### **1. Problem Statement:**

**Problem:**

* Predicting customer churn in a bank, based on features such as demographics, financial behavior, and account activity.

**How did you come up with this problem?**

* Customer churn is a critical problem as retaining existing customers is more cost-effective than acquiring new ones. With advancements in machine learning, banks can leverage data to predict churn and proactively engage with at-risk customers. The Kaggle dataset related to customer churn in banking inspired this problem statement.

### **2. Tech Stack:**

* **Python**: Because it has powerful libraries for handling data (like Pandas), performing calculations (NumPy), and visualizing results (Matplotlib).
* Python has a huge developer community. If I face any problems or challenges, there’s a vast amount of resources, documentation, and community forums available for help.
* Python offers all the tools I need for this project, from data cleaning to visualizing patterns in housing prices.
* **Jupyter Notebook** It allows interactive development, making it easy to see the results of each step of the data analysis process. You can quickly test and visualize your work.

### **3. High-Level Diagram:**

A high-level approach to solve this problem could be represented as follows:

* **Data Collection**: Download the dataset from Kaggle.
* **Data Preprocessing**: Handle missing values, categorical variables, and outliers.
* **Exploratory Data Analysis (EDA)**: Visualize data to understand patterns.
* **Feature Engineering**: Create new features or modify existing ones for model improvement.
* **Model Building**: Use machine learning algorithms such as Logistic Regression, Random Forest, or Neural Networks to build predictive models.
* **Model Evaluation**: Assess the model's performance using metrics like accuracy, precision, recall, and F1-score.
* **Prediction & Action**: Predict which customers are at risk of leaving and suggest strategies for retention.

### **5. Low-Level Explanation:**

**Code Breakdown:**

**Data Loading and Preprocessing**:  
python  
Copy code  
df = pd.read\_csv("Churn\_Modelling.csv")

df = df.dropna() # Handle missing values

df = pd.get\_dummies(df, columns=['Geography', 'Gender']) # Convert categorical to numeric

* The dataset is loaded and basic preprocessing steps such as removing missing values and converting categorical data into numerical values using one-hot encoding are performed.

**Exploratory Data Analysis**:  
python  
Copy code  
plt.figure(figsize=(10,6))

sns.countplot(x='Exited', data=df)

plt.show()

* Visualize the distribution of customers who have exited the bank to understand the imbalance in the dataset.

**Modeling**:  
python  
Copy code  
from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report

X = df.drop(['Exited'], axis=1)

y = df['Exited']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

model = RandomForestClassifier()

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

print(classification\_report(y\_test, y\_pred))

* This code splits the data into training and testing sets, trains a RandomForestClassifier, and evaluates its performance using classification metrics.

### **6. Challenges and Solutions:**

* **Data Imbalance**: One challenge with churn data is that the majority of customers tend to stay, leading to class imbalance. This was addressed by using techniques like oversampling, undersampling
* **Feature Selection**: Too many features can lead to overfitting. Feature selection was performed by analyzing feature importance from models like Random Forest or by using domain knowledge.
* **Model Overfitting**: Some models may perform too well on the training data but poorly on unseen data. Regularization techniques and cross-validation were used to mitigate this.