Applied Data Science

UNIT-II

Basic Machine Learning Algorithms

- Machine learning algorithms are largely used to predict, classify, or cluster.

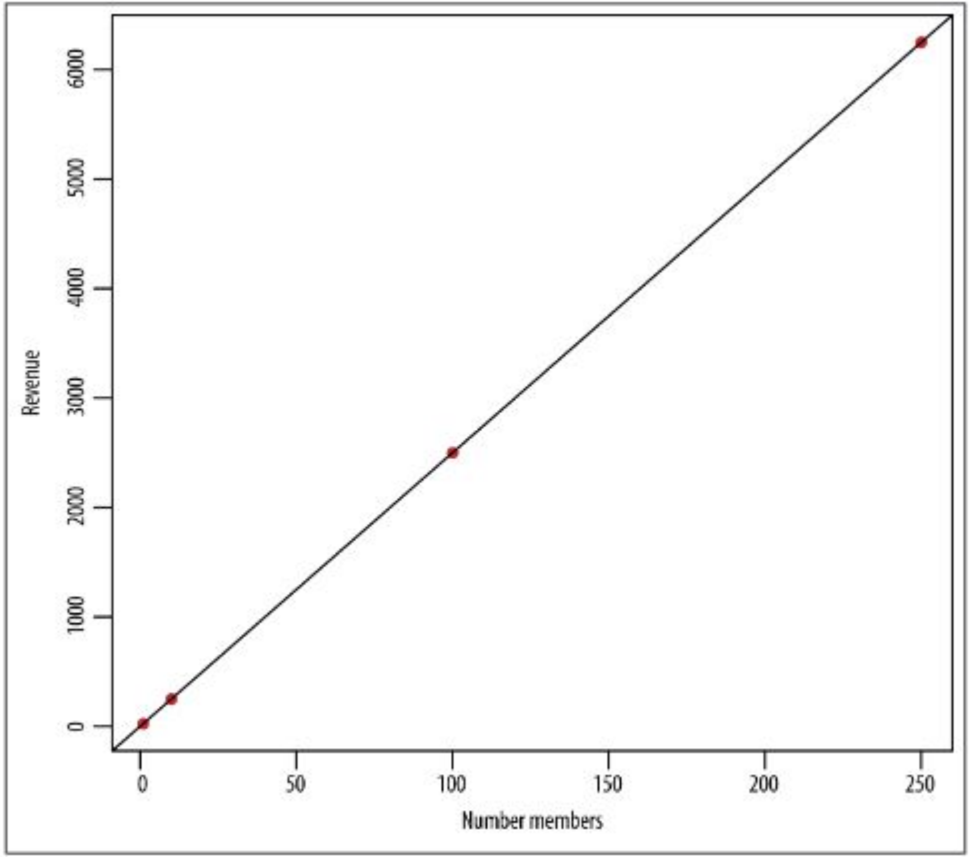
<u>Linear Regression</u>

- One of the most common statistical methods is linear regression.
- It is used when you want to express the mathematical relationship between two variables or attributes.
- The word ‘regression’ was used by **Sir Francis Galton** to describe the relationship between heights of parents and their children. Tall parents tend to have tall children, but shorter than themselves while short parents tend to have short children, but taller than themselves.
- We make an assumption that there is a linear relationship between an outcome variable (sometimes also called the response variable, dependent variable, or label) and a predictor (independent variable).
- Example: studying the relationship between a company’s sales and how much that company spends on advertising.
- Example: Number of friends someone has on a social networking site and the time that person spends on that site daily.

Linear Regression:Example

- Suppose you run a social networking site that charges a monthly subscription fee of $25, and that this is your only source of revenue. Each month you collect data and count your number of users and total revenue. You've done this daily over the course of two years, recording it all in a spreadsheet. You could express this data as a series of points. Here are the first four
- S = {(x, y) = (1,25) , (10,250) , (100,2500) , (200,5000)}
- There is a clear relationship between the data points, namely y=25x.
- There's a linear pattern.
- The coefficient relating x and y is 25.
- It seems deterministic (no randomness).

Linear Regression: Example



It is a linear pattern

Linear Regression: Example

- Suppose you have a dataset of users (meaning each row contains data for a single user), and the columns represent user behavior on a social networking site over a period of a week.
- The names of the columns are total_num_friends, total_new_friends_this_week, num_visits, time_spent, number_apps_downloaded, number_ads_shown, gender, age, and so on.
- Data is clean at this stage and that you have on the order of hundreds of thousands of users.
- During the exploratory data analysis, you've randomly sampled 100 users to keep it simple, and you plot pairs of these variables, for example, x = total_new_friends and y = time_spent (in seconds).

Linear Regression: Example

The business context is to promise advertisers who bid for space on your website in advance a certain number of users.

Examine the first few rows of the data.

7 276

3 43

4 82

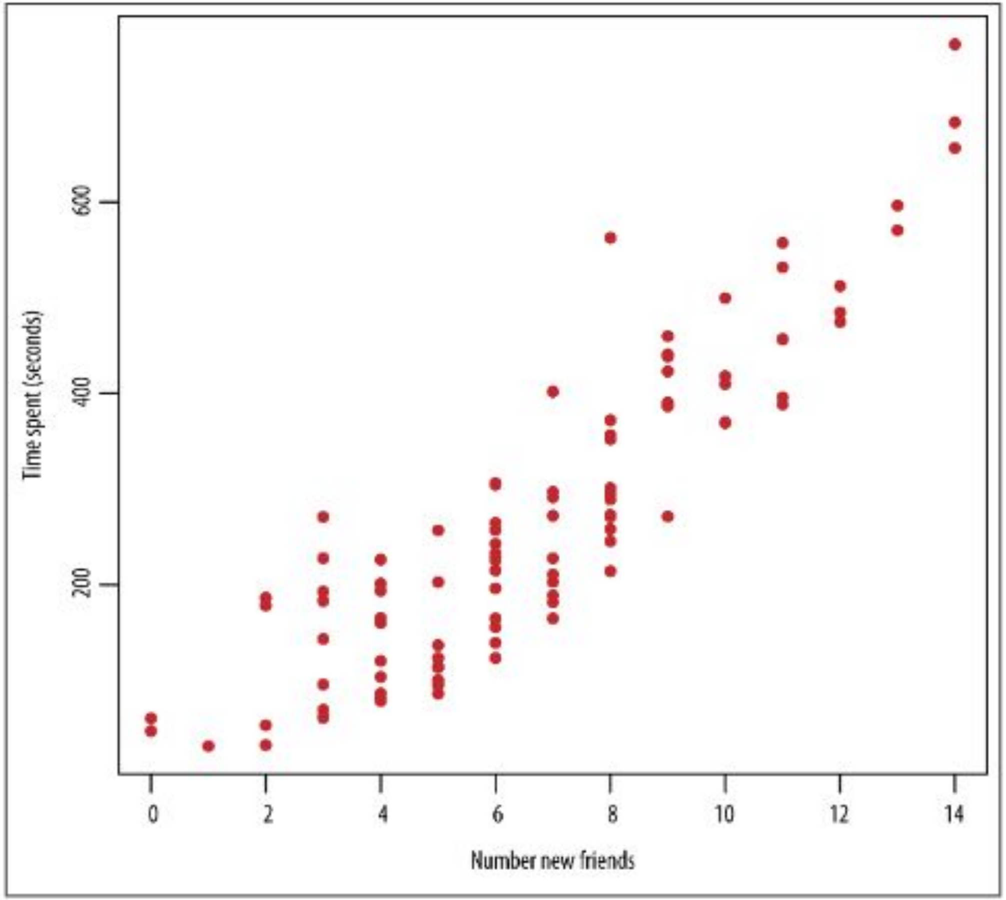
6 136

10 417

9 269

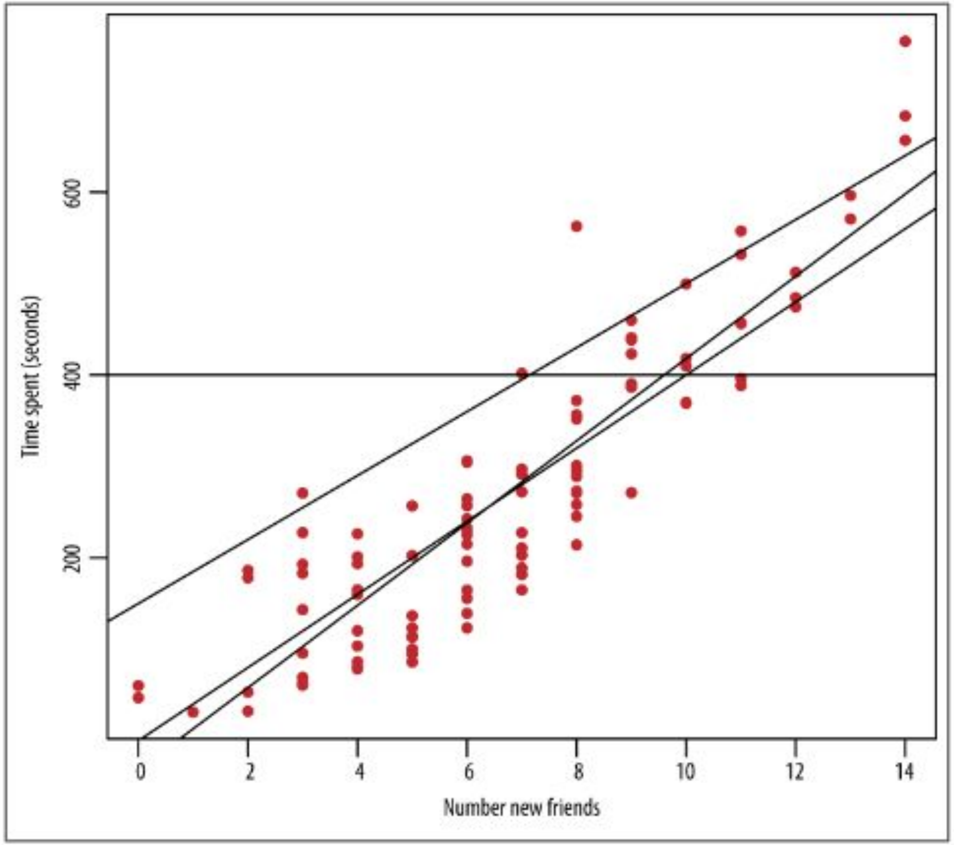
It is not possible to figure out the relationship between the data.

Linear Regression: Example



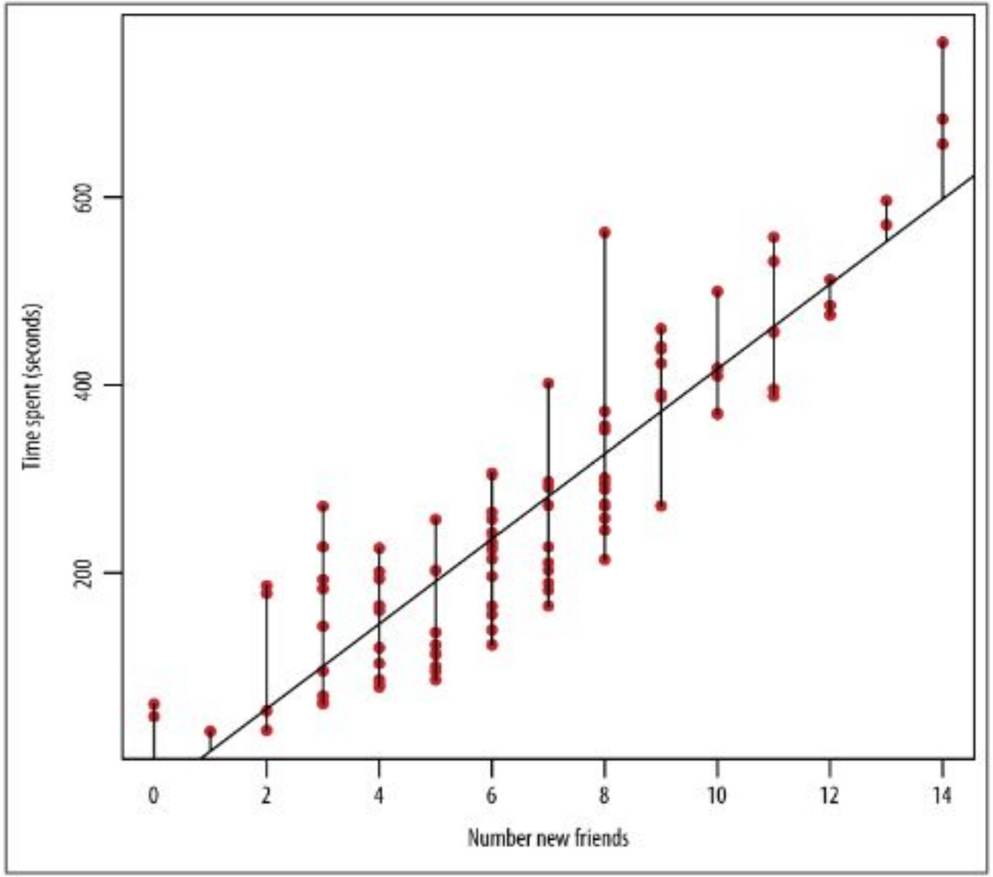
- Plot all the hundred points.
- It looks like there is a linear relationship.
- The more new friends you have, the more time you might spend on the site.
- There is no perfectly deterministic relationship between number of new friends and time spent on the site.
- But it makes sense that there is an association between these two variables.

Linear Regression: Example



- There are many lines, which line is the best fit?
- Many lines look approximately correct, but your goal is to find the optimal one.
- The intuition behind linear regression is that you want to find the line that minimizes the distance between all the points and the line.
- Assuming a linear relationship, start your model by assuming the functional form to be:
  $y = \beta_0 + \beta_1 x$.
- Find the best choices for β0 and β1 using the observed data to estimate them: (x1, y1) , (x2, y2) , . . . (xn, yn).

Linear Regression: Example



- Linear regression seeks to find the line that minimizes the sum of the squares of the vertical distances between the predicted yi’s and the observed yi’s.
- This is to minimize the prediction errors and this method is called least squares estimation.
- Minimize the residual sum of squares given as

$$\text{RSS} \quad = \sum_{i=1}^{n}(y_i - \hat{y})^2$$

y_i is the observed variable value

$\hat{y}$ is the value estimated by the regression line

Linear Regression: Example

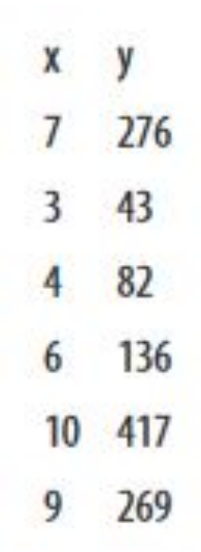
According to least squares estimation,

β1= $$\frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

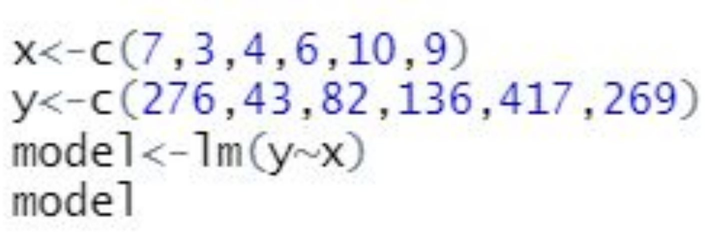
β_0= y-β_1 x

https://www.amherst.edu/system/files/media/1287/SLR_Leastsquares.pdf

R code (for 6 data points)

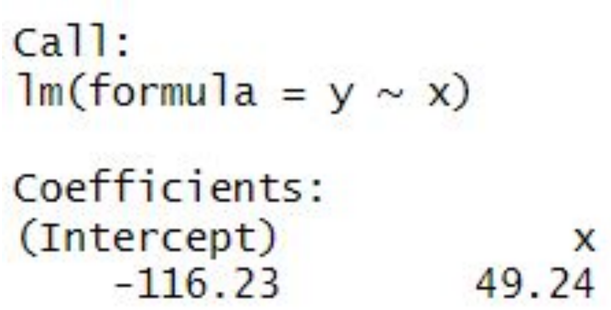


x	y
7	276
3	43
4	82
6	136
10	417
9	269



```
x<-c(7,3,4,6,10,9)
y<-c(276,43,82,136,417,269)
model<-lm(y~x)
model
```

Output

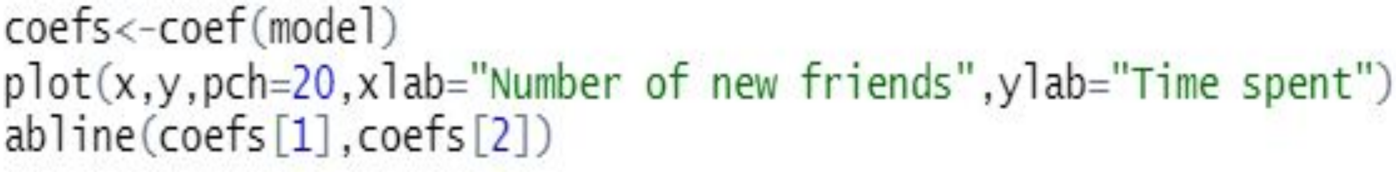


```
Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
    -116.23        49.24
```

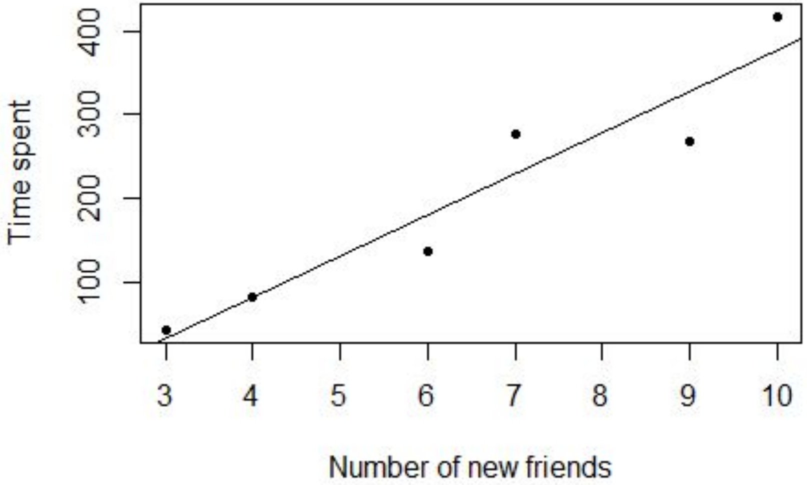
The estimated line is
y=-116.23+49.24x

R code

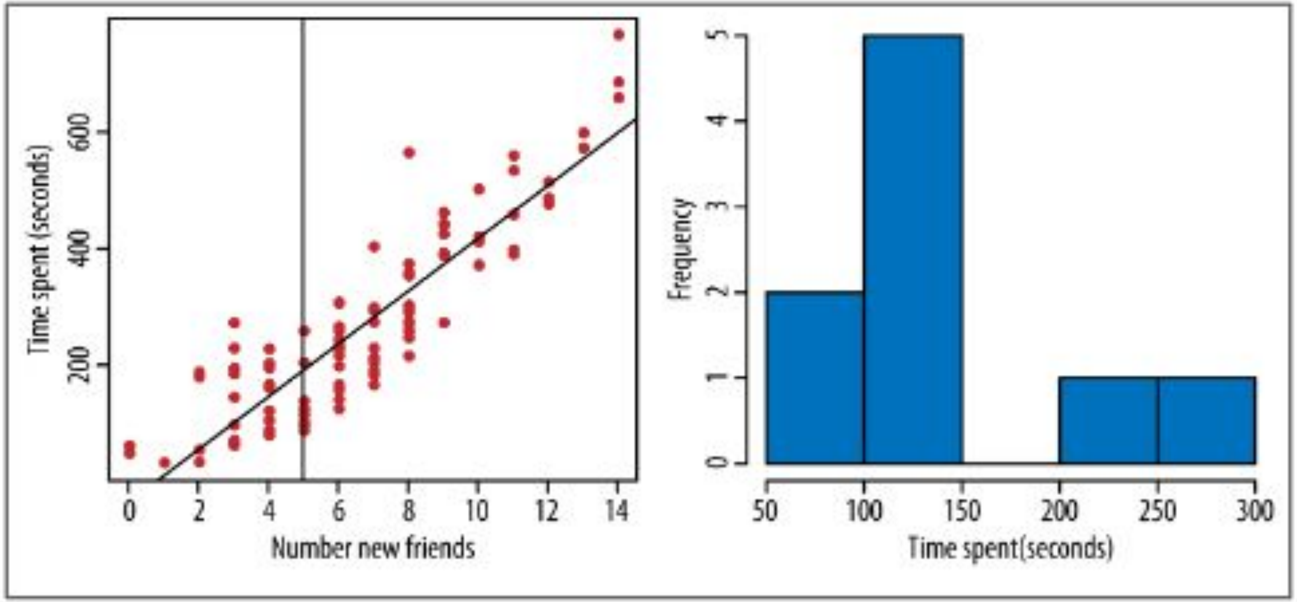


```
coefs<-coef(model)
plot(x,y,pch=20,xlab="Number of new friends",ylab="Time spent")
abline(coefs[1],coefs[2])
```

The estimated line (for 6 points) is y=-116.23+49.24x
The corresponding plot is shown below



R example (for 100 data points)



- The R statement abline(v=5) inserts a vertical line at 5.
- It is observed that for any fixed value 5, the values of y vary.

- For people with 5 new friends, we display their time spent in the plot on the right.

hist(y,xlab="time spent",ylab="frequency")

Creates the histogram for y

Error calculation

- The estimated line (for 100 points) is $y = -32.08 + 45.92x$
- If a new x-value of 5 came in, meaning the user had five new friends, then output value y is 195.7 seconds.
- Does everyone with five new friends is guaranteed to spend 195.7 seconds on the site? The answer is NO.
- For a fixed value of $x = 5$, there is variability among the time spent on the site. You want to capture this variability in your model, so you extend your model to:

  $$y = \beta_0 + \beta_1 x + \varepsilon$$

  where the new term ε is referred to as noise or error term. It is assumed that the distribution of noise is normal distribution.

Error calculation

Error, ε = $\frac{\sum_i e_i^2}{n-2}$

- Where, ei = yi − yi^, difference between actual value and observed value for n data points.
- This is called mean squared error and captures how much the predicted value varies from the observed.
- Dividing by n–2, rather than just n, produces an unbiased estimator.
- The 2 corresponds to the number of model parameters.

Multiple Linear Regression

We can extend the simple linear regression model by adding more predictors (independent variables). This is called multiple linear regression.

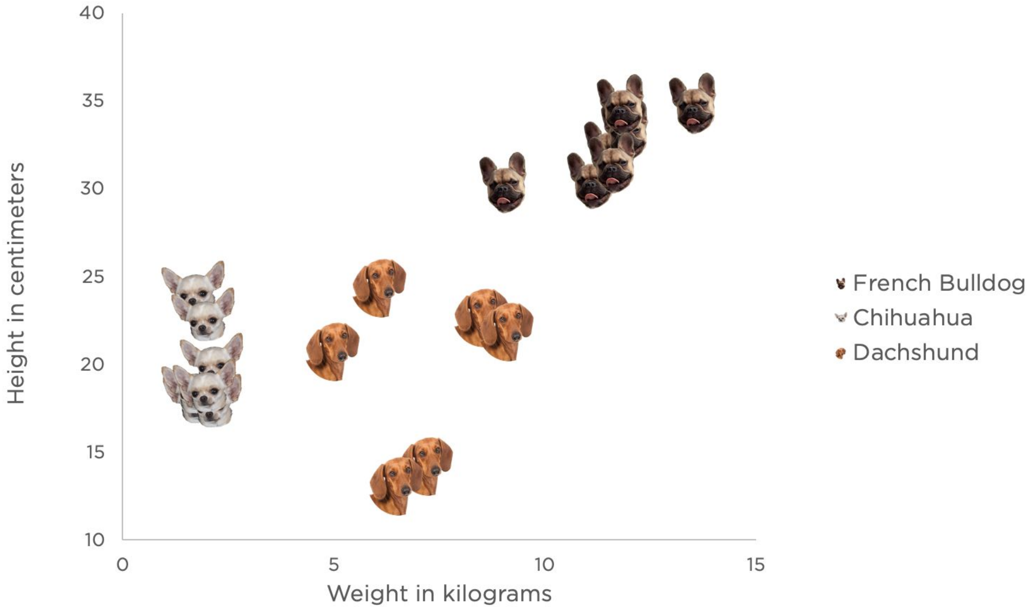
$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$

The R code will just be:

model <- lm(y ~ x_1 + x_2 + x_3), where y is outcome variable and x_1, x_2, and x_3 are predictor variables.

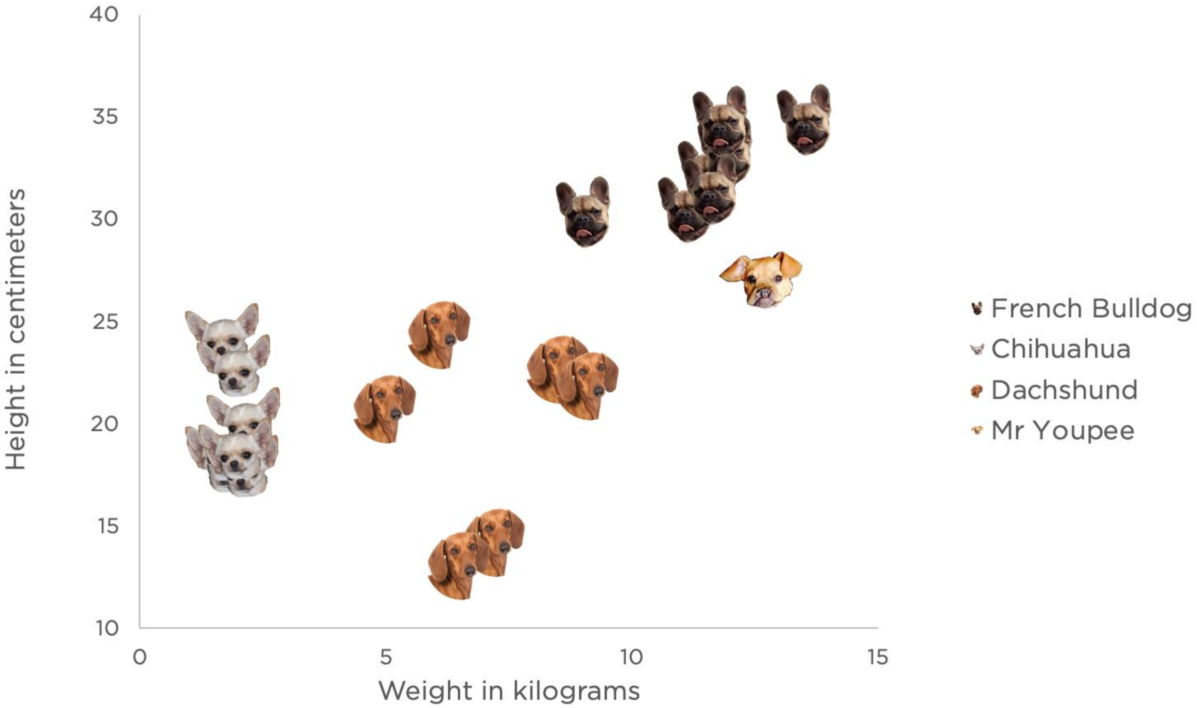
Example: Predict the time spent by a person in facebook based on number of new friends, age, and gender.

k-Nearest Neighbors (k-NN)

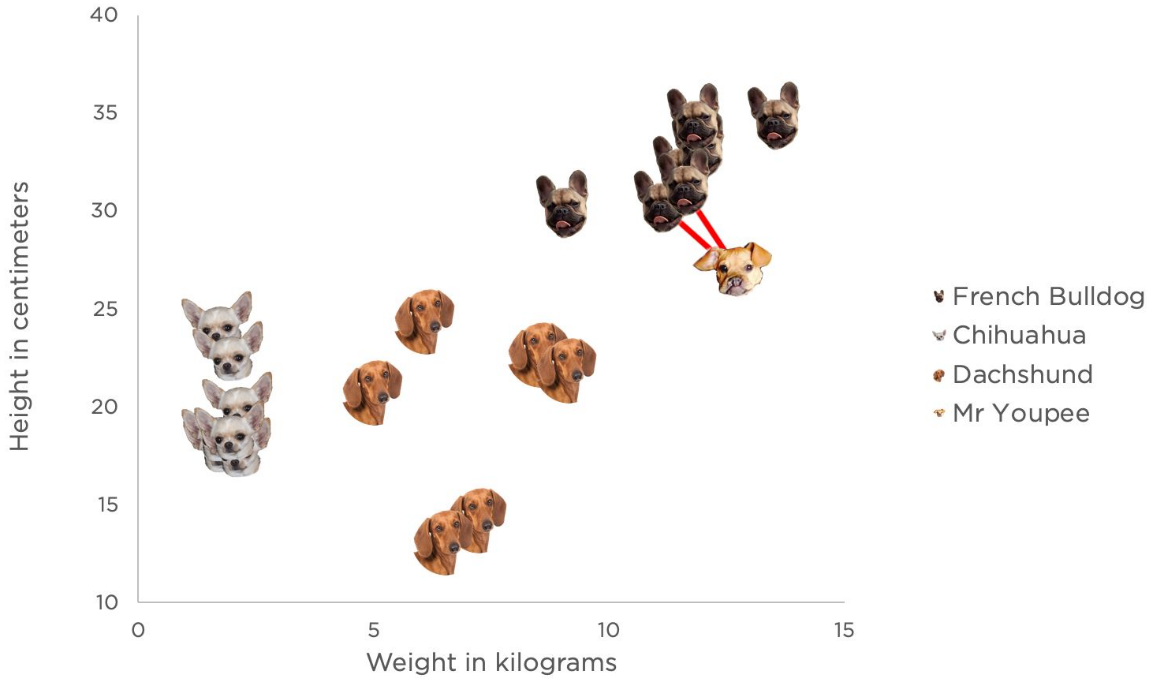


k-Nearest Neighbors (k-NN)

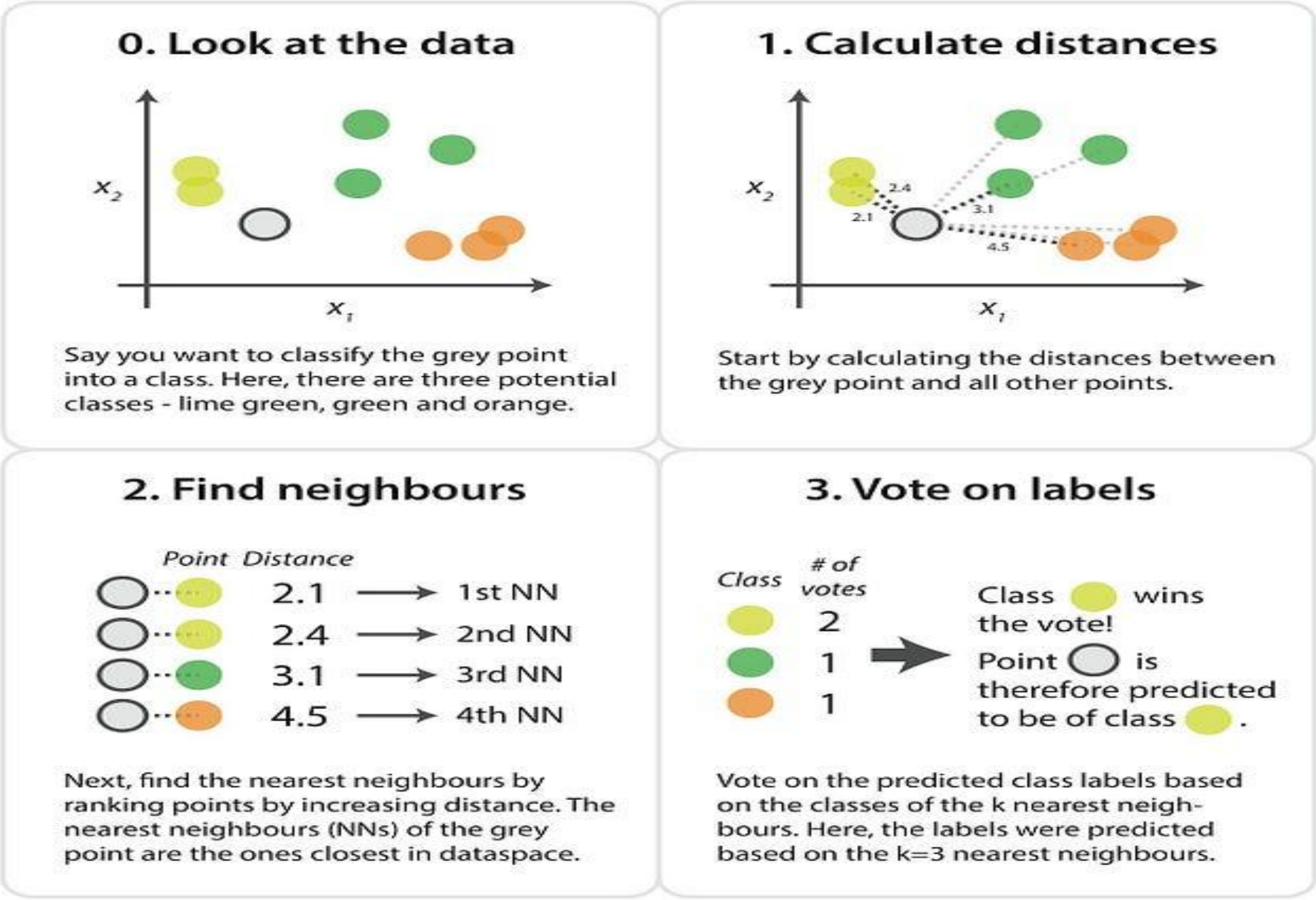
Mr. Youpee belong to which breed?
Mr. Youpee is 27cm high and weighs 12.6kgs.



k-Nearest Neighbors (k-NN)



kNN Algorithm



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

Point	Distance	
○··🟡	2.1	→ 1st NN
○··🟡	2.4	→ 2nd NN
○··🟢	3.1	→ 3rd NN
○··🟠	4.5	→ 4th NN

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

Class	# of votes
🟡	2
🟢	1
🟠	1

Class 🟡 wins the vote!

Point ○ is therefore predicted to be of class 🟡.

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.

k-Nearest Neighbors (k-NN)

- K-NN is a classification algorithm that can be used when you have a set of objects that have been classified (labeled) in some way, and other similar objects that are not classified (labeled) yet, and you want a way to automatically label them.
- Example: The objects can be patients who have been labeled as “high cancer risk” or “low cancer risk”

  Restaurants that have been labeled as “five star” or “four star” or “three star” or “two star” or “one star”.

- The intuition behind k-NN is: Consider the most similar other objects defined in terms of their attributes, look at their labels and give the unassigned objects the majority vote.

k-Nearest Neighbors (k-NN)

- k-NN is based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it.
- The training tuples are described by n attributes.
- Each tuple represents a point in an n-dimensional space. In this way, all the training tuples are stored in an n-dimensional pattern space.
- When given an unknown tuple, a k-nearest-neighbor classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k “nearest neighbors” of the unknown tuple.

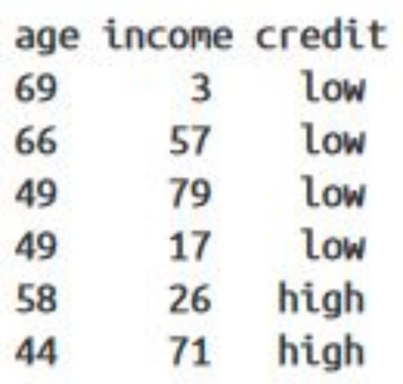
k-Nearest Neighbors (k-NN)

- Example: If there are bunch of movies that were labeled as “thumbs up” and “thumbs down” and you had a movie “new movie” that is not rated yet. Look at the attributes of the movie such as length of the movie, genre, budget, actors, director, etc. Find the movies with the similar attribute values and give the rating to “new movie”.
- Two decisions to be taken are
  - How to measure the similarity or closeness between the objects? Once this is known, for an unlabeled object, we can say how similar all the labeled objects to it and call the most similar objects as neighbors.
  - How many neighbors should you look? This value is k.

k-Nearest Neighbors (k-NN)

Example:

- You have the age, income, and a credit category of high or low for a bunch of people and you want to use the age and income to predict the credit label of “high” or “low” for a new person.
- First few rows of the dataset is shown below:
- Income is represented in thousands.



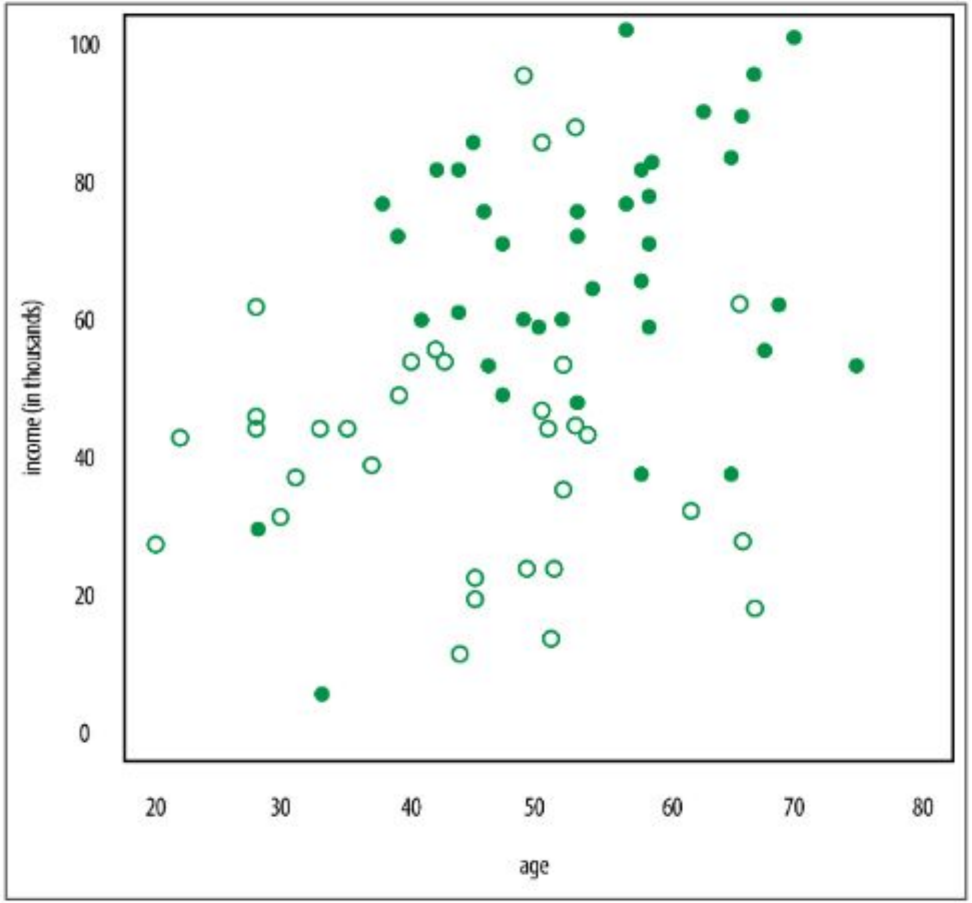
age	income	credit
69	3	low
66	57	low
49	79	low
49	17	low
58	26	high
44	71	high

k-Nearest Neighbors (k-NN)

Plot people as points on the plane and label people with an empty circle if they have low credit ratings.

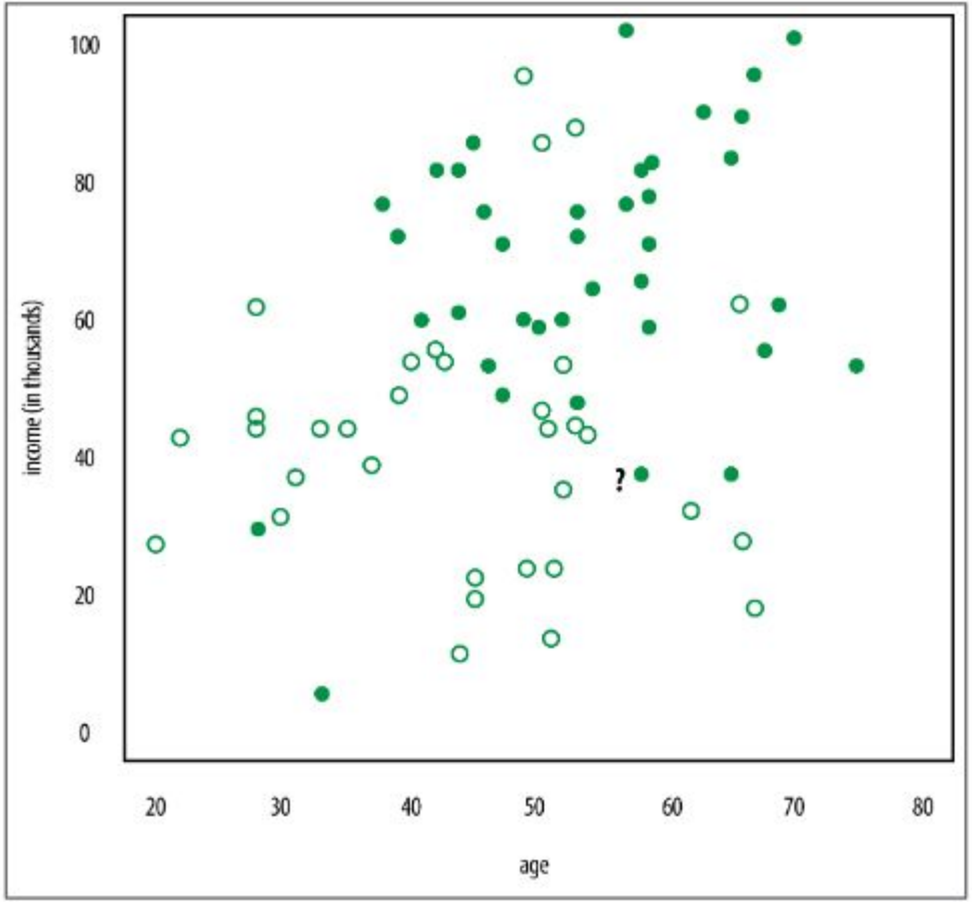
R Code (for sample)

```
age<-c(69,66,49,49,58,44)
income<-c(3,57,79,17,26,71)
credit<-c('low','low','low','low','high','high')
plot(age,income,col=factor(credit))
```



k-Nearest Neighbors (k-NN)

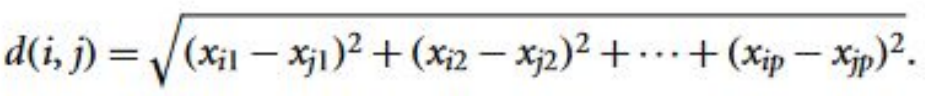
What if a new guy comes in who is 57 years old and who makes 37,000? What's his likely credit rating label?



Similarity or distance metrics

1. Euclidean distance

The most popular distance measure is Euclidean distance. Let i = (x_{i1}, x_{i2}, . . . , x_{ip}) and j = (x_{j1}, x_{j2}, . . . , x_{jp}) be two objects described by p numeric attributes. The Euclidean distance between objects i and j is defined as



$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2}.$$

Example: x = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0) and y = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1). Find (d(x,y)

d(x,y)=sqrt($(5\text{-}3)^2+(0\text{-}0)^2+(3\text{-}2)^2+(0\text{-}0)^2+(2\text{-}1)^2+(0\text{-}1)^2+(0\text{-}0)^2+(2\text{-}1)^2+(0\text{-}0)^2+(0\text{-}1)^2$))=sqrt(4+1+1+1+1+1)=sqrt(9)=3

Similarity or distance metrics

2. Cosine similarity

$$sim(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{||\boldsymbol{x}||\,||\boldsymbol{y}||}$$

where ||x|| is the Euclidean norm of vector x = (x1, x2, . . . , xp), defined as

$$\sqrt{x_1^2 + x_2^2 + \cdots + x_p^2}.$$

Example: x = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0) and y = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1). How similar are x and y?

$$\begin{aligned}\boldsymbol{x}' \cdot \boldsymbol{y} &= 5 \times 3 + 0 \times 0 + 3 \times 2 + 0 \times 0 + 2 \times 1 + 0 \times 1 + 0 \times 0 + 2 \times 1 \\ &\quad + 0 \times 0 + 0 \times 1 = 25\end{aligned}$$

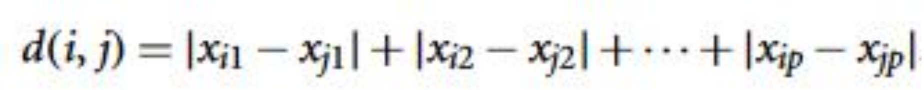
$$||\boldsymbol{x}|| = \sqrt{5^2 + 0^2 + 3^2 + 0^2 + 2^2 + 0^2 + 0^2 + 2^2 + 0^2 + 0^2} = 6.48$$

$$||\boldsymbol{y}|| = \sqrt{3^2 + 0^2 + 2^2 + 0^2 + 1^2 + 1^2 + 0^2 + 1^2 + 0^2 + 1^2} = 4.12$$

$$sim(\boldsymbol{x}, \boldsymbol{y}) = 0.94$$

Similarity or distance metrics

3. Manhattan Distance

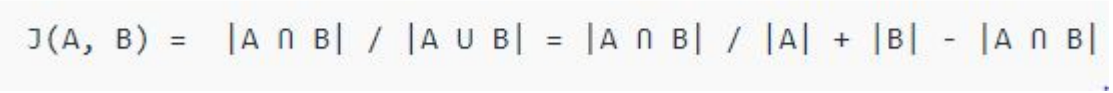


$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|$$

Example: x = (5, 0, 3, 0, 2, 0, 0, 2, 0, 0) and y = (3, 0, 2, 0, 1, 1, 0, 1, 0, 1). Find (d(x,y)

d(x,y)=|(5-3)|+|(0-0)|+|(3-2)|+|(0-0)|+|(2-1)|+|(0-1)|+|(0-0)|+|(2-1)|+|(0-0)|+|(0-1)| = 2+1+1+1+1+1=7

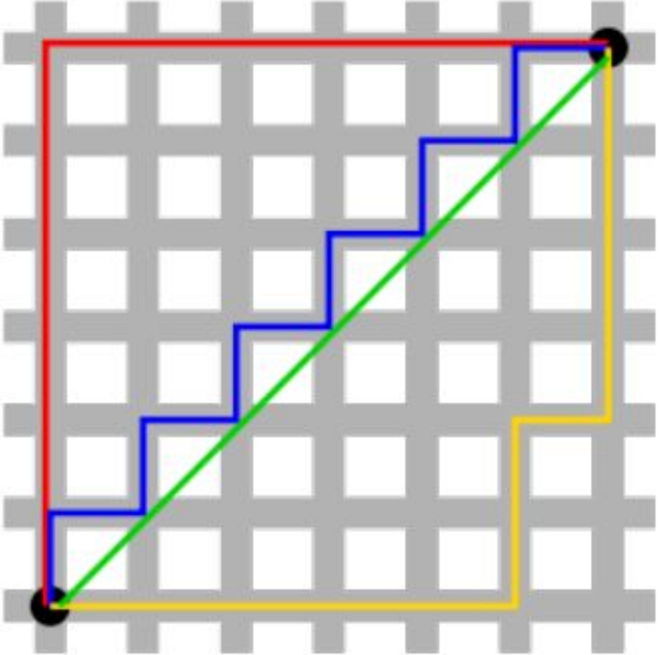
4. Jaccard similarity



```
J(A, B) =  |A ∩ B| / |A ∪ B| = |A ∩ B| / |A| + |B| - |A ∩ B|
```

Friends of A = {C,D,F,G,H}
Friends of B = {D,F,I,J}
J(A,B)=2/7=0.28

Jaccard distance = 1- Jaccard similarity



Green line is Euclidean distance.
Red, Bue and Yellow lines are
Manhattan distance.

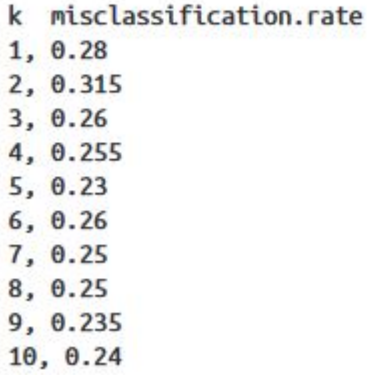
Similarity or distance metrics

5. Hamming Distance

- It is used to find the distance between two strings or pairs of words or DNA sequences of the same length.
- Example: The distance between “olive” and “ocean” is 4. Only the letter ‘o’ is same and the others are different.
- The distance between shoe and hose is 3 because aside from the “e” the other 3 letters are different.

Choosing k value

- This can be determined experimentally.
- Starting with k = 1, we use a test set to estimate the error rate of the classifier. This process can be repeated each time by incrementing k to allow for one more neighbor.
- The k value that gives the minimum error rate may be selected. In general, the larger the number of training tuples, the larger the value of k will be.
- K can be set to 5 because it has the lowest misclassification rate.
- Misclassification rate = 1-accuracy, where accuracy is the ratio of number of correctly predicted labels to the number of total predicted labels.



```
k  misclassification.rate
1, 0.28
2, 0.315
3, 0.26
4, 0.255
5, 0.23
6, 0.26
7, 0.25
8, 0.25
9, 0.235
10, 0.24
```

Steps in k-NN

1. Choose the parameter k, where k is the number of nearest neighbors.
2. Calculate the distance between the query tuple and all the training samples.
3. Sort the distance and find the nearest neighbors based on the minimum distance.
4. Gather the class of the k nearest neighbors.
5. Use the majority class as the class of query tuple.

k-NN Example:

Suppose we have height, weight and T-shirt size of some customers and we need to predict the T-shirt size of a new customer given only height and weight information we have. Data including height, weight and T-shirt size information is shown below:

Height (cms)	Weight (kgs)	T shirt size
158	58	M
158	59	M
158	63	M
160	64	L
163	64	L
165	61	L

Find the T-shirt size of new customer with height=161 and weight=61

k-NN Example:

1. Select k value = 3
2. Euclidean distance between (161,61) to all the training samples.

Height (cms)	Weight (kgs)	T shirt size	Distance
158	58	M	4.24
160	59	M	2.2
160	60	M	1.4
160	64	L	3.16
163	64	L	3.6
165	61	L	4

(160,60,M), (160,59,M), (160,64,L) are nearest to (161,61). The majority class is ‘M’, hence the tuple (161,61) is assigned the class label ‘M’.

Training and test sets

- For any machine learning algorithm, the general approach is to have a training phase, during which you create a model and “train it”; and then you have a testing phase, where you use new data to test how good the model is.
- For k-NN, the training phase is straightforward: it’s just reading in your data with the “M” or “L” T-shirt size marked. In testing, you pretend you don’t know the true label and see how good you are at guessing using the k-NN algorithm.
- Select the test data randomly, usually 20% of the data is selected.

k-NN Example:

S.No.	Height (cms)	Weight (kgs)	T shirt size
1	158	58	M
2	158	59	M
3	158	63	M
4	160	59	M
5	160	60	M
6	163	60	M
7	163	61	M
8	160	64	L
9	163	64	L
10	165	61	L
11	165	62	L
12	165	65	L
13	168	62	L
14	168	63	L
15	168	66	L
16	170	63	L
17	170	64	L
18	170	68	L

The tuples marked in colour are test tuples and the remaining are training tuples

k-NN Example

Select 4 random tuples as the test data (tuple 6,7,15,17).

(160, 60, M), (163, 61, M), (168, 66, L), (170, 64, L)

Pretend that you don't know the labels and find the labels.

Calculate the distance between each test tuple to the remaining 14 tuples and assign the majority label to each of the test tuple.

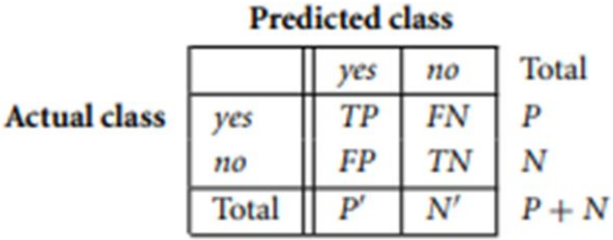
Evaluation metric

1. Accuracy, which is the ratio of the number of correctly predicted tuples to the total number of tested tuples.
2. Misclassification = 1-accuracy.
3. Recall/Sensitivity/True positive rate is the ratio of correctly predicted positive tuples to the total number of actual positive tuples.

   (In our examples, let positive tuples be tuples with class label ‘M’ and the negative tuples are tuples with class label ‘L’)

4. Precision is the ratio of correctly predicted positive tuples to the total number of predicted positive tuples.

Confusion matrix



		Predicted class		
		yes	*no*	Total
Actual class	*yes*	TP	FN	P
	no	FP	TN	N
	Total	P'	N'	$P + N$

The confusion matrix is a useful tool for analyzing how well your classifier can recognize tuples of different classes.

P’ is the number of tuples that were labeled as positive (TP + FP) and N’ is the number of tuples that were labeled as negative (TN + FN).

The total number of tuples is TP + TN + FP + TN, or P + N, or P’ + N’ .

Accuracy = (TP+TN)/(P+N)
Precision = TP/(TP+FP)
Recall = TP/P

Confusion matrix: Multi-class

Actual	Predicted				
		M	L	XL	Total
	M	7	1	3	11
	L	8	2	2	12
	XL	9	3	1	13
	Total	24	6	6	36

Find TP,TN,FP,FN of each class

	TP	TN	FP	FN
M	7	(2+2+3+1)	(8+9)	(1+3)
L	2	(7+3+9+1)	(1+3)	(8+2)
XL	1	(7+1+8+2)	(3+2)	(9+3)

	Accuracy	Precision	Recall
M	(7+8)/36	(7/24)	7/11
L	(2+20)/36	(2/6)	2/12
XL	(1+18)/36	(1/6)	1/13
Macro-avg	0.51	0.26	0.29

Confusion matrix: Multi-class

Macro-averaging scores are arithmetic mean of individual classes' score.
Micro-precision and micro-recall are calculated based on total TP, FP, and FN.
Total TP=(7+2+1)
Total FP=(8+9)+(1+3)+(3+2)=26
Total FN=(1+3)+(8+2)+(9+3)=26
Micro-average precision=10/(10+26)=0.28
Micro-average recall=10/(10+26)=0.28

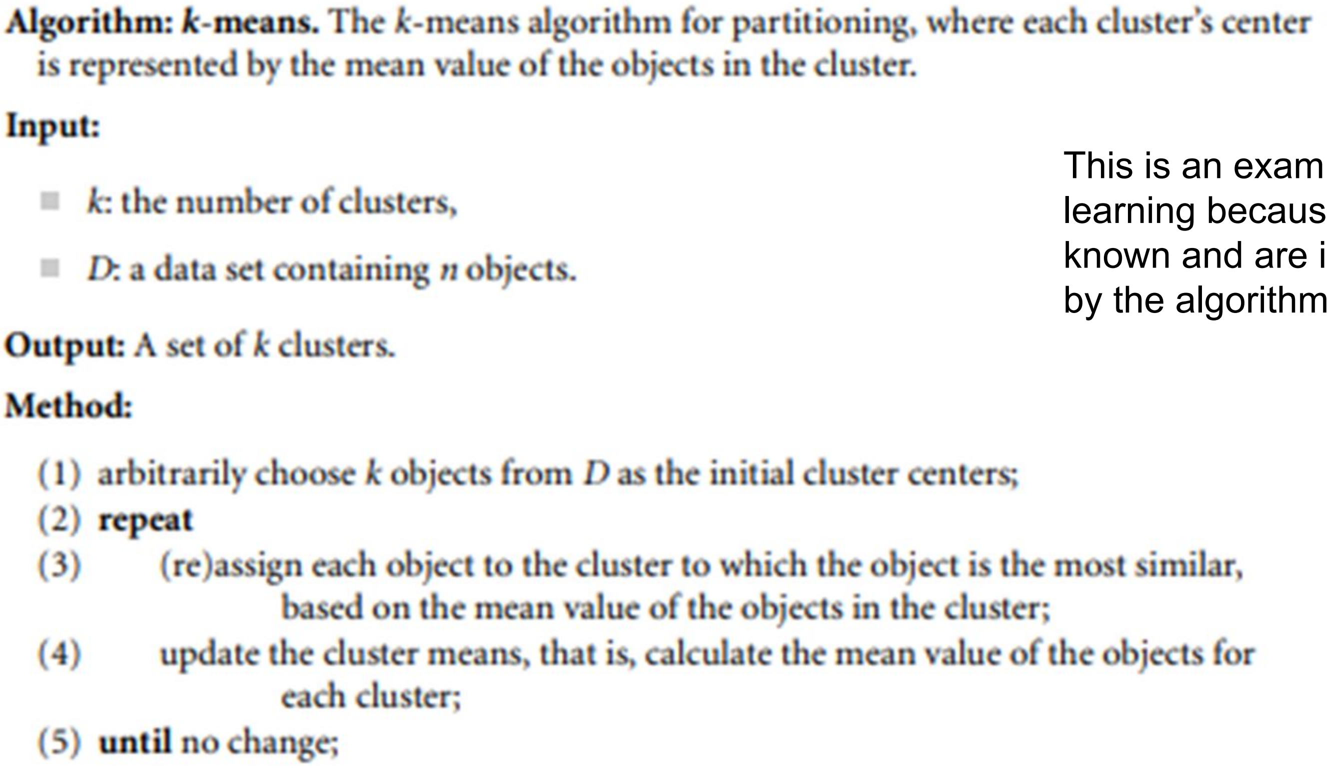
The difference between macro and micro averaging is that macro averaging gives equal weight to each class while micro averaging gives equal weight to each sample.

If we have the same number of samples for each class, both macro and micro will provide the same score.

k-means

- K-means algorithm is used to partition a set of data objects into subsets.
- Each subset is a cluster, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters.
- It is a centroid-based clustering technique which use the centroid of a cluster, Ci , to represent that cluster.
- First, it randomly selects k of the objects in D, each of which initially represents a cluster mean or center.
- For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the Euclidean distance between the object and the cluster mean.
- For each cluster, it computes the new mean using the objects assigned to the cluster in the previous iteration.
- All the objects are then reassigned using the updated means as the new cluster centers.

k-means



Algorithm: *k*-means. The *k*-means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

Output: A set of *k* clusters.

Method:

```
arbitrarily choose $k$ objects from $D$ as the initial cluster centers;
repeat
    (re)assign each object to the cluster to which the object is the most similar,
            based on the mean value of the objects in the cluster;
    update the cluster means, that is, calculate the mean value of the objects for
            each cluster;
until no change;
```

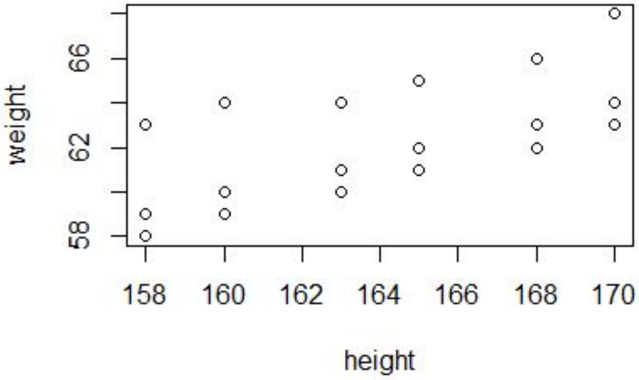
This is an example of unsupervised learning because the labels are not known and are instead discovered by the algorithm.

k-means

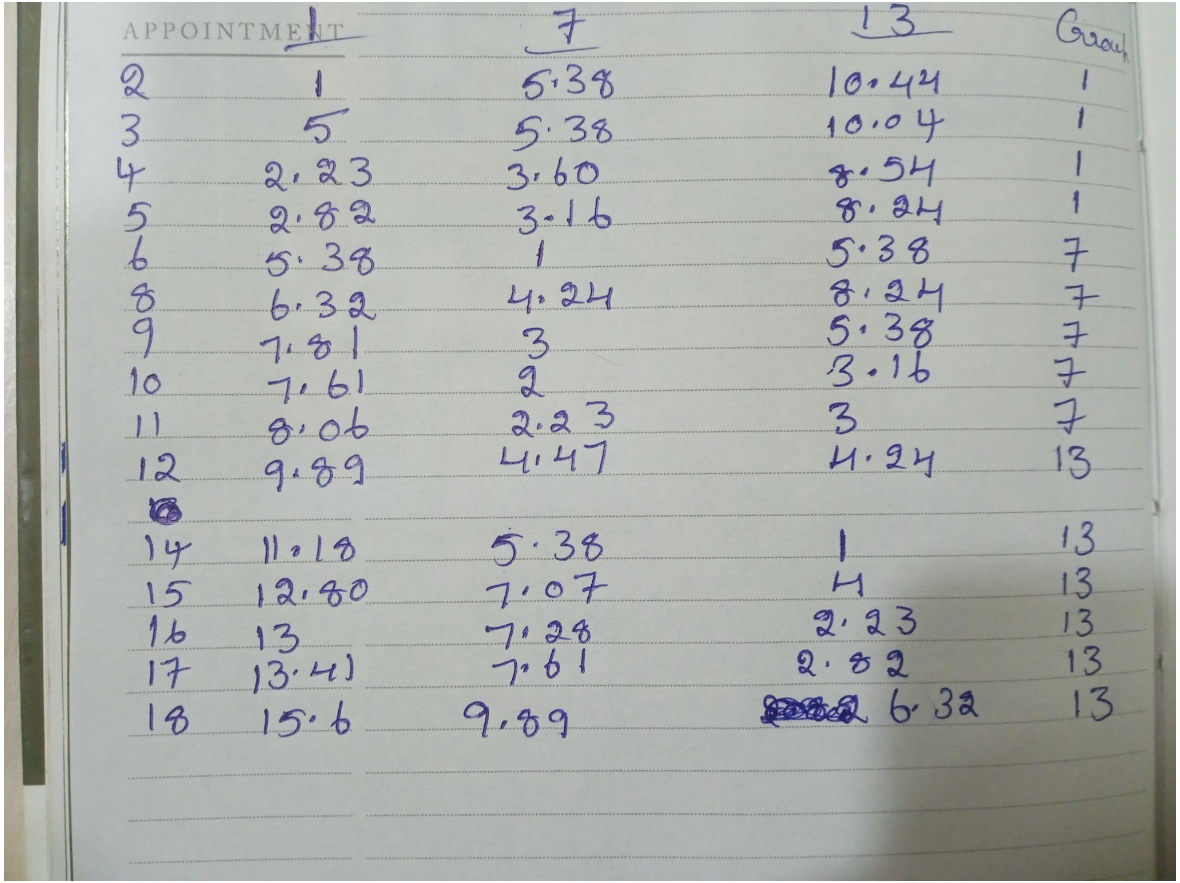
S.No.	Height (cms)	Weight (kgs)
1	158	58
2	158	59
3	158	63
4	160	59
5	160	60
6	163	60
7	163	61
8	160	64
9	163	64

S.No.	Height (cms)	Weight (kgs)
10	165	61
11	165	62
12	165	65
13	168	62
14	168	63
15	168	66
16	170	63
17	170	64
18	170	68

Your goal is to group the persons with similar height and weight.
Consider grouping into 3 clusters.
Randomly select 3 tuples, let it be 1, 7, 13
You have to suggest different users different sized clothes.



k-means



APPOINTMENT	1	7	13	Group
2	1	5.38	10.44	1
3	5	5.38	10.04	1
4	2.23	3.60	8.54	1
5	2.82	3.16	8.24	1
6	5.38	1	5.38	7
8	6.32	4.24	8.24	7
9	7.81	3	5.38	7
10	7.61	2	3.16	7
11	8.06	2.23	3	7
12	9.89	4.47	4.24	13
14	11.18	5.38	1	13
15	12.80	7.07	4	13
16	13	7.28	2.23	13
17	13.41	7.61	2.82	13
18	15.6	9.89	6.32	13

Group1: 1,2,3,4,5

Group7: 6,7,8,9,10,11

Group13: 12,13,14,15,16,17,18

Mean of Group1:
((158+158+158+160+160)/5, (58+59+63+59+60)/5) = (158.8,59.8)

Mean of Group7:
((163+163+160+163+165+165)/6 , (60+61+64+64+61+62)/6) = (163.16,62)

Mean of Group13:
(168.4,64.42)

Issues with k-means

- Choosing k is difficult and initial centers are selected randomly.
- It can recognize only spherical clusters.
- It is sensitive to outliers.
- There are convergence issues - the solution can fail to exist, if the algorithm falls into loop.
- It is applicable only when the mean of a set of objects is defined.

Evaluating cluster quality

- The result of k-means may depend on the initial random selection of cluster centers.
- To obtain good results in practice, it is common to run the k-means algorithm multiple times with different initial cluster centers.
- The quality of cluster Ci can be measured by the within-cluster variation, which is the sum of squared error between all objects in Ci and the centroid ci , defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, c_i)^2,$$

- dist(p,ci) is the Euclidean distance between p and ci.
- Where E is the sum of the squared error for all objects in the data set; p is the point in space representing a given object; and ci is the centroid of cluster Ci.

Evaluating cluster quality

Consider the height and weight attributes

Result of two clustering are shown below. Evaluate each cluster quality.

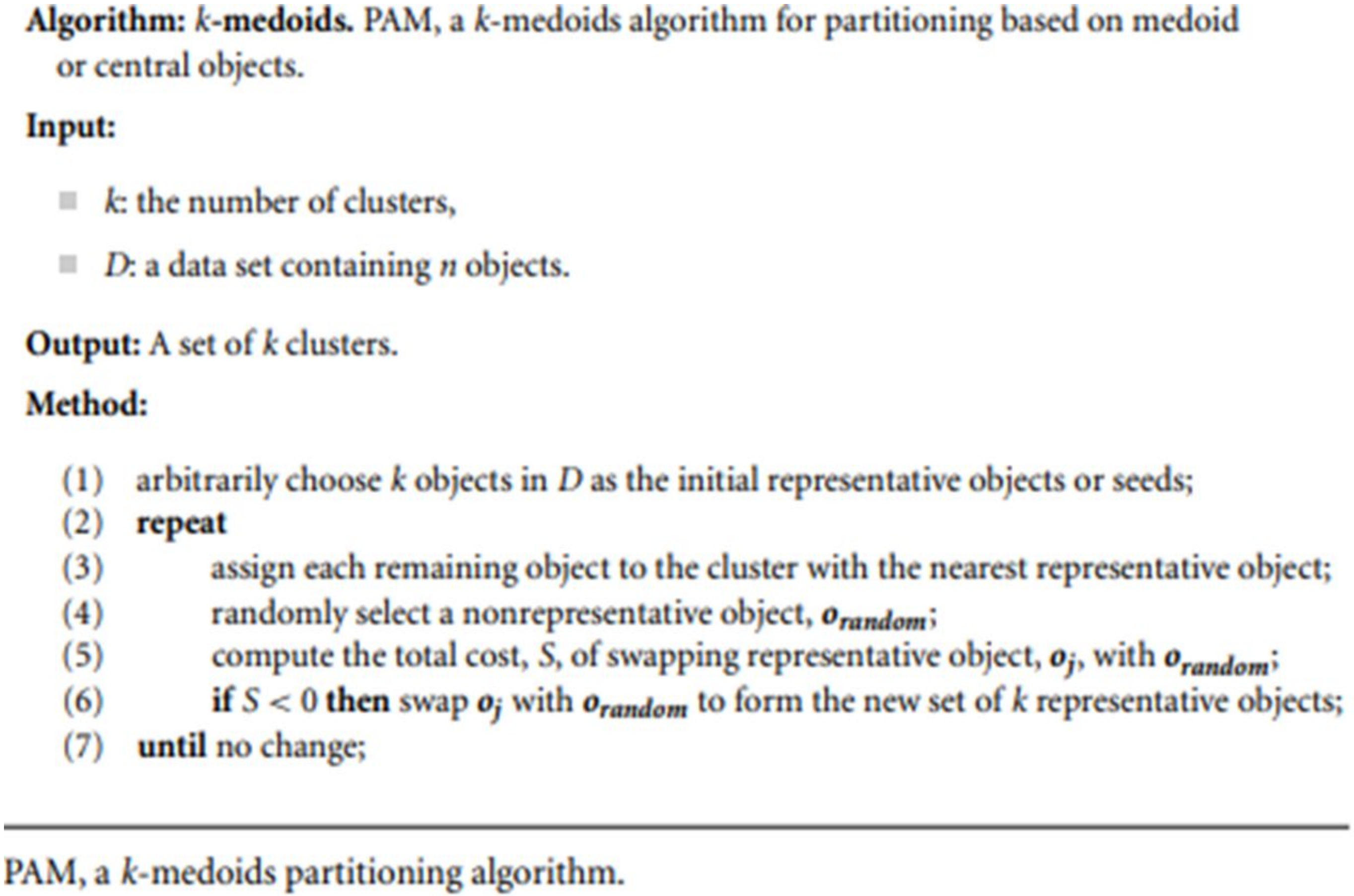
C1 = (158,58) (158,59) (158,63)	C1=(158,58) (163,64) (158,63)
C2 = (160,64) (163,64) (165,61)	C2=(158,59) (160,64) (165,61)

k-Medoids

- K-medoids is also a partitioning technique of clustering that clusters the data set of n objects into k clusters with k known in advance.
- In contrast to the k-means algorithm, k-medoids chooses data points as centers.
- A medoid of a finite dataset is a data point from this set, whose average dissimilarity to all the data points is minimal.
- Let X={x1,x2,...xn} be a se of n points in a space with a distance function d, medoid is defined as

$$x_{\text{medoid}} = \arg\min_{y \in \mathcal{X}} \sum_{i=1}^{n} d(y, x_i)$$

k-Medoids



Algorithm: ***k*-medoids.** PAM, a *k*-medoids algorithm for partitioning based on medoid or central objects.

Input:

- *k*: the number of clusters,
- *D*: a data set containing *n* objects.

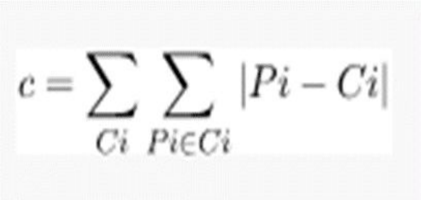
Output: A set of *k* clusters.

Method:

```
arbitrarily choose $k$ objects in $D$ as the initial representative objects or seeds;
repeat
    assign each remaining object to the cluster with the nearest representative object;
    randomly select a nonrepresentative object, $o_{random}$;
    compute the total cost, $S$, of swapping representative object, $o_j$, with $o_{random}$;
    if $S < 0$ then swap $o_j$ with $o_{random}$ to form the new set of $k$ representative objects;
until no change;
```

PAM, a *k*-medoids partitioning algorithm.

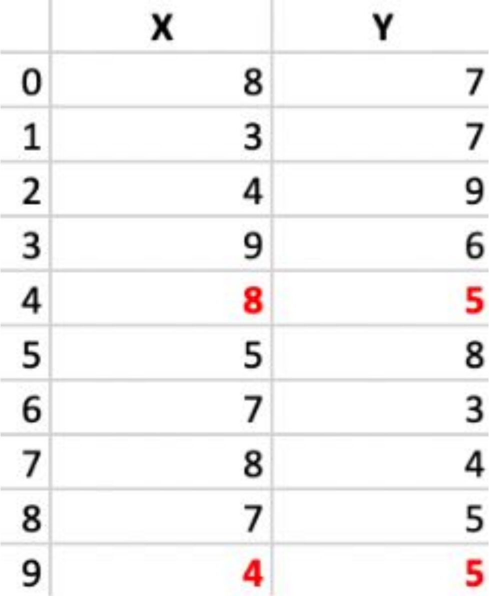
Cost in k-Medoid



$$c = \sum_{Ci} \sum_{Pi \in Ci} |Pi - Ci|$$

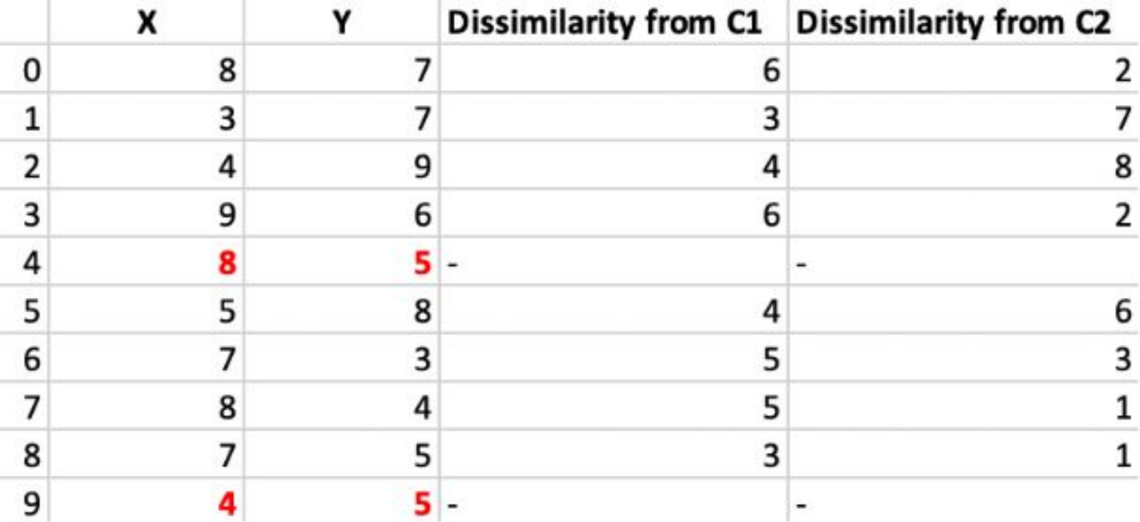
k-Medoids: Example

Input data



	X	Y
0	8	7
1	3	7
2	4	9
3	9	6
4	8	5
5	5	8
6	7	3
7	8	4
8	7	5
9	4	5

- Step-1: Let the randomly selected 2 medoids, so select k = 2, and let C1 (4, 5) and C2 (8, 5) are the two medoids.
- Step-2: Calculating cost. The dissimilarity of each non-medoid point with the medoids is calculated and tabulated:

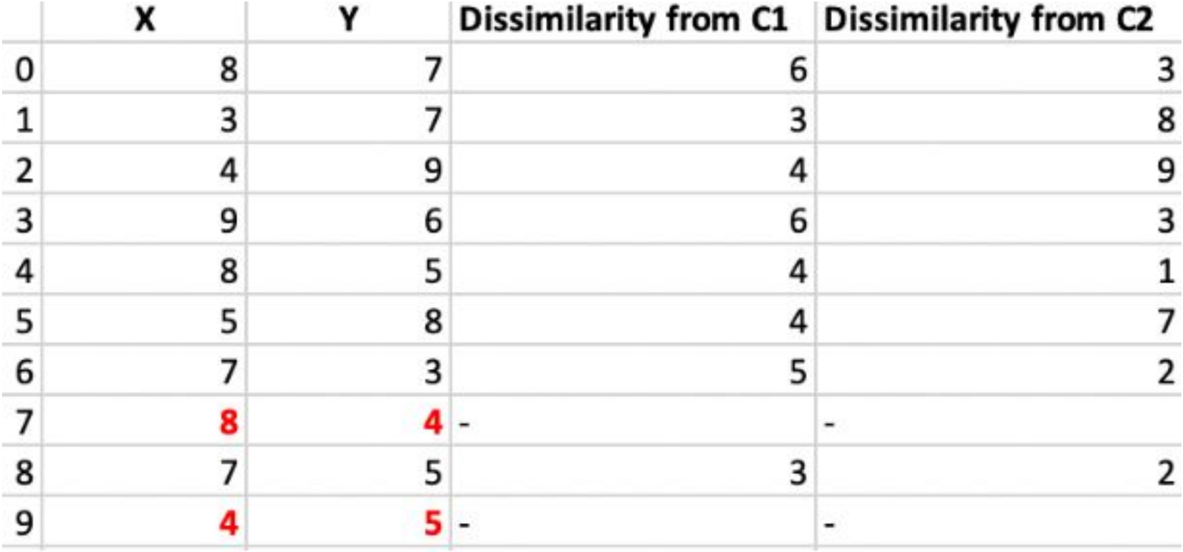


	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	2
1	3	7	3	7
2	4	9	4	8
3	9	6	6	2
4	8	5	-	-
5	5	8	4	6
6	7	3	5	3
7	8	4	5	1
8	7	5	3	1
9	4	5	-	-

Here we have used Manhattan distance formula to calculate the distance matrices between medoid and non-medoid points. That formula is:
Distance = |X1-X2| + |Y1-Y2|.

k-Medoids: Example

- Each point is assigned to the cluster of that medoid whose dissimilarity is less. Points 1, 2, and 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.
- The Cost = (3 + 4 + 4) + (3 + 1 + 1 + 2 + 2) = 20.
- Step-3: Randomly select one non-medoid point and recalculate the cost.
- Let the randomly selected point be (8, 4). The dissimilarity of each non-medoid point with the medoids – C1 (4, 5) and C2 (8, 4) is calculated and tabulated.



	X	Y	Dissimilarity from C1	Dissimilarity from C2
0	8	7	6	3
1	3	7	3	8
2	4	9	4	9
3	9	6	6	3
4	8	5	4	1
5	5	8	4	7
6	7	3	5	2
7	8	4	-	-
8	7	5	3	2
9	4	5	-	-

k-Medoids: Example

Each point is assigned to that cluster whose dissimilarity is less. So, points 1, 2, and 5 go to cluster C1 and 0, 3, 6, 7, 8 go to cluster C2.

The New cost = (3 + 4 + 4) + (2 + 2 + 1 + 3 + 3) = 22

Swap Cost = New Cost – Previous Cost = 22 – 20 and 2 >0

As the swap cost is not less than zero, we undo the swap.

k-Medoids

k-Medoid is more robust than k-Means, because mode is less sensitive to outliers than mean.

Choosing k is difficult and initial medoids are selected randomly.

It may obtain different results for different runs on the same dataset because the first k medoids are chosen randomly.

It can recognize only spherical clusters.

Filtering Spam emails

Spam emails are irrelevant emails sent over the internet, typically to a large number of users.

Nobody likes spam emails.

The filter will examine emails and classify them as either spam or ham (the word for non-spam emails) based on their content.

The outcome of the spam filtering algorithm is either 1 (spam) or 0 (ham).

Filtering Spam emails

Can we apply Linear Regression for this kind of problem?

No.

In order to use linear regression, we need to have a training set of emails where the messages have already been labeled with some outcome variable. In this case, the outcomes are either spam or not.

Once you build a model, email messages would come in without a label, and you’d use your model to predict the labels.

Linear regression cannot give 0 or 1, it is aimed at giving a continuous output. Hence, Linear regression is not appropriate for this data.

Filtering Spam emails

- Can we use k-nearest neighbors to create a spam filter?
- No.
- Consider 10,000 emails and 100,000 words (these may be the frequently occurred words in all the emails), and each word is a feature in the dataset. Each row is about an email.
- The value of the feature is either 1 or 0 depending on whether the word is present or not. Two Emails are near each other based on which words they have in common.
- Computing distances in a 100,000-dimensional space requires lots of computational work. This is called curse of dimensionality and it makes k-NN a poor algorithm in this case.
- Too many dimensions causes every observation in your dataset to appear equidistant from all the others.

Naive Bayes

A Naive Bayes classifier is a probabilistic machine learning model that’s used for classification task.

- Given events x and y, there’s a relationship between the probabilities of either event, denoted p(x) and p(y), the joint probabilities (both happen, which is denoted p(x,y), and conditional probabilities (event x happens given y happens, denoted p(x|y)):
- $p(y|x)\,p(x) = p(x,y) = p(x|y)\,p(y)$

$$p(y|x) = \frac{p(x|y)\,p(y)}{p(x)}$$

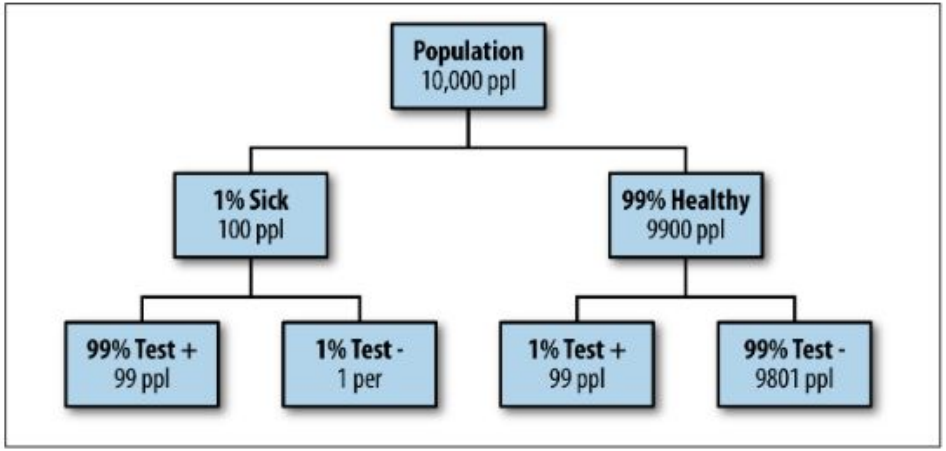
←

Bayes’ law

Naive Bayes

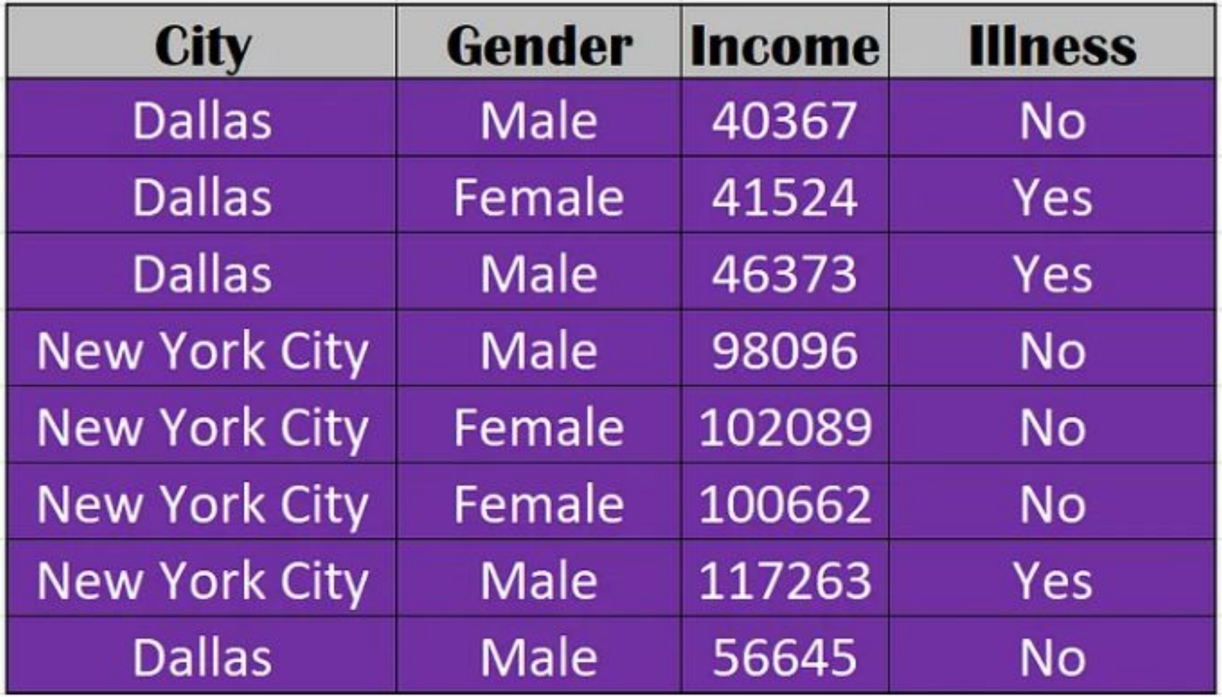
- Testing for a rare disease, where 1% of the population is infected.
- 99% of sick patients test positive.
- 99% of healthy patients test negative

Given that a patient tests positive, what is the probability that the patient is actually sick?Let x refers to “the test is positive” and y refers to “Sick”, Given that a patient tests positive, what is the probability that the patient is actually sick?



$$p(sick|+) = \frac{p(+|sick)p(sick)}{p(+)} = \frac{0.99 \cdot 0.01}{0.99 \cdot 0.01 + 0.01 \cdot 0.99} = 0.50 = 50\%$$

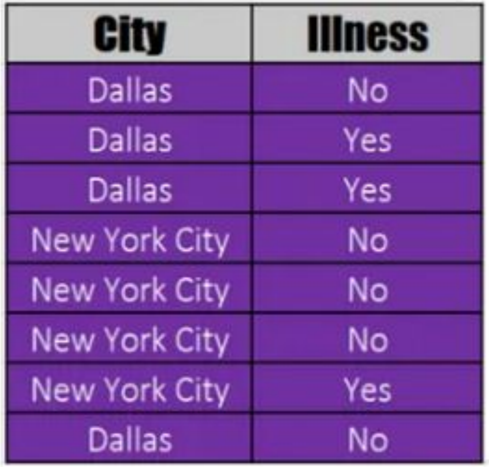
Naive Bayes



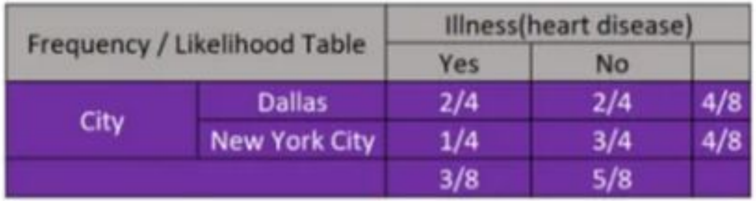
City	Gender	Income	Illness
Dallas	Male	40367	No
Dallas	Female	41524	Yes
Dallas	Male	46373	Yes
New York City	Male	98096	No
New York City	Female	102089	No
New York City	Female	100662	No
New York City	Male	117263	Yes
Dallas	Male	56645	No

Heart Disease Diagnosis dataset

Naive Bayes



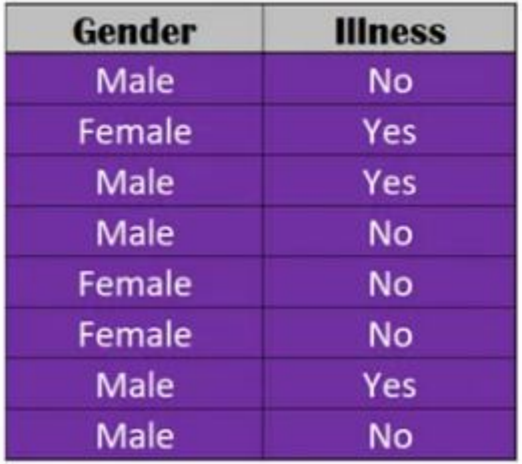
City	Illness
Dallas	No
Dallas	Yes
Dallas	Yes
New York City	No
New York City	No
New York City	No
New York City	Yes
Dallas	No



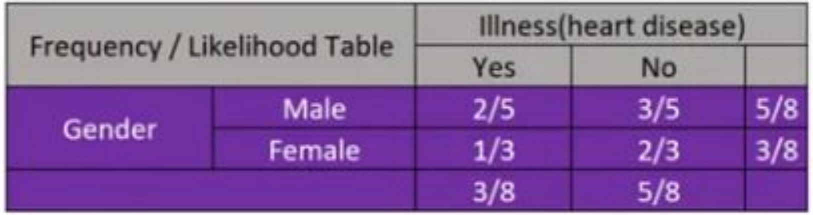
Frequency / Likelihood Table		Illness(heart disease)		
		Yes	No	
City	Dallas	2/4	2/4	4/8
	New York City	1/4	3/4	4/8
		3/8	5/8	

So , the various probabilities we get from the above table are :
P(Dallas|Yes) = 2/4 and P(Dallas | No) = 2/4
P(New York City|Yes) = 1/4 and P(New York City|No) = 3/4
P(Dallas) = 4/8 and P(New York City) = 4/8
P(Yes) = 3/8 and P(No) = 5/8
P(Yes|Dallas) = (P(Yes) * P(Dallas|Yes))/P(Dallas) = ((3/8)*(2/4))/(4/8) = 0.375
Similarly , P(No|Dallas) = 0.625 , P(Yes|New York City) = 0.18 , P(No| New York City) = 0.93

Naive Bayes



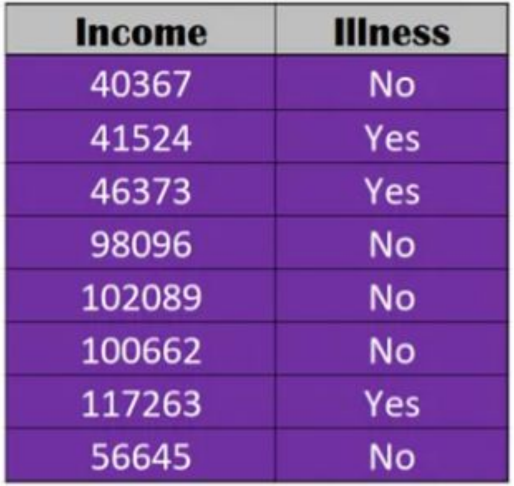
Gender	Illness
Male	No
Female	Yes
Male	Yes
Male	No
Female	No
Female	No
Male	Yes
Male	No



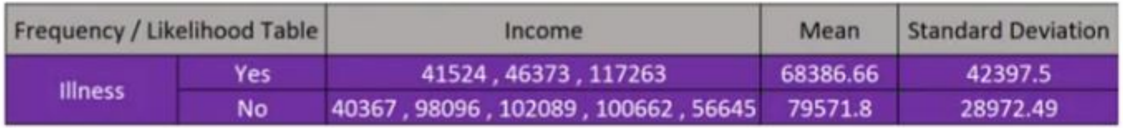
Frequency / Likelihood Table		Illness(heart disease)		
		Yes	No	
Gender	Male	2/5	3/5	5/8
	Female	1/3	2/3	3/8
		3/8	5/8	

P(Male|Yes) = 2/5 and P(Male | No) = 3/5
P(Female|Yes) = 1/3 and P(Female|No) = 2/3
P(Male) = 5/8 and P(Female) = 3/8
P(Yes) = 3/8 and P(No) = 5/8
P(Yes|Male) = (P(Yes) * P(Male|Yes))/P(Male) = ((3/8)*(2/5))/(5/8) = 0.24
Similarly , P(No|Male) = 0.6 , P(Yes|Female) = 0.33 , P(No| Female) = 1.11

Naive Bayes



Income	Illness
40367	No
41524	Yes
46373	Yes
98096	No
102089	No
100662	No
117263	Yes
56645	No



Frequency / Likelihood Table		Income	Mean	Standard Deviation
Illness	Yes	41524 , 46373 , 117263	68386.66	42397.5
	No	40367 , 98096 , 102089 , 100662 , 56645	79571.8	28972.49

For , those columns which are not categorical like the “Income” column , we will use Mean , Standard Deviation and Normal Distribution formula for calculating Likelihood of the patient having a Heart disease or , not .

Mean

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Standard deviation

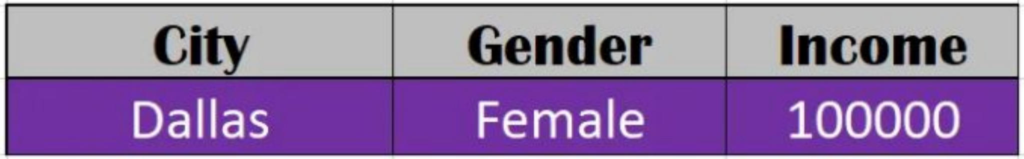
$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2\right]^{0.5}$$

Normal distribution

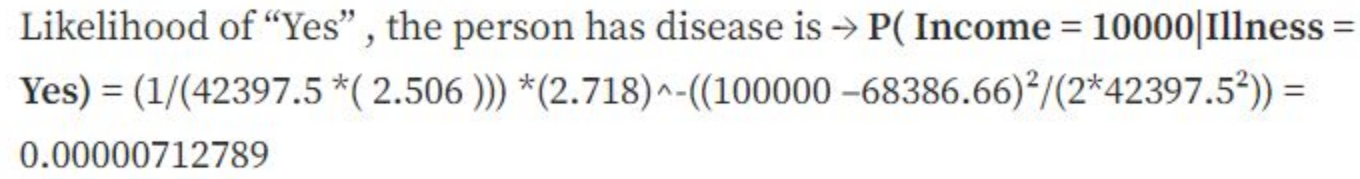
$$f(x) = \frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Naive Bayes

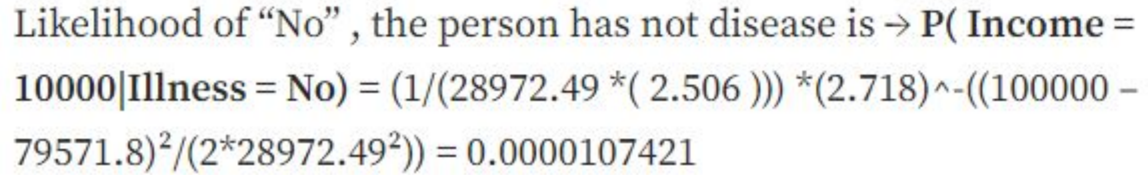
Prediction of new test dataset



City	Gender	Income
Dallas	Female	100000



Likelihood of “Yes” , the person has disease is → **P(Income = 10000|Illness = Yes)** = (1/(42397.5 *(2.506))) *(2.718)^-((100000 –68386.66)^2/(2*42397.5^2)) = 0.00000712789



Likelihood of “No” , the person has not disease is → **P(Income = 10000|Illness = No)** = (1/(28972.49 *(2.506))) *(2.718)^-((100000 – 79571.8)^2/(2*28972.49^2)) = 0.0000107421

Naive Bayes

P(Class=Yes|X) = P(X|Class=yes) * P(Class=yes) = P(X|Class=yes) * 3/8

P(Class=No|X) = P(X|Class=No) * P(Class=No) = P(X|Class=No) * 5/8

P(X|Class=Yes) = P(City = Dallas|Class=Yes) * P(Gender = Female| Class=Yes) * P(Income=100K| Class=Yes) = 2/4 * 1/3 * 0.00000712789 = 1.1879e-6

P(X|Class=No) = P(City = Dallas|Class=No) * P(Gender = Female| Class=No) * P(Income=100K| Class=No) = 2/4 * 2/3 * 0.0000107421 = 3.5809e-6

P(Class=Yes|X) = 1.1879e-6 * 3/8 = 0.445e-6

P(Class=No|X) = 3.5809e-6 * 5/8 = 2.238e-6

Since P(Class=No|X) > P (Class=Yes|X), the class label is No.

Naive Bayes classification

Let D be a training set of tuples and their associated class labels. As usual, each tuple is represented by an n-dimensional attribute vector, X = (x1, x2,… , xn), depicting n measurements made on the tuple from n attributes, respectively, A1, A2, …. , An.

Suppose that there are m classes, C1, C2, …. , Cm. Given a tuple, X, the classifier will predict that X belongs to the class having the highest posterior probability, conditioned on X. That is, the naive Bayesian classifier predicts that tuple X belongs to the class Ci if and only if

$$P(C_i|X) > P(C_j|X) \quad \text{for } 1 \leq j \leq m, j \neq i.$$

Thus, we maximize P(Ci /X).

By Bayes’ theorem,

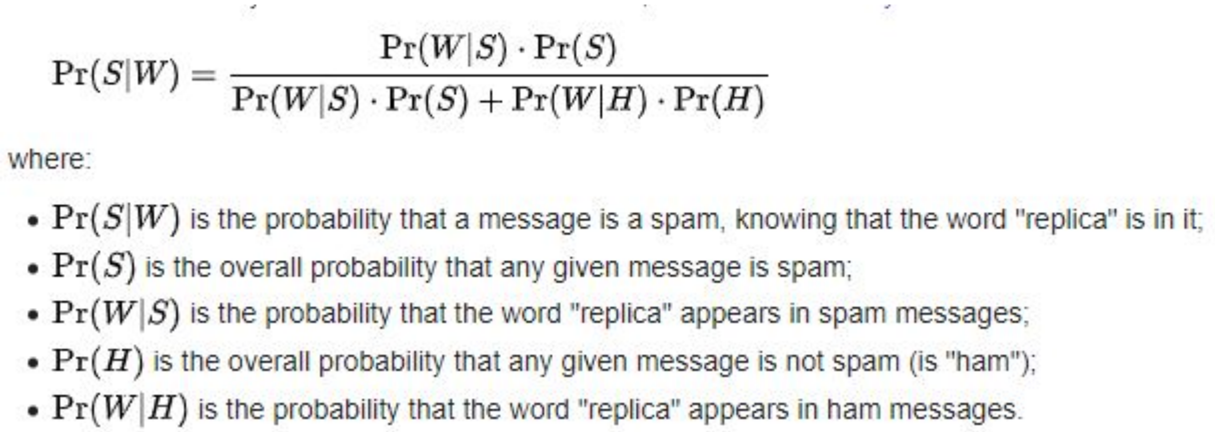
$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

$$p(\mathbf{x}) = \sum_k p(C_k)\, p(\mathbf{x} \mid C_k)$$

As P(X) is constant for all classes, only P(X | Ci) P(Ci) needs to be maximized.

Naive Bayes

The formula used by the spam detecting software using Naive Bayes is:



$$\Pr(S|W) = \frac{\Pr(W|S) \cdot \Pr(S)}{\Pr(W|S) \cdot \Pr(S) + \Pr(W|H) \cdot \Pr(H)}$$

where:

- $\Pr(S|W)$ is the probability that a message is a spam, knowing that the word "replica" is in it;
- $\Pr(S)$ is the overall probability that any given message is spam;
- $\Pr(W|S)$ is the probability that the word "replica" appears in spam messages;
- $\Pr(H)$ is the overall probability that any given message is not spam (is "ham");
- $\Pr(W|H)$ is the probability that the word "replica" appears in ham messages.

Naive Bayes is called as naive as it assumes each variable is independent of each other, but this is unrealistic for real data.

Naive Bayes

- Bayesian spam detection software makes the assumption that any incoming message have equal probability to be a spam or ham.
- The filters that use this hypothesis are said to be "not biased".
- This assumption permits simplifying the general formula to:

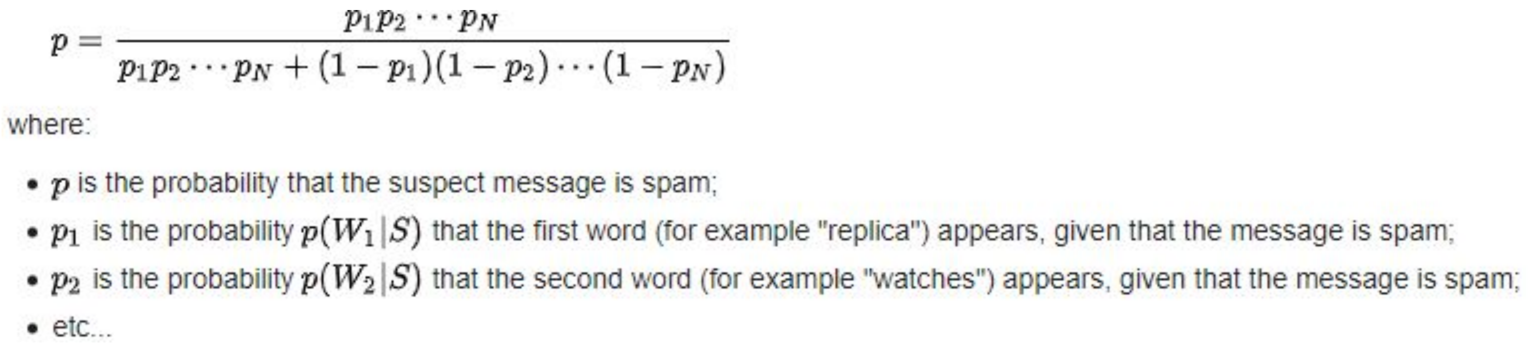
$$\Pr(S) = 0.5; \Pr(H) = 0.5$$

$$\Pr(S|W) = \frac{\Pr(W|S)}{\Pr(W|S) + \Pr(W|H)}$$

- This is known as Spamicity.
- Determining whether a message is spam or ham based only on the presence of the word "replica" is error-prone, which is why bayesian spam software tries to consider several words and combine their spamicities to determine a message's overall probability of being spam.

Naive Bayes

The following formula from Bayes' theorem is used:



$$p = \frac{p_1 p_2 \cdots p_N}{p_1 p_2 \cdots p_N + (1 - p_1)(1 - p_2) \cdots (1 - p_N)}$$

where:

- p is the probability that the suspect message is spam;
- p_1 is the probability $p(W_1|S)$ that the first word (for example "replica") appears, given that the message is spam;
- p_2 is the probability $p(W_2|S)$ that the second word (for example "watches") appears, given that the message is spam;
- etc...

The result p is typically compared to a given threshold to decide whether the message is spam or not. If p is lower than the threshold, the message is considered as likely ham, otherwise it is considered as likely spam.

Example:

Given 1500 spam mails and 3672 ham mails, the frequency of the word “meeting” in spam mails is 16, and the frequency of the word “meeting” in ham mails is 153. Calculate the chance of the email is spam given the word meeting is in the spam folder.

Answer

P(spam) = 1500/(1500+3672) = 0.29

P(ham) = 3672/(1500+3672) = 0.71

P(meeting/spam) = 16/1500 = 0.0106

P(meeting/ham) = 153/3672 = 0.0416

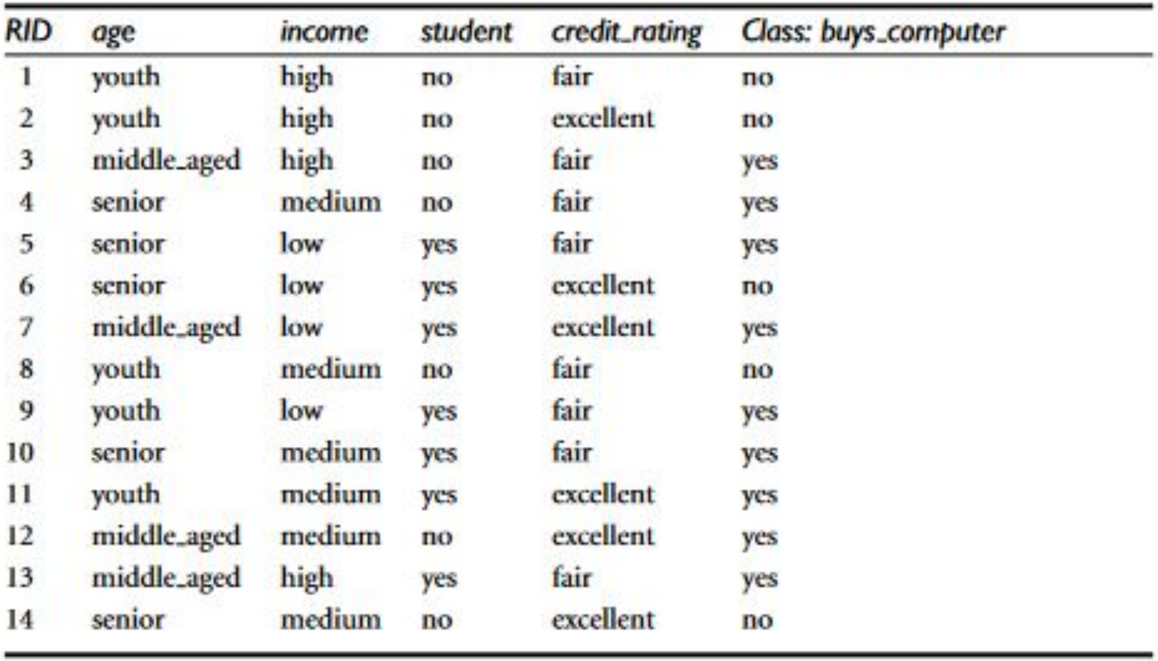
P(spam/meeting) = (P(meeting/spam)*P(spam)) / P(meeting) = (0.0106 * 0.29)/ P(meeting)

P(meeting) = P(meeting/spam)*P(spam) + P(meeting/ham)*P(ham)

= (0.0106 * 0.29) + (0.0416 * 0.71) = 0.03261

P(spam/meeting)=0.003074 / 0.03261 = 0.09 = 9%

Naive Bayes Classification:Example



RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Classify the tuple X = (age=youth,income=medium,student=yes,credit_rating=fair)

Feature Selection

- The idea of feature selection is identifying the subset of data or transformed data that you want to put into your model.
- It is possible to have many redundancies or correlated variables in your raw data, and so you don’t want to include all those variables in your model.
- Similarly you might want to construct new variables by transforming the variables, say, a continuous variable into a binary variable, before feeding them into the model.
- We are getting bigger and bigger datasets, but that’s not always helpful. If the number of features is larger than the number of observations then large isn’t necessarily good.
- To improve the performance of your models, you want to improve your feature selection process.

Feature Selection

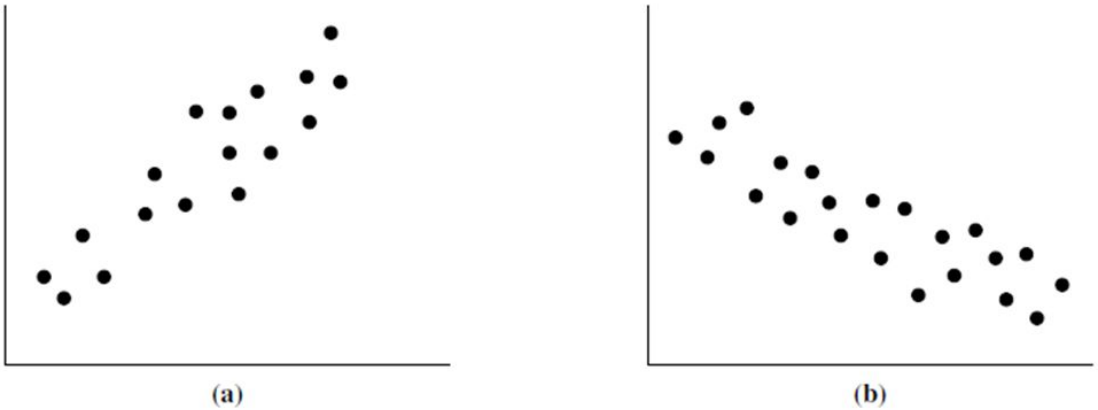
Finding **correlation** between the attributes

Given two numeric attributes A, B, the correlation can be found using the formula:

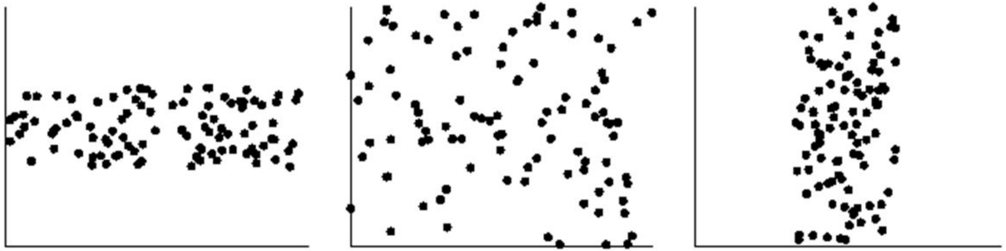
$$r_{A,B} = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{n\sigma_A\sigma_B} = \frac{\sum_{i=1}^{n}(a_i b_i) - n\bar{A}\bar{B}}{n\sigma_A\sigma_B}$$

where n is the number of tuples, ai and bi are the respective values of A and B in tuple i, $\bar{A}$ and $\bar{B}$ are the respective mean values of A and B, and σA and σB are the respective standard deviations of A and B.

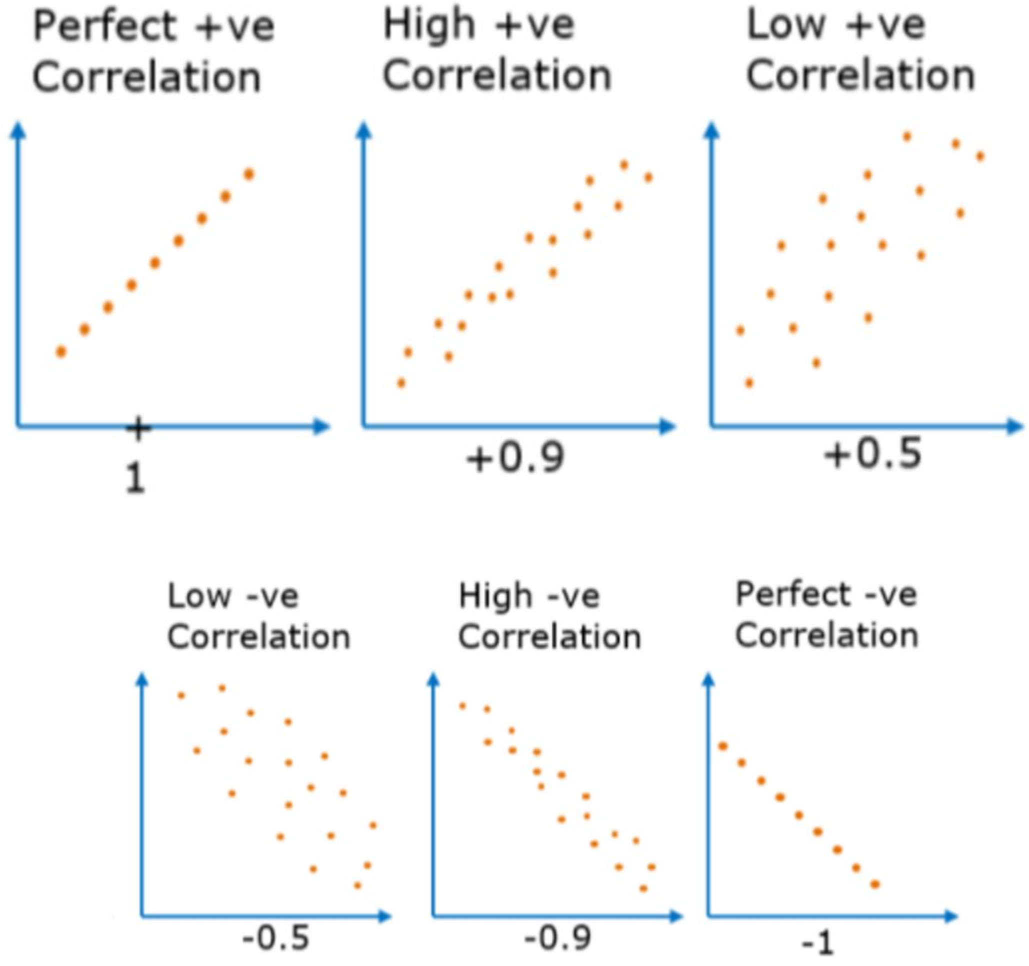
- Note that -1 ≤r ≤ +1. If r is greater than 0, then A and B are positively correlated, meaning that the values of A increase as the values of B increase.
- If the resulting value is less than 0, then A and B are negatively correlated, where the values of one attribute increase as the values of the other attribute decrease.
- If the resulting value is equal to 0, then A and B are independent and there is no correlation between them



Scatter plots can be used to find (a) positive or (b) negative correlations between attributes.



Three cases where there is no observed correlation between the two plotted attributes in each of the data sets.



Perfect +ve Correlation

High +ve Correlation

Low +ve Correlation

Low -ve Correlation

High -ve Correlation

Perfect -ve Correlation

Feature Selection

Finding **covariance** between the attributes

Given two numeric attributes A,B, the covariance can be found using the formula:

$$Cov(A,B) = E((A-\bar{A})(B-\bar{B})) = \frac{\sum_{i=1}^{n}(a_i-\bar{A})(b_i-\bar{B})}{n}$$

Relationship between correlation coefficient and covariance:

$$r_{A,B} = \frac{Cov(A,B)}{\sigma_A \sigma_B}$$

Covariance depends on units of A and B. If we change the scale of A and B, then covariance will also be changed.

Feature Selection

Positive Covariance number indicates a positive relationship [x increases, y also increases]

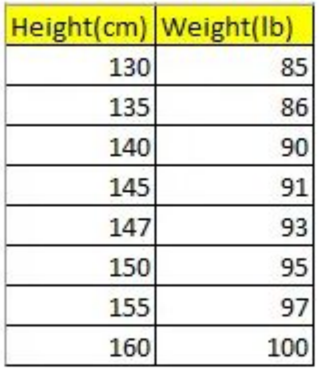
Covariance number ranges from -∞ to +∞.

Negative Covariance number indicates an inverse relationship [x increases then y decreases and vice versa]

Covariance is near to 0, if they are independent.

Feature Selection

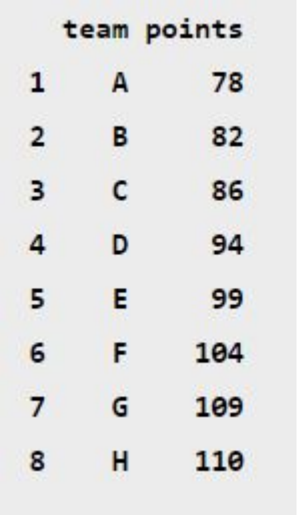
Find the correlation and covariance between the attributes height and weight



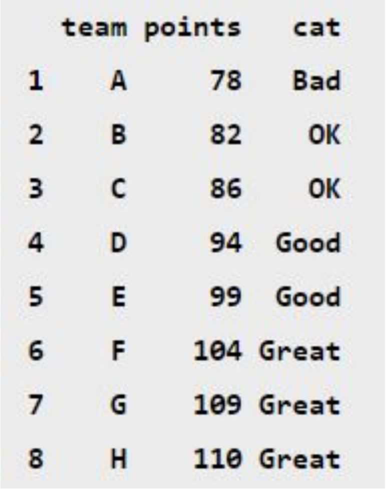
Height(cm)	Weight(lb)
130	85
135	86
140	90
145	91
147	93
150	95
155	97
160	100

Feature selection

Converting continuous variable to categorical:



```
  team points
1    A     78
2    B     82
3    C     86
4    D     94
5    E     99
6    F    104
7    G    109
8    H    110
```

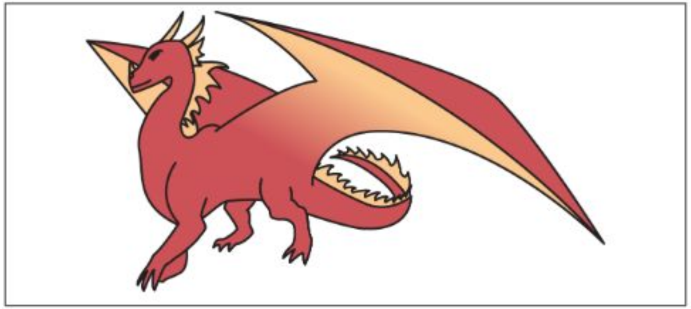


```
  team points   cat
1    A     78   Bad
2    B     82    OK
3    C     86    OK
4    D     94  Good
5    E     99  Good
6    F    104 Great
7    G    109 Great
8    H    110 Great
```

points is a continuous data and it can be converted to a categorical data

User Retention

- Consider an app ‘Chasing Dragons’, and users pay a monthly subscription fee to use it. The more users, the more money is generated.
- Generally it costs less to keep an existing customer around than to market and advertise to new users.
- Build a model that predicts whether or not a new user will come back next month based on their behavior this month.
- Record each user’s behavior for the first 30 days after sign-up.
- Log every action the user took with timestamp - time-stamped event logs.



Chasing Dragons

Feature generation:

• Number of days the user visited in the first month

• Amount of time until second visit

• Number of points on day j for j = 1, . . .,30 (this would be 30 separate features)

• Total number of points in first month (sum of the other features)

• Did user fill out Chasing Dragons profile (binary 1 or 0)

• Age and gender of user

• Screen size of device

Feature Selection Methods

1. Filters
2. Wrappers
3. Embedded methods

Filters

- Filters select possible features with respect to a ranking based on a metric or statistic, such as correlation with the outcome variable.
- Features are dropped based on their relation to the output, or how they are correlating to the output. A correlation check is done if the features are positively or negatively correlated to the output labels and drop features accordingly.
- Example of a filter: run a linear regression with only that feature as a predictor. Each time, note either the p-value or R-squared, and rank order according to the lowest p-value or highest R-squared.

Feature Selection Methods

R-squared:

- It is the proportion of variance explained by the model.
- The value of R-Squared is always between 0 to 1 (0% to 100%).
- A high R-Squared value means that many data points are close to the linear regression function line.
- A low R-Squared value means that the linear regression function line does not fit the data well.

$$R^2 = 1 - \frac{\Sigma_i\left(y_i - \widehat{y_i}\right)^2}{\Sigma_i\left(y_i - \bar{y}\right)^2}$$

Feature Selection Methods

P-value:

- The P-value is a statistical number to conclude if there is a relationship between dependent and independent variables.
- A low P-value (< 0.05) means that the coefficient is likely not to equal zero.
- A high P-value (> 0.05) means that we cannot conclude that the independent variable affects the dependent variable. A high P-value is also called an insignificant P-value.
- We test if the true value of the coefficient is equal to zero (no relationship). The statistical test for this is called Hypothesis testing.

Hypothesis testing:

- Hypothesis testing is a statistical procedure to test if your results are valid.
- Hypothesis test has two statements. The null hypothesis and the alternative hypothesis.
- The null hypothesis is coefficient is zero. The alternative hypothesis is coefficient is not zero.

Feature Selection Methods

The null hypothesis can either be rejected or not.

If we reject the null hypothesis, we conclude that it exist a relationship between dependent and independent variables.

The P-value is used for this conclusion. A common threshold of the P-value is 0.05.

If the P-value is lower than 0.05, we can reject the null hypothesis and conclude that it exist a relationship between the variables.

Feature Selection Methods

Wrappers

Wrapper feature selection tries to find subsets of features.

Forward selection:

Start with no features and gradually add one feature according to which feature improves the model the most based on a selection criterion.

Backward elimination:

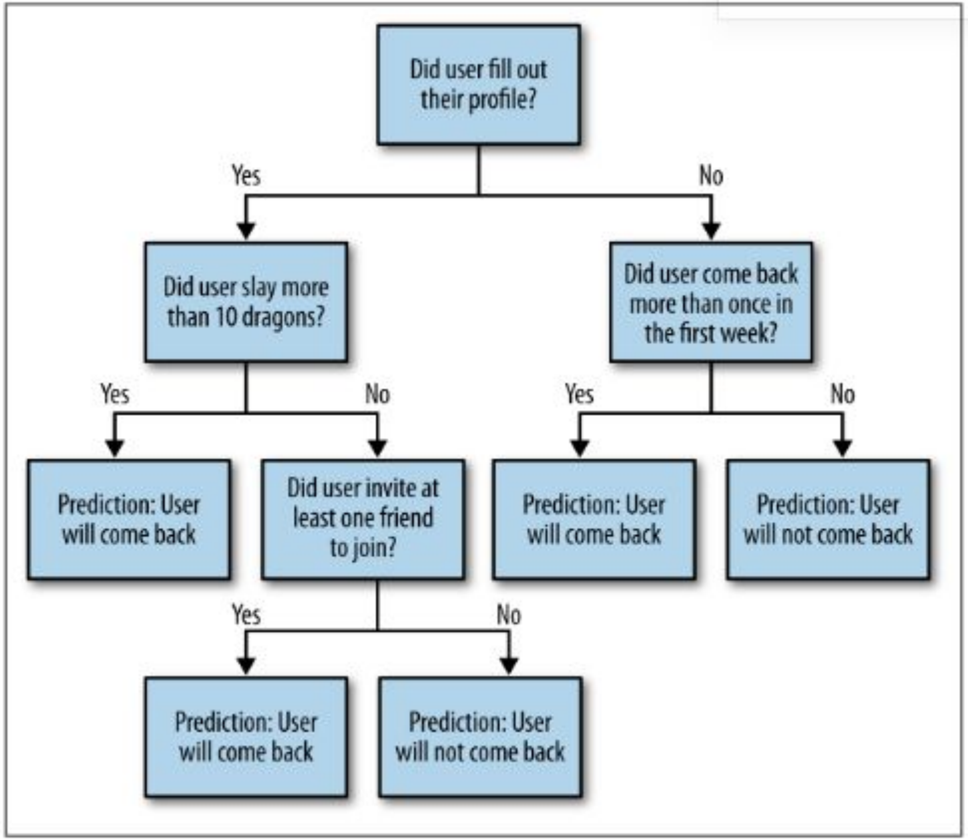
Includes all the features, and gradually remove one feature at a time according to the feature whose removal makes the biggest improvement in the selection criterion. Stop removing features when removing the feature makes the selection criterion get worse.

Combined approach:

Integration of forward selection and backward elimination.

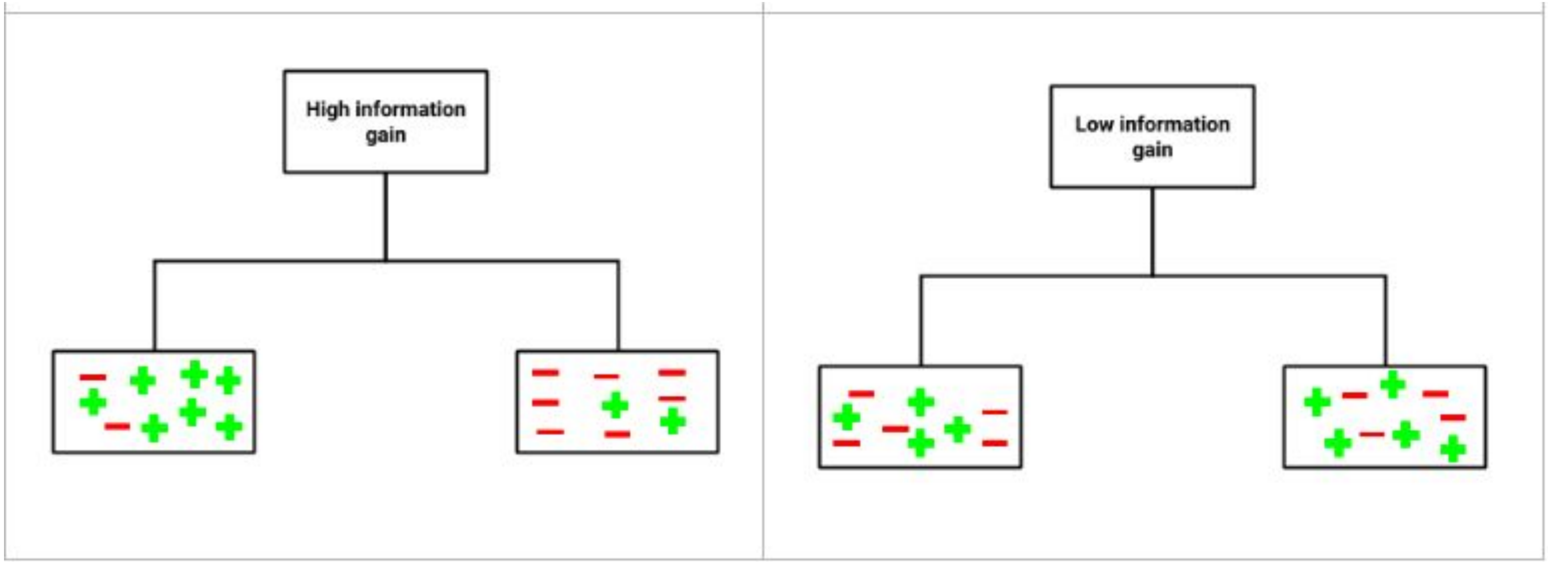
Embedded methods: Decision tree

A decision tree is a classification algorithm.
For the Chasing Dragons example, you want to classify users as “Yes, going to come back next month” or “No, not going to come back next month”.


Did user fill out their profile?
Yes
No
Did user slay more than 10 dragons?
Did user come back more than once in the first week?
Yes
No
Yes
No
Prediction: User will come back
Did user invite at least one friend to join?
Prediction: User will come back
Prediction: User will not come back
Yes
No
Prediction: User will come back
Prediction: User will not come back

Decision tree

Information gain is a statistical property that measures how well a given attribute separates the training examples according to their target classification.



Decision tree

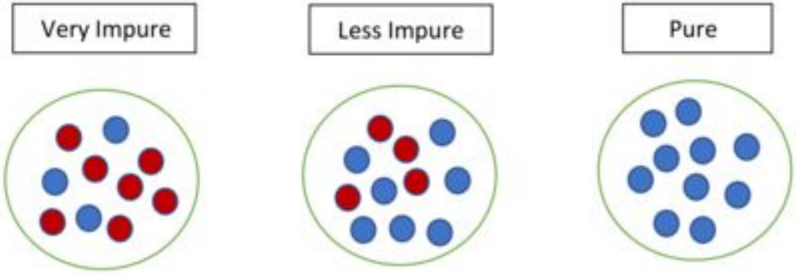
Information Gain:

Information gain is the reduction in information entropy, what is entropy? Basically, entropy is the measure of impurity or uncertainty in a group of observations.

Consider a dataset with N classes. The entropy may be calculated using the formula below:

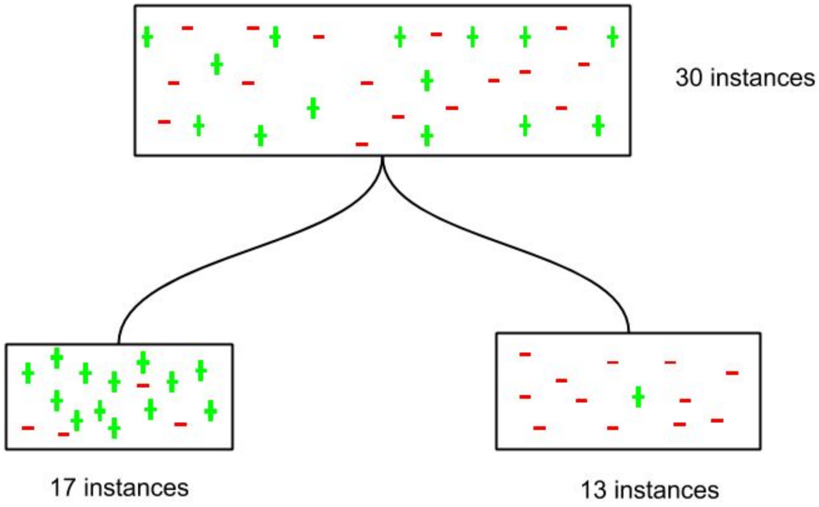
$$E = -\sum_{i=1}^{N} p_i log_2 p_i$$

Where pi is the probability of a tuple belongs to class i.



Decision tree

Suppose a sample (S) has 30 instances (14 positive and 16 negative labels) and an attribute A divides the samples into two subsamples of 17 instances (4 negative and 13 positive labels) and 13 instances (1 positive and 12 negative labels)



$$InformationGain = Entropy(parentnode) - [AverageEntropy(children)]$$

Entropy of parent = Entropy of S

$$= \frac{-14}{30}\log_2\frac{14}{30} - \frac{16}{30}\log_2\frac{16}{30}$$

$$= (-0.47)(-1.0893) - (0.53)(-0.916)$$

$$= 0.996$$

Entropy of children with 17 instances

$$= \frac{-13}{17}\log_2\frac{13}{17} - \frac{4}{17}\log_2\frac{4}{17}$$

$$= (-0.7647)(-0.387) - (0.2352)(-2.088)$$

$$= 0.7869$$

Entropy of children with 13 instances

$$= \frac{-12}{13}\log_2\frac{12}{13} - \frac{1}{13}\log_2\frac{1}{13}$$

$$= 0.391$$

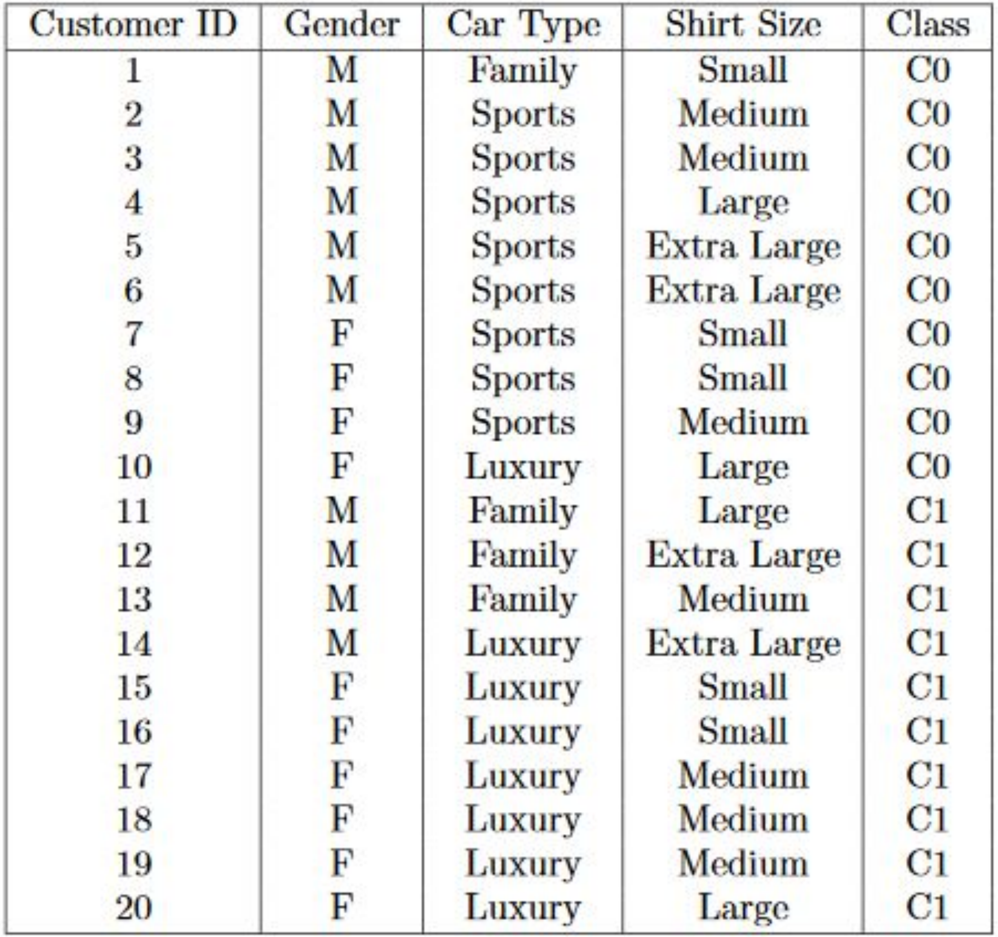
(weighted) Average Entropy of children

$$= \frac{17}{30}(0.7869) + \frac{13}{30}(0.391)$$

$$= 0.615$$

Information Gain $= 0.996 - 0.615$

$$= 0.38$$



Customer ID	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1

Find the attribute with the highest information gain?

Decision tree

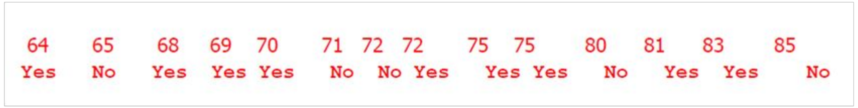
Basic Steps in Decision Tree Construction

1. Tree starts at the root node representing all data.
2. If samples are all same class then node becomes a leaf labeled with class label.
3. Otherwise, select feature that maximizes the information gain
4. Recursion stops when:
    a. Samples in node belong to the same class.
    b. There are no remaining attributes on which to split. In this case, you take the majority vote.

This is an example of an embedded feature selection algorithm. You don’t need to use a filter here because the information gain method is doing your feature selection for you.

DEALING WITH CONTINUOUS VARIABLES

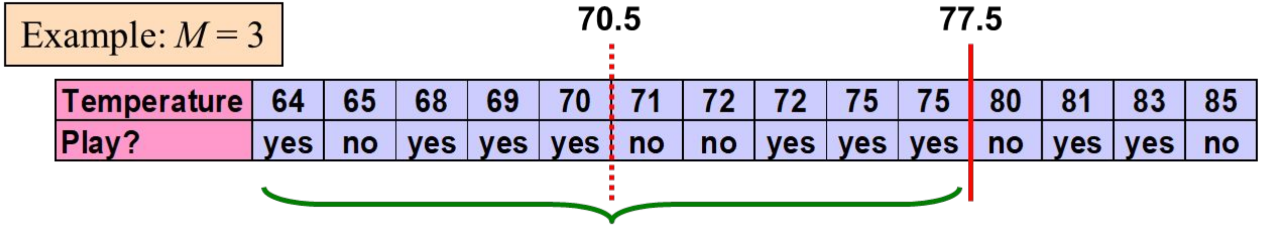
- Partition continuous attribute into a discrete set of intervals
  - sort the examples according to the continuous attribute ***A***
  - identify adjacent examples that differ in their target classification and place breakpoint
  - problem: may generate too many intervals



```
 64   65   68   69   70    71  72  72    75   75    80    81   83    85
Yes   No  Yes  Yes  Yes   No  No Yes   Yes  Yes    No   Yes  Yes    No
```

- Another Solution:
  - take a minimum threshold *M* of the examples of the majority class in each adjacent partition; then merge adjacent partitions with the same majority class

Example: $M = 3$



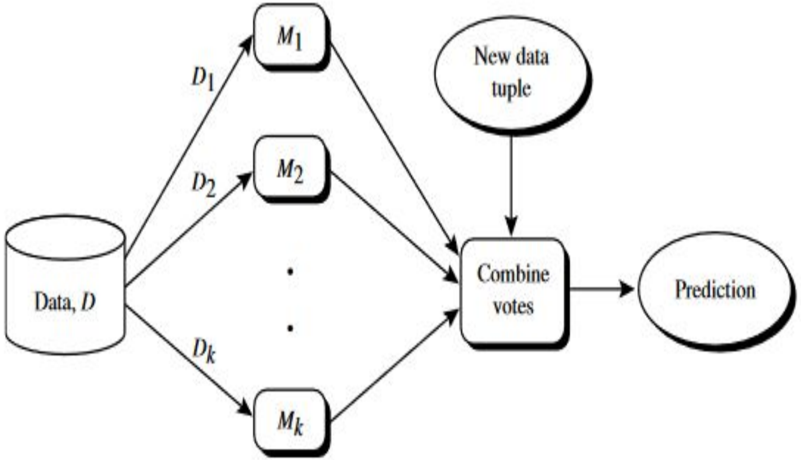
Temperature	64	65	68	69	70	71	72	72	75	75	80	81	83	85
Play?	yes	no	yes	yes	yes	no	no	yes	yes	yes	no	yes	yes	no

Same majority, so they are merged

Final mapping: temperature ≤ 77.5 ==> "yes"; temperature > 77.5 ==> "no"

Random Forest

- Random forest is an ensemble method.
- An ensemble combines a series of k learned models (or base classifiers), M1, M2, . . . , Mk , with the aim of creating an improved composite classification model, M∗.
- An ensemble tends to be more accurate than its base classifiers.
- Each of the classifiers in the ensemble is a decision tree classifier so that the collection of classifiers is a “forest”.
- The individual decision trees are generated using a random selection of attributes at each node to determine the split.



Ensemble method

Random forest

It is an ensemble technique that makes use of multiple decision trees.

To construct a random forest, you construct N decision trees as follows:

1. For each tree, take a bootstrap sample of your data, and for each node you randomly select F features, say 5 out of the 100 total features.

   A bootstrap sample is a sample with replacement, which means we might sample the same data point more than once.

1. Use the entropy-information-gain to decide which among those features you will split your tree on at each stage.

Random Forest

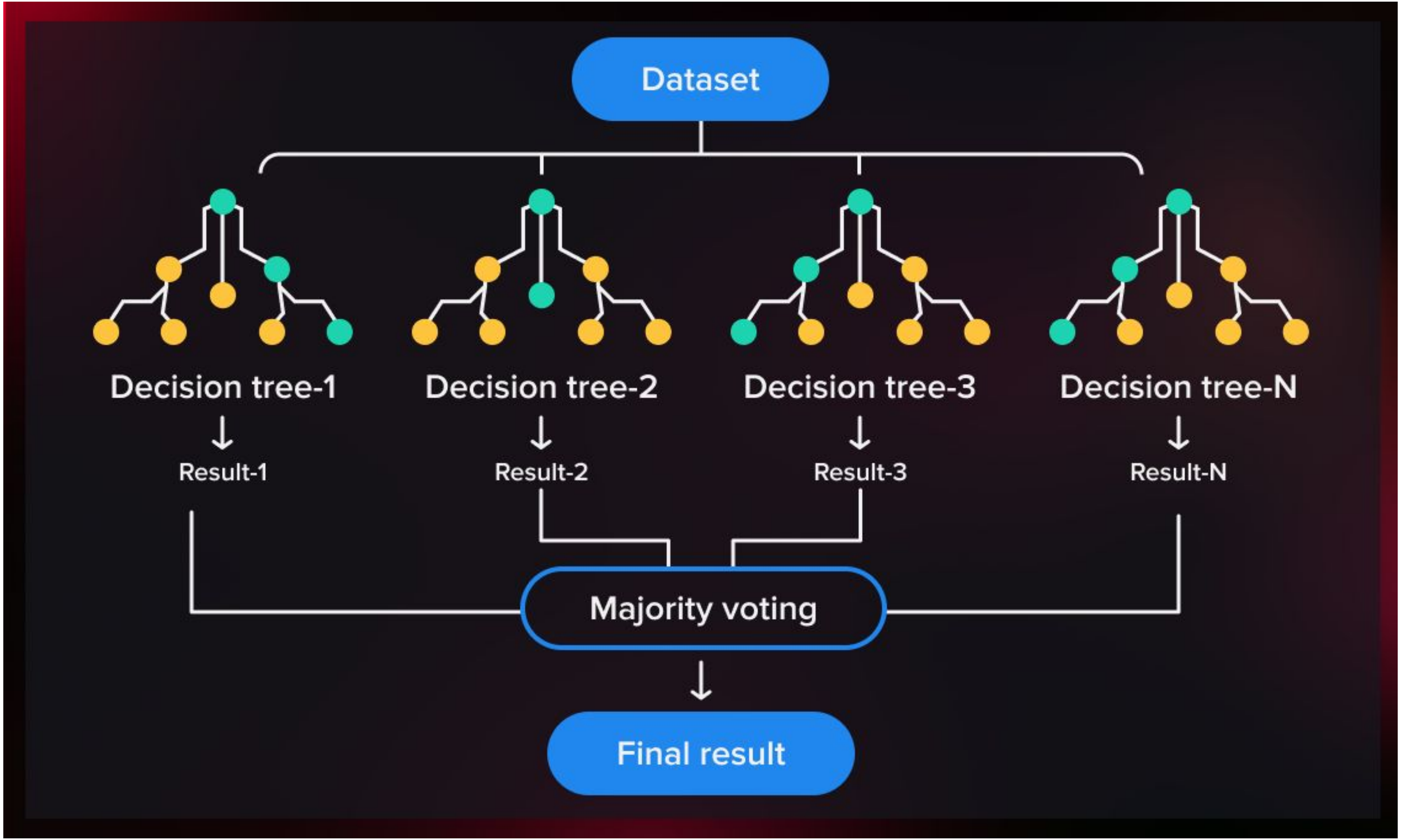
Steps in Random Forest

Step 1: In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.



Dataset

Decision tree-1

Decision tree-2

Decision tree-3

Decision tree-N

Result-1

Result-2

Result-3

Result-N

Majority voting

Final result