MLOps Report : Bank Customer Churn Prediction Model

1. Introduction

- In this report, we detail the **MLOps** pipeline that has been applied to the **Bank Customer Churn Prediction** model. The purpose of this model is to predict whether a customer is likely to leave the bank (churn) or remain a customer. MLOps practices have been implemented to streamline the lifecycle of the model, from data collection and training to deployment, monitoring, and continuous improvement.
- **MLOps** (Machine Learning Operations) combines DevOps practices with machine learning to ensure that models are efficiently built, deployed, and monitored in a production environment. This ensures the continuous availability of high-quality models.

---

2. MLOps Pipeline for Customer Churn Prediction

2.1 Data Collection and Preparation

- The foundation of any machine learning model is the data. For this project, customer data was collected from the bank's customer database. The data includes:
  - Customer age
  - Account balance
  - CreditScore
  - EstimatedSalary
  - HasCcard
  - Exited
  - Credit score
  - Gender
  - Active member status
  - Number of products
  - Geography (Country)

The data was cleaned and preprocessed using **Pandas** and **Scikit-learn**:

- Missing values were handled.
- Categorical variables (e.g., Gender, Geography) were encoded using **One-Hot Encoding, and Label encoder**.
- Numerical features were scaled using **StandardScaler** to normalize the data.

2.2 Exploratory Data Analysis (EDA)

- To gain insights from the data, we performed **Exploratory Data Analysis (EDA)**:
    - We visualized the distribution of key features such as age, balance, and credit score.
    - We examined the relationships between different features (e.g., geography and churn, number of products and churn).
    - Correlation matrices were plotted to understand the relationship between features and the target variable (churn).

2.3 Model Selection and Training

The following models were initially considered:

- **Logistic Regression**
- **Random Forest**
- **SVC**
- **Knn**
- **Support Vector Machine (SVM)**
- **XGBoost** (The final model used)

We used **GridSearchCV** to tune hyperparameters and select the best model. **XGBoost** was selected as the best-performing model, based on:

- High **Precision**, **Recall**, and **F1-Score**.
- The ability to handle imbalanced datasets.
- Faster training times with large datasets.

2.4 Model Evaluation

The model's performance was evaluated using multiple metrics:

- **Accuracy**: To measure the overall correctness of the model.
- **Precision**: To ensure that the model predicts churn correctly without many false positives.
- **Recall**: To ensure the model identifies as many churned customers as possible.
- **AUC-ROC**: To evaluate the model's performance in distinguishing between churned and non-churned customers.

---

3. Model Deployment

3.1 Deployment Environment

The model was deployed as an API using **Streamlit**. This allows the bank's internal systems to send customer data to the API and receive churn predictions in real-time.

The deployment process involved:

- Containerizing the model using **MLflow** to ensure that the deployment environment is consistent across all platforms.
- Hosting the API on **Streamlit Cloud** for scalability and availability.

3.2 Continuous Integration and Continuous Deployment (CI/CD)

We implemented a CI/CD pipeline using **GitHub Actions** to automate:

- Code testing
- Model retraining with new data
- Deployment to production

This ensures that new updates, including model improvements, are automatically tested and deployed to production without downtime.

---

4. Model Monitoring and Maintenance

4.1 Monitoring the Model

To ensure the model performs well in production, we will use **Prometheus** and **Grafana** for monitoring:

- We track performance metrics such as prediction response times and model accuracy.
- Alerts are set up for when the model's performance drops or when there is a significant change in incoming data patterns (data drift).

4.2 Re-training the Model

The model is retrained periodically based on new data. If there is a decline in the model's performance or if data drift is detected, the model will be retrained with updated data.

We use **MLflow** to manage and track different model versions and experiments, ensuring that the best-performing model is always deployed.

---

5. Conclusion

After applying **MLOps** practices, we successfully deployed a robust model for predicting customer churn. The final model selected is **XGBoost**, which provided the best performance in terms of **Precision**, **Recall**, and **AUC-ROC**.

The model has been deployed into production and is now integrated into the bank's internal systems. It is monitored for performance and retrained periodically to ensure that it remains accurate and effective.

This approach ensures the bank can proactively identify customers who are at risk of leaving, allowing for targeted retention strategies and improved customer satisfaction.

---

6. Future Recommendations

- **Model Enhancement**: Consider implementing **deep learning** models if the data volume increases significantly, or using **ensemble models** to further improve performance.
- **Customer Segmentation**: Use clustering techniques to segment customers based on behavior and apply personalized retention strategies.
- **Real-time Monitoring**: Expand real-time monitoring to detect shifts in customer behavior instantly, allowing quicker interventions.

Acknowledgment

We would like to thank the bank's team for providing the necessary data and resources, and for their support throughout the MLOps implementation process.