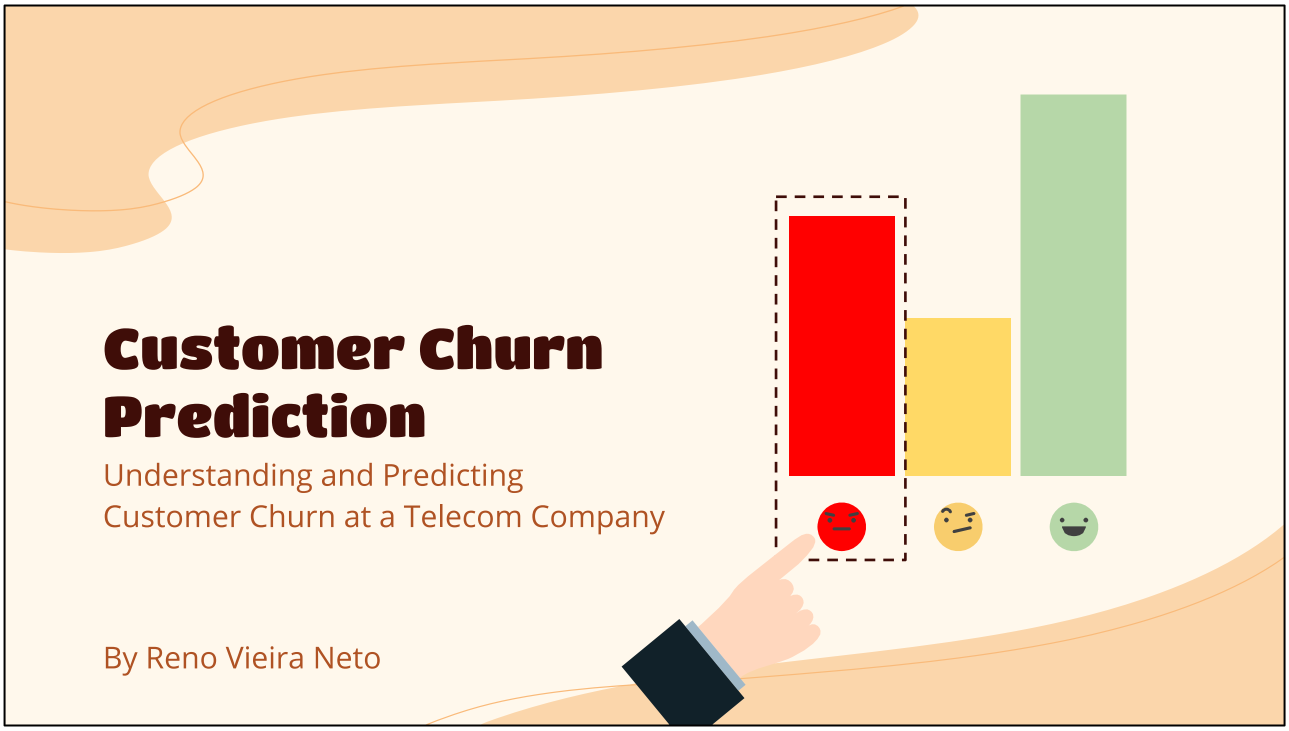
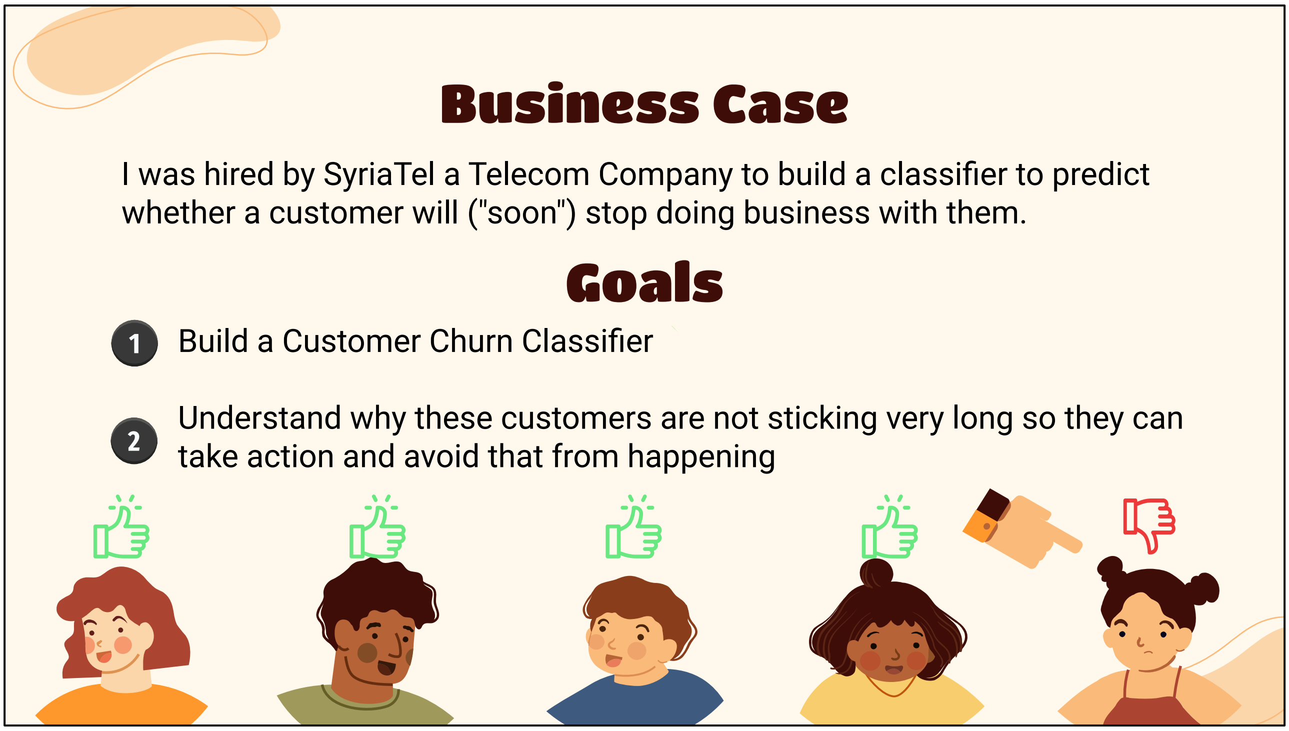


Customer Churn Prediction

Understanding and Predicting
Customer Churn at a Telecom Company

By Reno Vieira Neto

Hello, I’m Reno and in this project I’m talking about a model created to predict which clients are about to churn.



Business Case

I was hired by SyriaTel a Telecom Company to build a classifier to predict whether a customer will ("soon") stop doing business with them.

Goals

1. Build a Customer Churn Classifier
2. Understand why these customers are not sticking very long so they can take action and avoid that from happening

The Business Case is the following:
A Telecom Company hired me to create a classifier to predict whether a customer will churn. The business reason is that they are trying to reduce the No. of Churned Clients by identifying them before they churn with the expectation of avoiding the churn and/or make product/process enhancements to reduce the probability of churning.

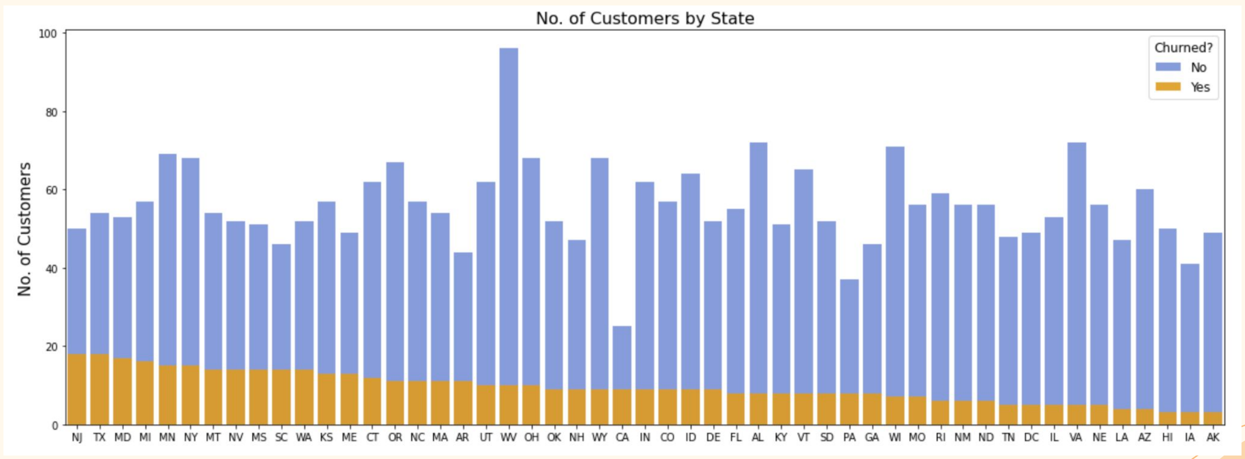
Therefore, my goals are:

1. Create a Customer Churn Classifier
2. Understand why these customers are churning and, based on that, identify opportunities to reduce the chance of churning.

Data Overview

Before jumping into the model, I will start by explaining the data a little, giving an overview of the dataset.

Some States have more Clients/Churned than others



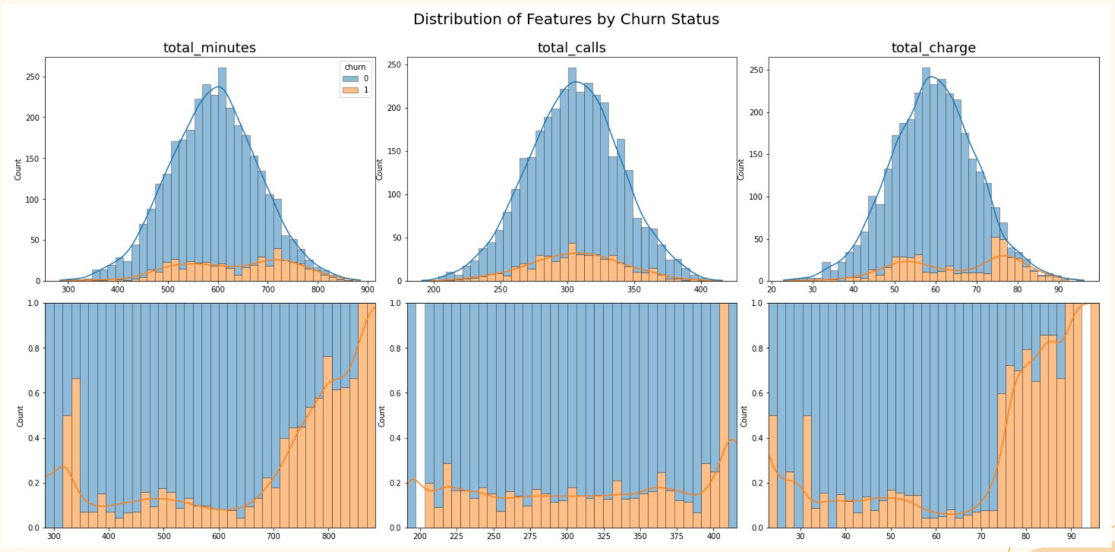
In this chart we see the No. of Customers by State by Churn Status. Blue Bars represent Non-Churned Clients, while Orange Bars represent Churned Clients. The chart is intentionally ordered by No. of Churned Clients (from highest to lowest, left-to-right).

In this chart it's possible to see two things:

1. Some States have more Customers than others
2. Some States have a higher Churn Rate than others

Therefore, the Customer's State might be an important feature to analyze while building the classifier.

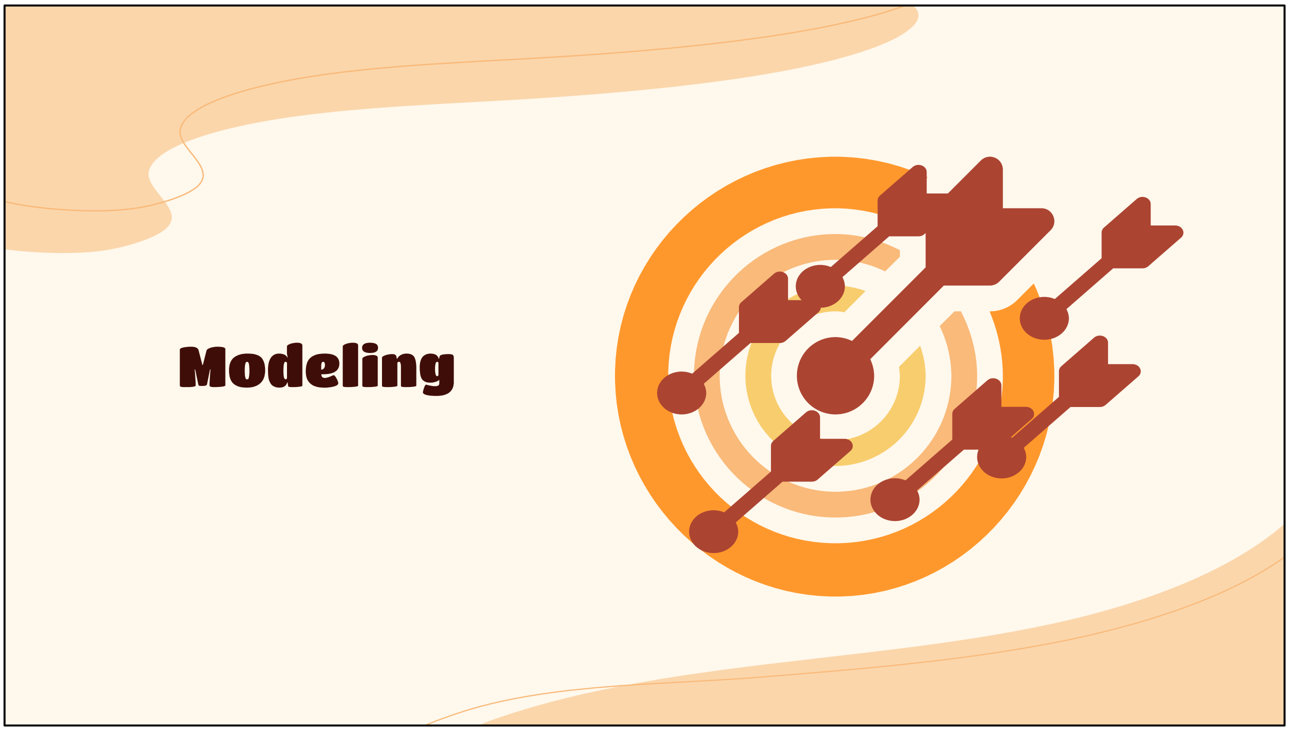
High Spenders are Churning



Here we have six charts. The Top Three Charts show the distribution of Clients by **Total No. of Minutes, Total No. of Calls and Total Charge**. Orange representing Churned Clients and Blue Non-Churned Clients. The Bottom Three Charts are 100% stacked bar charts. In these charts, it's straightforward to see at which values the probability of churning increases.

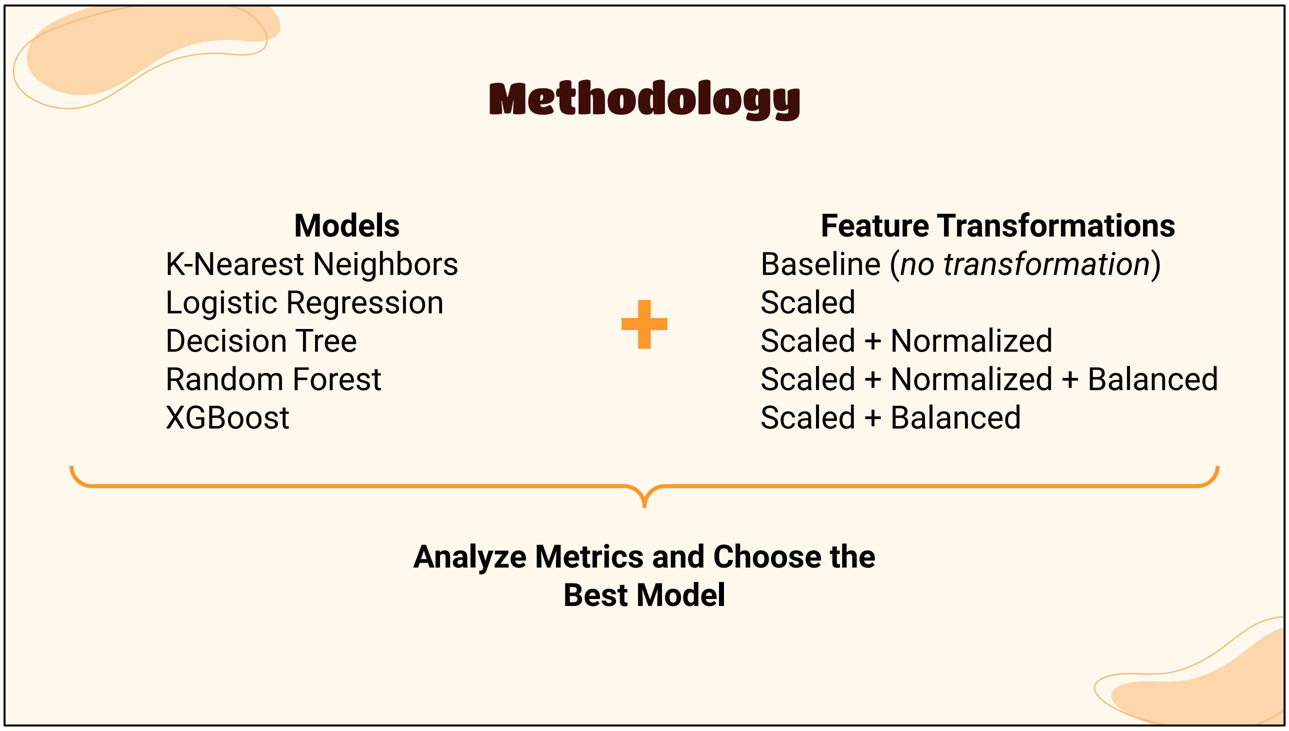
The conclusions from these charts are the following:

- Non-Churned clients have normal distributions in all three features. The same can't be said about Churned Clients.
- By looking at the stacked bar charts, it's possible to see an increase of Churned clients at high values of **'Total Minutes'** and **'Total Charge'**.



Modeling

Now it’s time to talk about the model, the approach, steps that I took and conclusions/results.



Methodology

Models

K-Nearest Neighbors
Logistic Regression
Decision Tree
Random Forest
XGBoost

+

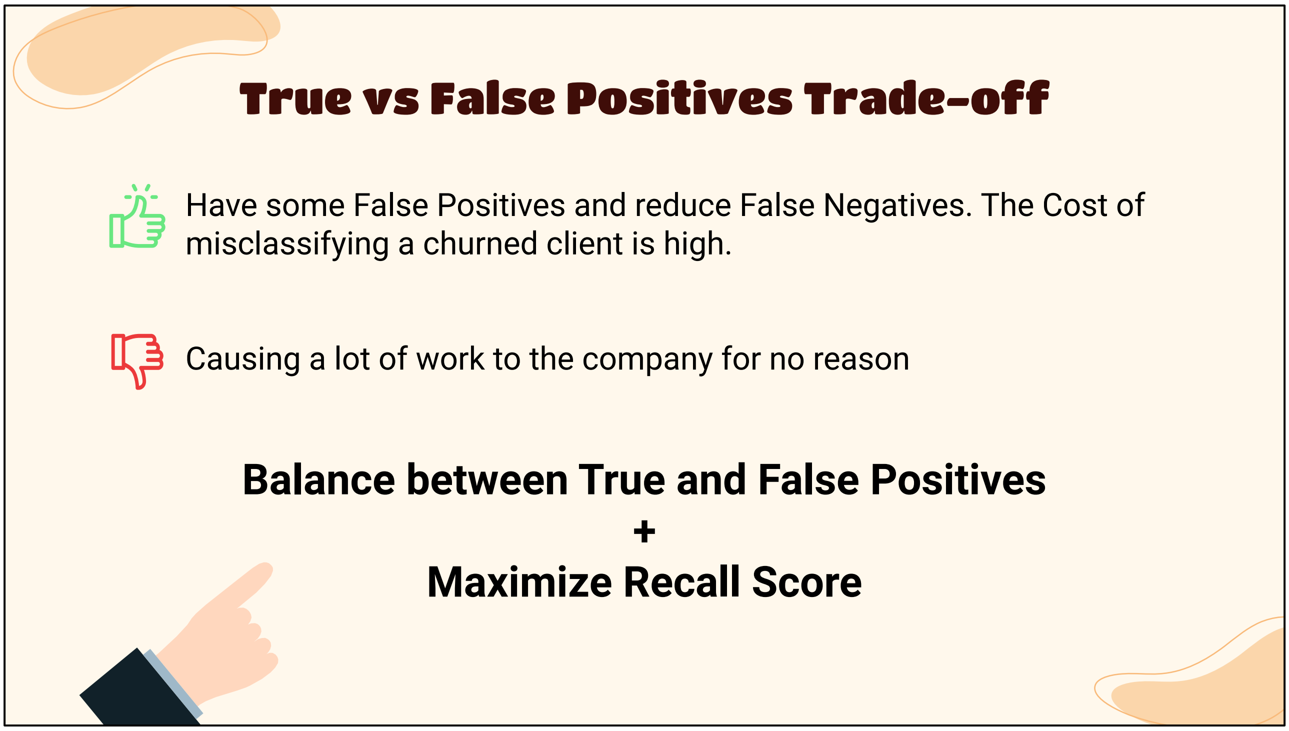
Feature Transformations

Baseline (*no transformation*)
Scaled
Scaled + Normalized
Scaled + Normalized + Balanced
Scaled + Balanced

Analyze Metrics and Choose the Best Model

The methodology for finding the best model was the following: I used a mix of five different models and five different feature transformations. The idea was to try various classifier models, see which feature transformations would perform better and find the best combination.

The best model then was chosen by analyzing performance metrics.



True vs False Positives Trade-off

- Have some False Positives and reduce False Negatives. The Cost of misclassifying a churned client is high.
- Causing a lot of work to the company for no reason

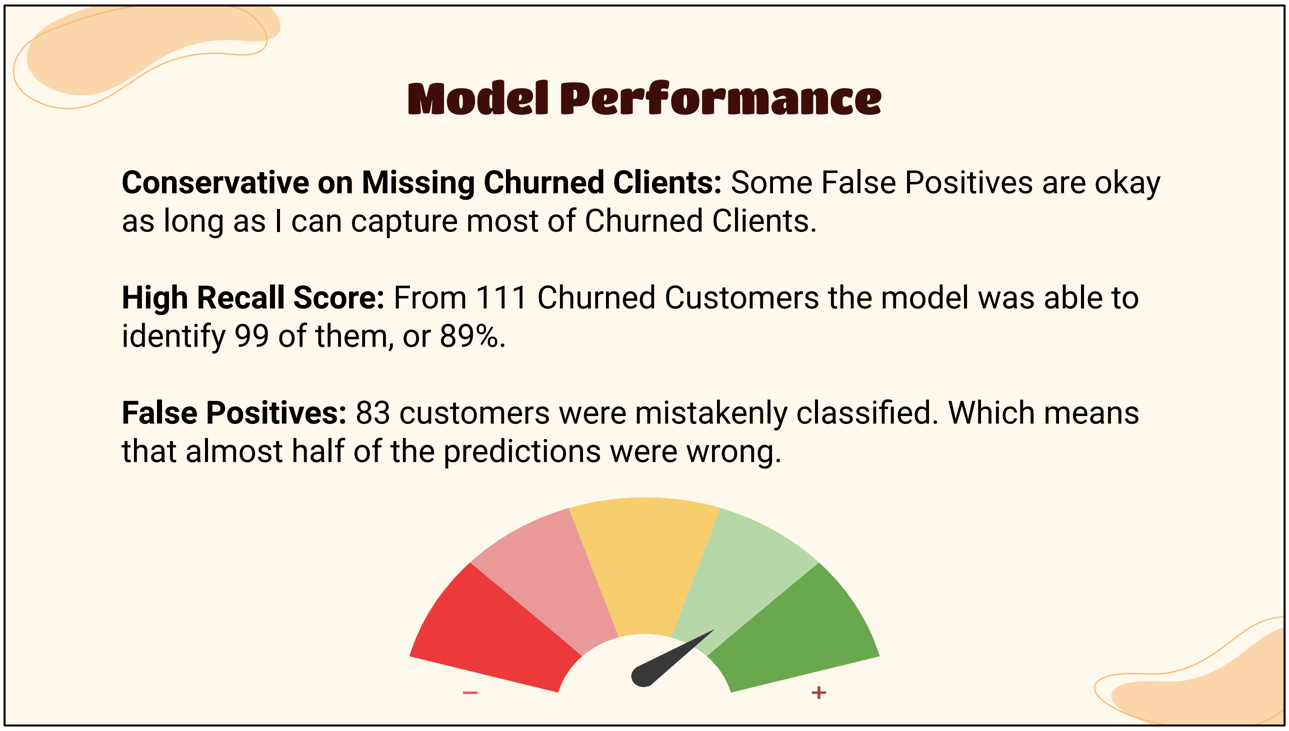
Balance between True and False Positives
+
Maximize Recall Score

To identify the best suitable metric for the given problem, I had to deal with a Trade-off between True and False Positives.

Since I'm building a Model to predict which Clients will churn, it's very important to reduce the number of False Negatives because we don't want to say that a Client is OK when in reality, they are about to churn. That would be a very costly misclassification for the Telecom Company.

At the same time, I can't create a super conservative model with many False Positives because that would cause more work and wasted resources for the Company. I assume that after predicting a churn, someone would try to avoid that by reaching out to the client. The time that employees will waste reaching out to a Healthy Client is money wasted for the Company.

Therefore, while picking the best model, I kept in mind the Trade-Off and aimed to find the best balance between True/False Positives. Moreover, rather than prioritizing Accuracy, I'm focused on maximizing Recall, which measures the capacity of predicting True Positives (churned clients in this case) since it takes True Positives / (True Positives + False Negatives) or the percentage of Churned Clients that the model identified.



Model Performance

Conservative on Missing Churned Clients: Some False Positives are okay as long as I can capture most of Churned Clients.

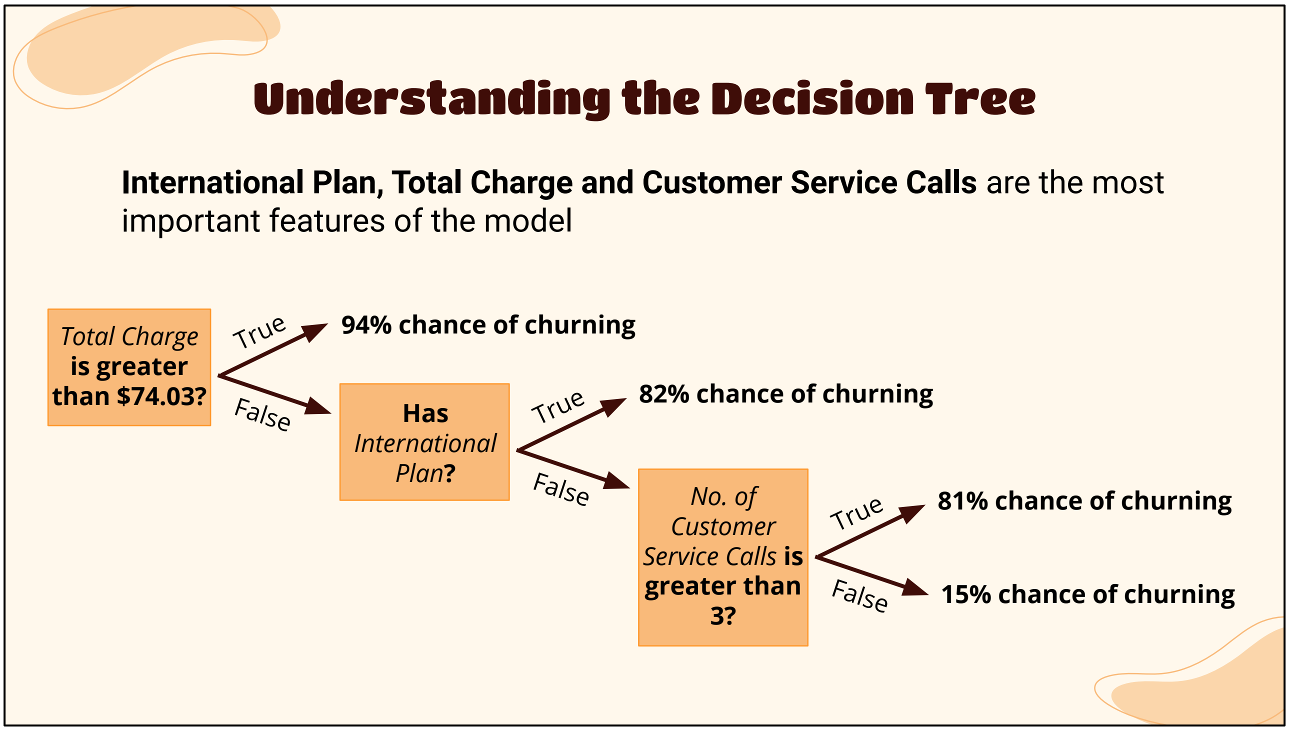
High Recall Score: From 111 Churned Customers the model was able to identify 99 of them, or 89%.

False Positives: 83 customers were mistakenly classified. Which means that almost half of the predictions were wrong.

Given the Trade-Off I talked about and the business problem, I'm conservative about Missing Churned Clients, which means that some False Positives are okay as long as I can capture most Churned Clients.

The Recall Score from the chosen model was 89%, or from 100 Churned Clients, the model was able to identify 89 of them. However, that came at the cost of misclassifying 73 customers.

That gives us roughly a 55% chance of being correct in a prediction, which can seem to be very low. However, that can be adjusted depending on the stakeholder's opinion. It's important to add that by increasing precision, most likely Recall would go down.



Understanding the Decision Tree

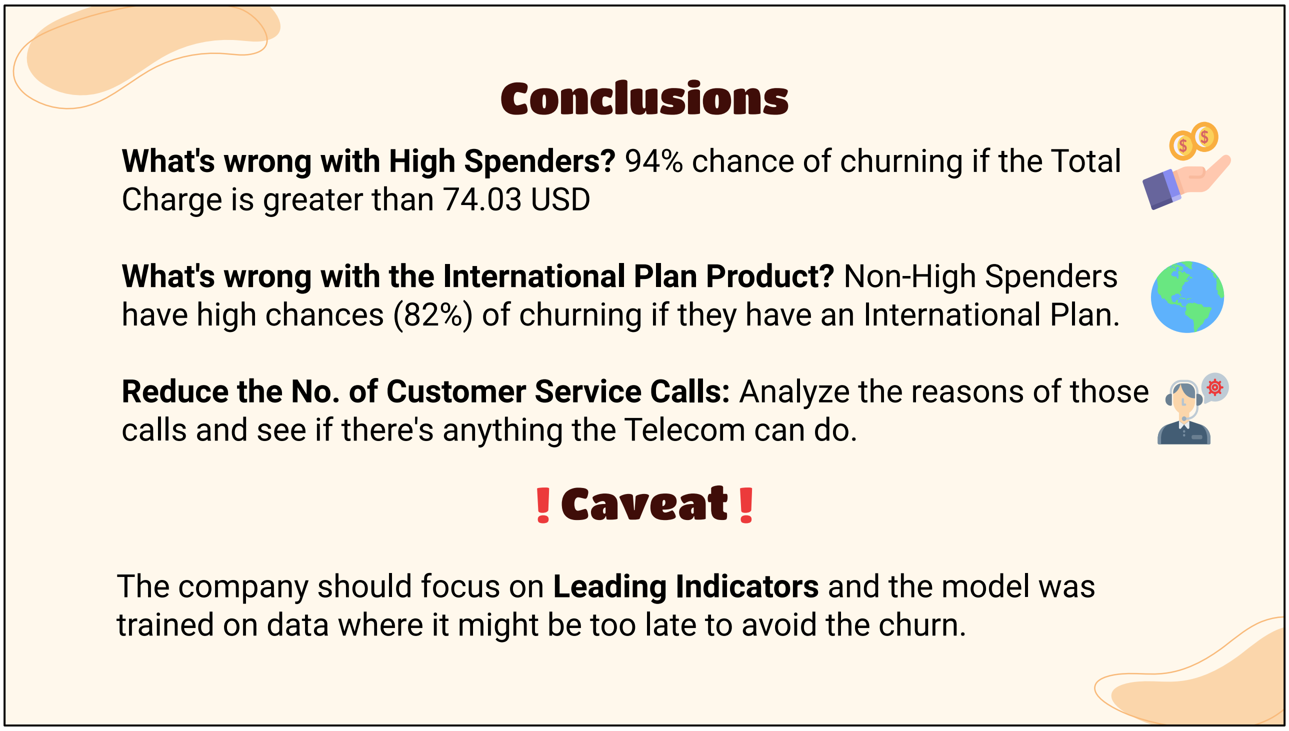
International Plan, Total Charge and Customer Service Calls are the most important features of the model

Total Charge **is greater than $74.03?**
True
False
94% chance of churning
Has *International Plan*?
True
False
82% chance of churning
No. of Customer Service Calls **is greater than 3?**
True
False
81% chance of churning
15% chance of churning

Another important aspect of the model is its interpretability. I believe that understanding why clients are churning is beneficial for the Telecom Company so they can make product/process adjustments.

The main features of the model are **whether the Client has an international plan**, **the Total Charge paid by the Client and the No. of Customer Service Calls.**

The Decision Tree model starts by looking at the **Total Charge Amount**, which gives us a 94% chance of churning if the Amount is greater than $74.03. Then if the Client is not a High Spender, there's an 82% chance of churning if **they have an International Plan.** Finally, if the Client is not a High Spender and doesn't have an International plan, there's an 81% of churning if they have **reached out to Customer Service more than three times.**



Conclusions

What's wrong with High Spenders? 94% chance of churning if the Total Charge is greater than 74.03 USD

What's wrong with the International Plan Product? Non-High Spenders have high chances (82%) of churning if they have an International Plan.

Reduce the No. of Customer Service Calls: Analyze the reasons of those calls and see if there's anything the Telecom can do.

❗Caveat❗

The company should focus on **Leading Indicators** and the model was trained on data where it might be too late to avoid the churn.

The Conclusions are the following:

- There's something wrong with High Spenders. Why there's a 94% chance of churning if the Charge is greater than $74.03?
- What's wrong with the International Plan Product? An investigation into that product would make sense.
- They should try to reduce the No. of Customer Service Calls. It would be interesting to see why customers are reaching out and finding patterns and enhancement opportunities.

Before finishing the presentation, I want to add that a big caveat to all of this is that I trained the model on *after-the-fact* data, which means that when clients hit those thresholds, it might be too late to avoid the churn. It would be more interesting to train the model on data a few weeks before churning or lower the thresholds, so we have time to avoid the churn

Thanks

Do you have any questions?

Reno Neto - renoneto@gmail.com
GitHub: https://github.com/renoneto

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

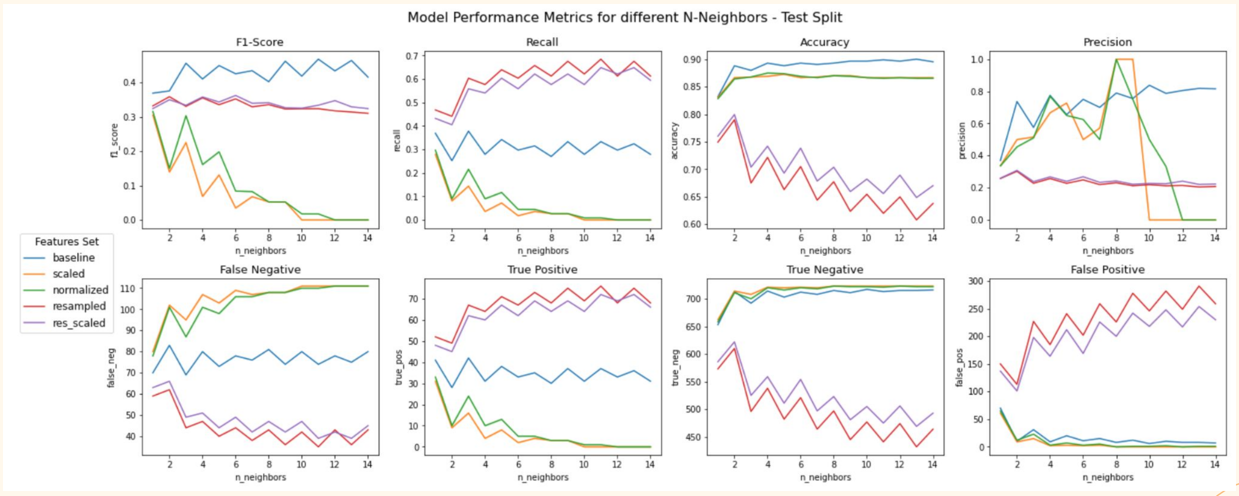
Appendix

All Models were trained on variations of features

They are the following:

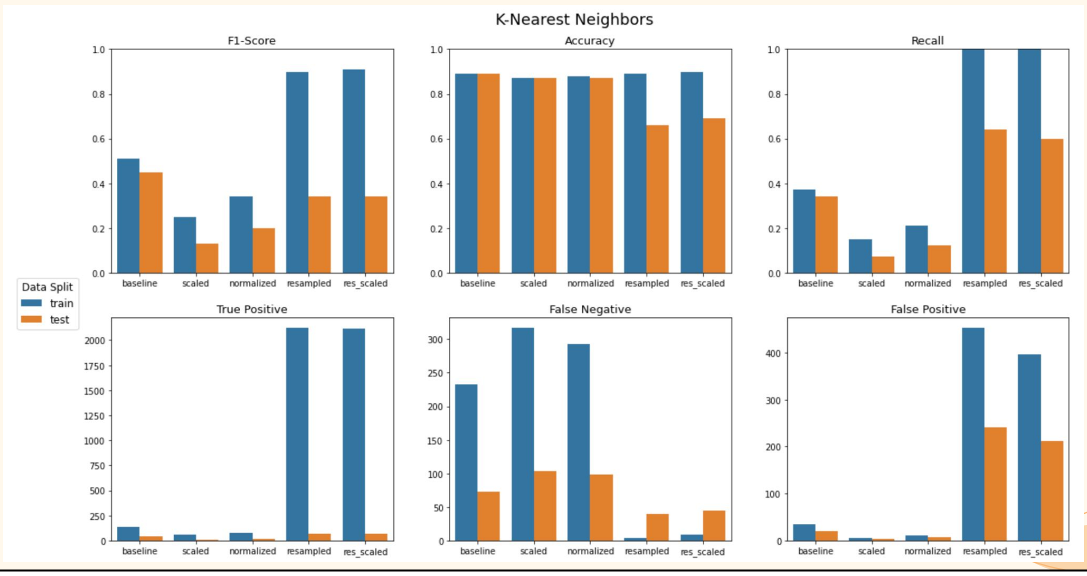
- ***Baseline:*** no transformation.
- ***Scaled*:** Scaled Features (using Standard Scaler from sklearn)
- ***Normalized*:** Scaled + Normalized Features (using Normalizer from sklearn)
- ***Resampled:*** Scaled + Normalized + Resampled (using SMOTE). The decision of resampling features came from the fact that I'm working with an unbalanced dataset, with more regular than churned clients.
- ***Resampled Scaled or res_scaled*:** Scaled + Resampled features. After noticing a decline on performance of certain models when trained on normalized features, I decided to create another variation of Scaled and Resampled features

Choosing the Best N-Neighbors for the K-Nearest Neighbors Classifier



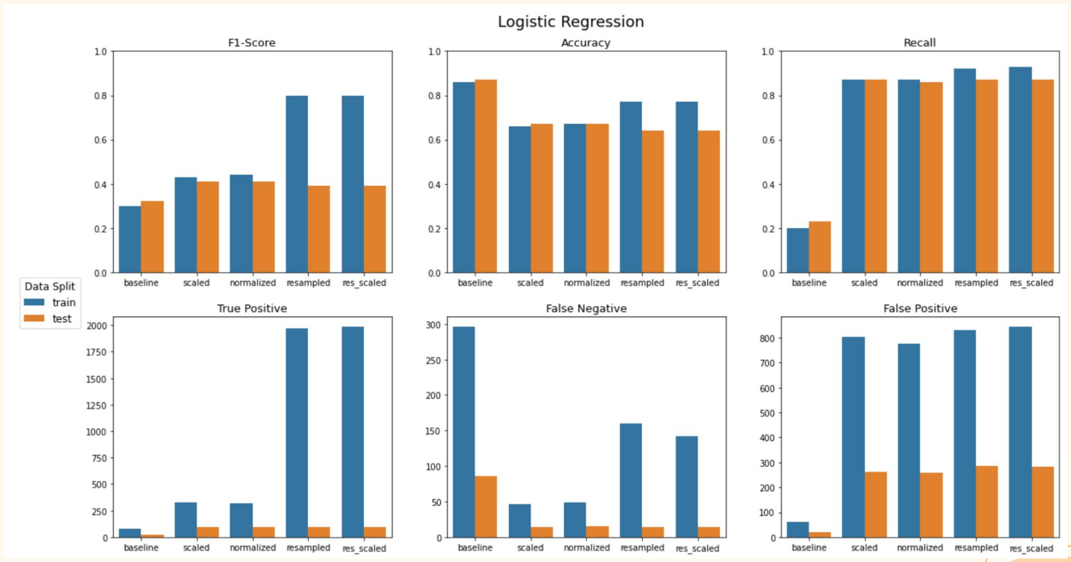
Given the business problem, the goal of the model is to maximize Recall Score keeping in mind a good balance of False Negatives and False Positives. The importance of keeping a healthy balance between two numbers is that I'm trying to avoid unnecessary work for the Company on misclassified clients. At the same time, I don't want to create an extremely accurate model with no False Positives because missing a churned client is more costly than misclassifying a couple of clients. Of course, that decision would have to be made by taking into consideration the costs associated with losing a client vs reaching out to healthy clients that won't churn. Because of that, the chosen No. of Neighbors is five and as you can see Resampling features had a good impact on performance.

Performance Analysis of the K-Neighbors Classifier Model



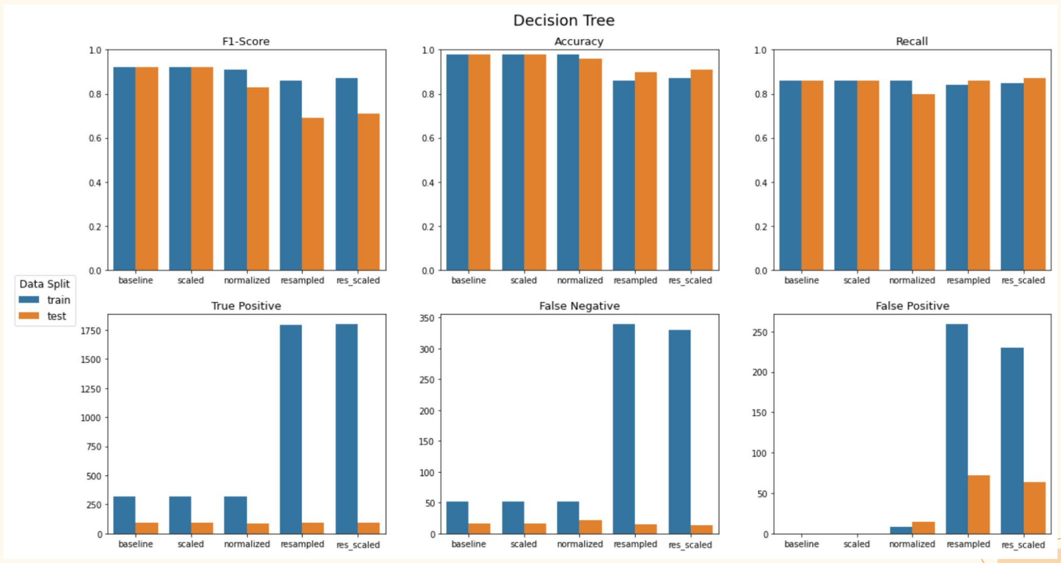
Conclusions:
- It's interesting to see that the `F1-Score` improved drastically after resampling.
- As expected, `Accuracy` dropped after resampling and that's because most of the data is of Non-Churned clients (85%, specifically). Therefore, having a model that classifies everything as 'Not Churned' will have an accuracy score of 0.85. After resampling and making classes 50/50, the model will behave differently and classifying everything as 'Not Churned' won't maximize the score.
- `Recall` improved drastically after resampling, for the same reason as stated before. However, it's still very low (~0.6).

Performance Analysis of the Logistic Regression Model



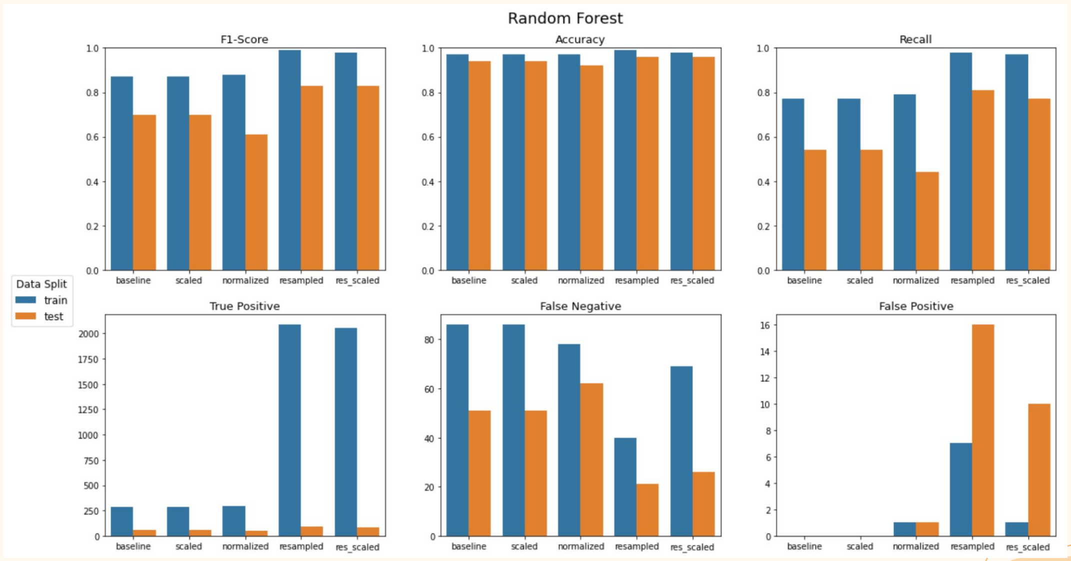
Conclusions:
- `F1-Score`: Dramatic improvement on Train F1-Score after resampling the dataset (`Resampled Models`), however, the same can't be seen when looking at the test set, which makes me think that the model is overfitting the Train set. As matter of fact, the Test F1-Score decreased after resampling.
- `Accuracy`: As expected the metric decreased (relative to `Baseline`) after performing transformations on the dataset. That's due to the unbalanced dataset that we are working with. The `Resampled Models` train set had a higher accuracy score. However, the test scores were similar to the `Scaled/Normalized models`.
- `Recall`: It's possible to see a big improvement on the metric after scaling the data (`Scaled Model`). The score on the test set is pretty much the same for the `Scaled, Normalized and Resampled Models`. The conclusion here is that there are no major improvements after scaling the set. Normalization and Resampling didn't improve the Recall Score on the test set.
- `True Positive / False Negative / False Positive`: It's interesting to see that even though the `Resampled Model` was trained on "new" observations, the No. of False Positive cases are the same as the `Scaled and Normalized Models`. I'd expect to see a similar chart as the True Positive and False Negative, with a bigger blue bar relative to the `Scaled and Normalized Models`.

Performance Analysis of the Decision Tree Model



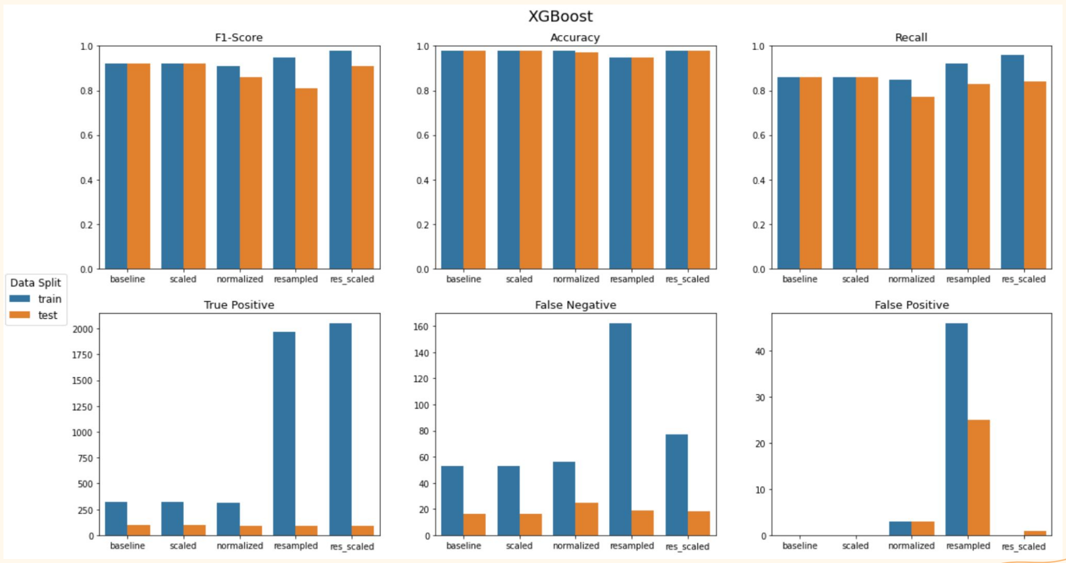
Conclusions:
- Excluding the `Resampled Model`, it's clear that the model is overfitting the training set. The `F1-Score`, `Recall` and `Accuracy` scores are 1 (or very close to it) for the training set. If I decide to move forward with Decision Tree I need to address that problem.
- Focusing on the `Recall Score`, it's interesting to see a drop after normalizing the features and then an increase after resampling them.

Performance Analysis of the Random Forest Model



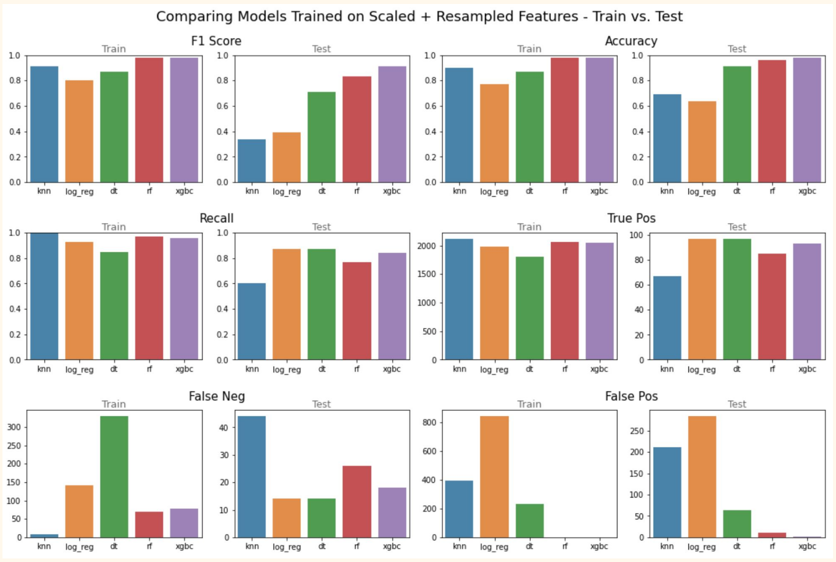
Conclusions:
- Same as before, based on `F1-Score`, `Accuracy` and `Recall`, it's possible to see that the model is most likely overfitting the training data.
- Resampling the features had a positive impact in all three scores.
- Even though it has a lower `Recall` score relative to other models, it has a very low No. of False Positives. That's good because it would mean that the company is not spending resources with clients that didn't have the intention to churn.

Performance Analysis of the XGBoost Model



Conclusions:
- Given the Scores on the Training sets, the model is possibly overfitting.
- `Accuracy Scores` on the Training Set are just too high to be true.
- For the Scaled + Resampled Set, `Recall` is great, almost as good as the Logistic Regression's model (0.84 vs 0.87 on the Logistic Model).
- Comparing it to the Random Forest Model, it's interesting to see that the No. of False Negatives for the training set is high which usually would indicate a similar behavior for the False Positives. However, the No. of False Positives is low compared to other models. This indicates that the model is not overfitting the training set as much as the Random Forest model.
- It looks like normalizing the data had a negative impact in performance.

Comparing Models



Legend:

- **KNN:** K-Nearest Neighbors
- **Log_reg:** Logistic Regression
- **DT:** Decision Tree
- **RF:** Random Forest
- **XGBC:** XGBoost

Conclusions:

- `F-1 and Accuracy Scores`:

- The Training Set's Scores are very high for the Random Forest and XGBoost models which could indicate that the models are overfitting the data. However, by looking at the Test Set's Scores, it's possible to see that both have also high values and are performing better than other models. In an overfitted model I'd expect to see them not performing well on the Test Set.

- `Recall Score`:

- It's clear that the K-Nearest Neighbors model is overfitting the Training Set.
- The Logistic Regression, Decision Tree and XGBoost models have similar performance on the Test Set. With the XGBoost model coming in third place.

- `True Positives`:

- The Logistic Regression, Decision Tree and XGBoost models have similar performance on the Test Set. With the XGBoost model coming in third place, again.

- `False Negatives`:

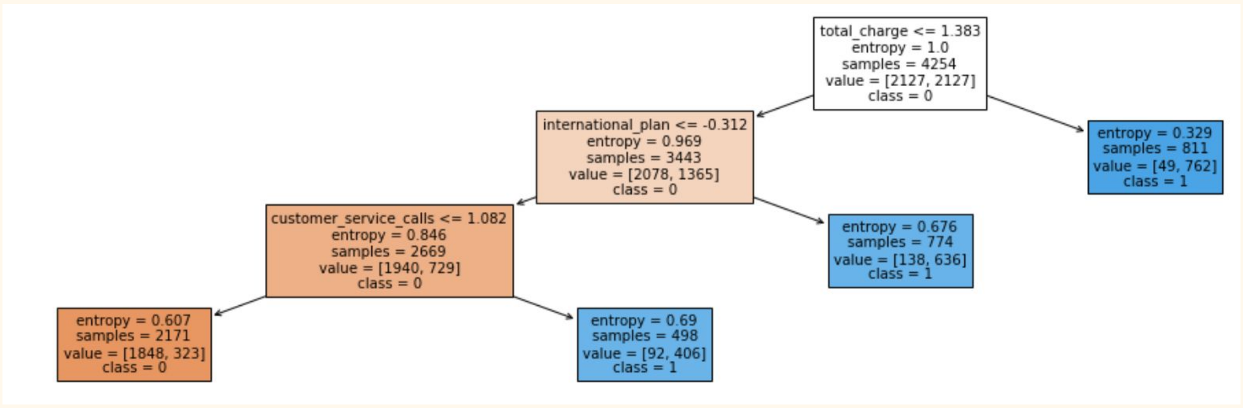
- The Decision Tree model is underperforming and missing more churned clients than others on the Training Set. However, surprisingly the model is performing on par

with the Logistic Regression and XGBoost models on the Test Set.
    - Here's another data point showing that the K-Nearest Neighbors is overfitting the Training Set with almost no False Negatives on the Training Set and more than others on the Test Set.
    - The Logistic Regression and Decision Tree models are per**forming better than others with XGBoost coming in third place.**
- `False Positives`:
    - Here we can see that the XGBoost Model is performing better than the Decision Tree and Logistic Regression ones. The three models have similar Recall Scores, but the Logistic Regression model is creating a lot more False Positives and the Decision Tree model is generating more than the XGBoost model. This is important because of all the effort that the Telecom Company will probably put to avoid these "about to churn" clients from churning. It's going to be way more expensive for the company to reduce Churn Rate if they decide to use the Logistic Regression model.

Choosing a model
I'm between the Decision Tree and XGBoost models.
- `Recall Score`: Even though it's not performing as well as XGBoost on the Training Set, the **Decision Tree is the winner on the Testing Set.**
- `False Negative/True Positive`: Even though it's not performing as well as XGBoost on the Training Set, the **Decision Tree is the winner on the Testing Set.**
- `False Positive`: Even though the XGBoost is performing better than the Decision Tree, I'm afraid that the model is not conservative. Not having any False Positives is a bit concerning. I rather have some and be conservative than missing a client who is about to churn.

Therefore, the Decision Tree model is the winner.

Understanding the Decision Tree Model I



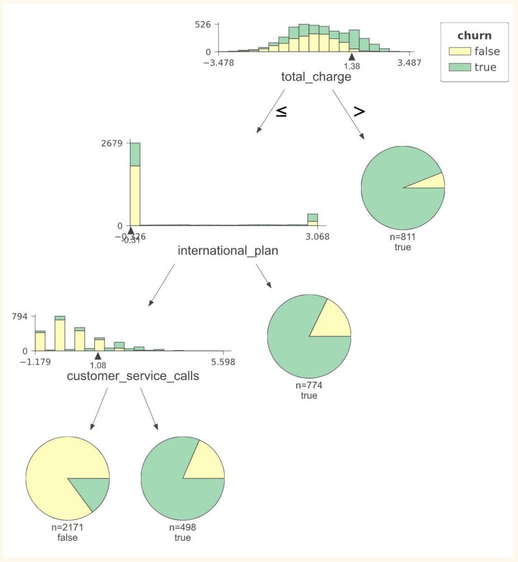
Interpretation - Walk-through the Classification of a Record

- If `total_charge`:
- **> 74.03 USD:** the chance of churning is 94% (762/811).
- **<= 74.03 USD:** the chance of churning is 39% (1365/3443).

- If `total_charge <= 74.03` + `international_plan`:
- **Is True:** the chance of churning is 82% (636/774).
- **Is False:** the chance of churning is 27% (729/2669).

- If `total_charge <= 74.03` + `international_plan == False` + No. of `customer_service_calls`:
- **<= 3:** the chance of churning is 15% (323/2171).
- **> 3:** the chance of churning is 81% (406/498).

Understanding the Decision Tree Model II



Same conclusions, different visual