Machine Learning Lab Homework 3

May 10, 2021

∗ Please note that all homework should be your own work. You should also not copy answers from other person's, books or internet resources.
∗ I didn't proofread the questions. If you find any typos/errors, let me know.

∗ Use the same dataFind **ONE** datasets, and drop a line at e-class Q & A saying "OOO uses OOO"

0. Read File

read your csv file into 2-dimenson list (**a_list**).

1. Preprocessing

If your data needs preprocssing (Label Encoding & Normalization), please do that.

2. Divide into train & test

From a_list, construct X_train, X_test, Y_train, Y_test

3. Running AdaBoost

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
```

prepare your X_train, X_test, Y_train, Y_test files

```
# build ensemble model
clf = AdaBoostClassifier()
clf.fit(X_train, Y_train)

# Predict the class labels for the provided data
predictions = clf.predict(X_test)

# probability estimates for the test data X.
pred_prob = clf.predict_proba(X_test)

# show the mean accuracy
score OR accuracy_score
```

O Refer to the following page
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html

1) run by changing n_estimators = 5, 7, 10, 30, 100, and show the accuracies of each run.

2) Plot your results and explain the effect of the n_estimators.

3) compare the best performance of AdaBoostClassifier with that of IBL. Which is better ? Explain the results.

4. Running Random Forest

```
from sklearn import RandomForestClassifier
```

prepare your X_train, X_test, Y_train, Y_test files

```
# build random forest model
clf = RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
clf.fit(X, y)

# Predict the class labels for the provided data
predictions = clf.predict(X_test)

# show the mean accuracy
score OR accuracy_score
```

O refer to the following page
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

1) Run by changing n_estimators = 2, 5, 30, 50, 100, respectively, and show the accuracies of each run.

2) Plot your results and explain the effect of the n_estimators.

3) Choose the optimal n_estimators from q. 1), and run the model by changing oob_score = True/False. respectively. Show the accuracies of each run, and explain the effect of the oob_score.

4) Choose the optimal n_estimators from q. 1), and run the model by changing max_features = "auto", "sqrt", "log2", respectively. Show the accuracies of each run, and explain the effect of the max_features.

5. Running SVM

```
from sklearn import svm

clf = svm.SVC()
clf.fit(X_train, Y_train)
clf.predict(X_test)
```

O refer to the following page
https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

1) calculate the accuracy of SVC.

2) run SVC by changing kernel to 'linear', 'poly', 'rbf', and 'sigmoid', and show the accuracies of each. Which kernel function shows the best accuracy ? and explain why ?

3) compare the accuracy of SVC with that of IBL, RandomForest, and AdaBoost, respectively. Explain the results.

6. Clustering (K Means)

```
import matplotlib.pyplot as pyplot

from sklearn.cluster import KMeans
import numpy as np

kmeans = KMeans(n_clusters=4, max_iter=600, algorithm = 'auto', random_state=0)
kmeans.fit(X_train)
labels = kmeans.predict(X_train)

# cluster labels for each data
labels = kmeans.labels_

# center of each clusters
centroids = kmeans.cluster_centers_

# distance within cluster
print("Inertia for KMeans with 4 clusters = %lf " %(kmeans.inertia_))
```

O refer to the following page
http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

1) run KMeans 3 times by changing n_clusters = 2, 5, 7 respectively and show the mean of each cluster.

2) For the clustering of k=2, pick one cluster. Calculate the average value of each attribute of the data in that cluster.

3) For each cluster, calculate majority (the most frequent) value of class/target value. (let's call this 'cluster label')

4) Suppose each of X_test is classified based on 'cluster labels', calculate the accuracy.

5) run KMeans 3 times by changing n_init values (your own choice of n_init). Compare the performance of each.

7. Clustering (EM)

```
import matplotlib.pyplot as plt

import numpy as np
from sklearn.mixture import GaussianMixture

gmm = GaussianMixture(n_components=4).fit(X_train)

# predict the labels for data points
labels = gmm.predict(X_train)

# probabilistic cluster assignments
probs = gmm.predict_proba(X_train)
print(probs[:5].round(3))
```

O refer to the following page
https://scikit-learn.org/stable/modules/generated/sklearn.mixture.GaussianMixture.html

1) run GaussianMixture 3 times by changing n_clusters = 2, 3, 7 respectively.

2) For the clustering of k=3, show the predicted labels for the input data.

3) show the probabilistic cluster assignments. This returns a matrix of size [n_samples, n_clusters].

Hand In

upload the following files at e-class.
- **report file, ii) program source file, iii) data file**

Due: 5/21(Fri) 23:59 PM