Investigate_a_Dataset

May 8, 2020

> **Tip**: Welcome to the Investigate a Dataset project! You will find tips in quoted sections like this to help organize your approach to your investigation. Before submitting your project, it will be a good idea to go back through your report and remove these sections to make the presentation of your work as tidy as possible. First things first, you might want to double-click this Markdown cell and change the title so that it reflects your dataset and investigation.

1 Project: Investigate a Dataset (No Show Apointments)

1.1 Table of Contents

Introduction
Data Wrangling
Exploratory Data Analysis
Conclusions

## Introduction
In this project, we going to study the data about medical appointments in Brazil

> This dataset collects information from 100k medical appointments in Brazil and is focused on the question of whether or not patients show up for their appointment. A number of characteristics about the patient are included in each row.
>
> 'ScheduledDay' tells us on what day the patient set up their appointment. 'Neighborhood' indicates the location of the hospital. 'Scholarship' indicates whether or not the patient is enrolled in Brasilian welfare program Bolsa Família. Be careful about the encoding of the last column: it says 'No' if the patient showed up to their appointment, and 'Yes' if they did not show up.

```
In [2]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        % matplotlib inline
```

## Data Wrangling >Read your dataset

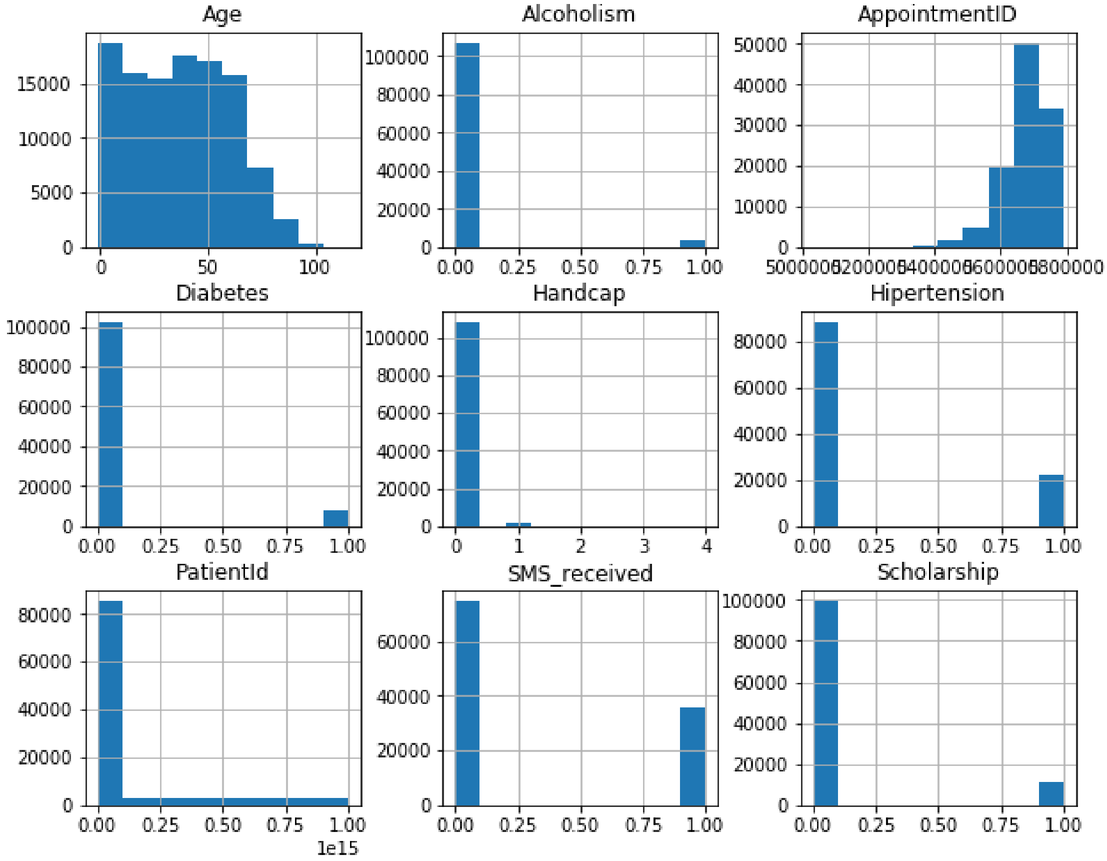
```
In [3]: df = pd.read_csv('noshowappointments-kagglev2-may-2016.csv')
        df.head()
```

```
Out[3]:       PatientId  AppointmentID Gender          ScheduledDay  \
        0  2.987250e+13        5642903      F  2016-04-29T18:38:08Z
        1  5.589978e+14        5642503      M  2016-04-29T16:08:27Z
        2  4.262962e+12        5642549      F  2016-04-29T16:19:04Z
        3  8.679512e+11        5642828      F  2016-04-29T17:29:31Z
        4  8.841186e+12        5642494      F  2016-04-29T16:07:23Z

                 AppointmentDay  Age      Neighbourhood  Scholarship  Hipertension  \
        0  2016-04-29T00:00:00Z   62    JARDIM DA PENHA            0             1
        1  2016-04-29T00:00:00Z   56    JARDIM DA PENHA            0             0
        2  2016-04-29T00:00:00Z   62      MATA DA PRAIA            0             0
        3  2016-04-29T00:00:00Z    8  PONTAL DE CAMBURI            0             0
        4  2016-04-29T00:00:00Z   56    JARDIM DA PENHA            0             1

           Diabetes  Alcoholism  Handcap  SMS_received No-show
        0         0           0        0             0      No
        1         0           0        0             0      No
        2         0           0        0             0      No
        3         0           0        0             0      No
        4         1           0        0             0      No
```

```
In [8]: df.hist(figsize = (10,8));
```



2 Data Cleaning

```
In [4]: df.info()#hence ther is no missing data
        sum(df.duplicated()) #sum = 0 ;means there is no duplicated values
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 14 columns):
PatientId         110527 non-null float64
AppointmentID     110527 non-null int64
Gender            110527 non-null object
ScheduledDay      110527 non-null object
AppointmentDay    110527 non-null object
Age               110527 non-null int64
Neighbourhood     110527 non-null object
Scholarship       110527 non-null int64
Hipertension      110527 non-null int64
Diabetes          110527 non-null int64
Alcoholism        110527 non-null int64
Handcap           110527 non-null int64
SMS_received      110527 non-null int64
No-show           110527 non-null object
dtypes: float64(1), int64(8), object(5)
memory usage: 11.8+ MB
```

```
Out[4]: 0
```

## Exploratory Data Analysis

2.0.1 Rate of Hipertention patient

```
In [42]: counts = df['Hipertension'].value_counts()
         hiper_yes =int( counts[1])
         print('People suffering from hypertention =',hiper_yes)
         hiper_total = counts.sum()
         hiper_percentage = int(((hiper_yes/hiper_total)*100))
         print('Percentage = ' + str(hiper_percentage) +' %' )
```

```
People suffering from hypertention = 21801
Percentage = 19 %
```

2.0.2 Rate of Diabetes

```
In [22]: counts = df['Diabetes'].value_counts()
         D_yes = int(counts[1])
         print('People suffering from Diabetes =',D_yes)
         D_total = counts.sum()
         D_percentage=int(((D_yes/D_total)*100))
         print('Percentage = ' + str(D_percentage) +' %' )
```

```
People suffering from Diabetes = 7943
Percentage = 7%
```

2.0.3 Rate of Alcoholism

```
In [41]: counts = df['Alcoholism'].value_counts()
         Alcoholism_yes = int(counts[1])
         print('People suffering from Alcoholism =',Alcoholism_yes)
         Alcoholism_total = counts.sum()
         Alcoholism_percentage =  int(((Alcoholism_yes/Alcoholism_total)*100))
         print('Percentage = ' + str(Alcoholism_percentage) +' %' )
```

```
People suffering from Alcoholism = 3360
Percentage = 3 %
```

2.0.4 Rate of Handcap Appointed

```
In [39]: counts = df['Handcap'].value_counts()
         Handcap_yes = int(counts[1])
         print('People suffering from Handcap =',Handcap_yes)
         Handcap_total = counts.sum()
         Handcap_percentage =  int(((Handcap_yes/Handcap_total)*100))
         print('Percentage = ' + str(Handcap_percentage) +' %' )
```

```
People suffering from Handcap = 2042
Percentage = 1 %
```

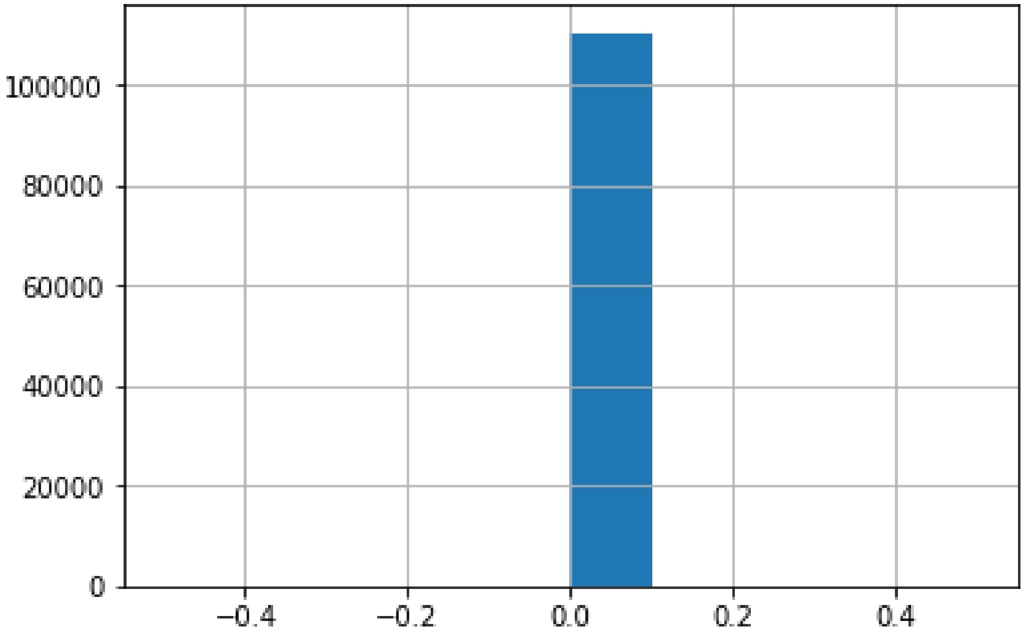
2.0.5 How many show up for their scheduled appointment?

```
In [9]: df1 = df['ScheduledDay'] == df['AppointmentDay']
        print(df1.sum())# 0 indicates that no scheduled day is matched with appointed day
```

```
0
```

```
In [10]: df1.hist()
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x7effefbc7518>
```

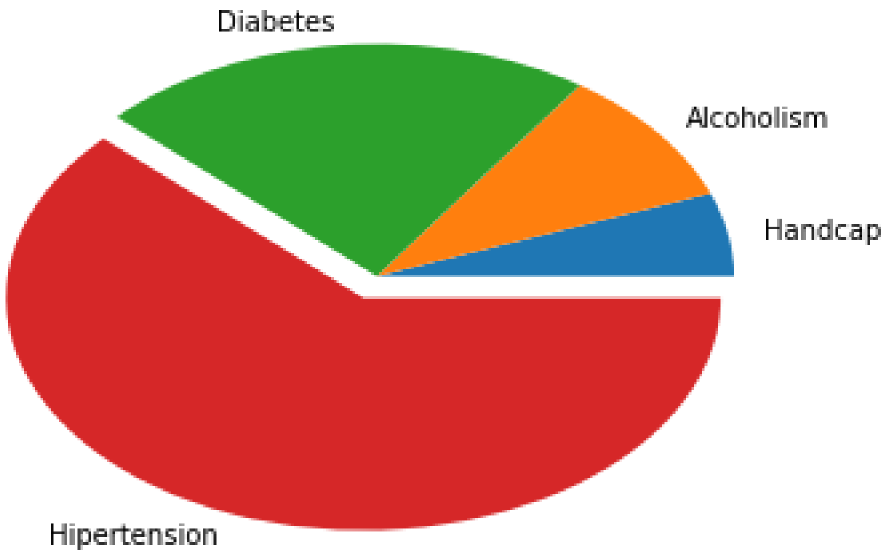


## Conclusions

In this project, we have calculated the number of patient appointed for particular type of dieseas

Below graph shows that maximum number of patient is suffering from hypertension

```
In [38]: labels = 'Handcap','Alcoholism','Diabetes','Hipertension'
         sizes = [Handcap_yes,Alcoholism_yes,D_yes,hiper_yes]
         explode = [0,0,0,0.1]
         plt.pie(sizes,explode = explode, labels = labels);
```



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