breast_cancer

February 27, 2023

1 Tugas Kecil 1 Machine Learning

1.1 Eksplorasi library Algoritme Pembelajaran pada Jupyter Notebook

13520092 - Vieri Mansyl
13520099 - Vincent Prasetiya Atmadja

[ ]:

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import pickle

# Load the breast cancer dataset
cancer = load_breast_cancer()

# Allocate training data 80% and test data 20%
X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target,␣
  ↪test_size=0.2, random_state=0)

print(len(cancer.feature_names))
```

```
30
```

1.2 Decision Tree Classifier

[ ]:

```
from sklearn.tree import DecisionTreeClassifier, export_text
from sklearn.model_selection import cross_validate
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
  ↪f1_score, confusion_matrix

# initialize DT classifier
clf = DecisionTreeClassifier()

# train the model
clf.fit(X_train, y_train)

# export the decision tree in text format
r = export_text(clf, feature_names=cancer.feature_names.tolist())
```

```

# show decision tree
print("decision tree:\n", r)
```

[ ]:

```
# save model using pickle
with open('models/decisionTreeClassifier.pkl', 'wb') as file:
    pickle.dump(clf, file)

#  load model using pickle
with open('models/decisionTreeClassifier.pkl', 'rb') as file:
    clf = pickle.load(file)

# predict datasets with model
predictions = clf.predict(X_test)

# evaluate metrics
dtAccuracy = accuracy_score(y_test, predictions)
dtPrecision = precision_score(y_test, predictions)
dtRecall = recall_score(y_test, predictions)
dtF1 = f1_score(y_test, predictions)
dtCm = confusion_matrix(y_test, predictions)

# measure model's performance using 10-fold cross validation
dtCv = cross_validate(clf, cancer.data, cancer.target, cv=10,␣
 ↪scoring=['accuracy', 'f1'])
```

1.3 ID3 Estimator

[ ]:

```
from id3 import Id3Estimator

estimator = Id3Estimator()
estimator = estimator.fit(X_train, y_train)

with open('models/id3Estimator.pkl', 'wb') as file:
    pickle.dump(estimator, file)

#  load model using pickle
with open('models/id3Estimator.pkl', 'rb') as file:
    clf = pickle.load(file)

estimatorPredictions = estimator.predict(X_test)

# evaluate metrics
id3Accuracy = accuracy_score(y_test, estimatorPredictions)
id3Precision = precision_score(y_test, estimatorPredictions)
id3Recall = recall_score(y_test, estimatorPredictions)
id3F1 = f1_score(y_test, estimatorPredictions)
```

```
id3Cm = confusion_matrix(y_test, estimatorPredictions)
```

1.4 K Means

[ ]:

```
from sklearn.cluster import KMeans

# initialize KMeans with 5 clusters with n_init=10
kmeans = KMeans(n_clusters=2, n_init=10)

# train the model
kmeans.fit(X_train)

# save model using pickle
with open('models/kMeans.pkl', 'wb') as file:
    pickle.dump(kmeans, file)

#  load model using pickle
with open('models/kMeans.pkl', 'rb') as file:
    kmeans = pickle.load(file)

# predict datasets with model
predictions = kmeans.predict(X_test)


print("prediction:\n" , predictions)
```

1.5 Logistic Regression

[ ]:

```
from sklearn.linear_model import LogisticRegression

# Initialize Logistic Regression

lRegression = LogisticRegression(random_state=0, max_iter=3850)

# Train Model
lRegression.fit(X_train, y_train)

# save model using pickle
with open('models/logisticRegression.pkl', 'wb') as file:
    pickle.dump(kmeans, file)

#  load model using pickle
with open('models/logisticRegression.pkl', 'rb') as file:
    lRegression = pickle.load(file)

estimatorPredictions = lRegression.predict(X_test)
```

```
# evaluate metrics
lrAccuracy = accuracy_score(y_test, estimatorPredictions)
lrPrecision = precision_score(y_test, estimatorPredictions)
lrRecall = recall_score(y_test, estimatorPredictions)
lrF1 = f1_score(y_test, estimatorPredictions)
lrCm = confusion_matrix(y_test, estimatorPredictions)
```

1.6 Neural Network - Multi-layer Perceptron (MLP) Classifier

[ ]:

```
from sklearn.neural_network import MLPClassifier
from sklearn.datasets import make_classification
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
 ↪f1_score, confusion_matrix

# initialize MLP classifier
clf = MLPClassifier(random_state=1, max_iter=300)

# train the model
clf.fit(X_train, y_train)

# save model using pickle
with open('models/mlpClassifier.pkl', 'wb') as file:
    pickle.dump(clf, file)

#  load model using pickle
with open('models/mlpClassifier.pkl', 'rb') as file:
    clf = pickle.load(file)

# predict datasets with model
predictions = clf.predict(X_test)

# evaluate metrics
mlpAccuracy = accuracy_score(y_test, predictions)
mlpPrecision = precision_score(y_test, predictions)
mlpRecall = recall_score(y_test, predictions)
mlpF1 = f1_score(y_test, predictions)
mlpCm = confusion_matrix(y_test, predictions)
```

1.7 Support Vector Machine

[ ]:

```
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
import numpy as np
```

```
# initialize SVM classifier
svm = make_pipeline(StandardScaler(), SVC(gamma='auto'))

# Train data
svm.fit(X_train, y_train)

# save model using pickle
with open('models/svmClassifier.pkl', 'wb') as file:
    pickle.dump(svm, file)

#  load model using pickle
with open('models/svmClassifier.pkl', 'rb') as file:
    svm = pickle.load(file)

# predict datasets with model
predictions = svm.predict(X_test)

# evaluate metrics
svmAccuracy = accuracy_score(y_test, predictions)
svmPrecision = precision_score(y_test, predictions)
svmRecall = recall_score(y_test, predictions)
svmF1 = f1_score(y_test, predictions)
svmCm = confusion_matrix(y_test, predictions)
```

1.8 Evaluation Matrix

1.8.1 Decision Tree

[ ]:

```
# show metrics
print('Accuracy: ', dtAccuracy)
print('Precision: ', dtPrecision)
print('Recall: ', dtRecall)
print('F1: ', dtF1)
print('Confusion Matrix:\n', dtCm)

# show cross validation metrics
print('Cross Validation Accuracy: ', dtCv['test_accuracy'].mean())
print('Cross Validation F1', dtCv['test_f1'].mean())
```

```
Accuracy:  0.9122807017543859
Precision:  0.8701298701298701
Recall:  1.0
F1:  0.9305555555555556
Confusion Matrix:
 [[37 10]
 [ 0 67]]
Cross Validation Accuracy:  0.9280075187969924
Cross Validation F1 0.9426659098997107
```

1.8.2 ID3 Estimator

[ ]:

```
# show metrics
print('Accuracy: ', id3Accuracy)
print('Precision: ', id3Precision)
print('Recall: ', id3Recall)
print('F1: ', id3F1)
print('Confusion Matrix:\n', id3Cm)
```

```
Accuracy:  0.9122807017543859
Precision:  0.9384615384615385
Recall:  0.9104477611940298
F1:  0.9242424242424243
Confusion Matrix:
 [[43  4]
 [ 6 61]]
```

1.8.3 Logistic Regression

[ ]:

```
# show metrics
print('Accuracy: ', lrAccuracy)
print('Precision: ', lrPrecision)
print('Recall: ', lrRecall)
print('F1: ', lrF1)
print('Confusion Matrix:\n', lrCm)
```

```
Accuracy:  0.8157894736842105
Precision:  0.7613636363636364
Recall:  1.0
F1:  0.8645161290322582
Confusion Matrix:
 [[26 21]
 [ 0 67]]
```

1.8.4 Neural Network

[ ]:

```
# show metrics
print('Accuracy: ', mlpAccuracy)
print('Precision: ', mlpPrecision)
print('Recall: ', mlpRecall)
print('F1: ', mlpF1)
print('Confusion Matrix:\n', mlpCm)
```

```
Accuracy:  0.9122807017543859
Precision:  0.8701298701298701
Recall:  1.0
F1:  0.9305555555555556
Confusion Matrix:
 [[37 10]
 [ 0 67]]
```

1.8.5 SVM

[ ]:

```
# show metrics
print('Accuracy: ', svmAccuracy)
print('Precision: ', svmPrecision)
print('Recall: ', svmRecall)
print('F1: ', svmF1)
print('Confusion Matrix:\n', svmCm)
```

```
Accuracy:  0.9824561403508771
Precision:  0.9710144927536232
Recall:  1.0
F1:  0.9852941176470589
Confusion Matrix:
 [[45  2]
 [ 0 67]]
```

1.9 Analisis Evalution Matrix pada seluruh model pembelajaran

Untuk K-Means : Karena K-Means model merupakan model pembelajaran Unsupervised, maka hasil prediksi dari K-Means yang merupakan prediksi letak data terhadap cluster-cluster yang terbentuk tidak dapat diukur dengan metrik seperti accuracy, precision, recall, F1, dan confusion matrix yang digunakan hanya untuk mengukur model pembelajaran Supervised

Model pembelajaran lainnya: - Accuracy Model pembelajaran yang memiliki akurasi tertinggi ialah model SVM dengan akurasi 98.246%.

- Precision Model pembelajaran yang memiliki presisi tertinggi ialah model SVM dengan presisi sebesar 97.10%.
- Recall Nilai Recall tertinggi yang didapatkan adalah 1.0, yang mana didapatkan oleh model pembelajaran Decision Tree , Logistic Regression , dan Neural Network
- F1 Model pembelajaran yang memiliki F1 tertinggi ialah model SVM nilai F1 sebesar 98.52%

1.9.1 Kesimpulan

Dari analisis terhadap perfoma matriks yang ada, SVM merupakan model pembelajaran yang terbaik. SVM cocok digunakan untuk data dengan atribut yang banyak. Dalam kasus ini, Breast Cancer memilki 30 atribut yang mana cocok dengan use-case dari SVM.
Selain itu, Breast Cancer memilki 2 label saja, yang mana cocok dengan algoritma dari SVM

1.10 Perbandingan 10-Fold Cross Validation pada Decision Tree Classifier dengan hasil analisis

Hasil cross validation pada DTF untuk metrik accuracy dan F1 berturut-turut adalah 0.9280075187969924 dan 0.9426659098997107.

Walaupun begitu, nilai cross validation dari DTF masih lebih kecil dibanding dengan hasil metrik dari SVM. Ini menunjukkan bahwa pada dataset breast cancer, model SVM masih lebih baik dibanding model DTF.