Tugas Kecil 1

Eksplorasi library Algoritme Pembelajaran pada Jupyter Notebook

Disusun oleh:

13520133 - Jevant Jedidia

13520160 - Willy Wilsen

In [ ]:
```
# Import Dataset

from sklearn import datasets
from sklearn.model_selection import train_test_split
import pickle

breast_cancer = datasets.load_breast_cancer()
x_train, x_test, y_train, y_test = train_test_split(breast_cancer.data, breast_cancer.
```

In [ ]:
```
# Decission Tree Classifier

from sklearn import tree, metrics
from sklearn.model_selection import cross_validate
from sklearn.model_selection import cross_val_score

breast = []
clf = tree.DecisionTreeClassifier(random_state=0)
clf = clf.fit(x_train, y_train)
for item in breast_cancer['feature_names']:
    breast.append(item)
r = tree.export_text(clf, feature_names=breast)
print(r)

DTC = pickle.dumps(clf)
pred = pickle.loads(DTC).predict(x_test)

print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred)))
print("Recall: " + str(metrics.recall_score(y_test, pred)))
print("F1: " + str(metrics.f1_score(y_test, pred)))
print("Confusion Matrix: ")
print(metrics.confusion_matrix(y_test, pred))

cvAcc = cross_val_score(clf, x_train, y_train, cv=10, scoring='accuracy')
cvF1 = cross_val_score(clf, x_train, y_train, cv=10, scoring='f1')
print("Accuracy with Cross Validate:", cvAcc.mean())
print("F1 Score with Cross Validate:", cvF1.mean())
```

```
|--- worst perimeter <= 106.05
|   |--- worst smoothness <= 0.18
|   |   |--- worst concave points <= 0.16
|   |   |   |--- worst fractal dimension <= 0.06
|   |   |   |   |--- class: 0
|   |   |   |--- worst fractal dimension >  0.06
|   |   |   |   |--- worst texture <= 30.15
|   |   |   |   |   |--- area error <= 48.98
|   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |--- area error >  48.98
|   |   |   |   |   |   |--- mean compactness <= 0.06
|   |   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |   |--- mean compactness >  0.06
|   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |--- worst texture >  30.15
|   |   |   |   |   |--- mean fractal dimension <= 0.06
|   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |--- mean fractal dimension >  0.06
|   |   |   |   |   |   |--- smoothness error <= 0.01
|   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |--- smoothness error >  0.01
|   |   |   |   |   |   |   |--- mean radius <= 12.38
|   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |   |--- mean radius >  12.38
|   |   |   |   |   |   |   |   |--- class: 0
|   |   |--- worst concave points >  0.16
|   |   |   |--- smoothness error <= 0.01
|   |   |   |   |--- class: 0
|   |   |   |--- smoothness error >  0.01
|   |   |   |   |--- class: 1
|   |--- worst smoothness >  0.18
|   |   |--- class: 0
|--- worst perimeter >  106.05
|   |--- worst perimeter <= 117.45
|   |   |--- mean texture <= 19.36
|   |   |   |--- mean perimeter <= 91.97
|   |   |   |   |--- class: 0
|   |   |   |--- mean perimeter >  91.97
|   |   |   |   |--- class: 1
|   |   |--- mean texture >  19.36
|   |   |   |--- worst smoothness <= 0.11
|   |   |   |   |--- class: 1
|   |   |   |--- worst smoothness >  0.11
|   |   |   |   |--- class: 0
|   |--- worst perimeter >  117.45
|   |   |--- mean concave points <= 0.03
|   |   |   |--- class: 1
|   |   |--- mean concave points >  0.03
|   |   |   |--- class: 0

Accuracy: 0.8421052631578947
Precision: 0.9605263157894737
Recall: 0.8295454545454546
F1: 0.8902439024390244
Confusion Matrix:
[[23  3]
 [15 73]]
Accuracy with Cross Validate: 0.9232367149758455
F1 Score with Cross Validate: 0.9351074327489423
```

In [ ]:
```
# Id3Estimator

import six
import sys
sys.modules['sklearn.externals.six'] = six
import mlrose
from id3 import Id3Estimator
from id3 import export_graphviz

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

ID = pickle.dumps(estimator)
pred = pickle.loads(ID).predict(x_test)

print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred)))
print("Recall: " + str(metrics.recall_score(y_test, pred)))
print("F1: " + str(metrics.f1_score(y_test, pred)))
print("Confusion Matrix: ")
metrics.confusion_matrix(y_test, pred)
```

Out[ ]:
```
Accuracy: 0.9210526315789473
Precision: 0.9876543209876543
Recall: 0.9090909090909091
F1: 0.9467455621301774
Confusion Matrix:
array([[25,  1],
       [ 8, 80]])
```

In [ ]:
```
# K Means

from sklearn.cluster import KMeans
import numpy as np

X = np.array(x_train)
kmeans = KMeans(n_clusters=2, random_state=0).fit(X)

KM = pickle.dumps(kmeans)
pred = pickle.loads(KM).predict(x_test)

print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred, average='weighted')))
print("Recall: " + str(metrics.recall_score(y_test, pred, average='weighted')))
print("F1: " + str(metrics.f1_score(y_test, pred, average='weighted')))
print("Confusion Matrix: ")
metrics.confusion_matrix(y_test, pred)
```

Out[ ]:
```
Accuracy: 0.9122807017543859
Precision: 0.9149610136452242
Recall: 0.9122807017543859
F1: 0.9062131613619028
Confusion Matrix:
array([[17,  9],
       [ 1, 87]])
```

In [ ]:
```
# Logistic Regression

from sklearn.linear_model import LogisticRegression
```

```
clf = LogisticRegression(random_state=0).fit(x_train, y_train)

LR = pickle.dumps(clf)
pred = pickle.loads(LR).predict(x_test)

print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred)))
print("Recall: " + str(metrics.recall_score(y_test, pred)))
print("F1: " + str(metrics.f1_score(y_test, pred)))
print("Confusion Matrix: ")
metrics.confusion_matrix(y_test, pred)
```

```
Accuracy: 0.9298245614035088
Precision: 0.9878048780487805
Recall: 0.9204545454545454
F1: 0.9529411764705882
Confusion Matrix:
/shared-libs/python3.9/py/lib/python3.9/site-packages/sklearn/linear_model/_logistic.
py:444: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[ ]:
```
array([[25,  1],
       [ 7, 81]])
```

In [ ]:
```
# Neural_network

from sklearn.neural_network import MLPClassifier
from sklearn.datasets import make_classification

clf = MLPClassifier(random_state=1, max_iter=50).fit(x_train, y_train)

MLP = pickle.dumps(kmeans)
pred = pickle.loads(MLP).predict(x_test)


print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred, average='weighted')))
print("Recall: " + str(metrics.recall_score(y_test, pred, average='weighted')))
print("F1: " + str(metrics.f1_score(y_test, pred, average='weighted')))
print("Confusion Matrix: ")
metrics.confusion_matrix(y_test, pred)
```

```
Accuracy: 0.9122807017543859
Precision: 0.9149610136452242
Recall: 0.9122807017543859
F1: 0.9062131613619028
Confusion Matrix:
/shared-libs/python3.9/py/lib/python3.9/site-packages/sklearn/neural_network/_multila
yer_perceptron.py:702: ConvergenceWarning: Stochastic Optimizer: Maximum iterations
(50) reached and the optimization hasn't converged yet.
  warnings.warn(
```

Out[ ]:
```
array([[17,  9],
       [ 1, 87]])
```

In [ ]:
```
# SVM

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

clf = make_pipeline(StandardScaler(), SVC(gamma='auto')).fit(x_train, y_train)

SVM = pickle.dumps(clf)
pred = pickle.loads(SVM).predict(x_test)

print("Accuracy: " + str(metrics.accuracy_score(y_test, pred)))
print("Precision: " + str(metrics.precision_score(y_test, pred)))
print("Recall: " + str(metrics.recall_score(y_test, pred)))
print("F1: " + str(metrics.f1_score(y_test, pred)))
print("Confusion Matrix: ")
metrics.confusion_matrix(y_test, pred)
```

Out[ ]:
```
Accuracy: 0.9736842105263158
Precision: 1.0
Recall: 0.9659090909090909
F1: 0.9826589595375723
Confusion Matrix:
array([[26,  0],
       [ 3, 85]])
```

Analisis:

Pada Decission Tree Classifier, didapatkan 23 data berlabel True Positive, 3 data berlabel False Negative, 15 data berlabel False Positive, dan 73 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 84.21%, 96.05%, 82.95%, dan 89%. Lalu, dapat dilihat nilai Accuracy dan F1 untuk 10-fold cross validation adalah 92.32% dan 93.51%. Apabila dibandingkan dengan nilai yang didapat dari Decission Tree Classifier, kita lihat perbedaan nilai yang cukup signifikan pada Accuracy dan F1. Dengan adanya hal tersebut berarti bahwa model yang didapat dari Decision Tree Classifier bukanlah model yang terbaik dan dapat dicari lagi model yang lebih baik karena perbedaan nilai tersebut.

Pada Id3Estimator, didapatkan 25 data berlabel True Positive, 1 data berlabel False Negative, 8 data berlabel False Positive, dan 80 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 92.1%, 98.76%, 90.9%, dan 94.67%.

Pada K Means, didapatkan 17 data berlabel True Positive, 9 data berlabel False Negative, 1 data berlabel False Positive, dan 87 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 91.22%, 91.49%, 91.22%, dan 90.62%.

Pada Logistic Regression, didapatkan 25 data berlabel True Positive, 1 data berlabel False Negative, 7 data berlabel False Positive, dan 81 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 92.98%, 98.78%, 92.04%, dan 95.29%.

Pada Neural Network, didapatkan 17 data berlabel True Positive, 9 data berlabel False Negative, 1 data berlabel False Positive, dan 87 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 91.22%, 91.49%, 91.22%, dan 90.62%.

Pada SVM, didapatkan 26 data berlabel True Positive, 0 data berlabel False Negative, 3 data berlabel False Positive, dan 85 data berlabel False Negative dari keseluruhan data. Accuracy, Precision, Recall, dan F1 yang didapatkan berturut-turut bernilai 97.36%, 100%, 96.59%, dan 98.26%.

Dari setiap algoritma pembelajaran, dapat dibandingkan nilai-nilai performanya, yaitu accuracy, precision, recall, dan F1. Secara umum nilai yang didapat cukup baik sehingga model yang didapat dari algoritma masing-masing sudah cukup baik. Algoritma Decision Tree Classifier memiliki nilai accuracy dan F1 yang terkecil dari semua algoritma sehingga dapat dibilang bahwa model yang dihasilkan dari algoritma Decision Tree Classifier merupakan model yang terburuk di antara semua model yang dihasilkan. SVM menjadi algoritma yang terbaik apabila dilihat dari penilaian metrik yang dihasilkan.