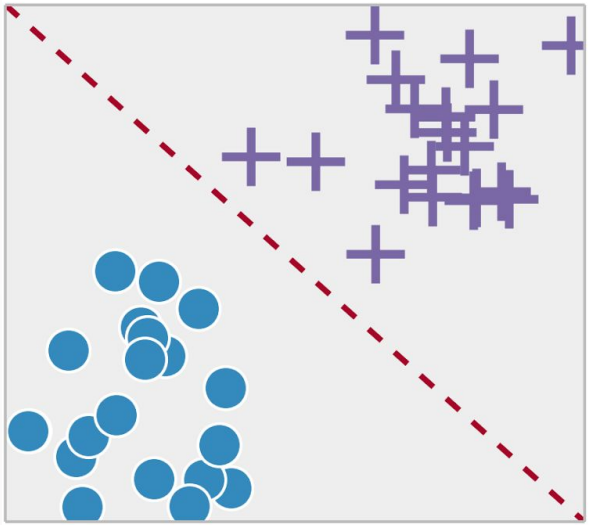
Classification Algorithms & Use-Cases

Analytiks

Supervised: Classification

Which of a set of categories a new observation belongs to, based on a training dataset containing observations whose category membership is known



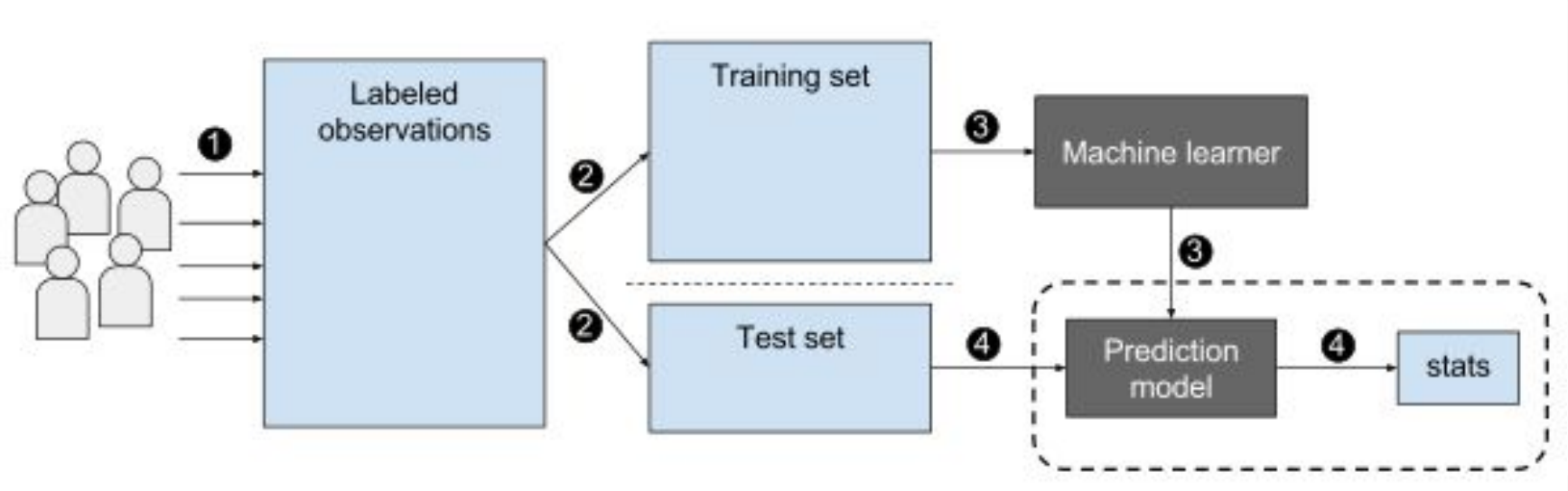
Two Types: 2-Class / Binary | Multi-Class

Examples:

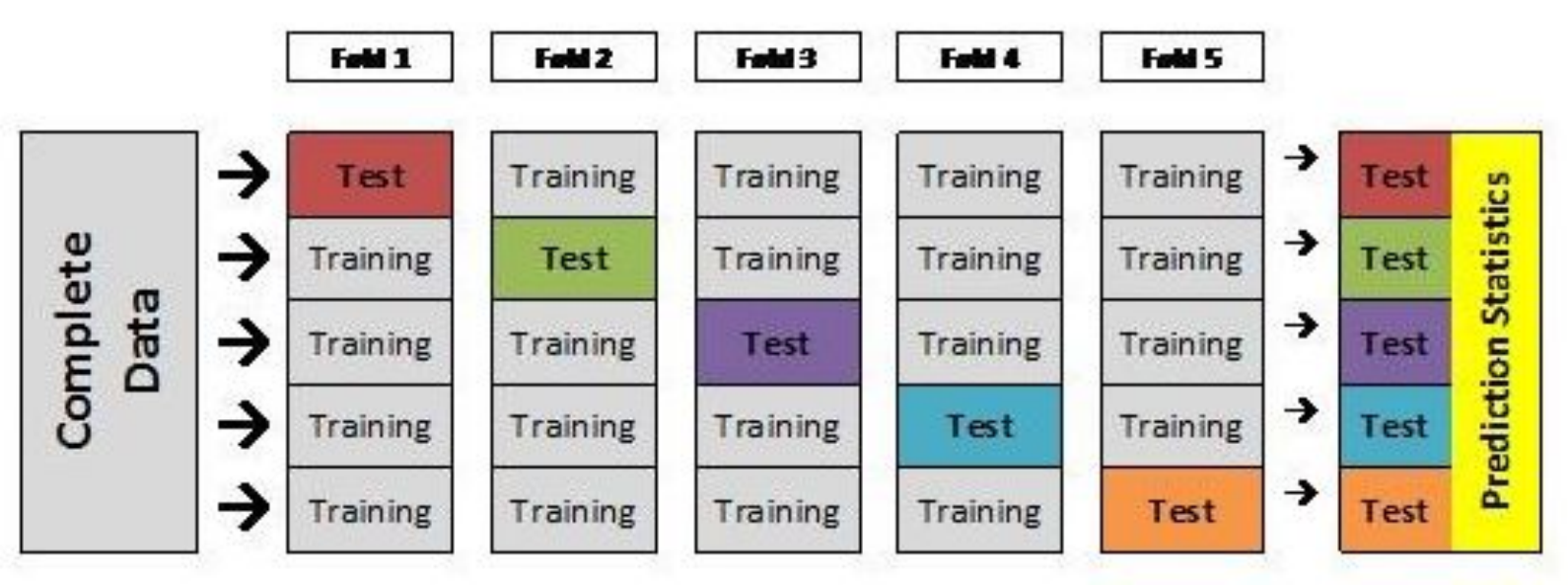
- Spam Filtering
- Risk Analysis
- Churn Analysis
- Medical Diagnosis
- Fraud Detection
- Employee Retention

First, A Review

How does a machine learn?



Second: A new validation technique - Cross Validation



Algorithms

Classification

Algorithms

- Logistic Regression
- K Nearest Neighbor (KNN)
- Naïve Bayes
- Decision Trees

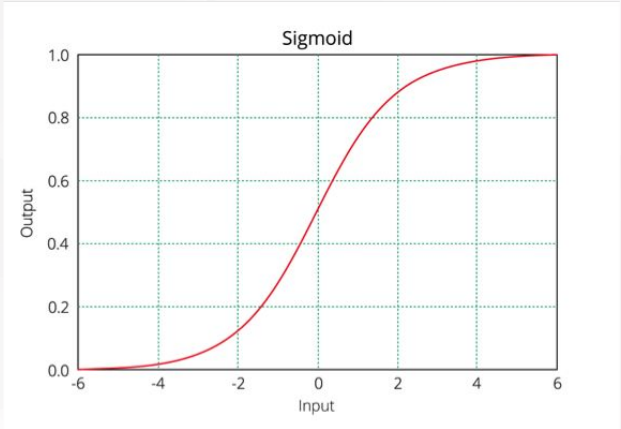
Logistic Regression: Overview

- Used to describe the data and the relationship between the class/target variable and the predictor variable(s)
- How does the probability of getting lung cancer change for every additional pound of overweight and for every X cigarettes smoked per day?
- Do body weight calorie intake, fat intake, and age have an influence on heart attacks (yes vs. no)?
- Assumptions:
    - Prediction must be binary
    - No outliers in the data

Logistic Regression: Formula

log-odds of a categorical response being "true" (1) is modeled as a linear combination of the features:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$



Can be rearranged into the logistic function:

$$P(Y=1|X) = \frac{e^{(\beta_o+\beta_1 x)}}{e^{(\beta_o+\beta_1 x)}+1}$$

- Outputs the probabilities of class “true” (1)
- Probabilities can be converted into class predictions

- Where:
  - e - 2.71828; Euler's number
  - β_0 - intercept
  - β_1 - coefficient of X

- The logistic function has some nice properties:
  - Takes on an "s" shape
  - Output is bounded by 0 and 1

Logistic Regression: Pros and Cons

+	-
• Fast prediction • Allows understanding of relationship between variables • Doesn't require any tuning • Outputs well-calibrated predicted probabilities variables	• Presumes a linear relationship between the features and the log-odds of the response • Performance is (generally) not competitive with the best supervised learning methods • Can't automatically learn feature interactions

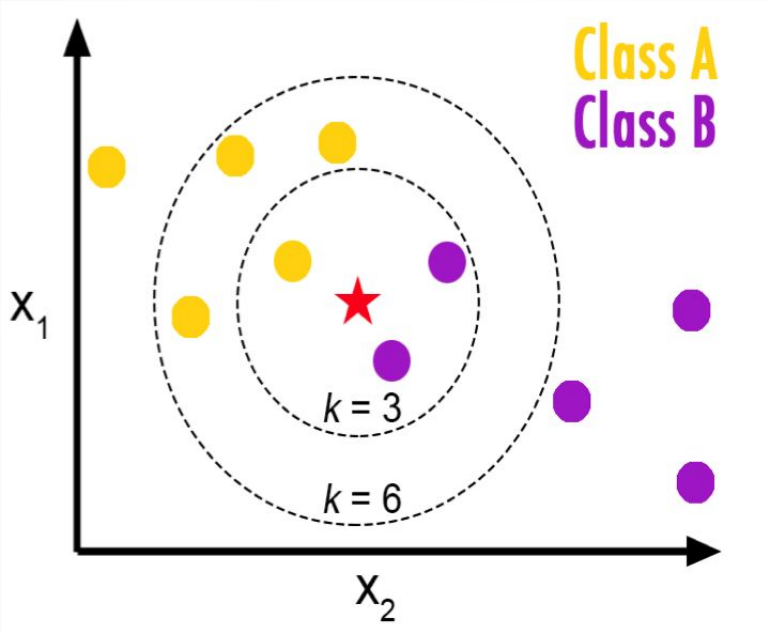
LAB:
Logistic Regression

Analytiks

K-Nearest Neighbors: Overview

An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors

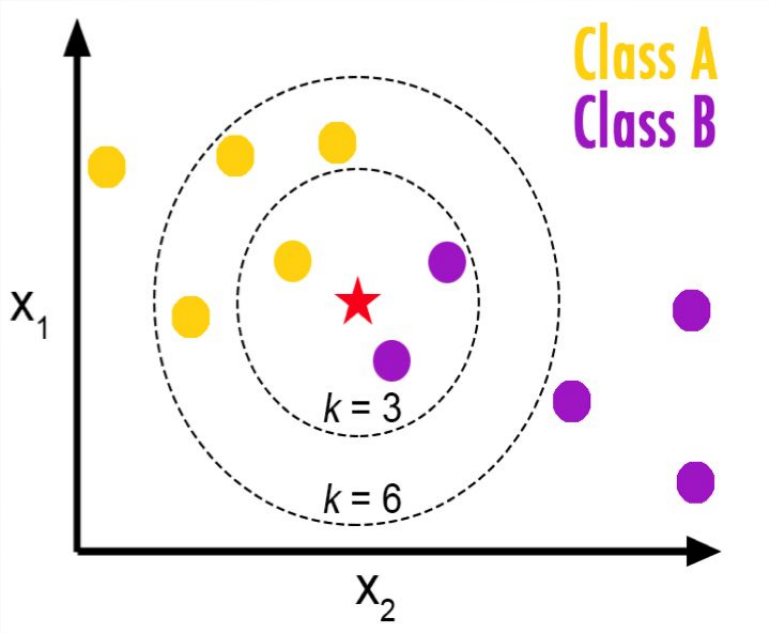
- k is a positive integer, typically small
- If k = 1, then the object is simply assigned to the class of that single nearest neighbor.
- When to Use:
  - Use when dataset is small
  - Input: the k closest training examples in the feature space
  - Output: class membership



K-Nearest Neighbors: Prediction

Predict x's class by:

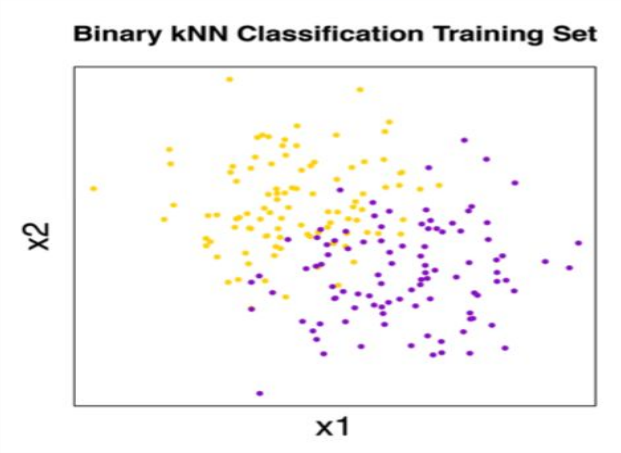
- Look at training data
- [k = 3, k = 6]
- Find x's nearest neighbor from the training data
- The label of x’s nearest neighbor becomes x's label
- Can use euclidean distance in finding the nearest neighbor/point

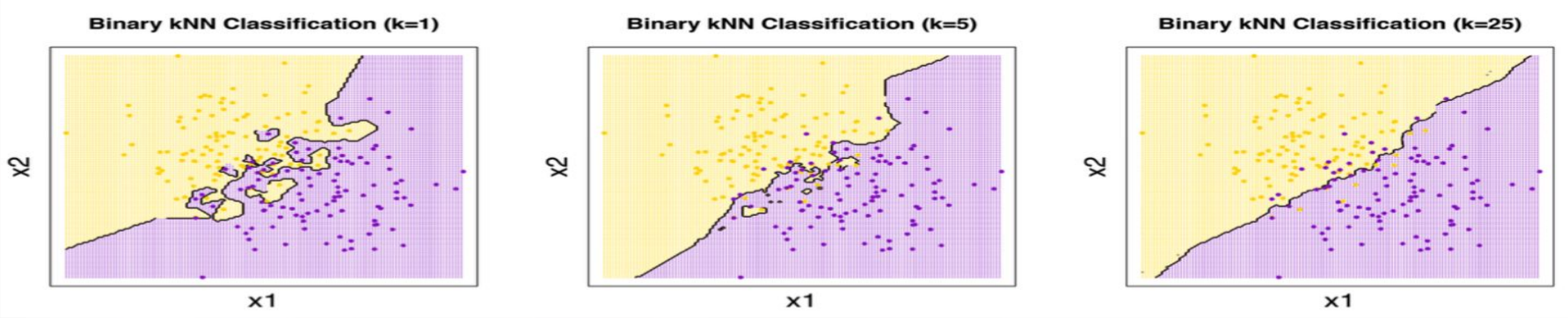


K-Nearest Neighbors: Effect of K

Large k produces smoother boundaries

If k is too large: oversimplified boundaries; always predict the majority class





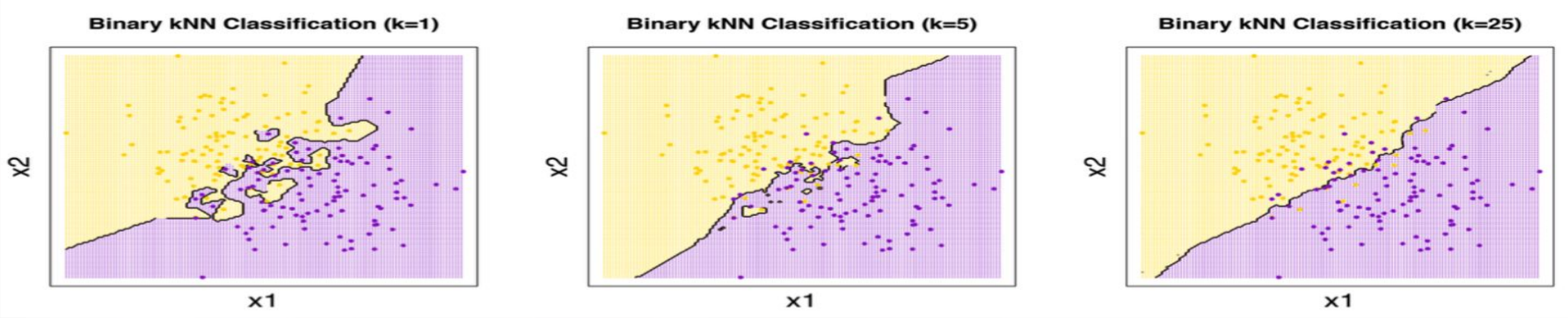
K-Nearest Neighbors: Choosing K

Choosing K

- Rule of thumb: square root of the number of training samples
- Choose an odd number
- Based on training error
    - Cross validation

10-FOLD CROSS VALIDATION

- Break data into 10 sets of size n/10.
- Train on 9 datasets and test on 1.
- Repeat 10 times and take a mean accuracy.



K-Nearest Neighbors: Pros and Cons

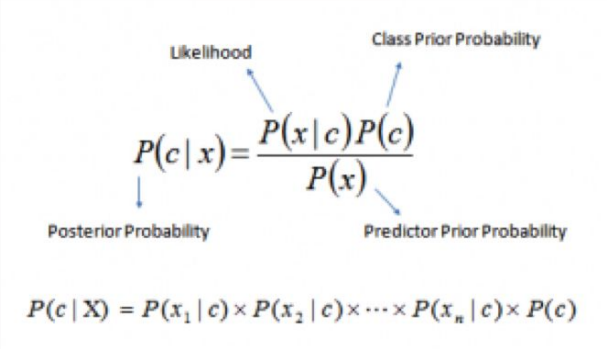
+	-
• Simple to implement • Quickly responds to changes in input • Able to learn non-linear boundaries learned by local approximation • Does not assume any probability distributions on the input data • Good for Small Data Sets	• Computationally expensive • Lazy learner - does not generate a model; does nothing until a new observation is given for prediction • Not recommended for high dimensional and Large Data sets

LAB:
K-Nearest Neighbors

Analytiks

Naïve Bayes: Overview

Basis: Bayes theorem with the assumption of independence among predictors (i.e., the presence of a particular feature in a class is unrelated to the presence of any other feature)



Where:

- P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

Naïve Bayes: When to Use?

- Have an initial hypothesis and understanding of the data
- Features have been reduced so that they are independent
- Need a fast predictor
- Huge data points but few features in the training data
- Ideal Applications:
  - Real time prediction
  - Multi class prediction
  - Text classification / Spam Filtering / Sentiment Analysis
  - Recommendation System

Naïve Bayes: Example

Weather	Traffic	Worked
Sunny	Medium	No
Rainy	Heavy	Yes
Rainy	Medium	No
Cloudy	Heavy	Yes
Cloudy	Light	Yes
Sunny	Heavy	Yes
Rainy	Medium	No
Sunny	Light	Yes
Rainy	Medium	Yes
Sunny	Light	Yes
Cloudy	Heavy	No
Rainy	Light	No

- Let's assume that:
  - Weather = Cloudy
  - Traffic = Medium
- **Will he/she go to work?**
- To Solve:
  - Convert data into frequency table
  - Create likelihood table by computing for the probabilities
  - Use NB equation to calculate the posterior probability for each class.
  - The class with the highest posterior probability is the outcome of prediction.

Naïve Bayes: Example

Weather	Yes	No
Sunny	3 (3/7)	1 (1/5)
Rainy	2 (2/7)	3 (3/5)
Cloudy	2 (2/7)	1 (1/5)

Traffic	Yes	No
Light	3 (3/7)	1 (1/5)
Medium	1 (1/7)	3 (3/5)
Heavy	3 (3/7)	1 (1/5)

P(Yes) = 7/12
P(No) = 5/12

L(Yes | (Cloudy, Medium) = P(Cloudy|Yes) * P (Medium|Yes) * P(Yes)
L(Yes | (Cloudy, Medium) = (2/7) * (1/7) * (7/12)
L(Yes | (Cloudy, Medium) = 0.02

L(No | (Cloudy, Medium) = P(Cloudy|No) * P (Medium|No) * P(No)
L(No | (Cloudy, Medium) = (1/5) * (3/5) * (5/12)
L(No | (Cloudy, Medium) = 0.05

P(Yes | (Cloudy, Medium) = 0.02 / (0.02 + 0.05) = 32%
P(No | (Cloudy, Medium) = 0.05 / (0.02 + 0.05) = 68 %

Naïve Bayes: Pros and Cons

+	-
• Fast prediction • Easy to build • Good for multiclass prediction • Useful for large datasets • Can perform better than other algorithms • Need less training data • Performs well in case of categorical input variables compared to numerical variables	• "Zero Frequency" issue • Limitation due to the assumption of independent predictors

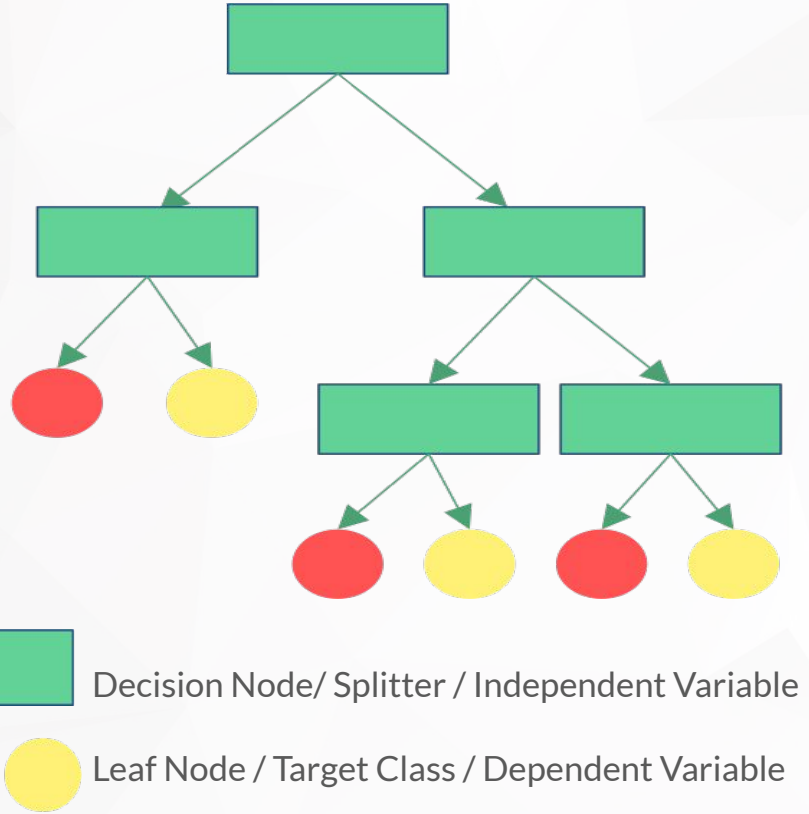
LAB:
Naïve Bayes

Analytiks

Decision Tree: Overview

A classifier in the form of a tree structure.

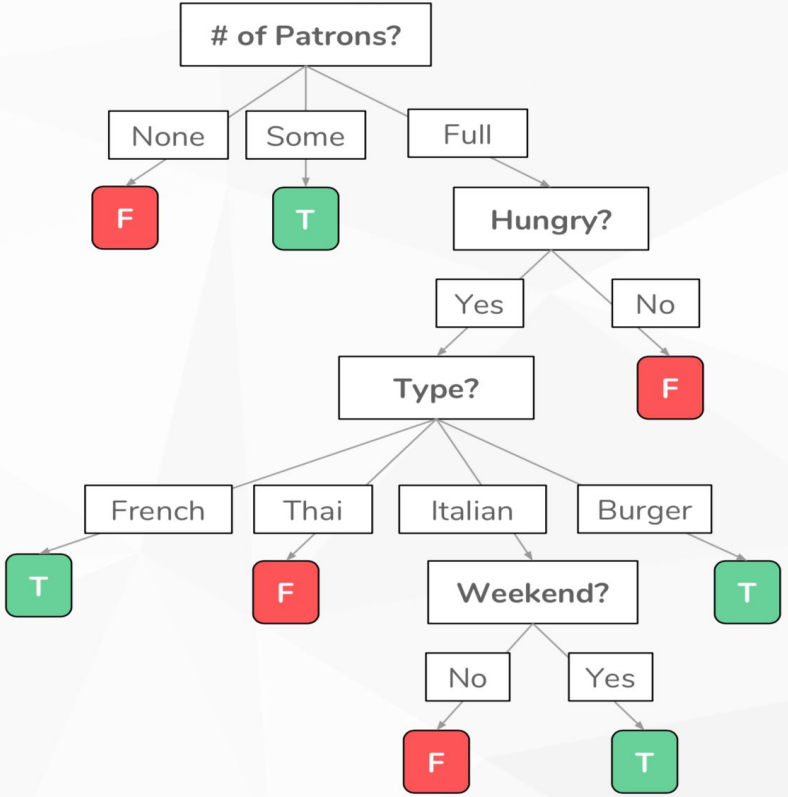
- It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.
- Powerful and popular tool for classification
- Applications include:
  - Customer segmentation
  - Medical Text Classification
  - Credit Scoring
- Key requirements:
  - Predefined classes
  - Discrete classes
  - Sufficient data

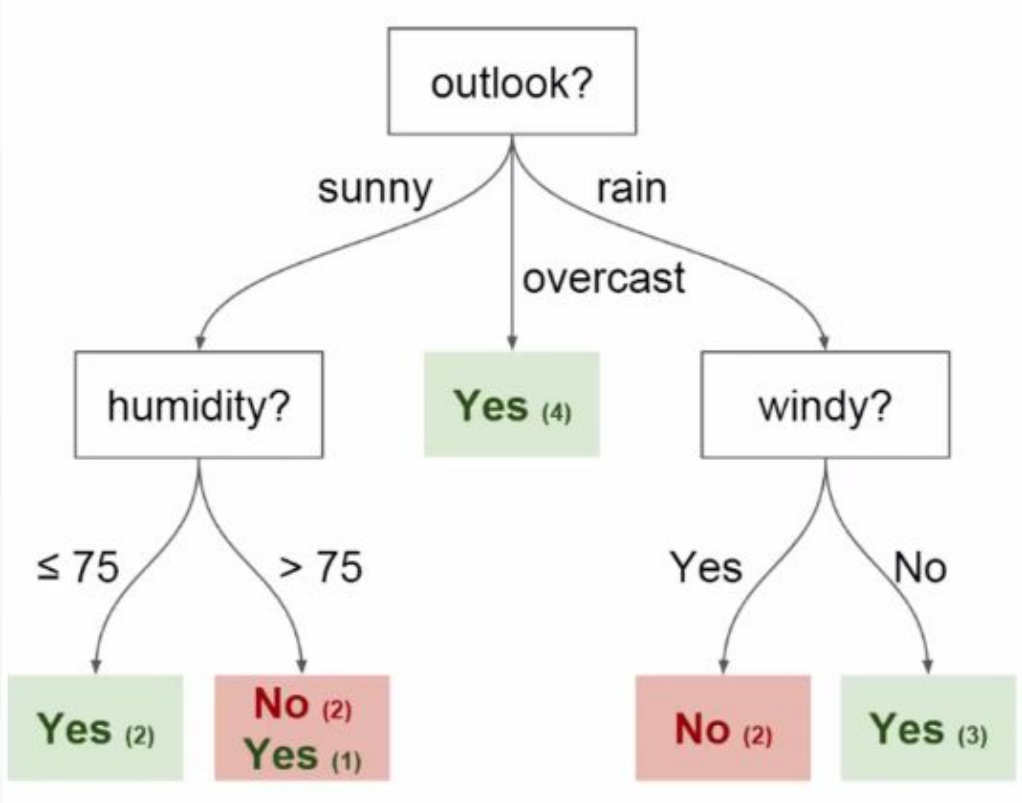


Decision Tree: Example

Decision: Will a customer wait for a table or not?

Classification process: start at the root and move through until the machine reaches a leaf node, which provides the classification of the instance




outlook?
sunny
overcast
rain
humidity?
Yes (4)
windy?
≤ 75
> 75
Yes
No
Yes (2)
No (2)
Yes (1)
No (2)
Yes (3)

Decision Tree: Construction

Most decision tree algorithms use a top-down, greedy search through the space of possible decision trees

1. Rules based on variables' values are selected to get the best split to differentiate observations based on the dependent variable

2. Once a rule is selected and splits a node into two, the same process is applied to each "child" node (i.e. it is a recursive procedure)

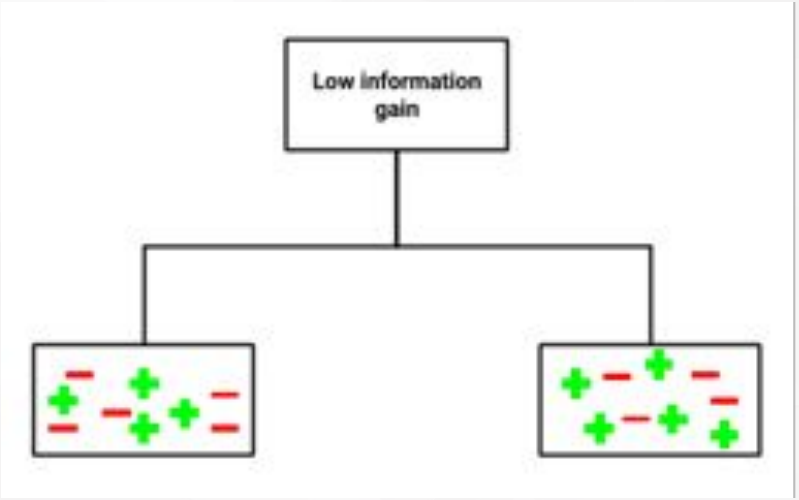
3. Splitting stops when it detects no further gain can be made, or some pre-set stopping rules are met.

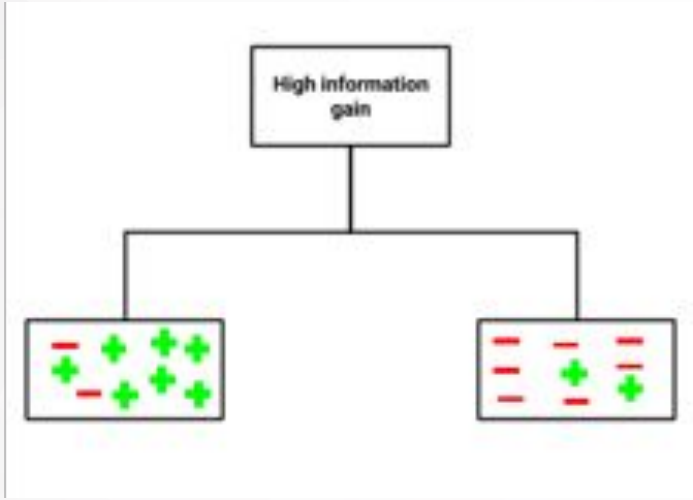
Attribute Selection: What is the attribute that best separates the given examples (split) ?

Two widely used Metrics:

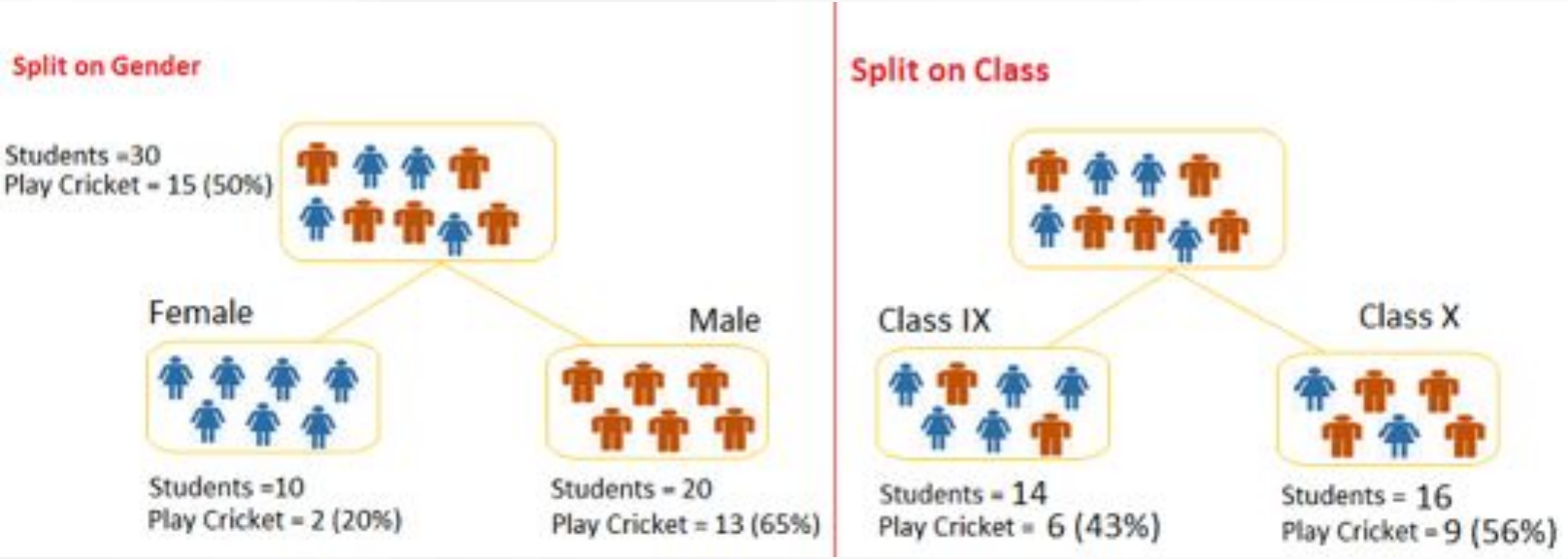
- Gini Index - Identifies which splits create more homogenous subnodes.
- Information Gain / Entropy - A measure to define this degree of disorganization in a system known as Entropy, low entropy means it is easier to describe a node

Split on Information Gain





Split on Gini



Decision Tree Parameters

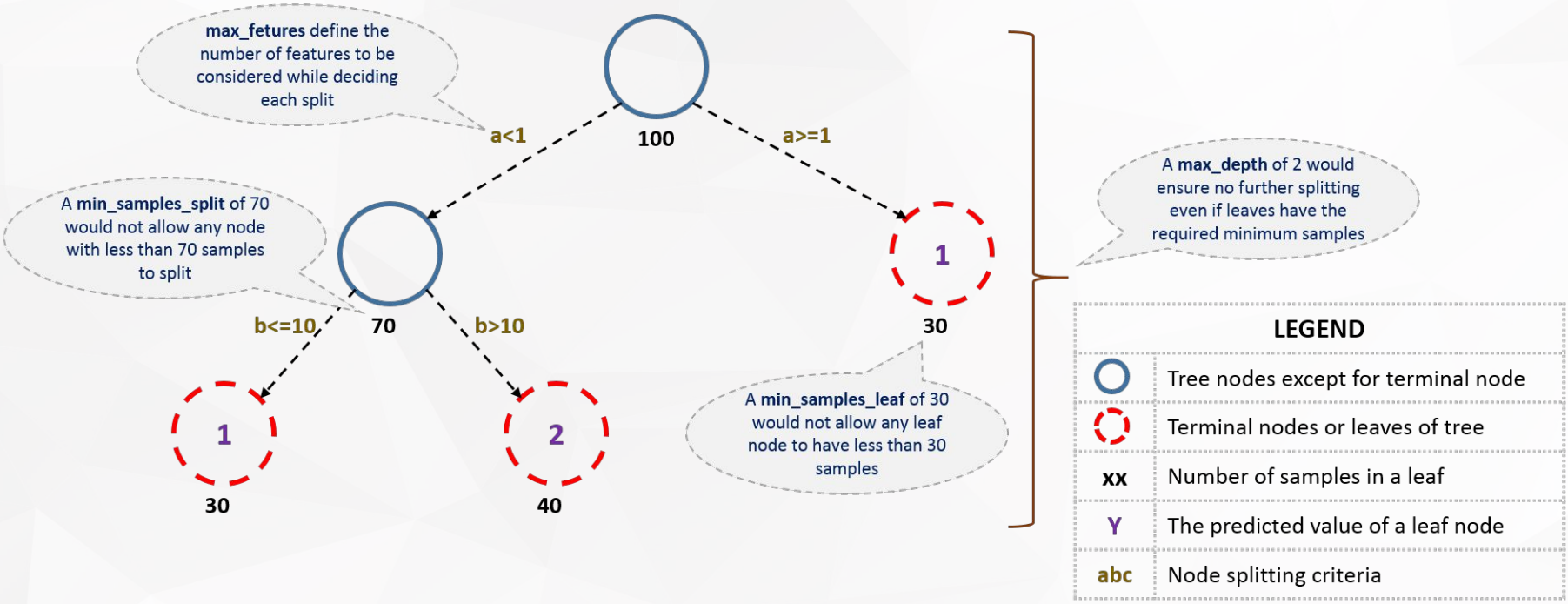
- max_features
  - maximum number of features Random Forest is allowed to try in individual tree
  - sqrt, 0.2, Auto/None
  - > max_features, slower but reduces the diversity of each tree

- min_samples_split
  - minimum number of samples (or observations) required in a node to be considered for splitting.
  - Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.

- min_sample_leaf
  - smaller leaf makes the model more prone to capturing noise in train data
  - Used to control over-fitting as higher depth will allow model to learn relations very specific to a particular sample.

- max_depth
  - The maximum depth of a tree
  - Limiting depth may help resolve overfitting

Decision Tree



Decision Tree: Pros and Cons

+	-
• Simple to understand and to interpret. Trees can be visualised. • Implicitly performs feature selection • Requires little data preparation. • Uses a white box model. • Does not require much computation for prediction. • Able to handle both continuous and categorical variables/features • Able to handle multi-output problems • Nonlinear relationships between parameters do not affect tree performance	• Overfitting • Can be unstable because small variations in the data might result in a completely different tree being generated • Doesn’t guarantee to return the globally optimal decision tree. • Decision tree learners create biased trees if some classes dominate.

LAB:
Decision Tree

Metrics

Classification

Metrics

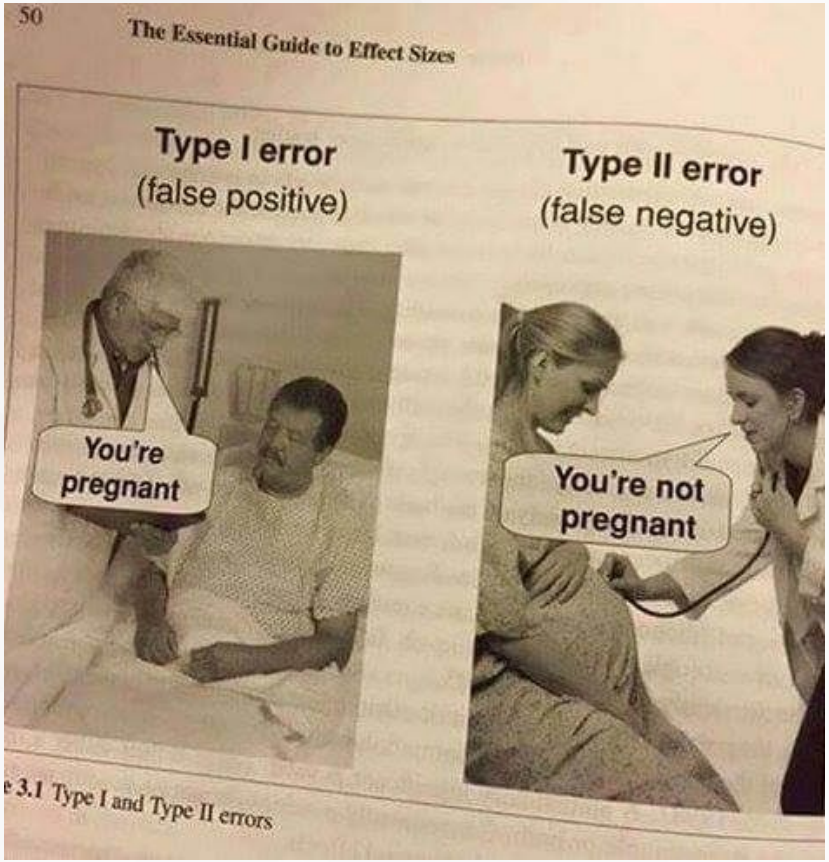
- Confusion Matrix
- Accuracy
- Error/Misclassification
- Precision
- Recall
- F1-score
- Advanced Metrics
    - K-S Statistic
    - AUC / ROC
    - Gini Coefficient

Metrics: Confusion Matrix

- A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.
- Basic terms in Confusion Matrix:
  - True Positives (TP): The prediction is “yes” and the actual label is “yes”
  - True Negatives (TN): The prediction is “no” and the actual label is “no”
  - False Positives (FP): The prediction is “yes” but the actual label is “no”
    - Also known as a Type I error.
  - False Negatives (FN): The prediction is “no” but the actual label is “yes:
    - Also known as a Type II error

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: Confusion Matrix



Metrics: Accuracy Rate

- Accuracy answers “How often is the classifier correct?”
- Formula:

$$\text{Accuracy} = \frac{(TP+TN)}{N}$$

  - Where
    - TP = True Positive
    - TN = True Negative
    - N = Total Number of samples

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: Error Rate

- Error or Misclassification Rate answers “How often is it wrong?”
- Formula:

$$\text{Error} = \frac{(FP+FN)}{N}$$

  - Where
    - FP = False Positive
    - FN = False Negative
    - N = Total Number of samples

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: Precision

- Precision of Sensitivity answers “When it's actually yes, how often does it predict yes?”
- Formula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

  - Where
    - TP = True Positive
    - FP = False Positive

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: Recall

- Recall answers “What is the proportion of actual YES that are correctly classified?”
- Formula:

$$\text{Recall} = \frac{TP}{TP + FN}$$

  - Where
    - TP = True Positive
    - FN = False Negative

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: F1 Score

- F1 Score Can be interpreted as the weighted average of Precision and Recall
- Formula:

$$\text{F1 Score} = 2 * \frac{P * R}{P + R}$$

  - Where
    - P = Precision
    - R = Recall

n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

Metrics: Example

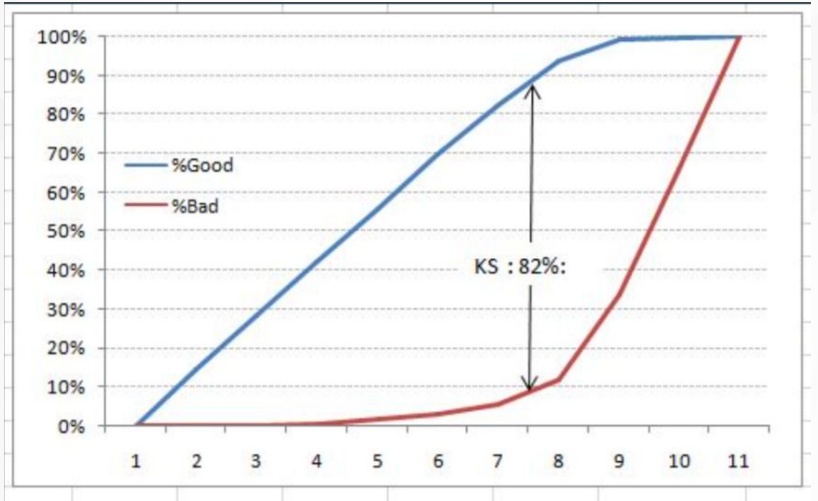
n=165	PREDICTED: NO	PREDICTED: YES	
ACTUAL: NO	TN = 50	FP = 10	60
ACTUAL: YES	FN = 5	TP = 100	105
	55	110	

ACCURACY	TP + TN / (TP + TN + FN + FP)	90.91
ERROR	FP + FN / (TP + TN + FN + FP)	9.09
PRECISION	TP / (TP + FP)	90.91
RECALL	TP / (TP + FN)	95.24
F1 SCORE	2 * ((PRECISION * RECALL) / (PRECISION + RECALL))	93.02

Advanced Metrics: K-S Statistic

- A measure of the degree of separation between the positive and negative distributions.
- Value will fall between 0 and 100, the higher the value the better the model is at separating the positive from negative cases.
    - K-S is 100, if the scores partition the population into two separate groups in which one group contains all the positives and the other all the negatives.
    - The K-S is 0 if the model cannot differentiate between positives and negatives, as if the model selects cases randomly from the population

Video



X axis = credit score values

Y axis = cumulative proportions of observations in each outcome class (Good Credit vs. Bad Credit) in the hold-out sample.

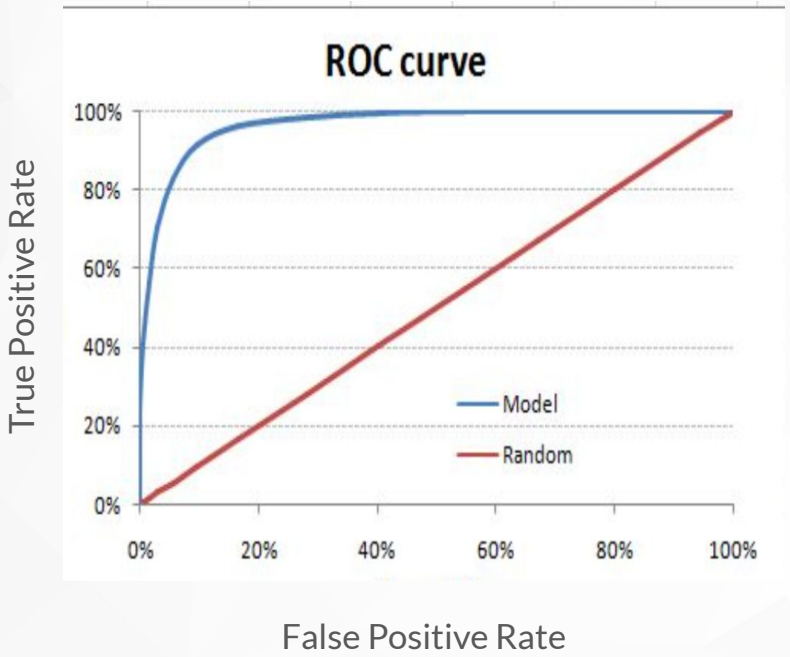
Advanced Metrics: K-S Statistic

- Measures the distance between the plotted cumulative distributions of the two classes (i.e. positive and negative)
- Each classification score can be transformed to lie between 0 and 1
- The score that generates the greatest vertical separability between the functions is considered the threshold value for accepting/rejecting a positive class (i.e. credit application)
- The model producing the greatest amount of separability between the two distributions would be considered the superior model

Score Range		Count		Raw %		Cumulative %		
Lower	Upper	+	-	% +	% -	% +	% -	K-S
0	10	0	543	0%	14%	0%	14%	14%
10	20	2	542	0%	14%	0%	28%	28%
20	30	7	537	0%	14%	1%	42%	42%
30	40	15	529	1%	14%	2%	56%	54%
40	50	20	524	1%	14%	3%	69%	67%
50	60	42	502	3%	13%	5%	83%	77%
60	70	104	440	7%	11%	12%	94%	82%
70	80	345	199	22%	5%	34%	99%	65%
80	90	515	29	32%	1%	66%	100%	34%
90	100	540	5	34%	0%	100%	100%	0%
		1590	3850					

Area Under the ROC curve (AUC – ROC)

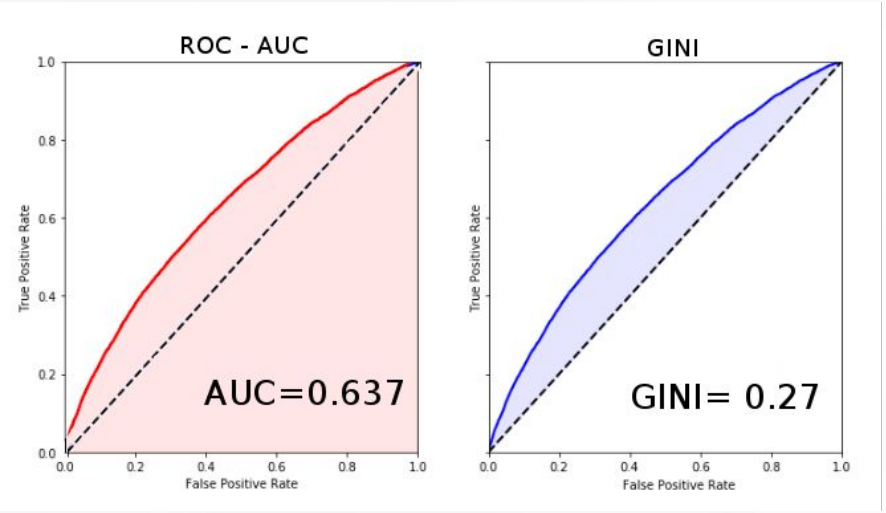
- The ROC curve is the plot between the True Positive Rate (Sensitivity) and the False Positive Rate (1-Specificity) at various threshold settings.
- Diagonal Line: Correctly Classified proportionate to Incorrectly Classified
- True Positive Rate (Sensitivity) = $\frac{TP}{TP + FP}$
- False Positive Rate = $\frac{FP}{FP + TN}$
- Area under the Curve (AUC)
  - AUC = 1.0 - the perfect classifier (Square)
  - AUC = .5 - random guessing (diagonal)



Interactive Demo: http://www.navan.name/roc/

Gini Coefficient

- Gini is the ratio between area between the ROC curve and the diagonal line & the area of the above triangle
- A scale of predictive power from 0 to 1. A Gini of 0 is the statistical equivalent of a coin toss/ A Gini of 1 is perfectly predictive



- Used to evaluate the predictive power of credit scoring tools.

Advanced Metrics: K-S Statistic

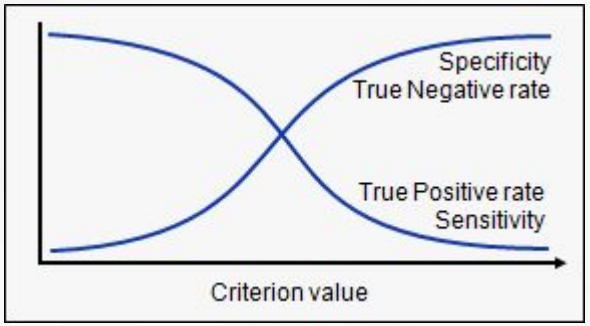
- Also used when making decisions on probability thresholds- to decide which observations to act on (i.e. which customers to approve, which customers to target for a marketing campaign)
- The significance of KS statistic is, it helps to understand, what portion of the population should be targeted to get the highest response rate (1's)
- How to compute for the KS Statistic?

						Cummulative			
Lift/Gain	Colum			%Rights	%Wrongs	Cum %Rig	Cum %Wrong	K-S	
Row La	0	1	Grand Tot	0%	0%	0%	0%	0%	
1		543	543	14%	0%	14%	0%	14%	
2	2	542	544	14%	0%	28%	0%	28%	
3	7	537	544	14%	0%	42%	1%	42%	
4	15	529	544	14%	1%	56%	2%	54%	
5	20	524	544	14%	1%	69%	3%	67%	
6	42	502	544	13%	3%	83%	5%	77%	
7	104	440	544	11%	7%	94%	12%	82%	K-S
8	345	199	544	5%	22%	99%	34%	65%	
9	515	29	544	1%	32%	100%	66%	34%	
10	540	5	545	0%	34%	100%	100%	0%	
Grand Tot	1590	3850	5440						

Area Under the ROC curve (AUC – ROC)

- Often, best model is a balance between predicting the one's accurately (TPR) or the zeroes (TNR) accurately. In other words sensitivity and specificity.

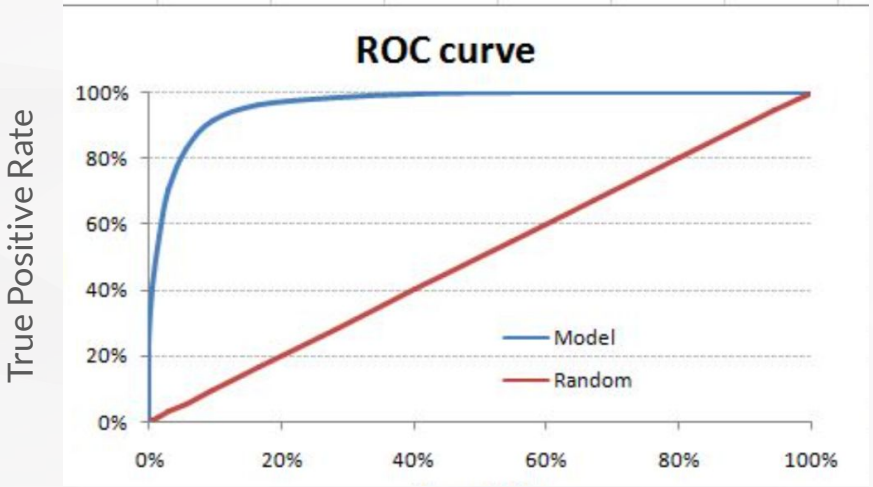
Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		



For each sensitivity, we get a different specificity

Area Under the ROC curve (AUC – ROC)

- The ROC curve is the plot between the True Positive Rate (Sensitivity) and the False Positive Rate (1-Specificity) at various threshold settings.
- Example: For threshold = .5



Count of ID	Target			
Model	1	0	Grand Total	
1	3,834	639	4,473	85.7%
0	16	951	967	1.7%
Grand Total	3,850	1,590	5,440	
	99.6%	40.19%		88.0%

AUC = 1.0 - the perfect classifier (Square)
AUC = .5 - random guessing (diagonal)

Exercise: Classification

analytiks

HW: Classification

analytiks