Social network Graph Link Prediction - Facebook Challenge

In [1]:

```
#Importing Libraries
# please do go through this python notebook:
import warnings
warnings.filterwarnings("ignore")

import csv
import pandas as pd#pandas to create small dataframes
import datetime #Convert to unix time
import time #Convert to unix time
# if numpy is not installed already : pip3 install numpy
import numpy as np#Do aritmetic operations on arrays
# matplotlib: used to plot graphs
import matplotlib
import matplotlib.pylab as plt
import seaborn as sns#Plots
from matplotlib import rcParams#Size of plots
from sklearn.cluster import MiniBatchKMeans, KMeans#Clustering
import math
import pickle
import os
# to install xgboost: pip3 install xgboost
import xgboost as xgb

import warnings
import networkx as nx
import pdb
import pickle
from pandas import HDFStore,DataFrame
from pandas import read_hdf
from scipy.sparse.linalg import svds, eigs
import gc
from tqdm import tqdm
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score
```

In [2]:

```
!wget --header="Host: doc-0o-bk-docs.googleusercontent.com" --header="User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212 Safari/537.36" --header="Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9" --header="Accept-Language: en-US,en;q=0.9" --header="Cookie: AUTH_nso6dcn1mbidkt5qr539a2jiefc09pqv_nonce=iak2ig7rpq664" --header="Connection: keep-alive" "https://doc-0o-bk-docs.googleusercontent.com/docs/securesc/nss2f5s2soorprev6d4t4qp3n5ekp9nh/evl2j2j4t5hronicnhsbdlsblnbl9qk3/1622116650000/06629147635963609455/13017565264516993811/1fDJptlCFEWNV5UNGPc4geTykgFI3PDCV?e=download&authuser=0&nonce=iak2ig7rpq664&user=13017565264516993811&hash=fvl5s6dohfnqle6k8q3koe9jr2mhe6jr" -c -O 'storage_sample_stage4.h5'
```

```
--2021-07-23 06:36:01--  https://doc-0o-bk-docs.googleusercontent.com/docs/securesc/nss2f5s2soorprev6d4t4qp3n5ekp9nh/evl2j2j4t5hronicnhsbdlsblnbl9qk3/1622116650000/06629147635963609455/13017565264516993811/1fDJptlCFEWNV5UNGPc4geTykgFI3PDCV?e=download&authuser=0&nonce=iak2ig7rpq664&user=13017565264516993811&hash=fvl5s6dohfnqle6k8q3koe9jr2mhe6jr
Resolving doc-0o-bk-docs.googleusercontent.com (doc-0o-bk-docs.googleusercontent.com)... 74.125.141.132, 2607:f8b0:400c:c06::84
Connecting to doc-0o-bk-docs.googleusercontent.com (doc-0o-bk-docs.googleusercontent.com)|74.125.141.132|:443... connected.
HTTP request sent, awaiting response... 403 Forbidden
2021-07-23 06:36:01 ERROR 403: Forbidden.
```

In [3]:

```
#reading
```

```
from pandas import read_hdf
df_final_train = read_hdf('storage_sample_stage4.h5', 'train_df',mode='r')
df_final_test = read_hdf('storage_sample_stage4.h5', 'test_df',mode='r')
```

In [4]:

```
df_final_train.columns
```

Out[4]:

```
Index(['source_node', 'destination_node', 'indicator_link',
       'jaccard_followers', 'jaccard_followees', 'cosine_followers',
       'cosine_followees', 'num_followers_s', 'num_followees_s',
       'num_followers_d', 'num_followees_d', 'inter_followers',
       'inter_followees', 'adar_index', 'follows_back', 'same_comp',
       'shortest_path', 'weight_in', 'weight_out', 'weight_f1', 'weight_f2',
       'weight_f3', 'weight_f4', 'page_rank_s', 'page_rank_d', 'katz_s',
       'katz_d', 'hubs_s', 'hubs_d', 'authorities_s', 'authorities_d',
       'svd_u_s_1', 'svd_u_s_2', 'svd_u_s_3', 'svd_u_s_4', 'svd_u_s_5',
       'svd_u_s_6', 'svd_u_d_1', 'svd_u_d_2', 'svd_u_d_3', 'svd_u_d_4',
       'svd_u_d_5', 'svd_u_d_6', 'svd_v_s_1', 'svd_v_s_2', 'svd_v_s_3',
       'svd_v_s_4', 'svd_v_s_5', 'svd_v_s_6', 'svd_v_d_1', 'svd_v_d_2',
       'svd_v_d_3', 'svd_v_d_4', 'svd_v_d_5', 'svd_v_d_6'],
      dtype='object')
```

In [5]:

```
y_train = df_final_train.indicator_link
y_test = df_final_test.indicator_link
```

In [6]:

```
df_final_train.drop(['source_node', 'destination_node','indicator_link'],axis=1,inplace=T
rue)
df_final_test.drop(['source_node', 'destination_node','indicator_link'],axis=1,inplace=Tr
ue)
```

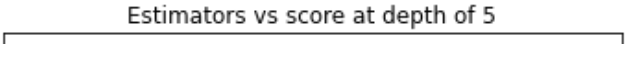
In [7]:

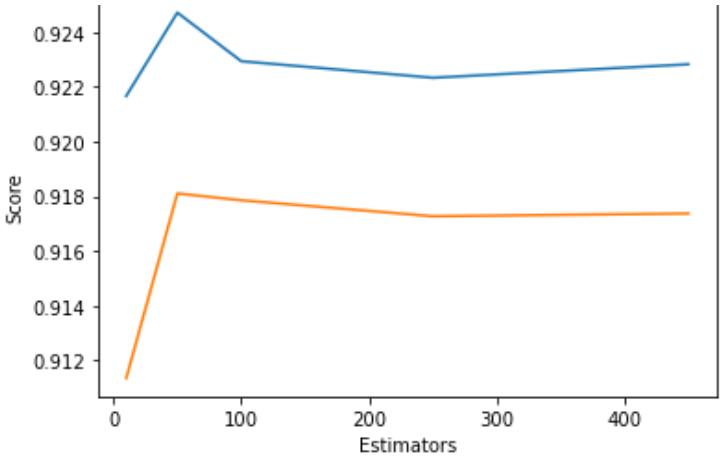
```
estimators = [10,50,100,250,450]
train_scores = []
test_scores = []
for i in estimators:
    clf = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=5, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=52, min_samples_split=120,
            min_weight_fraction_leaf=0.0, n_estimators=i, n_jobs=-1,random_state=25,verb
ose=0,warm_start=False)
    clf.fit(df_final_train,y_train)
    train_sc = f1_score(y_train,clf.predict(df_final_train))
    test_sc = f1_score(y_test,clf.predict(df_final_test))
    test_scores.append(test_sc)
    train_scores.append(train_sc)
    print('Estimators = ',i,'Train Score',train_sc,'test Score',test_sc)
plt.plot(estimators,train_scores,label='Train Score')
plt.plot(estimators,test_scores,label='Test Score')
plt.xlabel('Estimators')
plt.ylabel('Score')
plt.title('Estimators vs score at depth of 5')
```

```
Estimators =  10 Train Score 0.9216704714432321 test Score 0.9113566398178214
Estimators =  50 Train Score 0.9247164336586653 test Score 0.9181063685636855
Estimators =  100 Train Score 0.9229408792319352 test Score 0.9178562378579272
Estimators =  250 Train Score 0.9223348008341584 test Score 0.9172722663511916
Estimators =  450 Train Score 0.9228272438903583 test Score 0.917373015705257
```

Out[7]:

```
Text(0.5, 1.0, 'Estimators vs score at depth of 5')
```

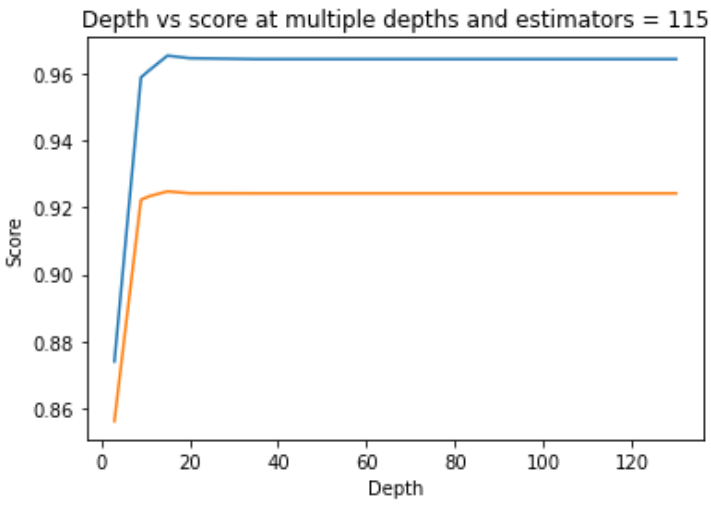




In [9]:

```
depths = [3,9,11,15,20,35,50,70,130]
train_scores = []
test_scores = []
for i in depths:
    clf = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=i, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=52, min_samples_split=120,
            min_weight_fraction_leaf=0.0, n_estimators=115, n_jobs=-1,random_state=25,ve
rbose=0,warm_start=False)
    clf.fit(df_final_train,y_train)
    train_sc = f1_score(y_train,clf.predict(df_final_train))
    test_sc = f1_score(y_test,clf.predict(df_final_test))
    test_scores.append(test_sc)
    train_scores.append(train_sc)
    print('depth = ',i,'Train Score',train_sc,'test Score',test_sc)
plt.plot(depths,train_scores,label='Train Score')
plt.plot(depths,test_scores,label='Test Score')
plt.xlabel('Depth')
plt.ylabel('Score')
plt.title('Depth vs score at multiple depths and estimators = 115')
plt.show()
```

```
depth =  3 Train Score 0.8740127786577022 test Score 0.8562011612170951
depth =  9 Train Score 0.9589131126834328 test Score 0.9223502499364569
depth =  11 Train Score 0.9611603809290004 test Score 0.9234376980605906
depth =  15 Train Score 0.9654200741217189 test Score 0.9248493816721276
depth =  20 Train Score 0.9646435728600058 test Score 0.924294359104566
depth =  35 Train Score 0.9643719065734037 test Score 0.9242683933219069
depth =  50 Train Score 0.9643719065734037 test Score 0.9242683933219069
depth =  70 Train Score 0.9643719065734037 test Score 0.9242683933219069
depth =  130 Train Score 0.9643719065734037 test Score 0.9242683933219069
```



In [15]:

```
from sklearn.metrics import f1_score
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import f1_score
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint as sp_randint
from scipy.stats import uniform

param_dist = {"n_estimators":sp_randint(105,125),
              "max_depth": sp_randint(10,15),
              "min_samples_split": sp_randint(110,190),
              "min_samples_leaf": sp_randint(25,65)}

clf = RandomForestClassifier(random_state=25,n_jobs=-1)

rf_random = RandomizedSearchCV(clf, param_distributions=param_dist,
                                    n_iter=5,cv=10,scoring='f1',random_state=25,return_t
rain_score=True)

rf_random.fit(df_final_train,y_train)
```

Out[15]:

```
RandomizedSearchCV(cv=10,
                   estimator=RandomForestClassifier(n_jobs=-1, random_state=25),
                   n_iter=5,
                   param_distributions={'max_depth': <scipy.stats._distn_infrastructure.r
v_frozen object at 0x7fc298e5f550>,
                                        'min_samples_leaf': <scipy.stats._distn_infrastr
ucture.rv_frozen object at 0x7fc298e5f350>,
                                        'min_samples_split': <scipy.stats._distn_infrast
ructure.rv_frozen object at 0x7fc298fa6e10>,
                                        'n_estimators': <scipy.stats._distn_infrastructu
re.rv_frozen object at 0x7fc298fd4e90>},
                   random_state=25, return_train_score=True, scoring='f1')
```

In [16]:

```
print('mean test scores',rf_random.cv_results_['mean_test_score'])
print('mean train scores',rf_random.cv_results_['mean_train_score'])
```

```
mean test scores [0.96260191 0.96286233 0.96081941 0.96272068 0.96468549]
mean train scores [0.96344597 0.96356616 0.96139864 0.96359746 0.96578533]
```

In [17]:

```
rf_random.cv_results_
```

Out[17]:

```
{'mean_fit_time': array([2.94827499, 2.63548112, 2.46571784, 2.56933882, 2.98594298]),
 'std_fit_time': array([0.39861755, 0.02335117, 0.04013886, 0.01684863, 0.03561   ]),
 'mean_score_time': array([0.11285799, 0.11243098, 0.11221573, 0.11203494, 0.11232536]),
 'std_score_time': array([0.00099884, 0.00029859, 0.00032369, 0.00033965, 0.00037569]),
 'param_max_depth': masked_array(data=[14, 12, 11, 13, 14],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'param_min_samples_leaf': masked_array(data=[51, 33, 56, 49, 28],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'param_min_samples_split': masked_array(data=[125, 138, 179, 165, 111],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'param_n_estimators': masked_array(data=[117, 109, 106, 108, 121],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'params': [{'max_depth': 14,
   'min_samples_leaf': 51,
   'min_samples_split': 125,
   'n_estimators': 117},
  {'max_depth': 12,
```

```
    'min_samples_leaf': 33,
    'min_samples_split': 138,
    'n_estimators': 109},
   {'max_depth': 11,
    'min_samples_leaf': 56,
    'min_samples_split': 179,
    'n_estimators': 106},
   {'max_depth': 13,
    'min_samples_leaf': 49,
    'min_samples_split': 165,
    'n_estimators': 108},
   {'max_depth': 14,
    'min_samples_leaf': 28,
    'min_samples_split': 111,
    'n_estimators': 121}],
 'split0_test_score': array([0.96220424, 0.96196319, 0.961818  , 0.96300082, 0.96426384])
,
 'split1_test_score': array([0.96039401, 0.95873537, 0.95764803, 0.95814049, 0.96165252])
,
 'split2_test_score': array([0.96285714, 0.96271325, 0.96123555, 0.96325781, 0.96432939])
,
 'split3_test_score': array([0.96276541, 0.96290223, 0.96008209, 0.96306177, 0.96481898])
,
 'split4_test_score': array([0.96290984, 0.96361406, 0.96253071, 0.96367543, 0.96486928])
,
 'split5_test_score': array([0.96446804, 0.96499643, 0.96212121, 0.96514582, 0.96642515])
,
 'split6_test_score': array([0.95982783, 0.96268275, 0.95893224, 0.96201495, 0.96281161])
,
 'split7_test_score': array([0.96264544, 0.96259096, 0.96029472, 0.96385542, 0.96629671])
,
 'split8_test_score': array([0.96494635, 0.96544695, 0.96232772, 0.96401264, 0.96699939])
,
 'split9_test_score': array([0.96300082, 0.96297811, 0.96120381, 0.96104162, 0.96438805])
,
 'mean_test_score': array([0.96260191, 0.96286233, 0.96081941, 0.96272068, 0.96468549]),
 'std_test_score': array([0.00148509, 0.00172578, 0.00150514, 0.00185818, 0.00155343]),
 'rank_test_score': array([4, 2, 5, 3, 1], dtype=int32),
 'split0_train_score': array([0.96393525, 0.96353054, 0.96285932, 0.9645287 , 0.96650891]
),
 'split1_train_score': array([0.96468372, 0.96289397, 0.96209144, 0.96294613, 0.96622371]
),
 'split2_train_score': array([0.96345341, 0.96349966, 0.96104192, 0.96381601, 0.96518011]
),
 'split3_train_score': array([0.9640739 , 0.96397123, 0.96099702, 0.96375815, 0.96673847]
),
 'split4_train_score': array([0.96281039, 0.96320629, 0.96201985, 0.96361486, 0.96495828]
),
 'split5_train_score': array([0.96337196, 0.96421234, 0.96117533, 0.96445597, 0.96564803]
),
 'split6_train_score': array([0.96146479, 0.96419315, 0.96025394, 0.96358276, 0.96482995]
),
 'split7_train_score': array([0.96358849, 0.96252978, 0.96081888, 0.96316907, 0.96663342]
),
 'split8_train_score': array([0.96283988, 0.9635645 , 0.960417  , 0.96280565, 0.96553679]
),
 'split9_train_score': array([0.96423787, 0.96406016, 0.96231173, 0.96329733, 0.9655956 ]
),
 'mean_train_score': array([0.96344597, 0.96356616, 0.96139864, 0.96359746, 0.96578533]),
 'std_train_score': array([0.00086689, 0.00053665, 0.00082351, 0.00054801, 0.00066507])}
```

In [18]:

```
print(rf_random.best_estimator_)
```

```
RandomForestClassifier(max_depth=14, min_samples_leaf=28, min_samples_split=111,
                       n_estimators=121, n_jobs=-1, random_state=25)
```

In [19]:

```
clf = RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
```

```
            max_depth=14, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=28, min_samples_split=111,
            min_weight_fraction_leaf=0.0, n_estimators=121, n_jobs=-1,
            oob_score=False, random_state=25, verbose=0, warm_start=False)
```

In [20]:

```
clf.fit(df_final_train,y_train)
y_train_pred = clf.predict(df_final_train)
y_test_pred = clf.predict(df_final_test)
```

In [21]:

```
from sklearn.metrics import f1_score
print('Train f1 score',f1_score(y_train,y_train_pred))
print('Test f1 score',f1_score(y_test,y_test_pred))
```

```
Train f1 score 0.9659754255623137
Test f1 score 0.9249371129008392
```

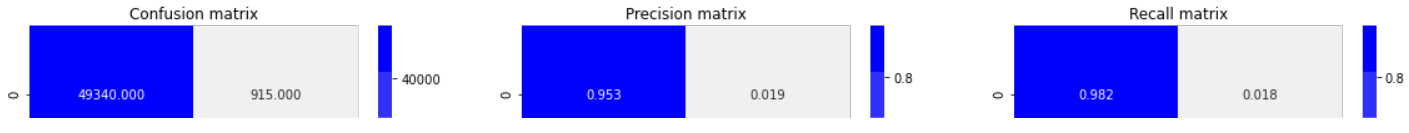
In [22]:

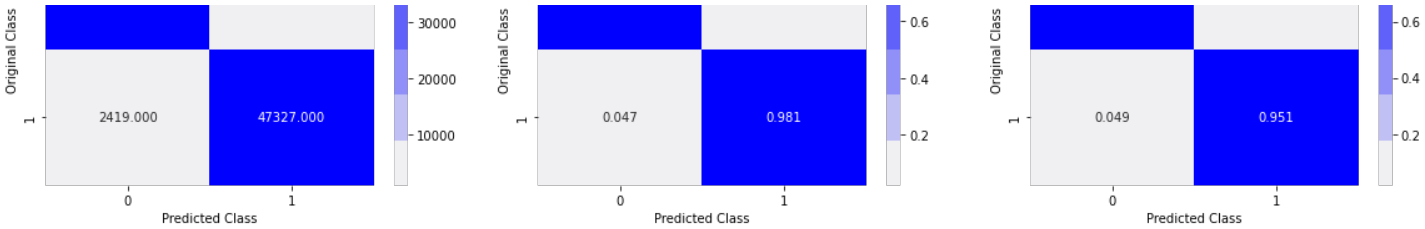
```
from sklearn.metrics import confusion_matrix
def plot_confusion_matrix(test_y, predict_y):
    C = confusion_matrix(test_y, predict_y)

    A =(((C.T)/(C.sum(axis=1))).T)

    B =(C/C.sum(axis=0))
    plt.figure(figsize=(20,4))

    labels = [0,1]
    # representing A in heatmap format
    cmap=sns.light_palette("blue")
    plt.subplot(1, 3, 1)
    sns.heatmap(C, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=lab
els)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Confusion matrix")

    plt.subplot(1, 3, 2)
    sns.heatmap(B, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=lab
els)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Precision matrix")

    plt.subplot(1, 3, 3)
    # representing B in heatmap format
    sns.heatmap(A, annot=True, cmap=cmap, fmt=".3f", xticklabels=labels, yticklabels=lab
els)
    plt.xlabel('Predicted Class')
    plt.ylabel('Original Class')
    plt.title("Recall matrix")

    plt.show()
```

In [23]:

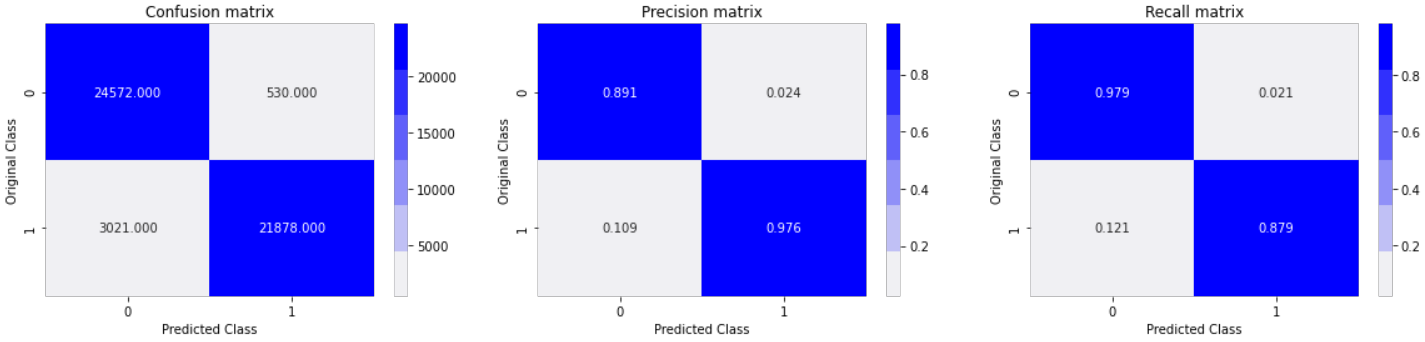
```
print('Train confusion_matrix')
plot_confusion_matrix(y_train,y_train_pred)
print('Test confusion_matrix')
plot_confusion_matrix(y_test,y_test_pred)
```

```
Train confusion_matrix
```



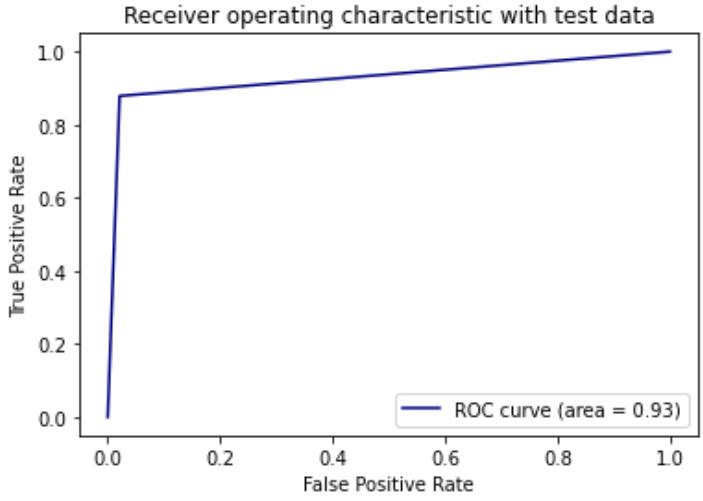


Test confusion_matrix



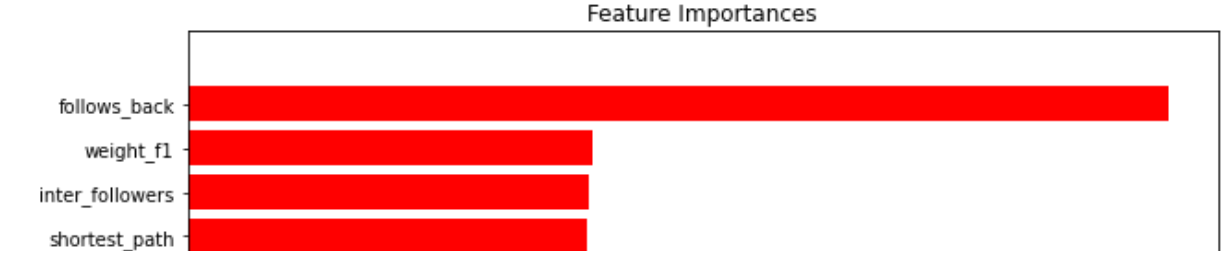
In [24]:

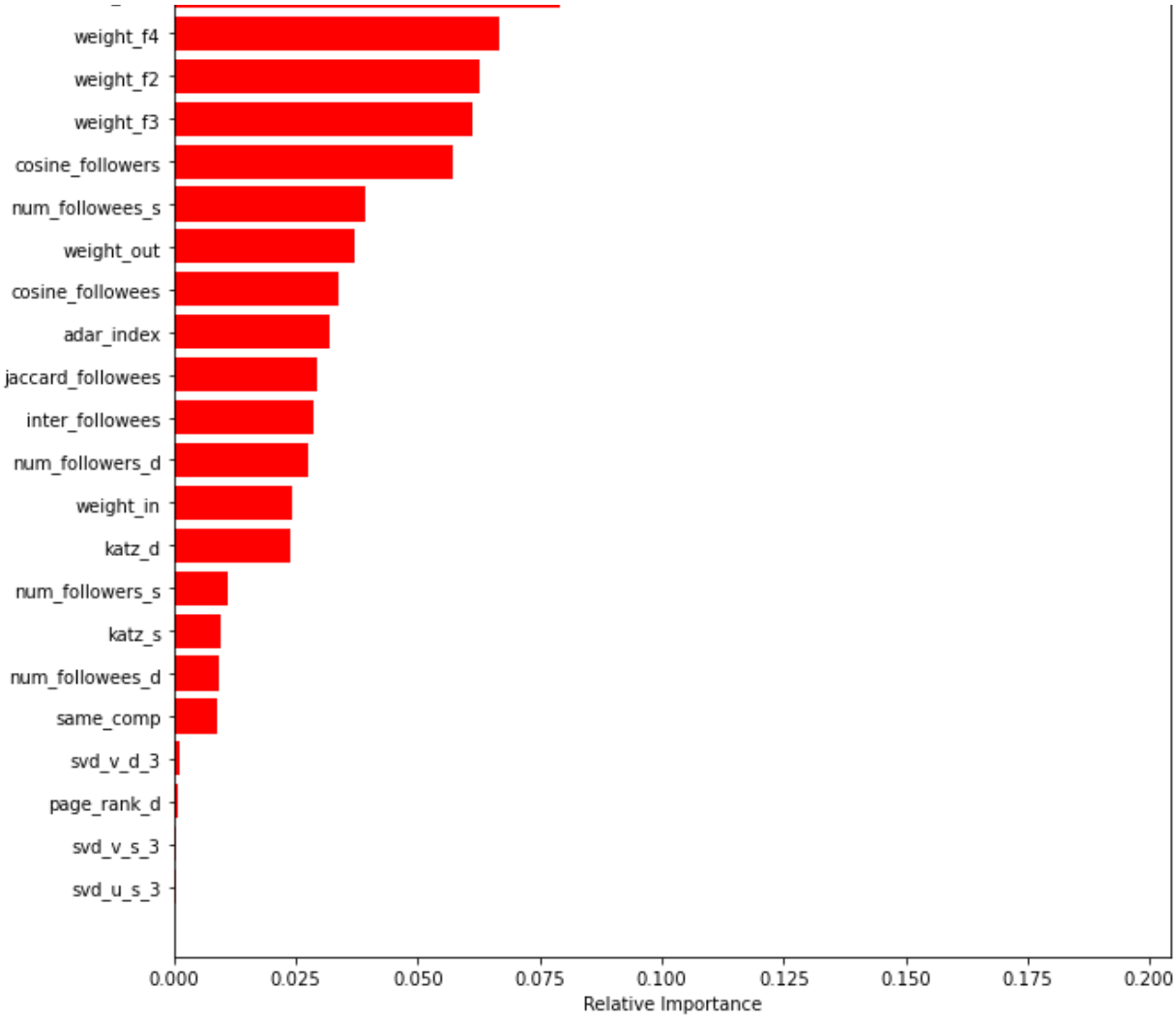
```
from sklearn.metrics import roc_curve, auc
fpr,tpr,ths = roc_curve(y_test,y_test_pred)
auc_sc = auc(fpr, tpr)
plt.plot(fpr, tpr, color='navy',label='ROC curve (area = %0.2f)' % auc_sc)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic with test data')
plt.legend()
plt.show()
```



In [25]:

```
features = df_final_train.columns
importances = clf.feature_importances_
indices = (np.argsort(importances))[-25:]
plt.figure(figsize=(10,12))
plt.title('Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='r', align='center')
plt.yticks(range(len(indices)), [features[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```





Assignments:

1. **Add another feature called Preferential Attachment with followers and followees data of vertex. you can check about Preferential Attachment in below link http://be.amazd.com/link-prediction/**
2. **Add feature called svd_dot. you can calculate svd_dot as Dot product between sourse node svd and destination node svd features. you can read about this in below pdf https://storage.googleapis.com/kaggle-forum-message-attachments/2594/supervised_link_prediction.pdf**
3. **Tune hyperparameters for XG boost with all these features and check the error metric.**

In [28]:

```
df_final_train = read_hdf('storage_sample_stage5.h5', 'train_df',mode='r')
df_final_test = read_hdf('storage_sample_stage5.h5', 'test_df',mode='r')
```

In [29]:

```
df_final_train.head(2)
```

Out[29]:

	source_node	destination_node	indicator_link	jaccard_followers	jaccard_followees	cosine_followers	cosine_followees	num
0	273084	1505602	1	0	0.000000	0.000000	0.000000	
1	1677755	298631	1	0	0.266667	0.192847	0.478091	

2 rows × 58 columns

In [30]:

```
y_train = df_final_train.indicator_link
y_test = df_final_test.indicator_link
```

In [31]:

```
df_final_train.drop(['source_node', 'destination_node','indicator_link'],axis=1,inplace=True)
df_final_test.drop(['source_node', 'destination_node','indicator_link'],axis=1,inplace=True)
```

Applying XGBOOST

In [46]:

```
import xgboost as xgb
clf = xgb.XGBClassifier()
param_dist = {"n_estimators":sp_randint(105,125),
              "max_depth": sp_randint(10,15)
              }
xgb_random = RandomizedSearchCV(clf, param_distributions=param_dist,n_iter=5,cv=10,scoring='f1',random_state=25,return_train_score=True)


xgb_random.fit(df_final_train,y_train)
```

```
[09:11:50] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:12:11] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:12:33] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:12:54] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:13:15] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:13:36] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:13:58] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:14:19] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:14:40] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:15:02] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:15:24] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:15:47] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:16:11] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:16:33] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:16:55] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'.
```

tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:17:17] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:17:39] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:18:01] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:18:23] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:18:45] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:19:08] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:19:26] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:19:44] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:20:02] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:20:19] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:20:37] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:20:56] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:21:14] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:21:31] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:21:49] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:22:07] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:22:25] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:22:44] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:23:02] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:23:20] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:23:38] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:23:57] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:24:16] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:24:34] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos

```
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:24:52] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:25:11] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:25:30] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:25:49] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:26:08] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:26:28] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:26:47] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:27:06] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:27:26] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:27:45] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:28:04] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
[09:28:23] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
```

Out[46]:

```
RandomizedSearchCV(cv=10,
                   estimator=XGBClassifier(base_score=None, booster=None,
                                           colsample_bylevel=None,
                                           colsample_bynode=None,
                                           colsample_bytree=None, gamma=None,
                                           gpu_id=None, importance_type='gain',
                                           interaction_constraints=None,
                                           learning_rate=None,
                                           max_delta_step=None, max_depth=None,
                                           min_child_weight=None, missing=nan,
                                           monotone_constraints=None,
                                           n_estimators=100...
                                           reg_lambda=None,
                                           scale_pos_weight=None,
                                           subsample=None, tree_method=None,
                                           validate_parameters=None,
                                           verbosity=None),
                   n_iter=5,
                   param_distributions={'max_depth': <scipy.stats._distn_infrastructure.r
v_frozen object at 0x7fc298dbe550>,
                                        'n_estimators': <scipy.stats._distn_infrastructu
re.rv_frozen object at 0x7fc298c53dd0>},
                   random_state=25, return_train_score=True, scoring='f1')
```

In [47]:

```
xgb_random.cv_results_
```

Out[47]:

```
{'mean_fit_time': array([21.2994365 , 22.21777775, 17.84682932, 18.23336666, 19.15721333]
),
```

```
 'std_fit_time': array([0.40883211, 0.68235166, 0.52035163, 0.30952166, 0.36527094]),
 'mean_score_time': array([0.01583848, 0.01599092, 0.01440217, 0.01476681, 0.01508167]),
 'std_score_time': array([0.00090847, 0.00073037, 0.0010548 , 0.00053449, 0.00077016]),
 'param_max_depth': masked_array(data=[14, 14, 10, 11, 11],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'param_n_estimators': masked_array(data=[120, 123, 109, 110, 112],
              mask=[False, False, False, False, False],
        fill_value='?',
             dtype=object),
 'params': [{'max_depth': 14, 'n_estimators': 120},
  {'max_depth': 14, 'n_estimators': 123},
  {'max_depth': 10, 'n_estimators': 109},
  {'max_depth': 11, 'n_estimators': 110},
  {'max_depth': 11, 'n_estimators': 112}],
 'split0_test_score': array([0.98099859, 0.98120453, 0.98333165, 0.98323571, 0.98293102])
,
 'split1_test_score': array([0.98156402, 0.98166346, 0.98139159, 0.98209408, 0.98199838])
,
 'split2_test_score': array([0.98039612, 0.98039216, 0.98081583, 0.98182553, 0.98202383])
,
 'split3_test_score': array([0.98076144, 0.98086464, 0.98249874, 0.98086851, 0.98066214])
,
 'split4_test_score': array([0.98181083, 0.98222222, 0.98253055, 0.98253407, 0.98263328])
,
 'split5_test_score': array([0.98403718, 0.98413661, 0.98382858, 0.98343434, 0.98363636])
,
 'split6_test_score': array([0.98026915, 0.98026915, 0.98199838, 0.97956293, 0.97956706])
,
 'split7_test_score': array([0.98191006, 0.98191006, 0.98292757, 0.98364627, 0.9832391 ])
,
 'split8_test_score': array([0.98323571, 0.98303373, 0.98445073, 0.98292757, 0.98322894])
,
 'split9_test_score': array([0.9812772 , 0.98148336, 0.98139159, 0.98058252, 0.9805786 ])
,
 'mean_test_score': array([0.98162603, 0.98171799, 0.98251652, 0.98207115, 0.98204987]),
 'std_test_score': array([0.0011463 , 0.00113044, 0.00109491, 0.00129064, 0.00129214]),
 'rank_test_score': array([5, 4, 1, 2, 3], dtype=int32),
 'split0_train_score': array([1., 1., 1., 1., 1.]),
 'split1_train_score': array([1., 1., 1., 1., 1.]),
 'split2_train_score': array([1., 1., 1., 1., 1.]),
 'split3_train_score': array([1., 1., 1., 1., 1.]),
 'split4_train_score': array([1., 1., 1., 1., 1.]),
 'split5_train_score': array([1., 1., 1., 1., 1.]),
 'split6_train_score': array([1., 1., 1., 1., 1.]),
 'split7_train_score': array([1., 1., 1., 1., 1.]),
 'split8_train_score': array([1., 1., 1., 1., 1.]),
 'split9_train_score': array([1., 1., 1., 1., 1.]),
 'mean_train_score': array([1., 1., 1., 1., 1.]),
 'std_train_score': array([0., 0., 0., 0., 0.])}
```

In [48]:

```
print('mean test scores',xgb_random.cv_results_['mean_test_score'])
print('mean train scores',xgb_random.cv_results_['mean_train_score'])
```

```
mean test scores [0.98162603 0.98171799 0.98251652 0.98207115 0.98204987]
mean train scores [1. 1. 1. 1. 1.]
```

In [49]:

```
print(xgb_random.best_estimator_)
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=10,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=109, n_jobs=16, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

In [50]:

```
clf = xgb.XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=10,
              min_child_weight=1,monotone_constraints='()',
              n_estimators=109, n_jobs=16, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

In [51]:

```
clf.fit(df_final_train,y_train)
y_train_pred = clf.predict(df_final_train)
y_test_pred = clf.predict(df_final_test)
```

```
[09:30:24] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evalua
tion metric used with the objective 'binary:logistic' was changed from 'error' to 'loglos
s'. Explicitly set eval_metric if you'd like to restore the old behavior.
```

In [52]:

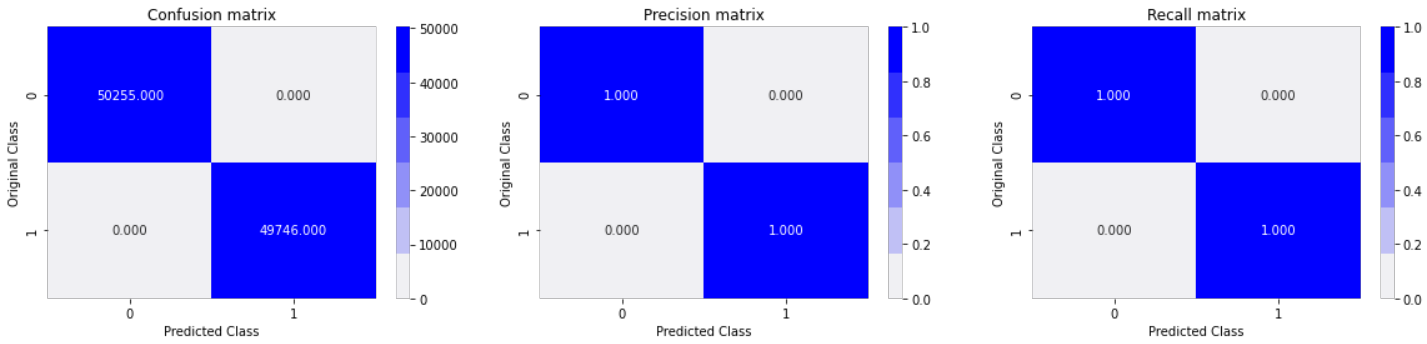
```
from sklearn.metrics import f1_score
print('Train f1 score',f1_score(y_train,y_train_pred))
print('Test f1 score',f1_score(y_test,y_test_pred))
```

```
Train f1 score 1.0
Test f1 score 0.9252414145935922
```

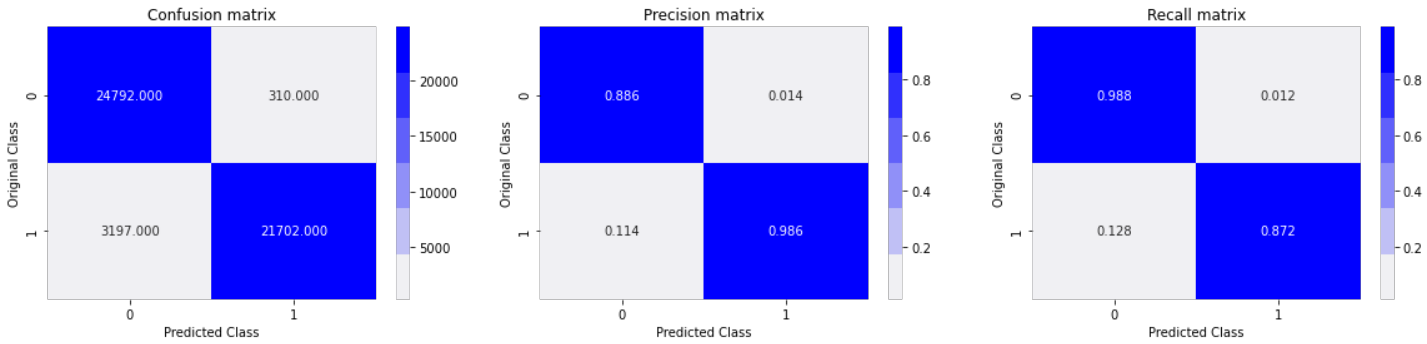
In [53]:

```
print('Train confusion_matrix')
plot_confusion_matrix(y_train,y_train_pred)
print('Test confusion_matrix')
plot_confusion_matrix(y_test,y_test_pred)
```

```
Train confusion_matrix
```



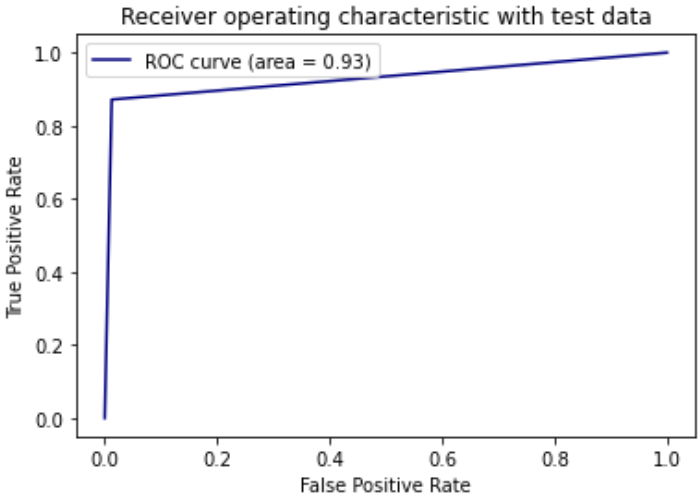

```
Test confusion_matrix
```



In [54]:

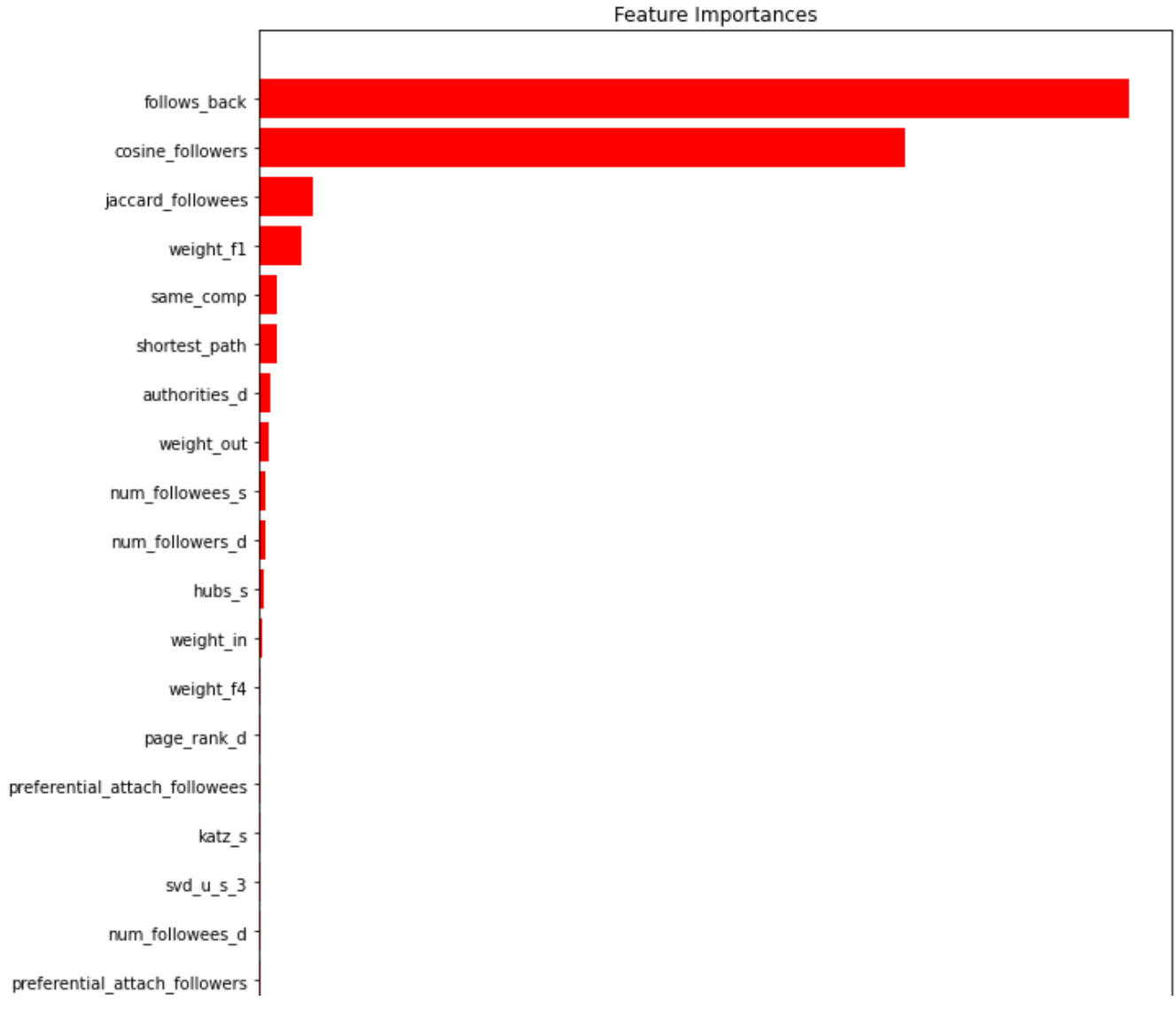
```
from sklearn.metrics import roc_curve, auc
fpr,tpr,ths = roc_curve(y_test,y_test_pred)
auc_sc = auc(fpr, tpr)
plt.plot(fpr, tpr, color='navy',label='ROC curve (area = %0.2f)' % auc_sc)
```

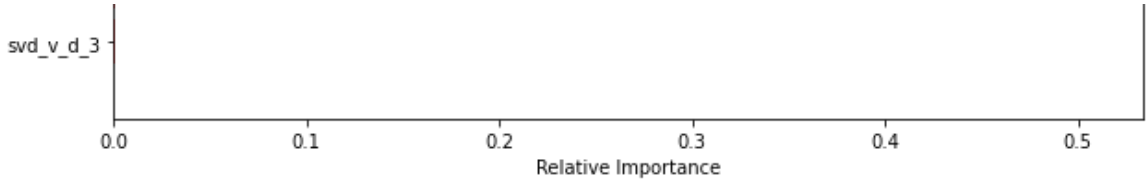
```
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic with test data')
plt.legend()
plt.show()
```



In [55]:

```
features = df_final_train.columns
importances = clf.feature_importances_
indices = (np.argsort(importances))[-20:]
plt.figure(figsize=(10,12))
plt.title('Feature Importances')
plt.barh(range(len(indices)), importances[indices], color='r', align='center')
plt.yticks(range(len(indices)), [features[i] for i in indices])
plt.xlabel('Relative Importance')
plt.show()
```





In [56]:

```
from prettytable import PrettyTable
ptable = PrettyTable()
ptable.field_names = ["Model", "n_estimators", "max_depth", "Train F1 Score","Test F1 Sc
ore"]
ptable.add_row(['RandomForest','121','14','0.9659','0.9249'])
ptable.add_row(['XGBOOST','109','10','0.9999','0.9252'])
print(ptable)
```

```
+--------------+--------------+-----------+----------------+---------------+
|    Model     | n_estimators | max_depth | Train F1 Score | Test F1 Score |
+--------------+--------------+-----------+----------------+---------------+
| RandomForest |     121      |     14    |     0.9659     |     0.9249    |
|   XGBOOST    |     109      |     10    |     0.9999     |     0.9252    |
+--------------+--------------+-----------+----------------+---------------+
```

In [ ]: