Daniel Marchese  
Portfolio Project

**Introduction**

To understand the business problem, I must describe the context in some detail. This problem originated from a call center for bill collections at a large financial company. Agents work independently and at a high capacity with little oversight. This is accomplished by incentives and the consequence of loss of incentive. Agents use a phone system that uses different phone states for different tasks. Agents are incentivized to collect, and this is mainly done by outbound work. There is some opportunity for inbound work, which is much more potent for collecting and therefore is naturally highly incentivized. To do inbound work an agent must change phone states to a state that prevents outbound work, so the cost is a loss of outbound work. Unfortunately, it is very difficult to know when the opportunity for inbound work will be available for agents on an individual basis. The result is that agents will waste time in the inbound state when the inbound work is not yet available for them. The goal of this project is to build a model that will predict when inbound work will be available for an agent.

The base model, and the currently recommended model, is a naïve model (lag 1), agents can track this themselves. Using data from agents already using this naïve model would just support the naïve model, right? This would be true except in practice the naïve model is ineffective, and agents are not allowed to stay in the inbound state for too long, over 3 minutes is discouraged. After 3 minutes they go back into outbound work and wait a personally decided period before trying again. In addition to that, there is far from strict adherence to the naïve model. See the autocorrelation plot in Figure 1, the only hint of significance is at lag 1, using a lag model without accounting for other features does not appear promising.

Graphical user interface, table

Description automatically generated

Figure : Auto-Correlation Python output

**Data Preparation**

The data are made of two timestamps, event start and event end. Event start being when the clock would start to measure elapsed time and event end is when an agent receives inbound work or clocks out. The goal is to predict the time delta between event start and event end. The currently available 3 weeks of data is plotted below in Figure 2.

Chart, line chart

Description automatically generated

Figure : Time delta by Date/Time

The natural conclusion is that more data will be helpful. For example, we’re not able to take advantage of any monthly patterns because a full month of training data is not available. The goal is to improve on the current lag 1 model, these data are sufficient to start trying some new models. The first two weeks will be used as training and the last week will be for testing. The weeks are discernable in our model from the long stretches of no events, those are the weekends.

For all models event end is converted into time delta by subtracting event start from event end. For time series models the time start is kept as a datetime value. For machine learning models the event start is broken into features:

* Minutes (this is time of the day as an integer)
* Day of week (categorical)
* Month of year (categorical)
* Week of month (categorical)

Note that these features fully capture the date information while avoiding multicollinearity. I am concerned that the relationship between integers is not appropriate to represent time of day, so I will try this against a categorical variable of the hour of the day.

Our target variable follows an approximately normal distribution, see Figure 3. There is noticeably a cluster of events occurring near 0. I’m not concerned about this because our target cannot be negative so the distribution is naturally limited by the floor. Also there may be a reason that is causing these low values that our model will pick up on.

Chart, histogram

Description automatically generated

Figure : Event Time Delta Distribution

**Models**

*Naïve (Lag 1) Model*

Chart

Description automatically generated

Figure :Lag 1 Model

Root Mean Square Error (rmse) gives us a usable value in minutes and will be used to compare models. Let’s put our base model on the table.

|  |  |
| --- | --- |
| Model | Test RMSE |
| Lag 1 | 50.69 |

The next model tested is an ARIMA model, this is appropriate because our data is a time series and lacks seasonality at this point. This model, and all tested models are tested with rolling forecasting. A single prediction is made and then the actual value is added to the training data and the model is retrained to make the next prediction. This is exactly how the model will be implemented.

*ARIMA*

The optimal parameters for this model are p=2, q=1, d=0. This model is a contender and could be the best model based on RMSE in the future with more data. When seasonality becomes available I will try a SARIMAX model too.

Chart, line chart

Description automatically generated

Figure : ARIMA Model

|  |  |
| --- | --- |
| Model | Test RMSE |
| Lag 1 | 50.69 |
| ARIMA (2,1,0) | 38.42 |

*Rolling Linear Regression*

Linear regression forecasting is accomplished by converting the date time values into features as noted in the Data Preparation section. Using categorical hours instead of integer minutes returned much better results and only the model with hours as a category is described below.

Chart, line chart

Description automatically generated

Figure : LR model

|  |  |
| --- | --- |
| Model | Test RMSE |
| Lag 1 | 50.69 |
| ARIMA (2,1,0) | 38.42 |
| Linear Regression | 35.97 |

The residuals of this model are plotted in Figure 7.

Chart, scatter chart

Description automatically generated

Figure : LR Model Residuals by Actual Values

The green lines are between -30:30, I would consider this a good range to fall in for now. There is some trend to our residuals that the model couldn’t pick up on. This shows that the model can be improved with more features, hopefully finding a feature that can pick up on this trend. I suspected the length of the previous inbound work may help with predicting the very low values that were missed but this had no effect. Notably the re

**Conclusion**

The Rolling Linear Model should be implemented for the time being, it has a 14-minute improvement on the currently advised naïve model. It’s likely that our model can improve with more features based on the trend in the residuals.