Research

January 29, 2018

In [577]:
```
from sklearn import linear_model
from sklearn import preprocessing
from sklearn import metrics
from sklearn.model_selection import train_test_split
import pandas as pd

import matplotlib.pyplot as plt
```

In [578]:
```
data = pd.read_csv('./data/train.csv')

data.columns
```

Out[578]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [579]:
```
ages = data[data.Age.notnull()].Age
ages.mean()
```

Out[579]:
```
29.69911764705882
```

In [580]:
```
features = []

# Sex
array, levels = pd.factorize(data.Sex)
data['factorized_sex'] = array
features.append('factorized_sex')

# Age
data['known_age'] = data.Age.notnull().astype(int)
features.append('known_age')
for age in [10, 20, 30, 40, 50, 60, 70]:
    name = 'more_{}_years'.format(age)
    data[name] = (data['Age'] >= age).astype(int)
    features.append(name)

levels
```

Out[580]:
```
Index(['male', 'female'], dtype='object')
```

In [581]:
```
X = data[features]
y = data.Survived
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=
```

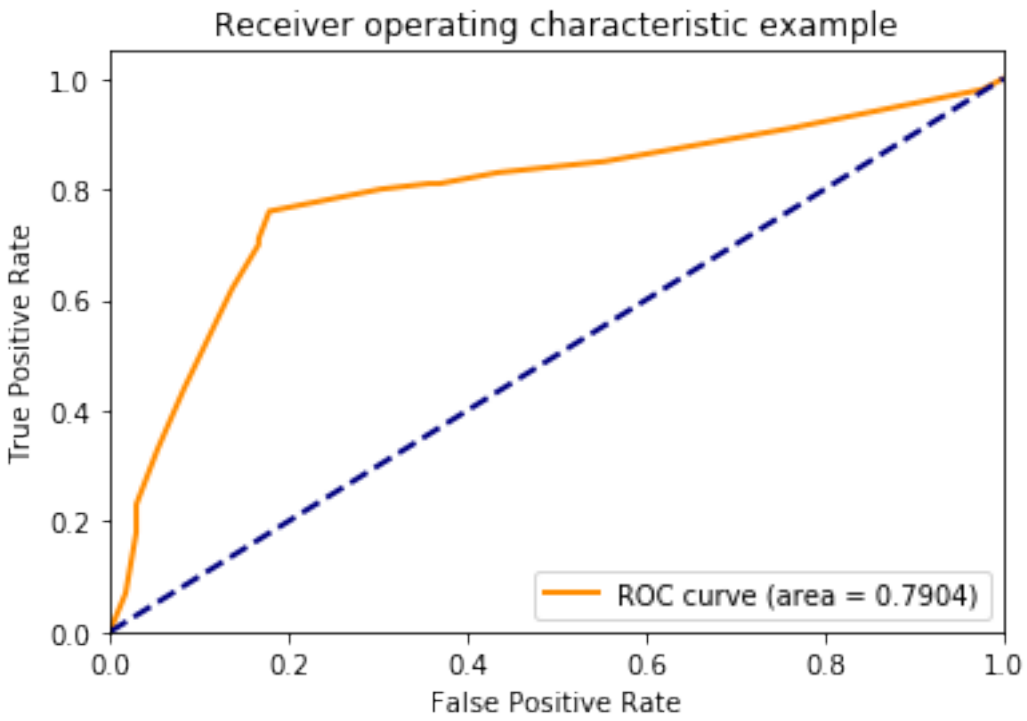
In [582]:
```
reg = linear_model.LinearRegression()
reg.fit(X_train, y_train)

sorted_features = list(sorted(zip(features, reg.coef_), key=lambda x: -abs(x[1])))
sorted_features
```

Out[582]:
```
[('factorized_sex', 0.5502789977837783),
 ('known_age', 0.2093346228418844),
 ('more_10_years', -0.1450545051890952),
 ('more_60_years', -0.09503559880384975),
 ('more_70_years', 0.08756974944594441),
 ('more_30_years', 0.06335049071641767),
 ('more_40_years', -0.05404286488022054),
 ('more_50_years', 0.012385558391905296),
 ('more_20_years', -0.0023821639994232213)]
```

In [583]:
```
y_predicted = reg.predict(X_test)
```

In [584]:
```
fpr, tpr, _ = metrics.roc_curve(y_test, y_predicted)
roc_auc = metrics.auc(fpr, tpr)
```

In [585]:
```
plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.4f)' % 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()
```



In [586]:
```
# Fare
min_max_scaler = preprocessing.MinMaxScaler()
data['scaled_fare'] = min_max_scaler.fit_transform(data[['Fare']])
features.append('scaled_fare')

# Class
for cl_num in [1, 3]:
    name = 'class{}'.format(cl_num)
    data[name] = (data['Pclass'] == cl_num).astype(int)
    features.append(name)

for sp in [1,2,3,4,5]:
    name = 'sib_sp_{}'.format(sp)
    data[name] = (data.SibSp == sp).astype(int)
    features.append(name)

for emb in ['C', 'Q', 'S']:
    name = 'embarked{}'.format(emb)
    data[name] = (data.Embarked == emb).astype(int)
    features.append(name)
```

In [587]:
```
X = data[features]
y = data.Survived
```

```
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=
```

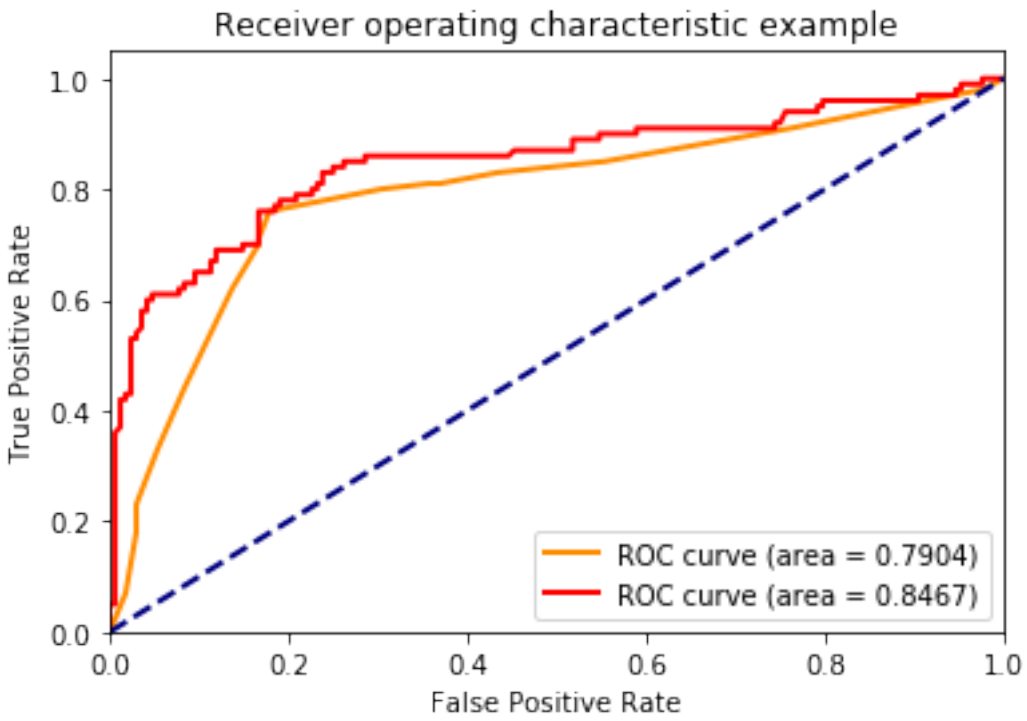
In [588]:
```
reg = linear_model.LinearRegression()
reg.fit(X_train, y_train)

sorted_features = list(sorted(zip(features, reg.coef_), key=lambda x: -abs(x[1])))
sorted_features
```

Out[588]:
```
[('sib_sp_4', -0.6094350492631067),
 ('factorized_sex', 0.4897021703624261),
 ('sib_sp_3', -0.44746407743397243),
 ('known_age', 0.44067846360381696),
 ('more_10_years', -0.3879381213695838),
 ('sib_sp_5', -0.36110806590807537),
 ('embarkedS', -0.21615587334465106),
 ('class3', -0.15833660008320938),
 ('class1', 0.14896855700336806),
 ('embarkedC', -0.14080616102548948),
 ('more_60_years', -0.13098754315111358),
 ('scaled_fare', 0.11314638777258867),
 ('sib_sp_2', -0.09535474620752135),
 ('embarkedQ', -0.08750071880558272),
 ('more_40_years', -0.07638274488641952),
 ('more_70_years', 0.07037725332428009),
 ('more_50_years', -0.05181982988051265),
 ('sib_sp_1', 0.02771502503090463),
 ('more_20_years', 0.005643842599454038),
 ('more_30_years', 0.0009953268513427094)]
```

In [589]:
```
y_predicted = reg.predict(X_test)
```

In [590]:
```
fpr_full, tpr_full, _ = metrics.roc_curve(y_test, y_predicted)
roc_auc_full = metrics.auc(fpr_full, tpr_full)
```

In [591]:
```
plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.4f)' % roc_auc)
plt.plot(fpr_full, tpr_full, color='red',
         lw=lw, label='ROC curve (area = %0.4f)' % roc_auc_full)
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()
```




```
In [592]: features_part = list(map(lambda x: x[0], sorted_features[:9]))
          features_part

Out[592]: ['sib_sp_4',
           'factorized_sex',
           'sib_sp_3',
           'known_age',
           'more_10_years',
           'sib_sp_5',
           'embarkedS',
           'class3',
           'class1']

In [593]: X = data[features_part]
          y = data.Survived
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=

In [594]: reg = linear_model.LinearRegression()
          reg.fit(X_train, y_train)

          reg.coef_

Out[594]: array([-0.59219715,  0.50413907, -0.4377684 ,  0.42059545, -0.39268102,
                 -0.34329523, -0.09384483, -0.15200828,  0.13527439])
```

In [595]:
```
y_predicted = reg.predict(X_test)
```

In [596]:
```
fpr_part, tpr_part, _ = metrics.roc_curve(y_test, y_predicted)
roc_auc_part = metrics.auc(fpr_part, tpr_part)
```

In [597]:
```
plt.figure()
lw = 2
plt.plot(fpr, tpr, color='darkorange',
         lw=lw, label='ROC curve (area = %0.4f)' % roc_auc)
plt.plot(fpr_full, tpr_full, color='red',
         lw=lw, label='ROC curve (area = %0.4f)' % roc_auc_full)
plt.plot(fpr_part, tpr_part, color='green',
         lw=lw, label='ROC curve (area = %0.4f)' % roc_auc_part)
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()
```