Analyzing US Economic Data and Building a Dashboard

Description

Extracting essential data from a dataset and displaying it is a necessary part of data science; therefore individuals can make correct decisions based on the data. In this assignment, you will extract some essential economic indicators from some data, you will then display these economic indicators in a Dashboard. You can then share the dashboard via an URL.

Gross domestic product (GDP) (https://en.wikipedia.org/wiki/Gross_domestic_product) is a measure of the market value of all the final goods and services produced in a period. GDP is an indicator of how well the economy is doing. A drop in GDP indicates the economy is producing less; similarly an increase in GDP suggests the economy is performing better. In this lab, you will examine how changes in GDP impact the unemployment rate. You will take screen shots of every step, you will share the notebook and the URL pointing to the dashboard.

Table of Contents

- Define a Function that Makes a Dashboard
- Question 1: Create a dataframe that contains the GDP data and display it
- Question 2: Create a dataframe that contains the unemployment data and display it
- Question 3: Display a dataframe where unemployment was greater than 8.5%
- Question 4: Use the function make_dashboard to make a dashboard
- **(Optional not marked)** Save the dashboard on IBM cloud and display it

Estimated Time Needed: **180 min**

Define Function that Makes a Dashboard

We will import the following libraries.

In [17]:

```
import pandas as pd
from bokeh.plotting import figure, output_file, show,output_notebook
output_notebook()
```

(https://bokeh.pydata.org) Bokeh 1.0.4 successfully loaded.

In this section, we define the function `make_dashboard` . You don't have to know how the function works, you should only care about the inputs. The function will produce a dashboard as well as an html file. You can then use this html file to share your dashboard. If you do not know what an html file is don't worry everything you need to know will be provided in the lab.

In [19]:

```
def make_dashboard(x, gdp_change, unemployment, title, file_name):
    output_file(file_name)
    p = figure(title=title, x_axis_label='year', y_axis_label='%')
    p.line(x.squeeze(), gdp_change.squeeze(), color="firebrick", line_width=4, legend="
% GDP change")
    p.line(x.squeeze(), unemployment.squeeze(), line_width=4, legend="% unemployed")
    show(p)
```

The dictionary `links` contain the CSV files with all the data. The value for the key `GDP` is the file that contains the GDP data. The value for the key `unemployment` contains the unemployment data.

In [18]:

```
links={'GDP':'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cogniti
veClass/PY0101EN/projects/coursera_project/clean_gdp.csv',\
       'unemployment':'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-dat
a/CognitiveClass/PY0101EN/projects/coursera_project/clean_unemployment.csv'}
```

Question 1: Create a dataframe that contains the GDP data and display the first five rows of the dataframe.

Use the dictionary `links` and the function `pd.read_csv` to create a Pandas dataframes that contains the GDP data.

Hint: `links["GDP"]` contains the path or name of the file.

In [20]:

```
#Type your code here
df = pd.read_csv(links["GDP"])
df.head()
```

Out[20]:

	date	level-current	level-chained	change-current	change-chained
0	1948	274.8	2020.0	-0.7	-0.6
1	1949	272.8	2008.9	10.0	8.7
2	1950	300.2	2184.0	15.7	8.0
3	1951	347.3	2360.0	5.9	4.1
4	1952	367.7	2456.1	6.0	4.7

Use the method `head()` to display the first five rows of the GDP data, then take a screen-shot.

Question 2: Create a dataframe that contains the unemployment data. Display the first five rows of the dataframe.

Use the dictionary `links` and the function `pd.read_csv` to create a Pandas dataframes that contains the unemployment data.

In [21]:

```
# Type your code here
df_unempl=pd.read_csv(links['unemployment'])
```

Use the method `head()` to display the first five rows of the GDP data, then take a screen-shot.

In [22]:

```
# Type your code here
df_unempl=pd.read_csv(links['unemployment'])
df_unempl.head()
```

Out[22]:

	date	unemployment
0	1948	3.750000
1	1949	6.050000
2	1950	5.208333
3	1951	3.283333
4	1952	3.025000

Question 3: Display a dataframe where unemployment was greater than 8.5%. Take a screen-shot.

In [16]:

```
# Type your code here
df_unempl_85=df_unempl[df_unempl['unemployment']>8.5]
df_unempl_85.head()
```

Out[16]:

	date	unemployment
34	1982	9.708333
35	1983	9.600000
61	2009	9.283333
62	2010	9.608333
63	2011	8.933333

Question 4: Use the function make_dashboard to make a dashboard

In this section, you will call the function `make_dashboard` , to produce a dashboard. We will use the convention of giving each variable the same name as the function parameter.

Create a new dataframe with the column `'date'` called `x` from the dataframe that contains the GDP data.

In [25]:

```
# Create your dataframe with column date
x = df['date']
x.head()
```

Out[25]:

```
0    1948
1    1949
2    1950
3    1951
4    1952
Name: date, dtype: int64
```

Create a new dataframe with the column `'change-current'` called `gdp_change` from the dataframe that contains the GDP data.

In [26]:

```
# Create your dataframe with column change-current
gdp_change = df['change-current']
gdp_change.head()
```

Out[26]:

```
0    -0.7
1    10.0
2    15.7
3     5.9
4     6.0
Name: change-current, dtype: float64
```

Create a new dataframe with the column `'unemployment'` called `unemployment` from the dataframe that contains the unemployment data.

In [29]:

```
# Create your dataframe with column unemployment
unemployment = df_unempl['unemployment']
unemployment.head()
```

Out[29]:

```
0    3.750000
1    6.050000
2    5.208333
3    3.283333
4    3.025000
Name: unemployment, dtype: float64
```

Give your dashboard a string title, and assign it to the variable `title`

In [31]:

```
title = 'Datas of GDP and unemployment ' # Give your dashboard a string title
```

Finally, the function `make_dashboard` will output an `.html` in your direictory, just like a `csv` file. The name of the file is `"index.html"` and it will be stored in the varable `file_name` .

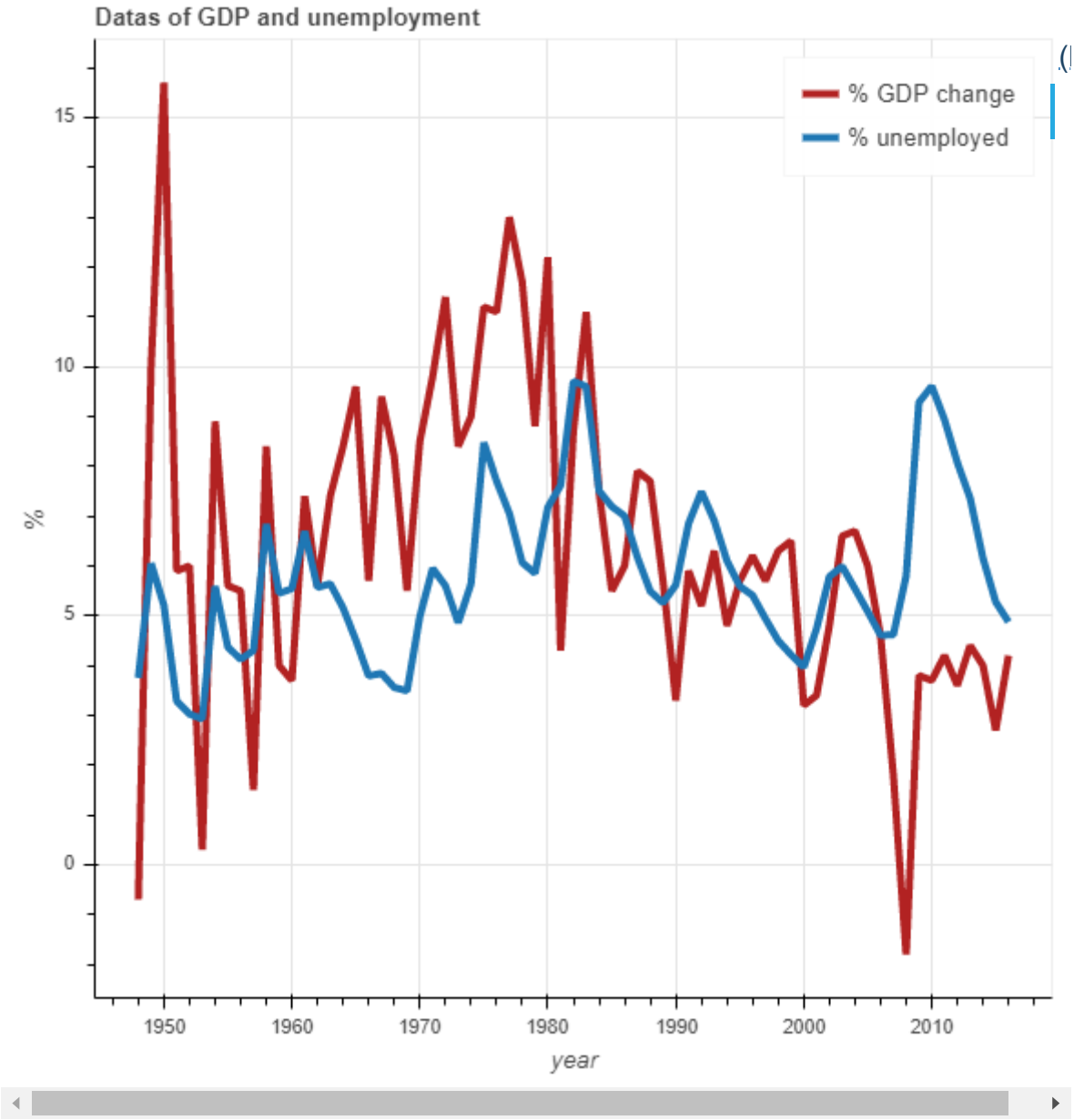
In [33]:

```
file_name = "index.html"
```

Call the function `make_dashboard` , to produce a dashboard. Assign the parameter values accordingly take a the **, take a screen shot of the dashboard and submit it**.

In [34]:

```
# Fill up the parameters in the following function:
 make_dashboard(x=x, gdp_change=gdp_change, unemployment=unemployment, title=title, fil
e_name=file_name)
```



(Optional not marked)Save the dashboard on IBM cloud and display it

From the tutorial *PROVISIONING AN OBJECT STORAGE INSTANCE ON IBM CLOUD* copy the JSON object containing the credentials you created. You'll want to store everything you see in a credentials variable like the one below (obviously, replace the placeholder values with your own). Take special note of your `access_key_id` and `secret_access_key` . **Do not delete `# @hidden_cell` as this will not allow people to see your credentials when you share your notebook.**

```
credentials = {

   "apikey": "your-api-key",

   "cos_hmac_keys": {

    "access_key_id": "your-access-key-here",

     "secret_access_key": "your-secret-access-key-here"

   },


    "endpoints": "your-endpoints",

   "iam_apikey_description": "your-iam_apikey_description",

   "iam_apikey_name": "your-iam_apikey_name",

   "iam_role_crn": "your-iam_apikey_name",

    "iam_serviceid_crn": "your-iam_serviceid_crn",

  "resource_instance_id": "your-resource_instance_id"

}
```

In [35]:

```
# @hidden_cell
#
```

You will need the endpoint make sure the setting are the same as *PROVISIONING AN OBJECT STORAGE INSTANCE ON IBM CLOUD* assign the name of your bucket to the variable `bucket_name`

In [36]:

```
endpoint = 'https://s3-api.us-geo.objectstorage.softlayer.net'
```

From the tutorial *PROVISIONING AN OBJECT STORAGE INSTANCE ON IBM CLOUD* assign the name of your bucket to the variable `bucket_name`

In [44]:

```
bucket_name = 'medha-python-dash' # Type your bucket name on IBM Cloud
```

We can access IBM Cloud Object Storage with Python useing the `boto3` library, which we'll import below:

In [38]:

```
import boto3
```

We can interact with IBM Cloud Object Storage through a `boto3` resource object.

In [46]:

```
resource = boto3.resource(
    's3',
    aws_access_key_id = credentials["cos_hmac_keys"]['03f2fba69dc04fd891e6429eb61b7157'
],
    aws_secret_access_key = credentials["cos_hmac_keys"]["34f735d61762313c3da43dab74d22
7efd1989cecece36bed"],
    endpoint_url ="https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints",
)
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
<ipython-input-46-4079e3768d63> in <module>
      1 resource = boto3.resource(
      2     's3',
----> 3     aws_access_key_id = credentials["cos_hmac_keys"]['03f2fba69dc0
4fd891e6429eb61b7157'],
      4     aws_secret_access_key = credentials["cos_hmac_keys"]["34f735d6
1762313c3da43dab74d227efd1989cecece36bed"],
      5     endpoint_url ="https://control.cloud-object-storage.cloud.ibm.
com/v2/endpoints",

NameError: name 'credentials' is not defined
```

We are going to use `open` to create a file object. To get the path of the file, you are going to concatenate the name of the file stored in the variable `file_name`. The directory stored in the variable directory using the `+` operator and assign it to the variable `html_path`. We will use the function `getcwd()` to find current the working directory.

In [40]:

```
import os

directory = os.getcwd()
html_path = directory + "/" + file_name
```

Now you must read the html file, use the function `f = open(html_path, mode)` to create a file object and assign it to the variable `f`. The parameter `file` should be the variable `html_path`, the mode should be `"r"` for read.

In [41]:

```
# Type your code here
f = open(html_path,'r')
```

To load your dataset into the bucket we will use the method `put_object` , you must set the parameter name to the name of the bucket, the parameter `Key` should be the name of the HTML file and the value for the parameter Body should be set to `f.read()` .

In [42]:

```
# Fill up the parameters in the following function:
resource.Bucket(name=bucket_name).put_object(Key=file_name, Body=f.read())
```

```
---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
<ipython-input-42-db59f2f5e2d6> in <module>
      1 # Fill up the parameters in the following function:
----> 2 resource.Bucket(name=bucket_name).put_object(Key=file_name, Body=f
.read())

NameError: name 'resource' is not defined
```

In the dictionary `Params` provide the bucket name as the value for the key *'Bucket'*. Also for the value of the key *'Key'* add the name of the `html` file, both values should be strings.

In [ ]:

```
# Fill in the value for each key
# Params = {'Bucket': ,'Key': }
```

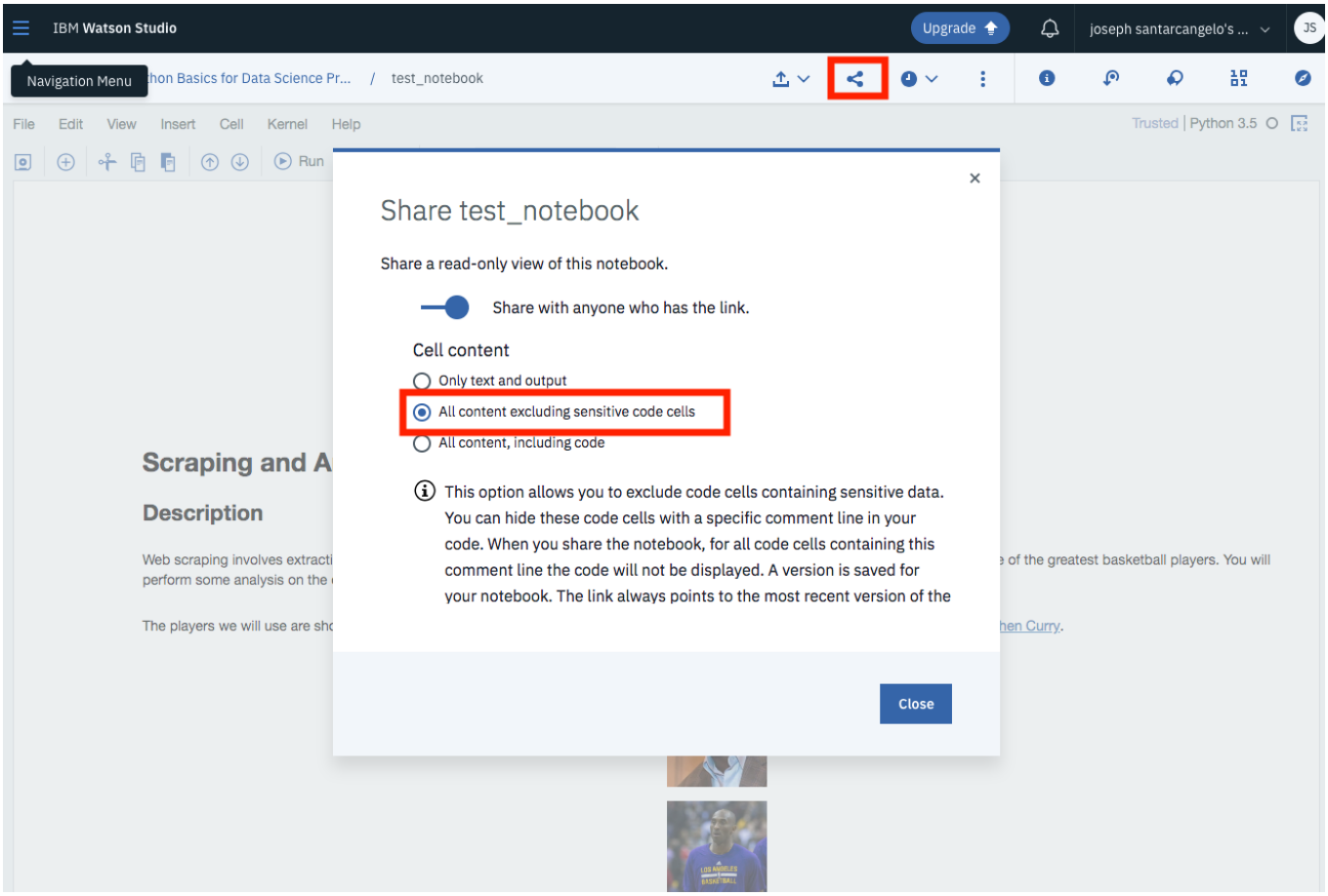
The following lines of code will generate a URL to share your dashboard. The URL only last seven days, but don't worry you will get full marks if the URL is visible in your notebook.

In [ ]:

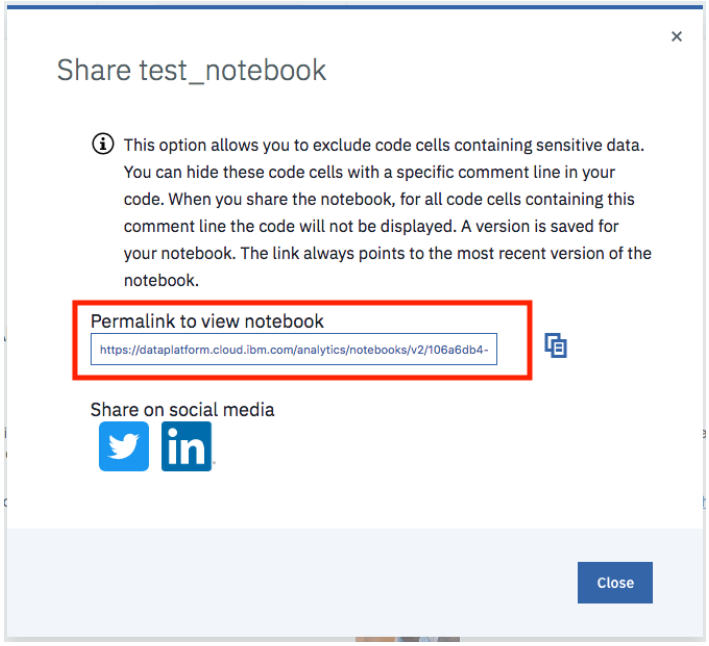
```
import sys
time = 7*24*60**2
client = boto3.client(
    's3',
    aws_access_key_id = credentials["cos_hmac_keys"]['access_key_id'],
    aws_secret_access_key = credentials["cos_hmac_keys"]["secret_access_key"],
    endpoint_url=endpoint,

)
url = client.generate_presigned_url('get_object',Params=Params,ExpiresIn=time)
print(url)
```

How to submit

Once you complete your notebook you will have to share it to be marked. Select the icon on the top right a marked in red in the image below, a dialogue box should open, select the option all content excluding sensitive code cells.



You can then share the notebook via a URL by scrolling down as shown in the following image:



About the Authors:

Joseph Santarcangelo (https://www.linkedin.com/in/joseph-s-50398b136/) has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Other contributors: Yi leng Yao (https://www.linkedin.com/in/yi-leng-yao-84451275/), Mavis Zhou (www.linkedin.com/in/jiahui-mavis-zhou-a4537814a)

References :

1) Economic Research at the St. Louis Fed (https://research.stlouisfed.org/): Civilian Unemployment Rate (https://fred.stlouisfed.org/series/UNRATE/)

2) Data Packaged Core Datasets (https://github.com/datasets)

