PERTEMUAN 6

Decision Tree

1. TUJUAN

- Mahasiswa mampu menjelaskan konsep Decision Tree
- Mahasiswa mampu menerapkan konsep Decision tree untuk klasifikasi
- Mahasiswa mampu menerapkan konsep Decision tree untuk regresi

2. MATERI

Pengertian Decision Tree: Decision tree merupakan salah satu model machine learning yang termasuk kedalam supervised learning. Decision tree berbentuk seperti *graph* yang memodelkan keputusan. Digunakan untuk merepresentasikan keputusan dan pengambilan keputusan secara visual dan eksplisit. Decision tree disebut juga dengan istilah CART (*Classification and Regression Trees*) yang diperkenalkan oleh *Leo Breiman* untuk merujuk pada algoritma Decision Tree yang dapat digunakan untuk masalah klasifikasi atau regresi. Algoritma CART menyediakan dasar untuk algoritma penting seperti ***bagged decision trees, random forest and boosted decision trees***.

Model decision tree adalah binary tree dimana tree tersebut terdiri dari beberapa komponen yaitu:

- Setiap **Node** mewakili Fitur (Atribut / variable x).
- Setiap **Branch** atau **Link** atau **Edge** mewakili Keputusan atau Aturan yang disimbolkan dengan panah.
- Setiap **leaf** mewakili Hasil keputusan yang merupakan output variable digunakan sebagai hasil prediksi.

Selain komponen diatas terdapat beberapa istilah yang ada dalam Decision tree, yaitu:

- **Splitting / Pemisahan**: Ini adalah proses membagi node menjadi dua atau lebih sub-node.
- **Decision Node**: Ketika sub-node terpecah menjadi sub-node lebih lanjut, maka itu disebut node keputusan.
- **Pruning / Pemangkasan**: Saat dilakukan penghapusan sub-node dari node keputusan, proses ini disebut pemangkasan.

- **Branch / Sub-Tree**: Sebuah sub-bagian dari seluruh pohon disebut cabang atau sub-pohon.
- **Parent and Child Node (Node Induk dan Anak)**: Node, yang dibagi menjadi beberapa sub-node disebut node induk dari sub-node sedangkan sub-node adalah anak dari node induk.

Pada permasalahan klasifikasi leaf node dari decision tree mewakili kelas sedangkan pada permasalahan regresi nilai variabel respons untuk instance yang terdapat dalam leaf node dapat dirata-ratakan untuk menghasilkan estimasi untuk variabel respons. Cara untuk melakukan prediksi pada decision tree untuk data pengujian hanya perlu mengikuti edge sampai mencapai leaf node.

TAHAPAN ALGORITMA DECISION TREE

Pada Decision tree pembuatan tree merupakan Langkah awal yang harus dilaksanan. Dalam pembuatan tree diperlukan perhitungan menggunakan Information Gain ataupun Gini Index untuk menentukan Root node /node selanjutnya. Berikut merupakah tahapan pembuatan tree secara top-down:

1. Pada iterasi pertama dilakukan pemilihan Root Node didasarkan pada Gini Indeks terendah atau Information Gain tertinggi dari keseluruhan fitur (variable x) pada data training.
2. Kemudian memisahkan himpunan S untuk menghasilkan subset data.
3. Selanjut proses akan diulang untuk mencari decision node yang akan menjadi cabang selanjutnya. Penentuan decision node juga menggunakan Gini Indeks terendah atau Information Gain tertinggi. Akan tetapi yang membedakan adalah fitur yang digunakan adalah fitur yang tidak pernah menggunakan sebelumnya (subset data).
4. Algoritma terus menerus berulang pada setiap subset data hingga menemukan leaf yang merupakan label kelas.

3. KEGIATAN PRAKTIKUM

Persiapan

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Decision Tree untuk Klasifikasi

- Classification-tree

- Urutan pertanyaan if-else tentang fitur individual.
    - Tujuan: menyimpulkan label kelas
    - Mampu menangkap hubungan non-linear antara fitur dan label
    - Tidak memerlukan penskalaan fitur (mis. Standardisasi)
- Decision Regions
    - Decision region: area di bagian fitur di mana semua instance ditugaskan ke satu label kelas
    - Decision Boundary: permukaan yang memisahkan area keputusan yang berbeda

Proses Training – classification tree

Dalam latihan ini Anda akan melakukan training dengan data Wisconsin Breast Cancer Dataset dari UCI machine learning repository. latihan ini akan melakukan prediksi memprediksi apakah tumor ganas atau jinak berdasarkan dua fitur: radius rata-rata tumor (radius_mean) dan jumlah titik cekung rata-rata tumor (concave points_mean).

Preprocess

```
wbc = pd.read_csv('./dataset/wbc.csv')
wbc.head()
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430

5 rows × 33 columns

```
X = wbc[['radius_mean', 'concave points_mean']]
y = wbc['diagnosis']
y = y.map({'M':1, 'B':0})
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

```
from sklearn.tree import DecisionTreeClassifier

# Instantiate DecisionTreeClassifier 'dt' dengan kedalaman maksimum 6
dt = DecisionTreeClassifier(max_depth=6, random_state=1)

# Sesuaikan dt ke set training
dt.fit(X_train, y_train)

# Pmemprediksi label set test
y_pred = dt.predict(X_test)
print(y_pred[0:5])
```

```
[1 0 0 1 0]
```

Mengevaluasi Classification Tree

Sekarang Anda telah menyesuaikan classification tree selanjutnya adalah mengevaluasi kinerjanya pada set pengujian. Anda akan melakukannya menggunakan metrik akurasi yang sesuai dengan fraksi prediksi yang benar yang dibuat pada set pengujian.

```
from sklearn.metrics import accuracy_score

# Memprediksi test set labels
y_pred = dt.predict(X_test)

#  menghitung set accuracy
acc = accuracy_score(y_test, y_pred)
print("Test set accuracy: {:.2f}".format(acc))
```

```
Test set accuracy: 0.89
```

Classification Tree Learning

- Blok diagram dari Decision-Tree
  - Decision-Tree: struktur data yang terdiri dari hierarki node
  - Node: question atau prediction
  - Tiga jenis nodes
    - **Root**: tidak memiliki parent node, question memunculkan dua simpul children nodes.
    - **Internal node**: memiliki satu parent node, question memunculkan dua simpul children nodes.

- **Leaf**: memiliki satu parent node, tidak memiliki children nodes --> prediction.

- Information Gain (IG)

$$IG(\underbrace{f}_{\text{feature}}, \underbrace{sp}_{\text{split-point}}) = I(\text{parent}) - \left(\frac{N_{\text{left}}}{N} I(\text{left}) + \frac{N_{\text{right}}}{N} I(\text{right})\right)$$

  - Kriteria untuk mengukur impurity sebuah node I(node):
    - gini index
    - entropy
    - dst...
- Pembelajaran Classification-Tree
  - Node bertumbuh secara rekursif.
  - Di setiap node, pisahkan data berdasarkan:
    - fitur ***f*** dan split-point ***sp*** untuk memaksimalkan IG (node).
    - jika ***IG(node)*** = 0, deklarasikan node daun

Menggunakan Entropy sebagai Kriteria

Dalam latihan ini, Anda akan melatih classification tree pada Wisconsin Breast Cancer dataset menggunakan entropi sebagai kriteria informasi. Anda akan melakukannya menggunakan keseluruhan dari 30 fitur dalam dataset, yang dibagi menjadi 80% untuk train dan 20% untuk test.

```
from sklearn.tree import DecisionTreeClassifier

# Instantiate dt_entropy, set 'entropy' sebagai kriteria informasi
dt_entropy = DecisionTreeClassifier(max_depth=8, criterion='entropy', random_state=1)

# Fit dt_entropy kedalam training set
dt_entropy.fit(X_train, y_train)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=8, random_state=1)
```

Entropy vs Gini Index

dalam latihan ini Anda akan membandingkan akurasi set tes dt_entropy dengan akurasi tree lain yang bernama dt_gini. sebuah tree dt_gini dilatih pada dataset yang sama menggunakan parameter yang sama kecuali untuk kriteria informasi yang ditetapkan ke indeks gini menggunakan kata kunci 'gini'.

```
dt_gini = DecisionTreeClassifier(max_depth=8, criterion='gini', random_state=1)
dt_gini.fit(X_train, y_train)
```

```
DecisionTreeClassifier(max_depth=8, random_state=1)
```

```
from sklearn.metrics import accuracy_score

# menggunakan dt_entropy untuk memprediksi test set labels
y_pred = dt_entropy.predict(X_test)
y_pred_gini = dt_gini.predict(X_test)

# mengevaluasi accuracy_entropy
accuracy_entropy = accuracy_score(y_test, y_pred)
accuracy_gini = accuracy_score(y_test, y_pred_gini)

# Print accuracy_entropy
print("Accuracy achieved by using entropy: ", accuracy_entropy)

# Print accuracy_gini
print("Accuracy achieved by using gini: ", accuracy_gini)
```

```
Accuracy achieved by using entropy:  0.8947368421052632
Accuracy achieved by using gini:  0.8859649122807017
```

Decision Tree untuk Regression

- Kriteria informasi untuk Regression Tree

$$I(\text{node}) = \underbrace{\text{MSE}(\text{node})}_{\text{mean-squared-error}} = \frac{1}{N_{\text{node}}} \sum_{i \in \text{node}} \left(y^{(i)} - \hat{y}_{\text{node}}\right)^2$$

$$\underbrace{\hat{y}_{\text{node}}}_{\text{mean-target-value}} = \frac{1}{N_{\text{node}}} \sum_{i \in \text{node}} y^{(i)}$$

- Prediction

$$\hat{y}_{\text{pred}}(\text{leaf}) = \frac{1}{N_{\text{leaf}}} \sum_{i \in \text{leaf}} y^{(i)}$$

Train menggunakan Regression Tree

Dalam latihan ini, Anda akan melatih regression tree untuk memprediksi konsumsi bahan bakar dalam satuan *mpg* (miles per gallon) pada mobil menggunakan dataset berikut auto-mpg dataset dengan menggunakan keenam fitur yang tersedia.

Preprocess

```
mpg = pd.read_csv('./dataset/auto.csv')
mpg.head()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino

```
mpg = pd.get_dummies(mpg)
```

```
mpg.head()
```

	mpg	cylinders	displacement	weight	acceleration	model year	origin	horsepower_100	horsepower_102	horsepower_103	...	car name_volvo 145e (sw)	car name_volvo 244dl	car name_volvo 245
0	18.0	8	307.0	3504	12.0	70	1	0	0	0	...	0	0	0
1	15.0	8	350.0	3693	11.5	70	1	0	0	0	...	0	0	0
2	18.0	8	318.0	3436	11.0	70	1	0	0	0	...	0	0	0
3	16.0	8	304.0	3433	12.0	70	1	0	0	0	...	0	0	0
4	17.0	8	302.0	3449	10.5	70	1	0	0	0	...	0	0	0

5 rows × 406 columns

```
X = mpg.drop('mpg', axis='columns')
y = mpg['mpg']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=3)
```

```
from sklearn.tree import DecisionTreeRegressor

# Instantiate dt
dt = DecisionTreeRegressor(max_depth=8, min_samples_leaf=0.13, random_state=3)

# Fit dt kedalam the training set
dt.fit(X_train, y_train)
```

```
DecisionTreeRegressor(max_depth=8, min_samples_leaf=0.13, random_state=3)
```

Mengevaluasi Regression Tree

Dalam latihan ini, Anda akan mengevaluasi kinerja set tes dari dt menggunakan metode *Root Mean Squared Error* (RMSE) metric. RMSE model mengukur rata-rata, seberapa besar prediksi model yang dihasilkan berbeda dari label yang sebenarnya. RMSE model dapat diperoleh dengan menghitung akar kuadrat dari model Mean Squared Error (MSE).

```
from sklearn.metrics import mean_squared_error

# Compute y_pred
y_pred = dt.predict(X_test)

# Compute mse_dt
mse_dt = mean_squared_error(y_test, y_pred)

# Compute rmse_dt
rmse_dt = mse_dt ** (1/2)

# Print rmse_dt
print("Test set RMSE dt: {:.2f}".format(rmse_dt))
```

```
Test set RMSE of dt: 3.68
```

Linear Regression vs Regression Tree

Dalam latihan ini, Anda akan membandingkan set tes RMSE dari dt berdasarkan pada apa yang didapat dari model regresi linier. pada jobsheet juga dibuat instance dari mode linear regression lrdan melatihnya menggunakan dataset yang sama dengan dt.

Preprocess

```
from sklearn.linear_model import LinearRegression

lr = LinearRegression()

lr.fit(X_train, y_train)
```

```
LinearRegression()
```

```
# Predict test set labels
y_pred_lr = lr.predict(X_test)

# Compute mse_lr
mse_lr = mean_squared_error(y_test, y_pred_lr)

# Compute rmse_lr
rmse_lr = mse_lr ** 0.5

# Print rmse_lr
print("Linear Regression test set RMSE: {:.2f}".format(rmse_lr))

# Print rmse_dt
print("Regression Tree test set RMSE: {:.2f}".format(rmse_dt))
```

```
Linear Regression test set RMSE: 4.07
Regression Tree test set RMSE: 3.68
```

4. TUGAS

Lakukanlah proses classification learning Decision Tree dengan ketentuan, sebagai berikut:

1) Menggunakan data mushrooms.csv
2) Evaluasi hasilnya!