KNN Irish Data

Ika Lulus Yuliatin

In [ ]:

```
# This Python 3 environment comes with many helpful analytics libraries installec
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/dock
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list c

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets
# You can also write temporary files to /kaggle/temp/, but they won't be saved ou
```

In [ ]:

```
from google.colab import files
data_to_load = files.upload()
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Irish.csv to Irish.csv
```

In [ ]:

```
df = pd.read_csv('Irish.csv')
df.head()
```

Out[4]:

	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 [ ]:

```
df.shape
```

Out[5]:

```
(150, 5)
```

In [ ]:

```
df.info
```

Out[6]:

```
<bound method DataFrame.info of      sepal length  sepal width  petal length  p
etal 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
..             ...          ...           ...          ...             ...
145            6.7          3.0           5.2          2.3  Iris-virginica
146            6.3          2.5           5.0          1.9  Iris-virginica
147            6.5          3.0           5.2          2.0  Iris-virginica
148            6.2          3.4           5.4          2.3  Iris-virginica
149            5.9          3.0           5.1          1.8  Iris-virginica

[150 rows x 5 columns]>
```

In [ ]:

```
df.describe
```

Out[7]:

```
<bound method NDFrame.describe of      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
..             ...          ...           ...          ...             ...
145            6.7          3.0           5.2          2.3  Iris-virginica
146            6.3          2.5           5.0          1.9  Iris-virginica
147            6.5          3.0           5.2          2.0  Iris-virginica
148            6.2          3.4           5.4          2.3  Iris-virginica
149            5.9          3.0           5.1          1.8  Iris-virginica

[150 rows x 5 columns]>
```

In [ ]:

```
df['class'].value_counts()
```

Out[8]:

```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: class, dtype: int64
```

In [ ]:

```
df.isnull().sum()
```

Out[9]:

```
sepal length    0
sepal width     0
petal length    0
petal width     0
class           0
dtype: int64
```

In [ ]:

```
data = df.drop_duplicates(subset = "class")
```

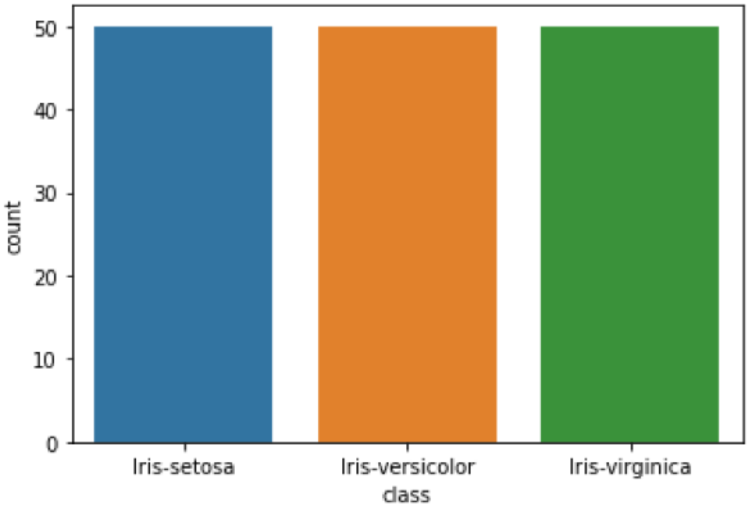
```
In [ ]: data
```

Out[11]:

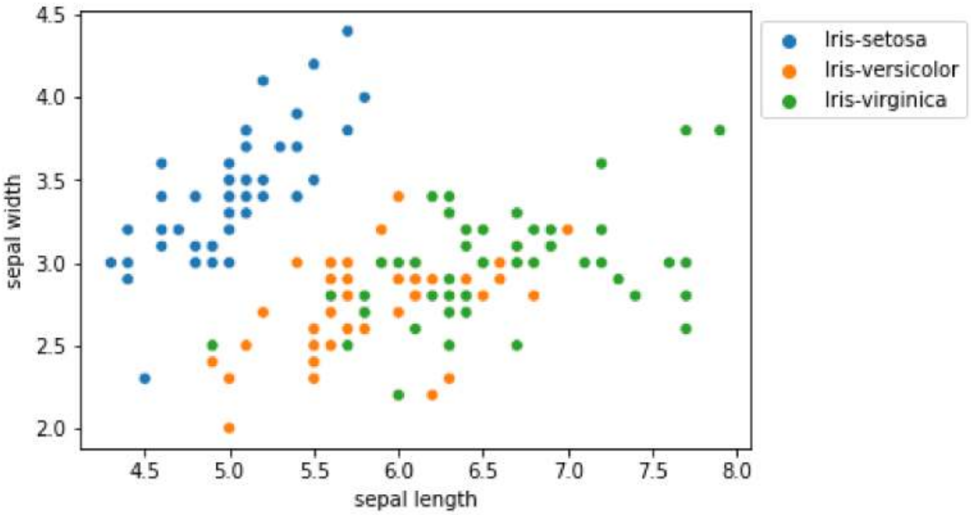
	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
50	7.0	3.2	4.7	1.4	Iris-versicolor
100	6.3	3.3	6.0	2.5	Iris-virginica

```
In [ ]: import matplotlib.pyplot as plt
        import seaborn as sns
```

```
In [ ]: sns.countplot(x='class', data=df )
        plt.show()
```



```
In [ ]: sns.scatterplot(x='sepal length', y='sepal width', hue='class', data=df,)
        plt.legend(bbox_to_anchor=(1, 1), loc=2)
        plt.show()
```



In [ ]:

```
from sklearn.model_selection import train_test_split
X = df.drop(columns=['class'])
Y = df['class']
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.30)
# knn - k-nearest neighbours
# By default the value of n_neighbors(k) = 5
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier()
#train model using fit funtion
model.fit(x_train, y_train)
```

Out[15]:

```
KNeighborsClassifier()
```

In [ ]:

```
predictions=model.predict(x_test)
```

In [ ]:

```
from sklearn.metrics import accuracy_score
print("Accuracy: ",accuracy_score(y_test,predictions)*100 ,"%")
```

```
Accuracy:  100.0 %
```