Графичекая характеристика качества бинарного классификатора, отображает зависимость доли верных положительных классификаций True Positive Rate (TPR) от доли ложных положительных классификаций False Positive Rate (FPR).

$$TPR = \frac{TP}{TP+FN} \quad FPR = \frac{FP}{FP+TN}$$ TPR=TPTP+FNFPR=FPFP+TN

ROCAUC - https://rebeccabilbro.github.io/xgboost-and-yellowbrick/¶

В [113]:

```
#!pip install scikit-plot  # Successfully installed scikit-plot-0.3.7
```

В [114]:

```
import scikitplot as skplt
import matplotlib.pyplot as plt
"""
y_true = # ground truth labels
y_probas = # predicted probabilities generated by sklearn classifier

skplt.metrics.plot_roc_curve(y_true, y_probas)
plt.show()
"""
```

Out[114]:

```
'\ny_true = # ground truth labels\ny_probas = # predicted probabilities gener
ated by sklearn classifier\n\nskplt.metrics.plot_roc_curve(y_true, y_probas)
\nplt.show() \n'
```

Как построить кривую ROC (кривая ошибок)

- https://coderoad.ru/25009284/как-plot-ROC-кривая-в-Puthon

  Количественную интерпретацию ROC даёт показатель AUC (англ. area under ROC curve, площадь под ROC-кривой) — площадь, ограниченная ROC-кривой и осью доли ложных положительных классификаций.

В [115]:

```
#!pip install plot-metric  # Successfully installed colorlover-0.3.0 plot-metric-0.0.6
```

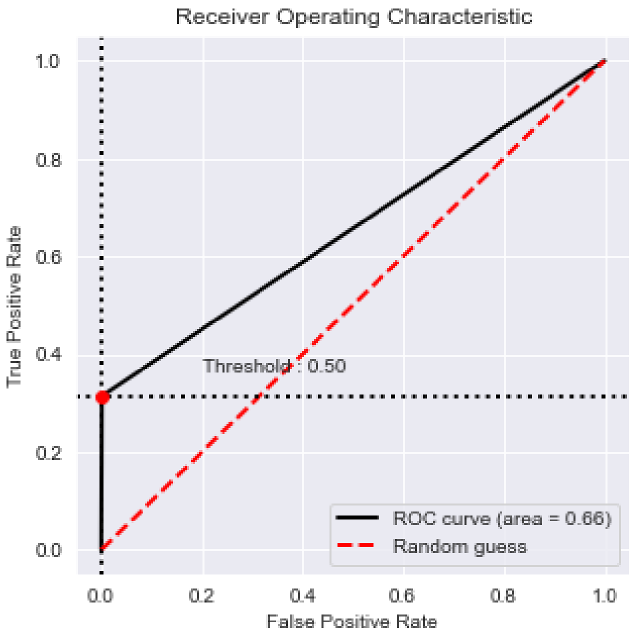
XGBoost plot Roc кривая

В [116]:

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score

TRAIN_DATASET_PATH = './data/assignment_2_train.csv'
train = pd.read_csv(TRAIN_DATASET_PATH)

numerical_features = train.select_dtypes(include=['float32', 'float64', 'int8', 'int16', 'i

data = train[numerical_features]
target = train['isFraud']

x_train, x_valid = train_test_split(data, train_size=0.8, random_state=1)
y_train, y_valid = train_test_split(target, train_size=0.8, random_state=1)

params = {
    "booster": "gbtree",
    "objective": "binary:logistic",
    "eval_metric": "auc",
    "learning_rate": 0.1,
    "n_estimators": 1000,
    "reg_lambda": 100,
    "max_depth": 4,
    "gamma": 10,
    "nthread": 6,
    "seed": 27
}

model = xgb.XGBClassifier(**params)
model.fit(
    X=x_train,
    y=y_train,
    eval_set=[(x_train, y_train), (x_valid, y_valid)],
    early_stopping_rounds=50,
    eval_metric="auc",
    verbose=10
)
# make predictions for test data
y_pred = model.predict(x_valid)
preds = [round(value) for value in y_pred]
y_test = list(y_valid)

from plot_metric.functions import BinaryClassification
# Visualisation with plot_metric
bc = BinaryClassification(y_test, y_pred, labels=["Class 1", "Class 2"])

# Figures
plt.figure(figsize=(5,5))
bc.plot_roc_curve()
plt.show()
```

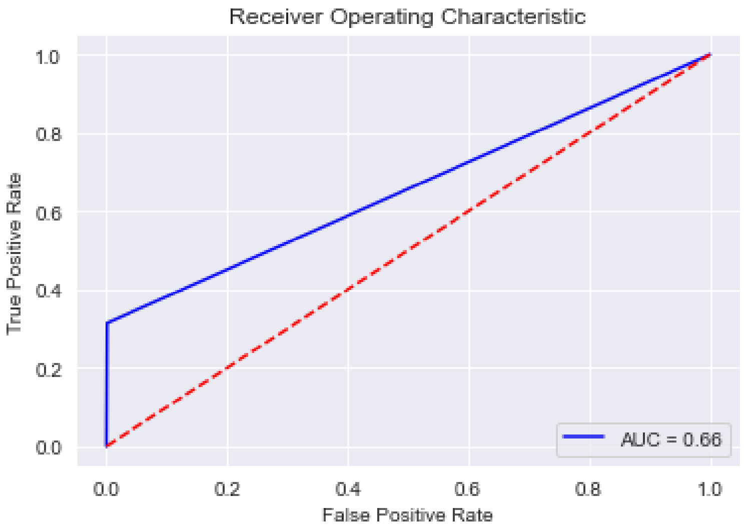
```
C:\ProgramData\Anaconda3\lib\site-packages\xgboost\sklearn.py:888: UserWarnin
g: The use of label encoder in XGBClassifier is deprecated and will be remove
d in a future release. To remove this warning, do the following: 1) Pass opti
on use_label_encoder=False when constructing XGBClassifier object; and 2) Enc
ode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_clas
s - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
[0]	validation_0-auc:0.64988	validation_1-auc:0.65040
[10]	validation_0-auc:0.78991	validation_1-auc:0.79059
[20]	validation_0-auc:0.83801	validation_1-auc:0.83302
[30]	validation_0-auc:0.86187	validation_1-auc:0.85714
[40]	validation_0-auc:0.87047	validation_1-auc:0.86388
[50]	validation_0-auc:0.87698	validation_1-auc:0.86934
[60]	validation_0-auc:0.88170	validation_1-auc:0.87415
[70]	validation_0-auc:0.88510	validation_1-auc:0.87785
[80]	validation_0-auc:0.88748	validation_1-auc:0.87974
[90]	validation_0-auc:0.88955	validation_1-auc:0.88149
[100]	validation_0-auc:0.89150	validation_1-auc:0.88314
[110]	validation_0-auc:0.89299	validation_1-auc:0.88477
[120]	validation_0-auc:0.89496	validation_1-auc:0.88597
[130]	validation_0-auc:0.89628	validation_1-auc:0.88710
[140]	validation_0-auc:0.89757	validation_1-auc:0.88846
[150]	validation_0-auc:0.89855	validation_1-auc:0.88944
[160]	validation_0-auc:0.89855	validation_1-auc:0.88944
[170]	validation_0-auc:0.89855	validation_1-auc:0.88944
[180]	validation_0-auc:0.89855	validation_1-auc:0.88944
[190]	validation_0-auc:0.89855	validation_1-auc:0.88944
[198]	validation_0-auc:0.89855	validation_1-auc:0.88944
```



В [117]:

```
#XGBoost plot Roc кривая
from sklearn import metrics
def buildROC(target_test,test_preds):
    fpr, tpr, threshold = metrics.roc_curve(target_test, test_preds)
    roc_auc = metrics.auc(fpr, tpr)
    plt.title('Receiver Operating Characteristic')
    plt.plot(fpr, tpr, 'b', label = 'AUC = %0.2f' % roc_auc)
    plt.legend(loc = 'lower right')
    plt.plot([0, 1], [0, 1],'r--')
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.gcf().savefig('roc.png')

buildROC(y_test, y_pred)
```


Receiver Operating Characteristic
True Positive Rate
AUC = 0.66
False Positive Rate