Ain Shams University
CESS-CHEP

Mohammed Ehab Elsaeed

CSE385
Data Mining and Business Intelligence

Instructor: Dr. Mahmoud Mounir

Decision Trees, what are they, algorithms used, and how to use them?

Table of Contents

1 Table of Figures, Tables, and Equations 4
2 Abstract 5
3 Introduction 5
4 The Classification Problem 5
5 How does Classification work? 6
6 What is a Decision Tree? 7
7 Decision Tree Induction 7
8 Mathematical Formulation of Decision Trees 7
  8.1 Introduction 7
  8.2 Information Gain 7
  8.3 Information Gain Ratio 8
  8.4 Gini Impurity 8
9 Terms (Jargons) of a Decision Trees 9
10 The ID3 Algorithm 9
  10.1 Introduction 9
  10.2 Metrics 10
  10.3 Implementation 10
11 The C4.5 Algorithm 10
  11.1 Introduction 10
  11.2 Metrics 10
  11.3 Implementation 10
  11.4 Improvements over ID3 11
12 How are ID3 and C4.5 related to probability? 11
13 Example of Building a Decision Tree 12
  13.1 Root node and General Guidelines 12
  13.2 Numeric Data 14
  13.3 Ranked Data 15
  13.4 Multiple Choice Data 15
14 Example of Using a Decision Tree 15
15 Practical Work Dataset Description 16
16 Dataset Exploratory Analysis using Python 16
  16.1 Introduction 16
  16.2 Removing Outliers 17
  16.3 Histograms and Bar charts 19
  16.4 Statistical Data Analysis 21
    16.4.1 Data Modes 21

16.4.2 Standard Deviations 21
16.4.3 Variance 21
16.4.4 Skewness of Data 21
16.4.5 Description of Attributes 21
17 Dataset Weka Operations 23
17.1 Visualization of Data 23
17.2 Outlier Analysis Using DBSCAN 23
17.2.1 Parameters 23
17.2.2 Results 23
17.3 Reprocessing Data for K-Means and REPTree Decision Tree builder 24
17.4 Clustering Data using K-Means 24
17.4.1 Parameters 24
17.4.2 Results 25
17.5 Building a Decision Tree with REPTree 27
17.5.1 Parameters 27
17.5.2 Results 27
18 Conclusion 29
19 References 29
19.1 Papers 29
19.2 Textbooks 29
19.3 Articles 29
20 Appendix 30

1 Table of Figures, Tables, and Equations

Figure 1 Shows the learning and prediction processes of the classification problem 6
Figure 2 A simple decision tree 7
Figure 3 Sample from the dataset 12
Figure 4 How chest pain would separate patients with and without heart disease 12
Figure 5 Gini Impurity for each of the attributes 13
Figure 6 Shows the selected root node and how it has separated patients with and without heart disease 13
Figure 7 Sample from the dataset showing only the attribute of weight 14
Figure 8 Shows ordered weights and the calculated averages between each row 14
Figure 9 Shows the Gini Impurity for each average between rows 14
Figure 10 Shows a Decision Tree 15
Figure 11 Shows box plots before and after outliers removal 18
Figure 12 Shows histograms and bar charts of attributes of the dataset 20
Figure 13 Modes of Attributes (NaN Values indicate there is not two modes for the given attribute) 21
Figure 14 Standard deviations of dataset attributes 21
Figure 15 Variance of dataset attributes 21
Figure 16 Skewness of dataset attributes 21
Figure 17 Description of dataset attributes 22
Figure 18 Weka's visualization of the dataset attributes 23
Figure 19 K-means final clustering centroids 26
Figure 20 Resultant decision tree 27
Figure 21 Performance evaluation summary 28
Figure 22 Shows detailed accuracy for each cluster 28
Figure 23 Shows the confusion matrix 28

Table 17-1 Shows Weka's DBSCAN clustering results 23
Table 17-2 Shows Weka's K-means clustering results 27

Equation 1 Entropy of Dataset D 8
Equation 2 Conditional entropy of D given the value of attribute A 8
Equation 3 Information Gain 8
Equation 4 Intrinsic Value 8
Equation 5 Gain Ratio 8
Equation 6 Gini Impurity 8

2 Abstract

In today's world, where we have the privilege to obtain big data amounts, and even more sources of structured data are being added to the internet every day, data mining has become a very fundamental tool for decision makers all over the world be it in governments or businesses.

3 Introduction

In this paper, we discuss the classification of data using decision trees. We start by discussing what the classification problem, the importance of classification, and the contrast between solving the classification problem by human brains and machines. We then abstractly describe how machine classifiers are built by learning algorithms. We then discuss what decision trees are, and why they are considered a good choice for classification.

In addition, we go through explaining the mathematical formulation of decision trees and how we use it in ID3 and C4.5 algorithms. After having discussed the algorithms, we go on to build and use a decision tree using a numerical example.

Moreover, we obtain a dataset and explore it using Python and Weka. Exploration is done through visualization and statistical analysis of the dataset's attributes. Using Weka, we then run different clustering algorithms on our dataset and build a decision tree.

4 The Classification Problem

The classification problem is one of the main interests/fields in machine learning that has to do with making machines learn how to group data by specific features. Classification is a type of supervised learning where the objective is to classify data into predetermined groups. On the other hand, clustering is an unsupervised version of classification. In clustering, computers find common features by which to group data when categories are predetermined.

An example where classification is needed is the detection of spam email. To develop a system that can filter out spam emails, a machine learning algorithm is trained by being given a set of spam-like emails labeled as spam and non-spam emails labeled as non-spam. The goal is to make an algorithm that can learn features of spam emails from the training set to be used in spam filtering.

In today's world where big amounts of data are available and are used to make all kinds of decisions in the governmental, economical, medicinal fields, and more, classification is crucial. Classification is one tool that helps researchers with access privilege to massive masses of data to make sense of the data and find patterns.

To humans, classification is a natural ability. When visiting a car showroom, one can very accurately and easily classify vans, sedans, SUVs, and so on with very little error. In machine learning, classification is all about teaching computers to do the same. Unlike humans, classification done by machines sometimes requires the use of highly sophisticated machine learning algorithms.

[3]

5 How does Classification work?

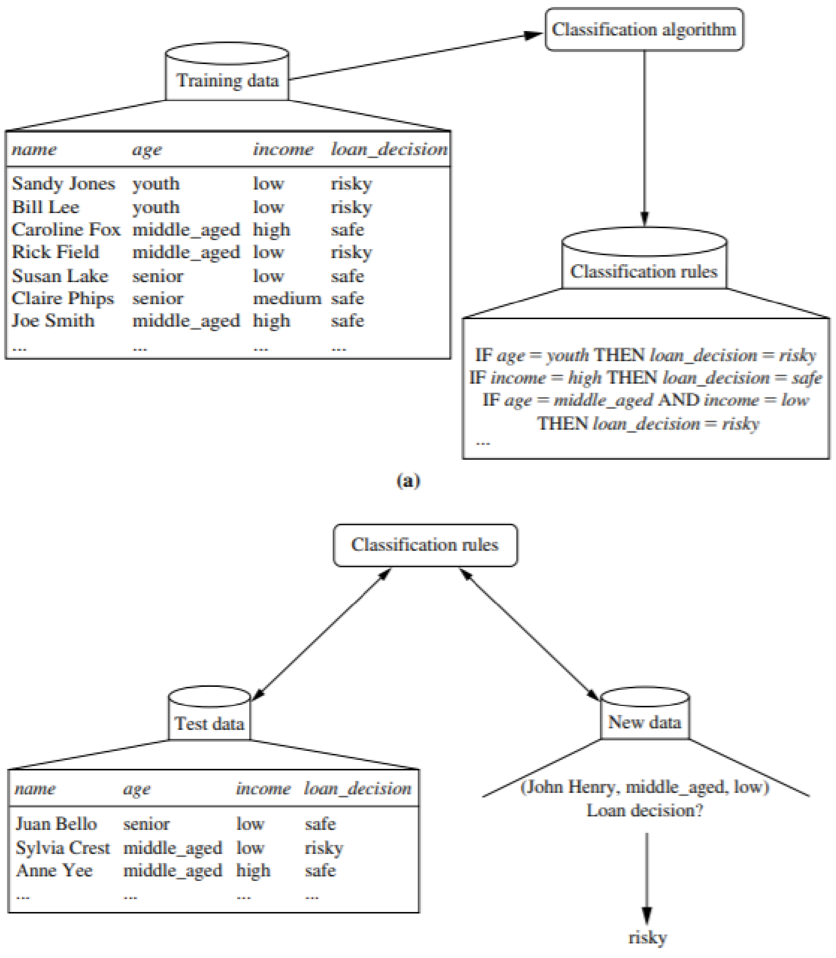
To implement classification of data, we need a two-phase process. The first phase is where a classification model is built from a dataset, this is called the learning (training) phase. The second phase is where we use the built model to predict class labels for given data, this is called the classification phase.

Building a classification model in the learning (training) phase uses a classification algorithm to build the classifier by analyzing or “learning from” a training dataset made up of entries from a database and their associated class labels.

To further clarify, we will assume we are building a classifier to tell us whether a certain greyscale image of dimensions 32x32 pixels is a cat or not. In our case, there are 32x32 = 1024 attributes (features) where each feature’s value is from 0 to 255 depicting the greyscale value of the pixel. We will gather a training dataset of 300 images where 200 of those images are cats and have the class label “cat” while the rest are images of other animals and have the class label “Not a Cat”.

We feed the dataset into our learning algorithm so that the learning algorithm can build the model by adjusting the model’s parameters using a loss(distance) function, and in turn, making the model better at predicting class labels by adjusting its “classification rules”.

Following the learning(training) phase, we feed the data we want to predict a class label for the built model where it is run against the “classification rules” formed during the learning (training) phase to produce a prediction for the class label.


Training data
Classification algorithm
Classification rules
IF *age* = *youth* THEN *loan_decision* = *risky*
IF *income* = *high* THEN *loan_decision* = *safe*
IF *age* = *middle_aged* AND *income* = *low* THEN *loan_decision* = *risky*
...
(a)
Classification rules
Test data
New data
(John Henry, middle_aged, low)
Loan decision?
risky

name	*age*	*income*	*loan_decision*
Sandy Jones	youth	low	risky
Bill Lee	youth	low	risky
Caroline Fox	middle_aged	high	safe
Rick Field	middle_aged	low	risky
Susan Lake	senior	low	safe
Claire Phips	senior	medium	safe
Joe Smith	middle_aged	high	safe
...	...	...	...

name	*age*	*income*	*loan_decision*
Juan Bello	senior	low	safe
Sylvia Crest	middle_aged	low	risky
Anne Yee	middle_aged	high	safe
...	...	...	...

Figure 1 Shows the learning and prediction processes of the classification problem

[4, CH. 8]

6 What is a Decision Tree?

A decision tree is a flowchart-like tree structure, where each node represents a condition on an attribute value, each branch from the node represents an outcome of the condition, and leaf nodes represent classes or class labels (distributions). Classification rules can easily be deduced from decision trees.

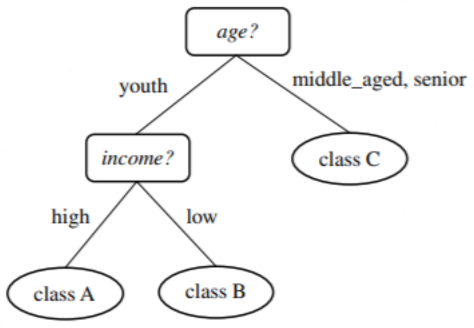


Figure 2 A simple decision tree

[4, Ch. 1]

7 Decision Tree Induction

Decision tree algorithms like ID3, C4.5, and CART were initially invented for the classification problem. A flowchart-like structure where each non-leaf node represents a condition on an attribute, where each branch corresponds to an outcome of that condition, and where each leaf node represents a prediction of a class or class label prediction is constructed using induction.

The algorithm chooses an attribute that best splits the data into distinct classes at each node. A tree is constructed from the given data when decision tree induction is used for attribute subset selection. Only attributes that appear in the tree are assumed to be relevant, others are assumed to be irrelevant. The reduced subset of attributes is the set of attributes appearing in the tree form.

[4, Ch.3]

8 Mathematical Formulation of Decision Trees

8.1 Introduction

As mentioned before, we need to choose attributes that are chosen for classification. To do this, tests of statistical significance which assume that the attributes are independent of one another are typically used. Different mathematical methods are herein discussed.

8.2 Information Gain

Generally, the expected information gain is the difference in information entropy of a dataset D “Info(D)” from a past state to a state that has some attribute’s value as a given, In other words, the conditional entropy of “Info(D)” given the value of some attribute “A”. This is denoted as “$\text{Info}_A(D)$”.

Equation 1 Entropy of Dataset D

$$Info(D) = -\sum^{m} p_i \log_2(p_i),$$

Equation 2 Conditional entropy of D given the value of attribute A

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

Equation 3 Information Gain

$$Gain(A) = Info(D) - Info_A(D).$$

8.3 Information Gain Ratio

The information gain ratio is the ratio of Information Gain and Intrinsic Information. The information gain ratio makes the decision tree biased against considering attributes with too many different values. Information Gain hence solves a disadvantage of information gain.

Equation 4 Intrinsic Value

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right).$$

Equation 5 Gain Ratio

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}.$$

8.4 Gini Impurity

Another way used in the CART algorithm is by using the Gini Impurity. It measures how frequently a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset.

Equation 6 Gini Impurity

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

Where pi is the probability that an entry in the dataset belongs to class Ci, the sum is computed over m classes.

[9]

9 Terms (Jargons) of a Decision Trees

- Sub-Node:
    - Meaning: A node within another node.
    - Intuition: A sub-X means something that descends from/is part of X.
- Root Node
    - Meaning: the base of the tree, where under its branches lies the entire dataset.
    - Intuition: It is the root of the decision tree from which all nodes originate.
- Splitting
    - Meaning: Dividing a node into two or more sub-nodes.
    - Intuition: Like using a knife to split a fruit, just for nodes.
- Decision Node
    - Meaning: A sub-node that has branches to other sub-nodes.
    - Intuition: Nodes that help determine your path.
- Leaf Node
    - Meaning: Nodes at the tip of the tree, the do not further split.
    - Intuition: Like a real tree, nothing branches out of a leaf.
- Pruning:
    - Meaning: Removing sub-nodes of a decision node, In other words, the reverse process of splitting.
    - Intuition: Pruning trees is trimming them, equivalently, it is also the removal of nodes.
- Sub-Tree:
    - Meaning: A part of the entire tree.
    - Intuition: As mentioned, A sub-X means something that descends from/is part of X, so, a subtree is part of a tree.
- Parent and Child Node:
    - Meaning: A node that divides to sub-nodes is called a parent node. Its sub-nodes are its children.
    - Intuition: Nodes are modeled as parents and their children.

[7]

10 The ID3 Algorithm

10.1Introduction

ID3 stands for Iterative Dichotomiser 3 and is called as such since at each step the algorithm repeatedly divides (iteratively dichotomizes) data into two or more groups. It was developed by Ross Quinlan, a machine learning researcher, and utilizes a greedy top-down approach to construct the tree. Simply put, the top-down approach implies that the construction of the tree is started from above and the greedy approach implies that we pick the best attribute now to construct a node at each iteration. Mostly, ID3 is used for nominal feature classification problems only.

10.2 Metrics

ID3 uses information gain to select the best attribute. As mentioned in the section on "Mathematical Formulation of Decision Trees," Information Gain calculates the change in entropy and measures how well data is separated or classified by a given attribute. The feature with the highest gain is chosen.

For binary classification, which means only two class labels exist, entropy will be zero if all class labels are homogeneous (similar) and 1 if not.

10.3 Implementation

1. Information Gain is calculated for all attributes.
2. Assuming that all rows are not of the same class, we split the dataset into subsets using the attribute with the maximum Information Gain.
3. Make a node using the attribute with the maximum information gain.
4. If all rows belong to the same class, make the current node a leaf node with the class as its label.
5. Repeat for the remaining attributes we are done with all attributes, or the decision tree has all leaf nodes.

[10]

11 The C4.5 Algorithm

11.1 Introduction

After ID3, Ross Quinlan presented C4.5, often referred to as a statistical classifier and ID3's successor, which then became a benchmark to which newer supervised machine learning algorithms are often evaluated. Developers of Weka, the machine learning software, described the C4.5 algorithm in 2011 as "a landmark decision tree program that is probably the machine learning workhorse most widely used in practice to date". After ranking at the top at the 10 Algorithms in Data Mining pre-eminent paper published by Springer LNCS in 2008, it became very popular.

[5]

11.2 Metrics

Like ID3, at each node of the tree, C4.5 chooses the attribute of the data that best splits its dataset using normalized information gain. The C4.5 algorithm then recurses on the partitioned sub-lists.

[1]

11.3 Implementation

To begin, the C4.5 algorithm has some base cases:

- If all rows belong to the same class, create a leaf node with the class name as its label.
- If no information gain is provided by any attribute. create a decision node higher up the tree using the expected value of the class.
- If an unencountered class is encountered, create a decision node higher up the tree using the expected value.

Algorithm:

1. base cases are checked.
2. Calculate the normalized information gain ratio from splitting on all attributes.
3. Keep the variable a_best as the attribute with the maximum normalized information gain.
4. Create a decision node that splits on a_best.

5. Call the function recursively on the sub-lists created by splitting on attribute a_best and add those nodes as children of the node.

11.4Improvements over ID3

- Unlike ID3, C4.5 can create a threshold on which it then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it so that it can deal with continuous data in addition to being able to deal with discrete data like ID3.
- Can handle entries with missing attribute values. Missing values are simply not used in gain and entropy calculations. It allows attribute values to be marked as "?" for missing.
- Can handle attributes with differing costs.
- C4.5 passes again over the built trees to start pruning them attempting to remove branches that do not help by replacing them with leaf nodes.

[2]

12 How are ID3 and C4.5 related to probability?

As stated before, both ID3 and C4.5 use Information gain which, as mentioned before, for some state is the effective change in entropy. But what is entropy, and how is it related to **probability**?

Entropy is the measure of data in terms of uncertainty or randomness. Intuitively, it is showing the predictability of an event. If an outcome of an event is definite, having 100 percent **probability**, entropy is zero (because there is no randomness), and if an outcome has 50 percent **probability**, entropy takes the maximum value (i.e. equals 1 as it is the log base 2) as it projects perfect randomness. Consider, for example, a coin toss that has a head **probability** of 0.5, and a tail **probability** of 0.5. Here, entropy is the highest possible value (i.e., equals 1), since there is no chance of determining the result accurately.

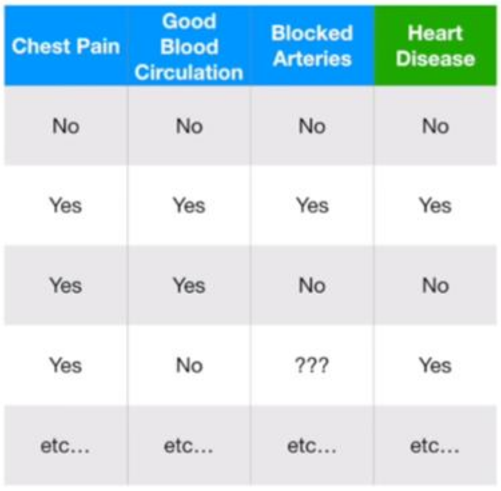
Alternatively, imagine a coin that has heads on both sides, the result of such a case can be accurately calculated because we know in advance it will always be heads. This event, in other words, has no randomness, and therefore its entropy is zero. ID3 and C4.5 follow the rule: A 0 entropy branch is an endpoint (leaf node). A division with more than 0 entropy needs to be further broken. In the event that zero entropy in the leaf nodes cannot be achieved, the decision is made by a simple majority method.

[8]

13 Example of Building a Decision Tree

13.1 Root node and General Guidelines

Let us assume we are trying to build a decision tree for the following dataset. We are building a decision tree to predict whether a patient has heart disease by using chest pain, good blood circulation, and blocked arteries as our attributes, we will add more later. We will be using the Gini impurity as a metric to build the decision tree.



Chest Pain	Good Blood Circulation	Blocked Arteries	Heart Disease
No	No	No	No
Yes	Yes	Yes	Yes
Yes	Yes	No	No
Yes	No	???	Yes
etc...	etc...	etc...	etc...

Figure 3 Sample from the dataset

We first need to determine which node should be the root node, we start by checking how well chest pain separates heart disease by creating a small tree with chest pain as the root. We then examine all data in our dataset counting how many people with chest pain have heart disease and how many do not, we then count how many people without chest pain have heart disease and how many do not. We then calculate the Gini impurity for using chest pain as a root node.

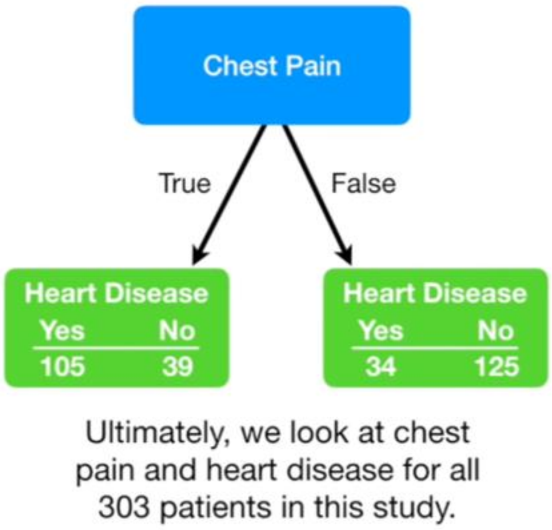

Chest Pain
True
False
Heart Disease
Yes No
105 39
Heart Disease
Yes No
34 125
Ultimately, we look at chest pain and heart disease for all 303 patients in this study.

Figure 4 How chest pain would separate patients with and without heart disease

We repeat the process of calculating Gini for Good blood circulation and blocked arteries.

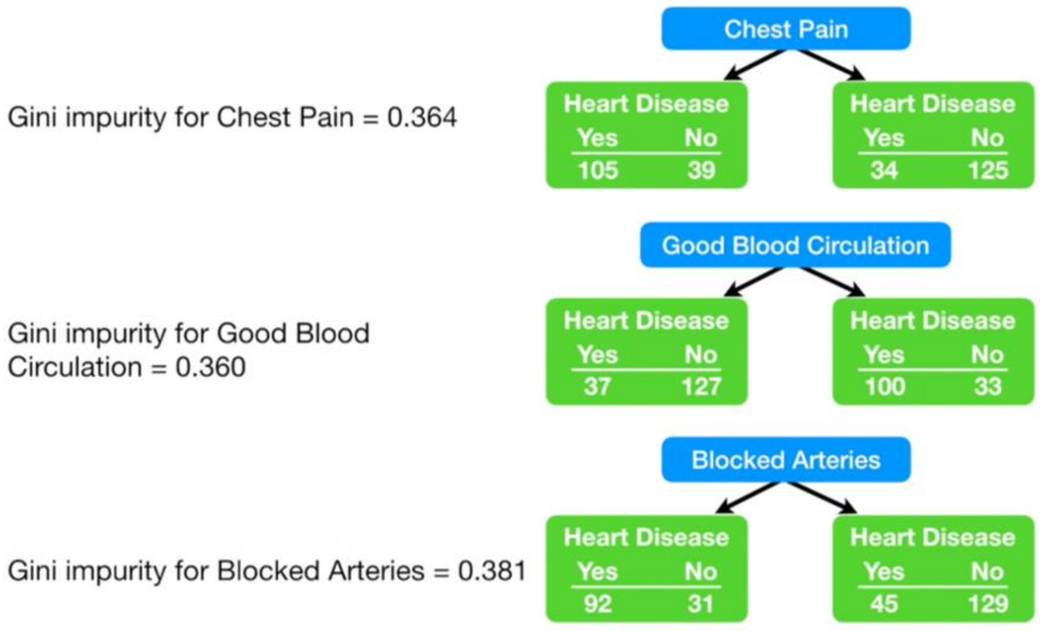


Figure 5 Gini Impurity for each of the attributes

We then select the attribute with the **least** Gini Impurity to be our root. It turns out it is good blood circulation.

We have now decided on the root node attribute, and we need to the same to each branch we have, using the yet unused attributes.

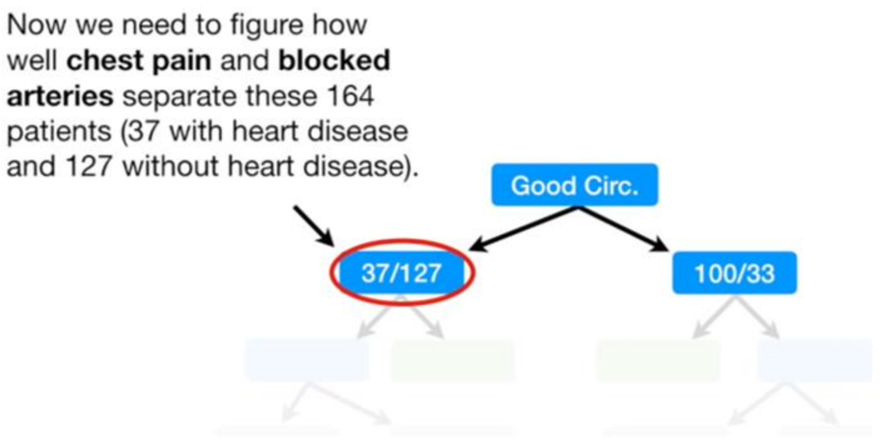

Now we need to figure how well **chest pain** and **blocked arteries** separate these 164 patients (37 with heart disease and 127 without heart disease).
Good Circ.
37/127
100/33

Figure 6 Shows the selected root node and how it has separated patients with and without heart disease

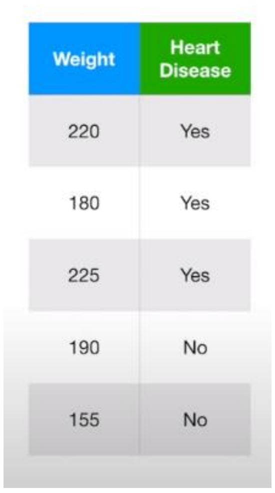
We keep repeating the same process repeatedly until we build the whole tree.

Steps:

1. Calculate all the Gini Impurity scores.
2. If the node itself has the lowest score, then there is no point in separating the patients anymore and it becomes a leaf node
3. If separating the data results in an improvement, then pick the separation with the lowest Gini Impurity Value

13.2 Numeric Data

So far, we have seen how to build a tree with yes/no (Boolean) attributes at each step. For numeric data, imagine we have the following attribute.

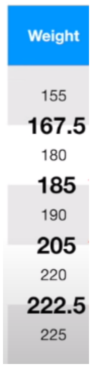


Weight	Heart Disease
220	Yes
180	Yes
225	Yes
190	No
155	No

Figure 7 Sample from the dataset showing only the attribute of weight

Steps:

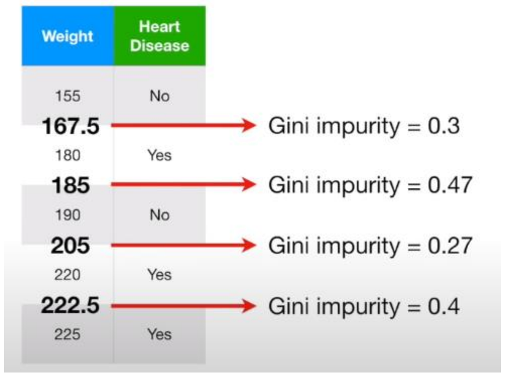
1- Sort the patients by weight from lowest to highest
2- Calculate the average weight of all adjacent patients.



Weight
155
167.5
180
185
190
205
220
222.5
225

Figure 8 Shows ordered weights and the calculated averages between each row

3- Calculate the Gini Impurity for each average weight .



Weight	Heart Disease	
155	No	
167.5		Gini impurity = 0.3
180	Yes	
185		Gini impurity = 0.47
190	No	
205		Gini impurity = 0.27
220	Yes	
222.5		Gini impurity = 0.4
225	Yes	

Figure 9 Shows the Gini Impurity for each average between rows

4- Pick the average with the lowest Gini Impurity to be the cutoff and use the Gini Impurity to compare the attribute weight against other attributes like chest pain or blocked arteries.

13.3 Ranked Data

For ranked data, it is a similar process to numeric data, except instead we calculate impurity scores for all the possible ranks. For example, if we have a rank attribute taking values 0, 1, 2, 3, and 4. We will calculate how good "rank less than or equal to one" separates data using Gini Impurity, then calculate how good "rank less than or equal to two" separates data using Gini Impurity, and so on until three.

The reason why we don't use "rank less than or equal to zero" and "rank less than or equal to four" is because the first is no data while the other is all data, which means they provide no separation at all.

13.4 Multiple Choice Data

For multiple-choice data, we calculate Gini Impurity for each choice as well as each possible combination. For example, if our choices are blue, green, and red, we will calculate how each one of the colors separates the data as well as "Blue or Green", "Blue or Red", and "Red or Green". Note that like ranked data, we will not measure how good "Blue or Red or Green" separates the data since this will include everyone.

[6]

14 Example of Using a Decision Tree

For the following tree, left-branching is **Yes,** and right-branching is **No.**

Let us assume we have someone who is

- 45 Years Old
- His Resting blood rate is 120
- Does not eat doughnuts

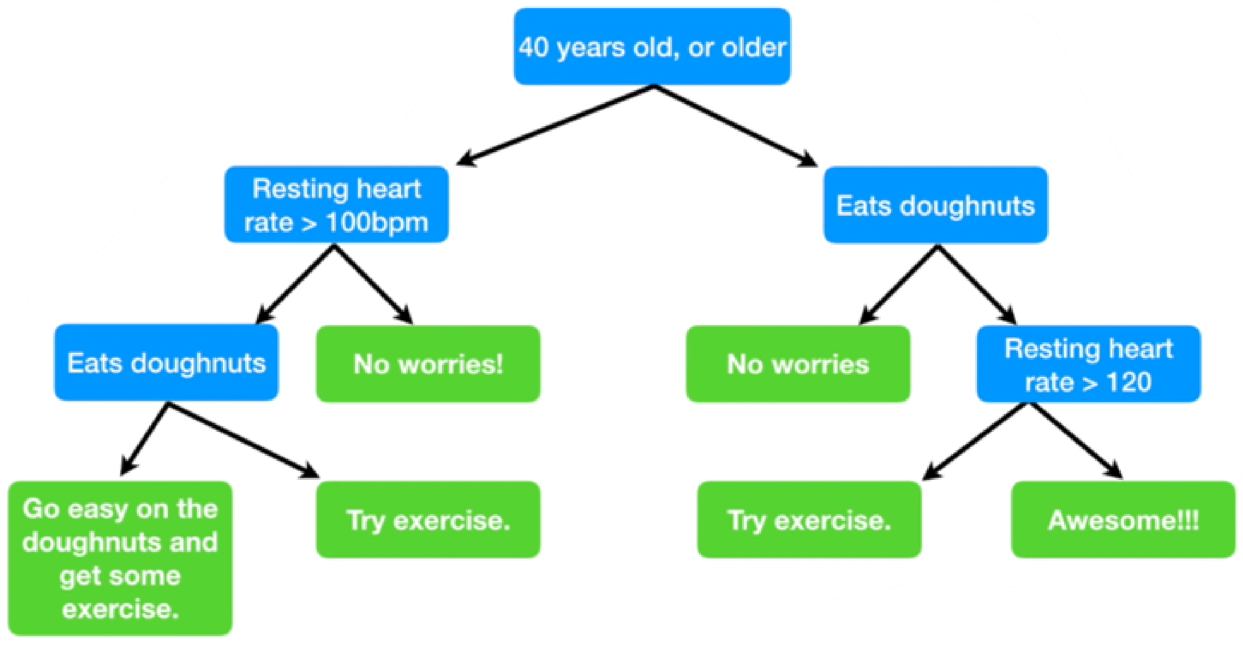


Figure 10 Shows a Decision Tree

We start navigating the decision tree from the root node. We see that our patient is more than 40 years old, we hence proceed to the left branch of the node. Next, we see that our patient's resting heart rate is more than 100 beats per minute which makes us proceed, again, to the left branch of the node. Lastly, our patient does not eat doughnuts which makes us reach the leaf node "Try Exercise" and by reaching a leaf node, we are done.

15 Practical Work Dataset Description

For the practical part of this research, we will be using a dataset obtained from the UC Irvine Machine Learning Repository. The dataset is a Heart Disease data set with 13 attributes and a "goal" attribute which specifies if a patient has more or less than 50% diameter narrowing implying whether the patient has/is prone to heart disease.

The description of the attributes in the dataset is as follows:

1. age: age in years
2. sex: sex (1 = male; 0 = female)
3. cp: chest pain type
   -- Value 1: typical angina
   -- Value 2: atypical angina
   -- Value 3: non-anginal pain
   -- Value 4: asymptomatic
4. trestbps: resting blood pressure (in mm Hg on admission to the hospital)
5. chol: serum cholesterol in mg/dl
6. fbs: (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7. restecg: resting electrocardiographic results
   -- Value 0: normal
   -- Value 1: having ST-T wave abnormality (T wave inversions and/or ST
   elevation or depression of > 0.05 mV)
   -- Value 2: showing probable or definite left ventricular hypertrophy
   by Estes' criteria
8. thalach: maximum heart rate achieved
9. exang: exercise-induced angina (1 = yes; 0 = no)
10. oldpeak: ST depression induced by exercise relative to rest
11. slope: the slope of the peak exercise ST segment
    -- Value 1: upsloping
    -- Value 2: flat
    -- Value 3: down sloping
12. ca: number of major vessels (0-3) colored by fluoroscopy
13. thal: thallium stress test results, 3 = normal; 6 = fixed defect; 7 = reversible defect
14. goal: diagnosis of heart disease (Values from 0 to 4) (angiographic disease status)
    -- Value 0: < 50% diameter narrowing
    -- Value 1: > 50% diameter narrowing

The Link to the dataset webpage containing all the details is found in the appendix.

16 Dataset Exploratory Analysis using Python

16.1 Introduction

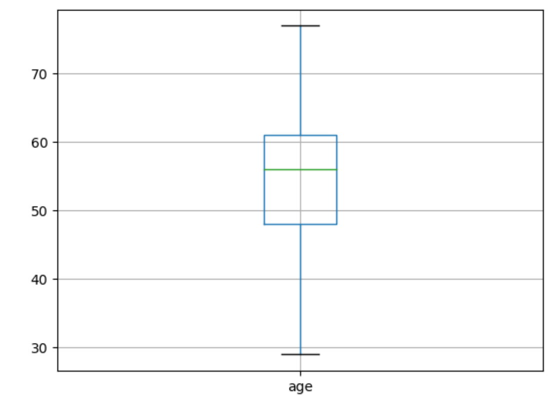
Python libraries Pandas, Matplotlib's pyplot, and NumPy were for data processing and analysis during the exploratory analysis process. The link to the code is found in the Appendix section of this paper. The filename is "data_analysis.py".

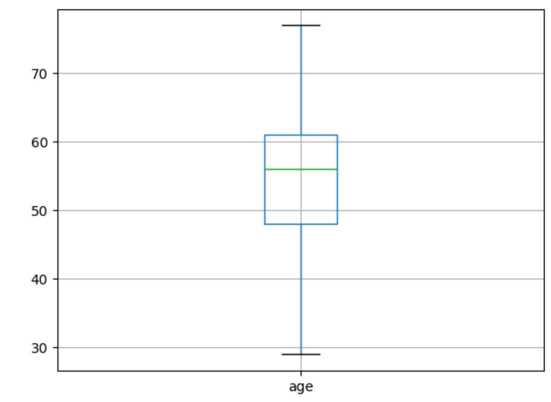
16.2 Removing Outliers

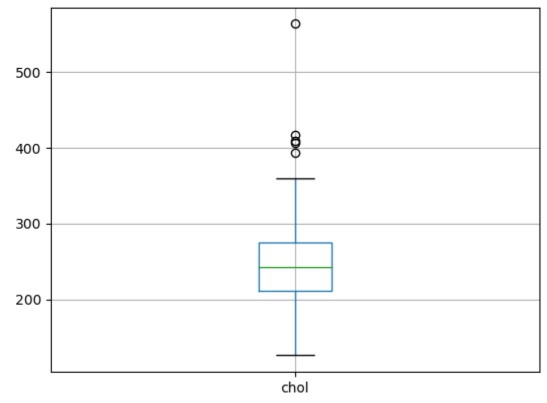
Before starting any analysis, box plots were used for visual outlier analysis of the data. Following this, data with numerical attributes not within plus or minus three standard deviations from the mean were considered outliers and were programmatically removed. The dataset contained 288 rows instead of 297 rows after outliers were removed. The following figure shows box plots of data before and after outliers were removed.

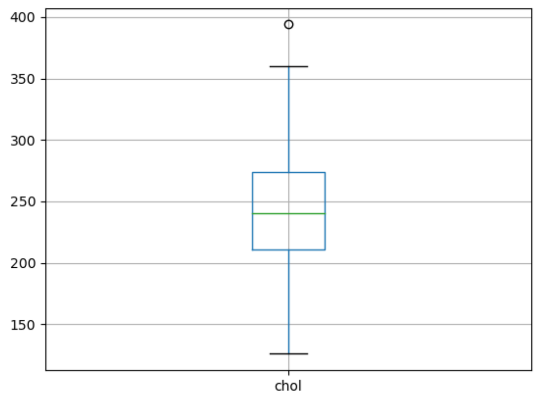
Before

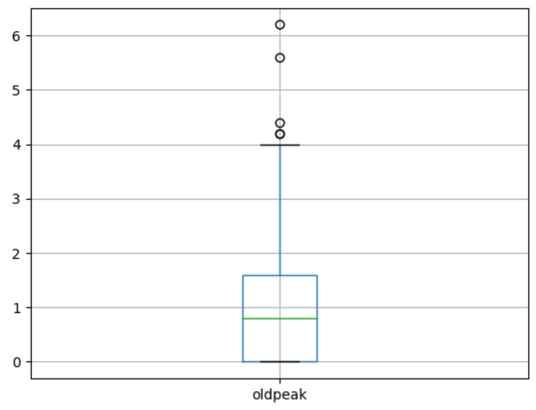
After

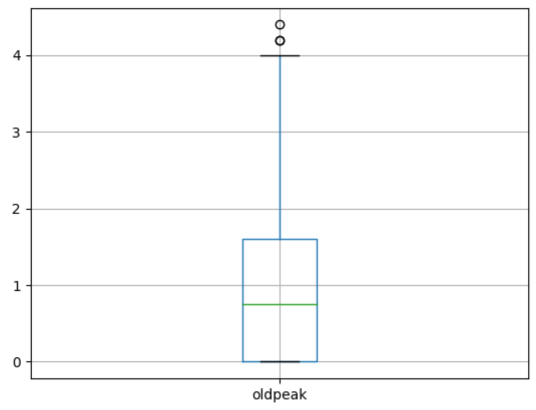












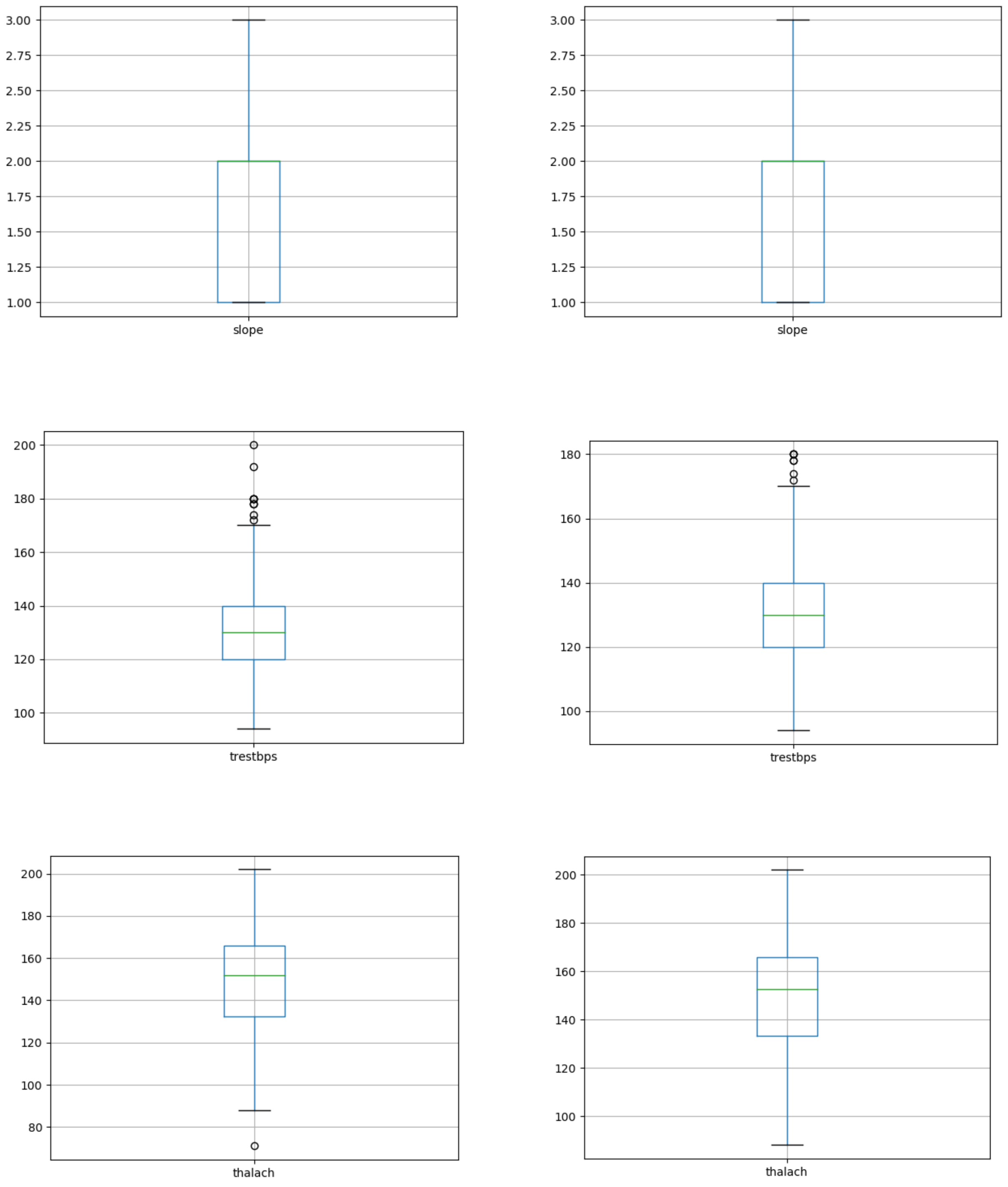
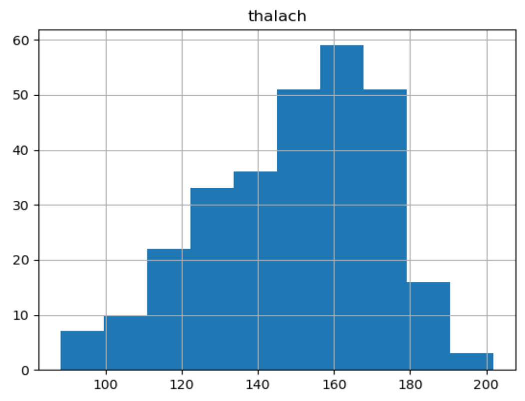
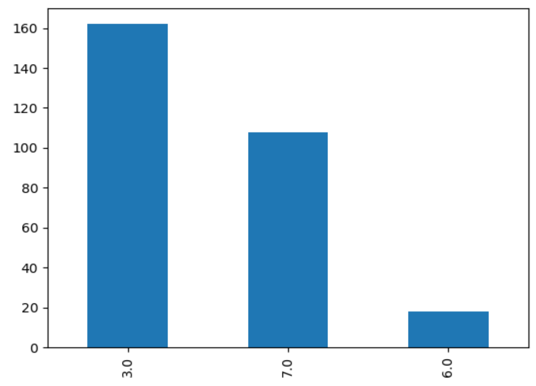


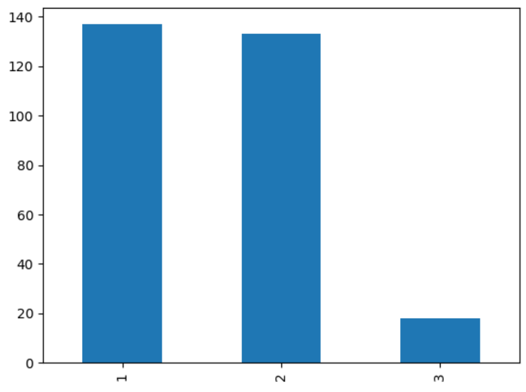
Figure 11 Shows box plots before and after outliers removal

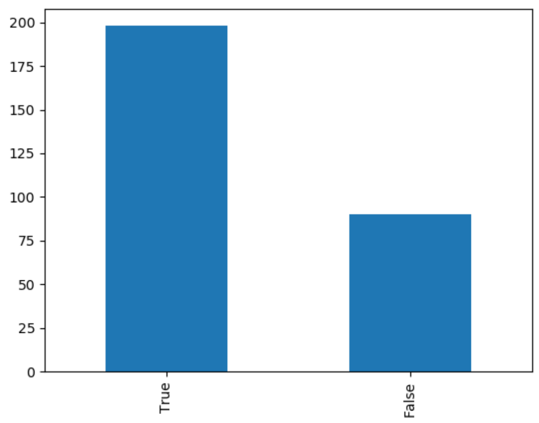
16.3 Histograms and Bar charts

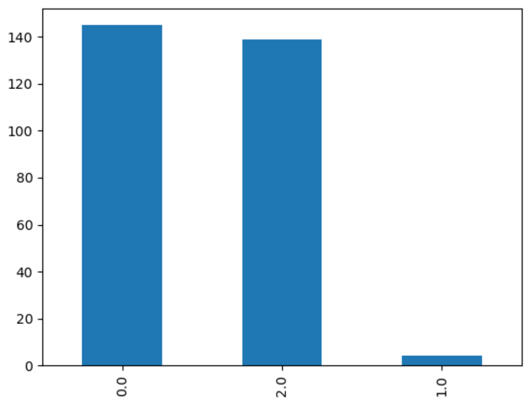
To visualize the nature of the data, histograms or bar charts were plotted depending on the nature of the given attribute.

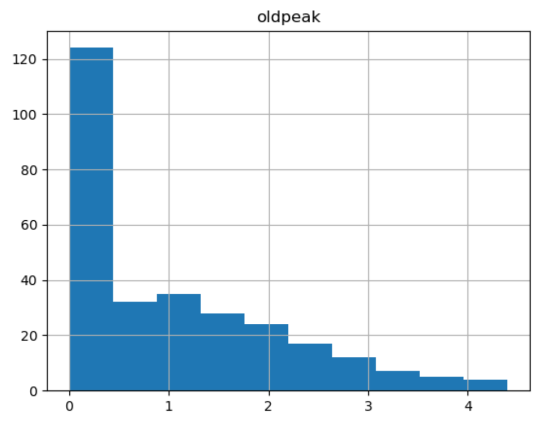


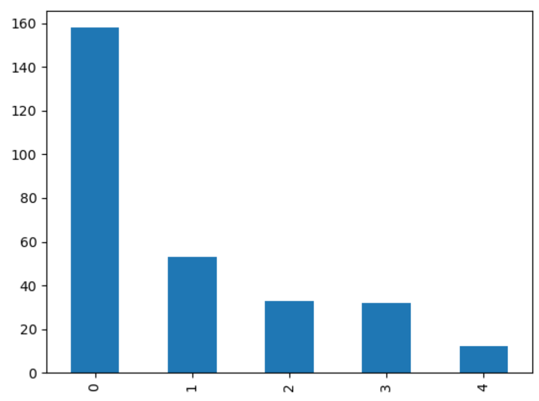


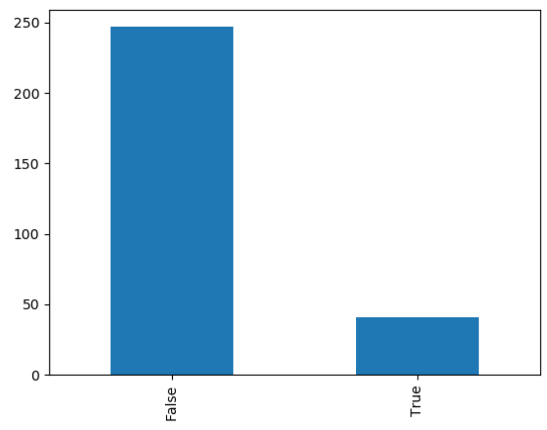












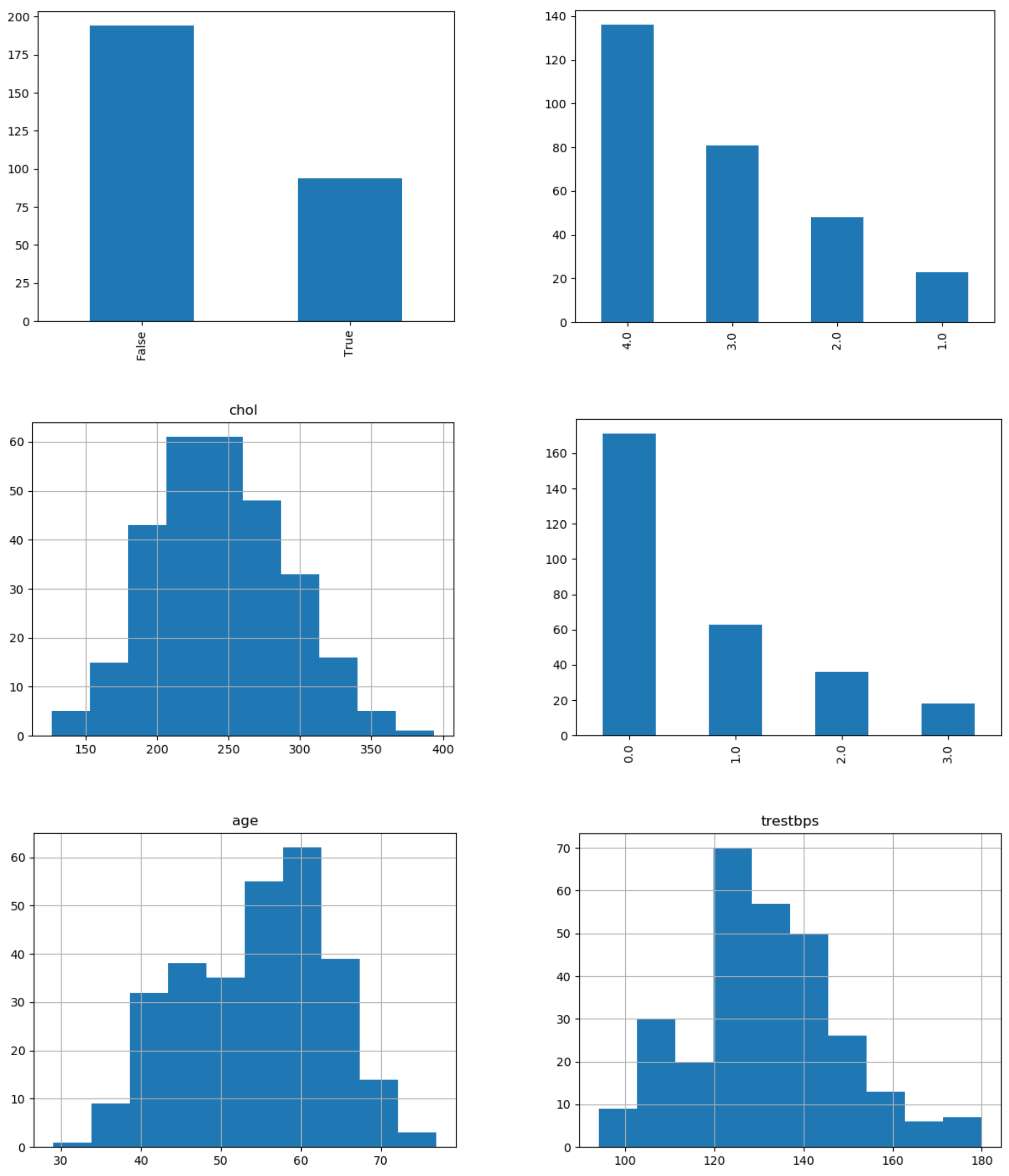
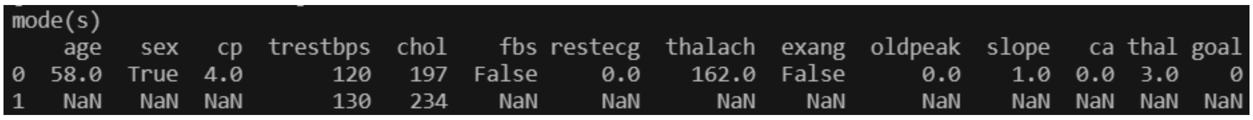


Figure 12 Shows histograms and bar charts of attributes of the dataset

16.4 Statistical Data Analysis

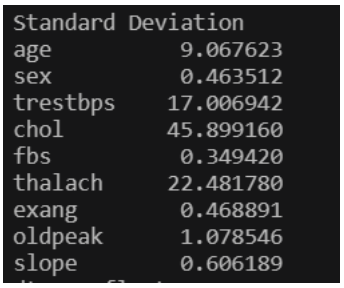
16.4.1 Data Modes



```
mode(s)
    age   sex   cp  trestbps  chol    fbs restecg  thalach  exang  oldpeak  slope   ca thal goal
0  58.0  True  4.0       120   197  False     0.0    162.0  False      0.0    1.0  0.0  3.0    0
1   NaN   NaN  NaN       130   234    NaN     NaN      NaN    NaN      NaN    NaN  NaN  NaN  NaN
```

Figure 13 Modes of Attributes (NaN Values indicate there is not two modes for the given attribute)

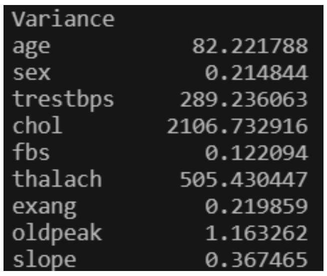
16.4.2 Standard Deviations



```
Standard Deviation
age          9.067623
sex          0.463512
trestbps    17.006942
chol        45.899160
fbs          0.349420
thalach     22.481780
exang        0.468891
oldpeak      1.078546
slope        0.606189
```

Figure 14 Standard deviations of dataset attributes

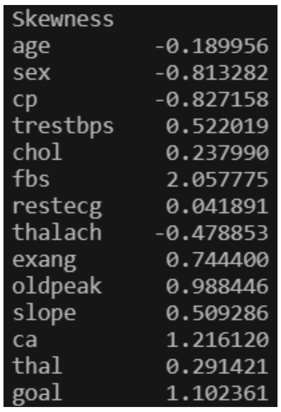
16.4.3 Variance



```
Variance
age           82.221788
sex            0.214844
trestbps     289.236063
chol        2106.732916
fbs            0.122094
thalach      505.430447
exang          0.219859
oldpeak        1.163262
slope          0.367465
```

Figure 15 Variance of dataset attributes

16.4.4 Skewness of Data



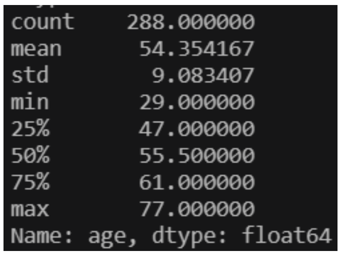
```
Skewness
age        -0.189956
sex        -0.813282
cp         -0.827158
trestbps    0.522019
chol        0.237990
fbs         2.057775
restecg     0.041891
thalach    -0.478853
exang       0.744400
oldpeak     0.988446
slope       0.509286
ca          1.216120
thal        0.291421
goal        1.102361
```

Figure 16 Skewness of dataset attributes

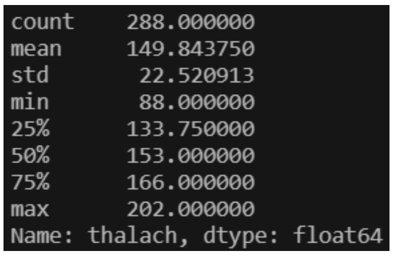
16.4.5 Description of Attributes

For numerical attributes, the count, mean, standard deviation, minimum, maximum, the 25th percentile, the 50th percentile, and the 75th percentile are shown.

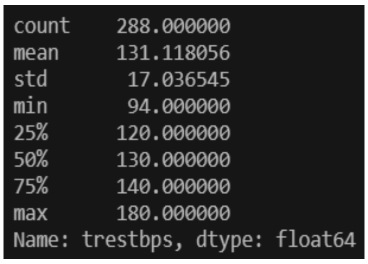
For nominal attributes, the count, the number of different (Unique) values, the most frequent (top) value, and the frequency of the most frequent (top) value are shown.



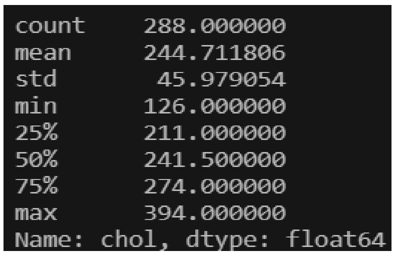
```
count    288.000000
mean      54.354167
std        9.083407
min       29.000000
25%       47.000000
50%       55.500000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```



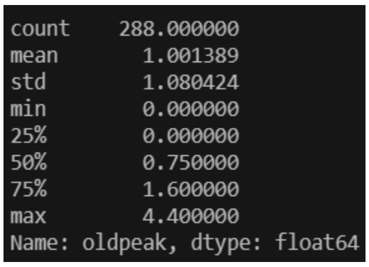
```
count    288.000000
mean     149.843750
std       22.520913
min       88.000000
25%      133.750000
50%      153.000000
75%      166.000000
max      202.000000
Name: thalach, dtype: float64
```



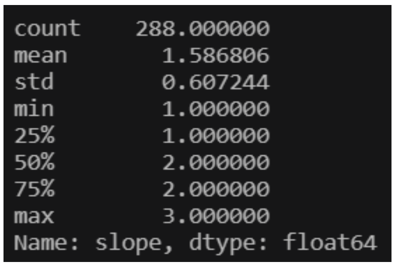
```
count    288.000000
mean     131.118056
std       17.036545
min       94.000000
25%      120.000000
50%      130.000000
75%      140.000000
max      180.000000
Name: trestbps, dtype: float64
```



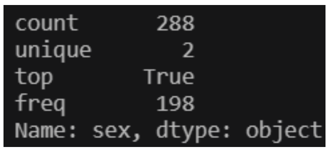
```
count    288.000000
mean     244.711806
std       45.979054
min      126.000000
25%      211.000000
50%      241.500000
75%      274.000000
max      394.000000
Name: chol, dtype: float64
```



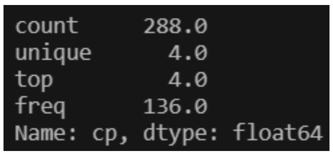
```
count    288.000000
mean       1.001389
std        1.080424
min        0.000000
25%        0.000000
50%        0.750000
75%        1.600000
max        4.400000
Name: oldpeak, dtype: float64
```



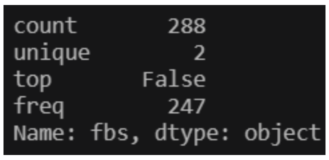
```
count    288.000000
mean       1.586806
std        0.607244
min        1.000000
25%        1.000000
50%        2.000000
75%        2.000000
max        3.000000
Name: slope, dtype: float64
```



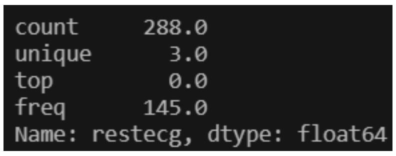
```
count      288
unique       2
top       True
freq       198
Name: sex, dtype: object
```



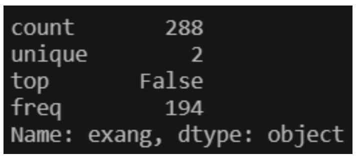
```
count     288.0
unique      4.0
top         4.0
freq      136.0
Name: cp, dtype: float64
```



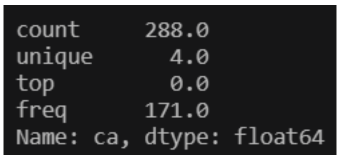
```
count      288
unique       2
top      False
freq       247
Name: fbs, dtype: object
```



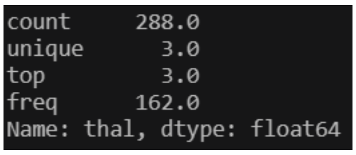
```
count     288.0
unique      3.0
top         0.0
freq      145.0
Name: restecg, dtype: float64
```



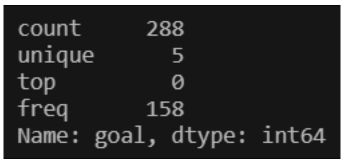
```
count      288
unique       2
top      False
freq       194
Name: exang, dtype: object
```



```
count     288.0
unique      4.0
top         0.0
freq      171.0
Name: ca, dtype: float64
```



```
count     288.0
unique      3.0
top         3.0
freq      162.0
Name: thal, dtype: float64
```



```
count     288
unique      5
top         0
freq      158
Name: goal, dtype: int64
```

Figure 17 Description of dataset attributes

17 Dataset Weka Operations

17.1 Visualization of Data

The following visualization was produced by Weka's visualization tool. It shows the distribution of data for all attributes.

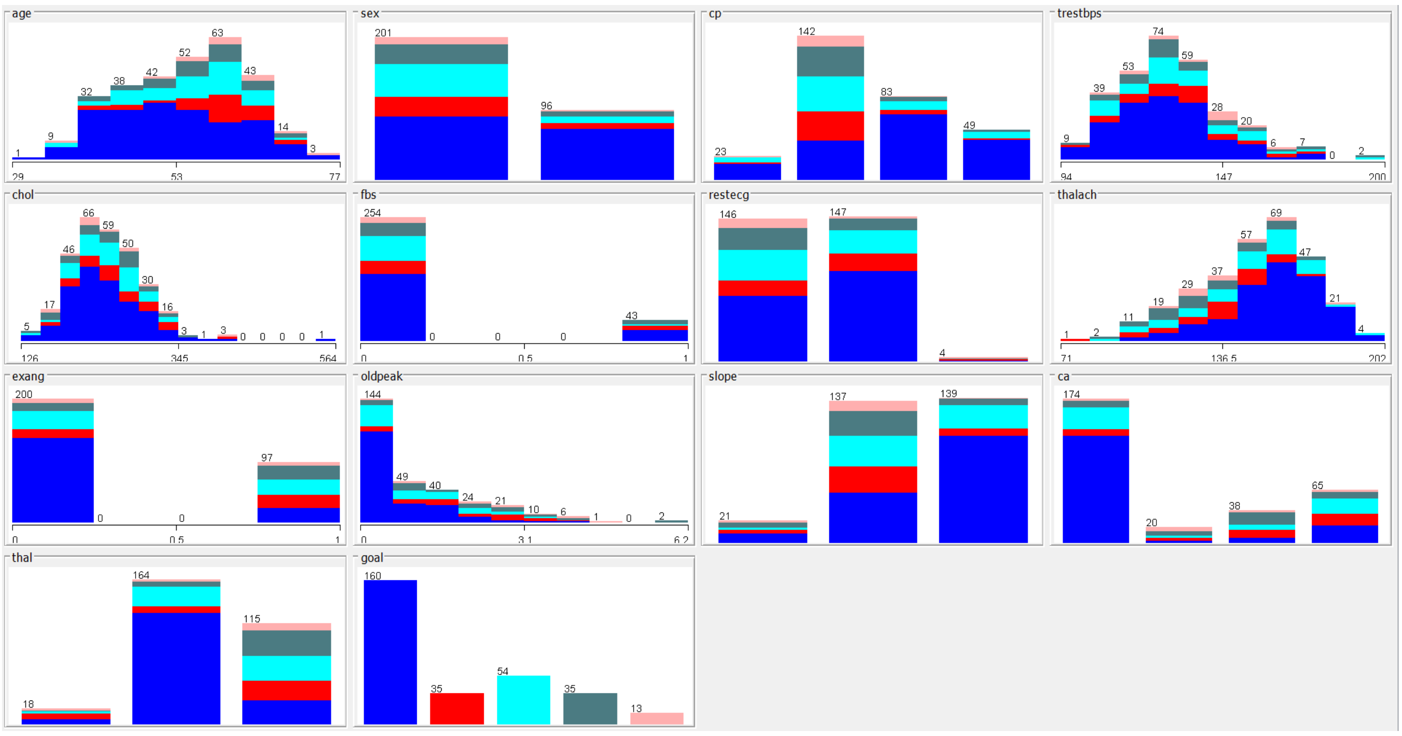


Figure 18 Weka's visualization of the dataset attributes

17.2 Outlier Analysis Using DBSCAN

17.2.1 Parameters

DBSCAN's Weka implementation was run using

- Euclidian distance as the distance function
- Epsilon = 0.9
- minimum Points = 6

17.2.2 Results

Clustered Instances

Cluster Number	Members Count	Percentage
0	21	9%
1	49	21%
2	58	25%
3	26	11%
4	6	3%
5	30	13%
6	33	14%
7	7	3%

Table 17-1 Shows Weka's DBSCAN clustering results

Unclustered instances = 67

The full output of the operation is included in the git repository hosted on GitHub linked in the Appendix. The filename is “DBSCAN.out”.

17.3 Reprocessing Data for K-Means and REPTree Decision Tree builder

For us to make Weka understand that not all classes were numerical, we had to write python code to map numbers to strings. The code is included in the git repository hosted on GitHub linked in the Appendix. The filename is “weka_preprocessing.py”

The mapping was as follows

- Sex
    - 0: 'F'
    - 1: 'M'
- cp
    - 1: 'typical angina'
    - 2: 'atypical angina'
    - 3: 'non-anginal pain'
    - 4: 'asymptomatic'
- restecg
    - 0: 'normal'
    - 1: 'STT wave abnormality'
    - 2: 'left ventricular hypertrophy'
- slope
    - 1: 'up'
    - 2: 'flat'
    - 3: 'down'
- ca
    - 0: 'zero'
    - 1: 'one'
    - 2: 'two'
    - 3: 'three'
- thal
    - 3: "normal"
    - 6: "fixed defect"
    - 7: "reversible defect"
- goal
    - 0: 'A'
    - 1: 'B'
    - 2: 'C'
    - 3: 'D'
    - 4: 'E'

17.4 Clustering Data using K-Means

17.4.1 Parameters

K-Means’ Weka implementation was run using

- K (Number of Clusters) = 6
- Max Iterations = 1000
- Euclidian distance as the distance function

- Random Initialization

17.4.2 Results

Missing values globally replaced with mean/mode

Number of iterations: 7

Within cluster sum of squared errors: 662.6708187833327

Initial starting points (random):

Cluster 0: 66,F,'typical angina',150,226,0,normal,114,0,2.6,down,zero,normal,A

Cluster 1: 39,M,asymptomatic,118,219,0,normal,140,0,1.2,flat,zero,'reversable defect',D

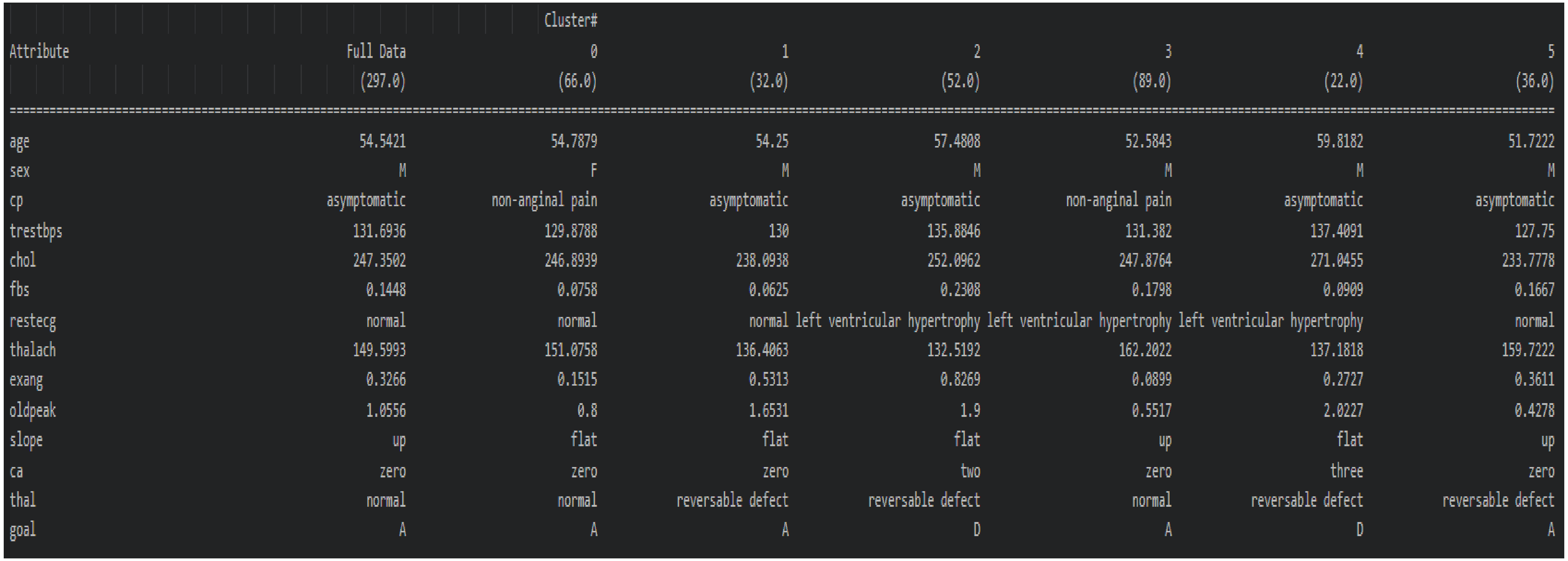
Cluster 2: 58,M,asymptomatic,128,259,0,'left ventricular hypertrophy',130,1,3,flat,two,'reversable defect',D

Cluster 3: 42,M,'typical angina',148,244,0,'left ventricular hypertrophy',178,0,0.8,up,two,normal,A

Cluster 4: 45,M,asymptomatic,142,309,0,'left ventricular hypertrophy',147,1,0,flat,three,'reversable defect',D

Cluster 5: 40,M,asymptomatic,152,223,0,normal,181,0,0,up,zero,'reversable defect',B

Final cluster centroids



```
                                            Cluster#
Attribute                 Full Data                0                1                2                3                4                5
                            (297.0)           (66.0)           (32.0)           (52.0)           (89.0)           (22.0)           (36.0)
================================================================================================================================================
age                         54.5421          54.7879            54.25          57.4808          52.5843          59.8182          51.7222
sex                               M                F                M                M                M                M                M
cp                     asymptomatic  non-anginal pain     asymptomatic     asymptomatic  non-anginal pain     asymptomatic     asymptomatic
trestbps                   131.6936         129.8788              130         135.8846          131.382         137.4091           127.75
chol                       247.3502         246.8939         238.0938         252.0962         247.8764         271.0455         233.7778
fbs                          0.1448           0.0758           0.0625           0.2308           0.1798           0.0909           0.1667
restecg                      normal           normal           normal left ventricular hypertrophy left ventricular hypertrophy left ventricular hypertrophy           normal
thalach                    149.5993         151.0758         136.4063         132.5192         162.2022         137.1818         159.7222
exang                        0.3266           0.1515           0.5313           0.8269           0.0899           0.2727           0.3611
oldpeak                      1.0556              0.8           1.6531              1.9           0.5517           2.0227           0.4278
slope                            up             flat             flat             flat               up             flat               up
ca                             zero             zero             zero              two             zero            three             zero
thal                         normal           normal reversable defect reversable defect         normal reversable defect reversable defect
goal                              A                A                A                D                A                D                A
```

Figure 19 K-means final clustering centroids

Clustered Instances

Cluster Number	Members Count	Percentage
0	66	22%
1	32	11%
2	52	18%
3	89	30%
4	22	7%
5	36	12%

Table 17-2 Shows Weka's K-means clustering results

The full output of the operation is included in the git repository hosted on GitHub linked in the Appendix. The filename is "K_Means.out".

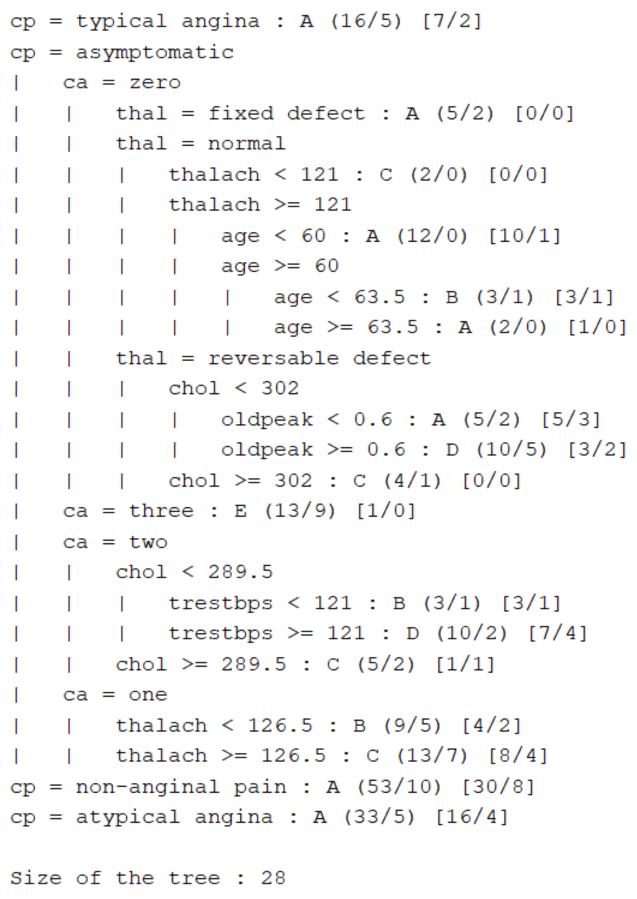
17.5 Building a Decision Tree with REPTree

17.5.1 Parameters

- Batch Size is 300
- No Maximum Depth
- Using Pruning

17.5.2 Results

The resultant decision tree.

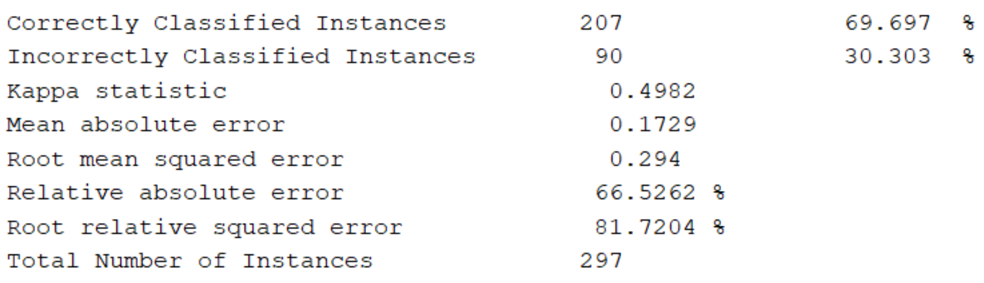


```
cp = typical angina : A (16/5) [7/2]
cp = asymptomatic
|   ca = zero
|   |   thal = fixed defect : A (5/2) [0/0]
|   |   thal = normal
|   |   |   thalach < 121 : C (2/0) [0/0]
|   |   |   thalach >= 121
|   |   |   |   age < 60 : A (12/0) [10/1]
|   |   |   |   age >= 60
|   |   |   |   |   age < 63.5 : B (3/1) [3/1]
|   |   |   |   |   age >= 63.5 : A (2/0) [1/0]
|   |   thal = reversable defect
|   |   |   chol < 302
|   |   |   |   oldpeak < 0.6 : A (5/2) [5/3]
|   |   |   |   oldpeak >= 0.6 : D (10/5) [3/2]
|   |   |   chol >= 302 : C (4/1) [0/0]
|   ca = three : E (13/9) [1/0]
|   ca = two
|   |   chol < 289.5
|   |   |   trestbps < 121 : B (3/1) [3/1]
|   |   |   trestbps >= 121 : D (10/2) [7/4]
|   |   chol >= 289.5 : C (5/2) [1/1]
|   ca = one
|   |   thalach < 126.5 : B (9/5) [4/2]
|   |   thalach >= 126.5 : C (13/7) [8/4]
cp = non-anginal pain : A (53/10) [30/8]
cp = atypical angina : A (33/5) [16/4]

Size of the tree : 28
```

Figure 20 Resultant decision tree

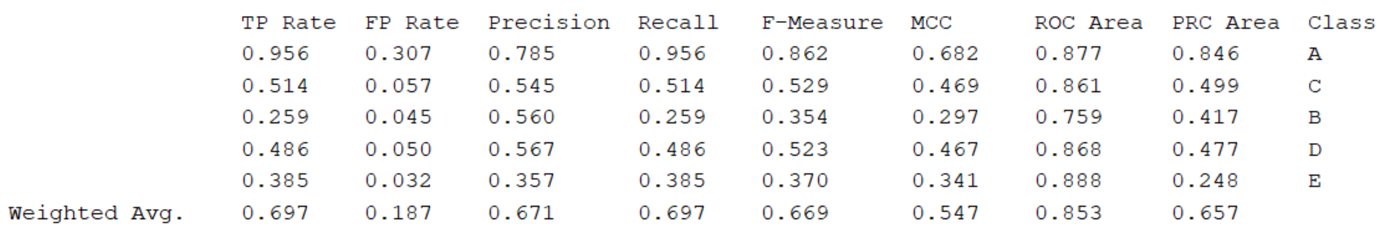
Performance Evaluation Summary



```
Correctly Classified Instances         207               69.697  %
Incorrectly Classified Instances        90               30.303  %
Kappa statistic                          0.4982
Mean absolute error                      0.1729
Root mean squared error                  0.294
Relative absolute error                 66.5262 %
Root relative squared error             81.7204 %
Total Number of Instances              297
```

Figure 21 Performance evaluation summary

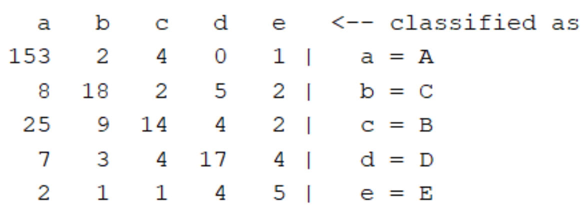
Detailed Accuracy by Class



```
                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.956    0.307    0.785      0.956    0.862      0.682    0.877     0.846     A
                 0.514    0.057    0.545      0.514    0.529      0.469    0.861     0.499     C
                 0.259    0.045    0.560      0.259    0.354      0.297    0.759     0.417     B
                 0.486    0.050    0.567      0.486    0.523      0.467    0.868     0.477     D
                 0.385    0.032    0.357      0.385    0.370      0.341    0.888     0.248     E
Weighted Avg.    0.697    0.187    0.671      0.697    0.669      0.547    0.853     0.657
```

Figure 22 Shows detailed accuracy for each cluster

Confusion Matrix Visually describing detailed accuracy by class



```
   a   b   c   d   e   <-- classified as
 153   2   4   0   1 |   a = A
   8  18   2   5   2 |   b = C
  25   9  14   4   2 |   c = B
   7   3   4  17   4 |   d = D
   2   1   1   4   5 |   e = E
```

Figure 23 Shows the confusion matrix

The full output of the operation is included in the git repository hosted on GitHub linked in the Appendix. The filename is “REPTree(Decision Tree).out”.

18 Conclusion

We now know what decision trees are, why they are needed, how they solve the classification problem, and how they are built and used. Furthermore, to accomplish the objective of this paper, we approached explaining the ideas using both theoretical and practical means.

19 References

19.1 Papers

1. J. R. Quinlan (1996). Improved use of continuous attributes in c4.5. Journal of Artificial Intelligence Research, 4:77-90.
2. S.B. Kotsiantis (2007). Supervised Machine Learning: A Review of Classification Techniques. Informatica, 31:249-268.

19.2 Textbooks

3. Schurmann, J. (1996). Pattern Classification. John Wiley &amp; Sons Inc.
4. HAN, J., & KAMBER, M. (2001). Data mining: concepts and techniques, 3rd Edition. San Francisco, Morgan Kaufmann Publishers.
5. Ian H. Witten; Eibe Frank; Mark A. Hall (2011). Data Mining: Practical machine learning tools and techniques, 3rd Edition. Morgan Kaufmann, San Francisco. p. 191.

19.3 Articles

6. Josh Starmer (2018). Decision Trees. StatQuest.
7. Rajesh S. Brid (2018). Decision Trees A Simple Way to Visualize A Decision. Medium.
8. The Learning Machine Team (2018). Supervised Learning: Classification, Decision Tree with ID3 Algorithm. The Learning Machine.
9. Nikita Sharma (2020). Understanding the Mathematics Behind Decision Trees. Fritz Ai.
10. Yaser Sakkaf (2020). Decision Trees: ID3 Algorithm Explained. Medium.

20 Appendix

- Root GitHub Repository
  - https://github.com/M-Elsaeed/Data-Mining
- Dataset Source
  - https://archive.ics.uci.edu/ml/datasets/heart+Disease
- The Dataset (In the form that was used)
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/heart.csv
- The Weka-Special Modification of the Dataset (In the form that was used)
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/heart2.csv
- Python Data Analysis Code
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/code/data_analysis.py
- Weka's DBSCAN Output
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/DBSCAN.out
- Weka's K-Means Output
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/K_Means.out
- Weka's REPTree Output
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/REPTree(Decision%20Tree).out
- Weka Special Preprocessing Python Code
  - https://github.com/M-Elsaeed/Data-Mining/blob/master/weka/weka_preprocessing.py