Investigate_a_Dataset

July 23, 2022

1 Project: Investigate a Dataset - [No Show Appointment]

1.1 Table of Contents

1.1.1 Dataset Description

I will be working with a dataset known as 'No show Appointment'. This dataset is focused on patients and how they keep up with their medical appointments. We will also be analysing a number of characteristics about patients and how they relate to their medical appointments. This dataset contains 14 columns that show the following; PatientID - This is a unique identifier for all patients. AppointmentID - This is also a unique identifier for every appointment. Gender - This indicates the gender of patient. ScheduledDay - It shows the date the an appointment was created. AppointmentDay - This shows the actual day of the appointment. Age - This indicates the age of the patient. Neighbourhood - This tells us the location of the hospital. Scholarship - This indicates if a patient is enrolled in the brasilian welfare program or not. Hipertension - This indicates if a patient is hypertensive or not Diabetes - This indicates if a patient is diabetic or not Alcoholism - This indicates if a patient is an alcoholic or not Handcap - This indicates if a patient is handicap or not SMSReceived - This indicates if a patient received SMS or not NoShow - indicates if a patient showed up or not for an appointment. A yes in this column indicates that the patient did not show up.

1.1.2 Question(s) for Analysis

I will be analysing the following questions from the given dataset. 1. What percentage of each variable booked appointment? Are patients in from each variable caterory likely to show up for their appointment? 2. Which hospital has the highest appointment booking and which hospital was visited the most in relation to their appointment 3. What age has the highest number of appointment and what age kept to their appointment the most

```
In [1]: # Use this cell to set up import statements for all of the packages that you
        #   plan to use.
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

        %matplotlib inline
        # Remember to include a 'magic word' so that your visualizations are plotted
        #   inline with the notebook. See this page for more:
        #   http://ipython.readthedocs.io/en/stable/interactive/magics.html

In [2]: # Upgrade pandas to use dataframe.explode() function.
        #!pip install --upgrade pandas==0.25.0
```

Data Wrangling

```
In [3]: # Load your data and print out a few lines. Perform operations to inspect data
        #   types and look for instances of missing or possibly errant data.
        df = pd.read_csv('./Database_No_show_appointments/noshowappointments-kagglev2-may-2016.c
        df.head(2)
```

Out[3]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0	0	0	0	0	No

I checked for null values using isnull() function and there was no null or missing values. I also used dtype() and info() to check for datatype and they were all ok. I also checked for duplicate rows and there was none.

```
In [4]: df.shape

Out[4]: (110527, 14)

In [5]: df.isnull().sum()

Out[5]: PatientId         0
        AppointmentID     0
        Gender            0
        ScheduledDay      0
        AppointmentDay    0
```

```
Age               0
Neighbourhood     0
Scholarship       0
Hipertension      0
Diabetes          0
Alcoholism        0
Handcap           0
SMS_received      0
No-show           0
dtype: int64
```

From the dataset, I can see a total of 110,527 rows with 14 columns. The dataset has no null values. I have also noticed that the column names are inconsistent so I will rename all the column names to use small letters and underscore to join column names with more than one word.

1.1.3 Data Cleaning

```
In [6]: # After discussing the structure of the data and any problems that need to be
        #   cleaned, perform those cleaning steps in the second part of this section.
        df.columns = ['patient_id', 'appointment_id', 'gender', 'scheduled_day', 'appointment_da
```

```
In [7]: df.head(1)
```

```
Out[7]:      patient_id  appointment_id gender         scheduled_day  \
        0  2.987250e+13         5642903      F  2016-04-29T18:38:08Z

                appointment_day  age    neighbourhood  scholarship  hypertension  \
        0  2016-04-29T00:00:00Z   62  JARDIM DA PENHA            0             1

           diabetes  alcoholism  handicap  sms_received no_show
        0         0           0         0             0      No
```

```
In [8]: df.shape
```

```
Out[8]: (110527, 14)
```

For consistency and ease of reference, I removed spaces, underscore and hyphen from the column names. I also renamed all the columns and changed them to lower case.

```
In [41]: df["age"].min(), df["age"].max()
```

```
Out[41]: (0, 115)
```

I checked to see if there is any number that is not a possible age and to get the range of age of patients that showed up for appointment. I noticed that there is a number that is a negative number.

```
In [10]: df.query('age == "-1"')
```

```
Out[10]:          patient_id  appointment_id gender         scheduled_day  \
         99832  4.659432e+14         5775010      F  2016-06-06T08:58:13Z

                     appointment_day  age neighbourhood  scholarship  hypertension  \
         99832  2016-06-06T00:00:00Z   -1         ROMÃO            0             0

                diabetes  alcoholism  handicap  sms_received no_show
         99832         0           0         0             0      No
```

since age cannot be -1 and it is just one row, I decided to drop it

```
In [11]: df.drop(df[df['age'] == -1].index, inplace = True)

In [12]: df.shape

Out[12]: (110526, 14)

In [13]: df["patient_id"].nunique() == df["appointment_id"].nunique()

Out[13]: False

In [14]: df["patient_id"].nunique()

Out[14]: 62298

In [15]: df["appointment_id"].nunique()

Out[15]: 110526
```

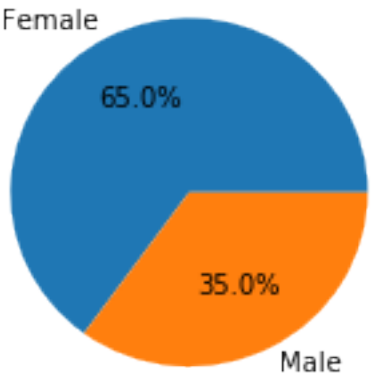
The preceeding cells rechecked the shape of the dataset to confirmed the row was dropped. It also checked the number of unique patient_id if it equals the number of unique appointment_id and it returned false which means they are not equal. So I checked for the numbers afterwards

Exploratory Data Analysis

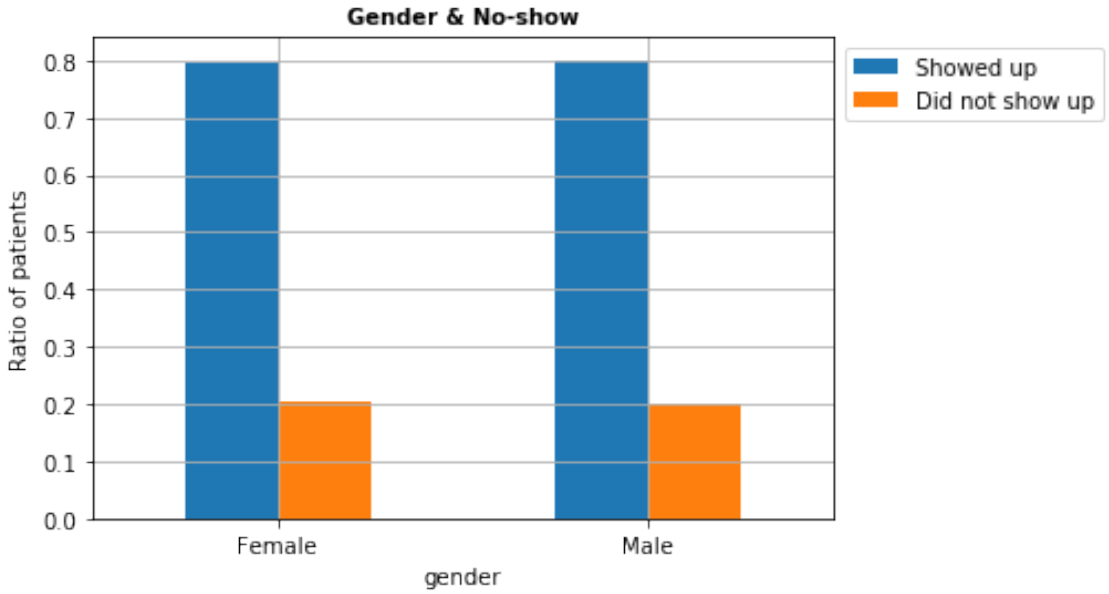
1.1.4 The following EDA and visualisation look at the percentage of each column and ratio of each column to no_show column to be able to predict an outcome.

1.1.5 Research Question 1 (What percentage of each variable booked appointment in the hospital? Does the variable have effect on patients showing up at the hospital?)

```
In [16]: df['gender'].value_counts()

Out[16]: F    71839
         M    38687
         Name: gender, dtype: int64

In [128]: fig = plt.figure(figsize =(3, 3))
          plt.pie(df['gender'].value_counts().values, autopct='%.1f%%', labels = ['Female', 'Mal
```



In [129]:
```
genderValue = df.groupby(['gender'])['no_show'].count()
showNoShowValue = df.groupby(['gender','no_show'])['no_show'].count()
ratio = showNoShowValue/genderValue
ratio.unstack().plot(kind='bar');
plt.xticks([0,1],['Female', 'Male'], rotation=0);
plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
plt.ylabel('Ratio of patients')
plt.title('Gender & No-show', fontsize=10, fontweight='bold');
plt.grid();
```

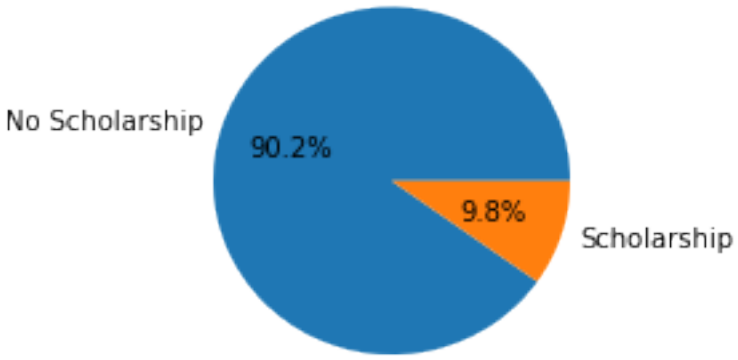


From the pie chart, we have more females patients than male patients. Female patients accounted for about 65% of the patients. From the above bar chart, we can see that we can not determine if a patient will show up using gender because we had equal ratio of patients who showed up or did not show up by gender.

```
In [19]: df['scholarship'].value_counts()
```

```
Out[19]: 0    99665
         1    10861
         Name: scholarship, dtype: int64
```

```
In [17]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['scholarship'].value_counts().values, autopct='%.1f%%', labels = ['No Schola
```



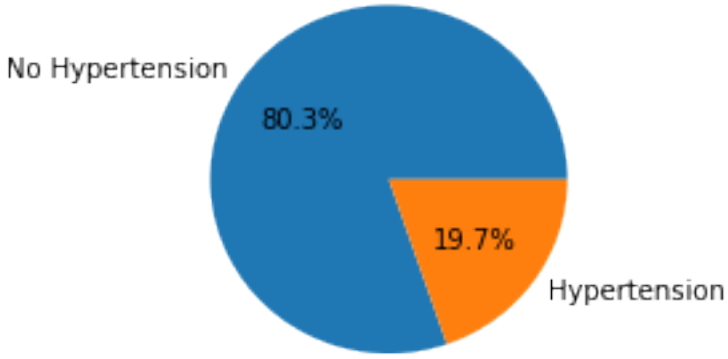

```
In [18]: scholarshipValue = df.groupby(['scholarship'])['no_show'].count()
         showNoShowValue = df.groupby(['scholarship','no_show'])['no_show'].count()
         ratio = showNoShowValue/scholarshipValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['No Scholarship', 'Got Scholarship'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Scholarship & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```

From the above pie chart, over 90% did not enroll in the Brazillian welfare programme. From the above chart, we can see that those who did not register for the Brazillian welfare programme are even more likely to show up for an appointment

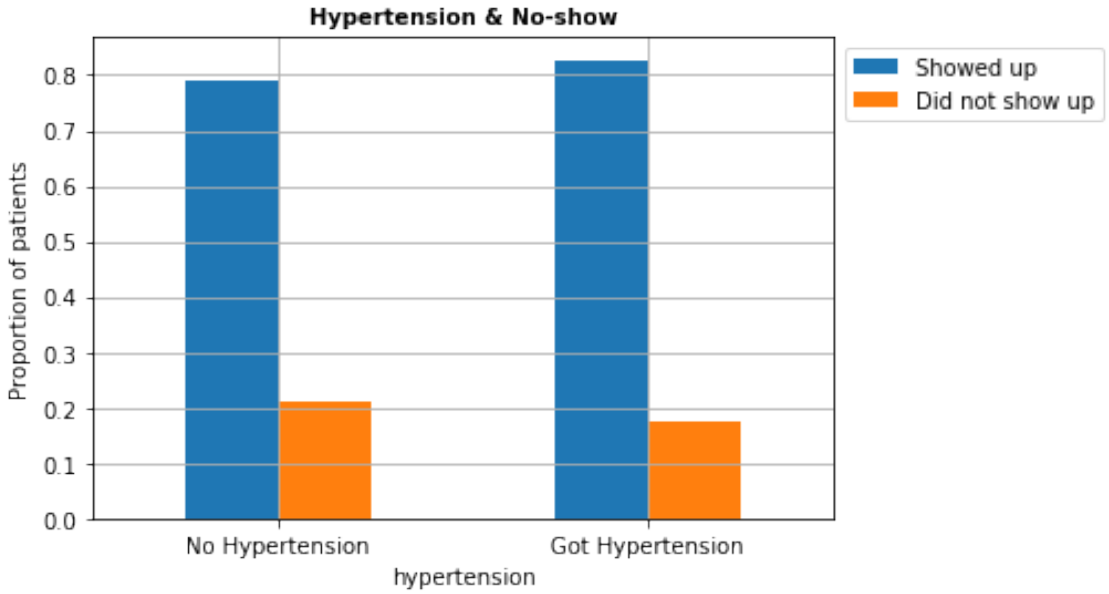
```
In [19]: df['hypertension'].value_counts()
```

```
Out[19]: 0    88725
         1    21801
         Name: hypertension, dtype: int64
```

```
In [20]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['hypertension'].value_counts().values, autopct='%.1f%%', labels = ['No Hyper
```

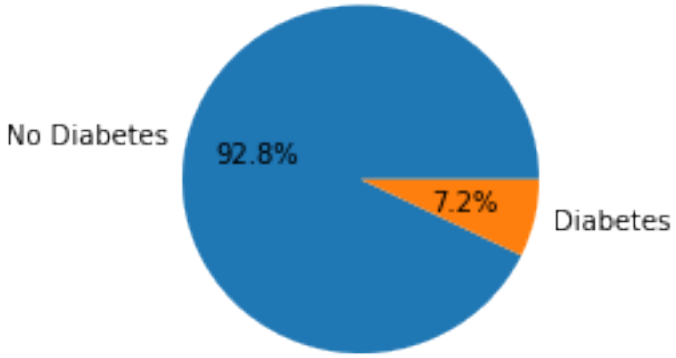


```
In [21]: hypertensionValue = df.groupby(['hypertension'])['no_show'].count()
         showNoShowValue = df.groupby(['hypertension','no_show'])['no_show'].count()
         ratio = showNoShowValue/hypertensionValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['No Hypertension', 'Got Hypertension'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Hypertension & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```

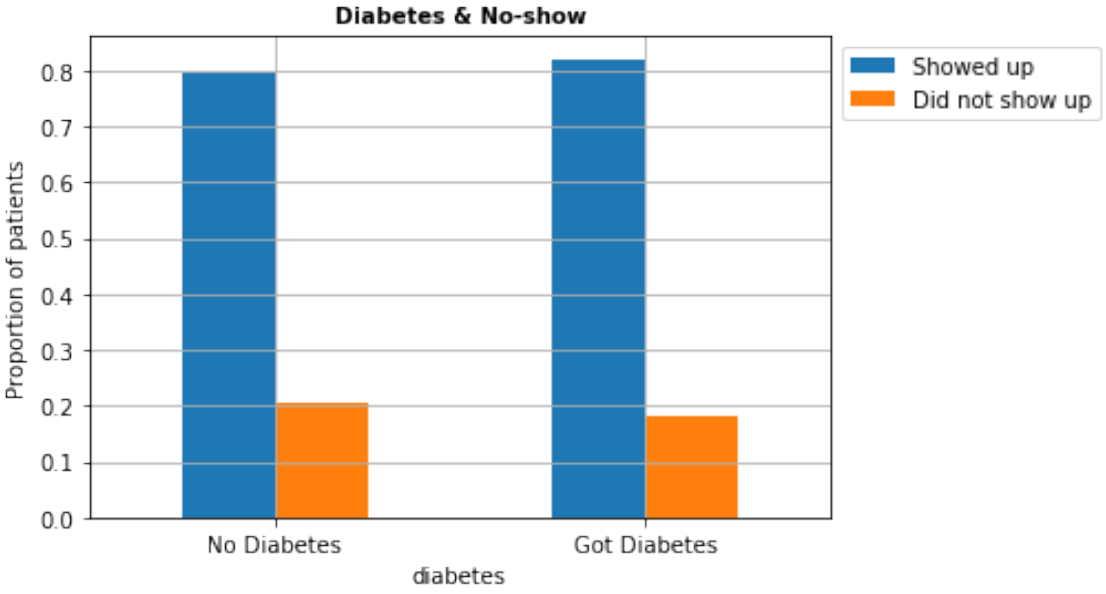


From the above chart, about 80% of the patients were free from hypertension. From the chart, we can see that patients with Hypertension are more likely to show up for an appointment

```
In [22]: df['diabetes'].value_counts()

Out[22]: 0    102583
         1      7943
         Name: diabetes, dtype: int64

In [23]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['diabetes'].value_counts().values, autopct='%.1f%%', labels = ['No Diabetes'
```

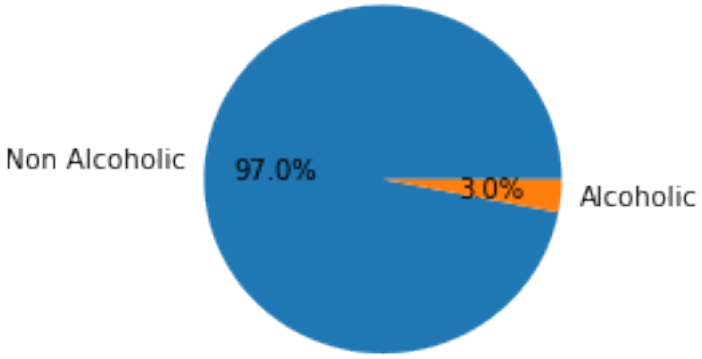



```
In [24]: diabetesValue = df.groupby(['diabetes'])['no_show'].count()
         showNoShowValue = df.groupby(['diabetes','no_show'])['no_show'].count()
         ratio = showNoShowValue/diabetesValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['No Diabetes', 'Got Diabetes'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Diabetes & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```

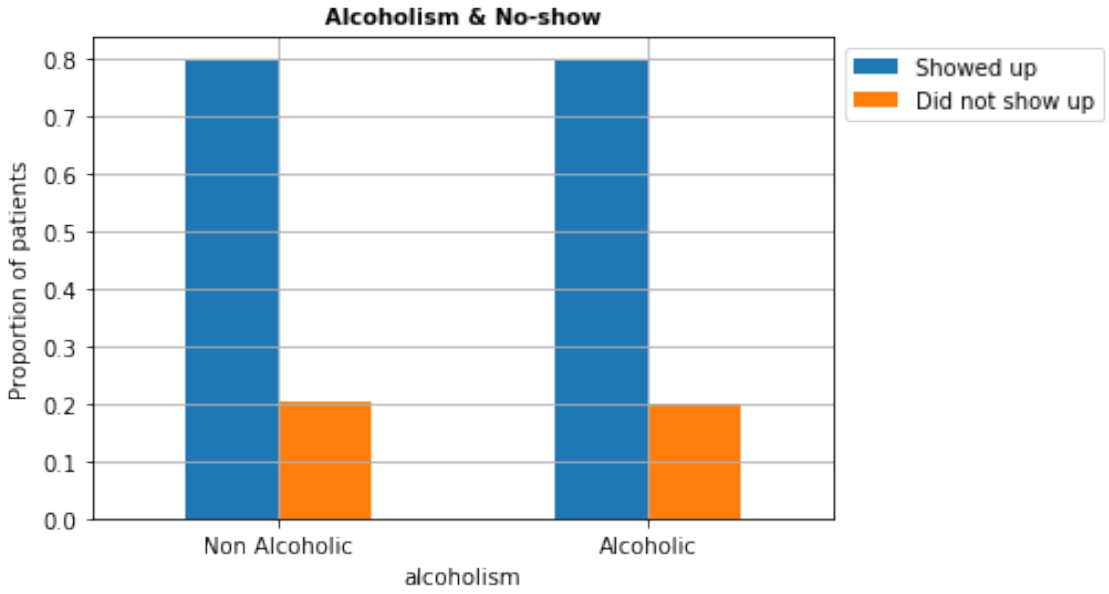


from the chart, less than 8% of the patients have diabetes. From the chart, we can see that patients with Diabetes are more likely to show up for an appointment

```
In [25]: df['alcoholism'].value_counts()

Out[25]: 0    107166
         1      3360
         Name: alcoholism, dtype: int64

In [26]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['alcoholism'].value_counts().values, autopct='%.1f%%', labels = ['Non Alcoho
```

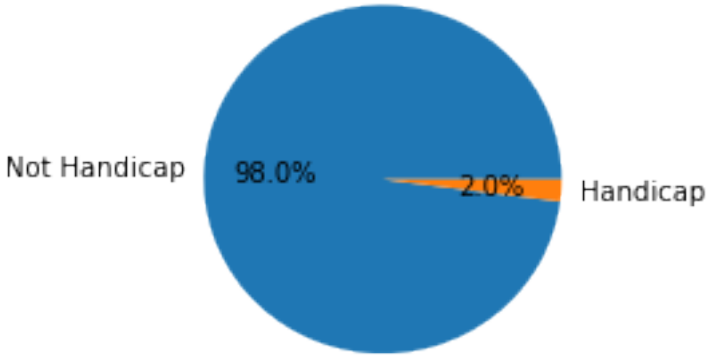



```
In [27]: alcoholismValue = df.groupby(['alcoholism'])['no_show'].count()
         showNoShowValue = df.groupby(['alcoholism','no_show'])['no_show'].count()
         ratio = showNoShowValue/alcoholismValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['Non Alcoholic', 'Alcoholic'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Alcoholism & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```

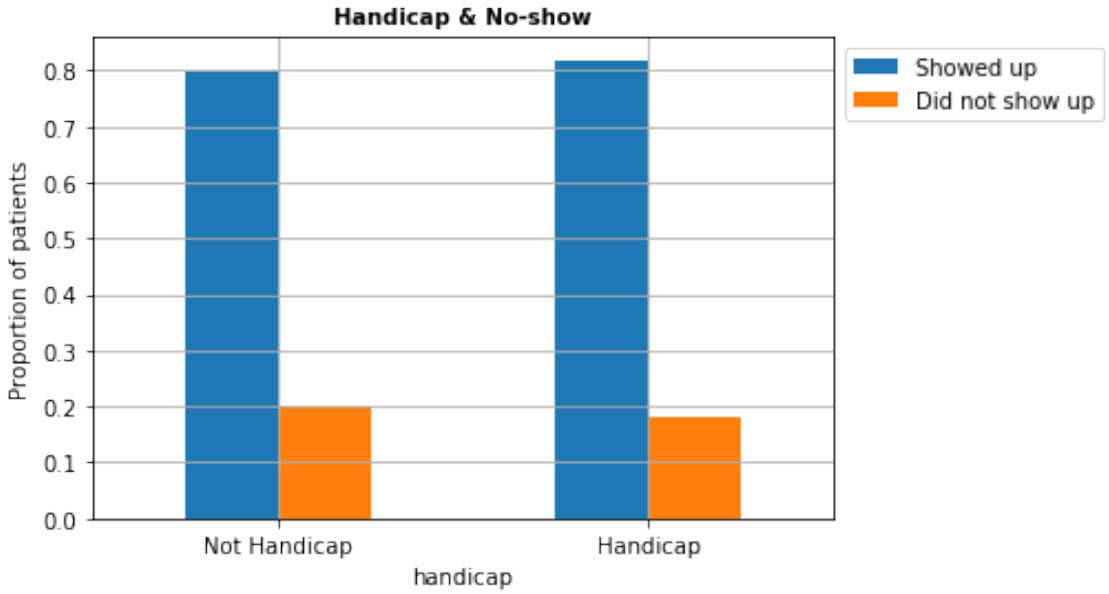


From the chart, just about 3% were alcoholics. From the chart, we can see that alcohol did not have any effect on if a patient will show up for an appointment

```
In [30]: df['handicap'].value_counts()

Out[30]: 0    108285
         1      2042
         2       183
         3        13
         4         3
         Name: handicap, dtype: int64

In [34]: df['handicap'] = df['handicap'].apply(lambda val: 1 if val >= 1 else 0)
         df['handicap'].value_counts()

Out[34]: 0    108285
         1      2241
         Name: handicap, dtype: int64

In [35]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['handicap'].value_counts().values, autopct='%.1f%%', labels = ['Not Handicap
```



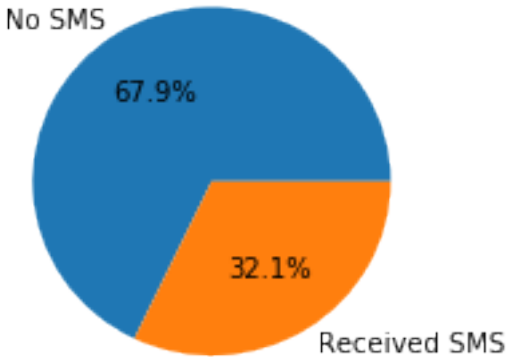

```
In [36]: handicapValue = df.groupby(['handicap'])['no_show'].count()
         showNoShowValue = df.groupby(['handicap','no_show'])['no_show'].count()
         ratio = showNoShowValue/handicapValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['Not Handicap', 'Handicap'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Handicap & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```



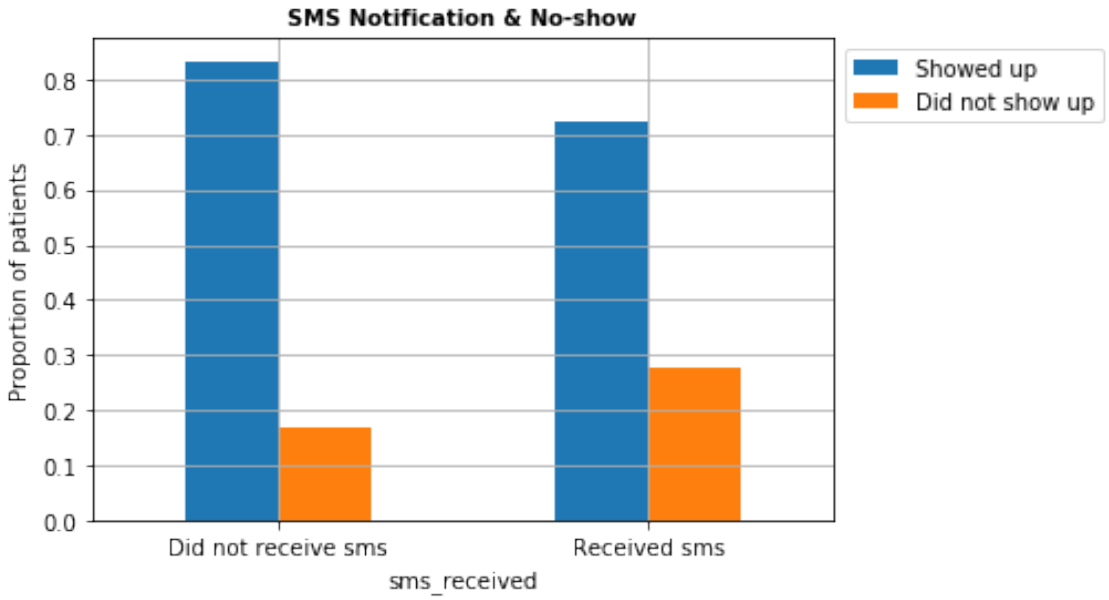
From the above, I noticed that there were some errors. It is either suppose to be 0 or 1 but I noticed values like 2, 3 and 4. Since the patient is either suppose to be handicapped or not, any positive real number should be assigned 1.

I rechecked to see if the errors have been corrected and went ahead to plot the chart. From the chart, only 2% of the patients were handcapped.From the chart, we can see that handicapped patients are more likely to show up for an appointment

```
In [37]: df['sms_received'].value_counts()

Out[37]: 0    75044
         1    35482
         Name: sms_received, dtype: int64

In [38]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['sms_received'].value_counts().values, autopct='%.1f%%', labels = ['No SMS',
```

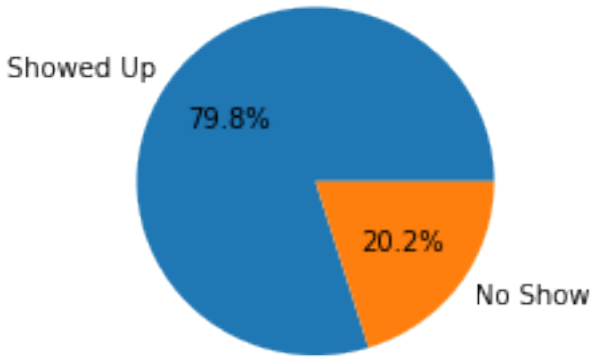



```
In [39]: smsValue = df.groupby(['sms_received'])['no_show'].count()
         showNoShowValue = df.groupby(['sms_received','no_show'])['no_show'].count()
         ratio = showNoShowValue/smsValue
         ratio.unstack().plot(kind='bar');
         plt.xticks([0,1],['Did not receive sms', 'Received sms'], rotation=0);
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('SMS Notification & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```



A high number of the population which is about 68% did not receive sms. From the chart, we can see that patients who did not receive sms are more likely to show up for an appointment

```
In [32]: df['no_show'].value_counts()

Out[32]: No     88207
         Yes    22319
         Name: no_show, dtype: int64

In [40]: fig = plt.figure(figsize =(3, 3))
         plt.pie(df['no_show'].value_counts().values, autopct='%.1f%%', labels = ['Showed Up', '
```



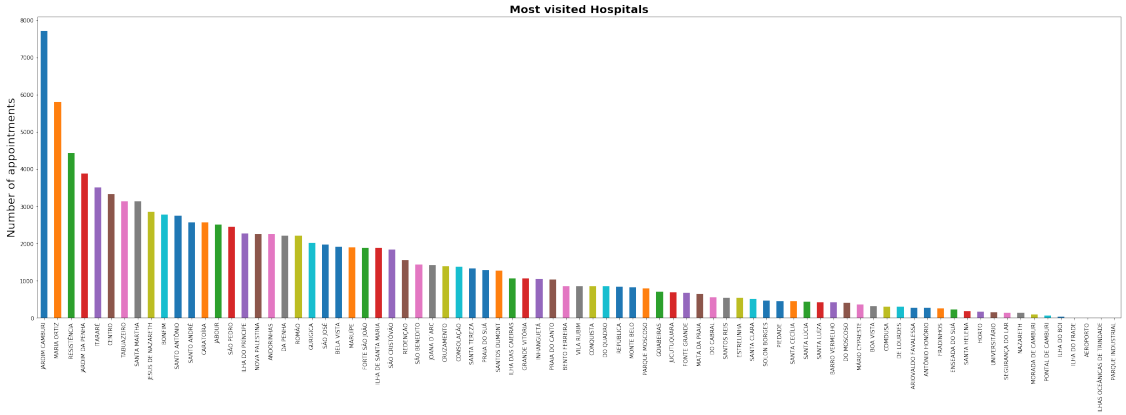
From the chart, about 80% of the patients showed up for their appointment.

1.1.6 Research Question 2 (Which hospital has the highest appointment booking and which hospital was visited the most in relation to their appointment)

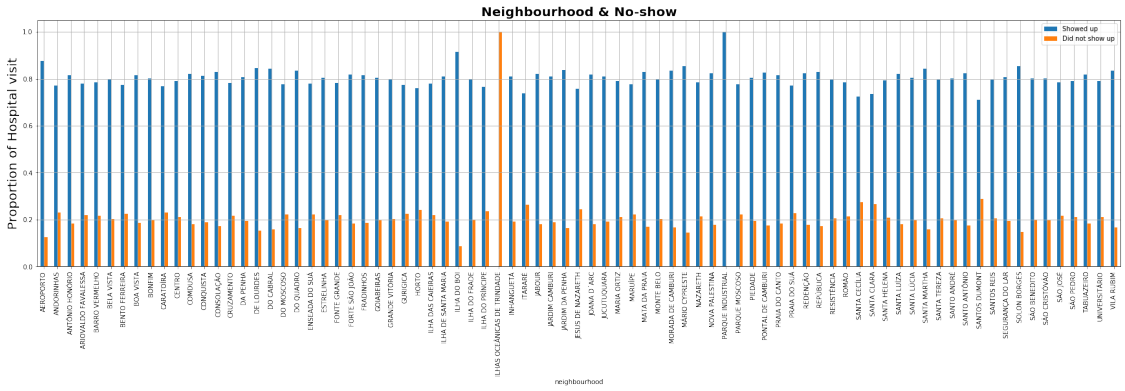
```
In [127]: df['neighbourhood'].value_counts().tail(5)
```

```
Out[127]: ILHA DO BOI                    35
          ILHA DO FRADE                  10
          AEROPORTO                       8
          ILHAS OCEÂNICAS DE TRINDADE     2
          PARQUE INDUSTRIAL               1
          Name: neighbourhood, dtype: int64
```

```
In [122]: df['neighbourhood'].value_counts().plot(figsize = (35,10),kind='bar');
          plt.xticks(rotation=90);
          plt.title('Most visited Hospitals', fontsize=20, fontweight='bold')
          plt.ylabel('Number of appointments', fontsize=20);
```




```
In [124]: neighbourhoodValue = df.groupby(['neighbourhood'])['no_show'].count()
          showNoShowValue = df.groupby(['neighbourhood','no_show'])['no_show'].count()
          ratio = showNoShowValue/neighbourhoodValue
          ratio.unstack().plot(kind='bar', figsize=(30,7));
          plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
          plt.ylabel('Proportion of Hospital visit', fontsize=20)
          plt.title('Neighbourhood & No-show', fontsize=20, fontweight='bold');
          plt.grid()
```

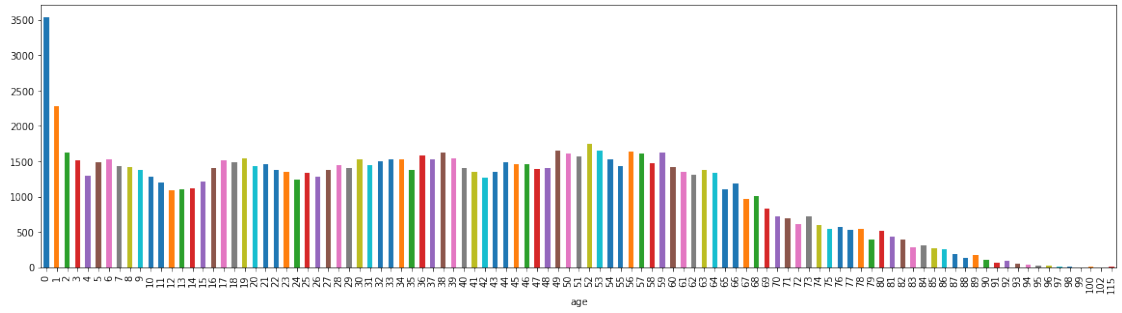


JADIM CAMBURI had the highest number of appointment followed by MARIA CRITZ while PARQUE INDUSTRIAL had the least number of appointment.

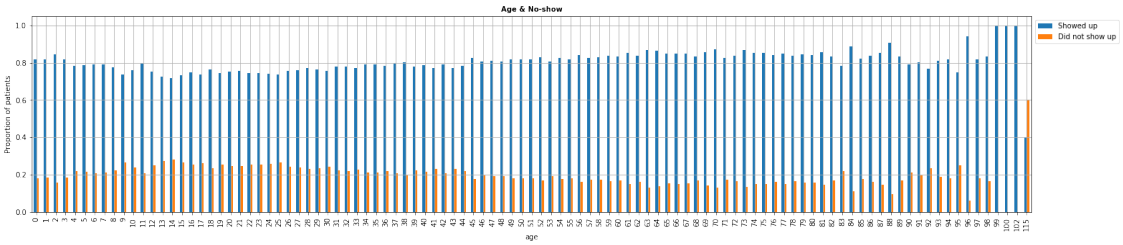
From the graph, we can not say the fewer the appointment the more likely the patients will visit because PARQUE INDUSTRIAL had 1 appointment and the patient came but ILHAS OCEANICAS DE TRINIDADE had 2 appointments and the 2 patients did not come. But predicting using the graph, we can say that PARQUE INDUSTRIAL has the most turn out while ILHAS OCEANICAS DE TRINIDADE had the least turnout.

1.1.7 Research Question 3 (What age has the highest number of appointment and what age kept to their appointment the most)

```
In [105]: df.groupby('age')['appointment_id'].count().plot(figsize = (20,5),kind='bar');
```




```
In [99]: ageValue = df.groupby(['age'])['no_show'].count()
         showNoShowValue = df.groupby(['age','no_show'])['no_show'].count()
         ratio = showNoShowValue/ageValue
         ratio.unstack().plot(kind='bar', figsize=(25,5));
         plt.legend(labels=['Showed up', 'Did not show up'], bbox_to_anchor=(1.0, 1.0));
         plt.ylabel('Proportion of patients')
         plt.title('Age & No-show', fontsize=10, fontweight='bold');
         plt.grid()
```



We can say that children of 0 age had the most appointment followed by age. but ages 99, 100 and 102 took their appointment more seious and had showed up all the time.

Conclusions >We have more females patients than male patients that booked for appointment at the various hospitals. Female patients accounted for about 65% of the patients.We can not also determine if a patient will show up using gender because we had equal ratio of patients who showed up or did not show up by gender.

Over 90% did not enroll in the Brazillian welfare programme. Those who did not register for the Brazillian welfare programme are even more likely to show up for an appointment.

About 80% of the patients were free from hypertension. Patients with Hypertension are more likely to show up for an appointment.

Less than 8% of the patients have diabetes. Patients with Diabetes are more likely to show up for an appointment.

3% of patients were alcoholics. We can not determine using alcohol if a patient will show up for an appointment.

Only 2% of the patients were handicapped. Handicapped patients are more likely to show up for an appointment.

A high number of the population which is about 68% did not receive sms. Despite not receiving sms notification patients who did not receive sms are more likely to show up for an appointment.

In general, about 80% of the patients showed up for their appointment.

We can conclude that patients with ailments are more likely to show up for an appointment.

We can not say the fewer the appointment the more likely the patients will visit because PARQUE INDUSTRIAL had 1 appointment and the patient came but ILHAS OCEANICAS DE TRINIDADE had 2 appointments and the 2 patients did not come. But predicting using the graph, we can say that PARQUE INDUSTRIAL has the most turn out while ILHAS OCEANICAS DE TRINIDADE had the least turnout.

We can say that children of 0 age had the most appointment followed by age 1 but ages 99, 100 and 102 took their appointment more seious and showed up all the time. We can say the elderly patients took their appointment more serious.

There were some limitations to the dataset 1. a number which is not real was discovered among the dataset and was dropped. 2. The handicap dataset which is suppose to have just 0 and 1(yes or no) had up to 5 categories which was also corrected.

1.2 Submitting your Project

```
In [1]: from subprocess import call
        call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])
```

```
Out[1]: 0
```