Multiclass_classification

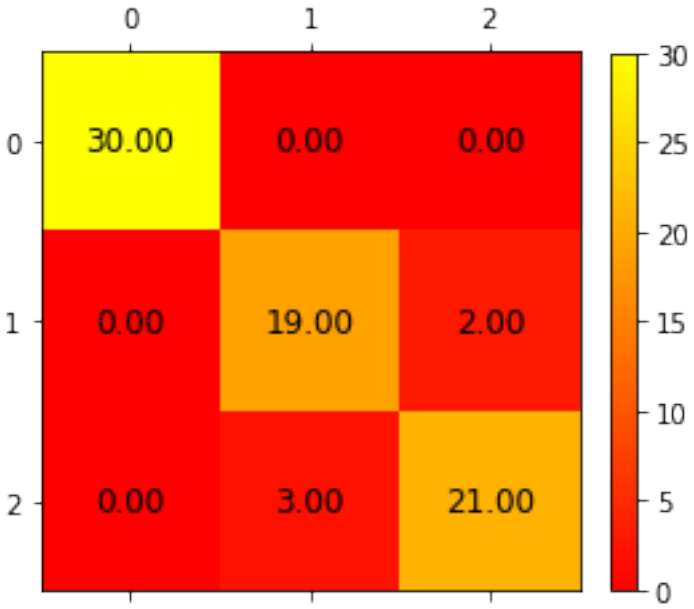
January 27, 2019

In [3]:
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
iris = datasets.load_iris()
X_train, X_test, Y_train, Y_test = train_test_split(iris.data, iris.target, test_size=(
classifier = DecisionTreeClassifier(max_depth=2) # low efficiency classifier
classifier.fit(X_train, Y_train)
Y_pred = classifier.predict(X_test)
```

In [4]:
```
# confuxion matrix
from sklearn import metrics
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(Y_test, Y_pred)
print(cm)
```

```
[[30  0  0]
 [ 0 19  3]
 [ 0  2 21]]
```

In [6]:
```
import matplotlib.pyplot as plt
img = plt.matshow(cm, cmap=plt.cm.autumn)
plt.colorbar(img, fraction=0.045)
for x in range(cm.shape[0]):
    for y in range(cm.shape[1]):
        plt.text(x, y, "%0.2f" % cm[x, y], size=12, color='black', ha='center', va='cen
plt.show()
```




```
In [7]: # accuracy
        metrics.accuracy_score(Y_test, Y_pred)

Out[7]: 0.9333333333333333

In [11]: # precision 1
         metrics.precision_score(Y_test, Y_pred, average='weighted')

Out[11]: 0.9337301587301586

In [12]: # precision 2
         metrics.precision_score(Y_test, Y_pred, average='macro')

Out[12]: 0.9265873015873015

In [13]: # precision 3
         metrics.precision_score(Y_test, Y_pred, average='micro')

Out[13]: 0.9333333333333333

In [17]: # recall 1
         metrics.recall_score(Y_test, Y_pred, average='weighted')

Out[17]: 0.9333333333333333

In [18]: # recall 2
         metrics.recall_score(Y_test, Y_pred, average='macro')

Out[18]: 0.9255599472990778
```

```
In [19]: # recall 3
         metrics.recall_score(Y_test, Y_pred, average='micro')

Out[19]: 0.9333333333333333

In [22]: # f1 1
         metrics.f1_score(Y_test, Y_pred, average='weighted')

Out[22]: 0.9332673593930398

In [23]: # f1 2
         metrics.f1_score(Y_test, Y_pred, average='macro')

Out[23]: 0.925779317169718

In [24]: # f1 3
         metrics.f1_score(Y_test, Y_pred, average='micro')

Out[24]: 0.9333333333333333

In [26]: from sklearn.metrics import classification_report
         print(classification_report(Y_test, Y_pred, target_names=iris.target_names))

              precision    recall  f1-score   support

      setosa       1.00      1.00      1.00        30
  versicolor       0.90      0.86      0.88        22
   virginica       0.88      0.91      0.89        23

   micro avg       0.93      0.93      0.93        75
   macro avg       0.93      0.93      0.93        75
weighted avg       0.93      0.93      0.93        75


In [ ]:
```