Sehar Nawaz

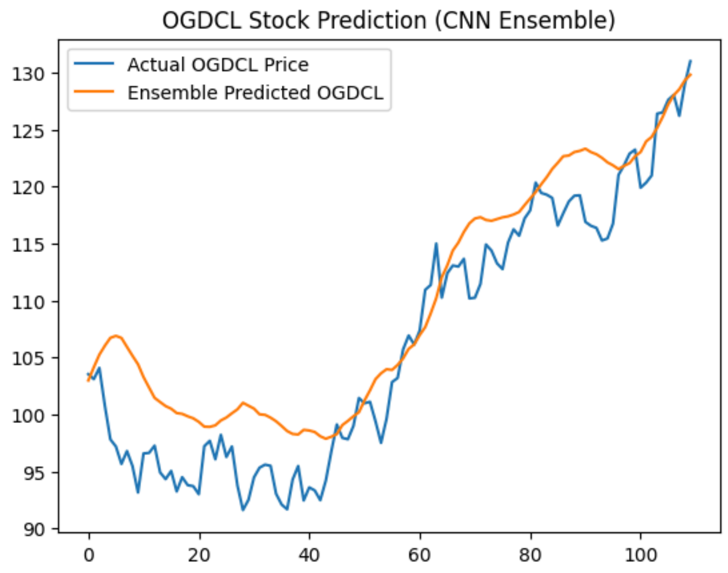
Rollno:13

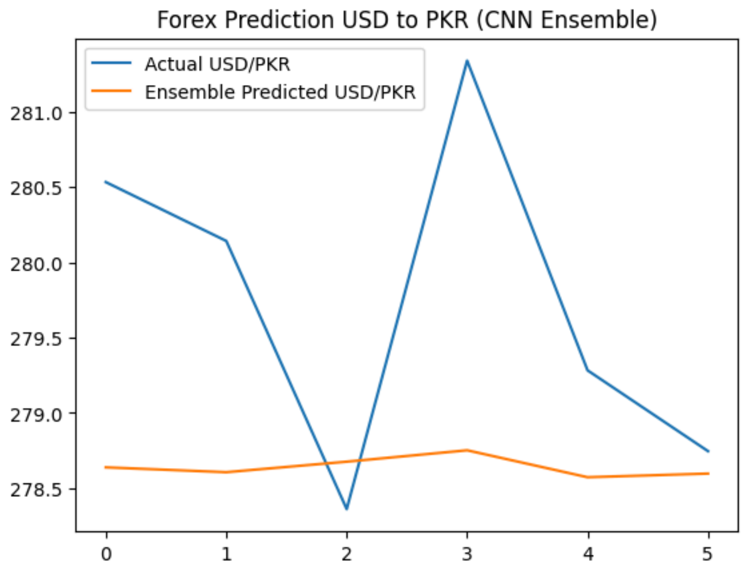
Assignment:no:03

```python
import numpy as np
import pandas as pd
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense
import matplotlib.pyplot as plt

np.random.seed(42)

dates = pd.date_range(start="2025-10-01", periods=120)
ogdcl_data = pd.DataFrame({
    "Date": dates,
    "Close": np.cumsum(np.random.normal(0.5, 2, 120)) + 90
})

forex_dates = pd.date_range(end="2025-11-20", periods=10)
forex_data = pd.DataFrame({
    "Date": forex_dates,
    "USD_PKR": np.random.uniform(278, 282, 10)
})

def prepare_data(series, window=10):
    X, y = [], []
    for i in range(len(series) - window):
        X.append(series[i:i+window])
        y.append(series[i+window])
    return np.array(X), np.array(y)

scaler_stock = MinMaxScaler()
stock_scaled = scaler_stock.fit_transform(ogdcl_data[["Close"]])
X_stock, y_stock = prepare_data(stock_scaled)
X_stock = X_stock.reshape(X_stock.shape[0], X_stock.shape[1], 1)

scaler_fx = MinMaxScaler()
fx_scaled = scaler_fx.fit_transform(forex_data[["USD_PKR"]])
X_fx, y_fx = prepare_data(fx_scaled, window=4)
X_fx = X_fx.reshape(X_fx.shape[0], X_fx.shape[1], 1)

def cnn_model(input_shape):
    model = Sequential()
    model.add(Conv1D(64, 3, activation='relu', input_shape=input_shape))
    model.add(MaxPooling1D(2))
    model.add(Flatten())
    model.add(Dense(50, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='adam', loss='mse')
    return model

# -------- ENSEMBLE LEARNING --------
n_models = 3

stock_preds = []
for _ in range(n_models):
    model = cnn_model((X_stock.shape[1], 1))
    model.fit(X_stock, y_stock, epochs=3, batch_size=16, verbose=0)
    stock_preds.append(model.predict(X_stock))

stock_pred_ensemble = np.mean(stock_preds, axis=0)

fx_preds = []
for _ in range(n_models):
    model = cnn_model((X_fx.shape[1], 1))
    model.fit(X_fx, y_fx, epochs=3, verbose=0)
    fx_preds.append(model.predict(X_fx))

fx_pred_ensemble = np.mean(fx_preds, axis=0)

stock_pred_real = scaler_stock.inverse_transform(stock_pred_ensemble)
y_stock_real = scaler_stock.inverse_transform(y_stock)

fx_pred_real = scaler_fx.inverse_transform(fx_pred_ensemble)
y_fx_real = scaler_fx.inverse_transform(y_fx)
```

```python

plt.figure()
plt.plot(y_stock_real, label="Actual OGDCL Price")
plt.plot(stock_pred_real, label="Ensemble Predicted OGDCL")
plt.legend()
plt.title("OGDCL Stock Prediction (CNN Ensemble)")
plt.show()

plt.figure()
plt.plot(y_fx_real, label="Actual USD/PKR")
plt.plot(fx_pred_real, label="Ensemble Predicted USD/PKR")
plt.legend()
plt.title("Forex Prediction USD to PKR (CNN Ensemble)")
plt.show()

```

```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/convolutional/base_conv.py:113: UserWarning: Do not pass an `input_
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 22ms/step
WARNING:tensorflow:5 out of the last 9 calls to <function TensorFlowTrainer.make_predict_function.<locals>.one_step_on_data_
1/4 ━━━━━━━━━━━━━━━━━━━━ 0s 57ms/stepWARNING:tensorflow:6 out of the last 12 calls to <function TensorFlowTrainer.make_predi
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 23ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 96ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 110ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 110ms/step
```


OGDCL Stock Prediction (CNN Ensemble)
Actual OGDCL Price
Ensemble Predicted OGDCL


Forex Prediction USD to PKR (CNN Ensemble)
Actual USD/PKR
Ensemble Predicted USD/PKR