



Name:- Venkatesh Rudrapati

SID :- 2461075

Subject :- Machine Learning in Finance

WEEK 11 :- Logbook

Code :-

```
[9]
✓ 30s ▶ import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt

df = pd.read_csv("/content/credit_risk_dataset.csv")

target_col = "loan_status"

y = df[target_col]
X = df.drop(target_col, axis=1)

categorical_cols = X.select_dtypes(include=["object"]).columns
X = pd.get_dummies(X, columns=categorical_cols, drop_first=True)
```

```

if y.dtype == "object":
    y = y.astype("category").cat.codes

X = X.values
y = y.values

scaler = MinMaxScaler()
X = scaler.fit_transform(X)

X = X.reshape(X.shape[0], 1, X.shape[1])

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

timesteps = X_train.shape[1]
features = X_train.shape[2]

model = Sequential()
model.add(LSTM(64, input_shape=(timesteps, features)))

model.add(Dropout(0.3))
model.add(Dense(32, activation="relu"))
model.add(Dropout(0.3))
model.add(Dense(1, activation="sigmoid"))

model.compile(loss="binary_crossentropy", optimizer=Adam(1e-3), metrics=["accuracy"])

model.fit(X_train, y_train, epochs=10, batch_size=32, validation_split=0.2, verbose=1)

y_pred_proba = model.predict(X_test)

y_pred_LSTM = (y_pred_proba >= 0.5).astype(int)

global lstm_acc, lstm_prec, lstm_rec, lstm_f1, lstm_auc

lstm_acc = np.round(accuracy_score(y_test, y_pred_LSTM) * 100, 2)
lstm_prec = np.round(precision_score(y_test, y_pred_LSTM, average='weighted') * 100, 2)
lstm_rec = np.round(recall_score(y_test, y_pred_LSTM, average='weighted') * 100, 2)
lstm_f1 = np.round(f1_score(y_test, y_pred_LSTM, average='weighted') * 100, 2)
lstm_auc = np.round(roc_auc_score(y_test, y_pred_LSTM) * 100, 2)

print("\nLSTM Model Metrics:")
print(f"Accuracy: {lstm_acc}")
print(f"Precision: {lstm_prec}")
print(f"Recall: {lstm_rec}")
print(f"F1-Score: {lstm_f1}")
print(f"AUC: {lstm_auc}")

```

OUTPUT:-

```
*** /usr/local/lib/python3.12/dist-packages/keras/src/layers/rnn/rnn.py:199: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a
    super().__init__(**kwargs)
Epoch 1/10
652/652 ————— 6s 5ms/step - accuracy: 0.7809 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 2/10
652/652 ————— 4s 4ms/step - accuracy: 0.7872 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 3/10
652/652 ————— 2s 3ms/step - accuracy: 0.7833 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 4/10
652/652 ————— 2s 3ms/step - accuracy: 0.7861 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 5/10
652/652 ————— 2s 3ms/step - accuracy: 0.7855 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 6/10
652/652 ————— 3s 5ms/step - accuracy: 0.7826 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 7/10
652/652 ————— 2s 3ms/step - accuracy: 0.7866 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 8/10
652/652 ————— 2s 3ms/step - accuracy: 0.7835 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 9/10
652/652 ————— 3s 3ms/step - accuracy: 0.7916 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
Epoch 10/10
652/652 ————— 2s 3ms/step - accuracy: 0.7807 - loss: nan - val_accuracy: 0.7763 - val_loss: nan
204/204 ————— 1s 3ms/step

LSTM Model Metrics:
Accuracy: 77.83
Precision: 60.57
Recall: 77.83
F1-Score: 68.12
AUC: 50.0
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defined and being s
    _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```