

# Machine Learning Concepts

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Machine Learning Terminology

## **Machine Learning Terminology**

Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

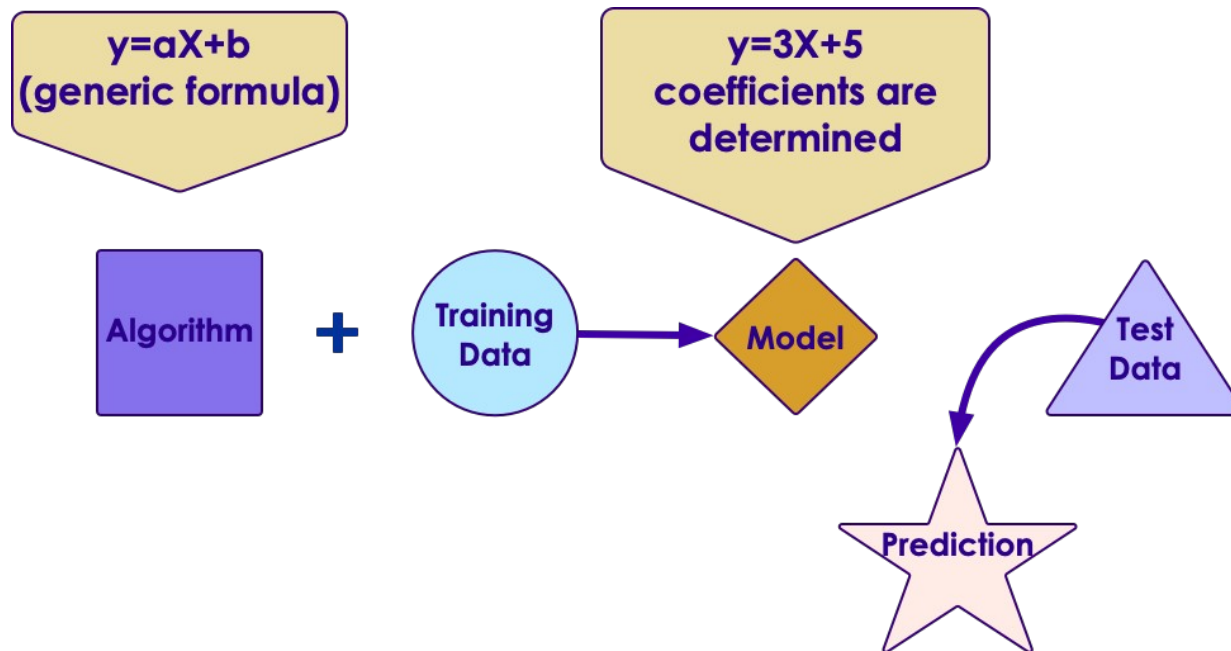
# ML Terminology

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

Algorithm	Example	Input	Output
Classification – produces 'discrete' or 'qualitative'	Spam or not	Features (email text, origin IP address)	Label (Spam / Not-Spam)
Regression – produces 'continuous' or 'quantitative'	Weather forecasting, predicting temperature	Variables (current temperature, pressure ..etc)	Target (predicted temperature -50.5 ' F)

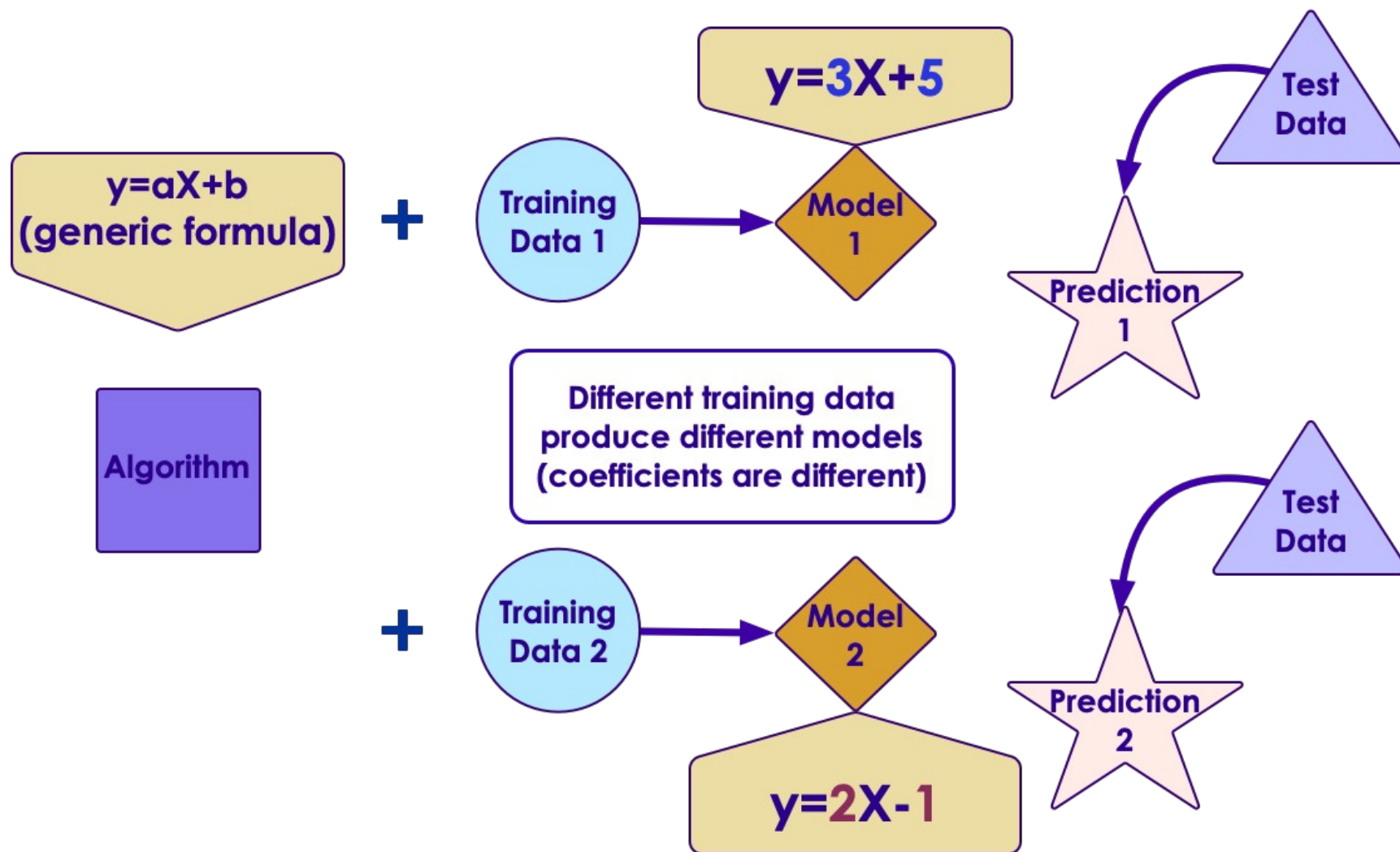
# Algorithm vs. Model

- ◆ Often Algorithm and Model are used interchangeably
- ◆ Algorithm: Mathematical / statistical formula or methodology
- ◆ Training the algorithm with data yields a model
  - Algorithm + training data -> model
  - Model = Algorithm(data)



# Algorithm and Model

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @ 2019-03-12



Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

- ◆ We can treat ML model as a 'black box'
- ◆ Input goes in, model produces an output
- ◆ **'Model'** :
  - Mathematical object describing the relationship between input and output



# Feature Engineering

Machine Learning Terminology  
**Feature Engineering**  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Feature Engineering

- ◆ **Feature Engineering:**
  - *"Using transformations of raw input data to create new features to be used in ML model"*
- ◆ **Feature Engineering examples**
  - Join two database tables to have all data in one place
  - Convert to same units of measurements (imperial to metric)
  - Enriching data by combining with other data sources e.g. combining house sales prices with census data



# Features To Consider: Class Quiz

- ◆ Assume we are evaluating a credit card application
- ◆ **Q: What data points to consider?**
- ◆ **Q: How will you get this data?**
  - For existing customer?
  - For new customer?

Customer_id	Name	Zipcode	Age	Income	Marital Status	Owns a Home
1	Joe	11111	24	45,000	Single	No
2	Jane	22222	34	84,000	Married	Yes

# Feature Extraction

- ◆ Here is a sample data for credit card applications
- ◆ Our algorithm only uses selected input (features) to determine credit worthiness
- ◆ Here 'name' and 'zipcode' aren't considered

# Feature Extraction

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

Customer_id	Name	Zipcode	Age	Income	Marital Status	Owns a Home
1	Joe	11111	24	45,000	Single	No
2	Jane	22222	34	84,000	Married	Yes

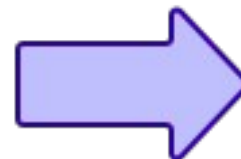


**Feature Extraction**

			Age	Income	Marital Status	Owns a Home
			24	45,000	Single	No
			34	84,000	Married	Yes



**Algorithm**



**Credit Limit**

# Categorical Variables

- ◆ Some of the variables have string content
- ◆ Example: Marital Status / Owns a Home
- ◆ Feature vectors must be numeric.
- ◆ We have to convert the variable to a numeric value.
- ◆ Example: "Owns A Home"  $\rightarrow 0 = \text{No}, 1 = \text{Yes}$
- ◆ Categorical Variables are essentially structured data, despite being strings.
- ◆ Unstructured data would include things like: documents, emails, tweets

			Age	Income	Marital Status	Owns a Home
			24	45,000	Single	No
			34	84,000	Married	Yes

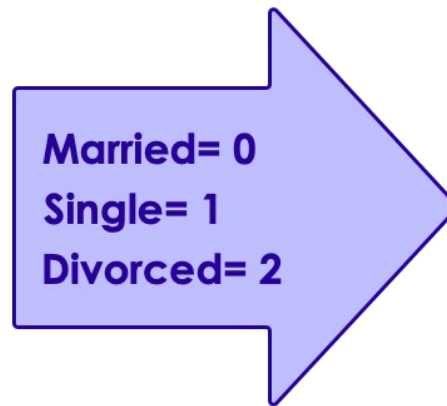
# Encoding Categorical Variables

- ◆ We have to convert our categorical variables into numbers
- ◆ 3 Strategies:
  - Factorization / Indexing
  - One-Hot-Encoding/Dummy Variables
  - Quantization

# Example of Factorization / Indexing

- ◆ We can convert our string variables into factors / numbers
- ◆ This means we assign a number to each unique value of the column
- ◆ Added benefits
  - Numbers are more efficient to store
  - And compute!

id	status
1	married
2	single
3	married
4	Divorced
5	single



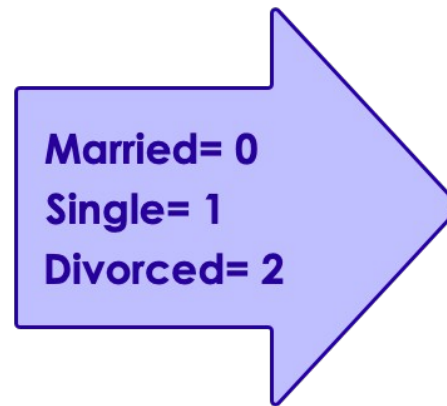
id	status idx
1	0
2	1
3	0
4	2
5	1

# Potential Problems With Factorization / Indexing

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ Some ML algorithms can start interpreting the numbers!
- ◆ In the example below, an ML algorithm can think
  - 2 (Divorced) > 1 (Single) > 0 (Married)
- ◆ This can lead to surprising outcomes
- ◆ We can fix this by 'one-hot-encoding' method

id	status
1	married
2	single
3	married
4	Divorced
5	single



id	status idx
1	0
2	1
3	0
4	2
5	1

# Dummy Variables / One-Hot-Encoding

- ◆ Dummy variables can help us treat the different values separately
  - Without trying to infer some relationship between values.
- ◆ 'dummy variables' assigns true / false to each.
  - Note, only one bit is on
  - This is called **ONE-HOT-Encoding**

id	status
1	married
2	single
3	married
4	Divorced
5	single



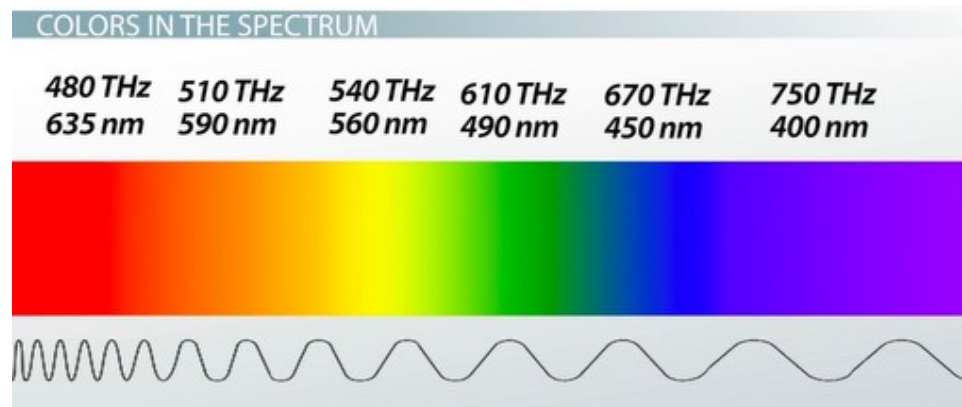
id	Status idx
1	0
2	1
3	0
4	2
5	1



id	is married	is single	is divorced
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0



- ◆ Sometimes we do want the ML model to interpret categorical variables
  - Grades :  $A > B > C > D$
  - Domain specific meaning
- ◆ For example, colors in physics has a numeric meaning:
  - Red: 480 THz frequency of light
  - Green: 600 THz
- ◆ This might allow our models to make inferences
  - e.g., Orange is close to red on the spectrum, but more distant from violet.



# FE: Generating New Dimensions

- ◆ Problem: Comparing house prices
- ◆ Can we say Mountain View is most expensive city?
- ◆ On first table, there is no data point for 'size of the house'
- ◆ May be an 'apples-to-apples' comparison would be 'price per sq. foot'

City	House Price
San Jose	800k
Mountain View	1,200 k (1.2M)
San Francisco	1,000 k (1 M)
Gilroy	700 k

City	House Price	Sq. ft	Price / sq. feet
San Jose	\$ 800k	2000	\$ 400 / sqft
Mountain View	\$ 1,200 k (1.2M)	1500	\$ 800 / sqft
<b>San Francisco</b>	<b>\$ 1,000 k (1 M)</b>	<b>1000</b>	<b>\$ 1000 / sqft</b>
Gilroy	\$ 700 k	4000	\$175 / sqft

# FE: Group Discussion

## ◆ Problem:

- A comment is left on the website. Determine if it is a 'spam' comment or legitimate comment

## ◆ Data we have:

- Comment text
- IP address of user

## ◆ Discuss:

- What other data you may need to determine if the comment is 'spam' ?
- How can we acquire the data?

# Converting Word to Vectors

## Document 1

Cat dog cow

## Document 2

Cat dog

## Document 3

Cat Cow

## Document 4

Cow Cow Dog

Document/ Term  
Matrix

word frequency	cat	cow	dog	vector
doc 1	1	1	1	[ 1, 1, 1 ]
doc 2	1	0	1	[ 1, 0, 1 ]
doc 3	1	1	0	[ 1, 1, 0 ]
doc 4	0	2	1	[ 0, 2, 1 ]

# FE: Dealing With Time

- ◆ Usually timestamps are stored in two formats:
  - Human readable format: '2017-02-01 14:10:45 PST'
  - Unix timestamp: 1489792175 (time elapsed since 1970-01-01 00:00:00)
- ◆ Sometimes we need to 'slice / dice' timestamp
- ◆ Hypothesis: "On Facebook users click on photos more during lunch time"
  - Include 'time of day' on model calculation
- ◆ Some other interesting attributes:
  - 'time of day' (work hours, after hours)
  - 'day of week' (Mon / Tue / Wed ... work day / weekend)
- ◆ **Question for class:**
  - Any other time based behavior?

# FE: Incorporating Domain Knowledge

- ◆ Using knowledge acquired in the domain and add that to raw input data. **VERY IMPORTANT**
- ◆ Observation: Online shopping is usually higher on Fridays
  - Include an attribute: 'Day of week'
- ◆ Observation: Spam emails typically come from free email accounts
  - Add 'is free account' field to data
- ◆ **Question for the class:**
  - Can you think of any other examples?

# Lab 3.4: Exploratory Data Analysis (EDA)

- ◆ **Overview:**
  - Perform EDA on a dataset
- ◆ **Approximate Time:**
  - 10 – 15 mins
- ◆ **Instructions:**
  - **3.4: 'exploration/explore-house-sales' lab for Python / R / Spark**
- ◆ **To Instructor:**
  - Demo this lab on screen first, and explain the results

# BONUS Lab 3.5: Exploratory Data Analysis (EDA) 2: Graphing And Visualizing

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ **Overview:**
  - Visualize house-sales dataset
- ◆ **Approximate Time:**
  - 10 – 15 mins
- ◆ **Instructions:**
  - **3.5: 'exploration/visualize-house-sales' lab for Python / R / Spark**
- ◆ **To Instructor:**
  - Demo this lab on screen first, and explain the results



# Bonus Lab 4.1: Feature Engineering

- ◆ **Overview:**
  - Feature engineering exercises
- ◆ **Approximate Time:**
  - 15 – 20 mins
- ◆ **Instructions:**
  - 4.1 'feature-eng' lab for Python / R / Spark

# Developing A Model

Machine Learning Terminology  
Feature Engineering  
**Developing A Model**  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Sample Data Set: Cars

- ◆ We want to predict MPG of a car
- ◆ What attributes to consider?

	row.names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
2	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
6	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
7	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
8	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
9	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
10	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
11	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
12	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
13	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
14	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
15	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4

# Sample Model for Predicting MPG

Goal: Find  $f()$

$$Y = f(x_1, x_2, x_3) + E$$

	row.names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
2	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
4	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
5	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
6	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
7	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
8	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
9	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
10	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
11	Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
12	Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
13	Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
14	Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
15	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4

- ◆ Designate inputs as  $X$ 
  - $X_1$ : first input (e.g. number of cylinders)
  - $X_2$ : second input (e.g. weight of car)
  - $X_i$ :  $i$ th input (e.g. horsepower)
- ◆ Output / target variable is denoted as  $Y$
- ◆ Our Mathematical model is
  - $Y = f(X) + E$ 
    - $Y$ : Target
    - $X$ : Inputs ( $X_1, X_2$  ..etc)
    - $f$ : function
    - $E$ : error / noise

# Using the Model

- ◆ Our Model:
  - **MPG = f ( cylinders, horse power, weight) + E**
- ◆ **Goal: figure out 'f' from given data**
  - Once we figure out 'f' then we can plug-in values for *cylinders* , *horse power and weight* and *predict MPG*
  - Prediction is the most common use of ML
- ◆ **Inference**
  - Which of the 3 attributes (cylinders / horse power / weight) influences MPG more?
  - Figuring this out might give us insight into better products. *'what is the best way to get best MPG?'*

# Modeling Techniques (Little Math!)

- ◆ ML model has two types: Parametric / Non-Parametric
- ◆ Parametric models assume a strong 'f'
  - Tend to be simple models
  - May not be very accurate
- ◆ Non-parametric models don't assume a rigid 'f'
  - Adopt to data very well
  - More accurate
  - More difficult to understand than parametric models

# Parametric vs. Non Parametric

	Parametric	Non Parametric
Advantages	- Simpler - Very fast to learn from data - Don't required, a lot of training data	-Flexible: can adopt to complex data, -No assumptions about underlying function, -good prediction performance
Disadvantages	- limited by function - Can not adopt to complex data - Can underfit	-Complex to understand and explain, -Require more data for learning, -Slower to train as they have more parameters to tweak, - Canover-fit
Algorithms	- Linear Regression - Logistic Regression - Linear Discriminant Analysis	-Decision Trees, -Support Vector Machines, -Naïve Bayes
Best for	small size data with previous knowledge of features	when having lots of data and no prior knowledge of features



# Parametric vs. Non Parametric

	Parametric	Non Parametric
Model complexity	Simple	More complex
Training speed	Fast	Slow
Amount of training data	Doesn't need a lot	Needs more data
Explainability	Simple to explain	Harder to explain
Fit	under-fit	over-fit
Adopting to data	simple data	complex data
Prediction accuracy	good	better

# Evaluating A Model

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
**Evaluating A Model**  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Data Science Methodology: Iterative Learning Process

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

## 1 - Prep

- Collect Data
- Clean Data
- Process Data



## 2 - Model Building & Improvement

- Explore Data
- Feature Extraction
- Choose Algorithm
- Build model



## 5 - Deploy Model

- Production
- Visualizations



## 3 - Model Evaluation & Testing

- Estimate error
- Calculate precision / recall



## 4 - Optimize Model

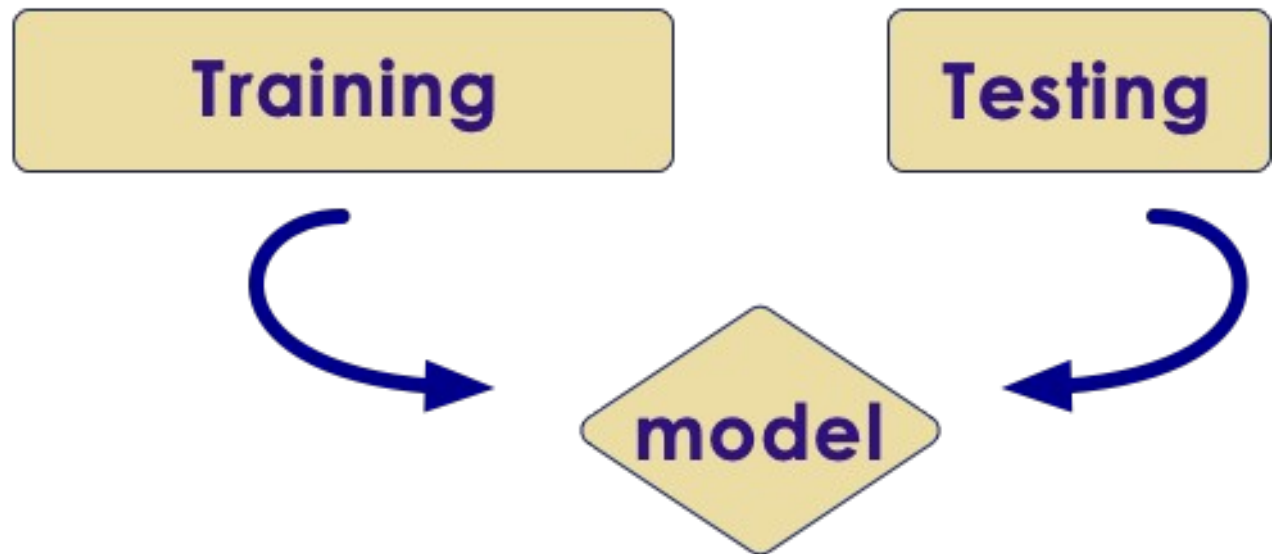
- More training data
- Tweak Model parameters



# Evaluating A Model

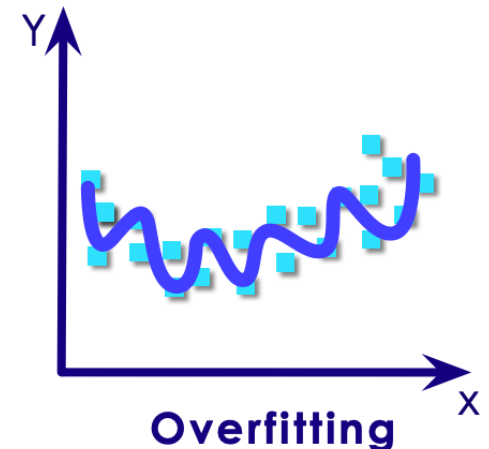
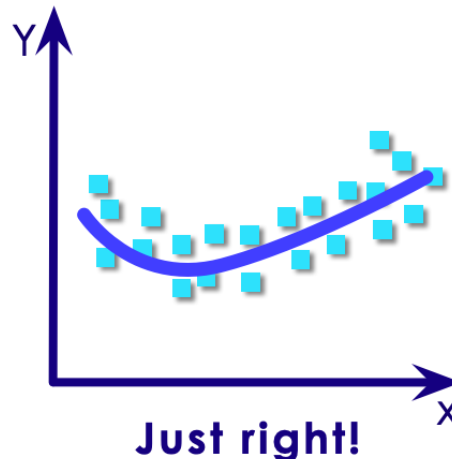
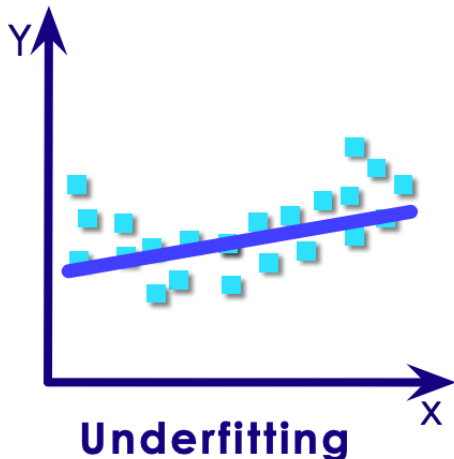
Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ How do we know our model is 'good'?
- ◆ One way to measure the performance, is how well it is predicting on 'new data'
  - Model is trained with 'training data'
  - Measure its performance on 'test data' (the model hasn't seen 'test data')



# Under-fitting / Over-fitting

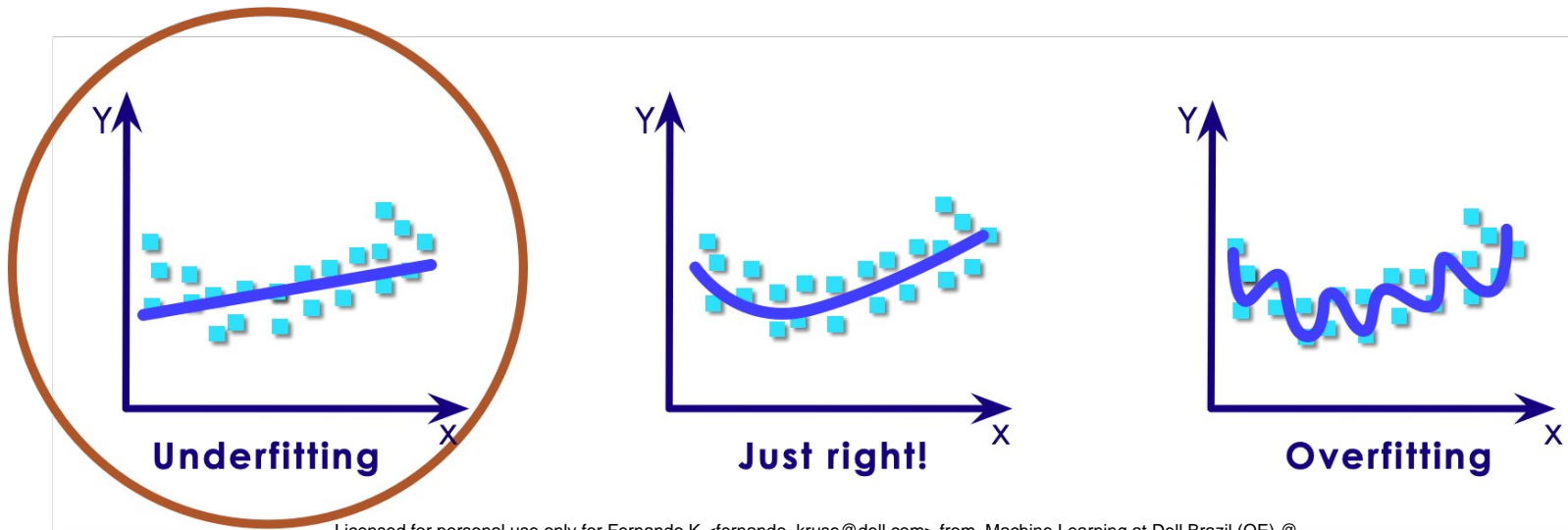
- ◆ Here we have 3 models
- ◆ One on left: is not really capturing the essence of the data
  - Underfitting
- ◆ One on right: following every small variation of data, not really generalizing
  - Overfitting
- ◆ One in the middle is just right



# Under-fitting

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ Model is 'too simple' to capture the trends in input data
- ◆ How to detect under-fitting?
  - We will get poor performance in both training & testing data
  - E.g.:
    - Training accuracy : 45%
    - Testing accuracy : 42%
- ◆ Resolution:
  - Try a different algorithm / model, that better fits the data



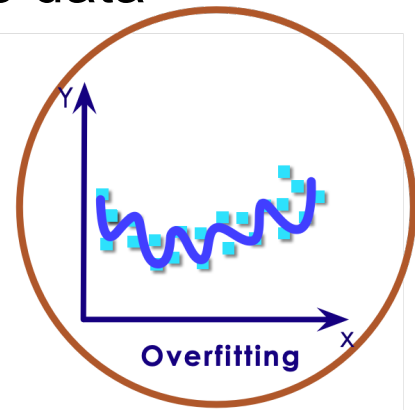
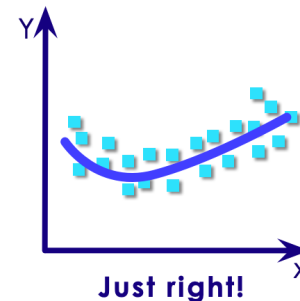
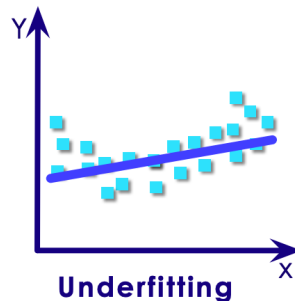
Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

# Over-fitting

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ Model is 'too complex' that is 'memorizing' training data, but not '**generalizing**' for new data
- ◆ How to detect over-fitting?
  - Excellent performance on training data, but poor performance on testing (new) data
  - E.g.:
    - Training accuracy : 95%
    - Testing accuracy : 62%
- ◆ Resolution:
  - Try a different algorithm / model, that better fits the data
  - Simplify inputs

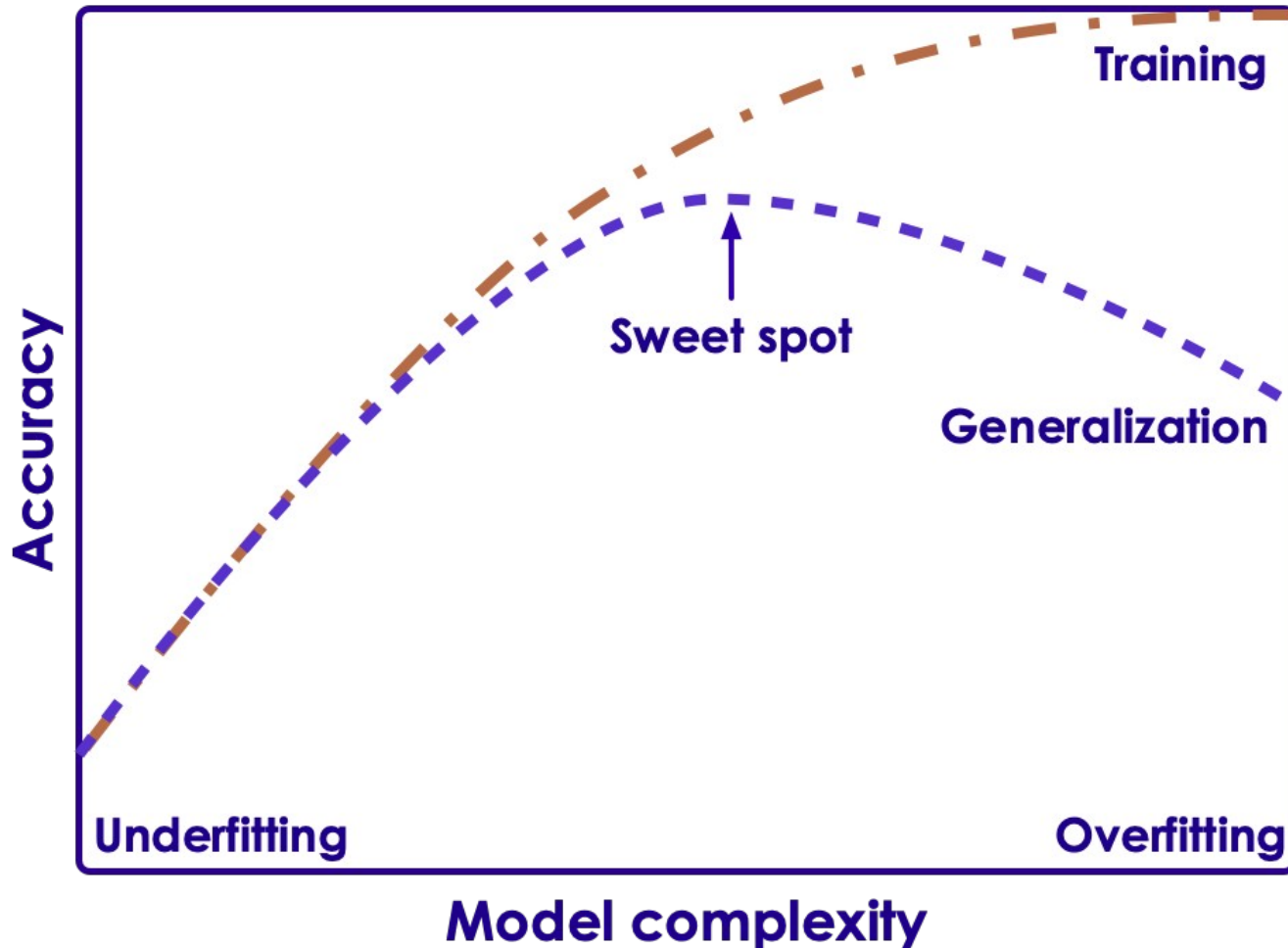


Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

# Achieving a Good Fit

- ◆ In ML we strive to find the 'sweet spot' between under-fitting models and over-fitting models





# Achieving a Good Fit

- ◆ Both overfitting and underfitting can lead to poor model performance
- ◆ underfitting is easier to spot
  - Bad performance on training data
  - Bad performance on test data
- ◆ Overfitting can be hard to spot
  - because it performs well on training data
  - But doesn't do well on 'unseen' test data
- ◆ Avoiding overfitting
  - Resampling technique
  - Hold back a validation dataset
  - Most popular method is: k-fold validation (more on this later)
- ◆ For **decision trees** , we **prune** the tree to limit overfitting

# Cross Validation

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
**Cross Validation**  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Common Mistakes in Model Validation

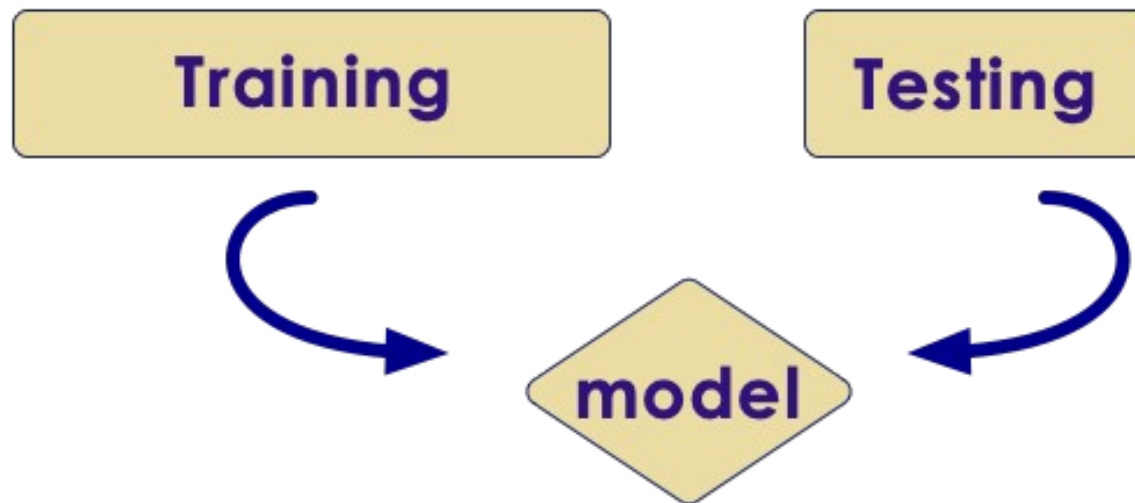
- ◆ **Mistake: Re-using 'training data' as 'testing data'**
  - Model can predict accurately on testing (because it has 'seen' the data before)
  - Giving the designer 'false confidence'
  - But will do badly on new data
- ◆ **Solution:**
  - **Cross Validation**

# Common Mistakes in Model Validation

- ◆ **Mistake: Using overly simplistic validation methods**
  - Model might appear to be working well
- ◆ **Mistake: Using more parameters (20 attributes) than actual observations (10 observations)**
  - Model will 'memorize' the data rather than learning
  - Will do well on testing data / but poorly on new data
- ◆ **Solution:**
  - Use more data

# Hold Out Method (Validation Set)

- ◆ Do not use the same data for training and testing!
  - Model will do well in testing (it has seen the questions before!)
- ◆ Separate the data set into
  - Training set (60-70%)
  - Testing set (30-40%)
- ◆ This is done randomly



# Hold Out Method Drawbacks

## ◆ Drawbacks

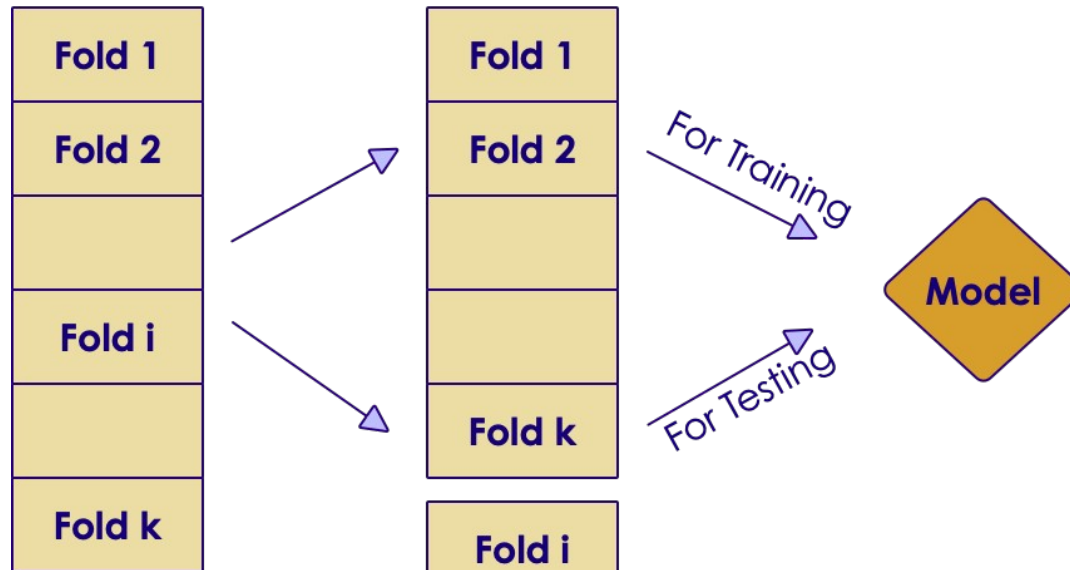
- Error rate can significantly fluctuate based on how data is divided (randomly)
- When we do a randomly split data into training/test
- If we are 'lucky', we get an easy test set -> resulting in higher than usual accuracy
  - If we are 'unlucky' we get a hard test set -> resulting in lower than usual accuracy

# Hold Out Method Drawbacks

- ◆ Example
  - Assume we want to test a student's knowledge in a subject
  - We have a pool of 20 questions
  - Test 1:
    - Out of 20, we randomly choose 15 questions And the student scores 60%
    - Is this the final score? No.
    - We need to do more tests and average out the score
- ◆ Solution: k-fold Cross validation
  - Rigorously tests model accuracy

# K-Fold Cross Validation

- ◆ Divide the data into equal k sections (k folds, usually 10 to 20)
- ◆ Reserve one fold for testing (say fold-i)
- ◆ Use others folds for training
- ◆ Then test with fold-i
- ◆ After we have cycled through all k folds, prediction accuracies are aggregated and compared





# Cross-Validation Example

Runs	Splits					Model	Accuracy
Run 1	Test	Train	Train	Train	Train	Model 1	80%
Run 2	Train	Test	Train	Train	Train	Model 2	84%
Run 3	Train	Train	Test	Train	Train	Model 3	90%
Run 4	Train	Train	Train	Test	Train	Model 4	86%
Run 5	Train	Train	Train	Train	Test	Model 5	82%

- ◆ Here we are doing a 5-fold cross validation
- ◆ Data is split into 5 splits – one held for testing, remaining 4 used for training
- ◆ Accuracy varies from 80% to 90%
  - If the accuracy range is big (say from 40% to 80%) then it might be an indication of high-variance (sensitive to input data)
- ◆ Average accuracy is  $\text{AVG}(80, 84, 90, 86, 82) = 85\%$

# Cross Validation Takeaways

- ◆ We don't choose the 'best performing model' from CV
  - CV is used to understand a particular algorithm's performance for the given data
  - And how well it can generalize to new data
- ◆ Pros
  - Helps us systematically tests a model through the data
  - Can identify high-variance / over-fitting models
- ◆ Cons
  - Increased compute time to create multiple models and test
    - Solution: run CV tasks in parallel across multiple CPU-cores or on a cluster (embarrassingly parallelizable problem)

# Bootstrapping

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
**Bootstrapping**  
Errors / Residuals  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

- ◆ Randomly selecting data for training with replacement
- ◆ It may seem counter-intuitive to draw the same data again and again
- ◆ But in some scenarios, bootstrapping really helps to train the model
- ◆ See next slides to understand sampling with and without replacement

# Sampling Without Replacement

Original Dataset

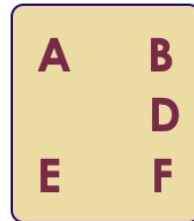
Sample Dataset



select  
one (C) →



select  
another (D) →



select  
another (A) →



**Sampling  
Without  
Replacement**

# Sampling With Replacement (aka Bootstrapping)

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @ 2019-03-12

Original Dataset

Sample Dataset

A	B
C	D
E	F

select  
one (C) →

C
---

A	B
E	F

← Put (C) back

A	B
C	D
E	F

select  
another (D) →

C	D
---	---

**Sampling  
Without  
Replacement**

A	B
C	
E	F

← Put (D) back

A	B
C	D
E	F

select  
another (C) →

C	C
D	

(C) is selected  
again

# Bootstrapping Example 1

- ◆ Data points :
- ◆ Bootstrap selection 1 :
- ◆ Bootstrap selection 2 :

# Bootstrapping Example 2

Original Dataset

$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
-------	-------	-------	-------	-------	-------	-------	-------	-------	----------

Bootstrap 1

$X_8$	$X_6$	$X_2$	$X_9$	$X_5$	$X_8$	$X_1$	$X_4$	$X_8$	$X_2$
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

$X_3$	$X_7$	$X_{10}$
-------	-------	----------

Bootstrap 2

$X_{10}$	$X_1$	$X_3$	$X_5$	$X_1$	$X_7$	$X_4$	$X_2$	$X_1$	$X_8$
----------	-------	-------	-------	-------	-------	-------	-------	-------	-------

$X_6$	$X_9$
-------	-------

Bootstrap 3

$X_6$	$X_5$	$X_4$	$X_1$	$X_2$	$X_4$	$X_2$	$X_6$	$X_9$	$X_2$
-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

$X_3$	$X_7$	$X_8$	$X_{10}$
-------	-------	-------	----------

Training Sets

Test Sets

no overlap between  
testing & training sets



# Errors / Residuals

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
**Errors / Residuals**  
Confusion Matrix and ROC curve  
Bias Variance Tradeoff

# Problem : Estimating Tips for Meals

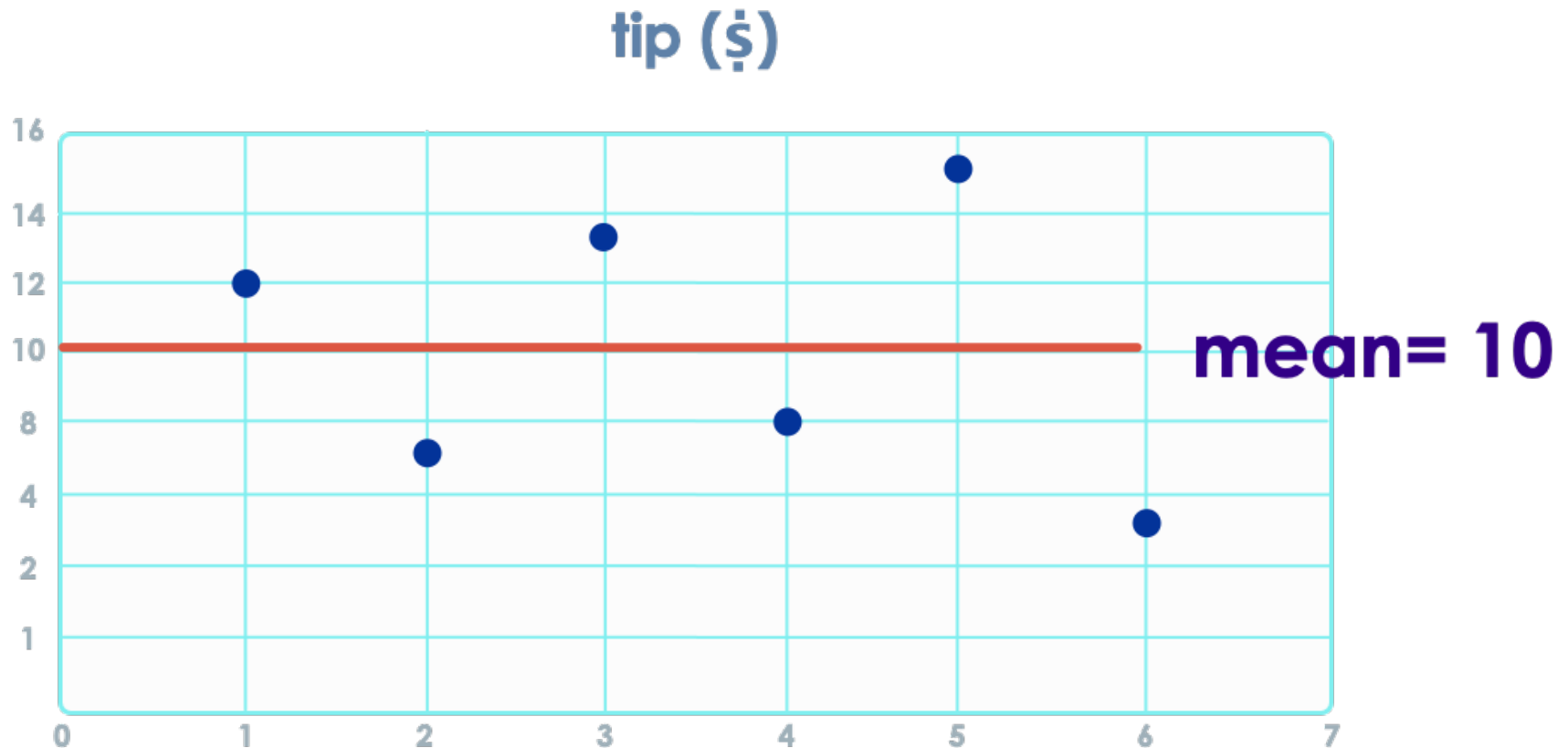
- ◆ A waiter at a restaurant is collecting data on tips
- ◆ Usually tips correspond to total of the bill.
- ◆ Higher the bill -> higher the tip
- ◆ But in this case, bill total is missing, we only have tip amounts
- ◆ How can we predict the tip amount for next meal?

Meal #	Tip (\$)
1	12
2	7
3	13
4	8
5	15
7	5

# Tip Calculation

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ We can calculate average tip calculating AVERAGE of all the tip amounts
- ◆ Next tip = AVERAGE(all tips) = MEAN (all tips) = \$10

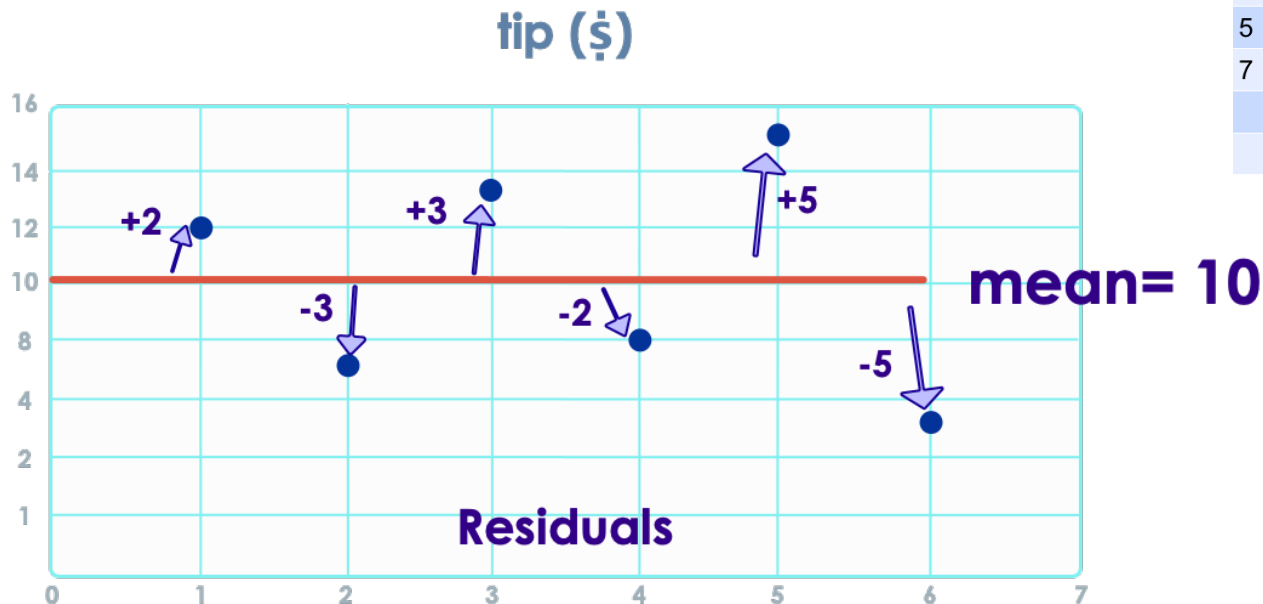


# Residuals / Errors

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ None of the tip amounts are exactly \$10
- ◆ \$10 is just an average of all tip amounts
- ◆ **Residual** = difference between actual tip and predicted tip
- ◆ Sum of all residuals = **ZERO**

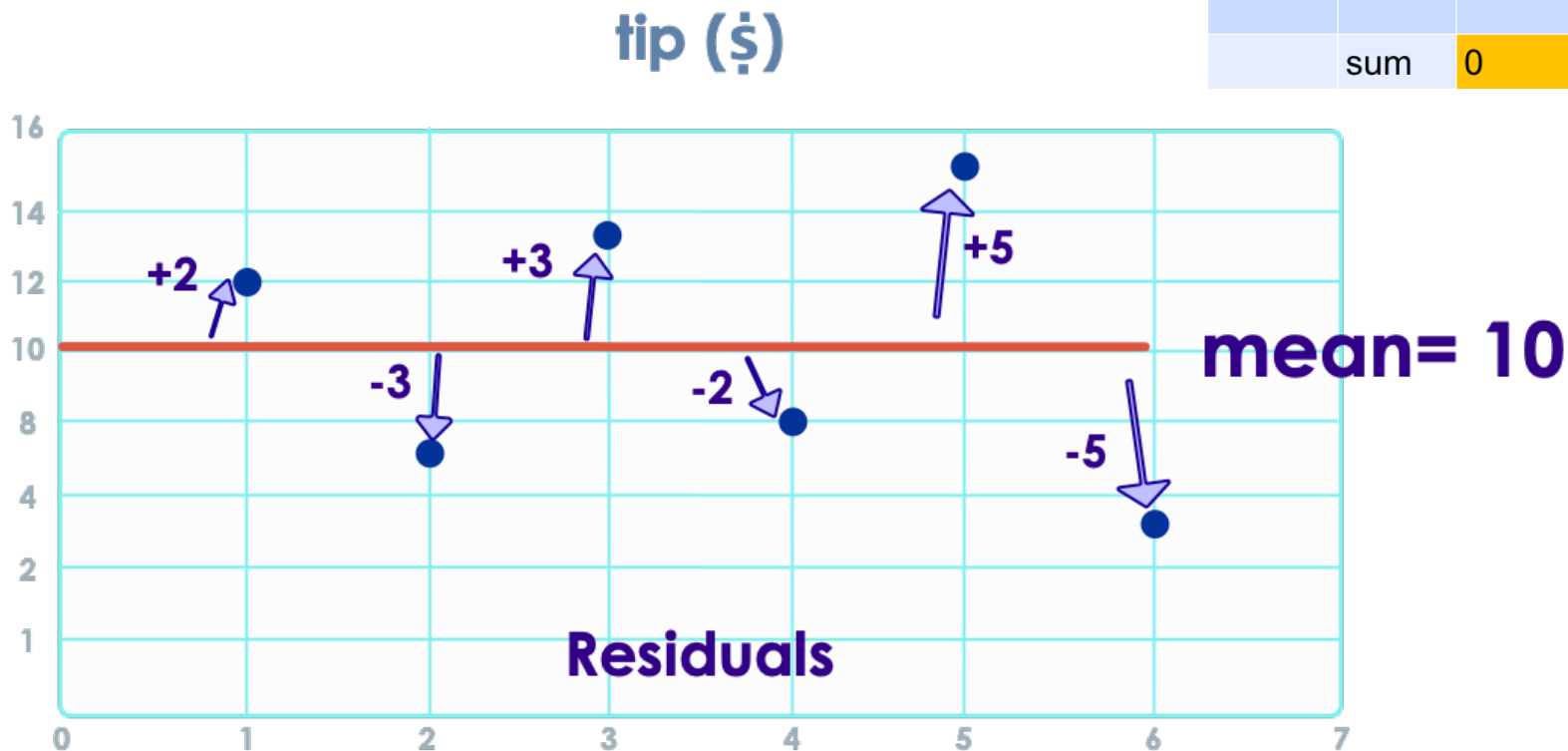
Meal #	Tip (\$)	Residual
1	12	+2
2	7	-3
3	13	+3
4	8	-2
5	15	+5
7	5	-5
	sum	0



# Sum of Squared Errors (SSE)

- ◆ Squaring residuals
  - To make the positive
  - To amplify 'outliers' (large deviations)
- ◆ Goal is to minimize SSE
  - Minimize errors

Meal #	Tip(\$)	Residual	Residual <sup>2</sup>
1	12	+2	4
2	7	-3	9
3	13	+3	9
4	8	-2	4
5	15	+5	25
7	5	-5	25
	sum	0	76



# Sum of Squared Errors (SSE)

- ◆ Also known as
  - Sum of Squared Residuals (SSR)
- ◆ In Regressions RSS/SSE is a measure used to select / optimize models
- ◆ Lower RSS indicates a tighter model fit to data
- ◆ Example:
  - If model-A yields  $RSS = 70$
  - And model-B yields  $RSS=50$
  - Model-B might be better fit
- ◆ In this formula
  - $Y_i$ : actual value
  - $\hat{Y}_i$ : predicted value

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

# Confusion Matrix and ROC curve

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
**Confusion Matrix and ROC curve**  
Bias Variance Tradeoff

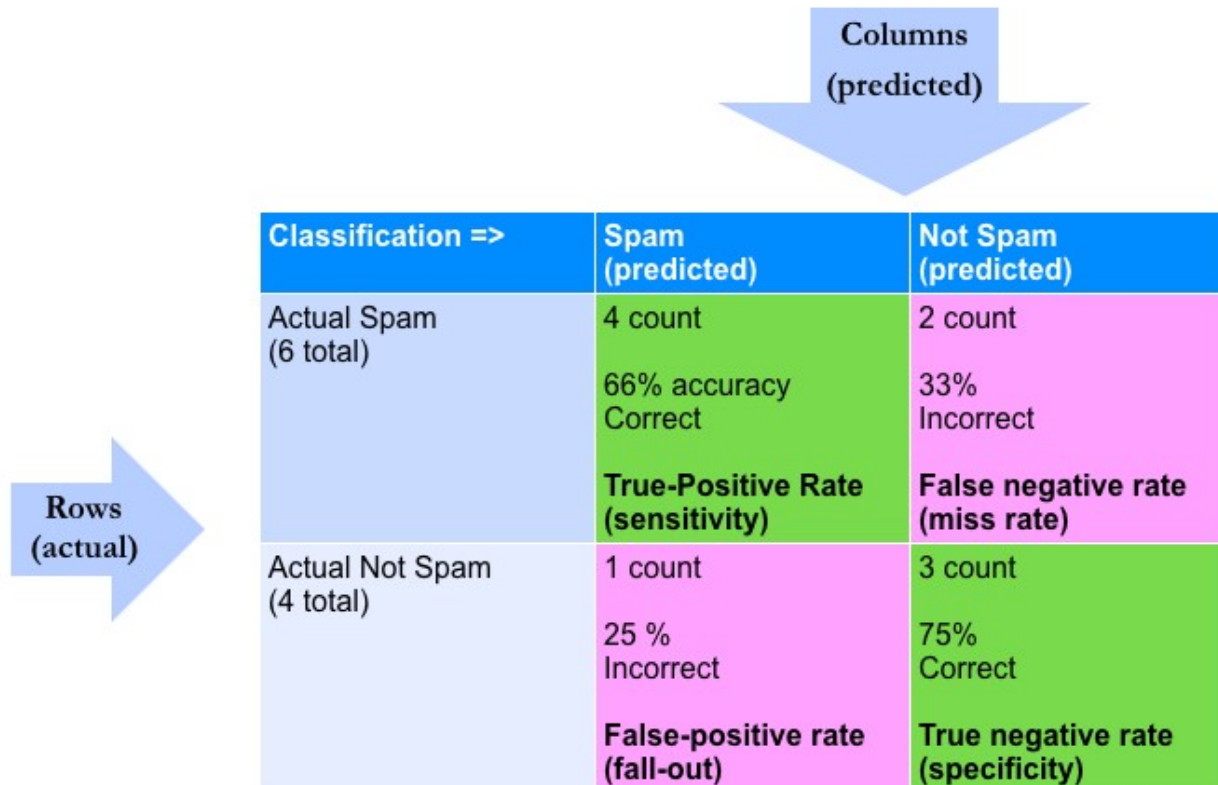
# Evaluating Classification Models

- ◆ Let's consider a binary classifier
  - Picks one of two outcomes (spam / not-spam)
- ◆ Two approaches
  - Confusion matrix
  - ROC curve



# Confusion Matrix / Error Matrix

- ◆ Let's consider a binary classifier
  - Picks one of two outcomes (spam / not-spam)
- ◆ Say we are classifying 10 emails (6 spam, 4 not-spam)



Classification =>		Columns (predicted)	
		Spam (predicted)	Not Spam (predicted)
Rows (actual)	Actual Spam (6 total)	4 count  66% accuracy Correct  <b>True-Positive Rate (sensitivity)</b>	2 count  33% Incorrect  <b>False negative rate (miss rate)</b>
	Actual Not Spam (4 total)	1 count  25 % Incorrect  <b>False-positive rate (fall-out)</b>	3 count  75% Correct  <b>True negative rate (specificity)</b>

# Confusion Matrix: More Than 2 Outcomes

		Predicted		
		Cat	Dog	Rabbit
Actual	Cat (8)	5	3	0
	Dog (6)	2	3	1
	Rabbit (13)	0	2	11

- ◆ Which animal the algorithm has trouble classifying? (too many misclassifications)
- ◆ Which animal the algorithm is good at classifying?

# Interpreting Confusion Matrix (True/False Positives/Negatives)

Licensed for personal use only for Fernando K <fernando\_kruse@ dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (a cancer diagnostic)	<b>Positive (has cancer)</b>	<b>True positive</b>  - Patients who have cancer are correctly identified	<b>False negative</b>  <u>Miss rate</u> - A cancer patient is missed - Guilty prisoner was not convicted
	<b>Negative (doesn't have cancer)</b>	<b>False Positive</b>  <u>Sensitivity</u> - A healthy patient is flagged incorrectly - False alarm - 'Crying wolf' - Hiring someone who is not qualified	<b>True negative</b>  - Patients who do not have cancer are correctly identified

Licensed for personal use only for Fernando K <fernando\_kruse@ dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

# Confusion Matrix: Accuracy / Error Rate

- ◆ Accuracy
  - Overall how accurate is the model? =  $(TP + TN) / \text{total} = (90 + 70) / 200 = 0.8$  or 80%
- ◆ Misclassifications / Error rate
  - How wrong is the model? =  $(FP + FN) / \text{total} = (10 + 30) / 200 = 0.2$  or 20% =  $1 - \text{accuracy}$

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (n = 200)	Positive (n = 120)	True positive (n = 90)	False negative (n = 30)
	Negative (n = 80)	False Positive (n = 10)	True negative (n = 70)

# Confusion Matrix: Accuracy May Not Be Enough

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

- ◆ Let's say our classifier is used to diagnose cancer patients.
- ◆ We have total 100 patients
  - 98 healthy
  - 2 have cancer
- ◆  $\text{Accuracy} = (\text{TP} + \text{TN}) / \text{total} = (1 + 98) / 100 = 99\%$  (very good!)
- ◆  $\text{Misclassifications} / \text{Error rate} = (\text{FP} + \text{FN}) / \text{total} = (0 + 1) / 100 = 1\%$  ( $1 - \text{accuracy}$ )

# Confusion Matrix: Accuracy May Not Be Enough

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ **Question for class:**
- ◆ What is the implication of 'False Positive'?
- ◆ What is the implication of 'False Negative' ?
- ◆ Which is more serious?

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (n = 100)	Positive (n = 2)	True positive (n = 1)	False negative (n = 1)
	Negative (n = 98)	False Positive (n = 0)	True negative (n = 98)

- ◆ Since accuracy may not be enough of a metric, there are other metrics
  - Precision
  - Recall
- ◆ Next few slides will explain these

# Confusion Matrix: TPR / FPR

- ◆ **True Positive Rate (TPR) /Sensitivity / Hit Rate / Recall**  
How often model predicts 'positive' as 'positive' (correctly) ? =  $TP / (TP + FN)$  actual Positive=  $90 / 120 = 0.75$  or 75%
- ◆ **False Positive Rate (FPR)** How often model predicts 'negative' as 'positive' (incorrectly)=  $FP / (FP + TN)$  actual negative=  $10 / 80 = 0.125$  or 12.5%

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (n = 200)	Positive (n = 120)	True positive (n = 90)  TPR = 75%	False negative (n = 30)
	Negative (n = 80)	False Positive (n = 10)  FPR = 12.5%	True negative (n = 70)



# Confusion Matrix: Specificity / Precision / Prevalence

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ **Specificity** How often model predicts negative' as negative' (correctly)?=  $TN / (TN + FP)$  actual no=  $70 / (70 + 10) = 0.875$  or 87.5 % =  $1 - FPR$
- ◆ **Precision / Positive Predictive Value (PPV)** When model predicts 'positive' how often it is right?=  $TP / (TP + FP) = \text{true} / \text{predicted positive} = 90 / (90 + 10) = 0.9$  or 90%

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (n = 200)	Positive (n = 120)	True positive (n = 90)  TPR = 75%	False negative (n = 30)
	Negative (n = 80)	False Positive (n = 10)  FPR = 12.5%	True negative (n = 70)  Specificity = 87.5%

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

# Confusion Matrix: PPV / Null Error Rate

- ◆ **Prevalence** How often does 'positive' occurs in our sample=  
actual positive / total=  $120 / 200 = 0.6$  or 60%
- ◆ **Null Error Rate** How often would the model be wrong if it  
always predicted the majority class?
- ◆ Here our majority = Positive
- ◆ If we always predicted 'positive' we would be wrong 80 times  
(negative)
- ◆  $80/200 = 40\%$

		Predicted Condition	
		Predicted Positive	Predicted Negative
Actual condition (n = 200)	Positive (n = 120)	True positive (n = 90)  TPR = 75%	False negative (n = 30)
	Negative (n = 80)	False Positive (n = 10)  FPR = 12.5%	True negative (n = 70)  Specificity = 87.5%

# Confusion Matrix : F-Score

- ◆ So, while precision and recall are very important measures, looking at only one of them will not provide us with the full picture.
- ◆ One way to summarize them is the f-score or f-measure, which is with the harmonic mean of precision and recall
- ◆  **$F = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$**

# How is ROC Curve Generated

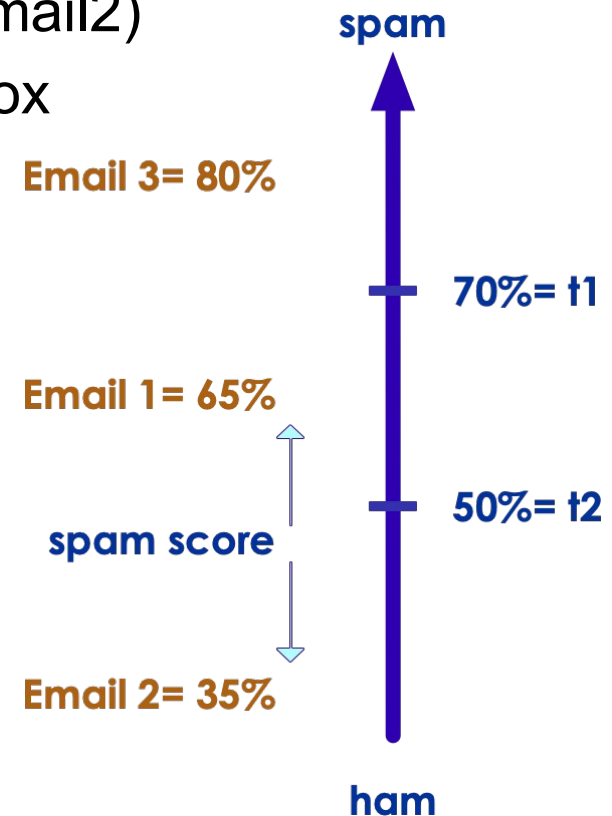
- ◆ Y-axis: True Positive Rate (TPR)
  - Actual=positive, predicted=positive
  - Correct!
- ◆ X-axis: False Positive Rate (FPR)
  - Actual=negative, predicted=positive
  - Incorrect!
- ◆  $0.0 \leq \text{TPR} \ \& \ \text{FPR} \leq 1.0$
- ◆ Plot TPR / FPR while varying 'threshold'

- ◆ Our spam classifier provides a 'spam probability' for each email
  - Probability is between 0.0. and 1.0 (or 0 to 100%)
  - 1.0 definitely spam
  - 0.0 definitely not spam
- ◆ When an email's 'spam score' is above a certain number we mark it as spam
- ◆ This is called 'threshold'

# Threshold

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ If spam threshold is lower (say 50%)
  - more emails will be classified as spam (email2, email3)
  - Users will miss emails (as they are in Spam folder)
- ◆ If spam threshold is higher (70%)
  - Fewer emails will be classified as spam (email2)
  - Users will see more spam emails be in Inbox
- ◆ We need to find the sweet spot!



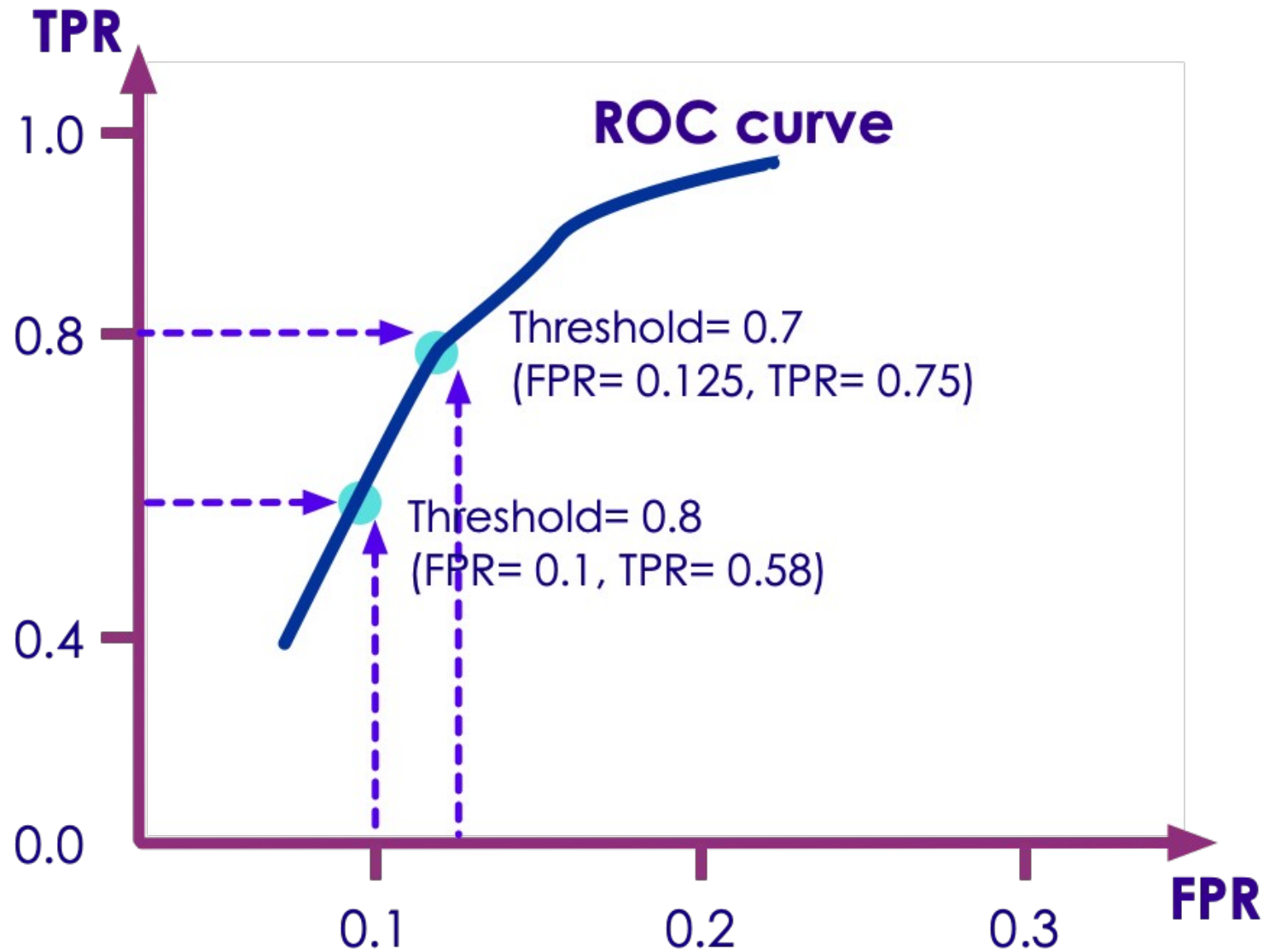
# Threshold

- ◆ In first table our threshold is 0.7
  - 90 emails are correctly predicted as spam
- ◆ Next table, our threshold is higher 0.8
- ◆ Only 70 emails are classified as spam
  - Lower TPR

		Predicted Condition	
Threshold = 0.8 (higher)		Predicted Spam	Predicted Not Spam
Actual condition (total = 200)	Spam (n = 120)	True positive (n = 70) TPR = TP / positive = 70 / 120 = 58.33%	False negative (n = 50)
	Not Spam (n = 80)	False Positive (n = 8) FPR = FP / negative = 8 / 80 = 10%	True negative (n = 72)

		Predicted Condition	
Threshold = 0.7		Predicted Spam	Predicted Not Spam
Actual condition (total = 200)	Spam (n = 120)	True positive (n = 90) TPR = TP / positive = 90/120 = <b>75%</b>	False negative (n = 30)
	Not Spam (n = 80)	False Positive (n = 10) FPR = FP / negative = 10/80 = <b>12.5%</b>	True negative (n = 70)

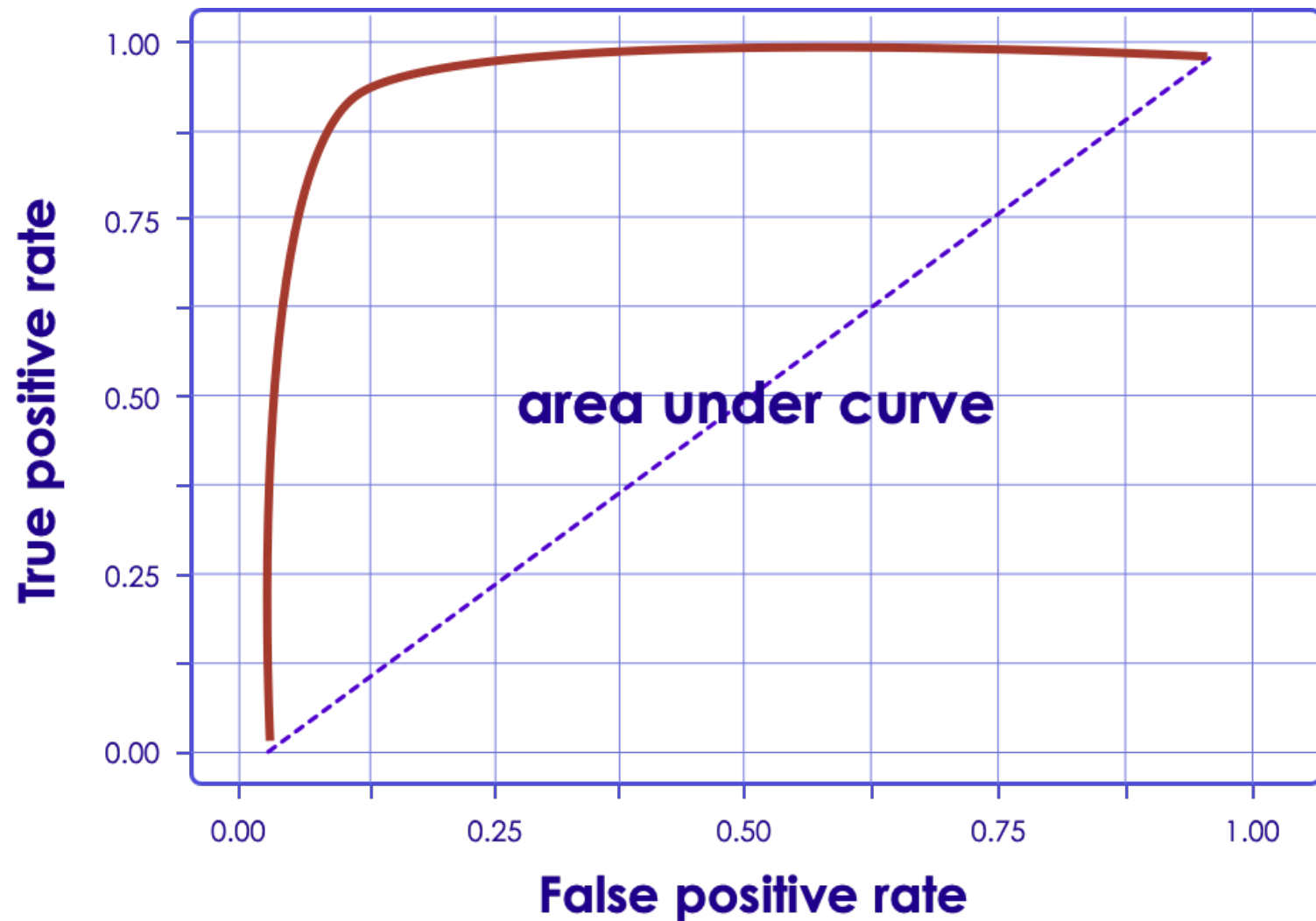
# ROC Curve: Receiver Operating Characteristic





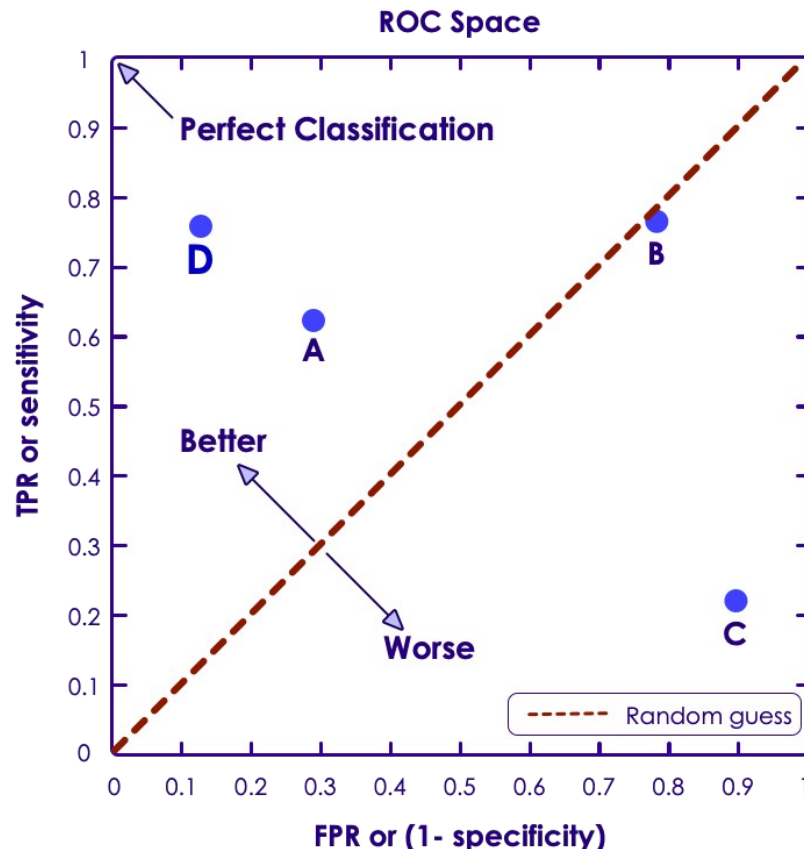
# ROC Curve Example

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12



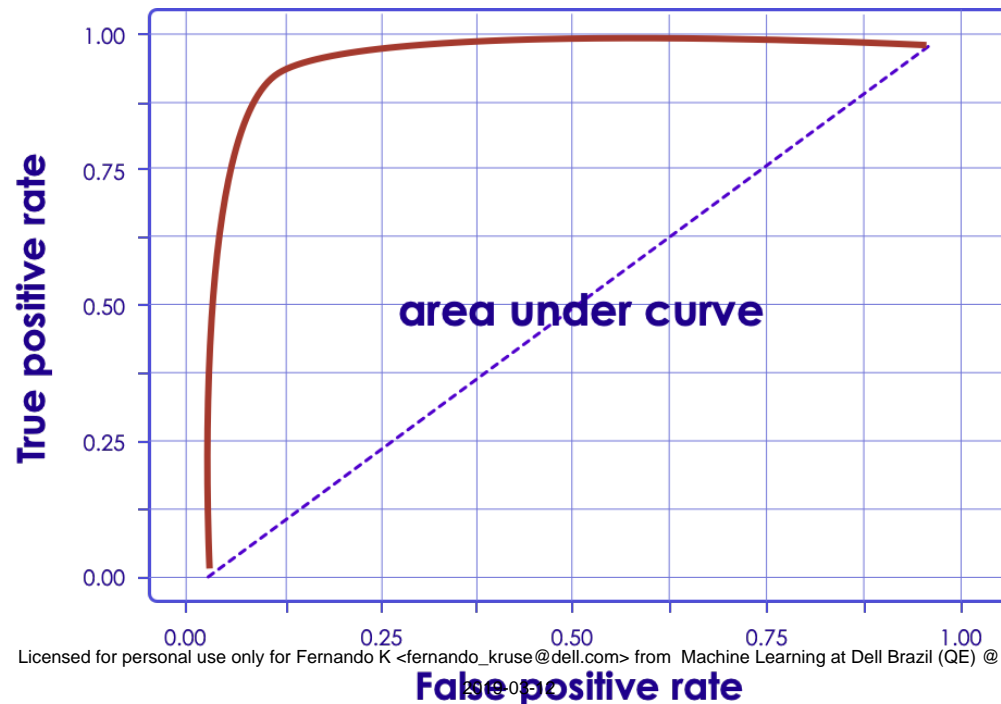
# Interpreting ROC Curve

- ◆ The red line plots 'random guess' = B
- ◆ Approaching 'top left' corner would be a perfect classifier! So D is better A
- ◆ C performs worse than random --> bad



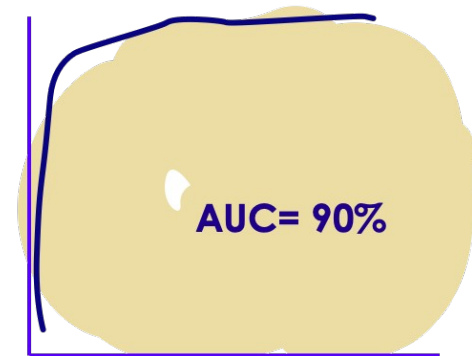
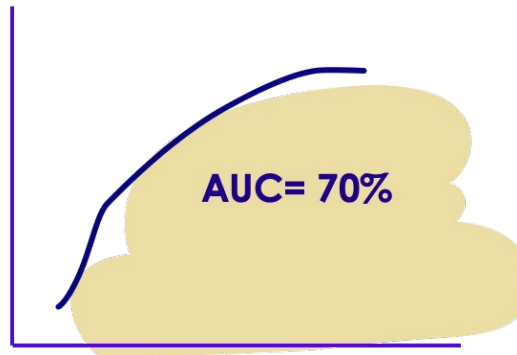
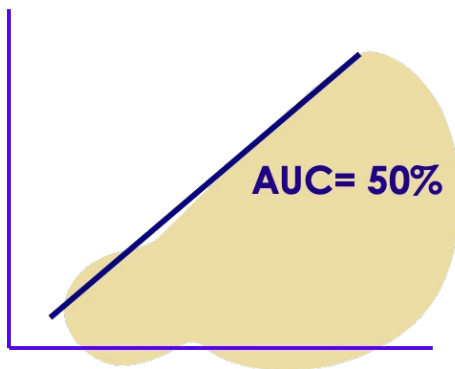
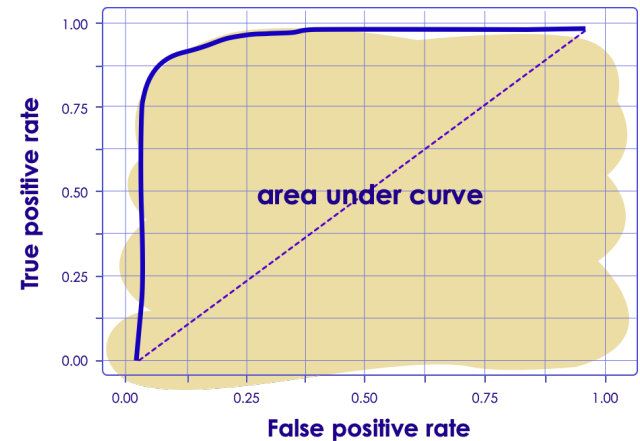
# Interpreting ROC Curve

- ◆ Shows tradeoff of TPR (sensitivity) vs. FPR (1 – specificity)
- ◆ The closer to top-left , the more accurate the model
- ◆ Upper left corner (0,1) = perfect classification!
- ◆ The closer to middle line (45 degree) the less accurate the test
  - Middle line represents: random classification (50%)



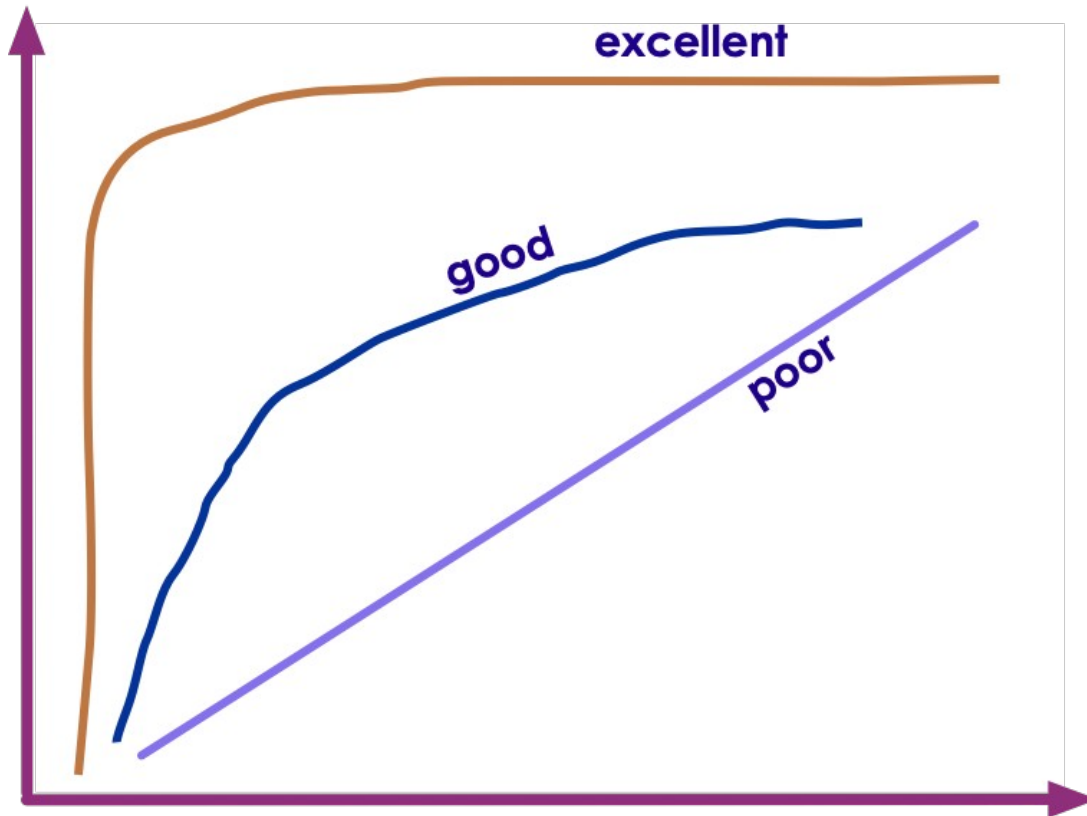
# Area Under Curve – AUC (ROC Space)

- ◆ Measures the percentage of area 'under the curve'
- ◆ AUC is between 0 and 1
- ◆ Higher AUC --> more accurate the model
- ◆ See 3 scenarios below
  - Leftmost is bad (50%)
  - Middle: OK (70%)
  - Rightmost: very good (90%)



# Using AUC to Measure Accuracy

- ◆ Accuracy can be specified using a grading system

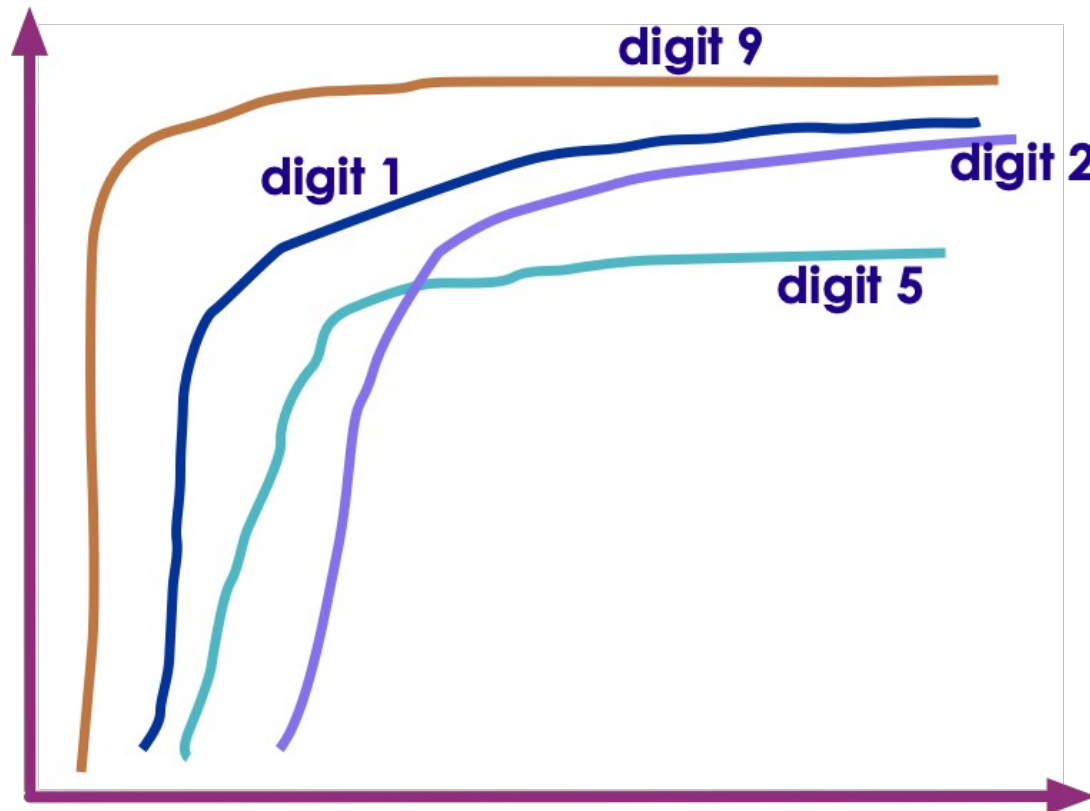


AUC	Grade
0.9 – 1.00	A - Excellent
0.80 – 0.90	B - good
0.70 - 0.80	C - fair
0.60 – 0.70	D - poor
0.50 – 0.60	F - Fail

# ROC / AUC For Multiclass Classifiers

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ Say our algorithm recognizes hand-written digits (postal code) into numbers.
- ◆ Its ROC can be drawn as follows



Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

# Bias Variance Tradeoff

Machine Learning Terminology  
Feature Engineering  
Developing A Model  
Evaluating A Model  
Cross Validation  
Bootstrapping  
Errors / Residuals  
Confusion Matrix and ROC curve  
**Bias Variance Tradeoff**

# Estimating Target Function

In supervised algorithms try to estimate target function

- ◆ 'f'  $Y = f(X)$  ` `Y: output, X: input
- ◆ The error can be broken down to
  - Bias error
  - Variance error
  - Irreducible error
- ◆ Irreducible error can not be minimized. May be caused by unknown variables, noise ..etc



- ◆ Bias are the **simplifying assumptions** made by a model to make target function easier to learn
- ◆ Generally **parametric algorithms have high bias**
  - Fast learning
  - Easier to understand
  - But less flexible
  - Lower predicting performance on complex data (That doesn't fit the simplifying assumptions of algorithm)
- ◆ **Low Bias (good!):**
  - Less assumptions made about target function
  - E.g. Decision Trees, KNN, SVM
- ◆ **High Bias (not good):**
  - more assumptions of target function
  - E.g. Linear regression, Logistic regression

- ◆ Target function is estimated from training data
  - So it is influenced by training data
- ◆ **Variance is the amount that the estimate of the target function will change if different training data was used**
- ◆ Ideally target function should not change drastically from one training set to next
  - meaning that the algorithm is good at picking out the hidden underlying mapping between the inputs and the output variables

## ◆ Low Variance:

- Suggests small changes to the estimate of the target function with changes to the training dataset
- E.g. parametric algorithms: Linear Regression, Logistic Regression

## ◆ High Variance:

- Suggests large changes to the estimate of the target function with changes to the training dataset.
- Generally nonparametric machine learning algorithms that have a lot of flexibility have a high variance
- E.g. decision trees have a high variance, that is even higher if the trees are not pruned before use

# Bias - Variance Tradeoff

- ◆ Goal of supervised algorithm is to achieve **low bias and low variance**
- ◆ Low bias: less assumptions of target function form --> more flexibility
- ◆ Low variance: less swings in target function for changes in training data --> stable algorithm
- ◆ Parametric or linear machine learning algorithms often have a high bias but a low variance

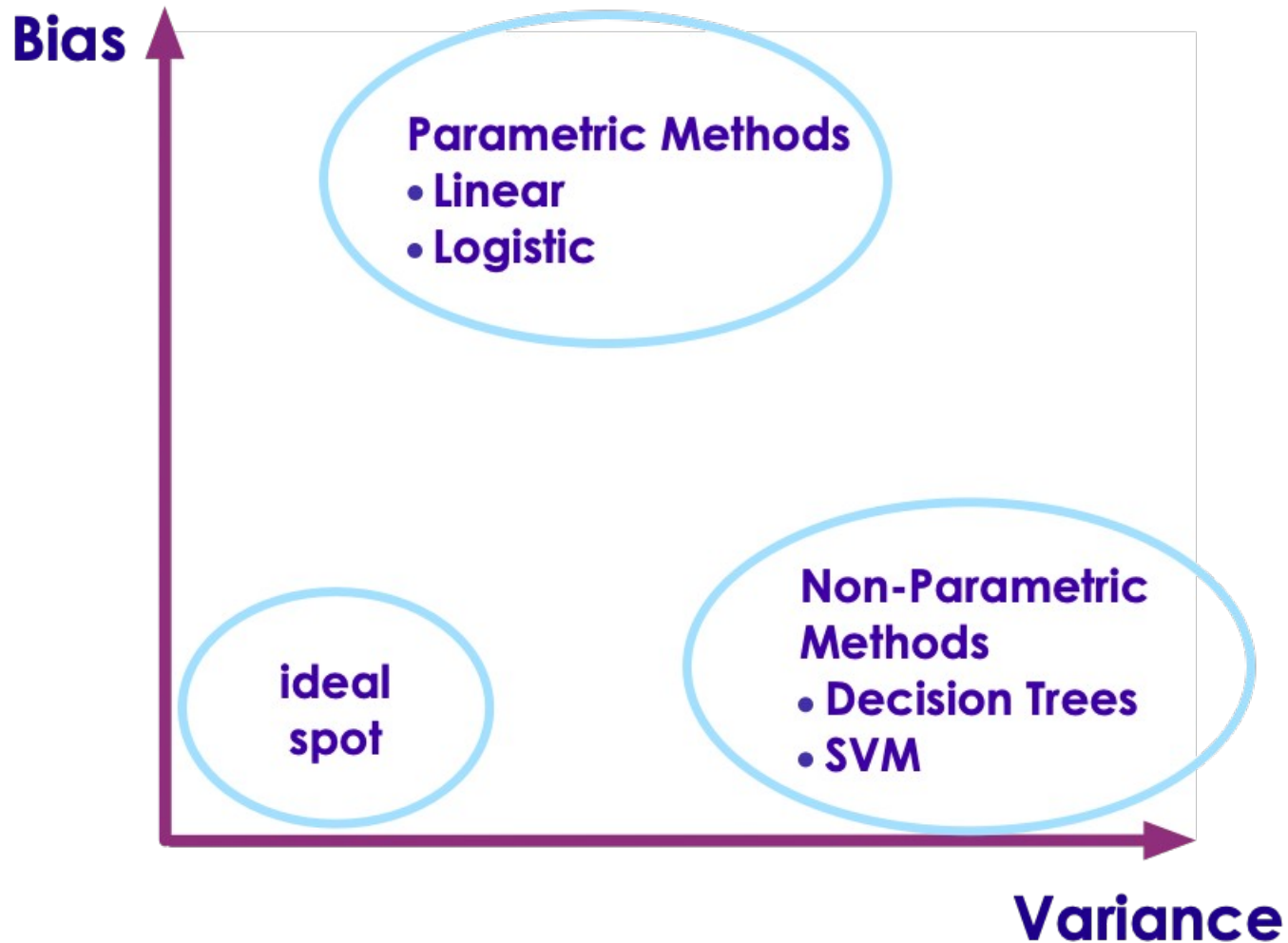
# Bias - Variance Tradeoff

- ◆ Nonparametric or nonlinear machine learning algorithms often have a low bias but a high variance
- ◆ There is no escaping the relationship between bias and variance in machine learning
  - Increasing the bias will decrease the variance
  - Increasing the variance will decrease the bias
- ◆ Bias-Variance can be adjusted for particular algorithms

# Bias Variance Trade Off

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12



Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

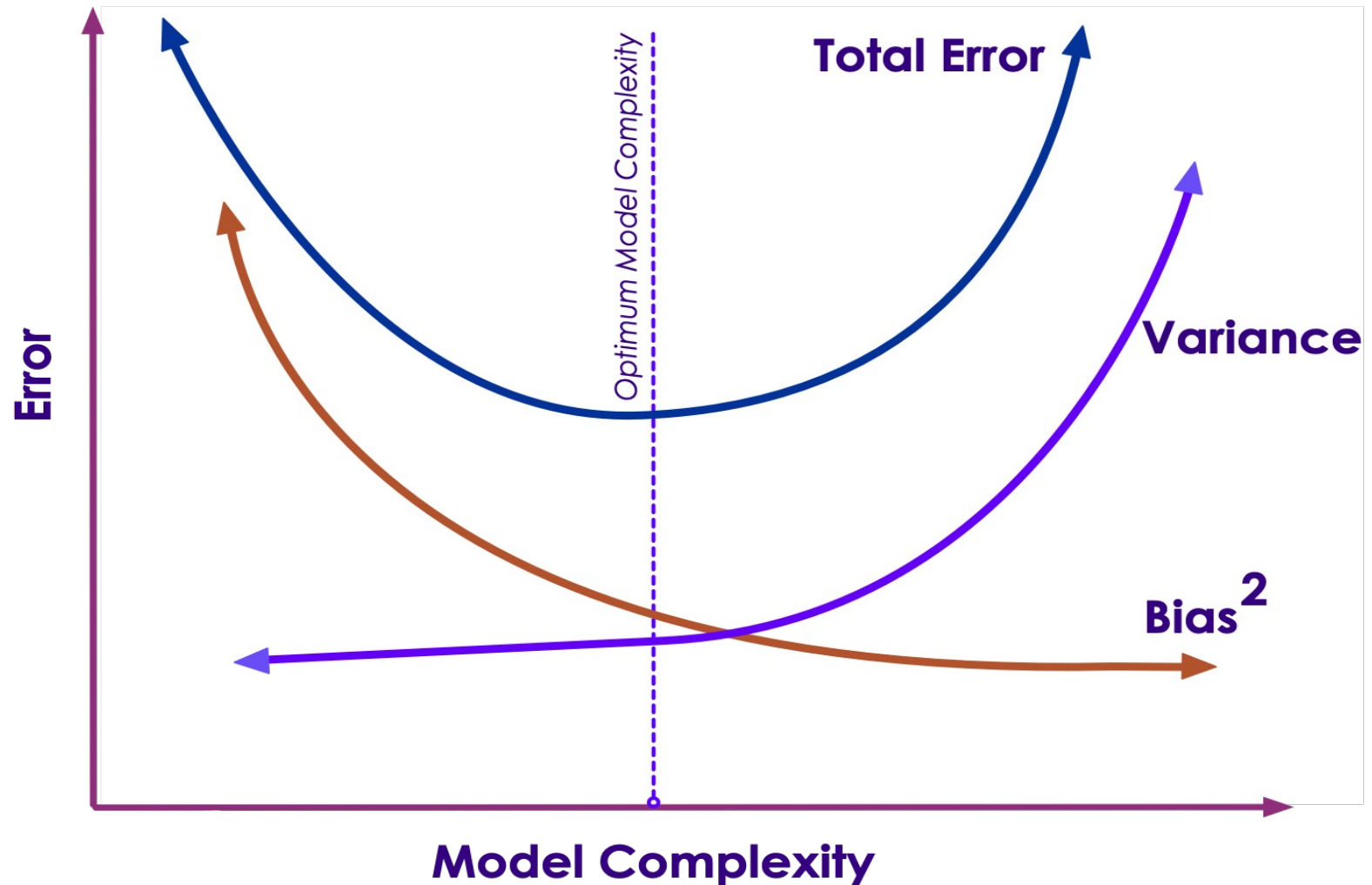
2019-03-12

# Bias-Variance Tradeoff

Low Bias (good)	High Bias,(not good)
Decision Trees, k-Nearest Neighbors and Support Vector Machines	Linear Regression, Linear Discriminant Analysis and Logistic Regression
More able to adopt to complex data	May not be able to adopt to complex data
Low Variance (good)	High Variance (not good)
Modestly influenced by change of data	Strongly influenced by change of data
Parametric methods usually have low variance	nonparametric machine learning algorithms that have a lot of flexibility have a high variance
Linear Regression, Linear Discriminant Analysis and Logistic Regression	Decision Trees, k-Nearest Neighbors and Support Vector Machines.

# Bias-Variance Trade Off

2019-03-12





# Review Questions

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @

2019-03-12

- ◆ Define the following:
  - Supervised learning
  - Model / Algorithm
  - Feature
  - Target / Label

# Review Questions

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12

- ◆ Explain the following
- ◆ Confusion Matrix
- ◆ ROC curve
- ◆ Area under Curve (AUC)
- ◆ Over / under fitting
- ◆ Cross validation / Boosting
- ◆ Feature Engineering

# Further Reading

Licensed for personal use only for Fernando K <fernando\_kruse@dell.com> from Machine Learning at Dell Brazil (QE) @  
2019-03-12