AI IN AEC CONFERENCE 2021
Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

Definition

Project Overview

I am proposing an algorithm-based solution for cost estimating in the Architecture, Engineering, and Construction (AEC) industry. There are two major approaches nowadays for a preconstruction team to complete estimates after receiving design packages. The first one is to dig into the drawing details and build up all the estimate items line by line, and the second one is to generate a Rough Order of Magnitude (ROM) estimate by selecting comparable projects from the project database and simply averaging their costs. The detailed estimate is much more accurate and yet more time-consuming. On the other hand, one can create a high-level estimate in a considerably short amount of period with less confidence, however.

I am a preconstruction engineer/estimator in Hathaway Dinwiddie Construction Company (HDCCo), one of the largest general contractors in the Bay Area, and is still continuously growing. Along with the expansion of the company, our daily estimating work becomes heavier yet allows less time to process. As a company with prestige over 100 years in the industry, we refuse to compromise and will always deliver at our best with no excuses. Thus, I would like to propose a new approach to respond to the challenge. I want to balance the two current workflows, to maximize the pros and minimize the cons of both, by harvesting the legacy data with Machine Learning techniques. I believe that data should have a better as well as faster idea.

Problem Statement

The problem that is to be solved is how to predict the construction cost based on the project parameters with the algorithm-based approach. The solution I am proposing is Machine Learning Regression Model. The inputs of the problem are the cost-related project descriptions, which will be further discussed in the following section, and the output is the cost prediction in the format of United States Dollar (USD). The problem is expected to receive both non-numerical, which can then be translated into true/false values, and numerical inputs, and to generate numerical output; thus, the problem is quantifiable and measurable. The problem is also expected to be replicable as once the questions/inputs are asked correctly, one can retrieve the answer through the same project/algorithm pipeline.

Metrics

There are three evaluation metrics applicable to the Machine Learning regression algorithms. R2 score will be used to evaluate both the benchmark as well as the Machine Learning models and justify their performances.

- Mean absolute error
  The error is equal to the sum of all the absolute values of the differences between the predicted and the true values. One drawback is that the function of absolute value is not differentiable. Therefore, it does not apply to methods like gradient descent.

- Mean squared error
  The error is equal to the sum of the square values, instead of the absolute one, of the differences between the predicted and the true values. It is more commonly used for evaluation purposes than the previous metric.

- R2 Score
  The most common approach for performance evaluation. R2 score is calculated based on comparing the current model to the simplest possible model. R2 score is represented as 1 minus the ratio of mean squared errors of the Machine learning model and the simplest model. The range of the score is between 0 and 1, and a higher score indicates a better model.

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

Machine Learning Pipeline

The workflow is broken down into 12 steps. Please refer to the following diagram (amended from the presentation[1] in Autodesk University, 2019) for the sequences and relations for all the steps. Most of the steps will have their dedicated sections for the more detailed discussions in the rest of the report.

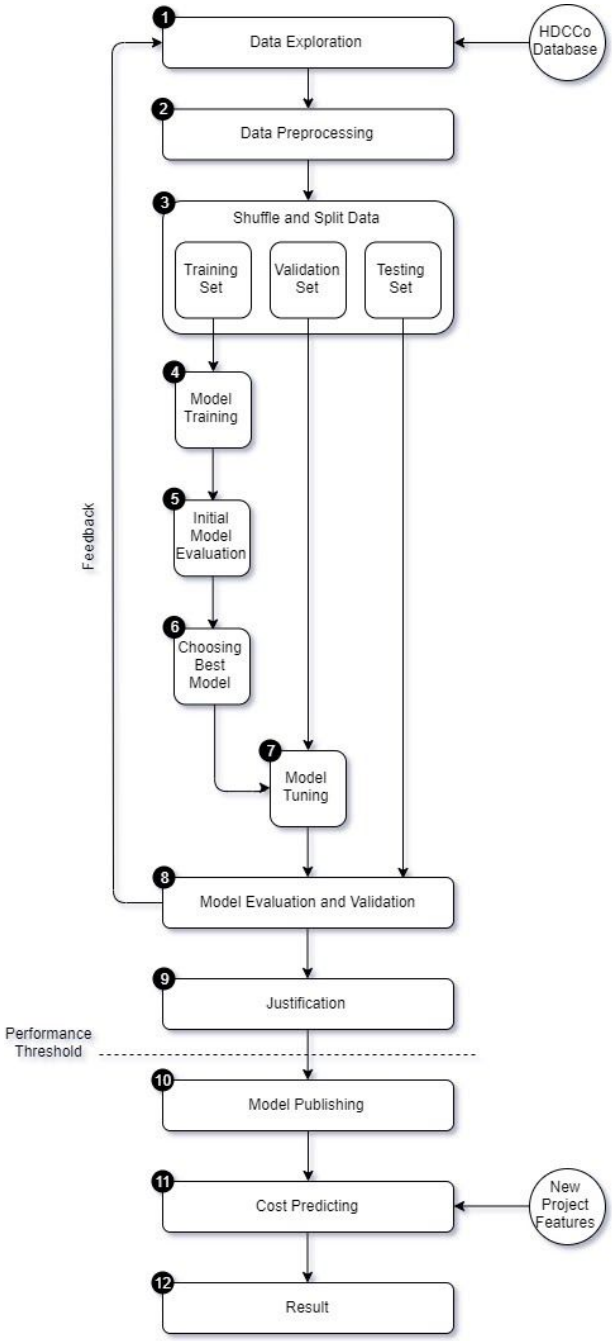

Figure 1. Machine learning model pipeline

[1] Takahiro Oka, *Legacy data analysis on web with forge - Cost prediction in initial design stage*, Nov. 2018, https://www.autodesk.com/autodesk-university/class/Legacy-Data-Analysis-Web-Forge-Cost-Prediction-Initial-Design-Stage-2018

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

Analysis

Data Exploration (ML Pipeline #1)

- Features and target
  The dataset comes from the company's database. Our preconstruction team has constantly been tracking historical project costs and cost-related information. The dataset contains 117 real-world projects constructed by HDCCo, and all are tenant improvement (TI) works. The dataset will be broken down into two sections as inputs and target; each section will be further discussed in detail below. Please also refer to the attached .CSV file for the full dataset.

    - Inputs (X)
        - Numerical
            - Usable square footage (USF)
              The actual space tenants occupy from wall to wall. USF does not include common/service areas of a building such as lobbies, restrooms, stairwells, storage rooms, and shared hallways. For tenants leasing an entire floor or several floors, the usable square footage would include the hallways and restrooms exclusively serving their floor(s).[2]
            - Rentable square footage (RSF)
              USF plus a portion of the building's shared space. Thus, RSF has a strong correlation with USF. These two matrices are very commonly used for representing the project size, and the size of the project should heavily determine the construction cost.
            - Number of floors
              Floor number will decide on the number of flights of communicating stair, the number of Mechanical, Electrical, Plumbing, & Fireproofing (MEPF) shafts, etc. Thus, this input will then determine the cost of the project.
            - Construction duration
              The time required for constructing a certain project reflects its difficulty, and the difficulty should be a factor in determining the cost. The unit of this input is by week.

        - Non-numerical
            - Client type
              The input is single-selection and contains 14 options, including technology hardware and equipment, software and services, hotels and resorts, retailing, and more. The classification of clients reflects their general needs; thus, client type is cost-related.
            - Project scope
              The input is single-selection and contains 5 options, which include building renovation, cosmetic upgrade/partial TI, full TI built-out, guestroom renovation, and lobby renovation. Project scope reflects how many works in general are included in a project, so that is cost-related.
            - Project function
              The input is single-selection and contains 9 options, including education, hospitality, office, etc. Project function reflects the typical programs will be in a project and the code requirements. Therefore, this input is cost-related.

[2] Ben O'Grady, *Difference between rentable square feet versus usable square feet*, 07 July 2016, https://propertymetrics.com/blog/rentable-square-feet/

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

- Project form
  The input is single-selection and contains 4 options, including low-rise and high-rise building, bridge, and business park. Project form is building-code-related and, therefore, cost-related.
- Logistic condition
  The input is multi-selection and contains 10 options, including occupied building, noise restrictions, night works, etc. Logistic condition reflects the difficulty and the manpower and equipment needs of a project. Thus, it is cost-related.
- Design method
  The input is single-selection and contains 3 options, including design-bid-build, design-build, and design-build MEP. Design method reflects the approach of project management and is directly related to the fees for general condition and overhead.
- Fit & finish
  The input is single-selection and contains 3 options, including economical, mid-range, and high-end. Fit & finish provides a rough idea of how costly a project could be based on the level of interiors the owner desires.
- Architectural systems
  The input is multi-selection and contains 6 options. Architectural systems show us how the building is designed architectural-wise and therefore is cost-related.
- MEPF systems
  The input is multi-selection and contains 5 options. MEPF systems show us how the building is designed MEPF-wise and therefore is cost-related.
- Project location
  The input is single-selection and contains 2 options of San Francisco and South Bay. Project location reflects costs of labor, material, and equipment. Also, different cities will have different building-code and union rules, which are both cost-related.
- Project environment
  The input is single-selection and contains 3 options, including campus, urban, and suburban. With similar reasons as location, the project environment is cost-related.

  - Target (y)
    The target value is the total cost of the project, represented as a number with the format of USD. What is noteworthy is that all the amounts are already escalated to today's dollar as the 4th quarter in 2019. As a result, no escalation-related information such as the substantial completion date is listed as input in the dataset.

- Dataset Statistics
  Statistics are listed out by calling *data.describe( )*. There are 8 statistical indices, including count, mean, standard deviation, minimum, 25%, 50%, 75%, and maximum. The indices are calculated for all 69 columns in the dataset, including 68 features and 1 target. The relevant results, abnormalities, and characteristics are discussed as the following.

  - Count
    The dataset contains 117 data points; however, not each and every count shows as 117. Counts of USF and RSF are 115, count of floor shows as 66, and count of duration is 110. The missing counts reveal that missing data are existing in the dataset and need to be dealt with before running algorithms.

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

- Mean & 50%
  It is very often that the mean and 50% (median) numbers in the same column are wildly off. This indicates that the data points are not distributed symmetrically; the points might locate toward one extreme value. Feature scaling might be required to re-distribute the dataset.

- Standard deviation & Mean
  The coefficient of variation (CV), or also called relative standard deviation (RSD), can be further calculated by comparing/dividing standard deviation with mean value. The index is a useful statistic for comparing the degree of variation from one data series to another, even if the means are drastically different from one another. CVs of amount, USF, RSF, floor, and duration equal to 171%, 158%, 159%, 172%, and 79%. The index indicates that except for duration, all the other numerical features are comparatively scattered. Outlier detection is therefore required during data preprocessing.

Exploratory Visualization

In this section, all 4 numerical features are further plotted as graphs to visually explore the characteristics of the dataset.

- Scatter Plot & Linear Regression
  The charts provide a certain degree of insights about the relations between features and target. The scatter plots and the algorithm of linear regression are straightforward yet powerful tools to reinforce the intuition gained from the last step. By looking at the charts of USF versus amount and RSF versus amount, one can easily realize the strong relationships between the two. The discovery also aligns with industry practice. In the early design stage, in order to quickly generate a rough number, people will commonly use the amount per USF or RSF as a basic matric/factor, multiplying the number with the size of the project to retrieve the construction cost. On the other hand, compared to USF and RSF, floor and duration have relatively weak relationships with the amount.

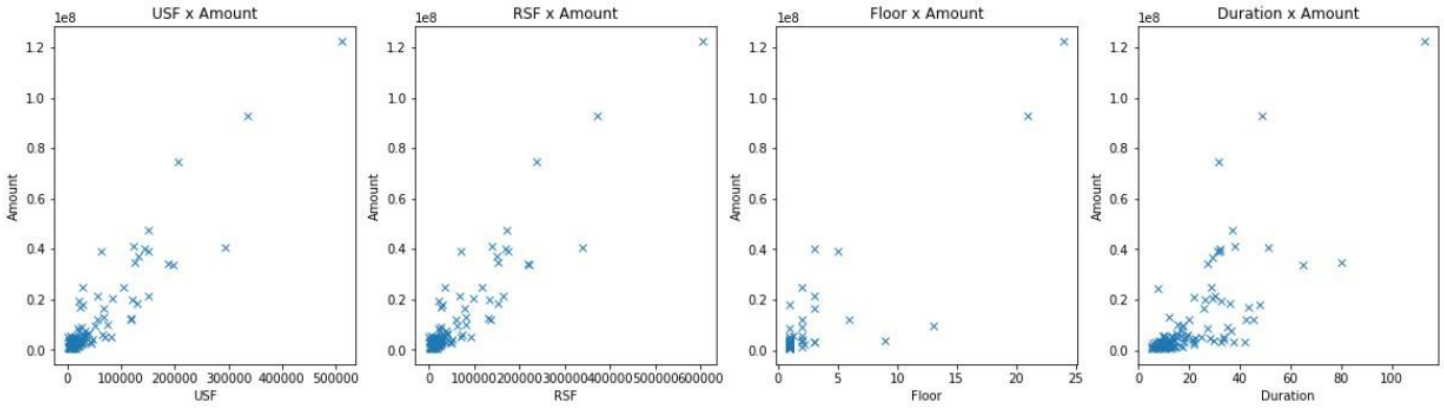

Figure 2. Scatter plots, numerical features versus amount

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

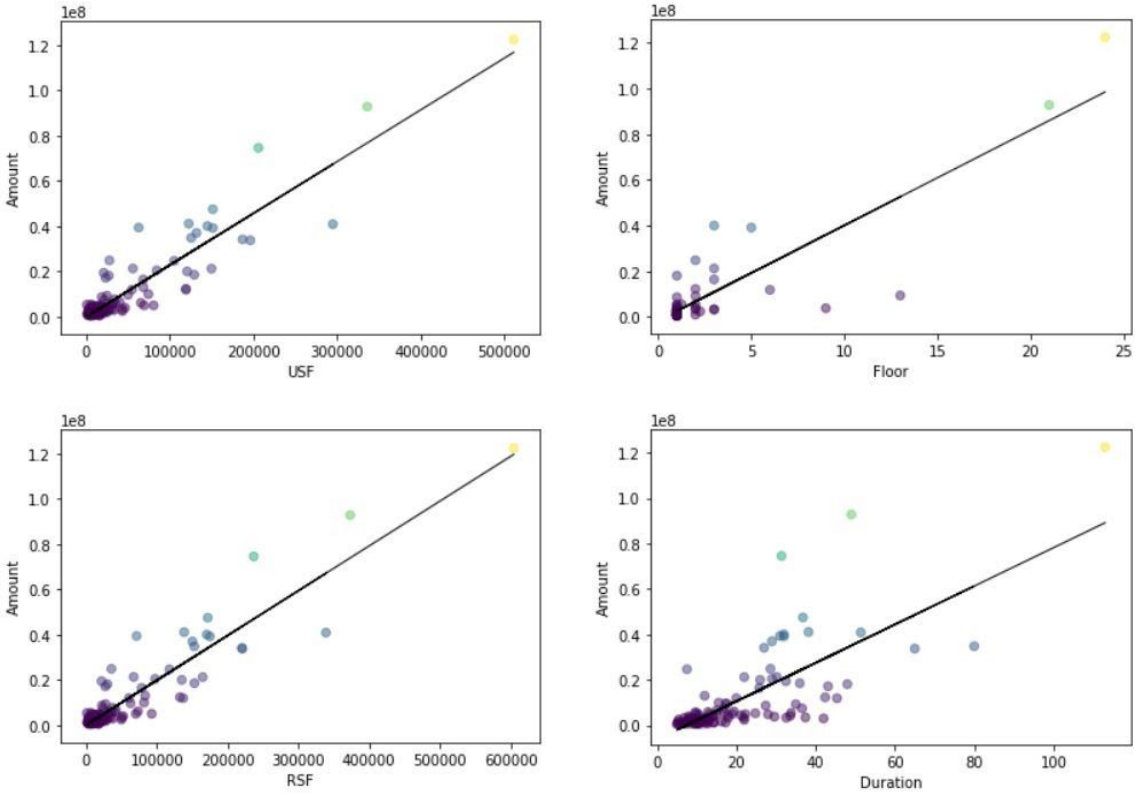

Figure 3. Linear regression plots, numerical features versus amount

- Feature Distribution
  Based on the charts below, one can realize that all four numerical features are heavily skewed. None of the features is close to the normal distribution, and the ranges are not properly normalized. Most of their values tend to lie toward a small number; however, the dataset also contains a few much larger numbers. Data transformation with a certain scale, as well as data normalization, might be required during preprocessing.

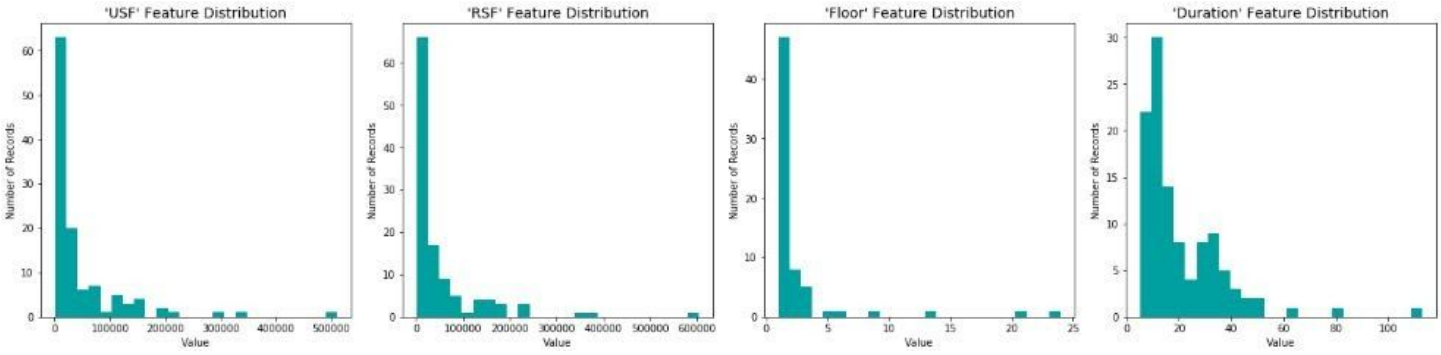

Figure 4. Numerical feature distributions

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

- Feature Correlation
  The scatter matrix below further reinforces my understanding of data distribution. All of the features are skewed and not normalized. Furthermore, the matrix also shows a very strong correlation between USF and RSF; the correlation coefficient is as high as 1. The find is aligned with the industry practices as one will simply multiply USF by 1.15 to retrieve an estimated RSF number while the real number is not accessible. Project size (USF & RSF) achieves high correlation numbers in between Floor and Duration, which is aligned with the intuition, too.

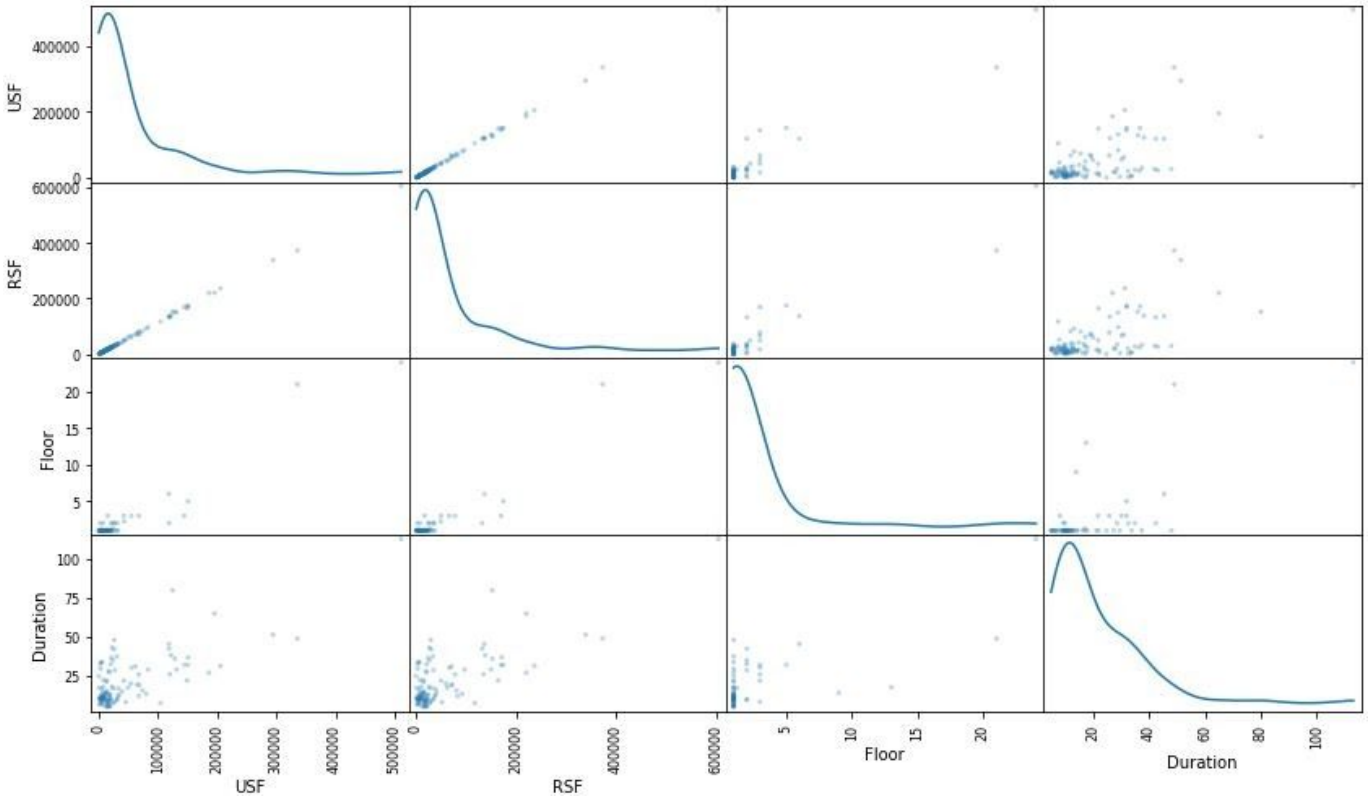

Figure 5. Scatter matrix plot for numerical features

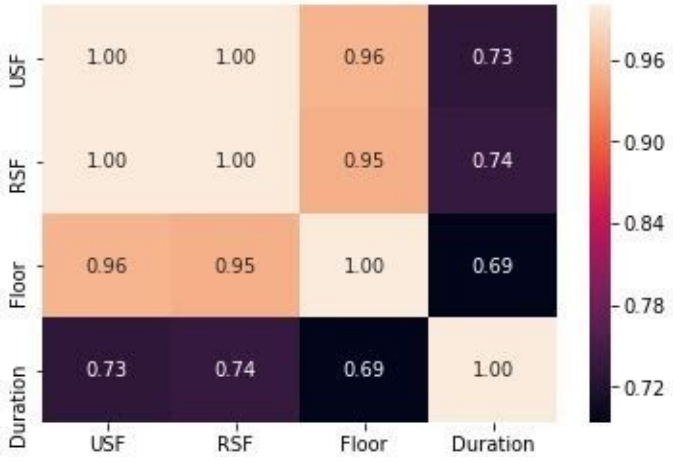

Figure 6. Correlation heat map for numerical features

Algorithms and Techniques

The solution I am proposing here is Machine Learning Regression algorithms. According to the Scikit-learn algorithm cheat-sheet[3] listed below and the characteristics of the given dataset, with more than 50 and less than 100K data points and with more than a few important features, the following models will be adopted to solve the problem:

- Ridge Regression
- Ensemble Regressors
- SVR (kernel = 'linear')
- SVR (kernel = 'rbf')

In addition to the above estimates, I would also like to try out more regression models mentioned during the class:

- K Neighbor
- Random Forest, as an ensemble regressor

Per the proposal review, I will further apply the following powerful algorithms on the problem:

- Xgboost
- LightGBM

Regression models will generate a numerical outcome so that the solution will then be quantifiable and measurable. One can expect to receive a similar outcome when the appropriate inputs are provided; thus, the solution is replicable.

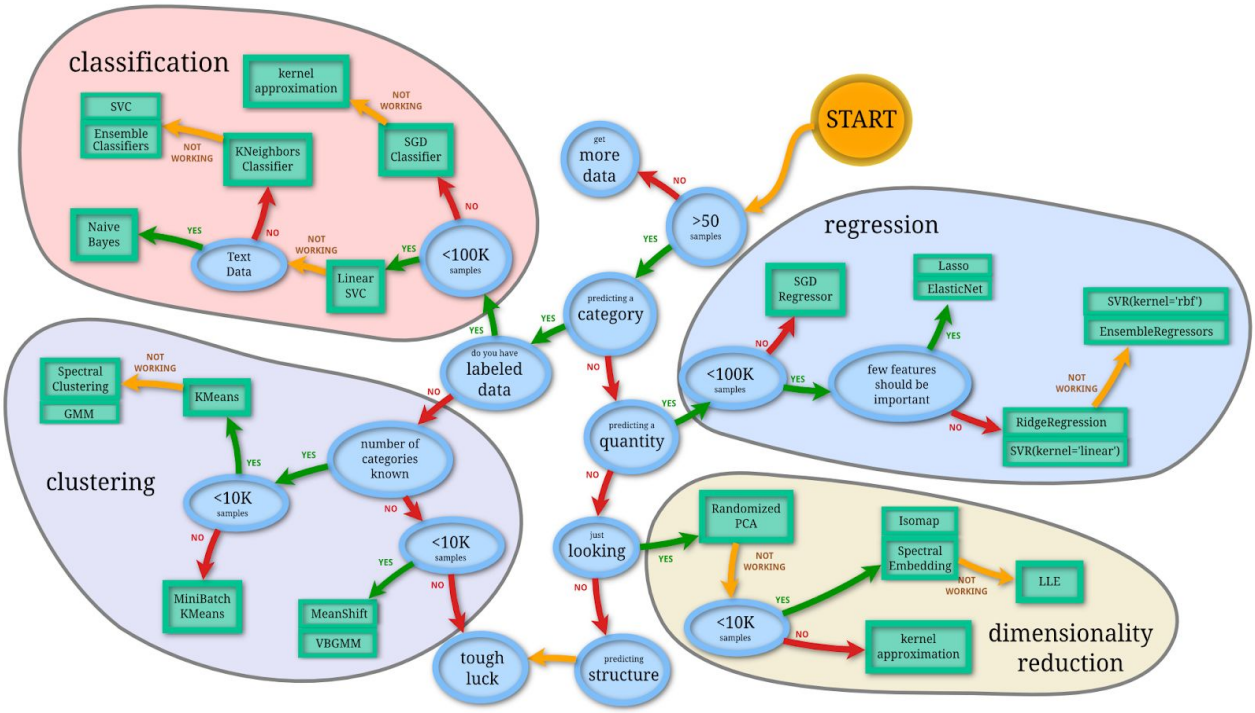

Figure 7. Scikit-learn algorithm cheat-sheet

[3] Scikit-learn developers, *Choosing the right estimator*, 2007 - 2019, https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

Benchmark

- Existing Workflow: Matric of Amount per Area
  The benchmark model I am proposing here is our current approach to leverage historical data. And, the existing methods in our preconstruction team for cost prediction are as straightforward as filtering and averaging. What is noteworthy is that the current workflow is a manual process and typically full of trial-and-error. It is very likely that one is required to play with the model a few times to obtain the desired result.

  When a new project comes in, we will first collect all the project characteristics as inputs. And, Instead of directly using the total amount as output, we create another metric called amount per square footage for each and every past project. Based on the inputs from the new project, we will then filter the database down to only the projects perfectly aligned with the given characteristics. Finally, we will eliminate the extreme data and average the amounts per square footage, then multiply the averaged number by the project size to retrieve the predicted cost. The following shows our HDCCo database and dashboard built with Google sheet and Google Apps Script.

  Due to the limitation of data number and the infinite possibilities of feature combination, it is apparent that sometimes the criteria are too strict so that no data point remains after the filtering process. As a result, users need to use their own judgments to determine the features they would like to use to select/filter the data, which eventually makes the process become not replicable. For the purpose of consistency, I am going to use only the top 3 common features for the filtering process, including client type, project scope, and project function.

  The output of the benchmark model is the project predicted cost, which is exactly the same as the Machine Learning Regressions. Thus, the model is quantifiable as well as measurable. Moreover, its performance will be evaluated by the same metrics, too; it then becomes comparable to the regression models.

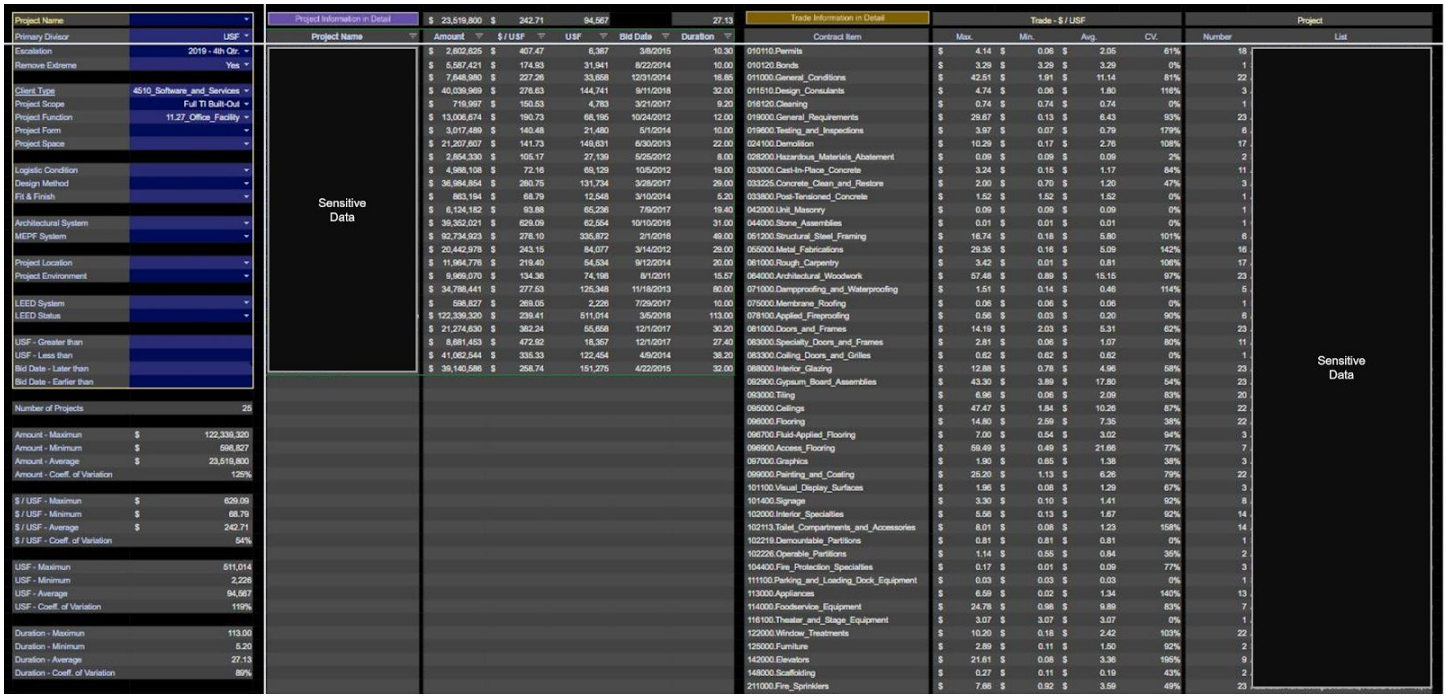

Figure 8. HDCCo database dashboard

- Simple Algorithm-based Approach: Linear Regression
  The most commonly used metric in the industry to generate the estimated cost is the amount per area. Estimators will first retrieve the metric value through their own workflows and methods, ours is filtering and averaging, for example, and they will then multiply the unit cost and project size to obtain the total cost. What is noteworthy is that the approach itself, as a matter of fact, is built upon a fundamental assumption of the strong correlation between total costs and project sizes. What is more is that if one chooses to neglect the economies of scale while estimating, he/she predicts the cost purely based on the assumption of the simple linear correlation between project amounts and project sizes.

  Thus, the algorithm of linear regression can be considered as a naive benchmark. And, since the correlation coefficient between USF and RSF equals 1, there should not be a big difference in the evaluation result while picking either one for training purposes. Here, USF is chosen and will be treated as the input for the simple linear regression to predict the amount of project. The matrix of R2 score will be applied to the model to evaluate its performance so that the benchmark can be compared with the more complex algorithm.

Methodology

Data Preprocessing (ML Pipeline #2)
Only the dataset with quality can lead to a meaningful model. Data preprocessing is a fundamental step to assure the model reliability through cleaning, formatting, infilling, and scaling the raw data. In this section, 6 preprocessing steps will be performed sequentially to help with the prediction/estimation outcomes based on the understanding of the data characteristics in the previous section.

- Escalation
  All the data points/projects are recorded at different timings; some of them might have a decade difference in between. The process of escalation is to convert the project amounts in various years to today's dollar. Several construction cost indices are adopted, mixed, and weighted to generate the index for our own needs. The indices include but not limit to:

  - Engineering News-Record (ENR), Building Cost Index (BCI) & Construction Cost Index (CCI)[4]
  - TBD Bid Index[5]
  - Turner's Building Cost Index (BCI)[6]
  - Mortenson's Construction Index (CCI)[7]
  - Local subcontractors' labor rates

  What is noteworthy is that the step of escalation happens in our own database/Google sheet environment and completed in advance to the entire machine learning workflow. Therefore, the methodology will only be discussed here and will not be included under the code blocks in Jupyter Notebook.

[4] Engineering News-Record, *Construction Economics*, https://www.enr.com/economics

[5] TBD Consultants, *TBD Bid Index*, http://www.tbdconsultants.com/mobi/TBDBidIndex.htm

[6] Turner Construction Company, *Turner's Building Cost Index*, http://www.turnerconstruction.com/cost-index

[7] Mortenson Construction Company, *Mortenson's Construction Index*, https://www.mortenson.com/cost-index

- One-hot Encoding
  According to the previous data exploration, the dataset contains both numerical and non-numeric features. Since machine learning algorithms only accept numerics as inputs, all the categorical variables must be converted into the format of numeric. One-hot encoding will generate variables of 0 & 1 and re-represented the data for each and every possible category of each non-numeric feature.

  The step of one-hot encoding also happens in our own database/Google sheet environment and completed prior to the entire machine learning workflow. Therefore, the preprocessing step will only be discussed here and will not be included under the code blocks in Jupyter Notebook.

- Missing Data
  Based on the process of data exploration and the missing matrix plotted below, one can understand that the following features do contain missing data points:

  - USF, 2 missing points
  - RSF, 2 missing points
  - Floor, 51 missing points
  - Duration, 7 missing points

  Fortunately, all the features listed above are numerical features, and based on the suggestion from the article[8], the method of mean is chosen to infill the voids. By calling *Imputer* from sklearn, the mean value of each feature will then be adopted for all the missing values.

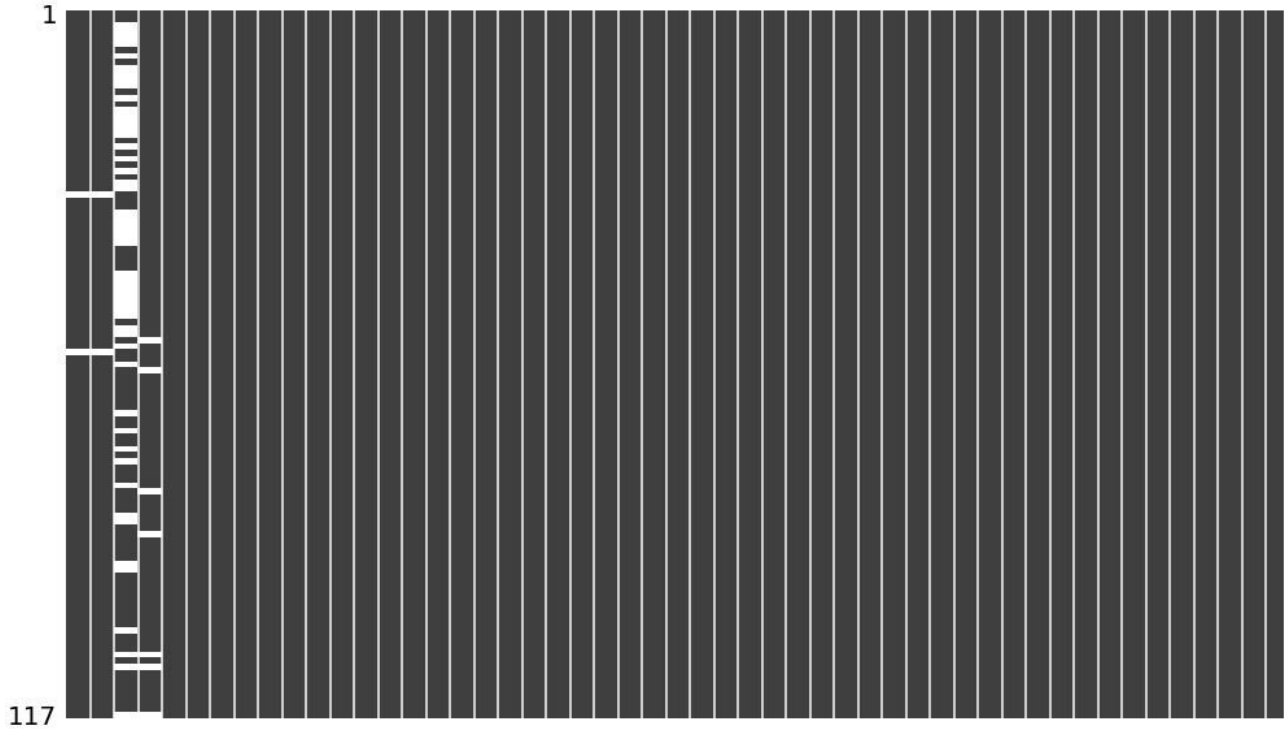

Figure 9. Missing matrix

[8] Boyan Angelov, *Working with Missing Data in Machine Learning*, Dec. 10, 2017, https://towardsdatascience.com/working-with-missing-data-in-machine-learning-9c0a430df4ce

- Feature Transformation
  Based on the distribution plots in the data exploration steps, one can realize that means and medians vary significantly for all of the numerical features, which indicates that they are not normally distributed. Furthermore, all the numeric features are all skewed, having most of the data points located toward a small value, while having a few data points on the extreme right simultaneously.

  One of the best approaches for the issue is data transformation[9], and one of the most commonly used methods is natural logarithmic transformation. After the transformation/scaling process, the distributions are more alike normal distribution, and those outliers (very large or very small values) will have much less negative impacts on machine learning algorithms, too.

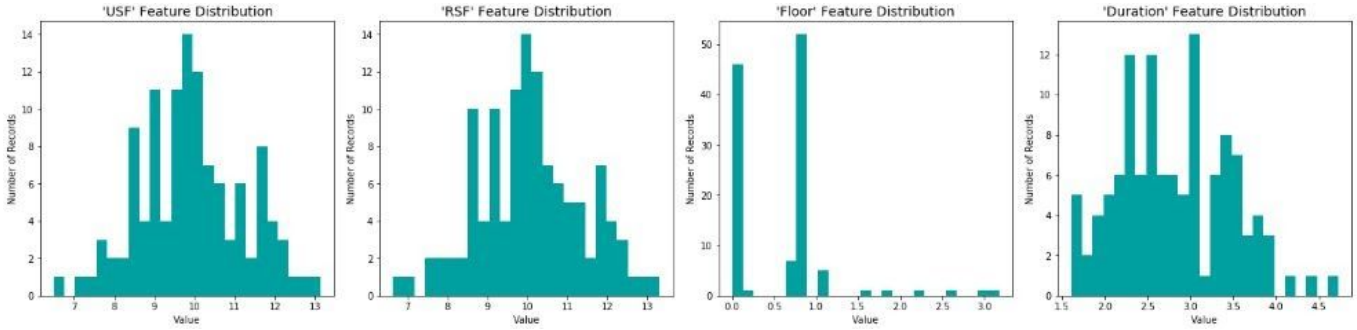

Figure10. Log-transformed numerical feature distributions

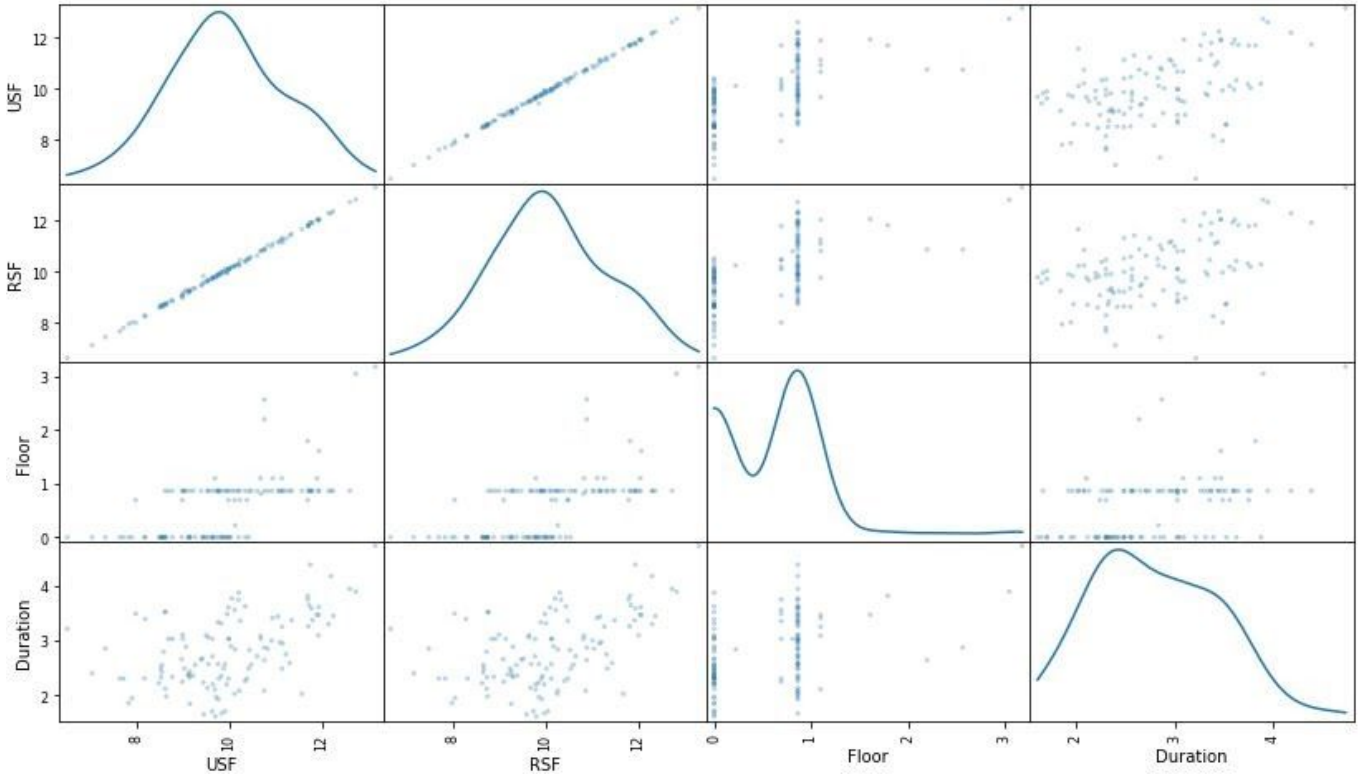

Figure 11. Scatter matrix plot for log-transformed numerical features

[9] Wikipedia, *Data transformation (statistics)*, https://en.wikipedia.org/wiki/Data_transformation_(statistics)

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

- Outlier Detection
  According to the scatter matrix above, I assume that there are still a few outliers in the dataset, even after the process of feature scaling. Based on the suggestion from the article[10], the method of standard deviation/z-score is chosen to detect and remove outliers/anomalous. The thresholds are set as the following, in which 98.7%[11] of the data points should lie within for a normal distribution:

  - Lower limit = mean - 2.5 x standard deviation
  - Upper limit = mean + 2.5 x standard deviation

  The results confirm my initial assumption. The following numerical features do contain outliers which are located outside of the 2.5 standard deviation boundaries and removed accordingly:

  - USF: Data point #42
  - RSF: Data points #0 and #42
  - Floor: Data points #0, #1, #30, and #56
  - Duration: Data point #0

- Normalizing Numerical Features
  In addition to feature transformation, feature scaling should also be applied to the dataset. Normalizing the data is to make sure that all the features are within the same scale, and therefore to be treated equally during the training process. What is noteworthy is that even though the values/meanings are changed from the raw data, the shapes/distributions still remain the same.

Implementation

- Shuffle and Split Data (ML Pipeline #3)
  In this section, the dataset (including both features and target) will be split into 2 subsets as training and testing with a certain ratio, 80-20 in our case here. Data points will be randomly shuffled in the splitting process to avoid bias in the ordering of the dataset.

  Validation set, on the other hand, will be split and manipulated later for grid-search/cross-validation to fine-tune the model parameters and select the optimized combination. The process will be further discussed in detail under the model tuning section.

  - Training Set: 80%, 89 samples
  - Testing Set: 20%, 23 samples
  - Validation Set

- Creating a Training and Predicting Pipeline (Model Training, ML Pipeline #4)
  In this section, the training and predicting pipeline is coded with its maximum flexibility. It can be quickly applied to a variety of machine learning algorithms and data sizes, and the predicted results can be calculated and displayed in an efficient and clean manner. The inputs and output of the pipeline are listed below:

[10] Will Badr, *5 Ways to Detect Outliers/Anomalies That Every Data Scientist Should Know (Python Code)*, Mar. 5, 2019, https://towardsdatascience.com/5-ways-to-detect-outliers-that-every-data-scientist-should-know-python-code-70a54335a623

[11] Wikipedia, *68-95-99.7 rule*, https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule

  - Inputs
    - learner: The learning algorithm to be trained and predicted on
    - sample_size: The size of samples (number) to be drawn from training set
    - X_train: Features training set
    - y_train: Income training set
    - X_test: Features testing set
    - y_test: Income testing set

  - Output, a Python library including the information below
    - learner: The learning algorithm to be trained and predicted on
    - training_size: The size of samples (number) to be drawn from training set
    - R2_train: R2 score of the training set
    - R2_test: R2 score of the testing set

Refinement

- Initial Model Evaluation (ML Pipeline #5)
  All of the 7 regression models listed in the Algorithms and Techniques section are fed into the pipeline above. What is noteworthy is that since the training set only contains 89 samples, the training process should run fairly fast, and no duration issue needs to be considered; all of the data points are used to train the model here.

  The performances of 7 models are then evaluated by the matrix of R2 score. The results are as follows:
  - Ridge Regression
    - R2 score on training set: 0.902
    - R2 score on testing set: 0.662
  - Ensemble Regressors (Random Forest)
    - R2 score on training set: 0.945
    - R2 score on testing set:0.647
  - SVR (kernel = ‘linear’)
    - R2 score on training set: -0.180
    - R2 score on testing set: -0.185
  - SVR (kernel = ‘rbf’)
    - R2 score on training set: -0.180
    - R2 score on testing set: -0.185
  - K Neighbor
    - R2 score on training set: 0.683
    - R2 score on testing set: 0.599
  - Xgboost
    - R2 score on training set: 0.998 (Highest)
    - R2 score on testing set: 0.781 (Highest)
  - LightGBM
    - R2 score on training set: 0.838
    - R2 score on testing set: 0.548

- Choosing the Best Model (ML Pipeline #6)
  Xgboost is chosen as the best model as it achieves the highest scores on both training and testing datasets compared to all the other models, and the scores of 0.998 and 0.781 are absolutely high, too. The regression is most appropriate for the task of predicting construction costs based on the given data.

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

- Model Tuning (ML Pipeline #7)
  In this section, the selected regression of Xgboost will then be fine-tuned to achieve a better performance using grid-search technique. "Make a table, think of all the possibilities, and pick the best one." Xgboost has multiple hyper-parameters, and through the tuning technique, the optimized combination of those can be determined. The parameters chosen for grid-search are as follows:

  - Max depth, with values of 1, 2, 3, 4, and 5
  - N estimators, with values of 50, 100, 500, 1000, 2000, and 4000

  The matrix of R2 score is adopted once again to evaluate the model performances. The one with the highest score will be determined as the optimized model, and will be used for the final cost prediction. According to the following line chart and heatmap, one can realize that the highest score occurs with the combination of:

  - Max depth as 2
  - N estimators as 100
  - R2 score as 0.73

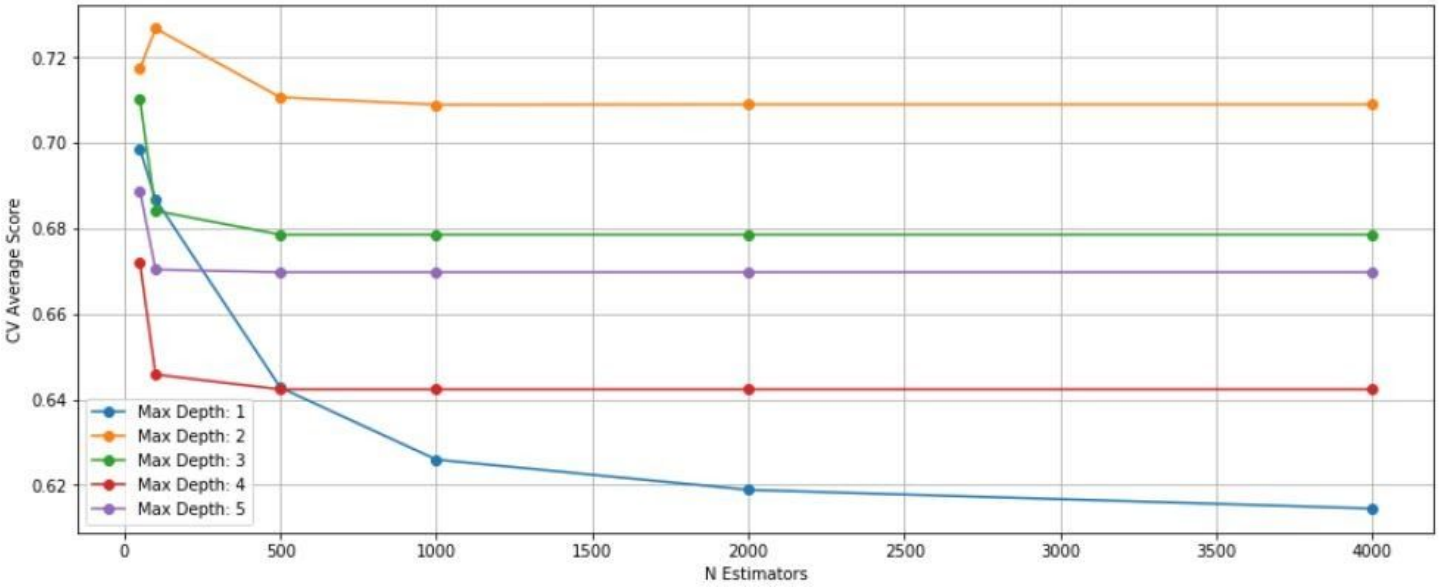

Figure 12. Grid search line chart, max depth vs. N estimators

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

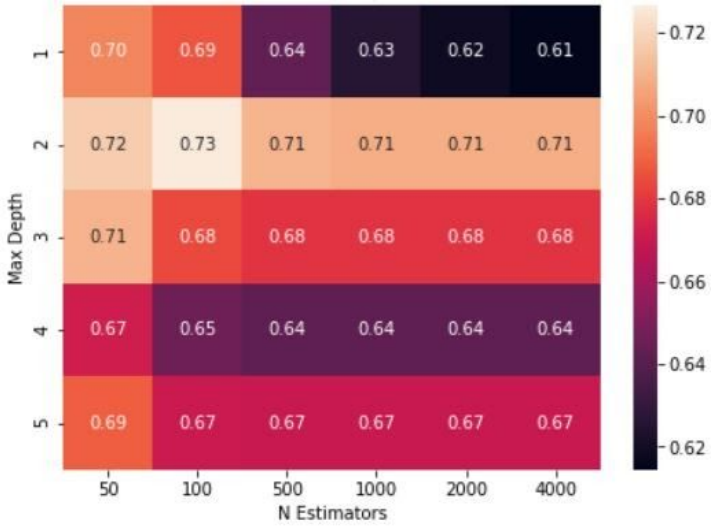

Figure 13. Grid search heatmap, max depth vs. N estimators

Results

Model Evaluation and Validation (ML Pipeline #8)

- Scores Before-and-After Model Tuning
  After choosing the best regression model and fine-tuning the selected model with its hyper-parameters, now is the time to evaluate the performance of the optimized model. The matrix of R2 score is used here, and the comparison of the scores between before and after model tuning shows as follows:

  - Unoptimized model: 0.7812
  - Optimized model: 0.9311

  One can easily realize that there is a significant performance increase (19.18%) between the two models. A great success is achieved through grid-search, and the R2 results further emphasize the importance of model tuning.

- Feature Importance
  By calling *feature_importance_* from Xgboost, the ranking and the weights of the features while making cost predictions will be listed out. One can then extract the most relevant features, remove the irrelevant ones, and train the model purely with those important features. By doing this, the duration of the training process can be decreased significantly since there are lesser features that need to be considered. However, as a trade-off, the performance matrix is expected to be lower, too.

  According to the following graph, one can understand that the top 5 most important features contribute almost 80% of the importance of all features existing in the dataset, which is 68, by the way. This implies that it might be appropriate to reduce the feature space for the purpose of simplicity. The R2 scores for both models before and after feature selection are listed below. This is a significant decrease (92.65%) in size at the cost of the drop in performance (11.19%).

  - Training on full data (68 features): 0.9311
  - Training on reduced data (5 features): 0.8292

Shih-Chi Liu
Hathaway Dinwiddie Construction Company
United States

Apparently, size reduction is not a necessity in our case while only having 112 data points as the training time is very short and is not an issue at all. However, what is noteworthy is that the dataset contains a significant number of features, and along with the increase of data points, the training time will increase quickly. When one reaches the point that the model takes a long period to train, that is the time to consider extracting features and to only train the model on those importances, even with a small trade-off of model performance.

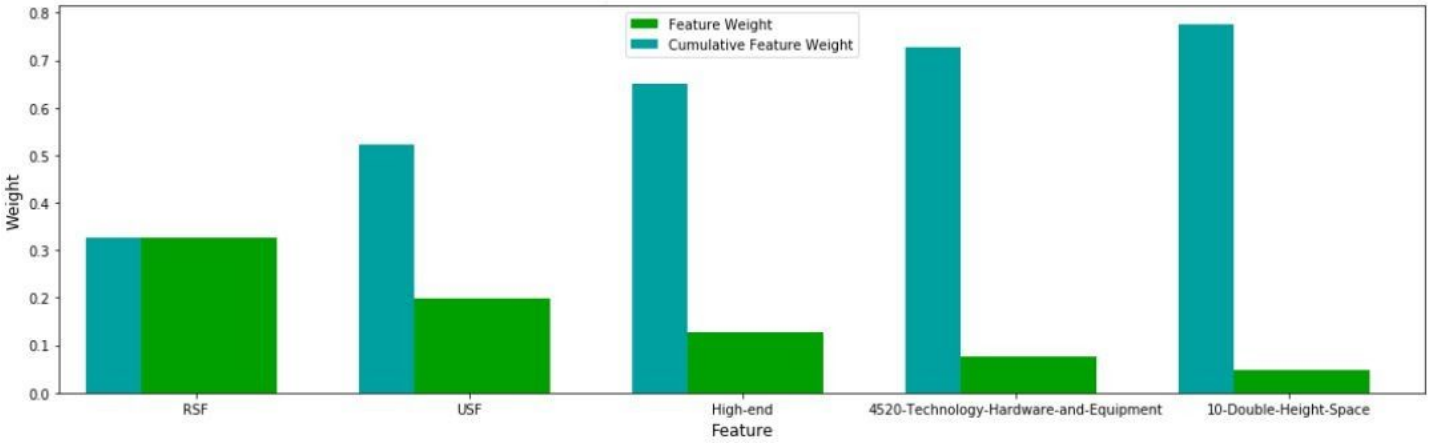

Figure 14. Normalized weights for first five most predictive features

Justification (ML Pipeline #9)

In this section, the optimized Xgboost model and its performance will be compared to the benchmarks established earlier in the project. The metric of R2 score is adopted here to evaluate the model results.

- Existing Workflow: Matric of Amount per Area

  Per the discussion in the section of Benchmark, the existing approach of filtering and averaging might lead to some severe overfittings because of the lack of data points, and the workflow should be a trial-and-error process. After several tries, I notice that if there are more than 1 criteria to be used for filtering, the chance of having only 1 or even 0 project data filtered increases significantly. Therefore, the decision is made to only use one of the most commonly used project characteristics at a time to filter the data and to retrieve the matric of the amount per area through averaging the target values of filtered projects. The R2 scores with different filters are calculated in our in-house database/Google sheet environment and listed as follows:

  - Filter with Client Type: 0.7940
  - Filter with Project Scope: 0.8579
  - Filter with Project Function: 0.7968

  Although they are not as high as the R2 score of the optimized Xgboost regression model at 0.9311, the scores calculated based on the existing workflow are not bad at all. However, what is noteworthy is that during the calculation, I find that there are still some cases only having 1 data point filtered. Plus, only one feature being considered will also cause the issue of overfitting. The existing workflow might seem to do a decent job by only looking at its R2 score; nevertheless, it is not a robust model at all due to its drawback of overfitting. Part of the reason is due to the data shortage, while the primary reason being its simple algorithm of true/false filtering.

- Simple Algorithm-based Approach: Linear Regression
  The R2 score of the linear regression model trained on the feature of USF is 0.4035. This is indeed a very low score compared to the optimized Xgboost algorithm. The graph below further reinforces the result as one can understand that the predictions are very off from the true values. Therefore, the justification can be made that these results and the solution are significant enough to have solved the problem posed in the project.

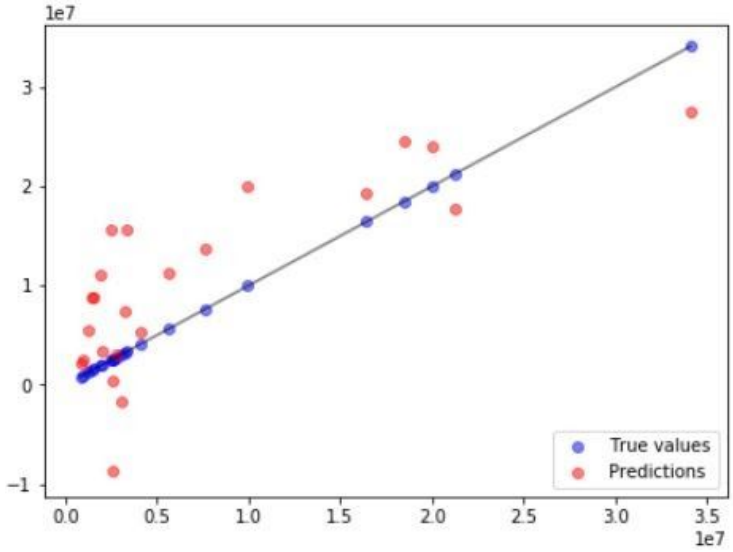

Figure 15. Linear regression, true values vs. predictions

Conclusion

Free-Form Visualization

All the free-form visualizations, including diagram, plot, chart, heatmap, snapshot, matrix, and et cetera that I considered meaningful, storytelling, and necessary are placed and discussed thoroughly in the corresponding sections earlier in the report. Please refer to the desired section for more information.

Reflection

- Interesting Aspects of the Project
  - Domain Background/Problem Statement
    As I mentioned previously, The AEC industry is not very good at embracing advanced technology and opening to new ideas. I have only been working less than 2 years, and as a newbie in the industry, I have an innovative mind and am keen to propose an alternative, perhaps a more robust solution to the long-lasting problem. I am thrilled that the prototype is created successfully, in which more opportunities are opened to me. The industry has been run purely by human experiences for so long, and I believe this is the exact reason for its slow development pace. A data-driven strategy could be a way out to break through the existing barriers and lead the industry to a new era.

  - Data Exploration/Feature Importance
    The project allows me to examine our current database, as well as its structure with a different perspective. We did not have a logical/mathematical way to evaluate the importance and the relationship of all the features before but simply manipulated them with our intuitions. With the help from numpy and

sklearn, all the data points are analyzed and displayed in a systematic manner. We can then determine which features are more relevant and which are not based on the analytics. After all, a feature number of 68 is not only too much for the computer but also for the human being to process.

- Benchmark/Justification
  Again, it is always interesting comparing the existing workflow and the proposed one. The process helps me to understand the strengths, weaknesses, opportunities, and threats (SWOT) for both of the methodologies. According to the justification, our current filtering approach actually works pretty well with a small number of data points. This does explain why the industry is not willing to/lacking the motivation to seek a more robust and trustworthy solution. However, it heavily depends on human experiences to pick the appropriate criteria and to minimize the occurrence of overfitting. Therefore, the results are not replicable and will vary based on who is the user. On the other hand, the algorithm-based approach ensures the prediction consistency and avoids overfitting through the much more complex and thorough calculation process. Furthermore, the technique of machine learning will truly shine and beat the traditional workflow when the database scales up in the near future and along with the development of the company/industry.

- Difficult Aspects of the Project
  The difficulties are basically the same for the following three topics: too many options out there and not sure which one to pick. Based on the research I have done, there are multiple approaches to handle even one issue, and each decision could potentially affect the final results significantly. After a long period of trial-and-error, I realize this is indeed an iterative process, and per my reviewer, "... This is what makes this a science." Indeed, all the feedback from the step of model evaluation and validation will provide some insights on determining the best combination, improving the predictions, and making us a better data scientist.

  - Data Preprocessing
  - Algorithms and Techniques
  - Free-Form Visualization

- Comparison between Final Model and Expectations
  I must confess that the results of the final model exceed my expectations, considering only the minimal data points are provided. An R2 score of 0.9311 is indeed very good. And, I really appreciate the insights given by the feature importance as the raw dataset contains 68 features, and 5 of them can represent 80% of the data. This indicates that our current database is probably not built in a cleanest way, and there are more rooms for improvement. Saying that, though the project is an exciting prototype and has successfully proved itself as a legit workflow, I personally think it should not be used in a general setting to solve these types of problems until having a big enough dataset. The algorithm-based approach can only be as good as the number of useful data points. Before that happens, human inputs/experiences must be added in to help with decision-making.

<u>Improvement</u>

- Machine-learning-related
  - Number of Data Points
    This is the most fundamental issue, and it will get improved over time naturally. I believe that along with the growth of the dataset, the algorithm-based approach will gradually achieve its maximum capacity, becoming one of the best workflows for cost prediction in the entire industry. For now, the model is not robust enough as the model's solution will be affected by tweaking the input values. However, the

robustness will, for sure, grow once the size of the dataset becomes bigger, and the model will be well generalized simultaneously.

- Feature Selection
  For now, the dataset contains more features than data points, and this can be a big issue. According to the proposal review, one of the potential solutions is to reduce the feature space and only keep the essential features. sklean also provides some built-in functions that are able to select the most relevant features, such as *feature_selection*[12], *SelectKBest*[13], and *SelectPercentile*[14]. More possible solutions are *likelihood encoding of categorical features*[15] or *smoothing*[16].

- Approach Selection of Data Preprocessing, Algorithms and Techniques, and Free-Form Visualization
  This potential improvement has been discussed thoroughly in the above section of Difficult Aspects of the Project. Please refer to the section for more information.

- Deployment-related
  - Cloud-based
    All the codes and the supporting files are saved locally on my personal computer now; however, the ultimate goal is to provide access for the entire company. Keeping everything on the cloud is a necessity. There are a lot of possible solutions/services on the market, including AWS, Azure, and et cetera.

  - User Interface
    The front-end must be built as the expectation of everyone in the company being able to understand the codes and to type in inputs is not realistic. There are multiple accessible ways to construct the user interface, including some customized APIs.

  - Integration with Other Platform
    We currently save all our data in the Google sheet environment, and we might migrate to a real database in the future when the number of data points grows. It is necessary either way to create a lived-link between the database/spreadsheet and the machine learning pipeline. Users won't like the idea of exporting data from other places and feeding it into the algorithm every time they run the model. Also, it will be great if the pipeline can be linked with some outer data visualization tools, such as Google Data Studio, Power BI, and Tableau for better graphs and charts.

[12] Scikit-learn, *API Reference*, https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_selection
[13] Scikit-learn, *sklearn.feature_selection.SelectKBest*, https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html#sklearn.feature_selection.SelectKBest
[14] Scikit-learn, *sklearn.feature_selection.SelectPercentile*, https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectPercentile.html#sklearn.feature_selection.SelectPercentile
[15] Eduardo Gómez, *Likelihood encoding of categorical features*, 2017, https://www.kaggle.com/tnarik/likelihood-encoding-of-categorical-features
[16] Olivier Grellier, *Python target encoding for categorical features*, 2017, https://www.kaggle.com/ogrellier/python-target-encoding-for-categorical-features