KNN (K Nearest Neighbors) Classification

```
1 import pandas as pd
2 from sklearn.datasets import load_iris
3 iris = load_iris()
```

```
1 iris.feature_names
```

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

```
1 iris.target_names
```

```
array(['setosa', 'versicolor', 'virginica'], dtype='<U10')
```

```
1 df = pd.DataFrame(iris.data,columns=iris.feature_names)
2 df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
1 df['target'] = iris.target
2 df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

```
1 df[df.target==1].head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
50	7.0	3.2	4.7	1.4	1
51	6.4	3.2	4.5	1.5	1
52	6.9	3.1	4.9	1.5	1
53	5.5	2.3	4.0	1.3	1
54	6.5	2.8	4.6	1.5	1

```
1 df[df.target==2].head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
100	6.3	3.3	6.0	2.5	2
101	5.8	2.7	5.1	1.9	2
102	7.1	3.0	5.9	2.1	2
103	6.3	2.9	5.6	1.8	2
104	6.5	3.0	5.8	2.2	2

```
1 df['flower_name'] =df.target.apply(lambda x: iris.target_names[x])
2 df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target	flower_name
0	5.1	3.5	1.4	0.2	0	setosa
1	4.9	3.0	1.4	0.2	0	setosa
2	4.7	3.2	1.3	0.2	0	setosa
3	4.6	3.1	1.5	0.2	0	setosa
4	5.0	3.6	1.4	0.2	0	setosa

```
1 df[45:55]
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target	flower_name
45	4.8	3.0	1.4	0.3	0	setosa
46	5.1	3.8	1.6	0.2	0	setosa
47	4.6	3.2	1.4	0.2	0	setosa
48	5.3	3.7	1.5	0.2	0	setosa
49	5.0	3.3	1.4	0.2	0	setosa
50	7.0	3.2	4.7	1.4	1	versicolor
51	6.4	3.2	4.5	1.5	1	versicolor
52	6.9	3.1	4.9	1.5	1	versicolor
53	5.5	2.3	4.0	1.3	1	versicolor
54	6.5	2.8	4.6	1.5	1	versicolor

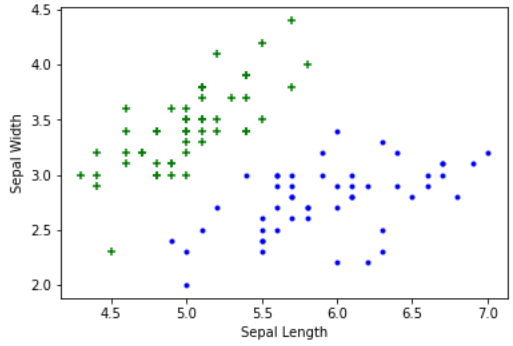
```
1 df0 = df[:50]
2 df1 = df[50:100]
3 df2 = df[100:]
```

```
1 import matplotlib.pyplot as plt
2 %matplotlib inline
```

Sepal length vs Sepal Width (Setosa vs Versicolor)

```
1 plt.xlabel('Sepal Length')
2 plt.ylabel('Sepal Width')
3 plt.scatter(df0['sepal length (cm)'], df0['sepal width (cm)'],color="green",marker='+')
4 plt.scatter(df1['sepal length (cm)'], df1['sepal width (cm)'],color="blue",marker='.')
```

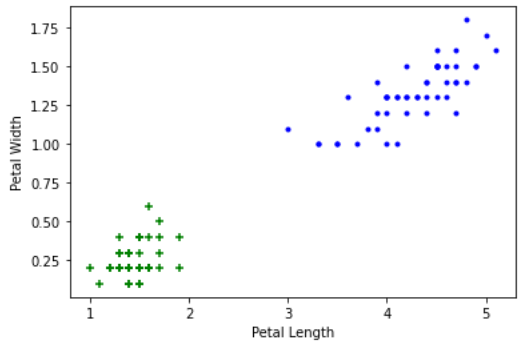
```
<matplotlib.collections.PathCollection at 0x7fa69adf7700>
```



Petal length vs Pepal Width (Setosa vs Versicolor)

```
1 plt.xlabel('Petal Length')
2 plt.ylabel('Petal Width')
3 plt.scatter(df0['petal length (cm)'], df0['petal width (cm)'],color="green",marker='+')
4 plt.scatter(df1['petal length (cm)'], df1['petal width (cm)'],color="blue",marker='.')
```

```
<matplotlib.collections.PathCollection at 0x7fa69ad65850>
```



Train test split

```
1 from sklearn.model_selection import train_test_split
```

```
1 X = df.drop(['target','flower_name'], axis='columns')
2 y = df.target
3 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

```
1 len(X_train)
```

```
120
```

```
1 len(X_test)
```

```
30
```

Create KNN (K Neighrest Neighbour Classifier)

```
1 from sklearn.neighbors import KNeighborsClassifier
2 knn = KNeighborsClassifier(n_neighbors=10)
```

```
1 knn.fit(X_train, y_train)
```

```
KNeighborsClassifier(n_neighbors=10)
```

```
1 knn.score(X_test, y_test)
```

```
0.9666666666666667
```

```
1 knn.predict([[4.8,3.0,1.5,0.3]])
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/base.py:450: UserWarning: X does not have valid feature names, but KNeighborsClassifier w
  warnings.warn(
array([0])
```

Plot Confusion Matrix

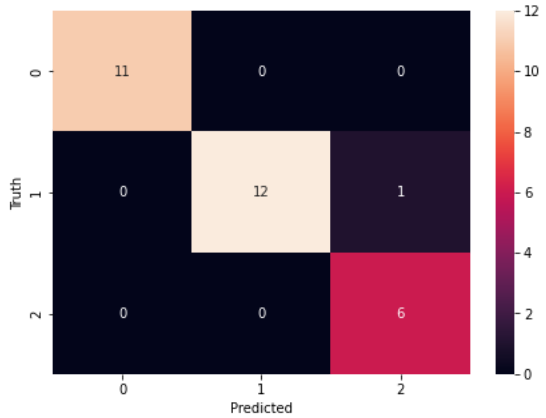
```
1 from sklearn.metrics import confusion_matrix
2 y_pred = knn.predict(X_test)
3 cm = confusion_matrix(y_test, y_pred)
4 cm
```

```
array([[11,  0,  0],
       [ 0, 12,  1],
       [ 0,  0,  6]])
```

```
1 %matplotlib inline
2 import matplotlib.pyplot as plt
3 import seaborn as sn
4 plt.figure(figsize=(7,5))
5 sn.heatmap(cm, annot=True)
```

```
6 plt.xlabel('Predicted')
7 plt.ylabel('Truth')
```

```
Text(42.0, 0.5, 'Truth')
```




```
1 from sklearn.metrics import classification_report
2
3 print(classification_report(y_test, y_pred))
```

```
                  precision    recall  f1-score   support

               0       1.00      1.00      1.00        11
               1       1.00      0.92      0.96        13
               2       0.86      1.00      0.92         6

        accuracy                           0.97        30
       macro avg       0.95      0.97      0.96        30
    weighted avg       0.97      0.97      0.97        30
```