Travel Package Purchase Prediction

Contents –

By :- Vaibhav Sharma

Let's get started

1. Introduction to our project

"Travel Package Purchase Prediction" –

As the name suggests we'll build a model that predicts the probability of purchasing the travelling package.

If talking about importance of it. It is quite a bit interesting scenario – Suppose a travels and tours company wants to know their sales, deals are working or not. They do some analysis like

1. Age,
2. Monthly income,
3. Number of trips they want,
4. Are they male ,female or both,
5. Their job roles

   And many many aspects…….

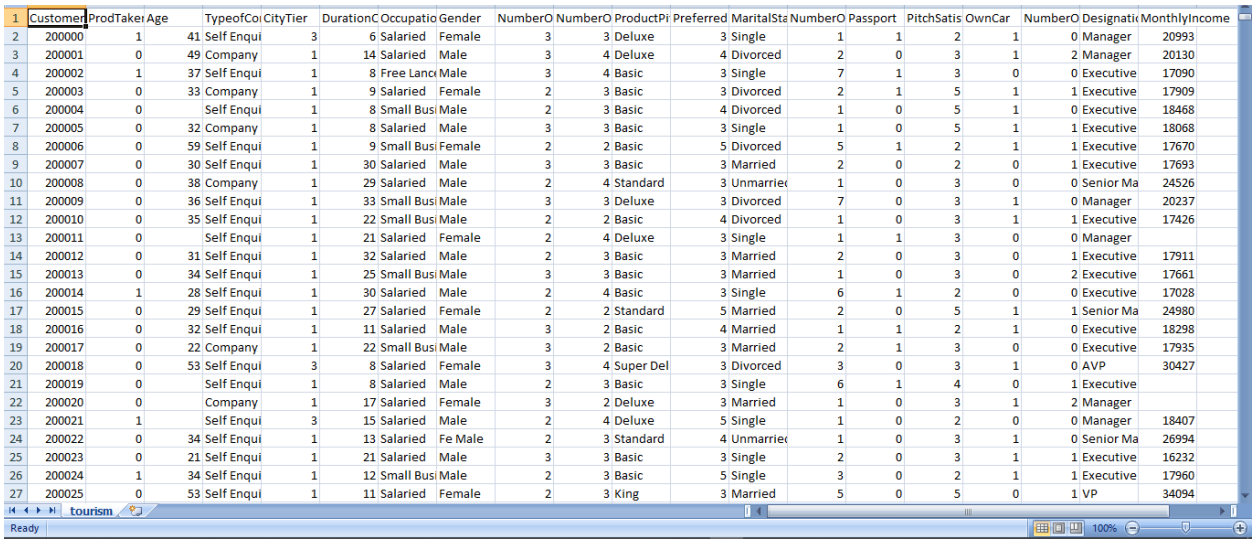
In this project , we'll do this analysis by my machine learning knowledge.

We'll perform every aspect of the data and explore it.

2. How to think to start

We have our dataset and it contains a lot features.

Have a look at our dataset



	CustomerI	ProdTaker	Age	TypeofCo	CityTier	DurationC	Occupatio	Gender	NumberO	NumberO	ProductPi	Preferred	MaritalSta	NumberO	Passport	PitchSatis	OwnCar	NumberO	Designati	MonthlyIncome
2	200000	1	41	Self Enqui	3	6	Salaried	Female	3	3	Deluxe	3	Single	1	1	2	1	0	Manager	20993
3	200001	0	49	Company	1	14	Salaried	Male	3	4	Deluxe	4	Divorced	2	0	3	1	2	Manager	20130
4	200002	1	37	Self Enqui	1	8	Free Lanc	Male	3	4	Basic	3	Single	7	1	3	0	0	Executive	17090
5	200003	0	33	Company	1	9	Salaried	Female	2	3	Basic	3	Divorced	2	1	5	1	1	Executive	17909
6	200004	0		Self Enqui	1	8	Small Busi	Male	2	3	Basic	4	Divorced	1	0	5	1	0	Executive	18468
7	200005	0	32	Company	1	8	Salaried	Male	3	3	Basic	3	Single	1	0	5	1	1	Executive	18068
8	200006	0	59	Self Enqui	1	9	Small Busi	Female	2	2	Basic	5	Divorced	5	1	2	1	1	Executive	17670
9	200007	0	30	Self Enqui	1	30	Salaried	Male	3	3	Basic	3	Married	2	0	2	0	1	Executive	17693
10	200008	0	38	Company	1	29	Salaried	Male	2	4	Standard	3	Unmarrie	1	0	3	0	0	Senior Ma	24526
11	200009	0	36	Self Enqui	1	33	Small Busi	Male	3	3	Deluxe	3	Divorced	7	0	3	1	0	Manager	20237
12	200010	0	35	Self Enqui	1	22	Small Busi	Male	2	2	Basic	4	Divorced	1	0	3	1	1	Executive	17426
13	200011	0		Self Enqui	1	21	Salaried	Female	2	4	Deluxe	3	Single	1	1	3	0	0	Manager	
14	200012	0	31	Self Enqui	1	32	Salaried	Male	2	3	Basic	3	Married	2	0	3	0	1	Executive	17911
15	200013	0	34	Self Enqui	1	25	Small Busi	Male	3	3	Basic	3	Married	1	0	3	0	2	Executive	17661
16	200014	1	28	Self Enqui	1	30	Salaried	Male	2	4	Basic	3	Single	6	1	2	0	0	Executive	17028
17	200015	0	29	Self Enqui	1	27	Salaried	Female	2	2	Standard	5	Married	2	0	5	1	1	Senior Ma	24980
18	200016	0	32	Self Enqui	1	11	Salaried	Male	3	2	Basic	4	Married	1	1	2	1	0	Executive	18298
19	200017	0	22	Company	1	22	Small Busi	Male	3	2	Basic	3	Married	2	1	3	0	0	Executive	17935
20	200018	0	53	Self Enqui	3	8	Salaried	Female	3	4	Super Del	3	Divorced	3	0	3	1	0	AVP	30427
21	200019	0		Self Enqui	1	8	Salaried	Male	2	3	Basic	3	Single	6	1	4	0	1	Executive	
22	200020	0		Company	1	17	Salaried	Female	3	2	Deluxe	3	Married	1	0	3	1	2	Manager	
23	200021	1		Self Enqui	3	15	Salaried	Male	2	4	Deluxe	5	Single	1	0	2	0	0	Manager	18407
24	200022	0	34	Self Enqui	1	13	Salaried	Fe Male	2	3	Standard	4	Unmarrie	1	0	3	1	0	Senior Ma	26994
25	200023	0	21	Self Enqui	1	21	Salaried	Male	3	3	Basic	3	Single	2	0	3	1	1	Executive	16232
26	200024	1	34	Self Enqui	1	12	Small Busi	Male	2	3	Basic	5	Single	3	0	2	1	1	Executive	17960
27	200025	0	53	Self Enqui	1	11	Salaried	Female	2	3	King	3	Married	5	0	5	0	1	VP	34094

tourism

Ready

100%

It has many features like we talked above about it

Features (Column)	Definition
CustomerID	Unique customer ID
ProdTaken	Product taken or not
Age	Age of customer
PreferredLoginDevice	Preferred login device of customer in last month
CityTier	City tier
DurationOfPitch	Duration of pitch by a sales man to customer
Occupation	Occupation of customer
Gender	Gender of customer
NumberOfPersonVisited	Total number of person came with customer
NumberOfFollowups	Total number of follow up has been done by sales person after sales pitch
ProductPitched	Product pitched by sales person
PreferredPropertyStar	Preferred hotel property rating by customer

MaritalStatus	Marital status of customer
NumberOfTrips	Average number of trip in a year by customer
Passport	Customer passport flag

PitchSatisfactionScore	Sales pitch satisfactory score
OwnCar	Customers owns a car or not
NumberOfChildrenVisited	Total number of children visit with customer
Designation	Designation of customer in current organization
MonthlyIncome	Gross monthly income of customer

You can get a brief description of every feature (column) using the table above.

3. How I Start

I'll start with cleaning the data, which we'll talk about in Data Preprocessing Section and then try to decorate my room I mean creating the model and deploy it.

4. Data Preprocessing

Steps we'll follow in data preprocessing

- Handling Missing Values
- Handling Categorical Values
- Checking the distributions of features
- Removing the outliers

- Handling Missing Values :-

	CustomerID	ProdTaken	Age	TypeofContact	CityTier
2	200000	1	41	Self Enquiry	3
3	200001	0	49	Company Invited	1
4	200002	1	37	Self Enquiry	1
5	200003	0	33	Company Invited	1
6	200004	0		Self Enquiry	1
7	200005	0	32	Company Invited	1
8	200006	0	59	Self Enquiry	1
9	200007	0	30	Self Enquiry	1
10	200008	0	38	Company Invited	1
11	200009	0	36	Self Enquiry	1
12	200010	0	35	Self Enquiry	1
13	200011	0		Self Enquiry	1
14	200012	0	31	Self Enquiry	1
15	200013	0	34	Self Enquiry	1
16	200014	1	28	Self Enquiry	1
17	200015	0	29	Self Enquiry	1
18	200016	0	32	Self Enquiry	1
19	200017	0	22	Company Invited	1
20	200018	0	53	Self Enquiry	3
21	200019	0		Self Enquiry	1

Data Dict | Tourism

Ready

Designation	MonthlyIncome
Manager	20993
Manager	20130
Executive	17090
Executive	17909
Executive	18468
Executive	18068
Executive	17670
Executive	17693
Senior Manager	24526
Manager	20237
Executive	17426
Manager	
Executive	17911
Executive	17661
Executive	17028
Senior Manager	24980
Executive	18298
Executive	17935
AVP	30427
Executive	

As we can above there are empty spaces in the columns i.e. Age, MonthlyIncome etc and we don't want our data to be distorted like this. We have many methods to remove it like

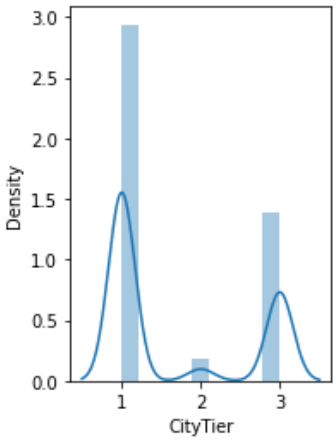
First find the columns which have missing values

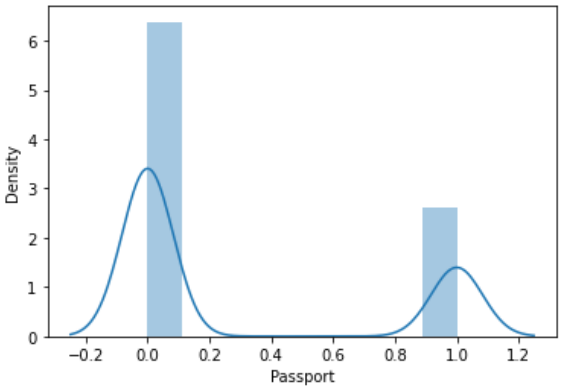
```
CustomerID                  0
ProdTaken                   0
Age                       226
TypeofContact              25
CityTier                    0
DurationOfPitch           251
Occupation                  0
Gender                      0
NumberOfPersonVisited       0
NumberOfFollowups          45
ProductPitched              0
PreferredPropertyStar      26
MaritalStatus               0
NumberOfTrips             140
Passport                    0
PitchSatisfactionScore      0
OwnCar                      0
NumberOfChildrenVisited    66
Designation                 0
MonthlyIncome             233
```

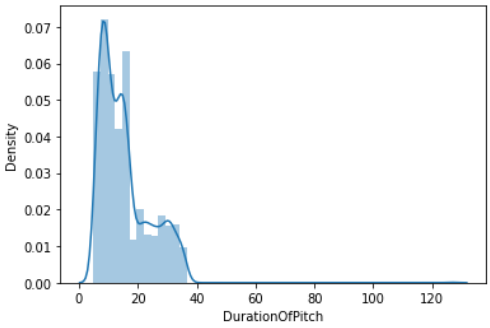
We have missing values in 8 columns

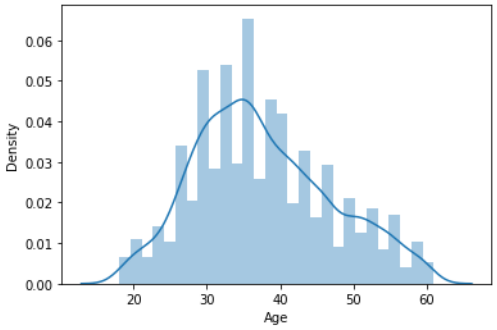
- Dropping the columns which have missing values.
- Dropping the rows which have missing values.
- Mean/Median replacement
- Random Sample replacement
- End of Distribution value replacement

Let's us see the distribution of some columns









As we can see many distributions are not Gaussian so replace it with median.

1. Random Sample replacement

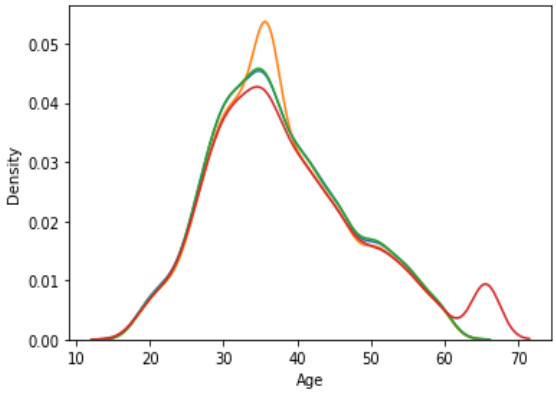
We'll take random values of our column which have missing values and replace it with our random samples.

2. End of distribution replacement

Now we get the value which is present after the 3rd standard deviation of distribution and replace it with missing values

Remember* - when we replacing the nan values with some other values, the distribution of that columns need not to be changed (less variance).

After imputing the missing values, check the distribution using kdeplot, we get better result with the method of random sample imputation.



Blue plot - original feature with missing values.

Orange plot - feature with median replacement.

Green plot - feature with random sample imputation.

Red plot –feature with end distribution imputation.

We can see that blue and green plot are almost coinciding means less change in variance.

- Handling categorical feature

Categorical features are those features which have values in form of string like our features below

```
['TypeofContact',
 'Occupation',
 'Gender',
 'ProductPitched',
 'MaritalStatus',
 'Designation']
```

We can replace the missing values of categorical feature (column) with the most frequent value.

This would be the good approach for imputing categorical features.

Now all machine learning algorithms/models does work well with categorical features and its not good approach to drop the columns as we discussed above.

So let's replace the value of categorical features with a number.

We are doing onehotencoding with categorical features using get_dummies function in pandas.

Let's have a look at one of the categorical feature after OneHotEncoding.

Income	TypeofContact_Self Enquiry	Occupation_Large Business	Occupation_Salaried	Occupation_Small Business	Gender_Male	ProductPitched_Deluxe	ProductPitched_King	ProductPitched_Standar
0993.0	1	0	1	0	0	1	0	
0130.0	0	0	1	0	1	1	0	
7090.0	1	0	0	0	1	0	0	
7909.0	0	0	1	0	0	0	0	
8468.0	1	0	0	1	1	0	0	

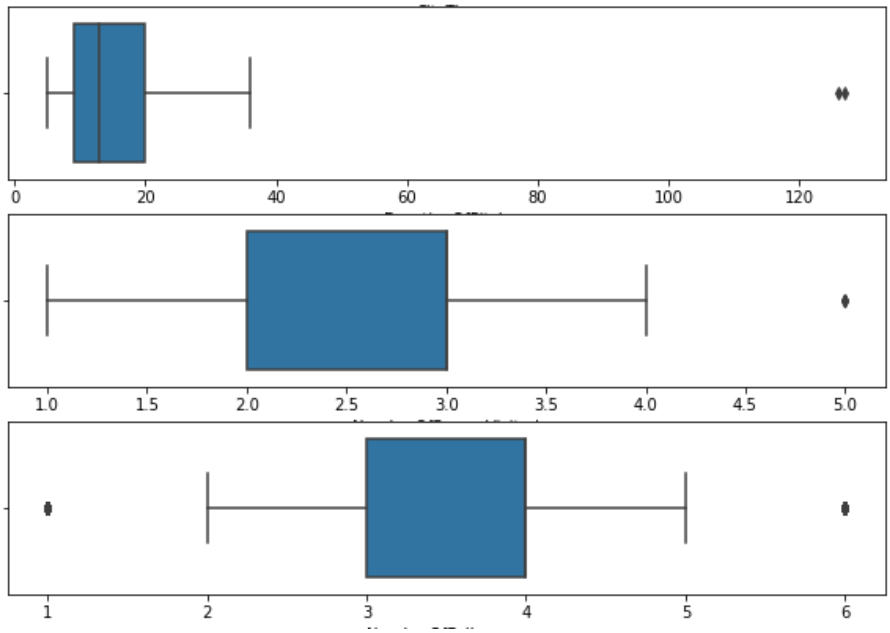
5. Feature Engineering

- Checks the distribution and treat the outliers

Outliers are those which are damaging our models and have an reverse relation with our model.

We can see the outliers using boxplot for every feature.

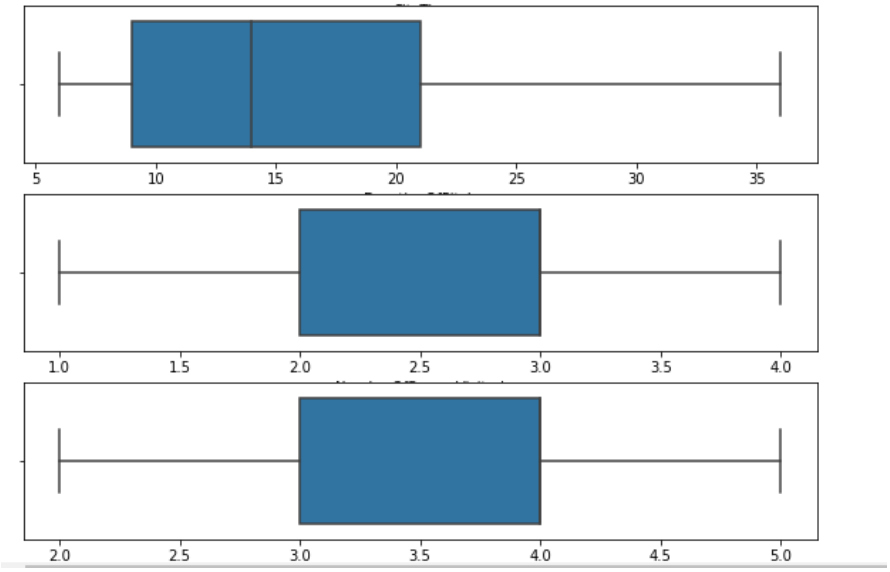
Know more about outliers.



We'll remove the outliers using InterQuantileRange method.

After removing outliers

Plots lookalike



6. Feature Selection

We'll find the relation using corr() method of pandas to find out the relation between each column.

We have two type of feature i.e. Dependent Feature and Independent Features.

Corr() gives a number which shows the strength of relation .

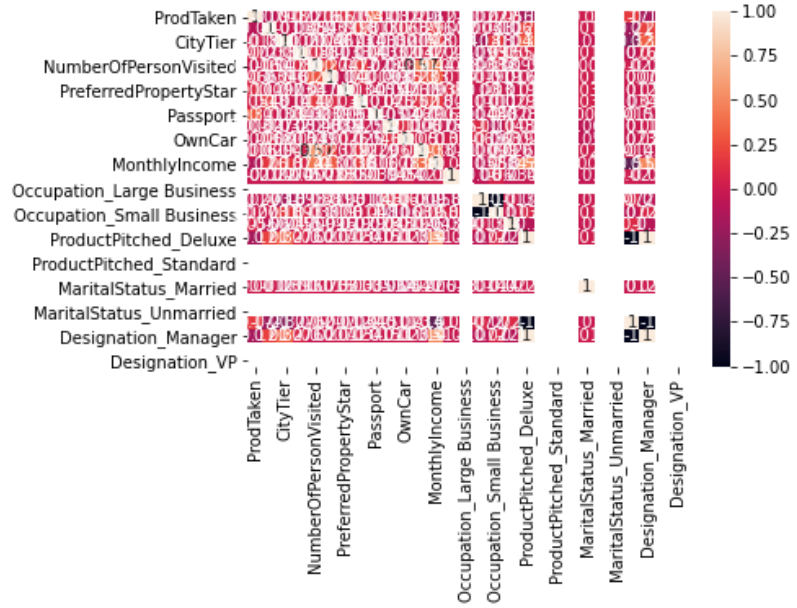
If we see a high relation between two independent features, we'll drop one of them.

If we see a high relation between independent feature and dependent feature, we'll keep the relation because it is important for our model.

We can see the relation by plotting it using heatmap.

```
sns.heatmap(df.corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4f8de97450>
```



I've used feature_selection library of sklearn module to find the top 10 essential features that we can keep.

We are using chi_2 method to find it.

Top 10 most essential features are:-

```
feat_imp #important features
```

```
['MonthlyIncome',
 'Passport',
 'DurationOfPitch',
 'Designation_Executive',
 'ProductPitched_Deluxe',
 'Designation_Manager',
 'Age',
 'NumberOfTrips',
 'PreferredPropertyStar',
 'Gender_Male']
```

7. Playing with different models and checks the accuracy

Now we build the model for prediction.

I've splitted the data in training data and testing data using train_test_split and classifies the model

- KNeighborsClassifier
- SVC
- RandomForestClassifier
- AdaBoostClassifier
- GaussianNB

I've build the models one by one and tune it also using hyperparameter tuning with GridSearchCV library.

1. Random Forest Classification

```
[ ] clf1 = RandomForestClassifier()
    param_rfc = {'n_estimators':[100,200,250,300,350,400],'criterion' : ["gini", "entropy"],'max_depth':[5,6,7,4]}
    rfc = GridSearchCV(clf1,param_rfc,scoring='precision')
    rfc.fit(x_train,y_train)
    print(rfc.best_params_)         #best parameters are :- {'criterion': 'entropy', 'max_depth': 7, 'n_estimators': 250}
    print(rfc.best_score_)          #best score by applying best parameter is :- 0.8893650793650794

    #accuracy_score(y_test,pred_rfc)
```

```
{'criterion': 'gini', 'max_depth': 7, 'n_estimators': 350}
0.8980745341614906
```

```
pred_rfc = rfc.predict(x_test)
confusion_matrix(y_test,pred_rfc)
```

```
array([[309,   0],
       [ 30,  24]])
```

2. Support Vector Classification

```
[ ] clf2 = SVC()
    param_svc = {'C':[0.1,1,10,100],'kernel':['linear','poly','rbf','sigmoid']}
    svc = GridSearchCV(clf2,param_svc,scoring = 'precision')
    svc.fit(x_train,y_train)
    print(svc.best_params_)         #best parameters are :- {'C': 0.1, 'kernel': 'linear'}
    print(svc.best_score_)          #best score by applying best parameter is :- 0.4

    #accuracy_score(y_test,pred_svc)
```

```
{'C': 0.1, 'kernel': 'linear'}
0.4
```

```
pred_svc = svc.predict(x_test)
confusion_matrix(y_test,pred_svc)
```

```
array([[309,   0],
       [ 54,   0]])
```

3. KNN Classification

```
[ ] clf3 = KNeighborsClassifier()
    param_knn = {'n_neighbors':list(range(1,6)),'algorithm':['auto','ball_tree','kd_tree','brute']}
    knn = GridSearchCV(clf3,param_knn,scoring = 'precision')
    knn.fit(x_train,y_train)
    print(knn.best_params_)      #best parameters are :- {'algorithm': 'auto', 'n_neighbors': 1}
    print(knn.best_score_)       #best score is :- 0.5028759398496241

    #accuracy_score(y_test,pred_knn)
```

```
{'algorithm': 'auto', 'n_neighbors': 1}
0.5028759398496241
```

```
pred_knn = knn.predict(x_test)
confusion_matrix(y_test,pred_knn)
```

```
array([[307,   2],
       [ 13,  41]])
```

4. Gaussian Naive Bayes Classification

```
[ ]
clf5 = GaussianNB()
param_nb = {'var_smoothing': np.logspace(0,-9, num=100)}
nb = GridSearchCV(clf5,param_nb,scoring = 'precision')
nb.fit(x_train,y_train)
print(nb.best_params_)          #best parameters are:- {'var_smoothing': 3.5111917342151277e-08}
print(nb.best_score_)           #best score is :- 0.7573856209150327

#accuracy_score(y_test,pred_nb)
```

```
{'var_smoothing': 3.5111917342151277e-08}
0.7573856209150327
```

```
pred_nb = nb.predict(x_test)
confusion_matrix(y_test,pred_nb)
```

```
array([[308,   1],
       [ 51,   3]])
```

5. AdaBoost Classification

```
clf4 = AdaBoostClassifier()
param_ada = {'n_estimators':list(range(50,101))}
ada = GridSearchCV(clf4,param_ada,scoring = 'precision')
ada.fit(x_train,y_train)
print(ada.best_params_)         #best parameters are :- {'n_estimators': 99}
print(ada.best_score_)          #best score is :- 0.5433766233766233

#accuracy_score(y_test,pred_ada)
```

```
{'n_estimators': 99}
0.5433766233766233
```

```
[ ] pred_ada = ada.predict(x_test)
confusion_matrix(y_test,pred_ada)
```

```
array([[299,  10],
       [ 42,  12]])
```

I've used accuracy and classification report for accuracy checking and Random Forest Classifier works best for our dataset.

So we'll use the Random Forest Classifier for our dataset.

8. Pickle it

Now we have to import our model to other source,editor,codebase etc. We cant take our whole file ,so for this we'll make a model file (extension of .pkl). For this I'm using joblib library.

```
[ ]  from sklearn.externals import joblib
```

```
joblib.dump(nb, 'model.pkl')

# Load the model from the file
pklmodel = joblib.load('model.pkl')

# Use the loaded model to make predictions
pklmodel.predict(x_test)
```

My model is saved with model.pkl file.

9. Build the GUI interface

Now to globalize the model and increase the user interaction with model, I need some frontend stuff.

Streamlit comes into picture.

Using streamlit I've build the frontend that consist

- Three sections:-
  1. Header :- This section consist my dataset so anybody can see what it contains and displaying list of features and some plots.
  2. Midsection :- This section includes the main part that is prediction of package purchasing. Some input boxes, checkboxes,selectboxes etc.
  3. Footer :- This section has something about me.

10. Deployment of our model

I've also deployed our model using streamlit sharing platform.

Having your files on GitHub with accurate requirements.txt is only you require for deploying on streamlit.

All library that are required for this project are:-

1. Pandas
2. Numpy
3. Matplotlib
4. Seaborn
5. Itertools
6. Sklearn
7. Joblib
8. Streamlit
9. Os
10. Pickle

You can get the whole source code on my GitHub profile.