National College of Ireland

Configuration Manual

MSc Academic Internship
MSc CYBERSECURITY

Oluwaseun Odunibosi
Student ID: X18123970

School of Computing
National College of Ireland

Supervisor: Mr VIKAS SAHNI

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	OLUWASEUN ODUNIBOSI
Student ID:	X18123970
Programme:	Msc CYBERSECURITY **Year:** 2019
Module:	ACADEMIC INTERNSHIP
Lecturer:	VIKAS SAHNI
Submission Due Date:	12/12/2019
Project Title:	CLASSIFICATION OF EMAIL HEADERS USING RANDOM FOREST ALGORITHM TO DETECT EMAIL SPOOFING
Word Count:	1951 **Page Count:** 19

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: ……

Date: ……

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

Forename Surname
Student ID:

1 SYSTEM CONFIGURATION

The system configuration of the system used is Windows 10, 64bits Operating System, ICORE 5 with 8BIT RAM and 500 GIG hard drive.

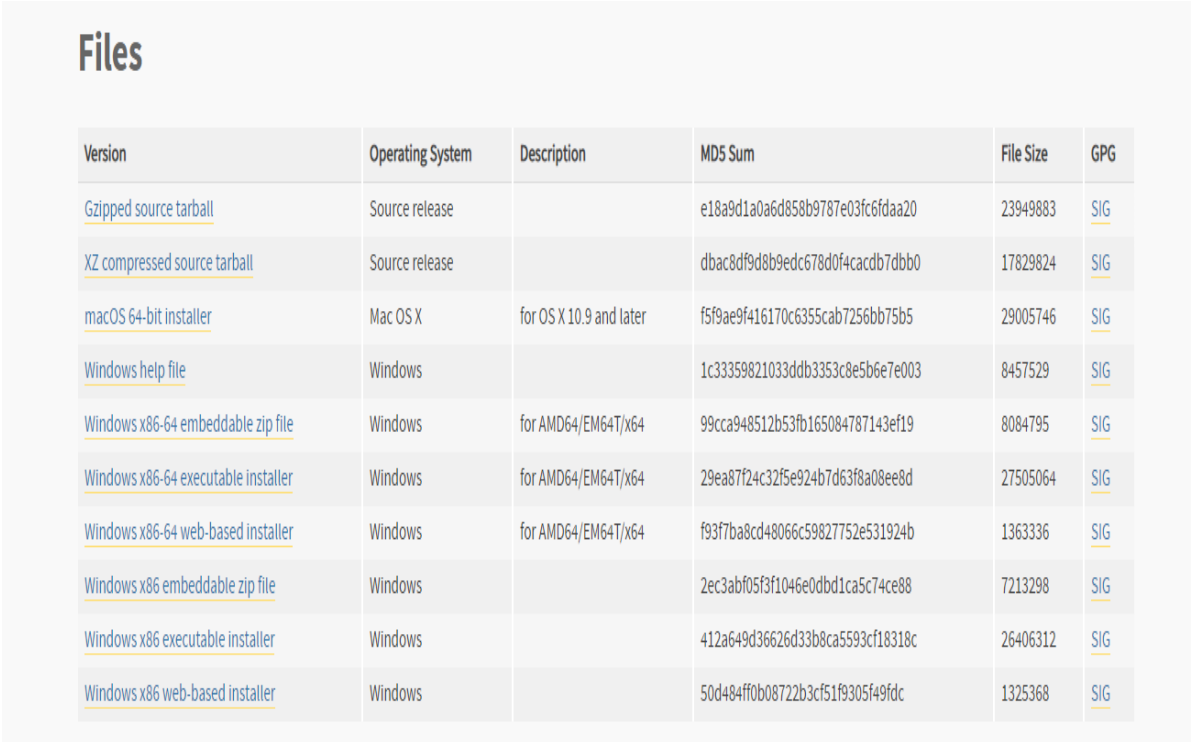
2 INSTALLING REQUIRED SOFTWARE

The various software to be used in the research will be installed as shown below,

2.1 DOWNLOAD PYTHON 3.8.0

Python version 3.8.0 must be downloaded on user system following the steps below. Download Python version 3.8.0 from this site.[1]

- Select "Latest python 3 Release – Python 3.8.0", Scroll down to Files and click on "Windows X86-64 executables installer" as shown below



Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		e18a9d1a0a6d858b9787e03fc6fdaa20	23949883	SIG
XZ compressed source tarball	Source release		dbac8df9d8b9edc678d0f4cacdb7dbb0	17829824	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	f5f9ae9f416170c6355cab7256bb75b5	29005746	SIG
Windows help file	Windows		1c33359821033ddb3353c8e5b6e7e003	8457529	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	99cca948512b53fb165084787143ef19	8084795	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	29ea87f24c32f5e924b7d63f8a08ee8d	27505064	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	f93f7ba8cd48066c59827752e531924b	1363336	SIG
Windows x86 embeddable zip file	Windows		2ec3abf05f3f1046e0dbd1ca5c74ce88	7213298	SIG
Windows x86 executable installer	Windows		412a649d36626d33b8ca5593cf18318c	26406312	SIG
Windows x86 web-based installer	Windows		50d484ff0b08722b3cf51f9305f49fdc	1325368	SIG

FIG. 1: shows download page of Python.

- Once downloaded, go to the location the file is saved and click on the installer.

[1] https://www.python.org/downloads/windows/

- Select “Install launcher for all users (recommended)” and Select ‘Install Now’ which include installation of IDLE, pip and documentation and creates shortcuts and file associations as shown in the Fig. 2.

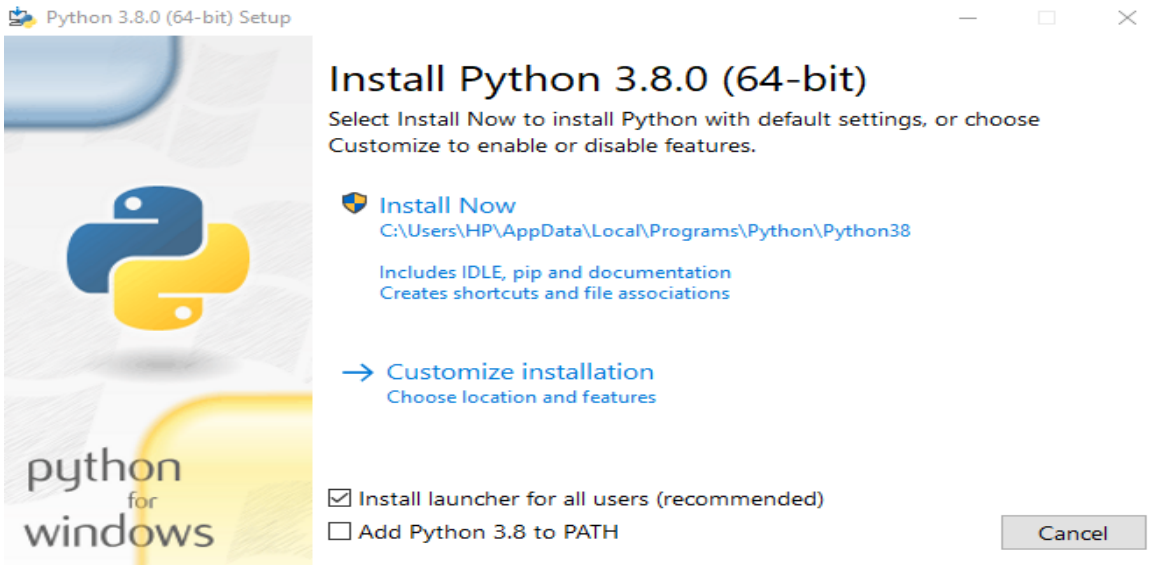


FIG. 2: Shows the installation process

- If “Add python 3.8.0 is selected”, then then the installation directory will be added to your PATH.
- You will see a screen showing the installation process, when installation is finished, a screen showing as shown below in Fig. 3 will appear on the screen.

FIG.3: Installation Successful

Running Python Application

- Click close and navigate to the windows command prompt.
- Once the command prompt is opened, execute this command to check if python is working and version installed as shown below.

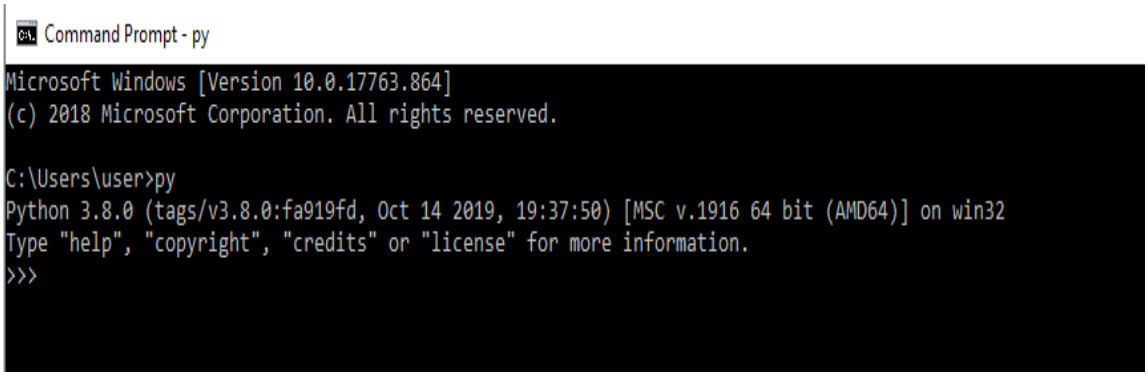


FIG. 4: Shows Python application is working and the Version

User may encounter an alert during installation that instruct user to "Remove the MAX_PATH Limitation" before installation can continue. This is because Windows has path length of 260 characters and any paths longer than 260 would result in error. If encountered; In latest version of Windows, this limitation, user can expand this limitation using like 32,000 characters. Activate the "enable Win32 long paths" group policy. This can also be resolved by setting the registry key value (HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystem@LongPaths Enabled) to 1.

2.2 DOWNLOAD PYCHARM VERSION 2019.3

- Download the PyCharm Version 2019.3 from this site[2]
- Click Download Now.
- Click on the Download Button under the Community Edition. Only free trial of Professional edition will be available if you select that except you want to purchase.

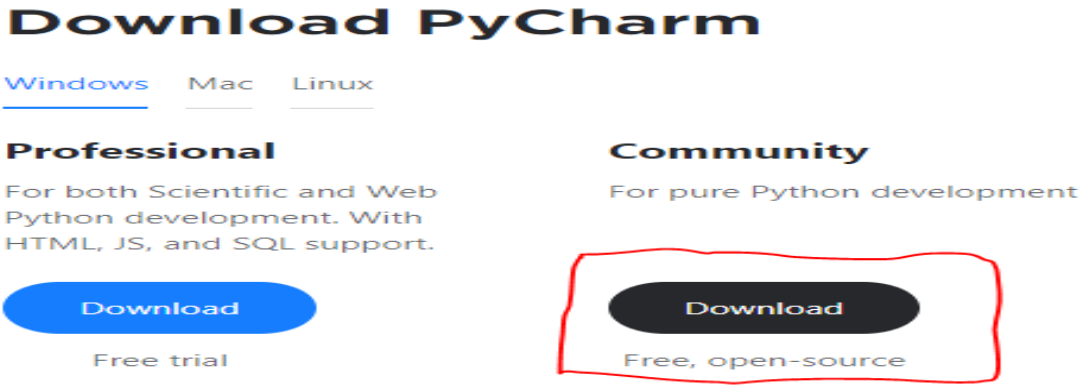
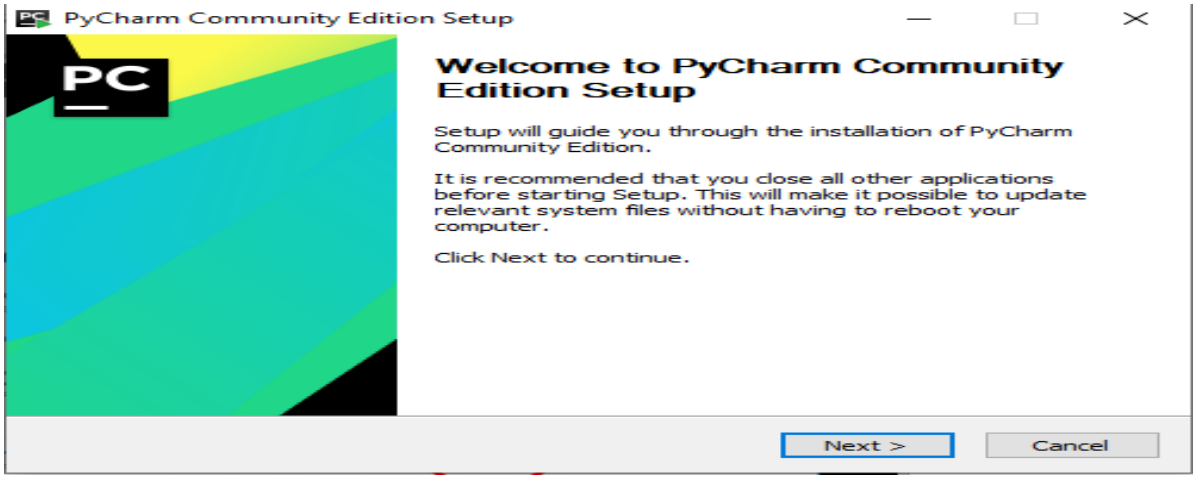


FIG. 5: PyCharm Installation

- Double click on the Downloaded PyCharm, a pop up will appear on the screen as shown below. Click next

[2] https://www.jetbrains.com/pycharm/



- Choose the location where installation will be saved, I recommend choosing the suggested location. Click next when done.

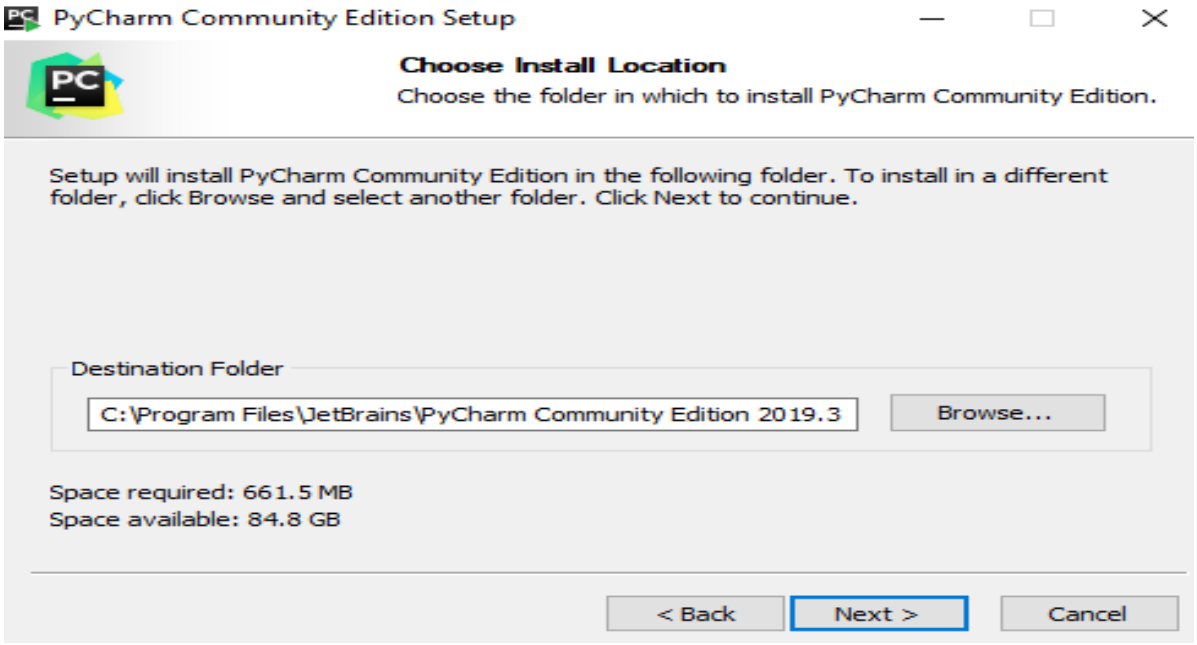


FIG. 7: Location to Save

- Select the first and the third options as shown in the diagram below and click next. The first option will create a desktop shortcut where user can easily navigate to PyCharm. The third option will create association so that when Python file is opened, it will automatically open in PyCharm.

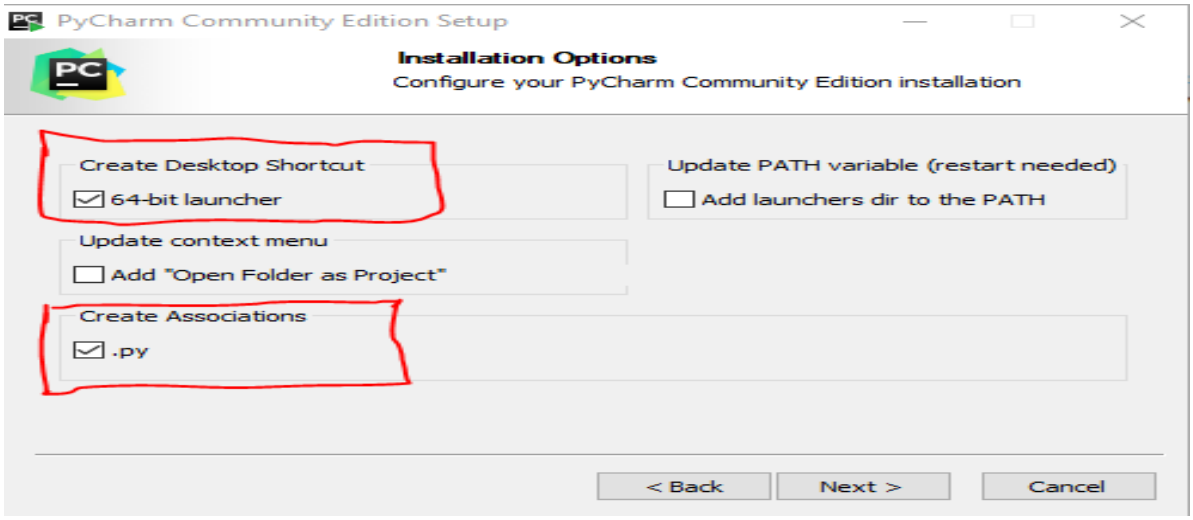


FIG. 8: Installation process

- Select the start menu folder and click Install, I recommend leaving the default on the screen "JetBrains"

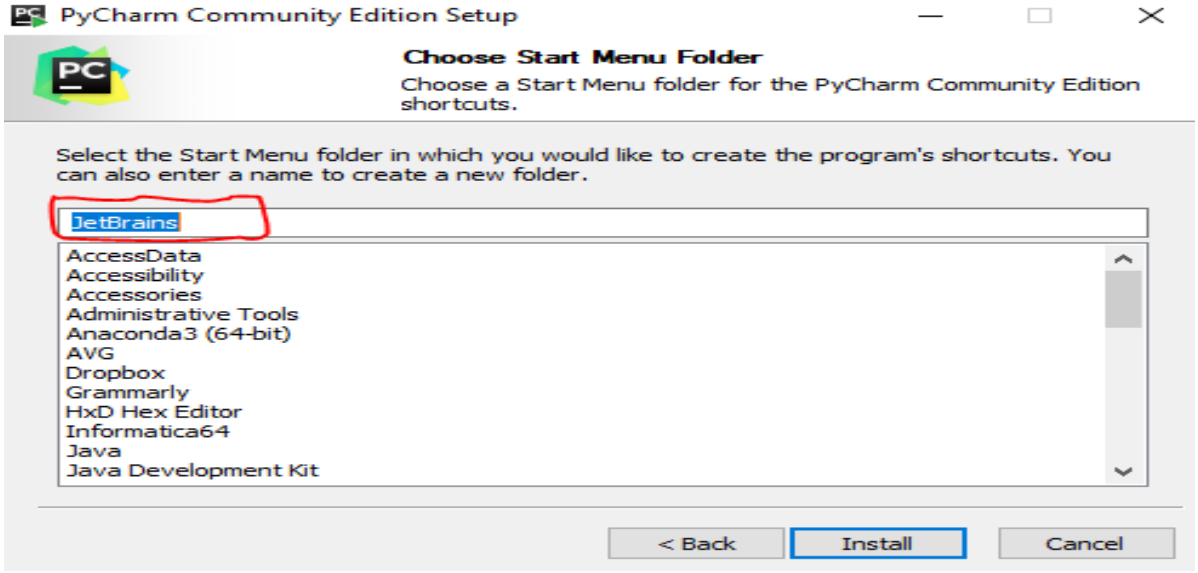


FIG. 9: Selecting Start Menu Folder

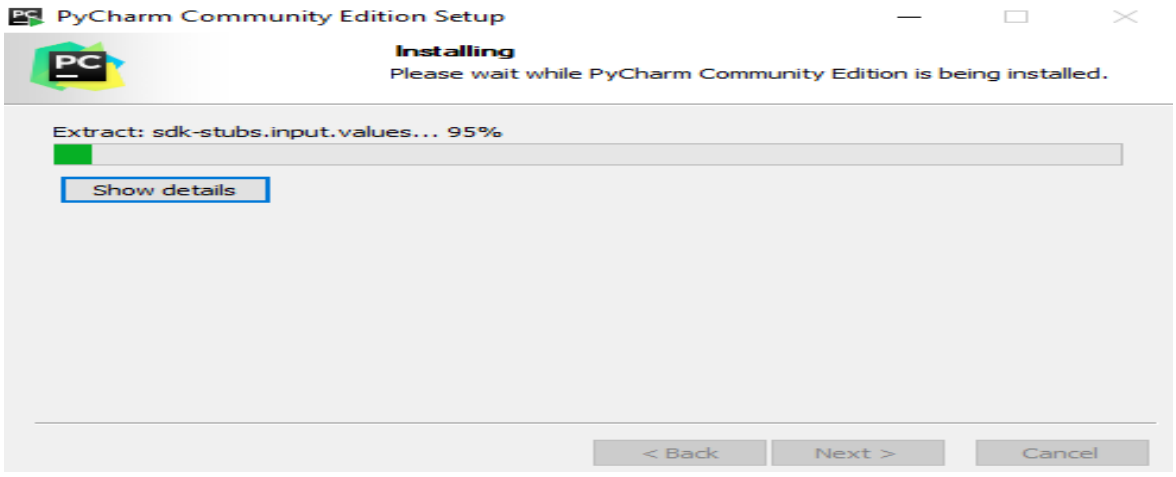


FIG. 9: Shows Installation Process

- Once installation is done, a message that PyCharm is installed is shown. Click “Finish”. If you want to run the PyCharm application immediately, click “Run PyCharm Community Edition” first before you click “Finish”

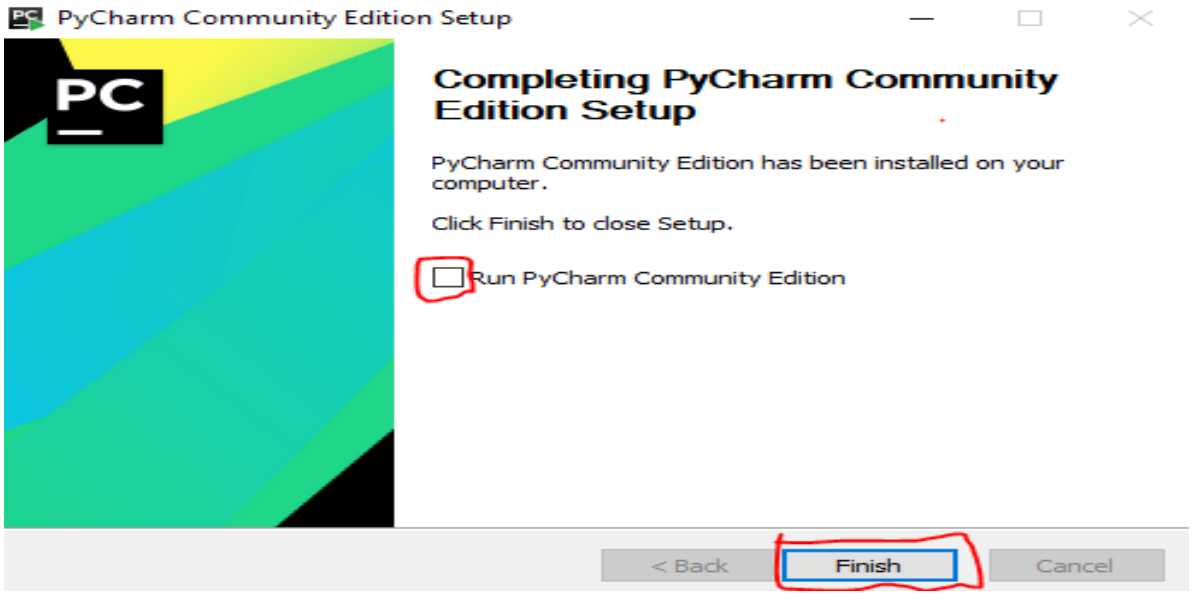


FIG. 10: PyCharm Installation finished.

Running PyCharm Application

- User will get a message box asking about import settings when you run the PyCharm application for the first time. Select “Do not Import Settings” and click Ok.

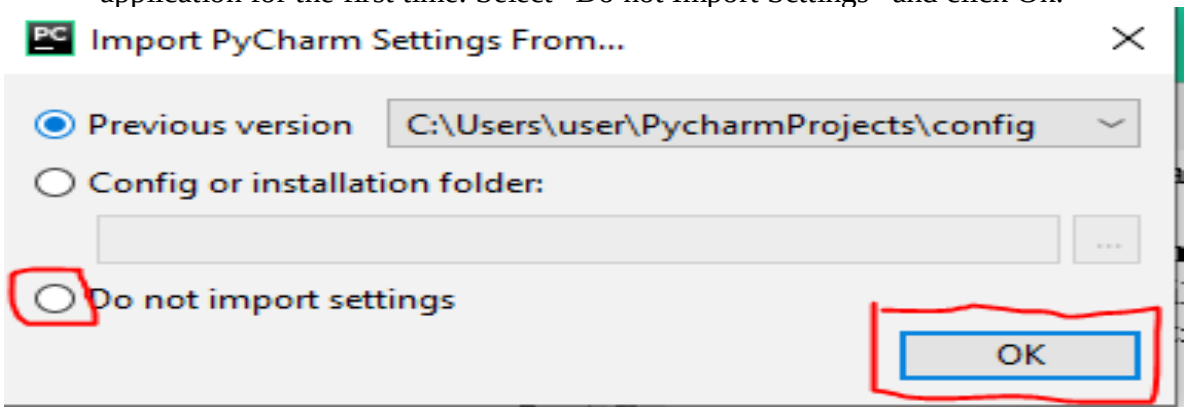


FIG. 11: PyCharm Import Setting

- Some initial configuration will have to be performed when running the PyCharm for the first time as shown below. Select the Customization menu that suit you (Dracula or Light) and click next featured plugins. I recommend skipping remaining setup and Set all as Default shown on the bottom left of the screen.

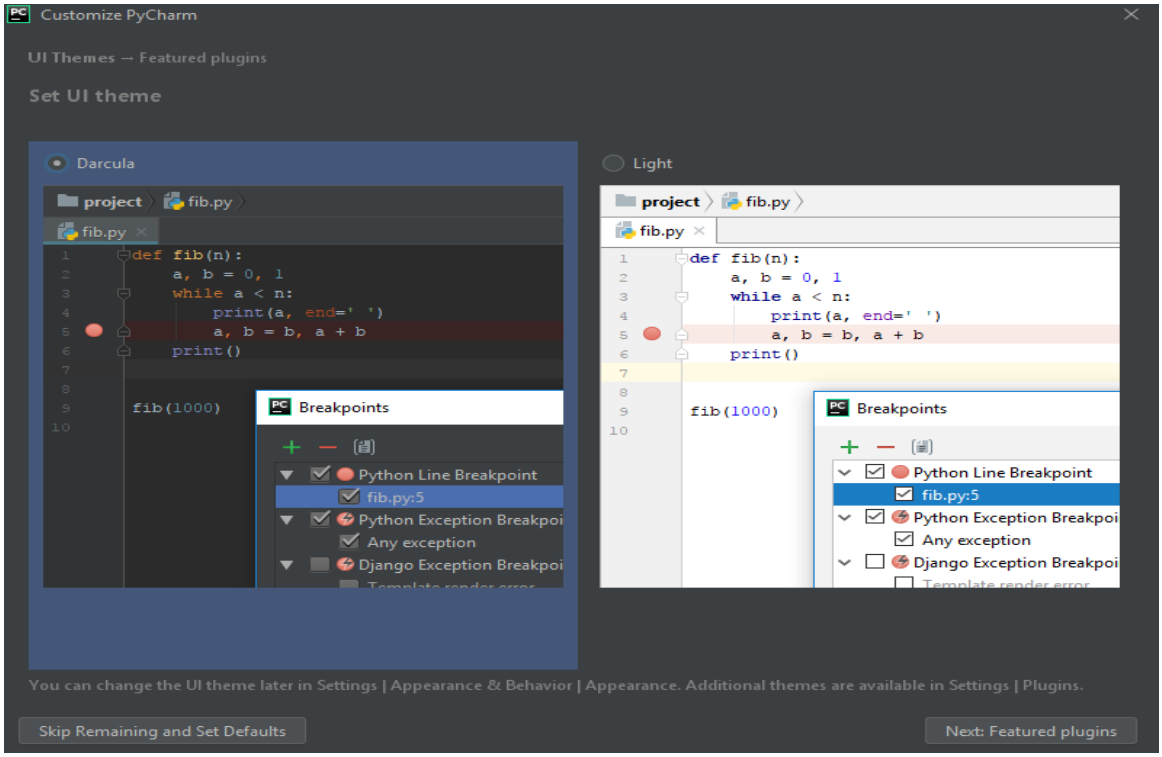


FIG. 12: Configuration Setting

- A intro screen will appear on the screen for PyCharm. Click on “Create New Project”

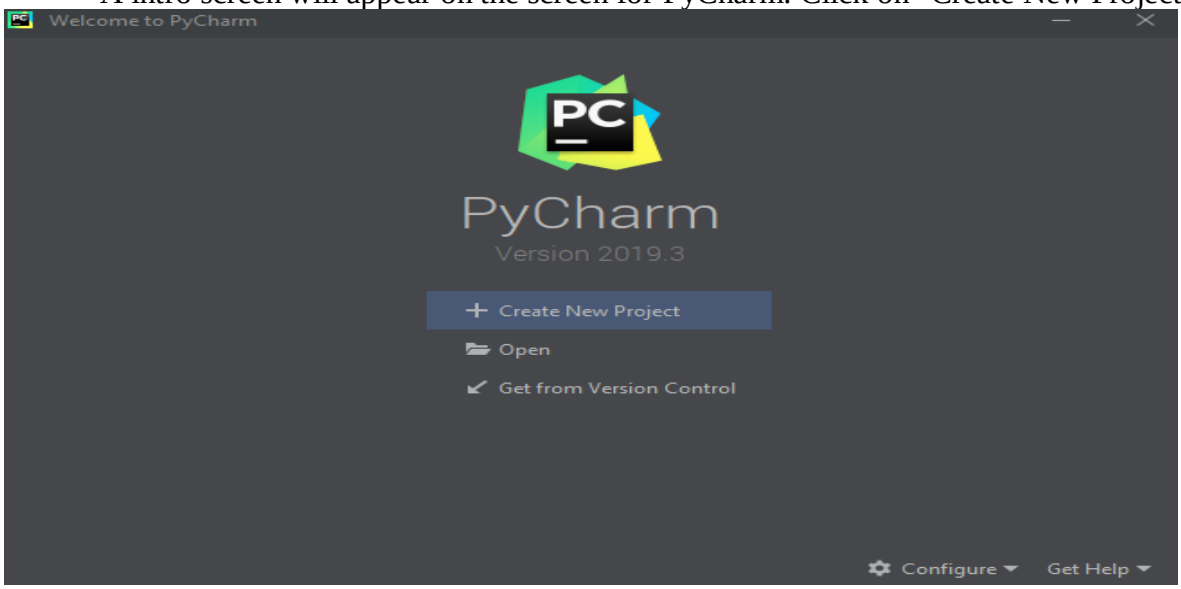


FIG. 13: New Project Screen

- Select location where the project created will be saved, change the name of the “Untitled” to “MyProject” and on the “Project Interpreter: New Virtualenv environment” dropdown and select “Inherit global site-packages” and “Make available to all projects”. PyCharm would have found the Python interpreter installed earlier in section 1 and this must appear in the base interpreter. If nothing is showing in the base interpreter field, then it must be resolved before you click “Create”.

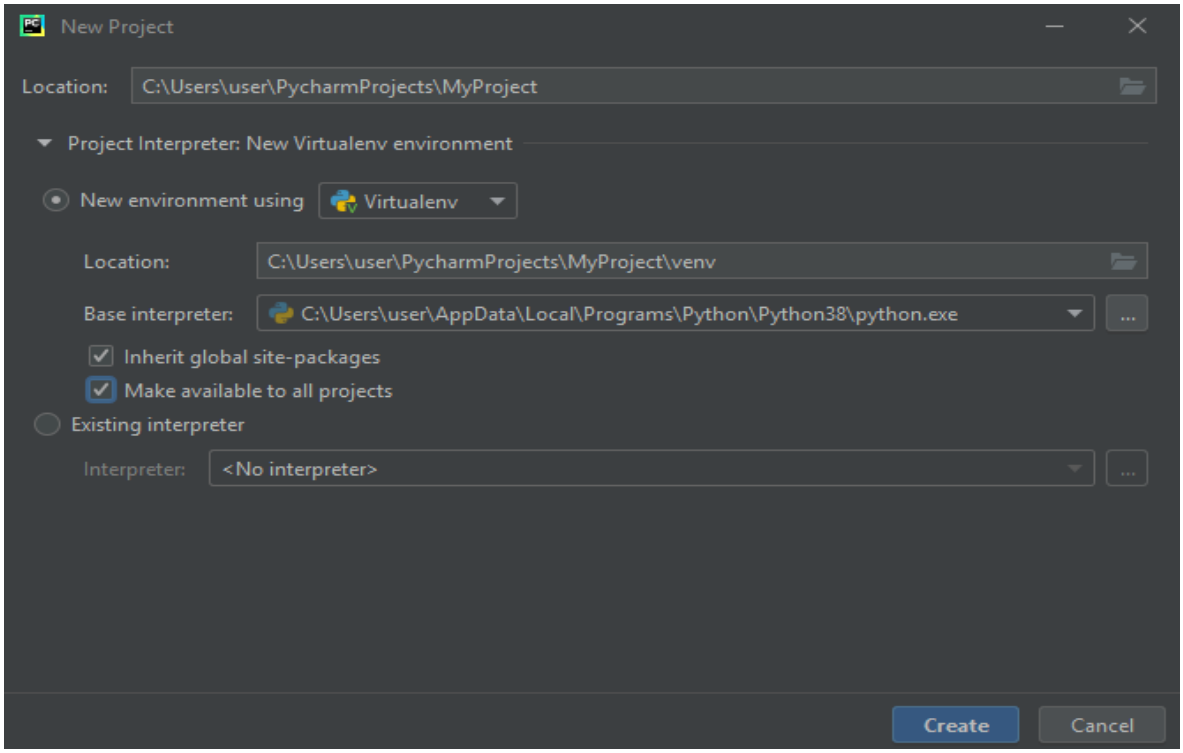


FIG. 14: New Project Creation.

- The PyCharm environment is shown once the "Create" button is selected, close the tips menu so you can have access to the PyCharm main environment.

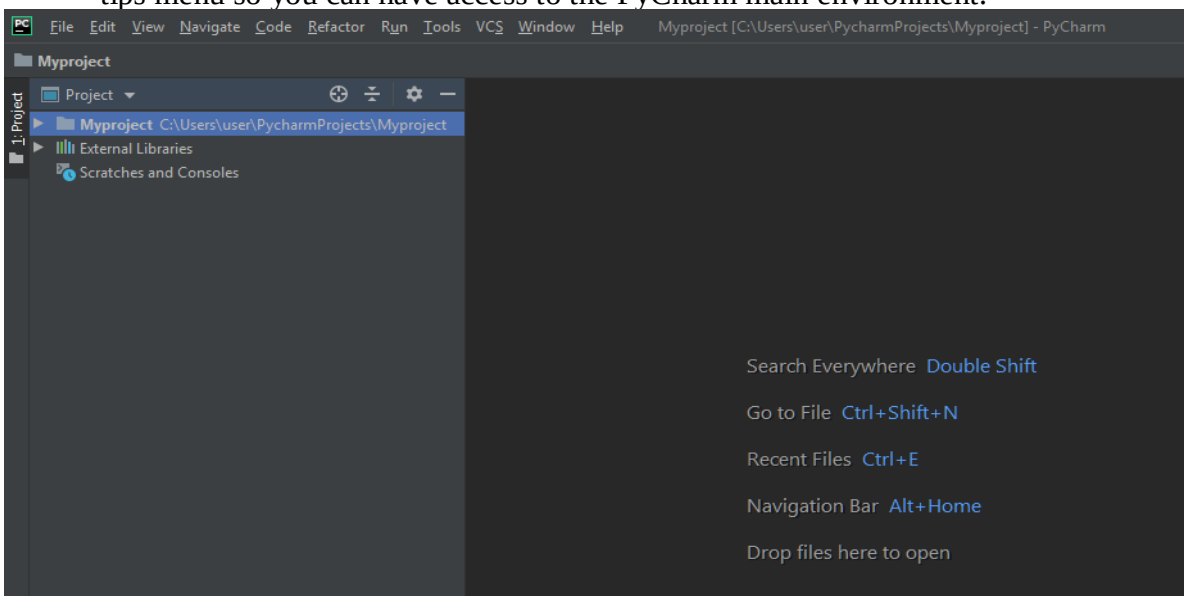


FIG. 15: PyCharm Environment

- To create and run the Email_Spoofing_Classifier_final of this research, navigate "File" menu and select "New" and select "Python File".

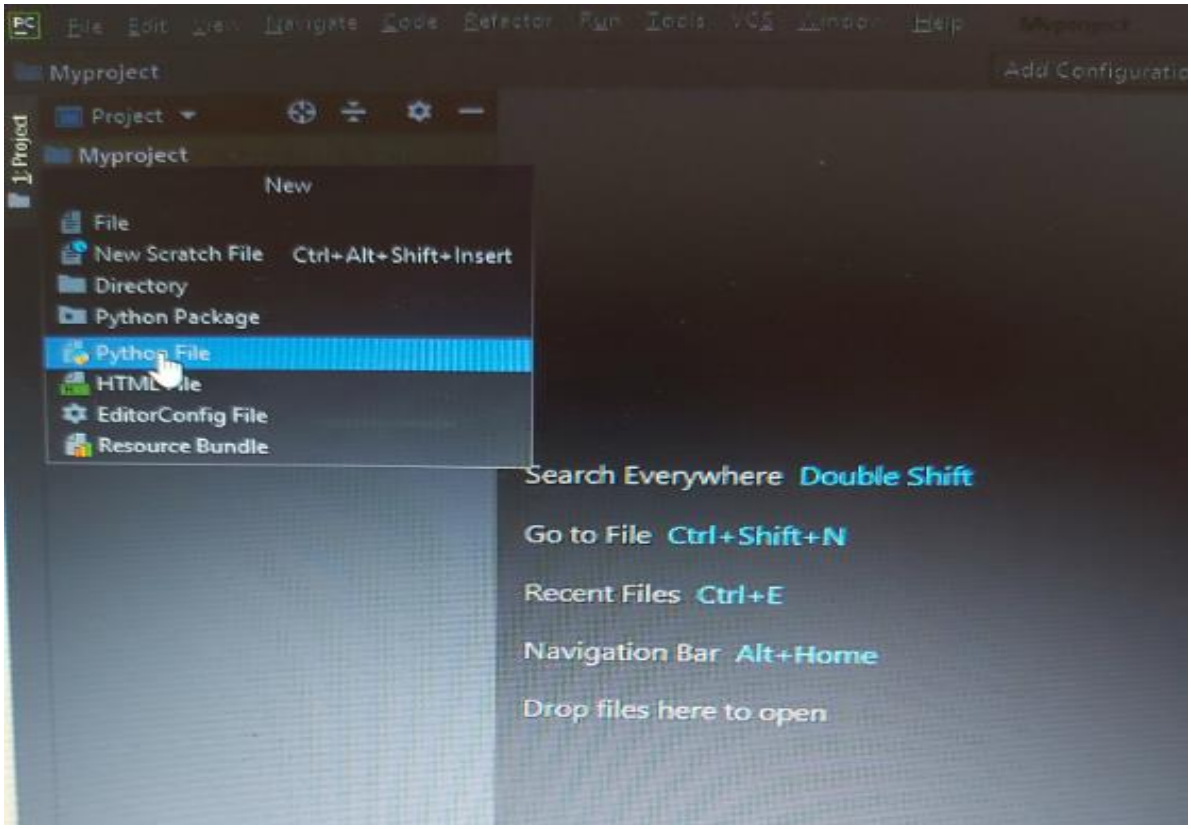


FIG. 16: Creating a program

- A menu will appear where you will need to type in the python file name then click "Python file" as seen below

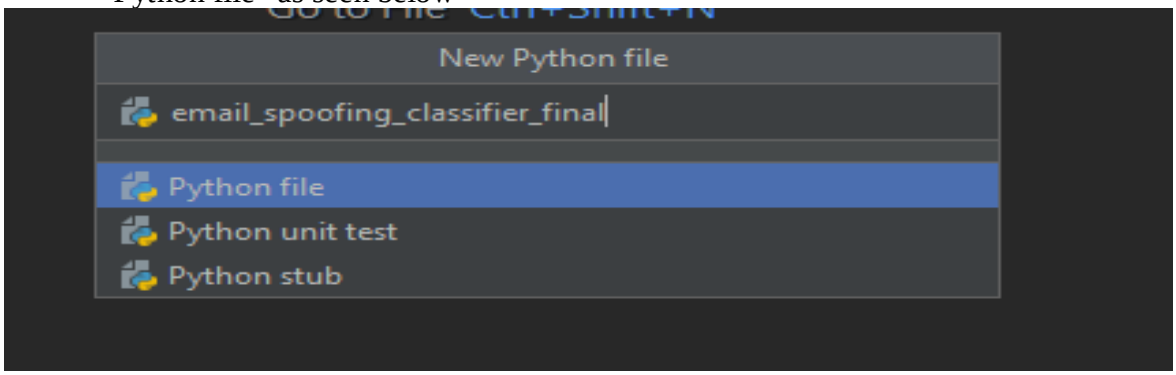


FIG. 17: Python File name Creation

- This brings the user to the new created Python file named "Email_Spoofing_Classifier_final". This menu is where user will start building the code for the research.

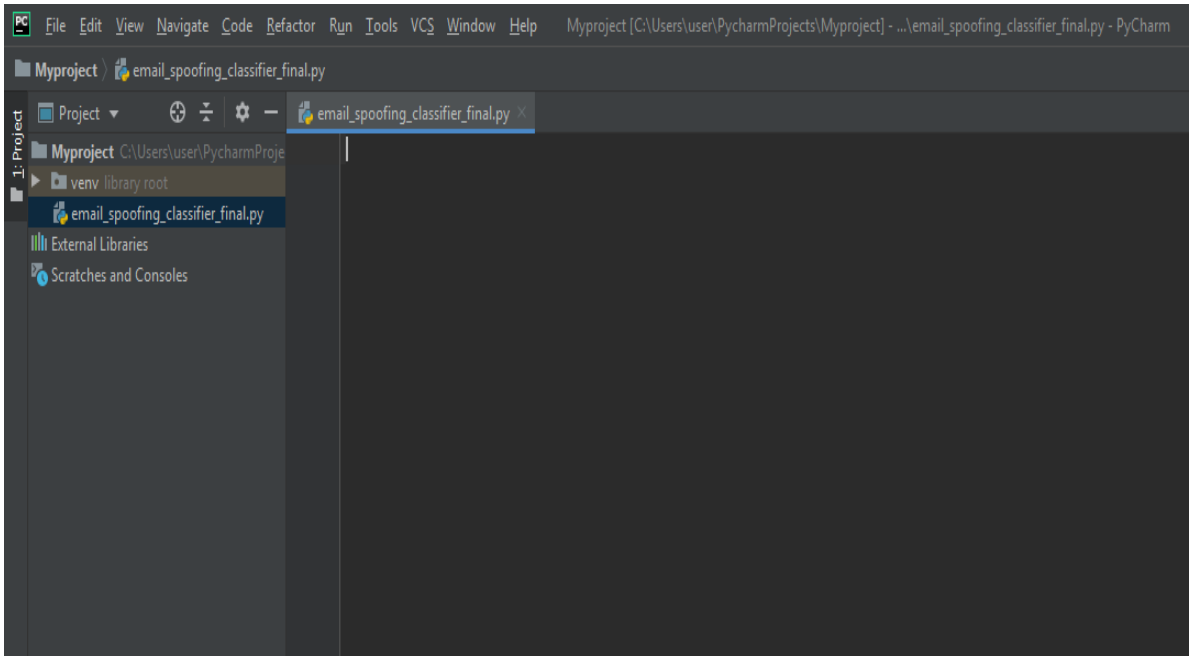


FIG. 18: The Python Menu for coding

3 CODE FOR THE RESEARCH WORK

The code below will be pasted on the Email_Spoofing_Classifier_final python menu. The code must be as specified below except the FROM_EMAIL and FROM_PWD field which must be changed to user own Gmail username and Password. Note that some of this code are general convention of using the module imported so they are not change. More information can be found in the code reference or in this site.[3]

3.1 Email_Spoofing_Classifier_final python

```
import smtplib
import time
import imaplib
import email
import csv
from datetime import datetime
import randforest as rf
import argparse

# GMail login
# ---------------------------------------------------
# Tittle: How to Read Email From Gmail Using Python, 2017. . Code Handbook.
# Author: JAY
# Date: 2017
#Availability:https://codehandbook.org/how-to-read-email-from-gmail-using-
python/#comment-4217804161
# ---------------------------------------------------
```

[3] https://codehandbook.org/how-to-read-email-from-gmail-using-python/#comment-4217804161

```
SMTP_SERVER = "imap.gmail.com"
SMTP_PORT = 993
FROM_EMAIL = "cainersteph@gmail.com"
FROM_PWD = "Stephen-1901"


# ------------------------------------------------
#
# Save message to CSV file
#
# ------------------------------------------------
def save_output(filename, row):
    with open(filename, 'a') as f:
        writer = csv.writer(f)
        writer.writerow(row)


# ------------------------------------------------
#
# Read mails from GMAIL Inbox
#
# ------------------------------------------------
def read_mails(filename, email_counts):
    # Handle connection exception
    try:
        # connect to smtp server
        mail = imaplib.IMAP4_SSL(SMTP_SERVER)

        # Login
        mail.login(FROM_EMAIL, FROM_PWD)
        mail.select('inbox')

        # Search for emails
        type, data = mail.search(None, 'ALL')
        mail_ids = data[0]

        # Process emails
       # ------------------------------------------------
       # Tittle:How to Read Email From Gmail Using Python, 2017. . Code Handbook.
       # Author: JAY
       # Date: 2017
       #Availability:https://codehandbook.org/how-to-read-email-from-gmail-using-
       python/#comment-4217804161
       # ------------------------------------------------

        id_list = mail_ids.split()

        first_email_id = int(id_list[0])
        latest_email_id = int(id_list[-1])
```

```
        # Fetch emails from latest to old
        for i in range(latest_email_id, first_email_id, -1):
            typ, data = mail.fetch(str(i), '(RFC822)')

            if (email_counts > 0):
                for response_part in data:
                    if isinstance(response_part, tuple):
                        # Decode message from response string
                        msg = email.message_from_string(response_part[1].decode('utf-
8').strip())

                        # Prepare output to CSV file
                        email_subject = msg['subject']
                        email_from = msg['from']
                        email_to = msg['to']
                        msg_id = msg['message-id']
                        date = msg['date']

                        # Message
                        csv_content = [email_from, email_to, email_subject, msg_id, date]

                        print("Reading and storing messages in " + filename)

                        # Save message to CSV file
                        save_output(filename, csv_content)

                        # Progress report
                        print()

                        print("Successfully written to " + filename + ": " + str(csv_content))

            else:
                break

            email_counts = email_counts - 1;

    except Exception as e:
        print(str(e))

    return filename


def main():
    # initiate the parser
    parser = argparse.ArgumentParser()
    parser.add_argument("-e", "--emails", help="Input the number of emails to read")
    parser.add_argument("-d", "--dump", help="Provide an exist email header dump
file")

    # Execute command line arguments
    args = parser.parse_args()
```

```
    if (args.emails):
        email_counts = int(args.emails)
    else:
        # Default email counts
        email_counts = 1000

    if (args.dump):
        # Read emails for local file
        filename = args.dump
        print("Email header file to classify: {}".format(filename))
    else:
        # Read emails from gmail
        # Create csv file
        filename = "csvfile_{}.csv".format(datetime.now().strftime("%H-%M-%S"))

        # CSV Fields
        fields = ['From', 'To', 'Subject', 'Message-ID', 'Date']
        save_output(filename, fields)

        filename = read_mails(filename, email_counts)

    # New line
    print()

    # Prompt the user to decide on data classification
    confirm = input("Do you want to classify the data as well? (Yes or No) ");
    confirm = confirm.lower()

    if ("yes".find(confirm) != -1):  # classifly data if yes
        rf.random_forest_algo(filename)


if __name__ == "__main__":
    main()
```

3.2 Creating the Random Forest Algorithm

Following the procedure used in creating Email_Spoofing_Classifier_final above.

- Repeat procedures carried out in Fig. 16
- Follow procedure on Fig. 17 but change the name of the new python file to be created to “randomforest” as shown below

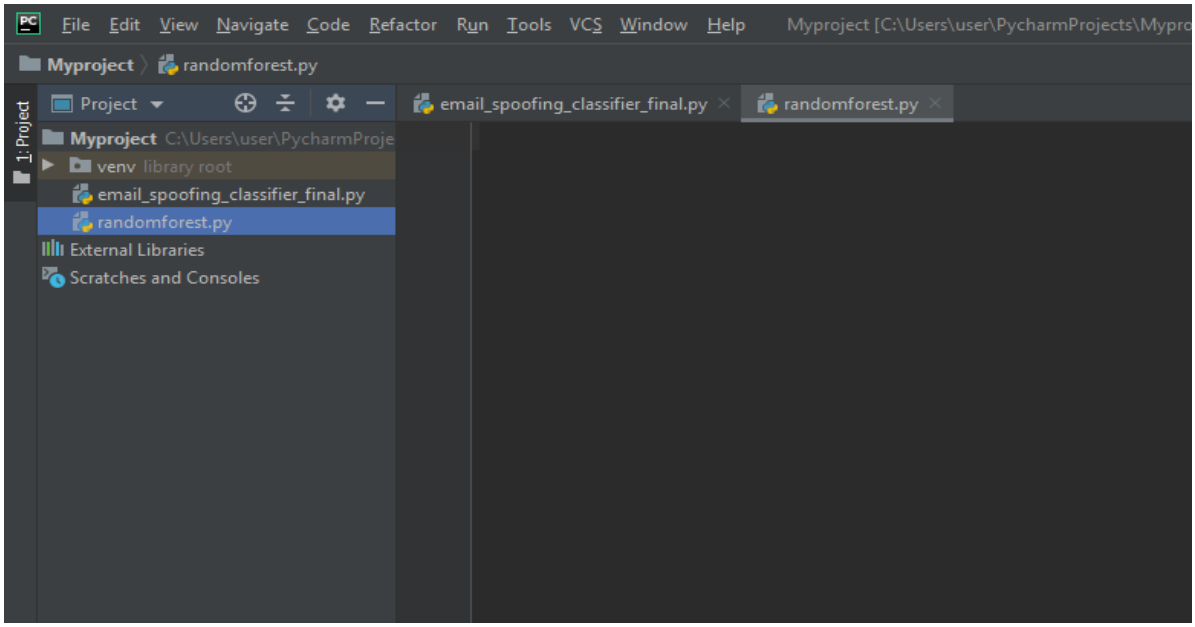


FIG. 19: Creation of randomforest python file

- Put in this code below

```
import warnings
from sklearn.exceptions import DataConversionWarning
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score


#turn off warning error
warnings.simplefilter("ignore")

def random_forest_algo(filename):
    df = pd.read_csv(filename)
    df.head()

    with pd.option_context('display.max_rows', 1000):
        print (df)
#Spliting data to test and training set
        X, y = make_classification(n_samples=1000, n_features=2,
n_informative=2, n_redundant=0, random_state=5)
        X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3)
        X_train
        X_test
    classifier = RandomForestClassifier(n_estimators=20, random_state=5)
    classifier
    classifier.fit(X, y)
    print(classifier.feature_importances_)

    y_prediction = classifier.predict(X_test)
    y_prediction
```

```
# Accuracy
print(metrics.accuracy_score(y_test, y_prediction))
classifier.apply(X)
classifier.get_params(deep=True)
classifier.predict_log_proba(X)
classifier.predict_proba(X)
classifier.score(X, y)

# Confusion Matrix
print(confusion_matrix(y_test,y_prediction))
print(classification_report(y_test,y_prediction))
print(accuracy_score(y_test, y_prediction))
```

4 SECURE LESS APPLICATION ON GMAIL ACCOUNT.

User must be log in to Gmail account before carrying out the following process.

- Login to Gmail account from the browser.
- Sign in with Username and Password.
- Once the inbox of Gmail is opened, navigate to “Google Account” on the upper right of the gmail menu and click “Manage your Google Account”
- A new screen is opened showing Personal info, Data and personalization etc. Select “Security” and scroll down to “Less secure app access”. Click on “Turn on” and go back to inbox. It is important to turn on less secure application for our code to capture user received mail.

5 RUN THE Email_Spoofing_Classifier_final CODE.

The final stage is to go back to the opened Email_Spoofing_Classifier_final code in PyCharm

- Highlight the “Email_Spoofing_Classifier_final” and right click on it and on the dropdown menu, click “Run Email_Spoofing_Classifier_final”.

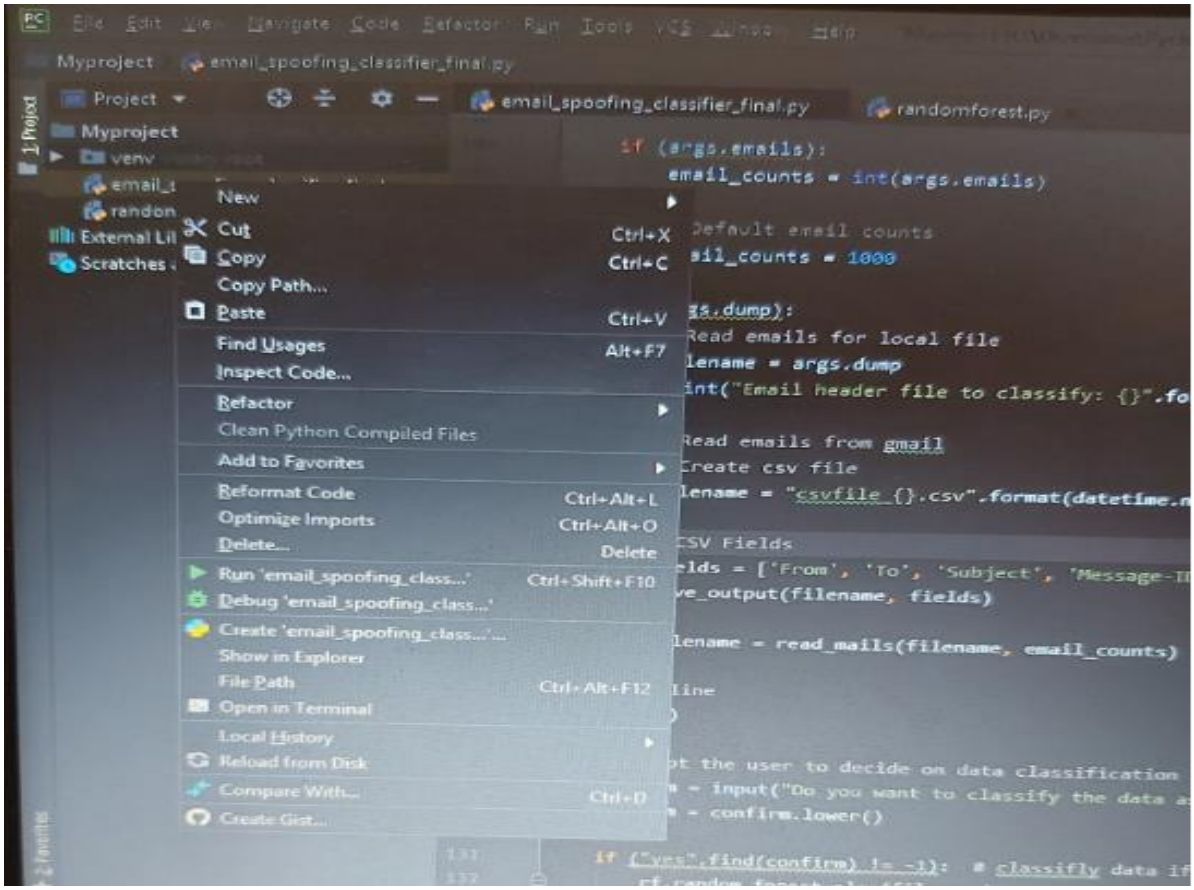


FIG. 22: Executing the code

- The result of executing the Email_Spoofing_Classifier_final code will be shown at the bottom on the “Run Terminal”.

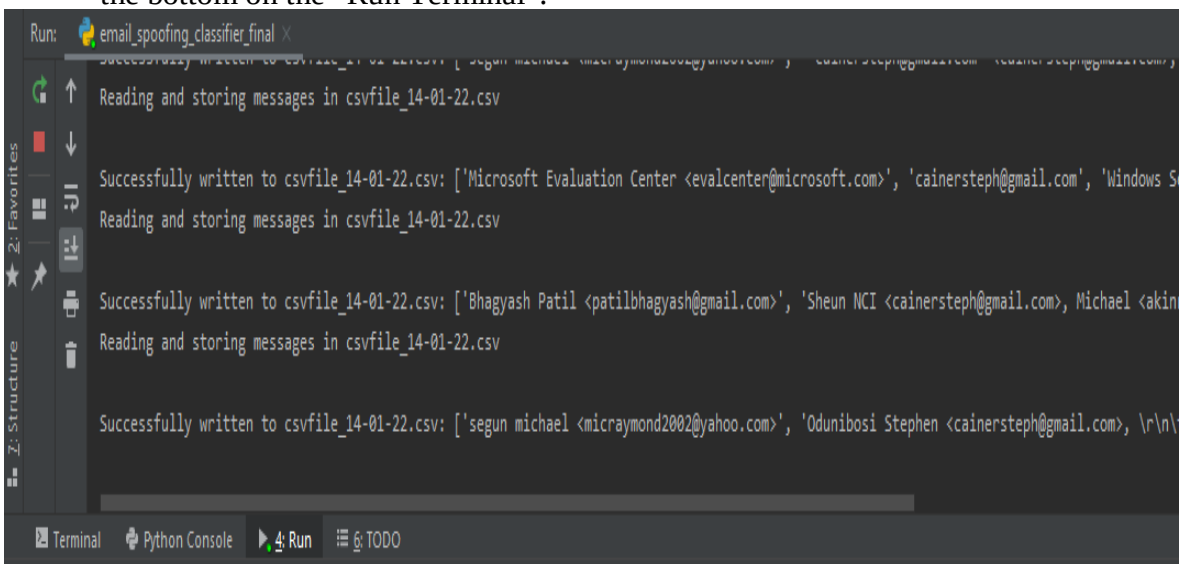


FIG. 23: Results Of code Execution

- In the course of execution of the code User is prompted on the Run terminal if classification of mails should be carried out. User should write Yes and click enter.

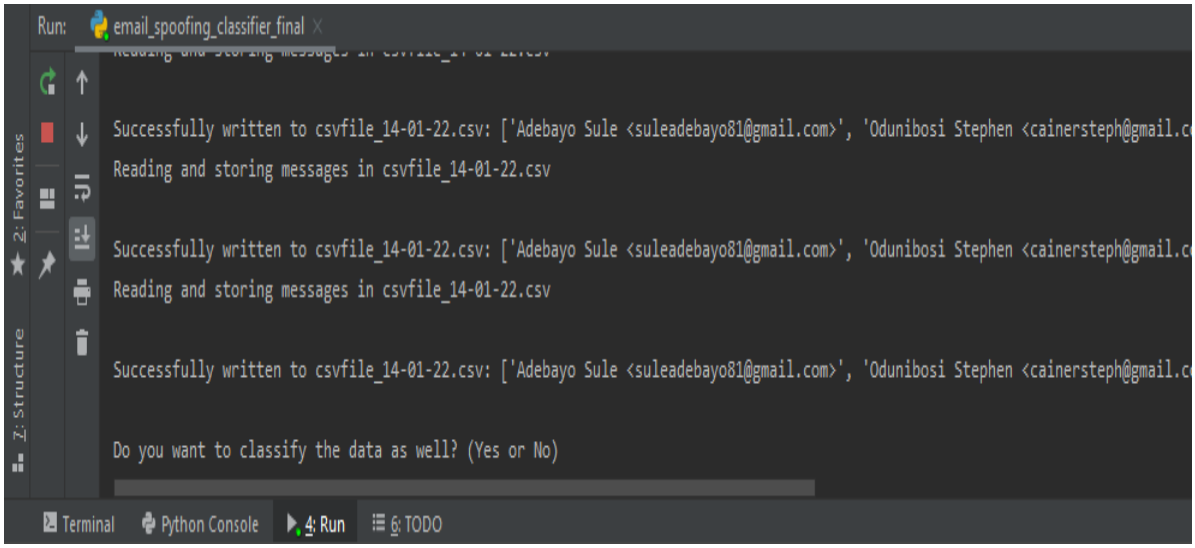


FIG. 24: Confirmation of Classification

- The extracted email saved in CSV can also be viewed as below.

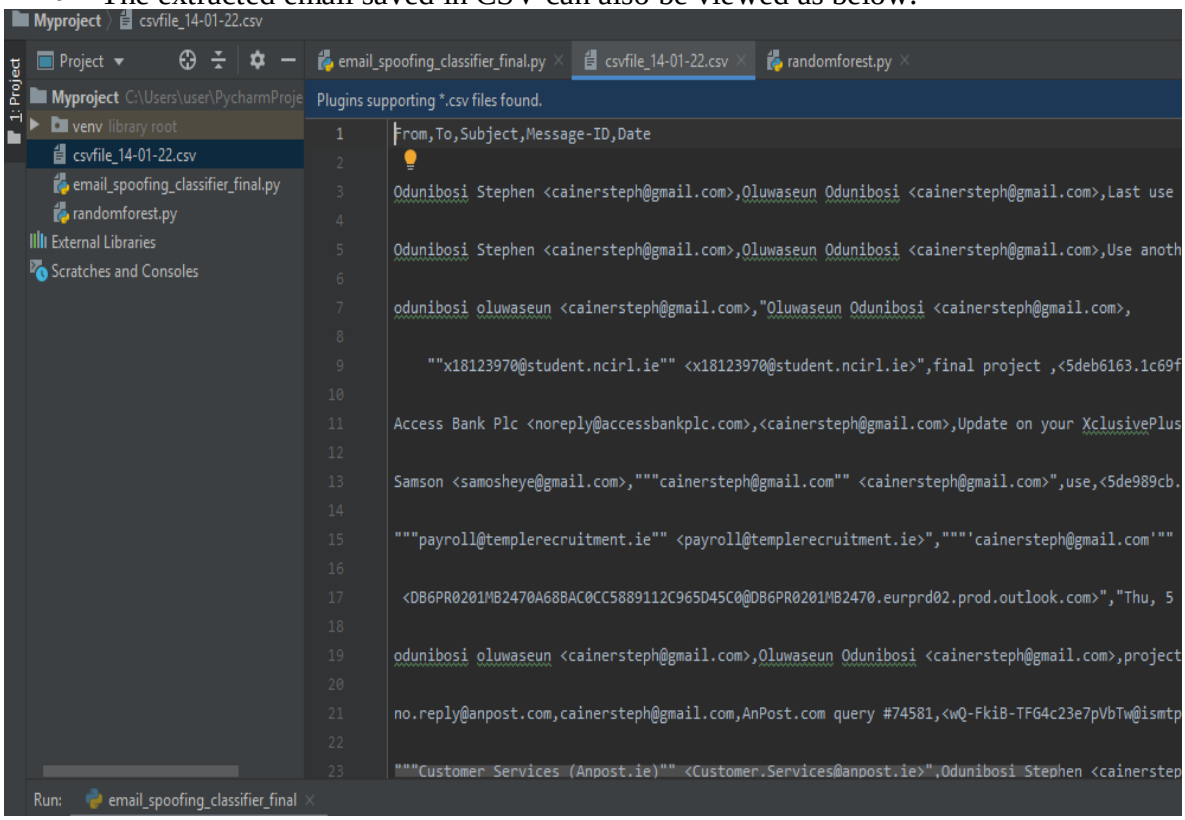


FIG. 25: CSV format email