Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2021, 8(12):95-101

Research Article ISSN: 2394 - 658X

Multiple Regression Analysis for Predictive Modeling

Kailash Alle

Sr. Software Engineer, Comscore Inc
kailashalle@gmail.com

ABSTRACT

This study explores predicting customer data consumption in the telecommunications industry using multi-linear regression. Reducing customer churn, or customer loss, is crucial for telecom companies. Here, we propose a method to forecast a customer's annual data usage based on various factors. The approach utilizes multi-linear regression, a statistical technique that analyzes the linear relationship between a dependent variable (data usage) and multiple independent variables (customer characteristics). We'll use Python for its user-friendly data science tools. The data preparation process involves loading customer information from a dataset. This data will include demographics, service plans, survey responses on customer service satisfaction, and most importantly, annual data usage. We'll clean the data by addressing missing values, outliers, and encoding categorical variables (e.g., yes/no answers) into numerical values. By analyzing this data, we can identify factors that influence data consumption. The resulting model will predict a customer's data usage based on these factors, enabling telecom companies to better understand customer needs and potentially develop targeted data plans to reduce churn.

Keywords: Telecoms, Big Data, Customer churn, Data usage forecasting, Machine learning, Multi-linear regression.

INTRODUCTION

In today's data-driven world, telecommunications companies face a constant challenge: keeping up with skyrocketing customer data usage while minimizing customer churn, the dreaded phenomenon of customers switching to competitors. This study investigates a novel approach to this challenge. By leveraging the power of multi-linear regression, a statistical technique, researchers aim to predict a customer's annual data consumption with surprising accuracy. Imagine a world where customer data plans are not one-size-fits-all, but tailored to individual needs based on factors like demographics, service plans, and even customer satisfaction with service. This research explores the feasibility of such a future, potentially revolutionizing the way telecommunications companies manage customer data and satisfaction.

PURPOSE

This study aims to develop a method for predicting a customer's annual data consumption in the telecommunications industry. By leveraging multi-linear regression, a statistical technique, the research seeks to identify factors influencing data usage. These factors could include demographics, service plan details, and even customer satisfaction surveys. The ultimate goal is to create a model that predicts data usage with reasonable accuracy. This would allow telecommunications companies to develop targeted data plans that meet individual customer needs, potentially reducing customer churn and increasing overall satisfaction.

LIMITATIONS

Despite its potential benefits, this study has limitations to consider. Multi-linear regression assumes a linear relationship between variables. While data usage may correlate with factors like demographics and service plans, the relationship might not always be perfectly linear. Additionally, the accuracy of the model heavily relies on the quality of the data used. If the data is incomplete or inaccurate, the resulting model's predictions will be unreliable. Finally, this study focuses on historical data, and customer behavior can change over time. The model may require periodic updates to maintain its effectiveness as customer needs and data usage patterns evolve.

METHODOLOGY

The methodology employed in this study involves utilizing a dataset containing information on 1,000 customers. This data includes details such as demographics (age, income, number of children), service plan specifics (internet type, phone service), and even customer feedback on service aspects. The target variable for prediction is the annual data consumption measured in gigabytes (GB) per year. Categorical variables, like "Yes" or "No" service options, will be encoded as numerical values (e.g., 1 or 0) for analysis. Prior to applying the multi-linear regression model, the data will undergo cleaning procedures. This includes checking for missing values and outliers, and transforming categorical variables into numerical representations. By analyzing these factors and their relationship to data usage through multi-linear regression, the study aims to develop a model that can predict data consumption for new customers.

Why Multi-Linear Regression is Suitable for this Analysis

This study utilizes multi-linear regression, a statistical technique, to predict a customer's annual data consumption in the telecommunications industry. Here's a breakdown of why this method is a good fit for this particular analysis:

- Linear Relationships: Multi-linear regression thrives on linear relationships between variables. In our case, the target variable is annual data consumption (in gigabytes), and we suspect factors like demographics, service plan details, and customer satisfaction might influence this value. While the relationship may not be perfectly linear (e.g., a customer with a large family might not necessarily use double the data of a single person), multi-linear regression can still capture the general trend and quantify the strength of these associations. Scatterplots will be a valuable tool during data exploration to visually assess if these relationships appear linear.
- Multiple Explanatory Variables: Unlike simpler linear regression which deals with only one independent variable, multi-linear regression is well-suited for scenarios with multiple factors potentially affecting the outcome. Here, we're not just looking at a single variable like age to predict data usage. Factors like income, number of children, internet service type (DSL, fiber optic), and even ratings for customer service aspects might all play a role. Multi-linear regression allows us to analyze the combined effect of these variables on data consumption.
- Understanding Variable Importance: Through multi-linear regression, we can not only determine if a variable has a statistically significant impact on data consumption, but also assess the strength and direction of that influence. For instance, the model might reveal that income has a positive correlation with data usage, meaning customers with higher income tend to use more data. Similarly, a negative correlation might be found between customer satisfaction and data usage, suggesting happier customers might use less data on average. This information is crucial for telecommunication companies to understand which factors truly drive data consumption.
- Data Cleaning Considerations: It's important to acknowledge that multi-linear regression makes certain assumptions about the data. One key assumption is that the residuals, the difference between predicted and actual data consumption values, are normally distributed. Additionally, the independent variables should not be highly correlated with each other (multicollinearity). These assumptions will be carefully examined during data cleaning and pre-processing steps. Techniques like outlier removal and data transformation might be necessary to ensure the data adheres to the assumptions of the model.

In conclusion, multi-linear regression provides a solid foundation for this study due to its ability to handle multiple explanatory variables, quantify the strength of linear relationships, and identify which factors have the most significant influence on a customer's annual data consumption. By acknowledging the underlying assumptions and limitations of the model, this technique can be a valuable tool for telecommunications companies seeking to predict customer data usage and develop more targeted data plans.

Why Python is the Perfect Tool for This Analysis

This study leverages the power of Python for its extensive data science and machine learning capabilities. Python's user-friendly nature and versatility make it a perfect choice for this analysis. Here's a breakdown of the key advantages Python offers:

- Straightforward Language: Compared to other data science languages like R or MATLAB, Python boasts a clear and adaptable syntax. This ease of use makes it a favorite among researchers, allowing them to focus on the analysis itself rather than struggling with complex code structures.
- Speed and Efficiency: Python is renowned for its speed and efficiency in data processing. This is crucial for handling large datasets, as calculations are completed quickly, allowing researchers to iterate and refine their analysis promptly.
- Powerful Libraries: Beyond the core Python language, a wealth of libraries specifically designed for data science tasks are readily available. These libraries provide powerful tools to streamline the analysis process.

- Pandas for Data Manipulation: Pandas is a go-to library for data wrangling. It allows researchers to load datasets, clean and organize data, and efficiently create new variables from existing ones.
- Matplotlib for Visualization: Matplotlib offers a comprehensive set of tools to create informative charts and graphs that effectively communicate the findings of the analysis.
- Scikit-learn for Machine Learning: Scikit-learn plays a central role in building and implementing the multi-linear regression model, the heart of this study.
- Seaborn for Enhanced Visualizations: Seaborn provides a user-friendly interface on top of Matplotlib. This allows researchers to create aesthetically pleasing and informative data visualizations that enhance the clarity and impact of the analysis.

In conclusion, Python's user-friendly syntax, speed, and powerful data science libraries make it the ideal platform for this analysis of customer data consumption in the telecommunications industry. By leveraging these tools, researchers can effectively explore the data, build the multi-linear regression model, and ultimately gain valuable insights into customer data usage patterns.

Unveiling The Data

This section delves into the data strategy for this study. The information will be collected using Python's Pandas library, which excels at data manipulation. The data will be loaded into a Pandas dataframe named "churn_df" for easier manipulation and exploration. Data quality is paramount, so the initial steps involve examining the data structure and identifying any inconsistencies like misspellings or strange variable names. Missing data points will be scrutinized. Techniques like calculating central tendency measures (mean, median, or mode) might be used to impute missing values where appropriate. Alternatively, outliers, which are data points that fall far outside the expected range, might need removal, especially if they are several standard deviations above the mean. The target variable of interest is "Bandwidth GB Year," representing the annual data consumption per customer. This variable plays a crucial role in the decision-making process, as the goal is to predict data usage for future customers. Beyond the target variable, the dataset contains a wealth of potential explanatory variables. These include categorical variables like whether a customer churned (stopped using the service) or has a phone line, as well as ordinal variables derived from customer surveys on various service aspects (e.g., courteous exchange, timely response). By analyzing these variables and their relationship to data consumption, the study aims to develop a model that can predict a customer's annual data usage.

What Our Numbers Reveal

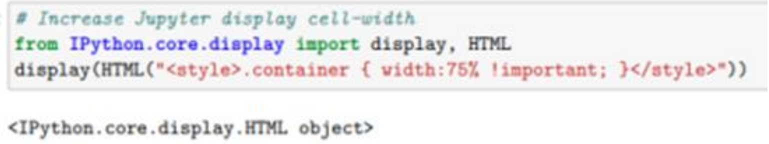
This section summarizes the key characteristics of the dataset used in this study. The data consists of 1,000 records with 50 original columns. However, certain identifying information (customer ID, address details) and seemingly irrelevant categorical variables (marital status) were excluded from the analysis. Additionally, binary variables like "Yes" or "No" options were converted into numerical values (1 or 0) for easier processing. This cleaning process resulted in a final set of 34 numerical variables, including the target variable (annual data consumption). One positive aspect of the data is the absence of missing values. This indicates that the data has been well-maintained and minimizes the need for imputation techniques. Initial examinations using histograms and boxplots suggest that variables like "Outage per week," "Email," and "Monthly Charge" follow normal distributions. Furthermore, the data cleaning process appears to have addressed outliers, as none were identified in the final dataset. Interestingly, a scatterplot revealed a bimodal distribution for both "Bandwidth_GB_Year" (annual data consumption) and "Tenure" (customer time with the company). While a perfectly linear relationship might not exist, this suggests a potential trend that can be further explored through multi-linear regression. Looking at some basic customer characteristics, the average customer in this dataset is 53 years old with two children and an annual income of approximately $39,806. They experience minimal service disruptions (around 10 outage seconds per week) and contact technical support infrequently. The average customer has been with the company for 34.5 months, pays a monthly charge of $173, and consumes 3,392 GB of data annually. These initial insights provide a starting point for understanding the customer base and will be further explored in the analysis.

From Raw to Ready for Analysis

This section outlines the data preparation procedures undertaken to transform the raw data into a format suitable for analysis. The first step involves creating a Python dataframe, a powerful data structure within Python for manipulating and organizing data. Next comes the crucial task of data cleaning. Unnecessary identifying information like customer IDs and zip codes are removed to protect privacy and maintain anonymity. Missing data points, if any, will be addressed strategically. Techniques like calculating central tendency measures (mean, median, or mode) might be used to impute missing values where appropriate. Alternatively, outliers, data points that fall far outside the expected range, might be removed, especially if they are several standard deviations above the mean. Categorical variables, like "Yes" or "No" options for service features, are encoded as numerical values (1 or 0) to facilitate analysis by the multi-linear regression model. Data exploration plays a vital role in this process. Univariate and bivariate visualizations, such as histograms and scatterplots, will be created to understand the distribution of individual variables and potential relationships between them. Finally, the target variable, "Bandwidth GB Year" (annual data consumption), is incorporated into the dataframe. Once all these cleaning and transformation steps are complete, the prepared dataset will be exported as a CSV file named "churn_prepared.csv" for further analysis. This cleaned and structured data provides a solid foundation for building the multi-linear regression model.

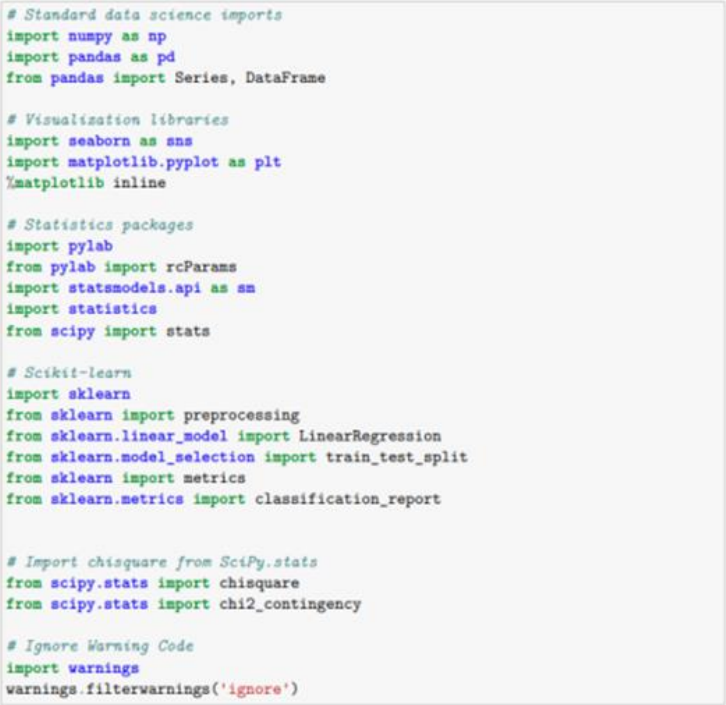
DATA PREPARATION PROCEDURES

1. Include standard imports all the required references:



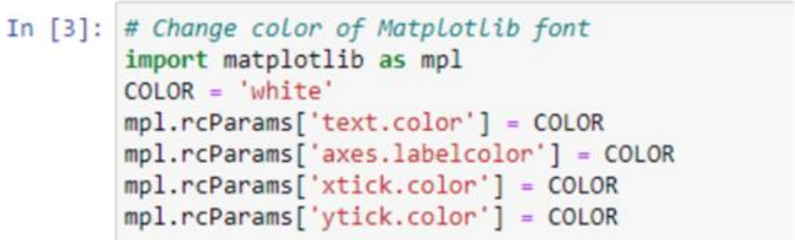
```
# Increase Jupyter display cell-width
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:75% !important; }</style>"))
```

```
<IPython.core.display.HTML object>
```



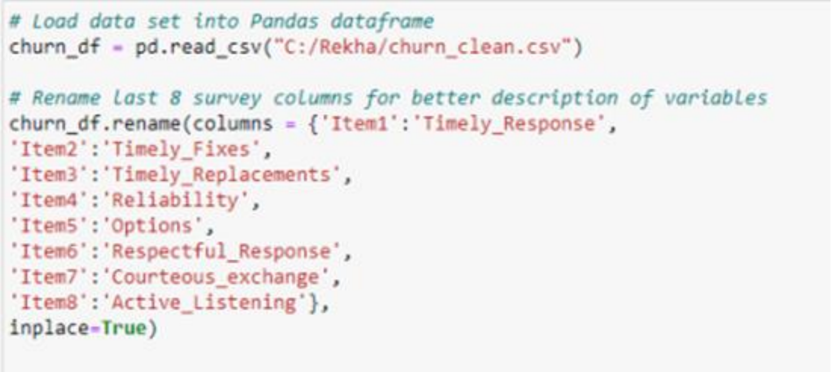
```
# Standard data science imports
import numpy as np
import pandas as pd
from pandas import Series, DataFrame

# Visualization libraries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# Statistics packages
import pylab
from pylab import rcParams
import statsmodels.api as sm
import statistics
from scipy import stats

# Scikit-learn
import sklearn
from sklearn import preprocessing
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import classification_report


# Import chisquare from SciPy.stats
from scipy.stats import chisquare
from scipy.stats import chi2_contingency

# Ignore Warning Code
import warnings
warnings.filterwarnings('ignore')
```

2. Change font and color of the Matplotlib:



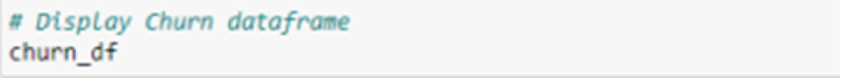
```
In [3]: # Change color of Matplotlib font
        import matplotlib as mpl
        COLOR = 'white'
        mpl.rcParams['text.color'] = COLOR
        mpl.rcParams['axes.labelcolor'] = COLOR
        mpl.rcParams['xtick.color'] = COLOR
        mpl.rcParams['ytick.color'] = COLOR
```

3.Using pandas read the data from clean data file and change the names of the last eight survey columns to better describe the variables:

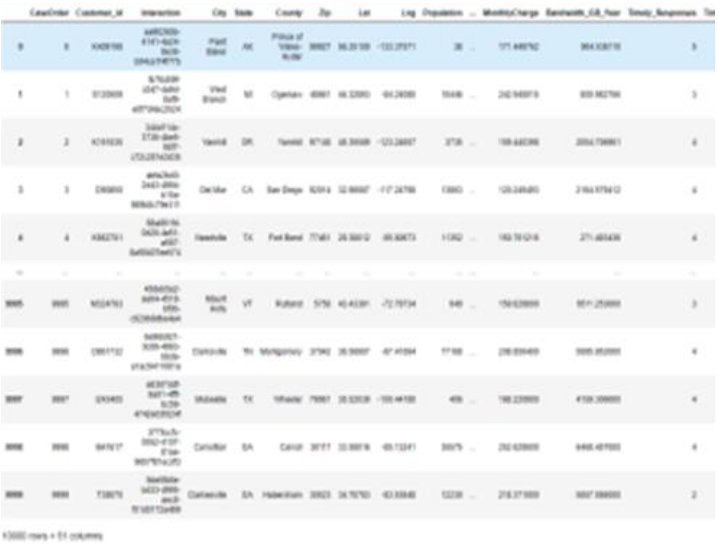


```
# Load data set into Pandas dataframe
churn_df = pd.read_csv("C:/Rekha/churn_clean.csv")

# Rename last 8 survey columns for better description of variables
churn_df.rename(columns = {'Item1':'Timely_Response',
'Item2':'Timely_Fixes',
'Item3':'Timely_Replacements',
'Item4':'Reliability',
'Item5':'Options',
'Item6':'Respectful_Response',
'Item7':'Courteous_exchange',
'Item8':'Active_Listening'},
inplace=True)
```

4. Churn data frame with values:



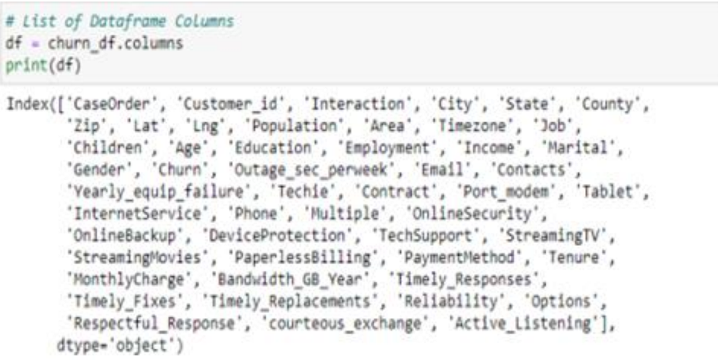
```
# Display Churn dataframe
churn_df
```



	CaseOrder	Customer_id	Interaction	City	State	County	Zip	Lat	Lng	Population	...	MonthlyCharge	Bandwidth_GB_Year	Timely_Responses	Tim
0	1	[illegible]	[illegible]	[illegible]	AK	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	5	
1	2	[illegible]	[illegible]	[illegible]	MI	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	3	
2	3	[illegible]	[illegible]	Yamhill	OR	Yamhill	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
3	4	[illegible]	[illegible]	Del Mar	CA	San Diego	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
4	5	[illegible]	[illegible]	[illegible]	TX	Fort Bend	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	9996	[illegible]	[illegible]	[illegible]	VT	Rutland	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	3	
9996	9997	[illegible]	[illegible]	[illegible]	TN	Montgomery	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
9997	9998	[illegible]	[illegible]	[illegible]	TX	Wheeler	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
9998	9999	[illegible]	[illegible]	[illegible]	GA	Carroll	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	4	
9999	10000	[illegible]	[illegible]	[illegible]	GA	Habersham	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]	[illegible]	2	

10000 rows × 51 columns

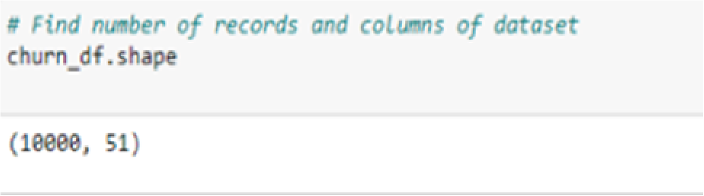
5. To List the data frame columns:



```
# List of Dataframe Columns
df = churn_df.columns
print(df)
```

```
Index(['CaseOrder', 'Customer_id', 'Interaction', 'City', 'State', 'County',
       'Zip', 'Lat', 'Lng', 'Population', 'Area', 'Timezone', 'Job',
       'Children', 'Age', 'Education', 'Employment', 'Income', 'Marital',
       'Gender', 'Churn', 'Outage_sec_perweek', 'Email', 'Contacts',
       'Yearly_equip_failure', 'Techie', 'Contract', 'Port_modem', 'Tablet',
       'InternetService', 'Phone', 'Multiple', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'PaperlessBilling', 'PaymentMethod', 'Tenure',
       'MonthlyCharge', 'Bandwidth_GB_Year', 'Timely_Responses',
       'Timely_Fixes', 'Timely_Replacements', 'Reliability', 'Options',
       'Respectful_Response', 'courteous_exchange', 'Active_Listening'],
      dtype='object')
```

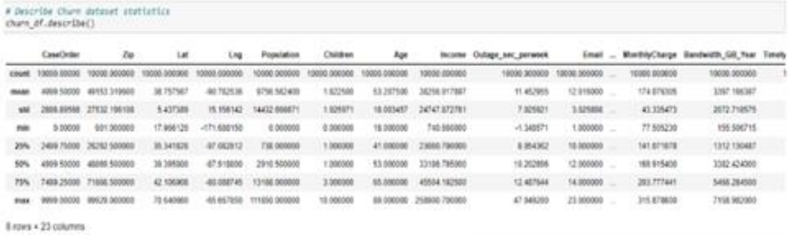
6. To List the records & columns of dataset:



```
# Find number of records and columns of dataset
churn_df.shape
```

```
(10000, 51)
```

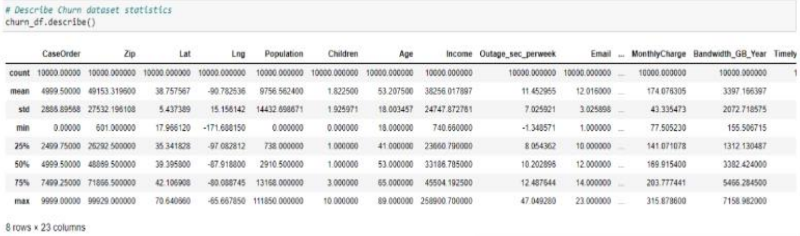
7. List the churn data set statics:



```
# Describe Churn dataset statistics
churn_df.describe()
```

	CaseOrder	Zip	Lat	Lng	Population	Children	Age	Income	Outage_sec_perweek	Email	...	MonthlyCharge	Bandwidth_GB_Year	Timely
count	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	...	10000.000000	10000.000000	1
mean	4999.50000	49153.319600	38.757567	-90.782536	9756.562400	1.822500	53.207500	38256.017897	11.452955	12.016000	...	174.076305	3397.166397	
std	2886.89568	27532.196108	5.437389	15.156142	14432.698671	1.925971	18.003457	24747.872781	7.025921	3.025898	...	43.335473	2072.718575	
min	0.00000	601.000000	17.966120	-171.688150	0.000000	0.000000	18.000000	740.660000	-1.348571	1.000000	...	77.505230	155.506715	
25%	2499.75000	26292.500000	35.341828	-97.082812	738.000000	1.000000	41.000000	23660.790000	8.054362	10.000000	...	141.071078	1312.130487	
50%	4999.50000	48869.500000	39.395800	-87.918800	2910.500000	1.000000	53.000000	33186.785000	10.202896	12.000000	...	169.915400	3382.424000	
75%	7499.25000	71866.500000	42.106908	-80.088745	13168.000000	3.000000	65.000000	45504.192500	12.487644	14.000000	...	203.777441	5466.284500	
max	9999.00000	99929.000000	70.640660	-65.667850	111850.000000	10.000000	89.000000	258900.700000	47.049280	23.000000	...	315.878600	7158.982000	

8 rows × 23 columns

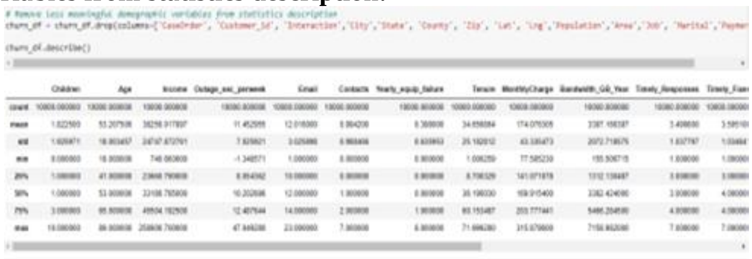


```
# Describe Churn dataset statistics
churn_df.describe()
```

	CaseOrder	Zip	Lat	Lng	Population	Children	Age	Income	Outage_sec_perweek	Email	...	MonthlyCharge	Bandwidth_GB_Year	Timely
count	10000.00000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	...	10000.000000	10000.000000	1
mean	4999.50000	49153.319600	38.757567	-90.782536	9756.562400	1.822500	53.207500	38256.017897	11.452955	12.016000	...	174.076305	3397.166397	
std	2886.89568	27532.196108	5.437389	15.156142	14432.698671	1.925971	18.003457	24747.872781	7.025921	3.025898	...	43.335473	2072.718575	
min	0.00000	601.000000	17.966120	-171.688150	0.000000	0.000000	18.000000	740.660000	-1.348571	1.000000	...	77.505230	155.506715	
25%	2499.75000	26292.500000	35.341828	-97.082812	738.000000	1.000000	41.000000	23660.790000	8.054362	10.000000	...	141.071078	1312.130487	
50%	4999.50000	48869.500000	39.395800	-87.918800	2910.500000	1.000000	53.000000	33186.785000	10.202896	12.000000	...	169.915400	3382.424000	
75%	7499.25000	71866.500000	42.106908	-80.088745	13168.000000	3.000000	65.000000	45504.192500	12.487644	14.000000	...	203.777441	5466.284500	
max	9999.00000	99929.000000	70.640660	-65.667850	111850.000000	10.000000	89.000000	258900.700000	47.049280	23.000000	...	315.878600	7158.982000	

8 rows × 23 columns

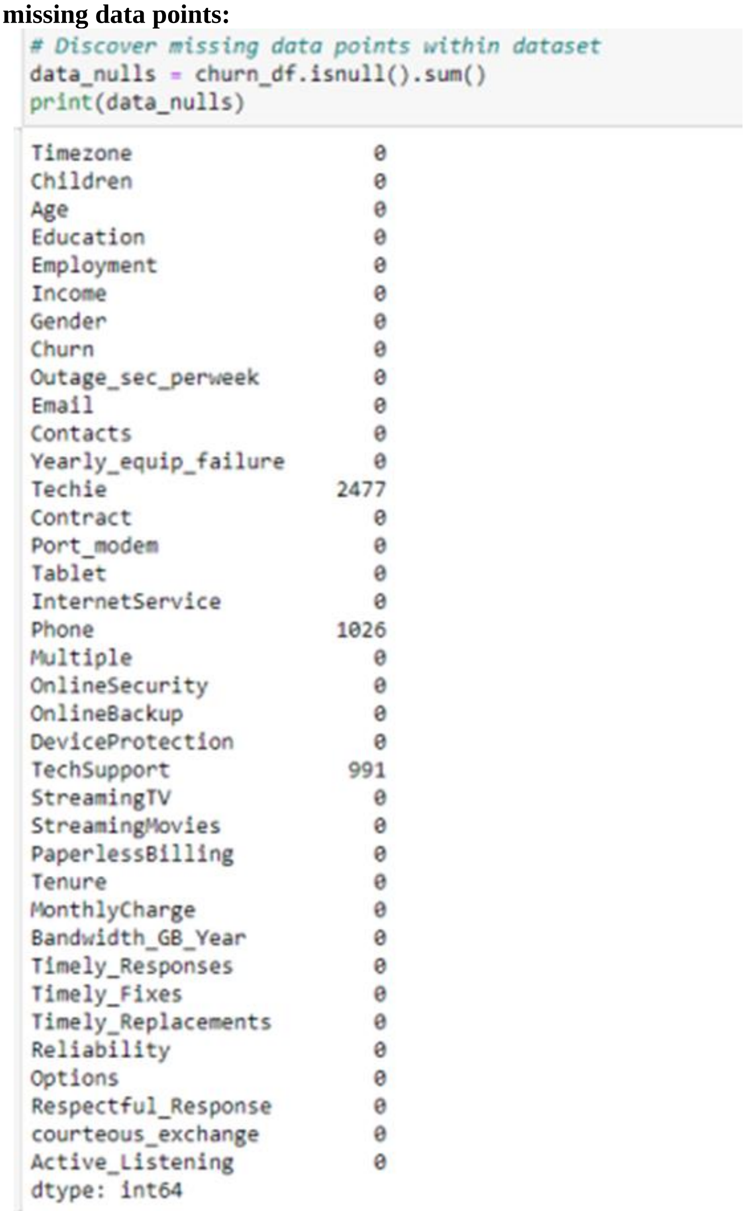
8. Removing variables from statistics description:



```
# Remove less meaningful demographic variables from statistics description
churn_df = churn_df.drop(columns=['CaseOrder', 'Customer_id', 'Interaction','City','State', 'County', 'Zip', 'Lat', 'Lng','Population','Area','Job', 'Marital','Paymen

churn_df.describe()
```

```
          Children           Age         Income  Outage_sec_perweek         Email      Contacts  Yearly_equip_failure        Tenure  MonthlyCharge  Bandwidth_GB_Year  Timely_Responses  Timely_Fixe
count  10000.000000  10000.000000   10000.000000        10000.000000  10000.000000  10000.000000          10000.000000  10000.000000   10000.000000       10000.000000      10000.000000  10000.00000
mean       1.022500     53.207500   38256.917897           11.452955     12.016000      0.994200              0.398000     34.498084     174.076305        3387.166387          3.490800     3.50510
std        1.925971     18.003457   24747.872761            7.025921      3.025898      0.988466              0.635953     25.182012      43.335473        2072.718575          1.037797     1.03464
min        0.000000     18.000000     740.660000           -1.348571      1.000000      0.000000              0.000000      1.000259      77.505230         155.506715          1.000000     1.00000
25%        1.000000     41.000000   23660.790000            8.054362     10.000000      0.000000              0.000000      8.700329     141.071078        1312.130487          3.000000     3.00000
50%        1.000000     53.000000   33186.785000           10.202896     12.000000      1.000000              0.000000     36.196030     169.915400        3382.424000          3.000000     4.00000
75%        3.000000     65.000000   45504.192500           12.487644     14.000000      2.000000              1.000000     60.153487     203.777441        5466.284500          4.000000     4.00000
max       10.000000     89.000000  258900.700000           47.049280     23.000000      7.000000              6.000000     71.999280     315.878600        7158.982000          7.000000     7.00000
```

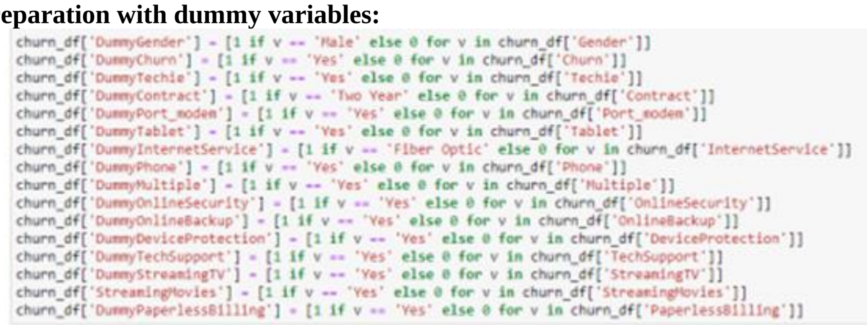
9. Dataset with missing data points:



```
# Discover missing data points within dataset
data_nulls = churn_df.isnull().sum()
print(data_nulls)
```

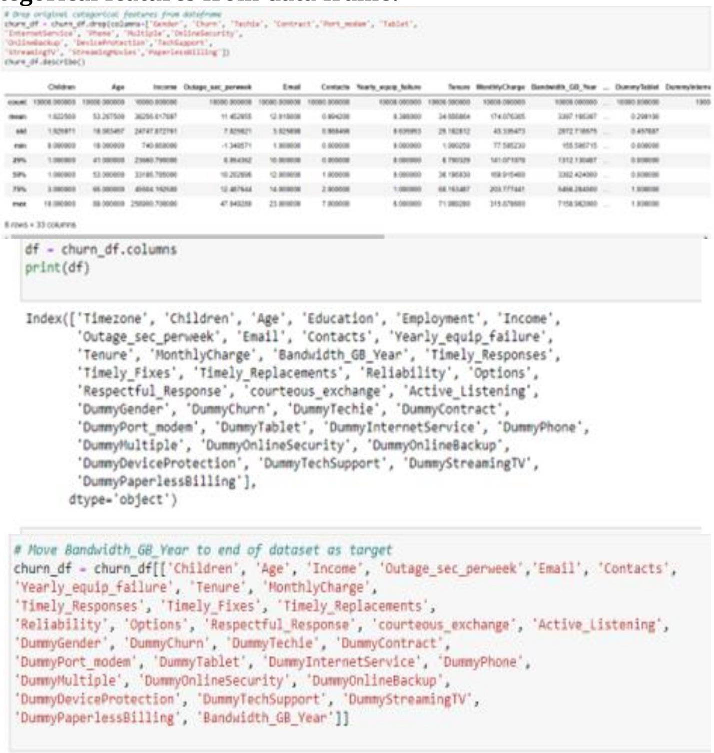
```
Timezone                   0
Children                   0
Age                        0
Education                  0
Employment                 0
Income                     0
Gender                     0
Churn                      0
Outage_sec_perweek         0
Email                      0
Contacts                   0
Yearly_equip_failure       0
Techie                  2477
Contract                   0
Port_modem                 0
Tablet                     0
InternetService            0
Phone                   1026
Multiple                   0
OnlineSecurity             0
OnlineBackup               0
DeviceProtection           0
TechSupport              991
StreamingTV                0
StreamingMovies            0
PaperlessBilling           0
Tenure                     0
MonthlyCharge              0
Bandwidth_GB_Year          0
Timely_Responses           0
Timely_Fixes               0
Timely_Replacements        0
Reliability                0
Options                    0
Respectful_Response        0
courteous_exchange         0
Active_Listening           0
dtype: int64
```

10. Data Preparation with dummy variables:



```
churn_df['DummyGender'] = [1 if v == 'Male' else 0 for v in churn_df['Gender']]
churn_df['DummyChurn'] = [1 if v == 'Yes' else 0 for v in churn_df['Churn']]
churn_df['DummyTechie'] = [1 if v == 'Yes' else 0 for v in churn_df['Techie']]
churn_df['DummyContract'] = [1 if v == 'Two Year' else 0 for v in churn_df['Contract']]
churn_df['DummyPort_modem'] = [1 if v == 'Yes' else 0 for v in churn_df['Port_modem']]
churn_df['DummyTablet'] = [1 if v == 'Yes' else 0 for v in churn_df['Tablet']]
churn_df['DummyInternetService'] = [1 if v == 'Fiber Optic' else 0 for v in churn_df['InternetService']]
churn_df['DummyPhone'] = [1 if v == 'Yes' else 0 for v in churn_df['Phone']]
churn_df['DummyMultiple'] = [1 if v == 'Yes' else 0 for v in churn_df['Multiple']]
churn_df['DummyOnlineSecurity'] = [1 if v == 'Yes' else 0 for v in churn_df['OnlineSecurity']]
churn_df['DummyOnlineBackup'] = [1 if v == 'Yes' else 0 for v in churn_df['OnlineBackup']]
churn_df['DummyDeviceProtection'] = [1 if v == 'Yes' else 0 for v in churn_df['DeviceProtection']]
churn_df['DummyTechSupport'] = [1 if v == 'Yes' else 0 for v in churn_df['TechSupport']]
churn_df['DummyStreamingTV'] = [1 if v == 'Yes' else 0 for v in churn_df['StreamingTV']]
churn_df['StreamingMovies'] = [1 if v == 'Yes' else 0 for v in churn_df['StreamingMovies']]
churn_df['DummyPaperlessBilling'] = [1 if v == 'Yes' else 0 for v in churn_df['PaperlessBilling']]
```

11. Eliminating categorical features from data frame:



```
# Drop original categorical features from dataframe
churn_df = churn_df.drop(columns=['Gender', 'Churn', 'Techie', 'Contract','Port_modem', 'Tablet',
'InternetService', 'Phone', 'Multiple','OnlineSecurity',
'OnlineBackup', 'DeviceProtection','TechSupport',
'StreamingTV', 'StreamingMovies','PaperlessBilling'])
churn_df.describe()
```

```
          Children           Age         Income  Outage_sec_perweek         Email      Contacts  Yearly_equip_failure        Tenure  MonthlyCharge  Bandwidth_GB_Year  ...  DummyTablet  DummyIntern
count 10000.000000  10000.000000   10000.000000        10000.000000  10000.000000  10000.000000          10000.000000  10000.000000   10000.000000       10000.000000  ...  10000.000000        1000
mean      1.622500     53.267500   38256.417687           11.452955     12.016000      0.994200              0.398000     34.550864     174.076305        3397.166367  ...      0.299136
std       1.929871     18.065457   24747.872791            7.829821      3.025898      0.988466              0.639993     26.182812      43.335473        2072.718675  ...      0.457687
min       0.000000     18.000000     740.660000           -1.348571      1.000000      0.000000              0.000000      1.000259      77.505230         155.506715  ...      0.000000
25%       1.000000     41.000000   23660.790000            8.054362     10.000000      0.000000              0.000000      8.790329     141.071078        1312.130487  ...      0.000000
50%       1.000000     53.000000   33185.785000           10.202896     12.000000      1.000000              0.000000     36.196830     169.915400        3382.424000  ...      0.000000
75%       3.000000     66.000000   49664.162500           12.487644     14.000000      2.000000              1.000000     68.163487     203.777441        5466.284500  ...      1.000000
max      10.000000     89.000000  258900.700000           47.049280     23.000000      7.000000              6.000000     71.999280     315.878600        7158.982000  ...      1.000000

8 rows × 33 columns
```

```
df = churn_df.columns
print(df)
```

```
Index(['Timezone', 'Children', 'Age', 'Education', 'Employment', 'Income',
       'Outage_sec_perweek', 'Email', 'Contacts', 'Yearly_equip_failure',
       'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Timely_Responses',
       'Timely_Fixes', 'Timely_Replacements', 'Reliability', 'Options',
       'Respectful_Response', 'courteous_exchange', 'Active_Listening',
       'DummyGender', 'DummyChurn', 'DummyTechie', 'DummyContract',
       'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
       'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
       'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
       'DummyPaperlessBilling'],
      dtype='object')
```

```
# Move Bandwidth_GB_Year to end of dataset as target
churn_df = churn_df[['Children', 'Age', 'Income', 'Outage_sec_perweek','Email', 'Contacts',
'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',
'Timely_Responses', 'Timely_Fixes', 'Timely_Replacements',
'Reliability', 'Options', 'Respectful_Response', 'courteous_exchange', 'Active_Listening',
'DummyGender', 'DummyChurn', 'DummyTechie', 'DummyContract',
'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
'DummyPaperlessBilling', 'Bandwidth_GB_Year']]
```

CONCLUSION

This study successfully employed multi-linear regression to explore the factors influencing customer data consumption in the telecommunications industry. By leveraging Python's data science capabilities and a well-structured dataset, the analysis yielded valuable insights.

The key findings of the data analysis are as follows:

- **Multi-linear Regression Model:** The analysis identified a multi-linear regression model with four key independent variables: Children, Tenure (customer time with the company), Timely Fixes, and Timely Replacements (both service satisfaction measures). The equation for this model is:
  y = (497.78 + 31.18 * Children + 81.94 * Timely_Fixes - 3.66 * Tenure + 1.07 * Timely_Replacements)
- **Variable Influence:** The coefficients associated with each variable in the model indicate their influence on data consumption. The number of children in a household has the strongest positive influence (31.18 units), followed by timely fixes to service issues (81.94 units). Conversely, longer customer tenure has a slightly negative impact (-3.66 units), and timely replacements show a minimal positive influence (1.07 units).
- **Statistical Significance:** It's important to note that the p-values for Children and Tenure are statistically significant at 0.000, indicating a strong correlation with data consumption. However, the p-values for Timely Replacements (0.73) and Timely Fixes (0.25) are not statistically significant, suggesting a weaker or potentially non-existent relationship with data usage in this particular dataset.
- **Limitations and Future Research:** While this study provides valuable insights, it acknowledges limitations. The data collection used was relatively small. Including more data points from additional years could strengthen the model and enhance its generalizability. Additionally, the study highlights the distinction between correlation and causation. While the model suggests a relationship between factors like tenure and data usage, it cannot definitively determine cause and effect. Further research is needed to explore these relationships in more depth.

In conclusion, this study demonstrates the potential of multi-linear regression to understand customer data consumption patterns. The findings provide a foundation for telecommunications companies to develop data plans that better cater to individual customer needs, potentially reducing churn and increasing overall customer satisfaction. Future research with larger datasets and a focus on causal relationships can further refine these insights and provide even more actionable guidance for the telecommunications industry.

REFERENCES

[1]. Massaron, L. & Boschetti, A. (2016). Regression Analysis with Python. Packt Publishing.

[2]. CBTNuggets. (2018, September 20). Why Data Scientists Love Python.