Machine Learning Lab 3

July 22, 2021

1 Practical 3: Logistic,SVM,Decision Tree,Random Forest with sklearn and Kflod with sklearn

```
[2]: import pandas as pd
     from numpy import mean,std
     import sklearn
     from sklearn.datasets import load_digits
     import matplotlib.pyplot as plt
```

```
[3]: digits = load_digits()
     X = digits.data
     y = digits.target
```

2 KFold Cross-validation

That k-fold cross validation is a procedure used to estimate the skill of the model on new data. There are common tactics that you can use to select the value of k for your dataset. There are commonly used variations on cross-validation such as stratified and repeated that are available in scikit-learn.

```
[4]: from  sklearn.model_selection import KFold
     from sklearn.model_selection import cross_val_score

     from sklearn.linear_model import LogisticRegression

     from sklearn.svm import SVC
     from sklearn.pipeline import make_pipeline
     from sklearn.preprocessing import StandardScaler

     from sklearn.tree import DecisionTreeClassifier

     from sklearn.ensemble import RandomForestClassifier
```

Logistic Regression

```
[5]: cv = KFold(n_splits=10, random_state=1, shuffle=True)
     logistic = LogisticRegression()
```

```
scoresL = cross_val_score(logistic, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
print('Logistic Accuracy: %.3f (%.3f)' % (mean(scoresL), std(scoresL)))
```

Logistic Accuracy: 0.968 (0.010)

Support Vector Machine

[6]:
```
cv = KFold(n_splits=10, random_state=1, shuffle=True)
svc = make_pipeline(StandardScaler(), SVC(gamma='auto'))
scoresS = cross_val_score(svc, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
print('SVM Accuracy: %.3f (%.3f)' % (mean(scoresS), std(scoresS)))
```

SVM Accuracy: 0.982 (0.008)

Decision Tree

[7]:
```
cv = KFold(n_splits=10, random_state=1, shuffle=True)
dt = DecisionTreeClassifier(random_state=0)
scoresdt = cross_val_score(dt, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
print('Decision Tree Accuracy: %.3f (%.3f)' % (mean(scoresdt), std(scoresdt)))
```

Decision Tree Accuracy: 0.856 (0.023)

Random Forest

[8]:
```
cv = KFold(n_splits=10, random_state=1, shuffle=True)
rf = RandomForestClassifier(random_state=0)
scoresrf = cross_val_score(rf, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
print('Random Forest Accuracy: %.3f (%.3f)' % (mean(scoresrf), std(scoresrf)))
```

Random Forest Accuracy: 0.974 (0.010)

3   StratifiedKFold

[9]:
```
from  sklearn.model_selection import StratifiedKFold

from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression

from sklearn.svm import SVC
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier
```

Logistic Regression with StratifiedKFold

```
[10]: cv = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
      logistic = LogisticRegression()
      scoresL = cross_val_score(logistic, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
      print('Logistic Accuracy: %.3f (%.3f)' % (mean(scoresL), std(scoresL)))
```

```
Logistic Accuracy: 0.965 (0.011)
```

Support Vector Machine with StratifiedKFold

```
[11]: cv = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
      svc = make_pipeline(StandardScaler(), SVC(gamma='auto'))
      scoresS = cross_val_score(svc, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
      print('SVM Accuracy: %.3f (%.3f)' % (mean(scoresS), std(scoresS)))
```

```
SVM Accuracy: 0.982 (0.012)
```

Decision Treen with StratifiedKFold

```
[12]: cv = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
      dt = DecisionTreeClassifier(random_state=0)
      scoresdt = cross_val_score(dt, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
      print('Decision Tree Accuracy: %.3f (%.3f)' % (mean(scoresdt), std(scoresdt)))
```

```
Decision Tree Accuracy: 0.853 (0.024)
```

Random Forest with StratifiedKFold

```
[13]: cv = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
      rf = RandomForestClassifier(random_state=0)
      scoresrf = cross_val_score(rf, X, y, scoring='accuracy', cv=cv, n_jobs=-1)
      print('Random Forest Accuracy: %.3f (%.3f)' % (mean(scoresrf), std(scoresrf)))
```

```
Random Forest Accuracy: 0.980 (0.007)
```