Part 1

A1.
Research question: What is the impact of customer outages, equipment failure, amount of customer data used, customer geographical location, monthly charge, tenure, number of times the customer contacts technical support, payment methods, and area type on predicting customer churn?

A2.
The goal of my analysis is to determine how customer's factors influence their decision to leave the telecommunications companies. Utilizing logistic regression should permit this investigation, and the ability to provide recommendations to the telecommunication companies.

Part 2

B1.
The four assumptions of a logistic regression model are:

1. The dependent variable should be a dichotomous variable.
2. There are one or more independent variables that are either continuous (interval or ratio) or categorical (ordinal or nominal).
3. The observations in the analysis are independent of each other, meaning the values of each dependent variable do not influence observations. The dependent variable should have mutually exclusive and exhaustive categories.
4. There is a linear relationship between continuous independent variables and the logit transformation of the dependent variable.

B2.

Python is the selected programming language because the data set did not require detailed visualizations over periods of time, and it did not require intense statistical technique to clean data. Python is a much more general, robust program that does not require specific libraries and syntax, which makes it easier and more time effective to achieve our goal of conducting multiple linear regression. Furthermore, pandas, numpy, missingno, seaborn, matplotlib, scikitlearn, fancy impute, and statsmodels were all used to conduct various phases of the analysis. The various phases of the analysis for this report included data cleaning, data transformation, feature selection, model construction, and statistical calculations. In comparison to R, I would have had to use specific syntax for each library installed, and this could have unnecessarily increased the complexity of my investigation. Also, Python is considered the more efficient programming language to use and implement forward stepwise procedure for feature selection.

B3.
Logistic regression technique allows you to model the relationship between the binary dependent variable and multiple independent variables or predictors simultaneously, which allows us to better understand real world problems that involve how multiple factors jointly affect a decision. This is the appropriate technique for analyzing the research question because the target variable, whether a customer will churn, is a binary variable. Also, it allows me to investigate how multiple independent variables influence the binary dependent variable, the customer's decision to churn.

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

Logistic regression provides probable quantitative insights into the strength and direction of the relationship between the independent variables and the log-odds of the dependent variable. This helps to determine the four outcomes of the model that determine if the model accurately predicted positive and negative responses or if the model was inaccurate and provided false positives and false negatives. This matrix helps us to validate the model through calculations such as accuracy, specificity, and sensitivity. Additionally, logistic regression offers the ability to assess the probability of an event occurring through most likely outcome calculations and prediction data that are influential in predicting customer churn.

In conclusion, logistic regression is the appropriate technique for analyzing the research question because it accommodates the binary dependent variable and allows for the investigation of multiple independent variables at once, provides insights into the relationships, and offers statistical tests to assess the probability of event occurrence.

In text citations:
("Multiple logistic regression", n.d.)
("Forward stepwise variable selection", n.d.)
("Laerd Statistics", n.d.)

Part 3

C1.
The data cleaning process entails investigating missing and duplicate values to decide how to impute missing values. The goal of this process is to gain insight into missing variables' values for effective data imputation. Clean data is critical for preparing the dataset to conduct multiple linear regression and provide meaningful recommendations to answer the research question.

Data Cleaning Plan:

Step 1: Exploratory Data Analysis

1. Loaded CSV file into Jupyter notebook using pandas library and converted the CSV file into a Data Frame.
2. Used the methods, 'describe()', 'info()', and 'dtypes()' to observe the summary statistics, data types, and nullity, the data types of the data frame.
3. Identified if there was duplicated information using the 'duplicated()' method.
4. Visualized missing data across the entire data frame using missingno and matplotlib libraries, using the 'bar()' and 'show()' methods.
5. Created a correlation matrix using the 'corr()' method, and utilized the correlation matrix to create a heatmap using the seaborn library to understand the correlation between variables missing values.
6. Determined missingness type(s) if applicable.

Step 2: Clean Data

1. Upon completion of investigation, the variables below had missing values:
    a. Children- 7505 (numerical)
    b. Income- 7510 (numerical)
    c. Techie- 7523 (non-numerical)

    d. Age- 7525 (numerical)
    e. Phone- 8974 (non-numerical)
    f. Bandwidth GB Year- 8979 (numerical)
    g. Tech Support- 9009 (non-numerical)
    h. Tenure- 9069 (numerical)
2. By analyzing the summary statistics, I noticed that the population had a minimum value of zero. Since it is highly unlikely population is zero, I replaced all population records with values equal to zero with NaN values utilizing the numpy library and the 'nan' and 'isnan' methods. This allows Python to recognize missing values properly prior to imputation. The missing Population values were imputed using the K-Nearest Neighbor technique.
3. The numerical values appear to be missing at random because the missing values could probably be predicted by other observed values in the data set. For example, missing number of children values could be relative to age and/or marital status (e.g., if a customer is an unmarried young adult, they are likely to be within the age range 18-30, and not list the number of children that they have since it might not be applicable). To treat numerical missing values, I decided to use the K-Nearest Neighbor imputation technique because missing values can likely be predicted based on similar observed variables and values.
4. The non-numerical values also appear to be missing at random because the missing values could also probably be predicted by other observed values in the data set. For example, missing Techie values could also be relative to age (e.g., if a customer is between ages 50-90, it is unlikely that they would be considered a Techie). Likewise, missing values for Phone and Tech Support could likely be related to age due to the fast advancement of technology overtime. To impute non-numerical variables' values, I decided to use the Simple Imputation mode technique, and used frequency to fill in missing data. Furthermore, to validate this imputation technique, I analyzed the clean data to ensure that it fit the well-considered estimate, and the clean data portrayed that the group that are not techie's lies between ages 50-90.

Code for C1:
See code/script attached and below:

Please see Churn Data.ipynb attached for this code.

```
import pandas as pd
import numpy as np
import missingno as msno
churn_raw_data = pd.read_csv ('/Users/jasminemoniquecooper/Downloads/churn_raw_data.csv')
pd.set_option('display.max_columns', None)
churn_raw_data.head(10)

churn_raw_data.dtypes

churn_raw_data.describe()

churn_duplicates = churn_raw_data.duplicated()
churn_raw_data[churn_duplicates]

churn_raw_data.isna().sum()
```

```
#overall summary of missing data in the data frame

#visualize missingness
column_order = churn_raw_data.isnull().sum().sort_values().index
msno.bar(churn_raw_data[column_order])
plt.show()

import seaborn as sns
import matplotlib.pyplot as plt

# Generate a correlation matrix
correlation_matrix = churn_raw_data.corr()

# Set the figure size to make the heatmap larger
plt.figure(figsize=(12, 10))

# Create a heat map using seaborn with customizations
heatmap = sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5,
annot_kws={"size": 8})

# Increase the font size of the color bar (optional)
cbar = heatmap.collections[0].colorbar
cbar.ax.tick_params(labelsize=12)

# Increase the font size of the annotations
for text in heatmap.texts:
    text.set_size(8)

# Display the heat map
plt.show()

churn_raw_data.Population[churn_raw_data.Population == 0].count()

#determine if missing income values are related to employment

# Select the rows where income is missing
missing_income = churn_raw_data[churn_raw_data['Income'].isnull()]

# Group the missing income data by employment status
grouped = missing_income.groupby(['Employment'])

# Count the number of missing income values for each employment status
missing_count = grouped.size()

# Calculate the percentage of missing income values for each employment status
percentage_missing = missing_count / churn_raw_data.groupby(['Employment']).size() * 100
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
# Display the results
print(percentage_missing)
#determine if missing income values are related to employment and age

# Select the rows where income is missing
missing_income = churn_raw_data[churn_raw_data['Income'].isnull()]

# Group the missing income data by employment status
grouped = missing_income.groupby(['Employment', pd.cut(missing_income['Age'], bins=[0, 18, 30, 50,
np.inf])])

# Count the number of missing income values for each employment status
missing_count = grouped.size()

# Calculate the percentage of missing income values for each employment status
percentage_missing = missing_count / churn_raw_data.groupby(['Employment',
pd.cut(churn_raw_data['Age'], bins=[0, 18, 30, 50, np.inf])]).size() * 100

# Display the results
print(percentage_missing)

#part time employment between ages 0-18 is missing data the most

import pandas as pd

# Select the rows where the 'Techie' column is null
missing_techie = churn_raw_data[churn_raw_data['Techie'].isnull()]

# Define age group bins
age_bins = [18, 30, 50, float('inf')]

# Group the missing 'Techie' data by age
grouped = missing_techie.groupby(pd.cut(missing_techie['Age'], bins=age_bins))

# Count the number of missing 'Techie' values for each age group
missing_count = grouped.size()

# Calculate the percentage of missing 'Techie' values for each age group
total_count_by_age_group = churn_raw_data.groupby(pd.cut(churn_raw_data['Age'],
bins=age_bins)).size()
percentage_missing = (missing_count / total_count_by_age_group) * 100

# Display the results
print(percentage_missing)

import pandas as pd
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
# Select the rows where 'number of children' is missing
missing_children = churn_raw_data[churn_raw_data['Children'].isnull()]

# Define age group bins
age_bins = [18, 30, 50, float('inf')]  # Adjust the bins as needed

# Group the missing 'number of children' data by marital status and age
grouped = missing_children.groupby(['Marital', pd.cut(missing_children['Age'], bins=age_bins)])

# Count the number of missing 'number of children' values for each marital status and age group
missing_count = grouped.size()

# Calculate the percentage of missing 'number of children' values for each marital status and age group
total_count_by_group = churn_raw_data.groupby(['Marital', pd.cut(churn_raw_data['Age'],
bins=age_bins)]).size()
percentage_missing = (missing_count / total_count_by_group) * 100

# Display the results
print(percentage_missing)

import numpy as np
churn_raw_data.loc[churn_raw_data.Population == 0, 'Population'] = np.nan
churn_raw_data.Population[np.isnan(churn_raw_data.Population)]

pip install fancyimpute
from fancyimpute import KNN

columns_to_impute = ['Children', 'Income', 'Age', 'Bandwidth_GB_Year', 'Population', 'Tenure']

# Create an instance of the KNN imputer
knn_imputer = KNN()

# Get the column indices of the columns to impute
columns_to_impute_indices = [churn_raw_data.columns.get_loc(col) for col in columns_to_impute]

# Perform imputation on the selected columns
imputed_values = knn_imputer.fit_transform(churn_raw_data.iloc[:, columns_to_impute_indices])

# Assign the imputed values back to the original dataset
churn_raw_data.iloc[:, columns_to_impute_indices] = imputed_values

# Verify if the imputations are in the original dataset
print(churn_raw_data.head())

from sklearn.impute import SimpleImputer

column_to_impute_two = ['Techie', 'Phone', 'TechSupport']
```

```
# Create the SimpleImputer object with strategy='most_frequent'
imputer = SimpleImputer(strategy='most_frequent')

# Fit and transform the categorical variables using the imputer
churn_raw_data[column_to_impute_two] =
imputer.fit_transform(churn_raw_data[column_to_impute_two])

import pandas as pd
# Define age group bins
age_bins = [18, 30, 50, float('inf')]

# Group the data by age
grouped = churn_raw_data.groupby(pd.cut(churn_raw_data['Age'], bins=age_bins))

# Count the number of 'Techie' values for each age group
tech_count = grouped['Techie'].value_counts().unstack(fill_value=0)

# Calculate the percentage of 'Techie' values for each age group
tech_percentage = (tech_count / tech_count.sum(axis=1).values[:, None]) * 100

# Display the results
print(tech_percentage)

#data clean
import matplotlib.pyplot as plt
column_order = churn_raw_data.isnull().sum().sort_values().index
msno.bar(churn_raw_data[column_order])
plt.show()

cleaned_data_two = churn_raw_data
new_name_two = 'churn_cleaned_data'
new_cleaned_data_two = cleaned_data_two.copy()
new_cleaned_data_two.name = new_name_two
new_cleaned_data_two.to_csv('new_cleaned_data_two.csv', index=False)
churn_raw_data.to_csv(r'/Users/jasminemoniquecooper/Downloads/new_cleaned_data_two.csv')
```

C2.
For the dependent variable, churn has two categories: yes or no.

Since the churn variable is a non-numerical, categorical dependent variable then I can evaluate summary statistics by examining the frequency of occurrence for response types.

Frequency of Occurrence for Each Response Type:

- Yes – count of observations that observe the number (or percentage) of customers that discontinued service within the last month

- No – count of observations that observe the number (or percentage) of customers that did not discontinue service within the last month

Based on the output below, it appears that the 10,000 observations are not evenly distributed and are heavily skewed towards customers that did not discontinue service within the last month.

```
Mode (most frequent 'Churn'):  No

Frequency counts of 'Churn':
No     7350
Yes    2650
Name: Churn, dtype: int64

Percentage distribution of 'Churn':
No     73.5
Yes    26.5
Name: Churn, dtype: float64
```

There are eight independent variables: Outage_sec_perweek, MonthlyCharge, Area, Tenure, Yearly_equip_failure, Bandwidth_GB_Year, Contacts, PaymentMethod

For the independent variables, I have 10,000 observations to examine the summary statistics:

Independent variable 1: Outage_sec_perweek – average number of seconds per week of system outages in the customer's neighborhood

- The mean indicates that on average customers experience an 11 second average outage in their neighborhood per week
- The standard deviation of 7 seconds shows that average outages per week in the customer's neighborhood can vary from the mean by plus or minus 7 seconds
- The minimum seconds of outages per week is negative, which is not a valid measure of time, so I will say that the minimum amount of time a customer on average experiences an outage in their neighborhood is 0 seconds. In comparison, the maximum amount of time a customer may on average experience an outage in their neighborhood is 47 seconds.
- The 25th percentile is around 8 seconds, meaning that 25% of customers on average experience an outage of 8 seconds per week or less
- The 75th percentile is around 12 seconds, meaning that 75% of customers on average experience an outage of 12 seconds per week or less
- The median, which is around 10 seconds, meaning that 50% of customers on average experience an outage for more than 10 seconds per week, and 50% of customers experience an outage, on average, for less than 10 seconds per week

The proximity of the mean and median suggests that the distribution may approximate a bell curve or normal distribution.

Independent variable 2: MonthlyCharge – The amount charged to the customer monthly

- The mean indicates that on average customers are charged $174 monthly

- The standard deviation of $43 shows that customer's monthly charges can vary from the mean by plus or minus $43
- The minimum amount a customer is charged monthly is $77, while the maximum monthly charge to a customer is $315, which is indicative of the range of customer monthly charges
- The 25th percentile is $141, meaning that 25% of customers receive a monthly charge of $141 or less
- The 75th percentile is $203, meaning that 75% of customers receive a monthly charge of $203 or less
- The median, which is $169, meaning that 50% of customers receive a monthly charge more than $169, and 50% of customers receive a monthly charge less than $169

The proximity of the mean and median suggests that the distribution may approximate a bell curve or normal distribution.

Independent variable 3: Area – area type (rural, urban, or suburban)

Since the area is a non-numerical, categorical independent variable then I can evaluate summary statistics by examining the frequency of occurrence for each area type.

Frequency of Occurrence for Each Area Type:
- Rural – count of observations in the rural area
- Urban – count of observations in the urban area
- Suburban – count of observations in the suburban area

Based on the output below, it appears that the 10,000 observations are evenly distributed between all 3 area types: rural, urban, and suburban. The balance in the distribution of area types indicates that the dataset provides a representative sample, which is critical for drawing meaningful conclusions in our research.

	MonthlyCharge	Outage_sec_perweek
count	10000.000000	10000.000000
mean	174.076305	11.452955
std	43.335473	7.025921
min	77.505230	-1.348571
25%	141.071078	8.054362
50%	169.915400	10.202896
75%	203.777441	12.487644
max	315.878600	47.049280

```
Mode (most frequent 'Area'):  Suburban

Frequency counts of 'Area':
Suburban    3346
Rural       3327
Urban       3327
Name: Area, dtype: int64

Percentage distribution of 'Area':
Suburban    33.46
Rural       33.27
Urban       33.27
Name: Area, dtype: float64
```

Independent variable 4: Tenure – the number of months the customer has stayed with the provider

- The mean indicates that on average customers stay with telecommunications companies for approximately 34 days
- The standard deviation of 25 days shows that individual customer tenures can vary from the mean by plus or minus 25 days
- The minimum tenure observed is 1 day, while the maximum tenure is approximately 71 days which is indicative of the range of customer durations
- The 25th percentile is around 8 days, meaning that 25% of customers stay for 8 days or less
- The 75th percentile is around 60 days, meaning that 75% of customers stay for 60 days or less
- The median, which is around 36 days, meaning that 50% of customers stay for more than 36 days, and 50% of customers stay for less than 36 days

	Tenure
count	10000.000000
mean	34.624193
std	25.903916
min	1.000259
25%	8.197844
50%	36.852175
75%	60.457955
max	71.999280

The proximity of the mean and median suggests that the distribution may approximate a bell curve or normal distribution.

Independent variable 5: Yearly_equip_failure – the number of times customer's equipment failed and had to be reset/replaced in the past year

- The mean indicates that on average the customer's equipment failed and had to be reset/replaced 0.398 times in the past year.
- The standard deviation of 0.636 shows that the number of times the customer's equipment failed and had to be reset/replaced in the past year can vary from the mean plus or minus 0.636 times
- The minimum number of times a customer experienced equipment failure in the past year is 0 times, while the maximum number of times a customer experienced equipment failure in the past year is 6 times, which is indicative of the range of the number of times a customer experienced equipment failure
- The 25th percentile is 0, meaning that 25% of customers have not experienced equipment failure in the past year
- The 75th percentile is 1, meaning that 75% of customer's equipment failed and had to be reset/replaced in the past year 1 time or less
- The median is 0, meaning that 50% of customers have not experienced equipment failure in the past year

	Yearly_equip_failure
count	10000.000000
mean	0.398000
std	0.635953
min	0.000000
25%	0.000000
50%	0.000000
75%	1.000000
max	6.000000

Independent variable 6: Bandwidth_GB_Year – the average amount of data used, in GB, in a year by the customer

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

	Bandwidth_GB_Year
count	10000.000000
mean	3363.569891
std	2153.675868
min	155.506715
25%	1244.222704
50%	3046.038041
75%	5508.571250
max	7158.982000

- The mean indicates that on average the customer used approximately 3364 GB of data in a year
- The standard deviation of approximately 2154 GB shows that the average amount of data used in the past year can vary from the mean plus or minus 2154 GB
- The minimum amount of data used on average by the customer is approximately 155 GB, while the maximum amount of data used on average by the customer is approximately 7159 GB, which is indicative of the range of the average amount of data used in a year by the customer
- The 25th percentile is around 1244 GB, meaning that on average 25% of customers use 1244 GB of data in a year or less
- The 75th percentile is around 5508 GB, meaning that on average 75% of customers use 5508 GB of data in a year or less
- The median is around 3046 GB, meaning that 50% of customers on average use more than 3046 GB in a year, and 50% of customers on average use less than 3046 GB in a year

Independent variable 7: Contacts – the number of times the customer contacted technical support

	Contacts
count	10000.000000
mean	0.994200
std	0.988466
min	0.000000
25%	0.000000
50%	1.000000
75%	2.000000
max	7.000000

- The mean indicates that on average the customer contacted technical support 0.994 times
- The standard deviation of 0.988 shows that the number of times the customer contacted technical support can vary from the mean plus or minus 0.988 times
- The minimum number of times the customer contacted technical support is 0, while the maximum number of times the customer contacted technical support is 7 times, which is indicative of the range of the number of times the customer contacted technical support
- The 25th percentile is 0, meaning that 25% of customers have not contacted technical support
- The 75th percentile is 2, meaning that 75% of customers contacted technical support 2 times or less
- The median is 1, meaning that 50% of customers contacted technical support more than 1 time, and 50% of customers contacted technical support less than 1 time

Independent variable 8: Payment method – the customer's payment method (electronic check, mailed check, bank transfer, credit card)

Since the payment method is a non-numerical, categorical independent variable then I can evaluate summary statistics by examining the frequency of occurrence for each area type.

Frequency of Occurrence for Each Area Type:

- Electronic check – count of observations for customers that pay with an electronic check
- Mailed check – count of observations for customers that pay with a mailed check
- Bank transfer – count of observations for customers that pay with a bank transfer

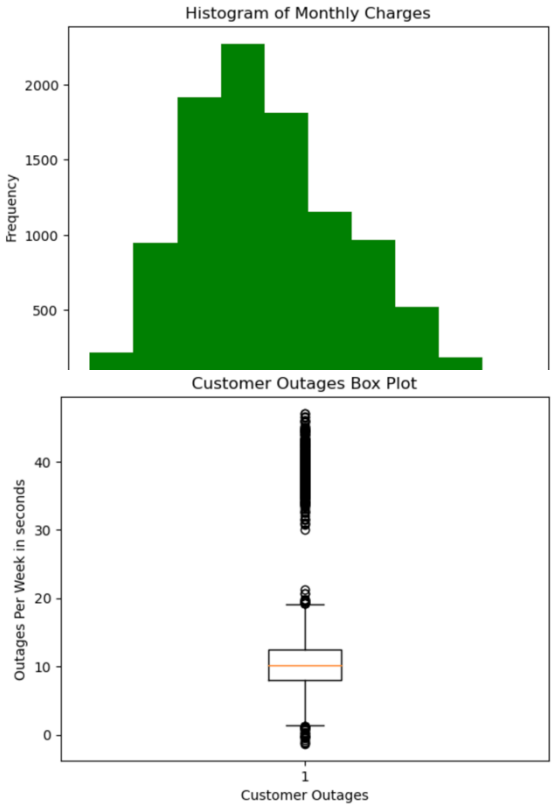
- Credit card – count of observations for customers that pay with a credit card

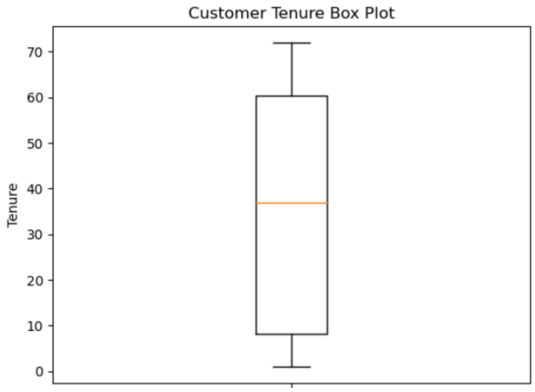
Based on the output below, it appears that the 10,000 observations are somewhat evenly distributed between all the 4 payment types. It appears that even distribution occurs between mailed check, bank transfer, and credit card payments, but there appears to be some skew for electronic check payments, meaning that there seem to be more patients using electronic check as their payment method. The balance in the distribution of payment methods indicates that the dataset provides a representative sample, which is critical for drawing meaningful conclusions in our research.

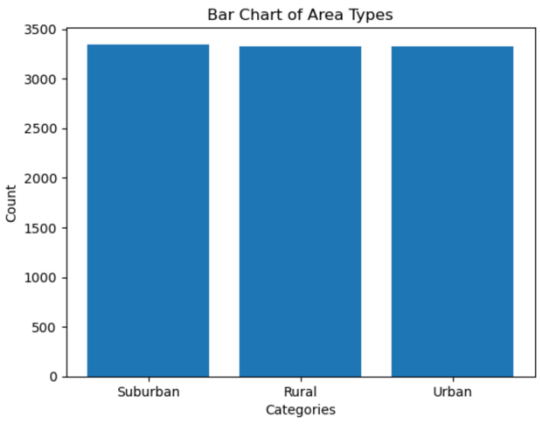
```
Mode (most frequent 'Payment Method'):  Electronic Check

Frequency counts of 'Payment Method':
Electronic Check            3398
Mailed Check                2290
Bank Transfer(automatic)    2229
Credit Card (automatic)     2083
Name: PaymentMethod, dtype: int64

Percentage distribution of 'Payment Method':
Electronic Check            33.98
Mailed Check                22.90
Bank Transfer(automatic)    22.29
Credit Card (automatic)     20.83
Name: PaymentMethod, dtype: float64
```

C3.

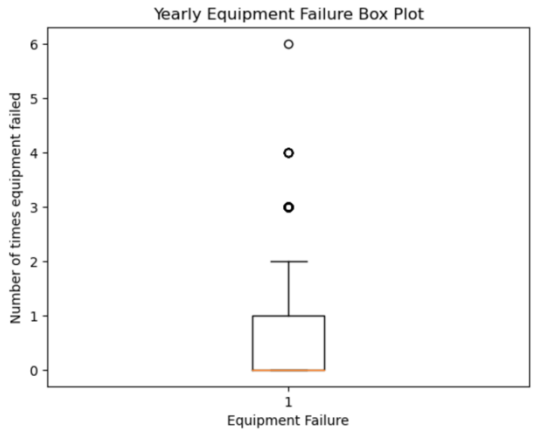
Univariate visualizations:

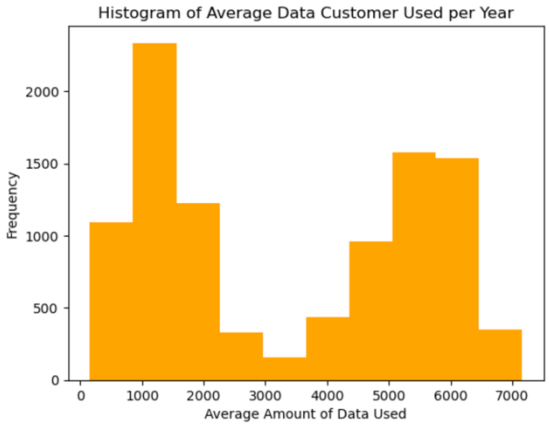


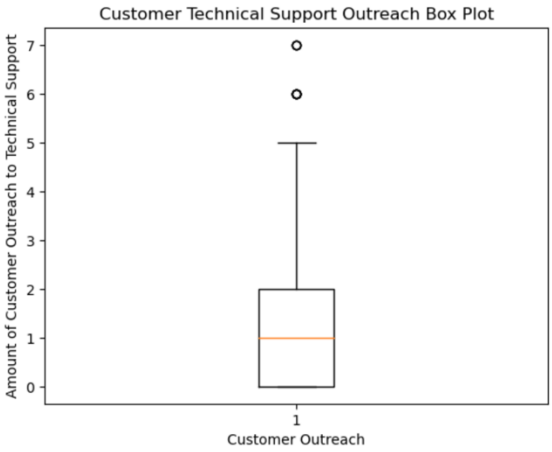


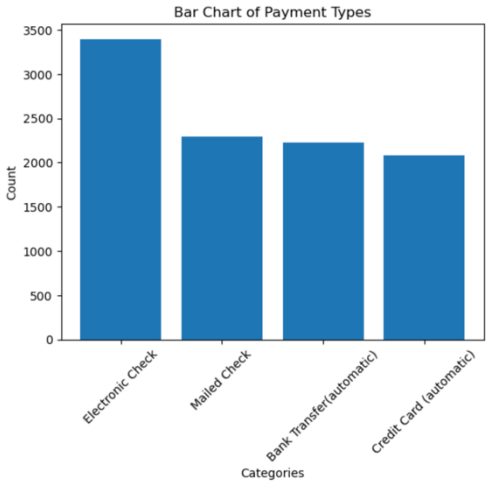


Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University



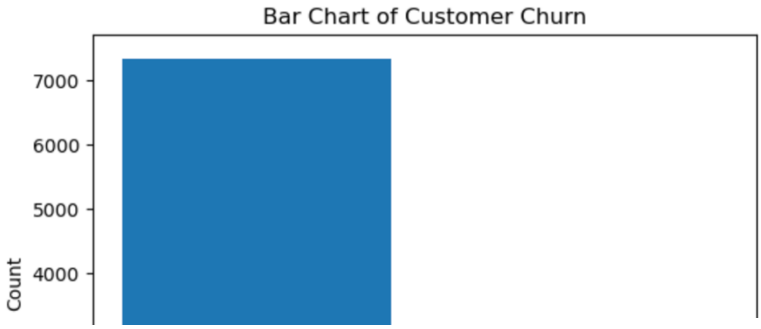


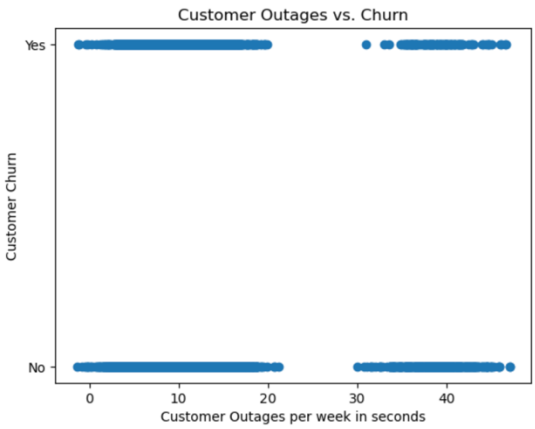


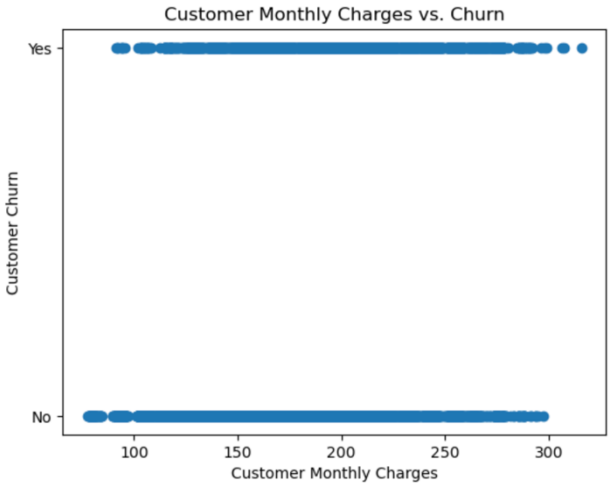


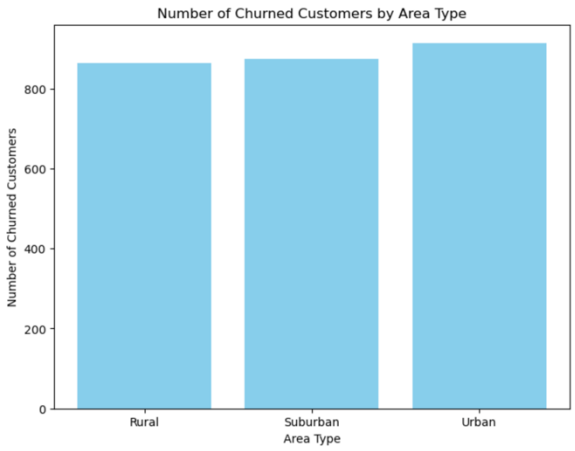
Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

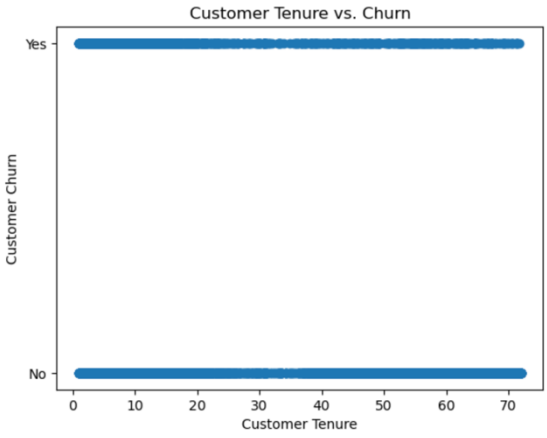
Bivariate visualizations:



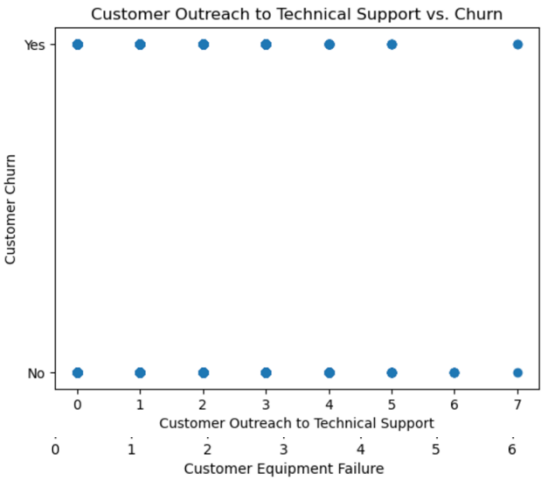


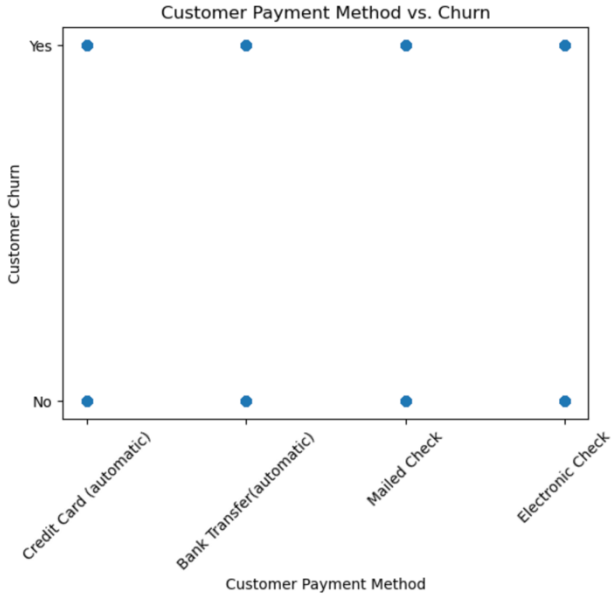






Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University





C4.

Converting columns from data type object to data type category was necessary because there are only a few different values to be selected from each variable and it helps to save memory and ensures that Python recognizes these variables as categorical variables in other libraries and visualizations. Specifically, converting the Churn, Area, PaymentMethod variables allows me to properly calculate the frequency of each area type and portray accurate univariate and bivariate visualizations. Also, to properly construct the model, I transformed the Churn variable by mapping the 'yes' or 'no' options to numerical values. Furthermore, to effectively construct the model and conduct feature selection for the reduced model, I employed one hot encoding, which created three separate binary variables for the Area variable and four separate binary variables for the PaymentMethod variable. The new binary variables created were renamed for proper Python syntax and for ease of use in code.

List of columns converted from data type object to data type category:

- Employment, Marital, Gender, Churn, Techie, Contract, Post_modem, Tablet, InternetService, Phone, Multiple, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, PaperlessBilling, PaymentMethod, and Area

Code for C4 (changed variable names as needed):

```
churn_clean_data["Area"] = churn_clean_data["Area"].astype('category')
churn_clean_data["Area"].dtypes

# One-hot encoding
df_encoded = pd.get_dummies(churn_clean_data, columns=['Area', 'PaymentMethod'])

# Map the "Churn" variable to numeric values
df_encoded["Churn"] = (df_encoded["Churn"] == 'Yes').astype(int)
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
# Rename columns for proper Python syntax
column_name_mapping = {
   "PaymentMethod_Bank Transfer(automatic)": "Bank_Transfer",
   "PaymentMethod_Credit Card (automatic)": "Credit_Card",
   "PaymentMethod_Electronic Check": "Electronic_Check",
   "PaymentMethod_Mailed Check": "Mailed_Check",
   "Area_Suburban": "Suburban",
   "Area_Urban": "Urban",
   "Area_Rural": "Rural"
}

df_encoded.rename(columns=column_name_mapping, inplace=True)
```

C5.
Please see 'new_cleaned_data_two.csv' file attached to submission

In text citations:
("Completeness", n.d.)
("Importing flat files using pandas", n.d.)
("Is data missing at random?", n.d.)
("Mean, median, & mode imputations", n.d.)
("Imputing using facncyimpute", n.d.)
("Measures of center", n.d.)
("Measures of spread", n.d.)
(Gudikandula, 2018)
("GeeksforGeeks," n.d.)

Part 4

D1.
Initial logistic regression model:

```
formula = "Churn ~ Outage_sec_perweek + MonthlyCharge + Tenure + Yearly_equip_failure +
Bandwidth_GB_Year + Contacts + Rural + Suburban + Urban + Bank_Transfer + Credit_Card +
Electronic_Check + Mailed_Check"
mdl_initial = logit(formula, data=df_encoded).fit()
print(mdl_initial.params)
print(mdl_initial.summary())
```

Initial model summary:

Logit Regression Results

Dep. Variable:	Churn	No. Observations:	10000
Model:	Logit	Df Residuals:	9988
Method:	MLE	Df Model:	11
Date:	Thu, 05 Oct 2023	Pseudo R-squ.:	0.3789
Time:	16:21:20	Log-Likelihood:	-3591.4
converged:	True	LL-Null:	-5782.2
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-2.9144	1.05e+06	-2.78e-06	1.000	-2.06e+06	2.06e+06
Outage_sec_perweek	-0.0263	0.004	-6.186	0.000	-0.035	-0.018
MonthlyCharge	0.0326	0.001	37.599	0.000	0.031	0.034
Tenure	-0.0299	0.003	-10.846	0.000	-0.035	-0.025
Yearly_equip_failure	-0.0472	0.047	-1.004	0.315	-0.139	0.045
Bandwidth_GB_Year	-0.0005	3.41e-05	-13.943	0.000	-0.001	-0.000
Contacts	0.0324	0.030	1.086	0.278	-0.026	0.091
Rural	-0.9742	nan	nan	nan	nan	nan
Suburban	-1.0144	nan	nan	nan	nan	nan
Urban	-0.9258	nan	nan	nan	nan	nan
Bank_Transfer	-0.8744	nan	nan	nan	nan	nan
Credit_Card	-0.7823	nan	nan	nan	nan	nan
Electronic_Check	-0.5077	nan	nan	nan	nan	nan
Mailed_Check	-0.7501	nan	nan	nan	nan	nan

D2.

To evaluate the performance and accuracy as well as reduce the initial model, I will use the Area Under the Curve, the AUC, model evaluation metric and recursive feature selection. It quantifies the performance of predictive models with a number between 0 and 1 that conveys how well the model can rank variables based on their likelihood of belonging to the "target" category, providing a valuable measure of predictive performance. An AUC closer to 1 indicates that the model is able perform better at distinguishing between instances that are less likely to belong to the target category. It conveys that the model's predictions are well-calibrated and effectively separate whether or not variables belong to the target category, which makes it valuable for classification and ranking. This aligns with the research question because it helps to classify and rank independent variables, which will be indicative of the impact each variable has on the outcome of the customer churn. Additionally, using a combination of recursive feature selection, next best variable approach, and the AUC model evaluation metric I can determine which variables should be used for the reduced model. By iteratively selecting and evaluating variables based on their contribution to the model's performance, I can identify and retain the most relevant predictors for the reduced model. In conclusion, the AUC model evaluation metric and feature selection techniques enhance model performance and aligns with the research goal of effectively identifying the most impactful predictors of customer churn.

D3.

Reduced linear regression model:

formula = "Churn ~ Tenure + MonthlyCharge + Outage_sec_perweek + Bandwidth_GB_Year + Bank_Transfer + Credit_Card"

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
mdl_reduced_final = logit(formula, data=df_encoded).fit()
print(mdl_reduced_final.summary())
```

Reduced model summary:

```
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9993
Method:                           MLE   Df Model:                            6
Date:                Fri, 06 Oct 2023   Pseudo R-squ.:                  0.3777
Time:                        16:13:41   Log-Likelihood:                -3598.0
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                         coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept             -4.4600      0.146    -30.446      0.000      -4.747      -4.173
Tenure                -0.0300      0.003    -10.860      0.000      -0.035      -0.025
MonthlyCharge          0.0325      0.001     37.598      0.000       0.031       0.034
Outage_sec_perweek    -0.0265      0.004     -6.242      0.000      -0.035      -0.018
Bandwidth_GB_Year     -0.0005    3.4e-05    -13.908      0.000      -0.001      -0.000
Bank_Transfer         -0.2646      0.074     -3.576      0.000      -0.410      -0.120
Credit_Card           -0.1767      0.076     -2.324      0.020      -0.326      -0.028
==============================================================================
```

E1.

The data analysis process involved the following:

1. Data Transformation – To construct the initial model, multiple data transformation steps were conducted first. I transformed the Churn variable by mapping the ‘yes’ or ‘no’ options to numerical values. Then, I employed one hot encoding, which created three separate binary variables for the Area variable and four separate binary variables for the PaymentMethod variable. These new binary variables were then renamed for proper Python syntax and ease of use in code.
2. Initial Model Analysis – I initiated the data analysis process by constructing an initial logistic regression model. Then, to assess the performance, I computed the AUC as the model evaluation metric to serve as a base line for comparison.
3. Recursive feature selection – To refine the model, I choose to conduct recursive feature selection to identify the most influential variables. I did not provide the number of features to select in the code, and identified variables based on its automated feature ranking process. Through this process it was determined that six variables (MonthlyCharge, Tenure, Suburban, Urban, Bank_Transfer, and Credit_Card) were the most significant contributors to the model’s performance.
4. First Reduced Model – Using the selected features in step 3, I built a reduced logistic regression model. Then, I recalculated the AUC for the reduced model.

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor’s University

5. Next Best Variable – I applied the next best variable approach to identify the rank of the most significant predictors, and this involved iteratively adding one feature at a time based on the AUC improvement. I choose the first 6 variables (Tenure, MonthlyCharge, Outage_sec_perweek, Bandwidth_GB_Year, Bank_Transfer, and Credit_Card) since the recursive feature selection identified six variables as the most significant using the automated feature ranking process. As noted, the six variables selected from the recursive feature selection process are different from the six variables selected from the next best variable process. To determine which variables to select for the reduced model, I decided to calculate the AUC for both set of variables. As a result, the AUC for the six variables selected in the next best variable process had a higher AUC indicating better model performance.
6. Second and Final Reduced Model Analysis – I constructed a reduced model utilizing six variables from the next best variable approach. Then, to reassess model performance, I calculated the AUC for the reduced model.
7. Comparison Analysis – Upon comparing the AUC values of the initial and reduced models, it was determined that both models yielded an AUC of approximately 0.89. This means that both models can effectively determine between variables that do or do not belong to the target category. This outcome indicates that the reduced model, which only includes the most essential features, can achieve a comparable predictive performance with a simpler model.

E2.
Output and all calculations of data analysis performed:

1. Data transformations: Only calculations are below since no output code was entered, and I know transformations worked because I was able to run model and subsequential code without error.

```
# One-hot encoding
df_encoded = pd.get_dummies(churn_clean_data, columns=['Area', 'PaymentMethod'])

# Map the "Churn" variable to numeric values
df_encoded["Churn"] = (df_encoded["Churn"] == 'Yes').astype(int)

# Rename columns for proper Python syntax
column_name_mapping = {
    "PaymentMethod_Bank Transfer(automatic)": "Bank_Transfer",
    "PaymentMethod_Credit Card (automatic)": "Credit_Card",
    "PaymentMethod_Electronic Check": "Electronic_Check",
    "PaymentMethod_Mailed Check": "Mailed_Check",
    "Area_Suburban": "Suburban",
    "Area_Urban": "Urban",
    "Area_Rural": "Rural"
}

df_encoded.rename(columns=column_name_mapping, inplace=True)
```

2. Initial Model Analysis:
    a. Calculations and output summary provided in D1
    b. Initial Model AUC calculation:

```
from sklearn.metrics import roc_auc_score

# Get predicted class labels (0 or 1)
predicted_labels = mdl_initial.predict(df_encoded)

# Calculate ROC AUC score
roc_auc = roc_auc_score(df_encoded["Churn"], predicted_labels)
print("ROC AUC Score:", roc_auc)
```

```
ROC AUC Score: 0.8878320112950842
```

3. Recursive Feature selection and outputs:

```
# Create a new DataFrame with only the numeric columns
numeric_columns = df_encoded.select_dtypes(include=[np.number]).columns.tolist()

df_numeric = df_encoded[numeric_columns]

#independent variables:
independent_vars = ["Churn", "Outage_sec_perweek", "MonthlyCharge", "Tenure", "Yearly_equip_failure", "Ban

df_independent = df_numeric[independent_vars]
print(df_independent)
```

```
ntacts", "Rural", "Suburban", "Urban", "Bank_Transfer", "Credit_Card", "Electronic_Check", "Mailed_Check"]
```

```
      Churn  Outage_sec_perweek  MonthlyCharge     Tenure  \
0         0            6.972566     171.449762   6.795513
1         1           12.014541     242.948015   1.156681
2         0           10.245616     159.440398  15.754144
3         0           15.206193     120.249493  17.087227
4         1            8.960316     150.761216   1.670972
...     ...                 ...            ...        ...
9995      0            9.265392     159.828800  68.197130
9996      0            8.115849     208.856400  61.040370
9997      0            4.837696     168.220900  13.446717
9998      0           12.076460     252.628600  71.095600
9999      0           12.641760     218.371000  63.350860

      Yearly_equip_failure  Bandwidth_GB_Year  Contacts  Rural  Suburban  \
0                        1         904.536110         0      0         0
1                        1         800.982766         0      0         0
2                        1        2054.706961         0      0         0
3                        0        2164.579412         2      0         1
4                        1         271.493436         2      0         1
...                    ...                ...       ...    ...       ...
9995                     0        6511.253000         2      1         0
9996                     0        5695.952000         2      1         0
9997                     0        4159.306000         0      1         0
9998                     0        6468.457000         1      0         0
9999                     0        5857.586000         1      0         0

      Urban  Bank_Transfer  Credit_Card  Electronic_Check  Mailed_Check
0         1              0            1                 0             0
1         1              1            0                 0             0
2         1              0            1                 0             0
3         0              0            0                 0             1
4         0              0            0                 0             1
...     ...            ...          ...               ...           ...
9995      0              0            0                 1             0
9996      0              0            0                 1             0
```

```
      Urban  Bank_Transfer  Credit_Card  Electronic_Check  Mailed_Check
0         1              0            1                 0             0
1         1              1            0                 0             0
2         1              0            1                 0             0
3         0              0            0                 0             1
4         0              0            0                 0             1
...     ...            ...          ...               ...           ...
9995      0              0            0                 1             0
9996      0              0            0                 1             0
9997      0              1            0                 0             0
9998      1              0            1                 0             0
9999      1              0            0                 1             0
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

[105]:
```
# Import necessary libraries
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression

# Extract your target variable (y) and feature matrix (X) from your data
X = df_independent.drop('Churn', axis=1)  # Assuming 'Churn' is the target variable
y = df_independent['Churn']  # Assuming 'Churn' is the target variable

# Create the logistic regression model as the estimator for RFE
estimator = LogisticRegression()

# Create the RFE object with the estimator and number of features to select
rfe = RFE(estimator)
rfe.fit(X, y)  # X is the feature matrix, y is the target variable

# Get the selected features; it returns a boolean mask to indicate which features are selected
selected_features = rfe.support_
print("Selected Features:", selected_features)

# Rank the selected features; lower rankings mean the features are more important
feature_ranking = rfe.ranking_
print("Feature Ranking:", feature_ranking)

# Use the selected features to create a reduced feature matrix
# Train your logistic regression model with only these features
X_reduced = X.loc[:, selected_features]
print("Reduced Feature Matrix (X_reduced):", X_reduced)
```

```
Selected Features: [False False  True  True False False False False  True  True  True  True
 False]
Feature Ranking: [6 5 1 1 8 7 3 4 1 1 1 1 2]
Reduced Feature Matrix (X_reduced):          Tenure  Yearly_equip_failure  Urban  Bank_Transfer  Credit_Ca
rd  \
0      6.795513                     1      1              0           1
1      1.156681                     1      1              1           0
2     15.754144                     1      1              0           1
3     17.087227                     0      0              0           0
4      1.670972                     1      0              0           0
...         ...                   ...    ...            ...         ...
9995  68.197130                     0      0              0           0
9996  61.040370                     0      0              0           0
9997  13.446717                     0      0              1           0
9998  71.095600                     0      1              0           1
9999  63.350860                     0      1              0           0

      Electronic_Check
0                    0
1                    0
2                    0
3                    0
4                    0
...                ...
9995                 1
9996                 1
9997                 0
9998                 0
9999                 1

[10000 rows x 6 columns]
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

4. First reduced model and outputs:

```
formula = "Churn ~ MonthlyCharge + Tenure + Suburban + Urban + Bank_Transfer + Credit_Card"
mdl_reduced = logit(formula, data=df_encoded).fit()
print(mdl_reduced.params)
print(mdl_reduced.summary())
```

```
Optimization terminated successfully.
```

```
Optimization terminated successfully.
         Current function value: 0.371702
         Iterations 7
Intercept       -4.598793
MonthlyCharge    0.029351
Tenure          -0.063369
Suburban        -0.041062
Urban            0.074082
Bank_Transfer   -0.256587
Credit_Card     -0.175275
dtype: float64
                           Logit Regression Results
==============================================================================
Dep. Variable:                  Churn   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9993
Method:                           MLE   Df Model:                            6
Date:                Thu, 05 Oct 2023   Pseudo R-squ.:                 0.3572
Time:                        22:40:03   Log-Likelihood:                -3717.0
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept       -4.5988      0.146    -31.425      0.000      -4.886      -4.312
MonthlyCharge    0.0294      0.001     36.483      0.000       0.028       0.031
Tenure          -0.0634      0.002    -40.998      0.000      -0.066      -0.060
Suburban        -0.0411      0.072     -0.574      0.566      -0.181       0.099
Urban            0.0741      0.071      1.046      0.296      -0.065       0.213
Bank_Transfer   -0.2566      0.073     -3.527      0.000      -0.399      -0.114
Credit_Card     -0.1753      0.075     -2.345      0.019      -0.322      -0.029
==============================================================================
```

```
]: from sklearn.metrics import roc_auc_score

   # Get predicted class labels (0 or 1)
   predicted_labels_two = mdl_reduced.predict(df_encoded)

   # Calculate ROC AUC score
   roc_auc = roc_auc_score(df_encoded["Churn"], predicted_labels_two)
   print("ROC AUC Score:", roc_auc)
```

```
ROC AUC Score: 0.8793508920549351
```

5. Next Best Approach:

```
def next_best(current_variables, candidate_variables, target, data):
    best_auc = -1
    next_variable = None

    for candidate_var in candidate_variables:
        # Create a list of current variables plus the candidate variable
        variables_to_use = current_variables + [candidate_var]

        # Calculate AUC using logistic regression
        import statsmodels.api as sm
        X = data[variables_to_use]
        model = sm.Logit(target, X).fit()
        predicted_probabilities = model.predict(X)
        auc = roc_auc_score(target, predicted_probabilities)

        # Update the next variable if AUC is better
        if auc > best_auc:
            best_auc = auc
            next_variable = candidate_var

    return next_variable

# Find the candidate variables
candidate_variables = ["Outage_sec_perweek", "MonthlyCharge", "Tenure", "Yearly_equip_failure", "Bandwidth
```

```
"Bandwidth_GB_Year", "Contacts", "Rural", "Suburban", "Urban", "Bank_Transfer", "Credit_Card", "Electronic_
```

```
ntacts", "Rural", "Suburban", "Urban", "Bank_Transfer", "Credit_Card", "Electronic_Check", "Mailed_Check"]
```

```
# Initialize the current variables
current_variables = []
target = df_encoded["Churn"]

# The forward stepwise variable selection procedure
number_iterations = 13
for i in range(number_iterations):
    next_variable = next_best(current_variables, candidate_variables, target, df_encoded)
    current_variables.append(next_variable)
    candidate_variables.remove(next_variable)
    print("Variable added in step " + str(i + 1) + " is " + next_variable + ".")
print("Selected variables:", current_variables)
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
Optimization terminated successfully.
         Current function value: 0.600321
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.619552
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.468003
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.656293
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.483506
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.635469
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.652982
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.653282
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.658131
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.664072
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.667967
         Iterations 5
```

```
Optimization terminated successfully.
         Current function value: 0.660183
         Iterations 5
Optimization terminated successfully.
         Current function value: 0.664972
         Iterations 5
Variable added in step 1 is Tenure.
Optimization terminated successfully.
         Current function value: 0.466166
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.443988
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467736
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.468003
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.466167
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467322
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467506
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.466545
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467962
         Iterations 6
Optimization terminated successfully.
```

```
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.466545
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467962
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467680
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.465537
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.467631
         Iterations 6
Variable added in step 2 is MonthlyCharge.
Optimization terminated successfully.
         Current function value: 0.430588
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.441640
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.433755
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.441051
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.442037
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.441325
         Iterations 7
```

```
Optimization terminated successfully.
         Current function value: 0.442908
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.441278
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.442630
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.443616
         Iterations 6
Optimization terminated successfully.
         Current function value: 0.442679
         Iterations 6
Variable added in step 3 is Outage_sec_perweek.
Optimization terminated successfully.
         Current function value: 0.428869
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.420671
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.428341
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.429277
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.428366
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.429664
         Iterations 7
```

```
Optimization terminated successfully.
         Current function value: 0.428242
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.429469
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.430400
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.429528
         Iterations 7
Variable added in step 4 is Bandwidth_GB_Year.
Optimization terminated successfully.
         Current function value: 0.419046
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.418561
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.419399
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.418590
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.419643
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.418298
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.419551
         Iterations 7
```

```
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.420505
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.419599
         Iterations 7
Variable added in step 5 is Bank_Transfer.
Optimization terminated successfully.
         Current function value: 0.416685
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.416390
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.417058
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.416233
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.417383
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.416028
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.417252
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.416010
         Iterations 7
Variable added in step 6 is Credit_Card.
Optimization terminated successfully.
         Current function value: 0.414507
         Iterations 7
```

```
Optimization terminated successfully.
         Current function value: 0.414262
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.414766
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.414073
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.415216
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.412088
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.411212
         Iterations 7
Variable added in step 7 is Mailed_Check.
Optimization terminated successfully.
         Current function value: 0.409735
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.409644
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.410059
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.409511
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.410497
         Iterations 7
Optimization terminated successfully.
```

```
Optimization terminated successfully.
         Current function value: 0.359333
         Iterations 7
Variable added in step 8 is Electronic_Check.
Optimization terminated successfully.
         Current function value: 0.359281
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359272
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359333
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359278
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359268
         Iterations 7
Variable added in step 9 is Urban.
Optimization terminated successfully.
         Current function value: 0.359218
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359210
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359253
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359253
         Iterations 7
Variable added in step 10 is Contacts.
Optimization terminated successfully.
         Current function value: 0.359159
```

```
Optimization terminated successfully.
         Current function value: 0.359194
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359194
         Iterations 7
Variable added in step 11 is Rural.
Optimization terminated successfully.
         Current function value: 0.359144
         Iterations 7
Optimization terminated successfully.
         Current function value: 0.359194
         Iterations 7
Variable added in step 12 is Yearly_equip_failure.
Optimization terminated successfully.
         Current function value: 0.359144
         Iterations 7
Variable added in step 13 is Suburban.
Selected variables: ['Tenure', 'MonthlyCharge', 'Outage_sec_perweek', 'Bandwidth_GB_Year', 'Bank_Transfe
r', 'Credit_Card', 'Mailed_Check', 'Electronic_Check', 'Urban', 'Contacts', 'Rural', 'Yearly_equip_failur
e', 'Suburban']
```

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

```
e', 'Suburban']
```

```
]: from sklearn.metrics import roc_auc_score
formula = "Churn ~ Tenure + MonthlyCharge + Outage_sec_perweek + Bandwidth_GB_Year + Bank_Transfer + Credi
mdl_reduced_final = logit(formula, data=df_encoded).fit()
predict_final = mdl_reduced_final.predict(df_encoded)
roc_auc = roc_auc_score(df_encoded["Churn"], predict_final)
print("ROC AUC Score:", roc_auc)
```

```
Optimization terminated successfully.
         Current function value: 0.359799
         Iterations 7
ROC AUC Score: 0.8874038249261968
```

6. Final Reduced Model Analysis:
    a. Calculations and output summary provided in D3
    b. Final reduced model AUC calculation:


```
e', 'Suburban']
```

```
]: from sklearn.metrics import roc_auc_score
formula = "Churn ~ Tenure + MonthlyCharge + Outage_sec_perweek + Bandwidth_GB_Year + Bank_Transfer + Credi
mdl_reduced_final = logit(formula, data=df_encoded).fit()
predict_final = mdl_reduced_final.predict(df_encoded)
roc_auc = roc_auc_score(df_encoded["Churn"], predict_final)
print("ROC AUC Score:", roc_auc)
```

```
Optimization terminated successfully.
         Current function value: 0.359799
         Iterations 7
ROC AUC Score: 0.8874038249261968
```


7. Confusion Matrix

```
: conf_matrix = mdl_initial.pred_table()

TP = conf_matrix[1, 1]
TN = conf_matrix[0, 0]
FP = conf_matrix[0, 1]
FN = conf_matrix[1, 0]

print("Confusion Matrix:")
print(conf_matrix)
print(f"True Positives (TP): {TP}")
print(f"True Negatives (TN): {TN}")
print(f"False Positives (FP): {FP}")
print(f"False Negatives (FN): {FN}")
```

```
Confusion Matrix:
[[6727.  623.]
 [1096. 1554.]]
True Positives (TP): 1554.0
True Negatives (TN): 6727.0
False Positives (FP): 623.0
False Negatives (FN): 1096.0
```

8. Accuracy calculation

```
: #accuracy calculation
#(TN + TP)/(TN + FN + FP + TP)

accuracy = (TN + TP)/(TN + FN + FP + TP)
print(accuracy)
```

```
0.8281
```

E3. Please see Churn Clean Data Task 2.ipynb for code

In text citations:
(Lekhana_Ganji, 2023)

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

("Working with model objects", n.d.)
("Multiple logistic regression", n.d.)
("Motivation for variable selection", n.d.)
("Forward stepwise variable selection", n.d.)
("Selecting features for better model performance", n.d.)
("Quantifying logistic regression fit", n.d.)

Part 5

F1.
Data Summary:

Regression equation for the reduced model:
Y = -0.03*Tenure + 0.03*MonthlyCharge – 0.03*Outage_sec_perweek – 0.00047*Bandwidth_GB_Year – 0.265*Bank_Transfer – 0.177*Credit_Card – 4.46

Interpretation of the coefficients of the reduced model:
$Y = the\ dependent\ variable\ the\ model\ is\ predicting$
$b = the\ constant\ intercept\ term$
$a_1 = the\ coefficient\ for\ the\ Tenure\ independent\ variable$
$a_2 = the\ coefficient\ for\ the\ MonthlyCharge\ indepedent\ variable$
$a_3 = the\ coefficient\ for\ the\ Outage$ sec $per\ week\ \ independent\ variable$
$a_4 = the\ coefficient\ for\ the\ Bandwidth\ GB\ Year\ independent\ variable$
$a_5 = the\ coefficient\ for\ the\ Bank\ Transfer\ independent\ variable$
$a_6 = the\ coefficient\ for\ the\ Credit\ Card\ independent\ variable$
$The\ independent\ variables\ represent\ a\ change\ in\ the$ log $odds\ \ of\ the\ dependent\ variable\ for\ a\ one\ unit\ change\ in\ the\ independent\ variable\ while\ holding\ all\ other\ variables\ constant.\ The\ coefficients\ represent\ the\ change\ in\ the$ log $odds\ and\ not\ the\ dependent\ variable\ directly.\ The\ logistic\ regression\ model\ conveys\ the\ probability\ of\ an\ event\ occurring.\ To\ properly\ interpret\ the\ probability\ of\ an\ event\ occurring,\ the\ odds\ ratio\ would\ be\ utilized,\ which\ is\ the\ probability\ of\ an\ event\ occurring\ divided\ by\ the\ probability\ of\ an\ event\ not\ occurring.$

Statistical and practical significance of the reduced model:

The p-values for the independent variables were below a well-known significance level of alpha = 0.05, and therefore, were found to be statistically significant. This indicates that these independent variables have a statistically significant impact on the dependent variable.

While the coefficients of the independent variables are statistically significant, it is also important to consider their practical significance. The reduced model could be meaningful in the real world if telecommunication companies want to assess where the highest customer churn occurs based on a plethora of selected factors including the number of months a customer has been with the provider, the average monthly charge per customer, the average number of seconds per week of system outages a customer experiences in their neighborhood, the average amount of data the customer utilizes yearly, and if a customer pays with a bank transfer or credit card.

Data Analysis limitations:

For this data analysis, I discovered three limitations, which include:

1. The challenge to determine the optimal number of variables for the reduced model without having to conduct a manual iterative and time-consuming process. To address this issue, I used a combination of recursive feature selection, the next best approach, and the AUC evaluation metric. The goal was to identify the smallest number of variables that could provide similar performance to the initial model. In my opinion, selecting a small set of independent variables (or predictors), for a reduced model, can be one of the most efficient ways to determine which independent variables have the greatest impact on the probability of an event occurring. Furthermore, it allows companies to make decisions quicker without hefty investigation which saves companies time and allows companies to implement solutions at a quicker pace.
2. The limitation of missing data was also noted. Addressing these limitations required thorough investigation and subsequentially imputing missing data points. Although missing data mirrors real-world scenarios where data is typically incomplete, it's important to acknowledge that a complete data set can provide a more comprehensive basis for constructing logistic regression models
3. The last limitation I noticed was that none of the customer's geographical location area types were identified as a highly selected feature. I found it interesting that outages_sec_perweek was selected without at least one area type since the outage_sec_perweek variable is based on how long customer's experienced outages within their neighborhood. I think this could be a limitation because it might be challenging for companies to determine how to decrease outages without understanding which area(s) are affected the most.

F2.

The recommended course of action would be to investigate the impact of payment methods, Bank_Transfer and Credit_Card, on the probability of customer churn. This recommendation is based on the observation that these variables have the largest absolute value coefficients in the logistic regression model, indicating these variables have a significant influence on the probability of customer churn. This suggestion could help the telecommunication companies implement new payment strategies that increase customer satisfaction and retention. In conclusion, further investigation would need to be conducted into these two forms of payments and how it impacts customer churn.

The telecommunications companies can begin by collecting more data regarding payment methods and customer churn to segment customers based on their payment preferences to identify patterns or trends. The team could also consider gathering direct customer feedback. The team could use this data to formulate hypotheses about how payment methods may influence customer churn and conduct further in-depth statistical and analytical investigations to understand the relationship. This analysis could provide insight into whether certain payment methods are associated with lower churn rates, and whether there are differences amongst customer segments.

In conclusion, the telecommunication companies can gain valuable insights into customer behavior by evaluating the impact of payment methods. This will allow the companies to make data driven decisions as to how to optimize payment strategies and customer retention efforts.

In text citations:
(LaMorte, 2016)

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University

("Logistic regression", n.d.)
("Using the logistic regression model", n.d.)
("Predictions and odds ratios", n.d.)

Part 6

H.
Citations for code:

DataCamp. (n.d.). Forward Stepwise Variable Selection [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/forward-stepwise-variable-selection-for-logistic-regression?ex=5

DataCamp. (n.d.). Completeness [Video file]. Retrieved from https://campus.datacamp.com/courses/cleaning-data-in-python/advanced-data-problems-3?ex=8

DataCamp. (n.d.). Importing flat files using pandas [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-importing-data-in-python/introduction-and-flat-files-1?ex=15

DataCamp. (n.d.). Mean, median, & mode imputations [Video file]. Retrieved from https://campus.datacamp.com/courses/dealing-with-missing-data-in-python/imputation-techniques?ex=1

DataCamp. (n.d.). Imputing using fancyimpute [Video file]. Retrieved from https://campus.datacamp.com/courses/dealing-with-missing-data-in-python/advanced-imputation-techniques?ex=1

DataCamp. (n.d.). Selecting features for better model performance [Video file]. Retrieved from https://campus.datacamp.com/courses/dimensionality-reduction-in-python/feature-selection-ii-selecting-for-model-accuracy?ex=1

Lekhana_Ganji. (2023, April 18). Machine Learning - One Hot Encoding of Datasets in Python. GeeksforGeeks. https://www.geeksforgeeks.org/ml-one-hot-encoding-of-datasets-in-python/

DataCamp. (n.d.). Working with model objects [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/predictions-and-model-objects-2?ex=5

GeeksforGeeks. (n.d.). How to Rename Columns in Pandas DataFrame. Retrieved from https://www.geeksforgeeks.org/how-to-rename-columns-in-pandas-dataframe/

DataCamp. (n.d.). Quantifying logistic regression fit [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/simple-logistic-regression-modeling?ex=10

I.
Citations for content:

DataCamp. (n.d.). Forward Stepwise Variable Selection [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/forward-stepwise-variable-selection-for-logistic-regression?ex=5

Laerd Statistics. (n.d.). Binomial Logistic Regression using SPSS Statistics. Retrieved from https://statistics.laerd.com/spss-tutorials/binomial-logistic-regression-using-spss-statistics.php

DataCamp. (n.d.). Is the data missing at random? [Video file]. Retrieved from https://campus.datacamp.com/courses/dealing-with-missing-data-in-python/does-missingness-have-a-pattern?ex=1

DataCamp. (n.d.). Measures of center [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-statistics-in-python/summary-statistics-1?ex=4

DataCamp. (n.d.). Measures of spread [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-statistics-in-python/summary-statistics-1?ex=7

Gudikandula, P. (2018, November 9). Exploratory Data Analysis (beginner), Univariate, Bivariate and Multivariate – Habberman dataset. purnasaigudikandula.medium.com. Retrieved from https://purnasaigudikandula.medium.com/exploratory-data-analysis-beginner-univariate-bivariate-and-multivariate-habberman-dataset-2365264b751

DataCamp. (n.d.). Logistic regression [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/building-logistic-regression-models?ex=5

DataCamp. (n.d.). Using the logistic regression model [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/building-logistic-regression-models?ex=9

DataCamp. (n.d.). Predictions and odds ratios [Video file]. Retrieved from https://campus.datacamp.com/courses/introduction-to-regression-with-statsmodels-in-python/simple-logistic-regression-modeling?ex=5

Jasmine Cooper
D208
Logistic Regression Modeling
October 7, 2023
Western Governor's University