Document highlight:

- Data location
- Tool Installation
  - Python
  - Jupyter Notebook
- Open a new notebook, import the libraries to be used for the project
- Activities
  - Pulling data and into a dataframe
  - Data Cleanup
  - Visualization

1. Data location

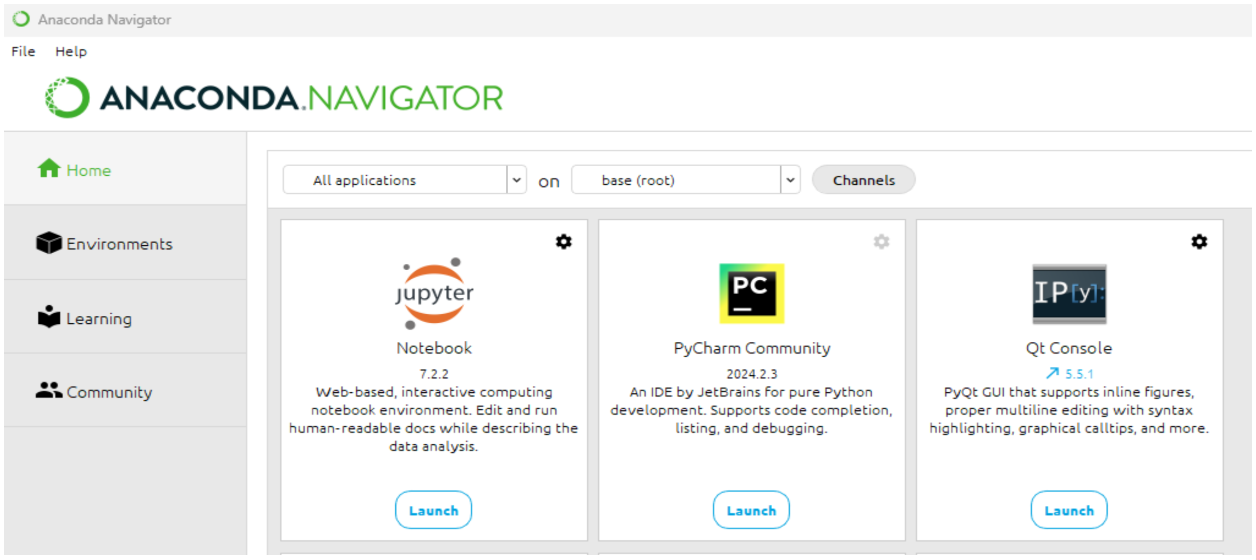
URL:
https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest

2. Tool Installation

Download and install (follow installation guide):

Python
https://www.python.org/downloads/

Jupyter Notebook
https://www.anaconda.com/download/success



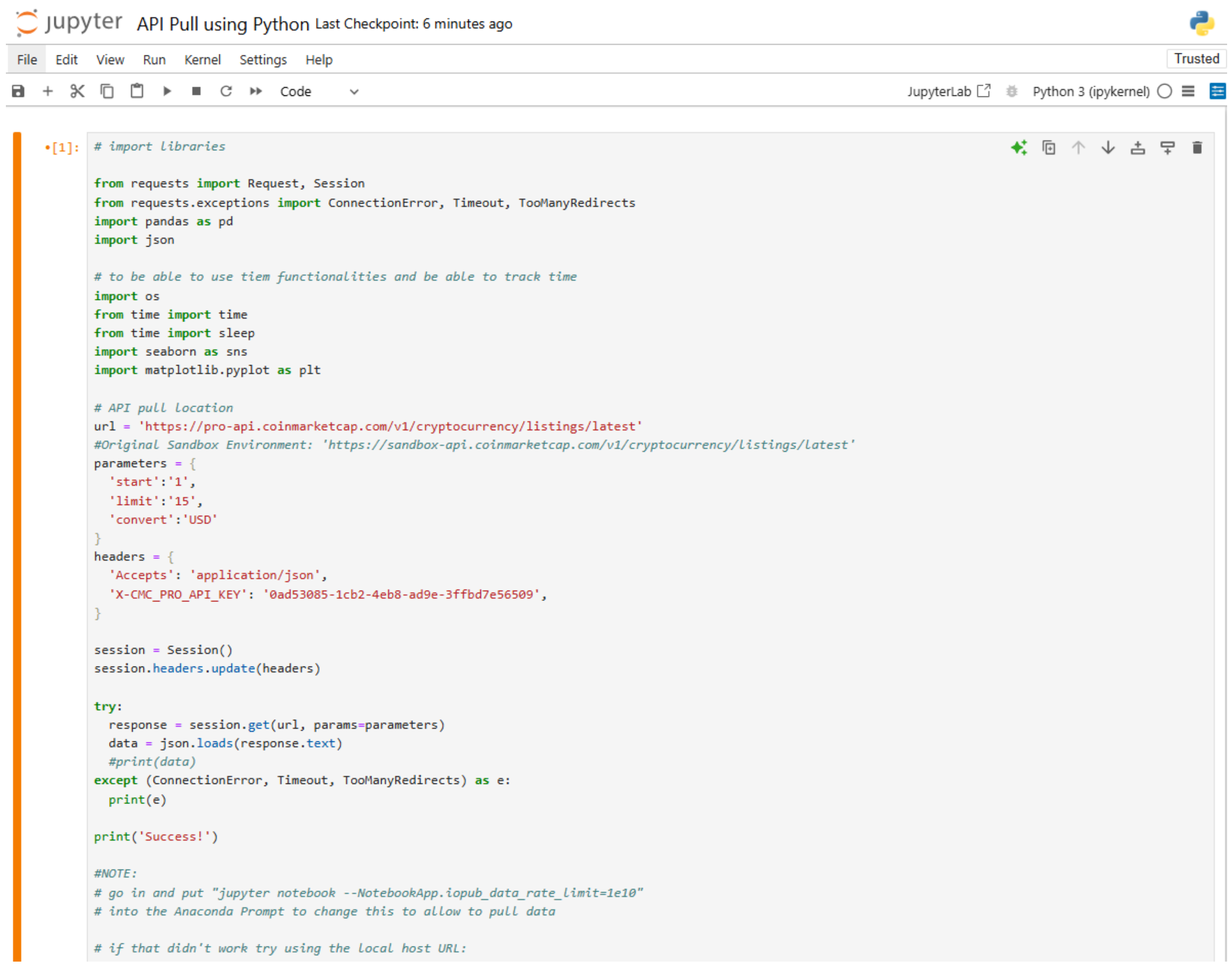
jupyter Notebook 7.2.2
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

PyCharm Community 2024.2.3
An IDE by JetBrains for pure Python development. Supports code completion, listing, and debugging.

Qt Console 5.5.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

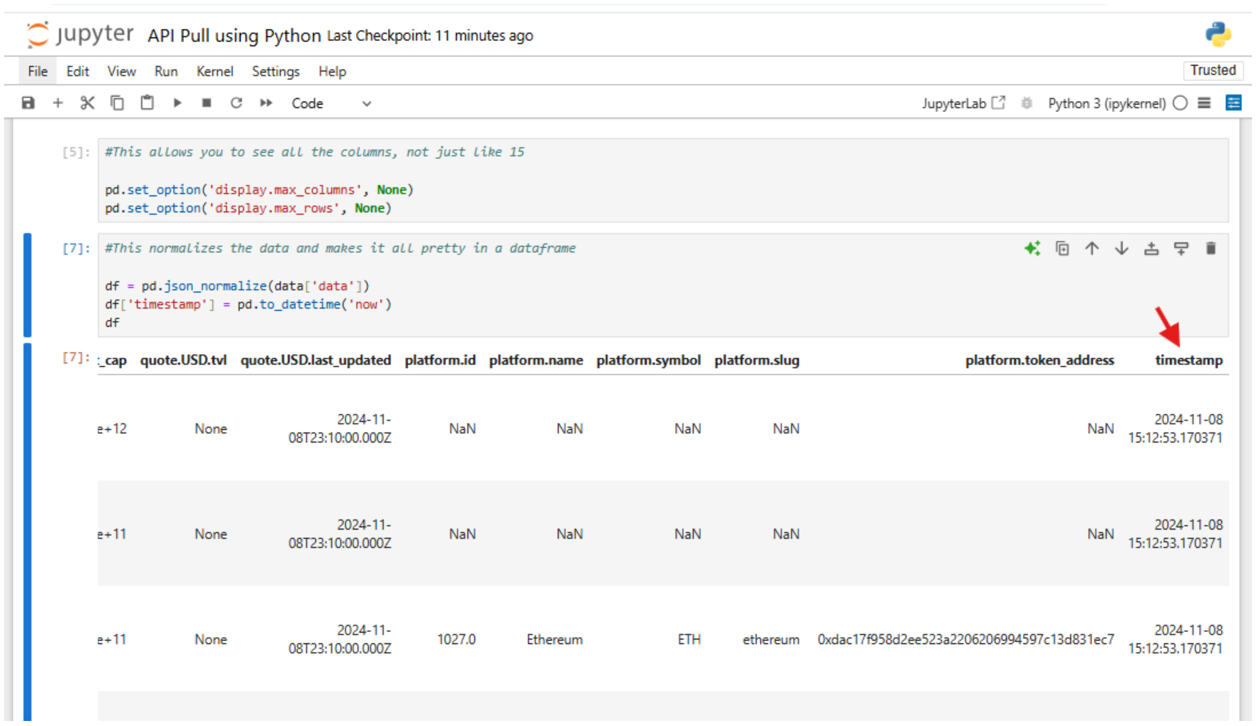
3. Open a new notebook, import the libraries to be used for the project

Import the libraries and other one-time configuration setup


jupyter
API Pull using Python Last Checkpoint: 6 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)
•[1]:

```
# import libraries

from requests import Request, Session
from requests.exceptions import ConnectionError, Timeout, TooManyRedirects
import pandas as pd
import json

# to be able to use tiem functionalities and be able to track time
import os
from time import time
from time import sleep
import seaborn as sns
import matplotlib.pyplot as plt

# API pull location
url = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest'
#Original Sandbox Environment: 'https://sandbox-api.coinmarketcap.com/v1/cryptocurrency/listings/latest'
parameters = {
  'start':'1',
  'limit':'15',
  'convert':'USD'
}
headers = {
  'Accepts': 'application/json',
  'X-CMC_PRO_API_KEY': '0ad53085-1cb2-4eb8-ad9e-3ffbd7e56509',
}

session = Session()
session.headers.update(headers)

try:
  response = session.get(url, params=parameters)
  data = json.loads(response.text)
  #print(data)
except (ConnectionError, Timeout, TooManyRedirects) as e:
  print(e)

print('Success!')

#NOTE:
# go in and put "jupyter notebook --NotebookApp.iopub_data_rate_limit=1e10"
# into the Anaconda Prompt to change this to allow to pull data

# if that didn't work try using the local host URL:
```

4. Activities - Pulling data and into a dataframe, Data Cleanup, and Visualization

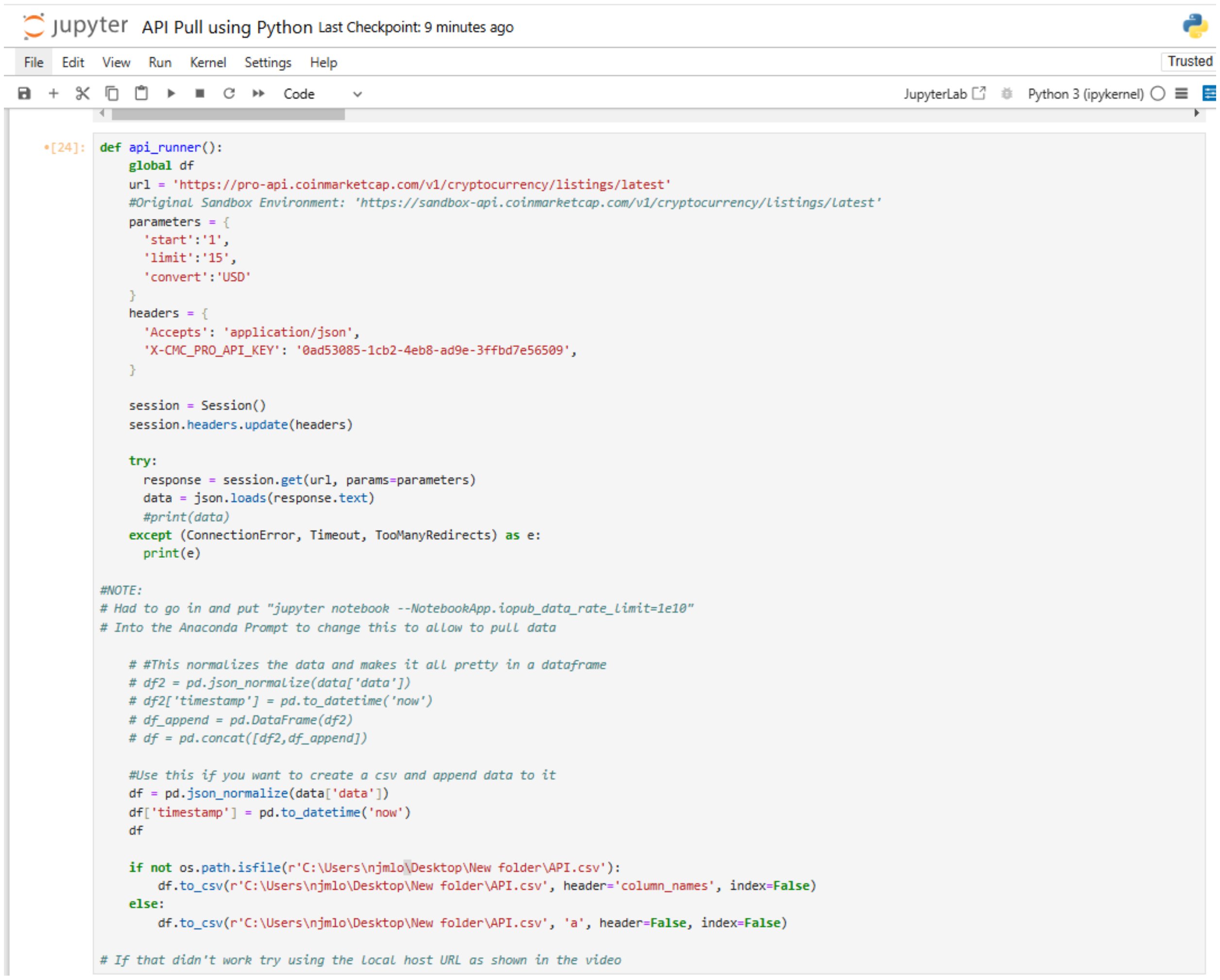

jupyter API Pull using Python Last Checkpoint: 11 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[5]: #This allows you to see all the columns, not just like 15

     pd.set_option('display.max_columns', None)
     pd.set_option('display.max_rows', None)
```

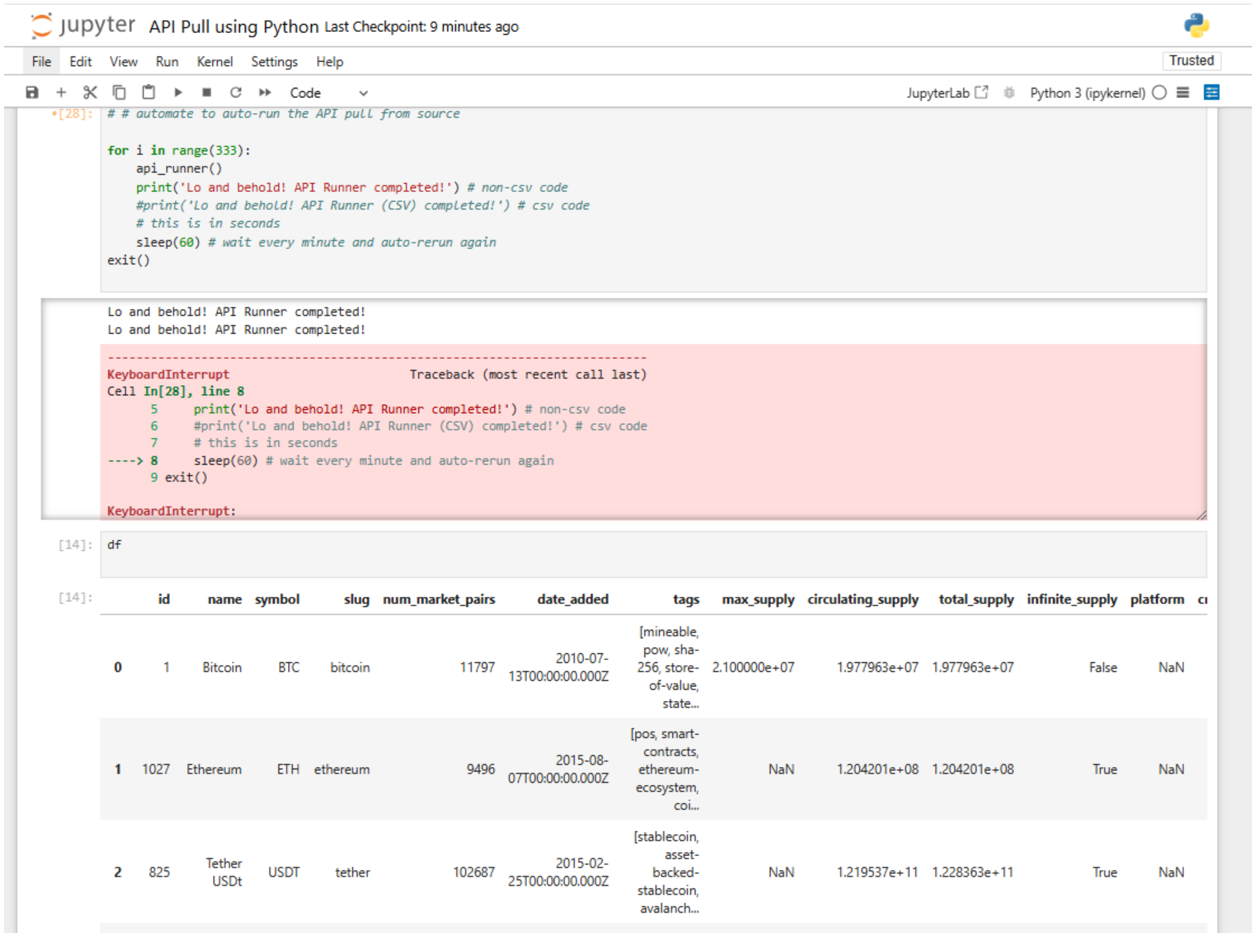
```
[7]: #This normalizes the data and makes it all pretty in a dataframe

     df = pd.json_normalize(data['data'])
     df['timestamp'] = pd.to_datetime('now')
     df
```

[7]:

_cap	quote.USD.tvl	quote.USD.last_updated	platform.id	platform.name	platform.symbol	platform.slug	platform.token_address	timestamp
e+12	None	2024-11-08T23:10:00.000Z	NaN	NaN	NaN	NaN	NaN	2024-11-08 15:12:53.170371
e+11	None	2024-11-08T23:10:00.000Z	NaN	NaN	NaN	NaN	NaN	2024-11-08 15:12:53.170371
e+11	None	2024-11-08T23:10:00.000Z	1027.0	Ethereum	ETH	ethereum	0xdac17f958d2ee523a2206206994597c13d831ec7	2024-11-08 15:12:53.170371


jupyter
API Pull using Python Last Checkpoint: 9 minutes ago
Trusted
File Edit View Run Kernel Settings Help
Code
JupyterLab
Python 3 (ipykernel)
•[24]:

```
def api_runner():
    global df
    url = 'https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest'
    #Original Sandbox Environment: 'https://sandbox-api.coinmarketcap.com/v1/cryptocurrency/listings/latest'
    parameters = {
      'start':'1',
      'limit':'15',
      'convert':'USD'
    }
    headers = {
      'Accepts': 'application/json',
      'X-CMC_PRO_API_KEY': '0ad53085-1cb2-4eb8-ad9e-3ffbd7e56509',
    }

    session = Session()
    session.headers.update(headers)

    try:
      response = session.get(url, params=parameters)
      data = json.loads(response.text)
      #print(data)
    except (ConnectionError, Timeout, TooManyRedirects) as e:
      print(e)

#NOTE:
# Had to go in and put "jupyter notebook --NotebookApp.iopub_data_rate_limit=1e10"
# Into the Anaconda Prompt to change this to allow to pull data

    # #This normalizes the data and makes it all pretty in a dataframe
    # df2 = pd.json_normalize(data['data'])
    # df2['timestamp'] = pd.to_datetime('now')
    # df_append = pd.DataFrame(df2)
    # df = pd.concat([df2,df_append])

    #Use this if you want to create a csv and append data to it
    df = pd.json_normalize(data['data'])
    df['timestamp'] = pd.to_datetime('now')
    df

    if not os.path.isfile(r'C:\Users\njmlo\Desktop\New folder\API.csv'):
        df.to_csv(r'C:\Users\njmlo\Desktop\New folder\API.csv', header='column_names', index=False)
    else:
        df.to_csv(r'C:\Users\njmlo\Desktop\New folder\API.csv', 'a', header=False, index=False)

# If that didn't work try using the local host URL as shown in the video
```


jupyter API Pull using Python Last Checkpoint: 9 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

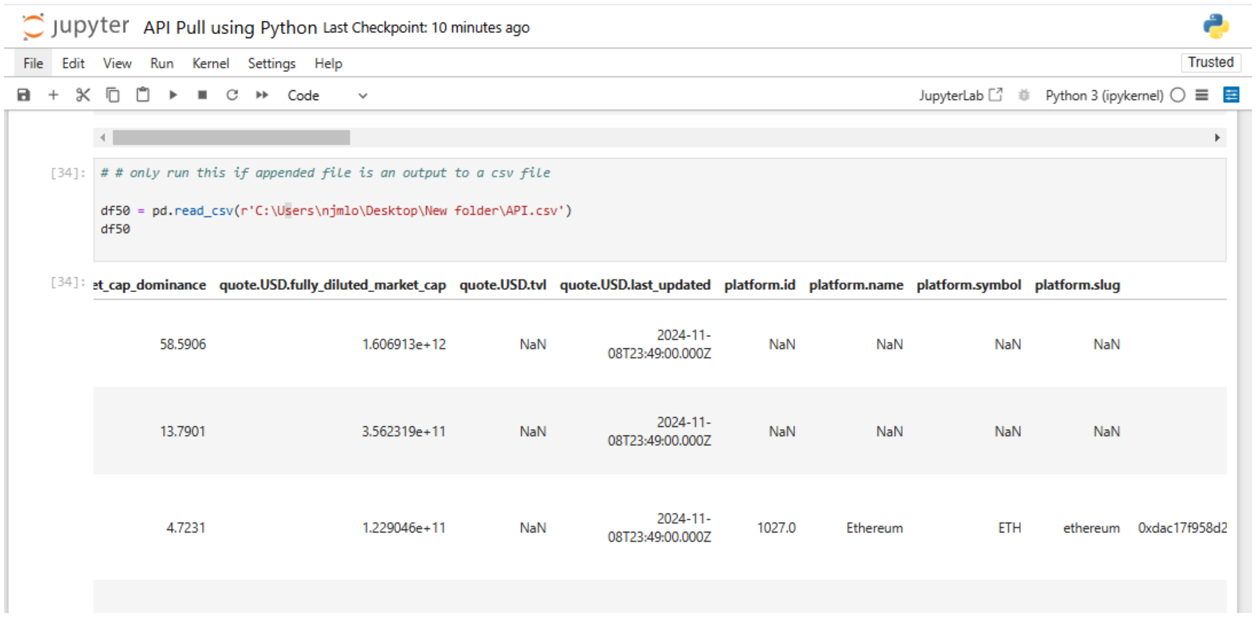
```
•[28]: # # automate to auto-run the API pull from source

       for i in range(333):
           api_runner()
           print('Lo and behold! API Runner completed!') # non-csv code
           #print('Lo and behold! API Runner (CSV) completed!') # csv code
           # this is in seconds
           sleep(60) # wait every minute and auto-rerun again
       exit()
```

```
Lo and behold! API Runner completed!
Lo and behold! API Runner completed!

---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[28], line 8
      5     print('Lo and behold! API Runner completed!') # non-csv code
      6     #print('Lo and behold! API Runner (CSV) completed!') # csv code
      7     # this is in seconds
----> 8     sleep(60) # wait every minute and auto-rerun again
      9 exit()

KeyboardInterrupt:
```

```
[14]: df
```

[14]:

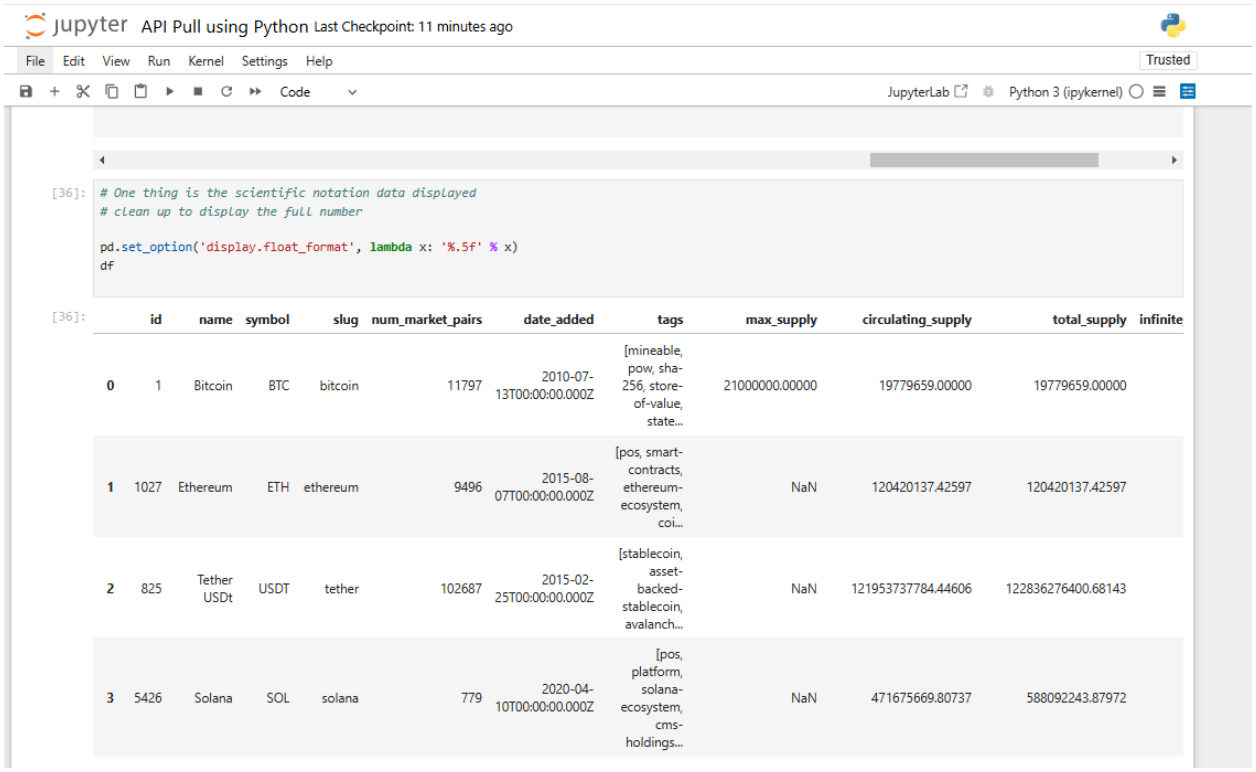
	id	name	symbol	slug	num_market_pairs	date_added	tags	max_supply	circulating_supply	total_supply	infinite_supply	platform	ci
0	1	Bitcoin	BTC	bitcoin	11797	2010-07-13T00:00:00.000Z	[mineable, pow, sha-256, store-of-value, state...	2.100000e+07	1.977963e+07	1.977963e+07	False	NaN	
1	1027	Ethereum	ETH	ethereum	9496	2015-08-07T00:00:00.000Z	[pos, smart-contracts, ethereum-ecosystem, coi...	NaN	1.204201e+08	1.204201e+08	True	NaN	
2	825	Tether USDt	USDT	tether	102687	2015-02-25T00:00:00.000Z	[stablecoin, asset-backed-stablecoin, avalanch...	NaN	1.219537e+11	1.228363e+11	True	NaN	


jupyter API Pull using Python Last Checkpoint: 10 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[34]: # # only run this if appended file is an output to a csv file

      df50 = pd.read_csv(r'C:\Users\njmlo\Desktop\New folder\API.csv')
      df50
```

[34]:

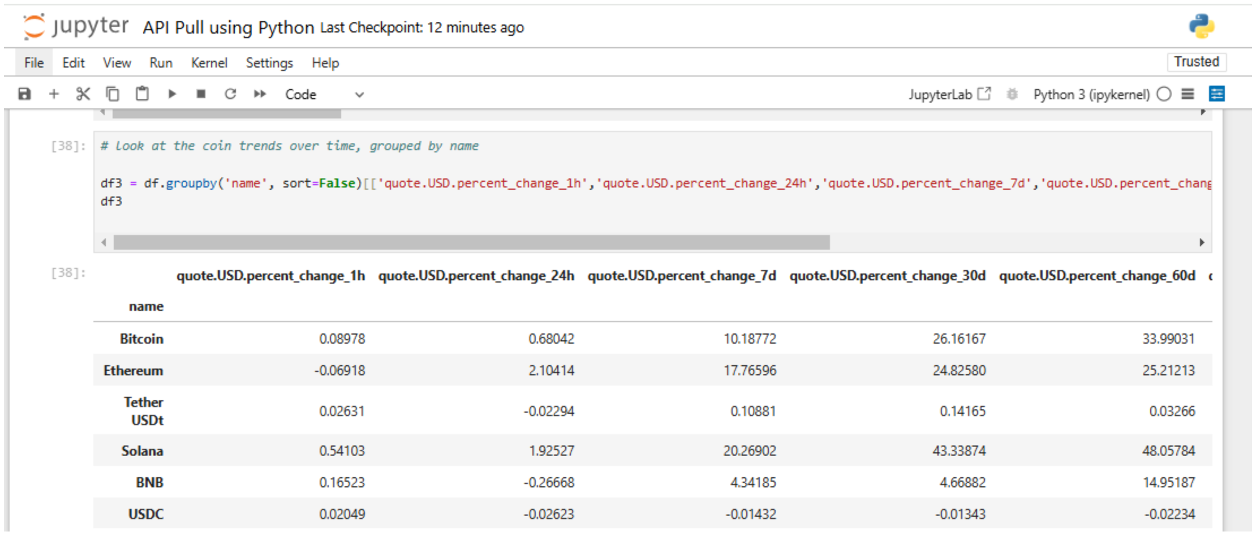
et_cap_dominance	quote.USD.fully_diluted_market_cap	quote.USD.tvl	quote.USD.last_updated	platform.id	platform.name	platform.symbol	platform.slug	
58.5906	1.606913e+12	NaN	2024-11-08T23:49:00.000Z	NaN	NaN	NaN	NaN	
13.7901	3.562319e+11	NaN	2024-11-08T23:49:00.000Z	NaN	NaN	NaN	NaN	
4.7231	1.229046e+11	NaN	2024-11-08T23:49:00.000Z	1027.0	Ethereum	ETH	ethereum	0xdac17f958d2


jupyter
API Pull using Python Last Checkpoint: 11 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[36]: # One thing is the scientific notation data displayed
      # clean up to display the full number

      pd.set_option('display.float_format', lambda x: '%.5f' % x)
      df
```

[36]:

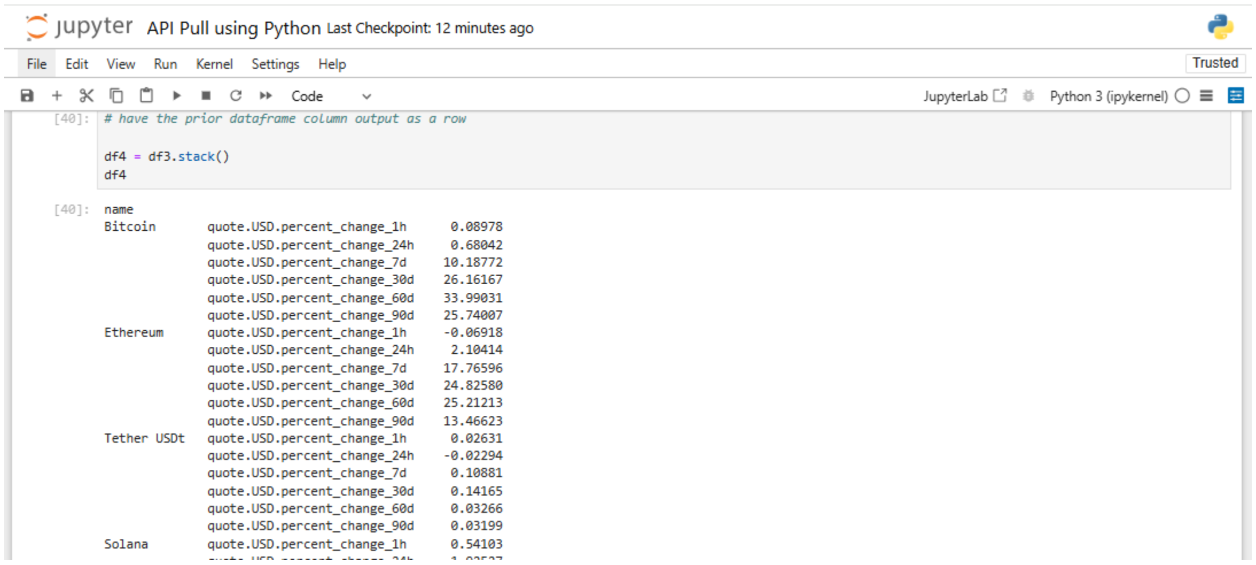
	id	name	symbol	slug	num_market_pairs	date_added	tags	max_supply	circulating_supply	total_supply	infinite
0	1	Bitcoin	BTC	bitcoin	11797	2010-07-13T00:00:00.000Z	[mineable, pow, sha-256, store-of-value, state...	21000000.00000	19779659.00000	19779659.00000	
1	1027	Ethereum	ETH	ethereum	9496	2015-08-07T00:00:00.000Z	[pos, smart-contracts, ethereum-ecosystem, coi...	NaN	120420137.42597	120420137.42597	
2	825	Tether USDt	USDT	tether	102687	2015-02-25T00:00:00.000Z	[stablecoin, asset-backed-stablecoin, avalanch...	NaN	121953737784.44606	122836276400.68143	
3	5426	Solana	SOL	solana	779	2020-04-10T00:00:00.000Z	[pos, platform, solana-ecosystem, cms-holdings...	NaN	471675669.80737	588092243.87972	


jupyter
API Pull using Python Last Checkpoint: 12 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[38]: # look at the coin trends over time, grouped by name

      df3 = df.groupby('name', sort=False)[['quote.USD.percent_change_1h','quote.USD.percent_change_24h','quote.USD.percent_change_7d','quote.USD.percent_chang
      df3
```

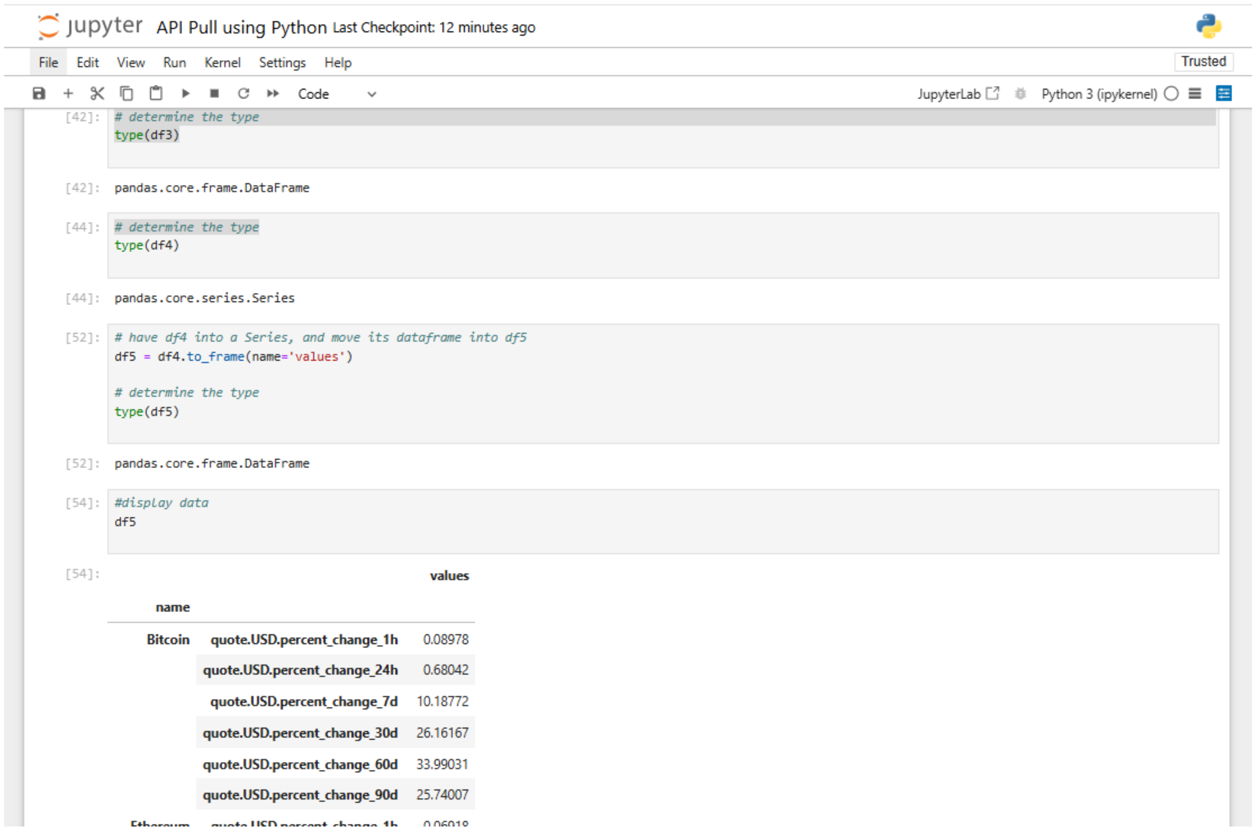
[38]:

name	quote.USD.percent_change_1h	quote.USD.percent_change_24h	quote.USD.percent_change_7d	quote.USD.percent_change_30d	quote.USD.percent_change_60d
Bitcoin	0.08978	0.68042	10.18772	26.16167	33.99031
Ethereum	-0.06918	2.10414	17.76596	24.82580	25.21213
Tether USDt	0.02631	-0.02294	0.10881	0.14165	0.03266
Solana	0.54103	1.92527	20.26902	43.33874	48.05784
BNB	0.16523	-0.26668	4.34185	4.66882	14.95187
USDC	0.02049	-0.02623	-0.01432	-0.01343	-0.02234


jupyter API Pull using Python Last Checkpoint: 12 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[40]: # have the prior dataframe column output as a row

      df4 = df3.stack()
      df4
```

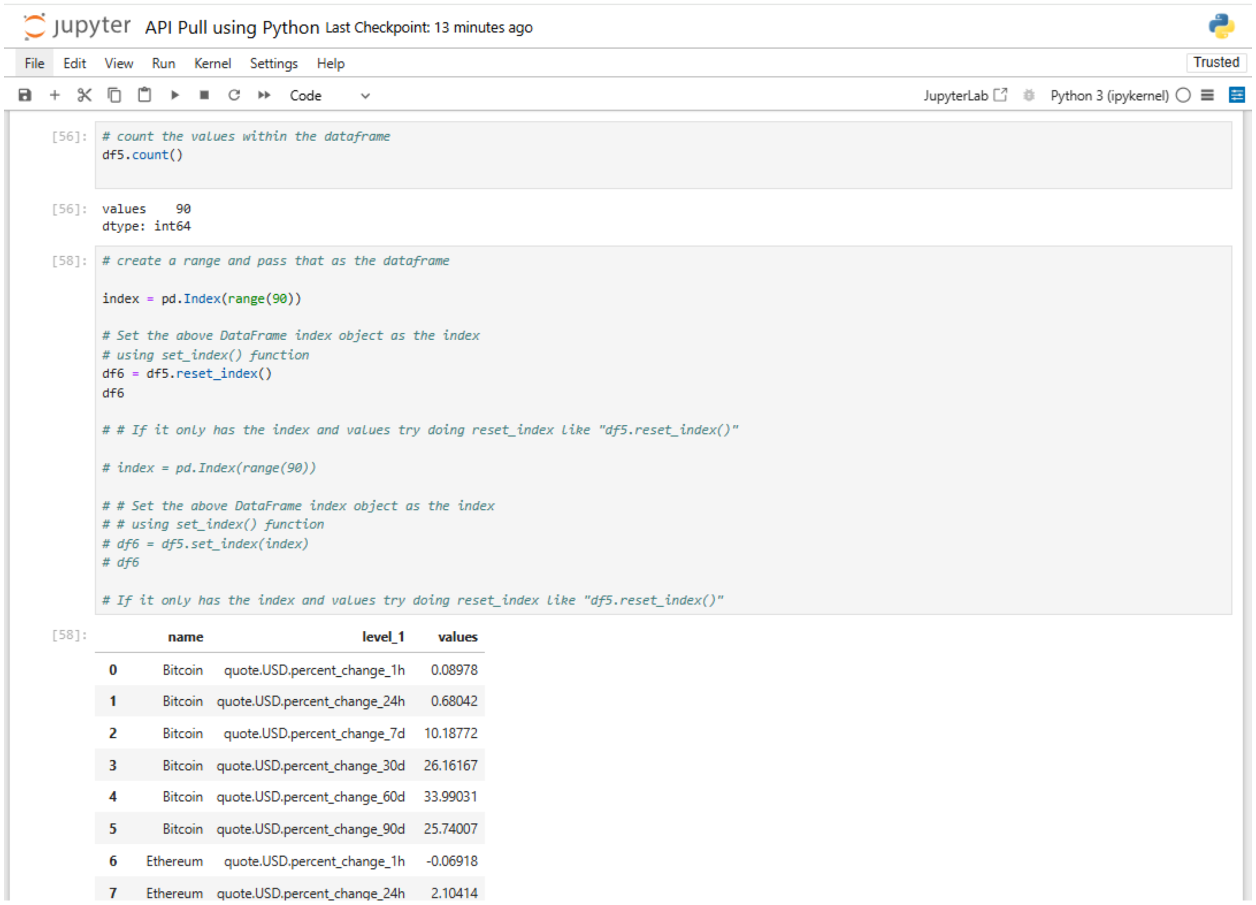
```
[40]: name
      Bitcoin       quote.USD.percent_change_1h      0.08978
                    quote.USD.percent_change_24h     0.68042
                    quote.USD.percent_change_7d     10.18772
                    quote.USD.percent_change_30d    26.16167
                    quote.USD.percent_change_60d    33.99031
                    quote.USD.percent_change_90d    25.74007
      Ethereum      quote.USD.percent_change_1h     -0.06918
                    quote.USD.percent_change_24h     2.10414
                    quote.USD.percent_change_7d     17.76596
                    quote.USD.percent_change_30d    24.82580
                    quote.USD.percent_change_60d    25.21213
                    quote.USD.percent_change_90d    13.46623
      Tether USDt   quote.USD.percent_change_1h      0.02631
                    quote.USD.percent_change_24h    -0.02294
                    quote.USD.percent_change_7d      0.10881
                    quote.USD.percent_change_30d     0.14165
                    quote.USD.percent_change_60d     0.03266
                    quote.USD.percent_change_90d     0.03199
      Solana        quote.USD.percent_change_1h      0.54103
```


jupyter API Pull using Python Last Checkpoint: 12 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[42]: # determine the type
      type(df3)

[42]: pandas.core.frame.DataFrame

[44]: # determine the type
      type(df4)

[44]: pandas.core.series.Series

[52]: # have df4 into a Series, and move its dataframe into df5
      df5 = df4.to_frame(name='values')

      # determine the type
      type(df5)

[52]: pandas.core.frame.DataFrame

[54]: #display data
      df5
```

[54]:

		values
name		
Bitcoin	**quote.USD.percent_change_1h**	0.08978
	quote.USD.percent_change_24h	0.68042
	quote.USD.percent_change_7d	10.18772
	quote.USD.percent_change_30d	26.16167
	quote.USD.percent_change_60d	33.99031
	quote.USD.percent_change_90d	25.74007


jupyter
API Pull using Python Last Checkpoint: 13 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

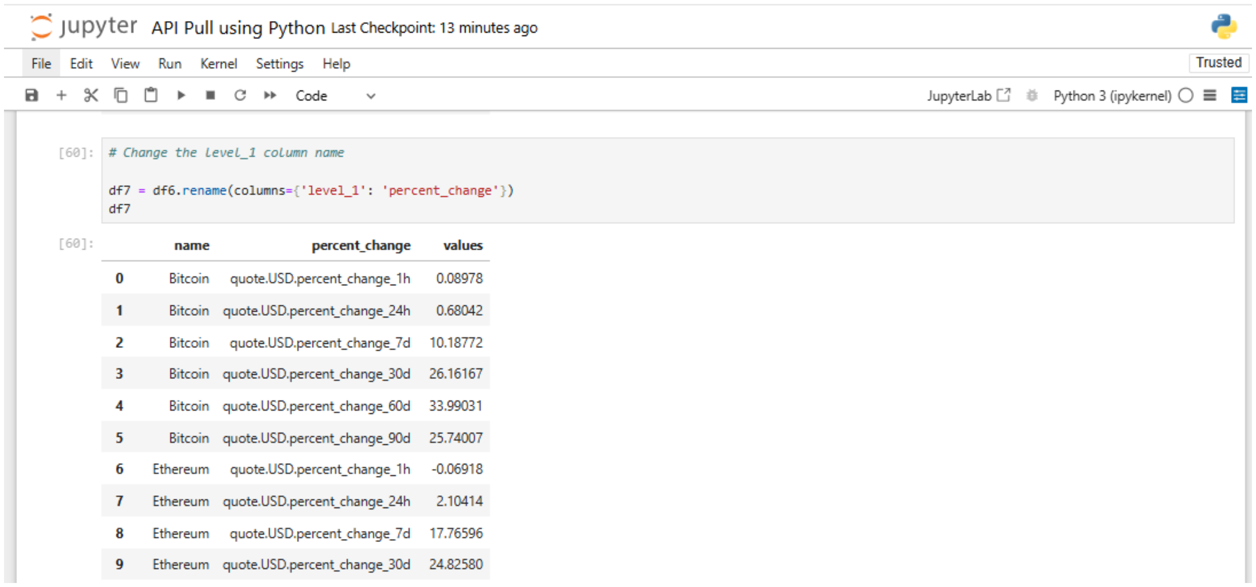
[56]:
```
# count the values within the dataframe
df5.count()
```

[56]:
```
values    90
dtype: int64
```

[58]:
```
# create a range and pass that as the dataframe

index = pd.Index(range(90))

# Set the above DataFrame index object as the index
# using set_index() function
df6 = df5.reset_index()
df6

# # If it only has the index and values try doing reset_index like "df5.reset_index()"

# index = pd.Index(range(90))

# # Set the above DataFrame index object as the index
# # using set_index() function
# df6 = df5.set_index(index)
# df6

# If it only has the index and values try doing reset_index like "df5.reset_index()"
```

[58]:

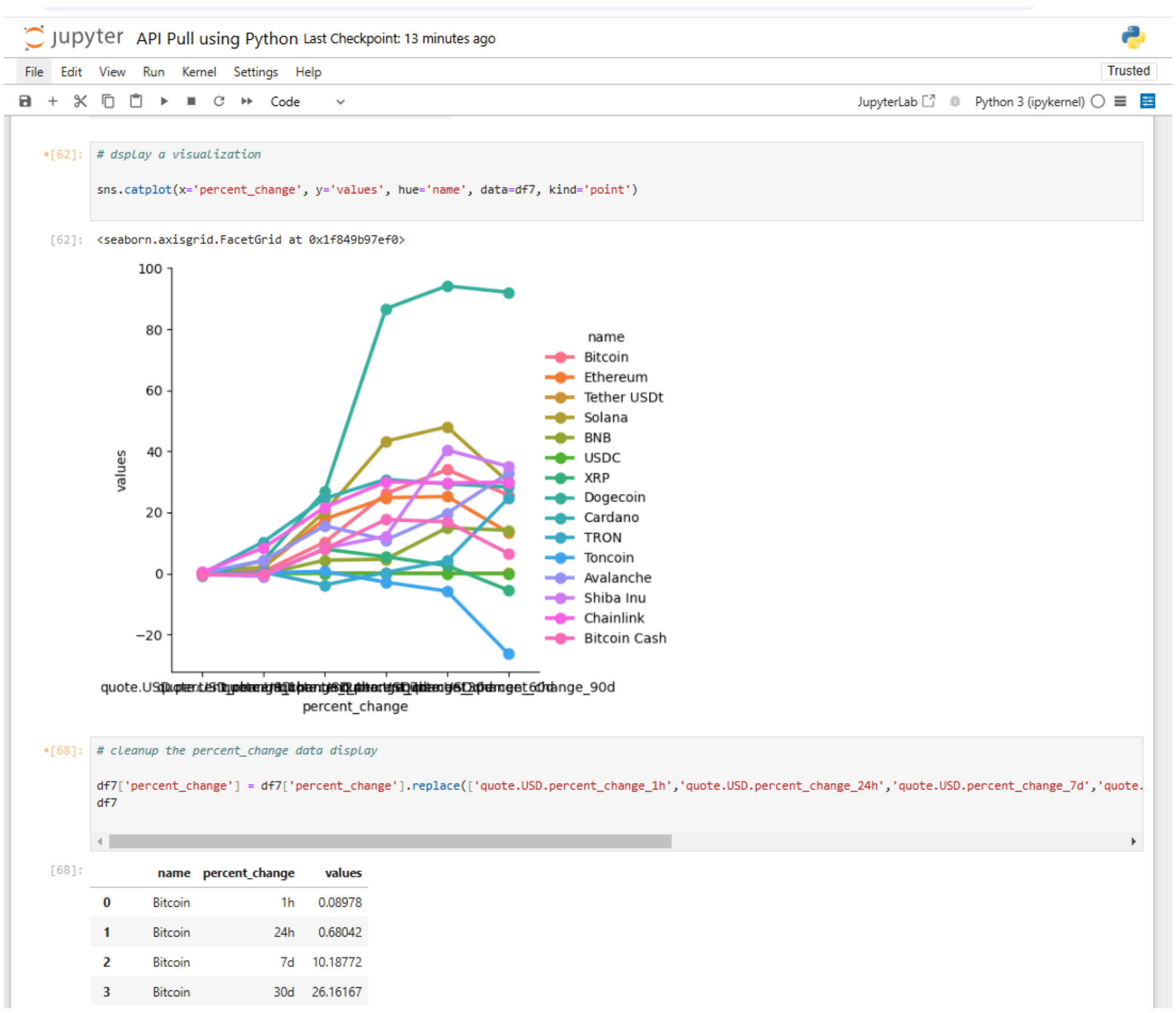
	name	level_1	values
0	Bitcoin	quote.USD.percent_change_1h	0.08978
1	Bitcoin	quote.USD.percent_change_24h	0.68042
2	Bitcoin	quote.USD.percent_change_7d	10.18772
3	Bitcoin	quote.USD.percent_change_30d	26.16167
4	Bitcoin	quote.USD.percent_change_60d	33.99031
5	Bitcoin	quote.USD.percent_change_90d	25.74007
6	Ethereum	quote.USD.percent_change_1h	-0.06918
7	Ethereum	quote.USD.percent_change_24h	2.10414


jupyter
API Pull using Python Last Checkpoint: 13 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

[60]:
```
# Change the level_1 column name

df7 = df6.rename(columns={'level_1': 'percent_change'})
df7
```

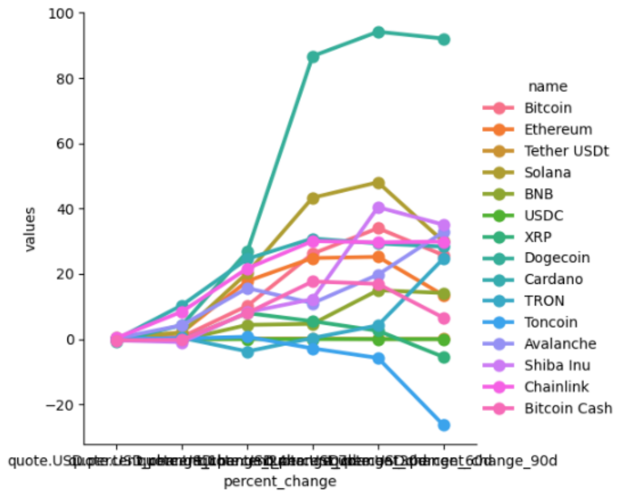
[60]:

	name	percent_change	values
0	Bitcoin	quote.USD.percent_change_1h	0.08978
1	Bitcoin	quote.USD.percent_change_24h	0.68042
2	Bitcoin	quote.USD.percent_change_7d	10.18772
3	Bitcoin	quote.USD.percent_change_30d	26.16167
4	Bitcoin	quote.USD.percent_change_60d	33.99031
5	Bitcoin	quote.USD.percent_change_90d	25.74007
6	Ethereum	quote.USD.percent_change_1h	-0.06918
7	Ethereum	quote.USD.percent_change_24h	2.10414
8	Ethereum	quote.USD.percent_change_7d	17.76596
9	Ethereum	quote.USD.percent_change_30d	24.82580



jupyter
API Pull using Python Last Checkpoint: 13 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

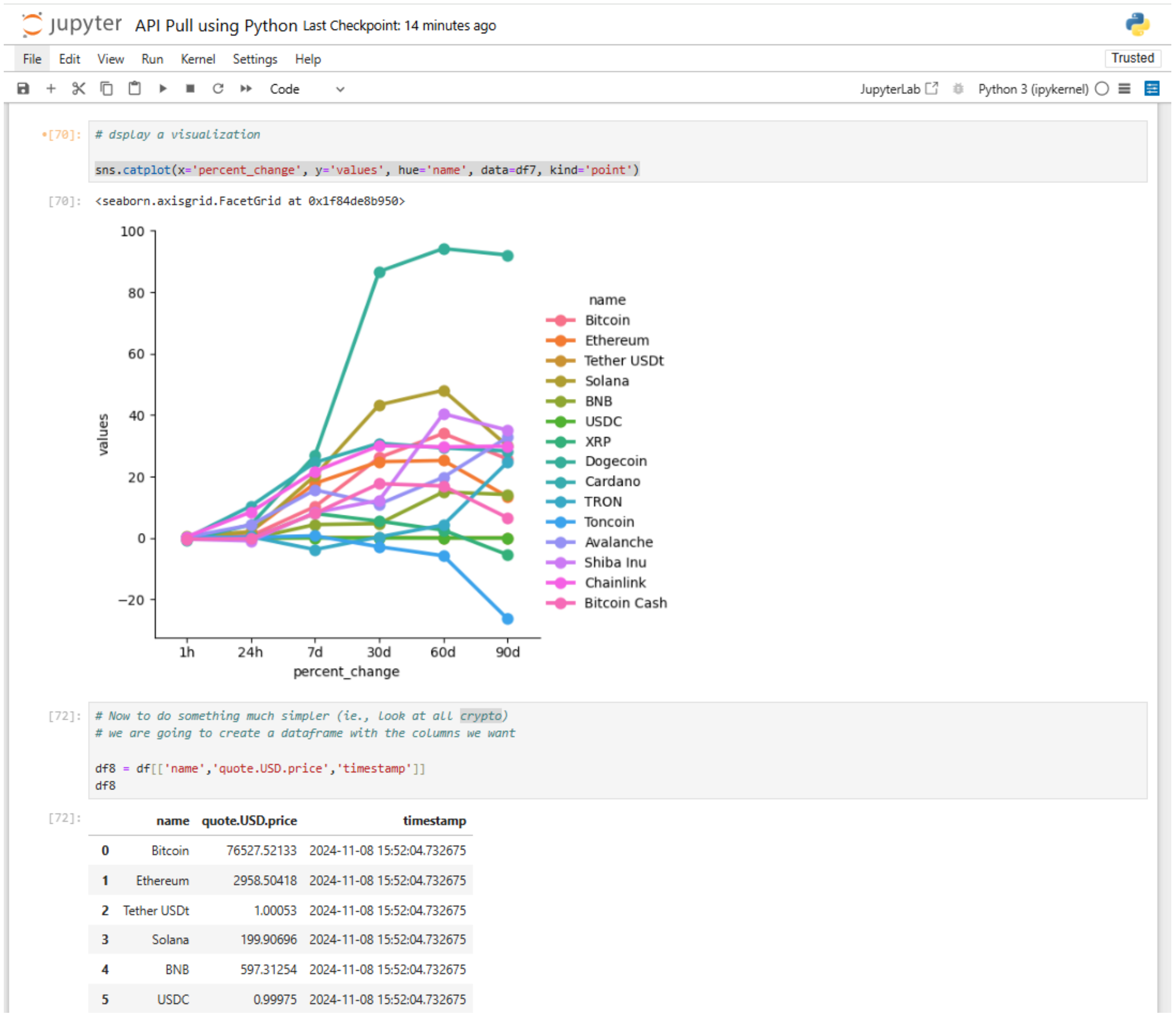
```
•[62]: # dsplay a visualization

       sns.catplot(x='percent_change', y='values', hue='name', data=df7, kind='point')

 [62]: <seaborn.axisgrid.FacetGrid at 0x1f849b97ef0>
```



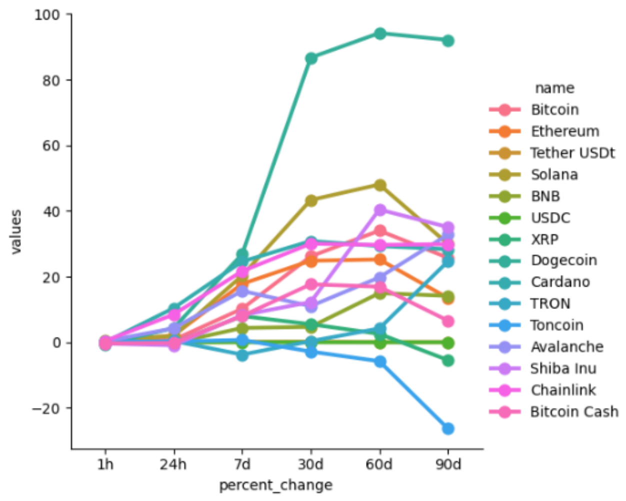
```
•[68]: # cleanup the percent_change data display

       df7['percent_change'] = df7['percent_change'].replace(['quote.USD.percent_change_1h','quote.USD.percent_change_24h','quote.USD.percent_change_7d','quote.
       df7
```

[68]:

	name	percent_change	values
0	Bitcoin	1h	0.08978
1	Bitcoin	24h	0.68042
2	Bitcoin	7d	10.18772
3	Bitcoin	30d	26.16167


jupyter API Pull using Python Last Checkpoint: 14 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

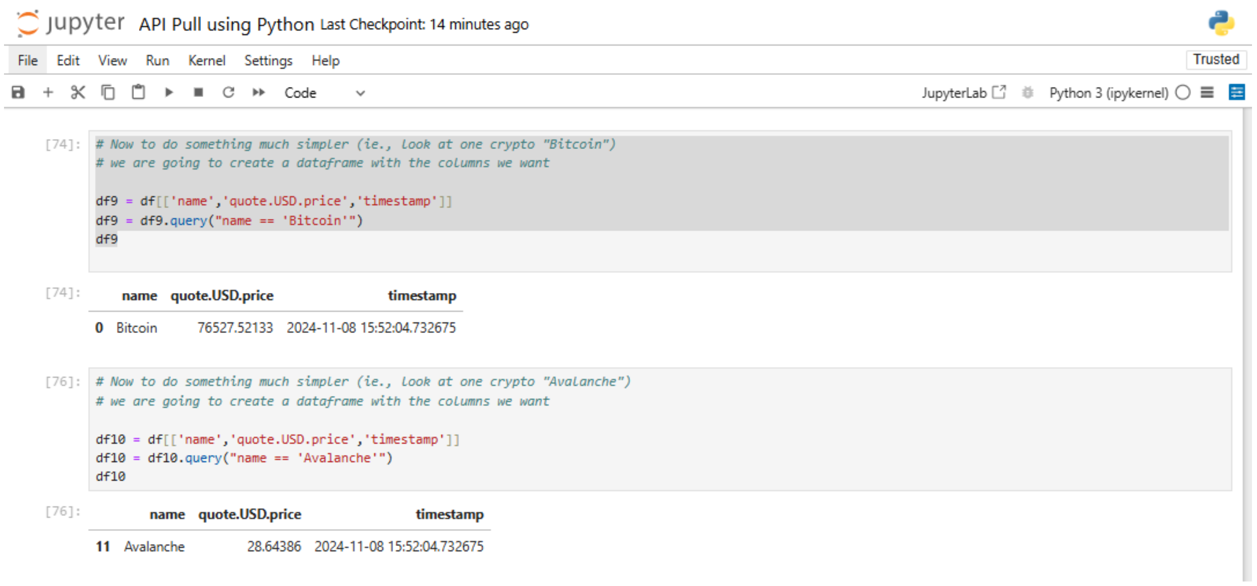
```
•[70]: # dsplay a visualization

       sns.catplot(x='percent_change', y='values', hue='name', data=df7, kind='point')

[70]: <seaborn.axisgrid.FacetGrid at 0x1f84de8b950>
```



```
[72]: # Now to do something much simpler (ie., look at all crypto)
      # we are going to create a dataframe with the columns we want

      df8 = df[['name','quote.USD.price','timestamp']]
      df8
```

[72]:

	name	quote.USD.price	timestamp
0	Bitcoin	76527.52133	2024-11-08 15:52:04.732675
1	Ethereum	2958.50418	2024-11-08 15:52:04.732675
2	Tether USDt	1.00053	2024-11-08 15:52:04.732675
3	Solana	199.90696	2024-11-08 15:52:04.732675
4	BNB	597.31254	2024-11-08 15:52:04.732675
5	USDC	0.99975	2024-11-08 15:52:04.732675


jupyter API Pull using Python Last Checkpoint: 14 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[74]: # Now to do something much simpler (ie., look at one crypto "Bitcoin")
      # we are going to create a dataframe with the columns we want

      df9 = df[['name','quote.USD.price','timestamp']]
      df9 = df9.query("name == 'Bitcoin'")
      df9
```

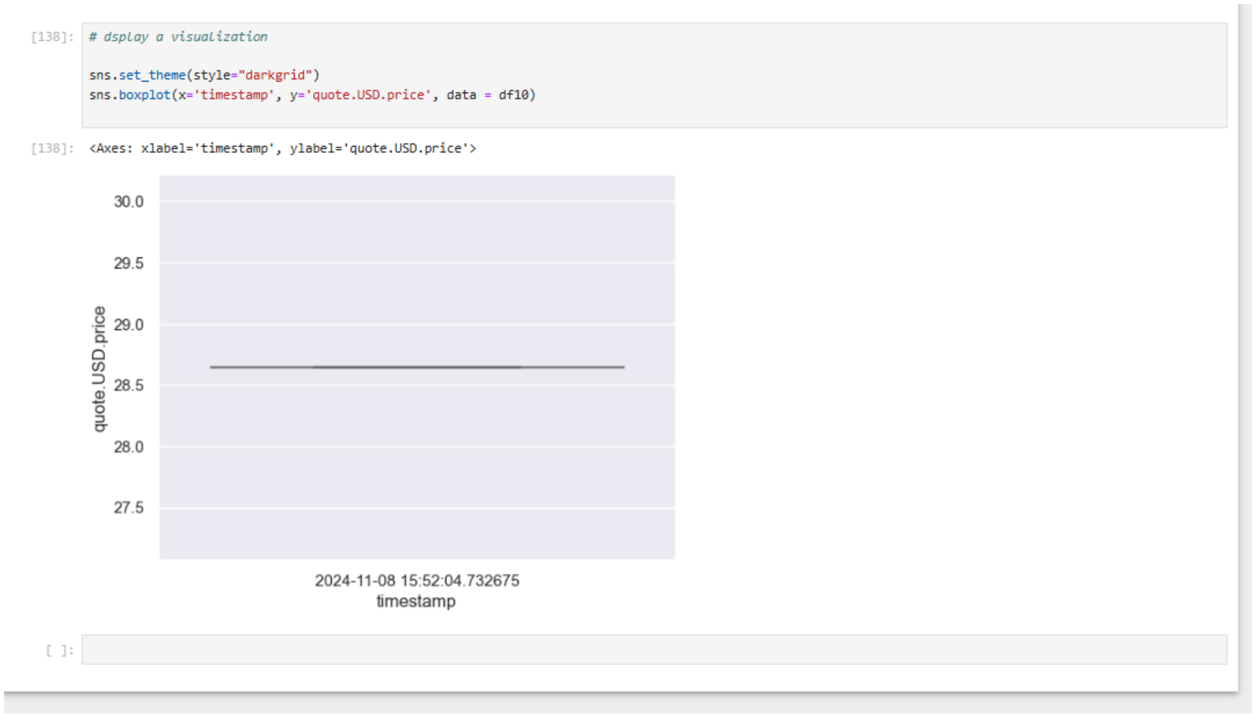
[74]:

	name	quote.USD.price	timestamp
0	Bitcoin	76527.52133	2024-11-08 15:52:04.732675

```
[76]: # Now to do something much simpler (ie., look at one crypto "Avalanche")
      # we are going to create a dataframe with the columns we want

      df10 = df[['name','quote.USD.price','timestamp']]
      df10 = df10.query("name == 'Avalanche'")
      df10
```

[76]:

	name	quote.USD.price	timestamp
11	Avalanche	28.64386	2024-11-08 15:52:04.732675



```
[138]: # dsplay a visualization

       sns.set_theme(style="darkgrid")
       sns.boxplot(x='timestamp', y='quote.USD.price', data = df10)

[138]: <Axes: xlabel='timestamp', ylabel='quote.USD.price'>
```

END*