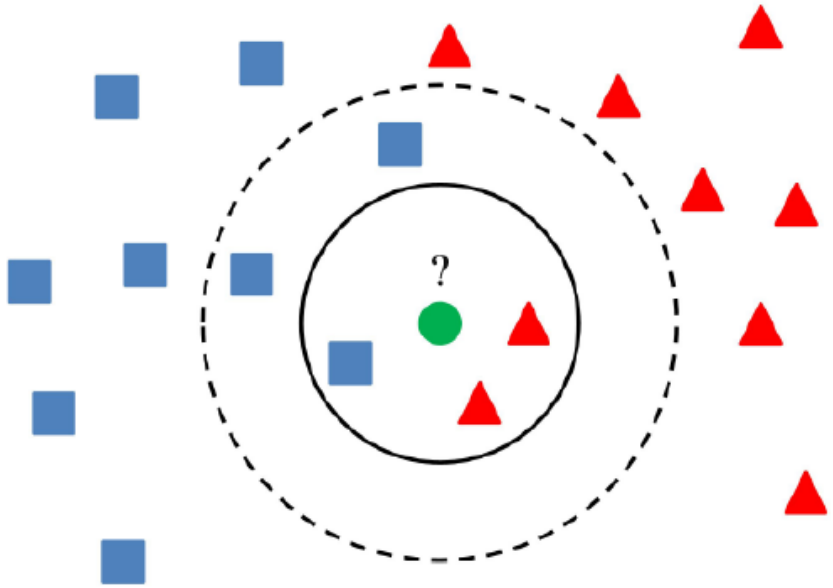
KLASIFIKASI

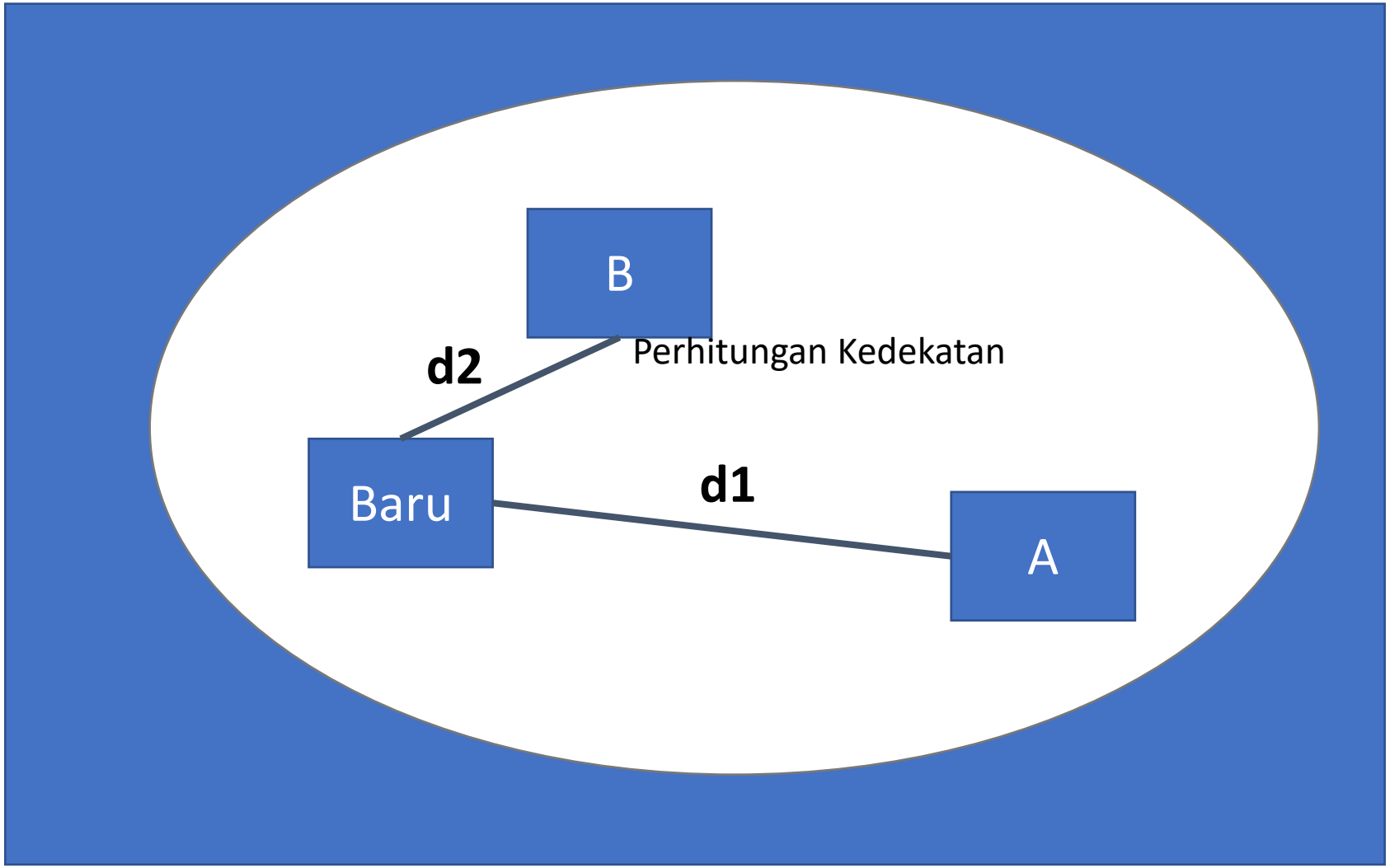
- Klasifikasi adalah algoritma yang menggunakan data dengan target/class/label berupa nilai kategorikal (nominal)
- Contoh, apabila target/class/label adalah pendapatan, maka bisa digunakan nilai nominal (kategorikal) sbb: pendapatan besar, menengah, kecil
- Contoh lain adalah rekomendasi contact lens, apakah menggunakan yang jenis soft, hard atau none
- Algoritma klasifikasi yang biasa digunakan adalah: Naive Bayes, K-Nearest Neighbor, C4.5, ID3, CART, Linear Discriminant Analysis, etc

NEAREST NEIGHBOR

- Nearest Neightbor merupakan pendekatan untuk mencari kasus lama, yaitu berdasarkan pada pencocokan bobot dari sejumlah fitur yang ada.
- Misal untuk mencari solusi terhadap pasien baru dengan menggunakan solusi dari pasien terdahulu.
- Dengan menghitung kedekatan kasus pasien baru dengan semua kasus pasien lama.
- Kasus pasien lama dengan kedekatan terbesarlah yang akan diambil solusinya untuk digunakan pada kasus pasien baru.



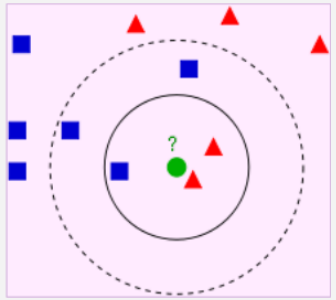
ILUSTRASI KEDEKATAN KASUS



K-NN

- Diberikan titik query, akan ditemukan sejumlah k obyek atau (titik training) yang paling dekat dengan titik query.
- Klasifikasi menggunakan voting terbanyak diantara klasifikasi dari k obyek
- Algoritma k-nearest neighbor (KNN) menggunakan klasifikasi ketetanggaan sebagai nilai prediksi dari query instance yang baru

- Jika k=3 maka lingkaran hijau akan masuk dalam kategori segitiga merah
- Jika k=5 maka lingkaran hijau akan masuk dalam kategori segiempat biru.



UKURAN JARAK

Dekat atau jauhnya tetangga biasanya dihitung berdasarkan *Euclidean Distance.*

$$D(a,b) = \sqrt{\sum_{k=1}^{d}(a_k - b_k)^2},$$

Dimana D(a,b) adalah jarak skalar dari dua buah vektor data a dan b yang berupa matrik berukuran d dimensi.

Beberapa macam jarak yang dapat digunakan

- Jarak Euclidean

$$d(x,y) = \|x-y\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

- Jarak Manhattan atau Cityblock

$$d(x,y) = \sum_{i=1}^{n}(|x_i - y_i|)$$

- Jarak Minkowski

$$d(x,y) = \|x-y\|_q = (\sum|x-y|^q)^{\frac{1}{q}}$$

- Jarak Mahalanobis

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1}(\vec{x}-\vec{y})}$$

KELEBIHAN DAN KEKURANGAN K-NN

Kelebihan KNN:

- Simpel
- Efektif jika data besar
- Intuitif
- Peforma cukup baik
- Tahan terhadap data training yang noisy

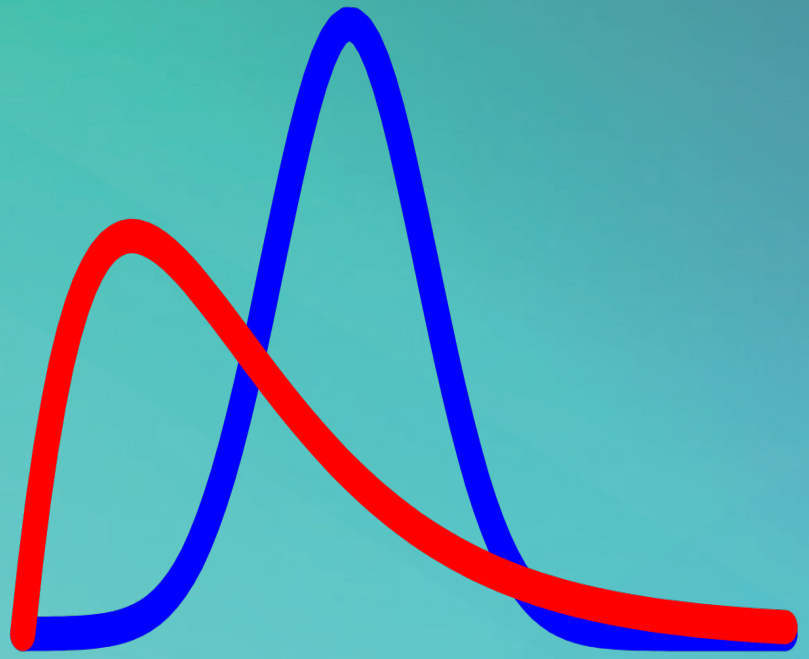
Kekurangan KNN :

- Waktu komputasi tinggi jika data latih besar. Disebabkan oleh semua data diukur jaraknya untuk setiap data uji.
- Sangat sensitif dengan ciri/data yang redundan atau tidak relevan.
- Tidak diketahui perhitungan jarak apa yang paling sesuai untuk dataset tertentu.

Algoritma K-NN

1. Menentukan parameter k (jumlah tetangga paling dekat).
2. Menghitung kuadrat jarak eucliden objek terhadap data training yang diberikan.
3. Mengurutkan hasil no 2 secara ascending
4. Mengumpulkan kategori Y (Klasifikasi nearest neighbor berdasarkan nilai k)
5. Dengan menggunakan kategori nearest neighbor yang paling mayoritas maka dapat dipredisikan kategori objek .

	KNN Algorithm
	Input: Set of training x_i and testing dataset y_k, k value
	Output: the label of testing dataset
1	for each testing dataset y_k
2	for each training dataset x_i
3	compute the distance $d(x_i, y_k)$
4	end for
5	sort the distance in the ascending order
6	select the top$-k$ training data
7	label the testing data based on the majority class of the top$-k$ training data
8	end for



Contoh Perhitungan

Klasifikasi menggunakan K-NN

KASUS 1 (K=4)

Terdapat beberapa data yang berasal dari survey questioner tentang klasifikasi kualitas kertas tissue apakah baik atau jelek, dengan objek training menggunakan dua attribute yaitu daya tahan terhadap asam dan kekuatan. Dengan menggunakan K = 4.

X1 = Daya tahan asam (detik)	X2 = Kekuatan (Kg/m^2)	Y = Klasifikasi
8	4	Baik
4	5	Jelek
4	6	Jelek
7	7	Baik
5	6	Jelek
6	5	Baik

Akan diproduksi kembali kertas tisu dengan attribute **X1=7** dan **X2=4**, maka dapat diklasifikasikan kertas tise tersebut termasuk yang ***baik*** atau ***jelek*** *?*

MENGHITUNG NILAI KEDEKATAN

X1 = Daya tahan asam (detik)	X2 = Kekuatan (kg/m^2)	Square distance to query distance (7,4)
8	4	$(8-7)^2 + (4-4)^2 = 1$
4	5	$(4-7)^2 + (5-4)^2 = 10$
4	6	$(4-7)^2 + (6-4)^2 = 13$
7	7	$(7-7)^2 + (7-4)^2 = 9$
5	6	$(5-7)^2 + (6-4)^2 = 8$
6	5	$(6-7)^2 + (5-4)^2 = 2$

PENYELESAIAN KASUS

X1= Daya tahan asam (detik)	X2= Kekuatan (Kg/m^2)	Square distance to query distance (7,4)	Jarak terkecil	Apakah termasuk nearest neighbor (K)	Y= kategori nearest neighbor
8	4	$(8-7)^2 + (4-4)^2 = 1$	1	Ya	Baik
4	5	$(4-7)^2 + (5-4)^2 = 10$	5	Tidak	-
4	6	$(4-7)^2 + (6-4)^2 = 13$	6	Tidak	-
7	7	$(7-7)^2 + (7-4)^2 = 9$	4	Ya	Baik
5	6	$(5-7)^2 + (6-4)^2 = 8$	3	Ya	Jelek
6	5	$(6-7)^2 + (5-4)^2 = 2$	2	Ya	Baik

- Ada 4 data yang paling dekat yaitu (8,4) , (6,5) , (5,6), dan (7,7). Kemudian ***hitung jumal kelas*** untuk ke empat data tersebut, sehingga diperoleh ***baik*** = 3 dan ***jelek*** = 1.
- Dengan voting maka diperoleh bahwa tissue dengan daya tahan 7 dan Kekuatan 4 hasilnya termasuk Kategori ***Baik***.

KASUS 2

Tentukan ***class*** dari ***test*** data dengan nilai atribut (**50,3,40**)

X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Y = Klasifikasi
40	5	60	Jelek
50	8	40	Bagus
50	7	30	Jelek
70	4	60	Bagus
80	4	80	Bagus
60	6	60	Bagus

MENGHITUNG NILAI KEDEKATAN

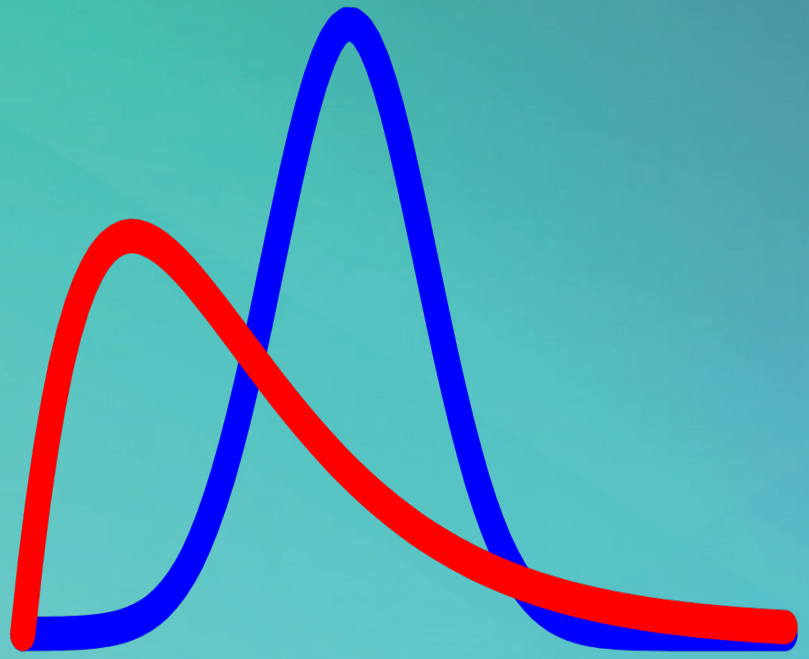
X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Square distance to query distance (50,3,40)
40	5	60	$(40-50)^2 + (5-3)^2 + (60-40)^2 = 504$
50	8	40	$(50-50)^2 + (8-3)^2 + (40-40)^2 = 25$
50	7	30	$(50-50)^2 + (7-3)^2 + (30-40)^2 = 116$
70	4	60	$(70-50)^2 + (4-3)^2 + (60-40)^2 = 801$
80	4	80	$(80-50)^2 + (4-3)^2 + (80-40)^2 = 2501$
60	6	60	$(60-50)^2 + (6-3)^2 + (60-40)^2 = 509$

PENYELESAIAN

X1 = Takaran saji (gr)	X2 = Jumlah Saji perkemasan	X3 = Energi Total	Square distance to query distance (50,3,40)	Jarak Terkecil	Apakah termasuk nearest neighbor (K)	Y= kategori nearest neighbor
40	5	60	$(40-50)^2 + (5-3)^2 + (60-40)^2 = 504$	3	Ya	Jelek
50	8	40	$(50-50)^2 + (8-3)^2 + (40-40)^2 = 25$	1	Ya	Bagus
50	7	30	$(50-50)^2 + (7-3)^2 + (30-40)^2 = 116$	2	Ya	Jelek
70	4	60	$(70-50)^2 + (4-3)^2 + (60-40)^2 = 801$	5	Tidak	-
80	4	80	$(80-50)^2 + (4-3)^2 + (80-40)^2 = 2501$	6	Tidak	-
60	6	60	$(60-50)^2 + (6-3)^2 + (60-40)^2 = 509$	4	Ya	Bagus

Jika digunakan ***K = 4*** maka voting akan **seimbang** Bagus = 2 dan Jelek = 2. Untuk menanggulangi hal tersebut maka nilai ***K dikurangi 1*** untuk setiap ditemukan hasil voting yang seimbang.

Dengan ***K = 3*** maka voting kelas Bagus = 1 dan Jelek = 2. Maka ***class*** dari test data dengan nilai atribut (50,3,40) termasuk Kelas yang **Jelek**.

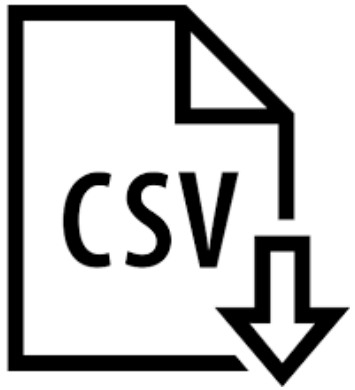


Implementasi Klasifikasi K-NN dengan python

Dataset

dataset - DataFrame

Index	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
5	15728773	Male	27	58000	0
6	15598044	Female	27	84000	0
7	15694829	Female	32	150000	1
8	15600575	Male	25	33000	0
9	15727311	Female	35	65000	0
10	15570769	Female	26	80000	0
11	15606274	Female	26	52000	0
12	15746139	Male	20	86000	0



Klik icon untuk download

Import library yang digunakan

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

Library yang digunakan untuk contoh diatas adalah library **numpy** dan **pandas**.

Import Dataset

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

Splitting the dataset into the Training set and Test set

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Training the K-NN model on the Training set

```
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)
```

Predicting the Test set results

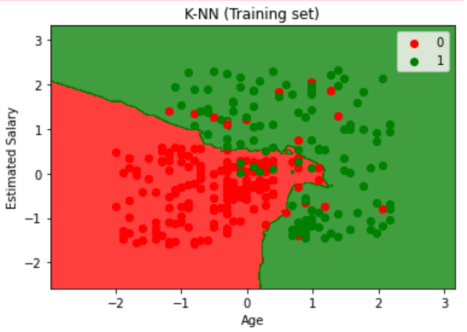
```
y_pred = classifier.predict(X_test)
```

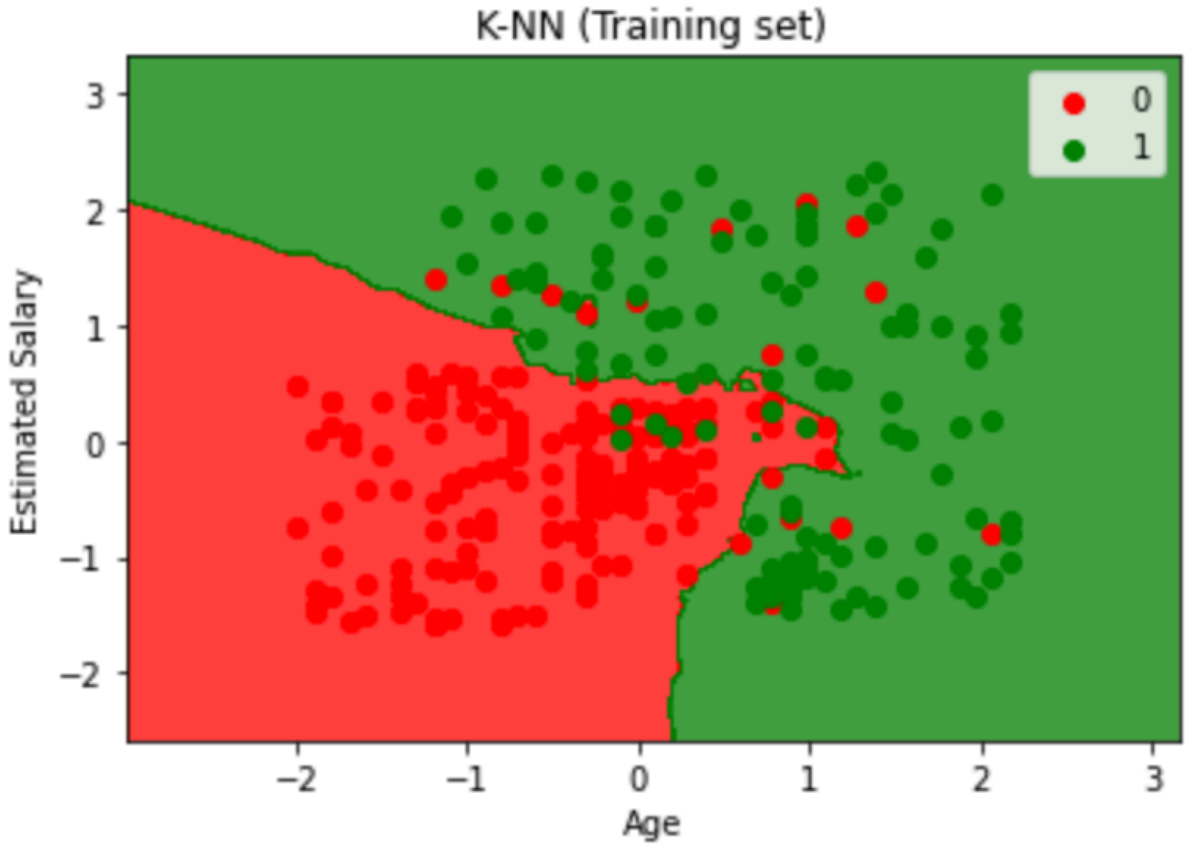
Making the Confusion Matrix

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

Visualising the Training set results

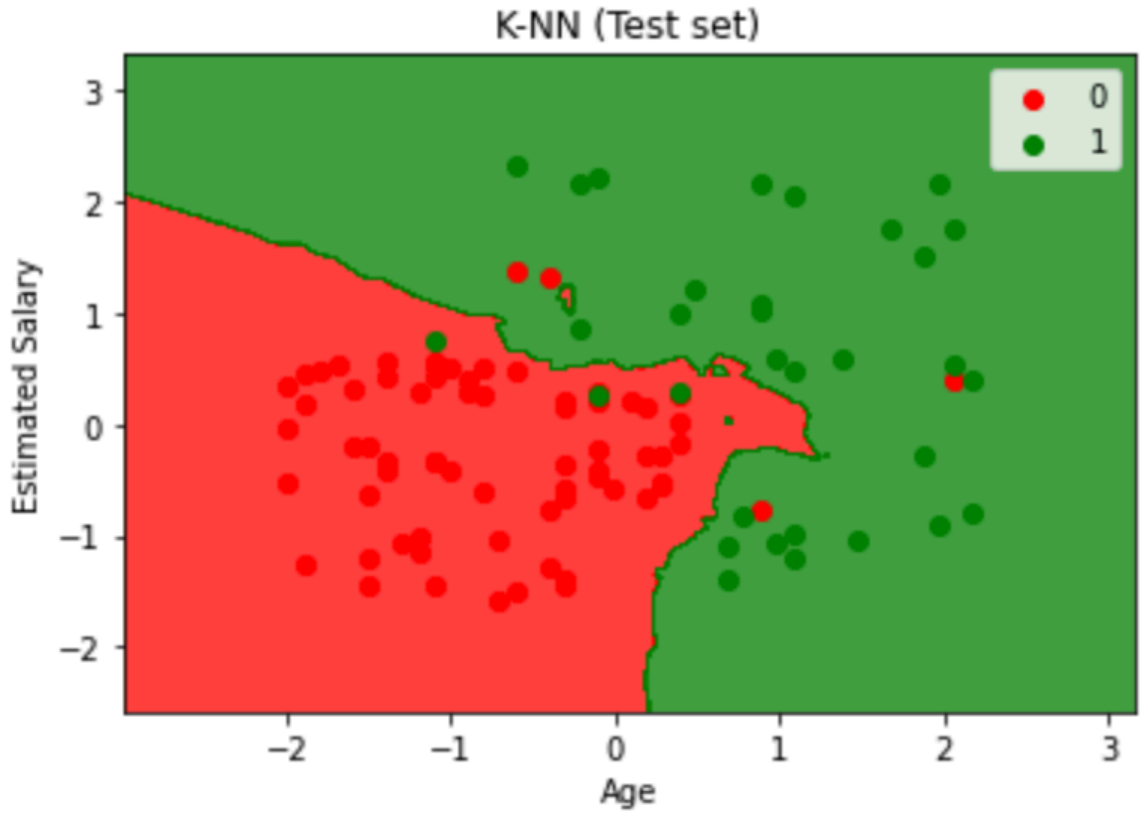
```
from matplotlib.colors import ListedColormap
X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('K-NN (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```




K-NN (Training set)
Estimated Salary
Age
0
1
3
2
1
0
−1
−2
−2
−1
0
1
2
3

Visualising the Test set results

```
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('K-NN (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```


K-NN (Test set)
Estimated Salary
Age
0
1
3
2
1
0
−1
−2
−2
−1
0
1
2
3