Report Explanation

Goal of the task: The main objective in this task is to match the suspicious bank transactions with the blockchain transaction hashes and calculate data correlation between necessary numerical values

so we have 4 datasets that has to be used to reach our goal: 3 datasets regarding the blockchain transaction (2015.csv, 2016.csv, 2017.csv), and the other one of the suspicious bank transactions (download_transactions_map.csv).

So to do that I have chosen the methods and tools below:
Method and tools: Python with data analysis libraries (Pandas, Numpy), under Ubuntu 18.04

after reading and examining the data, the data that represent the blockchain transactions are in the datasets 2015.csv, 2016.csv and 2017.csv and the suspicious bank transactions are in download_transactions_map.csv
So I have to match all this datasets together with common data between them.
The common data is the time when the transactions happen in each dataset, but the problem is that the datatype of the time or date here is of type object

let's start:
first I have to load our datasets to work with
Loading the data:

```
import pandas as pd
import numpy as np

# load and read the csv files
df1 = pd.read_csv("/home/bilal/inca/download_data_fincen_files/download_transactions_map.csv")
pd.set_option("display.max_columns", 20)
df_2015 = pd.read_csv("/home/bilal/inca/Bitcoin-large-transactions-2015_2016_2017/2015.csv")
df_2016 = pd.read_csv("/home/bilal/inca/Bitcoin-large-transactions-2015_2016_2017/2016.csv")
df_2017 = pd.read_csv("/home/bilal/inca/Bitcoin-large-transactions-2015_2016_2017/2017.csv")
```

Cleaning the data: after loading the data and having a view of our datasets, there are some unnecessary or duplicated columns in transactions_map dataset that I decided to clean and remove them as you can see below

```
# data cleaning: remove unnecessary columns
df1.drop(["originator_iso", "beneficiary_bank_id", "beneficiary_iso", "filer_org_name_id", "originator_bank_id", "end_date"],
        axis=1, inplace=True)
```

Now, the step is to match and cross the suspicious bank transactions with the blockchain ones, by the same date or time when the transactions happen, so the begin date in transaction_map and time in each blockchain transaction datasets.
But the problem is that in blockchain transactions datasets the datatype of time column is of type object and the begin date is of type string, so I had to transform the begin date string to numerical format then to datetime format, as well as column time in each blockchain transaction datasets

```python
# transform begin date of our dataset df1 to datetime format, for that I created a dictionary where I transforme the string format
# of months to numerical format then from numerical to datetime format

dictionary_months = {"Jan": "01", "Feb":"02", "Mar": "03", "Apr": "04", "May": "05", "Jun": "06", "Jul":"07", "Aug":"08",
                     "Sep": "09", "Oct": "10","Nov": "11", "Dec": "12"}

#1 Transforming months to numerical format
list = df1["begin_date"]

lst = []

for string in list:
    a = str(string)
    b = a.replace(",", " ")
    lst.append(b.split())

lst2 = []
for index in np.arange(len(lst)):
    for month in dictionary_months:
        if lst[index][0] == month:
            lst[index][0] = dictionary_months[month]
    lst2.append("/".join(lst[index]))


df1["begin_date"] = lst2
#transforming to datetime format
df1["begin_date"] = pd.to_datetime(df1["begin_date"], format="%m/%d/%Y")
```

Transforming time column in blockchain transaction datasets to datetime format

```python
#transform the time column format of our 2015,2016, 2017 csv  to datetime format to concatinate our dataset with because before
# the data type of time was object type
df_2015["time"] = pd.to_datetime(df_2015["time"])
df_2016["time"] = pd.to_datetime(df_2016["time"])
df_2017["time"] = pd.to_datetime(df_2017["time"])
```

This following below is the final step, I'm merging and joining the suspicious bank transaction in transaction_map dataset with the blockchain transaction and save all in one dataset, so I had to rename begin date columns first by time as it is in other blockchain datasets then merge all together.
After doing this operation, I got duplicated lines that I removed.

```python
#I have to rename begin date by time to concatenate it with other datasets by column time
df1.rename({"begin_date": "time"}, axis=1, inplace=True)
```

```python
# creating a list of bitcoin transaction datasets so as to concatinate them in one and merge it with the transaction map
#dataset
datasets = [df_2015, df_2016, df_2017]
df_merged = pd.concat(datasets)
#df3 = df3.fillna('na')
#print(df_merged.head())
df3 = pd.merge(df1, df_merged)
#print(df3.head())
#df3.info()

df3.drop_duplicates(subset="id", keep="first", inplace=True)
# below, I have just ordered the columns for easy reading and understanding the transactions
df3 = df3.iloc[:, [3,0,1,4,2,6,13,15,14,11,12,10,5,7,8,9]]
print(df3.head())
# export and save to csv file
df3.to_csv(r"/home/bilal/inca/sus_bank&blckchain_tr.csv")
```

Here is a screen shot of how the final suspicious transaction bank and blockchain hash data look like in our final dataset to see fully the data

open sus_bank&blckchain_tr.csv file to see all the columns

time	id	icij_sar_id	originator_bank	filer_org_name	beneficiary_bank
2015-03-25	223254	3297	CIMB Bank Berhad	The Bank of New York Mellon Corp.	Barclays Bank Plc
2015-03-25	235724	2865	Victoriabank	The Bank of New York Mellon Corp.	Expobank
2015-03-25	240099	3319	Barclays	Barclays Plc	Ukrsibbank
2015-03-30	223255	3297	CIMB Bank Berhad	The Bank of New York Mellon Corp.	Barclays Bank Plc
2016-07-28	223986	2765	Societe Generale Private Banking	Société Générale SA	Gazprombank
2015-05-26	223991	3062	Deutsche Bank AG Taunusanlage 12	OFG Bancorp	Oriental Bank
2015-05-26	227604	2462	JSC Norvik Banka	The Bank of New York Mellon Corp.	BTA Bank
2015-05-26	237753	4348	Amicorp Bank And Trust Ltd	The Bank of New York Mellon Corp.	Credit Suisse, Singapore Branch
2015-05-26	237762	4348	Bank of Communications	The Bank of New York Mellon Corp.	Riyad Bank
2015-04-16	224028	3064	Usaa Federal Savings Bank	The Bank of New York Mellon Corp.	Commonwealth Bank of Australia
2015-04-16	227610	2462	Norvik Banka JSC	The Bank of New York Mellon Corp.	Skandinaviska Enskilda Banken
2015-03-18	224057	3575	UniCredit Bank, Cjsc	The Bank of New York Mellon Corp.	Credit Suisse AG
2015-03-18	230173	3055	Fifth Third Bank	Fifth Third Bank, National Association	Korea Exchange Bank
2015-03-18	230180	3055	Fifth Third Bank	Fifth Third Bank, National Association	Fifth Third Bank
2015-03-18	230181	3055	Fifth Third Bank	Fifth Third Bank, National Association	Fifth Third Bank
2015-03-18	230182	3055	Fifth Third Bank	Fifth Third Bank, National Association	Lloyds Bank
2015-03-18	230183	3055	Fifth Third Bank	Fifth Third Bank, National Association	Skandinaviska Enskilda Banken
2015-03-18	230184	3055	Fifth Third Bank	Fifth Third Bank, National Association	Deutsche Bank
2015-03-18	230185	3055	Fifth Third Bank	Fifth Third Bank, National Association	HSBC Bank
2015-03-18	235718	2865	Expobank	The Bank of New York Mellon Corp.	Turkiye Finans Katilim Bankasi A.S.
2015-03-18	240103	3319	Barclays	Barclays Plc	Ukrsibbank
2016-12-14	224119	2716	Hellenic Bank Public Company	Deutsche Bank AG	Bank VTB 24
2016-12-14	235191	3997	Falcon Private Bank Ltd	The Bank of New York Mellon Corp.	AS Meridian Trade Bank

by reading infos about our data in python, we get 988 transactions in all

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 988 entries, 0 to 81973
Data columns (total 16 columns):
time                     988 non-null datetime64[ns]
id                       988 non-null int64
icij_sar_id              988 non-null int64
originator_bank          988 non-null object
filer_org_name           988 non-null object
beneficiary_bank         988 non-null object
Transaction_amount_BTC   988 non-null float64
Transaction_amount_USD   988 non-null float64
Price                    988 non-null float64
```

and after calculating correlation between values like prices, BTC and USD transactions we get

```
#calculating data correlations between numerical values:
print("correlation between the transaction  BTC and the transaction USD is: ",
      df3["Transaction_amount_BTC"].corr(df3["Transaction_amount_USD"]))
print("correlation between transaction BTC and Price is: ", df3["Transaction_amount_BTC"].corr(df3["Price"]))
print("correlation between transaction BTC and Price is:", df3["Transaction_amount_USD"].corr(df3["Price"]))
```

```
correlation between the transaction  BTC and the transaction USD is:  0.8727264245733766
correlation between transaction BTC and Price is:  -0.2677107868620912
correlation between transaction BTC and Price is: 0.05337404962060785
```

for example we see that there is a strong correlation between the BTC and USD transactions equal to 0,87.