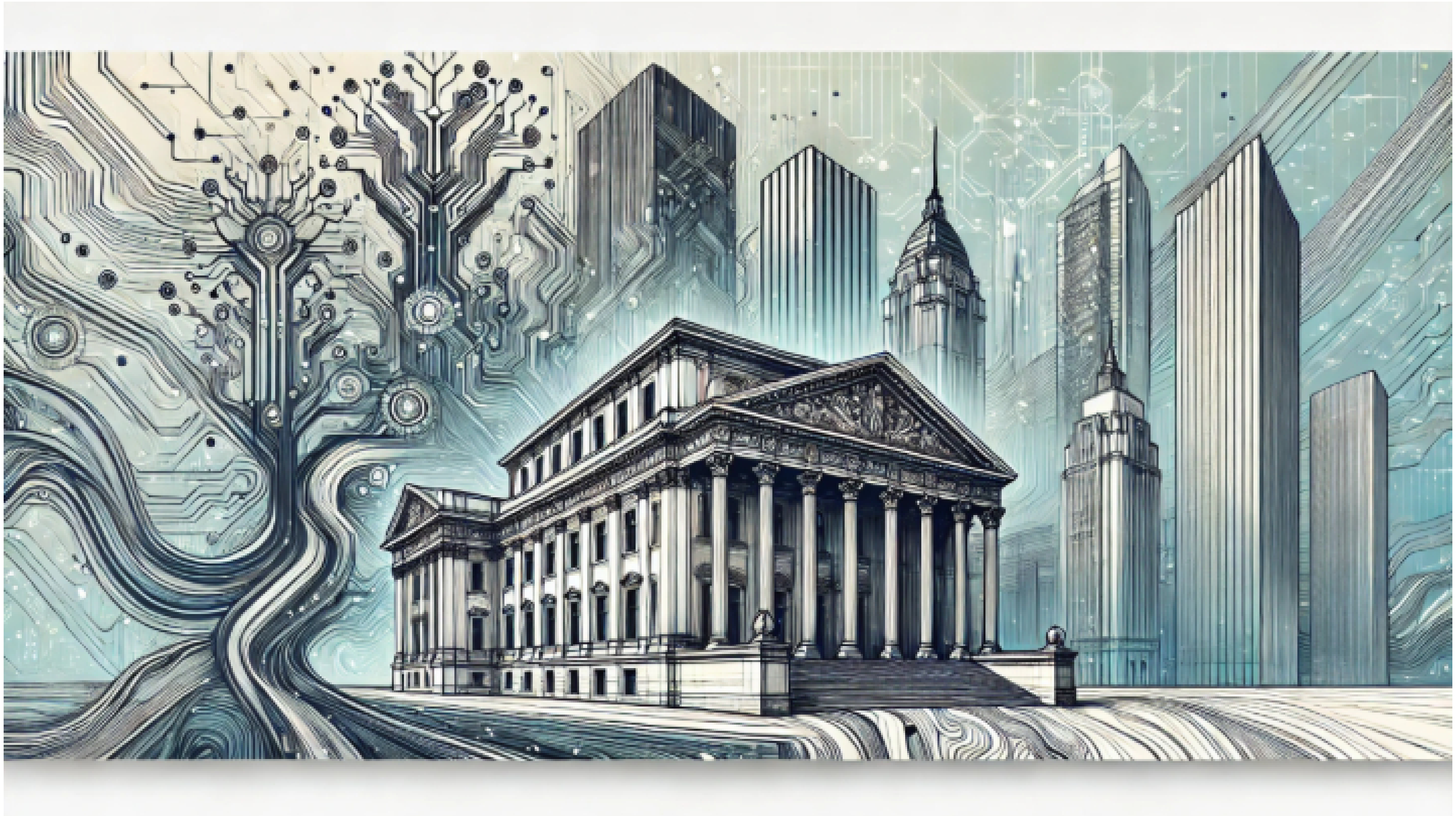


Image created with ChatGPT

Using Machine Learning to Predict the Next Command in Revit _ Part 1: Introduction & Index

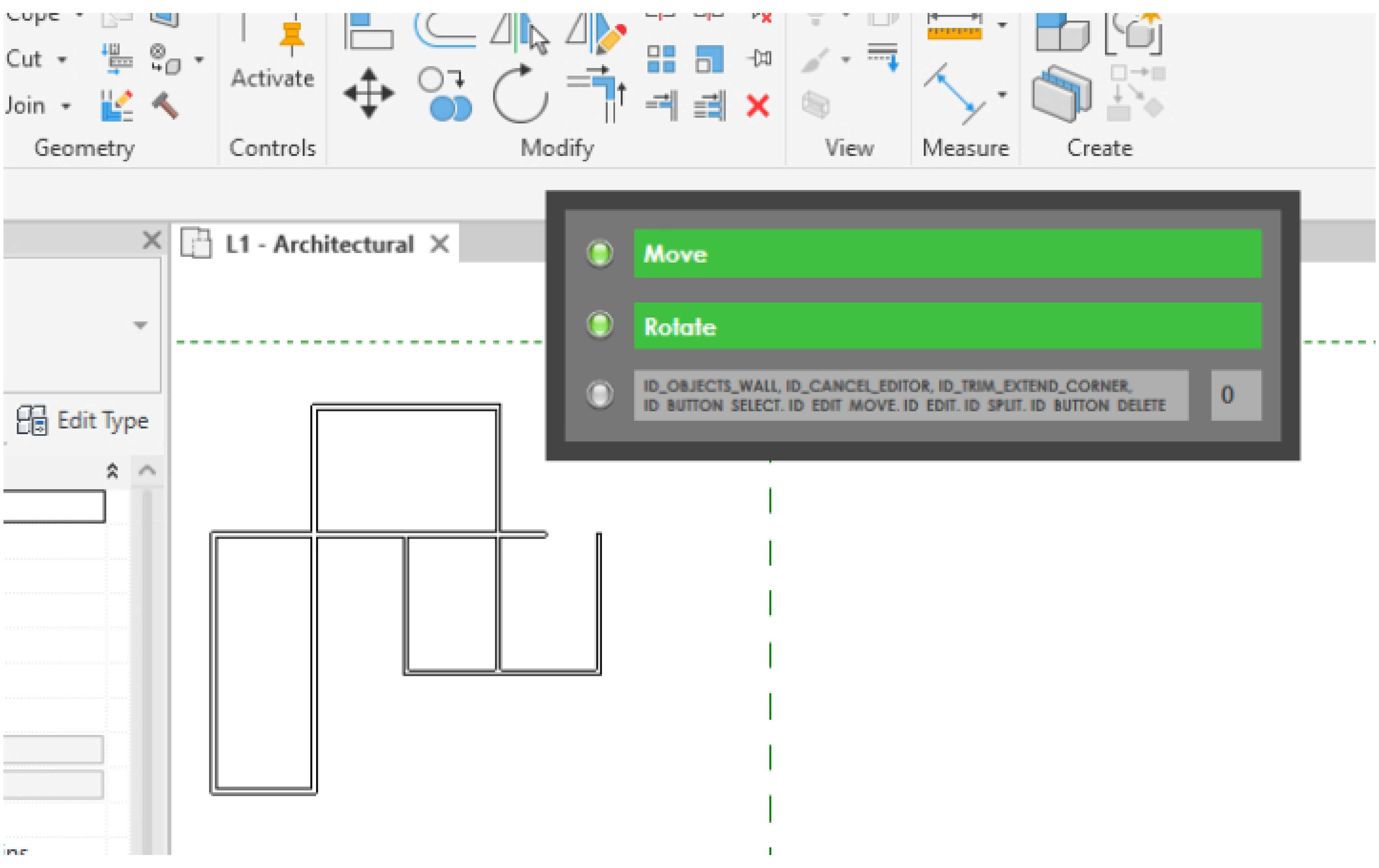
Let's BIM Together
728 followers

September 25, 2024

(Part 1 of a 7 series blog.)

Welcome to the blog where I journey into the creation of *AG Feeling Lucky*, a useless Revit plugin that predicts your next command based on the last four entered commands. I wanted to dip my toes into "AI", and this project became my "*Hello World*" for machine learning, *WPF*, and *Power BI*. ("*Hello World*" is the first application a student creates in a programming language that displays a simple message.) The name alludes to Google's "*Feeling Lucky*" button that used to be available on the search giant's website which would take you automatically to the first search result. This feature was deprecated because it's rarely the first search result users want. Similar to Google's "*Feeling Lucky*" button, *AG Feeling Lucky* hopes to achieve an impossible task: predict the next Revit command you want to use. Alongside this blog, there will be a complimentary YouTube video that demonstrates everything discussed. You can check that out here:

YouTube Video Link: https://www.youtube.com/watch?v=cJhZXAXUvWg



ID_OBJECTS_WALL, ID_CANCEL_EDITOR, ID_TRIM_EXTEND_CORNER, ID_BUTTON_SELECT, ID_EDIT_MOVE, ID_EDIT, ID_SPLIT, ID_BUTTON_DELETE

The process was anything but straightforward with plenty of trial and error along the way. For instance, I found myself analyzing data in *Notepad++* with the *Analyze* plugin to check entry frequencies instead of using *Power BI*. I also spent a fair amount of time manually tweaking parameters with a guess-and-check method instead of running optimization algorithms. These detours were part of the learning experience, and there are still some big gaps in the final product's logic. However, most of these detours will not be mentioned, and the story will be presented linearly. The aim of this blog is to demonstrate what is possible with all the tools involved in this Frankenstein.

Also, I want to mention that this 7-part series isn't just for those interested in machine learning. Even if building machine learning applications isn't a priority for a BIM Manager, tools like *PowerShell* and *Power BI* are worth exploring and are easier to learn than ever thanks to *ChatGPT* and similar LLMs. I struggled to determine which demographic would enjoy reading this. My best guess is a bored, curious AEC technologist with an understanding of the *Revit API*.

Going back to that BIM Manager, they could using *PowerShell* remotely pull Revit journals from a user's machine without disturbing their work, or even push new custom Revit settings to all users by modifying their *Revit.ini* file. *PowerShell* allows commands and scripts to be executed remotely, bypassing the Windows GUI. It is also a powerful tool for running software deployments as it allows scripts to be executed on multiple machines simultaneously. Here, I will be using *PowerShell* to manipulate text files.

As for *Power BI*, it's an incredible tool for data analysis and modification. It's similar to Excel with formulas and charts but conceptually akin to Revit schedules. Just as a Revit schedule provides a different view based on grouping, sorting, and filtering, *Power BI* "steps" your data to show various abstractions.

The project was developed in *Visual Studio* using *C#* and *ML.NET* because I spend most of my coding time in the *C# Revit API* making it a natural starting point. Plus, deploying *ML.NET* turned out to be much easier than other alternatives I considered, such as *PyTorch*. Although I initially aimed to avoid using *Python*, discovering what *PyRevit* & *Optuna* can easily do made it too tempting not to explore. I'll dive deeper into that in Parts 2 & 5 of this blog.

The final product is a Revit plugin that displays the top two command predictions to the user, then lets the user execute either of those. There's also a separate console application that can train, predict, and analyze the model in various ways.

To make this information more digestible, I've broken the blog into seven parts:

Part 1 - Introduction & Index

Part 2 - Data Preprocessing Using PowerShell & Python with PyRevit

Part 3 - Data Analysis Using Power BI

Part 4 - Model Trainer Application Using C#, ML.NET, PowerShell, Notepad++

Part 5 - Parameter Optimization Algorithms Using Python with Optuna

Part 6 - Revit Plugin Using C#, ML.NET & Revit API

Part 7 - Outro (Canva flow chart)

Download AG Feeling Lucky Plugin for Revit & Trainer/Analysis Console Application here:

https://letsbimtogether.com/blog.html

Go to the next part