OBJECTIVE

The problem statement is divided into two parts:

1. Attrited Customer / Churned Customer	**2. Existing customers**
To find Patterns to predicts why they left the services of banks and to predict which customers are at risk of churning.	By identifying patterns and trends in our data, we can create personalized offers and services that customers, ultimately elevating their satisfaction and potentially attract new customers.
The data-driven approach enables the bank to better understand the factors influencing customer departures and take proactive steps to mitigate such occurrences are tailored to the unique needs of our existing.	To improving customer interactions and offering more relevant services by analyzing credit card usage and financial attributes.

→ **Attrited Customer / Churned Customer**

Definition of Attrited Customer / Churned Customer: These terms refer to customers who have terminated their relationship with the bank and are no longer availing its services.

Proposed Solution: To address the issue of customer attrition, we aim to identify patterns within the dataset of churned customers. By doing so, we can predict which customers are at risk of churning. The dataset contains several key attributes, including:

- geographical location
- Gender
- Age
- Tenure
- account balance
- credit score
- membership status
- estimated salary.

Significance of Identifying Patterns: Analyzing these patterns can provide valuable insights into the reasons for customer attrition. Furthermore, it allows us to develop strategies to reduce attrition rates and enhance customer satisfaction. This data-driven approach enables the bank to better understand the factors influencing customer departures and take proactive steps to mitigate such occurrences.

→ **Data-Driven Solutions to Enhance the Banking Experience for Existing Customers :** Our strategy involves a thorough analysis of customer data with the goal of enhancing the banking experience for our existing customers. We will delve into various attributes within our dataset, including:

- Age
- Gender
- Credit Limit
- Average Utilization Ratio

- Income Category
- Credit Card Category
- Marital Status
- Education Level
- Months on Book/Tenure
- Active Status/Months of Inactivity

Let's discuss each solution in more detail:

1. **Customer Segmentation based on Age, Gender, and Credit Limit:**

Proposed Solution: Analyzing customer behaviour in relation to their age, gender, and credit limit. The primary objective here is to identify patterns in how these factors affect the utilization of banking products. We aim to determine which products or services customers are more likely to use based on these attributes.

Significance of Identifying Patterns: Understanding the relationship between age, gender, and credit limit and how it impacts customer behaviour to helps us create a more personalised banking experience.This, in turn, allows us to offer targeted promotions and coupons for the products that align with their preferences.

2. **Marital Status and Utilization Change:**

Proposed Solution: The second part focuses on studying how marital status impacts customer utilization of banking products. This analysis will delve into how customers' average utilization changes during significant life events, such as marriage or divorce, and how these changes influence their product choices.

Significance of Identifying Patterns: We can provide more tailored services based on customers' marital status, thereby increasing their satisfaction. By adapting our offerings to their specific needs.

3. **Customer Credit Card Analysis:**

Proposed Solution: Our initial focus will be on examining the types of credit cards held by customers. This analysis encompasses factors such as card category (e.g., gold, platinum, premium), the benefits associated with these cards, and how customers use them.

Significance of Identifying Patterns: Understanding how customers use different credit cards is vital. This analysis helps us identify the features they value the most and how effectively they leverage their card benefits.

4. **Income, Credit Limit, and Average Utilization Analysis:**

Proposed Solution: Exploring the relationship between a customer's income, credit limit, and their average utilization of credit. By understanding this, we can offer more tailored financial products and services that align with individual financial situations.

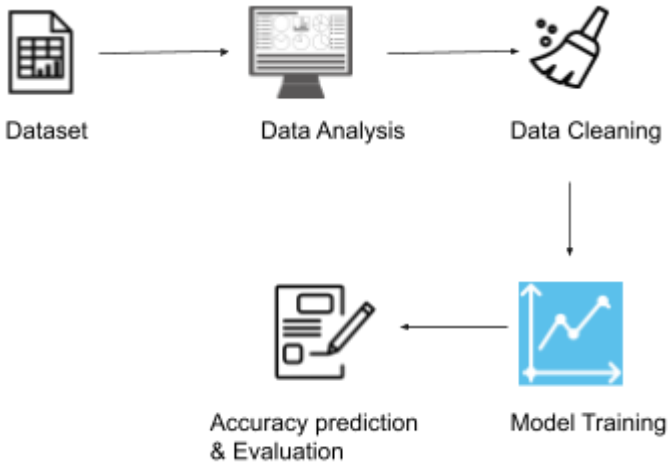
Significance of Identifying Patterns: This analysis allows us to tailor financial products to match each customer's unique financial situation. By identifying how income, credit limit, and credit utilization interplay, we can ensure the services offered are better aligned with their needs.

5. **Tenure and Activity Status Monitoring:**

Proposed Solution: Lastly, we will closely monitor customers' tenure with the bank and their activity status, assessing how frequently they utilize our banking services.

Significance of Identifying Patterns: Tracking customer tenure and activity status is crucial for recognizing and rewarding loyal and active customers. By personalizing our services based on this information, we enhance customer satisfaction and loyalty, ultimately improving their banking experience.

Time-line



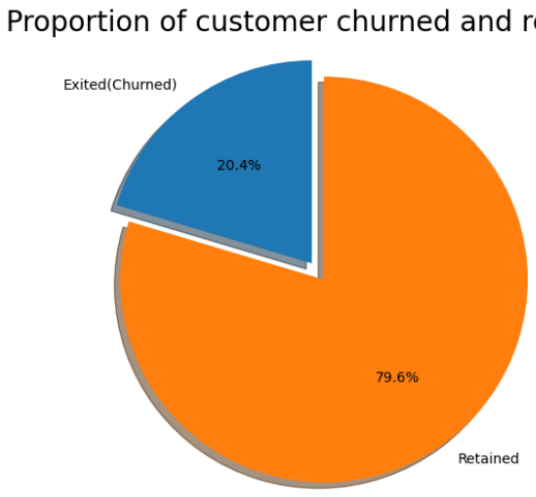
Data Analysis

Important Results

{ Note here that Attrition Flag has a value Attrited Customer, meaning this particular customer has already closed his/her, account and is not associated with the bank as of now.}
Now for predicting the card classes, we can remove the people/customers who have attrited or keep it as it is, it's your choice.

```
Existing Customer    8500
Attrited Customer    1627
Name: Attrition_Flag, dtype: int64
```

{ These pie chart beautifully show us the percentage of people churned/left }
#Thus we can focus our analysis of the 20.4% who left to find why they left
#Also predict if any other customer from the rest 79.6% are showing the same pattern



→ Result for Retained Customer Dataset

Found that the CREDIT LIMIT of Females are significantly less than Males

	Credit_Limit	
	mean	count
Gender		
F	5023.854274	5358
M	12685.674963	4769

#As expected the avg utilisation of females are more than males but there credit limit is low as well.

	Avg_Utilization_Ratio	
	mean	count
Gender		
F	0.341957	5358
M	0.199548	4769

{ Average Utilisation Ratio: it's how much you currently owe divided by your credit limit. It is generally expressed as a percent. }
{ For example, if you have a total of $10,000 in credit available on two credit cards, and a balance of $5,000 on one, your credit utilisation rate is 50% — you're using half of the total credit you have available.
A low credit utilisation rate shows you're using less of your available credit. }

{ Now I have grouped the card category column to find how many cards there are.
and used customer age to find usually which age customers are opting for which card
using max to find the maximum age of the holder
}

```
Card_Category
Blue        73
Gold        63
Platinum    56
Silver      65
Name: Customer_Age, dtype: int64
```

As the min age for platinum cards is high thus the majority will not own it.

```
Card_Category
Blue        26
Gold        29
Platinum    39
Silver      26
Name: Customer_Age, dtype: int64
```

#As the data depicts the mean value for the avg utilisation for blue card is
#very high thus the majority of people will be owning the blue card.

```
Card_Category
Blue        0.290859
Gold        0.057103
Platinum    0.043650
Silver      0.057310
Name: Avg_Utilization_Ratio, dtype: float64
```

{ These figure depicts the count of card type users based on there marital status. }
#As these is one of the most important part of our analysis
#The shift of people from cards as they move from single to married and the shift of people from cards from married to divorce can be calculated using these
#The majority of single people tend to shift towards blue card as they get married .

```
Marital_Status  Card_Category
Divorced        Blue              696
                Silver             46
                Gold                5
                Platinum            1
Married         Blue             4433
                Silver            206
                Gold               41
                Platinum            7
Single          Blue             3624
                Silver            251
                Gold               58
                Platinum           10
Unknown         Blue              683
                Silver             52
                Gold               12
                Platinum            2
Name: Card_Category, dtype: int64
```

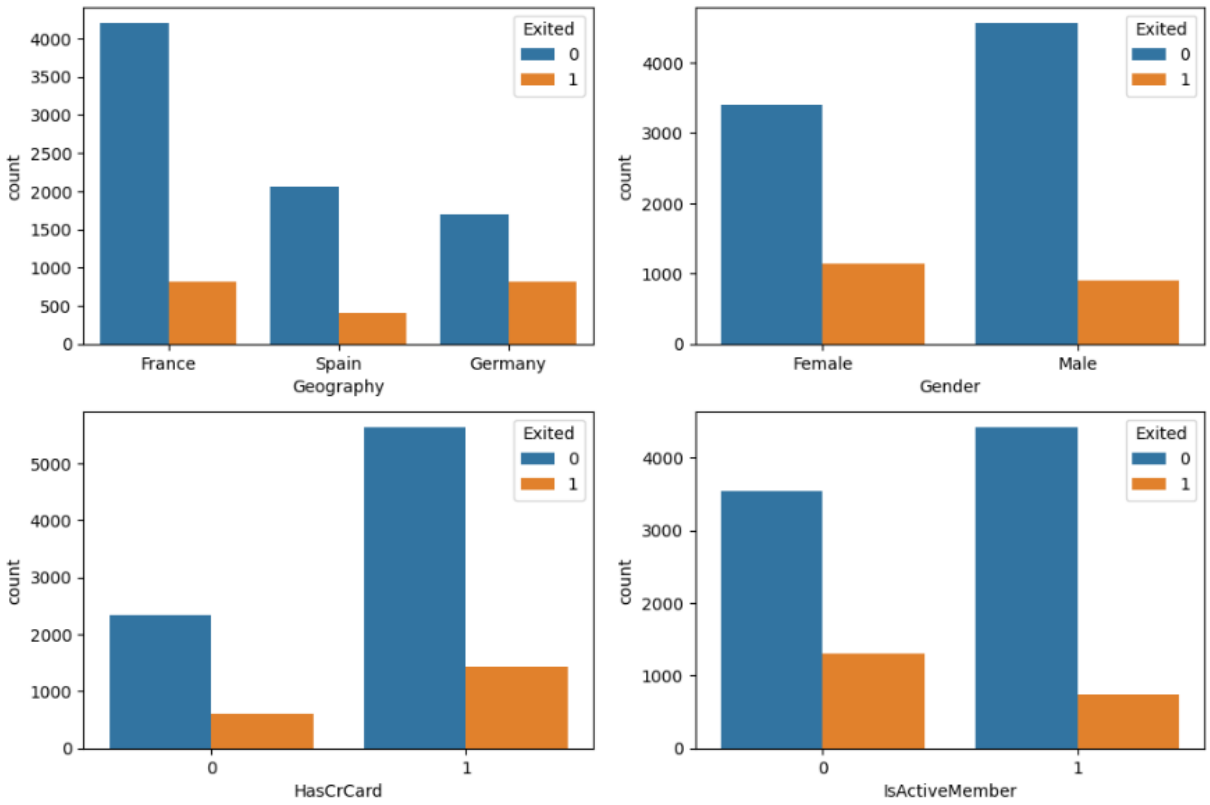
→ Result for Exited Customer Dataset

1 Geography graph
It shows us that france has max population the data around 4000 and
churn value around 600 thus has min churn ratio
While germany has the highest churn ratio with min population around 1500
and churn value around 1000

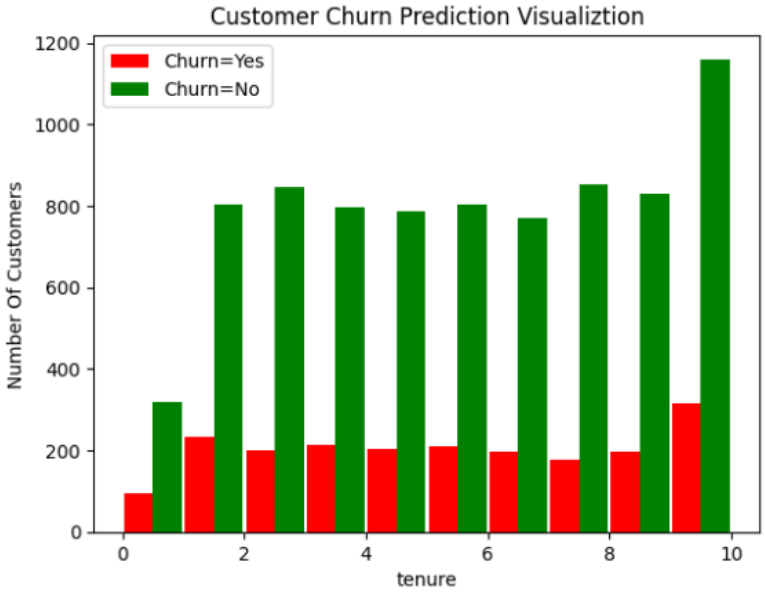
2 Gender
As found the churn/left ration in female is more than male
Population of females are also less than males with around 3500
and that of males near 5000

3 Has Credit Card
#As found the Churn Value of People with Credit Card is more
#Thus it might suggest the customers are not Happy with Bank Services

4 Is Active member
#Nothing unexpected here, the Churn ratio is more in Inactive Customers.

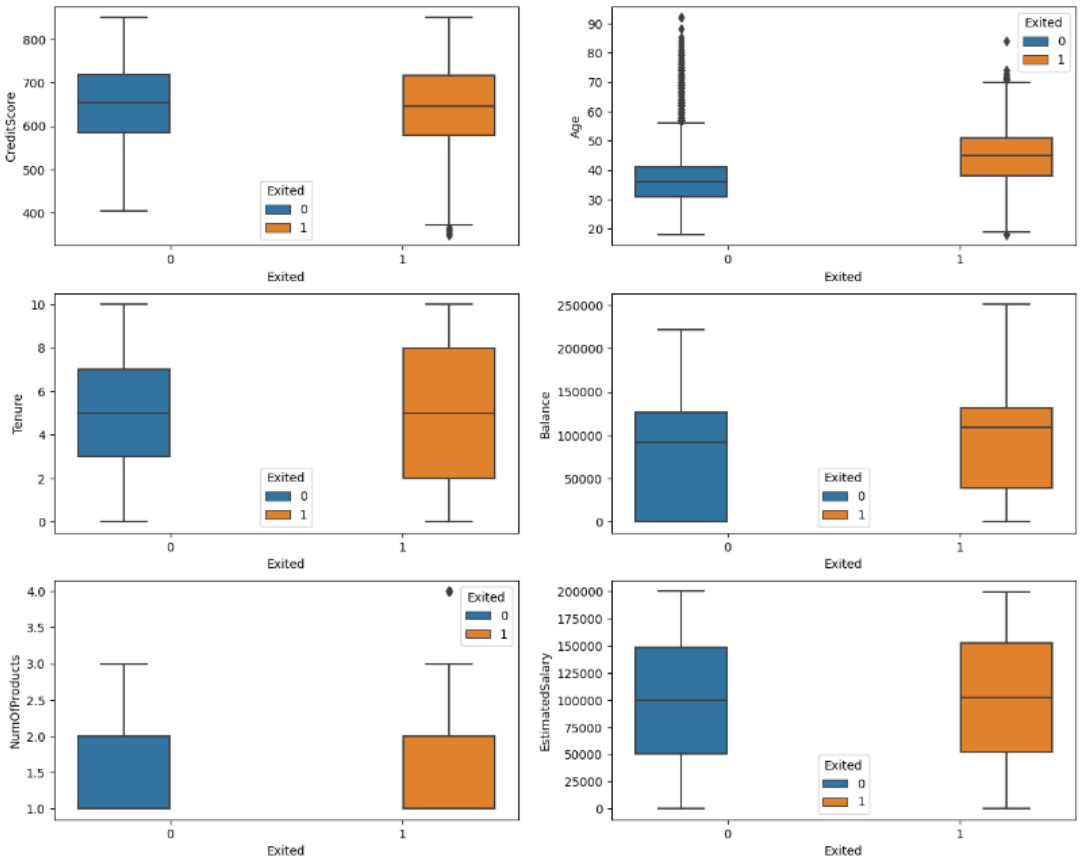


{ It depicts the ratio of customer with tenure and Churn }
The data show us that the ratio of new customer leaving is high which is common
but the Churn ratio of old customers is also high



{ Relations based on the continuous data attributes
y='CreditScore'x = 'Exited'
y='Age',x = 'Exited',
y='Tenure',x = 'Exited',

y='Balance',x = 'Exited',
y='Num of Products',x = 'Exited',
(y='EstimatedSalary',x = 'Exited',
}
#It can be said that only AGE and Tenure column has significant difference and odd data to start our analysis
As people with more tenure or has been with bank more is leaving rather than new one



CLEANING THE DATA

→ For Retained Customer Dataset

Converting Categorical data into numerical data.

{
Gender : Female 0
Male 1
}

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category
0	768805383	0	45	1	3	High School	Married	$60K - $80K	Blue
1	818770008	0	49	0	5	Graduate	Single	Less than $40K	Blue
2	713982108	0	51	1	3	Graduate	Married	$80K - $120K	Blue

3 rows × 21 columns

Extracting data from the Income_Category column

```
Less than $40K    3561
$40K - $60K       1790
$80K - $120K      1535
$60K - $80K       1402
Unknown           1112
$120K +            727
Name: Income_Category, dtype: int64
```

#Label Encoding

{

X['Income_Category'] = label_encoded(X['Income_Category'])

X['Education_Level'] = label_encoded(X['Education_Level'])

X['Marital_Status'] = label_encoded(X['Marital_Status'])

}

```
Income_Category ['$120K +' '$40K - $60K' '$60K - $80K' '$80K - $120K' 'Less than $40K'
 'Unknown']
Education_Level ['College' 'Doctorate' 'Graduate' 'High School' 'Post-Graduate'
 'Uneducated' 'Unknown']
Marital_Status ['Divorced' 'Married' 'Single' 'Unknown']
```

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category
0	768805383	0	45	1	3	3	1	2	Blue
1	818770008	0	49	0	5	2	2	4	Blue
2	713982108	0	51	1	3	2	1	3	Blue

3 rows × 21 columns

```
X.describe()
```

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Months_on_book	Total_Relationship_Count	Months_Inactive_12_mon
count	1.012700e+04	10127.0	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000
mean	7.391776e+08	0.0	46.325960	0.470919	2.346203	3.096574	1.463415	2.863928	35.928409	3.812580	2.341167
std	3.690378e+07	0.0	8.016814	0.499178	1.298908	1.834812	0.737808	1.504700	7.986416	1.554408	1.010622
min	7.080821e+08	0.0	26.000000	0.000000	0.000000	0.000000	0.000000	0.000000	13.000000	1.000000	0.000000
25%	7.130368e+08	0.0	41.000000	0.000000	1.000000	2.000000	1.000000	2.000000	31.000000	3.000000	2.000000
50%	7.179264e+08	0.0	46.000000	0.000000	2.000000	3.000000	1.000000	3.000000	36.000000	4.000000	2.000000
75%	7.731435e+08	0.0	52.000000	1.000000	3.000000	5.000000	2.000000	4.000000	40.000000	5.000000	3.000000
max	8.283431e+08	0.0	73.000000	1.000000	5.000000	6.000000	3.000000	5.000000	56.000000	6.000000	6.000000

→ Result for Exited Customer Dataset

Initial Dataset

```
df.head(7)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0
5	6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	1	0	149756.71	1
6	7	15592531	Bartlett	822	France	Male	50	7	0.00	2	1	1	10062.80	0

```
#AS few of the coloums are not necessary and might affect our analysis and model training
df.drop(['CustomerId','RowNumber','Surname'],axis='columns',inplace=True)
```

```
df.head(7)
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	619	France	Female	42	2	0.00	1	1	1	101348.88	1
1	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0
2	502	France	Female	42	8	159660.80	3	1	0	113931.57	1
3	699	France	Female	39	1	0.00	2	0	0	93826.63	0
4	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0
5	645	Spain	Male	44	8	113755.78	2	1	0	149756.71	1
6	822	France	Male	50	7	0.00	2	1	1	10062.80	0

Printing the categorical variables

```
Geography: ['France' 'Spain' 'Germany']
Gender: ['Female' 'Male']
```

Label Encoding
As Categorical Data cannot be passed on to our model , it must be converted to Numerical Data
{
Gender : Female 0
Male 1
}

One Hot Encoding method
{
it will separate the columns with 3 geographical location present
}

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	BalanceSalaryRatio	TenureByAge	Geography_France	Geography_Germany	Geography_Spain
0	619	0	42	2	0.00	1	1	1	101348.88	1	0.000000	0.047619	1	0	0
1	608	0	41	1	83807.86	1	0	1	112542.58	0	0.744677	0.024390	0	0	1
2	502	0	42	8	159660.80	3	1	0	113931.57	1	1.401375	0.190476	1	0	0
3	699	0	39	1	0.00	2	0	0	93826.63	0	0.000000	0.025641	1	0	0
4	850	0	43	2	125510.82	1	1	1	79084.10	0	1.587055	0.046512	0	0	1
5	645	1	44	8	113755.78	2	1	0	149756.71	1	0.759604	0.181818	0	0	1
6	822	1	50	7	0.00	2	1	1	10062.80	0	0.000000	0.140000	1	0	0

#The problem with the neural network model is that we cant feed it the data with huge variations
such as credit score with 460 unique value , balance range from 0 to more than 1 lakhs
#thus we have to scale them
{
scale_var = ['Tenure','CreditScore','Age','Balance','NumOfProducts','EstimatedSalary']
}

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	BalanceSalaryRatio	TenureByAge	Geography_France	Geography_Germany	Geography_Spain
0	0.538	0	0.324324	0.2	0.000000	0.000000	1	1	0.506735	1	0.000000	0.047619	1	0	0
1	0.516	0	0.310811	0.1	0.334031	0.000000	0	1	0.562709	0	0.744677	0.024390	0	0	1
2	0.304	0	0.324324	0.8	0.636357	0.666667	1	0	0.569654	1	1.401375	0.190476	1	0	0
3	0.698	0	0.283784	0.1	0.000000	0.333333	0	0	0.469120	0	0.000000	0.025641	1	0	0
4	1.000	0	0.337838	0.2	0.500246	0.000000	1	1	0.395400	0	1.587055	0.046512	0	0	1
5	0.590	1	0.351351	0.8	0.453394	0.333333	1	0	0.748797	1	0.759604	0.181818	0	0	1
6	0.944	1	0.432432	0.7	0.000000	0.333333	1	1	0.050261	0	0.000000	0.140000	1	0	0

MODEL TRAINING AND ACCURACY

→ Result for Retained Customer Dataset

```
from sklearn.decomposition import PCA
pca = PCA(n_components = 7)
pca2 = PCA(n_components = 10)
pca_fit = pca.fit_transform(X)
pca_fit2 = pca2.fit_transform(X)
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.ensemble import RandomForestClassifier
```

```
# You can use both of the reduced dataset to see how much accuracy or predictive power of the model is affected.

Xtrain, Xtest, ytrain, ytest = train_test_split(pca_fit2, y, test_size=0.2, random_state = 42)
```

```
random_model = RandomForestClassifier(n_estimators=300, n_jobs = -1)
```

```
#Fit
random_model.fit(Xtrain, ytrain)

y_pred = random_model.predict(Xtest)

#Checking the accuracy
random_model_accuracy = round(random_model.score(Xtrain, ytrain)*100,2)
print(round(random_model_accuracy, 2), '%')
```

```
100.0 %
```

```
random_model_accuracy1 = round(random_model.score(Xtest, ytest)*100,2)
print(round(random_model_accuracy1, 2), '%')
```

```
95.61 %
```

Accuracy score is: **95.61 %**

→ Result for Exited Customer Dataset

The Sequential model :

A Sequential model is appropriate for a plain stack of layers where each layer has exactly one input tensor and one output tensor.

A Sequential model is not appropriate when:

{
Your model has multiple inputs or multiple outputs
Any of your layers has multiple inputs or multiple outputs
You need to do layer sharing
You want non-linear topology (e.g. a residual connection, a multi-branch model)
}

```
model.fit(X_train, y_train, epochs=100)
```

```
Epoch 1/100
250/250 [==============================] - 2s 3ms/step - loss: 0.6241 - accuracy: 0.7347
Epoch 2/100
250/250 [==============================] - 0s 2ms/step - loss: 0.4884 - accuracy: 0.7960
Epoch 3/100
250/250 [==============================] - 1s 2ms/step - loss: 0.4964 - accuracy: 0.7921
Epoch 4/100
250/250 [==============================] - 0s 2ms/step - loss: 0.4562 - accuracy: 0.7959
Epoch 5/100
250/250 [==============================] - 1s 2ms/step - loss: 0.4461 - accuracy: 0.7968
Epoch 6/100
250/250 [==============================] - 0s 2ms/step - loss: 0.4380 - accuracy: 0.8000
Epoch 7/100
250/250 [==============================] - 0s 2ms/step - loss: 0.4304 - accuracy: 0.8066


Epoch 90/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3438 - accuracy: 0.8566
Epoch 91/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3435 - accuracy: 0.8551
Epoch 92/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3433 - accuracy: 0.8546
Epoch 93/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3427 - accuracy: 0.8550
Epoch 94/100
250/250 [==============================] - 1s 2ms/step - loss: 0.3448 - accuracy: 0.8553
Epoch 95/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3435 - accuracy: 0.8559
Epoch 96/100
250/250 [==============================] - 1s 2ms/step - loss: 0.3413 - accuracy: 0.8561
Epoch 97/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3460 - accuracy: 0.8545
Epoch 98/100
250/250 [==============================] - 0s 2ms/step - loss: 0.3417 - accuracy: 0.8565
Epoch 99/100
250/250 [==============================] - 1s 3ms/step - loss: 0.3434 - accuracy: 0.8576
Epoch 100/100
250/250 [==============================] - 1s 3ms/step - loss: 0.3435 - accuracy: 0.8571
<keras.src.callbacks.History at 0x7c656f019f30>
```

Converting our predictions to 0,1 to check accuracy

```
model.evaluate(X_test, y_test)
```

```
63/63 [==============================] - 0s 2ms/step - loss: 0.3549 - accuracy: 0.8505
[0.354936808347702, 0.8504999876022339]
```

```
yp = model.predict(X_test)
yp
```

```
63/63 [==============================] - 0s 4ms/step
array([[0.02784799],
       [0.0834761 ],
       [0.05151472],
       ...,
       [0.01324045],
       [0.01115167],
       [0.06177692]], dtype=float32)
```

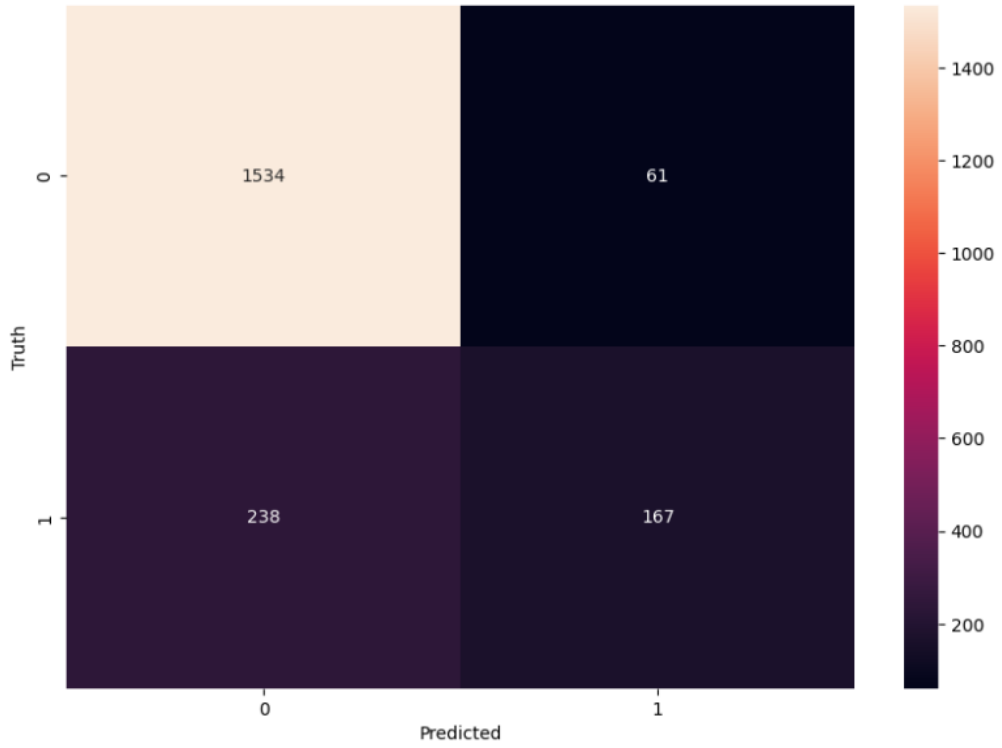
Checking the accuracy

	precision	recall	f1-score	support
0	0.87	0.96	0.91	1595
1	0.73	0.41	0.53	405
accuracy			0.85	2000
macro avg	0.80	0.69	0.72	2000
weighted avg	0.84	0.85	0.83	2000

---

xlabel('Predicted')
ylabel('Truth')

```
Text(95.72222222222221, 0.5, 'Truth')
```



Accuracy score is: **85.05 %**
