Machine Learning: K-nearest Neighbors

STUDI KASUS

Kali ini kita akan memecahkan permasalahan yang sama persis yang semoat dibahas di metode *logistic regression*, yaitu permasalahan pemilik showroom mobil.

Saya ulang sedikit bahwa pemiliki showroom ingin mecari tahu solusi iklan di internet, di mana ia memiliki data pelanggannya. Melalui data usia dan estimasi penghasilan, ia ingin fokus untuk mengiklankan produknya di kelompok yang memiliki kemungkinan membeli mobilnya lebih tinggi.

Langsung saja, kita bahas dengan menggunakan dua bahasa, yaitu python dan R.

Pertama download dulu datasetnya di sini.

Bahasa Python

```
# Mengimpor library
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Mengimpor dataset
dataset = pd.read_csv('Iklan_sosmed.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Membagi dataset menjadi 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)

# Membuat model k-NN
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p
= 2)
classifier.fit(X_train, y_train)

# Memprediksi Test set
y_pred = classifier.predict(X_test)

# Membuat Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

# Visualisasi hasil Training set
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('Usia')
plt.ylabel('Estimasi Gaji')
```

```
plt.legend()
plt.show()

# Visualisasi hasil Test set
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('Usia')
plt.ylabel('Estimasi Gaji')
plt.legend()
plt.show()
```

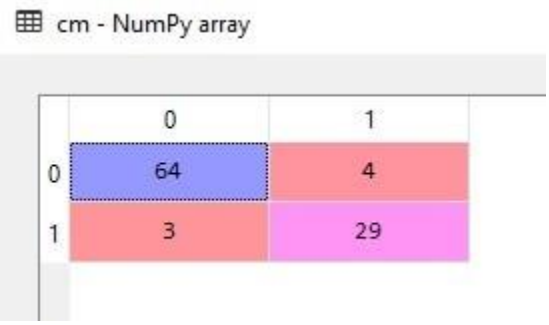
Penjelasan:

- Line 1-4 mengimpor library yang diperlukan.
- Line 7 mengimpor dataset.
- Line 8 mendefinisikan variabel X yaitu kolom usia dan estimasi gaji.
- Line 9 mendefinisikan variabel y yaitu keputusan beli/tidak.
- Line 12 mengimpor library dari sklearn untuk membagi dataset ke dalam training dan test set dengan konfigurasi 75:25 dan *random number generator* 0.
- Line 13 mendefinisikan variabel training dan test set.
- Line 16-19 melakukan feature scaling.
- Line 22 mengimpor library k-NN.
- Line 23 adalah mendefinisikan variabel classifier sebagai model k-NN. Untuk bisa melihat parameter apa saja cukup arahkan kursor pada classifier lalu ketik di keyboard CTRL+i, maka akan muncul parameternya di help spyder seperti ini:

Tampilan parameter k-NN di spyder.

Melalui gambaran parameter k-NN di atas, maka kita memilih jumlah K 5 sesuai default nya. Kemudian metric kit pilih ‘minkowski’, dan p kita isi 2, karena menggunakan euclidean distance. Semua keterangan ini bisa dilihat di keterangan parameter sesuai gambar di atas.

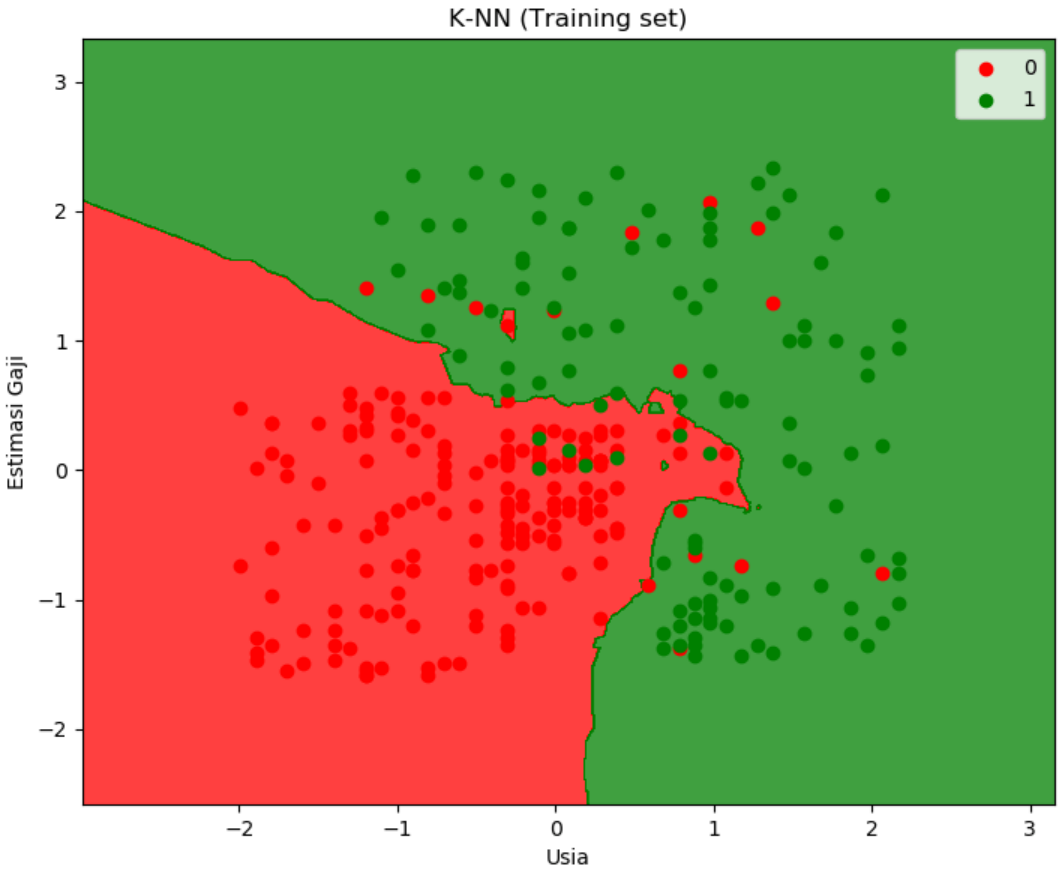
- Line 24 mengaplikasikan model ini terhadap X_train dan y_train, karena kita ingin melatih modelnya dari training set.
- Line 27 adalah mendefinisikan variabel y_pred untuk memprediksi test set. Kita memprediksinya dari X_test, yang nantinya y_pred ini akan dibandingkan dengan y_test. Jika y_pred hasilnya mendekati y_test, maka bisa disimpulkan performanya cukup baik.
- Line 30 mengimpor library confusion_matrix dari sklearn untuk membuat confusion_matrix (CM)-nya. CM adalah perbandingan antara benar dan salah antara y_pred dengan y_test.
- Line 31 membuat variabel cm sebagai confusion matrix nya. Jika kita klik cm di *variable explorer* maka akan tampak sebagai berikut:



Hasil confuion matrix dengan metode k-NN.

Jika dilihat CM di atas, maka kita mendapatkan prediksi benar sebanyak 64+29, dan prediksi salah 3+4, sehingga komposisinya 93:7. Bisa dikatakan akurasi kita 93%, sangat baik!

- Line 34-49 adalah perintah untuk visualisasi model k-NN terhadap training set. Hasilnya akan tampak sebagai berikut:

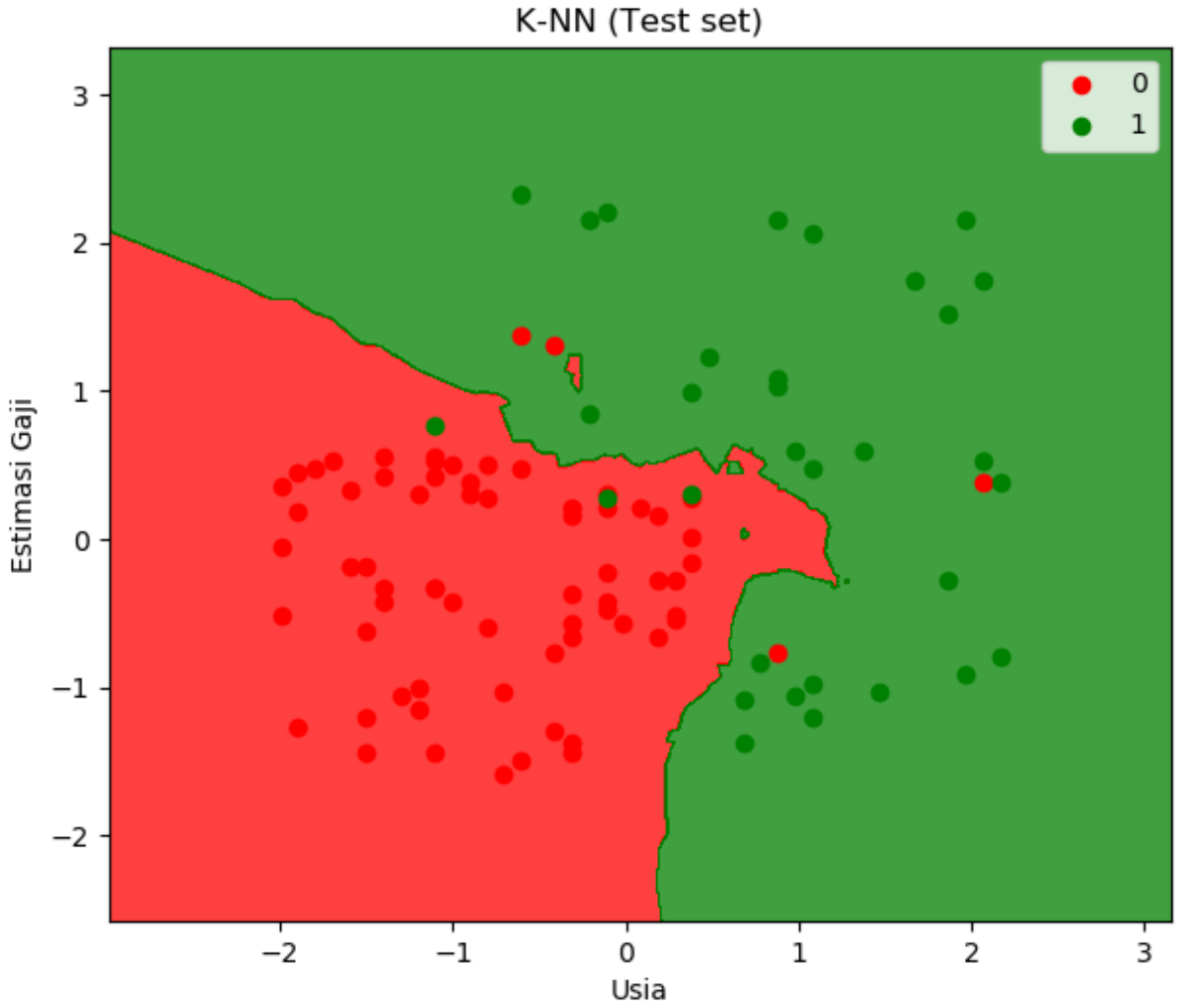


Tampilan model k-NN dari training set.
Gambar ini adalah hasil prediksi keputusan beli/tidak berdasarkan usia dan gaji yang dimiliki pelanggan.

Pada gambar di atas dapat kita lihat bahwa zona merah adalah kelompok di mana pelanggan memutuskan untuk tidak membeli, sementara zona hijau adalah kelompok pelanggan memutuskan untuk beli.

Titik-titik merah adalah data-data pelanggan yang memutuskan untuk tidak membeli, sedangkan titik-titik hijau adalah data-data pelanggan yang memutuskan untuk membeli.

- Line 52-67 adalah perintah untuk visualisasi model k-NN terhadap test set. Tampilannya akan tampak sebagai berikut:



Tampilan model k-NN dari test set.
Gambar ini adalah hasil prediksi keputusan beli/tidak berdasarkan usia dan gaji yang dimiliki pelanggan.

Gambar di atas menunjukkan bahwa model kita bisa memprediksi dengan baik.

Zona hijau dan merah adalah modelnya, sementara titik-titik hijau dan merah adalah data aslinya. Hasil model kita memprediksi data asli sangat baik dengan tingkat akurasi 93%. Terlihat bahwa titik-titik merah sebagian besar benar-benar berada di zona merah. Begitu pula dengan zona hijau dan titik-titik hijau.

MEMPREDIKSI DATA TAMBAHAN

Lalu bagaimana jika kita ingin memprediksi 1 data tambahan saja, misalnya orang dengan usia 40 tahun dan gaji 3 juta?

Maka cukup tambahkan *script* ini di python:

```
Usia=int(input('Masukkan usia '))
# masukkan angka 40 di keyboard

EstimasiGaji=int(input('Masukkan gaji '))
# masukkan angka 3000000 di keyboard

array=[[Usia,EstimasiGaji]]
sc1=StandardScaler()
array_pre=classifier.predict(sc1.fit_transform(array))
```

Jika dieksekusi, maka data baru ini akan masuk ke kelompok 0.