Rensselaer

why not change the world?®

Naïve Bayes, KNN, KMeans & Decision Trees

Ahmed Eleish

Data Analytics ITWS-4600/ITWS-6600/MATP-4450/CSCI-4960

January 28th , 2025

Tetherless World Constellation
Rensselaer Polytechnic Institute

Contents

- Naïve Bayes
- Decision Trees
- k-Nearest Neighbors
- k-Means Clustering

Naïve Bayes – what is it?

- Components of the Baye's Rule
- What does it mean by Naïve
- Naïve Bayes Classifier

Naïve Bayes

• Naïve Bayes model uses a **probabilistic approach to do a classification**.
•In Naïve Bayes, the relationship between the input features and the classes are defined using the probabilities.
• Before diving into Naïve Bayes, you should understand the basic Probability Theories/Concepts.
• Probability is the measure of the likelihood that an event will occur.

https://en.wikipedia.org/wiki/Naive_Bayes_classifier
https://en.wikipedia.org/wiki/Probability

Naïve Bayes

- The class with the highest probability value determines the label for the sample.

- Naïve Bayes classifier uses the Bayes Theorem by Thomas Bayes.

https://en.wikipedia.org/wiki/Naive_Bayes_classifier
Image/Photo Credit: https://en.wikipedia.org/wiki/Thomas_Bayes

Naïve Bayes

- In Naïve Bayes, **we assume that input features are statistically independent from each other**, which means, for a given class the value of one particular feature doesn't affect the value of another feature.

- **In reality, this assumption is little bit oversimplified and not true most of the time, because of that we say this assumption is *naïve***

https://en.wikipedia.org/wiki/Naive_Bayes_classifier
https://en.wikipedia.org/wiki/Bayes%27_theorem

Probability ...

- Lets go over some definitions in probability.

- Probability is the measure of the likelihood that an event will occur.

- *Probability of event **A***:

$$P(A) = \frac{Number\ of\ ways\ for\ A}{Total\ number\ of\ possible\ outcomes}$$

Reference: https://en.wikipedia.org/wiki/Probability

Probability ...

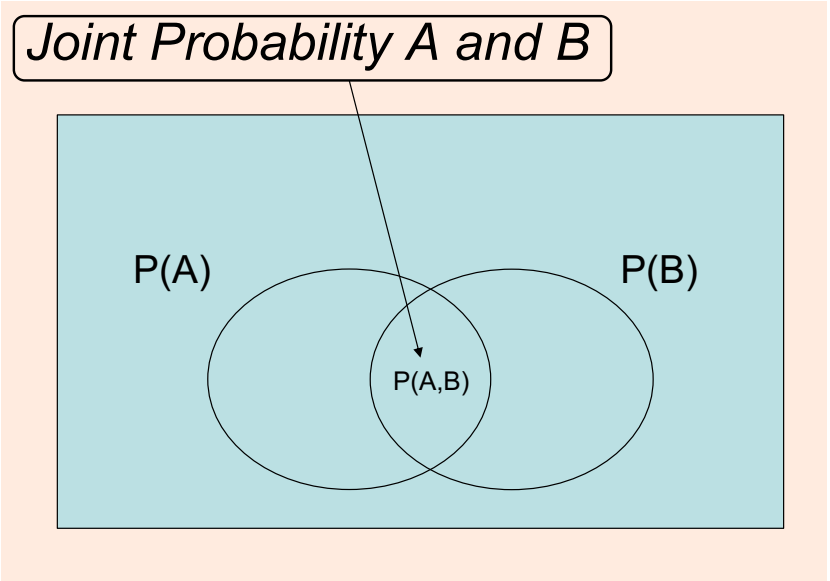
• Before dive into Naïve Bayes, You should know/understand the two probability concepts:

1) Joint Probability
2) Conditional Probability

Reference: https://en.wikipedia.org/wiki/Probability

Probability

Joint Probability: *specifies the probability of event A and event B occurring together*.



https://en.wikipedia.org/wiki/Joint_probability_distribution

Probability

Joint Probability: *specifies the probability of event A and event B occurring together*.

If the two events are independent,

What is the probability of getting two 6's when you roll two dice?

The probability of rolling(getting) two 6's:

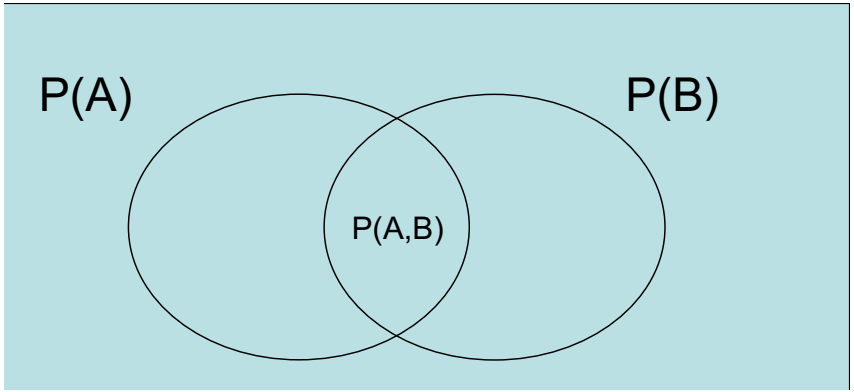
$$P(A,B) = P(A) * P(B) = \frac{1}{6} * \frac{1}{6} = \frac{1}{36}$$

https://en.wikipedia.org/wiki/Joint_probability_distribution Image/Photo Credit: https://pngtree.com/freepng/two-dice_1504759.html

Probability

Conditional Probability: *probability of event A occurring, given that event B occurred.*

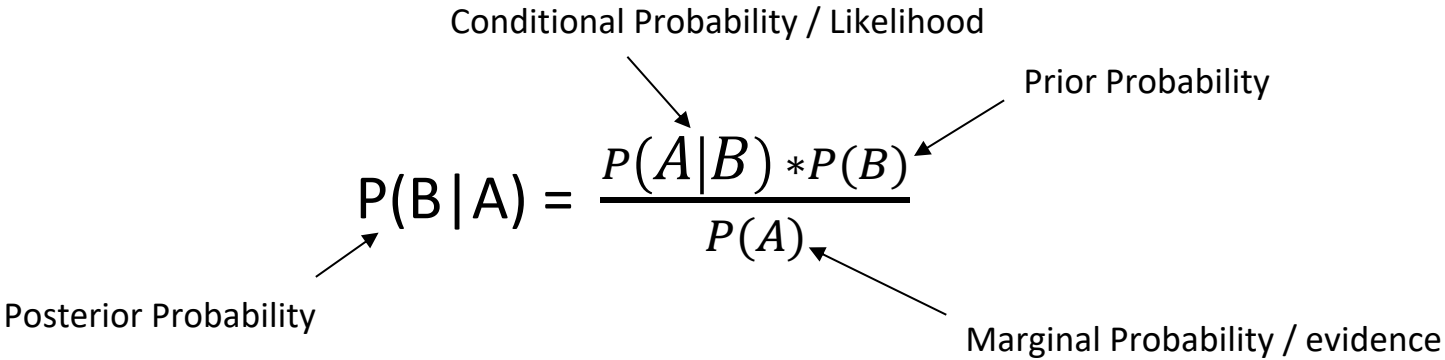
$P(A|B) = \frac{P(A,B)}{P(B)}$ = *Probability of A, given B ; P(B)>0*



https://en.wikipedia.org/wiki/Conditional_probability

Bayes Theorem

- The relationship between conditional probabilities, P(B|A) and P(A|B) can be expressed using the Bayes Theorem.



~ Probability of cause given effect

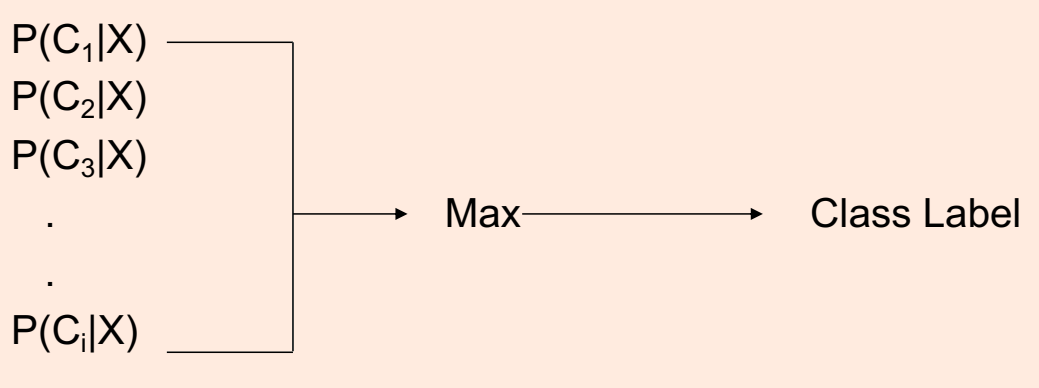
Reference: https://en.wikipedia.org/wiki/Bayes%27_theorem

Classification using Probabilities

- With given features $X = (x_1, x_2, x_3, \ldots, x_k)$

Predict the class $C = (C_1, C_2, C_3, \ldots, C_i)$

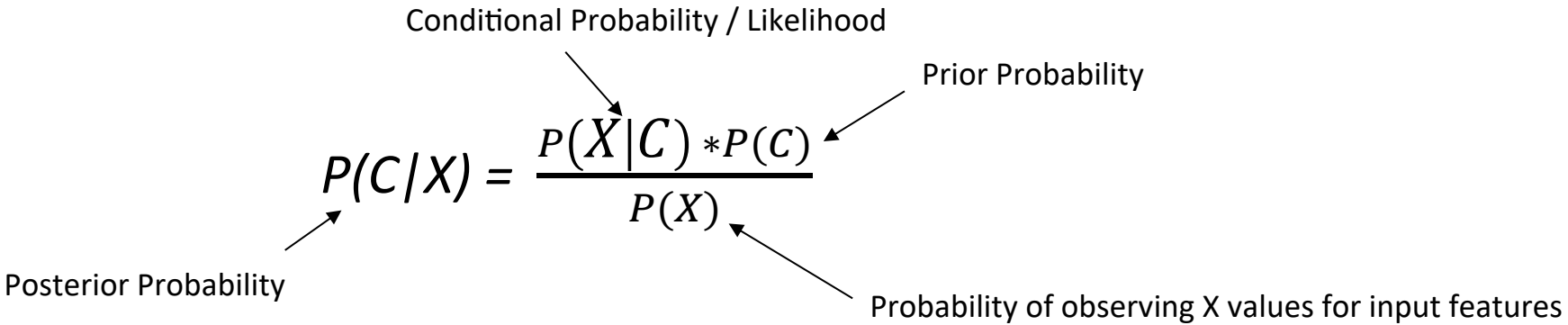
- We can do this by finding the value of C that, maximize the P(C|X)



Classification using Probabilities

- In order to find the class label C, we need to find the conditional probability of class C given X for all classes and choose the class that has the highest probability.

- It is difficult to estimate the P(C|X) so we use Bayes theorem to simplify this problem.



Bayes Theorem in Classification

These can be estimated from the training sample data

$$P(C|X) = \frac{P(X|C) * P(C)}{P(X)}$$

This is what we are looking for

It is a constant, and it can be Ignored (probability of X doesn't depend on the class C, since it is same for all classes, it can be removed from the calculation of probability of C given X)

To find P(C|X) we only need to find the P(X|C) and P(C) which can be estimated from the data

Bayes Theorem in Classification

- $P(x_k \mid C_i)$ is estimated from the training samples

– Categorical: Estimate $P(x_k \mid C_i)$ as percentage of samples of class i with value x_k
  - Training involves counting percentage of occurrence of each possible value for each class

– Numeric: Actual form of density function is generally not known, so "normal" density (i.e. distribution) is often assumed

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

https://www.seas.upenn.edu/~cis520/papers/Bishop_2.3.pdf

Exercise: Naïve Bayes with iris

```
## Call the NaiveBayes Classifier Package e1071, which auto calls the Class package ##
library("e1071")
classifier<-naiveBayes(iris[,1:4], iris[,5])
table(predict(classifier, iris[,-5]), iris[,5], dnn=list('predicted','actual'))
classifier$apriori
classifier$tables$Petal.Length
```

mean Standard Deviation

```
plot(function(x) dnorm(x, 1.462, 0.1736640), 0, 8, col="red", main="Petal length distribution for the 3 different species")

curve(dnorm(x, 4.260, 0.4699110), add=TRUE, col="blue")
curve(dnorm(x, 5.552, 0.5518947 ), add=TRUE, col = "green")
```

Decision Trees

Decision trees can be applied to both regression and classification problems

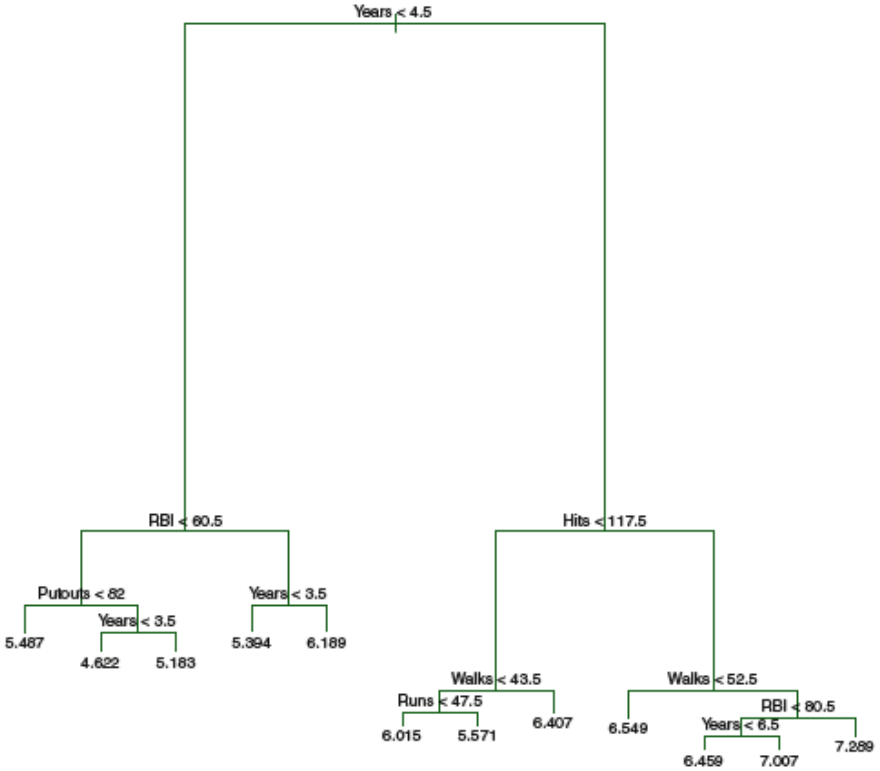


FIGURE 8.4. *Regression tree analysis for the* `Hitters` *data. The unpruned tree that results from top-down greedy splitting on the training data is shown.*

Reference/Resources: Introduction to Statistical Learning with R -7 Edition: Chapter 8

Advantages and Disadvantages of Decision Trees

Decision trees for regression and classification have number of advantages and disadvantages

Advantages:

- Trees are very easy to explain to people, in fact, they are even easier to explain than linear regression.
- Some people believe that decision trees are more closely mirror human decision-making than other regression and classification techniques.
- Trees can be displayed graphically, and easily interpreted even non-experts (especially if the tree is small) can understand.

Reference/Resources: Introduction to Statistical Learning with R -7 Edition: Chapter 8

Advantages and Disadvantages of Decision Trees

Decision trees for regression and classification have number of advantages and disadvantages

Disadvantages:

• Unfortunately, trees generally do not have the same level of predictive accuracy as some of the other regression and classification approaches

• Additionally, trees can be very non-robust. In other words, a small changes in the data can cause a large change in the final estimated tree

• However, by aggregating many decision trees, using methods like *bagging*, *random forest* and *boosting*, the predictive performance of trees can be substantially improved.

Decision Tree – Classification

• When we implement decision trees for classification, the idea is to split the data into subsets. So that each subset belongs to one particular class.

• In other words, splitting the data into regions, that are separated by decision boundaries, where each region's samples have only one class.

Decision Tree

• Decision Tree has a hierarchical structure with “nodes” and “directed edges”

• The top node is defined as the “root node” and the nodes at the bottom are called as “leaf nodes”

• Nodes that are neither root node nor the leaf nodes are identified as “internal nodes” in the decision tree.

• There is a “class label” associated with each leaf node

Decision Tree

- Classification decisions are made by traversing the decision tree

- Traversing starts from the root node (from the top of the tree).

- The root node and the internal nodes have test conditions. Those test conditions determine which path to traverse on the tree.

Decision Tree

- **When a leaf node is reached through traversing, the category of the leaf node determines the classification**.

- The **depth is measured from the root node** and the **depth at the root node is zero**.

- **The depth of the decision tree**: Tree Depth is calculated by counting the number of edges in the longest path from the root node to a leaf node.

Decision Tree

• Number of nodes in the decision tree determine the size of the tree.

• **The decision tree constructing algorithm is referred to as a tree induction algorithm**.

Impurity Measure

• The goal is to have the resulting subsets to be homogeneous as possible and minimize the impurity.

• In practice, we don’t get pure homogeneous subsets, there are impurities.

• A **common impurity measure to determining the best split is “Gini-index”**

• The **lower the Gini-index value, the higher the purity of the split**.

Impurity Measure

• The decision tree will select the split that minimize the Gini-index.

• Besides the Gini-index, there are other impurity measures available such as:

- entropy or information gain
- misclassification rate

• The decision tree will test all variables to determine the best way to split a node using a purity measure such as Gini-index to compare different possibilities

• **Tree induction algorithms repeatedly split nodes to get more and more homogeneous subsets**.

• When this process stops? When does the algorithm stop growing the tree?

When to Stop splitting the nodes?

• There are several criteria that can be used to determine the when a node shouldn't be split into subsets.

• The **induction algorithm can stop expanding a node when all samples in the node have the same class label**.

• Since getting pure subsets is difficult to archive with real world data, **the stopping criteria can be modified to archive certain percentage of the samples in the node. i.e 95% of have the same class label.**

Stopping criteria

• Algorithm can stop expanding a node when the number of samples in the node falls below a certain minimum number.

• Induction algorithm can stop expanding a node when the improvement in impurity measure is way too small to measure (too small to make a much difference in classification result).

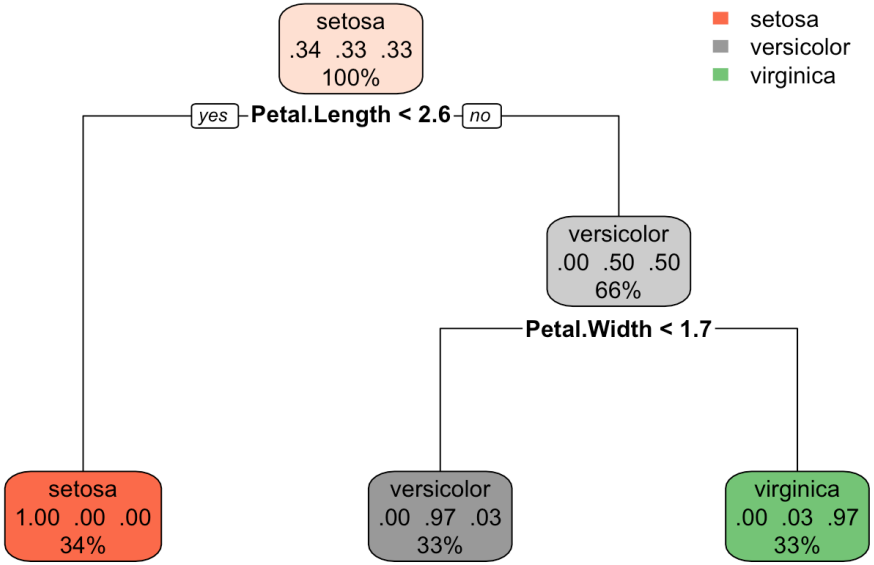
• Also, algorithm can stop expanding when it reach the maximum of the tree-depth.

Exercise: Random Forest

```
# Decision Trees --
# Classification Tree
# Install the following libararies/packages in RStudio library(rpart)
library(rpart.plot)

# we will be using the iris dataset
iris
dim(iris)
# creating a sample from the iris dataset
s_iris <- sample(150,100)
s_iris

# creat testing and training sets
iris_train <-iris[s_iris,]
iris_test <-iris[-s_iris,]
dim(iris_test)
dim(iris_train)
```

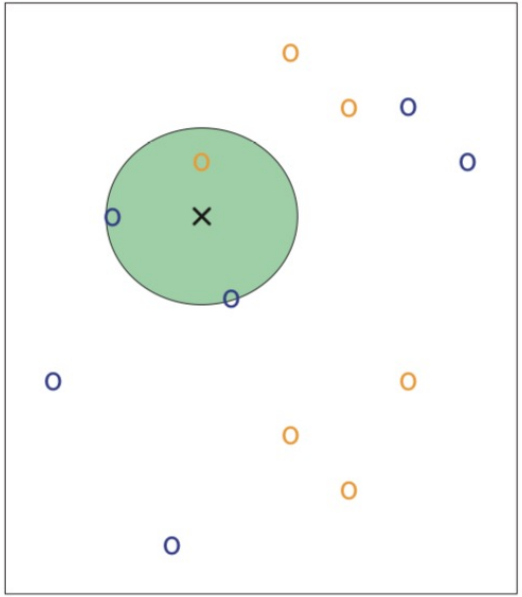
Exercise: Random Forest

```
# generate the decision tree model
dectionTreeModel <- rpart(Species~., iris_train,
method = "class")
dectionTreeModel

#plotting the decision tree model using
#rpart.plot() function
rpart.plot(dectionTreeModel)
```



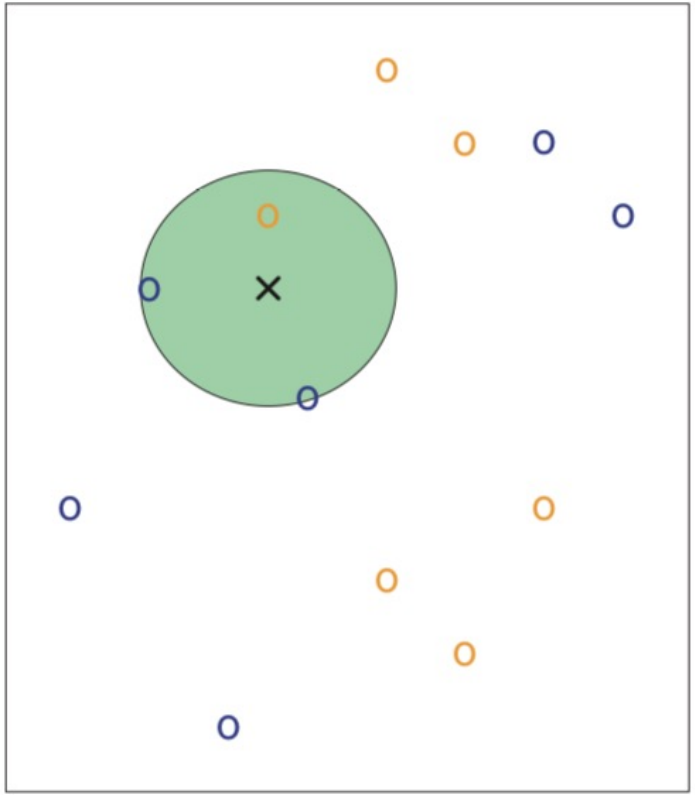
kNN Classifier

- In the figure a dataset is shown consisting 6 blue and 6 orange observations.
- Our goal is to make a prediction for the point labeled by the black cross.
- Suppose we choose K=3, then KNN will first identify the three observations that are closest to the black cross as shown in the figure.
- This neighborhood is shown as a circle. It consist of 2 blue points and 1 orange point, resulting in estimated probabilities of 2/3 for the blue class and 1/3 for the orange class.
- Hence, kNN will predict that the black cross belongs to the blue class.



1.Image/photo Credit: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 2
Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 2 – KNN Classifier

6 blue points and 6 orange points



1.Image/photo Credit: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 2
Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 2 – KNN Classifier

k-Means

• k-Means clustering is an unsupervised learning algorithm that, as the name hints, finds a fixed number (*k*) of clusters in a set of data.

• A *cluster* is a group of data points that are grouped together due to similarities in their features. When using a K-Means algorithm, a cluster is defined by a *centroid*, which is a point (either imaginary or real) at the center of a cluster.

• Every point in a data set is part of the cluster whose centroid is most closely located. To put it simply, K-Means finds *k* number of centroids, and then assigns all data points to the closest cluster, with the aim of keeping the centroids small

Resource: https://blog.easysol.net/machine-learning-algorithms-3/
https://blog.easysol.net/author/acorrea/

K-Means Algorithm

```
randomly chose k examples as initial centroids
while true:
    create k clusters by assigning each
        example to closest centroid
    compute k new centroids by averaging
        examples in each cluster
    if centroids don't change:
        break
```

Resource: MIT 6.0002 lecture 12 (MIT Open Courseware)
https://ocw.mit.edu/index.htm

Algorithm 10.1 *K-Means Clustering*

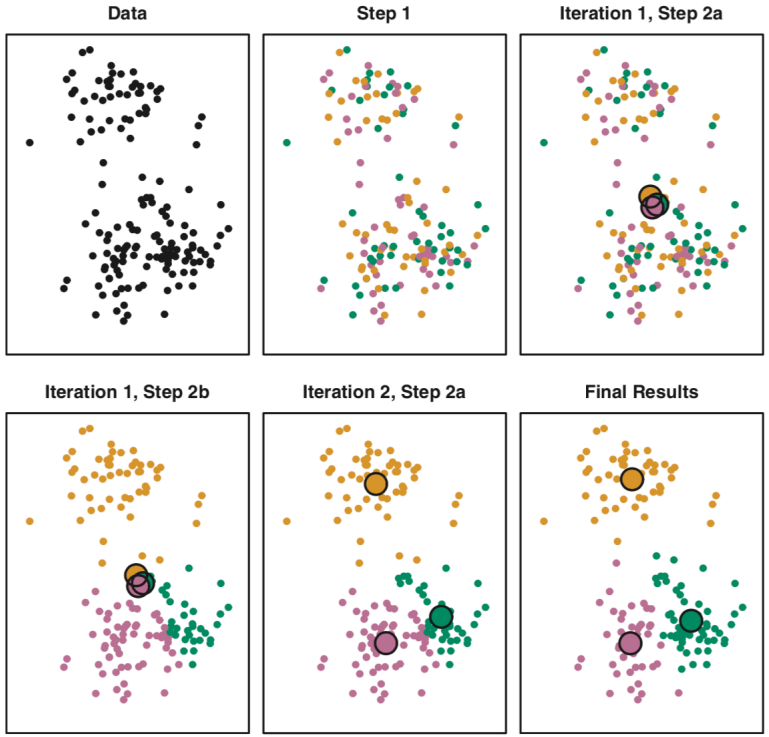
1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

K-Means Algorithm

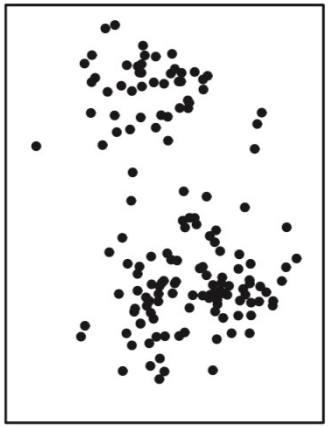
Algorithm 10.1 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.
   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

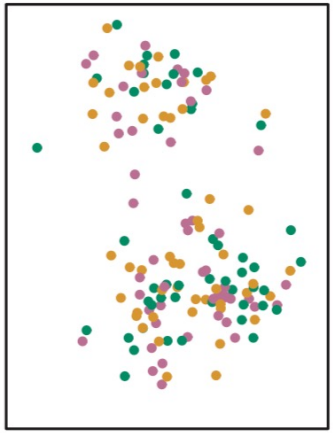


Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

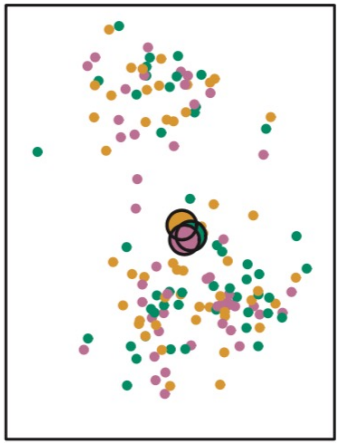
K-Means Algorithm



Observations (data) is shown



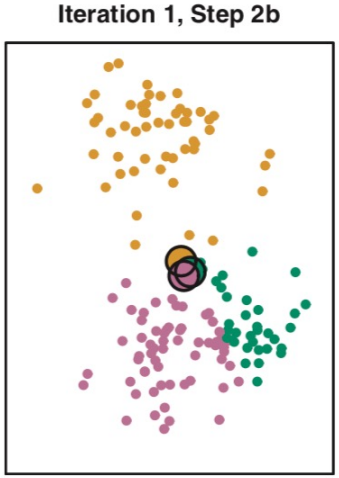
Step 1 of the algorithm: each observation is randomly assigned to a cluster



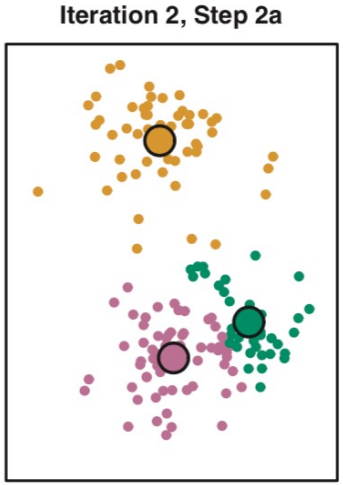
Iteration1 Step 2(a): The cluster centroids are computed; these are shown in large colored disks. Initially centroids are almost completely overlapping because the initial cluster assignment were chosen at random

Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

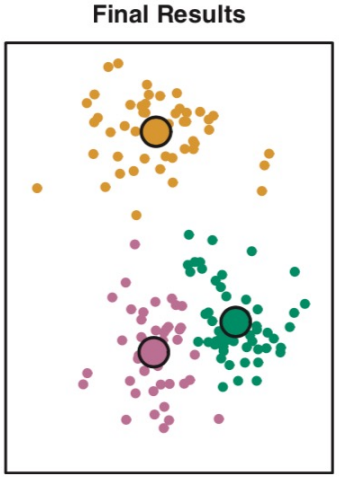
K-Means Algorithm



Iteration 1 Step 2(b) : each observation is assigned to the nearest centroid



Iteration 2, Step 2(a): the step 2(a) is once again performed, leading to new cluster centroids.



Final Results: the results obtained after ten iterations. You can see the distinct clusters with their centroids.

Image/Photo Credit: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans
Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

- K-Means clustering Animation
- http://shabal.in/visuals/kmeans/6.html

In-Class Exercise: KNN - Abalone

```
# abalone dataset from UCI repository
# reading the dataset from UCI repository URL
abalone <- read.csv(url("https://archive.ics.uci.edu/ml/machine-learning-
databases/abalone/abalone.data"), header = FALSE, sep = ",")
# Column names
colnames(abalone) <- c("sex", "length", 'diameter', 'height', 'whole_weight', 'shucked_wieght',
'viscera_wieght', 'shell_weight', 'rings' )
# summary on abalone
summary(abalone)
# structure of the abalone data
str(abalone)
# summary of the abalone rings column
summary(abalone$rings)
```

In-Class Exercise: KNN - Abalone

```
# As shown above, the “rings” variable has a range between 1-29.
# This is the variable that we want to predict, and predicting this many levels
# might not give us the insight we’re looking for.
# For now, we’ll break the rings variable
# into 3 levels" “young” for abalones less than 8, “adult” for abalones between 8-11,
# and “old” for abalones older than 11.
abalone$rings <- as.numeric(abalone$rings)
abalone$rings <- cut(abalone$rings, br=c(-1,8,11,35), labels = c("young", 'adult', 'old'))
abalone$rings <- as.factor(abalone$rings)
summary(abalone$rings)
# remove the "sex" variable in abalone, because KNN requires all numeric variables for prediction #
z <- abalone
aba <- abalone
aba$sex <- NULL
```

In-Class Exercise KNN – Abalone continue...

```
# normalize the data using min max normalization
normalize <- function(x) {return ((x - min(x)) / (max(x) - min(x))) }
aba[1:7] <- as.data.frame(lapply(aba[1:7], normalize))
summary(aba$shucked_wieght)
# After Normalization, each variable has a min of 0 and a max of 1.
# in other words, values are in the range from 0 to 1.
# We'll now split the data into training and testing sets.
ind <- sample(2, nrow(aba), replace=TRUE, prob=c(0.7, 0.3))
KNNtrain <- aba[ind==1,]
KNNtest <- aba[ind==2,]
sqrt(2918)
```

In-Class Exercise KNN – Abalone continue...

```
# make k equal to the square root of 2918, the number of observations in the training set.
# sqrt(2918) ~= 54.01852 round it to 55 and use k = 55
# We usually take an Odd number for k value,
# knn model
# knn() is in the "class" library. Make sure to install it first on your RStudio.
library(class)
help("knn") # Read the knn documentation on RStudio.
KNNpred <- knn(train = KNNtrain[1:7], test = KNNtest[1:7], cl = KNNtrain$rings, k = 55)
KNNpred
table(KNNpred)
```

In-Class Exercise K-Means Iris dataset

```
# iris dataset is from UCI ML repository.
library(ggplot2) # we will use ggplot2 to visualize the data.
head(iris) # first 6 rows of the
str(iris) # take a look at the structure of the iris data using str() function in R.
# dataset has 150 observations equally distributed observations among
# the three species: Setosa, Versicolor and Verginica.
summary(iris) # summary statistics of all the 4 variables Sepal.Length,Sepal.Width,
# Petal.Length and Petal.Width
help("sapply")
sapply(iris[,-5], var)
summary(iris)
```

In-Class Exercise K-Means Iris dataset

```
# plot Sepal.Length Vs Sepal.Width using ggplot
ggplot(iris,aes(x = Sepal.Length, y = Sepal.Width, col= Species)) + geom_point()
# plot Petal.Length Vs Sepal.Width using ggplot
ggplot(iris,aes(x = Petal.Length, y = Petal.Width, col= Species)) + geom_point()
# kmeans clustering
# Read the documentation for kmeans() function
# https://stat.ethz.ch/R-manual/R-devel/library/stats/html/kmeans.html
set.seed(300)
k.max <- 12
# tot.withinss = Total within-cluster sum of square
# iter.max = the maximum number of iterations allowed
# nstart = if centers is a number, how many random sets should be chosen.
wss<- sapply(1:k.max,function(k){kmeans(iris[,3:4],k,nstart = 20,iter.max = 20)$tot.withinss})
```

In-Class Exercise K-Means Iris dataset

wss # within sum of squares.
plot(1:k.max,wss, type= "b", xlab = "Number of clusters(k)", ylab = "Within cluster sum of squares") icluster <- kmeans(iris[,3:4],3,nstart = 20)
table(icluster$cluster,iris$Species)
# In the table we can see that most of the observations have been clustered correctly
# however, 2 of the versicolor have been put in the cluster with all the virginica
# and 4 of the verginica have been put in cluster 3 which mostly has versicolor.

In-Class Exercises on Trees

```
# Classification ctrees
# iris data set
# Install the following libararies/packages library(rpart)
library(rpart.plot)
# we will be using the iris dataset
iris
dim(iris) # check the dimensions of the iris dataset
# creating a sample from the iris dataset s_iris <- sample(150,100)
s_iris
# creat testing and training sets iris_train <-iris[s_iris,]
iris_test <-iris[-s_iris,] dim(iris_test)
dim(iris_train)
# generate the decision tree model
dectionTreeModel <- rpart(Species~., iris_train, method = "class") dectionTreeModel
#plotting the decision tree model using rpart.plot() function rpart.plot(dectionTreeModel)
```

Thanks!