D206 - Telecommunications Churn Objective Assessment

Bradley Holt

12/17/2020

Contents

Part I: Research Question

A. Question or Decision

What survey factors are most important to customers who have churned within the last month?

B. Required Variables

The original raw data set contains 52 variables (columns) and 10000 rows of data about churned customer information. This section describes each variable in the data set by referencing the data dictionary and data frame structure in Table 1.

```
#List of libraries used.
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readr)
library(knitr)
library(patchwork)
library(visdat)
library(ggplot2)
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(writexl)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

Table 1: Table Structure of churn_raw_data.csv

variable	class	first_two_values
ï..	integer	1, 2
CaseOrder	integer	1, 2
Customer_id	character	K409198, S120509
Interaction	character	aa90260b-4141-4a24-8e36-b04ce1f4f77b, fb76459f-c047-4a9d-8af9-e0f7d4ac2524
City	character	Point Baker, West Branch
State	character	AK, MI
County	character	Prince of Wales-Hyder, Ogemaw
Zip	integer	99927, 48661
Lat	double	56.251, 44.32893
Lng	double	-133.37571, -84.2408
Population	integer	38, 10446
Area	character	Urban, Urban
Timezone	character	America/Sitka, America/Detroit
Job	character	Environmental health practitioner, Programmer, multimedia
Children	integer	NA, 1
Age	integer	68, 27
Education	character	Master's Degree, Regular High School Diploma
Employment	character	Part Time, Retired
Income	double	28561.99, 21704.77
Marital	character	Widowed, Married
Gender	character	Male, Female
Churn	character	No, Yes
Outage_sec_perweek	double	6.972566093, 12.01454108
Email	integer	10, 12
Contacts	integer	0, 0
Yearly_equip_failure	integer	1, 1
Techie	character	No, Yes
Contract	character	One year, Month-to-month
Port_modem	character	Yes, No
Tablet	character	Yes, Yes
InternetService	character	Fiber Optic, Fiber Optic
Phone	character	Yes, Yes
Multiple	character	No, Yes
OnlineSecurity	character	Yes, Yes
OnlineBackup	character	Yes, No
DeviceProtection	character	No, No
TechSupport	character	No, No
StreamingTV	character	No, Yes
StreamingMovies	character	Yes, Yes
PaperlessBilling	character	Yes, Yes
PaymentMethod	character	Credit Card (automatic), Bank Transfer(automatic)
Tenure	double	6.795512947, 1.156680997
MonthlyCharge	double	171.4497621, 242.9480155
Bandwidth_GB_Year	double	904.5361102, 800.9827661
item1	integer	5, 3
item2	integer	5, 4
item3	integer	5, 3
item4	integer	3, 3
item5	integer	4, 4

variable	class	first_two_values
item6	integer	4, 3
item7	integer	3, 4
item8	integer	4, 4

blank: This variable has a data type of integer value and is an unnamed variable in the raw data.

CaseOrder: This variable labels the original order of the rows of the raw data file. This variable could prove useful if future joins of data are needed via the raw data file. As referenced in Table 1, this variable has a double data type.

Customer_id: The Customer_id variable is the primary key for identifying individual customers by using unique key identifiers. This variable is a character data type from the raw data file.

Interaction: The Interaction variable is a unique ID related to customer transactions, technical support, and sign-ups and character data type.

City: The City variable contains a vector of character data type values that describes the customer's city of residence.

State: The State variable is a vector of two-character data type that describes the customer's state of residence.

County: The Country variable contains a vector of character data type values that describes the customer's county of residence.

Zip: The Zip variable contains 5 digit integer data type that describes the customer's residential zip code.

Lat: The latitudinal GPS coordinates of the customer's residence. This variable is double data type and can be useful for visualizing data.

Lng: The longitudinal GPS coordinates of the customer's residence. This variable is double data type and can be useful for visualizing data.

Population: A integer data type format that states the population within a mile radius of the customer, based on census data.

Area: A categorical variable with character data type of area type of the customer's residential neighborhood (rural, urban, suburban)

Timezone: A categorical variable with character data type of of the customer's residence time zone.

Job: A character data type variable describing the job of the customer.

Children: A integer data type variable detailing the number of children in the customer's household.

Age: A integer data type variable of the customer's age.

Education: A categorical character data type variable of the highest degree earned by the customer.

Employment: A categorical character data type variable that describes the employment status of the customer.

Income: A double data type variable containing the customer's annual income.

Marital: A categorical character data type variable describing the customer's marital status.

Gender: A categorical character data type variable of the customer self-identified gender. (Male/Female/Prefer Not to Answer)

Churn: This variable is a categorical (yes/no) variable with character data that describes whether the customer discontinued service within the last month or not.

Outage_sec_perweek: A double data type variable that details the average number of seconds per week of system outages in the customer's neighborhood.

Email: A integer data type variable detailing the number of emails sent to the customer within the last year.

Contacts: A integer data type variable that states the number of times the customer contacted technical support.

Yearly_equip_failure: A integer data type variable that states the number of times the customer's equipment failed and needed to be reset or replaced in the past year.

Techie: A categorical (yes/no) character data type variable of whether the customer considers themselves technically inclined.

Contract: A categorical (month-to-month, one year, two-year) character data type variable of the customer's contract term.

Port_modem: A categorical (yes/no) character data type variable stating whether or not the customer has a portable modem.

Tablet: A categorical (yes/no) character data type variable stating whether or not the customer owns a tablet.

InternetService: A categorical (DSL, fiber optic, None) character data type of the customer's internet service provider.

Phone: A categorical (yes/no) character data type variable stating whether the customer has phone service.

Multiple: A categorical (yes/no) character data type variable stating whether the customer has multiple lines.

OnlineSecurity: A categorical (yes/no) character data type variable stating whether the customer has online security add-on.

OnlineBackup: A categorical (yes/no) character data type variable stating whether the customer has online backup add-on.

DeviceProtection A categorical (yes/no) character data type variable stating whether the customer has device protection add-on.

TechSupport: A categorical (yes/no) character data type variable stating whether the customer has technical support add-on.

StreamingMovies A categorical (yes/no) character data type variable stating whether the customer has streaming TV.

PaperlessBilling: A categorical (yes/no) character data type variable stating whether the customer has paperless billing.

Payment Method: A categorical character data type variable describing the customer's method of payment.

Tenure: A double data type variable describing the number of months the customer has stayed with the provider.

MonthlyCharge A double data type variable describing the average monthly charge for the customer.

Bandwidth_GB_year: A double data type variable describing the average amount of data used, in GB, in a year by the customer.

item 1-8: These variables represent the responses from an eight-question survey asking customers to rate the importance of various factors/surfaces on a scale of 1 to 8. These variables contain ordinal categorical data as there is a clear ordering (1-8) to the categories. The variable '1' represents most important and '8' represents the least important. From the raw data file, these variables are integer data type. Each item variable corresponds to the following subjects:

- item1; Timely response
- item2: Timely fixes
- item3: Timely replacements
- item4: Reliability
- item5: Options
- item6: Respectful response
- item7: Courteous exchange
- item8: Evidence of active listening

Part II: Data-Cleaning Plan

C1: Plan to Find Anomalies

Step 1: Import Data

During the first step of the data cleaning plan, the 'readr' package will be used to import the churn_raw_data.csv file. This step will also consist of completing an initial exploration of the data by getting familiar with the different variables and data types in the data set. Additionally, further exploration will be conducted to determine the size and number of observations within the data set.

Initially, most of the data set will be uploaded but unneeded columns will be skipped as determined through the initial data exploration phase.

Categorical columns with variables that can easily be determined will be converted to factor data types with the readr package. After importing, the remaining categorical variables can be converted to factor using R base.

Step 2: Identify and Remove Duplicates

The second step of the data cleaning plan consists of locating and removing any duplicate observations within the data set. The duplicated function will be used first for all rows in the data set. Second, the duplicate function will be run on any columns that should not have duplicate values. Third, if there are no duplicates returned a duplicate function will be run on a column that should have duplicate values to ensure the coding is correct.

Step 3: Manage Missing Values

The third step of the data cleaning plan will determine missing values within the data set and the approach of how missing values will be handled. The 'visdat' package will be used to visualize the missing data in the data set. The resulting visualization will be used to determine what amount of data is missing in each variable, if the values are missing at random or not, and what values to impute, if any, to replace the missing data. Before imputing missing data, a summary of the data will be developed, and once data is imputed another summary will be developed for comparison. The two summaries will then be compared to ensure the imputed values do not significantly deviate from the original distribution statistics of the data set.

Step 4: Detect and Handle Outliers

The final step of the data-cleaning plan will be to identify any outliers in specified variables and determine what action, if any, needs to be taken to address these outliers. Three potential actions can be taken depending on the nature of the outlier: Delete the outlier, replace the outlier with another value, or leave it as is. The boxplot function from the 'ggpolt2' package will be used to detect outliers in the data set.

C2: Justification of Approach

The data being assessed is a relatively small data set of 10,000 observations of churned customers. The observations are highly detailed with 52 different variables that describe each. During importation, instead of removing each variable that doesn't pertain to the specific research question stated above, most of the data set was imported. This way the entire data set can be cleaned and become useful for other data analysis research questions and specified variables can be manipulated for other uses. The chosen approach was created for a step by step design to address key elements of the data cleaning process to ensure that a high-quality data set is produced. The key elements are removing duplicate and irrelevant data, handling

missing data, and filtering outliers of the data. While moving through each step, different data sets are created to backtrack and identify quality issues or utilize different tools and functions to perform each step and analyze the results.

C3: Justification of Tools

R Language: The R language is a flexible and powerful language that is widely used in both professional and academic settings for analyzing data. Compared to other languages, the R language is less generalized and more statistics focused and assists with creating data analytic products. Being a widely used language is a benefit as code created by other users can easily be found and reused on different data sets.

Rstudio: Rstudio is an open-source software product with libraries and packages that are continually being developed and updated by the R community. The software is widely adopted by millions of users and requires no specialized training to begin using. Additionally, tools on R studio is scalable, allowing the integration of large numbers of users and large amounts of data into existing systems. ("What Makes Rstudio Different")

Tidyverse: The Tidyverse library contains several packages designed for data science used to ease the process of coding and manipulating data in R. The core packages used from Tidyverse are readr and ggplot2. Tidyverse packages have "cheat sheets" available that allows users to easily reference functions in each package.

Readr: Readr was used to simplify the process of importing the data and changing the data type of specified columns. Both these tasks were accomplished simultaneously via readr's graphical interface function. Using readr greatly reduced the time needed to import the data and convert the data types of data set columns.

ggplot2: ggplot2 is a package that creates elegant data visualizations using the components from Grammar of Graphics. With ggplot2 plots, graphs, charts, etc. can be easily created and customized to meet an audience's specific needs.

Knitr: The Knitr package is a report generation package and was used for the kable function, which is a simple table generator. The justification for using this tool was to improve the layout and looks of the knitted markup document making it easier to read.

patchwork: patchwork was used to improve the layout of the knitted markup document by adjusting the plot's output formats.

visdat: The visdat library assists in the visualization of the data. The package has commands that graphically display the date frame, what the data classes of the columns are, and what data is missing. It provides a very simple process of determining how much data is missing and where in the data frame it is specifically missing.

zoo: The zoo package provides various functions use to manipulate data in ordered index observations. The package contains different functions that can be used to fill in missing data. The specific function used from zoo was the na.locf function. This function fills in missing values with the value found in the row proceeding it.

writexl: The writexl package is used to read, write, and format excel files. The package makes it simple to export data into Excel documents.

factoextra: factoextra is a package that makes it easy to extract, explore, and visualize the output of multivariate data analyses. This is an essential item when performing primary component analysis in R.

C4: Provide the Code

Step 1: Import Data

```r
# Importing data using 'readr' package and converting data types

 churn_data_import <- read_csv("C:/Users/holtb/Data/D206 Project/churn_raw_data.csv",
     col_types = cols(X1 = col_skip(), CaseOrder = col_integer(),
         Interaction = col_skip(), Zip = col_integer(),
         Population = col_number(), Area = col_factor(levels = c("Rural",
             "Urban", "Suburban")), Age = col_integer(),
         Gender = col_factor(levels = c("Male",
             "Female", "Prefer not to answer")),
         Churn = col_factor(levels = c("Yes",
             "No")), Techie = col_factor(levels = c("Yes",
             "No")), Contract = col_factor(levels = c("Month-to-month",
             "One year", "Two Year")), Port_modem = col_factor(levels = c("Yes",
             "No")), Tablet = col_factor(levels = c("Yes",
             "No")), InternetService = col_factor(levels = c("None",
             "DSL", "Fiber Optic")), Phone = col_factor(levels = c("Yes",
             "No")), Multiple = col_factor(levels = c("Yes",
             "No")), OnlineSecurity = col_factor(levels = c("Yes",
             "No")), OnlineBackup = col_factor(levels = c("Yes",
             "No")), DeviceProtection = col_factor(levels = c("Yes",
             "No")), TechSupport = col_factor(levels = c("Yes",
             "No")), StreamingTV = col_factor(levels = c("Yes",
             "No")), StreamingMovies = col_factor(levels = c("Yes",
             "No")), PaperlessBilling = col_factor(levels = c("Yes",
             "No")), PaymentMethod = col_factor(levels = c("Electronic Check",
             "Bank Transfer(automatic)", "Mailed Check",
             "Credit Card (automatic)"))))
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```r
#Note: Warning received possibly due to an issue with readr and .csv saving. https://github.com/tidyver
```

Step 2: Identify and Remove Duplicates

```r
# Identifying any duplicate rows within the data set
duplicates = nrow(churn_data_import[duplicated(churn_data_import),])
cat(sprintf("There are %s number of rows duplicated in the dateset.", duplicates))
```

```
## There are 0 number of rows duplicated in the dateset.
```

```r
# Identifying any duplicate values in the Customer_ID variable
Customer_id_duplicates = nrow(churn_data_import[duplicated(churn_data_import$Customer_id,),])
cat(sprintf("There are %s values duplicated in the 'Customer_id' column.", Customer_id_duplicates))
```

```
## There are 0 values duplicated in the 'Customer_id' column.
```

```r
# Testing code on State variable to ensure it is correct
State_duplicates = nrow(churn_data_import[duplicated(churn_data_import$State,),])
cat(sprintf("There are %s values duplicated in the 'State' column.", State_duplicates))
```

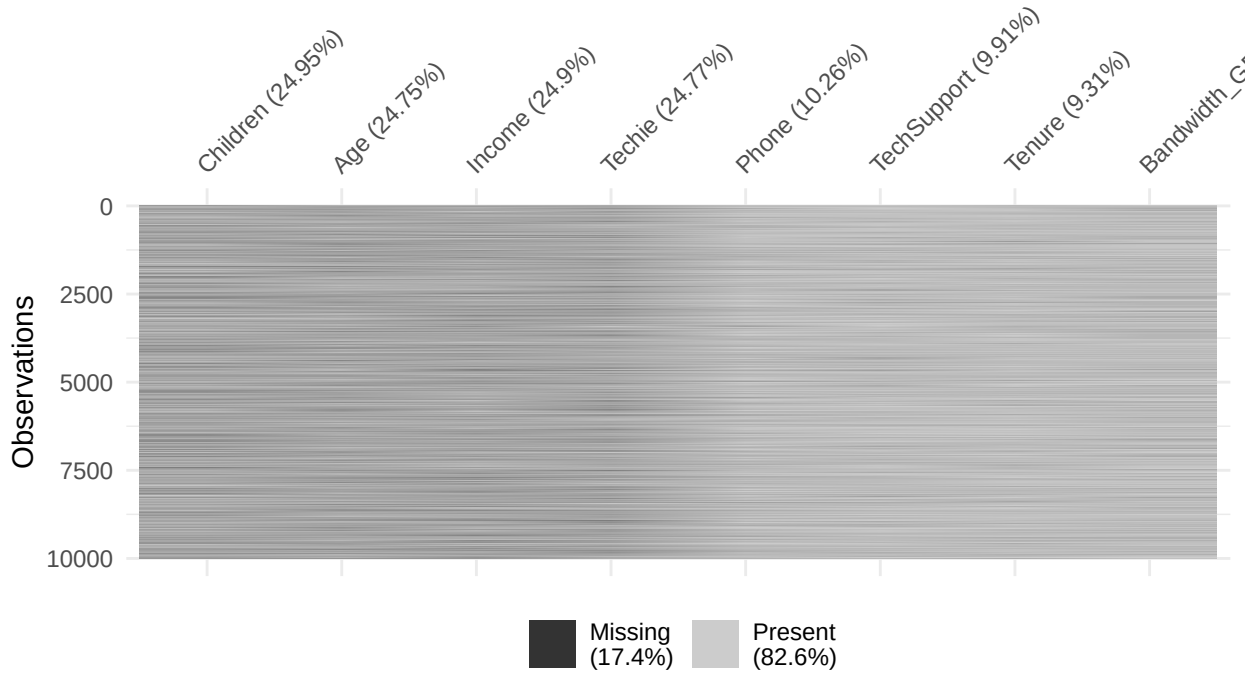
```
## There are 9948 values duplicated in the 'State' column.
```

Step 3: Manage Missing Values

```
# Using visdat and ggplot2 package to visualize missing data within the data set
vis_miss(churn_data_import) +
  theme(axis.text.x = element_text(angle = 90), plot.title = element_text(hjust = 0.5)) +
  ggtitle("Figure 1: Missing Data from churn_data_import")
```


```
# Using visdat and tidyverse package to concatenate a separate table to ease visualization
# of missing data.
churn_data_missing <-  select(churn_data_import, Children, Age, Income, Techie, Phone, TechSupport, Ten

vis_miss(churn_data_missing) +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle("Figure 2: Missing Data from Columns with NA values")
```

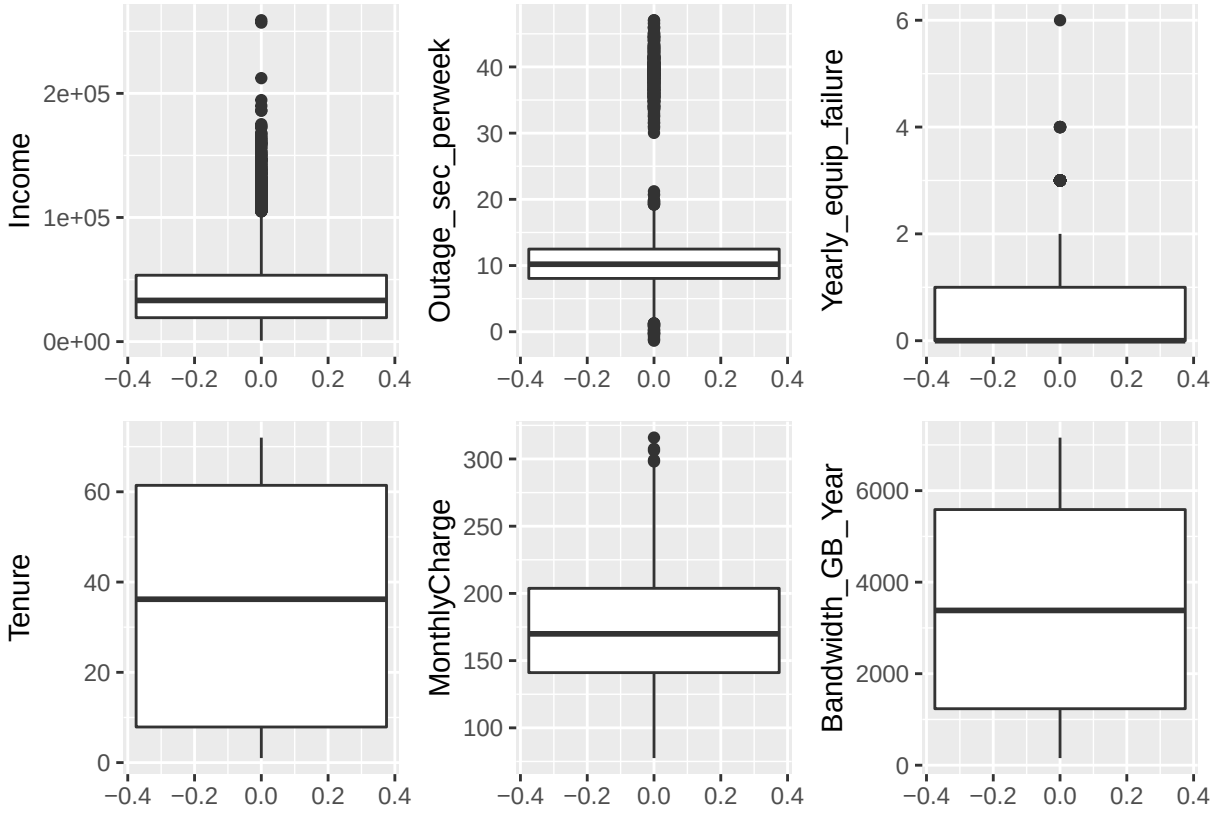
Figure 2: Missing Data from Columns with NA values



Step 4: Detect and Handle Outliers

```
p1 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = Income))

p2 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = Outage_sec_perweek))

p3 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = Yearly_equip_failure))

p4 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = Tenure))

p5 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = MonthlyCharge))

p6 <- ggplot(churn_data_import)+
  geom_boxplot(aes(y = Bandwidth_GB_Year))

p1 + p2 + p3 + p4 + p5 + p6
```

```
## Warning: Removed 2490 rows containing non-finite values (stat_boxplot).

## Warning: Removed 931 rows containing non-finite values (stat_boxplot).

## Warning: Removed 1021 rows containing non-finite values (stat_boxplot).
```



D1: Cleaning Findings

This section highlights key findings for each step during the data cleaning process.

Importing Data

- An extra column field ('X1') is included and should be deleted.
- Many numerical values in the raw data had very high precision values.
- The 'State' column contains 52 variables. After inspection, it was found the territory of Puerto Rico(PR) and the District of Columbia(DC) are included and the variables were verified.

Identify and Remove Duplicates

- No significant findings were noted for this step.

Manage Missing Values

- 2.8% of the total data set had missing values.
- The missing values were located in the 'Children', 'Age', 'Income', 'Techie', 'Phone', 'TechSupport', 'Tenure' and 'Bandwidth_GB_Year' columns.
- 17.4% of data was missing from the total data describe in the columns above.
- Almost a quarter of the data is missing from the 'Children', 'Age', 'Income', and 'Techie' columns.
- The 'Tenure' column, which is directly related to the research question, is missing 9.31% of data.

Detect and Handle Outliers

- 'Income' and 'Outage_sec_perweek', and 'Yearly_equip_failure' data columns are right-skewed.
- The 'Income' column being right-skewed and having numerous outliers matches the typical United States household income distribution.
- The 'Outage_sec_perweek' data contains numerous outliers between 30 and 60 seconds but are likely not erroneous.
- The 'Yearly_equip_failure' contains 3 outliers. These outliers should be simple to confirm and should be verified for accuracy.
- Negative values were found in the "Outage_sec_perweek" column, but they should all be positive.

D2: Justification of Mitigation Methods

Importing Data

Initial inspection of the data set revealed that the file is in .csv format. This characteristic of the data set allowed the use of the 'readr' package and the read_csv function. The 'readr' package was used due to the simplicity of importing data, removing variables, and changing data types of variables as needed. Columns that were obviously not needed such as the 'X1' and 'Interaction' were skipped during importation to reduce data as these fields were irrelevant to analyzing the data. Although many of the other columns may not be relevant to the specific research question, leaving other variables intact will allow an opportunity to explore the data in the event the research question changes. Numerical data was rounded down to 2 decimal places as the precision provided in the original data set was not needed. The same reasoning applied to the income variable which was rounded to the nearest whole number. Finally, variables with categorical character data were converted to factors. Converting categorical data to factors improves the efficiency of the data set as the character values are only stored once and the data in the variable is stored as a vector of integers.("Factor Variables", UCLA)

Identifying Duplicate Values

A simple function was written using R base to complete this task. Next, this function was also run on any columns that should not have duplicate values. The Customer_ID column was identified because each value should be unique. Finally, a test was conducted on a column that has known duplicates to test the code and ensure the function performed correctly. After reviewing the results, no known duplicated values were found in the data set that needed to be addressed.

Managing Missing Values

The 'visdat' library was used for the process of finding missing values. While the summary function or a 'isna' function could have been used to identify where missing data was, the 'visdat' package was simple to use and provided a better visual overview of what and where data was missing. To fill in the missing data the zoo package's na.locf function was used to fill in missing data. This function filled in missing data with the value that proceeded the NA value. Analysis was conducted using ggplot2 on the data set before and after filling to ensure the method of filling NA values did not significantly change the probability density of the variables. While bias is likely in the variables with 25% of data missing, a multiple imputation method was not used. The variable with large amounts of data missing is not significant to the research question.

Detecting and Handling Outliers

After reviewing the summary data set, initial outlier analysis was performed on columns in which values seemed abnormal or could be prone to outliers. Boxplots were created for the columns "Income", "Outage_sec_perweek", "Yearly_equip_failure", "Tenure", "Monthly_charge" and "bandwidth_GB_Year".

This method was used because the visualization of the distributions made it easy to detect outliers that may need to be addressed. During this phase, it was also noted that the "Outage_sec_perweek" column contained negative values. These values are obviously erroneous and most likely should be positive numbers or zero. Follow-up should be conducted to confirm which assumption is correct. These numbers will be converted to "0" as there are only 11 values that need to be corrected and all values are low.

D3: Summary of the Outcomes

Importing Data

- Categorical data was converted to factors
- Numerical data precision was reduced to, at most, 2 decimal places.
- 'X1' and 'Interaction' columns were removed from the data set.

Managing Missing Values

- Missing data in the data set was filled using visdat's na.locf function.
- Analysis was conducted to ensure distribution probability was not significantly altered.

Detecting and Handling Outliers

- Negative values in "Outage_sec_perweek" were revalued to "0".
- All other outliers were not modified. No other outliers seemed erroneous.
- Further analysis should be conducted to determine if there are any correlations between the outages outliers and specific locations.

D4: Mitigation Code

```
#Using Base R to convert remaining categorical variables to factors
factor_cols <- c("City", "State", "County", "Timezone", "Education", "Employment", "Marital" )
churn_data_import[factor_cols] <-  lapply(churn_data_import[factor_cols], factor)

#Using Base R to round decimals to 2 digits for variables Tenure, MonthlyCharge and Bandwidth_GB_Year)
round_cols <-  c("Outage_sec_perweek","Tenure","MonthlyCharge","Bandwidth_GB_Year")
churn_data_import[round_cols] <- lapply(churn_data_import[round_cols], round, 2)

#Using base R to remove decimals from Income variable
churn_data_import$Income <- round(churn_data_import$Income)
```

```
churn_data_import$Outage_sec_perweek <- ifelse(churn_data_import$Outage_sec_perweek < 0, 0, churn_data_
```

```
#Summary of churn_data_missing prior to filling missing data
summary(churn_data_missing)
```

```
##     Children           Age             Income          Techie      Phone
##  Min.   : 0.000   Min.   :18.00   Min.   :   740.7   Yes :1257   Yes :8128
##  1st Qu.: 0.000   1st Qu.:35.00   1st Qu.: 19285.5   No  :6266   No  : 846
##  Median : 1.000   Median :53.00   Median : 33186.8   NA's:2477   NA's:1026
##  Mean   : 2.096   Mean   :53.28   Mean   : 39936.8
```

```
##  3rd Qu.: 3.000   3rd Qu.:71.00   3rd Qu.: 53472.4
##  Max.   :10.000   Max.   :89.00   Max.   :258900.7
##  NA's   :2495     NA's   :2475    NA's   :2490
##  TechSupport     Tenure       Bandwidth_GB_Year
##  Yes :3374   Min.   : 1.00   Min.   : 155.5
##  No  :5635   1st Qu.: 7.89   1st Qu.:1234.1
##  NA's: 991   Median :36.20   Median :3382.4
##              Mean   :34.50   Mean   :3398.8
##              3rd Qu.:61.43   3rd Qu.:5587.1
##              Max.   :72.00   Max.   :7159.0
##              NA's   :931     NA's   :1021
```

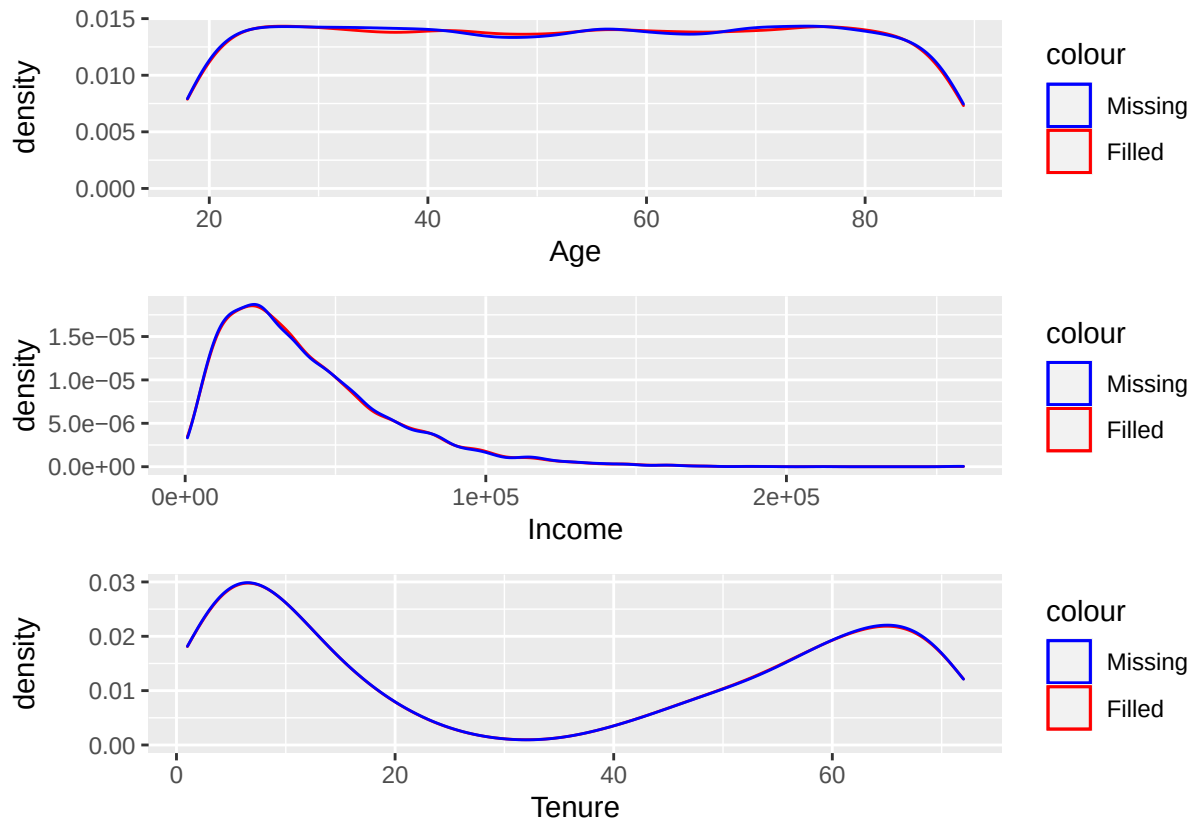
```
#Use of zoo package to fill missing data within the data set
churn_data_missingfilled = na.locf(churn_data_missing, na.rm = FALSE) %>%
  replace(is.na(.), 1)
```

```
#Summary of churn_data_missingfilled after filling missing data
summary(churn_data_missingfilled)
```

```
##     Children           Age             Income          Techie     Phone
##  Min.   : 0.000   Min.   :18.00   Min.   :   740.7   Yes:1688   Yes:9063
##  1st Qu.: 0.000   1st Qu.:35.00   1st Qu.: 19167.7   No :8312   No : 937
##  Median : 1.000   Median :53.00   Median : 33197.7
##  Mean   : 2.119   Mean   :53.27   Mean   : 39922.3
##  3rd Qu.: 3.000   3rd Qu.:71.00   3rd Qu.: 53711.4
##  Max.   :10.000   Max.   :89.00   Max.   :258900.7
##  TechSupport     Tenure       Bandwidth_GB_Year
##  Yes:3708    Min.   : 1.000   Min.   : 155.5
##  No :6292    1st Qu.: 7.916   1st Qu.:1232.6
##              Median :35.431   Median :3287.4
##              Mean   :34.497   Mean   :3394.5
##              3rd Qu.:61.439   3rd Qu.:5584.3
##              Max.   :71.999   Max.   :7159.0
```

```
#Using ggplot to compare distribution density from before and after filling missing data.
d1 <- ggplot() +
  geom_density(mapping = aes(x= churn_data_missing$Age, color="red"))+
  geom_density(mapping = aes(x= churn_data_missingfilled$Age, color="blue"))+
  labs(x = "Age")+
  scale_color_manual(labels= c("Missing", "Filled"), values = c("blue", "red"))

d2 <- ggplot() +
  geom_density(mapping = aes(x=churn_data_missing$Income, color="red"))+
  geom_density(mapping = aes(x=churn_data_missingfilled$Income, color="blue"))+
  labs(x = "Income")+
  scale_color_manual(labels= c("Missing", "Filled"), values = c("blue", "red"))

d3 <- ggplot() +
  geom_density(mapping = aes(x=churn_data_missing$Tenure, color="red"))+
  geom_density(mapping = aes(x=churn_data_missingfilled$Tenure, color ="blue"))+
  labs(x = "Tenure")+
  scale_color_manual(labels= c("Missing", "Filled"), values = c("blue", "red"))

d1 + d2 + d3 + plot_layout(ncol=1)
```

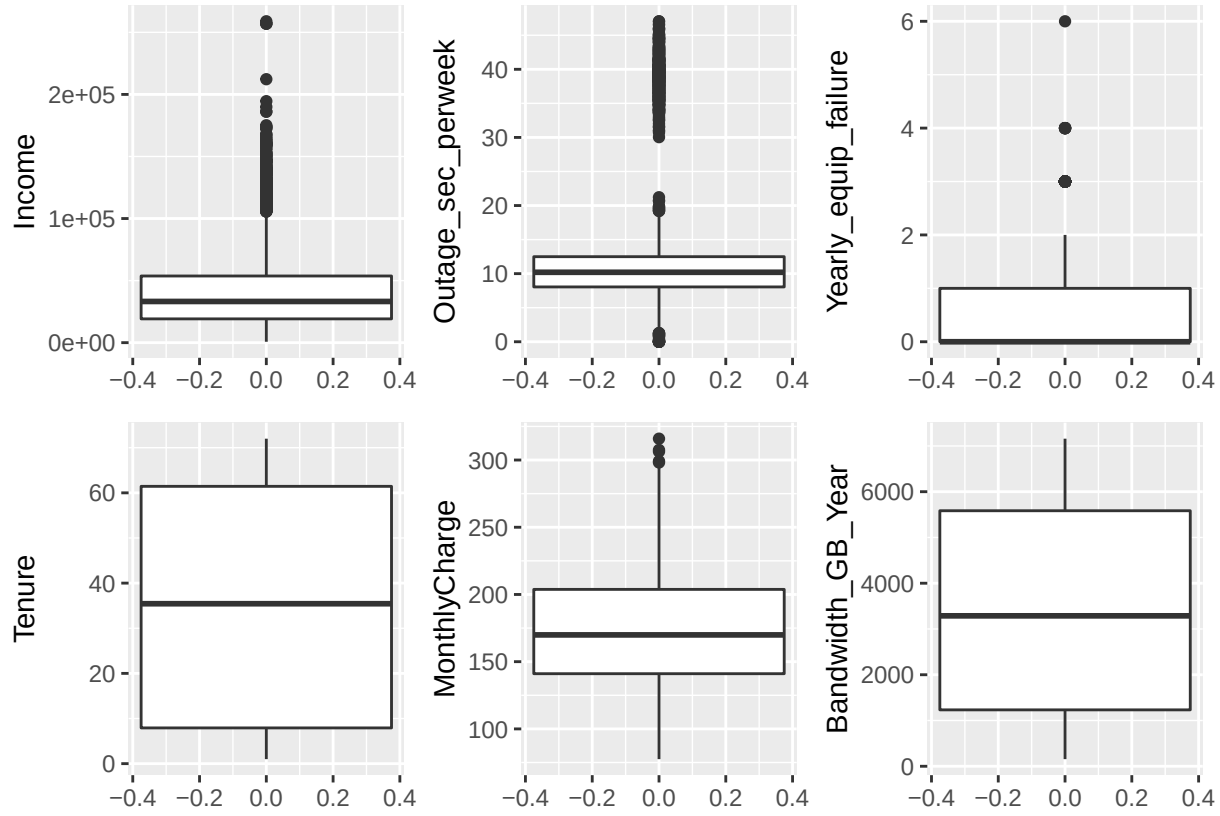
```
## Warning: Removed 2475 rows containing non-finite values (stat_density).

## Warning: Removed 2490 rows containing non-finite values (stat_density).

## Warning: Removed 931 rows containing non-finite values (stat_density).
```




```
#Filling missing data in original data set.
churn_data_filled = (na.locf(churn_data_import, na.rm = FALSE))%>%
  replace(is.na(.), 1)
```

```
p1 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = Income))

p2 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = Outage_sec_perweek))

p3 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = Yearly_equip_failure))

p4 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = Tenure))
```

```
p5 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = MonthlyCharge))

p6 <- ggplot(churn_data_filled)+
  geom_boxplot(aes(y = Bandwidth_GB_Year))

p1 + p2 + p3 + p4 + p5 + p6
```



D5: Clean Data

```
churn_data_clean = churn_data_filled

write_xlsx(churn_data_clean, "C:/Users/holtb/Desktop/D206 Project/churn_data_clean.xlsx")
```

D6: Limitations

One limitation is the method used while cleaning the data. Creating a new table for each step works for relatively small data sets, however if this method were used on data sets with millions of observations it would be very time-consuming and a waste of resources and storage. A better way to approach data cleaning on a large data set would be to create a smaller table using a random sampling of the observations and use that data to create code and manipulate the data. Afterward the data cleaning model and coding can be used on the actual data set.

Another limitation is the accuracy of the data. The method used to fill the missing values may not have significantly changed the overall distribution of the data however, in some variables, a quarter of the data was filled in. Variables with more than 10% of missing data are likely to be biased.(Dong & Peng, 2013) This should be taken into account when using these variables to perform analysis. If less biased results are required, a multiple imputation method can be used such as using the 'Mice' library.

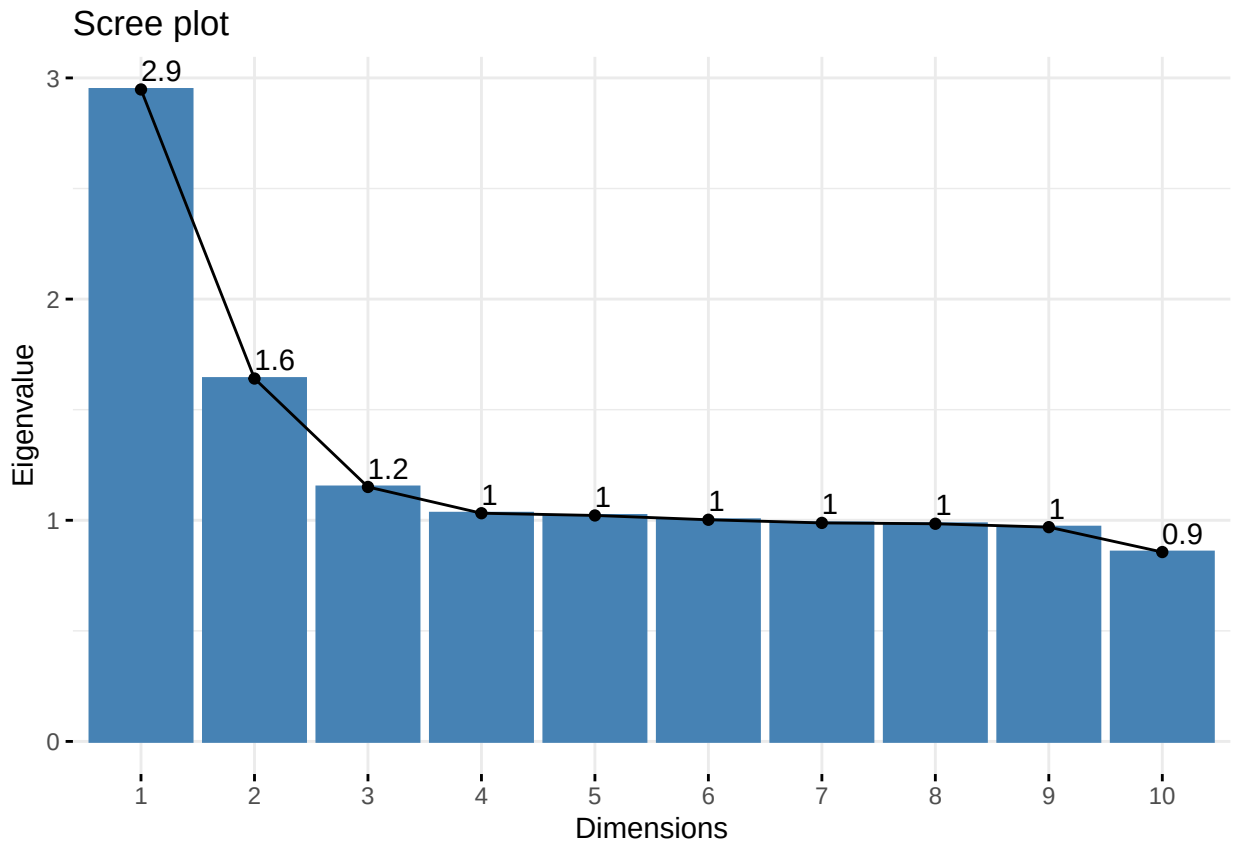
D7: Impact of the Limitations

The limitation impact on the research question is minimal. The majority of the variables will not be used when performing analysis for the question. A separate smaller table can be created by extracting the specific variables needed to perform the analysis. The columns that would most likely be needed are the "Churn" column and "items 1-8" columns. Neither of these columns had missing data imputed into them so bias caused by the imputation will not be a factor.

E1: Principal Components

```
churn_data_clean.pca <- prcomp(churn_data_clean[,c(9,13,14,17,21,24,41:50)],center = TRUE, scale = TRUE)
```

```
fviz_eig(churn_data_clean.pca, choice = "eigenvalue", addlabels=TRUE)
```




```
factanal(churn_data_clean[,c(9,13,14,17,21,24,41:50)], factors = 6)
```

```
##
## Call:
## factanal(x = churn_data_clean[, c(9, 13, 14, 17, 21, 24, 41:50)],     factors = 6)
##
## Uniquenesses:
##           Population             Children                  Age
##                1.000                0.999                0.005
##               Income   Outage_sec_perweek Yearly_equip_failure
##                0.999                0.962                0.005
##        MonthlyCharge    Bandwidth_GB_Year                item1
##                0.532                0.992                0.262
##                item2                item3                item4
##                0.403                0.547                0.005
##                item5                item6                item7
##                0.579                0.547                0.676
##                item8
##                0.799
##
## Loadings:
##                      Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
## Population
## Children
## Age                                          0.969  -0.200  -0.123
## Income
## Outage_sec_perweek                                           0.193
## Yearly_equip_failure                 0.996
## MonthlyCharge                                                0.676
## Bandwidth_GB_Year
## item1                 0.853
## item2                 0.768
## item3                 0.668
## item4                         0.328           0.204   0.917
## item5                        -0.593                  -0.251
## item6                 0.415   0.521
## item7                 0.344   0.446
## item8                 0.302   0.325
##
##                Factor1 Factor2 Factor3 Factor4 Factor5 Factor6
## SS loadings      2.153   1.059   0.995   0.993   0.966   0.523
## Proportion Var   0.135   0.066   0.062   0.062   0.060   0.033
## Cumulative Var   0.135   0.201   0.263   0.325   0.385   0.418
##
## Test of the hypothesis that 6 factors are sufficient.
## The chi square statistic is 37 on 39 degrees of freedom.
## The p-value is 0.561
```

E2: Criteria Used

After observing the results of the scree plot and factnal functions, the results show the first two principle components have eigenvalues above 1. The first principle component has items 1 through 3 and items 6 through 8 as the variables with items 1 through 3 having the highest weight. The second principle component has items 4 through 8 as its variables with items 5 and 6 having the highest weight. While the third principle component has an eigenvalue above 1, it only contains one variable.

E3: Benefits

Principle analysis combines variables into groups to reduce data and simplify data analysis. In the above PCA, items 1-8 are responses by customers based of a survey they were given. These responses were grouped together between PC1 and PC2 so that these two principle components can be used to conduct analysis, instead of each of the variables in the components separately.

F: Video

Link to Video

G: Sources for Third-Party Code

R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686

Hadley Wickham and Jim Hester (2020). readr: Read Rectangular Text Data. R package version 1.4.0. https://CRAN.R-project.org/package=readr

Yihui Xie (2020). knitr: A General-Purpose Package for Dynamic Report Generation in R. R package version 1.30.

Thomas Lin Pedersen (2020). patchwork: The Composer of Plots. R package version 1.1.1. https://CRAN.R-project.org/package=patchwork

Tierney N (2017). "visdat: Visualising Whole Data Frames." https://doi.org/10.21105/joss.00355

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Achim Zeileis and Gabor Grothendieck (2005). zoo: S3 Infrastructure for Regular and Irregular Time Series. Journal of Statistical Software, 14(6), 1-27. doi:10.18637/jss.v014.i06

Jeroen Ooms (2020). writexl: Export Data Frames to Excel 'xlsx' Format. R package version 1.3.1. https://CRAN.R-project.org/package=writexl

Alboukadel Kassambara and Fabian Mundt (2020). factoextra: Extract and Visualize the Results of Multivariate Data Analyses. R package version 1.0.7. https://CRAN.R-project.org/package=factoextra

H: Sources

Dong, Y., & Peng, C. (2013, May 14). Principled missing data methods for researchers. Retrieved February 01, 2021, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3701793/#:~:text=Proportion%20of%20missing%20data,-The%20proportion%20of&text=For%20example%2C%20Schafer%20(%201999%20),10%25%20of%20data%20are%20missing.

Factor Variables, UCLA: Statistical Consulting Group. from https://stats.idre.ucla.edu/r/modules/factor-variables/ (accessed January 26, 2021).

What Makes RStudio Different? (n.d.). Retrieved January 29, 2021, from https://rstudio.com/about/what-makes-rstudio-different/