Jose Eduardo Cadena Bernal A01704641

Predict Heart Disease with decision tree/random forest.

Abstract

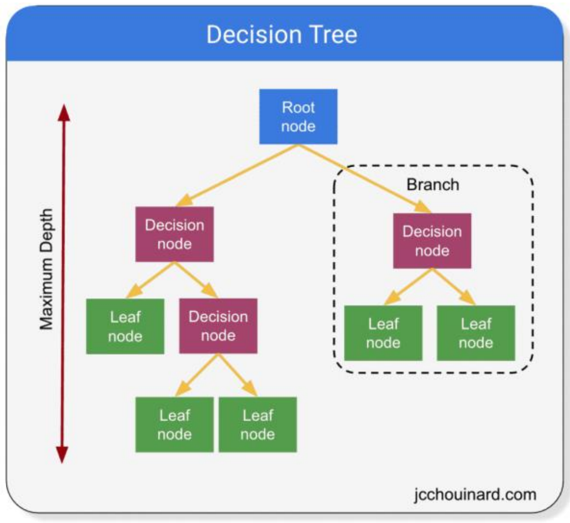
The current document presents the implementation of the Decision Tree and Random Forest algorithm to predict heart disease given your age, levels of cholesterol, blood pressure and sex and see the difference between them.

Introduction

Heart disease refers to several types of heart conditions. The most common type of heart disease in United States and Mexico is coronary artery disease, which affects the blood flow to the heart. Sometimes heart disease may be silent, thus diagnosis frequently occurs after the individual experiences' signs or symptoms of a heart attack, heart failure or arrhythmia. Artificial Intelligence can be a good tool to predict heart disease on time to prevent the symptoms already mentioned and reduce deaths from this disease.

State of the Art

Decision Tree is the most powerful and popular tool for classification and prediction. It is composed of flowchart-like structures in which each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. The paths from root to every leaf node represents the classifications rules.



Some advantages of using Decision Trees for classification tasks, are:

Simple to understand and to interpret, since trees can be represented graphically.

Requires little data preparation.

The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.

Able to manage both numerical and categorical data.

Able to manage multi-output problems.

Possible to validate a model using statistical tests.

Even though, decision tree has many advantages, not every problem can be solved using them. The disadvantages of this algorithm, are:

Decision-tree learners can create over-complex trees that do not generalize the data well, thus overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.

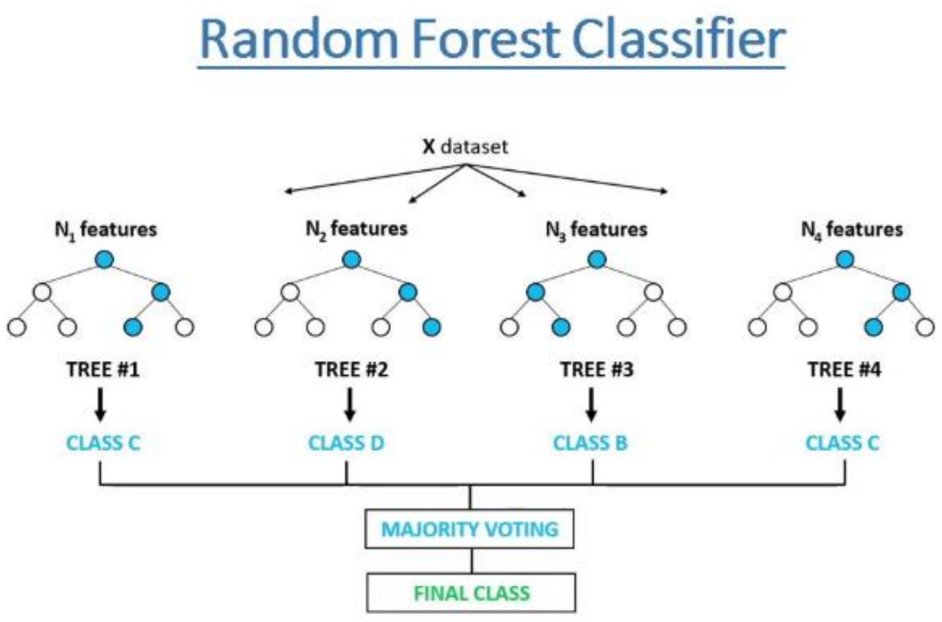
Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.

There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.

Decision tree learners create biased trees if the dataset is unbalanced. Therefore, it is recommended to balance the dataset prior to fitting with the decision tree.

Random forest algorithm is used to solve regression and classification problems, it utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex solutions. This consist of many decisions trees, and is trained through bagging or bootstrap aggregating

This algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.



Advantages

Highly accurate

No need to normalizing

Can handle several features at once

Can run trees in parallel ways

Perform both regression and classification tasks.

Disadvantages

They are biased to certain features sometimes

Slow- One of the major disadvantages of random forest is that due to the presence of many decision trees, the algorithm can become quite slow and ineffective for real-time predictions.

Can not be used for linear methods

Worse for high dimensional data

Dataset

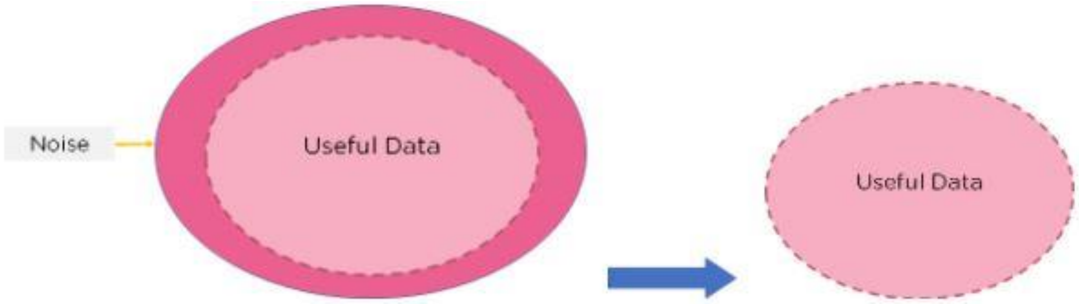
The data used was found in Kaggle, it contains 5 columns (age, sex, Blood pressure, cholesterol and heart disease) and 269 samples with int values, the dataset was downloaded as an .csv file, and read using pandas library in python

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 269 entries, 0 to 268
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   age            269 non-null    int64
 1   sex            269 non-null    int64
 2   BP             269 non-null    int64
 3   cholestrol     269 non-null    int64
 4   heart disease  269 non-null    int64
dtypes: int64(5)
memory usage: 10.6 KB
```

Implementation

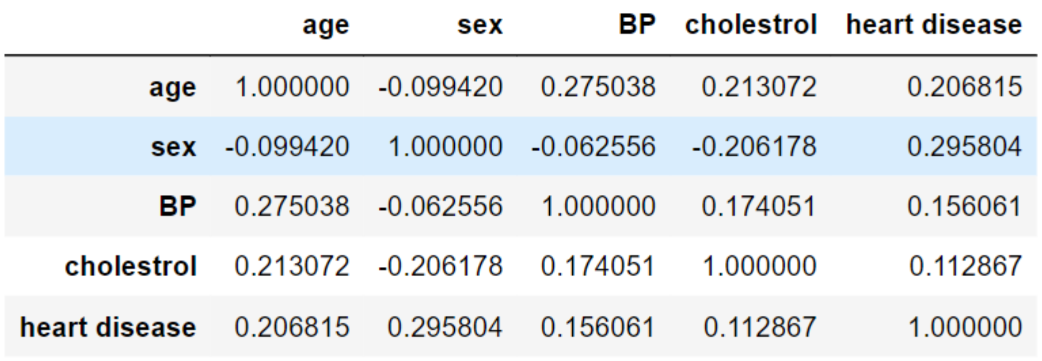
Feature Selection

Feature selection is very important before feeding data into machine learning algorithms, this is a method of reducing the input variable to your model by using only relevant data and getting rid of noise in data.



For feature selection I am using all the variables, because if I don't use all the variables the accuracy would go down.

Correlation explains how 2 variables are related to each other, the higher the correlation, is easier to predict one variable from another.

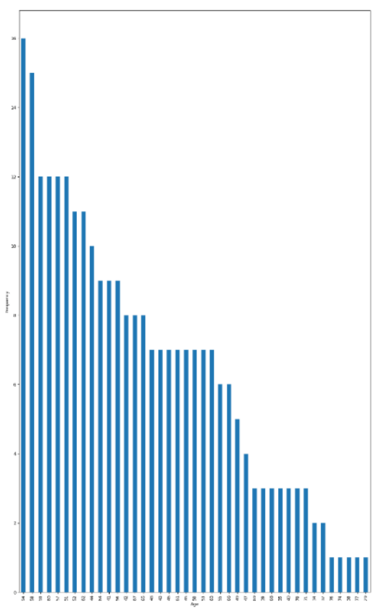


	age	sex	BP	cholestrol	heart disease
age	1.000000	-0.099420	0.275038	0.213072	0.206815
sex	-0.099420	1.000000	-0.062556	-0.206178	0.295804
BP	0.275038	-0.062556	1.000000	0.174051	0.156061
cholestrol	0.213072	-0.206178	0.174051	1.000000	0.112867
heart disease	0.206815	0.295804	0.156061	0.112867	1.000000

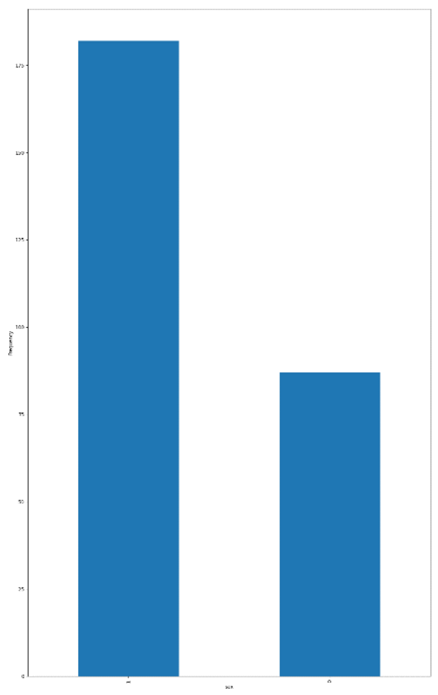
Exploratory data analysis

This refers to the process of performing initial investigation on data to discover patterns, anomalies, test hypothesis and check assumptions.

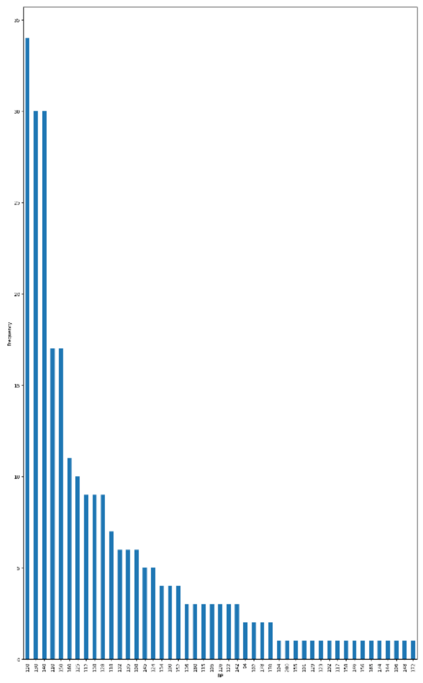
Age



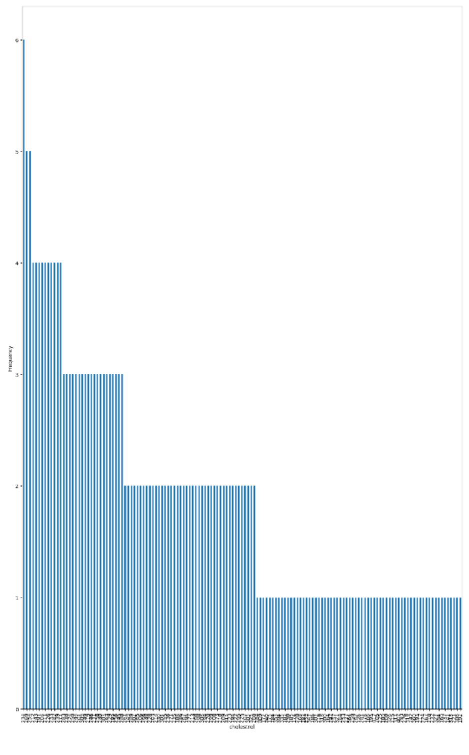
Sex



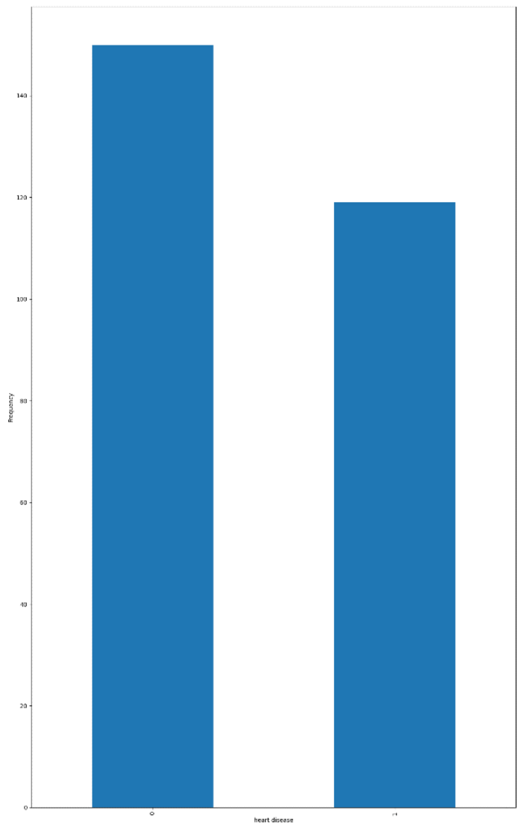
Blood Pressure



Cholesterol



Heart disease



Finding the best max depth for decision tree and random forest.

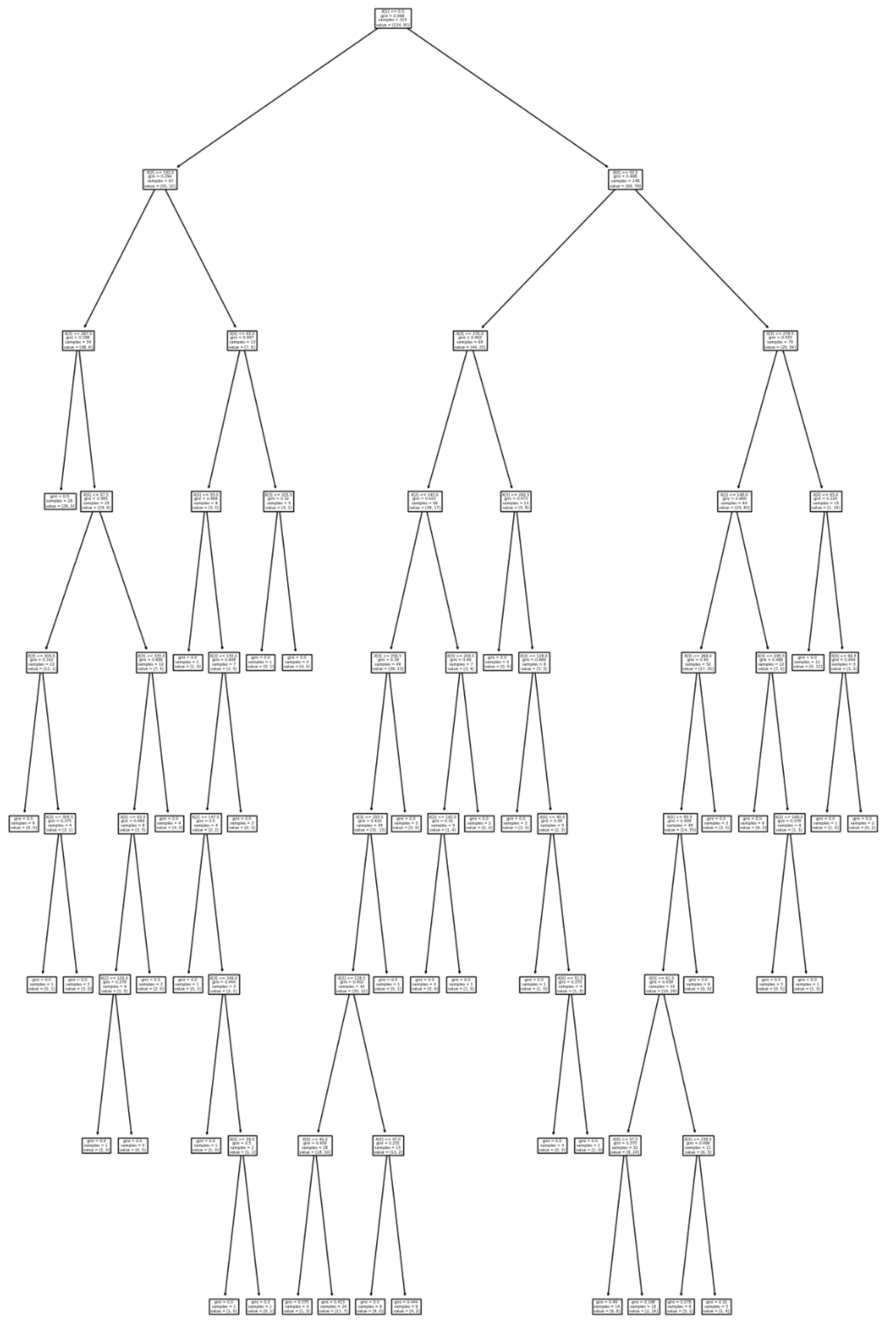
The max depth refers to the maximum depth of the tree.

```
clf = RandomForestClassifier(max_depth=6, random_state=0)
```

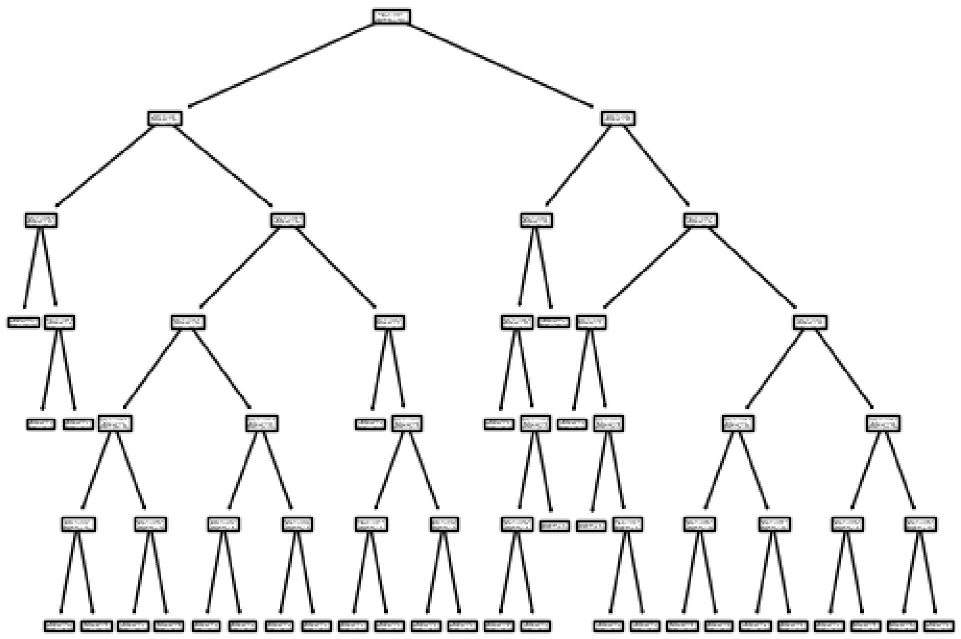
```
clf = tree.DecisionTreeClassifier(random_state=0,max_depth=8)
```

Decision tree

Jose Eduardo Cadena Bernal A01704641



Estimator in random forest



Classifying

Once the best parameters and the best max depth for decision tree and random forest was found, the dataset was split into training(80%) and testing(20%) data and then using RandomForestClassifier and DecisionTreeClassifier from sklearn the model was fitted and predictions were made.

For Random forest:

```
#split data
X_train, X_test, y_train, y_test = train_test_split(df_x, df_y,
                                                    test_size = 0.2,
                                                    random_state = 0)

clf = RandomForestClassifier(max_depth=6, random_state=0)


clf = clf.fit(X_train, y_train.values.ravel())

# Make predictions using the testing set
y_pred = clf.predict(X_test)

accuracy_score(y_test, y_pred)
```

For decision tree:

```
X_train, X_test, y_train, y_test = train_test_split(df_x,
                                                    df_y, test_size = 0.2,
                                                    random_state = 0)

clf = tree.DecisionTreeClassifier(random_state=0,max_depth=8)

clf = clf.fit(X_train, y_train)

# Make predictions using the testing set
y_pred = clf.predict(X_test)


accuracy_score(y_test, y_pred)
```

Evaluating Predictions

To evaluate the predictions of our model, I used sklearn metrics, which basically it compares real data (test data in our case) , and the predictions from the algorithm, in this case we had an accuracy for decision tree of 68% and 64% for random forest.

Conclusions

Notice that there is a different of 4% between the algorithms, furthermore they have a range between 60% and 70%.

This is due to the number of variables we have, to determine is someone have a heart disease exist more variables that could help (resting blood pressure, maximum heart rate achieved, levels of blood sugar are some examples)

Which one should you choose- decision tree or random forest?

Random forest is suitable for situations when we have a large dataset, and interpretability is not a major concern.

Decision trees are much easier to interpret and understand. Since random forest combines multiple decision trees, it becomes more difficult to interpret

In this specific case, the lack of variables and the amount of information the decision tree algorithm fits better.