udacity project1_Asmaa Mostafa

September 16, 2020

1 Project : No-show appointments

1.1 Table of Contents

1.2 Introduction

In this project I worked on Medical Appointment No Shows dataset. The dataset contains some attributes for patients including if the patients meet their appointments or not . The analysis target is to provide insights related to the factors that may influance the absence or presence of patient at appointment date depending on the data included in the dataset.

Suggested questions needed to be answered :

- What is the percentage of missed and attended appointments?
- Which gender did meet its appointment dates ?
- What is the distribution of days of the week according to Scheduled Day?
- Does the day of the week impact appointment showup?
- What factors are important to predict if a patient will show up their appointment?
- How does waiting days number affect patient showup appointment?

1.3 Data Wrangling

In this section I will explore the data to handle it.

```
In [1]: # Load data to a dataframe

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

        df= pd.read_csv(r"D:\udacity_project\noshowappointments-kagglev2-may-2016.csv")
        df.head()
```

```
Out[1]:       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 [2]: # Numbers of rows and columns
        df.shape
```

```
Out[2]: (110527, 14)
```

There are 110527 rows and 14 columns in the dataset

```
In [3]: # Get datatypes of columns
        df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   PatientId       110527 non-null  float64
 1   AppointmentID   110527 non-null  int64  
 2   Gender          110527 non-null  object 
 3   ScheduledDay    110527 non-null  object 
 4   AppointmentDay  110527 non-null  object 
 5   Age             110527 non-null  int64  
 6   Neighbourhood   110527 non-null  object 
 7   Scholarship     110527 non-null  int64  
 8   Hipertension    110527 non-null  int64  
 9   Diabetes        110527 non-null  int64  
 10  Alcoholism      110527 non-null  int64  
 11  Handcap         110527 non-null  int64  
 12  SMS_received    110527 non-null  int64  
 13  No-show         110527 non-null  object
dtypes: float64(1), int64(8), object(5)
memory usage: 11.8+ MB
```

```
In [4]: # Get type of objects

        for col in df.columns:
            print(col, type(df[col].iloc[0]))
```

```
PatientId <class 'numpy.float64'>
AppointmentID <class 'numpy.int64'>
Gender <class 'str'>
ScheduledDay <class 'str'>
AppointmentDay <class 'str'>
Age <class 'numpy.int64'>
Neighbourhood <class 'str'>
Scholarship <class 'numpy.int64'>
Hipertension <class 'numpy.int64'>
Diabetes <class 'numpy.int64'>
Alcoholism <class 'numpy.int64'>
Handcap <class 'numpy.int64'>
SMS_received <class 'numpy.int64'>
No-show <class 'str'>
```

```
In [5]: # check duplicated rows numbers
        df.duplicated().sum()

Out[5]: 0
```

```
In [6]: # check Null values
        df.isnull().sum(axis = 1)

Out[6]: 0         0
        1         0
        2         0
        3         0
        4         0
                 ..
        110522    0
        110523    0
        110524    0
        110525    0
        110526    0
        Length: 110527, dtype: int64
```

Data Cleaning This step including :

- Converting PatientId column data type from float to int.
- Converting the data type of The ScheduledDay and AppointmentDay columns from string to datetime.
- Naming convention for database attributes .
- Check if age column doesn't contain negative value.
- Check unique values for attributes.
- Mapping no-show colmn values No to 0 and Yes to 1 to facilitate analysis.
- Create new column waiting_days to calculate the number of waiting days between scheduled day and appointment day.
- Create new columns appointment_Day_name and Schedule_Day_name to know the distribution for days of week.

```
In [7]: # Check  duplicated records

        df.duplicated().sum()

Out[7]: 0

In [8]: # Convert  PatientId column data type from float to int
        df['PatientId'] = df['PatientId'].astype('int64')
        # check change
        df['PatientId'].dtypes

Out[8]: dtype('int64')

In [9]: #Fix ScheduledDay and AppointmentDay columns type to datetime


        df['ScheduledDay'] =pd.to_datetime(df['ScheduledDay']).dt.date.astype('datetime64[ns]')
        df['AppointmentDay'] = pd.to_datetime(df['AppointmentDay']).dt.date.astype('datetime64[n

        df['ScheduledDay'].dtypes
        df.head()

Out[9]:          PatientId  AppointmentID Gender ScheduledDay AppointmentDay  Age  \
        0   29872499824296        5642903      F   2016-04-29     2016-04-29   62
        1  558997776694438        5642503      M   2016-04-29     2016-04-29   56
        2    4262962299951        5642549      F   2016-04-29     2016-04-29   62
        3     867951213174        5642828      F   2016-04-29     2016-04-29    8
        4    8841186448183        5642494      F   2016-04-29     2016-04-29   56

               Neighbourhood  Scholarship  Hipertension  Diabetes  Alcoholism  \
        0    JARDIM DA PENHA            0             1         0           0
        1    JARDIM DA PENHA            0             0         0           0
```

```
2      MATA DA PRAIA            0             0         0            0
3  PONTAL DE CAMBURI            0             0         0            0
4    JARDIM DA PENHA            0             1         1            0

   Handcap  SMS_received No-show
0        0             0      No
1        0             0      No
2        0             0      No
3        0             0      No
4        0             0      No
```

```
In [10]: # Check min and max values for ScheduledDay column and AppointmentDay column

         print('Start scheduling  on : {}.'.format(df['ScheduledDay'].min()))
         print('End scheduling  on : {}.'.format(df['ScheduledDay'].max()))


         print('Start appointments on : {}.'.format(df['AppointmentDay'].min()))
         print('End appointments on : {}.'.format(df['AppointmentDay'].max()))
```

```
Start scheduling  on : 2015-11-10 00:00:00.
End scheduling  on : 2016-06-08 00:00:00.
Start appointments on : 2016-04-29 00:00:00.
End appointments on : 2016-06-08 00:00:00.
```

```
In [11]: # Rename all columns labels to replace spaces with underscores and convert everything t
         # replace spaces with underscores and lowercase labels for 2018 dataset
         df.columns = ['patient_id', 'appointment_id', 'gender', 'scheduled_day',
                       'appointment_day', 'age', 'neighbourhood', 'scholarship', 'hypertension',
                       'diabetes', 'alcoholism', 'handicap', 'sms_received', 'no_show']
         df.columns
         # confirm changes
         df.head(1)
```

```
Out[11]:        patient_id  appointment_id gender scheduled_day appointment_day  age  \
         0  29872499824296         5642903      F    2016-04-29      2016-04-29   62

              neighbourhood  scholarship  hypertension  diabetes  alcoholism  handicap  \
         0  JARDIM DA PENHA            0             1         0           0         0

            sms_received no_show
         0             0      No
```

```
In [12]: # check if age not less than 0
         df[df['age']< 0].shape

         # Drop age rows lessthan 0
```

```
        dropId=df[df['age']<0].index
        df.drop(dropId, axis=0 ,inplace = True)

        # check number of rows
        df.shape
```

```
Out[12]: (110526, 14)
```

```
In [13]: # check gender value

         df['gender'].unique()

         # Getting Numbers of females and males
         df.groupby('gender').groups
```

```
Out[13]: {'F': Int64Index([     0,      2,      3,      4,      5,      6,      7,      8,
                      9,     10,
                    ...
                    110517, 110518, 110519, 110520, 110521, 110522, 110523, 110524,
                    110525, 110526],
                   dtype='int64', length=71839),
          'M': Int64Index([     1,     11,     13,     16,     22,     25,     28,     31,
                     32,     35,
                    ...
                    110490, 110492, 110493, 110495, 110497, 110501, 110506, 110509,
                    110513, 110515],
                   dtype='int64', length=38687)}
```

In this analysis we need to calculate the waiting time between schedule date and appointment date.

```
In [14]: # Create new column waiting_days
         df['waiting_days'] = (df['appointment_day'] - df['scheduled_day']).dt.days
         df.head (20)
```

```
Out[14]:          patient_id  appointment_id gender scheduled_day appointment_day  age  \
         0    29872499824296         5642903      F    2016-04-29      2016-04-29   62
         1   558997776694438         5642503      M    2016-04-29      2016-04-29   56
         2     4262962299951         5642549      F    2016-04-29      2016-04-29   62
         3      867951213174         5642828      F    2016-04-29      2016-04-29    8
         4     8841186448183         5642494      F    2016-04-29      2016-04-29   56
         5    95985133231274         5626772      F    2016-04-27      2016-04-29   76
         6   733688164476661         5630279      F    2016-04-27      2016-04-29   23
         7     3449833394123         5630575      F    2016-04-27      2016-04-29   39
         8    56394729949972         5638447      F    2016-04-29      2016-04-29   21
         9    78124564369297         5629123      F    2016-04-27      2016-04-29   19
         10  734536231958495         5630213      F    2016-04-27      2016-04-29   30
         11    7542951368435         5620163      M    2016-04-26      2016-04-29   29
         12  566654781423437         5634718      F    2016-04-28      2016-04-29   22
```

```
13  911394617215919         5636249      M    2016-04-28       2016-04-29   28
14   99884723334928         5633951      F    2016-04-28       2016-04-29   54
15      99948393975         5620206      F    2016-04-26       2016-04-29   15
16   84574392942817         5633121      M    2016-04-28       2016-04-29   50
17   14794966191172         5633460      F    2016-04-28       2016-04-29   40
18   17135378245248         5621836      F    2016-04-26       2016-04-29   30
19    7223289184215         5640433      F    2016-04-29       2016-04-29   46

        neighbourhood  scholarship  hypertension  diabetes  alcoholism  \
0     JARDIM DA PENHA            0             1         0           0
1     JARDIM DA PENHA            0             0         0           0
2       MATA DA PRAIA            0             0         0           0
3   PONTAL DE CAMBURI            0             0         0           0
4     JARDIM DA PENHA            0             1         1           0
5           REPÚBLICA            0             1         0           0
6          GOIABEIRAS            0             0         0           0
7          GOIABEIRAS            0             0         0           0
8          ANDORINHAS            0             0         0           0
9           CONQUISTA            0             0         0           0
10     NOVA PALESTINA            0             0         0           0
11     NOVA PALESTINA            0             0         0           0
12     NOVA PALESTINA            1             0         0           0
13     NOVA PALESTINA            0             0         0           0
14     NOVA PALESTINA            0             0         0           0
15     NOVA PALESTINA            0             0         0           0
16     NOVA PALESTINA            0             0         0           0
17          CONQUISTA            1             0         0           0
18     NOVA PALESTINA            1             0         0           0
19           DA PENHA            0             0         0           0

    handicap  sms_received no_show  waiting_days
0          0             0      No             0
1          0             0      No             0
2          0             0      No             0
3          0             0      No             0
4          0             0      No             0
5          0             0      No             2
6          0             0     Yes             2
7          0             0     Yes             2
8          0             0      No             0
9          0             0      No             2
10         0             0      No             2
11         0             1     Yes             3
12         0             0      No             1
13         0             0      No             1
14         0             0      No             1
15         0             1      No             3
16         0             0      No             1
```

```
17          0              0     Yes               1
18          0              1      No               3
19          0              0      No               0
```

Waiting days values can not be less than 0 . As appointment day can not be before scheduled day so i will drop rows that contain waiting days less than 0.

```
In [15]: #check waiting day value
         df[(df.waiting_days < 0)].shape

         # Drop waiting day value less than 0  records

         dropId=df[df['waiting_days']<0].index
         df.drop(dropId, axis=0 ,inplace = True)
```

I need to know days of week and its affect on appointment show up .

```
In [16]: # Create scheduled_day_name and appointment_day_name columns

         df['scheduled_day_name'] = df[['scheduled_day']].apply(lambda x: dt.datetime.strftime(x
         df['appointment_day_name'] = df[['appointment_day']].apply(lambda x: dt.datetime.strfti

         # Check the values
         df['scheduled_day_name'].value_counts()

Out[16]: Tuesday      26167
         Wednesday    24259
         Monday       23084
         Friday       18915
         Thursday     18072
         Saturday        24
         Name: scheduled_day_name, dtype: int64

In [17]: # Map no_show values from Yes to 1 and No to 0
         #inplace=True
         df['no_show'].replace("No", 0 ,inplace=True)
         df['no_show'].replace("Yes", 1,inplace=True)

In [18]: #check sms_received value
         df.sms_received.unique()

         df['sms_received'].value_counts()

Out[18]: 0    75039
         1    35482
         Name: sms_received, dtype: int64

In [19]: #check handicap value
         df.handicap.unique()

         df['handicap'].value_counts()
```

```
Out[19]: 0    108282
         1      2040
         2       183
         3        13
         4         3
         Name: handicap, dtype: int64
```

```
In [20]: # Check alcoholism  Values
         df.alcoholism.unique()

         df['alcoholism'].value_counts()
```

```
Out[20]: 0    107161
         1      3360
         Name: alcoholism, dtype: int64
```

```
In [21]: # Check diabetes  Values
         df.diabetes.unique()

         df['diabetes'].value_counts()
```

```
Out[21]: 0    102578
         1      7943
         Name: diabetes, dtype: int64
```

```
In [22]: # Check hypertension  Values
         df.hypertension.unique()

         df['hypertension'].value_counts()
```

```
Out[22]: 0    88720
         1    21801
         Name: hypertension, dtype: int64
```

```
In [23]: # Check hypertension  Values
         df.scholarship.unique()

         df['scholarship'].value_counts()
```

```
Out[23]: 0    99660
         1    10861
         Name: scholarship, dtype: int64
```

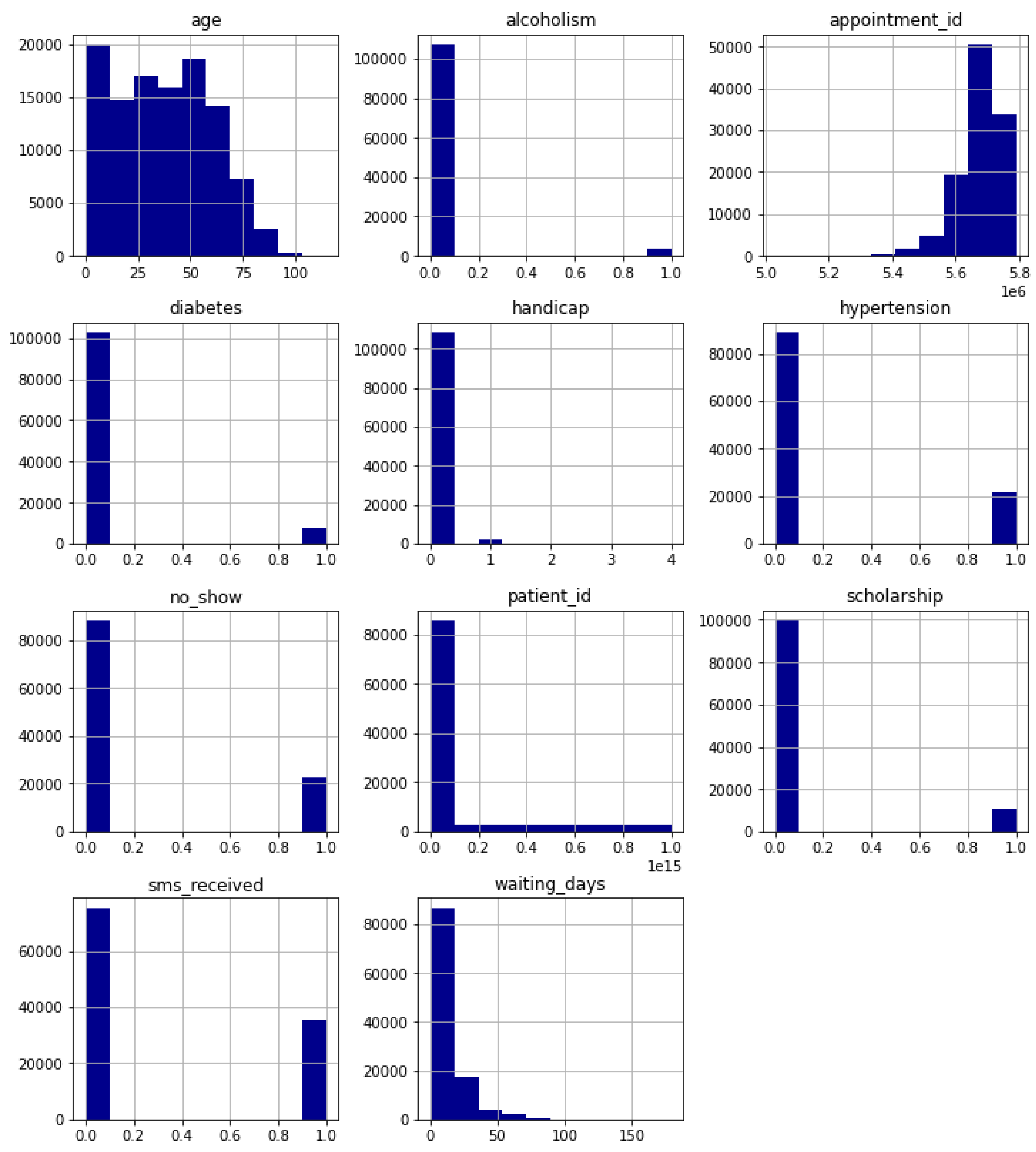
```
In [24]: df.describe()
```

```
Out[24]:          patient_id  appointment_id            age    scholarship  \
         count  1.105210e+05    1.105210e+05  110521.000000  110521.000000
         mean   1.474906e+14    5.675304e+06      37.089386       0.098271
         std    2.560860e+14    7.129691e+04      23.109885       0.297682
```

```
min    3.921700e+04  5.030230e+06    0.000000    0.000000
25%    4.172457e+12  5.640284e+06   18.000000    0.000000
50%    3.173185e+13  5.680573e+06   37.000000    0.000000
75%    9.438963e+13  5.725524e+06   55.000000    0.000000
max    9.999816e+14  5.790484e+06  115.000000    1.000000

        hypertension       diabetes     alcoholism       handicap  \
count  110521.000000  110521.000000  110521.000000  110521.000000
mean        0.197257       0.071869       0.030401       0.022231
std         0.397929       0.258272       0.171690       0.161494
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        no_show   waiting_days
count  110521.000000  110521.000000  110521.000000
mean        0.321043       0.201898      10.184345
std         0.466879       0.401419      15.255153
min         0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000
50%         0.000000       0.000000       4.000000
75%         1.000000       0.000000      15.000000
max         1.000000       1.000000     179.000000
```

Exploratory Data Analysis

```
In [25]: df.hist(figsize=(12,14), color = "darkblue", lw=0);
```

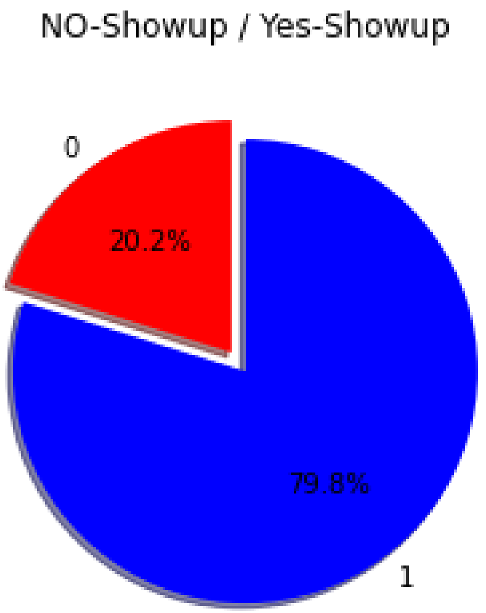


What is the percentage of missed and attended appointments?

```
In [26]: # Percentage  of missed and  attended  appointments

         no_show_labels = df.no_show.unique()
         total_show_no = df['no_show'].value_counts()[0]
         total_show_yes = df['no_show'].value_counts()[1]
         total_show = total_show_no + total_show_yes

         # Get percentage of each gender with respect to tatal number patients
```

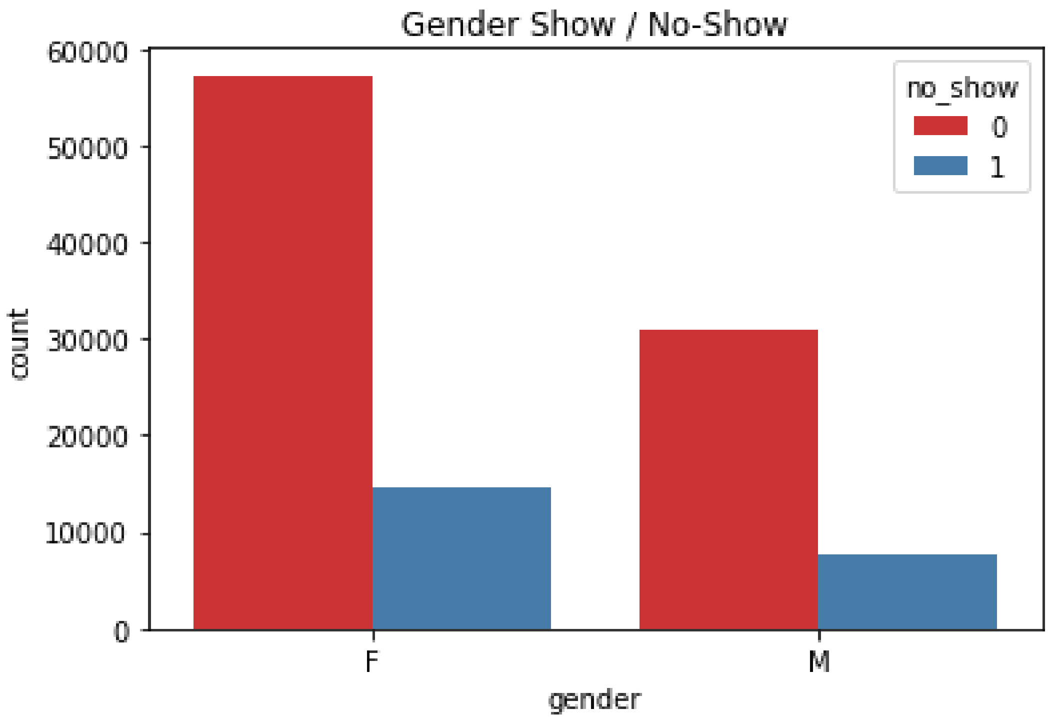
```
        show_no_percentage =(total_show_no / total_show) * 100
        show_yes_percentage  = (total_show_yes / total_show) * 100
        show_count = [total_show_yes,total_show_no]

        # Visualize ratio
        fig = plt.figure()
        plt.pie(x= show_count ,labels=no_show_labels, autopct='%1.1f%%',
                colors  = ['red','blue'],shadow=True,explode=(0.1,0),startangle=90 )
        plt.title('NO-Showup / Yes-Showup', x=0.5, y=1.05, ha='center');
        print('The dataset has {:} % Yes value records '.format(show_yes_percentage.round(1)))
        print('The dataset has {:} %  No value records'.format(show_no_percentage.round(1)))
```

```
The dataset has 20.2 % Yes value records
The dataset has 79.8 %  No value records
```



Which gender did meet its appointment dates ?

```
In [27]: # Gender impact on  appointment show up

        df['gender'].unique()

        all_female_appointments = len(df.loc[df['gender'] == "F"])
        all_male_appointments = len(df.loc[df['gender'] == "M"])

        missed_female_appointments = len(df.loc[(df['gender'] == "F") & (df['no_show'] == 1)])
        missed_male_appointments = len(df.loc[(df['gender'] == "M") & (df['no_show'] == 1)])

        female_missed_ratio = int(round(missed_female_appointments/all_female_appointments*100)
        male_missed_ratio = int(round(missed_male_appointments/all_male_appointments*100))

        ax = sns.countplot(x=df.gender, hue=df.no_show, data=df,palette="Set1")
        ax.set_title(" Gender Show / No-Show ",)
        x_ticks_labels=['Female', 'Male']
        plt.show();

        print('Total females appointments are {} , {}  missed their appointments, with  ratio {
        print('Total males appointments are {} , {}  missed their appointments, with  ratio {}%
```

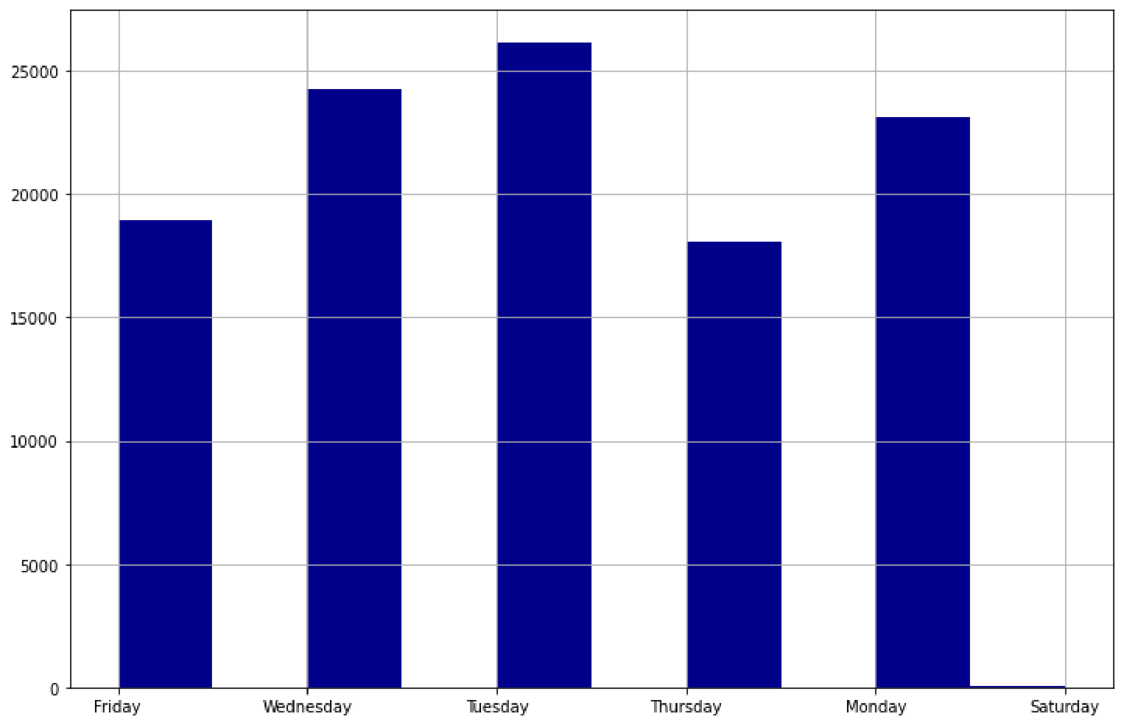


```
Total females appointments are 71836 , 14591  missed their appointments, with  ratio 20%.
Total males appointments are 38685 , 7723  missed their appointments, with  ratio 20%.
```

What is the distribution of days of the week according to Scheduled Day?

```
In [28]: # plot schedule_day distribution to no-show
         df['scheduled_day_name'].hist(figsize=(12,8),color = "darkblue", lw=0);
         plt.suptitle('Scheduled weekdays distribution', ha='center')

Out[28]: Text(0.5, 0.98, 'Scheduled weekdays distribution')
```

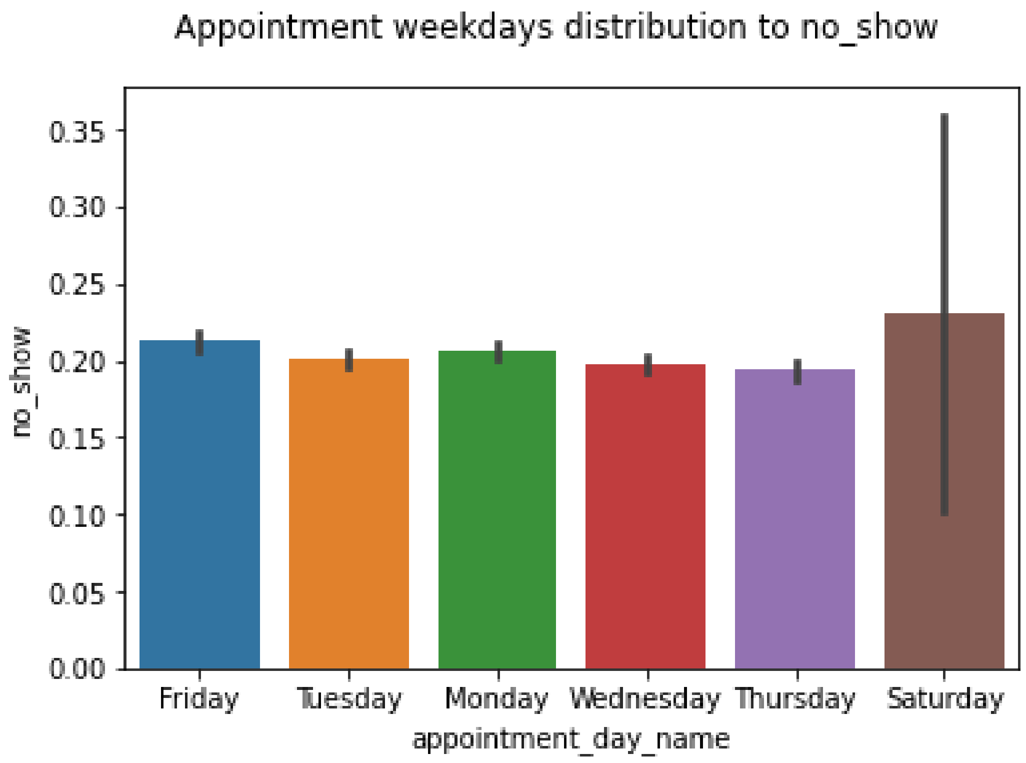
Scheduled weekdays distribution



The Scheduled days distribution among days of the week (Friday-Monday) is almost equal with less scheduled appointments on Friday and Thursday.There are only 24 scheduled appointments on Saturday and 0 on Sunday.

Does the day of the week impact appointment showup?

```
In [29]: #plot appointment_day distribution to no-show

         sns.barplot(y='no_show', x='appointment_day_name', data=df)
         plt.suptitle('Appointment weekdays distribution to no_show', ha='center')

Out[29]: Text(0.5, 0.98, 'Appointment weekdays distribution to no_show')
```



The appointment days distribution among days of the week (Friday-saturday) is almost equal with littel high appointments on saturday.

What factors are important to predict if a patient will show up their appointment?

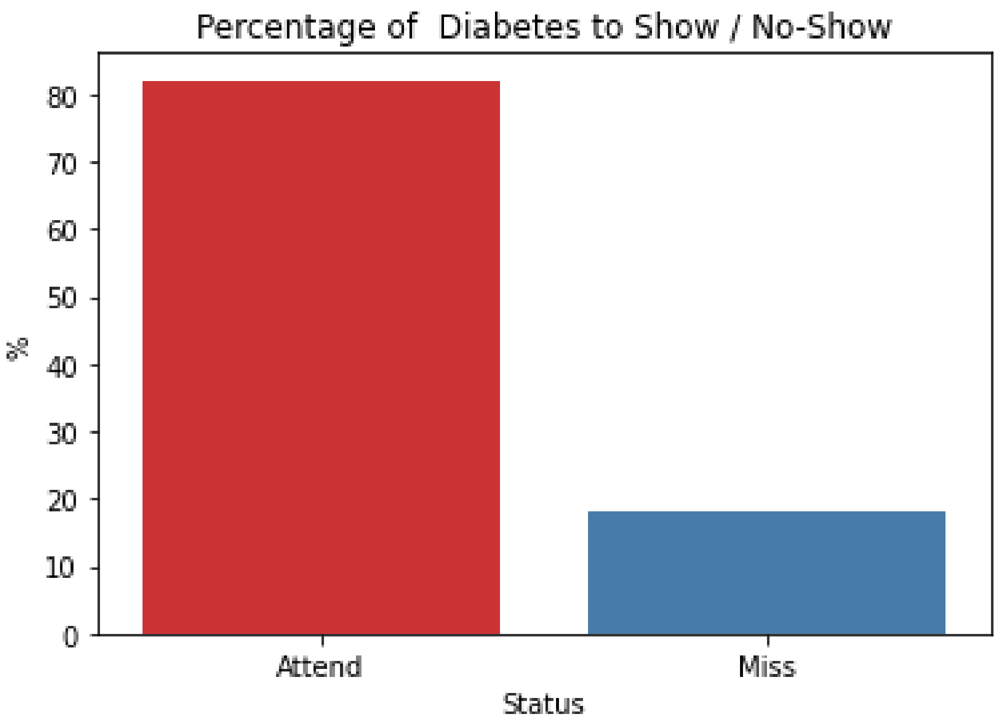
To answer this question we will check the impact of some attributes ['hypertension','diabetes', 'alcoholism', 'sms_received','scholarship', 'age'] on no_show.

```
In [30]: # Function to visualize  the attendance of patients with diseases

         def disease_to_appointment(df, disease_attribute, disease_attribute_printing_name) :

         ## check gender value

             df[disease_attribute].unique()

             all_disease = len(df.loc[df[disease_attribute] == 1])
             all_health = len(df.loc[df[disease_attribute] == 0])

             disease_missed__appointments = len(df.loc[(df[disease_attribute] == 1) & (df['no_sh
             disease_attended__appointments = len(df.loc[(df[disease_attribute] == 1) & (df['no_

             disease_missed_ratio = int(round(disease_missed__appointments/all_disease*100))
             disease_attended_ratio = int(round(disease_attended__appointments/all_disease*100))
```

```
            # ax = sns.countplot(x=df[disease_attribute], hue=df.no_show, data=df,palette="Set3'
             ax = sns.barplot(x=['Attend','Miss'],y=[disease_attended_ratio, disease_missed_rati
             ax.set_xlabel('Status')
             ax.set_ylabel('%')
             ax.set_title('Percentage of  {:} to Show / No-Show'.format(disease_attribute_printi
             #x_ticks_labels=['attend', 'miss']
            # plt.show();

             print('Total patients with  {} , are {} , {}  missed their appointments, with  rati
            , disease_missed__appointments, disease_missed_ratio))
             print('Total patients with   {}, are {} , {}  attended their appointments, with  ra
        print()
```

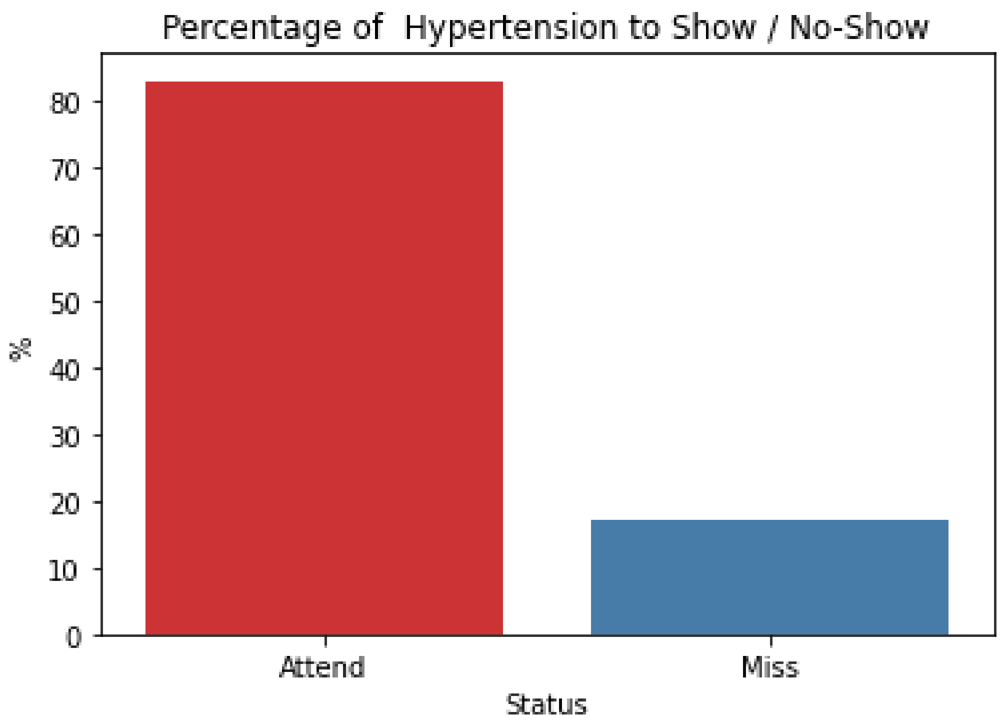
```
In [31]: disease_to_appointment(df, 'diabetes', 'Diabetes')
```

```
Total patients with  Diabetes , are 7943 , 1430  missed their appointments, with  ratio 18%.
Total patients with   Diabetes, are 7943 , 6513  attended their appointments, with  ratio 82%.
```



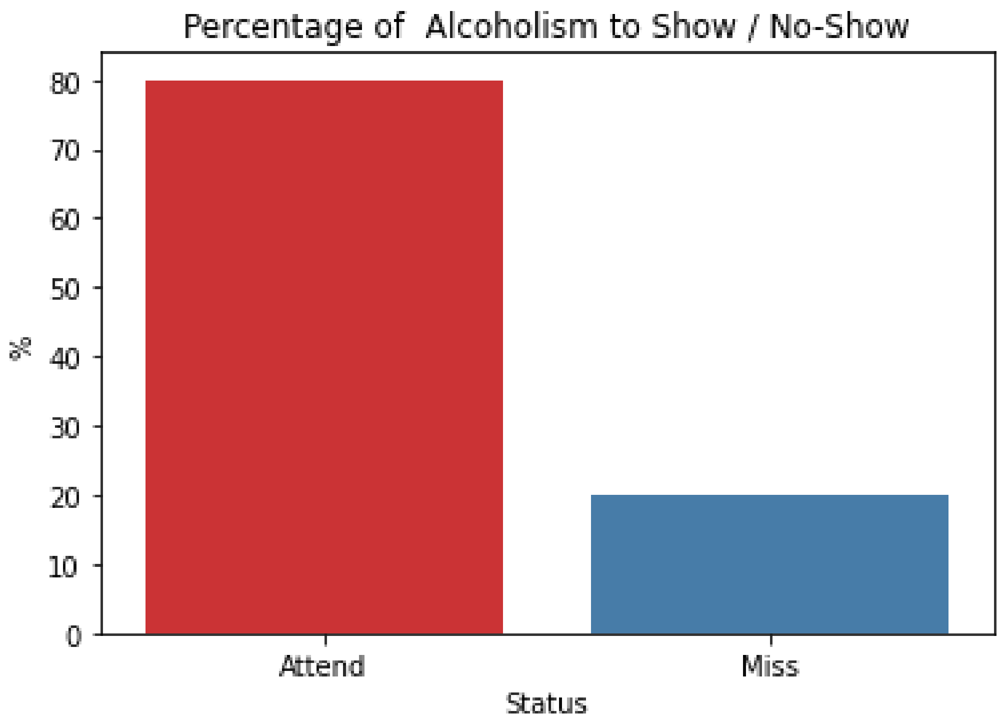
```
In [32]: disease_to_appointment(df, 'hypertension', 'Hypertension')
```

```
Total patients with  Hypertension , are 21801 , 3772  missed their appointments, with  ratio 17%
Total patients with   Hypertension, are 21801 , 18029  attended their appointments, with  ratio
```

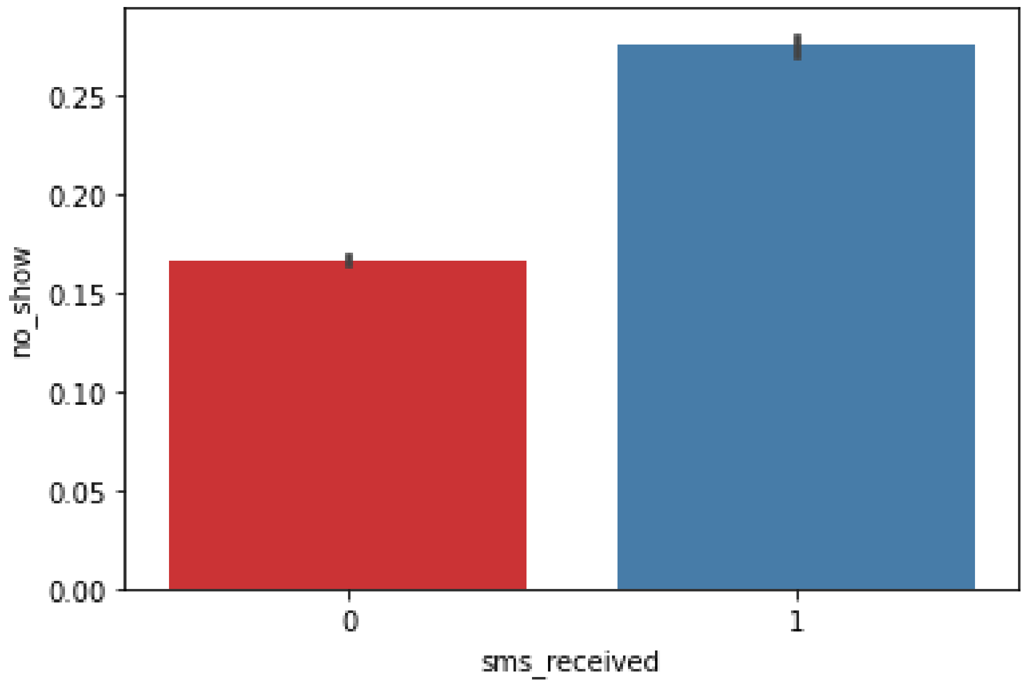


```
In [33]: disease_to_appointment(df, 'alcoholism', 'Alcoholism')
```

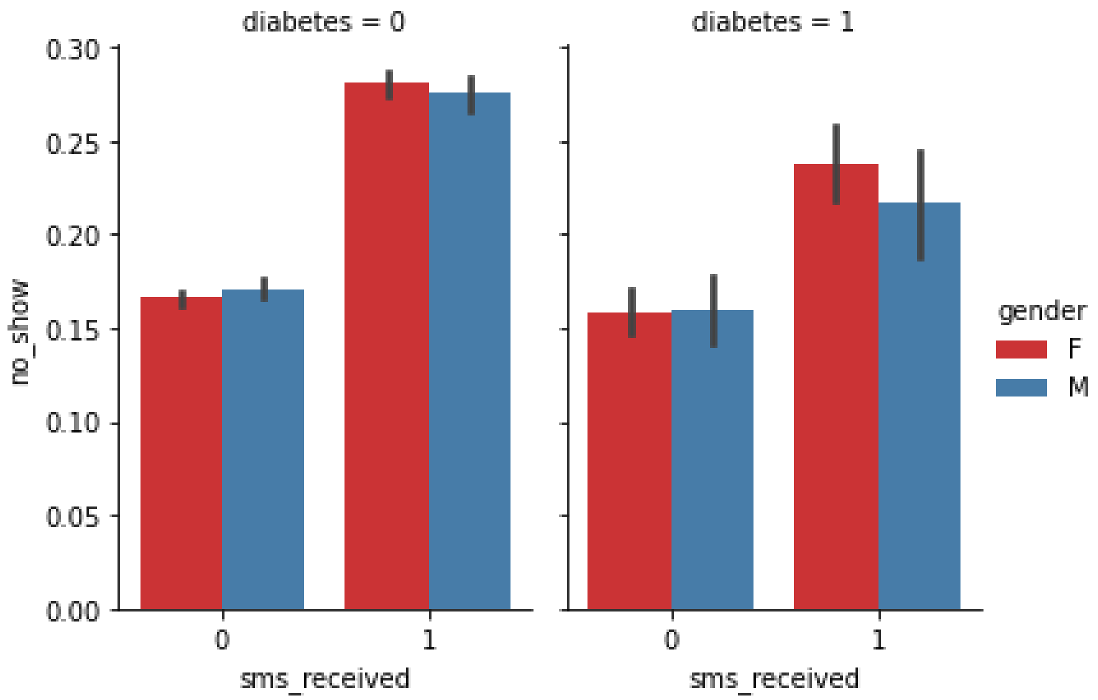
```
Total patients with  Alcoholism , are 3360 , 677  missed their appointments, with  ratio 20%.
Total patients with   Alcoholism, are 3360 , 2683  attended their appointments, with  ratio 80%.
```



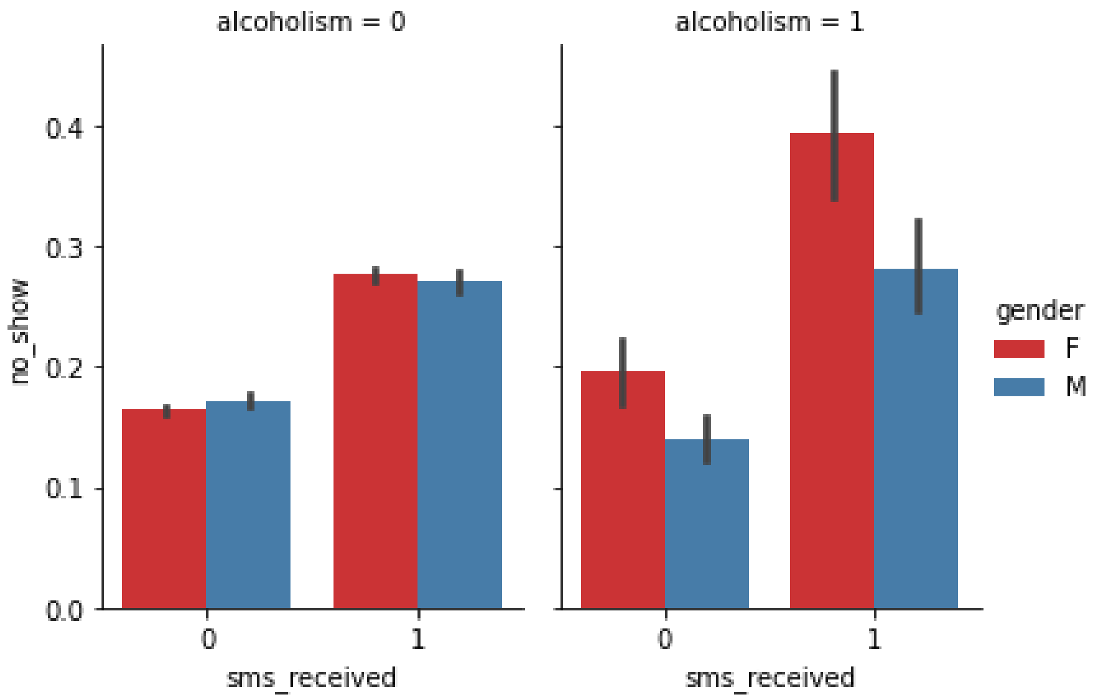
```
In [34]: # plot relationship between recieving sms and no_show

         sns.barplot(y='no_show', x='sms_received', data=df,palette='Set1')

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0xe5b7041850>
```



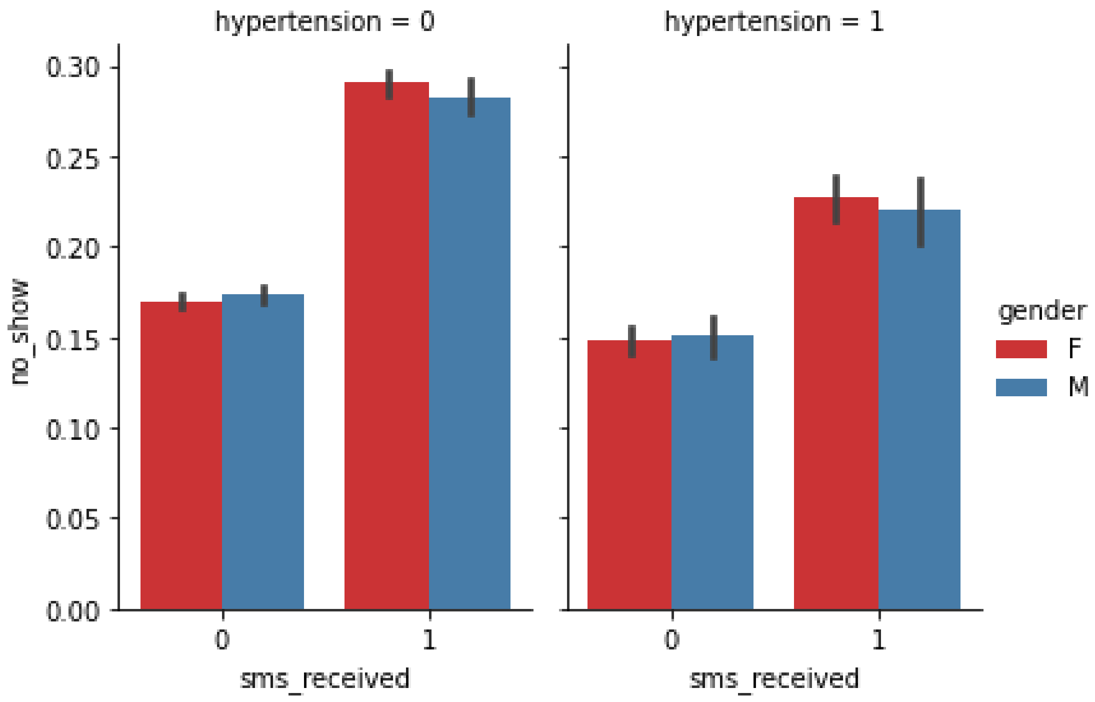
```
In [35]: #plot the impact of receiving sms to diabetes patients on no_show
         g = sns.catplot(x="sms_received", y="no_show",hue="gender", col="diabetes", data=df, ki
```



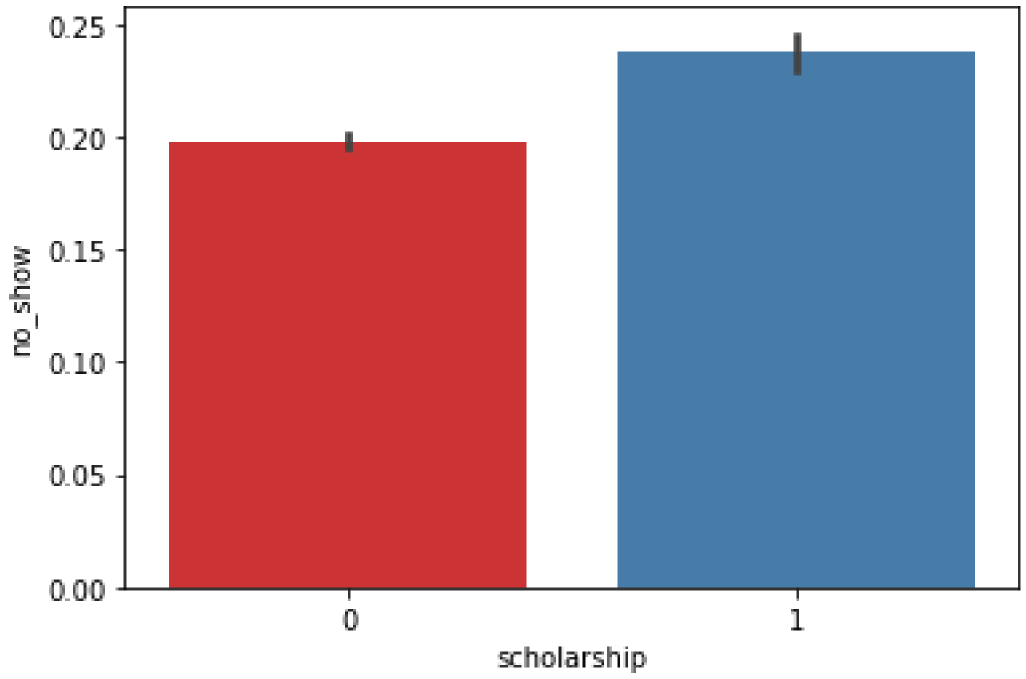
```
In [36]: #plot the impact of receiving sms to alcoholism patients on no_show
         g = sns.catplot(x="sms_received", y="no_show",hue="gender", col="alcoholism", data=df,
```



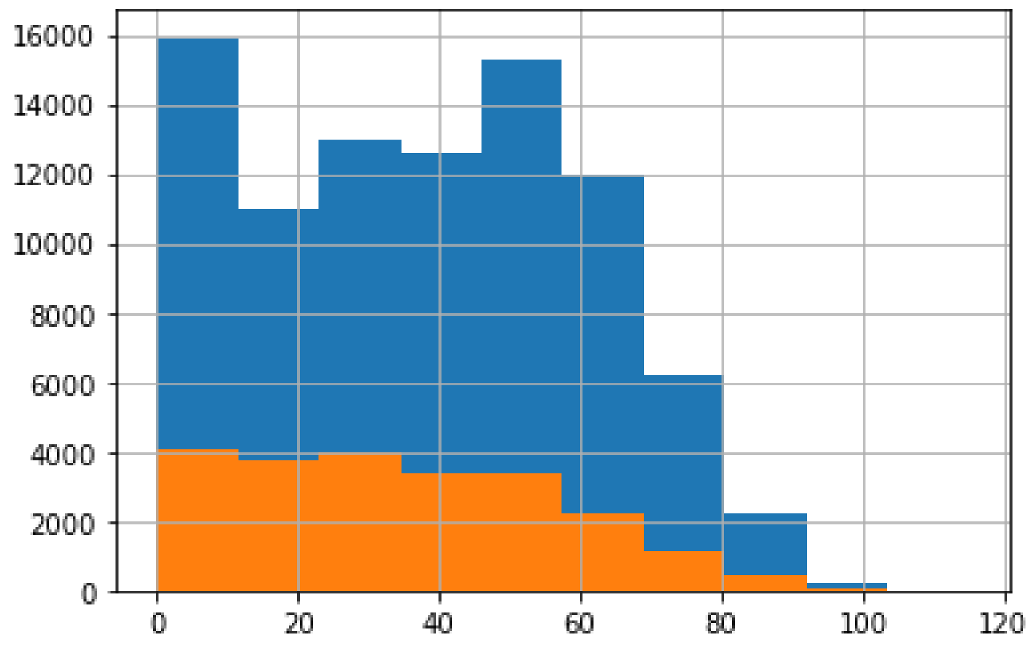
```
In [37]: #plot the impact of receiving sms to hypertension patients on no_show
         g = sns.catplot(x="sms_received", y="no_show",hue="gender", col="hypertension", data=df
```



```
In [38]: # plot relationship between scholarship and no_show

         sns.barplot(y='no_show', x='scholarship', data=df, palette='Set1')

Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0xe5b70332e0>
```



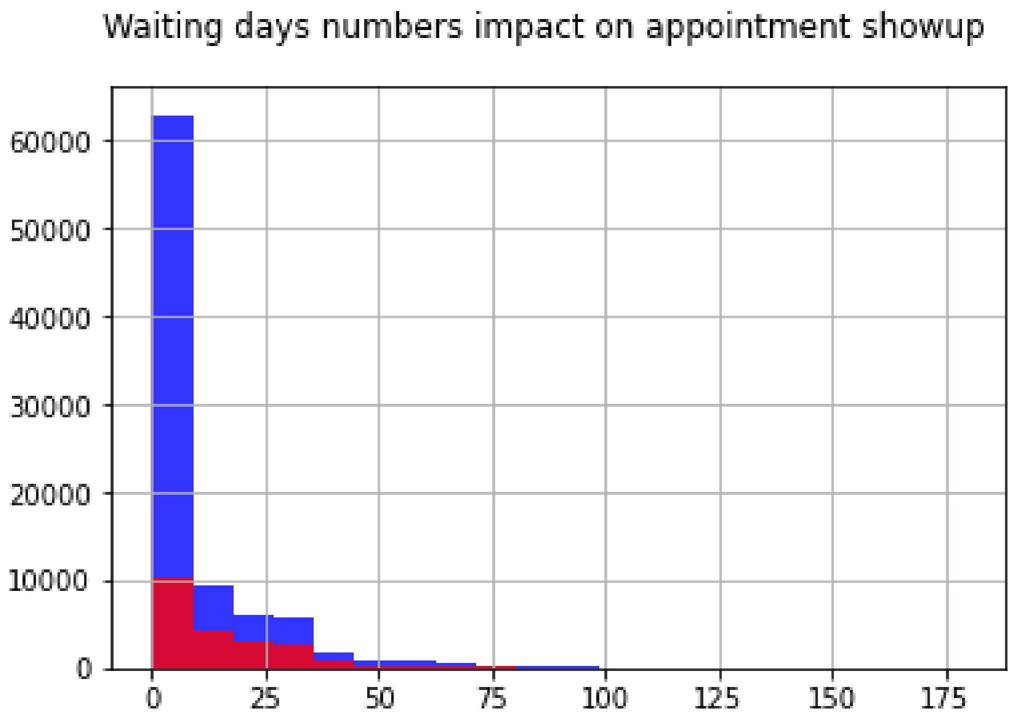
```
In [39]: # plot relationship between age and no_show

         df.groupby('no_show').age.hist()

Out[39]: no_show
         0    AxesSubplot(0.125,0.125;0.775x0.755)
         1    AxesSubplot(0.125,0.125;0.775x0.755)
         Name: age, dtype: object
```



How does waiting days number affect patient showup appointment?

```
In [40]: #plot relationship between waiting days and no_show

         attaned = (df.no_show == 0)
         missed = (df.no_show == 1)
         df.waiting_days[attaned].hist(alpha=0.8, bins=20, color  = 'blue');
         df.waiting_days[missed].hist(alpha=0.8, bins=20, color = 'red');
         plt.suptitle('Waiting days numbers impact on appointment showup', ha='center')
```

```
Out[40]: Text(0.5, 0.98, 'Waiting days numbers impact on appointment showup')
```



The shorter the waiting period, the more patients meet their appointment.

2 Conclusions

After cleaning and investigateing the data few insights were inferred:

- The dataset have 110527 records.
- 71839 of records gender are females with 65% ratio and the rest are males.
- Total females appointments are 71836 , 14591 missed their appointments, with ratio 20%.
- Total males appointments are 38685 , 7723 missed their appointments, with ratio 20%.
- The age distribution from 0 to 115 as The average are 37 years . 25% of patients under 18 and 75 % of patients under 55.
- The Scheduling days started on 2015-11-10 and ended on 2016-06-08.
- The appointments started on 2016-04-29 and ended on 2016-06-08.
- The Scheduled days distribution among days of the week (Friday-Monday) is almost equal with less scheduled appointments on Friday and Thursday. There are only 24 scheduled appointments on Saturday and 0 on Sunday.
- The appointment days distribution among days of the week (Friday-saturday) is almost equal with littel high appointments on saturday.

- The average of waiting days is 10 days. 25% of patients waiting 0 days it means they scheduled thier appoinpment in the same day while Up to 4 waiting days to 50 % of patients and up to 15 days to 75 % of patients. The maximum waiting days was 179 days.
- The shorter the waiting period, the more patients meet their appointment.
- Most of the patients are not alcoholics. Total patients with alcoholism , are 3360 , 677 missed their appointments, with ratio 20% and 2683 attended their appointments, with ratio 80%.
- Most of the patients are not diabetes. Total patients with Diabetes , are 7943 , 1430 missed their appointments, with ratio 18% and 6513 attended their appointments, with ratio 82%.
- Most of the patients are not hypertension but more than diabetes and alcoholism . Total patients with Hypertension , are 21801 , 3772 missed their appointments, with ratio 17% and 18029 attended their appointments, with ratio 83%.
- Most of the patients do not receive sms but (alcoholism,diabetes,hypertension) patients that receive sms meet their appointment compared to others.
- 75 % of patients do not have scholarship but the portion that have it meet their appointment more compared to the others.
- According to charts most of attributes values distributions to no-show attribute look very similar. There is no clear impact on no-show behaviour.
- There is limitation in database as the appointments period covered in database is very short .