機器學習 HW2

2023/04/13

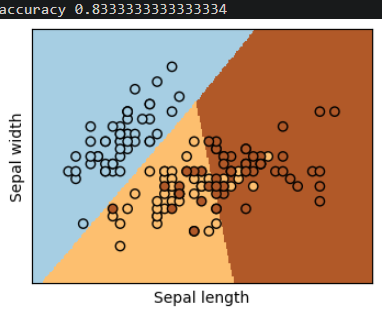
611121213 莊雅卉

1. LogisticRegression
2. LogisticRegression (C=1e5)

LogisticRegression (C=1e5,penalty='l2')

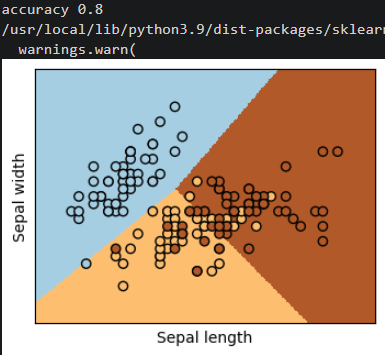
LogisticRegression (C=1e5,penalty='l2',multi\_class='multinomial')

**Accuracy : 83.3%**



1. LogisticRegression (C=1e5,penalty='l1',solver='liblinear')

**Accuracy : 80.0%**

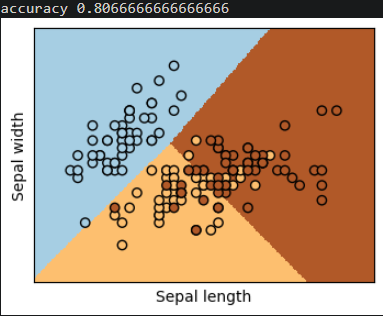


1. LogisticRegression (C=1e5,penalty='l2',solver='liblinear')

LogisticRegression (C=1e5,penalty='l2',solver='saga')

LogisticRegression (C=1e5,penalty='l1',solver='saga')

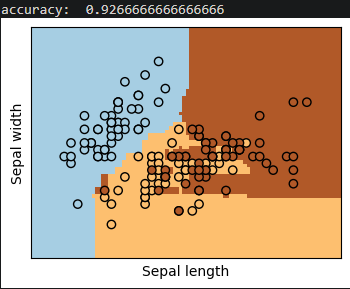
**Accuracy : 80.7%**



1. RandomForestClassifier
2. RandomForestClassifier(n\_estimators=100)

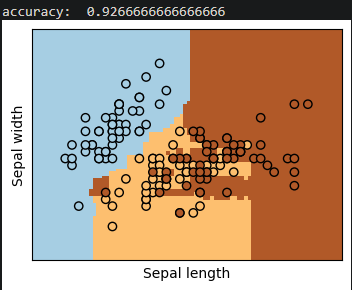
RandomForestClassifier (n\_estimators=100,criterion='gini')

**Accuracy : 92.7%**



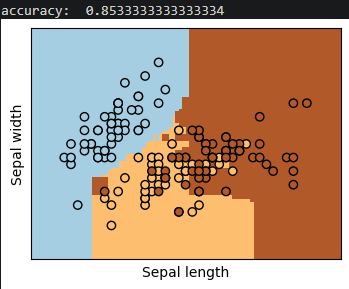
1. RandomForestClassifier (n\_estimators=100,criterion='entropy')

**Accuracy : 92.7%**



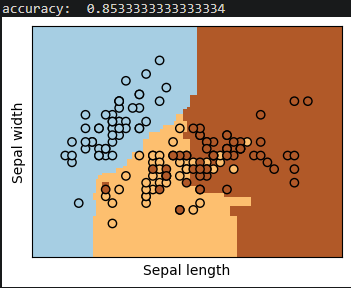
1. RandomForestClassifier(n\_estimators=100,criterion='entropy',max\_depth=5)

**Accuracy : 85.3%**



1. RandomForestClassifier(n\_estimators=100,criterion='gini',max\_depth=5 )

**Accuracy : 85.3%**

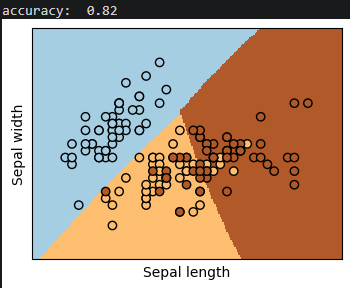


1. SVC
2. SVC()

SVC(C=1,kernel='rbf')

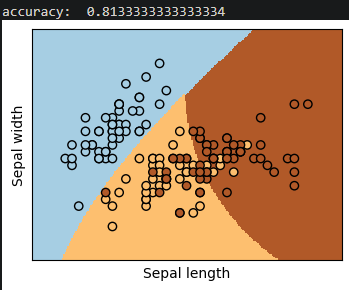
SVC(C=1,kernel='poly',degree=2)

**Accuracy : 82.0%**



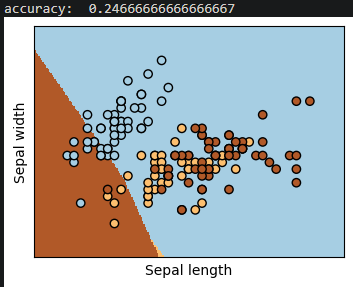
1. SVC(C=1,kernel='poly')

**Accuracy : 81.3%**



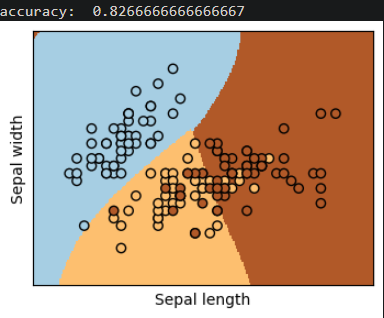
1. SVC(C=1,kernel='sigmoid')

**Accuracy : 24.7%**



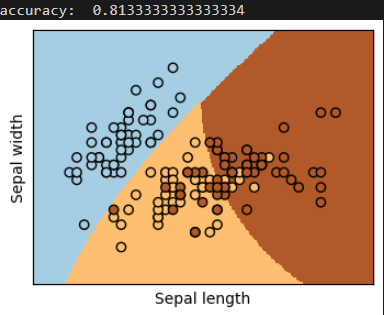
1. SVC(C=1,kernel='rbf',gamma=0.7)

**Accuracy : 82.7%**



1. SVC(C=1,kernel='poly',degree=3)

**Accuracy : 81.3%**

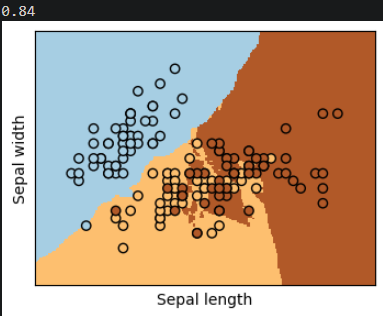


1. KNeighborsClassifier
2. KNeighborsClassifier(n\_neighbors=5)

KNeighborsClassifier(n\_neighbors=5,weights ='uniform',algorithm ='ball\_tree')

KNeighborsClassifier(n\_neighbors=5,weights ='uniform',algorithm ='kd\_tree')

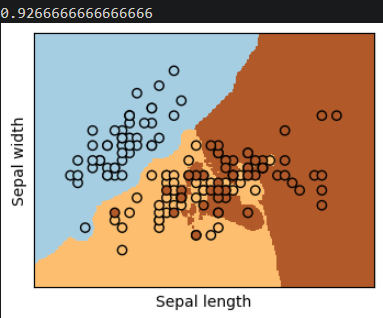
**Accuracy : 84.0%**



1. KNeighborsClassifier (n\_neighbors=5,weights ='distance')

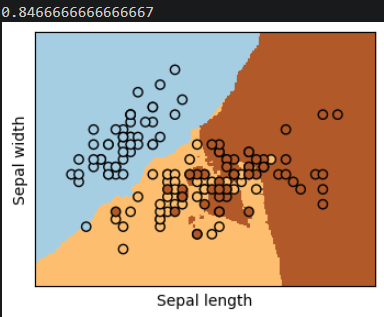
KNeighborsClassifier (n\_neighbors=5,weights ='distance',algorithm ='brute')

**Accuracy : 92.7%**



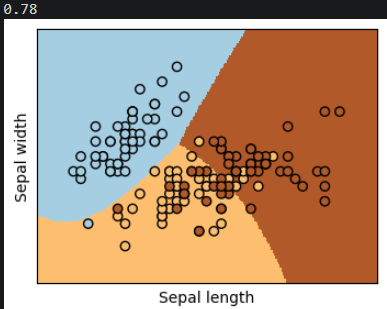
1. KNeighborsClassifier (n\_neighbors=5,weights ='uniform',algorithm ='brute')

**Accuracy : 84.7%**



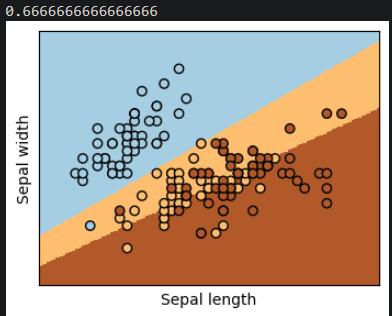
1. GaussianNB MultinomialNB
2. GaussianNB(priors=None)

**Accuracy : 78.0%**



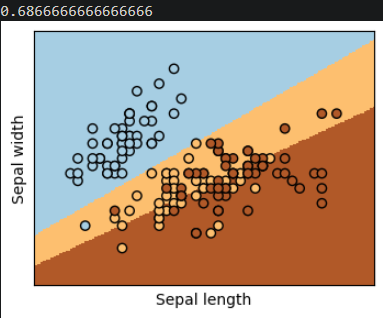
1. MultinomialNB(alpha=1.0, fit\_prior=True/False, class\_prior=None)

**Accuracy : 66.7%**



1. MultinomialNB(alpha=2.0, fit\_prior=True/False, class\_prior=None)

**Accuracy : 68.7%**



1. 統整結果
2. LogisticRegression = 83.3%
3. RandomForestClassifier = 92.7%
4. SVC = 82.7%
5. KNeighborsClassifier = 92.7
6. GaussianNB MultinomialNB = 78.0%
7. 結論

每種方法透過調整參數後皆可以提升準確度，在LogisticRegression情境中，參數penalty='l2',multi\_class='multinomial'為本實驗中準確率最好的一個配置；而在RandomForestClassifier情境中無論是使用gini還是entropy皆能達到很好的分類，而max\_depth設定的高反而會使準確度下降；在SVC情境中，參數設定為C=1,kernel='rbf'已經能夠達到82%的準確率，但再新增設置gamma=0.7時，能夠再將準確率提升一些些，而在kernel='sigmoid'時，準確度明顯地低於rbf、poly；在KNeighbors-Classifier 情境中，最主要的參數為二，一個是weights 在設定為uniform時的表現不如distance，而n\_neighbors在5時的表現度優於3；最後是GaussianNB、MultinomialNB，在本次實驗中GaussianNB的訓練結果優於MultinomialNB，調整了其中的參數也不見好轉。

就這五種分類法而言，RandomForestClassifier、KNeighborsClassifier表現為最佳，GaussianNB MultinomialNB為最差。準確度會依照資料集、參數、方法等因素會有不同的表現，無法斬釘截鐵地說哪個方法是最佳的，而是要依照資料集的屬性去做適合他的分類方式。

透過本次作業能夠清楚的了解五種分類法的參數設定，以及圖片的樣貌，來對機器學習有更深一層的認識。