Churn Prediction

Akshay Kumar Chaturvedi

6 June 2018

Contents

Chapter 1

Introduction

1.1 Problem Statement

In today's age, competition is high in every sector. Although, machine learning cannot always help in every sector but in some sectors it definitely can.

One of the problems faced by the companies is to find new customers. Companies find it more expensive to acquire a new customer than to keep their existing one from leaving. This problem can be alleviated if the companies can predict if a person will become the customer by looking at his past behavior, this is where machine learning comes in.

1.2 Data

Data was provided in two sets, 'train' set and 'test' set, 'train' set was used to train the models whereas 'test' set was used to test the models. There are total 21 columns in the data; 'train' and 'test' set contain 3333 and 1667 rows respectively. Please find below a sample of data in tables 1.1, 1.2 and 1.3.

Table 1.1: Sample Data (Columns: 1-7)

state	account length	area code	phone number	international plan	voice mail plan	number vmail messages
KS	128	415	382-4657	No	Yes	25
OH	107	415	371-7191	No	Yes	26
NJ	137	415	358-1921	No	No	0
OH	84	408	375-9999	Yes	No	0
OK	75	415	330-6626	Yes	No	0

Table 1.2: Sample Data (Columns: 8-13)

total day minutes	total day calls	total day charge	total eve minutes	total eve calls	total eve charge
265.1	110	45.07	197.4	99	16.78
161.6	123	27.47	195.5	103	16.62
243.4	114	41.38	121.2	110	10.3
299.4	71	50.9	61.9	88	5.26
166.7	113	28.34	148.3	122	12.61

Table 1.3: Sample Data (Columns: 14-21)

total night minutes	total night calls	total night charge	total intl minutes	total intl calls	total intl charge	number customer service calls	Churn
244.7	91	11.01	10	3	2.7	1	False.
254.4	103	11.45	13.7	3	3.7	1	False.
162.6	104	7.32	12.2	5	3.29	0	False.
196.9	89	8.86	6.6	7	1.78	2	False.
186.9	121	8.41	10.1	3	2.73	3	False.

Out of 21 variables (or columns), 6 are categorical namely 'state', 'area code', 'phone number', 'international plan', 'voice mail plan' and 'Churn'. 'Churn' is the target variable, rest of the categorical variables can be used as features in machine learning models. Apart from these 6 variables, there are 15 numerical variables; their correlation plot can be seen below in figure 1.1.

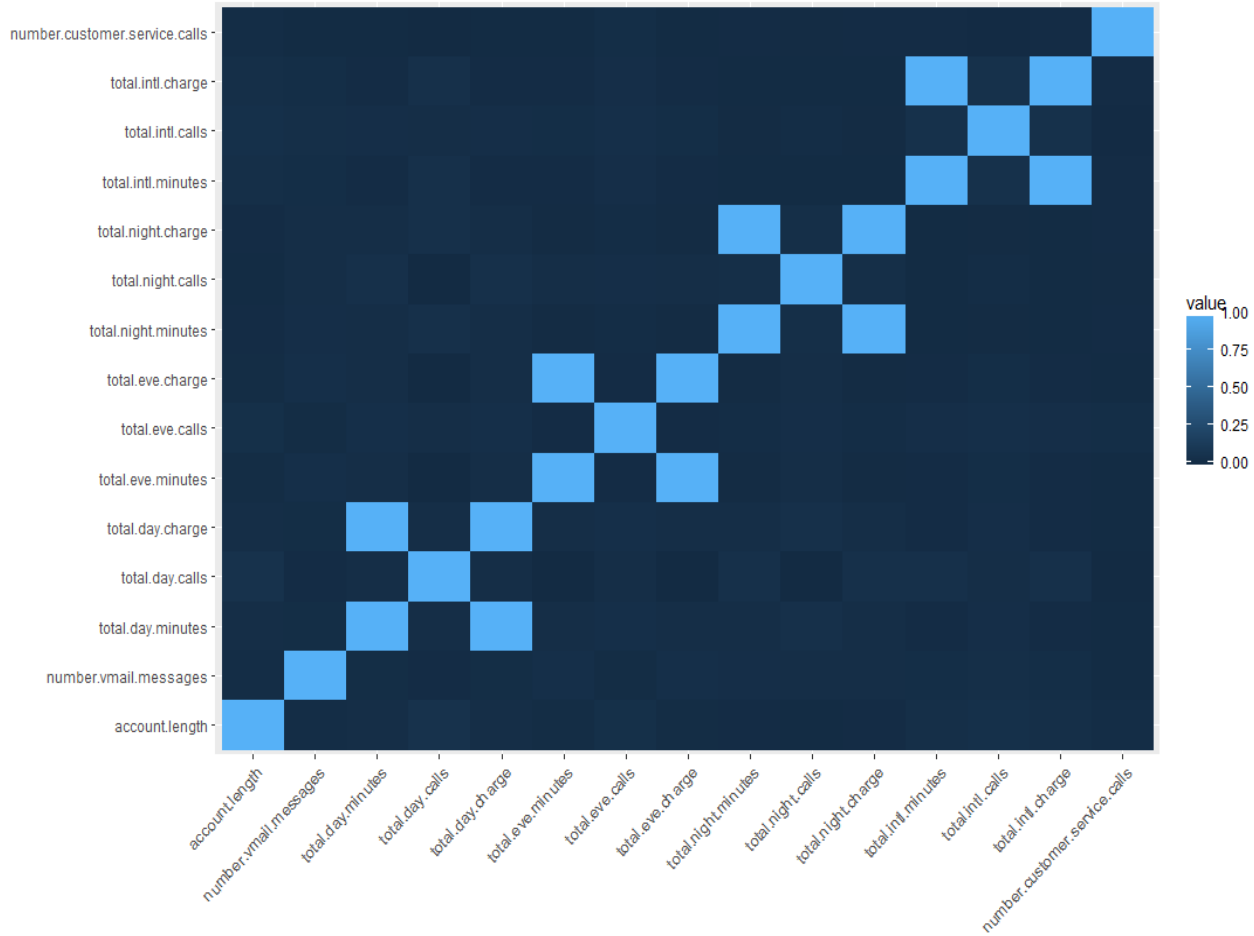

Figure 1.1: Correlation Plot of Numerical Variables

From the correlation plot in figure 1.1, we can observe perfect or near perfect correlation between 'total day charge' and 'total day minutes', 'total eve charge' and 'total eve minutes', 'total night charge' and 'total night minutes' and 'total intl charge' and 'total intl minutes'.

Distribution of these numerical variables can be seen below in figure 1.2.

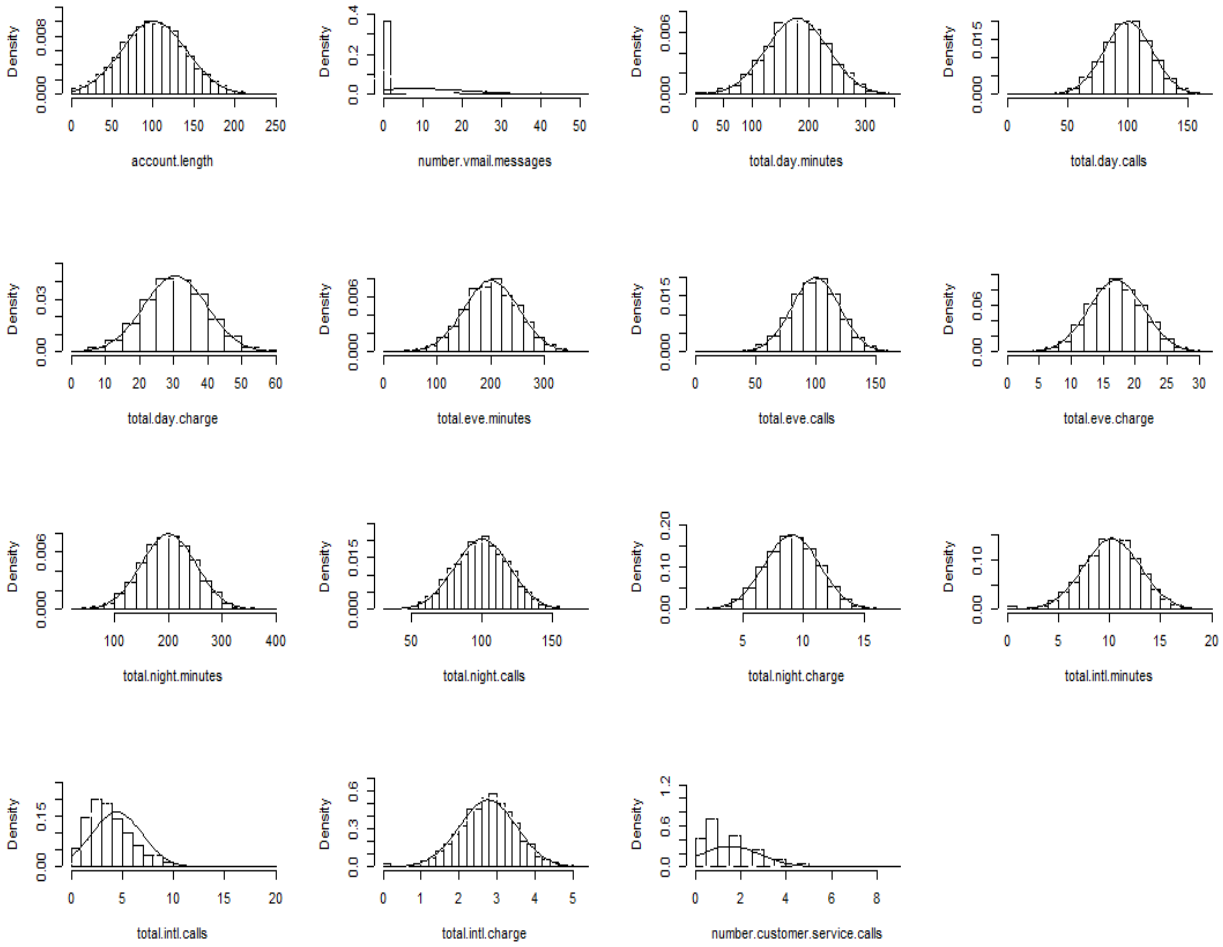

Figure 1.2: Distribution of Numerical Variables

From figure 1.2 we can observe that most of the numerical variables follow a normal distribution approximately. Boxplots were also created for these variables to visualize the outliers if any, they can be seen below in figure 1.3.

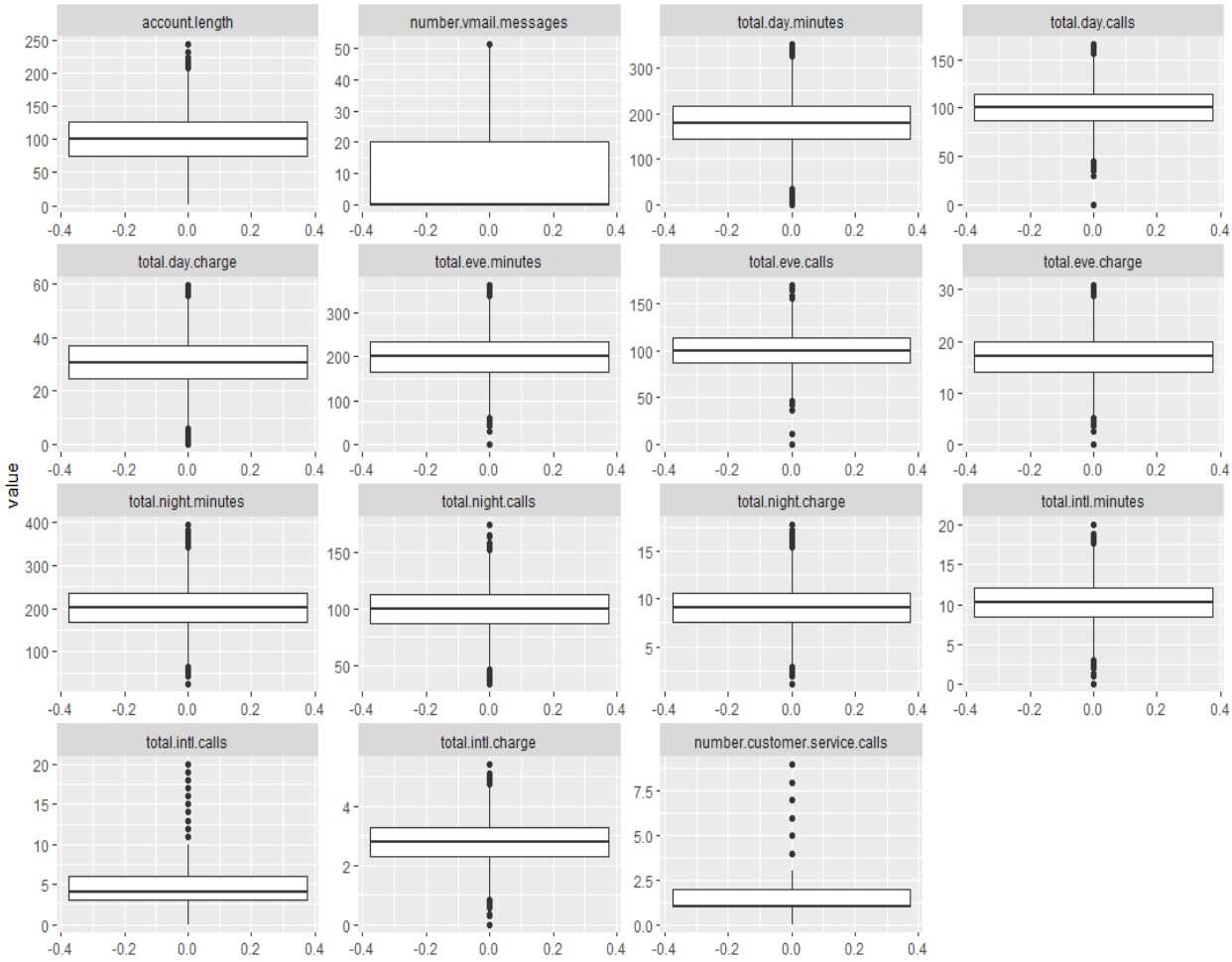

Figure 1.3: Boxplots of Numerical Variables

Boxplots show us that all of the numerical variables contain outliers.

We also looked at the distribution of target class 'Churn' and found out that we have an imbalanced data set, 'train' set contains 2850 'False.' values and 483 'True.' values whereas 'test' set contains 1443 'False.' values and 224 'True.' values.

From above information we can conclude that:

- We have highly correlated variables which can help us in feature selection.
- As most of our numerical variables have approximately normal distribution, we can standardize them to have mean 0 and standard deviation 1.
- Our data has outliers, so we will need to check its effect on models.
- We have imbalanced data set, so we cannot rely on accuracy only, we will have to use some other performance metric to rate our models.

Chapter 2

Methodology

2.1 Pre-processing

In figure 1.3, we can see that range of variables is different, this difference of range among variables may lead a model to favor a feature with greater range over others, so that no feature loses its importance due to lesser range, numerical features were standardized with mean 0 and standard deviation 1.

Another important component of data pre-processing is feature selection. There are 5 categorical features namely 'state', 'area code', 'phone number', 'international plan' and 'voice mail plan'.

Feature 'phone number' was dropped as it had a different value for every row, so it did not have any predictive power. Feature 'state' was dropped as it also had too many unique values which resulted in many unique 'state' values having nearly none 'True.' target values of 'Churn' variable, thereby affecting its predictive power.

For features 'area code', 'international plan' and 'voice mail plan', chi-square test was run, as a result 'area code' did not seem to have any effect on the target variable 'Churn' whereas other two 'international plan' and 'voice mail plan' did, so out of 5 categorical variables 3 were dropped and 2 were selected namely 'international plan' and 'voice mail plan'.

In case of numerical variables, using correlation plot in figure 1.1, we observed perfect or near perfect correlation between 'total day charge' and 'total day minutes', 'total eve charge' and 'total eve minutes', 'total night charge' and 'total night minutes' and 'total intl charge' and 'total intl minutes'. But correlation is not always a wise way of feature selection as the target variable could be dependent on combination of features rather than just one, so we kept all the numerical features and experimented with removing one set of highly correlated features at a time.

2.2 Modeling

All the models were trained on 'train' set and tested on 'test' set. Six models were used in experiments namely 'Logistic Regression'-LR, 'Naïve Bayes'-NB, 'Support Vector Machine'-SVM, 'Gradient Boosted Tree'-GBT, 'Decision Tree'-DT and 'Random Forest'-RF. For evaluation, accuracy score is not sufficient as our data set is imbalanced so we have used average precision from precision recall curve. Performance metrics below are from R programming language, for Python numbers (also visualizations) please check out the appendix.

Below table 2.1 contains results when all the numerical features were present.

Table 2.1: Results with all numerical features

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.19	0.4641	87.40	0.4694
NB	86.64	0.5192	87.34	0.5115
SVM	95.92	0.8965	93.46	0.80
GBT	94.42	0.8470	94.48	0.8251
DT	95.47	0.8382	94.66	0.8018
RF	100	1	96.22	0.8803

Below table 2.2 contains results when highly correlated features with 'charges' in their name have been removed. These features are 'total day charge', 'total eve charge', 'total night charge' and 'total intl charge'.

Table 2.2: Results with 'charges' numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.26	0.4639	87.16	0.4721
NB	85.36	0.4958	85.84	0.4896
SVM	95.74	0.8922	93.10	0.7815
GBT	94.42	0.8470	94.49	0.8252
DT	95.47	0.8382	94.66	0.8018
RF	100	1	95.62	0.8706

Below table 2.3 contains results when highly correlated features with 'minutes' in their name have been removed. These features are 'total day minutes', 'total eve minutes', 'total night minutes' and 'total intl minutes'

Table 2.3: Results with 'minutes' numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.26	0.4639	87.16	0.4721
NB	85.36	0.4958	85.84	0.4896
SVM	95.74	0.8922	93.10	0.7816
GBT	94.54	0.8479	94.30	0.8251
DT	95.47	0.8382	94.66	0.8014
RF	100	1	95.98	0.8734

Below table 2.4 contains results when all the three discrete numerical features not distributed normally namely 'total intl calls', 'number vmail messages', 'number customer service calls' were removed.

Table 2.4: Results with discrete numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	85.98	0.4677	87.16	0.4902
NB	86.47	0.4411	87.46	0.4531
SVM	92.83	0.7620	91.48	0.6389
GBT	91.27	0.6976	91.36	0.6482
DT	91.36	0.6367	91.48	0.5980
RF	100	1	92.38	0.6679

Below table 2.5 contains results when outliers were removed.

Table 2.5: Results with all outliers removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	90.81	0.5703	87.16	0.4721
NB	91.56	0.5253	89.62	0.4964
SVM	96.53	0.8668	93.10	0.7815
GBT	96.42	0.8333	94.48	0.8251
DT	96.03	0.7934	94.66	0.8018
RF	100	1	95.62	0.8705

Some key points on models used in the experiments above:

- Support Vector Machine has been used with radial basis function as kernel, it was chosen as linear kernel performed very poorly.
- Decision Tree here has used ‘entropy’ or ‘information gain’ to measure the quality of a split rather than ‘gini’, this resulted in better performance.
- Random Forest uses 500 trees.
- Gradient Boosted Tree uses 100 trees and maximum depth of tree is 2.

Below in figure 2.1, we can see the precision recall curve for the models used in table 2.1 (as that is the best performing one). In appendix, visualizations and performance metrics from Python programming language are available, those figures and numbers might be a bit different from the one’s here, but the overall trend is same, if you go through only those numbers and visualizations, you will arrive at the same conclusions as you have here.

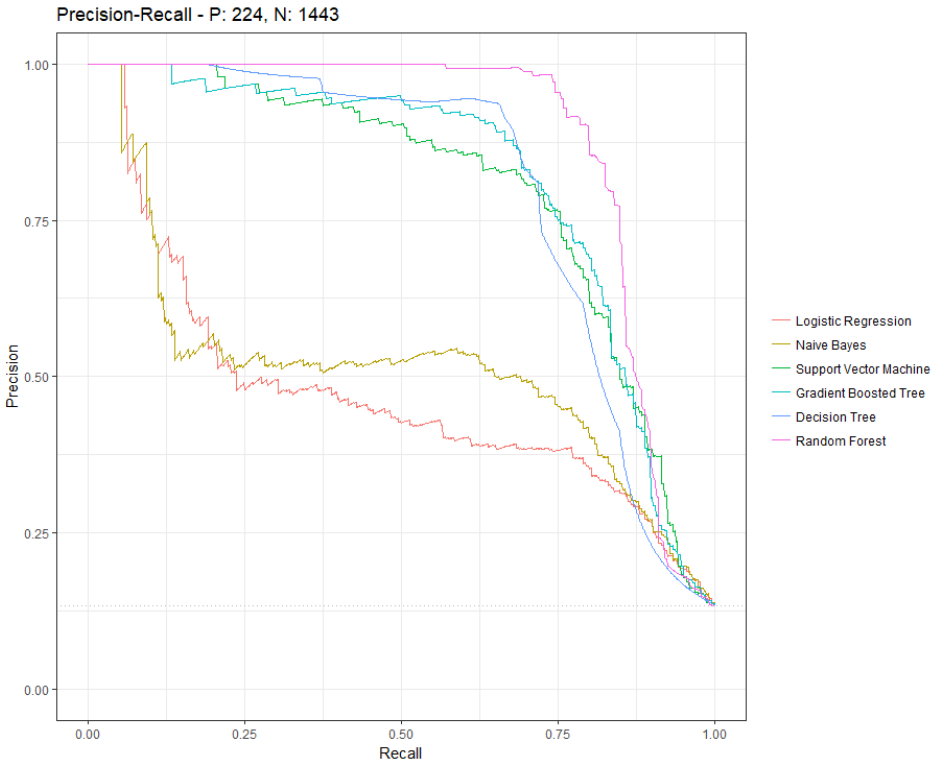

Figure 2.1: Precision Recall Curve of every model for all the features used in table 2.1

Chapter 3

Conclusion

In conclusion, out of all the models shown here and others not shown here but tried Random Forest (RF) with all the numerical features and two categorical features namely ‘international plan’ and ‘voice mail plan’ emerges as the best model according to performance metrics, giving an overall test set accuracy of 96.22 % and average precision on test set being 0.8803. It outperforms other models but it is prone to overfitting, as can be seen when its test set metrics are compared with train set metrics.

Gradient Boosted Tree (GBT) and Decision Tree (DT) are just behind Random Forest. Gradient Boosted Tree always a bit ahead of Decision Tree, but both do not suffer from overfitting. Behind both these models is Support Vector Machine (SVM), performing equally as well as both GBT and DT but like Random Forest suffering from overfitting, kernel used for support vector machine is radial basis function, linear kernel was used but it performed very poorly so radial basis function was chosen.

Logistic Regression (LR) and Naïve Bayes (NB) have performed very poorly in every case. So, ensemble methods like Random Forest and Gradient Boosted Tree have performed well as expected but from our experiments we have observed how powerful Decision Tree is, in ensemble methods several of these trees are used, in Random Forest, 500 were used whereas in Gradient Boosted Tree, 100 were used, just a single Decision Tree performed very well without overfitting and taking less time as well.

We also got to know that we cannot always rely on correlation for feature selection, as can be seen in table 2.2 and 2.3; features deselected by us based on correlation did not make much difference. We also saw that removing outliers also doesn’t help.

We also saw that discrete numerical features 'total intl calls', 'number vmail messages' and 'number customer service calls’ were very useful as when we removed them, performance of every model fell drastically as can be seen in table 2.4; what is interesting is that the model hit most hard by the removal of these features was Decision Tree, ensemble methods like Random Forest and Gradient Boosted Tree were also hit pretty hard by it but not as much as Decision Tree, a reason could be that Decision Tree is only a single tree whereas ensemble methods are built upon many weak learners or trees, this may have saved ensemble methods a bit, so in terms of robustness to features, ensemble methods come out looking better against Decision Tree.

Also, we saw how useful average precision score of precision curve recall curve is in imbalanced data set, if you at accuracy of all models, all of them are above 85% which to anyone at first glance would seem pretty impressive, but when we look further into the data we realize that accuracy is just not the right metric here. For example, suppose the ratio of false: true in the data set is 85:15, now if we predict everything as false we will get 85% accuracy, but that’s bad because we have not been able to correctly predict even a single true observation, this is where precision-recall comes in, precision measures the

ratio of true positives to number of positives predicted by the model and recall measures ratio of true positives to total number of positives in the data set. If both are high, we will have a high average precision (area under precision recall curve) as can be seen in figure 2.1, Random Forest, Decision Tree, Gradient Boosted Tree and Support Vector Machine have a large area under curve and have high average precision.

Appendix A – Python Figures

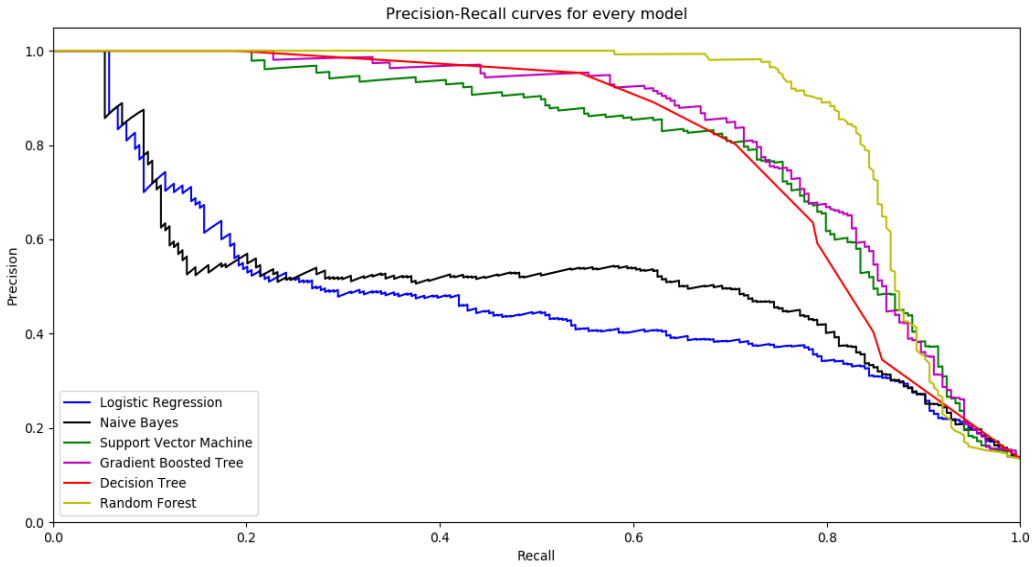

Figure A.1: Precision Recall Curve of every model for all the features used in table B.1

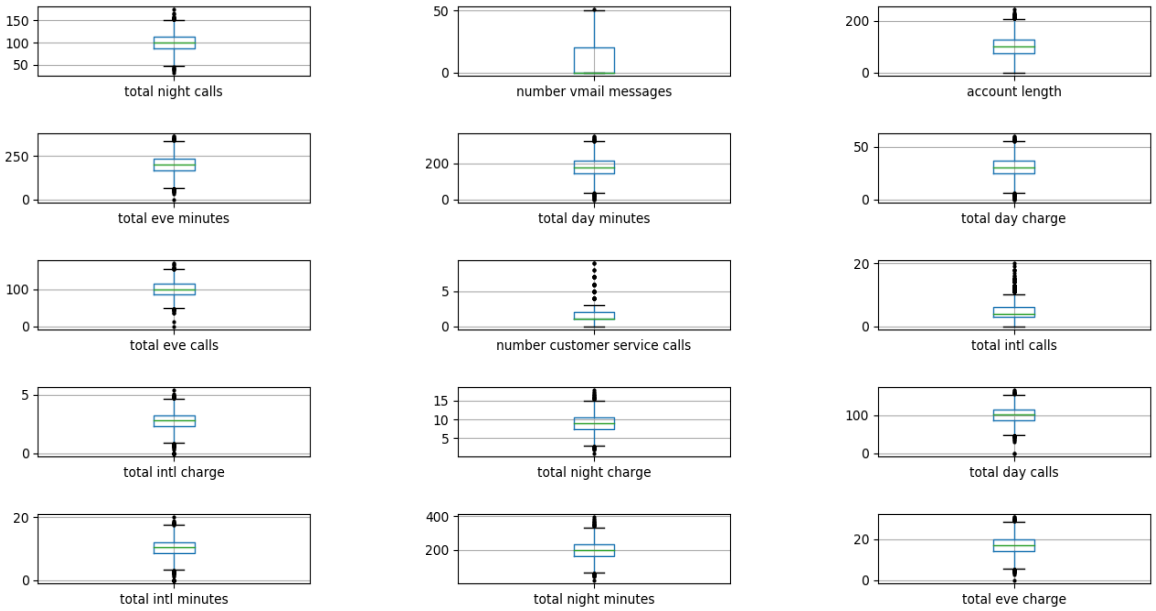

Figure A.2: Boxplots of Numerical Variables

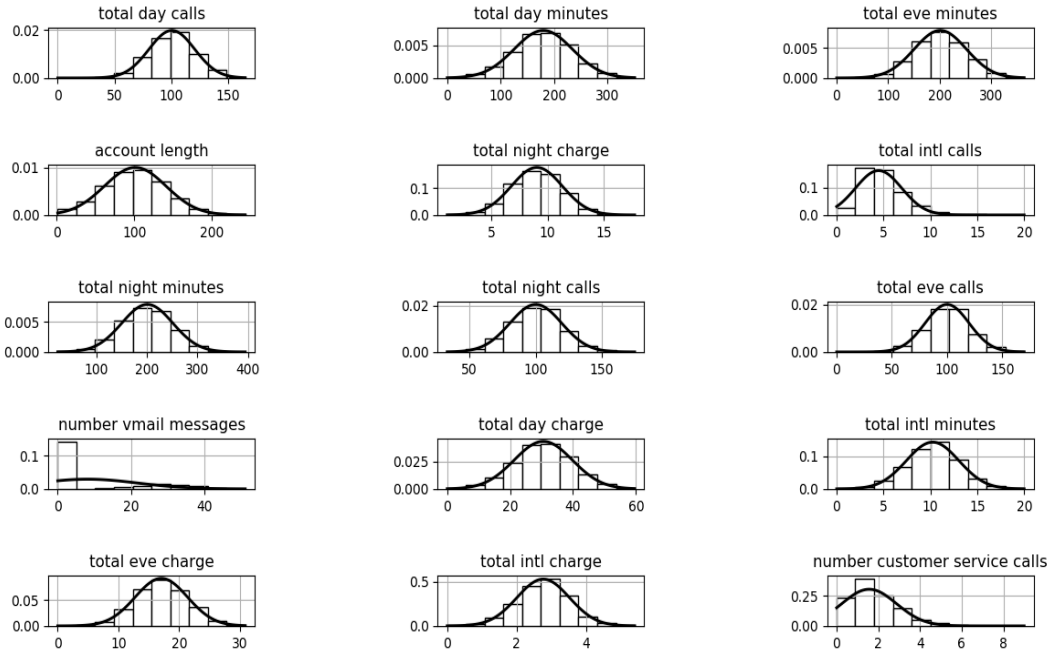

Figure A.3: Distribution of Numerical Variables

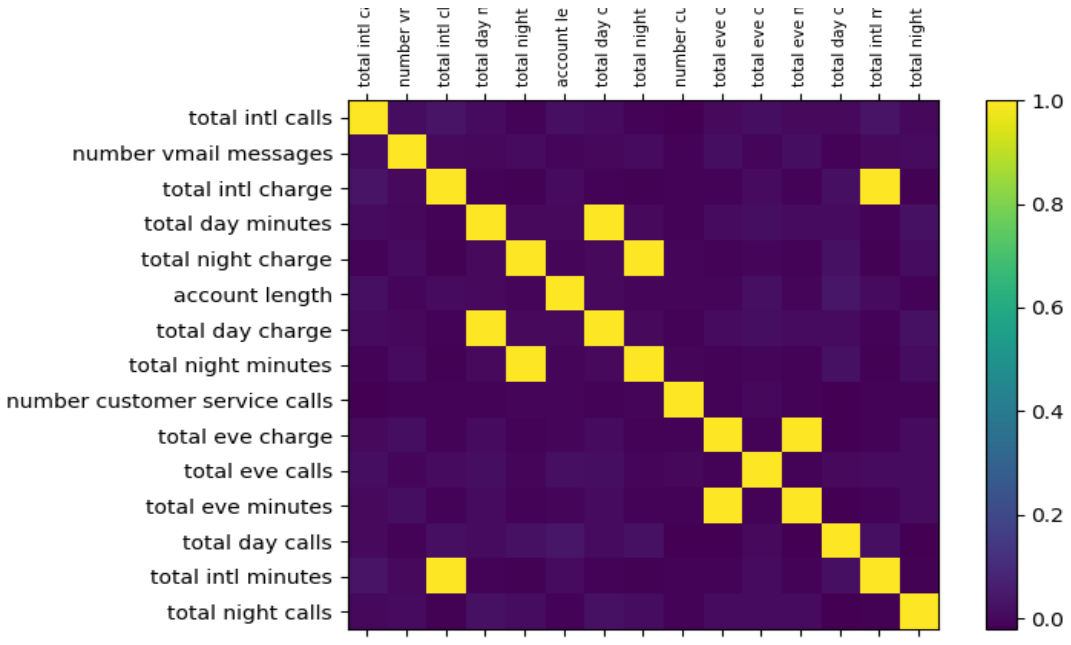

Figure A.4: Correlation Plot of Numerical Variables

Appendix B – Python Predictive Modeling Results

All the points made in the context of tables 2.1, 2.2, 2.3, 2.4 and 2.5 are also true for tables B.1, B.2, B.3, B.4 and B.5.

Table B.1: Results with all numerical features

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.2	0.465	87.1	0.473
NB	86.6	0.52	87.3	0.513
SVM	95	0.897	92.7	0.80
GBT	94.7	0.869	94.1	0.826
DT	93.5	0.786	93.8	0.764
RF	100	1	96.2	0.878

Table B.2: Results with 'charges' numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.2	0.465	87.1	0.473
NB	85.4	0.497	85.8	0.491
SVM	94.6	0.892	92.4	0.782
GBT	94.7	0.869	94.1	0.825
DT	93.5	0.786	93.8	0.764
RF	100	1	95.9	0.872

Table B.3: Results with 'minutes' numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	86.2	0.465	87.1	0.473
NB	85.4	0.497	85.8	0.491
SVM	94.6	0.892	92.4	0.782
GBT	94.7	0.869	94.1	0.827
DT	93.5	0.786	93.8	0.766
RF	100	1	95.9	0.872

Table B.4: Results with discrete numerical features removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	85.9	0.468	87	0.493
NB	86.5	0.442	87.5	0.455
SVM	92	0.762	91.2	0.64

GBT	91.6	0.717	91	0.635
DT	89.8	0.493	90.1	0.471
RF	100	1	92	0.662

Table B.5: Results with all outliers removed

Model	Train_Accuracy	Train_average_precision	Test_Accuracy	Test_average_precision
LR	90.6	0.57	89	0.531
NB	91.5	0.528	89.6	0.497
SVM	96.5	0.867	91.9	0.697
GBT	96.5	0.847	92.5	0.687
DT	94.7	0.662	92.1	0.564
RF	100	1	93.6	0.711