import matplotlib.pyplot as plt import json import os from datetime import datetime, date, timedelta from matplotlib.dates import date2num import yfinance as yf import sys import glob import numpy as np from collections import Counter # path='../Febrian/Bang Nino/Samples/NEE/2020-05-08/'

def django_pass(func_stock, func_param): # my_args = sys.argv[1:] my_args = func_stock tick = my_args # filter=int(my_args[1]) filter = func_param visual=50 print(tick) # path='./Samples/'+tick+'/2020-12-07/' path='/home/kraken/Stock/v04/Samples/'+tick+'/' all_dirs = glob.glob(path+"*") print(path) print(all_dirs) latest_dir = max(all_dirs, key=os.path.getmtime) print("Latest Data = ", latest_dir) latest_dir = latest_dir +"/" listsymbol = os.listdir(latest_dir+'/') print(listsymbol)

```
with open(latest_dir+listsymbol[0],'r') as f:
    print(latest_dir+listsymbol[0])
    hasil=json.load(f)
    # print(hasil)
    print(len(hasil[0]))
    print(len(hasil[1]))
    print(len(hasil[2]))
    akhir=datetime.strptime(str(hasil[2][-1]),'%Y-%m-%d').date()
    print("Current Log : ",akhir)
    extend1=[akhir+timedelta(days=x) for x in range(1,50,1)]
    extend=[]

    for x in extend1:
       if x.weekday() not in [5,6]:
           extend.append(x)

    extend=extend[:14]
    extend=[str(x) for x in extend]
    print("ASasdasdsa",extend)
    # date=date2num(date)
    date=hasil[2]
    plt.plot(hasil[1],color='grey',linewidth=3, linestyle='--', marker='x', alpha=0.8, label
    plt.plot(hasil[2],hasil[1][0:-7],color='b',linewidth=2)
    # plt.show()

symbol=yf.Ticker(tick)
symbol=symbol.history(start=akhir,end=akhir+timedelta(days=30),interval='1d')
# print(symbol)
symbol=symbol.drop(symbol.index[0])
symbol=symbol.drop(symbol.index[0])
symbol=symbol['Close'][0:14].tolist()
# plt.plot(extend,symbol,color='g',linewidth=10)
avg=[]
gap_avg = []
```

```
for ex,x in enumerate(listsymbol):
    gap_val = []
    with open(latest_dir+x) as f:
        print("##################################################")
        print(x)
        hasil=json.load(f)
        # a=hasil[0][-14:][:14]
        a = hasil[0][-14:][:7]
        b=hasil[1][-7:]
        print("A & B Temp :")
        print(a)
        print(b)
        count=0
        print("Predict - Real ")
        for x in range(len(b)):
            print("%.2f" % (a[x]-b[x]))
            if (b[x]-(filter/10) <= a[x] <= b[x]+(filter/10)):
                count=count+1
        if count>4:
            print("ACCEPTED . . .")
            for i in range(len(b)):
                gap_val.append(abs(a[i]-b[i]))
                print(len(gap_val))
            predict_val = hasil[0][-14:]
            avg.append(predict_val)
            # avg.append(hasil[0][-10][:14])
            # plt.plot(date+extend,hasil[0], label='Sample %s'%ex, alpha=0.3)
            gap_avg.append(gap_val)
            print()
        else:
            print("NOT ACCEPTED . . .")
            print()
# print(avg)
print("##################################################")
print(avg)
print(len(avg))
print(gap_avg)
print(len(gap_avg))
print("##################################################")
avg_total=[]
good_index=[]
for x in range(len(extend)):
    print("NOW ", x)
    temp=[]
    gap_temp = []
    pred_temp = []
```

```
    if x < 7:
        for a in range(len(gap_avg)):
            # print(gap_temp.size())
            gap_temp.append(gap_avg[a][x])
        print("Gap Temp")
        print(gap_temp)
        # for b in range(len(gap_avg)):
        good_index.append(gap_temp.index(min(gap_temp)))
    else:
        print("LOL")
        c = Counter(good_index)
        c = c.most_common(1)
        good_index.append(c[0][0])

    print("Good Index")
    print(len(good_index))
    print(good_index)
    print()
for c in range(len(extend)):
    print(c)
    temp.append(avg[int(good_index[c])][c])
    print(temp)
    # avg_total.append(sum(temp)/len(temp))
 # mean_pred = np.mean(temp)
 # print(mean_pred)
avg_total = temp

print(avg_total)
#   print(avg_total)

atas=[x+(1/visual*x) for x in avg_total]
bawah=[x-(1/visual*x) for x in avg_total]
print([hasil[1][0:-5][-1]]+bawah)
print([hasil[1][0:-5][-1]]+atas)
print([date[-1]]+extend)
plt.fill_between([date[-1]]+extend,[hasil[1][0:-5][-1]]+bawah,[hasil[1][0:-5][-1]]+atas,alph
plt.plot([date[-1]]+extend,[hasil[1][-5]]+avg_total,color='y',linewidth=3,label='Prediction'
plt.grid()
#   plt.show()

# print("Date Extended ",(date+extend))
symbol  =   yf.Ticker(tick)
symbol  =   symbol.history(start=akhir,end=akhir+timedelta(days=20),interval='1d')
print(symbol)
symbol  =   symbol.drop(symbol.index[0])
symbol  =   symbol.drop(symbol.index[0])
```

```
symbol  =   symbol['Close'][0:14].tolist()
# symbol  =   symbol['Close'].tolist()
#   print(symbol)
plt.plot(extend[0:len(symbol)],symbol,color='r',label='Actual',linewidth=2)
# plt.plot(hasil[1],color='r', linestyle='--', label='Confirmation',linewidth=2)
# plt.plot(hasil[2]+extend[0:14],hasil[0],color='b',label='Train',linewidth=2, alpha=0.4)
# plt.plot(symbol, label="Symbol Real")
# plt.plot(hasil[0], label="Real Predictioh")
#   print(symbol)
#   print(symbol)
# detail = str(akhir)+"\n"+"Prediction :"+str(avg_total[-1:])+"\n"+"Real : "
# plt.text(0.05, 120, detail, color='black', bbox=dict(facecolor='none', edgecolor='black',
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.01),
          fancybox=True, shadow=True, ncol=7)
plt.title(tick+" Date: "+str(hasil[2][-1])+" to "+str(extend[-1]))
#   plt.get_xaxis().set_ticks([])
plt.xticks([])
# plt.show()
# plt.xlim([len(date)+len(extend)-30, len(date)+len(extend)])
ret_data = plt.gcf()
return ret_data
```

def django_pass2(func_stock, func_param): # my_args = sys.argv[1:] my_args = func_stock tick = my_args # filter=int(my_args[1]) filter = func_param visual=50 print(tick) # path='./Samples/'+tick+'/2020-12-07/' path='/home/kraken/Stock/v04/Samples/'+tick+'/' all_dirs = glob.glob(path+"*“) print(path) print(all_dirs) latest_dir = max(all_dirs, key=os.path.getctime) print(”Latest Data = “, latest_dir) latest_dir = latest_dir +”/" listsymbol = os.listdir(latest_dir+'/') print(listsymbol)

```
with open(latest_dir+listsymbol[0],'r') as f:
    print(latest_dir+listsymbol[0])
    hasil=json.load(f)
    # print(hasil)
    print(len(hasil[0]))
    print(len(hasil[1]))
    print(len(hasil[2]))
    akhir=datetime.strptime(str(hasil[2][-1]),'%Y-%m-%d').date()
    print("Current Log : ",akhir)
    extend1=[akhir+timedelta(days=x) for x in range(1,50,1)]
    extend=[]

    for x in extend1:
       if x.weekday() not in [5,6]:
           extend.append(x)

    extend=extend[:14]
```

```
    extend=[str(x) for x in extend]
    print("ASasdasdsa",extend)
    # date=date2num(date)
    date=hasil[2]
    plt.plot(hasil[1],color='grey',linewidth=3, linestyle='--', marker='x', alpha=0.8, label
    plt.plot(hasil[2],hasil[1][0:-7],color='b',linewidth=2)
    # plt.show()

symbol=yf.Ticker(tick)
symbol=symbol.history(start=akhir,end=akhir+timedelta(days=30),interval='1d')
# print(symbol)
symbol=symbol.drop(symbol.index[0])
symbol=symbol.drop(symbol.index[0])
symbol=symbol['Close'][0:14].tolist()
# plt.plot(extend,symbol,color='g',linewidth=10)
avg=[]
gap_avg = []
for ex,x in enumerate(listsymbol):
    gap_val = []
    with open(latest_dir+x) as f:
        print("##################################################")
        print(x)
        hasil=json.load(f)
        # a=hasil[0][-14:][:14]
        a = hasil[0][-14:][:7]
        b=hasil[1][-7:]
        print("A & B Temp :")
        print(a)
        print(b)
        count=0
        print("Predict - Real ")
        for x in range(len(b)):
            print("%.2f" % (a[x]-b[x]))
            if (b[x]-(filter/10) <= a[x] <= b[x]+(filter/10)):
                count=count+1
        if count>4:
            print("ACCEPTED . . .")
            for i in range(len(b)):
                gap_val.append(abs(a[i]-b[i]))
                print(len(gap_val))
            predict_val = hasil[0][-14:]
            avg.append(predict_val)
            # avg.append(hasil[0][-10][:14])
            # plt.plot(date+extend,hasil[0], label='Sample %s'%ex, alpha=0.3)
            gap_avg.append(gap_val)
            print()
```

```
        else:
            print("NOT ACCEPTED . . .")
            print()
# print(avg)
print("################################################")
print(avg)
print(len(avg))
print(gap_avg)
print(len(gap_avg))
print("################################################")
avg_total=[]
good_index=[]
for x in range(len(extend)):
    print("NOW ", x)
    temp=[]
    gap_temp = []
    pred_temp = []
    if x < 7:
        for a in range(len(gap_avg)):
            # print(gap_temp.size())
            gap_temp.append(gap_avg[a][x])
        print("Gap Temp")
        print(gap_temp)
        # for b in range(len(gap_avg)):
        good_index.append(gap_temp.index(min(gap_temp)))
    else:
        print("LOL")
        c = Counter(good_index)
        c = c.most_common(1)
        good_index.append(c[0][0])

    print("Good Index")
    print(len(good_index))
    print(good_index)
    print()
for c in range(len(extend)):
    print(c)
    temp.append(avg[int(good_index[c])][c])
    print(temp)
    # avg_total.append(sum(temp)/len(temp))
 # mean_pred = np.mean(temp)
 # print(mean_pred)
avg_total = temp

print(avg_total)
#   print(avg_total)
```

```
atas=[x+(1/visual*x) for x in avg_total]
bawah=[x-(1/visual*x) for x in avg_total]
print([hasil[1][0:-5][-1]]+bawah)
print([hasil[1][0:-5][-1]]+atas)
print([date[-1]]+extend)
plt.fill_between([date[-1]]+extend,[hasil[1][0:-5][-1]]+bawah,[hasil[1][0:-5][-1]]+atas,alph
plt.plot([date[-1]]+extend,[hasil[1][-5]]+avg_total,color='y',linewidth=3,label='Prediction'
plt.grid()
#   plt.show()

# print("Date Extended ",(date+extend))
symbol  =   yf.Ticker(tick)
symbol  =   symbol.history(start=akhir,end=akhir+timedelta(days=20),interval='1d')
print(symbol)
symbol  =   symbol.drop(symbol.index[0])
symbol  =   symbol.drop(symbol.index[0])
symbol  =   symbol['Close'][0:14].tolist()
# symbol  =   symbol['Close'].tolist()
#   print(symbol)
plt.plot(extend[0:len(symbol)],symbol,color='r',label='Actual',linewidth=2)
# plt.plot(hasil[1],color='r', linestyle='--', label='Confirmation',linewidth=2)
# plt.plot(hasil[2]+extend[0:14],hasil[0],color='b',label='Train',linewidth=2, alpha=0.4)
# plt.plot(symbol, label="Symbol Real")
# plt.plot(hasil[0], label="Real Predictioh")
#   print(symbol)
#   print(symbol)
# detail = str(akhir)+"\n"+"Prediction :"+str(avg_total[-1:])+"\n"+"Real : "
# plt.text(0.05, 120, detail, color='black', bbox=dict(facecolor='none', edgecolor='black',
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.01),
          fancybox=True, shadow=True, ncol=7)
plt.title(tick+" Date: "+str(hasil[2][-1])+" to "+str(extend[-1]))
#   plt.get_xaxis().set_ticks([])
plt.xticks([])
# plt.show()
plt.xlim([len(date)+len(extend)-30, len(date)+len(extend)])
# ret_data = plt.gcf()
ret_data = []
ret_data.append(hasil[1])
ret_data.append(hasil[0])
return ret_data
```