

Information

Matthew Blasa

Student ID: 001781641

MS Data Analytics (05/01/2021)

Program Mentor: Kirk Kelly

(503)805-0297

mblasa@wgu.edu

Part I: Research Question

This project uses the data that was used in a previous task. The objective of this analysis is to use the majority of the data set provided in the course to determine how customer churn might be predicted using logistic regression. I will be performing calculations that will reduce the available features to significant variables that affect customer churn. I hope to obtain a strong predictive model to assist the telecommunications company with decisions on how to avoid the loss of customers.

A1. Research Questions

What feature features are most significant that affect customer churn? And what is the weight that those factors most affect customer churn?

A2. Objectives & Goals:

Stakeholders in the company will benefit from this analysis by identifying what factors are heavily responsible for customer churn. This will provide insight for decisions in whether or not to expand customer data limits, provide unlimited (or metered) media streaming, improve customer experience, or any other factors that will reduce customer churn.

B1. Assumptions of Logistic Regression

- Response variable is binary. There only two possible outcomes.
- Observations are Independent. Observations are independent of each other.
- There is No Multicollinearity Among Explanatory Variables. Two or more are not highly correlated with each other
- There is a Linear Relationship Between Explanatory Variables and the Logit of the Response Variable
- The Sample Size is Sufficiently Large.
- It is the logarithm of the odds of achieving 1. (Linear Relationship between explanatory variables and logit variables)

<https://www.statology.org/assumptions-of-logistic-regression/>

B2 Tool Benefits:

I will use the Python, since I use it at work to perform EDAs and clean data. I'll be using jupyter notebook, since it allows me to use markdowns to answer questions and program at the same time. To save time, I will be using several data science Python libraries to simplify the problem. The following will be used:

- NumPy - to work with arrays
- Pandas - used to create dataframes.
- Matplotlib - plotting charts
- Scikit-learn - for PCA, Machine Learning, and Normalization.
- SciPy - used for mathematical transformations.
- Seaborn - visualization of more complex graphs

B3. Appropriateness of Logistic Regression

Logistic regression is an appropriate technique to analyze the research question because or dependent variable is binomial, Yes or No. We want to find out what the likelihood of customer churn is for individual customers, based on a list of independent variables (area type, job, children, age, income, etc.). It will improve our understanding of increased probability of churn as we include or remove different independent variables & find out whether or not they have a positive or negative relationship to our target variable.

C1 Data Goals

This project uses the data that was previously cleaned in the Data Cleaning Assignment. The majority of the data set is used to determine the amount of average amount of data used, in GB, in a year by the customer. I plan to reduce the list of variables not by what might intuitively belong together, but by looking at numerical calculations that suggest which variables are significant to answering the research question. With variables, I aim to get a predictive model to assist the telecommunications company to predict customers and target customers who use the most

The following steps will be taken to prepare the data for analysis:

1. Import data to pandas dataframe
2. Determine variable types and those which may require further investigation
3. Convert binary variables to yes = 1 and no = 0
4. Investigate potential categorical variables using bar charts, then convert categorical values to dummy variables
5. Drop columns that will not be used in regression analysis

My approach will include:

1. Import data to pandas dataframe

2. Determine variable types and those which may require further investigation
3. Convert binary variables to yes = 1 and no = 0
4. Investigate potential categorical variables using bar charts, then convert categorical values to dummy variables
5. Drop columns that will not be used in regression analysis
6. Find outliers that may create or hide statistical significance using histograms.
7. Substitute missing data with meaningful measures of central tendency (mean, median or mode)

Important to the process is the variable of "Bandwidth_GB_Year" (the average yearly amount of data used, in GB, per customer) which will be our target variable.

When analyzing the data we will examine the following continuous predictor variables:

- Children
- Income
- Outage_sec_perweek
- Email
- Contacts
- Yearly_equip_failure
- MonthlyCharge
- Bandwidth_GB_Year

We will also examine the following categorical predictor variables:

- Churn: Whether the customer discontinued service within the last month (yes, no)
- Techie: Whether the customer considers themselves technically inclined (based on customer questionnaire when they signed up for services) (yes, no)
- Contract: The contract term of the customer (month-to-month, one year, two year)
- Port_modem: Whether the customer has a portable modem (yes, no)
- Tablet: Whether the customer owns a tablet such as iPad, Surface, etc. (yes, no)
- InternetService: Customer's internet service provider (DSL, fiber optic, None)
- Phone: Whether the customer has a phone service (yes, no)
- Multiple: Whether the customer has multiple lines (yes, no)
- OnlineSecurity: Whether the customer has an online security add-on (yes, no)
- OnlineBackup: Whether the customer has an online backup add-on (yes, no)
- DeviceProtection: Whether the customer has device protection add-on (yes, no)
- TechSupport: Whether the customer has a technical support add-on (yes, no)
- StreamingTV: Whether the customer has streaming TV (yes, no)
- StreamingMovies: Whether the customer has streaming movies (yes, no)

We will also examine ordinal predictor variables from the survey responses from customers. In the surveys, customers rated eight customer service factors on a scale from 1 to 8.

- Item1: Timely response
- Item2: Timely fixes
- Item3: Timely replacements
- Item4: Reliability
- Item5: Options
- Item6: Respectful response
- Item7: Courteous exchange

C2 Summary Statistics

Logistic regression is an appropriate technique to analyze the research question because our dependent variable is binomial, Yes or No. We want to find out what the likelihood of customer churn is for individual customers, based on a list of independent variables (area type, job, children, age, income, etc.). It will improve our understanding of increased probability of churn as we include or remove different independent variables & find out whether or not they have a positive or negative relationship to our target variable.

C3. Steps to Prepare the Data

The following steps will be taken to prepare the data for analysis:

1. Import data to pandas dataframe
2. Determine variable types and those which may require further investigation
3. Convert binary variables to yes = 1 and no = 0
4. Investigate potential categorical variables using bar charts, then convert categorical values to dummy variables
5. Drop columns that will not be used in logistic regression

This project uses the data that was previously cleaned in the Jupyter Notebook "Data Cleaning". The objective of this analysis is to use the majority of the data set provided in the course to determine how customer churn might be predicted using logistic regression. I will be performing calculations that will reduce the available features to those variables which will be significant in answering the research question "Which factors can help predict whether a customer will end their contract (churn)?" I hope to obtain a strong predictive model to assist the telecommunications company with decisions on how to avoid the loss of customers.

Steps to Prepare the Data

The following steps will be taken to prepare the data for analysis:

1. Import data to pandas dataframe
2. Determine variable types and those which may require further investigation
3. Convert binary variables to yes = 1 and no = 0
4. Investigate potential categorical variables using bar charts, then convert categorical values to dummy variables using one-hot encoding
5. Drop columns that will not be used in logistic regression

Step 1

In [156...

```
# Import necessary libraries
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
import statsmodels.formula.api as smf
from statsmodels.stats.outliers_influence import variance_inflation_factor
from sklearn.model_selection import cross_val_predict, train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFE
from sklearn.metrics import classification_report
import warnings
warnings.filterwarnings('ignore') # Ignore warning messages for readability
```

In [157...

```
# Read in dataset and view head
df = pd.read_csv('churn_use.csv')
pd.options.display.max_columns = None
df.head()
```

Out[157...

	CaseOrder	Customer_id	Interaction	UID	City	State
0	1	K409198	aa90260b-4141-4a24-8e36-b04ce1f4f77b	e885b299883d4f9fb18e39c75155d990	Point Baker	AK
1	2	S120509	fb76459f-c047-4a9d-8af9-e0f7d4ac2524	f2de8bef964785f41a2959829830fb8a	West Branch	MI
2	3	K191035	344d114c-3736-4be5-98f7-c72c281e2d35	f1784cfa9f6d92ae816197eb175d3c71	Yamhill	OR
3	4	D90850	abfa2b40-2d43-4994-b15a-989b8c79e311	dc8a365077241bb5cd5ccd305136b05e	Del Mar	CA
4	5	K662701	68a861fd-0d20-4e51-a587-8a90407ee574	aabb64a116e83fdc4befc1fbab1663f9	Needville	TX

Step 2

In [158...

```
# View list of columns, data types, and missing values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 50 columns):
#   Column                Non-Null Count  Dtype
---  -
0   CaseOrder              10000 non-null  int64
1   Customer_id            10000 non-null  object
```

```

2   Interaction      10000 non-null object
3   UID              10000 non-null object
4   City             10000 non-null object
5   State            10000 non-null object
6   County           10000 non-null object
7   Zip              10000 non-null int64
8   Lat              10000 non-null float64
9   Lng              10000 non-null float64
10  Population       10000 non-null int64
11  Area             10000 non-null object
12  TimeZone         10000 non-null object
13  Job              10000 non-null object
14  Children         10000 non-null int64
15  Age              10000 non-null int64
16  Income           10000 non-null float64
17  Marital          10000 non-null object
18  Gender           10000 non-null object
19  Churn            10000 non-null object
20  Outage_sec_perweek 10000 non-null float64
21  Email            10000 non-null int64
22  Contacts         10000 non-null int64
23  Yearly_equip_failure 10000 non-null int64
24  Techie           10000 non-null object
25  Contract         10000 non-null object
26  Port_modem       10000 non-null object
27  Tablet           10000 non-null object
28  InternetService  10000 non-null object
29  Phone            10000 non-null object
30  Multiple         10000 non-null object
31  OnlineSecurity   10000 non-null object
32  OnlineBackup     10000 non-null object
33  DeviceProtection 10000 non-null object
34  TechSupport      10000 non-null object
35  StreamingTV      10000 non-null object
36  StreamingMovies  10000 non-null object
37  PaperlessBilling 10000 non-null object
38  PaymentMethod    10000 non-null object
39  Tenure            10000 non-null float64
40  MonthlyCharge    10000 non-null float64
41  Bandwidth_GB_Year 10000 non-null float64
42  Item1            10000 non-null int64
43  Item2            10000 non-null int64
44  Item3            10000 non-null int64
45  Item4            10000 non-null int64
46  Item5            10000 non-null int64
47  Item6            10000 non-null int64
48  Item7            10000 non-null int64
49  Item8            10000 non-null int64
dtypes: float64(7), int64(16), object(27)
memory usage: 3.8+ MB

```

Convert to Binary and Dummy Variables

In [159...

```
df2 = df.copy()
```

In [160...

```

# Convert binary variables into yes = 1, no = 0 (ref 1)
cols = ['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'PaperlessBilling', 'PaymentMethod', 'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8']
df[cols] = df[cols].replace(to_replace = ['No', 'Yes'], value = [0, 1])

```

```
In [161... # Create separate variables for each categorical value, with a 1 if the value
df = pd.get_dummies(data=df, columns=['Area', 'Marital', 'Gender', 'Contract',
```

```
In [162... # Drop columns not needed for analysis
drops = ['CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City', 'State', 'C
df = df.drop(drops, axis = 1)
```

```
In [163... df.rename(columns = {'Item1':'TimelyResponse',
                        'Item2':'Fixes',
                        'Item3':'Replacements',
                        'Item4':'Reliability',
                        'Item5':'Options',
                        'Item6':'Respectfulness',
                        'Item7':'Courteous',
                        'Item8':'Listening'},
            inplace=True)
```

```
In [164... df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 10000 entries, 0 to 9999
```

```
Data columns (total 56 columns):
```

#	Column	Non-Null Count	Dtype
0	Zip	10000 non-null	int64
1	Lat	10000 non-null	float64
2	Lng	10000 non-null	float64
3	Population	10000 non-null	int64
4	Children	10000 non-null	int64
5	Age	10000 non-null	int64
6	Income	10000 non-null	float64
7	Churn	10000 non-null	int64
8	Outage_sec_perweek	10000 non-null	float64
9	Email	10000 non-null	int64
10	Contacts	10000 non-null	int64
11	Yearly_equip_failure	10000 non-null	int64
12	Techie	10000 non-null	int64
13	Port_modem	10000 non-null	int64
14	Tablet	10000 non-null	int64
15	Phone	10000 non-null	int64
16	Multiple	10000 non-null	int64
17	OnlineSecurity	10000 non-null	int64
18	OnlineBackup	10000 non-null	int64
19	DeviceProtection	10000 non-null	int64
20	TechSupport	10000 non-null	int64
21	StreamingTV	10000 non-null	int64
22	StreamingMovies	10000 non-null	int64
23	PaperlessBilling	10000 non-null	int64
24	Tenure	10000 non-null	float64
25	MonthlyCharge	10000 non-null	float64
26	Bandwidth_GB_Year	10000 non-null	float64
27	TimelyResponse	10000 non-null	int64
28	Fixes	10000 non-null	int64
29	Replacements	10000 non-null	int64
30	Reliability	10000 non-null	int64
31	Options	10000 non-null	int64
32	Respectfulness	10000 non-null	int64

```

33 Courteous 10000 non-null int64
34 Listening 10000 non-null int64
35 Area_Rural 10000 non-null uint8
36 Area_Suburban 10000 non-null uint8
37 Area_Urban 10000 non-null uint8
38 Marital_Divorced 10000 non-null uint8
39 Marital_Married 10000 non-null uint8
40 Marital_Never Married 10000 non-null uint8
41 Marital_Separated 10000 non-null uint8
42 Marital_Widowed 10000 non-null uint8
43 Gender_Female 10000 non-null uint8
44 Gender_Male 10000 non-null uint8
45 Gender_Nonbinary 10000 non-null uint8
46 Contract_Month-to-month 10000 non-null uint8
47 Contract_One year 10000 non-null uint8
48 Contract_Two Year 10000 non-null uint8
49 PaymentMethod_Bank Transfer(automatic) 10000 non-null uint8
50 PaymentMethod_Credit Card (automatic) 10000 non-null uint8
51 PaymentMethod_Electronic Check 10000 non-null uint8
52 PaymentMethod_Mailed Check 10000 non-null uint8
53 InternetService_DSL 10000 non-null uint8
54 InternetService_Fiber Optic 10000 non-null uint8
55 InternetService_None 10000 non-null uint8
dtypes: float64(7), int64(28), uint8(21)
memory usage: 2.0 MB

```

C4. Visualizations

Univariate Visualizations

In [165...

```
df.hist(figsize=(20, 35))
```

Out[165...

```

array([[<AxesSubplot:title={'center':'Zip'}>,
        <AxesSubplot:title={'center':'Lat'}>,
        <AxesSubplot:title={'center':'Lng'}>,
        <AxesSubplot:title={'center':'Population'}>,
        <AxesSubplot:title={'center':'Children'}>,
        <AxesSubplot:title={'center':'Age'}>,
        <AxesSubplot:title={'center':'Income'}>],
       [<AxesSubplot:title={'center':'Churn'}>,
        <AxesSubplot:title={'center':'Outage_sec_perweek'}>,
        <AxesSubplot:title={'center':'Email'}>,
        <AxesSubplot:title={'center':'Contacts'}>,
        <AxesSubplot:title={'center':'Yearly_equip_failure'}>,
        <AxesSubplot:title={'center':'Techie'}>,
        <AxesSubplot:title={'center':'Port_modem'}>],
       [<AxesSubplot:title={'center':'Tablet'}>,
        <AxesSubplot:title={'center':'Phone'}>,
        <AxesSubplot:title={'center':'Multiple'}>,
        <AxesSubplot:title={'center':'OnlineSecurity'}>,
        <AxesSubplot:title={'center':'OnlineBackup'}>,
        <AxesSubplot:title={'center':'DeviceProtection'}>,
        <AxesSubplot:title={'center':'TechSupport'}>],
       [<AxesSubplot:title={'center':'StreamingTV'}>,
        <AxesSubplot:title={'center':'StreamingMovies'}>,
        <AxesSubplot:title={'center':'PaperlessBilling'}>,
        <AxesSubplot:title={'center':'Tenure'}>,
        <AxesSubplot:title={'center':'MonthlyCharge'}>,
        <AxesSubplot:title={'center':'Bandwidth_GB_Year'}>,
        <AxesSubplot:title={'center':'TimelyResponse'}>],
       [<AxesSubplot:title={'center':'Fixes'}>,
        <AxesSubplot:title={'center':'Replacements'}>],

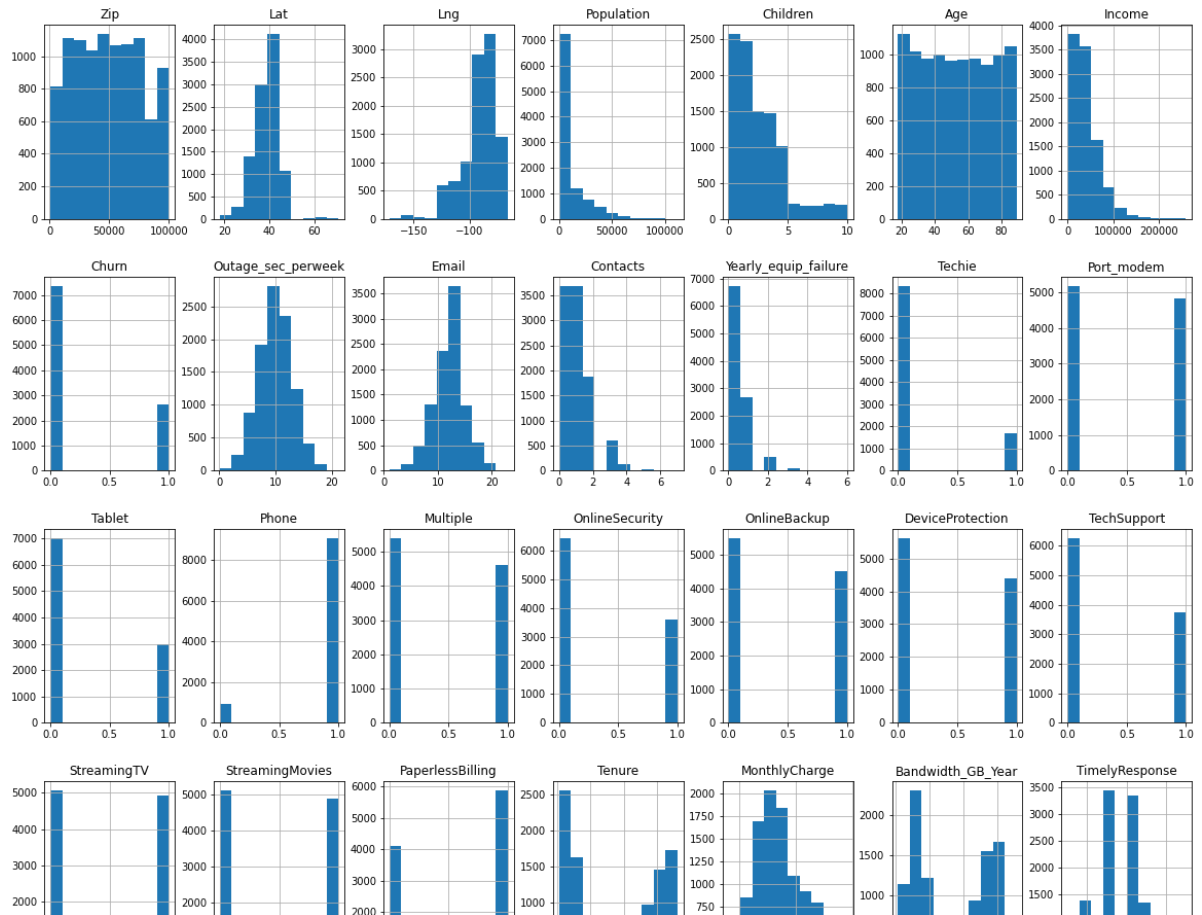
```



```

<AxesSubplot:title={'center':'Reliability'}>,
<AxesSubplot:title={'center':'Options'}>,
<AxesSubplot:title={'center':'Respectfulness'}>,
<AxesSubplot:title={'center':'Courteous'}>,
<AxesSubplot:title={'center':'Listening'}>]],
[<AxesSubplot:title={'center':'Area_Rural'}>,
<AxesSubplot:title={'center':'Area_Suburban'}>,
<AxesSubplot:title={'center':'Area_Urban'}>,
<AxesSubplot:title={'center':'Marital_Divorced'}>,
<AxesSubplot:title={'center':'Marital_Married'}>,
<AxesSubplot:title={'center':'Marital_Never Married'}>,
<AxesSubplot:title={'center':'Marital_Separated'}>]],
[<AxesSubplot:title={'center':'Marital_Widowed'}>,
<AxesSubplot:title={'center':'Gender_Female'}>,
<AxesSubplot:title={'center':'Gender_Male'}>,
<AxesSubplot:title={'center':'Gender_Nonbinary'}>,
<AxesSubplot:title={'center':'Contract_Month-to-month'}>,
<AxesSubplot:title={'center':'Contract_One year'}>,
<AxesSubplot:title={'center':'Contract_Two Year'}>],
[<AxesSubplot:title={'center':'PaymentMethod_Bank Transfer(automatic)'}
>,
    <AxesSubplot:title={'center':'PaymentMethod_Credit Card (automatic)'}
>,
    <AxesSubplot:title={'center':'PaymentMethod_Electronic Check'}>,
    <AxesSubplot:title={'center':'PaymentMethod_Mailed Check'}>,
    <AxesSubplot:title={'center':'InternetService_DSL'}>,
    <AxesSubplot:title={'center':'InternetService_Fiber Optic'}>,
    <AxesSubplot:title={'center':'InternetService_None'}>]]],

```

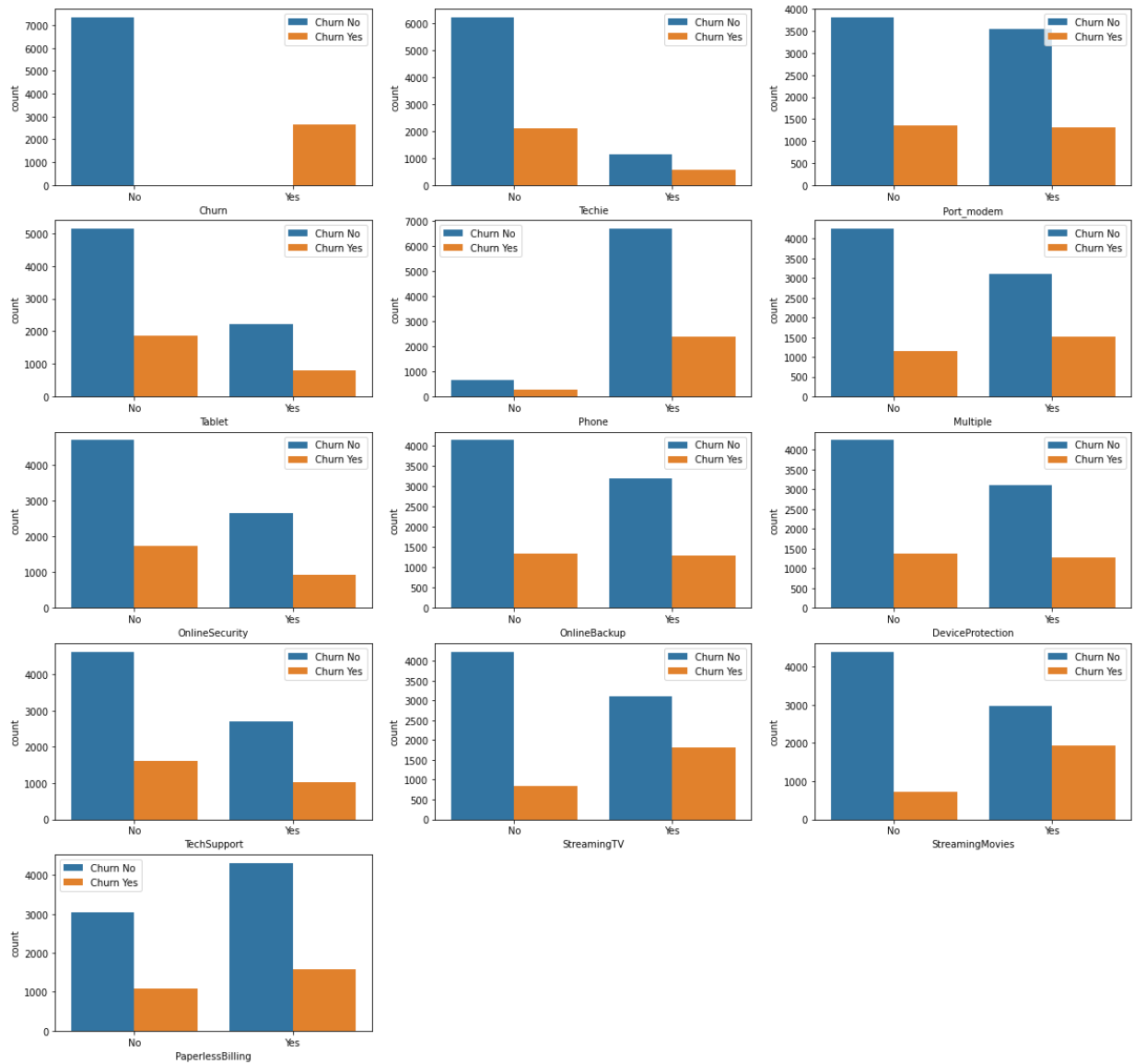


Bivariate Visualizations

In [166...

```
# Split bar charts for binary variables by Churn
df_bin = df[['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone', 'Multiple',
              'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
              'StreamingMovies', 'PaperlessBilling', 'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'TimelyResponse']]

count=1
plt.subplots(figsize=(20, 80))
for i in df_bin.columns:
    plt.subplot(20,3,count)
    ax = sns.countplot(x = i, hue = 'Churn', data = df_bin)
    legend_labels, _ = ax.get_legend_handles_labels()
    ax.legend(legend_labels, ['Churn No', 'Churn Yes'])
    ax.set_xticklabels(('No', 'Yes'))
    count+=1
plt.show();
```

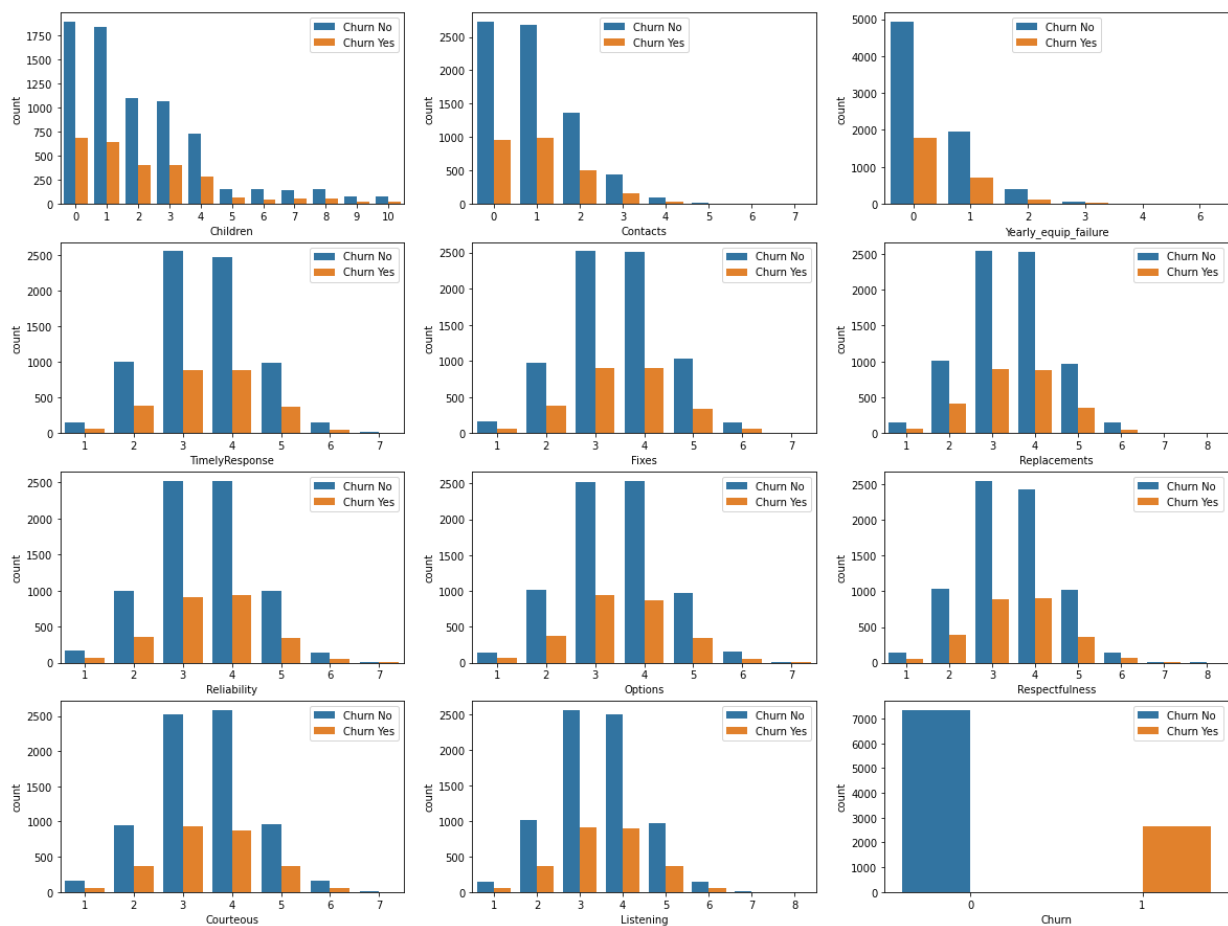


In [167...

```
# Display plots for all discrete value variables by Churn

# Split bar charts for binary variables by Churn
df_dis = df[['Children', 'Contacts', 'Yearly_equip_failure', 'TimelyResponse',
             'Reliability', 'Options', 'Respectfulness', 'Courteous', 'Listen:
count=1
plt.subplots(figsize=(20, 80))
for i in df_dis.columns:
    plt.subplot(20,3,count)
    ax = sns.countplot(x = i, hue = 'Churn', data = df_dis)
    legend_labels, _ = ax.get_legend_handles_labels()
    ax.legend(legend_labels, ['Churn No','Churn Yes'])

    count+=1
plt.show();
```



In [168...

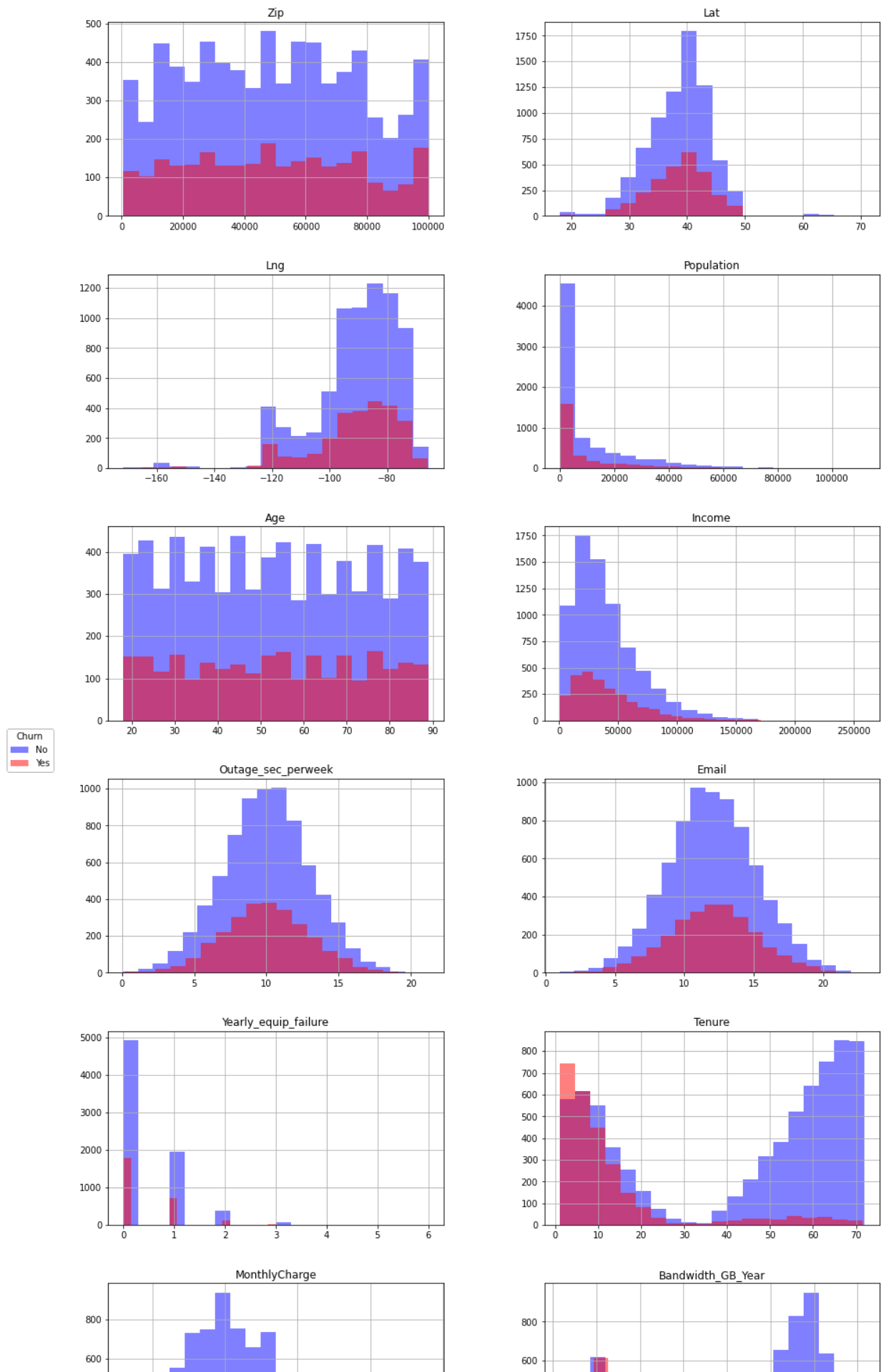
```
# Display histograms for continuous variables by Churn

# Array of columns to use
numerical_features = ['Zip', 'Lat', 'Lng', 'Population', 'Age', 'Income', 'Out
                      'Yearly equip. failure', 'Tenure', 'MonthlyCharge', 'Band

# Labels for chart
sub_labels = ["No", "Yes"]

# Split hisogram by churn
fig, ax = plt.subplots(6, 2, figsize=(15, 30))
no = df[df.Churn == 0][numerical_features].hist(bins=20, color="blue", alpha=0.5)
yes = df[df.Churn == 1][numerical_features].hist(bins=20, color="red", alpha=0.5)

# Create labels and ledgend
fig.legend([no, yes], labels=sub_labels, loc="center left", borderaxespad=0.1)
plt.subplots_adjust(left=0.1)
plt.show();
```

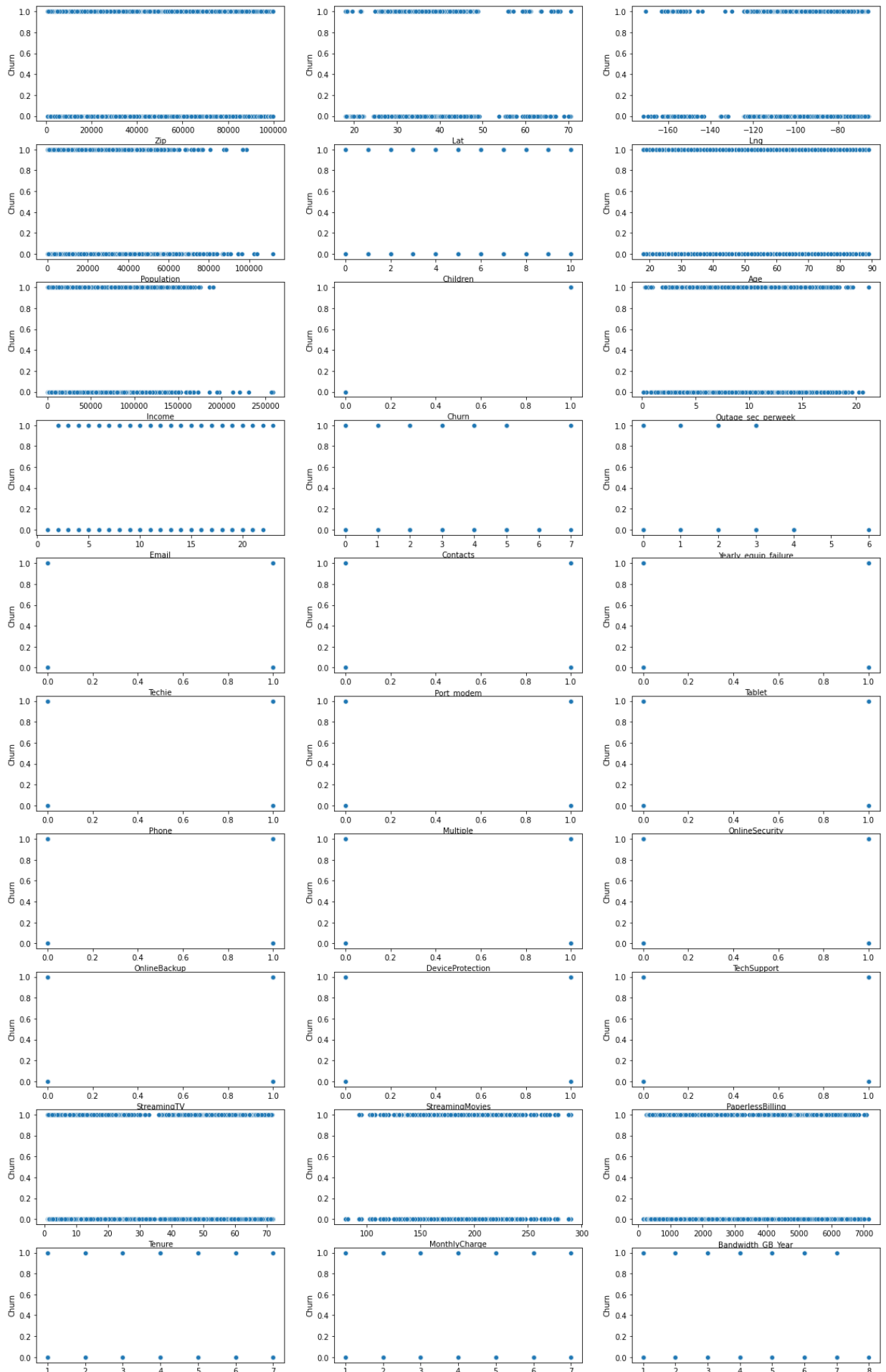


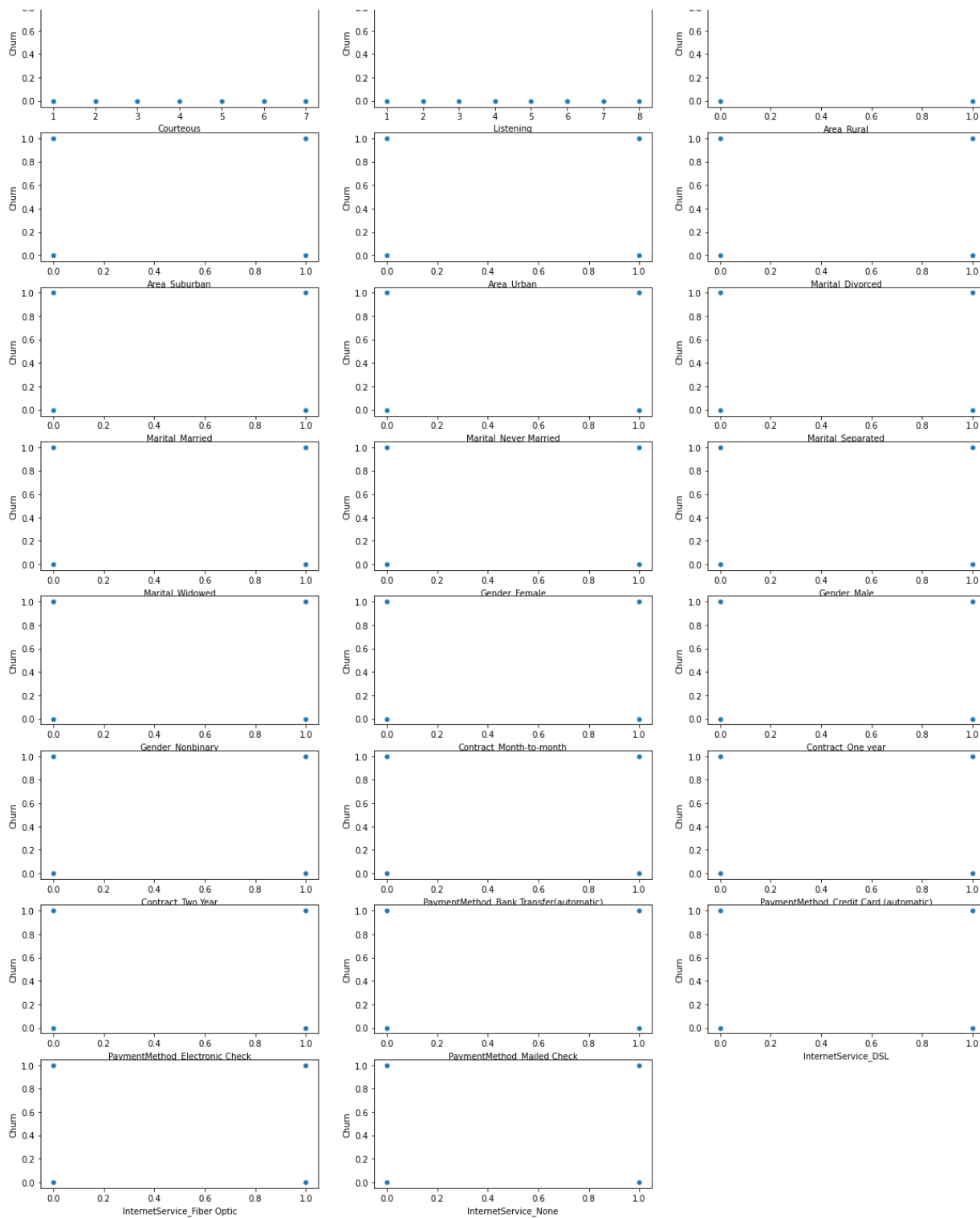
Bivariate Statistics

In [169...

```
count=1
plt.subplots(figsize=(20, 80))
for i in df.columns:
    plt.subplot(24,3,count)
    sns.scatterplot(df[i], df["Churn"])
    count+=1

plt.show()
```





Prepared Data


```
In [170... # Save cleaned dataframe to CSV
df.to_csv('churn_clean_data_final.csv', index = False, encoding = 'utf-8')
```

D1. Initial Model

```
In [171... # Set up input matrix and response variable
Xinit = df.drop('Churn', axis = 1)
y = df[['Churn']].values
```

```
In [172... y.shape
```

```
Out[172... (10000, 1)
```

```
In [173... # Veiw number of independent variables
print ("There are", Xinit.shape[1], "independent variables in the initial model")
```

There are 55 independent variables in the initial model.

```
In [174... df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 56 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   Zip                                       10000 non-null  int64
1   Lat                                       10000 non-null  float64
2   Lng                                       10000 non-null  float64
3   Population                               10000 non-null  int64
4   Children                                 10000 non-null  int64
5   Age                                       10000 non-null  int64
6   Income                                   10000 non-null  float64
7   Churn                                    10000 non-null  int64
8   Outage_sec_perweek                       10000 non-null  float64
9   Email                                    10000 non-null  int64
10  Contacts                                 10000 non-null  int64
11  Yearly_equip_failure                     10000 non-null  int64
12  Techie                                   10000 non-null  int64
13  Port_modem                               10000 non-null  int64
14  Tablet                                   10000 non-null  int64
15  Phone                                    10000 non-null  int64
16  Multiple                                 10000 non-null  int64
17  OnlineSecurity                           10000 non-null  int64
18  OnlineBackup                             10000 non-null  int64
19  DeviceProtection                         10000 non-null  int64
20  TechSupport                              10000 non-null  int64
21  StreamingTV                              10000 non-null  int64
22  StreamingMovies                          10000 non-null  int64
23  PaperlessBilling                         10000 non-null  int64
24  Tenure                                   10000 non-null  float64
25  MonthlyCharge                             10000 non-null  float64
26  Bandwidth_GB_Year                       10000 non-null  float64
27  TimelyResponse                           10000 non-null  int64
28  Fixes                                    10000 non-null  int64
29  Replacements                             10000 non-null  int64
```

```

30 Reliability 10000 non-null int64
31 Options 10000 non-null int64
32 Respectfulness 10000 non-null int64
33 Courteous 10000 non-null int64
34 Listening 10000 non-null int64
35 Area_Rural 10000 non-null uint8
36 Area_Suburban 10000 non-null uint8
37 Area_Urban 10000 non-null uint8
38 Marital_Divorced 10000 non-null uint8
39 Marital_Married 10000 non-null uint8
40 Marital_Never Married 10000 non-null uint8
41 Marital_Separated 10000 non-null uint8
42 Marital_Widowed 10000 non-null uint8
43 Gender_Female 10000 non-null uint8
44 Gender_Male 10000 non-null uint8
45 Gender_Nonbinary 10000 non-null uint8
46 Contract_Month-to-month 10000 non-null uint8
47 Contract_One year 10000 non-null uint8
48 Contract_Two Year 10000 non-null uint8
49 PaymentMethod_Bank Transfer(automatic) 10000 non-null uint8
50 PaymentMethod_Credit Card (automatic) 10000 non-null uint8
51 PaymentMethod_Electronic Check 10000 non-null uint8
52 PaymentMethod_Mailed Check 10000 non-null uint8
53 InternetService_DSL 10000 non-null uint8
54 InternetService_Fiber Optic 10000 non-null uint8
55 InternetService_None 10000 non-null uint8
dtypes: float64(7), int64(28), uint8(21)
memory usage: 2.9 MB

```

In [175...

```

# Add y-intercept, create model, and view summary
Xcinit = sm.add_constant(Xinit)
logistic_regression = sm.Logit(y,Xcinit)
fitted_model1 = logistic_regression.fit()
fitted_model1.summary()

```

Optimization terminated successfully.
Current function value: 0.217085
Iterations 9

Out[175...

Logit Regression Results

Dep. Variable:	y	No. Observations:	10000
Model:	Logit	Df Residuals:	9949
Method:	MLE	Df Model:	50
Date:	Sat, 11 Sep 2021	Pseudo R-squ.:	0.6246
Time:	16:05:42	Log-Likelihood:	-2170.9
converged:	True	LL-Null:	-5782.2
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P> z	[0.025	0.975]
const	-3.0031	nan	nan	nan	nan	nan
Zip	3.632e-07	3.35e-06	0.109	0.914	-6.2e-06	6.92e-06
Lat	0.0035	0.008	0.461	0.645	-0.011	0.018
Lng	-0.0013	0.006	-0.211	0.833	-0.013	0.011

Population	-1.473e-07	2.8e-06	-0.053	0.958	-5.64e-06	5.35e-06
Children	-0.0055	0.137	-0.040	0.968	-0.274	0.263
Age	0.0028	0.015	0.194	0.846	-0.026	0.032
Income	3.996e-07	1.38e-06	0.290	0.772	-2.3e-06	3.1e-06
Outage_sec_perweek	-0.0027	0.013	-0.207	0.836	-0.028	0.023
Email	-0.0094	0.013	-0.739	0.460	-0.034	0.016
Contacts	0.0626	0.039	1.603	0.109	-0.014	0.139
Yearly equip_failure	-0.0349	0.061	-0.570	0.569	-0.155	0.085
Techie	1.0981	0.103	10.673	0.000	0.896	1.300
Port_modem	0.1432	0.077	1.849	0.064	-0.009	0.295
Tablet	-0.0509	0.085	-0.602	0.547	-0.217	0.115
Phone	-0.2988	0.133	-2.246	0.025	-0.560	-0.038
Multiple	0.3602	0.202	1.787	0.074	-0.035	0.755
OnlineSecurity	-0.2935	0.312	-0.942	0.346	-0.905	0.317
OnlineBackup	-0.1173	0.181	-0.649	0.516	-0.471	0.237
DeviceProtection	-0.1085	0.234	-0.465	0.642	-0.566	0.349
TechSupport	-0.2144	0.173	-1.237	0.216	-0.554	0.125
StreamingTV	1.1137	0.510	2.184	0.029	0.114	2.113
StreamingMovies	1.2743	0.363	3.506	0.000	0.562	1.987
PaperlessBilling	0.1659	0.079	2.105	0.035	0.011	0.320
Tenure	-0.1687	0.363	-0.465	0.642	-0.880	0.542
MonthlyCharge	0.0394	0.014	2.850	0.004	0.012	0.067
Bandwidth_GB_Year	0.0006	0.004	0.143	0.886	-0.008	0.009
TimelyResponse	-0.0189	0.055	-0.346	0.730	-0.126	0.088
Fixes	-0.0041	0.052	-0.078	0.938	-0.106	0.098
Replacements	0.0206	0.047	0.438	0.662	-0.072	0.113
Reliability	-0.0301	0.042	-0.716	0.474	-0.113	0.052
Options	-0.0362	0.044	-0.816	0.415	-0.123	0.051
Respectfulness	-0.0193	0.045	-0.427	0.670	-0.108	0.069
Courteous	-0.0025	0.043	-0.058	0.954	-0.087	0.082
Listening	-0.0140	0.040	-0.347	0.729	-0.093	0.065
Area_Rural	-1.3840	1.92e+06	-7.2e-07	1.000	-3.77e+06	3.77e+06
Area_Suburban	-1.4321	1.87e+06	-7.65e-07	1.000	-3.67e+06	3.67e+06
Area_Urban	-1.3326	1.89e+06	-7.03e-07	1.000	-3.71e+06	3.71e+06

Marital_Divorced	-0.3846	5.46e+05	-7.05e-07	1.000	-1.07e+06	1.07e+06
Marital_Married	-0.2796	5.51e+05	-5.07e-07	1.000	-1.08e+06	1.08e+06
Marital_Never Married	-0.3707	5.46e+05	-6.79e-07	1.000	-1.07e+06	1.07e+06
Marital_Separated	-0.2662	5.48e+05	-4.86e-07	1.000	-1.07e+06	1.07e+06
Marital_Widowed	-0.1241	5.36e+05	-2.32e-07	1.000	-1.05e+06	1.05e+06
Gender_Female	-1.1031	1.16e+06	-9.53e-07	1.000	-2.27e+06	2.27e+06
Gender_Male	-0.8757	1.03e+06	-8.47e-07	1.000	-2.03e+06	2.03e+06
Gender_Nonbinary	-1.1811	1.05e+06	-1.13e-06	1.000	-2.05e+06	2.05e+06
Contract_Month-to-month	1.2958	8.75e+05	1.48e-06	1.000	-1.72e+06	1.72e+06
Contract_One year	-2.1189	8.75e+05	-2.42e-06	1.000	-1.72e+06	1.72e+06
Contract_Two Year	-2.2223	8.76e+05	-2.54e-06	1.000	-1.72e+06	1.72e+06
PaymentMethod_Bank Transfer(automatic)	-0.6822	2.01e+06	-3.39e-07	1.000	-3.94e+06	3.94e+06
PaymentMethod_Credit Card (automatic)	-0.4729	2.01e+06	-2.35e-07	1.000	-3.94e+06	3.94e+06
PaymentMethod_Electronic Check	-0.0514	2.01e+06	-2.56e-08	1.000	-3.94e+06	3.94e+06
PaymentMethod_Mailed Check	-0.4430	2.01e+06	-2.2e-07	1.000	-3.94e+06	3.94e+06
InternetService_DSL	-0.0934	nan	nan	nan	nan	nan
InternetService_Fiber Optic	-2.0076	nan	nan	nan	nan	nan

In [176...

```
# View prediction
clf = LogisticRegression()
clf.fit(Xinit, y.astype(int))
y_clf = clf.predict(Xinit)
print(classification_report(y, y_clf))
```

	precision	recall	f1-score	support
0	0.86	0.91	0.89	7350
1	0.71	0.59	0.64	2650
accuracy			0.83	10000
macro avg	0.78	0.75	0.76	10000
weighted avg	0.82	0.83	0.82	10000

D2 Model Reduction

The model has several variables, with p-values over 0.5, that may be causing inaccurate predictions. These include variables are irrelevant to whether a customer is likely to churn and should be removed. The variables need to be checked for relevance. We also need to check these variables for multicollinearity whihc may be affecting the results.

In [177...

```
# Use recursive feature elimination to choose most important features (ref 6)
model = LogisticRegression()
rfe = RFE(model, 11)
rfe = rfe.fit(Xinit, y)
print(rfe.support_)
print(rfe.ranking_)
f = rfe.get_support(1) # the most important features
Xfin = Xinit[Xinit.columns[f]] # final features`
```

```
[False False False False False False False False False False False  True
 False False False False False  True  True False  True  True False False
 False False False False False False False False False False False False
 False False False False False False  True False False  True  True  True
 False False  True False  True False False]
[43 24 35 44 34 36 45 21 42 33 17  1 41 10  4 32  8  1  1  3  1  1  9 14
 28 39 29 37 27 19 20 23 22 18 16 15 40  7  6  5 38 26  1 31  2  1  1  1
 11 12  1 13  1 25 30]
```

In [178...

```
# Look for evidence of Variance Inflation Factors (ref 7) causing multicollinearity

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["feature"] = Xfin.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(Xfin.values, i)
                   for i in range(len(Xfin.columns))]

print(vif_data)
```

	feature	VIF
0	Techie	1.000742
1	OnlineBackup	1.000229
2	DeviceProtection	1.001364
3	StreamingTV	1.001122
4	StreamingMovies	1.000657
5	Gender_Female	1.001250
6	Contract_Month-to-month	4.198087
7	Contract_One year	2.258283
8	Contract_Two Year	2.411442
9	PaymentMethod_Electronic Check	1.000559
10	InternetService_DSL	1.000234

All variable VIF scores are below the recommended score of 5. There is no evidence for multicollinearity. However, even though it is below 5, contract_month_to_month is showing signs of multicollinearity. We will drop it from the model.

In [179...

```
Xfin = Xfin.drop(columns='Contract_Month-to-month', axis=1)
```

In [180...

```
# Re-run the model
Xcfin = sm.add_constant(Xfin)
logistic_regression = sm.Logit(y, Xcfin)
fitted_model2 = logistic_regression.fit()
fitted_model2.summary()
```

Optimization terminated successfully.

Current function value: 0.446443

Out[180...

Logit Regression Results

Dep. Variable:	y	No. Observations:	10000
Model:	Logit	Df Residuals:	9989
Method:	MLE	Df Model:	10
Date:	Sat, 11 Sep 2021	Pseudo R-squ.:	0.2279
Time:	16:05:48	Log-Likelihood:	-4464.4
converged:	True	LL-Null:	-5782.2
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P> z	[0.025	0.975]
const	-2.7468	0.082	-33.460	0.000	-2.908	-2.586
Techie	0.5077	0.067	7.527	0.000	0.376	0.640
OnlineBackup	0.3132	0.053	5.939	0.000	0.210	0.417
DeviceProtection	0.3043	0.053	5.767	0.000	0.201	0.408
StreamingTV	1.4226	0.056	25.555	0.000	1.313	1.532
StreamingMovies	1.7039	0.057	30.034	0.000	1.593	1.815
Gender_Female	-0.1414	0.053	-2.689	0.007	-0.244	-0.038
Contract_One year	-1.5911	0.076	-20.990	0.000	-1.740	-1.443
Contract_Two Year	-1.7426	0.075	-23.350	0.000	-1.889	-1.596
PaymentMethod_Electronic Check	0.1711	0.055	3.104	0.002	0.063	0.279
InternetService_DSL	0.5931	0.054	10.882	0.000	0.486	0.700

In [181...

```
# Split the data to be used in final model evaluation
X_train, X_test, y_train, y_test = train_test_split(Xfin, y.astype(float), test_size=0.2)
```

In [182...

```
Xcfin = sm.add_constant(X_train)
logistic_regression = sm.Logit(y_train, Xcfin)
fitted_model2 = logistic_regression.fit()
fitted_model2.summary()
```

Optimization terminated successfully.

Current function value: 0.439120

Iterations 7

Out[182...

Logit Regression Results

Dep. Variable:	y	No. Observations:	7000
Model:	Logit	Df Residuals:	6989
Method:	MLE	Df Model:	10
Date:	Sat, 11 Sep 2021	Pseudo R-squ.:	0.2303

Time:	16:05:48	Log-Likelihood:	-3073.8
converged:	True	LL-Null:	-3993.5
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P> z	[0.025	0.975]
const	-2.7750	0.099	-27.973	0.000	-2.969	-2.581
Techie	0.5209	0.082	6.334	0.000	0.360	0.682
OnlineBackup	0.2617	0.064	4.113	0.000	0.137	0.386
DeviceProtection	0.2892	0.064	4.539	0.000	0.164	0.414
StreamingTV	1.4706	0.068	21.687	0.000	1.338	1.604
StreamingMovies	1.7229	0.068	25.170	0.000	1.589	1.857
Gender_Female	-0.1577	0.064	-2.483	0.013	-0.282	-0.033
Contract_One year	-1.5752	0.092	-17.196	0.000	-1.755	-1.396
Contract_Two Year	-1.7606	0.090	-19.571	0.000	-1.937	-1.584
PaymentMethod_Electronic Check	0.1513	0.066	2.281	0.023	0.021	0.281
PaymentMethod_Cash	0.5048	0.066	7.615	0.000	0.366	0.724

In [183...

```
# View prediction
clf = LogisticRegression()
clf.fit(X_train, y_train.astype(int))
y_clf = clf.predict(X_test)
print(classification_report(y_test, y_clf))
```

	precision	recall	f1-score	support
0.0	0.79	0.92	0.85	2153
1.0	0.66	0.37	0.47	847
accuracy			0.77	3000
macro avg	0.72	0.65	0.66	3000
weighted avg	0.75	0.77	0.74	3000

In [184...

```
print ("There are", Xfin.shape[1], "independent variables in the final model.")
```

There are 10 independent variables in the final model.

E1. Model Comparison

Using recursive feature elimination, I identified the first 7 variables. This method uses model accuracy to identify which combination of attributes most contribute to predicting the target variable (Brownlee, 2020).

To check for multicollinearity, I used variance inflation factors. Given that the VIF scores for the reduced 7 variables were less than 4, this suggests that

Finally taking a look at the reduced model, I chose to remove all remaining variables which had p-scores above the 0.05 threshold. This final step ensured that all remaining independent variables were significant. I began with 55 independent variables, which was then reduced to six remaining variables.

My initial model had better scores for the target descriptive statistics than the reduced model. The initial model would predict churn correctly 83% of the time, while the reduced model was 77%. The initial model was likely to catch actual cases of churn 59% of the time, while the reduced model was only 38% of the time. The combined F1 score was also higher on the initial model at 64% compared to 47% on the reduced model.

In [185...

```
# View prediction (initial)
clf = LogisticRegression()
clf.fit(Xinit, y.astype(int))
y_clf = clf.predict(Xinit)
print(classification_report(y, y_clf))
```

	precision	recall	f1-score	support
0	0.86	0.91	0.89	7350
1	0.71	0.59	0.64	2650
accuracy			0.83	10000
macro avg	0.78	0.75	0.76	10000
weighted avg	0.82	0.83	0.82	10000

In [186...

```
# View prediction (reduced)
clf = LogisticRegression()
clf.fit(X_train, y_train.astype(int))
y_clf = clf.predict(X_test)
print(classification_report(y_test, y_clf))
```

	precision	recall	f1-score	support
0.0	0.79	0.92	0.85	2153
1.0	0.66	0.37	0.47	847
accuracy			0.77	3000
macro avg	0.72	0.65	0.66	3000
weighted avg	0.75	0.77	0.74	3000

E3. Code

See above for code

F1. Results

The equation of the final regression model is: $y =$

$-2.76 + 0.5209(\text{Techie}) + 0.2617(\text{OnlineBackup}) + 0.2892(\text{DeviceProtection}) + 1.4706(\text{StreamingTV}) + 1.7229$
 $0.1577(\text{Gender_Female}) - 1.5752(\text{Contract_One year}) - 1.7606(\text{Contract_Two Year}) -$
 $0.1513(\text{PaymentMethod_Electronic Check}) + 0.5948(\text{InternetService_DSL}).$

In this equation, -2.76 is the y intercept value on the y-axis. Each coefficient(slope) of the predictor variables describes how much the target variable is estimated to change if all other variables remain constant (Massaron). If DeviceProtection increased by one unit, the mean value of Churn would increase by 0.2892. The Contract_Two Year and streaming movies seems to have the largest effect on churn negatively by 1.76 and postively by 1.722 respectively.

P-values for for most variables are statistically significant at 0.000, with low multicollinearity (McKinney), as evidenced by the the low Variance Inflation Factor scores under 5. Confusion matrix suggests that overall accuracy of the model stays high, even when cross validated using test train split, with an only a small drop of 0.05 from 0.83 to 0.77 from the original model to the trained model.

Data limitations include the data set is small (only 10,000 rows) and that it is not adjusted for seasonality. Seasonality could be affecting times of the year that customers churn more frequently. For example, we could have higher churn during the holidays, since customers would be interested in trying new services. We should also identify if ther are seasons of lower churn during different times of the year.

F2 Recommendations

Structurally, the model is good enough to deploy to a production environment for predictive analytics to predict churn. However, before this can be done, it would be important to label/identify when the customer exactly churned. Seasonality is important for retail businesses like telecommunications companies, and identifying periods of time that a customer is likely to churn is just as important as knowing if they will churn. Without the date of when the customer churned, we have no context of when they churned. If the model is adjusted for seasonality, or is run individually for certain seasons, the model would generate an accurate result. These adjustments would be useful for creating a model that departments such as retention, marketing, and customer service could use.

Panapto Recording

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=137cb112-6660-48e2-960b-ad7e0114e72b>

Sources

1. Bhandari, Aniruddha. "What Is Multicollinearity? Here's Everything You Need to Know." Analytics Vidhya, 16 Apr. 2020, www.analyticsvidhya.com/blog/2020/03/what-is-multicollinearity/.
2. Kautumn06. "Yellowbrick — Regression Visualizer Examples." Kaggle, 18 Aug. 2018, www.kaggle.com/kautumn06/yellowbrick-regression-visualizer-examples.
3. Massaron, Luca, and Alberto Boschetti. Regression Analysis with Python. Packt Publishing,

2016.

4. McKinney, W. (2018). *Python for Data Analysis*. O'Reilly.
5. Sree. (2020, October 26). *Predict Customer Churn in Python*. Towards Data Science. <https://towardsdatascience.com/predict-customer-churn-in-python-e8cd6d3aaa7>
6. Walker, Michael. *Python Data Cleaning Cookbook: Modern Techniques and Python Tools to Detect and Remove Dirty Data and Extract Key Insights*. Packt Publishing, 2020.

Third Party Code

1. <https://stackoverflow.com/questions/51672709/convert-no-and-yes-into-0-and-1-in-pandas-dataframe/51672855>
2. Farrell, Peter, et al. *The Statistics and Calculus with Python Workshop: A Comprehensive Introduction to Mathematics in Python for Artificial Intelligence Applications*. Packt Publishing, 2020.
3. <https://analyticsvidhya.com/blog/2020/03/what-is-multicollinearity/>
4. <https://stackoverflow.com/questions/57924484/finding-coefficients-for-logistic-regression-in-python>

In []:

In []: