/\*

Written by Myriam Changoluisa

Term: Spring 2020

\*/

Importing Packages

import pandas as pd

import numpy as np

import scipy.stats as sp

import matplotlib.pyplot as plt

%matplotlib inline

import time

from datetime import datetime

from sklearn import linear\_model

Importing data

#Enddate : 2020 - 07 - 01

AHETPI = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\AHETPI\_quaterly.xlsx')

AHETPI.set\_index('Date', inplace=True)

EMRATIO = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\EMRATIO\_quaterly.xlsx', skiprows = range(1,65) )

EMRATIO.set\_index('Date', inplace=True)

POPTHM = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\POPTHM\_quaterly.xls', skiprows = range(1,21))

POPTHM.set\_index('Date', inplace=True)

UNRATE = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\UNRATE\_quaterly.xlsx', skiprows = range(1,65))

UNRATE.set\_index('Date', inplace=True)

#------------

TOTLQ = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\TOTLQ.xls')

TOTLQ.set\_index('Date', inplace=True)

W019RCQ027SBEA = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\W019RCQ027SBEA.xls', skiprows = range(1,20))

W019RCQ027SBEA.set\_index('Date', inplace=True)

#-------------

#Enddate : 2020 - 04 - 01

Government= pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\Goernment\_Tax\_receipts.xls', skiprows = range(1,25))

Government.set\_index('Date', inplace=True)

HNICTIQ027S= pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\HNICTIQ027S.xls', skiprows = range(1,70))

HNICTIQ027S.set\_index('Date', inplace=True)

NationalIncome = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\National Income.xls', skiprows = range(1,69))

NationalIncome.set\_index('Date', inplace=True)

GDI = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\GDI.xlsx')

GDI.set\_index('Date', inplace=True)

#-----------

A023RC1Q027SBEA = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\A023RC1Q027SBEA.xls', skiprows = range(1,69))

A023RC1Q027SBEA.set\_index('Date', inplace=True)

ASCOEPQ027S = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\ASCOEPQ027S.xls', skiprows = range(1,70))

ASCOEPQ027S.set\_index('Date', inplace=True)

ASTIWEQ027S = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\ASTIWEQ027S.xls', skiprows = range(1,21))

ASTIWEQ027S.set\_index('Date', inplace=True)

GDP = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\GDP.xls', skiprows = range(1,69))

GDP.set\_index('Date', inplace=True)

GNP = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\GNP.xls', skiprows = range(1,69))

GNP.set\_index('Date', inplace=True)

W006RC1Q027SBEA = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\_2\W006RC1Q027SBEA.xls', skiprows = range(1,69))

W006RC1Q027SBEA.set\_index('Date', inplace=True)

#-----------

#Enddate : 2019 - 10 - 01

FYGFD = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\FYGFD\_quaterly.xlsx', skiprows = range(1,65) )

FYGFD.set\_index('Date', inplace=True)

GINIALLRF= pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\GINIALLRF\_quaterly.xls', skiprows = range(1,65))

GINIALLRF.set\_index('Date', inplace=True)

#Enddate : 2016 - 10 - 01

IITTRHB = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\IITTRHB\_quaterly.xls', skiprows = range(1,65))

IITTRHB.set\_index('Date', inplace=True)

#Enddate : 2020 - 04 - 01 / start 2002 can not be predicted

NPPTTL = pd.read\_excel (r'C:\Users\mchangol\Desktop\Myriam\ml\_preprocesing\Quaterly\Quaterly\_data\NPPTTL\_Quaterly.xlsx')

NPPTTL.set\_index('Date', inplace=True)

Linear Regression

def linearFunctionToPredictValues(nameString):

y=np.array(nameString.values, dtype=float)

x=np.array(pd.to\_datetime(nameString.dropna()).index.values, dtype=float)

slope, intercept, r\_value, p\_value, std\_err =sp.linregress(x,y)

xf = np.linspace(min(x),max(x),100)

xf1 = xf.copy()

xf1 = pd.to\_datetime(xf1)

yf = (slope\*xf)+intercept

#print(' r = ', r\_value, '\n', 'p = ', p\_value, '\n', 's = ', std\_err, '\n','slope = ', slope, '\n', 'intercept = ', intercept )

#print(' slope = ', slope, '\n', 'intercept = ', intercept )

return slope, intercept

def createNewDates(inital, final, slope1, intercept1):

index = pd.date\_range(inital, final, 2)

index, len(index)

new\_arr = np.array(index.values, dtype = float)

yf\_new\_values = (slope1\*new\_arr)+intercept1

print('The new predicted values are:' , yf\_new\_values)

def createNewDates\_16(inital, final, slope1, intercept1):

index = pd.date\_range(inital, final, 16)

index, len(index)

new\_arr = np.array(index.values, dtype = float)

yf\_new\_values = (slope1\*new\_arr)+intercept1

print('The new predicted values are:' , yf\_new\_values)

def createNewDates\_4(inital, final, slope1, intercept1):

index = pd.date\_range(inital, final, 4)

index, len(index)

new\_arr = np.array(index.values, dtype = float)

yf\_new\_values = (slope1\*new\_arr)+intercept1

print('The new predicted values are:' , yf\_new\_values)

Prediction values

import datetime

slope , intercept = linearFunctionToPredictValues(AHETPI['AHETPI'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(EMRATIO['EMRATIO'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(POPTHM['POPTHM'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(UNRATE['UNRATE'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(TOTLQ['TOTLQ'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(W019RCQ027SBEA['W019RCQ027SBEA'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(A023RC1Q027SBEA['A023RC1Q027SBEA'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(ASCOEPQ027S['ASCOEPQ027S'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(ASCOEPQ027S['ASCOEPQ027S'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(ASTIWEQ027S['ASTIWEQ027S'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(GDP['GDP'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(GNP['GNP'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(W006RC1Q027SBEA['W006RC1Q027SBEA'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(Government['Government'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(HNICTIQ027S['HNICTIQ027S'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(NationalIncome['NationalIncome'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(GDI['GDI'])

start1 = datetime.datetime(2020, 7, 1)

end1 = datetime.datetime(2020, 10, 1)

createNewDates(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(FYGFD['FYGFD'])

start = datetime.datetime(2020, 1, 1)

end = datetime.datetime(2020, 10, 1)

createNewDates\_4(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(GINIALLRF['GINIALLRF'])

start = datetime.datetime(2020, 1, 1)

end = datetime.datetime(2020, 10, 1)

createNewDates\_4(start1, end1, slope, intercept )

import datetime

slope , intercept = linearFunctionToPredictValues(IITTRHB['IITTRHB'])

start = datetime.datetime(2017, 1, 1)

end = datetime.datetime(2020, 10, 1)

createNewDates\_16(start1, end1, slope, intercept )

Concatenating DataFrames

# Place the DataFrames side by side

horizontal\_stack = pd.concat([AHETPI, EMRATIO, POPTHM, TOTLQ, W019RCQ027SBEA, A023RC1Q027SBEA, ASCOEPQ027S, ASTIWEQ027S, GDP, GNP, W006RC1Q027SBEA, UNRATE, Government, HNICTIQ027S, NationalIncome, FYGFD, IITTRHB, GDI,GINIALLRF ], axis=1)

horizontal\_stack

newdata = horizontal\_stack.dropna()

newdata

newdata.to\_csv('preprocessdata\_2.csv', index = True)