Decision Trees and Cost Estimating

Josh Wilson

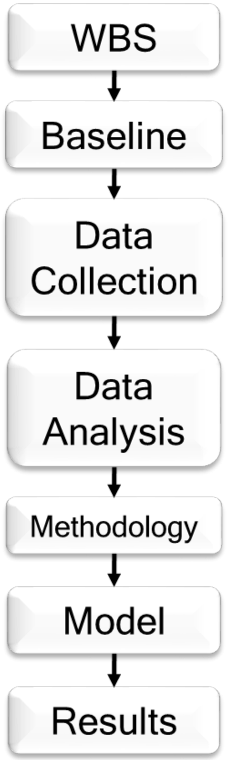
Booz Allen Hamilton

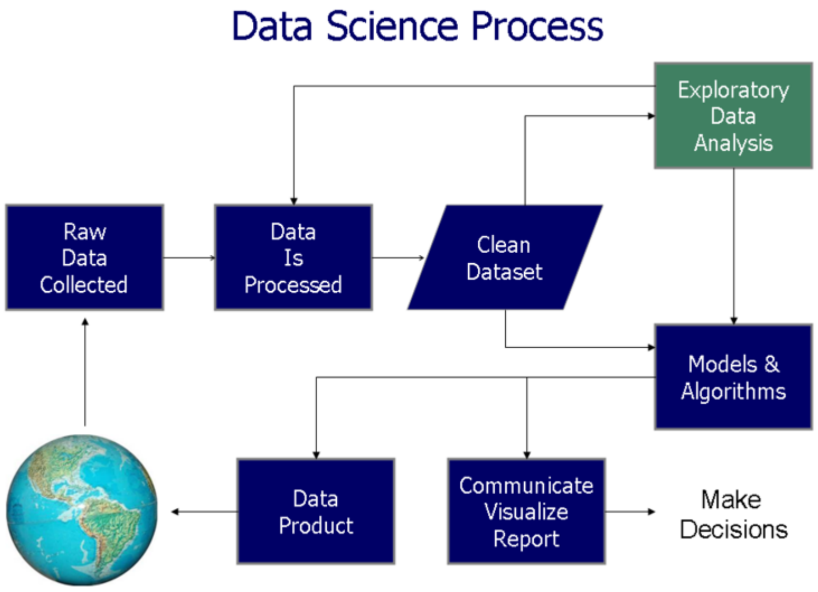
Agenda

- Motivation
  - Integration of Data Science Methods within Cost Estimating Field
- Obligatory Data Science slide
- Decision Trees
  - Definition & Explanation
  - Strengths & Weaknesses
  - Extensions
- Applicability to Cost Estimating
  - Data Challenges
  - Example – Can we predict installation cost overruns?
- Conclusions

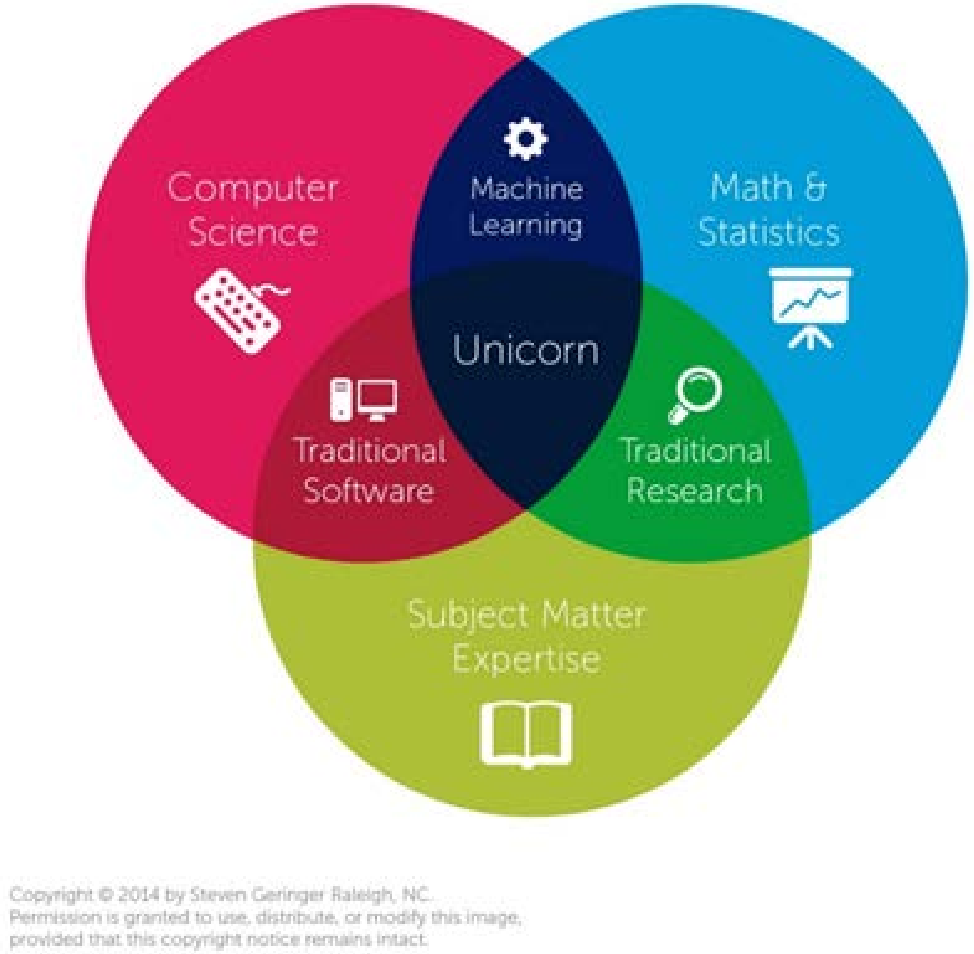
Motivation

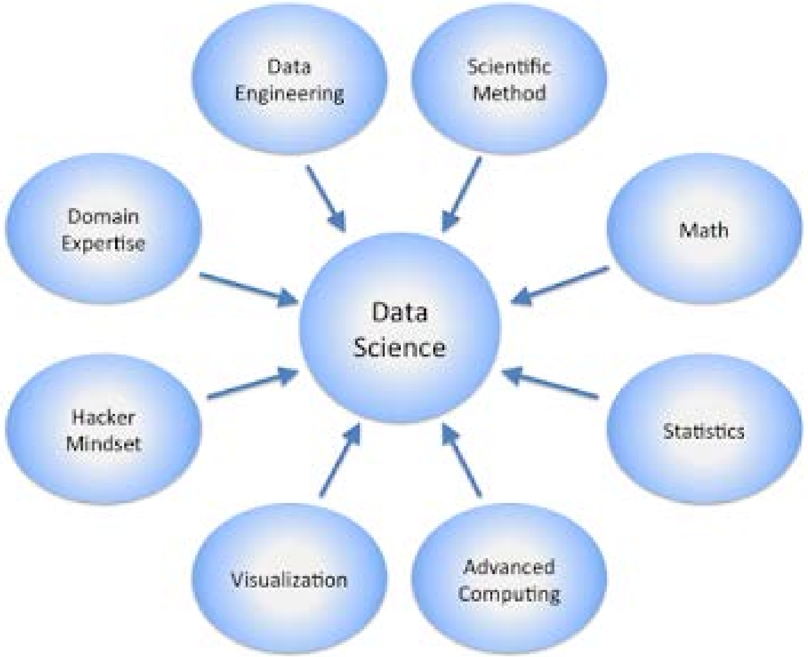
- Background in cost estimating
- Interest in data science
- Exploring application of data science to cost estimating





Data Science?

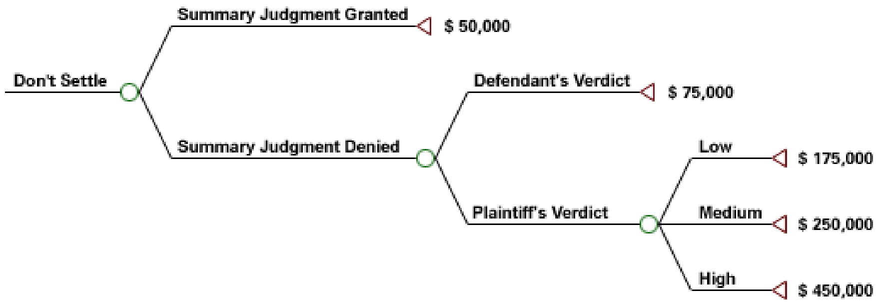




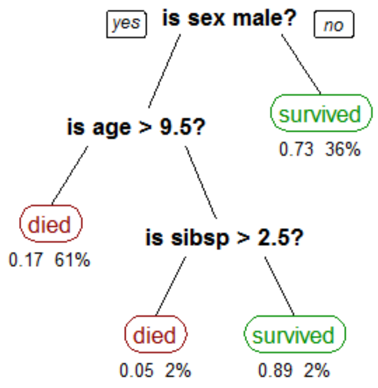
- http://www.prooffreader.com/2016/09/battle-of-data-science-venn-diagrams.html

Decision Trees
First, a clarification…

- There are two types of "decision trees"
- Decision trees for *decision analysis*
  - Model decisions and consequences
  - https://en.wikipedia.org/wiki/Decision_tree
  - These types of trees ARE NOT the topic of this presentation



- Decision trees for *prediction*
  - Maps observations to outcomes
  - https://en.wikipedia.org/wiki/Decision_tree_learning
  - These types of trees ARE the topic of this presentation



Decision Trees
What are they?

- Nonparametric supervised learning method
  - *Nonparametric* = makes no assumptions about underlying data distributions
  - *Supervised* = model learns from examples where we know the outcome
- Can be used for classification or regression
  - Classification if we are trying to predict a categorical outcome
  - Regression if we are trying to predict a continuous outcome
- Makes predictions by learning simple “if-then-else” decision rules from data
  - Recursively partition data into subgroups and apply simple prediction models

- Example: Predicting passenger survival on Titanic
  - If sex is female, then predict passenger survived, else…
  - If age > 9.5, then predict passenger died, else… (and so on)

Decision Trees
How do they work? (the basic idea)

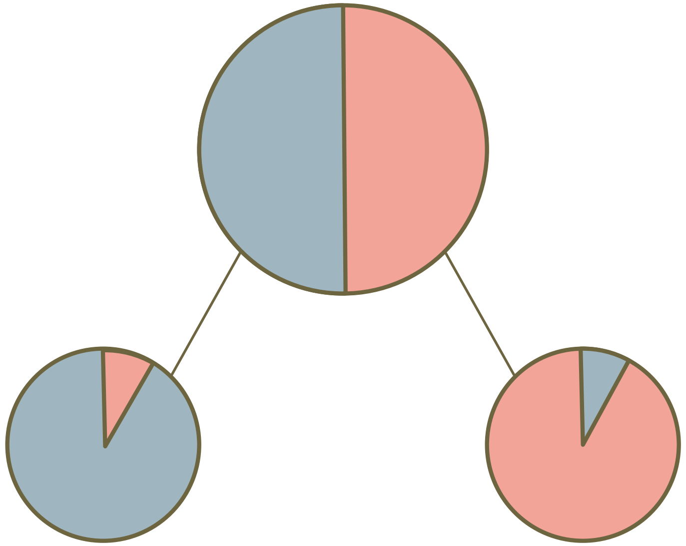
- At each step, split data to maximize homogeneity of target variable within resulting subgroups
  - i.e. We want to separate out the different outcomes as best we can
  - Algorithm scans all possible splits and chooses the "best"

- Process continues on resulting subgroups until stopping condition reached:
  - Maximum # levels reached
  - All subgroups are smaller than some specified threshold size
  - No possible split improves the result

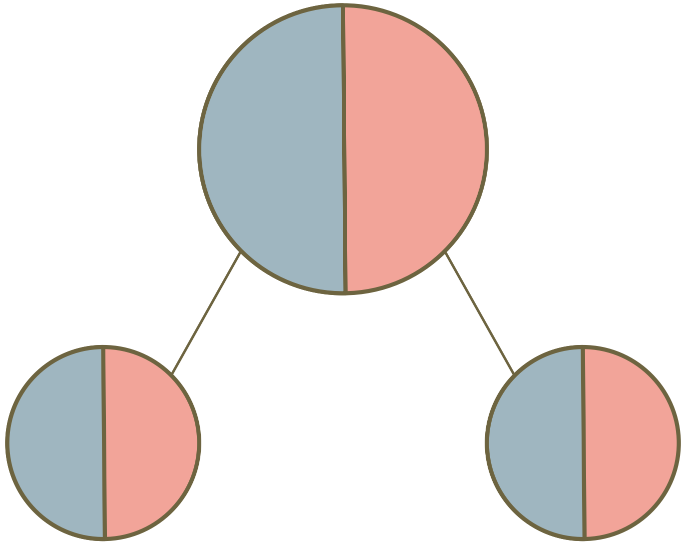
Decision Trees

How do they work? (good vs. bad splits)

- Good split - Separates classes:



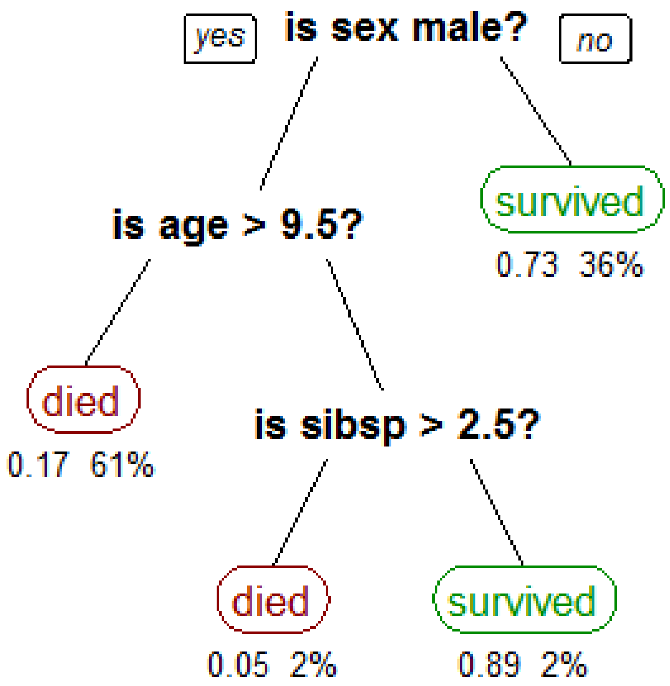
- Bad split – Classes still "impure"



Decision Trees

How do they work? (Titanic example)

- We can predict survival using Titanic passenger demographic info
  - If sex is female, then predict passenger survived, else…
  - If (male) passenger age > 9.5, then predict passenger died, else…
  - If (male, child) passenger is traveling with 3+ family members, predict passenger died, else…
  - Predict passenger survived
- “sibsp” = number of siblings/spouses (i.e. family members) onboard



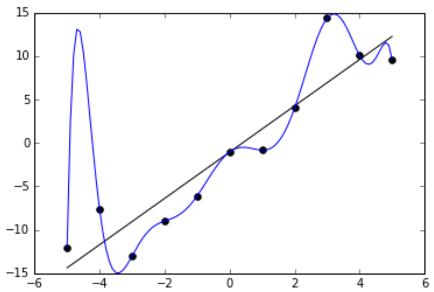
Decision Trees

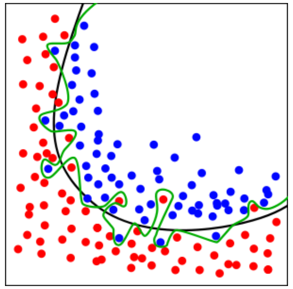
Strengths

- Easy to interpret, explain, and visualize
- Little data preparation or cleaning
  - Can handle both numerical and categorical input data
  - Robust to outliers and missing data
  - Handles nonlinear relationships and correlated variables
  - Ignores useless variables
- Automates modeling of variable interactions
  - i.e. Perhaps age is important if you’re male, but not if you’re female

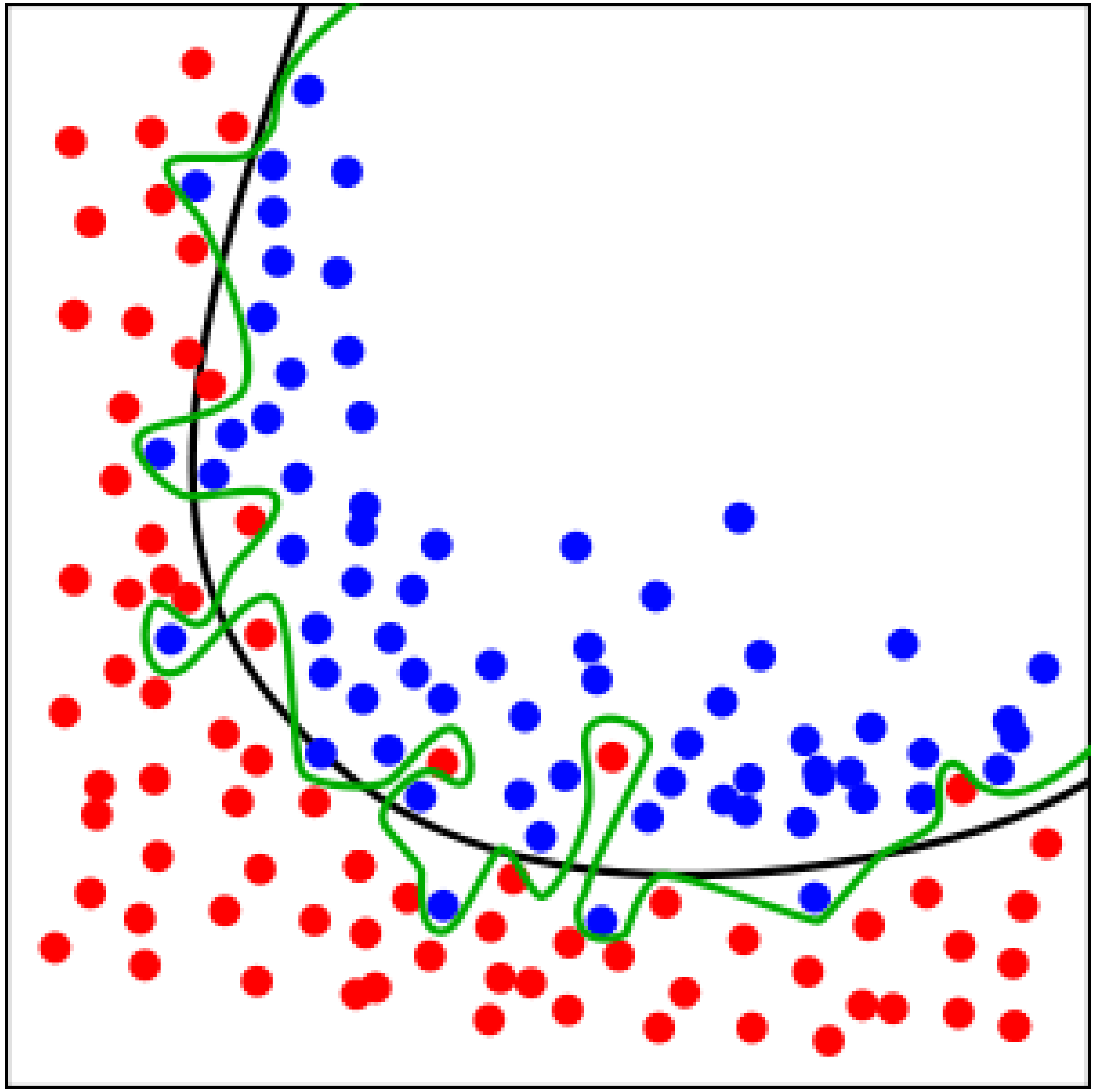
Decision Trees
Weaknesses

- Susceptible to overfitting
  - *Overfitting* = model captures random peculiarities of training data and does not generalize well to new data





- Splitting decisions tend to favor categorical variables with many levels
  - Consider a full name variable in tree to predict Titanic survival…
- “Greedy algorithm” – makes best current decision, possibly bad for long-term



Decision Trees
Extensions

- *Ensemble method* = prediction based on multiple individual models

- Random Forests
  - Ensemble of many individual decision trees, each built from a subset of the data and/or features
  - Generalize to new data better than single trees
- Boosted Trees
  - Ensemble method where new trees are built to improve performance of their sums
    - E.g. by increasing the weight of incorrectly classified data points
  - Overall prediction based on individual trees weighted by accuracy

Decision Trees
Applicability to Cost Estimating

- Another method to predict cost, or things useful for predicting cost
  - Examples:
    - Efforts likely to result in cost over/under runs
    - Categories of SW code growth
- Less impacted by certain types of cost estimating challenges
  - Messy data
    - Mixture of numeric/categorical? Outliers? Missing values? Inconsistent units across different variables?
  - Time constraints
    - Which independent variables are useful? Which are correlated?

Example: *Can we predict installation cost overruns?*

Data / Background

- Raw installation data is from SPIDER database
  - *SPIDER* = "**S**PAWAR **P**EO C4I **I**nformation **D**ata **E**nterprise **R**epository"
- Data for >6k install efforts from a single program office
- 141 columns of data – mostly text/categorical, some numeric, some dates
  - Descriptors of effort – Ship type, location, system, type of install, etc.
  - Cost estimates – Includes initial estimate and actual cost if completed
  - Key event dates – Ship availability, planned installation dates, etc.
- Lots of missing data – eliminating rows with missing data results in 0 rows left

Example: *Can we predict installation cost overruns?*

General Process

- Data preprocessing
  - Filtered data to remove incomplete efforts
  - Removed various ID number columns
  - Converted dates to number of days prior to ship availability
- Defined target variable "Cost Growth Category" as
  - "Over Low" if 0% < Cost Growth % < 40%
  - "Over High" if Cost Growth % > 40%
  - "Under Low" if -40% < Cost Growth % < 0%
  - "Under High" if Cost Growth % < –40%
- Split data into training and test datasets
- Built various models to predict "Cost Growth Category"

Example: *Can we predict installation cost overruns?*

Confusion Matrix for Characterizing Classification Errors

- *Confusion Matrix* = visualization of predicted versus actual outcomes
  - Good if high values along diagonal, low values elsewhere

		Prediction	
		Positive	**Negative**
Actual	**Positive**	TP	FN
	Negative	FP	TN

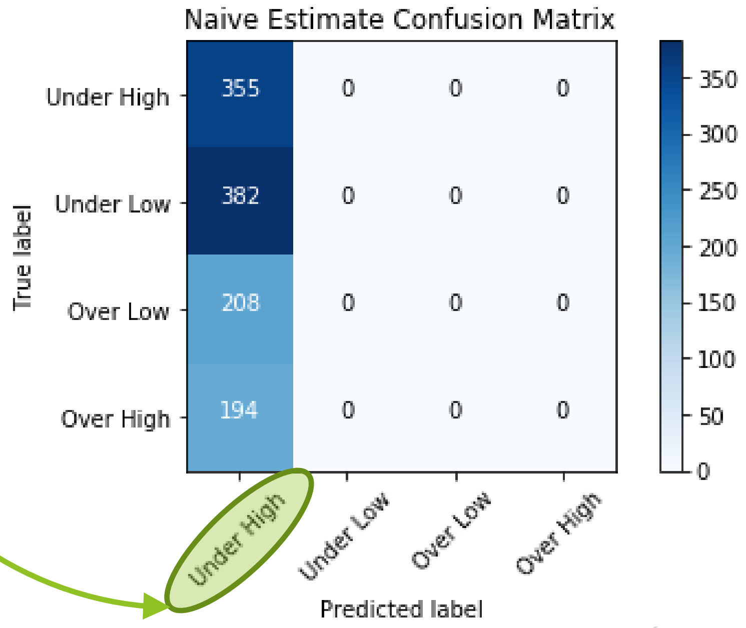
		Prediction				
		Class 1	Class 2	Class 3	...	Class n
Actual	Class 1	**Accurate**				
	Class 2		**Accurate**			
	Class 3			**Accurate**		
	...				**Accurate**	
	Class n					**Accurate**

Example: *Can we predict installation cost overruns?*

"Naïve" Results – Baseline for Comparison

- What if we predict the most common outcome from our training data?
  - Then we correctly predict that outcome, but miss everything else
- 31% prediction accuracy

```
Overrun Category Counts from Training Data:
Under High        557
Under Low         530
Over Low          303
Over High         317
```

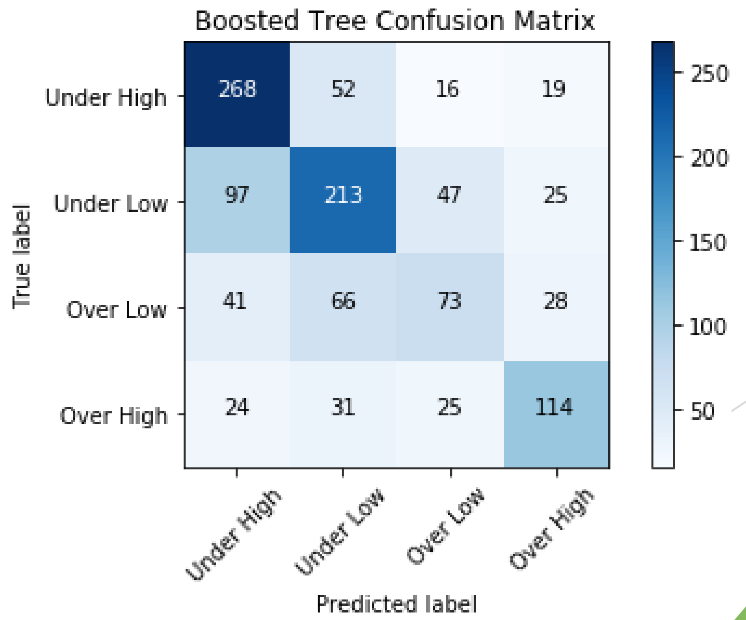


Example: *Can we predict installation cost overruns?*

Current Results - Boosted Tree Model

- Almost 60% prediction accuracy
- Highest accuracy for extreme cases (i.e. high underruns and high overruns)
- Most important features = ship avail duration, lead time for ship check, drawings, system test

```
Overrun Category Value Counts from Test Data:
Under High      355
Under Low       382
Over Low        208
Over High       194
```



Example: *Can we predict installation cost overruns?*

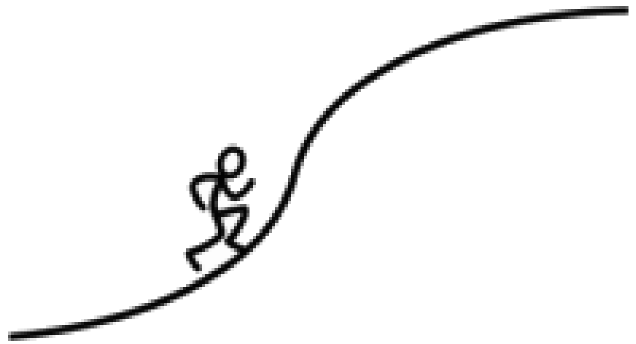
Next Steps

- Find other sources of complementary data
  - Performer? Weather/temperature/season?
  - In general, having more/better data is much better than having a better model!
- Feature Engineering
  - Number of concurrent installations?
- Direct prediction of install cost (i.e. regression instead of classification)

Conclusions

- Decision Trees are a viable tool for the cost estimator
  - Easy to interpret and explain
  - Robust to common deficiencies in data quality
  - Little overhead for variable screening
  - Ensemble methods to address weaknesses of single tree models
  - Good method to expose non-technical people to data science approaches

Way Forward



- Learning curve can be a challenge
- Self-study resources are available
  - Python – http://scikit-learn.org/stable/modules/tree.html
  - R - http://www.statmethods.net/advstats/cart.html
  - Titanic tutorials - https://www.kaggle.com/c/titanic#tutorials
- Other methods that may be appropriate when considering decision trees
  - Naïve Bayes
  - k-Nearest Neighbors (k-NN)
  - Logistic Regression / Linear Regression
  - Support Vector Machines (SVM)

Questions?

Josh Wilson
Associate

Booz | Allen | Hamilton

Booz Allen Hamilton Inc.
1615 Murray Canyon Road
Suite 900
San Diego, CA 92108
Tel (619) 278-3855
Mobile (619) 820-6226
wilson_joshua@bah.com

BACKUP

All Model Accuracy Results

- Most Common Occurrence (Naïve Model) = 31%
- Logistic Regression = 38%
- Logistic Regression + PCA Transform = 48%
- Single Decision Tree Classifier = 50%
- Support Vector Classifier = 50%
- Random Forest Classifier = 55%
- Gradient Boosted Tree Classifier = 59%

Decision Trees

Impurity Functions

- Various decision tree algorithms have been implemented, and various "impurity" metrics are used to measure node homogeneity
  - ID3, C4.5, C5.0 use entropy/information gain:

$$H(T) = I_E(p_1, p_2, \ldots, p_n) = -\sum_{i=1}^{J} p_i \log_2 p_i \qquad IG(T,a) = H(T) - H(T|a)$$

  - CART uses Gini impurity for classification:

$$I_G(f) = \sum_{i=1}^{J} f_i(1-f_i) = \sum_{i=1}^{J}(f_i - f_i{}^2) = \sum_{i=1}^{J} f_i - \sum_{i=1}^{J} f_i{}^2 = 1 - \sum_{i=1}^{J} f_i{}^2 = \sum_{i \neq k} f_i f_k$$

  - CART uses variance reduction for regression:

$$I_V(N) = \frac{1}{|S|^2}\sum_{i\in S}\sum_{j\in S}\frac{1}{2}(x_i - x_j)^2 - \left(\frac{1}{|S_t|^2}\sum_{i\in S_t}\sum_{j\in S_t}\frac{1}{2}(x_i - x_j)^2 + \frac{1}{|S_f|^2}\sum_{i\in S_f}\sum_{j\in S_f}\frac{1}{2}(x_i - x_j)^2\right)$$

- Any strictly convex function can be used