Assignment: Spotify Churn Prediction using ANN

Problem Statement

You will build an **Artificial Neural Network (ANN)** model to predict whether a Spotify user will churn (cancel subscription or stop using) based on user behavior and account features in the **Spotify Dataset for Churn Analysis**. Students will carry out a full pipeline: Exploratory Data Analysis (EDA), preprocessing, baseline ANN, optimized ANN and evaluation, and also deploy the best model into a **Streamlit app**.

Dataset Link

Spotify Dataset for Churn Analysis (nabihazahid) – Kaggle (Kaggle)

Guidelines for Students

Data Understanding

- Load the dataset and inspect its shape, column names, data types.
- Explore what features are available (user demographics, usage patterns, subscription plan, etc.).
- Identify the target variable (Churn) and the input features.
- Check for missing values, duplicates, or inconsistent data.

Exploratory Data Analysis (EDA)

- Plot the distribution of the **Churn** target (how many churn vs. non-churn users).

- Examine numerical feature distributions (e.g. usage metrics, time-based features, counts etc.).
- Compare churn vs non-churn for numerical features using boxplots or violin plots.
- For categorical features (subscription type, region, plan etc.) plot churn rates by category.
- Compute correlation matrix for numerical features; visualize with heatmap.
- Check for class imbalance (whether churned users are much fewer or many).

Preprocessing

- Deal with missing or invalid values.
- Encode categorical variables appropriately (Label Encoding or One-Hot Encoding).
- Possibly transform skewed numerical features (log or other transforms).
- Scale/normalize numerical features.
- Split into training & validation (and possibly test) sets (e.g., 80/20 or 70/30).

Model Building

Baseline ANN

- Simple architecture:
  - Input layer matching number of features
  - One or two hidden Dense layers with ReLU activation
  - Output layer with sigmoid activation (for binary classification)
- Loss: binary crossentropy; Optimizer: Adam (or similar)

- Train for a fixed number of epochs (say 30-50), with a batch size (e.g., 32)
- Evaluate training & validation performance (accuracy etc.)

Optimized ANN

- Increase depth / more hidden layers or more hidden units.
- Add regularization like Dropout and/or BatchNormalization.
- Use callbacks (e.g. EarlyStopping, ReduceLROnPlateau) to avoid overfitting.
- Hyperparameter tuning: different learning rates, batch sizes, number of epochs, layer sizes.
- Possibly try other architectures / techniques (e.g. different activation functions)

Evaluation

- Use metrics beyond accuracy: **Precision, Recall, F1-Score**, also **ROC-AUC**.
- Plot training vs. validation loss & accuracy curves.
- Compute confusion matrix.
- Show some example predictions: for some users, display the actual vs predicted churn (maybe for edge cases).
- Identify areas where model is doing poorly (which classes, which kinds of users) and discuss why.
- Compare the ANN's performance with at least one classical machine learning model (e.g. Logistic Regression, Random Forest).

Streamlit Application Deployment

- Choose the best performing model (could be the optimized ANN or a ML model if it's better) and save it (e.g. `model.save()` for Keras / TensorFlow, or using `pickle`/`joblib` for scikit-learn).

- Build a simple **Streamlit app** that:
    1. Provides input form(s) for relevant features (e.g. subscription plan, usage metrics, region etc.).
    2. Takes user input and preprocesses to the same format as your training data.
    3. Loads the saved model.
    4. Makes prediction: probability of churn, and outputs either “Likely to Churn” / “Not Likely to Churn” (or similar).
    5. Optionally shows extra information (confidence, maybe top features influencing prediction).
- Test the app with a few different hypothetical user profiles.

Expected Outcomes

- A clear EDA showing insights on what features correlate with churn in the Spotify dataset.
- A working ANN model that performs reasonably well, and comparisons with simpler ML models.
- Demonstrated understanding of preprocessing (handling missing data, encoding, scaling).
- Ability to mitigate overfitting, with regularization / callbacks / tuning.
- Interpretations of evaluation metrics, not just accuracy.
- A deployed Streamlit app that lets someone interact and predict churn for new user data.