```
                ^^^^^^
FileNotFoundError: [Errno 2] No such file or directory: 'User_product_purchase_details_p2.csv'
PS C:\python\mock1> & c:/python/mock1/venv/Scripts/python.exe "c:/python/mock1/vivek_kumar_milestone 1/weekly2.py"
2025-11-22 16:20:29.200909: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors
from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-11-22 16:20:31.164670: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors
from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
WARNING:tensorflow:From C:\python\mock1\venv\Lib\site-packages\tf_keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses
.sparse_softmax_cross_entropy instead.

Dataset shape after merge: (550068, 12)

First few rows:
   User_ID Product_ID City_Category Stay_In_Current_City_Years  Marital_Status  Product_Category_1  Product_Category_2  Product_Category_3  Purchase Gender   Age  Occupation
0  1000001  P00069042             A                          2               0                   3                 NaN                 NaN      8370      F  0-17          10
1  1000001  P00248942             A                          2               0                   1                 6.0                14.0     15200      F  0-17          10
2  1000001  P00087842             A                          2               0                  12                 NaN                 NaN      1422      F  0-17          10
3  1000001  P00085442             A                          2               0                  12                14.0                 NaN      1057      F  0-17          10
4  1000002  P00285442             C                         4+               0                   8                 NaN                 NaN      7969      M   55+          16

Target distribution:
High_Value_Purchase
0    360529
1    189539
Name: count, dtype: int64

Missing values before filling:
City_Category                     0
Stay_In_Current_City_Years        0
Marital_Status                    0
Product_Category_1                0
Product_Category_2           173638
Product_Category_3           383247
Purchase                          0
Gender                            0
Age                               0
Occupation                        0
High_Value_Purchase               0
dtype: int64

Dataset shape after encoding: (550068, 20)
```

Dataset shape after encoding: (550068, 20)

Column names after encoding:
['Product_Category_1', 'Product_Category_2', 'Product_Category_3', 'Purchase', 'Occupation', 'High_Value_Purchase', 'Gender_M', 'Age_18-25', 'Age_26-35', 'Age_36-45', 'Age_46-50', 'Age_
51-55', 'Age_55+', 'City_Category_B', 'City_Category_C', 'Stay_In_Current_City_Years_1', 'Stay_In_Current_City_Years_2', 'Stay_In_Current_City_Years_3', 'Stay_In_Current_City_Years_4+',
 'Marital_Status_1']

Training set size: 440054
Test set size: 110014


==================================================
LOGISTIC REGRESSION MODEL
==================================================


LR Accuracy: 0.7661752140636646

Confusion Matrix:
[[64702  7404]
 [18320 19588]]

Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.90      0.83     72106
           1       0.73      0.52      0.60     37908

    accuracy                           0.77    110014
   macro avg       0.75      0.71      0.72    110014
weighted avg       0.76      0.77      0.75    110014


Top 10 Most Important Features:
            feature  importance
0   Product_Category_1    0.845970
2   Product_Category_3    0.430037
4            Gender_M    0.110769
12    City_Category_C    0.109819
7           Age_36-45    0.076448
6           Age_26-35    0.072334
9           Age_51-55    0.064008

```
==================================================
MLP NEURAL NETWORK MODEL
==================================================
WARNING:tensorflow:From C:\python\mock1\venv\Lib\site-packages\tf_keras\src\backend.py:873: The name tf.get_default_graph is deprecated. Please use tf.compat.v1.get_default_graph instead.

2025-11-22 16:20:35.635632: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX512F AVX512_VNNI AVX512_BF16 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:tensorflow:From C:\python\mock1\venv\Lib\site-packages\tf_keras\src\optimizers\__init__.py:317: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.


Model Architecture:
Model: "sequential"

_____
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 64)                1216

 dense_1 (Dense)             (None, 32)                2080

 dense_2 (Dense)             (None, 1)                 33

=================================================================
Total params: 3329 (13.00 KB)
Trainable params: 3329 (13.00 KB)
Non-trainable params: 0 (0.00 Byte)
_____


Training MLP...
Epoch 1/20
WARNING:tensorflow:From C:\python\mock1\venv\Lib\site-packages\tf_keras\src\utils\tf_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From C:\python\mock1\venv\Lib\site-packages\tf_keras\src\engine\base_layer_utils.py:384: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.

11002/11002 [==============================] - 9s 768us/step - loss: 0.3943 - accuracy: 0.8309 - val_loss: 0.3182 - val_accuracy: 0.8583
Epoch 2/20
```

```
11002/11002 [==============================] - 9s 768us/step - loss: 0.3943 - accuracy: 0.8309 - val_loss: 0.3182 - val_accuracy: 0.8583
Epoch 2/20
11002/11002 [==============================] - 8s 767us/step - loss: 0.2997 - accuracy: 0.8785 - val_loss: 0.3022 - val_accuracy: 0.8858
Epoch 3/20
11002/11002 [==============================] - 8s 752us/step - loss: 0.2907 - accuracy: 0.8834 - val_loss: 0.2835 - val_accuracy: 0.8839
Epoch 4/20
11002/11002 [==============================] - 8s 752us/step - loss: 0.2846 - accuracy: 0.8850 - val_loss: 0.2778 - val_accuracy: 0.8868
Epoch 5/20
11002/11002 [==============================] - 9s 790us/step - loss: 0.2783 - accuracy: 0.8868 - val_loss: 0.2722 - val_accuracy: 0.8911
Epoch 6/20
11002/11002 [==============================] - 9s 779us/step - loss: 0.2715 - accuracy: 0.8901 - val_loss: 0.2765 - val_accuracy: 0.8948
Epoch 7/20
11002/11002 [==============================] - 8s 748us/step - loss: 0.2669 - accuracy: 0.8932 - val_loss: 0.2675 - val_accuracy: 0.8959
Epoch 8/20
11002/11002 [==============================] - 8s 751us/step - loss: 0.2642 - accuracy: 0.8943 - val_loss: 0.2618 - val_accuracy: 0.8928
Epoch 9/20
11002/11002 [==============================] - 9s 773us/step - loss: 0.2627 - accuracy: 0.8943 - val_loss: 0.2569 - val_accuracy: 0.8974
Epoch 10/20
11002/11002 [==============================] - 9s 788us/step - loss: 0.2614 - accuracy: 0.8945 - val_loss: 0.2585 - val_accuracy: 0.8963
Epoch 11/20
11002/11002 [==============================] - 7s 674us/step - loss: 0.2603 - accuracy: 0.8950 - val_loss: 0.2568 - val_accuracy: 0.8955
Epoch 12/20
11002/11002 [==============================] - 8s 705us/step - loss: 0.2587 - accuracy: 0.8955 - val_loss: 0.2574 - val_accuracy: 0.8964
Epoch 13/20
11002/11002 [==============================] - 8s 690us/step - loss: 0.2576 - accuracy: 0.8961 - val_loss: 0.2554 - val_accuracy: 0.8974
Epoch 14/20
11002/11002 [==============================] - 8s 705us/step - loss: 0.2567 - accuracy: 0.8964 - val_loss: 0.2545 - val_accuracy: 0.8976
Epoch 15/20
11002/11002 [==============================] - 8s 698us/step - loss: 0.2559 - accuracy: 0.8970 - val_loss: 0.2541 - val_accuracy: 0.8977
Epoch 16/20
11002/11002 [==============================] - 8s 708us/step - loss: 0.2555 - accuracy: 0.8969 - val_loss: 0.2527 - val_accuracy: 0.8983
Epoch 17/20
11002/11002 [==============================] - 8s 767us/step - loss: 0.2547 - accuracy: 0.8969 - val_loss: 0.2565 - val_accuracy: 0.8977
Epoch 18/20
11002/11002 [==============================] - 8s 734us/step - loss: 0.2548 - accuracy: 0.8970 - val_loss: 0.2515 - val_accuracy: 0.8986
Epoch 19/20
11002/11002 [==============================] - 8s 695us/step - loss: 0.2541 - accuracy: 0.8971 - val_loss: 0.2519 - val_accuracy: 0.8990
Epoch 20/20
11002/11002 [==============================] - 8s 700us/step - loss: 0.2540 - accuracy: 0.8974 - val_loss: 0.2571 - val_accuracy: 0.8950

MLP Test Accuracy: 0.8948
```

Python
bash

Activate Windows
Go to Settings to activate Windows.

```
MLP Test Accuracy: 0.8948
MLP Test Loss: 0.2570
3438/3438 [==============================] - 1s 409us/step
3438/3438 [==============================] - 2s 450us/step

Confusion Matrix:
[[63512  8594]
 [ 2976 34932]]

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.88      0.92     72106
           1       0.80      0.92      0.86     37908

    accuracy                           0.89    110014
   macro avg       0.88      0.90      0.89    110014
weighted avg       0.90      0.89      0.90    110014


==================================================
MODEL COMPARISON - ALL METRICS
==================================================

   Metric  Logistic Regression  MLP Neural Network  Difference
 Accuracy             0.766175            0.894832    0.128656
Precision             0.725697            0.802555    0.076858
   Recall             0.516725            0.921494    0.404769
 F1-Score             0.603636            0.857922    0.254285
  ROC-AUC             0.780826            0.947777    0.166951


==================================================
SUMMARY
==================================================

✓ MLP performed better overall!
Reason: Neural networks can capture non-linear relationships
between features that logistic regression cannot model.

==================================================
```