## Churn Prediction Exercise

In this exercise, you will build and tune six different classification models on a business dataset (Customer Churn), applying feature engineering, model evaluation, overfitting checks, and hyperparameter tuning.

### Dataset

We will use the **Telco Customer Churn** dataset from Kaggle, which contains **7,043** customer records and **21** columns, including the target variable Churn (Yes/No).

Notable features include (as detailed in EDA and feature descriptions):

- customerID, gender, SeniorCitizen, Partner, Dependents, tenure
- PhoneService, MultipleLines, InternetService
- OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies
- Contract, PaperlessBilling, PaymentMethod, MonthlyCharges, TotalCharges, Churn

Link: https:// https://www.kaggle.com/datasets/blastchar/telco-customer-churn

### Step 1 – Load & Explore Data

- Load dataset with Pandas.
- Explore distributions, missing values, and class imbalance.
- Identify categorical vs. numerical features.

### Step 2 – Data Preparation

1. Handle missing values.
2. Encode categorical features (one-hot encoding).
3. Scale numerical features (e.g., StandardScaler for Logistic Regression & k-NN).
4. Define X (features) and y (target).
5. Train/test split (80/20).

### Step 3 – Baseline Models

Train the following classifiers:

1. Logistic Regression
2. Decision Tree
3. Random Forest
4. XGBoost
5. CatBoost
6. k-Nearest Neighbors (k-NN)

### Step 4 – Model Evaluation

For each model, compute:

- Accuracy
- Precision, Recall, F1-score
- Confusion Matrix
- ROC Curve & AUC

Summarize results in a comparison table.

## Step 5 – Overfitting Analysis

- Compare train vs test performance (especially for Decision Tree, Random Forest, XGBoost, CatBoost).
- Tune max_depth, min_samples_split, n_estimators, etc., to reduce overfitting.

## Step 6 – Model Tuning

Use GridSearchCV / RandomizedSearchCV for hyperparameter optimization:

- Logistic Regression → C, penalty
- Decision Tree → max_depth, min_samples_split
- Random Forest → n_estimators, max_depth, max_features
- XGBoost → learning_rate, n_estimators, max_depth
- CatBoost → learning_rate, iterations, depth
- k-NN → n_neighbors, weights

Compare tuned vs baseline models.

## Step 7 – Interpret Results

- Which features are most predictive?
- Logistic Regression → coefficients
- Tree-based models → feature importance
- Which model performs best?
- Trade-offs: Which model balances recall (catching churners) vs precision (avoiding false alarms)?

## Step 8 – Extensions (Optional 🌟)

- Apply cross-validation and compare results.
- Experiment with Stacking Classifier (combine Logistic Regression + Random Forest + XGBoost).

## Tasks

- Load & explore dataset.
- Preprocess categorical & numerical features.
- Train Logistic Regression, Decision Tree, Random Forest, XGBoost, CatBoost, k-NN.
- Evaluate with Accuracy, Precision, Recall, F1, ROC/AUC.
- Diagnose and reduce overfitting.
- Perform hyperparameter tuning.
- Compare results in a table.
- Interpret key business insights.