Investigate_a_Dataset

February 9, 2021

1 Project: Investigate a Dataset _No-show Appointments

1.1 Table of Contents

Introduction
Data Wrangling
Exploratory Data Analysis
Conclusions
Introduction

No-show Appointments

This dataset collects information from 100k medical appointments in Brazil with various characteristics about the patients.

Questions that are planned to explore :

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

2) can we predict the show up with the variables such as hypertension,diabetes,alcoholism etc ?

3) Whether the values of variable handicap helps to predict the show up?

4)Can the show up depends on the variable appointment day ?

Will try to explore with various variable to conclude, which factors play important role in show up for the patients.

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

Data Wrangling

```
In [2]: df=pd.read_csv('noshowappointments.csv')
        df.head(1)
```

```
Out[2]:       PatientId  AppointmentID Gender          ScheduledDay  \
        0  2.987250e+13        5642903      F  2016-04-29T18:38:08Z

                AppointmentDay  Age    Neighbourhood  Scholarship  Hipertension  \
        0  2016-04-29T00:00:00Z   62  JARDIM DA PENHA            0             1

           Diabetes  Alcoholism  Handcap  SMS_received No-show
        0         0           0        0             0      No
```

To find the total number of records in noshowappointments.csv file

```
In [3]: df.shape

Out[3]: (110527, 14)
```

There are totally 110527 rows and 14 columns.
To find null values and duplicates.

```
In [4]: df.isnull().any()

Out[4]: PatientId         False
        AppointmentID     False
        Gender            False
        ScheduledDay      False
        AppointmentDay    False
        Age               False
        Neighbourhood     False
        Scholarship       False
        Hipertension      False
        Diabetes          False
        Alcoholism        False
        Handcap           False
        SMS_received      False
        No-show           False
        dtype: bool

In [5]: #df.duplicated()
        duplicate = df[df.duplicated()]
        print("Duplicate Rows :")
        duplicate

Duplicate Rows :


Out[5]: Empty DataFrame
        Columns: [PatientId, AppointmentID, Gender, ScheduledDay, AppointmentDay, Age, Neighbour
        Index: []
```

The above results shows that there are no null values and duplicates in the records.
Seems that the column names can be make better in uniform readable format.

```
In [6]: df.rename(columns={'PatientId':'patient_id','AppointmentID':'appointment_id','Gender':'g

In [7]: list(df.columns)

Out[7]: ['patient_id',
         'appointment_id',
         'gender',
         'scheduled_day',
         'appointment_day',
         'age',
         'neighbour_hood',
         'scholar_ship',
         'hyper_tension',
         'diabetes',
         'alcoholism',
         'handicap',
         'sms_received',
         'no_show']

In [8]: df.head()

Out[8]:      patient_id  appointment_id gender         scheduled_day  \
        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

                appointment_day  age     neighbour_hood  scholar_ship  hyper_tension  \
        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  handicap  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 [9]: df['appointment_day'].unique()

Out[9]: array(['2016-04-29T00:00:00Z', '2016-05-03T00:00:00Z',
               '2016-05-10T00:00:00Z', '2016-05-17T00:00:00Z',
               '2016-05-24T00:00:00Z', '2016-05-31T00:00:00Z',
               '2016-05-02T00:00:00Z', '2016-05-30T00:00:00Z',
               '2016-05-16T00:00:00Z', '2016-05-04T00:00:00Z',
```

```
       '2016-05-19T00:00:00Z', '2016-05-12T00:00:00Z',
       '2016-05-06T00:00:00Z', '2016-05-20T00:00:00Z',
       '2016-05-05T00:00:00Z', '2016-05-13T00:00:00Z',
       '2016-05-09T00:00:00Z', '2016-05-25T00:00:00Z',
       '2016-05-11T00:00:00Z', '2016-05-18T00:00:00Z',
       '2016-05-14T00:00:00Z', '2016-06-02T00:00:00Z',
       '2016-06-03T00:00:00Z', '2016-06-06T00:00:00Z',
       '2016-06-07T00:00:00Z', '2016-06-01T00:00:00Z',
       '2016-06-08T00:00:00Z'], dtype=object)
```

Since the time in appointment_day column is 0 in all the rows, its better to remove the time from the values.

```
In [10]: df['appointment_day'] = pd.to_datetime(df['appointment_day']).dt.date

In [11]: df.head()
```

Out[11]:

	patient_id	appointment_id	gender	scheduled_day	appointment_day	age	neighbour_hood	scholar_ship	hyper_tension	diabetes	alcoholism	handicap	sms_received	no_show
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29	56	JARDIM DA PENHA	0	0	0	0	0	0	No
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29	62	MATA DA PRAIA	0	0	0	0	0	0	No
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29	8	PONTAL DE CAMBURI	0	0	0	0	0	0	No
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29	56	JARDIM DA PENHA	0	1	1	0	0	0	No

appointment_day column looks clean with clear data.

Its better to split the time and date in scheduled_day column to explore few things only with date.

```
In [12]: df['scheduled_date'] = pd.to_datetime(df['scheduled_day']).dt.date
         df['scheduled_time'] = pd.to_datetime(df['scheduled_day']).dt.time

In [13]: df.head(2)
```

Out[13]:

	patient_id	appointment_id	gender	scheduled_day	appointment_day	age	neighbour_hood	scholar_ship	hyper_tension	diabetes	alcoholism	handicap	sms_received	no_show
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29	62	JARDIM DA PENHA	0	1	0	0	0	0	No

```
1  5.589978e+14         5642503      M  2016-04-29T16:08:27Z      2016-04-29

   age   neighbour_hood  scholar_ship  hyper_tension  diabetes  alcoholism  \
0   62  JARDIM DA PENHA             0              1         0           0
1   56  JARDIM DA PENHA             0              0         0           0

   handicap  sms_received no_show scheduled_date scheduled_time
0         0             0      No     2016-04-29       18:38:08
1         0             0      No     2016-04-29       16:08:27
```

since scheduled_date and scheduled_time columns are newly populated, its better to delete the scheduled_day column

```
In [14]: del df['scheduled_day']

In [15]: df.head(2)

Out[15]:      patient_id  appointment_id gender appointment_day  age   neighbour_hood  \
         0  2.987250e+13         5642903      F      2016-04-29   62  JARDIM DA PENHA
         1  5.589978e+14         5642503      M      2016-04-29   56  JARDIM DA PENHA

            scholar_ship  hyper_tension  diabetes  alcoholism  handicap  sms_received  \
         0             0              1         0           0         0             0
         1             0              0         0           0         0             0

           no_show scheduled_date scheduled_time
         0      No     2016-04-29       18:38:08
         1      No     2016-04-29       16:08:27

In [16]: df.shape

Out[16]: (110527, 15)
```

After cleaning up the data, total number rows and columns are 110527 and 15 respectively in the dataframe.

Exploratory Data Analysis

To find the number of patients who show up and who not show up on their appointments.

```
In [17]: df.groupby(df['no_show']).count()['patient_id']

Out[17]: no_show
         No     88208
         Yes    22319
         Name: patient_id, dtype: int64
```

The number of patients who showed up on their appointments are 88208 and those who not show up are 22319 out of 110527 records.

```
In [18]: len(df['patient_id'].unique())
```

```
Out[18]: 62299

In [19]: len(df['appointment_id'].unique())

Out[19]: 110527
```

Since the unique number of patient_id is only 62299 out of 110527, it shows that each patient has more than one appointment record. And all the appointment records are unique.

To find the interval between scheduled_date and appointment_date: With the help of columns scheduled_date and appointment_day, the interval can be calculated.

```
In [20]: df['interval']=df['appointment_day']-df['scheduled_date']

In [21]: df['interval']=df['interval'].dt.days

In [22]: df.groupby(df['interval']).count()['patient_id'].reset_index()

Out[22]:
      interval  patient_id
0           -6           1
1           -1           4
2            0       38563
3            1        5213
4            2        6725
5            3        2737
6            4        5290
7            5        3277
8            6        4037
9            7        4906
10           8        2332
11           9        1605
12          10        1391
13          11         987
14          12        1115
15          13        1682
16          14        2913
17          15        1503
18          16        1151
19          17        1107
20          18        1021
21          19        1044
22          20        1187
23          21        1861
24          22        1173
25          23         822
26          24         622
27          25         637
28          26         731
29          27        1013
..         ...         ...
```

```
101        101           1
102        102           4
103        103           5
104        104           8
105        105           4
106        107           2
107        108           5
108        109           5
109        110           2
110        111           5
111        112           5
112        115           2
113        117           1
114        119           4
115        122           3
116        123           1
117        125           1
118        126           1
119        127           1
120        132           1
121        133          11
122        139           1
123        142           8
124        146           1
125        151           1
126        155          10
127        162          11
128        169           8
129        176          16
130        179          10

[131 rows x 2 columns]
```

```
In [23]: df['interval'].count()
```

```
Out[23]: 110527
```

```
In [24]: (df[df['interval']>=0]).count()['patient_id']
```

```
Out[24]: 110522
```

The values of interval include -6 and -1 which indicates the interval could not be in negative values.since the total number of records are only 5 out of 110527, no need to disturb those records.

To find the characteristics of patients show up to their appointments, we need the records of those patients seperately. Lets make one more dataframe which contains only the records of patients who show up.

```
In [25]: df.shape
```

```
Out[25]: (110527, 16)
```

```
In [26]: df_no=df[df['no_show']=='No']

In [27]: df_no.shape

Out[27]: (88208, 16)

In [28]: df_no.head()
```

Out[28]:

	patient_id	appointment_id	gender	appointment_day	age	\
0	2.987250e+13	5642903	F	2016-04-29	62	
1	5.589978e+14	5642503	M	2016-04-29	56	
2	4.262962e+12	5642549	F	2016-04-29	62	
3	8.679512e+11	5642828	F	2016-04-29	8	
4	8.841186e+12	5642494	F	2016-04-29	56	

	neighbour_hood	scholar_ship	hyper_tension	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	

	handicap	sms_received	no_show	scheduled_date	scheduled_time	interval
0	0	0	No	2016-04-29	18:38:08	0
1	0	0	No	2016-04-29	16:08:27	0
2	0	0	No	2016-04-29	16:19:04	0
3	0	0	No	2016-04-29	17:29:31	0
4	0	0	No	2016-04-29	16:07:23	0

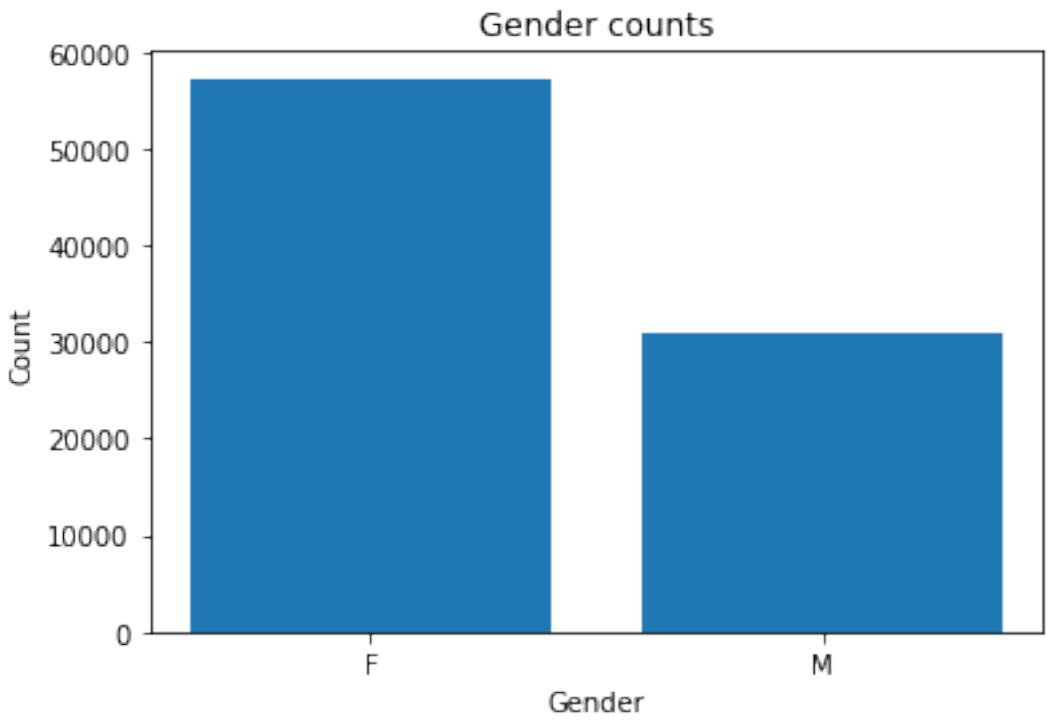
To find the male female counts

```
In [29]: df_gender=df_no.groupby('gender').count()[['patient_id']].reset_index()
         df_gender
```

Out[29]:

	gender	patient_id
0	F	57246
1	M	30962

```
In [30]: plt.bar(df_gender['gender'],df_gender['patient_id'])
         plt.xlabel('Gender')
         plt.ylabel('Count')
         plt.title('Gender counts')
         plt.show()
```



Female patients are showing up more compared to male.
To find records based on appointment_day

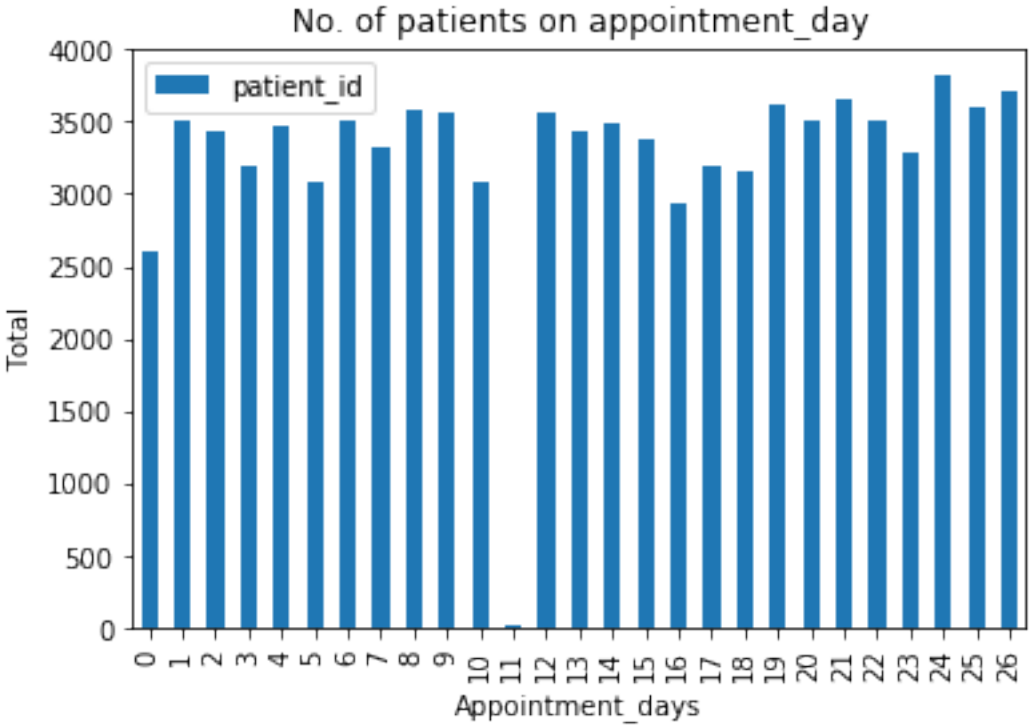
```
In [31]: df_appointment_day=df_no.groupby('appointment_day').count()[['patient_id']].reset_index
         df_appointment_day
```

Out[31]:

	appointment_day	patient_id
0	2016-04-29	2602
1	2016-05-02	3515
2	2016-05-03	3425
3	2016-05-04	3195
4	2016-05-05	3466
5	2016-05-06	3084
6	2016-05-09	3501
7	2016-05-10	3316
8	2016-05-11	3589
9	2016-05-12	3557
10	2016-05-13	3082
11	2016-05-14	30
12	2016-05-16	3564
13	2016-05-17	3437
14	2016-05-18	3483
15	2016-05-19	3378
16	2016-05-20	2929
17	2016-05-24	3198

```
18      2016-05-25        3150
19      2016-05-30        3626
20      2016-05-31        3512
21      2016-06-01        3652
22      2016-06-02        3508
23      2016-06-03        3285
24      2016-06-06        3819
25      2016-06-07        3600
26      2016-06-08        3705
```

```
In [32]: df_appointment_day.plot.bar(stacked=True)
         plt.xlabel('Appointment_days')
         plt.ylabel('Total')
         plt.title('No. of patients on appointment_day')
         plt.show()
```



couldnt predict anything with is data excluding that 11th record dated 2016-05-14 has very minimum count of 30.

Will check the records based on age.

```
In [33]: df_age=df_no.groupby('age').count()['patient_id'].reset_index()
         df_age
```

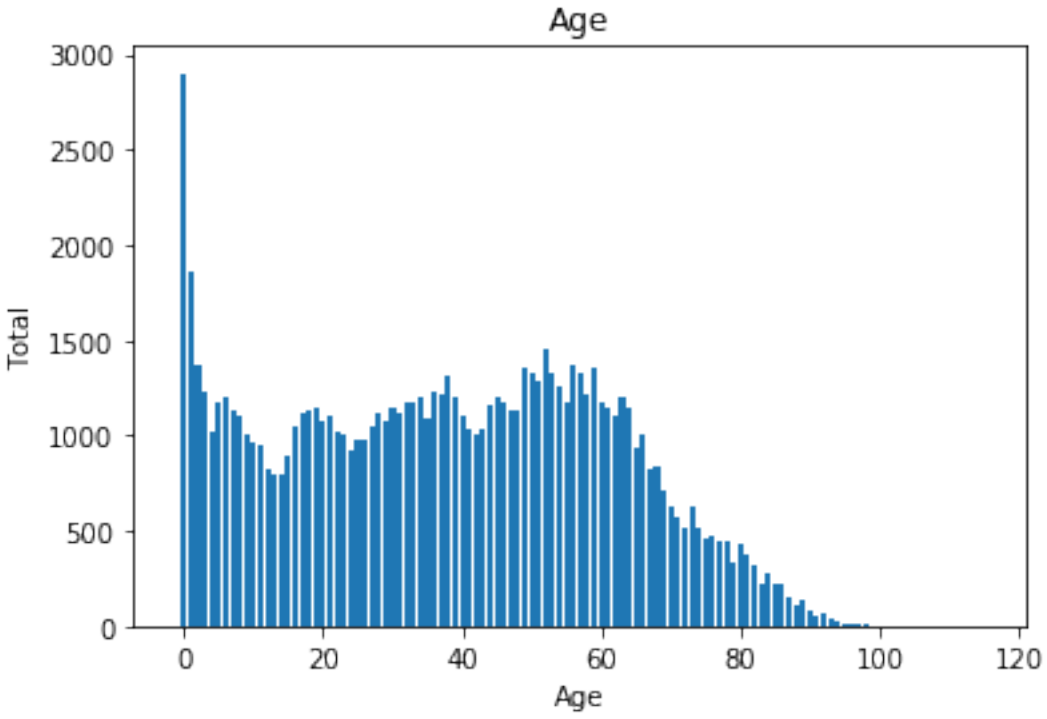
```
Out[33]:      age  patient_id
         0     -1           1
```

1	0	2900
2	1	1858
3	2	1366
4	3	1236
5	4	1017
6	5	1169
7	6	1205
8	7	1126
9	8	1106
10	9	1008
11	10	970
12	11	948
13	12	820
14	13	800
15	14	802
16	15	889
17	16	1049
18	17	1113
19	18	1137
20	19	1151
21	20	1082
22	21	1097
23	22	1025
24	23	1006
25	24	921
26	25	980
27	26	971
28	27	1048
29	28	1116
..	...	...
74	73	629
75	74	513
76	75	463
77	76	480
78	77	448
79	78	452
80	79	329
81	80	430
82	81	371
83	82	326
84	83	219
85	84	276
86	85	226
87	86	218
88	87	157
89	88	114
90	89	144
91	90	86

```
92     91          53
93     92          66
94     93          43
95     94          27
96     95          18
97     96          16
98     97           9
99     98           5
100    99           1
101   100           4
102   102           2
103   115           2

[104 rows x 2 columns]
```

In [34]:
```
plt.bar(df_age['age'],df_age['patient_id'])
plt.xlabel('Age')
plt.ylabel('Total')
plt.title('Age')
plt.show()
```



In [35]:
```
df_age['age'].mean()
```

Out[35]:
```
50.634615384615387
```

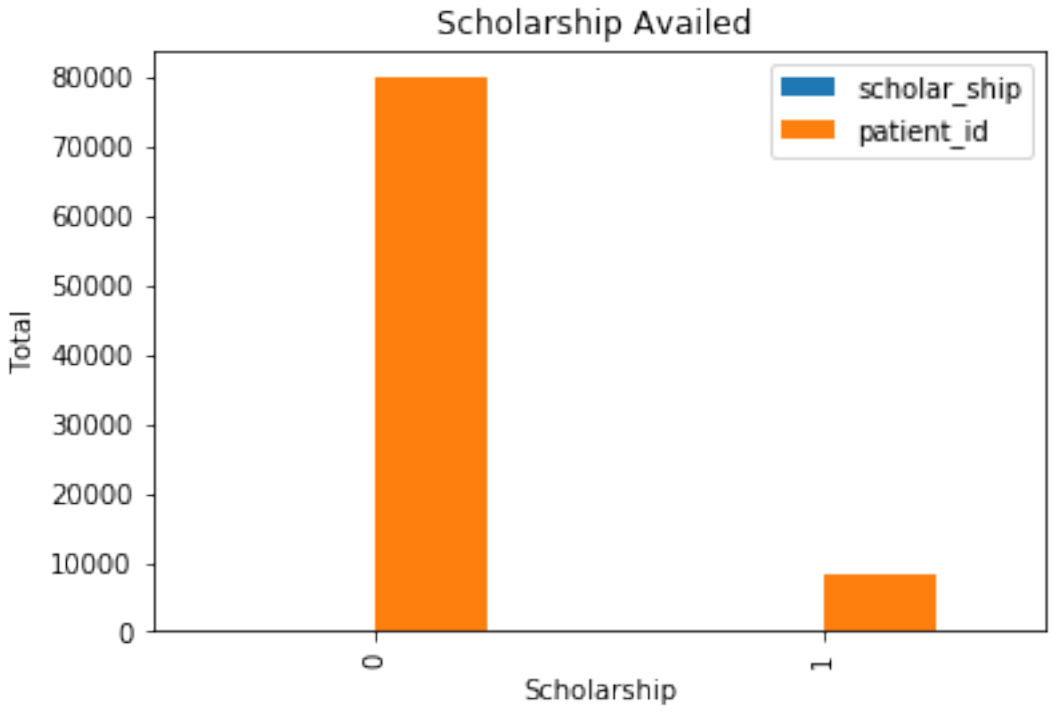
The age 0 contains 2900 records. This may be records of either children below 1 year or the age value might not be recorded so default may be of 0. So couldnt predict anything with this age variable.

Will evaluate the scholarship variable:

```
In [36]: df_scholar_ship=df_no.groupby('scholar_ship').count()['patient_id'].reset_index()
         df_scholar_ship

Out[36]:    scholar_ship  patient_id
         0             0       79925
         1             1        8283

In [37]: df_scholar_ship.plot.bar()
         plt.xlabel('Scholarship')
         plt.ylabel('Total')
         plt.title('Scholarship Availed ')
         plt.show()
```



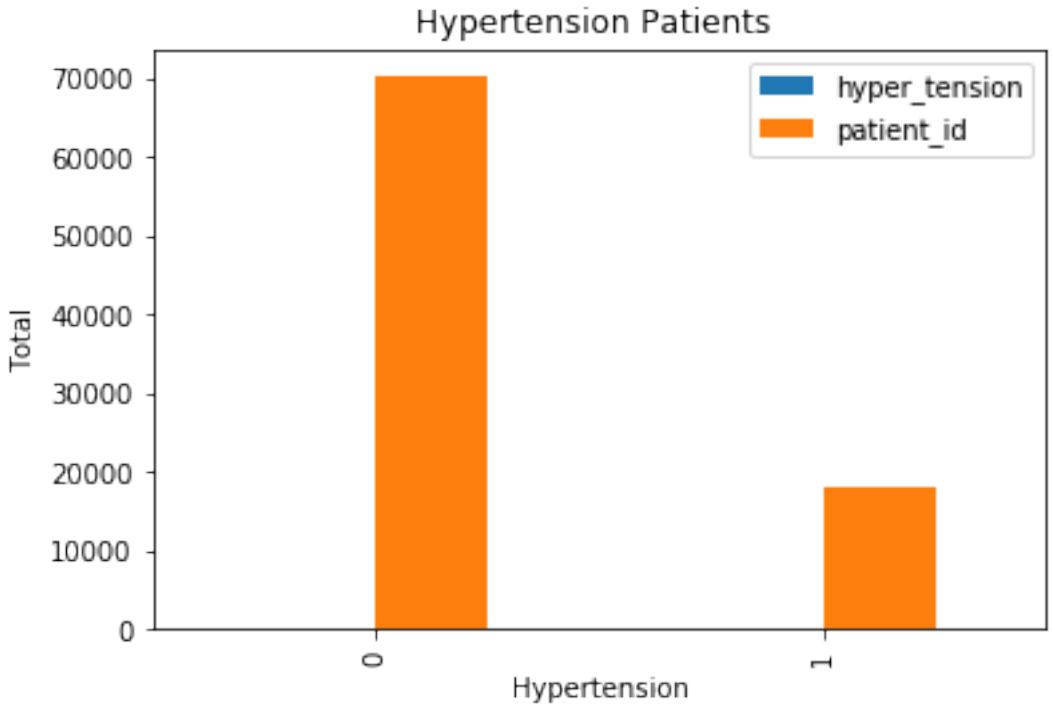
The majority of the patients who show up on their appointments were not enrolled in the scholarship of Brazil.

Lets get into hypertension variable

```
In [38]: df_hyper_tension=df_no.groupby('hyper_tension').count()['patient_id'].reset_index()
         df_hyper_tension
```

```
Out[38]:    hyper_tension  patient_id
         0              0       70179
         1              1       18029
```

```
In [39]: df_hyper_tension.plot.bar()
         plt.xlabel('Hypertension')
         plt.ylabel('Total')
         plt.title('Hypertension Patients ')
         plt.show()
```

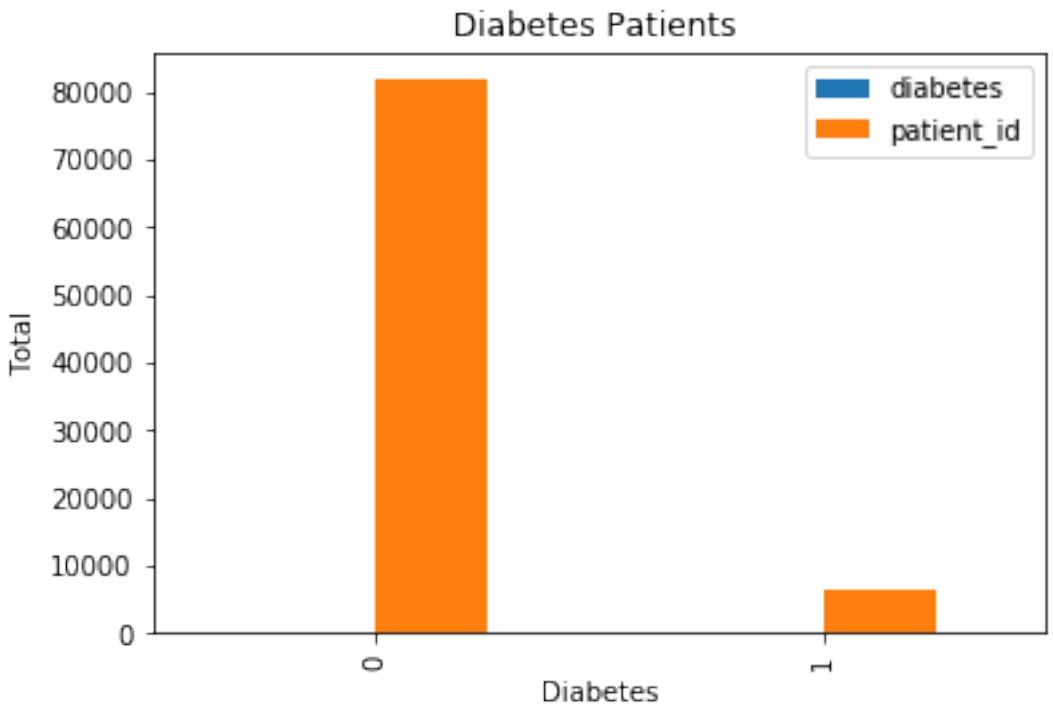


The majority of the patients who show up on their appointments does not have hypertension. Will check the records of diabetes patients.

```
In [40]: df_diabetes=df_no.groupby('diabetes').count()['patient_id'].reset_index()
         df_diabetes
```

```
Out[40]:    diabetes  patient_id
         0         0       81695
         1         1        6513
```

```
In [41]: df_diabetes.plot.bar()
         plt.xlabel('Diabetes')
         plt.ylabel('Total')
         plt.title('Diabetes Patients ')
         plt.show()
```

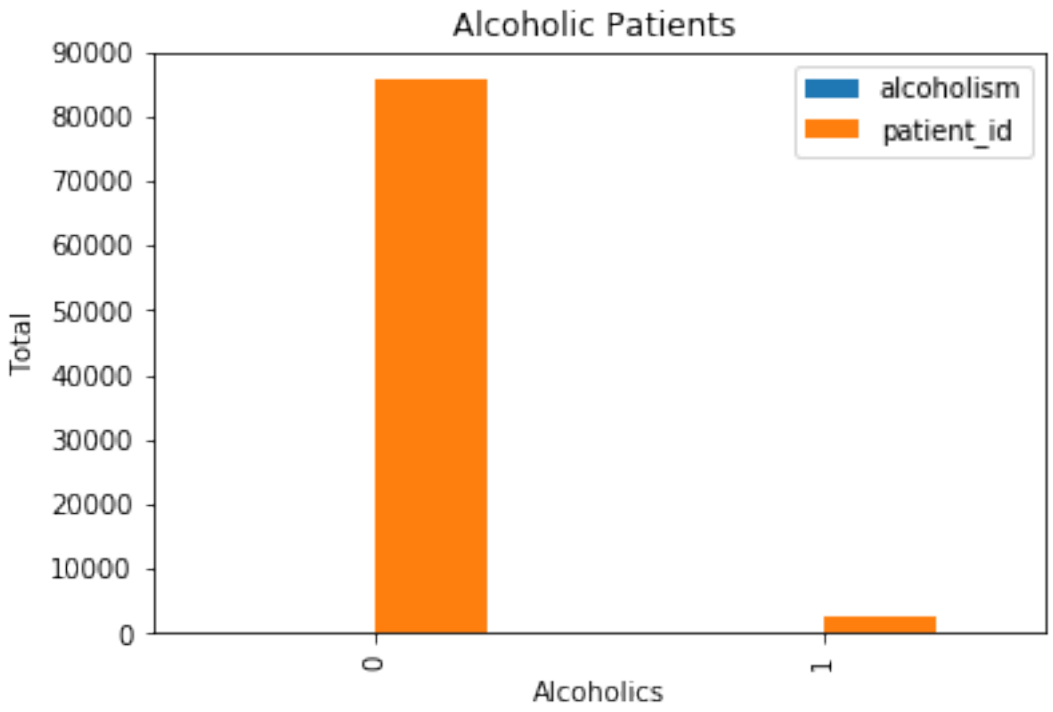


The patients showed up on their appointments are almost non diabetics.
Lets get into the recods of alcoholic patients.

```
In [42]: df_alcohol=df_no.groupby('alcoholism').count()['patient_id'].reset_index()
         df_alcohol

Out[42]:    alcoholism  patient_id
         0           0       85525
         1           1        2683

In [43]: df_alcohol.plot.bar()
         plt.xlabel('Alcoholics')
         plt.ylabel('Total')
         plt.title('Alcoholic Patients ')
         plt.show()
```

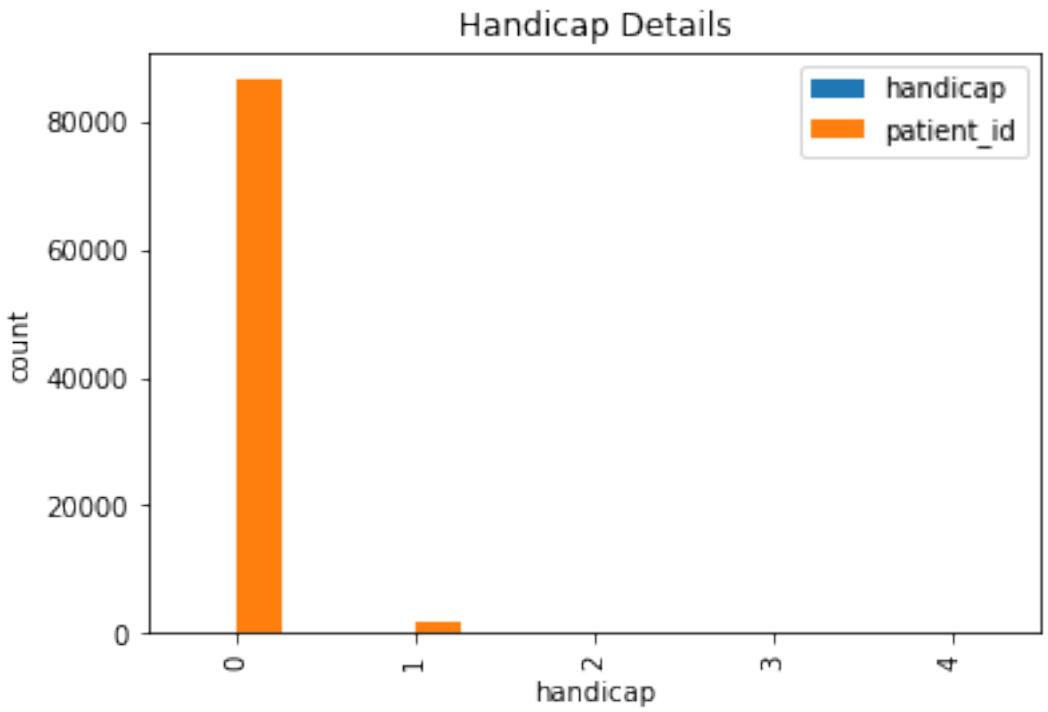


Alcoholic patients who show up are very minimum and most of the patients are non-alcoholic. To analyse from the handicap patients records.

```
In [44]: df_handicap=df_no.groupby('handicap').count()['patient_id'].reset_index()
         df_handicap
```

```
Out[44]:    handicap  patient_id
         0         0       86374
         1         1        1676
         2         2         146
         3         3          10
         4         4           2
```

```
In [45]: df_handicap.plot.bar()
         plt.xlabel('handicap')
         plt.ylabel('count')
         plt.title('Handicap Details')
         plt.show()
```

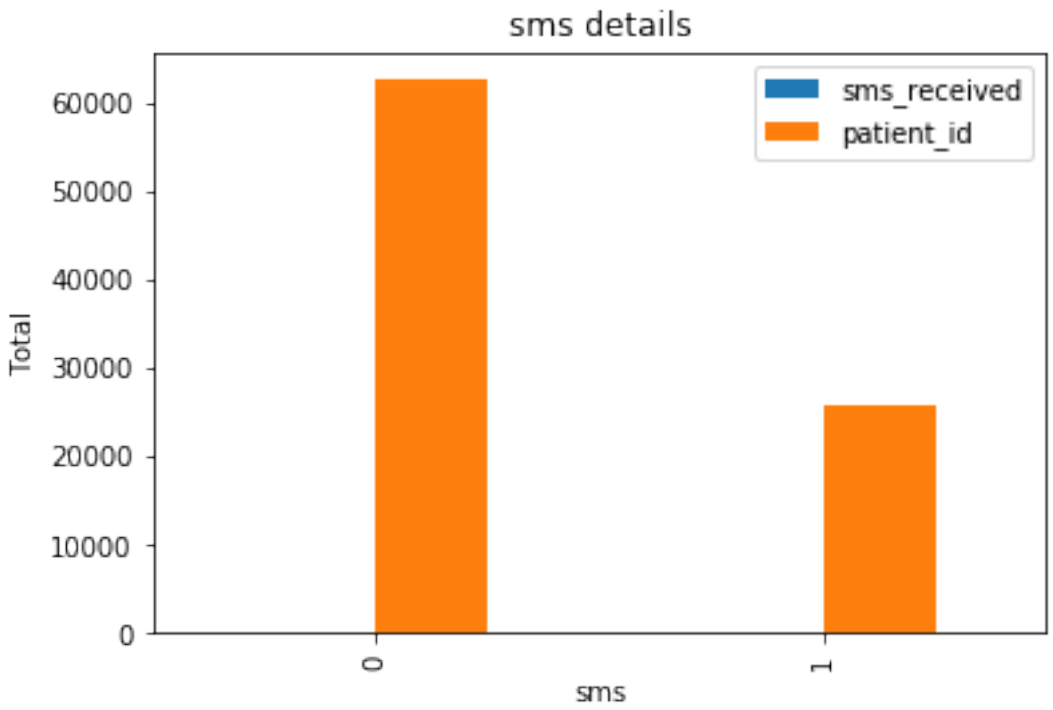


Among 88k plus patients 86374 patients are healthy without handicaps.
Will compute something based on the details of sms received by the patients.

```
In [46]: df_sms_received=df_no.groupby('sms_received').count()['patient_id'].reset_index()
         df_sms_received

Out[46]:    sms_received  patient_id
         0             0       62510
         1             1       25698

In [47]: df_sms_received.plot.bar()
         plt.xlabel('sms')
         plt.ylabel('Total')
         plt.title('sms details')
         plt.show()
```



Most of the patients who show up did not receive sms for their appointments. so this sms does not play any role here.

Lets look into the interval of days between scheduled and appointment days.

```
In [48]: df_interval=df_no.groupby('interval').count()['patient_id'].reset_index()
         df_interval
```

Out[48]:

	interval	patient_id
0	0	36771
1	1	4100
2	2	5123
3	3	2093
4	4	4059
5	5	2405
6	6	3036
7	7	3597
8	8	1662
9	9	1165
10	10	951
11	11	675
12	12	762
13	13	1146
14	14	2000
15	15	1001
16	16	800

```
17          17          757
18          18          709
19          19          681
20          20          779
21          21         1286
22          22          769
23          23          546
24          24          387
25          25          381
26          26          468
27          27          693
28          28         1203
29          29          733
..         ...          ...
94          94            2
95          95            4
96          96            3
97          97            2
98          98            4
99         101            1
100        102            3
101        103            2
102        104            2
103        105            4
104        107            2
105        108            5
106        109            5
107        110            1
108        111            4
109        112            5
110        115            2
111        117            1
112        119            4
113        122            3
114        123            1
115        125            1
116        127            1
117        133            7
118        142            5
119        155            4
120        162            9
121        169            7
122        176           10
123        179            8

[124 rows x 2 columns]
```

```
In [49]: df_interval_lt08=df_interval[df_interval['interval']<=8]
```

```
In [49]: df_interval_lt08

Out[49]:    interval  patient_id
         0         0       36771
         1         1        4100
         2         2        5123
         3         3        2093
         4         4        4059
         5         5        2405
         6         6        3036
         7         7        3597
         8         8        1662

In [50]: df_interval_lt08['patient_id'].sum()

Out[50]: 62846

In [51]: df_interval_lt15=df_interval[df_interval['interval']<=15]
         df_interval_lt15

Out[51]:     interval  patient_id
         0          0       36771
         1          1        4100
         2          2        5123
         3          3        2093
         4          4        4059
         5          5        2405
         6          6        3036
         7          7        3597
         8          8        1662
         9          9        1165
         10        10         951
         11        11         675
         12        12         762
         13        13        1146
         14        14        2000
         15        15        1001

In [54]: df_interval_lt15['patient_id'].sum()

Out[54]: 70546
```

Among 88k patients, around 62k were showed up with the interval of 8 days, which shows that the patients who has scheduled their appointments within a week have more chances to show up for the appointments.

Lets do some exploration with neighbourhood datas.

```
In [55]: df_neighbourhood=df_no.groupby('neighbour_hood').count()['patient_id'].sort_values(asce
         df_neighbourhood
```

Out[55]:

	neighbour_hood	patient_id
0	JARDIM CAMBURI	6252
1	MARIA ORTIZ	4586
2	RESISTÊNCIA	3525
3	JARDIM DA PENHA	3246
4	SANTA MARTHA	2635
5	CENTRO	2631
6	ITARARÉ	2591
7	TABUAZEIRO	2559
8	SANTO ANTÔNIO	2262
9	BONFIM	2223
10	JESUS DE NAZARETH	2157
11	SANTO ANDRÉ	2063
12	JABOUR	2058
13	CARATOÍRA	1974
14	SÃO PEDRO	1933
15	NOVA PALESTINA	1862
16	DA PENHA	1788
17	ANDORINHAS	1741
18	ROMÃO	1741
19	ILHA DO PRÍNCIPE	1734
20	GURIGICA	1562
21	SÃO JOSÉ	1549
22	FORTE SÃO JOÃO	1543
23	ILHA DE SANTA MARIA	1524
24	BELA VISTA	1523
25	MARUÍPE	1478
26	SÃO CRISTÓVÃO	1473
27	REDENÇÃO	1278
28	JOANA D'ARC	1169
29	SÃO BENEDITO	1152
..	...	...
50	DO CABRAL	472
51	SANTOS REIS	435
52	ESTRELINHA	432
53	SOLON BORGES	400
54	SANTA CLARA	372
55	PIEDADE	364
56	SANTA LÚCIA	352
57	SANTA LUÍZA	351
58	BARRO VERMELHO	332
59	SANTA CECÍLIA	325
60	DO MOSCOSO	321
61	MÁRIO CYPRESTE	317
62	DE LOURDES	258
63	BOA VISTA	254
64	COMDUSA	254
65	ANTÔNIO HONÓRIO	221

```
66  ARIOVALDO FAVALESSA         220
67            FRADINHOS         210
68       ENSEADA DO SUÁ         183
69         SANTA HELENA         141
70                HORTO         133
71        UNIVERSITÁRIO         120
72     SEGURANÇA DO LAR         117
73              NAZARETH         106
74    MORADA DE CAMBURI          80
75    PONTAL DE CAMBURI          57
76           ILHA DO BOI          32
77        ILHA DO FRADE           8
78            AEROPORTO           7
79    PARQUE INDUSTRIAL           1

[80 rows x 2 columns]
```

```
In [72]: df_top10=df_neighbourhood[:10]
         df_top10
```

Out[72]:

	neighbour_hood	patient_id
0	JARDIM CAMBURI	6252
1	MARIA ORTIZ	4586
2	RESISTÊNCIA	3525
3	JARDIM DA PENHA	3246
4	SANTA MARTHA	2635
5	CENTRO	2631
6	ITARARÉ	2591
7	TABUAZEIRO	2559
8	SANTO ANTÔNIO	2262
9	BONFIM	2223

Among 80 places listed in neighbourhood, i have shortlisted top 10 places depends on the count of patients.Those ten places have a total count of more than 2k and the most number of patients showed up were from JARDIM CAMBURI(count as 6252).

While comparing top 4 cities which has count more than 3k, (JARDIM CAMBURI,MARIA ORTIZ,RESISTÊNCIA,JARDIM DA PENHA) we came to know through Brazil map, JARDIM CAMBURI,MARIA ORTIZ and JARDIM DA PENHA belongs to State of Espírito Santo, Brazil and the third place city of RESISTÊNCIA is in Argentina, it is twinned with São Vicente, Brazil. From these places the patients were showed up to their appointments are higher when compared to other places.

To find out the days of the appointments for the patients.

```
In [78]: df_no['appointment_day']=pd.to_datetime(df_no['appointment_day'])
         df_no['appointment_day_week']=df_no['appointment_day'].dt.day_name()
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
  """Entry point for launching an IPython kernel.
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#
```

```
In [79]: df_no.head(2)
```

Out[79]:

	patient_id	appointment_id	gender	appointment_day	age	neighbour_hood	\
0	2.987250e+13	5642903	F	2016-04-29	62	JARDIM DA PENHA	
1	5.589978e+14	5642503	M	2016-04-29	56	JARDIM DA PENHA	

	scholar_ship	hyper_tension	diabetes	alcoholism	handicap	sms_received	\
0	0	1	0	0	0	0	
1	0	0	0	0	0	0	

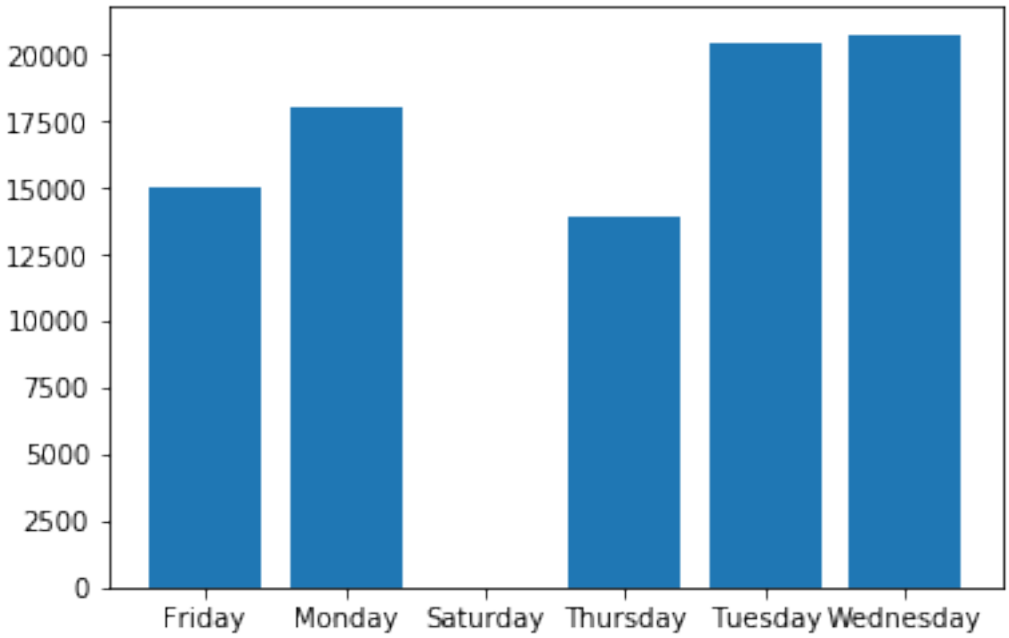
	no_show	scheduled_date	scheduled_time	interval	appointment_day_week
0	No	2016-04-29	18:38:08	0	Friday
1	No	2016-04-29	16:08:27	0	Friday

```
In [80]: df_days=df_no.groupby('appointment_day_week').count()['patient_id'].reset_index()
         df_days
```

Out[80]:

	appointment_day_week	patient_id
0	Friday	14982
1	Monday	18025
2	Saturday	30
3	Thursday	13909
4	Tuesday	20488
5	Wednesday	20774

```
In [108]: plt.bar(df_days['appointment_day_week'],df_days['patient_id'])
          plt.xlabel='Days'
          plt.ylabel='Total'
          plt.title='count per days'
          plt.show()
```



As pe the above figure, the number of patients are high on Tuesday and Wednesday. Also on Monday. No records were given for Sunday, and Saturday has the minimum count of 30.

```
In [95]: df_days_interval = df_no.groupby(["appointment_day_week","interval"]).count()['patient_
```

```
In [96]: df_days_interval.head()
```

```
Out[96]:   appointment_day_week  interval  patient_id
         0              Friday         0        6140
         1              Friday         1         925
         2              Friday         2        1525
         3              Friday         3         584
         4              Friday         4         450
```

```
In [97]: df_days_interval = df_days_interval[df_days_interval['interval']<=8]
```

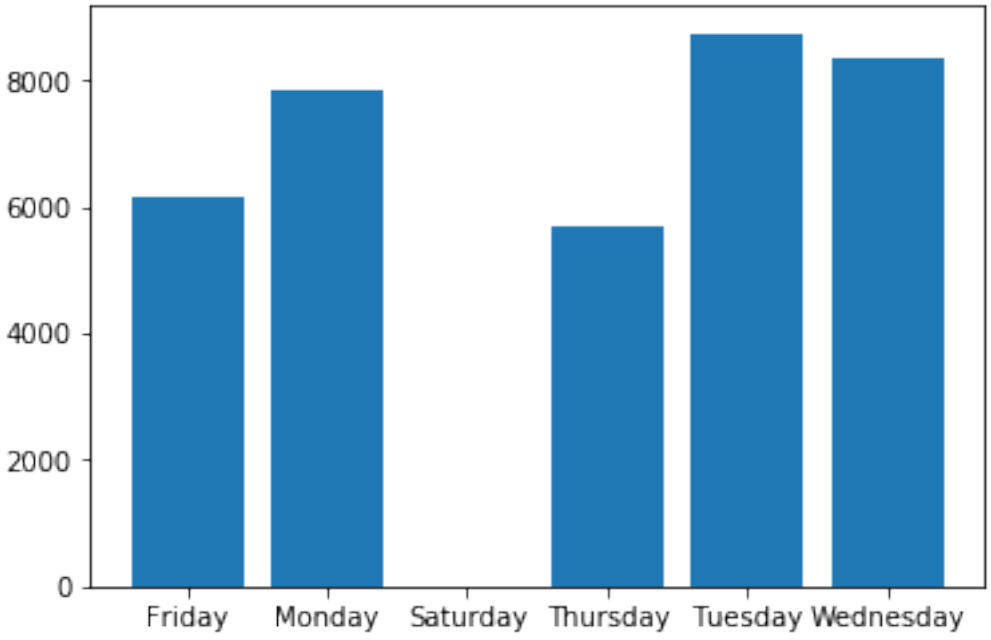
```
In [98]: df_days_interval.groupby('appointment_day_week').agg({'patient_id':'sum'}).reset_index(
```

```
Out[98]:   appointment_day_week  patient_id
         0              Friday       10300
         1              Monday       12818
         2            Saturday          25
         3            Thursday        9869
         4             Tuesday       14897
         5           Wednesday       14937
```

```
In [109]: #plt.bar(df_days_interval['appointment_day_week'],df_days_interval['patient_id'])
          #plt.xlabel("Days of Week")
```

```
#plt.ylabel('Total')
#plt.title('No. of patients per week')
#plt.show()

plt.bar(df_days_interval['appointment_day_week'],df_days_interval['patient_id'])
plt.xlabel='Days of Week'
plt.ylabel='Total'
plt.title='No. of patients per week'
plt.show()
```




```
In [112]: df_days_interval.sum()['patient_id']
```

```
Out[112]: 62846
```

So, out of 88k patients, 62846 were showed up within a week time of scheduled appointment and mostly on Tuesday,Wednesday and Monday.

Conclusions

With the above analysis, so far we knew that, 1) patients who has scheduled appointments within a week are showed up higher 2) Patients whose appointments on Tuesday, Wednesday are higher, even on Mondays too. 3) Patients from the places of JARDIM CAMBURI,MARIA ORTIZ,RESISTÊNCIA,JARDIM DA PENHA were showing up more than compared to others.

In order to predict if a patient will show up for the scheduled appointment, the important factors to be considered are Interval between scheduled day and appointment day, Day of the week and neighbourhood.

```
In [114]: from subprocess import call
          call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])
```

```
Out[114]: 0
```

```
In [ ]:
```