# ĐỀ TÀI: DỰ ĐOÁN KHÁCH HÀNG RỜI BỎ NGÂN HÀNG (BANK CUSTOMER CHURN PREDICTION)

## Mô tả:

Xây dựng một hệ thống dự đoán khả năng khách hàng rời bỏ ngân hàng (churn prediction) bằng mô hình Deep Learning, bao gồm:  
- Thu thập và xử lý dữ liệu khách hàng từ tập dữ liệu có sẵn.  
- Xây dựng, huấn luyện và đánh giá mô hình mạng neural (MLP - Multilayer Perceptron).  
- Thiết kế REST API đơn giản (bằng FastAPI) để dự đoán churn từ thông tin đầu vào.  
- Xây dựng giao diện trực quan bằng Streamlit cho phép người dùng nhập dữ liệu và xem kết quả dự đoán.

## Nhiệm vụ của người thực hiện:

1. Xử lý dữ liệu:  
 - Đọc dữ liệu từ tập tin CSV.  
 - Làm sạch dữ liệu, mã hóa biến phân loại, chuẩn hóa dữ liệu số.  
 - Phân chia tập dữ liệu huấn luyện và kiểm thử.  
  
2. Xây dựng mô hình Deep Learning:  
 - Thiết kế kiến trúc mạng MLP (nhiều lớp ẩn, hàm kích hoạt ReLU, dropout để chống overfitting).  
 - Huấn luyện mô hình với TensorFlow hoặc PyTorch.  
 - Đánh giá hiệu suất mô hình với độ chính xác, độ nhạy, độ đặc hiệu, AUC.  
  
3. Triển khai REST API:  
 - Dùng FastAPI để xây dựng API cho phép gửi dữ liệu khách hàng và nhận lại kết quả dự đoán.  
  
4. Xây dựng dashboard:  
 - Sử dụng Streamlit để thiết kế giao diện đơn giản nhập liệu và hiển thị kết quả dự đoán churn.

## Cấu trúc thư mục dự án gợi ý:

bank\_churn\_prediction/  
├── data/  
│ └── churn\_data.csv  
├── model/  
│ ├── train\_model.py  
│ ├── predict.py  
│ └── model.h5  
├── api/  
│ ├── main.py  
│ └── requirements.txt  
├── dashboard/  
│ └── app.py  
└── README.md

## Công cụ và thư viện sử dụng:

- Python 3.10  
- Pandas, NumPy (xử lý dữ liệu)  
- TensorFlow / Keras (xây dựng mô hình)  
- scikit-learn (đánh giá mô hình)  
- FastAPI (xây dựng REST API)  
- Streamlit (xây dựng giao diện người dùng)

## Gợi ý tập dữ liệu:

- Bank Customer Churn Dataset – https://www.kaggle.com/datasets/shubhendra7/customer-churn-prediction

## Chi tiết về Mô hình Deep Learning

### 1. Mục tiêu của mô hình:

Dự đoán xác suất khách hàng rời bỏ ngân hàng (churn) dựa trên các đặc trưng đầu vào như độ tuổi, giới tính, số dư tài khoản, sản phẩm ngân hàng sử dụng, số lần tương tác, v.v.

### 2. Lý do chọn Deep Learning:

- Mô hình Multilayer Perceptron (MLP) có khả năng học mối quan hệ phi tuyến giữa các biến đầu vào và churn.  
- Có thể mở rộng với nhiều tầng, nhiều đơn vị ẩn để cải thiện hiệu suất mô hình.  
- Cho phép tích hợp kỹ thuật Dropout, Batch Normalization để cải thiện độ tổng quát.

### 3. Kiến trúc mô hình đề xuất:

Tầng | Loại | Chi tiết  
-------------|----------------------|-----------------------------  
Input | Dense | Số chiều = số đặc trưng đầu vào (ví dụ: 20)  
Hidden 1 | Dense + ReLU | 64 neurons  
Hidden 2 | Dropout | 50%  
Hidden 3 | Dense + ReLU | 32 neurons  
Hidden 4 | Batch Normalization | Ổn định phân phối dữ liệu giữa các tầng  
Output | Dense + Sigmoid | 1 neuron (dự đoán xác suất churn)

Ví dụ khởi tạo mô hình MLP bằng Keras:

from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense, Dropout, BatchNormalization  
  
model = Sequential()  
model.add(Dense(64, input\_dim=X\_train.shape[1], activation='relu'))  
model.add(Dropout(0.5))  
model.add(Dense(32, activation='relu'))  
model.add(BatchNormalization())  
model.add(Dense(1, activation='sigmoid'))  
  
model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy', 'AUC'])

### 4. Tiền xử lý dữ liệu:

- Mã hóa biến phân loại: One-Hot Encoding hoặc Label Encoding.  
- Chuẩn hóa dữ liệu số: Dùng StandardScaler hoặc MinMaxScaler.  
- Chia dữ liệu: 70% train – 30% test.  
  
from sklearn.preprocessing import StandardScaler  
scaler = StandardScaler()  
X\_train\_scaled = scaler.fit\_transform(X\_train)

### 5. Đánh giá mô hình:

- Accuracy, Precision, Recall, F1-score, AUC – ROC.  
from sklearn.metrics import classification\_report, roc\_auc\_score  
  
y\_pred = (model.predict(X\_test) > 0.5).astype("int32")  
print(classification\_report(y\_test, y\_pred))  
print("AUC:", roc\_auc\_score(y\_test, model.predict(X\_test)))

### 6. Kỹ thuật nâng cao:

- EarlyStopping.  
- GridSearchCV hoặc KerasTuner.  
- Class weighting hoặc oversampling.

### 7. Hiển thị kết quả:

import matplotlib.pyplot as plt  
  
plt.plot(history.history['accuracy'], label='Train Acc')  
plt.plot(history.history['val\_accuracy'], label='Val Acc')  
plt.legend()  
plt.title('Training vs Validation Accuracy')  
plt.show()