PROJECT TITLE: Telecom Customer Churn Analysis

🎯 Project Objective

To analyze customer churn using demographic, geographic, account, and service information, and to predict future churners using machine learning.

🗃️ STEP 1: ETL Process in SQL Server

✅ Tools Used: SQL Server, SSMS

📥 Data Loading

- Created database: db_Churn
- Imported CSV file into stg_Churn (staging table) using the **Import Wizard**
- customer_id set as **Primary Key**, changed BIT columns to VARCHAR(50) to avoid import issues

🔍 Data Exploration

- Used SELECT ... GROUP BY queries to explore:
    - Gender distribution
    - Contract types
    - Churn status and total revenue by status
    - State-wise distribution
- Checked **null values** using conditional SUM(CASE WHEN ...)

🧹 Data Cleaning and Transformation

- Used ISNULL() to handle nulls (e.g., 'No', 'None', 'Others')
- Loaded cleaned data from stg_Churn into prod_Churn

```
SELECT ...

INTO prod_Churn

FROM stg_Churn

WHERE ...
```

👁️ Created SQL Views

- vw_ChurnData: only customers who "Churned" or "Stayed"
- vw_JoinData: customers who "Joined"

📊 STEP 2: Power BI Data Transformations

✅ Tools Used: Power BI Desktop

🛠️ Transformations

- Added columns:
    - Churn Status = 1 if "Churned", else 0
    - Monthly Charge Range: created bins like < 20, 20-50, etc.

🧾 Created Reference Tables

1. mapping_AgeGrp: Derived Age Group and AgeGrpSorting
2. mapping_TenureGrp: Derived Tenure Group and TenureGrpSorting
3. prod_Services: Unpivoted service-related columns into Services and Status

---

📐 STEP 3: Power BI Measures

Total Customers = COUNT(prod_Churn[Customer_ID])

New Joiners = CALCULATE(COUNT(prod_Churn[Customer_ID]), prod_Churn[Customer_Status] = "Joined")

Total Churn = SUM(prod_Churn[Churn Status])

Churn Rate = [Total Churn] / [Total Customers]

📈 STEP 4: Power BI Visualizations

🧾 Summary Page

- KPIs: Total Customers, New Joiners, Total Churn, Churn Rate%
- Demographic Analysis: Gender, Age Group

- Account Info: Payment Method, Contract, Tenure Group
- Geographic: Top 5 States by Churn Rate
- Churn Distribution: Churn Category and Reason (Tooltip)
- Services: Internet Type & % Churn by Service Usage

🧾 Churn Reason Page (Tooltip)

- Drill-through to see Churn Reason-wise churn count

🤖 STEP 5: Predict Customer Churn (Machine Learning)

✅ Tools Used: Jupyter Notebook (Python), Excel (Prediction Data), scikit-learn

🧪 Model Used: Random Forest Classifier

📦 Data Preparation

- Imported vw_ChurnData and vw_JoinData using Excel → "Prediction_Data.xlsx"
- Dropped irrelevant columns (Customer_ID, Churn_Category, Churn_Reason)
- Label Encoded categorical features using LabelEncoder
- Mapped Customer_Status: "Stayed" = 0, "Churned" = 1

✂️ Train/Test Split

```
X = data.drop('Customer_Status', axis=1)

y = data['Customer_Status']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

🎓 Model Training

```
from sklearn.ensemble import RandomForestClassifier

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

rf_model.fit(X_train, y_train)
```

📊 Model Evaluation

- Printed Confusion Matrix & Classification Report

- Plotted Feature Importance using sns.barplot()

🔮 Churn Prediction on New Data

- Loaded vw_JoinData from Excel
- Encoded same columns using saved label encoders
- Predicted with rf_model.predict()
- Filtered churners and saved output to Predictions.csv

📈 STEP 6: Power BI Dashboard for Predictions

📂 Imported Predictions.csv into Power BI

🧾 Measures

Count Predicted Churner = COUNT(Predictions[Customer_ID]) + 0

Title Predicted Churners = "COUNT OF PREDICTED CHURNERS : " & COUNT(Predictions[Customer_ID])

📊 Churn Prediction Page

- Right Grid: Customer ID, Monthly Charge, Total Revenue, Refunds, Referrals
- Demographics: Gender, Age Group, Marital Status
- Account Info: Payment Method, Contract, Tenure Group
- Geographic: State-wise churn predictions

✅ Final Outcome:

You successfully created a complete **end-to-end Churn Analysis Portfolio Project**, covering:

- ETL and SQL Data Engineering
- Power BI Dashboarding
- Predictive Modeling in Python
- Real-time Insight Generation & Visualization