MACHINE LEARNING CLASSIFIER (SESSION- 2018-19)

NAÏVE BAYES

Naive Bayes is among one of the most simple and powerful algorithms for classification based on Bayes' Theorem with an assumption of independence among predictors.

Naive Bayes model is easy to build and particularly useful for very large data sets.

There are two parts to this algorithm:

1. Naïve
2. Bayes

NAÏVE ?

The Naïve Bayes classifier assumes that the presence of a feature in a class is unrelated to any other feature.

Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that a particular fruit is an apple or an orange or a banana and that is why it is known as "Naive".

BAYES?

LIKELIHOOD
The probability of "B" being True, given "A" is True

PRIOR
The probability "A" being True. This is the knowledge.

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

POSTERIOR
The probability of "A" being True, given "B" is True

MARGINALIZATION
The probability "B" being True.

Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event.

BAYES' THEOREM EXAMPLE

Problem : Probability of the Card we picked at random to be a King given that it is a Face Card

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

$$= \frac{1.(1/13)}{3/13} = \mathbf{1/3}$$

$P(\text{King}) = 4/52 = 1/13$

$P(\text{Face}|\text{King}) = 1$

$P(\text{Face}) = 12/52 = 3/13$

BAYES' THEOREM PROOF

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

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

$$P(A \cap B) = P(A|B).P(A) = P(B|A).P(B)$$

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

NAÏVE BAYES WORKING

Day	Outlook	Humidity	Wind	Play
D1	Sunny	High	Weak	No
D2	Sunny	High	Strong	No
D3	Overcast	High	Weak	Yes
D4	Rain	High	Weak	Yes
D5	Rain	Normal	Weak	Yes
D6	Rain	Normal	Strong	No
D7	Overcast	Normal	Strong	Yes
D8	Sunny	High	Weak	No
D9	Sunny	Normal	Weak	Yes
D10	Rain	Normal	Weak	Yes
D11	Sunny	Normal	Strong	Yes
D12	Overcast	High	Strong	Yes
D13	Overcast	Normal	Weak	Yes
D14	Rain	High	Strong	No

Frequency Table		Play	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	3	2

Frequency Table		Play	
		Yes	No
Humidity	High	3	4
	Normal	6	1

Frequency Table		Play	
		Yes	No
Wind	Strong	6	2
	Weak	3	3

CONT….

Likelihood Table		Play		
		Yes	No	
Outlook	Sunny	3/10	2/4	5/14
	Overcast	4/10	0/4	4/14
	Rainy	3/10	2/4	5/14
		10/14	4/14	

$P(x|c) = P(Sunny|Yes) = 3/10 = 0.3$

$P(x) = P(Sunny) = 5/14 = 0.36$

$P(c) = P(Yes) = 10/14 = 0.71$

Likelihood of 'Yes' given Sunny is

$P(c|x) = P(Yes|Sunny) = P(Sunny|Yes)* P(Yes) / P(Sunny)$ = (0.3 x 0.71) /0.36 = 0.591

Similarly Likelihood of 'No' given Sunny is

$P(c|x) = P(No|Sunny) = P(Sunny|No)* P(No) / P(Sunny)$ = (0.4 x 0.36) /0.36 = 0.40

CONT….

Likelihood table for Humidity

Likelihood Table		Play		
		Yes	No	
Humidity	High	3/9	4/5	7/14
	Normal	6/9	1/5	7/14
		9/14	5/14	

Likelihood table for Wind

Likelihood Table		Play		
		Yes	No	
Wind	Weak	6/9	2/5	8/14
	Strong	3/9	3/5	6/14
		9/14	5/14	

P(Yes|High) = 0.33 x 0.6 / 0.5 = 0.42

P(No|High) = 0.8 x 0.36 / 0.5 = 0.58

P(Yes|Weak) = 0.67 x 0.64 / 0.57 = 0.75

P(No|Weak) = 0.4 x 0.36 / 0.57 = 0.25

CONT….

Suppose we have a day with the following values

Outlook	=	Rain
Humidity	=	High
Wind	=	Weak
Play	=	?

Likelihood of 'Yes' on that Day = P(Outlook = Rain|Yes)*P(Humidity= High|Yes)* P(Wind= Weak|Yes)*P(Yes)
= 2/9 * 3/9 * 6/9 * 9/14 = 0.0199

Likelihood of 'No' on that Day = P(Outlook = Rain|No)*P(Humidity= High|No)* P(Wind= Weak|No)*P(No)
= 2/5 * 4/5 * 2/5 * 5/14 = 0.0166

CONT….

P(Yes) = 0.0199 / (0.0199+ 0.0166) = 0.55

P(No) = 0.0166 / (0.0199+ 0.0166) = 0.45

NAÏVE BAYES IN THE INDUSTRY

TYPES OF NAÏVE BAYES

WHAT IS THE SUPPORT VECTOR MACHINE ?

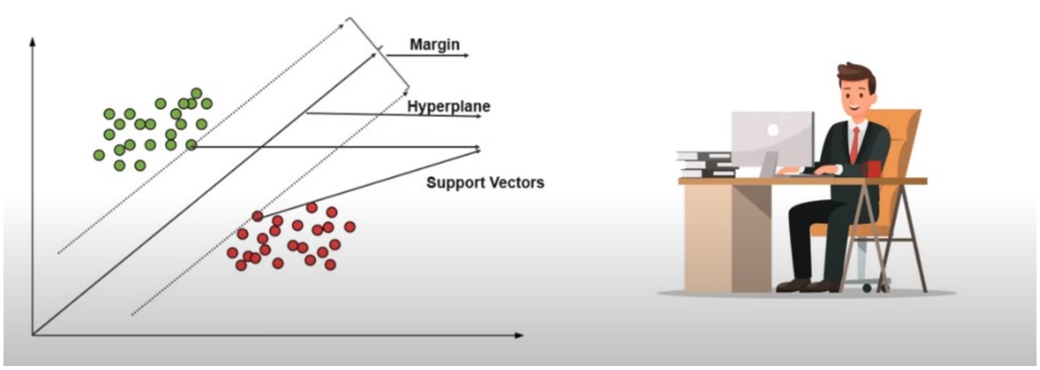
- A Support Vector Machine was first introduced in the 1960s and later improvised in the 1990s.
- It is a supervised learning machine learning classification algorithm.
- An SVM is implemented in a slightly different way than other machine learning algorithms. It is capable of performing classification, regression and outlier detection.
- Support Vector Machine is a discriminative classifier that is formally designed by a separative hyperplane.
- SVM can also perform non-linear classification.

ADVANTAGES & DISADVANTAGES

- Effective in high dimensional spaces
- Still effective in cases where the number of dimensions is greater than the number of samples
- Uses a subset of training points in the decision function that makes it memory efficient
- Different kernel functions can be specified for the decision function that also makes it versatile

- If the number of features is much larger than the number of samples, avoid over-fitting in choosing kernel functions and regularization term is crucial.
- SVMs do not directly provide probability estimates, these are calculated using five-fold cross-validation.

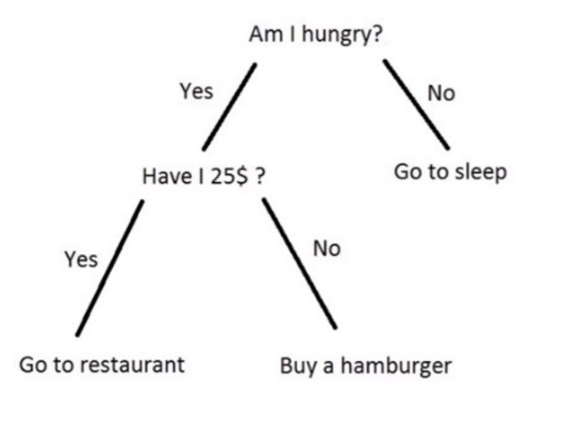
WORKING



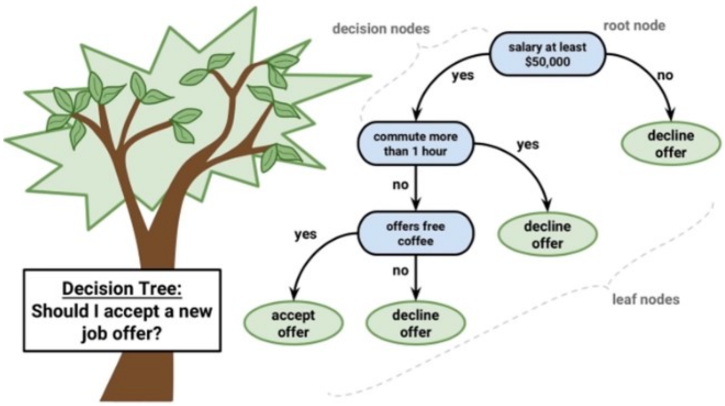
SVM KERNELS

- An SVM kernel basically adds more dimensions to a low dimensional space to make it easier to segregate the data.
- It converts the inseparable problem to separable problems by adding more dimensions using the kernel trick.
- A support vector machine is implemented in practice by a kernel.
- The kernel trick helps to make a more accurate classifier.
- Different kernels:
  - Linear Kernel
  - Polynomial Kernel
  - Radial Basis Function Kernel

DECISION TREE

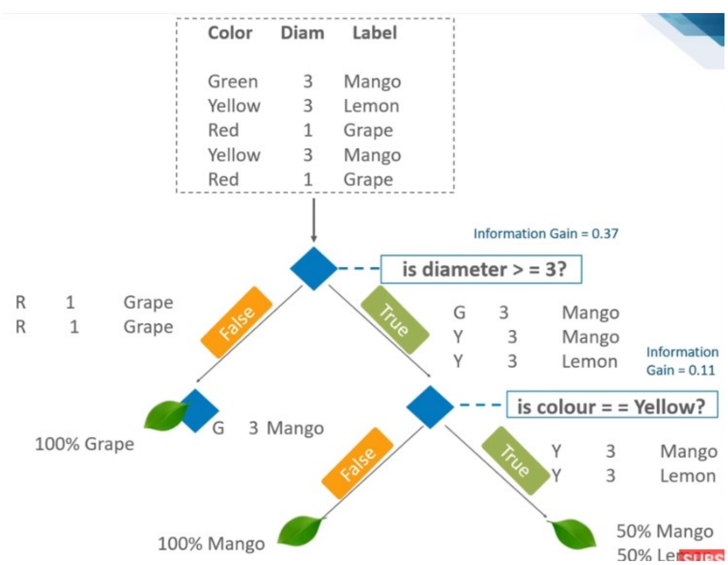


WHAT IS DECISION TREE?

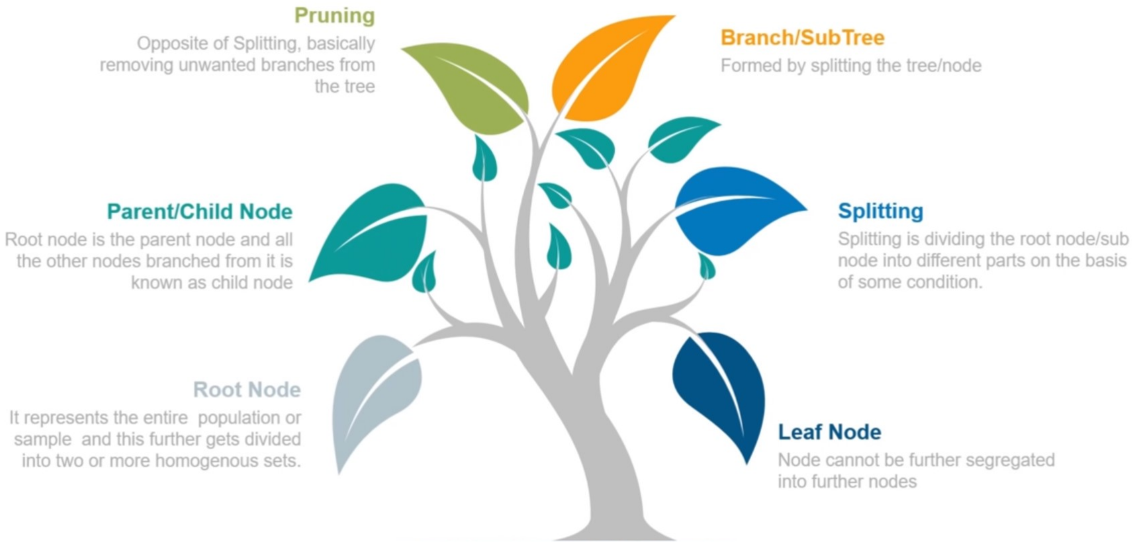


A decision tree is a graphical representation of all the possible solution to a decision based on certain conditions.

EXAMPLE



DECISION TREE TERMINOLOGY



CART (CLASSIFICATION & REGRESSION TREES) ALGORITHM

The algorithm is based on Classification and Regression Trees by Breiman et al (1984). A CART tree is a binary decision tree that is constructed by splitting a node into two child nodes repeatedly, beginning with the root node that contains the whole learning sample.

The main elements of CART (and any decision tree algorithm) are:

- Rules for splitting data at a node based on the value of one variable;
- Stopping rules for deciding when a branch is terminal and can be split no more; and
- Finally, a prediction for the target variable in each terminal node.

EXAMPLE

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Q: Which one among them should you pick first?

Ans: Determine the attribute that best classifies the training data.

Q: How do we choose the best attribute?

OR

How does a tree decide where to split?

HOW DOES A TREE DECIDE WHERE TO SPLIT?

Gini Index

The measure of impurity (or purity) used in building decision tree in CART is Gini Index

Information Gain

The information gain is the decrease in entropy after a dataset is split on the basis of an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain

Chi Square

It is an algorithm to find out the statistical significance between the differences between sub-nodes and parent node

Reduction in Variance

Reduction in variance is an algorithm used for continuous target variables (regression problems). The split with lower variance is selected as the criteria to split the population

BUILD OUR DECISION TREE (STEP 1: COMPUTE THE ENTROPY FOR THE DATASET)

Out of 14 instances we have 9 YES and 5 NO

So we have the formula,

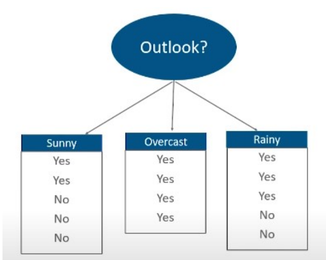
E(S) = -P(Yes) $\log_2 P(Yes)$ – *P(No)* $\log_2 P(No)$

*E(S) = - (9/14)** $\log_2 9/14$ - *(5/14)** $\log_2 5/14$

E(S) = 0.41+0.53 = 0.94

	outlook	temp.	humidity	windy	play
D1	sunny	hot	high	false	no
D2	sunny	hot	high	true	no
D3	overcast	hot	high	false	yes
D4	rainy	mild	high	false	yes
D5	rainy	cool	normal	false	yes
D6	rainy	cool	normal	true	no
D7	overcast	cool	normal	true	yes
D8	sunny	mild	high	false	no
D9	sunny	cool	normal	false	yes
D10	rainy	mild	normal	false	yes
D11	sunny	mild	normal	true	yes
D12	overcast	mild	high	true	yes
D13	overcast	hot	normal	false	yes
D14	rainy	mild	high	true	no

BUILD OUR DECISION TREE (STEP 2: WHICH NODE TO SELECT AS ROOT NODE)



E(Outlook = Sunny) = -2/5 $\log_2 2/5 - 3/5 \log_2 3/5$ *= 0.971*

E(Outlook = Overcast) = -1 $\log_2 1 - 0 \log_2 0$ *= 0*

E(Outlook = Sunny) = -3/5 $\log_2 3/5 - 2/5 \log_2 2/5$ *= 0.971*

Information from outlook,

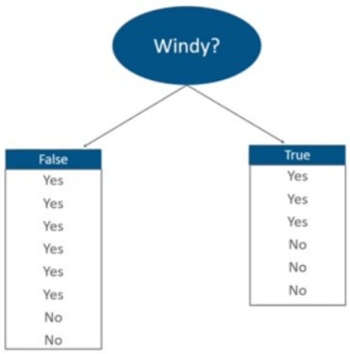
I(Outlook) = 5/14 x 0.971 + 4/14 x 0 + 5/14 x 0.971 = 0.693

Information gained from outlook,

Gain(Outlook) = E(S) – I(Outlook)

0.94 – 0.693 = 0.247

CONT….



E(Windy = True) = 1

E(Windy = False) = 0.811

Information from windy,

I(Windy) = 8/14 x 0.811+ 6/14 x 1 = 0.892

Information gained from outlook,

Gain(Windy) = E(S) – I(Windy)

0.94 – 0.892 = 0.048

CONT…..

Outlook:
Info 0.693
Gain: 0.940-0.693 $\boxed{0.247}$

Temperature:
Info 0.911
Gain: 0.940-0.911 0.029

Humidity:
Info 0.788
Gain: 0.940-0.788 0.152

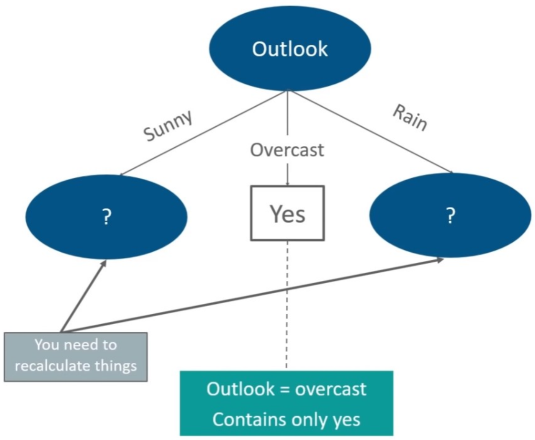
Windy:
Info 0.892
Gain: 0.940-0.982 0.048

Since Max gain = 0.247,

Outlook is our ROOT Node

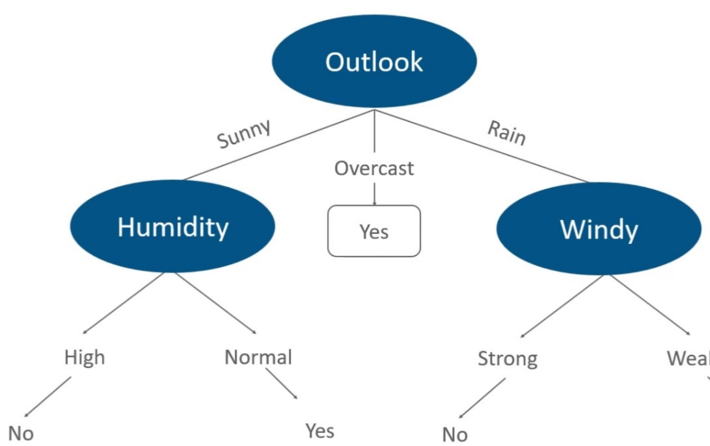
outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

CONT….



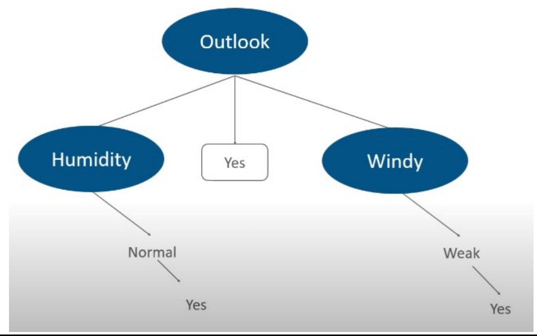
outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

THIS IS HOW YOUR COMPLETE TREE WILL LOOK LIKE

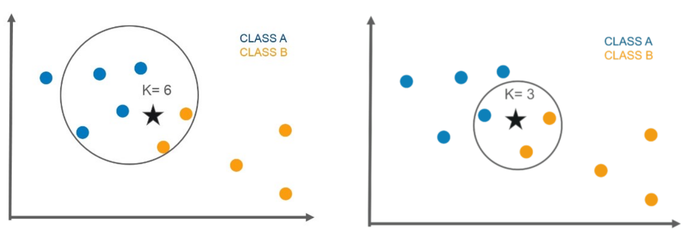


WHAT IS PRUNING?

A decision tree is a graphical representation of all the possible solution to a decision based on certain condition.



KNN



K Nearest Neighbor is a simple algorithm that stores all the available cases and classifies the new data or case based on a similarity measure.

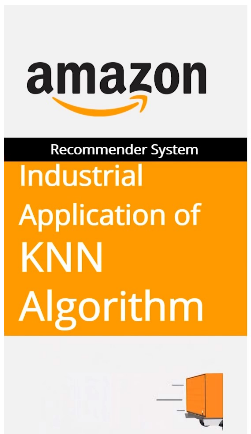
Q: What does 'k' in KNN Algorithm represent?

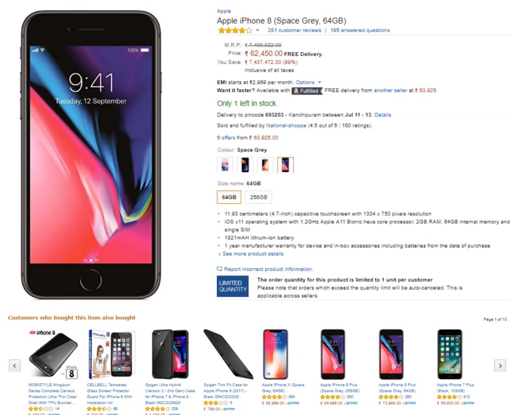
Ans: k in KNN algorithm represents the number of nearest neighbor points which are voting for the new test data's class.

Note:

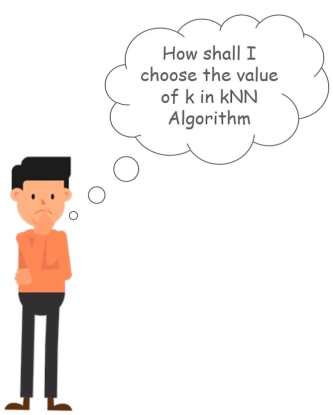
- If k=1, then test examples are given the same label as the closest example in the training set.
- If k=3, the labels of the three closest classes are checked and the most common (i.e., occurring at least twice) label is assigned, and so on for larger ks.

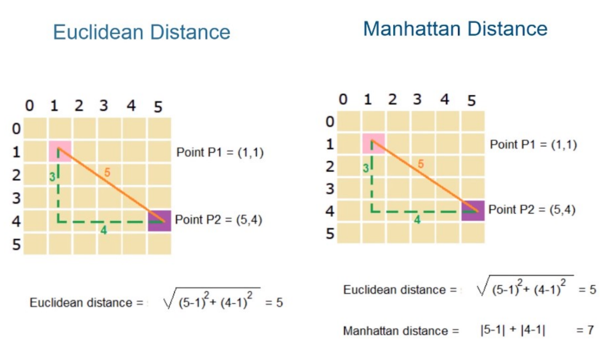
APPLICATION OF KNN IN INDUSTRY





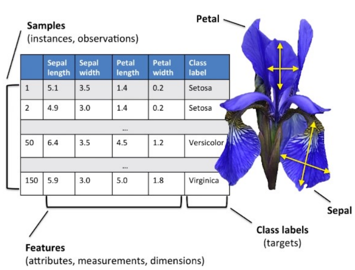
HOW TO CHOOSE THE VALUE OF K IN KNN ALGORITHM



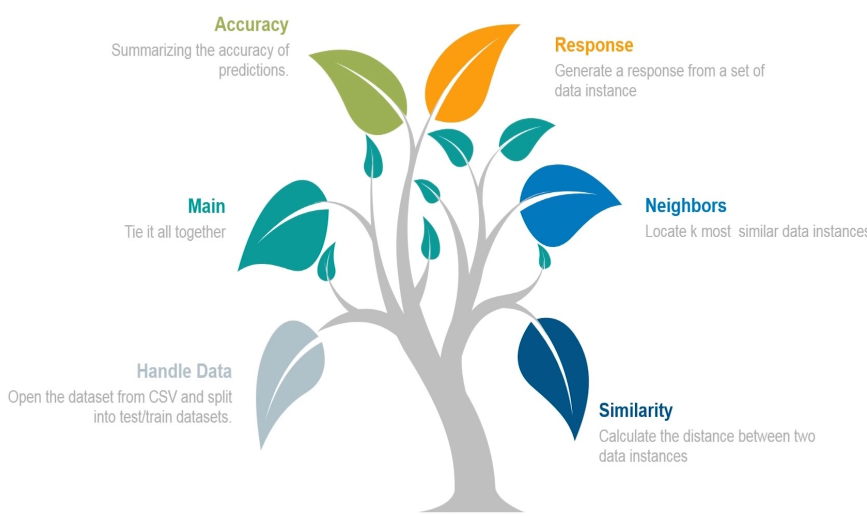


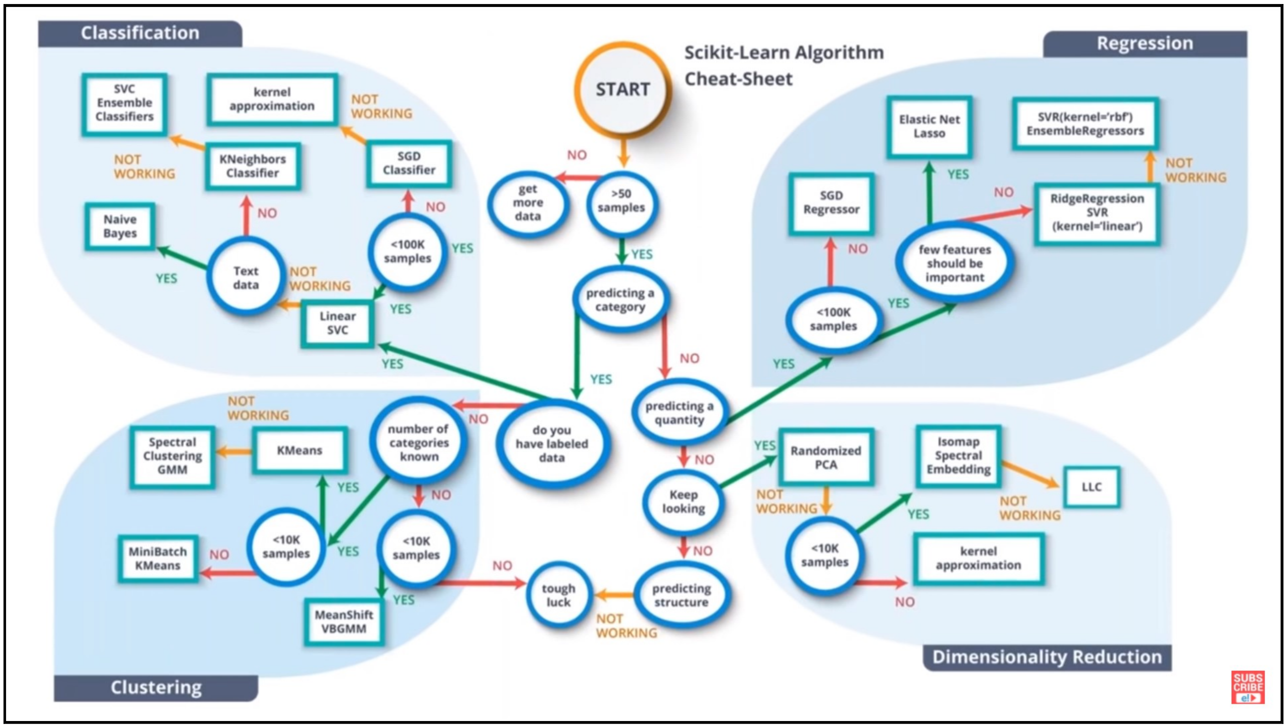
KNN IS LAZY LEARNER

EXAMPLE



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica





Classification
Regression
Clustering
Dimensionality Reduction
START
Scikit-Learn Algorithm Cheat-Sheet
>50 samples
get more data
predicting a category
do you have labeled data
<100K samples
SGD Classifier
kernel approximation
KNeighbors Classifier
SVC Ensemble Classifiers
Text data
Naive Bayes
Linear SVC
number of categories known
<10K samples
KMeans
Spectral Clustering GMM
MiniBatch KMeans
MeanShift VBGMM
tough luck
predicting a quantity
few features should be important
Elastic Net Lasso
RidgeRegression SVR (kernel='linear')
SVR(kernel='rbf') EnsembleRegressors
SGD Regressor
Keep looking
predicting structure
Randomized PCA
Isomap Spectral Embedding
LLC
kernel approximation
YES
NO
NOT WORKING

Thank you… ☺