Data Science with Python

Data Exploration

	id	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	0	15674932	Okwudilichukwu	668	France	Male	33.0	3	0.00	2	1.0	0.0	181449.97	0
1	1	15749177	Okwudiliolisa	627	France	Male	33.0	1	0.00	2	1.0	1.0	49503.50	0
2	2	15694510	Hsueh	678	France	Male	40.0	10	0.00	2	1.0	0.0	184866.69	0
3	3	15741417	Kao	581	France	Male	34.0	2	148882.54	1	1.0	1.0	84560.88	0
4	4	15766172	Chiemenam	716	Spain	Male	33.0	5	0.00	2	1.0	1.0	15068.83	0

About Dataset

The bank customer churn dataset is a commonly used dataset for predicting customer churn in the banking industry. It contains information on bank customers who either left the bank or continue to be a customer. The dataset includes the following attributes:

1. Customer ID: A unique identifier for each customer
2. Surname: The customer's surname or last name
3. Credit Score: A numerical value representing the customer's credit score
4. Geography: The country where the customer resides (France, Spain or Germany)
5. Gender: The customer's gender (Male or Female)
6. Age: The customer's age.
7. Tenure: The number of years the customer has been with the bank
8. Balance: The customer's account balance
9. NumOfProducts: The number of bank products the customer uses (e.g., savings account, credit card)
10. HasCrCard: Whether the customer has a credit card (1 = yes, 0 = no)
11. IsActiveMember: Whether the customer is an active member (1 = yes, 0 = no)

12. EstimatedSalary: The estimated salary of the customer
13. Exited: Whether the customer has churned (1 = yes, 0 = no)

Data Summary

```
df_train.describe()
```

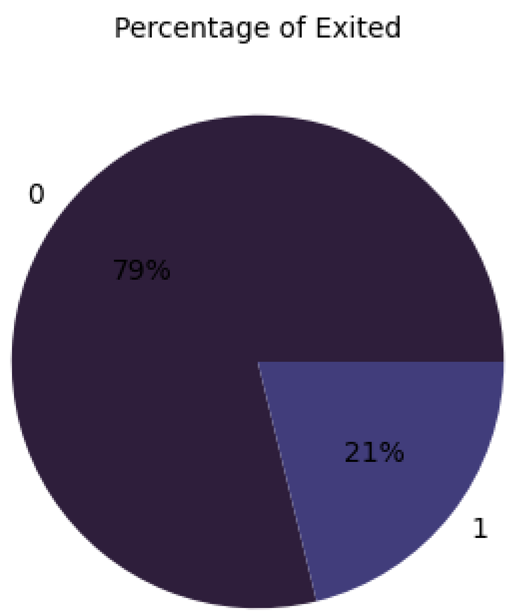
	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000	165034.000000
mean	656.454373	38.125888	5.020353	55478.086689	1.554455	0.753954	0.497770	112574.822734	0.211599
std	80.103340	8.867205	2.806159	62817.663278	0.547154	0.430707	0.499997	50292.865585	0.408443
min	350.000000	18.000000	0.000000	0.000000	1.000000	0.000000	0.000000	11.580000	0.000000
25%	597.000000	32.000000	3.000000	0.000000	1.000000	1.000000	0.000000	74637.570000	0.000000
50%	659.000000	37.000000	5.000000	0.000000	2.000000	1.000000	0.000000	117948.000000	0.000000
75%	710.000000	42.000000	7.000000	119939.517500	2.000000	1.000000	1.000000	155152.467500	0.000000
max	850.000000	92.000000	10.000000	250898.090000	4.000000	1.000000	1.000000	199992.480000	1.000000

Data Visualization

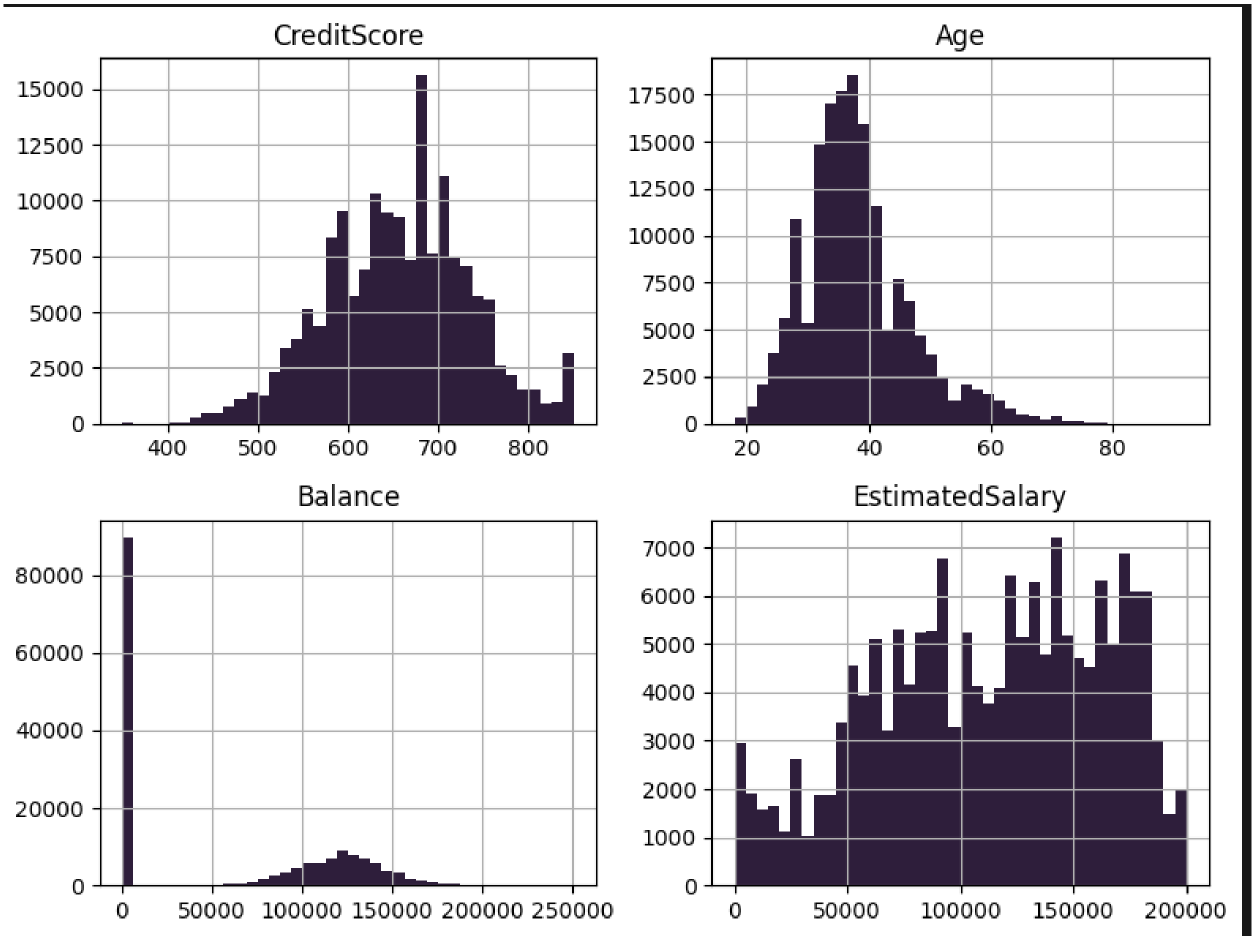
Categorical Columns: ['Geography','Gender','NumOfProducts','HasCrCard','Tenure','IsActiveMember']

Numerical Columns:['CreditScore', 'Age', 'Balance', 'EstimatedSalary']

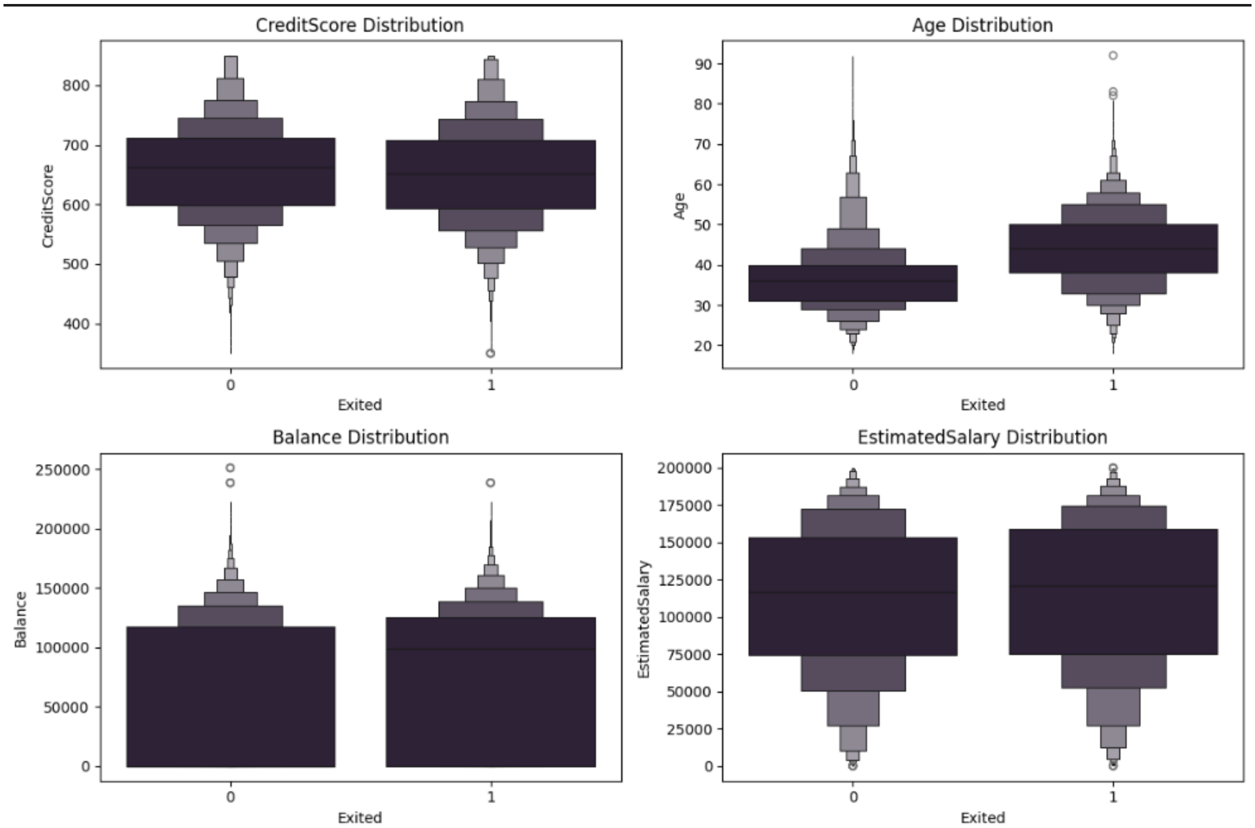
Target column "Exited" Visualization

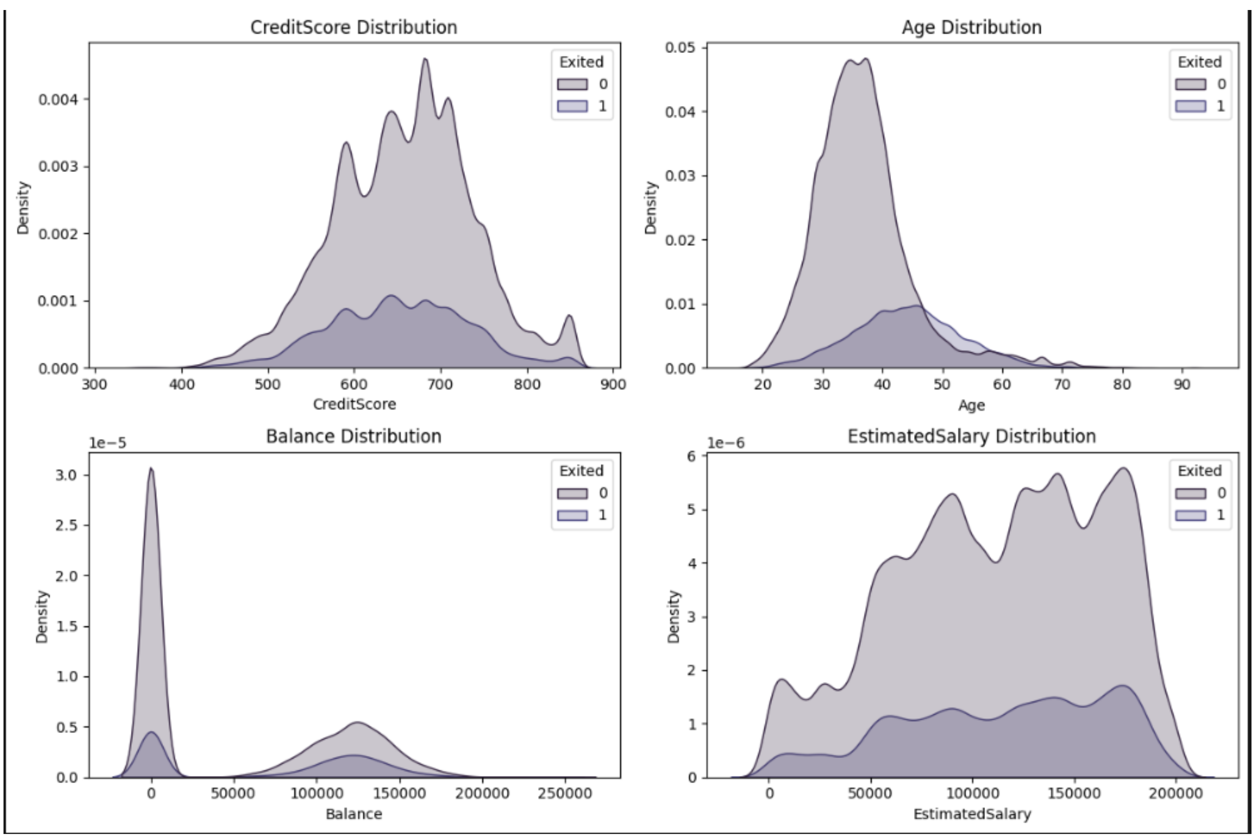


Plot Numerical Columns

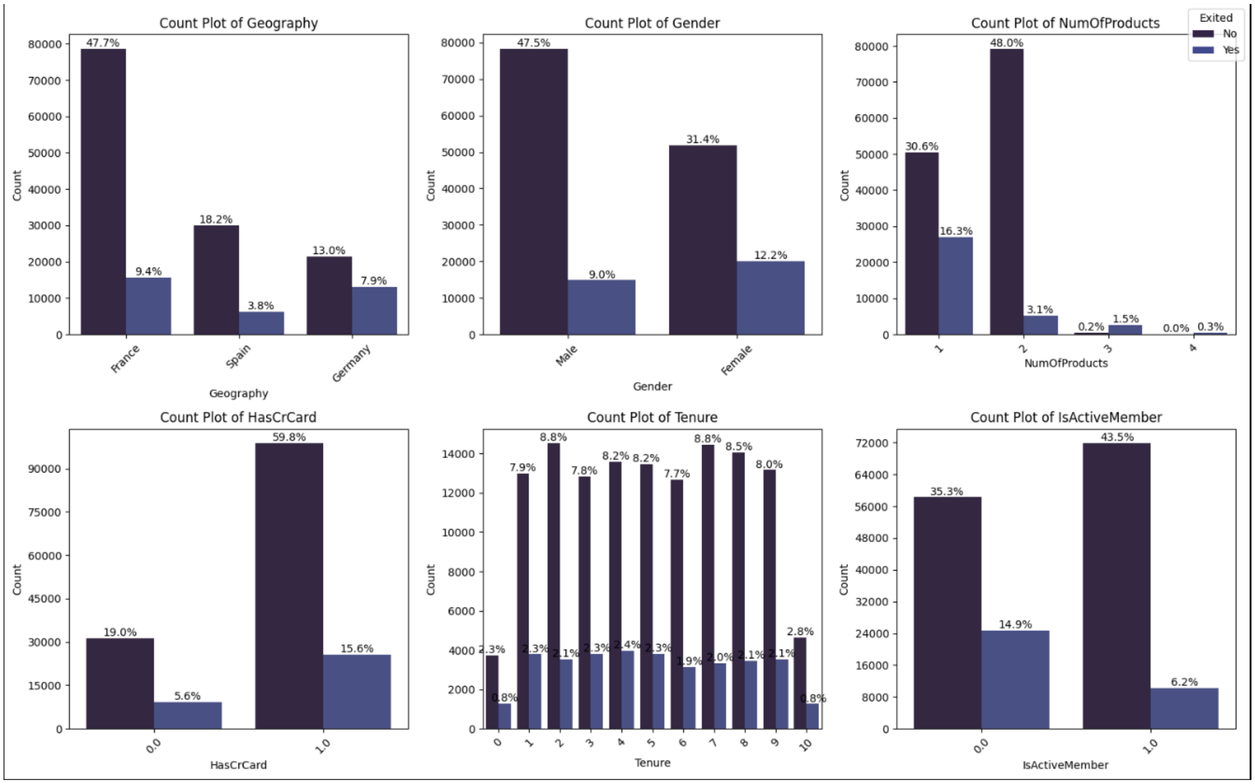


Plot Categorical Columns

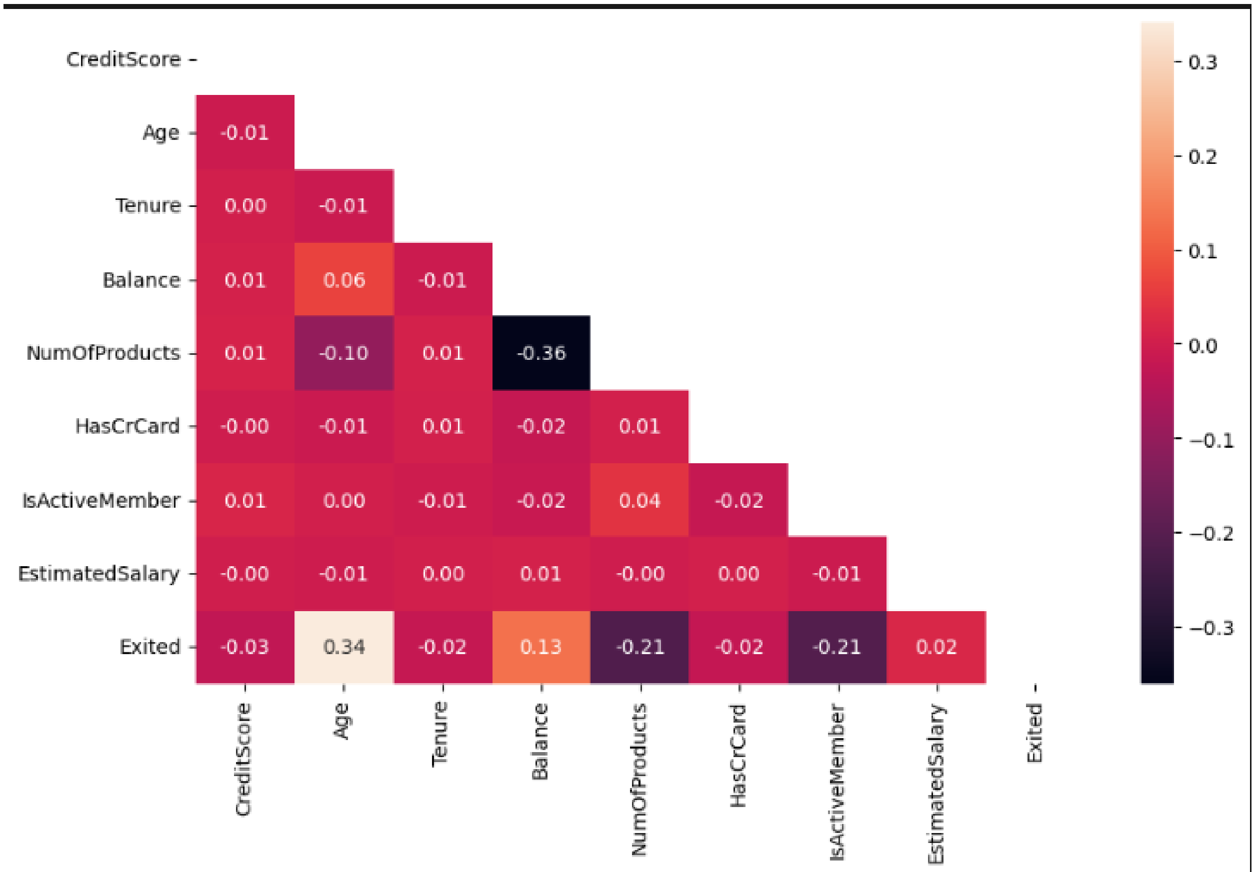

CreditScore Distribution
CreditScore
Exited
Age Distribution
Age
Exited
Balance Distribution
Balance
Exited
EstimatedSalary Distribution
EstimatedSalary
Exited


CreditScore Distribution
Density
CreditScore
Exited
0
1
Age Distribution
Density
Age
Exited
0
1
Balance Distribution
1e−5
Density
Balance
Exited
0
1
EstimatedSalary Distribution
1e−6
Density
EstimatedSalary
Exited
0
1

Categorical Columns EDA



Plotting Collinearity



ML Packages

ML Packages

+ Code + Markdown

```python
from sklearn.model_selection import cross_val_score,train_test_split
from sklearn.metrics import accuracy_score,classification_report,f1_score,mean_squared_error,roc_auc_score,precision_score,recall_score,roc_curve,ConfusionMatrixDisplay,confusion_matrix,auc
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,LabelEncoder,OneHotEncoder,OrdinalEncoder,RobustScaler
from sklearn.linear_model import LogisticRegression,SGDClassifier, RidgeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier,ExtraTreesClassifier,AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.base import BaseEstimator,TransformerMixin
from xgboost import XGBClassifier
from sklearn.impute import SimpleImputer,KNNImputer
from catboost import CatBoostClassifier
from sklearn.model_selection import RandomizedSearchCV
import category_encoders as ce
```

Preprocessing Pipeline

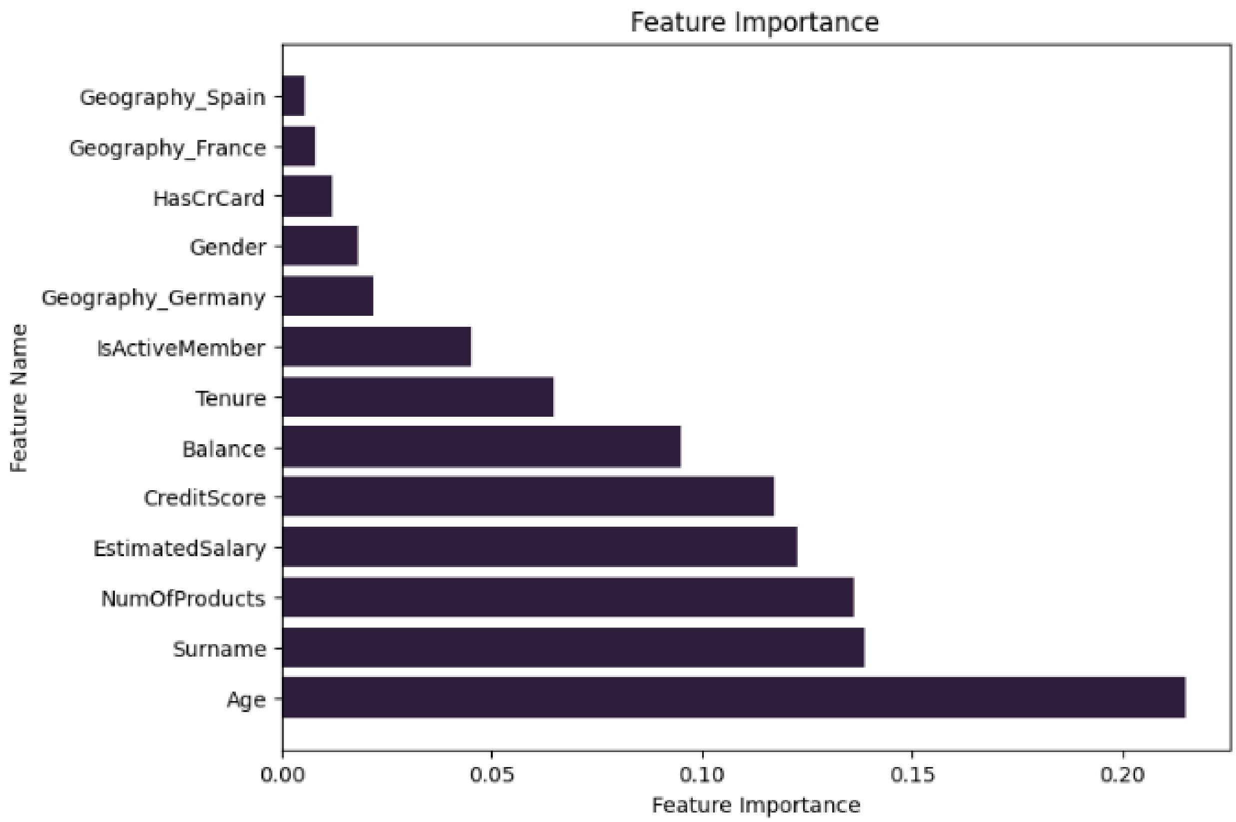
```python
class FullPipeline1:
    def __init__(self):
        self.numerical_cols = ['CreditScore', 'Age', 'Balance', 'EstimatedSalary']
        self.categorical_cols = ['Geography', 'Gender', 'NumOfProducts', 'HasCrCard', 'Tenure', 'IsActiveMember']
        self.OD_cols = ['Gender', 'NumOfProducts', 'HasCrCard', 'Tenure', 'IsActiveMember']
        self.OH_cols = ['Geography']
        self.TE_cols = ['Surname']
        self.full_pipeline = Pipeline([
            ('impute_num', DataFrameImputer(knn_cols=self.numerical_cols)),
            ('impute_cat', DataFrameImputer(freq_cols=self.categorical_cols)),
            ('scale', StandardScaleTransform(self.numerical_cols)),
            ('ordinal_encode', OrdinalEncodeColumns(self.OD_cols)),
            ("one_hot_encode", CustomOneHotEncoder(self.OH_cols)),
            ('target_encode', TargetEncoderTransformer(self.TE_cols))
        ])
    def fit(self, X_train, y_train):
        self.full_pipeline.fit(X_train, y_train)

    def transform(self, X_test):
        return self.full_pipeline.transform(X_test)
    def fit_transform(self, X_train, y_train):
        X_train = self.full_pipeline.fit_transform(X_train, y_train)
        return X_train, y_train

f1 = FullPipeline1()
```

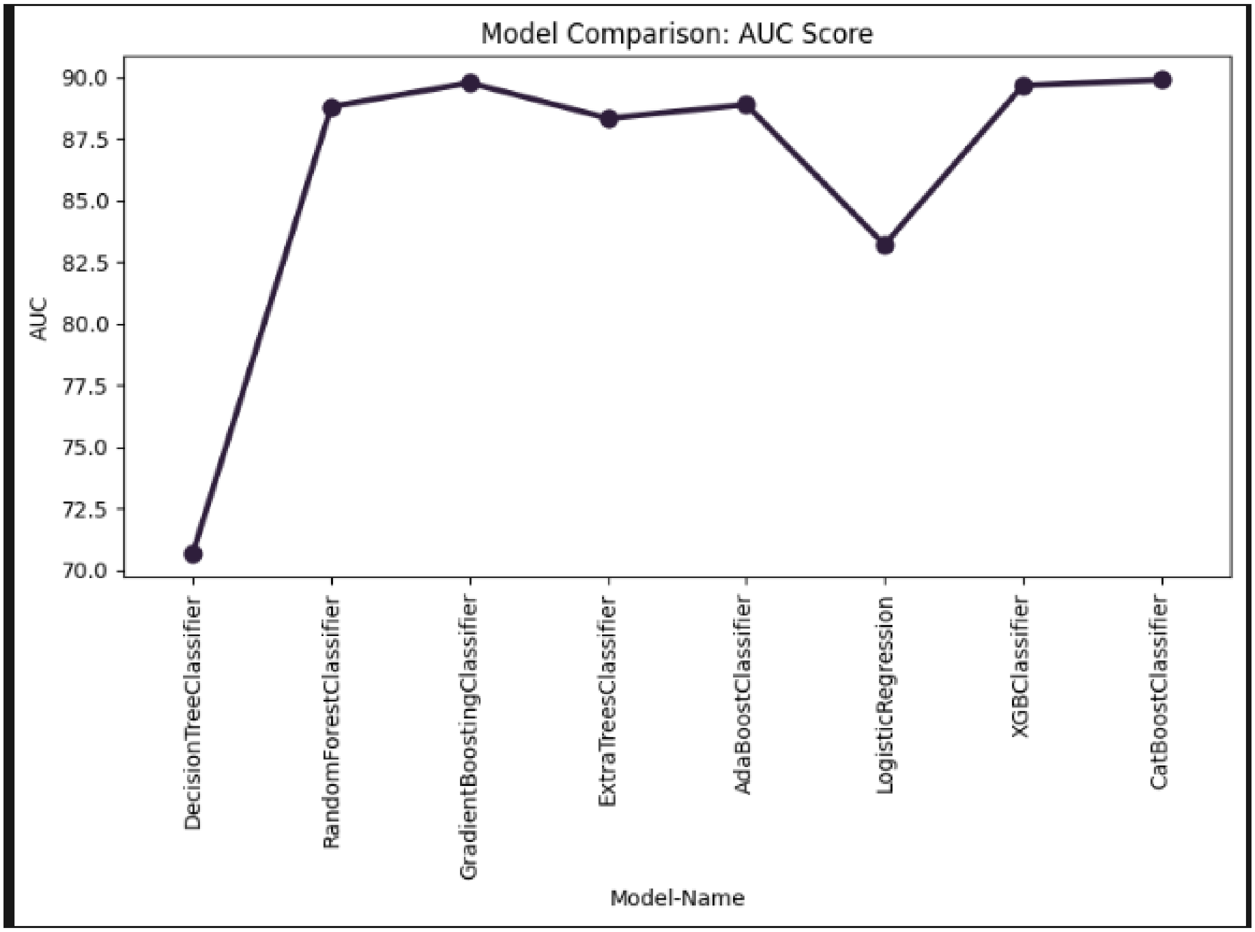
Step	Description
1	Numerical Imputation (KNN): CreditScore, Age, Balance, EstimatedSalary
2	Categorical Imputation (Frequency): Geography, Gender, NumOfProducts, HasCrCard, Tenure, IsActiveMember
3	Scaling Numerical Columns: CreditScore, Age, Balance, EstimatedSalary
4	Ordinal Encoding: Gender, NumOfProducts, HasCrCard, Tenure, IsActiveMember
5	One-Hot Encoding: Geography
6	Target Encoding: Surname

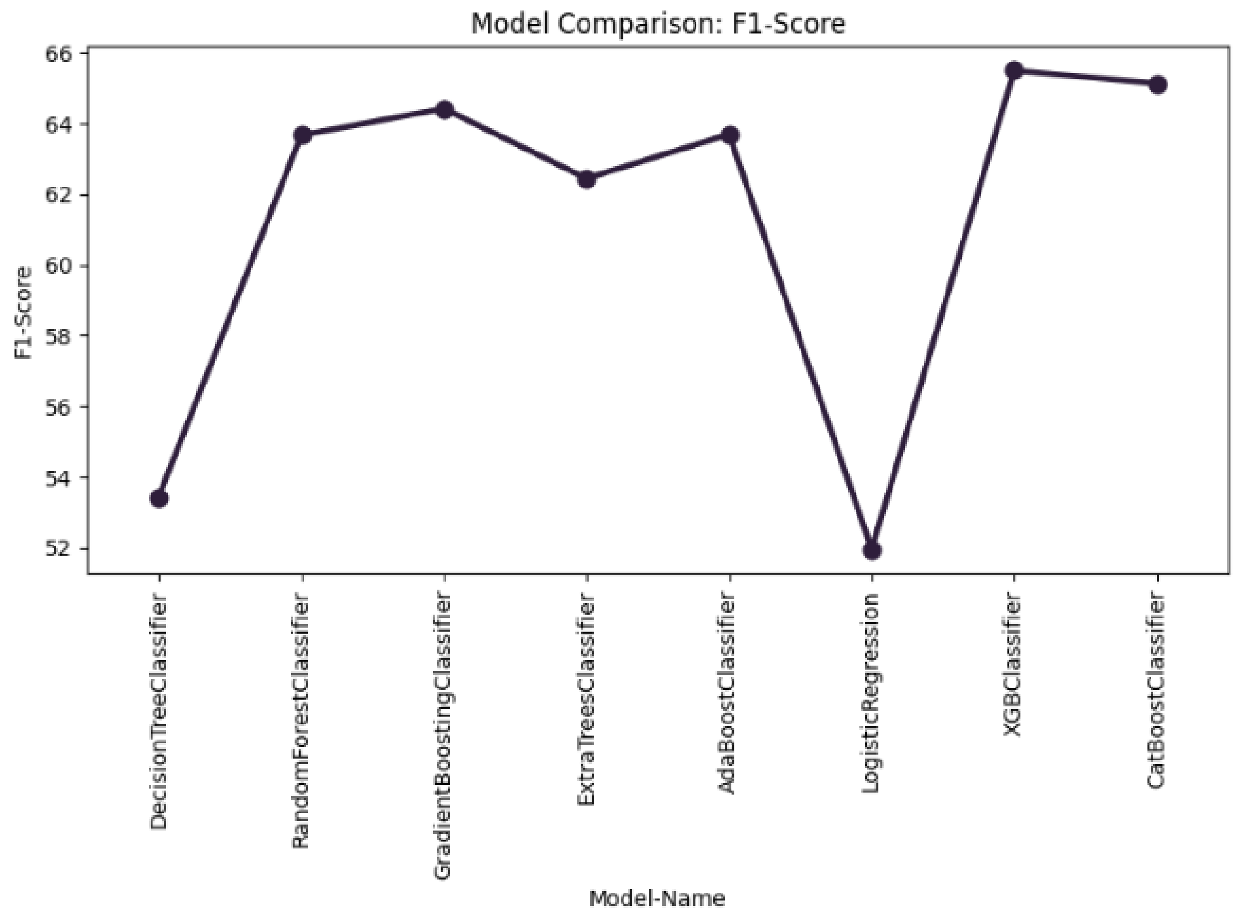
Feature Importance



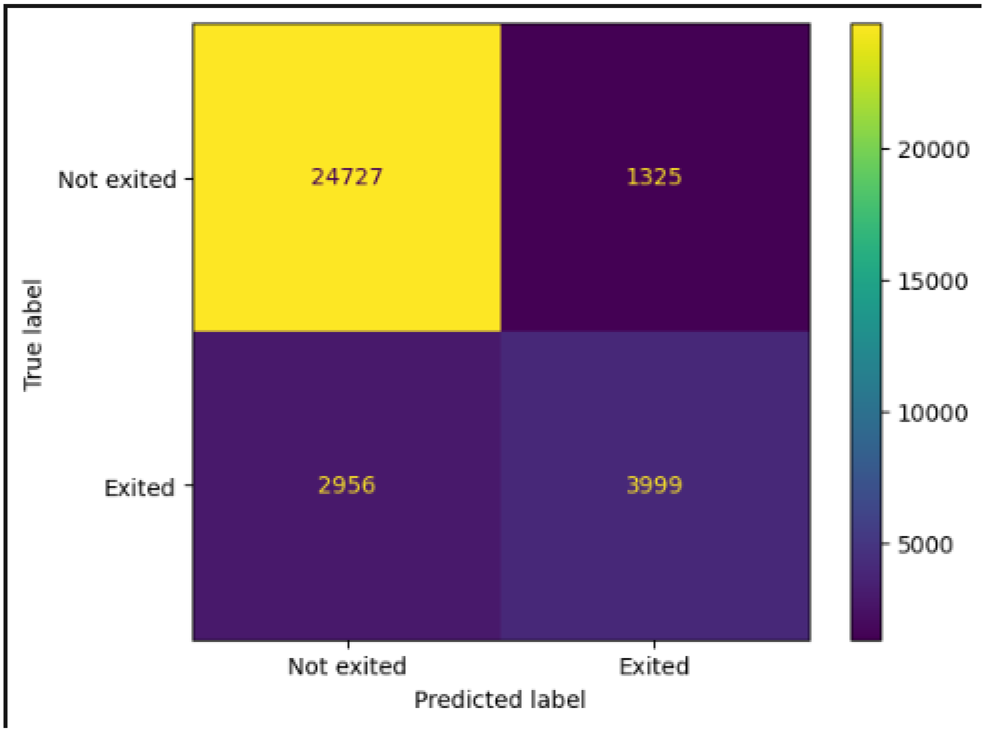
Model Evaluation

	Model-Name	Accuracy	AUC	F1-Score
7	CatBoostClassifier	87.030024	89.883171	65.135597
2	GradientBoostingClassifier	86.914897	89.766216	64.426324
6	XGBClassifier	86.987609	89.661380	65.504779
4	AdaBoostClassifier	86.705850	88.877745	63.699537
1	RandomForestClassifier	86.584664	88.803202	63.675144
3	ExtraTreesClassifier	86.239283	88.326611	62.450397
5	LogisticRegression	83.876147	83.196907	51.958837
0	DecisionTreeClassifier	79.849729	70.702893	53.440672

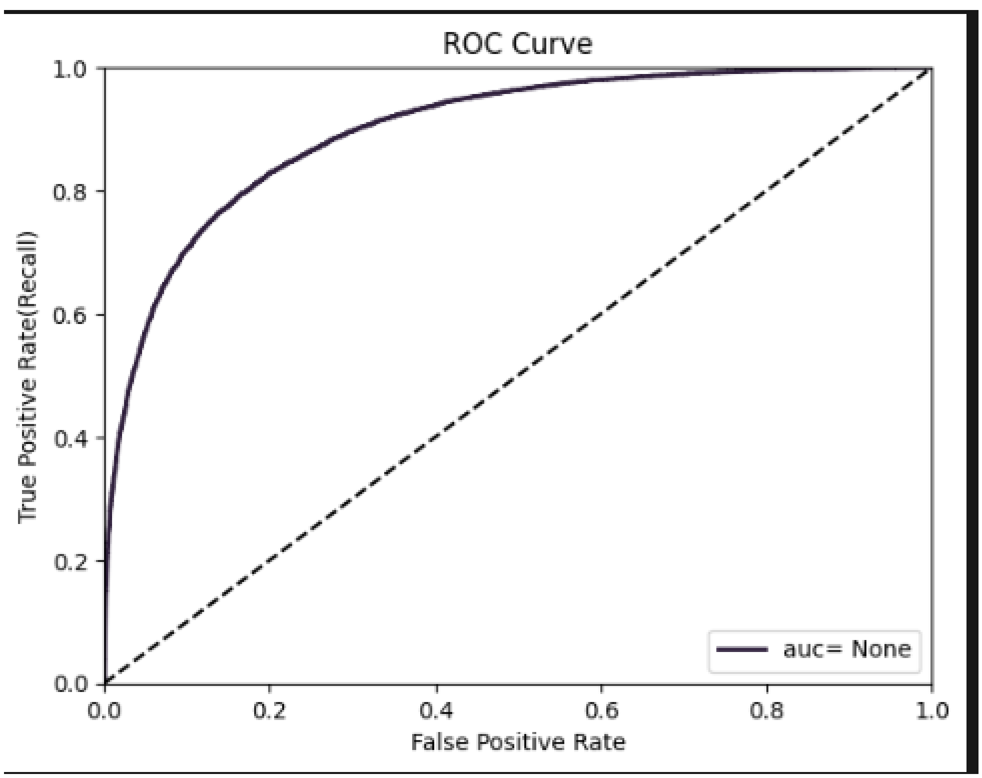

Model Comparison: AUC Score
90.0
87.5
85.0
82.5
80.0
77.5
75.0
72.5
70.0
AUC
DecisionTreeClassifier
RandomForestClassifier
GradientBoostingClassifier
ExtraTreesClassifier
AdaBoostClassifier
LogisticRegression
XGBClassifier
CatBoostClassifier
Model-Name



Confusion Matrix


24727
1325
2956
3999
Not exited
Exited
True label
Not exited
Exited
Predicted label
20000
15000
10000
5000

ROC Plot


ROC Curve
True Positive Rate(Recall)
False Positive Rate
auc= None
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

Final Model

submission.csv Complete (after deadline) · 4d ago	**0.89191**	**0.88701**