機器學習 HW3

2023/05/11

611121213 莊雅卉

1. LogisticRegression
2. LogisticRegression (C=1e5,class\_weight='balanced')

**Accuracy : 97.0%**

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

Accuracy : 94.2%

1. LogisticRegression (C=1e5)

Accuracy : 96.4%

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

Accuracy : 94.1%

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

Accuracy : 92.3%

1. LogisticRegression (C=1e5,class\_weight='balanced',solver='liblinear')

Accuracy : 95.3%

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

Accuracy : 95.9%

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

Accuracy : 90.6%

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

Accuracy : 90.0%

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

Train\_Accuracy:100%,Test\_Accuracy : 95.3%

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

Train\_Accuracy:100.0%,Test\_Accuracy : 96.4%

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

Train\_Accuracy:100.0%,Test\_Accuracy : 98.2%

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

Train\_Accuracy:98.9%,Test\_Accuracy : 95.9%

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

Train\_Accuracy:99.7%,Test\_Accuracy : 92.9%

1. RandomForestClassifier(n\_estimators=100,class\_weight='balanced' )

Train\_Accuracy:100.0%,Test\_Accuracy : 96.4%

1. RandomForestClassifier(n\_estimators=100, class\_weight='balanced\_subsample')

Train\_Accuracy:100.0%,**Test\_Accuracy : 98.2%**

1. SVC
2. SVC(C=1e5,kernel="linear",gamma="scale",class\_weight="balanced")

Train\_Accuracy:100.0%,**Test\_Accuracy : 98.2%**

1. SVC(C=1e5,kernel="linear",gamma="scale")

Train\_Accuracy:97.2%,Test\_Accuracy : 94.7%

1. SVC(C=1e5,kernel="linear",gamma="auto",class\_weight="balanced")

Train\_Accuracy:98.2%,Test\_Accuracy : 92.3%

1. SVC(C=1e5,kernel="linear",gamma="auto")

Train\_Accuracy:98.2%,Test\_Accuracy : 94.1%

1. SVC(C=1e5,kernel="poly",gamma="auto",class\_weight="balanced")

Train\_Accuracy:98.2%,Test\_Accuracy : 94.1%

1. BaggingClassifier
2. BaggingClassifier(base\_estimator=SVC(), n\_estimators=10)

Accuracy : 91.8%

1. BaggingClassifier(base\_estimator=SVC(C=1e5,kernel="linear",gamma="scale",class\_weight="balanced"), n\_estimators=10)

**Accuracy : 96.4%**

1. VotingClassifier
2. VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3)], voting='hard')

Accuracy : 94.1%

1. VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3)], voting='soft')

**Accuracy : 97.1%**

1. VotingClassifier(estimators=[('lr', clf1), ('rf', clf2), ('gnb', clf3)], voting='hard', n\_jobs=12)

Accuracy : 96.4%

1. 統整結果
2. LogisticRegression = 97.0%
3. RandomForestClassifier = 98.2%
4. SVC = 98.2%
5. BaggingClassifier= 96.4%
6. VotingClassifier = 97.1%
7. 結論

本次實驗針對breast\_cancer這個資料集去進行分類預測，透過改變參數來去對分類達到一個良好的預測，在LogisticRegression情境中，參數class\_weight='balanced' 為本實驗中準確率最好的一個配置；而在RandomForestClassifier情境中，參數n\_estimators=100, class\_weight='balanced\_subsample'所訓練出來的模組準確率有高達98%，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=1e5,kernel="linear", gamma = "scale", class\_weight="balanced"也有高達98%的準確率，改變gamma值也會影響準確率。

而在Bagging中，若將基本模型建立的越好，準確率也就會跟著提升，透過調整模組來達到更好的分類預測，那透過隨機取樣的方式，可以有效地去減少一些noises的選取，來避免overfitting；Voting透過hard、soft模式來變換他的計算權種方式，由實驗可以看出利用soft voting 的準確率是高於hard voting。

以上三種傳統訓練方法都是透過增加class\_weight來提高模組的準確度，在計算權重時，權重越大的loss就會越大，來去避免overfitting。配合bagging、voting分類來去求得一個較佳的分類模組。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Bagging與voting在機器學習中所扮演的角色，透過參數的設定來去改變他們計算權重的方式，以及運作的過程，來對機器學習有更深一層的認識。