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(tvu10520CA47)

Team member : NITHYASRI N(tvu10520CA33)

Team member : NIVETHA K(tvu10520CA34)

Team member : VANITHA E(tvu10520CA46)

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 amongcompanies, 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.

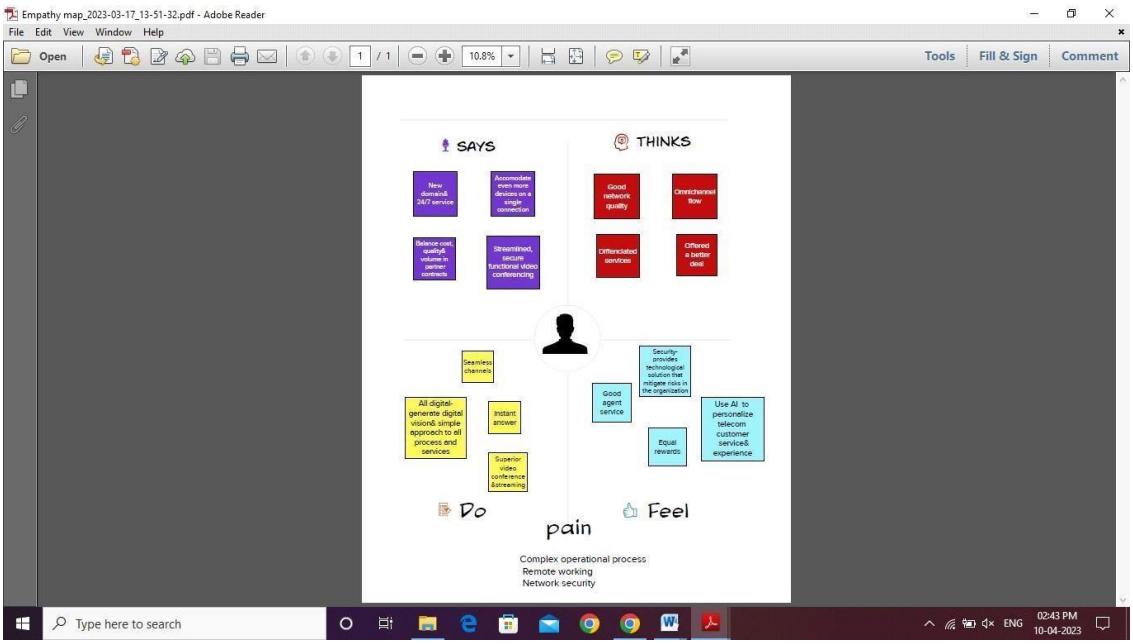
1.2 Purpose

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-tocustomer 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 subscriptionbased 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:

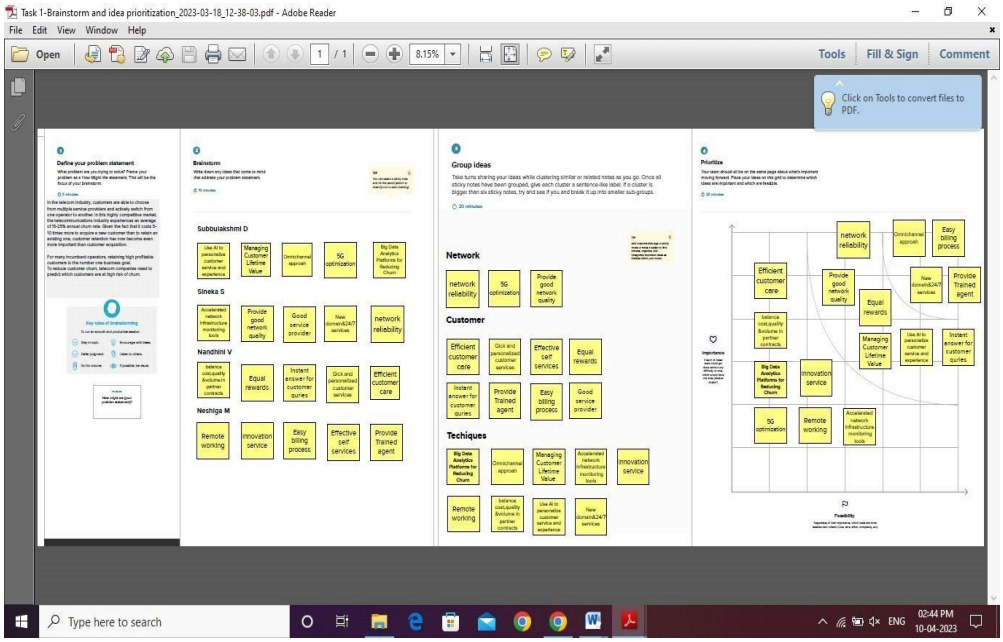
i) construct a binary churn prediction model, ii) compare different types of algorithms for the built model, and iii) study the effects of balancing the training dataset for the considered model

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.
EXIT
Click me to continue with prediction

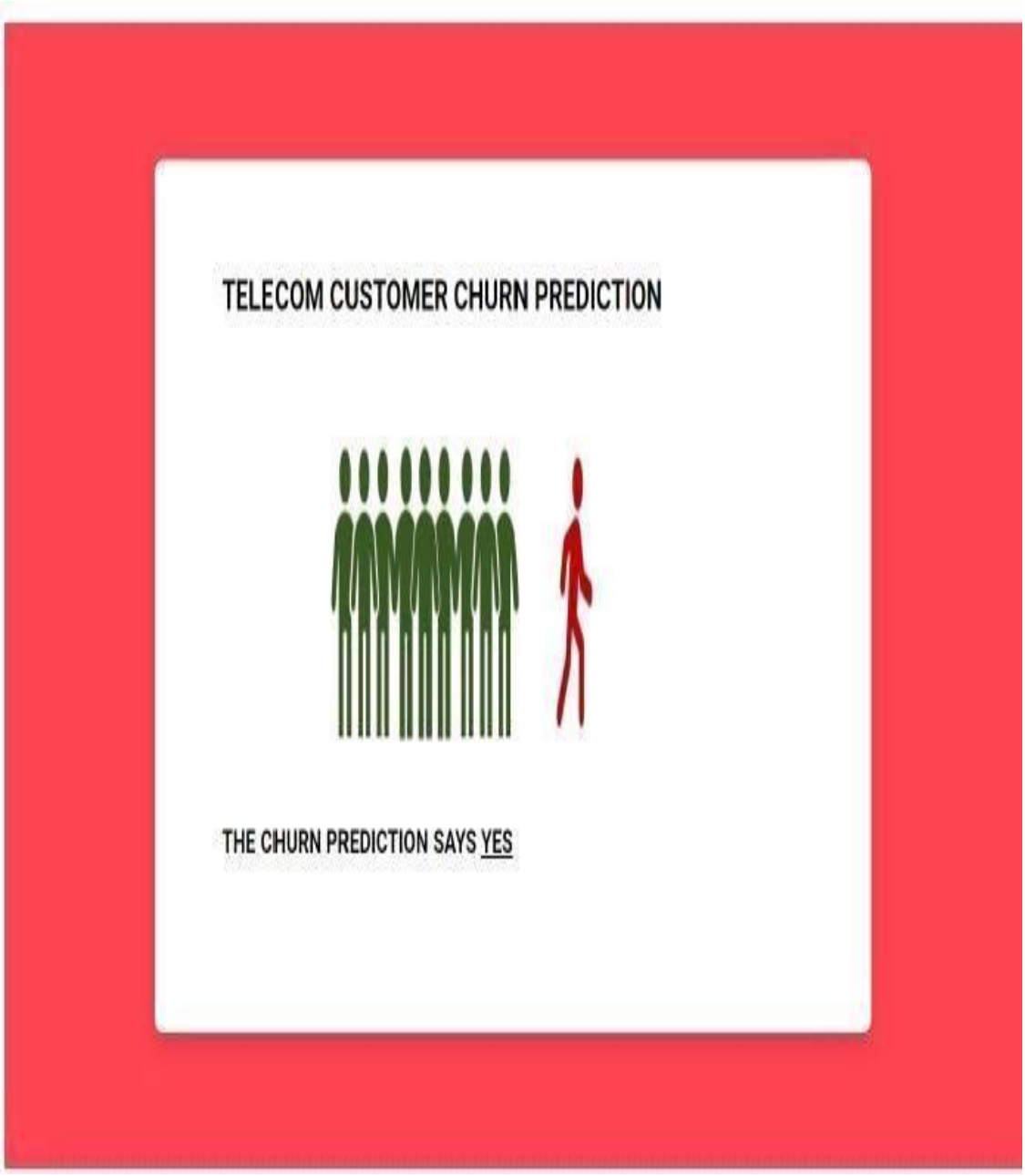
PREDICTION FORM

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:

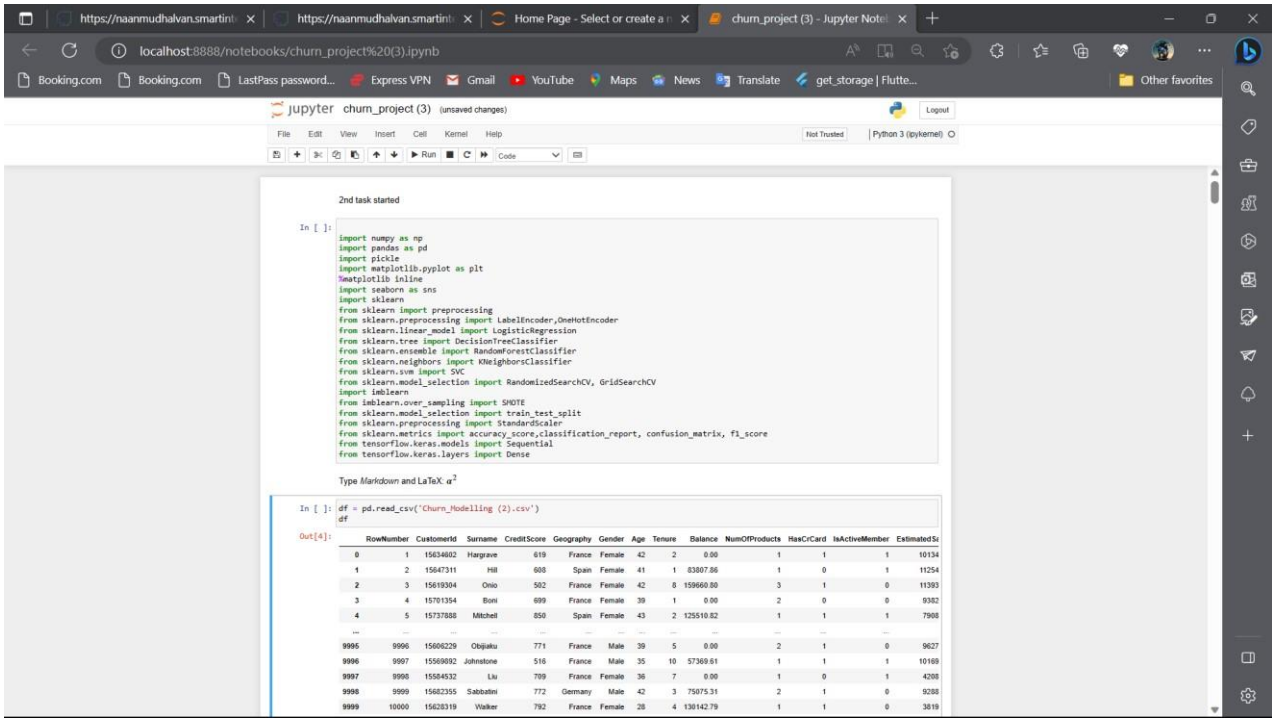
- Records that contain more than 90% of missing features were deleted.
- Features that have more than 70% of missing values were deleted.
- For the missing categories in categorical features, they were replaced by a new category called 'Other'.
- The missing numerical values were replaced with the average 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.

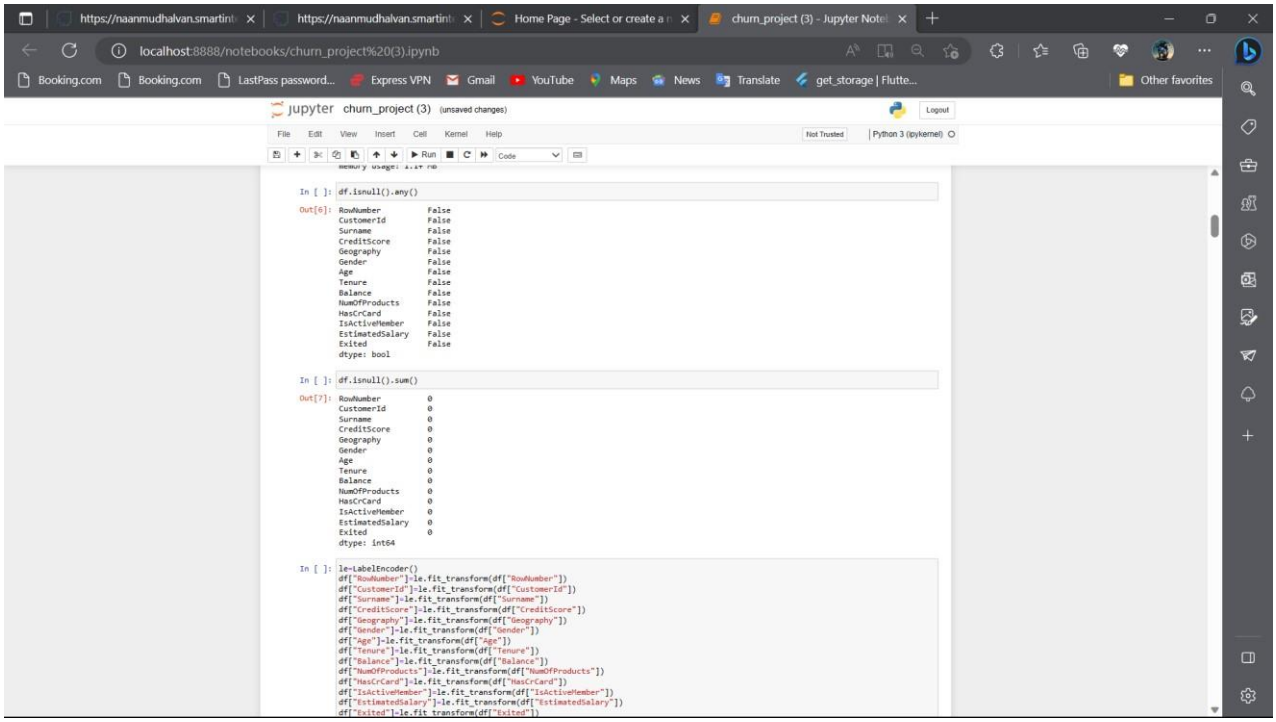
	• We have also calculated the correlation between numerical features using Pearson and removed the correlated features. This removal had no effect on the final result. Many other methods were tested, but this applied approach gave the best performance of the four algorithms. The number of features after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

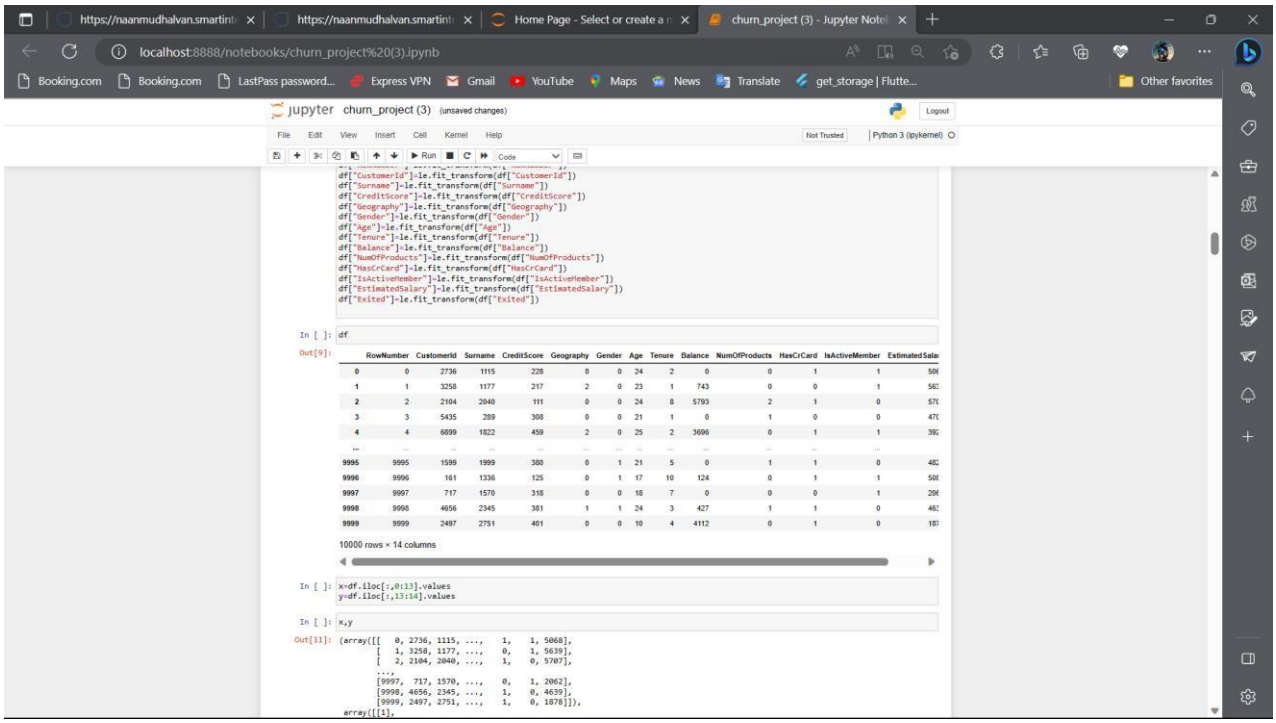
8 . APPENDIX

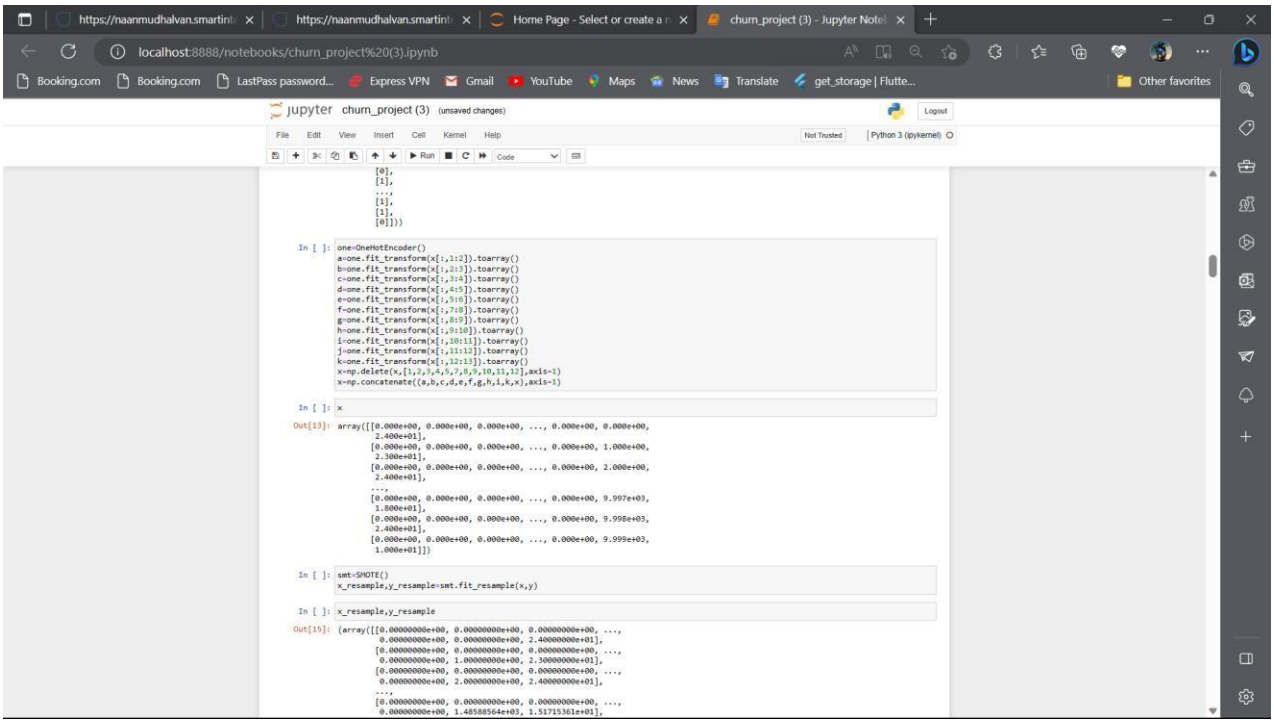
SOURCE CODE

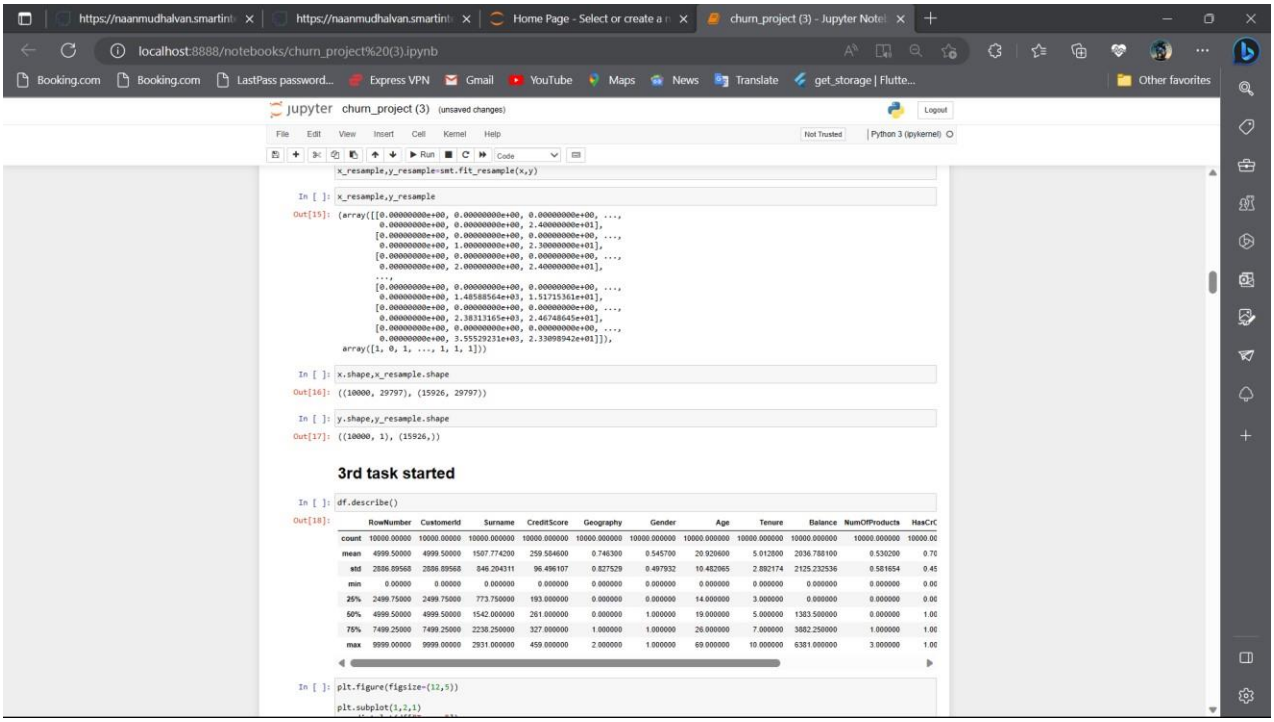
churn_project (3).ipynb

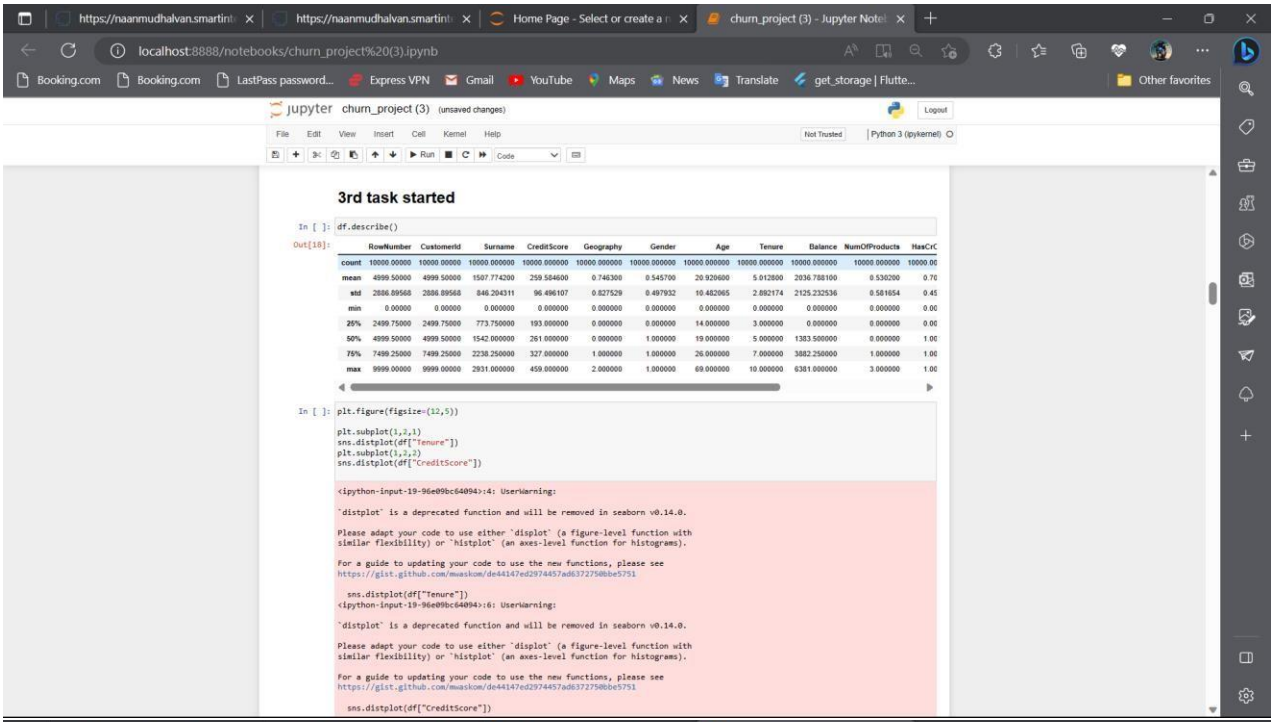


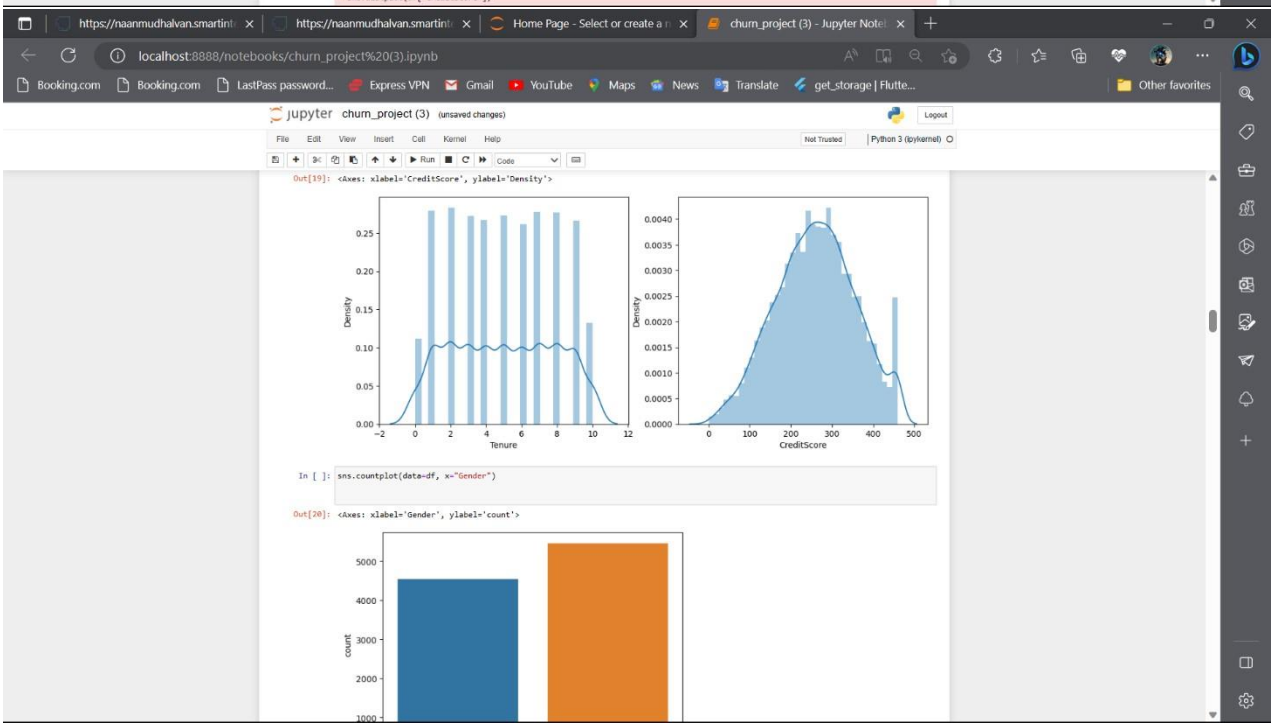


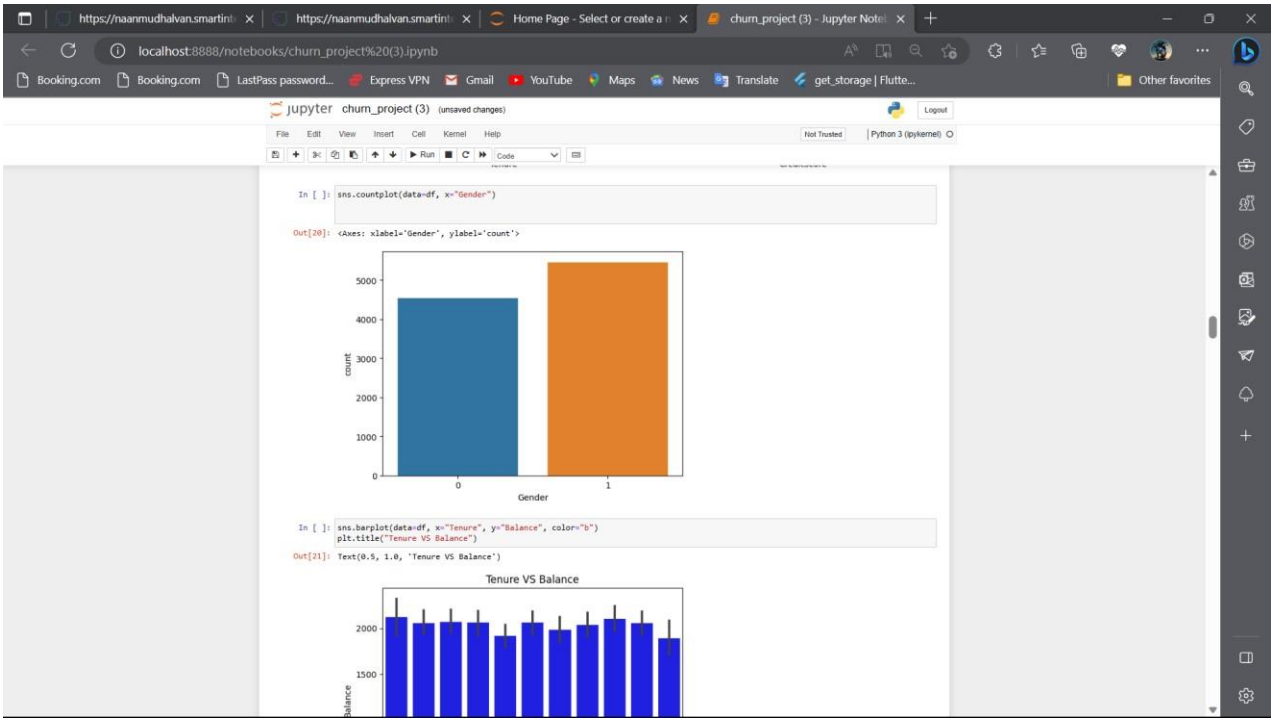

churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (unsaved changes)
df["CustomerId"]=le.fit_transform(df["CustomerId"])
df["Surname"]=le.fit_transform(df["Surname"])
df["CreditScore"]=le.fit_transform(df["CreditScore"])
df["Geography"]=le.fit_transform(df["Geography"])
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
0 0 2736 1115 228 0 0 24 2 0 0 1 1
1 1 3258 1177 217 2 0 23 1 743 0 0 1
2 2 2104 2040 111 0 0 24 8 5793 2 1 0
3 3 5435 289 308 0 0 21 1 0 1 0 0
4 4 6899 1822 459 2 0 25 2 3696 0 1 1
9995 9995 1599 1999 380 0 1 21 5 0 1 1 0
9996 9996 161 1336 125 0 1 17 10 124 0 1 1
9997 9997 717 1570 318 0 0 18 7 0 0 0 1
9998 9998 4656 2345 381 1 1 24 3 427 1 1 0
9999 9999 2497 2751 401 0 0 10 4 4112 0 1 0
10000 rows × 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, 1, 5068],
[1, 3258, 1177, ..., 0, 1, 5639],
[2, 2104, 2040, ..., 1, 0, 5707],
...,
[9997, 717, 1570, ..., 0, 1, 2062],
[9998, 4656, 2345, ..., 1, 0, 4639],
[9999, 2497, 2751, ..., 1, 0, 1878]]),
array([[1],


churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (unsaved changes)
[0],
[1],
...,
[1],
[1],
[0]]))
In [ ]: one=OneHotEncoder()
a=one.fit_transform(x[:,1:2]).toarray()
b=one.fit_transform(x[:,2:3]).toarray()
c=one.fit_transform(x[:,3:4]).toarray()
d=one.fit_transform(x[:,4:5]).toarray()
e=one.fit_transform(x[:,5:6]).toarray()
f=one.fit_transform(x[:,7:8]).toarray()
g=one.fit_transform(x[:,8:9]).toarray()
h=one.fit_transform(x[:,9:10]).toarray()
i=one.fit_transform(x[:,10:11]).toarray()
j=one.fit_transform(x[:,11:12]).toarray()
k=one.fit_transform(x[:,12:13]).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,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,
1.800e+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 [ ]: smt=SMOTE()
x_resample,y_resample=smt.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, 1.48588564e+03, 1.51715361e+01],


https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Notel
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)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
x_resample,y_resample=smt.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, 1.48588564e+03, 1.51715361e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 2.38313165e+03, 2.46748645e+01],
[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
0.00000000e+00, 3.55529231e+03, 2.33098942e+01]]),
array([1, 0, 1, ..., 1, 1, 1]))
In [ ]: x.shape,x_resample.shape
Out[16]: ((10000, 29797), (15926, 29797))
In [ ]: y.shape,y_resample.shape
Out[17]: ((10000, 1), (15926,))
3rd task started
In [ ]: df.describe()
Out[18]:
RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrC
count 10000.00000 10000.00000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.00
mean 4999.50000 4999.50000 1507.774200 259.584600 0.746300 0.545700 20.920600 5.012800 2036.788100 0.530200 0.70
std 2886.89568 2886.89568 846.204311 96.496107 0.827529 0.497932 10.482065 2.892174 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 3882.250000 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
In [ ]: plt.figure(figsize=(12,5))
plt.subplot(1,2,1)

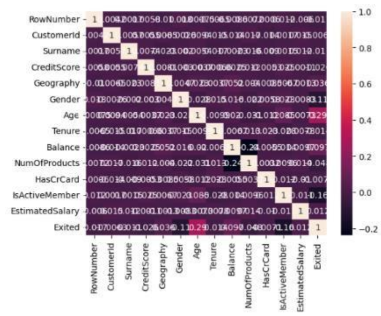

https://naanmudhalvan.smartintr
https://naanmudhalvan.smartintr
Home Page - Select or create a n
churn_project (3) - Jupyter Notel
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
3rd task started
In [ ]: df.describe()
Out[18]:
RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrC
count 10000.00000 10000.00000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.00
mean 4999.50000 4999.50000 1507.774200 259.584600 0.746300 0.545700 20.920600 5.012800 2036.788100 0.530200 0.70
std 2886.89568 2886.89568 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 3882.250000 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
In [ ]: plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
sns.distplot(df["Tenure"])
plt.subplot(1,2,2)
sns.distplot(df["CreditScore"])
<ipython-input-19-96e09bc64094>:4: UserWarning:
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).
For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
sns.distplot(df["Tenure"])
<ipython-input-19-96e09bc64094>:6: UserWarning:
`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).
For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
sns.distplot(df["CreditScore"])


https://naanmudhalvan.smartintr
https://naanmudhalvan.smartintr
Home Page - Select or create a n
churn_project (3) - Jupyter Notel
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)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Out[19]: <Axes: xlabel='CreditScore', ylabel='Density'>
Density
Tenure
CreditScore
In [ ]: sns.countplot(data=df, x="Gender")
Out[20]: <Axes: xlabel='Gender', ylabel='count'>
count


https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Notel
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)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [ ]: sns.countplot(data=df, x="Gender")
Out[20]: <Axes: xlabel='Gender', ylabel='count'>
count
Gender
In [ ]: sns.barplot(data=df, x="Tenure", y="Balance", color="b")
plt.title("Tenure VS Balance")
Out[21]: Text(0.5, 1.0, 'Tenure VS Balance')
Tenure VS Balance
Balance

jupyter churn_project (3) (unsaved changes)

```
In [ ]: sns.heatmap(df.corr(),annot=True)
Out[22]: <Axes: >
```




```
In [ ]: sns.pairplot(data=df, markers=["^","v"], palette="inferno")
```

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

jupyter churn_project (3) (unsaved changes)

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

```
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

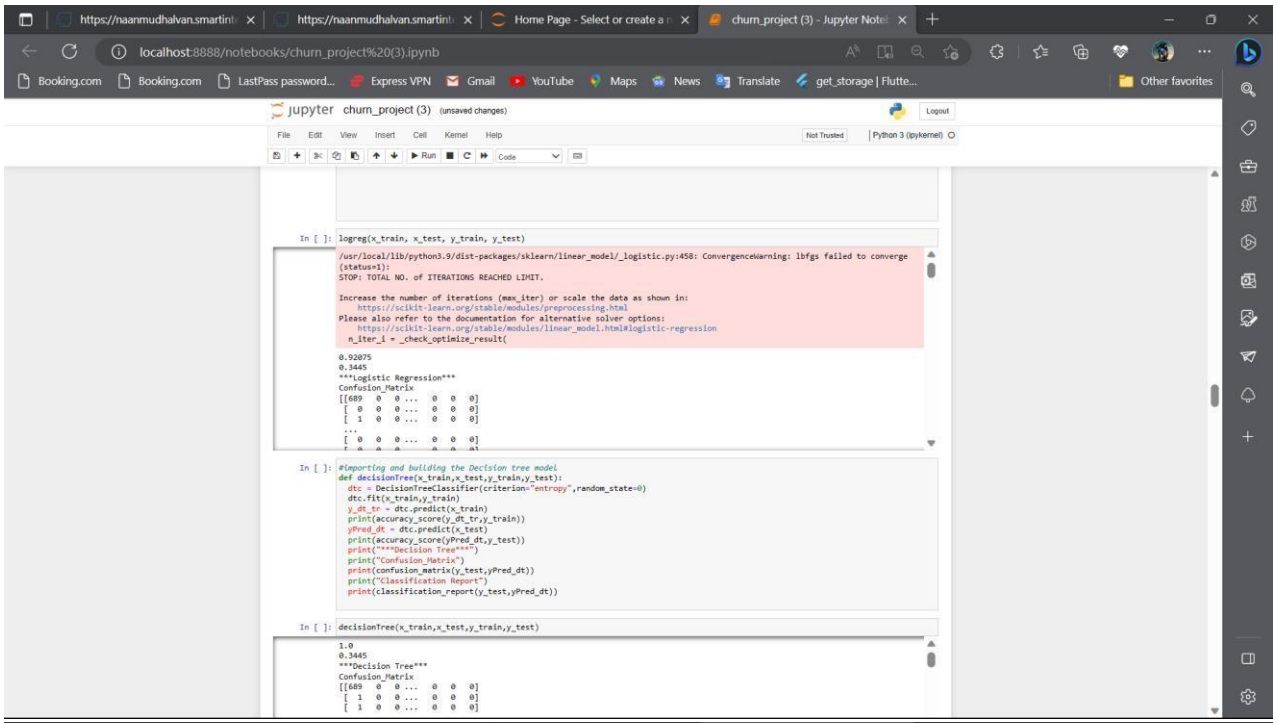
```
In [ ]: x_train.shape
Out[26]: (8000, 13)
```

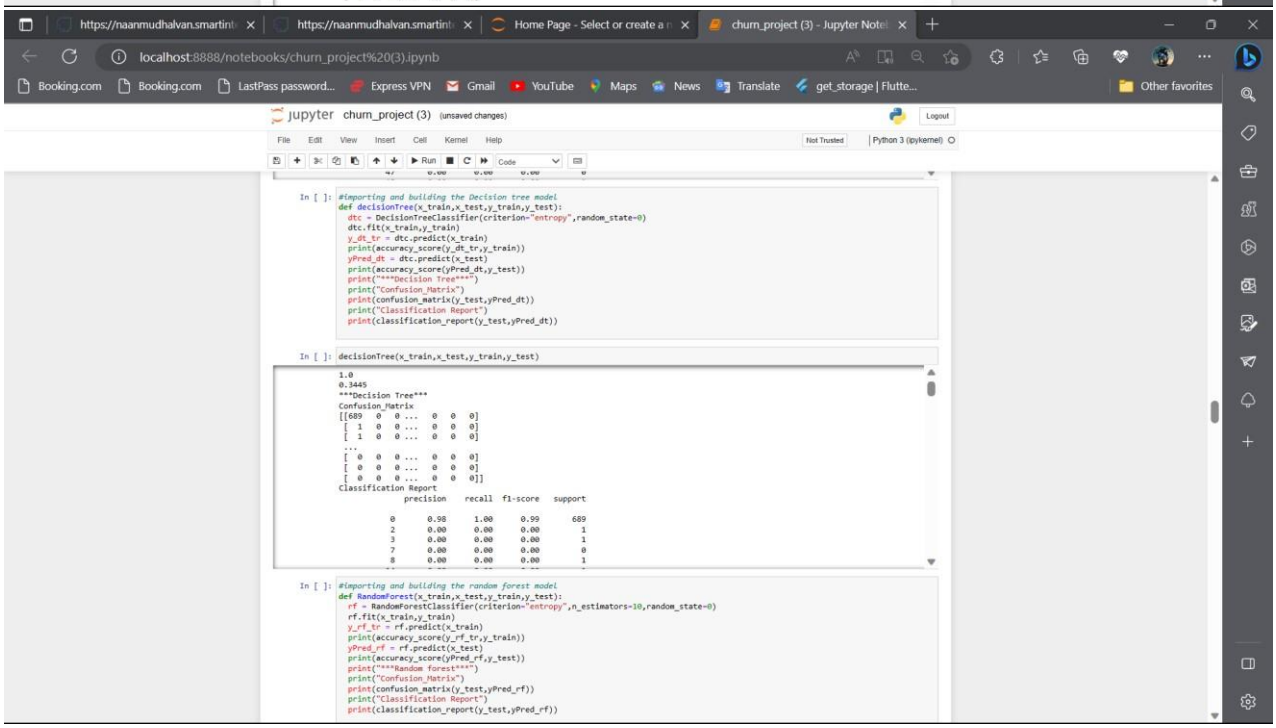
```
In [ ]: def logreg(x_train,x_test,y_train,y_test):
          lr=LogisticRegression(random_state=0)
          lr.fit(x_train,y_train)
          y_lr_tr = lr.predict(x_train)
```

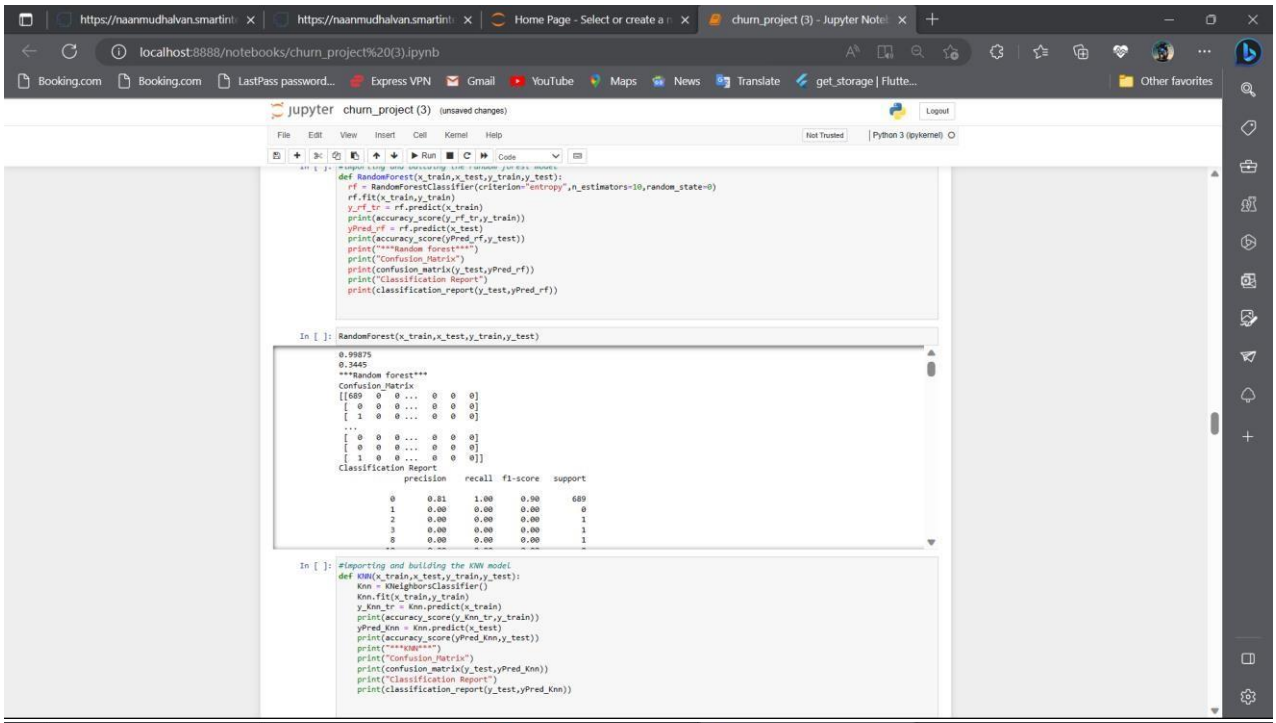
```
In [ ]: #importing and building the Decision tree model
        def logreg(x_train,x_test,y_train,y_test):
          lr=LogisticRegression(random_state=0)
          lr.fit(x_train,y_train)
          y_lr_tr = lr.predict(x_train)
          print(accuracy_score(y_lr_tr,y_train))
          yPred_lr = lr.predict(x_test)
          print(accuracy_score(yPred_lr,y_test))
          print("***Logistic Regression***")
          print("Confusion_Matrix")
          print(confusion_matrix(y_test,yPred_lr))
          print("Classification Report")
          print(classification_report(y_test,yPred_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:
```


churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (unsaved changes)
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
In [ ]: #importing 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))
yPred_dt = dtc.predict(x_test)
print(accuracy_score(yPred_dt,y_test))
print("***Decision Tree***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,yPred_dt))
print("Classification Report")
print(classification_report(y_test,yPred_dt))
In [ ]: decisionTree(x_train,x_test,y_train,y_test)
1.0
0.3445
Decision Tree
Confusion_Matrix


churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (unsaved changes)
In [ ]: #importing 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))
yPred_dt = dtc.predict(x_test)
print(accuracy_score(yPred_dt,y_test))
print("***Decision Tree***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,yPred_dt))
print("Classification Report")
print(classification_report(y_test,yPred_dt))
In [ ]: decisionTree(x_train,x_test,y_train,y_test)
1.0
0.3445
Decision Tree
Confusion_Matrix
Classification Report
precision recall f1-score support
In [ ]: #importing 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))
yPred_rf = rf.predict(x_test)
print(accuracy_score(yPred_rf,y_test))
print("***Random forest***")
print("Confusion_Matrix")
print(confusion_matrix(y_test,yPred_rf))
print("Classification Report")
print(classification_report(y_test,yPred_rf))

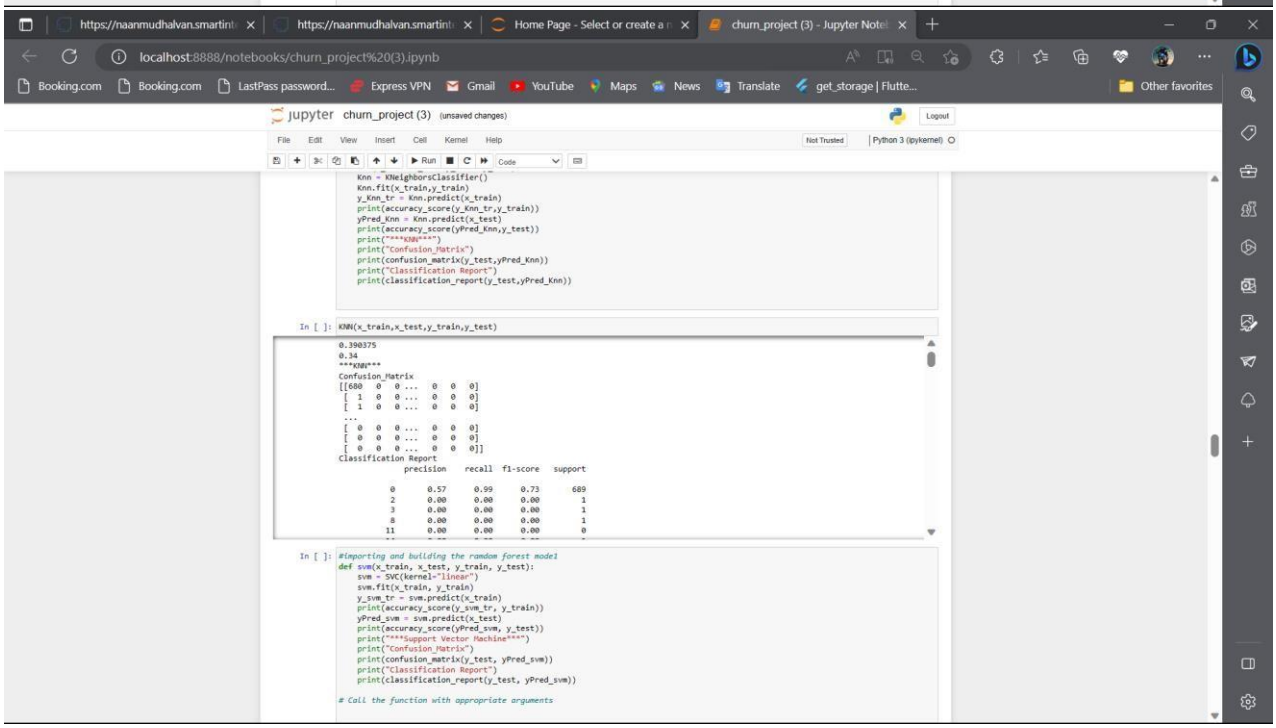


```
def RandomForest(x_train,x_test,y_train,y_test):
    rf = RandomForestClassifier(criterion="entropy",n_estimators=10,random_state=0)
    rf.fit(x_train,y_train)
    y_rf_tr = rf.predict(x_train)
    print(accuracy_score(y_rf_tr,y_train))
    yPred_rf = rf.predict(x_test)
    print(accuracy_score(yPred_rf,y_test))
    print("***Random forest***")
    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
0.3445
***Random forest***
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]
 [  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         0
           2       0.00      0.00      0.00         1
           3       0.00      0.00      0.00         1
           8       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("***KNN***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_Knn))
    print("Classification Report")
    print(classification_report(y_test,yPred_Knn))
```

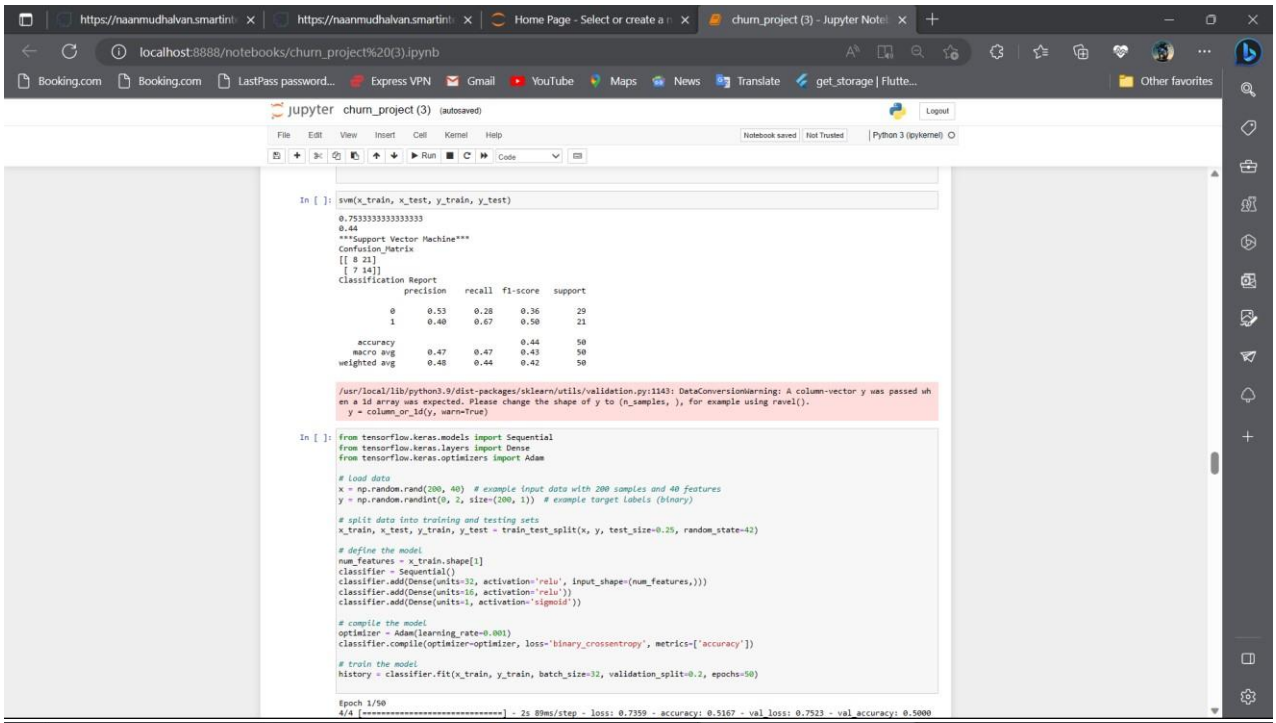


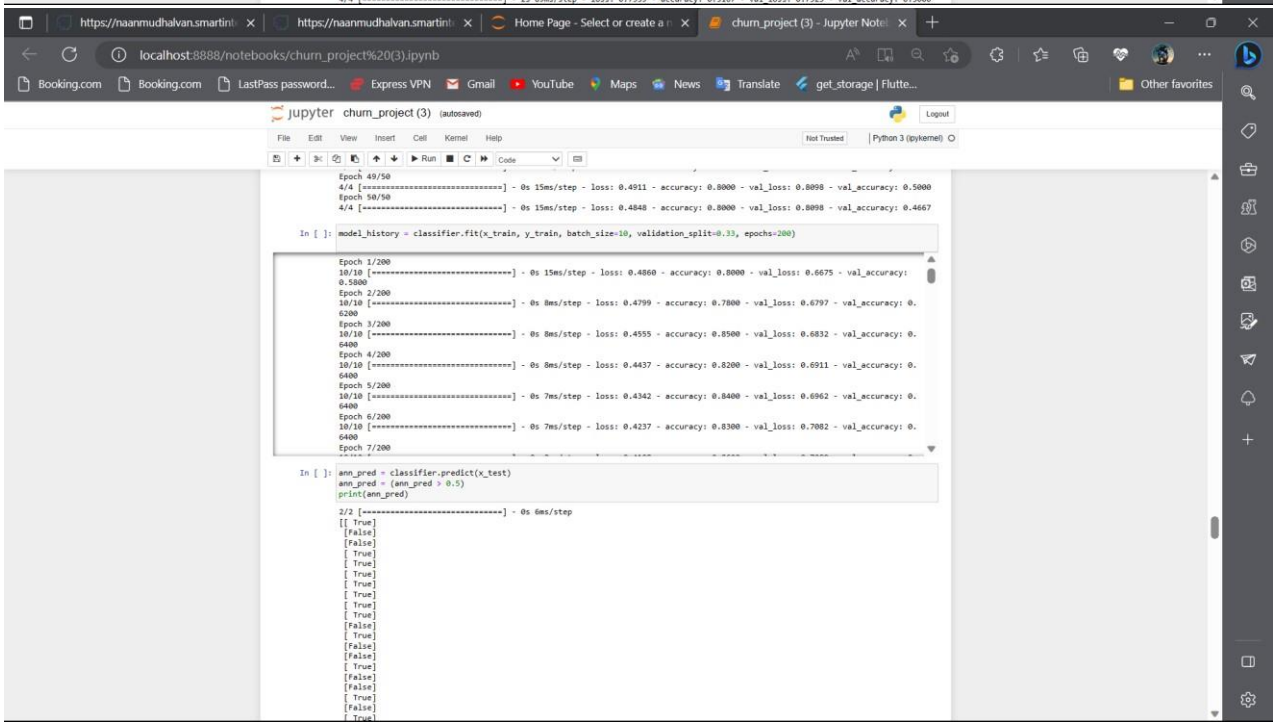
```
    Knn = KNeighborsClassifier()
    Knn.fit(x_train,y_train)
    y_Knn_tr = Knn.predict(x_train)
    print(accuracy_score(y_Knn_tr,y_train))
    yPred_Knn = Knn.predict(x_test)
    print(accuracy_score(yPred_Knn,y_test))
    print("***KNN***")
    print("Confusion_Matrix")
    print(confusion_matrix(y_test,yPred_Knn))
    print("Classification Report")
    print(classification_report(y_test,yPred_Knn))
```

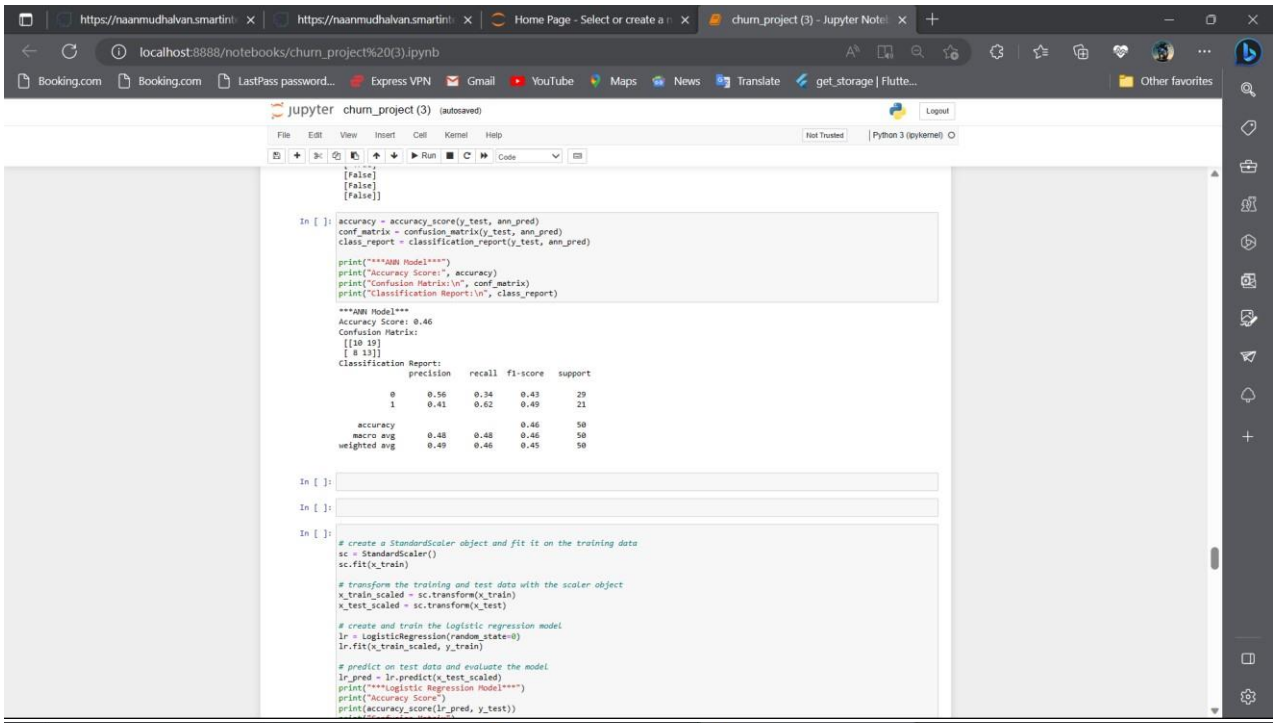
In [ ]:
```
KNN(x_train,x_test,y_train,y_test)
```

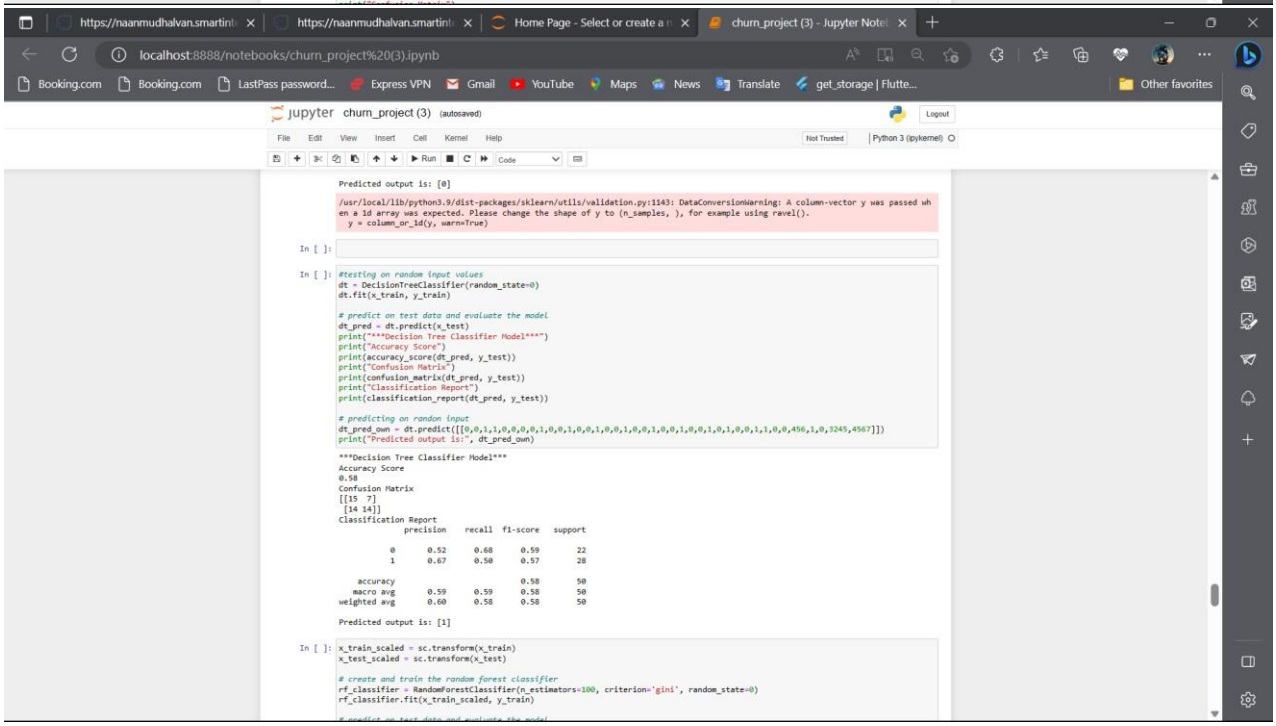
```
0.390375
0.34
***KNN***
Confusion_Matrix
[[680   0   0 ...   0   0   0]
 [  1   0   0 ...   0   0   0]
 [  1   0   0 ...   0   0   0]
 ...
 [  0   0   0 ...   0   0   0]
 [  0   0   0 ...   0   0   0]
 [  0   0   0 ...   0   0   0]]
Classification Report
              precision    recall  f1-score   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
           8       0.00      0.00      0.00         1
          11       0.00      0.00      0.00         0
```

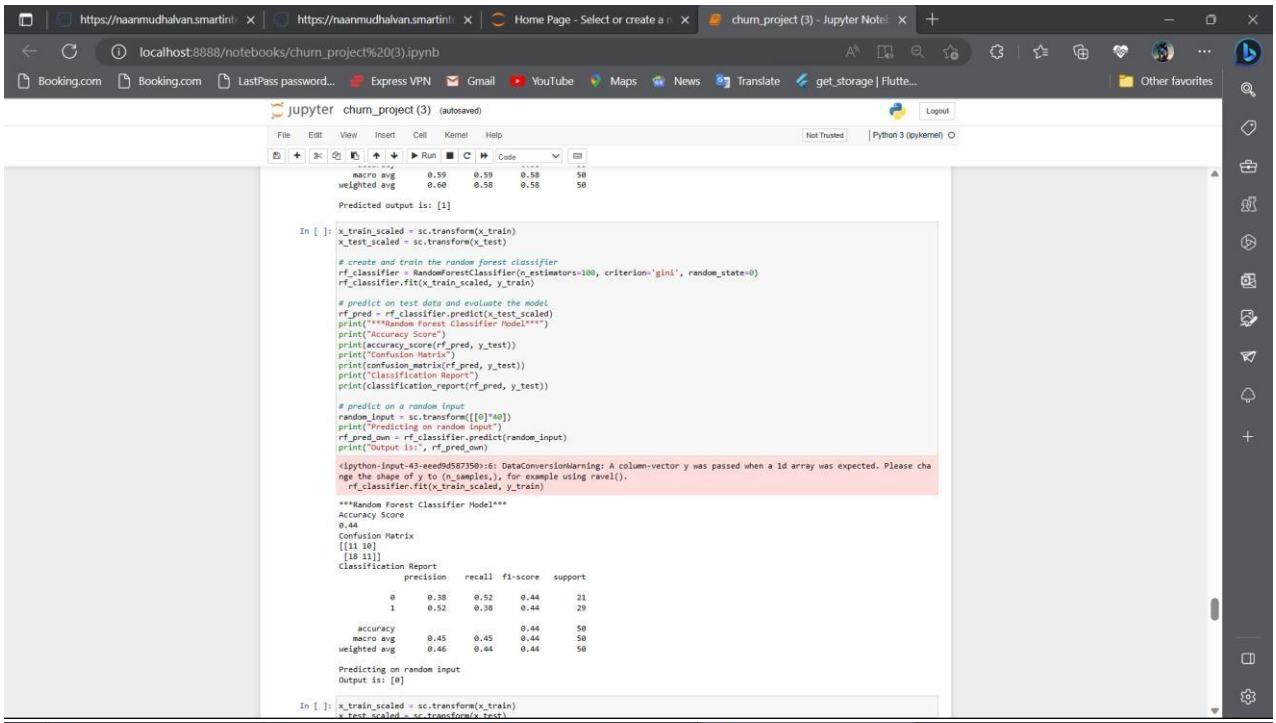
In [ ]:
```
#importing and building the random forest model
def svm(x_train, x_test, y_train, y_test):
    svm = SVC(kernel='linear')
    svm.fit(x_train, y_train)
    y_svm_tr = svm.predict(x_train)
    print(accuracy_score(y_svm_tr, y_train))
    yPred_svm = svm.predict(x_test)
    print(accuracy_score(yPred_svm, y_test))
    print("***Support Vector Machine***")
    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
```

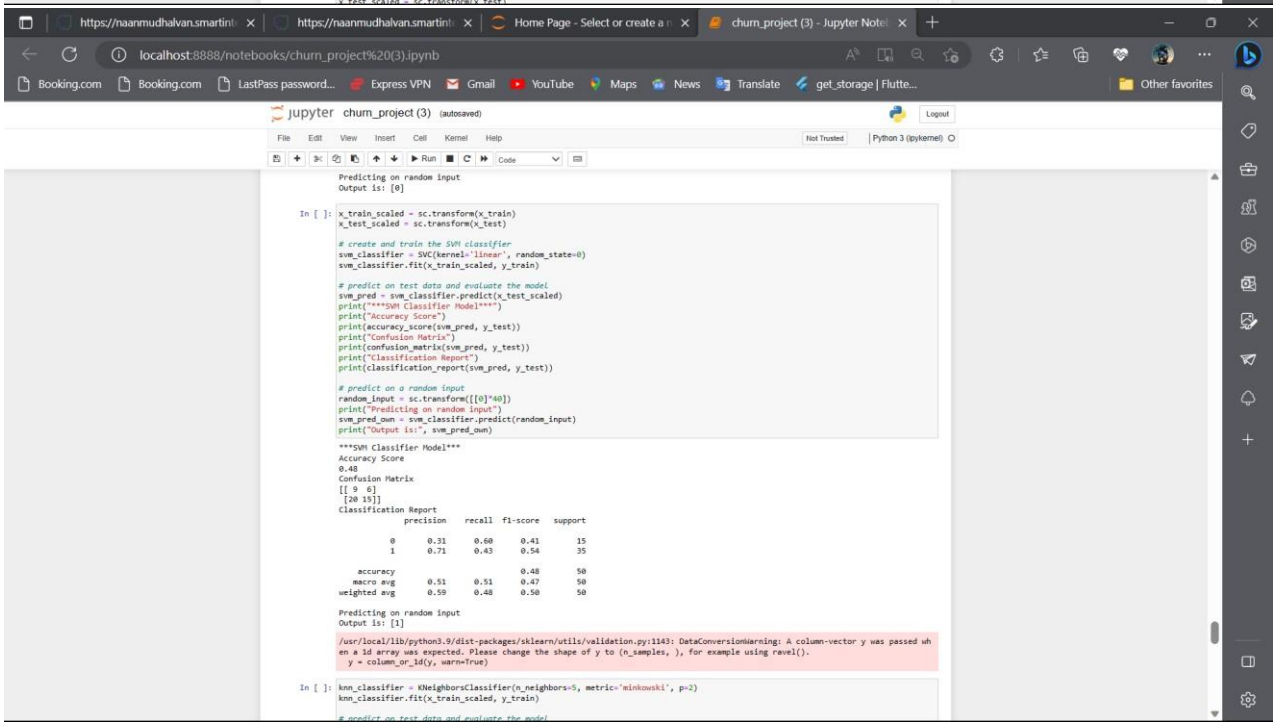

https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Note
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Notebook saved
Not Trusted
Python 3 (ipykernel)
Run
Code
In [ ]:
svm(x_train, x_test, y_train, y_test)
0.7533333333333333
0.44
Support Vector Machine
Confusion_Matrix
[[8 21]
[7 14]]
Classification Report
precision recall f1-score support
0 0.53 0.28 0.36 29
1 0.40 0.67 0.50 21
accuracy 0.44 50
macro avg 0.47 0.47 0.43 50
weighted avg 0.48 0.44 0.42 50
/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed wh
en 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 [ ]:
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
# load data
x = np.random.rand(200, 40) # example input data with 200 samples and 40 features
y = np.random.randint(0, 2, size=(200, 1)) # example target labels (binary)
# split data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=42)
# define the model
num_features = x_train.shape[1]
classifier = Sequential()
classifier.add(Dense(units=32, activation='relu', input_shape=(num_features,)))
classifier.add(Dense(units=16, activation='relu'))
classifier.add(Dense(units=1, activation='sigmoid'))
# compile the model
optimizer = Adam(learning_rate=0.001)
classifier.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])
# train the model
history = classifier.fit(x_train, y_train, batch_size=32, validation_split=0.2, epochs=50)
Epoch 1/50
4/4 [==============================] - 2s 89ms/step - loss: 0.7359 - accuracy: 0.5167 - val_loss: 0.7523 - val_accuracy: 0.5000

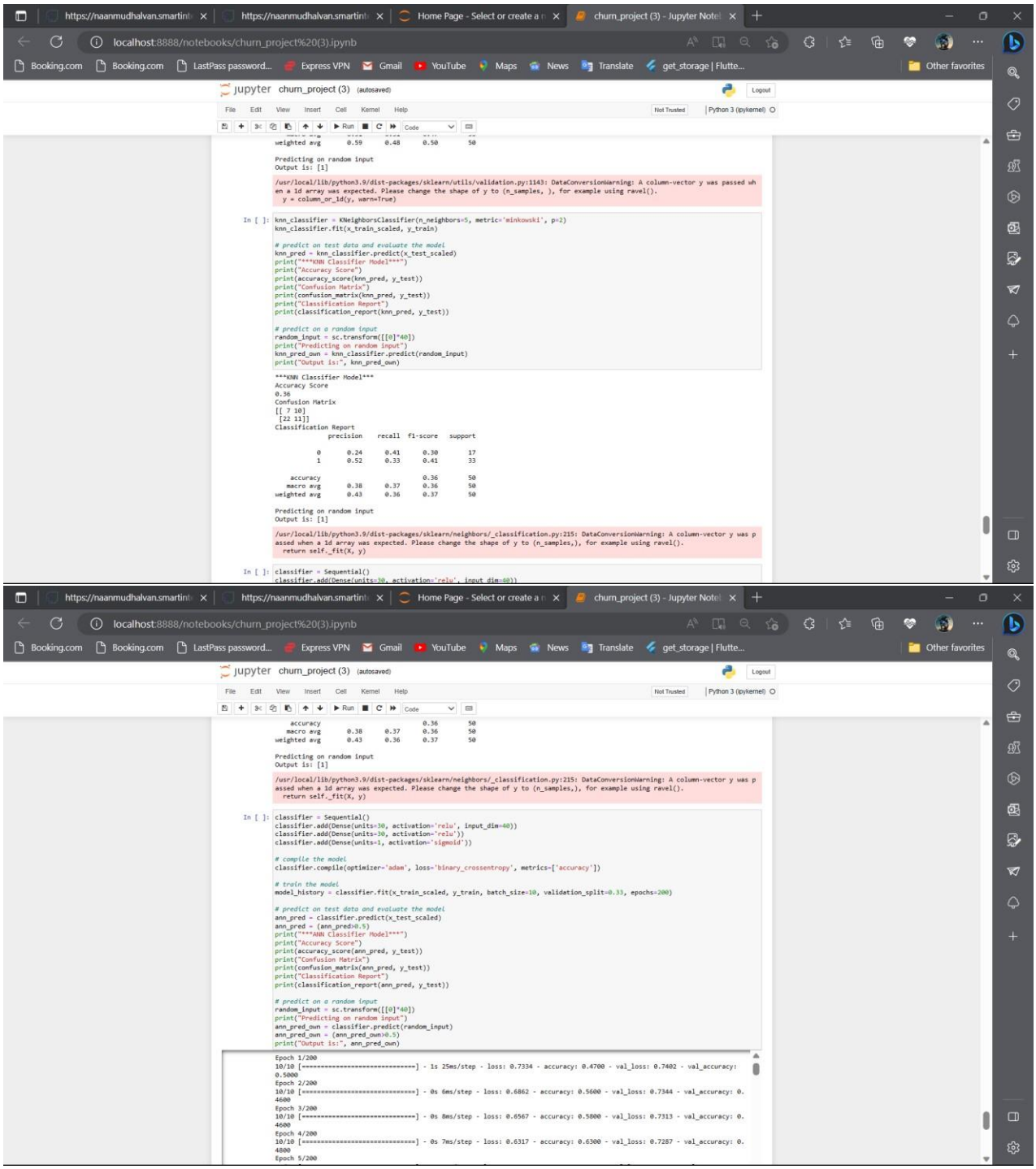

https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Note
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Epoch 49/50
4/4 [==============================] - 0s 15ms/step - loss: 0.4911 - accuracy: 0.8000 - val_loss: 0.8098 - val_accuracy: 0.5000
Epoch 50/50
4/4 [==============================] - 0s 15ms/step - loss: 0.4848 - accuracy: 0.8000 - val_loss: 0.8098 - val_accuracy: 0.4667
In [ ]:
model_history = classifier.fit(x_train, y_train, batch_size=10, validation_split=0.33, epochs=200)
Epoch 1/200
10/10 [==============================] - 0s 15ms/step - loss: 0.4860 - accuracy: 0.8000 - val_loss: 0.6675 - val_accuracy:
0.5800
Epoch 2/200
10/10 [==============================] - 0s 8ms/step - loss: 0.4799 - accuracy: 0.7800 - val_loss: 0.6797 - val_accuracy: 0.
6200
Epoch 3/200
10/10 [==============================] - 0s 8ms/step - loss: 0.4555 - accuracy: 0.8500 - val_loss: 0.6832 - val_accuracy: 0.
6400
Epoch 4/200
10/10 [==============================] - 0s 8ms/step - loss: 0.4437 - accuracy: 0.8200 - val_loss: 0.6911 - val_accuracy: 0.
6400
Epoch 5/200
10/10 [==============================] - 0s 7ms/step - loss: 0.4342 - accuracy: 0.8400 - val_loss: 0.6962 - val_accuracy: 0.
6400
Epoch 6/200
10/10 [==============================] - 0s 7ms/step - loss: 0.4237 - accuracy: 0.8300 - val_loss: 0.7082 - val_accuracy: 0.
6400
Epoch 7/200
In [ ]:
ann_pred = classifier.predict(x_test)
ann_pred = (ann_pred > 0.5)
print(ann_pred)
2/2 [==============================] - 0s 6ms/step
[[True]
[False]
[False]
[True]
[True]
[True]
[True]
[True]
[True]
[False]
[True]
[False]
[False]
[True]
[False]
[False]
[True]
[False]
[True]

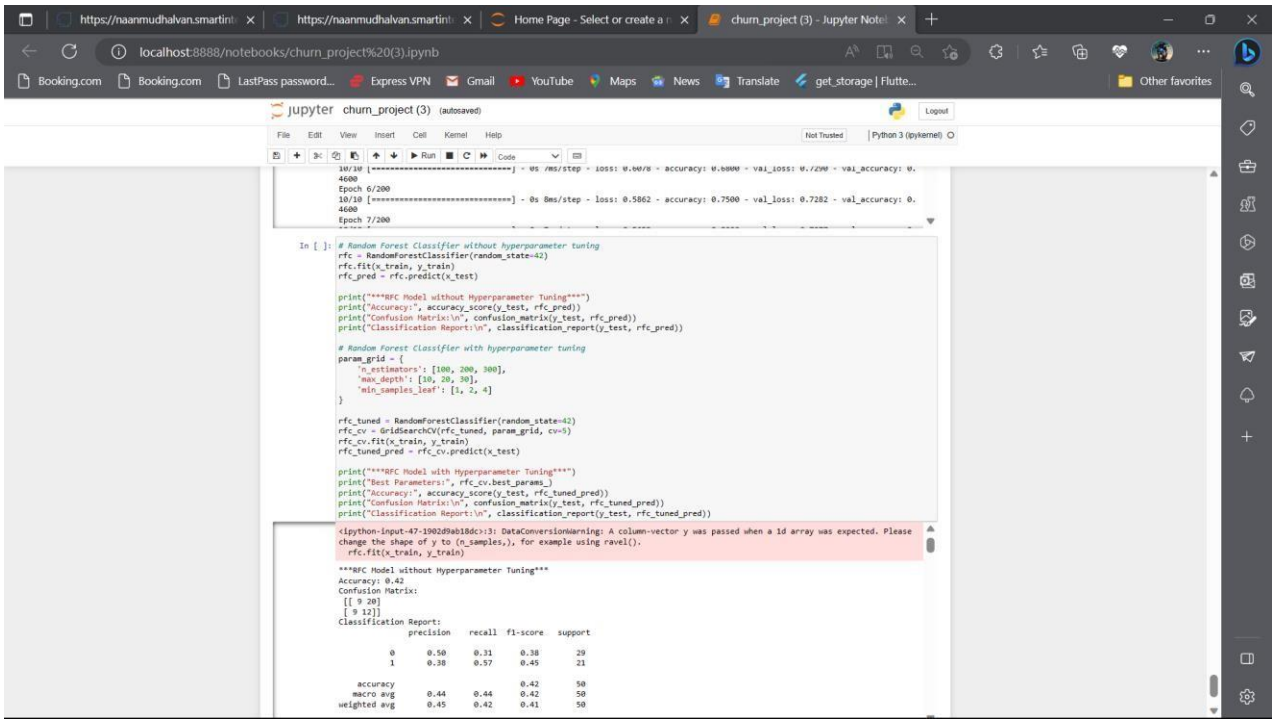

https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Note
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
[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 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))

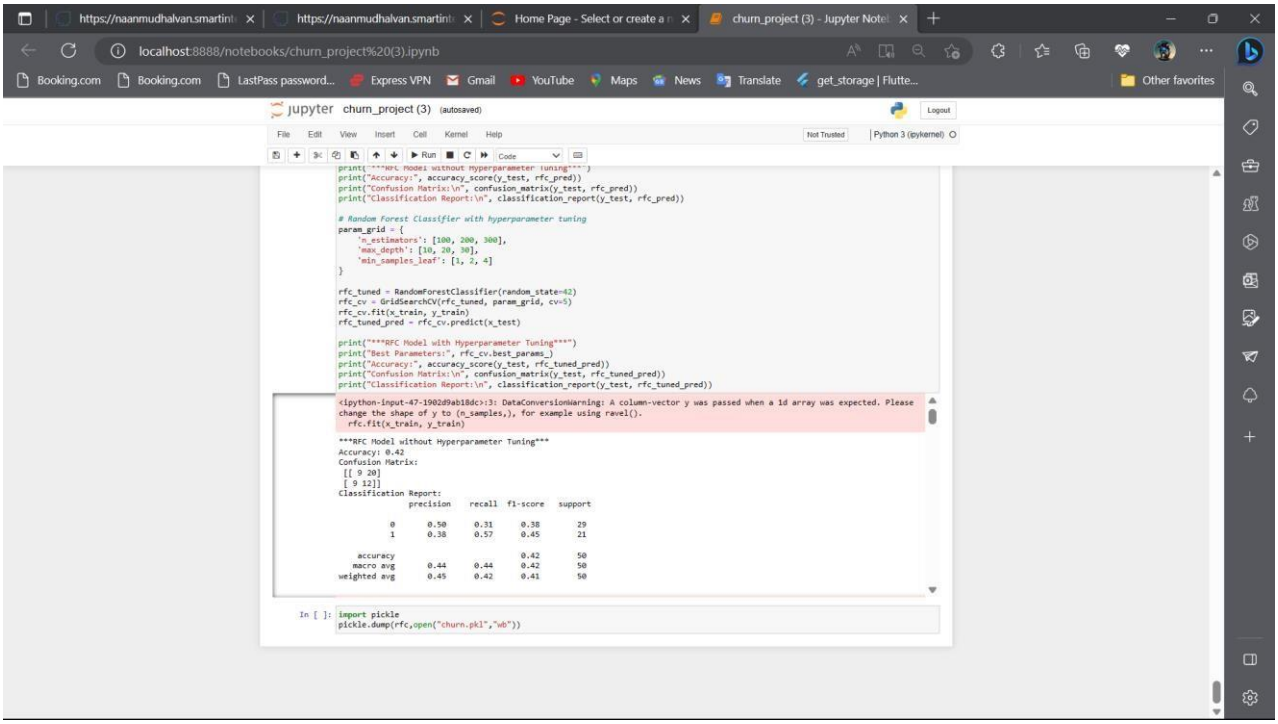

https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Note
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Predicted output is: [0]
/usr/local/lib/python3.9/dist-packages/sklearn/utils/validation.py:1143: DataConversionWarning: A column-vector y was passed wh
en 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([[0,0,1,1,0,0,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,456,1,0,3245,4567]])
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 0.50 0.57 28
accuracy 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)


https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
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) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
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
rf_pred = rf_classifier.predict(x_test_scaled)
print("***Random Forest Classifier Model***")
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-eeed9d587350>:6: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please cha
nge 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 11]]
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)

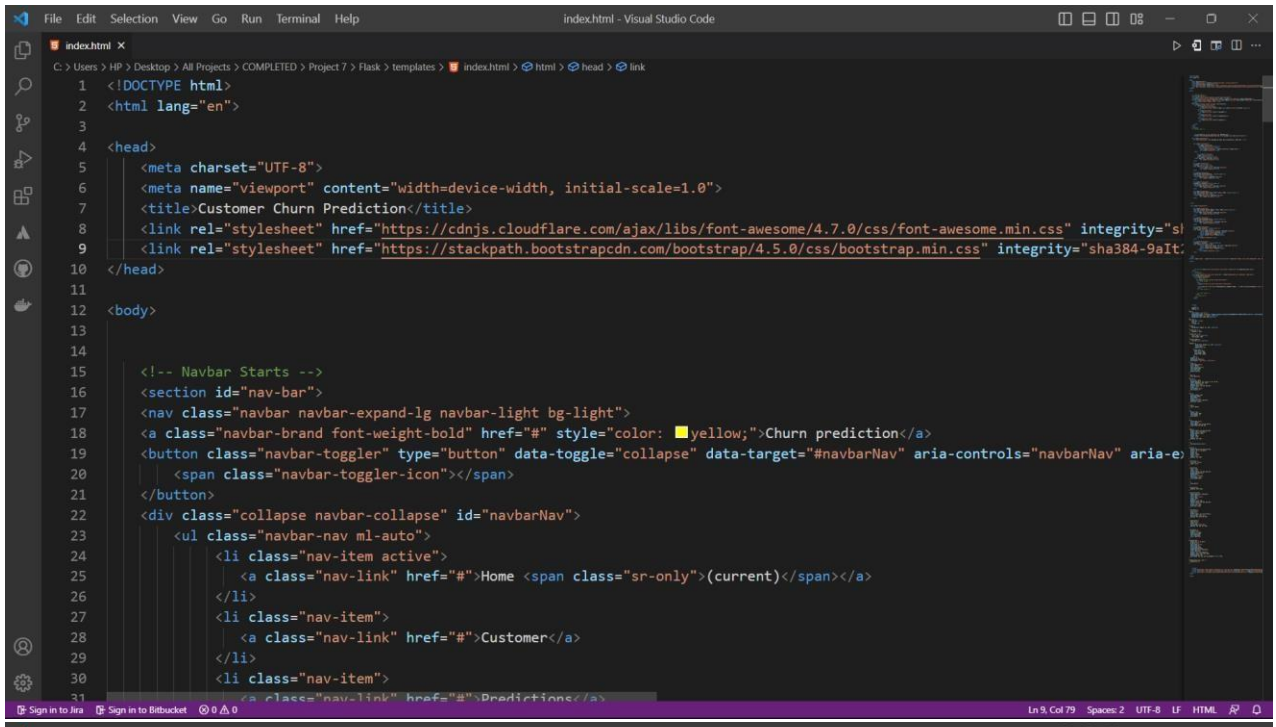

https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
churn_project (3) - Jupyter Note
localhost:8888/notebooks/churn_project%20(3).ipynb
Booking.com
Booking.com
LastPass password...
Express VPN
Gmail
YouTube
Maps
News
Translate
get_storage | Flutte...
Other favorites
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
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("***SVM Classifier Model***")
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 wh
en 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)


https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
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) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
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 wh
en 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 0.36 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 p
assed 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))
https://naanmudhalvan.smartint
https://naanmudhalvan.smartint
Home Page - Select or create a n
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) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
accuracy 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 p
assed 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 = (ann_pred>0.5)
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>0.5)
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


churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Epoch 6/200
10/10 [==============================] - 0s 8ms/step - 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-47-1902d9ab18dc>: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 0.42 50
macro avg 0.44 0.44 0.42 50
weighted avg 0.45 0.42 0.41 50


churn_project (3) - Jupyter Notebook
localhost:8888/notebooks/churn_project%20(3).ipynb
jupyter churn_project (3) (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Not Trusted
Python 3 (ipykernel)
Run
Code
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-47-1902d9ab18dc>: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 0.42 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

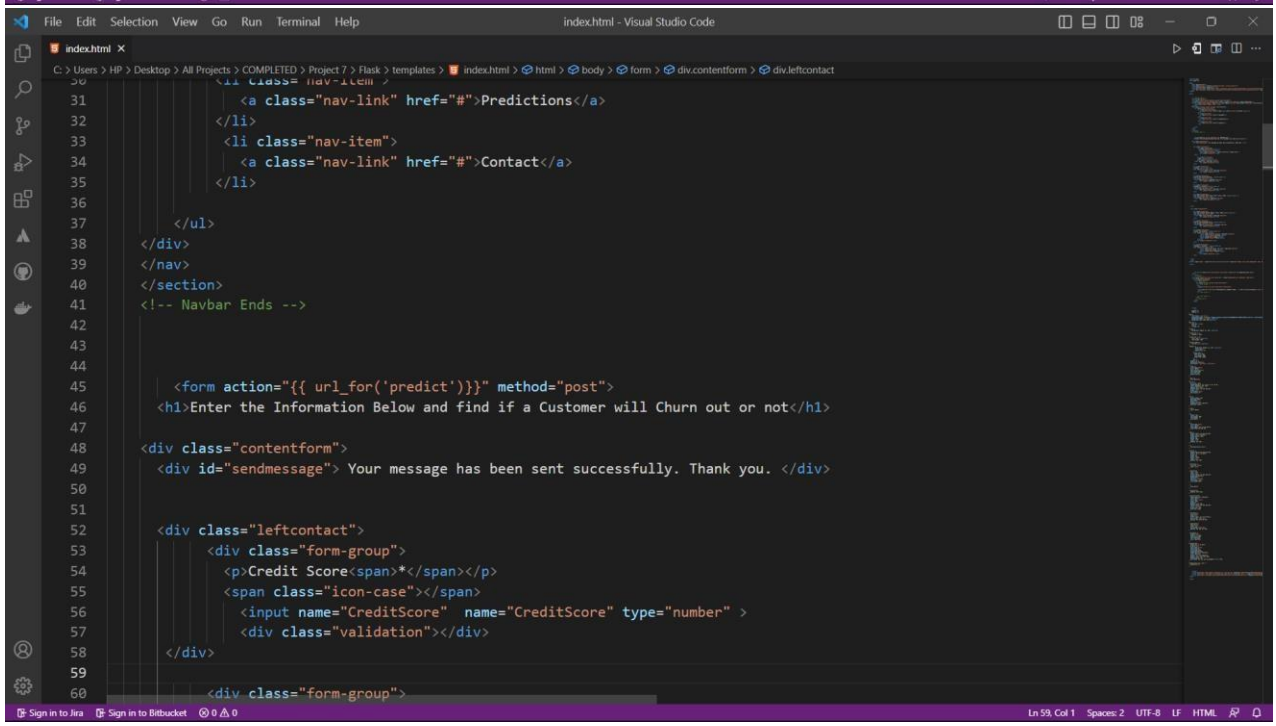

File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > head > link
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Customer Churn Prediction</title>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" integrity="sh
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-9aIt2
</head>
<body>
<!-- Navbar Starts -->
<section id="nav-bar">
<nav class="navbar navbar-expand-lg navbar-light bg-light">
Churn prediction
<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-e

</button>

<ul class="navbar-nav ml-auto">
<li class="nav-item active">
Home (current)

<li class="nav-item">
Customer

<li class="nav-item">
Sign in to Jira Sign in to Bitbucket 0 0
Ln 9, Col 79 Spaces: 2 UTF-8 LF HTML


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > form > div.contentform > div.leftcontact
Predictions

<li class="nav-item">
Contact

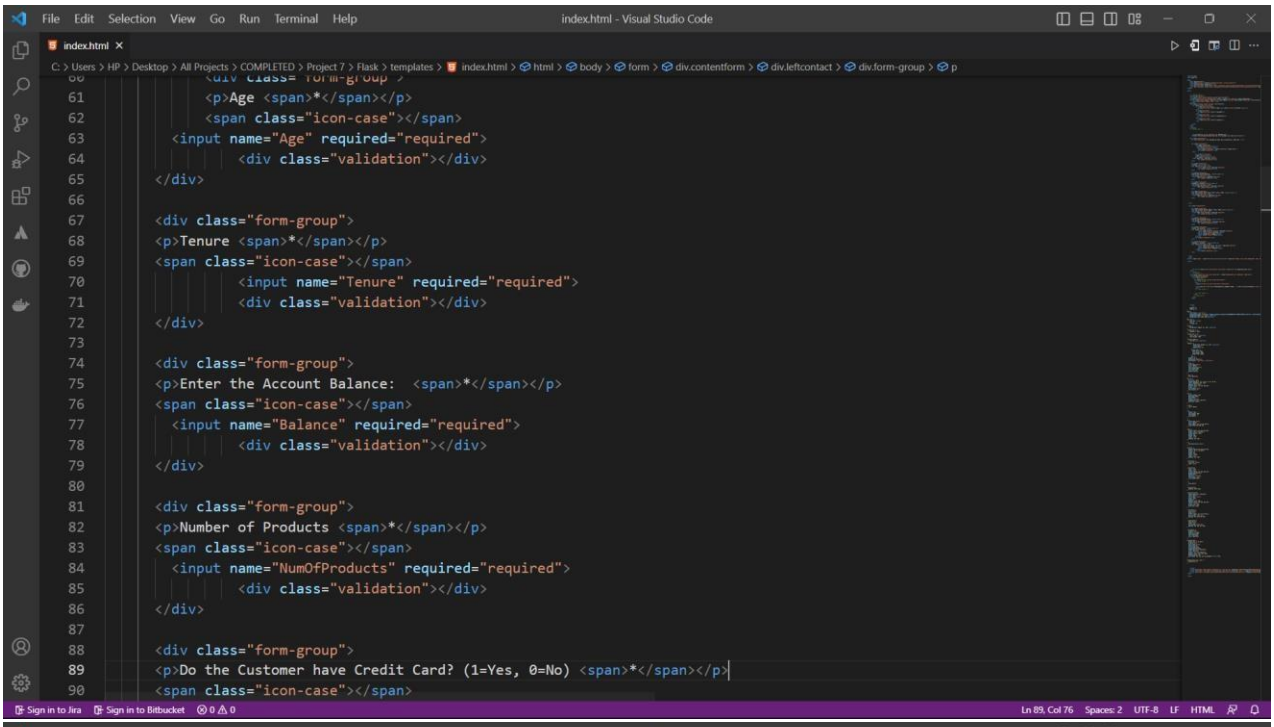
</nav>
</section>
<!-- Navbar Ends -->
<form action="{{ url_for('predict')}}" method="post">
<h1>Enter the Information Below and find if a Customer will Churn out or not</h1>

 Your message has been sent successfully. Thank you.

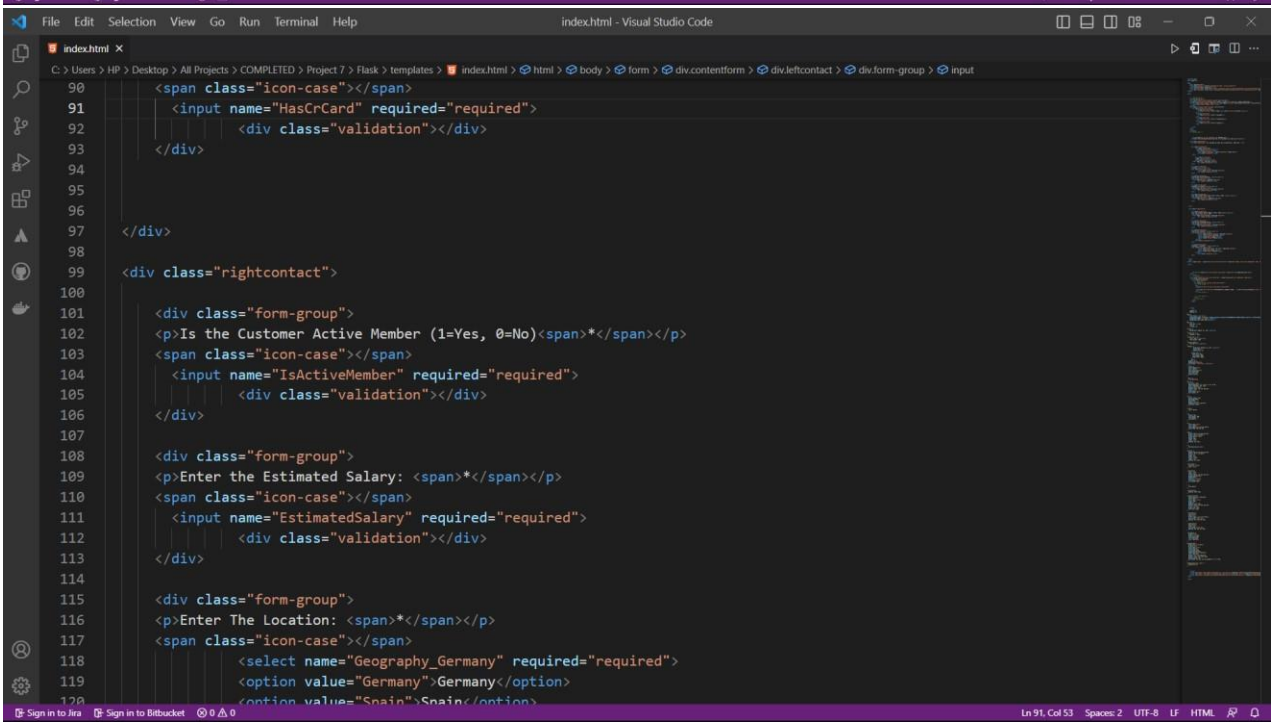
Credit Score*

<input name="CreditScore" name="CreditScore" type="number" >

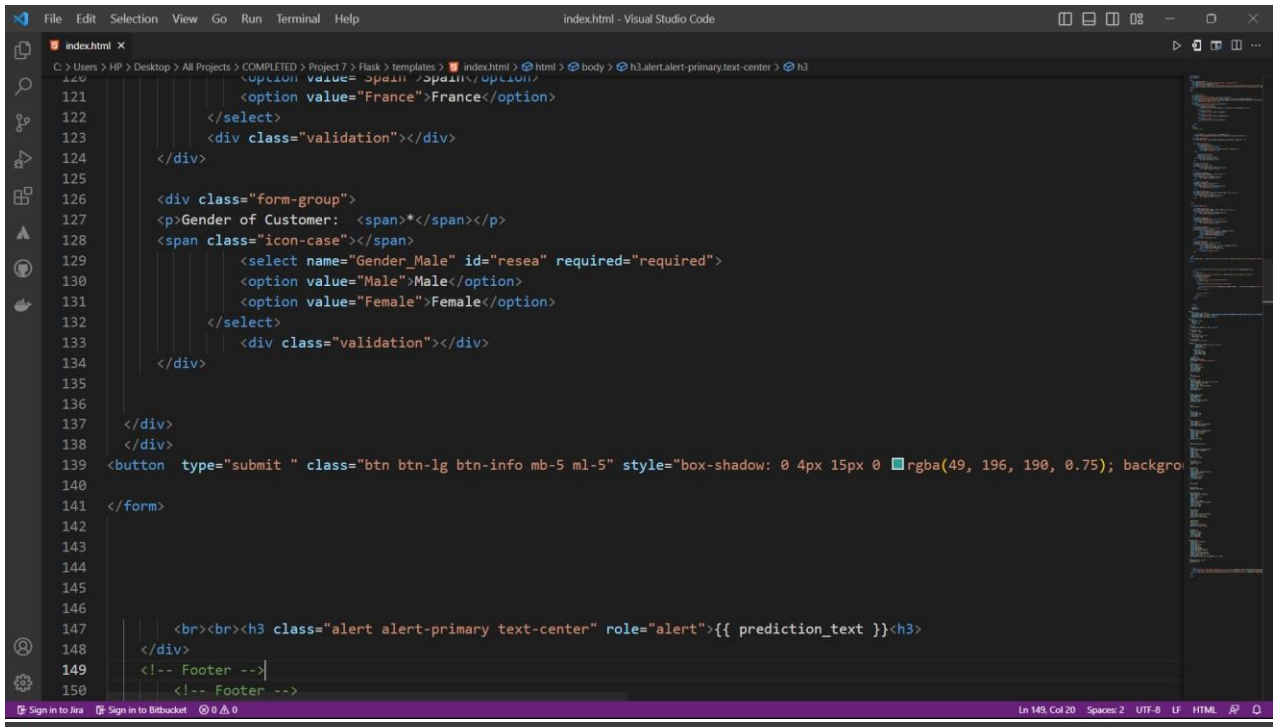
Sign in to Jira Sign in to Bitbucket 0 0
Ln 59, Col 1 Spaces: 2 UTF-8 LF HTML



```
<div class="form-group">
<p>Age <span>*</span></p>
<span class="icon-case"></span>
<input name="Age" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Tenure <span>*</span></p>
<span class="icon-case"></span>
<input name="Tenure" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Enter the Account Balance:  <span>*</span></p>
<span class="icon-case"></span>
<input name="Balance" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Number of Products <span>*</span></p>
<span class="icon-case"></span>
<input name="NumOfProducts" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Do the Customer have Credit Card? (1=Yes, 0=No) <span>*</span></p>
<span class="icon-case"></span>
```



```
<span class="icon-case"></span>
<input name="HasCrCard" required="required">
<div class="validation"></div>
</div>



</div>

<div class="rightcontact">

<div class="form-group">
<p>Is the Customer Active Member (1=Yes, 0=No)<span>*</span></p>
<span class="icon-case"></span>
<input name="IsActiveMember" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Enter the Estimated Salary: <span>*</span></p>
<span class="icon-case"></span>
<input name="EstimatedSalary" required="required">
<div class="validation"></div>
</div>

<div class="form-group">
<p>Enter The Location: <span>*</span></p>
<span class="icon-case"></span>
<select name="Geography_Germany" required="required">
<option value="Germany">Germany</option>
<option value="Spain">Spain</option>
```


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3
<option value="France">France</option>
</select>

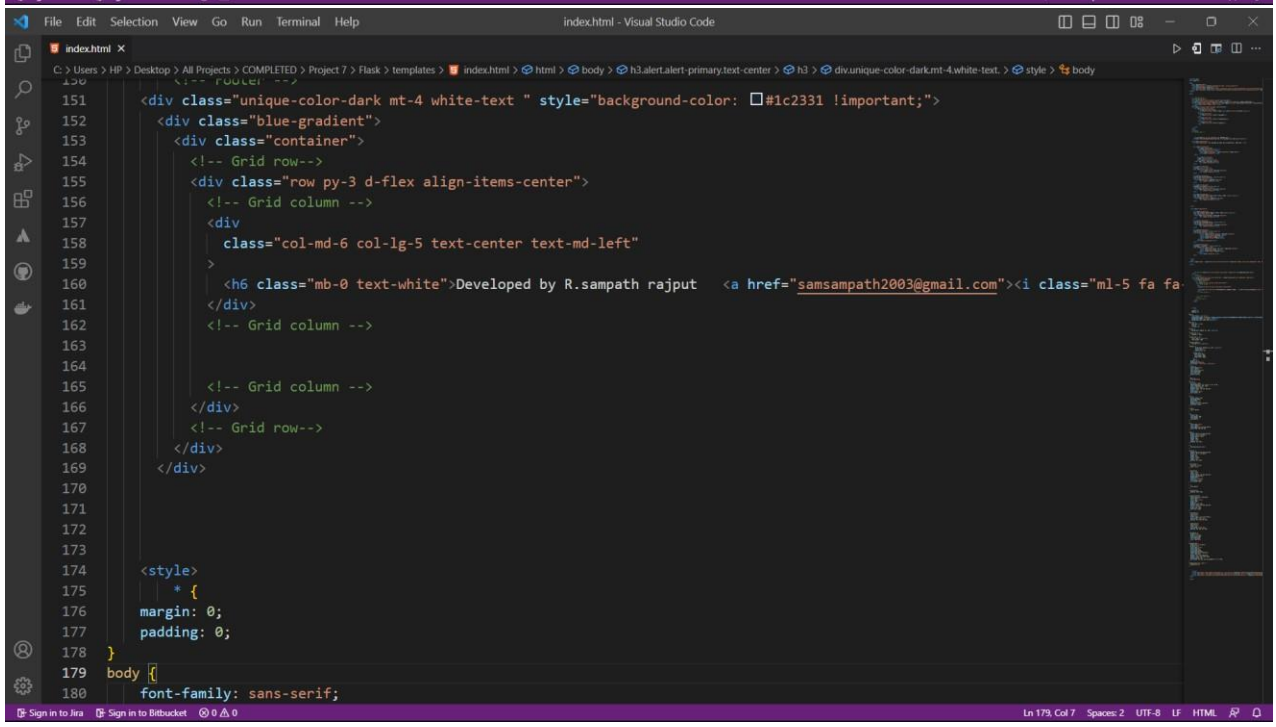
Gender of Customer: *

<select name="Gender_Male" id="resea" required="required">
<option value="Male">Male</option>
<option value="Female">Female</option>
</select>

<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); backgro
</form>

<h3 class="alert alert-primary text-center" role="alert">{{ prediction_text }}<h3>

<!-- Footer -->
<!-- Footer -->
Sign in to Jira
Sign in to Bitbucket
Ln 149, Col 20
Spaces: 2
UTF-8
LF
HTML


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
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

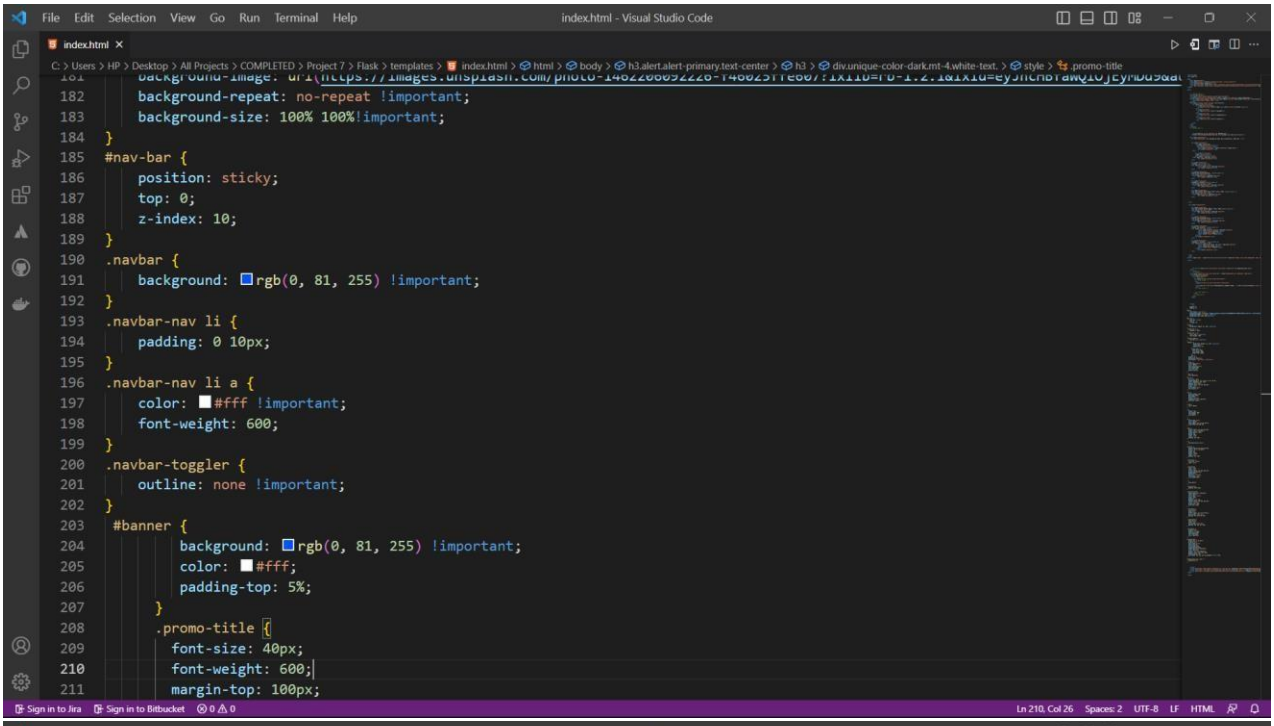
<!-- Grid row-->

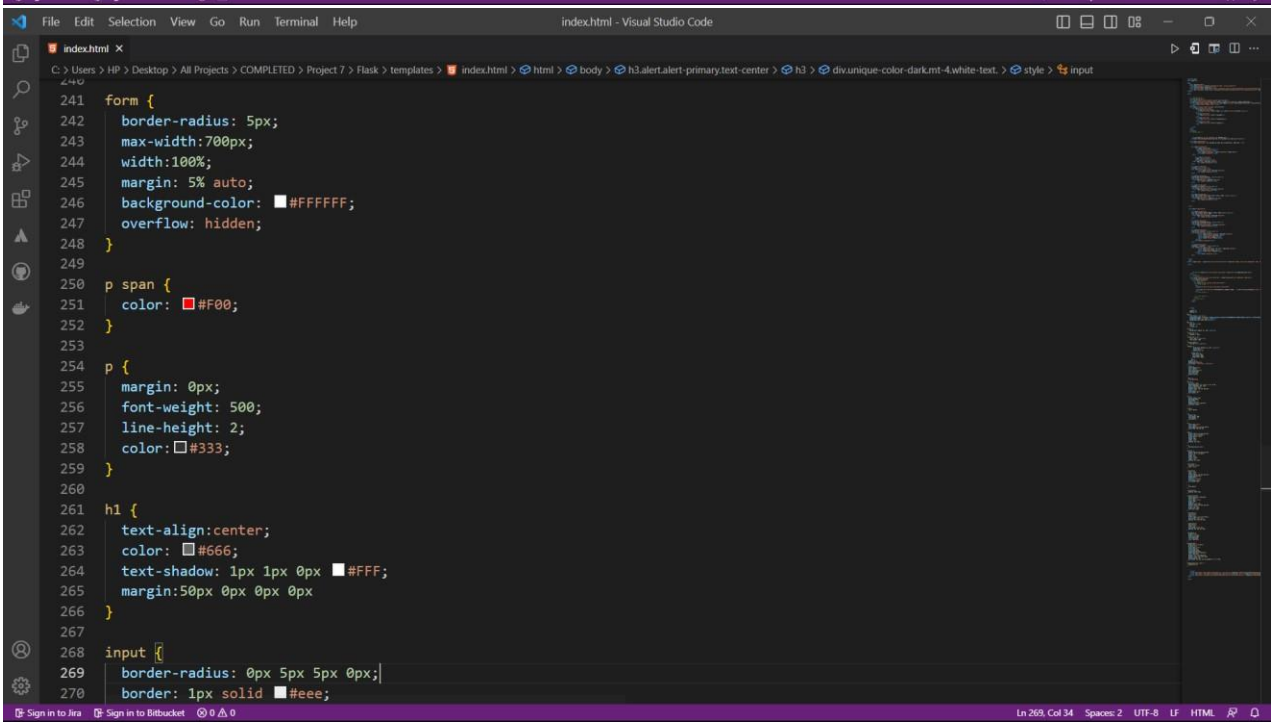
<!-- Grid column -->
<div
class="col-md-6 col-lg-5 text-center text-md-left"
>
<h6 class="mb-0 text-white">Developed by R.sampath rajput <a href="samsampath2003@gmail.com"><i class="ml-5 fa fa

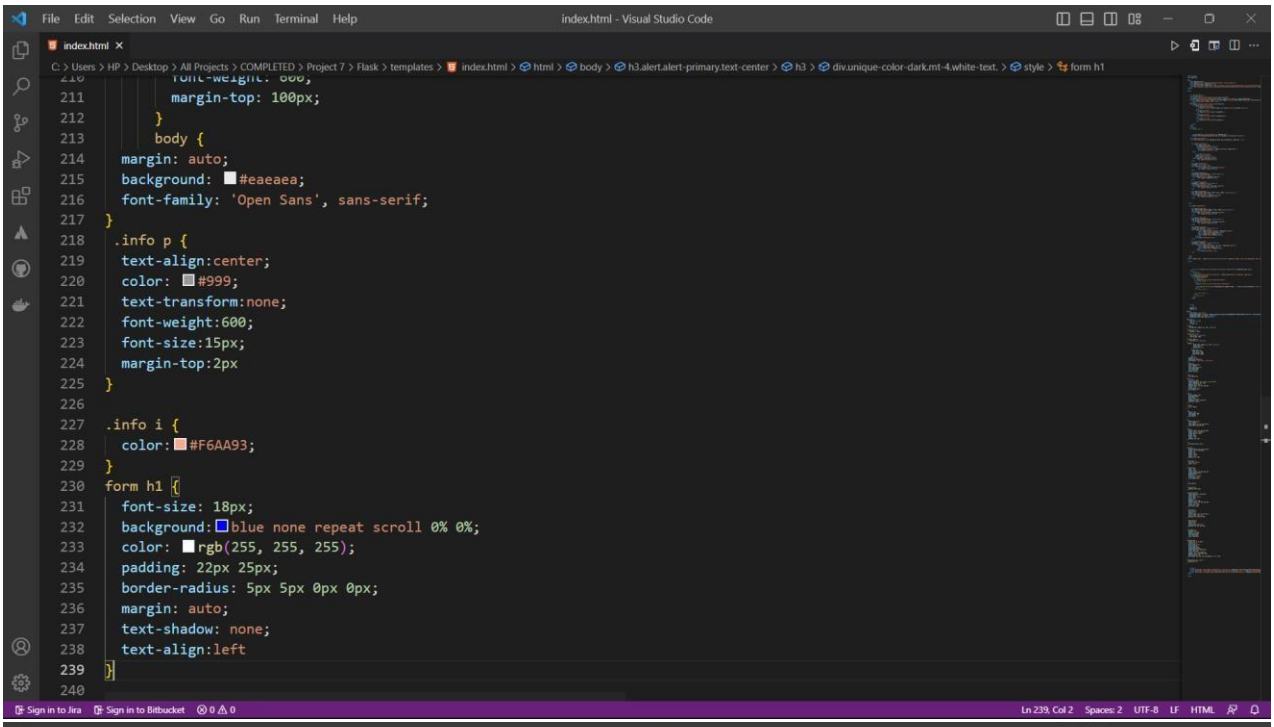
<!-- Grid column -->
<!-- Grid column -->

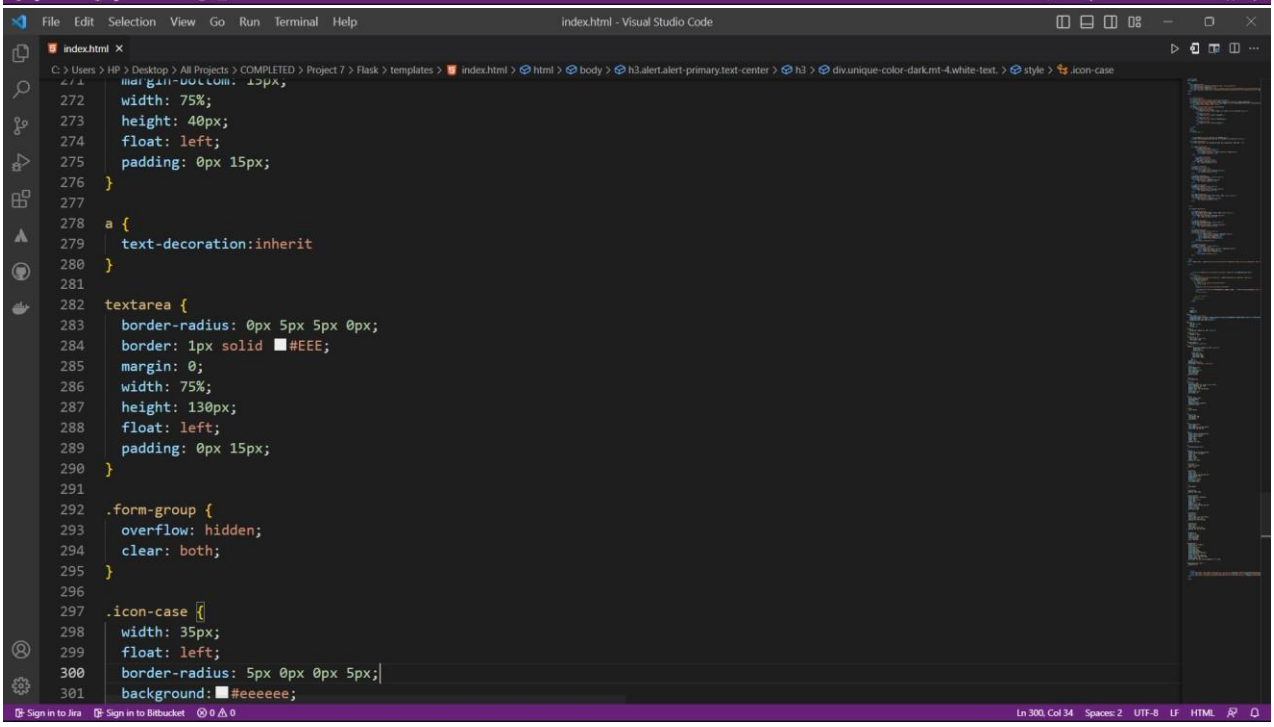
<!-- Grid row-->

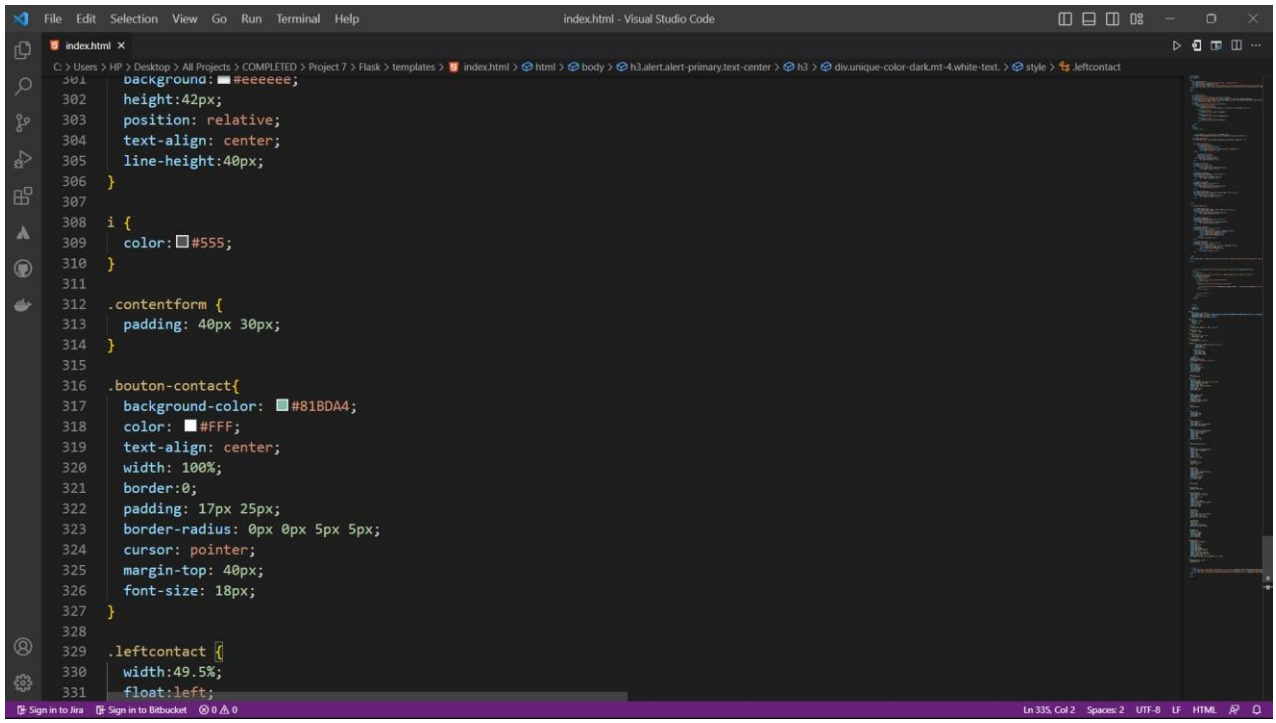
<style>
* {
margin: 0;
padding: 0;
}
body {
font-family: sans-serif;
Sign in to Jira
Sign in to Bitbucket
Ln 179, Col 7
Spaces: 2
UTF-8
LF
HTML

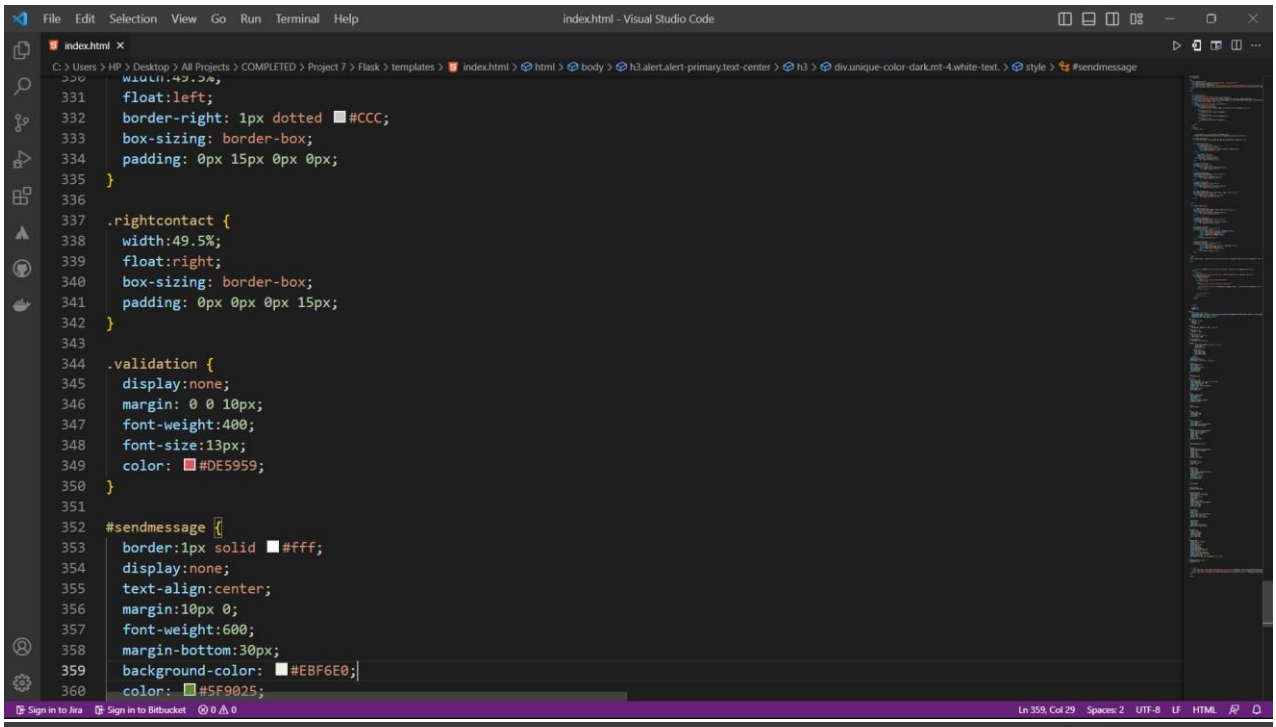

File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text. > style > .promo-title
background-repeat: no-repeat !important;
background-size: 100% 100%!important;
}
#nav-bar {
position: sticky;
top: 0;
z-index: 10;
}
.navbar {
background: rgb(0, 81, 255) !important;
}
.navbar-nav li {
padding: 0 10px;
}
.navbar-nav li a {
color: #fff !important;
font-weight: 600;
}
.navbar-toggler {
outline: none !important;
}
#banner {
background: rgb(0, 81, 255) !important;
color: #fff;
padding-top: 5%;
}
.promo-title {
font-size: 40px;
font-weight: 600;
margin-top: 100px;
Sign in to Jira
Sign in to Bitbucket
Ln 210, Col 26
Spaces: 2
UTF-8
LF
HTML

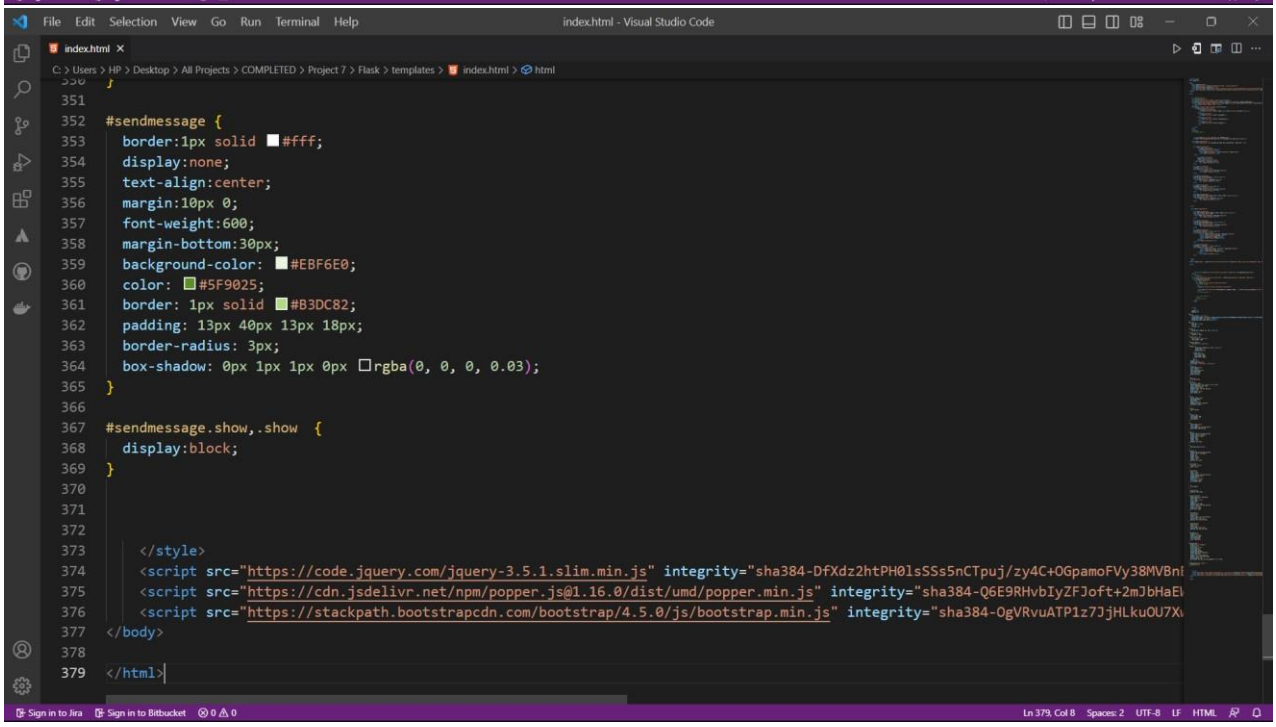

File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text. > style > input
form {
border-radius: 5px;
max-width:700px;
width:100%;
margin: 5% auto;
background-color: #FFFFFF;
overflow: hidden;
}
p span {
color: #F00;
}
p {
margin: 0px;
font-weight: 500;
line-height: 2;
color:#333;
}
h1 {
text-align:center;
color: #666;
text-shadow: 1px 1px 0px #FFF;
margin:50px 0px 0px 0px
}
input {
border-radius: 0px 5px 5px 0px;
border: 1px solid #eee;
Sign in to Jira
Sign in to Bitbucket
Ln 269, Col 34
Spaces: 2
UTF-8
LF
HTML


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
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 > form h1
font-weight: 600;
margin-top: 100px;
}
body {
margin: auto;
background: #eaeaea;
font-family: 'Open Sans', sans-serif;
}
.info p {
text-align:center;
color: #999;
text-transform:none;
font-weight:600;
font-size:15px;
margin-top:2px
}
.info i {
color:#F6AA93;
}
form h1 {
font-size: 18px;
background:blue none repeat scroll 0% 0%;
color: rgb(255, 255, 255);
padding: 22px 25px;
border-radius: 5px 5px 0px 0px;
margin: auto;
text-shadow: none;
text-align:left
}
Sign in to Jira
Sign in to Bitbucket
Ln 239, Col 2
Spaces: 2
UTF-8
LF
HTML

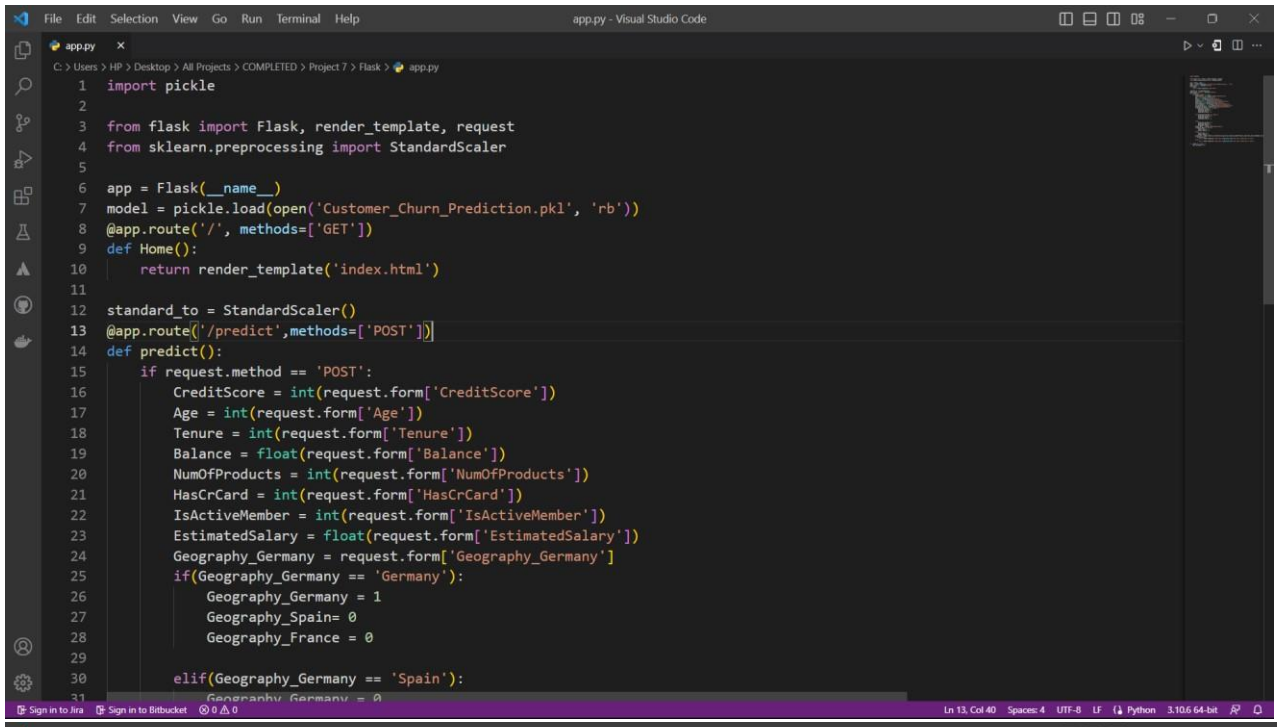

File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text. > style > .icon-case
margin-bottom: 15px;
width: 75%;
height: 40px;
float: left;
padding: 0px 15px;
}
a {
text-decoration:inherit
}
textarea {
border-radius: 0px 5px 5px 0px;
border: 1px solid #EEE;
margin: 0;
width: 75%;
height: 130px;
float: left;
padding: 0px 15px;
}
.form-group {
overflow: hidden;
clear: both;
}
.icon-case {
width: 35px;
float: left;
border-radius: 5px 0px 0px 5px;
background:#eeeeee;
Sign in to Jira
Sign in to Bitbucket
Ln 300, Col 34
Spaces: 2
UTF-8
LF
HTML


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
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 > .leftcontact
background: #eeeeee;
height:42px;
position: relative;
text-align: center;
line-height:40px;
}
i {
color: #555;
}
.contentform {
padding: 40px 30px;
}
.bouton-contact{
background-color: #81BDA4;
color: #FFF;
text-align: center;
width: 100%;
border:0;
padding: 17px 25px;
border-radius: 0px 0px 5px 5px;
cursor: pointer;
margin-top: 40px;
font-size: 18px;
}
.leftcontact {
width:49.5%;
float:left;
Sign in to Jira
Sign in to Bitbucket
0 0
Ln 335, Col 2
Spaces: 2
UTF-8
LF
HTML

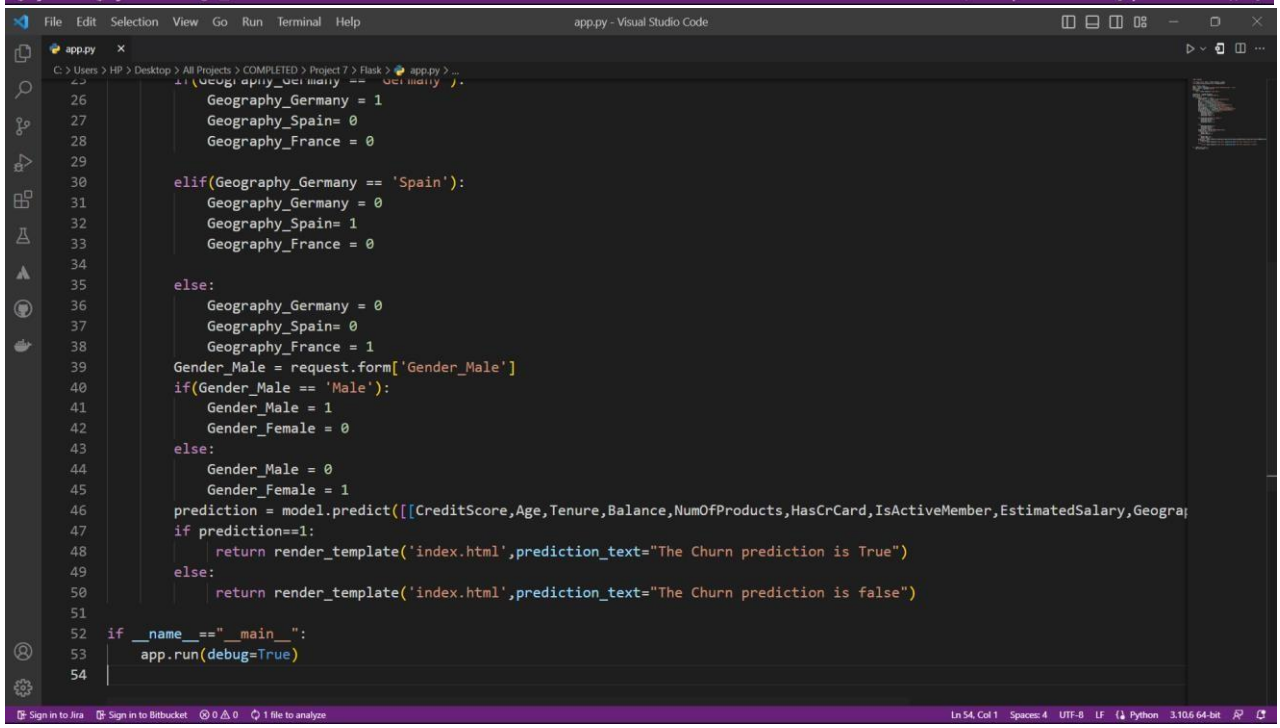

File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html > body > h3.alert.alert-primary.text-center > h3 > div.unique-color-dark.mt-4.white-text. > style > #sendmessage
width:49.5%;
float:left;
border-right: 1px dotted #CCC;
box-sizing: border-box;
padding: 0px 15px 0px 0px;
}
.rightcontact {
width:49.5%;
float:right;
box-sizing: border-box;
padding: 0px 0px 0px 15px;
}
.validation {
display:none;
margin: 0 0 10px;
font-weight:400;
font-size:13px;
color: #DE5959;
}
#sendmessage {
border:1px solid #fff;
display:none;
text-align:center;
margin:10px 0;
font-weight:600;
margin-bottom:30px;
background-color: #EBF6E0;
color: #5F9025;
Sign in to Jira
Sign in to Bitbucket
Ln 359, Col 29
Spaces: 2
UTF-8
LF
HTML


File Edit Selection View Go Run Terminal Help
index.html - Visual Studio Code
index.html
C: > Users > HP > Desktop > All Projects > COMPLETED > Project 7 > Flask > templates > index.html > html
}
#sendmessage {
border:1px solid #fff;
display:none;
text-align:center;
margin:10px 0;
font-weight:600;
margin-bottom:30px;
background-color: #EBF6E0;
color: #5F9025;
border: 1px solid #B3DC82;
padding: 13px 40px 13px 18px;
border-radius: 3px;
box-shadow: 0px 1px 1px 0px rgba(0, 0, 0, 0.03);
}
#sendmessage.show,.show {
display:block;
}
</style>
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBn
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaE
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7X
</body>
</html>
Sign in to Jira
Sign in to Bitbucket
Ln 379, Col 8
Spaces: 2
UTF-8
LF
HTML

App.py



```python
import pickle

from flask import Flask, render_template, request
from sklearn.preprocessing import StandardScaler

app = Flask(__name__)
model = pickle.load(open('Customer_Churn_Prediction.pkl', 'rb'))
@app.route('/', methods=['GET'])
def Home():
    return render_template('index.html')

standard_to = StandardScaler()
@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        CreditScore = int(request.form['CreditScore'])
        Age = int(request.form['Age'])
        Tenure = int(request.form['Tenure'])
        Balance = float(request.form['Balance'])
        NumOfProducts = int(request.form['NumOfProducts'])
        HasCrCard = int(request.form['HasCrCard'])
        IsActiveMember = int(request.form['IsActiveMember'])
        EstimatedSalary = float(request.form['EstimatedSalary'])
        Geography_Germany = request.form['Geography_Germany']
        if(Geography_Germany == 'Germany'):
            Geography_Germany = 1
            Geography_Spain= 0
            Geography_France = 0

        elif(Geography_Germany == 'Spain'):
```



```python
            Geography_Germany = 1
            Geography_Spain= 0
            Geography_France = 0

        elif(Geography_Germany == 'Spain'):
            Geography_Germany = 0
            Geography_Spain= 1
            Geography_France = 0

        else:
            Geography_Germany = 0
            Geography_Spain= 0
            Geography_France = 1
        Gender_Male = request.form['Gender_Male']
        if(Gender_Male == 'Male'):
            Gender_Male = 1
            Gender_Female = 0
        else:
            Gender_Male = 0
            Gender_Female = 1
        prediction = model.predict([[CreditScore,Age,Tenure,Balance,NumOfProducts,HasCrCard,IsActiveMember,EstimatedSalary,Geogra
        if prediction==1:
            return render_template('index.html',prediction_text="The Churn prediction is True")
        else:
            return render_template('index.html',prediction_text="The Churn prediction is false")

if __name__=="__main__":
    app.run(debug=True)
```