# -\*- coding: utf-8 -\*-

"""

Created on Sun Sep 9 17:36:36 2018

@author: James

Run experiments to predict player position from stats

"""

import process\_nfl\_data as nfl

import numpy as np

from sklearn.svm import SVC

from sklearn.metrics import confusion\_matrix, accuracy\_score

combine\_stats = nfl.load\_nflcombine\_data()

df = combine\_stats[['Pos','Height','Weight', '40YD', 'Vertical', 'BenchReps', 'Broad Jump','3Cone', 'Shuttle']]

df\_train, df\_test = nfl.split\_test\_training\_sets(df)

"""

summary\_stats(df\_train,'Pos')

cam = combine\_stats[combine\_stats['Player'].str.contains('Cam Newton')]

display\_scatter\_pairs(combine\_stats,['Height','Weight', '40YD'],cam)

"""

"""

# Run decision tree, find best parameters

DT = nfl.decision\_tree(df\_train,'Pos')

params = {"min\_samples\_leaf":range(1,40,1),"criterion":["gini","entropy"]}

df\_cv = nfl.grid\_search(df\_train,DT,'Pos',params,save=True)

nfl.display\_parameter\_curve(df\_cv,'param\_min\_samples\_leaf','mean\_test\_score',

'Min Samples Leaf','Mean Test Score','param\_criterion')

nfl.display\_parameter\_curve(df\_cv,'param\_min\_samples\_leaf','mean\_train\_score',

'Min Sample Leaf','Mean Train Score','param\_criterion')

"""

"""

# Run support vector machine, find best parameters

SVM = nfl.svm(df\_train,'Pos')

params = {"C":np.power(10.0,np.arange(-6,5,1)),"kernel":["linear","rbf"]}

df\_cv=nfl.grid\_search(df\_train,SVM,'Pos',params,save=True)

nfl.display\_parameter\_curve(df\_cv,'param\_C','mean\_test\_score',

'Penalty Param','Mean Test Score','param\_kernel',True)

nfl.display\_parameter\_curve(df\_cv,'param\_C','mean\_train\_score',

'Penalty Param','Mean Train Score','param\_kernel',True)

"""

"""

# Run random forest, find best parameters

RF = nfl.rf(df\_train,'Pos')

params = {"n\_estimators":range(1,420,20),"criterion":["gini","entropy"]}

df\_cv = nfl.grid\_search(df\_train,RF,'Pos',params,save=True)

nfl.display\_parameter\_curve(df\_cv,'param\_n\_estimators','mean\_test\_score',

'# Estimators','Mean Test Score','param\_criterion')

"""

# Run grid search multiple times for RT n\_estimators

RF = nfl.rf(df\_train,'Pos')

params = {"n\_estimators":range(1,420,20),"criterion":["gini","entropy"]}

for i in range(0,10):

print(i)

try:

df\_cv = df\_cv.append(nfl.grid\_search(df\_train,RF,'Pos',params,save=False))

except:

df\_cv = nfl.grid\_search(df\_train,RF,'Pos',params,save=False)

"""

df\_cv = df\_cv.groupby(['param\_n\_estimators']).mean().reset\_index()

nfl.display\_parameter\_curve(df\_cv,'param\_n\_estimators','mean\_test\_score',

'# Estimators','Mean Test Score','param\_criterion')

nfl.display\_parameter\_curve(df\_cv,'param\_n\_estimators','mean\_train\_score',

'# Estimators','Mean Train Score','param\_criterion')

"""

"""

# Run grid search multiple times for max features

RF = nfl.rf(df\_train,'Pos')

params = {"n\_estimators":[220],"criterion":["gini"],"max\_features":range(1,df\_train.shape[1]-1,1)}

for i in range(0,10):

print(i)

try:

df\_cv = df\_cv.append(nfl.grid\_search(df\_train,RF,'Pos',params,save=False))

except:

df\_cv = nfl.grid\_search(df\_train,RF,'Pos',params,save=False)

df\_cv = df\_cv.groupby(['param\_max\_features']).mean().reset\_index()

nfl.display\_parameter\_curve(df\_cv,'param\_max\_features','mean\_test\_score',

'n\_estimators','Mean Test Score')

nfl.display\_parameter\_curve(df\_cv,'param\_max\_features','mean\_train\_score',

'n\_estimators','Mean Train Score')

"""

"""

SVM = SVC(C=10,kernel="linear")

SVM.fit(df\_train.ix[:,1:],df\_train.ix[:,0])

y\_test = df\_test.ix[:,0]

y\_pred = SVM.predict(df\_test.ix[:,1:])

print(accuracy\_score(y\_pred,y\_test))

labels = sorted(df['Pos'].unique())

cm = confusion\_matrix(y\_test,y\_pred,labels=labels)

nfl.save\_confusion\_matrix(cm,labels,'cm')

#pos\_prob = SVM.predict\_proba(df\_test.ix[:,1:])

"""