Configuration Manual

MSc Research Project
Data Analytics

Tiernan Barry
Student ID: x20199121

School of Computing
National College of Ireland

Supervisor: Dr. Catherine Mulwa

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Tiernan Barry		
Student ID:	X20199121		
Programme:	Master of Science - Data Analytics	**Year:**	2021
Module:	Master of Science - Research Project		
Lecturer:	Dr. Catherine Mulwa		
Submission Due Date:	16/08/2021		
Project Title:	A Comparative Approach between Batch and Online Machine Learning for Predicting Next-Minute Cryptocurrency Price Direction		

Word Count: 10,003 **Page Count: 33**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Tiernan Barry…………………………………………………………………………

Date: 15/08/2021…………………………………………………………………………

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Tiernan Barry, Student ID: x20199121

Contents

1 Overview 2

2 Analytics Tier Configuration: 2

2.1 Hardware: 2

2.2 Software: 2

2.3 Open code in PyCharm/IDE: 3

2.4 Create conda environment: 3

2.5 Set PyCharm project interpreter: 4

3 Analytics Tier Deployment: 5

3.1 Please note: 5

3.2 Update config.py file with output location: 6

3.3 Feature Selection: 7

3.4 Batch Grid Search: 8

3.5 Batch Machine Learning: 11

3.6 Online Machine Learning: 12

4 Visual Tier Configuration: 14

4.1 Setting up and SSH into an AWS EC2 instance: 16

4.2 Installing R and RStudio onto Ubuntu AWS EC2 18

4.3 Get code onto server (clone from Github) 20

4.4 Installing and Configuring Shiny service 21

4.5 Install R Packages: 22

4.6 Launch Visual Tier Dashboard (Shiny App): 23

5 Data Persistent Tier Configuration: 24

5.1 Create a new EC2 Ubuntu Instance: 24

5.2 Create CloudFormation template: 24

5.3 Create Schedule in Dynamo DB: 27

5.4 Apply Instance Scheduling tag to EC2 instance: 29

5.5 Cron-Job configuration: 30

6 References 33

1 Overview

This document provides a detailed, step-by-step manual for deploying this research project. Because this research utilises 3 different operating systems for each tier in the design (Visual Tier, Analytics Tier and Data Persistent Tier), this manual provides procedures for each, starting with the Analytics Tier. **Note: To replicate the results, only the Analytics Tier needs to be configured.**

2 Analytics Tier Configuration:

This is local desktop machine where the vast majority of analytics was developed:

- Feature selection
- Grid Search
- Batch Machine Learning
- Online Machine Learning

2.1 Hardware:

The following hardware is configured by default on current laptop (Analytics Tier). These are therefore not prerequisites:

- Laptop/Desktop Computer: HP Pavilion Power Laptop 15-cb0xx
- CPU/Processor: Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz, 2496 Mhz, 4 Core(s), 4 Logical Processor(s)
- RAM: 16GB
- Graphics Card: Nividia GEFORCE GTX

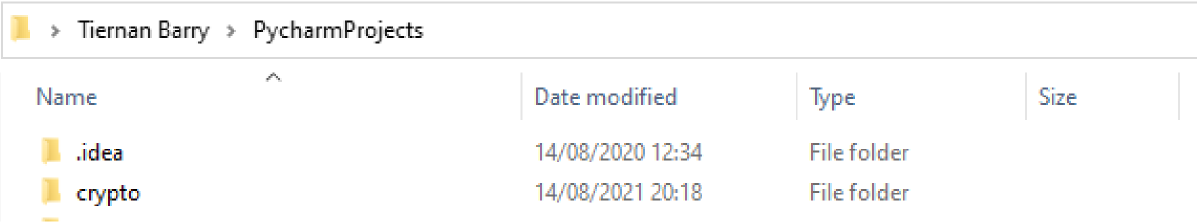
2.2 Software:

Similarly, the following software is configured on current laptop (Analytics Tier). While the following are not prerequisites, it will make life easier for replicating results.

- Operating system: Microsoft Windows 10 Home
- Interactive Development Environment (IDE): Pycharm Community Edition 2019.2.3
  - Note: Any other IDE will work fine too, but this document is PyCharm centric.
- Anaconda Python 3 Distribution:
  - Version: 4.10.1

2.3 Open code in PyCharm/IDE:

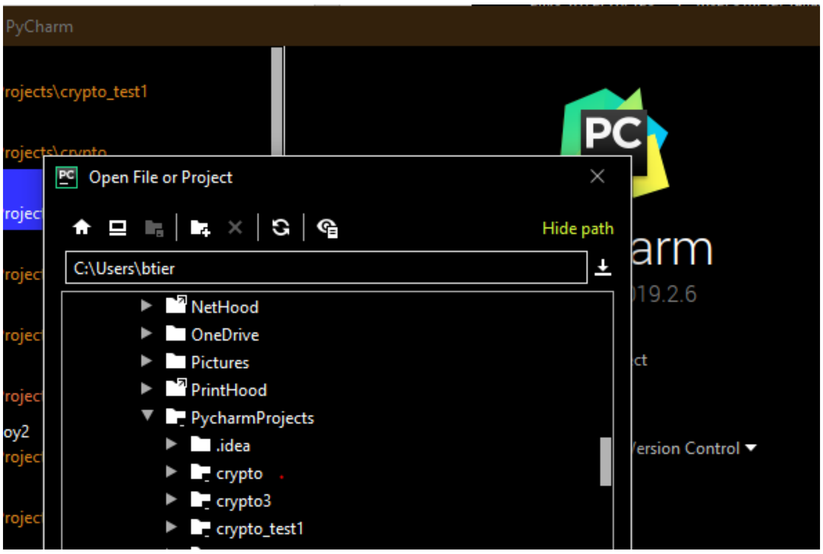
- Open Windows Explorer, and navigate to PyCharm folder (if using another IDE, go to wherever the desired/default location is)
- Unzip the source code into the folder, until you can see the project root folder (crypto) directly under PyCharmProjects like so:



Tiernan Barry > PycharmProjects

Name	Date modified	Type	Size
.idea	14/08/2020 12:34	File folder	
crypto	14/08/2021 20:18	File folder	

- Open the PyCharm IDE, and then open the 'crypto' project:



C:\Users\btier

- Now, you will see the folder structure on the LHS of the screen. Next, we need to install packages in the following section using conda.
- Note: All code is also version controlled using a Github private repository. Please reach out if access to this is needed.

2.4 Create conda environment:

Once Anaconda is installed, a conda environment can now be created using the 'env.yml' file provided in the source code repository (crypto\env.yml). This will install all required packages for the Analytics Tier, and will avoid having to manually install packages individually (Anaconda, 2021):

- Launch Anaconda Prompt as follows
- Open 'Anaconda Prompt' from the Start Menu. A terminal will pop up.

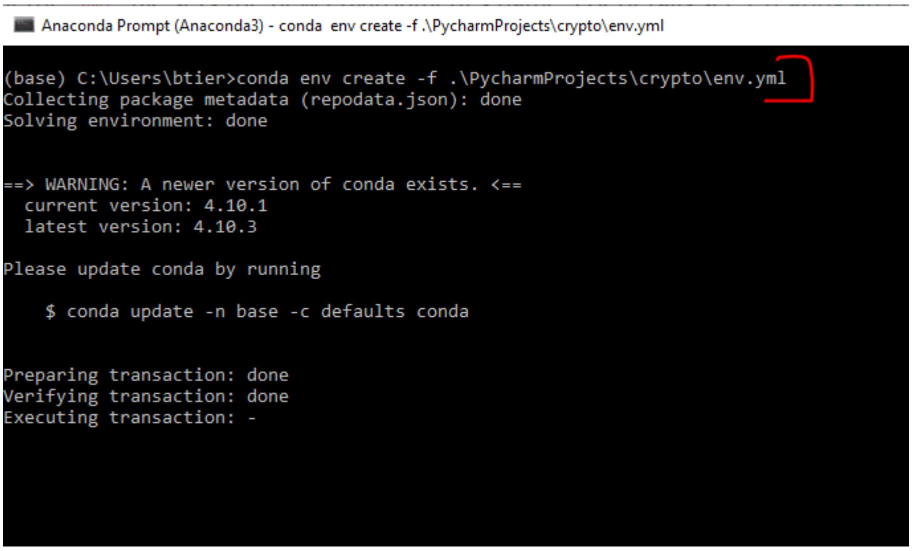
- Depending on your folder structure, run the following terminal command by providing the path to env.yml as follows:

conda env create -f .\path\to\env.yml

- If using PyCharm it should look something like this:

conda env create -f .\PycharmProjects\crypto\Scripts\env.yml

- The packages will begin installing as follows:



Anaconda Prompt (Anaconda3) - conda env create -f .\PycharmProjects\crypto\env.yml

```
(base) C:\Users\btier>conda env create -f .\PycharmProjects\crypto\env.yml
Collecting package metadata (repodata.json): done
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.10.1
  latest version: 4.10.3

Please update conda by running

    $ conda update -n base -c defaults conda


Preparing transaction: done
Verifying transaction: done
Executing transaction: -
```

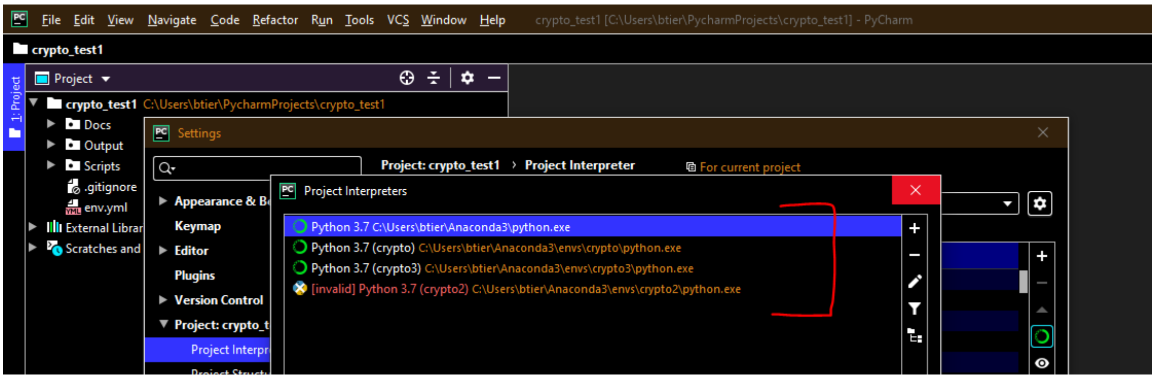
- Once complete (few minutes), run the following command to activate the conda environment:

conda activate crypto3

- crypto3 is the name of the conda environment (as per the env.yml file), which now needs to be applied as the interpreter in PyCharm as follows.

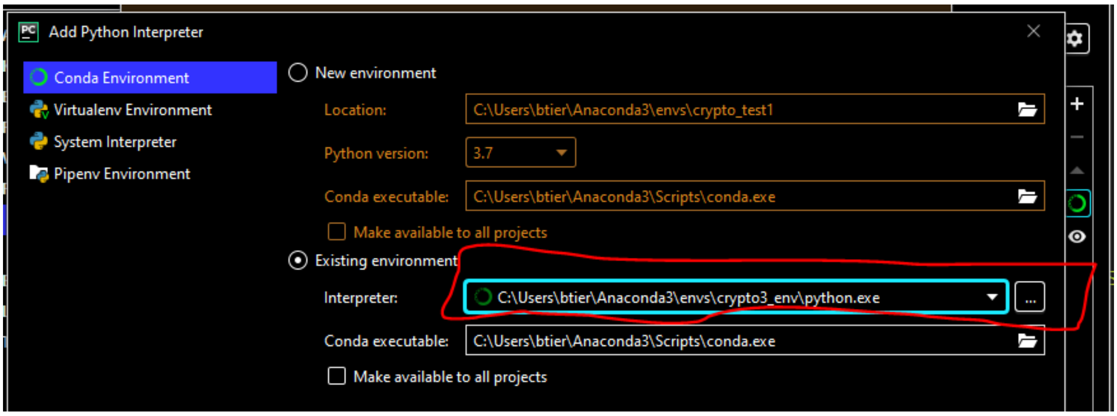
2.5 Set PyCharm project interpreter:

- In Pycharm, go to File > Settings > Project Interpreter
- Show all interpreters (drop-down):


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crypto_test1 [C:\Users\btier\PycharmProjects\crypto_test1] - PyCharm
crypto_test1
1: Project
Project
crypto_test1 C:\Users\btier\PycharmProjects\crypto_test1
Docs
Output
Scripts
.gitignore
env.yml
External Librar
Scratches and
Settings
Appearance & B
Keymap
Editor
Plugins
Version Control
Project: crypto_t
Project Interpr
Project: crypto_test1 › Project Interpreter
For current project
Project Interpreters

Python 3.7 C:\Users\btier\Anaconda3\python.exe
Python 3.7 (crypto) C:\Users\btier\Anaconda3\envs\crypto\python.exe
Python 3.7 (crypto3) C:\Users\btier\Anaconda3\envs\crypto3\python.exe
[invalid] Python 3.7 (crypto2) C:\Users\btier\Anaconda3\envs\crypto2\python.exe

- If crypto3 is not present as shown above, click the + sign to the RHS as follows. This will allow the interpreted to be added PyCharm. Find the required environment from the drop down, and apply this by clicking ‘Okay’.



○ New environment

Location: C:\Users\btier\Anaconda3\envs\crypto_test1

Python version: 3.7

Conda executable: C:\Users\btier\Anaconda3\Scripts\conda.exe

□ Make available to all projects

● Existing environment

Interpreter: C:\Users\btier\Anaconda3\envs\crypto3_env\python.exe

Conda executable: C:\Users\btier\Anaconda3\Scripts\conda.exe

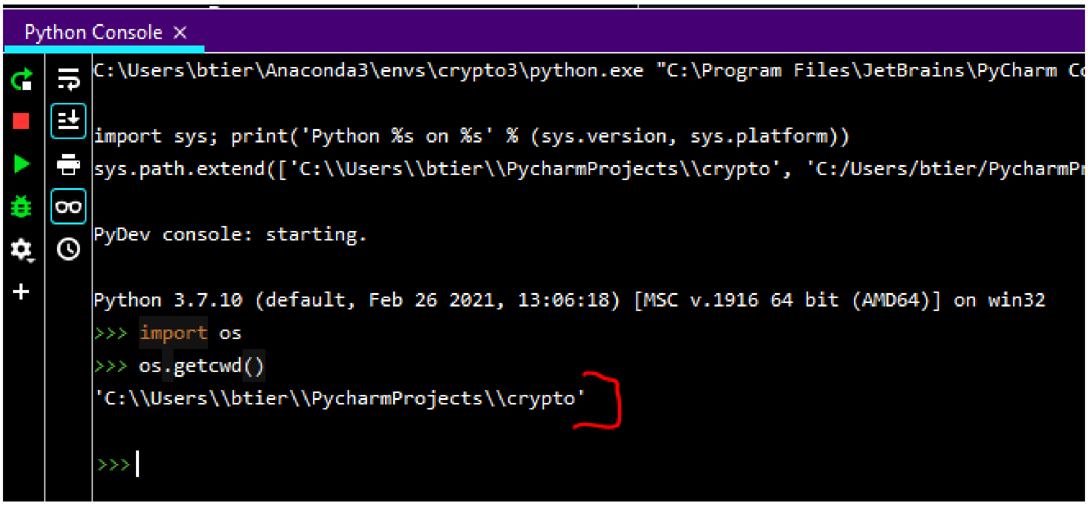
□ Make available to all projects

- Now, all packages will be available within PyCharm, and we can now deploy the analytics.

3 Analytics Tier Deployment:

3.1 Please note:

- The code is ran from the PyCharm Console
- **Because of this, the working directory automatically defaults to the root project folder of ‘crypto’.**
- **Working directory being set to ‘crypto’ is a prerequisite for running the code. In my case, the working directory is:**
  - **C:\Users\btier\PycharmProjects\crypto**
  - **Check your working directory is ‘crypto’:**

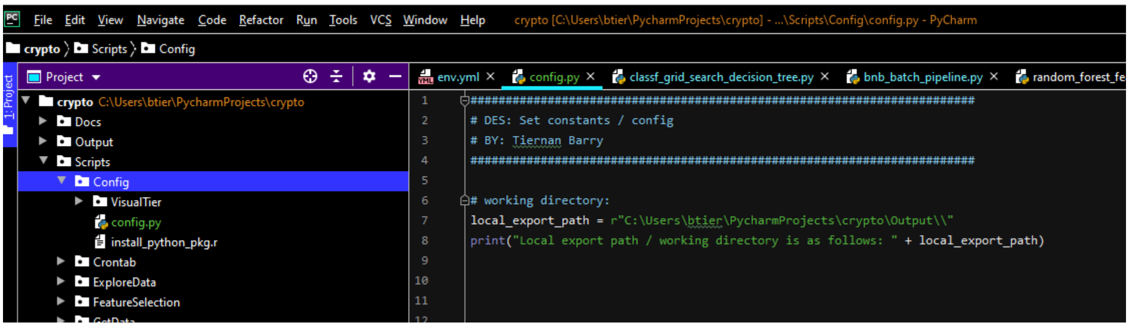


```
C:\Users\btier\Anaconda3\envs\crypto3\python.exe "C:\Program Files\JetBrains\PyCharm Co

import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['C:\\Users\\btier\\PycharmProjects\\crypto', 'C:/Users/btier/PycharmPr

PyDev console: starting.

Python 3.7.10 (default, Feb 26 2021, 13:06:18) [MSC v.1916 64 bit (AMD64)] on win32
>>> import os
>>> os.getcwd()
'C:\\Users\\btier\\PycharmProjects\\crypto'

>>>
```

3.2 Update config.py file with output location:

To make the results easier to reproduce, a config python script is defined in the below location. Please update this with a local output folder location (just the folder, not a file name) for writing results to. This is needed to upload files to S3:

- crypto\Scripts\Config\config.py
- Note that there is a double '\\' at the end. Please ensure this is applied.

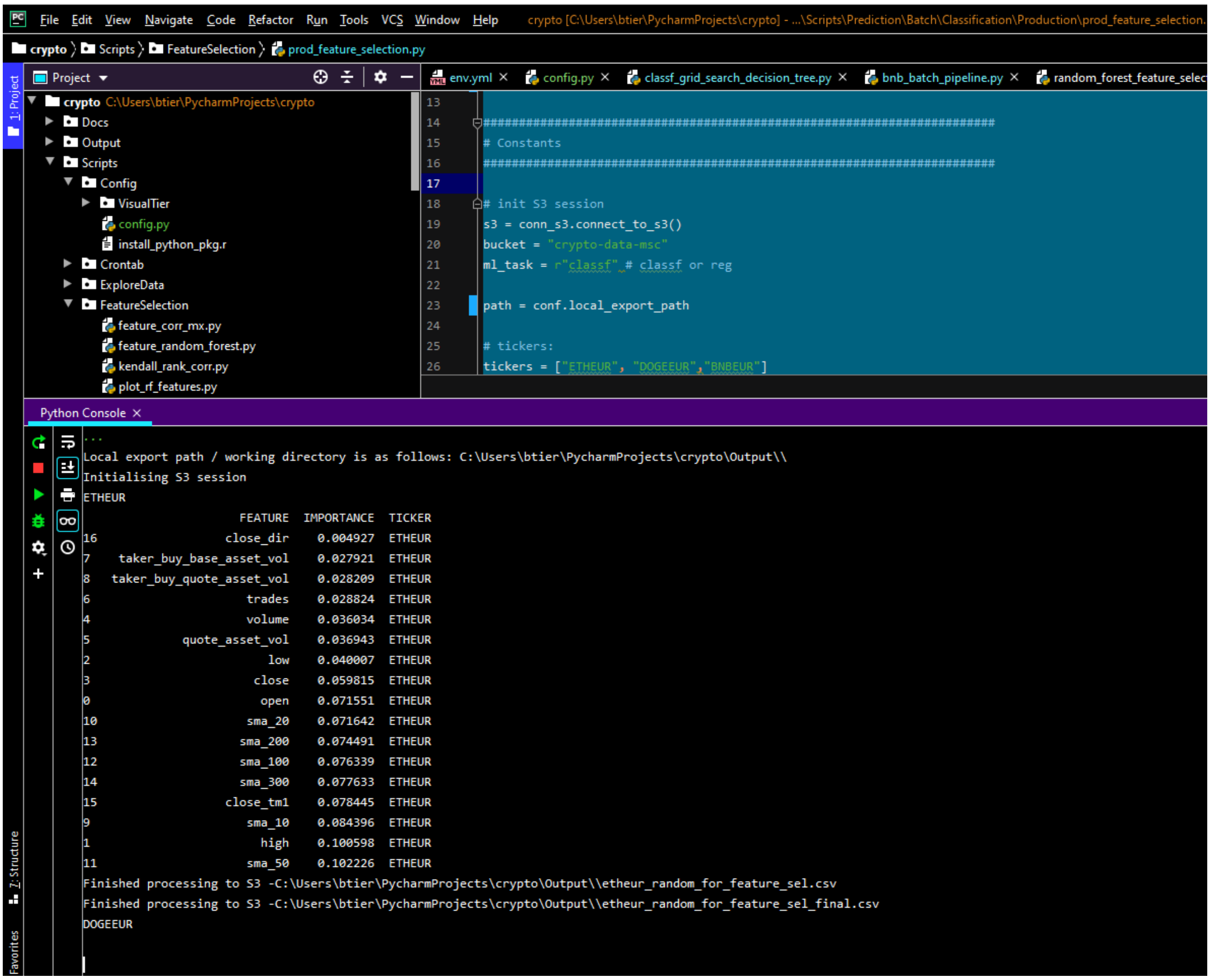


```
########################################################################
# DES: Set constants / config
# BY: Tiernan Barry
########################################################################

# working directory:
local_export_path = r"C:\Users\btier\PycharmProjects\crypto\Output\\"
print("Local export path / working directory is as follows: " + local_export_path)
```

3.3 Feature Selection:

Feature selection is run for all 3 alt-coin as follows:

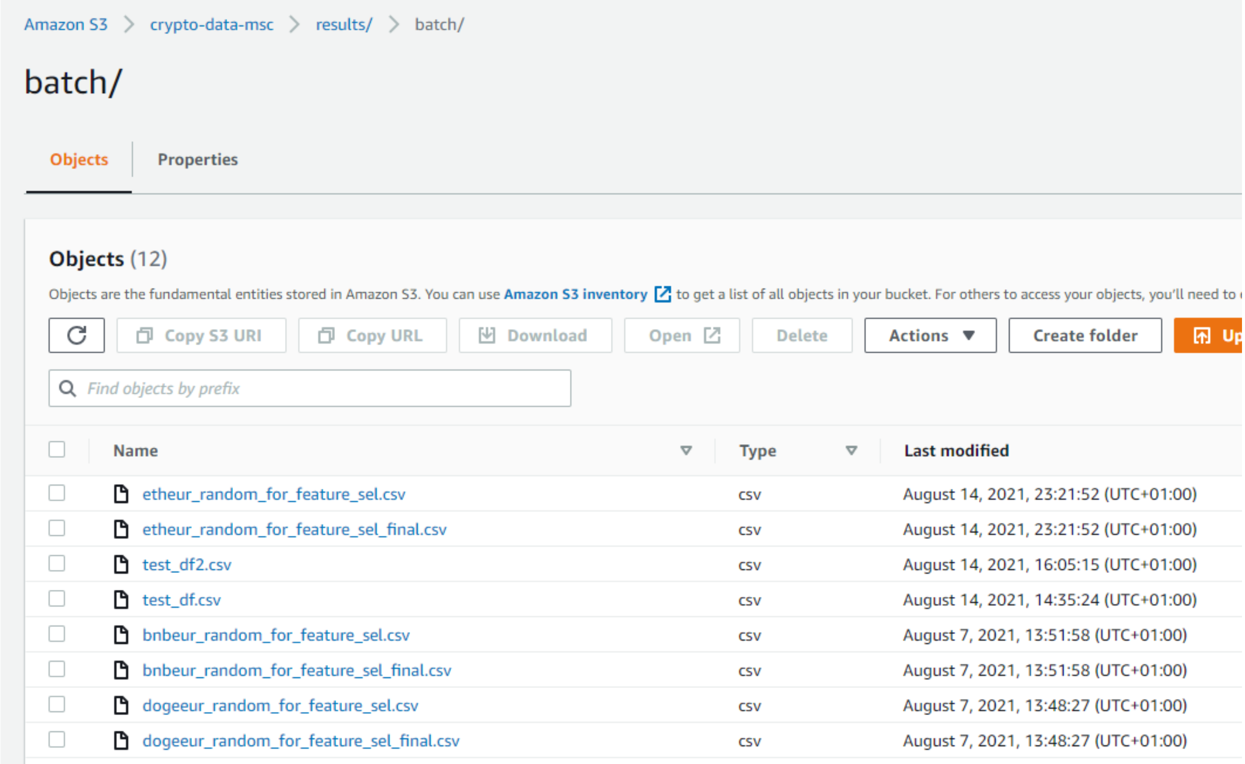
- crypto\Scripts\FeatureSelection\prod_feature_selection.py


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crypto [C:\Users\btier\PycharmProjects\crypto] - ...\Scripts\Prediction\Batch\Classification\Production\prod_feature_selection.
crypto 〉 Scripts 〉 FeatureSelection 〉 prod_feature_selection.py
Project
env.yml
config.py
classf_grid_search_decision_tree.py
bnb_batch_pipeline.py
random_forest_feature_selec
crypto C:\Users\btier\PycharmProjects\crypto
Docs
Output
Scripts
Config
VisualTier
config.py
install_python_pkg.r
Crontab
ExploreData
FeatureSelection
feature_corr_mx.py
feature_random_forest.py
kendall_rank_corr.py
plot_rf_features.py
Python Console
1: Project
7: Structure
Favorites

```
########################################################################
# Constants
########################################################################

# init S3 session
s3 = conn_s3.connect_to_s3()
bucket = "crypto-data-msc"
ml_task = r"classf" # classf or reg

path = conf.local_export_path

# tickers:
tickers = ["ETHEUR", "DOGEEUR","BNBEUR"]
```

```
...
Local export path / working directory is as follows: C:\Users\btier\PycharmProjects\crypto\Output\\
Initialising S3 session
ETHEUR
                   FEATURE  IMPORTANCE  TICKER
16               close_dir    0.004927  ETHEUR
7     taker_buy_base_asset_vol    0.027921  ETHEUR
8    taker_buy_quote_asset_vol    0.028209  ETHEUR
6                   trades    0.028824  ETHEUR
4                   volume    0.036034  ETHEUR
5           quote_asset_vol    0.036943  ETHEUR
2                      low    0.040007  ETHEUR
3                    close    0.059815  ETHEUR
0                     open    0.071551  ETHEUR
10                  sma_20    0.071642  ETHEUR
13                 sma_200    0.074491  ETHEUR
12                 sma_100    0.076339  ETHEUR
14                 sma_300    0.077633  ETHEUR
15               close_tm1    0.078445  ETHEUR
9                   sma_10    0.084396  ETHEUR
1                     high    0.100598  ETHEUR
11                  sma_50    0.102226  ETHEUR
Finished processing to S3 -C:\Users\btier\PycharmProjects\crypto\Output\\etheur_random_for_feature_sel.csv
Finished processing to S3 -C:\Users\btier\PycharmProjects\crypto\Output\\etheur_random_for_feature_sel_final.csv
DOGEEUR
```

- As you can see at the end, the selected features are sent to AWS S3, and are then queried back during the following analyses. Please see below AWS S3 screenshot:



Amazon S3 > crypto-data-msc > results/ > batch/

batch/

Objects (12)

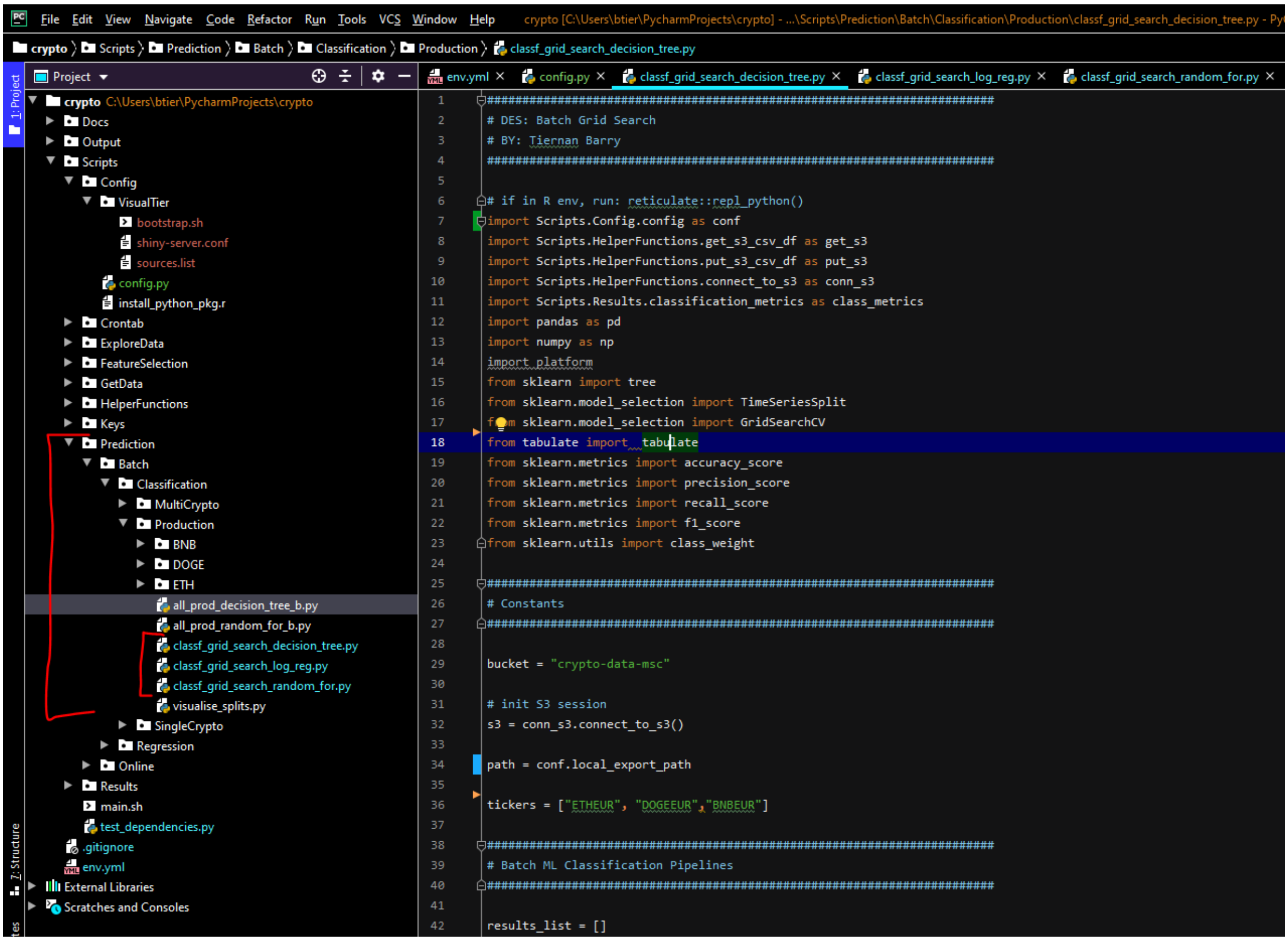
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to

□	Name	Type	Last modified
□	etheur_random_for_feature_sel.csv	csv	August 14, 2021, 23:21:52 (UTC+01:00)
□	etheur_random_for_feature_sel_final.csv	csv	August 14, 2021, 23:21:52 (UTC+01:00)
□	test_df2.csv	csv	August 14, 2021, 16:05:15 (UTC+01:00)
□	test_df.csv	csv	August 14, 2021, 14:35:24 (UTC+01:00)
□	bnbeur_random_for_feature_sel.csv	csv	August 7, 2021, 13:51:58 (UTC+01:00)
□	bnbeur_random_for_feature_sel_final.csv	csv	August 7, 2021, 13:51:58 (UTC+01:00)
□	dogeeur_random_for_feature_sel.csv	csv	August 7, 2021, 13:48:27 (UTC+01:00)
□	dogeeur_random_for_feature_sel_final.csv	csv	August 7, 2021, 13:48:27 (UTC+01:00)

3.4 Batch Grid Search:

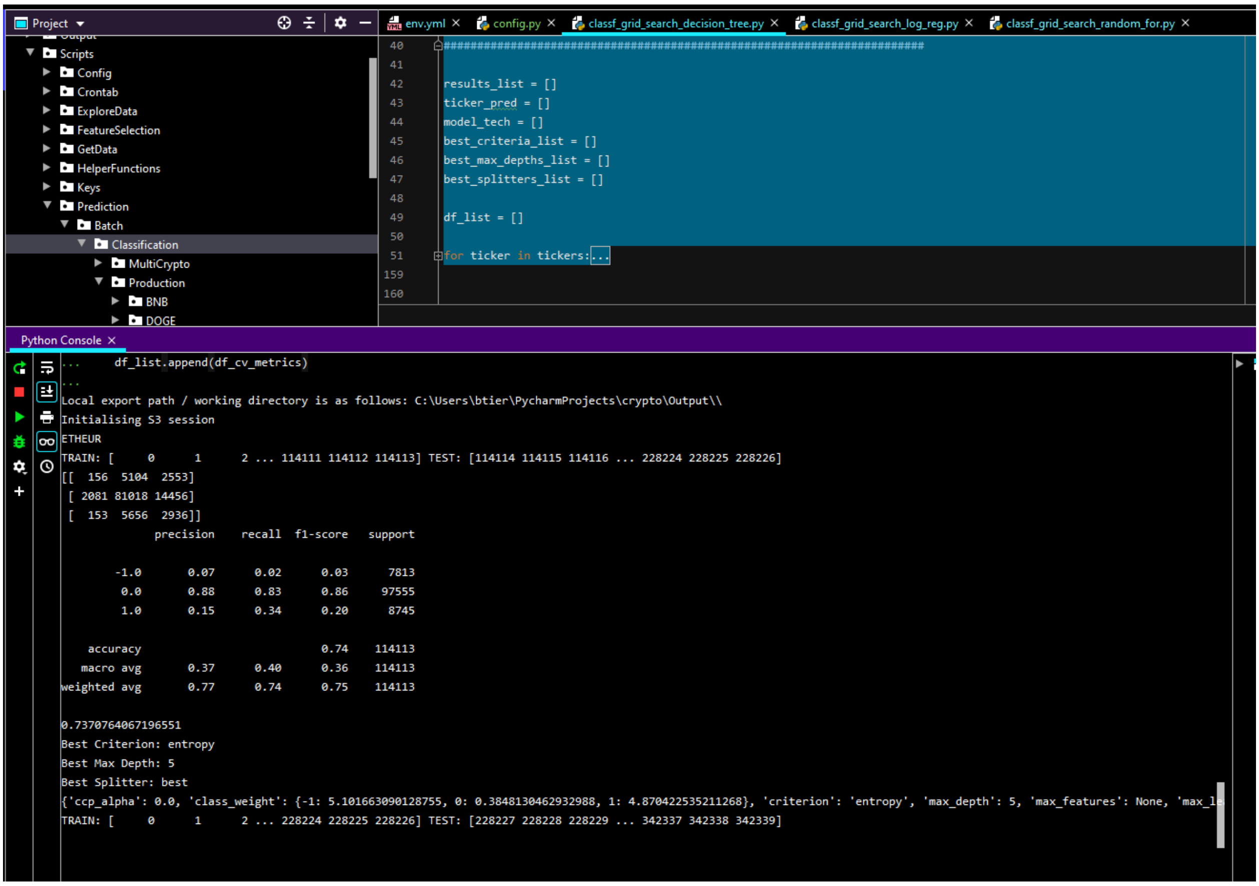
As discussed in the technical report, grid search is used to find the final parameters which are to be used. Grid search scripts are located in the following location for each batch model (runs for all 3 alt-coins):

- crypto\Scripts\Prediction\Batch\Classification\Production:
  - classf_grid_search_decision_tree.py
  - classf_grid_search_log_reg.py
  - classf_grid_search_random_for.py


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crypto [C:\Users\btier\PycharmProjects\crypto] - ...\Scripts\Prediction\Batch\Classification\Production\classf_grid_search_decision_tree.py - Py
crypto 〉 Scripts 〉 Prediction 〉 Batch 〉 Classification 〉 Production 〉 classf_grid_search_decision_tree.py
1: Project
Project
env.yml ×
config.py ×
classf_grid_search_decision_tree.py ×
classf_grid_search_log_reg.py ×
classf_grid_search_random_for.py ×
crypto C:\Users\btier\PycharmProjects\crypto
Docs
Output
Scripts
Config
VisualTier
bootstrap.sh
shiny-server.conf
sources.list
config.py
install_python_pkg.r
Crontab
ExploreData
FeatureSelection
GetData
HelperFunctions
Keys
Prediction
Batch
Classification
MultiCrypto
Production
BNB
DOGE
ETH
all_prod_decision_tree_b.py
all_prod_random_for_b.py
classf_grid_search_decision_tree.py
classf_grid_search_log_reg.py
classf_grid_search_random_for.py
visualise_splits.py
SingleCrypto
Regression
Online
Results
main.sh
test_dependencies.py
.gitignore
env.yml
External Libraries
Scratches and Consoles
Z: Structure

```
##########################################################################
# DES: Batch Grid Search
# BY: Tiernan Barry
##########################################################################

# if in R env, run: reticulate::repl_python()
import Scripts.Config.config as conf
import Scripts.HelperFunctions.get_s3_csv_df as get_s3
import Scripts.HelperFunctions.put_s3_csv_df as put_s3
import Scripts.HelperFunctions.connect_to_s3 as conn_s3
import Scripts.Results.classification_metrics as class_metrics
import pandas as pd
import numpy as np
import platform
from sklearn import tree
from sklearn.model_selection import TimeSeriesSplit
from sklearn.model_selection import GridSearchCV
from tabulate import tabulate
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.utils import class_weight

##########################################################################
# Constants
##########################################################################

bucket = "crypto-data-msc"

# init S3 session
s3 = conn_s3.connect_to_s3()

path = conf.local_export_path

tickers = ["ETHEUR", "DOGEEUR","BNBEUR"]

##########################################################################
# Batch ML Classification Pipelines
##########################################################################

results_list = []
```

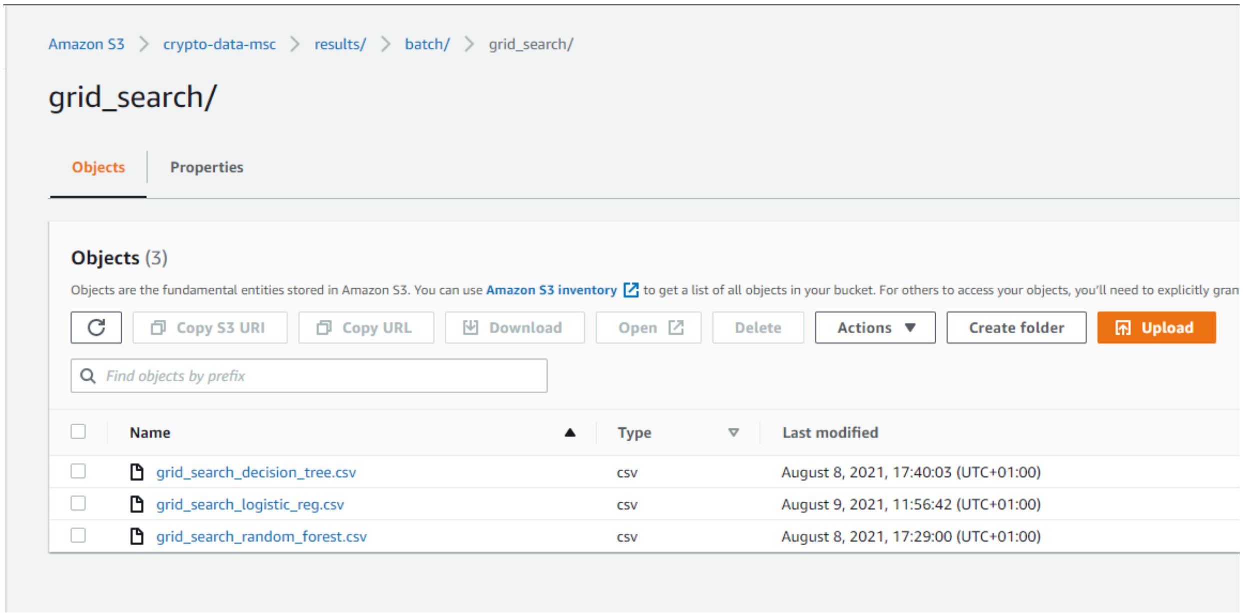
- For each script, once they have completed (can take significant time, 10+ hours in the case of classf_grid_search_log_reg.py), the results will be loaded into the AWS S3 buckets, which are then visualised through the Visual Tier. Below is a snippet of the code running, with time series splitting and print outs:



```
########################################################################

results_list = []
ticker_pred = []
model_tech = []
best_criteria_list = []
best_max_depths_list = []
best_splitters_list = []

df_list = []

for ticker in tickers:...
```

```
...     df_list.append(df_cv_metrics)
...
Local export path / working directory is as follows: C:\Users\btier\PycharmProjects\crypto\Output\\
Initialising S3 session
ETHEUR
TRAIN: [     0      1      2 ... 114111 114112 114113] TEST: [114114 114115 114116 ... 228224 228225 228226]
[[  156  5104  2553]
 [ 2081 81018 14456]
 [  153  5656  2936]]
              precision    recall  f1-score   support

        -1.0       0.07      0.02      0.03      7813
         0.0       0.88      0.83      0.86     97555
         1.0       0.15      0.34      0.20      8745

    accuracy                           0.74    114113
   macro avg       0.37      0.40      0.36    114113
weighted avg       0.77      0.74      0.75    114113

0.7370764067196551
Best Criterion: entropy
Best Max Depth: 5
Best Splitter: best
{'ccp_alpha': 0.0, 'class_weight': {-1: 5.101663090128755, 0: 0.3848130462932988, 1: 4.870422535211268}, 'criterion': 'entropy', 'max_depth': 5, 'max_features': None, 'max_le
TRAIN: [     0      1      2 ... 228224 228225 228226] TEST: [228227 228228 228229 ... 342337 342338 342339]
```

- Results output to S3:


Amazon S3 > crypto-data-msc > results/ > batch/ > grid_search/

grid_search/

Objects (3)

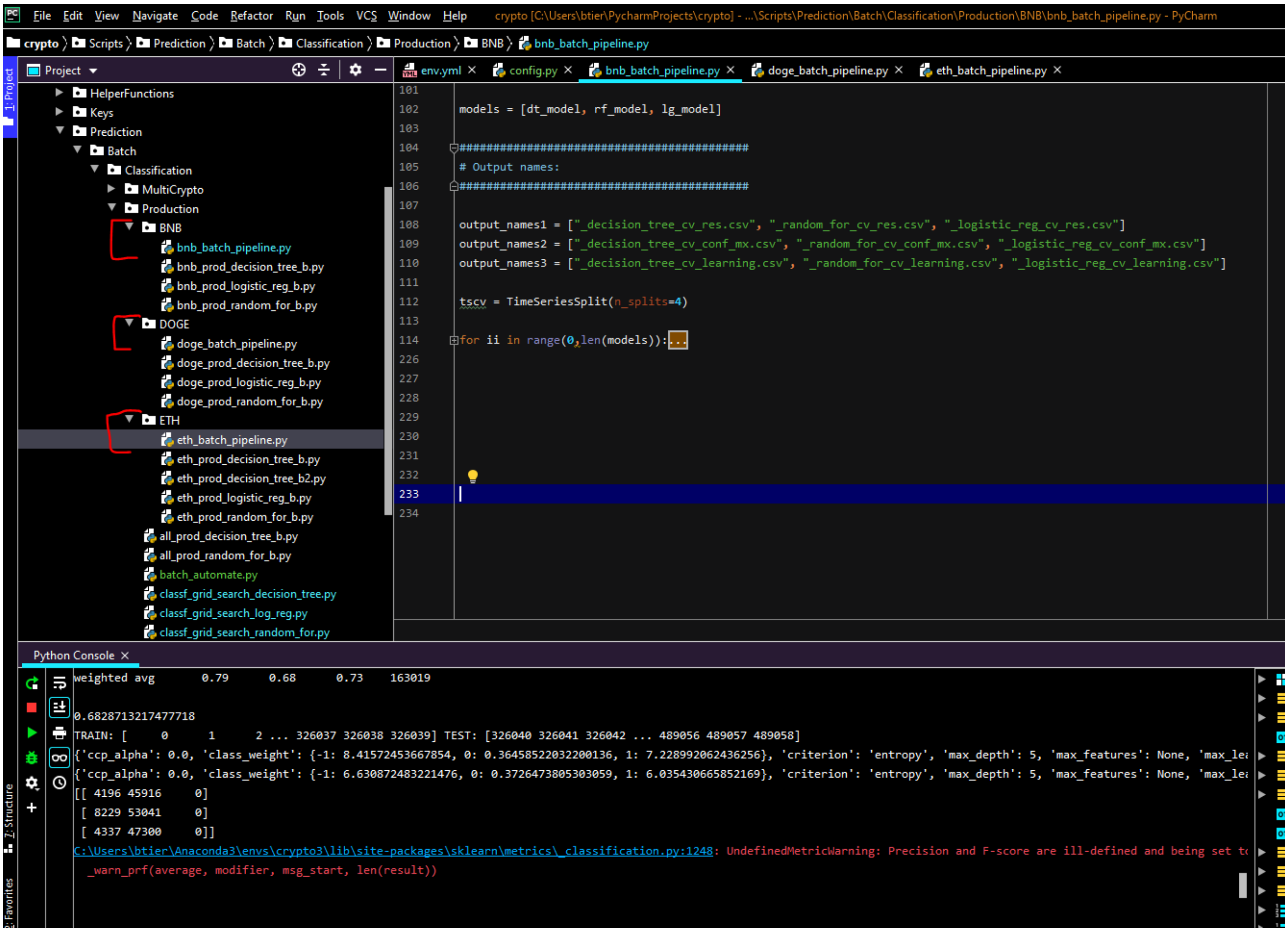
Objects are the fundamental entities stored in Amazon S3. You can use **Amazon S3 inventory** to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly gran

□	Name ▲	Type ▽	Last modified
□	grid_search_decision_tree.csv	csv	August 8, 2021, 17:40:03 (UTC+01:00)
□	grid_search_logistic_reg.csv	csv	August 9, 2021, 11:56:42 (UTC+01:00)
□	grid_search_random_forest.csv	csv	August 8, 2021, 17:29:00 (UTC+01:00)

3.5 Batch Machine Learning:

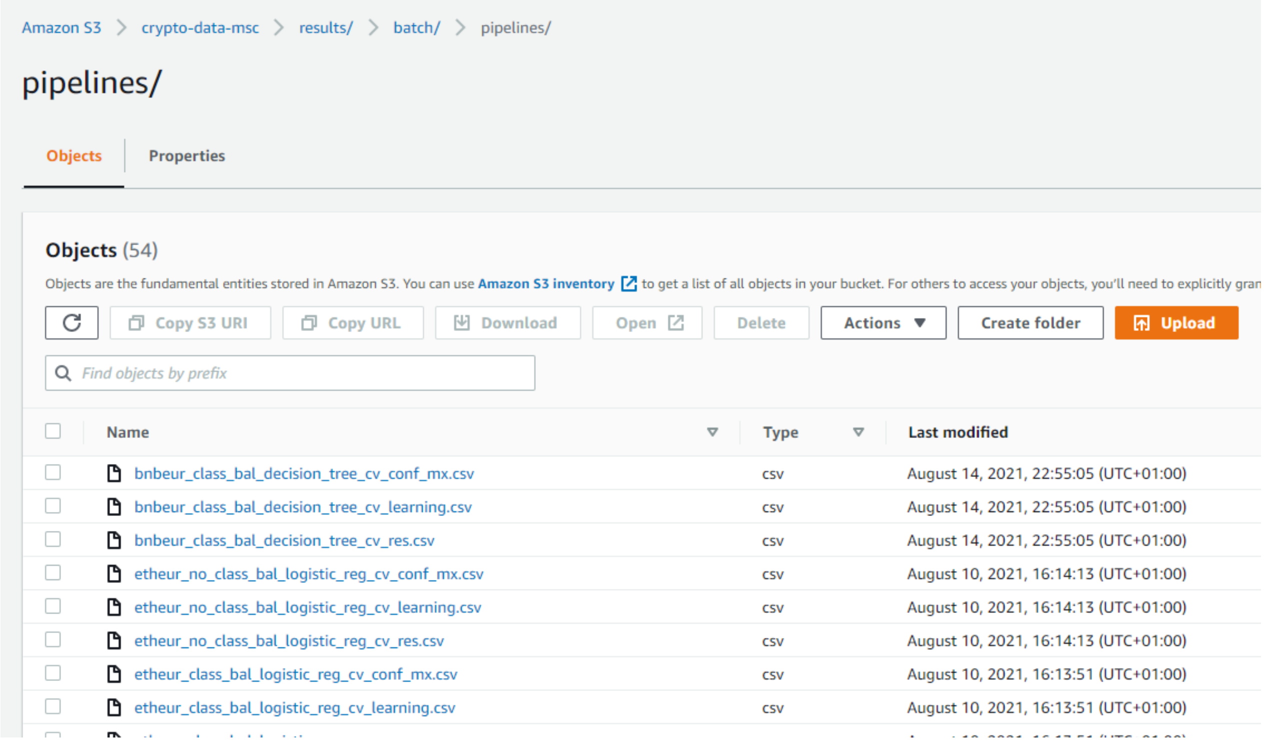
For each alt-coin, 1 batch machine learning script is developed for all 3 models under the following files. Within each file, a batch decision tree, logistic regression and random forest was developed.

- crypto\Scripts\Prediction\Batch\Classification\Production:
    - \BNB\bnb_batch_pipeline.py
    - \DOGE\ doge_batch_pipeline.py
    - \ETH\eth_batch_pipeline.py
- As each of these scripts run, some results are printed out as follows, while the final results are also sent to AWS S3. Final results also feed into the Visual Tier from AWS S3.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crypto [C:\Users\btier\PycharmProjects\crypto] - ...\Scripts\Prediction\Batch\Classification\Production\BNB\bnb_batch_pipeline.py - PyCharm
crypto › Scripts › Prediction › Batch › Classification › Production › BNB › bnb_batch_pipeline.py
Project
env.yml
config.py
bnb_batch_pipeline.py
doge_batch_pipeline.py
eth_batch_pipeline.py
HelperFunctions
Keys
Prediction
Batch
Classification
MultiCrypto
Production
BNB
bnb_batch_pipeline.py
bnb_prod_decision_tree_b.py
bnb_prod_logistic_reg_b.py
bnb_prod_random_for_b.py
DOGE
doge_batch_pipeline.py
doge_prod_decision_tree_b.py
doge_prod_logistic_reg_b.py
doge_prod_random_for_b.py
ETH
eth_batch_pipeline.py
eth_prod_decision_tree_b.py
eth_prod_decision_tree_b2.py
eth_prod_logistic_reg_b.py
eth_prod_random_for_b.py
all_prod_decision_tree_b.py
all_prod_random_for_b.py
batch_automate.py
classf_grid_search_decision_tree.py
classf_grid_search_log_reg.py
classf_grid_search_random_for.py
Python Console
1: Project
7: Structure
2: Favorites

```

models = [dt_model, rf_model, lg_model]

############################################
# Output names:
############################################

output_names1 = ["_decision_tree_cv_res.csv", "_random_for_cv_res.csv", "_logistic_reg_cv_res.csv"]
output_names2 = ["_decision_tree_cv_conf_mx.csv", "_random_for_cv_conf_mx.csv", "_logistic_reg_cv_conf_mx.csv"]
output_names3 = ["_decision_tree_cv_learning.csv", "_random_for_cv_learning.csv", "_logistic_reg_cv_learning.csv"]

tscv = TimeSeriesSplit(n_splits=4)

for ii in range(0,len(models)):...
```

```
weighted avg       0.79      0.68      0.73    163019

0.6828713217477718
TRAIN: [     0      1      2 ... 326037 326038 326039] TEST: [326040 326041 326042 ... 489056 489057 489058]
{'ccp_alpha': 0.0, 'class_weight': {-1: 8.41572453667854, 0: 0.36458522032200136, 1: 7.228992062436256}, 'criterion': 'entropy', 'max_depth': 5, 'max_features': None, 'max_lea
{'ccp_alpha': 0.0, 'class_weight': {-1: 6.630872483221476, 0: 0.3726473805303059, 1: 6.035430665852169}, 'criterion': 'entropy', 'max_depth': 5, 'max_features': None, 'max_lea
[[ 4196 45916     0]
 [ 8229 53041     0]
 [ 4337 47300     0]]
C:\Users\btier\Anaconda3\envs\crypto3\lib\site-packages\sklearn\metrics\_classification.py:1248: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
  _warn_prf(average, modifier, msg_start, len(result))
```



Amazon S3 > crypto-data-msc > results/ > batch/ > pipelines/
pipelines/
Objects
Properties

Objects (54)

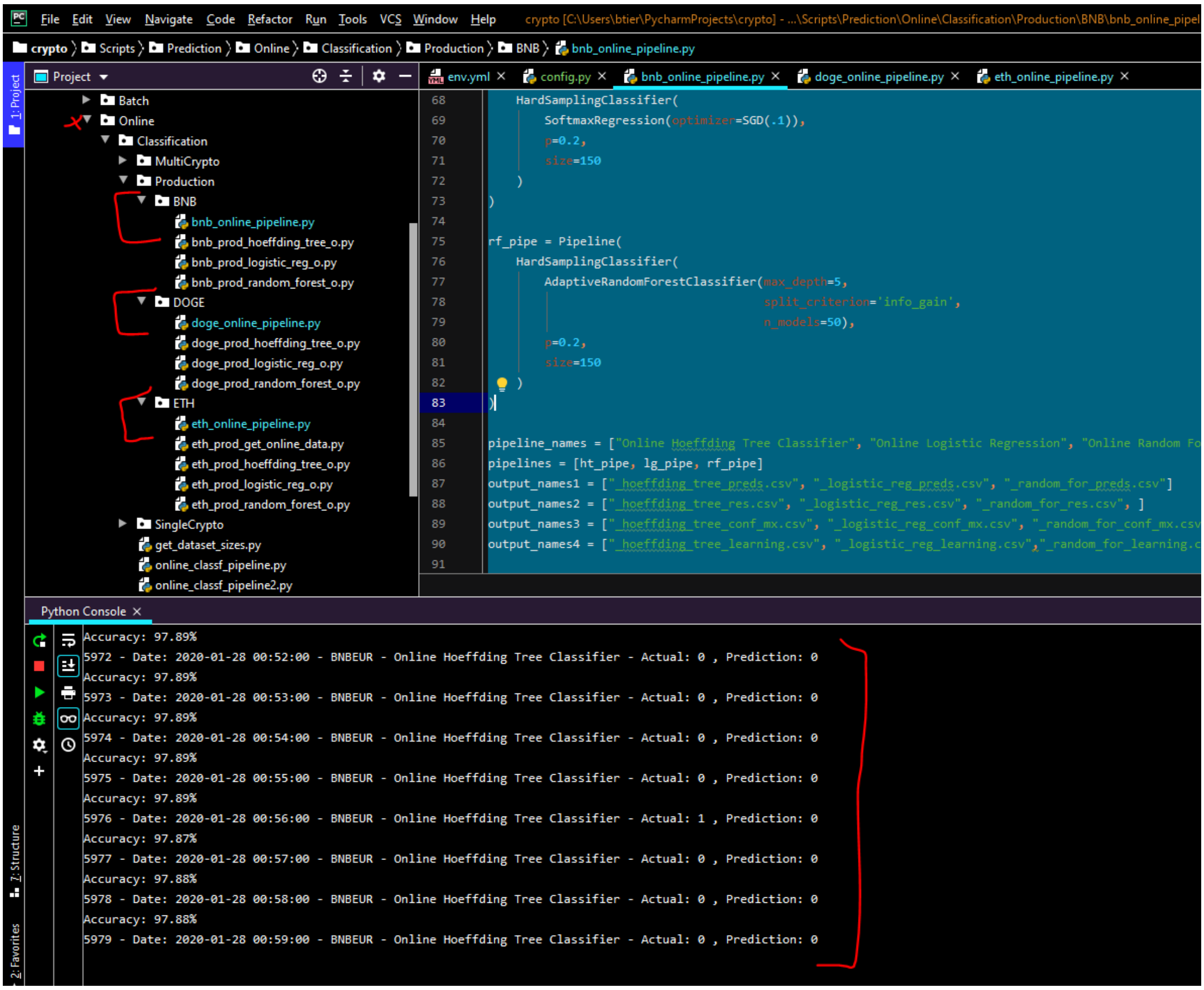
Objects are the fundamental entities stored in Amazon S3. You can use **Amazon S3 inventory** to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly gran

□	Name	Type	Last modified
□	bnbeur_class_bal_decision_tree_cv_conf_mx.csv	csv	August 14, 2021, 22:55:05 (UTC+01:00)
□	bnbeur_class_bal_decision_tree_cv_learning.csv	csv	August 14, 2021, 22:55:05 (UTC+01:00)
□	bnbeur_class_bal_decision_tree_cv_res.csv	csv	August 14, 2021, 22:55:05 (UTC+01:00)
□	etheur_no_class_bal_logistic_reg_cv_conf_mx.csv	csv	August 10, 2021, 16:14:13 (UTC+01:00)
□	etheur_no_class_bal_logistic_reg_cv_learning.csv	csv	August 10, 2021, 16:14:13 (UTC+01:00)
□	etheur_no_class_bal_logistic_reg_cv_res.csv	csv	August 10, 2021, 16:14:13 (UTC+01:00)
□	etheur_class_bal_logistic_reg_cv_conf_mx.csv	csv	August 10, 2021, 16:13:51 (UTC+01:00)
□	etheur_class_bal_logistic_reg_cv_learning.csv	csv	August 10, 2021, 16:13:51 (UTC+01:00)

3.6 Online Machine Learning:

Likewise, a script for running online learning models is developed for each alt-coin under the following paths, and as shown in following screenshot:

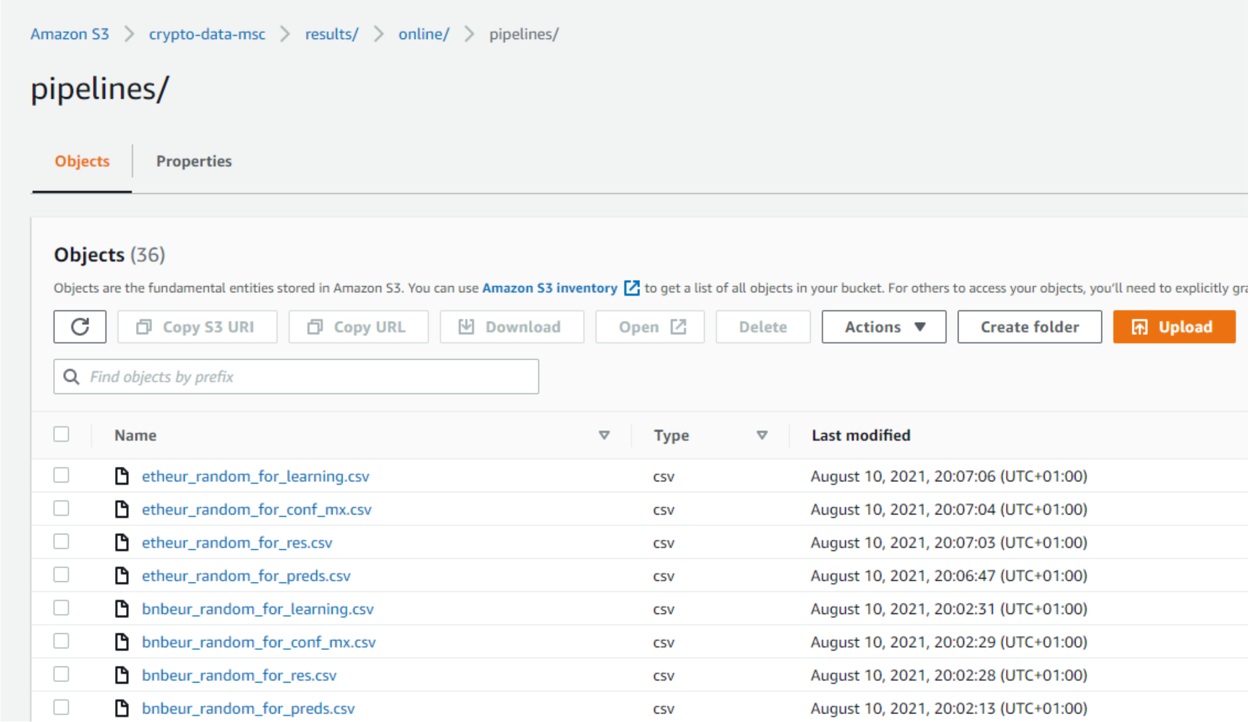
- crypto\Scripts\Prediction\Online\Classification\Production:
  - \BNB\bnb_online_pipeline.py
  - \DOGE\ doge_online_pipeline.py
  - \ETH\eth_online_pipeline.py


File Edit View Navigate Code Refactor Run Tools VCS Window Help
crypto [C:\Users\btier\PycharmProjects\crypto] - ...\Scripts\Prediction\Online\Classification\Production\BNB\bnb_online_pipel
crypto 〉 Scripts 〉 Prediction 〉 Online 〉 Classification 〉 Production 〉 BNB 〉 bnb_online_pipeline.py
Project
env.yml
config.py
bnb_online_pipeline.py
doge_online_pipeline.py
eth_online_pipeline.py
Batch
Online
Classification
MultiCrypto
Production
BNB
bnb_online_pipeline.py
bnb_prod_hoeffding_tree_o.py
bnb_prod_logistic_reg_o.py
bnb_prod_random_forest_o.py
DOGE
doge_online_pipeline.py
doge_prod_hoeffding_tree_o.py
doge_prod_logistic_reg_o.py
doge_prod_random_forest_o.py
ETH
eth_online_pipeline.py
eth_prod_get_online_data.py
eth_prod_hoeffding_tree_o.py
eth_prod_logistic_reg_o.py
eth_prod_random_forest_o.py
SingleCrypto
get_dataset_sizes.py
online_classf_pipeline.py
online_classf_pipeline2.py
Python Console
1: Project
7: Structure
2: Favorites

```
    HardSamplingClassifier(
        SoftmaxRegression(optimizer=SGD(.1)),
        p=0.2,
        size=150
    )
)

rf_pipe = Pipeline(
    HardSamplingClassifier(
        AdaptiveRandomForestClassifier(max_depth=5,
                                       split_criterion='info_gain',
                                       n_models=50),
        p=0.2,
        size=150
    )
)

pipeline_names = ["Online Hoeffding Tree Classifier", "Online Logistic Regression", "Online Random Fo
pipelines = [ht_pipe, lg_pipe, rf_pipe]
output_names1 = ["_hoeffding_tree_preds.csv", "_logistic_reg_preds.csv", "_random_for_preds.csv"]
output_names2 = ["_hoeffding_tree_res.csv", "_logistic_reg_res.csv", "_random_for_res.csv", ]
output_names3 = ["_hoeffding_tree_conf_mx.csv", "_logistic_reg_conf_mx.csv", "_random_for_conf_mx.csv
output_names4 = ["_hoeffding_tree_learning.csv", "_logistic_reg_learning.csv","_random_for_learning.c
```

```
Accuracy: 97.89%
5972 - Date: 2020-01-28 00:52:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.89%
5973 - Date: 2020-01-28 00:53:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.89%
5974 - Date: 2020-01-28 00:54:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.89%
5975 - Date: 2020-01-28 00:55:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.89%
5976 - Date: 2020-01-28 00:56:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 1 , Prediction: 0
Accuracy: 97.87%
5977 - Date: 2020-01-28 00:57:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.88%
5978 - Date: 2020-01-28 00:58:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
Accuracy: 97.88%
5979 - Date: 2020-01-28 00:59:00 - BNBEUR - Online Hoeffding Tree Classifier - Actual: 0 , Prediction: 0
```

- As can be seen in above example of BinanceCoin, the online learning is running and predicting at each time step (1-minute) and printing to console. Once it is done, the results export to AWS S3:



Objects (36)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly gra

□	Name	Type	Last modified
□	etheur_random_for_learning.csv	csv	August 10, 2021, 20:07:06 (UTC+01:00)
□	etheur_random_for_conf_mx.csv	csv	August 10, 2021, 20:07:04 (UTC+01:00)
□	etheur_random_for_res.csv	csv	August 10, 2021, 20:07:03 (UTC+01:00)
□	etheur_random_for_preds.csv	csv	August 10, 2021, 20:06:47 (UTC+01:00)
□	bnbeur_random_for_learning.csv	csv	August 10, 2021, 20:02:31 (UTC+01:00)
□	bnbeur_random_for_conf_mx.csv	csv	August 10, 2021, 20:02:29 (UTC+01:00)
□	bnbeur_random_for_res.csv	csv	August 10, 2021, 20:02:28 (UTC+01:00)
□	bnbeur_random_for_preds.csv	csv	August 10, 2021, 20:02:13 (UTC+01:00)

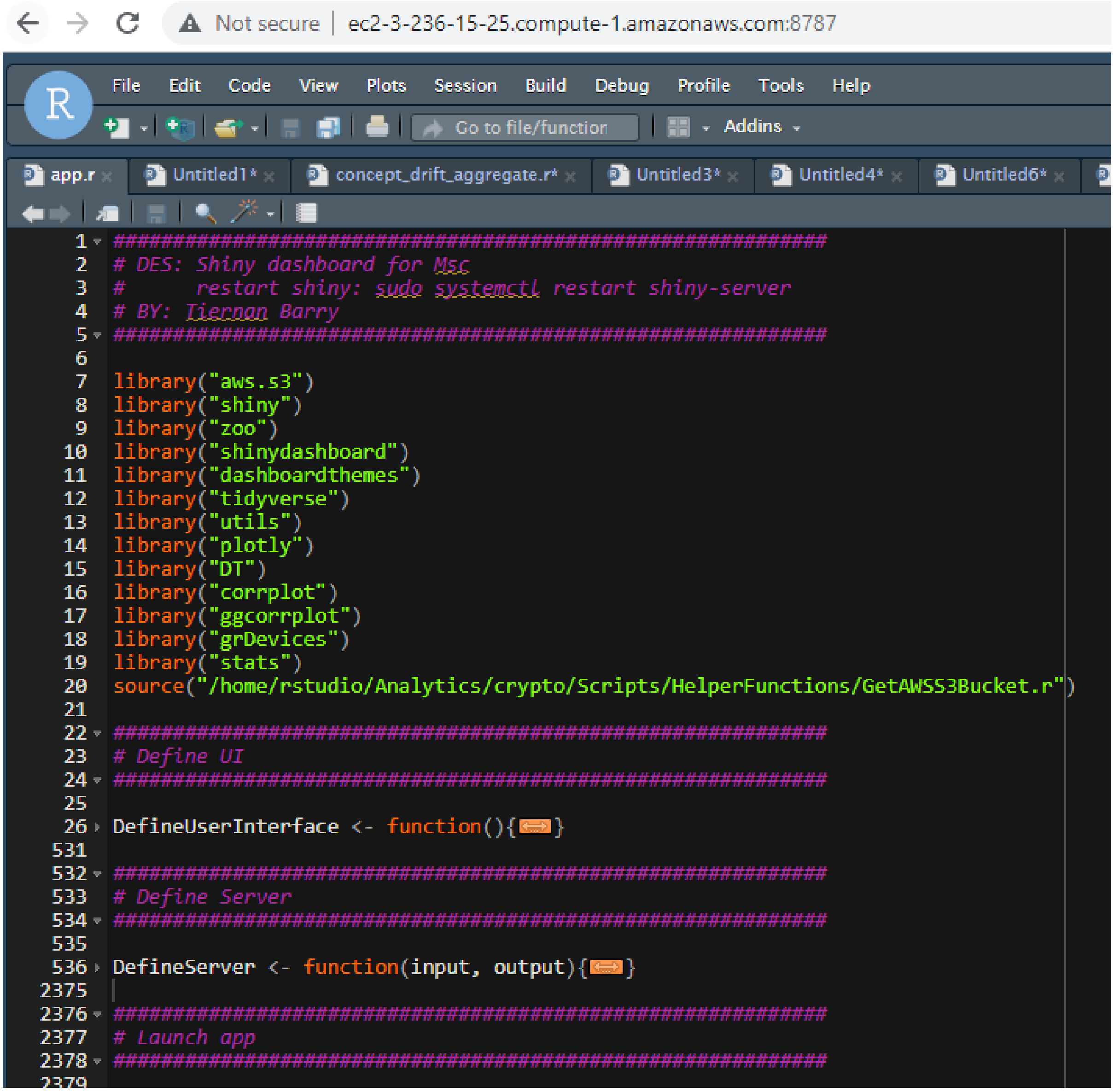
4 Visual Tier Configuration:

As discussed in the technical report, this is hosted on an Amazon EC2 cloud instance. Please note the following:

- To access the Visual Tier dashboard, you only need to open the following URL link to see the results (this is now kept running until grading is complete):

  http://ec2-3-236-15-25.compute-1.amazonaws.com:3838/crypto/

- The source code is in the below R script, and looks as follows:
  crypto/Scripts/Results/Shiny/crypto/app.r



```
##############################################################
# DES: Shiny dashboard for Msc
#       restart shiny: sudo systemctl restart shiny-server
# BY: Tiernan Barry
##############################################################

library("aws.s3")
library("shiny")
library("zoo")
library("shinydashboard")
library("dashboardthemes")
library("tidyverse")
library("utils")
library("plotly")
library("DT")
library("corrplot")
library("ggcorrplot")
library("grDevices")
library("stats")
source("/home/rstudio/Analytics/crypto/Scripts/HelperFunctions/GetAWSS3Bucket.r")

##############################################################
# Define UI
##############################################################

DefineUserInterface <- function(){↔}

##############################################################
# Define Server
##############################################################

DefineServer <- function(input, output){↔}

##############################################################
# Launch app
##############################################################
```

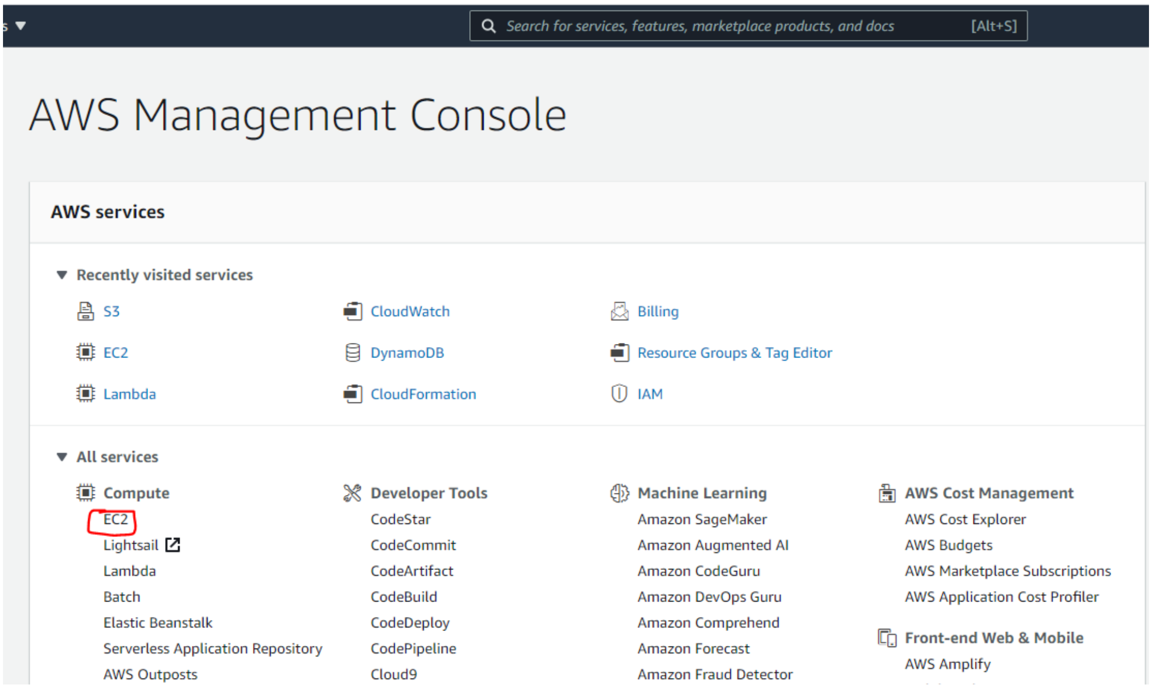
- To access the RStudio login, go to below link (password is needed, please reach out for details):

  http://ec2-3-236-15-25.compute-1.amazonaws.com:8787/

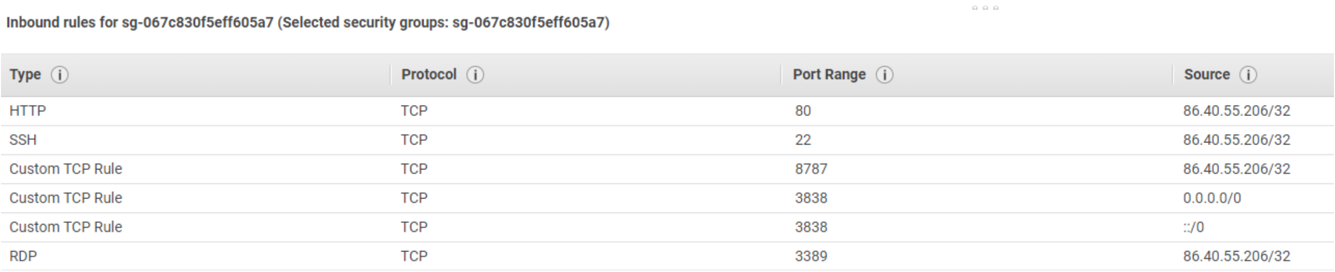
- However, for completeness, this section will also outline the required steps taken to set up an AWS EC2 instance, and also to install R and RStudio.
- **Note: if required and/or easier, please reach out for gaining user access to the exact EC2 instances used in this project, and to avoid setting everything up from scratch. These currently need my AWS credentials to get the SSH address and keys.**

4.1 Setting up and SSH into an AWS EC2 instance:

- Log into AWS management Console (or sign up first)



- Click into EC2, and click Launch Instance
- Choose AMI: Ubuntu Server 20.04 LTS (HVM), SSD Volume Type
- Instance type: t.2 large
- Next: Configure Instance (skip)
- Add storage: 16GB of Elastic Block Storage (EBS).
- Configure Security Group:
  - Create and apply new security group as follows:
  - Apply 3838 port to be available to all IP addresses (this allows anyone to open the Visual Tier dashboard)

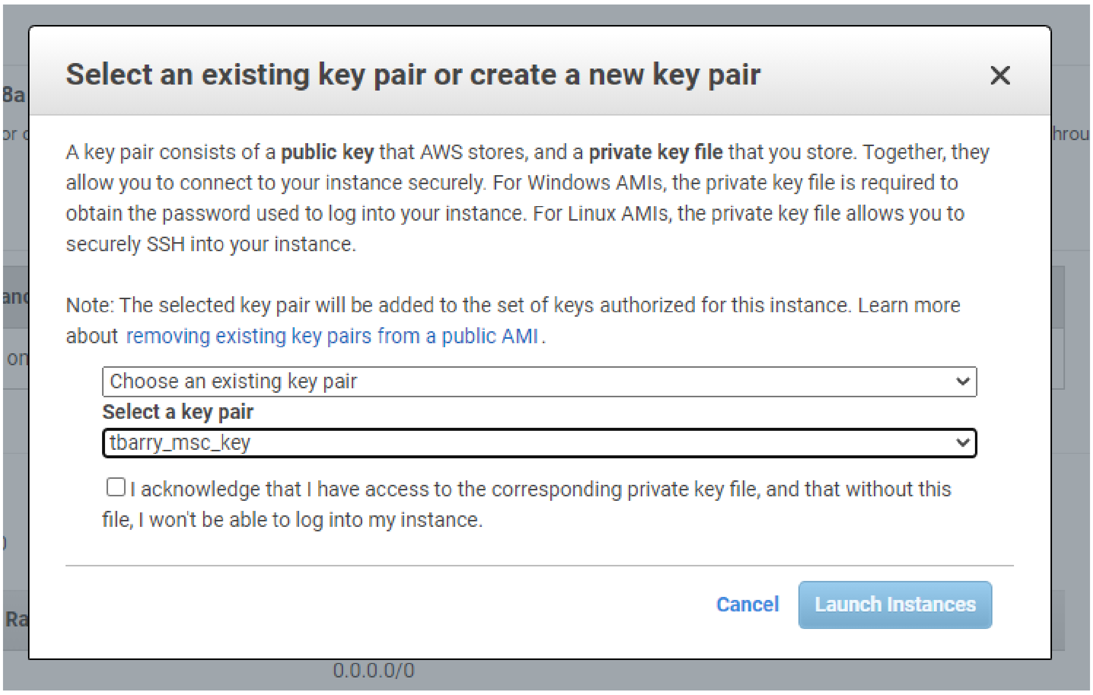


Inbound rules for sg-067c830f5eff605a7 (Selected security groups: sg-067c830f5eff605a7)

Type	Protocol	Port Range	Source
HTTP	TCP	80	86.40.55.206/32
SSH	TCP	22	86.40.55.206/32
Custom TCP Rule	TCP	8787	86.40.55.206/32
Custom TCP Rule	TCP	3838	0.0.0.0/0
Custom TCP Rule	TCP	3838	::/0
RDP	TCP	3389	86.40.55.206/32

  - I suggest allows custom IP addresses (ie. only your own) for other ports (ie RStudio, SSH, etc) as shown above.
- Review and launch
- Create new or use existing key pair:

- Ensure you save the path as we need this to SSH into EC2



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

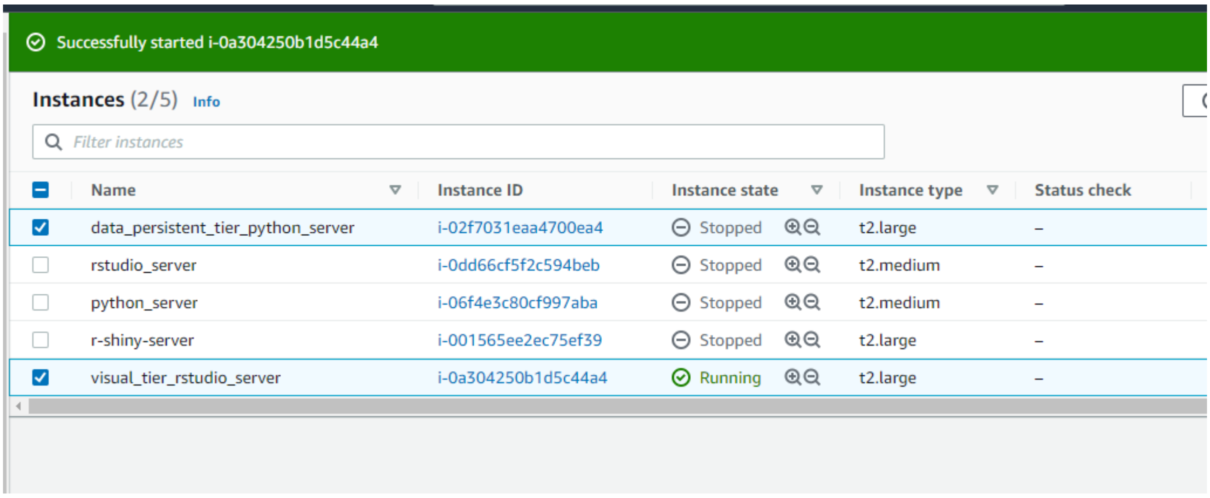
Choose an existing key pair

Select a key pair

tbarry_msc_key

□ I acknowledge that I have access to the corresponding private key file, and that without this file, I won't be able to log into my instance.

- Now, we need to SSH into the server.
- Navigate to EC2 instance console to get the IP address

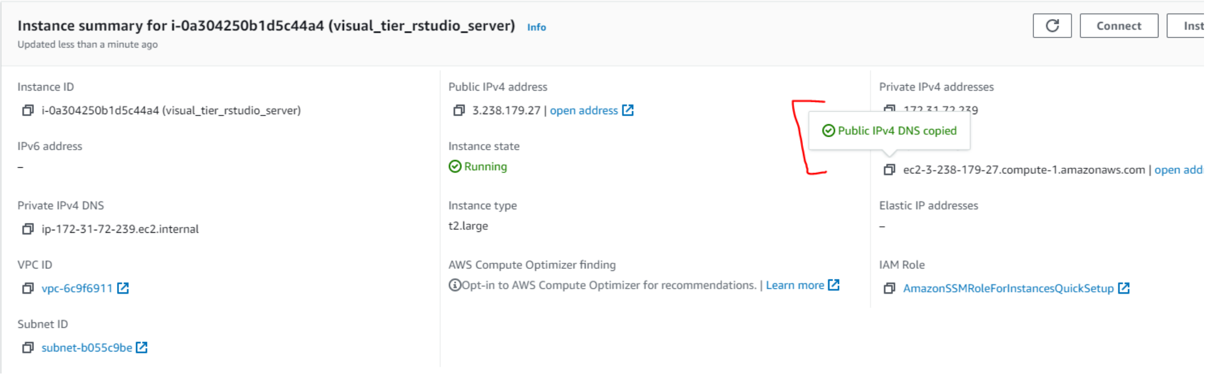


Successfully started i-0a304250b1d5c44a4

Instances (2/5) Info

☐	Name	Instance ID	Instance state	Instance type	Status check
☑	data_persistent_tier_python_server	i-02f7031eaa4700ea4	Stopped	t2.large	–
☐	rstudio_server	i-0dd66cf5f2c594beb	Stopped	t2.medium	–
☐	python_server	i-06f4e3c80cf997aba	Stopped	t2.medium	–
☐	r-shiny-server	i-001565ee2ec75ef39	Stopped	t2.large	–
☑	visual_tier_rstudio_server	i-0a304250b1d5c44a4	Running	t2.large	–

- By clicking into the instance ID of the visual tier, we can now see the following, and can copy the IP address which has just been generated:



Instance summary for i-0a304250b1d5c44a4 (visual_tier_rstudio_server)

Updated less than a minute ago

Instance ID
i-0a304250b1d5c44a4 (visual_tier_rstudio_server)

IPv6 address
–

Private IPv4 DNS
ip-172-31-72-239.ec2.internal

VPC ID
vpc-6c9f6911

Subnet ID
subnet-b055c9be

Public IPv4 address
3.238.179.27 | open address

Instance state
Running

Instance type
t2.large

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Private IPv4 addresses
172.31.72.239

Public IPv4 DNS copied

ec2-3-238-179-27.compute-1.amazonaws.com | open add

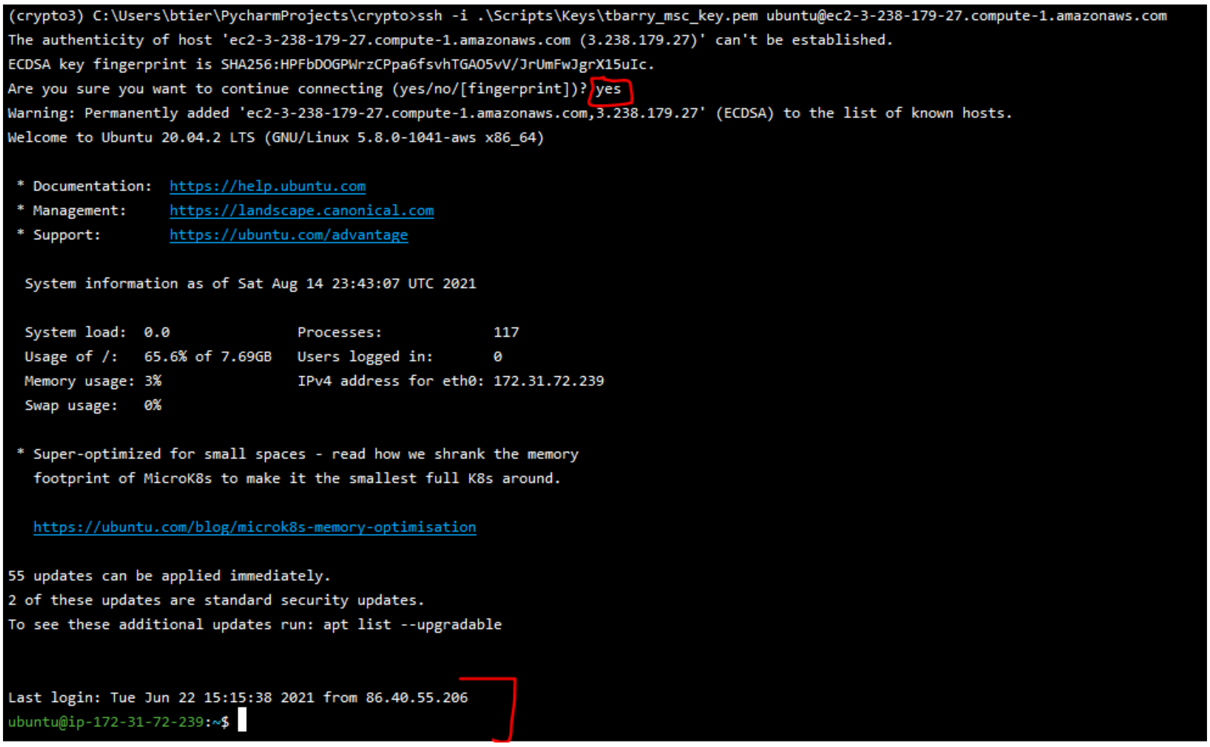
Elastic IP addresses
–

IAM Role
AmazonSSMRoleForInstancesQuickSetup

- With the IP address and the key pair saved, we can now SSH into the EC2 instance using a command as below:

ssh -i .\Scripts\Keys\tbarry_msc_key.pem ubuntu@ec2-3-236-15-25.compute-1.amazonaws.com

- Click yes when prompted, and you will be logged in as follows:



```
(crypto3) C:\Users\btier\PycharmProjects\crypto>ssh -i .\Scripts\Keys\tbarry_msc_key.pem ubuntu@ec2-3-238-179-27.compute-1.amazonaws.com
The authenticity of host 'ec2-3-238-179-27.compute-1.amazonaws.com (3.238.179.27)' can't be established.
ECDSA key fingerprint is SHA256:HPFbDOGPWrzCPpa6fsvhTGAO5vV/JrUmFwJgrX15uIc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-238-179-27.compute-1.amazonaws.com,3.238.179.27' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-1041-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Aug 14 23:43:07 UTC 2021

  System load:  0.0               Processes:             117
  Usage of /:   65.6% of 7.69GB   Users logged in:       0
  Memory usage: 3%                IPv4 address for eth0: 172.31.72.239
  Swap usage:   0%

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

55 updates can be applied immediately.
2 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable


Last login: Tue Jun 22 15:15:38 2021 from 86.40.55.206
ubuntu@ip-172-31-72-239:~$
```

4.2 Installing R and RStudio onto Ubuntu AWS EC2

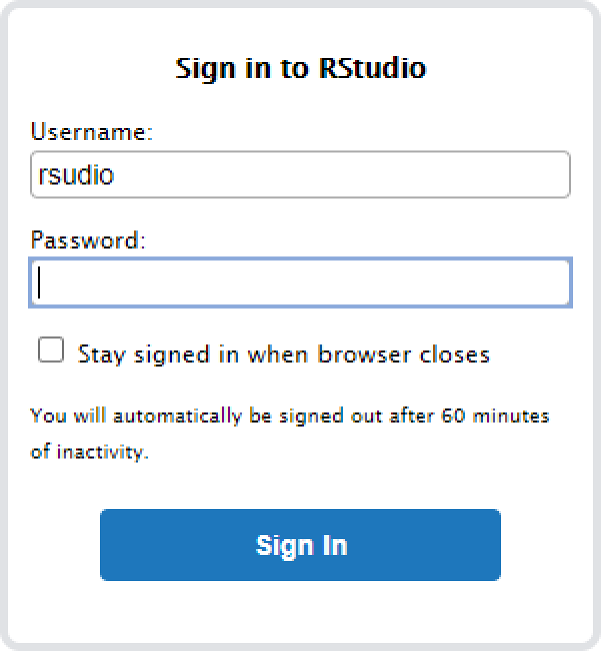
Following this very useful guide, R and RStudio can be installed (Zwitch, 2013). A few slight adjustments are made to ensure latest versions are available (crypto\Docs\rserver_install.txt):

```
# notes for R studio install
# source: https://www.r-bloggers.com/2013/04/instructions-for-installing-using-r-on-amazon-ec2/

sudo useradd rstudio
sudo mkdir /home/rstudio
sudo passwd rstudio
sudo chmod -R 0777 /home/rstudio

#Update all files from the default state
sudo apt-get update
sudo apt-get upgrade

#Add CRAN mirror to custom sources.list file using vi
sudo vi /etc/apt/sources.list.d/sources.list

#Add following line (or your favorite CRAN mirror)
deb http://lib.stat.cmu.edu/R/CRAN/bin/linux/ubuntu precise/

#Update files to use CRAN mirror
#Don't worry about error message
sudo apt-get update

#Install latest version of R
#Install without verification
sudo apt-get install r-base

#Install in order to use RCurl & XML
sudo apt-get install libcurl4-openssl-dev
sudo apt-get install libxml2-dev

#Install a few background files
sudo apt-get install gdebi-core
sudo apt-get install libapparmor1

#Change to a writeable directory
#Download & Install RStudio Server
cd /tmp



wget https://download2.rstudio.org/server/bionic/amd64/rstudio-server-1.4.1106-amd64.deb
sudo gdebi rstudio-server-1.4.1106-amd64.deb

# run in terminal for packages
# https://stackoverflow.com/questions/55855898/installing-aws-s3-r-package
sudo apt-get install -y build-essential libssl-dev libxml2-dev libcurl4-openssl-dev
```

```
# restart shiny service
sudo systemctl restart shiny-server
```

- Once RStudio is installed, we can log into RStudio through a web browser using the IP address, followed by the RStudio port of 8787.

http://ec2-3-236-15-25.compute-1.amazonaws.com:8787/



Sign in to RStudio

Username:
rsudio

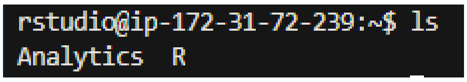
Password:

□ Stay signed in when browser closes

You will automatically be signed out after 60 minutes of inactivity.

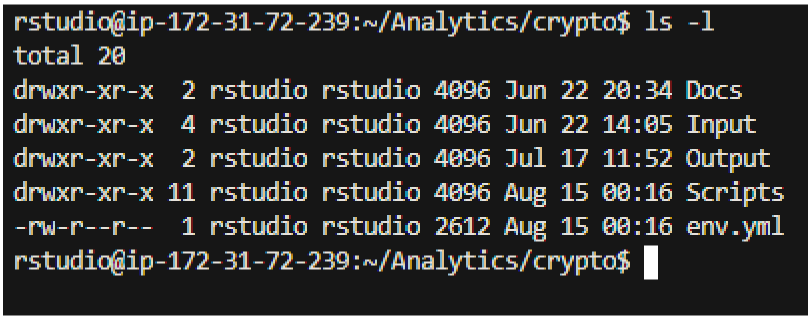
4.3 Get code onto server (clone from Github)

- Code was managed through Github in a private repo. Please request access to repository if needed to clone the code into the Visual Tier.
- Once logged into RStudio, you can use the terminal plugin on the bottom left to run linux commands.
- Create an Analytics folder:
  cd ~
  mkdir Analytics



```
rstudio@ip-172-31-72-239:~$ ls
Analytics  R
```

- Clone Github repository (access needs to be granted)



```
rstudio@ip-172-31-72-239:~/Analytics/crypto$ ls -l
total 20
drwxr-xr-x  2 rstudio rstudio 4096 Jun 22 20:34 Docs
drwxr-xr-x  4 rstudio rstudio 4096 Jun 22 14:05 Input
drwxr-xr-x  2 rstudio rstudio 4096 Jul 17 11:52 Output
drwxr-xr-x 11 rstudio rstudio 4096 Aug 15 00:16 Scripts
-rw-r--r--  1 rstudio rstudio 2612 Aug 15 00:16 env.yml
rstudio@ip-172-31-72-239:~/Analytics/crypto$
```

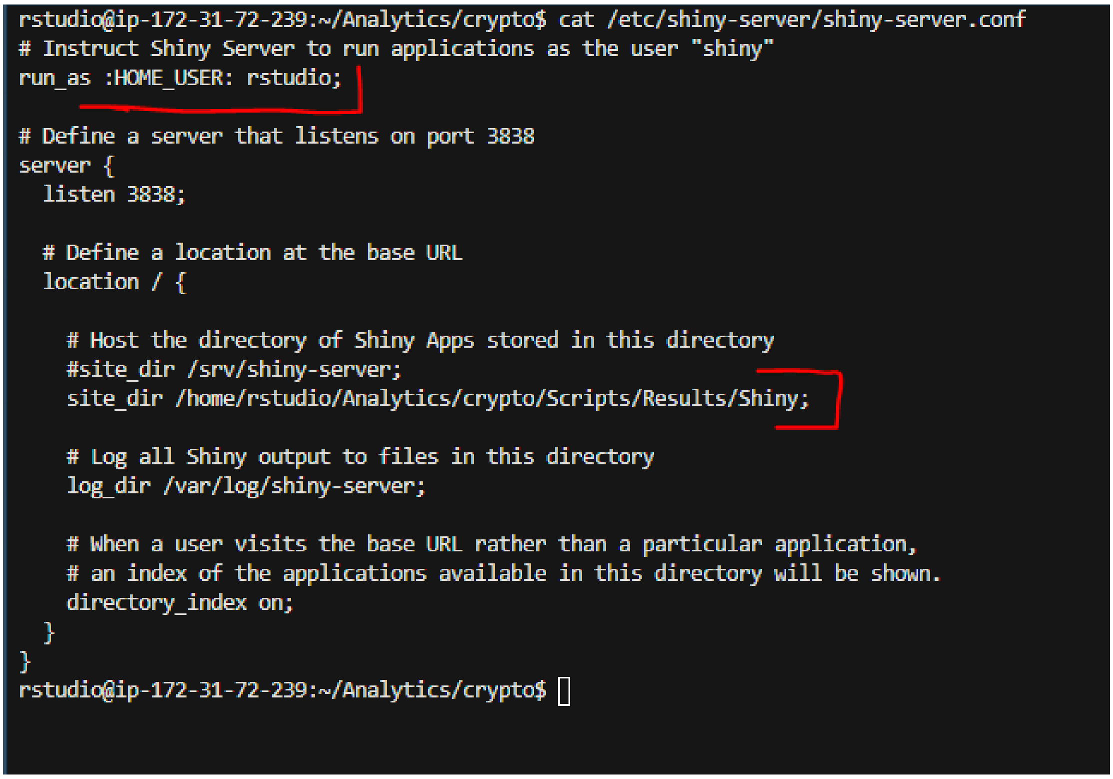
4.4 Installing and Configuring Shiny service

The Shiny service needs to be configured to enable Shiny apps to be published to the web. This is done by configuring the following file as follows:

- /etc/shiny-server/shiny-server.conf
- Change the default settings to the ones provided below (run as, site dir).

```
# Instruct Shiny Server to run applications as the user "shiny"
run_as :HOME_USER: rstudio;

# Define a server that listens on port 3838
server {
  listen 3838;

  # Define a location at the base URL
  location / {

    # Host the directory of Shiny Apps stored in this directory
    #site_dir /srv/shiny-server;
    site_dir /home/rstudio/Analytics/crypto/Scripts/Results/Shiny;

    # Log all Shiny output to files in this directory
    log_dir /var/log/shiny-server;

    # When a user visits the base URL rather than a particular application,
    # an index of the applications available in this directory will be shown.
    directory_index on;
  }
}
```

- Save and ensure it is correct in server:



```
rstudio@ip-172-31-72-239:~/Analytics/crypto$ cat /etc/shiny-server/shiny-server.conf
# Instruct Shiny Server to run applications as the user "shiny"
run_as :HOME_USER: rstudio;

# Define a server that listens on port 3838
server {
  listen 3838;

  # Define a location at the base URL
  location / {

    # Host the directory of Shiny Apps stored in this directory
    #site_dir /srv/shiny-server;
    site_dir /home/rstudio/Analytics/crypto/Scripts/Results/Shiny;

    # Log all Shiny output to files in this directory
    log_dir /var/log/shiny-server;

    # When a user visits the base URL rather than a particular application,
    # an index of the applications available in this directory will be shown.
    directory_index on;
  }
}
rstudio@ip-172-31-72-239:~/Analytics/crypto$ 
```

4.5 Install R Packages:

Once logged into an R session, the following packages must be installed first in the R console:

```
install.packages('curl')
install.packages('httr')
install.packages('xml2')
```

Then, the following packages are needed to be installed (using the above command in R) launch the web-app:

install.packages("aws.s3")
install.packages("shiny") # install shiny R package now
install.packages("zoo")
install.packages("shinydashboard")
install.packages("dashboardthemes")
install.packages("tidyverse")
library("utils") # should be installed already – can skip this
install.packages("plotly")
install.packages("DT")
install.packages("corrplot")
install.packages("ggcorrplot")
install.packages("grDevices")

library("stats") # should be installed already – can skip this

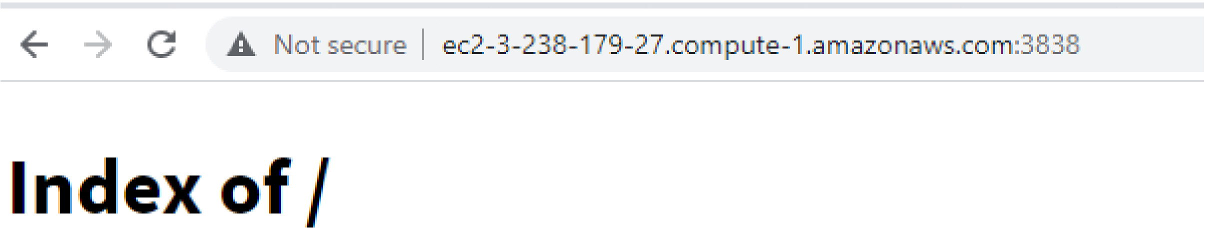
4.6 Launch Visual Tier Dashboard (Shiny App):

- First, from the command line, restart the Shiny Service:

  sudo systemctl restart shiny-server

- Now, we can navigate to the Web App from a browser by changing the port to 3838 as follows:

http://ec2-3-238-179-27.compute-1.amazonaws.com:3838/



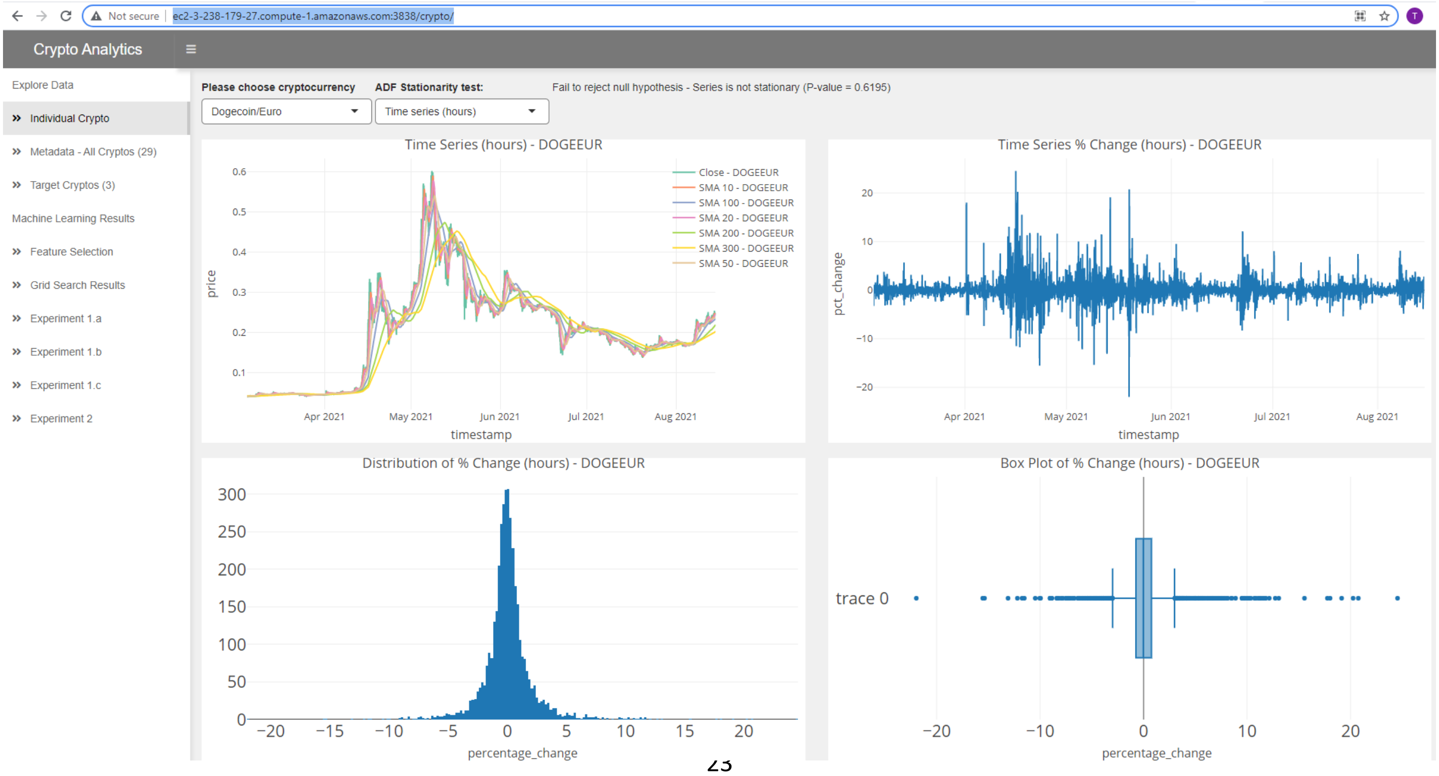
Not secure | ec2-3-238-179-27.compute-1.amazonaws.com:3838

Index of /

- crypto/

- Now, we can click on ‘crypto’ to open the app and complete the URL link:

http://ec2-3-238-179-27.compute-1.amazonaws.com:3838/crypto/



Not secure | ec2-3-238-179-27.compute-1.amazonaws.com:3838/crypto/

Please choose cryptocurrency: Dogecoin/Euro

ADF Stationarity test: Time series (hours)

Fail to reject null hypothesis - Series is not stationary (P-value = 0.6195)

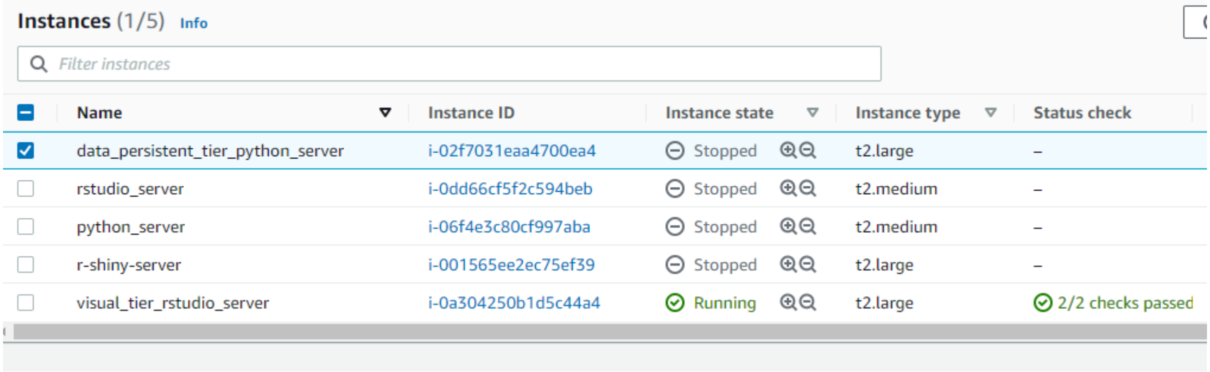
5 Data Persistent Tier Configuration:

Likewise, this is also hosted on an Amazon EC2 cloud instance, and combines a number of AWS Services as discussed here. Similar to the Visual Tier, please reach out if access credentials are required for logging into the exact instance, or AWS account (due to the large number of steps, etc). The following AWS Services need to be configured (useful Youtube video here: (TotalCloud, 2020)):

- AWS EC2
- AWS CloudFormation
- AWS Dynamo DB
- Apply Tag to AWS EC2 instance

5.1 Create a new EC2 Ubuntu Instance:

- Follow same steps as 4.1.
- Log into the EC2 instance



Instances (1/5) Info

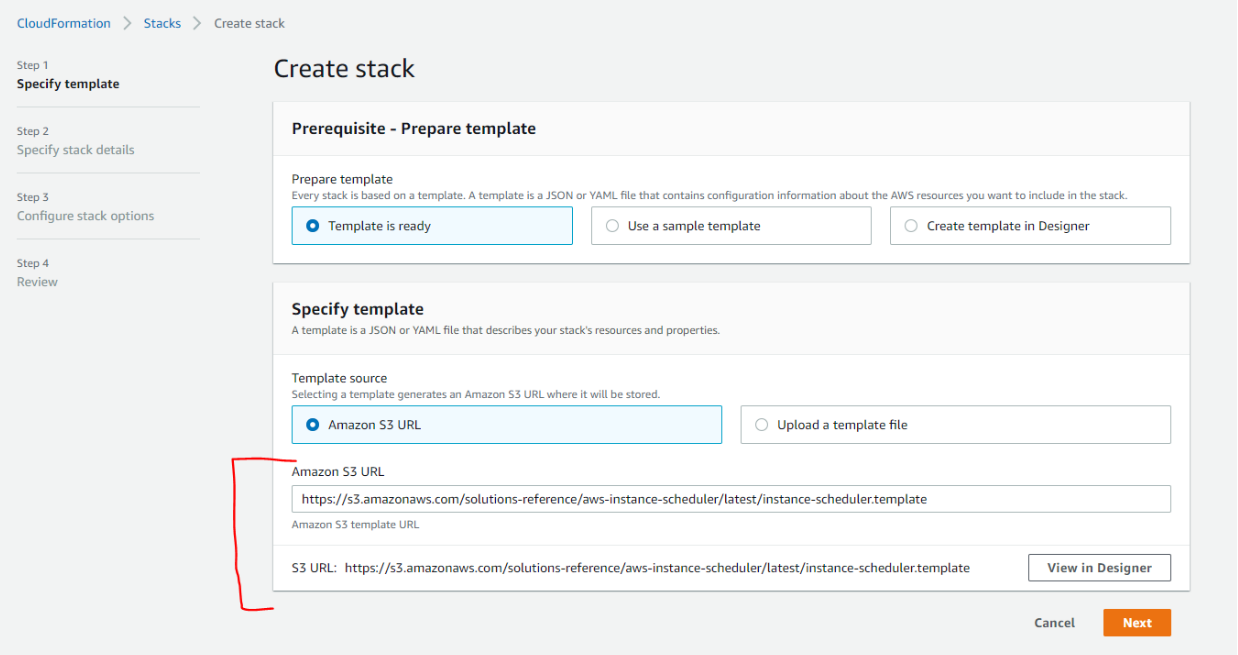
Filter instances

☐	Name	Instance ID	Instance state	Instance type	Status check
☑	data_persistent_tier_python_server	i-02f7031eaa4700ea4	⊖ Stopped	t2.large	–
☐	rstudio_server	i-0dd66cf5f2c594beb	⊖ Stopped	t2.medium	–
☐	python_server	i-06f4e3c80cf997aba	⊖ Stopped	t2.medium	–
☐	r-shiny-server	i-001565ee2ec75ef39	⊖ Stopped	t2.large	–
☐	visual_tier_rstudio_server	i-0a304250b1d5c44a4	✓ Running	t2.large	✓ 2/2 checks passed

5.2 Create CloudFormation template:

- Create Stack
- Using the below S3 URL, load the instance scheduler template:

https://s3.amazonaws.com/solutions-reference/aws-instance-scheduler/latest/instance-scheduler.template


CloudFormation > Stacks > Create stack

Step 1
Specify template

Step 2
Specify stack details

Step 3
Configure stack options

Step 4
Review

Create stack

Prerequisite - Prepare template

Prepare template
Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

● Template is ready ○ Use a sample template ○ Create template in Designer

Specify template
A template is a JSON or YAML file that describes your stack's resources and properties.

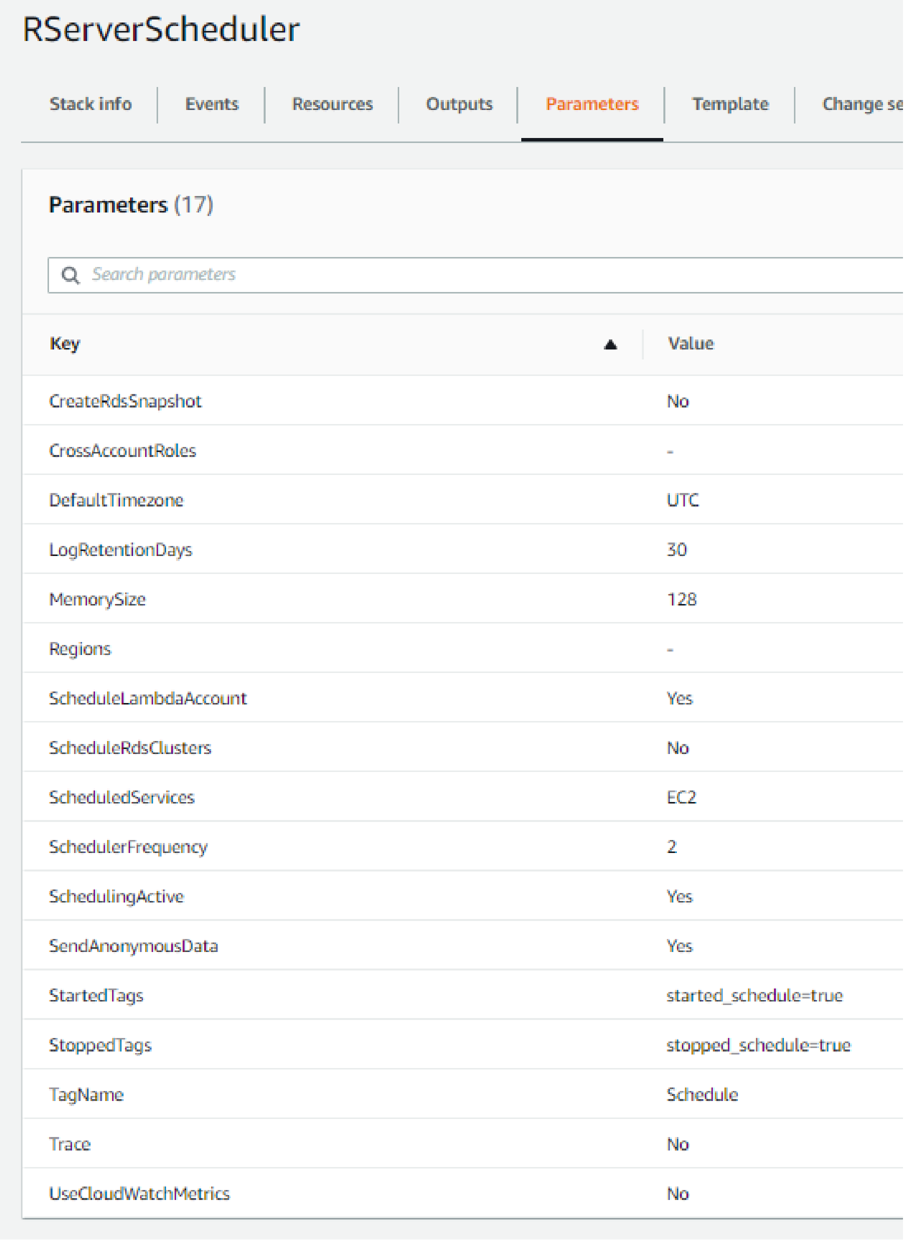
Template source
Selecting a template generates an Amazon S3 URL where it will be stored.

● Amazon S3 URL ○ Upload a template file

Amazon S3 URL
https://s3.amazonaws.com/solutions-reference/aws-instance-scheduler/latest/instance-scheduler.template
Amazon S3 template URL

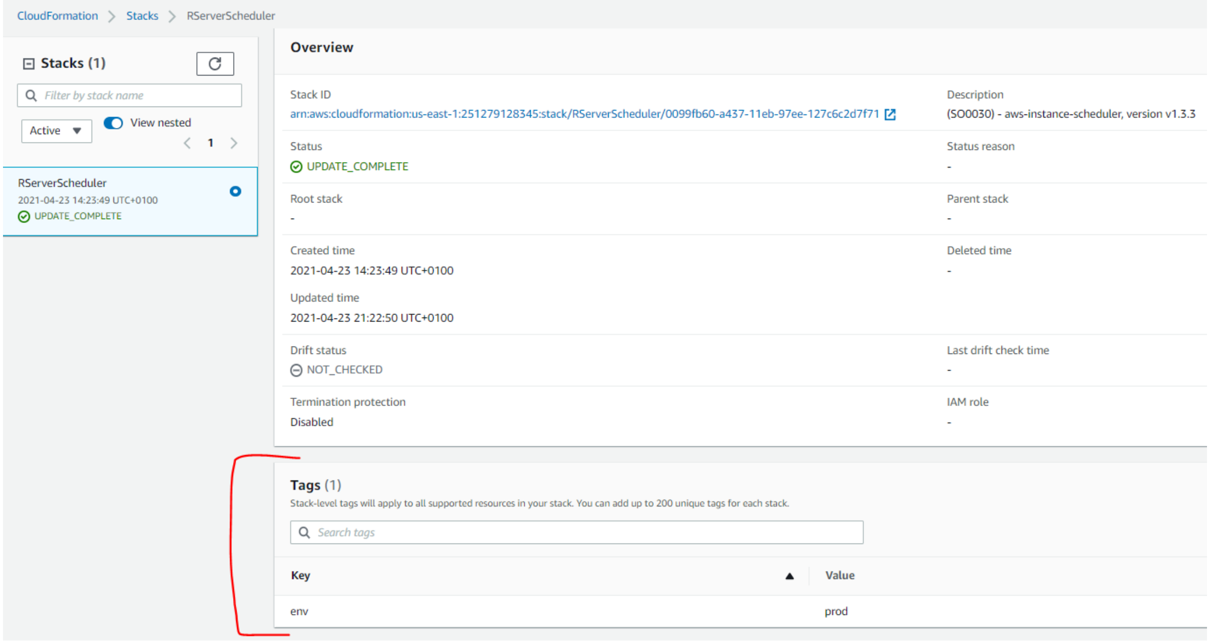
S3 URL: https://s3.amazonaws.com/solutions-reference/aws-instance-scheduler/latest/instance-scheduler.template

- Apply a stack name, RServerScheduler is used in this case
- The following parameters are applied when setting up:


RServerScheduler

Parameters (17)

Key	Value
CreateRdsSnapshot	No
CrossAccountRoles	-
DefaultTimezone	UTC
LogRetentionDays	30
MemorySize	128
Regions	-
ScheduleLambdaAccount	Yes
ScheduleRdsClusters	No
ScheduledServices	EC2
SchedulerFrequency	2
SchedulingActive	Yes
SendAnonymousData	Yes
StartedTags	started_schedule=true
StoppedTags	stopped_schedule=true
TagName	Schedule
Trace	No
UseCloudWatchMetrics	No



CloudFormation > Stacks > RServerScheduler

Stacks (1)

Filter by stack name

Active ▼ View nested

< 1 >

RServerScheduler
2021-04-23 14:23:49 UTC+0100
UPDATE_COMPLETE

Overview

Stack ID
arn:aws:cloudformation:us-east-1:251279128345:stack/RServerScheduler/0099fb60-a437-11eb-97ee-127c6c2d7f71

Description
(SO0030) - aws-instance-scheduler, version v1.3.3

Status
UPDATE_COMPLETE

Status reason
-

Root stack
-

Parent stack
-

Created time
2021-04-23 14:23:49 UTC+0100

Deleted time
-

Updated time
2021-04-23 21:22:50 UTC+0100

Drift status
NOT_CHECKED

Last drift check time
-

Termination protection
Disabled

IAM role
-

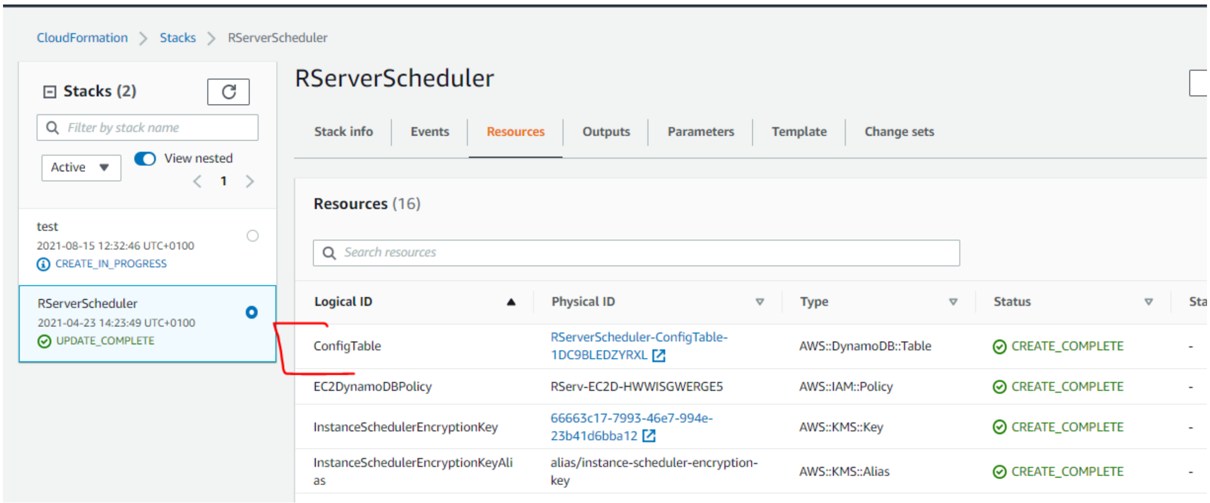
Tags (1)

Stack-level tags will apply to all supported resources in your stack. You can add up to 200 unique tags for each stack.

Search tags

Key ▲	Value
env	prod

- Once Stack is created, open the following config table:



CloudFormation > Stacks > RServerScheduler

Stacks (2)

Filter by stack name

Active ▼ View nested

< 1 >

test
2021-08-15 12:32:46 UTC+0100
CREATE_IN_PROGRESS

RServerScheduler
2021-04-23 14:23:49 UTC+0100
UPDATE_COMPLETE

RServerScheduler

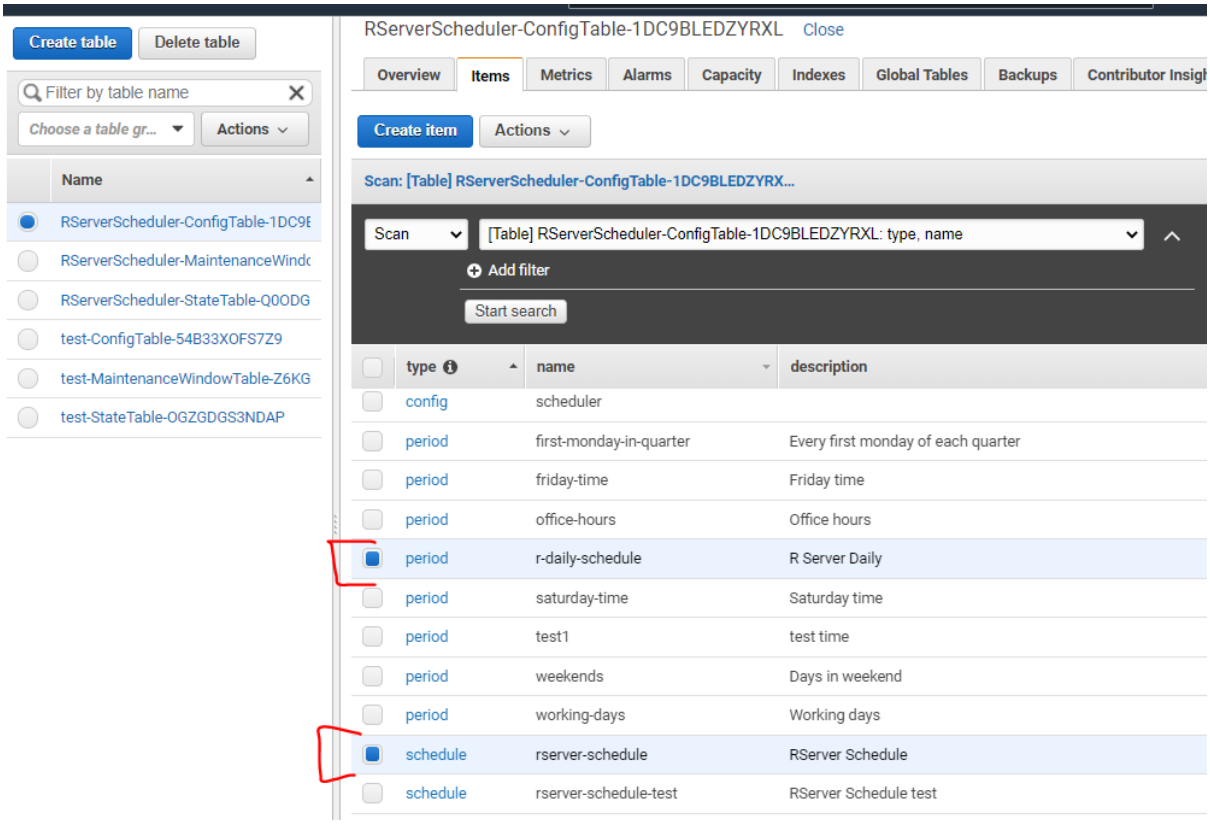
Resources (16)

Search resources

Logical ID ▲	Physical ID	Type	Status	Sta
ConfigTable	RServerScheduler-ConfigTable-1DC9BLEDZYRXL	AWS::DynamoDB::Table	CREATE_COMPLETE	-
EC2DynamoDBPolicy	RServ-EC2D-HWWISGWERGE5	AWS::IAM::Policy	CREATE_COMPLETE	-
InstanceSchedulerEncryptionKey	66663c17-7993-46e7-994e-23b41d6bba12	AWS::KMS::Key	CREATE_COMPLETE	-
InstanceSchedulerEncryptionKeyAlias	alias/instance-scheduler-encryption-key	AWS::KMS::Alias	CREATE_COMPLETE	-

5.3 Create Schedule in Dynamo DB:

- Open the following page:



Name
● RServerScheduler-ConfigTable-1DC9E
○ RServerScheduler-MaintenanceWindc
○ RServerScheduler-StateTable-Q0ODG
○ test-ConfigTable-54B33XOFS7Z9
○ test-MaintenanceWindowTable-Z6KG
○ test-StateTable-OGZGDGS3NDAP

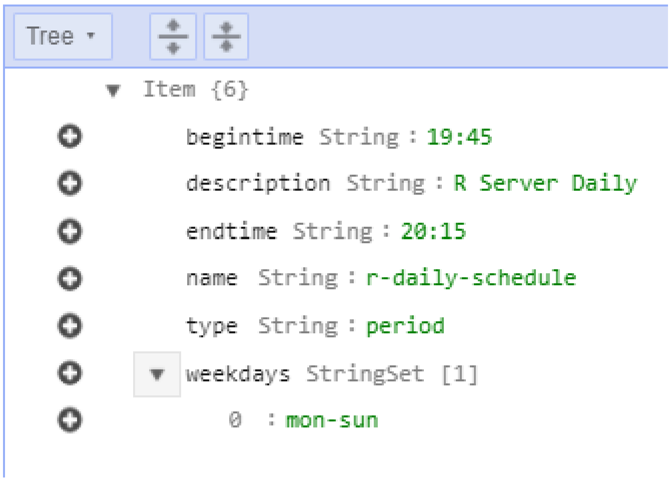
RServerScheduler-ConfigTable-1DC9BLEDZYRXL

Scan: [Table] RServerScheduler-ConfigTable-1DC9BLEDZYRX...

Scan [Table] RServerScheduler-ConfigTable-1DC9BLEDZYRXL: type, name

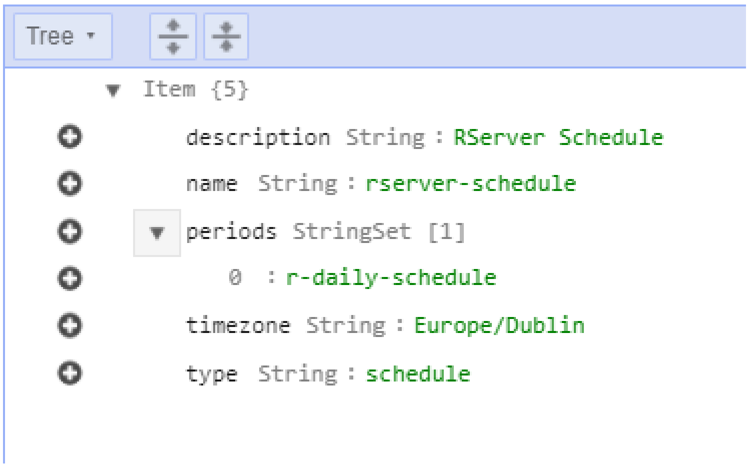
☐	type	name	description
☐	config	scheduler	
☐	period	first-monday-in-quarter	Every first monday of each quarter
☐	period	friday-time	Friday time
☐	period	office-hours	Office hours
☑	period	r-daily-schedule	R Server Daily
☐	period	saturday-time	Saturday time
☐	period	test1	test time
☐	period	weekends	Days in weekend
☐	period	working-days	Working days
☑	schedule	rserver-schedule	RServer Schedule
☐	schedule	rserver-schedule-test	RServer Schedule test

- Here, we need to create the above files (use/edit example templates provided):
  - Period:



```
▼ Item {6}
    begintime String : 19:45
    description String : R Server Daily
    endtime String : 20:15
    name String : r-daily-schedule
    type String : period
  ▼ weekdays StringSet [1]
        0 : mon-sun
```

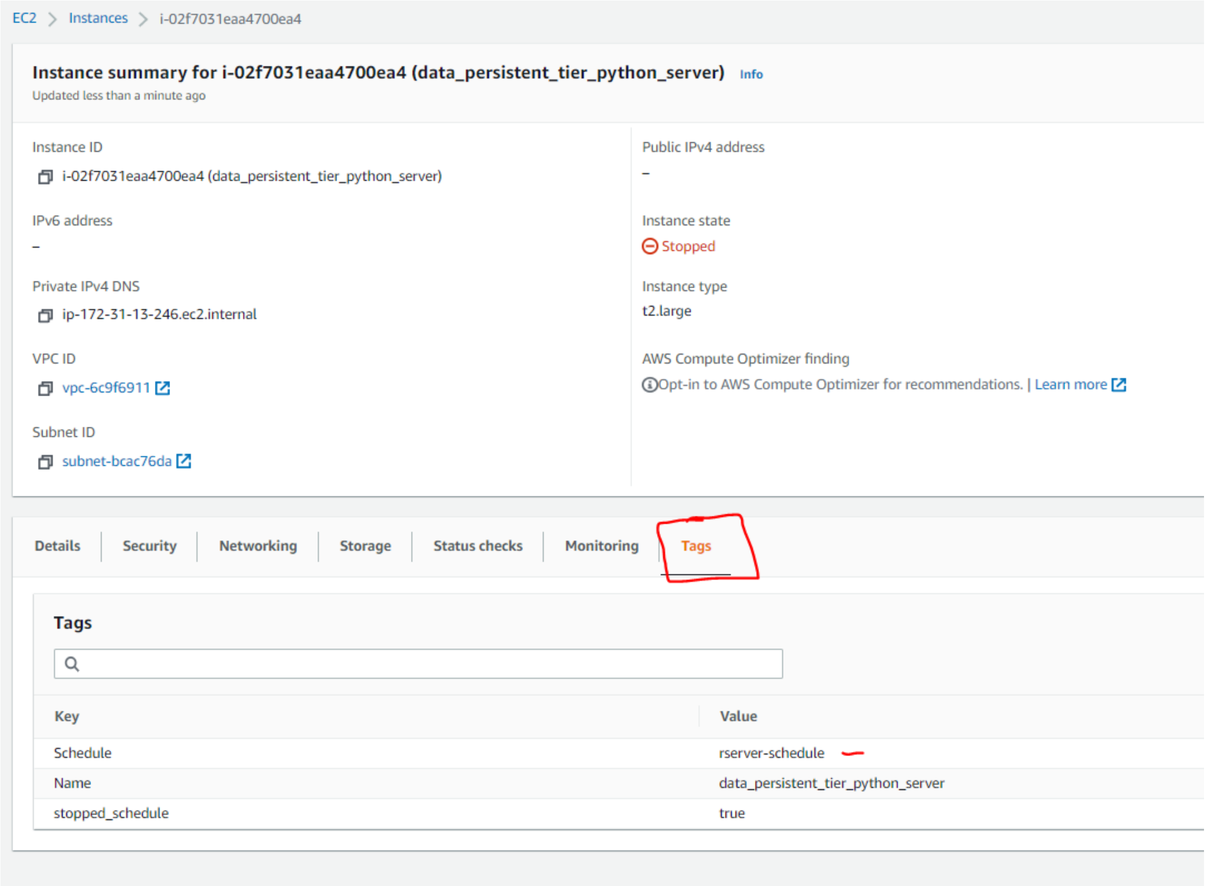
  - Schedule:



```
▼ Item {5}
    description String : RServer Schedule
    name String : rserver-schedule
  ▼ periods StringSet [1]
      0 : r-daily-schedule
    timezone String : Europe/Dublin
    type String : schedule
```

5.4 Apply Instance Scheduling tag to EC2 instance:

- Go to EC2 dashboard and open the EC2 tags as follows:



Instance summary for i-02f7031eaa4700ea4 (data_persistent_tier_python_server) Info

Updated less than a minute ago

Instance ID
i-02f7031eaa4700ea4 (data_persistent_tier_python_server)

Public IPv4 address
–

IPv6 address
–

Instance state
Stopped

Private IPv4 DNS
ip-172-31-13-246.ec2.internal

Instance type
t2.large

VPC ID
vpc-6c9f6911

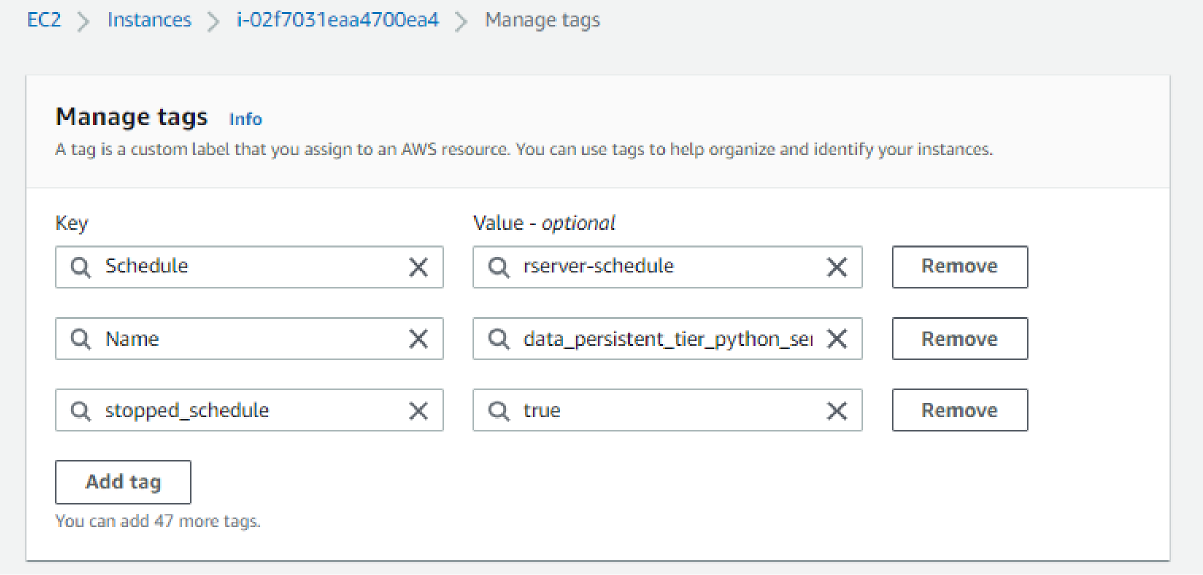
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Subnet ID
subnet-bcac76da

Tags

Key	Value
Schedule	rserver-schedule
Name	data_persistent_tier_python_server
stopped_schedule	true

- Apply the Key/Value of the scheduler which was created in previous steps:



EC2 > Instances > i-02f7031eaa4700ea4 > Manage tags

Manage tags Info

A tag is a custom label that you assign to an AWS resource. You can use tags to help organize and identify your instances.

Key	Value - *optional*	
Schedule	rserver-schedule	Remove
Name	data_persistent_tier_python_sei	Remove
stopped_schedule	true	Remove

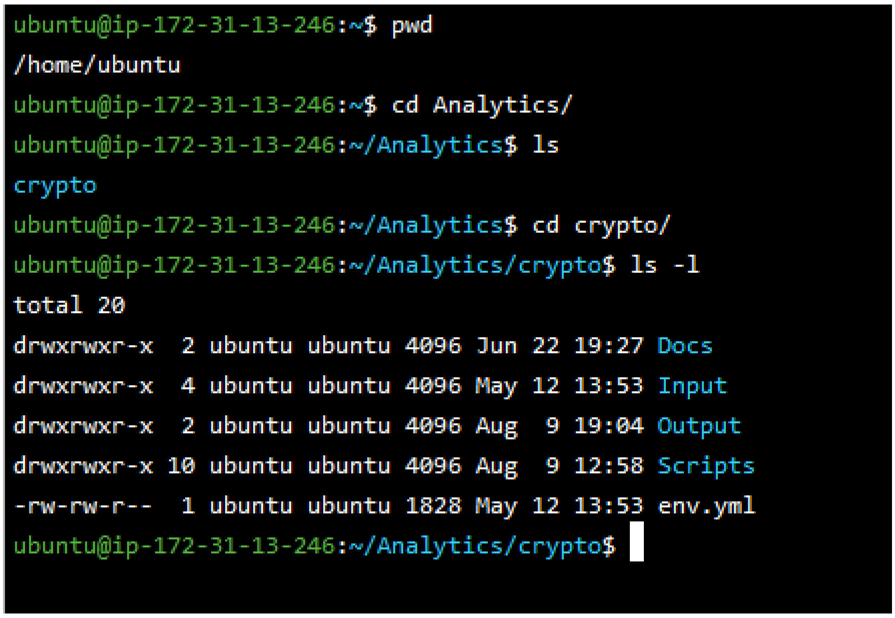
Add tag

You can add 47 more tags.

- This is now scheduled to launch at the defined times.
- More information on Aws instance scheduling can be found here (Amazon, 2020)

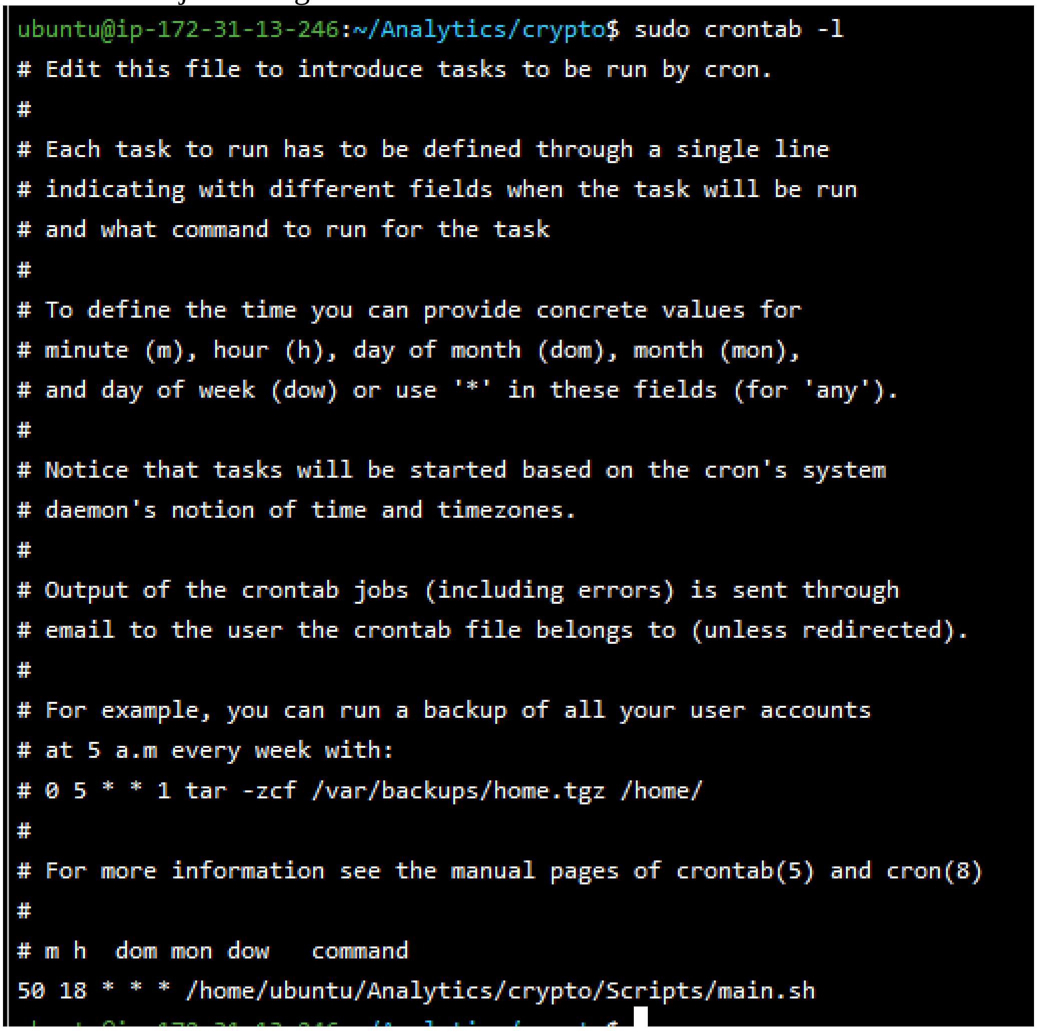
5.5 Cron-Job configuration:

- Similar to the Visual tier, the code is managed using Github to replicate the code.
- Once the code is available in the Data Persistent Tier under the Analytics directory, we then schedule a shell script to run using cron.



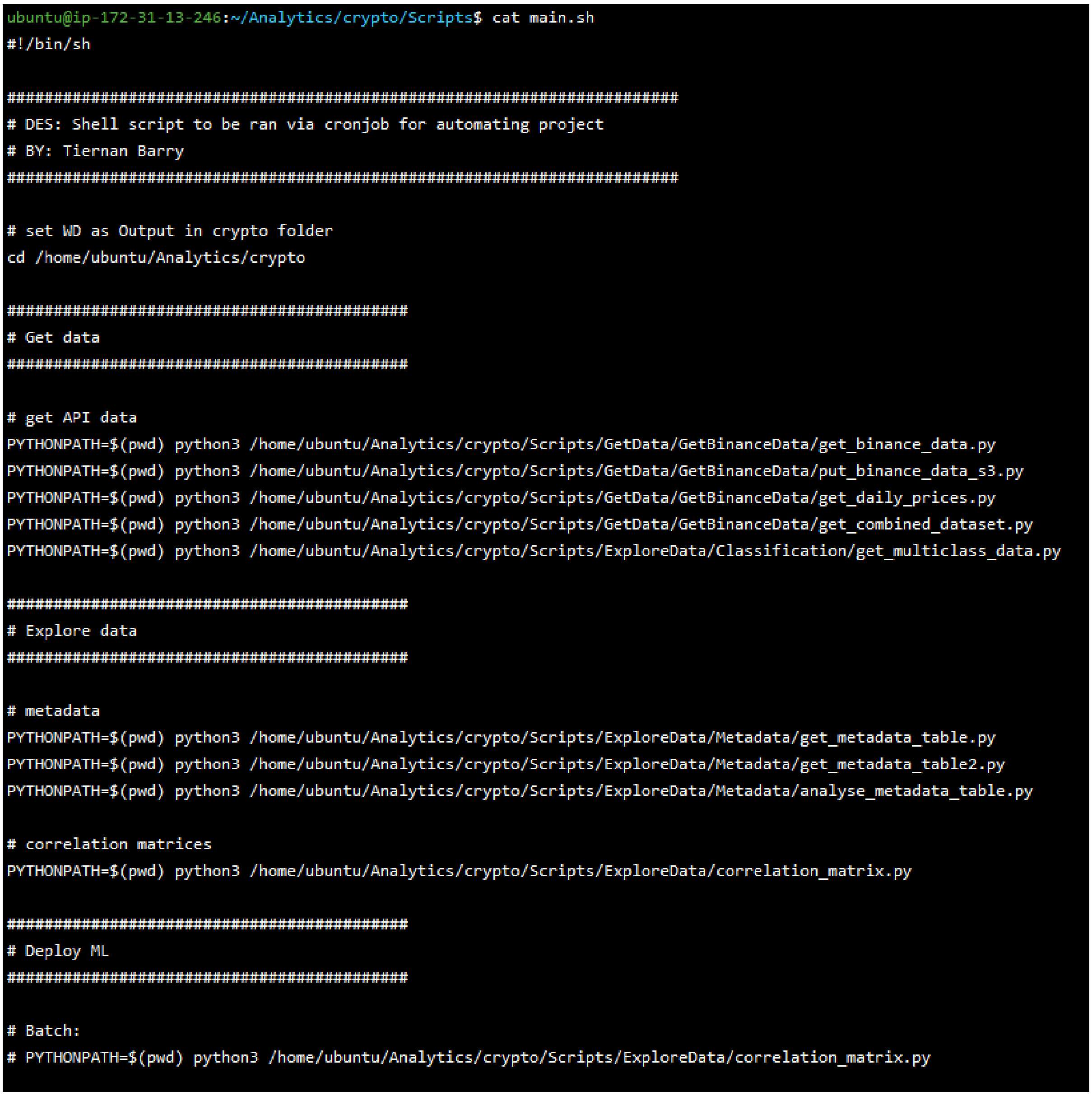
```
ubuntu@ip-172-31-13-246:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-13-246:~$ cd Analytics/
ubuntu@ip-172-31-13-246:~/Analytics$ ls
crypto
ubuntu@ip-172-31-13-246:~/Analytics$ cd crypto/
ubuntu@ip-172-31-13-246:~/Analytics/crypto$ ls -l
total 20
drwxrwxr-x  2 ubuntu ubuntu 4096 Jun 22 19:27 Docs
drwxrwxr-x  4 ubuntu ubuntu 4096 May 12 13:53 Input
drwxrwxr-x  2 ubuntu ubuntu 4096 Aug  9 19:04 Output
drwxrwxr-x 10 ubuntu ubuntu 4096 Aug  9 12:58 Scripts
-rw-rw-r--  1 ubuntu ubuntu 1828 May 12 13:53 env.yml
ubuntu@ip-172-31-13-246:~/Analytics/crypto$
```

- Cron job config:



```
ubuntu@ip-172-31-13-246:~/Analytics/crypto$ sudo crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
50 18 * * * /home/ubuntu/Analytics/crypto/Scripts/main.sh
```

- As seen above, there is 1 shell script which runs at 18:50 system time.
- Therefore, once the instance is launched automatically using AWS instnace scheduling, 5 minutes later the above cron job kicks off and collects and loads latest data to S3.
- The shell script is as follows, and is provided in the code:



```
ubuntu@ip-172-31-13-246:~/Analytics/crypto/Scripts$ cat main.sh
#!/bin/sh

########################################################################
# DES: Shell script to be ran via cronjob for automating project
# BY: Tiernan Barry
########################################################################

# set WD as Output in crypto folder
cd /home/ubuntu/Analytics/crypto

###########################################
# Get data
###########################################

# get API data
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/GetData/GetBinanceData/get_binance_data.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/GetData/GetBinanceData/put_binance_data_s3.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/GetData/GetBinanceData/get_daily_prices.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/GetData/GetBinanceData/get_combined_dataset.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/Classification/get_multiclass_data.py

###########################################
# Explore data
###########################################

# metadata
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/Metadata/get_metadata_table.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/Metadata/get_metadata_table2.py
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/Metadata/analyse_metadata_table.py

# correlation matrices
PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/correlation_matrix.py

###########################################
# Deploy ML
###########################################

# Batch:
# PYTHONPATH=$(pwd) python3 /home/ubuntu/Analytics/crypto/Scripts/ExploreData/correlation_matrix.py
```

- This runs every day, ensuring a live, voluminous dataset.
- More information on using the linux Cron service found here (Mehra, 2017)

6 References

Amazon. (2020). *AWS Instance Scheduler*. Retrieved from https://docs.aws.amazon.com/: https://docs.aws.amazon.com/solutions/latest/instance-scheduler/security.html

Anaconda. (2021). *Managing environments - Creating an environment from an environment.yml file*. Retrieved from https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-from-an-environment-yml-file

Mehra, A. (2017). *The Ultimate Crontab Cheatsheet*. Retrieved from https://devhints.io/: https://www.codementor.io/@akul08/the-ultimate-crontab-cheatsheet-5op0f7o4r

TotalCloud. (2020). *AWS Instance Scheduler | Step by Step tutorial to Start and Stop EC2 and RDS Instances*. Retrieved from https://www.youtube.com/watch?v=oooxZsz0hS4

Zwitch, R. (2013). *Instructions for Installing & Using R on Amazon EC2*. Retrieved from https://www.r-bloggers.com/2013/04/instructions-for-installing-using-r-on-amazon-ec2/