processing.

Data visualization: in order to better understand the data, we first visualized it with different plots and graphs.

Data knowledge: considering which parts of the data should be used (e.g. 14 attributes rather than all 76).