Domain Background

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 potential solution is Machine Learning Regression Model. The inputs of the problem are the cost-related project descriptions, which will be further discussed in the next 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.

Datasets and Inputs

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).[1]
    - *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.

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

- *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 as Facebook, banks as JP Morgan Chase, etc. 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.
  - *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, 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.

Solution Statement

The solution I am proposing here is Machine Learning Regression algorithms. According to the Scikit-learn algorithm cheat-sheet 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')

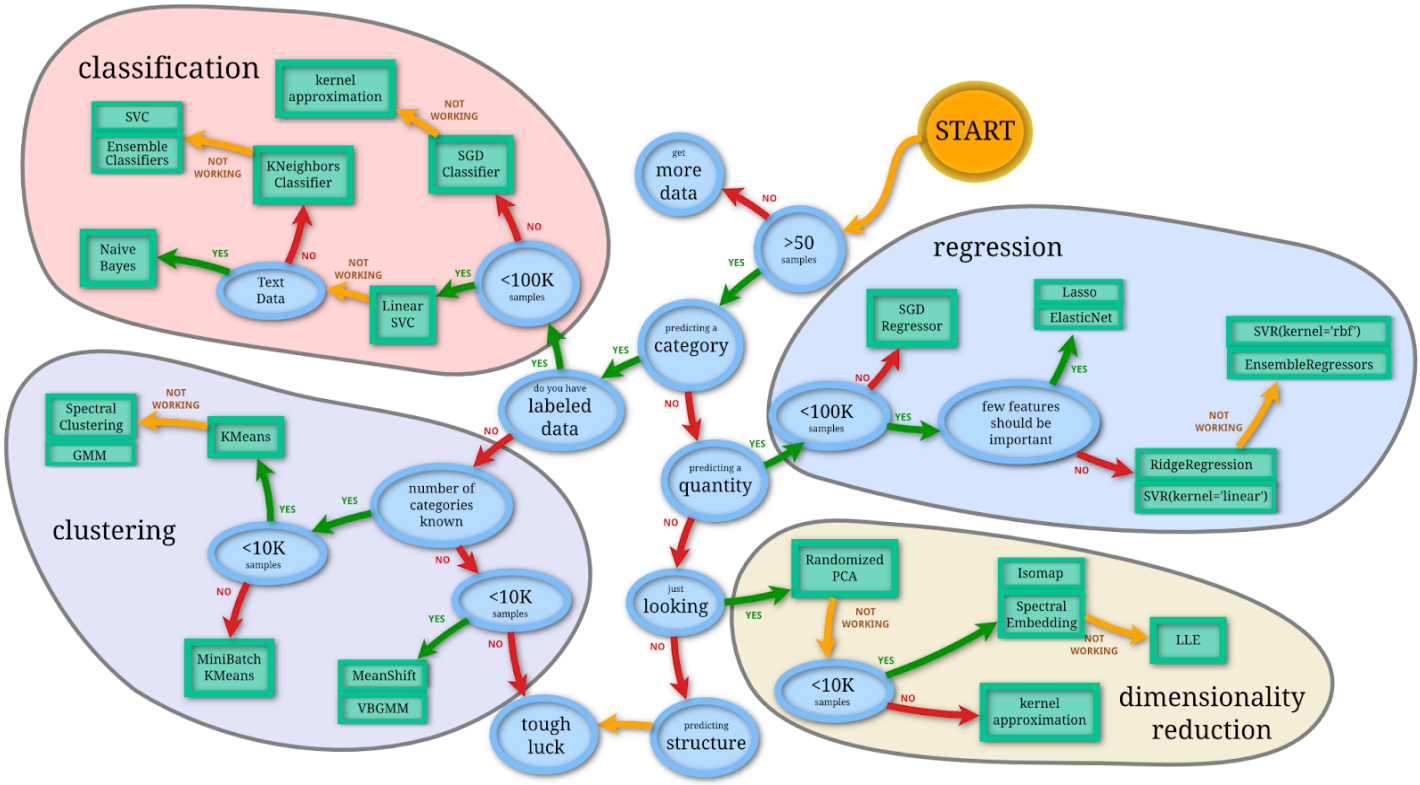

Figure 1. Scikit-learn algorithm cheat-sheet[2]

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

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

- K Neighbor
- Random Forest
- SVM

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

Benchmark Model

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 multiple the averaged number by the project size to retrieve the predicted cost.

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 becomes 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.

Evaluation Metrics

There are three evaluation metrics applicable to the above regression algorithms. All of the three will be used to evaluate the performances for both the benchmark as well as the Machine Learning models.

- *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.

Project Design

In this section, the project is broken down into 9 steps to illustrate the pipeline. Please also refer to the following diagram for the sequences and relations for all the steps.

1. *Data acquisition*
   The dataset is from HDCCo in-house database, containing 117 data points with 68 features and 1 target. Please also refer to the section of Datasets and Inputs for more information.
2. *Data preprocessing*
   - Removing irrelevant inputs
   - Transforming skewed continuous features
   - One-hot encoding
   - Dropping NaN and filling 0 for all blank cells
   - Removing outliers
   - Normalizing numerical features
3. *Data split*
   - Training data
   - Cross-validation/K-fold cross-validation
   - Testing data
4. *Model training*
   The training data will then be used to train various regression models in this step. Please also refer to the section of Solution Statement for more information.
5. *Model selection*
   Models will be fine-tuned in this step, and the following approaches will be used to select the optimized model.
   - Model complexity graph
   - Learning curve
   - Grid search
6. *Model testing/evaluation*
   Testing data will be evaluated by the metrics to determine the model performance. Please refer to the section of Evaluation Metrics for more information. The testing score will also provide an insight into how good the dataset is and whether the preprocessing steps are appropriate.
7. *Model publishing*
   Only the model passing the performance threshold should be published.
8. *Cost predicting*
   New project features will be treated as inputs in this step for the published model.
9. *Result*
   The predicted result will then be generated by the published model.

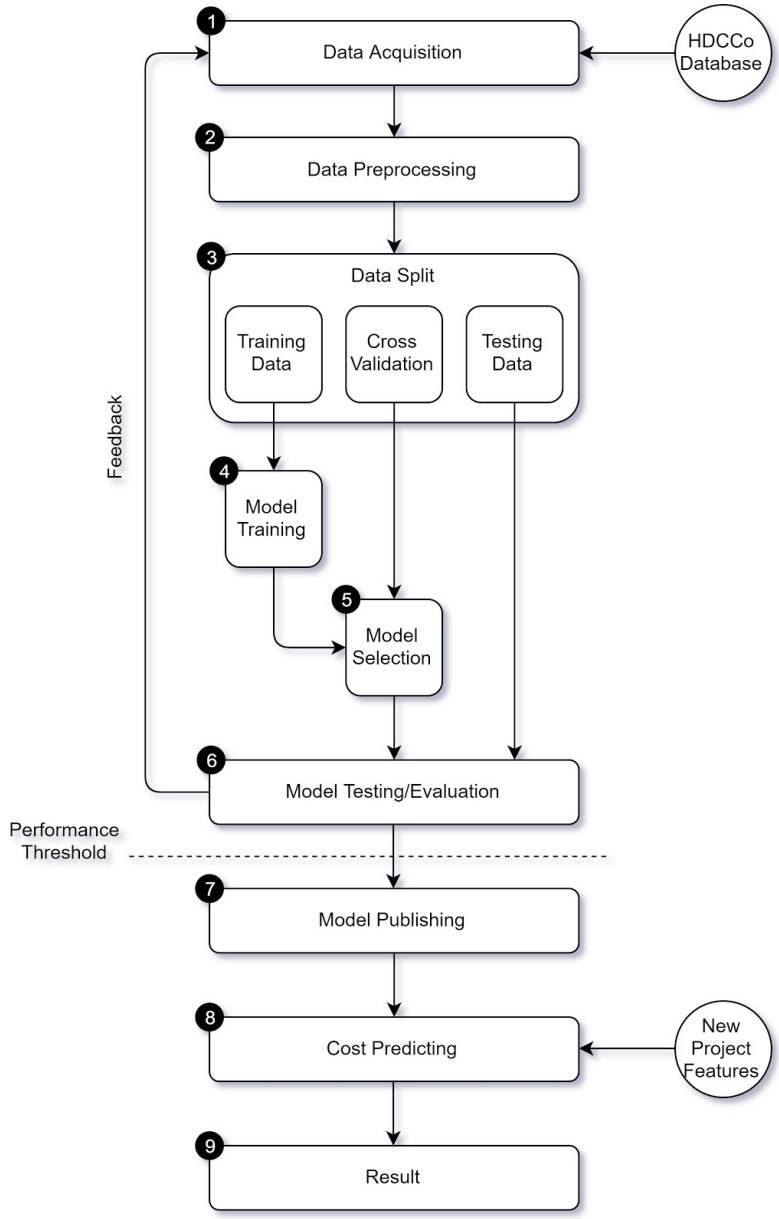

Figure 2. Machine learning model pipeline[3]

Concerns

There are some questions/concerns raised while writing the proposal, and they are listed below.

- Is it appropriate to have so many features (68 in total after one-hot encoding) while only have 118 data points?
- Is it fair to compare the benchmark model, which only accepts 3 inputs, with Machine Learning models, which contains 68 features?
- How to determine which data preprocessing steps should be taken and which should not?

[3] 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