Thomas Henry

Statistical & ML Approaches for Marketing

Prof. Minh Phan

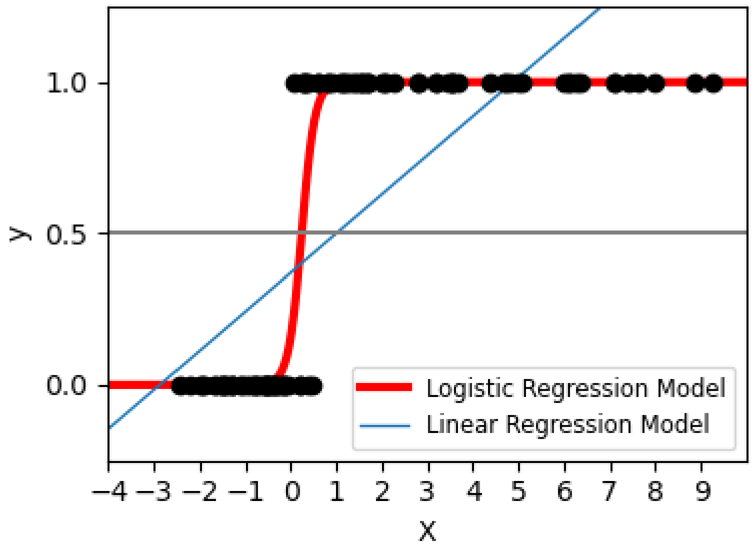
Sunday April 18th, 2021

Final Individual Report

In this report, five different classification algorithms will be discussed, including Logistic Regression, LDA, Gradient Boosting, Random Forrest and Bagging. These models were all used in the context of a Churn Dataset on Kaggle as a part of the annual in-class competition. With a problem being to classify Orange's customers as churners and non-churner, it seemed relevant to benchmark all most popular algorithms to find the most accurate results.

I) Logistic Regression Algorithm

The Logistic regression algorithm is one of the easiest ones to implement in machine learning and can sometimes outperform more advanced models depending on the dataset. The main idea behind a logistic regression algorithm is to predict a binary outcome (0 or 1 usually), and therefore can be used for classification problems such as churn prediction. The typical shape of a logistic regression curve involves a S shape, with the independent variables on the x-axis and the dependant variable on the y-axis. The y-axis also represents the probability that an event 0 or 1 will occur at that particular x. We can therefore observe a line in the middle of the graph, representing a probability value of 0.5, where all values above that line would be classified as 1 and where all values under that line would be 0s (of course, in a model where all predictions would be 100% accurate).



In a logistic regression problem, the concept of odds is very relevant and represents a ratio that is in favour or against a certain event taking place. In a sports betting context, the odds are used to logically multiply the amount of money that a player bets on a certain team based on the certainty that that team can win the game. Though, one very important concept is to understand that this "certainty" of winning the game is based on the level of the opposing team as well, meaning that the deeper the gap in ability between the teams, the more a team will be favoured compared to the other (in this context, the value will be close to 1, where a player will have trouble winning a lot of money). In the context of a logistic regression, the formula for the odds is P(X)/1 – P(X).

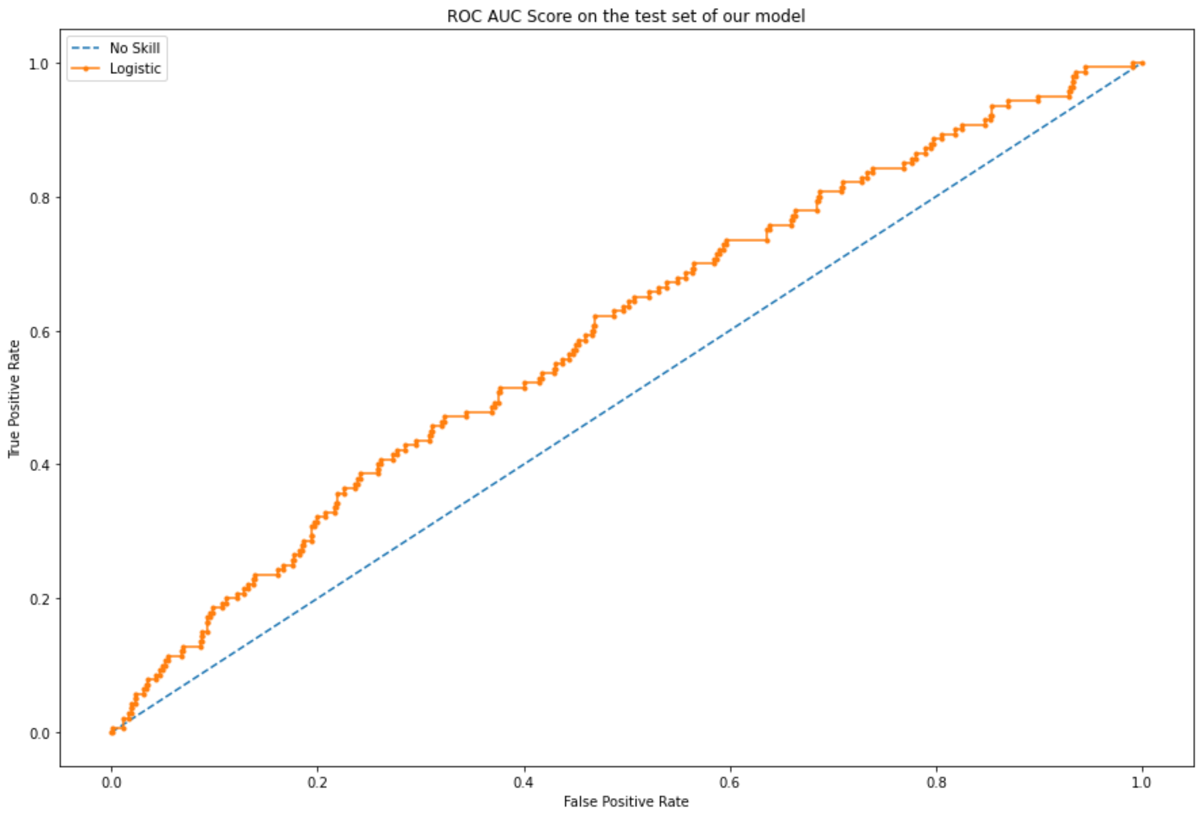
The logistic regression function, also called logit function, is based on taking the logarithm of the odds, which then yield the coefficients of the model, similarly to the linear regression function. Mathematically, this starts by using the probability function to obtain the odds formula where the logarithm is then applied in order to retrieve the coefficients.

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}. \Rightarrow \frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}. \Rightarrow \log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X.$$

As a way of estimating the best possible model, the logistic function uses the concept of Maximum Likelihood, which minimizes the distance between the actual and predicted values. To calculate the maximum likelihood function, all probabilities of an event are multiplied, which can then be transformed using the mathematical process shown above involving the logarithm of the odds.

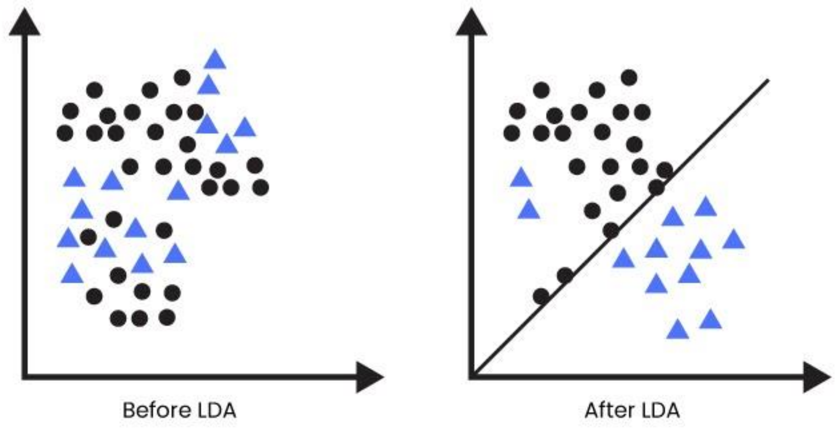
Application to the Kaggle Dataset

In the case of our in-class Kaggle competition, the logistic regression model was used in the first place, due to the simplicity of its implementation. Usually, it is used as a first glance into the predictions, providing with a fast measure of accuracy for the model. When running the logistic regression model on the Kaggle dataset, it was possible to observe that converting the variables using a standardization method allowed the model to perform significantly better. Although, this model did not yield great accuracy numbers, which can be explained by its lack of complexity compared to the dataset.



II) Linear Discriminant Analysis (LDA)

The linear discriminant analysis algorithm, or LDA, is a classification algorithm used when the responses classes are superior to two. This model consists into modelling the distribution of the predictors X separately for each of the responses (given Y). The model uses Bayes' theorem to transform the distribution into estimates for the conditional distribution of the response Y given the predictors X. Three main uses can be drawn out of the LDA models including classification tasks, dimension reduction, data visualization.



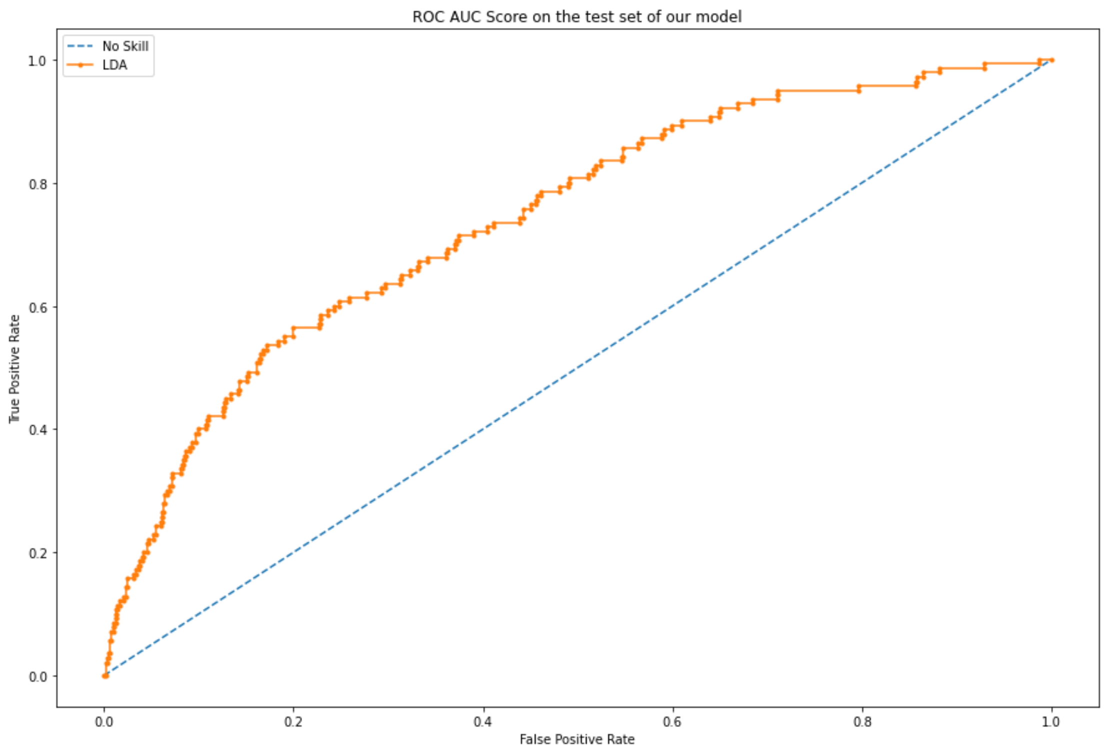
The LDA model uses a discriminant rule to divide the data space into K disjoint regions, where one region represents one class. Two main allocation rules are used in this model: the maximum likelihood rule (where we assume that each class would occur with equal probabilities, and the Bayesian rule (when the class of the probability is known prior). Therefore, the algorithm classifies the data observation to the class that has the highest likelihood.

The function used for LDA is the discriminant function (thus, being called linear discriminant analysis), which tells us how likely the data belong to each class. Additionally, the concept of decision boundary is very important as it represents the set of x values where two discriminant functions have the same value (equally possible to be from each class separated from the boundary). Each decision boundary corresponds to a single pair of classes, where the data space is divided into regions.

Note: The main difference lying between LDA and QDA (quadratic discriminant analysis) is that the latter does not require an assumption of equal covariance assumption, which prevents the quadratic term of the function to be cancelled out such as the LDA function.

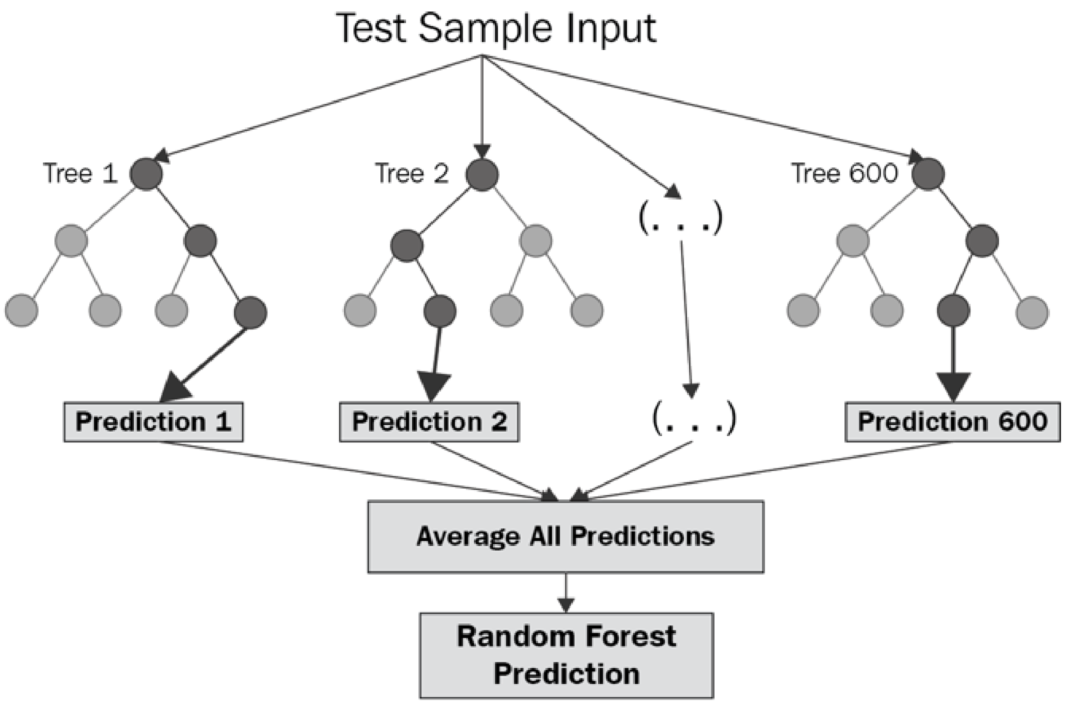
Application to the Kaggle Dataset

When using LDA on the Kaggle dataset, the results were quite satisfying when looking at accuracy measures but were disappointing when submitting the result on the platform. The “least squares” solver was found to the most performing, outscoring “svd” and “eigen”.



III) Random Forest

The Random Forest algorithm is a supervised model that can be used for both regression and classification problems. The main goal of this model is to base the prediction on multiple decision trees in order to get the best accuracy, by using different subsets of the dataset. The number of trees highly influences the prediction accuracy in the end, usually meaning that more trees will yield better results and reduce over and underfitting.

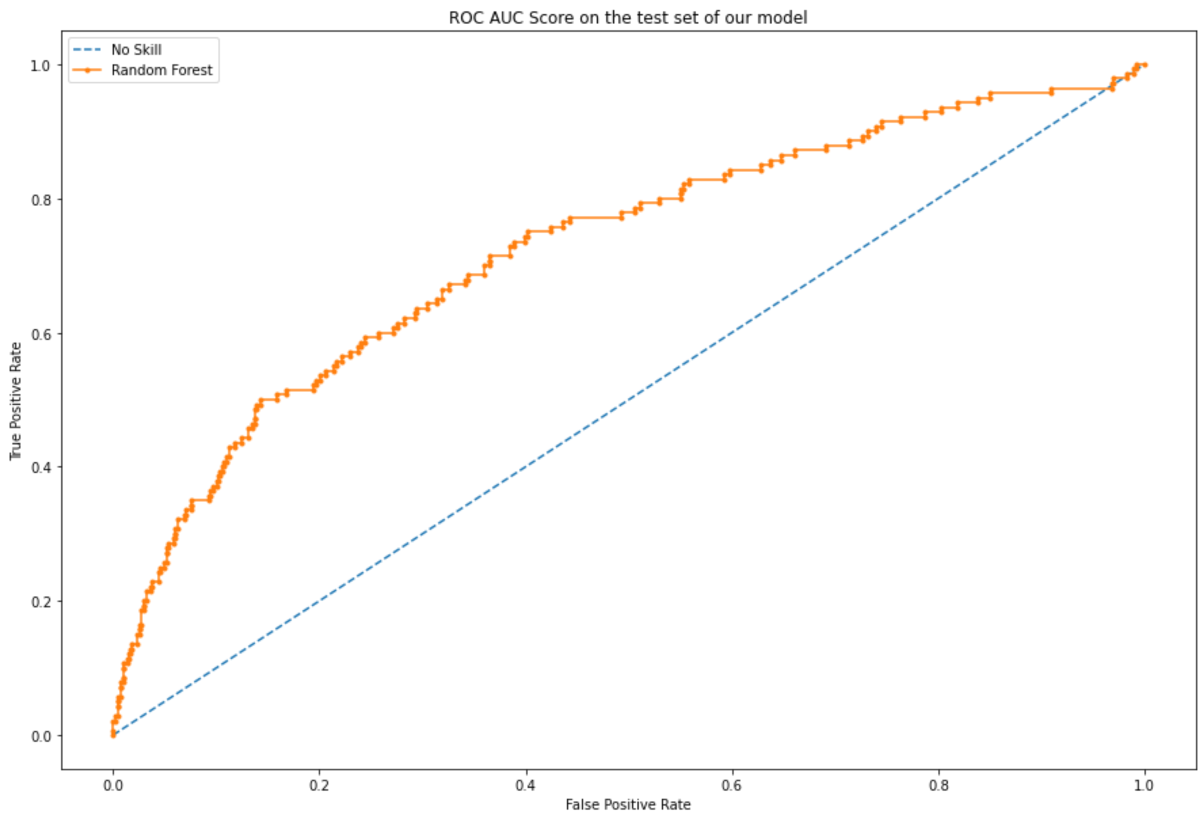


Random Forest algorithms are part of the ensemble learning family, where several decisions trees are built on a certain number of training samples created via the bootstrap method. The randomness of the classifier lies in the fact that each time a tree is created, random predictors are chosen for that split. For each split, one of the predictors is used to make a prediction, which will then be averaged to provide the final algorithm prediction.

The advantages of a random forest algorithm are such that the model can handle larger datasets with high dimensionality, but also work well when handling missing values. Additionally, random forest algorithms are very robust to outliers meaning that there is no need for treating them in the data pre-processing, which can be computationally expensive depending on the dataset. Unfortunately, the computational expenses of a RF model can make it very slow to train, especially when a cross validation is ran to find the most relevant parameters for the model.

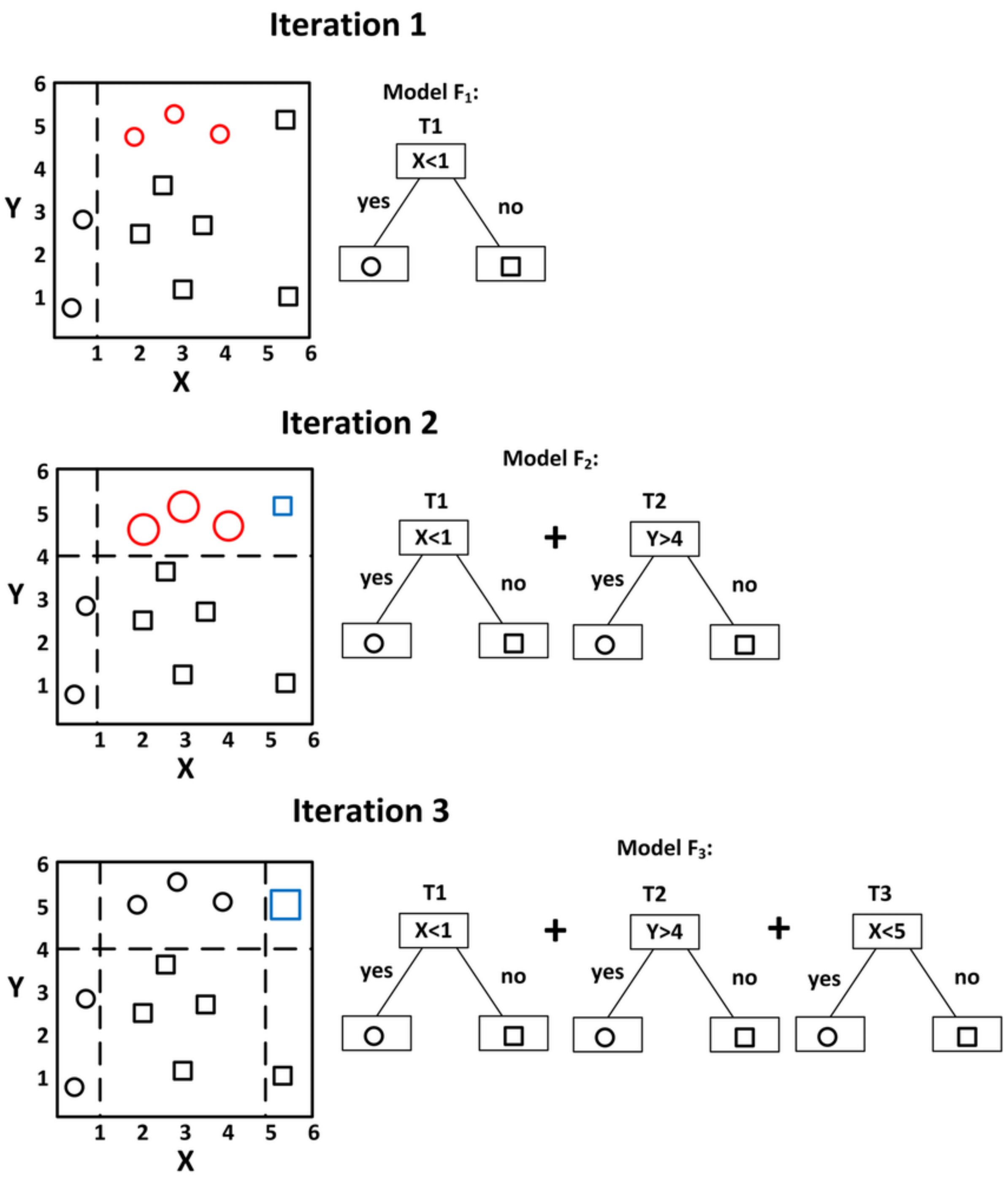
Application to the Kaggle Dataset:

In the case of the Kaggle dataset, the random forest algorithm performed fairly well since there was a high number of features. When modifying the hyperparameters (i.e., min_samples_leaf, min_samples_split and n_estimators), the performance of the model were slightly better than using the parameters by default. We observe that the AUC curve is showing good performance until the very end where the curve suddenly gets under the "No-Skill" line.



IV) Gradient Boosting

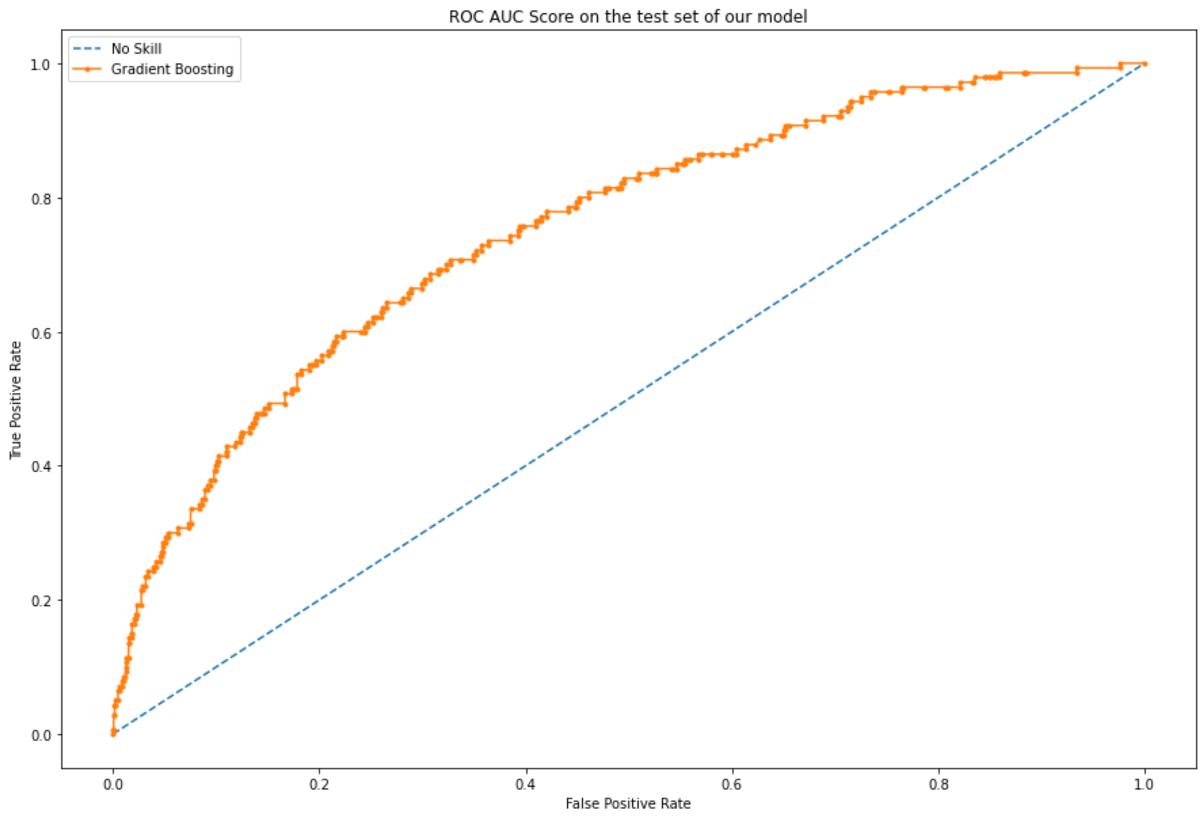
The gradient boosting algorithm is resembling Random Forest, as it can be used for both regression and classification problems and it is based on the concept of ensemble learning. In this case though, the bootstrapping method used on the training set for Random Forest is not used, and each tree is based on a version of the original dataset. Additionally, the randomness aspect of random forest does not appear here, as the goal is to grow the trees sequentially, by reusing all the information gathered in the previous splits. The prediction's accuracy is built on the residuals, and each subtree is built using these updated residuals. This allow for the error to be corrected as the algorithm progresses through the trees where each of the predictors end up being trained using the residual values obtained previously.



Gradient Boosting provides the same advantages than the random forest model since it reduces overfitting, allowing to get an easily interpretable prediction while performing at a high accuracy level. Unfortunately, the learning speed of the algorithm makes it very slow to train, especially when using hyperparameters tuning to improve performance. Additionally, gradient boosting models are more sensitive to overfitting when using noisy data, which means that considering outliers in the data pre-processing would be a great way of dealing with that issue.

Application in the Kaggle Dataset:

In the case of our competition, gradient boosting yielded the best results out of all the algorithms tried on the data. When tuning the parameters, performance increased even more, as the n_estimators parameter seemed to have the greatest influence on our dataset. When trying to improve the model further, the subsample parameter played a small role as well in improving the accuracy, while other parameters remained silent when trying to perform cross-validation on the sample.



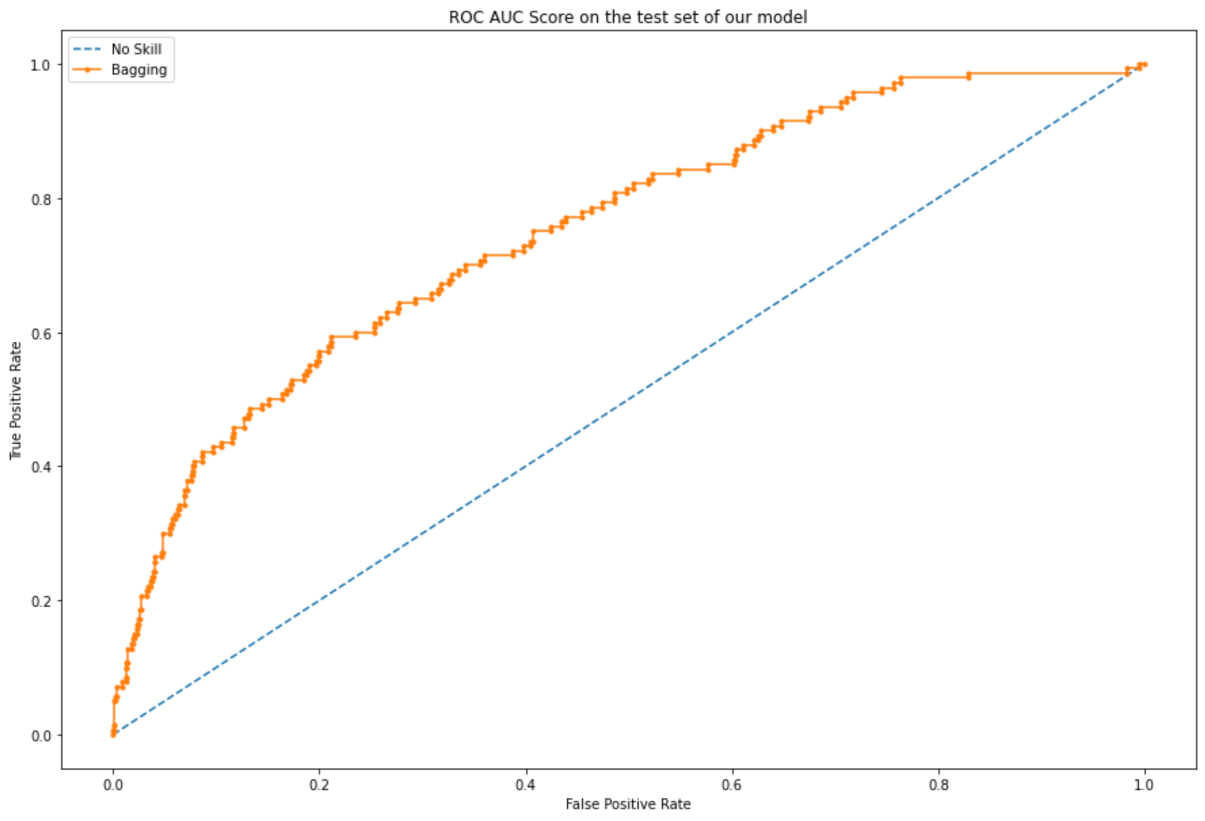
V) Bagging

Bagging algorithms are a part of the ensemble learning group and have the particularity of using a base classifier (which could be random forest or gradient boosting for example). After that base model, other models are used on top of the first one to refine the predictions and therefore try to provide a more accurate final result. Here, the bootstrap method is used just like in the random forest model, where random subsets from the original dataset are averaged to provide an accurate measure in the end.

The goal of a bagging algorithm is to reduce the variance as much as possible, which comes to the cost of having no insight on what is going on in the model. Therefore, we can classify this model as a black-box, and interpretability can seriously be affected by this issue.

Application to the Kaggle Dataset

In the case of the Kaggle dataset, bagging had a good performance, which did not translate into the Kaggle submission results afterwards. Therefore, the model was probably overfitted which could explain the poor performance when testing on unseen data.



VI) Project Description

After explaining each of the algorithms used in this Kaggle competition, it seems relevant to note a few ideas about the process used to obtain the best possible results. First, the data pre-processing part of the project was rather time-consuming considering the variety of variables for which we did not have any information. A process of dropping all columns using a 66% threshold for missing values was used, with the goal of reducing the number of features and create the most robust model possible.

Once the data was cleaned, a feature selection method was used using Pearson as the metric to select the variables. A value of 0.05 was retained as the p-value for the selection yielding a total number of 51 features, which corresponds to approximately 25% of the original number of variables.

Then, a benchmark experiment was conducted allowing to obtain AUC metrics as a graphical representation of the curves. This method was strengthened by the use of a GridSearch CV function using scikit-learn, as well as traditional cross-validation function using 10 k-fold CV. Once all models were tested, the best performing algorithms were used to predict the final probabilities of churning for all customers in the test data.

References

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2017). *An introduction to statistical learning: With applications in R*. New York: Springer.

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.analyticssteps.com%2Fblogs%2Fintroduction-linear-discriminant-analysis-supervised-learning&psig=AOvVaw0jdiQDvyPjIdeohR_-dFE7&ust=1618860263202000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCLiK3PDCiPACFQAAAAAdAAAAABAD

https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.researchgate.net%2Ffigure%2FA-simple-example-of-visualizing-gradient-boosting_fig5_326379229&psig=AOvVaw0h52Rp3lIqEDNlGtV2ndld&ust=1618855971674000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCKDG9ZXXiPACFQAAAAAdAAAAABAJ

https://medium.com/@aravanshad/gradient-boosting-versus-random-forest-cfa3fa8f0d80

https://www.google.com/search?q=random+forest&rlz=1C1GCEA_enFR924FR924&tbm=isch&source=iu&ictx=1&fir=ebtKFdkQ894YYM%252CVIl9g-DBZeStLM%252C_&vet=1&usg=AI4_-kQRMU2nIMIQQoJbta1jr74kJ5L7Bg&sa=X&ved=2ahUKEwih_d6hn4jwAhUBahQKHTb6D3cQ_h16BAgLEAE#imgrc=ebtKFdkQ894YYM