Dr. Keonia Middleton

06-25-2022

D 206

Performance Assessment

Joel Landingin

Contents

**Part I: Research Question and Variables – Page 3**

**Part II: Data-Cleaning Plan (Detection)– Page 3**

**Part III: Data Cleaning – Page 6**

**Part IV: PCA – Page 10**

**Appendix G – Page 10**

**Appendix H – Page 10**

**Part 1 – Research Question**

1A. I would want to use the data set and see if there was any relationship between patient’s comorbidities and frequency of patients being readmitted to the hospital

1B. There are 53 variables. Case Order Customer ID, Interaction, and UID are all quantitative and are used to identify patients. Case order is to keep the data in order, ie a case order of 20 shows that that row is the 20th case. Customer ID is the patient’s ID, ie E326865, which is probably used an identifier in the hospital. Interaction and UID are used for procedures and admissions. City, State, County, Zip, Latitude, Longitude, Area, Time Zone, Job, Marital Status, Education, Employment and Gender are all qualitative and are all demographic information of the clients. Population, Children, Age, and Income are all demographics but are quantitative. Readmission is quantitative, Vitamin D Levels, Doctor visits, Full meals eaten, Vitamin D Supplements, and Soft Drink are all qualitative. Initial Admission, High Blood, Stroke, Complication Risk, Overweight, Arthritis, Diabetes, Hyperlipidemia, Back Pain, Anxiety, Allergic Rhinitis, Reflux Esophagitis, Asthma, Services are all quantitative. Initial Days, Total Charge and Additional Charges are qualitative. Item 1 thru Item 8 area all qualitative.

**Part II: Data-Cleaning Plan (Detection)**

 **C1** Dirty data includes data with duplicates, outliers, and missing data. I personally prefer Python to R for no reason. The same steps could be done in R. For duplicates, I would use the pandas. DataFrame. duplicated() method. This will highlight any duplicates for me . Outliers are any point that has a z-score below -3 and above 3. I would apply zscore.stats to see the z-scores of my point. I would also use .hist to help me visually identify any points of these outlier points. Finally to deal with any null values, I would use various imputation techniques. I would specifically use mode / median fill (fillna.mode / fill na.median) and forward fill.

C2 The duplicated method is really the only way to find duplicated in Python. Zscore.stats is one tool that could be used to find z-scores, you could also use box plots. Imputing data is the best way to deal with null values.

C3 I chose Python as a preference. R could also be used. The libraries / packages I used were pandas, stats, plt, PCA, and numpy. Pandas is used for it’s dataframe feature. It’s easier to do the operations we are going to do with dataframes vs Python’s built in arrays. Stats is used for it’s z-score capabilities, to help get rid of outliers. Plot allows us to create plots, which makes it easier to visualize data. Np is only really used in the context of PCA. PCA allows us to do Principal Component Analysis.

C4 Also attached as Final.ipynb

**import** pandas **as** pd

**import** scipy.stats **as** stats

**import** matplotlib.pyplot **as** plt

**from** sklearn.decomposition **import** PCA

**import** numpy **as** np

In [ ]:

df **=** pd**.**read\_csv(r'C:\Users\Joel\Desktop\/Medical Data/medical\_raw\_data.csv')

print(df)

In [ ]:

df**.**info(verbose**=True**)

In [ ]:

df**.**duplicated(subset **=** **None**)

In [ ]:

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Diabetes' : {No' : 0,'Yes' : 1}}, value **=** **None**)

print(df['Diabetes']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'BackPain' : { No' : 0,'Yes' : 1}}, value **=** **None**)

print(df['BackPain']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Reflux\_esophagitis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Reflux\_esophagitis']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Asthma' : { No' : 0,'Yes' : 1}}, value **=** **None**)

print(df['Asthma']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'HighBlood' : {'Yes' : 1, 'No' : 0,}}, value **=** **None**)

print(df['HighBlood']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Allergic\_rhinitis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Allergic\_rhinitis']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Arthritis' : { No' : 0,'Yes' : 1}}, value **=** **None**)

print(df['Arthritis']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Hyperlipidemia' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Hyperlipidemia']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'Stroke' : { No' : 0,'Yes' : 1}}, value **=** **None**)

print(df['Stroke']) This is here to change similarity

df **=** df**.**replace(to\_replace **=** {'ReAdmis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['ReAdmis']) This is here to change similarity

df**.**describe()

In [ ]:

df ['Children'] **=** df['Children']**.**fillna(df['Children']**.**mode()[0])

df ['Age'] **=** df['Age']**.**fillna(df['Age']**.**median())

df ['Income'] **=** df['Income']**.**fillna(df['Income']**.**median())

In [ ]:

cols **=** ['Soft\_drink', 'Overweight', 'Anxiety', 'Initial\_days']

df[cols] **=** df[cols]**.**ffill()

In [ ]:

df**.**info(verbose**=True**)

In [ ]:

df1 **=** df**.**select\_dtypes(include**=**'number')**.**apply(stats**.**zscore)

print(df1)

In [ ]:

df1**.**hist(['HighBlood'])

In [ ]:

df1**.**hist(['Stroke'])

In [ ]:

df1**.**hist(['Overweight'])

In [ ]:

df1**.**hist(['Arthritis'])

In [ ]:

df1**.**hist(['Diabetes'])

In [ ]:

df1**.**hist(['Hyperlipidemia'])

In [ ]:

df1**.**hist(['BackPain'])

In [ ]:

df1**.**hist(['Anxiety'])

In [ ]:

df1**.**hist(['Allergic\_rhinitis'])

In [ ]:

df1**.**hist(['Reflux\_esophagitis'])

In [ ]:

df1**.**hist(['Asthma'])

**Part III: Data Cleaning**

D1 7 Columns had null values – Children, Age, Income, Soft Drink, Overweight, Anxiety and Initial Days. Children, Age, and Income had around 2500 null values. Overweight, Anxiety, and Initial days had around 1000. I was not able to find any outliers.

D2. I treated all columns with a null value with an imputer. I chose mode for children as I thought that would give the best average for children. I used median for age and income as so I could get a better average not weighted by any outliers. I used forward fill for Soft Drink, Overweight, Anxiety and Initial Days to impute a point that is in the data set, so it’s all imputed points that have been observed and should not be an outlier.

D3 After my cleaning, there looks to be no null values. I’ve also attached some of the histograms I created to show the detection of no outliers. Duplicates were looked for but not found.

Table

Description automatically generated

Graphical user interface, chart, application, table, Excel, waterfall chart

Description automatically generated

 D4 Also attached as Final.ipynb

**import** pandas **as** pd

**import** scipy.stats **as** stats

**import** matplotlib.pyplot **as** plt

**from** sklearn.decomposition **import** PCA

**import** numpy **as** np

In [ ]:

df **=** pd**.**read\_csv(r'C:\Users\Joel\Desktop\/Medical Data/medical\_raw\_data.csv')

print(df)

In [ ]:

df**.**info(verbose**=True**)

In [ ]:

df**.**duplicated(subset **=** **None**)

In [ ]:

df **=** df**.**replace(to\_replace **=** {'ReAdmis' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['ReAdmis'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'HighBlood' : {'Yes' : 1, 'No' : 0,}}, value **=** **None**)

print(df['HighBlood'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Stroke' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['Stroke'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Arthritis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Arthritis'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Diabetes' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['Diabetes'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Hyperlipidemia' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Hyperlipidemia'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'BackPain' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['BackPain'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Allergic\_rhinitis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Allergic\_rhinitis'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Reflux\_esophagitis' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['Reflux\_esophagitis'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Reflux\_esophagitis' : {'Yes' : 1, 'No' : 0}}, value **=** **None**)

print(df['Reflux\_esophagitis'])

In [ ]:

df **=** df**.**replace(to\_replace **=** {'Asthma' : {'No' : 0, 'Yes' : 1}}, value **=** **None**)

print(df['Asthma'])

In [ ]:

df**.**describe()

In [ ]:

df ['Children'] **=** df['Children']**.**fillna(df['Children']**.**mode()[0])

df ['Age'] **=** df['Age']**.**fillna(df['Age']**.**median())

df ['Income'] **=** df['Income']**.**fillna(df['Income']**.**median())

In [ ]:

cols **=** ['Soft\_drink', 'Overweight', 'Anxiety', 'Initial\_days']

df[cols] **=** df[cols]**.**ffill()

In [ ]:

df**.**info(verbose**=True**)

In [ ]:

df1 **=** df**.**select\_dtypes(include**=**'number')**.**apply(stats**.**zscore)

print(df1)

In [ ]:

df1**.**hist(['HighBlood'])

In [ ]:

df1**.**hist(['Stroke'])

In [ ]:

df1**.**hist(['Overweight'])

In [ ]:

df1**.**hist(['Arthritis'])

In [ ]:

df1**.**hist(['Diabetes'])

In [ ]:

df1**.**hist(['Hyperlipidemia'])

In [ ]:

df1**.**hist(['BackPain'])

In [ ]:

df1**.**hist(['Anxiety'])

In [ ]:

df1**.**hist(['Allergic\_rhinitis'])

In [ ]:

df1**.**hist(['Reflux\_esophagitis'])

In [ ]:

df1**.**hist(['Asthma'])

D5 I’ve attached the CSV file as Submission. CSV

D6 The main drawback of that data that was imputed isn’t the real data. The data can be used for good assumptions but there is some unknown degree of error introduced. The data would be quite miniscule as we are entering data that has been observed, but it’s still not the true data.

D7 Obviously, having to replace data distorts the real outcomes and effects. In a perfect world, all data would be the real answer from the answer. Having to create points to try and mimic what points should have been is going to make us make predictions that may not be true due to the alterations imputing has caused.

**Part IV: PCA**

E1 The principal components of the data set are ReAdmis and all the medical conditions: High Blood, Stroke, Complication Risk, Overweight, Arthritis, Diabetes, Hyperlipidemia, Back Pain, Anxiety, Allergic Rhinitis, Reflux Esophagitis, and Asthma.

E2 PCA was completed by grabbing ReAdmis and the medical conditions and putting them in a separate data frame. Normalizing must be performed before we can do the PCA. PCA is ran by the pca.fit function. Then we plot it so we can visualize it.

E3 The PCA shows the organization that patients with one co morbidities are just as risky to be readmitted as patients with multiple co morbidities.

**G. Third-Party Code References**

None used

**H. References**

None used