Digital Egypt Pioneers Initiative (Round 3)

Data Science Track

# Customer Churn Prediction & Analysis

## (Telecom Industry Case Study)


Group Name: Cognitix

Toqa Mohamed
Salma Mohamed

Marian Milad
Khalid Salim

Yousef Salah
Mahmoud Seif

# Table of Contents

# 1. Introduction

This project is about building a machine learning model to predict customer churn in a telecom firm. The project is built on 4 main phases:

1. The first phase of the project utilizes data analysis techniques such as data cleaning and EDA to understand the data and deliver insights about any hidden patterns within data.
2. Then based on these insights we use feature transformation and feature engineering to transform the data in a form that would be most appropriate for machine learning.
3. The third phase is the machine learning model development, where we train and test different machine learning models (classifiers) with the obtained - cleaned and feature engineered - data, namely:
    - Logistic Regression
    - Random Forest
    - Gradient Boosting
    - XGBoosting
    - Support Vector Machines
4. Based on the results of the classifiers, we choose the one with the best results for deployment.

The data was downloaded from the following link:

https://www.kaggle.com/code/zakriasaad1/customer-churn-prediction-on-telecom-dataset/data

# 2. Data Preprocessing & EDA

## 2.1. Data Inspection

The raw data has 38 columns (15 numerical and 23 categorical) and 7043 rows that . The columns' description is in Appendix (A).

### 2.1.1. Data `info()` and `nunique()` values

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 38 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Customer ID                        7043 non-null   object
 1   Gender                             7043 non-null   object
 2   Age                                7043 non-null   int64
 3   Married                            7043 non-null   object
 4   Number of Dependents               7043 non-null   int64
 5   City                               7043 non-null   object
 6   Zip Code                           7043 non-null   int64
 7   Latitude                           7043 non-null   float64
 8   Longitude                          7043 non-null   float64
 9   Number of Referrals                7043 non-null   int64
 10  Tenure in Months                   7043 non-null   int64
 11  Offer                              3166 non-null   object
 12  Phone Service                      7043 non-null   object
 13  Avg Monthly Long Distance Charges  6361 non-null   float64
 14  Multiple Lines                     6361 non-null   object
 15  Internet Service                   7043 non-null   object
 16  Internet Type                      5517 non-null   object
 17  Avg Monthly GB Download            5517 non-null   float64
 18  Online Security                    5517 non-null   object
 19  Online Backup                      5517 non-null   object
 20  Device Protection Plan             5517 non-null   object
 21  Premium Tech Support               5517 non-null   object
 22  Streaming TV                       5517 non-null   object
 23  Streaming Movies                   5517 non-null   object
 24  Streaming Music                    5517 non-null   object
 25  Unlimited Data                     5517 non-null   object
 26  Contract                           7043 non-null   object
 27  Paperless Billing                  7043 non-null   object
 28  Payment Method                     7043 non-null   object
 29  Monthly Charge                     7043 non-null   float64
 30  Total Charges                      7043 non-null   float64
 31  Total Refunds                      7043 non-null   float64
 32  Total Extra Data Charges           7043 non-null   int64
 33  Total Long Distance Charges        7043 non-null   float64
 34  Total Revenue                      7043 non-null   float64
 35  Customer Status                    7043 non-null   object
 36  Churn Category                     1869 non-null   object
 37  Churn Reason                       1869 non-null   object
dtypes: float64(9), int64(6), object(23)
memory usage: 2.0+ MB
```

```
Customer ID                          7043
Gender                                  2
Age                                    62
Married                                 2
Number of Dependents                   10
City                                 1106
Zip Code                             1626
Latitude                             1626
Longitude                            1625
Number of Referrals                    12
Tenure in Months                       72
Offer                                   5
Phone Service                           2
Avg Monthly Long Distance Charges    3583
Multiple Lines                          2
Internet Service                        2
Internet Type                           3
Avg Monthly GB Download                49
Online Security                         2
Online Backup                           2
Device Protection Plan                  2
Premium Tech Support                    2
Streaming TV                            2
Streaming Movies                        2
Streaming Music                         2
Unlimited Data                          2
Contract                                3
Paperless Billing                       2
Payment Method                          3
Monthly Charge                       1591
Total Charges                        6540
Total Refunds                         500
Total Extra Data Charges               16
Total Long Distance Charges          6068
Total Revenue                        6975
Customer Status                         3
Churn Category                          5
Churn Reason                           20
dtype: int64
```

The info() shows that Nulls exist in both the numerical and categorical columns.

### 2.1.2. Data `head()`

| | Customer ID | Gender | Age | Married | Number of Dependents | City | Zip Code | Latitude | Longitude | Number of Referrals | ... | Payment Method | Monthly Charge | Total Charges | Total Refunds |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0002-ORFBO | Female | 37 | Yes | 0 | Frazier Park | 93225 | 34.827662 | -118.999073 | 2 | ... | Credit Card | 65.6 | 593.30 | 0.00 |
| **1** | 0003-MKNFE | Male | 46 | No | 0 | Glendale | 91206 | 34.162515 | -118.203869 | 0 | ... | Credit Card | -4.0 | 542.40 | 38.33 |
| **2** | 0004-TLHLJ | Male | 50 | No | 0 | Costa Mesa | 92627 | 33.645672 | -117.922613 | 0 | ... | Bank Withdrawal | 73.9 | 280.85 | 0.00 |
| **3** | 0011-IGKFF | Male | 78 | Yes | 0 | Martinez | 94553 | 38.014457 | -122.115432 | 1 | ... | Bank Withdrawal | 98.0 | 1237.85 | 0.00 |
| **4** | 0013-EXCHZ | Female | 75 | Yes | 0 | Camarillo | 93010 | 34.227846 | -119.079903 | 3 | ... | Credit Card | 83.9 | 267.40 | 0.00 |

### 2.1.3. Data Stats `describe()` (Transposed)

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Age** | 7043.0 | 46.509726 | 16.750352 | 19.000000 | 32.000000 | 46.000000 | 60.000000 | 80.000000 |
| **Number of Dependents** | 7043.0 | 0.468692 | 0.962802 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 9.000000 |
| **Zip Code** | 7043.0 | 93486.070567 | 1856.767505 | 90001.000000 | 92101.000000 | 93518.000000 | 95329.000000 | 96150.000000 |
| **Latitude** | 7043.0 | 36.197455 | 2.468929 | 32.555828 | 33.990646 | 36.205465 | 38.161321 | 41.962127 |
| **Longitude** | 7043.0 | -119.756684 | 2.154425 | -124.301372 | -121.788090 | -119.595293 | -117.969795 | -114.192901 |
| **Number of Referrals** | 7043.0 | 1.951867 | 3.001199 | 0.000000 | 0.000000 | 0.000000 | 3.000000 | 11.000000 |
| **Tenure in Months** | 7043.0 | 32.386767 | 24.542061 | 1.000000 | 9.000000 | 29.000000 | 55.000000 | 72.000000 |
| **Avg Monthly Long Distance Charges** | 6361.0 | 25.420517 | 14.200374 | 1.010000 | 13.050000 | 25.690000 | 37.680000 | 49.990000 |
| **Avg Monthly GB Download** | 5517.0 | 26.189958 | 19.586585 | 2.000000 | 13.000000 | 21.000000 | 30.000000 | 85.000000 |
| **Monthly Charge** | 7043.0 | 63.596131 | 31.204743 | -10.000000 | 30.400000 | 70.050000 | 89.750000 | 118.750000 |
| **Total Charges** | 7043.0 | 2280.381264 | 2266.220462 | 18.800000 | 400.150000 | 1394.550000 | 3786.600000 | 8684.800000 |
| **Total Refunds** | 7043.0 | 1.962182 | 7.902614 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 49.790000 |
| **Total Extra Data Charges** | 7043.0 | 6.860713 | 25.104978 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 150.000000 |
| **Total Long Distance Charges** | 7043.0 | 749.099262 | 846.660055 | 0.000000 | 70.545000 | 401.440000 | 1191.100000 | 3564.720000 |
| **Total Revenue** | 7043.0 | 3034.379056 | 2865.204542 | 21.360000 | 605.610000 | 2108.640000 | 4801.145000 | 11979.340000 |

The preliminary stats shows that some numerical columns, such as Total Long Distance Charges and Total Revenue columns, might have outliers.

### 2.1.4. `info_plus()`: A More Insightful View of Data

We made a function `info_plus()` which, for every column in a single table, shows `info()`, Nulls count, `nunique()`, first 5 unique values, and percentage of Nulls.

| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| 0 | Customer ID | 7043 | 0 | object | 7043 | [0002-ORFBO, 0003-MKNFE, 0004-TLHLJ, 0011-IGKFF, 0013-EXCHZ] | 0 |
| 1 | Gender | 7043 | 0 | object | 2 | [Female, Male] | 0 |
| 2 | Age | 7043 | 0 | int64 | 62 | [37, 46, 50, 78, 75] | 0 |
| 3 | Married | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| 4 | Number of Dependents | 7043 | 0 | int64 | 10 | [0, 3, 1, 2, 4] | 0 |
| 5 | City | 7043 | 0 | object | 1106 | [Frazier Park, Glendale, Costa Mesa, Martinez, Camarillo] | 0 |
| 6 | Zip Code | 7043 | 0 | int64 | 1626 | [93225, 91206, 92627, 94553, 93010] | 0 |
| 7 | Latitude | 7043 | 0 | float64 | 1626 | [34.827662, 34.162515, 33.645672, 38.014457, 34.227846] | 0 |
| 8 | Longitude | 7043 | 0 | float64 | 1625 | [-118.999073, -118.203869, -117.922613, -122.115432, -119.079903] | 0 |
| 9 | Number of Referrals | 7043 | 0 | int64 | 12 | [2, 0, 1, 3, 8] | 0 |
| 10 | Tenure in Months | 7043 | 0 | int64 | 72 | [9, 4, 13, 3, 71] | 0 |
| 11 | Offer | 3166 | 3877 | object | 5 | [nan, Offer E, Offer D, Offer A, Offer B] | 55 |
| 12 | Phone Service | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| 13 | Avg Monthly Long Distance Charges | 6361 | 682 | float64 | 3583 | [42.39, 10.69, 33.65, 27.82, 7.38] | 9 |
| 14 | Multiple Lines | 6361 | 682 | object | 2 | [No, Yes, nan] | 9 |
| 15 | Internet Service | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| 16 | Internet Type | 5517 | 1526 | object | 3 | [Cable, Fiber Optic, DSL, nan] | 21 |
| 17 | Avg Monthly GB Download | 5517 | 1526 | float64 | 49 | [16.0, 10.0, 30.0, 4.0, 11.0] | 21 |
| 18 | Online Security | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 19 | Online Backup | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 20 | Device Protection Plan | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 21 | Premium Tech Support | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 22 | Streaming TV | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 23 | Streaming Movies | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 24 | Streaming Music | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 25 | Unlimited Data | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 26 | Contract | 7043 | 0 | object | 3 | [One Year, Month-to-Month, Two Year] | 0 |
| 27 | Paperless Billing | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| 28 | Payment Method | 7043 | 0 | object | 3 | [Credit Card, Bank Withdrawal, Mailed Check] | 0 |
| 29 | Monthly Charge | 7043 | 0 | float64 | 1591 | [65.6, -4.0, 73.9, 98.0, 83.9] | 0 |
| 30 | Total Charges | 7043 | 0 | float64 | 6540 | [593.3, 542.4, 280.85, 1237.85, 267.4] | 0 |
| 31 | Total Refunds | 7043 | 0 | float64 | 500 | [0.0, 38.33, 21.25, 30.53, 44.42] | 0 |
| 32 | Total Extra Data Charges | 7043 | 0 | int64 | 16 | [0, 10, 20, 40, 120] | 0 |
| 33 | Total Long Distance Charges | 7043 | 0 | float64 | 6068 | [381.51, 96.21, 134.6, 361.66, 22.14] | 0 |
| 34 | Total Revenue | 7043 | 0 | float64 | 6975 | [974.81, 610.28, 415.45, 1599.51, 289.54] | 0 |
| 35 | Customer Status | 7043 | 0 | object | 3 | [Stayed, Churned, Joined] | 0 |
| 36 | Churn Category | 1869 | 5174 | object | 5 | [nan, Competitor, Dissatisfaction, Other, Price] | 73 |
| 37 | Churn Reason | 1869 | 5174 | object | 20 | [nan, Competitor had better devices, Product dissatisfaction, Network reliability, Limited range of services] | 73 |

We will use this function later in order to get a comprehensive `info()` for newly created DataFrames.

Now if we filter the info plus only for the columns with missing values (Nulls > 0) we get 15 categorical columns and only 1 Numerical Column:

```
[22]: df_info[df_info['Nulls'] > 0]
```

[22]:

| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| 11 | Offer | 3166 | 3877 | object | 5 | [nan, Offer E, Offer D, Offer A, Offer B] | 55 |
| 13 | Avg Monthly Long Distance Charges | 6361 | 682 | float64 | 3583 | [42.39, 10.69, 33.65, 27.82, 7.38] | 9 |
| 14 | Multiple Lines | 6361 | 682 | object | 2 | [No, Yes, nan] | 9 |
| 16 | Internet Type | 5517 | 1526 | object | 3 | [Cable, Fiber Optic, DSL, nan] | 21 |
| 17 | Avg Monthly GB Download | 5517 | 1526 | float64 | 49 | [16.0, 10.0, 30.0, 4.0, 11.0] | 21 |
| 18 | Online Security | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 19 | Online Backup | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 20 | Device Protection Plan | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 21 | Premium Tech Support | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 22 | Streaming TV | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 23 | Streaming Movies | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 24 | Streaming Music | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 25 | Unlimited Data | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 36 | Churn Category | 1869 | 5174 | object | 5 | [nan, Competitor, Dissatisfaction, Other, Price] | 73 |
| 37 | Churn Reason | 1869 | 5174 | object | 20 | [nan, Competitor had better devices, Product dissatisfaction, Network reliability, Limited range of services] | 73 |

The info_plus() function gives us several advantages, in one table it shows us:

1. Which columns that we should drop without analysis (Nulls > %50).
2. Which columns have 2 values (Yes/No), or which have multiple unique values.
3. Which categorical columns are ordinal and which are nominal.

For instance, a column like “Churn Category” should be dropped without analysis since the number of missing values is huge (%73). The “Streaming TV” column should be considered for filling the Nulls since it has only %21 missing values. On the other hand, “Multiple Lines” column can either be considered for filling the missing values or dropping the rows with missing values since it only has %9 missing values.

The real power of using the `info_plus()` function is that it gives us a bird’s eye view on which columns and rows to drop, and which to consider for filling Nulls.

## 2.2. Data Cleaning

### 2.2.1. Dropping Rows

- Our data has no duplicate rows, or rows with less than %5 missing values. So no rows were dropped.
- The target variable (Customer Status column) has 3 unique values: Stayed: Churned, and Joined. Our classifier needs only 2 of them.
- Since the number of Joined rows doesn't exceed 7%. So we are left with one of two choices, delete all "Joined" rows or turn them into "Stayed".
- We chose to turn them into "Stayed" because we found out that the "Joined" state corresponds to "Tenure in Months" column values of 1, 2 and 3 months. But at the same time for these same 3 "Tenure in Months" values there are Customer Status with "Churned" state. Hence we can consider "Joined" to be equivalent to "Stayed" which means we'll turn those rows.

### 2.2.2. Dropping Columns

Our columns can be divided into categories according to their significance:

#### A. Relevancy

Drop irrelevant columns:

- Customer ID
- Online Security
- Online Backup
- Paperless Billing
- Payment Method

#### B. Number of Categories

Categorical columns with many unique values can't be categorized:

- Latitude → Drop
- Longitude → Drop
- Zip Code → Drop
- City (vital) → Keep because it's very significant

#### C. Missing Values %:

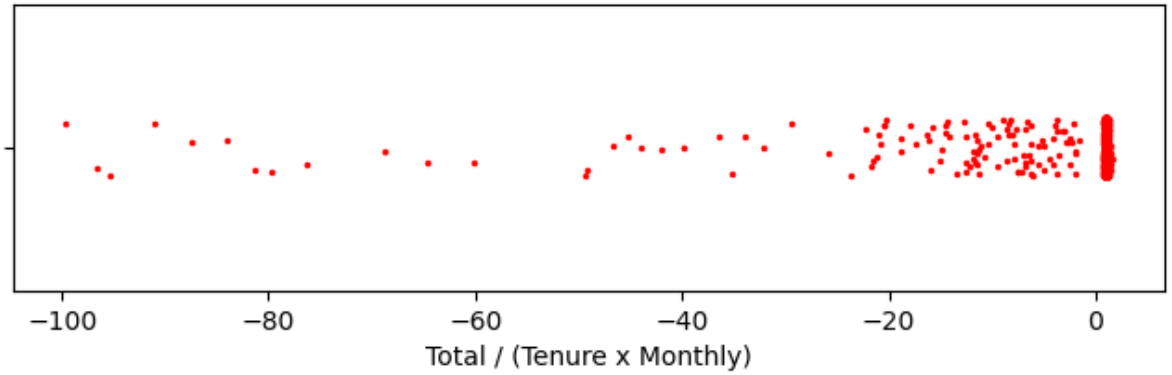
Columns with too many missing values (over %50 Nulls)

- Offers → Drop
- Churn Category → Drop
- Churn Reason → Drop

## D. Related columns (highly correlated):

### a. Monthly Charge, Total Charges → Drop Monthly Charge

- Total Charges column = Monthly Charges x Tenure in Months.
- This is true except for the rows with Monthly Charges outliers (%80 of the column values fall between 0.95 and 1.05) as shown in the Strip Plot below.




- Hence, we'll keep one of them. And since the rest of all financial charges are calculated till last quarter, we'll drop the Monthly Charge column and keep the Total Charges for consistency of data.
- Another reason to remove the Monthly Charge column is that it has outliers while the Total Charges column doesn't.

### b. Total Refunds, Total Extra Data Charges, Total Long Distance Charges, Total Revenue

- From the dataset columns dictionary in Appendix A:
  Total Revenue = Total Charges - Total Refunds + Total Extra Data Charges + Total Long Distance Charges
- To avoid multicollinearity, So we can remove total revenue or the other 4 values.
- However, since charges can be of great significance in a client's churn decision, we'll remove the Total Revenue column and keep the other 4.

### c. Average Monthly Long Distance Charges, Total Long Distance Charges

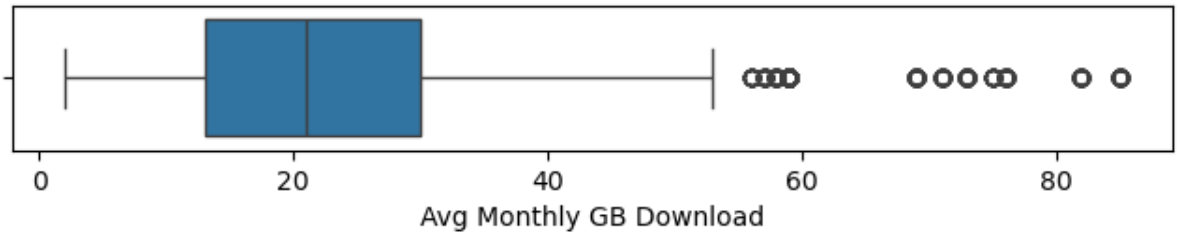
- Similar to the previous section, the Avg Monthly Long Distance Charges and the Total Long Distance Charges columns can be calculated from each other directly. Hence, we will drop one of them to avoid multicollinearity.
- We chose to drop the Avg Monthly Long Distance Charges for two reasons:
  i. It has 9 missing values while the Total Long Distance Charges column doesn't have any.
  ii. The Total Long Distance Charges column had already been chosen to keep in the previous section.

### 2.2.3. Handling Missing Values

#### A. Missing Values of Numerical Columns

| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| 11 | Avg Monthly GB Download | 5517 | 1526 | float64 | 49 | [16.0, 10.0, 30.0, 4.0, 11.0] | 21 |

Checking the numerical columns with missing values for outliers using BoxPlot.




Since the Avg Monthly GB Download column has outliers, we'll fill the missing values with the median rather than the mean.

#### B. Missing Values of Categorical Columns

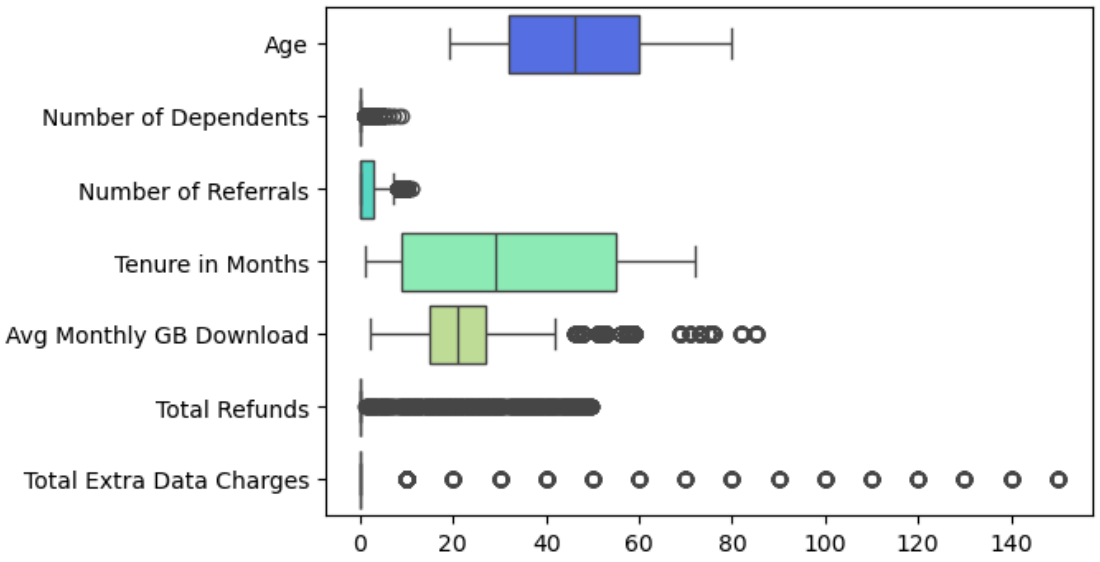
| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| 8 | Multiple Lines | 6361 | 682 | object | 2 | [No, Yes, nan] | 9 |
| 10 | Internet Type | 5517 | 1526 | object | 3 | [Cable, Fiber Optic, DSL, nan] | 21 |
| 12 | Device Protection Plan | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 13 | Premium Tech Support | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 14 | Streaming TV | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |
| 15 | Streaming Movies | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 16 | Streaming Music | 5517 | 1526 | object | 2 | [No, Yes, nan] | 21 |
| 17 | Unlimited Data | 5517 | 1526 | object | 2 | [Yes, No, nan] | 21 |

We will fill the categorical values missing values using the mode.

### 2.2.4. Handling Outliers

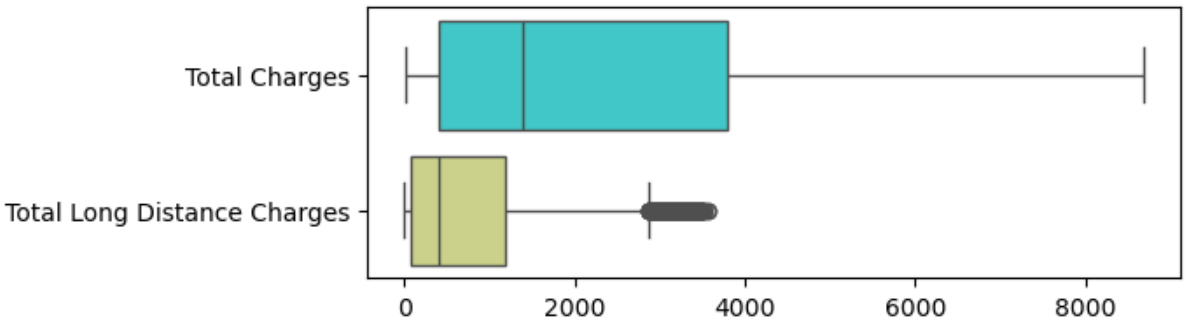
For a better view of the values, we separated the categorical columns that has missing values into two categories, the small values category and the large values columns:

C. Small Values Columns:

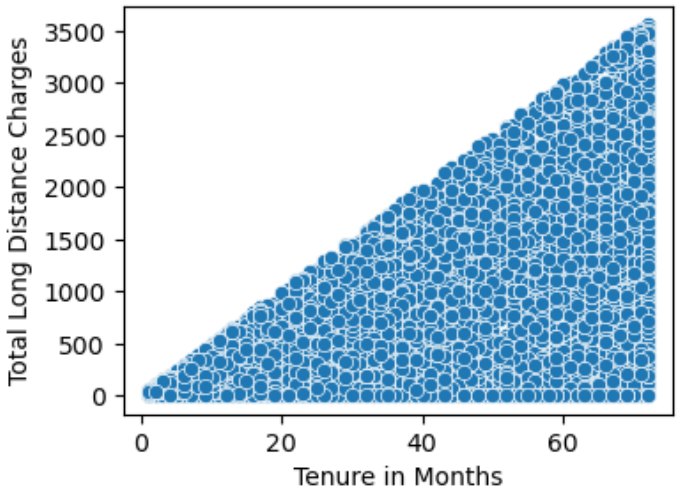



- Age, Number of Dependants, and Tenure in Months columns: no outliers.
- Number of Referrals column: Shows outliers, but from the stats description the max value is 11 which is reasonable. So, no outliers here as well.
- Avg Monthly GB Download: it can be explained as some users have excessive internet usage compared to others. So even if they are treated as outliers in filling nulls they are not outliers in the "wrong values" sense.

D. Large Values Columns:



This scatterplot shows that the Total Long Distance Charges column outliers correspond to high Tenure in Months column values, i.e older subscribers. Hence these are not actually outliers




## 2.3. Encoding Categorical Data Columns

This table shows the current data categorical columns after filling the missing data.

| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| **0** | Gender | 7043 | 0 | object | 2 | [Female, Male] | 0 |
| **1** | Married | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **2** | City | 7043 | 0 | object | 1106 | [Frazier Park, Glendale, Costa Mesa, Martinez, Camarillo] | 0 |
| **3** | Phone Service | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **4** | Multiple Lines | 7043 | 0 | object | 2 | [No, Yes] | 0 |
| **5** | Internet Service | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **6** | Internet Type | 7043 | 0 | object | 3 | [Cable, Fiber Optic, DSL] | 0 |
| **7** | Device Protection Plan | 7043 | 0 | object | 2 | [No, Yes] | 0 |
| **8** | Premium Tech Support | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **9** | Streaming TV | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **10** | Streaming Movies | 7043 | 0 | object | 2 | [No, Yes] | 0 |
| **11** | Streaming Music | 7043 | 0 | object | 2 | [No, Yes] | 0 |
| **12** | Unlimited Data | 7043 | 0 | object | 2 | [Yes, No] | 0 |
| **13** | Contract | 7043 | 0 | object | 3 | [One Year, Month-to-Month, Two Year] | 0 |
| **14** | Customer Status | 7043 | 0 | object | 2 | [Stayed, Churned] | 0 |

### 2.3.1. Encoding Ordinal Data

The Contract column is the only ordinal data column. Using mapping to replace the column values:

| Before | Encoded |
|---|---|
| Month-to-Month | 1 |
| One Year | 2 |
| Two Year | 3 |

### 2.3.2. Encoding Nominal Data

#### A. Columns with 2-State Nominal Data

These are columns with values that range between one of two values:

1. Gender Column [Male/Femal]
2. Customer Status Column [Stayed/Churned]
3. Married, Phone Service, Multiple Lines, Internet Service, Device Protection Plan, Premium Technical Support, Streaming TV, Streaming Movies, Streaming Music and Unlimited Data [Yes/No]

We will also use mapping here:

| Before | | | Encoded |
|---|---|---|---|
| Male | Churned | Yes | 1 |
| Female | Stayed | No | 0 |

#### B. Columns with Number of States > 2

In these cases we will use Binary encoding instead of One-Hot Encoding to reduce the number of features as much as possible.

1. Internet Type [Cable, Fiber Optic, DSL] → 2 bits → replaced by 2 columns
2. City [1106 Cities] → 11 bits → replaced by 11 columns

# 3. ML Model Development & MLOps

## 3.1. Introduction

After the Data-Preprocessing Phase, we obtained the following DataFrame:

| | Column | Non-Null | Nulls | DType | N Unique | First 5 Unique | % Missing |
|---|---|---|---|---|---|---|---|
| 0 | Gender | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 1 | Age | 7043 | 0 | int64 | 62 | [37, 46, 50, 78, 75] | 0 |
| 2 | Married | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 3 | Number of Dependents | 7043 | 0 | int64 | 10 | [0, 3, 1, 2, 4] | 0 |
| 4 | City_0 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 5 | City_1 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 6 | City_2 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 7 | City_3 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 8 | City_4 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 9 | City_5 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 10 | City_6 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 11 | City_7 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 12 | City_8 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 13 | City_9 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 14 | City_10 | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 15 | Number of Referrals | 7043 | 0 | int64 | 12 | [2, 0, 1, 3, 8] | 0 |
| 16 | Tenure in Months | 7043 | 0 | int64 | 72 | [9, 4, 13, 3, 71] | 0 |
| 17 | Phone Service | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 18 | Multiple Lines | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 19 | Internet Service | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 20 | Internet Type_0 | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 21 | Internet Type_1 | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 22 | Avg Monthly GB Download | 7043 | 0 | float64 | 49 | [16.0, 10.0, 30.0, 4.0, 11.0] | 0 |
| 23 | Device Protection Plan | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 24 | Premium Tech Support | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 25 | Streaming TV | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 26 | Streaming Movies | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 27 | Streaming Music | 7043 | 0 | int64 | 2 | [0, 1] | 0 |
| 28 | Unlimited Data | 7043 | 0 | int64 | 2 | [1, 0] | 0 |
| 29 | Contract | 7043 | 0 | int64 | 3 | [2, 1, 3] | 0 |
| 30 | Total Charges | 7043 | 0 | float64 | 6540 | [593.3, 542.4, 280.85, 1237.85, 267.4] | 0 |
| 31 | Total Refunds | 7043 | 0 | float64 | 500 | [0.0, 38.33, 21.25, 30.53, 44.42] | 0 |
| 32 | Total Extra Data Charges | 7043 | 0 | int64 | 16 | [0, 10, 20, 40, 120] | 0 |
| 33 | Total Long Distance Charges | 7043 | 0 | float64 | 6068 | [381.51, 96.21, 134.6, 361.66, 22.14] | 0 |
| 34 | Customer Status | 7043 | 0 | int64 | 2 | [0, 1] | 0 |

## 3.2. Machine Learning Models Used

We will use 5 different machine learning models as classifiers for churn prediction:

1. Logistic Regression (Baseline Model)
2. Random Forest
3. Gradient Boosting
4. eXtreme Gradient Boosting (XGBoosting)
5. Support Vector Machines

With each of these models, we will use the basic version of the model algorithm, and then use the model with the assistance of the Grid Search Algorithm. Finally, we will compare the results in all cases and select the model with the best results for deployment.

## 3.3. Model Performance Evaluation

In our evaluation of each of the models, we will use the following metrics:

1. Accuracy = $\frac{T_P + T_N}{T_P + T_N + F_P + F_N}$
2. Precision = $\frac{T_P}{T_P + F_P}$
3. Recall (sensitivity) = $\frac{T_P}{T_P + F_N}$
4. F1-Score = $2 \times \frac{Precision \times Recall}{Precision + Recall}$

Where:
$T_P$: True Positive
$T_N$ is the True Negative
$F_P$ is the False Positive (Type I Error)
$F_N$ is the False Negative (Type II Error)

However, since customer churn prediction works like an alarming system mainly made to identify if someone is going to churn, a False Negative would be the worst type of error. Hence to minimize False Negatives, we need to maximize the Recall metric. Consequently, Recall would be the most important metric to us, and the most desired to be maximized compared to the other metrics.

## 3.4. Logistic Regression (Baseline Model)

### 3.4.1. Why Logistic Regression?

It is one of the simplest classification models. It will be our baseline model used as the reference or benchmark for other models' performances.

### 3.4.2. Data Preparation

Before training the model:

1. Features and targets were separated.
2. Data was split into Training (70%) and Testing (30%) with stratification.
3. For the basic Logistic Regression model, StandardScaler was applied to normalize numerical features.
4. For Logistic Regression with Grid Search model, a Scikit-Learn Scaler-Model pipeline is used to conduct that train/test feature scaling.

### 3.4.3. Logistic Regression Basic Model

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| Random State | 30 | Accuracy | 0.85 |
| Max Iterations | 300 | Precision | 0.74 |
| | | Recall | 0.73 |
| | | F1 Score | 0.73 |

### 3.4.4. Logistic Regression with Grid Search

Grid Search Cross-Validation was used to tune key hyperparameters to optimize recall, which is critical for detecting churners. Regularization L2 was used, and the hyperparameter under tuning was the model complexity C = $1/\lambda$ which is the inverse of the regularization parameter (penalty value) $\lambda$.

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| C (Tunable) | 0.1, 1, 10, 100 | Accuracy | 0.82 |
| Penalty | L2 Reg. | Precision | 0.85 |
| Max Iterations | 200, 300, 500, 700 | Recall | 0.82 |
| | | F1 Score | 0.82 |

### 3.4.5. Conclusion

Regularized Logistic Regression model with Grid Search Cross Validation shows promising results and can be considered deployment in our situation. However, we are seeking to deploy the model with the highest performance metrics, which we will find out after comparing all the models in the project.

## 3.5. Random Forest

### 3.5.1. Why Random Forest?

The random forest classifier is an ensemble learning method that builds multiple decision trees during training and merges their predictions to produce a more accurate and stable classification:

- It is considered a Tree-Based ML algorithm, it is immune to large value difference between features, which means it doesn't need feature scaling.
- It can provide high performance in cases of small dataset size and a reasonable number of features, just like our case.

### 3.5.2. Data Preparation

Before training the model:

1. Features and targets were separated.
2. Data was split into Training (70%) and Testing (30%) with stratification.
3. no feature scaling was introduced to either model.

### 3.5.3. Random Forest Basic Model Training

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| Random State | 30 | Accuracy | 0.87 |
| Class Weight | Balanced | Precision | 0.87 |
| | | Recall | 0.73 |
| | | F1 Score | 0.86 |

### 3.5.4. Random Forest with Grid Search

Grid Search was used to tune the following hyperparameters to optimize recall:

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| n_estimators / # Trees (Tune) | 200, 300, 400, 500 | Accuracy | 0.87 |
| Max Depth (Tune) | 10, 20, 30, None | Precision | 0.87 |
| Min Samples Split (Tune) | 2, 5, 10 | Recall | 0.87 |
| Min Samples Leaf (Tune) | 1, 2, 4 | F1 Score | 0.87 |
| Max Features (Tune) | Sqrt, Log2 | | |

### 3.5.5. Conclusion

Random Forest has the highest results among all models and appears to be far superior compared to Logistic Regression. However, Grid Search Cross Validation doesn't appear to have added much to the performance of the Random Forest model. So, Random Forest with Grid Search will be our choice for deployment.

## 3.6. Gradient Boosting

Gradient Boosting is an ensemble learning technique that builds a strong predictive model by combining many weak learners, typically decision trees.

The model is trained sequentially, where each tree attempts to correct the errors made by the previous one. This makes Gradient Boosting especially effective for complex, non-linear datasets such as telecom customer churn.

### 3.6.1. Why Gradient Boosting?

- Works well for binary classification problems like churn prediction.
- Captures complex patterns and variable interactions.
- Offers a balanced performance without heavy hyperparameter tuning.
- Serves as a strong baseline model for comparison with more advanced boosting methods.

### 3.6.2. Data Preparation

Before training the model:

1. Features and targets were separated.
2. Data was split into Training (80%) and Testing (20%) with stratification.
3. StandardScaler was applied to normalize numerical features.

### 3.6.3. Gradient Boosting Baseline Model Training

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| n_estimators / # Trees | 100 | Accuracy | 0.87 |
| Learning Rate | 0.1 | Precision | 0.86 |
| max_depth | 3 | Recall | 0.67 |
| random_state | 42 | F1 Score | 0.86 |

This model served as the initial benchmark.

### 3.6.4. Gradient Boosting Hyperparameter Tuning

GridSearchCV was used to tune the following key hyperparameters to optimize recall, which is critical for detecting churners.

Search space included:

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| n_estimators / # Trees (Tune) | 100, 150, 200, 250 | Accuracy | 0.87 |
| Max Depth (Tune) | 3, 4, 5, 6 | Precision | 0.86 |
| Min Samples Split (Tune) | 2, 5, 10 | Recall | 0.70 |
| Min Samples Leaf (Tune) | 1, 2, 4 | F1 Score | 0.86 |
| Learning Rate | 0.01, 0.02, 0.05, 0.1 | | |
| Subsample | 0.8, 0.85, 0.9, 0.095 | | |

### 3.6.5. Conclusion

- Gradient Boosting provided a strong starting point and offered competitive performance.
- However, more advanced boosting algorithms like XGBoost delivered superior accuracy and recall, especially on imbalanced telecom churn data.

## 3.7. eXtreme Gradient Boosting (XGBoost)

XGBoost is an optimized and high-performance implementation of Gradient Boosting.
It is widely used across machine learning competitions and real-world applications due to its speed, regularization capabilities, and excellent handling of complex datasets.

### 3.7.1. Why XGBoost?

- Significantly faster than traditional Gradient Boosting through parallel computation.
- Includes built-in regularization to reduce overfitting.
- Offers excellent control for handling imbalanced datasets using `scale_pos_weight`.
- Delivers robust and stable performance on churn prediction tasks.

### 3.7.2. Data Preparation

Since XGBoost does not directly accept string categorical features:

- All categorical columns were encoded prior to training.

### 3.7.3. XGBoosting Basic Model Training

XGBoost classifier was trained with the default values of the hyperparameters:

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| n_estimators / # Trees | Default = 100 | Accuracy | 0.84 |
| Max Depth | Default = 6 | Precision | 0.84 |
| colsample_bytree | Default = 1.0 | Recall | 0.72 |
| scale_pos_weight | 2.53 | F1 Score | 0.84 |
| Learning Rate | Default = 0.3 | | |
| Subsample | Default = 1.0 | | |

### 3.7.4. XGBoosting Hyperparameter Tuning (GridSearchCV)

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| n_estimators / # Trees | 100, 150, 200, 250 | Accuracy | 0.83 |
| Max Depth | 3, 5, 7 | Precision | 0.85 |
| colsample_bytree | 0.8, 0.9 | Recall | 0.84 |
| scale_pos_weight | 2.53 | F1 Score | 0.84 |
| Learning Rate | 0.01, 0.02, 0.05, 0.1 | | |
| Subsample | 0.8, 0.85, 0.9, 0.095 | | |

Using GridSearchCV, the model was tuned to maximize recall, ensuring better identification of high-risk churn customers.

### 3.7.5. Results

- It was more effective at handling the class imbalance present in churn data.
- Delivered more stable and accurate predictions overall.

### 3.7.6. Conclusion

XGBoost outperformed the traditional Gradient Boosting model, making it one of the best-performing models in this project for predicting telecom customer churn.

## 3.8. Support Vector Machines

SVM works by finding the best boundary (called a hyperplane) that separates the data into classes.It chooses this boundary so that the margin (distance between the boundary and the closest data points) is as large as possible.

### 3.8.1. Data Preparation

1. Categorical columns were encoded
2. Splitting data into training and testing 80-20 respectively
3. Scaling the featured columns ( training and testing data each one separately)

### 3.8.2. Model Training

The Hyperparameters used in SVM:

1. C Values: controls how much the model tries to avoid misclassifying training data
2. Kernel: A kernel defines how the SVM decides the shape of the boundary that separates the classes.
3. Gamma: controls how far the influence of a single training point reaches.(Has no affect on linear kernel)

Use 5-Fold Cross-Validation: This reduces overfitting and provides a stable estimate of model performance.

1. Training data is divided into 5 equal folds
2. The model trains 5 times
3. Each run uses a different fold as validation
4. The final score is the average of the 5 results

### 3.8.3. Evaluation Metrics

1) Accuracy
2) Recall
3) Loss
4) Precision
5) F1-score

### 3.8.4. Model Logging with MLflow

MLflow was used to track:

- Hyperparameters (C, kernel, gamma)
- Model metrics (accuracy, precision, recall, F1)
- Final trained model

The MLflow UI provides an easy comparison between runs, making it simple to identify the best-performing configuration.

### 3.8.5. Results

SVM without GridSearch

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| C | 0.01 | Accuracy | 0.84 |
| Kernel | Linear | Precision | 0.84 |
| | | Recall | 0.84 |
| | | F1 Score | 0.84 |

SVM with GridSearch

| Hyper Parameter | Value | Results | Value |
|---|---|---|---|
| C | 0.001, 0.01, 0.1, 1, 10 | Accuracy | 0.85 |
| Kernel | Linear, RBF, Polynomial | Precision | 0.85 |
| Gamma | Scale, Auto | Recall | 0.85 |
| | | F1 Score | 0.85 |

# 4. Model Deployment

## 4.1. Model of Choice: Random Forest with Grid Search

The table below shows a comparison between the results of the 5 Machine Learning models. The Random Forest is obviously the winner here. Thus, it will be our chosen machine learning model for deployment

| # | Model | Metric | Algorithm | |
|---|---|---|---|---|
| | | | Basic | Grid Search |
| 1 | Logistic Regression | Accuracy | 0.85 | 0.82 |
| | | Precision | 0.74 | 0.85 |
| | | Recall | 0.73 | 0.82 |
| | | F1-Score | 0.73 | 0.82 |
| **2** | **Random Forest** | **Accuracy** | 0.87 | **0.87** |
| | | **Precision** | 0.87 | **0.87** |
| | | **Recall** | 0.73 | **0.87** |
| | | **F1-Score** | 0.86 | **0.87** |
| 3 | Gradient Boosting | Accuracy | 0.87 | 0.87 |
| | | Precision | 0.86 | 0.86 |
| | | Recall | 0.67 | 0.70 |
| | | F1-Score | 0.86 | 0.86 |
| 4 | XGBoosting | Accuracy | 0.84 | 0.83 |
| | | Precision | 0.84 | 0.85 |
| | | Recall | 0.72 | 0.84 |
| | | F1-Score | 0.84 | 0.84 |
| 5 | Support Vector Machine | Accuracy | 0.84 | 0.85 |
| | | Precision | 0.84 | 0.85 |
| | | Recall | 0.84 | 0.85 |
| | | F1-Score | 0.84 | 0.85 |

## 4.2. Streamlit Setup

Required Files:

### 4.2.1. File: `streamlit_app.py`

A Python file (script) where you write code using the Streamlit library to build interactive web applications for data science, machine learning, and dashboards, allowing you to turn Python scripts into shareable, beautiful apps with minimal code, often without needing HTML, CSS, or JavaScript. It's the file you run to launch your Streamlit app, which can contain UI elements like charts, text, sliders, and buttons.

### 4.2.2. File: `requirements.txt`

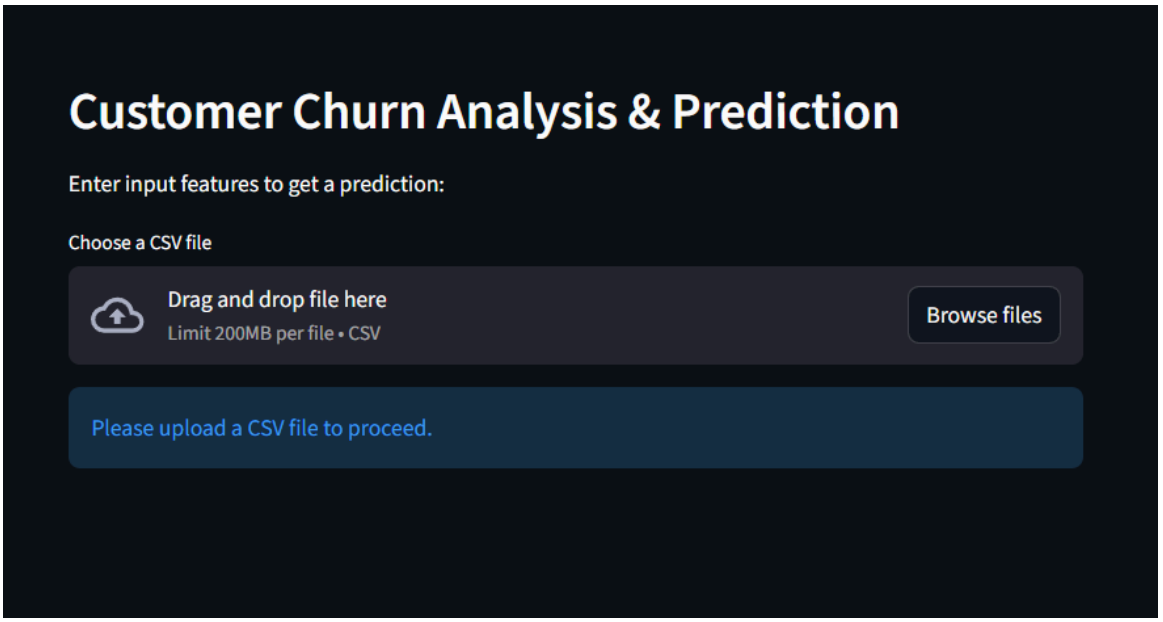
A plain text file listing all Python libraries (like pandas, numpy, streamlit itself) your app needs, enabling easy environment setup and deployment by telling pip what to install, ensuring your app runs consistently by specifying package versions.

### 4.2.3. Platform Accounts

1. We need a `streamlit.io` account and a `github.com` account.
2. Create a GitHub repo to host streamlit application files (`streamlit_app.py` and `requirements.txt` files).
3. Create a streamlit application and connect it to the GitHub repo.

### 4.2.4. Project Streamlit Webapp

The web application URL for this project is `https://cognitix.streamlit.app`.



# Appendix (A): Dataset Columns Dictionary

| No | Column | Description |
|---|---|---|
| 0 | CustomerID | A unique ID that identifies each customer |
| 1 | Gender | The customer's gender: Male, Female |
| 2 | Age | The customer's current age, in years, at the time the fiscal quarter ended (Q2 2022) |
| 3 | Married | Indicates if the customer is married: Yes, No |
| 4 | Number of Dependents | Indicates the number of dependents that live with the customer (dependents could be children, parents, grandparents, etc.) |
| 5 | City | The city of the customer's primary residence in California |
| 6 | Zip Code | The zip code of the customer's primary residence |
| 7 | Latitude | The latitude of the customer's primary residence |
| 8 | Longitude | The longitude of the customer's primary residence |
| 9 | Number of Referrals | Indicates the number of times the customer has referred a friend or family member to this company to date |
| 10 | Tenure in Months | Indicates the total amount of months that the customer has been with the company by the end of the quarter specified above |
| 11 | Offer | Identifies the last marketing offer that the customer accepted: None, Offer A, Offer B, Offer C, Offer D, Offer E |
| 12 | Phone Service | Indicates if the customer subscribes to home phone service with the company: Yes, No |
| 13 | Avg Monthly Long Distance Charges | Indicates the customer's average long distance charges, calculated to the end of the quarter specified above (if the customer is not subscribed to home phone service, this will be 0) |
| 14 | Multiple Lines | Indicates if the customer subscribes to multiple telephone lines with the company: Yes, No (if the customer is not subscribed to home phone service, this will be No) |
| 15 | Internet Service | Indicates if the customer subscribes to Internet service with the company: Yes, No |
| 16 | Internet Type | Indicates the customer's type of internet connection: DSL, Fiber Optic, Cable (if the customer is not subscribed to internet service, this will be None) |
| 17 | Avg Monthly GB Download | Indicates the customer's average download volume in gigabytes, calculated to the end of the quarter specified above (if the customer is not subscribed to internet service, this will be 0) |
| 18 | Online Security | Indicates if the customer subscribes to an additional online security service provided by the company: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 19 | Online Backup | Indicates if the customer subscribes to an additional online backup service provided by the company: Yes, No (if the customer is not subscribed to internet service, this will be No) |

| No | Column | Description |
|---|---|---|
| 20 | Device Protection Plan | Indicates if the customer subscribes to an additional device protection plan for their Internet equipment provided by the company: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 21 | Premium Tech Support | Indicates if the customer subscribes to an additional technical support plan from the company with reduced wait times: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 22 | Streaming TV | Indicates if the customer uses their Internet service to stream television programing from a third party provider at no additional fee: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 23 | Streaming Movies | Indicates if the customer uses their Internet service to stream movies from a third party provider at no additional fee: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 24 | Streaming Music | Indicates if the customer uses their Internet service to stream music from a third party provider at no additional fee: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 25 | Unlimited Data | Indicates if the customer has paid an additional monthly fee to have unlimited data downloads/uploads: Yes, No (if the customer is not subscribed to internet service, this will be No) |
| 26 | Contract | Indicates the customer's current contract type: Month-to-Month, One Year, Two Year |
| 27 | Paperless Billing | Indicates if the customer has chosen paperless billing: Yes, No |
| 28 | Payment Method | Indicates how the customer pays their bill: Bank Withdrawal, Credit Card, Mailed Check |
| 29 | Monthly Charge | Indicates the customer's current total monthly charge for all their services from the company |
| 30 | Total Charges | Indicates the customer's total charges, calculated to the end of the quarter specified above |
| 31 | Total Refunds | Indicates the customer's total refunds, calculated to the end of the quarter specified above |
| 32 | Total Extra Data Charges | Indicates the customer's total charges for extra data downloads above those specified in their plan, by the end of the quarter specified above |
| 33 | Total Long Distance Charges | Indicates the customer's total charges for long distance above those specified in their plan, by the end of the quarter specified above |
| 34 | Total Revenue | Indicates the company's total revenue from this customer, calculated to the end of the quarter specified above (Total Charges - Total Refunds + Total Extra Data Charges + Total Long Distance Charges) |
| 35 | Customer Status | Indicates the status of the customer at the end of the quarter: Churned, Stayed, or Joined |
| 36 | Churn Category | A high-level category for the customer's reason for churning, which is asked when they leave the company: Attitude, Competitor, Dissatisfaction, Other, Price (directly related to Churn Reason) |
| 37 | Churn Reason | A customer's specific reason for leaving the company, which is asked when they leave the company (directly related to Churn Category) |