In [1]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix,accuracy_score,classification_report
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
```

In [3]:
```
bank=pd.read_csv('bank-full.csv',sep =';' )
bank.head()
```

Out[3]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	camp
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	

In [4]:
```
bank.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        45211 non-null  int64 
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64 
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64 
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64 
 12  campaign   45211 non-null  int64 
 13  pdays      45211 non-null  int64 
 14  previous   45211 non-null  int64 
 15  poutcome   45211 non-null  object
 16  y          45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

In [5]:
```
bank.shape
```

Out[5]:
```
(45211, 17)
```

In [17]:
```
categorical = bank.columns[:-1]
bank[categorical].isnull().sum()
```

Out[17]:
```
age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
dtype: int64
```

In [18]:
```
bank[['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact', 'month', '
```

In [19]:
```
X = bank.iloc[:,[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15]]
Y = bank.iloc[:,16]
classifier = LogisticRegression()
classifier.fit(X,Y)
```

```
C:\Users\ROHIT\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: Conver
genceWarning: 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(
```

Out[19]:
```
LogisticRegression()
```

In [20]:
```
classifier.coef_
```

Out[20]:
```
array([[-4.36662380e-02, -8.07510142e-02, -4.06278811e-02,
        -1.35674416e-01, -6.23571466e-03,  2.49143211e-05,
         1.32272447e-01, -6.95360809e-02,  3.16190985e-02,
        -3.33486021e-02,  6.48981045e-02,  3.31466269e-03,
        -3.33095578e-01, -7.75053899e-04,  1.18399646e-01,
         2.60749121e-01]])
```

In [21]:
```
classifier.predict_proba (X)
```

Out[21]:
```
array([[0.90301051, 0.09698949],
       [0.9083905 , 0.0916095 ],
       [0.89777492, 0.10222508],
       ...,
       [0.66206691, 0.33793309],
       [0.93480094, 0.06519906],
       [0.45678541, 0.54321459]])
```

In [22]:
```
y_pred = classifier.predict(X)
bank["y_pred"] = y_pred
bank
```

Out[22]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign
0	58	0	0	0	0	2143	0	0	0	5	0	261	1
1	44	1	1	1	0	29	0	0	0	5	0	151	1
2	33	2	0	1	0	2	0	1	0	5	0	76	1
3	47	3	0	2	0	1506	0	0	0	5	0	92	1
4	33	4	1	2	0	1	1	0	0	5	0	198	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	1	0	0	0	825	1	0	1	17	5	977	3
45207	71	5	2	3	0	1729	1	0	1	17	5	456	2
45208	72	5	0	1	0	5715	1	0	1	17	5	1127	5
45209	57	3	0	1	0	668	1	0	2	17	5	508	4
45210	37	2	0	1	0	2971	1	0	1	17	5	361	2

45211 rows × 18 columns

In [23]:
```
y_prob = pd.DataFrame(classifier.predict_proba(X.iloc[:,:]))
new_df = pd.concat([bank,y_prob],axis=1)
new_df
```

Out[23]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign
0	58	0	0	0	0	2143	0	0	0	5	0	261	1
1	44	1	1	1	0	29	0	0	0	5	0	151	1
2	33	2	0	1	0	2	0	1	0	5	0	76	1
3	47	3	0	2	0	1506	0	0	0	5	0	92	1
4	33	4	1	2	0	1	1	0	0	5	0	198	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	1	0	0	0	825	1	0	1	17	5	977	3
45207	71	5	2	3	0	1729	1	0	1	17	5	456	2
45208	72	5	0	1	0	5715	1	0	1	17	5	1127	5
45209	57	3	0	1	0	668	1	0	2	17	5	508	4
45210	37	2	0	1	0	2971	1	0	1	17	5	361	2

45211 rows × 20 columns

In [24]:
```
confusion_matrix = confusion_matrix(Y,y_pred)
print (confusion_matrix)
```

```
[[39119   803]
 [ 4266  1023]]
```

In [25]:
```
pd.crosstab(y_pred,Y)
```

Out[25]:

y	0	1
row_0		
0	39119	4266
1	803	1023

In [26]:
```
accuracy = sum(Y==y_pred)/bank.shape[0]
accuracy
```

Out[26]:
```
0.8878812678330494
```

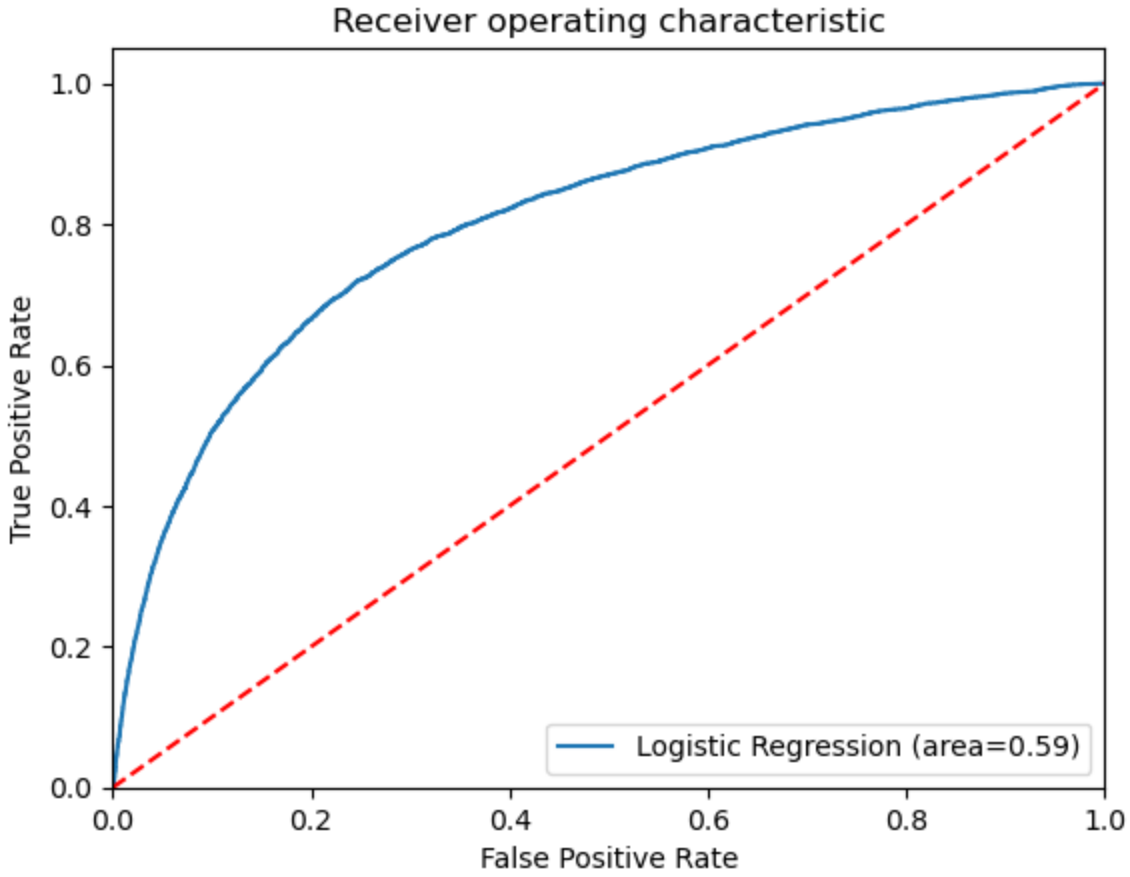
In [27]:
```
print (classification_report (Y, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.90      0.98      0.94     39922
           1       0.56      0.19      0.29      5289

    accuracy                           0.89     45211
   macro avg       0.73      0.59      0.61     45211
weighted avg       0.86      0.89      0.86     45211
```

In [28]:
```
Logit_roc_score=roc_auc_score(Y,classifier.predict(X))
Logit_roc_score
```

Out[28]:
```
0.5866530417808772
```

In [29]:
```
fpr, tpr, thresholds = roc_curve(Y,classifier.predict_proba(X)[:,1])
plt.plot(fpr, tpr, label='Logistic Regression (area=%0.2f)'% Logit_roc_score)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```



In [30]:
```
y_prob1 = pd.DataFrame(classifier.predict_proba(X)[:,1])
y_prob1
```

Out[30]:

	0
0	0.096989
1	0.091610
2	0.102225
3	0.055177
4	0.129614
...	...
45206	0.439362
45207	0.035582
45208	0.337933
45209	0.065199
45210	0.543215

45211 rows × 1 columns

In [ ]:
```

```