D206 – DATA CLEANING

Performance Assessment Task

WGU - MSDA

Data Cleaning Assessment using the Churn Dataset

Richard Flores
rflo147@wgu.edu
Student ID: 006771163

Table of Contents

Part I

A. Research Question

For this assignment, I have opted in favor of using the Telecommunications Churn Database. As stated in the databases' accompanying file Data Cleaning Churn Data Consideration and Dictionary, "Customer 'churn' is defined as the percentage of customers who stopped using a provider's product or service". The objective of this assignment is to provide meaningful and clean data to assess telecommunications organizational need of the understanding of churn as the document goes on to further state, "It costs 10 times more to acquire a new customer than to retain an existing one." (Larose & Larose, 2019) The raw data used in this assignment will come from the provided file 'churn_raw_data.csv'. The question answered in this assignment will be:

1. What are the characteristics of customers who choose to continue telecommunication services with the company and what are common characteristics of customers who terminate services and contribute to churn?

B. Required Variables and Examples

To fully describe the data used in this assignment I will use both the churn_raw_data.csv and Data Cleaning Churn Data Consideration and Dictionary files provided.

The churn_raw_data.csv file contains data formatted into 50 columns and 10,000 rows. The rows are defined as follows:

CaseOrder, Customer_id, Interaction, City, State, County, Zip, Lat, Lng, Population, Area, Timezone, Job, Children, Age, Education, Employment, Income, Marital, Gender, Churn, Outage_sec_perweek, Email, Contacts, Yearly_equip_failure, Techie, Contract, Port_modem, Tablet, InternetService, Phone, Multiple, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, PaperlessBilling, PaymentMethod, Tenure, MonthlyCharge, Bandwidth_GB_Year, item1, item2, item3, item4, item5, item6, item7, item8.

As previously mentioned, the dataset contains 10,000 records of customer data across a variety of demographic features. Relevant to the question in Part A is whether a customer has continued or disconnected service recently which constitutes "churn".

Variables (Independent) and/or predictors that may assist in correlating a relationship with the variable (dependent) of "churn" include (Larose & Larose, 2019):

1. The services a customer subscribes to i.e tech support add-on, multiple phone lines, streaming media
2. Subscriber account information including a customer's length of tenure, payment method, and data usage
3. Subscriber demographic information i.e. income, martial status, gender
4. Lastly the dataset includes eight variables representing customer responses to company services and features.

The data includes both numerical data i.e. income, population, age and also categorical data such as 'Yes' or 'No' responses.

Part II

C1. Data Cleaning Plan

To clean the churn_raw_data.csv data several steps and techniques will need to be used.

Initially, to begin cleaning the data, several preliminary techniques will be used including:

Verifying the variable types – For analysis to function correctly we need to verify the variable type of each column. To ensure functionality and prevent errors and unexpected outcomes the data types must be verified (Larose & Larose, 2019).

Displaying unique values – Observing and interpreting unique values in a column can help understand the range of the dataset (Larose & Larose, 2019).

Detecting Duplicate Values – Detecting duplicate rows is crucial for the integrity of churn analysis. This technique will flag the secondary duplicate row which can then be removed (Larose & Larose, 2019).

After the initial cleaning, several additional techniques will need to be used to identify and correct outliers including:

Displaying missing values – This technique will help to locate columns with missing data and show the number of values missing in every column. We can then consider imputing or removing the rows with missing data (Larose & Larose, 2019).

Identifying standard deviation – Information from standard deviation can help during data cleaning to identify locations of outliers. Standard deviation will give information on how far an individual value falls from its mean value (Larose & Larose, 2019).

Detecting duplicate rows – Identifying duplicate rows is essential for ensuring the integrity of data. Python can be used to identify and flag the duplicate rows which can then be removed (Larose & Larose, 2019).

Defining outliers – By defining criteria to measure z-scores in variables we can use this technique to identify outliers for consideration. In this assessment, the z-score will be measured as greater than 3 or less than -3. This is accomplished by subtracting each value in the column by its mean and then diving by its standard deviation (Larose & Larose, 2019).

C2. Characteristics and Approach

Verifying the variable types, we can conclude the following relations:

```
CaseOrder                  int64        Contract                object
Customer_id               object        Port_modem              object
Interaction               object        Tablet                  object
City                      object        InternetService         object
State                     object        Phone                   object
County                    object        Multiple                object
Zip                        int64        OnlineSecurity          object
Lat                      float64        OnlineBackup            object
Lng                      float64        DeviceProtection        object
Population                 int64        TechSupport             object
Area                      object        StreamingTV             object
Timezone                  object        StreamingMovies         object
Job                       object        PaperlessBilling        object
Children                 float64        PaymentMethod           object
Age                      float64        Tenure                 float64
Education                 object        MonthlyCharge          float64
Employment                object        Bandwidth_GB_Year      float64
Income                   float64        item1                    int64
Marital                   object        item2                    int64
Gender                    object        item3                    int64
Churn                     object        item4                    int64
Outage_sec_perweek       float64        item5                    int64
Email                      int64        item6                    int64
Contacts                   int64        item7                    int64
Yearly_equip_failure       int64        item8                    int64
Techie                    object
```

The data consists of both numeric and string data, and noticeably the missing data consists of 'NA' values where either through entry or collection error the information is missing. The best approach for the dataset is to first identify where missing values are located and then depending on frequency and type of data either replace or impute the missing values (Lianne & Justin @ Just into Data, 2021). I will use techniques described in the WGU provided Textbook and the Python labs to accomplish the data cleaning.

C3. Programming Language and Libraries

For this assignment, I have opted to use the Python language for data cleaning. Cleaning the data will involve the following Python Libraries:

Pandas – This package is most often used to import the dataset into a Python dataframe using the read attribute. Pandas may also be used for the groupby function. Pandas is also useful for the loc attribute which provides access to a group of rows and columns by the label in the dataframe's array. Other Panda functions include describe, shape, and dtypes (Pandas 2021).

Numpy – A useful attribute of NumPy is .std which helps compute the standard deviation in a given array. Another useful NumPy attribute is .mean which returns the average of the array elements. NumPy helps observe and detect outliers in the dataset. Other useful NumPy attributes are .dtype, .array, .shape, and .arrange (Numpy 2021).

Sklearn – The Sklearn subpackage decomposition will help by accessing the PCA attribute. This Python package will be used in the Principal component analysis. The PCA attribute will be used for linear dimensionality reduction in the analysis. The module is implemented as a transformer object for n components in the fit() method (VanderPlas).

Seaborn – This Python package will be used in conjunction with Matplotlib to help visualize the data and detect anomalies and outliers. Using Seaborn helps to easily visualize the data while Matplotlib will be used to get insights from multiple graphs. Seaborn is better capable of displaying a high-level interface for drawing informative statistical graphics (Matplotlib 2021).

Matplotlib – The sub-attribute PyPlot is used both to help execute commands easier and make use of the range of commands necessary to create and edit plots. PyPlot can be utilized to create scatter plots to convey an array of information such as overlapping data, clutter, and overall trends. This can aid greatly in visually inspecting data to view anomalies and outliers (Matplotlib 2021).

C4. Identifying Anomalies Code

```
import pandas

#Import the churn_raw_data.csv dataset
df = pandas.read_csv('/Users/Richard/OneDrive - Western Governors University/MSDA
/Databases/Churn DB/churn_raw_data.csv')

#Verifying the Variable Types

data = df.dtypes
print(data)

#Displaying Unique Values
#This code can be utilized across different columns to assess unique values, in
this example the "age" column is checked.

data = df['Age'].value_counts()
print(data)

#Detecting Duplicate Values

data = df.loc[df.duplicated()]
print(data)

#Detecting Missing Data

data = df.isnull().sum()
print(data)

#Identifying Standard Deviation

data = df.std()
print(data)

#Defining Outliers
#Used to define an outlier whose z-score value is greater than 3 or less than -
3 and uses the code to displays rows with outliers
In this example the "Children" column is analyzed

children_z = (df['Children'] - df['Children'].mean()) /df['Children'].std()
data = df.loc[(children_z > 3) | (children_z < -3)]
print(data)
```

Part III

D1. Data Cleaning Process Summary

Displaying unique values

In [14]:
```
data = df['CaseOrder'].value_counts()
print(data)
```

```
2049    1
8865    1
6806    1
4759    1
8857    1
       ..
9526    1
5432    1
7481    1
1338    1
2047    1
Name: CaseOrder, Length: 10000, dtype: int64
```

In [15]:
```
data = df['Customer_id'].value_counts()
print(data)
```

```
F259824    1
U773599    1
H676214    1
Q877999    1
X685005    1
          ..
F707493    1
E782376    1
W368790    1
X557239    1
M314123    1
Name: Customer_id, Length: 10000, dtype: int64
```

In [16]:
```
data = df['Interaction'].value_counts()
print(data)
```

```
de9a502c-75ee-492c-939f-281ff3e7ce40    1
9b67895f-e475-4f97-8134-ba076f60f1f9    1
a4b937d5-1b4f-46ef-8ba7-3295ffd32c2f    1
f28c1a54-5664-45a2-af2b-c8ee083844cf    1
daf1d0aa-2568-4137-9c45-e2be41b71a7d    1
                                       ..
c4f96071-fb53-477b-80ce-2584f6c9d975    1
5a4a3220-a7de-438d-b688-5d1f9457db51    1
61bc51d4-a9fa-41f8-879b-feb591f6405c    1
3c8e3497-b57f-43fd-96ab-0e9ec7fe80a4    1
37c2716d-3bb3-44b2-a46d-e74cb655f3db    1
Name: Interaction, Length: 10000, dtype: int64
```

Detecting Duplicate Values

```
In [17]: data = df.loc[df.duplicated()]
         print(data)
```

```
Empty DataFrame
Columns: [Unnamed: 0, CaseOrder, Customer_id, Interaction, City, State, County, Zip, Lat, Lng, Popula
tion, Area, Timezone, Job, Children, Age, Education, Employment, Income, Marital, Gender, Churn, Outa
ge_sec_perweek, Email, Contacts, Yearly_equip_failure, Techie, Contract, Port_modem, Tablet, Internet
Service, Phone, Multiple, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, S
treamingMovies, PaperlessBilling, PaymentMethod, Tenure, MonthlyCharge, Bandwidth_GB_Year, item1, ite
m2, item3, item4, item5, item6, item7, item8]
Index: []

[0 rows x 52 columns]
```

Detecting Missing Data

In [18]:
```
data = df.isnull().sum()
print(data)
```

```
Unnamed: 0                 0
CaseOrder                  0
Customer_id                0
Interaction                0
City                       0
State                      0
County                     0
Zip                        0
Lat                        0
Lng                        0
Population                 0
Area                       0
Timezone                   0
Job                        0
Children                2495
Age                     2475
Education                  0
Employment                 0
Income                  2490
Marital                    0
Gender                     0
Churn                      0
Outage_sec_perweek         0
Email                      0
Contacts                   0
Yearly_equip_failure       0
Techie                  2477
Contract                   0
Port_modem                 0
Tablet                     0
InternetService            0
Phone                   1026
Multiple                   0
OnlineSecurity             0
OnlineBackup               0
DeviceProtection           0
TechSupport              991
StreamingTV                0
StreamingMovies            0
PaperlessBilling           0
PaymentMethod              0
Tenure                   931
MonthlyCharge              0
Bandwidth_GB_Year       1021
item1                      0
item2                      0
item3                      0
item4                      0
item5                      0
item6                      0
item7                      0
item8                      0
dtype: int64
```

Identifying Standard Deviation

In [19]:
```
data = df.std()
print(data)
```

```
Unnamed: 0                2886.895680
CaseOrder                 2886.895680
Zip                      27532.196108
Lat                          5.437389
Lng                         15.156142
Population               14432.698671
Children                     2.154758
Age                         20.753928
Income                   28358.469482
Outage_sec_perweek           7.025921
Email                        3.025898
Contacts                     0.988466
Yearly_equip_failure         0.635953
Tenure                      26.438904
MonthlyCharge               43.335473
Bandwidth_GB_Year         2187.396807
item1                        1.037797
item2                        1.034641
item3                        1.027977
item4                        1.025816
item5                        1.024819
item6                        1.033586
item7                        1.028502
item8                        1.028633
dtype: float64
```

Z-Scores

In [18]:
```
outlier_z = (df['Population'] - df['Population'].mean()) /df['Population'].std()
data = df.loc[(outlier_z > 4) | (outlier_z < -4)]
print(data)
```

```
      Unnamed: 0  CaseOrder Customer_id                           Interaction
\
157          158        158     K265986  9f0486c9-e0fc-4762-8f35-e676fbefb689
203          204        204      K33780  10a165f2-6ce5-47ff-bec5-925b3d5eebf0
442          443        443     B442948  eb61eb8e-8180-4ad0-85b5-eff490b10fca
555          556        556      F05516  0bdfd1ca-312e-48fe-ab07-9515c5e55342
829          830        830     E682726  cedb18f0-6255-424c-9594-c87fae41b2cd
...          ...        ...         ...                                   ...
8947        8948       8948      P01863  9f8f0dbe-426f-40c9-a59d-dc42b5f2eff2
9056        9057       9057      S95379  9400e6fe-e466-4baa-bd7e-e771cd3653ac
9616        9617       9617     P315303  06ca7b75-c1b6-4b4d-a5f1-09fa4dcc6ceb
9987        9988       9988     C454652  c4cb88a8-dd44-46a4-84e7-891edf25cbaf
9996        9997       9997     D861732  6e96b921-0c09-4993-bbda-a1ac6411061a

              City State        County    Zip       Lat        Lng  ...  \
157    League City    TX     Galveston  77573  29.50205  -95.08652  ...
203        Chicago    IL          Cook  60639  41.92056  -87.75603  ...
442       Brooklyn    NY         Kings  11206  40.70189  -73.94237  ...
555   Philadelphia    PA  Philadelphia  19120  40.03365  -75.11998  ...
829         Folsom    CA    Sacramento  95630  38.66707 -121.14176  ...
...            ...   ...           ...    ...       ...        ...  ...
8947       El Paso    TX       El Paso  79938  31.83091 -105.97010  ...
9056     Hollywood    FL       Broward  33024  26.02697  -80.24528  ...
9616   Los Angeles    CA   Los Angeles  90026  34.07927 -118.26300  ...
9987       Chicago    IL          Cook  60647  41.92068  -87.70167  ...
9996   Clarksville    TN    Montgomery  37042  36.56907  -87.41694  ...

      MonthlyCharge Bandwidth_GB_Year item1 item2  item3  item4 item5 item6  \
157      175.495369        967.981914     4     5      4      4     3     3
203      128.445831       1430.761492     2     3      1      3     3     1
442      183.866945        544.123260     2     3      4      5     2     5
555      163.289467        844.871172     2     4      3      3     5     3
829      195.197902       1120.116258     5     3      4      4     5     4
...             ...               ...   ...   ...    ...    ...   ...   ...
8947     273.471900       6484.572000     3     3      2      3     4     3
9056     113.754000               NaN     1     3      3      4     4     3
9616     245.442700       5231.660000     2     3      2      3     4     2
9987     219.019400       5135.576000     4     4      3      5     3     3
```

```
9996    208.856400    5695.952000    4    5    5    4    4    5

      item7  item8
157       3      5
203       4      3
442       5      3
555       3      2
829       3      3
...     ...    ...
8947      5      2
9056      4      3
9616      3      2
9987      3      3
9996      2      5

[70 rows x 52 columns]
```

In [14]:
```python
outlier_z = (df['Income'] - df['Income'].mean()) /df['Income'].std()
data = df.loc[(outlier_z > 4) | (outlier_z < -4)]
print(data)
```

```
3953      3      3
3985      5      3
4249      5      2
4406      3      5
4904      4      4
5583      5      4
5801      4      3
6130      3      2
6837      4      2
7963      4      3
8457      4      5
8830      4      2
9032      4      5
9157      5      4
9180      4      3
9233      4      5
9249      3      3

[29 rows x 52 columns]
```

D2. Methods for mitigating Anomalies

Detecting Missing Data - Children, Age, Income, Techie, Phone, TechSupport, Tenure, Bandwidth_GB_Year contains rows that are missing data. The missing data values will be corrected with imputation using median values.

Standard Deviation and Identifying and Removing Outliers using Z-Scores - From calculating standard deviation, we can see Population and Income columns have rows with possible outliers. We can further examine this by checking z-scores for the two columns. From calculating Z-scores on the Population and Income columns we can see that Population has 70 outliers and Income has 29 outliers. In this situation it is best not to remove the outliers since they contain essential data but to impute the outliers with median values.

Duplicate Columns – There exists a duplicate feature named "Unnamed: 0" which serves no purpose and therefore the column can be safely removed.

Displaying Unique Values - CaseOrder, Customer_ID, and Interaction have been verified to have unique rows of data. Therefore, no further action is necessary to clean or correct these columns.

Detecting Duplicate Values - The data has been verified to have unique values in each row showing that there is no duplicate data. While each column may have multiple instances with identical data, no columns exist that have identical data in each row. Therefore, no further action is necessary to clear or correct the rows.

D3. Summary of Implementation

No action is necessary for Verifying Unique value in the Identifier columns.

There is also no action necessary for correcting duplicate rows.

For Missing Data the following actions are to be taken for columns with missing data:

'Children' change values from NA/NaN to 0,

'Age' Impute for NA/NAN values,

'Income' Impute for NA/NAN values,

'Technie' change values from NA/NaN to No,

'Phone' change values from NA/NaN to No,

'TechSupport' change values from NA/NaN to No,

'Tenure' Impute for NA/NAN values,

'Bandwidth_GB_Year' Impute for NA/NAN values.

From our examination of outliers earlier, the Population and Income features contain many outliers based on Z-score. Due to the relatively low amount, it seems best to impute rows in population and Income with significant outliers (Lianne & Justin @ Just into Data, 2021).

D4. **Mitigating Anomalies Code**

```
import pandas as pd
import numpy as np
import seaborn as sns

#Drop unnecessary ‘Unnamed: 0’ column
df = df.drop('Unnamed: 0', 1)

#‘Children’ change values from NA/NaN to 0
df["Children"].fillna("0", inplace = True)

#‘Age’ Impute for NA/NAN values
med = df['Age'].median()
df['Age'] = df['Age'].fillna(med)

#‘Income’ Impute for NA/NAN values
med = df['Income'].median()
df['Income'] = df['Income'].fillna(med)

#‘Technie’ change values from NA/NaN to No
df["Techie"].fillna("No", inplace = True)

#‘Phone’ change values from NA/NaN to No
df["Phone"].fillna("No", inplace = True)

#‘TechSupport’ change values from NA/NaN to No
df["TechSupport"].fillna("No", inplace = True)

#‘Tenure’ Impute for NA/NAN values
med = df['Tenure'].median()
df['Tenure'] = df['Tenure'].fillna(med)

#‘Bandwidth_GB_Year’ Impute for NA/NAN values
med = df['Bandwidth_GB_Year'].median()
df['Bandwidth_GB_Year'] = df['Bandwidth_GB_Year'].fillna(med)

#Impute Outliers in Income Feature
med = df['Income'].median()
df['Income'] = df['Income'].fillna(med)
```

D5. Copy of Cleaned Data Set

Cleaned Dataset Provided in attached file: churn_clean.csv

D6. Summary of Limitations

There are of course, as with any data set, limitations on cleaning the dataset. The biggest limitation is the inability to discuss the data with the staff that has collected and organized the data. A few questions can grant clarification to the data and better improve the cleaning such as, why are latitude and longitude included in the dataset? Or to ask how data is collected to improve the imputation of NA/NaN values in the dataset. The ability to understand the origin of the inconsistencies would greatly help in deciding whether to replace, impute, or drop values that can cause trouble with later analysis. In a different setting with access to coworkers, these limitations may still exist but insight into the data will greatly enhance later observations.

D7. Effect of Limitations on Analysis Question

The question from part A is "What are the characteristics of customers who choose to continue telecommunication services with the company and what are common characteristics of customers who terminate services and contribute to churn?"

The limitations in D6 can affect later observations as features such as children, age, income, tech support, and tenure can all have information that can give insight into customers who choose to stay or leave the telecommunications company. While the limitations could lead to better acquisition of data and greater accountability for missing data, the dataset is sufficient to provide meaningful analysis of customer characteristics and common trends among customers who contribute to churn.

E1. **Principal Components List**

The items available for PCA are:

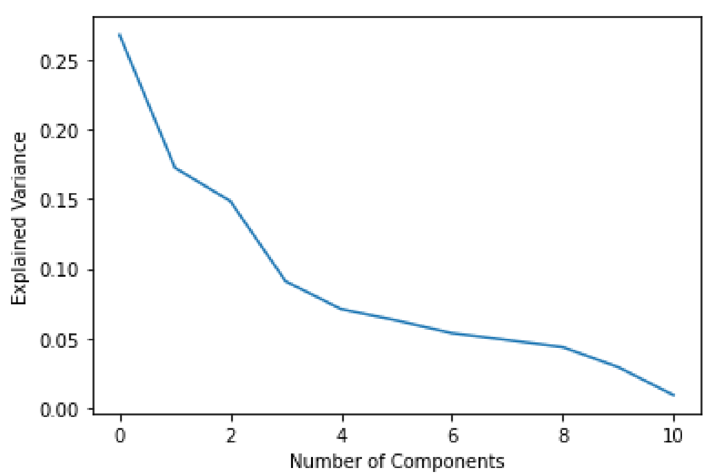
Tenure
MonthlyCharge
Bandwidth_GB_Year
Item1: Timely response
Item2: Timely fixes
Item3: Timely replacements
Item4: Reliability
Item5: Options
Item6: Respectful response
Item7: Courteous exchange
Item8: Evidence of active listening

```
data = df.loc[:, 'Tenure':'item8']
data.head()
```

	Tenure	MonthlyCharge	Bandwidth_GB_Year	item1	item2	item3	item4	item5	item6	item7	item8
0	6.795513	171.449762	904.536110	5	5	5	3	4	4	3	4
1	1.156681	242.948015	800.982766	3	4	3	3	4	3	4	4
2	15.754144	159.440398	2054.706961	4	4	2	4	4	3	3	3
3	17.087227	120.249493	2164.579412	4	4	4	2	5	4	3	3
4	1.670972	150.761216	271.493436	4	4	4	3	4	4	4	5

Table of Principal Components

```
plt.plot(pca.explained_variance_ratio_)
plt.xlabel('Number of Components')
plt.ylabel('Explained Variance')
plt.show();
```



SCREE PLOT of PCA Values

E2. Identifying Principal Components

The Principal Components were identified based on relevance to churn through a customer's point of view from interactions with customer service and the quality of service received. These four components serve as good indicators when analyzing telecommunication churn rates.

To analytically justify the PCA factors the eight items were visually represented using a scree plot and from this graph, the eigenvalues were extracted from the elbow bend (VanderPlas). The bend showed at a factor of 3 and lowered until 1 at the last component. Using Numpy and calculating the cumsum revealed these as important features of the churn data.

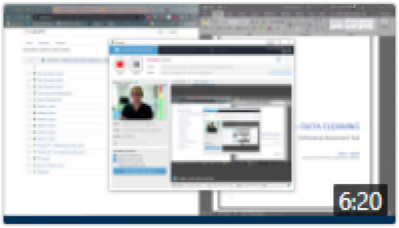
E3. Benefits of PCA results

The weight of the four variables in the analysis (Timely Responses, Fixes, Replacements, and Respectful Responses) suggests that customer satisfaction can be enhanced with emphasis on these factors. Spending more effort in these areas can directly lead to a reduction in churn and thus increasing retention rates of customers. Before a more targeted analysis using the clean churn data, we can directly observe that increasing resources in these areas will correlate to a positive response in churn rate (VanderPlas).

Part IV

F. Panopto Recording

Panopto recording can be found here:
https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=b84cf5f4-aff3-4dfd-b0ab-adb100046b78



D206 - Data Cleaning - Richard Flores

Data Cleaning NUM2 | D206 (student creators) [assignments]

G. Web Source References

Lianne & Justin @ Just into Data. (2021, May 3). *Data Cleaning in python: The ultimate guide (2020)*. Medium. Retrieved September 23, 2021, from https://towardsdatascience.com/data-cleaning-in-python-the-ultimate-guide-2020-c63b88bf0a0d.

VanderPlas, J. (n.d.). *In depth: Principal component analysis*. In Depth: Principal Component Analysis | Python Data Science Handbook. Retrieved September 23, 2021, from https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html.

H. Sources and References

Larose, C. D., & Larose, D. T. (2019). *Data Science using python and R*. Wiley.

User guide. User Guide - pandas 1.3.3 documentation. (n.d.). Retrieved September 23, 2021, from https://pandas.pydata.org/docs/user_guide/index.html#user-guide.

The absolute basics for beginners. NumPy. (n.d.). Retrieved September 23, 2021, from https://numpy.org/doc/stable/user/absolute_beginners.html.

Overview. Overview - Matplotlib 3.4.3 documentation. (n.d.). Retrieved September 23, 2021, from https://matplotlib.org/stable/contents.html.