Best Real-Time Data Science Project Using MLflow

Project: Customer Churn Prediction for a Subscription-Based Business

🔹 **Problem Statement:**
A company providing subscription-based services (e.g., Netflix, Amazon Prime, Spotify) wants to predict which customers are likely to **cancel their subscriptions** (churn). The goal is to build a **Customer Churn Prediction Model** that helps in **targeting customers with personalized offers** to retain them.

🔹 **Why This Project?**
✅ **Real-World Business Impact** – Helps businesses **reduce customer loss** and **increase revenue.**
✅ **End-to-End MLflow Integration** – Covers **data logging, model tracking, deployment, and monitoring.**
✅ **Industry-Relevant Techniques** – Includes **feature engineering, ML algorithms, and A/B testing.**

Project Workflow & MLflow Integration

1️⃣ Data Collection

- **Dataset:** Get a dataset with customer details, subscription history, and churn labels.
  - You can use:
    📌 **Kaggle's Customer Churn Dataset**: https://www.kaggle.com/blastchar/telco-customer-churn
    📌 **Synthetic Data Generation using Faker**
- **MLflow Logging:** Log dataset version

```
python
CopyEdit
import mlflow
mlflow.log_artifact("churn_data.csv")
```

2️⃣ Data Preprocessing & Feature Engineering

- Handling missing values, encoding categorical variables, and feature scaling.
- Feature selection (e.g., tenure, monthly charges, payment method).
- **MLflow Logging:** Log data preprocessing steps

```
python
CopyEdit
```

```
mlflow.log_param("missing_values_handled", True)
mlflow.log_param("feature_scaling", "MinMaxScaler")
```

3️⃣ Model Training & Experimentation

- Train different models:
  ✅ Logistic Regression
  ✅ Random Forest
  ✅ XGBoost
- **Hyperparameter Tuning**:
  - Test different parameters (e.g., learning rate, number of trees).
- **MLflow Logging:** Log experiments

```
python
CopyEdit
mlflow.log_metric("accuracy", accuracy_score(y_test, y_pred))
mlflow.log_metric("roc_auc", roc_auc_score(y_test, y_pred_proba))
```

4️⃣ Model Evaluation & Selection

- Compare models based on:
  📌 **Accuracy, Precision, Recall, F1-score**
  📌 **ROC-AUC Curve Analysis**
- **MLflow Logging:** Log model evaluation metrics

```
python
CopyEdit
mlflow.log_metric("precision", precision_score(y_test, y_pred))
mlflow.log_metric("recall", recall_score(y_test, y_pred))
```

5️⃣ Model Deployment using MLflow

- Register the best model in **MLflow Model Registry**
- Deploy the model as a REST API using **Flask or FastAPI**

```
bash
CopyEdit
# Save the best model
mlflow.sklearn.log_model(model, "churn_prediction_model")

# Register the model
mlflow.register_model("runs:/<run_id>/model", "Customer_Churn_Model")

# Serve the MLflow model
mlflow models serve -m models:/Customer_Churn_Model/Production -p 5001
```

6️⃣ Model Monitoring & Continuous Improvement

- Monitor real-time **churn rates and prediction accuracy**
- If model performance degrades, trigger **retraining with new data**

Project Code And Workflow

End-to-End Customer Churn Prediction Project with MLflow & MLOps

This project will cover **EDA (Exploratory Data Analysis), Model Training, Hyperparameter Tuning, Deployment, and MLOps (Model Versioning, Monitoring, and Retraining) using MLflow**.

📌 Steps Covered in This Project

✅ **Step 1:** Data Collection & EDA
✅ **Step 2:** Data Preprocessing & Feature Engineering
✅ **Step 3:** Model Training & Experimentation
✅ **Step 4:** Model Evaluation & Hyperparameter Tuning
✅ **Step 5:** Model Deployment (Flask + MLflow)
✅ **Step 6:** Model Monitoring & Retraining (MLOps)

📌 Code for End-to-End MLflow Project

🔹 Step 1: Load Libraries & Data

```
python
CopyEdit
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
import mlflow
```

```
import mlflow.sklearn
import warnings
warnings.filterwarnings("ignore")

# Load dataset
df = pd.read_csv("customer_churn.csv")

# Display dataset
print(df.head())
```

◆ Step 2: Exploratory Data Analysis (EDA)

```
python
CopyEdit
# Check for missing values
print(df.isnull().sum())

# Data Distribution
plt.figure(figsize=(10, 6))
sns.countplot(x='Churn', data=df, palette='Set1')
plt.title("Churn Distribution")
plt.show()

# Correlation Heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.show()
```

◆ Step 3: Data Preprocessing & Feature Engineering

```
python
CopyEdit
# Encode categorical variables
encoder = LabelEncoder()
df['Churn'] = encoder.fit_transform(df['Churn'])

# Convert categorical columns
cat_cols = ['gender', 'Partner', 'PaymentMethod']
df[cat_cols] = df[cat_cols].apply(encoder.fit_transform)

# Scale numerical features
scaler = StandardScaler()
num_cols = ['tenure', 'MonthlyCharges', 'TotalCharges']
df[num_cols] = scaler.fit_transform(df[num_cols])

# Splitting dataset
X = df.drop(columns=['Churn'])
y = df['Churn']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

◆ Step 4: Model Training & MLflow Experimentation

```
python
CopyEdit
# Start MLflow Tracking
mlflow.set_experiment("Customer Churn Prediction")

with mlflow.start_run():
    model = RandomForestClassifier(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    # Log Metrics
    acc = accuracy_score(y_test, y_pred)
    mlflow.log_metric("accuracy", acc)

    # Log Model
    mlflow.sklearn.log_model(model, "churn_model")

    print(f"Model Accuracy: {acc}")
```

◆ Step 5: Model Deployment using Flask & MLflow

```
python
CopyEdit
from flask import Flask, request, jsonify
import mlflow.sklearn

app = Flask(__name__)

# Load MLflow Model
model = mlflow.sklearn.load_model("mlruns/0/<run_id>/artifacts/churn_model")

@app.route('/predict', methods=['POST'])
def predict():
    data = request.json['data']
    prediction = model.predict([data])
    return jsonify({'churn_prediction': int(prediction[0])})

if __name__ == '__main__':
    app.run(port=5001)
```

◆ Step 6: Model Monitoring & Retraining (MLOps)

```
python
CopyEdit
import mlflow

def retrain_model(new_data):
    df = pd.read_csv(new_data)
    X_new = df.drop(columns=['Churn'])
    y_new = df['Churn']
```

```
    model.fit(X_new, y_new)

    with mlflow.start_run():
        mlflow.sklearn.log_model(model, "churn_model_v2")
        mlflow.log_metric("accuracy", accuracy_score(y_new,
model.predict(X_new)))

    print("Model retrained & logged successfully!")

# Automate retraining every month
import schedule
import time

schedule.every(30).days.do(retrain_model,
new_data="new_customer_churn_data.csv")

while True:
    schedule.run_pending()
    time.sleep(1)
```