Deskripsi Singkat dataset

Wheat Seed merupakan dataset yang terdiri atas 8 kolom dan 199 baris. Kolom terdiri atas 7 feature dan 1 target, seluruh data di dalam Wheat Seed merupakan data numerik. Dataset ini berisi mengenai pengukuran terhadap gandum mulai dari area tanam, perimeter, compactness, ukuran bulir gandum, dan lain lain. Berdasarkan variabel tersebut, biji gandum diklasifikasikan menjadi tiga tipe, dilambangkan dengan angka 1, 2, dan 3.

Dataset yang diambil dari Repositori

1. Wheat Seeds – sumber : https://www.kaggle.com/jmcaro/wheat-seedsuci

Hasil kerja

<u>Naïve Bayes</u>

1. Import Data

   Tahap pertama yang dilakukan adalah memanggil libraries yang dibutuhkan. Pada bagian ini, library yang digunakan adalah pandas, numpy, mglearn, sklearn, matplotlib, dan seaborn. Library ini memiliki fungsi yang berbeda-beda, mulai dari pengambilan data hingga modelling

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()
import pandas as pd
import mglearn
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import make_scorer, accuracy_score,precision_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score ,precision_score,recall_score,f1_score
```

2. Read Data

   Dataset yang digunakan adalah data mengenai biji gandum yang terdiri atas 199 baris, 7 features, dan 1 target. Target pada dataset ini merupakan variabel tipe yang berisi tiga kelas biji gandum, dilambangkan dengan angka 1, 2, dan 3.

```
In [4]: df.describe()
```

Out[4]:

	Area	Perimeter	Compactness	Kernel.Length	Kernel.Width	Asymmetry.Coeff	Kernel.Groove	Type
count	199.000000	199.000000	199.000000	199.000000	199.000000	199.000000	199.000000	199.000000
mean	14.918744	14.595829	0.870811	5.643151	3.265533	3.699217	5.420653	1.994975
std	2.919976	1.310445	0.023320	0.443593	0.378322	1.471102	0.492718	0.813382
min	10.590000	12.410000	0.808100	4.899000	2.630000	0.765100	4.519000	1.000000
25%	12.330000	13.470000	0.857100	5.267000	2.954500	2.570000	5.046000	1.000000
50%	14.430000	14.370000	0.873400	5.541000	3.245000	3.631000	5.228000	2.000000
75%	17.455000	15.805000	0.886800	6.002000	3.564500	4.799000	5.879000	3.000000
max	21.180000	17.250000	0.918300	6.675000	4.033000	8.315000	6.550000	3.000000

3. Data Preparation

   a. Missing value

      Tahap pertama pada data preparation adalah pengecekan terhadap nilai NA (missing value) dengan fungsi df.isna().sum(). Pada kasus Wheat Seeds, tidak

ada data NA yang ditemukan, artinya dataset ini tidak perlu melalui proses pembuangan atau penggantian nilai NA.

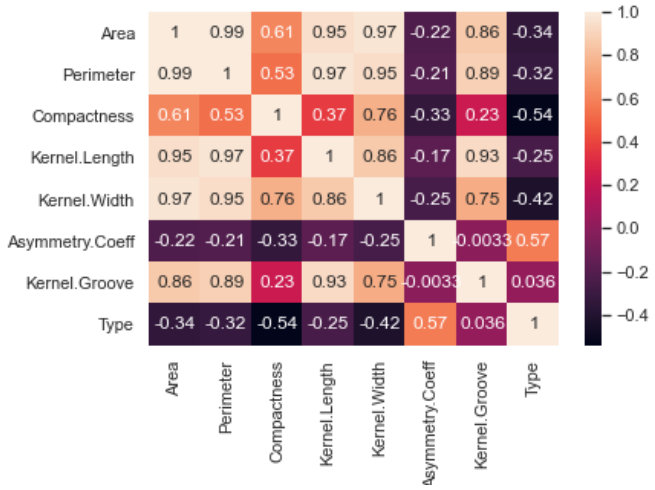
```
In [5]: df.isna().sum()
```

```
Out[5]: Area              0
        Perimeter         0
        Compactness       0
        Kernel.Length     0
        Kernel.Width      0
        Asymmetry.Coeff   0
        Kernel.Groove     0
        Type              0
        dtype: int64
```

b. Feature selection

Tahap kedua pada persiapan data adalah feature selection atau pemiliha fitur-fitur yang penting untuk analisis. Feature selection dilakukan dengan pengecekan korelasi atau hubungan antar seluruh variabel. Hasil korelasi divisualisasikan ke dalam heatmap seperti di bawah ini. Dapat dilihat bahwa variabel asymmetry coeff memiliki hubungan yang paling rendah dengan variabel-variabel lainnya.

```
corrMatrix = df.corr()
sns.heatmap(corrMatrix, annot=True)
plt.show()
```



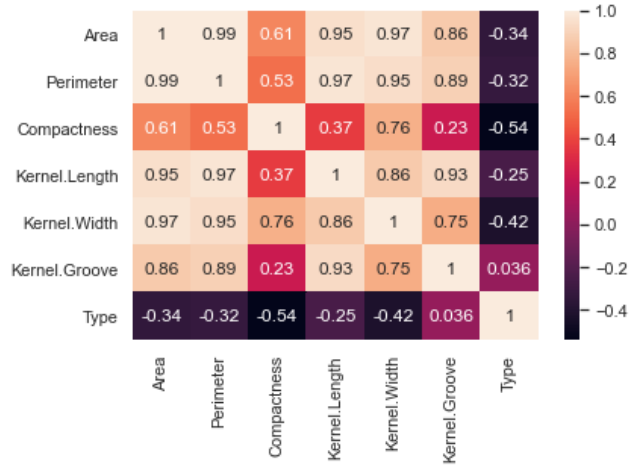
Mengetahui hal tersebut, dilakukan penghapusan fitur asymmetry coeff agar tidak merusak model yang akan dibuat. Hasilnya adalah sebagai berikut:

```
In [7]: del df['Asymmetry.Coeff']
        df.head()
```

Out[7]:

	Area	Perimeter	Compactness	Kernel.Length	Kernel.Width	Kernel.Groove	Type
0	15.26	14.84	0.8710	5.763	3.312	5.220	1
1	14.88	14.57	0.8811	5.554	3.333	4.956	1
2	14.29	14.09	0.9050	5.291	3.337	4.825	1
3	13.84	13.94	0.8955	5.324	3.379	4.805	1
4	16.14	14.99	0.9034	5.658	3.562	5.175	1

```
corrMatrix = df.corr()
sns.heatmap(corrMatrix, annot=True)
plt.show()
```



4. Modelling: Naïve Bayes
    a. Split feature & target

    Dataset Wheat Seeds terbagi menjadi dua, yaitu features dan target. Kedua jenis variabel ini akan dipisahkan pada variabel yang terpisah untuk kebutuhan modelling. Feature (X) terdiri atas area, perimeter, compactness, kernel length, kernel width, dan kernel groove. Sedangkan target (y) terdisi atas variabel tipe.

In [9]:
```
y = df['Type']
X = df.drop(columns='Type')

X.head()
```

Out[9]:

	Area	Perimeter	Compactness	Kernel.Length	Kernel.Width	Kernel.Groove
0	15.26	14.84	0.8710	5.763	3.312	5.220
1	14.88	14.57	0.8811	5.554	3.333	4.956
2	14.29	14.09	0.9050	5.291	3.337	4.825
3	13.84	13.94	0.8955	5.324	3.379	4.805
4	16.14	14.99	0.9034	5.658	3.562	5.175

    b. Split training & testing

    Pada bagain ini, data feature dan target dipecah menjadi 70% data training serta 30% testing menggunakan fungsi train_test_split(). Jumlah masing-masing data testing dan training terlampir di bawah ini:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    random_state=0,
                                                    train_size = 0.7)
```

```
print("Jumlah baris: " + str(len(X)))
print("Jumlah train: " + str(len(X_train)))
print("Jumlah test: " + str(len(X_test)))
```

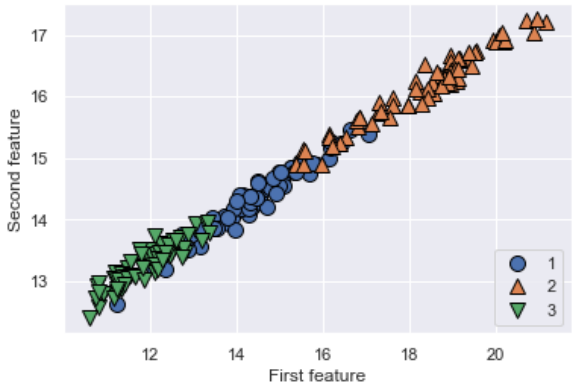
```
Jumlah baris: 199
Jumlah train: 139
Jumlah test: 60
```

c. Scatter plot

Pada bagian ini, dibentuk scatter plot berdasarkan 2 kolom area dan perimeter, setiap data point ditandai dengan angka dan shape yang berbeda-beda sesuai kelas 1, 2, dan 3. Pada visualisasi ini, dapat dilihat bahwa hubungan keduanya berifat linear dan mempengaruhi secara kuat.

```
mglearn.discrete_scatter(X['Area'], X['Perimeter'], y)
plt.legend(["1", "2", "3"], loc=4)
plt.xlabel("First feature")
plt.ylabel("Second feature")
print("X.shape: {}".format(X.shape))
```

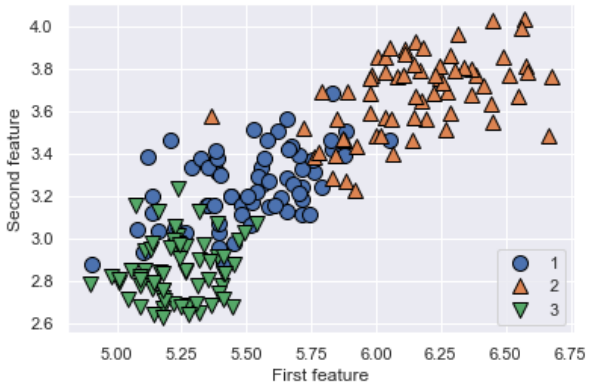
```
X.shape: (199, 6)
```



Selanjutnya, dilakukan visualisasi terhadap variabel kernel length dan kernel width sebagai ukuran butir gandum. Variabel ini yang nantinya akan menjadi penentu untuk klasifikasi. Hasilnya adalah sebaga berikut, terlihat bahwa antar kelas berkumpul dan membentuk kelompoknya sendiri.

```
mglearn.discrete_scatter(X['Kernel.Length'], X['Kernel.Width'], y)
plt.legend(["1", "2", "3"], loc=4)
plt.xlabel("First feature")
plt.ylabel("Second feature")
print("X.shape: {}".format(X.shape))
```

```
X.shape: (199, 6)
```



5. Naïve Bayes

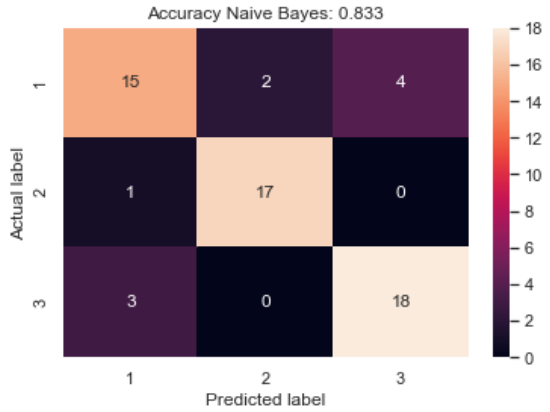
a. Modelling

Pada bagian ini, data training berupa variabel X_train dan y_train dibuat ke dalam model Naïve Bayes dengan fungsi GaussianNB(). Model dimasukkan ke dalam variabel gaussian, sedangkan hasil prediksi dimasukkan ke dalam

variabel Y_pred. Dilakukan juga pengecekan akurasi prediksi klasifikasi Naïve Bayes dan visualisasi melalui confusion matrix untuk melihat performa dari model. Dapat dilihat bahwa hasil akurasi mencapai angka 83% yang tergolong baik. Secara lebih spesifik, model ini melakukan kesalahan klasifikasi sebanyak 10 data dari 60 data.

```
gaussian = GaussianNB()
gaussian.fit(X_train, y_train)
Y_pred = gaussian.predict(X_test)
accuracy_nb=round(accuracy_score(y_test,Y_pred)* 100, 2)
acc_gaussian = round(gaussian.score(X_train, y_train) * 100, 2)
```

```
from sklearn.metrics import confusion_matrix

conf = confusion_matrix(y_test, Y_pred)
# Transform to df for easier plotting
conf_seeds = pd.DataFrame(conf,
                     index = ['1','2','3'],
                     columns = ['1','2','3'])

sns.heatmap(conf_seeds, annot=True)
plt.title('Accuracy Naive Bayes: {0:.3f}'.format(accuracy_score(y_test,Y_pred)))
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```



b. Accuracy

Hasil prediksi Wheat Seeds berupa kelas dengan nilai 1, 2, dan 3 yang merupakan representasi dari tipe biji gandum. Variabel y_test dan Y_pred kemudian dibandingkan untuk mengetahui seberapa akurat prediksi dari Naïve Bayes. Akurasi mendapatkan hasil 83,33%.

```
accuracy = accuracy_score(y_test,Y_pred)
precision =precision_score(y_test, Y_pred,average='micro')
recall =  recall_score(y_test, Y_pred,average='micro')
f1 = f1_score(y_test,Y_pred,average='micro')

print('accuracy_Naive Bayes: %.3f' %accuracy)
print('precision_Naive Bayes: %.3f' %precision)
print('recall_Naive Bayes: %.3f' %recall)
print('f1-score_Naive Bayes : %.3f' %f1)
```

```
accuracy_Naive Bayes: 0.833
precision_Naive Bayes: 0.833
recall_Naive Bayes: 0.833
f1-score_Naive Bayes : 0.833
```

SVM (Support Vector Machine)

1. Import library
   Pada bagian ini, ditambahkan beberapa library yang akan digunakan untuk analisis.

```
In [15]:
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats

import seaborn as sns; sns.set()
```

2. Read data
   Data yang sama dipanggil kembali untuk diolah dengan algoritma yang berbeda

```
In [16]:
df.head()
```

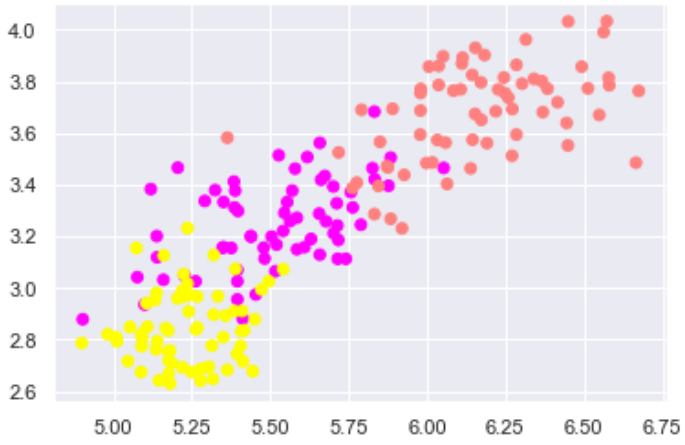
Out[16]:

	Area	Perimeter	Compactness	Kernel.Length	Kernel.Width	Kernel.Groove	Type
0	15.26	14.84	0.8710	5.763	3.312	5.220	1
1	14.88	14.57	0.8811	5.554	3.333	4.956	1
2	14.29	14.09	0.9050	5.291	3.337	4.825	1
3	13.84	13.94	0.8955	5.324	3.379	4.805	1
4	16.14	14.99	0.9034	5.658	3.562	5.175	1

3. Scatter plot
   Scatter plot ini sama dengan scatter plot sebelumnya, terlihat bahwa tipe 1, 2, dan 3 terpisah membentuk kelompok, meskipun masih ada beberapa data point yang tidak biasa (outlier).

```
In [27]:
plt.scatter(X['Kernel.Length'], X['Kernel.Width'], c = y, cmap='spring')
```

```
Out[27]: <matplotlib.collections.PathCollection at 0x235ffae7490>
```



4. Support Vector Machine
   a. Modelling

Pada bagian ini, data training berupa variabel X_train dan y_train dibuat ke dalam model SVM dengan fungi SVC(). Model dimasukkan ke dalam variabel model, sedangkan hasil prediksi dimasukkan ke dalam variabel y_pred. y_pred merupakan variabel yang menampung hasil prediksi untuk data testing

```
In [22]:
from sklearn.svm import SVC
model = SVC(kernel = 'linear', C=1E10)
model.fit(X_train, y_train)
```

```
Out[22]: SVC(C=10000000000.0, kernel='linear')
```

```
In [57]:
y_pred = model.predict(X_test)
y_pred
```

```
Out[57]: array([1, 3, 2, 2, 3, 3, 1, 3, 1, 3, 1, 2, 3, 3, 2, 1, 1, 2, 1, 2, 3, 3,
       1, 3, 1, 2, 3, 3, 2, 1, 1, 1, 2, 2, 3, 1, 3, 1, 3, 3, 3, 2, 1, 2,
       3, 1, 1, 2, 2, 3, 2, 3, 2, 2, 3, 1, 1, 2, 1, 1], dtype=int64)
```

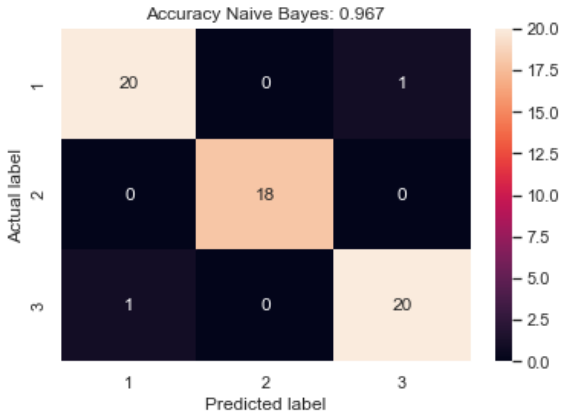
b. Akurasi

Pengecekan akurasi untuk SVM juga dilakukan. Pada algoritma ini, didapatkan akurasi prediksi sebesar 96,67% (sangat baik) dengan dataset dan variabel yang sama. Detail prediksi klasifikasi tipe dapat dilihat menggunakan confusion matrix. Secara umum, terlihat bahwa model SVM dapat membedakan tipe dengan sangat benar dan akurat, hanya ada 1 data point yang kurang tepat ditebak.

```
from sklearn import metrics

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

```
Accuracy: 0.9666666666666667
```

```
from sklearn.metrics import confusion_matrix

conf = confusion_matrix(y_test, y_pred)
# Transform to df for easier plotting
conf_seeds = pd.DataFrame(conf,
                          index = ['1','2','3'],
                          columns = ['1','2','3'])

sns.heatmap(conf_seeds, annot=True)
plt.title('Accuracy Naive Bayes: {0:.3f}'.format(accuracy_score(y_test,y_pred)))
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```



Simpulan kerja

Wheat Seeds merupakan dataset mengenai klasifikasi tipe biji gandum berdasarkan data penelitian terkait gandum. Data ini terdiri atas 199 baris dan 8 kolom yang terbagi menjadi 7 features dan 1 target. Data ini diproses menggunakan algoritma Naïve Bayes dan SVM untuk mengetahui klasifikasi seeds menjadi kelompok tipe tertentu. Dataset Wheat Seeds terlebih dahulu mengalami proses persiapan atau pre-processing, proses yang dilakukan adalah:

a. Pengecekan nilai NA (missing value) yang mendapatkan hasil bahwa dataset Wheat Seeds tidak memiliki nilai NA. Tahapan penghapusan atau penggantian missing value tidak perlu dilakukan
b. Penghapusan kolom dengan korelasi rendah

Setelah melalui tahap pre-processing sederhana, selanjutnya dilakukan eksplorasi sederhana dengan membentuk scatter plot. Scatter plot membagi Wheat Seeds menjadi 3 tipe, yaitu 1, 2, dan 3. Algoritma Naïve Bayes dan SVM diguanakan untuk membagi Wheat Seeds menjadi kelompok tipe, hasil model adalah sebagai berikut:

a. Naïve Bayes: model memiliki akuasi 83,33%, yang berarti akurasi prediksi model ini tergolong baik. Model ini melakukan kesalahan penebakan sebanyak 10 data point.
b. SVM: model memiliki akurasi 96,67%, yang berarti akurasi sangat tinggi dan sangat baik. Model dengan algoritma ini hanya melakukan satu kesalahan klasifikasi.

Secara umum, klasifikasi tipe Wheat Seeds dilakukan dengan baik oleh kedua algoritma, namun penggunaan SVM menghasilkan tingkat akurasi yang jauh lebih baik. Hal ini dibuktikan dari kemampuan klasifikasi yang lebih tepat dibandingkan Naïve Bayes, sehingga dalam kasus Wheat Seeds, algoritma SVM lebih direkomendasikan.