In [3]:
```
# Importing Liberaries:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1- Data Assessing:

In [5]:
```
# Load the Dataset:
data= pd.read_csv('no_show_appointments.csv')
data.head()
```

Out[5]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	H
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	

In [6]:
```
# Save the Dataset to Excel:
data.to_excel("Show_and_No_Show_Appointments.xlsx", sheet_name="Appointments", index=Fal
```

In [7]:
```
# Load the Dataset & Explore the Charcteristics of the Dataset:
df=pd.read_excel("Show_and_No_Show_Appointments.xlsx")
df.head()
```

Out[7]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	H
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	
4	8.841186e+12	5642494	F	2016-04-29T16:07:23Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	

In [8]:
```
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 [9]:
```
print('The dataset has a shape of:',df.shape)
```

```
The dataset has a shape of: (110527, 14)
```

In [10]:
```
if df.isna().sum().sum()== 0:
    print ('No NULL values in this dataset')
else:
    print('Total number of Null Values is: ',df.isna().sum().sum())
```

```
No NULL values in this dataset
```

In [11]:
```
if df.duplicated().sum()== 0:
    print ('No DUPLICATED values in this dataset')
else:
    print('Total number of Duplicated Values is: ',df.duplicated().sum())
```

```
No DUPLICATED values in this dataset
```

In [12]:
```
print('The number of UNIQUE values in this dataset:\n\n',df.nunique())
```

```
The number of UNIQUE values in this dataset:

 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 [13]:
```
df.head(1)
```

Out[13]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	H
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	

In [14]:
```
print('"Gender" Categories:        ',df.Gender.unique())
print('\n"Scolarship" Categories:    ', df.Scholarship.unique())
print('\n"Hipertension" Category:    ', df.Hipertension.unique())
print('\n"Diabetes" Category:        ', df.Diabetes.unique())
print('\n"Alcoholism" Category:      ', df.Alcoholism.unique())
print('\n"Handcap" Category:         ', df.Handcap.unique())
print('\n"SMS_received" Category:    ', df['SMS_received'].unique())
print('\n"No-show" Category:         ', df['No-show'].unique())
```

```
"Gender" Categories:         ['F' 'M']

"Scolarship" Categories:     [0 1]

"Hipertension" Category:     [1 0]

"Diabetes" Category:         [0 1]

"Alcoholism" Category:       [0 1]

"Handcap" Category:          [0 1 2 3 4]

"SMS_received" Category:     [0 1]

"No-show" Category:          ['No' 'Yes']
```

In [15]:
```
#The describtive statistics for the whole dataset:
df.describe()
```

Out[15]:

	PatientId	AppointmentID	Age	Scholarship	Hipertension	Diabetes	Alcoholism
count	1.105270e+05	1.105270e+05	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
std	2.560949e+14	7.129575e+04	23.110205	0.297675	0.397921	0.258265	0.171686

min	3.921784e+04	5.030230e+06	-1.000000	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
50%	3.173184e+13	5.680573e+06	37.000000	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
max	9.999816e+14	5.790484e+06	115.000000	1.000000	1.000000	1.000000	1.000000

In [16]:
```
df.head(1)
```

Out[16]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	H
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	

In [17]:
```
# Count of patients according to Gender Clasification:
df.groupby('Gender').PatientId.nunique().reset_index().rename(columns={'PatientId':'coun
```

Out[17]:

Gender	count_patient
F	40046
M	22253

In [18]:
```
# Count of Patients Based on Scholarship Grants:
df.groupby('Scholarship').PatientId.nunique().reset_index().rename(columns={'PatientId':
```

Out[18]:

Scholarship	count_patient
0	56511
1	5788

In [19]:
```
# Count of Patients Based on Hypertension Diagnosis:
df.groupby('Hipertension').PatientId.nunique().reset_index().rename(columns={'PatientId'
```

Out[19]:

Hipertension	count_patient
0	50057
1	12242

In [20]:
```
# Count of Patients Based on Diabetes Diagnosis
df.groupby('Diabetes').PatientId.nunique().reset_index().rename(columns={'PatientId':'co
```

Out[20]:

Diabetes	count_patient
0	57883
1	4416

In [21]:
```
# Count of Patients Based on Their Alcohol-Related Problems:
df.groupby('Alcoholism').PatientId.nunique().reset_index().rename(columns={'PatientId':'
```

Out[21]:

Alcoholism	count_patient
0	60793
1	1506

In [22]:
```
# Count of Patients Based on Their Handicap Status (if any):
df.groupby('Handcap').PatientId.nunique().reset_index().rename(columns={'PatientId':'cou
```

Out[22]:

Handcap	count_patient
0	61166
1	1025
2	99
3	6
4	3

In [23]:
```
# Count of Patients Based on Receiving Appointment Confirmation Messages:
df.groupby('SMS_received').AppointmentID.nunique().reset_index().rename(columns={'Patien
```

Out[23]:

SMS_received	AppointmentID
0	75045
1	35482

In [24]:
```
# Count of Patients Based on Attending Their Scheduled Appointment:
df.groupby('No-show').AppointmentID.nunique().reset_index().rename(columns={'PatientId':
```

Out[24]:

No-show	AppointmentID
No	88208
Yes	22319

2 - Data Cleaning:

a) Renaming Column labels:

In order to investigate easily the columns label nees to be changedas follows:

- *PatientId* to ***patient_id***
- *AppointmentID* to ***appoint_id***
- *Gender* to ***gender***
- *ScheduledDay* to ***sched_day***
- *AppointmentDay* to ***appoint_day***
- *Age* to ***age***
- *Neighbourhood* to ***neighbourhood***
- *Scholarship* to ***scholarship***
- *Hipertension* to ***htn***
- *Diabetes* to ***dm***
- *Alcoholism* to ***aud***
- *Handcap* to ***hcp***
- _SMS*received* to ***sms_received***
- *No-show* to ***no_show***

In [27]:
```
# Renaming the dataset columns:
```

```
df = df.rename(columns = {"PatientId":"patient_id", "AppointmentID":"appoint_id", 'Gende
                          "AppointmentDay":"appoint_day","Age":"age", "Neighbourhood":"
                          "Hipertension":"htn", "Diabetes":"dm", "Alcoholism":"aud","Ha
df.head(1)
```

Out[27]:

	patient_id	appoint_id	gender	sched_day	appoint_day	age	neighbourhood	scholarship	htn	dm	auc
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0

b) Changing the data types of:

- _patient*id* to ***integer***
- _sched*day* & _appoint*day* to ***date format***

In [29]:
```
# Changing the data types:
df['patient_id'] = df['patient_id'].astype('int64')
df['sched_day'] = pd.to_datetime(df['sched_day']).dt.date
df['appoint_day'] = pd.to_datetime(df['appoint_day']).dt.date
df.head(1)
```

Out[29]:

	patient_id	appoint_id	gender	sched_day	appoint_day	age	neighbourhood	scholarship	htn	dm	auc
0	29872499824296	5642903	F	2016-04-29	2016-04-29	62	JARDIM DA PENHA	0	1	0	0

In [30]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 14 columns):
 #   Column          Non-Null Count   Dtype 
---  ------          --------------   ----- 
 0   patient_id      110527 non-null  int64 
 1   appoint_id      110527 non-null  int64 
 2   gender          110527 non-null  object
 3   sched_day       110527 non-null  object
 4   appoint_day     110527 non-null  object
 5   age             110527 non-null  int64 
 6   neighbourhood   110527 non-null  object
 7   scholarship     110527 non-null  int64 
 8   htn             110527 non-null  int64 
 9   dm              110527 non-null  int64 
 10  aud             110527 non-null  int64 
 11  hcp             110527 non-null  int64 
 12  sms_received    110527 non-null  int64 
 13  no_show         110527 non-null  object
dtypes: int64(9), object(5)
memory usage: 11.8+ MB
```

c) Inserting New Columns in the dataset:

- ***date_diff:*** To measure the difference in days between *The Schedule Date* and *The Appointment Date*.
- ***age_stages:*** groupping the *age* column according to the different stages of human life, as follows:
  - **child:** Ages from *0* to *12*
  - **teenage:** Ages greater than *12* to *21*
  - **adult:** Ages greater than *21* to *40*

- **middle_age:** Ages greater than *40* to *65*
- **elderly:** Ages greater than *65*

In [32]:
```
# Inserting the "date_diff" column:
df['date_diff']= (df['appoint_day']- df['sched_day']).dt.days
df.head(1)
```

Out[32]:

	patient_id	appoint_id	gender	sched_day	appoint_day	age	neighbourhood	scholarship	htn	dm	auc
0	29872499824296	5642903	F	2016-04-29	2016-04-29	62	JARDIM DA PENHA	0	1	0	(

In [33]:
```
# relocating the "date_diff" column to be in the 5th column:
df.insert(5,'date_diff',df.pop('date_diff'))
df.head(1)
```

Out[33]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	age	neighbourhood	scholarship	htn
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	62	JARDIM DA PENHA	0	1

In [34]:
```
# Inserting the "age_stages" column:
df['age_stages']= ['child' if x <= 12
                   else 'teenage' if 12 < x <= 21
                   else 'adult' if 21 < x <= 40
                   else 'middle_aged' if 40 < x <= 65
                   else 'elderly' for x in df['age']]
df.head(1)
```

Out[34]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	age	neighbourhood	scholarship	htn
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	62	JARDIM DA PENHA	0	1

In [35]:
```
# relocating the "age_stages" column to be in the 6th column:
df.insert(6,'age_stages',df.pop('age_stages'))
df.head(1)
```

Out[35]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	age_stages	age	neighbourhood	sc
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	middle_aged	62	JARDIM DA PENHA	

d) Exploring the charcteristics of _"date*diff*",_"age*stages*", "*age*", and "*hcp*" :

1. *date_diff:*

- Applying descriptive statistics for further investigation.
- ***Note:***
  - The **sched_day** should occur before the **appoint_day** (*i.e.,* appoint_day > sched_day)

In [38]:
```
# Exploring date_diff column:
df.date_diff.describe().reset_index().rename(columns={'index':'stat','date_diff': 'amoun
```

Out[38]:

	stat	amount

0	count	110,527
1	mean	10
2	std	15
3	min	-6
4	25%	0
5	50%	4
6	75%	15
7	max	179

In [39]:
```
# Determining the dates where the schedule dates were after the appointment dates:
df.query('date_diff<0').iloc[:,3:6]
```

Out[39]:

	sched_day	appoint_day	date_diff
27033	2016-05-10	2016-05-09	-1
55226	2016-05-18	2016-05-17	-1
64175	2016-05-05	2016-05-04	-1
71533	2016-05-11	2016-05-05	-6
72362	2016-05-04	2016-05-03	-1

In [40]:
```
# Adjusting those appointment dates to be the same as the schedule dates:
df['appoint_day']=np.where(df['appoint_day']<df['sched_day'],df['sched_day'],df['appoint
df['date_diff']= (df['appoint_day']- df['sched_day']).dt.days
time_stats=df.date_diff.describe().reset_index().rename(columns={'index':'stat','date_di
time_stats
```

Out[40]:

	stat	amount
0	count	110,527
1	mean	10
2	std	15
3	min	0
4	25%	0
5	50%	4
6	75%	15
7	max	179

In [41]:
```
# Determining the most frequent date_diff:
print(f'The most frequent date diff is {df.date_diff.value_counts().idxmax()} day(s)')
```

```
The most frequent date diff is 0 day(s)
```

2. *age_stages:*

- Checking if a patient's ID is repeated across two categories instead of just one.
- ***Note:***
  - Sometimes, due to the time span of the dataset, a patient's age may shift between categories. For instance, if a patient was initially recorded as 12 years old (categorized as a child), and later, after a

considerable time gap, is recorded as 13 (categorized as a teenager).

In [43]:
```
# The number of patients (Unique):
patient_count=df.patient_id.nunique()
print(f'The Total Number of Patients is {patient_count} Patients\n')

# The number of patients when classified into categories (Unique):
patient_count_category=df.groupby('age_stages').patient_id.nunique().sum()
print(f'The Total Number of Patients (when categorized) is {patient_count_category} Pati

# The number of Patients that are repeated in 2 categories:
print(f'The Difference = {patient_count_category - patient_count} Patients')
```

```
The Total Number of Patients is 62299 Patients

The Total Number of Patients (when categorized) is 62352 Patients

The Difference = 53 Patients
```

In [44]:
```
# Determining the number of patients that are recorded in two categories:
c_patient = df.query('age_stages == "child"').patient_id.unique()
t_patient= df.query('age_stages=="teenage"').patient_id.unique()
a_patient = df.query('age_stages == "adult"').patient_id.unique()
m_patient= df.query('age_stages=="middle_aged"').patient_id.unique()
e_patient = df.query('age_stages == "elderly"').patient_id.unique()

repeated_1 = set(c_patient).intersection(t_patient)
repeated_2 = set(t_patient).intersection(a_patient)
repeated_3 = set(a_patient).intersection(m_patient)
repeated_4 = set(m_patient).intersection(e_patient)

print(f'''repeated patients number:\n\n -Child & Teenage = {len(repeated_1)} Patients\n
 - Adult & Middle Aged = {len(repeated_3)} Patients \n - Middle_aged & Elderly = {len(re
```

```
repeated patients number:

 -Child & Teenage = 12 Patients
 -Teenage & Adult = 16 Patients
 - Adult & Middle Aged = 13 Patients
 - Middle_aged & Elderly = 12 Patients
```

In [45]:
```
# Replacing the category of repeated patients to be the same as recorded at the begining
# (Sine the change in age was 1 year)
df.loc[df['patient_id'].isin(repeated_1), 'age_stages'] = 'child'
df.loc[df['patient_id'].isin(repeated_2), 'age_stages'] = 'teenage'
df.loc[df['patient_id'].isin(repeated_3), 'age_stages'] = 'adult'
df.loc[df['patient_id'].isin(repeated_4), 'age_stages'] = 'middle_aged'
```

In [46]:
```
# The amended number of patients when classified into categories (Unique):
patient_count_category=df.groupby('age_stages').patient_id.nunique().sum()
print(f'The Total Number of Patients (when categorized) is {patient_count_category} Pati
```

```
The Total Number of Patients (when categorized) is 62299 Patients
```

3. ***age:***

- Investigating the cause of negative values that appeared in the minimum value during descriptive statistics.
- ***Note:***
  - Since the data type for age is an integer, ages less than one year ***(in months)*** are recorded as **0** years

In [48]:
```
# Descriptive statistics for age:
```

```
df.age.describe().reset_index().style.format({'age':'{:,.0f}'})
```

Out[48]:

	index	age
0	count	110,527
1	mean	37
2	std	23
3	min	-1
4	25%	18
5	50%	37
6	75%	55
7	max	115

In [49]:
```
df.query("age < 0")
```

Out[49]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	age_stages	age	neighbourhood
99832	465943158731293	5775010	F	2016-06-06	2016-06-06	0	child	-1	ROMÃO

In [50]:
```
# replacing the value of -1 years with 1:
df.age= np.where(df.age< 0,1,df.age)
```

In [51]:
```
df.age.describe().reset_index().style.format({'age':'{:,.0f}'})
```

Out[51]:

	index	age
0	count	110,527
1	mean	37
2	std	23
3	min	0
4	25%	18
5	50%	37
6	75%	55
7	max	115

3. ***hcp:***

- Handicap (HCP): an illness, injury, or condition that makes it difficult for someone to do some things that other people do. Where, True = [1,2,3,4] and False = 0.
- ***Note:***
  - The dataset description does not clarify whether 1 indicates the mildest level of handicap and 4 the most severe, or if it's the other way around.
  - Since the total number of patients classified under handicap levels 1, 2, 3, and 4 is 1,133, representing less than 2% of the total, it would be more practical to combine these categories into a single classification.

In [53]:
```
# Identifying the Count of Patients with a Handicap:
df.groupby('hcp').patient_id.nunique().reset_index().rename(columns={'patient_id':"patie
```

Out[53]:

	hcp	patient_count
0	0	61166
1	1	1025
2	2	99
3	3	6
4	4	3

In [54]:
```
# Replacing 2,3,& 4 with 1:
df.hcp= np.where(df.hcp > 1,1,df.hcp)
```

In [55]:
```
# Adjusted Patients with a Handicap:
df.groupby('hcp').patient_id.nunique().reset_index().rename(columns={'patient_id':"patie
```

Out[55]:

	hcp	patient_count
0	0	61166
1	1	1133

e) Inserting appoint_gap column:

- ***appoint_gap:*** groupping the _date*diff* column according to the different gaps between *Schedule Date* and *Appointment Date* (According to the descriptive statistics made above), as follows:
  - **same_day:** *0* day.
  - **narrow_gap:** Days more than *0* to *4* days.
  - **moderate_gap:** Days more than *4* to *15* days.
  - **long_gap:** Days more than *15*.

In [57]:
```
# Inserting the "appoint_gap" column:
df['appoint_gap']= ['same_day' if x == 0
                    else 'narrow_gap' if 0 < x <= 4
                    else 'moderate_gap' if 4 < x <= 15
                    else 'long_gap' for x in df.date_diff]
df.head(1)
```

Out[57]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	age_stages	age	neighbourhood	sc
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	middle_aged	62	JARDIM DA PENHA	

In [58]:
```
# relocating the "appoint_gap" column to be in the 6th column:
df.insert(6,'appoint_gap',df.pop('appoint_gap'))
df.head(1)
```

Out[58]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	appoint_gap	age_stages	age	neigh
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	same_day	middle_aged	62	

In [59]:
```
# Exploring the charcteristics of the amended dataframe:
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 110527 entries, 0 to 110526
Data columns (total 17 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   patient_id      110527 non-null  int64
 1   appoint_id      110527 non-null  int64
 2   gender          110527 non-null  object
 3   sched_day       110527 non-null  object
 4   appoint_day     110527 non-null  object
 5   date_diff       110527 non-null  int64
 6   appoint_gap     110527 non-null  object
 7   age_stages      110527 non-null  object
 8   age             110527 non-null  int64
 9   neighbourhood   110527 non-null  object
 10  scholarship     110527 non-null  int64
 11  htn             110527 non-null  int64
 12  dm              110527 non-null  int64
 13  aud             110527 non-null  int64
 14  hcp             110527 non-null  int64
 15  sms_received    110527 non-null  int64
 16  no_show         110527 non-null  object
dtypes: int64(10), object(7)
memory usage: 14.3+ MB
```

In [60]:
```
# Save rhe Dataset to Excel:
df.to_excel("Show_and_No_Show_Appointments_Edited.xlsx", sheet_name="Appointments", inde
```

2 - Data Exploring:

In [62]:
```
# Load the Edited Dataset:
new_df=pd.read_excel('Show_and_No_Show_Appointments_Edited.xlsx')
```

Investigating Columns

In [64]:
```
new_df.head(1)
```

Out[64]:

	patient_id	appoint_id	gender	sched_day	appoint_day	date_diff	appoint_gap	age_stages	age	neigh
0	29872499824296	5642903	F	2016-04-29	2016-04-29	0	same_day	middle_aged	62	

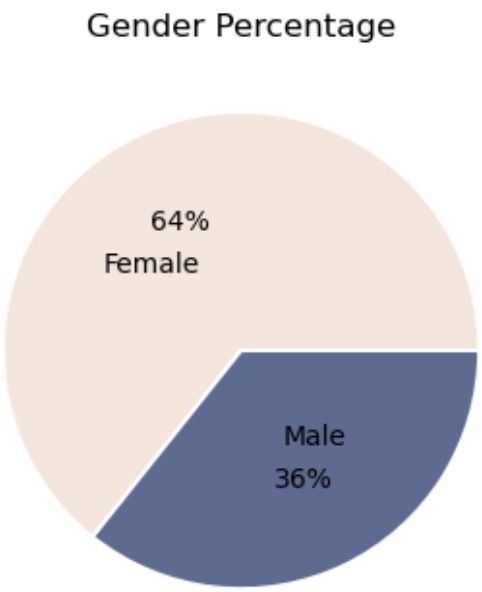
In [65]:
```
# Exploring Gender Column:
new_df.groupby('gender').patient_id.nunique().reset_index().rename(columns={'index':'gen
```

Out[65]:

gender	patient_id
F	40046
M	22253

In [66]:
```
x=new_df.groupby('gender').patient_id.nunique()

label = 'Female','Male'

text = ''' The dataset recorded:\n\n - 40,046 female patients with a percentage of 64%\n
 - 22,253 male patients with a percentage of 36%'''

plt.subplots(figsize = (4,4))
plt.pie(x, labels=label, colors= ['#f4e5dd','#5e6b91'], autopct='%1.0f%%',labeldistance=
```

```
plt.title('Gender Percentage')
plt.text(1.5,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');
```

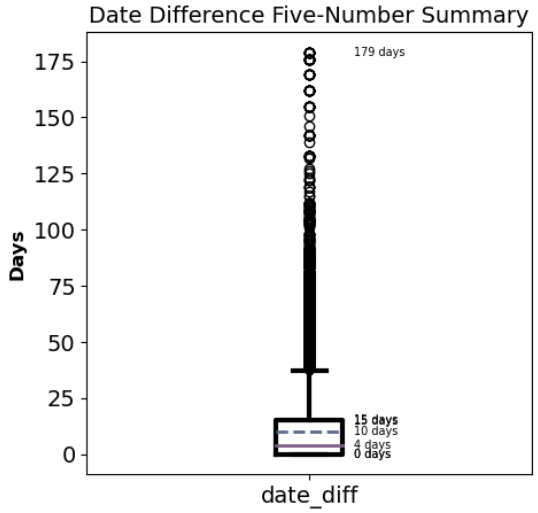


The dataset recorded:

- 40,046 female patients with a percentage of 64%

- 22,253 male patients with a percentage of 36%

In [67]:
```
# Exploring the date_diff:
x= new_df.date_diff

time_stats =new_df.date_diff.describe().reset_index().rename(columns={'index':'stat','da

notes= '''- Average = 10 days (between schedule date & appointment date)\n\n
- Minimum = 0 days (the appointment date was at the same day of schedule date)\n\n
- Maximum = 179 days (between schedule date & appointment date)\n\n
- 1st Quartile = 0 days (equals Minimum & equals Mode)\n\n
- Median = 4 days (between schedule date & appointment date)\n\n
- 3rd Quartile = 15 days (between schedule date & appointment date)\n\n
"The data regarding the comparison between schedule date & Appointment date are right sk

plt.subplots(figsize = (5,5))
plt.boxplot( x,showmeans = True, meanline  = True,showcaps = True,medianprops={ "color"
            boxprops={"color":"black","linewidth": 3},whiskerprops={"color": "black", "l
            capprops={"color": "black", "linewidth": 3},meanprops = {"color": "#5e6b91",

for i, v in enumerate(time_stats.iloc[1:,1]):
  plt.text(1.1,v, f'{v:.0f} days', ha='left', va='center',fontsize=7)

plt.title('Date Difference Five-Number Summary', fontsize = 14)
plt.xticks([1],['date_diff'], fontsize = 14)
plt.yticks(fontsize = 14)
plt.ylabel('Days',fontsize = 12, weight = 'bold')
plt.text(1.6,100,notes,ha='left',va='center',fontsize = 10, weight = 'normal');
```



- Average = 10 days (between schedule date & appointment date)

- Minimum = 0 days (the appointment date was at the same day of schedule date)

- Maximum = 179 days (between schedule date & appointment date)

- 1st Quartile = 0 days (equals Minimum & equals Mode)

- Median = 4 days (between schedule date & appointment date)

- 3rd Quartile = 15 days (between schedule date & appointment date)

"The data regarding the comparison between schedule date & Appointment date are right skewed "

In [68]:
```
# Exploring age_stages column:
# Category Count:
age_stage_count=new_df.groupby(['age_stages']).patient_id.nunique().reset_index().rename
age_stage_count.sort_values('count').style.hide()
```

Out[68]:

age_stages	count
teenage	6962
elderly	7711
child	12675
adult	14897
middle_aged	20054

In [69]:
```
# Calculating age averages of each category:
avg_age=new_df.groupby('age_stages').age.mean().reset_index().rename(columns={'age':'avg
avg_age.sort_values('avg_age').style.hide().format({'avg_age':'{:,.2f}'})
```

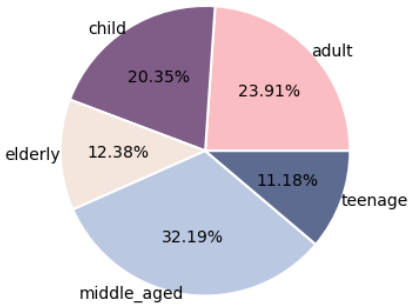
Out[69]:

age_stages	avg_age
child	4.98
teenage	17.27
adult	31.29
middle_aged	52.91
elderly	74.73

In [70]:
```
# Exploring Counts & Averages of age_stages by visualization:

notes = '''
The dataset recorded:\n\n
- 12,675 child patients, representing 20.35% of the total data, with an average age of 4
 - 6,962 teenage patients, representing 11.18% of the total data, with an average age of
 - 14,897 adult patients, representing 23.91% of the total data, with an average age of
 - 20,054 middle-aged patients, representing 32.19% of the total data, with an average a
 - 7,711 elderly patients, representing 11.18% of the total data, with an average age of
 '''

# Distribution of patients according to Age Stages:
x=new_df.groupby(['age_stages']).patient_id.nunique()

plt.subplots(figsize = (4,4))
plt.pie(x,labels = age_stage_count.age_stages,  colors= ['#f9bdc2','#805d87','#f4e5dd','
```

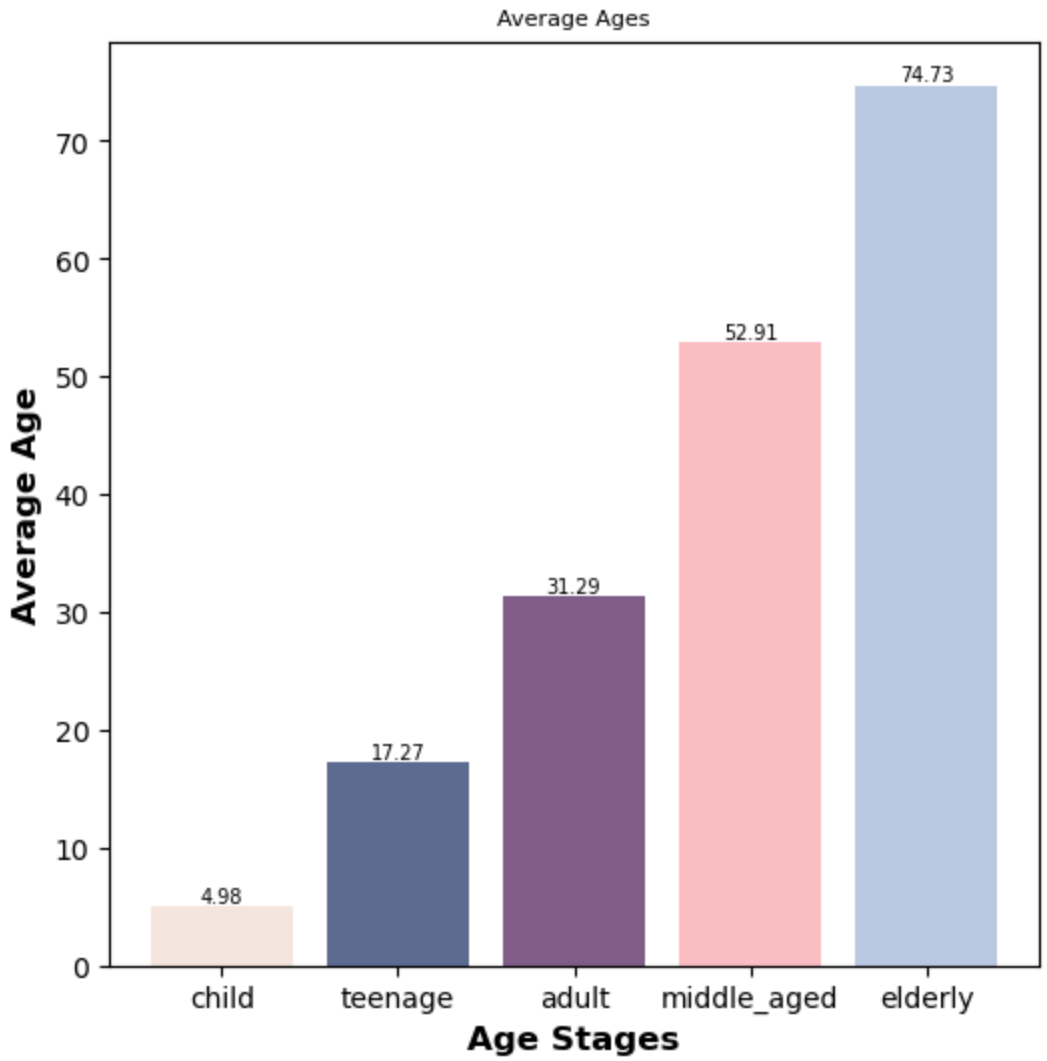
```
        labeldistance=1,pctdistance=.6, wedgeprops={"linewidth": 1.5, "edgecolor": "whit
plt.title('Distribution of patients according to Age Stage')
plt.text(1.8,0,notes,ha='left',va='top',fontsize = 10, weight = 'normal')
plt.show();

# Average Ages:
x=avg_age.sort_values('avg_age').age_stages.to_list()
y=avg_age.sort_values('avg_age').avg_age.to_list()

plt.subplots(figsize = (6,6))
plt.bar(x,y,color= ['#f4e5dd','#5e6b91','#805d87','#f9bdc2','#bbc8e2'], alpha = 1)

for i, v in enumerate(avg_age.sort_values('avg_age').avg_age):
  plt.text(i,v, f"{v:.2f}", ha='center', va='bottom',fontsize=7)

plt.title('Average Ages', fontsize = 8)
plt.xticks(fontsize = 10, rotation = 0)
plt.yticks(fontsize = 10)
plt.xlabel('Age Stages',fontsize = 12, weight = 'bold')
plt.ylabel('Average Age',fontsize = 12, weight = 'bold')
plt.show();
```

Distribution of patients according to Age Stage



The dataset recorded:

- 12,675 child patients, representing 20.35% of the total data, with an average age of 4.98 years.
- 6,962 teenage patients, representing 11.18% of the total data, with an average age of 17.26 years.
- 14,897 adult patients, representing 23.91% of the total data, with an average age of 31.29 years.
- 20,054 middle-aged patients, representing 32.19% of the total data, with an average age of 52.91 years
- 7,711 elderly patients, representing 11.18% of the total data, with an average age of 74.73 years.



In [71]:
```
# Category count when classified into gender:
count_stages_gender=new_df.groupby(['age_stages','gender']).patient_id.nunique().reset_i
count_stages_gender.sort_values('age_stages').style.hide()
```

Out[71]:

age_stages	gender	count
adult	F	10563
adult	M	4334
child	F	6156
child	M	6519
elderly	F	5129
elderly	M	2582
middle_aged	F	13641
middle_aged	M	6413
teenage	F	4557
teenage	M	2405

In [72]:
```
# Category averages when classifies by gender:
avg_age_gender=new_df.groupby(['age_stages','gender']).age.mean().reset_index().rename(c
avg_age_gender.sort_values('average_age').style.hide().format({'average_age':'{:,.2f}'})
```

Out[72]:

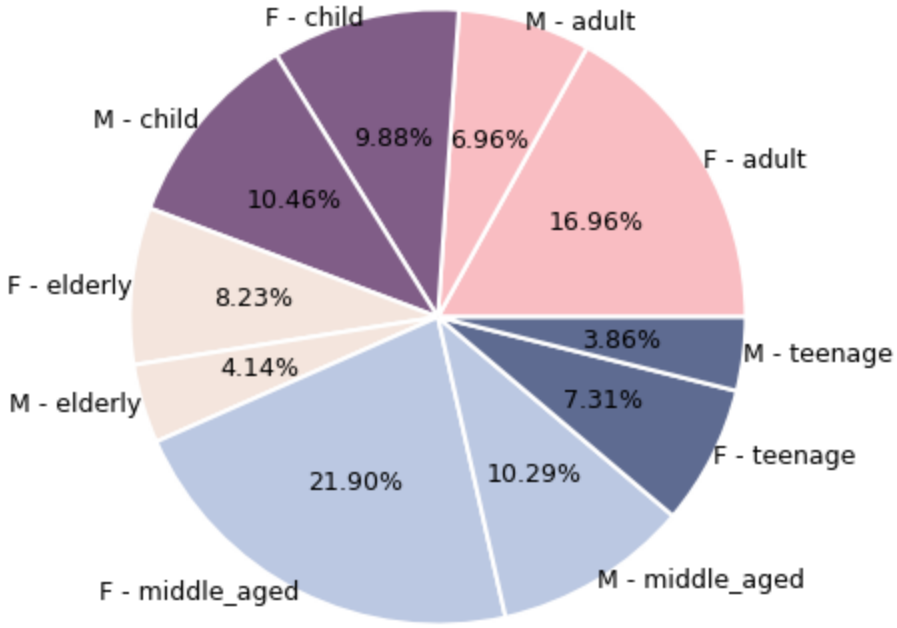
age_stages	gender	average_age

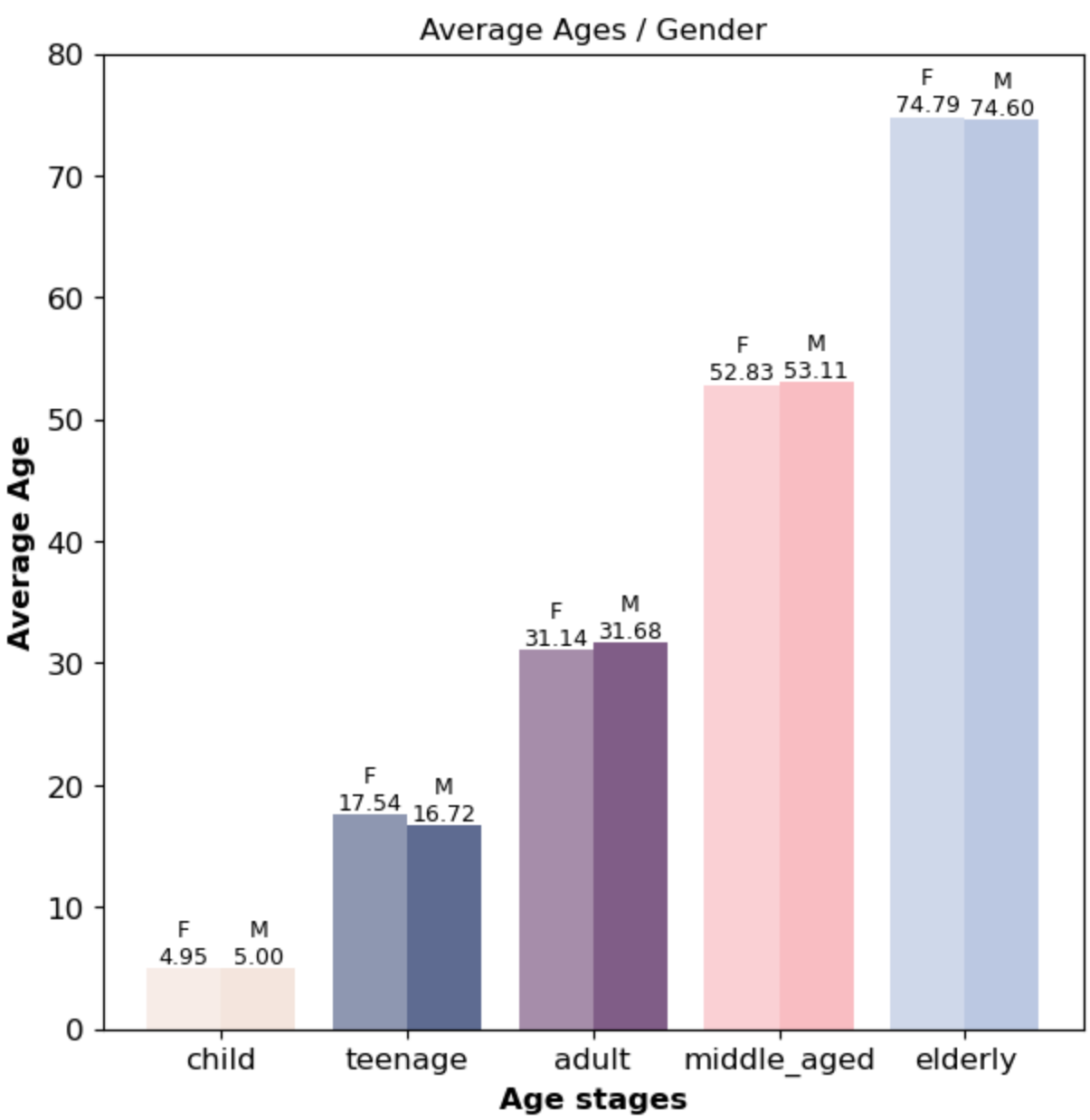
child	F	4.95
child	M	5.00
teenage	M	16.72
teenage	F	17.54
adult	F	31.14
adult	M	31.68
middle_aged	F	52.83
middle_aged	M	53.11
elderly	M	74.60
elderly	F	74.79

In [73]:

```
# Visualizing counts & averages of age_stages when classified by gender:
# Distribution of Patients according to Age Stages & Gender:
x=new_df.groupby(['age_stages','gender']).patient_id.nunique()

plt.subplots(figsize = (5,5))
plt.pie(x,labels = count_stages_gender.gender +" - "+ count_stages_gender.age_stages,
        colors= ['#f9bdc2','#f9bdc2','#805d87','#805d87','#f4e5dd','#f4e5dd','#bbc8e2','
        autopct='%1.2f%%',labeldistance=1,pctdistance=.6,textprops={'fontsize': 9},
        wedgeprops={"linewidth": 1.5, "edgecolor": "white"} )

plt.title('Distribution of patients according to Age Stage & Gender',fontsize =12)
plt.show();

# Average Ages per Gender:
stage=avg_age_gender.sort_values('average_age').age_stages.unique().tolist()
avg = avg_age_gender.sort_values('average_age').gender.tolist()
gender_f=avg_age_gender.sort_values('average_age').query('gender=="F"').average_age.toli
gender_m=avg_age_gender.sort_values('average_age').query('gender=="M"').average_age.toli

width=.4
x=np.arange(len(gender_f))
locations= x+width/2
plt.subplots(figsize = (7,7))

plt.bar(x,gender_f,width,color=['#f4e5dd','#5e6b91','#805d87','#f9bdc2','#bbc8e2'], alph
plt.bar(x + width,gender_m,width,color=['#f4e5dd','#5e6b91','#805d87','#f9bdc2','#bbc8e2

for i, v in enumerate(gender_f):
  plt.text(i,v, f"F\n{v:.2f}", ha='center', va='bottom',fontsize=9)

for i, v in enumerate(gender_m):
  plt.text(i+width,v, f"M\n{v:.2f}", ha='center', va='bottom',fontsize=9)

plt.xticks(locations, stage,fontsize = 12, rotation = 0)
plt.yticks(np.arange(0,90,10),fontsize = 12)
plt.xlabel('Age stages',fontsize = 12, weight = 'bold')
plt.ylabel('Average Age',fontsize = 12, weight = 'bold')
plt.title('Average Ages / Gender', fontsize =12)
plt.show();
```

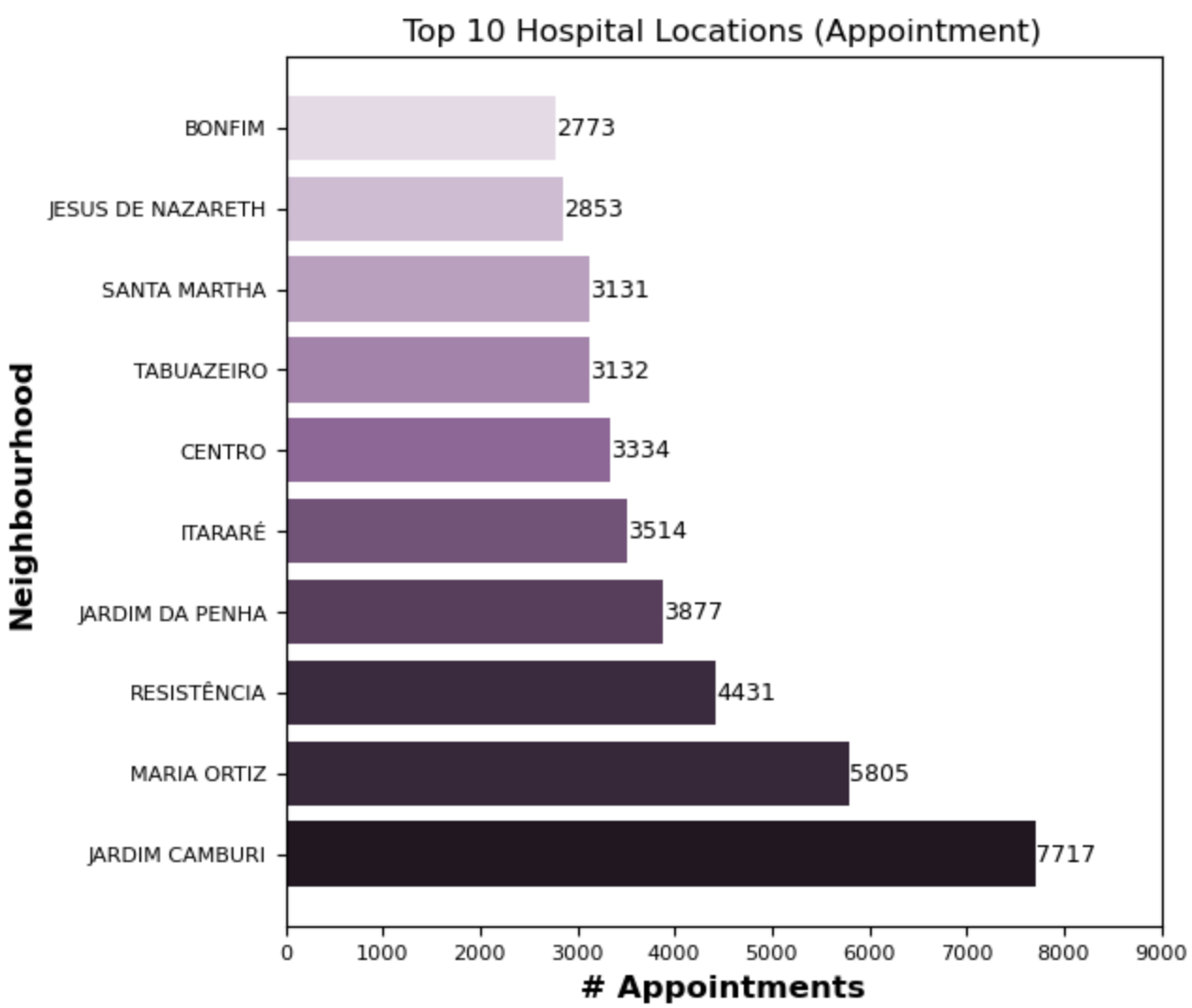
Distribution of patients according to Age Stage & Gender





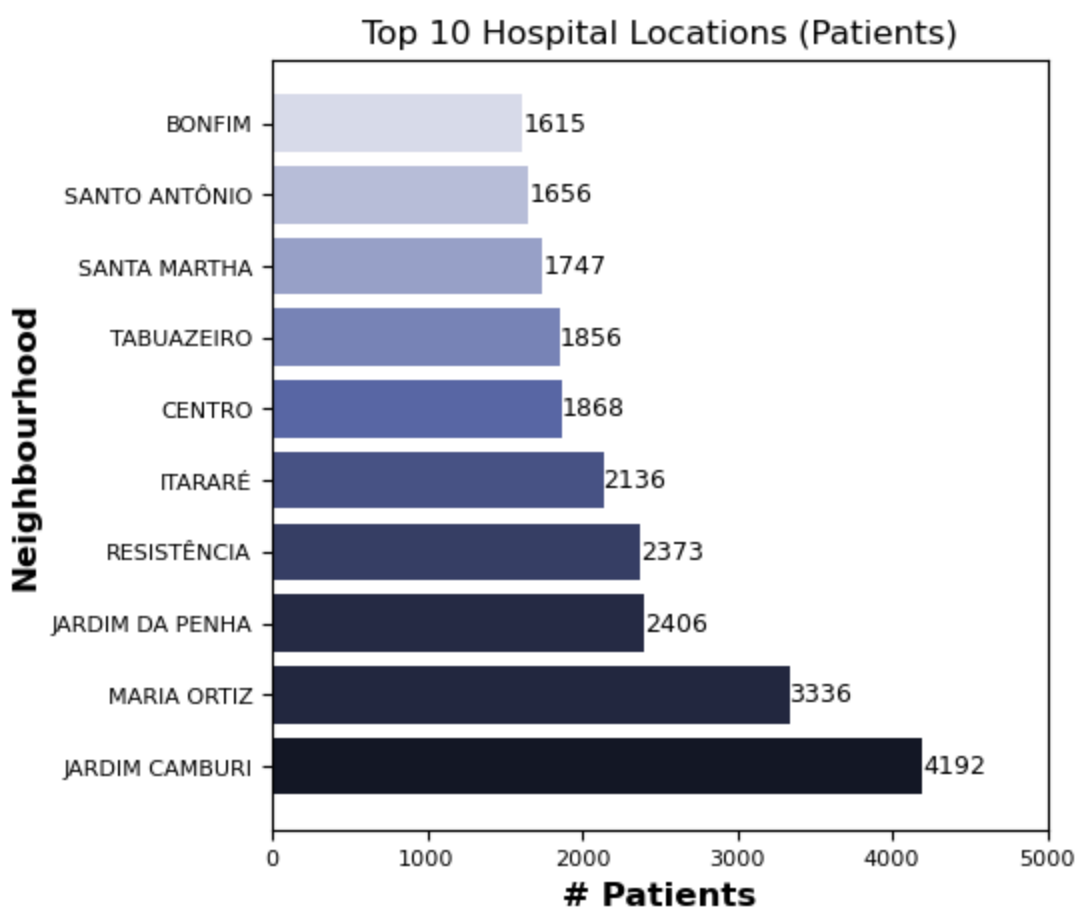
In [74]:

```
# Determining the top 10 hospital locations that had the highest appointment numbers:
appointment_neibourhood=new_df.groupby('neighbourhood').appoint_id.count().reset_index()
neighbourhood_a=appointment_neibourhood.sort_values('count_appointment',ascending= False
Appoint_count=appointment_neibourhood.sort_values('count_appointment',ascending= False).

plt.subplots(figsize = (6,6))
plt.barh(neighbourhood_a,Appoint_count,color= ['#201721','#362739','#3b2b3e','#573f5b','

for i, v in enumerate(Appoint_count):
  plt.text(v,i, f"{v:.0f}", ha='left', va='center',fontsize=9)

plt.title('Top 10 Hospital Locations (Appointment)', fontsize = 12)
plt.xticks(np.arange(0,10000,1000),fontsize = 8)
plt.yticks(fontsize = 8)
plt.xlabel('# Appointments',fontsize = 12, weight = 'bold')
plt.ylabel('Neighbourhood',fontsize = 12, weight = 'bold')
plt.show();
```



In [75]:

```
# Determining the top 10 hospital locations that had the most recorded patients:
patient_neibourhood=new_df.groupby('neighbourhood').patient_id.nunique().reset_index().r
neighbourhood_p=patient_neibourhood.sort_values('count_patient',ascending= False).head(1
p_count=patient_neibourhood.sort_values('count_patient',ascending= False).head(10).count

plt.subplots(figsize = (5,5))

plt.barh(neighbourhood_p,p_count,color= ['#131725','#22273f','#252a44','#363e64','#47528

for i, v in enumerate(p_count):
  plt.text(v,i, f"{v:.0f}", ha='left', va='center',fontsize=9)

plt.title('Top 10 Hospital Locations (Patients)', fontsize = 12)
plt.xticks(np.arange(0,6000,1000),fontsize = 8)
plt.yticks(fontsize = 8)
plt.xlabel('# Patients',fontsize = 12, weight = 'bold')
plt.ylabel('Neighbourhood',fontsize = 12, weight = 'bold')
plt.show();
```



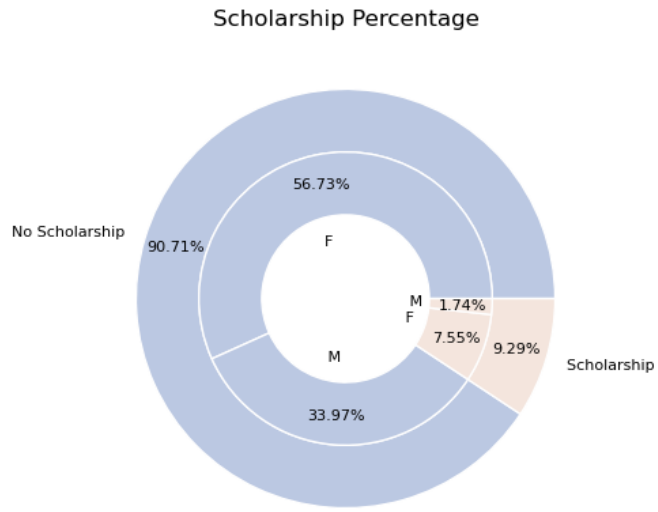
In [76]:
```
# Exploring the Scholarship Column (Count of Patients With or Without Scholarships, Clas
s_patients=new_df.groupby(['scholarship','gender']).patient_id.nunique().reset_index().r
s_patients
```

Out[76]:

	scholarship	gender	patient_count
0	0	F	35345
1	0	M	21166
2	1	F	4701
3	1	M	1087

In [77]:
```
# Visualizing the Scholarship Column (Count of Patients With or Without Scholarships, Cl
x=new_df.groupby(['scholarship','gender']).patient_id.nunique()

size = 0.3
label_s = 'No Scholarship','Scholarship '
label_g = 'F',' M',' F',' M'
text = '''
The dataset recorded:\n\n
- 5,788 patients with scholarship (with a percentage of 9.29%)\n
    "4,701 females (7.55%) & 1,087 males (1.74%)"\n\n
 - 56,511 patients with scholarship (with a percentage of 90.7%)\n
    "35,345 females (56.73%) & 21,166 males (33.97%)"
    '''

plt.subplots(figsize = (5,5))
plt.pie(x.groupby('scholarship').sum(), radius=1, colors= ['#bbc8e2','#f4e5dd'],labels =
        pctdistance=.85,textprops={'fontsize': 8}, wedgeprops=dict(width=size, edgecolor

plt.pie(x, radius=1-size, colors= ['#bbc8e2','#bbc8e2','#f4e5dd','#f4e5dd'],labels=label
        labeldistance=.4,textprops={'fontsize': 8},wedgeprops=dict(width=size, edgecolor
```

```
plt.title('Scholarship Percentage')
plt.text(2,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');
```



The dataset recorded:

- 5,788 patients with scholarship (with a percentage of 9.29%)

  "4,701 females (7.55%) & 1,087 males (1.74%)"

- 56,511 patients with scholarship (with a percentage of 90.7%)

  "35,345 females (56.73%) & 21,166 males (33.97%)"

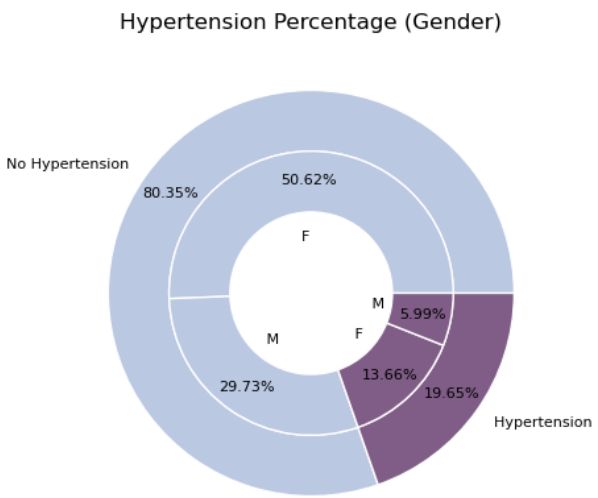
In [78]:
```
# Exploring patients with or without Hypertension (HTN):
htn_count=new_df.groupby(['age_stages','htn','gender']).patient_id.nunique().reset_index
htn_count['pct']=htn_count['count_patient']/htn_count['count_patient'].sum()
htn_count.style.format({'pct':'{:,.2%}'})
```

Out[78]:

	age_stages	htn	gender	count_patient	pct
0	adult	0	F	9938	15.95%
1	adult	0	M	4097	6.58%
2	adult	1	F	625	1.00%
3	adult	1	M	237	0.38%
4	child	0	F	6154	9.88%
5	child	0	M	6512	10.45%
6	child	1	F	2	0.00%
7	child	1	M	7	0.01%
8	elderly	0	F	1969	3.16%
9	elderly	0	M	1145	1.84%
10	elderly	1	F	3160	5.07%
11	elderly	1	M	1437	2.31%
12	middle_aged	0	F	8934	14.34%
13	middle_aged	0	M	4373	7.02%
14	middle_aged	1	F	4707	7.56%
15	middle_aged	1	M	2040	3.27%
16	teenage	0	F	4538	7.28%
17	teenage	0	M	2397	3.85%
18	teenage	1	F	19	0.03%
19	teenage	1	M	8	0.01%

In [79]:
```
# Visualizing Hypertension Column:
```

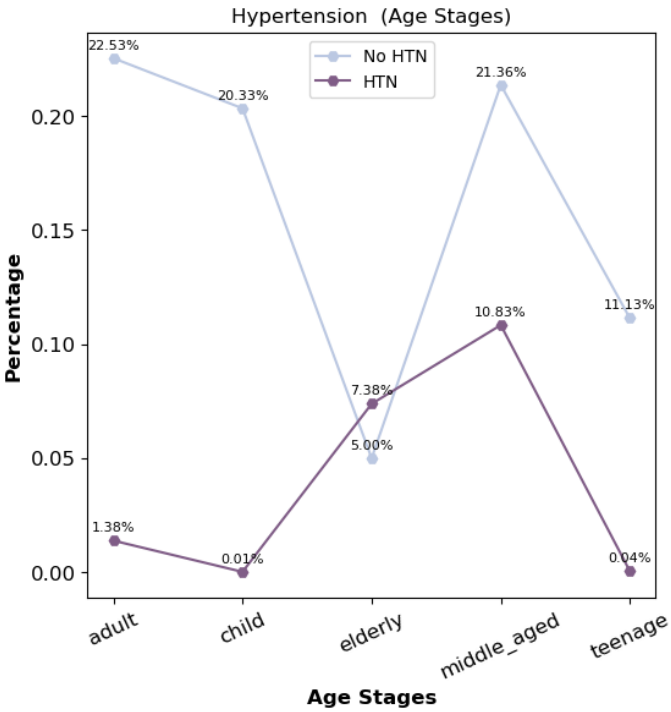
```
# By Gender:
htn_count_g=new_df.groupby(['htn','gender']).patient_id.nunique()

size = 0.3
label_s = 'No Hypertension','Hypertension '
label_g = 'F',' M',' F',' M'
text = '''
The dataset recorded:\n\n
- 12,242 patients with Hypertension (with a percentage of 19.65%)\n
    "8,513 females (13.66%) & 3,729 males (5.99%)"\n\n
 - 50,057 patients with No Hypertension (with a percentage of 80.35%)\n
    "31,533 females (50.62%) & 18,524 males (29.73%)"
    '''

plt.subplots(figsize = (5,5))
plt.pie(htn_count_g.groupby('htn').sum(), radius=1, colors= ['#bbc8e2','#805d87'],labels
        pctdistance=.85,textprops={'fontsize': 8}, wedgeprops=dict(width=size, edgecolor

plt.pie(htn_count_g, radius=1-size, colors= ['#bbc8e2','#bbc8e2','#805d87','#805d87'],la
        labeldistance=.4,textprops={'fontsize': 8},wedgeprops=dict(width=size, edgecolor

plt.title('Hypertension Percentage (Gender)')
plt.text(2,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');

# By Age Stages:
htn_count_a=new_df.groupby(['age_stages','htn']).patient_id.nunique().reset_index().rena
htn_count_a['pct']=htn_count_a['patient_count']/htn_count_a['patient_count'].sum()
stages=htn_count_a.age_stages.unique().tolist()
htn_values = htn_count_a.htn.tolist()
htn_0=htn_count_a.query('htn==0').pct.tolist()
htn_1=htn_count_a.query('htn==1').pct.tolist()


note = '''This chart shows that:\n
    - The majority of patients are not diagnosed with hypertension, \n
      except in the elderly group.\n\n
    - In this stage, patients diagnosed with hypertension make up \n
      a larger percentage (7.38%) compared to those without hypertension (5%).\n\n
    - Most patients diagnosed with hypertension fall within the Middle-Aged and\n
      Elderly Stages.'''

plt.subplots(figsize = (6,6))

plt.plot(stages,htn_0,color='#bbc8e2',marker = 'H', label= 'No HTN')
plt.plot(stages,htn_1,color='#805d87',marker = 'H', label = 'HTN')

for i, v in enumerate(htn_0):
  plt.text(i,v+.003, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

for i, v in enumerate(htn_1):
  plt.text(i,v+.003, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

plt.xticks(fontsize = 12, rotation = 25)
plt.yticks(fontsize = 12)
plt.xlabel('Age Stages',fontsize = 12, weight = 'bold')
plt.ylabel('Percentage',fontsize = 12, weight = 'bold')
plt.title('Hypertension  (Age Stages)', fontsize =12)
plt.legend(loc="upper center",fontsize = 10)
plt.text(4.5,.2,note,ha='left',va='top',fontsize = 10, weight = 'normal')
plt.show();
```



The dataset recorded:

- 12,242 patients with Hypertension (with a percentage of 19.65%)
  "8,513 females (13.66%) & 3,729 males (5.99%)"

- 50,057 patients with No Hypertension (with a percentage of 80.35%)
  "31,533 females (50.62%) & 18,524 males (29.73%)"



This chart shows that:

- The majority of patients are not diagnosed with hypertension, except in the elderly group.

- In this stage, patients diagnosed with hypertension make up a larger percentage (7.38%) compared to those without hypertension (5%).

- Most patients diagnosed with hypertension fall within the Middle-Aged and Elderly Stages.

In [80]:
```
# Exploring Diabetic & Non-Diabetic Patients:
dm_count=new_df.groupby(['age_stages','dm','gender']).patient_id.nunique().reset_index()
dm_count['pct']=htn_count['count_patient']/htn_count['count_patient'].sum()
dm_count.style.format({'pct':'{:,.2%}'})
```

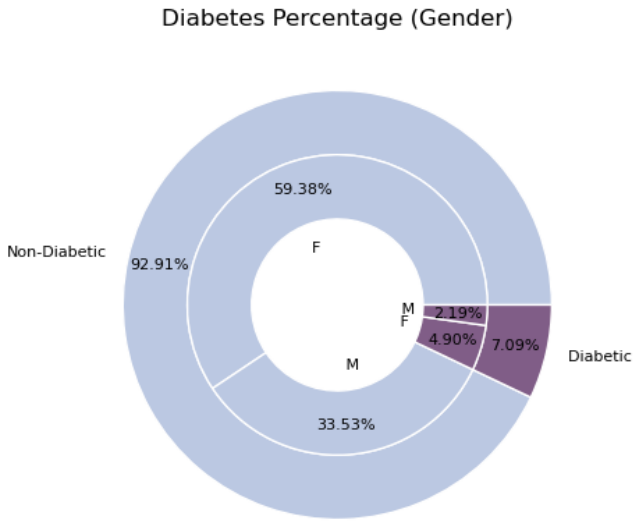
Out[80]:

	age_stages	dm	gender	count_patient	pct
0	adult	0	F	10367	15.95%
1	adult	0	M	4266	6.58%
2	adult	1	F	196	1.00%
3	adult	1	M	68	0.38%
4	child	0	F	6149	9.88%
5	child	0	M	6515	10.45%
6	child	1	F	7	0.00%
7	child	1	M	4	0.01%
8	elderly	0	F	3896	3.16%
9	elderly	0	M	2078	1.84%

10	elderly	1	F	1233	5.07%
11	elderly	1	M	504	2.31%
12	middle_aged	0	F	12043	14.34%
13	middle_aged	0	M	5631	7.02%
14	middle_aged	1	F	1598	7.56%
15	middle_aged	1	M	782	3.27%
16	teenage	0	F	4539	7.28%
17	teenage	0	M	2399	3.85%
18	teenage	1	F	18	0.03%
19	teenage	1	M	6	0.01%

In [81]:
```
# Visualizing Diabetes "dm" Column:
# By Gender:
dm_count_g=new_df.groupby(['dm','gender']).patient_id.nunique()

size = 0.3
label_s = 'Non-Diabetic','Diabetic '
label_g = 'F',' M',' F',' M'
text = '''
The dataset recorded:\n\n
- 4,416 Diabetic Patients (with a percentage of 7.09%)\n
    "3,052 females (4.9%) & 1,364 males (2.19%)"\n\n
 - 57,883 Non-diabetic Patients (with a percentage of 92.91%)\n
    "36,994 females (59.38%) & 20,889 males (33.53%)"
    '''

plt.subplots(figsize = (5,5))

plt.pie(dm_count_g.groupby('dm').sum(), radius=1, colors= ['#bbc8e2','#805d87'],labels =
        pctdistance=.85,textprops={'fontsize': 8}, wedgeprops=dict(width=size, edgecolor

plt.pie(dm_count_g, radius=1-size, colors= ['#bbc8e2','#bbc8e2','#805d87','#805d87'],lab
        labeldistance=.4,textprops={'fontsize': 8},wedgeprops=dict(width=size, edgecolor

plt.title('Diabetes Percentage (Gender)')
plt.text(2,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');

# By Age Stages:
dm_count_a=df.groupby(['age_stages','dm']).patient_id.nunique().reset_index().rename(col
dm_count_a['pct']=dm_count_a['patient_count']/dm_count_a['patient_count'].sum()
stages=dm_count_a.age_stages.unique().tolist()
dm_values = dm_count_a.dm.tolist()
dm_0=dm_count_a.query('dm==0').pct.tolist()
dm_1=dm_count_a.query('dm==1').pct.tolist()


note = '''This chart shows that:\n\n
    ~ The majority of patients are Non-Diabetic\n
    ~ Most Diabetic Patients fall within the Middle-Aged and Elderly Stages.
    '''

plt.subplots(figsize = (6,6))

plt.plot(stages,dm_0,color='#bbc8e2',marker = 'o',  label= 'No DM')
plt.plot(stages,dm_1,color='#805d87',marker = 'o',  label = 'DM')

for i, v in enumerate(dm_0):
  plt.text(i,v+.005, f"{v:.2%}", ha='center', va='bottom',fontsize=8)
```

```
for i, v in enumerate(dm_1):
  plt.text(i,v+.005, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

plt.xticks(fontsize = 12, rotation = 25)
plt.yticks(fontsize = 12)
plt.xlabel('Age Stages',fontsize = 12, weight = 'bold')
plt.ylabel('Percentage',fontsize = 12, weight = 'bold')
plt.title('Hypertension  (Age Stages)', fontsize =12)
plt.legend(loc="upper center",fontsize = 10)
plt.text(4.5,.2,note,ha='left',va='top',fontsize = 10, weight = 'normal')
plt.show();
```



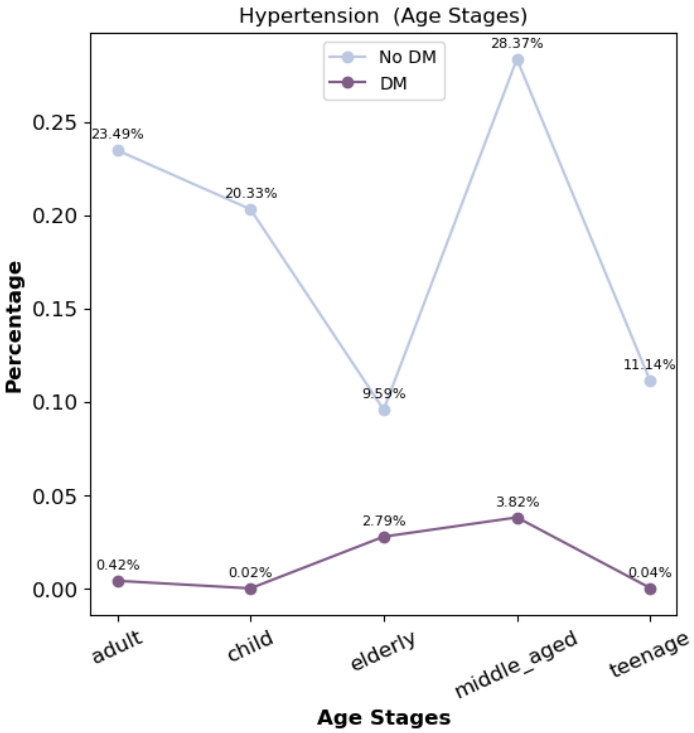
The dataset recorded:

- 4,416 Diabetic Patients (with a percentage of 7.09%)

  "3,052 females (4.9%) & 1,364 males (2.19%)"

- 57,883 Non-diabetic Patients (with a percentage of 92.91%)

  "36,994 females (59.38%) & 20,889 males (33.53%)"



This chart shows that:

~ The majority of patients are Non-Diabetic

~ Most Diabetic Patients fall within the Middle-Aged and Elderly Stages.

In [82]:
```
# Exploring patients with alcohol-related problems:
aud_count=new_df.groupby(['age_stages','aud','gender']).patient_id.nunique().reset_index
aud_count['pct']=aud_count['count_patient']/aud_count['count_patient'].sum()
aud_count.style.format({'pct':'{:,.2%}'})
```

Out[82]:

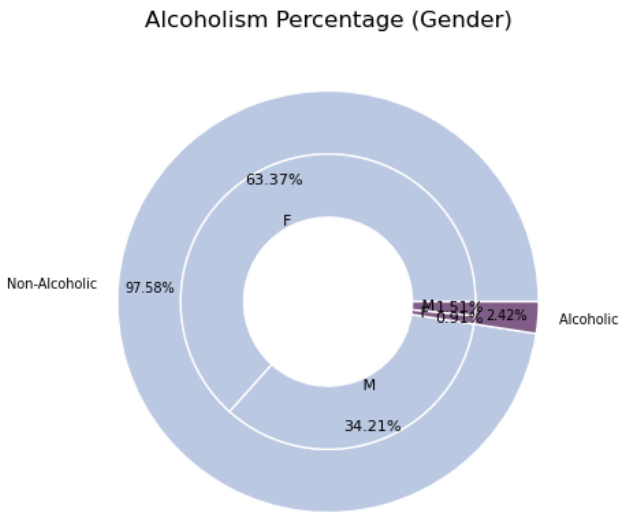
	age_stages	aud	gender	count_patient	pct
0	adult	0	F	10367	16.64%
1	adult	0	M	4177	6.70%

2	adult	1	F	196	0.31%
3	adult	1	M	157	0.25%
4	child	0	F	6155	9.88%
5	child	0	M	6514	10.46%
6	child	1	F	1	0.00%
7	child	1	M	5	0.01%
8	elderly	0	F	5091	8.17%
9	elderly	0	M	2470	3.96%
10	elderly	1	F	38	0.06%
11	elderly	1	M	112	0.18%
12	middle_aged	0	F	13320	21.38%
13	middle_aged	0	M	5756	9.24%
14	middle_aged	1	F	321	0.52%
15	middle_aged	1	M	657	1.05%
16	teenage	0	F	4545	7.30%
17	teenage	0	M	2398	3.85%
18	teenage	1	F	12	0.02%
19	teenage	1	M	7	0.01%

In [83]:

```
# Visualizing Alcoholism "dm" Column:
# By Gender:
aud_count_g=new_df.groupby(['aud','gender']).patient_id.nunique()

size = 0.3
label_s = 'Non-Alcoholic','Alcoholic '
label_g = 'F',' M',' F',' M'
text = ''' The dataset recorded:\n\n
- 1,506 patients with alcohol-related problems (with a percentage of 2.42%)\n
    "568 females (0.91%) & 938 males (1.51%)"\n\n
- 60,793 Non-Alcoholic Patients (with a percentage of 97.58%)\n
    "39,478 females (63.37%) & 21,315 males (34.21%)"
    '''

plt.subplots(figsize = (5,5))

plt.pie(aud_count_g.groupby('aud').sum(), radius=1, colors= ['#bbc8e2','#805d87'],labels
        pctdistance=.85,textprops={'fontsize': 7}, wedgeprops=dict(width=size, edgecolor

plt.pie(aud_count_g, radius=1-size, colors= ['#bbc8e2','#bbc8e2','#805d87','#805d87'],la
        labeldistance=.6,textprops={'fontsize': 8},wedgeprops=dict(width=size, edgecolor

plt.title('Alcoholism Percentage (Gender)')
plt.text(1.5,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');

# By Age Stages:
aud_count_a=new_df.groupby(['age_stages','aud']).patient_id.nunique().reset_index().rena
aud_count_a['pct']=aud_count_a['patient_count']/aud_count_a['patient_count'].sum()
stages=aud_count_a.age_stages.unique().tolist()
aud_values = aud_count_a.aud.tolist()
aud_0=aud_count_a.query('aud==0').pct.tolist()
aud_1=aud_count_a.query('aud==1').pct.tolist()
```

```
note = '''This chart shows that:\n\n
    ~ The majority of patients are Non-Alcoholic\n\n
    ~ Most patients with alcohol-related problems fall within\n
       the Middle-Aged and Elderly Stages.'''

plt.subplots(figsize = (6,6))

plt.plot(stages,aud_0,color='#bbc8e2', marker= '*', label= 'No AUD')
plt.plot(stages,aud_1,color='#805d87', marker= '*', label = 'AUD')

for i, v in enumerate(aud_0):
  plt.text(i,v+.002, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

for i, v in enumerate(aud_1):
  plt.text(i,v+.002, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

plt.xticks(fontsize = 12, rotation = 25)
plt.yticks(fontsize = 12)
plt.xlabel('Age Stages',fontsize = 12, weight = 'bold')
plt.ylabel('Percentage',fontsize = 12, weight = 'bold')
plt.title('Alcoholism  (Age Stages)', fontsize =12)
plt.legend(loc="upper center",fontsize = 10)
plt.text(4.5,.2,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



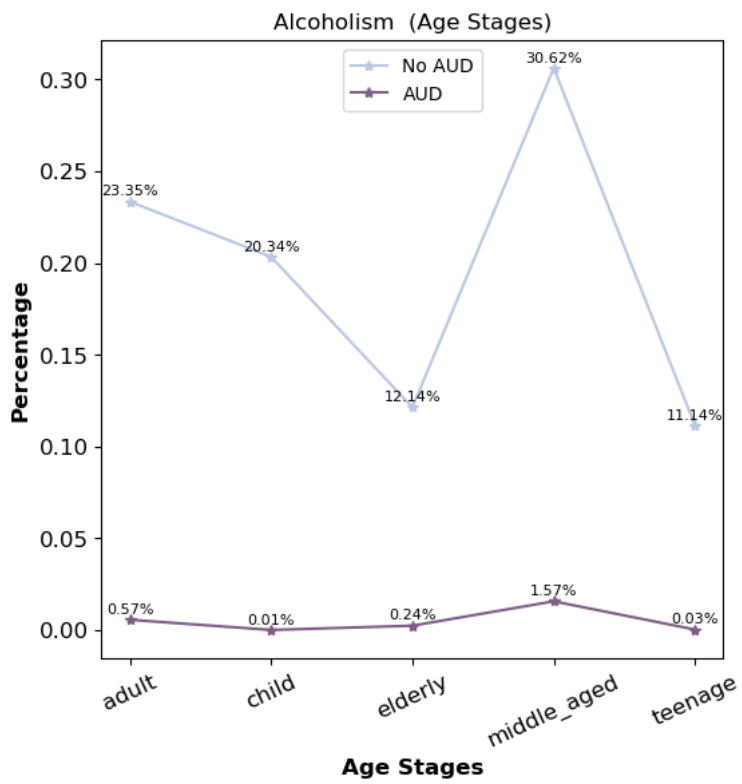
The dataset recorded:

- 1,506 patients with alcohol-related problems (with a percentage of 2.42%)

  "568 females (0.91%) & 938 males (1.51%)"

- 60,793 Non-Alcoholic Patients (with a percentage of 97.58%)

  "39,478 females (63.37%) & 21,315 males (34.21%)"



This chart shows that:

~ The majority of patients are Non-Alcoholic

~ Most patients with alcohol-related problems fall within the Middle-Aged and Elderly Stages.

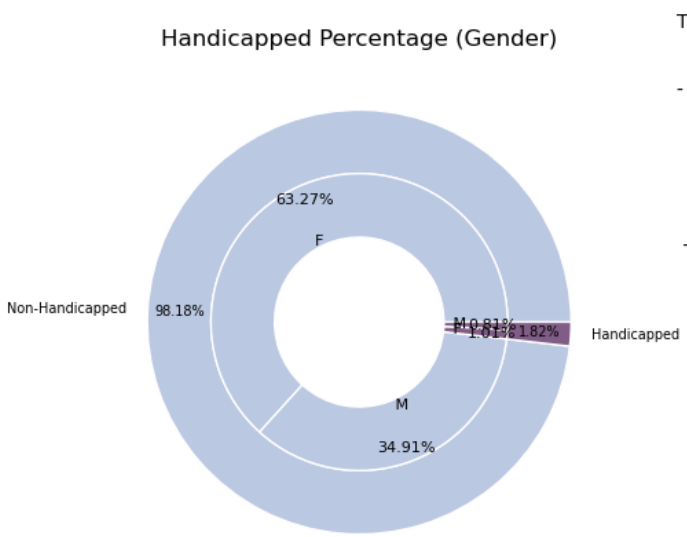
In [84]:
```
# Exploring patients with a handicap:
hcp_count=new_df.groupby(['age_stages','hcp','gender']).patient_id.nunique().reset_index
hcp_count['pct']=hcp_count['count_patient']/hcp_count['count_patient'].sum()
hcp_count.style.format({'pct':'{:,.2%}'})
df.groupby(['hcp','gender']).patient_id.nunique().reset_index().rename(columns={'patient
```

Out[84]:

	hcp	gender	count_patient
0	0	F	39417
1	0	M	21749
2	1	F	629
3	1	M	504

In [85]:
```
# Visualizing Alcoholism "dm" Column:
# By Gender:
hcp_count_g=new_df.groupby(['hcp','gender']).patient_id.nunique()


size = 0.3
label_s = 'Non-Handicapped','Handicapped '
label_g = 'F',' M',' F',' M'
text = '''
The dataset recorded:\n\n
- 1,133 patients were classified as having a handicap, ranging from low \n
  to severe levels  (with a percentage of 1.82% of the total)\n
    "629 females (1.01%) & 504 males (0.81%)"\n\n
 - There were 61,166 non-handicap patients (98.18% of the total)\n
    "39,417 females (63.27%) & 21,749 males (34.91%)"
    '''

plt.subplots(figsize = (5,5))

plt.pie(hcp_count_g.groupby('hcp').sum(), radius=1, colors= ['#bbc8e2','#805d87'],labels
        pctdistance=.85,textprops={'fontsize': 7}, wedgeprops=dict(width=size, edgecolor

plt.pie(hcp_count_g, radius=1-size, colors= ['#bbc8e2','#bbc8e2','#805d87','#805d87'],la
```

```
        labeldistance=.6,textprops={'fontsize': 8},wedgeprops=dict(width=size, edgecolor

plt.title('Handicapped Percentage (Gender)')
plt.text(1.5,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');

# By Age Stages:
hcp_count_a=new_df.groupby(['age_stages','hcp']).patient_id.nunique().reset_index().rena
hcp_count_a['pct']=hcp_count_a['patient_count']/hcp_count_a['patient_count'].sum()
stages=hcp_count_a.age_stages.unique().tolist()
hcp_values = hcp_count_a.hcp.tolist()
hcp_0=hcp_count_a.query('hcp==0').pct.tolist()
hcp_1=hcp_count_a.query('hcp==1').pct.tolist()

note = '''This chart shows that:\n\n
    ~ The majority of patients are not Handicapped\n
    ~ Most handicapped patients fall within the Middle-Aged and Elderly Stages.
    '''

plt.subplots(figsize = (6,6))

plt.plot(stages,hcp_0,color='#bbc8e2', marker= "D", label= 'No HCP')
plt.plot(stages,hcp_1,color='#805d87', marker= "D", label = 'HCP')

for i, v in enumerate(hcp_0):
  plt.text(i,v+.005, f"{v:.2%}", ha='center', va='bottom',fontsize=8)

for i, v in enumerate(hcp_1):
  plt.text(i,v+.005, f"{v:.2%}", ha='center', va='bottom',fontsize=9)

plt.xticks(fontsize = 12, rotation = 25)
plt.xlabel('Age Stages',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12)
plt.ylabel('Percentage',fontsize = 12, weight = 'bold')
plt.title('Handicap  (Age Stages)', fontsize =12)
plt.legend(loc="upper center",fontsize = 10)
plt.text(4.5,.2,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



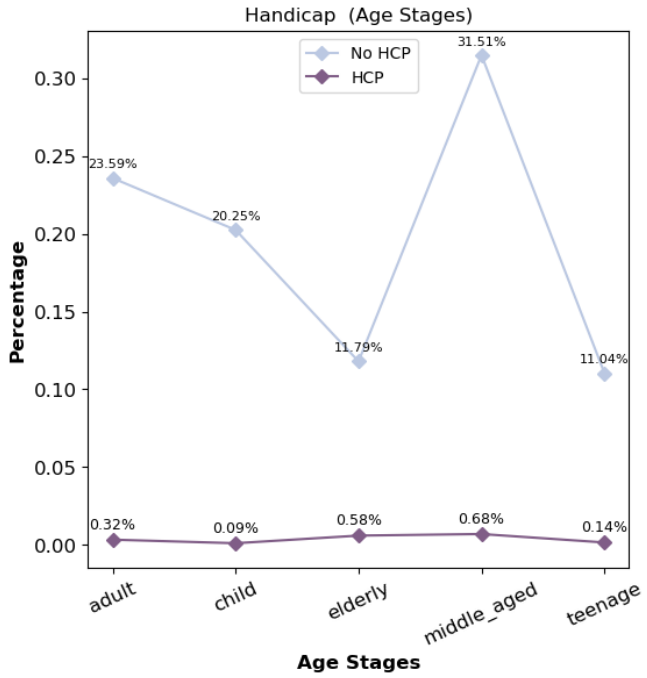
The dataset recorded:

- 1,133 patients were classified as having a handicap, ranging from low to severe levels (with a percentage of 1.82% of the total)

  "629 females (1.01%) & 504 males (0.81%)"

- There were 61,166 non-handicap patients (98.18% of the total)

  "39,417 females (63.27%) & 21,749 males (34.91%)"



This chart shows that:

~ The majority of patients are not Handicapped

~ Most handicapped patients fall within the Middle-Aged and Elderly Stages.

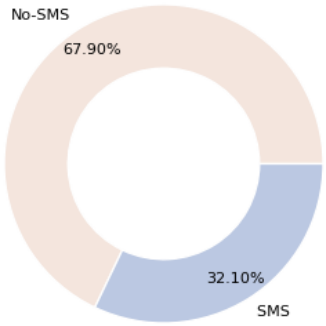
In [86]:
```
# Exploring ther charcteristics of SMS_Recieved Column:
new_df.groupby('sms_received').appoint_id.count().reset_index().rename(columns={'appoint
```

Out[86]:

	sms_received	appointments_number
0	0	75045
1	1	35482

In [87]:
```
# Visualizing sms_received Column:
sms_count=new_df.groupby('sms_received').appoint_id.count()
size = 0.3
label_s = 'No-SMS','SMS '

text = ''' The dataset recorded:\n\n
    - 35,482 appointment confirmations were sent to patients (32.1% of total appointment
    - 75,045 scheduled appointments had no confirmation messages sent (67.9%).'''

plt.subplots(figsize = (5,5))

plt.pie(sms_count, radius=.75, colors= ['#f4e5dd','#bbc8e2'],labels = label_s,autopct='%
        pctdistance=.85,textprops={'fontsize': 8}, wedgeprops=dict(width=size, edgecolor

plt.title('Appointment Confirmation Messages')
plt.text(1,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');
```

Appointment Confirmation Messages



The dataset recorded:

- 35,482 appointment confirmations were sent to patients (32.1% of total appointments).
- 75,045 scheduled appointments had no confirmation messages sent (67.9%).

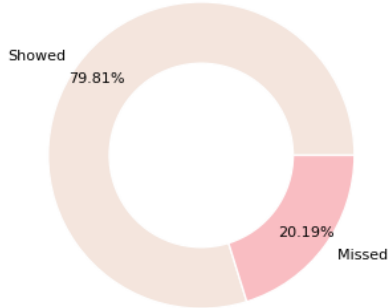
In [88]:
```
new_df.groupby('no_show').appoint_id.count().reset_index().rename(columns={'appoint_id':
```

Out[88]:

	no_show	appointments_number
0	No	88208
1	Yes	22319

In [89]:
```
# Visualizing no_show Column:
show_count=new_df.groupby('no_show').appoint_id.count()
size = 0.3
label_s = 'Showed','Missed '

text = ''' The dataset recorded:\n\n
    - 88,208 patients attended their scheduled appointments (79.81% of total appointment
    - 22,319 patients missed their appointments (20.19%).'''

plt.subplots(figsize = (5,5))

plt.pie(show_count, radius=.75, colors= ['#f4e5dd','#f9bdc2'],labels = label_s,autopct='
        pctdistance=.85,textprops={'fontsize': 8}, wedgeprops=dict(width=size, edgecolor

plt.title('Percentage of Attendance for Scheduled Appointments')
plt.text(1,0,text,ha='left',va='bottom',fontsize = 10, weight = 'normal');
```

Percentage of Attendance for Scheduled Appointments



The dataset recorded:

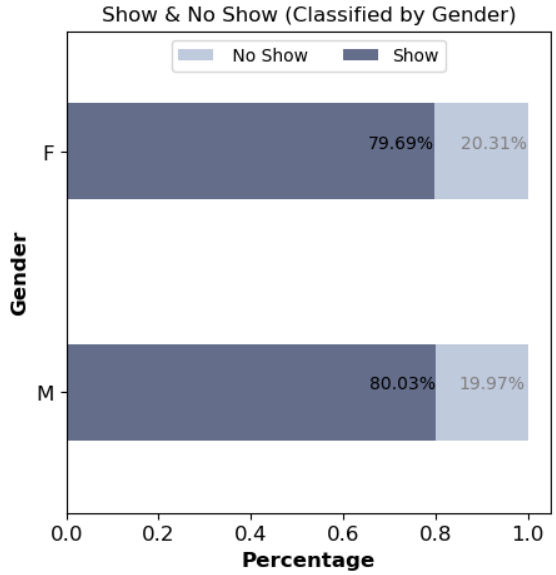
- 88,208 patients attended their scheduled appointments (79.81% of total appointments).
- 22,319 patients missed their appointments (20.19%).

Questions to be asked:

1. Do gender differences impact showing up to the appointment?
2. Does the time between the scheduled date and the appointment date impact the likelihood of showing up?
3. Does the patient's age affect their likelihood of attending their appointment?
4. What is the impact of the neighborhood on the level of commitment to showing up for appointments?
5. Is there a relationship between acquiring the Bolsa Família scholarship and the percentage of attendance?
6. Does a diagnosis of hypertension, diabetes, alcoholism, or disability impact the level of appointment attendance?
7. Does receiving messages impact patients' likelihood of attending their appointments?

Q1. Do gender differences impact showing up to the appointment?

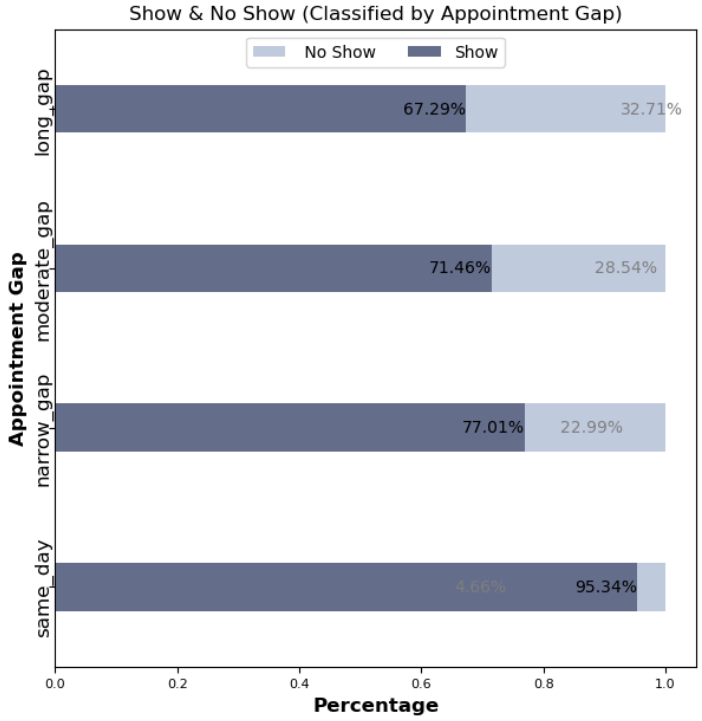
In [92]:
```
appoint_count_g=new_df.groupby('gender').appoint_id.count().reset_index().rename(columns
appoint_count_g['pct']=appoint_count_g.num_appointments/appoint_count_g.num_appointments

appoint_show_g= new_df.query('no_show == "No"').groupby('gender').appoint_id.count().res
appoint_show_g['pct']=appoint_show_g.num_appointments/appoint_count_g.num_appointments

note = '''With 79.69% of females attending their appointments, a figure close to the mal
attendance rate of 80.03%, this indicates that gender has no significant impact on \n
appointment attendance.'''

plt.subplots(figsize = (5,5))

sns.barplot(x=appoint_count_g.pct,y=appoint_show_g.gender.tolist() ,color='#bbc8e2',labe
sns.barplot(x=appoint_show_g.pct,y=appoint_show_g.gender.tolist(),color = "#5e6b91", lab

for i, v in enumerate(appoint_count_g.pct-appoint_show_g.pct):
  plt.text(v+.65,i, f"{v:.2%}", ha='left', va='bottom',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_g.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='bottom',fontsize=10)

plt.xticks(fontsize = 12)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12)
plt.ylabel('Gender',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Gender)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



With 79.69% of females attending their appointments, a figure close to the male attendance rate of 80.03%, this indicates that gender has no significant impact on appointment attendance.

Q2. Does the time between the scheduled date and the appointment date impact the likelihood of showing up?

In [94]:
```
appoint_count_a=new_df.groupby('appoint_gap').appoint_id.count().reset_index().rename(co
appoint_count_a['pct']=appoint_count_a.num_appointments/appoint_count_a.num_appointments

appoint_show_a= new_df.query('no_show == "No"').groupby('appoint_gap').appoint_id.count(
appoint_show_a['pct']=appoint_show_a.num_appointments/appoint_count_a.num_appointments

note = '''
- 95.34% of patients attended their appointments when scheduled \n
  on the same day.\n \n
- Attendance dropped to 77.01% for appointments scheduled \n
  within 1 to 4 days.\n\n
- 71.46% for those scheduled within 5 to 15 days.\n\n
- 67.29% for appointments scheduled with a gap of more than 15 days.\n\n \n
        This pattern indicates that the longer the scheduling gap,\n
                    the lower the attendance rate.'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_a.pct,y=appoint_show_a.appoint_gap.tolist() ,color='#bbc8e2'
sns.barplot(x=appoint_show_a.pct,y=appoint_show_a.appoint_gap.tolist(),color = "#5e6b91"

for i, v in enumerate(appoint_count_a.pct-appoint_show_a.pct):
  plt.text(v+.65,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_a.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Appointment Gap',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Appointment Gap)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



- 95.34% of patients attended their appointments when scheduled on the same day.

- Attendance dropped to 77.01% for appointments scheduled within 1 to 4 days.

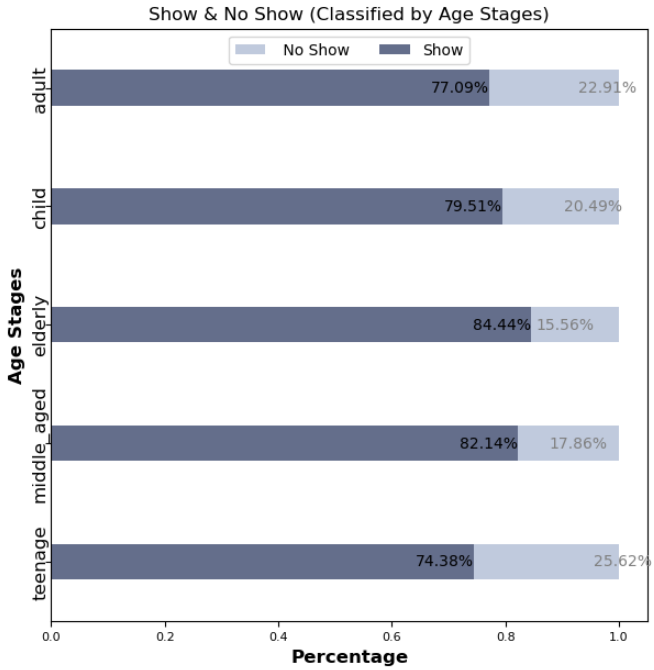
- 71.46% for those scheduled within 5 to 15 days.

- 67.29% for appointments scheduled with a gap of more than 15 days.

This pattern indicates that the longer the scheduling gap, the lower the attendance rate.

Q3. Does the patient's age affect their likelihood of attending their appointment?

In [96]:
```
appoint_count_s=new_df.groupby('age_stages').appoint_id.count().reset_index().rename(col
appoint_count_s['pct']=appoint_count_s.num_appointments/appoint_count_s.num_appointments

appoint_show_s= new_df.query('no_show == "No"').groupby('age_stages').appoint_id.count()
appoint_show_s['pct']=appoint_show_s.num_appointments/appoint_count_s.num_appointments

note = '''
- The Elderly category shows the highest commitment to attending appointments, \n
  with a percentage of 84.44%.\n\n
- This is followed by the Middle-aged category at 82.14%.\n\n
- The Child stage comes next with a percentage of 79.51%.\n\n
     "These three categories generally show the highest frequency of hospital \n
      visits for regular check-ups, which explains their higher commitment \n
      to attending appointments."\n\n\n
- The Adult and Teenage stages have the lowest attendance rates,\n
  with percentages of 77.09% and 74.38%, respectively.\n\n
        This pattern indicates that age stages have a positive \n
                  impact on the attendance rate.'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_s.pct,y=appoint_count_s.age_stages.tolist() ,color='#bbc8e2'
sns.barplot(x=appoint_show_s.pct,y=appoint_show_s.age_stages.tolist(),color = "#5e6b91",

for i, v in enumerate(appoint_count_s.pct-appoint_show_s.pct):
  plt.text(v+.75,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_s.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Age Stages',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Age Stages)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



- The Elderly category shows the highest commitment to attending appointments, with a percentage of 84.44%.

- This is followed by the Middle-aged category at 82.14%.

- The Child stage comes next with a percentage of 79.51%.

"These three categories generally show the highest frequency of hospital visits for regular check-ups, which explains their higher commitment to attending appointments."

- The Adult and Teenage stages have the lowest attendance rates, with percentages of 77.09% and 74.38%, respectively.

This pattern indicates that age stages have a positive impact on the attendance rate.

Q4. What is the impact of the neighborhood on the level of commitment to showing up for appointments?

In [98]:
```
# The Total Appointments scheduled are distributed on 81 Neighbourhoods:
appoint_count_n=new_df.groupby('neighbourhood').appoint_id.count()
appoint_count_n=appoint_count_n.sort_values()
appoint_count_n = appoint_count_n.reset_index().rename(columns={'appoint_id':'num_appoin
appoint_count_n.sort_values('neighbourhood').shape
```

Out[98]:
```
(81, 2)
```

In [99]:
```
# The total attended appointments are distributed into 80 Neigbourhoods:
appoint_show_N= new_df.query('no_show == "No"').groupby('neighbourhood').appoint_id.coun
appoint_show_N.shape
```

Out[99]:
```
(80, 2)
```

In [100…
```
# Determining the missing neighbourhood:
total_appointments = set(appoint_count_n['neighbourhood'])
total_show = set(appoint_show_N['neighbourhood'])
only_in_appointments = total_appointments - total_show
only_in_appointments
```

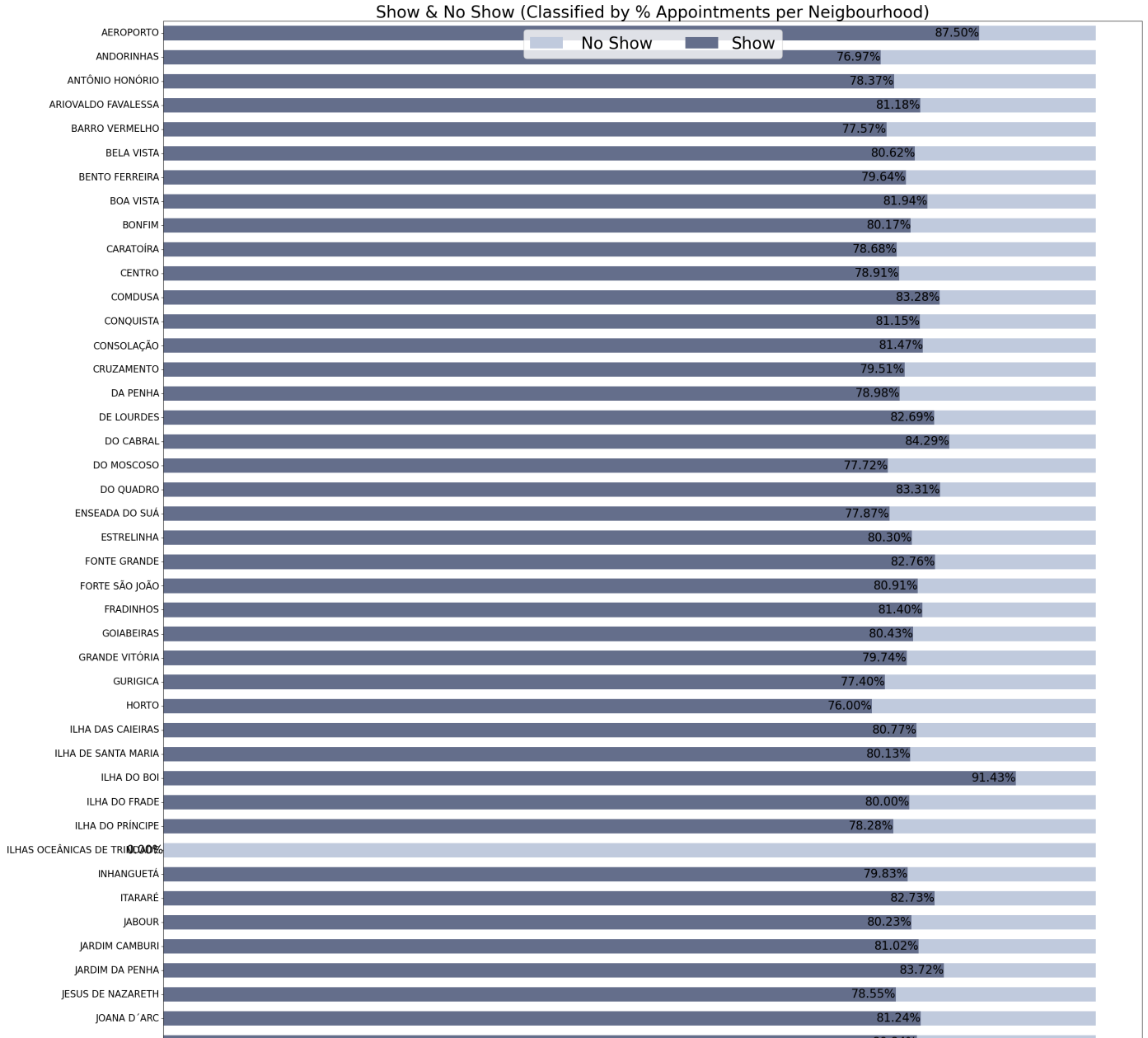
Out[100]:
```
{'ILHAS OCEÂNICAS DE TRINDADE'}
```

In [101…
```
# The Scheduled appointments at "ILHAS OCEÂNICAS DE TRINDADE" Neighbourhood weren't atte
# Add Row for "ILHAS OCEÂNICAS DE TRINDADE" with 0 value.
appoint_show_n= new_df.query('no_show == "No"').groupby('neighbourhood').appoint_id.coun
appoint_show_n['ILHAS OCEÂNICAS DE TRINDADE'] = 0
appoint_show_n=appoint_show_n.sort_values()
appoint_show_n = appoint_show_n.reset_index().rename(columns={'appoint_id':'num_appointm
appoint_show_n.sort_values('neighbourhood').shape
```

Out[101]:
```
(81, 2)
```

In [102…
```
# Attendance Rate per Neighbourhood:
appoint_count_n['pct']=appoint_count_n.sort_values('neighbourhood').num_appointments/app
appoint_show_n['pct']=appoint_show_n.sort_values('neighbourhood').num_appointments/appoi

note = '''
```

```
- "ILHA Do Boi" has recorded the highest attendance rate (91.43%),\n
  despite not being among the top ten hospital locations in terms \n
  of appointment count or number of recorded patients.\n \n
- This is followed by "AEROPORTO" with an attendance rate of 87.5%,\n
  which is also not part of the top ten locations mentioned earlier.\n
- In contrast, "JARDIM CAMBURI," classified as the top hospital \n
  location for both recorded patients and appointment count,\n
  has an attendance rate of 81.02%.'''

plt.subplots(figsize = (30,60))

sns.barplot(x=appoint_count_n.sort_values('neighbourhood').pct,y=appoint_count_n.sort_va
            color='#bbc8e2',label=" No Show",width=.6)
sns.barplot(x=appoint_show_n.sort_values('neighbourhood').pct,y=appoint_count_n.sort_val
            color = "#5e6b91", label= "Show",width=.6)


for i, v in enumerate(appoint_show_n.sort_values('neighbourhood').pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=20)

plt.xticks(fontsize = 16)
plt.yticks(fontsize = 16, rotation = 0, va='center')
plt.title('Show & No Show (Classified by % Appointments per Neigbourhood)', fontsize =28
plt.legend(ncol=2,loc="upper center",fontsize = 28)
plt.text(0,87,note,ha='left',va='center',fontsize =25, weight = 'normal')
plt.show();
```



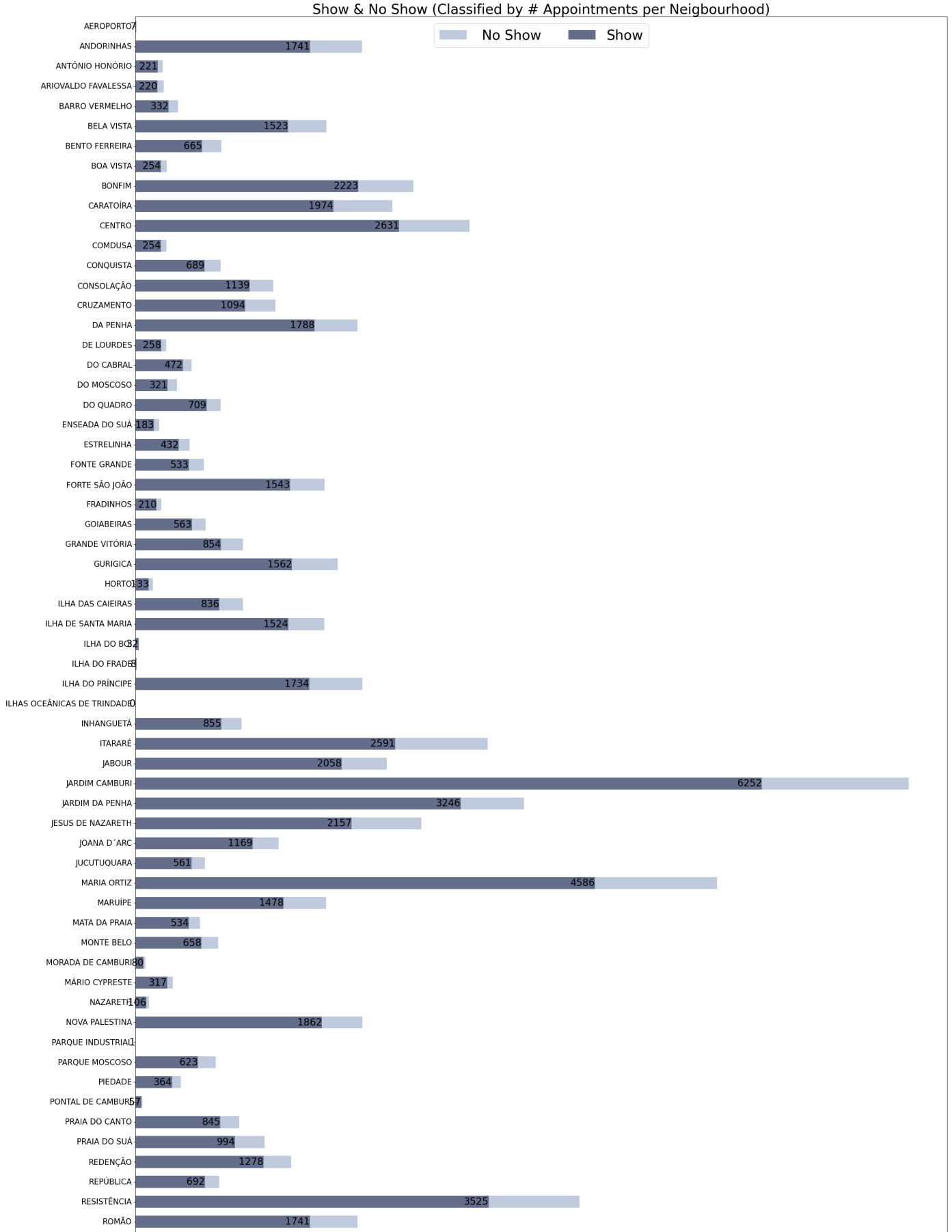
- "ILHA Do Boi" has recorded the highest attendance rate (91.43%), despite not being among the top ten hospital locations in terms of appointment count or number of recorded patients.

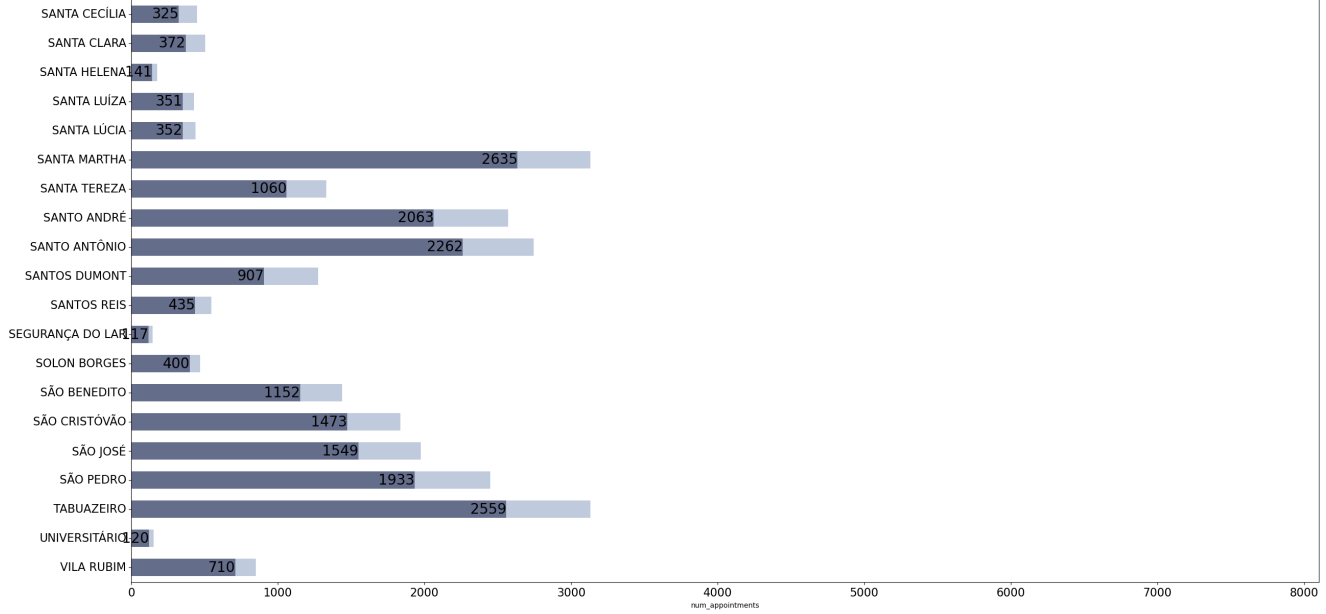
- This is followed by "AEROPORTO" with an attendance rate of 87.5%, which is also not part of the top ten locations mentioned earlier.
- In contrast, "JARDIM CAMBURI," classified as the top hospital location for both recorded patients and appointment count, has an attendance rate of 81.02%.

In [103…

```
# Number of Scheduled Appointments per Neighbiurhood:
note = '''
- Upon reviewing the actual number of appointments,\n
  it was found that neighborhoods with the highest attendance rates \n
  tend to have the lowest number of appointments per neighborhood,\n
  which may explain their higher attendance rates compared to others.
- Neighborhoods with the lowest number of scheduled appointments\n
  have a higher attendance rate compared to those with the highest\n
  number of scheduled appointments.'''

plt.subplots(figsize = (30,60))

sns.barplot(x=appoint_count_n.sort_values('neighbourhood').num_appointments,y=appoint_co
            color='#bbc8e2',label=" No Show",width=.6)
sns.barplot(x=appoint_show_n.sort_values('neighbourhood').num_appointments,y=appoint_cou
            color = "#5e6b91", label= "Show",width=.6)
```

```
for i, v in enumerate(appoint_show_n.sort_values('neighbourhood').num_appointments):
  plt.text(v,i, f"{v:.0f}", ha='right', va='center',fontsize=20)

plt.xticks(fontsize = 16)
plt.yticks(fontsize = 16, rotation = 0, va='center')
plt.title('Show & No Show (Classified by # Appointments per Neigbourhood)', fontsize =28
plt.legend(ncol=2,loc="upper center",fontsize = 28)
plt.text(.2,87,note,ha='left',va='center',fontsize =25, weight = 'normal')
plt.show();
```



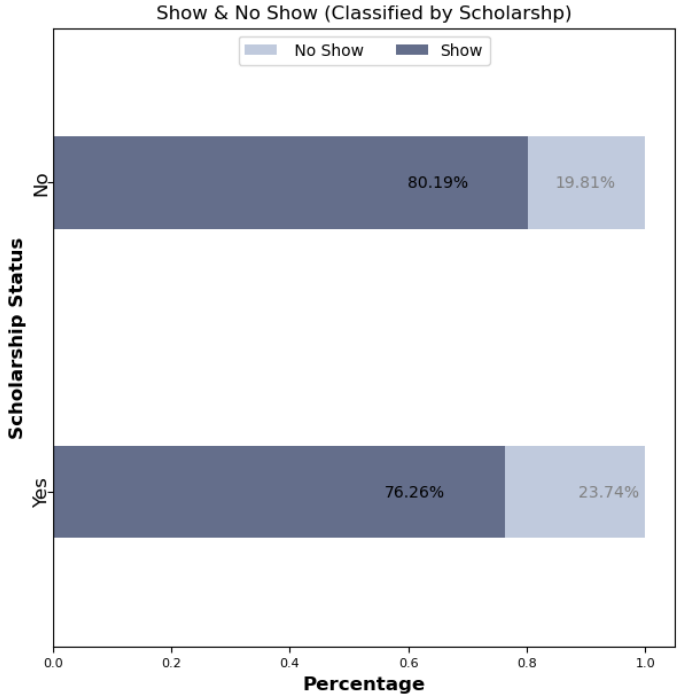


- Upon reviewing the actual number of appointments,
  it was found that neighborhoods with the highest attendance rates
  tend to have the lowest number of appointments per neighborhood,
  which may explain their higher attendance rates compared to others.
- Neighborhoods with the lowest number of scheduled appointments
  have a higher attendance rate compared to those with the highest
  number of scheduled appointments.

Q5. Is there a relationship between acquiring the Bolsa Família scholarship and the percentage of attendance?

In [105…

```
appoint_count_s=new_df.groupby('scholarship').appoint_id.count().reset_index().rename(co
appoint_count_s['pct']=appoint_count_s.num_appointments/appoint_count_s.num_appointments

appoint_show_s= new_df.query('no_show == "No"').groupby('scholarship').appoint_id.count(
appoint_show_s['pct']=appoint_show_s.num_appointments/appoint_count_s.num_appointments

note = '''
- Patients without scholarships are more likely to attend their appointments,\n
  with a rate of 80.19%, compared to 76.26% for those with scholarships.'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_s.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",width
sns.barplot(x=appoint_show_s.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width=.

for i, v in enumerate(appoint_count_s.pct-appoint_show_s.pct):
  plt.text(v+.7,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_s.pct):
  plt.text(v-.1,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Scholarship Status',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Scholarshp)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```

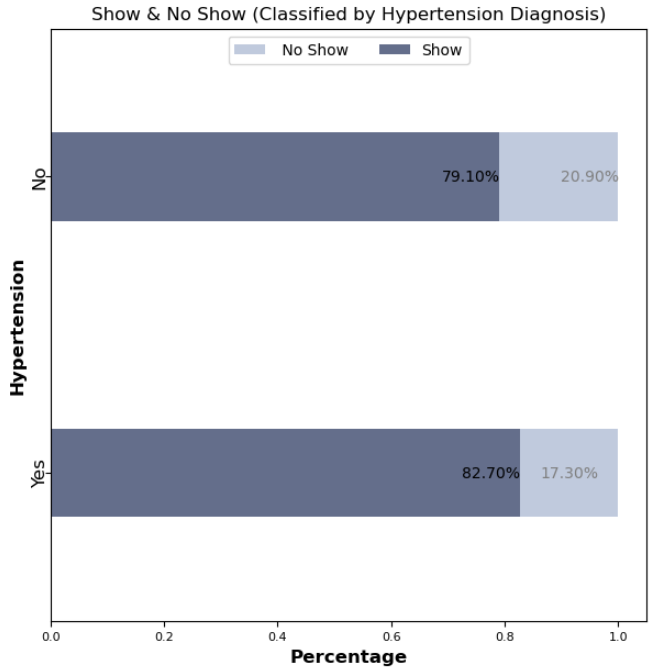


- Patients without scholarships are more likely to attend their appointments, with a rate of 80.19%, compared to 76.26% for those with scholarships.

Q6. Does a diagnosis of hypertension, diabetes, alcoholism, or disability impact the level of appointment attendance?

In [107…

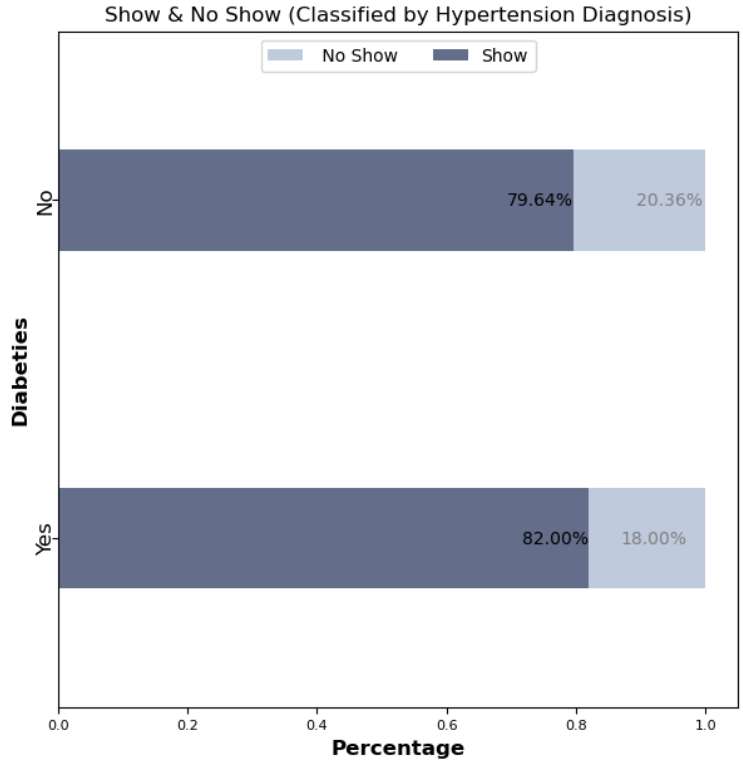
```
# 1 - Hypertension (HTN):
appoint_count_h=new_df.groupby('htn').appoint_id.count().reset_index().rename(columns={'
appoint_count_h['pct']=appoint_count_h.num_appointments/appoint_count_h.num_appointments

appoint_show_h= new_df.query('no_show == "No"').groupby('htn').appoint_id.count().reset_
appoint_show_h['pct']=appoint_show_h.num_appointments/appoint_count_h.num_appointments

note = '''
Patients with Hypertension Diagnosis are more likely to attend their appointments,\n
with a rate of 82.70%, compared to 79.1% for those who weren't diagnoses with \n
Hypertension.'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_h.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",width
sns.barplot(x=appoint_show_h.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width=.

for i, v in enumerate(appoint_count_h.pct-appoint_show_h.pct):
  plt.text(v+.74,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_h.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Hypertension',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Hypertension Diagnosis)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



Patients with Hypertension Diagnosis are more likely to attend their appointments, with a rate of 82.70%, compared to 79.1% for those who weren't diagnoses with Hypertension.

In [108…

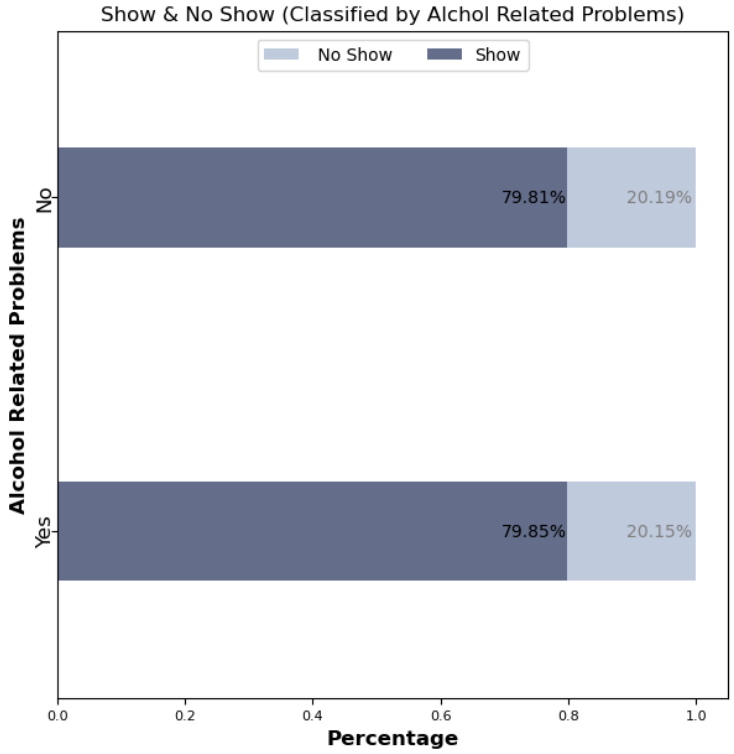
```
# 2 - Diabetes (DM):
appoint_count_d=new_df.groupby('dm').appoint_id.count().reset_index().rename(columns={'a
appoint_count_d['pct']=appoint_count_d.num_appointments/appoint_count_d.num_appointments

appoint_show_d= new_df.query('no_show == "No"').groupby('dm').appoint_id.count().reset_i
appoint_show_d['pct']=appoint_show_d.num_appointments/appoint_count_d.num_appointments

note = '''
Diabetic Patients are more likely to attend their appointments,\n
with a rate of 82%, compared to 79.64% Non-Diabetic'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_d.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",width
sns.barplot(x=appoint_show_d.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width=.

for i, v in enumerate(appoint_count_d.pct-appoint_show_d.pct):
  plt.text(v+.74,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_d.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Diabeties',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Hypertension Diagnosis)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



Diabetic Patients are more likely to attend their appointments, with a rate of 82%, compared to 79.64% Non-Diabetic

In [109…

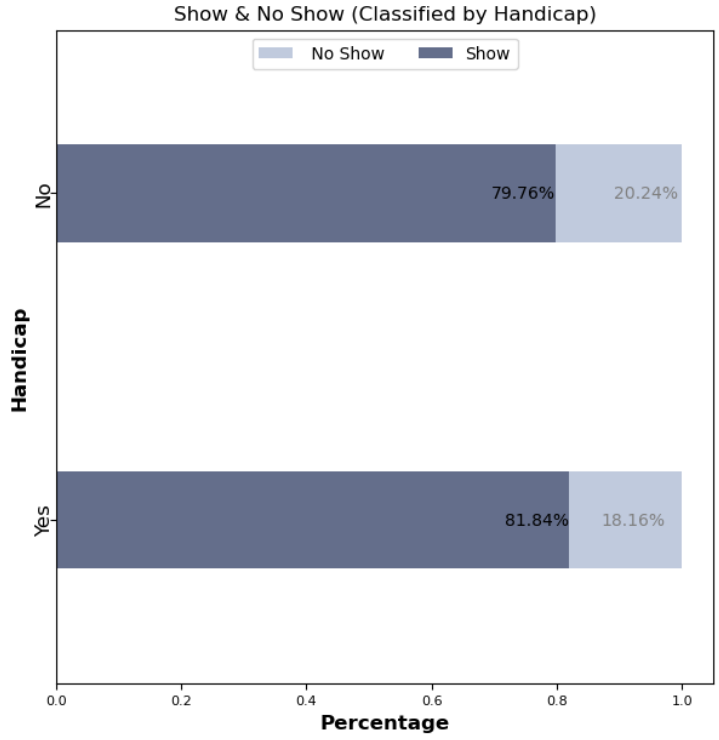
```
# 3 - Alcoholism (AUD):
appoint_count_u=new_df.groupby('aud').appoint_id.count().reset_index().rename(columns={'
appoint_count_u['pct']=appoint_count_u.num_appointments/appoint_count_u.num_appointments

appoint_show_u= new_df.query('no_show == "No"').groupby('aud').appoint_id.count().reset_
appoint_show_u['pct']=appoint_show_u.num_appointments/appoint_count_u.num_appointments

note = '''
Patients with and without Alcohol Related Probelems are almost
have the same attendence rate'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_u.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",width
sns.barplot(x=appoint_show_u.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width=.

for i, v in enumerate(appoint_count_u.pct-appoint_show_u.pct):
  plt.text(v+.74,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_u.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Alcohol Related Problems',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Alchol Related Problems)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



Patients with and without Alcohol Related Probelems are almost have the same attendence rate

In [110…

```
# 4 - Handicap (HCP):
appoint_count_p=new_df.groupby('hcp').appoint_id.count().reset_index().rename(columns={'
appoint_count_p['pct']=appoint_count_p.num_appointments/appoint_count_p.num_appointments

appoint_show_p= new_df.query('no_show == "No"').groupby('hcp').appoint_id.count().reset_
appoint_show_p['pct']=appoint_show_p.num_appointments/appoint_count_p.num_appointments

note = '''
Handicapped Patients are more likely to attend their appointments,\n
with a rate of 81.84%, compared to 79.76% with no Handicap'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_p.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",width
sns.barplot(x=appoint_show_p.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width=.

for i, v in enumerate(appoint_count_p.pct-appoint_show_p.pct):
  plt.text(v+.74,i, f"{v:.2%}", ha='center', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_p.pct):
  plt.text(v,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('Handicap',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Handicap)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```

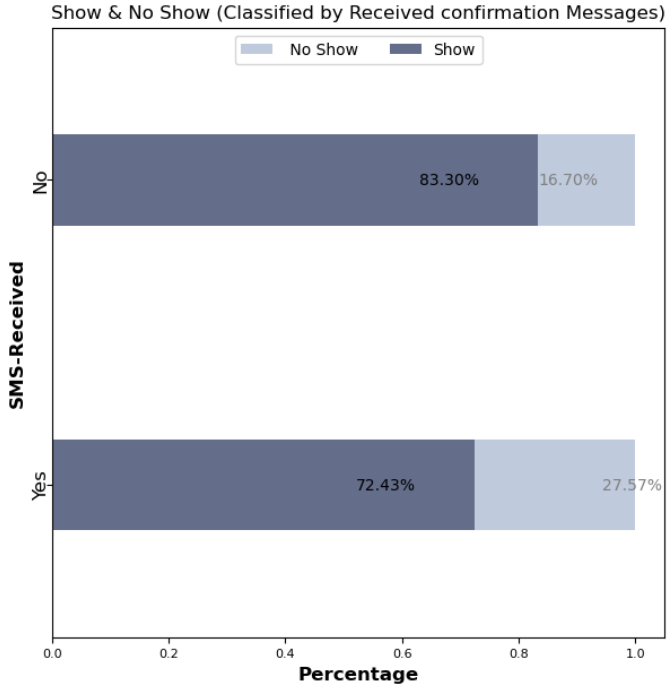


Handicapped Patients are more likely to attend their appointments, with a rate of 81.84%, compared to 79.76% with no Handicap

Q7. Does receiving messages impact patients' likelihood of attending their appointments?

In [112…

```
appoint_count_sms=new_df.groupby('sms_received').appoint_id.count().reset_index().rename
appoint_count_sms['pct']=appoint_count_sms.num_appointments/appoint_count_sms.num_appoin

appoint_show_sms= new_df.query('no_show == "No"').groupby('sms_received').appoint_id.cou
appoint_show_sms['pct']=appoint_show_sms.num_appointments/appoint_count_sms.num_appointm

note = '''
The attendance rate was 83.30% for patients who received an appointment
confirmation message, but it dropped to 72.43% for those who did not receive
a confirmation.'''

plt.subplots(figsize = (7,7))

sns.barplot(x=appoint_count_sms.pct,y=['No','Yes'] ,color='#bbc8e2',label=" No Show",wid
sns.barplot(x=appoint_show_sms.pct,y=['No','Yes'],color = "#5e6b91", label= "Show",width

for i, v in enumerate(appoint_count_sms.pct-appoint_show_sms.pct):
  plt.text(v+.77,i, f"{v:.2%}", ha='right', va='center',fontsize=10,color="#808080")

for i, v in enumerate(appoint_show_sms.pct):
  plt.text(v-.1,i, f"{v:.2%}", ha='right', va='center',fontsize=10)

plt.xticks(fontsize = 8)
plt.xlabel('Percentage',fontsize = 12, weight = 'bold')
plt.yticks(fontsize = 12, rotation = 90, ha='center',va='center')
plt.ylabel('SMS-Received',fontsize = 12,weight = 'bold')
plt.title('Show & No Show (Classified by Received confirmation Messages)', fontsize =12)
plt.legend(ncol=2,loc="upper center",fontsize = 10)
plt.text(1.1,0,note,ha='left',va='top',fontsize = 11, weight = 'normal')
plt.show();
```



The attendance rate was 83.30% for patients who received an appointment confirmation message, but it dropped to 72.43% for those who did not receive a confirmation.