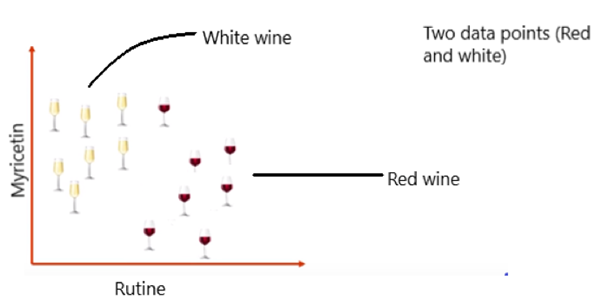
**K-Nearest Neighbours Algorithm**

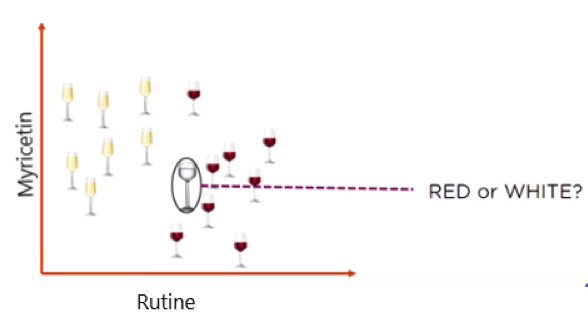
K Nearest Neighbour is a simple algorithm that stores all the available cases and classifies the new data or case based on a similarity measure. It is mostly used to classifies a data point based on how its neighbours are classified.

Let us consider a particular example of two categories of wine “Red” and “White”. Two chemical components called Rutime and Myricetin. Consider a measurement of Rutine vs Myricetin level with two data points, Red and White wines. They have tested and where then fall on that graph based on how much Rutine and how much Myricetin chemical content present in the wines.

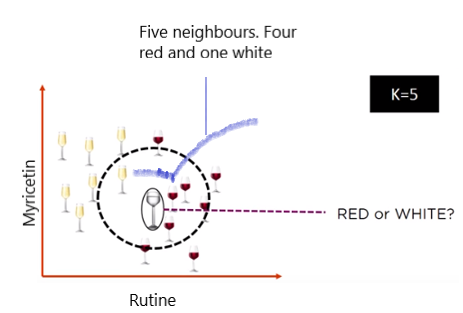


‘k’ in KNN is a parameter that refers to the number of nearest neighbours to include in the majority of the voting process.

Suppose, if we add a new glass of wine in the dataset. We would like to know whether the new wine is red or white?



So, we need to find out what the neighbours are in this case. Let’s say k = 5 and the new data point is classified by the majority of votes from its five neighbours and the new point would be classified as red since four out of five neighbours are red.



**How to choose the value of k**

1) There is no structured method to find the best value for “K”. We need to find out with various values by trial and error and assuming that training data is unknown.

2) Choosing smaller values for K can be noisy and will have a higher influence on the result.

3) Larger values of K will have smoother decision boundaries which mean lower variance but increased bias. Also, computationally expensive.

4) Another way to choose K is though cross-validation. One way to select the cross-validation dataset from the training dataset. Take the small portion from the training dataset and call it a validation dataset, and then use the same to evaluate different possible values of K. This way we are going to predict the label for every instance in the validation set using with K equals to 1, K equals to 2, K equals to 3.. and then we look at what value of K gives us the best performance on the validation set and then we can take that value and use that as the final setting of our algorithm so we are minimizing the validation error .

5) In general, practice, choosing the value of **k** is **k = sqrt(N)** where **N** stands for the **number of samples in your training dataset**.

6) Try and keep the value of k odd in order to avoid confusion between two classes of data