Medical Appointment No-Show

In this project, we will analyze the no-show apointments.This dataset is from kaggle.We will find the factors which affects the patient's no-show for a scheduled appointment. This dataset contains 14 columns. 01 - PatientId: Identification of a patient 02 - AppointmentID: Identification of each appointment 03 - Gender: Male or Female . 04 - ScheduledDay: is the day someone called or registered the appointment, this is before appointment 05 - Appointment day: is the day of the actual appointment 06 - Age: How old is the patient. 07 - Neighbourhood: Where the appointment takes place. 08 - Scholarship: True of False . 09 - Hipertension: True or False 10 - Diabetes: True or False 11 - Alcoholism: True or False 12 - Handcap: True or False 13 - SMS_received: 1 or more messages sent to the patient. 14 - No-show: True or False.

How to run the code

This is an executable *Jupyter notebook* hosted on Jovian.ml, a platform for sharing data science projects. You can run and experiment with the code in a couple of ways: *using free online resources* (recommended) or *on your own computer*.

Option 1: Running using free online resources (1-click, recommended)

The easiest way to start executing this notebook is to click the "Run" button at the top of this page, and select "Run on Binder". This will run the notebook on mybinder.org, a free online service for running Jupyter notebooks. You can also select "Run on Colab" or "Run on Kaggle".

Option 2: Running on your computer locally

1. Install Conda by following these instructions. Add Conda binaries to your system `PATH`, so you can use the `conda` command on your terminal.
2. Create a Conda environment and install the required libraries by running these commands on the terminal:

   ```
   conda create -n zerotopandas -y python=3.8
   conda activate zerotopandas
   pip install jovian jupyter numpy pandas matplotlib seaborn opendatasets --upgrade
   ```

3. Press the "Clone" button above to copy the command for downloading the notebook, and run it on the terminal. This will create a new directory and download the notebook. The command will look something like this:

   ```
   jovian clone notebook-owner/notebook-id
   ```

4. Enter the newly created directory using `cd directory-name` and start the Jupyter notebook.

   ```
   jupyter notebook
   ```

You can now access Jupyter's web interface by clicking the link that shows up on the terminal or by visiting http://localhost:8888 on your browser. Click on the notebook file (it has a `.ipynb` extension) to open it.

Downloading the Dataset

Dataset Link - https://www.kaggle.com/datasets/joniarroba/noshowappointments

```
!pip install jovian opendatasets --upgrade --quiet
```

Let's begin by downloading the data, and listing the files within the dataset.

```
dataset_url = 'https://www.kaggle.com/datasets/joniarroba/noshowappointments'
```

```
import opendatasets as od
od.download(dataset_url)
```

The dataset has been downloaded and extracted.

```
data_dir = './noshowappointments'
```

```
import os
os.listdir(data_dir)
```

```
['KaggleV2-May-2016.csv']
```

```
project_name = "medical-appointment-no-show"
```

```
!pip install jovian --upgrade -q
```

Data Preparation and Cleaning

- As can be seen from the result of .info method, there is no null values in the dataset.
- We will modify the date and time values from ScheduledDay and AppointmentDay column into standard form and add 2 columns ScheduledDay_weekday and AppointmentDay_weekday, which will store the weekdays.
- Rename the columns Hipertension,Handcap.
- Drop Columns PatientID, AppointmentID, Neighbourhood.

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

```
appointment_df = pd.read_csv('./noshowappointments/KaggleV2-May-2016.csv')
```

```
appointment_df
```

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	H
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0
...	...	...	...	...	...	...	...	...
110522	2.572134e+12	5651768	F	2016-05-03T09:15:35Z	2016-06-07T00:00:00Z	56	MARIA ORTIZ	0
110523	3.596266e+12	5650093	F	2016-05-03T07:27:33Z	2016-06-07T00:00:00Z	51	MARIA ORTIZ	0
110524	1.557663e+13	5630692	F	2016-04-27T16:03:52Z	2016-06-07T00:00:00Z	21	MARIA ORTIZ	0
110525	9.213493e+13	5630323	F	2016-04-27T15:09:23Z	2016-06-07T00:00:00Z	38	MARIA ORTIZ	0
110526	3.775115e+14	5629448	F	2016-04-27T13:30:56Z	2016-06-07T00:00:00Z	54	MARIA ORTIZ	0

110527 rows × 14 columns

```
appointment_df.shape
```

```
(110527, 14)
```

```
appointment_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   PatientId       110527 non-null  float64
 1   AppointmentID   110527 non-null  int64
 2   Gender          110527 non-null  object
 3   ScheduledDay    110527 non-null  object
 4   AppointmentDay  110527 non-null  object
 5   Age             110527 non-null  int64
 6   Neighbourhood   110527 non-null  object
 7   Scholarship     110527 non-null  int64
 8   Hipertension    110527 non-null  int64
 9   Diabetes        110527 non-null  int64
 10  Alcoholism      110527 non-null  int64
 11  Handcap         110527 non-null  int64
 12  SMS_received    110527 non-null  int64
 13  No-show         110527 non-null  object
```

```
dtypes: float64(1), int64(8), object(5)
memory usage: 11.8+ MB
```

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

```
appointment_df.head()
```

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hiperte
0	2.987250e+13	5642903	F	2016-04-29	2016-04-29	62	JARDIM DA PENHA	0	
1	5.589978e+14	5642503	M	2016-04-29	2016-04-29	56	JARDIM DA PENHA	0	
2	4.262962e+12	5642549	F	2016-04-29	2016-04-29	62	MATA DA PRAIA	0	
3	8.679512e+11	5642828	F	2016-04-29	2016-04-29	8	PONTAL DE CAMBURI	0	
4	8.841186e+12	5642494	F	2016-04-29	2016-04-29	56	JARDIM DA PENHA	0	

```
appointment_df['ScheduledDay_weekday'] = appointment_df['ScheduledDay'].dt.dayofweek
appointment_df['AppointmentDay_weekday'] = appointment_df['AppointmentDay'].dt.dayofwee
```

```
appointment_df['ScheduledDay_weekday'].value_counts()
```

```
1    26168
2    24262
0    23085
4    18915
3    18073
5       24
Name: ScheduledDay_weekday, dtype: int64
```

```
appointment_df['AppointmentDay_weekday'].value_counts()
```

```
2    25867
1    25640
0    22715
4    19019
3    17247
5       39
Name: AppointmentDay_weekday, dtype: int64
```

```
appointment_df.columns
```

```
Index(['PatientId', 'AppointmentID', 'Gender', 'ScheduledDay',
       'AppointmentDay', 'Age', 'Neighbourhood', 'Scholarship', 'Hipertension',
       'Diabetes', 'Alcoholism', 'Handcap', 'SMS_received', 'No-show',
       'ScheduledDay_weekday', 'AppointmentDay_weekday'],
      dtype='object')
```

```
appointment_df = appointment_df.rename(columns={'Hipertension':'Hypertension', 'Handcap
```

```
appointment_df.drop(['PatientId', 'AppointmentID', 'Neighbourhood'], axis=1, inplace =
```

```
appointment_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 13 columns):
 #   Column                  Non-Null Count   Dtype         
---  ------                  --------------   -----         
 0   Gender                  110527 non-null  object        
 1   ScheduledDay            110527 non-null  datetime64[ns]
 2   AppointmentDay          110527 non-null  datetime64[ns]
 3   Age                     110527 non-null  int64         
 4   Scholarship             110527 non-null  int64         
 5   Hypertension            110527 non-null  int64         
 6   Diabetes                110527 non-null  int64         
 7   Alcoholism              110527 non-null  int64         
 8   Handicap                110527 non-null  int64         
 9   SMS_received            110527 non-null  int64         
 10  No-show                 110527 non-null  object        
 11  ScheduledDay_weekday    110527 non-null  int64         
 12  AppointmentDay_weekday  110527 non-null  int64         
dtypes: datetime64[ns](2), int64(9), object(2)
memory usage: 11.0+ MB
```

Exploratory Analysis and Visualization

Let's begin by importing `matplotlib.pyplot` and `seaborn`.

```
import seaborn as sns
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline

sns.set_style('darkgrid')
matplotlib.rcParams['font.size'] = 14
matplotlib.rcParams['figure.figsize'] = (9, 5)
matplotlib.rcParams['figure.facecolor'] = '#00000000'
```

```
appointment_df.describe()
```

	Age	Scholarship	Hypertension	Diabetes	Alcoholism	Handicap	SMS_re
count	110527.000000	110527.000000	110527.000000	110527.000000	110527.000000	110527.000000	110527.0
mean	37.088874	0.098266	0.197246	0.071865	0.030400	0.022248	0.3
std	23.110205	0.297675	0.397921	0.258265	0.171686	0.161543	0.4
min	-1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	18.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
50%	37.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
75%	55.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.0
max	115.000000	1.000000	1.000000	1.000000	1.000000	4.000000	1.0

- As we can see from above statistics that Age column has value less than 1 and greater than 100.
- Handicap column has value greater than 1.

We will drop these values.

```
appointment_df = appointment_df[(appointment_df['Age'] >= 0) & (appointment_df['Age'] <
```

```
appointment_df['Age'].value_counts()
```

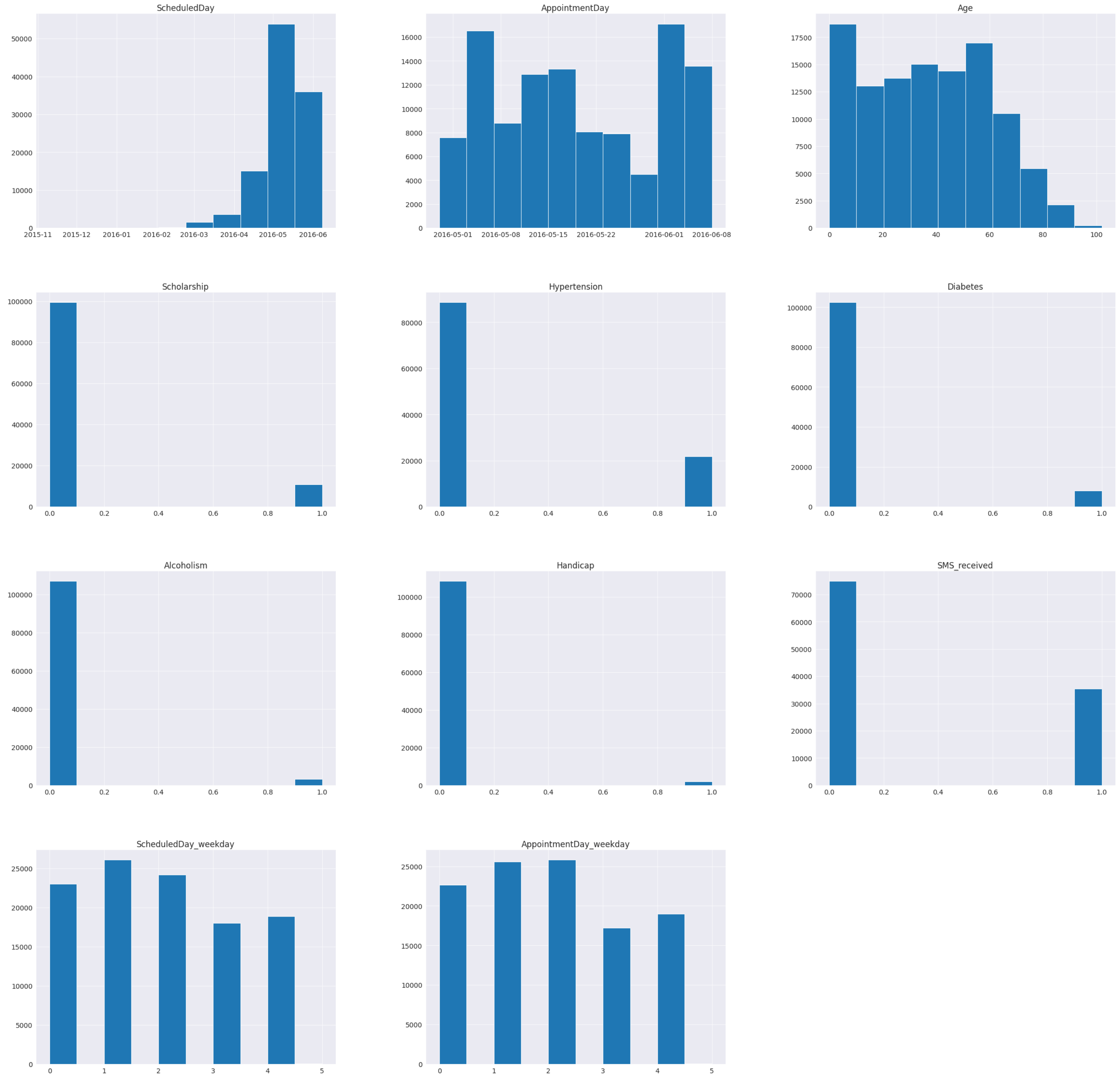
```
0      3539
1      2273
52     1746
49     1652
53     1651
       ...
97       11
98        6
100       4
102       2
99        1
Name: Age, Length: 102, dtype: int64
```

```
appointment_df = appointment_df[appointment_df['Handicap'] <= 1]
```

```
appointment_df['Handicap'].value_counts()
```

```
0    108284
1      2038
Name: Handicap, dtype: int64
```

```
appointment_df.hist(figsize=(40,40));
```



```
youth_count = appointment_df[(appointment_df['Age'] >= 0) & (appointment_df['Age'] < 40
old_count = appointment_df[appointment_df['Age'] > 40].shape[0]
youth_count, old_count
```

```
(59195, 49727)
```

```
non_hypertension_count = appointment_df[appointment_df['Hypertension'] == 0].shape[0]
hypertension_count = appointment_df[appointment_df['Hypertension'] == 1].shape[0]
percent_hypertension = round(hypertension_count / appointment_df.shape[0] * 100)
non_hypertension_count, hypertension_count, percent_hypertension
```

```
(88607, 21715, 20)
```

```
non_diabetic = appointment_df[appointment_df['Diabetes'] == 0].shape[0]
diabetic = appointment_df[appointment_df['Diabetes'] == 1].shape[0]
non_diabetic, diabetic
```

```
(102422, 7900)
```

```
non_alcoholic = appointment_df[appointment_df['Alcoholism'] == 0].shape[0]
alcoholic = appointment_df[appointment_df['Alcoholism'] == 1].shape[0]
percent_alcohol = round(alcoholic / appointment_df.shape[0] * 100)
non_alcoholic, alcoholic, percent_alcohol
```

```
(106970, 3352, 3)
```

```
non_handicap = appointment_df[appointment_df['Handicap'] == 0].shape[0]
handicap = appointment_df[appointment_df['Handicap'] == 1].shape[0]
non_handicap, handicap
```

```
(108284, 2038)
```

```
sms_received = appointment_df[appointment_df['SMS_received'] == 1].shape[0]
sms_not_received = appointment_df[appointment_df['SMS_received'] == 0].shape[0]
percent_sms_not_rec = round(sms_not_received / appointment_df.shape[0] * 100)
sms_received, sms_not_received, percent_sms_not_rec
```

```
(35434, 74888, 68)
```

Asking and Answering Questions

Q1: Which gender visit hospital more?

```
appointment_df['Gender'].value_counts()
```

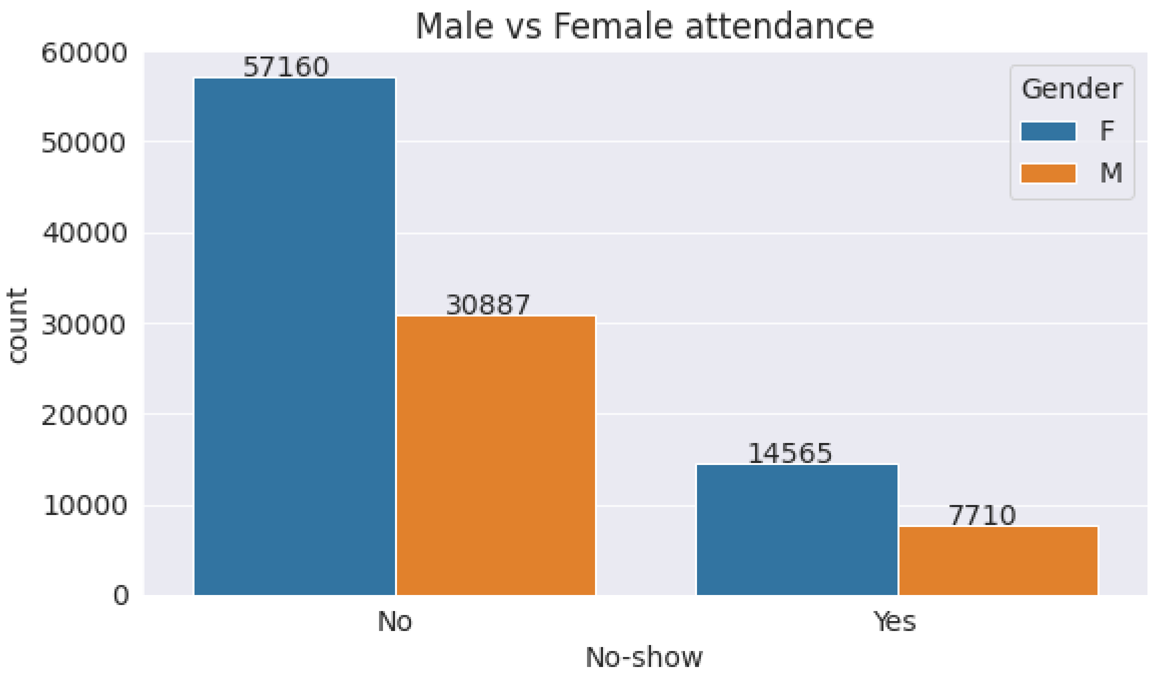
```
F    71725
M    38597
Name: Gender, dtype: int64
```

```
female_pat = appointment_df[appointment_df['Gender'] == 'F'].shape[0]
male_pat = appointment_df[appointment_df['Gender'] == 'M'].shape[0]
percent_female_pat = round(female_pat / appointment_df['Gender'].shape[0] * 100)
percent_female_pat
```

```
65
```

As we can see, 65% of Patients are Female.

```
ax = sns.countplot(x=appointment_df['No-show'], hue=appointment_df['Gender']);
for p in ax.patches:
   ax.annotate('{}'.format(p.get_height()), (p.get_x()+0.1, p.get_height()+0.01))
plt.title('Male vs Female attendance');
plt.show()
```



The above plot shows that the number of females is more than males for both show and no-show.

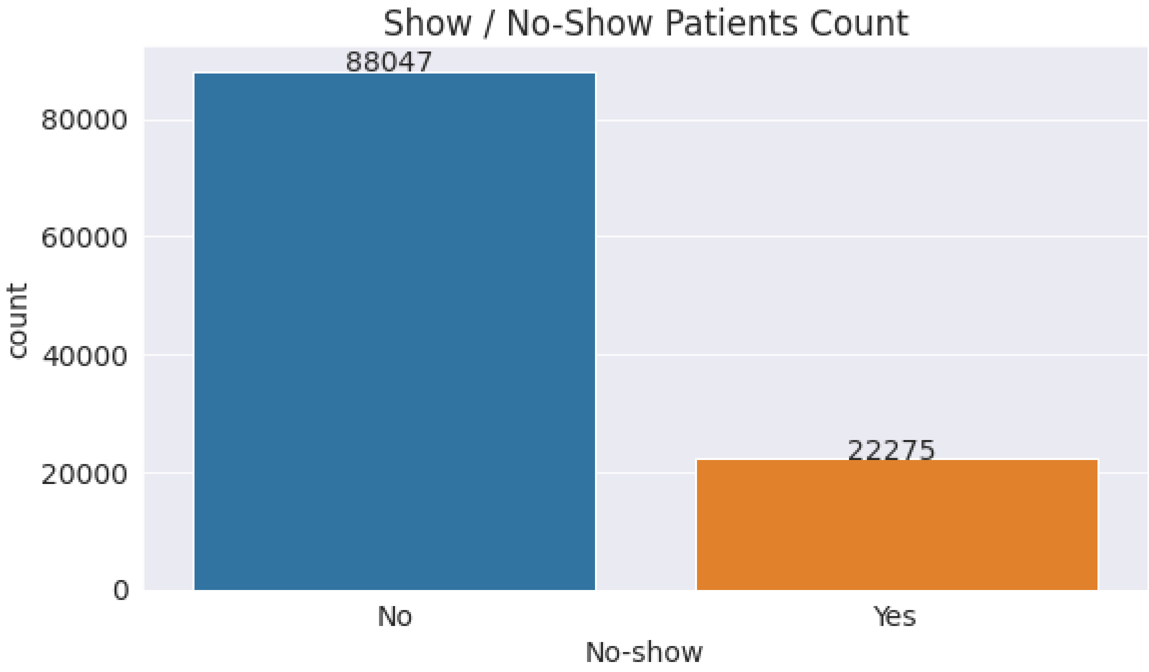
```
percent_female_noshow = round(appointment_df[(appointment_df['Gender'] == 'F') & (appoi
percent_male_noshow = round(appointment_df[(appointment_df['Gender'] == 'M') & (appoint
percent_female_noshow, percent_male_noshow
```

```
(20, 20)
```

As we can see from above calculation that rate of No-Show for both male and female patients is 20%.

Q2: How many percent of patients missed the appointment?

```
ax = sns.countplot(x = appointment_df['No-show'],data=appointment_df);
for p in ax.patches:
   ax.annotate('{}'.format(p.get_height()), (p.get_x()+0.3, p.get_height()+0.01))
plt.title('Show / No-Show Patients Count');
plt.show()
```



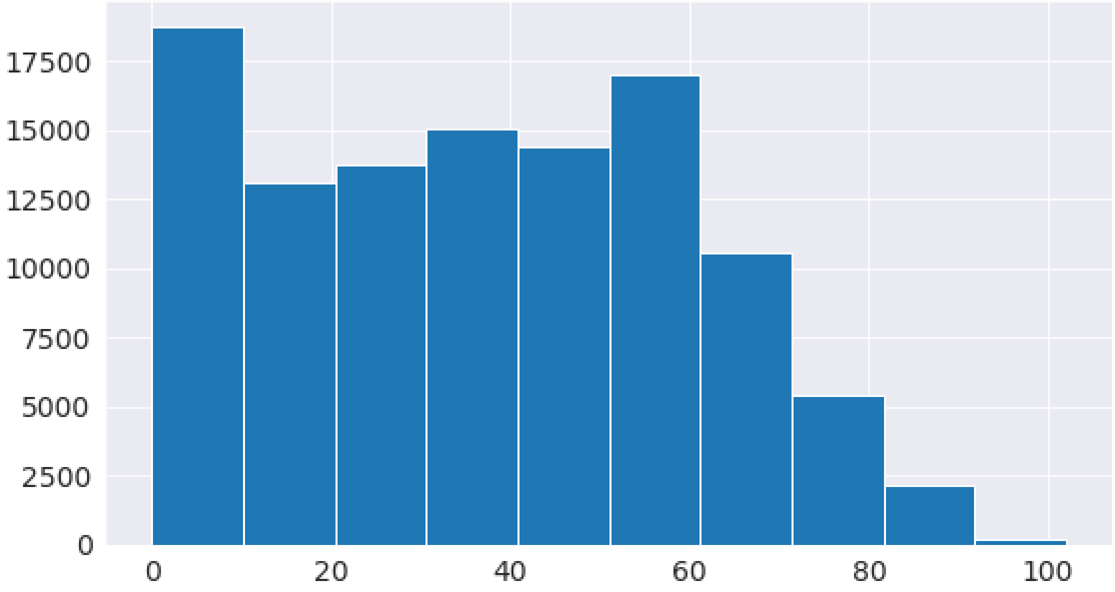
```
total_appointments = appointment_df.shape[0]
missed_appointments = appointment_df['No-show'].value_counts()['Yes']
missed_percent = int(round(missed_appointments/total_appointments*100))
missed_percent
```

```
20
```

The plot and calculation shows that 20% of the patients didn't show up.

Q3: Which Patients show up more: youth or old ?

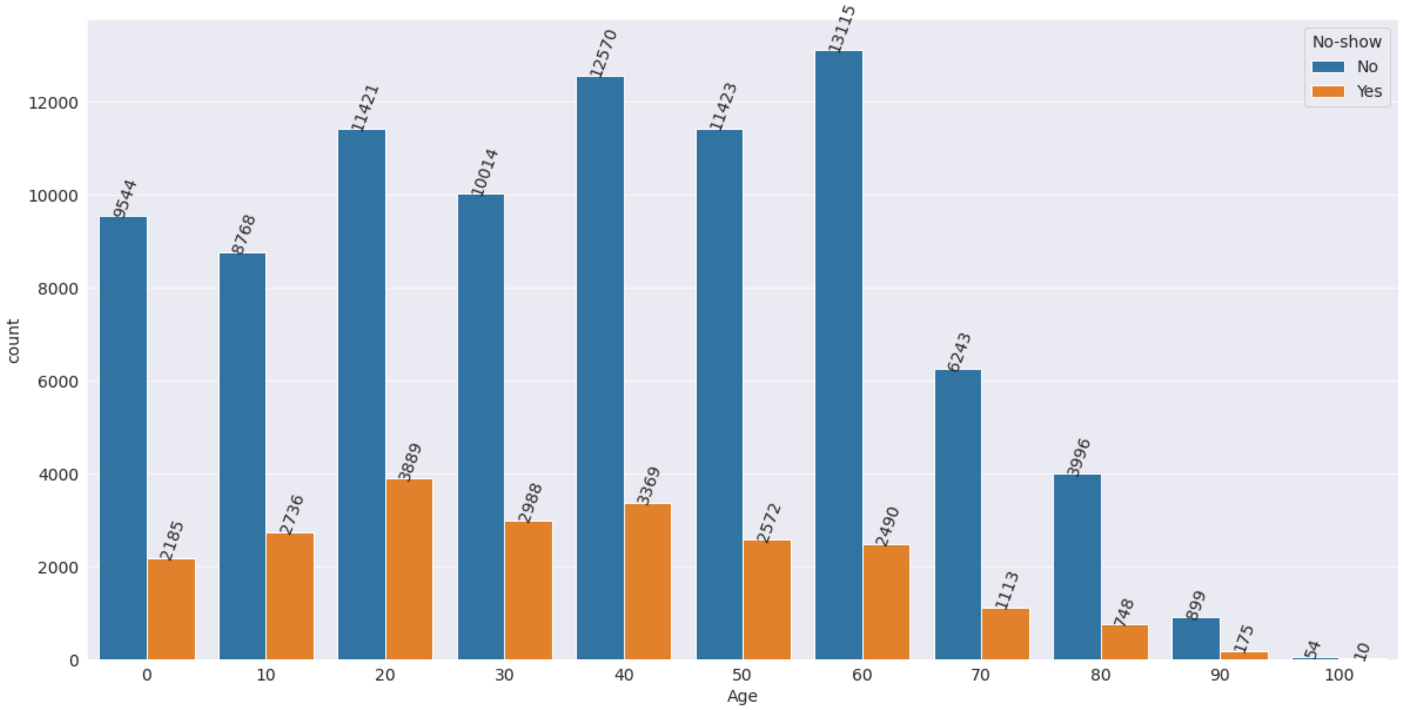
```
appointment_df['Age'].hist();
```



We can see that more patients showing up are Youth.

```
appointment_df['Age'] = [round(a,-1) for a in appointment_df['Age']]
```

```
plt.figure(figsize=(20,10))
ax = sns.countplot(x=appointment_df['Age'],hue=appointment_df['No-show']);
for p in ax.patches:
   ax.annotate('{}'.format(p.get_height()), (p.get_x()+0.1, p.get_height()+0.01),rotati
plt.show()
```

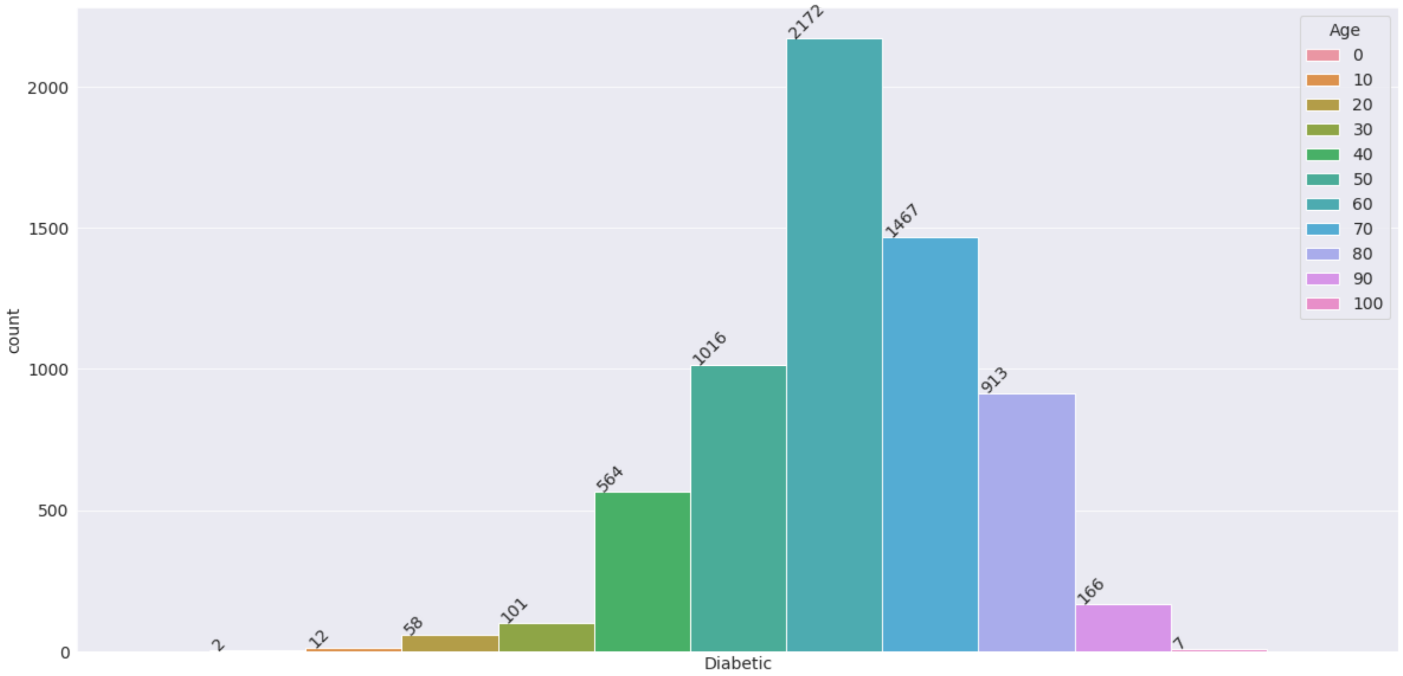


Above plot confirms that patients that showed up more are Youth.We have also calculated earlier that 59195 Patients are in the age range 0-40 Years.

Q4: What is the age distribution of patients that show up and are diabetic?

```
diabetic_show_up_df = appointment_df[(appointment_df['Diabetes'] == 1) & (appointment_c
```

```
plt.figure(figsize=(20,10))
ax = sns.countplot(x=diabetic_show_up_df['Diabetes'],hue=diabetic_show_up_df['Age']);
for p in ax.patches:
   ax.annotate('{}'.format(p.get_height()), (p.get_x(), p.get_height()+0.01),rotation=4
plt.xticks([1],[''])
plt.xlabel('Diabetic')
plt.show()
```



Q5: What percent of patients that recieved sms don't show up?

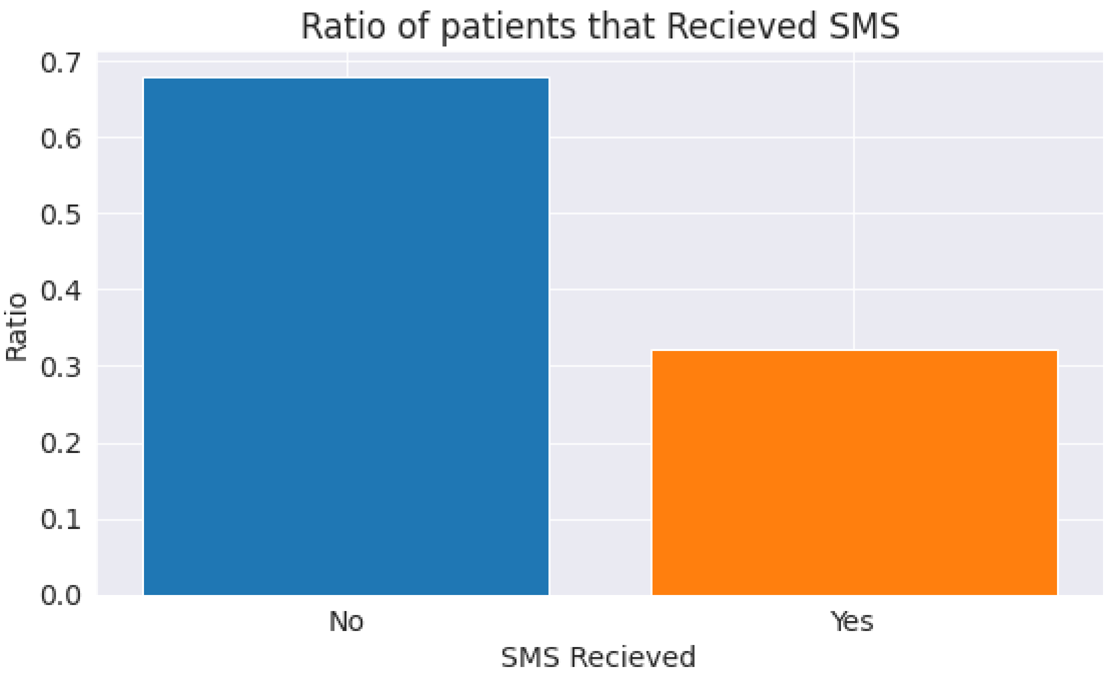
As we have seen in earlier calculation that 68% of patients didn't receive SMS.

```
no_sms_rec = appointment_df['SMS_received'].value_counts()[0] / appointment_df.shape[0]
sms_rec = appointment_df['SMS_received'].value_counts()[1] / appointment_df.shape[0]
```

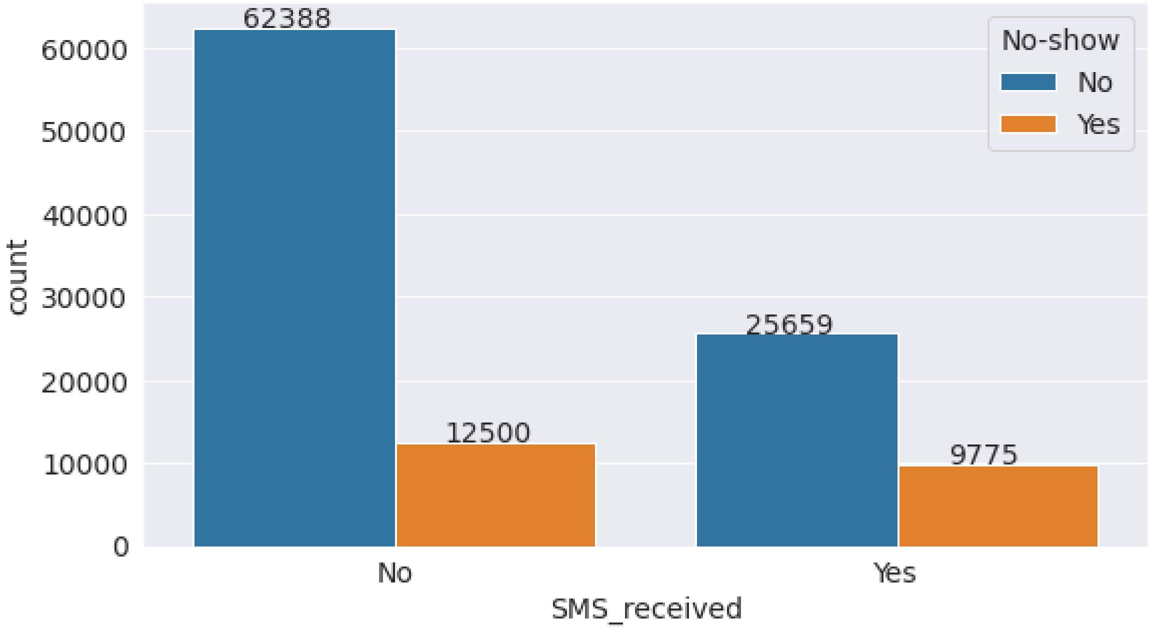
```
sms_rec,no_sms_rec
```

```
(0.3211870705752253, 0.6788129294247748)
```

```
sms_bar_plt = plt.bar(0, no_sms_rec);
no_sms_bar_plt = plt.bar(1, sms_rec);
plt.xticks([0, 1], ['No', 'Yes'])
plt.ylabel('Ratio')
plt.xlabel('SMS Recieved')
plt.title('Ratio of patients that Recieved SMS')
plt.show();
```



```
ax = sns.countplot(x=appointment_df['SMS_received'],hue=appointment_df['No-show'] );
for p in ax.patches:
   ax.annotate('{}'.format(p.get_height()), (p.get_x()+0.1, p.get_height()+0.01))
plt.xticks([0,1],['No','Yes']);
plt.show();
```



```
pat_sms_rec_showup = appointment_df[(appointment_df['SMS_received'] == 1) & (appointmen
pat_sms_rec_noshow = appointment_df[(appointment_df['SMS_received'] == 1) & (appointmen
pat_no_sms_rec_showup = appointment_df[(appointment_df['SMS_received'] == 0) & (appoint
pat_no_sms_rec_noshow = appointment_df[(appointment_df['SMS_received'] == 0) & (appoint
percent_no_show_sms_rec = round(pat_sms_rec_noshow / (pat_sms_rec_noshow + pat_sms_rec_
percent_no_show_no_sms_rec = round(pat_no_sms_rec_noshow / (pat_no_sms_rec_noshow + pat
percent_no_show_sms_rec,percent_no_show_no_sms_rec
```

```
(28, 17)
```

Inferences and Conclusion

Few Column names were spelled wrong,those were corrected.Columns which were not necessary were removed and relevant columns were added. Conclusions drawn from dataset are:-

1. 65% of Patients is Female.No-Show rate is 20% for both male and female patients.
2. 54% patients are in the age range 0-40 Years.
3. The average age of patients is 37 Years,this is in accordance with the above conclusion.
4. Only 3% Patients are alcoholic.
5. Around 2% Patients are handicap.
6. 20% Patients have hypertension.
7. Diabetic Patients in the age range of 40-70 Years Show up more for appointment.
8. 68% Patients didn't recieve SMS.
9. No-Show for patients that received SMS is 28% and is more than patients that didn't receive SMS(17%).
10. Overall No-Show Rate is 20%.

As there is no clear indication of any variable having more impact than others on No-Show Rate,This dataset can be further investigated.

```python
import jovian
```

```python
jovian.commit()
```

```
[jovian] Detected Colab notebook...
[jovian] Uploading colab notebook to Jovian...
Committed successfully! https://jovian.ai/ahmedatif655/medical-appointment-no-show
```

```
'https://jovian.ai/ahmedatif655/medical-appointment-no-show'
```