MACHINE LEARNING

Nama : Hanif Ridal Warits

NPM : 41155050210060

Kelas : Informatika A2 - 2021

Tugas Pertemuan 5

1.0. K-Nearest Neighbours (KNN). Lakukan praktik dari https://youtu.be/4zARMcgc7hA?si=x6RoHQXFF4NY76X8 , buat screenshot dengan nama kalian pada coding, kumpulkan dalam bentuk pdf, dari kegiatan ini:

1.1. Persiapan sample dataset

```
[ ] print('Hanif Ridal Warits - 41155050210060')

Hanif Ridal Warits - 41155050210060
```

Classification dengan KNN (K Nearest Neighbhours)

- KNN adalah model machine learning yangdapat digunakan untuk melakukan prediksi berdasarkan kedekatan karakteristik dengan sejumlah tetangga terdekat.
- Prediksi yang dilakukan dapat diterapkan baik pada classification maupun regression tasks.

Sample Dataset

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    import pandas as pd

    sensus = {
        'tinggi': [158, 170, 183, 191, 155, 163, 180, 158, 178],
        'berat': [64, 86, 84, 80, 49, 59, 67, 54, 67],
        'jk': [
            'pria', 'pria', 'pria', 'pria', 'wanita', 'wanita', 'wanita', 'wanita',
            'wanita'
        ]
    }

    sensus_df = pd.DataFrame(sensus)
    sensus_df
```

```
[ ] Hanif Ridal Warits - 41155050210060
```

	tinggi	berat	jk
0	158	64	pria
1	170	86	pria
2	183	84	pria
3	191	80	pria
4	155	49	wanita
5	163	59	wanita
6	180	67	wanita
7	158	54	wanita
8	178	67	wanita

Next steps: Generate code with sensus_df | View recommended plots | New interactive sheet

1.2. Visualisasi dataset

Visualisasi Data

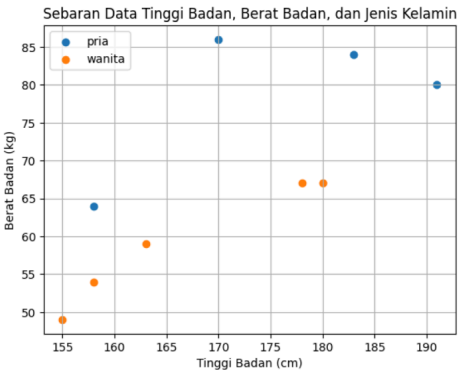
```
print('Hanif Ridal Warits - 41155050210060')

import matplotlib.pyplot as plt

fig, ax = plt.subplots()
for jk, d in sensus_df.groupby('jk'):
    ax.scatter(d['tinggi'], d['berat'], label=jk)

plt.legend(loc='upper left')
plt.title('Sebaran Data Tinggi Badan, Berat Badan, dan Jenis Kelamin')
plt.xlabel('Tinggi Badan (cm)')
plt.ylabel('Berat Badan (kg)')
plt.grid(True)
plt.show()
```

```
Hanif Ridal Warits - 41155050210060
```



1.3. Pengantar classification dengan K-Nearest Neighbours | KNN

Classification dengan KNN (K Nearest Neighbhours)

- **KNN adalah model machine learning yangdapat digunakan untuk melakukan prediksi berdasarkan kedekatan karakteristik dengan sejumlah tetangga terdekat.**
- **Prediksi yang dilakukan dapat diterapkan baik pada classification maupun regression tasks.**

1.4. Preprocessing dataset dengan Label Binarizer

Classification dengan KNN

Preprocessing Dataset

```
print('Hanif Ridal Warits - 41155050210060')
import numpy as np

X_train = np.array(sensus_df[['tinggi', 'berat']])
y_train = np.array(sensus_df['jk'])

print(f'X_train:\n{X_train}\n')
print(f'y_train: {y_train}')
```

```
Hanif Ridal Warits - 41155050210060
X_train:
[[158  64]
 [170  86]
 [183  84]
 [191  80]
 [155  49]
 [163  59]
 [180  67]
 [158  54]
 [178  67]]

y_train: ['pria' 'pria' 'pria' 'pria' 'wanita' 'wanita' 'wanita' 'wanita' 'wanita']
```

```
print('Hanif Ridal Warits - 41155050210060')
from sklearn.preprocessing import LabelBinarizer

lb = LabelBinarizer()
y_train = lb.fit_transform(y_train)
print(f'y_train:\n{y_train}')
```

```
Hanif Ridal Warits - 41155050210060
y_train:
[[0]
 [0]
 [0]
 [0]
 [1]
 [1]
 [1]
 [1]
 [1]]
```

```
print('Hanif Ridal Warits - 41155050210060')
y_train = y_train.flatten()
print(f'y_train: {y_train}')
```

```
Hanif Ridal Warits - 41155050210060
y_train: [0 0 0 0 1 1 1 1 1]
```

1.5. Training KNN Classification Model

Training KNN Classification Model

```
print('Hanif Ridal Warits - 41155050210060')
from sklearn.neighbors import KNeighborsClassifier

K = 3
model = KNeighborsClassifier(n_neighbors=K)
model.fit(X_train, y_train)
```

```
Hanif Ridal Warits - 41155050210060
KNeighborsClassifier
KNeighborsClassifier(n_neighbors=3)
```

1.6. Prediksi dengan KNN Classification Model

Prediksi Jenis Kelamin

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    tinggi_badan = 155
    berat_badan = 70
    X_new = np.array([tinggi_badan, berat_badan]).reshape(1, -1)
    X_new
```

```
Hanif Ridal Warits - 41155050210060
array([[155,  70]])
```

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    y_new = model.predict(X_new)
    y_new
```

```
Hanif Ridal Warits - 41155050210060
array([1])
```

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    lb.inverse_transform(y_new)
```

```
Hanif Ridal Warits - 41155050210060
array(['wanita'], dtype='<U6')
```

1.7. Visualisasi Nearest Neighbours

Visualisasi Nearest Neighbours

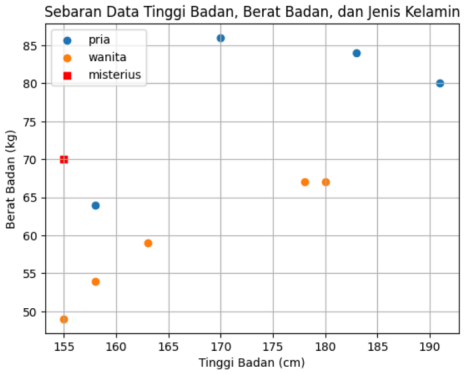
```
print('Hanif Ridal Warits - 41155050210060')

fig, ax = plt.subplots()
for jk, d in sensus_df.groupby('jk'):
    ax.scatter(d['tinggi'], d['berat'], label=jk)

plt.scatter(tinggi_badan,
            berat_badan,
            marker='s',
            color='red',
            label='misterius')

plt.legend(loc='upper left')
plt.title('Sebaran Data Tinggi Badan, Berat Badan, dan Jenis Kelamin')
plt.xlabel('Tinggi Badan (cm)')
plt.ylabel('Berat Badan (kg)')
plt.grid(True)
plt.show()
```

```
Hanif Ridal Warits - 41155050210060
```



1.8. Kalkulasi jarak dengan Euclidean Distance

Kalkulasi Distance (Euclidean Distance)

$$distance = \sqrt{(t_1 - t_2)^2 + (b_1 - b_2)^2}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    misterius = np.array([tinggi_badan, berat_badan])
    misterius
```

```
Hanif Ridal Warits - 41155050210060
array([155,  70])
```

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    X_train
```

```
Hanif Ridal Warits - 41155050210060
array([[158,  64],
       [170,  86],
       [183,  84],
       [191,  80],
       [155,  49],
       [163,  59],
       [180,  67],
       [158,  54],
       [178,  67]])
```

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from scipy.spatial.distance import euclidean

    data_jarak = [euclidean(misterius, d) for d in X_train]
    data_jarak
```

```
Hanif Ridal Warits - 41155050210060
[6.708203932499369,
 21.93171219946131,
 31.304951684997057,
 37.36308338453881,
 21.0,
 13.601470508735444,
 25.179356624028344,
 16.278820596099706,
 23.194827009486403]
```

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    sensus_df['jarak'] = data_jarak
    sensus_df.sort_values(['jarak'])
```

```
Hanif Ridal Warits - 41155050210060
```

	tinggi	berat	jk	jarak
0	158	64	pria	6.708204
5	163	59	wanita	13.601471
7	158	54	wanita	16.278821
4	155	49	wanita	21.000000
1	170	86	pria	21.931712
8	178	67	wanita	23.194827
6	180	67	wanita	25.179357
2	183	84	pria	31.304952
3	191	80	pria	37.363083

1.9. Evaluasi KNN Classification Model | Persiapan testing set

Evaluasi KNN Classification Model

Testing Set

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    X_test = np.array([[168, 65], [180, 96], [160, 52], [169, 67]])
    y_test = lb.transform(np.array(['pria', 'pria', 'wanita', 'wanita'])).flatten()

    print(f'X_test:\n{X_test}\n')
    print(f'y_test:\n{y_test}')
```

```
Hanif Ridal Warits - 41155050210060
X_test:
[[168  65]
 [180  96]
 [160  52]
 [169  67]]

y_test:
[0 0 1 1]
```

Prediksi terhadap testing set

```
[ ] print('Hanif Ridal Warits - 41155050210060')

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

```
Hanif Ridal Warits - 41155050210060
array([1, 0, 1, 1])
```

1.9. Evaluasi model dengan accuracy score

Accuracy

Accuracy is the proportion of test instances that were classified correctly.

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import accuracy_score

    acc = accuracy_score(y_test, y_pred)

    print(f'Accuracy: {acc}')
```

```
Hanif Ridal Warits - 41155050210060
Accuracy: 0.75
```

1.10. Evaluasi model dengan precision score

Precission

Precision is the proportion of test instances that were predicted to be positive that are truly positive.

$$precission = \frac{tp}{tp + fp}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import precision_score

    prec = precision_score(y_test, y_pred)

    print(f'Precission: {prec}')
```

```
Hanif Ridal Warits - 41155050210060
Precission: 0.6666666666666666
```

1.11. Evaluasi model dengan recall score

Recall

Recall is the proportion of truly positive test instances that were predicted to be positive.

$$recall = \frac{tp}{tp + fn}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import recall_score

    rec = recall_score(y_test, y_pred)

    print(f'Recall: {rec}')
```

```
Hanif Ridal Warits - 41155050210060
Recall: 1.0
```

1.12. Evaluasi model dengan F1 score

F1 Score

The F1 score is the harmonic mean of precision and recall.

$$F1 = 2 \times \frac{precission \times recall}{precission + recall}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import f1_score

    f1 = f1_score(y_test, y_pred)

    print(f'F1-score: {f1}')
```

```
Hanif Ridal Warits - 41155050210060
F1-score: 0.8
```

1.13. Evaluasi model dengan classification report

Classification Report

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import classification_report

    cls_report = classification_report(y_test, y_pred)

    print(f'Classification Report:\n{cls_report}')
```

```
Hanif Ridal Warits - 41155050210060
Classification Report:
              precision    recall  f1-score   support

           0       1.00      0.50      0.67         2
           1       0.67      1.00      0.80         2

    accuracy                           0.75         4
   macro avg       0.83      0.75      0.73         4
weighted avg       0.83      0.75      0.73         4
```

1.14. Evaluasi model dengan Mathews Correlation Coefficient

Matthews Correlation Coefficient (MCC)

- MCC is an alternative to the F1 score for measuring the performance of binary classifiers.
- A perfect classifier's MCC is 1.
- A trivial classifier that predicts randomly will score 0, and a perfectly wrong classifier will score -1.

$$MCC = \frac{tp \times tn + fp \times fn}{\sqrt{(tp + fp) \times (tp + fn) \times (tn + fp) \times (tn + fn)}}$$

```
[ ] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import matthews_corrcoef

    mcc = matthews_corrcoef(y_test, y_pred)

    print(f'MCC: {mcc}')
```

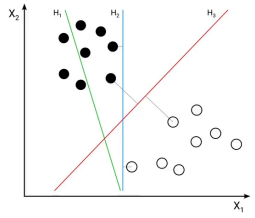
```
Hanif Ridal Warits - 41155050210060
MCC: 0.5773502691896258
```

2.0. Support Vector Machine (SVM). Lakukan praktik dari https://youtu.be/z69XYXpvVrE?si=KR_hDSlwjGIMcT0w , buat screenshot dengan nama kalian pada coding, kumpulkan dalam bentuk pdf, dari kegiatan ini:

2.1. Pengenalan Decision Boundary & Hyperplane

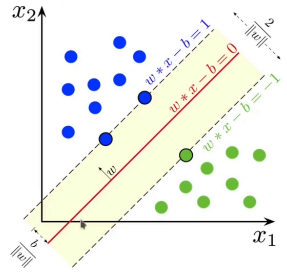
Konsep Dasar

Decision Boundary (Hyperplane)



2.2. Pengenalan Support Vector & Maximum Margin

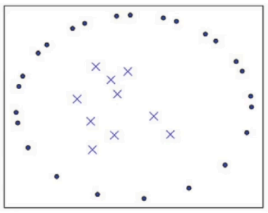
Maximum Margin

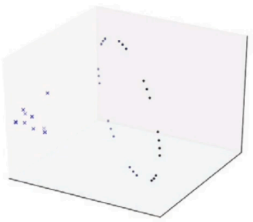


2.3. Pengenalan kondisi Linearly Inseparable dan Kernel Tricks

Linearly Inseperable
& Kernel Tricks

Referensi: https://www.quora.com/What-is-the-kernel-trick





2.4. Pengenalan MNIST Handwritten Digits Dataset

Classification Task dengan Support Vector Machine (SVM)

Referensi: https://www.svm-tutorial.com/

Dataset: The MNIST database of handwritten digits

```
[1] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.datasets import fetch_openml

    X, y = fetch_openml('mnist_784', data_home='./dataset/mnist', return_X_y=True)
    X.shape
```

```
Hanif Ridal Warits - 41155050210060
/usr/local/lib/python3.10/dist-packages/sklearn/datasets/_openml.py:107: UserWarning: A network error occurred while downloading https://api.openml.org/api/v1/json/data/qualities/554.
  warn(
(70000, 784)
```

```
[2] print('Hanif Ridal Warits - 41155050210060')

    import matplotlib.pyplot as plt
    import matplotlib.cm as cm

    pos = 1
    for data in X.to_numpy()[:8]:
        plt.subplot(1, 8, pos)
        plt.imshow(data.reshape((28, 28)),
                   cmap=cm.Greys_r)
        plt.axis('off')
        pos += 1

    plt.show()
```

```
Hanif Ridal Warits - 41155050210060
```

```
[3] print('Hanif Ridal Warits - 41155050210060')

    y[:8]
```

```
print('Hanif Ridal Warits - 41155050210060')

y[:8]
```

```
Hanif Ridal Warits - 41155050210060
```

	class
0	5
1	0
2	4
3	1
4	9
5	2
6	1
7	3

dtype: category

```
[4] print('Hanif Ridal Warits - 41155050210060')

    # X_train = X[:60000]
    # y_train = y[:60000]
    # X_test = X[60000:]
    # y_test = y[60000:]
```

```
Hanif Ridal Warits - 41155050210060
```

2.5. Klasifikasi dengan Support Vector Classifier | SVC

Classification dengan SVC (Support Vector Classifier)

```
[5] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.svm import SVC

    model = SVC(random_state=0)
    model.fit(X_train, y_train)
```

```
Hanif Ridal Warits - 41155050210060
SVC(random_state=0)
```

```
[6] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.metrics import classification_report

    y_pred = model.predict(X_test)
    print(classification_report(y_test, y_pred))
```

```
Hanif Ridal Warits - 41155050210060
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       102
           1       0.97      0.99      0.98       119
           2       0.85      0.82      0.84        99
           3       0.97      0.87      0.92       102
           4       0.88      0.95      0.91        92
           5       0.91      0.86      0.88        85
           6       0.93      0.95      0.94       102
           7       0.92      0.94      0.93       115
           8       0.89      0.94      0.91        94
           9       0.92      0.84      0.88        90

    accuracy                           0.92      1000
   macro avg       0.92      0.91      0.91      1000
weighted avg       0.92      0.92      0.92      1000
```

2.6. Hyperparameter Tuning dengan Grid Search

Hyperparameter Tuning dengan GridSearchCV

```
[7] print('Hanif Ridal Warits - 41155050210060')

    from sklearn.model_selection import GridSearchCV

    parameters = {
        'kernel': ['rbf', 'poly', 'sigmoid'],
        'C': [0.5, 1, 10, 100],
        'gamma': ['scale', 1, 0.1, 0.01, 0.001]
    }

    grid_search = GridSearchCV(estimator=SVC(random_state=0),
                               param_grid=parameters,
                               n_jobs=6,
                               verbose=1,
                               scoring='accuracy')

    grid_search.fit(X_train, y_train)
```

```
Hanif Ridal Warits - 41155050210060
Fitting 5 folds for each of 60 candidates, totalling 300 fits
GridSearchCV
best_estimator_: SVC
SVC
```

```
[8] print('Hanif Ridal Warits - 41155050210060')

    print(f'Best Score: {grid_search.best_score_}')

    best_params = grid_search.best_estimator_.get_params()
    print(f'Best Parameters:')
    for param in parameters:
        print(f'\t{param}: {best_params[param]}')
```

```
Hanif Ridal Warits - 41155050210060
Best Score: 0.907
Best Parameters:
        kernel: rbf
        C: 10
        gamma: scale
```

2.7. Evaluasi Model

Predict & Evaluate

```
print('Hanif Ridal Warits - 41155050210060')

y_pred = grid_search.predict(X_test)

print(classification_report(y_test, y_pred))
```

```
Hanif Ridal Warits - 41155050210060
              precision    recall  f1-score   support

           0       0.93      0.98      0.96       102
           1       0.98      0.99      0.98       119
           2       0.87      0.85      0.86        99
           3       0.99      0.89      0.94       102
           4       0.91      0.95      0.93        92
           5       0.92      0.89      0.90        85
           6       0.93      0.94      0.94       102
           7       0.93      0.93      0.93       115
           8       0.89      0.95      0.92        94
           9       0.92      0.88      0.90        90

    accuracy                           0.93      1000
   macro avg       0.93      0.92      0.92      1000
weighted avg       0.93      0.93      0.93      1000
```