DEPI GRADUATION PROJECT

Customer Churn Prediction

Banks need to understand the factors driving churn

MEET OUR TEAM

Mohamed Seif	Youssef Hatem	Shahd Tarek
Aya Hamdy	Yasmin Kamal	Mohamed Abdel mohsen

AGENDA

1. Business problem
2. EDA
3. Data Visualization
4. Machine Learning
5. Key insights & Deployment

BUSINESS PROBLEM

Predict customer churn in a bank to identify at-risk customers and implement targeted retention strategies. Managing customer churn is crucial for maintaining financial health, profitability, and competitive advantage. Banks need to understand the factors driving churn and implement effective strategies to enhance customer retention.

EDA

01

Data Cleaning

02

Data Transformation

03

Data Exploration

DATA CLEANING

We had many Outliers in three columns when we made a Boxplot
They were dealt with by removing
We also removed the customer_id column because we don't need it for analytics.

```
Data Shape: (10000, 11)
Total Outliers in credit_score: 15 -- 0.15%
Total Outliers in age: 359 -- 3.59%
Total Outliers in tenure: 0 -- 0.0%
Total Outliers in balance: 0 -- 0.0%
Total Outliers in products_number: 60 -- 0.6%
Total Outliers in credit_card: 0 -- 0.0%
Total Outliers in active_member: 0 -- 0.0%
Total Outliers in estimated_salary: 0 -- 0.0%
```

DATA TRANSFORMATION

We have created a lot of features (age group, age category, credit score category, age category A, age category B, age category C, credit to salary ratio, credit to salary ratio, credit to credit score ratio, credit score category, credit score category, tenure category, gender category, gender number, gender number, country Spain, country Germany, gender country, gender country, male gender, credit category)

```
12  gender_num
13  country_Germany
14  country_Spain
15  gender_Male
16  age_seg_B
17  age_seg_C
18  credit_score_seg
19  balance_seg
20  tenure_seg
21  age_group
22  gender_country
23  balance_salary_ratio
24  balance_credit_ratio
25  age_seg_A
```

DATA TRANSFORMATION

With some features broken down into a set of categories

```
data['credit_score_seg'] = pd.cut(data['credit_score'], bins=[349, 500, 590, 620, 660, 690, 720, np.inf],
                                  labels=['A', 'B', 'C', 'D','E','F','G'])

data['balance_seg'] = pd.cut(data['balance'], bins=[-1, 50000, 90000, 127000, np.inf],
                             labels=['A','B','C','D'])

data['age_seg'] = pd.cut(data['age'], bins=[17, 36, 55, np.inf],
                         labels=['A','B','C'])

data['tenure_seg'] = pd.cut(data['tenure'], bins=[-1, 3, 5, 7, np.inf],
                            labels=['A','B','C','D'])
```

DATA TRANSFORMATION

We used Label Encoder to convert categorical features to numeric values to make machine learning easier Here are the features we converted

1- Credit score seg

2- Balance seg

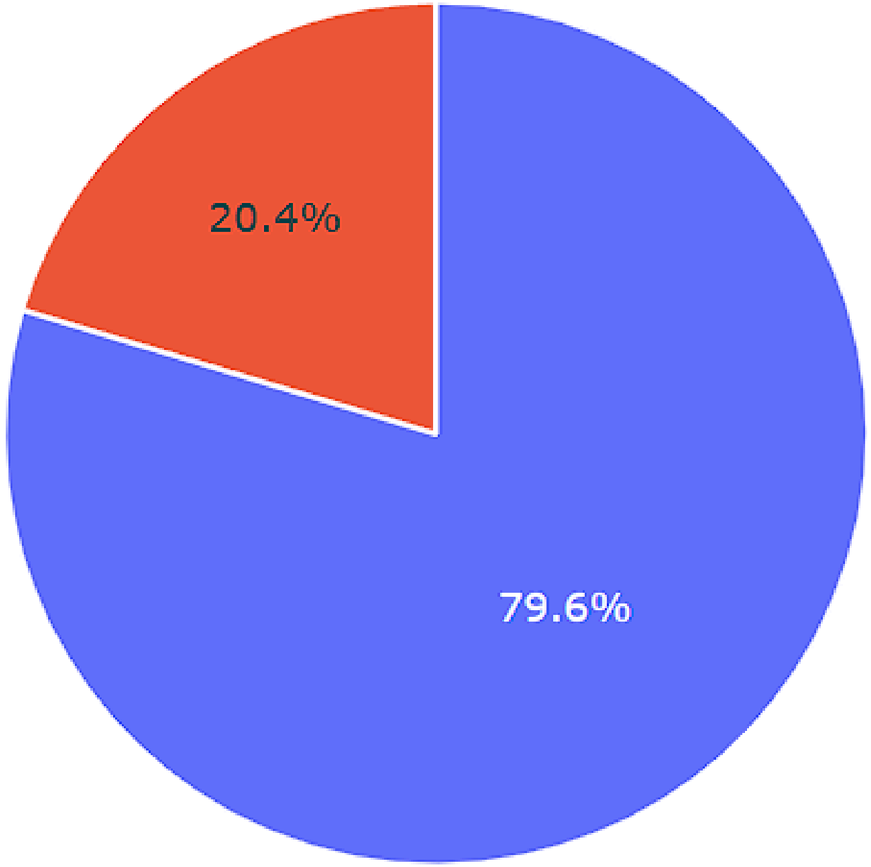
3- Tenure seg

This transformation allows machine learning models to effectively interpret categorical data

credit_score_seg	balance_seg	tenure_seg
2	0	0
2	1	0
1	3	3
5	0	0
6	2	0
3	2	3
6	0	2
0	2	1
1	3	1
4	3	0
1	2	2

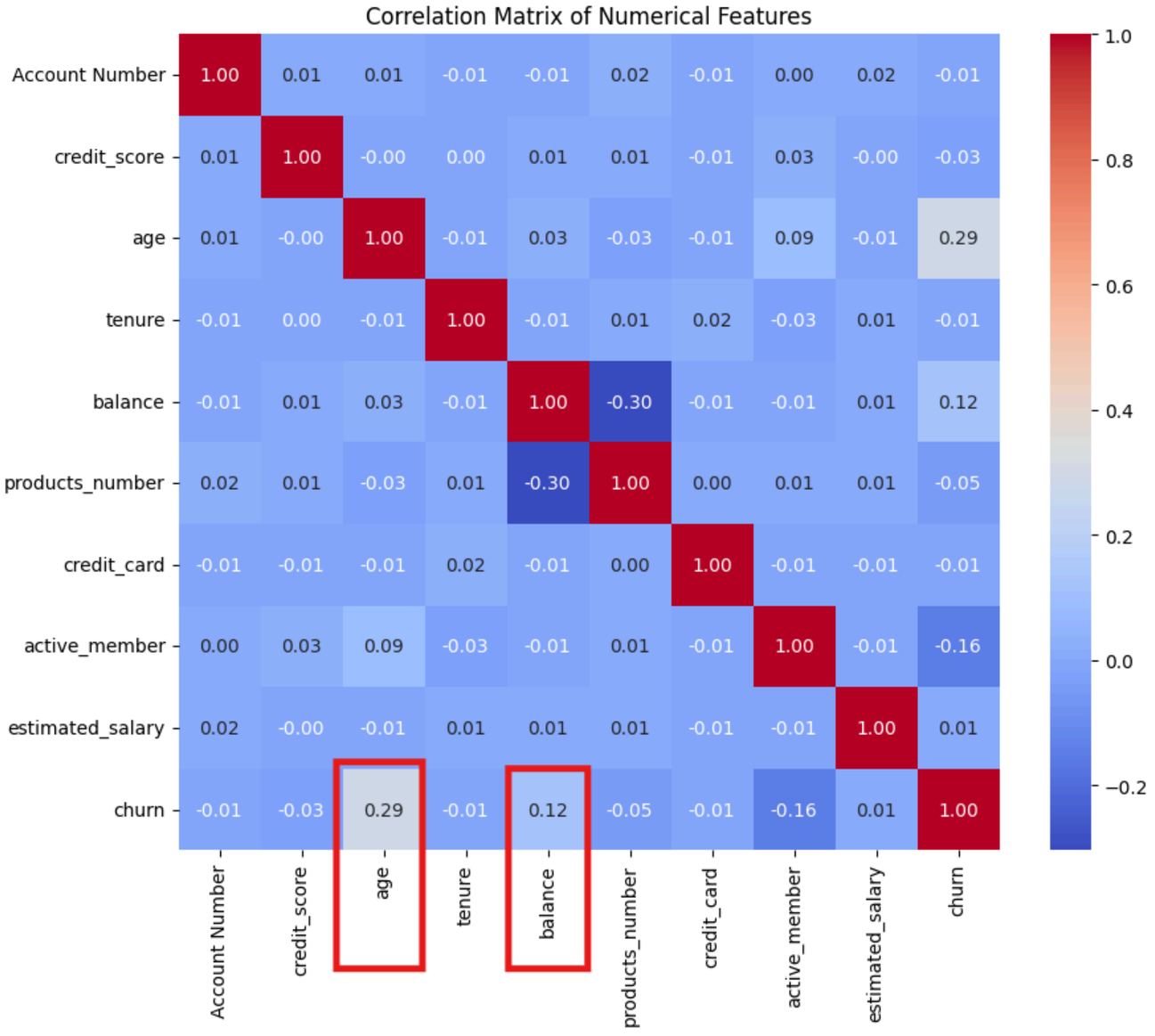
DATA EXPLORATION

Segmentation of customer churned and retained

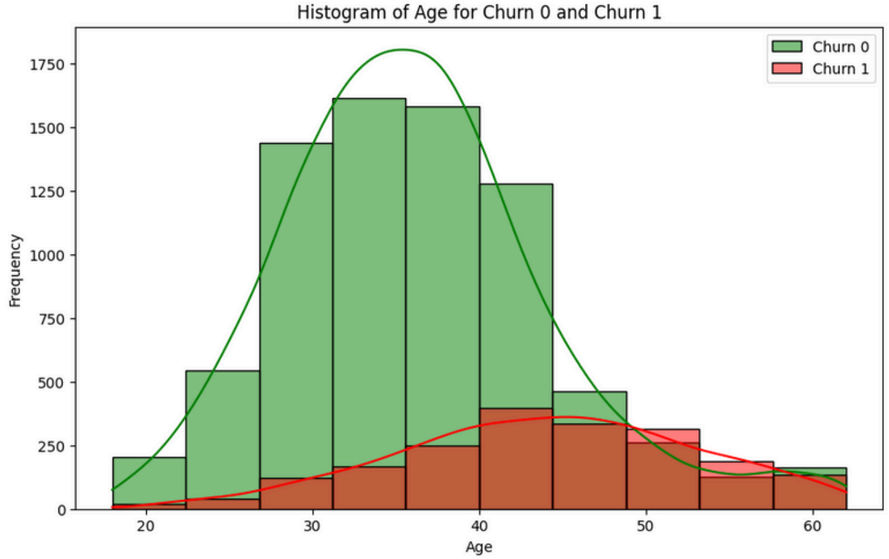


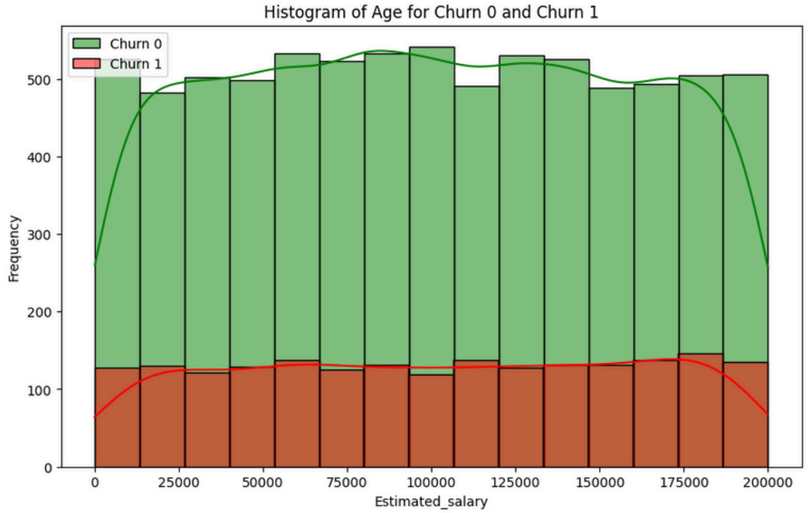
20% of customers leaving the bank

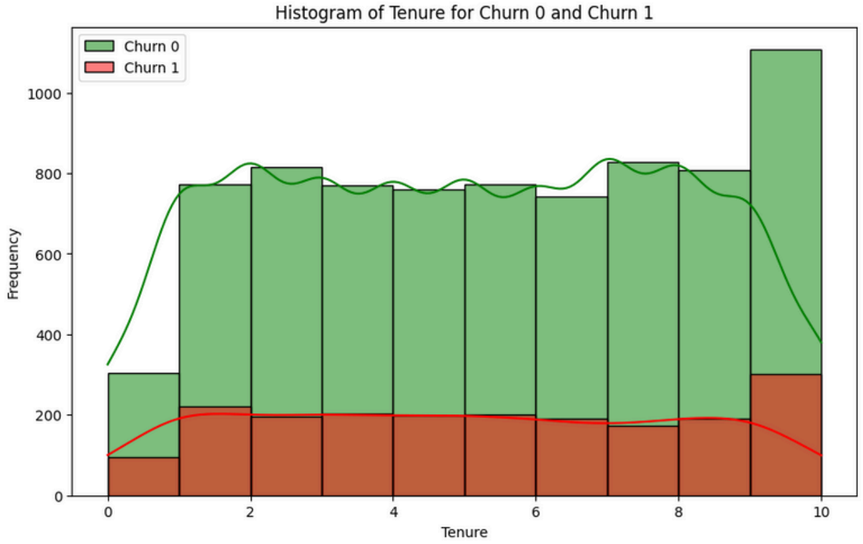
DATA EXPLORATION



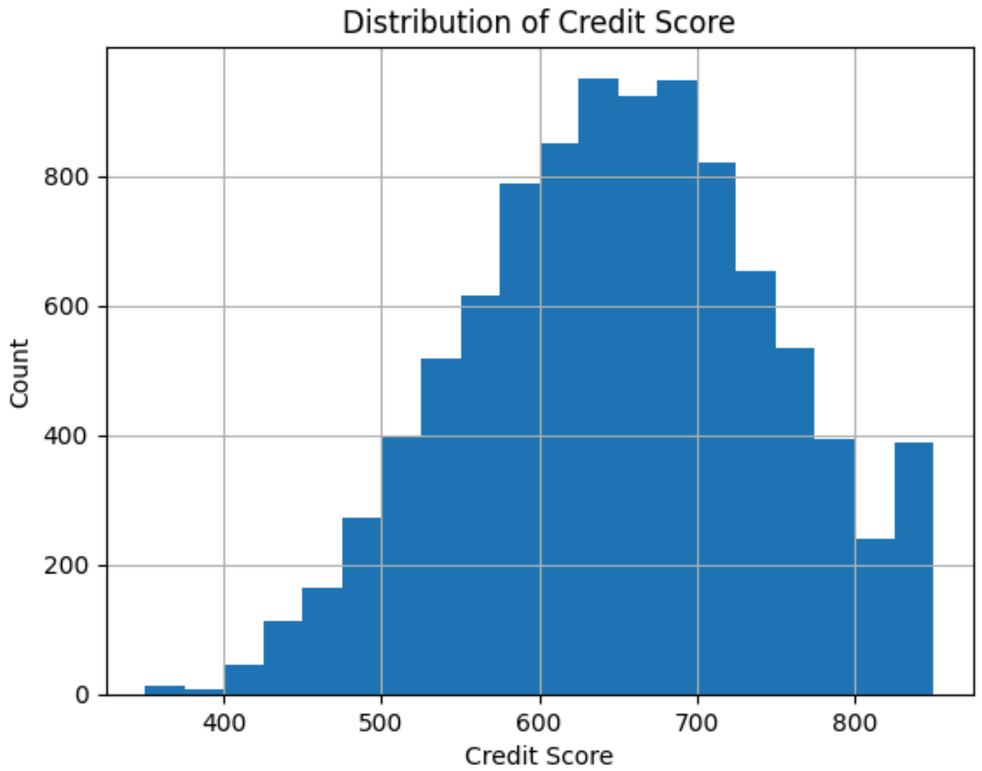
DATA EXPLORATION

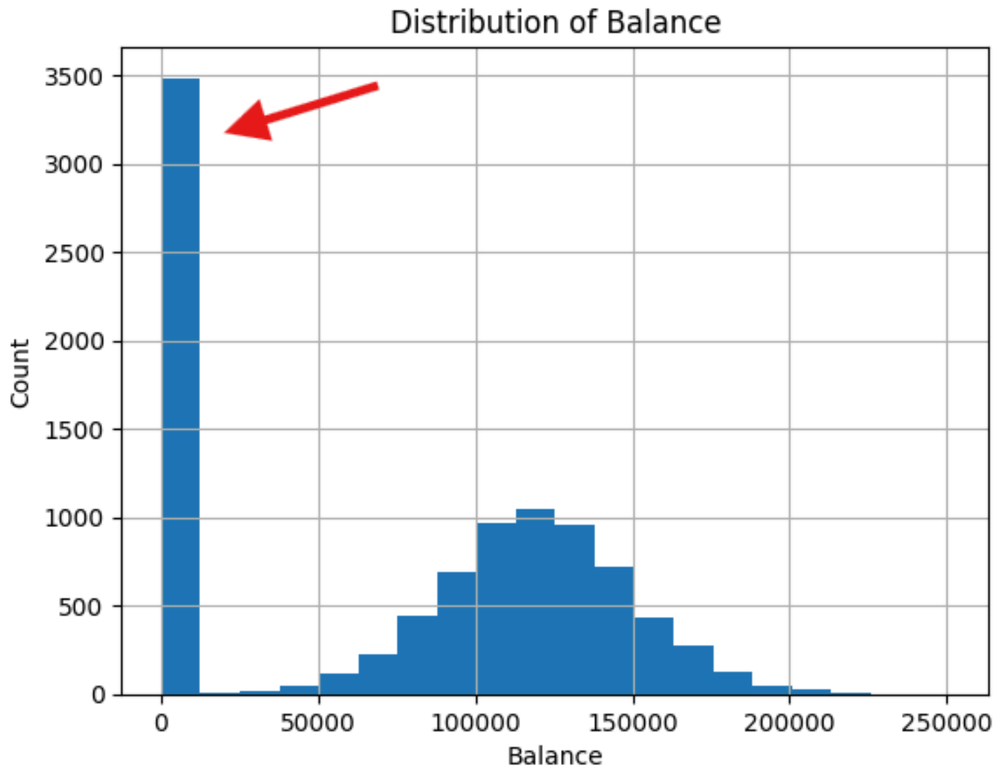




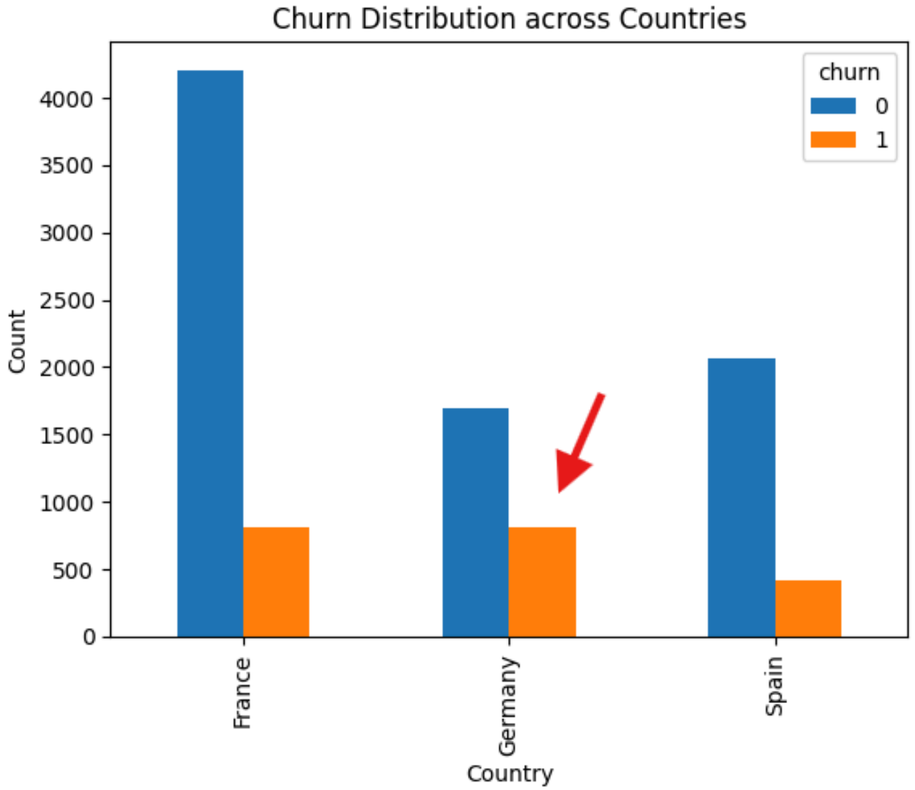


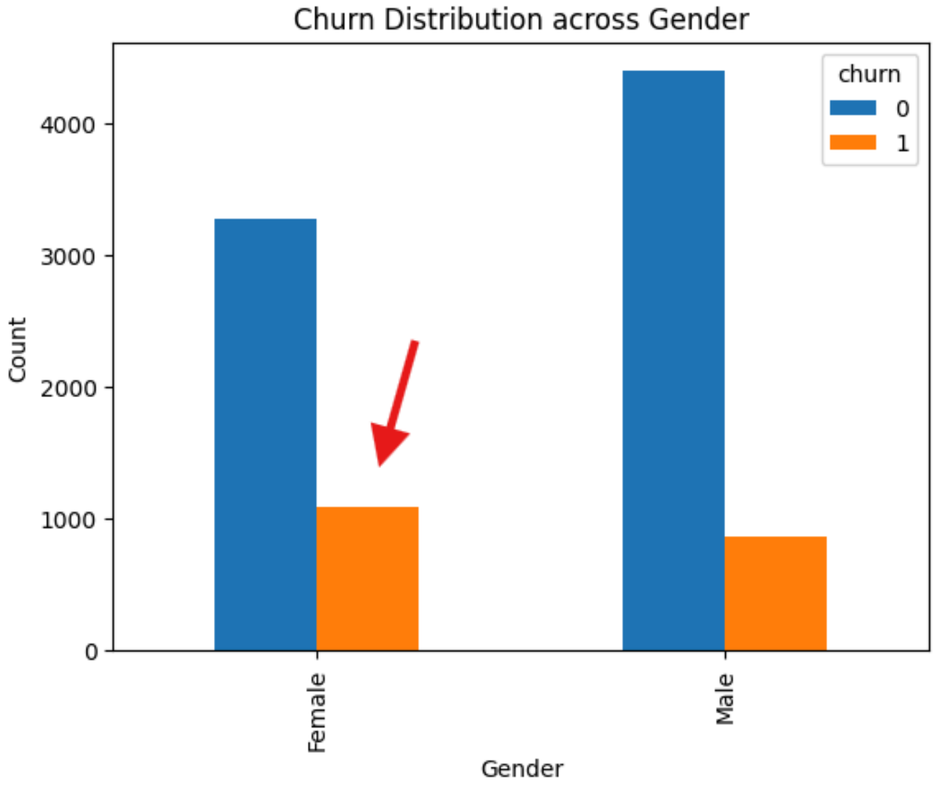
DATA EXPLORATION



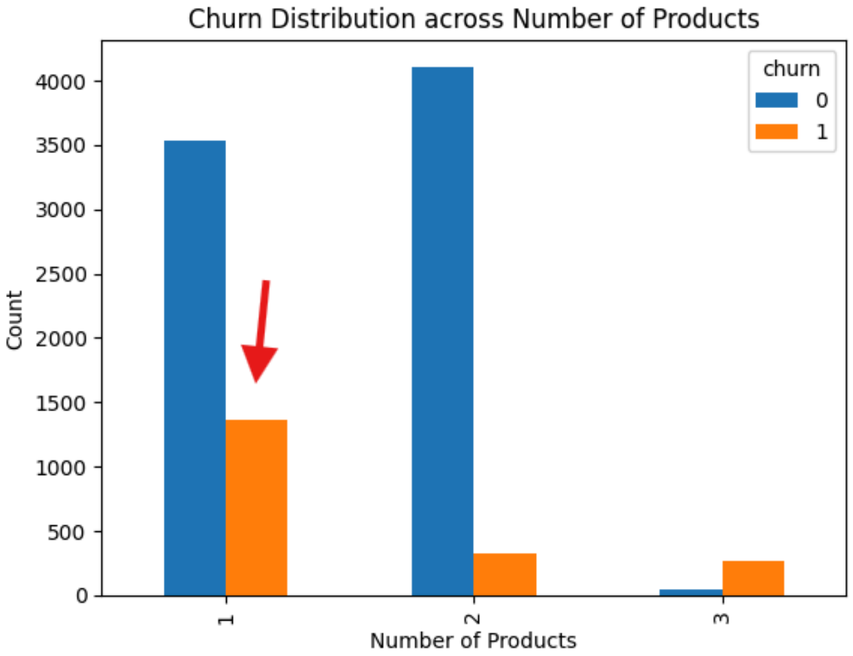


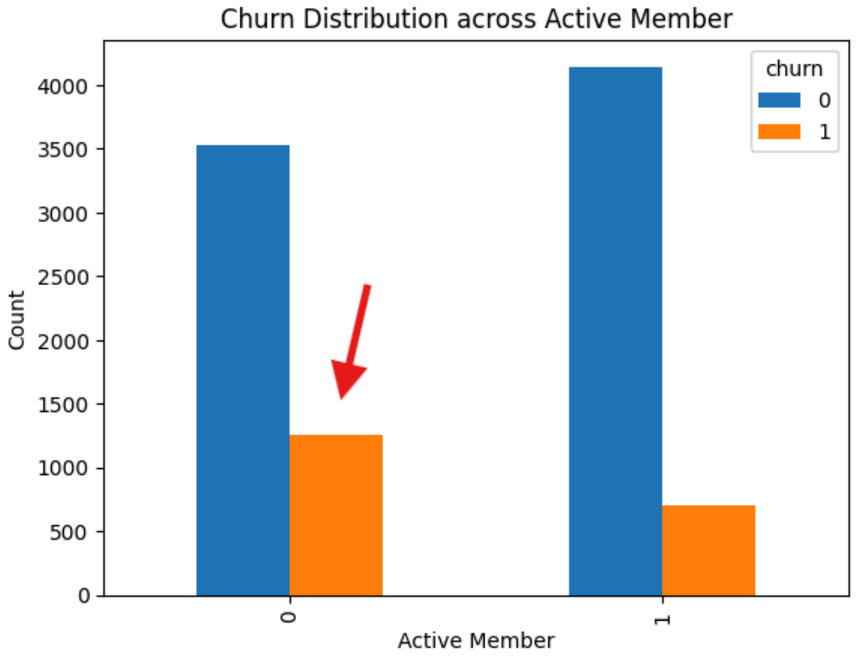
DATA EXPLORATION

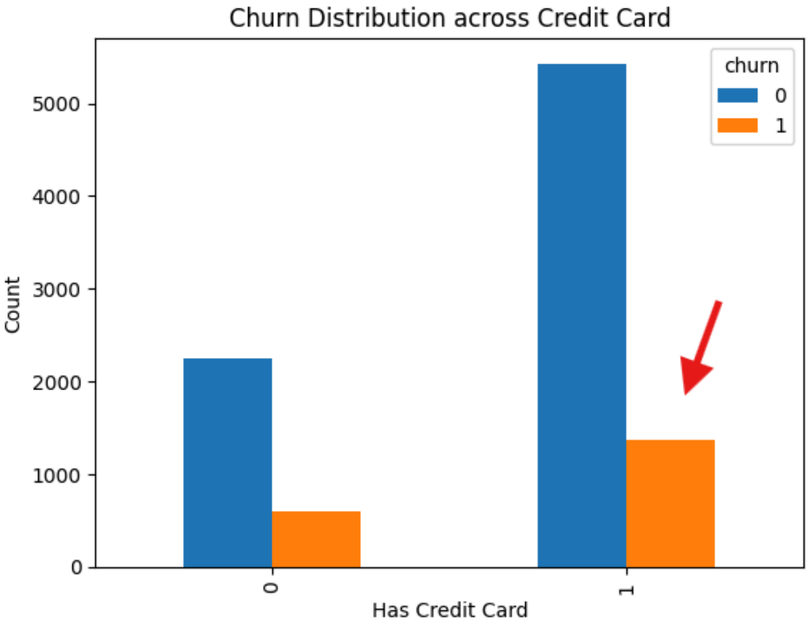




DATA EXPLORATION

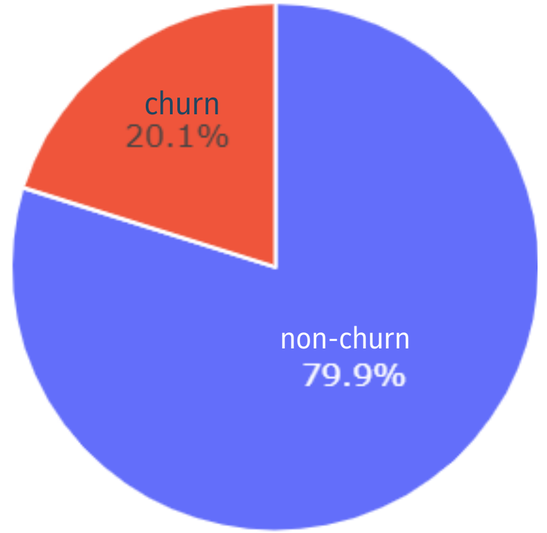


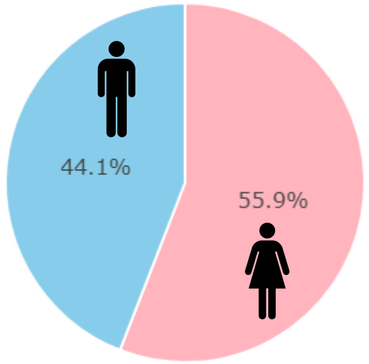




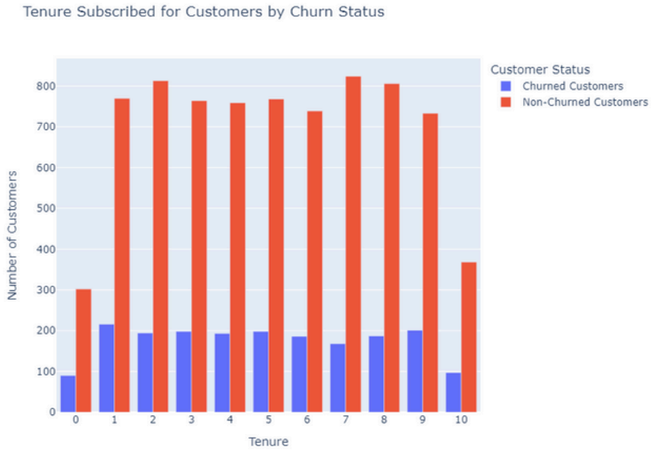
VISUALIZATION

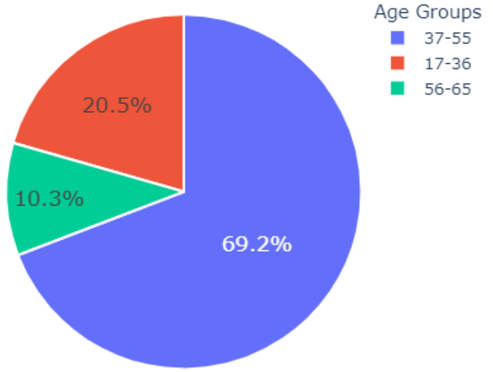
THE MAIN FEATURES FOR CHURNED CUSTOMERS :

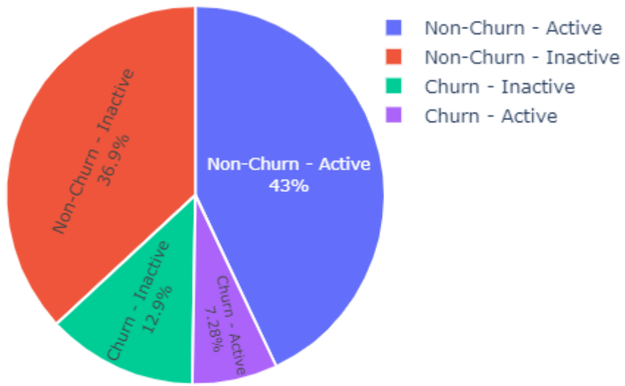


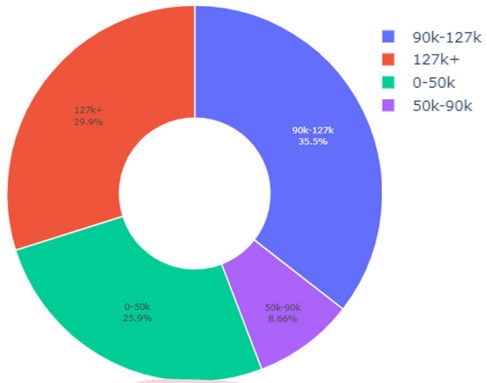


Gender

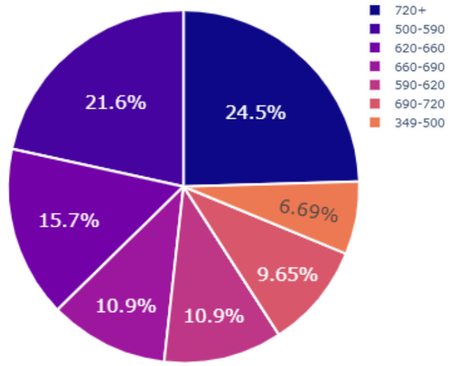




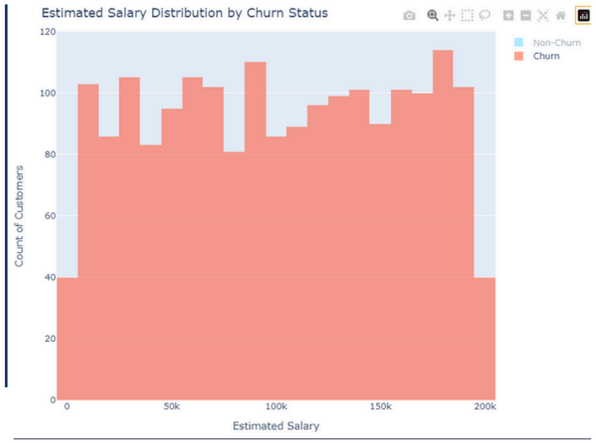




Balance



credit score



GENDER

SOME ANALYSES FOR IMPORTANT FUEATUERS:

FIGURE (1 ,2):

NON-CHURN: THERE IS A SIGNIFICANT GAP BETWEEN MEN AND WOMEN IN ALL POINTS.

CHURN: MEN WITH A LOW BALANCE ARE MORE LIKELY TO CHURN.

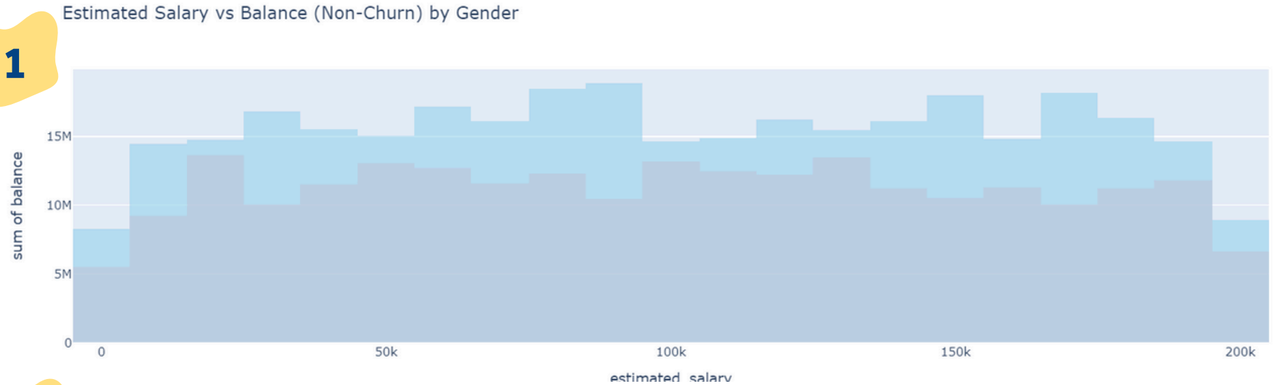
FIGURE (3):

FRANCA HAS THE MOST COUSTOMERS PARTICIPATING IN THE BANK , BUT THE NUMBER OF MEN IS GREATER THAN WOMEN.

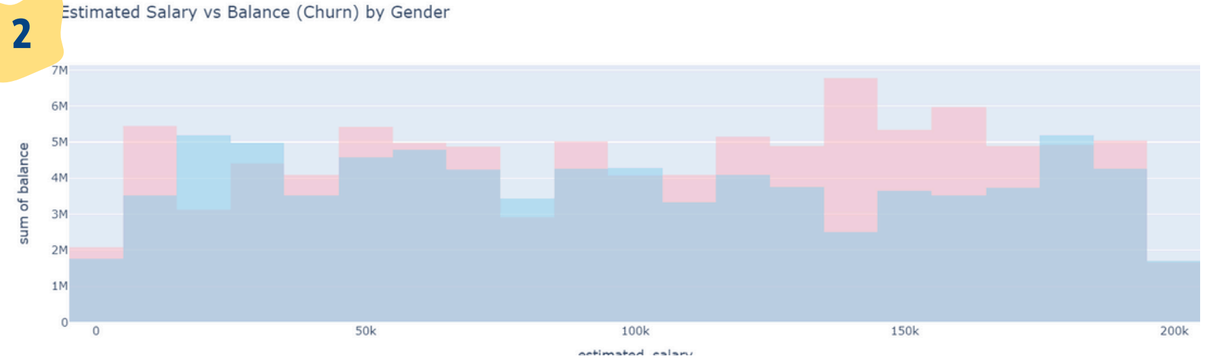
FIGURE (4):

WOMEN WITH A CREDIT_SCORE > 720 HAVE A HIGHST NUMBER OF CHURN AND NON-CHURN COMPARED TO OTHER CREDIT_SCORES

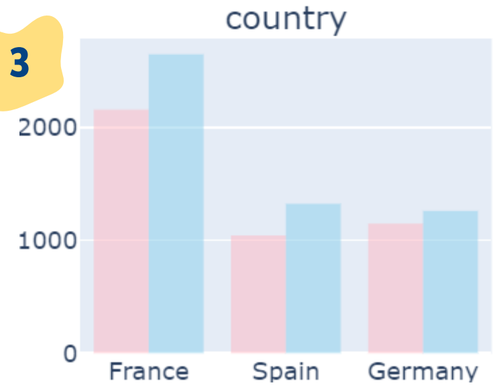
1



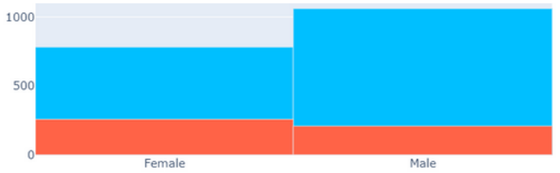
2



3



4



CREDIT SCORE>270

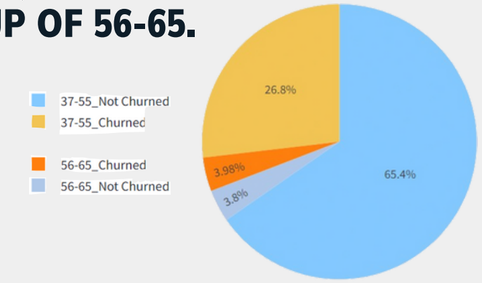
AGE

SOME ANALYSES FOR IMPORTANT FUEATUERS:

1

AS WE SAW BEFORE IN THE PIE CHART :

NO MATTER IF IT'S CHURN OR NOT, THE AGE GROUP OF 37-55 IS STILL LARGER THAN THE GROUP OF 56-65.



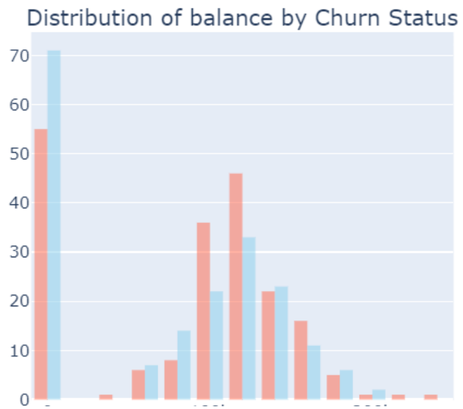
2

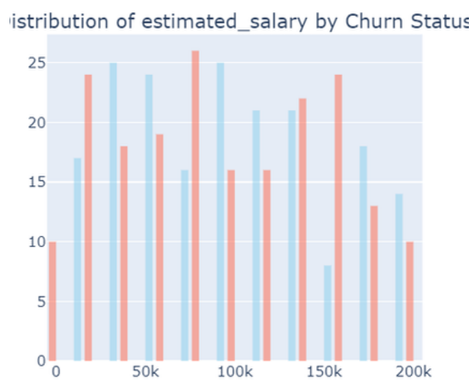
FIGURE (1 ,2 ,3):

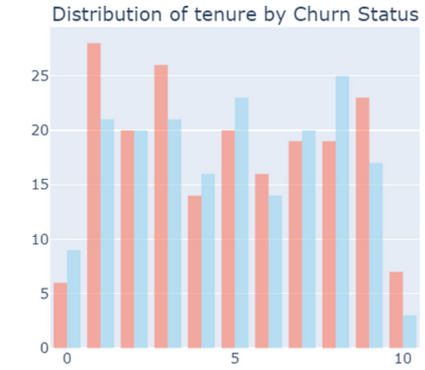
FOR CHURN ,AND NON-CHURN : FROM SOME FEATURES EXAPLES (BALANCE, ESTIMATED_SALARY, TENURE), THERE ARE A BIG GAP OF NUMBERS BETWEEN AGE_GROUP 37-55 AND 56-65 AS WE CAN SEE IN THE Y-AXIS, SO IT MEANS THAT THE BANK DOES NOT HAVE BIG NUMBERS OF CUSTOMERS FROM 55-65 GROUP

3

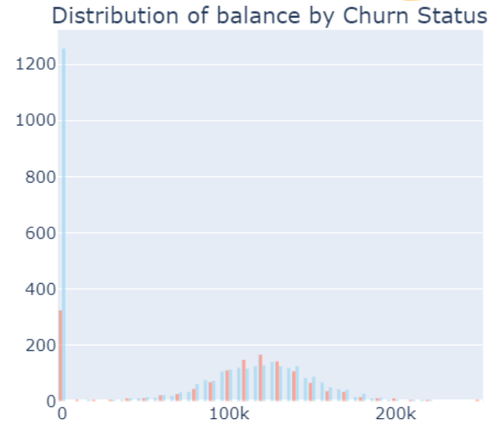
56-65

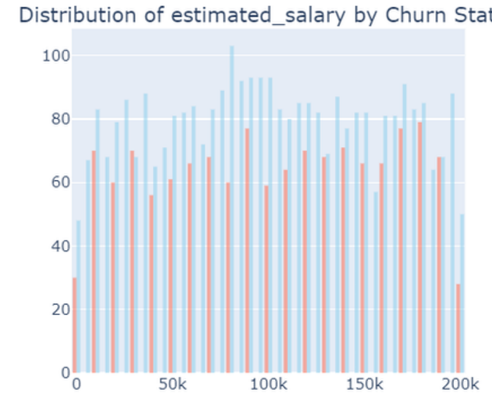


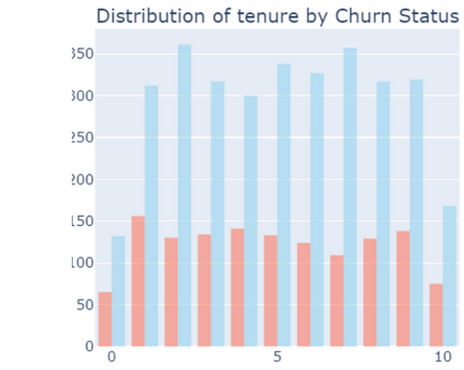




37-55







PRODUCTS NUMBER

SOME ANALYSES FOR IMPORTANT FUEATUERS:

FIGURE (1 ,2):

A FEW NUMBERS OF PEOPLE HAVE 3 PRODUCTS NUMBER.

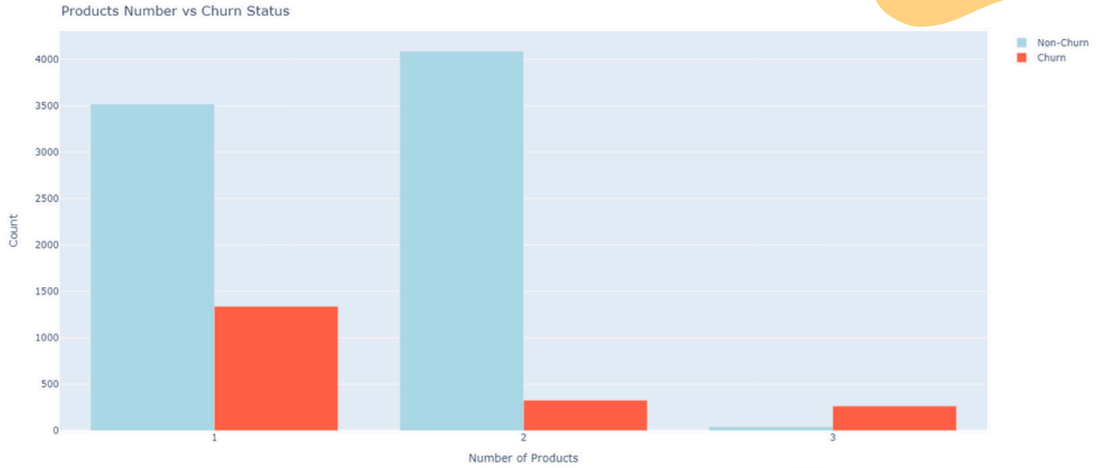
FIGURE (3 ,4):

CHURN ACTIVE: THE NUMBERS OF CUSTOMERS ARE INCREASED IN 1 PRODUCTS_NUMBER EITHER THAN THERE ACTIVE OR NON-ACTIVE COMPARED TO THE 2 PRODUCTS_NUMBER.

NON-CHURN ACTIVE: THE RATIO IS ALMOST IDENTICAL BETWEEN 1 PRODUCTS_NUMBER ,AND 2 PRODUCTS_NUMBER.

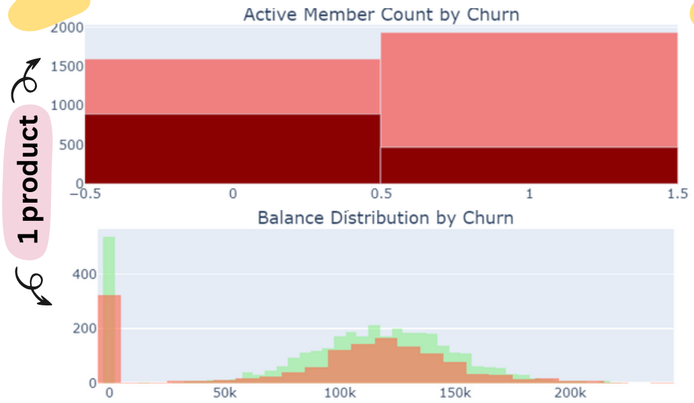
BALANCE: THE BALANCE FOR 1 PRODUCTS_NUMBER EITHER THAN FOR CHURN OR NON-CHURN COUSTOMERS IS THE BIGIST , SO IT MEANS THAT THE COUSTOMERS PERFER 1 PRODUCTS_NUMBER.

1

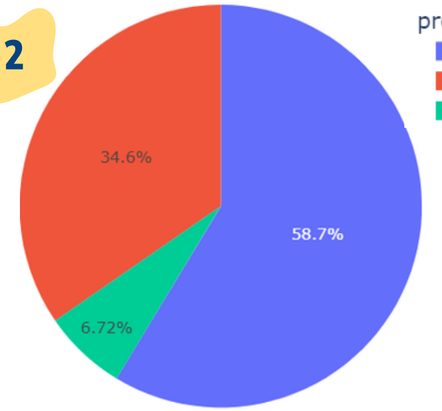


3

1 product

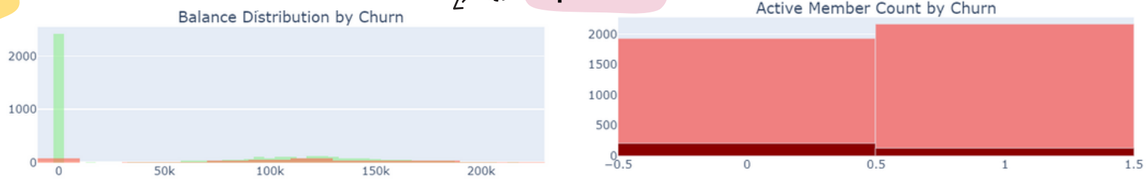


2



4

2 product

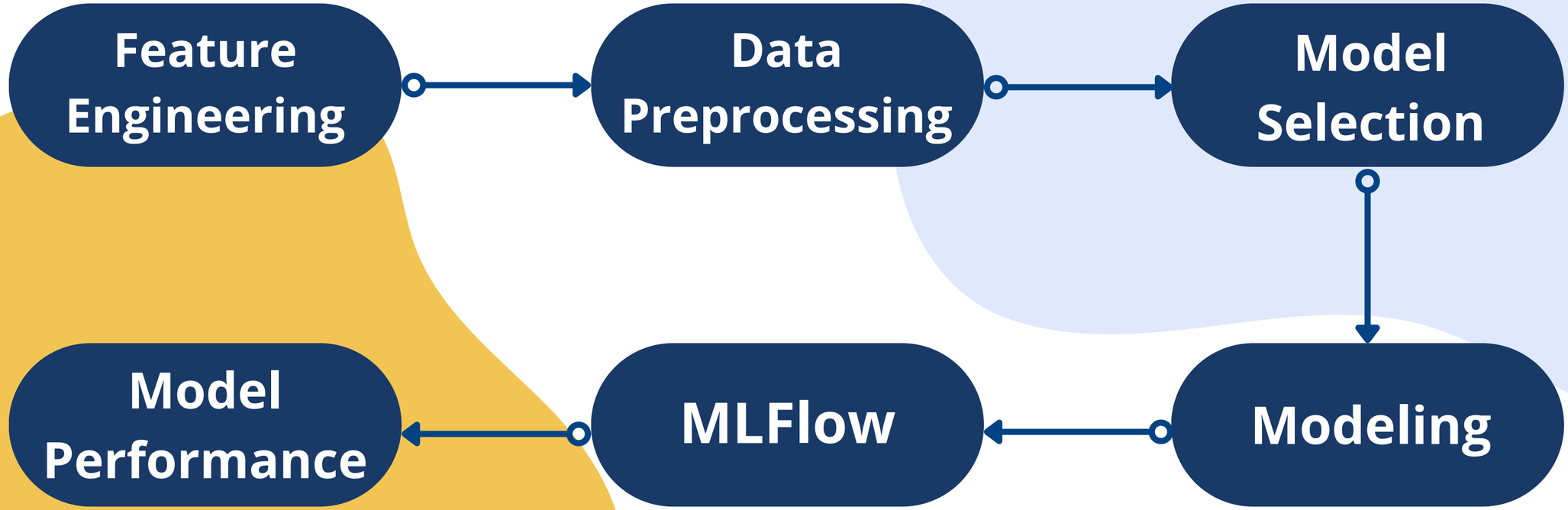


MACHINE LEARNING MODELS

Churn Prediction

Predicting whether an employee will leave the bank using machine learning models.

Process to Build the Model



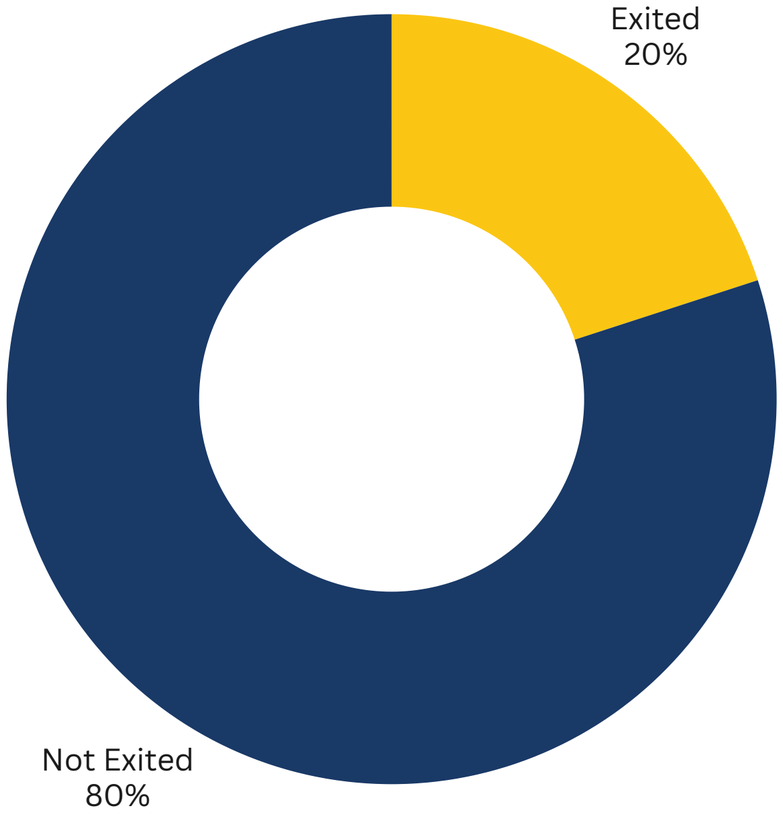
Feature Selection

```python
x = data.drop(columns=['churn'], axis=1) # Features

X = pd.get_dummies(x, columns=['country','gender']) # One-Hot Encoding

y = data['churn'] # Target
x.head()
```

In this step, I performed encoding on categorical variables, specifically the country and gender columns, using One-Hot Encoding.

Age	credit_score	tenure
gender_Male	gender_Female	balance
country_France	country_Spain	country_Germany
credit_card	active_member	estimated_salary

PREPROCESSING PIPELINE



We can see that data is highly imbalanced. Almost 80% of our data is from class 0 (not exited) and 20% data is from class 1 (exited).

In a real life we only care about the people who are leaving (Exited) the bank, and we only want to analyze the patterns of those people.

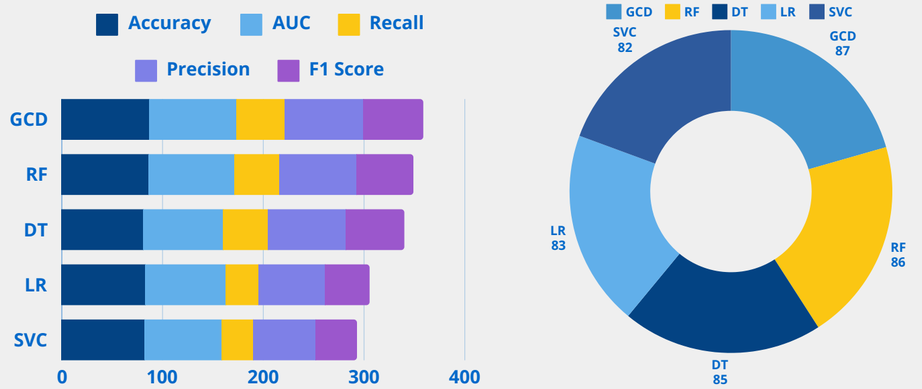
We applied SMOTE to balance the dataset, ensuring the model can predict customer churn without bias toward the majority class.

MODEL SELECTION

BEST CHURN MODEL

	Model	Accuracy	AUC	Recall	Prec.	F1
gbc	Gradient Boosting Classifier	0.8678	0.8684	0.4789	0.7797	0.5928
rf	Random Forest Classifier	0.8609	0.8550	0.4455	0.7663	0.5629
lightgbm	Light Gradient Boosting Machine	0.8608	0.8586	0.4878	0.7307	0.5842
et	Extra Trees Classifier	0.8578	0.8481	0.4492	0.7428	0.5592
ada	Ada Boost Classifier	0.8575	0.8468	0.4818	0.7177	0.5761
xgboost	Extreme Gradient Boosting	0.8543	0.8509	0.4877	0.6972	0.5735
lr	Logistic Regression	0.8314	0.7965	0.3284	0.6636	0.4388
ridge	Ridge Classifier	0.8287	0.7954	0.2276	0.7399	0.3470
dt	Decision Tree Classifier	0.7903	0.7903	0.8102	0.7794	0.7944
svm	SVM - Linear Kernel	0.8205	0.7699	0.3143	0.6247	0.4056
dt	Decision Tree Classifier	0.7903	0.7903	0.8102	0.7794	0.7944

ALGORITHM COMPARISON



These results indicate that **Gradient Boosting Classifier (GBC)** was the best choice for this churn prediction task. The decision to proceed with GBC was driven by its **higher accuracy and AUC scores**, which are critical for predicting customer churn.

MODELING

Decision Tree

timized Decision Tree Accuracy: 0.8605744125326371

	precision	recall	f1-score	support
0	0.88	0.96	0.92	1544
1	0.71	0.47	0.56	371
accuracy			0.86	1915
macro avg	0.80	0.71	0.74	1915
ighted avg	0.85	0.86	0.85	1915

86%

Logistic Regression

Classification Report:

	precision	recall	f1-score	support
0	0.82	0.82	0.82	1532
1	0.82	0.82	0.82	1583
accuracy			0.82	3115
macro avg	0.82	0.82	0.82	3115
weighted avg	0.82	0.82	0.82	3115

AUC-ROC: 0.9026483244789201

82%

Support Vector Classifier

Classification Report:

	precision	recall	f1-score	support
0	0.82	0.82	0.82	1532
1	0.82	0.82	0.82	1583
accuracy			0.82	3115
macro avg	0.82	0.82	0.82	3115
weighted avg	0.82	0.82	0.82	3115

AUC-ROC: 0.9027015169333437

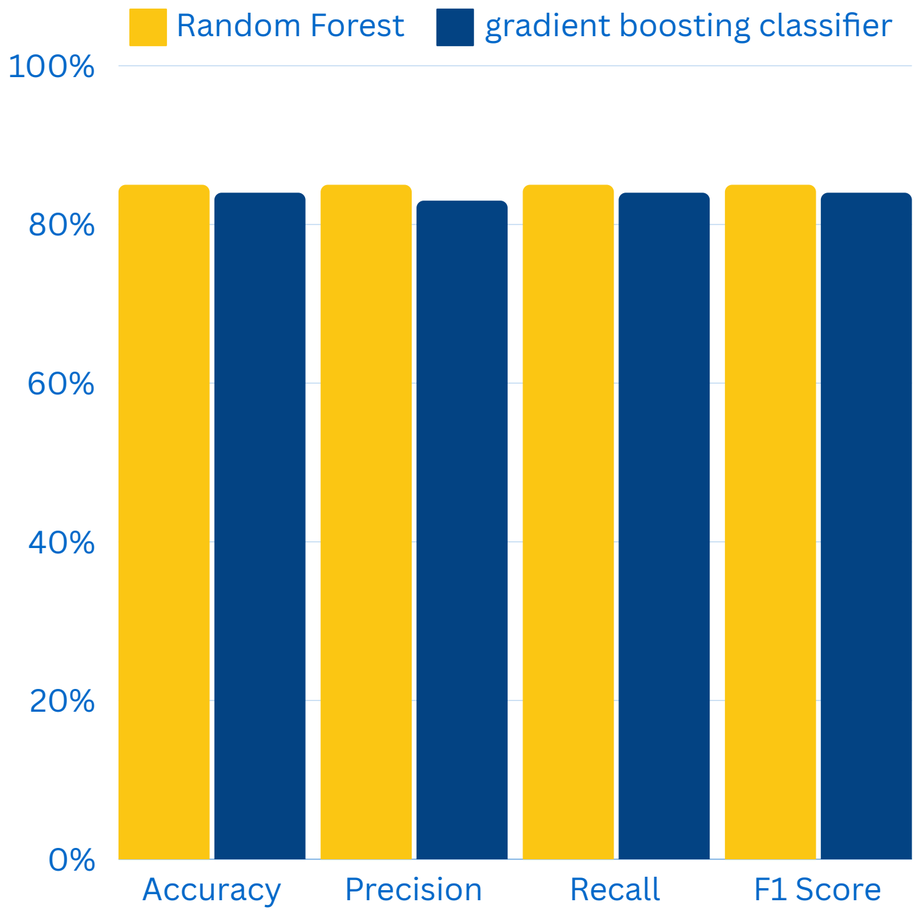
82%

GRADIENT BOOSTING CLASSIFIER AND RANDOM FOREST

- **Random Forest excels in Accuracy, Precision, and F1 Score, making it a more balanced and reliable model overall.**

Best Model

Metric	Gradient Boosting Classifier	Random Forest
Accuracy	~84%	~85%
Precision	~83%	~85%
Recall	~84%	~85%
F1 Score	~84%	~85%



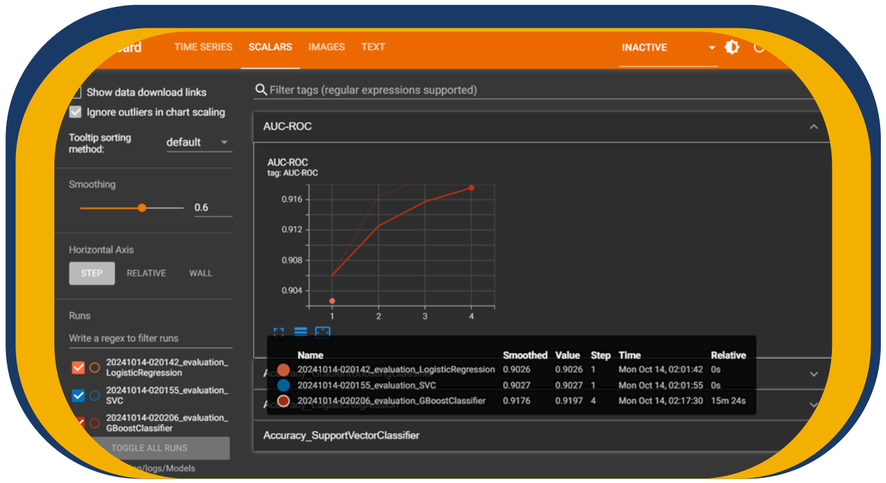
MLFLOW

WHY TENSORBOARD?

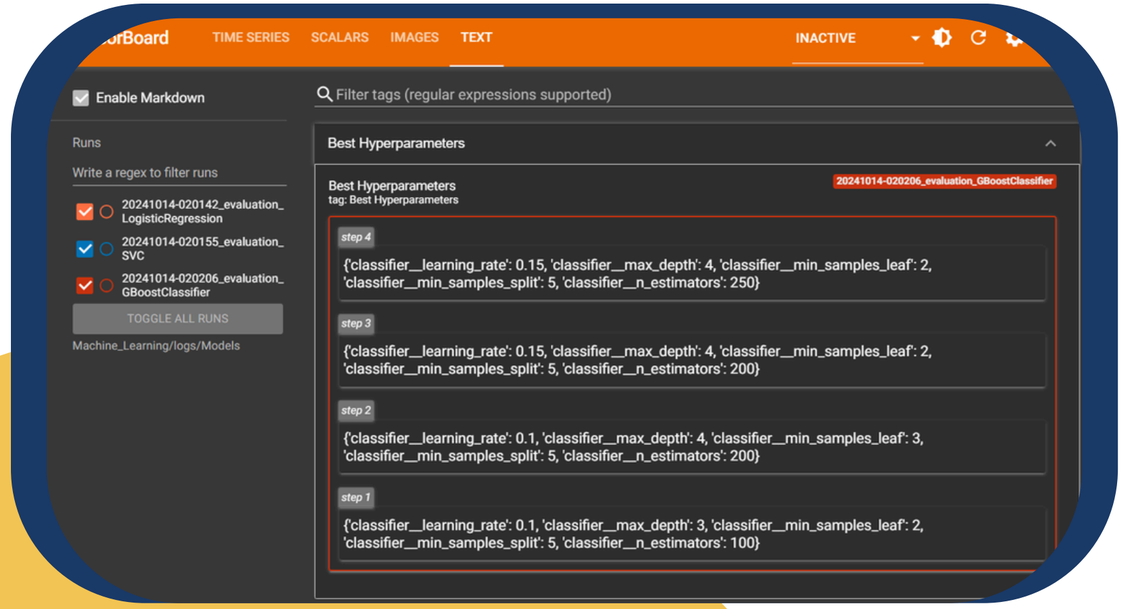
We used TensorBoard to monitor the performance of machine learning models during training, tracking key metrics like accuracy, AUC-ROC, and confusion matrices for models such as Logistic Regression, SVM, and Gradient Boosting. This helped me visualize improvements and optimize hyperparameters effectively.

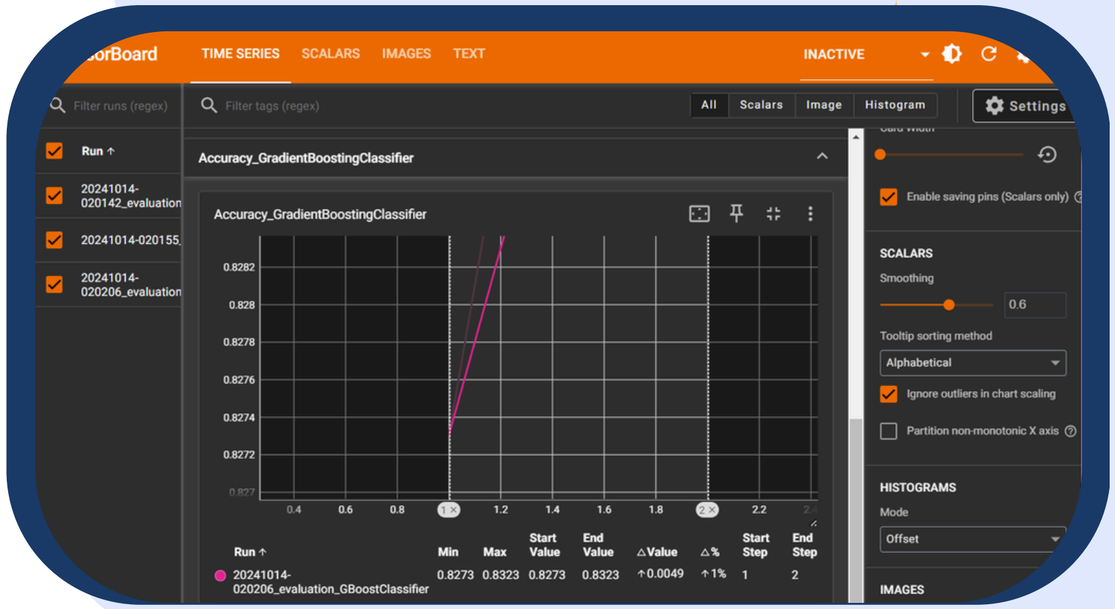
HOW WE USED IT:

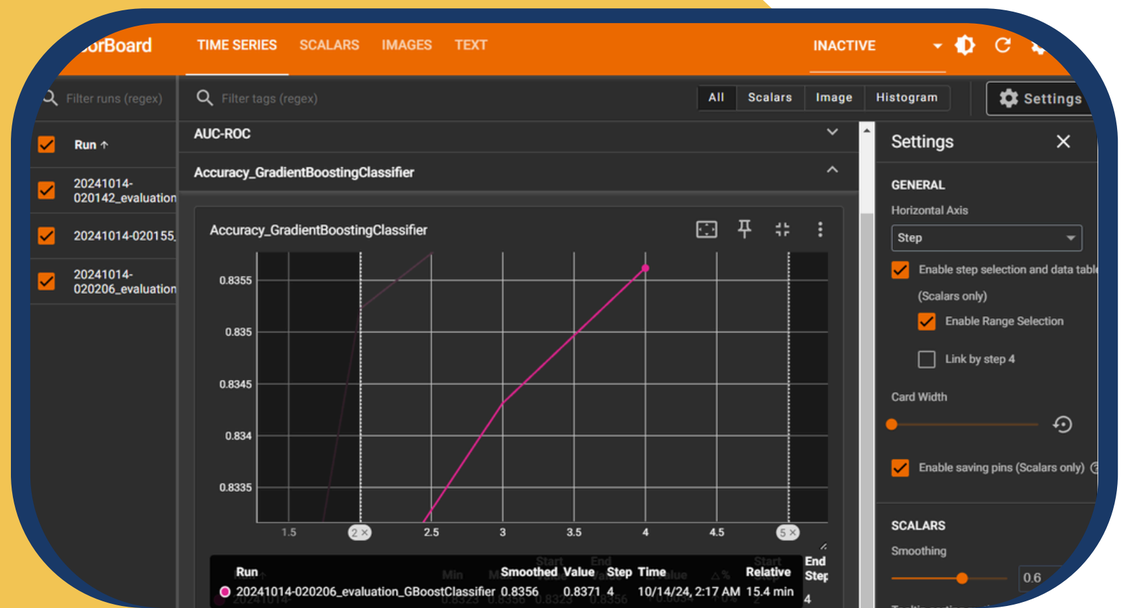
- Logged accuracy, AUC-ROC, and confusion matrices for models
- Tracked and visualized the best hyperparameters during model tuning.

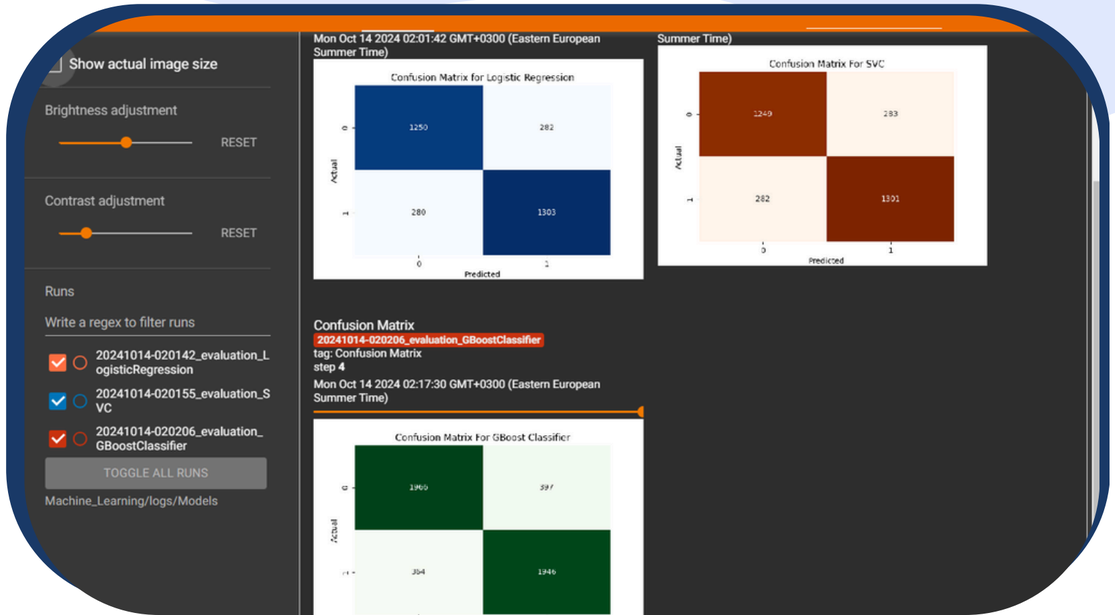


Efficient Hyperparameter Tuning







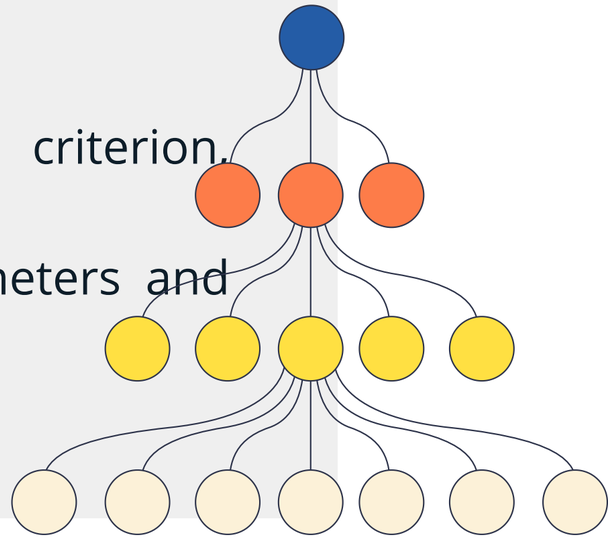


MODEL IMPROVEMENTS

Is a measurement of how accurate predictions or classifications a model makes on new, unseen data. You typically measure model performance using a test set, where you compare the predictions on the test set to the actual outcomes

DECISION TREE (INITIAL MODEL)

- Description: A simple model that splits data based on feature values to predict outcomes.
- Initial Performance:
- Accuracy: ~86%
- Strength: Easy to interpret and fast.
- Limitation: Tended to overfit and was sensitive to noisy data.
- Improvement: Introduced hyperparameter tuning using GridSearchCV:
- Parameters Tuned: max_depth, min_samples_split, min_samples_leaf, criterion, ccp_alpha (pruning).
- Result: Improved accuracy by reducing overfitting with optimal parameters and pruning, achieving ~86% accuracy, F1 Score ~56%.



RANDOM FOREST (ENSEMBLE LEARNING)

- Description: An ensemble model that combines multiple decision trees to reduce overfitting and improve robustness.
- Initial Performance:
- Accuracy: Higher than Decision Tree due to reduced variance and better generalization.
- Improvement: Applied hyperparameter tuning using GridSearchCV:
- Parameters Tuned: n_estimators, max_depth, min_samples_split, min_samples_leaf.
- Key Improvement: Increased n_estimators to 200, found the optimal max_depth of 20, and adjusted min_samples_split for better data handling.
- Result: Achieved the highest accuracy (~85%), F1 Score ~85%.

Deployment

GitHub

GitHub simplifies teamwork by organizing code in a shared repository. Contributors work on feature branches,
the repository is prepared for deployment, making the web app launch smoother.

Web app

A user-friendly Streamlit app that integrates a deployed machine learning model with interactive dashboards. The app is designed for smooth interaction, making complex data accessible and actionable for users.

Hugging face

A user-friendly app deployed on Hugging Face, utilizing Docker for containerization and scalability. It provides an API for seamless integration, allowing users to interact with the model and make predictions efficiently.

GitHub Repo

Graduation-project-DEPI

Public

Pin | Watch 0 | Fork 0 | Star 0

main | 5 Branches | 0 Tags | Go to file t | Add file | Code

mohamedseif-10 Update README.md — 76bc5f6 · 7 minutes ago — 108 Commits

.devcontainer	Added Dev Container Folder	20 hours ago
.vscode	uisng hugging face API	6 hours ago
Data	Merge pull request #11 from mohamedseif-10/Web_app_bra...	3 days ago
EDA	Add files via upload	3 days ago
Machine_Learning	Streamlit	2 days ago
visualization	Add files via upload	3 days ago
web_app	Update 2_😈_Model prediction.py	11 hours ago
Dockerfile	finally done streamlit	yesterday
Model_API.py	uisng hugging face API	6 hours ago
README.md	Update README.md	7 minutes ago
requirements.txt	finally done streamlit	yesterday

README

Graduation-project-DEPI Bank Churn Prediction

About

Streamlit web app link

graduation-project-depi-2024-data-scine...

Readme
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

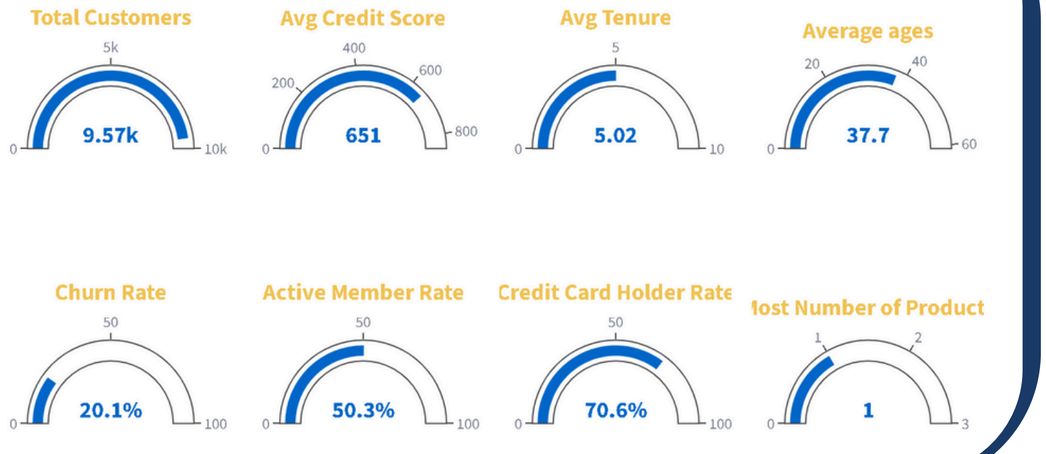
Contributors 6

Languages

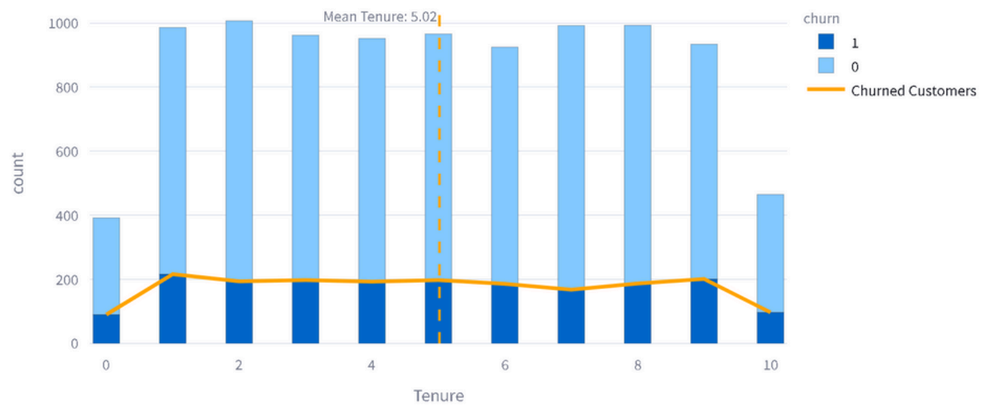
Jupyter Notebook 99.9% Other 0.1%

Key insights

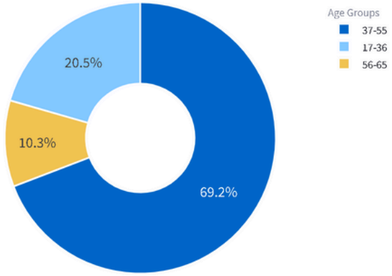
Descriptive Statistics



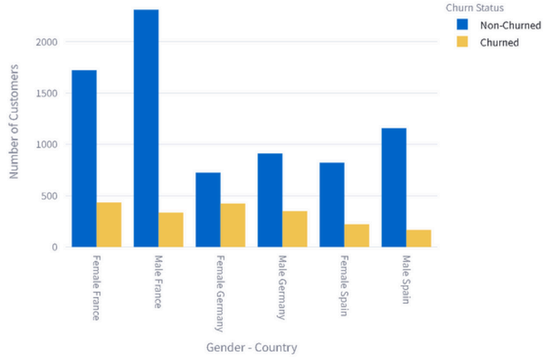
Tenure Distribution with Churn Trend



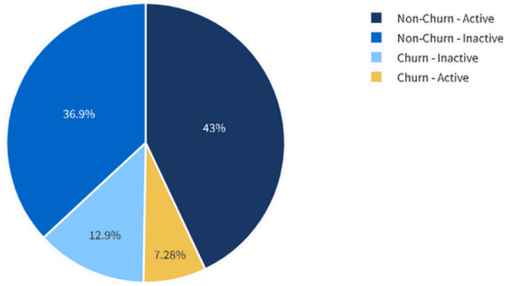
Percentage of Churned Customers by Age Group



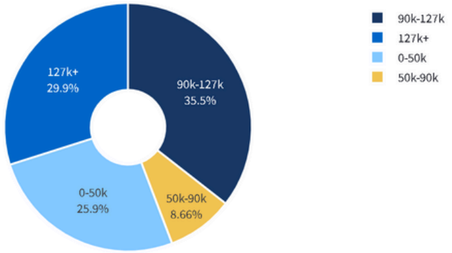
Customer Count by Gender and Country for Churn Status



Churn vs Non-Churn by Active Membership

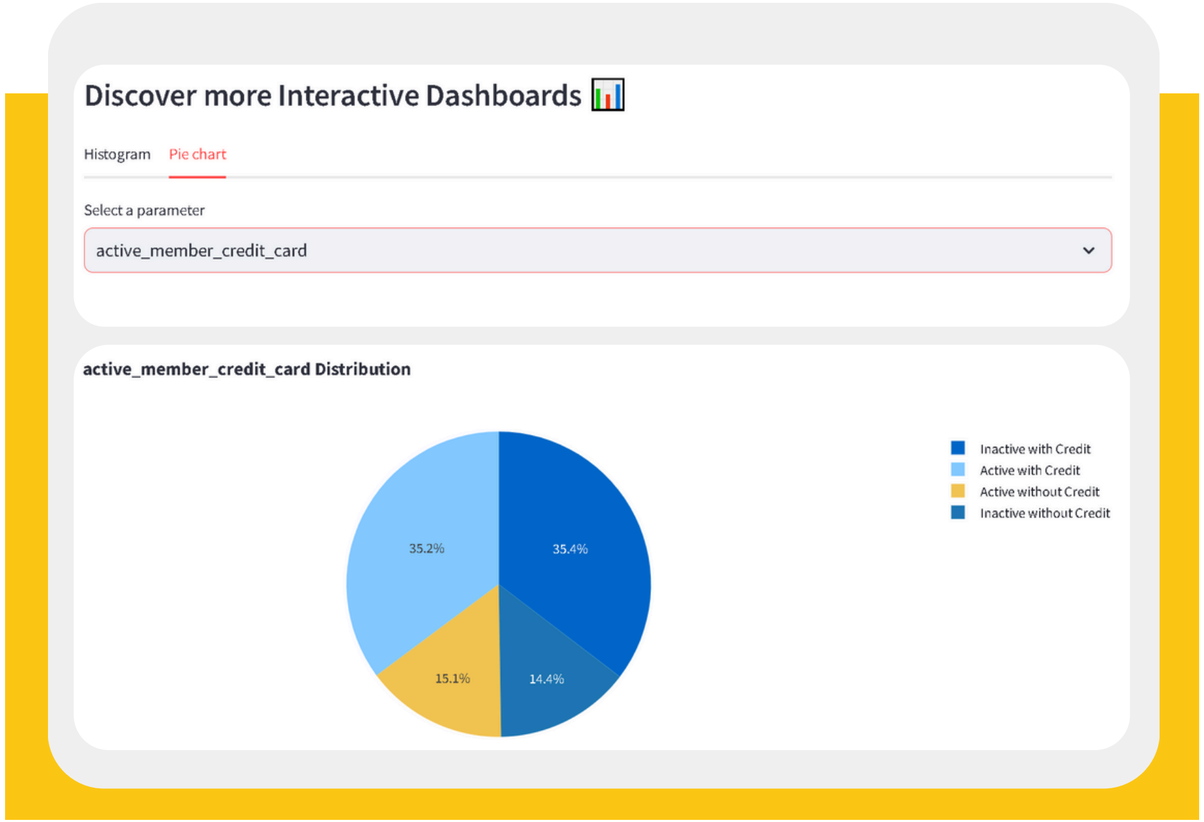


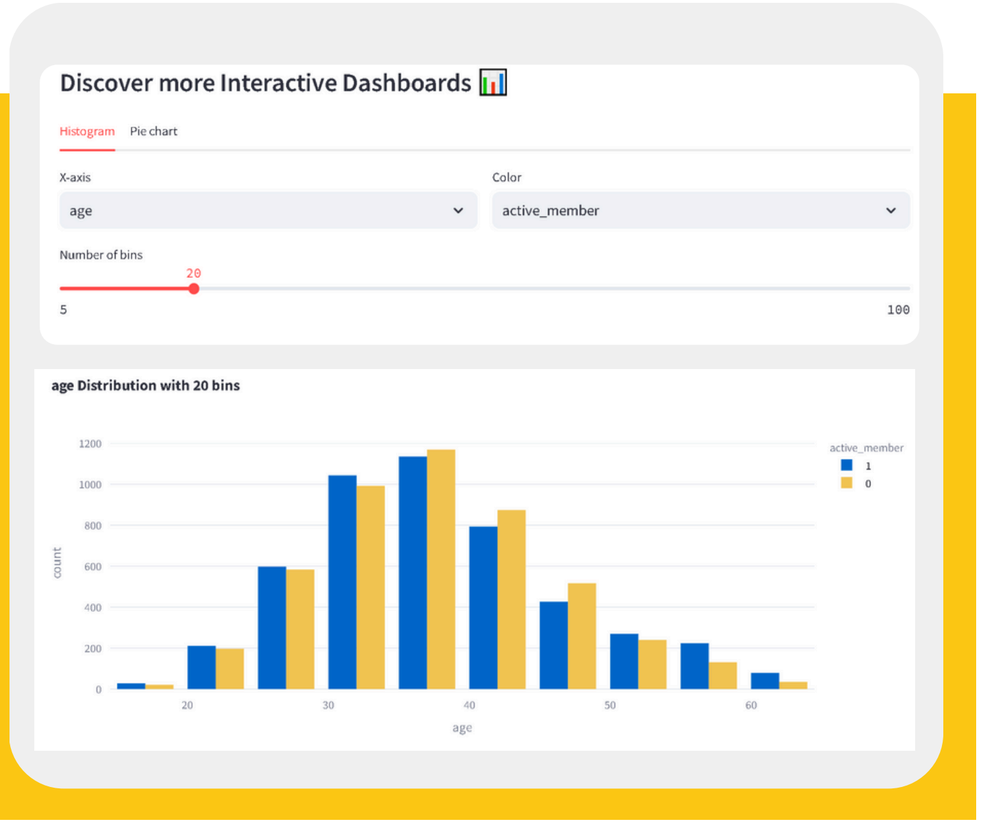
Churned Customers Distribution by Balance Segmentation



Interactive Dashboards

freely Unlocking Data Insights Through Visual Discovery





MODEL DEPLOYMENT

User friendly model to get predictoins

Please fill in the following details:

Enter the credit score

699

Select the gender of the customer

Female

Enter the number of years the customer has been with the bank

0

Enter the account balance

90100.00

Enter the age of the customer

57

Enter the number of bank products the customer uses

2

Does the customer have a credit card?

Yes

Is the customer an active member?

No

Enter the estimated annual salary

122911.58

Select the country of residence

Germany

Predict

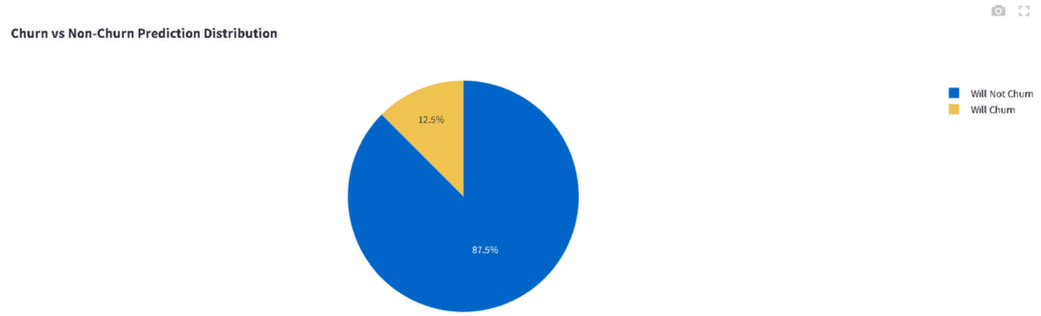
Session predictions history with ability to save it to CSV file

This customer will churn! 😢

Prediction History

	credit_score	gender	tenure	balance	age	products_number	credit_card	active_member	estimated_salary	country	prediction
0	500	Male	5	0	30	2	Yes	Yes	50,000	France	Will Not Churn
1	500	Female	0	0	37	2	No	No	50,000	Germany	Will Not Churn
2	500	Female	0	0	38	3	No	No	122,911.58	Germany	Will Not Churn
3	449	Female	3	3,400	38	3	No	No	122,911.58	Germany	Will Not Churn
4	449	Female	3	3,400	38	3	No	No	122,911.58	Germany	Will Not Churn
5	449	Female	3	3,400	38	3	No	No	122,911.58	Spain	Will Not Churn
6	699	Male	0	3,400	38	2	Yes	No	122,911.58	Spain	Will Not Churn
7	699	Female	0	90,100	57	2	Yes	No	122,911.58	Germany	Will Churn

Real time pie chart for your predictions



Model pipeline

We built and saved our own machine learning pipeline, which is loaded from a pickle file (model_pipeline.pkl) during deployment. This pipeline handles all pre-processing and model prediction steps. By feeding new data, as shown in the example, we can easily generate churn predictions, making the deployment process smooth and efficient.

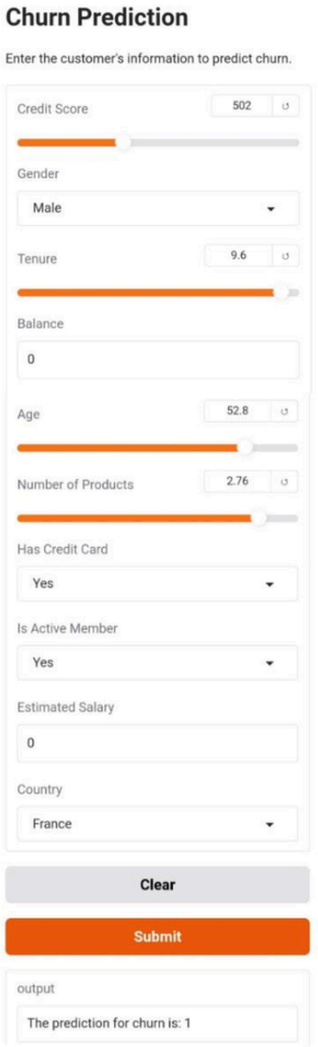
```python
import pandas as pd  # type: ignore
import joblib # type: ignore


model_pipeline = joblib.load('model_pipeline.pkl')

# Sample data for prediction (using the provided values)
sample_data = pd.DataFrame({
    'credit_score': [502],
    'gender': ['Female'],
    'tenure': [10],
    'balance': [159660.80],
    'age': [42],
    'products_number': [3],
    'credit_card': ['Yes'],
    'active_member': ['Yes'],
    'estimated_salary': [113931.57],
    'country': ['France']
})

prediction = model_pipeline.predict(sample_data)

# Print the prediction
print(f"The prediction for churn is: {prediction[0]}")
```

0.4s

```
The prediction for churn is: 1
```

HUGGING FACE

- We deployed our machine learning model on Hugging Face. Providing :
- an easy and user-friendly application.
- API integration.
- The platform offers scalability, security, and reliability, ensuring the model can handle increased demand.
- By using Hugging Face, we simplified the deployment process and made the model accessible for collaboration and further development.



Using the API

```python
from gradio_client import Client

client = Client("Youssef-Hatem/Banking-churn")
result = client.predict(
    credit_score=502,
    gender="Male",
    tenure=2,
    balance=3,
    age=42,
    products_number=3,
    credit_card="Yes",
    active_member="Yes",
    estimated_salary=3,
    country="France",
    api_name="/predict",
)
print(result)
```

```
 API: https://youssef-hatem-banking-churn.hf
ction for churn is: 1
```

Docker

Docker file

```
ockerfile > ...
FROM python:3.10-slim-bullseye

WORKDIR /app

COPY . /app
COPY requirements.txt /app/requirements.txt

RUN pip install -r requirements.txt

RUN pip install streamlit

EXPOSE 8501

CMD ["sh", "-c", "cd web_app && streamlit run 🏠_Home.py
```

Requirements file

requirements.txt

```
1  pandas
2  numpy
3  matplotlib
4  seaborn
5  plotly
6  scikit-learn
7  mlflow
8  jupyter
9  nbformat
10 joblib
11 streamlit
```

Instructions to build and run

Build and run doccker image

- first Command
- docker build -t final_depi .
- Second Command
- docker run -p 8501:8501 final_depi

Docker image

latest	In use	2 days ago	1.55 GB

APPLICATIONS

Git Hub & docker

Web App

Model deploy & API

THANK YOU