TU 257 – Fundamentals of Data Science

Data Analytics

L4 – Classification – Part 2

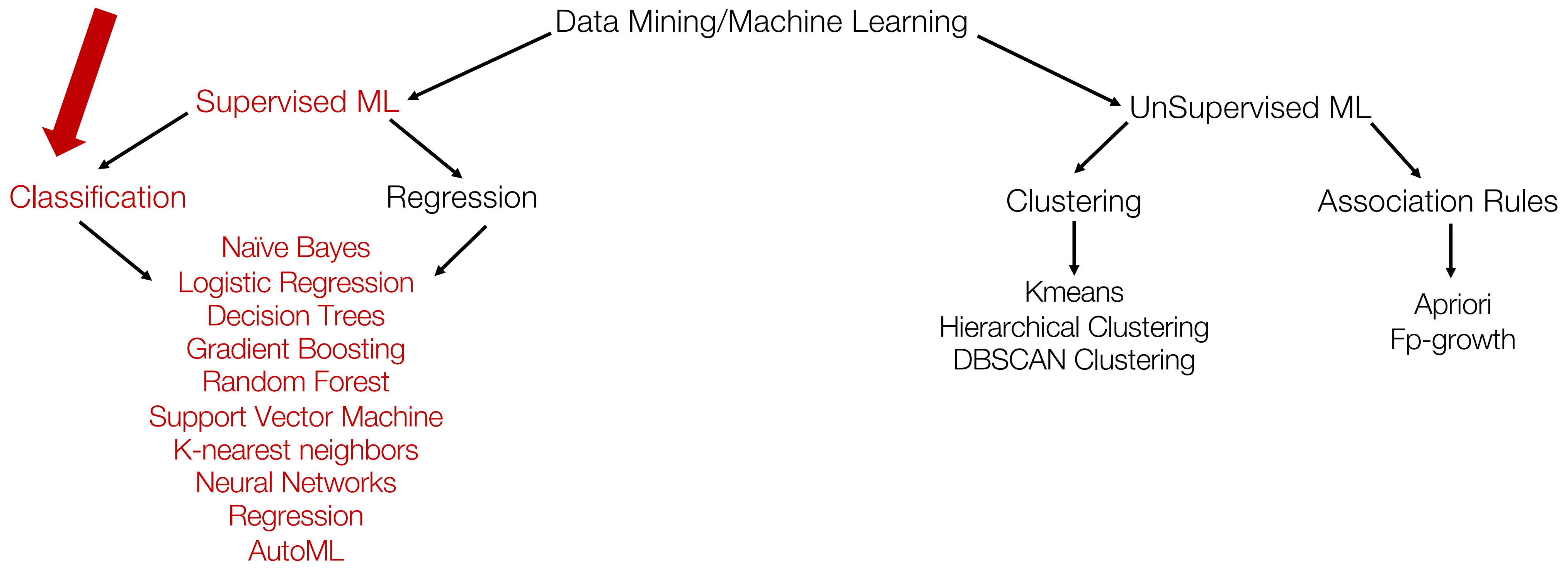
Brendan Tierney

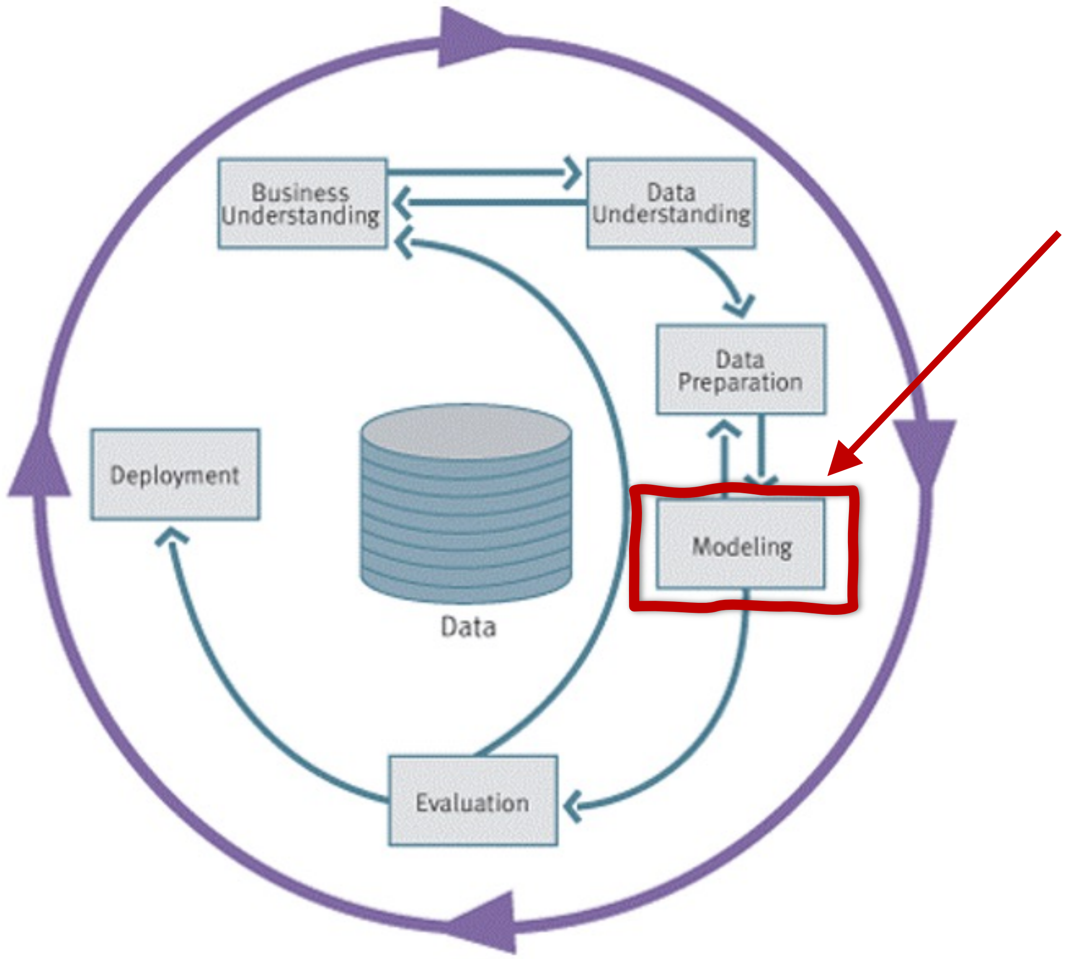
Agenda

- Recap of last week
- Lots of Algorithms
    - More this week
    - Not all possible algorithms will be covered
- Some details/background/under-the-hood at the algorithms
    - Inner details are not needed. Can be explored in a Machine Learning module
- Recap - How do you measure if it's any good
    - Some additional ways to test and measure

Machine Learning

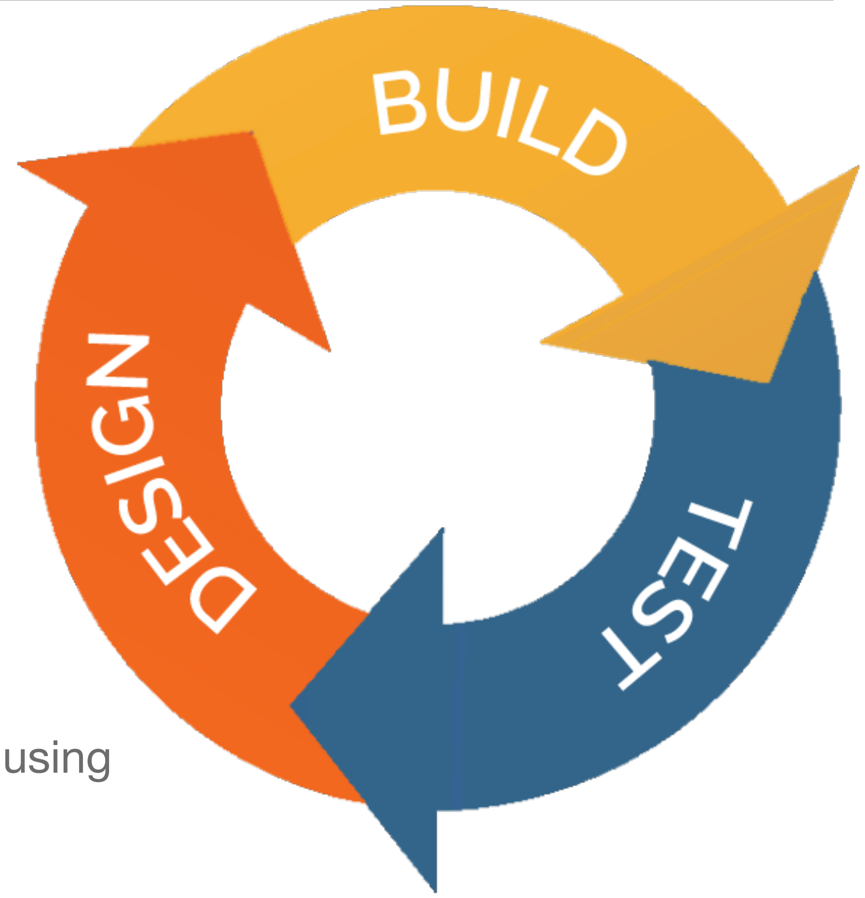
- the use and development of computer systems that are able to learn and adapt without following explicit instructions, by using algorithms and statistical models to analyse and draw inferences from patterns in data.




Business Understanding
Data Understanding
Data Preparation
Deployment
Data
Modeling
Evaluation

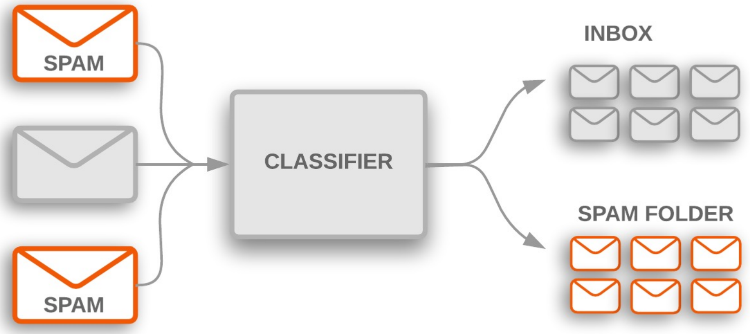
Iteratively Building Your Knowledge

- You are building your knowledge each week
- This week builds upon last week
- Next week builds upon this week
- You build a wall block-by-block
- This is not a programming module – although we are using code to illustrate how to do it



What is Classification

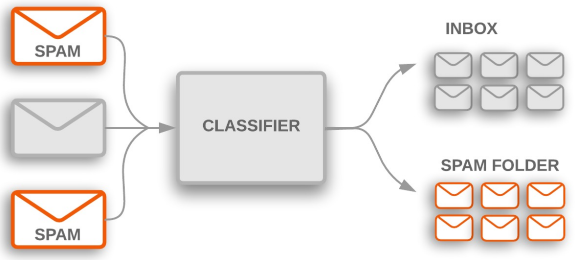
- Classification is a task that requires the use of (machine learning) algorithms that learn how to assign a class label to examples from the problem domain.
  - We learn from the past to predict the future
  - An easy to understand example is classifying emails as “*spam*” or “*not spam*.”



- Classification predictive modeling involves
  - Looking at historical data representing a particular scenario
  - Using algorithms to find patterns in the data
    - What attributes/features contribute towards determining the scenario being investigated
    - Assigning a class label.

Classification – Different types

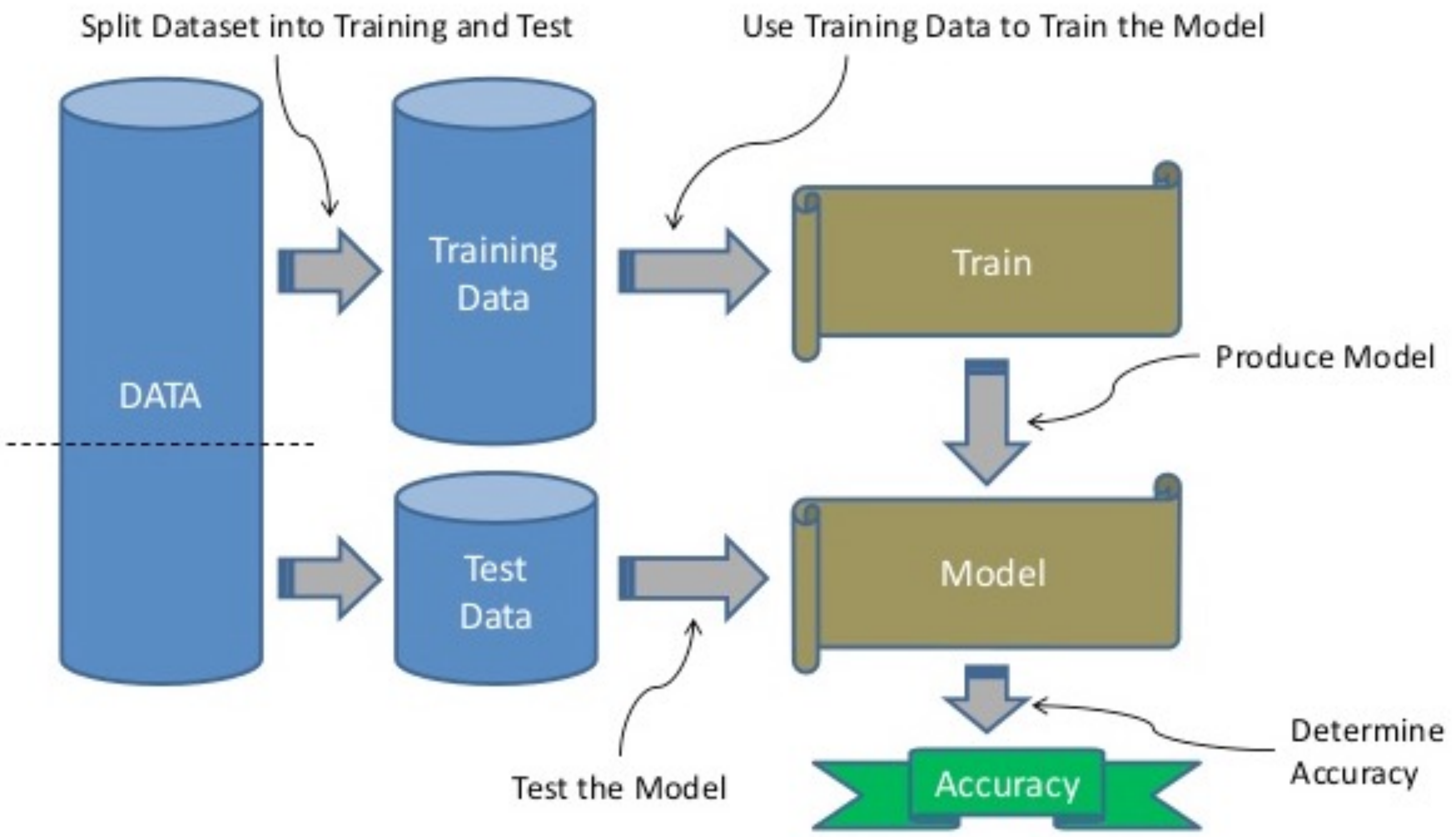
- Binary classification refers to predicting one of two classes
  - Yes / No
  - 0 / 1
  - Buy / Not-Buy
  - Spam / Not-Spam

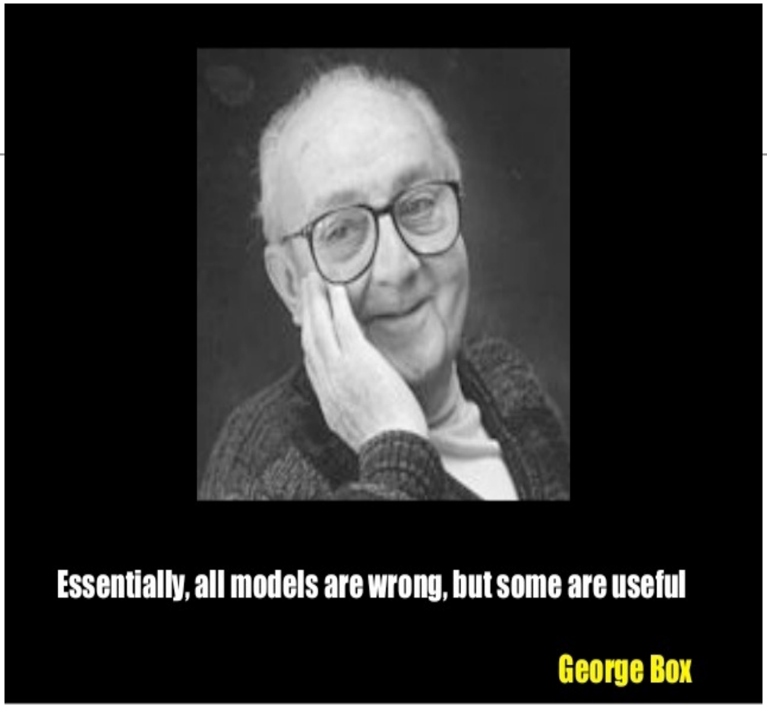


- Multi-class Classification is when we have more than 2 class values
  - Different Fruits
  - Credit Ratings
  - Different Products

PAPER
METAL
PLASTIC
E-WASTE
GLASS
ORGANIC

Data Set = Training + Test sata sets





A model is a simplification or approximation of reality and hence will not reflect all of reality.

His paper was published in the Journal of the American Statistical Association, 1976
Book *Empirical Model-Building and Response Surfaces*, 1987

The Algorithms

- Last week
  - Naive Bayes
  - Decision Trees
  - Random Forests
  - XGBoost
- This week
  - Knn
  - Support Vector Machines
  - Neural Networks (basics)

Naive Bayes

Decision Trees

Random Forests

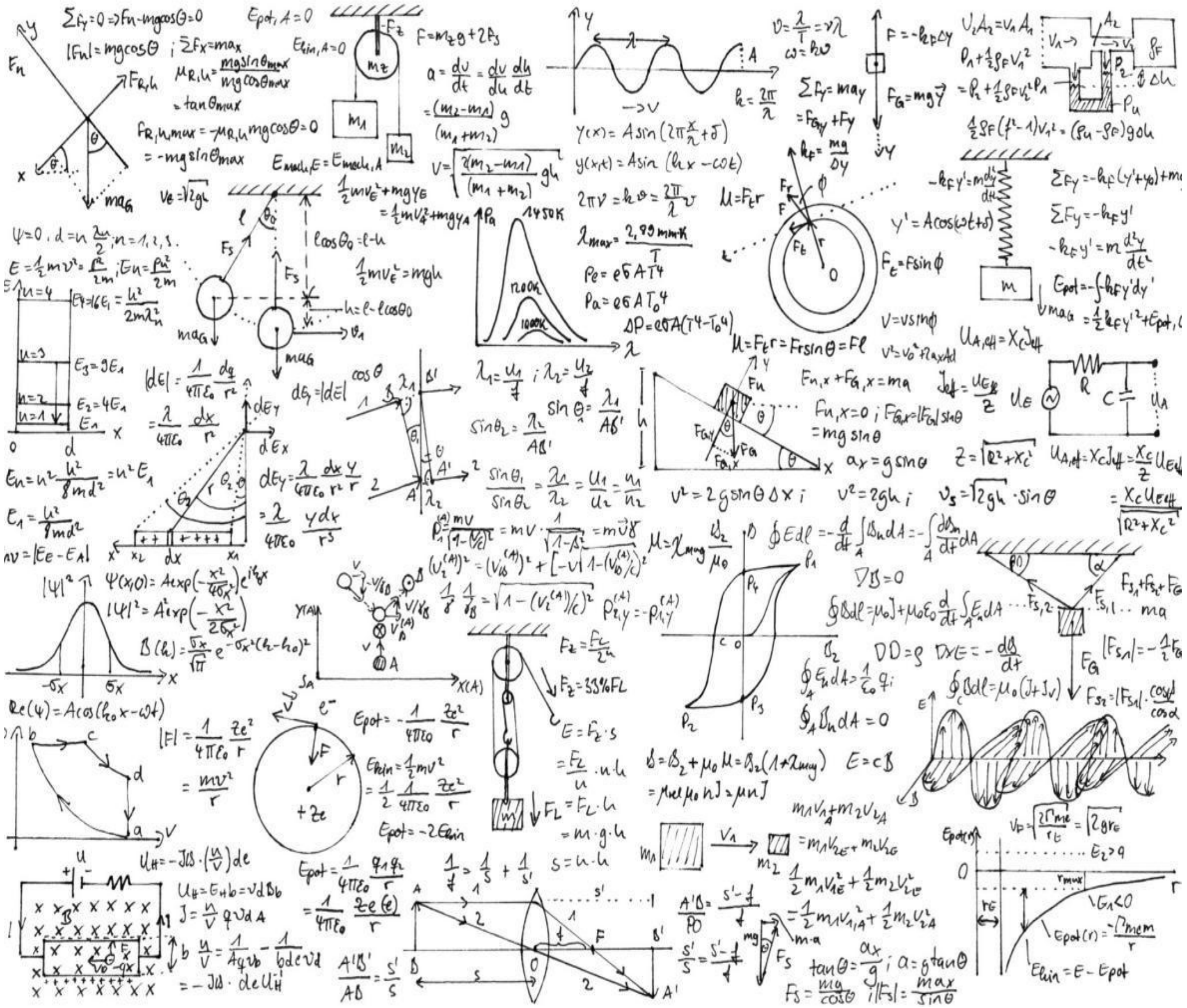
XGBoost

The Algorithms

KNN

SVM

Neural Networks



Nearest Neighbour Example

Wave Size (ft)	Wave Period (secs)	Good Surf?
6	15	Yes
1	6	No
5	11	Yes
7	10	Yes
6	11	Yes
2	1	No
3	4	No
6	12	Yes
4	2	No

Query

10	10	?

Nearest Neighbour Example

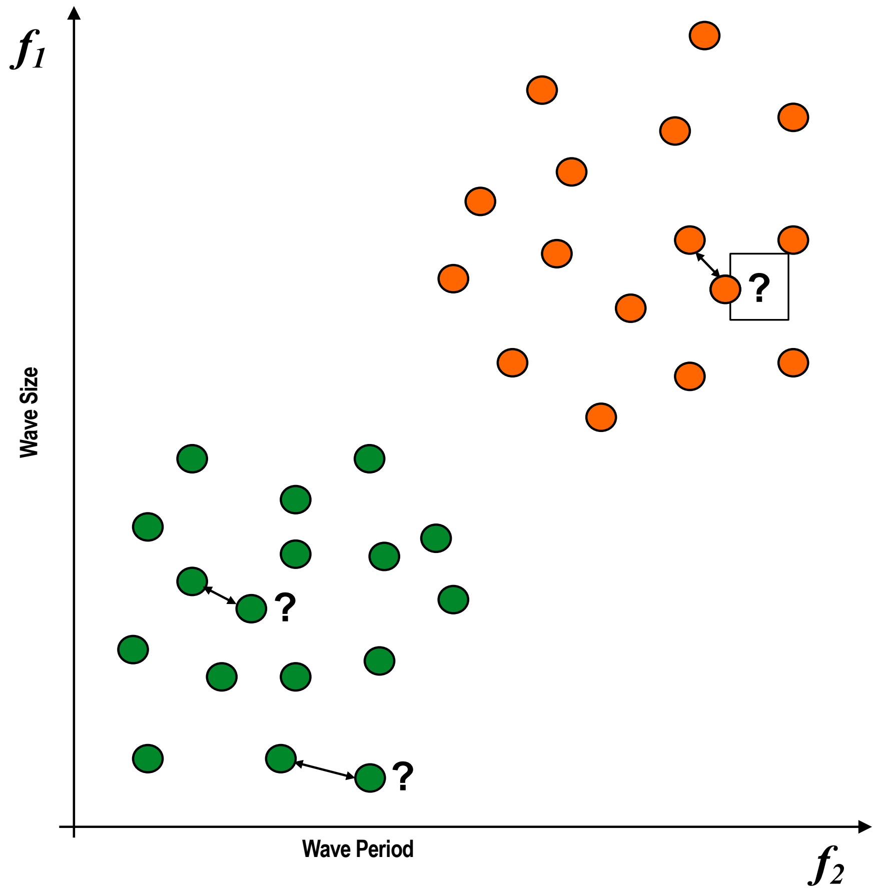
Wave Size (ft)	Wave Period (secs)	Good Surf?
6	15	Yes
1	6	No
5	11	Yes
7	10	Yes
6	11	Yes
2	1	No
3	4	No
6	12	Yes
4	2	No
6	15	Yes
1	6	No
5	11	Yes
7	10	Yes
6	11	Yes
2	1	No
3	4	No
6	12	Yes
4	2	No

Get gets more difficult as the number of cases increase

Knn

- Knn = K nearest neighbors
- Similarity based learning
- How similar is a record, compared to others in the data set
- Based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points. By choosing *K*, the user can select the number of nearby observations to use in the algorithm.
- Can be used in missing value imputation

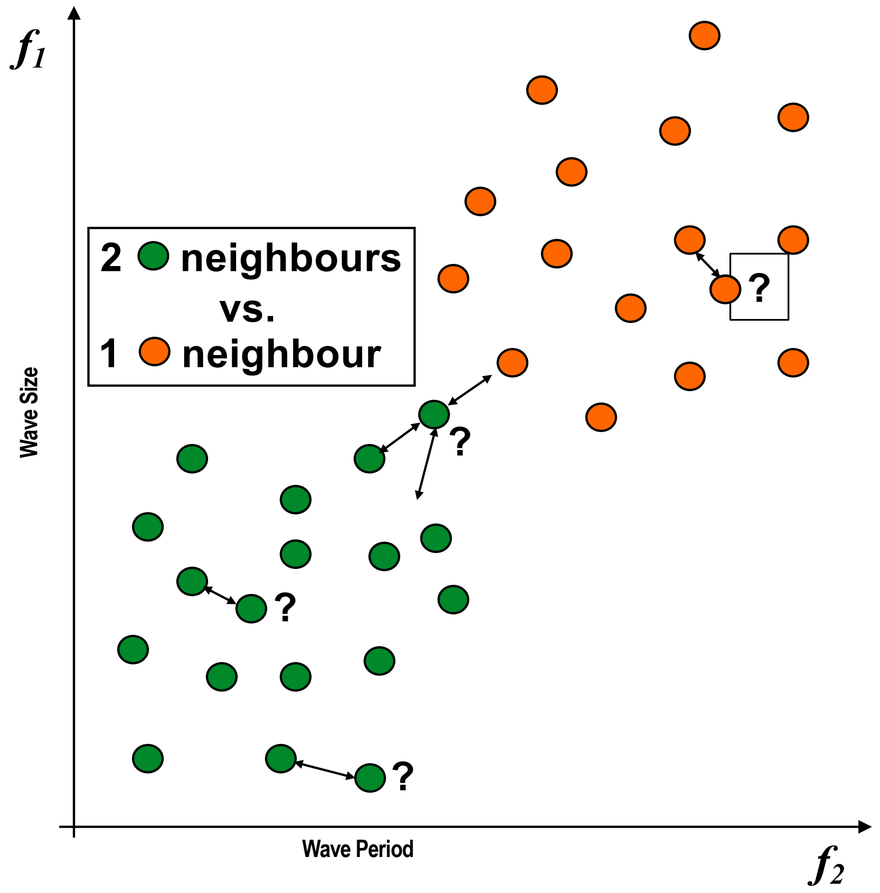
Nearest Neighbour Example



When a new case is to be classified:

- Calculate the distance from the new case to all training cases
- Put the new case in the same class as its nearest neighbour

Nearest Neighbour Example



When a new case is to be classified:

- Calculate the distance from the new case to all training cases
- Put the new case in the same class as its nearest neighbour

What about when it’s too close to call?

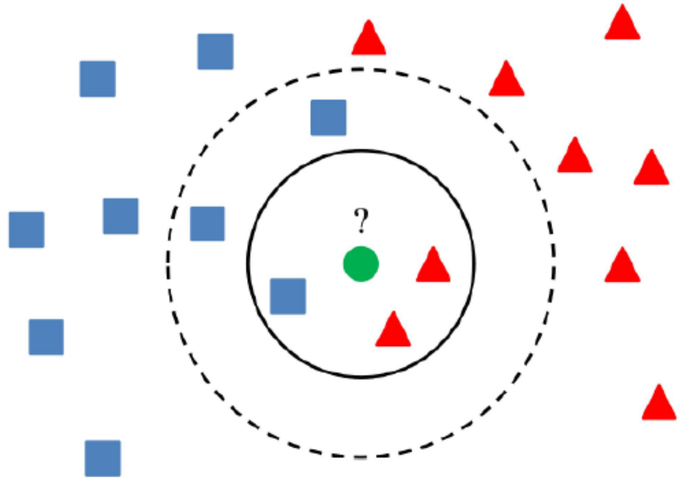
Use the *k*-nearest neighbour technique

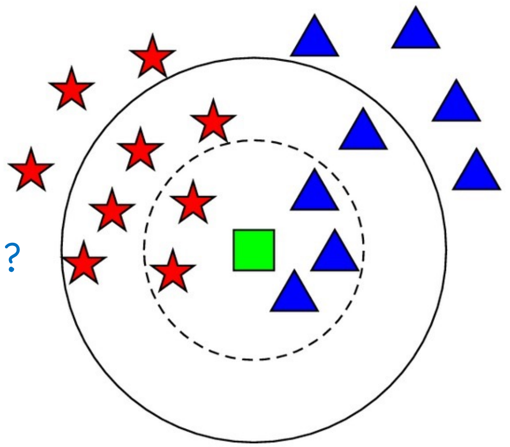
- Determine the *k* nearest neighbours to the query case
- Put the new case into the same class as the majority of its nearest neighbours

How many K value to evaluate

- K = small => potential errors
- K= large => lots of calculations => time
- (sometimes system determined)
- K = sqrt(N)/2,
  - where N stands for the number of samples in your training dataset
  - IF k is an even number THEN + 1
    - 5, 7 , 9, etc
  - To avoid any ties

K=3

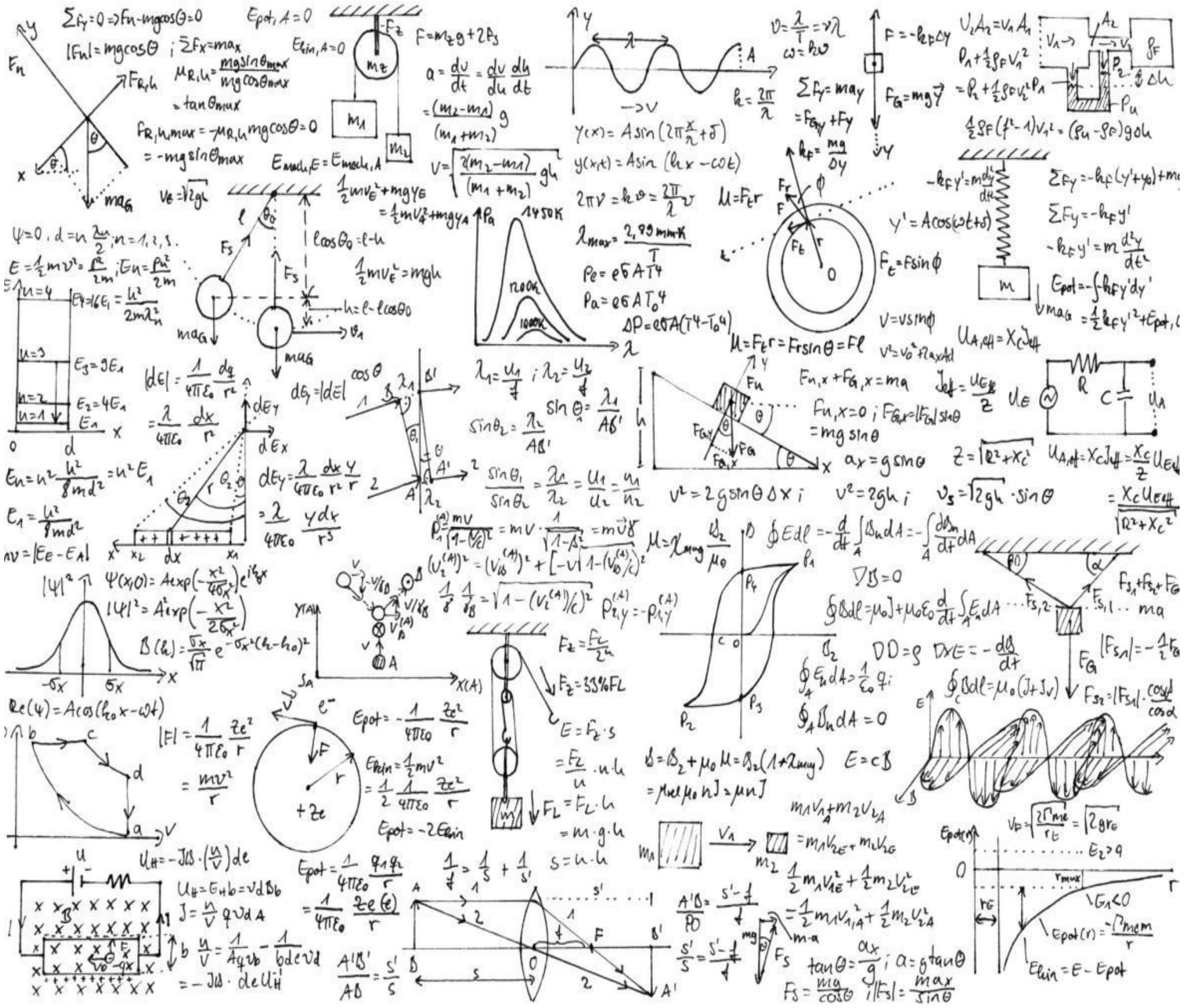




Which would be better?
K=3
or
K=5

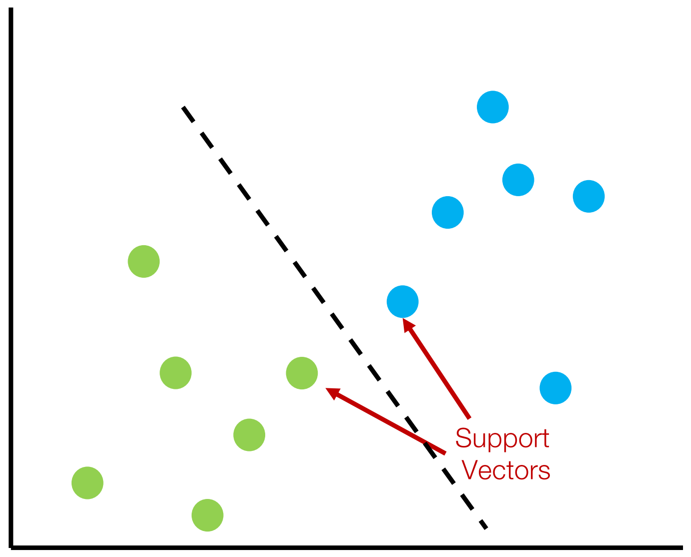
The Algorithms

- KNN
- SVM
- Neural Networks



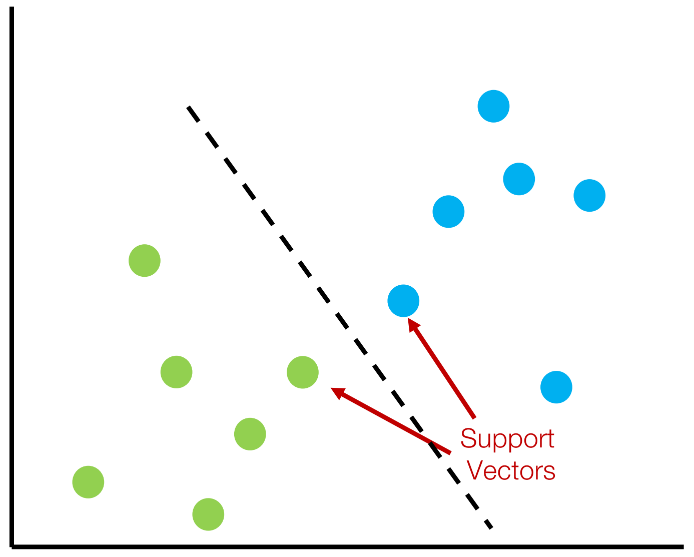
Support Vector Machines (SVMs)

- A Support Vector Machine (SVM) is a supervised machine learning algorithm that can be employed for both classification and regression purposes.
- SVMs are more commonly used in classification problems.
- SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes



Support Vector Machines (SVMs)

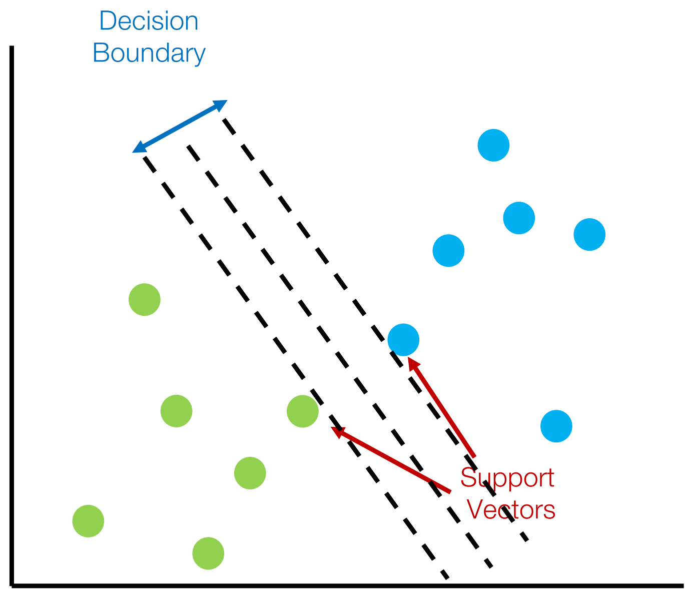
- Support Vectors
  - Support vectors are the data points nearest to the hyperplane, the points of a data set that, if removed, would alter the position of the dividing hyperplane. Because of this, they can be considered the critical elements of a data set.
- Hyperplane
  - Is a line that linearly separates and classifies a set of data.
  - The further from the hyperplane our data points lie, the more confident we are that they have been correctly classified. We therefore want our data points to be as far away from the hyperplane as possible, while still being on the correct side of it.
  - So when new testing data is added, whatever side of the hyperplane it lands will decide the class that we assign to it.



Support Vector Machines (SVMs)

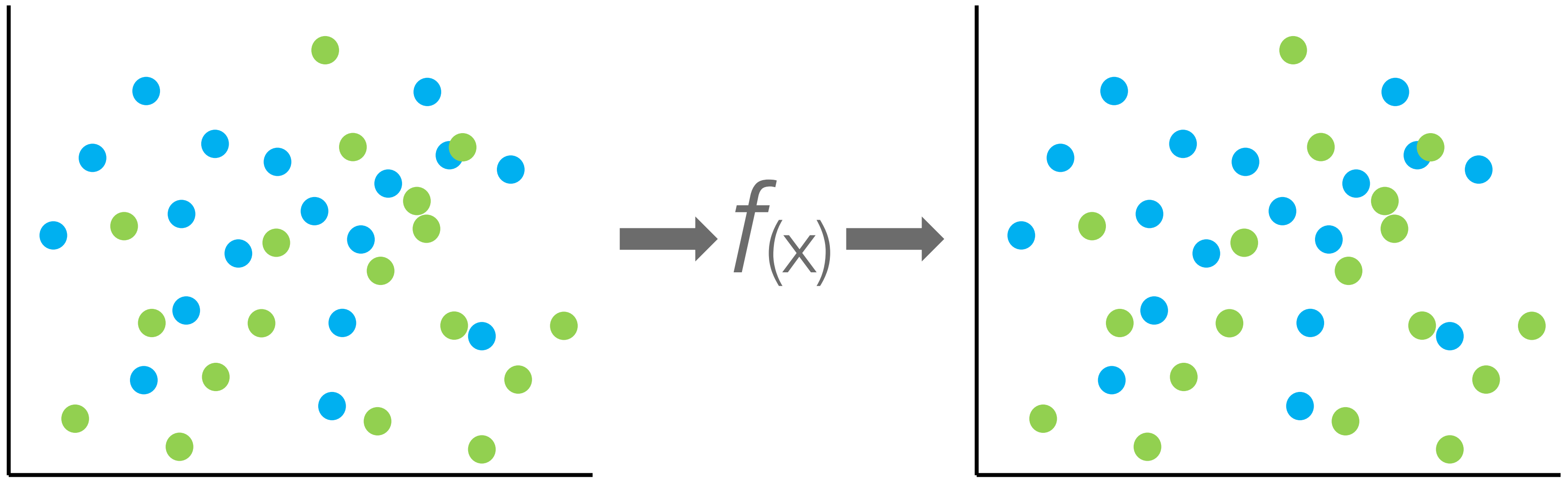
- Decision Boundary
    - Is the gap between the data points representing the different classes
    - The bigger or wider the gap the better
    - Allows for classification of data not seen in training data

    - Applies internal linear functions to separate data data for each class
    - Searches for function which gives the widest/biggest Decisions Boundary



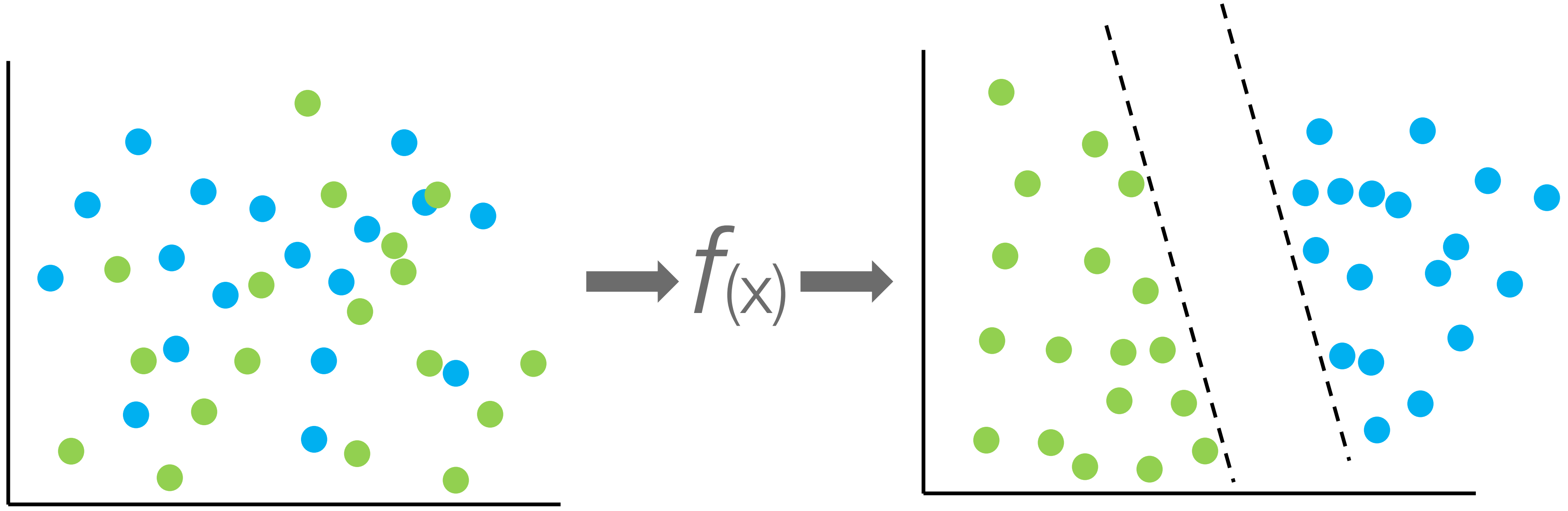
Support Vector Machines (SVMs)

- Applies internal function f(x)
  - Maps data into dimensional space
  - Iterates to find function f(x) with biggest Decision Boundary
  - This is difficult to visualise



Support Vector Machines (SVMs)

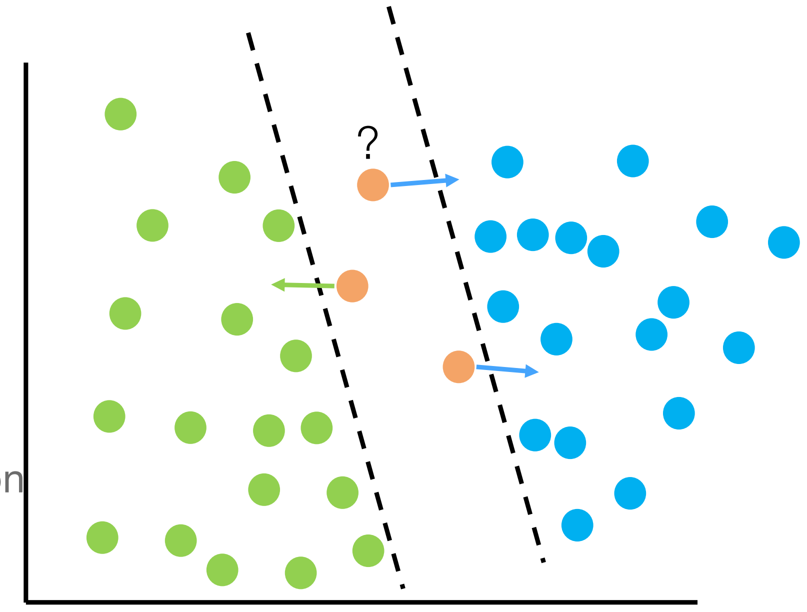
- Applies internal function f(x)
  - Maps data into dimensional space
  - Iterates to find function f(x) with biggest Decision Boundary
  - This is difficult to visualise



Support Vector Machines (SVMs)

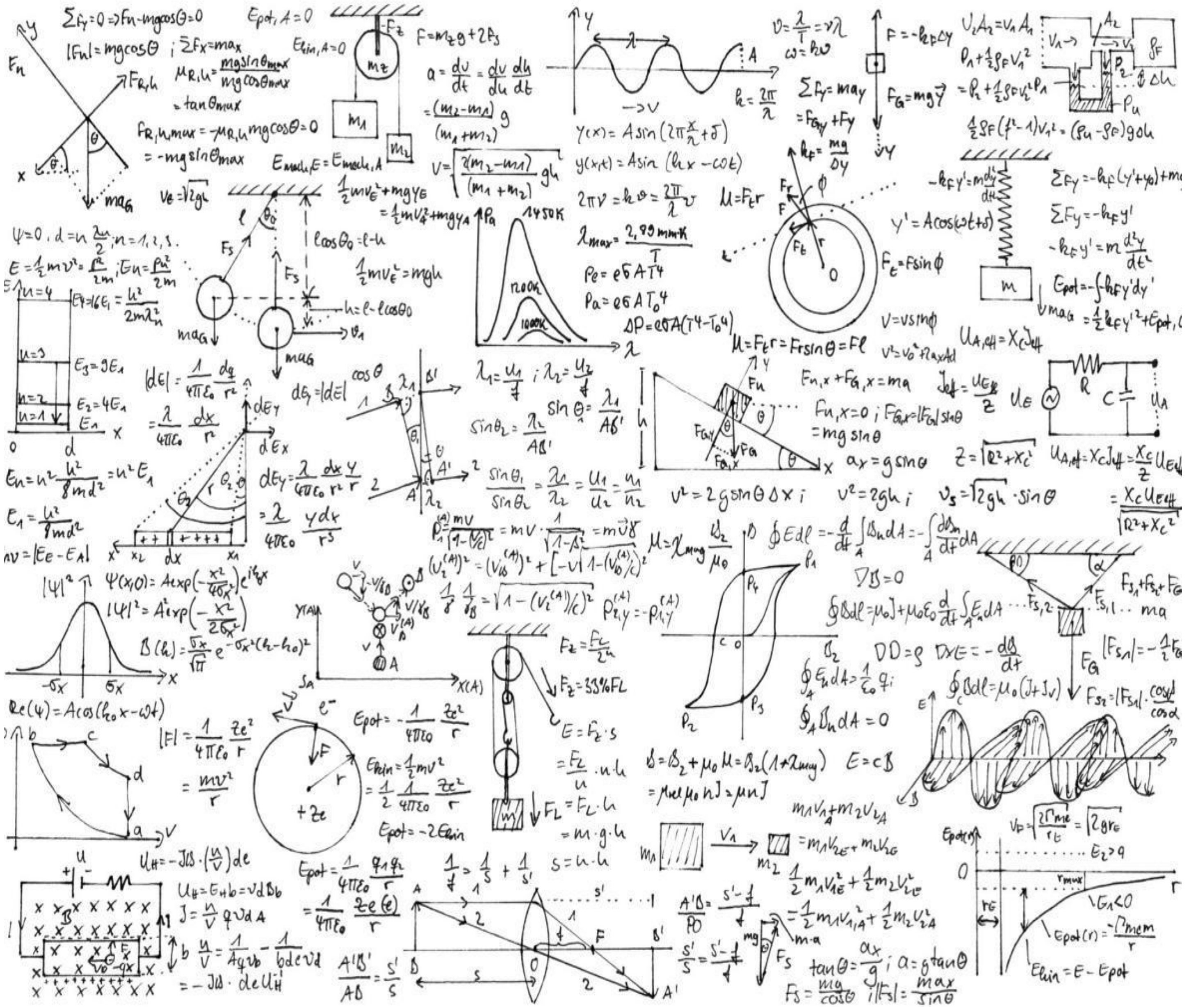
- Labelling/Classifying new data
- Good accuracy can be achieved
- Because of f(x)
- Even if data point is in middle
- Good for Unseen data

- Longer Computation time to build model
    - Understandable given f(x) and trying to create larger decision boundaries.
- Memory constrained
    - Limitation on Size of dataset
    - Up-to 250-300K records
- Very fast for labelling/Classifying new data
    - It is just applying a formula f(x)



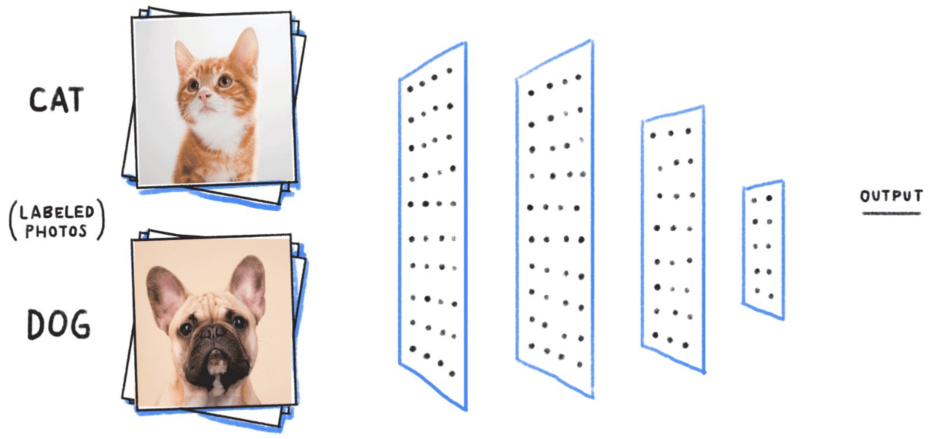
The Algorithms

- KNN
- SVM
- Neural Networks



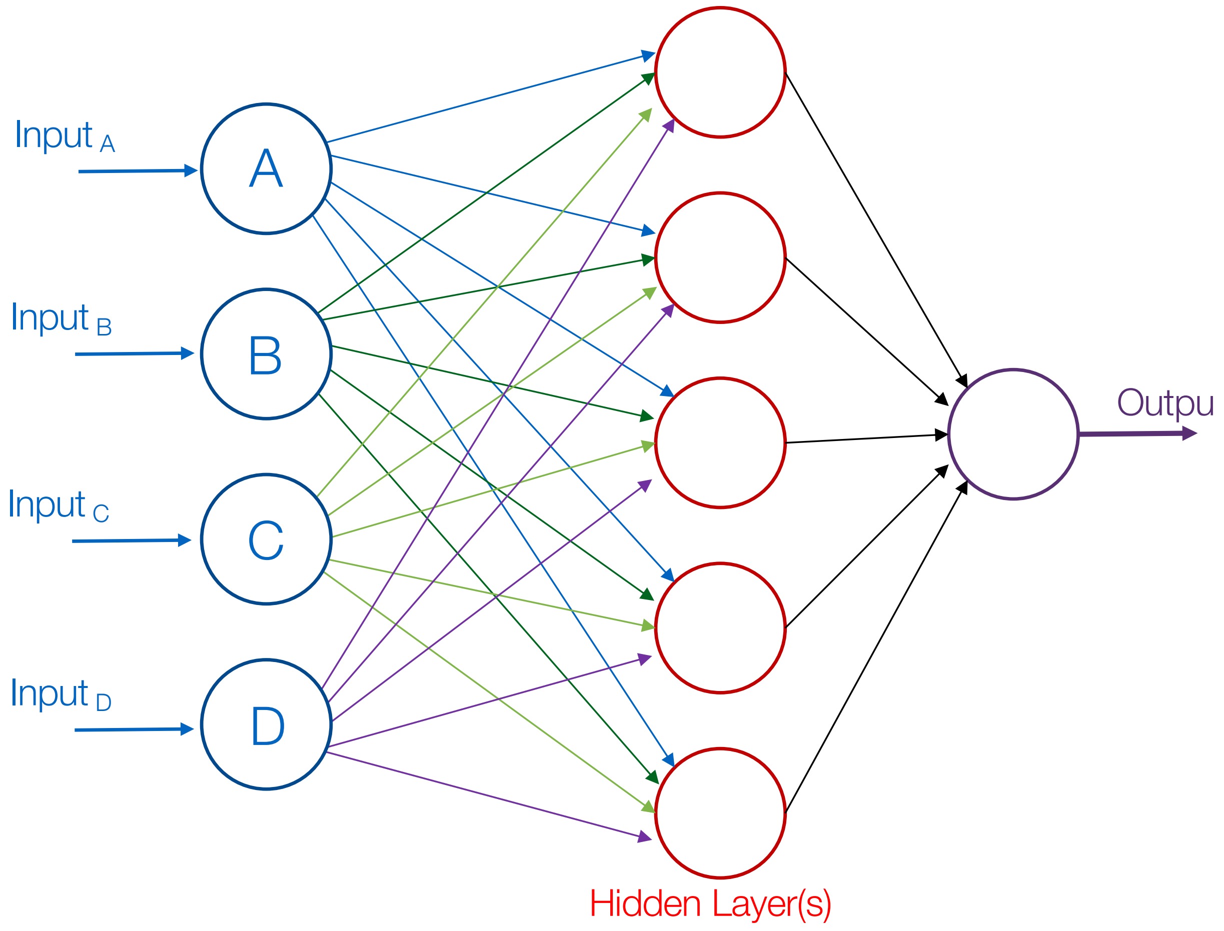
Neural Networks

- Neural networks, also known as artificial neural networks (ANNs), are a subset of machine learning and are at the heart of deep learning algorithms.
- Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.
- Suitable for a large number of problem areas
- Typically associated with Deep Learning
    - Image processing
    - Large Language Models (LMM)
- But
    - Can be applied to traditional data sets
    - Very slow at building models. Too much complexity
    - Difficulty real time ?

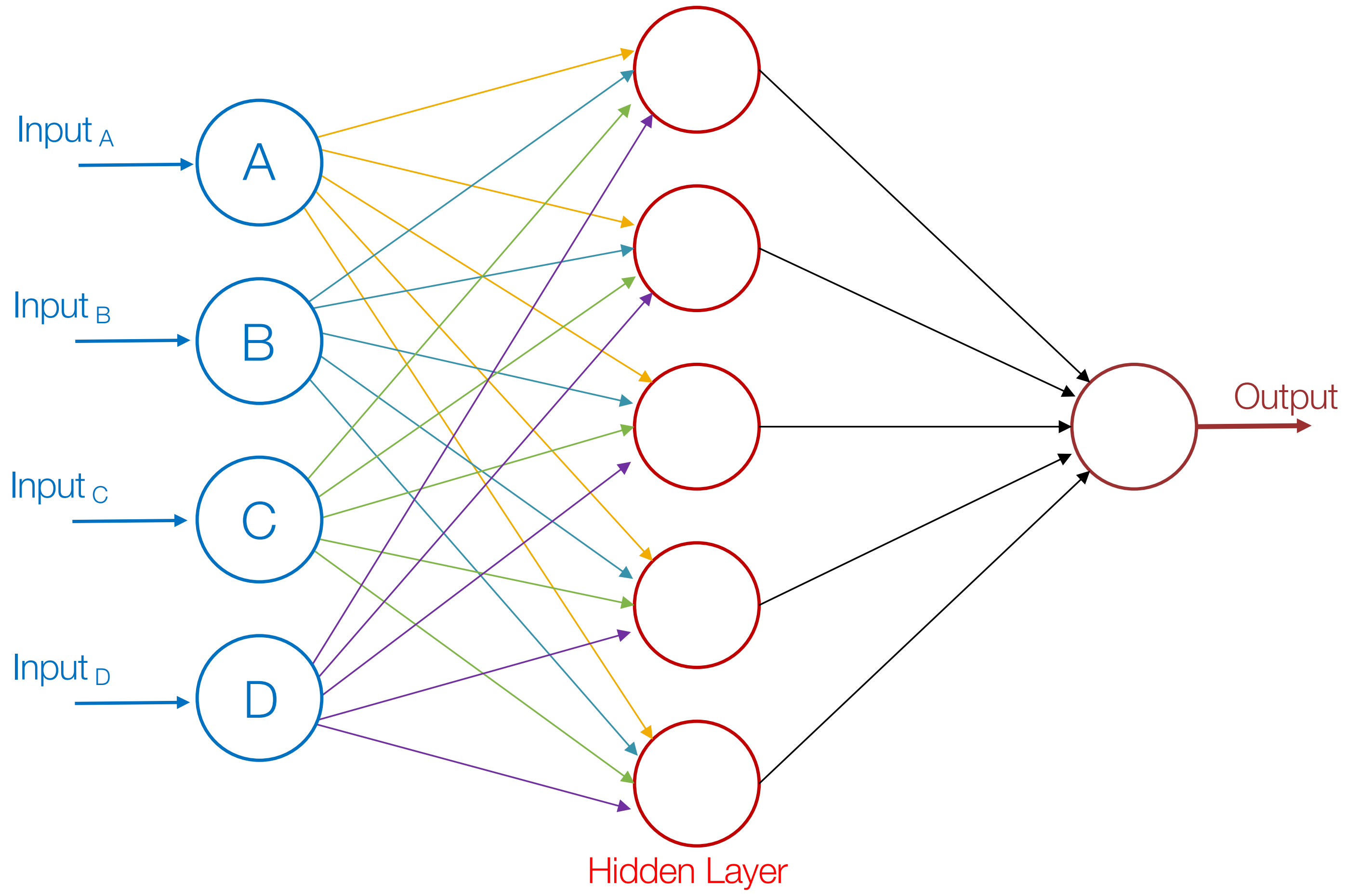


Neural Networks

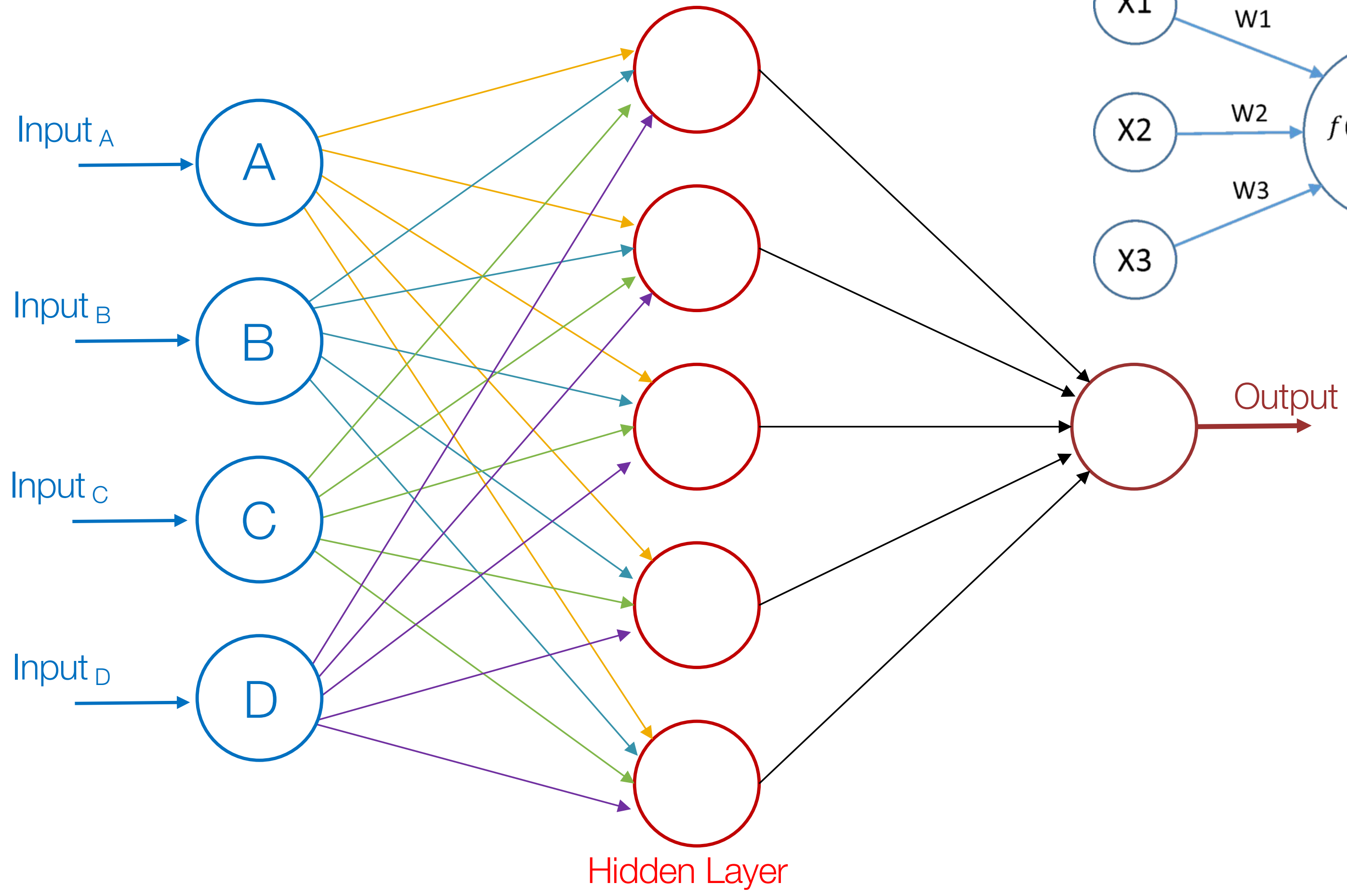
- Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer.
- Each node, or artificial neuron, connects to another and has an associated weight and threshold.
- If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

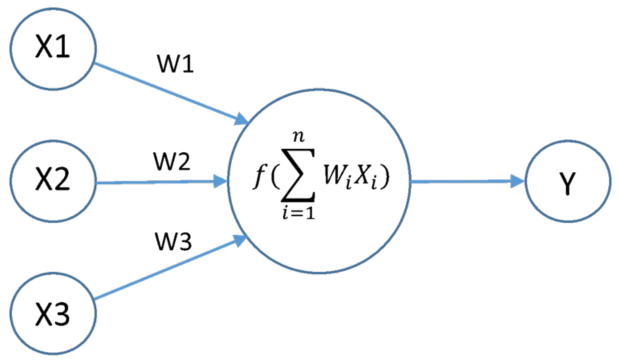


- All the inputs are connected each neuron in the hidden layer (red circles).
- Weights are applied to each input to a neuron
- A neuron takes a set of numeric values as input and maps them to a single output value.
- A neuron is a simple multi-input linear regression function, where the output is passed through an activation function.
    - Two common activations functions are logistic and tanh functions. There are many others including logistic sigmoid function, arctan function, bipolar sigmoid function, etc.

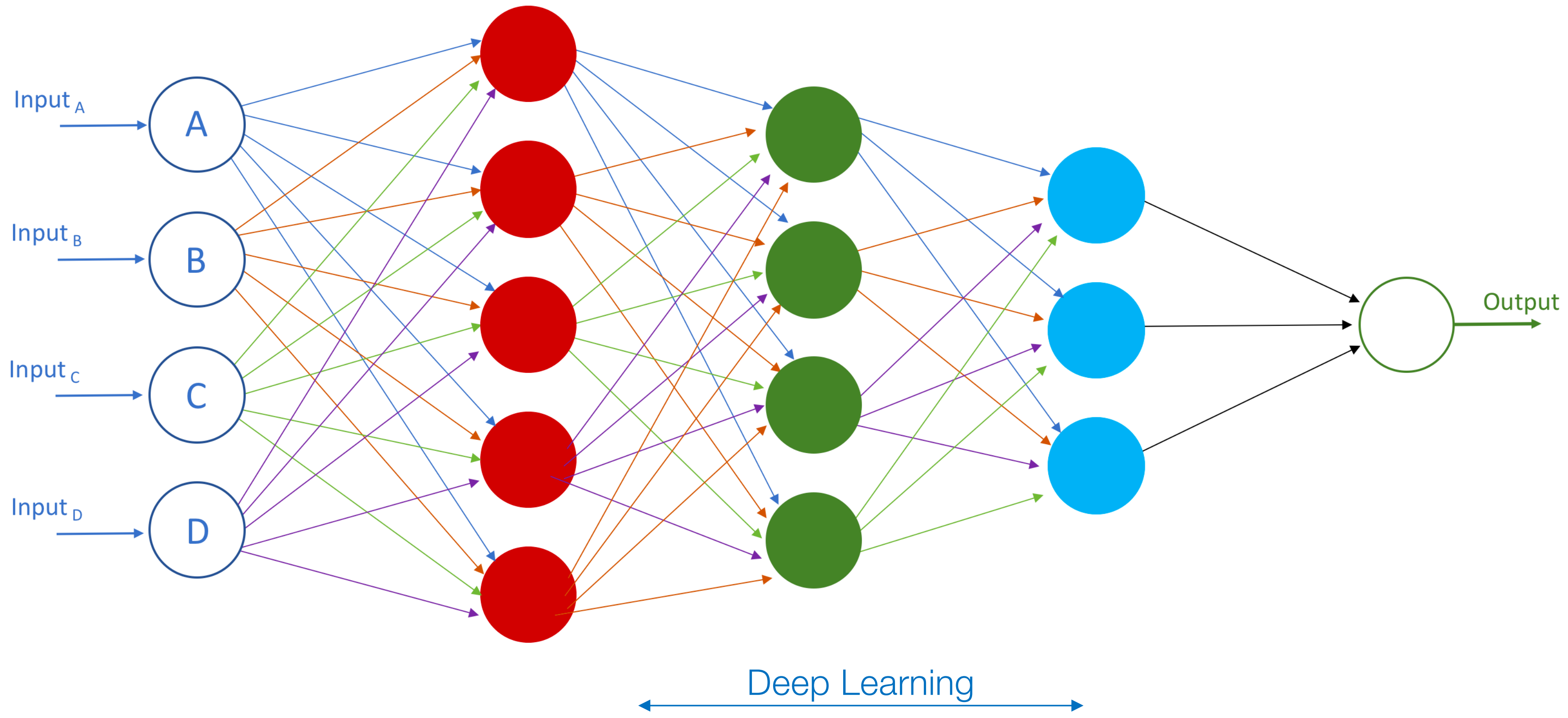


- All the inputs are connected each neuron in the hidden layer (red circles).
- Weights are applied to each input to a neuron
- A neuron takes a set of numeric values as input and maps them to a single output value.
- A neuron is a simple multi-input linear regression function, where the output is passed through an activation function.
    - Two common activations functions are logistic and tanh functions. There are many others including logistic sigmoid function, arctan function, bipolar sigmoid function, etc.

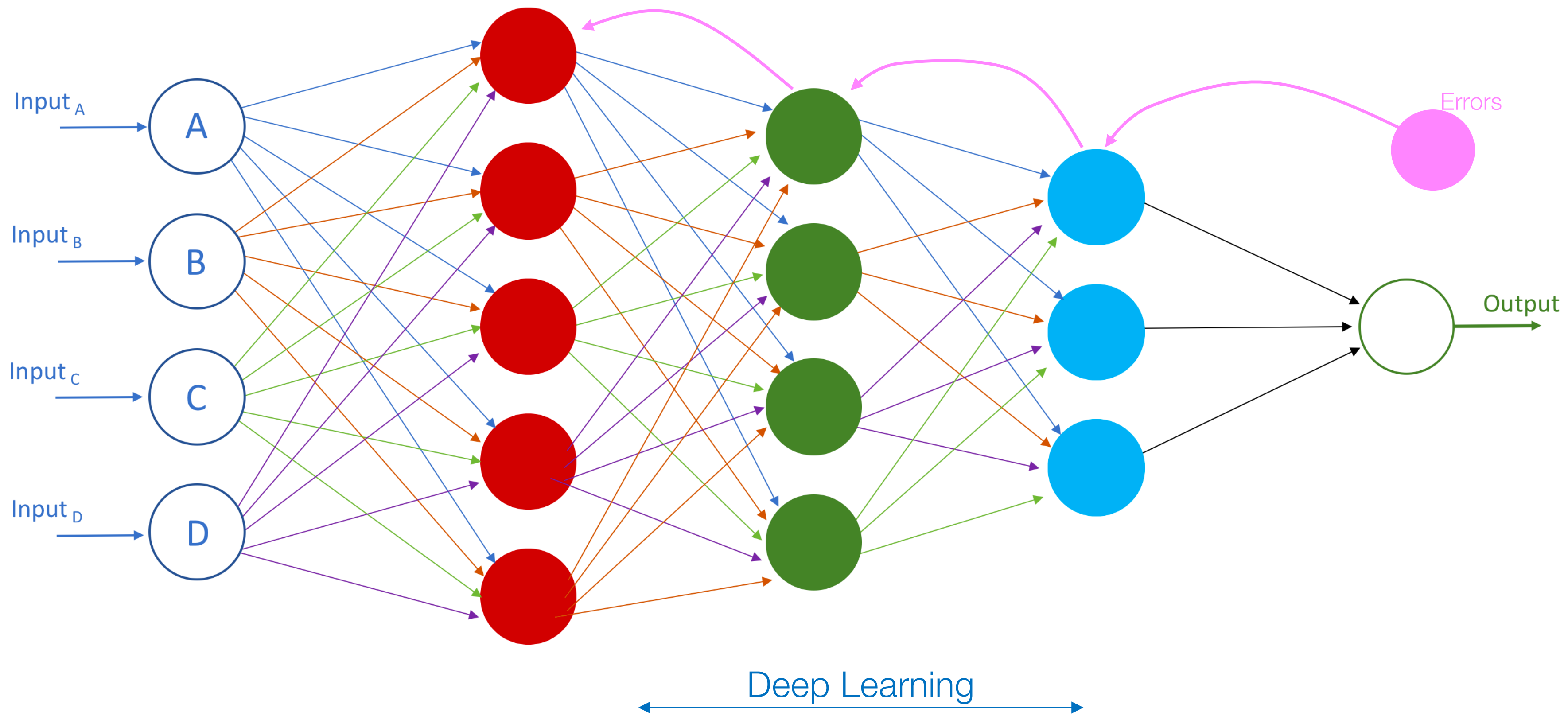




- Additional hidden layers can be added
- The complexity of the computations involved can increases dramatically.
- But the addition of additional hidden layers gives the Neural Network to find additional and at time very complex patterns hidden in the data.
    - Deep Learning
- Each hidden layer finds deeper connections between the different input features
- More hidden layers => more complex processing and time



- Back Propagation Neural Networks
  - Training Data Set and Error rates are fed into the network (hidden layers)
  - The processing at each neuron is updated and the weights rules adjusted to reduce the error rate.
  - The Neural Network model starts by assuming random weights to each of the connections in the network. The algorithm then iteratively updates the weights in the network by showing training data to the network and updating the network weights until the network is optimized, in a generalized way to avoid over fitting.
    - Neural Networks learn and then relearn/update themselves

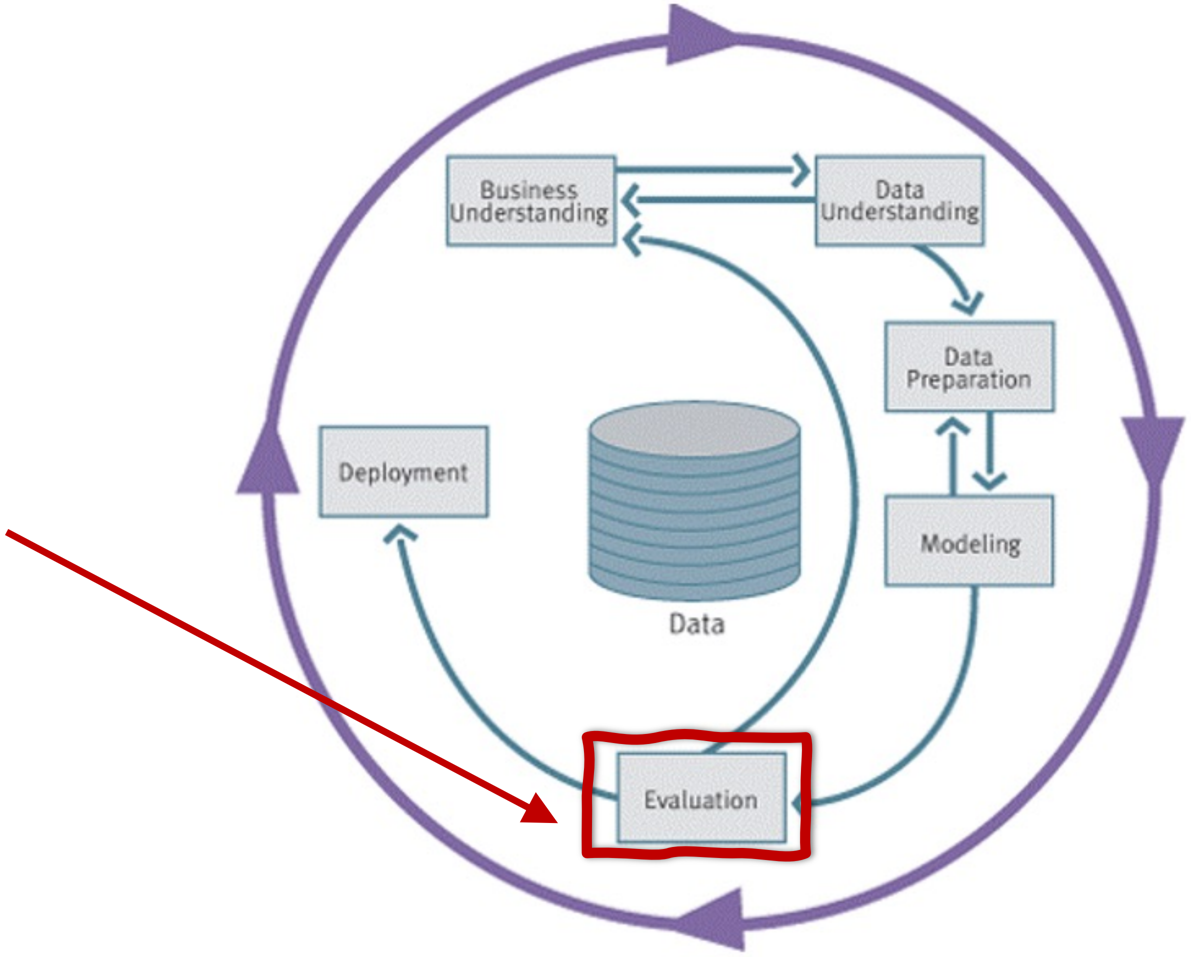


- Challenge 1 – Compile all the Accuracy Scores for each of the models
    - Which one has the best Accuracy score?
    - Can you create a bar chart to represent this data

    - We didn’t create a Naïve Bayes model for this data set
    - Go create one (using the same code) and calculate the Accuracy

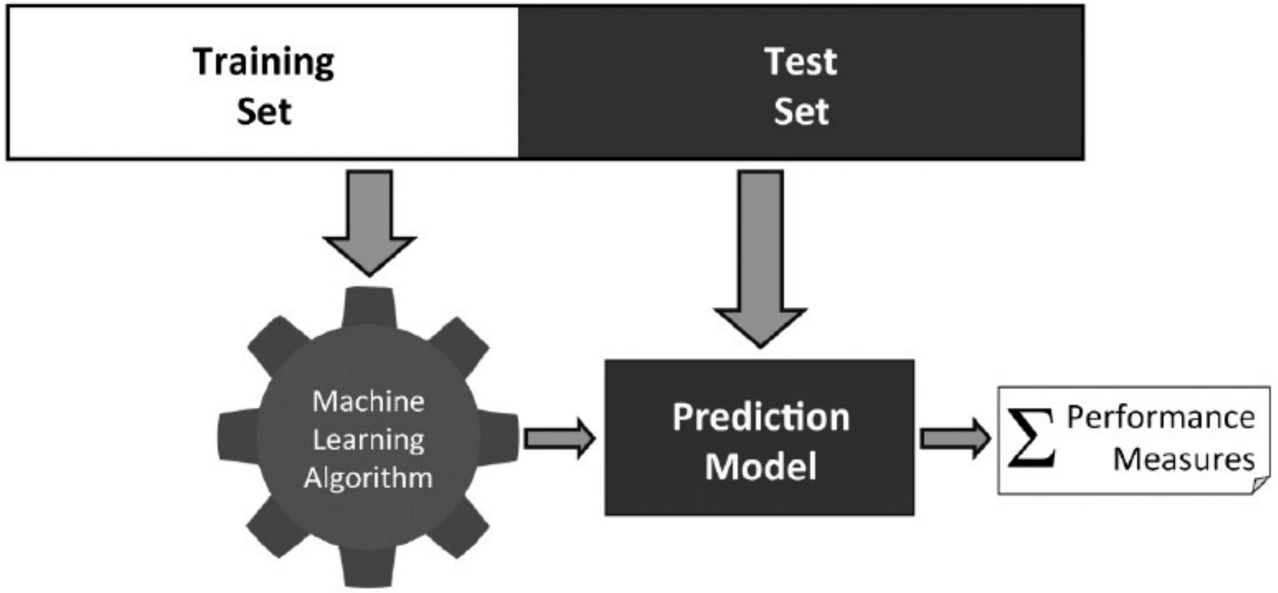
Test & Evaluate

Test & Evaluation



Test Dataset

- We run the model against the “unseen” dataset -> Test Dataset



The process of building and evaluating a model using a **hold-out test set**.

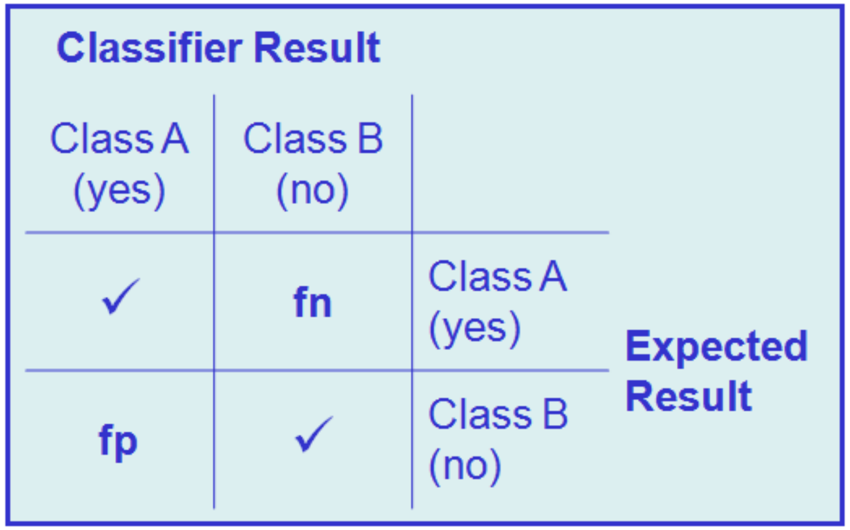
False Positives Vs False Negatives

- While it is useful to generate the simple accuracy of a classifier, sometimes we need more
- When is the classifier wrong?
  - False positives vs false negatives
  - Related to type I and type II errors in statistics

- Often there is a different cost associated with false positives and false negatives
  - Think about diagnosing diseases

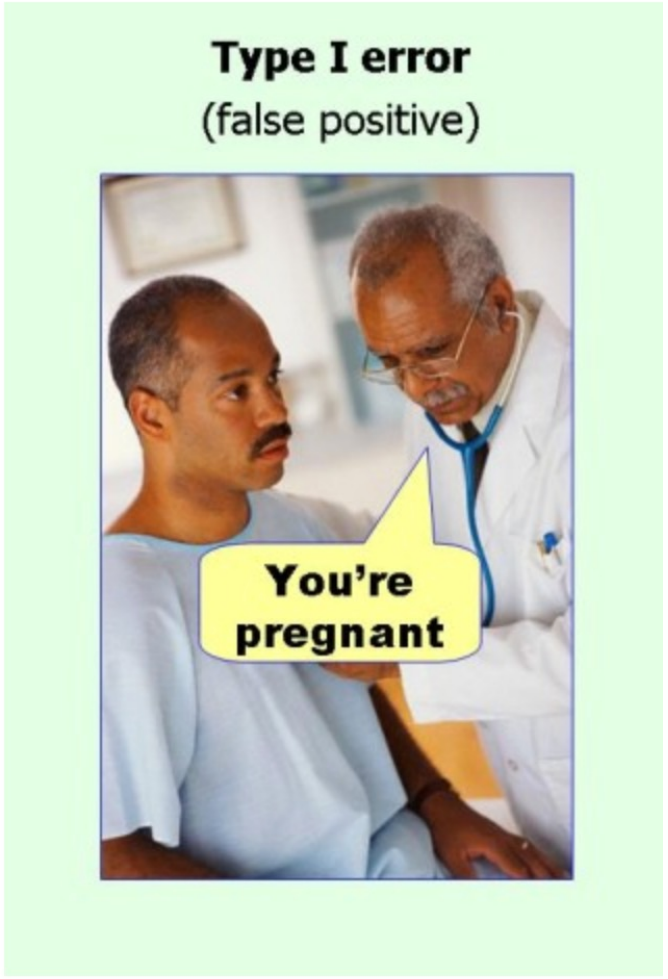
Confusion Matrix

- Confusion Matrix used to illustrate how a classifier is performing in terms of false positives and false negatives
- Gives us more information than a single accuracy figure
- Allows us to think about the cost of mistakes
- Can be extended to any number of classes
  - Binary Classification
  - Multi-Class Classification



Classifier Result

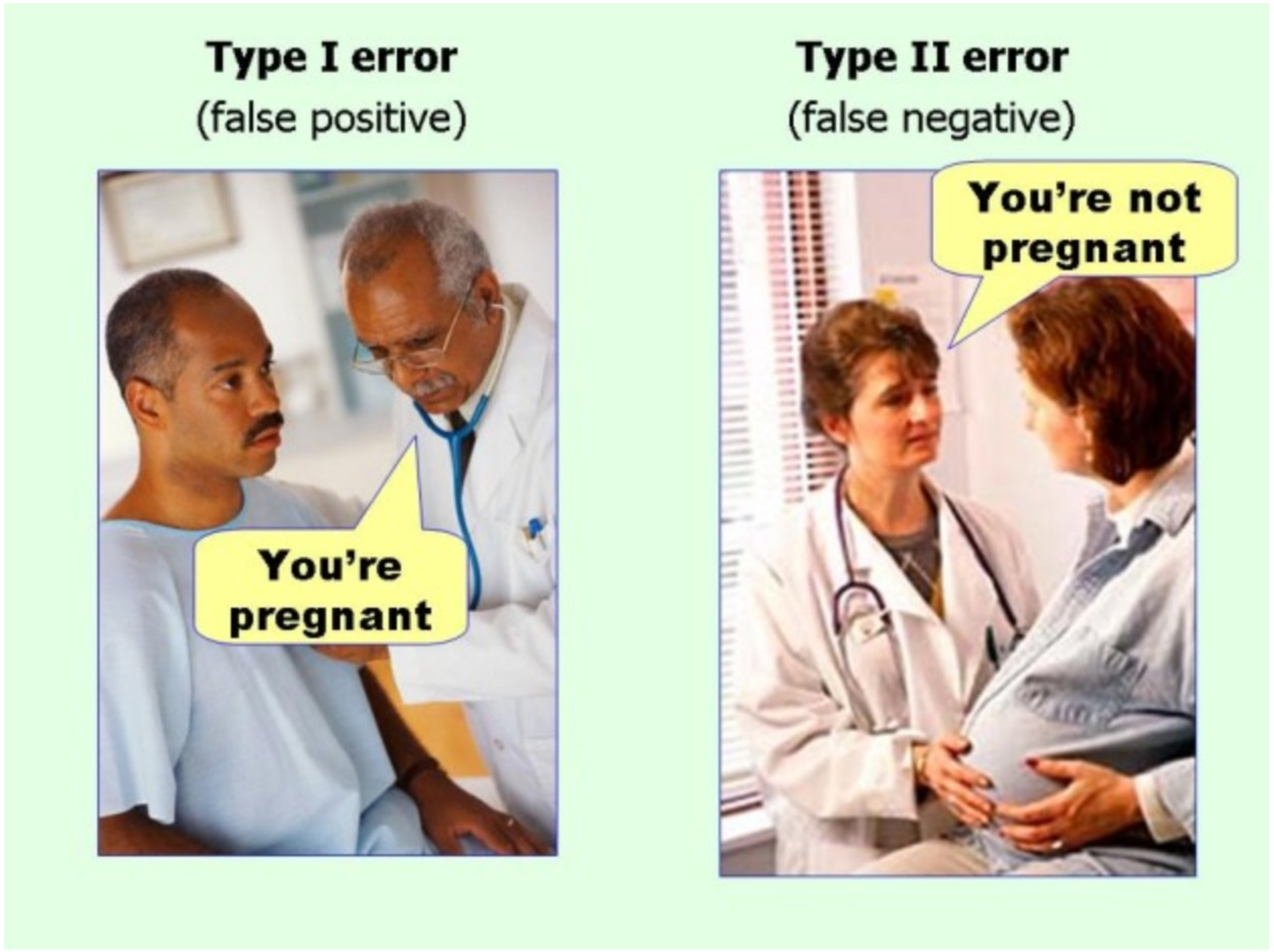
Class A (yes)	Class B (no)		
✓	**fn**	Class A (yes)	**Expected Result**
fp	✓	Class B (no)	



Type I & Type II Errors

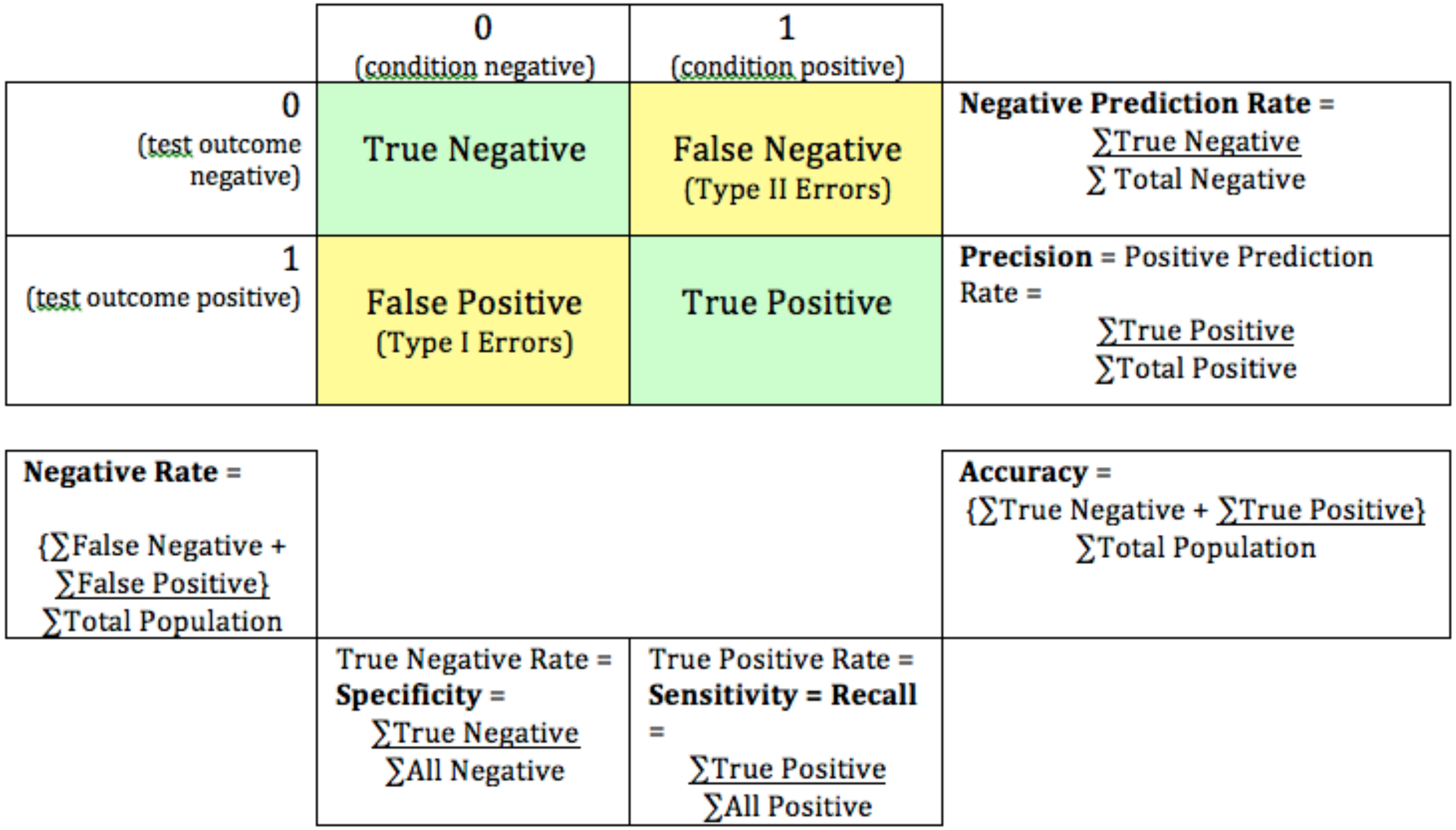
"Type I" and "Type II" errors, names first given by Jerzy Neyman and Egon Pearson to describe rejecting a null hypothesis when it's true and accepting one when it's not, are too vague for stat newcomers (and in general). This is better. [via]

Type I & Type II Errors

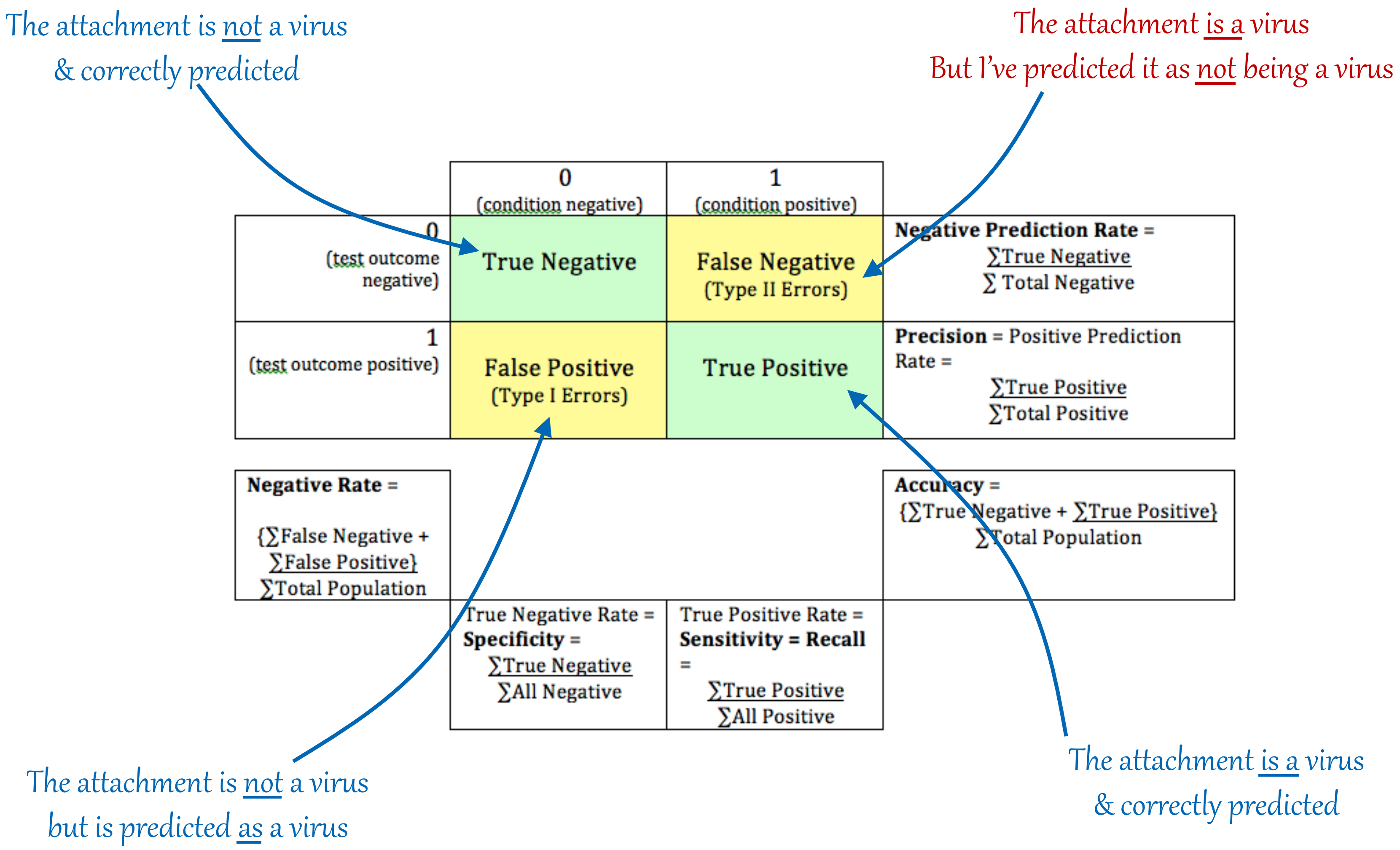


"Type I" and "Type II" errors, names first given by Jerzy Neyman and Egon Pearson to describe rejecting a null hypothesis when it's true and accepting one when it's not, are too vague for stat newcomers (and in general). This is better. [via]

Lots of different versions of this.
All are saying the same thing



	0 (condition negative)	1 (condition positive)	
0 (test outcome negative)	True Negative	False Negative (Type II Errors)	**Negative Prediction Rate** = $\frac{\sum \text{True Negative}}{\sum \text{Total Negative}}$
1 (test outcome positive)	False Positive (Type I Errors)	True Positive	**Precision** = Positive Prediction Rate = $\frac{\sum \text{True Positive}}{\sum \text{Total Positive}}$
Negative Rate = $\frac{\{\sum \text{False Negative} + \sum \text{False Positive}\}}{\sum \text{Total Population}}$			**Accuracy** = $\frac{\{\sum \text{True Negative} + \sum \text{True Positive}\}}{\sum \text{Total Population}}$
	True Negative Rate = **Specificity** = $\frac{\sum \text{True Negative}}{\sum \text{All Negative}}$	True Positive Rate = **Sensitivity = Recall** = $\frac{\sum \text{True Positive}}{\sum \text{All Positive}}$	



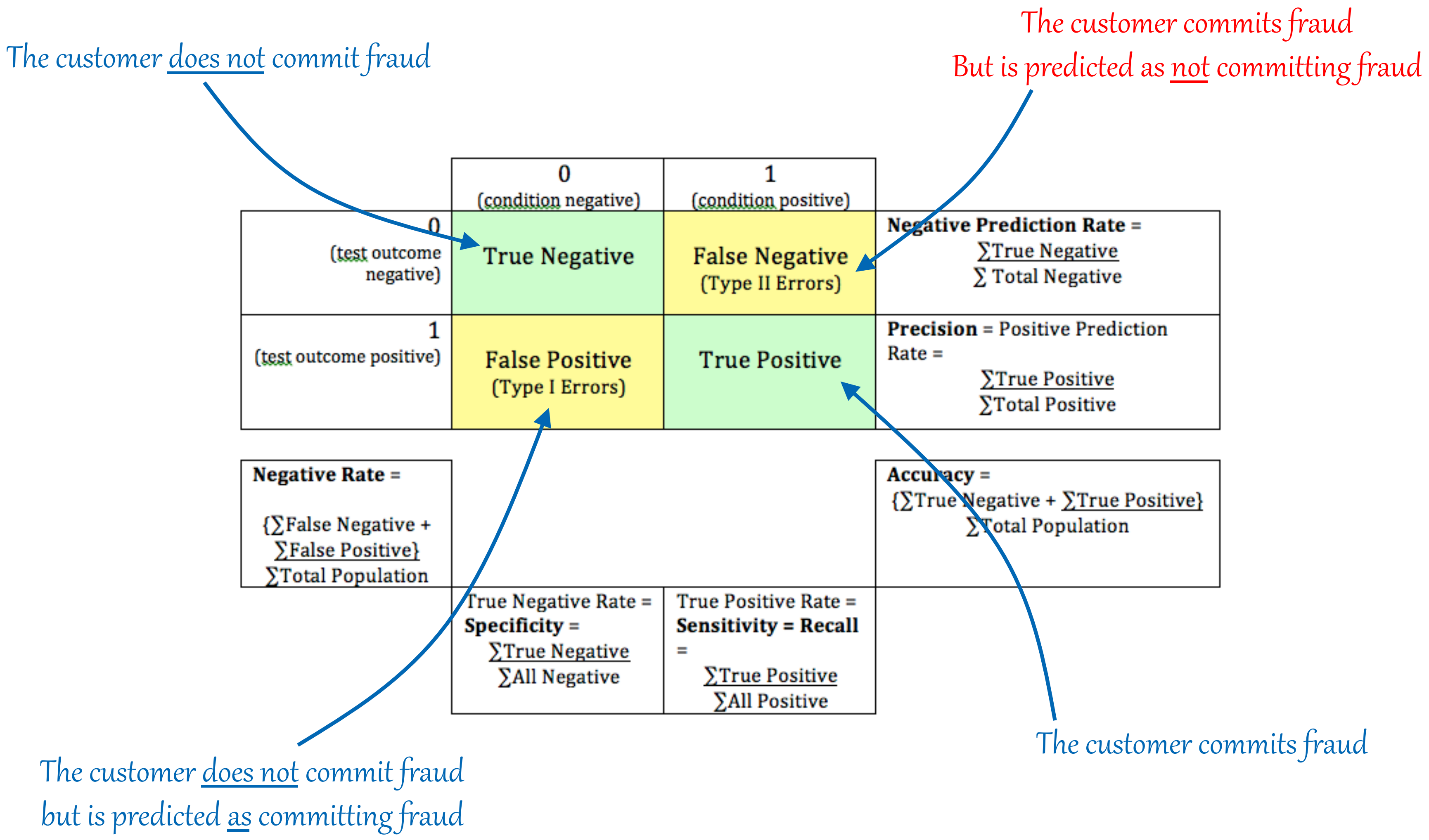
The attachment is not a virus
& correctly predicted

The attachment is a virus
But I've predicted it as not being a virus

	0 (condition negative)	1 (condition positive)	
0 (test outcome negative)	**True Negative**	**False Negative** (Type II Errors)	**Negative Prediction Rate** = $\frac{\sum\text{True Negative}}{\sum\text{ Total Negative}}$
1 (test outcome positive)	**False Positive** (Type I Errors)	**True Positive**	**Precision** = Positive Prediction Rate = $\frac{\sum\text{True Positive}}{\sum\text{Total Positive}}$
Negative Rate = $\frac{\{\sum\text{False Negative} + \sum\text{False Positive}\}}{\sum\text{Total Population}}$			**Accuracy** = $\frac{\{\sum\text{True Negative} + \sum\text{True Positive}\}}{\sum\text{Total Population}}$
	True Negative Rate = **Specificity** = $\frac{\sum\text{True Negative}}{\sum\text{All Negative}}$	True Positive Rate = **Sensitivity = Recall** = $\frac{\sum\text{True Positive}}{\sum\text{All Positive}}$	

The attachment is not a virus
but is predicted as a virus

The attachment is a virus
& correctly predicted



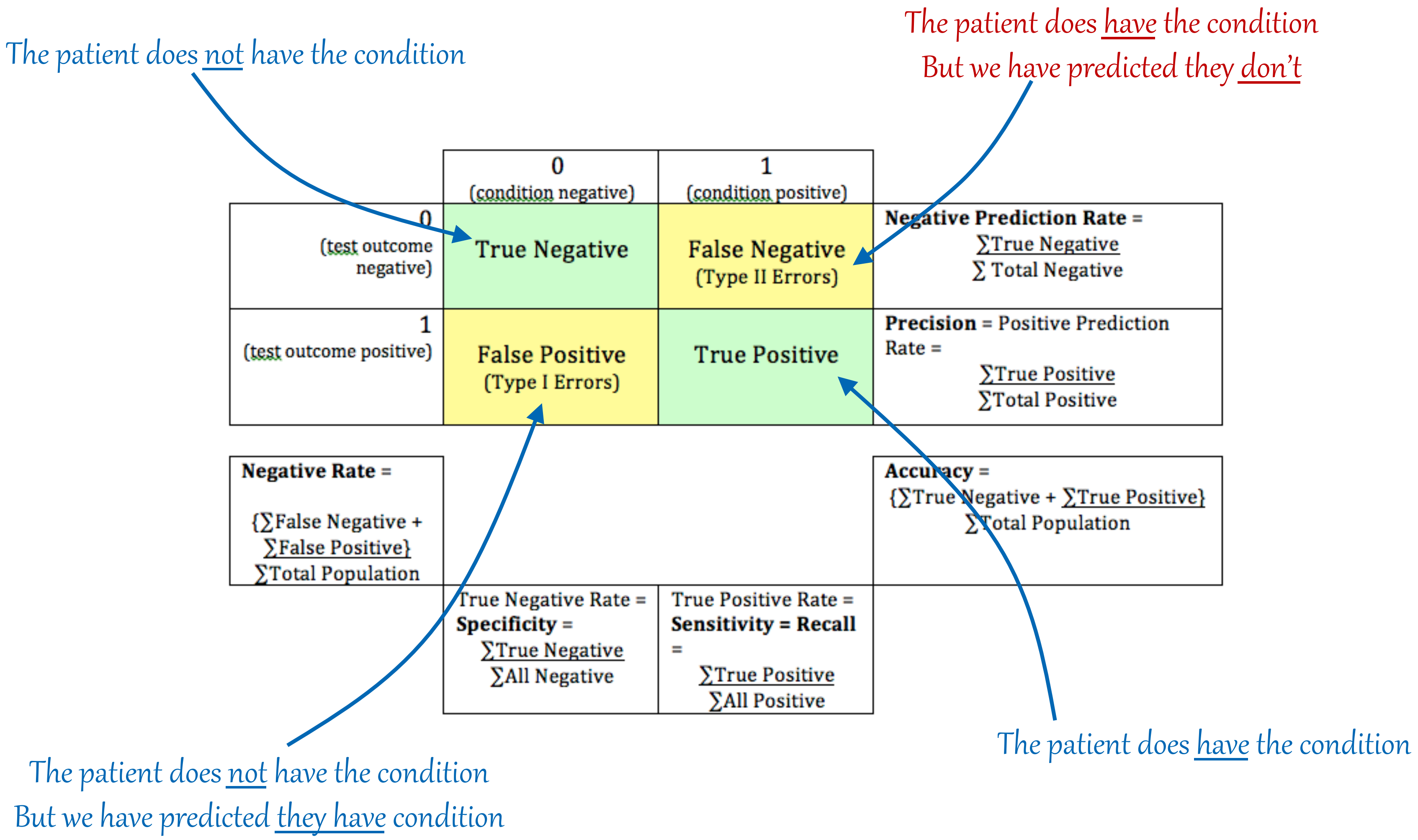
The customer does not commit fraud

The customer commits fraud
But is predicted as not committing fraud

	0 (condition negative)	1 (condition positive)	
0 (test outcome negative)	True Negative	False Negative (Type II Errors)	**Negative Prediction Rate** = ∑True Negative / ∑ Total Negative
1 (test outcome positive)	False Positive (Type I Errors)	True Positive	**Precision** = Positive Prediction Rate = ∑True Positive / ∑Total Positive
Negative Rate = {∑False Negative + ∑False Positive} / ∑Total Population			**Accuracy** = {∑True Negative + ∑True Positive} / ∑Total Population
	True Negative Rate = **Specificity** = ∑True Negative / ∑All Negative	True Positive Rate = **Sensitivity = Recall** = ∑True Positive / ∑All Positive	

The customer does not commit fraud
but is predicted as committing fraud

The customer commits fraud



The patient does not have the condition

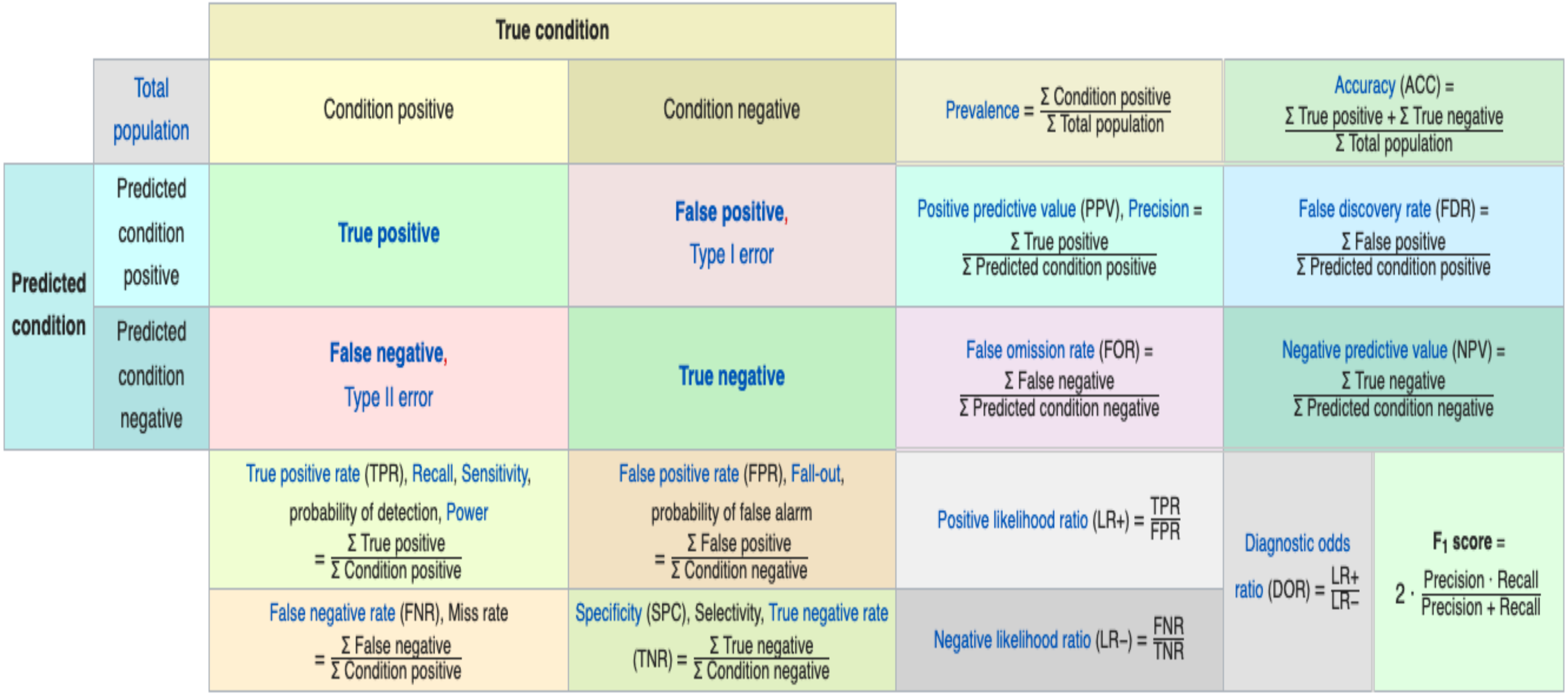
The patient does have the condition
But we have predicted they don't

	0 (condition negative)	1 (condition positive)	
0 (test outcome negative)	True Negative	False Negative (Type II Errors)	**Negative Prediction Rate** = $\frac{\sum \text{True Negative}}{\sum \text{Total Negative}}$
1 (test outcome positive)	False Positive (Type I Errors)	True Positive	**Precision** = Positive Prediction Rate = $\frac{\sum \text{True Positive}}{\sum \text{Total Positive}}$
Negative Rate = $\frac{\{\sum \text{False Negative} + \sum \text{False Positive}\}}{\sum \text{Total Population}}$			**Accuracy** = $\frac{\{\sum \text{True Negative} + \sum \text{True Positive}\}}{\sum \text{Total Population}}$
	True Negative Rate = **Specificity** = $\frac{\sum \text{True Negative}}{\sum \text{All Negative}}$	True Positive Rate = **Sensitivity = Recall** = $\frac{\sum \text{True Positive}}{\sum \text{All Positive}}$	

The patient does not have the condition
But we have predicted they have condition

The patient does have the condition

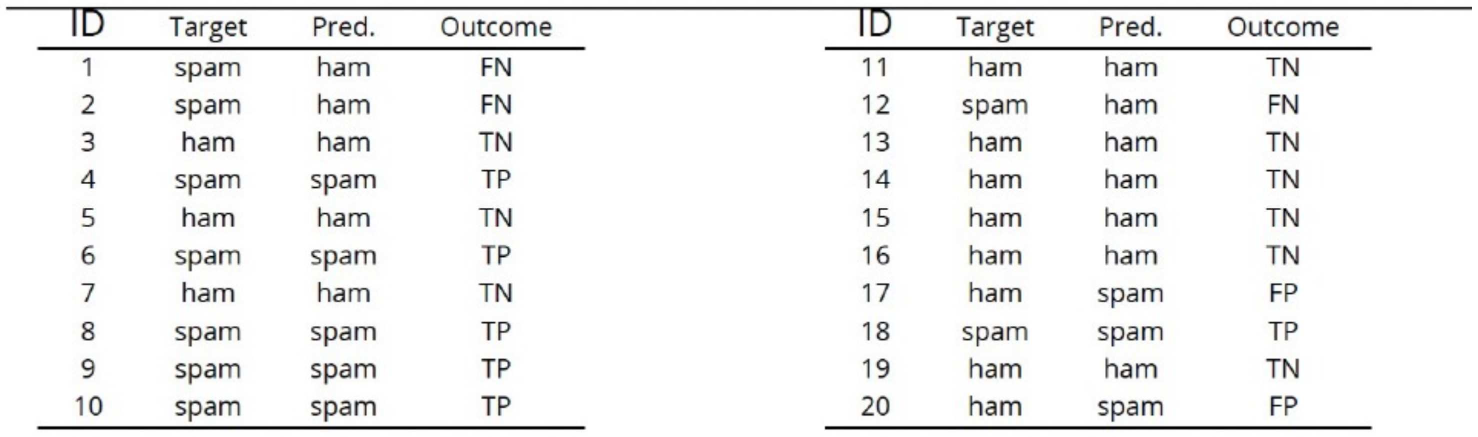
Confusion Matrix



		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive,** Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative,** Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	**F_1 score** $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

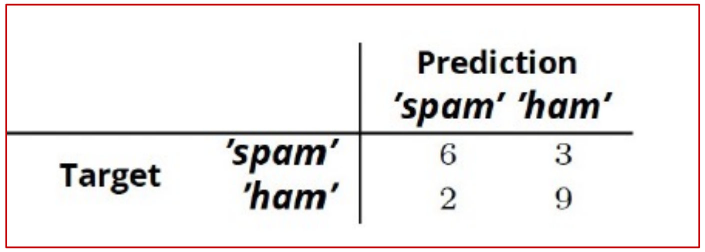
Confusion Matrix - Example

A sample test set with model predictions.



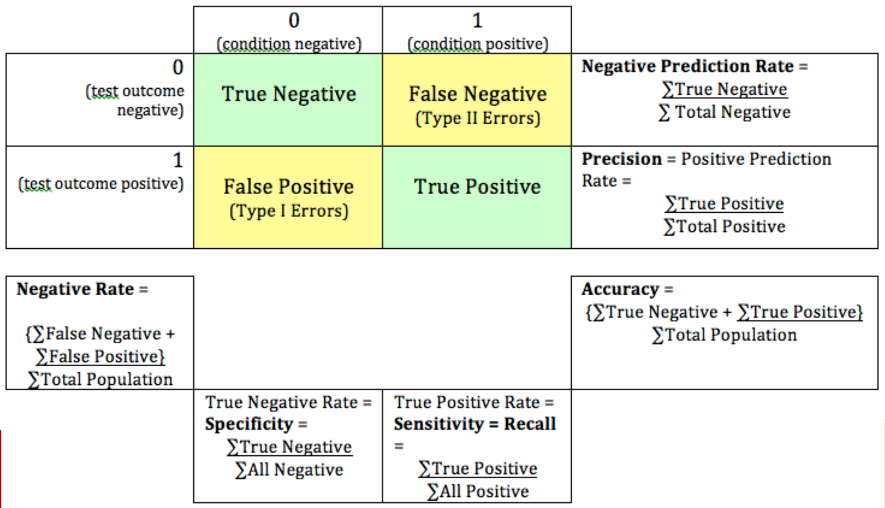
ID	Target	Pred.	Outcome
1	spam	ham	FN
2	spam	ham	FN
3	ham	ham	TN
4	spam	spam	TP
5	ham	ham	TN
6	spam	spam	TP
7	ham	ham	TN
8	spam	spam	TP
9	spam	spam	TP
10	spam	spam	TP

ID	Target	Pred.	Outcome
11	ham	ham	TN
12	spam	ham	FN
13	ham	ham	TN
14	ham	ham	TN
15	ham	ham	TN
16	ham	ham	TN
17	ham	spam	FP
18	spam	spam	TP
19	ham	ham	TN
20	ham	spam	FP



		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	6	3
	'ham'	2	9

Confusion Matrix - Example



	0 (condition negative)	1 (condition positive)	
0 (test outcome negative)	True Negative	False Negative (Type II Errors)	**Negative Prediction Rate** = ∑True Negative / ∑ Total Negative
1 (test outcome positive)	False Positive (Type I Errors)	True Positive	**Precision** = Positive Prediction Rate = ∑True Positive / ∑Total Positive
Negative Rate = {∑False Negative + ∑False Positive} / ∑Total Population			**Accuracy** = {∑True Negative + ∑True Positive} / ∑Total Population
	True Negative Rate = **Specificity** = ∑True Negative / ∑All Negative	True Positive Rate = **Sensitivity = Recall** = ∑True Positive / ∑All Positive	

$$\text{precision} = \frac{6}{(6+2)} = 0.75$$

$$\text{misclassification accuracy} = \frac{(2+3)}{(6+9+2+3)} = 0.25$$

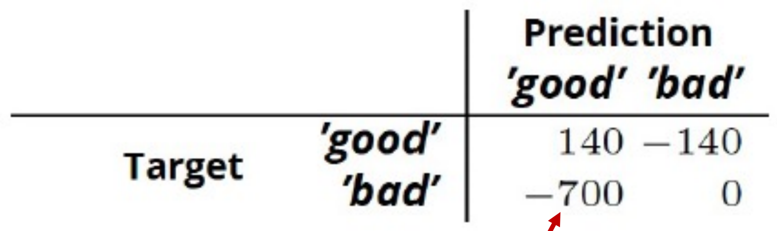
$$\text{classification accuracy} = \frac{(6+9)}{(6+9+2+3)} = 0.75$$

$$\text{recall} = \frac{6}{(6+3)} = 0.667$$

Confusion Matrix – Example – What about Costs

- Not every outcome (or classification) has the same value
- A positive outcome could be worth money €
- A negative outcome could be work lots of money lost -€€€
- We can apply monetary values to the outcomes

Sample profit matrix for a credit scoring problem.



		Prediction	
		'good'	*'bad'*
Target	*'good'*	140	−140
	'bad'	−700	0

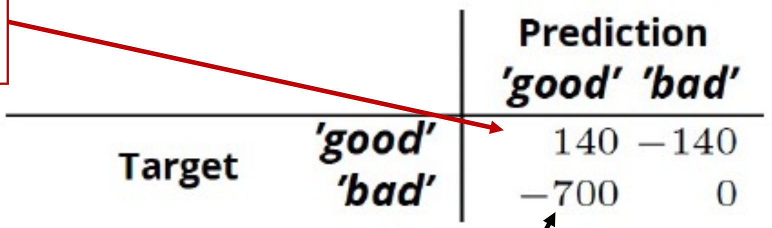
It was predicted as a "Good" Risk
But in reality it turned out to be "Bad".
In this case, how much on average would such a scenario cost us

Confusion Matrix – Example – What about Costs

- Not every outcome (or classification) has the same value
- A positive outcome could be worth money €
- A negative outcome could be work lots of money lost -€€€
- We can apply monetary values to the outcomes

Sample profit matrix for a credit scoring problem.

How much € can we make by getting this correct



		Prediction	
		'good'	*'bad'*
Target	*'good'*	140	-140
	'bad'	-700	0

We should have approved these.
But we didn't.
Missed opportunity cost

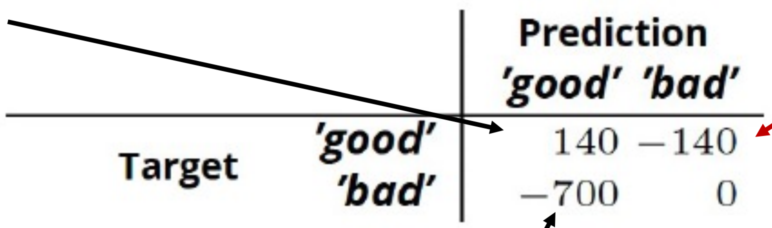
We will do nothing with these.
So no cost/€

It was predicted as a "Good" Risk
But in reality it turned out to be "Bad".
In this case, how much on average would such a scenario cost us

Confusion Matrix – Example – What about Costs

- Not every outcome (or classification) has the same value
- A positive outcome could be worth money €
- A negative outcome could be work lots of money lost **-€€€**
- We can apply monetary values to the outcomes

Sample profit matrix for a credit scoring problem.



		Prediction	
		'good'	***'bad'***
Target	***'good'***	140	−140
	'bad'	−700	0

How much € can we make by getting this correct

We should have approved these.
But we didn't.
Missed opportunity cost

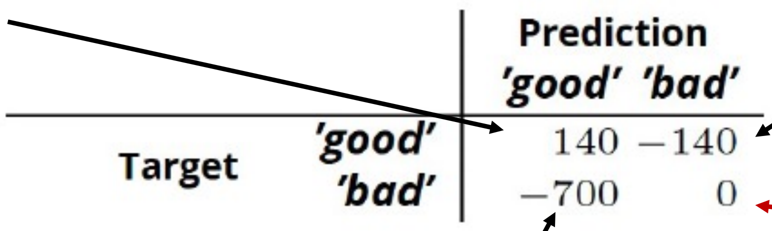
We will do nothing with these.
So no cost/€

It was predicted as a "Good" Risk
But in reality it turned out to be "Bad".
In this case, how much on average would such a scenario cost us

Confusion Matrix – Example – What about Costs

- Not every outcome (or classification) has the same value
- A positive outcome could be worth money €
- A negative outcome could be work lots of money lost **-€€€**
- We can apply monetary values to the outcomes

Sample profit matrix for a credit scoring problem.



		Prediction	
		'good'	***'bad'***
Target	***'good'***	140	−140
	'bad'	−700	0

How much € can we make by getting this correct

We should have approved these. But we didn't. Missed opportunity cost

We will do nothing with these. So no cost/€

It was predicted as a "Good" Risk
But in reality it turned out to be "Bad".
In this case, how much on average would such a scenario cost us

Confusion Matrix – Example – What about Costs

- Add up the numbers from each part/box of the Confusion Matrix
    - Total = € potential value for model

- Accountants like to see these numbers
- Your managers like to see these numbers
- Bosses like to see these numbers

- It quantifies/costs the potential € for each model

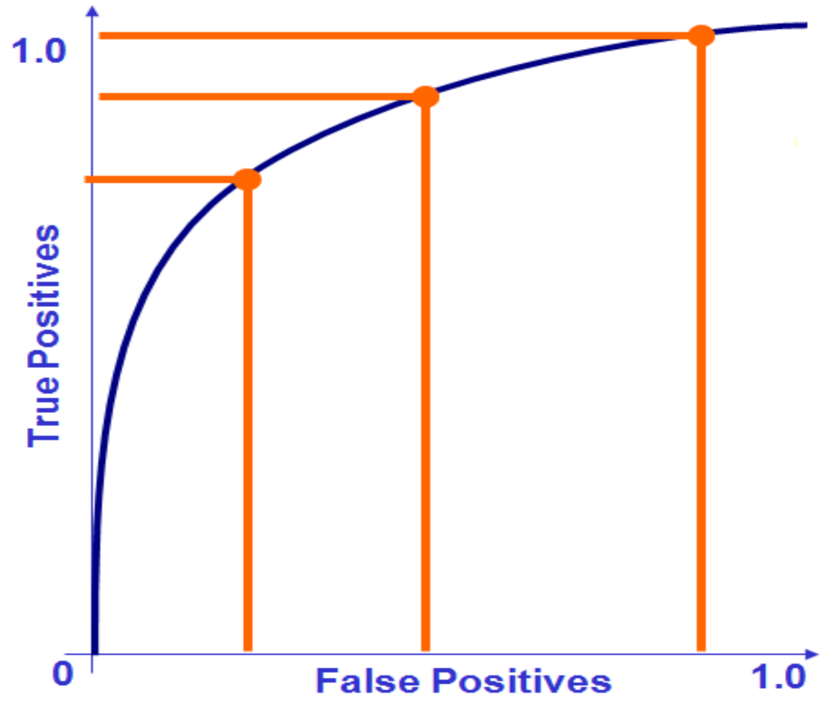
- The Business will understand using € (Language of Business)
    - Vs using Numbers + Percentages + Unusual Terms   (Language of Analysts, Machine Learning, etc)

ROC Curves

- *Receiver Operating Characteristic* (ROC) curves were originally used to make sense of noisy radio signals
- Can be used to help us talk about classifier performance and determine the best operating point for a classifier

Considers how the relationship between true positives and false positives can change

We need to choose the best operating point



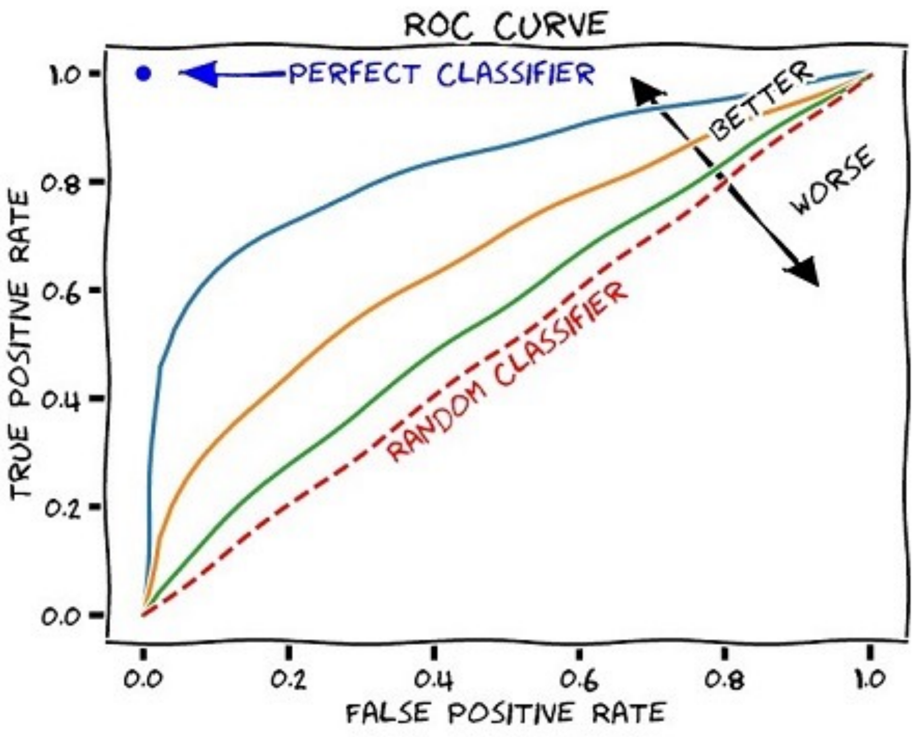
ROC Curves (cont…)

- ROC curves can be used to compare classifiers
- The greater the Area Under the Curve (AUC) the more accurate the classifier

AUC stands for *Area under the ROC Curve.* It provides an aggregate measure of performance across all possible classification thresholds.

The higher the area under the ROC curve (AUC), the better the classifier.

A perfect classifier would have an AUC of 1. Usually, if your model behaves well, you obtain a good classifier by selecting the value of the threshold that gives TPR close to 1 while keeping FPR near 0.



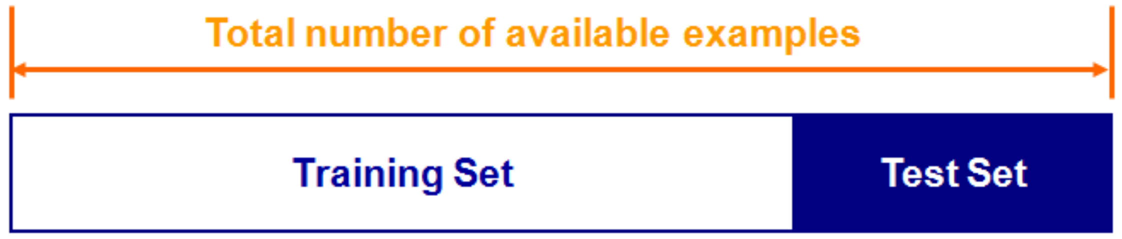
- Challenge 1 – Compile all the Accuracy Scores for each of the models
  - Which one has the best Accuracy score?
  - Can you create a bar chart to represent this data

  - We didn’t create a Naïve Bayes model for this data set
  - Go create one (using the same code) and calculate the Accuracy

- Challenge 2 – Calculate the ROC and AUC for all the models
  - Which one has the best AUC value?

Hold-Out Testing Sets

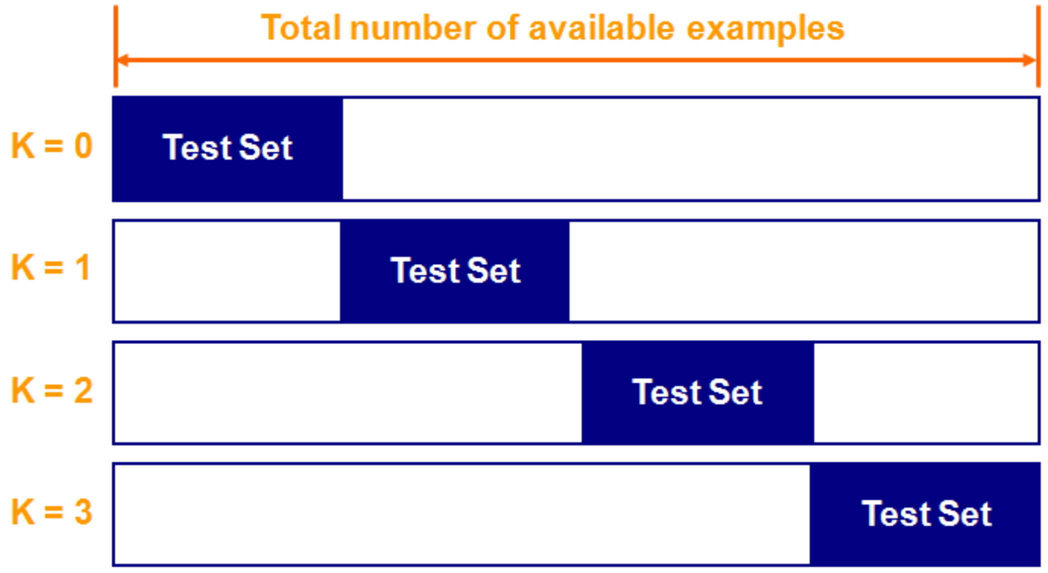
- Split the available data into a *training set* and a *test set*



- Train the classifier in the training set and evaluate based on the test set
- A couple of drawbacks
  - We may not have enough data
  - We may happen upon an *unfortunate split*

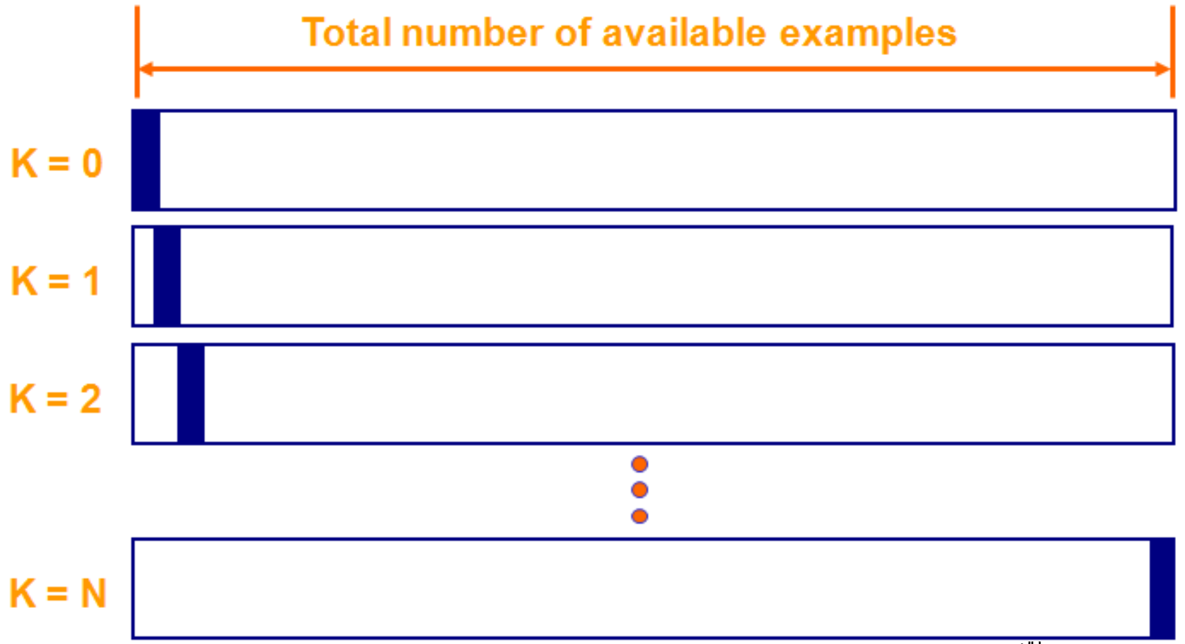
K-Fold Cross Validation

- An alternate is to divide the dataset into smaller chunks (Train & Test)
- k *folds* – where k is the number of times to divide the data
- For each of k experiments, use kth fold for testing and everything else for training
- Average the results across the k folds



K-Fold Cross Validation

- The accuracy of the system is calculated as the average error across the k folds
- The main advantages of k-fold cross validation are that every example is used in testing at some stage and the problem of an *unfortunate split* is avoided
- Any value can be used for k
  - 10 is most common
  - Depends on the data set



- Challenge 1 – Compile all the Accuracy Scores for each of the models
    - Which one has the best Accuracy score?
    - Can you create a bar chart to represent this data

    - We didn’t create a Naïve Bayes model for this data set
    - Go create one (using the same code) and calculate the Accuracy

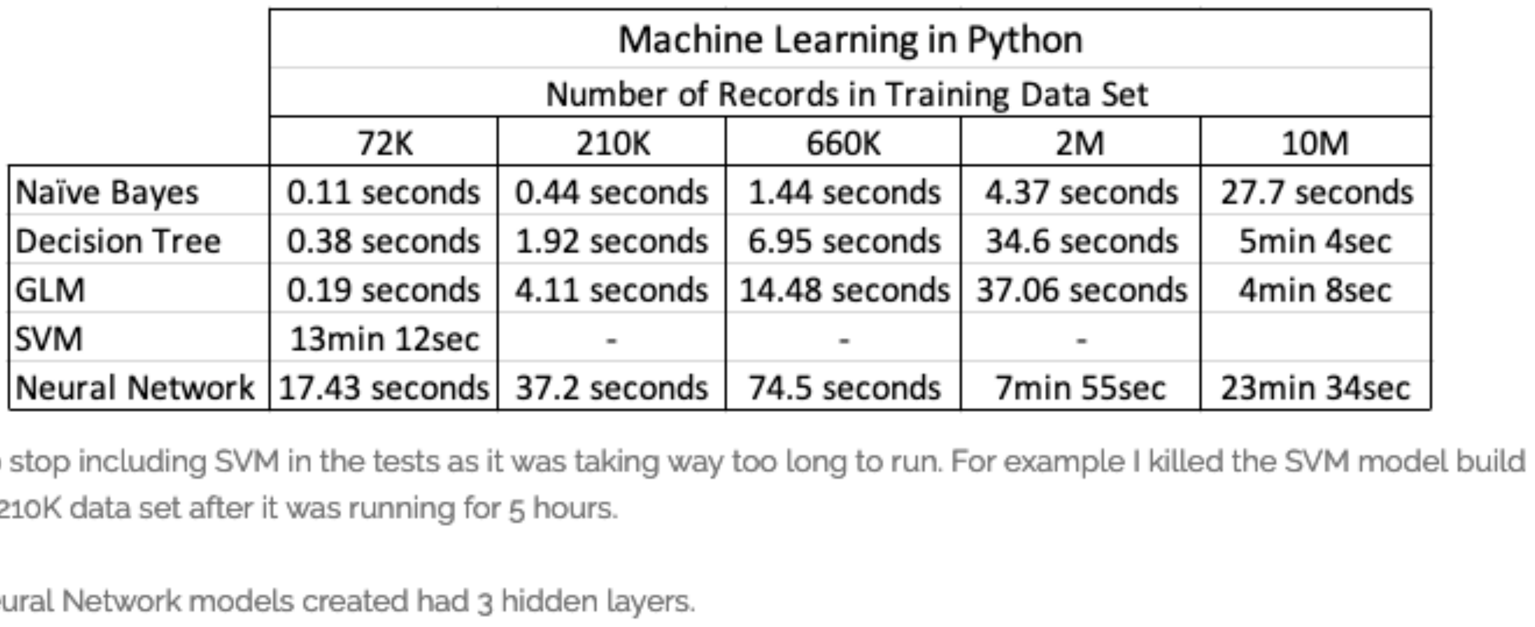
- Challenge 2 – Calculate the ROC and AUC for all the models
    - Which one has the best AUC value?

- Challenge 3 – Using Kfold Cross-Validation for all models
    - Which one has the best average Accuracy score?

Experiment

Experience

How long does Machine Learning take



	Machine Learning in Python				
	Number of Records in Training Data Set				
	72K	210K	660K	2M	10M
Naïve Bayes	0.11 seconds	0.44 seconds	1.44 seconds	4.37 seconds	27.7 seconds
Decision Tree	0.38 seconds	1.92 seconds	6.95 seconds	34.6 seconds	5min 4sec
GLM	0.19 seconds	4.11 seconds	14.48 seconds	37.06 seconds	4min 8sec
SVM	13min 12sec	-	-	-	
Neural Network	17.43 seconds	37.2 seconds	74.5 seconds	7min 55sec	23min 34sec

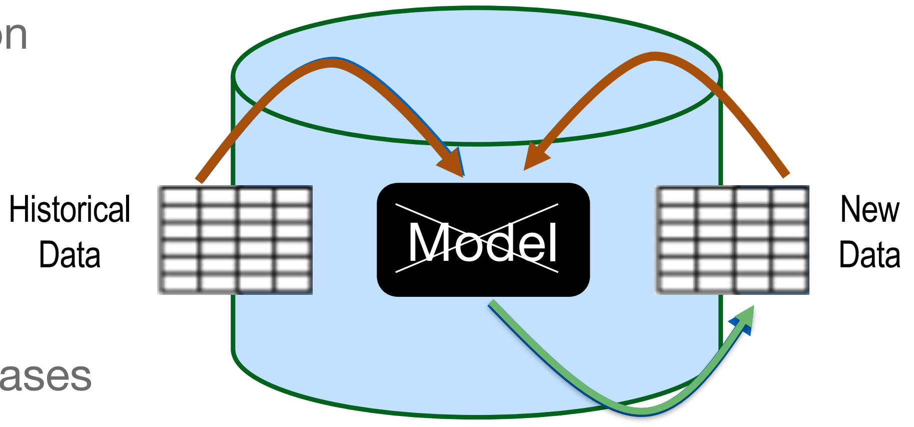
I had to stop including SVM in the tests as it was taking way too long to run. For example I killed the SVM model build on the 210K data set after it was running for 5 hours.

The Neural Network models created had 3 hidden layers.

https://oralytics.com/2019/02/05/machine-learning-models-in-python-how-long-does-it-take/

Model Update

- Do we need to update the models ?
  - New (initial) Model
    - Reflects the current position



  - We keep gathering new data/cases
  - Update a Model

Concept Drift

- How often do we need to update the models ?

A lot covered

- We have covered a lot in this class (and previous)

- What is Classification & What it can be used for
- Different Algorithms
- How to Evaluate

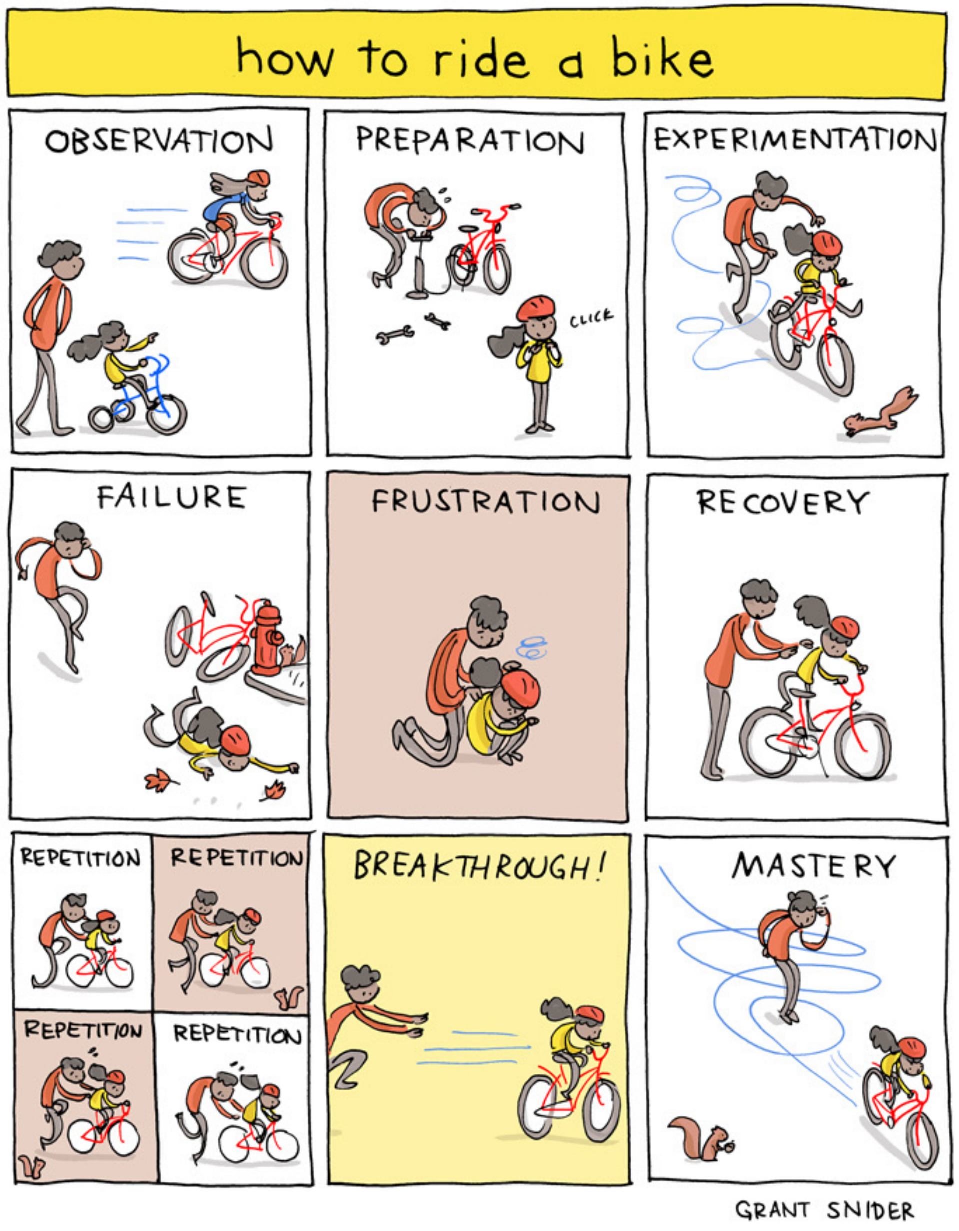
- Keep It Simple!
- Lab Work
  - Examples of the Algorithms
  - Examples of Evaluation

- It's only a few lines of code!

- Assignment A – is based on topics covered up to and including this week + Next Week
  - Topics after next week (Week 7) are not part of or required for Assessment-A
  - Topics in Week 7 and after will be part of Assessment-B
  - Topics in Week 7 – first half for Assessment-A + second half for Assessment-B

EASY
HARD

EASY
HARD


how to ride a bike
OBSERVATION
PREPARATION
CLICK
EXPERIMENTATION
FAILURE
FRUSTRATION
RECOVERY
REPETITION
REPETITION
REPETITION
REPETITION
BREAKTHROUGH!
MASTERY
GRANT SNIDER

Time for an Example

Any Questions ?

What Now/Next ?