Business Analytics

Random Forest

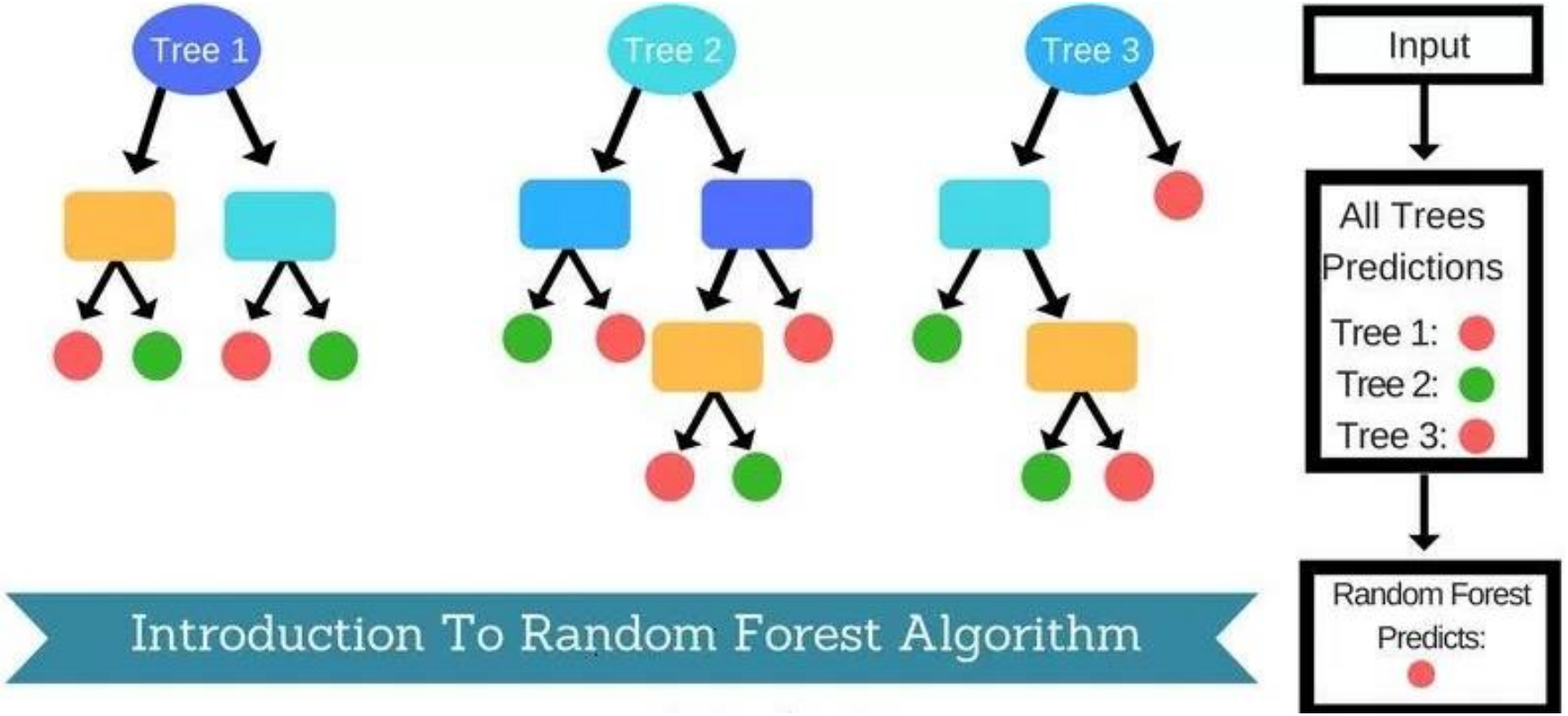
Proschool
An IMS Initiative

Random Forest

A commonly used class of ensemble algorithms are forests of randomized trees.

Random forest algorithm is a supervised classification algorithm. As the name suggest, this algorithm creates the forest with a number of trees.

Lets understand it better with an example.

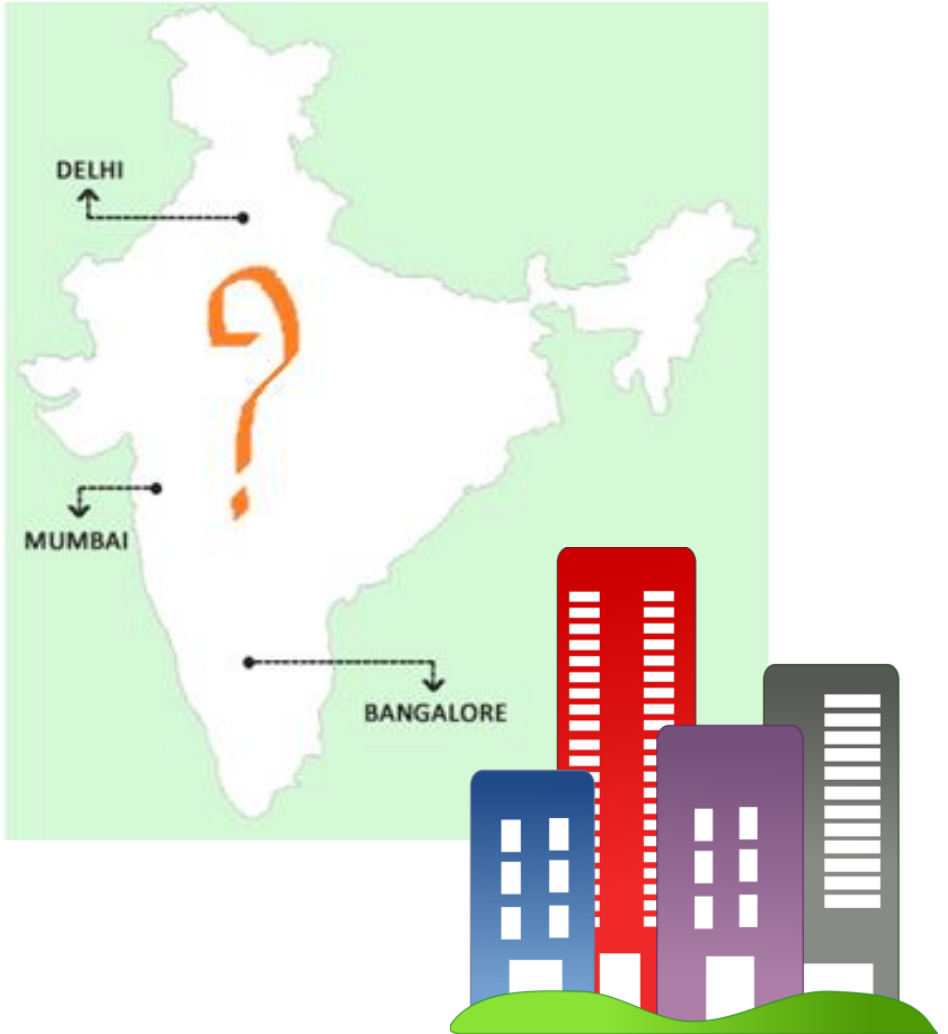


Random forest - Real life example

Suppose **Kiara** decides to buy a house and settle in one of these cities – Mumbai, Delhi or Bangalore. She wants to make a right and optimal choice considering all the perspectives since it is a very important decision for her.

So she decided to ask her **best friend** about the places she may like. Then her friend started asking about her opinions. It's just like her best friend will ask, You have visited X city. Did you like it?, etc..

Based on the answers which are given by Kiara, her best friend will start recommending the place Kiara may like. Here her best friend forms the decision tree with the answer given by Kiara.



As Kiara's best friend may recommend HER the best place by virtue of being her friend.
The model will be **biased** with the closeness of their friendship.
So she decided to ask few more friends to recommend the best place she may like.

Now her friends asked some **random questions** and each one recommended one place to Kiara. Now Kiara considered the place which has **highest votes** from her friends as the final place to settle down.

In the above decision process, two main interesting algorithms decision tree algorithm and random forest algorithm are used.

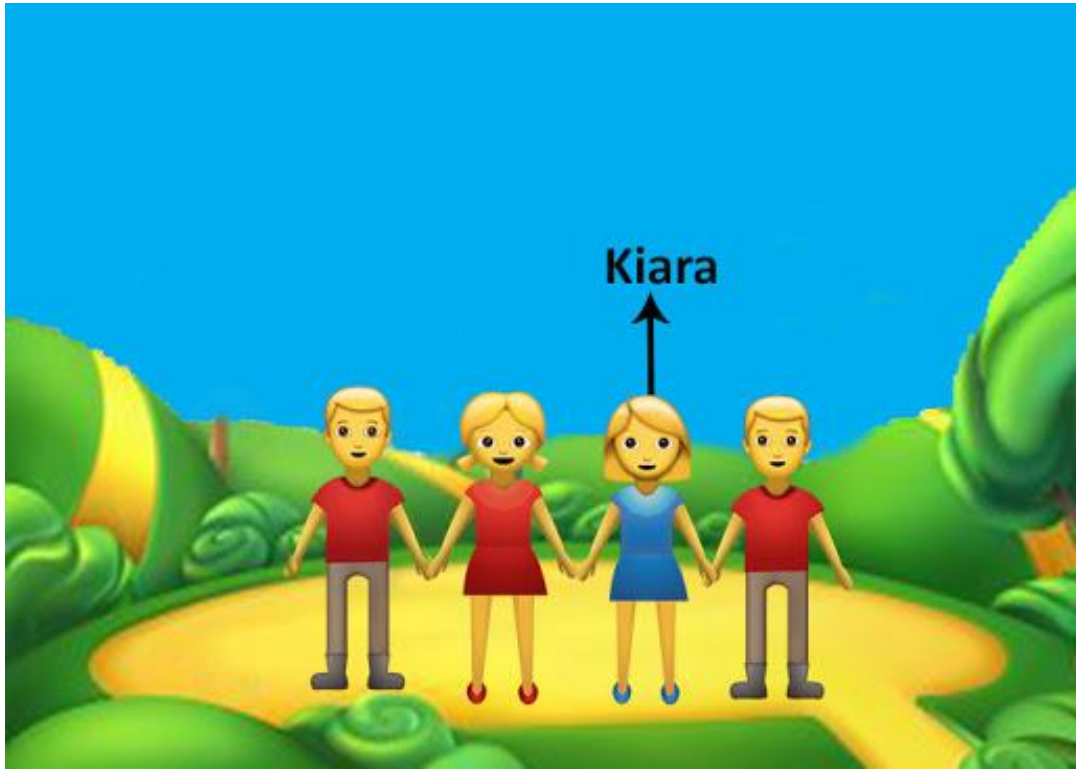
Lets see how…

To recommend the best place to Kiara, her best friend asked some questions. Based on the answers given by Kiara, she recommended a place. This is **decision tree algorithm** approach.

Here Kiara's best friend is the decision tree. The vote (recommended place) is the leaf of the decision tree (Target class). The target is finalized by a single person, In a technical way of saying, using an only single decision tree.

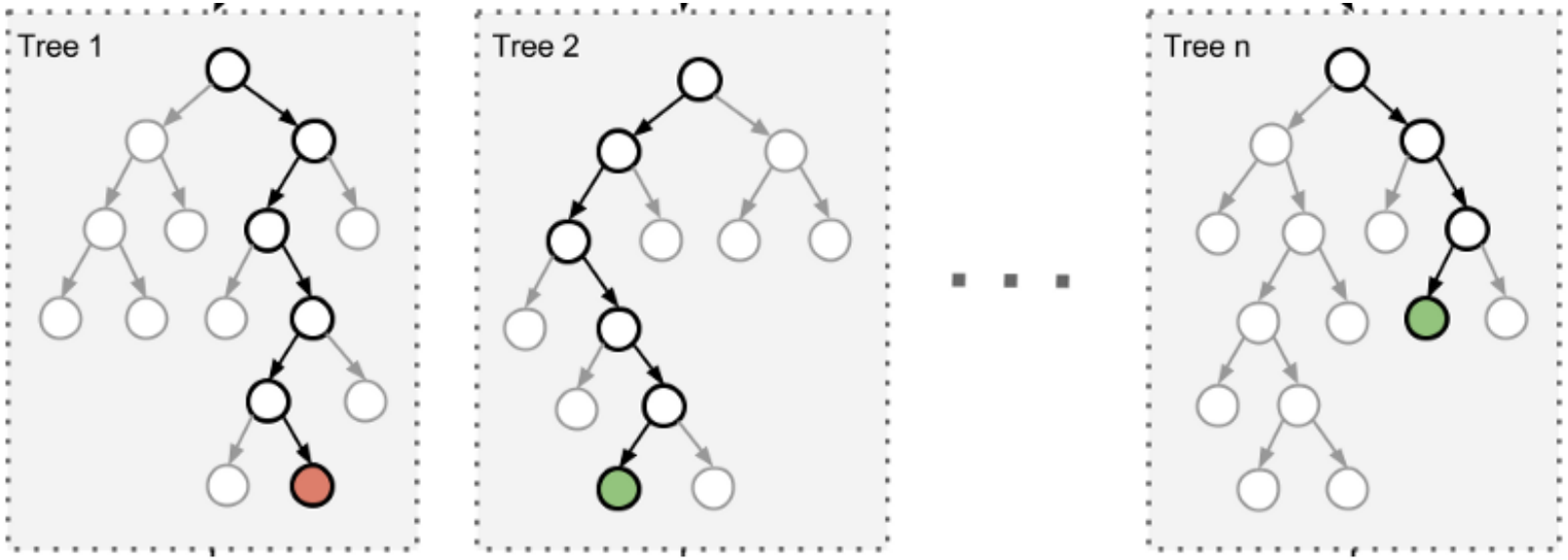
In this case when Kiara asked her friends to recommend the best place to settle among the three. Each friend asked her different questions and came up with their recommendation of a place.
Later Kiara considered all the recommendations and calculated the votes. Votes basically, to pick the popular place from the recommended places from all her friends.
Here, each friend is the tree and the combination of all friends (trees) will form the forest.

This forest is the **random forest**,
as each friend asked random questions to recommend the best place to settle down among the three.



In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e. a bootstrap sample) from the training set. Hence, instead of using all the features, a random subset of features is selected, further randomizing the tree.

As a result, the bias of the forest increases slightly, but due to the averaging of less correlated trees, its variance decreases, resulting in an overall better model.



Random Forest Algorithm - Working

•Assume number of cases in the training set is N. Then, sample of these N cases is taken at random but *with replacement*. This sample will be the training set for growing the tree.

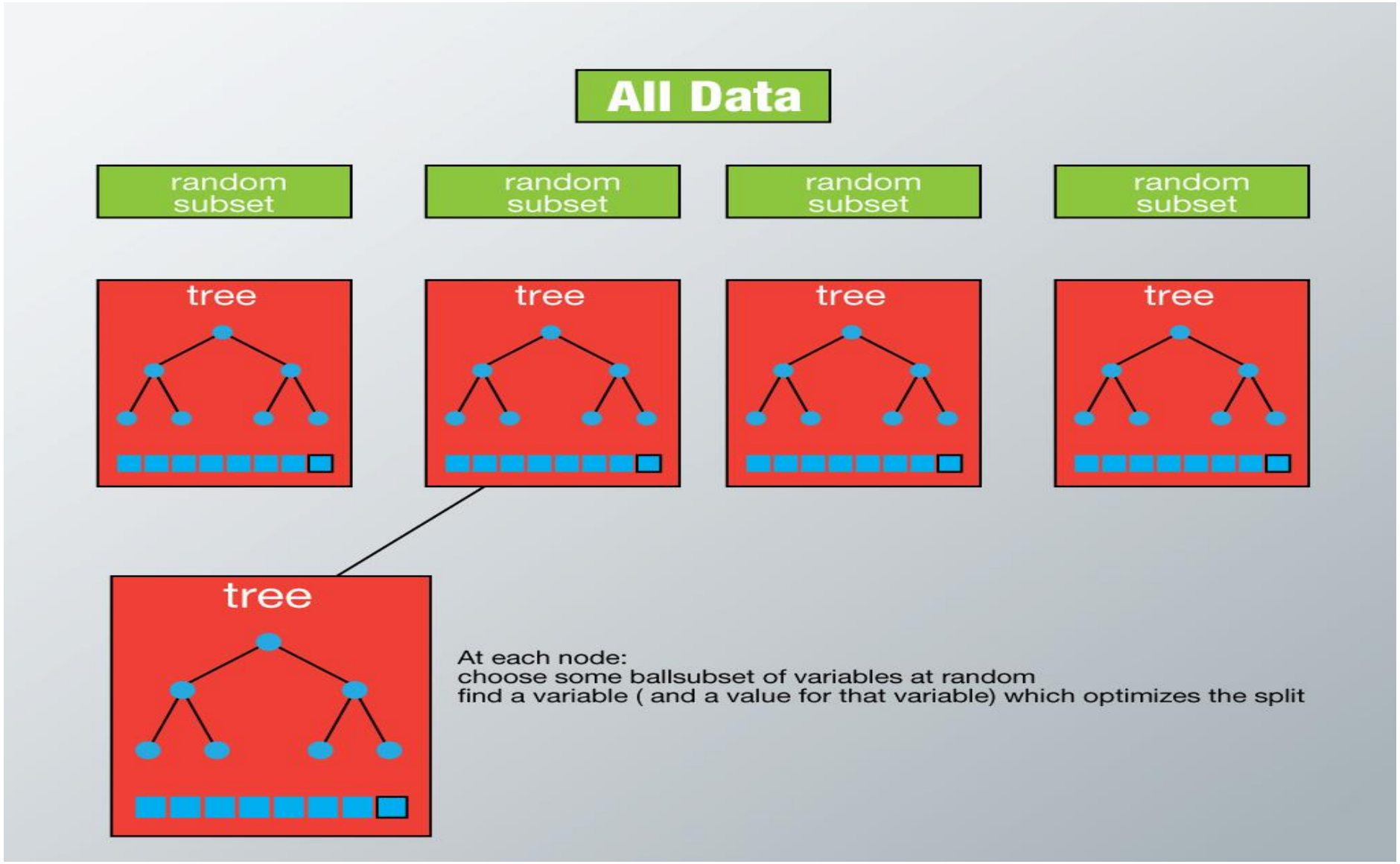
•If there are M input variables, a number m<M is specified such that at each node, m variables are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.

•Each tree is grown to the largest extent possible and there is no pruning.
Predict new data by aggregating the predictions of the ntree trees (i.e., majority votes for classification, average for regression).

Depending upon the value of *m*, there are three slightly different systems:

- ✓Random splitter selection: *m* =1
- ✓Breiman's bagger: *m* = total number of predictor variables
- ✓Random forest: *m* << number of predictor variables. Brieman suggests three possible values for m: ½√*m*, √*m*, and 2√*m*

Visual representation of the working of Random Forest Algorithm:



Advantages of Random Forest

- The same **random forest algorithm** or the random forest classifier can use for both classification and the regression task.

- Random forest classifier will **handle the missing** values.

- It doesn't **overfit** the model.

- Can model the random forest classifier for **categorical values** also.

CASE STUDY

Other Classification Models

Besides, Decision Tree and Random forest, we have a few more Classification Algorithms.

- NAIVE BAYES
- *K-NEAREST NEIGHBORS*

They are also supervised learning algorithms

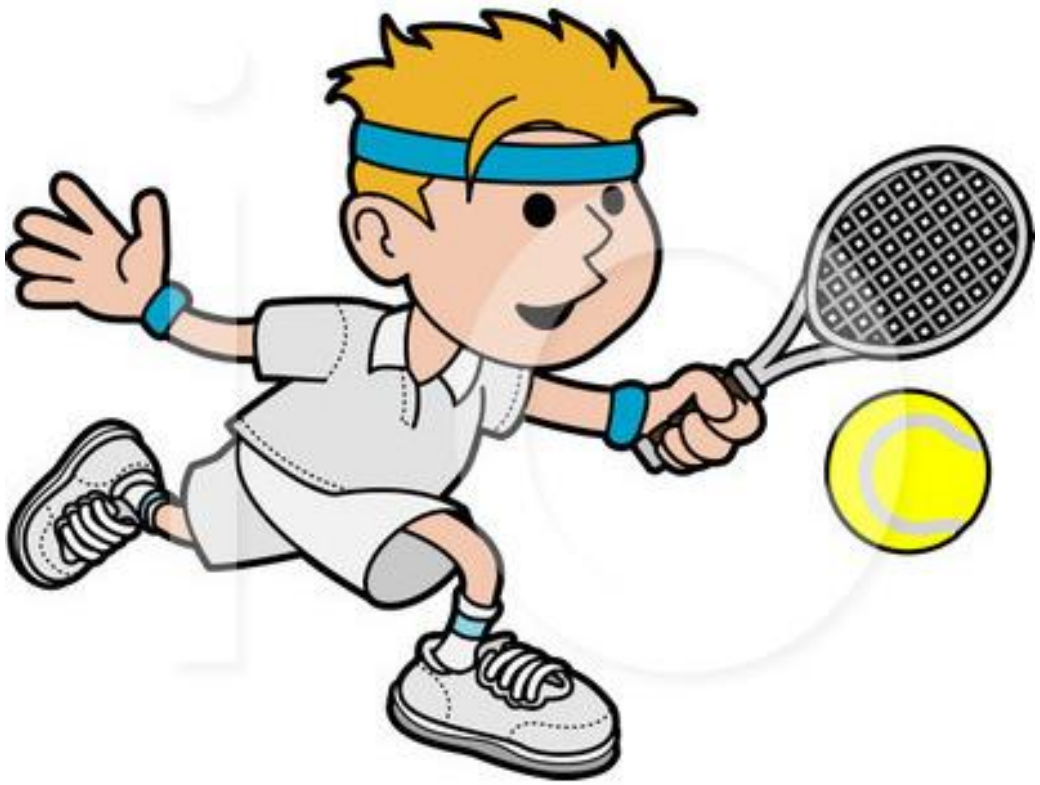
Naïve Bayes

Example:
Consider, John has to decide that whether or not to play ***Tennis*** today. He won't play if its too sunny or humid since he will get tired soon. Also, he would definitely prefer to play if it cool.

Hence, Some deciding factors are as listed below:

- Outlook (sunny/overcast/rainy)
- Temperature (hot/cool/mild)
- Humidity (High/normal)
- Windy (Yes/No)

Depending on these factors, we can Figure out that John will play if its Sunny? Or Sunny and Windy both? Etc..Lets try to solve this with Naïve Bayes Algorithm!



Understanding Naïve Bayes Algorithm

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. The algorithm learns the probability of an object with certain features belonging to a particular group/class. The class with the highest probability is considered as the most likely class.

In short, it is a probabilistic classifier.

The Naive Bayes algorithm is called "naive" because it makes the assumption that the occurrence of a certain feature is independent of the occurrence of other features.

For instance, if we are trying to identify a fruit based on its color, shape, and taste, then an orange colored, spherical, and tangy fruit would most likely be an orange. Even if these features depend on each other or on the presence of the other features, all of these properties individually contribute to the probability that this fruit is an orange and that is why it is known as "naive."

Bayes Theorem … Revisited

Bayes' law gives us a method to calculate the conditional probability, i.e., the probability of an event based on previous knowledge available on the events. More formally, Bayes' Theorem is stated as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Let us understand the statement first and then we will look at the proof of the statement. The components of the above statement are:

P(A|B): Probability (conditional probability) of occurrence of event A given the event B is true
P(A) and P(B): Probabilities of the occurrence of event A and B respectively
P(B|A): Probability of the occurrence of event B given the event A is true

Bayes Theorem … Example

Suppose you have to draw a single card from a standard deck of 52 cards. Now the probability that the card is a Queen is P(Q) = 4/52 = 1/13.
If you are given evidence that the card that you have picked is a face card, the posterior probability P(Q|FC) can be calculated using Bayes' Theorem as follows:

$$P(\text{Queen}|\text{Face}) = \frac{P(\text{Face}|\text{Queen})}{P(\text{Face})}.P(\text{Queen})$$

P(FC|Q) = 1, because given the card is Queen, it is definitely a face card.
We know the probability of the Queen card.
Now, P(FC) = 12/52 = 3/13, as there are three face cards for every suit in a deck.

Hence,

$$P(Q|FC) = \frac{P(FC|Q) * P(Q)}{P(FC)} = 1 * 1/13 * 13/3 = 1/3$$

How Naive Bayes algorithm works?

Let’s understand it using our Tennis example!

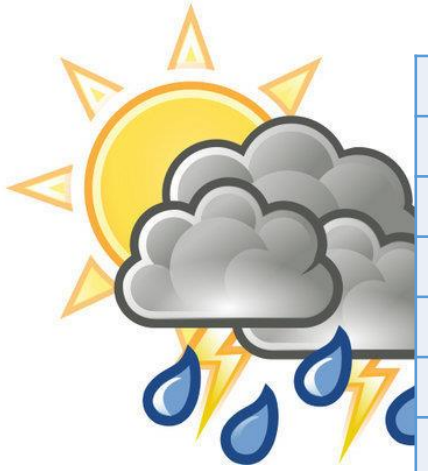
We have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether John will play or not based on weather condition.

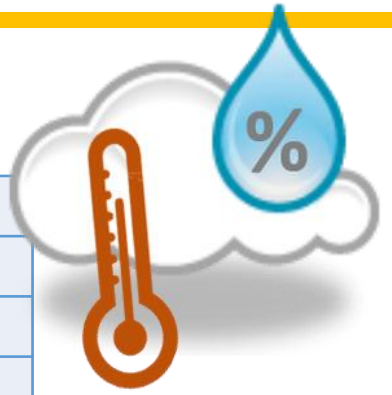
We will use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

The terminology in the Bayesian method of probability (more commonly used) is as follows:

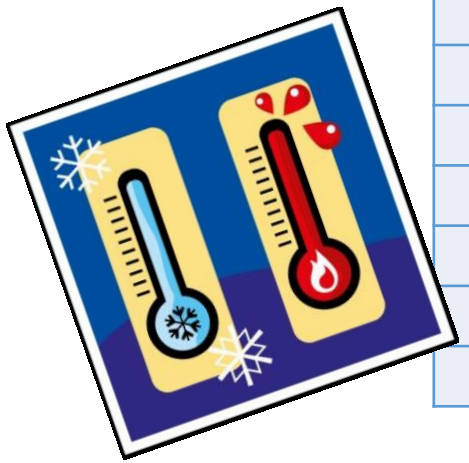
- *A is called the proposition and B is called the evidence.*
- *P(A) is called the prior probability of proposition and P(B) is called the prior probability of evidence.*
- *P(A|B) is called the posterior.*
- *P(B|A) is the likelihood.*

The data below gives us the circumstances of 14 days:





#	Outlook	Temperature	Humidity	Windy	Play
1	sunny	hot	high	no	no
2	sunny	hot	high	yes	no
3	overcast	hot	high	no	yes
4	rainy	mild	high	no	yes
5	rainy	cool	normal	no	yes
6	rainy	cool	normal	yes	no
7	overcast	cool	normal	yes	yes
8	sunny	mild	high	no	no
9	sunny	cool	normal	no	yes
10	rainy	mild	normal	no	yes
11	sunny	mild	normal	yes	yes
12	overcast	mild	high	yes	yes
13	overcast	hot	normal	no	yes
14	rainy	mild	high	yes	no



The probability of playing "**Yes**" = 9/14, and the probability of playing "**No**" = 5/14

Let us find out whether John will play if its Sunny, Temperature is cool, Humidity is high and its Windy!
So first lets calculate individual probabilities w.r.t . each features that is weather conditions.

Outlook	Play Yes	Play No	P(Yes)	P(No)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0
Rainy	3	2	3/9	2/5
Total	9	5	1	1

Temperature	Play Yes	Play No	P(Yes)	P(No)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	1	1

Humidity	Play Yes	Play No	P(Yes)	P(No)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	1	1

Wind	Play Yes	Play No	P(Yes)	P(No)
No	6	2	6/9	2/5
Yes	3	3	3/9	3/5
Total	9	5	1	1

Play	#	Prob
No	5	5/14
Yes	9	9/14
Total	14	1

Probability that John **can** play the game is

- P(Outlook = Sunny|Play = Yes) = 2/9
- P(Temp = Cool|Play = Yes) = 3/9
- P(Humidity = High|Play = Yes) = 3/9
- P(Wind = Yes|Play = Yes) = 3/9
- P(Play = Yes) = 9/14

Probability that John **cannot** play the game is

- P(Outlook = Sunny|Play = No) = 3/5
- P(Temp= Cool|Play = No) = 1/5
- P(Humidity = High|Play = No) = 4/5
- P(Wind = Yes|Play = No) = 3/5
- P(Play = No) = 5/14

So basically to find out whether John will play if its sunny, temperature is cool, humidity is high and its windy, we need to find the following probability:
P(Play = Yes | Sunny, Cool, High Humidity, Windy)

Applying Bayes Theorem, we get
(Using S: Sunny, C:Cool Temp, H:High Humidity, W:Windy)

$$P(\text{Play} = \text{Yes} \mid S,C,H,W) = \frac{P(S,C,H,W \mid \text{Yes}) * P(\text{Yes})}{P(S,C,H,W)}$$

$$= \frac{(2/9 * 3/9 * 3/9 * 3/9) * 9/14}{5/14 * 4/14 * 7/14 * 6/14} = 0.0053/\ 0.02186 = 0.2424$$

Hence the probability that John will play Tennis play in the mentioned weather conditions is 0.2424.

Similarly, we can find out the probability that John will **NOT** play if its sunny, temperature is cool, humidity is high and its windy, we need to find the following probability:
P(Play = No| Sunny, Cool, High Humidity, Windy)

Applying Bayes Theorem, we get
(Using S: Sunny, C:Cool Temp, H:High Humidity, W:Windy)

$$P(\text{Play} = \text{No} \mid S,C,H,W) = \frac{P(S,C,H,W \mid \text{No}) * P(\text{No})}{P(S,C,H,W)}$$

$$= \frac{(3/5 * 1/5 * 4/5 * 3/5) * 5/14}{5/14 * 4/14 * 7/14 * 6/14} = 0.0206/\ 0.02186 = 0.9421$$

Hence the probability that John will NOT play Tennis play in the mentioned weather conditions is 0.9421 which is higher.

In the example discussed, we can see that the probability of NOT playing Tennis (0.9421) is higher that playing tennis(0.2424) in given weather conditions. Hence, we would classify that data point under Play = NO.

Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

Advantages of Naïve Bayes Algorithm:

•It is easy and fast to predict class of test data set. It also perform well in multi class prediction

•When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.

•It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Applications of Naïve Bayes

The Naive Bayes algorithm is used in multiple real-life scenarios such as

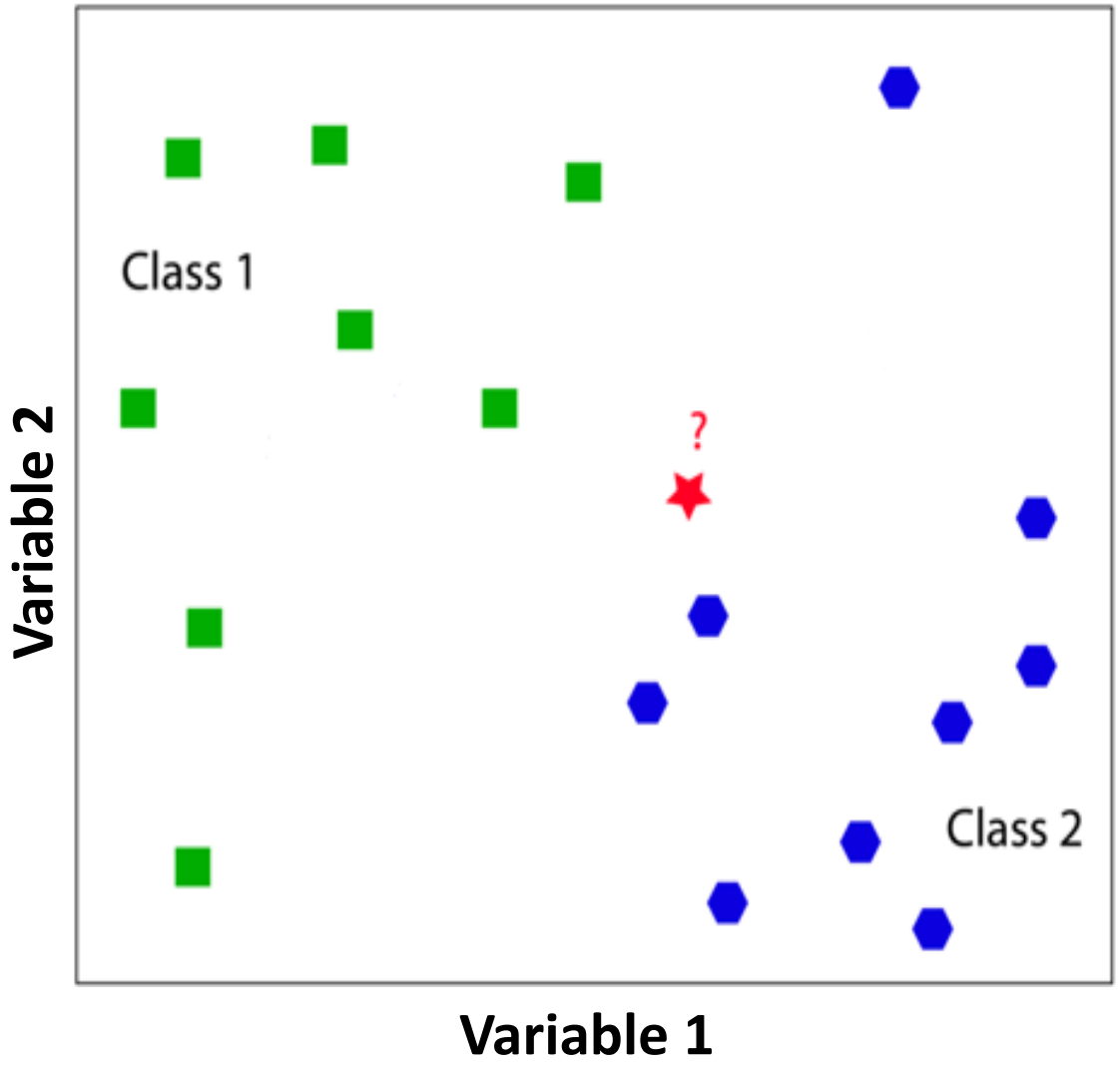
Text classification: The Naive Bayes classifier is one of the most successful known algorithms when it comes to the classification of text documents, i.e., whether a text document belongs to one or more categories (classes).

Spam filtration: This has become a popular mechanism to distinguish spam email from legitimate email. Several modern email services implement Bayesian spam filtering.

Sentiment Analysis: It can be used to analyze the tone of tweets, comments, and reviews—whether they are negative, positive or neutral.

Recommendation System: The Naive Bayes algorithm in combination with collaborative filtering is used to build hybrid recommendation systems which help in predicting if a user would like a given resource or not.

K Nearest Neighbors – kNN Algorithm

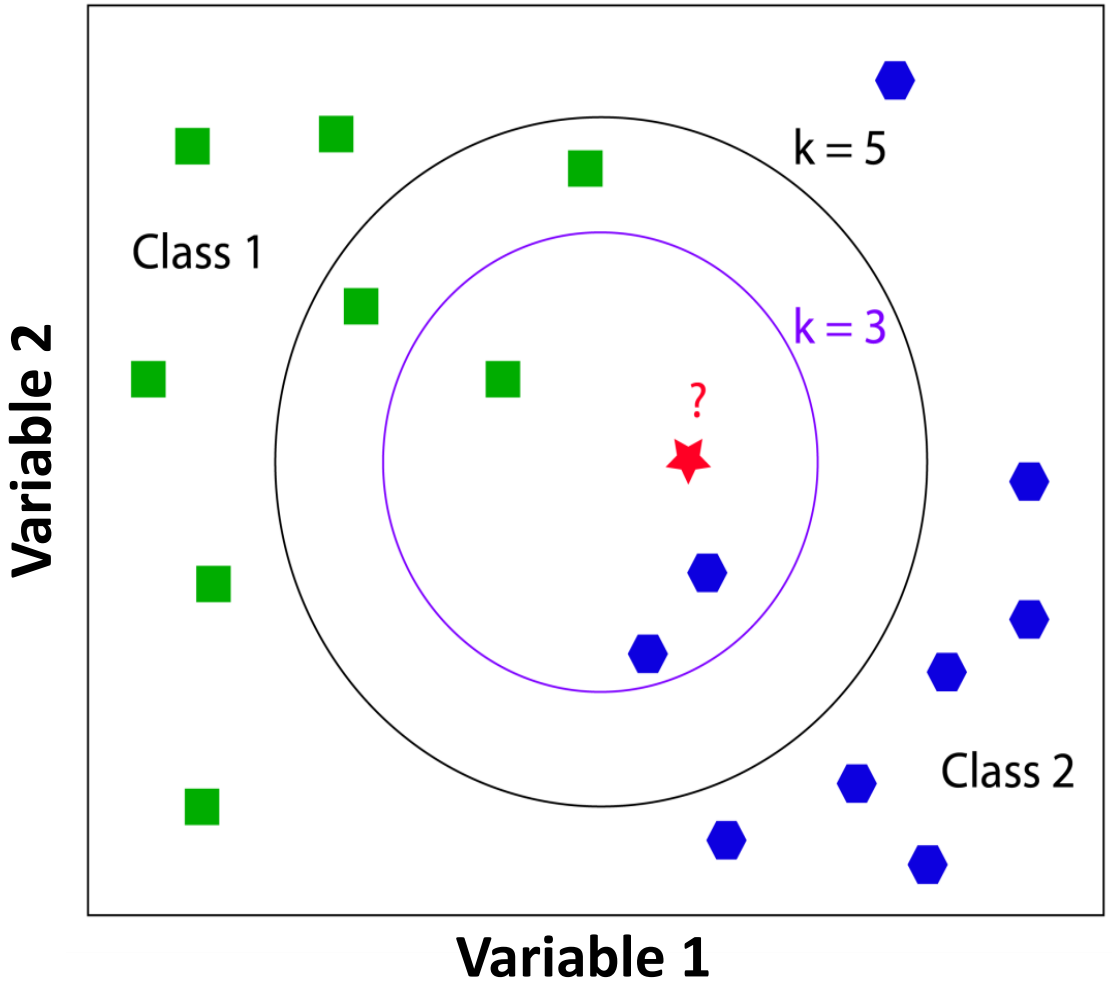


Here, we observe that
Green squares represent '*Class 1*' and
Blue hexagons represent '*Class 2*'.

The **Red star** is a new data point in the area

Problem statement: To classify the Red Star in either Class 1 or Class 2. We can use k Nearest Neighbors(kNN) Algorithm to classify the Red star.

K Nearest Neighbors – kNN Algorithm



In this example, the test sample, Red Star (★) should be classified either to the
Class 1 of Green squares (■) or to Class 2 of Blue hexagons (⬢).

If k = 3 (solid purple line circle) it is assigned to the Class 2 because there are 2 hexagons and only 1 square inside the inner circle.

If k = 5 (solid black line circle) it is assigned to the Class 1(3 squares vs. 2 hexagons inside the outer circle).

KNN Algorithm is based on **feature similarity**: How closely out-of-sample(test sample) features resemble our training set determines how we classify a given data point.

kNN Algorithm

- **KNN** is a **non-parametric, lazy** learning algorithm.
  - The technique is **non-parametric** , it means that it does not make any assumptions on the underlying data distribution
  - The technique is **lazy** since it does not use the training data points to do any *generalization*. To be more exact, all (or most) the training data is needed during the testing phase

- Its purpose is to use a database in which the data points are separated into several classes to predict the classification of a new sample point.

- The KNN algorithm is a robust and versatile classifier

- 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.

How does kNN work?

•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. Other measures used are Manhattan, Chebyshev, Minkowski, Hamming distance.

•Selection of K

➢The K in KNN is a hyper parameter that must be picked in order to get the best possible fit for the data set.

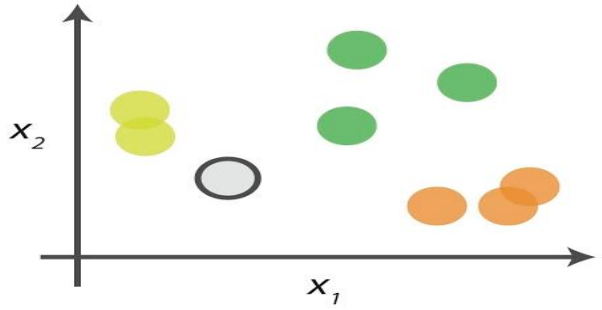
➢A small value for K provides the most flexible fit, which will have low bias but high variance.

➢Larger values of K will have smoother decision boundaries which means lower variance but increased bias.

➢Usually it is taken as a square-root of the sample size(n), but it is not a restrictive guideline

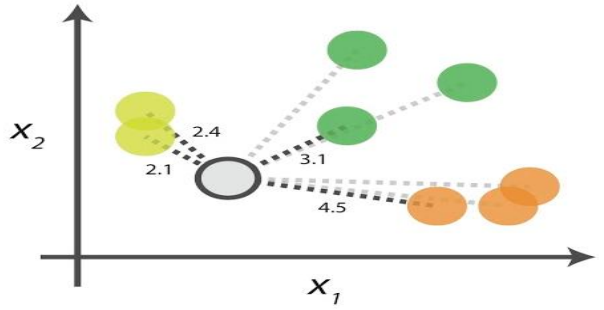
kNN Algorithm - Working

0. Look at the data



Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances

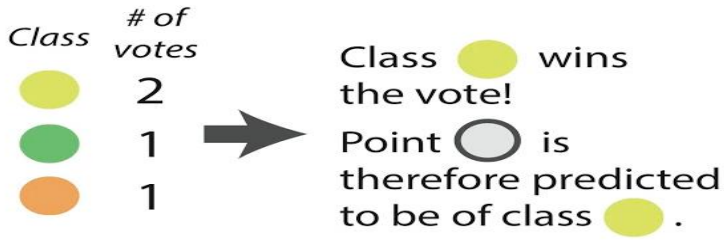


Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels



Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Thank You