Flower Classification Using KNN

**Introduction:**

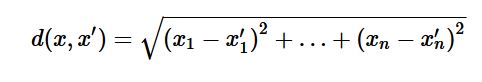
The KNN algorithm is a robust and versatile classifier that is often used as a benchmark for more complex classifiers such as Artificial Neural Networks (ANN) and Support Vector Machines (SVM). Despite its simplicity, KNN can outperform more powerful classifiers and is used in a variety of applications such as economic forecasting, data compression and genetics. KNN falls in the **supervised learning** family of algorithms. Informally, this means that we are given a labelled dataset consisting of training observations (*x*,*y*) and would like to capture the relationship between *x* and *y*. More formally, our goal is to learn a function *h*:*X*→*Y* so that given an unseen observation *x*, *h*(*x*) can confidently predict the corresponding output *y.*

The KNN classifier is also a **non parametric** and **instance-based** learning algorithm.

* **Non-parametric** means it makes no explicit assumptions about the functional form of h, avoiding the dangers of mismodeling the underlying distribution of the data. For example, suppose our data is highly non-Gaussian but the learning model we choose assumes a Gaussian form. In that case, our algorithm would make extremely poor predictions.
* **Instance-based** learning means that our algorithm doesn’t explicitly learn a model. Instead, it chooses to memorize the training instances which are subsequently used as “knowledge” for the prediction phase. Concretely, this means that only when a query to our database is made (i.e. when we ask it to predict a label given an input), will the algorithm use the training instances to spit out an answer.

**Methodologies:**

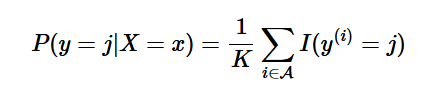
In the classification setting, the K-nearest neighbor algorithm essentially boils down to forming a majority vote between the K most similar instances to a given “unseen” observation. Similarity is defined according to a distance metric between two data points. A popular choice is the Euclidean distance given by :



but other measures can be more suitable for a given setting and include the Manhattan, Chebyshev and Hamming distance. More formally, given a positive integer K, an unseen observation *x* and a similarity metric *d* ; KNN classifier performs the following two steps:

* It runs through the whole dataset computing *d* between *x* and each training observation. We’ll call the K points in the training data that are closest to *x* the set A. Note that K is usually odd to prevent tie situations.

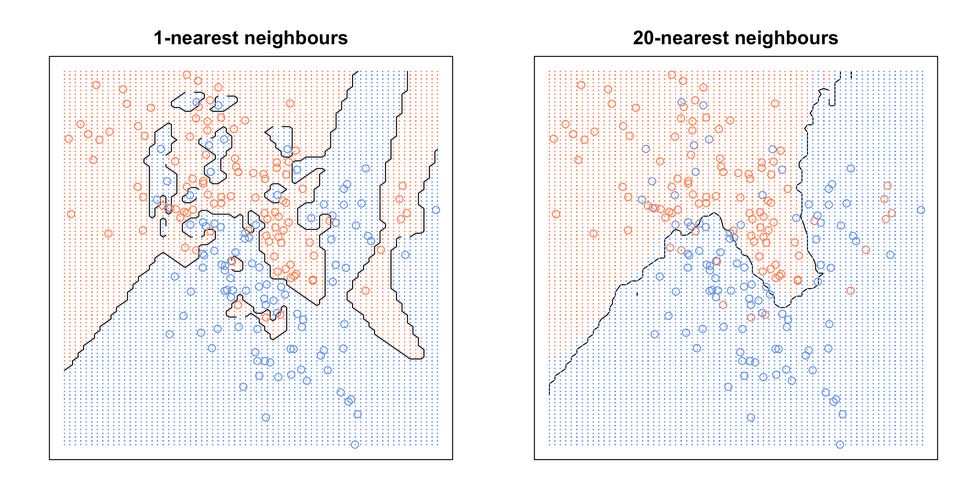
 It then estimates the conditional probability for each class, that is, the fraction of points in A with that given class label. (Note *I*(*x*) is the indicator function which evaluates to 1 when the argument *x* is true and 0 otherwise)



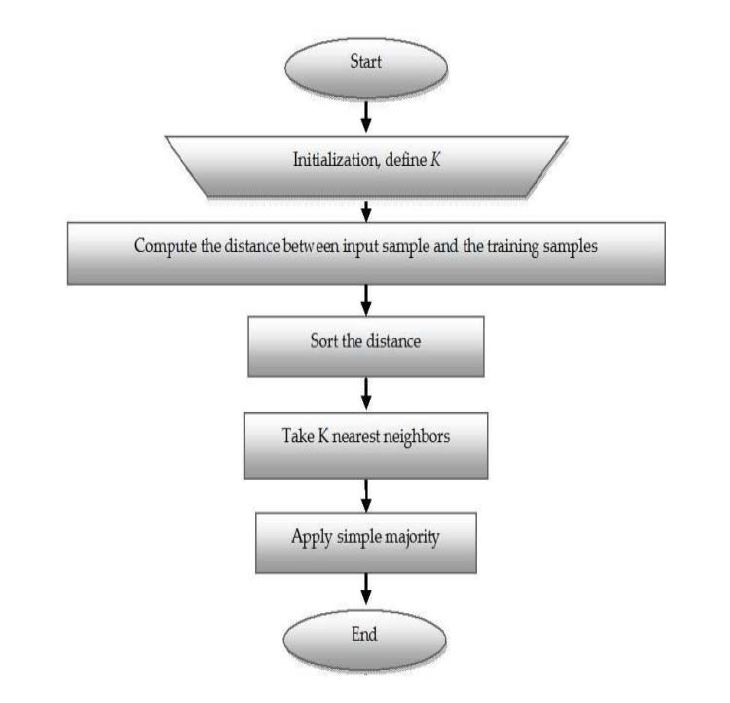
Finally, our input *x* gets assigned to the class with the largest probability.

**How to Choose K :**

When K is small, we are restraining the region of a given prediction and forcing our classifier to be “more blind” to the overall distribution. A small value for K provides the most flexible fit, which will have low bias but high variance. Graphically, our decision boundary will be more jagged. On the other hand, a higher K averages more voters in each prediction and hence is more resilient to outliers. Larger values of K will have smoother decision boundaries which means lower variance but increased bias. Usually data scientists uses only odd values for k (to cut tie cases) but sometimes even is used as well. Another simple intuition is to assign k as sqrt (N) , where is N = no of total samples. But the best method is to take a range of numbers e.g 1 to 50 and test all the values against the test data and check for which k best accuracy is found.

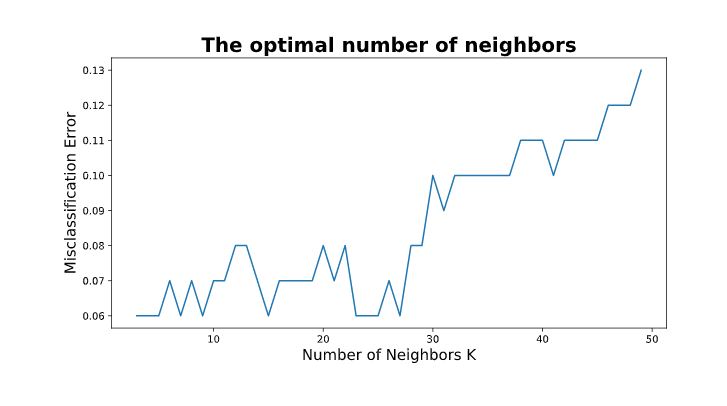


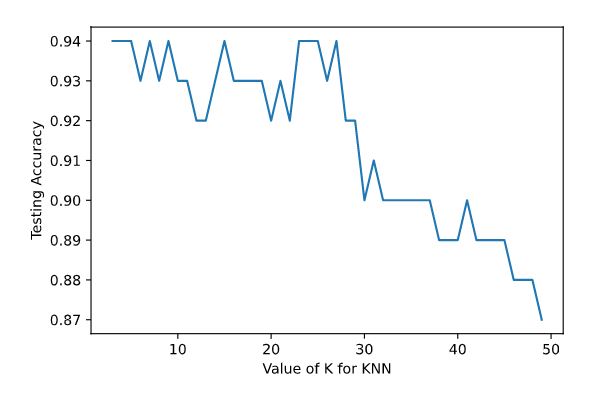
**Algorithm :**

****

**Result and Discussion:**

The data set used in this project is the Iris Flower Dataset (IFD) which was first introduced in 1936 by the famous statistician Ronald Fisher and consists of 50 observations from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor). Four features were measured from each sample: the length and the width of the sepals and petals. A KNN classifier was trained to be able to distinguish the species from one another given the measurements of the 4 features. Now for this project the result of KNN was compared with a random forest classifier and neural network classifier.





Accuracy of models is given below:

|  |  |
| --- | --- |
| Model Name | Test Accuracy |
| KNN Classifier | 98% |
| Random Forest | 98.67% |
| Neural Network | 98.67% |

So we can see that for iris flower classification random forest and Neural Networks performs better. With better param tuning it may be possible to improve the current KNN network.