Project: Examining No-show appointments Dataset

Table of Contents

Introduction

> This dataset collects information from 100,000 plus medical appointments in Brazil and is focused on the question of whether the patients show up for their appointment or not. Here, we will be studying the features which will determine the response to appointments.

> For analysing this data set, the very first process we will do is frame questions which we can answer from the information given in the dataset.

Asking Questions:

> 1.Which age group visits the doctor more often?

> 2.Who shows up for the appoinment most of the times among males and females?

> 3.Do the patients show up for the hospitals loacted in specific areas?

> 4.What factors are important for us to know in order to predict if a patient will show up for their scheduled appointment?

In [82]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Data Wrangling

> After framing our questions,we need to wrangle our data to make sure all the data we need is in great quality. Three steps are involved in Data Wrangling.

1st Step: Gathering of Data.

In [ ]:

```
df=pd.read_csv('C:/Users/somya/Desktop/Investigate/noshowappointments-kagglev2-may-2016.csv
df.head()
```

2nd Step: Assessing of Data.

> We will find out problems in data's quality or structure.

In [84]:

```
df.shape
```

Out[84]:

```
(110527, 14)
```

In [85]:

```
df.duplicated().sum()
```

Out[85]:

```
0
```

In [86]:

```
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 [87]:

```
df.dtypes
```

Out[87]:

```
PatientId         float64
AppointmentID       int64
Gender             object
ScheduledDay       object
AppointmentDay     object
Age                 int64
Neighbourhood      object
Scholarship         int64
Hipertension        int64
Diabetes            int64
Alcoholism          int64
Handcap             int64
SMS_received        int64
No-show            object
dtype: object
```

In [ ]:

```
df.columns
```

In [88]:

```
df.describe()
```

Out[88]:

	PatientId	AppointmentID	Age	Scholarship	Hipertension	Diabete
count	1.105270e+05	1.105270e+05	110527.000000	110527.000000	110527.000000	110527.00000
mean	1.474963e+14	5.675305e+06	37.088874	0.098266	0.197246	0.07186
std	2.560949e+14	7.129575e+04	23.110205	0.297675	0.397921	0.25826
min	3.921784e+04	5.030230e+06	-1.000000	0.000000	0.000000	0.00000
25%	4.172614e+12	5.640286e+06	18.000000	0.000000	0.000000	0.00000
50%	3.173184e+13	5.680573e+06	37.000000	0.000000	0.000000	0.00000
75%	9.439172e+13	5.725524e+06	55.000000	0.000000	0.000000	0.00000
max	9.999816e+14	5.790484e+06	115.000000	1.000000	1.000000	1.00000

In [89]:

```
df.nunique()
```

Out[89]:

```
PatientId          62299
AppointmentID     110527
Gender                 2
ScheduledDay      103549
AppointmentDay        27
Age                  104
Neighbourhood         81
Scholarship            2
Hipertension           2
Diabetes               2
Alcoholism             2
Handcap                5
SMS_received           2
No-show                2
dtype: int64
```

Observations after assessment of Data-

There are no duplicate rows in the dataset.

There are no missing values in the data.

Some of the datatypes are incorrect.

> Some of the column names are wrongly spelled.

3rd Step :Data Cleaning

> First we will drop the columns which we will not use to predict the no-show appointments.

In [90]:

```
df.drop(['PatientId','AppointmentID'],axis=1,inplace=True)
```

In [91]:

```
df = df[(df['Age'] > 0) & (df['Age'] < 95)]
df.shape
```

Out[91]:

```
(106917, 12)
```

```
> Now, we will convert the incorrect datatype of scheduled day and appointment day which
is given as String to dateTime.
> Also, we observed that their is no time specified for Appointment day so we don't need
time for predicting.
  Hence, we will remove time from the rows.
```

In [92]:

```
df['ScheduledDay']=pd.to_datetime(df['ScheduledDay']).dt.date
```

In [93]:

```
df['AppointmentDay']=pd.to_datetime(df['AppointmentDay']).dt.date
df.head()
```

Out[93]:

	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	D
0	F	2016-04-29	2016-04-29	62	JARDIM DA PENHA	0	1	
1	M	2016-04-29	2016-04-29	56	JARDIM DA PENHA	0	0	
2	F	2016-04-29	2016-04-29	62	MATA DA PRAIA	0	0	
3	F	2016-04-29	2016-04-29	8	PONTAL DE CAMBURI	0	0	
4	F	2016-04-29	2016-04-29	56	JARDIM DA PENHA	0	1	

Some of the columns are spelled wrongly.We will correct them first in order to avoid confusion afterwards.

In [94]:

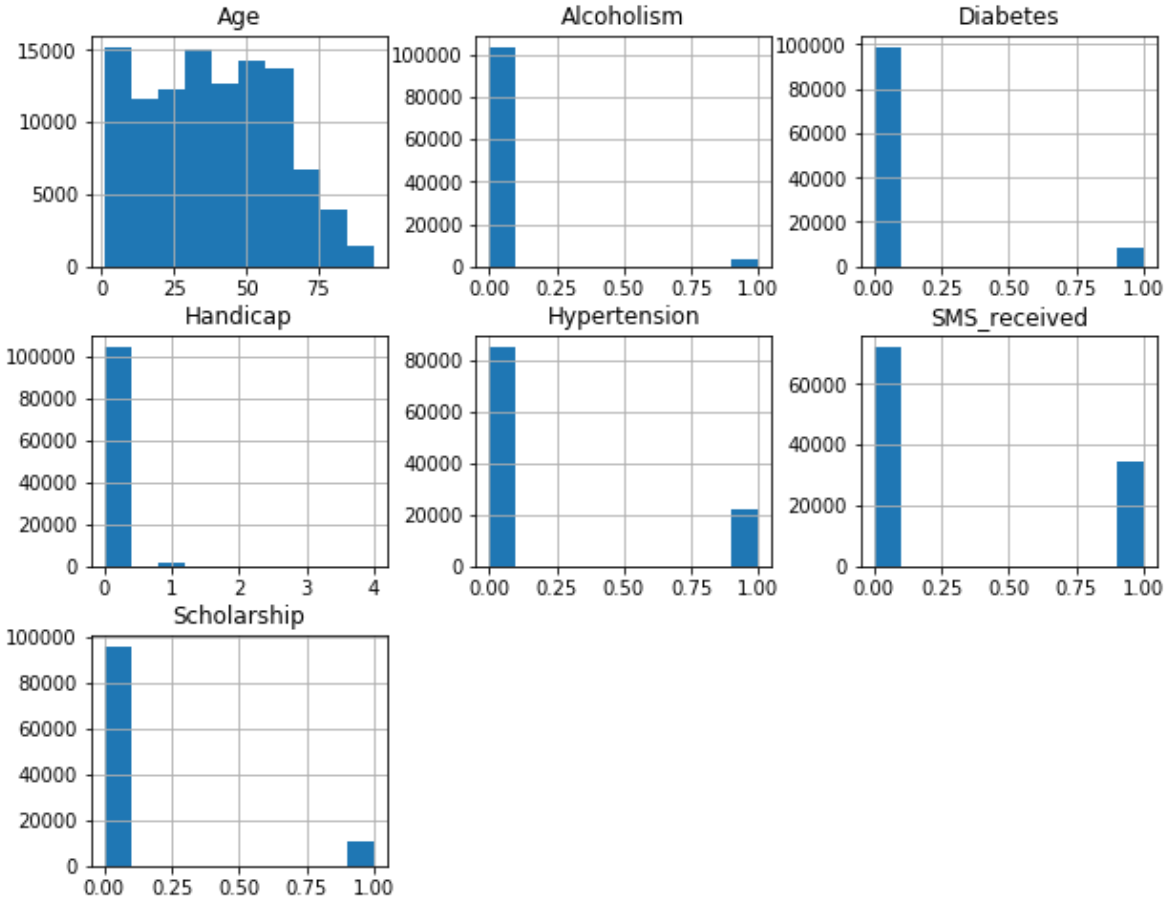
```
df=df.rename(columns={'Handcap': 'Handicap', 'Hipertension': 'Hypertension','No-show':'NoSh

df.head(1)
```

Out[94]:

	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hypertension	D
0	F	2016-04-29	2016-04-29	62	JARDIM DA PENHA	0	1	

In [95]:

```
df.hist(figsize=(10,8));
```



Exploratory Data Analysis

Research Question 1 : Which age group visits the doctor more often?

In [96]:

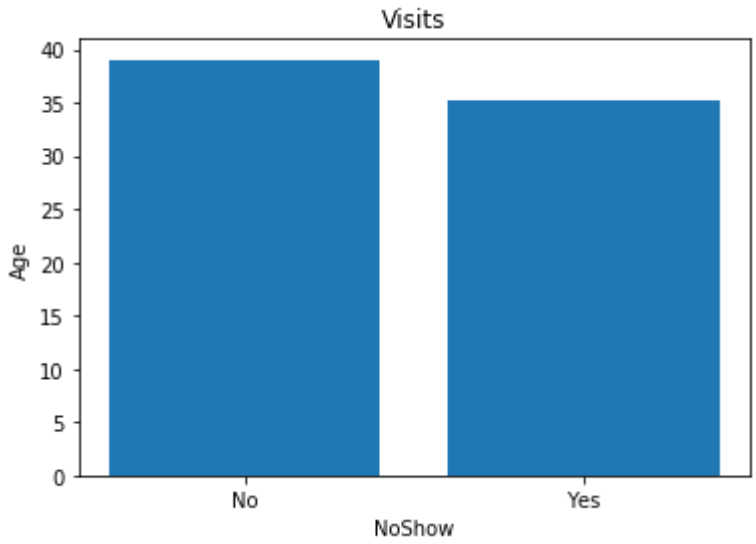
```
df.groupby('NoShow').mean().Age
```

Out[96]:

```
NoShow
No     39.036305
Yes    35.290211
Name: Age, dtype: float64
```

In [97]:

```
plt.bar(['No', 'Yes'], [39.036, 35.290]);
plt.title('Visits')
plt.xlabel('NoShow')
plt.ylabel('Age');
```



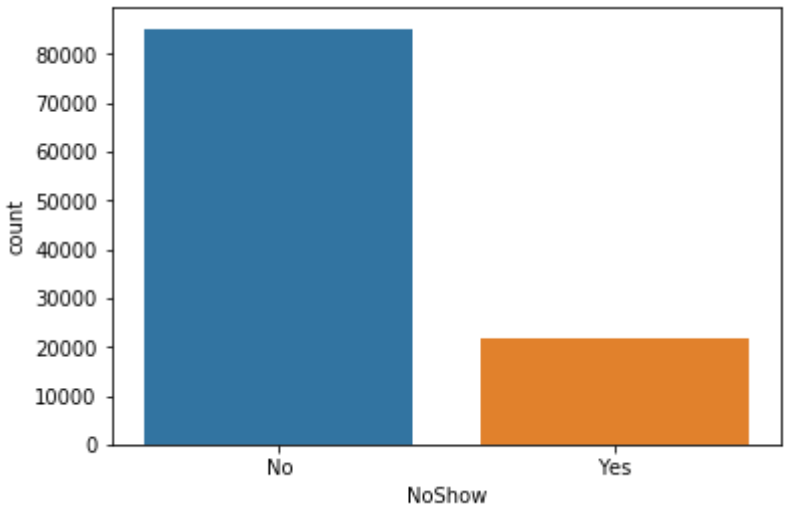
Research Question 2 : Who shows up for the appointment most of the times among males and females?

In [98]:

```
sns.countplot(x="NoShow", data=df)
```

Out[98]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1868710a9e8>
```



In [99]:

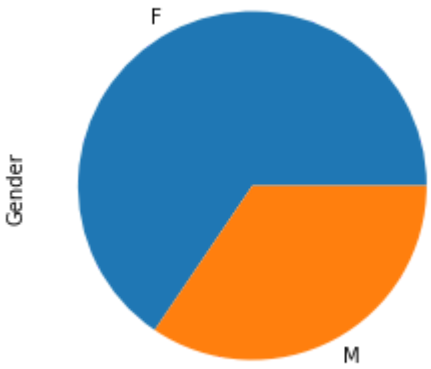
```
df['Gender'].value_counts(normalize=True)
```

Out[99]:

```
F    0.655303
M    0.344697
Name: Gender, dtype: float64
```

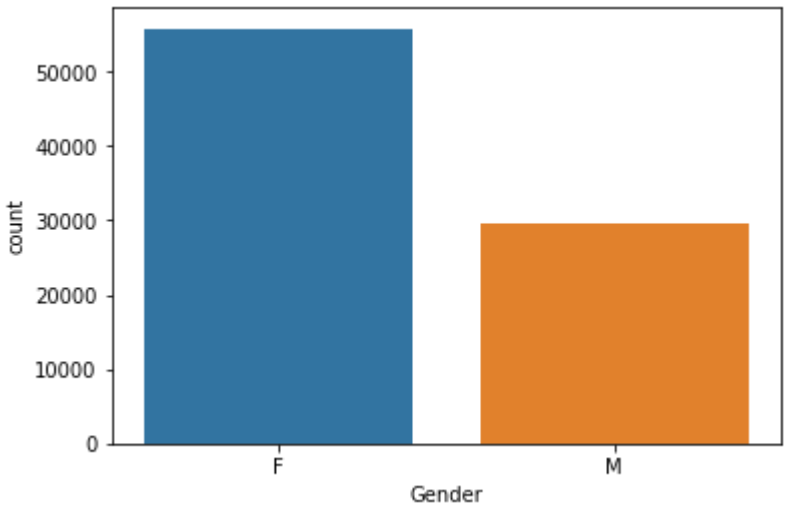
In [100]:

```
df['Gender'].value_counts(normalize=True).plot(kind='pie');
```



In [101]:

```
shows_up = df[df['NoShow'] == 'No']
sns.countplot(x='Gender', data=shows_up);
```



Research Question 3: Do the patients show up for the hospitals loacted in specific areas?

In [102]:

```
shows = df[df['NoShow'] == 'No']
shows.groupby('Neighbourhood')['NoShow'].count().sort_values(ascending=False).head(10)
```

Out[102]:

```
Neighbourhood
JARDIM CAMBURI     6149
MARIA ORTIZ        4367
RESISTÊNCIA        3360
JARDIM DA PENHA    3220
CENTRO             2586
SANTA MARTHA       2547
ITARARÉ            2514
TABUAZEIRO         2465
SANTO ANTÔNIO      2195
BONFIM             2161
Name: NoShow, dtype: int64
```

Research Question 4: What factors are important for us to know in order to predict if a patient will show up for their scheduled appointment?

In [104]:

```
df.groupby('NoShow').mean().Handicap
```

Out[104]:

```
NoShow
No     0.02339
Yes    0.02063
Name: Handicap, dtype: float64
```

In [105]:

```
df.groupby('NoShow').mean().Scholarship
```

Out[105]:

```
NoShow
No     0.096633
Yes    0.118660
Name: Scholarship, dtype: float64
```

In [106]:

```
df.groupby('NoShow').mean().Hypertension
```

Out[106]:

```
NoShow
No     0.211144
Yes    0.173674
Name: Hypertension, dtype: float64
```

In [107]:

```
df.groupby('NoShow').mean().Diabetes
```

Out[107]:

```
NoShow
No     0.076317
Yes    0.065861
Name: Diabetes, dtype: float64
```

In [108]:

```
df.groupby('NoShow').mean().SMS_received
```

Out[108]:

```
NoShow
No     0.293361
Yes    0.441455
Name: SMS_received, dtype: float64
```

In [109]:

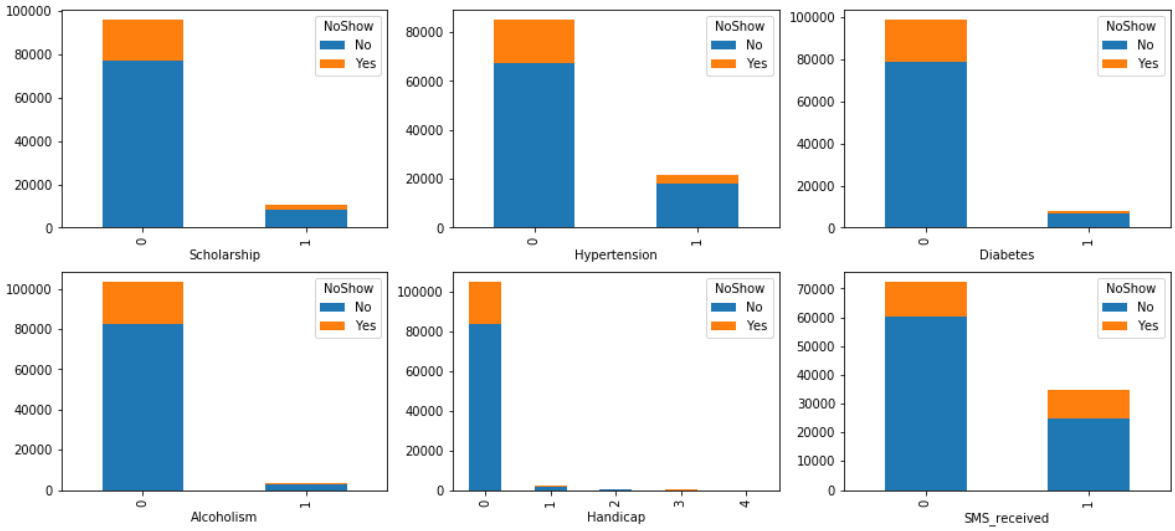
```
df.groupby('NoShow').mean().Alcoholism
```

Out[109]:

```
NoShow
No     0.031472
Yes    0.031246
Name: Alcoholism, dtype: float64
```

In [114]:

```
categories = [ 'Scholarship', 'Hypertension', 'Diabetes', 'Alcoholism', 'Handicap', 'SMS_re

fig = plt.figure(figsize=(16, 11))
for i, v in enumerate(categories):
    ax = fig.add_subplot(3, 3, i+1)
    df.groupby([v, 'NoShow'])[v].count().unstack('NoShow').plot(ax=ax, kind='bar', stacked=
```



Conclusions

Tip: Finally, summarize your findings and the results that have been performed. Make sure that you are clear with regards to the limitations of your exploration. If you haven't done any statistical tests, do not imply any statistical conclusions. And make sure you avoid implying causation from correlation!

Tip: Once you are satisfied with your work, you should save a copy of the report in HTML or PDF form via the **File** > **Download as** submenu. Before exporting your report, check over it to make sure that the flow of the report is complete. You should probably remove all of the "Tip" quotes like this one so that the presentation is as tidy as possible. Congratulations!

In [ ]: