appointment data analysis

December 14, 2022

1 Project: Appointment dataset Analysis

1.1 Introduction

I will be working on the 'noshowappointments-kagglev2-may-2016.csv' dataset. The dataset provides information on patients and whether or not they showed up for their appointments

1.2 Questions

1.3 Below are the questions I will be addressing

1.

- how many females and males are there in the dataset?
- how many males and females showed up for their appointment and what's the percentage ?
- how many males and females missed their appointment and what's the percentage?

2.how many females received sms and showed up as against those that did not receive any sms but still showed up?

3.

- what percentage of females are diabetic?
- are there female children who are diabetic? how many?
- how many diabetics and alcoholics showed up for the appointment and how many did not?

1.4 Data Wrangling

In this section of the report, I will load in the data, check for cleanliness, and then trim and clean the dataset for analysis.

I will inspect the dataset and check for : * datatypes * missing data values or null values * duplicates * and among others where necessary

Import the required packages and load the dataset

[1]:
```
# import the required packages
# and load the dataset

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

```
%matplotlib inline

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

1.4.1 Inspect the dataset (General overview)

[2]:
```
# print/look at the first line of the dataset
df.head(1)
```

[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
```

[3]:
```
# print/look at the first five lines of the dataset
df.head()
```

[3]:
```
      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
```

[4]:
```
# check the shape or dimensions of the dataset(rows and columns)
df.shape
```

[4]:
```
(110527, 14)
```

```
[5]: # inspect the columns
     df.columns
```

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

```
[6]: # funtion to print the columns of the dataset
     def printColumns():
       for i,v in enumerate(df):
         print(i,v)
     printColumns()
```

```
0 PatientId
1 AppointmentID
2 Gender
3 ScheduledDay
4 AppointmentDay
5 Age
6 Neighbourhood
7 Scholarship
8 Hipertension
9 Diabetes
10 Alcoholism
11 Handcap
12 SMS_received
13 No-show
```

from the above, we can tell that some of the columns have typos, let's rename them and also rename the last column 'No-show' to 'No_show' to get a consistent naming format

```
[7]: #let's rename them and also rename the last column 'No-show' to 'No_show' to␣
      ↪get a consistent naming format
     df.rename(columns ={'Hipertension' : 'Hypertension', 'Handcap' : 'Handicap', ␣
      ↪'No-show' :'No_show'}, inplace = True)
```

```
[8]: # confirm the changes by printing a few lines of the dataset
     df.head()
```

```
[8]:       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  Hypertension  \
     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
```

1.5 Inspect the datasets(detailed)

check for : * datatypes * missing data values or null values * duplicates * and among others where necessary

[9]:
```
# use the info() to inspect the dataset
# use pandas's info function to get a concise summary of our dataset.
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   Hypertension    110527 non-null  int64
 9   Diabetes        110527 non-null  int64
 10  Alcoholism      110527 non-null  int64
 11  Handicap        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
```

change the ‘ScheduledDay’ and ‘AppointmentDay’ datatypes to datetime

[10]:
```
#change the 'ScheduledDay' and 'AppointmentDay' datatypes to datetime
df['ScheduledDay'] = pd.to_datetime(df['ScheduledDay'])
```

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

[11]:
```
# let's confirm the changes using the info() function
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  datetime64[ns, UTC]
 4   AppointmentDay  110527 non-null  datetime64[ns, UTC]
 5   Age             110527 non-null  int64
 6   Neighbourhood   110527 non-null  object
 7   Scholarship     110527 non-null  int64
 8   Hypertension    110527 non-null  int64
 9   Diabetes        110527 non-null  int64
 10  Alcoholism      110527 non-null  int64
 11  Handicap        110527 non-null  int64
 12  SMS_received    110527 non-null  int64
 13  No_show         110527 non-null  object
dtypes: datetime64[ns, UTC](2), float64(1), int64(8), object(3)
memory usage: 11.8+ MB
```

[12]:
```
# display first five rows
df.head()
```

[12]:
```
      PatientId  AppointmentID Gender              ScheduledDay  \
0  2.987250e+13        5642903      F 2016-04-29 18:38:08+00:00
1  5.589978e+14        5642503      M 2016-04-29 16:08:27+00:00
2  4.262962e+12        5642549      F 2016-04-29 16:19:04+00:00
3  8.679512e+11        5642828      F 2016-04-29 17:29:31+00:00
4  8.841186e+12        5642494      F 2016-04-29 16:07:23+00:00

             AppointmentDay  Age      Neighbourhood  Scholarship  \
0 2016-04-29 00:00:00+00:00   62    JARDIM DA PENHA            0
1 2016-04-29 00:00:00+00:00   56    JARDIM DA PENHA            0
2 2016-04-29 00:00:00+00:00   62      MATA DA PRAIA            0
3 2016-04-29 00:00:00+00:00    8  PONTAL DE CAMBURI            0
4 2016-04-29 00:00:00+00:00   56    JARDIM DA PENHA            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
```

[13]:
```
# check the number of unique entries
df.nunique()
```

[13]:
```
PatientId          62299
AppointmentID     110527
Gender                 2
ScheduledDay      103549
AppointmentDay        27
Age                  104
Neighbourhood         81
Scholarship            2
Hypertension           2
Diabetes               2
Alcoholism             2
Handicap               5
SMS_received           2
No_show                2
dtype: int64
```

[14]:
```
# check for missing data entries
df.isnull().sum()
```

[14]:
```
PatientId         0
AppointmentID     0
Gender            0
ScheduledDay      0
AppointmentDay    0
Age               0
Neighbourhood     0
Scholarship       0
Hypertension      0
Diabetes          0
Alcoholism        0
Handicap          0
SMS_received      0
No_show           0
dtype: int64
```

From the above operation, it's clear that there are no missing data entries in the dataset

[15]:
```
# check for possible duplicates in the dataset
df.duplicated().sum()
```

[15]:
```
0
```

From the above operation, it's clear that there are no duplicates in the dataset

1.5.1 inspect the columns using unique() function

```
[16]: # check unique entries of the AppointmentID column
      df.AppointmentID.unique()
```

```
[16]: array([5642903, 5642503, 5642549, …, 5630692, 5630323, 5629448],
            dtype=int64)
```

```
[17]: # check the unique entries of the Gender column
      df.Gender.unique()
```

```
[17]: array(['F', 'M'], dtype=object)
```

```
[18]: # check the unique entries of the ScheduledDay column
      df.ScheduledDay.unique()
```

```
[18]: <DatetimeArray>
      ['2016-04-29 18:38:08+00:00', '2016-04-29 16:08:27+00:00',
       '2016-04-29 16:19:04+00:00', '2016-04-29 17:29:31+00:00',
       '2016-04-29 16:07:23+00:00', '2016-04-27 08:36:51+00:00',
       '2016-04-27 15:05:12+00:00', '2016-04-27 15:39:58+00:00',
       '2016-04-29 08:02:16+00:00', '2016-04-27 12:48:25+00:00',
       …
       '2016-06-07 07:45:16+00:00', '2016-06-07 07:38:34+00:00',
       '2016-04-27 15:15:06+00:00', '2016-05-03 07:51:47+00:00',
       '2016-05-03 08:23:40+00:00', '2016-05-03 09:15:35+00:00',
       '2016-05-03 07:27:33+00:00', '2016-04-27 16:03:52+00:00',
       '2016-04-27 15:09:23+00:00', '2016-04-27 13:30:56+00:00']
      Length: 103549, dtype: datetime64[ns, UTC]
```

```
[19]: # check the unique entries of the Age column

      df.Age.unique()
```

```
[19]: array([ 62,  56,   8,  76,  23,  39,  21,  19,  30,  29,  22,  28,  54,
              15,  50,  40,  46,   4,  13,  65,  45,  51,  32,  12,  61,  38,
              79,  18,  63,  64,  85,  59,  55,  71,  49,  78,  31,  58,  27,
               6,   2,  11,   7,   0,   3,   1,  69,  68,  60,  67,  36,  10,
              35,  20,  26,  34,  33,  16,  42,   5,  47,  17,  41,  44,  37,
              24,  66,  77,  81,  70,  53,  75,  73,  52,  74,  43,  89,  57,
              14,   9,  48,  83,  72,  25,  80,  87,  88,  84,  82,  90,  94,
              86,  91,  98,  92,  96,  93,  95,  97, 102, 115, 100,  99,  -1],
            dtype=int64)
```

```
[20]: # from the above operation we observed that there's a -1 value in the Age column
      # let's check to confirm for sure if there are any more negative values
      df[df.Age < 0]
```

```
[20]:           PatientId  AppointmentID Gender             ScheduledDay  \
      99832  4.659432e+14        5775010      F 2016-06-06 08:58:13+00:00

                       AppointmentDay  Age Neighbourhood  Scholarship  Hypertension  \
      99832 2016-06-06 00:00:00+00:00   -1         ROMÃO            0             0

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

```
[21]: # same operation performed above but using the query() funtion
      df.query('Age < 0')
```

```
[21]:           PatientId  AppointmentID Gender             ScheduledDay  \
      99832  4.659432e+14        5775010      F 2016-06-06 08:58:13+00:00

                       AppointmentDay  Age Neighbourhood  Scholarship  Hypertension  \
      99832 2016-06-06 00:00:00+00:00   -1         ROMÃO            0             0

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

we found out that the only negative value was -1 which is an unrealistic value for age so we will treat it as an outlier and thus remove it from the original dataset

```
[22]: # remove the outlier from the age column and re-assign the change to the␣
       ↪original dataset
      # the new dataset will not contain the outlier removed
      df = df[df.Age >  0]
```

let's confirm if it has been removed

```
[23]: # confirm if it has been removed
      df[df.Age <  0 ]
```

```
[23]: Empty DataFrame
      Columns: [PatientId, AppointmentID, Gender, ScheduledDay, AppointmentDay, Age,
      Neighbourhood, Scholarship, Hypertension, Diabetes, Alcoholism, Handicap,
      SMS_received, No_show]
      Index: []
```

from the above we can confirm that it has been removed. now let's check the dimensions of the orignal dataset after the -1 has been removed and the unique values of the Age column to further confirm

```
[24]: # print the shape/dimension of the dataset after remove the outlier (-1)
      df.shape
```

```
[24]: (106987, 14)
```

```
[25]: # re-check the number of unique entries of the Age column
      df.Age.unique()
```

```
[25]: array([ 62,  56,   8,  76,  23,  39,  21,  19,  30,  29,  22,  28,  54,
              15,  50,  40,  46,   4,  13,  65,  45,  51,  32,  12,  61,  38,
              79,  18,  63,  64,  85,  59,  55,  71,  49,  78,  31,  58,  27,
               6,   2,  11,   7,   3,   1,  69,  68,  60,  67,  36,  10,  35,
              20,  26,  34,  33,  16,  42,   5,  47,  17,  41,  44,  37,  24,
              66,  77,  81,  70,  53,  75,  73,  52,  74,  43,  89,  57,  14,
               9,  48,  83,  72,  25,  80,  87,  88,  84,  82,  90,  94,  86,
              91,  98,  92,  96,  93,  95,  97, 102, 115, 100,  99], dtype=int64)
```

we can confirm from the above that the -1 has been removed

```
[26]: # check the unique entries of the Handicap column
      df.Handicap.unique()
```

```
[26]: array([0, 1, 2, 3, 4], dtype=int64)
```

```
[27]: # check the unique entries of the Neighbourhood column
      df['Neighbourhood'].unique()
```

```
[27]: array(['JARDIM DA PENHA', 'MATA DA PRAIA', 'PONTAL DE CAMBURI',
             'REPÚBLICA', 'GOIABEIRAS', 'ANDORINHAS', 'CONQUISTA',
             'NOVA PALESTINA', 'DA PENHA', 'TABUAZEIRO', 'BENTO FERREIRA',
             'SÃO PEDRO', 'SANTA MARTHA', 'SÃO CRISTÓVÃO', 'MARUÍPE',
             'GRANDE VITÓRIA', 'SANTO ANDRÉ', 'SOLON BORGES', 'BONFIM',
             'JARDIM CAMBURI', 'MARIA ORTIZ', 'JABOUR', 'ANTÔNIO HONÓRIO',
             'RESISTÊNCIA', 'ILHA DE SANTA MARIA', 'JUCUTUQUARA',
             'MÁRIO CYPRESTE', 'SANTO ANTÔNIO', 'BELA VISTA', 'PRAIA DO SUÁ',
             'SANTA HELENA', 'ITARARÉ', 'INHANGUETÁ', 'UNIVERSITÁRIO',
             'SÃO JOSÉ', 'REDENÇÃO', 'SANTA CLARA', 'CENTRO', 'PARQUE MOSCOSO',
             'DO MOSCOSO', 'SANTOS DUMONT', 'CARATOÍRA', 'ARIOVALDO FAVALESSA',
             'ILHA DO FRADE', 'GURIGICA', 'JOANA D´ARC', 'CONSOLAÇÃO',
             'SÃO BENEDITO', 'PRAIA DO CANTO', 'BOA VISTA', 'SANTA LÚCIA',
             'BARRO VERMELHO', 'ESTRELINHA', 'FORTE SÃO JOÃO', 'FONTE GRANDE',
             'MORADA DE CAMBURI', 'ENSEADA DO SUÁ', 'SANTOS REIS', 'PIEDADE',
             'JESUS DE NAZARETH', 'SANTA LUÍZA', 'SANTA TEREZA', 'CRUZAMENTO',
             'ILHA DO PRÍNCIPE', 'ROMÃO', 'ILHA DAS CAIEIRAS', 'COMDUSA',
             'SANTA CECÍLIA', 'VILA RUBIM', 'DE LOURDES', 'MONTE BELO',
             'DO QUADRO', 'DO CABRAL', 'HORTO', 'SEGURANÇA DO LAR',
             'ILHA DO BOI', 'FRADINHOS', 'NAZARETH', 'AEROPORTO',
             'ILHAS OCEÂNICAS DE TRINDADE', 'PARQUE INDUSTRIAL'], dtype=object)
```

```
[28]: # convert the Neighbourhood column to string
      # convert from 'object' datatype  to string datatype
      df['Neighbourhood'] = df['Neighbourhood'].astype(str)
```

[29]:
```
# check the unique entries of the Neighbourhood column
df['Neighbourhood'].unique()
```

[29]:
```
array(['JARDIM DA PENHA', 'MATA DA PRAIA', 'PONTAL DE CAMBURI',
       'REPÚBLICA', 'GOIABEIRAS', 'ANDORINHAS', 'CONQUISTA',
       'NOVA PALESTINA', 'DA PENHA', 'TABUAZEIRO', 'BENTO FERREIRA',
       'SÃO PEDRO', 'SANTA MARTHA', 'SÃO CRISTÓVÃO', 'MARUÍPE',
       'GRANDE VITÓRIA', 'SANTO ANDRÉ', 'SOLON BORGES', 'BONFIM',
       'JARDIM CAMBURI', 'MARIA ORTIZ', 'JABOUR', 'ANTÔNIO HONÓRIO',
       'RESISTÊNCIA', 'ILHA DE SANTA MARIA', 'JUCUTUQUARA',
       'MÁRIO CYPRESTE', 'SANTO ANTÔNIO', 'BELA VISTA', 'PRAIA DO SUÁ',
       'SANTA HELENA', 'ITARARÉ', 'INHANGUETÁ', 'UNIVERSITÁRIO',
       'SÃO JOSÉ', 'REDENÇÃO', 'SANTA CLARA', 'CENTRO', 'PARQUE MOSCOSO',
       'DO MOSCOSO', 'SANTOS DUMONT', 'CARATOÍRA', 'ARIOVALDO FAVALESSA',
       'ILHA DO FRADE', 'GURIGICA', 'JOANA D´ARC', 'CONSOLAÇÃO',
       'SÃO BENEDITO', 'PRAIA DO CANTO', 'BOA VISTA', 'SANTA LÚCIA',
       'BARRO VERMELHO', 'ESTRELINHA', 'FORTE SÃO JOÃO', 'FONTE GRANDE',
       'MORADA DE CAMBURI', 'ENSEADA DO SUÁ', 'SANTOS REIS', 'PIEDADE',
       'JESUS DE NAZARETH', 'SANTA LUÍZA', 'SANTA TEREZA', 'CRUZAMENTO',
       'ILHA DO PRÍNCIPE', 'ROMÃO', 'ILHA DAS CAIEIRAS', 'COMDUSA',
       'SANTA CECÍLIA', 'VILA RUBIM', 'DE LOURDES', 'MONTE BELO',
       'DO QUADRO', 'DO CABRAL', 'HORTO', 'SEGURANÇA DO LAR',
       'ILHA DO BOI', 'FRADINHOS', 'NAZARETH', 'AEROPORTO',
       'ILHAS OCEÂNICAS DE TRINDADE', 'PARQUE INDUSTRIAL'], dtype=object)
```

[30]:
```
# re-inspect the dataset uing the info() function
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 106987 entries, 0 to 110526
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   PatientId       106987 non-null  float64
 1   AppointmentID   106987 non-null  int64
 2   Gender          106987 non-null  object
 3   ScheduledDay    106987 non-null  datetime64[ns, UTC]
 4   AppointmentDay  106987 non-null  datetime64[ns, UTC]
 5   Age             106987 non-null  int64
 6   Neighbourhood   106987 non-null  object
 7   Scholarship     106987 non-null  int64
 8   Hypertension    106987 non-null  int64
 9   Diabetes        106987 non-null  int64
 10  Alcoholism      106987 non-null  int64
 11  Handicap        106987 non-null  int64
 12  SMS_received    106987 non-null  int64
 13  No_show         106987 non-null  object
dtypes: datetime64[ns, UTC](2), float64(1), int64(8), object(3)
```

```
memory usage: 12.2+ MB
```

[31]:
```
# check the unique entries of the Scholarship column
df['Scholarship'].unique()
```

[31]:
```
array([0, 1], dtype=int64)
```

[32]:
```
# check the unique entries of the Hypertension column
df['Hypertension'].unique()
```

[32]:
```
array([1, 0], dtype=int64)
```

[33]:
```
# check the unique entries of the Diabetes column
df['Diabetes'].unique()
```

[33]:
```
array([0, 1], dtype=int64)
```

[34]:
```
# check the unique entries of the Alcoholism column
df['Alcoholism'].unique()
```

[34]:
```
array([0, 1], dtype=int64)
```

[35]:
```
# check the number of unique entries of the Handicap column
df['Handicap'].unique()
```

[35]:
```
array([0, 1, 2, 3, 4], dtype=int64)
```

[36]:
```
# check the unique entries of the SMS_received column
df['SMS_received'].unique()
```

[36]:
```
array([0, 1], dtype=int64)
```

[37]:
```
# check the unique entries of the No_show column
df['No_show'].unique()
```

[37]:
```
array(['No', 'Yes'], dtype=object)
```

2 Exploratory Data Analysis

In this section,I will explore the dataset, perform decriptive statistics and visualizations in oder to address the questions posed in the 'Questions' section.

[38]:
```
# descriptive statistics
df.describe()
```

[38]:
```
          PatientId  AppointmentID            Age    Scholarship  \
count  1.069870e+05   1.069870e+05  106987.000000  106987.000000
mean   1.472814e+14   5.675434e+06      38.316085       0.101031
```

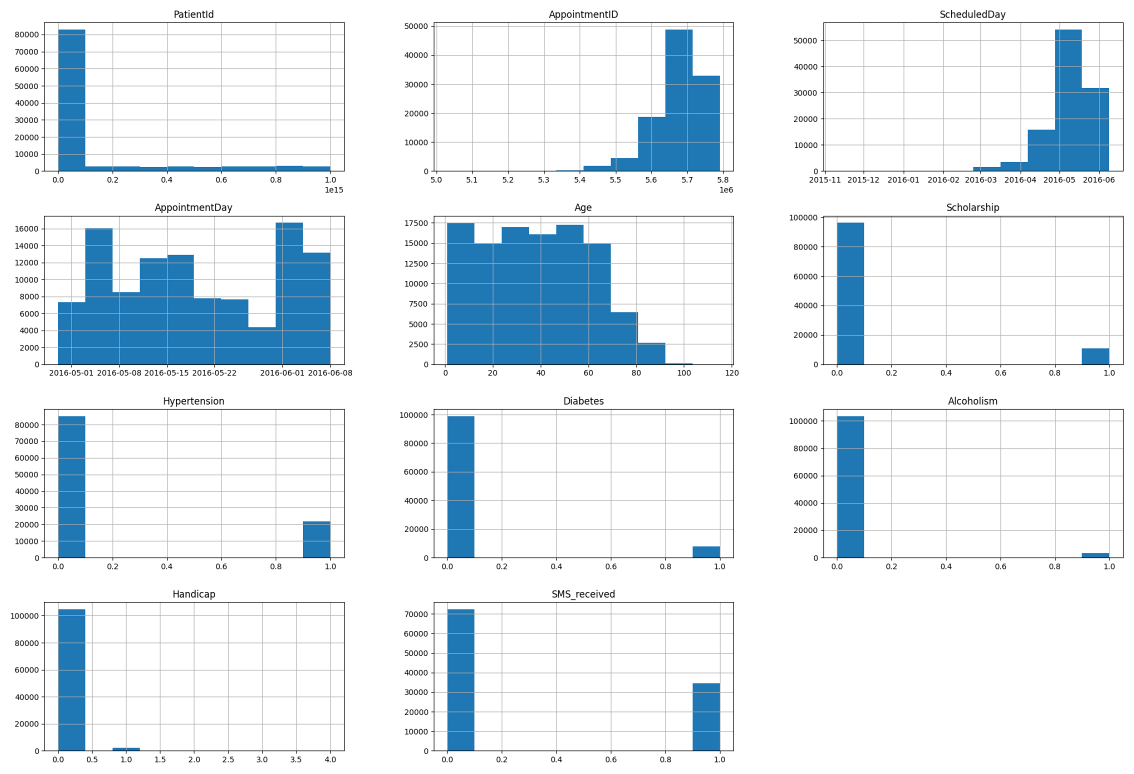
```
std    2.558267e+14  7.133274e+04     22.466214       0.301371
min    3.921784e+04  5.030230e+06      1.000000       0.000000
25%    4.173523e+12  5.640490e+06     19.000000       0.000000
50%    3.172463e+13  5.680744e+06     38.000000       0.000000
75%    9.433600e+13  5.725634e+06     56.000000       0.000000
max    9.999816e+14  5.790484e+06    115.000000       1.000000

        Hypertension       Diabetes     Alcoholism       Handicap  \
count  106987.000000  106987.000000  106987.000000  106987.000000
mean        0.203772       0.074243       0.031406       0.022975
std         0.402804       0.262167       0.174412       0.164115
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  106987.000000
mean        0.323264
std         0.467725
min         0.000000
25%         0.000000
50%         0.000000
75%         1.000000
max         1.000000
```

[39]:
```
#  histograms of the dataset

df.hist( figsize = (25, 17));
```



```
[40]: # a query to check the females in the dataset
      df.query('Gender == "F"')
```

```
[40]:           PatientId  AppointmentID Gender              ScheduledDay  \
      0       2.987250e+13        5642903      F 2016-04-29 18:38:08+00:00
      2       4.262962e+12        5642549      F 2016-04-29 16:19:04+00:00
      3       8.679512e+11        5642828      F 2016-04-29 17:29:31+00:00
      4       8.841186e+12        5642494      F 2016-04-29 16:07:23+00:00
      5       9.598513e+13        5626772      F 2016-04-27 08:36:51+00:00
      …                …            …      …                       …
      110522  2.572134e+12        5651768      F 2016-05-03 09:15:35+00:00
      110523  3.596266e+12        5650093      F 2016-05-03 07:27:33+00:00
      110524  1.557663e+13        5630692      F 2016-04-27 16:03:52+00:00
      110525  9.213493e+13        5630323      F 2016-04-27 15:09:23+00:00
      110526  3.775115e+14        5629448      F 2016-04-27 13:30:56+00:00

                         AppointmentDay  Age      Neighbourhood  Scholarship  \
      0       2016-04-29 00:00:00+00:00   62    JARDIM DA PENHA            0
      2       2016-04-29 00:00:00+00:00   62      MATA DA PRAIA            0
      3       2016-04-29 00:00:00+00:00    8  PONTAL DE CAMBURI            0
      4       2016-04-29 00:00:00+00:00   56    JARDIM DA PENHA            0
      5       2016-04-29 00:00:00+00:00   76          REPÚBLICA            0
      …                              …  …                 …          …
```

```
110522 2016-06-07 00:00:00+00:00   56        MARIA ORTIZ            0
110523 2016-06-07 00:00:00+00:00   51        MARIA ORTIZ            0
110524 2016-06-07 00:00:00+00:00   21        MARIA ORTIZ            0
110525 2016-06-07 00:00:00+00:00   38        MARIA ORTIZ            0
110526 2016-06-07 00:00:00+00:00   54        MARIA ORTIZ            0

        Hypertension  Diabetes  Alcoholism  Handicap  SMS_received No_show
0                  1         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
…                …       …         …        …          …      …
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

[70118 rows x 14 columns]
```

[41]:
```
# value counts of the SMS_received column
df.SMS_received.value_counts()
```

[41]:
```
0    72402
1    34585
Name: SMS_received, dtype: int64
```

from the above, 72402 people did not receive sms while 34585 did. Thus more people did not received sms

[42]:
```
# value counts of the No_show column
df.No_show.value_counts()
```

[42]:
```
No     85307
Yes    21680
Name: No_show, dtype: int64
```

[43]:
```
# a query  to check the data on those that missed or did not show up for their␣
↪appointment
df1 =df['No_show'] == 'Yes'
```

[44]:
```
# same operation as performed above
df.query('No_show =="Yes"')
```

[44]:
```
           PatientId  AppointmentID Gender              ScheduledDay  \
6       7.336882e+14        5630279      F 2016-04-27 15:05:12+00:00
7       3.449833e+12        5630575      F 2016-04-27 15:39:58+00:00
```

```
11      7.542951e+12        5620163      M 2016-04-26 08:44:12+00:00
17      1.479497e+13        5633460      F 2016-04-28 09:28:57+00:00
20      6.222575e+14        5626083      F 2016-04-27 07:51:14+00:00
…           …               …    …                        …
110484  5.133650e+14        5772155      F 2016-06-03 14:43:56+00:00
110492  6.456342e+14        5786741      M 2016-06-08 08:50:19+00:00
110496  8.544295e+13        5779046      F 2016-06-06 17:35:38+00:00
110515  6.456342e+14        5778621      M 2016-06-06 15:58:05+00:00
110516  6.923772e+13        5780205      F 2016-06-07 07:45:16+00:00

                  AppointmentDay  Age   Neighbourhood  Scholarship  \
6      2016-04-29 00:00:00+00:00   23      GOIABEIRAS            0
7      2016-04-29 00:00:00+00:00   39      GOIABEIRAS            0
11     2016-04-29 00:00:00+00:00   29  NOVA PALESTINA            0
17     2016-04-29 00:00:00+00:00   40       CONQUISTA            1
20     2016-04-29 00:00:00+00:00   30  NOVA PALESTINA            0
…                            …  …             …          …
110484 2016-06-07 00:00:00+00:00   45  BARRO VERMELHO            0
110492 2016-06-08 00:00:00+00:00   33     MARIA ORTIZ            0
110496 2016-06-08 00:00:00+00:00   37     MARIA ORTIZ            0
110515 2016-06-08 00:00:00+00:00   33     MARIA ORTIZ            0
110516 2016-06-08 00:00:00+00:00   37     MARIA ORTIZ            0

        Hypertension  Diabetes  Alcoholism  Handicap  SMS_received No_show
6                  0         0           0         0             0     Yes
7                  0         0           0         0             0     Yes
11                 0         0           0         0             1     Yes
17                 0         0           0         0             0     Yes
20                 0         0           0         0             0     Yes
…                …       …         …         …            …      …
110484             0         0           0         0             0     Yes
110492             1         0           0         0             0     Yes
110496             1         0           0         0             0     Yes
110515             1         0           0         0             0     Yes
110516             0         0           0         0             0     Yes

[21680 rows x 14 columns]
```

2.0.1 Questions

2.0.2 As stated in the 'Questions' section above, I will be addressing these questions

1.

- how many females and males are there in the dataset?
- how many males and females showed up for their appointment and what's the percentage ?
- how many males and females missed their appointment and what's the percentage?

2.how many females received sms and showed up as against those that did not receive any sms but still showed up?

3.

- what percentage of females are diabetic?
- are there female children who are diabetic? how many?
- how many diabetics and alcoholics showed up for the appointment and how many did not?

2.0.3 let's start addressing the questions one at a time

2.0.4 1.how many males and females :

- are there in the dataset
- showed up for their appointment and what's the percentage
- missed their appointment and what's the percentage

[45]:
```
# value count of the females in the dataset
females = (df['Gender'] == 'F').value_counts()
females
```

[45]:
```
True     70118
False    36869
Name: Gender, dtype: int64
```

From the output above, there are 70118 females and 36869 males. Obviously there are more females than males

2.0.5 how many females showed up for their appointment ?

'No' in the 'No_show' column actually means the person showed up, and 'Yes' otherwise

[46]:
```
#females that showed up for their appointment
females_and_show_up = (df['Gender'] == 'F') & (df['No_show'] == 'No')
```

[47]:
```
# get the number or the value count of the females that showed up
females_and_show_up.value_counts()
```

[47]:
```
True     55843
False    51144
dtype: int64
```

[48]:
```
# get the first output of the above operation.i.e the 'True' part of the output
females_and_show_up.value_counts()[0]
```

[48]:
```
55843
```

55843 females showed up for the appointment

2.0.6 percentage of females that showed up

```
[49]: # percentage of females that showed up
      # round to the nearest whole number

      x =females_and_show_up.value_counts()[0]
      y =females_and_show_up.value_counts()[1]
      z =females_and_show_up.value_counts()[0] + females_and_show_up.value_counts()[1]
      ((x / z) * 100).round()
```

```
[49]: 52.0
```

percentage of females that showed up is Approximately 52% of the total number

2.0.7 how many males showed up for their appointment ?

```
[50]: #males that showed up for their appointment
      males_and_show_up = (df['Gender'] == 'M') & (df['No_show'] == 'No')
```

```
[51]: # get the number or the value count of the males that showed up
      males_and_show_up.value_counts()
```

```
[51]: False    77523
      True     29464
      dtype: int64
```

```
[52]: # get the second output of the above operation.i.e the 'True' part of the output
      y = males_and_show_up.value_counts()[1]
      y
```

```
[52]: 29464
```

29464 males showed up for the appointment

2.0.8 percentage of males that showed up

```
[53]: # percentage of males that showed up
      # round to the nearest whole number
      x = males_and_show_up.value_counts()[0]
      y = males_and_show_up.value_counts()[1]
      z = males_and_show_up.value_counts()[0] + males_and_show_up.value_counts()[1]
      ((y / z) * 100).round()
```

```
[53]: 28.0
```

Approximately 28% of males showed up.

We can thus deduct that approximately 80% of the people(both males and females) showed up for their appointment and 20% of them didn't show up.

let's get the proportion of the males and females that didn't show up or missed their appointment in their respective percentages.

2.0.9 percentage of males that did not show up

[54]:
```python
# percentage of males that did not show up
# and the value counts
males_and_no_show  = (df['Gender'] == 'M') & (df['No_show'] == 'Yes')
males_and_no_show.value_counts()
```

[54]:
```
False    99582
True      7405
dtype: int64
```

[55]:
```python
# get the second output of the above operation.i.e the 'True' part of the output
y =males_and_no_show.value_counts()[1]
y
```

[55]:
```
7405
```

7405 males did not show up for the appointment

[56]:
```python
# percentage of males that did not show up

x =males_and_no_show.value_counts()[0]
y =males_and_no_show.value_counts()[1]
z =males_and_no_show.value_counts()[0] + males_and_no_show.value_counts()[1]
((y / z) * 100).round()
```

[56]:
```
7.0
```

7% of the males did not showed up

2.0.10 percentage of females that did not show up

[57]:
```python
# females that did not show up

females_and_no_show = (df['Gender'] == 'F') & (df['No_show'] == 'Yes')
females_and_no_show.value_counts()
```

[57]:
```
False    92712
True     14275
dtype: int64
```

[58]:
```python
# get the second line of the output above
females_and_no_show.value_counts()[1]
```

[58]:
```
14275
```

14275 females did not show up

[59]:
```
# percentage of females that did not showed up

females_and_no_show = (df['Gender'] == 'F') & (df['No_show'] == 'Yes')
females_and_no_show.value_counts()

x =females_and_no_show.value_counts()[0]
y =females_and_no_show.value_counts()[1]
z =females_and_no_show.value_counts()[0] + females_and_no_show.value_counts()[1]
((y / z) * 100).round()
```

[59]:
```
13.0
```

13% of the females did not show up

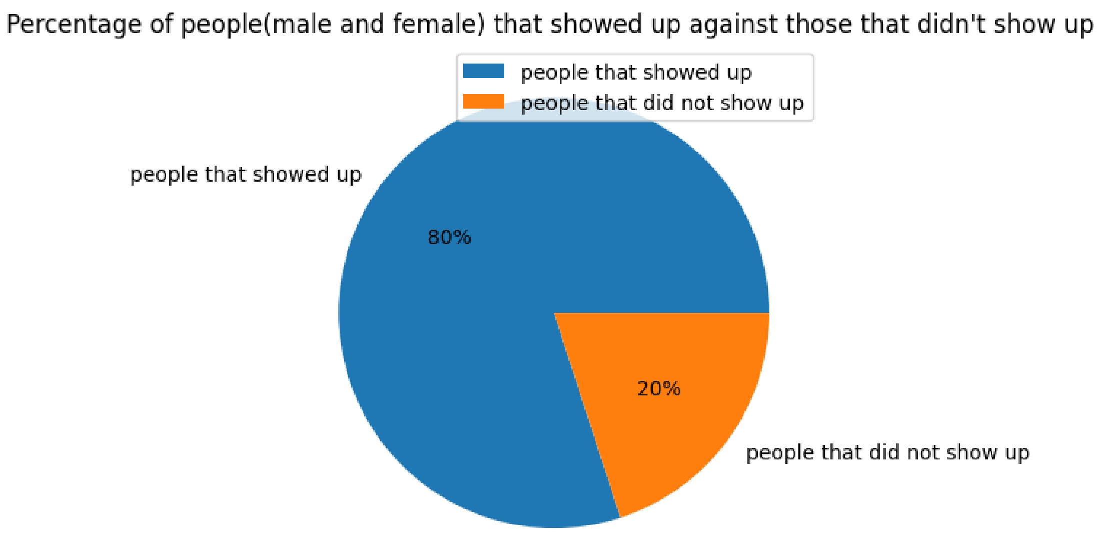
2.0.11 We can confidently conclude that among the people that didn't show up, approximately 7% of them are males and 13% of them are females

2.0.12 let's plot the percentages on a pie chart:

- 52% females showed up
- 28% males showed up
- 13% females did not show up
- 7% males did not show up

2.0.13 Percentage of people(male and female) that showed up against the percentage of the people(male and female) that did not show up on a pie chart

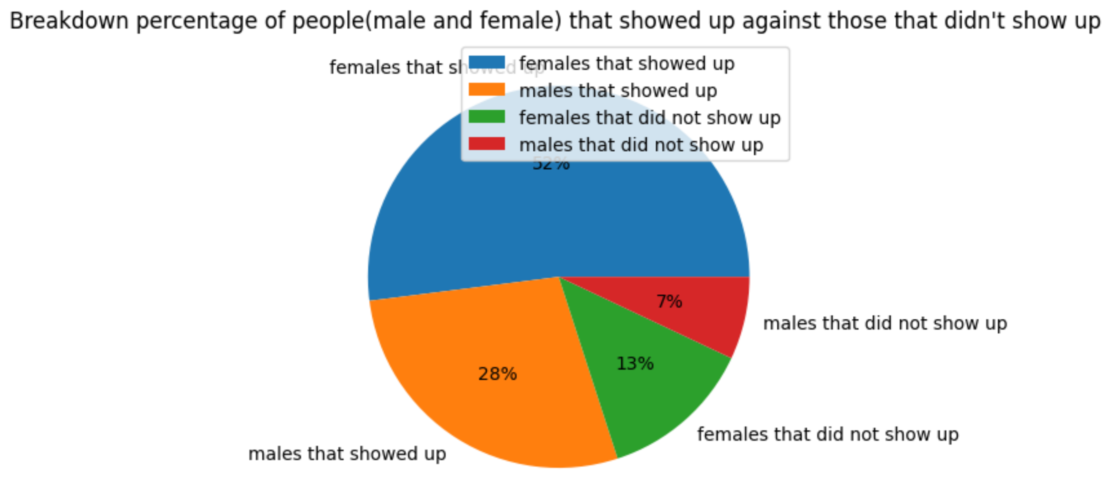
[60]:
```
# 80% of the total number showed up
# 20% didn't show up
# let's put these values in a numpy array but without the percentage signs
a =np.array([80, 20])
labels =['people that showed up','people that did not show up']
plt.pie(a,labels = labels, autopct = '%1.0f%%');
plt.title('Percentage of people(male and female) that showed up against those␣
  ↪that didn\'t show up ')
plt.legend();
```



Clearly from the pie chart above, a large number of the people, both males and females showed up for their appointment as opposed to the very few that missed their appointment

2.0.14 Percentage of males and females that showed up against the percentage of the males and females that did not show up on a pie chart

[61]:
```
# 52% females showed up
# 28% males showed up
# 13% females did not show up
# 7% males did not show up
# let's put these values in a numpy array but without the percentage signs
a =np.array([52, 28, 13, 7])
labels =['females that showed up','males that showed up', 'females that did not␣
↪show up','males that did not show up']
plt.pie(a,labels = labels, autopct = '%1.0f%%');
plt.title('Breakdown percentage of people(male and female) that showed up␣
↪against those that didn\'t show up ')
plt.legend();
```



from the pie charts above : a large number of the people, both males and females showed up for their appointment which makes up 80% of the people. Among the 80%, 52% of them are females and 28% of them are males as opposed to the very few that missed their appointment making up 20% of the people which 13% of them are females and 7% being males.

2.0.15 2.how many females received sms and showed up as against those that did not receive any sms but still showed up ?

[62]:
```
# females that receive sms and showed up
females_received_show_up = (df['Gender'] == 'F') & (df['SMS_received'] == 1)  &␣
 ↪(df['No_show'] == 'No')
```

[63]:
```
# dataframe  of the first five females that receive sms and showed up
df.loc[females_received_show_up].head()
```

[63]:
```
       PatientId  AppointmentID Gender              ScheduledDay  \
15  9.994839e+10        5620206      F 2016-04-26 08:47:27+00:00
18  1.713538e+13        5621836      F 2016-04-26 10:54:18+00:00
33  7.653517e+12        5616921      F 2016-04-25 15:01:04+00:00
62  3.647762e+13        5614045      F 2016-04-25 10:01:13+00:00
68  5.434176e+12        5552915      F 2016-04-06 18:00:29+00:00

               AppointmentDay  Age    Neighbourhood  Scholarship  Hypertension  \
15 2016-04-29 00:00:00+00:00   15   NOVA PALESTINA            0             0
18 2016-04-29 00:00:00+00:00   30   NOVA PALESTINA            1             0
33 2016-04-29 00:00:00+00:00   38    SÃO CRISTÓVÃO            1             0
62 2016-04-29 00:00:00+00:00    3        CONQUISTA            1             0
68 2016-04-29 00:00:00+00:00   69  JARDIM DA PENHA            0             1
```

```
    Diabetes  Alcoholism  Handicap  SMS_received No_show
15         0           0         0             1      No
18         0           0         0             1      No
33         0           0         0             1      No
62         0           0         0             1      No
68         0           0         0             1      No
```

[64]:
```
# value counts
females_received_show_up.value_counts()
```

[64]:
```
False    89840
True     17147
dtype: int64
```

[65]:
```
# get the second line of the output above and assign it to f_r
f_r = females_received_show_up.value_counts()[1]
f_r
```

[65]:
```
17147
```

17147 females received sms and showed up

2.0.16 how many females did not receive sms but showed up?

[66]:
```
# females that did not receive sms but showed up
females_received_no_show = (df['Gender'] == 'F') & (df['SMS_received'] == 0)  &␣
 ↪(df['No_show'] == 'No')
```

[67]:
```
# dataframe  of the first five females that did not receive sms but showed up
df.loc[females_received_no_show].head()
```

[67]:
```
      PatientId  AppointmentID Gender              ScheduledDay  \
0  2.987250e+13        5642903      F 2016-04-29 18:38:08+00:00
2  4.262962e+12        5642549      F 2016-04-29 16:19:04+00:00
3  8.679512e+11        5642828      F 2016-04-29 17:29:31+00:00
4  8.841186e+12        5642494      F 2016-04-29 16:07:23+00:00
5  9.598513e+13        5626772      F 2016-04-27 08:36:51+00:00

             AppointmentDay  Age      Neighbourhood  Scholarship  \
0 2016-04-29 00:00:00+00:00   62    JARDIM DA PENHA            0
2 2016-04-29 00:00:00+00:00   62      MATA DA PRAIA            0
3 2016-04-29 00:00:00+00:00    8  PONTAL DE CAMBURI            0
4 2016-04-29 00:00:00+00:00   56    JARDIM DA PENHA            0
5 2016-04-29 00:00:00+00:00   76           REPÚBLICA            0

   Hypertension  Diabetes  Alcoholism  Handicap  SMS_received No_show
0             1         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
```

[ ]:

```

```

[68]:

```
# value counts of female who received sms but did not show up for their␣
 ↪appointment
females_received_no_show.value_counts()
```

[68]:

```
False    68291
True     38696
dtype: int64
```

[69]:

```
# get the second line of the output above and assign it to f_nr
f_nr = females_received_no_show.value_counts()[1]
f_nr
```

[69]:

```
38696
```

38696 females did not receive sms but still showed up

2.0.17 plot the values of the females that received sms and showed up against those that did not receive sms but still showed up on a bar chart

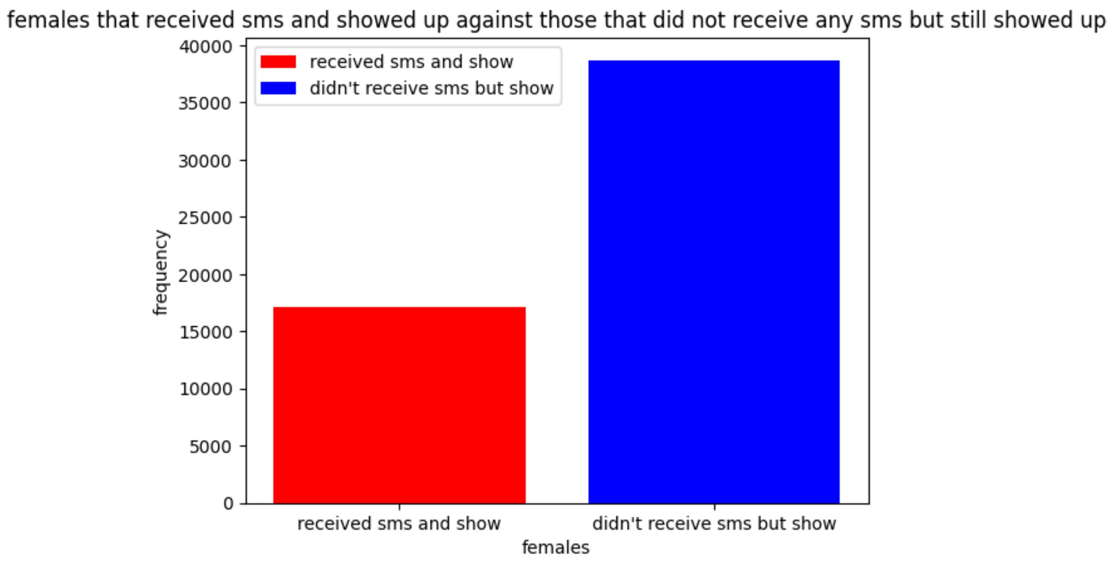
[70]:

```
#values of the females that received sms and showed up
#against those that did not receive sms but still showed up on a bar chart

# funtion to receive a list of values
# which will be used in a numpy array for plotting
def fun( arr = []):
    return (arr)

# labels = np.array(['received sms and show', 'didn\'t receive sms but show'])
labels = ['received sms and show', 'didn\'t receive sms but show']
freq = np.array(fun([f_r,f_nr]))
# freq = np.array([f_r,f_nr])
color = ['red', 'blue']


plt.bar( labels,freq, color = color , label = labels)
plt.xlabel('females');
plt.ylabel('frequency');
plt.title('females that received sms and showed up against those that did not␣
 ↪receive any sms but still showed up');
plt.legend();
```



from the plot above, we can clearly see that more females showed up for their appointment despite not receiving any sms and that is nearly twice the size of the females that received the sms and showed up

2.0.18 3.

- what percentage of females are diabetic ?
- are there children who are diabetic ?
- how many diabetic and alcohollic showed up for the appointment ?
-

2.0.19 percentage of females who are diabetic

females who are diabetic

[71]:
```
# females who are diabetic
# get the value count of diabetic females
females_and_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 1)
females_and_diabetic.value_counts()
```

[71]:
```
False    101381
True       5606
dtype: int64
```

[72]:
```
# get the True part of the output above
# to get the number of diabetic females
f_and_d = females_and_diabetic.value_counts()[1]
f_and_d
```

```
[72]: 5606
```

5606 females are diabetic

non-diabetic females

```
[73]: # non-diabetic females
      # get the value count of  non-diabetic females
      females_and_not_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 0)
      females_and_not_diabetic.value_counts()
```

```
[73]: True     64512
      False    42475
      dtype: int64
```

```
[74]: # get the True part of the output above
      # to get the number of diabetic females
      f_not_d =females_and_not_diabetic.value_counts()[0]
      f_not_d
```

```
[74]: 64512
```

64512 females are not diabetic

the total number of females

```
[75]: # get the value count of the total number of females
      # and assign it to females_
      females_  = (df['Gender'] == 'F').value_counts()
      females_
```

```
[75]: True     70118
      False    36869
      Name: Gender, dtype: int64
```

```
[76]: # get the True part of the output above
      # to get the total number of females
      females_[0]
```

```
[76]: 70118
```

70118 are females

percentage of diabetic females

```
[77]: # percentage of diabetic females
      # round to the nearest whole number
      ((f_and_d / females_[0]) * 100).round()
```

```
[77]: 8.0
```

of 70118 females, 8% of them are diabetic

percentage of non-diabetic females

[78]:
```
# percentage of non-diabetic females
# round to the nearest whole number
((f_not_d / females_[0]) * 100).round()
```

[78]:
```
92.0
```

92% of the females are not diabetic.i.e the majority

-

2.0.20 are there female children who are diabetic? how many?

female children who are diabetic

[79]:
```
# female children who are diabetic
child_females_and_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 1) &␣
 ↪(df['Age'] < 18)
```

[80]:
```
# count of female children who are diabetic
child_females_and_diabetic.value_counts()
```

[80]:
```
False    106949
True         38
dtype: int64
```

[81]:
```
# get the True part of the output above
# to get the number of diabetic female children
# and assign it to f_d_c
f_d_c = child_females_and_diabetic.value_counts()[1]
f_d_c
```

[81]:
```
38
```

from the above, 38 female children are diabetic

female adults who are diabetic

[82]:
```
# female adults who are diabetic
adult_females_and_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 1) &␣
 ↪(df['Age'] >= 18)
```

[83]:
```
# count of female adults who are diabetic
adult_females_and_diabetic.value_counts()
```

[83]:
```
False    101419
True       5568
dtype: int64
```

[84]:
```
# get the True part of the output above
# to get the number of diabetic adults
# # and assign it to f_d_a
```

```
f_d_a =adult_females_and_diabetic.value_counts()[1]
f_d_a
```

```
[84]: 5568
```

5568 female adults are diabetic

female adults who are not diabetic

```
[85]: # female adults who are not diabetic
adult_females_and_not_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 0)␣
 ↪& (df['Age'] >= 18)
```

```
[86]: # count of female adults who are not diabetic
adult_females_and_not_diabetic.value_counts()
```

```
[86]: False    54614
True     52373
dtype: int64
```

```
[87]: # get the True part of the output above
# to get the number of non-diabetic adults(female)
# and assign it to a_f_n_d
a_f_n_d = adult_females_and_not_diabetic.value_counts()[1]
a_f_n_d
```

```
[87]: 52373
```

52373 are non-diabetic adults (female)

children who are not diabetic

```
[88]: # female children who are not diabetic
child_females_and_not_diabetic = (df['Gender'] == 'F') & (df['Diabetes'] == 0)␣
 ↪& (df['Age'] < 18)
```

```
[89]: # count of female children who are not diabetic
child_females_and_not_diabetic.value_counts()
```

```
[89]: False    94848
True     12139
dtype: int64
```

```
[90]: # get the True part of the output above
# to get the number of diabetic adults
# and assign it to c_f_n_d
c_f_n_d = child_females_and_not_diabetic.value_counts()[1]
c_f_n_d
```

```
[90]: 12139
```

12139 female children are not diabetic

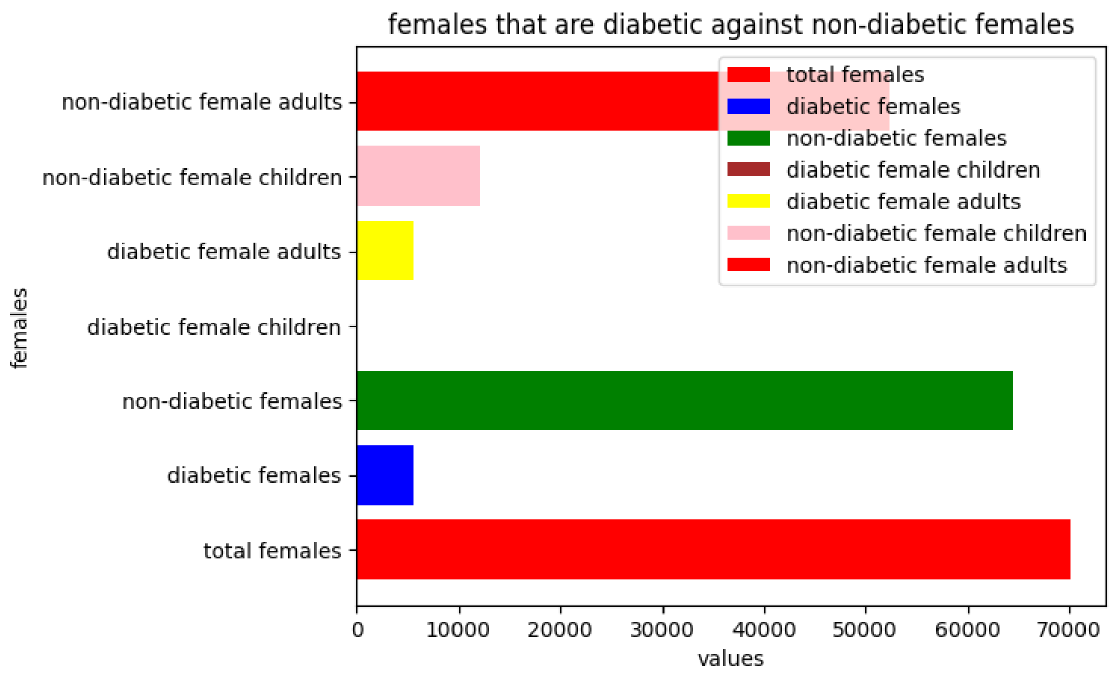
2.0.21 plot of the respective values of female diabetics in their respective categories (diabetic females, non-diabetic females , diabetic female children, diabetic female adults, non-diabetic female children,non-diabetic female adults)

[91]:
```
# plot of the respective values of female diabetics in their respective␣
 ↪categories
#(diabetic females, non-diabetic females , diabetic female children, diabetic␣
 ↪female adults,
#non-diabetic female children,non-diabetic female adults)

# funtion to receive a list of values
# which will be used in a numpy array for plotting
def fun( arr = []):
    return (arr)

labels = np.array(['total females', 'diabetic females','non-diabetic females',␣
 ↪'diabetic female children',
                  'diabetic female adults', 'non-diabetic female␣
 ↪children','non-diabetic female adults'])
freq = np.array(fun([females_[0], f_and_d , f_not_d , f_d_c , f_d_a, c_f_n_d,␣
 ↪a_f_n_d]))
color = ['red', 'blue', 'green', 'brown', 'yellow', 'pink']

plt.barh(labels, freq, color = color,label = labels);
plt.xlabel('values')
plt.ylabel('females')
plt.title('females that are diabetic against non-diabetic females');
plt.legend();
```



Out of about 70 thousand females, a large number of them are non-diabetic, which a large proportion of them are adults and very few being children. There more non-diabetic females as compared to the diabetic females. NB: from the bar chart above, we can see that the 'diabetic female children' bar isn't showing, this is because it is too small.

```
[ ]:
```

-

2.0.22 how many diabetics and alcoholics showed up for the appointment and how many did not?

diabetics and alcoholics patients that showed up

```
[92]: # diabetics and alcoholics patients that showed up

diabetic_and_alcoholic_show = (df['Diabetes'] == 1) & (df['Alcoholism'] == 1 )␣
 ↪& (df['No_show'] == 'No')
# diabetic_and_alcoholic.value_counts()
```

```
[93]: # value counts of  diabetics and alcoholics patients that showed up
diabetic_and_alcoholic_show.value_counts()
```

```
[93]: False    106714
True        273
dtype: int64
```

```
[94]: # get the second line of the output above
      # and assign to dia_and_alco_show
      dia_and_alco_show = diabetic_and_alcoholic_show.value_counts()[1]
      dia_and_alco_show
```

```
[94]: 273
```

273 diabetic and alcoholic patients showed up

diabetic and alcoholic patients that did not show up

```
[95]: # diabetic and alcoholic patients that did not show up

      diabetic_and_alcoholic_no_show = (df['Diabetes'] == 1) & (df['Alcoholism'] == 1␣
       ↪) & (df['No_show'] == 'Yes')
      diabetic_and_alcoholic_no_show.value_counts()
```

```
[95]: False    106928
      True         59
      dtype: int64
```

```
[96]: # value diabetic and alcoholic patients that did not show up
      # and assign to dia_and_alco_no_show
      dia_and_alco_no_show = diabetic_and_alcoholic_no_show.value_counts()[1]
      dia_and_alco_no_show
```

```
[96]: 59
```

59 diabetic and alcoholic patients did not show up

total number of diabetics and alcoholics

```
[97]: # total number of diabetics and alcoholics
      # and the value counts
      diabetic_and_alcoholic = (df['Diabetes'] == 1) & (df['Alcoholism'] == 1 )
      diabetic_and_alcoholic.value_counts()
```

```
[97]: False    106655
      True        332
      dtype: int64
```

```
[98]: # get the second line of the output above and assign it to total
      total = diabetic_and_alcoholic.value_counts()[1]
      total
```

```
[98]: 332
```

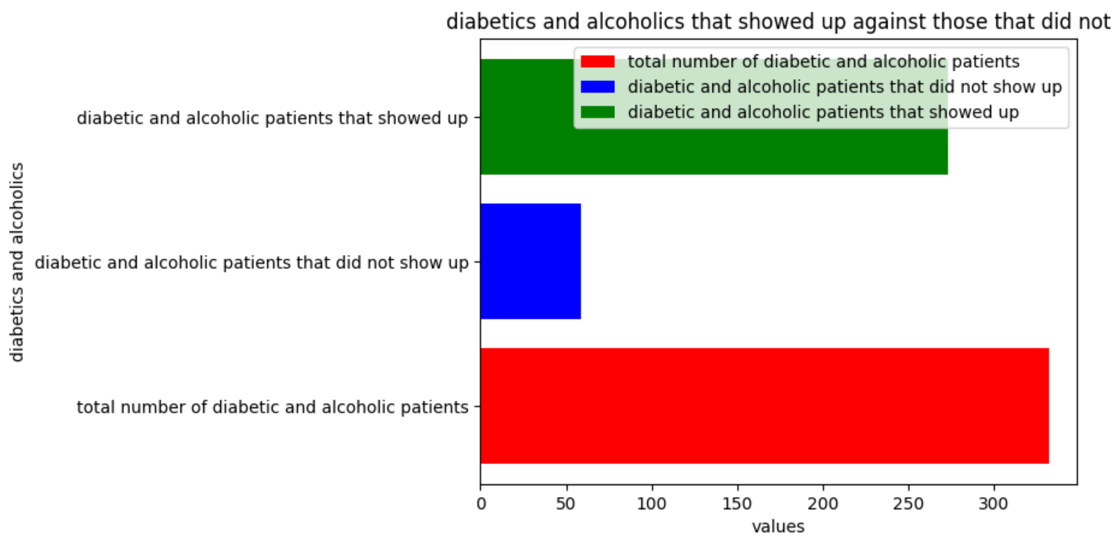
332 are diabetics and alcoholics

2.0.23 plot of the respective values of the number of diabetic and alcoholic patients that did not show up against those that did on a bar chart

[99]:
```
# plot of the respective values of the number of diabetic and alcoholic␣
↪patients that did not show up
# against those that did on a bar chart

labels = np.array(['total number of diabetic and alcoholic patients','diabetic␣
↪and alcoholic patients that did not show up',
                  'diabetic and alcoholic patients that showed up'])
freq = np.array([total, dia_and_alco_no_show , dia_and_alco_show  ])
color = ['red', 'blue', 'green']

plt.barh(labels, freq, color = color, label = labels);
plt.xlabel('values')
plt.ylabel('diabetics and alcoholics')
plt.title('diabetics and alcoholics that showed up against those that did not');
plt.legend();
```



2.0.24 Out of 332 diabetics and alcoholics, roughly about 82% of them showed up for their appointment. i.e the majority while roughly 18% of them did not show up.

3 Conclusion

A large number of the people are females and thus much of my analyses were conducted on the female gender where I took a look at those who are diabetic or not among which some were children. Also I was interested in those that received sms and showed up against those that didn't receive any sms but still showed up(nothing conclusive can be deducted out of that, as to whether the sms played a major part in they showing up or not as other factors haved to be considered before any solid assumption/conclusion can be made)

3.1 Limitation

Ideally, I did not really encounter any limitations per se, per my analyses. But then again, per my analyses the sms received or not is not enough predictor to determine whether or not a patient will show up for their appointment as other factors have to be considered too.