Politecnico di Milano
School of Management

Master's in business Analytics and Big Data (BABD)

Machine Learning & Business Applications
ML Classification Assignment

Students: Camila Rigone, Marco Iannotta, Joaquin Bembhy

Professors: Mauricio Soto & Andrea Mor

1. Introduction

1.1 Data Preparation

The first step was to import the dataset into the jupyter notebook. We then visualized it and realized that there was an ID column which is supposed to be unique for each customer. We checked unique values for ID and got 49,999. We proceed to check for duplicates and saw that two different customers had the same ID which is an error. Given this, we concluded that no customer did more than 1 transaction, therefore it was insignificant for the model, so we dropped the ID column.

Then we proceed to preprocess the data. We checked for missing values, we balanced the dataset, we split into categorical and numerical values, analyzed it to check for patterns and to extract important insights, we standardized our data and finally we separated into train and test.

1.2 Missing values

We checked for missing values because the models we want to run in the future can't understand null values. For this we checked NaN for each category and discovered that almost 8% of the age observations were missing. We tried three different approaches: first we replaced the nulls with the mean, then we filled in the mean age of people by grouping by customer type and satisfaction, and, at last, we dropped the values. From these three approaches, we chose the last one because it provided us with the higher F1 Score and ROC curve.

After this, we changed the target variable from "Satisfied" and "Not Satisfied" to be 1 and 0 respectively. This allows us to insert the target variable into models, but also will help us in the next step of balancing.

1.3 Balancing

We balance our target variable in the dataset in order to give equal priority to each class. If we have unbalanced data, the model could assign more weight to the majority category. In order to do this we used a downsampling technique, given that our dataset was very large and it wouldn't affect our model scores.

At first we ran the models with the unbalanced data, and later on we balanced it and ran it again. We got that the models were performing better with balanced data. A comparison table will be displayed later on in this report.

1.4 Split into categorical and numerical values

In order to be able to run the model, we must convert categorical values into dummies. For this, we did an exploration of our variables checking first the type of the columns, but also we checked the number of unique values for each variable. The variables that were of type

integer but had only 5 or 6 unique values, were considered categorical, together with the object type columns, which were converted into dummies.

After this we performed an analysis of the numerical and categorical variables to search for patterns that were going to help us keep variables that were significant for the modeling.

1.4.1 Categorical

- Category: doesn't help us predict much.
- Customer type: Premium users have a higher rate of satisfaction than not premium users. It is a good predictor.
- NewUsed: doesn't help predict at all.
- Category: it is a bad predictor.
- Product description accuracy: seems that the higher, the higher dissatisfaction rate, but not very highly differentiable.
- Manufacture sustainability: not a clear trend, but could help predict
- Packaging quality: the higher, the better satisfaction. It is a good predictor.
- Additional options: the higher, the better satisfaction. It is a good predictor.
- Helpfulness and review of ratings: the higher, the better satisfaction. It is a good predictor.
- Integrity of packaging: the higher, the better satisfaction. It is a good predictor.
- Ease check-out procedure: the higher, the better satisfaction. It is a good predictor.
- Relevance of related products: the higher, the better satisfaction. It is a good predictor.
- Customer insurance: the higher, the better satisfaction. It is a good predictor.

1.4.2 Numerical

After analyzing the categorical variables we shifted to the numerical ones, those are: *Price*, *Age*, *Shipping delay in days* and *Arrival delay in days*. We also kept satisfaction between the variables to be able to display the pairplot.

The first thing we noticed was that all of them except for *Age* had an exponential distribution, so we converted those variables transforming these into logarithms. Thanks to this step *Price* improved a lot, the other two not much. We decided to make this so the model would give the same weight to the observations and not overweight those on the tails.

In the end we plotted the pairplot, in that we observed that there were some variables that help us differentiate by values the satisfaction of the customers. For example, low *Age* and high *Price*, seems to indicate higher satisfaction in the customer.

1.5 Standardization

Then we displayed a boxplot and realized that variables varied significantly on their scale. To solve this problem, in order for our model to take equal consideration of our variables, we standardized it using the standard scaler from sklearn.

1.6 Separate into train and test

Before passing into the construction of the model we concatenated the processed numerical and categorical dataframe and later separated our dataset into train and test to be able to provide a score for the models we tried in the next steps.

2. Modelling

It is important to note that the results provided are the best we could achieve. We discard the result we got both by replacing the *Age* null values instead of dropping and by deleting two categorical variables (*NewUsed* and *Category*). We decided not to proceed with those two approaches because both the f1 scores for train and test, and the AOC in the ROC curve were lower.

2.1 KNN

For the KNN model we conducted a GridSearch in order to find the best parameters for the model. We started in a big range but quickly realized that the best numbers were between 10 and 20. We did a GridSearch again for a lower range and found out that the model performed the best with 13 nearest neighbors. (The total scores will be illustrated at the end).

2.2 Decision Tree

Also conducted a GridSearch in order to variate the parameters of criterion, max_depth, min_samples_split and min_samples_leaf. After running the search several times, the parameters for which we got the best results were: gini, 13, 9, 81 respectively.

2.3 Naive Bayes

This model at first gave as very low scores. It wasn't performing any good compared to the first two models that we runned. Anyhow, the f1 score increased significantly after downsampling. But even like this, the scores were much lower than expected.

2.4 Logistic Regression

The logistic regression model didn't perform that good either. It was in the middle point between the first two models and the third one tried. The best parameter for the C was of 0.3.

2.5 Support-Vector Machine

For the SVM, we had issues running the model several times for different parameters because it took a long time to finish it. Even though we could run it a couple of times and the performance that we got from this model was the best till the moment. Both the F1 and the ROC curve area were the best until this point.

2.6 MLP Neural Network

For this model, we tried extensively with different parameters, as the metric scores we started having were really high. The SVM had been the best model until the moment, but this model performed even better. We tried optimizing it the most we could to get the best parameters, because we knew that it had the potential to perform really well.

2.6 Random Forest

At this point of the analysis, we started playing with ensemble models. The aim was to see if with any of these types of models we could get higher scores.

For the random forest, at first we got low metrics, but after doing some parameter tuning, we got the second highest scores at the moment (after MLP). It cost a lot of computing power because we searched extensively, using a GridSearch, the best parameters for the model. We thought the model had the potential to perform better than the neural network model, but we couldn't get it to perform as well.

2.8 Adaboost

Given that the RandomForest had performed really well, we expected the Ada Boost to also follow this path. But to our surprise, we couldn't get parameters that made this model perform as good as other models.

3. Improving the model

3.1 PCA

Up until this point, we tried to run the models without doing a dimensionality reduction. For this means, we proposed ourselves to do a PCA and re-run the models with the dataset containing just the significant principal components, and analyze if the results improved.

Once we divided our data for the PCA, we realized that there was not a clear number of principal components that explained most of the variability of our model. Therefore, we decided to try with different amounts of components. After this, we tried running some of the classification models with the PCA dataset. For this, we got metrics much lower than the ones we had before (without PCA), therefore, we decided that the PCA wasn't a good way of improving our predictions.

4. Conclusion

4.1 Comparing the metrics

The following table provides the best score obtained for each model.

Model	F1 train	F1 test	AUC ROC
KNN	0.804	0.805	0.89
Decision Tree	0.823	0.828	0.90
Naives Bayes	0.729	0.729	0.81
Logistic Regression	0.753	0.751	0.83
SVM	0.819	0.824	0.90
MLP	0.840	0.846	0.92
Random Forest	0.830	0.830	0.92
Ada Boost	0.765	0.763	0.84

We can see clearly that the best performer overall is the MLP model. It has the highest F1 and ROC score for all the different models. We expect that with this model we can predict with the highest accuracy.

On the other hand, the Naives Bayes model is the worst, together with the logistic regression model. These are the models that performed worst with respect to the f1 score and the ROC curve.

It is also important to remark that none of the models was overfitted. This can be seen because the scores for the F1 are practically the same for the train and for the test datasets.

4.2 Final comments

With this analysis, we were able to clean the dataset, prepare it for our modeling, and finally create several different models that allowed us to predict if our customers were going to be satisfied or not with the service.

After conducting the whole investigation, we got some models that performed better than others, which will be kept to predict the satisfaction of customers in the future. We are confident that with the data wrangling, pre-processing, and modeling, we are going to be able to predict with a high level of accuracy the satisfaction of our customers.

Given what was demonstrated and explained in this report, we are going to move forward and try to predict the released dataset with the MLP Neural Networks model.