Churn classification

Definition

Project Overview

Businesses face a significant challenge with customer churn, which occurs when customers stop using a company's products or services. High churn rates can have detrimental effects on a company's revenue and profitability [1]. To address this challenge, machine learning algorithms can be used to identify the factors that contribute to churn. Churn models are designed to detect early warning signs and recognize customers who are more likely to leave voluntarily [2], [3], and [4]. In this project, we explore three different algorithms: logistic regression, decision tree, and random forest. By utilizing these three powerful tools, we aim to develop a highly accurate classifier that can predict which customers are likely to churn and which are not. We obtained the churn dataset from Kaggle, specifically the Bank Customer Churn Dataset, and trained three different classification models: Logistic Regression, Decision Tree, and Random Forest. The performance of these models was evaluated using the AUC-ROC score.

Problem Statement

In today's fiercely competitive industry, customer churn presents a significant challenge and is crucial to the success of many businesses. Churn occurs when a customer decides to switch to a competitor that offers a better product or solution. It is essential for businesses to identify customers who are likely to churn and implement effective retention strategies for long-term success, given that acquiring new customers is more expensive than retaining current ones. However, manually identifying such customers or using conventional techniques can be time-consuming and challenging. Therefore, developing machine learning techniques that can accurately identify customers at risk of churning based on their past behavior and interactions can improve the ability to spot and prevent future churning customers. The aim of this project is to build a model capable of accurately predicting churn and improving customer retention

Dataset description

Column	Description
customer_id	The first feature in the dataset is a unique identifier assigned to each customer. This column is not used as an input for the model and is solely for identification purposes.
credit_score	The second feature in the dataset is a numerical value that represents the creditworthiness of a customer. This feature is used as an input to the model to predict churn, where higher scores indicate lower risk and vice versa.
country	The third feature in the dataset is the country where the customer resides. This feature is used as an input to the model to predict churn.
gender	The gender of the customer. It is used as an input to the model to predict churn.
age	the gender of the customer. This feature is utilized as an input to the model to predict churn.
tenure	the number of years that the customer has been with the bank. This feature is utilized as an input to the model to predict churn.
balance	the amount of money that the customer has in their account as another feature. This feature is utilized as an input to the model to predict churn.
products_number	The number of banking products that the customer has with the bank. It is used as an input to the model to predict churn.
credit_card	whether the customer has a credit card with the bank or not. This feature is utilized as an input to the model to predict churn.
active_member	Whether the customer is an active member of the bank or not. It is used as an input to the model to predict churn.
estimated_salary	The estimated salary of the customer. It is used as an input to the model to predict churn.
churn	The target variable is a binary feature that indicates whether the customer has left the bank or not during a certain period. This variable takes a value of 1 if the customer has left and 0 if they haven't. The target variable is used to train and evaluate the machine learning model for predicting customer churn

In this project, the dataset will be divided into three subsets: **Training (70%), Validation (20%),** and **Test (10%).** This division allows us to train the model on the training set, tune its hyperparameters using the validation set, and evaluate its performance on the test set. The use of three separate subsets helps to prevent overfitting and ensures that the model generalizes well to new, unseen data.

Metrics

During the training, evaluation, and hyperparameter tuning of the models, I used cross-validation accuracy as my primary metric. To compare the performance of the models, I computed the AUC-ROC score against the test data and also calculated the accuracy score for the test data. Additionally, I generated a confusion matrix to better understand the model's performance.

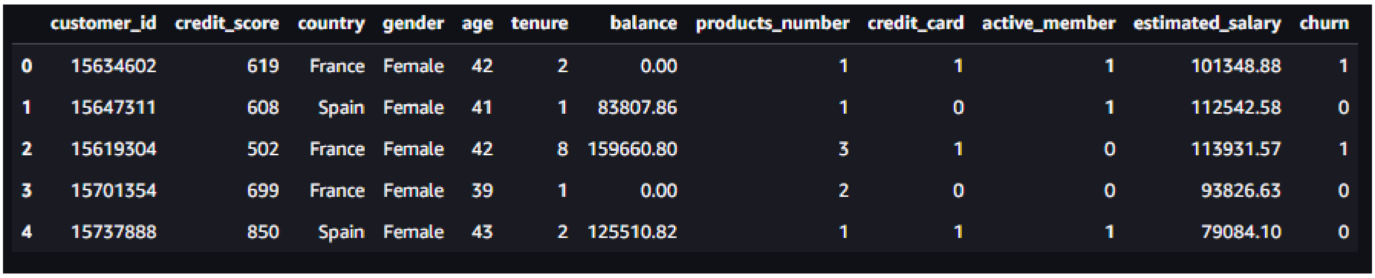
Analysis

Data Exploration

The dataset contains 10,000 observations with 12 features and is classified into two classes: 0 (not churn or negative class) and 1 (churn or positive class) , no null values in all features

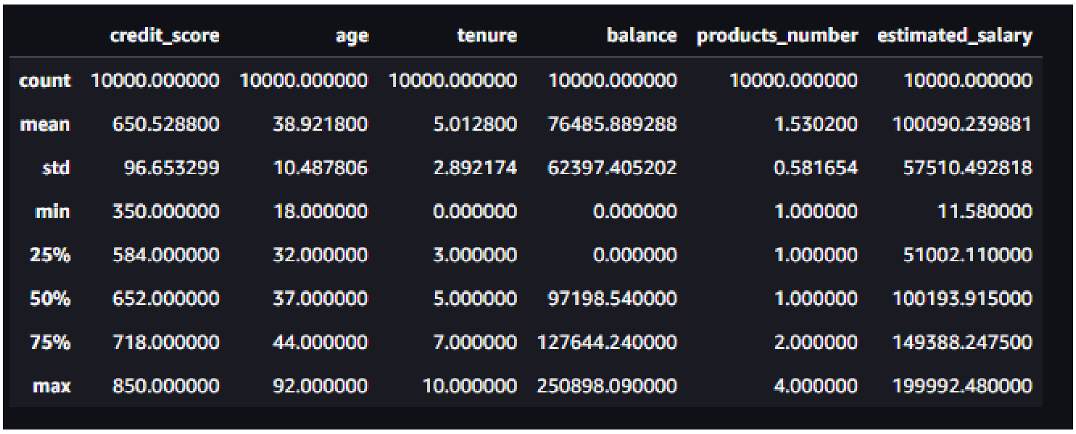
Figure 1 shows a subset of the dataset, while Figure 2 presents the statistics.

Figure 1: Dataset features

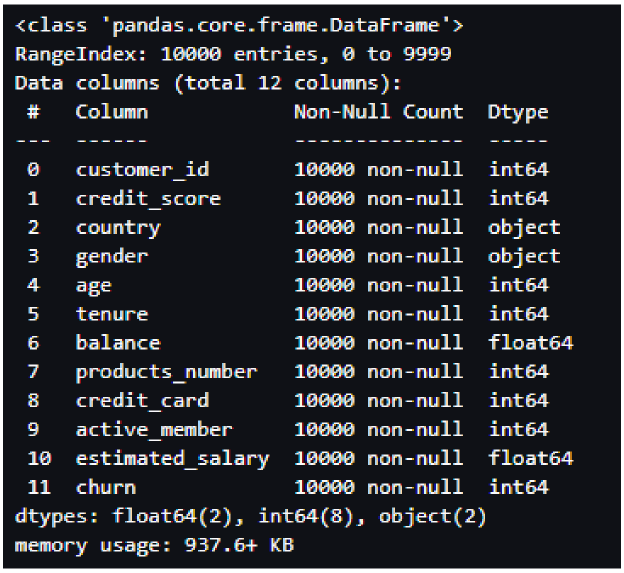


	customer_id	credit_score	country	gender	age	tenure	balance	products_number	credit_card	active_member	estimated_salary	churn
0	15634602	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	15647311	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	15619304	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	15701354	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	15737888	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0

Figure 2: Dataset statistics



	credit_score	age	tenure	balance	products_number	estimated_salary
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	650.528800	38.921800	5.012800	76485.889288	1.530200	100090.239881
std	96.653299	10.487806	2.892174	62397.405202	0.581654	57510.492818
min	350.000000	18.000000	0.000000	0.000000	1.000000	11.580000
25%	584.000000	32.000000	3.000000	0.000000	1.000000	51002.110000
50%	652.000000	37.000000	5.000000	97198.540000	1.000000	100193.915000
75%	718.000000	44.000000	7.000000	127644.240000	2.000000	149388.247500
max	850.000000	92.000000	10.000000	250898.090000	4.000000	199992.480000

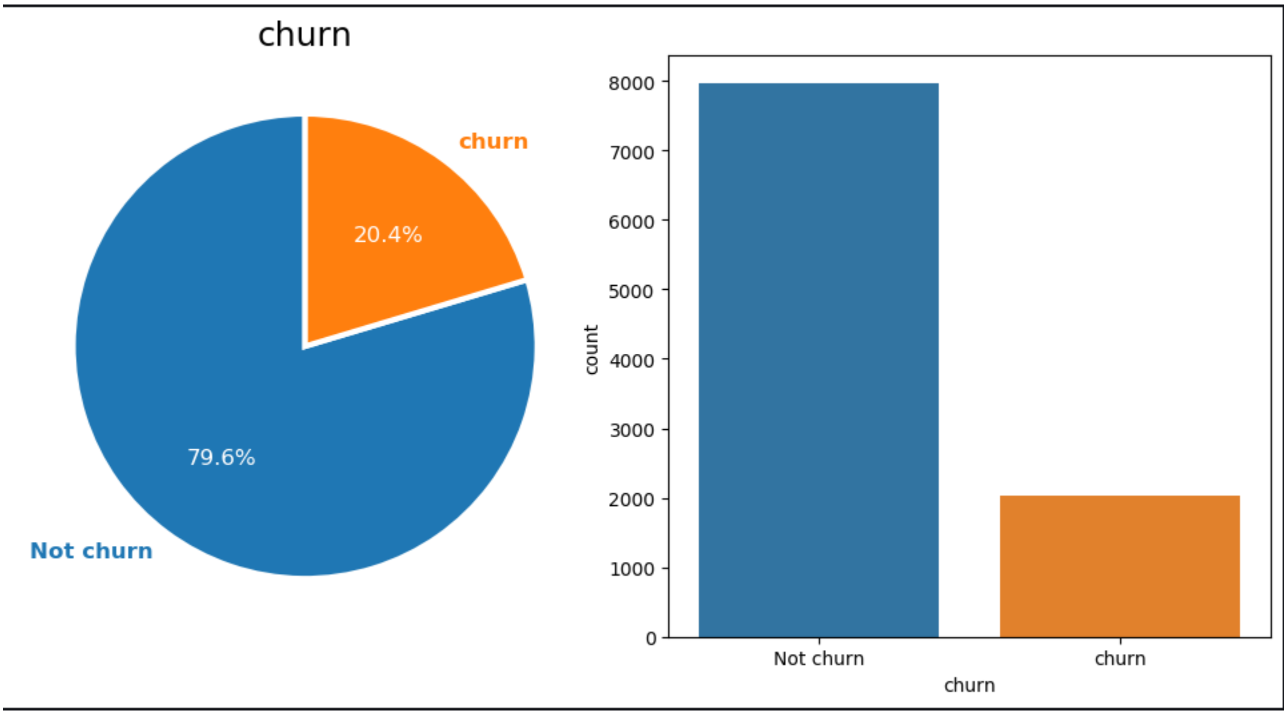


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   customer_id       10000 non-null  int64  
 1   credit_score      10000 non-null  int64  
 2   country           10000 non-null  object 
 3   gender            10000 non-null  object 
 4   age               10000 non-null  int64  
 5   tenure            10000 non-null  int64  
 6   balance           10000 non-null  float64
 7   products_number   10000 non-null  int64  
 8   credit_card       10000 non-null  int64  
 9   active_member     10000 non-null  int64  
 10  estimated_salary  10000 non-null  float64
 11  churn             10000 non-null  int64  
dtypes: float64(2), int64(8), object(2)
memory usage: 937.6+ KB
```

Exploratory Visualization

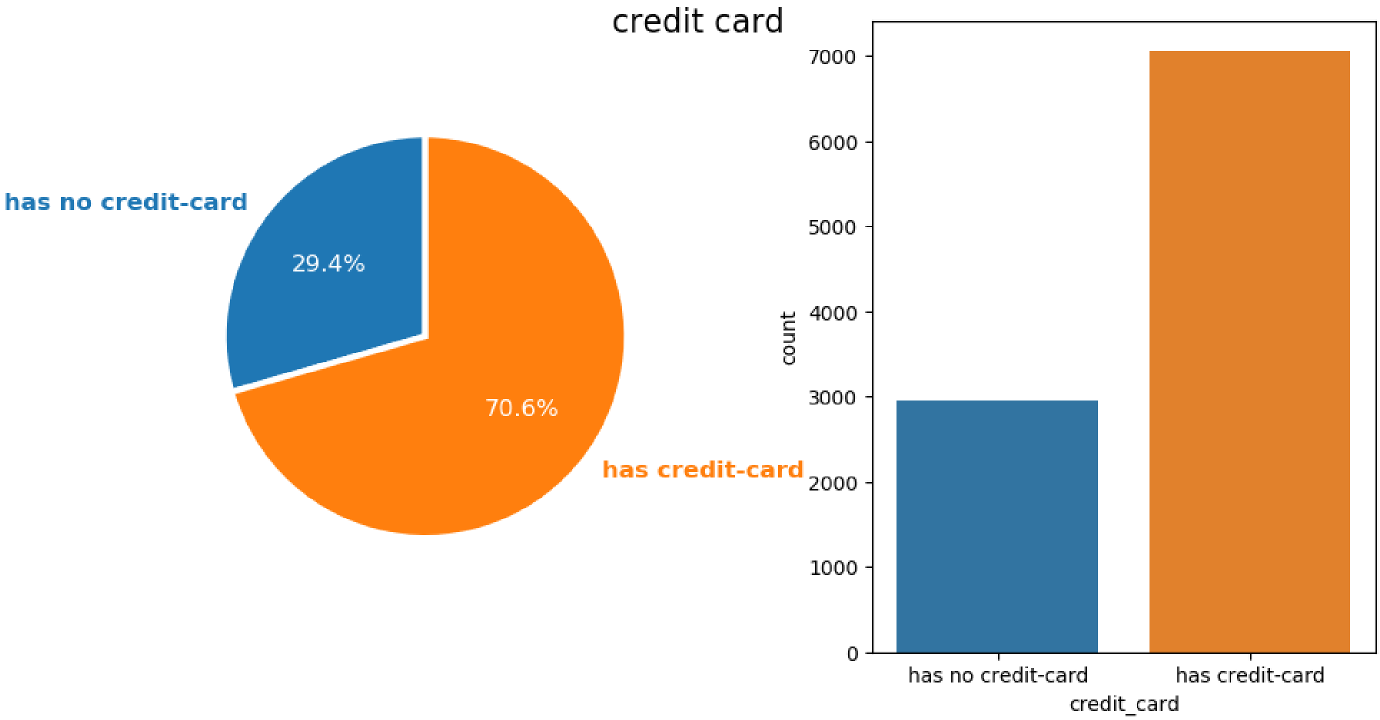
Churn column

total number of customers in the dataset, 20.4% (2037) are churning customers and 79.6% (7963) are not churning customers.



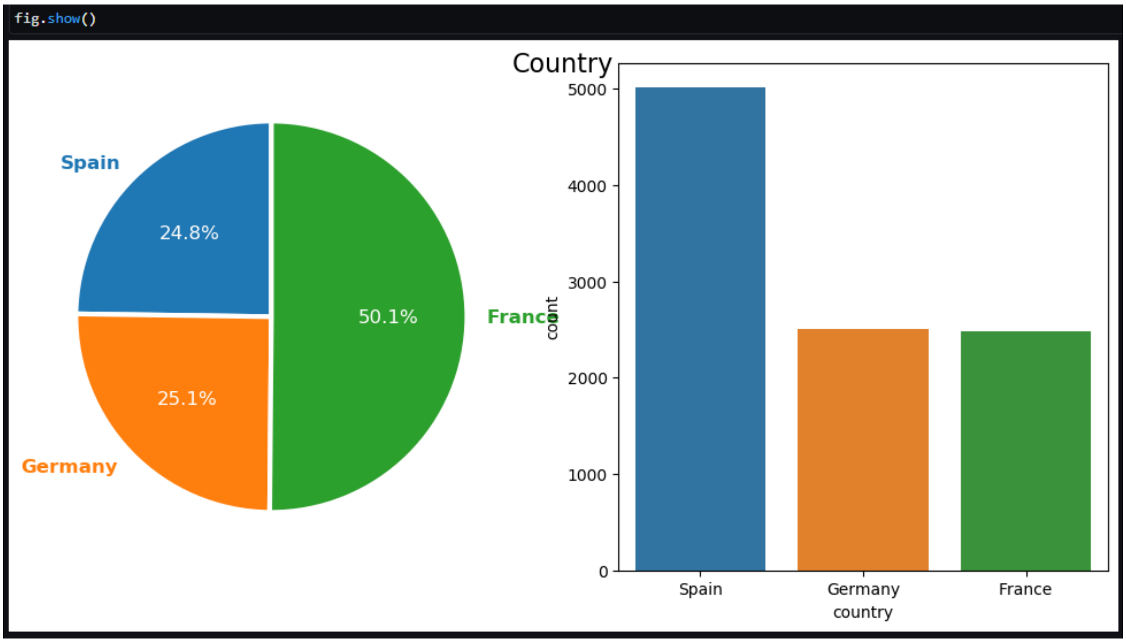
Credit Card

No Customers that have credit card vs non



Country

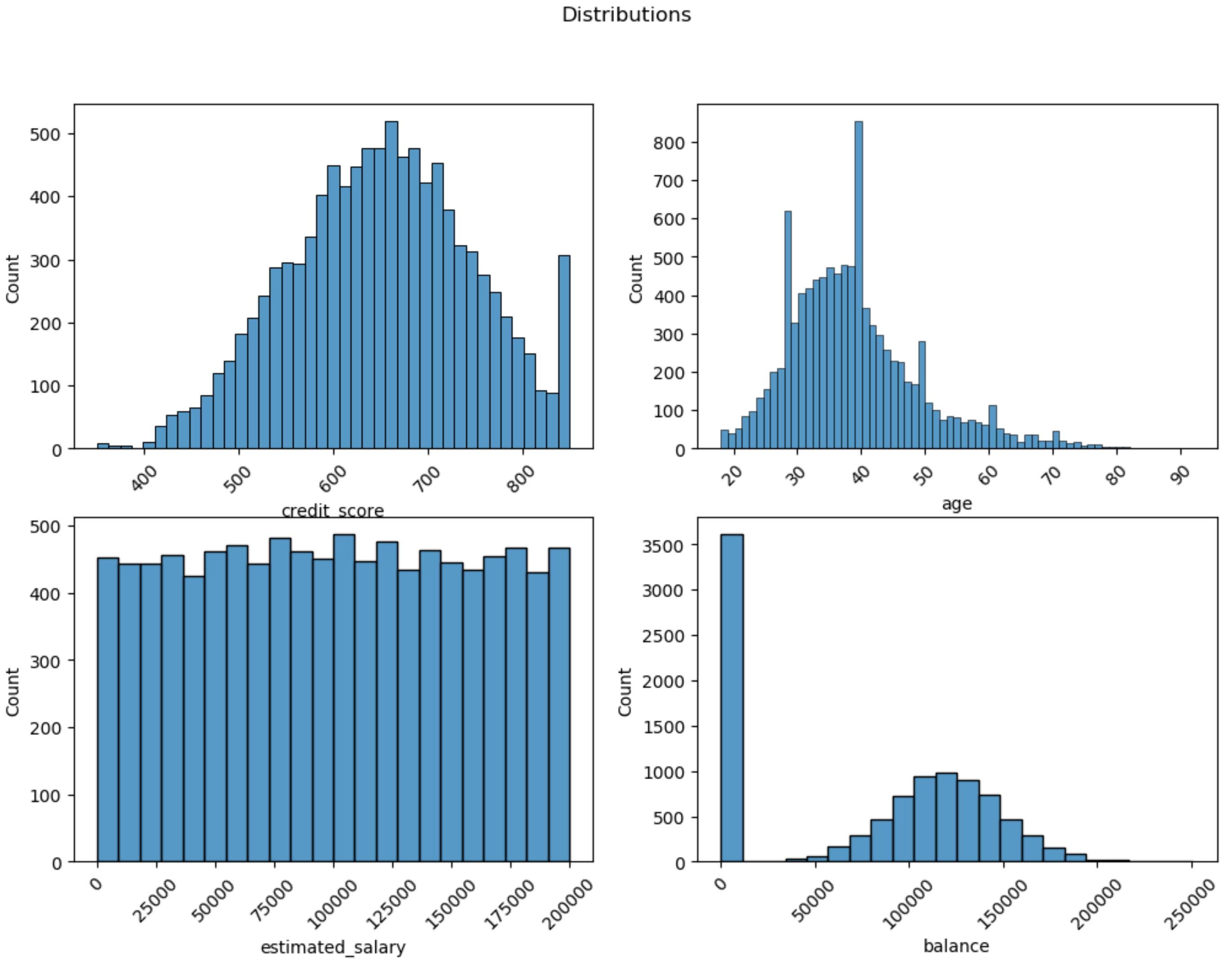
Count and percentage of each country in the data



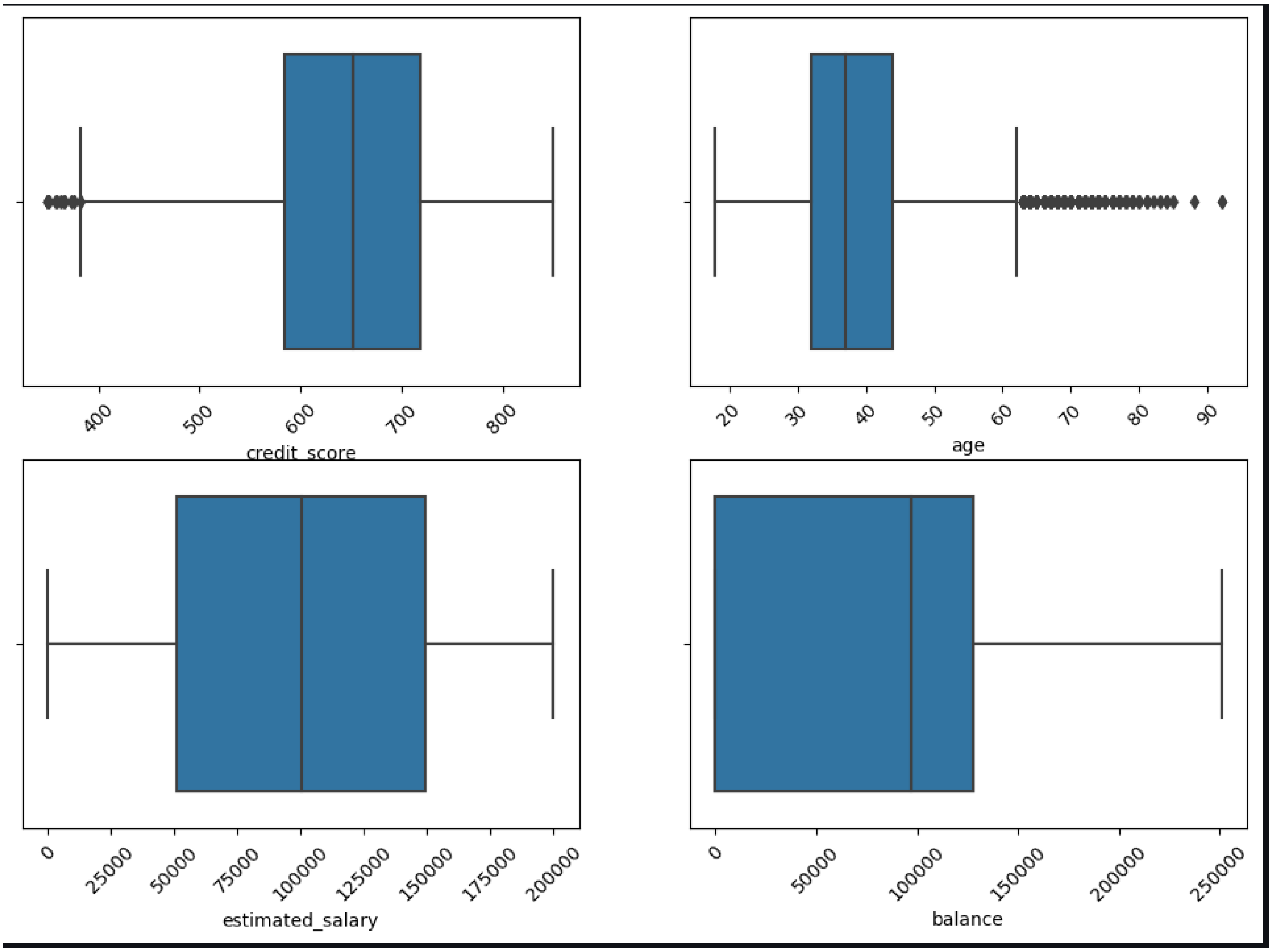
```
fig.show()
```

Continuous features distributions:

Credit score, age, estimated salary, account balance distributions



Continuous features ranges and outliers:



Algorithms and Techniques

Supervised classification models are algorithms that use labeled data to classify new, unseen data. These models are commonly applied in tasks such as image classification, text classification, and speech recognition. There are two categories of supervised classification models: linear models and nonlinear models.

Linear models use a linear decision boundary to separate observations into classes, while nonlinear models can capture complex relationships between input features and output classes using nonlinear decision boundaries. Logistic Regression is a popular linear model for binary classification tasks, where it generates a probability value between 0 and 1 using a logistic function to represent the likelihood of belonging to a specific class.

Decision Trees are another type of supervised learning algorithm that can perform both classification and regression tasks. They are structured like a tree, where each internal node represents a test on a feature, each branch denotes the outcome of the test, and each leaf node represents a class label or numerical value. Decision Trees use Gini impurity or Entropy to select the feature that can best split the data into homogeneous subsets.

Random Forest is an ensemble learning algorithm that uses a method called bagging to train multiple models on different subsets of the training data and combine their results to make a final prediction. It uses a large number of decision trees, and their results are combined to make a final prediction. Random Forest differs from Decision Trees in how it selects the best feature among a random subset of features, which can reduce overfitting and improve the model's accuracy.

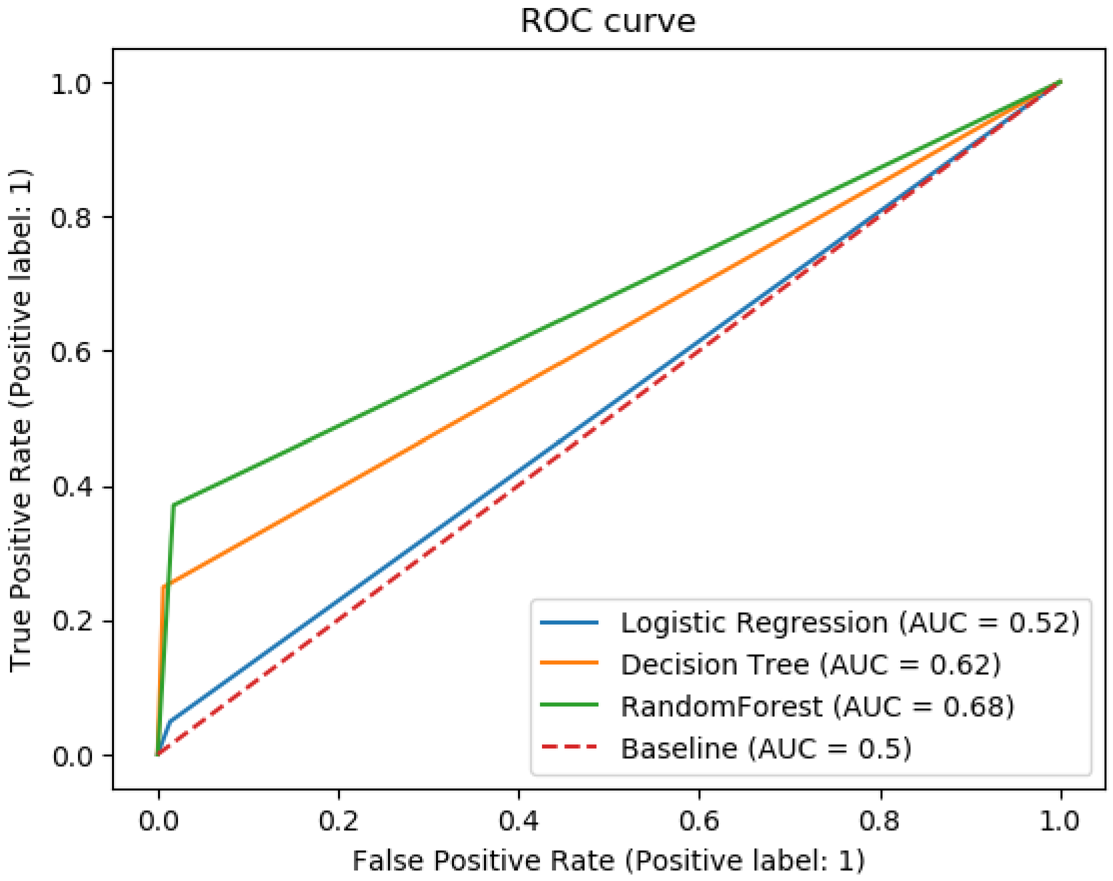
Logistic Regression is a simple, interpretable model that can handle both numerical and categorical input variables, but may not perform well with highly nonlinear relationships or interactions between input variables. Decision Trees can handle both categorical and continuous input features, but can overfit the training data, which can be mitigated by pruning and setting minimum samples to split an internal node. Random Forest can handle high-dimensional data, nonlinear relationships between features, and missing data, but requires careful tuning of hyperparameters and can be computationally expensive to train.

For this project, I trained and evaluated several models on the training data, then used Synthetic Minority Oversampling Technique SMOTE to address the imbalance in the dataset. I retrained and re-evaluated the models on this new dataset, then optimized their performance using hyperparameter tuning and compared them based on their final AUC-ROC score.

Benchmark

To establish a baseline for model performance, default configurations of the aforementioned models were used while working with imbalanced data. The results are presented in Figure 5, which displays the ROC curves and AUC scores for each model, as well as a dashed line baseline that represents a random classification model

ROC curves and AUC scores for benchmark models.



Methodology

Data Download

To retrieve the dataset, I downloaded the data directly from kaggle and add it to sagemaker notebook environment

Data Preprocessing

Changed datatypes:

Gender→category
credit_card→category
active_member→category
country→category
churn →category
credit_card→category

Dropped features:

customer_id → unique identifier doesn’t represent anything

do hot-encoding to:

country and gender features

Implementation

To implement the machine learning training and testing, the scikit-learn library was utilized due to its easy-to-use interface and efficient workflows, which include the use of pipelines for combining tasks such as transformation, dimensionality reduction, and training. The SageMaker script mode was also used to enable training and testing of models from scratch. For this purpose, Python scripts were created for each model, along with a script that integrated Sckit-learn pipelines.

I tried three algorithms

1. Logistic regression
2. Random forest
3. Decision tree

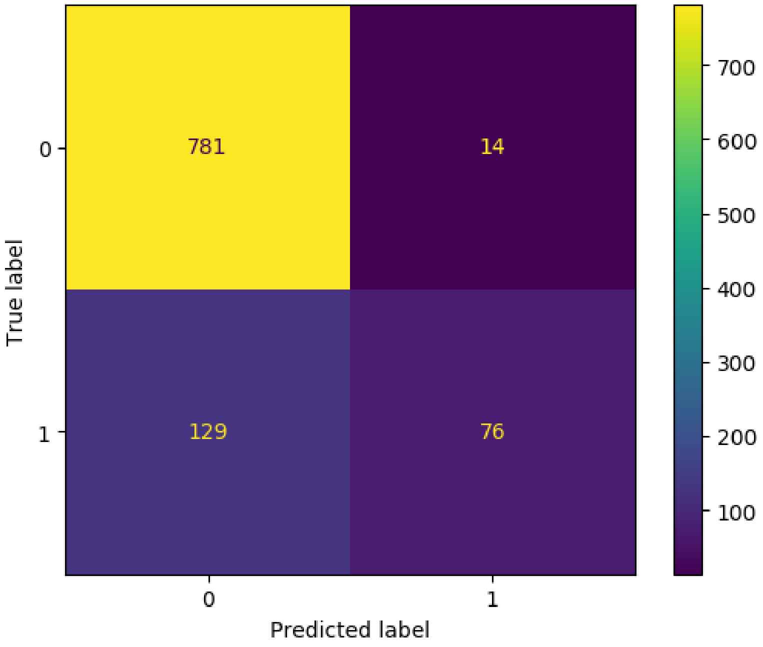
You can find the script for each one of them in src folder

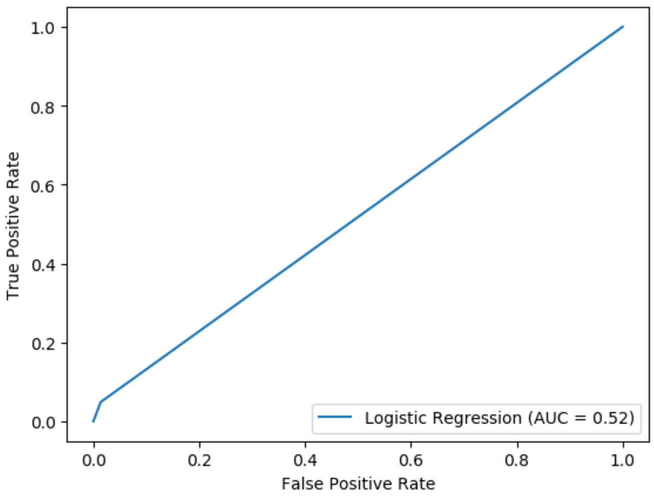
Results

Model Evaluation and Validation

To assess the performance of the models, a validation set was used during training to conduct k-fold cross-validation and calculate the AUC score. Subsequently, a separate test set was employed to validate the models and compute the final accuracy and AUC. To gain further insights into the results, confusion matrices were generated for the final models.

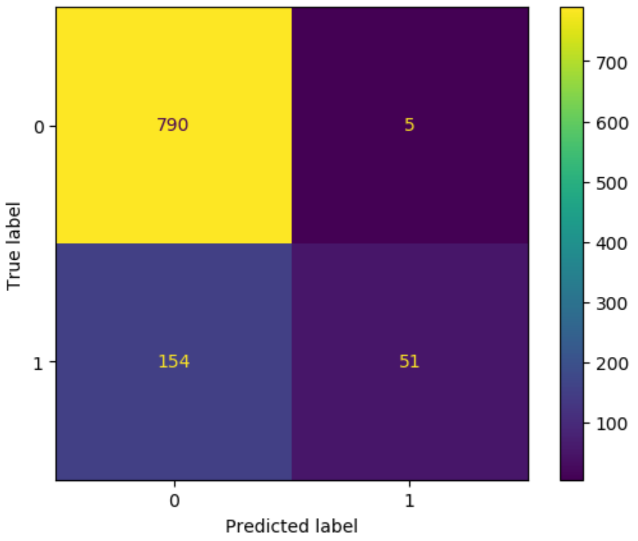
Logistic regression

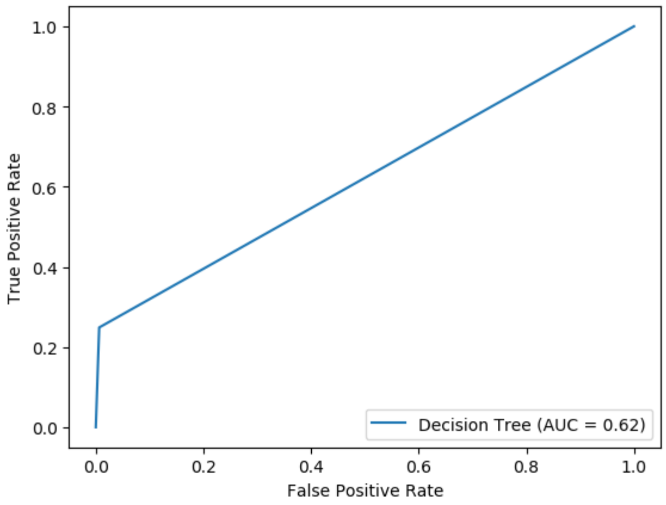




Accuracy → 79.4%

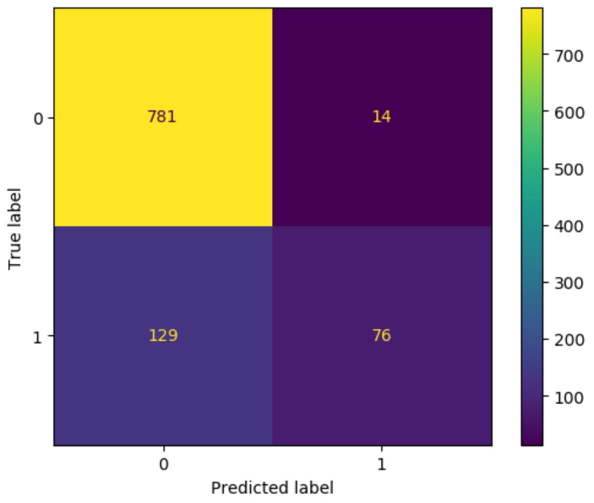
Decision Tree

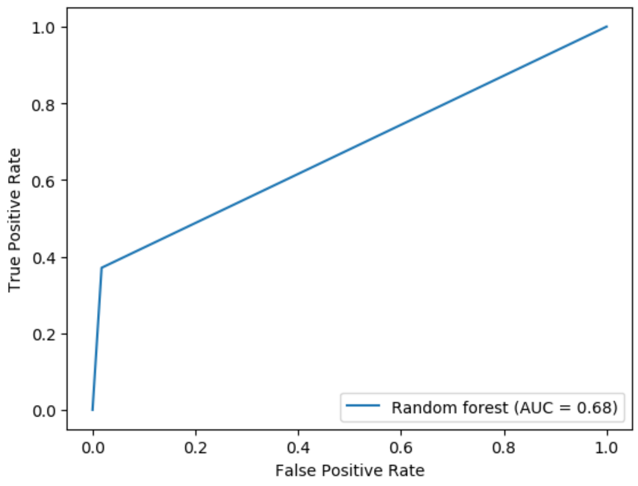




Accuracy → 84.1%

Random forest

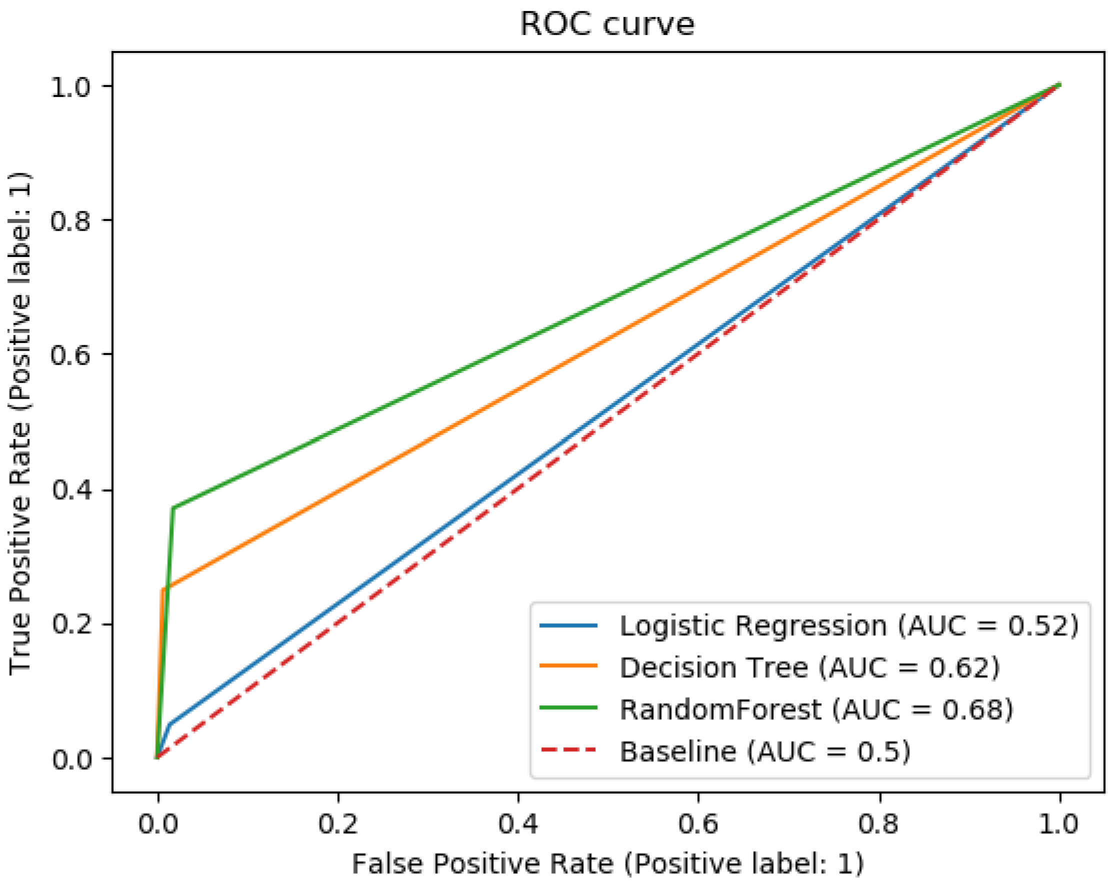




Accuracy → 85.7%

Comparison

Algorithm	Accuracy
Logistic regression	79.94%
Decision Tree	84.1%
Random forest	85.57%



Justification

The Random Forest model showed superior performance compared to other models in all scenarios when using the AUC score as the metric. The ensembling technique used in the Random Forest model might be the reason for its superior performance. Furthermore, the model also demonstrated a close score with hyperparameter optimization. However, to further improve this model, I suggest increasing the number or time of hyperparameter tuning. The current setting of 10 jobs may not have demonstrated optimal improvement, and a more extensive search may yield better results.

Conclusion

In conclusion, the evaluation of the Random Forest model using a confusion matrix and AUC score indicates that the model has shown decent performance in correctly classifying both positive and negative samples. The results suggest that the model can be a valuable tool for predicting churned customers