Booz | Allen | Hamilton

Collaboration space, Alexandria, VA

DON'T BE SCARED, MACHINE LEARNING IS EASY!

Mary Johnson

Dakota Shafer

ICEAA CONFERENCE - MAY 2019

AGENDA

- PYTHON & DATA ENGINEERING
  - Data Engineering
  - Feature Engineering
  - Data Engineering Case Studies
- MACHINE LEARNING
  - What is it?
  - How is it different from Statistical Modeling?
  - Applications and Examples
- CASE STUDY
  - Data Engineering: Compiling Messy Data
  - Model Selection
  - Model Tuning
  - Performance
- WHAT'S NEXT?

PYTHON & DATA ENGINEERING

PYTHON? LIKE MONTY PYTHON?

- Actually– Yes.
  - Named after Monty Python's Flying Circus
- Python is a programming language which has grown exponentially because it is:
  - Easily Digestible
    - Python is a very 'readable' language
  - Less Typing – 3 to 5 times shorter than equivalent Java programs, and 5-10 times shorter than equivalents C++ programs!

Python

```
stuff = ["Hello, World!", "Hi there, Everyone!", 6]
for i in stuff:
    print(i)
```

Java

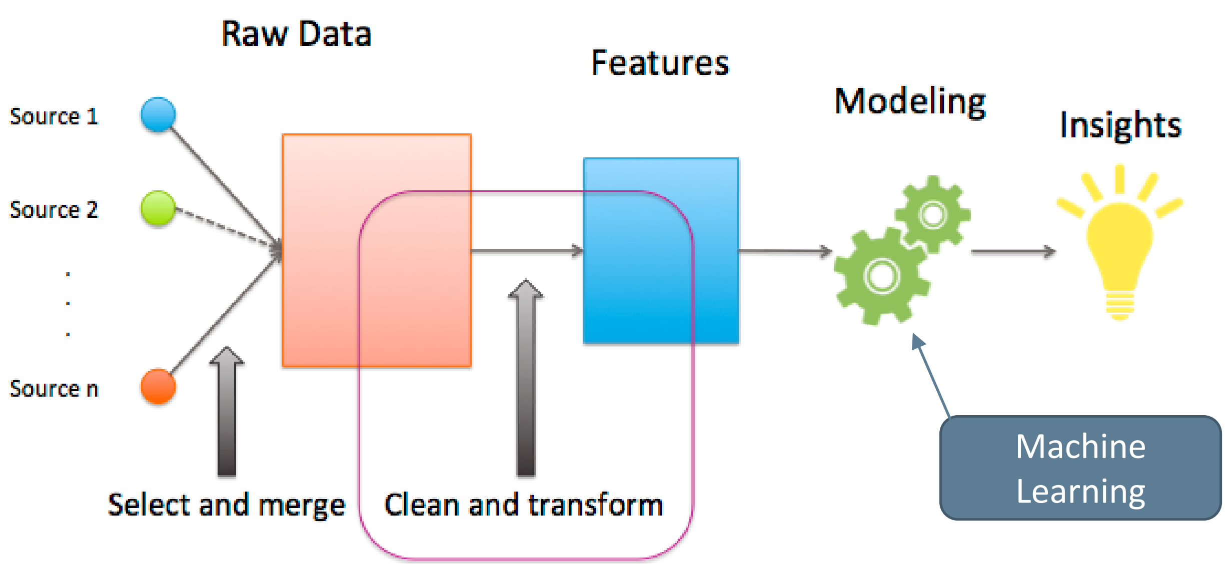
```
public class Test {
    public static void main(String args[]) {
        String array[] = {"Hello, World", "Hi there, Everyone", "6"};
        for (String i : array) {
          System.out.println(i);
        }
    }
}
```

Source: Raygun, Java vs. Python Example

Both programs will output the elements of the array on separate lines, with Python executing the function with much less code

DATA ENGINEERING

- *What is Data Engineering?*
  - Data engineering is the field of transforming data into a useful format for analysis.
  - Data engineers enable data scientists to do their jobs more effectively.
- Steps in Data Engineering:
  - Gathering the data
  - Storing data
  - Cleaning and wrangling data into a usable state



FEATURE ENGINEERING

- *What is Feature Engineering?*
  - Feature engineering is the process of using domain knowledge to create features that enable machine learning algorithms work
    - A feature is an attribute or property shared by all of the independent units on which analysis or prediction is to be done
  - Feature engineering is about creating new input features from your existing ones
- Creating Dummy Variables
  - Dummy variables are a great way to quantify categorical data
  - Create a column for each unique value in a series of data and correspond 1's or 0's (yes's or no's) to that label's attributes

Original Data

UID	Color
1.0	red
1.1	blue
1.2	red
1.3	orange
1.4	yellow
1.5	blue

Data With Dummy Variables

UID	Color_red	Color_blue	Color_orange	Color_yellow
1.0	1	0	0	0
1.1	0	1	0	0
1.2	1	0	0	0
1.3	0	0	1	0
1.4	0	0	0	1
1.5	0	1	0	0

DATA AND FEATURE ENGINEERING IN EXCEL VS PYTHON

Task	Excel	Python
• Data Cleaning (Data Engineering) - Quantitative Values: Ensure everything is on the same scale - Qualitative Values: Correcting different spellings, and inconsistent data entries	- Correct inconsistent entries by hand - Difficult to trace previous steps	- Apply Dictionaries to automate fixing inconsistent data across large amounts of data - Maintain visibility into previous steps
• Manipulating Data (Data Engineering) - Transforming data into a 'Label -> Attribute' format - 'Un-pivoting' data tables	- Manually copy and paste transformed data for each attribute	- Utilize open source libraries such as Pandas to employ built-in functions like 'melt'
• Creating Dummy Variables (Feature Engineering) - Machine Learning models require categorical features be converted to 'dummy variables'	- Create formulas for each separate dummy variable column	- Employ 'get dummies' function in Python

DATA ENGINEERING CASE STUDIES

DOD COST FORM EXCEL EXAMPLE

- **Problem:** Cost Data is received in an Excel Workbook (as seen below), with a new Worksheet/Tab for each WBS element, unconducive to quick analysis.
- **Solution:** Wrote a Python script to read the excel file, loop through all worksheets in the workbook and pull the data into a label-attribute formatted flat file.
  - Originally coded in VBA, which regularly took over 5 minutes to run and would often cause Excel to crash
  - Python script takes roughly 15 seconds to run on average (10 trials)
- **Advantages:**
  - Data is formatted in a way that can easily be analyzed in any software
  - Only needs to be coded once; every time the data is delivered, new WBS elements are added automatically

Cost Data	Costs Incurred to Date		
	Nonrecurring	Recurring	Total
Engineering			
(1) Direct Engineering			
(2) Indirect Engineering			
(3) Engineering Overhead			
(4) Engineering Total			
Support Services			
(1) Logistics			
(2) Program Management			
(3) Financial Support			
(4) Support Overhead			
(5) Support Total			
Materials			
(1) Raw Materials			
(2) Purchased Materials			
(3) Material Overhead			
(4) Material Total			
Other Costs			
(1)Other Costs Total			
Summary			
Total			

	A	B	C	D	E
1	wbs_element	wbs_description	nre_rec_ttl	hours	labor_cost
2	01010204	SOFTWARE DEVELOPMENT	Nonrecurring	124.60	$1,128
3	01010204	SOFTWARE DEVELOPMENT	Recurring	138.00	$103
4	01010204	SOFTWARE DEVELOPMENT	Total	116.60	$1,117
5	01010205	HARDWARE ENGINEERING MANAGEMENT	Nonrecurring	126.00	$60
6	01010205	HARDWARE ENGINEERING MANAGEMENT	Recurring	133.75	$443
7	01010205	HARDWARE ENGINEERING MANAGEMENT	Total	72.75	$532
8	01010206	INTEGRATION MANAGEMENT	Nonrecurring	134.31	$81
9	01010206	INTEGRATION MANAGEMENT	Recurring	60.03	$678
10	01010206	INTEGRATION MANAGEMENT	Total	146.34	$653
11	01010207	TEST MANAGEMENT	Nonrecurring	98.00	$75
12	01010207	TEST MANAGEMENT	Recurring	79.53	$403
13	01010207	TEST MANAGEMENT	Total	68.53	$417
14	01010208	CONFIGURATION MANAGEMENT	Nonrecurring	108.00	$99
15	01010208	CONFIGURATION MANAGEMENT	Recurring	72.50	$416
16	01010208	CONFIGURATION MANAGEMENT	Total	138.50	$470
17	01010209	DATA MANAGEMENT	Nonrecurring	130.00	$84
18	01010209	DATA MANAGEMENT	Recurring	103.74	$269
19	01010209	DATA MANAGEMENT	Total	91.74	$264

TASKBOOK (PDF) EXAMPLE

- **Problem:** Needed to identify travel amounts by WBS number, stored within a 200+ page PDF file. Previous solution would have required a manual inspection of the PDF in order to visually identify the travel numbers to transfer to an Excel workbook.
- **Solution:** Created a Python script to read in the PDF, identify when a travel number was indicated, and place that number and it’s accompanying WBS number into an organized format
- **Advantages:** Investing time writing a Python script allows you to create a program that can reused on future taskbooks, or on the same taskbook, should it be updated

Funding Document

WBS No:	X.X.XX			Band I	$ 50,000.00
TITLE:	XX-XX Hardware			Band II	$ 100,000.00
PI/Code:	M. Johnson		1111	Band III	$ 150,000.00
Func Lead/Code:	M. Johnson		1111	Band IV	$ 200,000.00
Duration	XX/X/XXXX	XX/X/XXXX			

LABOR		(See Box Above)				
Employee Name	WY's	Labor $ Band	Labor Total	NMCI	NEBO	Total
TBD	0.2	Band I	$ 10,000.00	$ 1,000.00	$ 1,000.00	$ 12,000.00
TBD	0.2	Band I	$ 10,000.00	$ 1,000.00	$ 1,000.00	$ 12,000.00
TBD	0.2	Band II	$ 20,000.00	$ 1,000.00	$ 1,000.00	$ 22,000.00
TBD	0.1	Band III	$ 30,000.00	$ 1,000.00	$ 1,000.00	$ 32,000.00
TBD	0.1	Band IV	$ 20,000.00	$ 1,000.00	$ 1,000.00	$ 22,000.00
TOTAL LABOR	**0.8**		**$ 90,000.00**	**$ 5,000.00**	**$ 5,000.00**	**$ 100,000.00**

NON-LABOR			
Description	Non-Labor $	Surcharge	Total
Material	$ 100,000.00	$ 1,000.00	$ 101,000.00
Shipping	$ 1,000.00	$ 1,000.00	$ 2,000.00
NON-LABOR TO	**$ 101,000.00**	**$ 2,000.00**	**$ 103,000.00**

TRAVEL	
Type of travel	Travel $
Regular	$ 100,000.00
TRAVEL TOTAL	**$ 100,000.00**

CONT SUPPORT			
Vendor	Cont $	Surcharge	Total
Tech Services	$ 100,000.00	$ 1,000.00	$ 101,000.00
Admin	$ 100,000.00	$ 1,000.00	$ 101,000.00
NON-LABOR TO	**$ 200,000.00**	**$ 2,000.00**	**$ 202,000.00**

Deliverables	Occurrence	POC	Start Date	End Date	Due Date
Hardware	As Required	M. Johnson			
Software	As Required	M. Johnson			

Impact If Not Funded

Failure

Task Description

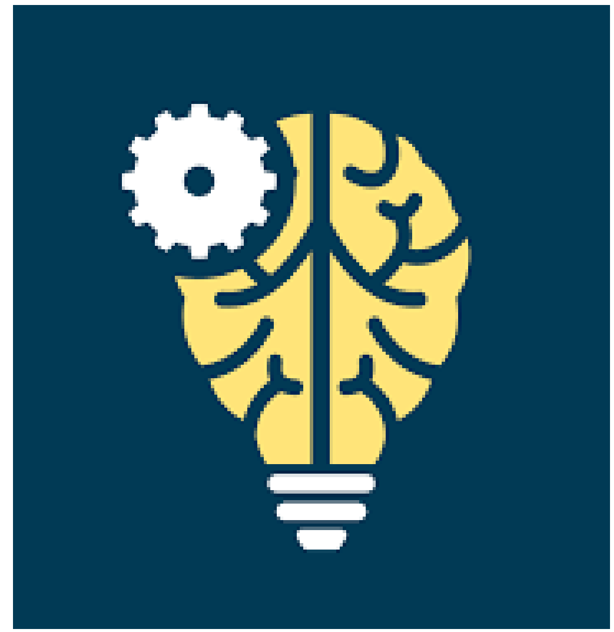
Deliver Hardware

The above funding document is a generalized example

MACHINE LEARNING

GENERAL INTRODUCTION TO MACHINE LEARNING

- *What is Machine Learning*?
  - Machine Learning is a subset of Artificial intelligence that allows software applications to predict outcomes without being explicitly programmed.
- *What Does That Mean*?
  - Machine learning differs from traditional computer programing by teaching the machine through examples instead of coding instructions
- Types of Machine Learning
  - Supervised Learning
  - Unsupervised Learning
  - Reinforcement Learning
  - Deep Learning
  - And many many more!



TYPES OF MACHINE LEARNING

- **Supervised Learning**
  - Finds patterns using both input data and output data
  - Allows analysts to make predictions for unavailable, future, or unseen data based on the training data
  - Examples: Price prediction in sales, trend forecasting in stock trading
- Unsupervised Learning
  - Finds patterns based exclusively on input data
  - Useful when you do not know for what to look – helps to describe existing data
  - Examples: Exploring customer information in digital marketing
- Reinforcement Learning
  - Commonly understood as machine learning artificial intelligence
  - Relies on creating a self-sustained system that improves itself based on labeled data and incoming data
  - Examples: Self-Driving cars, video Games
- Deep Learning
  - Inspired by the structure and function of the human brain, namely the interconnection of many neurons
  - Neural Networks: algorithms that mimic the biological structure of the brain
  - Examples: Image identification

STATISTICAL MODELING VS MACHINE LEARNING

Statistical Modeling

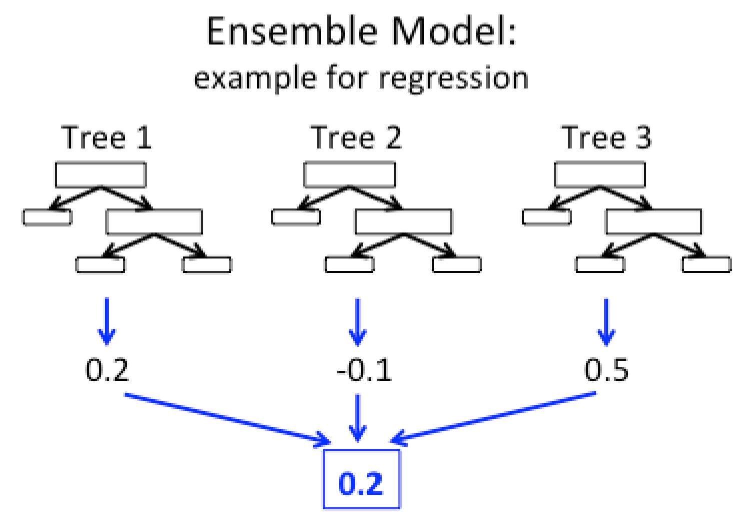
- *Definition:* A mathematical model that embodies a set of statistical assumptions concerning specific sample data
- Mathematical school of though
- Many Assumptions
- Formulation
  - $y = B_0 + B_1 x_1 + e$
- Purpose: To derive inferences about the relationships between variables
- Cannot handle large amounts of variables

Machine Learning

- *Definition*: Method of data analysis that automates analytics model building
- Computer science school of thought
- Few Assumptions
- Formulation
  - Input → output
- Purpose: To make the most accurate predictions possible
- Needs More Data
- Error Focused

SUPERVISED LEARNING: RANDOM FOREST REGRESSION

- **Problem:** You want to know how much buying a used Honda Civic will cost, so you gather a group of car owners to get their opinions.
- **Solution:**
  - You create a list of questions (features) that will explain the cost of the car a little better such as:
    - Mileage
    - Color
    - Navigation
    - Custom Wheels
  - The car owners will have differing opinions on how the features impact cost
  - Each owner creates a decision tree based on their opinion.
  - The combination of all the decision trees results in a forest. The prediction is the average of all trees.



For a **black** civic with **30k miles** with **Navigation Included** and **no custom wheels**:

Person 1: $13,000
Person 2: $17,000
Person 3: $14,000

Random Forest Prediction: $14,667

CASE STUDY EXAMPLE

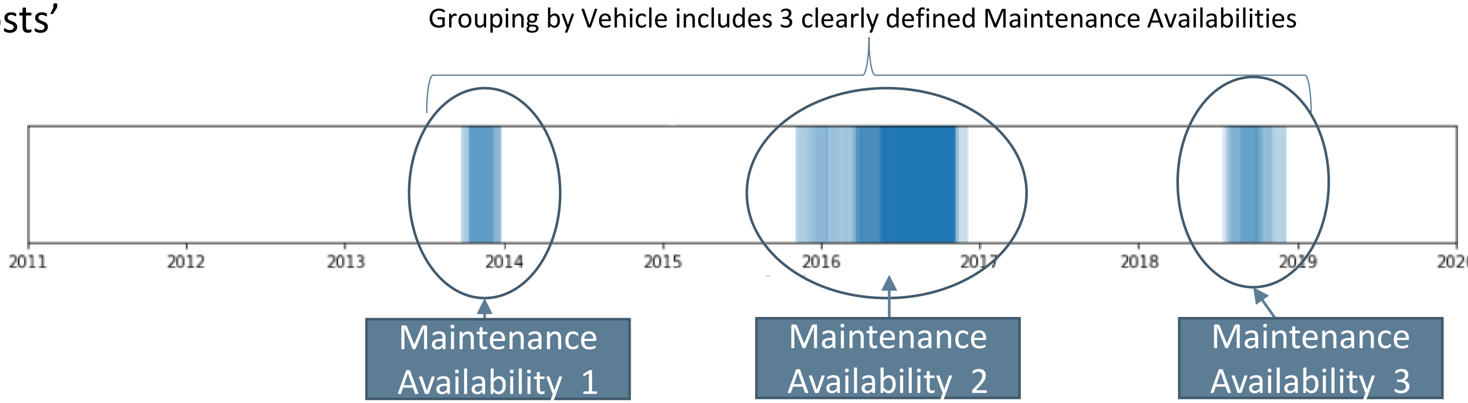
CASE STUDY INTRODUCTION

- **Problem:** Government struggles to forecast installation costs for systems
  - Limited Data
  - Difficult to Understand the Data
- **Solution:** Implement cost analysis using Machine Learning techniques:

1. Data engineering
   - Identify 'Unit Cost'
   - Merge Multiple Datasets
2. Feature Engineering
3. Modeling
   - Train-Test-Split
   - Model Selection
   - Model Tuning

DATA ENGINEERING STEP 1

- **Original Data:**
  - Cost Data: Contains Vehicle, Date, System, Cost, and Hours
  - Installation Data: Contains 16 separate attributes including vehicle, maintenance location, Install Type, Install System, etc.
- **Problem**: Cost data does not identify the specific installation, so the cost data can not initially be merged with installation data
  - Only way to possibly determine an installation unit cost is to compare vehicle maintenance availability dates with cost data dates
  - Difficulties: Not all costs for one installation are within the defined maintenance availability period, costs could be ± a year from the maintenance availability , maintenance availabilities are too close together to determine which costs are for which maintenance availabilities, etc.
- **Solution**: Cost data were grouped by Vehicle and shown on a timeline to visually determine each Installation's 'apparent' start and end date, to then group cost data into Installation 'Unit Costs'



DATA ENGINEERING STEP 2

After completing Data Engineering Step 1, a 'join key' can be used to combine the multiple data sources.

Data Engineering Step 1: Identify 'Unit Cost'

Data Engineering Step 2: Merge Cost and Installation Data

Result: Final Dataset with cost data and installation attributes

Cost Data: 117k rows

Vehicle	Date	Cost
Vehicle1	1/1/2014	$250
Vehicle1	4/1/2014	$75
Vehicle1	5/1/2015	$98
Vehicle2	10/1/2014	$22
Vehicle2	1/1/2015	$59
Vehicle2	2/1/2016	$123

'Unit Costs': 115 rows

Vehicle	Start Date	End Date	Cost
Vehicle1	1/1/2014	5/1/2015	$423
Vehicle2	10/1/2014	2/1/2016	$204

Installation Data: 28k rows

Vehicle	Maintenance Availability Location	Maintenance Availability Type	Issues?
Vehicle1	Location1	Type1	Yes
Vehicle2	Location2	Type2	No

Final Dataset: 115 rows

Vehicle	Start Date	End Date	Cost	Maintenance Availability Location	Maintenance Availability Type	Issues?
Vehicle1	1/1/2014	5/1/2015	$423	Location1	Type1	Yes
Vehicle2	10/1/2014	2/1/2016	$204	Location2	Type2	No

FEATURE ENGINEERING

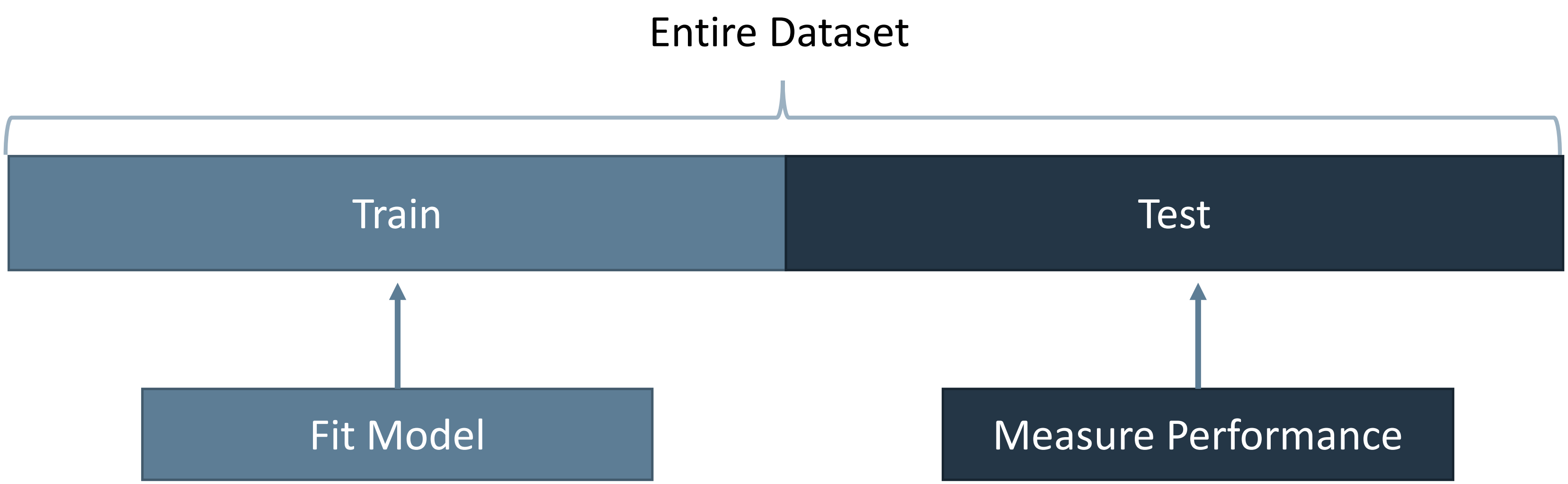
- Installation Data contained 16 Columns
  - 2 Continuous (Duration, LOE)
  - 14 Categorical (Location, Maintenance Availability Type, etc.)
- Categorical features were converted to dummy variables
  - Resulted in 43 total features

Vehicle	Start Date	End Date	Cost	Maintenance Availability Location	Maintenance Availability Type	Issues?
Vehicle1	1/1/2014	5/1/2015	$423	Location1	Type1	Yes
Vehicle2	10/1/2014	2/1/2016	$204	Location2	Type2	No

Vehicle	Start Date	End Date	Cost	Maintenance Availability Location_1	Maintenance Availability Location_2	Maintenance Availability Type_1	Maintenance Availability Type_2	Issues?
Vehicle1	1/1/2014	5/1/2015	$423	1	0	1	0	1
Vehicle2	10/1/2014	2/1/2016	$204	0	1	0	1	0

PREVENTING OVERFITTING: TRAIN-TEST SPLIT

- Overfitting: Refers to when a model fits the data *too* well, and thus the model cannot be generalized to the larger population (in our case, other installations for which we did not have input data)
- Train-Test split is a way to prevent a model from being overfit to a dataset
- Since we have a relatively small amount of data (We had slightly more than 100, note: some ML/AI models are fit to *millions* of observations…), we chose to do a 50/50 train-test split
  - This means that the model is fit to ½ of the data, and then 'scored' on the other half
- Important to note that it is not taking the first half and the second half, samples are chosen at random.



MODEL SELECTION

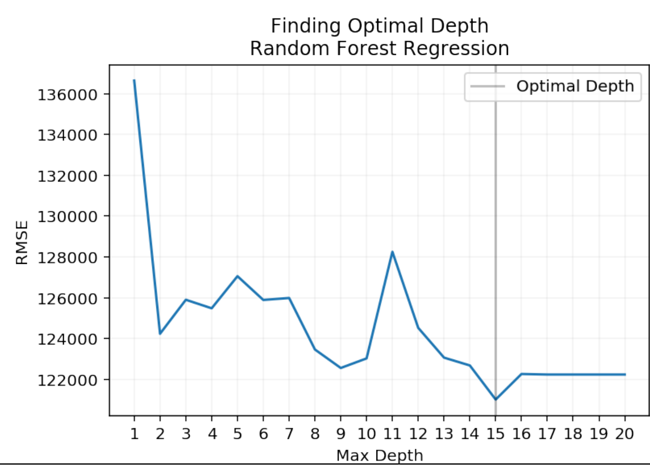
- 8 Models were hand-selected
  - Linear Models:
    - Least Squared Regression
    - Ridge
    - Bayesian Ridge Regression
    - Lasso
    - Elastic Net
    - RANSAC Regressor
  - Decision Trees:
    - Gradient Boosting Regressor
    - **Random Forest Regressor**
- Created a Python script to loop through the following steps:
  - Fit the model to the 'train' data
  - Score the model (think: R^2)
    - Shown as the 'Test Scores' in the table to the right
    - A higher test score is better
  - Find the Root Mean Squared Error
    - RMSE is our measure of performance for this model
    - A lower RMSE is good

Model	Test Score	Test RMSE
RandomForestRegressor	0.893	79,098.005
GradientBoostingRegressor	0.994	86,518.473
Ridge	0.841	87,552.634
RANSACRegressor	0.755	100,766.557
ElasticNet	0.438	109,703.362
BayesianRidge	0.000	151,930.969
LinearRegression	0.997	170,193.503
Lasso	0.997	179,630.348

The Random Forest Regressor was chosen because it had the lowest RMSE

MODEL TUNING

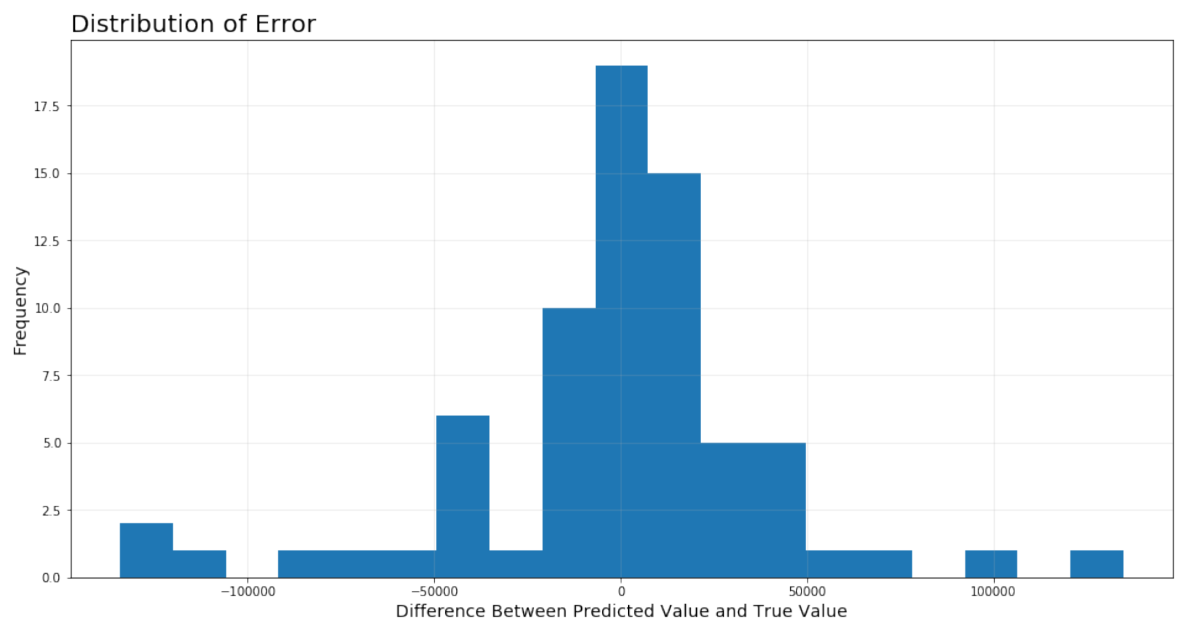
- *What is Model Tuning?*
  - All machine learning algorithms have a “default” set of internal variables (coefficient penalties, number of branches, number of layers, etc.)
  - Model tuning is the process by which you change the internal variables to create the most accurate model
- Tuning a Random Forest Regression Model
  - Optimal Depth:
    - A general machine learning tuning variable by which you determine how many features the model needs to most accurately predict the target variable
    - i.e. how many car features are needed to be the most accurate
  - Of the 43 features, only 15 are needed to be the most accurate
    - The top 15 features as determined by the model tuning are all categorical



	feature	importance
0	avlType_EDSRA	0.280
1	no_alts	0.192
2	TI-install	0.187
3		0.051
4		0.043
5	Shp Yrd Srv	0.025
6		0.025
7	at_PNSY	0.020
8	avlType_EOH	0.018
9		0.016
10	avlType_IMA/DSRA	0.016
11	avlType_DMP	0.014
12	is_VIRGINIA	0.014
13	at_NEWLONDON	0.014
14	avlType_TOA	0.013

PERFORMANCE

- Assuming the error is normally distributed, 75% of the error falls between -$14,737 and $15,731
  - i.e. the model can predict the cost within ± $15,000
- There are some major outliers that are being investigated
  - Multiple instances of error exceeding $100,000
    - 7% of all predictions exceed this threshold
    - Most are over-estimates that need to be investigated on a case-by-case basis



DON'T BE SCARED

The below Python script illustrates speed at which a Random Forest Regression model can be created

```
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(max_depth = 15, random_state = 42)

rf.fit(X,y)

rf.predict(X)
```

```
array([14.9, 15.5, 18.7, 21.7, 25.2, 28.8, 32.8, 37.4, 42.8, 50. ])
```

FUTURE STEPS

- Continue gathering data from external documents
- Quantify Risk
- Test the model on different systems to gauge overall performance
- Consider introducing natural language processing as a means of estimating
  - Some documents provide reasoning as to why a task went over budget - value may be able to be derived from these documents

QUESTIONS?