Investigate_a_Dataset

June 30, 2020

0.1 Introduction

I have chosen No-show appointment dataset in this project, which this dataset contains 100k medical appointment in Brazil, and show or describe the question why the patient no attend.

i have 4 questions in this project, I will analyze and answer it:

1-what is the average of the Age who attend to appointment?

2-is diabetes affects to attend to the appointment?

3-what is the average of the attendance who receive the SMS?

4-what is the percentage of the An alcoholic persons are not attend?

```
In [1]: # Import libraries
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        % matplotlib inline
```

Data Wrangling

0.1.1 General Properties

```
In [2]: #read the csv (dataset) and display it.
        df = pd.read_csv('noshowappointments-kagglev2-may-2016-1.csv')
        df.head()
```

Out[2]:

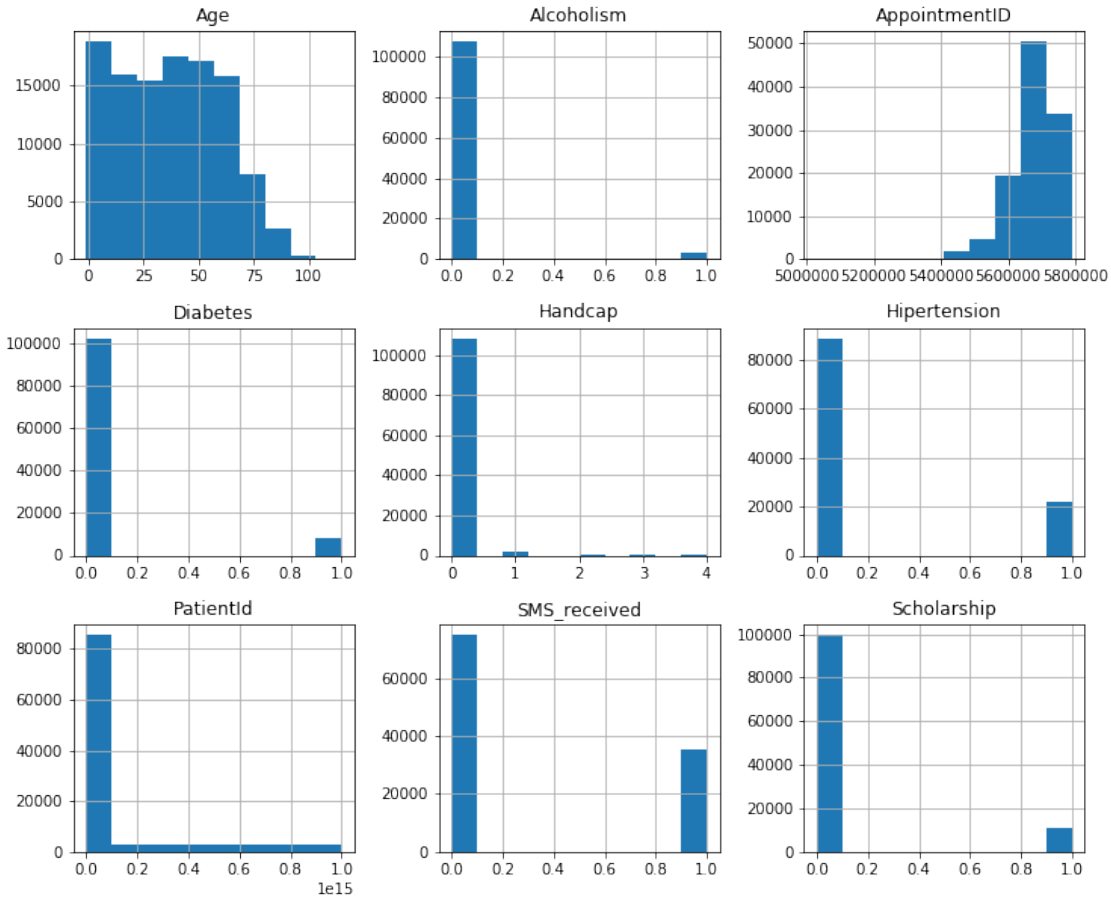
	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	0
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	0
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	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 [3]:
```
#show the information about the dataset, and if we looking here we don't have any missin
df.info()
```

```
<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
```

In [4]:
```
#here show us the histogram of each column
df.hist(figsize=(12,10));
```




```
In [5]: #here give us the shape of dataset
        df.shape
```

```
Out[5]: (110527, 14)
```

```
In [6]: #show us the number of duplicated value
        sum(df.duplicated())
```

```
Out[6]: 0
```

```
In [7]: #know the value for each column
        df.describe()
```

```
Out[7]:           PatientId  AppointmentID            Age    Scholarship  \
        count  1.105270e+05   1.105270e+05  110527.000000  110527.000000
        mean   1.474963e+14   5.675305e+06      37.088874       0.098266
        std    2.560949e+14   7.129575e+04      23.110205       0.297675
        min    3.921784e+04   5.030230e+06      -1.000000       0.000000
        25%    4.172614e+12   5.640286e+06      18.000000       0.000000
        50%    3.173184e+13   5.680573e+06      37.000000       0.000000
```

```
75%    9.439172e+13   5.725524e+06      55.000000       0.000000
max    9.999816e+14   5.790484e+06     115.000000       1.000000

        Hipertension       Diabetes     Alcoholism        Handcap  \
count  110527.000000  110527.000000  110527.000000  110527.000000
mean        0.197246       0.071865       0.030400       0.022248
std         0.397921       0.258265       0.171686       0.161543
min         0.000000       0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       0.000000
75%         0.000000       0.000000       0.000000       0.000000
max         1.000000       1.000000       1.000000       4.000000

        SMS_received
count  110527.000000
mean        0.321026
std         0.466873
min         0.000000
25%         0.000000
50%         0.000000
75%         1.000000
max         1.000000
```

0.1.2 Data Cleaning

```
In [8]: #drop the AppointmentID column, because i didn't need it
        df.drop(['AppointmentID'],axis=1,inplace=True)

In [9]: #show us the coloumn after drop the AppointmentID column
        df.describe()
```

```
Out[9]:           PatientId            Age    Scholarship   Hipertension  \
count  1.105270e+05  110527.000000  110527.000000  110527.000000
mean   1.474963e+14      37.088874       0.098266       0.197246
std    2.560949e+14      23.110205       0.297675       0.397921
min    3.921784e+04      -1.000000       0.000000       0.000000
25%    4.172614e+12      18.000000       0.000000       0.000000
50%    3.173184e+13      37.000000       0.000000       0.000000
75%    9.439172e+13      55.000000       0.000000       0.000000
max    9.999816e+14     115.000000       1.000000       1.000000

            Diabetes     Alcoholism        Handcap   SMS_received
count  110527.000000  110527.000000  110527.000000  110527.000000
mean        0.071865       0.030400       0.022248       0.321026
std         0.258265       0.171686       0.161543       0.466873
min         0.000000       0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       0.000000
```

```
        75%          0.000000       0.000000       0.000000       1.000000
        max          1.000000       1.000000       4.000000       1.000000
```

In [10]:
```
#show the number of duplicated value after drop
sum(df.duplicated())
```

Out[10]: 618

In [11]:
```
#drop duplicated value
df = df.drop_duplicates()
```

In [12]:
```
#show the number of duplicated value after drop it
sum(df.duplicated())
```

Out[12]: 0

In [13]:
```
#here i rename the No-show column to Attend, and display the modification
df.rename({'No-show':'Attend'},axis=1,inplace=True)
df.head()
```

Out[13]:

	PatientId	Gender	ScheduledDay	AppointmentDay	Age	\
0	2.987250e+13	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	
1	5.589978e+14	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	
2	4.262962e+12	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	
3	8.679512e+11	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	
4	8.841186e+12	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	

	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	\
0	JARDIM DA PENHA	0	1	0	0	
1	JARDIM DA PENHA	0	0	0	0	
2	MATA DA PRAIA	0	0	0	0	
3	PONTAL DE CAMBURI	0	0	0	0	
4	JARDIM DA PENHA	0	1	1	0	

	Handcap	SMS_received	Attend
0	0	0	No
1	0	0	No
2	0	0	No
3	0	0	No
4	0	0	No

In [14]:
```
#here i rename the SMS_received column to SMS, and display the modification
df.rename({'SMS_received':'SMS'},axis=1,inplace=True)
df.head()
```

Out[14]:

	PatientId	Gender	ScheduledDay	AppointmentDay	Age	\
0	2.987250e+13	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	
1	5.589978e+14	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	
2	4.262962e+12	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	

```
3  8.679512e+11      F  2016-04-29T17:29:31Z  2016-04-29T00:00:00Z    8
4  8.841186e+12      F  2016-04-29T16:07:23Z  2016-04-29T00:00:00Z   56

       Neighbourhood  Scholarship  Hipertension  Diabetes  Alcoholism  \
0    JARDIM DA PENHA            0             1         0           0
1    JARDIM DA PENHA            0             0         0           0
2      MATA DA PRAIA            0             0         0           0
3  PONTAL DE CAMBURI            0             0         0           0
4    JARDIM DA PENHA            0             1         1           0

   Handcap  SMS Attend
0        0    0     No
1        0    0     No
2        0    0     No
3        0    0     No
4        0    0     No
```

```
In [15]: #here i have count the value of attendance and no attendance
         df['Attend'].value_counts()
```

```
Out[15]: No     87804
         Yes    22105
         Name: Attend, dtype: int64
```

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

```
Out[16]: 54954.5
```

```
In [17]: #here i have change the value type of ScheduledDay and AppointmentDay the datetime
         cols = ['ScheduledDay','AppointmentDay']
         df[cols] = df[cols].apply(pd.to_datetime)
```

```
In [18]: #show the modification
         df.head()
```

```
Out[18]:       PatientId Gender        ScheduledDay AppointmentDay  Age  \
         0  2.987250e+13      F 2016-04-29 18:38:08     2016-04-29   62
         1  5.589978e+14      M 2016-04-29 16:08:27     2016-04-29   56
         2  4.262962e+12      F 2016-04-29 16:19:04     2016-04-29   62
         3  8.679512e+11      F 2016-04-29 17:29:31     2016-04-29    8
         4  8.841186e+12      F 2016-04-29 16:07:23     2016-04-29   56

                Neighbourhood  Scholarship  Hipertension  Diabetes  Alcoholism  \
         0    JARDIM DA PENHA            0             1         0           0
         1    JARDIM DA PENHA            0             0         0           0
         2      MATA DA PRAIA            0             0         0           0
         3  PONTAL DE CAMBURI            0             0         0           0
         4    JARDIM DA PENHA            0             1         1           0
```

```
   Handcap  SMS Attend
0        0    0     No
1        0    0     No
2        0    0     No
3        0    0     No
4        0    0     No
```

```
In [19]: #show us the dataset information after modifications above
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 109909 entries, 0 to 110526
Data columns (total 13 columns):
PatientId         109909 non-null float64
Gender            109909 non-null object
ScheduledDay      109909 non-null datetime64[ns]
AppointmentDay    109909 non-null datetime64[ns]
Age               109909 non-null int64
Neighbourhood     109909 non-null object
Scholarship       109909 non-null int64
Hipertension      109909 non-null int64
Diabetes          109909 non-null int64
Alcoholism        109909 non-null int64
Handcap           109909 non-null int64
SMS               109909 non-null int64
Attend            109909 non-null object
dtypes: datetime64[ns](2), float64(1), int64(7), object(3)
memory usage: 11.7+ MB
```

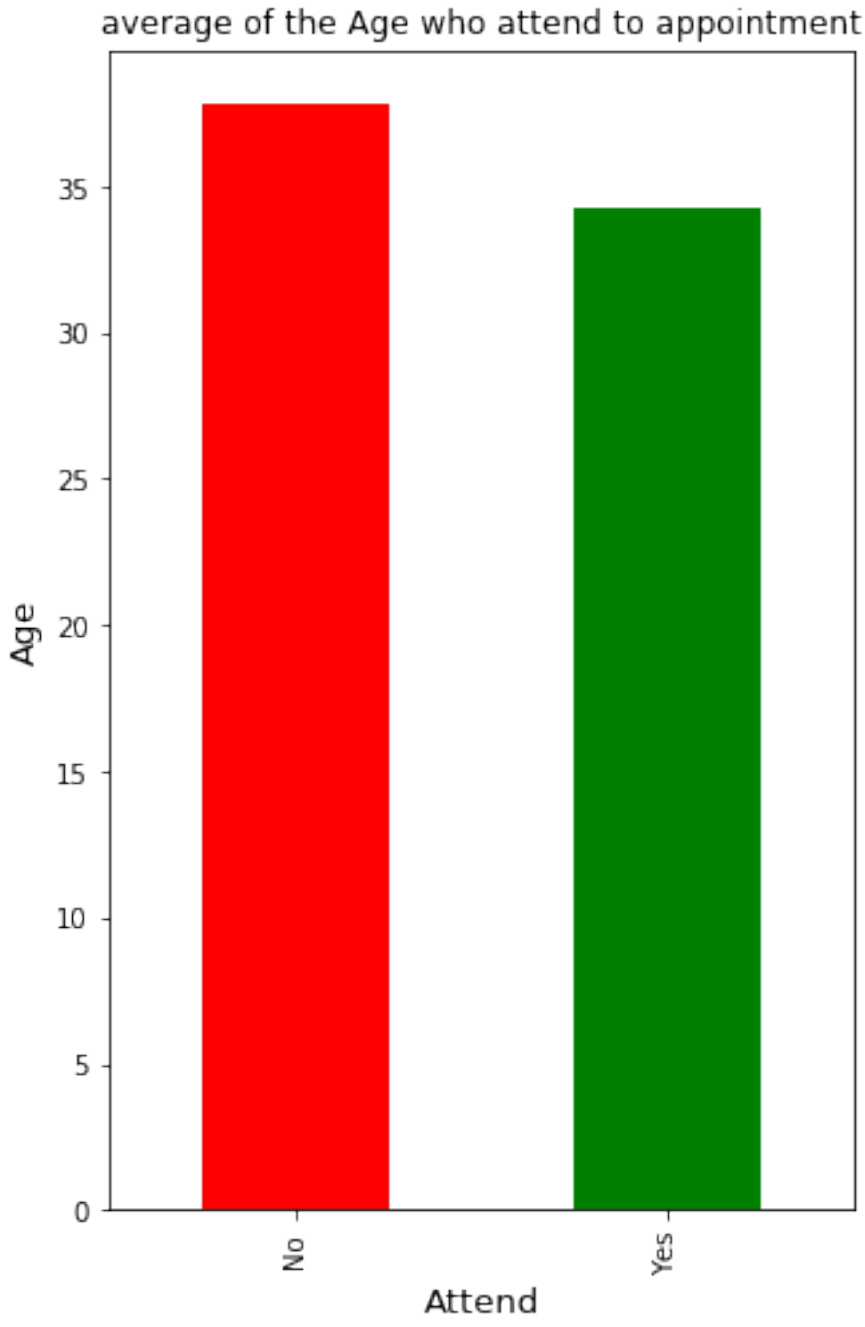
```
In [20]: #show us the number of null value for each column
         df.isnull().sum()
```

```
Out[20]: PatientId         0
         Gender            0
         ScheduledDay      0
         AppointmentDay    0
         Age               0
         Neighbourhood     0
         Scholarship       0
         Hipertension      0
         Diabetes          0
         Alcoholism        0
         Handcap           0
         SMS               0
         Attend            0
         dtype: int64
```

1 Exploratory Data Analysis

1.1 Research Question 1 (what is the average of the Age who attend to appointment?)

In [21]:
```
#Q1 sol:
#show us the average of the Age who attend to appointment
av=df.groupby('Attend')['Age'].mean().plot(kind='bar',figsize=(5,8),title='average of t
av.set_xlabel("Attend",fontsize=13);
av.set_ylabel("Age",fontsize=13);
```

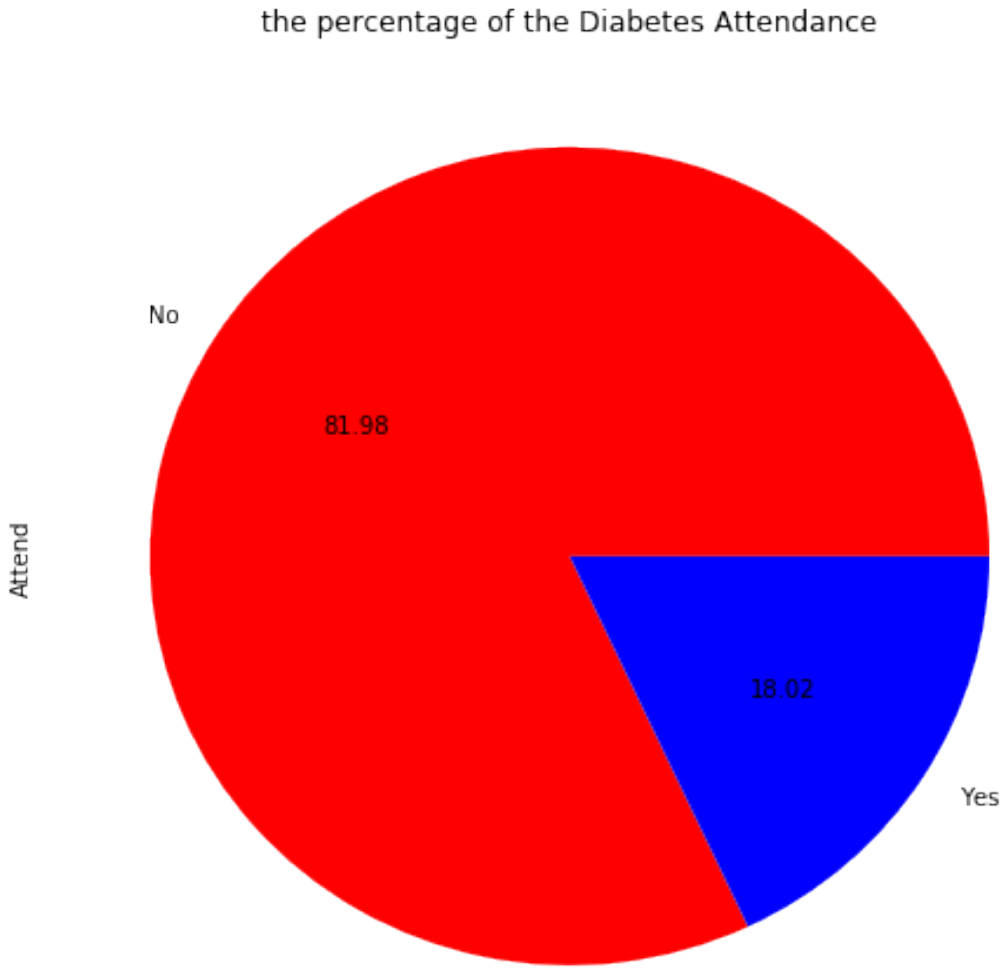


1.1.1 Findings:

we see above the average of the Age who no attend to Appointment are higher than who attend to Appointment.

1.2 Research Question 2 (is diabetes effects to attend to appointment?)

```
In [22]: #Q2 sol:
         #show us the percentage of the Diabetes Attendance, and look if is it affect to attend
         mtitle = 'the percentage of the Diabetes Attendance'
         NO = df[df['Diabetes'] == 0]
         Yes = df[df['Diabetes'] == 1]
         per=Yes.Attend.value_counts().plot(kind='pie',figsize=(8,8),colors=['red','blue'],title
```

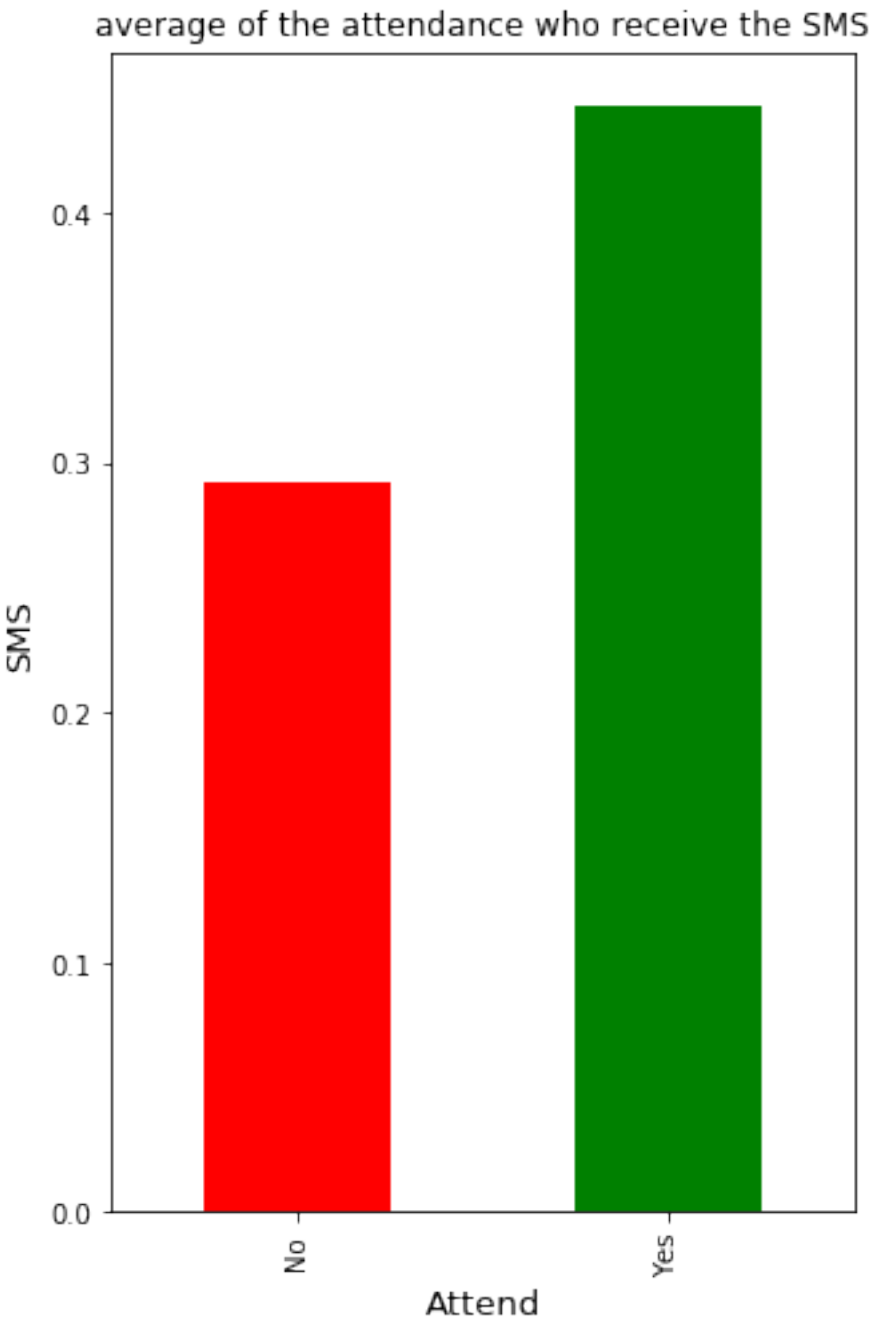


1.2.1 Findings:

we see above the diabetes will be affect to attend to the appointment. so the People with diabetes who no attend to the appointment by 81.98%.

2 Research Question 3 (what is the average of the attendance who receive the SMS?)

```
In [23]: #Q3 sol:
         #show us the average of the attendance who receive the SMS
         av=df.groupby('Attend')['SMS'].mean().plot(kind='bar',figsize=(5,8),title='average of t
         av.set_xlabel("Attend",fontsize=13);
         av.set_ylabel("SMS",fontsize=13);
```

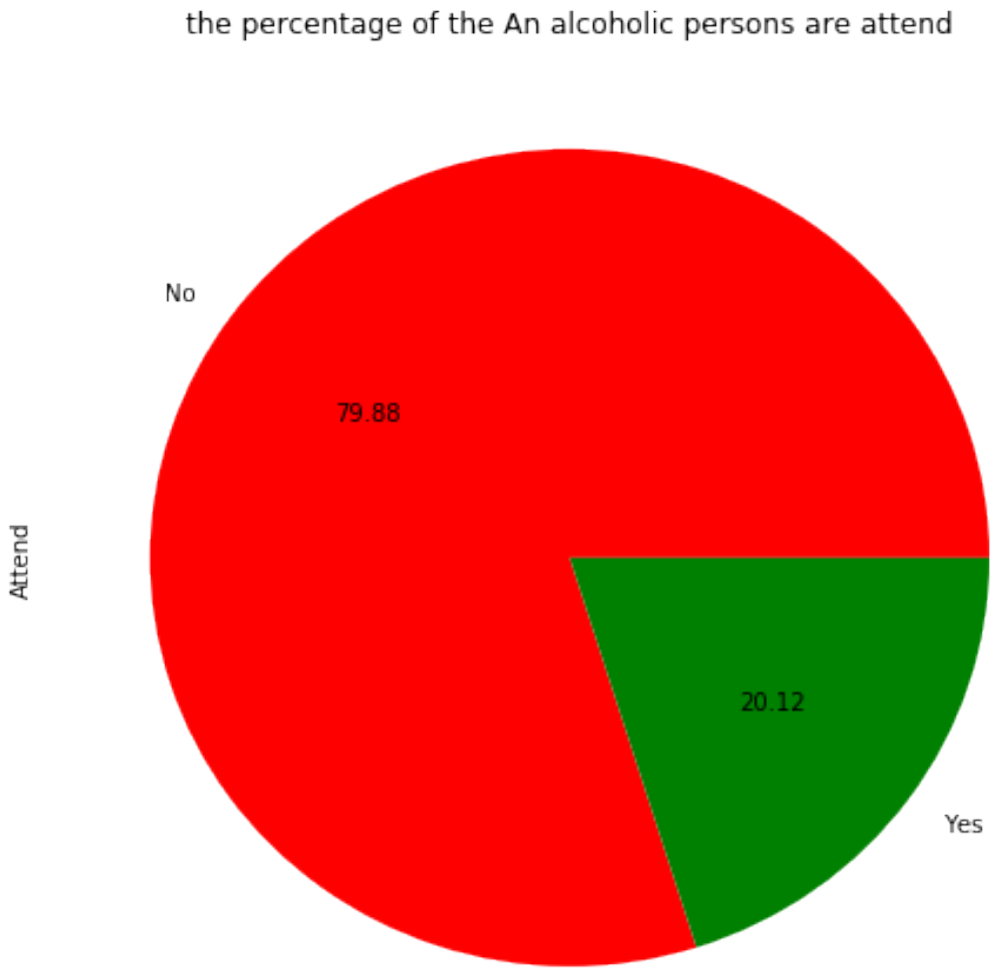


2.0.1 Findings:

we see above the Patients who receive the SMS and attend to the appointment are higher than who receive the SMS and no attend to the appoinment.

3 Research Question 4 (what is the percentage of the An alcoholic persons are not attend?)

```
In [28]: #Q4 sol:
         #show us the percentage of the An alcoholic persons attendance, and look if is it affec
         mtitle = 'the percentage of the An alcoholic persons are attend'
         NO = df[df['Alcoholism'] == 0]
         Yes = df[df['Alcoholism'] == 1]
         per=NO.Attend.value_counts().plot(kind='pie',figsize=(8,8),colors=['red','green'],title
```



3.0.1 Findings:

we see above the alcoholic will be affect to attend to the appointment. so the People who drink the alcoholic no attend to the appointment by 79.88%.

Conclusions After analyzing the data and the questions above we saw many facts that might be affect to attend to the appointment, some of this facts:

1- we saw above in the question 2 the diabetes might be affect to no attend to appointment by 81.98%.

2- we saw above in the question 4 the alcoholic might be affect to no attend to appointment by 79.88%.

-we saw above in the question 1 the average of age who attend to the appointment.

-we saw above in the question 3 the average of the attendance who receive the SMS message, the person who recieve the SMS message and attend to the appointment most than the person who recieve the SMS message and no attend to the appointment.

3.1 limitations

in the analyzing above, I was hoping to have more information like for example the departments name in the hospital can the Patient attend to take treatment, and give some diseases that might be afect to attend to the appointment like for example heart disease and Pressure disease.

i think the result or the information above only limit to one place or hospital so you can't depend on it.

```
In [25]: df.to_csv('noshowappointments-kagglev2-may-2016.csv', index=False)

In [ ]:

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

Out[1]: 0

In [ ]:
```