# KNN

* Supervised multi-class classifier model.
* While building the KNN model we have to be really careful in data cleaning, outlier treatment and the data should be normalized
* Data should be normalized because this involves distance calculation. Distance (Euclidian distance mostly)can be calculated only on numerical columns (One hot encoding)

**Model building:**

1. Pick number of neighbours you want for classification k= something.
2. Choose a distance metrics
3. Have a dataset with records / training data.
4. For every new point (every point in test data) find the nearest neighbours
5. Now based on the vote/count classify the item.

All the techniques we have learnt are little different (logistic, decision trees, random forest, SVM). In these techniques we train a model on the training data and only take the model equation to the test dataset for prediction (These learners are called eager learners).

But in case of KNN. It uses all the training data or subset which represents the population in test dataset. In terms of huge datasets it builds a voronoi cells classifying each object. This type of learning is called lazy learning or instance based learning. This model requires more memory as complete data (train data) is load. Hence KNN suffers from curse of dimensionality.

**Model performance:**

We will perform cross-validation, split the data into train and test datasets (70-30). Build a model on training dataset and check the training accuracy.

* Now apply the model on test dataset (using predict function). Check test accuracy.
* If the train and test accuracy is not satisfactory, then change the model parameters and build new models.
* Continue this step until u get a decent train and test accuracies.(By doing this you will not over fit the model on train dataset)
* Also check precision, recall, F1 score and Support.
* Finally we can check explained variance value (best is 1) and mean absolute error (best is 0)

**NOTE: This should be performed for all the models. K- Fold validation can also be performed.**