Tim Steed

MSDS 434 Fall Quarter

Final Project

The purpose of my project was to build a replacement for a service that I currently pay for called ‘ClickInvest’. Myself and my brother, outside of our w2 jobs, try to purchase, rehab, then sell a few distressed single family homes per year. Our goal is one per quarter, but since the housing market took off in mid 2020 it has been increasingly difficult to find properties where the numbers make sense to take on the risk. Our solution was to subscribe to a service called ‘ClickInvest’ which, for an annual fee, accesses MLS data through the MRED API, programmatically determines which new listings are likely to be investment properties, does some basic calculations like the cost of carry (determined by the purchase price, construction costs, and users borrowing rate), after rehab value (I assume determined by a regression model, but they don’t provide their methodology), and ROI (determined by the previous metrics). Based on minimum thresholds for ROI that the user provides to ClickInvest and filtered to user provided municipalities of interest, they send you emails within minutes of a listing agent submitting a property to the MLS. The idea being that a user can have a significant speed advantage against other rehabbers that are finding properties in more traditional ways, and if they are confident enough in the numbers submit offer through ClickInvest’s website immediately. One obvious benefit of building something similar myself would be avoiding the annual subscription (which is around 2500), but another perk would be making it more customizable. For instance, their projected rehab costs seem to be based largely off a price per square foot to rehab metric (ie if a property is 1500 sq ft and their data suggests rehabbing a property in the Chicagoland area is roughly $65/sq foot they will suggest budgeting 97.5k for rehab work). In reality, all lenders require you to provide a scope of a work projection when taking out a construction loan that itemizes how you plan to spend their money, and I have done enough projects to know how much specific tasks (for example tearing off a roof, replacing a furnace, putting in new flooring, etc.) actually cost in a much more accurate way than just a simple by the square foot estimate. Actually replacing ‘ClickInvest’ will require me to build significantly on what I started here (for instance, I will need to get a license and register with the Chicago MLS so that I can access the same data that they have), but what I have attempted to do here is a start. My project has two github branches, a dev and prod branch, with pointers in the GCP aimed at them, to detect any changes in repository (I plan on doing daily batch runs once I have MLS access), push updated csvs to the repositories with new listings, use the regression model I had Vertex AI build to predict a property’s value (for the project I used a regression model, moving forward I may change to a clustering one), use BigQuery to filter the data to zips or municipalities that I am interested in working in and group by those municipalities or zips, pick out the properties where the predicted price less the actual price is largest, and email me those properties daily.

I do not have CircleCI implementation set up for this project, as it was not a requirement for the class, and the triggers in GCP pointing at my github repository will serve as the deployment signal for the time being, but down road for future iterations of the application, I will probably utilize CircleCI because a) it is a tool I am currently pretty unfamiliar with and would like to learn and b) my brother is a programmer by trade, and although I would anticipate we won’t be making frequent code changes, the ability to have a shared repository where we can both make and integrate updates when necessary without risking destabilizing a working product is appealing. For the time being I think the majority of the functionality provided by CircleCI can be supported using Github. The core uses of CircleCi are building pipelines that are used for building, testing, and deploying a project, with a pipeline being a set of instructions that CircleCI will execute every time a line of code is changed. Jobs within these pipelines are the specific tasks for the building, testing and deploying an application. You can configure CircleCI to send email notifications when an execution is complete to notify you of success or failure of your jobs. The idea is that it provides an environment where developers can frequently add new features without comprising the stability of your project. Using Github we can create separate branches with separate triggers for the build, test, and deploy environments, so that, for example, some code is changed in the build branch, it only triggers a run of that branch and does not jeopardize a failure in the production branch if the changes are bad. I suppose there is a risk of human error using Github, where a developer can inadvertently change code in an incorrect branch, but if the only goal is replicating CircleCI’s separate environment functionality it can be done via Github. Additionally, the Google Cloud Platform offers several tools for process automation within data pipelines that can replicate CircleCi’s functionality. I personally have not had the chance to use the Google Kubernetes Engine yet, but I know it exists and can be used managing, scaling, and deploying containerized applications.

For this project I elected to use the GCP over AWS or Azure, primarily due to the ease of use of the GCP. The majority of the utility provided by the GCP that we explored in this course was fairly easy to implement, and the dashboard is really easy to navigate. It may be naïve, but I was not really concerned with any idiosyncratic risks that one cloud provider is exposed to that the 2 others mentioned are not. The organizations behind them are three of the largest and most heavily resourced on the planet and it is difficult to imagine a scenario that would pose a threat to one’s stability that would not be present at the other two (although anecdotally we use AWS at my employer and in December of 2021 it went down for a few hours leaving many of our employees unable to do their tasks). I believe there are differences in the number of regions and zones supported by each service, so perhaps at really large scale there could be differences in performance that would make one platform a superior choice to the other two, but for the purposes of our project I don’t think that will present an issue. AWS provides the largest number of services of the three (GCP provides the least), but there are hundreds available in each platform, and without knowing them all and what each they do it’s difficult to say whether the sheer volume of services provided would make a difference, particularly on a relatively simple project like the one we are building. In looking at an overview of some of the specialized services provided by AWS vs GCP vs Azure, one note that I found interesting was that AWS supports robotics (AWS Robomaker) and game development (Amazon Gamelift), while the GCP doesn’t offer anything similar. While those specialized service are probably of interest to a lot of people deciding between platforms, they aren’t required for our project so I will go with the familiarity and ease of use that the GCP provides. In fact one of the criticisms I read of AWS is that it can tend to overwhelm new users with the volume of service options.

Similar to the ease of use advantage that the GCP provides, BigQueryML provides an intuitive, easy to use ML tool, that’s main selling point is that it democratizes ML model building without having to write code (other than SQL). This presents a pretty large advantage to those who lack programming skills, but are still interested in or require predictive models. An analyst competent in SQL but without exposure to, say, Python could start model building day 1 as opposed to taking the time to learn Python syntax and all the various ML libraries first. Not only does this help those without programming skills achieve ML solutions, but even for those of us who do have experience programming it still should speed up the process of model building significantly. I was really impressed by the ease at which the GCP made it to build models for audio and visual files. In MSDS 420 we spent an entire quarter learning how to develop image classification programs, compared to the module for this course where we were tasked in doing that, the only time consumed was that of the training. The main disadvantage as I see it for BigQueryML is the cost of use. For the final project to train my model in Vertex AI cost $70 for a regression model on a dataset that was not very large. It is easy to imagine how those type of costs would spiral out of control if you were doing something that required you to frequently train new models. Although the documentation in the GCP is made readily available in the platform, one other issue I had in Vertex AI was that I was getting fairly opaque error messages when my model training failed, and it was somewhat difficult to pinpoint what the issue was. I could see this being a point of frustration if building these models was crucial to your livelihood and not for coursework.

Overall I really enjoyed learning about and utilizing the Google Cloud Platform for the purposes of this application. The services it provides are more than adequate for the needs of this application. The tools are all provided within the platform to deliver a ‘turn-key’ application that will provide meaningful business aid immediately. In order to that really all I need is access to higher quality, more relevant data than is available to me at the moment.