Bank Customer Churn Analysis and Prediction Using AI Models

Cluster Innovation Centre, University Of Delhi,

Delhi, India, 110007

Submitted by -

Dinesh Yadav

Roll no - 152219

Email - prateekyadav525d@gmail.com

6th Semester, B.Tech (IT & MI)

29th April, 2025

*For the subject **DSE- 18 - Artificial Intelligence***

Abstract:

Customer churn is a critical issue faced by businesses, particularly in the highly competitive banking sector. This project aims to develop a robust predictive model to identify customers who are likely to leave a bank, enabling timely intervention and improved customer retention strategies. The dataset used, *Churn_Modelling.csv*, contains customer demographic information, account details, and transactional features.

To build effective churn prediction models, the data underwent rigorous preprocessing, including feature encoding and normalization. Multiple machine learning algorithms were implemented, including Random Forest, Support Vector Machine (SVM), Logistic Regression, and an Artificial Neural Network (ANN). Hyperparameter tuning using GridSearchCV was performed to optimize model performance.

The results were evaluated using classification metrics such as precision, recall, F1-score, and accuracy. Among the models tested, the Random Forest classifier demonstrated strong performance, while the ANN provided competitive results with deep learning-based generalization. Comparative analysis and visualization techniques helped highlight feature importance and churn trends across different customer segments, including age, geography, and financial variables.

This project showcases the potential of machine learning in enhancing customer relationship management by accurately forecasting churn and informing strategic decision-making.

Introduction

Problem Definition

Customer churn refers to the phenomenon where customers stop using a company's product or service. In highly competitive markets, retaining existing customers is often more cost-effective than acquiring new ones. For industries like banking, telecom, and subscription services, predicting customer churn is crucial, as it enables companies to proactively address issues, offer incentives, or improve services to retain their clientele.

This project focuses on building a predictive model to identify customers who are likely to churn from a bank based on various customer attributes such as age, credit score, balance, tenure, activity level, and more. By analyzing historical customer data, the model aims to classify customers into two categories: those **likely to stay and those likely to leave**.

Significance of the Problem

Reducing customer churn directly impacts a company's revenue and operational stability. Predictive models allow organizations to take preventive actions, enhance customer satisfaction, and tailor marketing strategies. Thus, this project holds practical significance for data-driven decision-making and customer relationship management.

Literature Review

Customer churn prediction has been a widely researched area in the domain of customer relationship management (CRM), especially in industries like telecommunications and banking where customer retention is more cost-effective than acquisition. Traditional statistical methods such as logistic regression have been widely used for churn analysis due to their interpretability and simplicity (Verbeke et al., 2012). However, with the availability of large datasets and complex customer behavior, machine learning approaches have proven to offer superior performance.

Decision trees and ensemble models like Random Forest and Gradient Boosting have gained popularity due to their ability to handle high-dimensional data and capture non-linear relationships (Burez & Van den Poel, 2009). Random Forest, in particular, is known for its robustness and ability to prevent overfitting through the use of multiple decision trees and averaging. Support Vector Machines (SVM) have also shown promise in churn prediction tasks, especially when the data is well-separated and properly scaled (Coussement & Van den Poel, 2008).

Recent advances in deep learning have led to the adoption of Artificial Neural Networks (ANNs), which can automatically extract hierarchical feature representations from data (Xie et al., 2009). Despite their higher computational requirements and need for larger datasets, ANNs offer strong performance, particularly when temporal patterns or complex interactions are involved.

Furthermore, techniques like SMOTE and ADASYN have been widely adopted to address class imbalance, a common issue in churn datasets where churners are the minority class (Chawla et al., 2002). Combining these with hyperparameter tuning strategies like GridSearchCV enhances model performance and generalizability.

Overall, literature suggests that while traditional models offer interpretability, machine learning models—especially ensemble and deep learning methods—are better suited for high-accuracy churn prediction.

The dataset used in this project is sourced from a bank's customer database, aimed at predicting customer churn. It contains 10,000 records and 14 attributes covering personal, demographic, and account-level data. The objective is to build classification models that accurately predict whether a customer is likely to leave the bank (churn) based on their credit score, geography, tenure, balance, age, and more. The dataset has been preprocessed by removing non-informative columns (e.g., RowNumber, CustomerId), encoding categorical features (e.g., Geography, Gender), and balancing the classes where required.

Dataset

churn_modelling_Dataset description :

The dataset comes from a banking customer churn prediction problem. It includes data for 10,000 customers of a bank, along with demographic, geographic, and account-related details. The goal is to predict whether a customer will leave the bank (churn) based on these features.

Columns and Their Description:

Column Name	Description
RowNumber	Row index of the record (not useful for modeling)
CustomerId	Unique ID assigned to each customer

Column Name	Description
Surname	Customer’s last name (can be dropped for modeling)
CreditScore	Credit score of the customer (range: ~350–850)
Geography	Customer’s country of residence (France, Spain, Germany)
Gender	Customer's gender: Male or Female
Age	Age of the customer in years
Tenure	Number of years the customer has been with the bank
Balance	Customer's account balance in dollars
NumOfProducts	Number of bank products used by the customer (e.g., savings, credit card)
HasCrCard	Whether the customer has a credit card (1 = Yes, 0 = No)
IsActiveMember	Whether the customer is active (1 = Active, 0 = Inactive)
EstimatedSalary	Estimated annual salary of the customer (in dollars)
Exited	Target variable – whether the customer left the bank (1 = Yes, 0 = No)

Target Variable:

- **Exited:** This is the binary classification target.
    - 1 means the customer left the bank (churned).
    - 0 means the customer stayed.

Purpose of the Dataset:

To build a predictive machine learning model that identifies customers who are likely to churn, based on their demographics and transaction behavior. This can help banks proactively retain customers.

Methodology

1. Data Preprocessing

To ensure the machine learning algorithms perform optimally, the dataset underwent several preprocessing steps:

- Irrelevant Columns Removal: Columns such as RowNumber, CustomerId, and Surname were dropped as they provide no predictive value.

- Label Encoding: Categorical variables such as Gender and Geography were encoded numerically to make them compatible with machine learning models.

- Feature Scaling: Numerical features were standardized using StandardScaler to ensure all features contribute equally to the model, particularly for distance-based algorithms like SVM.

- Train-Test Split: The data was split into training and testing sets to evaluate model performance accurately.

Feature Extraction

All remaining columns after preprocessing were considered relevant features. The target variable was Exited, indicating whether the customer had churned. No dimensionality reduction or feature selection techniques were applied, as the dataset already contains a manageable number of features.

Algorithm Selection

The project uses a combination of traditional and deep learning models for churn prediction:

- Random Forest (RF) – A robust ensemble learning method using multiple decision trees.

- Support Vector Machine (SVM) – A classification algorithm that finds the optimal hyperplane to separate classes.

- Logistic Regression (LR) – A statistical model used for binary classification.

- Artificial Neural Network (ANN) – Implemented using TensorFlow/Keras for deeper learning-based prediction.

Models -

1. **ANN :**

Model summary: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	832
dense_1 (Dense)	(None, 32)	2080
dense_2 (Dense)	(None, 1)	33

Total params: 2945 (11.50 KB)
Trainable params: 2945 (11.50 KB)
Non-trainable params: 0 (0.00 Byte)

2. SVM
3. Random Forest
4. Logistic regression

Tools and Libraries Used

- Python for overall implementation.
- Scikit-learn for preprocessing, classical models, and evaluation.
- TensorFlow/Keras for building and training the ANN.
- Matplotlib/Seaborn for visualization.
- Pandas for data manipulation.
- Pickle for saving trained models.

Hyperparameter Tuning

Hyperparameter tuning was performed using GridSearchCV, a method provided by scikit-learn to exhaustively search over specified parameter values for an estimator. This step helps in optimizing model performance by finding the best combination of hyperparameters.

Techniques Used

- GridSearchCV: Conducted cross-validation over a predefined hyperparameter grid.

- 5-Fold Cross Validation (cv=5): Used to evaluate each hyperparameter setting more reliably.

- Scoring Metric: Accuracy was used as the performance metric for selecting the best parameters.

Models Tuned and Parameters:

Random forest model parameters searches:

```
param_grid_rf = {
   'n_estimators': [100, 200, 300],
   'max_depth': [None, 10, 20, 30],
   'min_samples_split': [2, 5, 10],
   'min_samples_leaf': [1, 2, 4],
   'bootstrap': [True, False]
}
```

n_estimators: Number of trees in the forest.

max_depth: Maximum depth of each tree.

SVM model parameters searches:

```
param_grid_svm = {
   'C': [0.1, 1, 10, 100],
   'kernel': ['linear', 'rbf', 'poly'],
   'gamma': ['scale', 'auto'],
   'degree': [2, 3, 4]  # Only used for 'poly' kernel
}
```

C: Regularization parameter controlling trade-off between smooth decision boundary and classification error.
kernel: Type of kernel to use in the algorithm.

Logistic regression model parameters searches:

```
param_grid_lr = {
   'C': [0.01, 0.1, 1, 10, 100],
   'penalty': ['l1', 'l2'],
   'solver': ['liblinear', 'saga'],  # Both support L1 and L2
   'max_iter': [100, 500, 1000]
}
```

C: Inverse of regularization strength.

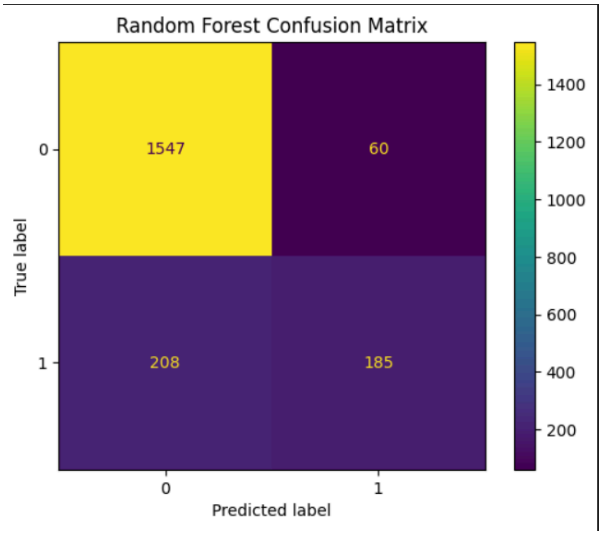
solver: Algorithm to use for optimization.

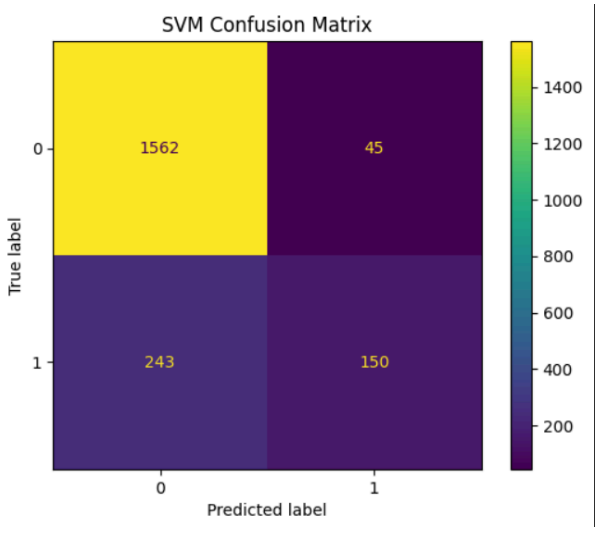
Results :

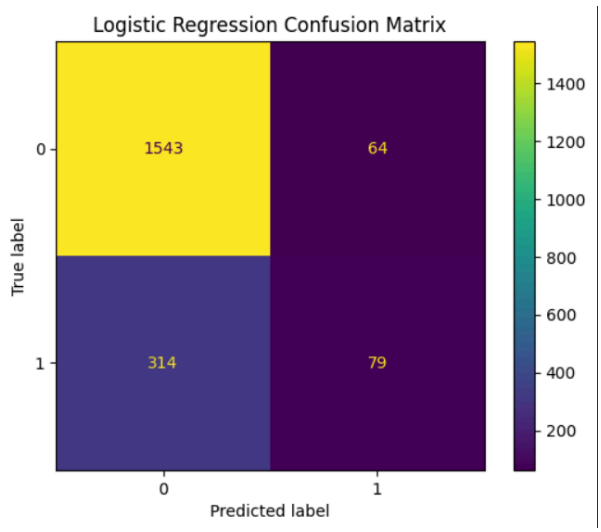
Comparative Analysis Table

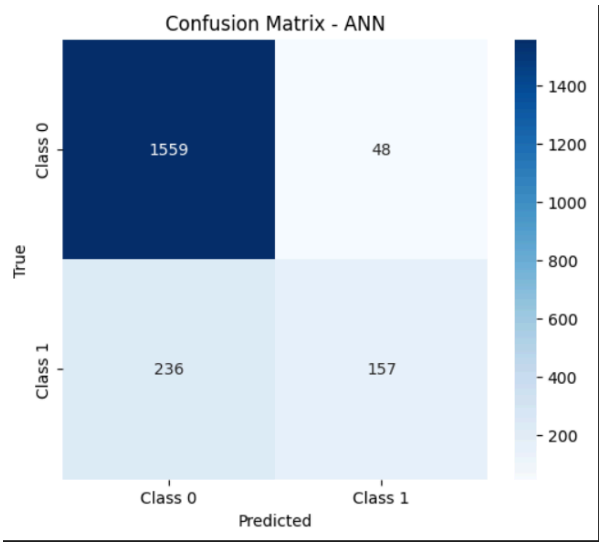
Model	Accuracy	Precision (Class 0)	Precision (Class 1)	Recall (Class 1)	F1-Score (Class 1)
Random Forest	0.866	0.88	0.76	0.47	0.58
Support Vector Machine	0.856	0.87	0.77	0.38	0.51
Logistic Regression	0.811	0.83	0.55	0.20	0.29
Artificial Neural Network	0.860	0.87	0.77	0.40	0.53

Confusion Matrices :



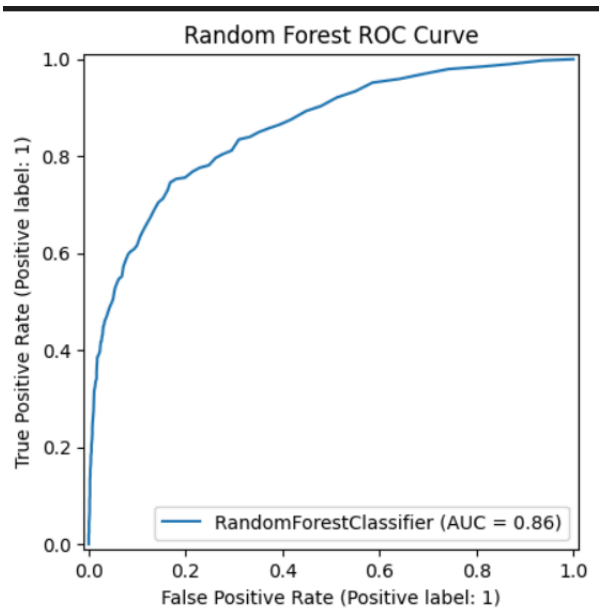


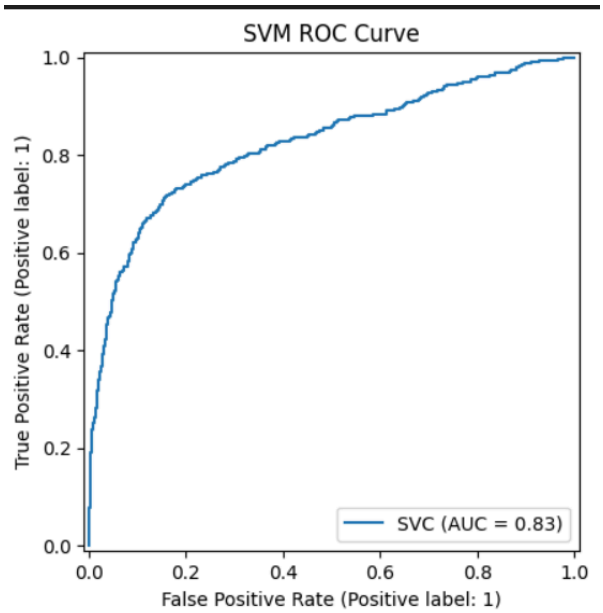


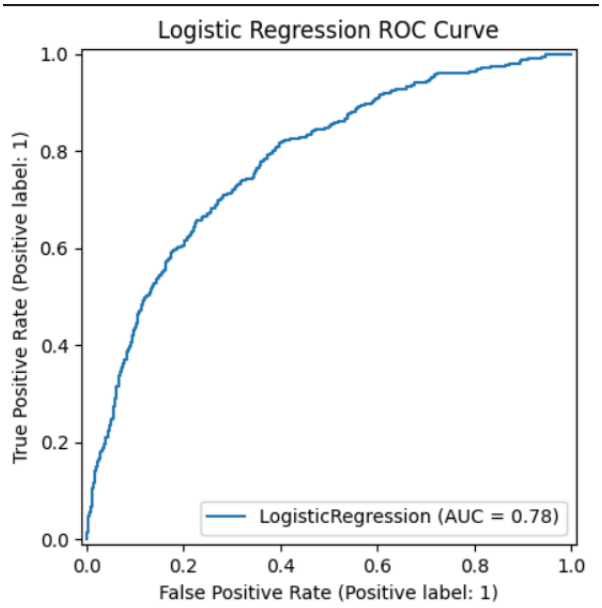


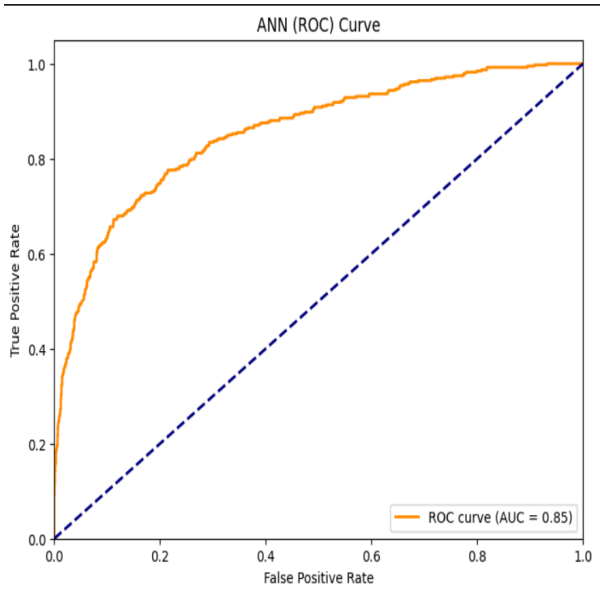
ROC curves of the models:

The ROC curve (Receiver Operating Characteristic curve) is a tool used to evaluate the performance of binary classifiers. It plots the True Positive Rate (TPR) against the False Positive Rate (FPR) across various classification thresholds. The curve illustrates how a model's sensitivity and specificity change as the decision threshold varies. The Area Under the Curve (AUC) provides a single-value summary, with higher AUC values indicating better model performance. The ROC curve is particularly useful for comparing models and assessing performance in imbalanced datasets, as it is threshold-independent and highlights trade-offs between true positives and false positives.









Tensorboard visualisation of ANN classification:

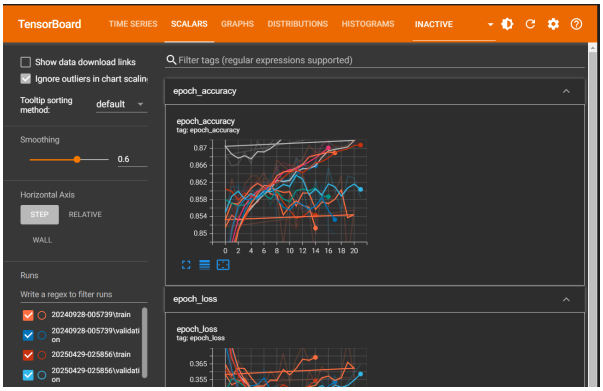
TensorBoard was used in this project to visualize and monitor the training process of the Artificial Neural Network (ANN) model. It provided real-time insights into metrics such as training and validation loss, accuracy, and learning curves. This helped in identifying overfitting or underfitting early and made model performance tracking more transparent and efficient during training.


TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Horizontal axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
Filter tags (regular expressions supported)
dense_1
dense_1/bias_0/histogram

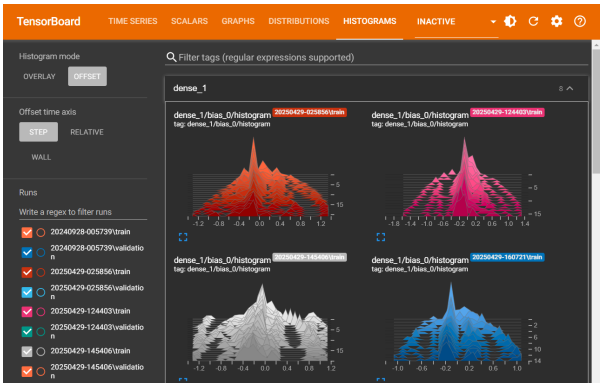

TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Horizontal axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
dense_1/kernel_0/histogram
dense_2

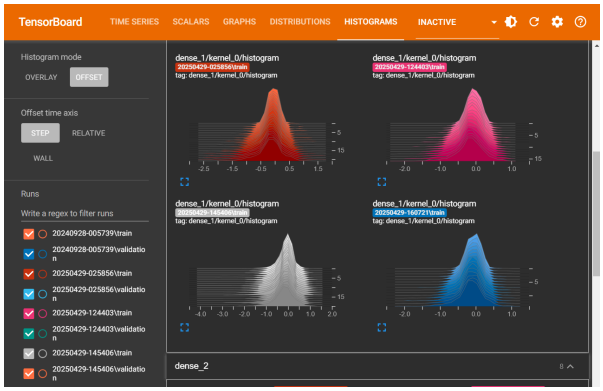

TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Horizontal axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
dense_2/kernel_0/histogram
dense_3

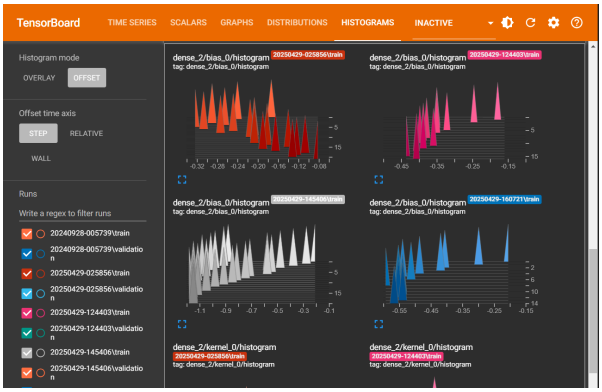

TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Horizontal axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
dense_3
dense_3/bias_0/histogram
dense_3/kernel_0/histogram


TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Show data download links
Ignore outliers in chart scalin
Tooltip sorting method:
default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
Filter tags (regular expressions supported)
epoch_accuracy
epoch_loss


TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Show data download links
Ignore outliers in chart scalin
Tooltip sorting method:
default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
epoch_loss


TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Histogram mode
OVERLAY
OFFSET
Offset time axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
Filter tags (regular expressions supported)
dense_1
dense_1/bias_0/histogram

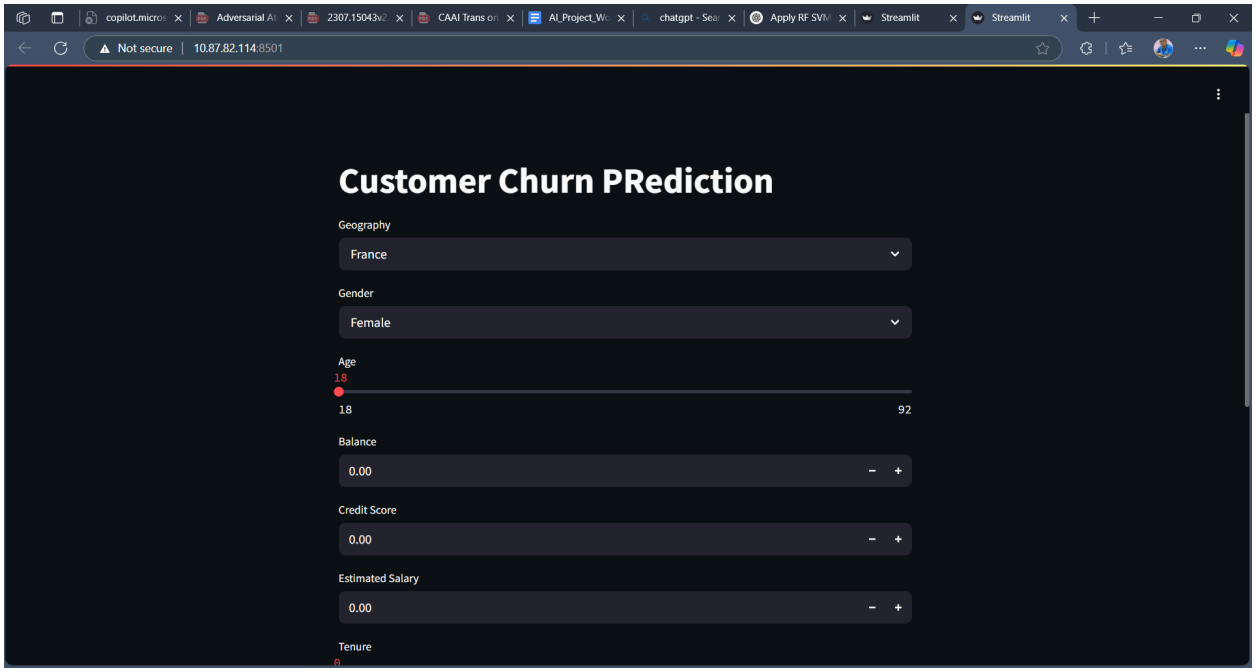

TensorBoard
TIME SERIES
SCALARS
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
INACTIVE
Histogram mode
OVERLAY
OFFSET
Offset time axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
dense_1/kernel_0/histogram
dense_2

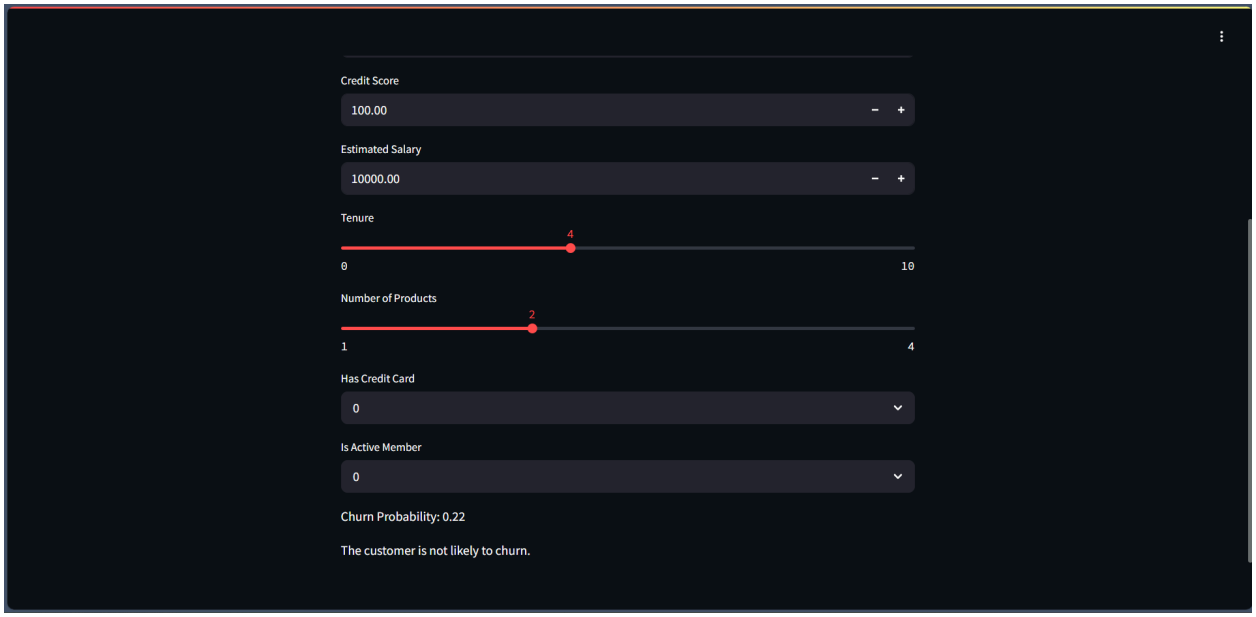


Web Application Interface using Streamlit

To make the churn prediction model more accessible and interactive, a user-friendly web application was developed using Streamlit. This app allows users to input customer details such as age, gender, geography, credit score, balance, estimated salary, and more, through a graphical interface. The backend loads a pre-trained Artificial Neural Network model along with associated preprocessing tools like label encoders, one-hot encoders, and a scalar to process the inputs in real time. Once the user submits their inputs, the model predicts the probability of customer churn. Based on this probability, the app provides an immediate, intuitive result indicating whether the customer is likely to churn or not. This deployment bridges the gap between machine learning models and real-world usability, enabling businesses or stakeholders to leverage predictive analytics in an efficient and practical way without needing any technical expertise.

Streamlit app webpage view:





Results and Discussion

The experimental results indicate that Random Forest consistently outperforms other models in terms of overall accuracy (86.6%) and F1-score for the minority class (churners), achieving a respectable 0.58. The model demonstrates strong generalization and effectively captures non-linear relationships, likely benefiting from its ensemble nature and handling of feature interactions.

The Artificial Neural Network (ANN) delivers competitive results with an accuracy of 86.0%, and an F1-score of 0.53 for the churn class. While slightly behind Random Forest in performance, ANN shows promising capability in capturing complex patterns in the data, especially when enough training data and regularization are applied.

Support Vector Machine (SVM) achieves similar overall accuracy (85.6%) but with a lower recall (0.38) and F1-score (0.51) for churners. This suggests that SVM struggles with correctly identifying minority class instances, possibly due to class imbalance and its sensitivity to parameter tuning.

Logistic Regression, a more interpretable baseline model, yielded the lowest performance across all metrics. Its recall for churners is particularly poor (0.20), with a corresponding F1-score of just 0.29, indicating that it misses most customers likely to churn.

In summary, Random Forest emerges as the best performer for churn prediction, balancing precision and recall. However, ANN also provides competitive results, making it a viable option where deeper learning is desired. The lower performance of Logistic Regression highlights the limitations of linear models in this context, while SVM could benefit from further parameter tuning and data balancing techniques.

Conclusion

In this project, we developed and evaluated multiple machine learning models—including Random Forest, Support Vector Machine, Logistic Regression, and Artificial Neural Network—to predict customer churn using a real-world banking dataset. Among the models tested, Random Forest provided the best overall performance with a balanced trade-off between accuracy, precision, and recall, especially for the minority churn class. The ANN also performed competitively, showcasing its ability to learn non-linear patterns effectively.

Through data preprocessing, feature selection, and hyperparameter tuning, we successfully built a pipeline that can help in identifying customers likely to leave the bank. This type of model can be valuable for businesses seeking to retain customers and reduce revenue loss by taking proactive steps, such as offering personalized services or loyalty benefits.

While the project was relatively simple in scope, it provided a strong foundation in supervised learning workflows and practical experience with classification tasks on imbalanced datasets.

Future Work

This project was simple and focused mainly on using basic customer data to predict churn. While it worked well, there are still a few realistic ways to make it better in the future:

1. Adding More Data: In real life, people leave a service for many reasons. If we had more details like how often they use banking services, their recent transactions, or feedback from surveys, the model could make better predictions.

2. Making It Work in Real Time: Right now, the model runs on past data. In the future, we could connect it to a live system so it gives real-time churn alerts. This way, banks can take quick action to keep valuable customers.

3. Focusing on Business Impact: Not all prediction errors are equal—missing a customer who leaves is more costly than wrongly predicting someone will leave. So, we can look into methods that consider the cost of mistakes and help make smarter business decisions.

References

- Burez, J., & Van den Poel, D. (2009). *Handling class imbalance in customer churn prediction*. Expert Systems with Applications, 36(3), 4626–4636.

- Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). *SMOTE: Synthetic Minority Over-sampling Technique*. Journal of Artificial Intelligence Research, 16, 321–357.

- Coussement, K., & Van den Poel, D. (2008). *Churn prediction in subscription services: An application of support vector machines while comparing two*

parameter-selection techniques. Expert Systems with Applications, 34(1), 313–327.

- Verbeke, W., Martens, D., & Baesens, B. (2012). *Social network analysis for customer churn prediction*. Applied Soft Computing, 14(2), 431–446.

- Xie, Y., Li, Y., & Ngai, E. W. T. (2009). *Customer churn prediction using improved balanced random forests*. Expert Systems with Applications, 36(3), 5445–5449.
- https://github.com/Dinesh525-web/AI_project_sem_6