

# Applied Analytics Practicum Final Report

OneBlood Project, July 19, 2020 – Arthur Wang

## Summary and Results

At any given point in time, the expected supply of donated blood is uncertain due to the voluntary nature of donation. For OneBlood and the hospitals to which it distributes blood, this uncertainty creates a planning problem. I addressed this problem using two approaches:

1. Fit a time series model using Facebook's Prophet model to forecast aggregate donations over a 30-day period
2. Fit a gradient boosted decision tree (GBDT) model using XGBoost to segment donors and predict probability of donation over a 30-day period

Results from my work include:

1. Prophet time series model with components for trend, holidays, weekly seasonality, and yearly seasonality
  - o Median daily mean absolute percentage error (MAPE) of 14.49% for forecasts over a horizon of 30 days
  - o Test error of 2.36% when forecasting total donations for a holdout time period
2. XGBoost GBDT model with 100 estimators, max tree depth of 5, and learning rate of 0.5
  - o Precision of 0.68
  - o Expected value prediction error of 0.019% using a test dataset
  - o Expected value prediction error of 5.5% using a holdout time period
  - o 3-fold cross validation fitting time of 5.7 minutes on a 1.4 GB dataset compared to over 35 minutes for other ensemble classifiers

## Insights

The main insights from my work are:

- A time series forecasting approach can accurately predict aggregate donations for a 30-day window, as well as generate prediction intervals to capture uncertainty
- A classifier capable of assigning class probabilities can correctly identify donors with the greatest **relative propensity** to donate, compared to other donors
- After segmenting donors into deciles ranked by propensity, it is possible to generate a fairly accurate prediction of each decile's expected number of donations using a simple expected value (EV) calculation:  $\sum P$  or  $\bar{P}N$

Additionally, I was able to characterize donors with the highest probability of donation again:

- The most likely donors have donated more recently, more frequently, and for longer
- Donors whose last donation was in a center are more likely to donate
- Donors who donate more frequently in centers are more likely to donate
- Donors who donate platelets more frequently are more likely to donate

These findings are further detailed in the "Predicting Probability of Donation" section below.

## Background and Objectives

OneBlood is a non-profit organization that collects blood donations and distributes blood products to health care providers. Unlike a manufacturer or distributor, their main “product” cannot be procured or produced at will – instead, they must find willing donors and get them “in the door” to donate blood. This introduces uncertainty with respect to their inventory of blood products; this uncertainty propagates to health care providers downstream who use OneBlood’s product, which makes planning more difficult.

The objectives of this project are to:

1. **Predict “inventory practicality”:** estimate the expected supply of blood products available over the next 30 days
2. **Guide direct marketing activities:** identify past donors who are most likely to donate again so that scarce marketing resources can be applied efficiently

## Data

Project participants were given a transactional dataset comprising individual donor registrations. A “registration” is defined as the first step in the donation process, and indicates intent to donate. For the purposes of this project, a registration is counted as a donation.

The dataset consists of the following fields:

- **Random\_ID:** an anonymized ID for a given donor
- **RegistrationTime:** the date and time of registration
- **Outcome:** the outcome of the registration – e.g. Registration, Donation, Deferral, or Incomplete
- **DonationType:** the donation procedure
- **Donation Location:** the type of location at which the donation took place – e.g. Mobile (donation bus) or Center (a physical “brick and mortar” location)

## Methodology

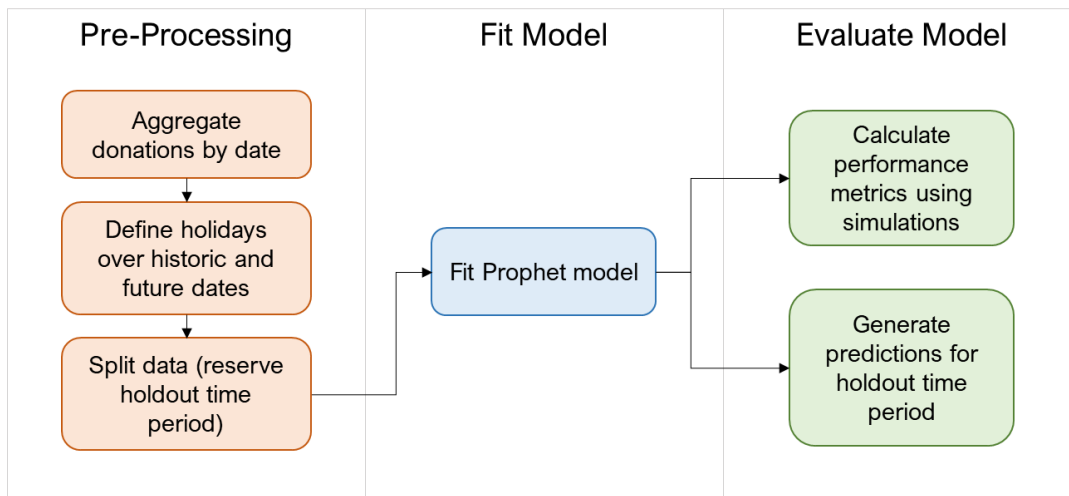
Given the stated objectives, I separated the problem into two parts:

1. Forecasting total donations over a 30-day period
2. Predicting probability of donation over a 30-day period and understanding characteristics of donors most likely to return

Below, I describe in detail my methods and findings for each part.

## Forecasting Donations

Forecasting donations answers the top-level question: “How much supply can we expect over the next 30 days?” I visualize my approach to this part of the project below:



**Figure 1:** Forecasting approach

### Pre-Processing

Facebook’s Prophet Python package (**fbprophet**) requires a dataset comprising two columns:

- **ds**: a date
- **y**: the variable

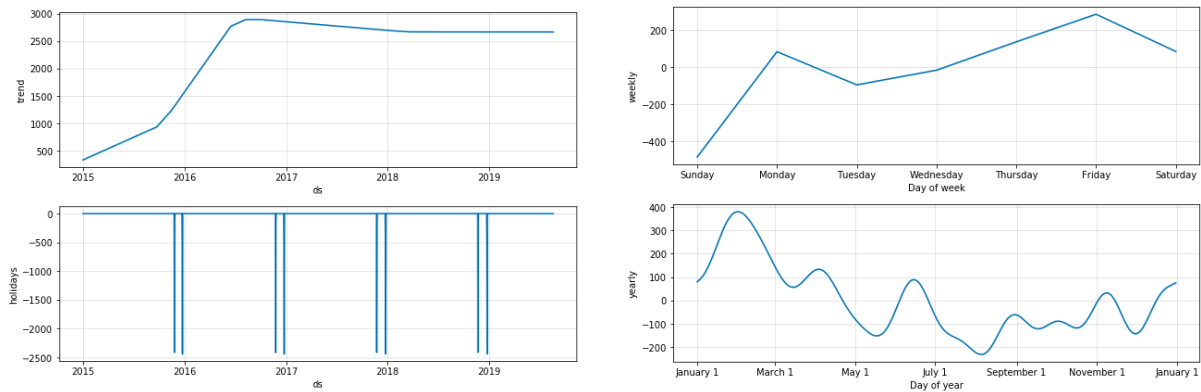
First, I converted **RegistrationTime** to a date (**RegistrationDate**) and aggregated the transactional dataset to record daily (**ds**) donation totals (**y**).

Next, I manually defined a set of dates for Thanksgiving and Christmas in the years 2015 – 2019. Prophet supports the addition of a holiday component to the additive time series model, since “their effects are not well modeled by a smooth cycle” (Taylor SJ 2017). During exploratory data analysis, I observed that these holidays showed a substantially lower number of donations compared to other dates.

Finally, I split the data into a training dataset and a test dataset by holding out the data for the final 30-day period in the original dataset: 7/22/2019 – 8/20/2019. This allowed me to evaluate aggregate prediction error for a 30-day forecast.

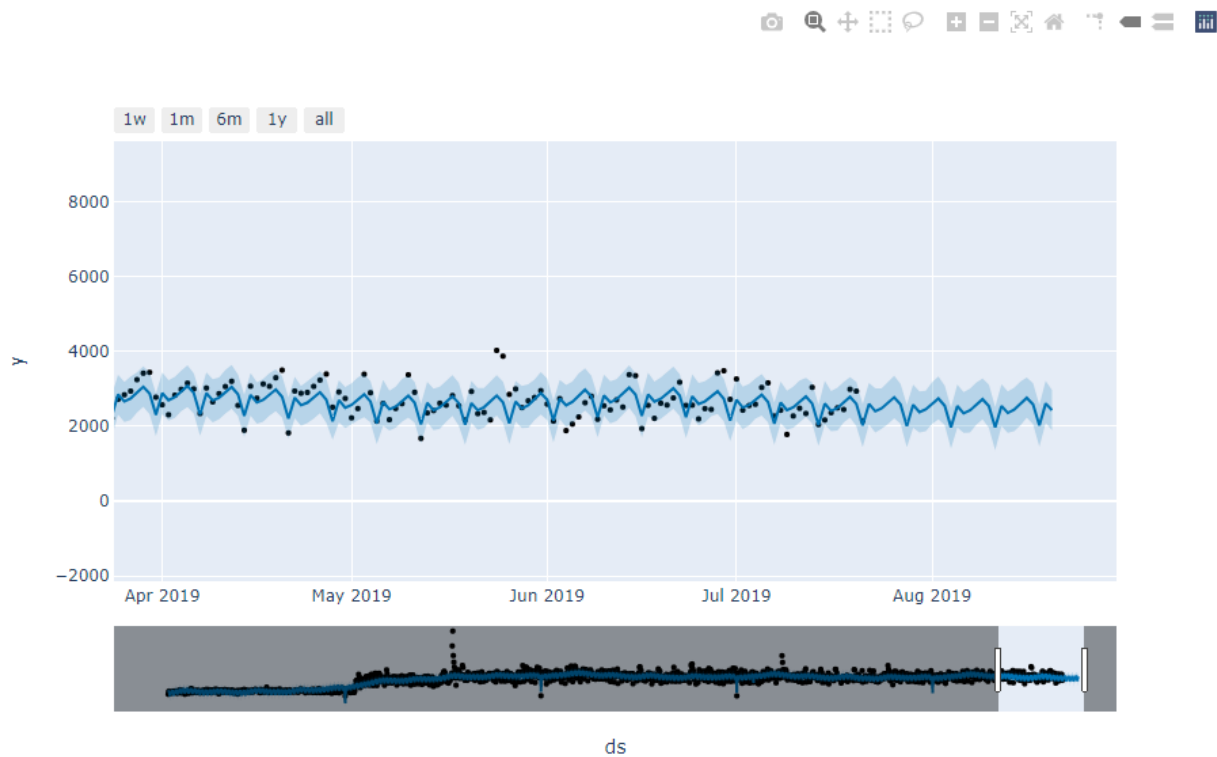
### Fit Model

After pre-processing, I fitted a Prophet model with components for trend, weekly seasonality, yearly seasonality, and the holidays defined during pre-processing. The model generates forecasts at a daily level.



**Figure 2:** Time series forecast components

Prophet also has methods for generating interactive plots of forecasts, allowing the user to view daily forecasts, prediction intervals, and actual historical values.



**Figure 3:** Interactive plot using `plot_plotly` method

## Test Results

To evaluate model performance, Prophet simulates historical forecasts by establishing rolling “cutoff points” in the history of the data. For each cutoff point, Prophet generates a forecast for the specified horizon – in this case, 30 days – using only data prior to that cutoff point, then compares forecasted amounts to actual amounts (Taylor SJ 2017). This method is roughly analogous to cross-validation.

Using Prophet's **diagnostics.cross\_validation** method, I generated an estimate of my model's performance for each day in the forecast horizon. The full set of metrics is included in the appendix as Table A1. The summarized metrics include:

- Median daily mean absolute percentage error (MAPE) of 14.49%
- Median daily mean absolute error (MAE) of 376.96

### Evaluation on Holdout Time Period

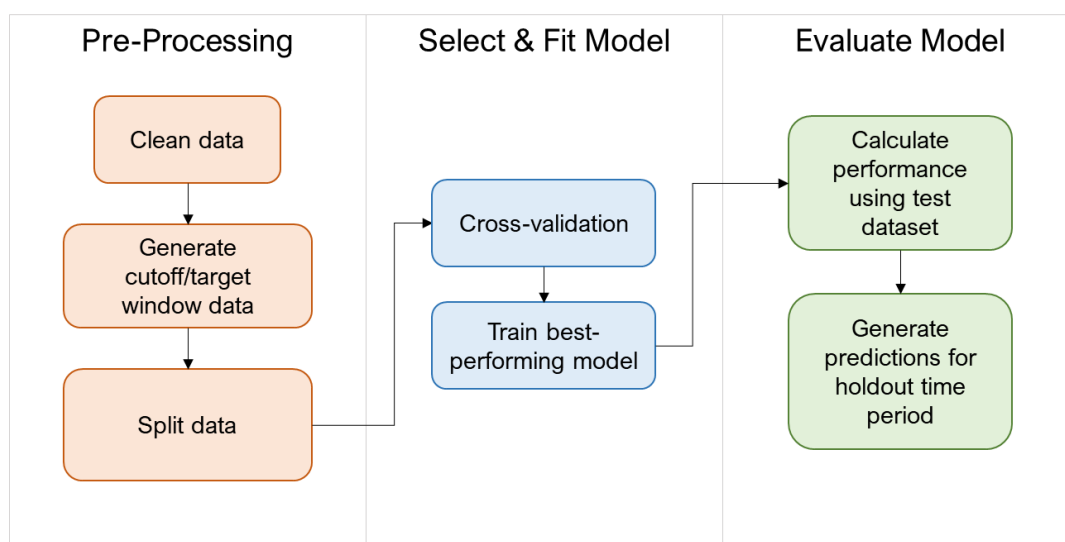
In addition to evaluating performance using simulated horizons, I kept a holdout time period (7/22/2019 – 8/20/2019) to use as a straightforward demonstration of performance, as well as to demonstrate how to “roll up” daily forecasts into a 30-day forecast with prediction intervals.

One approach to aggregating daily predictions is to simulate daily forecasts over a given horizon (e.g. 30 days). For each simulation, take the sum of all forecasts; the mean of those sums is the predicted total, and prediction intervals can be determined by taking specific quantiles from the simulated sums (Hyndman 2018).

After fitting the model, I generated a contiguous set of 30 dates spanning from 7/22/2019 – 8/20/2019 and used Prophet's **predictive\_samples** method to generate 1,000 simulated future sample paths (Letham 2018). Applying the method described above, I calculated a predicted 30-day total of 74,109.69 donations with an 80% prediction interval of (71,067.91, 77,188.17). The true total for this time period was 75,898 donations, giving me an absolute prediction error ( $|\hat{Y} - Y| / Y$ ) of 2.36%. This demonstrates that time series methods are an effective method for predicting inventory practicality.

### Predicting Probability of Donation

Predicting individual donors' probability (propensity) to donate can guide direct marketing efforts toward targets who are most likely to generate value (donations). It can also be used to predict expected donations from specific segments of the donor population. Below is an illustration of my approach to this part of the project:



**Figure 4:** Probability modeling approach

## Pre-Processing


The original dataset contains 17 records with missing data and many records with incorrectly formatted timestamps. As my first step, I removed the records with missing data and cleaned up the timestamps.

Next, I transformed the transactional data into a format suitable for propensity modeling. I selected a contiguous set of 30-day periods (“target windows”), starting from 7/22/2019 and working backward. I chose the day prior to each target window as a “cutoff date”; for each target window, all donors who had registered on or prior to the cutoff date were included in that target window’s set of potential donors. For each potential donor, I generated a set of features:

- **DaysSinceLastRegistration (Recency)**: the number of days between the cutoff date and the last time a given donor registered
- **DaysSinceFirstRegistration (Time)**: the number of days between the cutoff date and the first time a given donor registered
- **PastRegistrations (Frequency)**: the total number of times a given donor registered on or prior to the cutoff date
- **LastDonationLocation\_Center**: whether the last donation occurred in a center (1 or 0)
- **LastDonationType\_Platelets**: whether the last donation included platelets (1 or 0)
- **CenterRegistrationProportion**: the proportion of all of the donor’s past registrations (on or prior to the cutoff date) that included platelets
- **RegisteredInTargetPeriod**: whether the donor registered in the 30-day target window (1 or 0)

I repeated this process for each cutoff date and combining all subsets into a single dataset.

Random_ID	RegistrationTime					
1253498	2015-01-01 12:05:00					
8349921	2016-02-24 13:15:00					
5132947	2015-11-15 08:30:00					
4257796	2016-03-08 09:30:00					
...	...					



Random_ID	CutoffDate	Recency	Frequency	Time	Target
1253498	2016-03-31 23:59:59	23	5	396	1
8349921	2016-03-31 23:59:59	45	2	212	0
5132947	2016-03-31 23:59:59	6	1	200	1
4257796	2016-03-31 23:59:59	23	1	23	0
...	...	...	...	...	...

**Figure 5: Feature engineering illustration**

In addition to generating features, I eliminated certain donors based on their eligibility to donate within the target window. After donating blood, a donor cannot donate again until a certain number of days have passed. The number of days is based on the donation type.

Donation Type	Eligibility Days
Whole Blood	56
Platelets and Concurrent Plasma	28
2 Units RBC	112
RBC with Platelets and Plasma	56
Plasma Apheresis	28
Platelet Apheresis	7
RBC with Platelets	56
Single Unit Recovery	56
RBC with Plasma	56

**Table 1:** Donation type eligibility map

Using each donor's most recent donation type and the table above, I calculated the percentage of each target window for which a donor was still eligible. I eliminated donors with a percentage of 0, since the target variable would be a guaranteed negative.

Finally, I split the data. I kept the period from 7/22/2019 – 8/20/2019 as a holdout dataset for the purpose of additional evaluation and demonstration. Then, I randomly selected 80% of the remaining data as a training dataset. The other 20% was used as a test dataset.

### Cross-Validation

There are many methods capable of predicting two-class probabilities. I evaluated the following classification methods using 3-fold cross-validation:

- **Logistic regression** using scikit-learn's **LogisticRegressionClassifier**: once with no penalty and once with L2 regularization
- **Random forest** using scikit-learn's **RandomForestClassifier** with 100 estimators and max tree depth of 5
- **AdaBoost** using scikit-learn's **AdaBoostClassifier** with learning rate of 1
- **Gradient boosted decision tree (GDBT)** using XGBoost's **XGBClassifier** with 100 estimators, max tree depth of 5, and learning rate of 0.5
- **Decision tree/CART** using scikit-learn's **DecisionTreeClassifier**
- **Gaussian Naïve Bayes** using scikit-learn's **GaussianNB**

The best-performing classifier was the GDBT fitted using XGBoost, with a mean cross-validated average precision of 0.305. In addition, the cross-validation fitting process for XGBoost took 5.7 minutes, whereas the next best-performing methods (random forest and AdaBoost) took over 35 minutes each to fit. Cross-validation scores for all models are included in the appendix as Table A2.

I selected 3-fold cross-validation as a trade-off between model training time and the need to acquire unbiased performance estimates. The hyperparameter selections were also driven by this trade-off.

### Metric Choice

I scored models by cross-validated **average precision**, which “summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold” (scikit-learn Documentation 2019).

I judged a precision metric to be most suitable to the real-world objective, rather than alternatives such as recall or accuracy. Precision, calculated as  $TP / (TP + FP)$ , indicates the proportion of predicted donors ( $TP + FP$ ) that will donate again ( $TP$ ). On the other hand, recall indicates the proportion of all repeat donors ( $TP + FN$ ) who were identified correctly. Given finite marketing resources, reaching the entire universe of repeat donors is probably not feasible. Therefore, I chose a metric that would select a model whose predictions are mostly worth acting on.

In addition, I selected average precision instead of simple precision because relative probabilities may be more actionable than binary classifications for direct marketing. Average precision summarizes performance at various decision thresholds, giving a general measure of whether the highest-ranked donors are actually the most likely to donate again, rather than measuring the model’s classification performance at a defined threshold (e.g.  $P = 0.5$ ).

Given finite resources, it may not be possible to conduct outreach activities to all past donors who are likely to donate again. Having information on which donors are more likely to donate in the future – compared to other donors – enables more granular targeting.

### Fit Model

Having identified the highest-performing model through cross-validation, I fitted a GDBT model on the entire training dataset using the parameters evaluated during cross-validation. I serialized the fitted model using the Python **joblib** package in order to call it using separate scripts.

### Test Results

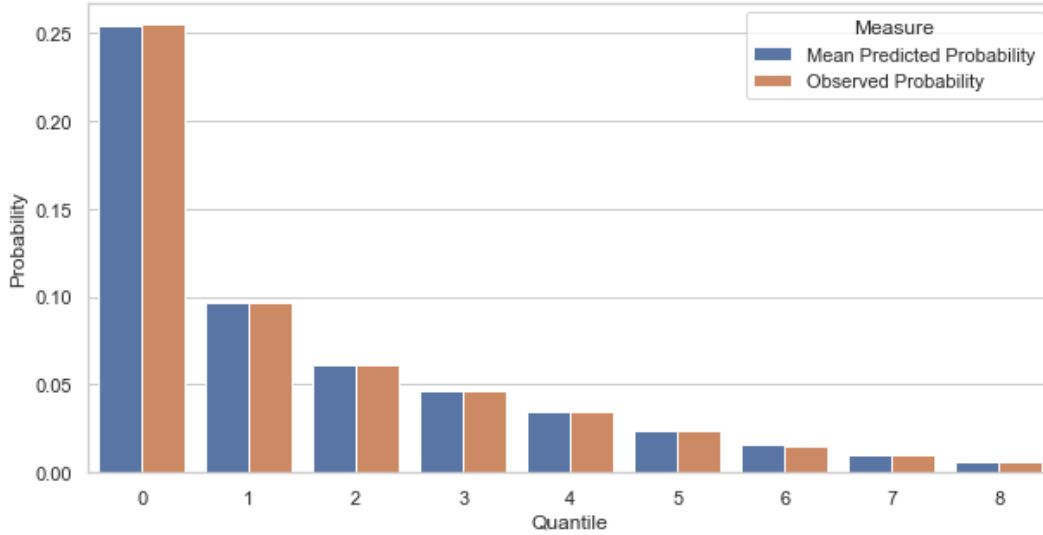
After fitting the model, I scored it on the test dataset using scikit-learn’s **classification\_report** function to calculate precision and recall for both classes. My model achieved a precision of 0.68. The full classification report is included in the appendix as Table A3.

In addition, I generated predicted probabilities for all donors in the test dataset, ranked donors by their predicted probability, and sorted them into deciles. After creating deciles, I calculated:

- **Observed probability of donation:** observed donations as a percentage of total donors in each decile
- **Mean predicted probability of donation:** the mean predicted probability for all donors in each quantile

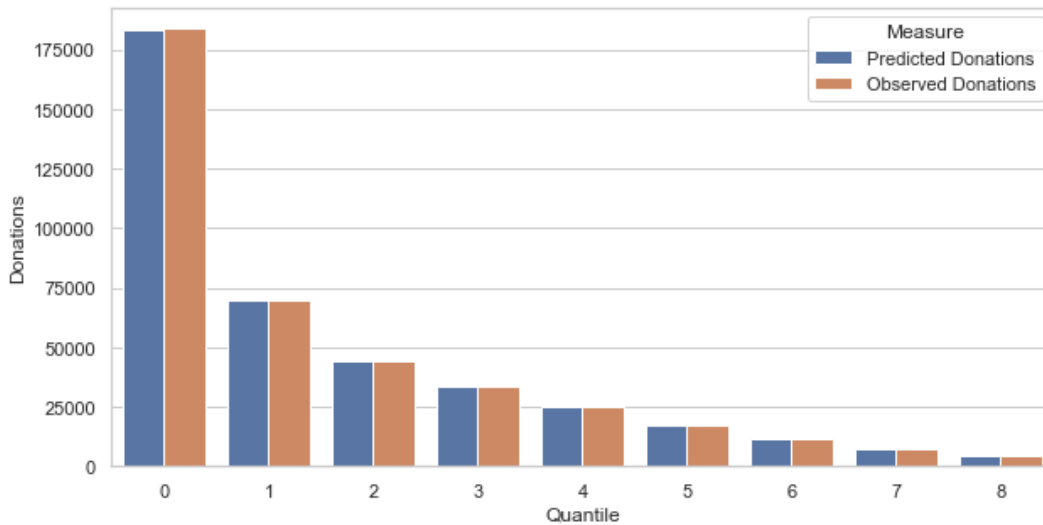
A well-performing model should assign greater probabilities to donors in the top deciles. Comparing these showed that the model generally ranks donors appropriately.





**Figure 6:** Predicted vs. observed probabilities of donation by decile in test dataset

I also predicted the expected number of donations for each decile. Expected donations for each decile can be predicted using an expected value calculation:  $\sum P$  or  $\bar{P}N$ , where  $P$  is the vector of predicted probabilities for donors in decile  $q$  and  $N$  is the number of donors in decile  $q$ . Using this calculation, I predicted a total of 396,610.64 donations. The observed number was 396,687, giving me an absolute prediction error of 0.019%.



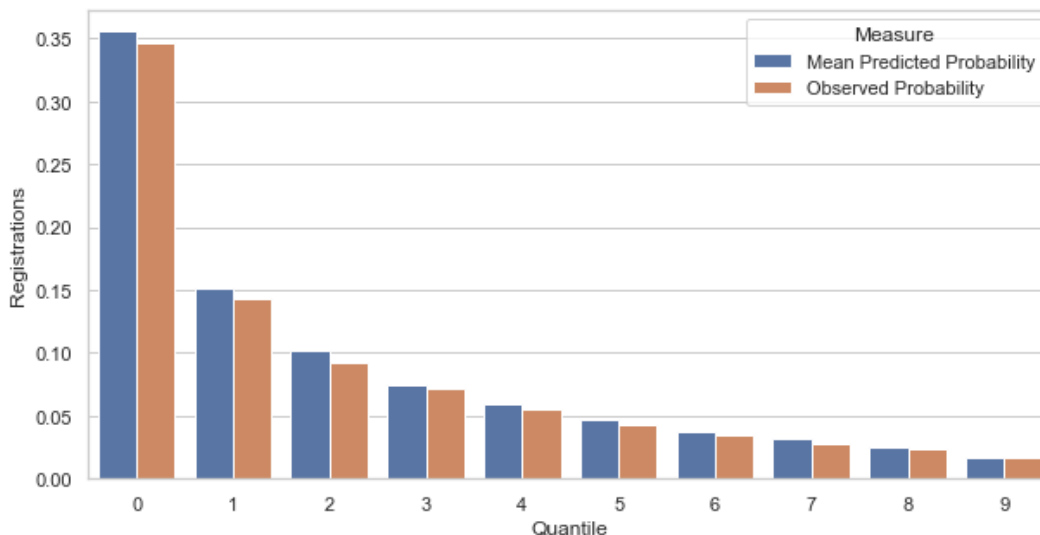
**Figure 7:** Predicted vs. observed donations by decile in test dataset

#### Evaluation on Holdout Time Period

Finally, I performed the same exercise using the holdout target window dataset for 7/22/2019 – 8/20/2019. I reserved this subset in order to demonstrate model performance on a “real-world” dataset. The randomly sampled test dataset comprises a mix of observations from various cutoff dates and target windows; this potentially obscures the effect of long-term trends that would be more apparent in a dataset for a single target window in the future. For example, if there is an

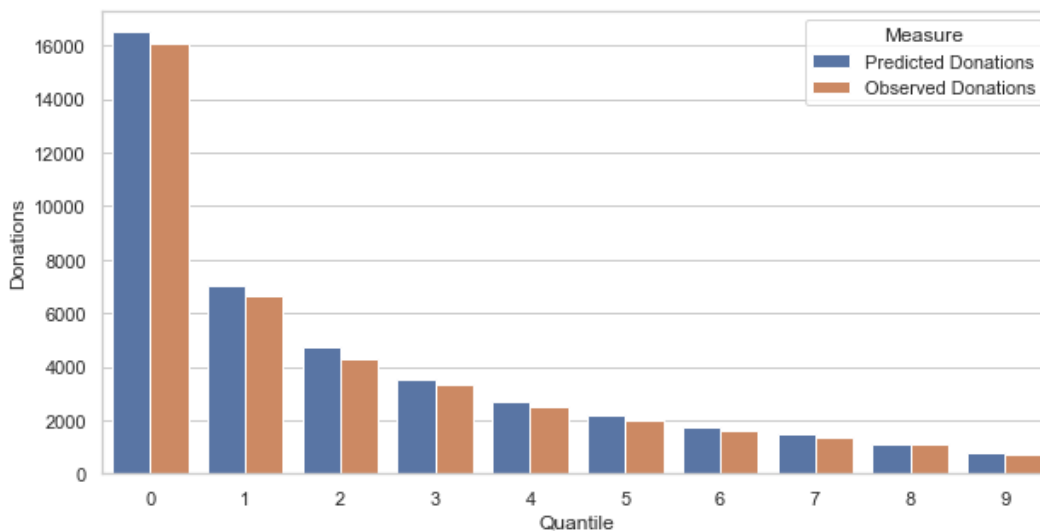
ongoing trend of decreasing donation rates, the effects of this trend will likely be more prominent for a prediction window of 1/1/2020 – 1/30/2020 than in a random sample of observations across multiple years.

My model scored a precision of 0.68 on the holdout dataset. The mean predicted probabilities followed a similar distribution to that of the test dataset.



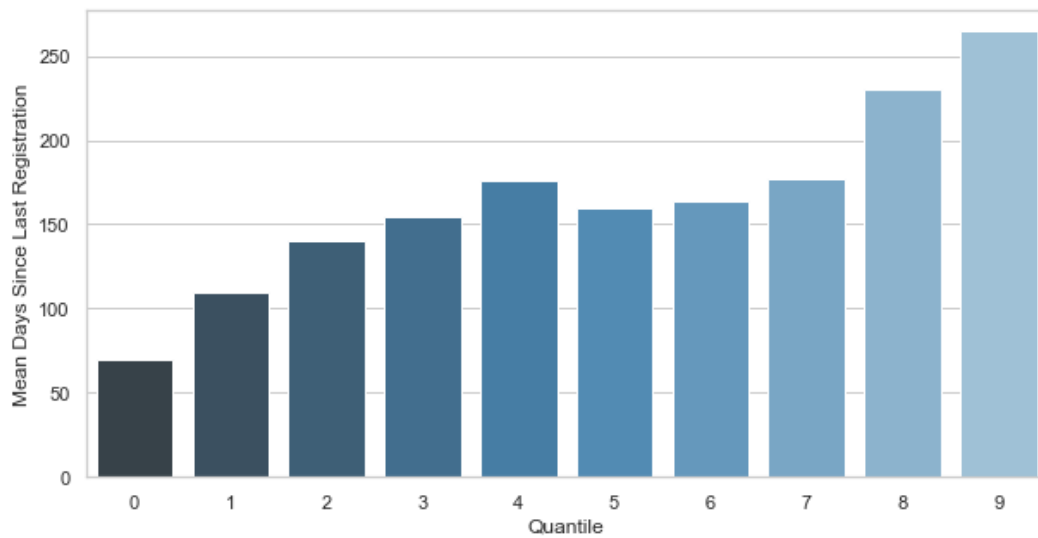
**Figure 8:** Predicted vs. observed probabilities of donation by decile in holdout dataset

Using the same expected value calculation as before, I predicted 41,802 donations. Compared to the observed total of 39,621, this represents an absolute prediction error of 5.5%. This demonstrates that a GDBT model can be used for both donor ranking and prediction of expected donations.

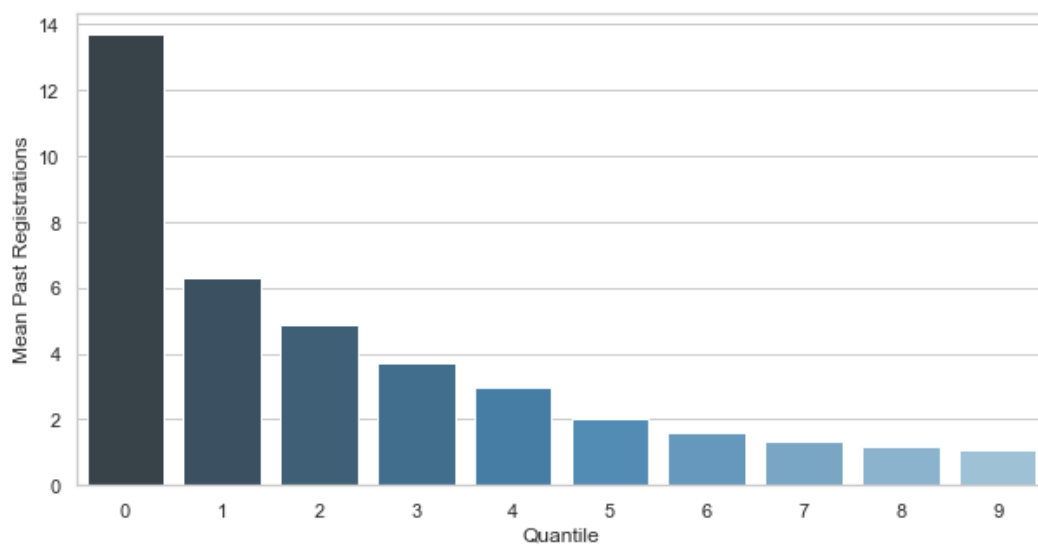


**Figure 9:** Predicted vs. observed donations by decile in holdout dataset

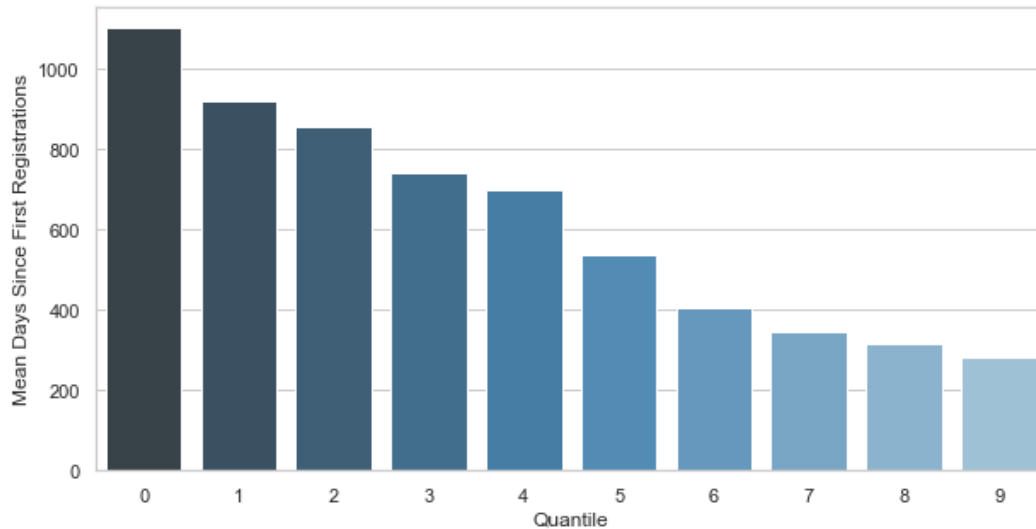
Finally, I calculated the mean of each feature for each decile. Plots of the three “primary” features (recency, frequency, and time) are shown below. A table of feature means is included in the appendix as Table A4.



**Figure 10:** Mean recency (days since last registration) by quantile



**Figure 11:** Mean frequency (total past registrations) by quantile



**Figure 12:** Mean time (days since first registration) by quantile

The plots above and Table A4 show that the groups of donors most likely to donate again:

- Have donated more recently, more frequently, and for longer
- Are more likely to have last donated in a center
- Are more likely to have last donated platelets
- Donate more frequently in centers, compared to other donors
- Donate platelets more frequently, compared to other donors

## Reproducing Results

My methodology has been captured in the Jupyter notebooks included with this report under the **notebooks** folder. Package requirements have been included in the **env/environment.yml** file, which can be cloned using the conda environment manager. Additionally, the fitted models have been saved using the Python **joblib** package, which can also be used to deserialize them. The included models are **forecaster.pkl** and **classifier\_full.pkl**.

## Conclusion

The question of inventory practicality can be answered well using time series methods. Facebook's Prophet model provides an easy-to-use API that generates accurate forecasts using default parameters. Using simulation methods available as part of Prophet's primary object class, daily forecasts can also be aggregated into forecasts for time periods of any length – for example, to forecast donations over the next 30 days.

The question of donor propensity can be answered using any classifier capable of calculating class probabilities. XGBoost's GBDT classifier performs well when assessing donors' relative propensities by ranking and segmenting them into deciles by predicted probability. Using predicted probabilities and expected value calculations, it is also possible to predict the number of donations from each decile. Finally, summarizing and visualizing features by quantile provides insight into the characteristics of donors who are most likely to donate again.

## Appendix

**Table A1:** Prophet simulation performance metrics

Horizon	MSE	RMSE	MAE	MAPE	MDAPE	Coverage
3 days	256493.6	506.452	388.8057	0.136318	0.113286	0.785255
4 days	310399.2	557.1348	420.557	0.150585	0.116353	0.747053
5 days	308535.5	555.4597	420.5864	0.155713	0.121922	0.738067
6 days	295118.3	543.2479	395.5793	0.152748	0.106867	0.765148
7 days	240647.1	490.5579	356.9514	0.136874	0.100198	0.803822
8 days	206003.7	453.8763	330.87	0.126271	0.092677	0.825559
9 days	181728.7	426.2965	319.5376	0.120446	0.086519	0.841228
10 days	174550.5	417.7924	316.6547	0.650424	0.085706	0.8463
11 days	175275.2	418.6588	322.9277	0.652624	0.101388	0.845304
12 days	188025.2	433.6187	337.7542	0.658874	0.107532	0.826918
13 days	220470.5	469.5429	359.6895	0.144472	0.10479	0.794936
14 days	233658.7	483.3826	365.4693	0.136869	0.106387	0.794567
15 days	262473.2	512.3214	389.2018	0.142859	0.111487	0.769133
16 days	242961.7	492.9114	383.6791	0.139229	0.121989	0.802118
17 days	267626.2	517.326	399.9916	0.142342	0.125183	0.786161
18 days	274848.3	524.2597	406.7059	0.142475	0.125183	0.785164
19 days	340451.6	583.4823	443.9136	0.159906	0.125923	0.747145
20 days	356199.7	596.8247	451.7276	0.168167	0.127594	0.749117
21 days	334341.4	578.2226	421.1946	0.160486	0.118258	0.776017
22 days	262419.9	512.2694	375.4971	0.141519	0.108993	0.803641
23 days	211629.9	460.0325	342.5423	0.127908	0.092157	0.81451
24 days	185646.6	430.8673	329.3409	0.124362	0.088174	0.835884
25 days	186321.1	431.6493	327.854	0.682632	0.092818	0.829816
26 days	192343.2	438.5695	340.805	0.687649	0.108518	0.828729
27 days	212959.6	461.4755	361.8871	0.695693	0.110546	0.805
28 days	238435.6	488.2986	378.4264	0.151099	0.119056	0.78361
29 days	246860.6	496.8507	382.2915	0.142076	0.119769	0.783425
30 days	271840.4	521.3831	398.7922	0.145242	0.119831	0.769224

**Table A2:** Binary classification cross-validation results

Estimator	Minimum Avg Precision	Mean Average Precision	Max Avg Precision	Std Avg Precision
XGBClassifier	0.305453	0.305535	0.305646	8.14249e-05
AdaBoostClassifier	0.285788	0.285867	0.285937	6.11006e-05
RandomForestClassifier	0.283268	0.283771	0.284144	0.000369363
LogisticRegression (L2)	0.252981	0.253321	0.253598	0.00025551
LogisticRegression (no penalty)	0.252981	0.253311	0.253569	0.000245361
GaussianNB	0.181932	0.182073	0.182184	0.000104941
DecisionTreeClassifier	0.142914	0.143243	0.143441	0.000234323

**Table A3:** XGBoost gradient boosted decision tree classification report for test dataset

	Precision	Recall	F1-Score	Support
0	0.94	1.00	0.97	6094080
1	0.68	0.09	0.16	396687
Accuracy			0.94	6490767
Macro Avg	0.81	0.54	0.57	6490767
Weighted Avg	0.93	0.94	0.92	6490767

**Table A4:** Mean feature values by propensity decile in holdout time period dataset

Decile	Days Since Last Registration	Days Since First Registration	Past Registrations	Last Donation Location_Center	Last Donation Type_Platelets	Center Registration Proportion	Donations Per Day	Platelet Registration Proportion	Registered In Target Period	Predicted Probability Of Registration
0	69.590	1100.073	13.670	0.473	0.125	0.463	0.013	0.115	0.346	0.355
1	109.642	918.023	6.289	0.277	0.035	0.282	0.008	0.036	0.143	0.151
2	140.735	856.814	4.861	0.175	0.025	0.179	0.007	0.026	0.092	0.102
3	154.990	741.071	3.704	0.117	0.019	0.119	0.006	0.020	0.071	0.075
4	176.547	700.013	2.994	0.080	0.015	0.082	0.007	0.015	0.055	0.059
5	160.192	534.804	2.044	0.053	0.013	0.056	0.007	0.013	0.043	0.046
6	163.880	404.276	1.594	0.039	0.008	0.039	0.008	0.008	0.035	0.038
7	176.748	343.610	1.355	0.031	0.007	0.031	0.007	0.006	0.029	0.032
8	230.628	315.978	1.163	0.038	0.011	0.038	0.005	0.010	0.023	0.025
9	264.498	281.034	1.058	0.023	0.061	0.022	0.005	0.055	0.016	0.017

## References

- Hyndman, R.J., & Athanasopoulos, G. 2018. *Forecasting: principles and practice, 2nd edition*. OTexts.com/fpp2.
- Letham, B. 2018. *How to aggregate daily results to weekly*. 2 1. <https://github.com/facebook/prophet/issues/426>.
2019. *scikit-learn Documentation*. [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average\\_precision\\_score.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.average_precision_score.html).
- Taylor SJ, Letham B. 2017. "Forecasting at scale." *PeerJ Preprints* 5:e3190v2 <https://doi.org/10.7287/peerj.preprints.3190v2>.