Jeepal Patel  
December 2021

Construction Management Software

# Problem:

For most companies, predicting constructions timelines is always a difficult task. They mostly rely on user experiences, existing plans, blueprints, and estimators. To save construction workers’ efforts, the company’s financial planning, and every other factor, timelines play a vital role. Automating the prediction task, so that even a junior project manager could accurately predict timelines, would be fundamentally useful in future projects planning. Moreover, adding a user interface for the prediction would allow the application to be accessed by anyone anywhere, adding to its usefulness.

To solve this problem, a historical/raw dataset with project transformations, descriptions, and units completed within such a period would be required. Since many details go to construction projects, the dataset would need to be as detailed as possible.

# Methodology:

**Front & Backend Development:**

A solution to this problem would be a novel resource to all in the industry. To solve the problem at hand, we split up the project into major tasks, each one further subdivided into manageable tasks.

Below are the steps we followed to solve the problem and divided into sequential tasks:

1. **Identify Raw Data & Define the scope of the Project**

We started using the data provided by the client. Searching for additional data to add to

our database, did not yield any publicly available results.

Working with our clients and stakeholders, we defined the scope of the project. It would include developing an application to predict construction timelines. The application would need to be easily accessible anywhere, and simple to use.

1. **Data Exploration & Feature Engineering:** The historical data consisted of 40+ different transformations, 1400+ different activities, start times, end times, type of units, and the number of units completed in the period. No repeat values were found. Some zero values for the number of units completed were identified.

New features were created from the available data. The new features were completion

rates: 1) days per unit; and the inverse 2) units per day. This new feature would become

the target that we would focus on predicting.

1. **Data wrangling**Some activities had a completion rate of 0 days per unit. Since this task did not take any

time, these activities were removed from the dataset. Columns that would not be used

in the analysis were also dropped.

1. **Pre-processing data**

The text data was processed vectorized to allow for the computer to complete proper

analysis on the text, and to aid in finding predictions.

1. **Split data**

Data were split for testing purposes using an 80%/20% training/test ratio.

1. **Build the Model**

On the **Back-end part**, several machine learning models were constructed and tested to see which model would perform the best at predicting completion rates.

On the **Front-end part**, what did we use so that Machine Learning Model and our website can talk to each other? **1. API:** It is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other.  
**2. Flask:** It is a micro web framework written in Python.  
**3. Flask-Cors:** It is the flask extension for handling cross-origin resource sharing (Cors), making cross-origin AJAX possible.  
**4. Postman:** It is an application used for **API testing**. It is an HTTP client that tests HTTP requests, utilizing a graphical user interface, through which we obtain different types of responses that need to be subsequently validated.

1. **Evaluate the Model**

A random forest model was selected to be used for predicting.

1. **Front-end Development**

The number of pages included in our website is given below:

1. **Home:** It includes the introduction, and why we are creating this website, who we are, our location, client testimonial, and where to contact for more information.

Graphical user interface, website

Description automatically generated

Graphical user interface

Description automatically generated with low confidence

A picture containing logo

Description automatically generated

1. **About:** This page explains our background and why one can trust our services when they are planning for their company’s construction project. Also, this software provides clients with a wide range of services to cover all their needs.

Graphical user interface, text

Description automatically generated

1. **Services:** This page has our numbers of best services the software can predict and brief details about those services.

Graphical user interface

Description automatically generated with medium confidence

A picture containing graphical user interface

Description automatically generated

1. **Predict:** This is the page where user can enter their desired quantity for transformation and description they want to predict for their span of period. And when they click the ‘**Predict’** button, it will respond by using the **Machine Learning Model** which shows the numbers of quantity a product user wants to predict in each period, and it also shows the entered period is sufficient or not. Moreover, It shows the tentative end date of the prediction.

Graphical user interface, application

Description automatically generated

1. **Meet the Team:** This page has members of our team who were equally productive, responsible, and enthusiastic to create such software which can help in real-world to solve many construction management problems to save time and money.
2. **Contact:** Here users can contact and find the location for more information.
3. **Deployment**

A function that could be called was designed and deployed. This function had various

inputs and returned a text blob which could be displayed on the screen.

1. **Add Additional Features**

**1. Predict Unit Type:** An additional prediction using the same text inputs was designed to predict the type of units the task was undertaking.

**2. Error Handling:** The codes were adapted to handle some of the most common errors and give feedback to the user.

**3. Allow for Multiple Consecutive Tasks:** The solution was generalized which allowed for multiple activities to be passed to it. The solution then predicted start and end times for each consecutive task.

**4. Spell Check:** A spell check was implemented in the solution to allow for errors in the user input text string. An English dictionary, as well as the client dataset, was used to

create a complete list of words that exist, and the string could be corrected too.

**5. Search for String Matches:** A direct search for string matches in the dataset from user input was also implemented. This allows the user to see the range of potential values and errors in the prediction.

# Results:

The results from the machine learning model were satisfactory. Predictions could be made from any search string and results would be displayed.

Various metrics were calculated for the predictions. The range of values from 0.002 to 826 days per unit in the dataset resulted in a large error rate in the prediction. More work would be needed to reduce the errors.

|  |  |
| --- | --- |
| **Metric** | **Error** |
| Median Absolute Error | 47.1 days per unit |
| Mean Absolute Error | 126 days per unit |
| Mean Absolute Percentage Error | 537% |
| Mean Squared Error | 45,978 days per unit |

Results were also calculated for the prediction of unit type. This classification problem had the same input as the regression problem but had its issues. The target classes were not evenly distributed, with some classes in the training dataset only having 3 instances with others having 631. Attempts were made to under-sample and over-sample, but nothing seemed to improve the results. With an accuracy score of 99%, the model was left as is, even though the procedure for designing the model was not ideal since it was imbalanced.

The results from the user interface are shown below.

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Chart, scatter chart

Description automatically generated

# Conclusion:

Lessons learned while developing the model were:

1. How to deploy a model.
2. How the connection between front-end and back-end work and what functionality can be used and how.
3. Understand and learn the concept and functionalities of HTML, CSS, JavaScript, Postman, and many more.

# 