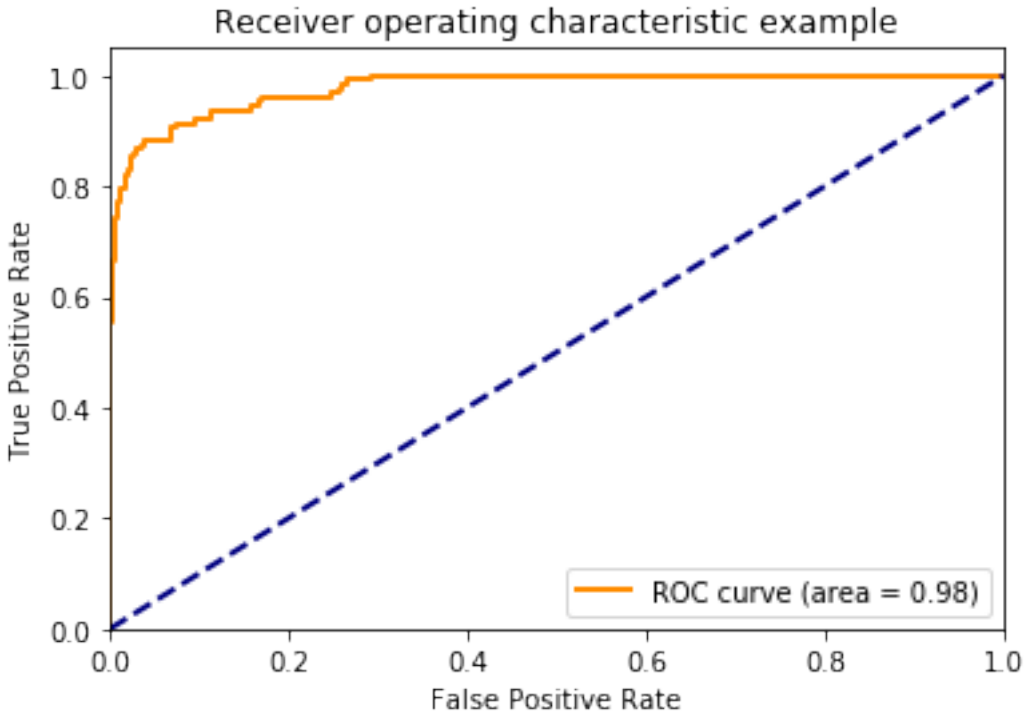
MuskClassify

October 19, 2018

In [2]:
```
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt

#load data
from sklearn.linear_model import LogisticRegression
import numpy as np
data = np.loadtxt('data.csv',delimiter = ',')
Y = data[:,0:1]
#Y = column_or_1d(Y,warn=True)
X = data[:,1:data.shape[1]]

import timeit
start = timeit.default_timer()
clf =LogisticRegression(random_state=0, solver="liblinear").fit(X, Y)
elapsed = (timeit.default_timer() - start)

#load test
test_data = np.loadtxt('test.csv',delimiter = ',')
test = test_data[:,1:test_data.shape[1]]
test_Y = test_data[:,0:1]
test_Y = test_Y.transpose()[0]
clf.predict(test)
fpr = dict()
tpr = dict()
roc_auc = dict()
y_score = clf.fit(X,Y).decision_function(test)
fpr, tpr,threshold = roc_curve(test_Y, y_score)
roc_auc = auc(fpr, tpr)


# Plot ROC
plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
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 example')
plt.legend(loc="lower right")
plt.show()
#calculate score
from sklearn.metrics import roc_auc_score
score = roc_auc_score(test_Y, y_score)
print("examing the test set, score is")
print(score)

#regression and record training time
print("time needed for training is")
print(elapsed)
```

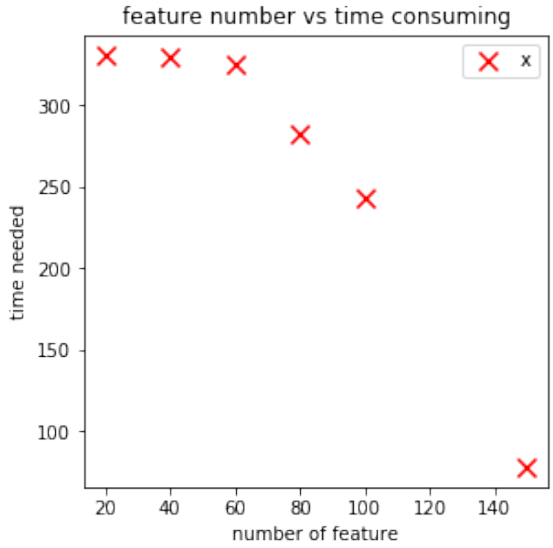
```
F:\software\1PYTHON\anaconda\lib\site-packages\sklearn\utils\validation.py:578: DataConversionWa
  y = column_or_1d(y, warn=True)
```

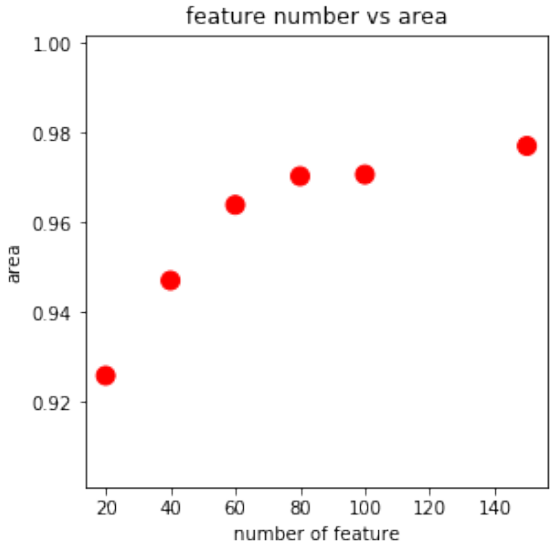



```
examing the test set, score is
0.9775851267688679
time needed for training is
5.332571918898696
```

In [22]:
```
from sklearn.feature_selection import RFE
estimator = LogisticRegression(random_state=0, solver="liblinear")
FeatureRange=[20,40,60,80,100,150]
#for test FeatureRange=[20]
Areas=[]
UsingTime=[]
for i in FeatureRange:
    selector = RFE(estimator, i, step=1)
    import timeit
    start = timeit.default_timer()
    selector = selector.fit(X, Y)
    UsingTime.append(timeit.default_timer() - start)
    selector.predict(test)
    y_score = selector.fit(X,Y).decision_function(test)
    from sklearn.metrics import roc_auc_score
    Areas.append(roc_auc_score(test_Y, y_score))

import matplotlib.pyplot as plt
```

In [25]:
```
fig = plt.figure(figsize=(10, 10))
ax1 = fig.add_subplot(222)
ax1.set_title('feature number vs area')
plt.xlabel('number of feature')
plt.ylabel('area')
lValue = Areas*10
ax1.scatter(FeatureRange,Areas,c='r',s= 100,linewidths=lValue,marker='o')


ax2 = fig.add_subplot(221)
ax2.set_title('feature number vs time consuming')
plt.xlabel('number of feature')
plt.ylabel('time needed')
lValue = Areas*10
ax2.scatter(FeatureRange,UsingTime,c='r',s= 100,linewidths=lValue,marker='x')

plt.legend('x1')

plt.show()
```


feature number vs time consuming
x
time needed
300
250
200
150
100
20
40
60
80
100
120
140
number of feature


feature number vs area
area
1.00
0.98
0.96
0.94
0.92
20
40
60
80
100
120
140
number of feature