Project: Investigate a Dataset (Medical Appointments)

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

Table of Contents

Introduction

Questions

1. Number of male and female in gender column?
2. What number showed and did not show up for appointment ?
3. Is there a relationship between age and appointment?
4. What is the average wait time between ScheduledDay and AppointmentDay?
5. Is there a relationship between age and hypertension?

In [70]:
```
# Use this cell to set up import statements for all of the packages that you
#   plan to use.
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
# Remember to include a 'magic word' so that your visualizations are plotted
#   inline with the notebook. See this page for more:
#   http://ipython.readthedocs.io/en/stable/interactive/magics.html
%matplotlib inline
```

Data Wrangling

General Properties

In [71]:
```
# Load your data and print out a few lines. Perform operations to inspect data
df = pd.read_csv('../Dataset/KaggleV2-May-2016.csv')
#   types and look for instances of missing or possibly errant data.
```

In [72]:
```
df.head(4)
```

Out[72]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
0	2.987250e+13	5642903	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	5.589978e+14	5642503	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0	0	0	0	0	No
2	4.262962e+12	5642549	F	2016-04-29T16:19:04Z	2016-04-29T00:00:00Z	62	MATA DA PRAIA	0	0	0	0	0	0	No
3	8.679512e+11	5642828	F	2016-04-29T17:29:31Z	2016-04-29T00:00:00Z	8	PONTAL DE CAMBURI	0	0	0	0	0	0	No

In [73]:
```
df.tail(4)
```

Out[73]:

	PatientId	AppointmentID	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
110523	3.596266e+12	5650093	F	2016-05-03T07:27:33Z	2016-06-07T00:00:00Z	51	MARIA ORTIZ	0	0	0	0	0	1	No
110524	1.557663e+13	5630692	F	2016-04-27T16:03:52Z	2016-06-07T00:00:00Z	21	MARIA ORTIZ	0	0	0	0	0	1	No
110525	9.213493e+13	5630323	F	2016-04-27T15:09:23Z	2016-06-07T00:00:00Z	38	MARIA ORTIZ	0	0	0	0	0	1	No
110526	3.775115e+14	5629448	F	2016-04-27T13:30:56Z	2016-06-07T00:00:00Z	54	MARIA ORTIZ	0	0	0	0	0	1	No

In [74]:
```
print('number of rows are:', df.shape[0])
print('number of columns are:', df.shape[1])
```

```
number of rows are: 110527
number of columns are: 14
```

In [75]:
```
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
```

After visualy inspecting the data

1. The PatientId and AppointmentID are just number that won't contribute to the analysis
2. Also the ScheduledDay and AppointmentDay are in the object dtype
3. drop null avlues if present
4. drop duplicated values if present
5. Rename the hipertension column to hypertension

Data Cleaning

In [76]:
```
# check for number of null values in each column
df.isnull().sum()
```

Out[76]:
```
PatientId         0
AppointmentID     0
Gender            0
ScheduledDay      0
AppointmentDay    0
Age               0
Neighbourhood     0
Scholarship       0
Hipertension      0
Diabetes          0
Alcoholism        0
Handcap           0
SMS_received      0
No-show           0
dtype: int64
```

They are no null values in each columns above

In [77]:
```
#check  for duplicates
print(f'They are {df.duplicated().sum()} duplicated values')
```

```
They are 0 duplicated values
```

In [78]:
```
#Drops PatientId and AppointmentID columns
df=df.drop(['PatientId', 'AppointmentID'], axis=1)
df.head(2)
```

Out[78]:

	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hipertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
0	F	2016-04-29T18:38:08Z	2016-04-29T00:00:00Z	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	M	2016-04-29T16:08:27Z	2016-04-29T00:00:00Z	56	JARDIM DA PENHA	0	0	0	0	0	0	No

In [79]:
```
#convert both  AppointmentDay and ScheduledDay to date time
df['ScheduledDay']=pd.to_datetime(df['ScheduledDay'])
df['AppointmentDay']=pd.to_datetime(df['AppointmentDay'])
```

In [80]:
```
#renames the column from Hipertension to Hypertension
df.rename(columns={'Hipertension' : 'Hypertension'}, inplace=True)
df.head(2)
```

Out[80]:

	Gender	ScheduledDay	AppointmentDay	Age	Neighbourhood	Scholarship	Hypertension	Diabetes	Alcoholism	Handcap	SMS_received	No-show
0	F	2016-04-29 18:38:08+00:00	2016-04-29 00:00:00+00:00	62	JARDIM DA PENHA	0	1	0	0	0	0	No
1	M	2016-04-29 16:08:27+00:00	2016-04-29 00:00:00+00:00	56	JARDIM DA PENHA	0	0	0	0	0	0	No

Exploratory Data Analysis

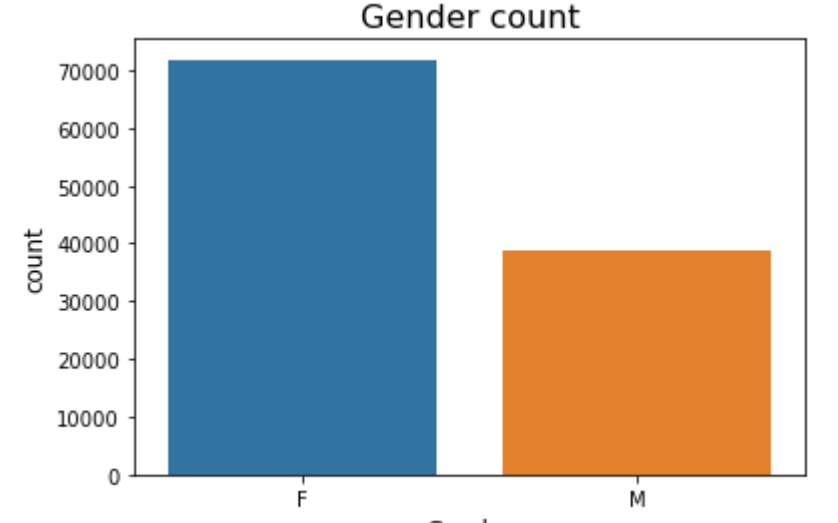
1. Number of male and female in gender column

In [81]:
```
female =df['Gender'].value_counts()[0]
male =df['Gender'].value_counts()[1]
print(f'They are {female} female and {male} male')
```

```
They are 71840 female and 38687 male
```

In [82]:
```
data=df['Gender'].value_counts()
sns.barplot(x=data.index, y=data.values )
plt.title('Gender count', fontsize=16)
plt.ylabel('count', fontsize=12)
plt.xlabel('Gender', fontsize=12)
```

Out[82]:
```
Text(0.5, 0, 'Gender')
```



In [83]:
```
print(f'They are more female to male with in {round(71840/(71840+38687)*100)}% to {round(38687/(71840+38687)*100)}')
```

```
They are more female to male with in 65% to 35
```

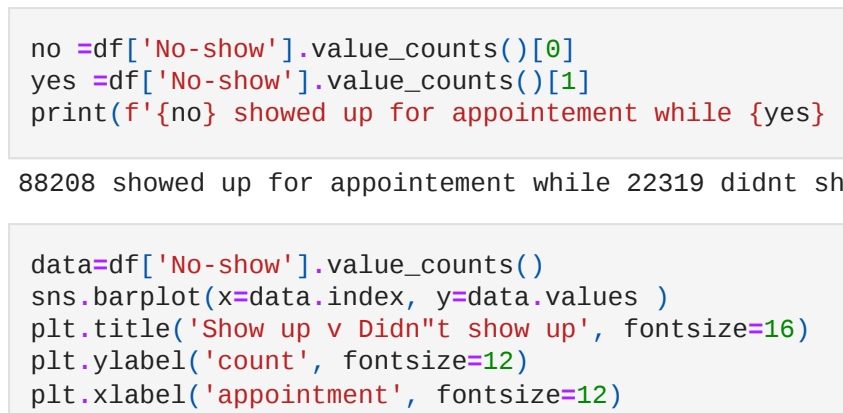
2. What number showed and did not show up for appointment

In [84]:
```
no =df['No-show'].value_counts()[0]
yes =df['No-show'].value_counts()[1]
print(f'{no} showed up for appointement while {yes} didnt show up')
```

```
88208 showed up for appointement while 22319 didnt show up
```

In [85]:
```
data=df['No-show'].value_counts()
sns.barplot(x=data.index, y=data.values )
plt.title('Show up v Didn"t show up', fontsize=16)
plt.ylabel('count', fontsize=12)
plt.xlabel('appointment', fontsize=12)
```

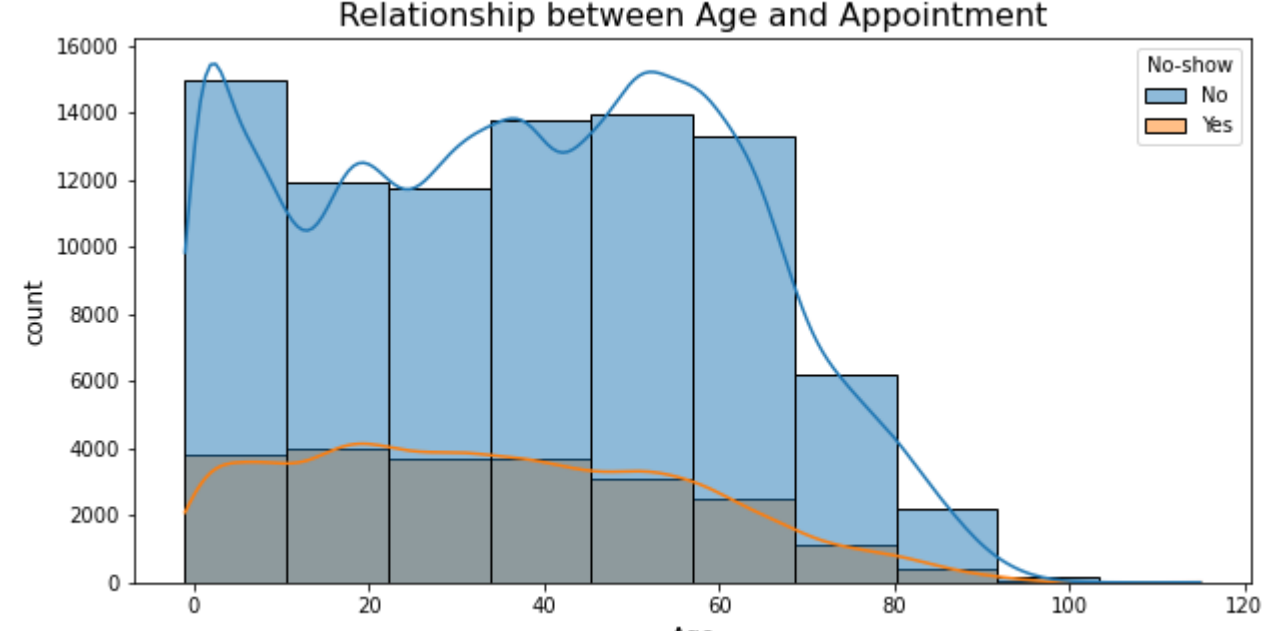
Out[85]:
```
Text(0.5, 0, 'appointment')
```



88208 showed up for appointement while 22319 didnt show up

3. Is there a relationship between age and keeping appointment?

In [86]:
```
plt.figure(figsize=(10,5))
sns.histplot(data=df, x= 'Age', bins=10, kde=True, color ='navy', hue = 'No-show')
plt.title('Relationship between Age and Appointment', fontsize=16)
plt.ylabel('count', fontsize=12)
plt.xlabel('Age', fontsize=12);
```



Those at Younger age tends to default on appointment

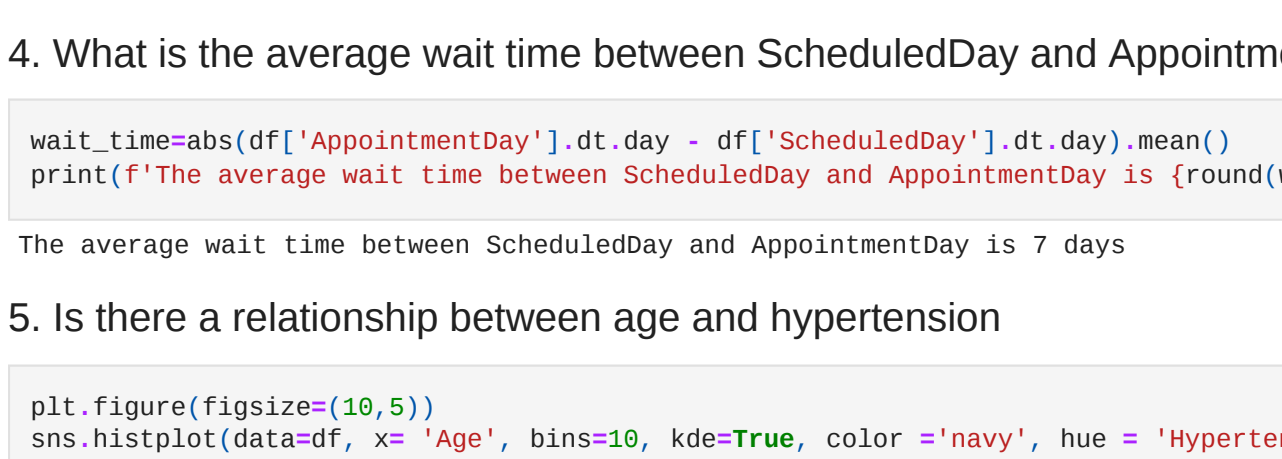
4. What is the average wait time between ScheduledDay and AppointmentDay

In [87]:
```
wait_time=abs(df['AppointmentDay'].dt.day - df['ScheduledDay'].dt.day).mean()
print(f'The average wait time between ScheduledDay and AppointmentDay is {round(wait_time)} days')
```

```
The average wait time between ScheduledDay and AppointmentDay is 7 days
```

5. Is there a relationship between age and hypertension

In [88]:
```
plt.figure(figsize=(10,5))
sns.histplot(data=df, x= 'Age', bins=10, kde=True, color ='navy', hue = 'Hypertension')
plt.title('Relationship between Age and hypertension', fontsize=16)
plt.ylabel('count', fontsize=12)
plt.xlabel('Age', fontsize=12);
```



Hypertension tends to increase from the age of 40 and above

Conclusions

Result

- They are 71840 female and 38687 male
- 88208 showed up for appointement while 22319 didnt show up
- Those at Younger age tends to default on appointment
- The average wait time between ScheduledDay and AppointmentDay is 10 days 00 hours 36 minutes and 34 seconds
- Older ages tends to be more hypertensive

Limitiation

- Data about Level and ownwership of hospital might be need, appointment with private hospital tend to be expensive and well manage. Appointment with a tertiary hostpital tend to deal with serious condition. all this might contribute signifcantlly to keeping appointment.

Refrence: NA

In [90]:
```
!jupyter nbconvert --to html notebook.ipynb
```

```
[NbConvertApp] WARNING | pattern 'notebook.ipynb' matched no files
This application is used to convert notebook files (*.ipynb)
        to various other formats.

        WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.

Options
=======
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
To see all configurable class-options for some <cmd>, use:
    <cmd> --help-all

--debug
    set log level to logging.DEBUG (maximize logging output)
    Equivalent to: [--Application.log_level=10]
--show-config
    Show the application's configuration (human-readable format)
    Equivalent to: [--Application.show_config=True]
--show-config-json
    Show the application's configuration (json format)
    Equivalent to: [--Application.show_config_json=True]
--generate-config
    generate default config file
    Equivalent to: [--JupyterApp.generate_config=True]
-y
    Answer yes to any questions instead of prompting.
    Equivalent to: [--JupyterApp.answer_yes=True]
--execute
    Execute the notebook prior to export.
    Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
    Continue notebook execution even if one of the cells throws an error and include the error message in the cell output (the default behaviour is to abort c
onversion). This flag is only relevant if '--execute' was specified, too.
    Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
    read a single notebook file from stdin. Write the resulting notebook with default basename 'notebook.*'
    Equivalent to: [--NbConvertApp.from_stdin=True]
--stdout
    Write notebook output to stdout instead of files.
    Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
    Run nbconvert in place, overwriting the existing notebook (only
            relevant when converting to notebook format)
    Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_format=notebook --FilesWriter.build_directory=]
--clear-output
    Clear output of current file and save in place,
            overwriting the existing notebook.
    Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_format=notebook --FilesWriter.build_directory= --ClearOutputPreprocessor.enab
led=True]
--no-prompt
    Exclude input and output prompts from converted document.
    Equivalent to: [--TemplateExporter.exclude_input_prompt=True --TemplateExporter.exclude_output_prompt=True]
--no-input
    Exclude input cells and output prompts from converted document.
            This mode is ideal for generating code-free reports.
    Equivalent to: [--TemplateExporter.exclude_output_prompt=True --TemplateExporter.exclude_input=True --TemplateExporter.exclude_input_prompt=True]
--allow-chromium-download
    Whether to allow downloading chromium if no suitable version is found on the system.
    Equivalent to: [--WebPDFExporter.allow_chromium_download=True]
--log-level=<Enum>
    Set the log level by value or name.
    Choices: any of [0, 10, 20, 30, 40, 50, 'DEBUG', 'INFO', 'WARN', 'ERROR', 'CRITICAL']
    Default: 30
    Equivalent to: [--Application.log_level]
--config=<Unicode>
    Full path of a config file.
    Default: ''
    Equivalent to: [--JupyterApp.config_file]
--to=<Unicode>
    The export format to be used, either one of the built-in formats
            ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook', 'pdf', 'python', 'rst', 'script', 'slides', 'webpdf']
            or a dotted object name that represents the import path for an
            ``Exporter`` class
    Default: ''
    Equivalent to: [--NbConvertApp.export_format]
--template=<Unicode>
    Name of the template to use
    Default: ''
    Equivalent to: [--TemplateExporter.template_name]
--template-file=<Unicode>
    Name of the template file to use
    Default: None
    Equivalent to: [--TemplateExporter.template_file]
--writer=<DottedObjectName>
    Writer class used to write the
                                        results of the conversion
    Default: 'FilesWriter'
    Equivalent to: [--NbConvertApp.writer_class]
--post=<DottedOrNone>
    PostProcessor class used to write the
                                        results of the conversion
    Default: ''
    Equivalent to: [--NbConvertApp.postprocessor_class]
--output=<Unicode>
    overwrite base name use for output files.
                can only be used when converting one notebook at a time.
    Default: ''
    Equivalent to: [--NbConvertApp.output_base]
--output-dir=<Unicode>
    Directory to write output(s) to. Defaults
                                  to output to the directory of each notebook. To recover
                                  previous default behaviour (outputting to the current
                                  working directory) use . as the flag value.
    Default: ''
    Equivalent to: [--FilesWriter.build_directory]
--reveal-prefix=<Unicode>
    The URL prefix for reveal.js (version 3.x).
            This defaults to the reveal CDN, but can be any url pointing to a copy
            of reveal.js.
            For speaker notes to work, this must be a relative path to a local
            copy of reveal.js: e.g., "reveal.js".
            If a relative path is given, it must be a subdirectory of the
            current directory (from which the server is run).
            See the usage documentation
            (https://nbconvert.readthedocs.io/en/latest/usage.html#reveal-js-html-slideshow)
            for more details.
    Default: ''
    Equivalent to: [--SlidesExporter.reveal_url_prefix]
--nbformat=<Enum>
    The nbformat version to write.
            Use this to downgrade notebooks.
    Choices: any of [1, 2, 3, 4]
    Default: 4
    Equivalent to: [--NotebookExporter.nbformat_version]

Examples
--------

    The simplest way to use nbconvert is

            > jupyter nbconvert mynotebook.ipynb --to html

            Options include ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook', 'pdf', 'python', 'rst', 'script', 'slides', 'webpdf'].

            > jupyter nbconvert --to latex mynotebook.ipynb

            Both HTML and LaTeX support multiple output templates. LaTeX includes
            'base', 'article' and 'report'.  HTML includes 'basic' and 'full'. You
            can specify the flavor of the format used.

            > jupyter nbconvert --to html --template lab mynotebook.ipynb

            You can also pipe the output to stdout, rather than a file

            > jupyter nbconvert mynotebook.ipynb --stdout

            PDF is generated via latex

            > jupyter nbconvert mynotebook.ipynb --to pdf

            You can get (and serve) a Reveal.js-powered slideshow

            > jupyter nbconvert myslides.ipynb --to slides --post serve

            Multiple notebooks can be given at the command line in a couple of
            different ways:

            > jupyter nbconvert notebook*.ipynb
            > jupyter nbconvert notebook1.ipynb notebook2.ipynb

            or you can specify the notebooks list in a config file, containing::

                c.NbConvertApp.notebooks = ["my_notebook.ipynb"]

            > jupyter nbconvert --config mycfg.py

To see all available configurables, use `--help-all`.
```

In [ ]: