❖ Day 5: Introduction to KNN

❖ KNN:

The K-Nearest Neighbors (KNN) algorithm is a robust and intuitive machine learning method employed to tackle classification and regression problems.

- K-Nearest Neighbors is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining, and intrusion detection.

❖ About KNN:

- Supervised learning model
- Used for both classification & regression.
- Can be used for non-linear data.
-

❖ Distance Metrics Used in KNN Algorithm:

that the KNN algorithm helps us identify the nearest points or the groups for a query point. But to determine the closest groups or the nearest points for a query point we need some metric. For this purpose, we use below distance metrics:

➢ Euclidean Distance:

It is the most common distance metric and represents the straight-line distance between two points in Euclidean space.

➢ Manhattan Distance:

Manhattan Distance metric is generally used when we are interested in the total distance traveled by the object instead of the displacement.

➢ Minkowski Distance:

it is a generalization of both Euclidean and Manhattan distances. The parameter p allows tuning between the two. When P=2, it is equivalent to Euclidean distance, and when P=1, it is equivalent to Manhattan distance.

❖ **KNN working:**

- Storing training data.
- When given a new data point, finding the k-nearest neighbors base on distance.
- For classification, assigning the majority class of these neighbors; for regression, predicting the average target value.
- Key parameters: k(number of neighbors) and distance metric.

❖ **Application:**

1.Data processing:

- When dealing with machine learning problems, exploratory data analysis (EDA) is essential.
- If the data contains missing values, knn imputer Is an effective method for imputation.

2.Pattern Recognition:

- KNN algorithms demonstrate high accuracy, especially in tasks like digit recognition using the MNIST dataset.
- The algorithm is well-suited for pattern recognition tasks.

3.Recommendation Engines:

- KNN is used to assign a new query point to a pre-existing group based on a large dataset.
- This functionality is crucial in recommender systems where users are assigned to groups, and recommendations are provided based on group preferences.

❖ **Advantage:**

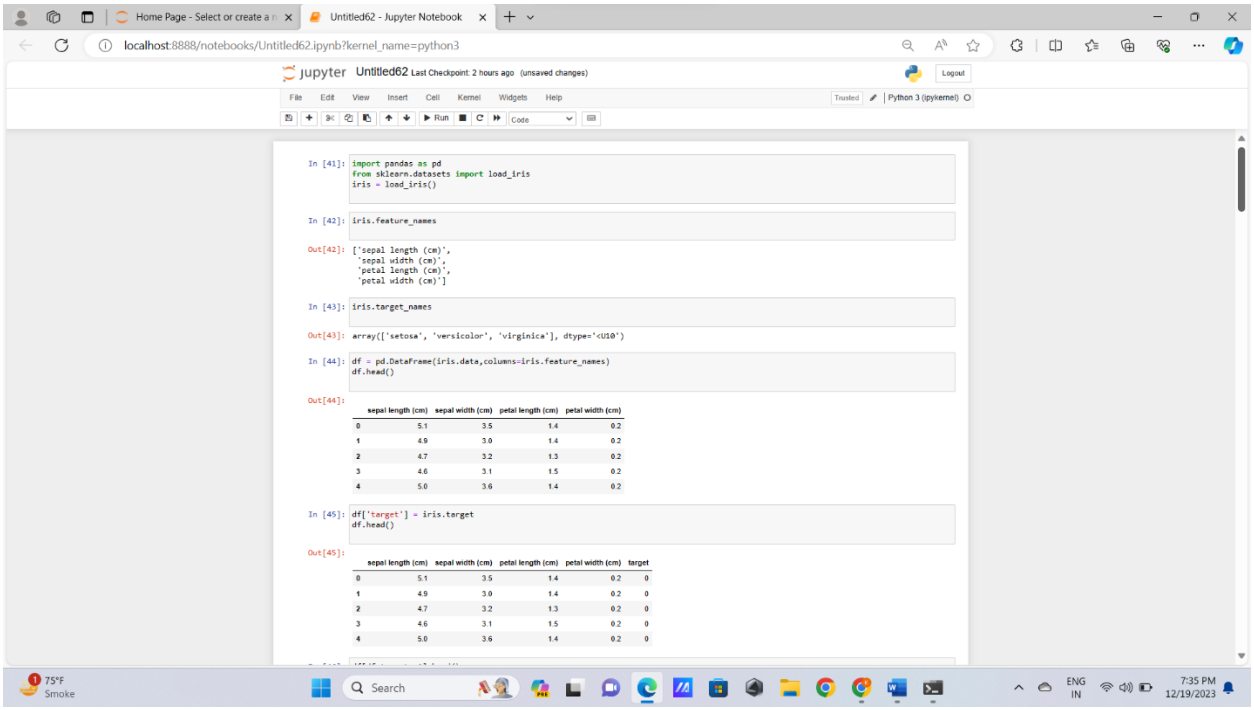
1. **Easy to implement** as the algorithm is simple and easy to implement.

2. There's no need to build a model, tune several parameters, or make additional assumptions.

3. The algorithm is versatile. It can be used for classification, regression, and search (as we will see in the next section).

❖ **Disadvantage:**

1. The algorithm gets significantly slower as the number of examples and/or predictors/independent variables increase.



Home Page - Select or create a n | Untitled62 - Jupyter Notebook

localhost:8888/notebooks/Untitled62.ipynb?kernel_name=python3

jupyter **Untitled62** Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

Run — Code

In [46]:
```
df[df.target==1].head()
```

Out[46]:

	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

In [47]:
```
df[df.target==2].head()
```

Out[47]:

	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

In [48]:
```
df['flower_name'] =df.target.apply(lambda x: iris.target_names[x])
df.head()
```

Out[48]:

	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

In [49]:
```
df[45:55]
```

Out[49]:

	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

75°F Smoke — Search — ENG IN — 7:35 PM 12/19/2023

Home Page - Select or create a n | Untitled62 - Jupyter Notebook

localhost:8888/notebooks/Untitled62.ipynb?kernel_name=python3

jupyter **Untitled62** Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

Run — Code

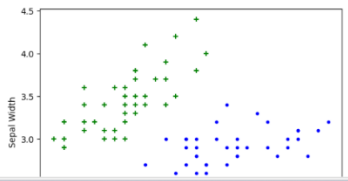
46	5.1	3.8	1.6	0.2	0	setosa
47	4.6	3.2	1.4	0.2	0	setosa
48	5.3	3.7	1.5	0.2	0	setosa
49	5.0	3.3	1.4	0.2	0	setosa
50	7.0	3.2	4.7	1.4	1	versicolor
51	6.4	3.2	4.5	1.5	1	versicolor
52	6.9	3.1	4.9	1.5	1	versicolor
53	5.5	2.3	4.0	1.3	1	versicolor
54	6.5	2.8	4.6	1.5	1	versicolor

In [50]:
```
df0 = df[:50]
df1 = df[50:100]
df2 = df[100:]
```

In [51]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

In [52]:
```
#Sepal Length vs Sepal Width (Setosa vs Versicolor)

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

Out[52]:
```
<matplotlib.collections.PathCollection at 0x255bb579d00>
```



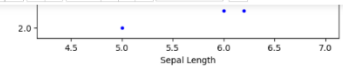
75°F Smoke — Search — ENG IN — 7:35 PM 12/19/2023

Home Page - Select or create a n | Untitled62 - Jupyter Notebook

localhost:8888/notebooks/Untitled62.ipynb?kernel_name=python3

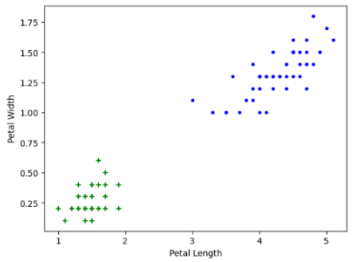
jupyter **Untitled62** Last Checkpoint: 2 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)



In [53]:
```
#Petal Length vs Pepal Width (Setosa vs Versicolor)

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

Out[53]:
```
<matplotlib.collections.PathCollection at 0x255bb66f460>
```



In [54]:
```
from sklearn.model_selection import train_test_split
```

In [55]:
```
X = df.drop(['target','flower_name'], axis='columns')
y = df.target
```

75°F Smoke — Search — ENG IN — 7:35 PM 12/19/2023

Home Page - Select or create a n | Untitled62 - Jupyter Notebook

localhost:8888/notebooks/Untitled62.ipynb?kernel_name=python3

jupyter **Untitled62** Last Checkpoint: 2 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Trusted — Python 3 (ipykernel)

In [56]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

In [57]:
```
len(X_train)
```

Out[57]:
```
120
```

In [58]:
```
len(X_test)
```

Out[58]:
```
30
```

In [59]:
```
#Create KNN (K Neighrest Neighbour Classifier)

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

In [60]:
```
knn.fit(X_train, y_train)
```

Out[60]:
```
KNeighborsClassifier(n_neighbors=10)
```

In [61]:
```
knn.score(X_test, y_test)
```

Out[61]:
```
0.9666666666666667
```

In [62]:
```
knn.predict([[4.8,3.0,1.5,0.3]])
```

```
C:\Users\Krunal\anaconda3\lib\site-packages\sklearn\base.py:465: UserWarning: X does not have valid feature names, but KNeighbo
rsClassifier was fitted with feature names
  warnings.warn(
```

Out[62]:
```
array([0])
```

In [63]:
```
from sklearn.metrics import confusion_matrix
y_pred = knn.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[63]:
```
array([[11,  0,  0],
       [ 0, 12,  1],
```

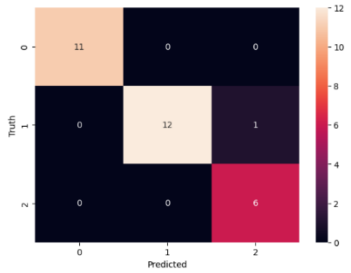
75°F Smoke — Search — ENG IN — 7:36 PM 12/19/2023

Out[63]:
```
array([[11,  0,  0],
       [ 0, 12,  1],
       [ 0,  0,  6]], dtype=int64)
```

In [64]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sn
```

In [65]:
```
plt.figure(figsize=(7,5))
sn.heatmap(cm, annot=True)
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

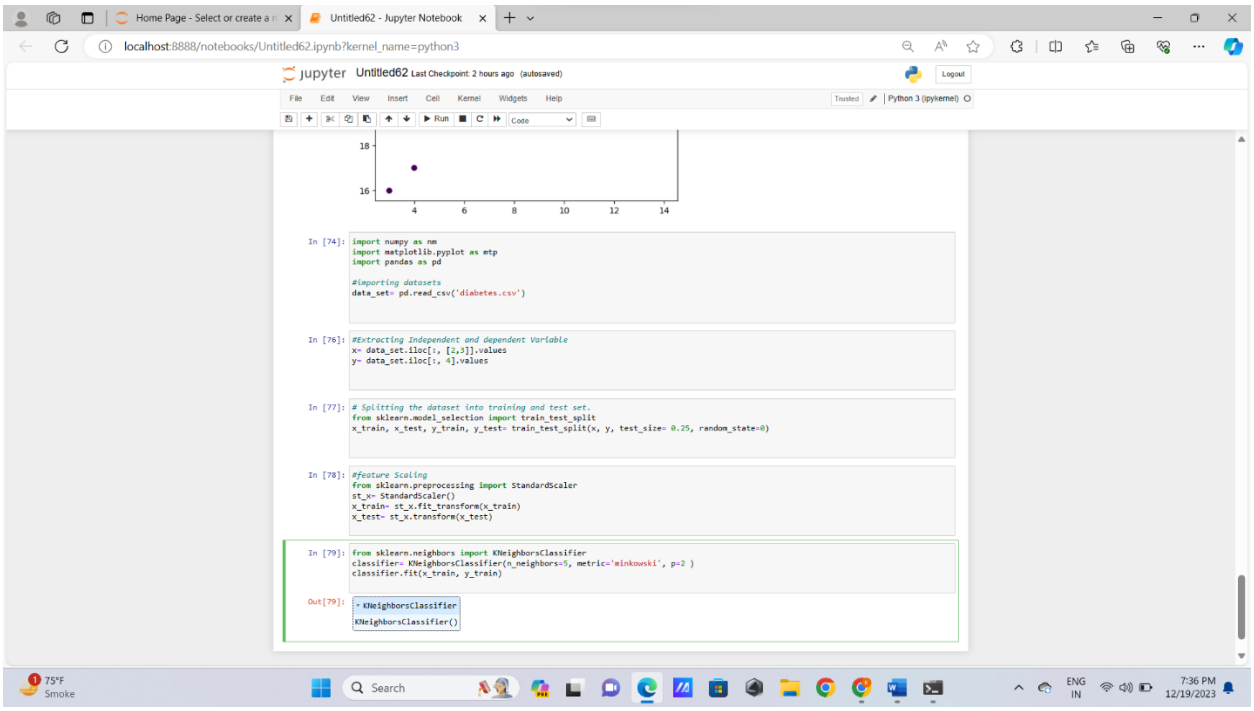
Out[65]:
```
Text(58.222222222222214, 0.5, 'Truth')
```



In [66]:
```
from sklearn.metrics import classification_report

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
```

In [71]:
```
import matplotlib.pyplot as plt

x = [4, 5, 10, 4, 3, 11, 14 , 8, 10, 12]
y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]

plt.scatter(x, y, c=classes)
plt.show()
```



❖**Description**:

I'm thrilled to apply this newfound knowledge to real-world scenarios, and the prospect of further honing these skills through the Skill Boost Internship Program adds an extra layer of excitement. Here's to the journey ahead and the exciting challenges awaiting in the Skill Boost Internship Program(www.Batweb.com).