Nama : Hafsah Islamiati Ayuningtyas

NPM : 41155050210039

Kelas : INF-A2

Mata Kuliah : Machine Learning

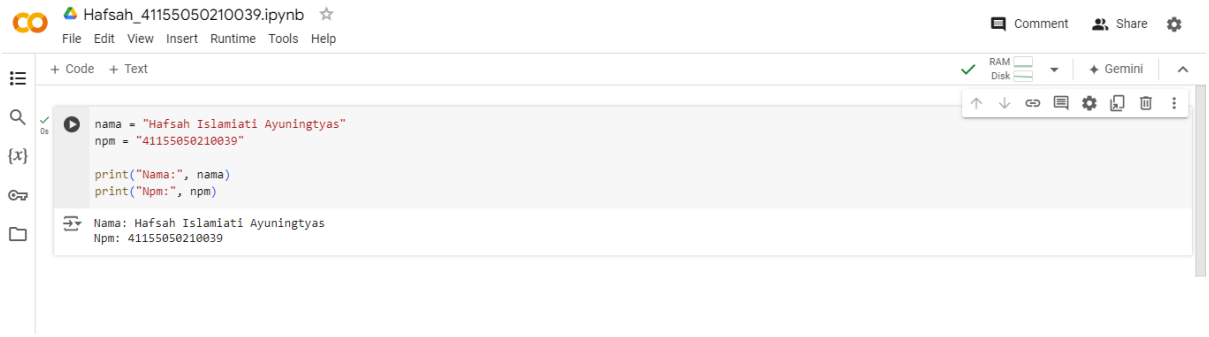
Tugas Pertemuan 1

1.2. Tuliskan nama dan nomor NPM anda pada Jupiter Notebook

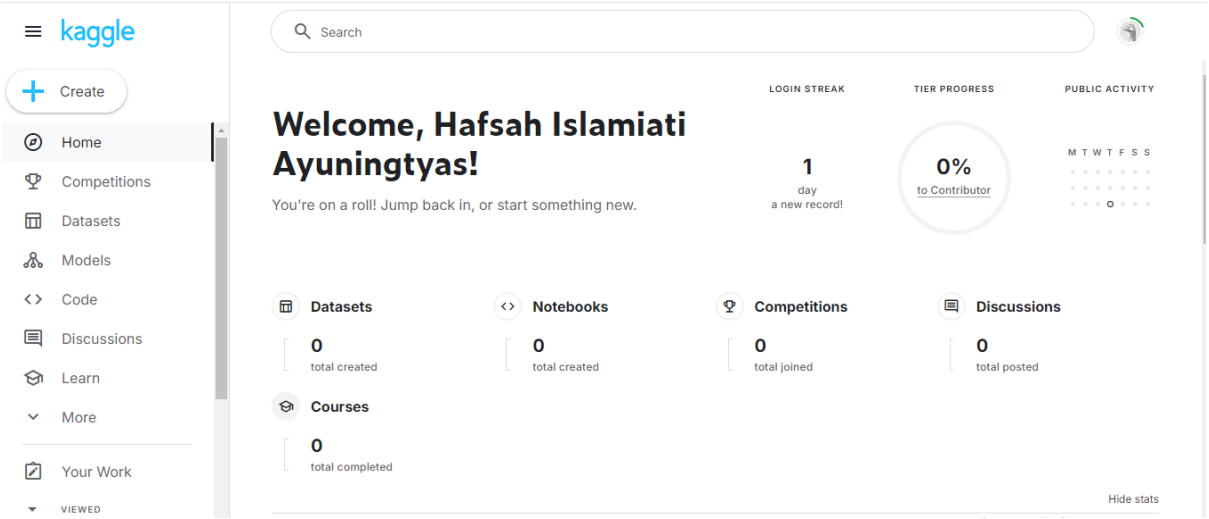
```
nama = "Hafsah Islamiati Ayuningtyas"
npm = "41155050210039"

print("Nama:", nama)
print("Npm:", npm)
```

```
Nama: Hafsah Islamiati Ayuningtyas
Npm: 41155050210039
```

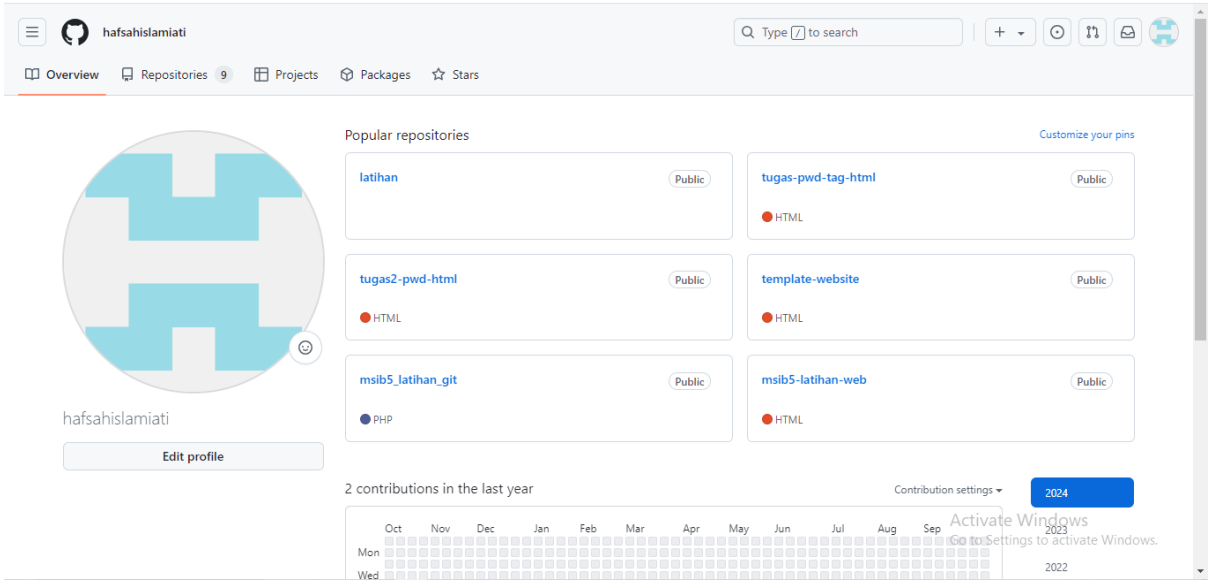
2.2. Tuliskan nama dan nomor NPM anda pada Google Colab.



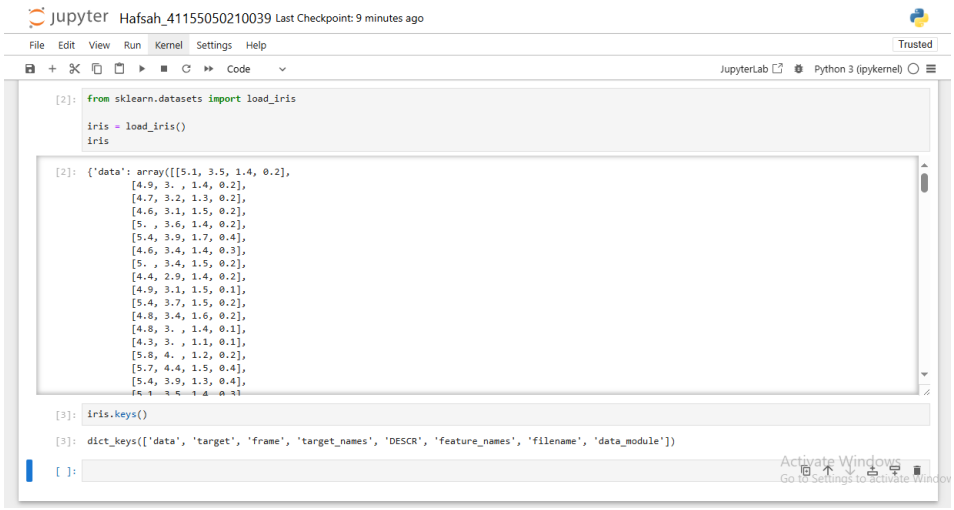
3.0 Akun Kaggle



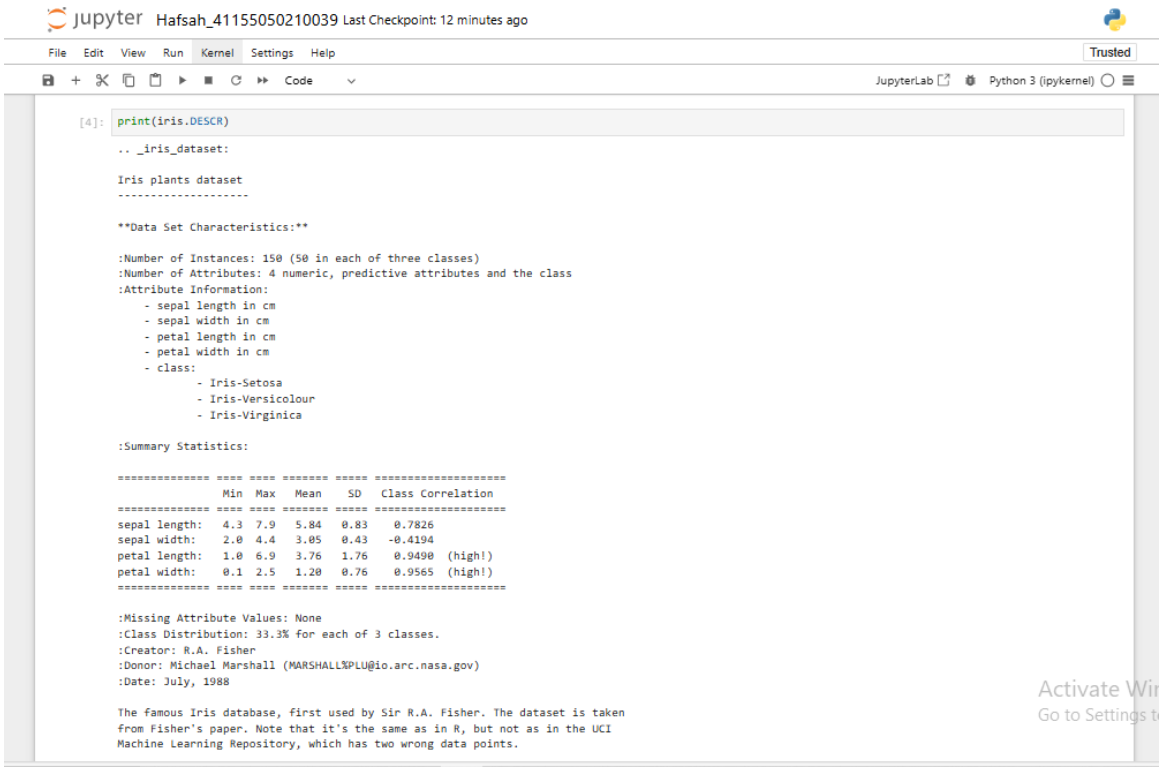
4.0 Akun GitHub



5.1 Load Sample Dataset

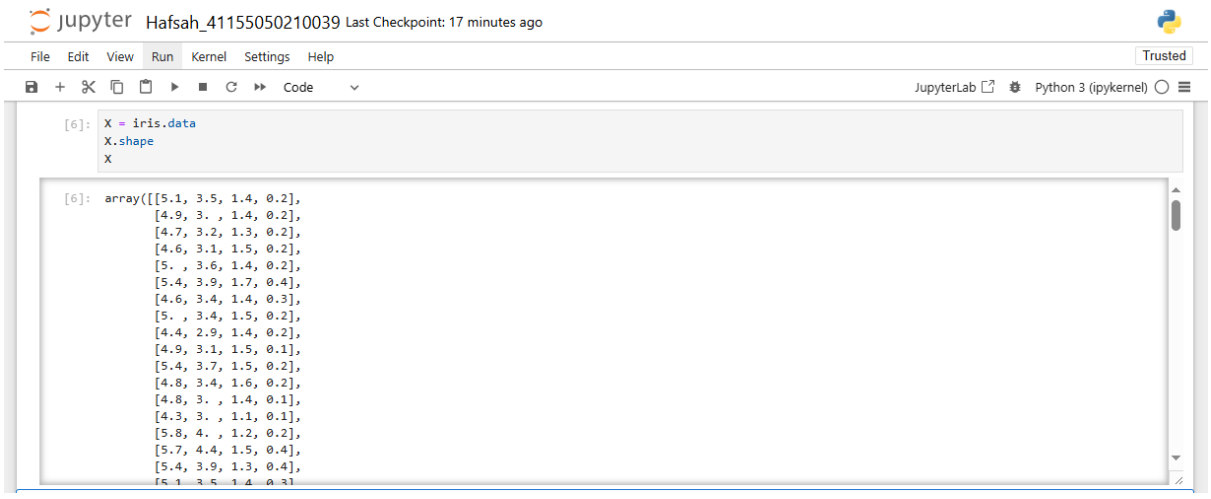


5.2 Metadata | Deskripsi dari sample dataset



5.3. Explanatory & Response Variables | Features & Target

```
[5]: X = iris.data
     X.shape
     # X

[5]: (150, 4)
```




```
[8]: y = iris.target
     y.shape
     # y

[8]: (150,)
```

```
[9]: y = iris.target
     y.shape
     y

[9]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
            1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
            2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])

[ ]:
```

5.4. Feature & Target Names

```
[11]: feature_names = iris.feature_names
      feature_names

[11]: ['sepal length (cm)',
       'sepal width (cm)',
       'petal length (cm)',
       'petal width (cm)']

[12]: target_names = iris.target_names
      target_names

[12]: array(['setosa', 'versicolor', 'virginica'], dtype='<U10')

[ ]:
```

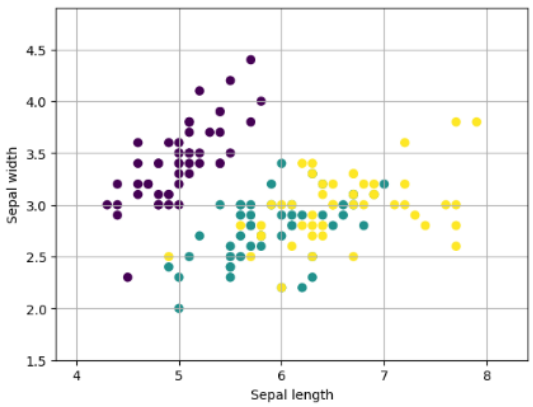
5.5. Visualisasi Data

jupyter Hafsah_41155050210039 Last Checkpoint: 29 minutes ago

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python 3 (ipykernel)

```
[14]: import matplotlib.pyplot as plt

      X = X[:, :2]

      x_min, x_max = X[:, 0].min() - 0.5, X[:, 0].max() + 0.5
      y_min, y_max = X[:, 1].min() -0.5, X[:, 1].max() + 0.5

      plt.scatter(X[:, 0], X[:, 1], c=y)
      plt.xlabel('Sepal length')
      plt.ylabel('Sepal width')

      plt.xlim(x_min, x_max)
      plt.ylim(y_min, y_max)
      plt.grid(True)
      plt.show()
```




```
[ ]:
```

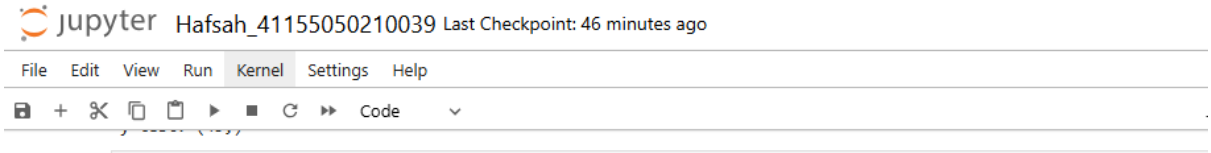
5.6. Training Set & Testing Set

```
[16]: from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    test_size=0.3,
                                                    random_state=1)
print(f'X train: {X_train.shape}')
print(f'X test: {X_test.shape}')
print(f'y train: {y_train.shape}')
print(f'y test: {y_test.shape}')
```

```
X train: (105, 2)
X test: (45, 2)
y train: (105,)
y test: (45,)
```

```
[ ]:
```

5.7. Load sample dataset sebagai Pandas Data Frame




```
[20]: iris = load_iris(as_frame=True)

iris_features_df = iris.data
iris_features_df
```

[20]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

6.1. Persiapan dataset | Loading & splitting dataset

```
[22]: from sklearn.datasets import load_iris

      iris = load_iris()

      x = iris.data
      y = iris.target
```

```
[23]: from sklearn.model_selection import train_test_split

      X_train, X_test, y_train, y_test = train_test_split(X,
                                                          y,
                                                          test_size=0.4,
                                                          random_state=1)
```

```
[ ]:
```

6.2. Training model Machine Learning

```
[25]: from sklearn.neighbors import KNeighborsClassifier

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

```
[25]: KNeighborsClassifier
      KNeighborsClassifier(n_neighbors=3)
```

6.3. Evaluasi model Machine Learning

```
from sklearn.metrics import accuracy_score

y_pred = model.predict(X_test)
acc = accuracy_score(y_test, y_pred)
print(f'Accuracy : {acc}')
```

```
Accuracy : 0.9833333333333333
```

6.4. Pemanfaatan trained model machine learning

```
data_baru = [[5, 5, 3, 2],
             [2, 4, 3, 5]]
preds = model.predict(data_baru)
preds
```

```
array([1, 2])
```

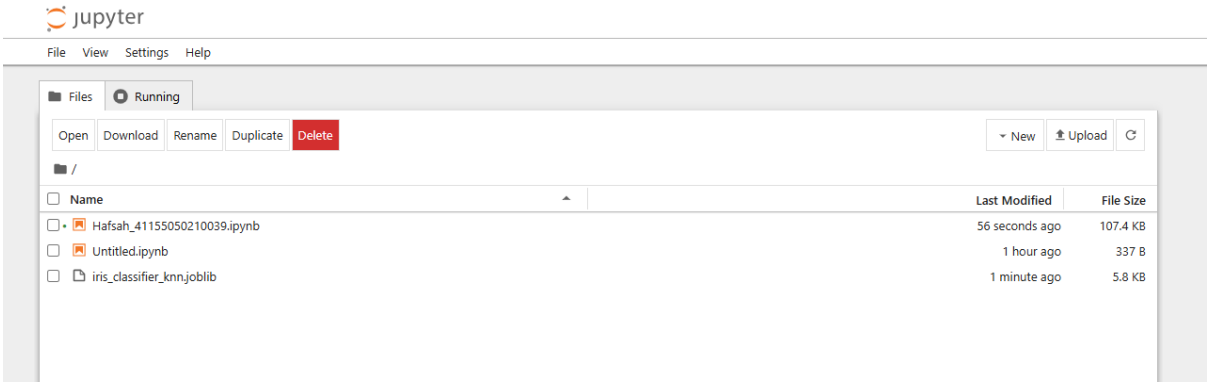
```
pred_species = [iris.target_names[p] for p in preds]
print (f'Hasil Prediksi : {pred_species}')
```

```
Hasil Prediksi : [np.str_('versicolor'), np.str_('virginica')]
```

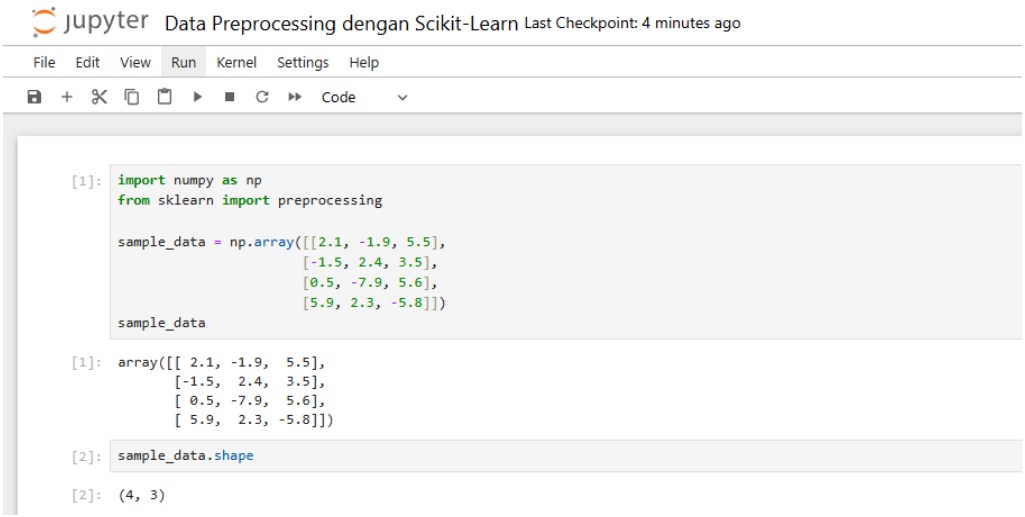
6.5. Deploy model Machine Learning | Dumping dan Loading model Machine Learning

```
[33]: import joblib
      joblib.dump(model, 'iris_classifier_knn.joblib')
```

```
[33]: ['iris_classifier_knn.joblib']
```



7.1. Persiapan sample dataset




```
[1]: import numpy as np
     from sklearn import preprocessing

     sample_data = np.array([[2.1, -1.9, 5.5],
                             [-1.5, 2.4, 3.5],
                             [0.5, -7.9, 5.6],
                             [5.9, 2.3, -5.8]])
     sample_data
```

```
[1]: array([[ 2.1, -1.9,  5.5],
            [-1.5,  2.4,  3.5],
            [ 0.5, -7.9,  5.6],
            [ 5.9,  2.3, -5.8]])
```

```
[2]: sample_data.shape
```

```
[2]: (4, 3)
```

7.2. Teknik data preprocessing 1: binarisation

```
[3]: sample_data
```

```
[3]: array([[ 2.1, -1.9,  5.5],
            [-1.5,  2.4,  3.5],
            [ 0.5, -7.9,  5.6],
            [ 5.9,  2.3, -5.8]])
```

```
[5]: preprocessor = preprocessing.Binarizer(threshold=0.5)
     binarised_data = preprocessor.transform(sample_data)
     binarised_data
```

```
[5]: array([[1., 0., 1.],
            [0., 1., 1.],
            [0., 0., 1.],
            [1., 1., 0.]])
```

7.3. Teknik data preprocessing 2: scaling

```
[6]: sample_data
```

```
[6]: array([[ 2.1, -1.9,  5.5],
            [-1.5,  2.4,  3.5],
            [ 0.5, -7.9,  5.6],
            [ 5.9,  2.3, -5.8]])
```

```
[7]: preprocessor = preprocessing.MinMaxScaler(feature_range=(0, 1))
     preprocessor.fit(sample_data)
     scaled_data = preprocessor.transform(sample_data)
     scaled_data
```

```
[7]: array([[0.48648649, 0.58252427, 0.99122807],
            [0.        , 1.        , 0.81578947],
            [0.27027027, 0.        , 1.        ],
            [1.        , 0.99029126, 0.        ]])
```

```
[8]: scaled_data = preprocessor.fit_transform(sample_data)
     scaled_data
```

```
[8]: array([[0.48648649, 0.58252427, 0.99122807],
            [0.        , 1.        , 0.81578947],
            [0.27027027, 0.        , 1.        ],
            [1.        , 0.99029126, 0.        ]])
```

7.4. Teknik data preprocessing 3: normalisation

- L1 Normalisation: Least Absolute Deviations

```
[6]: sample_data
```

```
[6]: array([[ 2.1, -1.9,  5.5],
            [-1.5,  2.4,  3.5],
            [ 0.5, -7.9,  5.6],
            [ 5.9,  2.3, -5.8]])
```

```
[7]: preprocessor = preprocessing.MinMaxScaler(feature_range=(0, 1))
     preprocessor.fit(sample_data)
     scaled_data = preprocessor.transform(sample_data)
     scaled_data
```

```
[7]: array([[0.48648649, 0.58252427, 0.99122807],
            [0.        , 1.        , 0.81578947],
            [0.27027027, 0.        , 1.        ],
            [1.        , 0.99029126, 0.        ]])
```

```
[8]: scaled_data = preprocessor.fit_transform(sample_data)
     scaled_data
```

```
[8]: array([[0.48648649, 0.58252427, 0.99122807],
            [0.        , 1.        , 0.81578947],
            [0.27027027, 0.        , 1.        ],
            [1.        , 0.99029126, 0.        ]])
```

- L2 Normalisation: Least Squares

[11]:
```
sample_data
```

[11]:
```
array([[ 2.1, -1.9,  5.5],
       [-1.5,  2.4,  3.5],
       [ 0.5, -7.9,  5.6],
       [ 5.9,  2.3, -5.8]])
```

[12]:
```
l2_normalised_data = preprocessing.normalize(sample_data, norm='l2')
l2_normalised_data
```

[12]:
```
array([[ 0.33946114, -0.30713151,  0.88906489],
       [-0.33325106,  0.53320169,  0.7775858 ],
       [ 0.05156558, -0.81473612,  0.57753446],
       [ 0.68706914,  0.26784051, -0.6754239 ]])
```