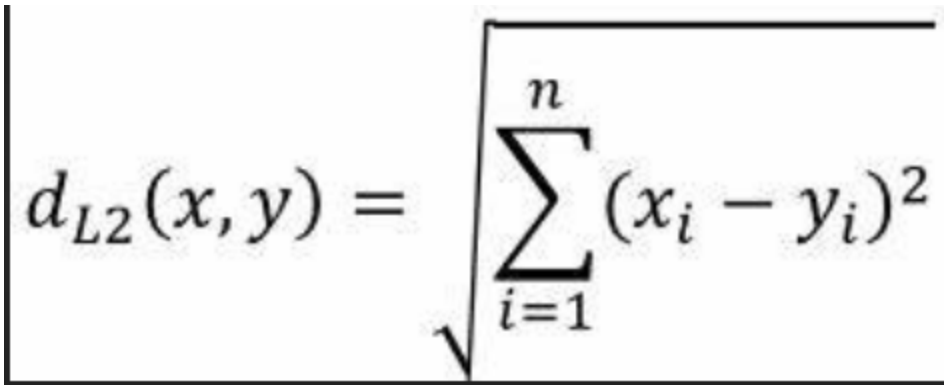
machine learning - K - nearest neighbour (KNN)

knn is the most lazy learning method . knn uses the “Euclidean diastance” method to get the nearest neighbours and then makes prediction according to it.

the first step for calculating the KNN is...

~calculate the euclidean distance

1.calculating the euclidean distance:



what is a euclidean distance?

Euclidean distance is a measure of the true straight line distance between two points in a plane. It is derived from Pythagoras' theorem and is used commonly in a variety of domains to measure the distance between data points.

the above given image has the formula for euclidean distance .

import pandas as pd
#here we are using the penguins dataset

```
from seaborn import load_dataset

df = load_dataset('penguins')
print(df.head())
```

Splitting our DataFrame into features and target

```
df = df.dropna()

X = df[['bill_length_mm']]
y = df['species']

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 100)

from sklearn.neighbors import KNeighborsClassifier

clf = KNeighborsClassifier(p=1) # where p is the power of the minkowski
clf.fit(X_train, y_train)

predictions = clf.predict(X_test)
print(predictions)

predictions = clf.predict([[44.2]])
print(predictions)

from sklearn.metrics import accuracy_score
print(accuracy_score(y_test, predictions))

import pandas as pd
import matplotlib.pyplot as plt
```

Assuming your data is in a pandas dataframe called 'data'

```
data_projected = data[["bill_length_mm", "species"]]  # Select relevant columns
```

Plot data points with different colors based on species

plt.scatter(data_projected["bill_length_mm"], data_projected["species"], c='red')

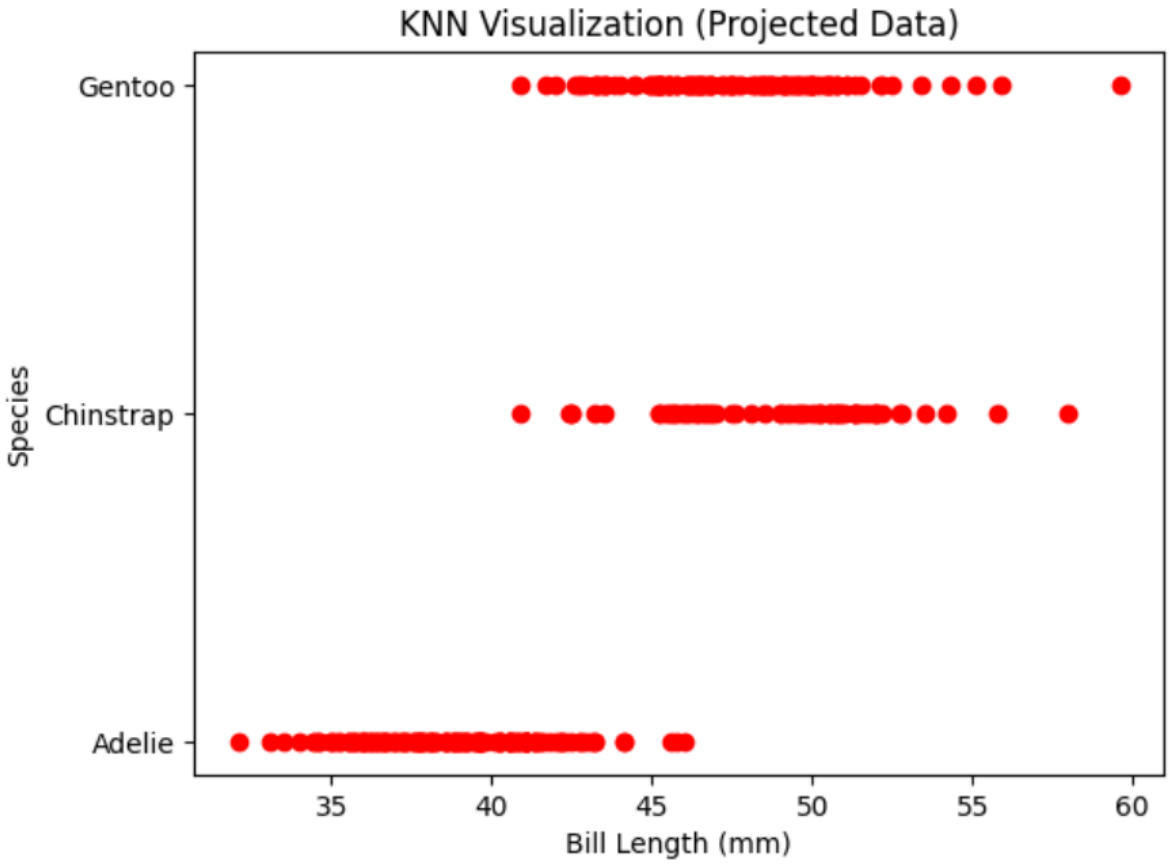
Implement logic to identify and highlight k nearest neighbors for some data points

plt.xlabel("Bill Length (mm)")
plt.ylabel("Species")
plt.title("KNN Visualization (Projected Data)")
plt.show()

the output will be :

```
  species     island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
0  Adelie  Torgersen            39.1           18.7              181.0
1  Adelie  Torgersen            39.5           17.4              186.0
2  Adelie  Torgersen            40.3           18.0              195.0
3  Adelie  Torgersen             NaN            NaN                NaN
4  Adelie  Torgersen            36.7           19.3              193.0

   body_mass_g     sex
0       3750.0    Male
1       3800.0  Female
2       3250.0  Female
3          NaN     NaN
4       3450.0  Female
['Chinstrap' 'Gentoo' 'Chinstrap' 'Adelie' 'Gentoo' 'Gentoo' 'Gentoo'
 'Chinstrap' 'Gentoo' 'Gentoo' 'Gentoo' 'Adelie' 'Adelie' 'Gentoo'
 'Gentoo' 'Chinstrap' 'Chinstrap' 'Adelie' 'Gentoo' 'Gentoo' 'Adelie'
 'Gentoo' 'Adelie' 'Adelie' 'Adelie' 'Adelie' 'Chinstrap' 'Chinstrap'
 'Adelie' 'Adelie' 'Adelie' 'Adelie' 'Gentoo' 'Adelie' 'Chinstrap'
 'Gentoo' 'Adelie' 'Gentoo' 'Chinstrap' 'Gentoo' 'Adelie' 'Gentoo'
 'Adelie' 'Adelie' 'Chinstrap' 'Chinstrap' 'Chinstrap' 'Adelie'
 'Chinstrap' 'Gentoo' 'Gentoo' 'Gentoo' 'Adelie' 'Adelie' 'Gentoo'
 'Gentoo' 'Adelie' 'Gentoo' 'Gentoo' 'Adelie' 'Gentoo' 'Gentoo' 'Gentoo'
 'Adelie' 'Adelie' 'Adelie' 'Chinstrap' 'Adelie' 'Gentoo' 'Gentoo'
 'Chinstrap' 'Chinstrap' 'Adelie' 'Chinstrap' 'Gentoo' 'Gentoo' 'Gentoo'
 'Chinstrap' 'Adelie' 'Chinstrap' 'Adelie' 'Adelie' 'Adelie' 'Chinstrap']
['Gentoo']
```



hence , this is how the KNN classification is done .