



Name :- Venkatesh Rudrapati

SID:- 2461075

Subject :- Machine Learning in Finance

WEEK :- 10 Logbook

Code:-

```
[4] ✓ 2m
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import classification_report, accuracy_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

ask = pd.read_csv("XAUUSD_5 Mins_Ask_2024.01.01_2024.06.30.csv")
bid = pd.read_csv("XAUUSD_5 Mins_Bid_2024.01.01_2024.06.30.csv")

time_col = ask.columns[0]
df = pd.merge(ask, bid, on=time_col, suffixes=("_ask", "_bid"))
df = df.sort_values(time_col).reset_index(drop=True)

close_col = [c for c in df.columns if "close" in c.lower() and "ask" in c.lower()]
if len(close_col) == 0:
    close_col = [c for c in ask.columns if "close" in c.lower()]

close_col = close_col[0]

df["target"] = (df[close_col].shift(-1) > df[close_col]).astype(int)
df.dropna(inplace=True)

features = df.select_dtypes(include=[np.number]).drop(columns=["target"])
scaler = MinMaxScaler()
scaled = scaler.fit_transform(features)

SEQ = 50
X, y = [], []
for i in range(len(scaled) - SEQ):
    X.append(scaled[i:i+SEQ])
    y.append(df["target"].iloc[i+SEQ])

X = np.array(X)
y = np.array(y)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)
```

```

model = Sequential([
    LSTM(64, return_sequences=True, input_shape=(SEQ, X.shape[2])),
    Dropout(0.2),
    LSTM(32),
    Dense(16, activation='relu'),
    Dense(1, activation='sigmoid')
])

model.compile(optimizer="adam", loss="binary_crossentropy", metrics=["accuracy"])

Z = 5
epochs = Z + 30
batch_size = Z + 10

es = EarlyStopping(monitor="val_loss", patience=5, mode="min", restore_best_weights=True, verbose=1)
mc = ModelCheckpoint("best_lstm_model.keras", monitor="val_loss", save_best_only=True, mode="min", verbose=1)

history = model.fit(
    X_train, y_train,

    epochs=epochs,
    batch_size=batch_size,
    validation_split=0.2,
    callbacks=[es, mc],
    verbose=1
)

y_pred = (model.predict(X_test) > 0.5).astype(int)

print("Test Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))

plt.figure(figsize=(10,5))
plt.plot(history.history["accuracy"])
plt.plot(history.history["val_accuracy"])
plt.title("More Detailed Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.legend(["Train", "Val"])
plt.grid(True)
plt.show()

```

Output:-

```

... Epoch 1/35
/usr/local/lib/python3.12/dist-packages/keras/src/layers/rnn/rnn.py:199: UserWarning: Do not pass an `input_shape` to `Input` layers.
  super().__init__(**kwargs)
1502/1502 _____ 0s 9ms/step - accuracy: 0.5010 - loss: 0.6938
Epoch 1: val_loss improved from inf to 0.69311, saving model to best_lstm_model.keras
1502/1502 _____ 18s 11ms/step - accuracy: 0.5010 - loss: 0.6938 - val_accuracy: 0.5045 - val_loss: 0.6931
Epoch 2/35
1502/1502 _____ 0s 9ms/step - accuracy: 0.5108 - loss: 0.6930
Epoch 2: val_loss improved from 0.69311 to 0.69311, saving model to best_lstm_model.keras
1502/1502 _____ 15s 10ms/step - accuracy: 0.5108 - loss: 0.6930 - val_accuracy: 0.5045 - val_loss: 0.6931
Epoch 3/35
1501/1502 _____ 0s 9ms/step - accuracy: 0.4970 - loss: 0.6932
Epoch 3: val_loss did not improve from 0.69311
1502/1502 _____ 15s 10ms/step - accuracy: 0.4970 - loss: 0.6932 - val_accuracy: 0.5045 - val_loss: 0.6932
Epoch 4/35
1498/1502 _____ 0s 9ms/step - accuracy: 0.5105 - loss: 0.6930
Epoch 4: val_loss did not improve from 0.69311
1502/1502 _____ 15s 10ms/step - accuracy: 0.5105 - loss: 0.6930 - val_accuracy: 0.5045 - val_loss: 0.6931
Epoch 5/35
1501/1502 _____ 0s 9ms/step - accuracy: 0.5098 - loss: 0.6931
Epoch 5: val_loss improved from 0.69311 to 0.69310, saving model to best_lstm_model.keras
1502/1502 _____ 16s 10ms/step - accuracy: 0.5098 - loss: 0.6931 - val_accuracy: 0.5045 - val_loss: 0.6931

```

220/220 1s 5ms/step
Test Accuracy: 0.4960920846951826

	precision	recall	f1-score	support
0	0.00	0.00	0.00	3546
1	0.50	1.00	0.66	3491
accuracy			0.50	7037
macro avg	0.25	0.50	0.33	7037
weighted avg	0.25	0.50	0.33	7037

/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
_warn_prf(average, modifier, f'{metric.capitalize()} is', len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
_warn_prf(average, modifier, f'{metric.capitalize()} is', len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
_warn_prf(average, modifier, f'{metric.capitalize()} is', len(result))

