In [ ]:

```
# import library

import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

In [ ]:

```
# data training di definisikan

dt_train = {
  'no'                : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
  'warna'             : ['Hitam', 'Putih', 'Cokelat', 'Hitam', 'Hitam', 'Cokelat', 'Cokelat', 'Putih', 'Putih', 'Cokelat'],
  'bulu'              : ['Panjang', 'Panjang', 'Pendek', 'Pendek', 'Gimbal', 'Gimbal', 'Panjang', 'Pendek', 'Pendek', 'Gimbal'],
  'makanan_kesukaan'  : ['Wortel', 'Wortel', 'Kangkung', 'Kangkung', 'Rumput', 'Rumput', 'Rumput', 'Wortel', 'Kangkung', 'Kangkung'],
  'jenis'             : ['A', 'A', 'B', 'B', 'C', 'A', 'B', 'B', 'C', 'C'],
}
```

In [ ]:

```
# data testing di definisikan

dt_test = {
  'no'                : [1, 2, 3, 4, 5],
  'warna'             : ['Hitam', 'Putih', 'Cokelat', 'Putih', 'Cokelat'],
  'bulu'              : ['Gimbal', 'Pendek', 'Panjang', 'Panjang', 'Gimbal'],
  'makanan_kesukaan'  : ['Rumput', 'Wortel', 'Kangkung', 'Rumput', 'Kangkung'],
  'prediksi_jenis'    : ['?', '?', '?', '?', '?'],
  'jenis_sebenarnya'  : ['A', 'B', 'C', 'C', 'B']
}
```

In [ ]:

```
# membandingkan dt_training dengan dt_testing

dt_train  = pd.DataFrame(dt_train)
dt_test   = pd.DataFrame(dt_test)
dt_hasil  = dt_test.copy()
```

In [ ]:

```
# run dt_training

dt_train
```

Out[ ]:

	no	warna	bulu	makanan_kesukaan	jenis
0	1	Hitam	Panjang	Wortel	A
1	2	Putih	Panjang	Wortel	A
2	3	Cokelat	Pendek	Kangkung	B
3	4	Hitam	Pendek	Kangkung	B
4	5	Hitam	Gimbal	Rumput	C
5	6	Cokelat	Gimbal	Rumput	A
6	7	Cokelat	Panjang	Rumput	B
7	8	Putih	Pendek	Wortel	B
8	9	Putih	Pendek	Kangkung	C
9	10	Cokelat	Gimbal	Kangkung	C

In [ ]:

```
# run dt_testing

dt_test
```

Out[ ]:

	no	warna	bulu	makanan_kesukaan	prediksi_jenis	jenis_sebenarnya
0	1	Hitam	Gimbal	Rumput	?	A
1	2	Putih	Pendek	Wortel	?	B
2	3	Cokelat	Panjang	Kangkung	?	C
3	4	Putih	Panjang	Rumput	?	C
4	5	Cokelat	Gimbal	Kangkung	?	B

In [ ]:

```
# dt_train mengubah label kategori manjadi nilai number

for cols in dt_train.select_dtypes(include='object') :
  dt_train[cols] = dt_train[cols].astype('category').cat.codes
```

In [ ]:

```
# dt_test mengubah label kategori manjadi nilai number

for cols in dt_test.select_dtypes(include='object') :
  dt_test[cols] = dt_test[cols].astype('category').cat.codes
```

In [ ]:

```
# run dt_training setelah diubah menjadi number

dt_train
```

Out[ ]:

	no	warna	bulu	makanan_kesukaan	jenis
0	1	1	1	2	0
1	2	2	1	2	0
2	3	0	2	0	1
3	4	1	2	0	1
4	5	1	0	1	2
5	6	0	0	1	0
6	7	0	1	1	1
7	8	2	2	2	1
8	9	2	2	0	2
9	10	0	0	0	2

In [ ]:

```
# run dt_testing setelah diubah menjadi number

dt_test
```

Out[ ]:

	no	warna	bulu	makanan_kesukaan	prediksi_jenis	jenis_sebenarnya
0	1	1	0	1	0	0
1	2	2	2	2	0	1
2	3	0	1	0	0	2
3	4	2	1	1	0	2
4	5	0	0	0	0	1

In [ ]:

```
# set feature dan target variabel

feature_cols = ['warna', 'bulu', 'makanan_kesukaan']
x_train = dt_train[feature_cols]
y_train = dt_train.jenis
x_test  = dt_test[feature_cols]
y_test  = dt_test.jenis_sebenarnya
```

In [ ]:

```
# running decision tree algorithm

clf = DecisionTreeClassifier(max_depth=2)
clf = clf.fit(x_train, y_train)
```

In [ ]:

```
y_prediksi  = clf.predict(x_test)
result      = confusion_matrix(y_test, y_prediksi)
print("Confusion Matrix:")
print(result)
result1     = classification_report(y_test, y_prediksi)
print("Classification Report:")
print(result1)
result2     = accuracy_score(y_test, y_prediksi)
print("Accuracy:", result2)
```

```
Confusion Matrix:
[[0 0 1]
 [0 1 1]
 [0 0 2]]
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00         1
           1       1.00      0.50      0.67         2
           2       0.50      1.00      0.67         2

    accuracy                           0.60         5
   macro avg       0.50      0.50      0.44         5
weighted avg       0.60      0.60      0.53         5

Accuracy: 0.6
D:\program\Python39\lib\site-packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
D:\program\Python39\lib\site-packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
D:\program\Python39\lib\site-packages\sklearn\metrics\_classification.py:1308: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [ ]:

```
# running naive bayes algorithm

clf2 = MultinomialNB()
clf2 = clf2.fit(x_train, y_train)
```

In [ ]:

```
y_prediksi2 = clf2.predict(x_test)
result      = confusion_matrix(y_test, y_prediksi2)
print("Confusion Matrix:")
print(result)
result1     = classification_report(y_test, y_prediksi2)
print("Classification Report:")
print(result1)
result2     = accuracy_score(y_test, y_prediksi2)
print("Accuracy:", result2)
```

```
Confusion Matrix:
[[1 0 0]
 [0 2 0]
 [0 1 1]]
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         1
           1       0.67      1.00      0.80         2
           2       1.00      0.50      0.67         2

    accuracy                           0.80         5
   macro avg       0.89      0.83      0.82         5
weighted avg       0.87      0.80      0.79         5

Accuracy: 0.8
```

In [ ]:

```
# hasil setelah dilakukannya decision tree dan naive bayes

# mengambil variabel dari algoritma decision tree
dt_hasil['prediksi_jenis_decision_tree']  = y_prediksi
# mengambil variabel dari algoritma naive bayes
dt_hasil['prediksi_jenis_naive_bayes']    = y_prediksi2

dt_hasil['prediksi_jenis_decision_tree'].replace({0:'A', 1:'B', 2:'C'}, inplace=True)
dt_hasil['prediksi_jenis_naive_bayes'].replace({0:'A', 1:'B', 2:'C'}, inplace=True)
```

In [ ]:

```
# run dt_hasil

dt_hasil
```

Out[ ]:

	no	warna	bulu	makanan_kesukaan	prediksi_jenis	jenis_sebenarnya	prediksi_jenis_decision_tree	prediksi_jenis_naive_bayes
0	1	Hitam	Gimbal	Rumput	?	A	C	A
1	2	Putih	Pendek	Wortel	?	B	B	B
2	3	Cokelat	Panjang	Kangkung	?	C	C	B
3	4	Putih	Panjang	Rumput	?	C	C	C
4	5	Cokelat	Gimbal	Kangkung	?	B	C	B