```
globals().clear
import time
import math
import pandas as pd
import numpy as np
from matplotlib import pyplot
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
%matplotlib inline
from datetime import datetime
pd.options.display.max_rows = 5000
pd.options.display.max_columns = 500


from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_percentage_error
from sklearn.preprocessing import MinMaxScaler
from sklearn.ensemble import RandomForestRegressor

import tensorflow       as tf
import keras.optimizers as op
from tensorflow                     import keras
from tensorflow.keras.models        import Sequential
from tensorflow.keras.layers        import Dense
from tensorflow.keras.layers        import Dropout
from tensorflow.keras.optimizers    import Adam


# Load dataset
df = pd.read_excel('merged_onehot_test.xlsx')


t1=df
t1.index=t1['Date']
t1.drop(columns=t1.columns[0:2],
        axis=1,
        inplace=True)
t1.head()
```

Date	TSLA_close	TSLA_vol_4_ave	TSLA_vwap_4_ave	TSLA_trans_4_ave	nasx_clos
2020-06-01 10:30:00	176.600	6531560.00	174.371825	29927.25	952
2020-06-01 10:45:00	176.748	4872685.00	175.236475	22062.00	953
2020-06-01 11:00:00	176.560	3717613.75	175.730850	17452.00	953
2020-06-					

```
train_df = t1.loc['2020-06-01 10:30:00':'2021-12-31 16:00:00']
test_df = t1.loc['2022-01-01 09:30:00':'2022-05-27 16:00:00']
```

```
import time


start = time.time()
predictions = list()
a=1
count_time=list()
scale_X = MinMaxScaler()


for i in test_df['week_label'].unique():

    test_subset = test_df[test_df['week_label']==i]
    print(train_df.index[0])
    print(train_df.index[-1])
    print(test_subset.index[0])
    print(test_subset.index[-1])
    train_stand = train_df.copy()
    test_stand = test_subset.copy()

    st = time.time()


    X_train, y_train = train_stand.iloc[:,2:65], train_stand.iloc[:,0]
    X_train = scale_X.fit_transform(X_train)
    X_test, y_test = test_stand.iloc[:,2:65], test_stand.iloc[:,0]
    X_test = scale_X.transform(X_test)

    model = Sequential()
    model.add(Dense(50, activation = 'relu', input_dim = df.iloc[:,2:65].shape[1]))
    model.add(Dropout(0.25))
    model.add(Dense(1))
    opt = Adam(amsgrad = True, learning_rate= 0.001, beta_1 = 0.79, beta_2 = 0.999)
```

```
        opt = Adam(amsgrad = True, learning_rate= 0.001, beta_1 = 0.95, beta_2 = 0.999)
        model.compile(loss = 'mse', optimizer = opt)
        model.fit(X_train, y_train,epochs=30)
        y_hat = model.predict(X_test, verbose=False)
        predictions.append(y_hat)
        et = time.time()
        used_time=et-st
        count_time.append(used_time)

        train_df = train_df.append(test_df[test_df['week_label']==i])
        train_df=train_df.drop(train_df[train_df['week_label']==a].index)
        a+=1


        print(train_df.index[0])
        print(train_df.index[-1])
        print('Time taken:'+str(used_time))
        print('--------------------------')

    end = time.time()
    print("total used time"+str(end-start))
```

```
     2020-10-26 09:30:00
     2022-05-20 16:00:00
     2022-05-23 09:30:00
     2022-05-27 16:00:00
     Epoch 1/30
     911/911 [==============================] - 2s 2ms/step - loss: 528017.1875
     Epoch 2/30
     911/911 [==============================] - 2s 2ms/step - loss: 50467.5508
     Epoch 3/30
     911/911 [==============================] - 2s 2ms/step - loss: 13897.3330
     Epoch 4/30
     911/911 [==============================] - 2s 2ms/step - loss: 12247.0088
     Epoch 5/30
     911/911 [==============================] - 2s 2ms/step - loss: 11379.5850
     Epoch 6/30
     911/911 [==============================] - 2s 2ms/step - loss: 10676.0586
     Epoch 7/30
     911/911 [==============================] - 2s 2ms/step - loss: 10470.1582
     Epoch 8/30
     911/911 [==============================] - 2s 2ms/step - loss: 10168.1104
     Epoch 9/30
     911/911 [==============================] - 2s 2ms/step - loss: 9888.5742
     Epoch 10/30
     911/911 [==============================] - 2s 2ms/step - loss: 9925.6523
     Epoch 11/30
     911/911 [==============================] - 2s 2ms/step - loss: 9730.5957
     Epoch 12/30
     911/911 [==============================] - 2s 2ms/step - loss: 9872.0938
     Epoch 13/30
     911/911 [==============================] - 2s 2ms/step - loss: 9837.3076
     Epoch 14/30
     911/911 [==============================] - 2s 2ms/step - loss: 9811.0908
     Epoch 15/30
     911/911 [==============================] - 2s 2ms/step - loss: 9888.4316
```

```
911/911 [==============================] - 2s 2ms/step - loss: 9888.4510
Epoch 16/30
911/911 [==============================] - 2s 2ms/step - loss: 9743.6670
Epoch 17/30
911/911 [==============================] - 2s 2ms/step - loss: 9544.3330
Epoch 18/30
911/911 [==============================] - 2s 2ms/step - loss: 9431.6299
Epoch 19/30
911/911 [==============================] - 2s 2ms/step - loss: 9542.9834
Epoch 20/30
911/911 [==============================] - 2s 2ms/step - loss: 9490.0996
Epoch 21/30
911/911 [==============================] - 2s 2ms/step - loss: 9447.2090
Epoch 22/30
911/911 [==============================] - 2s 2ms/step - loss: 9361.6660
Epoch 23/30
911/911 [==============================] - 2s 2ms/step - loss: 9213.5117
Epoch 24/30
911/911 [==============================] - 2s 2ms/step - loss: 9234.2158
Epoch 25/30
911/911 [==============================] - 2s 2ms/step - loss: 9183.3047
Epoch 26/30
911/911 [==============================] - 2s 2ms/step - loss: 9238.8271
Epoch 27/30
911/911 [==============================] - 2s 2ms/step - loss: 9282.3086
Epoch 28/30
```

```
df_expe = pd.DataFrame(test_df.iloc[:,0])
pred_list= list()
for i in range(len(predictions)):
  pred_list=pred_list+predictions[i].tolist()


df_pred = pd.DataFrame(pred_list,index=test_df.index,columns= ['predict'])

df_Result = pd.concat([df_expe,df_pred],axis=1)

df_Result
```

2022-05-26 15:00:00	703.8000	714.705994
2022-05-26 15:15:00	704.0733	716.312683
2022-05-26 15:30:00	708.6400	718.247192
2022-05-26 15:45:00	707.5500	724.308838
2022-05-26 16:00:00	708.2200	714.839355
2022-05-27 09:30:00	735.1000	706.068665
2022-05-27 09:45:00	741.9400	715.123962
2022-05-27 10:00:00	740.2100	716.295105
2022-05-27 10:15:00	747.6900	721.522583
2022-05-27 10:30:00	750.4500	729.419373
2022-05-27 10:45:00	747.0500	729.239136
2022-05-27 11:00:00	751.5034	725.875122
2022-05-27 11:15:00	751.1136	730.733765
2022-05-27 11:30:00	751.4000	732.688904
2022-05-27 11:45:00	753.1350	734.221252
2022-05-27 12:00:00	753.3241	727.522644
2022-05-27 12:15:00	749.5700	730.936523
2022-05-27 12:30:00	750.3192	732.808472
2022-05-27 12:45:00	750.4500	735.355713
2022-05-27 13:00:00	753.1285	727.532654
2022-05-27 13:15:00	752.0998	731.034241
2022-05-27 13:30:00	750.2650	731.824707
2022-05-27 13:45:00	751.5700	733.955933
2022-05-27 14:00:00	751.8050	724.560486
2022-05-27 14:15:00	754.2700	728.041931
2022-05-27 14:30:00	755.1675	732.239990
2022-05-27 14:45:00	755.5150	736.375183
2022-05-27 15:00:00	756.9600	726.274658
2022-05-27 15:15:00	758.7000	733.281677
2022-05-27 15:30:00	757.6500	737.803711
2022-05-27 15:45:00	759.6600	743.044861
2022-05-27 16:00:00	759.5000	737.526794