Improving the Invoice-to-Cash Process with Customer Payment Predictions

Adam Hilgenkamp
School of Informatics and Computer Engineering
Indiana University - Bloomington
ahilgenk@iu.edu

Kaitlin Pet
School of Informatics and Computer Engineering
Indiana University - Bloomington
kpet@iu.edu

Jashjeet Singh Madan
School of Informatics and Computer Engineering
Indiana University - Bloomington
jsmadan@iu.edu

Surya Prateek Soni
School of Informatics and Computer Engineering
Indiana University - Bloomington
susoni@iu.edu

December 2, 2019

Abstract

In this paper we will explore the use of machine learning to improve the invoice collection process for a Fortune 500 company. We present three business problems where the company and their customers would benefit from accurate predictions. The first question is to determine if a customer will pay more than 30 days past the due date of the invoice. The second is to predict the day that a customer will pay. The third is to identify invoices that may not be paid at all. Each question is approached with the same general strategy. We clean the collected data, apply feature engineering techniques, build a simple model as a baseline, and finally, explore a more complex model to attempt to achieve better results. Our models achieved similar accuracy to previous research using machine learning for collections predictions. We contextualize those results by studying the accuracy in relation to what is required for effective implementation by the corporation in this study. Finally, we provide recommendations on possible ways to improve accuracy through further model evaluation and improving the data collection process.

1 Introduction

According to Klaus Schwab, "[The Fourth Industrial Revolution] is fundamentally changing the way we live, work, and relate to one another" [1]. One trend of the Fourth Industrial Revolution highlighted by Klaus Schwab and the World Economic Forum is the increase of customer expectations. Customers increasingly expect a personalized and seamless experience, and businesses frequently rely on customer data to deliver that experience [1]. This trend is also highlighted in a study done by IBM which found that 81% of consumers desire improved response time from companies and 76% want companies to better address their individual needs [2]. Machine learning, or the

process of training statistical models to recognize patterns in data [3] promises to deliver actionable insights that can be used to deliver the timely and personalized experiences that consumers want. Often, the focus is on how this can be applied to individual consumers, but the trends also apply to business purchasers, individuals tasked with being the consumer for an organization.

An extension of the consumer and business purchaser trend is a change in business leadership expectations. Business managers are looking to data, a tool of the Fourth industrial revolution, to highlight potential actions that could increase company performance. According to Randy Bean, a writer for Forbes, "Business executives want to understand whether a technology or algorithmic approach is going to improve business, provide for better customer experience, and generate operational efficiencies such as speed, cost savings, and greater precision" [4]. The challenge that managers face is how to use the data available to them in order to provide the quantitative perspective desired in the decision making process. Many business process are complex, and the data may need a significant amount of cleaning and analysis before it can be reliably be used. Still, there is a lot of optimism for the potential of machine learning to provide insights. The Dresner Advisory Services study of the data science and machine learning market found that about 33% of the companies interviewed are deploying machine learning models in their organization [5]. The departments that had the highest adoption rates were the customer and product focused areas of Sales & Marketing and R&D, while the department with the lowest adoption was Finance. Finance

In this paper we look at three questions focused on improving the collections process for a Fortune 500 technology company, both from a customer service and financial planning perspective. Generally, the collections process begins when an invoice is sent to a customer and ends when the invoice is either paid or written off as bad debt. The first research question involves predicting if a customer will pay more than thirty days past the invoice due date. This question is aimed at improving customer experience by minimizing the calls and emails customers receive company. If the company is able to accurately predict excessively late payments they can improve the customer experience by reducing communications to "good" customers and contact "risky" customers sooner. In addition to better customer communication, answering this question can lead to improved performance and efficiencies for the teams responsible for collections. The second question involves predicting the how late a customer will pay their bill. This question has implications for business managers responsible for the cash flow of the company. If the collection amount can be accurately predicted within a few days business, managers may be able to make additional investments or pay their suppliers sooner. The third question involves determining if an invoice will be paid in full, partially paid, or not paid at all. "Bad debt" or the non-payment of an invoice is a liability to this company because it provides technology services on a subscription basis. The technology services are often delivered for a period before the invoice is collected, and during that period costs associated with delivering the service are accrued. To reduce this cost the company wants to address this risk before too much time passes. For each of these questions we explore multiple machine learning methods aimed at achieving satisfactory results and discuss challenges and recommendations for the company.

2 Background and Related Research

When you purchase a home and begin the move-in process, one of the first things you do is setting up a "subscription" to your electric company. Once you sign up and turn on the power you will get a bill at the end of the first month which is due by the first day of the following month. If you get caught up in the process of moving and forget to pay the bill on time you will probably hear from the electric company. In order for them to provide their services they need cash to purchase new equipment and pay their employees. Since this

is essential for their operations they will continue to follow up until the bill is paid or the power is shut off.

This is the collections process, and it plays out in the same way for businesses as for individuals. Businesses sign a purchase agreement, receive the product or service, and must pay the invoice by an agreed upon date. Just like a homeowner paying their electric bill, there can be factors that make paying on time a challenge. For a company to pay their vendors or suppliers they must first collect the cash from their customers. There are also external factors such as macro-economic trends, incorrect details on the invoice, and quality disputes that complicate this process [6].

Larger service providers for businesses will build collections teams or outsource the process in order to help customers work through these challenges and to ensure that the company can collect the cash that it needs to operate. Historically, the process of collecting invoices is very manual, requiring phone calls by the collector. This can be a slow, human resource intensive and sometimes inaccurate process [7].

The collections process has become smarter in recent years, with email driving basic automation of the process. In the past decade, there has been some research of how Big Data and statistical learning models can be used to improve the efficiency of the process. One study done by a research team from IBM in 2008 aimed to classify invoice data from four firms into one of five categories: on time, 1-30 days late, 31-60 days late, 61-90 days late, and 90+ days late. The research team focused on returning customers and used the historical data for those customers in their models. The study utilized a cost sensitive learning approach in order to maximize the predictive accuracy of the 90+ category of invoices. Using 17 different dimensions in their model the research team was able to achieve maximum accuracy of 95.8% using a C4.5 decision tree induction model and a unified model applied to one of the firms [7]. Later research on the use of machine learning to predict collections was done by Hu Peiguang at the Massachusetts Institute of Technology in 2015. Peiguang studied the binary classification of the invoice paying on time or late. The research also looked at the accuracy of predicting the number of days late that the invoice might be received. The author concluded that the best results were achieve utilizing pre-processing techniques and random forest models to fit the data [8]. Additionally, other areas of research in consumer credit rating and fraud detection could be studied and applied to the collections process. There are also products on the market such as the Pega Collections platform which provides tools for process automation and incorporates machine learning models to provide next best actions to consumers and personalized communications for customers [9]. Our research aims to achieve similar results to the research referenced in this section, but applied to the data with differing fields collected from a fortune 500 company not included in the previous studies.

3 Data Collection and Preprocessing

3.1 Data Summary

For the analysis in this paper we have been given access to invoice data for new and returning customers from a Fortune 500 technology company. The dataset contains 7.7 million observations and 139 features dating back to January of 2015. Before using this data in our analysis we needed to assess the predictive quality of the data. The first consideration was the time dimension of the dataset. Through discussions with the company we found that the invoice collection process was changed at the beginning of 2018. The new process added new fields that needed to be filled out on the invoice and no longer required the update of others. For this reason we filtered the data to the invoices created and paid between January 2018 and November 2019. Another consideration was the types of invoices contained in the dataset. The dataset contained both automated invoices set up for recurring payment with a credit card as well as

more complex transactions which were assigned to a collector on the collections team. In our analysis we focus on the invoices that are processed manually so all automated credit card invoices were removed. Lastly, we needed to filter out active invoices so that we only have invoices have been paid in order to make sure that the records contain a classification or regression value that we can use in our analysis. Taking these steps to clean the dataset we ended with 280,884 invoices that we use in our analysis.

In addition to filtering down the volume of data, we needed to assess the validity of the features in our data. Of the 139 features in the dataset 51 of them were more than 90% null. For the remaining fields we removed anything that might cause data leakage. This term refers to data that is only available or data that is updated when the phenomenon you are trying to predict occurs. For example, the last contacted date would be updated each time contact is made. This field would not be populated when a new invoice is observed by the model so the prediction using future information about the process will effect the error rate when the model is used to classify or predict real world data. We also talked with the team in the organization responsible for the dataset and determined that there were additional fields that were not standardized or filled out in a meaningful way. Excluding fields that failed these criteria, we extracted 11 features from the invoice dataset, listed in table 1 that were determined to be usable for our models.

The first iterations of our analysis were done using only the invoice dataset. To provide additional features for the model, we explored the addition of attributes for the account, contract, and the cases logged by the customer. The account and contract features will be defined when the invoice is created, avoiding the potential effects of data leakage. Cases will only exist for existing customers but were added to provide insight into customers who may be having more issues then average customers. We also created two new features derived from fields available in the invoice dataset. The first is the addition of the billing month, which captured any potential variation caused by the timing of the due date. The second field that we added was the length of the collector comments. The collector comments are entered as work is done on the invoice. We hypothesize longer comments signal invoices that are more difficult to collect. With the addition of these features, shown in table 2, we increased the dimensions in our dataset to 27.

Table 1. List of invoice dimensions used in our analysis

No.	Feature Name	Description
1	Invoice ID	The unique ID of the invoice (not used in prediction)
2	Account ID	The ID of the account to collect from (not used in prediction)
3	Collector	ID of the individual responsible for collecting the invoice
4	Collection Status	Status of the collections process updated by the collector
5	Tax Exemption Status	Used to identify non-profit customers
6	Billing Country	Country of the customer
7	Billing Language	Language of the customer
8	Payment Term	The payment terms agreed upon when signing the contract
9	Payment Type	This is how the customer will pay (Example: check or wire transfer)
10	30 day extension flag	Flag set to true if the customer has been granted a 30 day extension
11	Past Due Flag	Flag set to true if the due date has passed with out payment
12	Currency Code	Currency that will be collected
13	Total Amount	The total amount shown on the invoice
14	Payment > 30 days late	Y value for the binary classification problem (Question 1)
15	Bad Debt Flag	Y value for the multi-class classification problem (Question 3)
16	Days to pay	Y value for the regression problem (Question 2)

Table 2. Final list of dimensions used in our analysis

No.	Feature Name	Table Data is Stored in	Description
1	Invoice ID	Invoice	The unique ID of the invoice (not used in prediction)
2	Account ID	Invoice	The ID of the account to collect from (not used in prediction)
3	Credit Check Decision	Account	What was determined from a credit check of the customer
4	Customer Status	Account	Represents if they are a direct customer or through a reseller
5	Credit Score	Account	The credit score of the customer
6	Industry	Account	The industry that the customer operates in
7	Number of Licenses	Account	The number of software licenses that a customer has purchased
8	Usage Score	Account	Internal metric derived to measure customer adoption of the products
9	Employee Count	Account	Number of employees that a customer has reported in public filings
10	Customer Revenue	Account	Revenue that the customer has reported in public filings
11	Due Month	ENGENEERED FIELD	The month that the invoice is due
12	Collector	Invoice	ID of the individual responsible for collecting the invoice
13	Collection Status	Invoice	Status of the collections process updated by the collector
14	Tax Exemption Status	Invoice	Used to identify non-profit customers
15	Billing Country	Invoice	Country of the customer
16	Billing Language	Invoice	Language of the customer
17	Payment Term	Invoice	The payment terms agreed upon when signing the contract
18	Payment Type	Invoice	This is how the customer will pay (Example: check or wire transfer)
19	Comment Length	ENGENEERED FIELD	Length of the collectors comments on the invoice
20	30 day extension flag	Invoice	Flag set to true if the customer has been granted a 30 day extension
21	Past Due Flag	Invoice	Flag set to true if the due date has passed with out payment
22	Currency Code	Invoice	Currency that will be collected
23	Total Amount	Invoice	The total amount shown on the invoice
24	Billing Frequency	Contract	This is how often the customer is billed
25	Total Number of Cases	Case	Number of customer service cases that the customer has opened
26	Billing inquiry cases	Case	Number of billing inquiry cases. (Customer has a billing question)
27	Urgent level cases	Case	Number of urgent cases logged
28	Emergency level cases	Case	Number of emergency cases logged
29	Average case age	Case	Average time that it takes to resolve a customer case
30	Payment > 30 days late	Invoice	Y value for the binary classification problem (Question 1)
31	Bad Debt Flag	Invoice	Y value for the multi-class classification problem (Question 3)
32	Days to pay	Invoice	Y value for the regression problem (Question 2)

3.2 Dataset Analysis

After cleaning our dataset we performed an initial analysis using visual and statistical tools to highlight general patterns in our data. We explored the distribution of the outcome variables, correlation of numeric variables and the association between categorical variables. For the highest correlated variables we explored the relationship further and visualized the results.

First, we wanted to better understand the distribution of our outcome variables or "Y" values. The first research question aims to classify customers into two buckets: less than 30 days late and more than 30 days late. In the data set about 80% of customers pay later than the due date and 34% of customers pay more than 30 days late. However, as seen in Table 3, the mean and median of late days are 28 and 20 days, respectively. This indicates there could be a few outlier customers who pay extremely late. When plotting the kernel density estimate, shown in Figure 1, we find that there are some invoices that are over a year past due. These outliers could cause issues in our analysis and need to be considered for each research question. Our third research question's explicit aims to find these outliers and other never-paid transaction records classified as "bad debt". In our data approximately 95% of invoices are paid in full. The remaining 5% are either not paid at all or only a partial payment is received. The distribution for the outcome variable for research question three can be seen in Figure 2.

Table 3. High-level Summary of Invoice Dataset

	Invoice Dataset Summary
# of invoices	280,844
# of accounts	87,756
Avg Invoices per Customer	3
Avg Days to Pay	28
Median Days to Pay	20

Figure 1. Kernel Density Estimate of Days to Pay

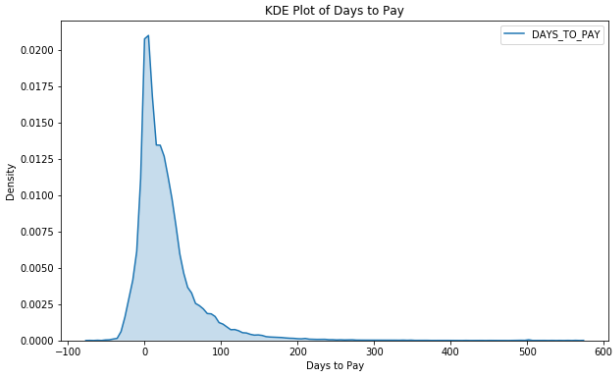
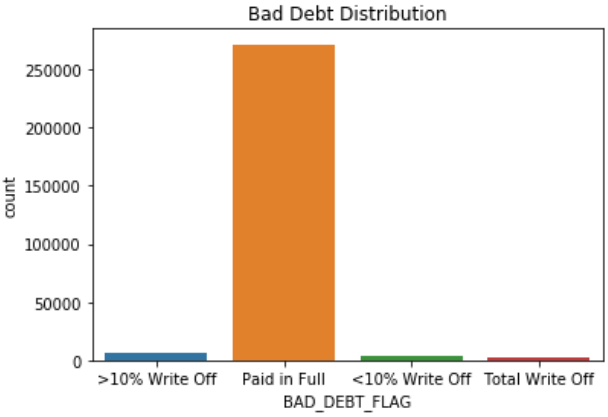


Figure 2. Count Plot of the Bad Debt Flag



Next, we looked at correlation between the numeric variables in the data. Understanding the correlation in our data will help us select the best features to use for each research question. In our analysis we relied on the heat map shown in figure 3 to highlight relationships between variables. The numeric feature with the highest correlation to days to pay was the comment length which had a Pearson correlation value of 0.51. This signifies a moderate relationship between the two and backs up the intuition that an invoice with longer notes is more complicated and takes longer to close. Another variable of some interest is the usage score. With a Pearson correlation of -0.13 there appears to be a slight negative relationship between usage and on-time payment.

Figure 3. Heat Map of Numerical Correlations

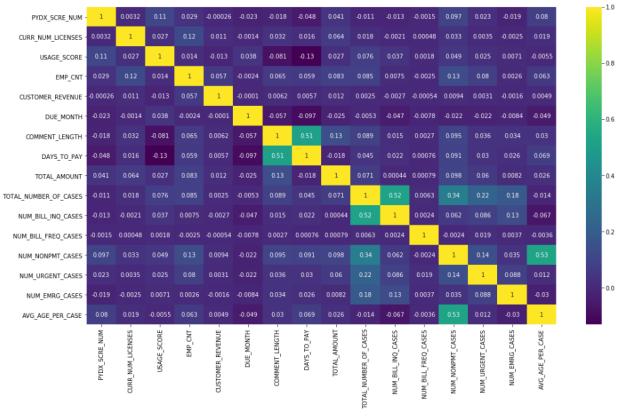
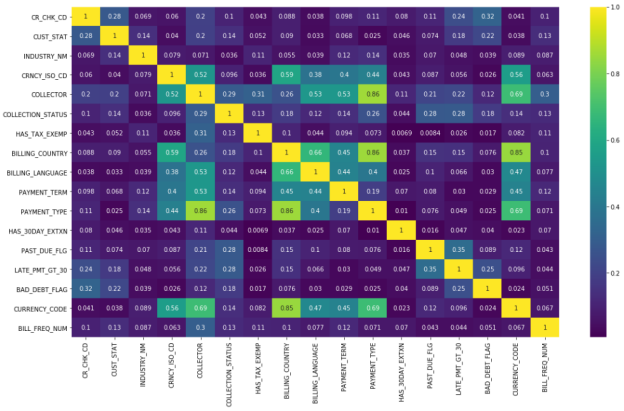


Figure 4. Heat Map of Categorical Associations



Our final analysis focused on the categorical variables in our data set. The Pearson correlation can only be calculated between numeric variables, so we needed a different method to understand the relationships between the categorical variables in our data set. For this we used Cramer's V statistic, which calculates the association between two categorical variables [10]. This value is between 0 and 1, and unlike correlation does not provide a direction for the relationship.

To view the associations between the categorical variables in our data we plotted this statistic using a heat map which is shown in Figure 4. The variable that has the highest association with the over 30 days late field is the past due flag with a Cramer's V statistic of 0.35. This is not very surprising; by definition a customer will have to be late before being 30 days late. A more interesting association is between the collector and a late payment, which has Cramer's V statistic of 0.21. It appears that the person assigned to collect the invoice has some relationship with the outcome. However, it is important to be careful with this information because the collector does not necessarily cause late payments. After studying the distributions, correlations and associations in our dataset, we use this information to guide our approach when studying our three research questions

4 Methods and Results

Our method for each research question follows a similar pattern. We approach each question by reviewing the data to determine if any specific pre-processing is necessary for each research question. We then transform our dataset so that it can be used in our models. Any categorical variables are transformed using one-hot-encoding, which creates a sparse binary matrix to represent the variables. Numeric variables are normalized using mean normalization which ensures that they are on the same scale and do not have an out-sized impact on the prediction. These steps increase the dimensionality of our dataset from 27 to approximately 480 features.

Once we have transformed the features we split our dataset into train, validation and test sets. Our test set contains 20% of the total data. The validation data is 20% of the remaining data and the other 80% is used to train our models. We use the train and validation datasets generated in this step to perform cross-validation on our models and to tune the hyper-parameters. The test data is reserved for the final analysis of model performance. Any variations in this scheme are referenced explicitly in the following sections.

Once we separated the data and formatted

it properly, we start our analysis by creating a baseline model that we can use to measure the performance of more complex models. For classification problems our baseline model is the logistic regression model and for regression problems we use a linear regression model. These models are simple to set up and allow for quick experimentation. After establishing a baseline we explore more complex models in an attempt to achieve better accuracy. This general structure of our method is used in the analysis of research questions discussed throughout this section.

4.1 Question 1: Can we predict if a customer will pay more than thirty days late?

In our first research question, we tried to implement a predictive model to determine the outcome of an invoice right after it is created. After an invoice is generated and sent to the customer, there is usually a 30-day buffer period following the due date of the invoice, after which the payment is considered late. We made a predictive model which forecasts whether or not a customer will pay later than 30 days.

4.1.1 Pre-processing

The dataset had over 12 million entries with each representing an invoice. Initially we only looked at features in the invoice, but later, we also included other features such as contract value, contract amount written off as bad debt, collector's comments, and customer's details such as customer's revenue, billing zip code, payment trends, vat tax and invoice language. Entries with null or zero values were removed. Cases that did not make sense, such as `LATE_PMT_GT_30 = False` and `BAD_DEBT_FLAG` > 50% write off, were also filtered out.

After the initial missing values and outliers check, normalized the data and performed one-hot encoding. Since, the model could take only numerical data, the variables for non-numerical categorical data were One-Hot Encoded using `Numpy get_dummies()`. This caused each record to have over 500 features. To reduce this multidimensionality, we used `sci-kit learn`'s `feature_selection` module. We first used `VarianceThreshold` with threshold value set at 0.01 to remove features that had the same values for most of the data points. This removed around 30% of the features. Then we used the univariate feature selection methods `SelectKBest and SelectPercentile` and `SelectPercentile` to choose the top 30 to 50 features based the following scoring functions: `chi2`, `f_classif`, `mutual_info_classif`. `chi2` was discarded because it could not handle negative values, thus it could not be applied to all the features.

We now had three datasets, all of which will be tested in the following models. The first is the original dataset only invoice-related features. The second had over 500 columns generated from one-hot encoding. The last dataset distills the top 30 percent of the features (as measured by the ANOVA f-value and the method used was f_classif) as chosen by univariate feature selection methods.

4.1.2 Models and results

The baseline model with logistic regression achieved 63% accuracy with the original dataset. Accuracy with the second dataset after trying with 4 different values of regularization coefficient and different penalty values ended up at 64%. Running this model with the third, reduced-feature dataset gave 69% accuracy.

We first allowed a Decision Tree model implemented from sklearn to run until its full depth with minimal parameters. This overfit the model on the training data. Although, it gave 100% accuracy on train data (), it could only manage 74% on test data. Tuning with different parameters, accuracy increased to 77% and running time also decreased drastically.

Random Forest with Hyperparameter Tun-

ing

Random Forest, as the name suggests, is a collection of Decision Trees. Random Forests randomly selects entries and features to build a large number of decision trees. To decide the class of a new entry, each tree "votes" for the class it classifies the entry in. The class receiving the most votes by a simple majority is the "winner" or ultimate predicted class.

To twiddle with different parameters of Random Forest model, we used `RandomizedSearchCV` and `GridSearchCV`. `RandomizedSearchCV` tunes hyperparameters by selecting subsets of different combinations of parameters and choosing the one which gives the best accuracy with cross validation. `GridSearchCV` runs an exhaustive search with every combination of parameters. Accuracy with this model was the highest at almost 78%, with the selected features and hyperparameters chosen by the models.

Testing with a Support Vector Machine Classifier did not yield good results. The most this model could achieve was close to 70%, which is worse than the other models.

Table 4. Summary of Model Results

Model	Baseline (Invoice Only)	Added Data Features	Feature Selections & Parameter Tuning
Logistic Regression	64.7%	71.3%	76.4%
Decision Tree	63.0%	77.1%	77.0%
Random Forest	64.0%	69.4%	79.5%
SVM	63.0%	69.0%	N/A

4.1.3 Conclusion

GridSearch gave the accuracy of 78.35% after running all the combinations in a relatively longer time. Whereas, RandomizedSearch gave 78.18% in our final run. It is interesting to note that, in 7 out of 10 runs with RandomizedSearch, the accuracy was 78.35% which is the highest that the model could achieve. So, it can be said that, considering the performance and time tradeoff, RandomizedSearch proves to be a better choice.

Figure 5. F1 Score of Decision Tree

score of decision tree with 77 accuracy.PNG
score of decision tree with 77 accuracy.PNG

```
from sklearn.metrics import f1_score

f1_macro=f1_score(y_test, y_pred_dt_orig_train, average='macro')
f1_macro

0.5856924884456325

f1_micro=f1_score(y_test, y_pred_dt_orig_train, average='micro')
f1_micro

0.6722858669199264

f1_weighted=f1_score(y_test, y_pred_dt_orig_train, average='weighted')
f1_weighted

0.6412440183133923

f1_average=f1_score(y_test, y_pred_dt_orig_train, average=None)
f1_average

array([0.39628212, 0.77510286])
```

Figure 6. F1 Score of Random Forest

score of random forest with 79 accuracy.PNG
score of random forest with 79 accuracy.PNG

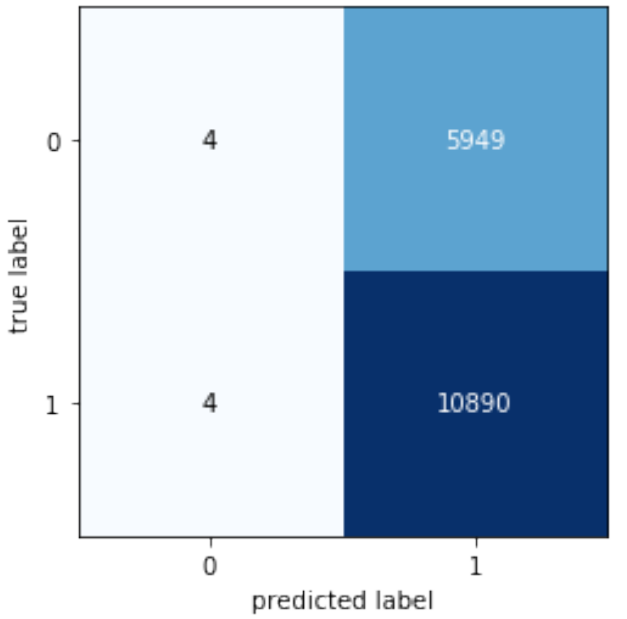
```
from sklearn.metrics import f1_score

f1_macro=f1_score(y_test, y_pred_rf_sel, average='macro')
f1_macro

0.6391979483612374

f1_micro=f1_score(y_test, y_pred_rf_sel, average='micro')
f1_micro

0.7220276607111058

f1_weighted=f1_score(y_test, y_pred_rf_sel, average='weighted')
f1_weighted

0.6898993369539901

f1_average=f1_score(y_test, y_pred_rf_sel, average=None)
f1_average

array([0.46632479, 0.81207111])
```

In random forest, we can see that, in spite of the accuracy being close to 80%, the f_score ranged from 63% and went up to 72%. This is because of the class imbalance. In the dataset, we had output ratio of 70:30. Our models look at the data and cleverly decide that the best thing to do is to predict LATE_PMT_GT_30 as True in most of the cases and achieve high accuracy.

This can also be verified by the output of Logistic Regression, which had classified most of the datapoints for Late Payment as True.

Figure 7. F1 Score of Random Forest

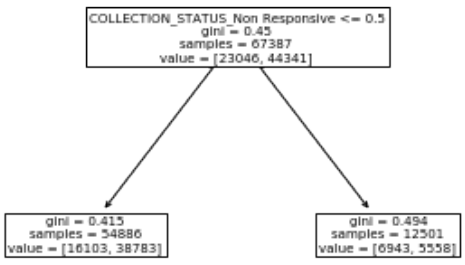
Baseline.png Baseline.png



Another point that can be taken from Decision Tree is that, majority of the customers that were Non Responsive to the Collectors, ended up paying late.

Figure 8. DT

responsive.png responsive.png



4.2 Question 2: Can we predict when a customer will pay?

Question 2 concerns forecasting when a customer will pay. This is relevant to a business because it affects income forecasting: predicting when revenue comes in will effect which quarter that income is reported, thus helping the company manage Wall Street expectations. In the data, how early/late a customer will pay is set in `DAYS_TO_PAY`, where negative values integer values indicate early payment and positive values indicate late payment. We are aiming to create a 'rolling' forecast of when a customer will pay. Since certain later-populated features correlate to paying later, it is useful to include these features so a company has a more refined idea of when a full payment To obtain a data set suitable for this goals, we removed features that directly correlated to certain lateness (e.g. `LATE_PAYMENT`) but preserved others that were populated later in the life of the transaction record.

Data Cleaning

Entries with null or identifiably incorrect data was removed. Entries with bad debt were removed because in those cases a full payment was not made. Features indicating a payment was late and 30 days late were also removed because they did not benefit prediction at any stage - those features would trivially correspond with certain payment lateness. Certain categorical features, such as `INDUSTRY` were compressed because of redundancy in the categories (i.e. the categories of `High Tech` and `High_Tech` were merged. After these reductions and one-hot encoding categorical variables, a total of 355 features remained.

Linear Regression

Linear regression was done with `scikit-learn`'s `Ridge` regression module, which uses least-squares regression with ridge regularization. The best α (regularization parameter) value was chosen by plotting a series of α against mean absolute error and choosing the α

that minimized error. Feature selection was done using scikit-learn's SelectKBest, which returns the k features with highest correlation to DAYS_TO_PAY. To pick the optimal set of features, we ran SelectKBest on the cleaned data set with k ranging from 1 to 355, the total number of features after one-hot encoding categorical variables. Linear Regression was performed on the test data with the optimal values $\alpha = .06$ and $k = 75$, resulting in an average prediction error of 22 days. Cross validation was not performed because test and training error were very similar (See Table 6)

To determine the relative importance of features, we selected the most import and three most important features through SelectKBest then regressed them with DAYS_TO_PAY. The most significant feature was CR_CHK_CD_Credit Hold which indicated a credit hold during the credit check. The linear model including only the 'best' feature gave an accuracy of 25% and the model with the top three features gave an accuracy of 24% on the test data. To see a breakdown of the three most significant features, see Table 5Interestingly, these results did not entirely match our initial prediction of significant features, though whether or not the invoice was processed by a certain collector was the second most significant feature.

Table 5. Most Significant Features

Feature Name	Feature Explanation	Feature Effect on Days to Pay
CR_CHK_CD_Credit Hold	Boolean Feature based on one-hot encoding of the credit category of the customer. In a credit hold, the customer has hold placed on their credit.	Highest
COLLECTOR_00530000003iMvZAAU	Identification number of collector	Second Highest
COLLECTION_STATUS_Red Account	Boolean feature based on whether the customer is deemed unhappy or likely to leave.	Third Highest

We explored the potential of a stacking model which would first divide customers into 'lateness buckets'(e.g. early, 0-30 days late, 31-60 days late, etc) then fitting entries in each bucket to their own regression model.[cite?] Linear regression models were fit using the same methods used above. These models averaged accuracy of 6-7 days within a 30 day window. This accuracy was not high enough to justify constructing a stacked model.

Support Vector Regression

Regression on DAYS_TO_PAY was attempted using Support Vector Regression (SVR). Support vector regression is based on finding "support vectors" that are at most ϵ distance from the y value specified by the regresssed function.[11] Other hyperparameters in SVR include γ, or the degree to which points outside the ϵ range are admissible, and C, which determines effective regularization strength.

GridSearchCV (explained in Question 1) was used to tune hyperparameters for a support vector regressor using the radial basis function kernel. Three-fold cross validation is performed within GridSearchCV. The optimal hyperparameters among the options given were calculated to be $C = 1$, $\gamma = .1$ and $\epsilon = 2$ The test data accuracy for this model was mean absolute error of 23 days, worse than the linear model.

Table 6. Regression Model Performance

Model	Validation Error (Mean Absolute Error)	Test Error (Mean Absolute Error)
Linear Model (all features excluding late payment signifiers)	21.182 days	22.429
Linear Model with only **CR_CHK_CD_Credit Hold** (Best Feature from SelectKBest)	23.115 days	24.200 days
Linear Model with **CR_CHK_CD_Credit Hold**, **COLLECTOR_00530000003iMvZAAU**, and **COLLECTION_STATUS_Red Account** (3 Best Features from SelectKBest)	22.933 days	23.960 days
Support Vector Regression using GridSearchCV (C=1, kernel = RBF)	N/A	22.897 days
Support Vector Regression using GridSearchCV (C=10, kernel = RBF)	N/A	22.916 days

Conclusions

Overall, regression gave fairly poor results and little insight on predicting how late a customer will pay on a fine-grained level. Given the high variance even within lateness buckets, it is unlikely a stacked model would have superior performance. Feature analysis indicated that in the linear model, the "best" feature had the highest impact because regression using just that feature

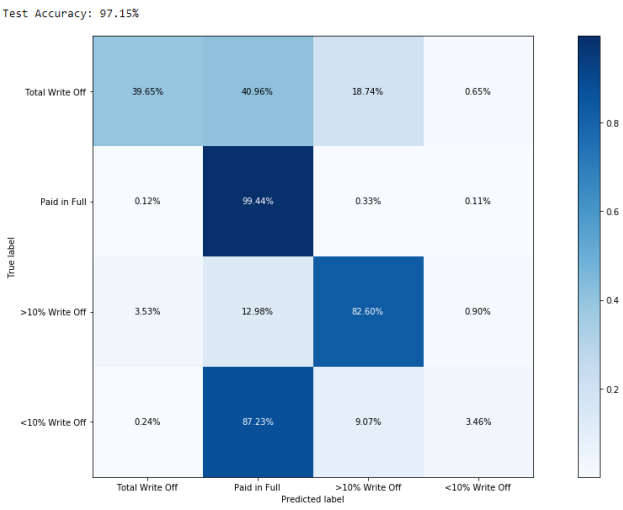
resulted in a average 24-day error, while using all features only improved prediction to an average 22-day error. Paired with the fact that the non-linear SVR model had lower accuracy that the full-feature linear model, it is likely the features in this dataset have very low correlation to how late a customer will pay. We suggest collecting other information about customers to get better results.

4.3 Question 3: Can we classify customers as likely to make a full payment, partial payment or no payment?

Our third research question looks at bad debt. Specifically, we focus on predicting the Bad Debt Flag in our dataset. This field is broken into four different classes: Paid in Full, $< 10\%$ Write Off, $> 10\%$ Write Off, and Total Write Off. As we discussed in the dataset analysis section of this paper and showed in figure 2, the data is imbalanced. This presents a problem for our model as it will disproportionately favor the Paid in Full case. In figure 9 we show the results of using imbalanced data. Our model achieves 97% accuracy but this is misleading as the "Paid in Full" class is over contributing to that result. To solve this issue we balanced the data set using the SMOTE oversampling algorithm. The algorithm generates new data points for the underrepresented classes until there are equal number of observations for each class.

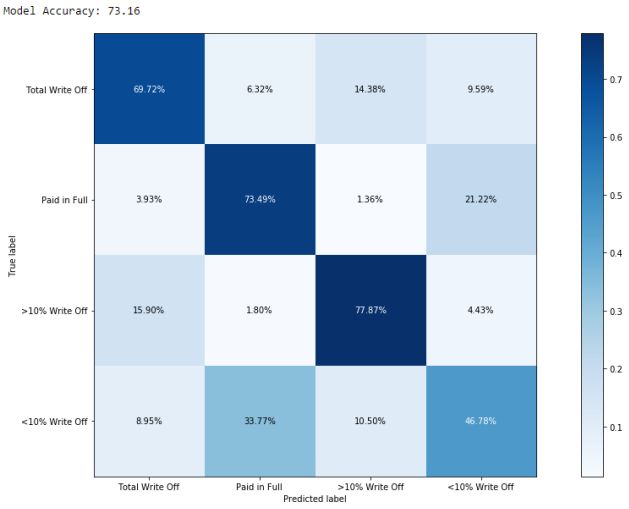
This process significantly increases the number of training observations transforming our original data set from around 180 thousand to almost 700 thousand observations. With the transformed data we established a baseline model using a multi-class logistic regression model. With this model we were able to achieve 73% accuracy. When studying the confusion matrix shown in figure 10 we find that the model predicts a total write off correctly about 70% of the time and has an accuracy of 73.5% for the "Paid in Full" case. The biggest contributor to that confusion is the $< 10\%$ write off case. This is usually caused by small underpayments due to error on an invoice that otherwise

Figure 9. Prediction Result with Imbalanced Dataset



looks very similar to an invoice that is paid in full.

Figure 10. Confusion Matrix of Logistic Regression Predictions



After establishing a baseline model we attempted to improve the accuracy of our predictions. We explored a two different models: Ada boost classification and decision trees. Ada boost gave us an accuracy of 78% and with the first iteration of decision trees we were able to achieve 79% accuracy. From these results we chose the decision

tree model to tune. The major parameter that we focused on was the maximum tree depth and its effect on prediction accuracy. We found that as the tree depth was allowed to increase the model appears to over fit the data as shown in figure 11. To better generalize the model the tree depth we chose for our final output was 12. Using these parameters we were able to achieve a model accuracy of 79.4%. Compared to the logistic regression model the decision tree was more accurate at predicting the "Paid in Full" case as well as the "> 10% Write Off" case. The logistic regression model still performed slightly better at predicting a total write off and the "< 10% Write Off" case.

Figure 11. Tree Depth Effect on Prediction Accuracy

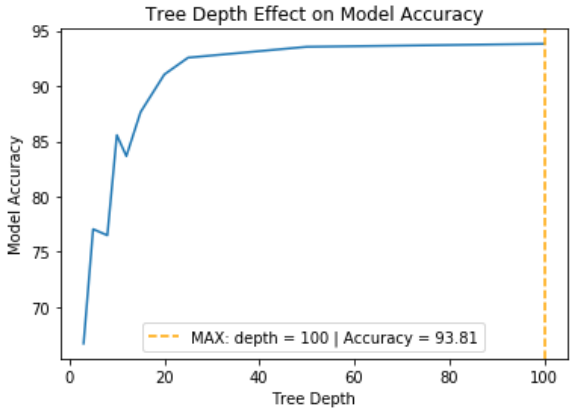
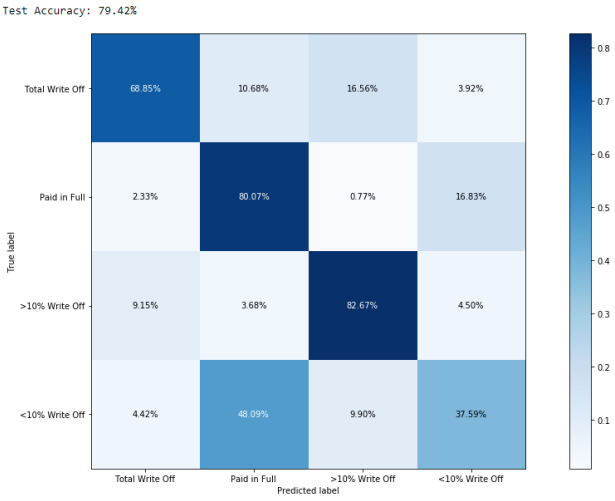


Figure 12. Confusion Matrix of Decision Tree Predictions



5 Conclusion

Summarize our results and draw conclusions about the research that we did.

References

[1] Klaus Schwab. *The Fourth Industrial Revolution*. World Economic Forum, 2016.

[2] Steven Davidson, Martin Harmer, and Anthony Marshall. The new age of ecosystems. Technical report, IBM Corporation, 2014.

[3] Wikipedia contributors. Machine learning, 2019. [Online; accessed 29-November-2019].

[4] Randy Bean. The state of machine learning in business today. *Forbes*, Online, September 2018.

[5] Louis Columbus. The state of ai and machine learning in 2019. *Forbes*, Online, September 2019.

[6] Vinayak Mungurwadi. A machine learning approach for cash flow prediction, August 2015.

[7] Sai Zeng, Prem Melville, Christian Lang, Ioana Boier-Martin, and Conrad Murphy. Using predictive analysis to improve invoice-to-cash collection. In *KDD 2008 Proceedings*. IBM Corporation, KDD Conference, August 2008.

[8] Peiguang Hu. *Predicting and improving invoice-to-cash collection through machine learning*. PhD thesis, Massachusetts Institute of Technology, 2015.

[9] Pegasystems Inc. Pega collections. [Online: accessed 29-November-2019].

[10] Wikipedia contributors. Cramer's v, 2019. [Online; accessed 29-November-2019].

[11] Bernhard Sch olkopf and Alex J. Smola. A tutorial on support vector regression. *NeuroCOLT Technical Report*, Online, September 2003.