Name: **Iman Mousaei**
Course: **Data Science**
Assignment **4**

Student Number: **98222099**
Date: January 7, 2022

1 Part 1

1.1 Abstract

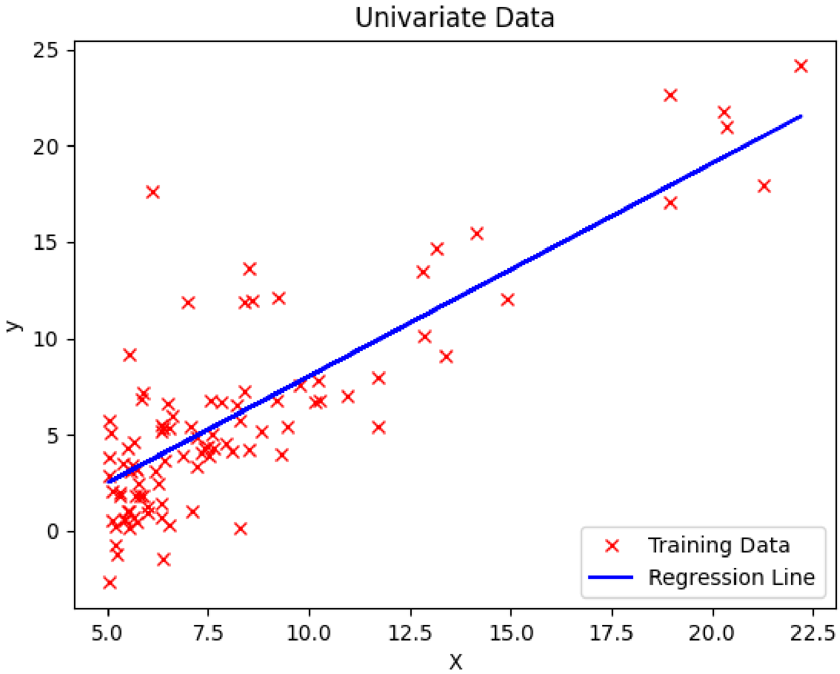
In this part, we want to predict a continuous variable using Linear Regression.

1.2 Introduction

We use Gradient Descent to fit a line to which, the cost function is minimal. Here, we used Mean Squared Error(MSE) as cost function.

1.3 Results

After fitting the line on the points of the dataset, we get this:



Which is a good line for prediction, but we could do better with polynomial functions.

2 Part 2

2.1 Abstract

In this part, we want to compare some models by classification using different evaluation methods.

2.2 Introduction

We are going to classify a constant set of training & test data. Then we are going to plot confusion matrix and calculate precision & recall & accuracy & F1 score for all of the models to compare them.
In this dataset labels are not balanced because almost all of labels are 0 and a few 1. So what we can do is generate rows with label 1. But here because we have a large dataset we can just downsample it. Ergo we select n rows with label=0 and n rows with label=1 and now its balanced. P.S. If you dont remember these evaluation methods:

$$F_1 = \frac{2TP}{2TP + FP + FN} \qquad Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

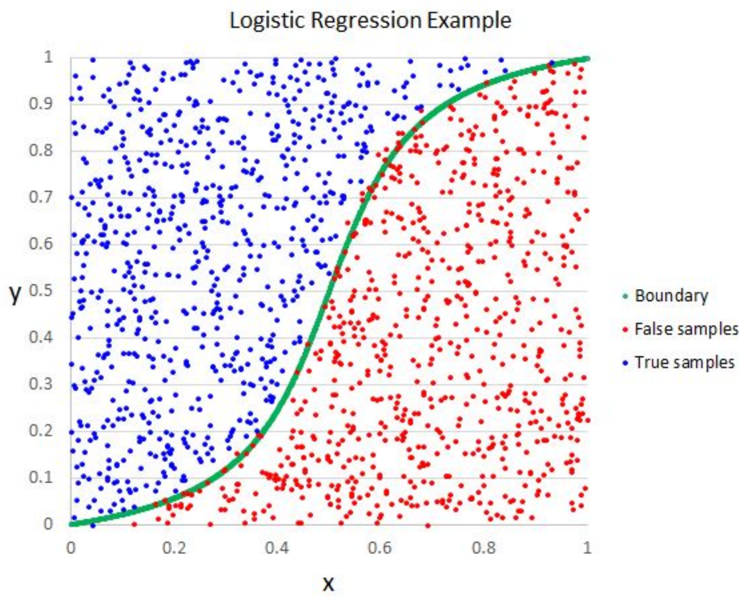
$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

Which in these formulas: **TP**=True Positive, **TN**=True Negative, **FP**=False Positive, **FN**=False Negative.

2.3 Classification Methods Explained

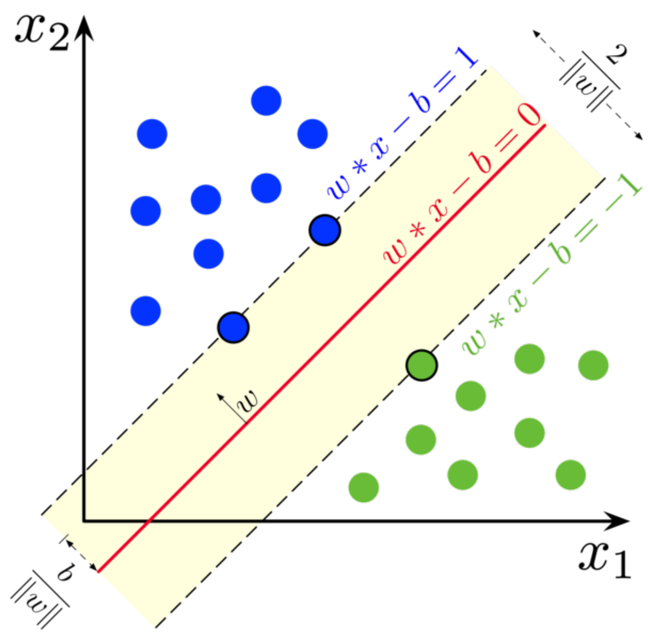
2.3.1 Logistic Regression

As shown in the figure, Logistic Regression fits a curve to split space into 2 parts. So the downside to this classifier is that it only supports 2 classes.



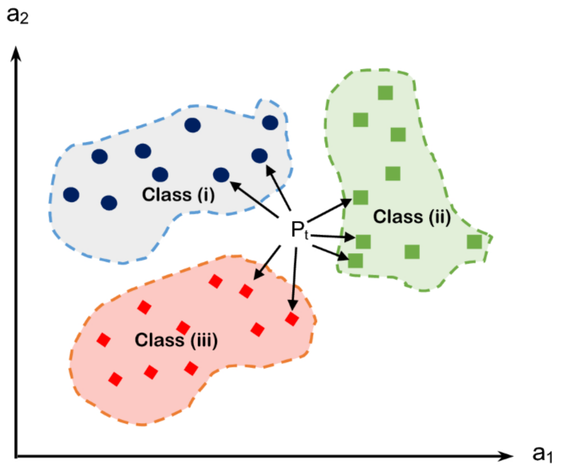
2.3.2 SVM

Support Vector Machine also separates space into 2 parts(so only supports 2 classes) but with a different algorithm. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training-data point of any class. Because in general the larger the margin, the lower the generalization error of the classifier.



2.3.3 KNN

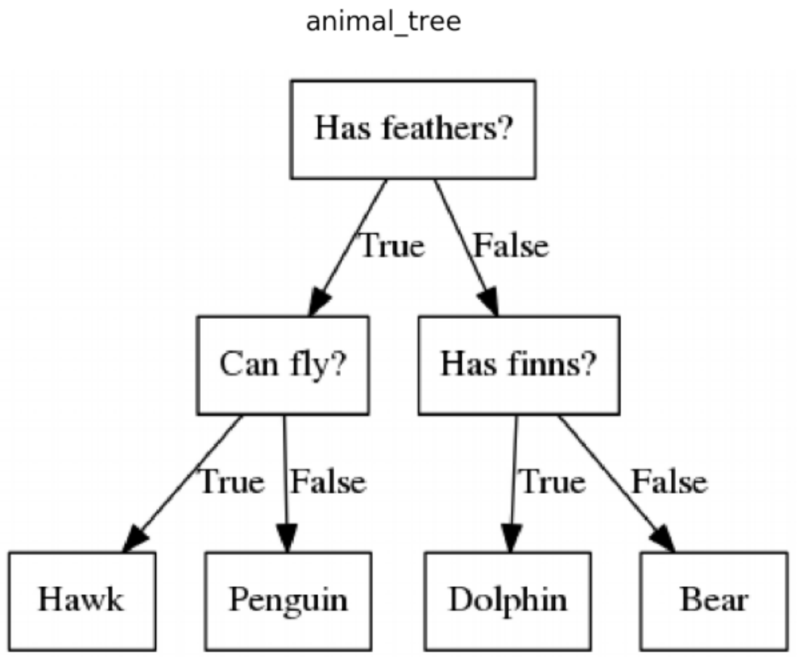
K Nearest Neighbor classifies every test data to class of the most of its K nearest training datas. Parameter K should be given in input, so we tried KNN for different K's and used the best one.



Example:
In this example we want to classify P_t. So we sort training set by distance and we select first K(here K=6). Since 3 of the 6 is from Class (ii) then we also classified P_t under Class (ii).

2.3.4 Decision Tree

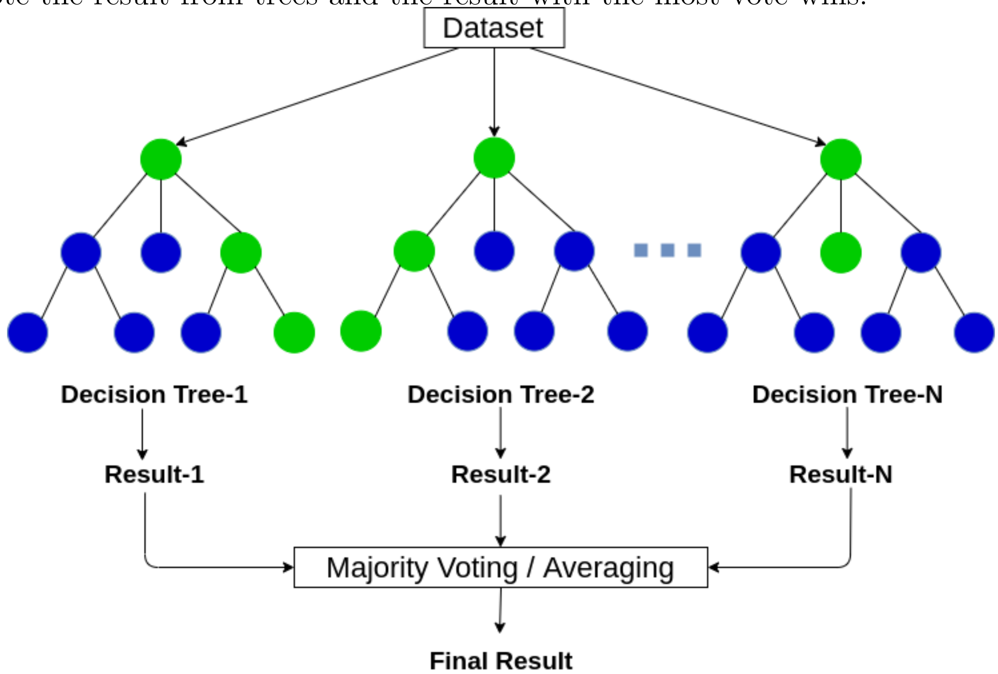
A decision tree is a flowchart-like structure in which each internal node represents a "test" on a feature, each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).
Example:



This example predicts the animal from the features "Has feathers","Can fly" and "Has flems".

2.3.5 Random Forest

As shown below, Random Forest is merely concatenation the result of n decision tree. It means that we vote the result from trees and the result with the most vote wins.

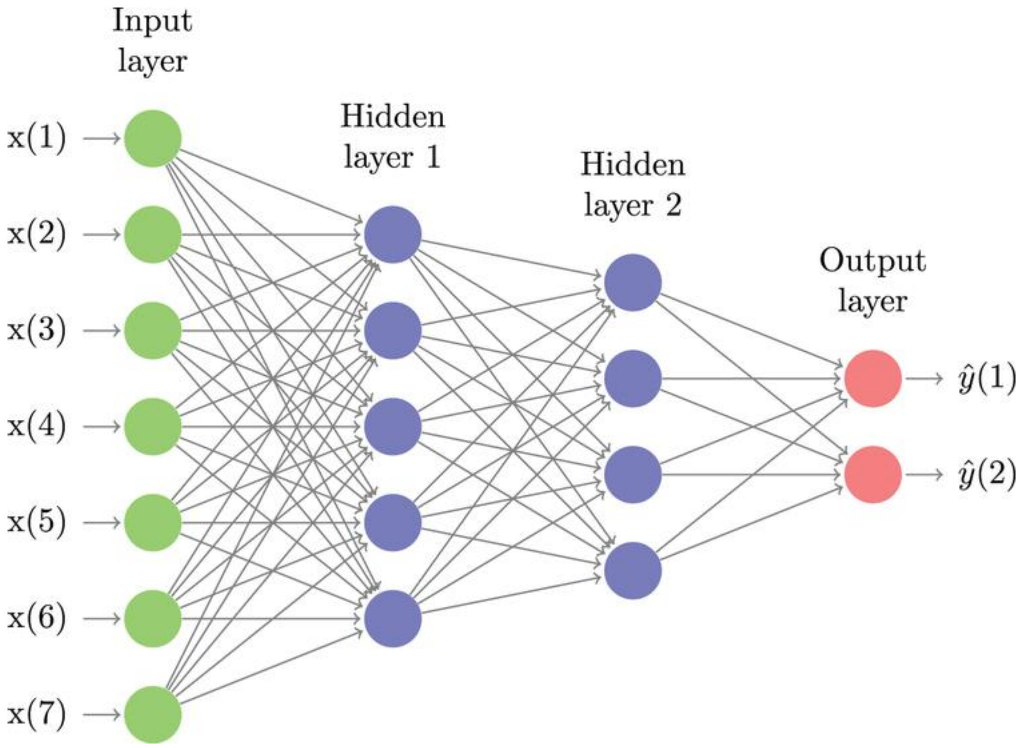


2.3.6 Naive Bayes

Naive Bayes is a non-learning classifier which merely calculates probability using Bayes principle which is: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ But since we don't know $P(B|A)$, we calculate it from Gaussian formula. So probably not a good idea when our data is not Gaussianly distributed.

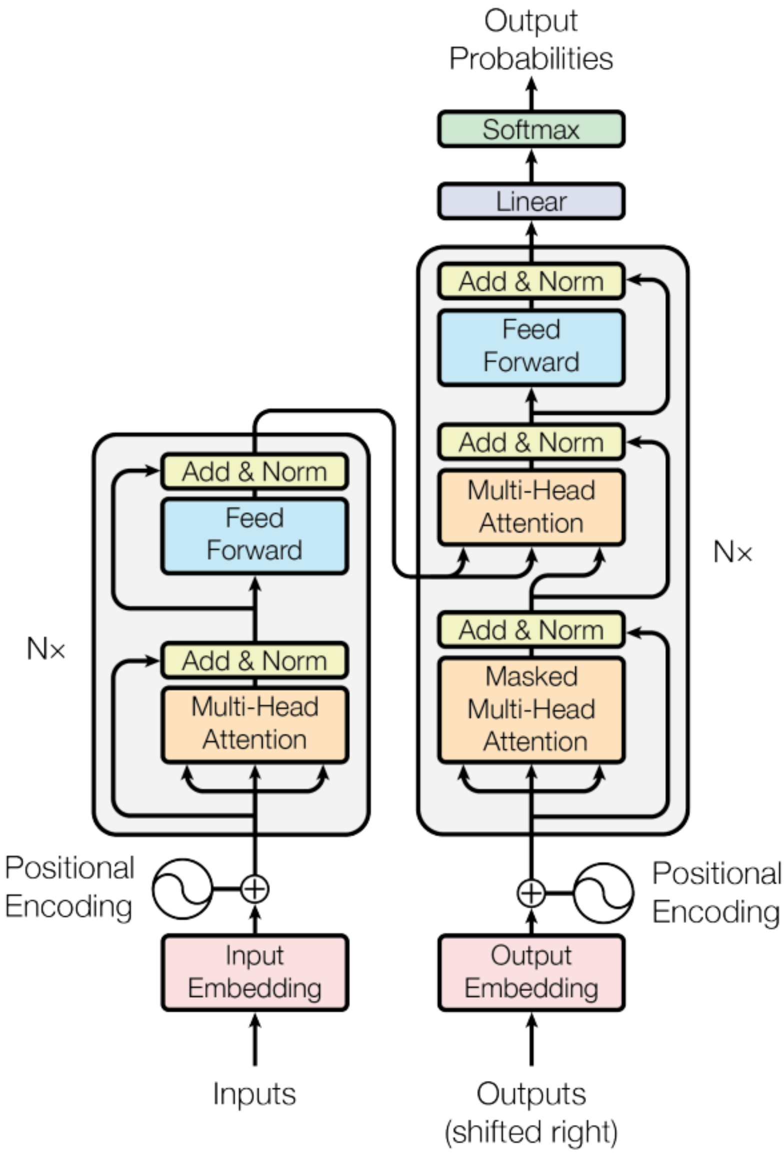
2.3.7 Neural Network

We can use neural networks for classification. For that, we should have f input nodes(number of features) and c nodes on the output layer(number of classes) and hidden layers can be anything. Then the values on each output node is the probability of the data being in that class. So we nominate the highest probability class for our answer.
In NN the input and output size is constant, but we tried different depths and size for the hidden layers and learning rate, then used the best one.
Example:



2.3.8 Transformers

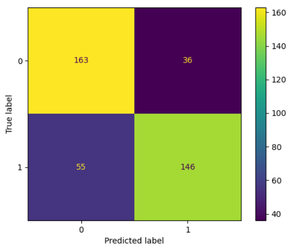
Transformer is the new type of model that was introduced by Google researchers in 2017 in this article. Its architecture is shown below:



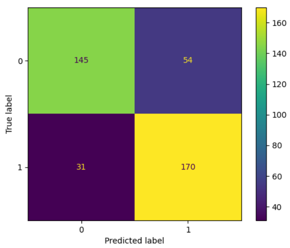
It's a complex Attention-based model that performs more accurate than RNN,CNN or even RCNN. For more information you can read my presentation here.

2.4 Results

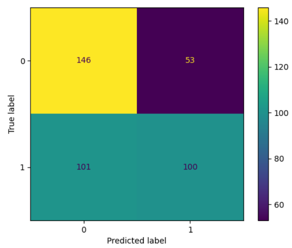
We trained all of the above models on the same set of training set and tested them with the same test set. And here's their confusion matrices:
For more intuitive understanding, we plot their evaluation charts:



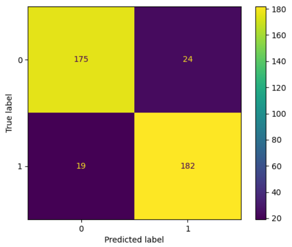
(a) Logistic-Regression Confusion-Matrix



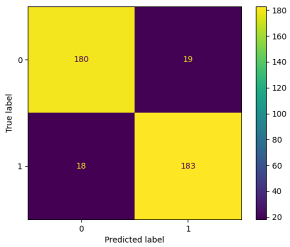
(b) SVM Confusion-Matrix



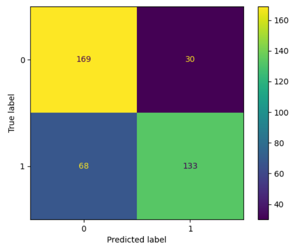
(c) KNN Confusion-Matrix



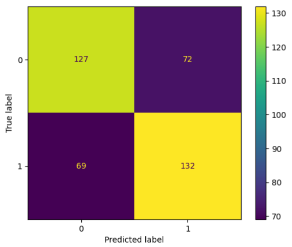
(d) Decission-Tree Confusion-Matrix



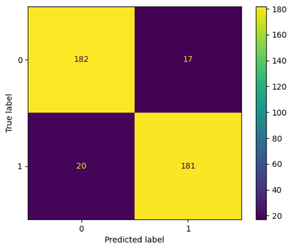
(e) Random-Forest Confusion-Matrix



(f) NaiveBayes Confusion-Matrix

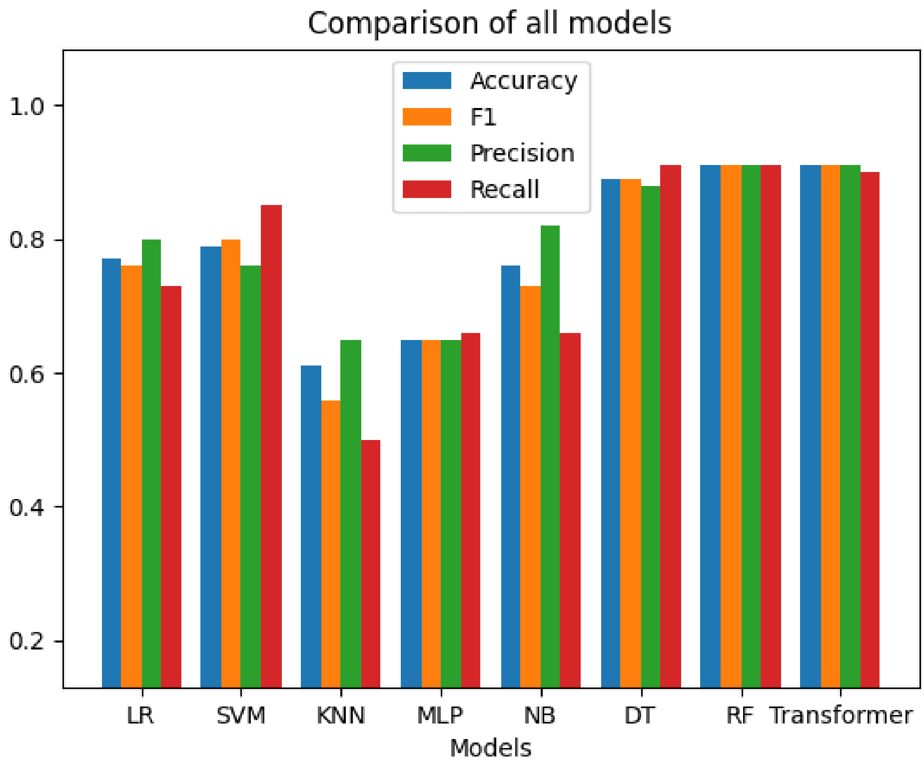


(g) MLP(simple neural network) Confusion-Matrix



(h) Transformer Confusion-Matrix

Figure 1: Confusion Matrices



As we can see, KNN and MLP did poorly on this set. But Decision Tree and Random Forest and Transformer performed really well; All of their scores are around 90% which is pretty good.

2.5 Acknowledgements

The full code of this assignment can be found here at my github.
Datasets were not uploaded due to their large size but can be found here: training-set, test-set.

References

[1] All you need is attention

[2] https://www.pico.net/kb/what-is-a-false-positive-rate/

[3] https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/

[4] https://www.analyticssteps.com/blogs/top-classification-algorithms-using-python

[5] https://content.iospress.com/articles/algorithmic-finance/af176

[6] https://link.springer.com/article/10.1007/s42452-019-1329-z

[7] https://www.analyticsvidhya.com/blog/2021/05/5-classification-algorithms-you-should-know-introductory-guide/

[8] https://docs.paperspace.com/machine-learning/wiki/logistic-regression

[9] https://en.wikipedia.org/wiki/Support-vector$_{m}achine$

[10] https://www.analyticsvidhya.com/blog/2021/05/5-classification-algorithms-you-should-know-introductory-guide/

[11] https://ai-pool.com/a/s/random-forests-understanding

[12] https://scikit-learn.org/stable/modules/neural$_{n}etworks_{s}upervised.html$

[13] https://arxiv.org/pdf/2005.05079.pdf

[14] Classification with Transformer

[15] https://www.geeksforgeeks.org/plotting-multiple-bar-charts-using-matplotlib-in-python/

[16]

[17]

[18]

[19]

[20]

[21]

[22]

[23]