In [20]:

```
#import files
import pandas as pd
import seaborn as sb
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression

import pdb;

from IPython.core.debugger import set_trace
import random
import matplotlib.pyplot as plt
```

In [21]:

```
#Read csv
electricity_data = pd.read_csv("energydata_complete.csv")
electricity_data_appliance_rand10 =  electricity_data
```

In [22]:

```
x_appliances = electricity_data_appliance_rand10.drop(lab
els = ['date','lights','Appliances','T6','RH_6','T7','RH_
7','T8','RH_8',

'T9','RH_9','rv1','rv2','T5','RH_5','T4','RH_4','Tdewpoin
t',

'RH_out'],axis = 1)
```

In [23]:

```
y_appliances = electricity_data_appliance_rand10[['Appliances']]
```

In [24]:

```
random.seed(1)
```

In [25]:

```
scaler = MinMaxScaler()
x_appliances_scaled = x_appliances
y_appliances_scaled = y_appliances

x_appliances_scaled = scaler.fit_transform(x_appliances)
y_appliances_scaled = scaler.fit_transform(y_appliances)
```

In [26]:

```
x_appliances_scaled = np.array(x_appliances_scaled)
```

In [27]:

```
x_appliances_scaled
```

Out[27]:

```
array([[0.32734952, 0.56618659, 0.22534529,
..., 0.09767442, 0.5       ,
        0.95384615],
       [0.32734952, 0.54132648, 0.22534529,
..., 0.1       , 0.47619048,
        0.89487179],
       [0.32734952, 0.53050179, 0.22534529,
..., 0.10232558, 0.45238095,
        0.83589744],
       ...,
       [0.91974657, 0.53866618, 0.69265118,
..., 0.60232558, 0.26190476,
        0.37435897],
       [0.91974657, 0.54949087, 0.67705355,
..., 0.60232558, 0.27380952,
        0.38717949],
       [0.91974657, 0.53875791, 0.66617051,
..., 0.60232558, 0.28571429,
        0.4       ]])
```

In [28]:

```
pd.DataFrame(y_appliances_scaled).median()
y_appliances_scaled = np.where(y_appliances_scaled<0.04,0
,1)
```

In [29]:

```
from sklearn.linear_model import LogisticRegression
random.seed(1)
X_Train_logistic, X_Test_logistic, Y_Train_logistic, Y_Te
st_logistic = train_test_split(x_appliances_scaled, y_app
liances_scaled, test_size=0.3, random_state=1)
```

In [30]:

```
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_auc_score


accuracy_Values = []


logi_model = SGDClassifier(loss='log', alpha = 0.0001 , m
ax_iter=10, random_state = 1)
logi_model.fit(X_Train_logistic, Y_Train_logistic)
validation_data = logi_model.predict(X_Test_logistic)
accuracy = accuracy_score(Y_Test_logistic.flatten(),valid
ation_data)
accuracy_Values.append(accuracy)

#confusion_matrix()
print(confusion_matrix(Y_Test_logistic,validation_data))
print(logi_model.coef_)
print(logi_model.intercept_)
roc_auc_score(Y_Test_logistic, validation_data)
```

```
[[ 686 1525]
 [ 337 3373]]
[[ 2.22681251e+00  8.32332098e+00  7.4408014
9e-01 -4.67563608e+00
  -3.44510305e-01 -4.32572214e+00 -1.3523401
8e-03 -9.03088098e-01
   7.70084881e-01 -3.23219871e-03]]
[1.54884412]

c:\users\siddharth\appdata\local\programs\py
thon\python37-32\lib\site-packages\sklearn\u
tils\validation.py:724: DataConversionWarnin
g: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of
y to (n_samples, ), for example using ravel
().
  y = column_or_1d(y, warn=True)
c:\users\siddharth\appdata\local\programs\py
thon\python37-32\lib\site-packages\sklearn\l
inear_model\stochastic_gradient.py:561: Conv
ergenceWarning: Maximum number of iteration
reached before convergence. Consider increas
ing max_iter to improve the fit.
  ConvergenceWarning)
```

Out[30]:

```
0.6097156340327279
```

In [31]:

```
auc = roc_auc_score(Y_Test_logistic,validation_data)
auc
```

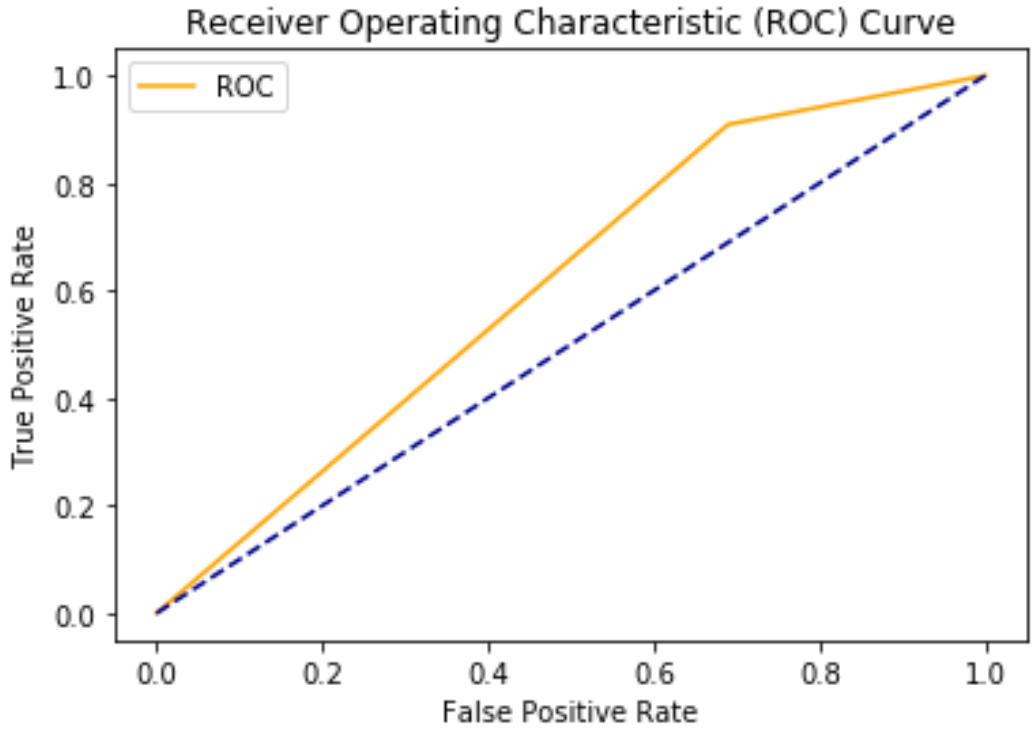
Out[31]:

```
0.6097156340327279
```

In [32]:

```
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt


fpr, tpr, _ = metrics.roc_curve(Y_Test_logistic,validatio
n_data)
plt.plot(fpr,  tpr, color='orange', label='ROC')
plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--'
)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve'
)
plt.legend()
plt.show()
```



In [33]:

```
from sklearn.metrics import classification_report
print(classification_report(Y_Test_logistic,validation_data, target_names=['0','1']))
```

```
              precision    recall  f1-score   support

           0       0.67      0.31      0.42      2211
           1       0.69      0.91      0.78      3710

    accuracy                           0.69      5921
   macro avg       0.68      0.61      0.60      5921
weighted avg       0.68      0.69      0.65      5921
```

In [34]:

```
x_appliances = electricity_data_appliance_rand10[['T1','RH_1','T3','RH_3','T4','RH_4','T8','RH_8','T9', 'RH_9']]
y_appliances = electricity_data_appliance_rand10[['Appliances']]
```

In [35]:

```
scaler = MinMaxScaler()
x_appliances_scaled = x_appliances
y_appliances_scaled = y_appliances

x_appliances_scaled = scaler.fit_transform(x_appliances)
y_appliances_scaled = scaler.fit_transform(y_appliances)
```

In [36]:

```
x_appliances_scaled = np.array(x_appliances_scaled)
```

In [37]:

```
pd.DataFrame(y_appliances_scaled).median()
y_appliances_scaled = np.where(y_appliances_scaled<0.04,0
,1)
```

In [38]:

```
from sklearn.linear_model import LogisticRegression
random.seed(1)
X_Train_logistic, X_Test_logistic, Y_Train_logistic, Y_Te
st_logistic = train_test_split(x_appliances_scaled, y_app
liances_scaled, test_size=0.3, random_state=1)
```

In [57]:

```
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_auc_score


accuracy_Values = []


logi_model = SGDClassifier(loss='log', alpha = 0.0001 , m
ax_iter=10, random_state = 1)
logi_model.fit(X_Train_logistic, Y_Train_logistic)
validation_data = logi_model.predict(X_Test_logistic)
accuracy = accuracy_score(Y_Test_logistic.flatten(),valid
ation_data)
accuracy_Values.append(accuracy)

#confusion_matrix()
print(confusion_matrix(Y_Test_logistic,validation_data))
print(logi_model.coef_)
print(logi_model.intercept_)
roc_auc_score(Y_Test_logistic, validation_data)
```

```
[[1053 1158]
 [ 565 3145]]
[[ 0.71978469  5.92606675  2.54289672 -1.058
61961  1.95048405  4.45227316
   4.26025306 -5.81277829 -5.8801393  -2.721
05914]]
[-0.91008385]

c:\users\siddharth\appdata\local\programs\py
thon\python37-32\lib\site-packages\sklearn\u
tils\validation.py:724: DataConversionWarnin
g: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of
y to (n_samples, ), for example using ravel
().
  y = column_or_1d(y, warn=True)
c:\users\siddharth\appdata\local\programs\py
thon\python37-32\lib\site-packages\sklearn\l
inear_model\stochastic_gradient.py:561: Conv
ergenceWarning: Maximum number of iteration
reached before convergence. Consider increas
ing max_iter to improve the fit.
  ConvergenceWarning)
```

Out[57]:

```
0.66198199153704 64
```

In [61]:

```
auc = roc_auc_score(Y_Test_logistic,validation_data)
auc
```

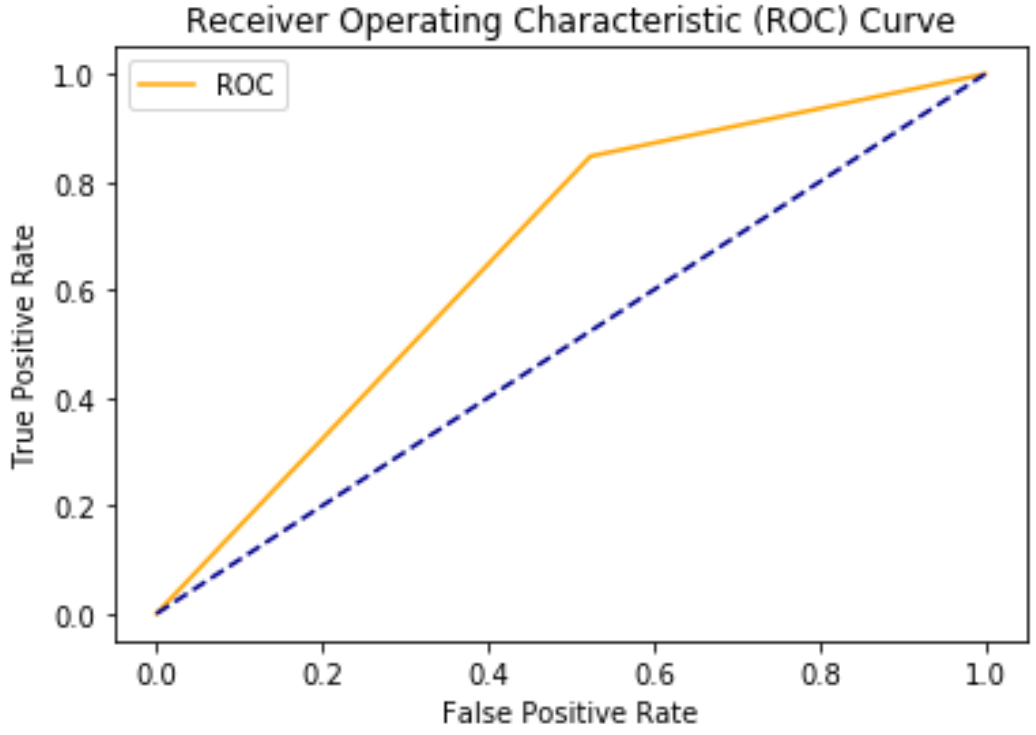
Out[61]:

```
0.66198199153704 64
```

In [59]:

```
from sklearn import metrics
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt


fpr, tpr, _ = metrics.roc_curve(Y_Test_logistic,validatio
n_data)
plt.plot(fpr,  tpr, color='orange', label='ROC')
plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--'
)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve'
)
plt.legend()
plt.show()
```



In [60]:

```
from sklearn.metrics import classification_report
print(classification_report(Y_Test_logistic,validation_da
ta, target_names=['0','1']))
```

```
              precision    recall  f1-score
support

           0       0.65      0.48      0.55
2211
           1       0.73      0.85      0.78
3710

    accuracy                           0.71
5921
   macro avg       0.69      0.66      0.67
5921
weighted avg       0.70      0.71      0.70
5921
```

In [ ]:

```

```