🧠 Project Title: Predicting User Churn for Waze Using Ensemble Learning

🔍 1. Objective

The goal is to analyze user behavior data from the Waze app to identify patterns associated with monthly churn and build a machine learning model that can predict at-risk users. The outcome is to help Waze design proactive retention strategies.

📊 2. Dataset Overview

Source: /kaggle/input/wazedataset/waze_dataset.csv **Size:** 14,999 records × 13 features

Key Features:

- sessions, drives, total_sessions: User engagement
- n_days_after_onboarding: Days since sign-up
- driven_km_drives, duration_minutes_drives: Driving activity
- total_navigations_fav1, total_navigations_fav2: Preferred location usage
- activity_days, driving_days: Behavioral frequency
- device: Categorical (iPhone or Android)
- label: Target (churned or retained)

Missing Values:

- label had 700 missing entries → Imputed using mode.

🧹 3. Data Preprocessing

- **Imputation:** Filled missing label values using the most frequent class.
- **Encoding:** Converted label to binary and device to one-hot encoding.
- **Scaling:** Standardized continuous features (driven_km_drives, duration_minutes_drives, total_sessions) using StandardScaler.
- **Train-Test Split:** 80/20 stratified split to preserve churn ratio.

📈 4. Exploratory Data Analysis (EDA)

- **Churn Distribution:** Retained users dominate the dataset, confirming class imbalance.
- **Feature Correlation (post-label encoding):** Strong positive correlation with churn was observed in lower activity metrics: fewer sessions, fewer drives, shorter durations.

- **Histograms & Boxplots:** Churned users typically had significantly lower usage in key features like total_sessions, activity_days, and driven_km_drives.

⚙️ 5. Modeling

Models Implemented:

- **XGBoost**
- **Random Forest**
- **Ensemble Model (Soft Voting of XGBoost + RF)**

Class Imbalance Strategy:

- Applied **SMOTE** to oversample the minority class (churners).
- Also experimented with scale_pos_weight (XGBoost) and class_weight (Random Forest).

Hyperparameter Tuning:

- Used GridSearchCV for both models.
- Tuned parameters: n_estimators, max_depth, learning_rate, min_samples_split, and more.

🧪 6. Evaluation Metrics

Metrics Used:

- Accuracy
- Precision, Recall, F1-score
- Support for churn class (minority focus)

Results:

Model	**Accuracy**	**Churn Recall**	**Churn Precision**	**F1-score**
XGBoost	75.1%	39%	32%	35%
Random Forest	72.2%	52%	32%	40%
Ensemble	73.1%	46%	32%	38%

> 🎯 The ensemble model achieved the best trade-off between accuracy and churn recall, making it ideal for deployment.

🔬 7. Feature Importance & Explainability

- Used **SHAP** values to interpret model predictions.
- Most predictive features:
    - activity_days
    - duration_minutes_drives
    - total_sessions
    - drives
    - driven_km_drives