## Abstract

In this project, we investigate linear classification. Logistic regression and k-Nearest Neighbor models are applied on 4 cleaned data sets: Ionosphere, Census Income, SpamBase, and Iris.

We found that in most cases KNN was more reliable at finding more accuracy in the data, while Logistic Regression was more reliable to apply to all datasets.In most cases the logistic regression would take several seconds and almost up to a minute in some cases, whereas the KNN would take less than ten seconds granted that it was working fine. This is interesting seeing as Logistic regression was easier to implement, one would think that it would be faster.

## Introduction

This project starts with 4 uncleaned datasets, which were then cleaned based on duplicate values, categorical values, and entries in which values were missing, or needed to be modified for calculations. We will then apply both Logistic Regression and KNN/K-fold cross validation to each dataset and compare the difference of computation time, accuracy and reliability, to see which model should be used for better results. We found that with the Adult dataset we could not achieve a KNN accuracy value due to the fact that the calculations were running infinitely. The spam base dataset was also not able to give us an accuracy value either due to an error in  the code which didn't allow the code to run because a value was too deep for the array error. This caused a lot of grief as we tried to fix both issues, but could not find a solution to either one of them. Nonetheless we still had sufficient basis to say that KNN has a better accuracy than logistic regression  and tends to compute faster when compared on the same dataset.

## Datasets

The Ionosphere dataset contains radar returns from the ionosphere, and the goal is to predict whether these returns are 'good' or 'bad'. The data set did not need to be cleaned, but the outputs did need to be converted from g/b to 1/0 for use with KNN and Logistic regression.

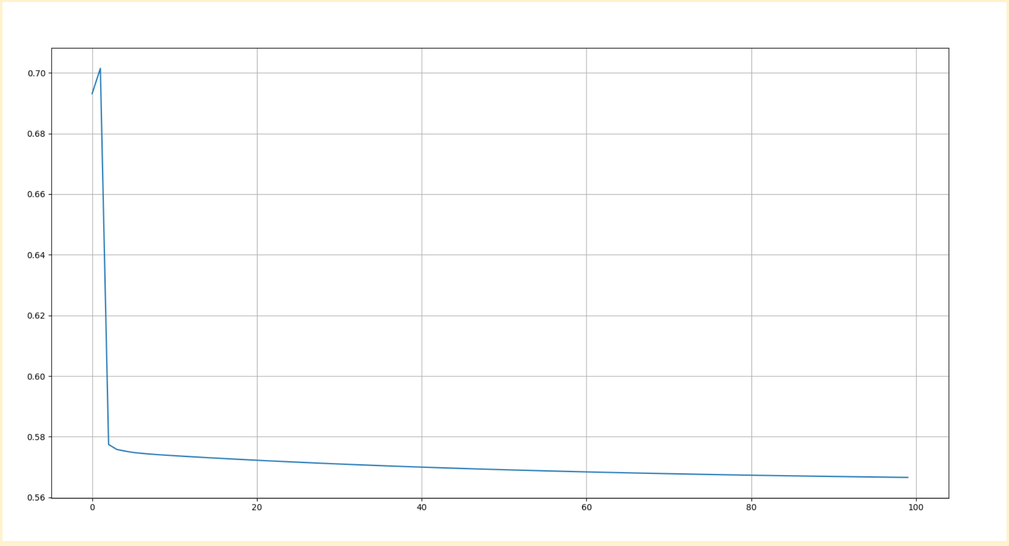
Within the SpamBase Dataset, the information provided is to determine whether an email is considered spam or not. The features within this dataset are of type word_freq_WORD, char_freq_CHAR, capital_run_length_average, capital_run_length_longest, and capital_run_length_total. These are crucial data points to determine whether the email is considered spam. The return values are either ‘yes, this email is spam (1)’ or ‘no, this email is not spam (0)’.

The Iris dataset uses pre recorded data to determine whether a plant is one of three different types. The setosa is the only one of the three that is linearly separable from the rest, whereas the other two are not, so we have opted to put the inseparable ones together into an output of 0 and the setosa as an output of 1.

The Adult dataset (also known as Census Income) uses various inputs such as education, age, hours worked per week and other attributes to determine whether a person gains more or less than 50 thousand dollars a year. The cleaning process involved removing all missing values and some categorical entries with a value of '?'. In the y variable detailing income, entries with a period at the end of their value were not removed but accordingly changed to '<=50K' or '>50K'. The income and sex columns were converted into boolean and all other categorical variables were one-hot-encoded. Analysis of a correlation matrix reveals information such as that having a bachelor's degree has the highest correlation with an income of over 50K/year out of all the education levels and that those who were never married had the least hours worked per week out of all marital statuses. A histogram shows that age is mostly distributed around 20 to 40, with a right skew. The mean age is 38.64.

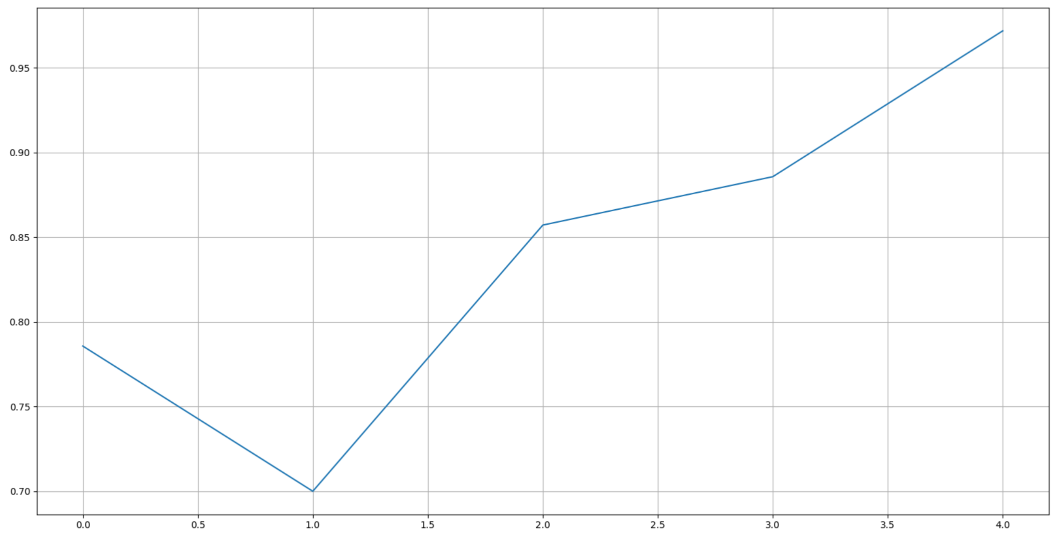
## Results

First, we will discuss the results of the Ionosphere dataset. We found that with the sample dataset that we were given, a learning rate of 0.1 was ideal for logistic regression, as with 100 epochs we were able to achieve an accuracy rate of 0.72. Looking at the graph we can see that we get a low value of 0.56 with only 100 epochs, if we were to increase the number of epochs that number would drop lower.



Logistic Regression for Ionosphere

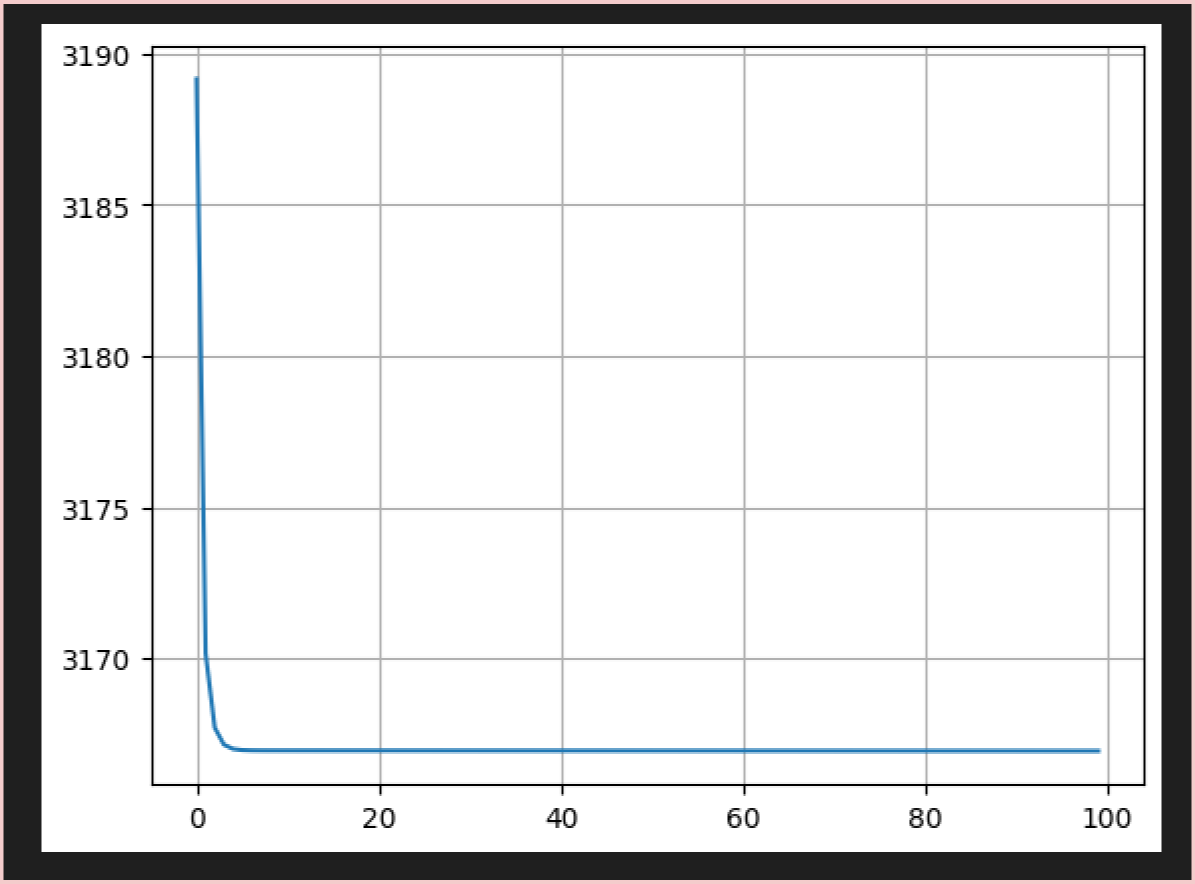
Looking at the KNN on the other hand we have interesting values when using a K of 4. For example when using 5 fold cross validation we get a final accuracy rate of 97.2%



KNN accuracy graph for Ionosphere

Comparing the two accuracies of both the KNN and the Logistic Regression we see that the KNN manages to get a much better accuracy with less iterations, this could be due to the way the dataset itself is set up because as we'll see with the other datasets this isn't always the case.

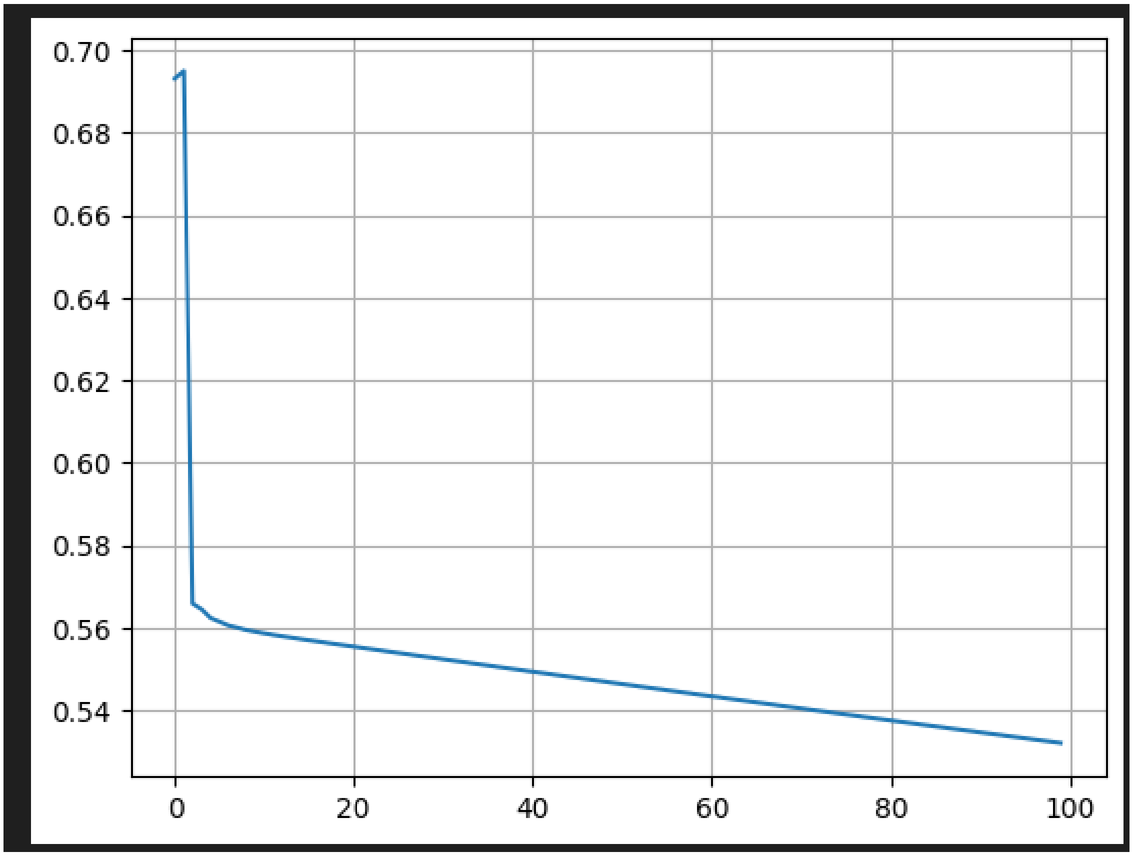
Next we'll move onto the Spam base dataset. This dataset was very tedious to work on because we found that using Log. Reg. We needed a learning rate of 0.000000001 in 100 epochs in order to get a nice curve with the lowest loss rate possible. This value could be made bigger by increasing the number of epochs but nonetheless it still takes a lot of time to calculate. Looking at the graph we have a very steep drop at the start and then a very gradual decline until the lowest point, which is a reflection of the very small learning rate as each iteration only declines by a very small and miniscule amount.



Logistic Regression for Spam Base

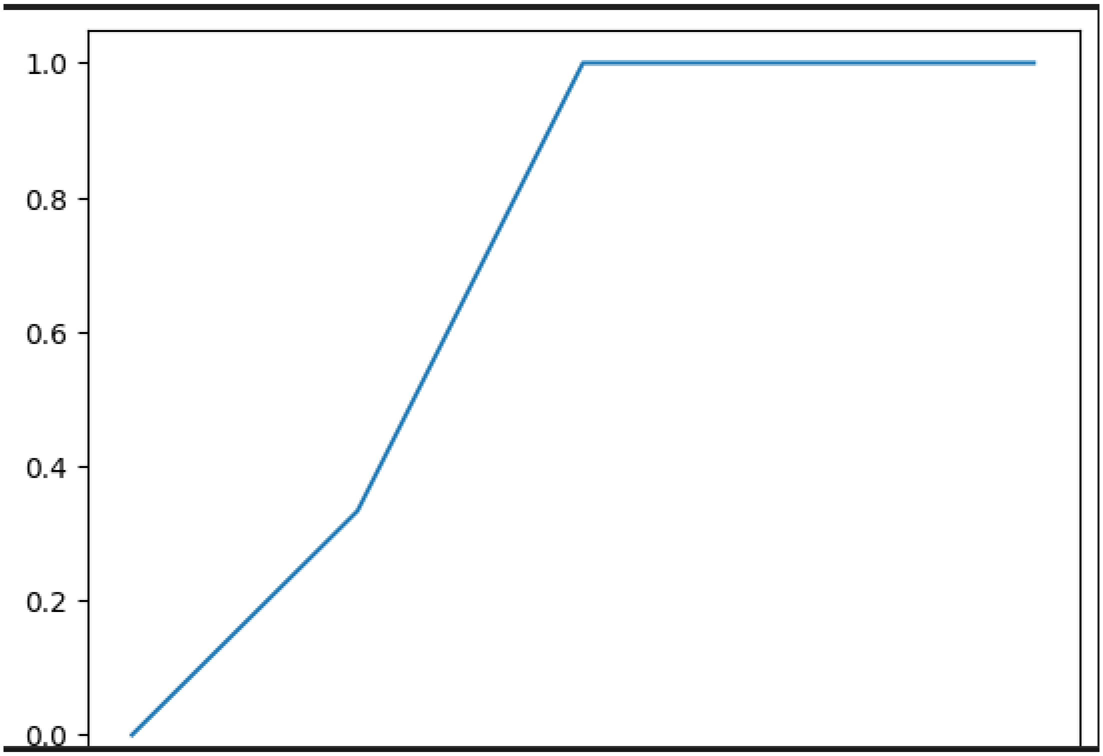
Unfortunately while trying to run KNN on this dataset, we were presented with an error that we could not find a solution to. This error stated that as we were traversing our table for KNN, the object we were looking for was too deep for the desired array, meaning that the code was interrupted and the KNN calculations would not complete.

Second to last we will discuss our results from the Iris data. The iris data was very interesting in the sense that because we combined 2 of the 3 outputs of the dataset, we were able to easily find high accuracy in the KNN, but relatively low accuracy in the Logistic Regression. Through testing we found that the learning rate which gave us the best results was the 0.05, which gave us a nice curve and a low loss rate of around 0.535 after 100 epochs but gave us an accuracy rate of only 66% which should have been higher.



Logistic Regression for Iris

From the graph we can see that we have a steep drop and then a steady decline with somewhat heavy change from iteration to iteration, this of course is due to the higher learning rate which causes the gradual decline to be more steep than in the other data sets. If we look at the KNN instead, we used a K of 100 as that gave use the most changes with the 5 folds, but even then, we still had 2 folds which both had an accuracy of 100%, whether this was a coding overlook in the k-fold calculation, the way we manipulated the data to combine two of the outputs, or just the way the data was set up, it is interesting to see how the KNN performed and was drastically different to the Logistic Regression. Using a K less than 100 resulted in three of the five folds being 100% accurate and any K under 40 resulted in a straight horizontal line at 100% accuracy.



KNN accuracy graph for Iris

Lastly we will discuss the Adult/Census dataset. This data set took a little more work as a lot of the entries needed to be converted into booleans for calculations. Taking a look at the Logistic regression we saw that we needed a very small learning rate in order to achieve any form of curve. When using a value larger than 0.0000000001 we found that the graph would have large spikes and dips, and that only with values equal to or smaller than our current learning rate would we get a curve or at least something resembling a curve and have an accuracy of 76%. Unfortunately our KNN implementation on this dataset was not possible as the calculation would infinitely run and would clock out after a while. This meant that we were not able to properly compare the different models on this data set, but the fact that it is taking so long to calculate could be a sign that the dataset is really inefficient or is not suited for KNN modeling.

## Discussion and conclusion

In conclusion, we discovered that although logistic regression was easier to implement, it ended up taking much longer, and would typically have an accuracy rate lower than that of KNN and K-fold cross validation. Even though our KNN only worked on two of our four datasets it was still evidently clear that KNN was more reliable when it came to the accuracy and speed of the calculations. Through our experiments we were able to see how learning rate and K really do affect the graph of a function, whether it was turning the logistic function from a graph of spikes to a nice curve, or turning an accuracy graph from a horizontal line to an increasing line to see progression. All around it was fascinating to see the changes and compare the differences between the two models regardless of coding errors or infinitely running code.

## Statement of contributions

**Coding**

Documentation: Chetas P, Fernando G
Knn code: Kevin L, Fernando G, Chetas P, Gavin W
Regression code: Fernando G, Chetas P

Census Income
Cleaning: Zhenyang D, Fernando G
Knn implementation: Fernando G
Logistic regression implementation: Fernando G, Chetas P

Spambase
Cleaning: Chetas P
Knn implementation: Fernando G
Logistic regression implementation: Chetas P

Iris
Cleaning: Kevin L, Chetas P
Knn implementation: Fernando G
Logistic regression implementation: Fernando G,Chetas P

Ionosphere
Cleaning: Fernando G
Knn implementation: Fernando G
Logistic regression implementation: Fernando G,Chetas P

**Writeup:**

Abstract: Zhenyang D
Introduction: Zhenyang D, Fernando G
Datasets: Zhenyang D, Fernando G
Results: Fernando G
Discussion and conclusion: Fernando G
Statement of contributions: Zhenyang D