Limiting Churn

This report provides an examination of our methods to find the factors that cause customer churn, and how we predicted whether or not a customer will churn using a Logistic Regression Classifier.

Machine Minds
Timothy Chong
Ethan Howat
Jessica Nguyen
Zach Soo
I310D (Spring 2023)

Introduction and Background

Churn is a phenomenon that occurs when customers stop paying for a product or service that a business offers. As a result, businesses would like to seek solutions that limit the “churning” of customers. The problem we want to solve is how to reduce customer churn. Our audience is Telco, a hypothetical telecommunications company, however, this project could benefit any similar company that wants to reduce the rate of customer churn. The purpose of this project is to find a solution or method to reduce customer churn. To achieve this, we need to identify the biggest factors that cause churn and choose a model that can accurately predict if customers will churn or not.

Data Description

The dataset that we used for this project was the Telco Customer Churn dataset. IBM created this dataset of a fictional, hypothetical telecommunication company’s customer data. The dataset has 7043 observations with 33 features.

	CustomerID	Count	Country	State	City	Zip Code	Lat Long	Latitude	Longitude	Gender	...	Contract	Paperless Billing	Payment Method	Monthly Charges	Total Charges	Churn Label	Churn Value	Churn Score	CLTV	Churn Reason
0	3668-QPYBK	1	United States	California	Los Angeles	90003	33.964131, -118.272783	33.964131	-118.272783	Male	...	Month-to-month	Yes	Mailed check	53.85	108.15	Yes	1	86	3239	Competitor made better offer
1	9237-HQITU	1	United States	California	Los Angeles	90005	34.059281, -118.30742	34.059281	-118.307420	Female	...	Month-to-month	Yes	Electronic check	70.70	151.65	Yes	1	67	2701	Moved
2	9305-CDSKC	1	United States	California	Los Angeles	90006	34.048013, -118.293953	34.048013	-118.293953	Female	...	Month-to-month	Yes	Electronic check	99.65	820.5	Yes	1	86	5372	Moved
3	7892-POOKP	1	United States	California	Los Angeles	90010	34.062125, -118.315709	34.062125	-118.315709	Female	...	Month-to-month	Yes	Electronic check	104.80	3046.05	Yes	1	84	5003	Moved
4	0280-XJGEX	1	United States	California	Los Angeles	90015	34.039224, -118.266293	34.039224	-118.266293	Male	...	Month-to-month	Yes	Bank transfer (automatic)	103.70	5036.3	Yes	1	89	5340	Competitor had better devices

5 rows × 33 columns

Before preprocessing, we wanted to understand the columns and the types of values in the data (code output below).

```
CustomerID Unique values: ['3668-QPYBK' '9237-HQITU' '9305-CDSKC' ... '2234-XADUH' '4801-JZAZL'
 '3186-AJIEK']
Number of unique values: 7043

Count Unique values: [1]
Number of unique values: 1

Country Unique values: ['United States']
Number of unique values: 1

State Unique values: ['California']
Number of unique values: 1

City Unique values: ['Los Angeles' 'Beverly Hills' 'Huntington Park' ... 'Standish' 'Tulelake'
 'Olympic Valley']
Number of unique values: 1129
```

Preprocessing

After understanding the columns and their values, we dropped these columns, and gave our reasonings:

- *CustomerID*: this column was just an identifier for the company
- *Count*: every observation (row) had a value of 1 for this column
- *Country*, *State*, *Zip Code*, *Lat Long*, *Latitude*, and *Longitude*: these columns contained sensitive customer data
- *Churn Label*, *Churn Score*, and *Churn Reason*: these columns were dropped to focus on the *Churn Value* label for our ML model

	Gender	Senior Citizen	Partner	Dependents	Tenure Months	Phone Service	Multiple Lines	Internet Service	Online Security	Online Backup	Device Protection	Tech Support	Streaming TV	Streaming Movies	Contract	Paperless Billing	Payment Method	Monthly Charges	Total Charges	Churn Value
0	Male	No	No	No	2	Yes	No	DSL	Yes	Yes	No	No	No	No	Month-to-month	Yes	Mailed check	53.85	108.15	1
1	Female	No	No	Yes	2	Yes	No	Fiber optic	No	No	No	No	No	No	Month-to-month	Yes	Electronic check	70.70	151.65	1
2	Female	No	No	Yes	8	Yes	Yes	Fiber optic	No	No	Yes	No	Yes	Yes	Month-to-month	Yes	Electronic check	99.65	820.5	1
3	Female	No	Yes	Yes	28	Yes	Yes	Fiber optic	No	No	Yes	Yes	Yes	Yes	Month-to-month	Yes	Electronic check	104.80	3046.05	1
4	Male	No	No	Yes	49	Yes	Yes	Fiber optic	No	Yes	Yes	No	Yes	Yes	Month-to-month	Yes	Bank transfer (automatic)	103.70	5036.3	1

After dropping those columns, we checked for duplicates, dropped the duplicates, and then double-checked.

```
Number of duplicates before : 22
Number of duplicates after removing : 0
```

The dataset now has 7021 observations with 20 features. At this point, we realize that we want our data for all of our qualitative values to be quantitative. Using a user-defined function called transform, which has one parameter (a data frame), we convert the qualitative values into quantitative ones in a process similar to one-hot encoding.

```
groups = {
    'Mailed check':1, 'Electronic check': 2, 'Bank transfer (automatic)': 3, 'Credit card (automatic)': 4
}
X['Payment Method'] = [groups.get(x) for x in df['Payment Method']]

groups = {
    'No': 0, 'DSL':1, 'Fiber optic': 2, 'Cable': 3
}
X['Internet Service'] = [groups.get(x) for x in df['Internet Service']]
```

Additionally, we drop our null values and check our data again.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7021 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Gender             7021 non-null   object
 1   Senior Citizen     7021 non-null   object
 2   Partner            7021 non-null   object
 3   Dependents         7021 non-null   object
 4   Tenure Months      7021 non-null   int64
 5   Phone Service      7021 non-null   object
 6   Multiple Lines     7021 non-null   object
 7   Internet Service   7021 non-null   object
 8   Online Security    7021 non-null   object
 9   Online Backup      7021 non-null   object
 10  Device Protection  7021 non-null   object
 11  Tech Support       7021 non-null   object
 12  Streaming TV       7021 non-null   object
 13  Streaming Movies   7021 non-null   object
 14  Contract           7021 non-null   object
 15  Paperless Billing  7021 non-null   object
 16  Payment Method     7021 non-null   object
 17  Monthly Charges    7021 non-null   float64
 18  Total Charges      7021 non-null   object
 19  Churn Value        7021 non-null   int64
dtypes: float64(1), int64(2), object(17)
memory usage: 1.1+ MB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4827 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Tenure Months      4827 non-null   int64
 1   Monthly Charges    4827 non-null   float64
 2   Total Charges      4827 non-null   object
 3   Churn Value        4827 non-null   int64
 4   Gender             4827 non-null   int64
 5   Senior Citizen     4827 non-null   int64
 6   Partner            4827 non-null   int64
 7   Dependents         4827 non-null   int64
 8   Phone Service      4827 non-null   int64
 9   Multiple Lines     4827 non-null   float64
 10  Online Security    4827 non-null   float64
 11  Online Backup      4827 non-null   float64
 12  Device Protection  4827 non-null   float64
 13  Tech Support       4827 non-null   float64
 14  Streaming TV       4827 non-null   float64
 15  Streaming Movies   4827 non-null   float64
 16  Contract           4827 non-null   int64
 17  Paperless Billing  4827 non-null   int64
 18  Payment Method     4827 non-null   int64
 19  Internet Service   4827 non-null   int64
dtypes: float64(8), int64(11), object(1)
memory usage: 791.9+ KB
```

The transform function converted our qualitative variables into floats, but we wanted them to be integers for styling reasons. We also noticed that Monthly Charges was quantitative but its value was an object, so we converted its values to floats.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4827 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Tenure Months      4827 non-null   int64
 1   Monthly Charges    4827 non-null   float64
 2   Total Charges      4824 non-null   float64
 3   Churn Value        4827 non-null   int64
 4   Gender             4827 non-null   int64
 5   Senior Citizen     4827 non-null   int64
 6   Partner            4827 non-null   int64
 7   Dependents         4827 non-null   int64
 8   Phone Service      4827 non-null   int64
 9   Multiple Lines     4827 non-null   int64
 10  Online Security    4827 non-null   int64
 11  Online Backup      4827 non-null   int64
 12  Device Protection  4827 non-null   int64
 13  Tech Support       4827 non-null   int64
 14  Streaming TV       4827 non-null   int64
 15  Streaming Movies   4827 non-null   int64
 16  Contract           4827 non-null   int64
 17  Paperless Billing  4827 non-null   int64
 18  Payment Method     4827 non-null   int64
 19  Internet Service   4827 non-null   int64
dtypes: float64(2), int64(18)
memory usage: 791.9 KB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4824 entries, 0 to 7042
Data columns (total 20 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Tenure Months      4824 non-null   int64
 1   Monthly Charges    4824 non-null   float64
 2   Total Charges      4824 non-null   float64
 3   Churn Value        4824 non-null   int64
 4   Gender             4824 non-null   int64
 5   Senior Citizen     4824 non-null   int64
 6   Partner            4824 non-null   int64
 7   Dependents         4824 non-null   int64
 8   Phone Service      4824 non-null   int64
 9   Multiple Lines     4824 non-null   int64
 10  Online Security    4824 non-null   int64
 11  Online Backup      4824 non-null   int64
 12  Device Protection  4824 non-null   int64
 13  Tech Support       4824 non-null   int64
 14  Streaming TV       4824 non-null   int64
 15  Streaming Movies   4824 non-null   int64
 16  Contract           4824 non-null   int64
 17  Paperless Billing  4824 non-null   int64
 18  Payment Method     4824 non-null   int64
 19  Internet Service   4824 non-null   int64
dtypes: float64(2), int64(18)
memory usage: 791.4 KB
```

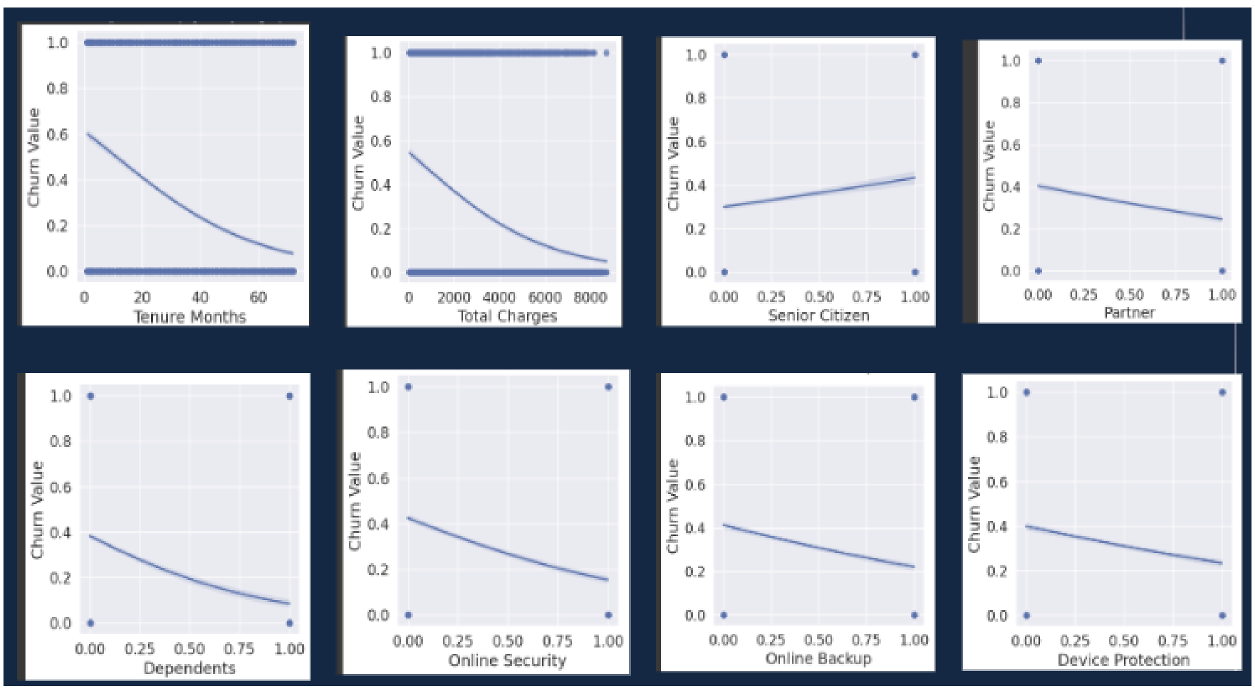
All of the features in the dataset are quantitative now. To make sure we got rid of any duplicates and null values, we checked for null and duplicate values again. At the end of preprocessing, the dataset now has 4824 observations with 20 features.

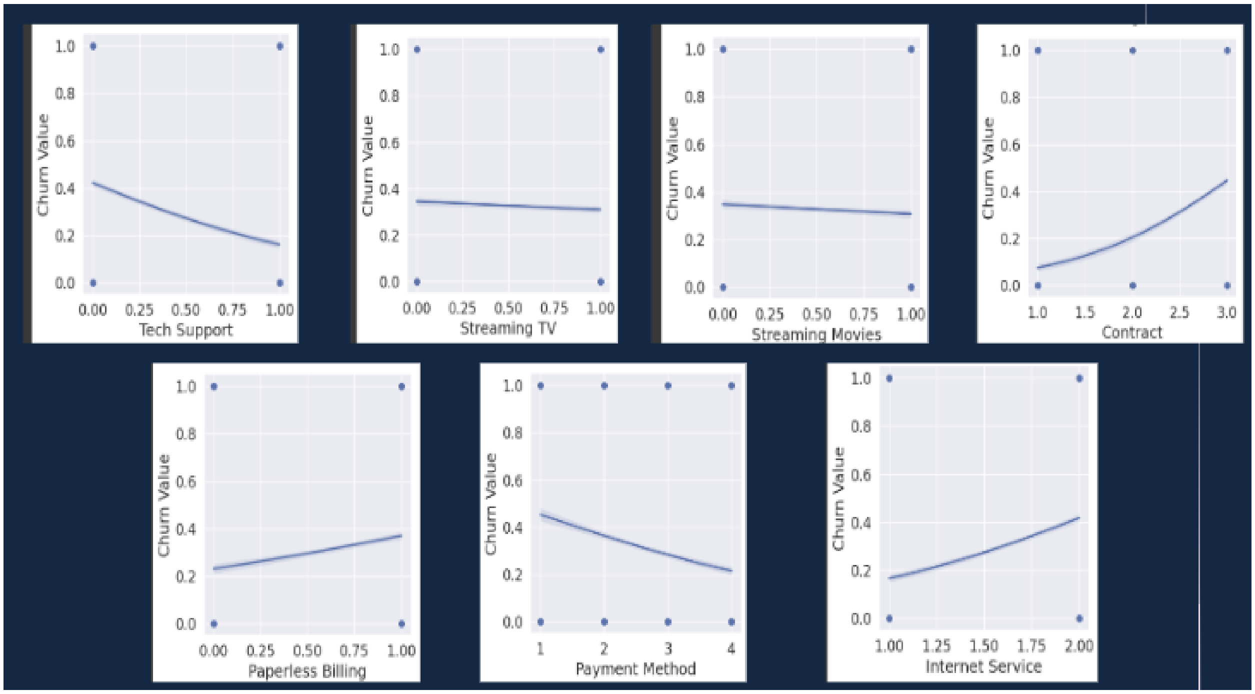
Methodology

With our cleaned and transformed data, the approach we took to analyze the data is to first perform feature analysis to pick which features matter the most in our model, and then use the features and labels from our data to implement them into our Logistic Regression Classifier ML model. For feature analysis, we utilized bivariate analysis; we specifically conducted Linear Regression significance tests that see if there is a significant association between the feature and the label (whether or not a customer churns). If the p-value is above 0.05, then we drop the feature from the dataset. A caveat for doing only the Linear Regression significance test is that the p-value was the only metric we were looking at to drop features.

```
Association between Tenure Months and Churn Value(p-value): 1.3410653913426676e-201
Association between Total Charges and Churn Value(p-value): 3.8813637546260397e-146
Association between Senior Citizen and Churn Value(p-value): 7.321210768988156e-14
Association between Partner and Churn Value(p-value): 2.5419991729222584e-32
Association between Dependents and Churn Value(p-value): 3.161290960028799e-102
/usr/local/lib/python3.10/dist-packages/scipy/stats/_stats_py.py:4424: ConstantInputWarning: An input a
  warnings.warn(stats.ConstantInputWarning(msg))
<ipython-input-14-34f83e826665>:14: RuntimeWarning: Precision loss occurred in moment calculation due t
  t_stat, p_value = ttest_ind(churned, not_churned, equal_var=False)
Association between Online Security and Churn Value(p-value): 5.683657031290356e-102
Association between Online Backup and Churn Value(p-value): 4.4955658740716785e-48
Association between Device Protection and Churn Value(p-value): 1.0941603664680183e-35
Association between Tech Support and Churn Value(p-value): 1.340796762361215e-93
Association between Streaming TV and Churn Value(p-value): 0.010632086966700538
Association between Streaming Movies and Churn Value(p-value): 0.002203553241381093
Association between Contract and Churn Value(p-value): 2.400090705069821e-149
Association between Paperless Billing and Churn Value(p-value): 9.243486258482429e-24
Association between Payment Method and Churn Value(p-value): 3.6079355904664377e-35
Association between Internet Service and Churn Value(p-value): 1.7594482896343278e-86

Excluded features (p-value > 0.05): ['Monthly Charges', 'Gender', 'Phone Service', 'Multiple Lines']
```

We then created visual representations of the remaining features in the dataset, which have some association with churn value.





Some features had a positive association:

- *Senior Citizen*: whether the person is a senior or not (0 if not, 1 if so)
- *Contract*: what type of contract a customer is on (1 represents one year, 2 represents two-year, 3 represents month-to-month)
- *Paperless Billing*: whether or not someone is on paperless billing (0 if not, 1 if so)
- *Internet Service*: whether or not someone has internet service (0 if not, if they do: 1 for DSL, 2, for fiber optic, 3 for cable)

All of the other features had a negative association:

- *Tenure Months*: the number of months that the customer has been with the company by the quarter end
- *Total Charges*: customer's total charges at quarter's end in USD
- *Partner*: whether the person has a partner or not (0 if not, 1 if so)
- *Dependents*: whether the person has dependents or not (0 if not, 1 if so)
- *Online Security*: whether the person subscribes to an online security service (0 if not, 1 if so)
- *Online Backup*: whether the person subscribes to an online backup service (0 if not, 1 if so)
- *Device Protection*: whether the person subscribes to a device protection service (0 if not, 1 if so)
- *Tech Support*: whether the person subscribes to a tech support service (0 if not, 1 if so)
- *Streaming TV*: whether the person streams television (0 if not, 1 if so)
- *Streaming Movies*: whether the person streams movies (0 if not, 1 if so)
- *Payment Method*: how the customer pays: 1 for a mailed check, 2 for an electronic check, 3 for a bank transfer, 4 for a credit card

We then split our data into a training set (80%) and a testing set (20%) and used both sets to create our Logistic Classifier Machine Learning Model.

```
[ ] X_train.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 3859 entries, 4490 to 997
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Tenure Months      3859 non-null   int64
 1   Total Charges      3859 non-null   float64
 2   Senior Citizen     3859 non-null   int64
 3   Partner            3859 non-null   int64
 4   Dependents         3859 non-null   int64
 5   Online Security    3859 non-null   int64
 6   Online Backup      3859 non-null   int64
 7   Device Protection  3859 non-null   int64
 8   Tech Support       3859 non-null   int64
 9   Streaming TV       3859 non-null   int64
 10  Streaming Movies   3859 non-null   int64
 11  Contract           3859 non-null   int64
 12  Paperless Billing  3859 non-null   int64
 13  Payment Method     3859 non-null   int64
 14  Internet Service   3859 non-null   int64
dtypes: float64(1), int64(14)
memory usage: 482.4 KB
```

```
[643] y_train.info()

<class 'pandas.core.series.Series'>
Int64Index: 3859 entries, 4490 to 997
Series name: Churn Value
Non-Null Count  Dtype
--------------  -----
3859 non-null   int64
dtypes: int64(1)
memory usage: 60.3 KB
```

```
[644] y_test.info()

<class 'pandas.core.series.Series'>
Int64Index: 965 entries, 4483 to 5433
Series name: Churn Value
Non-Null Count  Dtype
--------------  -----
965 non-null    int64
dtypes: int64(1)
memory usage: 15.1 KB
```

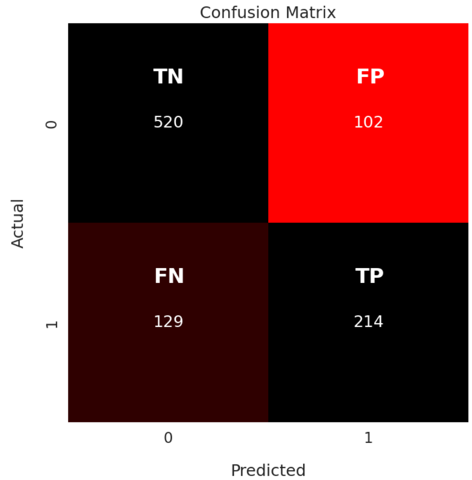
```
[642] X_test.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 965 entries, 4483 to 5433
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Tenure Months      965 non-null    int64
 1   Total Charges      965 non-null    float64
 2   Senior Citizen     965 non-null    int64
 3   Partner            965 non-null    int64
 4   Dependents         965 non-null    int64
 5   Online Security    965 non-null    int64
 6   Online Backup      965 non-null    int64
 7   Device Protection  965 non-null    int64
 8   Tech Support       965 non-null    int64
 9   Streaming TV       965 non-null    int64
 10  Streaming Movies   965 non-null    int64
 11  Contract           965 non-null    int64
 12  Paperless Billing  965 non-null    int64
 13  Payment Method     965 non-null    int64
 14  Internet Service   965 non-null    int64
dtypes: float64(1), int64(14)
memory usage: 120.6 KB
```

Results

Our logistic classifier's accuracy is 76% is not the best; however, because accuracy does not consider how the data is distributed, we also checked the precision: "correct positive predictions relative to total positive predictions", recall: "correct positive predictions relative to total actual positives", and F1 score: the "harmonic mean of precision and recall" (Bobbitt, 2021). Our logistic classifier's precision is 62%, meaning when the model predicts that a customer will churn, it is accurate 62% of the time. The recall for our classifier was 68%, meaning that the classifier correctly identifies 68% of the people who churned. The F1 score is .65, which indicates that the model's performance is okay.

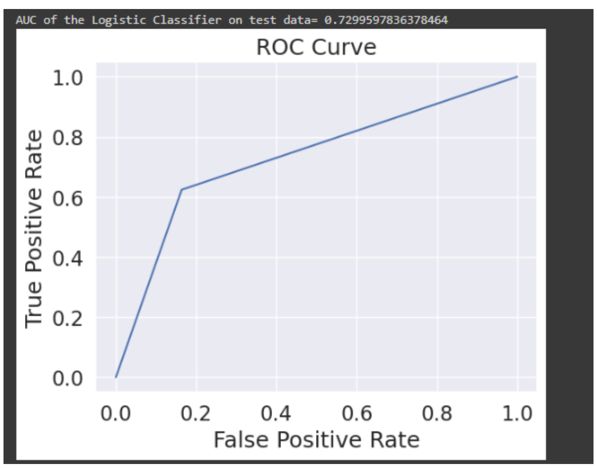
```
...Training successfully accomplished...
Accuracy of the Logistic Classifier on test data= 0.7606217616580311
Precision of the Logistic Classifier on test data= 0.6239067055393586
Recall of the Logistic Classifier on test data= 0.6772151898734177
F1 Score of the Logistic Classifier on test data= 0.6494688922610016
```

We then created a confusion matrix, a performance measurement visualized as "a table with 4 different combinations of predicted and actual values" (Narkhede, 2021). The confusion matrix helps us to understand the number of true negatives and true positives, as well as false negatives and false positives that our model produces.



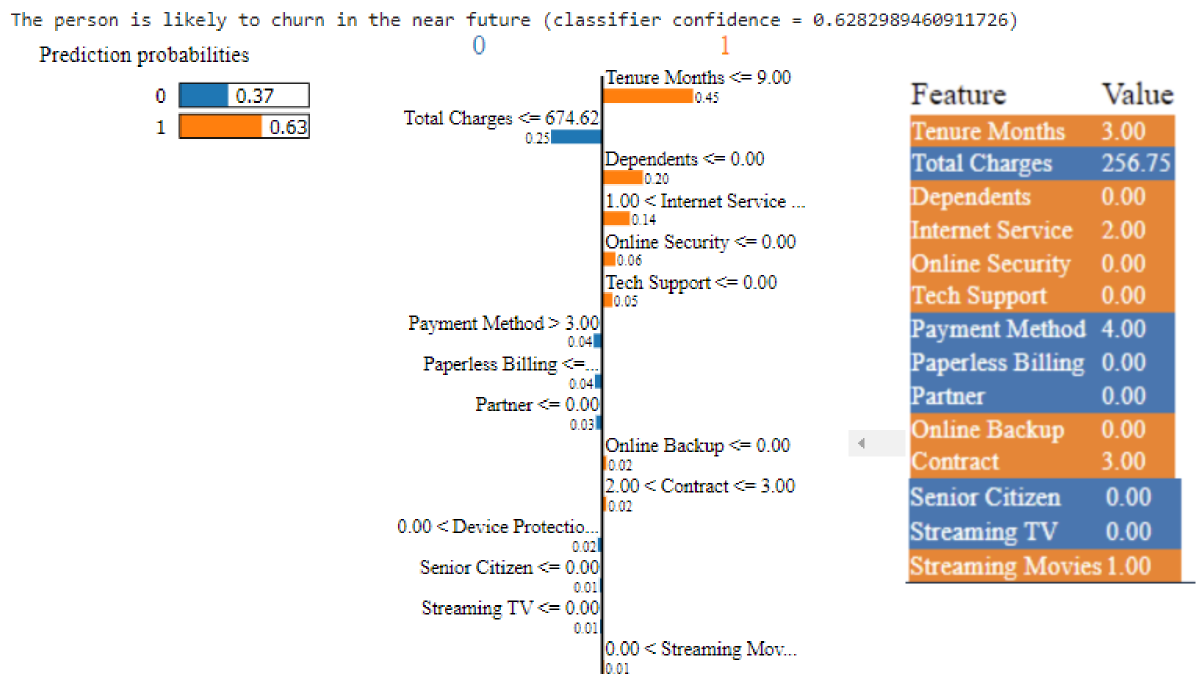
From the confusion matrix, we can see that the model predicted that 520 customers would not churn, and they did not (true negative). 102 customers were predicted to churn, but they did not (false positive). 129 customers were predicted to not churn, but they did (false negative). 214 customers were predicted to churn, and they did (true positive). There were more false negatives and positives than we expected.

To visually understand how our model works, we also utilized a ROC curve. We got the area under the curve (AUC) which was 0.7299597836378464, or about .73.



Because our AUC is between .5 and 1, "there is a high chance that the classifier will be able to distinguish the positive class values from the negative ones" because the model detects more true positives/negatives than false ones (Bhandari, 2023).

To gain more insight into our model's predictions, we used LIME on a random data point in our test set.



From the visualization, we can see that this customer is likely to churn, and the classifier has a confidence of about 63%. LIME shows us that the model predicted that this customer will churn based on the features in orange. Most of the features negatively associated with churn led to the classifier predicting that the customer will churn in this example, with *Partner*, *Payment Method*, and *Streaming TV* being the exceptions.

The results are significant because we know which services need improvement and which customers to target. Additionally, the model predicts the correct label (whether the customer will churn or not) most of the time; this means that we can get a good estimate of whether or not the customer will churn.

Conclusion

Our research questions were related to finding the biggest factors that cause churn, and if we could predict when the customer could churn. We answered our research questions by using bivariate analysis to understand and utilize the features associated with churn, creating a logistic classification machine learning model to predict customer churn, and evaluating our model by using various techniques.

Our project has a few limitations:

- One limitation of our model was that the F1 score was only .65, meaning that the classifier's performance is just OK.
- Utilizing only one machine learning classifier might have been a limitation because we do not have another classifier to compare ours to. Using another classification model such as a multi-layer perceptron could help us get better results.
- We did not test for biases, potentially lowering our accuracy and F1 score.

- The original Telco dataset was unbalanced because more people did not churn than those who did.

Overall, our model can be applied to real-world business contexts. Using our findings from feature analysis, we know who to target and which services need to be improved or removed. Additionally, we can utilize this model to predict whether someone will churn or not. In the future, we hope to create more machine-learning models (MLP, SVM, etc.) to see if other classifiers work better than our current one. We also plan to oversample the positive class (people who churn) for training to compensate for the unbalanced Telco dataset, which will help to create a more balanced data set and improve the results our model produces.

References

Bhandari, A. (2023, April 13). *Guide to AUC ROC curve in machine learning : What is specificity?* Analytics Vidhya. Retrieved April 30, 2023, from https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/#What_is_the_AUC-ROC_Curve

Bobbitt, Z. (2021, September 8). *F1 score vs. accuracy: Which should you use?* Statology. Retrieved April 30, 2023, from https://www.statology.org/f1-score-vs-accuracy/

Narkhede, S. (2021, June 15). *Understanding confusion matrix*. Medium. Retrieved April 30, 2023, from https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

References Related to Code / Data

GitHub Repository (Instructions to run code provided in the README):
https://github.com/ZacharySoo01/I310D_FinalProject
Telco Customer Churn IBM Dataset:
https://www.kaggle.com/datasets/yeanzc/telco-customer-churn-ibm-dataset
Test/Train Data: https://data.world/timmchong/churn-training-and-testing-data

Documentation For Libraries Used:

- Pandas: https://pandas.pydata.org/docs/index.html
- PyPi: https://pypi.org/project/requests/
- NumPy: https://numpy.org/
- Seaborn: https://seaborn.pydata.org/
- SciPy: https://docs.scipy.org/doc/scipy/index.html
- Scikit-learn: https://scikit-learn.org/stable/index.html
- Matplotlib: https://matplotlib.org/stable/index.html
- LIME: https://github.com/marcotcr/lime