investigate-a-dataset-template

September 7, 2021

1 Project: Medical Appointment No-Shows Investigation

1.1 Table of Contents

## Introduction

The dataset object of this investigation is a series of appointments records at the Brazilian Unified Health System (SUS). The SUS is a free and public healthcare system based in equality and universal access for the people of Brazil. In 2019 17.3 millions of brazilians looked for some primary attetion service at the SUS in the last six months before particiapation in the PNS-2019 assessment. (source:Ibge)

The SUS system is characterized by the tranditional limitation of public systems, the high demands rates, high no-show rates and long wating time for appoinments.

The primary posed question for this dataset is what factors or features of an appoiment can drive to a no-show event. Is there any correlation between, gender, age or schedule/appoinment caracteristcs (day of the week, time of the day, month, season, timespan,. etc..) that are important for a no-show event?

Dataset description:

> This dataset collects information from 100k medical appointments in Brazil and is focused on the question of whether or not patients show up for their appointment. A number of characteristics about the patient are included in each row. 'ScheduledDay' tells us on what day the patient set up their appointment. 'Neighborhood' indicates the location of the hospital. 'Scholarship' indicates whether or not the patient is enrolled in Brasilian welfare program Bolsa Família. Be careful about the encoding of the last column: it says 'No' if the patient showed up to their appointment, and 'Yes' if they did not show up.

Delete me !!! > **Tip**: In this section of the report, provide a brief introduction to the dataset you've selected for analysis. At the end of this section, describe the questions that you plan on exploring over the course of the report. Try to build your report around the analysis of at least one dependent variable and three independent variables. > > If you haven't yet selected and downloaded your data, make sure you do that first before coming back here. If you're not sure what questions to ask right now, then make sure you familiarize yourself with the variables and the dataset context for ideas of what to explore.

[1]:
```
# Use this cell to set up import statements for all of the packages that you
#   plan to use.
# Remember to include a 'magic word' so that your visualizations are plotted
#   inline with the notebook. See this page for more:
#   http://ipython.readthedocs.io/en/stable/interactive/magics.html
```

[2]:
```
# Importing pandas for data manipulation
# Importing matplolib ans seaborn for data visualization
# Import numpy for numerical operations

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

## Data Wrangling

1.1.1 General Properties

[3]:
```
df = pd.read_csv('noshowappointments-kagglev2-may-2016.csv')
```

[4]:
```
# Looking at dataset head for data import verification
df.head()
```

[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
```

Missing values and data types *using info for missing data and data types verification*

[5]:
```
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
```

using describe for visualizing statistics and ranges

[6]:
```
df.describe()
```

[6]:
```
          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
```

[7]:
```
## Numeber of unique values
df.nunique()
```

[7]:
```
PatientId          62299
AppointmentID     110527
Gender                 2
ScheduledDay      103549
AppointmentDay        27
Age                  104
Neighbourhood         81
Scholarship            2
Hipertension           2
Diabetes               2
Alcoholism             2
Handcap                5
SMS_received           2
No-show                2
dtype: int64
```

1.1.2 Data Cleaning:

Findings:

1. General:
   - Dataset with 110527 rows and 14 columns with types:float64(1), int64(8), object(5)
   - Upper and lower caps mixed in column names
   - No missing values found
   - "-" charcter at column name
2. Column PatientId - type float64. It makes more sense to have this as strings.
3. Column AppoitentID - type int64. It makes more sense to have this as strings
4. Colunn SchedulleDay and AppointmentDay make sense for analisys in datetime format.
5. There are multiple appoimments for the same patient identification, not really a problem, but require attention not to bias the analisys
6. Need fix Age of value or values == -1

Fixing column names seting lower and removing special chars

[8]:
```
columns = df.columns
columns = columns.str.lower()
columns = columns.str.replace('-','')
columns = columns.str.replace('_','')
df.columns = columns
```

Checking fix.

[9]:
```
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  smsreceived     110527 non-null  int64
 13  noshow          110527 non-null  object
dtypes: float64(1), int64(8), object(5)
memory usage: 11.8+ MB
```

Changing: *- patientid and appointimentid types to string.*

- *scheduleday and appoinmentday to datetime*

[10]:
```
df.patientid = df.patientid.astype(int).astype(str)
df.appointmentid = df.appointmentid.astype(str)
df.scheduledday = pd.to_datetime(df.scheduledday)
df.appointmentday = pd.to_datetime(df.appointmentday)
#verifing
df.head(1)
```

[10]:
```
        patientid appointmentid gender              scheduledday  \
0  29872499824296       5642903      F 2016-04-29 18:38:08+00:00

             appointmentday  age    neighbourhood  scholarship  hipertension  \
0 2016-04-29 00:00:00+00:00   62  JARDIM DA PENHA            0             1
```

```
   diabetes  alcoholism  handcap  smsreceived noshow
0         0           0        0            0     No
```

[11]:
```
df.describe()
```

[11]:
```
                 age    scholarship   hipertension       diabetes  \
count  110527.000000  110527.000000  110527.000000  110527.000000
mean       37.088874       0.098266       0.197246       0.071865
std        23.110205       0.297675       0.397921       0.258265
min        -1.000000       0.000000       0.000000       0.000000
25%        18.000000       0.000000       0.000000       0.000000
50%        37.000000       0.000000       0.000000       0.000000
75%        55.000000       0.000000       0.000000       0.000000
max       115.000000       1.000000       1.000000       1.000000

          alcoholism        handcap    smsreceived
count  110527.000000  110527.000000  110527.000000
mean        0.030400       0.022248       0.321026
std         0.171686       0.161543       0.466873
min         0.000000       0.000000       0.000000
25%         0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000
75%         0.000000       0.000000       1.000000
max         1.000000       4.000000       1.000000
```

[12]:
```
# Checking unique lists for mistakes, typos, etc.
print(sorted(df.neighbourhood.unique()))
print(sorted(df.handcap.unique()))
```

```
['AEROPORTO', 'ANDORINHAS', 'ANTÔNIO HONÓRIO', 'ARIOVALDO FAVALESSA', 'BARRO
VERMELHO', 'BELA VISTA', 'BENTO FERREIRA', 'BOA VISTA', 'BONFIM', 'CARATOÍRA',
'CENTRO', 'COMDUSA', 'CONQUISTA', 'CONSOLAÇÃO', 'CRUZAMENTO', 'DA PENHA', 'DE
LOURDES', 'DO CABRAL', 'DO MOSCOSO', 'DO QUADRO', 'ENSEADA DO SUÁ',
'ESTRELINHA', 'FONTE GRANDE', 'FORTE SÃO JOÃO', 'FRADINHOS', 'GOIABEIRAS',
'GRANDE VITÓRIA', 'GURIGICA', 'HORTO', 'ILHA DAS CAIEIRAS', 'ILHA DE SANTA
MARIA', 'ILHA DO BOI', 'ILHA DO FRADE', 'ILHA DO PRÍNCIPE', 'ILHAS OCEÂNICAS DE
TRINDADE', 'INHANGUETÁ', 'ITARARÉ', 'JABOUR', 'JARDIM CAMBURI', 'JARDIM DA
PENHA', 'JESUS DE NAZARETH', 'JOANA D´ARC', 'JUCUTUQUARA', 'MARIA ORTIZ',
'MARUÍPE', 'MATA DA PRAIA', 'MONTE BELO', 'MORADA DE CAMBURI', 'MÁRIO CYPRESTE',
'NAZARETH', 'NOVA PALESTINA', 'PARQUE INDUSTRIAL', 'PARQUE MOSCOSO', 'PIEDADE',
'PONTAL DE CAMBURI', 'PRAIA DO CANTO', 'PRAIA DO SUÁ', 'REDENÇÃO', 'REPÚBLICA',
'RESISTÊNCIA', 'ROMÃO', 'SANTA CECÍLIA', 'SANTA CLARA', 'SANTA HELENA', 'SANTA
LUÍZA', 'SANTA LÚCIA', 'SANTA MARTHA', 'SANTA TEREZA', 'SANTO ANDRÉ', 'SANTO
ANTÔNIO', 'SANTOS DUMONT', 'SANTOS REIS', 'SEGURANÇA DO LAR', 'SOLON BORGES',
'SÃO BENEDITO', 'SÃO CRISTÓVÃO', 'SÃO JOSÉ', 'SÃO PEDRO', 'TABUAZEIRO',
'UNIVERSITÁRIO', 'VILA RUBIM']
[0, 1, 2, 3, 4]
```

[13]:
```
# Checking Age column for typo error

print('Age min value:{} , Age Max value:{}'.format(df.age.min(),df.age.max()))
```

```
Age min value:-1 , Age Max value:115
```

[14]:
```
df.drop(df[df.age == -1].index, axis=0,inplace=True)

#confirm drop outlier

print('Age min value:{} , Age Max value:{}'.format(df.age.min(),df.age.max()))
```

```
Age min value:0 , Age Max value:115
```

One possible influece to noshow is the time between scheduled and the appoinment day.

- ***Creating the column timespan column as difference in days between appointment-day and scheduledday colouns.***
- ***Creating the column appdayofweek column as day of the week of appointmentday colouns.***
- ***Creating the column schddayofweek column day of the week of scheduledday colouns.***

[15]:
```
df['timespan']= df['appointmentday'].dt.date - df['scheduledday'].dt.date
df['timespan']=df['timespan']/np.timedelta64(1, 'D')
df['appdayofweek'] = df['appointmentday'].dt.dayofweek
df['schddayofweek'] = df['scheduledday'].dt.dayofweek
```

[16]:
```
df.head()
```

[16]:
```
         patientid  appointmentid gender              scheduledday  \
0   29872499824296        5642903      F 2016-04-29 18:38:08+00:00
1  558997776694438        5642503      M 2016-04-29 16:08:27+00:00
2    4262962299951        5642549      F 2016-04-29 16:19:04+00:00
3     867951213174        5642828      F 2016-04-29 17:29:31+00:00
4    8841186448183        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

   hipertension  diabetes  alcoholism  handcap  smsreceived noshow  timespan  \
0             1         0           0        0            0     No       0.0
1             0         0           0        0            0     No       0.0
2             0         0           0        0            0     No       0.0
```

```
3              0         0           0        0            0     No       0.0
4              1         1           0        0            0     No       0.0

   appdayofweek  schddayofweek
0             4              4
1             4              4
2             4              4
3             4              4
4             4              4
```

[17]:
```
df.describe()
```

[17]:
```
                 age    scholarship   hipertension       diabetes  \
count  110526.000000  110526.000000  110526.000000  110526.000000
mean       37.089219       0.098266       0.197248       0.071865
std        23.110026       0.297676       0.397923       0.258266
min         0.000000       0.000000       0.000000       0.000000
25%        18.000000       0.000000       0.000000       0.000000
50%        37.000000       0.000000       0.000000       0.000000
75%        55.000000       0.000000       0.000000       0.000000
max       115.000000       1.000000       1.000000       1.000000

          alcoholism        handcap    smsreceived       timespan  \
count  110526.000000  110526.000000  110526.000000  110526.000000
mean        0.030400       0.022248       0.321029      10.183794
std         0.171686       0.161543       0.466874      15.255034
min         0.000000       0.000000       0.000000      -6.000000
25%         0.000000       0.000000       0.000000       0.000000
50%         0.000000       0.000000       0.000000       4.000000
75%         0.000000       0.000000       1.000000      15.000000
max         1.000000       4.000000       1.000000     179.000000

        appdayofweek  schddayofweek
count  110526.000000  110526.000000
mean        1.858260       1.851971
std         1.371667       1.378515
min         0.000000       0.000000
25%         1.000000       1.000000
50%         2.000000       2.000000
75%         3.000000       3.000000
max         5.000000       5.000000
```

Previous decribe verfication catched up negative values and probably outlier of 179. Need verification and cleaning if necessary.

[18]:
```
df.query('timespan>100').count()
```

[18]:
```
patientid          138
appointmentid      138
gender             138
scheduledday       138
appointmentday     138
age                138
neighbourhood      138
scholarship        138
hipertension       138
diabetes           138
alcoholism         138
handcap            138
smsreceived        138
noshow             138
timespan           138
appdayofweek       138
schddayofweek      138
dtype: int64
```

It looks that there are a significant number of appoinments with timespan above 100 days, showing its not outliers. Do not require cleaning

[19]:
```
df.query('timespan<0')
```

[19]:
```
             patientid appointmentid gender              scheduledday  \
27033    7839272661752       5679978      M 2016-05-10 10:51:53+00:00
55226    7896293967868       5715660      F 2016-05-18 14:50:41+00:00
64175   24252258389979       5664962      F 2016-05-05 13:43:58+00:00
71533  998231581612122       5686628      F 2016-05-11 13:49:20+00:00
72362    3787481966821       5655637      M 2016-05-04 06:50:57+00:00

                 appointmentday  age  neighbourhood  scholarship  \
27033 2016-05-09 00:00:00+00:00   38    RESISTÊNCIA            0
55226 2016-05-17 00:00:00+00:00   19  SANTO ANTÔNIO            0
64175 2016-05-04 00:00:00+00:00   22     CONSOLAÇÃO            0
71533 2016-05-05 00:00:00+00:00   81  SANTO ANTÔNIO            0
72362 2016-05-03 00:00:00+00:00    7     TABUAZEIRO            0

       hipertension  diabetes  alcoholism  handcap  smsreceived noshow  \
27033             0         0           0        1            0    Yes
55226             0         0           0        1            0    Yes
64175             0         0           0        0            0    Yes
71533             0         0           0        0            0    Yes
72362             0         0           0        0            0    Yes

       timespan  appdayofweek  schddayofweek
27033      -1.0             0              1
55226      -1.0             1              2
```

```
64175       -1.0              2               3
71533       -6.0              3               2
72362       -1.0              1               2
```

*** The appoitments with negative timespan are all noshow values and majority with one day differece between scheduled and appoinment day. Because it is only 4 records I decide to drop them.***

```
[20]: df.drop(df.query('timespan<0').index,inplace=True)
```

verifying cleaning

```
[21]: df.query('timespan<0')
```

```
[21]: Empty DataFrame
      Columns: [patientid, appointmentid, gender, scheduledday, appointmentday, age,
      neighbourhood, scholarship, hipertension, diabetes, alcoholism, handcap,
      smsreceived, noshow, timespan, appdayofweek, schddayofweek]
      Index: []
```

```
[22]: df.describe()
```

```
[22]:                  age    scholarship   hipertension       diabetes  \
      count  110521.000000  110521.000000  110521.000000  110521.000000
      mean       37.089386       0.098271       0.197257       0.071869
      std        23.109885       0.297682       0.397929       0.258272
      min         0.000000       0.000000       0.000000       0.000000
      25%        18.000000       0.000000       0.000000       0.000000
      50%        37.000000       0.000000       0.000000       0.000000
      75%        55.000000       0.000000       0.000000       0.000000
      max       115.000000       1.000000       1.000000       1.000000

                alcoholism        handcap    smsreceived       timespan  \
      count  110521.000000  110521.000000  110521.000000  110521.000000
      mean        0.030401       0.022231       0.321043      10.184345
      std         0.171690       0.161494       0.466879      15.255153
      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       4.000000
      75%         0.000000       0.000000       1.000000      15.000000
      max         1.000000       4.000000       1.000000     179.000000

              appdayofweek  schddayofweek
      count  110521.000000  110521.000000
      mean        1.858280       1.851965
      std         1.371677       1.378539
      min         0.000000       0.000000
      25%         1.000000       1.000000
```

```
50%          2.000000       2.000000
75%          3.000000       3.000000
max          5.000000       5.000000
```

## Exploratory Data Analysis

> **Tip**: Now that you've trimmed and cleaned your data, you're ready to move on to exploration. Compute statistics and create visualizations with the goal of addressing the research questions that you posed in the Introduction section. It is recommended that you be systematic with your approach. Look at one variable at a time, and then follow it up by looking at relationships between variables.

[23]:
```
## masking

noshow = df.noshow == 'Yes'
show   = df.noshow == 'No'
```

1.1.3 Research Question 1 : *Is there a prefered gender for appointment no-show?*

First check the unviverse of appointments in respect to gender

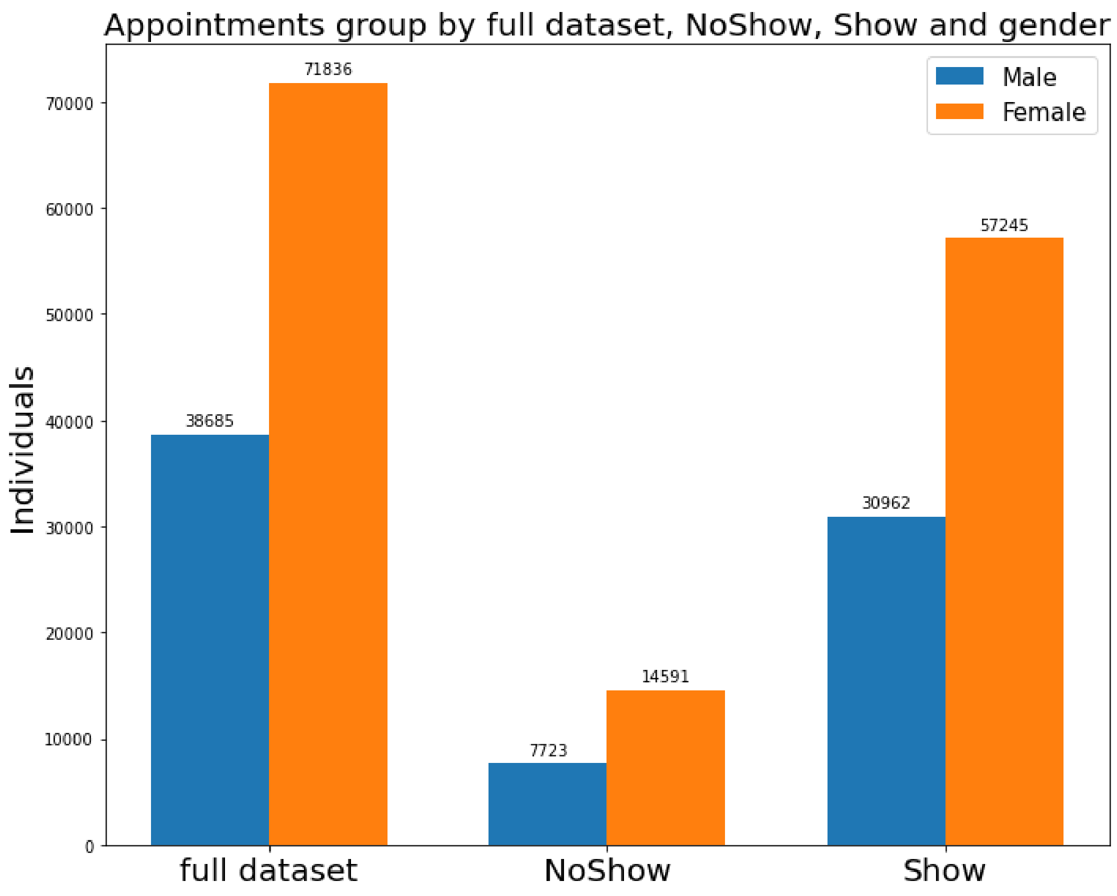
[24]:
```
total = df.gender.value_counts()
total.sum()
```

[24]:
```
110521
```

[25]:
```
noshow_gender = df[noshow].gender.value_counts()
```

[26]:
```
show_gender = df[show].gender.value_counts()
```

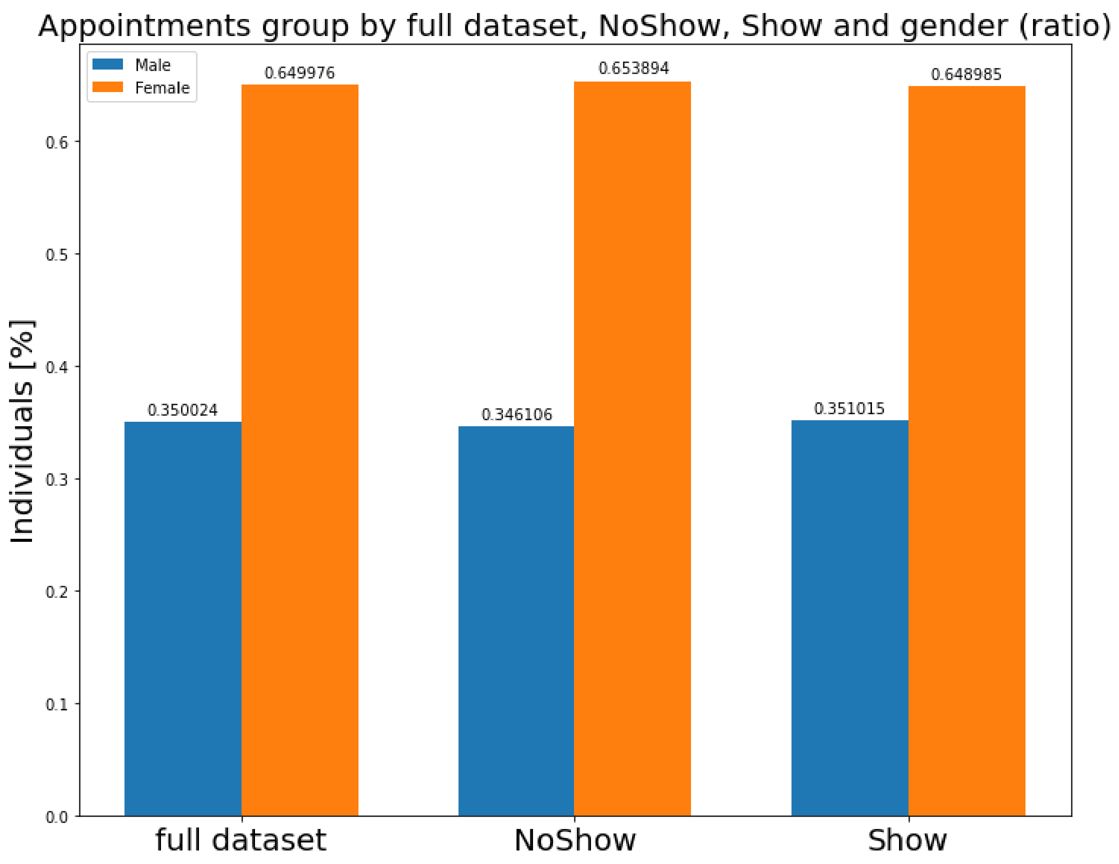
[27]:
```
# Supported by MatplotLib documnetation: https://matplotlib.org/stable/gallery/
↪lines_bars_and_markers/barchart.html

female_hights = [total[0], noshow_gender[0], show_gender[0]]
male_hights = [total[1], noshow_gender[1], show_gender[1]]

labels = ['full dataset', 'NoShow', 'Show']

x = np.arange(len(labels))

width = 0.35

fig, ax = plt.subplots(figsize=(10,8))

bars1 = ax.bar(x - width/2, male_hights, width, label='Male')
bars2 = ax.bar(x + width/2, female_hights, width, label='Female')

ax.set_ylabel('Individuals',fontsize=20)
ax.set_title('Appointments group by full dataset, NoShow, Show and␣
↪gender',fontsize=20)
ax.set_xticks(x)
ax.set_xticklabels(labels,fontsize=20)
ax.legend(fontsize=15)

ax.bar_label(bars1, padding=3)
ax.bar_label(bars2, padding=3)


fig.tight_layout()

plt.show()
```



Checking if the same gender ratio preserves for the total, NoShow and Show.

[28]:
```
# Supported by MatplotLib documnetation: https://matplotlib.org/stable/gallery/
↪lines_bars_and_markers/barchart.html
```

```
female_hights = [total[0]/total.sum(), noshow_gender[0]/noshow_gender.sum(),␣
 ↪show_gender[0]/show_gender.sum()]
male_hights = [total[1]/total.sum(), noshow_gender[1]/noshow_gender.sum(),␣
 ↪show_gender[1]/show_gender.sum()]

labels = ['full dataset', 'NoShow', 'Show']
x = np.arange(len(labels))

width = 0.35

fig, ax = plt.subplots(figsize=(10,8))

bars1 = ax.bar(x - width/2, male_hights, width, label='Male')
bars2 = ax.bar(x + width/2, female_hights, width, label='Female')

ax.set_ylabel('Individuals [%]',fontsize=20)
ax.set_title('Appointments group by full dataset, NoShow, Show and gender␣
 ↪(ratio)',fontsize=20)
ax.set_xticks(x)
ax.set_xticklabels(labels,fontsize=20)
ax.legend()

ax.bar_label(bars1, padding=3)
ax.bar_label(bars2, padding=3)


fig.tight_layout()

plt.show()
```



```
[29]: print('NoShow Female ratio:',noshow_gender[0]/noshow_gender.sum())
      print('NoShow Male ratio:',noshow_gender[1]/noshow_gender.sum())
      print('Show Female ratio:',show_gender[0]/show_gender.sum())
      print('Show Male ratio:',show_gender[1]/show_gender.sum())
```

```
NoShow Female ratio: 0.6538944160616653
NoShow Male ratio: 0.3461055839383347
Show Female ratio: 0.6489847744510073
Show Male ratio: 0.3510152255489927
```

Observation: It is possible to verify from graphics and numerical values that there is no prevalence of gender in noshow events. The ratios between gender for full dataset, Noshow and show are preserved.

1.1.4 Research Question 2: Is there any relation between the timespan between schedule and appointtment day and no-show.

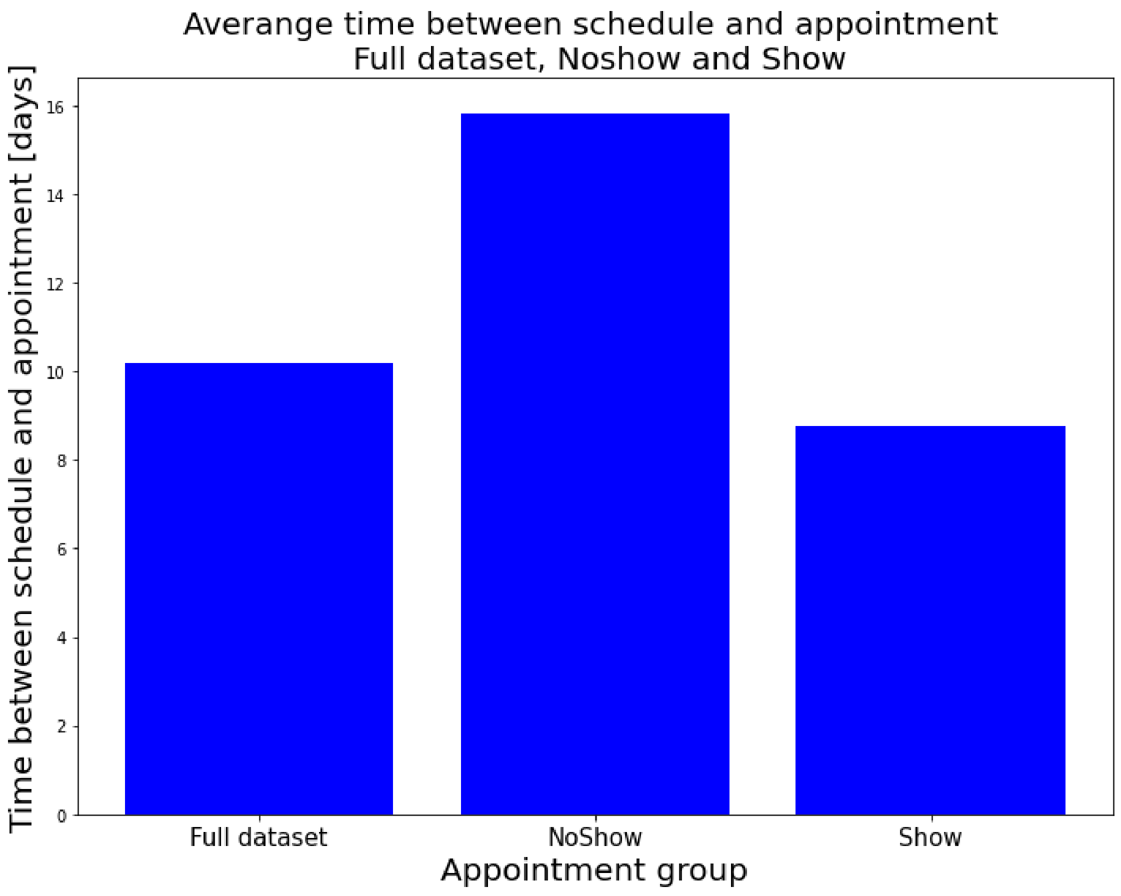
Calculating timespan means

```
[30]: noshow_mean = df[noshow].timespan.mean()
      show_mean = df[show].timespan.mean()
```

```
full_mean = df.timespan.mean()
```

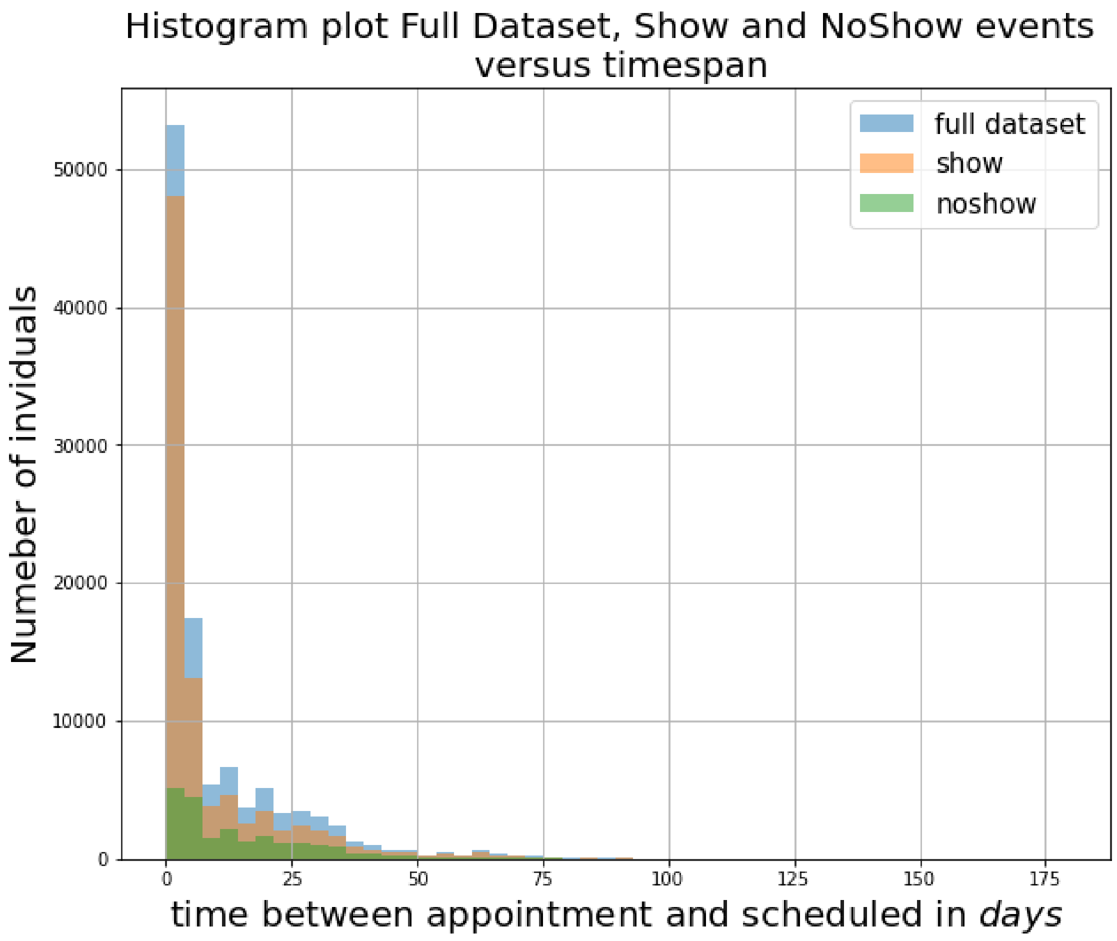
Creating bar plot for visualizing timespan means for the full dataset, show and noshow

```
[31]: fig, ax = plt.subplots(figsize=(10,8))
height = [full_mean,noshow_mean,show_mean]
bars = ('Full dataset', 'NoShow', 'Show')
x_pos = np.arange(len(bars))
ax.bar(x_pos, height, color='b')
ax.set_xticks(x_pos)
ax.set_xticklabels(bars,fontsize=15)
ax.set_title('Averange time between schedule and appointment \n Full dataset,␣
 ↪Noshow and Show',fontsize=20)
ax.set_xlabel('Appointment group',fontsize=20)
ax.set_ylabel('Time between schedule and appointment [days]',fontsize=20)
fig.tight_layout()
plt.show()
```



Visualizing data as histograms

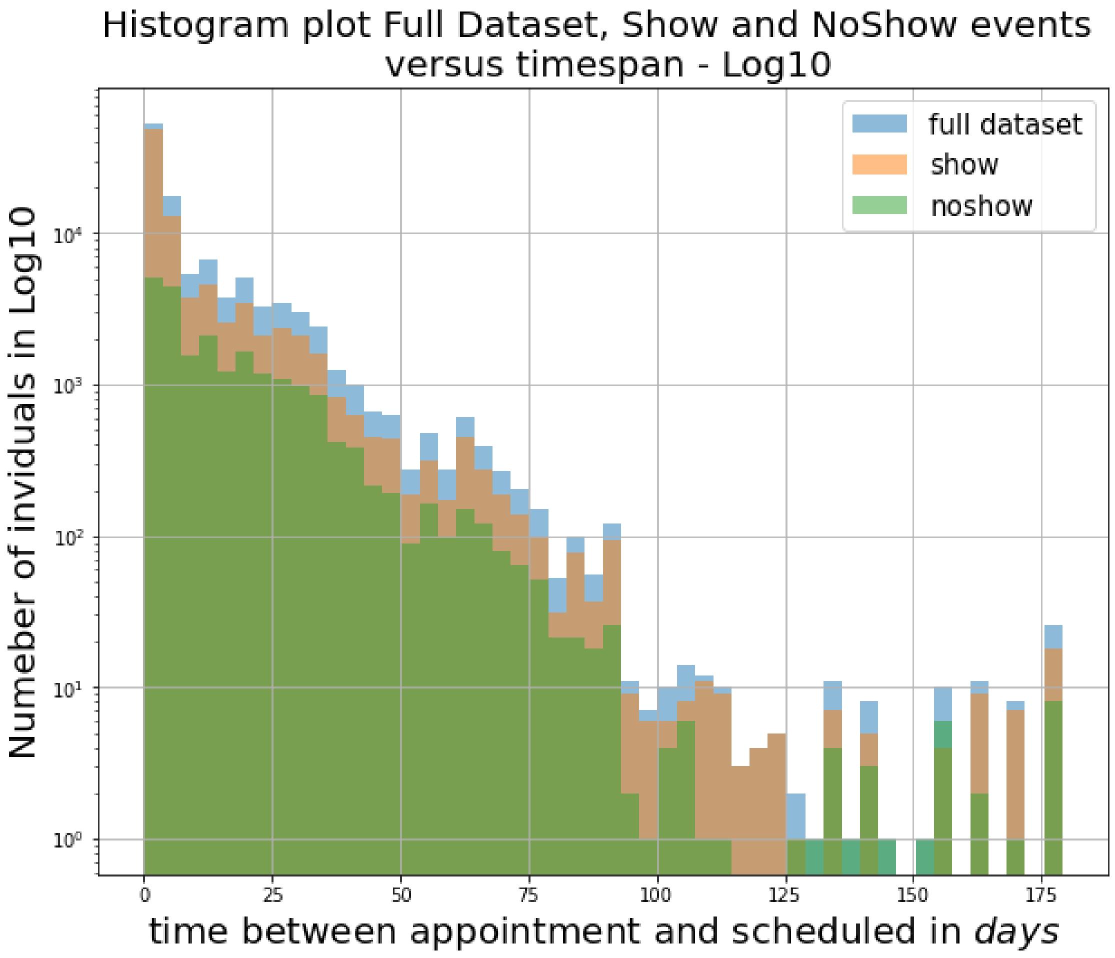
[32]:
```
df.timespan.hist(alpha=0.5,label='full dataset',bins=50,figsize=(10,8));
df[show].timespan.hist(alpha=0.5,label='show',bins=50);
df[noshow].timespan.hist(alpha=0.5,label='noshow',bins=50);
plt.xlabel(r'time between appointment and scheduled in $days$',fontsize=20);
plt.ylabel(r'Numeber of invividuals',fontsize=20)

plt.title("Histogram plot Full Dataset, Show and NoShow events \n versus␣
↪timespan",fontsize=20)

#plt.xlim([0,100])
plt.legend(fontsize=15);
```



Rescalling for Log10 in order to better visualize the histogram tail

[33]:
```
df.timespan.hist(alpha=0.5,label='full␣
↪dataset',log=True,bins=50,figsize=(10,8));
df[show].timespan.hist(alpha=0.5,label='show',log=True,bins=50);
df[noshow].timespan.hist(alpha=0.5,label='noshow',log=True,bins=50);
```

```
plt.xlabel(r'time between appointment and scheduled in $days$',fontsize=20);
plt.ylabel(r'Numeber of invividuals in Log10',fontsize=20)

plt.title("Histogram plot Full Dataset, Show and NoShow events \n versus␣
 ↪timespan - Log10",fontsize=20)

plt.legend(fontsize=15);
```



Observation: It is possible to observe that the avarege time distance between appointment and scheduled day (timespan) for Noshow events is sigificantly higher than show evants. Is possible to observe that the appointment timespan is important to determine a No-Show event.

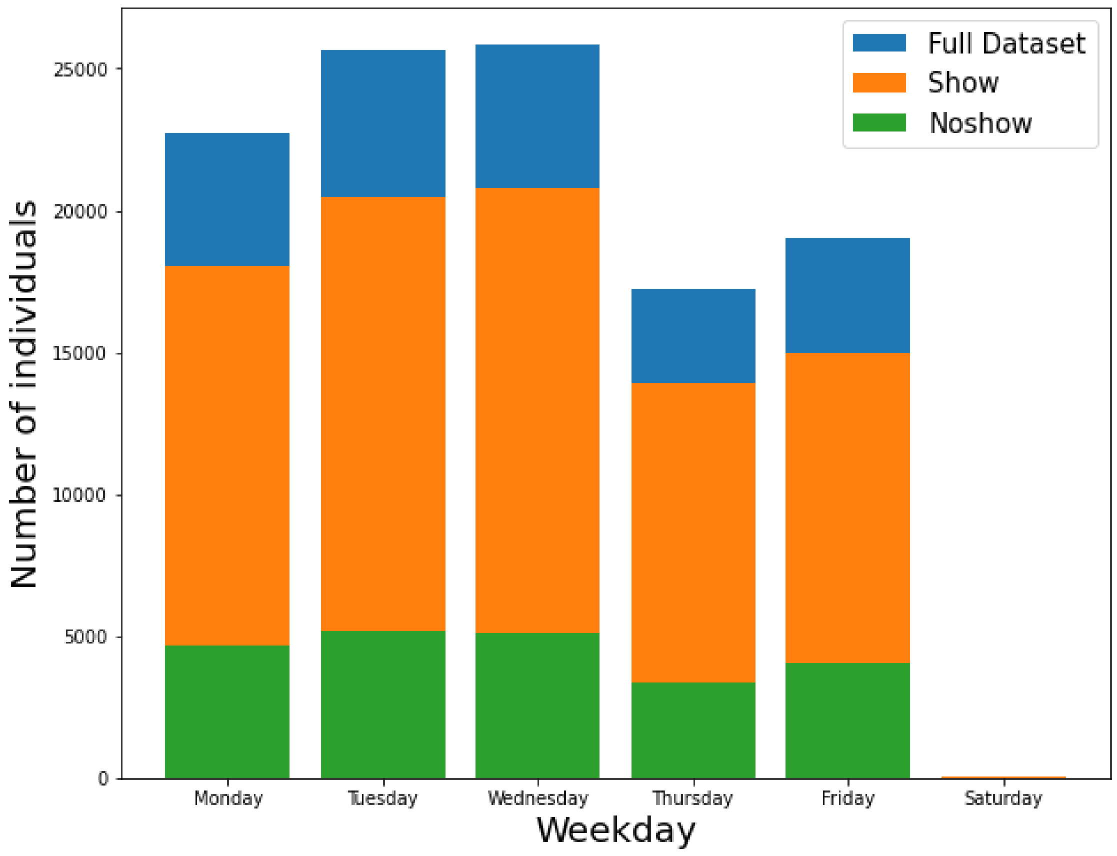
1.1.5 Research Question 3: Is there any relation between Appointment Day of the Week and no-show events.

[34]:
```
full_dataset = df.appdayofweek.value_counts()

show_dataset = df[show].appdayofweek.value_counts()

noshow_dataset = df[noshow].appdayofweek.value_counts()

xlabels = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday','Saturday']

xpos = ['0', '1', '2', '3', '4', '5']

#plt.title('Bar Graph - For week of the Day ')

#plt.legend();
```

[35]:
```
full_dataset
```

[35]:
```
2    25866
1    25638
0    22713
4    19019
3    17246
5       39
Name: appdayofweek, dtype: int64
```

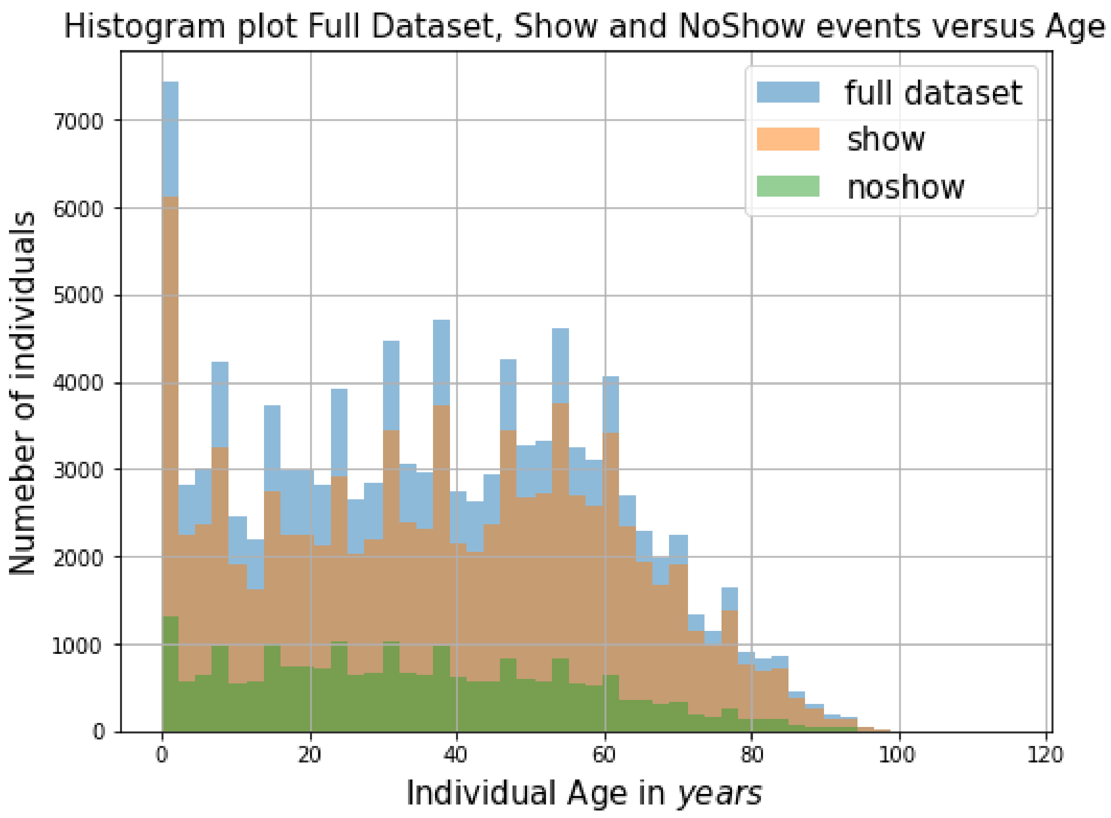
[36]:
```
plt.figure(figsize=(10,8));

loc1 =full_dataset.index
loc2 =show_dataset.index
loc3 =noshow_dataset.index

bar1 = full_dataset.values
bar2 = show_dataset.values
bar3 = noshow_dataset.values

Weekday={0:'Monday', 1:'Tuesday', 2:'Wednesday', 3:'Thursday', 4:'Friday', 5:
 ↪'Saturday', 6:'Sunday'}

plt.bar(loc1, bar1, tick_label=loc1.map(Weekday),label='Full Dataset');
plt.bar(loc2, bar2, tick_label=loc2.map(Weekday),label = 'Show');
plt.bar(loc3, bar3, tick_label=loc3.map(Weekday),label = 'Noshow');
plt.xlabel('Weekday',fontsize=20)
plt.ylabel('Number of individuals',fontsize=20)
plt.legend(fontsize=15);
```



Observation: It is not possible to visualize any preference for noShow events based on day of the week.

1.1.6 Research Question 4: Is there any relation between Age and no-show events.

[37]:
```
df.age.hist(alpha=0.5,label='full dataset',bins=50,figsize=(8,6));
df[show].age.hist(alpha=0.5,label='show',bins=50);
df[noshow].age.hist(alpha=0.5,label='noshow',bins=50);
plt.xlabel(r'Individual Age in $years$',fontsize=15);
plt.ylabel(r'Numeber of individuals',fontsize=15)
plt.title("Histogram plot Full Dataset, Show and NoShow events versus␣
 ↪Age",fontsize=15)
plt.legend(fontsize=15);
```



using describe to verify some statistics on Age

[38]:
```
df[show].age.describe()
```

[38]:
```
count    88207.000000
mean        37.790504
std         23.338645
min          0.000000
25%         18.000000
50%         38.000000
75%         56.000000
max        115.000000
Name: age, dtype: float64
```

[39]:
```
df[noshow].age.describe()
```

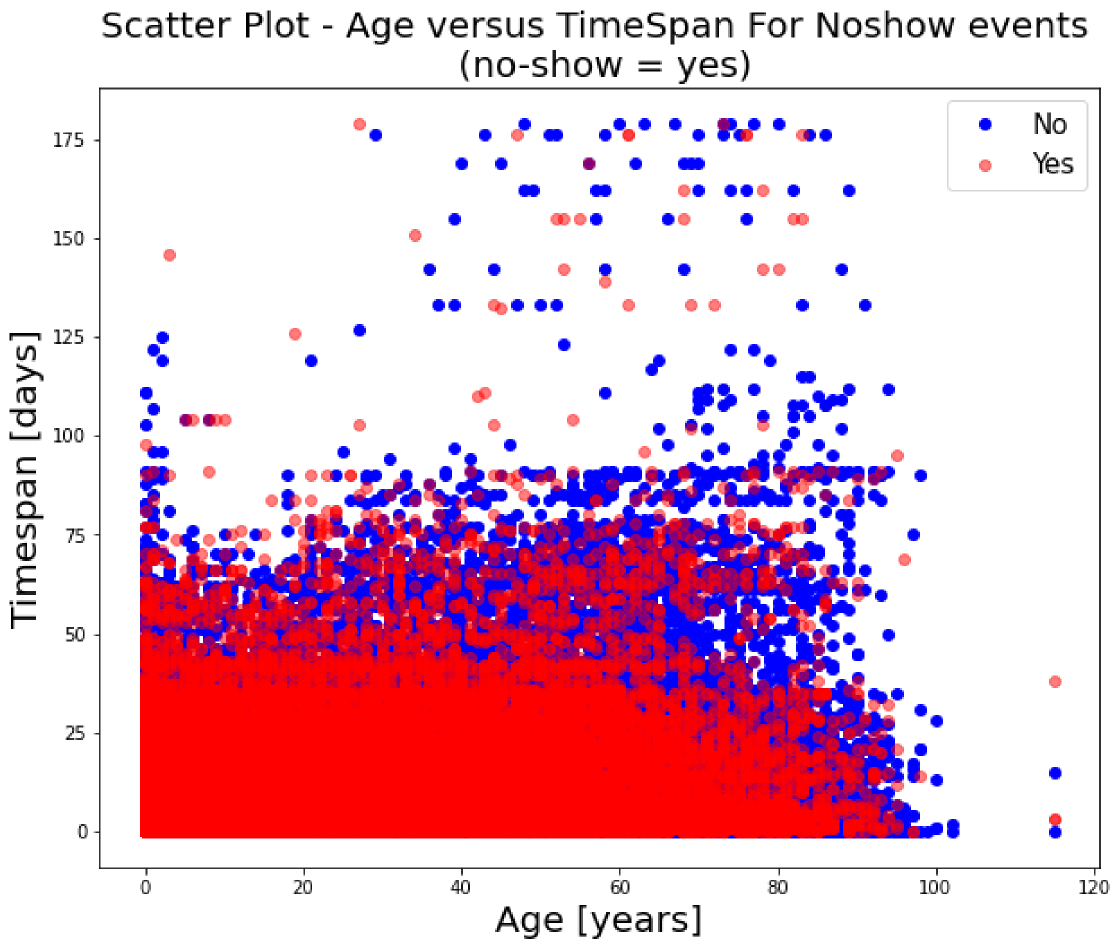
[39]:
```
count    22314.000000
mean        34.317872
std         21.965009
min          0.000000
25%         16.000000
```

```
50%          33.000000
75%          51.000000
max         115.000000
Name: age, dtype: float64
```

Observation: The Age distribution means is about the same for the No-Show and Show events. It is not possible to depict any prevalence on Age for the NoShow event

1.1.7 Research Question 5: Is there any relation between Age versus Timespan and no-show events?

[40]:
```python
groups = df.groupby('noshow')
i = 1
colors = {'Yes':'r', 'No':'b'}

#Function for scatter plotting
def plotting(x,y,i,names):
    ax.plot(x,y, marker='o', linestyle='',alpha=i,c=colors[names],label=names)
    ax.legend(fontsize=15)
    return

fig, ax = plt.subplots(figsize=(10,8))

for name, group in groups:
    plotting(group.age, group.timespan,i,name)
    i=i-0.5

plt.title('Scatter Plot - Age versus TimeSpan For Noshow events \n (no-show =␣
 ↪yes)',fontsize=20)
plt.xlabel('Age [years]',fontsize=20)
plt.ylabel('Timespan [days]',fontsize=20)

plt.show()
```



Observation: The Age versus timespan ploting do not help classify betweeen No-Show and Show events.

```
[ ]:
```

## Conclusions

Based on the dataset available, without considering the influence of same individual with multiple appointments or cross correlation between features. It is possible to observe for the No-Show event that:

1. There is no gender prevalence for the No-show event
2. Appointments with higher time distance between scheduled and appointment day is more prone to No-Show events
3. There is no prevalence of appointment day of the week for a No-Show event
4. The is no Age prevalence to a No-Show event.

References:

[1] https://agenciadenoticias.ibge.gov.br/agencia-sala-de-imprensa/2013-agencia-de-noticias/releases/29203-pns-2019-quem-mais-utiliza-o-sus-avaliou-mais-positivamente-a-qualidade-

dos-servicos-de-atencao-primaria-a-saude

```
[ ]:
```