­

**Churn Data Cleaning**

Stephen E. Porter

College of Information Technology, Western Governors University

Dr. Keiona Middleton

August 11, 2021

**Churn Data Cleaning**

In this paper, I will use a data set containing customer data from a fictional telecommunications company. The primary purpose is to explain the process for cleaning the data in the data set by identifying and mitigating missing values and then identifying and mitigating outliers as needed using Python and its relevant packages and libraries to enhance its capabilities.

# Part I: Research Question

What factor (or factors) are common among customers who have cancelled their service, and, using this information, which current customers appear likely to cancel their service?

The data set consists of 10000 rows of customer data which fall into several variable groupings:

* **Location** (City, State, County, Zip, Lat, Lng, Population, Area, Timezone)
* **Personal** (Customer\_id, Interaction, Job, Children, Age, Education, Income, Marital, Gender, Techie, Email, Tenure)
* **Issues** (Outage\_sec\_perweek, Contacts, Yearly\_equip\_failure,)
* **Services & Billing** (Contract, Port\_modem, Tablet, InternetService, Phone, Multiple, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, PaperlessBilling, PaymentMethod, MonthlyCharge, Bandwidth\_GB\_Year)
* **Survey** (Item1: Timely response, Item 2: Timely fixes, Item 3: Timely replacements, Item 4: Reliability, Item 5: Options, Item 6: Respectful response, Item 7: Courteous exchange, Item 8: Evidence of active listening)

# Part II: Data Cleaning Plan

## C1. Plan to Find Anomalies

I have followed a proven strategy of identifying and addressing null values followed by identifying and addressing outliers as Dr. Middleton described in her D206 Webinar 2 – Getting Started with Missing Data and Outliers.

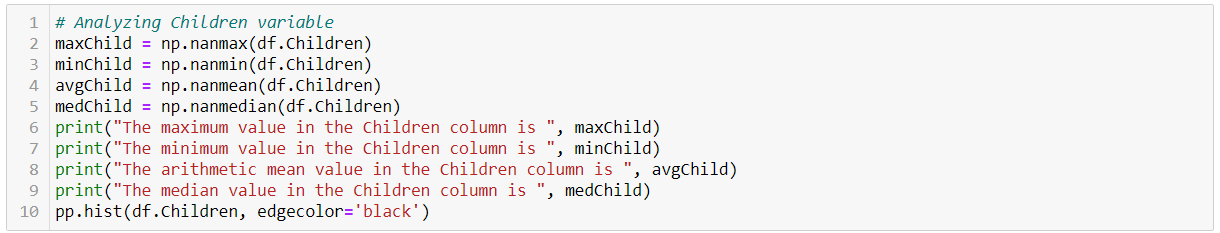
I plan to use Python and its relevant packages to clean these data. The following packages/libraries will be used: pandas (for reading and exporting the file), numpy (for handling null values), scipy.stats (for statistical analysis like z-scores), matplotlib.pyplot (to create and view graphs), seaborn (to create heatmaps), and PCA from sklearn.decomposition (to perform Principal Component Analysis). These tools work just as well as R for cleaning data, and in some cases R is more concise in terms of lines of code; however, I chose Python because I have prior experience with the language and therefore am more comfortable using it.

I will read the file and drop any duplicate rows (checking the shape before and after to determine if there was a significant change in the data set). Once read, I will use **.info()** to review the variables, types, and number of missing values. I will rename the columns containing the survey data (originally named item1, item2, …, item8) to make them more descriptive. I will visualize the null values with a heatmap and then drop any rows where a significant portion of the values are missing.

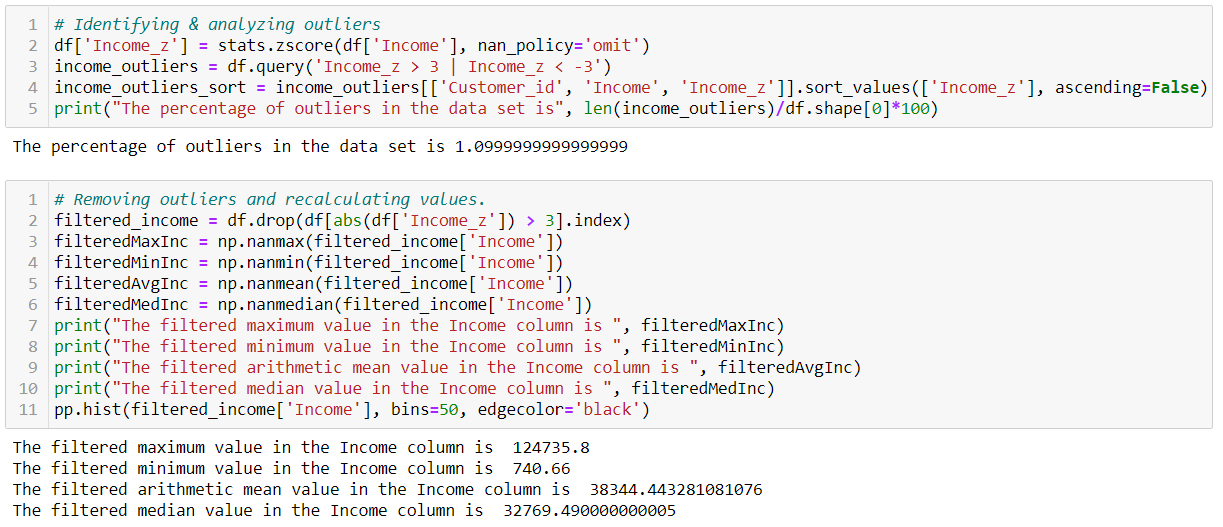
Once these initial items are completed, I will target the columns containing null values one at a time and determine if the missing values should be imputed or if the column should be dropped entirely. For numerical values, I will determine values such as maximum, minimum, measures of central tendency, and/or z-scores as needed. If there are significant outliers that could affect the measures of central tendency, I will remove the outliers and repeat the process on the remaining values. Following this analysis, I plan to impute using the median to minimize the effect of outliers on the data set. For categorical values I plan to impute using the mode.

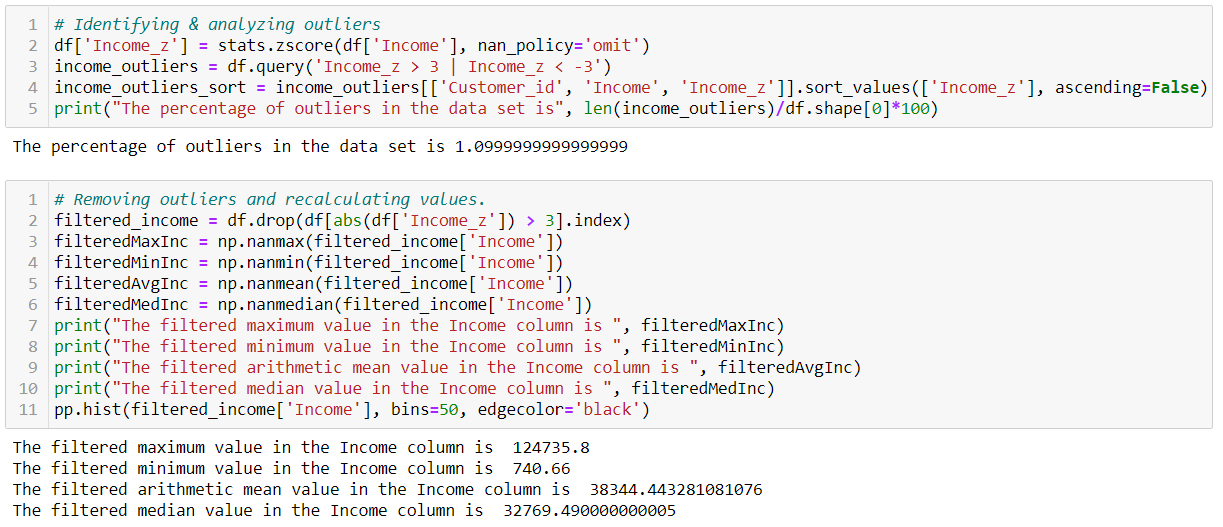
After removing null values from the data set, I will review the remaining columns to check for recording errors that could lead to drawing incorrect conclusions during analysis. For numerical values, I will visualize using histograms and/or boxplots to check for outliers. Then, if further analysis would be useful, I will find values like maximum, minimum, measures of central tendency, and/or z-scores as needed. For categorical values, I will check for spelling errors using **.unique()** and, for those with limited options (i.e.: yes/no, etc.), I will use **.value\_counts()** to quickly identify any values that should not be included.

For numerical fields, I will use the following code as a general template to determine the outliers and measures of central tendency. Visualizing these data with a histogram will provide greater context for analysis.



Should there be significant outliers, I will use z-scores to identify them, remove them, and recalculate the values using the following code as a template (Larose, 2019).





For categorical values, I will use df[‘column’]**.unique()** and df[‘column’]**.value\_counts()** and review the outputs manually (replacing “column” with the appropriate column name).

For the full code, please see the attached file titled *D206\_Churn\_Data\_Cleaning.pdf.*

# Part III: Data Cleaning

## Summary of the Process

The cleaning process followed the plan outlined above closely. During basic preparation, I found there were no duplicate rows or rows containing a significant number of missing values (>75%). The column titled “Unnamed: 0” was redundant with the column titled “Case Order” so the former was dropped, and the “Case Order” column was set as the index for the data frame.

The columns containing null values were Children, Age, Income, Techie, Phone, TechSupport, Tenure, and Bandwidth\_GB\_Year, and I addressed them in that order.

For the Children column, I found the maximum, minimum, arithmetic mean, and median, and created a histogram to visualize the data. While the histogram was skewed left, there were no unreasonable outliers (which I confirmed with the maximum and minimum values). I chose to impute the missing values using the median (1) to avoid any additional effect from the large values.

For the Age column, I followed a similar process. The age range was reasonable, and the histogram was more balanced. I chose to impute the missing values with the median (53) to remain consistent with my previous step.

For the Income column, the range was reasonable; however, the histogram revealed a collection of outliers with a significant separation from the rest of the values. I then found the z-scores for the income column to identify and filter out the outliers. I then created a new data frame named filtered\_income where I dropped the rows with a z-score above 3 or below -3. Using the filtered\_income data frame, I found the maximum, minimum, arithmetic mean, and median, and created a histogram. The maximum value decreased by nearly 134,200, the mean by nearly 1600, and the median by nearly 420. I chose to impute the missing values using the median from the filtered\_income data frame to minimize any additional affect the outliers may have.

The Techie column was the first non-numerical column with missing values. The entries in the column are self-reported and, therefore, it is likely the definition was not evenly applied. Conclusions drawn from this column could be misleading and therefore was dropped. If this field is deemed relevant to answering the question at the beginning of this paper, then there should be another attempt to collect this information from the customers.

The Phone and Multiple columns are related – if a customer has multiple phone lines, they will have to have phone service. Because the Multiple column contains no missing values, I used it to make the first imputation for the Phone column. I found the rows with a null value in the Phone column and a “Yes” in the Multiple column, and then filtered the Customer\_id values. I used those values to locate the proper rows in the data frame to impute “Yes” in the Phone column. The remaining missing values were replaced with the mode.

For the TechSupport column, I viewed the value counts and imputed using the mode.

For the Tenure column, I followed the same numeric analysis as the Children, Age, and Income columns. The range of values was reasonable; however, the histogram appeared to show a bimodal relationship. This could imply that we are losing customers at around the 2-3 year mark (perhaps after the new-customer discounts have expired) or that there is a concentration of missing values for customers in this tenure range. I chose to impute the missing values using the median (36.19603) to remain consistent with previous imputations.

The final column with null values was Bandwidth\_GB\_Year. I followed the same process for the other numeric columns. The range of values was reasonable, and I chose to replace the missing values with the median (3382.424).

Once all null values had been removed, I reviewed the remaining columns for potential recording errors. I reviewed them each in order; however, in my summary here I will group them differently.

Many of the columns were easily determined to be reasonable using **.value\_counts()** as they contained only a few potential values to check. These columns were State, Timezone, Education, Employment, Marital, Gender, Churn, Contract, Port\_modem, Tablet, InternetService, Multiple, OnlineSecurity, OnlineBackup, DeviceProtection, StreamingTV, StreamingMovies, PaperlessBilling, PaymentMethod, Response, Fix, Replacement, Reliability, Options, Respectful, Courteous, and Listening. All outputs were as expected.

The columns Zip, Lat, and Lng were verified by finding the maximum and minimum value in each ensuring they fell within the known ranges for zip codes, latitude, and longitude values. The values in each column were confirmed to fall within the ranges.

The numeric columns of Email, Contacts, Yearly\_equip\_failure, and MonthlyCharge were analyzed using histograms. Each histogram showed a reasonable spread of data with no apparent outliers.

There were two columns that could not be reasonably reviewed for accuracy with the resources available to me: City and County. Using the **.unique()** function, they contain 6058 and 1620 unique values, respectively. Without master lists to compare each column, each field would have to be checked manually. Because of this, no changes were made to the values in either column.

The Population column was reviewed using a histogram and there were apparent outliers. I filtered only the City, State, and Population columns and sorted by Population (from high to low). Viewing the top 20 fields revealed cities in heavily populated areas of the country, so no changes were made to the values in the Population column.

For the Outage\_sec\_perweek column, I performed the same analysis as for other numeric columns. The minimum value and histogram revealed that there were negative values which are not reasonable. I determined that there were only eleven (11) negative values which I was able to review individually. Because the largest negative value was only -1.348571, I chose to replace all negative values with zero (0). The negative values are likely due to a recording error. By replacing them with zero, there will be minimal effect on analysis done on that column.

The one remaining column is Job which has 639 unique values. These were self-reported, not standardized to a type of field, and may no longer be accurate as the field has not been updated since the customer first signed up for service. Because of the potential for flawed conclusions to be drawn from this column, I chose to drop it.

## Principal Component Analysis

I began by selecting all numeric columns and creating a new list with their names in order to create a data frame that contained only those columns. I then normalized the values in the columns as described in the course text. Because I did not manually select each numeric column, I used a while-loop to create a list to hold the names of columns PC1, PC2, and so on. After viewing the scree plot, I used a for-loop to count the number of principal components that had an eigenvalue of greater than one (>1) and found there to be 8. I then printed the loadings for PC1 through PC8 and found them grouped as follows:

* PC1: **Customer Service** (Response, Fix, Replacement, Respectful, Courteous, Listening)
* PC2: **Location 1** (Zip, Lng)
* PC3: **Tenure/Bandwidth** (Tenure, Bandwidth\_GB\_Year)
* PC4: **Services** (Reliability, Options)
* PC5: **Location 2** (Lat, Population)
* PC6: **Family/Support** (Children, Age, Contacts)
* PC7: **Emails/Equipment** (Email, Yearly\_equip\_failure)
* PC8: **Finances** (Income, MonthlyCharge)

What can be drawn from the PCA is that providing excellent customer service is clearly the most important component of retaining customers as the columns listed under PC1 all had positive eigenvalues greater than 0.3 with Response (or Timely Response) being most important at 0.45 followed closely by Timely Fix (0.43), Timely Replacement (0.40), and Respectful Response (0.40).

# Part IV: Supporting Documents

Please find the Panapto recording with my submission, and the references below.

**References**

Larose, C. D., & Larose, D. T. (2019). *Data science using Python and R.* John Wiley & Sons

<https://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=nlebk&AN=2091371&site=eds-live&scope=site&custid=ns017578&ebv=EB&ppid=pp_30>

Lianne & Justin @ Just into Data. (2021, May 3). *Data cleaning in Python: The Ultimate guide (2020)*. Medium. <https://towardsdatascience.com/data-cleaning-in-python-the-ultimate-guide-2020-c63b88bf0a0d>

Middelton, Keiona (2021). *Webinar 2 – Getting Started with Missing Data and Outliers*. <https://wgu.webex.com/wgu/ldr.php?RCID=a9a4d96aed5946d6952a26a89332ce21>

The Matplotlib Development Team (2021). *Visualization with python¶*. Matplotlib. (n.d.). <https://matplotlib.org/stable/index.html>

The NumPy Community (2021). *API REFERENCE*. Statistics - NumPy v1.21 Manual. (n.d.). <https://numpy.org/doc/stable/reference/routines.statistics.html>

The Pandas Development Team (2021). *API REFERENCE*. DataFrame - pandas 1.3.1 documentation. (n.d.). <https://pandas.pydata.org/docs/reference/frame.html>

The SciPy Community (2021). *API REFERENCE*. Statistical functions (scipy.stats) - SciPy v1.7.1 Manual. (n.d.). <https://docs.scipy.org/doc/scipy/reference/stats.html>

szli (2019, August 11) *How do I select rows from a Dataframe based on column values?* Stack Overflow. <https://stackoverflow.com/questions/17071871/how-do-i-select-rows-from-a-dataframe-based-on-column-values>

Waskom, Michael (2021). *API REFERENCE*. API reference - seaborn 0.11.1 documentation. (n.d.). <https://seaborn.pydata.org/api.html>