```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from collections import Counter
```

```
def euclidean_distance(x1, x2):
  return np.sqrt(np.sum((x1-x2)**2))
```

Class, Main Of KNN

```
class KNN:
    def __init__(self, k=3):
        self.k = k

    def fit(self, X, y):
        self.X_train = X
        self.y_train = y

    def predict(self, X):
        y_pred = [self._predict(x) for x in X]
        return np.array(y_pred)

    def _predict(self, x):
        # Compute distances between x and all examples in the training set
        distances = [euclidean_distance(x, x_train) for x_train in self.X_train]
        # Sort by distance and return indices of the first k neighbors
        k_idx = np.argsort(distances)[: self.k]
        # Extract the labels of the k nearest neighbor training samples
        k_neighbor_labels = [self.y_train[i] for i in k_idx]
        # return the most common class label
        most_common = Counter(k_neighbor_labels).most_common(1)
        return most_common[0][0]
```

```
if __name__ == "__main__":

    from sklearn import datasets
    from sklearn.model_selection import train_test_split

    def accuracy(y_true, y_pred):
        accuracy = np.sum(y_true == y_pred) / len(y_true)
        return accuracy

    datasett = pd.read_csv('/content/Social_Network_Ads.csv')
    X = datasett.iloc[:, [2,3]].values
    y = datasett.iloc[:, -1].values

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 1)

    clf = KNN(k = 3)
    clf.fit(X_train, y_train)
    predictions = clf.predict(X_test)
    print("KNN classification accuracy", accuracy(y_test, predictions))
```

```
KNN classification accuracy 0.76
```

Start coding or generate with AI.

Examples On Counters

```
a = [1, 1, 1, 2, 3, 4, 5, 6, 6, 3, 1]
from collections import Counter
most_common = Counter(a).most_common
print(most_common)
```

```
<bound method Counter.most_common of Counter({1: 4, 3: 2, 6: 2, 2: 1, 4: 1, 5: 1})>
```

```python
a = [1, 1, 1, 2, 3, 4, 5, 6, 6, 3, 1]
from collections import Counter
most_common = Counter(a).most_common(1)
print(most_common)
```

```
[(1, 4)]
```

```python
a = [1, 1, 1, 2, 3, 4, 5, 6, 6, 3, 1]
from collections import Counter
most_common = Counter(a).most_common(1)
print(most_common[0][0])
```

```
1
```