Predicting Churn to Retain Customers in Telecommunications

Deepkumar Patel
DATA-51000-001, Summer 2020
Data Mining and Analytics
Lewis University
deepkumarpatel@lewis.edu

I. Introduction

The dataset used in this report contains a set of features from a French multinational telecommunications corporation called Orange S.A. It is the tenth-largest mobile network operator in the world, and it has over 266 million customers [1]. The dataset contains individual customer activity, which is basically the features, and it also contains a churning column. The churn column contains a binary value, it tells whether the customer has canceled the subscription or not. The author has provided two different CSV files, one for training, and the other for testing. The training contains 80% and the testing contains 20% of the original dataset. For the purpose of this report, I only used the CSV that contains 80%, of the dataset. In the later section, we will see how this dataset is split for training and testing. There are two different sources where you can find this dataset. It is made available publicly on Kaggle [2] as well as on Amazon storage service, also known as Amazon S3 [3].

The main purpose of this report, is to predict the behavior of customers, more specifically, this report predicts whether a customer will cancel the subscription or not. This will not only help businesses to grow and make more profit, but it will also help customers by providing better services. Now, why is it important to save your old customers? It turns out, retaining an old customer is less expensive than acquiring a new customer. Therefore, it is better to focus on existing customers. The two methods that we will use to predict the outcome are the Logistic Regression and Support Vector Machine (SVM). There are many supervised machine learning models that can be tested and used with this dataset; however, this report only focuses on methods that are mentioned earlier.

The future sections of this report describe the data preprocessing, dataset, two different methodologies, and analyzing and understanding the results gathered from using the two methods along with a discussion, and a conclusion. Annexed is a list of sections with a more detailed description. Section II describes the process that is involved before any type of analysis is made on the dataset; this includes transforming and removing of features from the dataset. Section III describes the dataset. Section IV describes how the Logistic Regression can be used to predict binary classification along with model evaluation using different metrics. Section V describes how the Support Vector Machine model (SVM) can be used to predict binary classification along with model evaluation using different metrics. Section VI compares both the models based on the output metrics, along with a discussion. Section VII is the conclusion of this report.

II. DATA PREPROCESSING

Data preprocessing is done using python in Jupyter notebook, which is included with the submission of this report. In addition, all the work from now on related to models and metrics evaluation, is also done in the same Jupyter notebook file. As mentioned earlier, the original source contained two different files, one for training, and the other file is for testing. In this report, we only used the testing dataset. This is simply to demonstrate that we don't necessarily need two different files, the dataset itself can be split into training and testing datasets. The original raw dataset contains 2,666 instances and 20 features. To further understand how this dataset was processed and transformed, let's take a look at the following steps.

A. *Convert Churn to 0's and 1's*

This step is straight forward; it simply converts true to 1 and false to 0. The only purpose for this conversion is to make our final dataset numerical.

TABLE I
Pandas one hot encoding Input/Output

Index	***International plan***
0	No
1	No
2	Yes

Index	***International plan_No***	***International plan_Yes***
0	1	0
1	1	0
2	0	1

B. *One Hot Encoding*

Column "International plan" and "Voice mail plan" are converted to numerical values using a one-hot encoding method [4]. The basic idea is, it converts the variable into dummy variables as shown in Table I. The reason for using this method is because values such as "Yes" or "No" don't have a numeric ordering.

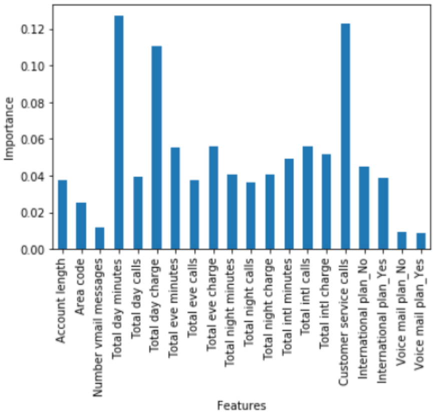

Fig. 1. Feature importance using extra tree classifier

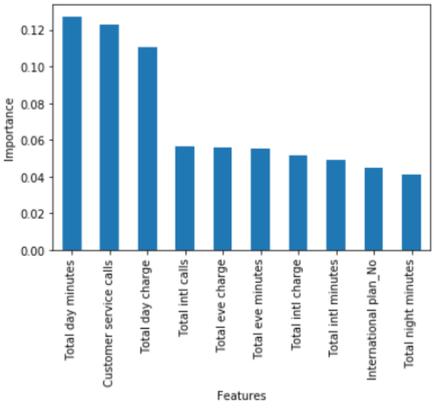

Fig. 2. Top ten correlated features

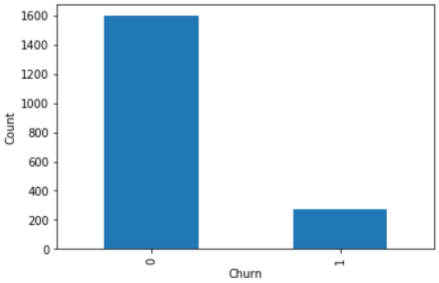

Fig. 3. Imbalanced target class

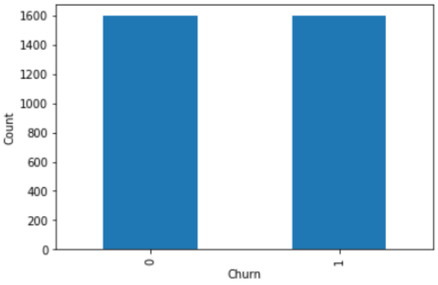

Fig. 4. Balanced target class

C. Feature Selection

This step is important because we want features that are highly correlated with target class. To achieve this, we will use feature importance. This will give us a score for each feature in our dataset. The higher the score, the more important the feature for our output variable. Extra tree classifier [5] has a built-in class for feature importance that we will use. Once we apply this method, we get Fig.1. As we can see, there are lots of features that are not as important as others. Therefore, we will only pick the top ten highest features from the dataset. Number ten is just an arbitrary number that was picked for this report, we could use more or fewer features. By picking the top ten features, we get Fig.2, and if we carefully observe Fig.1 and Fig.2, we see that the “State” column is missing. The state column is a categorical variable of nominal type, we could have applied label-encoder, but that would create ordinal values, and there is no natural ordering, therefore, it wouldn’t make sense. The other option is to apply one-hot encoding, however, this would create many dummy variables, as there were many states in this dataset. Before dropping the state column, a descriptive analysis was done on this column, which is made clearer in the next section. In the Jupyter notebook, I have also included a different way which can be used to show correlations between features using the Seaborn heatmap.

D. Handling Imbalanced Dataset

The reason we need to balance the training dataset is that our model could output biased results towards one particular class label. This dataset was highly imbalanced in terms of target class variables, as shown in Fig.3. The dataset was first split into 70% (3,190 instances) training, and 30% (800 instances) testing, after that, using Synthetic Minority Over-sampling Technique (SMOTE) [6], we oversample the training dataset. This method generates new samples in the classes which are under-represented. We also could have applied under-sampling, but by using this method, we would lose lots of valuable and important information about our customers. After the over-sampling is applied, we get balanced target class variables as shown in Fig.4.

E. Checking any Outliers

To make sure there are no outliers, a linear project was made using orange. This is also an important step because we don’t want any extreme values in our dataset. In Fig.5. we can see that are no outliers.

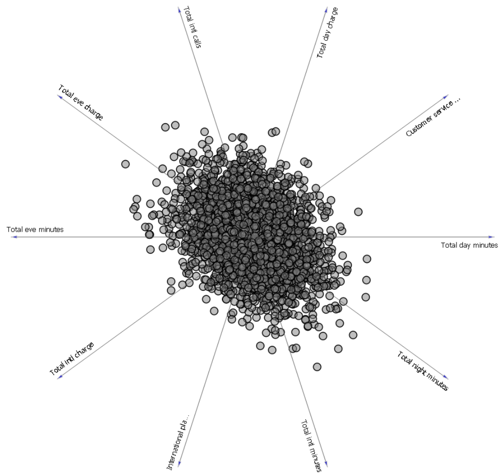

Fig. 5. Linear project of customer activities

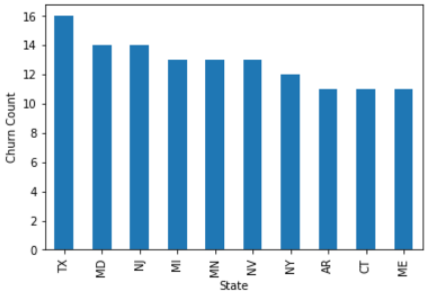

Fig. 6. Churn count per states

III. DATA DESCRIPTION

After the data preprocessing step, we are left with the following attributes shown in Table II.

As mentioned earlier, the state column was dropped from the final dataset for various reasons. We can still get valuable information before dropping this column. For instance, Fig.6 shows customers who did cancel the subscription by state. Keep in mind, that only the top ten states were picked for the analysis. Carefully, looking at Fig.6, we see that the company should really pay attention to Texas, as their churn counts are very high. Perhaps, give better deals to customers living in Texas.

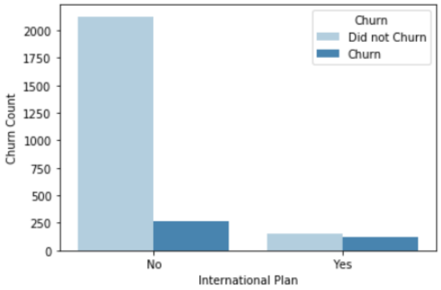

Fig. 7. Churn count per international plan

TABLE II
CUSTOMER ACTIVITY

Attribute	***Type***	***Example Value***	***Description***
Total day minutes	Numeric (float)	265.1	Total number of minutes used during the day
Customer service calls	Numeric (integer)	1	Total calls made to customer service
Today day charge	Numeric (float)	45.07	Total charge for the day
Total intl calls	Numeric (integer)	3	Total international calls made by the customer
Total eve charge	Numeric (float)	16.78	Total charge for the evening
Total eve minutes	Numeric (float)	197.4	Total number of minutes used in evening
Total intl charge	Numeric (float)	2.70	Total international charge
Total intl minutes	Numeric (float)	10.0	Total international minutes used
International plan_No	Numeric (integer)	1	Customers with no international plan
Total night minutes	Numeric (float)	244.7	Total number of minutes spent at night
TargetCol	Numeric (integer)	0	Target Column to predict

Some other analysis we can do with this dataset, is shown in Fig.7. It shows that the customers with an international plan are choosing to opt-out from the plan at a level that is almost as close as not churn. Ideally, the bar plot should look like the left-hand side, where the count of not churn is very high then the churn count. This shows that customers of this company do not like the international plan as much, and for a business to retain those customers, perhaps they would consider decreasing the international charges from their plan.

There is much more analysis we could do with this dataset, however, for this report, only two were shown as an example.

IV. LOGISTIC REGRESSION

The reason for using this method is simply because we are trying to predict a binary value, which is nothing but a categorical value, and logistic regression is one of the techniques that can help us predict binary value. We will use the Sklearn module that has a linear model [7] class in it. Internally, this class implements regularized logistic regression using the Liblinear library [8]. It is an open-source library for large-scale linear classification. This library not only supports logistic regression, but also linear support vector machines.

First, we will pass the balanced training dataset to fit the model, after that, we simply call predict method by passing our testing dataset. Let us first examine, the confusion matrix. It will give us some more insights into where our model is making the errors based on the predictions it made. The

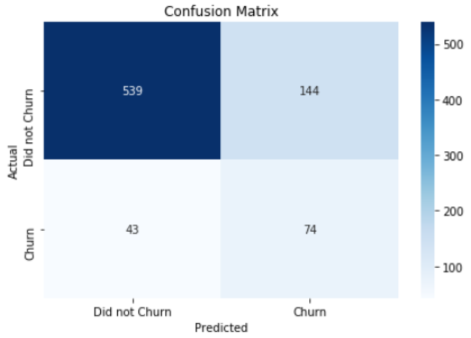

Fig. 8. Logistic regression confusion matrix

confusion matrix is also easily generated by passing the testing class labels and predicted class labels to the confusion matrix method present in the metrics module. Even though the output of the confusion matrix is a 2d array, we can plot the array on a heatmap as shown in Fig.8.

TABLE III
DERIVATION FROM CONFUSION MATRIX OF LOGISTIC REGRESSION

Class	***Values***
TP	74
TN	539
FP	144
FN	43

Let us interpret the confusion matrix with the help of Table III. Each of the four-boxes given in Fig.8 of the confusion matrix are associated with a label given in Table III. True positives (TP) indicate that in 74 cases, the classifier correctly predicted that a customer has churn. True negatives (TN) indicate that in 539 cases, the classifier correctly predicted that a customer did not churn. False positives (FP) indicate that in 144 cases, the classifier incorrectly predicted that a customer did churn but in fact, they did not churn. False positives are also known as Type I error. False negatives (FN) indicate that in 43 cases, the classifier incorrectly predicted that a customer did not churn, but in fact, they did churn. False negatives are also known as Type II error. Looking at the numbers of the confusion matrix is not very helpful. Therefore, let's compute some metrics using the confusion matrix; these metrics will help us choose between the models. Some metrics given in Table IV are calculated using formulas, they are all provided in the Jupyter notebook.

The classification accuracy is the overall accuracy of how often the classifier is correct. In our case, it is 76.6% accurate. The classification error is the opposite of the accuracy, it indicates how often the classifier is incorrect, which in this case, it is 23.3%. The sensitivity or recall indicates how often the predicted value is correct, given the actual value is positive, in this case, it is 63.2%. Sometimes, recall or sensitivity are also referred to as a true positive rate. Specificity indicates how often the predicted value is correct, given that the actual value is negative, in our case it is 78.9%. Sometimes, specificity is also referred to as a true negative rate. Precision indicates how often the predicted value is correct, given when a positive value is predicted, it shows how precise the classifier is when predicting positive instances, in this case, it is 33.9%. The f1-score is the harmonic mean of precision and recall, in our case, it is 44%.

TABLE IV
METRICS COMPUTED USING LOGISTIC REGRESSION CONFUSION MATRIX

Metrics	***Values***
Accuracy	0.76625
Classification Error	0.23375
Sensitivity/Recall	0.63247
Specificity	0.78916
Precision	0.33944
F1-score	0.44

Using the confusion matrix, we have derived various classification metrics, but now the issue is how do we decide which metric is the most useful for our dataset. The obvious case here is that we are not allowed to pick all of the metrics, as they are correlated with each other. Changing one metric is going to modify all the other metrics. Therefore, the question is how would we pick a metric that can maximize our business goals. In our case, we are trying to predict churn to retain customers.

Let us take a look at the Type I error (FP) and Type II error (FN). Earlier, we learned that we have 144 cases, where the classifier incorrectly predicted that a customer did churn but in fact, they did not churn. Therefore, from the business point of view, I would imagine that cases like FP would be more acceptable than FN. The goal is to minimize Type II error or FN. Now that we know which matrix to focus more sharply on, we can adjust the threshold in such a way where our model is more sensitive towards true values or churns.

Scikit learn provides a method which is called predict_proba that computes the probability of each class, and then chooses a class with the highest probability as the predicted class label. As it turns out, the default value for the threshold is 0.5. Therefore, in our case, if a probability exceeds 0.5, the class label is predicted as true, otherwise, false. In Fig.9, we can see that most of the values are below 0.5 thresholds, therefore customers who did churn are hardly predicted. In our case, we can lower the threshold to about 0.3, hence increasing the sensitivity towards true values. Internally, by increasing the sensitivity, we will decrease the type II error, which is what we want to accomplish. In order to lower the threshold, we will use Scikit learn binarized [9] method, which will return true, if the probability is above 0.3, and false otherwise. Before we look at the new confusion matrix generated using a new threshold, I want to mention an alternative that can be helpful to see how sensitivity and specificity are affected by different thresholds. The receiver operating characteristic curve or ROC curve helps us see the difference between the true positive and false positive rates using different probability thresholds as shown in Fig.10. It can also show us the area under the curve

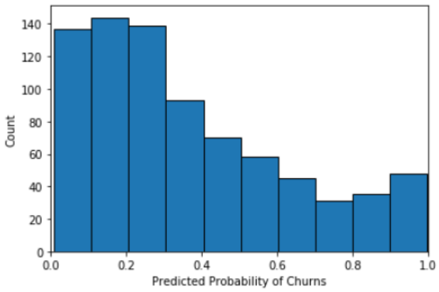

Fig. 9. Predicted probability of churns

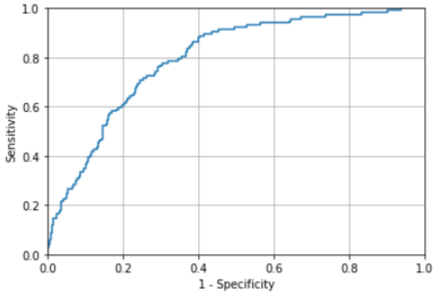

Fig. 10. ROC curve for churns

(AUC). Generally speaking, the higher the area, the better the classifier, and for the logistic regression, the AUC turned out to be 79.6%.

Let's examine the confusion matrix generated using 0.3 thresholds. If we take a look at Fig.11, we can see that it has successfully managed to decrease the Type II error (FN), this means that our model is now more sensitive towards the true values. Table V compares the previous threshold and the new threshold. It is very apparent that 0.3 thresholds have increased the sensitivity, and decreased the specificity. As mentioned earlier, these metrics are correlated, to be more specific, sensitivity and specificity are negatively correlated with each other.

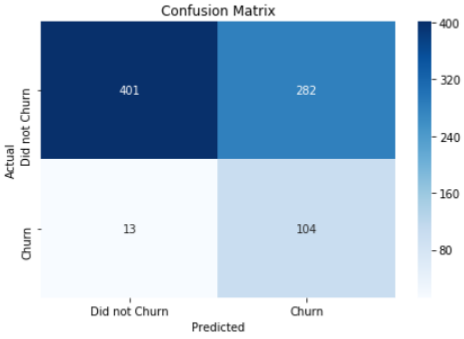

Fig. 11. Logistic regression confusion matrix

TABLE V
THRESHOLD COMPARISON

Threshold	*Sensitivity*	*Specificity*
0.5	0.63247	0.78916
0.3	0.88888	0.58711

V. SUPPORT VECTOR MACHINE (SVM)

The second method is Support Vector Machine or SVM, and it will help us classify our dataset into two classes; mainly true or false, by finding the best hyperplane that will separate all the points. Sklearn provides a class called SVC for classification, and internally it uses Libsvm [10], it is an open-source machine learning library for SVM classifications. Sklearn provides multiple kernels, essentially it is a type of algorithm SVC uses to predict the class value. Annexed is a list of algorithms: linear, poly, RBF, sigmoid, and precomputed. After many trials and errors, RBF worked the best for this dataset. Radial basis function (RBF) essentially projects points into an infinite-dimensional space to become non-linear rather than linear. Similar to logistic regression, we first pass the balanced training dataset, and then call the predict method by passing the balanced testing dataset.

TABLE VI
DERIVATION FROM CONFUSION MATRIX OF SVM

Class	*Values*
TP	91
TN	608
FP	75
FN	26

TABLE VII
METRICS COMPUTED USING SVM CONFUSION MATRIX

Metrics	*Values*
Accuracy	0.87375
Classification Error	0.12624
Sensitivity/Recall	0.77777
Specificity	0.89019
Precision	0.54819
F1-score	0.64

Let us interpret the confusion matrix with the help of Table VI. Each of the four-boxes given in Fig.12 of the confusion matrix are associated with a label given in Table VI. True positives (TP) indicate that in 91 cases, the classifier correctly predicted that a customer has churn. True negatives (TN) indicate that in 608 cases, the classifier correctly predicted that a customer did not churn. False positives (FP) also known as Type I error indicate that in 75 cases, the classifier incorrectly predicted that a customer did churn but in fact, they did not churn. False negatives (FN) also known as Type II error indicate that in 26 cases, the classifier incorrectly predicted that a customer did not churn, but in fact, they did churn. Table VII shows all the different metrics that were calculated

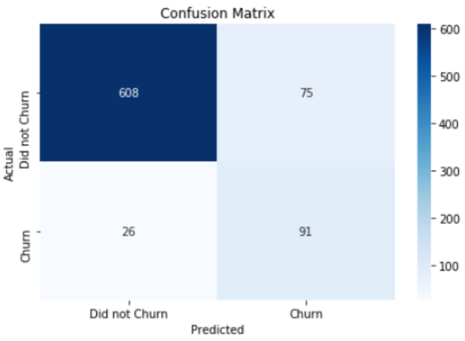

Fig. 12. Support vector machine confusion matrix

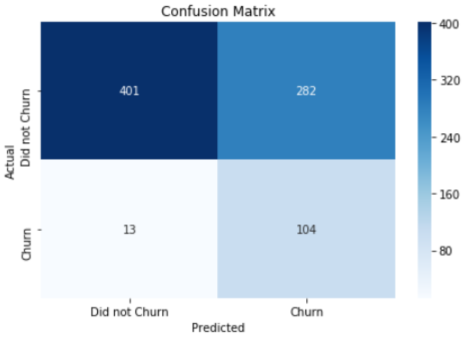

Fig. 13. Logistic regression confusion matrix with 0.3 threshold

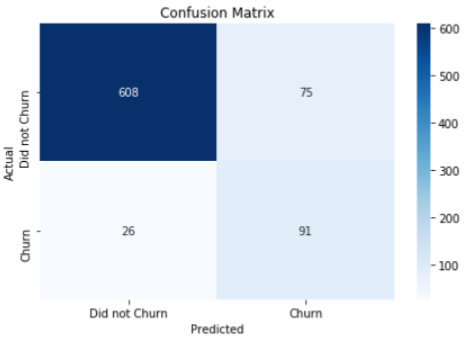

Fig. 14. Support vector machine confusion matrix

using the Sklearn metrics module. Since we have already discussed what these metrics mean, we won't go over them again. However, in the next section, we will compare these metrics for detailed analysis.

Before we compare both the methods, it is very important to keep in mind that for this dataset, we are trying to achieve high sensitivity and low specificity. As we have already learned that we are more acceptable to false positives than the false negatives.

VI. RESULTS AND DISCUSSION

In this section, we will compare the confusion matrix that was generated by both the models along with different metrics. Between Fig.13 and Fig.14, we can clearly see which model performed better in terms of Type II error (FN). As a reminder, our goal all along is to decrease the Type II error because the acceptance of Type I error is okay in our case. In the example of logistic regression, when we lowered the threshold, most of the values moved to the right column. Having said that, the SVM is also not a bad model, as it not only has low Type I error, but also Type II error. Keep in mind, that we did not alter any other properties to predict our classes for SVM. This flow was chosen to simply demonstrate the performance of SVM on this dataset.

For further analysis, let us compare the metrics of both the models. Jupyter notebook shows how these metrics were generated using different formulas. Even with the help of the confusion matrix, we know that by using logistic regression, we were able to lower the recall. Let's start by comparing the accuracy. SVM has an accuracy of about 87%, and in comparison, logistic regression has only 63%. Does this mean that SVM has better performance? Suppose we never balanced the dataset, in that case, if we were to predict the customers that do churn, our accuracy would be very low because as shown in Fig.3, the number of true values is much lower than false values. Therefore, we can't always rely on accuracy. Despite balancing the dataset, SVM seems to perform much better than the logistic regression. The classification error is simply one minus the accuracy, therefore, the better the accuracy, the lower the classification error. Since in this example, SVM has better accuracy, it has lower classification error. Precision is another metric that is shown in this figure, and it is usually used to minimize the false positives or Type I error. In our dataset, it wouldn't make sense to minimize the Type I error. One real-world example would be spam filtering, where the goal is to minimize the actual emails that are classified as spam. F1-score is a useful metric when we need both precision and sensitivity taken into account. For this dataset, we have already decided that optimizing sensitivity is our major goal, this also means that the model will predict most customers to be in the positive class.

When examining different metrics for a model having every value to a high number does not mean the model is accurate

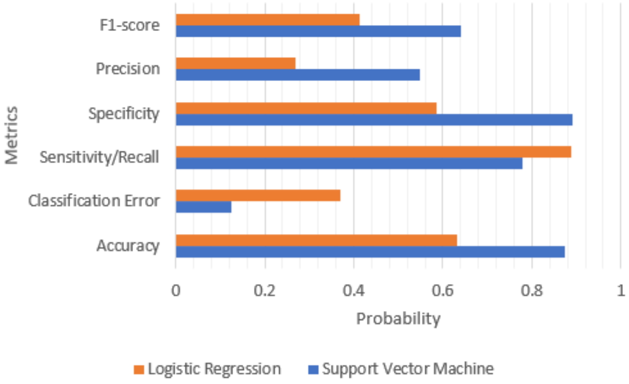

Fig. 15. Metrics comparison for models

and will perform better. We have to first look at what we are trying to achieve and then based on our goal we should pick a metric that makes the most sense for our business. In our example, increasing the sensitivity made the most sense, and it turns out, logistic regression gave us the best sensitivity or recall, therefore, we can conclude to using this model going further.

VII. Conclusion

Before we used any method, we first balanced the dataset as it is very important, otherwise, the prediction could show bias results. The first method used in this report is Logistic Regression, and using the confusion matrix, we derived many metrics which helped us further evaluate our model. We came to the conclusion that, by lowering the default threshold, the model became more sensitive towards true values and that was the main goal. The second method we used, is the Support Vector Machine or SVM, in addition, RBF was used as the kernel for this particular dataset. Similar to logistic regression, the results of the predicted values were output in a confusion matrix, along with a list of metrics.

This report has mentioned a number of times that the sensitivity was our main focus, and using logistic regression along with SMOTE, we successfully increased recall, and invariably decreased the specificity. Using this model, we will be able to predict if the customer will churn or not. If the output shows true, meaning the customer will churn, then we can analyze their activities which will further explain the reason behind their churning. Overall, perhaps we can provide them with better or extra services. In the beginning, we saw how some states have more churns than the others, we can use this information to our advantage, and further make our service better in those states. Usually, in businesses like this, pricing is the main culprit, because if customers find a cheaper service, then they may churn. Therefore, maybe this business should update their tactics for more competitive and reasonable pricing, or maybe provide some incentives to customers if they don't want them to churn. Businesses should rely heavily on analyzing this data, improving their existing customer's communication; because as the market shows, and the data proves, existing customers are the ones you need to focus on and manage with the utmost importance. The cost of retaining existing customers is 7x-10x lower than acquiring new ones [11]. Reducing churn, customer loyalty, should be considered top priorities in a subscription business, or any business, for that matter. Analyzing this data could lead businesses to offer incentives, to offer "pause" subscriptions, and to re-engage customers. Once you have established a customer or a subscriber, you should have methods in place for pleasing customers, incentives, promotions, etc., and it should be a top goal for that business. This data can accelerate time and insight to improve retention or upselling for existing clients. Research shows, that just a 5% reduction in customer churn can increase profits by 20% or more [11]. Getting the data, analyzing, and proactivity is the answer for businesses and clients.

References

[1] "Orange S.A." Wikipedia, 9 June 2020, en.wikipipedia.org/wiki/Orange S.A.

[2] B. Mnassri. "Telecom Churn Dataset." Kaggle, 5 July 2019, Kaggle.com/mnassrib/telecom-churn-datasets/metadata.

[3] Unknown, bml-data.s3.amazonaws.com/churn-bigml-80.csv.

[4] "Pandas get dummies." Pandas 1.0.3 Documentation, pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get dummies.html.

[5] "An Extra Tress Classifier" Sckikit learn 3.2.4 ensemble extra trees classifier, scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html.

[6] "Imblearn over sampling SMOTE" Imbalanced learn, imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over sampling.SMOTE.html.

[7] "Sklearn linear model logistic regression" Scikit learn logistic regression classifier, scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[8] R. Fan, K. Chang, C. Hsieh, X. Wang, C. Lin, "A library for large linear classification.", 8 May 2008, csie.ntu.edu.tw/cjlin/papers/liblinear.pdf.

[9] "Sklearn preprocessing binarize" Scikit feature binarization, scikit-learn.org/stable/modules/generated/sklearn.preprocessing.binarize.html.

[10] C. Chang, C. Lin, "A library for support vector machines", ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011, csie.ntu.edu.tw/cjlin/libsvm.

[11] "Partner with Dunn Solutions to Improve Customer Retention", Dunn Solutions, visit.dunnsolutions.com/customer-churn-reduction.