Investigate_a_Dataset

October 9, 2022

1 Project: Investigate a Dataset - [No-show Appointments]

1.1 Table of Contents

Introduction

1.1.1 Dataset Description

The dataset I investigated for this project is a set of medical records for hospital appointments in Brazil. The data shows some features about the patients and whether or not they showed up to their appointments. The goal of the analysis is to find patterns that explain why patients show up or don't show up to appointments.

'PatientId is the unique number given to a patient to Identify that patient
'AppointmentId' is the number assigned to a patient to identify each appointment
'Gender' specifies if the patient is male or female
'Age' tells how old the patient is
'ScheduledDay' informs us of the patient's scheduled appointment time and date
'Neighborhood' refers to the hospital's location
'Scholarship' specifies whether or not the patient is registered in Brazil's Bolsa Famlia welfare program
'Hipertension' indicates that the patient is hypertensive
'Diabetes' indicates that the patient is diabetic
'Alcoholism' indicates that the patient is addicted to alcohol
'Handcap' indicates that the patient has some disabilities
'Sms_recieved' indicates that a message or no message was sent to the patient as a reminder about the appointment
'No show' indicates if a patient showed up for appointment, it shows 'Yes' if a patient did not show up and 'No' if the patient showed up

1.1.2 Question(s) for Analysis

```
<li> What percentage of patients showed up for appointment? </li>
<li> Is there one feature that absolutely influence a patient showing up to an appointment </li>
```

 What gender show up more for appointment?
 Does Recieving SMS Reminder Influence a Patient to Show Up for Appointment?
 What neighbourhood hospital receive the most appointment?

In [2]:
```
# Import important modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

Data Wrangling

In [3]:
```
#load the no_show appointment dataset csv file into a dataframe named Patient_Apt

Patient_Apt = pd.read_csv ('noshowappointments-kagglev2-may-2016.csv')
```

In [4]:
```
#view the dataframe

Patient_Apt.head()
```

Out[4]:
```
      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 [5]:
```
Patient_Apt.shape
```

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

The code above shows that the dataframe consists of **110527 rows** and **14 columns**

In [6]:
```
Patient_Apt.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
```

The code above shows the data type for all columns in the dataframe.

I observed that the patient_Id column has a datatype of float which I think the appropraite data type should be an integer.

I also observed that the scheduledDay and AppointmentDay has data type as object, which I think the appropraite data type for date is date-time

Thirdly, I observed some column names have some typo errors, columns such as 'handcap' should be 'handicap', 'hipertension' should be 'hypertension', ScheduledDay and AppointmentDay should have an underscore seperating both words like, 'schedule_day'

```
In [7]: Patient_Apt.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
```

The above code gives a statistical description of the dataframe.

It can be seen that the average age of a patient is 37, minimum age is -1 which I conclude to be an error and needs to be dropped as no person can be of age -1.

The maximum age is 115.

25% of patients are around 18 years old and above, 50% of people are around the age of 37 years old, 75% of patients are around 55years old.

The minimum of scholarship, diabetes, alcoholism and sms_recieved is 0 and maximum is 1, while the minimum of handcap is 0 and the maximum is 4. According to this source on kaggle https://www.kaggle.com/datasets/joniarroba/noshowappointments/discussion/32174?page=2 a maximum of 4 means the patient has visual, physical and other disability condititions

In [8]:
```
Patient_Apt.isnull().sum()
```

Out[8]:
```
PatientId         0
AppointmentID     0
Gender            0
ScheduledDay      0
AppointmentDay    0
Age               0
Neighbourhood     0
Scholarship       0
Hipertension      0
Diabetes          0
Alcoholism        0
Handcap           0
SMS_received      0
No-show           0
dtype: int64
```

There are no null values in the dataframe

In [9]:
```
repeating_rows = Patient_Apt.duplicated(). sum()
print (repeating_rows)
```

```
0
```

There are no duplicated rows in the dataframe

1.1.3 Data Cleaning

From my observations in the previous section, I discovered some column names with typo errors and column names with no underscore to seperate both words for better interpretability.The first thing I am going to do in this section is to rename columns by correcting typographical errors in the column names. To do this, I created a user-defined function that takes in the old column names as input and output the renamed new columns.

In [10]:
```
#user-defined function to rename column

def rename_column(df, old_columns, new_columns):
    if len(old_columns) !=len(new_columns):
        return print('Error!!! Number of old_columns must be equal to number of new_col
    else:
        for i in range(len(old_columns)):
            df.rename(columns = { old_columns[i] : new_columns[i] }, inplace = True)
        return df
```

In [11]:
```
#renaming columns if user-defined function is called
old_columns = ['ScheduledDay', 'PatientId', 'AppointmentDay', 'AppointmentID', 'Handcap
new_columns = ['Scheduled_Day', 'Patient_Id', 'Appointment_Day', 'Appointment_ID', 'Han
rename_column(Patient_Apt, old_columns, new_columns)
```

Out[11]:
```
       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
5    9.598513e+13         5626772      F  2016-04-27T08:36:51Z
6    7.336882e+14         5630279      F  2016-04-27T15:05:12Z
7    3.449833e+12         5630575      F  2016-04-27T15:39:58Z
8    5.639473e+13         5638447      F  2016-04-29T08:02:16Z
9    7.812456e+13         5629123      F  2016-04-27T12:48:25Z
10   7.345362e+14         5630213      F  2016-04-27T14:58:11Z
11   7.542951e+12         5620163      M  2016-04-26T08:44:12Z
12   5.666548e+14         5634718      F  2016-04-28T11:33:51Z
13   9.113946e+14         5636249      M  2016-04-28T14:52:07Z
14   9.988472e+13         5633951      F  2016-04-28T10:06:24Z
15   9.994839e+10         5620206      F  2016-04-26T08:47:27Z
16   8.457439e+13         5633121      M  2016-04-28T08:51:47Z
17   1.479497e+13         5633460      F  2016-04-28T09:28:57Z
18   1.713538e+13         5621836      F  2016-04-26T10:54:18Z
19   7.223289e+12         5640433      F  2016-04-29T10:43:14Z
20   6.222575e+14         5626083      F  2016-04-27T07:51:14Z
21   1.215484e+13         5628338      F  2016-04-27T10:50:45Z
```

22	8.632298e+14	5616091	M	2016-04-25T13:29:16Z
23	2.137540e+14	5634142	F	2016-04-28T10:27:05Z
24	8.734858e+12	5641780	F	2016-04-29T14:19:19Z
25	5.819370e+12	5624020	M	2016-04-26T15:04:17Z
26	2.578785e+10	5641781	F	2016-04-29T14:19:42Z
27	1.215484e+13	5628345	F	2016-04-27T10:51:45Z
28	5.926172e+12	5642400	M	2016-04-29T15:48:02Z
29	1.225776e+12	5642186	F	2016-04-29T15:16:29Z
...	...	...	...	...
110497	7.935892e+14	5757745	M	2016-06-01T09:46:33Z
110498	9.433654e+13	5787655	F	2016-06-08T10:21:14Z
110499	8.219692e+14	5757697	F	2016-06-01T09:42:56Z
110500	4.434384e+14	5787233	F	2016-06-08T09:35:13Z
110501	4.544252e+11	5758133	M	2016-06-01T10:19:12Z
110502	7.316229e+14	5787937	F	2016-06-08T10:50:42Z
110503	2.362182e+13	5759473	F	2016-06-01T13:00:36Z
110504	9.947983e+12	5788052	F	2016-06-08T11:06:21Z
110505	5.667344e+13	5758455	F	2016-06-01T10:45:50Z
110506	8.973883e+11	5758779	M	2016-06-01T11:09:20Z
110507	4.769462e+14	5786918	F	2016-06-08T09:04:18Z
110508	9.433654e+13	5757656	F	2016-06-01T09:41:00Z
110509	4.952968e+14	5786750	M	2016-06-08T08:50:51Z
110510	2.362182e+13	5757587	F	2016-06-01T09:35:48Z
110511	8.235996e+11	5786742	F	2016-06-08T08:50:20Z
110512	9.876246e+13	5786368	F	2016-06-08T08:20:01Z
110513	8.674778e+13	5785964	M	2016-06-08T07:52:55Z
110514	2.695685e+12	5786567	F	2016-06-08T08:35:31Z
110515	6.456342e+14	5778621	M	2016-06-06T15:58:05Z
110516	6.923772e+13	5780205	F	2016-06-07T07:45:16Z
110517	5.574942e+12	5780122	F	2016-06-07T07:38:34Z
110518	7.263315e+13	5630375	F	2016-04-27T15:15:06Z
110519	6.542388e+13	5630447	F	2016-04-27T15:23:14Z
110520	9.969977e+14	5650534	F	2016-05-03T07:51:47Z
110521	3.635534e+13	5651072	F	2016-05-03T08:23:40Z
110522	2.572134e+12	5651768	F	2016-05-03T09:15:35Z
110523	3.596266e+12	5650093	F	2016-05-03T07:27:33Z
110524	1.557663e+13	5630692	F	2016-04-27T16:03:52Z
110525	9.213493e+13	5630323	F	2016-04-27T15:09:23Z
110526	3.775115e+14	5629448	F	2016-04-27T13:30:56Z

	Appointment_Day	Age	Neighbourhood	Scholarship \
0	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0
1	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0
2	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0
3	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0
4	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0
5	2016-04-29T00:00:00Z	76	REPÚBLICA	0
6	2016-04-29T00:00:00Z	23	GOIABEIRAS	0

7	2016-04-29T00:00:00Z	39	GOIABEIRAS	0
8	2016-04-29T00:00:00Z	21	ANDORINHAS	0
9	2016-04-29T00:00:00Z	19	CONQUISTA	0
10	2016-04-29T00:00:00Z	30	NOVA PALESTINA	0
11	2016-04-29T00:00:00Z	29	NOVA PALESTINA	0
12	2016-04-29T00:00:00Z	22	NOVA PALESTINA	1
13	2016-04-29T00:00:00Z	28	NOVA PALESTINA	0
14	2016-04-29T00:00:00Z	54	NOVA PALESTINA	0
15	2016-04-29T00:00:00Z	15	NOVA PALESTINA	0
16	2016-04-29T00:00:00Z	50	NOVA PALESTINA	0
17	2016-04-29T00:00:00Z	40	CONQUISTA	1
18	2016-04-29T00:00:00Z	30	NOVA PALESTINA	1
19	2016-04-29T00:00:00Z	46	DA PENHA	0
20	2016-04-29T00:00:00Z	30	NOVA PALESTINA	0
21	2016-04-29T00:00:00Z	4	CONQUISTA	0
22	2016-04-29T00:00:00Z	13	CONQUISTA	0
23	2016-04-29T00:00:00Z	46	CONQUISTA	0
24	2016-04-29T00:00:00Z	65	TABUAZEIRO	0
25	2016-04-29T00:00:00Z	46	CONQUISTA	0
26	2016-04-29T00:00:00Z	45	BENTO FERREIRA	0
27	2016-04-29T00:00:00Z	4	CONQUISTA	0
28	2016-04-29T00:00:00Z	51	SÃO PEDRO	0
29	2016-04-29T00:00:00Z	32	SANTA MARTHA	0
...	...	...	...	...
110497	2016-06-01T00:00:00Z	76	MARIA ORTIZ	0
110498	2016-06-08T00:00:00Z	59	MARIA ORTIZ	0
110499	2016-06-01T00:00:00Z	66	MARIA ORTIZ	0
110500	2016-06-08T00:00:00Z	59	MARIA ORTIZ	0
110501	2016-06-01T00:00:00Z	44	MARIA ORTIZ	0
110502	2016-06-08T00:00:00Z	22	GOIABEIRAS	0
110503	2016-06-01T00:00:00Z	64	SOLON BORGES	0
110504	2016-06-08T00:00:00Z	4	MARIA ORTIZ	0
110505	2016-06-01T00:00:00Z	55	MARIA ORTIZ	0
110506	2016-06-01T00:00:00Z	5	MARIA ORTIZ	0
110507	2016-06-08T00:00:00Z	0	MARIA ORTIZ	0
110508	2016-06-01T00:00:00Z	59	MARIA ORTIZ	0
110509	2016-06-08T00:00:00Z	33	MARIA ORTIZ	0
110510	2016-06-01T00:00:00Z	64	SOLON BORGES	0
110511	2016-06-08T00:00:00Z	14	MARIA ORTIZ	0
110512	2016-06-08T00:00:00Z	41	MARIA ORTIZ	0
110513	2016-06-08T00:00:00Z	2	ANTÔNIO HONÓRIO	0
110514	2016-06-08T00:00:00Z	58	MARIA ORTIZ	0
110515	2016-06-08T00:00:00Z	33	MARIA ORTIZ	0
110516	2016-06-08T00:00:00Z	37	MARIA ORTIZ	0
110517	2016-06-07T00:00:00Z	19	MARIA ORTIZ	0
110518	2016-06-07T00:00:00Z	50	MARIA ORTIZ	0
110519	2016-06-07T00:00:00Z	22	MARIA ORTIZ	0
110520	2016-06-07T00:00:00Z	42	MARIA ORTIZ	0

110521	2016-06-07T00:00:00Z	53	MARIA ORTIZ	0
110522	2016-06-07T00:00:00Z	56	MARIA ORTIZ	0
110523	2016-06-07T00:00:00Z	51	MARIA ORTIZ	0
110524	2016-06-07T00:00:00Z	21	MARIA ORTIZ	0
110525	2016-06-07T00:00:00Z	38	MARIA ORTIZ	0
110526	2016-06-07T00:00:00Z	54	MARIA ORTIZ	0

	Hypertension	Diabetes	Alcoholism	Handicap	SMS_received	No_show
0	1	0	0	0	0	No
1	0	0	0	0	0	No
2	0	0	0	0	0	No
3	0	0	0	0	0	No
4	1	1	0	0	0	No
5	1	0	0	0	0	No
6	0	0	0	0	0	Yes
7	0	0	0	0	0	Yes
8	0	0	0	0	0	No
9	0	0	0	0	0	No
10	0	0	0	0	0	No
11	0	0	0	0	1	Yes
12	0	0	0	0	0	No
13	0	0	0	0	0	No
14	0	0	0	0	0	No
15	0	0	0	0	1	No
16	0	0	0	0	0	No
17	0	0	0	0	0	Yes
18	0	0	0	0	1	No
19	0	0	0	0	0	No
20	0	0	0	0	0	Yes
21	0	0	0	0	0	Yes
22	0	0	0	0	1	Yes
23	0	0	0	0	0	No
24	0	0	0	0	0	No
25	1	0	0	0	1	No
26	1	0	0	0	0	No
27	0	0	0	0	0	No
28	0	0	0	0	0	No
29	0	0	0	0	0	No
...	...	...	...	...	...	...
110497	0	0	0	0	0	No
110498	0	0	0	0	0	No
110499	1	1	0	0	0	No
110500	0	0	0	0	0	No
110501	0	0	0	0	0	No
110502	0	0	0	0	0	No
110503	0	0	0	0	0	No
110504	0	0	0	0	0	No
110505	0	0	0	0	0	No

```
110506          0       0        0       0          0     No
110507          0       0        0       0          0     No
110508          0       0        0       0          0     No
110509          0       0        0       0          0     No
110510          0       0        0       0          0     No
110511          0       0        0       0          0     No
110512          0       0        0       0          0     No
110513          0       0        0       0          0     No
110514          0       0        0       0          0     No
110515          1       0        0       0          0    Yes
110516          0       0        0       0          0    Yes
110517          0       0        0       0          0     No
110518          0       0        0       0          1     No
110519          0       0        0       0          1     No
110520          0       0        0       0          1     No
110521          0       0        0       0          1     No
110522          0       0        0       0          1     No
110523          0       0        0       0          1     No
110524          0       0        0       0          1     No
110525          0       0        0       0          1     No
110526          0       0        0       0          1     No

[110527 rows x 14 columns]
```

The above code is renaming the columns with typo errors;
'PatientId' to 'Patient_Id'
'ScheduleDay' to 'schedule_Day
'AppointmentDay' to 'Appointment_Day'
'Handcap' to 'Handicap'
'Hipertension' to 'Hypertension'
'No-show' to 'No_show'

```
In [12]: #convert scheduled_day column datatype from object to datetime
         Patient_Apt['Scheduled_Day'] = pd.to_datetime(Patient_Apt['Scheduled_Day'])
```

From the previous section, it was observed that the schedule_day datatype is object which is inappropraite. The code above is converting the datatype from object datatype to datetime datatype

```
In [13]: #convert appointment_day column datatype from object to datetime
         Patient_Apt['Appointment_Day'] = pd.to_datetime(Patient_Apt['Appointment_Day'])
```

From the previous section, it was observed that the appointment_day datatype is object which is inappropraite. The code above is converting the datatype from object datatype to datetime datatype

```
In [14]: #convert "Patient_Id" from float to integer
         Patient_Apt = Patient_Apt.astype({'Patient_Id' : 'int'})
```

From the previous section, it was observed that the patient_Id datatype is float which is inappropraite. The code above is converting the datatype from float datatype to integer datatype

```
In [15]: #view all datatypes for columns in the dataframe
         print (Patient_Apt.dtypes)
```

```
Patient_Id                  int64
Appointment_ID              int64
Gender                     object
Scheduled_Day      datetime64[ns]
Appointment_Day    datetime64[ns]
Age                         int64
Neighbourhood              object
Scholarship                 int64
Hypertension                int64
Diabetes                    int64
Alcoholism                  int64
Handicap                    int64
SMS_received                int64
No_show                    object
dtype: object
```

All the columns in the dataframe now have their appropraite data type

```
In [16]: #rows with age less than 1
         Patient_Apt[Patient_Apt.Age < 0]
```

```
Out[16]:             Patient_Id  Appointment_ID Gender       Scheduled_Day  \
         99832  465943158731293         5775010      F 2016-06-06 08:58:13

               Appointment_Day  Age Neighbourhood  Scholarship  Hypertension  Diabetes  \
         99832      2016-06-06   -1         ROMÃO            0             0         0

                Alcoholism  Handicap  SMS_received No_show
         99832           0         0             0      No
```

The above code brings out the row that has age less than 0 which is -1. I consider this an error and resolve that the row would be dropped.

```
In [17]: #dropping rows with age less than 1
         Patient_Apt.drop(Patient_Apt[Patient_Apt['Age'] < 0]. index, inplace = True)
```

The code above code is deleting the row that has 'age' less than 0

```
In [18]: #rows older than 100 years
         Patient_Apt[Patient_Apt.Age > 100]
```

```
Out[18]:              Patient_Id  Appointment_ID Gender       Scheduled_Day  \
        58014  976294799775439         5651757      F 2016-05-03 09:14:53
        63912   31963211613981         5700278      F 2016-05-16 09:17:44
        63915   31963211613981         5700279      F 2016-05-16 09:17:44
        68127   31963211613981         5562812      F 2016-04-08 14:29:17
        76284   31963211613981         5744037      F 2016-05-30 09:44:51
        90372     234283596548         5751563      F 2016-05-31 10:19:49
        97666  748234579244724         5717451      F 2016-05-19 07:57:56

             Appointment_Day  Age Neighbourhood  Scholarship  Hypertension  Diabetes  \
        58014      2016-05-03  102     CONQUISTA            0             0         0
        63912      2016-05-19  115    ANDORINHAS            0             0         0
        63915      2016-05-19  115    ANDORINHAS            0             0         0
        68127      2016-05-16  115    ANDORINHAS            0             0         0
        76284      2016-05-30  115    ANDORINHAS            0             0         0
        90372      2016-06-02  102   MARIA ORTIZ            0             0         0
        97666      2016-06-03  115      SÃO JOSÉ            0             1         0

              Alcoholism  Handicap  SMS_received No_show
        58014          0         0             0      No
        63912          0         1             0     Yes
        63915          0         1             0     Yes
        68127          0         1             0     Yes
        76284          0         1             0      No
        90372          0         0             0      No
        97666          0         0             1      No
```

The above code shows rows that have age greater than 100. I resolve to drop rows with age 115 because this may result to outliers since the age range is too far off and it is rare to see people of that age.

```
In [19]: #keeping rows with age less than 115
         Patient_Apt = Patient_Apt[(Patient_Apt.Age < 115)]
```

The above code will keep only rows with ages less than 115

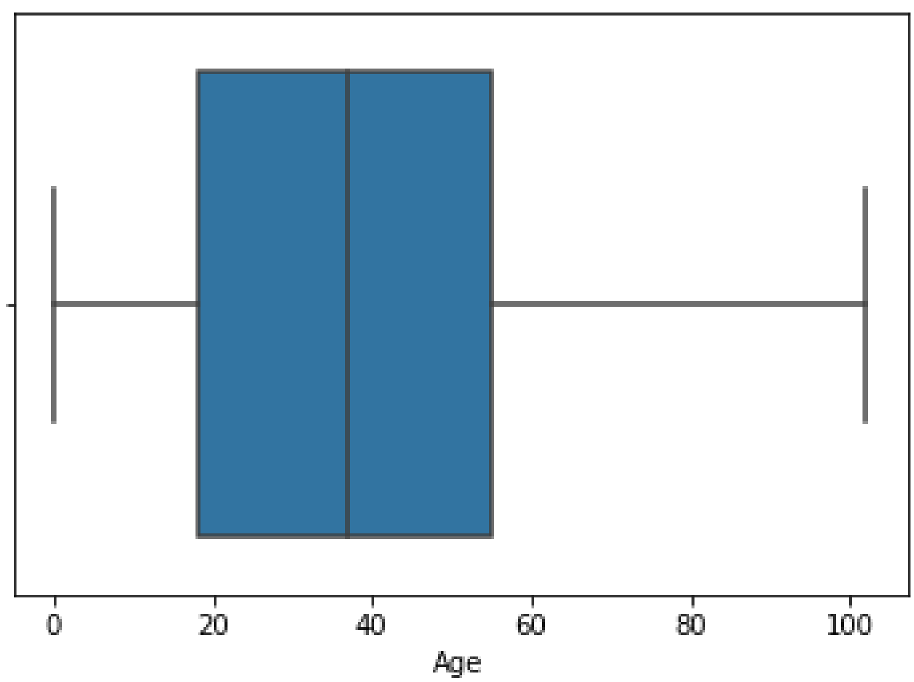
```
In [20]: Patient_Apt.describe()
```

```
Out[20]:          Patient_Id  Appointment_ID            Age    Scholarship  \
        count  1.105210e+05    1.105210e+05  110521.000000  110521.000000
        mean   1.474921e+14    5.675304e+06      37.085694       0.098271
        std    2.560928e+14    7.129576e+04      23.104606       0.297682
        min    3.921700e+04    5.030230e+06       0.000000       0.000000
        25%    4.172457e+12    5.640285e+06      18.000000       0.000000
        50%    3.172598e+13    5.680569e+06      37.000000       0.000000
        75%    9.438963e+13    5.725523e+06      55.000000       0.000000
        max    9.999816e+14    5.790484e+06     102.000000       1.000000

                Hypertension       Diabetes     Alcoholism       Handicap  \
```

```
count  110521.000000  110521.000000  110521.000000  110521.000000
mean        0.197248       0.071869       0.030401       0.022213
std         0.397923       0.258272       0.171690       0.161440
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  110521.000000
mean        0.321034
std         0.466876
min         0.000000
25%         0.000000
50%         0.000000
75%         1.000000
max         1.000000
```

We can see now that the minimum age is 0 (the patient could be babies that are not up to 1 year old) and the maximum age is 102 years old

In [21]:
```
#using a boxplot to check for outliers in age
sns.boxplot(Patient_Apt.Age)
plt.show()
```



The boxplot is used to check if there is any outlier in the age column. As we can see, there is no outlier.

In [22]:
```
#change 'No' to 'showed' in the 'No_show' column
Patient_Apt.loc [Patient_Apt['No_show'] == 'No', 'No_show'] = 'Showed'

#change 'Yes' to 'Missed' in the 'No_show' column
Patient_Apt.loc [Patient_Apt['No_show'] == 'Yes', 'No_show'] = 'Missed'
```

In [23]:
```
#viewing the first 10 rows of the dataframe to see if the change has been effected in t
Patient_Apt.head(10)
```

Out[23]:
```
        Patient_Id  Appointment_ID Gender       Scheduled_Day Appointment_Day  \
0   29872499824296         5642903      F 2016-04-29 18:38:08      2016-04-29
1  558997776694438         5642503      M 2016-04-29 16:08:27      2016-04-29
2    4262962299951         5642549      F 2016-04-29 16:19:04      2016-04-29
3     867951213174         5642828      F 2016-04-29 17:29:31      2016-04-29
4    8841186448183         5642494      F 2016-04-29 16:07:23      2016-04-29
5   95985133231274         5626772      F 2016-04-27 08:36:51      2016-04-29
6  733688164476661         5630279      F 2016-04-27 15:05:12      2016-04-29
7    3449833394123         5630575      F 2016-04-27 15:39:58      2016-04-29
8   56394729949972         5638447      F 2016-04-29 08:02:16      2016-04-29
9   78124564369297         5629123      F 2016-04-27 12:48:25      2016-04-29

   Age      Neighbourhood  Scholarship  Hypertension  Diabetes  Alcoholism  \
0   62    JARDIM DA PENHA            0             1         0           0
1   56    JARDIM DA PENHA            0             0         0           0
2   62      MATA DA PRAIA            0             0         0           0
3    8  PONTAL DE CAMBURI            0             0         0           0
4   56    JARDIM DA PENHA            0             1         1           0
5   76          REPÚBLICA            0             1         0           0
6   23         GOIABEIRAS            0             0         0           0
7   39         GOIABEIRAS            0             0         0           0
8   21         ANDORINHAS            0             0         0           0
9   19          CONQUISTA            0             0         0           0

   Handicap  SMS_received No_show
0         0             0  Showed
1         0             0  Showed
2         0             0  Showed
3         0             0  Showed
4         0             0  Showed
5         0             0  Showed
6         0             0  Missed
7         0             0  Missed
8         0             0  Showed
9         0             0  Showed
```

The 'yes' and 'No' entries in the 'no_show' column can get a bit confusing. For a clearer picture and better understanding, I opted to change the *'No'* entry to *'Showed'* denoting patients that showed up to their appointments and *'Yes'* entry to *'Missed'* indicating those that didnt show up for their appointment.

Next, I'd be dropping columns that are not needed for this analysis

```
In [24]: #dropping columns not needed for analysis
         Patient_Apt.drop(['Patient_Id', 'Appointment_ID'], axis = 1, inplace = True)
```

```
In [25]: Patient_Apt.head(3)
```

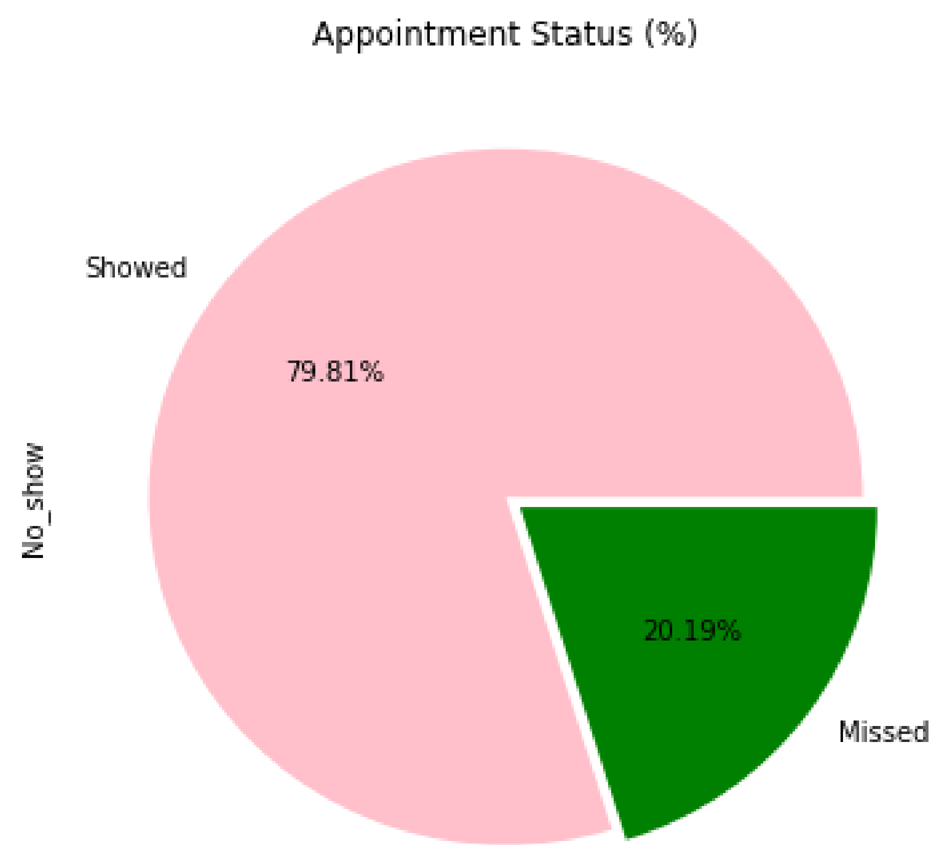
```
Out[25]:   Gender        Scheduled_Day Appointment_Day  Age    Neighbourhood  \
         0      F 2016-04-29 18:38:08      2016-04-29   62  JARDIM DA PENHA
         1      M 2016-04-29 16:08:27      2016-04-29   56  JARDIM DA PENHA
         2      F 2016-04-29 16:19:04      2016-04-29   62    MATA DA PRAIA

            Scholarship  Hypertension  Diabetes  Alcoholism  Handicap  SMS_received  \
         0            0             1         0           0         0             0
         1            0             0         0           0         0             0
         2            0             0         0           0         0             0

           No_show
         0  Showed
         1  Showed
         2  Showed
```

The column names 'Patient_iD' and 'Appointment_ID has been dropped because it will not be used for this analysis

Exploratory Data Analysis

1.1.4 Research Question 1: What Percentage of Patients Showed Up for Appointment

```
In [26]: #plotting a piechart to show the percentage of patients that missed an appointment and
         Patient_Apt.No_show.value_counts().plot.pie(figsize=(6,6), colors = ['pink', 'green'],
         plt.show()
```

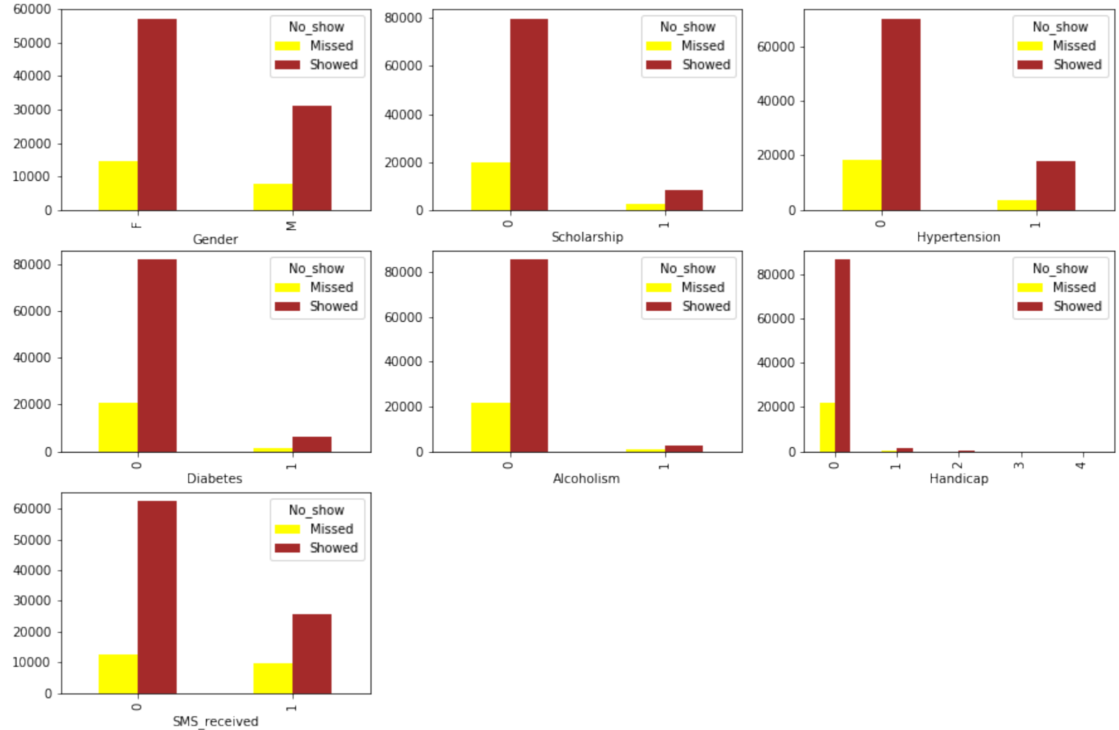


The bar chart shows that about 78.81% of patients showed up for their appointment and 20.19% missed their appointment

1.1.5 Research Question 2: Is there one feature that absolutely influence a patient not showing up to an appointment

To answer research question 2, we will look at both categorical features and numerical features seperately to see if any of the features can absolutely influence a patient not showing up to an appointment.

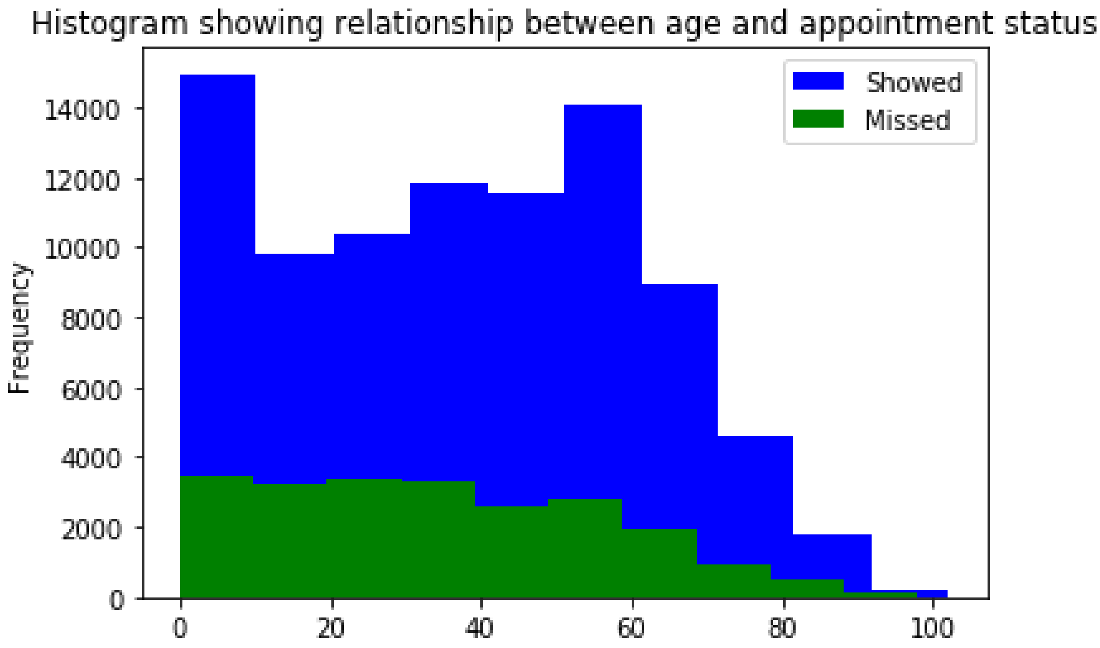
```
In [27]: #storing all the categorical column in a variable named 'categorical features'
         categorical_features = ['Gender', 'Scholarship', 'Hypertension', 'Diabetes', 'Alcoholis

         #for every feature in 'categorical_features', plot a bar chart grouped by the 'No_show'
         chart = plt.figure(figsize=(15, 10))
         for i, feature in enumerate(categorical_features):
             ax = chart.add_subplot(3, 3, i+1)
             colours = ['Yellow','Brown']
             Patient_Apt.groupby([feature, 'No_show'])[feature].count().unstack('No_show').plot(
```



Looking at all the categorical features at a glance, the bars look the same. A lot of patients are not handicapped, hypertensive, diabetic or alcoholic. I conclude that there is no significant categorical features that can influence a patient from missing an appointment.

Next, lets look at numerical variables

In [28]:
```
#declaring that the 'showed' and 'missed' entries in "No_show" column is equivalent to
Showed = (Patient_Apt.No_show == 'Showed')
Missed = (Patient_Apt.No_show == 'Missed')

#plotting a histogram showing the relationship between age and appointment status (shou
Patient_Apt[Showed].Age.plot(kind = 'hist', color = 'Blue')
Patient_Apt[Missed].Age.plot(kind = 'hist', color = 'Green')
plt.legend(['Showed', 'Missed'])
plt.title('Histogram showing relationship between age and appointment status')
plt.show()
```



The histogram above shows that age can affect whether patients show up to appointments or not.

We can see that from age 0 to about 3 years old (babies and infants) show up to appointments more. Then there is a decrease and then an increase at about age 50. Finally, there is subsequent decrease in appointment as the age gets older towards age 100.

1.1.6 Research Question 3: What gender show up more for appointment?

```
In [29]: #showing the number of patients that missed their appointment and those who showed up
         Patient_Apt[["Gender", "No_show"]].groupby("No_show").count()
```

```
Out[29]:           Gender
         No_show
         Missed     22316
         Showed     88205
```

The code above is giving a total count of patients that showed up and those that missed their appointment. A total of 22316 missed their appointment and a total of 88205 patients showed up.

```
In [30]: #showing the number of females that showed up and missed their appointment, also showin
         Patient_Apt.groupby('Gender').No_show.value_counts()
```

```
Out[30]: Gender  No_show
         F       Showed     57243
                 Missed     14591
         M       Showed     30962
                 Missed      7725
         Name: No_show, dtype: int64
```

After getting the total count of patients that showed up and missed their appointment, the code above is grouping the number of patients that missed/ showed by gender. i.e how many males showed up and how many didn't? likewise, how many females showed up and how many didn't show up?

In [31]:
```
#getting rows of all female appointment and male appointment
female_appointment = len(Patient_Apt.loc[Patient_Apt['Gender'] == "F"])
male_appointment= len(Patient_Apt.loc[Patient_Apt['Gender'] == "M"])
```

The code above is getting all the entries of all females and males that have an appointment and storing it in the variable named 'female_appointment' and 'male_appointment' respectively

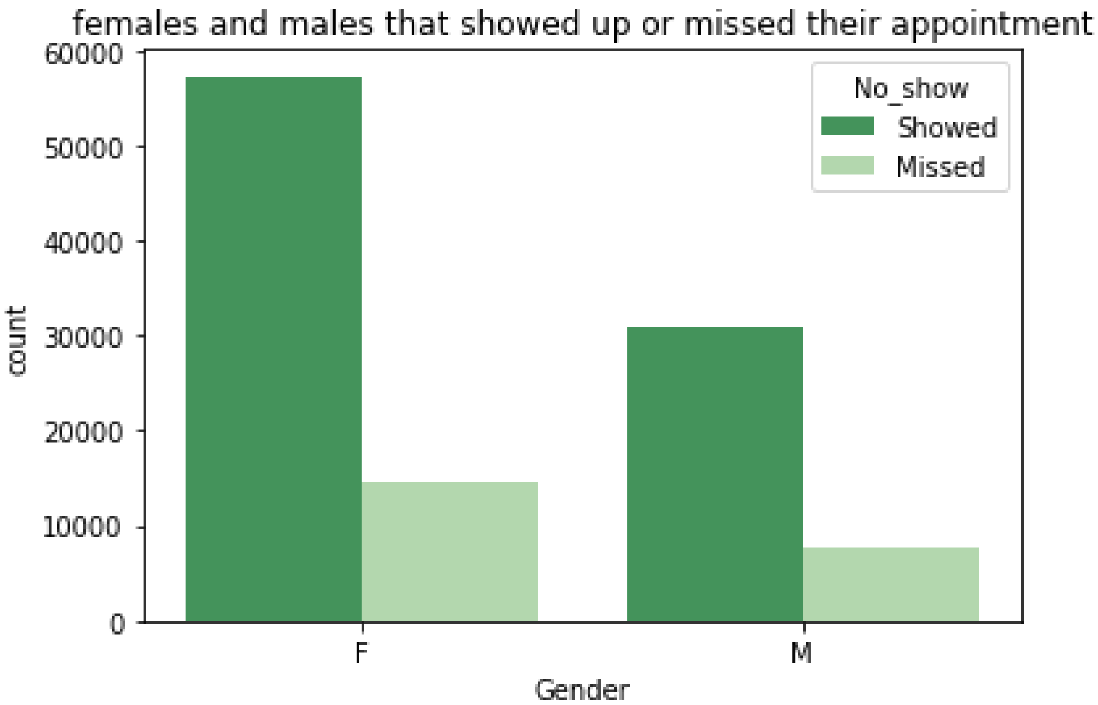
In [32]:
```
#getting rows of all female and male that missed their appointment
missedApt_female = len(Patient_Apt.query('No_show == "Missed" and Gender == "F"'))
missedApt_male = len(Patient_Apt.loc[(Patient_Apt['Gender'] == "M") & (Patient_Apt['No_
```

The code above is getting all the rows of females and males that missed their appointment and storing it in the variable named 'missedApt_female' and 'missedApt_male' respectively

In [33]:
```
#ratio of missed appointment to total appointment for female and male
ratio_female = int(round(missedApt_female/female_appointment*100))
ratio_male = int(round(missedApt_male/male_appointment*100))
```

The code above gives a ratio of the female appointment by dividing the number of missed appointment by total appointment and multiplying the number by 100.

In [34]:
```
#plotting the graph of the number of females and males that showed up or missed their a

ax = sns.countplot(x=Patient_Apt.Gender, hue=Patient_Apt.No_show, data=Patient_Apt, pal
ax.set_title("females and males that showed up or missed their appointment")
x_ticks_labels=['female', 'male']
plt.show();
```



57,243 females out of 71,834 showed up for their appointment and **14,591 females** missed their appointment **30,962 males** out of 38,687 showed up for their appointment and **7,725 males** missed their appointment

1.1.7 Research Question 4: Does Recieving SMS Reminder Influence a Patient to Show Up for Appointment?

In [35]:
```
#grouping the number of patients that recieved sms into show_up or missed and storing i
sms_reminder = Patient_Apt.groupby('SMS_received').No_show.value_counts()
```

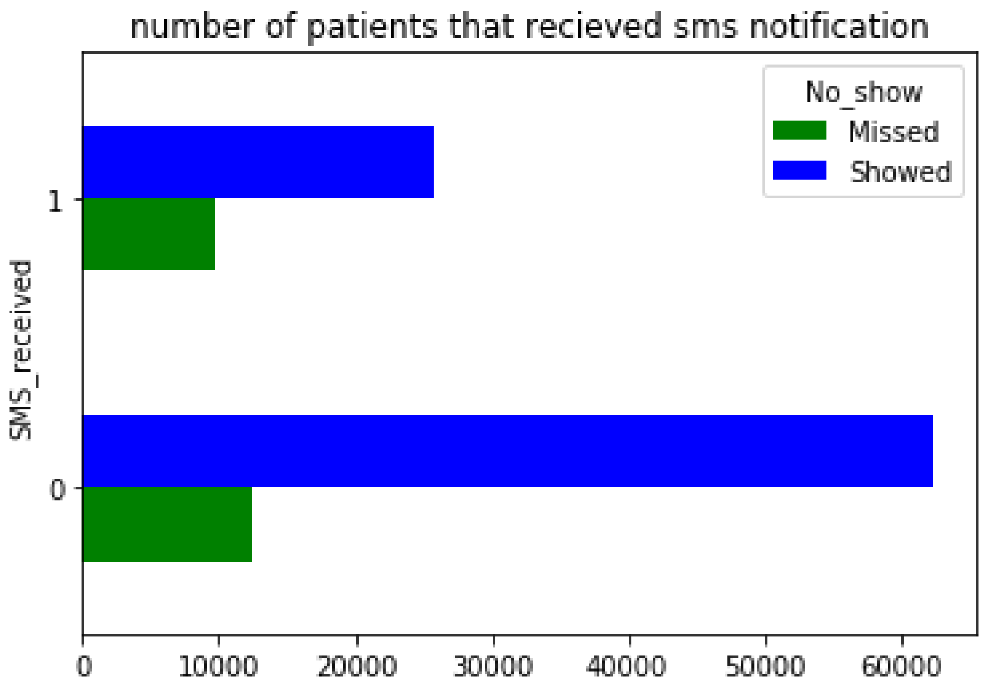
In [36]:
```
#view the grouping of people that recieved sms
sms_reminder
```

Out[36]:
```
SMS_received  No_show
0             Showed     62508
              Missed     12532
1             Showed     25697
              Missed      9784
Name: No_show, dtype: int64
```

In [37]:
```
#creating a dictinary that translate 1 and 0 to 'received' and 'not recieved'
sms_status = {1:'received', 0:'Not received'}

#plotting a chart that shows the number of patients that recieved an sms reminder
sms_reminder = sms_reminder.unstack()
```

```
sms_reminder.plot(kind='barh', color = ['green', 'blue'], title = 'number of patients t
print ('Patients that did not recieve sms = 0 and Patients that recieved sms = 1' )
```

```
Patients that did not recieve sms = 0 and Patients that recieved sms = 1
```



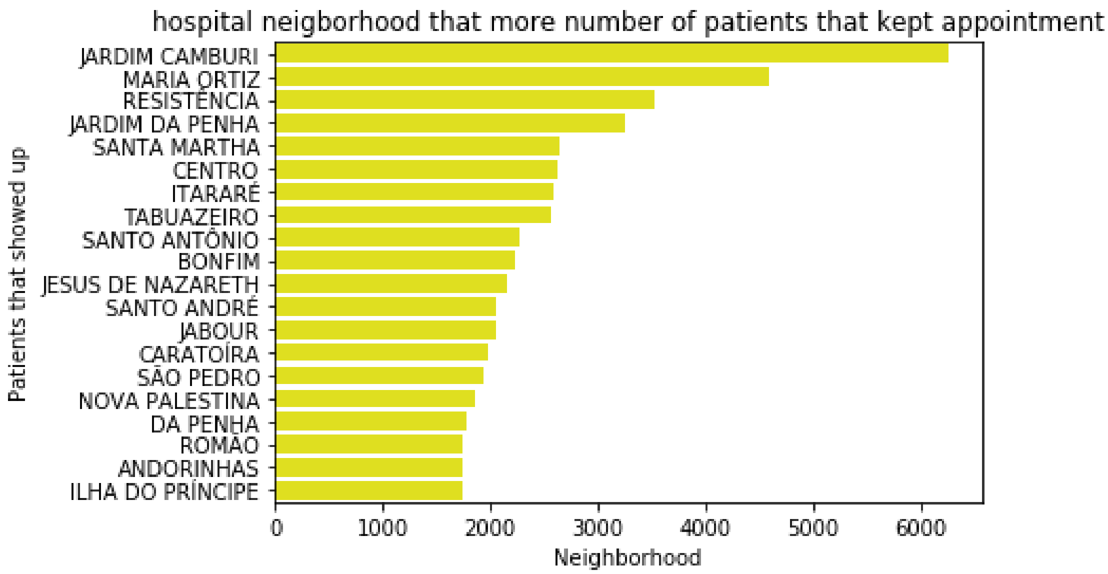
I thought to ask this question because an early sms reminder can help patients avoid missing their appointment. I considered it to be a major factor to influence a patient for showing up for appointment.

Surprisingly after the analysis from the plot above, we can see that patients that showed up more for their appointment did not recieve sms. Therefore receiving sms is not a major feature that influences if a patient will show up or not

1.1.8 Research Question 5: What Neigborhood Hospital Recieve the Most Appointment?

```
In [38]: #getting only the number of patients that showed for appointment
         showed_patients = Patient_Apt[Patient_Apt.No_show == 'Showed']

         #getting the top 20 neigbourhood hospitals with high count of appointment
         hospital_neigbourhood = showed_patients['Neighbourhood'].value_counts().index[:20]
```

```
In [39]: #plotting the number of patients that showed up to the neigborhood hospital
         sns.countplot(data = showed_patients, y= 'Neighbourhood', color = 'Yellow', order = hos
         plt.xlabel('Neighborhood')
         plt.ylabel('Patients that showed up')
         plt.title ('hospital neigborhood that more number of patients that kept appointment')
         plt.show();
```



I plotted a chart that shows the top 20 neigbourhood hospitals that receive the highest number of patients that showed up for appointment. From the above chart, the hospital at 'Jardim Camburi' has the highest number of appointments. They recieved about 6000 patients.

Conclusions

SUMMARY OF FINDINGS: From the data analysis, it was discovered that age is an important factor that can influence a patient from not showing up to a medical appointment. This was seen as a decline in the number of people that showed up as the age increases towards 102 years old. This could be because older people get weary, too tired or even forget to keep to an appointment. Babies and infants (0- about 3years) showed to be the age group with more appointment. This might be because their immune system is still weak and they tend to visit the hospitals regularly. Another reason could be because infants are accompanied by a guardian/parent, so they tend to show up more. Other features such as being an alcoholic, neigborhood of the hospital, medical condition are not strong factors to detemine if a patient will show up for appointment or not. A great number of patients are not hypertensive, diabetic, alcholic and hadicapped.

Another conclusion from the analysis is that females tend to show up more to medical appointments than males, probably because women are more health conscious than males.

Also, contrary to what was expected, there was no link between SMS received and No Shows. And, as always, it's important to remember that a link between two things does not mean that one caused the other.

LIMITATION: One limitation of this analysis was the interpretability of the 'No_show' column. It was very confusing mapping patients who showed up to be 'No' and those who didn't to be 'Yes'. To avoid mixing things up, I had to change 'Yes' to be 'Missed' and 'No' to be 'Showed_up'.

SUGGESTION: The dataset could have had a column that shows the distance from the patient's neigborhood to the hospital. It could have been helpful to see if the distance from the patient's home to the hospital is a factor that influence whether or not a patient would show up for an appointment.

1.2 Submitting your Project

Tip: Before you submit your project, you need to create a .html or .pdf version of this notebook in the workspace here. To do that, run the code cell below. If it worked correctly, you should get a return code of 0, and you should see the generated .html file in the workspace directory (click on the orange Jupyter icon in the upper left).

Tip: Alternatively, you can download this report as .html via the **File** > **Download as** submenu, and then manually upload it into the workspace directory by clicking on the orange Jupyter icon in the upper left, then using the Upload button.

Tip: Once you've done this, you can submit your project by clicking on the "Submit Project" button in the lower right here. This will create and submit a zip file with this .ipynb doc and the .html or .pdf version you created. Congratulations!

```
In [40]: from subprocess import call
         call(['python', '-m', 'nbconvert', 'Investigate_a_Dataset.ipynb'])

Out[40]: 0
```