In [1]:

```
# nama : donny pratama
# kelas : 5P52
# nim : 20.240.0116
# petemuan 9
```

In [8]:

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

from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay
```

In [11]:

```
columns = ["sepal-length", "sepal-width", "petal-length", "petal-width","class"]
df = pd.read_csv("D:\iris.data.csv", names=columns)
```

In [12]:

```
df.describe()
```

Out[12]:

	sepal-length	sepal-width	petal-length	petal-width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [13]:

```
df.head()
```

Out[13]:

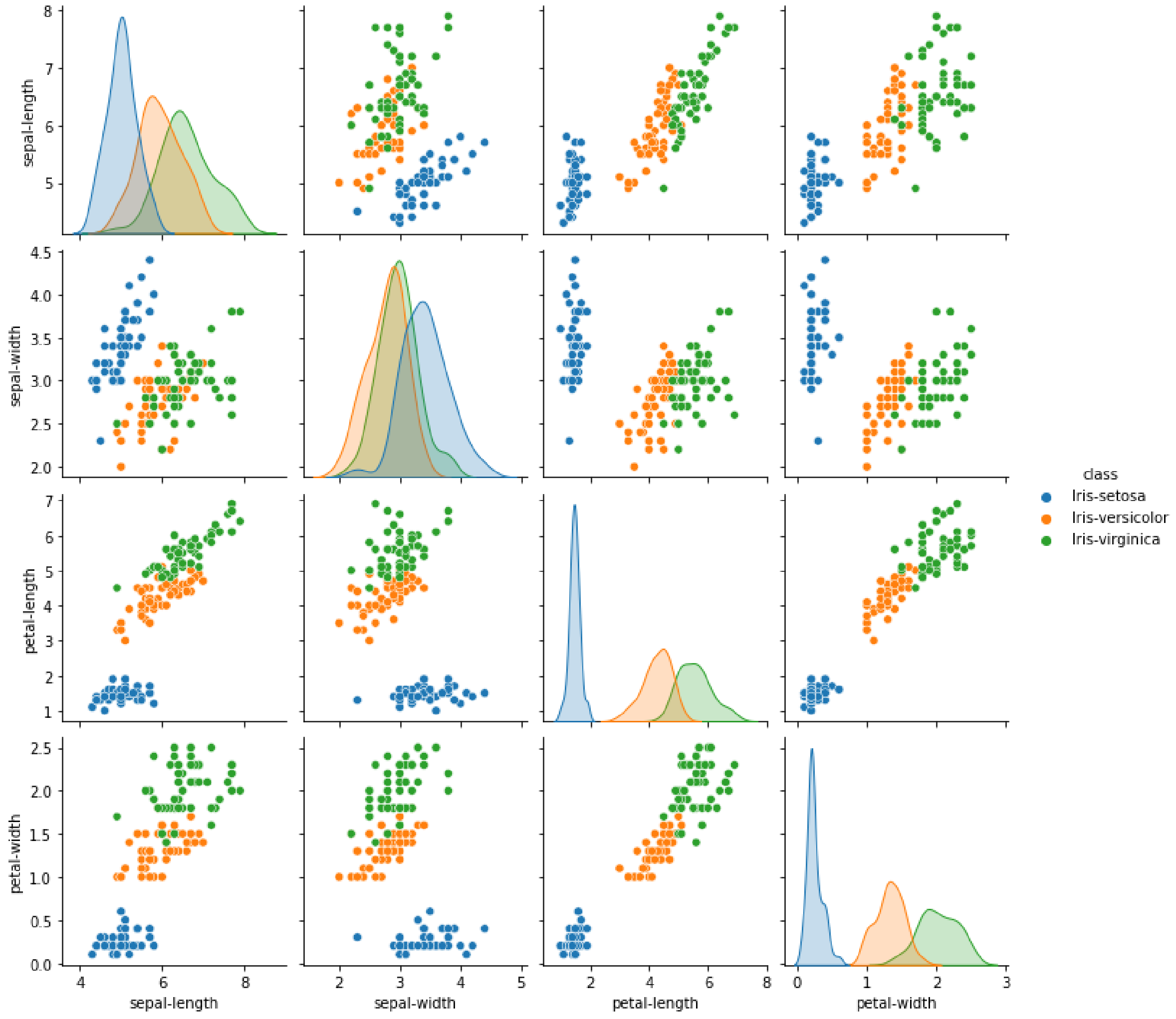
	sepal-length	sepal-width	petal-length	petal-width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In [16]:

```
# memvisualisasikan data
sns.pairplot(df, hue='class')
```

Out[16]:

```
<seaborn.axisgrid.PairGrid at 0x1b77cd4d130>
```



In [17]:

```
X = df.iloc[:, :-1].values
y = df.iloc[:,  4].values
```

In [19]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle = True, st
```

In [20]:

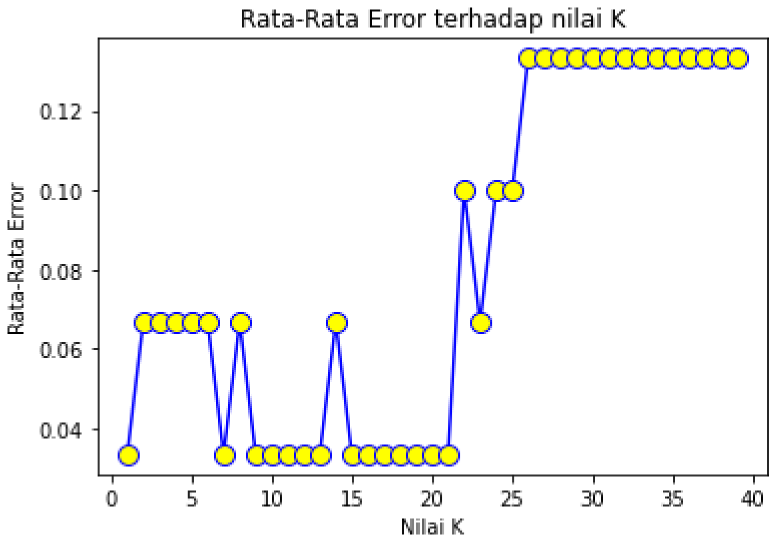
```
lb = LabelEncoder()
lb.fit(y_train)

y_train = lb.transform(y_train)
y_test = lb.transform(y_test)
```

In [21]:

```
# standarisasi
scaler = StandardScaler()
scaler.fit(X_train)

X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

In [27]:

```
error = []

for i in range(1,40):
    knn = KNeighborsClassifier(n_neighbors = i)
    knn.fit(X_train, y_train)

    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))

plt.plot(range(1, 40), error, color='blue', marker='o',
         markerfacecolor='yellow', markersize=10)
plt.title('Rata-Rata Error terhadap nilai K')
plt.xlabel('Nilai K')
plt.ylabel('Rata-Rata Error')
plt.show()
```



In [28]:

```
classifier = KNeighborsClassifier(n_neighbors=4)
classifier.fit(X_train, y_train)
```

Out[28]:

```
KNeighborsClassifier(n_neighbors=4)
```

In [29]:

```
y_pred = classifier.predict(X_test)
print(classification_report(y_test, y_pred, target_names=lb.classes_))
```

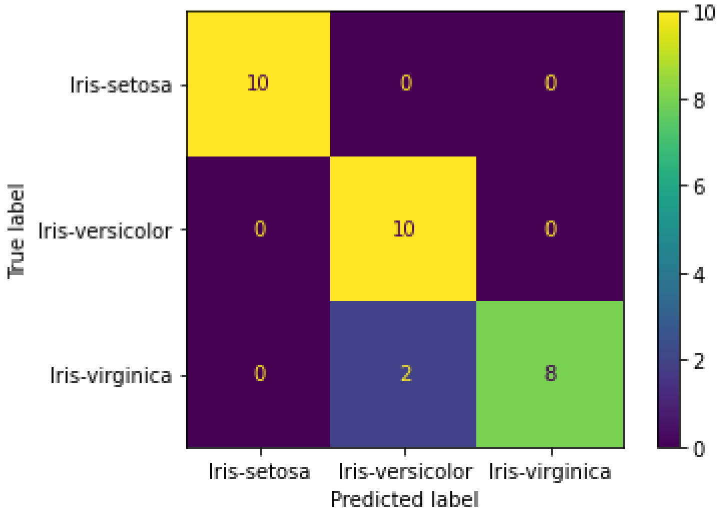
```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       0.83      1.00      0.91        10
 Iris-virginica       1.00      0.80      0.89        10

       accuracy                           0.93        30
      macro avg       0.94      0.93      0.93        30
   weighted avg       0.94      0.93      0.93        30
```

In [30]:

```
cm = confusion_matrix(y_test, y_pred)
ConfusionMatrixDisplay(cm, display_labels=lb.classes_).plot()
```

Out[30]:

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1b77ec61
820>
```



In [ ]:

```

```