

◆ Gemini

```
from google.colab import files

uploaded = files.upload()

for fn in uploaded.keys():
    print(f'User uploaded file "{fn}" with length {len(uploaded[fn])} bytes')
```



Choose Files Churn_Modelling.csv

- **Churn_Modelling.csv**(text/csv) - 684858 bytes, last modified: 8/10/2025 - 100% done
- Saving Churn_Modelling.csv to Churn_Modelling.csv
User uploaded file "Churn_Modelling.csv" with length 684858 bytes

```
import pandas as pd
```

```
# Read the CSV file
df = pd.read_csv("Churn_Modelling.csv")
```

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

```
df = pd.read_csv('Churn_Modelling.csv')
```

```
X = df.drop(columns=['RowNumber', 'CustomerId', 'Surname', 'Exited'])
y = df['Exited']
```


```
# Encode 'Gender'
le_gender = LabelEncoder()
X['Gender'] = le_gender.fit_transform(X['Gender'])
```

```
# One-hot encode 'Geography'
X = pd.get_dummies(X, columns=['Geography'], drop_first=True)
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```


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

```
model = Sequential()
model.add(Dense(units=16, activation='relu', input_dim=X_train.shape[1]))
```

 /usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:93: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential mode
super().__init__(activity_regularizer=activity_regularizer, **kwargs)

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

```
model.fit(X_train, y_train, batch_size=32, epochs=50, verbose=0)
```


 <keras.src.callbacks.history.History at 0x7d626cc55f10>

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

 63/63 ————— 0s 2ms/step

```
# Confusion matrix and accuracy
cm = confusion_matrix(y_test, y_pred_binary)
acc = accuracy_score(y_test, y_pred_binary)
```

```
print("Confusion Matrix:\n", cm)
print("Accuracy Score:", acc)
```

 Confusion Matrix:
[[1542 65]
 [209 184]]
Accuracy Score: 0.863

Double-click (or enter) to edit