2021 | By: Ahmed Abdelrahman Mohamed

Investigate No-Show Appointments Dataset

Investigate No-Show Appointments Dataset

Introduction

We are going to investigate No-Show Appointments Dataset that contains 110.527 medical appointments with its 14 associated variables.

Variables are:

1 - PatientId: Identification of a patient.

2 - AppointmentID: Identification of each appointment.

3 - Gender: Male or Female.

4 - ScheduledDay:The day someone called or registered the appointment.

5 - AppointmentDay:The day of the actual appointment, when they have to visit the doctor.

6 - Age: How old the patient is.

7 - Neighbourhood: Where the appointment takes place.

8 - Scholarship: True of False.

9 - Hipertension: True or False

10 - Diabetes: True or False

11- Alcoholism: True or False

12 - Handicap: The number of disabilities a person has. According to dataset creator https://www.kaggle.com/joniarroba/noshowappointments/discussion/29699

13- SMS_received: True or False.

14- No-show: True or False.

From the Dataset we can try to figure out the answer to some questions such as:

1- What gender has greater appointments?

2- Which Neighborhood has greater appointments?

3- What is the average age of people having Diabetes?

4- What is the average age of people having Hipertension?

5- Which Neighborhood has greater percent of people having Diabetes, Hipertension or Alcoholism?

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

7- What are the insights that we can conclude from data of people with chronic diseases like Hipertension and Diabetes?

I have chosen two questions for analysis which are:

1- What are the insights that we can conclude from data of people with chronic diseases like Hipertension and Diabetes?

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

Dataset Source

Dataset is available on [Kaggle] as .csv file.

Investigation Steps

1. **Data Assessing**
2. **Data Cleaning**
3. **Data analysis and Visualization.**
4. **Conclusion.**

Required libraries

Importing Libraries

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

Data Gathering

As we mentioned before we can download the dataset as .csv file from Kaggle.

In [2]:
```python
df = pd.read_csv("noshowappointments-kagglev2-may-2016.csv")
```

Data Assessing

Visual Assessing

In [3]:
```python
df.sample(10)
```

Out[3]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
62348	9.439411e+14	5641216	F	2016-04-29T12:55:13Z	2016-05-04T00:00:00Z	36	JABOUR	0	0	0	0	0	1	No
78209	7.565876e+13	5691411	M	2016-05-12T12:40:46Z	2016-05-12T00:00:00Z	45	SANTA MARTHA	0	0	0	1	0	0	No
38906	2.731816e+14	5615685	F	2016-04-25T12:53:58Z	2016-05-11T00:00:00Z	61	VILA RUBIM	0	1	1	0	1	1	Yes
49297	9.939218e+12	5673999	M	2016-05-09T10:14:55Z	2016-05-12T00:00:00Z	3	JARDIM DA PENHA	0	0	0	0	0	0	Yes
51101	1.555819e+13	5653005	F	2016-05-03T11:03:52Z	2016-05-18T00:00:00Z	3	CENTRO	0	0	0	0	0	0	Yes
56249	2.259192e+14	5714702	F	2016-05-18T12:51:03Z	2016-05-30T00:00:00Z	49	DA PENHA	0	1	1	0	0	1	No
24301	8.959992e+08	5605419	M	2016-04-20T08:58:50Z	2016-05-18T00:00:00Z	36	SANTO ANDRÉ	0	0	0	1	0	0	Yes
88867	8.668637e+13	5664069	M	2016-05-05T11:06:28Z	2016-06-03T00:00:00Z	23	SANTO ANTÔNIO	0	0	0	0	0	1	No
91550	8.288797e+14	5733094	F	2016-05-24T13:20:35Z	2016-06-07T00:00:00Z	44	CONSOLAÇÃO	0	0	0	0	0	0	Yes
61717	5.399321e+12	5685696	F	2016-05-11T10:56:28Z	2016-05-11T00:00:00Z	25	ITARARÉ	0	0	0	0	0	0	No

Programmatic Assessing

To get more descriptive information about data quality and tidiness.

```
In [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
```

```
In [10]:  df.nunique()

Out[10]: 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
```

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

Out[6]:

	PatientId	AppointmentID	Age	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received
count	1.105270e+05	1.105270e+05	110527.000000	110527.000000	110527.000000	110527.000000	110527.000000	110527.000000	110527.000000
mean	1.474963e+14	5.675305e+06	37.088874	0.098266	0.197246	0.071865	0.030400	0.022248	0.321026
std	2.560949e+14	7.129575e+04	23.110205	0.297675	0.397921	0.258265	0.171686	0.161543	0.466873
min	3.921784e+04	5.030230e+06	-1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.172614e+12	5.640286e+06	18.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	3.173184e+13	5.680573e+06	37.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	9.439172e+13	5.725524e+06	55.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
max	9.999816e+14	5.790484e+06	115.000000	1.000000	1.000000	1.000000	1.000000	4.000000	1.000000

```
In [11]:  df["Age"].value_counts().sort_index()

Out[11]: -1        1
          0     3539
          1     2273
          2     1618
          3     1513
                ...
          98       6
          99       1
          100      4
          102      2
          115      5
         Name: Age, Length: 104, dtype: int64
```

```
In [12]:  df["Handcap"].value_counts().sort_index()

Out[12]: 0    108286
         1      2042
         2       183
         3        13
         4         3
         Name: Handcap, dtype: int64
```

Note:

The handicap refers to the number of disabilities a person has. This is according to the dataset creator.

Assessment Summary

1- No Missing values found.

2- Wrong data in (Age) column as the min value is (-1).

3- Inappropriate data types in AppointmentDay and ScheduledDay columns

4- Wrong column name (Handcap).

4- Some columns should be added to aid the analysis.

Data Cleaning and preparing

1- **Drop values less than 0 in (Age) column**

```
In [16]:  # Drop values less than 0 in (Age) column
          df = df[df["Age"]>= 0]
```

```
In [17]:  #check
          df["Age"].value_counts().sort_index()

Out[17]:  0      3539
          1      2273
          2      1618
          3      1513
          4      1299
                 ...
          98        6
          99        1
          100       4
          102       2
          115       5
          Name: Age, Length: 103, dtype: int64
```

2- Change AppointmentDay and ScheduledDay columns datatype to datetime.

In [10]:
```
#CODE
df["ScheduledDay"] = pd.to_datetime(df["ScheduledDay"])
df["AppointmentDay"] = pd.to_datetime(df["AppointmentDay"])
```

In [11]:
```
#check
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 110526 entries, 0 to 110526
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   PatientId       110526 non-null  float64
 1   AppointmentID   110526 non-null  int64
 2   Gender          110526 non-null  object
 3   ScheduledDay    110526 non-null  datetime64[ns, UTC]
 4   AppointmentDay  110526 non-null  datetime64[ns, UTC]
 5   Age             110526 non-null  int64
 6   Neighbourhood   110526 non-null  object
 7   Scholarship     110526 non-null  int64
 8   Hipertension    110526 non-null  int64
 9   Diabetes        110526 non-null  int64
 10  Alcoholism      110526 non-null  int64
 11  Handcap         110526 non-null  int64
 12  SMS_received    110526 non-null  int64
 13  No-show         110526 non-null  object
dtypes: datetime64[ns, UTC](2), float64(1), int64(8), object(3)
memory usage: 12.6+ MB
```

3- Add 5 new columns to the dataframe that will help us in the analysis.

In [12]:
```
#CODE
df["Waiting_time"] =  df["AppointmentDay"]- df["ScheduledDay"]
df["appointment_day"] = df["AppointmentDay"].dt.day_name()
df["appointment_month"] = df["AppointmentDay"].dt.month_name()
df["Waiting_time_days"] = df["Waiting_time"].dt.days + df["Waiting_time"].dt.seconds/(3600*24)
df["appointment_daynum"] = df["AppointmentDay"].dt.day
```

In [13]:
```
#check
df.sample(5)
```

Out[13]:

ertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show	Waiting_time	appointment_day	appointment_month	Waiting_time_days	appointment_daynum
0	0	0	0	1	Yes	3 days 15:20:27	Friday	April	3.639201	29
0	0	0	0	0	No	-1 days +16:33:57	Thursday	May	-0.309757	5
0	0	0	0	1	No	21 days 15:50:56	Tuesday	May	21.660370	10
0	0	0	0	0	No	-1 days +11:30:55	Monday	May	-0.520197	30
0	0	1	0	0	No	-1 days +15:52:58	Friday	April	-0.338218	29

Note:

Another quality issue appeared as Negative values appeared in (Waiting_time) column where "ScheduledDay" is after "AppointmentDay".

In [14]:
```
#check
df["Waiting_time"].value_counts().sort_index()
```

Out[14]:
```
-7 days +10:10:40    1
-2 days +09:09:19    1
-2 days +10:16:02    1
-2 days +13:08:07    1
-2 days +17:09:03    1
                    ..
178 days 13:16:26    1
178 days 13:16:43    1
178 days 13:16:59    1
178 days 13:17:18    1
178 days 13:19:01    1
Name: Waiting_time, Length: 89711, dtype: int64
```

4- Drop wrong values from the Dataframe.

In [15]:
```
#CODE
# slicing only right values
df = df[df["ScheduledDay"]< df["AppointmentDay"]]
```

In [16]:
```
#check
df["Waiting_time"].value_counts().sort_index()
```

Out[16]:
```
0 days 03:16:20      1
0 days 03:19:13      1
0 days 03:36:54      1
0 days 03:37:24      1
0 days 03:39:51      1
                    ..
178 days 13:16:26    1
178 days 13:16:43    1
178 days 13:16:59    1
178 days 13:17:18    1
178 days 13:19:01    1
Name: Waiting_time, Length: 67588, dtype: int64
```

In [17]:
```
df["Waiting_time_days"].value_counts().sort_index()
```

Out[17]:
```
0.136343      1
0.138345      1
0.150625      1
0.150972      1
0.152674      1
             ..
178.553079    1
178.553275    1
178.553461    1
178.553681    1
178.554873    1
Name: Waiting_time_days, Length: 67588, dtype: int64
```

5- Rename wrong column's name.

In [18]:
```
#CODE
df.rename(columns = {"Handcap":"Handicap"},inplace = True)
```

In [19]:
```
#check
df.head(1)
```

Out[19]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handicap
5	9.598513e+13	5626772	F	2016-04-27 08:36:51+00:00	2016-04-29 00:00:00+00:00	76	REPÚBLICA	0	1	0	0	0

6- Reset index

In [20]:
```
#code
df = df.reset_index(drop = True)
```

In [21]:
```
#check
df.head()
```

Out[21]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood
0	9.598513e+13	5626772	F	2016-04-27 08:36:51+00:00	2016-04-29 00:00:00+00:00	76	REPÚBLICA
1	7.336882e+14	5630279	F	2016-04-27 15:05:12+00:00	2016-04-29 00:00:00+00:00	23	GOIABEIRAS
2	3.449833e+12	5630575	F	2016-04-27 15:39:58+00:00	2016-04-29 00:00:00+00:00	39	GOIABEIRAS
3	7.812456e+13	5629123	F	2016-04-27 12:48:25+00:00	2016-04-29 00:00:00+00:00	19	CONQUISTA
4	7.345362e+14	5630213	F	2016-04-27 14:58:11+00:00	2016-04-29 00:00:00+00:00	30	NOVA PALESTINA

7- Create show and noshow dataframes

In [22]:
```
show_df = df[df["No-show"] == "No"]
noshow_df = df[df["No-show"] == "Yes"]
```

Data Analysis and Visualization

Research Question 1

What are the insights that we can conclude from data of people with chronic diseases like Hipertension and Diabetes?

- **Create a Dataframe that only contains people with chronic diseases [Diabetes or Hipertension or Both]**

In [23]:
```
#create a dataframe that only contains people with chronic diseases[Diabetes or Hipertension or Both]
chronic_df = df[(df["Diabetes"]==1)|(df["Hipertension"]==1)]
chronic_df.head()
```

Out[23]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Ha
0	9.598513e+13	5626772	F	2016-04-27 08:36:51+00:00	2016-04-29 00:00:00+00:00	76	REPÚBLICA	0	1	0	0	
17	5.819370e+12	5624020	M	2016-04-26 15:04:17+00:00	2016-04-29 00:00:00+00:00	46	CONQUISTA	0	1	0	0	
22	5.873316e+12	5609446	M	2016-04-20 15:54:18+00:00	2016-04-29 00:00:00+00:00	85	SÃO CRISTÓVÃO	0	1	0	0	
24	8.224325e+14	5633339	F	2016-04-28 09:20:36+00:00	2016-04-29 00:00:00+00:00	71	MARUÍPE	0	0	1	0	
26	2.741649e+11	5635414	F	2016-04-28 13:27:27+00:00	2016-04-29 00:00:00+00:00	78	SÃO CRISTÓVÃO	0	1	1	0	

1.1 Which chronic disease is more prevalent?

In [24]:
```
Hipertension_details = df.groupby(["Hipertension"]).describe()
Hipertension_details["Age"]
```

Out[24]:

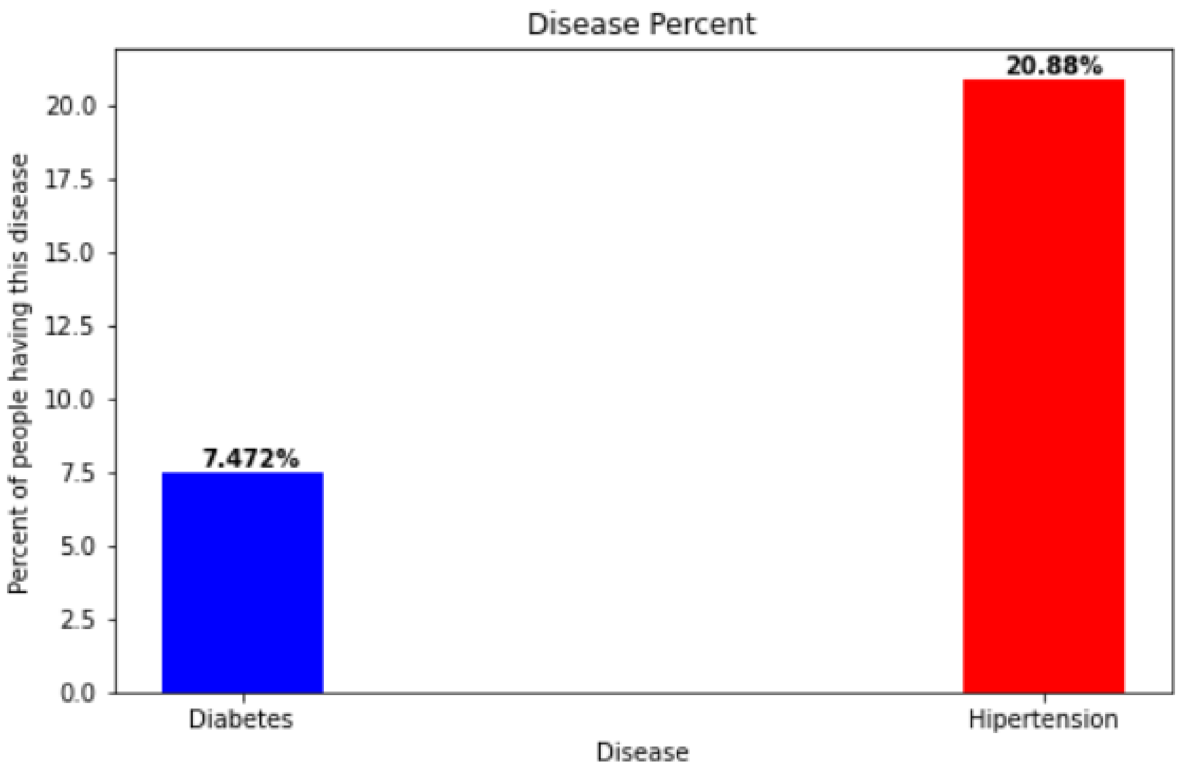
	count	mean	std	min	25%	50%	75%	max
Hipertension								
0	56927.0	32.603492	21.163359	0.0	15.0	31.0	49.0	115.0
1	15032.0	60.842669	13.744876	4.0	52.0	61.0	70.0	115.0

In [25]:
```
Diabetes_details = df.groupby(["Diabetes"]).describe()
Diabetes_details["Age"]
```

Out[25]:

	count	mean	std	min	25%	50%	75%	max
Diabetes								
0	66582.0	36.649230	22.525341	0.0	18.0	36.0	54.0	115.0
1	5377.0	61.451925	13.474650	1.0	53.0	62.0	70.0	98.0

In [47]:
```
percentage = [len(df[df["Diabetes"] ==1])*100/len(df),len(df[df["Hipertension"] ==1])*100/len(df)]
x = np.arange(2)
fig, ax = plt.subplots(figsize = (8,5))
plt.bar(x,percentage,width = 0.2,color = ["b","r"])
plt.xticks(x, ["Diabetes",'Hipertension'])
plt.text(x[0]-.05,percentage[0]+.2,str(percentage[0])[:5]+"%",fontweight ='bold')
plt.text(x[1]-.05,percentage[1]+.2,str(percentage[1])[:5]+"%",fontweight ='bold')
plt.title("Disease Percent")
plt.xlabel("Disease")
plt.ylabel("Percent of people having this disease");
```



We can see that Hipertension is more Prevalent than Diabetes.

1.2 What is the effect of Alcoholism on having chronic disease?

- **Effect of Alcoholism on having Hipertension.**

In [27]:

```
#Effect of Alcoholism on having Hipertension
Alcoholism_Hipertension = df.groupby(["Alcoholism","Hipertension"]).describe()
Alcoholism_Hipertension["Age"]
```

Out[27]:

Alcoholism	Hipertension	count	mean	std	min	25%	50%	75%	max
0	0	55880.0	32.370025	21.216680	0.0	15.0	31.0	49.0	115.0
	1	14257.0	61.078347	13.876518	4.0	52.0	61.0	70.0	115.0
1	0	1047.0	45.063992	13.010538	4.0	36.0	46.0	54.0	81.0
	1	775.0	56.507097	10.116742	26.0	50.0	57.0	63.0	85.0

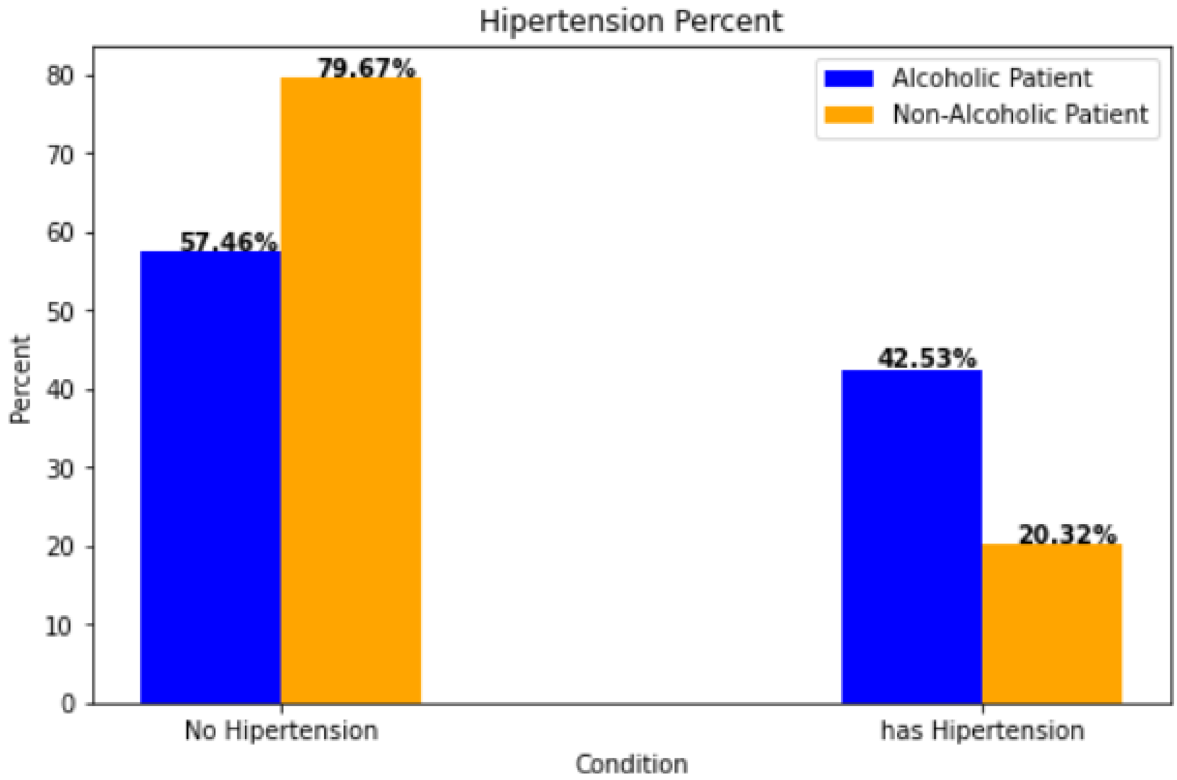
In [28]:

```
NoAlcoholism_percentage = [Alcoholism_Hipertension["Age"]["count"][0][0]*100/
                            sum(Alcoholism_Hipertension["Age"]["count"][0]),
                          Alcoholism_Hipertension["Age"]["count"][0][1]*100/
                            sum(Alcoholism_Hipertension["Age"]["count"][0])]

Alcoholism_percentage = [Alcoholism_Hipertension["Age"]["count"][1][0]*100/
                          sum(Alcoholism_Hipertension["Age"]["count"][1]),
                          Alcoholism_Hipertension["Age"]["count"][1][1]*100/
                          sum(Alcoholism_Hipertension["Age"]["count"][1])]


x = np.arange(2)
fig, ax = plt.subplots(figsize = (8,5))
plt.bar(x-.1,Alcoholism_percentage,width = 0.2,color = ["b"])
plt.bar(x+.1,NoAlcoholism_percentage,width = 0.2,color = ["orange"])

plt.xticks(x, ["No Hipertension",'has Hipertension'])
plt.text(x[0]-.15,Alcoholism_percentage[0]+.1,str(Alcoholism_percentage[0])[:5]+"%",fontweight ='bold')
plt.text(x[1]-.15,Alcoholism_percentage[1]+.1,str(Alcoholism_percentage[1])[:5]+"%",fontweight ='bold')
plt.text(x[0]+.05,NoAlcoholism_percentage[0]+.1,str(NoAlcoholism_percentage[0])[:5]+"%",fontweight ='bold')
plt.text(x[1]+.05,NoAlcoholism_percentage[1]+.1,str(NoAlcoholism_percentage[1])[:5]+"%",fontweight ='bold')
plt.title("Hipertension Percent")
plt.ylabel("Percent");
plt.legend(["Alcoholic Patient","Non-Alcoholic Patient"]);
```



We can see that about 42.5% of Alcoholic Patients suffer Hipertension compared to only 20.3% of Non-Alcoholic Patients.

➢ Effect of Alcoholism on having Diabetes

In [48]:

```
#Effect of Alcoholism on having Diabetes
Alcoholism_Diabetes = df.groupby(["Alcoholism","Diabetes"]).describe()
Alcoholism_Diabetes["Age"]
```

Out[48]:

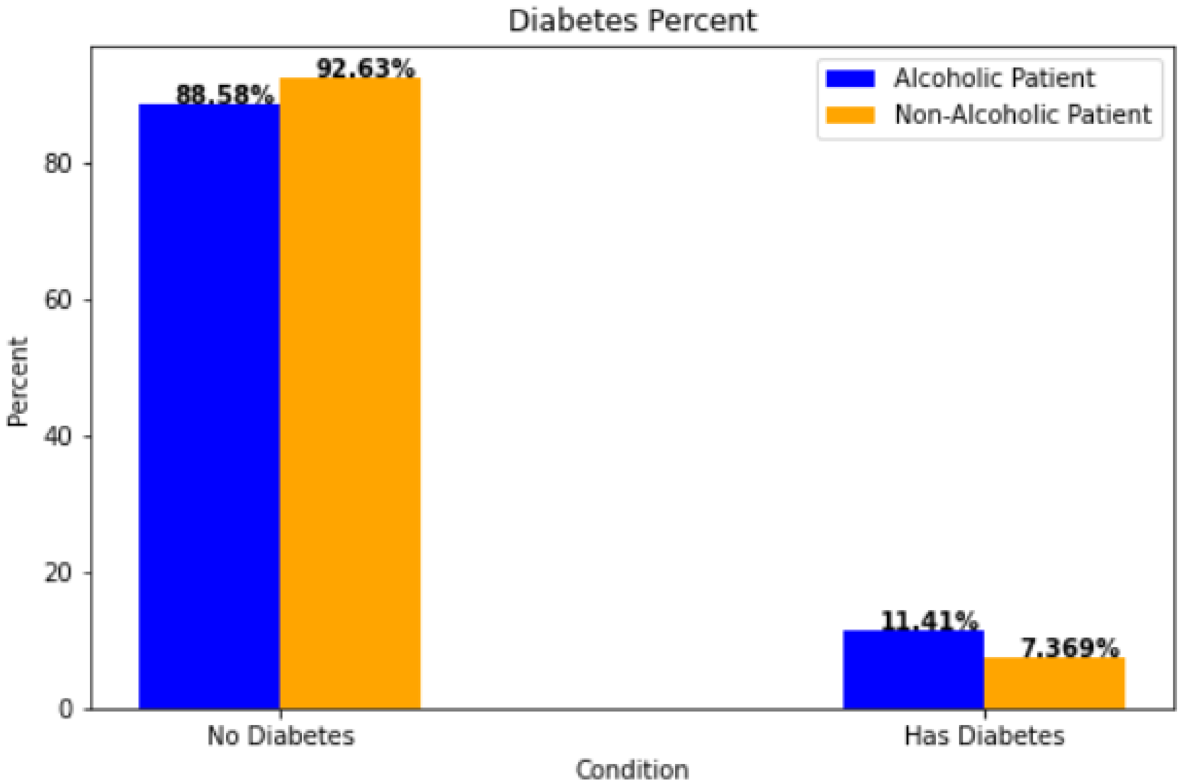
Alcoholism	Diabetes	count	mean	std	min	25%	50%	75%	max
0	0	64968.0	36.347279	22.626989	0.0	18.0	36.0	54.0	115.0
	1	5169.0	61.563358	13.588287	1.0	53.0	62.0	70.0	98.0
1	0	1614.0	48.803594	13.090338	4.0	40.0	50.0	57.0	85.0
	1	208.0	58.682692	9.881681	28.0	54.0	58.0	66.0	84.0

In [30]:

```
Alcoholism_percentage = [Alcoholism_Diabetes["Age"]["count"][1][0]*100/
                         sum(Alcoholism_Diabetes["Age"]["count"][1]),
                         Alcoholism_Diabetes["Age"]["count"][1][1]*100/
                         sum(Alcoholism_Diabetes["Age"]["count"][1])]

NoAlcoholism_percentage = [Alcoholism_Diabetes["Age"]["count"][0][0]*100/
                           sum(Alcoholism_Diabetes["Age"]["count"][0]),
                         Alcoholism_Diabetes["Age"]["count"][0][1]*100/
                           sum(Alcoholism_Diabetes["Age"]["count"][0])]
x = np.arange(2)
fig, ax = plt.subplots(figsize = (8,5))
plt.bar(x-.1,Alcoholism_percentage,width = 0.2,color = ["b"])
plt.bar(x+.1,NoAlcoholism_percentage,width = 0.2,color = ["orange"])

plt.xticks(x, ["No Diabetes",'Has Diabetes'])
plt.text(x[0]-.15,Alcoholism_percentage[0]+.1,str(Alcoholism_percentage[0])[:5]+"%",fontweight ='bold')
plt.text(x[1]-.15,Alcoholism_percentage[1]+.1,str(Alcoholism_percentage[1])[:5]+"%",fontweight ='bold')
plt.text(x[0]+.05,NoAlcoholism_percentage[0]+.1,str(NoAlcoholism_percentage[0])[:5]+"%",fontweight ='bold')
plt.text(x[1]+.05,NoAlcoholism_percentage[1]+.1,str(NoAlcoholism_percentage[1])[:5]+"%",fontweight ='bold')
plt.title("Diabetes Percent")
plt.ylabel("Percent");
plt.legend(["Alcoholic Patient","Non-Alcoholic Patient"]);
```



We can see about 11.4% of Alcoholic Patients suffer Diabetes compared to only 7.3% for Non-Alcoholic Patients.

So, we can conclude this:

1- The probability to get Hipertension are twice for Alcoholic Patients compared to non- Alcoholic Patients

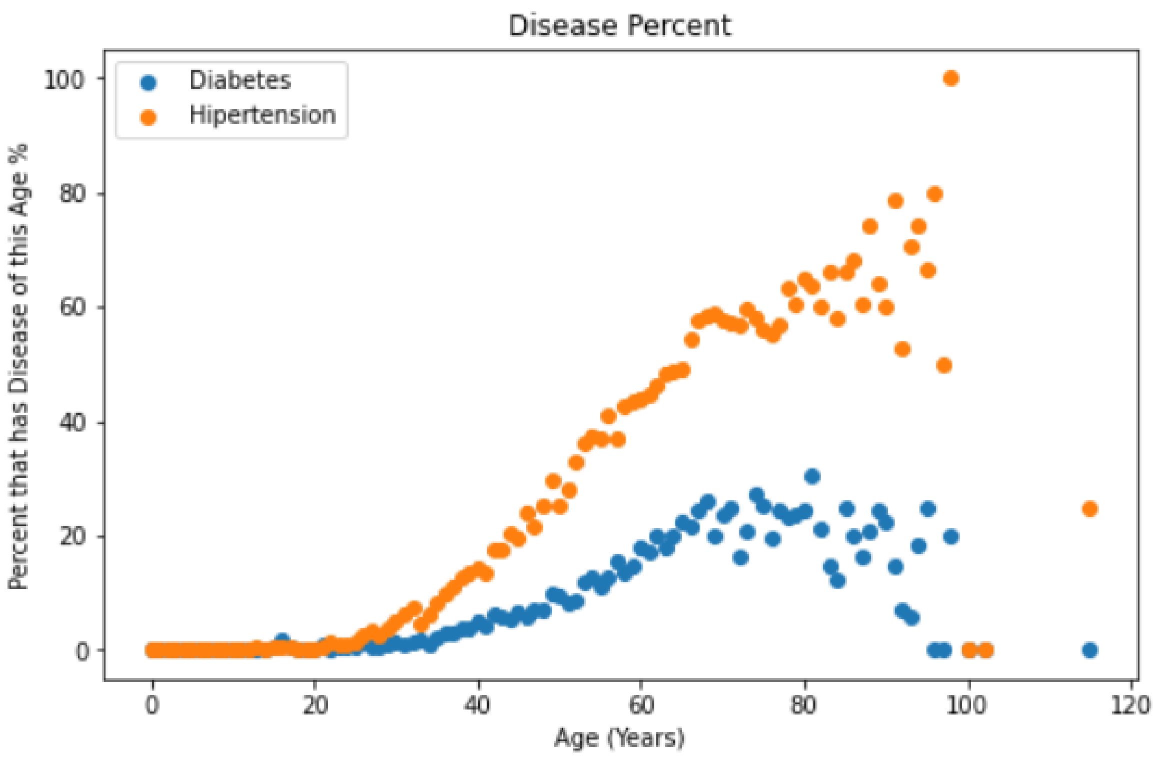
2- The probability to get Diabetes are one and half for Alcoholic Patients compared to non- Alcoholic Patients.

1.3 What is the effect of age on having chronic disease?

In [31]:

```
#Plot Age VS The Percent of of people in this age who suffer Diabetes and Hipertension
Diabetes_percent = (chronic_df[chronic_df["Diabetes"]==1]["Age"].value_counts()/
                    df["Age"].value_counts()).fillna(0)*100
Hipertension_percent= (chronic_df[chronic_df["Hipertension"]==1]["Age"].value_counts()/
                       df["Age"].value_counts()).fillna(0)*100

#fill nan values with zero as nan values means that Noone of this age has diabetes.
fig, ax = plt.subplots(figsize = (8,5))
plt.scatter(Diabetes_percent.index,Diabetes_percent.values)
plt.scatter(Hipertension_percent.index,Hipertension_percent.values)
plt.title("Disease Percent")
plt.xlabel("Age (Years)")
plt.ylabel("Percent that has Disease of this Age %")
plt.legend(["Diabetes","Hipertension"]);
```



We can see that:

1- As noticed before, the percent of people with Hipertension is more than those with Diabetes.

2- The ages from Mid-20th to Mid-30th is considered the beginning to get a chronic disease.

3- Getting older increase the risk of getting a chronic disease.

4- For people with about 65 years old and older, the risk of getting diabetes becomes some kind constant or less than younger patients.

5- Pervious note makes no sense, so those older patients should be examined and surveys about their food and daily routine should be conducted.

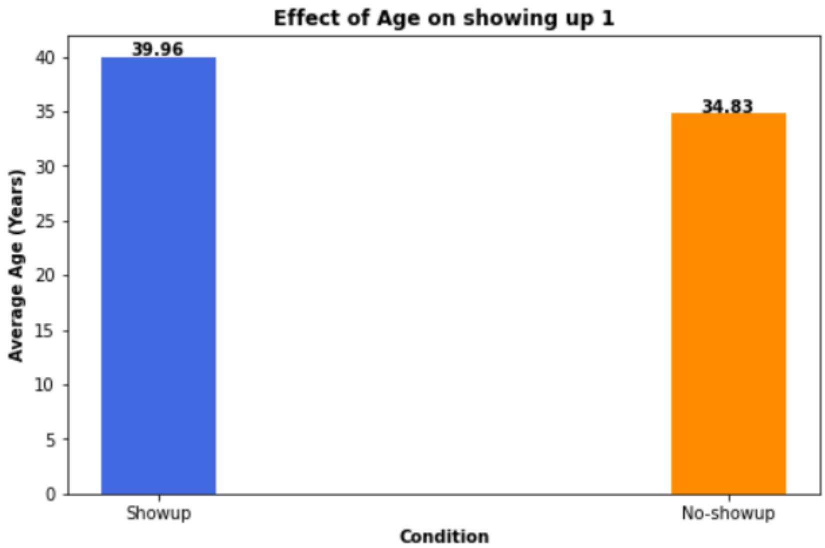
Research Question 2

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

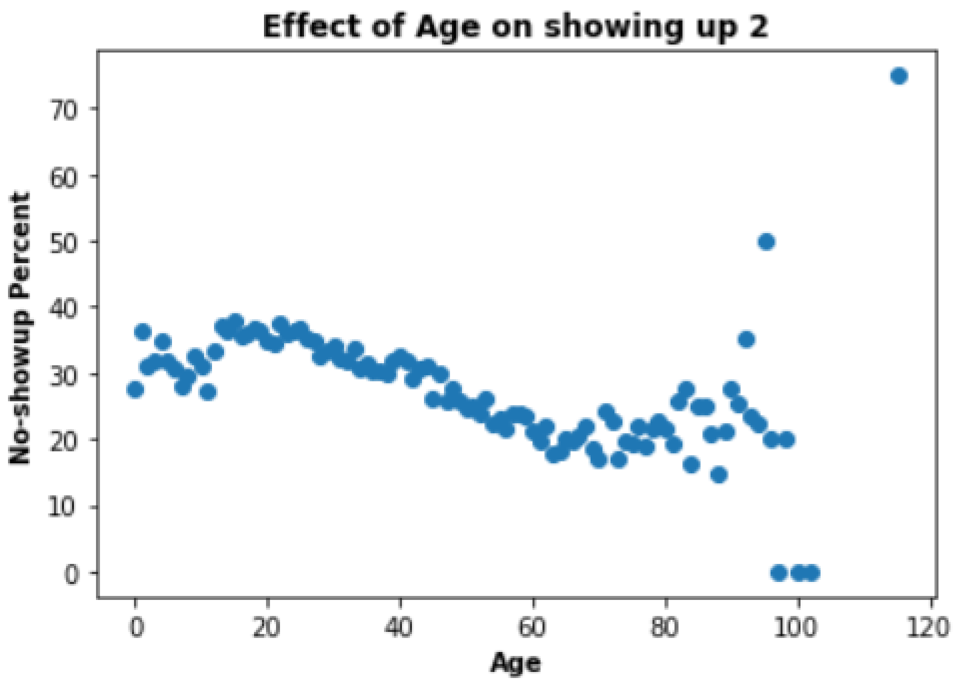
2.1 What is the effect of Age on probability of showing up?

In [87]:

```
#Get the average age of people who showed up and those who didn't.
fig, ax = plt.subplots(figsize = (8,5))
plt.title("Effect of Age on showing up 1",fontweight ='bold')
plt.xlabel("Condition",fontweight ='bold')
plt.ylabel("Average Age (Years)",fontweight ='bold')
x = np.arange(2)
average = df.groupby("No-show").mean()["Age"].values
ax.bar(x,average,color=["royalblue","darkorange"] ,width = .2)
plt.xticks(x, ["Showup",'No-showup'])
ax.text(x[0]-.05,average[0]+.1,str(average[0])[:5],fontweight ='bold')
ax.text(x[1]-.05,average[1]+.1,str(average[1])[:5],fontweight ='bold');
```



In [34]:
```
#percent of each age that didn't show up
Age_percent = (noshow_df["Age"].value_counts().sort_index()/df["Age"].value_counts().sort_index()).fillna(0)*100
plt.title("Effect of Age on showing up 2",fontweight ='bold')
plt.xlabel("Age",fontweight ='bold')
plt.ylabel("No-showup Percent",fontweight ='bold')
plt.scatter(Age_percent.index,Age_percent.values);
```



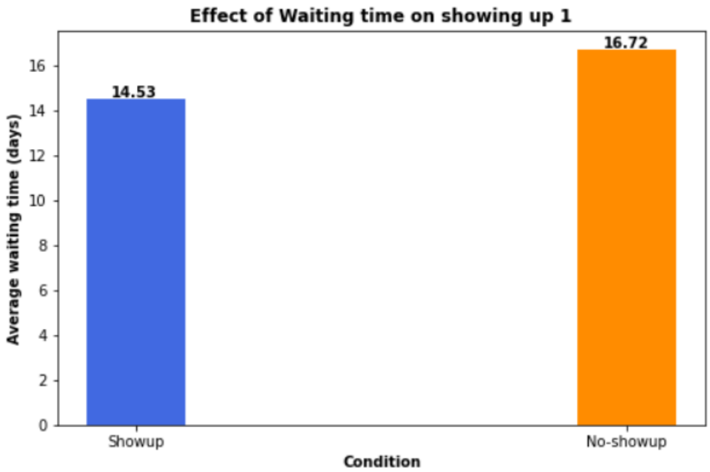
We can see that:

1- Average age for people who showed-up(about 40 years) is higher than for those who didn't(about 35 years).
2- In general, the higher the age is, the lower percentage of people who didn't show-up.

2.2 What is the effect of Waiting time on probability of showing up?

➢ **Get the average Waiting time between registering and the appointment of people who showed up and those who didn't.**

In [35]:
```
#Get the average Waiting time between registering and the appointment of people who showed up and those who didn't.
fig, ax = plt.subplots(figsize = (8,5))
plt.title("Effect of Waiting time on showing up 1",fontweight ='bold')
plt.xlabel("Condition",fontweight ='bold')
plt.ylabel("Average waiting time (days)",fontweight ='bold')
x = np.arange(2)
average = df.groupby("No-show").mean()["Waiting_time_days"].values
ax.bar(x,average,color=["royalblue","darkorange"] ,width = .2)
plt.xticks(x, ["Showup",'No-showup'])
ax.text(x[0]-.05,average[0]+.1,str(average[0])[:5],fontweight ='bold')
ax.text(x[1]-.05,average[1]+.1,str(average[1])[:5],fontweight ='bold');
```



➢ **Group Waiting time data by value ranges to plot each Waiting time range VS. the percentage of people who didn’t show up in this range.**

In [36]:
```
#group Waiting time data by value ranges
#to plot each Waiting time range VS the percentage of people who did't show up in this range.
bins = np.arange(0,150,5)
noshow_counts= pd.cut(noshow_df["Waiting_time_days"],bins).value_counts()
total_counts = pd.cut(df["Waiting_time_days"],bins).value_counts()
percentage = (noshow_counts/total_counts)*100
percentage
```

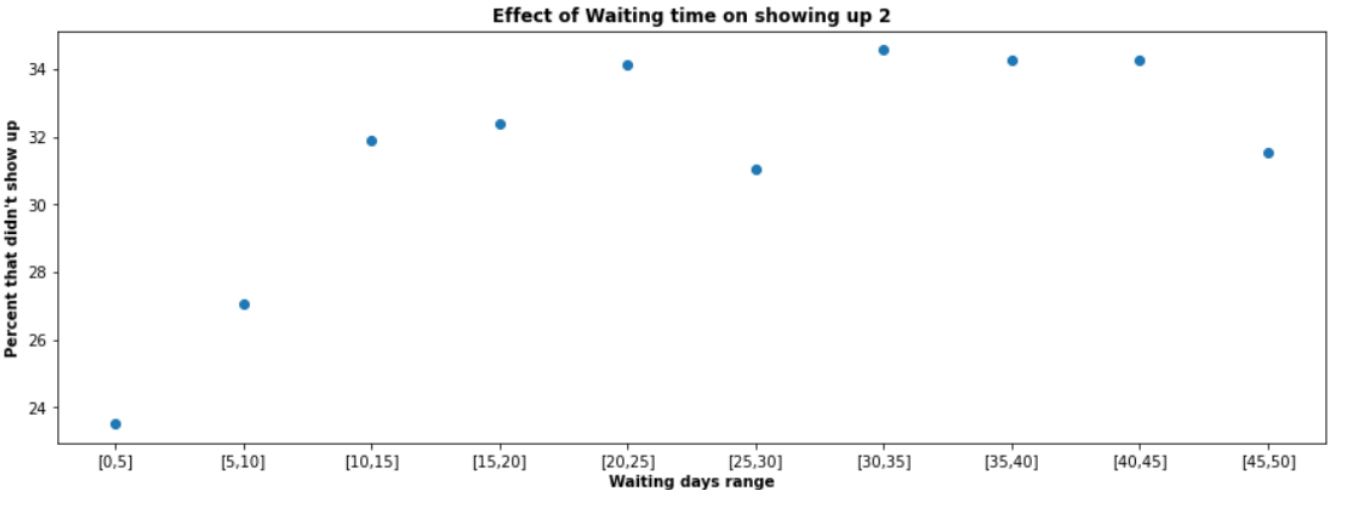
Out[36]:
```
(0, 5]        23.500559
(5, 10]       27.047859
(10, 15]      31.902439
(15, 20]      32.377495
(20, 25]      34.134897
(25, 30]      31.069998
(30, 35]      34.578933
(35, 40]      34.285714
(40, 45]      34.276970
(45, 50]      31.543624
```

In [37]:
```
#get the weight(number of observations) of each range in the dataset
weight = (total_counts/len(df))*100
weight
```

Out[37]:
```
(0, 5]      32.298948
(5, 10]     19.832127
(10, 15]    11.395378
(15, 20]     7.657138
(25, 30]     7.286093
(20, 25]     7.108214
(30, 35]     5.132089
(35, 40]     2.091469
(40, 45]     1.816312
(45, 50]     1.035312
(60, 65]     0.972776
(55, 60]     0.797676
(65, 70]     0.711516
```

So, we can only take waiting time ranges from 0 to 50 to get accurate explanation as other ranges percentages in dataset are very small and can be misleading.

In [38]:
```
locations = []
c = 0
for i in range(len((noshow_counts/total_counts).index)):
    locations.append("[{},{}]".format(c,c+5))
    c+=5
fig, ax = plt.subplots(figsize = (15,5))
plt.scatter(locations[:10],((noshow_counts/total_counts)*100).values[:10])
plt.title("Effect of Waiting time on showing up 2",fontweight ='bold')
plt.xlabel("Waiting days range",fontweight ='bold')
plt.ylabel("Percent that didn't show up",fontweight ='bold');
```



We can see that:

1- The longer the waiting time is, the higher percentage that didn't show up.
2- Average waiting time for patients who didn't show up is 16.72 days compared to 14.5 for those who showed up.

2.3 What is the effect of Scholarship, Hipertension, Diabetes, Alcoholism, SMS_received on probability of showing up?

We can do this for all categorical variables in the same way. Following next steps to get the percentage of people who didn’t show up of patients having this variable (Scholarship for example") and the percentage of people who didn’t show up of patients who don’t have this variable, then compare the two percentages to figure out the effect of this variable.

In [39]:

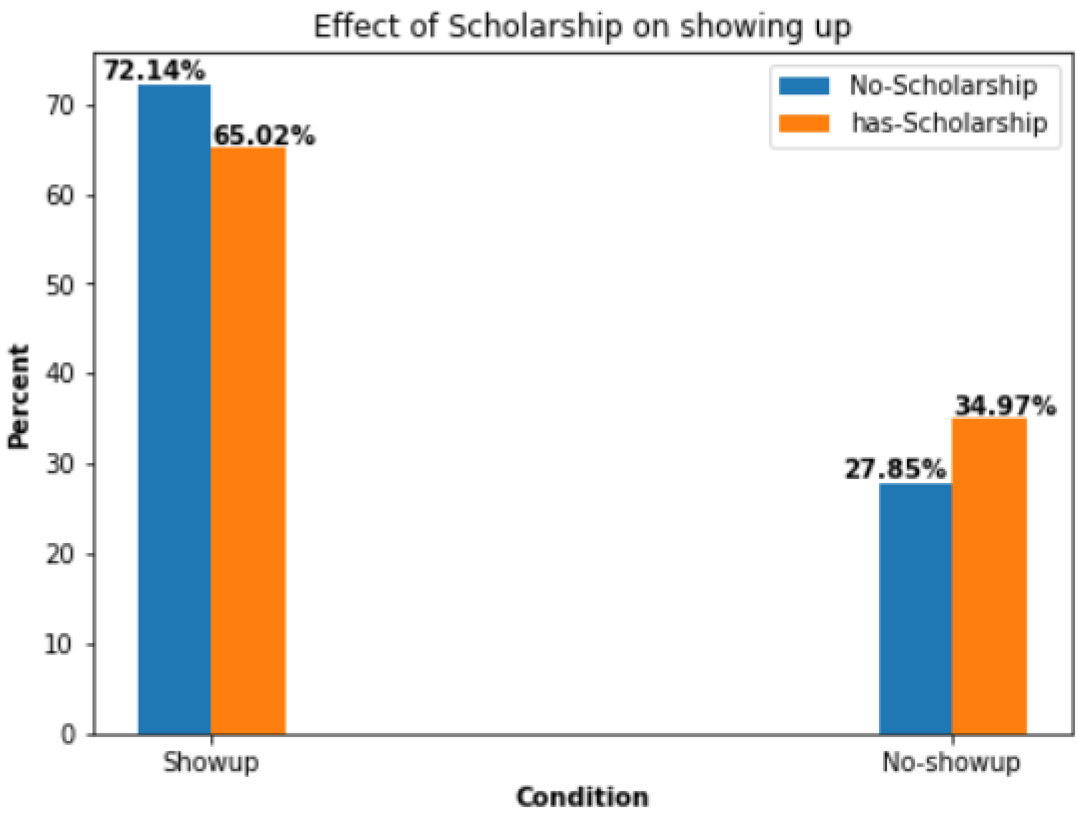
```
df.groupby(["Scholarship","No-show"]).count()["Age"]
```

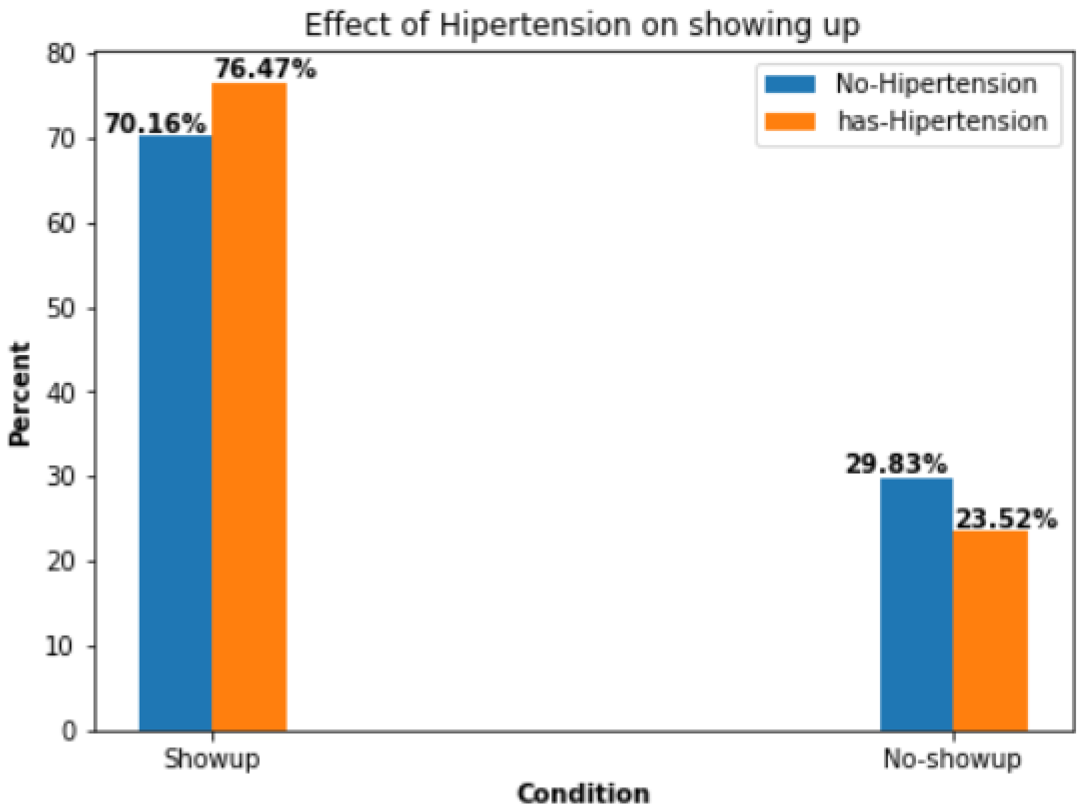
Out[39]:

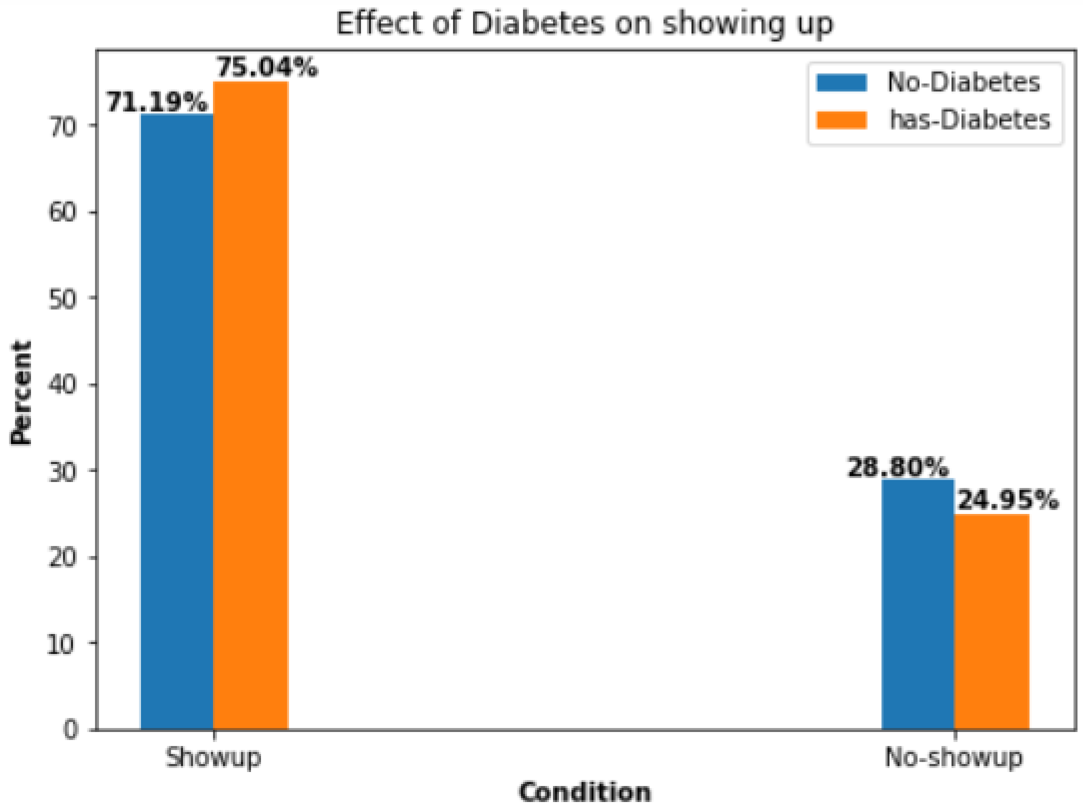
```
Scholarship  No-show
0            No         47099
             Yes        18189
1            No          4338
             Yes         2333
Name: Age, dtype: int64
```

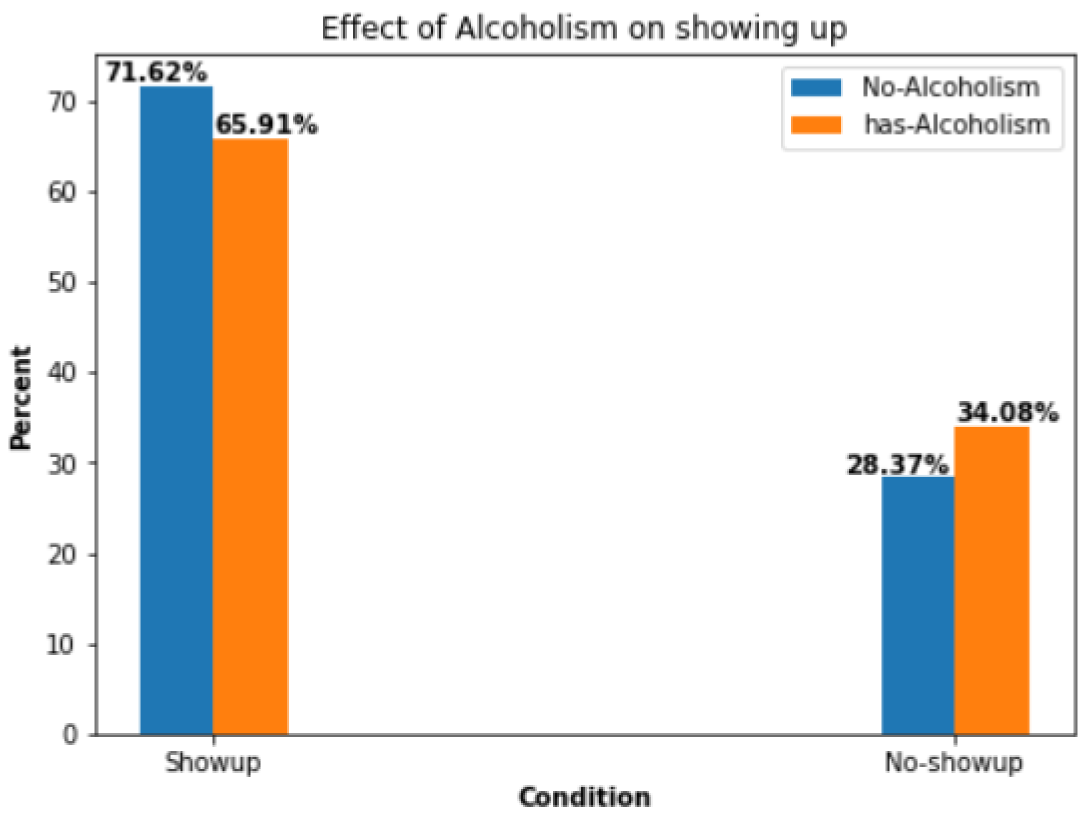
In [40]:

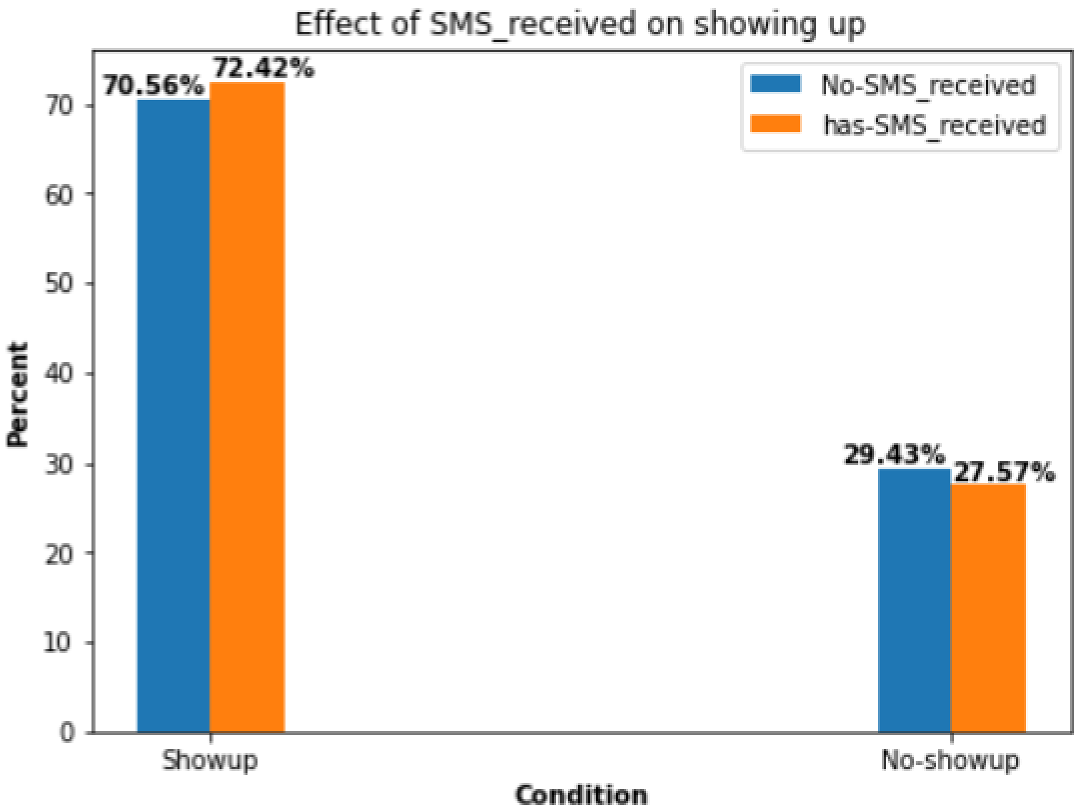
```
# Get effect of ('Scholarship', 'Hipertension', 'Diabetes', 'Alcoholism', 'SMS_received') the [0,1] columns,
# by comparing the percentage of people didn't show up of patients having Scholarship ("for example")
# and percent among those who don't have it.
variables = ['Scholarship', 'Hipertension', 'Diabetes', 'Alcoholism', 'SMS_received']
for variable in variables:
    fig, ax = plt.subplots(figsize = (7,5))
    plt.title("Effect of {} on showing up".format(variable))
    no = len(df[df[variable]==0])#total number of patients who don't have the [variable] Scholarship for example.
    yes = len(df[df[variable]==1])#total number of patients who have the [variable] Scholarship for example.
    Total = [no,no,yes,yes]
    #Get the percent of patients without Scholarship(for example) who showed up an not show up
    #And get the percent of patients with Scholarship (for example) who showed up an not show up
    variable_Percent = (df.groupby([variable,"No-show"]).count()["Age"]/Total)*100
    x = np.arange(2)
    #plot for percent of patients without Scholarship(for example) who showed up and didn't show up.
    ax.bar(x-0.05,variable_Percent[0],width = 0.1)
    ax.text(x[0]-.15 ,variable_Percent[0][0]+.5,str(variable_Percent[0][0])[:5]+"%", fontweight ='bold' )
    ax.text(x[1]-.15,variable_Percent[0][1]+.5,str(variable_Percent[0][1])[:5]+"%", fontweight ='bold' )
    #plot for percent of patients with Scholarship(for example) who showed up and didn't show up.
    ax.bar(x+0.05,variable_Percent[1],width = 0.1)
    ax.text(x[0] ,variable_Percent[1][0]+.5,str(variable_Percent[1][0])[:5]+"%", fontweight ='bold' )
    ax.text(x[1],variable_Percent[1][1]+.5,str(variable_Percent[1][1])[:5]+"%", fontweight ='bold' )
    plt.xticks(x, ['Showup','No-showup'])
    plt.legend(["No-{}".format(variable),"has-{}".format(variable)]);
    plt.xlabel("Condition",fontweight ='bold')
    plt.ylabel("Percent",fontweight ='bold')
```


Effect of Scholarship on showing up
No-Scholarship
has-Scholarship
72.14%
65.02%
34.97%
27.85%
Percent
Showup
No-showup
Condition


Effect of Hipertension on showing up
No-Hipertension
has-Hipertension
76.47%
70.16%
29.83%
23.52%
Percent
Showup
No-showup
Condition


Effect of Diabetes on showing up
No-Diabetes
has-Diabetes
71.19%
75.04%
28.80%
24.95%
Percent
0
10
20
30
40
50
60
70
Showup
No-showup
Condition


Effect of Alcoholism on showing up
No-Alcoholism
has-Alcoholism
71.62%
65.91%
28.37%
34.08%
Percent
0
10
20
30
40
50
60
70
Showup
No-showup
Condition



We can see that:

1- Higher percent of people with Scholarship didn't show up.

2- Higher percent of people without Hypertension didn't show up.

3 - Higher percent of people without Diabetes didn't show up.

4- Higher percent of people with Alcoholism didn't show up.

5- Slightly higher percent of people that didn't receive SMS didn't show up.

2.4 What is the effect of Gender on probability of showing up?

In [41]:
```
Gender_count = df.groupby(["Gender","No-show"]).count()["Age"]
Gender_count
```

Out[41]:
```
Gender  No-show
F       No         34396
        Yes        13674
M       No         17041
        Yes         6848
Name: Age, dtype: int64
```

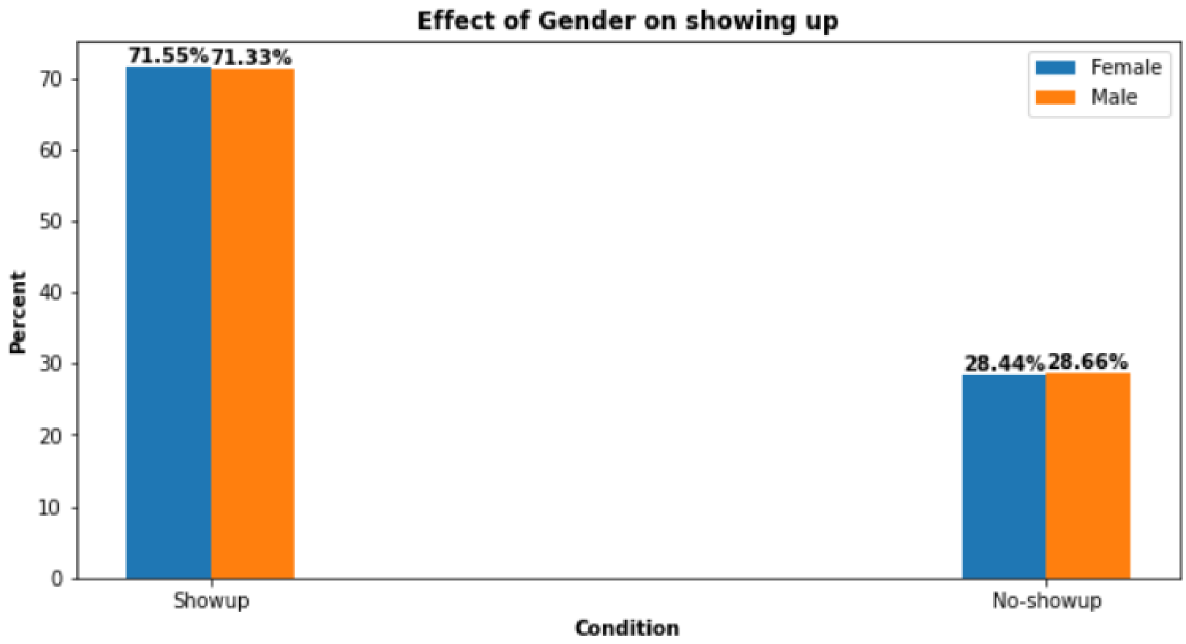
In [42]:
```
female_num = df[(df["Gender"]== "F")]["Gender"].count()
male_num = df[(df["Gender"]== "M")]["Gender"].count()
total_num = [female_num,female_num,male_num,male_num]
total_num
```

Out[42]:
```
[48070, 48070, 23889, 23889]
```

In [43]:
```
normalized_Gender = (Gender_count/total_num)*100
normalized_Gender
```

Out[43]:
```
Gender  No-show
F       No         71.553984
        Yes        28.446016
M       No         71.334087
        Yes        28.665913
Name: Age, dtype: float64
```

In [44]:
```
fig, ax = plt.subplots(figsize = (10,5))
plt.title("Effect of Gender on showing up",fontweight ='bold')
x = np.arange(2)
ax.bar(x-0.05,normalized_Gender[0:2],width = 0.1)
ax.text(x[0]-.1 ,normalized_Gender[0]+.5,str(normalized_Gender[0])[:5]+"%", fontweight ='bold' )
ax.text(x[1]-.1 ,normalized_Gender[1]+.5,str(normalized_Gender[1])[:5]+"%", fontweight ='bold' )
ax.bar(x+0.05,normalized_Gender[2:],width = 0.1)
ax.text(x[0] ,normalized_Gender[2]+.5,str(normalized_Gender[2])[:5]+"%", fontweight ='bold' )
ax.text(x[1] ,normalized_Gender[3]+.5,str(normalized_Gender[3])[:5]+"%", fontweight ='bold' )
plt.legend(["Female",'Male'])
plt.xticks(x, ["Showup","No-showup"])
plt.xlabel("Condition",fontweight ='bold')
plt.ylabel("Percent",fontweight ='bold');
```



We can see that:

1- About 71.5% of registered Females attended.
2- About 71.3% of registered Males attended.

So, it seems Gender doesn't have a significant effect on Attendance

2.5 What is the effect of AppointmentDay on probability of showing up?

In [88]:
```
df.groupby("appointment_day").count()["Age"]
```

Out[88]:
```
appointment_day
Friday       12516
Monday       14581
Saturday        31
Thursday     11325
Tuesday      16462
Wednesday    17044
Name: Age, dtype: int64
```

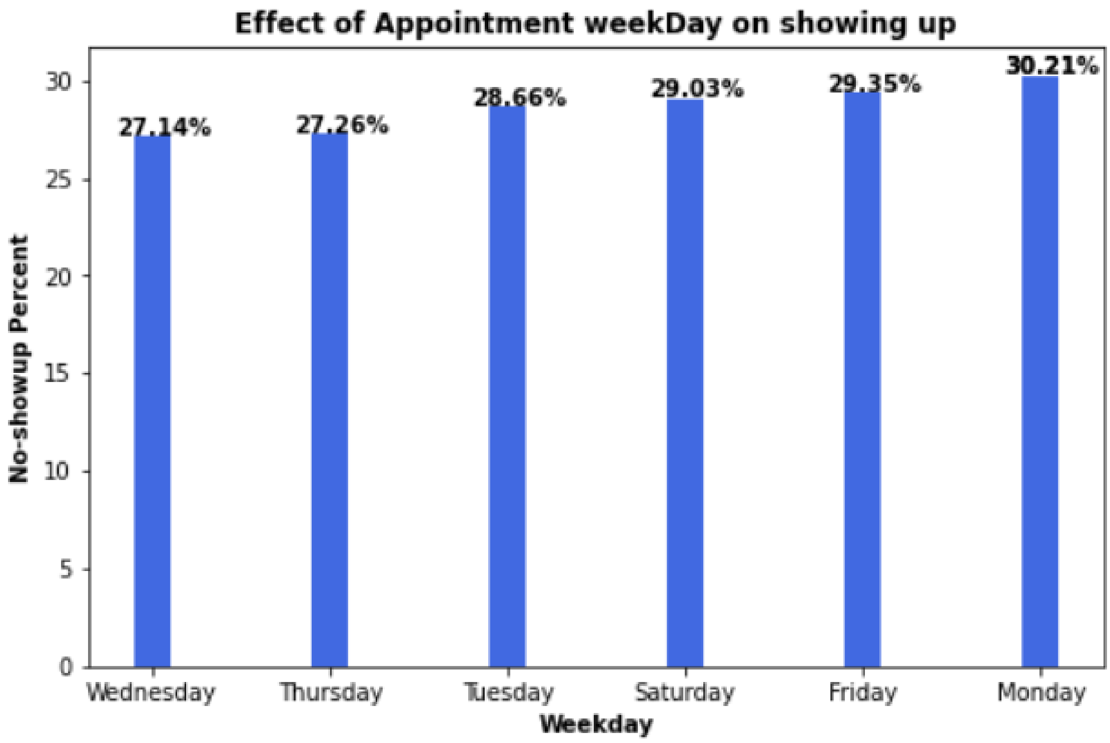
In [90]:
```
#total of no-show appointments in each week day
noshow_df.groupby("appointment_day").count()["Age"]
```

Out[90]:
```
appointment_day
Friday       3674
Monday       4405
Saturday        9
Thursday     3088
Tuesday      4719
Wednesday    4627
Name: Age, dtype: int64
```

In [47]:
```
#Get percentage of patients that didn't showup in each day of the wee
absense_percentage = ((noshow_df.groupby("appointment_day").count()
                        /df.groupby("appointment_day").count())["Age"]*100).sort_values()
absense_percentage
```

Out[47]:
```
appointment_day
Wednesday    27.147383
Thursday     27.267108
Tuesday      28.666019
Saturday     29.032258
Friday       29.354426
Monday       30.210548
Name: Age, dtype: float64
```

```python
In [48]:
fig, ax = plt.subplots(figsize = (8,5))
plt.title("Effect of Appointment weekDay on showing up",fontweight ='bold')
x = np.arange(6)
ax.bar(x,absense_percentage,color=["royalblue"] ,width = .2)
plt.xticks(x, absense_percentage.index)
ax.text(x[0]-.2,absense_percentage[0]+.1,str(absense_percentage[0])[:5]+"%",fontweight ='bold');
ax.text(x[1]-.2,absense_percentage[1]+.1,str(absense_percentage[1])[:5]+"%",fontweight ='bold');
ax.text(x[2]-.2,absense_percentage[2]+.1,str(absense_percentage[2])[:5]+"%",fontweight ='bold');
ax.text(x[3]-.2,absense_percentage[3]+.1,str(absense_percentage[3])[:5]+"%",fontweight ='bold');
ax.text(x[4]-.2,absense_percentage[4]+.1,str(absense_percentage[4])[:5]+"%",fontweight ='bold');
ax.text(x[5]-.2,absense_percentage[5]+.1,str(absense_percentage[5])[:5]+"%",fontweight ='bold');
ax.text(x[-1]-.2,absense_percentage[-1]+.1,str(absense_percentage[-1])[:5],fontweight ='bold');
plt.xlabel("Weekday",fontweight ='bold')
plt.ylabel("No-showup Percent",fontweight ='bold');
```



We can see that:

1- Days in the beginning and end of the week [Monday - Saturday - Friday] are more probable to have more patient that no-show up than midweek days.

2.6 What is the effect of Neighborhood on probability of showing up?

In [49]:
```
#Get percent of no-show up for each Neighborhood
neighbourhood_noshow_percent= (noshow_df["Neighbourhood"].value_counts().sort_index()/
                               df["Neighbourhood"].value_counts().sort_index()).sort_values()*100
neighbourhood_noshow_percent
```

Out[49]:
```
ILHA DO BOI                   8.695652
SOLON BORGES                 19.287834
AEROPORTO                    20.000000
DE LOURDES                   20.270270
MORADA DE CAMBURI            20.512821
                                ...
HORTO                        35.964912
ITARARÉ                      36.497270
JESUS DE NAZARETH            37.492877
GURIGICA                     38.371041
ILHAS OCEÂNICAS DE TRINDADE 100.000000
Name: Neighbourhood, Length: 80, dtype: float64
```

In [50]:
```
#Get the percent of patients in each Neighborhood relative to all patients
Neighbourhood_weight = (df["Neighbourhood"].value_counts()/len(df))*100
Neighbourhood_weight[:50]
```

Out[50]:
```
JARDIM CAMBURI         7.244403
MARIA ORTIZ            5.183507
RESISTÊNCIA            3.916119
JARDIM DA PENHA        3.689601
ITARARÉ                3.308829
CENTRO                 3.154574
TABUAZEIRO             2.673745
JESUS DE NAZARETH      2.438889
BONFIM                 2.373574
CARATOÍRA              2.349949
JABOUR                 2.337442
SANTA MARTHA           2.290193
SANTO ANTÔNIO          2.252672
SANTO ANDRÉ            2.242944
SÃO PEDRO              2.201253
ANDORINHAS             2.117873
```

We will deal only with Neighbourhoods that have at least 1% of all patients to make accurate observations.

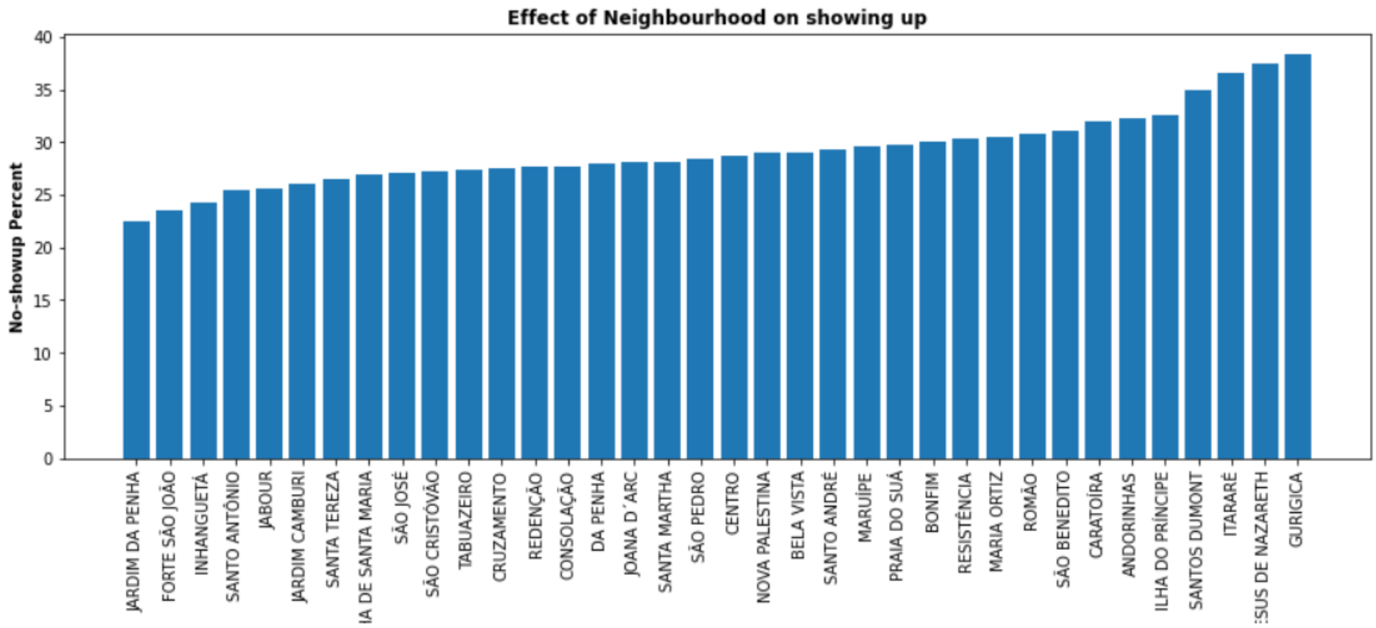
In [97]:
```
# slice neighbourhood_noshow_percent with only Neighbourhoods that have at least 1% of all patients
neighbourhood_noshow_percent = neighbourhood_noshow_percent.loc[(Neighbourhood_weight[Neighbourhood_weight>1]).index

neighbourhood_noshow_percent.sort_values()
```

Out[97]:
```
JARDIM DA PENHA       22.485876
FORTE SÃO JOÃO        23.511214
INHANGUETÁ            24.331210
SANTO ANTÔNIO         25.478100
JABOUR                25.564804
JARDIM CAMBURI        26.069442
SANTA TEREZA          26.473988
ILHA DE SANTA MARIA   26.869159
SÃO JOSÉ              27.107558
SÃO CRISTÓVÃO         27.158556
TABUAZEIRO            27.338877
CRUZAMENTO            27.512195
REDENÇÃO              27.604726
```

In [53]:

```
#plot Neighbourhood Vs. No-showup Percent
fig, ax = plt.subplots(figsize = (15,5))
plt.title("Effect of Neighbourhood on showing up",fontweight ='bold')
x = np.arange(len(neighbourhood_noshow_percent))
ax.bar(x,neighbourhood_noshow_percent.sort_values())
plt.xticks(x,neighbourhood_noshow_percent.sort_values().index,rotation ='vertical')
plt.xlabel("Neighbourhood",fontweight ='bold')
plt.ylabel("No-showup Percent",fontweight ='bold');
```



We can see that:

1- Neighborhoods like GURIGICA, JESUS DE NAZARETH and ITARARÉ, etc. have high percent of no-show ups. Surveys should be carried out to collect some data about quality of doctors and the way they are treating patients and the way the people of these places are thinking.

Conclusions

Results

1- Hipertension is more prevalent than Diabetes.

2- About 42.5% of Alcoholic Patients suffer Hipertension compared to only 20.3% for Non-Alcoholic Patients, So the probability to get Hipertension are twice for Alcoholic Patients compared to non-Alcoholic Patients.

3- About 11.4% of Alcoholic Patients suffer Diabetes compared to only 7.3% for Non-Alcoholic Patients, So the probability to get Diabetes are one and half for Alcoholic Patients compared to non-Alcoholic Patients.

4- The ages from Mid-20th to Mid-30th is considered the beginning to get a chronic disease.

5- Getting older increases the risk of getting a chronic disease.

6- The percent of diabetic patients with about 65 years old and older is decreasing. This makes no sense, so those older patients should be examined and surveys about their food and daily routine should be conducted.

7- In general, the higher the age is, the lower percentage of people who didn't show-up.

8- The longer the waiting time is, the higher percentage that didn't show up.

9- Higher percent of people with Scholarship didn’t show up.

10- Higher percent of people without Hypertension didn’t show up.

11- Higher percent of people without Diabetes didn’t show up.

12- Higher percent of people with Alcoholism didn’t show up.

13- Slightly higher percent of people that didn’t receive SMS didn’t show up.

14- Days in the beginning and end of the week [Monday - Saturday - Friday] are more probable to have more patient that no-show up.

15- Neighborhoods like GURIGICA, JESUS DE NAZARETH and ITARARÉ, etc. have high percent of no-show ups. Surveys should be carried out to collect some data about quality of doctors and the way they are treating patients and the way the people of these places are thinking.

Limitations

1- Hipertension, Diabetes and Alcoholism data are categorical, so our explanation is restricted as we don't have information about the level of disease in each patient and Patient and disease history.

2- Higher percent of people with Scholarship didn’t show up and this seems counter logic. More information is needed to make more accurate explanation.

3- Our explanation based on the percent of each Age that didn't show up or that has a chronic disease can’t be generalized, as we don't have equal amount of data for each age.