BANK CUSTOMER CHURN PREDICTION

An AWS Powered End to End Prediction System

Presented by: Richik Ghosh (G31267506)
Shreya Sahay (G36642286)
Date: 11/19/2024

Cloud Computing • DATS 6450 10 • Final Project Presentation

Introduction

Context

In the competitive banking sector, customer churn directly impacts profitability as dissatisfied clients can easily switch to alternative providers. Predicting churn helps banks retain customers by proactively addressing issues and offering personalized services, ultimately enhancing customer loyalty and revenue.

Objective

This project focuses on predicting customer churn by analyzing features from a European bank dataset, including credit score, account balance, and demographic information. By leveraging **AutoML** and **AWS** services, the patterns and risk factors are identified that indicate a higher likelihood of customers discontinuing their memberships, enabling proactive retention strategies.

Dataset Description

- The dataset is sourced from Kaggle
- It has **13** features and **10000** rows
- Features present in the data are:
- Target label is **'Exited'** which indicates whether a customer churned or not

Customer ID: A unique identifier for each customer

Surname: The customer's surname or last name

Credit Score: A numerical value representing the customer's credit score

Age: The customer's age.

Balance: The customer's account balance

EstimatedSalary: The estimated salary of the customer

Geography: The country where the customer resides (France, Spain or Germany)

Gender: The customer's gender

Tenure: The number of years the customer has been with the bank

NumOfProducts: The number of bank products the customer uses

HasCrCard: Whether the customer has a credit card

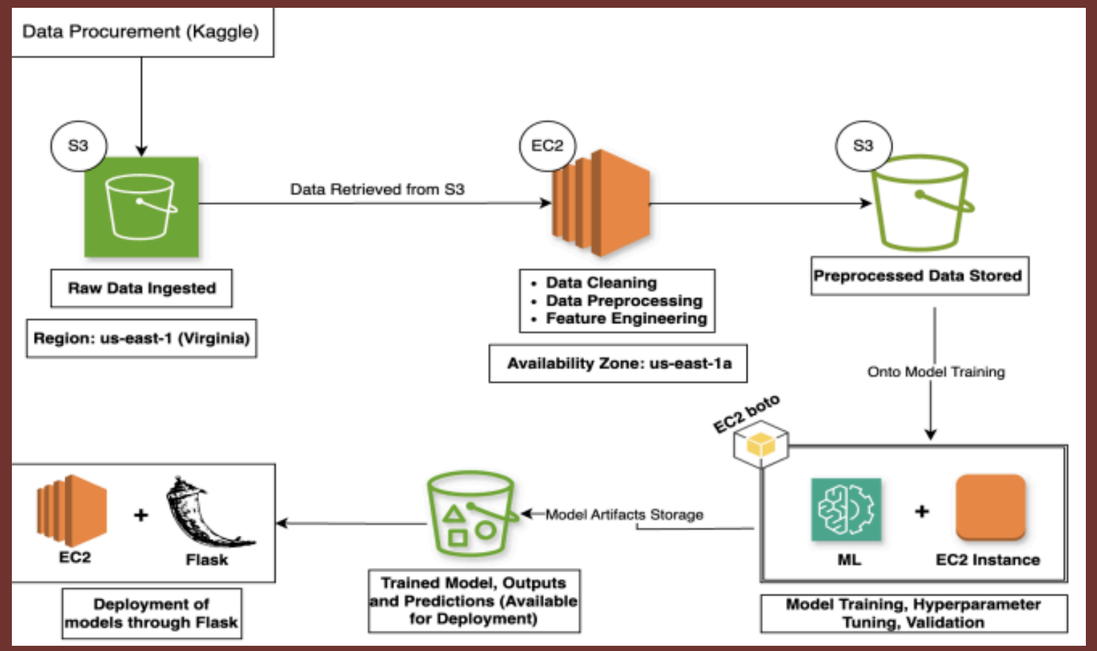
IsActiveMember: Whether the customer is an active member

Exited: Whether the customer has churned

Numerical

Categorical

Proposed Project Plan



Data Storage In S3

- The .csv file procured from Kaggle is stored in a new S3 bucket:

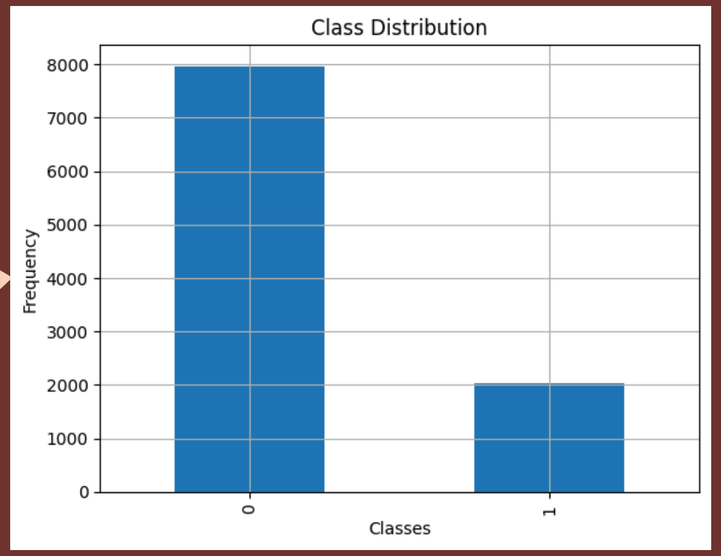
Name	Type	Last modified	Size	Storage
Customer-Churn-Records.csv	csv	October 23, 2024, 13:44:53 (UTC-04:00)	817.8 KB	Standard

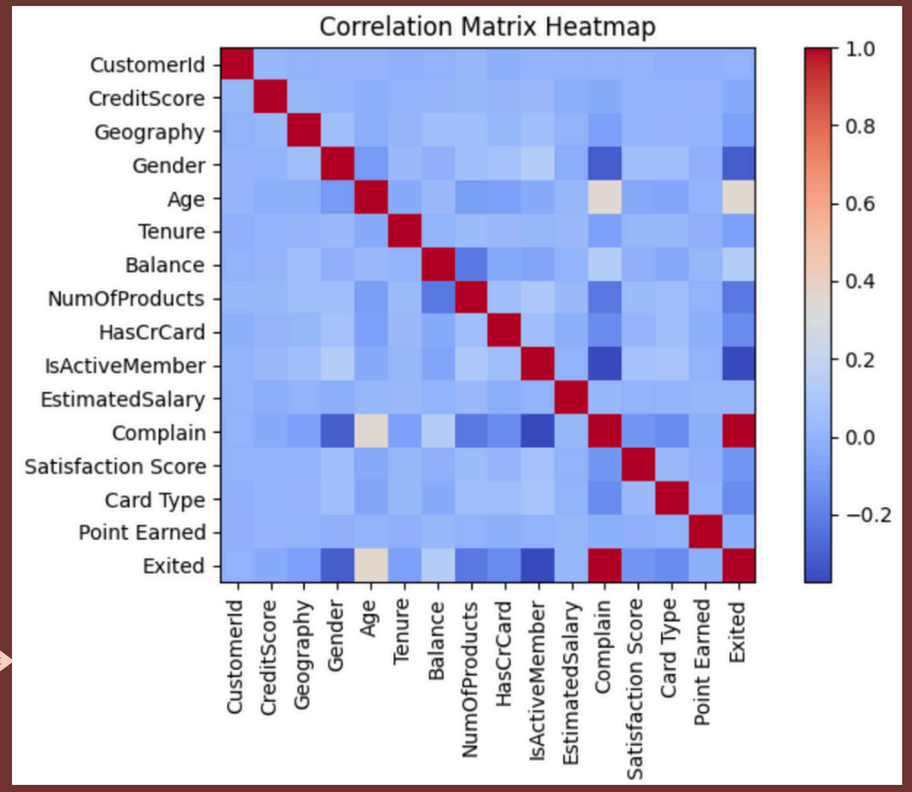
- Data is securely fetched from S3 into the VSCode environment using **boto3**

- Connected to **EC2** instance for model building

Data Preprocessing

- Dropped columns: **Surname** and **Customer ID**
- No **NULL** values found
- Addressed the class imbalance using **SMOTE**
- The **'Complaint'** column exhibited a high correlation with the target column. Since regularization techniques were ineffective in mitigating this relationship, the column was removed.





AutoML Application

H2O.ai

- Applied **H2O.ai** as the AutoML framework
- Used the most **significant features** derived from various statistical tests
- **max_models** = 20

```python
# Step 9: View the leaderboard for the best models
lb = aml.leaderboard
print("H2O AutoML Leaderboard:")
print(lb.head(rows=lb.nrows))
```

H2O AutoML Leaderboard:

model_id	auc	logloss	aucpr	mean_per_class_error	rmse	mse
GBM_4_AutoML_3_20241110_193552	0.944918	0.303028	0.946165	0.122581	0.302862	0.0917252
GBM_grid_1_AutoML_3_20241110_193552_model_2	0.943727	0.309454	0.943887	0.128718	0.305292	0.0932034
GBM_3_AutoML_3_20241110_193552	0.942307	0.307585	0.943232	0.127065	0.30627	0.0938012
XGBoost_grid_1_AutoML_3_20241110_193552_model_5	0.940567	0.320336	0.941014	0.131786	0.309545	0.0958179
XGBoost_grid_1_AutoML_3_20241110_193552_model_1	0.939676	0.337021	0.940566	0.129032	0.312821	0.0978573
GBM_2_AutoML_3_20241110_193552	0.939275	0.314694	0.940175	0.131157	0.310749	0.096565
GBM_grid_1_AutoML_3_20241110_193552_model_3	0.938658	0.316917	0.941312	0.136979	0.311918	0.0972926
XGBoost_grid_1_AutoML_3_20241110_193552_model_2	0.936896	0.331909	0.937379	0.133674	0.315171	0.0993328
XGBoost_grid_1_AutoML_3_20241110_193552_model_3	0.935836	0.332681	0.93373	0.134304	0.315109	0.0992934
GBM_1_AutoML_3_20241110_193552	0.935831	0.323219	0.936334	0.13926	0.315675	0.0996505
GBM_5_AutoML_3_20241110_193552	0.934594	0.326321	0.936196	0.138474	0.316985	0.100479
DRF_1_AutoML_3_20241110_193552	0.933046	0.362731	0.934843	0.141699	0.32168	0.103478
XRT_1_AutoML_3_20241110_193552	0.931753	0.380114	0.933302	0.143509	0.339441	0.11522
GBM_grid_1_AutoML_3_20241110_193552_model_1	0.931387	0.33417	0.931668	0.140441	0.321242	0.103197
XGBoost_3_AutoML_3_20241110_193552	0.930251	0.33653	0.931416	0.145633	0.322774	0.104183
XGBoost_2_AutoML_3_20241110_193552	0.930228	0.340598	0.931051	0.144138	0.32315	0.104426
XGBoost_1_AutoML_3_20241110_193552	0.928017	0.344332	0.927765	0.14705	0.325271	0.105802
XGBoost_grid_1_AutoML_3_20241110_193552_model_6	0.925076	0.348769	0.926163	0.152636	0.32969	0.108696
XGBoost_grid_1_AutoML_3_20241110_193552_model_4	0.918814	0.362669	0.920956	0.156098	0.336945	0.113532
GLM_1_AutoML_3_20241110_193552	0.852001	0.480149	0.852118	0.227695	0.395922	0.156755

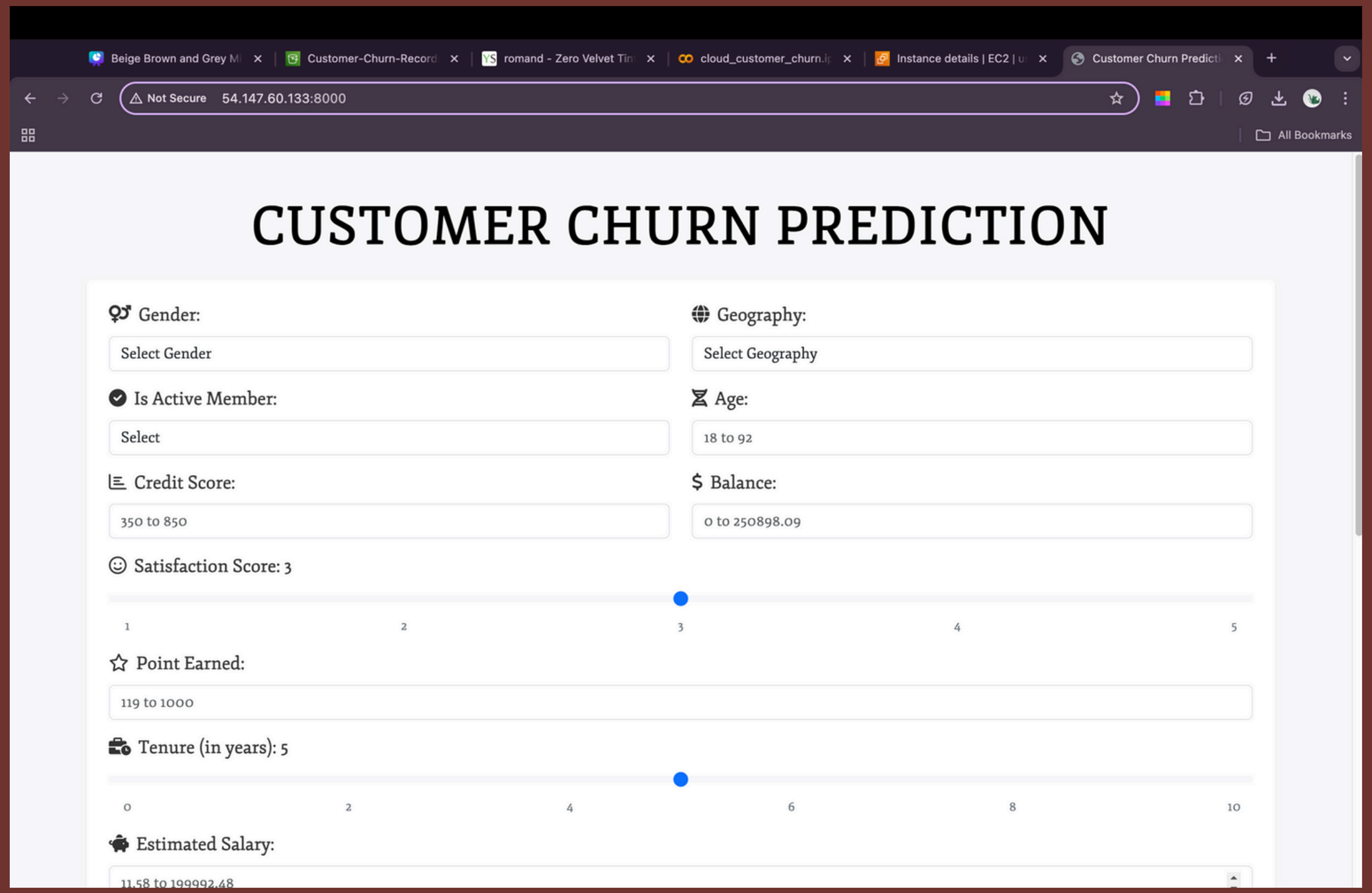
[20 rows x 7 columns]

- **The GBM 4 model (Gradient Boosting Machine) of the AutoML framework performed the best, with an AUC of ~94.5%**
- **The model is stored in S3 (*in mojo format*) for prediction**

Prediction Website

Flask

Created a web application through Flask that fetched the trained model from S3 to make predictions on the input provided



THANK YOU

Questions?