Import Necessary Libraries In [75]: import pandas as pd import seaborn as sns import numpy as np import matplotlib.pyplot as plt from sklearn.linear\_model import LogisticRegression from sklearn.metrics import confusion\_matrix from sklearn.metrics import roc\_curve from sklearn.metrics import roc\_auc\_score **Business Problem** Whether the client has subscribed a term deposit or not Binomial ("yes" or "no") Data Collection for Performing Logistic Regression In [76]: bank\_data=pd.read\_csv('bank-full.csv', sep=';') bank\_data Out[76]: marital education default balance housing loan contact day month duration campaign pdays previous poutcome unknown 58 management married tertiary 2143 yes no unknown 5 may 261 1 -1 no 44 technician single secondary no 29 yes unknown 5 may 151 unknown no no entrepreneur 1 -1 **2** 33 married secondary 2 unknown 5 76 unknown no yes may no blue-collar unknown 1506 -1 unknown married no yes no unknown 5 may -1 33 198 1 4 unknown single 1 5 unknown unknown no no no unknown may no 3 45206 51 technician married tertiary 825 cellular 17 977 -1 unknown yes no no no nov 45207 71 divorced 1729 cellular 17 456 2 retired primary no no no nov unknown yes 45208 72 secondary 5715 cellular 17 1127 5 184 retired married no no no nov success yes 45209 secondary 668 -1 unknown blue-collar married no telephone 17 nov 508 no no no 37 entrepreneur 188 45210 secondary 2971 married no cellular 17 nov 361 2 11 other no 45211 rows × 17 columns Data Understanding by performing Initial Analysis bank\_data.shape (45211, 17)Out[77]: In [78] bank\_data.isna().sum() # No null values 0 Out[78]: job 0 marital 0 education default 0 balance 0 housing loan contact day month duration campaign 0 pdays previous poutcome dtype: int64 In [79]: bank\_data.dtypes int64 Out[79]: age object job marital object education object default object balance int64 housing object loan object contact object day int64 object month duration int64 campaign int64 int64 pdays int64 previous poutcome object object dtype: object In [80]: bank\_data.describe <bound method NDFrame.describe of</pre> job marital education default balance housing loan \ age management yes 58 married tertiary no 2143 no 44 technician single secondary no 29 yes no 33 entrepreneur married secondary no yes yes 3 47 blue-collar married unknown 1506 no yes no 33 unknown single unknown no 1 no no 45206 51 technician married tertiary 825 no no 45207 71 retired divorced primary 1729 no no no 45208 72 retired married secondary no 5715 no no 45209 57 blue-collar married secondary no 668 no no 45210 37 entrepreneur married secondary 2971 no no contact day month duration campaign pdays previous poutcome unknown 5 may 261 1 -1 unknown no unknown 5 may 151 -1 unknown 76 2 unknown 5 1 -1 0 unknown may no 3 unknown 5 may 92 1 -1 0 unknown no unknown 5 may 198 1 -1 unknown no 977 45206 cellular 17 3 0 nov -1 unknown yes unknown 45207 cellular 17 nov 456 2 -1 0 yes 45208 cellular 17 nov 1127 184 success yes 45209 telephone 17 nov 508 -1 0 unknown 45210 cellular 17 nov 361 188 11 other no [45211 rows x 17 columns]> In [81]: sns.pairplot(bank\_data) Out[81]: <seaborn.axisgrid.PairGrid at 0x110256940d0> 80 age 60 40 100000 80000 60000 40000 20000 30 25 20 g 15 10 5000 4000 3000 2000 1000 60 30 20 10 800 600 skepd 400 200 250 200 S 150 를 100 25000 50000 7500 **1**00000 600 200 4000 100 pdays duration previous bank\_data['y'].value\_counts() 39922 Out[82]: 5289 Name: y, dtype: int64 In [83]: count\_no\_sub=len(bank\_data[bank\_data['y']=='no']) count\_sub=len(bank\_data[bank\_data['y']=='yes']) (count\_sub/(count\_sub+count\_no\_sub))\*100 11.698480458295547 Out[83]: In [84]: bank\_data1=pd.get\_dummies(data=bank\_data,columns=['job','marital','education','contact','poutcome','month']) y job\_admin. job\_bluejob\_self-employed Out[84]: age default balance housing loan day duration campaign pdays previous job\_entrepreneur job\_housemaid job\_management job\_retired 0 0 1 0 **0** 58 1 -1 0 no 0 0 0 2143 no 5 261 no yes **1** 44 29 5 151 no yes no no 0 0 0 0 0 33 2 5 76 1 -1 0 no 0 1 no yes yes **3** 47 1506 5 0 no yes no 0 no 0 0 0 0 0 0 0 33 5 198 -1 0 no 4 1 1 no no no 0 45206 3 0 yes 0 0 0 0 0 0 51 no 825 no no 17 977 -1 45207 71 no 1729 no no 17 456 0 yes 0 45208 5 184 3 yes 0 0 0 0 0 1 0 72 5715 17 1127 no no no 45209 57 17 508 0 0 0 668 no no 0 no no 0 0 0 0 0 0 45210 2 188 11 no 1 37 2971 17 361 no no no 45211 rows × 49 columns In [85]: pd.set\_option('display.max.columns', None) bank\_data1 job\_blue-Out[85]: y job\_admin. job\_entrepreneur job\_housemaid job\_management job\_retired age default balance housing loan day duration campaign pdays previous collar employed 0 58 2143 5 261 1 -1 no 1 0 no yes no 0 0 no 0 **1** 44 no 29 yes no 5 151 -1 33 no 2 yes yes 5 76 -1 0 no 0 0 1 0 0 0 0 1506 0 0 0 0 0 **3** 47 5 92 -1 0 no no yes no 33 5 198 1 -1 0 0 0 0 0 0 0 no 1 no no 0 no 45206 51 825 17 977 3 -1 0 0 0 0 0 0 0 no no no 0 yes **45207** 71 1729 456 0 0 0 0 1 0 no no no 17 -1 0 yes 45208 72 5715 17 1127 184 0 0 0 1 0 no no no 3 yes 0 0 0 0 45209 57 668 17 508 -1 no no 0 no 0 0 45210 37 2971 17 361 188 11 no 0 0 1 0 0 no no no 45211 rows × 49 columns In [86]: bank\_data1.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 45211 entries, 0 to 45210 Data columns (total 49 columns): Non-Null Count Dtype Column 0 45211 non-null int64 age 1 default 45211 non-null object balance 45211 non-null int64 housing 45211 non-null object loan 45211 non-null object day 45211 non-null int64 6 7 duration 45211 non-null int64 campaign 45211 non-null int64 8 45211 non-null int64 pdays 9 previous 45211 non-null int64 10 45211 non-null object 11 job\_admin. 45211 non-null uint8 job\_blue-collar 45211 non-null 12 uint8 job\_entrepreneur 45211 non-null uint8 13 job\_housemaid 45211 non-null uint8 job\_management 45211 non-null uint8 job\_retired 16 45211 non-null uint8 17 job\_self-employed 45211 non-null uint8 job\_services 18 45211 non-null uint8 19 job\_student 45211 non-null uint8 20 job\_technician 45211 non-null uint8 21 job\_unemployed 45211 non-null uint8 job\_unknown 45211 non-null uint8 marital\_divorced 45211 non-null uint8 24 marital\_married 45211 non-null uint8 25 marital\_single 45211 non-null uint8 education\_primary 26 45211 non-null uint8 27 education\_secondary 45211 non-null uint8 education\_tertiary 45211 non-null 28 uint8 29 education\_unknown 45211 non-null uint8 45211 non-null 30 contact\_cellular uint8 contact\_telephone 45211 non-null uint8 32 contact\_unknown 45211 non-null uint8 33 poutcome\_failure 45211 non-null uint8 45211 non-null 34 poutcome\_other uint8 35 poutcome\_success 45211 non-null uint8 36 poutcome\_unknown 45211 non-null uint8 37 month\_apr 45211 non-null uint8 38 month\_aug 45211 non-null uint8 39 month\_dec 45211 non-null uint8 40 month\_feb 45211 non-null uint8 month\_jan 41 45211 non-null uint8 month\_jul 42 45211 non-null uint8 43 month\_jun 45211 non-null uint8 month\_mar 45211 non-null 44 uint8 45 month\_may 45211 non-null uint8 46 month\_nov 45211 non-null uint8 47 month\_oct 45211 non-null uint8 48 month\_sep 45211 non-null uint8 dtypes: int64(7), object(4), uint8(38) memory usage: 5.4+ MB In [87]: bank\_data1['default'] = np.where(bank\_data1['default'].str.contains("yes"), 1, 0) bank\_data1['housing'] = np.where(bank\_data1['housing'].str.contains("yes"), 1, 0) bank\_data1['loan'] = np.where(bank\_data1['loan'].str.contains("yes"), 1, 0) bank\_data1['y'] = np.where(bank\_data1['y'].str.contains("yes"), 1, 0) bank\_data1 job\_blue-Out[87]: job\_selfage default balance housing loan day duration campaign pdays previous y job\_admin. job\_entrepreneur job\_housemaid job\_management job\_retired collar employed **0** 58 0 2143 0 5 261 1 -1 0 0 0 0 0 0 1 0 0 1 **1** 44 29 151 0 0 0 2 0 0 0 0 0 0 0 **2** 33 0 5 76 1 -1 0 1 1 1 47 1506 5 0 0 0 0 0 0 4 33 0 0 0 5 198 1 -1 0 0 0 0 0 0 1 45206 51 0 825 0 0 17 977 3 -1 0 1 0 0 0 0 0 0 0 0 45207 1729 0 17 456 45208 72 0 1127 184 0 0 0 0 0 1 0 5715 0 0 17 5 3 1 45209 57 668 17 508 0 0 0 **45210** 37 0 2971 0 17 361 2 188 11 0 0 0 1 0 0 0 0 45211 rows × 49 columns In [88]: bank\_data1.info() <class 'pandas.core.frame.DataFrame'> RangeIndex: 45211 entries, 0 to 45210 Data columns (total 49 columns): Non-Null Count Dtype Column 0 45211 non-null int64 age 45211 non-null default 1 int32 45211 non-null balance int64 45211 non-null housing int32 loan 45211 non-null int32 5 45211 non-null int64 day 6 duration 45211 non-null int64 45211 non-null campaign int64 8 pdays 45211 non-null 9 45211 non-null previous int64 45211 non-null 10 int32 job\_admin. 45211 non-null uint8 11 job\_blue-collar 45211 non-null uint8 job\_entrepreneur 45211 non-null uint8 job\_housemaid 45211 non-null 14 uint8 job\_management 15 45211 non-null uint8 16 job\_retired 45211 non-null uint8 job\_self-employed 45211 non-null 17 uint8 job\_services 45211 non-null uint8 18 job\_student 45211 non-null uint8 job\_technician 45211 non-null uint8 job\_unemployed 45211 non-null uint8 22 job\_unknown 45211 non-null uint8 23 marital\_divorced 45211 non-null uint8 marital\_married 45211 non-null uint8 marital\_single 25 45211 non-null uint8 education\_primary 45211 non-null 26 uint8 education\_secondary 45211 non-null uint8 education\_tertiary 45211 non-null uint8 29 education\_unknown 45211 non-null uint8 contact\_cellular 45211 non-null uint8 30 31 contact\_telephone 45211 non-null uint8 contact\_unknown 45211 non-null uint8 poutcome\_failure 33 45211 non-null uint8 45211 non-null 34 poutcome\_other uint8 45211 non-null 35 poutcome\_success uint8 poutcome\_unknown 45211 non-null uint8 45211 non-null 37 month\_apr uint8 38 45211 non-null month\_aug uint8 month\_dec 45211 non-null 39 uint8 40 month\_feb 45211 non-null uint8 month\_jan 41 45211 non-null uint8 month\_jul 45211 non-null 42 uint8 43 month\_jun 45211 non-null uint8 44 month\_mar 45211 non-null uint8 45 month\_may 45211 non-null uint8 46 month\_nov 45211 non-null uint8 45211 non-null uint8 47 month\_oct 48 month\_sep 45211 non-null uint8 dtypes: int32(4), int64(7), uint8(38) memory usage: 4.7 MB **Model Building** In [89]: import warnings warnings.filterwarnings('ignore') In [90]: x=pd.concat([bank\_data1.iloc[:,0:10],bank\_data1.iloc[:,11:]],axis=1) y=bank\_data1.iloc[:,10] In [91]: classifier=LogisticRegression() classifier.fit(x,y)Out[91]: LogisticRegression() **Model Predictions** In [92]: y\_pred=classifier.predict(x) y\_pred Out[92]: array([0, 0, 0, ..., 1, 0, 0]) In [93]: y\_pred\_bank\_data=pd.DataFrame({'actual\_y':y,'y\_pred\_prob':y\_pred}) y\_pred\_bank\_data actual\_y y\_pred\_prob Out[93]: 0 0 0 2 0 0 0 4 0 0 45206 1 1 45207 45208 1 1 45209 45210 0 45211 rows × 2 columns **Testing Model Accuracy** In [94]: # confusion Matrix for the model accuracy confusion\_matrix=confusion\_matrix(y,y\_pred) confusion\_matrix Out[94]: array([[39159, 763], [ 4140, 1149]], dtype=int64) In [95]: # The Model accuracy is calculated by (a+d/a+b+c+d)(39159+1149)/(39159+763+4140+1149) Out[95]: 0.8915529406560351 In [66]: classifier.predict\_proba(x)[:,1] Out[66]: array([0.03564107, 0.02150632, 0.01710036, ..., 0.63604954, 0.06555453, 0.13937723]) In [96]: # ROC Curve plotting and finding AUC value fpr, tpr, thresholds=roc\_curve(y, classifier.predict\_proba(x)[:,1]) plt.plot(fpr, tpr, color='red') auc=roc\_auc\_score(y,y\_pred) plt.plot(fpr,tpr,color='red',label='logit model(area = %0.2f)'%auc) plt.plot([0,1],[0,1],'k--') plt.xlabel('False Positive Rate or [1 - True Negative Rate]') plt.ylabel('True Positive Rate') plt.show() 1.0 0.8 0.4 0.2 0.0 0.4 0.6 False Positive Rate or [1 - True Negative Rate] In [97]: print('auc accuracy:',auc) auc accuracy: 0.5990655331498237