MODUL PRAKTIKUM

IS411 – DATA MODELING
PROGRAM SARJANA S1 SISTEM INFORMASI
FAKULTAS TEKNIK DAN INFORMATIKA

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNIK DAN INFORMATIKA
UNIVERSITAS MULTIMEDIA NUSANTARA
Gedung B Lantai 5, Kampus UMN
Jl. Scientia Boulevard, Gading Serpong, Tangerang, Banten-15811 Indonesia
Telp: +62-21.5422.0808 (ext. 1803), email: ict.lab@umn.ac.id, web: umn.ac.id

MODUL 10
CLASSIFICATION

DESKRIPSI TEMA

1. C5.0 Algorithm
2. Importance Metrics of Random Forest and C5.0 Algorithms.

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

Students are able to illustrate data modeling problems using classification techniques (C4)

PENUNJANG PRAKTIKUM

1. Anaconda Navigator
2. Jupyter Notebook

(+ Perlengkapan Apd/Alat Pelindung Diri Yang Harus Digunakan, Jika Ada)

LANGKAH-LANGKAH PRAKTIKUM

Import Library dan Load Dataset

1. Lakukan langkah untuk mengimport library Pandas.
2. Import dataset menggunakan file “diabetes.csv”.

```
dataset = pd.read_csv('diabetes.csv', header=0)
dataset
```

Note: gunakan delimiter sesuai settingan pada komputer Anda (bisa “;” atau “,”).

3. Tampilkan deskripsi dataset.

```
dataset.info()
```

Modeling

4. Menentukan variable X dan Y.

```
#split dataset in features and target variable
feature_cols = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness','Insulin','BMI','DiabetesPedigreeFunction', 'Age']
X = dataset[feature_cols] # Features
y = dataset['Outcome'] # Target variable
```

Kali ini kita menggunakan semua variabel independen sebagai X, dan variabel dependen (target) sebagai y.

5. Membagi data training dan data testing.

```python
# Split dataset into training set and test set
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and 30% test
```

Menggunakan Teknik Pruning

Decision Tree Classifier memiliki beberapa algoritma, diantaranya ID3, C4.5, C5.0 dan CART. Python menggunakan scikit-learn yang menggunakan metode CART. Secara kegunaan CART dan C4.5 sangat mirip tetapi yang membedakan adalah CART juga memiliki fungsi untuk melakukan regresi. Sedangkan C5.0 merupakan pengembangan dari C4.5.

Dalam melakukan Decision Tree, baik C4.5 maupun CART, dapat melakukan ***pruning***. Pruning bertujuan untuk "memotong" pohon-pohon dari decision tree, mengurangi cabang pengambilan keputusan sehingga mengurangi kemungkinan terjadinya *overfitting*.

Berikut adalah perbandingan klasifikasi Decision Tree normal dan Decision Tree yang menggunakan "Pruning"

6. Decision Tree biasa.

```python
from sklearn.tree import DecisionTreeClassifier

# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

7. Hasil prediksi menggunakan Decision Tree biasa.

```python
from sklearn import metrics

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

8. Menampilkan gambaran dari Decision Tree yang telah dibuat oleh sistem.

```
from matplotlib import pyplot as plt
from sklearn import tree

fig = plt.figure(figsize=(25,20))
tree.plot_tree(clf,filled=True)
```

Dari ilustrasi gambar tersebut, terlihat bahwa cabang pohon decision tree sangat banyak, sehingga sistem yang melakukan klasifikasi akan menjadi lebih sulit.

Dengan melakukan pruning, sistem akan mengelompokkan klasifikasi menjadi lebih spesifik dan mendukung sistem untuk melakukan klasifikasi lebih akurat. Salah satu cara melakukan pruning adalah mengurangi tingkat kedalaman pohon (**max_depth**).

9. Decision Tree dengan pruning.

```
# Create Decision Tree classifer object
clf_p = DecisionTreeClassifier(criterion="entropy", max_depth=3)

# Train Decision Tree Classifer
clf_p = clf_p.fit(X_train,y_train)

#Predict the response for test dataset
y_pred_p = clf_p.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred_p))
```

Dengan membatasi kedalaman pohon hanya ke tingkat 3, dapat dilihat bahwa hasil klasifikasi menjadi lebih baik.

10. Menampilkan gambaran pohon Decision Tree dengan pruning.

```
from matplotlib import pyplot as plt
from sklearn import tree

fig = plt.figure(figsize=(25,20))
tree.plot_tree(clf_p,filled=True)

```

11. Menggunakan Random Forest tanpa pruning.

```python
from sklearn.ensemble import RandomForestClassifier

# Create Decision Tree classifer object
rf = RandomForestClassifier()

# Train Decision Tree Classifer
rf = rf.fit(X_train,y_train)

#Predict the response for test dataset
y_pred_rf = rf.predict(X_test)
```

12. Hasil akurasi Random Forest tanpa pruning.

```python
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred_rf))
```

13. Menampilkan gambaran dari Random Forest tanpa pruning.

```python
fig = plt.figure(figsize=(25,20))
tree.plot_tree(rf.estimators_[0],filled = True);
```

14. Random Forest menggunakan pruning.

```python
from sklearn.ensemble import RandomForestClassifier

# Create Decision Tree classifer object
rf_p = RandomForestClassifier(criterion="entropy", max_depth=3, max_leaf_nodes=5,class_weight=None)

# Train Decision Tree Classifer
rf_p = rf_p.fit(X_train,y_train)

#Predict the response for test dataset
y_pred_rfp = rf_p.predict(X_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred_rfp))
```

Perhatikan, apakah hasil akurasi Random Forest sebelum dan sesudah menggunakan pruning mengalami peningkatan atau penurunan?

15. Menampilkan gambaran dari Random Forest dengan pruning.

```python
fig = plt.figure(figsize=(25,20))
tree.plot_tree(rf_p.estimators_[0],filled = True);
```

Feature Importance

Feature **Importance** merupakan salah satu cara *tuning* Random Forest dengan menentukan variable penting yang mempengaruhi hasil klasifikasi pada Random Forest.

Setelah menemukan variable yang penting dari Random Forest, variable lain yang kurang berpengaruh dapat dihapus dari model sehingga meningkatkan akurasi prediksi model Random Forest.

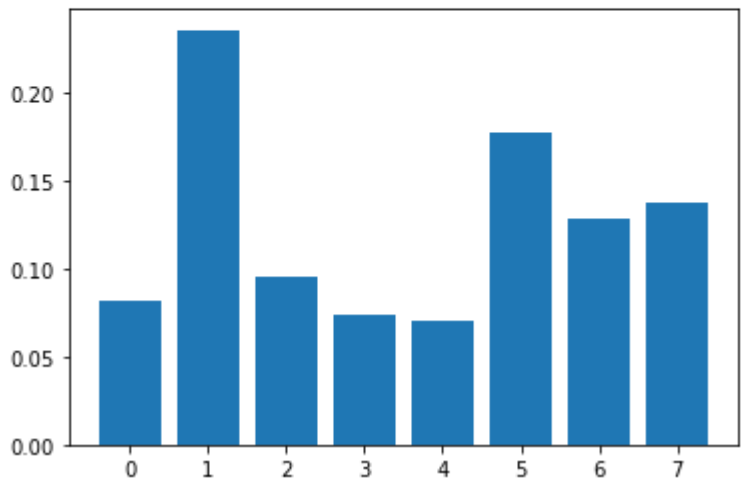
16. Menggunakan feature importance

```
importance =  rf.feature_importances_

for i,v in enumerate(importance):
      print('Feature: %0d, Score: %.5f' % (i,v))

# menampilkan plot feature importance
plt.bar([x for x in range(len(importance))], importance)
plt.show()
```

Perhatikan hasil output. Akan ada daftar setiap fitur/variabel X yang masing – masing memiliki skor kepentingan (importance). Berdasarkan hasil skor tersebut, coba buang 2 kolom yang memiliki skor paling rendah, kemudian tentukan variabel X dengan kolom/fitur sisanya.

Sebagai contoh, dari hasil berikut:

```
Feature: 0, Score: 0.08139
Feature: 1, Score: 0.23573
Feature: 2, Score: 0.09514
Feature: 3, Score: 0.07391
Feature: 4, Score: 0.07038
Feature: 5, Score: 0.17776
Feature: 6, Score: 0.12865
Feature: 7, Score: 0.13703
```



skor terendah adalah fitur 3 dan 4, maka kolom 3 dan 4 dari dataset yaitu **SkinThickness** dan **Insulin** dibuang atau tidak diikutsertakan sebagai X.

```
#split dataset in features and target variable
importance_cols = ['Pregnancies', 'Glucose','BloodPressure','BMI','DiabetesPedigreeFunction', 'Age']
X1 = dataset[importance_cols] # Features
y1 = dataset['Outcome'] # Target variable

# Split dataset into training set and test set
X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.3, random_state=1) # 70% training and 30% test
```

17. Hasil akurasi setelah menggunakan feature importance.

```
# Create Decision Tree classifer object
rf1 = RandomForestClassifier()

# Train Decision Tree Classifer
rf1 = rf1.fit(X1_train,y1_train)

#Predict the response for test dataset
y1_pred = rf1.predict(X1_test)

print("Accuracy:",metrics.accuracy_score(y1_test, y1_pred))
```

Perhatikan, apakah hasil akurasi dengan Random Forest sebelum dan sesudah menggunakan feature importance mengalami kenaikan atau penurunan?

Note: Hasil bisa berbeda untuk setiap mesin yang digunakan mahasiswa.

PENGUMPULAN

1. File yang dikumpulkan terdiri dari:
    a. File project (.ipynb)
    b. File PDF berisi screenshot output dan jawaban (jika ada pertanyaan)
2. File di-compress (.zip) dan diberi nama **KODEMATAKULIAH_KELAS_NIM_NAMA_WEEK KE-XX.zip** (contoh: IS5411_A_13110310017_Monika Evelin Johan_Week-01.zip).

REFERENSI

Deitel, P., & Deitel, H. (2020). Intro to Python for Computer Science and Data Science. Pearson Education.