

THIRUVALLUVAR UNIVERSITY
PERIYAR ARTS COLLEGE CUDDALORE
– 607001.



DEPARTMENT OF COMPUTER APPLICATIONS

MACHINE LEARNING WITH PYTHON

Project Title : Intelligent Customer Retention: Using Machine Learning for Enhanced Prediction of Telecom Customer Churn

Team Id : NM2023TMID35483

Team Leader : VENNILA R(tv10520CA47)

Team member : NITHYASRI N(tv10520CA33)

Team member : NIVETHA K(tv10520CA34)

Team member : VANITHA E(tv10520CA46)

TOPICS

- **INTRODUCTION**

- Overview

A brief description about your project

- Purpose

The use of this project. What can be achieved using this.

- **Problem Definition & Design Thinking 2.1 Empathy Map** Paste the empathy map screenshot

2.2 Ideation & Brainstorming Map

Paste the Ideation & brainstorming map screenshot

- **RESULT**

Final findings (Output) of the project along with screenshots.

- **ADVANTAGES & DISADVANTAGES**

List of advantages and disadvantages of the proposed solution

- **APPLICATIONS**

The areas where this solution can be applied

- **CONCLUSION**

Conclusion summarizing the entire work and findings.

- **FUTURE SCOPE**

Enhancements that can be made in the future.

- **APPENDIX**

A. Source Code

Attach the code for the solution built.

1 INTRODUCTION

1.1 Overview

Customer churn is often referred to as customer attrition, or customer defection which is the rate at which the customers are lost. Customer churn is a major problem and one of the most important concerns for large companies. Due to the direct effect on the revenues of the companies, especially in the telecom field, companies are seeking to develop means to predict potential customer to churn. Looking at churn, different reasons trigger customers to terminate their contracts, for example better price offers, more interesting packages, bad service experiences or change of customers' personal situations.

Customer churn has become highly important for companies because of increasing competition among companies, increased importance of marketing strategies and conscious behaviour of customers in the recent years. Customers can easily trend toward alternative services. Companies must develop various strategies to prevent these possible trends, depending on the services they provide. During the estimation of possible churns, data from the previous churns might be used. An efficient churn predictive model benefits companies in many ways. Early identification of customers likely to leave may help to build cost effective ways in marketing strategies. Customer retention campaigns might be limited to selected customers but it should cover most of the customer. Incorrect predictions could result in a company losing profits because of the discounts offered to continuous subscribers.

Telecommunication industry always suffers from a very high churn rates when one industry offers a better plan than the previous there is a high possibility of the customer churning from the present due to a better plan in such a scenario it is very difficult to avoid losses but through prediction we can keep it to a minimal level.

Telecom companies often use customer churn as a key business metrics to predict the number of customers that will leave a telecom service provider. A machine learning model can be used to identity the probable churn customers and then makes the necessary business decisions.

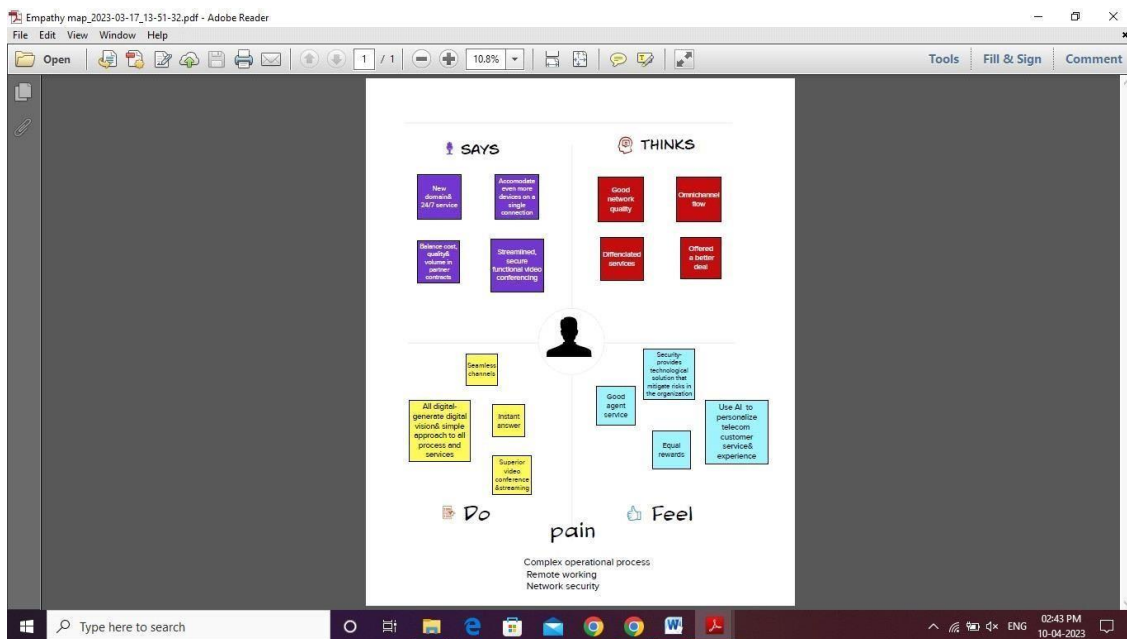
1.2 Purpose

As mentioned above, due to the digitalization and the access to information in large proportions, companies are under constant threats of customer churning, in particular subscription-based service providers that rely on such incomes. Unlike the business-to-customer domain, business to business characteristics, such as higher transactional value of each customer, constitutes a greater negative impact on the revenue stream when customers churn. Hence, it is important to study churn prediction and to support companies in this problem. The purpose of our study is to contribute knowledge in the field of churn prediction in a B2B subscription based service context. In this study we employ different machine learning techniques, in order to investigate the effects on the performance of churn prediction models in the B2B subscription based service context. In particular, the aim is to propose and evaluate different approaches and techniques of how machine learning can be used to predict churn; hence creating value for businesses offering subscription-based services. The objectives of this study are to:

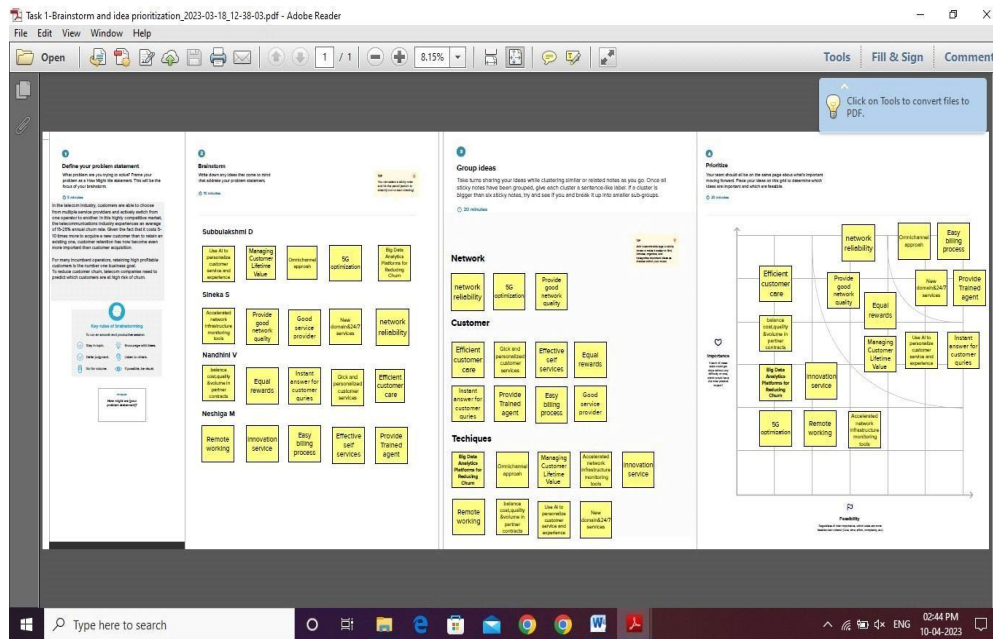
- i) construct a binary churn prediction model, ii) compare different types of algorithms for the built model, and iii) study the effects of balancing the training dataset for the considered model

2 Problem Definition & Design Thinking

2.1 Empathy map



2.2 Ideation & Brainstorming Map



3 RESULT

TELECOM CUSTOMER CHURN PREDICTION

Customer churn has become highly important for companies because of increasing competition among companies, increased importance of marketing strategies and conscious behaviour of customers in the recent years. Customers can easily trend toward alternative services. Companies must develop various strategies to prevent these possible trends, depending on the services they provide. During the estimation of possible churns, data from the previous churns might be used. An efficient churn predictive model benefits companies in many ways. Early identification of customers likely to leave may help to build cost effective ways in marketing strategies. Customer retention campaigns might be limited to selected customers but it should cover most of the customer. Incorrect predictions could result in a company losing profits because of the discounts offered to continuous subscribers.



[Click me to continue with prediction](#)

PREDICTION FORM

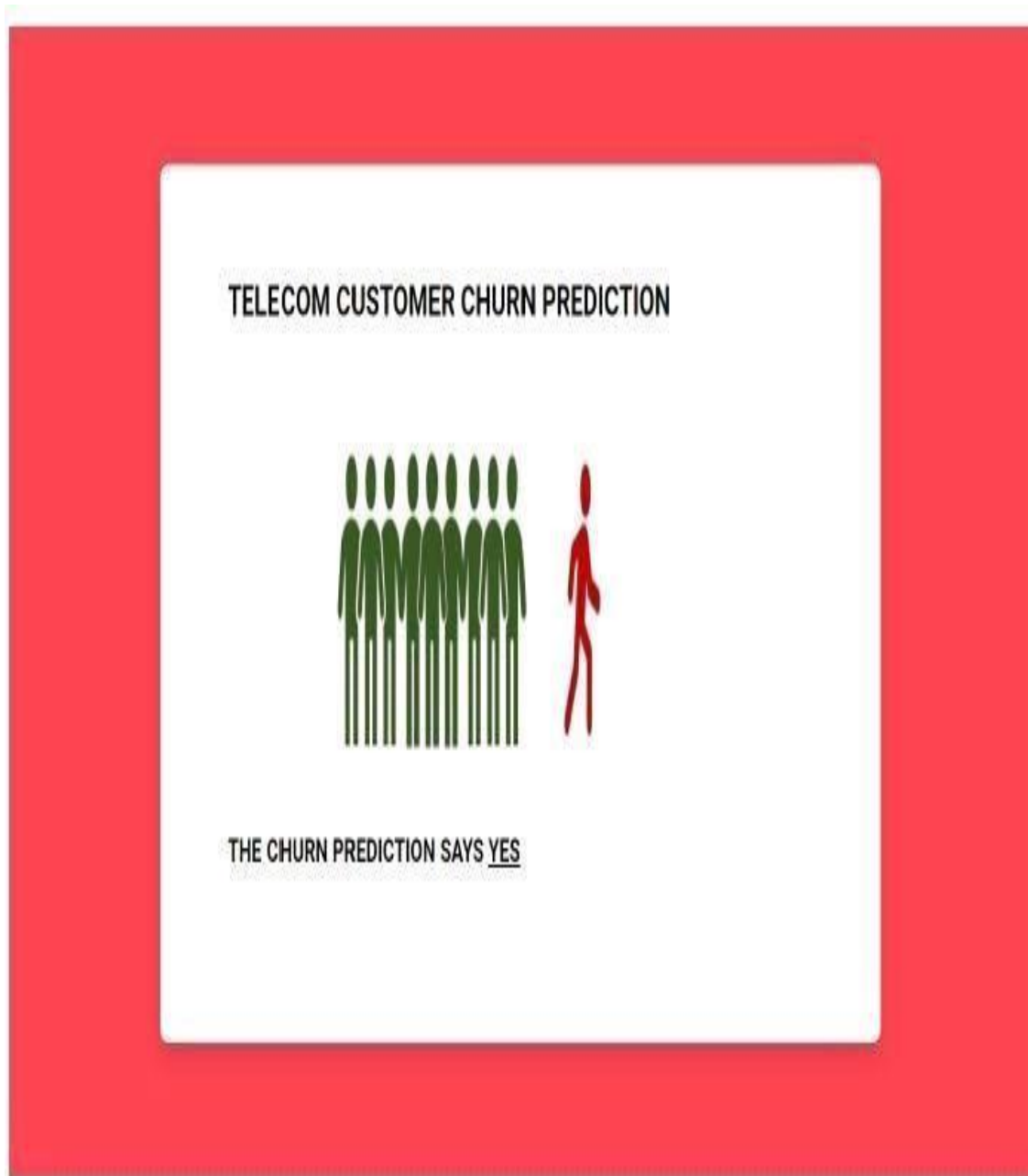
Gender	Yes
Yes	Yes
3	Yes
No Phone service	DSL
No	Yes
No	No
Yes	Yes
Month to Month	Yes
Bank Transfer(Automatic)	39.5
39.5	

Submit

TELECOM CUSTOMER CHURN PREDICTION



THE CHURN PREDICTION SAYS NO



4 ADVANTAGES & DISADVANTAGES

“Communications technology is a crucial part of both today’s and tomorrow’s society.”

In this system, we use various algorithms like Random Forest, XGBoost & Logistic Regression to find accurate values and which helps us to predict the churn of the customer. Here we implement the model by having a dataset that is trained and

tested, which makes us have maximum correct values. In the Initial step, data preprocessing is performed in which we do filtering data and convert data into a similar form, and then we make feature selection.

In the further step prediction and classification is done using the algorithms like Random Forest, XGBoost, Logistic Regression(LR). Training and testing the model with the data set, we observe the behavior of the customer and analyze them. In the final step, we do analysis based on the results obtained and predict the customer churn.

Advantages of Telecom customer churn prediction :

- Quick and accessible communication
- Lack of time period
- Saves time
- Saves gasoline (do not need to drive distance)
- More than two people can communicate with at least one another at an equivalent time
- Next “best thing” to being there
- Easy to exchange ideas and knowledge via phone and/or fax
- Worldwide access
- Easy access to the people you would like to contact.
- Less effort in using transportation just to satisfy a private personally.
- You can just occupy your home and use a telephone or a cellphone if you would like to speak to someone.
- Enable end-users to speak electronically and share hardware, software, and data resources.
- This make corporation to do the transaction at the point only and in a very fast way from many remote locations, exchange business documents electronically with customers and suppliers, or remotely monitor and control production processes.

- Interconnect the pc systems of a business so their computing power is often shared by end-users throughout an enterprise.
- Make the organization work with collaboration and communication among the staff inside and out of doors a corporation.
- Speed
- Develops new products and inventions

Disadvantages of Telecom customer churn prediction:

- Cultural Barrier
- Misunderstanding
- Prank calls
- Sometimes expensive
- High electric bills
- Remote areas don't have access
- Remote areas might not be ready to afford the necessary equipment
- Cannot see whom you're speaking with
- Cannot see facial expressions, therefore results in misunderstandings
- Cultural barriers
- Poor connections or downed power lines during/after storms .

5 APPLICATIONS

Customer churn is a common problem across businesses in many sectors. If you want to grow as a company, **you have to invest in acquiring new clients**. Every time a client leaves, it represents a significant investment lost. Both time and effort need to be channelled into replacing them. Being able to predict when a client is likely to leave, and offer them incentives to stay, can offer huge savings to a business. As a result, understanding what keeps customers engaged is extremely valuable knowledge, as it can help you to develop your retention strategies, and to roll out operational practices aimed at keeping customers from walking out the door.

Predicting churn is a fact of life for any subscription business, and even slight fluctuations in churn can have a significant impact on your bottom line.

6 CONCLUSION

The importance of this type of research in the telecom market is to help companies make more profit. It has become known that predicting churn is one of the most important sources of income to telecom companies. Hence, this research aimed to build a system that predicts the churn of customers in SyriaTel telecom company. These prediction models need to achieve high AUC values. To test and train the model, the sample data is divided into 70% for training and 30% for testing. We chose to perform cross-validation with 10folds for validation and hyperparameter optimization. We have applied feature engineering, effective feature transformation and selection approach to make the features ready for machine learning algorithms. In addition, we encountered another problem: the data was not balanced. Only about 5% of the entries represent customers' churn. This problem was solved by under sampling or using trees algorithms not affected by this problem. Four tree based algorithms were chosen because of their diversity and applicability in this type of prediction. These algorithms are Decision Tree, Random Forest, GBM tree algorithm, and XGBOOST algorithm. The method of preparation and selection of features and entering the mobile social network features had the biggest impact on the success of this model, since the value of AUC in SyriaTel reached 93.301%. XGBOOST tree model achieved the best results in all measurements. The AUC value was 93.301%. The GBM algorithm comes in the second place and the random forest and Decision Tree came third and fourth regarding AUC values. We have evaluated the models by fitting a new dataset related to different periods and without any proactive action from marketing, XGBOOST also gave the best result with 89% AUC. The decrease in result could be due to the non-stationary data model phenomenon, so the model needs training each period of time.

7 FUTURE SCOPE

Some features such as Contract ID, MSISDN and other unique features for all customers were removed. They are not used in the training process because they have a direct correlation with the target output (specific to the customer itself). We deleted features with identical values or missing values, deleted duplicated features, and features that have few numeric values. We found that more than half of the features have more than 98% of missing values. We tried to delete all features that have at least one null value, but this method gave bad results.

Finally, we filled out the missing values with other values derived from either the same features or other features. This method is preferable so that it enables us to use the information in most features for the training process. We applied the following:

- Records that contain more than 90% of missing features were deleted.
- Features that have more than 70% of missing values were deleted.
- For the missing categories in categorical features, they were replaced by a new category called 'Other'.
- The missing numerical values were replaced with the average of the feature.
- The number of categorical features were 78, the first 31 most frequent categories were chosen and the remaining categories were replaced with a new category, so the total number is 32 categories.
- There are some other features with a numeric character but they contain only a limited number of duplicate values in more than one record. This indicates that they are categorical so we have dealt with them as categorical features, but the experiment shows that they perform worse with the model, so that they have been deleted.

- We have also calculated the correlation between numerical features using Pearson and removed the correlated features. This removal had no effect on the final result. Many other methods were tested, but this applied approach gave the best performance of the four algorithms. The number of features after this operation exceeded 2000 features at the end.

8 . APPENDIX

SOURCE CODE

churn_project (3).ipynb

```
jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help
Run Code

2nd task started

In [ ]:
import numpy as np
import pandas as pd
import pickle
import matplotlib.pyplot as plt
import matplotlib inline
import seaborn as sns
import sklearn
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
import imblearn
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

Type Markdown and LaTeX: a²

In [ ]: df = pd.read_csv('Churn_Modelling (2).csv')
df

Out[4]:
```

RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	10134
1	2	15647311	Hill	600	Spain	Female	41	1	83007.86	1	0	1	11254
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	11393
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	9382
4	5	15737088	Mitchell	650	Spain	Female	43	2	125510.82	1	1	1	7900
...
9995	9996	150900229	Obijaku	771	France	Male	39	5	0.00	2	1	0	9627
9996	9997	150900230	Johnstone	510	France	Male	35	10	57369.91	1	1	1	10109
9997	9998	150546332	Liu	709	France	Female	36	7	0.00	1	0	1	4208
9998	9999	15682355	Sabotini	772	Germany	Male	42	3	75075.31	2	1	0	5205
9999	10000	15620319	Vlaeker	792	France	Female	38	4	130142.70	1	1	0	3619

```
jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help
Run Code

memory usage: 2.4+ MB

In [ ]: df.isnull().any()

Out[6]: RowNumber      False
CustomerId    False
Surname        False
CreditScore    False
Geography      False
Gender         False
Age            False
Tenure         False
Balance        False
NumOfProducts  False
HasCrCard      False
IsActiveMember False
EstimatedSalary False
Exited         False
dtype: bool

In [ ]: df.isnull().sum()

Out[7]: RowNumber      0
CustomerId    0
Surname        0
CreditScore    0
Geography      0
Gender         0
Age            0
Tenure         0
Balance        0
NumOfProducts  0
HasCrCard      0
IsActiveMember 0
EstimatedSalary 0
Exited         0
dtype: int64

In [ ]: le=LabelEncoder()
df["RowNumber"] = le.fit_transform(df["RowNumber"])
df["CustomerId"] = le.fit_transform(df["CustomerId"])
df["Surname"] = le.fit_transform(df["Surname"])
df["CreditScore"] = le.fit_transform(df["CreditScore"])
df["Geography"] = le.fit_transform(df["Geography"])
df["Gender"] = le.fit_transform(df["Gender"])
df["Age"] = le.fit_transform(df["Age"])
df["Tenure"] = le.fit_transform(df["Tenure"])
df["Balance"] = le.fit_transform(df["Balance"])
df["NumOfProducts"] = le.fit_transform(df["NumOfProducts"])
df["HasCrCard"] = le.fit_transform(df["HasCrCard"])
df["IsActiveMember"] = le.fit_transform(df["IsActiveMember"])
df["EstimatedSalary"] = le.fit_transform(df["EstimatedSalary"])
df["Exited"] = le.fit_transform(df["Exited"])
```



```
https://naanmudhalvan.smartint... x https://naanmudhalvan.smartint... x Home Page - Select or create a r... x churn_project (3) - Jupyter Note... x +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutter... Other favorites

jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help
Not Trusted Python 3 (ipykernel) O

df["CustomerId"] = le.fit_transform(df["CustomerId"])
df["Surname"] = le.fit_transform(df["Surname"])
df["CreditScore"] = le.fit_transform(df["CreditScore"])
df["Geography"] = le.fit_transform(df["Geography"])
df["Gender"] = le.fit_transform(df["Gender"])
df["Age"] = le.fit_transform(df["Age"])
df["Tenure"] = le.fit_transform(df["Tenure"])
df["Balance"] = le.fit_transform(df["Balance"])
df["NumOfProducts"] = le.fit_transform(df["NumOfProducts"])
df["HasCrCard"] = le.fit_transform(df["HasCrCard"])
df["IsActiveMember"] = le.fit_transform(df["IsActiveMember"])
df["EstimatedSalary"] = le.fit_transform(df["EstimatedSalary"])
df["Exited"] = le.fit_transform(df["Exited"])

In [ ]: df
Out[9]:
RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary
0 0 2736 1115 228 0 0 24 2 0 0 1 1 508
1 1 3258 1177 217 2 0 23 1 743 0 0 1 562
2 2 2184 2040 111 0 0 24 8 5763 2 1 0 571
3 3 5435 289 300 0 0 21 1 0 1 0 0 471
4 4 6899 1822 459 2 0 25 2 3696 0 1 1 393
...
9995 9995 1590 1989 386 0 1 21 5 0 1 1 0 402
9996 9996 161 1336 125 0 1 17 10 124 0 1 1 558
9997 9997 717 1570 310 0 0 18 7 0 0 0 1 298
9998 9998 4656 2345 381 1 1 24 3 427 1 1 0 482
9999 9999 2487 2751 481 0 0 10 4 4112 0 1 0 152
10000 rows x 14 columns

In [ ]: x=df.iloc[:,0:13].values
y=df.iloc[:,13:14].values

In [ ]: x,y
Out[11]: (array([[ 0, 2736, 1115, ..., 1, 5068],
[ 1, 3258, 1177, ..., 0, 5639],
[ 2, 2184, 2040, ..., 1, 5707],
...,
[9997, 717, 1570, ..., 0, 1, 2062],
[9998, 4656, 2345, ..., 1, 0, 4639],
[9999, 2487, 2751, ..., 1, 0, 1878]]),
array([[1],
[1],
[1],
[1],
[0]]))
```

```
https://naanmudhalvan.smartint... x https://naanmudhalvan.smartint... x Home Page - Select or create a r... x churn_project (3) - Jupyter Note... x +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutter... Other favorites

jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help
Not Trusted Python 3 (ipykernel) O

[0],
[1],
...,
[1],
[1],
[0]]))

In [ ]: one-hot-encoder()
a=one.fit_transform(x[:,1:12]).toarray()
b=one.fit_transform(x[:,13]).toarray()
c=one.fit_transform(x[:,14]).toarray()
d=one.fit_transform(x[:,15]).toarray()
e=one.fit_transform(x[:,16]).toarray()
f=one.fit_transform(x[:,17]).toarray()
g=one.fit_transform(x[:,18]).toarray()
h=one.fit_transform(x[:,19]).toarray()
i=one.fit_transform(x[:,20]).toarray()
j=one.fit_transform(x[:,21]).toarray()
k=one.fit_transform(x[:,22]).toarray()
x=np.delete(x[:,1,2,3,4,5,7,8,9,10,11,12],axis=1)
x=np.concatenate((a,b,c,d,e,f,g,h,i,j,k,x),axis=1)

In [ ]: x
Out[13]: array([[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 0.000e+00,
2.400e+01],
[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 1.000e+00,
2.300e+01],
[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 2.000e+00,
2.400e+01],
...,
[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 9.997e+03,
1.000e+01],
[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 9.998e+03,
2.400e+01],
[0.000e+00, 0.000e+00, 0.000e+00, ..., 0.000e+00, 9.999e+03,
1.000e+01]])

In [ ]: sm=SMOTE()
x_resample,y_resample=sm.fit_resample(x,y)

In [ ]: x_resample,y_resample
Out[15]: (array([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 0.00000000e+00, 2.40000000e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 1.00000000e+00, 2.30000000e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 2.00000000e+00, 2.40000000e+01],
...,
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 9.99999999e+03, 1.00000000e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 9.99999999e+03, 1.00000000e+01]]),
array([[1],
[1],
[1],
[1],
[0]]))
```

Browser tabs: <https://naanmudhalvan.smartint...> | <https://naanmudhalvan.smartint...> | Home Page - Select or create a... | chum_project (3) - Jupyter Note...

Address bar: localhost:8888/notebooks/chum_project%20(3).ipynb

Browser extensions: Booking.com, Booking.com, LastPass password..., Express VPN, Gmail, YouTube, Maps, News, Translate, get_storage | Flutter..., Other favorites

JupyterLab interface: chum_project (3) (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted Python 3 (ipykernel)

Code editor: `x_resample, y_resample = set_fit_resample(x, y)`

Input: `In []: x_resample, y_resample`

Output: `Out[15]: (array([[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 0.00000000e+00, 2.40000000e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 1.00000000e+00, 2.30000000e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 2.00000000e+00, 0.00000000e+01],
...,
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 1.4018654e+03, 1.51735161e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 2.18117165e+03, 2.46748645e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 3.55529231e+09, 2.33098942e+01]]),
array([1, 0, 1, ..., 1, 1, 1]))`

Input: `In []: x.shape, x_resample.shape`

Output: `Out[16]: ((10000, 29797), (15926, 29797))`

Input: `In []: y.shape, y_resample.shape`

Output: `Out[17]: ((10000, 1), (15926, 1))`

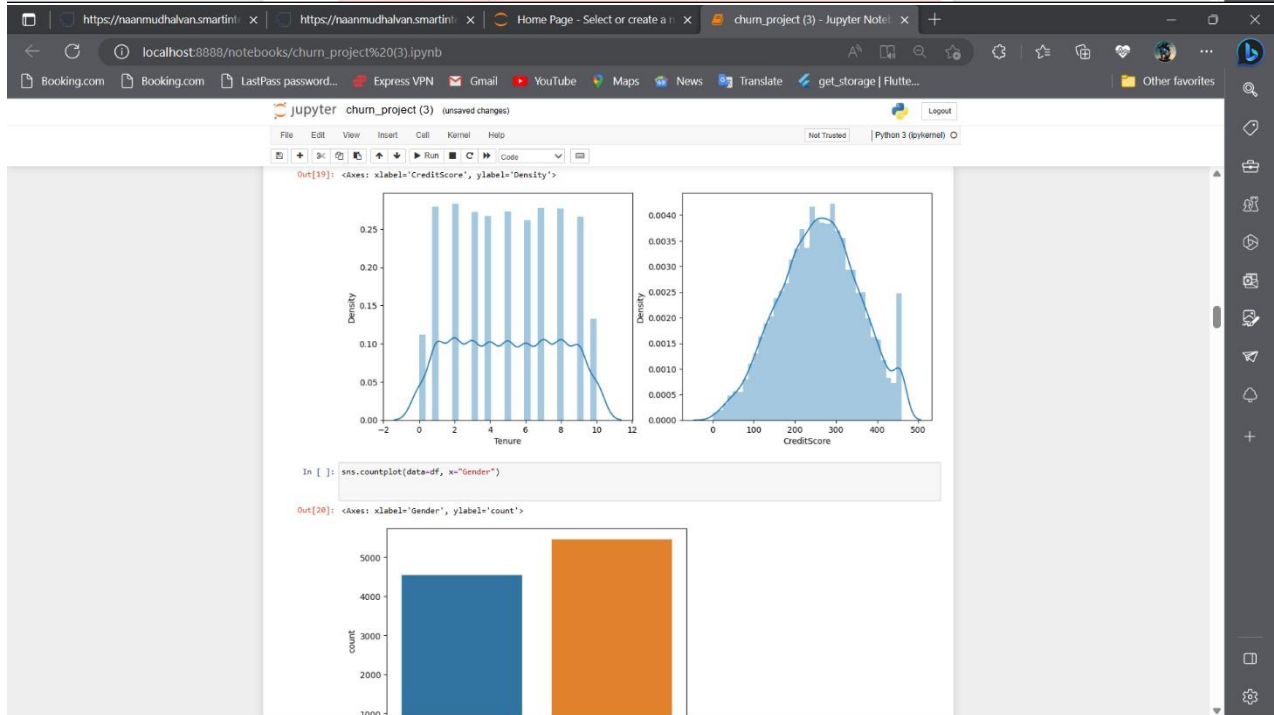
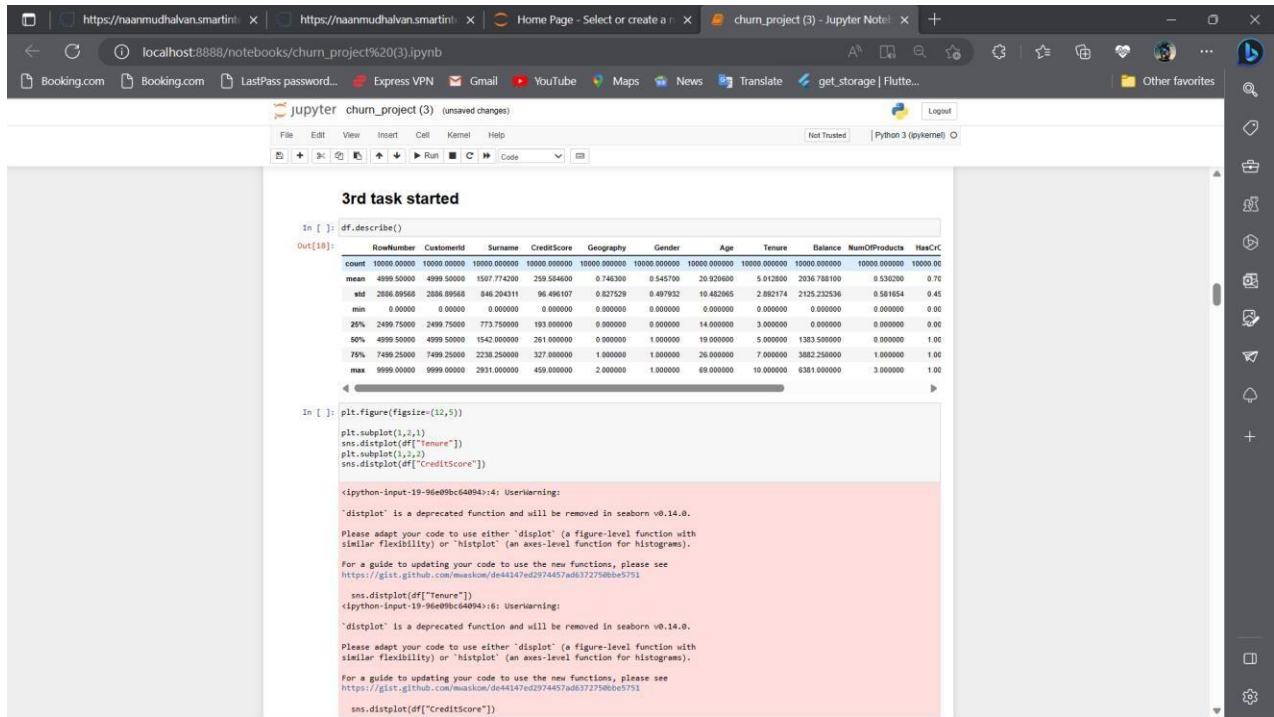
3rd task started

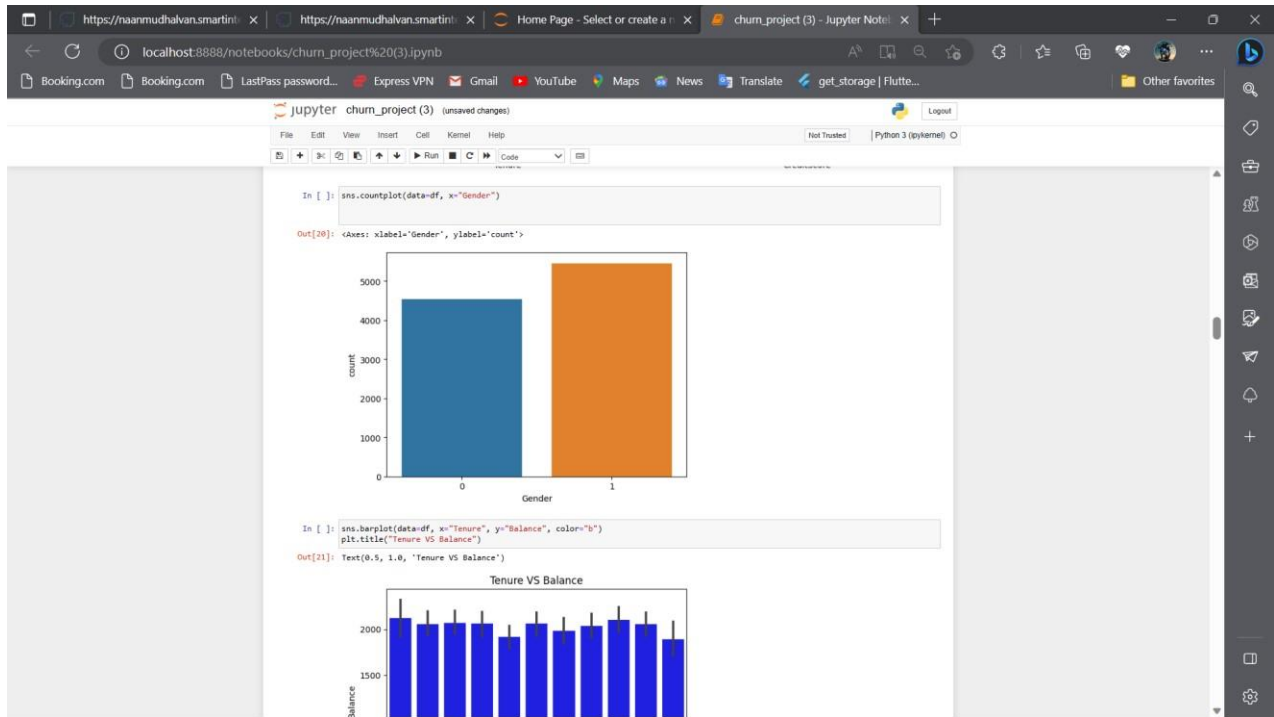
Input: `In []: df.describe()`

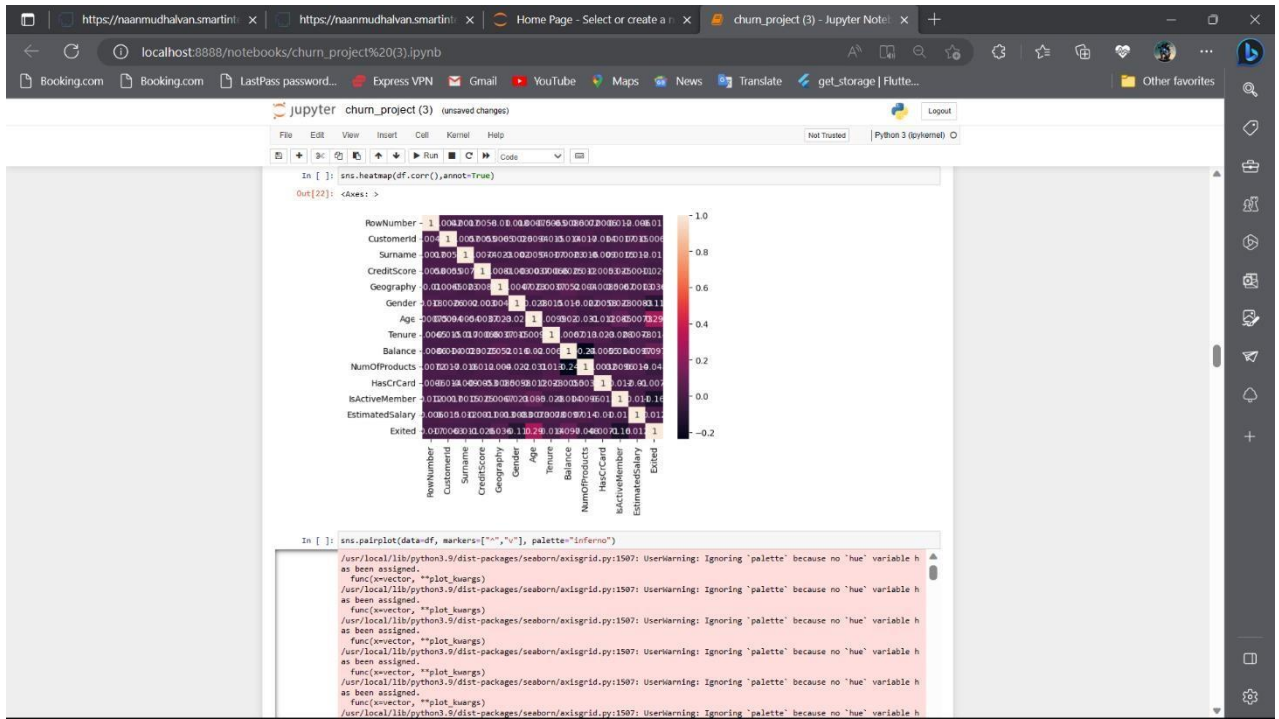
Output: `Out[18]:`

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HascCrC
count	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000
mean	4999.50000	4999.50000	1507.774200	259.504600	0.746300	0.545700	20.920000	5.912000	2036.788100	0.530200	0.70
std	2036.09568	2036.09568	846.204311	96.496107	0.827529	0.497932	10.482065	2.882174	2125.232536	0.581654	0.45
min	0.00000	0.00000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00
25%	2499.75000	2499.75000	773.750000	193.000000	0.000000	0.000000	14.000000	3.000000	0.000000	0.000000	0.00
50%	4999.50000	4999.50000	1542.000000	261.000000	0.000000	1.000000	19.000000	5.000000	1383.500000	0.000000	1.00
75%	7499.25000	7499.25000	2238.250000	327.000000	1.000000	1.000000	26.000000	7.000000	3082.250000	1.000000	1.00
max	9999.00000	9999.00000	2931.000000	459.000000	2.000000	1.000000	69.000000	10.000000	6381.000000	3.000000	1.00

Input: `In []: plt.figure(figsize=(12,5))
plt.subplot(1,2,1)`







https://naanmudhalvan.smartini... churn_project (3) - Jupyter Note...

localhost:8888/notebooks/churn_project%20(3).ipynb

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel) O

```
/usr/local/lib/python3.9/dist-packages/seaborn/axisgrid.py:1507: UserWarning: Ignoring 'palette' because no 'hue' variable h
as been assigned.
  func(xvector, **plot_kwards)

In [ ]: x = df.drop(columns=["Exited"])
        y = df["Balance"]
        x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)

In [ ]: sc=StandardScaler()
        x_train=sc.fit_transform(x_train)
        x_test=sc.fit_transform(x_test)

4th task started

In [ ]: x_train.shape

Out[26]: (8000, 13)

In [ ]: def logreg(x_train, x_test, y_train, y_test):
        lr=LogisticRegression(random_state=0)
        lr.fit(x_train, y_train)
        y_lr_tr = lr.predict(x_train)

In [ ]: #reporting and building the decision tree model
        def logreg(x_train, x_test, y_train, y_test):
            lr=LogisticRegression(random_state=0)
            lr.fit(x_train, y_train)
            y_lr_tr = lr.predict(x_train)
            print(accuracy_score(y_lr_tr, y_train))
            y_pred_lr = lr.predict(x_test)
            print(accuracy_score(y_pred_lr, y_test))
            print("Logistic Regression")
            print(confusion_matrix(y_test, y_pred_lr))
            print("Classification Report")
            print(classification_report(y_test, y_pred_lr))

In [ ]: logreg(x_train, x_test, y_train, y_test)

/usr/local/lib/python3.9/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
```

```
https://naanmudhalvan.smartini: X https://naanmudhalvan.smartini: X Home Page - Select or create a x churn_project (3) - Jupyter Note X +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutte... Other favorites

jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel) O

In [ ]: logreg(x_train, x_test, y_train, y_test)
/usr/local/lib/python3.9/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
0.92075
0.3445
***Logistic Regression***
Confusion Matrix
[[689  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 1  0  0 ...  0  0  0]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]]

In [ ]: #Exporting and building the Decision tree model
def decisionTree(x_train, x_test, y_train, y_test):
    dtc = DecisionTreeClassifier(criterion='entropy', random_state=0)
    dtc.fit(x_train, y_train)
    y_dt_tr = dtc.predict(x_train)
    print(accuracy_score(y_dt_tr, y_train))
    y_pred_dt = dtc.predict(x_test)
    print(accuracy_score(y_pred_dt, y_test))
    print("Decision Tree")
    print("Confusion Matrix")
    print(confusion_matrix(y_test, y_pred_dt))
    print("Classification Report")
    print(classification_report(y_test, y_pred_dt))

In [ ]: decisionTree(x_train, x_test, y_train, y_test)
1.0
0.3445
***Decision Tree***
Confusion Matrix
[[689  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 1  0  0 ...  0  0  0]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]]
```

```
https://naanmudhalvan.smartini: X https://naanmudhalvan.smartini: X Home Page - Select or create a x churn_project (3) - Jupyter Note X +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutte... Other favorites

jupyter churn_project (3) (unsaved changes)
File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel) O

In [ ]: #Exporting and building the Decision tree model
def decisionTree(x_train, x_test, y_train, y_test):
    dtc = DecisionTreeClassifier(criterion='entropy', random_state=0)
    dtc.fit(x_train, y_train)
    y_dt_tr = dtc.predict(x_train)
    print(accuracy_score(y_dt_tr, y_train))
    y_pred_dt = dtc.predict(x_test)
    print(accuracy_score(y_pred_dt, y_test))
    print("Decision Tree")
    print("Confusion Matrix")
    print(confusion_matrix(y_test, y_pred_dt))
    print("Classification Report")
    print(classification_report(y_test, y_pred_dt))

In [ ]: decisionTree(x_train, x_test, y_train, y_test)
1.0
0.3445
***Decision Tree***
Confusion Matrix
[[689  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]
 [ 1  0  0 ...  0  0  0]
 ...
 [ 0  0  0 ...  0  0  0]
 [ 0  0  0 ...  0  0  0]]
Classification Report
precision recall f1-score support
0 0.35 0.35 0.35 689
1 0.00 0.00 0.00 1
2 0.00 0.00 0.00 1
3 0.00 0.00 0.00 0
4 0.00 0.00 0.00 0
5 0.00 0.00 0.00 1

In [ ]: #Exporting and building the random forest model
def RandomForest(x_train, x_test, y_train, y_test):
    rf = RandomForestClassifier(criterion='entropy', n_estimators=10, random_state=0)
    rf.fit(x_train, y_train)
    y_rf_tr = rf.predict(x_train)
    print(accuracy_score(y_rf_tr, y_train))
    y_pred_rf = rf.predict(x_test)
    print(accuracy_score(y_pred_rf, y_test))
    print("Confusion Matrix")
    print(confusion_matrix(y_test, y_pred_rf))
    print("Classification Report")
    print(classification_report(y_test, y_pred_rf))
```


https://naanmudhalvan.smartini... | https://naanmudhalvan.smartini... | Home Page - Select or create a... | churn_project (3) - Jupyter Note... | +

localhost:8888/notebooks/churn_project%20(3).ipynb

Booking.com | Booking.com | LastPass password... | Express VPN | Gmail | YouTube | Maps | News | Translate | get_storage | Flutte... | Other favorites

jupyter churn_project (3) (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3 (ipykernel) | O

```
def RandomForest(x_train,x_test,y_train,y_test):
    rf = RandomForestClassifier(criterion='entropy',n_estimators=10,random_state=0)
    rf.fit(x_train,y_train)
    y_rf_tr = rf.predict(x_train)
    print(accuracy_score(y_rf_tr,y_train))
    yPred_rf = rf.predict(x_test)
    print(accuracy_score(yPred_rf,y_test))
    print("Confusion Matrix")
    print(confusion_matrix(y_test,yPred_rf))
    print("Classification Report")
    print(classification_report(y_test,yPred_rf))

In [ ]: RandomForest(x_train,x_test,y_train,y_test)
0.99875
0.9445
***Random forest***
Confusion Matrix:
[[489 0 0 ... 0 0 0]
 [ 0 0 0 ... 0 0 0]
 [ 1 0 0 ... 0 0 0]
 ...
 [ 0 0 0 ... 0 0 0]
 [ 0 0 0 ... 0 0 0]
 [ 1 0 0 ... 0 0 0]]
Classification Report
precision    recall  f1-score   support

0       0.81      1.00      0.90       689
1       0.00      0.00      0.00         1
2       0.00      0.00      0.00         1
3       0.00      0.00      0.00         1
5       0.00      0.00      0.00         1

In [ ]: #Importing and building the KNN model
def KNN(x_train,x_test,y_train,y_test):
    knn = KNeighborsClassifier()
    knn.fit(x_train,y_train)
    y_knn_tr = knn.predict(x_train)
    print(accuracy_score(y_knn_tr,y_train))
    yPred_knn = knn.predict(x_test)
    print(accuracy_score(yPred_knn,y_test))
    print("Confusion Matrix")
    print(confusion_matrix(y_test,yPred_knn))
    print("Classification Report")
    print(classification_report(y_test,yPred_knn))
```

https://naanmudhalvan.smartini... | https://naanmudhalvan.smartini... | Home Page - Select or create a... | churn_project (3) - Jupyter Note... | +

localhost:8888/notebooks/churn_project%20(3).ipynb

Booking.com | Booking.com | LastPass password... | Express VPN | Gmail | YouTube | Maps | News | Translate | get_storage | Flutte... | Other favorites

jupyter churn_project (3) (unsaved changes)

File Edit View Insert Cell Kernel Help

Not Trusted | Python 3 (ipykernel) | O

```
Knn = KNeighborsClassifier()
Knn.fit(x_train,y_train)
y_knn_tr = Knn.predict(x_train)
print(accuracy_score(y_knn_tr,y_train))
yPred_knn = Knn.predict(x_test)
print(accuracy_score(yPred_knn,y_test))
print("Confusion Matrix")
print(confusion_matrix(y_test,yPred_knn))
print("Classification Report")
print(classification_report(y_test,yPred_knn))

In [ ]: KNN(x_train,x_test,y_train,y_test)
0.998375
0.94
***Knn***
Confusion Matrix:
[[489 0 0 ... 0 0 0]
 [ 1 0 0 ... 0 0 0]
 [ 1 0 0 ... 0 0 0]
 ...
 [ 0 0 0 ... 0 0 0]
 [ 0 0 0 ... 0 0 0]
 [ 0 0 0 ... 0 0 0]]
Classification Report
precision    recall  f1-score   support

0       0.57      0.99      0.73       689
2       0.00      0.00      0.00         1
3       0.00      0.00      0.00         1
5       0.00      0.00      0.00         1
11      0.00      0.00      0.00         0

In [ ]: #Importing and building the random forest model
def svm(x_train,x_test,y_train,y_test):
    svm = SVC(kernel='linear')
    svm.fit(x_train,y_train)
    y_svm_tr = svm.predict(x_train)
    print(accuracy_score(y_svm_tr,y_train))
    yPred_svm = svm.predict(x_test)
    print(accuracy_score(yPred_svm,y_test))
    print("Confusion Matrix")
    print(confusion_matrix(y_test,yPred_svm))
    print("Classification Report")
    print(classification_report(y_test,yPred_svm))

# Call the function with appropriate arguments
```



```

[False]
[False]
[False]

In [ ]:
accuracy = accuracy_score(y_test, ann_pred)
conf_matrix = confusion_matrix(y_test, ann_pred)
class_report = classification_report(y_test, ann_pred)

print("ANN Model")
print("Accuracy Score:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", class_report)

***ANN Model***
Accuracy Score: 0.46
Confusion Matrix:
[[10 19]
 [ 8 13]]
Classification Report:
              precision    recall  f1-score   support

     0       0.56      0.34      0.43       29
     1       0.41      0.62      0.49       21

 accuracy      0.46      0.46      0.46       50
 macro avg     0.48      0.48      0.46       50
 weighted avg   0.49      0.46      0.45       50

In [ ]:

In [ ]:

In [ ]:
# create a StandardScaler object and fit it on the training data
sc = StandardScaler()
sc.fit(x_train)

# transform the training and test data with the scaler object
x_train_scaled = sc.transform(x_train)
x_test_scaled = sc.transform(x_test)

# create and train the logistic regression model
lr = LogisticRegression(random_state=0)
lr.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model
lr_pred = lr.predict(x_test_scaled)
print("Logistic Regression Model")
print("Accuracy Score")
print(accuracy_score(lr_pred, y_test))

```

```

/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

In [ ]:

In [ ]:
#testing on random input values
dt = DecisionTreeClassifier(random_state=0)
dt.fit(x_train, y_train)

# predict on test data and evaluate the model
dt_pred = dt.predict(x_test)
print("Decision Tree Classifier Model")
print("Accuracy Score")
print(accuracy_score(dt_pred, y_test))
print("Confusion Matrix")
print(confusion_matrix(dt_pred, y_test))
print("Classification Report")
print(classification_report(dt_pred, y_test))

# predicting on random input
dt_pred_own = dt.predict([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,456,1,0,2345,456]])
print("Predicted output is:", dt_pred_own)

***Decision Tree Classifier Model***
Accuracy Score
0.58
Confusion Matrix
[[15  7]
 [14 14]]
Classification Report
              precision    recall  f1-score   support

     0       0.52      0.68      0.59       22
     1       0.67      0.59      0.57       28

 accuracy      0.59      0.59      0.58       50
 macro avg     0.59      0.59      0.58       50
 weighted avg   0.60      0.58      0.58       50

Predicted output is: [1]

In [ ]:
x_train_scaled = sc.transform(x_train)
x_test_scaled = sc.transform(x_test)

# create and train the random forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, criterion='gini', random_state=0)
rf_classifier.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model

```

```
https://naanmudhalvan.smartini: X https://naanmudhalvan.smartini: X Home Page - Select or create a x churn_project (3) - Jupyter Note X +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutte... Other favorites

jupyter churn_project (3) (autosaved)
File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel) O

macro avg 0.39 0.39 0.38 50
weighted avg 0.60 0.58 0.58 50

Predicted output is: [1]

In [ ]: x_train_scaled = sc.transform(x_train)
x_test_scaled = sc.transform(x_test)

# create and train the random forest classifier
rf_classifier = RandomForestClassifier(n_estimators=100, criterion='gini', random_state=0)
rf_classifier.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model
rf_pred = rf_classifier.predict(x_test_scaled)
print("Accuracy Score")
print(accuracy_score(rf_pred, y_test))
print("Confusion Matrix")
print(confusion_matrix(rf_pred, y_test))
print("Classification Report")
print(classification_report(rf_pred, y_test))

# predict on a random input
random_input = sc.transform([[0]*40])
print("Predicting on random input")
rf_pred_own = rf_classifier.predict(random_input)
print("Output is:", rf_pred_own)

<ipython-input-43-eeed9587350>:6: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
rf_classifier.fit(x_train_scaled, y_train)

***Random Forest Classifier Model***
Accuracy Score
0.44
Confusion Matrix
[[11 10]
 [18 12]]
Classification Report
precision recall f1-score support
0 0.38 0.52 0.44 21
1 0.52 0.38 0.44 29
accuracy 0.44 50
macro avg 0.45 0.45 0.44 50
weighted avg 0.46 0.44 0.44 50

Predicting on random input
Output is: [0]

In [ ]: x_train_scaled = sc.transform(x_train)
x_test_scaled = sc.transform(x_test)
```

```
https://naanmudhalvan.smartini: X https://naanmudhalvan.smartini: X Home Page - Select or create a x churn_project (3) - Jupyter Note X +
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com Booking.com LastPass password... Express VPN Gmail YouTube Maps News Translate get_storage | Flutte... Other favorites

jupyter churn_project (3) (autosaved)
File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel) O

Predicting on random input
Output is: [0]

In [ ]: x_train_scaled = sc.transform(x_train)
x_test_scaled = sc.transform(x_test)

# create and train the SVM classifier
svm_classifier = SVC(kernel='linear', random_state=0)
svm_classifier.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model
svm_pred = svm_classifier.predict(x_test_scaled)
print("Accuracy Score")
print(accuracy_score(svm_pred, y_test))
print("Confusion Matrix")
print(confusion_matrix(svm_pred, y_test))
print("Classification Report")
print(classification_report(svm_pred, y_test))

# predict on a random input
random_input = sc.transform([[0]*40])
print("Predicting on random input")
svm_pred_own = svm_classifier.predict(random_input)
print("Output is:", svm_pred_own)

***SVM Classifier Model***
Accuracy Score
0.48
Confusion Matrix
[[ 9  6]
 [20 15]]
Classification Report
precision recall f1-score support
0 0.31 0.60 0.41 15
1 0.71 0.43 0.54 35
accuracy 0.48 50
macro avg 0.51 0.51 0.47 50
weighted avg 0.59 0.48 0.50 50

Predicting on random input
Output is: [1]

/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
y = column_or_1d(y, warn=True)

In [ ]: knn_classifier = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
knn_classifier.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model
```

```
weighted avg      0.59      0.48      0.50      50

Predicting on random input
Output is: [1]

/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  y = column_or_1d(y, warn=True)

In [ ]: knn_classifier = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)
knn_classifier.fit(x_train_scaled, y_train)

# predict on test data and evaluate the model
knn_pred = knn_classifier.predict(x_test_scaled)
print("KNN Classifier Model")
print("Accuracy Score")
print(accuracy_score(knn_pred, y_test))
print("Confusion Matrix")
print(confusion_matrix(knn_pred, y_test))
print("Classification Report")
print(classification_report(knn_pred, y_test))

# predict on a random input
random_input = sc.transform([[0]*40])
print("Predicting on random input")
knn_pred_own = knn_classifier.predict(random_input)
print("Output is:", knn_pred_own)

***KNN Classifier Model***
Accuracy Score
0.36
Confusion Matrix
[[ 7 10]
 [22 11]]
Classification Report
              precision    recall  f1-score   support

     0       0.24      0.41      0.30      17
     1       0.52      0.33      0.41      33

 accuracy      0.38      0.37      0.36      50
 macro avg     0.38      0.37      0.36      50
 weighted avg   0.43      0.36      0.37      50

Predicting on random input
Output is: [1]

/usr/local/lib/python3.9/dist-packages/sklearn/neighbors/_classification.py:215: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  return self._fit(X, y)

In [ ]: classifier = Sequential()
classifier.add(Dense(units=30, activation='relu', input_dim=40))
```

```
accuracy      0.38      0.37      0.36      50
macro avg     0.38      0.37      0.36      50
weighted avg   0.43      0.36      0.37      50

Predicting on random input
Output is: [1]

/usr/local/lib/python3.9/dist-packages/sklearn/neighbors/_classification.py:215: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  return self._fit(X, y)

In [ ]: classifier = Sequential()
classifier.add(Dense(units=30, activation='relu', input_dim=40))
classifier.add(Dense(units=30, activation='relu'))
classifier.add(Dense(units=1, activation='sigmoid'))

# compile the model
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# train the model
model_history = classifier.fit(x_train_scaled, y_train, batch_size=10, validation_split=0.33, epochs=200)

# predict on test data and evaluate the model
ann_pred = classifier.predict(x_test_scaled)
ann_pred_own = (ann_pred*5)
print("ANN Classifier Model")
print("Accuracy Score")
print(accuracy_score(ann_pred, y_test))
print("Confusion Matrix")
print(confusion_matrix(ann_pred, y_test))
print("Classification Report")
print(classification_report(ann_pred, y_test))

# predict on a random input
random_input = sc.transform([[0]*40])
print("Predicting on random input")
ann_pred_own = classifier.predict(random_input)
ann_pred_own = (ann_pred_own*5)
print("Output is:", ann_pred_own)

Epoch 1/200
10/10 [=====] - 1s 25ms/step - loss: 0.7334 - accuracy: 0.4700 - val_loss: 0.7402 - val_accuracy: 0.5000
Epoch 2/200
10/10 [=====] - 0s 6ms/step - loss: 0.6862 - accuracy: 0.5600 - val_loss: 0.7344 - val_accuracy: 0.4600
Epoch 3/200
10/10 [=====] - 0s 8ms/step - loss: 0.6567 - accuracy: 0.5800 - val_loss: 0.7313 - val_accuracy: 0.4600
Epoch 4/200
10/10 [=====] - 0s 7ms/step - loss: 0.6317 - accuracy: 0.6300 - val_loss: 0.7287 - val_accuracy: 0.4800
Epoch 5/200
```

```
10/10 [=====] - fit_loss: 0.5862 - accuracy: 0.7500 - val_loss: 0.7282 - val_accuracy: 0.4600
Epoch 6/200
10/10 [=====] - fit_loss: 0.5862 - accuracy: 0.7500 - val_loss: 0.7282 - val_accuracy: 0.4600
Epoch 7/200

In [ ]: # Random Forest Classifier without hyperparameter tuning
rfc = RandomForestClassifier(random_state=42)
rfc.fit(x_train, y_train)
rfc_pred = rfc.predict(x_test)

print("****RFC Model without Hyperparameter Tuning****")
print("Accuracy:", accuracy_score(y_test, rfc_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, rfc_pred))
print("Classification Report:\n", classification_report(y_test, rfc_pred))

# Random Forest Classifier with hyperparameter tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_leaf': [1, 2, 4]
}

rfc_tuned = RandomForestClassifier(random_state=42)
rfc_cv = GridSearchCV(rfc_tuned, param_grid, cv=5)
rfc_cv.fit(x_train, y_train)
rfc_tuned_pred = rfc_cv.predict(x_test)

print("****RFC Model with Hyperparameter Tuning****")
print("Best Parameters:", rfc_cv.best_params_)
print("Accuracy:", accuracy_score(y_test, rfc_tuned_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, rfc_tuned_pred))
print("Classification Report:\n", classification_report(y_test, rfc_tuned_pred))

<ipython-input-47-19029ab18dc>:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please
change the shape of y to (n_samples,) for example using ravel().
rfc.fit(x_train, y_train)

****RFC Model without Hyperparameter Tuning****
Accuracy: 0.42
Confusion Matrix:
[[ 9 20]
 [ 9 12]]
Classification Report:
      precision    recall  f1-score   support

 0       0.50      0.31      0.38        29
 1       0.38      0.57      0.45        21

 accuracy          0.44      0.44      0.42        50
 macro avg          0.44      0.44      0.42        50
 weighted avg          0.45      0.42      0.41        50
```

```
print("****RFC Model without hyperparameter tuning****")
print("Accuracy:", accuracy_score(y_test, rfc_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, rfc_pred))
print("Classification Report:\n", classification_report(y_test, rfc_pred))

# Random Forest Classifier with hyperparameter tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30],
    'min_samples_leaf': [1, 2, 4]
}

rfc_tuned = RandomForestClassifier(random_state=42)
rfc_cv = GridSearchCV(rfc_tuned, param_grid, cv=5)
rfc_cv.fit(x_train, y_train)
rfc_tuned_pred = rfc_cv.predict(x_test)

print("****RFC Model with Hyperparameter Tuning****")
print("Best Parameters:", rfc_cv.best_params_)
print("Accuracy:", accuracy_score(y_test, rfc_tuned_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, rfc_tuned_pred))
print("Classification Report:\n", classification_report(y_test, rfc_tuned_pred))

<ipython-input-47-19029ab18dc>:3: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please
change the shape of y to (n_samples,) for example using ravel().
rfc.fit(x_train, y_train)

****RFC Model without Hyperparameter Tuning****
Accuracy: 0.42
Confusion Matrix:
[[ 9 20]
 [ 9 12]]
Classification Report:
      precision    recall  f1-score   support

 0       0.50      0.31      0.38        29
 1       0.38      0.57      0.45        21

 accuracy          0.44      0.44      0.42        50
 macro avg          0.44      0.44      0.42        50
 weighted avg          0.45      0.42      0.41        50

In [ ]: import pickle
pickle.dump(rfc, open("churn.pkl", "wb"))
```

Index.html

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html X
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > flask > templates > index.html > html > head > link
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Customer Churn Prediction</title>
8   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sha384-9aIt"
9   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-9aIt
10 </head>
11
12 <body>
13
14
15   <!-- Navbar Starts -->
16   <section id="nav-bar">
17     <nav class="navbar navbar-expand-lg navbar-light bg-light">
18       <a class="navbar-brand font-weight-bold" href="#" style="color: yellow;">Churn prediction</a>
19       <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false"
20         <span class="navbar-toggler-icon"></span>
21     </button>
22     <div class="collapse navbar-collapse" id="navbarNav">
23       <ul class="navbar-nav ml-auto">
24         <li class="nav-item active">
25           <a class="nav-link" href="#">Home <span class="sr-only">(current)</span></a>
26         </li>
27         <li class="nav-item">
28           <a class="nav-link" href="#">Customer</a>
29         </li>
30         <li class="nav-item">
31           <a class="nav-link" href="#">Predictions</a>
```

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html X
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > flask > templates > index.html > html > body > form > div.contentform > div.leftcontact
31     <a class="nav-link" href="#">Predictions</a>
32   </li>
33   <li class="nav-item">
34     <a class="nav-link" href="#">Contact</a>
35   </li>
36 </ul>
37 </div>
38 </nav>
39 </section>
40 <!-- Navbar Ends -->
41
42
43
44
45   <form action="{{ url_for('predict')}}" method="post">
46     <h1>Enter the Information Below and find if a Customer will Churn out or not</h1>
47
48   <div class="contentform">
49     <div id="sendMessage"> Your message has been sent successfully. Thank you. </div>
50
51
52     <div class="leftcontact">
53       <div class="form-group">
54         <p>Credit Score<span>*</span></p>
55         <span class="icon-case"></span>
56         <input name="CreditScore" type="number" >
57       </div>
58     </div>
59
60     <div class="form-group">
```



```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > flask > templates > index.html > html > body > form > div.contentform > div.leftcontact > div.form-group > p
60 <div class="form-group">
61 <p>Age <span>*</span></p>
62 <span class="icon-case"></span>
63 <input name="Age" required="required">
64 <div class="validation"></div>
65 </div>
66
67 <div class="form-group">
68 <p>Tenure <span>*</span></p>
69 <span class="icon-case"></span>
70 <input name="Tenure" required="required">
71 <div class="validation"></div>
72 </div>
73
74 <div class="form-group">
75 <p>Enter the Account Balance: <span>*</span></p>
76 <span class="icon-case"></span>
77 <input name="Balance" required="required">
78 <div class="validation"></div>
79 </div>
80
81 <div class="form-group">
82 <p>Number of Products <span>*</span></p>
83 <span class="icon-case"></span>
84 <input name="NumOfProducts" required="required">
85 <div class="validation"></div>
86 </div>
87
88 <div class="form-group">
89 <p>Do the Customer have Credit Card? (1=Yes, 0=No) <span>*</span></p>
90 <span class="icon-case"></span>
```

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > flask > templates > index.html > html > body > form > div.contentform > div.leftcontact > div.form-group > input
90 <span class="icon-case"></span>
91 <input name="HasCrCard" required="required">
92 <div class="validation"></div>
93 </div>
94
95
96
97 </div>
98
99 <div class="rightcontact">
100
101 <div class="form-group">
102 <p>Is the Customer Active Member (1=Yes, 0=No)<span>*</span></p>
103 <span class="icon-case"></span>
104 <input name="IsActiveMember" required="required">
105 <div class="validation"></div>
106 </div>
107
108 <div class="form-group">
109 <p>Enter the Estimated Salary: <span>*</span></p>
110 <span class="icon-case"></span>
111 <input name="EstimatedSalary" required="required">
112 <div class="validation"></div>
113 </div>
114
115 <div class="form-group">
116 <p>Enter The Location: <span>*</span></p>
117 <span class="icon-case"></span>
118 <select name="Geography_Germany" required="required">
119 <option value="Germany">Germany</option>
120 <option value="Spain">Spain</option>
```

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3
120
121     <option value="Spain">Spain</option>
122     <option value="France">France</option>
123 </select>
124 <div class="validation"></div>
125
126 <div class="form-group">
127 <p>Gender of Customer: <span>*</span></p>
128 <span class="icon-case"></span>
129 <select name="Gender_Male" id="resea" required="required">
130 <option value="Male">Male</option>
131 <option value="Female">Female</option>
132 </select>
133 <div class="validation"></div>
134 </div>
135
136 </div>
137 </div>
138
139 <button type="submit" class="btn btn-lg btn-info mb-5 ml-5" style="box-shadow: 0 4px 15px 0 rgba(49, 196, 190, 0.75); background-color: #007bff; color: white;">Submit</button>
140
141 </form>
142
143
144
145
146
147 <br><br><h3 class="alert alert-primary text-center" role="alert">{{ prediction_text }}</h3>
148 </div>
149 <!-- Footer -->
150 <!-- Footer -->
```

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text > style > body
151 <div class="unique-color-dark mt-4 white-text" style="background-color: #1c2331 !important;">
152 <div class="blue-gradient">
153 <div class="container">
154 <!-- Grid row-->
155 <div class="row py-3 d-flex align-items-center">
156 <!-- Grid column -->
157 <div
158 class="col-md-6 col-lg-5 text-center text-md-left"
159 >
160 <h6 class="mb-0 text-white">Developed by R.sampath rajput <a href="mailto:samsampath2003@gmail.com"><i class="ml-5 fa fa-envelope" style="font-size: 1.2em;"></i></a>
161 </div>
162 <!-- Grid column -->
163
164
165 <!-- Grid column -->
166 </div>
167 <!-- Grid row-->
168 </div>
169 </div>
170
171
172
173
174 <style>
175 * {
176 margin: 0;
177 padding: 0;
178 }
179 body {
180 font-family: sans-serif;
```

```
index.html - Visual Studio Code
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text > style > .promo-title
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
```



```
index.html x
C:\Users\HP\Desktop\COMPLETED>Project 7>Flask>templates>index.html>html>body>h3.alert.alert-primary.text-center>h3>div.unique-color-dark.mt-4.white-text>style>form h1
211 font-weight: 600;
212 margin-top: 100px;
213 }
214 body {
215 margin: auto;
216 background: #eaeaea;
217 font-family: 'Open Sans', sans-serif;
218 }
219 .info p {
220 text-align: center;
221 color: #999;
222 text-transform: none;
223 font-weight: 600;
224 font-size: 15px;
225 margin-top: 2px
226 }
227 .info i {
228 color: #F6AA93;
229 }
230 form h1 {
231 font-size: 18px;
232 background: blue none repeat scroll 0% 0%;
233 color: rgb(255, 255, 255);
234 padding: 22px 25px;
235 border-radius: 5px 5px 0px 0px;
236 margin: auto;
237 text-shadow: none;
238 text-align: left
239 }
240
```

```
index.html x
C:\Users\HP\Desktop\COMPLETED>Project 7>Flask>templates>index.html>html>body>h3.alert.alert-primary.text-center>h3>div.unique-color-dark.mt-4.white-text>style>icon-case
272 width: 75%;
273 height: 40px;
274 float: left;
275 padding: 0px 15px;
276 }
277
278 a {
279 text-decoration: inherit
280 }
281
282 textarea {
283 border-radius: 0px 5px 5px 0px;
284 border: 1px solid #EEE;
285 margin: 0;
286 width: 75%;
287 height: 130px;
288 float: left;
289 padding: 0px 15px;
290 }
291
292 .form-group {
293 overflow: hidden;
294 clear: both;
295 }
296
297 .icon-case {
298 width: 35px;
299 float: left;
300 border-radius: 5px 0px 0px 5px;
301 background: #eeeeee;
```

```
File Edit Selection View Go Run Terminal Help index.html - Visual Studio Code
index.html x
C:\Users\HP\Desktop> All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.rtl-4.white-text > style > leftcontact
301 background-color: #ffffff;
302 height: 42px;
303 position: relative;
304 text-align: center;
305 line-height: 40px;
306 }
307
308 i {
309 color: #555;
310 }
311
312 .contentform {
313 padding: 40px 30px;
314 }
315
316 .bouton-contact{
317 background-color: #818DA4;
318 color: #FFF;
319 text-align: center;
320 width: 100%;
321 border: 0;
322 padding: 17px 25px;
323 border-radius: 0px 0px 5px 5px;
324 cursor: pointer;
325 margin-top: 40px;
326 font-size: 18px;
327 }
328
329 .leftcontact {
330 width: 49.5%;
331 float: left;
```

```
index.html - Visual Studio Code
File Edit Selection View Go Run Terminal Help
index.html x
C:\Users\HP\Desktop\All Projects\COMPLETED\Project 7> Flask> templates> index.html > html > body > h3.alert-primary.text-center > h3 > div.unique-color-darkmt-4.white-text > style > #sendMessage
331 float:left;
332 border-right: 1px dotted #CCC;
333 box-sizing: border-box;
334 padding: 0px 15px 0px 0px;
335 }
336
337 .rightcontact {
338 width:49.5%;
339 float:right;
340 box-sizing: border-box;
341 padding: 0px 0px 0px 15px;
342 }
343
344 .validation {
345 display:none;
346 margin: 0 0 10px;
347 font-weight:400;
348 font-size:13px;
349 color: #DE5959;
350 }
351
352 #sendMessage {
353 border:1px solid #fff;
354 display:none;
355 text-align:center;
356 margin:10px 0;
357 font-weight:600;
358 margin-bottom:30px;
359 background-color: #EBF6E0;
360 color: #5F9025;
361 }
362
363 #sendMessage.show, .show {
364 display:block;
365 }
366
367
368
369
370
371
372
373 </style>
374 <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpmoFV38MVBnI" >
375 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaE" >
376 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JhLku0U7X" >
377 </body>
378
379 </html>
```

App.py

```
File Edit Selection View Go Run Terminal Help
app.py - Visual Studio Code

app.py x
C:\Users\HP\Desktop > All Projects > COMPLETED > Project 7 > Flask > app.py

1 import pickle
2
3 from flask import Flask, render_template, request
4 from sklearn.preprocessing import StandardScaler
5
6 app = Flask(__name__)
7 model = pickle.load(open('Customer_Churn_Prediction.pkl', 'rb'))
8 @app.route('/', methods=['GET'])
9 def Home():
10     return render_template('index.html')
11
12 standard_to = StandardScaler()
13 @app.route('/predict', methods=['POST'])
14 def predict():
15     if request.method == 'POST':
16         CreditScore = int(request.form['CreditScore'])
17         Age = int(request.form['Age'])
18         Tenure = int(request.form['Tenure'])
19         Balance = float(request.form['Balance'])
20         NumOfProducts = int(request.form['NumOfProducts'])
21         HasCrCard = int(request.form['HasCrCard'])
22         IsActiveMember = int(request.form['IsActiveMember'])
23         EstimatedSalary = float(request.form['EstimatedSalary'])
24         Geography_Germany = request.form['Geography_Germany']
25         if Geography_Germany == 'Germany':
26             Geography_Germany = 1
27             Geography_Spain = 0
28             Geography_France = 0
29
30         elif Geography_Germany == 'Spain':
31             Geography_Germany = 0
```

```
File Edit Selection View Go Run Terminal Help
app.py - Visual Studio Code

app.py x
C:\Users\HP\Desktop > All Projects > COMPLETED > Project 7 > Flask > app.py

26         Geography_Germany = 1
27         Geography_Spain = 0
28         Geography_France = 0
29
30     elif Geography_Germany == 'Spain':
31         Geography_Germany = 0
32         Geography_Spain = 1
33         Geography_France = 0
34
35     else:
36         Geography_Germany = 0
37         Geography_Spain = 0
38         Geography_France = 1
39         Gender_Male = request.form['Gender_Male']
40         if Gender_Male == 'Male':
41             Gender_Male = 1
42             Gender_Female = 0
43         else:
44             Gender_Male = 0
45             Gender_Female = 1
46         prediction = model.predict([[CreditScore, Age, Tenure, Balance, NumOfProducts, HasCrCard, IsActiveMember, EstimatedSalary, Geography_Germany, Geography_Spain, Geography_France, Gender_Male, Gender_Female]])
47         if prediction==1:
48             return render_template('index.html', prediction_text="The Churn prediction is True")
49         else:
50             return render_template('index.html', prediction_text="The Churn prediction is false")
51
52 if __name__ == "__main__":
53     app.run(debug=True)
54
```