VIETNAM GENERAL CONFEDERATION OF LABOUR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**INTRODUCTION TO MACHINE LEARNING**

**FINAL PROJECT**

**FINAL REPORT**

*Instructor*: **PROF. LE ANH CUONG**

*Student*: **TRAN HUY – 519K0054**

**LY DANG MINH – 519K0060**

**NGUYEN DINH KHOI – 519K0013**

Class **: 19K50201**

Course  **: 23**

**HO CHI MINH CITY, DECEMBER 2022**

VIETNAM GENERAL CONFEDERATION OF LABOUR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**INTRODUCTION TO MACHINE LEARNING**

**FINAL PROJECT**

**FINAL REPORT**

*Instructor*: **PROF. LE ANH CUONG**

*Student*: **TRAN HUY – 519K0054**

**Ly Dang Minh – 519K0060**

**Nguyen Dinh Minh Khoi – 519K0013**

Class **: 19K50201**

Course  **: 23**

**HO CHI MINH CITY, JANUARY 2022**

ACKNOWLEDGEMENTS

In order to complete this report, my group has received helpful assistances from lecturers and others.

We would like to give the most sincere thanks to our instructor, Prof. Cuong, for enthusiastically providing information on the essay and explaining it clearly, as well as ours families and friends for giving us valuable moral support.

THE REPORT WAS COMPLETED AT TON DUC THANG UNIVERSITY

We assure that this project was completed by ourselves with the instruction of Prof. Cuong. Research contents and results in this topic are honest and had not ever been published in any form.

**If any fraud is detected, we will take full responsibility for the content of our project.** Ton Duc Thang University is not related to any copyright or copyright infringement caused by me during the implementation process (if any).

*Ho Chi Minh city , date 10 month 1 year 2022*

*Author*

*(sign with full name)*

*Trần Huy*

*Nguyễn Đình Minh Khôi*

*Lý Đăng Minh*

LECTURER'S CONFIRMATION AND EVALUATION SECTION

**The confirmation of instructing lecturer**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, date month year

(sign with full name)

**The assessments of marking lecturers**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, date month year

(sign with full name)

TABLE OF CONTENTS

[ACKNOWLEDGEMENTS 1](#_Toc17284)

[THE REPORT WAS COMPLETED AT TON DUC THANG UNIVERSITY 2](#_Toc31599)

[LECTURER'S CONFIRMATION AND EVALUATION SECTION 3](#_Toc23035)

[TABLE OF CONTENTS 4](#_Toc22751)

QUESTION 1 [5](#_Toc25293)

1.1. Definition [5](#_Toc23094)

1.2. LSTM VS RNN  [6](#_Toc23409)

1.3. Apply an LSTM model to solve a time series problem [7](#_Toc16393)

QUESTION 2 [8](#_Toc15703)

2.1. CountVectorizer() from sklearn [8](#_Toc22615)

2.2. Tokenizer() And Sequence  [9](#_Toc21002)

2.3. SVM Model [9](#_Toc14990)

2.4. Logistic Regression  [10](#_Toc7969)

2.5. Random Forest  [12](#_Toc31290)

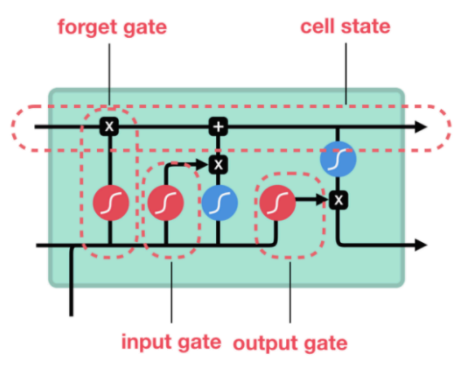
REFERENCES [13](#_Toc25828)

Question 1:

1.Definition

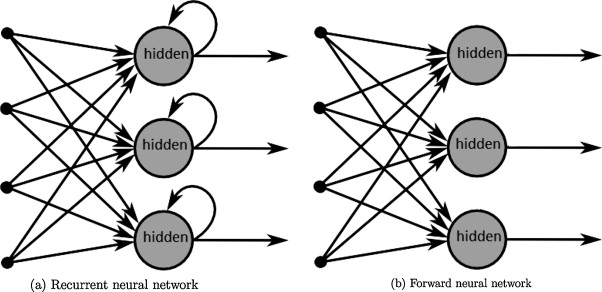
LSTM (Long Short-Term Memory) is a recurrent neural network (RNN) architecture capable of learning long-term dependencies. It is explicitly designed to avoid the long-term dependency problem.

LSTM networks were introduced by Hochreiter & Schmidhuber in 1997, and were refined and popularized by many people. They work extremely well on a large variety of problems, and are now widely used.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. 

Different from standard [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network), LSTM has feedback connections. It can process entire sequences of data (such as speech or video), instead of just single data points (such as images).

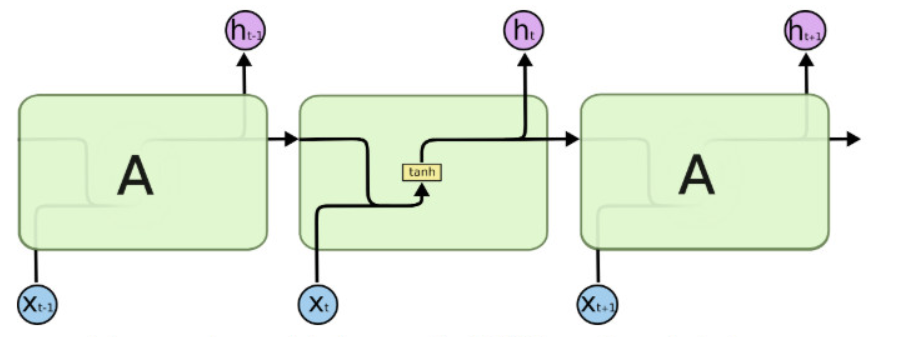
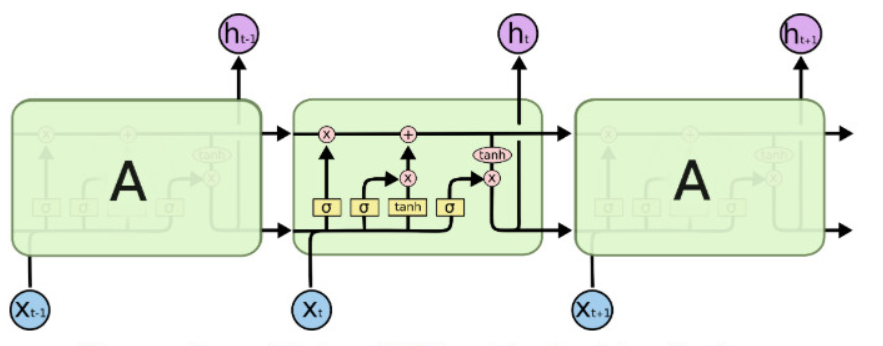
They are well-suited to [classifying](https://en.wikipedia.org/wiki/Classification_in_machine_learning), [processing](https://en.wikipedia.org/wiki/Computer_data_processing) and [making predictions](https://en.wikipedia.org/wiki/Predict) based on [time series](https://en.wikipedia.org/wiki/Time_series) data, since there can be lags of unknown duration between important events in a time series.



2. LSTM VS RNN

RNN (Recurrent Neural Networks) are the first kind of neural network algorithm that can memorize or remember the previous inputs in memory. However, it is difficult to train RNN that requires long-term memorization. This is because the gradient of the loss function decays exponentially with time, which is called the vanishing gradient problem:

* During back propagation, recurrent neural networks suffer from the vanishing gradient problem. Gradients are values used to update a neural networks weights.
* The vanishing gradient problem occurs when the gradient shrinks as it back propagates through time. If a gradient value becomes extremely small, it doesn’t contribute too much learning.

LSTM performs better in these kinds of datasets as it has more additional special units that can hold information longer. LSTMs include a 'memory cell' that can maintain information in memory for long periods of time. That memory cell provides the ability to learn longer-term dependencies.

Thanks to the gates, such as input and forget gates, unlike RNN, LSTMs can deal with vanishing and exploding gradient problem. This gives them a better control of gradient flow and enable better preservation of long-range dependencies.

3. Apply an LSTM model to solve a time series problem

3.1 Problem and dataset description

An alcoholic beverages selling agent wants to predict their future sales. Given a dataset that shows the number of alcohols sold through out the years, we need to predict future sales. Knowing that the sales varies based on the time of the year, we need to use an LSTM model to memorize the patterns and correctly predict the results.

3.2 Configurations description

Some LSTM model configurations used to solve the problem:

* Stacked LSTM: An LSTM model comprised of multiple LSTM layers. An LSTM layer above provides a sequence output rather than a single value output to the LSTM layer below. Specifically, one output per input time step, rather than one output time step for all input time steps.
* Bidirectional LSTM: A sequence processing model that consists of two LSTMs: one taking the input in a forward direction, and the other in a backwards direction. It connects two hidden layers of opposite directions to the same output.
* Vanilla LSTM: An LSTM model that has a single hidden layer of LSTM units, and an output layer used to make a prediction.

Question 2:

To explain the libraries that i used to filter spam:

1) CountVectorizer() from sklearn

This is used to convert a collection of text documents to a matrix of token counts.

If you do not provide an a-priori dictionary and you do not use an analyzer that does some kind of feature selection then the number of features will be equal to the vocabulary size found by analyzing the data.

For example:

- “Hello, my name is Khoi”

Then there will be a work count as

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 |
| Hello | my | name | is | Khoi |

Then if we transform other text, for xample: “Hi, John is the name”

Then it will return matrix where 1 is the text count

[ 0 0 1 0 1] first 1 is “is” and second 1 is “name” where we have the data that we fitted before

2) Tokenizer() And Sequence:

Tokenizer() class allows to vectorize a text corpus, by turning each text into either a sequence of integers (each integer being the index of a token in a dictionary) or into a vector where the coefficient for each token could be binary, based on word count, based on tf-idf...

It is very similar to CountVectorizer() but it will give you different format of out put when combined.

With sequence. Which lead to easy use in Keras model

For example:  
{“I love my dog”, “I love my cat” , “you love my dog”}

It will generate word index like this { ‘<OOV>’:1 , ‘my’:2, ‘love’:3, ‘dog’:4, ‘I’:5, ‘you’: 6, ‘cat’:7 }

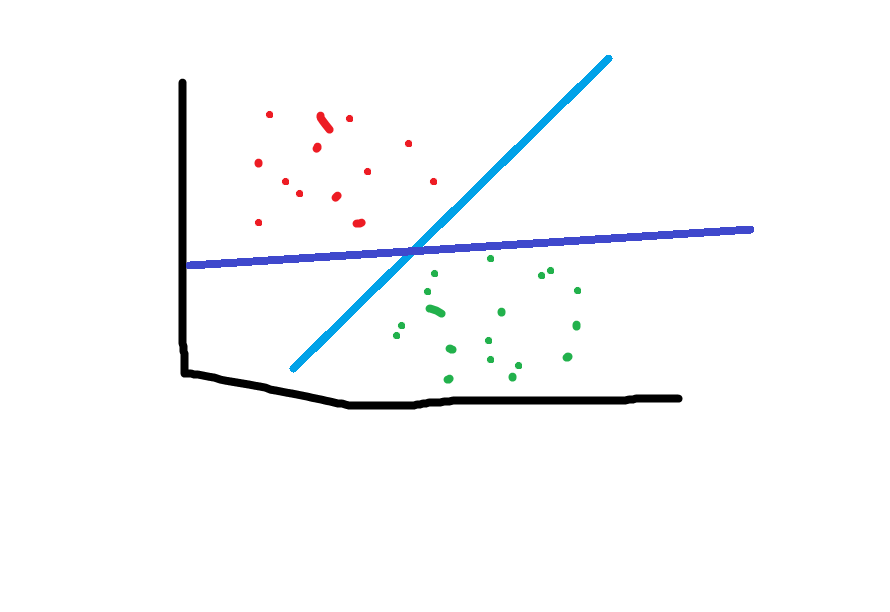
Then when give to sequence we receive output like

Sequence = {[5, 3, 2, 4], [5, 3, 2, 7], [6, 3, 2, 4]} where the number is relate to the word above

3) SVM Model

SVM for linearly separable binary sets

The goal is to design a hyperplane that classifies all training vectors into 2 classes.



Sorry for my bad drawing but as you can see here we have 2 hyperplan example and classifies correct.

The best choice is the hyperplane that leaves the maximum margin from both classes.

The margin is the closet between the hyperplane and the closet element.

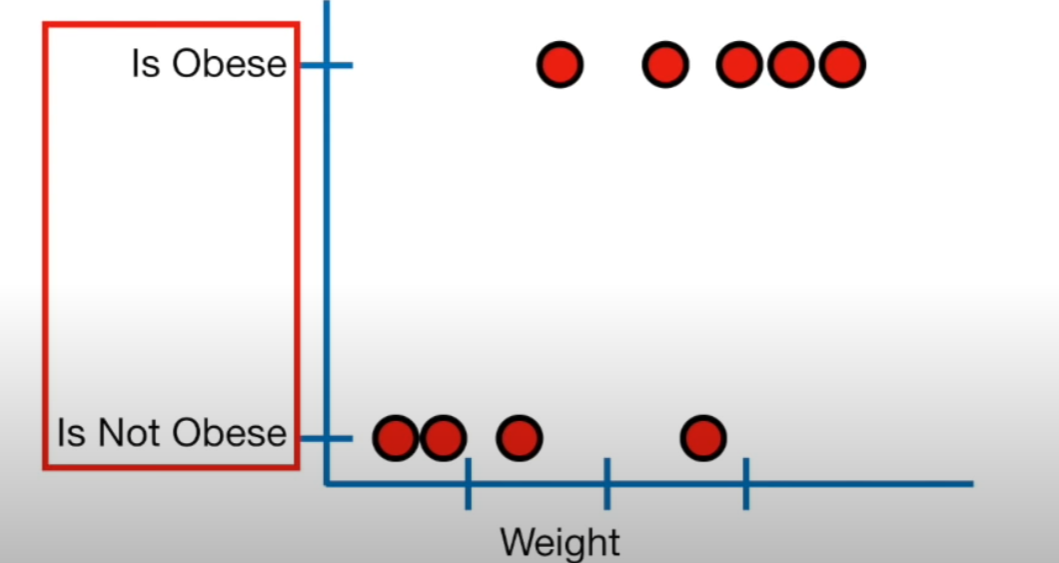
We can see the dark blue hyperplane is nearly hit the green element while the light blue stays a little bit farther, so in this case we get the light blue

So SVM will learn how to find the hyperplane that is having maximum margin. SVM will learn to maximize margin.

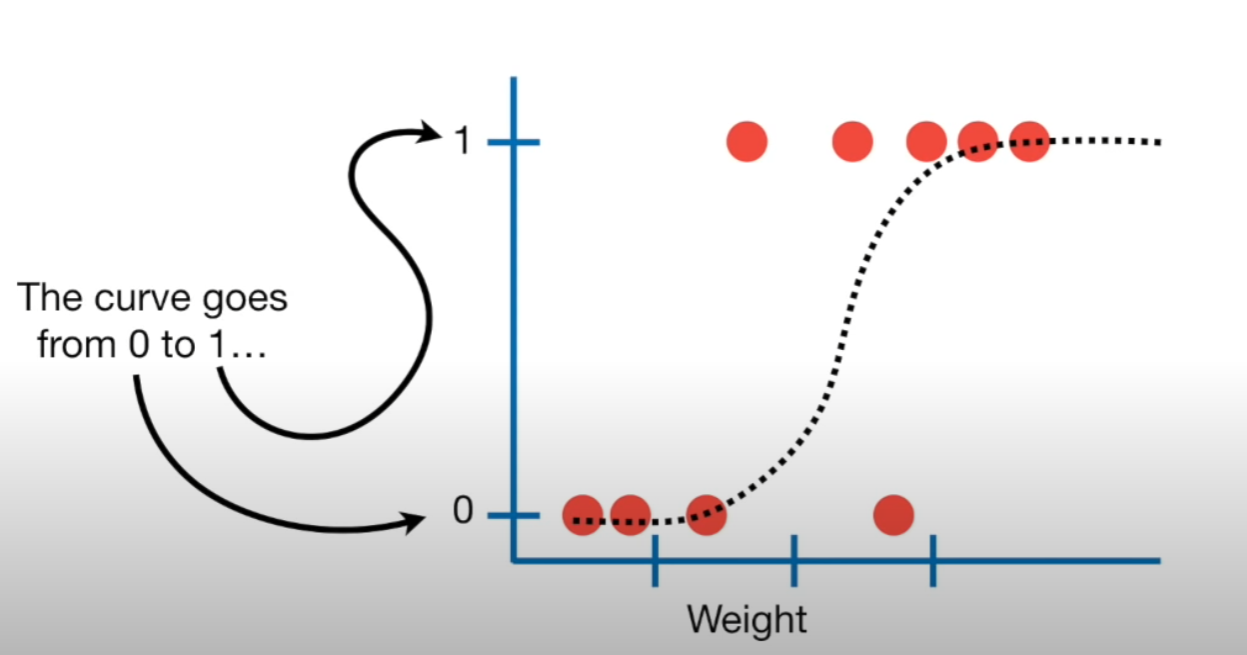
4) Logistic Regression

Logistic regression is very similar to linear regression but instead of predicting something continuous it predicts if something is True or False.

For example: about mice is obese or not obese:



And I draw a S shape to logistic function



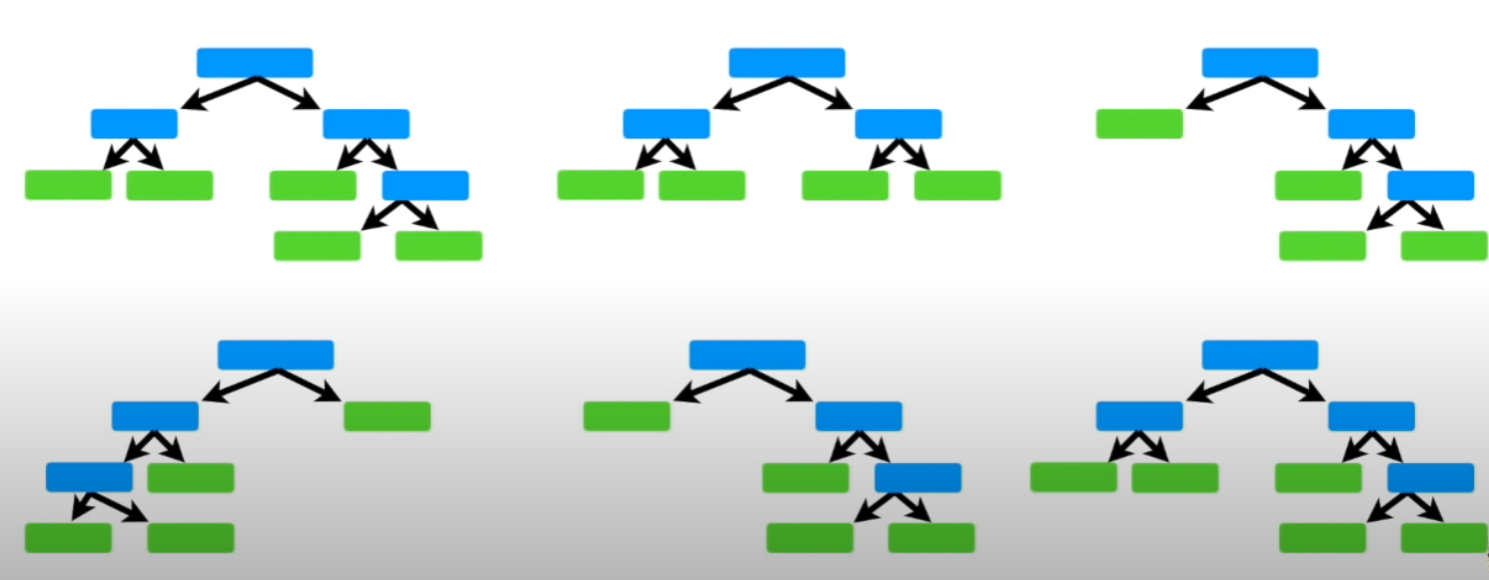
The curve will go from 0 to 1 and it presents the probability of something True or False, in the above example is the probability of a mice is obese or not obese base on Weight. Where 1 is obese.

If you weight a very heavy mouse, there’s a very high chance that it is obese due to the S. and 50% if it normal and very high chance that it is not obese with a light mouse.

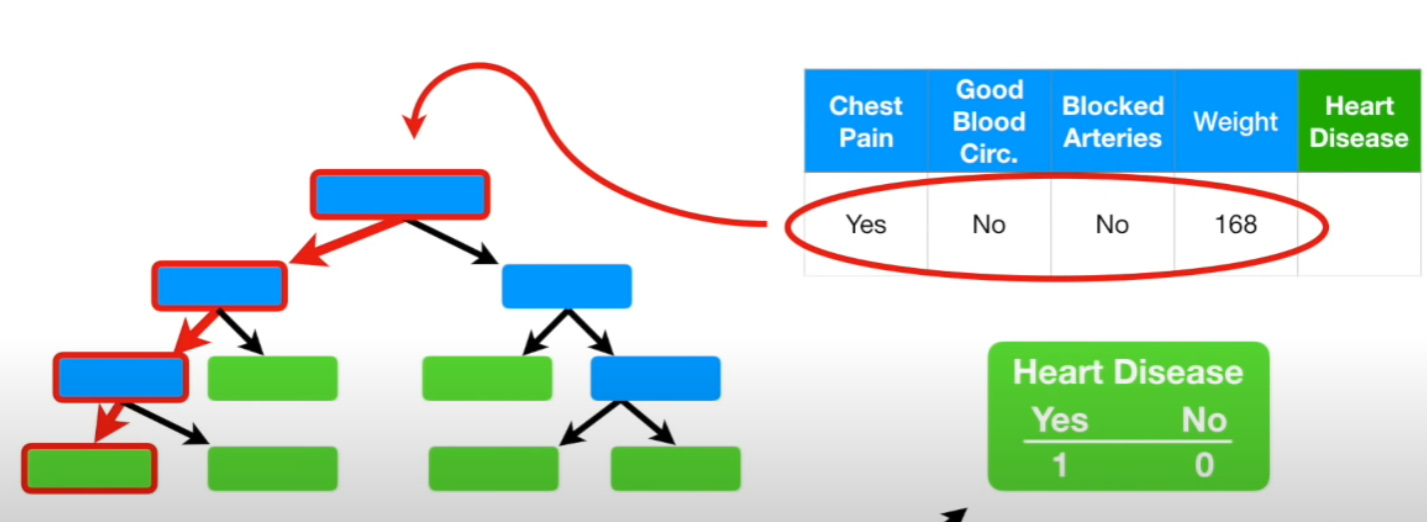
5) Random Forest:

Random forest is built very similar to decision tree. The reason why I use random forest is because decision tree is good with data used to create but no flexible when classify new sample.

Random forest will combine the simplicity of decision tree with flexibility resulting in a vast improvement in accuracy.



We create subsets of the variable at each step result in a wide variety of tree. So when new sample come



We will track the way variable go there for we can get the result.

REFERENCES

1. <https://www.youtube.com/watch?v=exHwwy9kVcg&ab_channel=CodeHeroku>
2. <https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/>
3. <https://ashutoshtripathi.com/2021/07/02/what-is-the-main-difference-between-rnn-and-lstm-nlp-rnn-vs-lstm/>
4. <https://www.kaggle.com/bulentsiyah/for-simple-exercises-time-series-forecasting?select=Alcohol_Sales.csv>
5. <https://en.wikipedia.org/wiki/Long_short-term_memory>