PART 2

Techniques to improve performance of Basic kNN and Distance-weighted

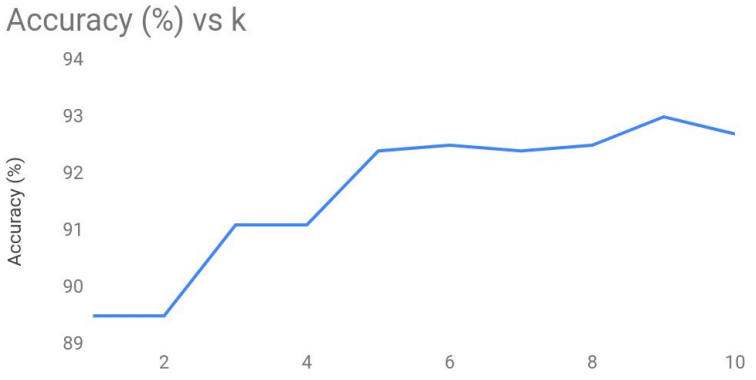
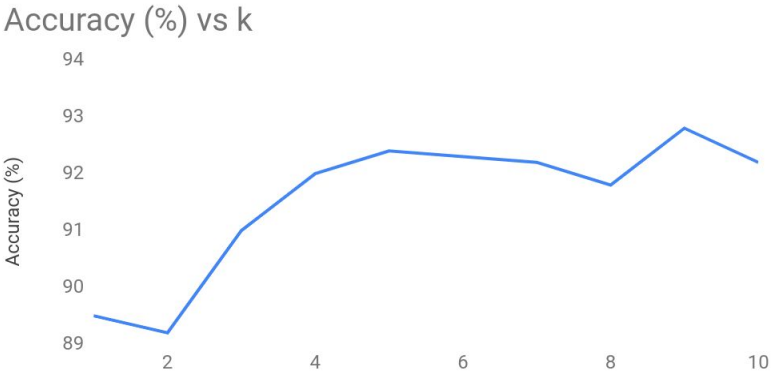
Introduction:

To investigate how the performance of **basic and Distance-weighted** kNN can be improved, I have altered and played around with different parameters. The different hyper-parameters which I have modified are values of 'k' and the distance metrics. Below charts will show the accuracy value when different distance metrics are evaluated.

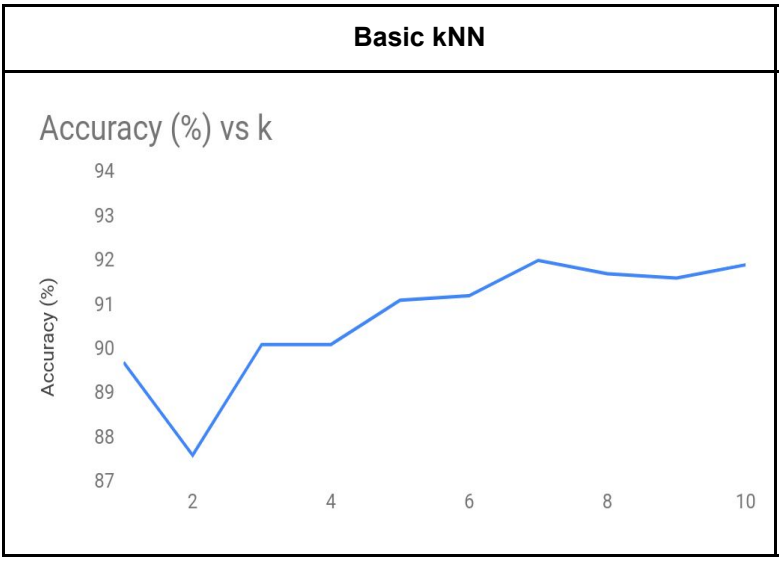
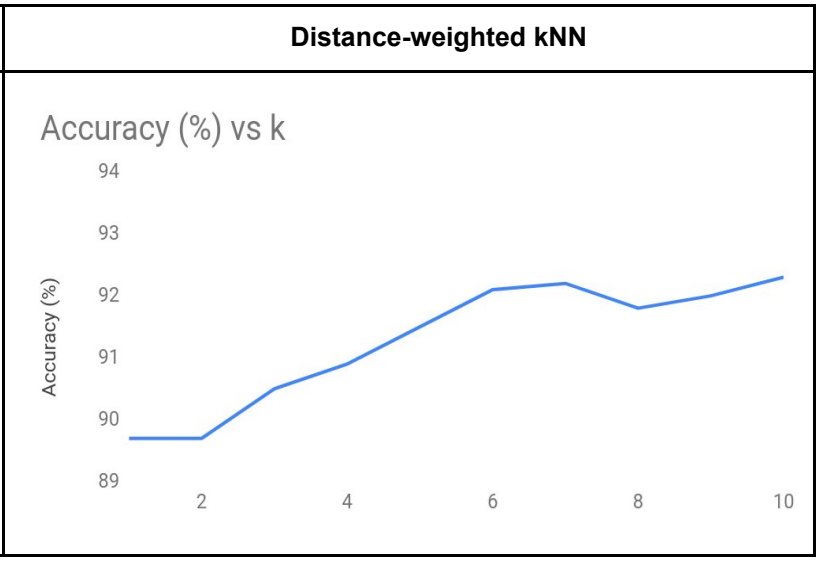
There 2 main sections below. 1. When **k=1 to 10** and the other is **k=1 to100(intervals of 10)**. Each section has 3 sub-sections. All these sub-sections are measured using different distance metrics. Each sub-section will have 2 graphs: **Basic kNN** and **distance-weighted kNN**.

Section 1: k: 1 to 10, Distance metrics: **Euclidean distance, Manhattan distance** and **Minkowski distance**.

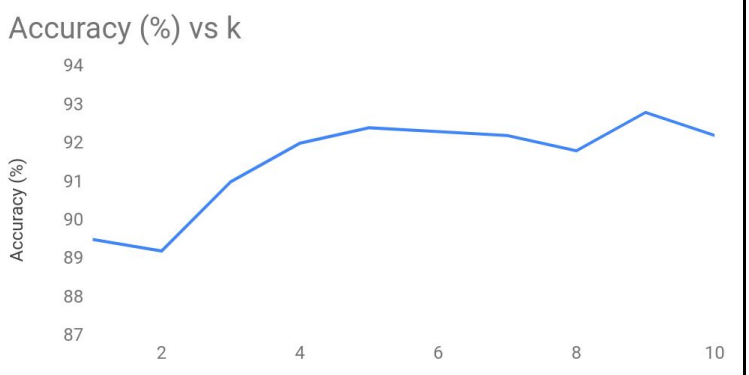
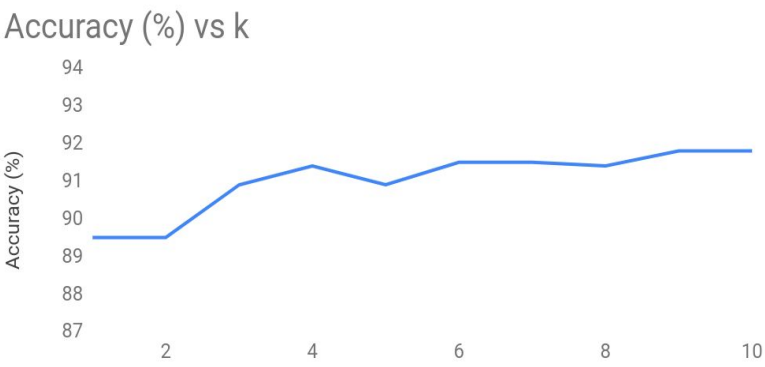
1) Distance Metric: **Euclidean Distance**

Basic kNN	Distance-weighted kNN
	

2) Distance Metric: **Manhattan Distance**

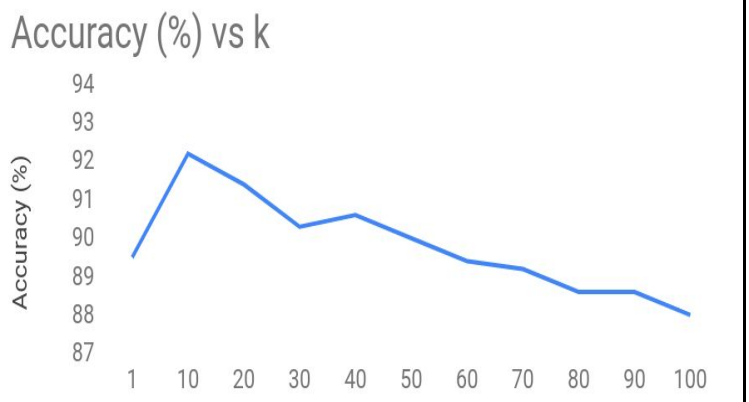
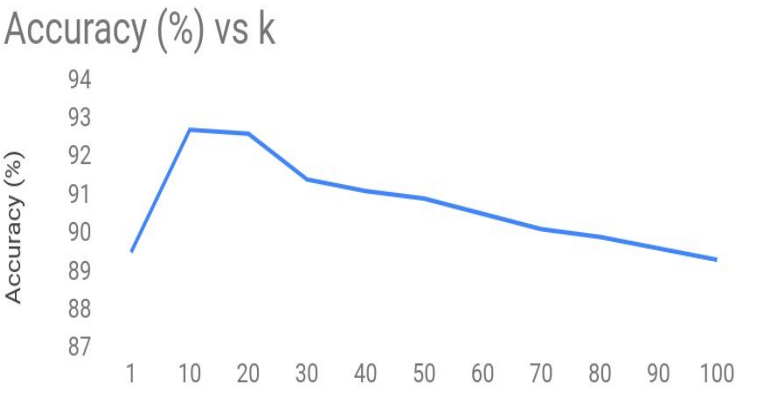
Basic kNN	Distance-weighted kNN
	

3) Distance Metric: **Minkowski Distance**

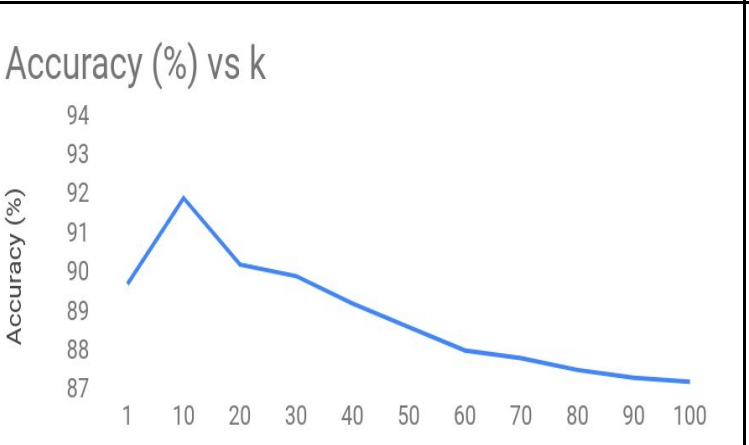
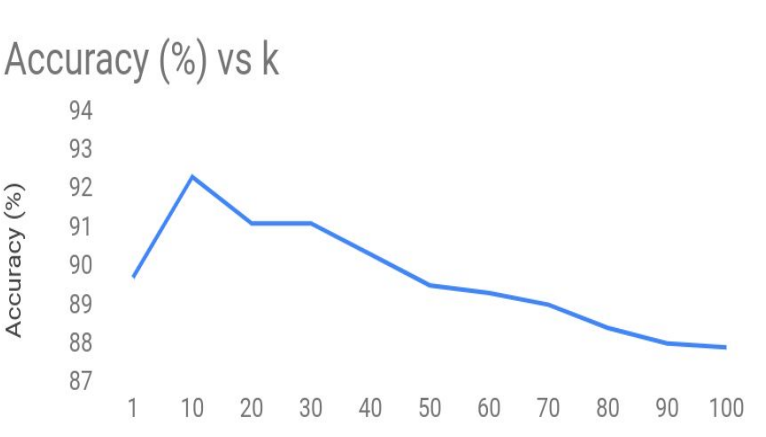
Basic kNN	Distance-weighted kNN
	

<u>Section 2</u>. Now, k**: 1-100 at intervals of 10**, Distance metrics: **Euclidean distance, Manhattan distance** and **Minkowski distance**.

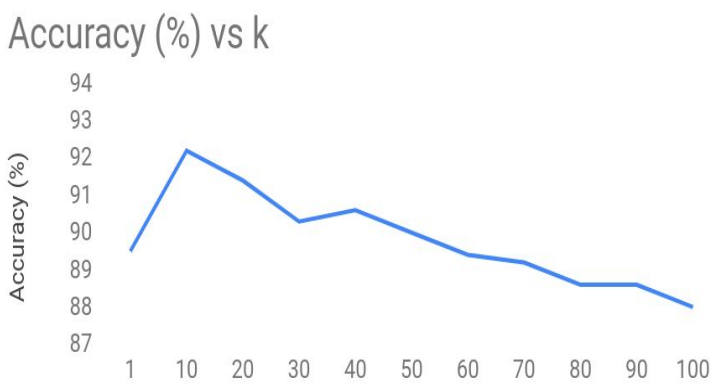
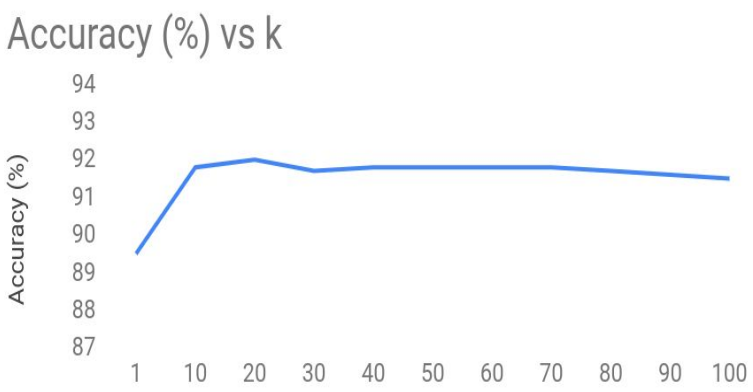
Distance Metric: **Euclidean Distance**

Basic kNN	Distance-weighted kNN
	

Distance Metric: **Manhattan Distance**

Basic kNN	Distance-weighted kNN
	

Distance Metric: **Minkowski Distance**

Basic kNN	Distance-weighted kNN
	

Observations:

<u>Section 1</u>:

We can observe that when we increase the value of 'k' from 1 to 10, there is an improvement in the accuracy of both variants of kNN. Even when using the different distance metrics, there is a change in accuracy.
When distance-metric is 'euclidean distance', the maximum accuracy is **93%** and it was when the value of **k was 9**. This was the **highest accuracy** received amongst all the different distance metrics for basic kNN as well distance-weighted kNN. Overall, for increasing values of 'k' accuracy tends to increase.

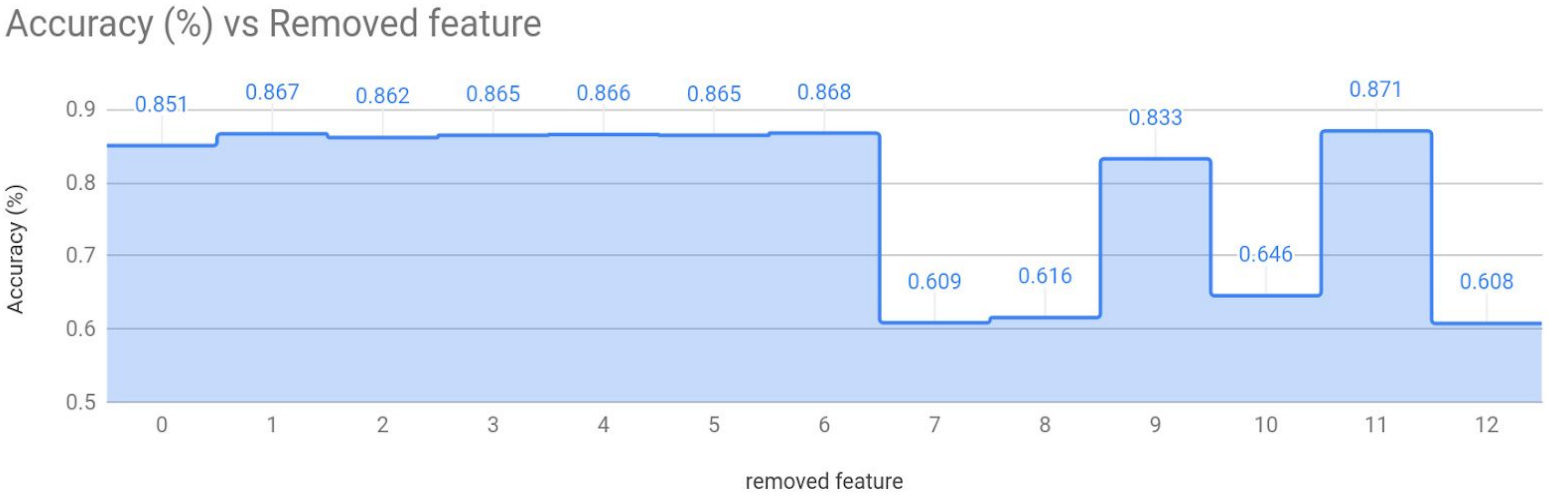
<u>Section 2</u>:

We observed a fall in accuracy when we take greater values of 'k'. For the values of k from 1 to 100 (intervals of 10), we see a lower accuracy. Even after changing the distance metrics, the accuracies do not tend to increase for greater values of 'k'.
The highest accuracy we recorded is **92.7%** when **k was 10** and distance metrics is 'euclidean distance'. The observation also states that euclidean distance gives the best accuracy amongst other distance metrics.

PART 3

1. The R^2 value of my kNN to solve regression problems for **k=10** is **0.85.**
2. The kNN model built by me predicts a class. The class prediction is calculated using euclidean distances of training data and test instance. As we know that euclidean distance considers all the feature values. There might be irrelevant features, redundant features or any features whose values are very high/low as compared to other features. These features may degrade the performance of the kNN algorithm.
   We have a range of possible methods to avoid this problem. Some of them are:
   1. Supervised methods (wrapper method).
   2. Unsupervised methods (filter method).

   After a thorough research about the feature selection I decided to implement feature selection using wrapper method. In wrapper method, I am using backward search method. In this method different subsets of feature sets are considered. For eg. first compute R^2 value for all the features $\{F^1, F^2, F^3 \ldots F^n\}$, then we will consider $\{F^2, F^3 \ldots F^n\}$, then $\{F^1, F^3, F^4 \ldots F^n\}$ and so on. In each iteration we will record accuracy and after all the iterations we will find the subset of features which gives the best accuracy.
   In this approach, I will be removing different features from the training and test dataset to analyze the best possible accuracy. After experimentation, I have discovered that by elimination of feature 11 (column 11) from the datasets, the R^2 value have improved.
   I have implemented this method using core python and numpy arrays. There were high-level libraries available but better understanding and improving the coding standards in numpy I have implemented on my own.



The original R2 value when all features('0') are considered and the accuracy is **0.85.**
We can see from the above chart that removing features 7, 8, 10 or 12 degraded the accuracy. This means that these features are extremely important. On the other hand, when we remove feature 11 we acquired the best R2 value (**0.871**). This means that we can remove feature 11 from the training and test instances as it **negatively impacts** our R2 value.

Conclusion:

After completion of assignment 1, I have learned and practically implemented different variants of k-Nearest Neighbours algorithm, i.e. **Basic kNN** and **Distance-weighted kNN**. Along with this I have also observed how different hyperparameters like ‘k’ and distance metric can affect the accuracy of the algorithm. At the end, I developed an k-NN algorithm to solve regression problems where I researched how **feature selection** plays an important role in improving the accuracy of the algorithm.