

TUGAS

MACHINE LEARNING



Disusun oleh :

Nama : Ichsan Haryadi Putra

NPM : 41155050210042

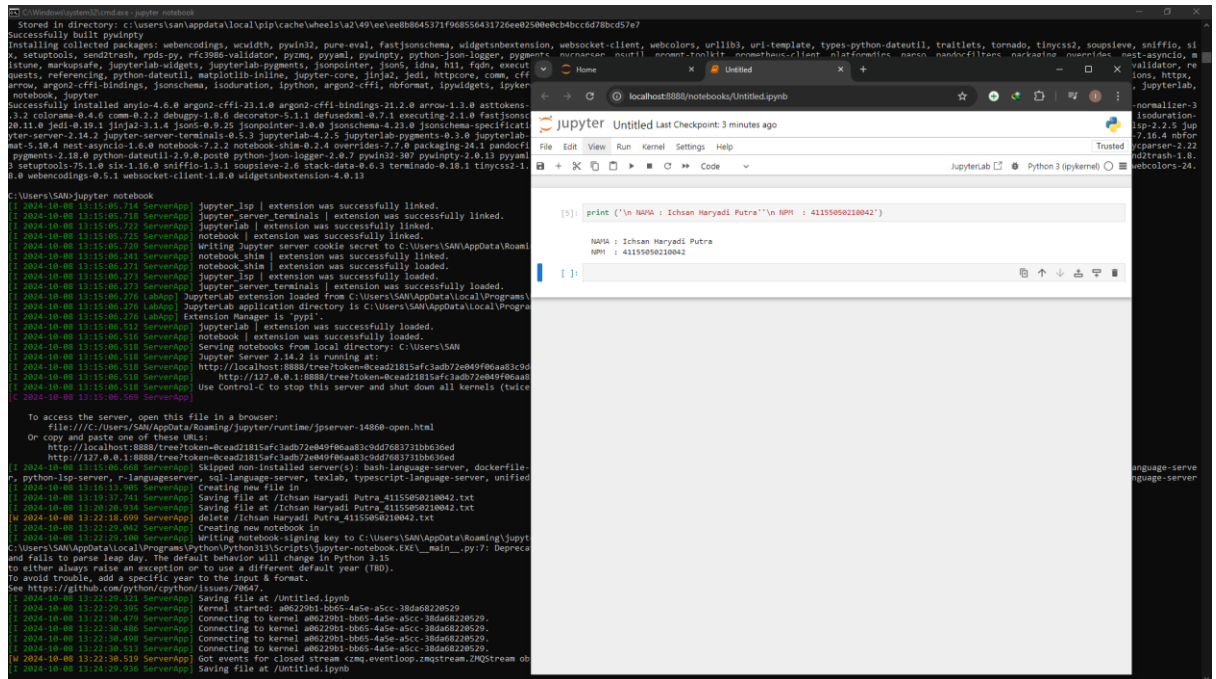
Kelas : Teknik Informatika B

UNIVERSITAS LANGLANGBUANA

BANDUNG

Tugas Pertemuan 1

1. Instalasi Jupyter Notebook menggunakan CMD



The screenshot shows the installation and execution of Jupyter Notebook. On the left, a Windows Command Prompt window displays the output of the `pip install jupyter` command. The output lists various installed packages and their versions, including `anyio-4.6.0`, `argon2-cffi-23.1.0`, `argon2-cffi-bindings-21.2.0`, `arrow-1.3.0`, `asttokens-2.2.0`, `colorama-0.4.6`, `comm-0.2.2`, `debugpy-1.8.6`, `decorator-5.1.1`, `defusedxml-0.7.1`, `executing-2.1.0`, `fastjsonschema-2.19.1`, `jedi-0.19.1`, `jinja2-3.1.4`, `json-0.9.25`, `jsonpointer-3.0.0`, `jsonschema-4.23.0`, `jsonschema-specific-0.0.1`, `jupyter-server-2.14.2`, `jupyter-server-terminals-0.5.3`, `jupyterlab-4.2.5`, `jupyterlab-pygments-0.3.0`, `jupyterlab-widgets-2.18.0`, `matplotlib-inline-0.1.6`, `python-dateutil-2.9.0.post0`, `python-json-logger-2.0.7`, `pywin32-307`, `pywinpty-2.0.13`, `pyyaml-6.0.1`, `setuptools-75.1.0`, `six-1.16.0`, `sniffio-1.3.1`, `soupsieve-2.6`, `stack-data-0.6.3`, `terminado-0.18.1`, `tinycss2-1.2.0`, `webencodings-0.5.1`, `websocket-client-1.8.0`, `widgitextension-4.0.11`, and `jupyter`.

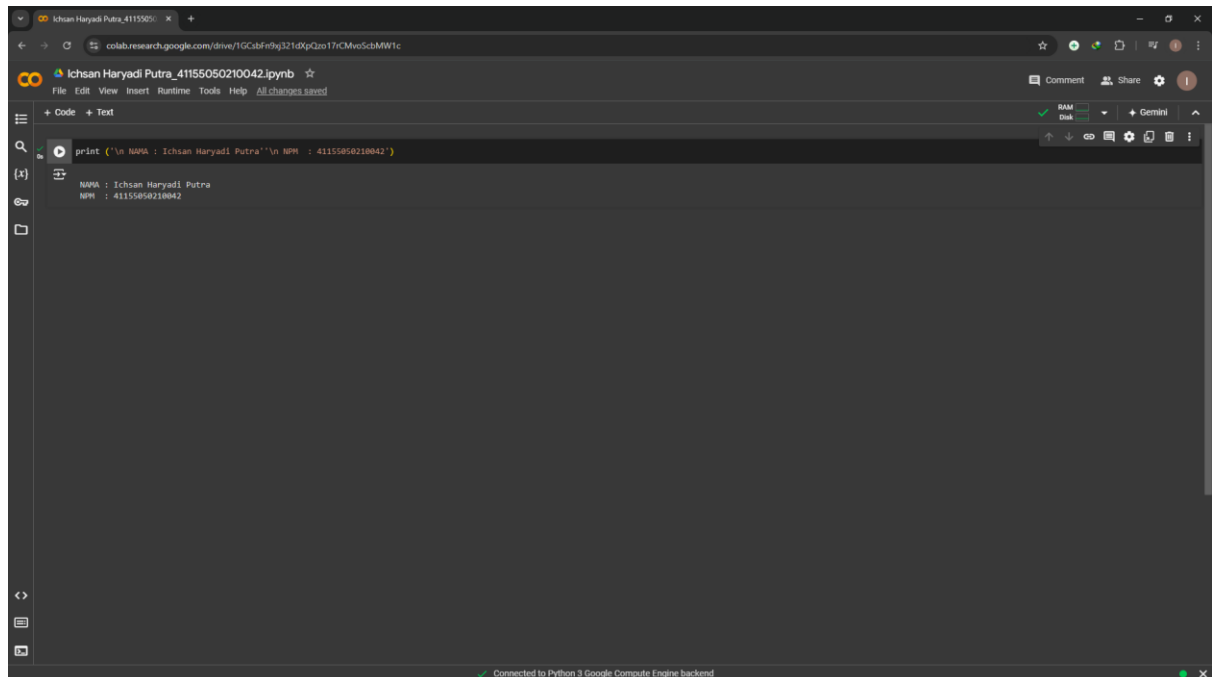
On the right, a web browser window shows the Jupyter Notebook interface. The URL bar displays `localhost:8888/notebooks/Untitled.ipynb`. The notebook content shows a single cell with the following code:

```
print('NAMA : Ichan Haryadi Putra' '\n NPM : 41155050210042')
```

The output of the code is displayed below the cell:

```
NAMA : Ichan Haryadi Putra
NPM : 41155050210042
```

2. Google Collab



The screenshot shows a Google Colab notebook interface. The URL bar displays `colab.research.google.com/drive/1GCaBfmg321dXpQzo17CMoScbMMW1c`. The notebook content shows a single cell with the following code:

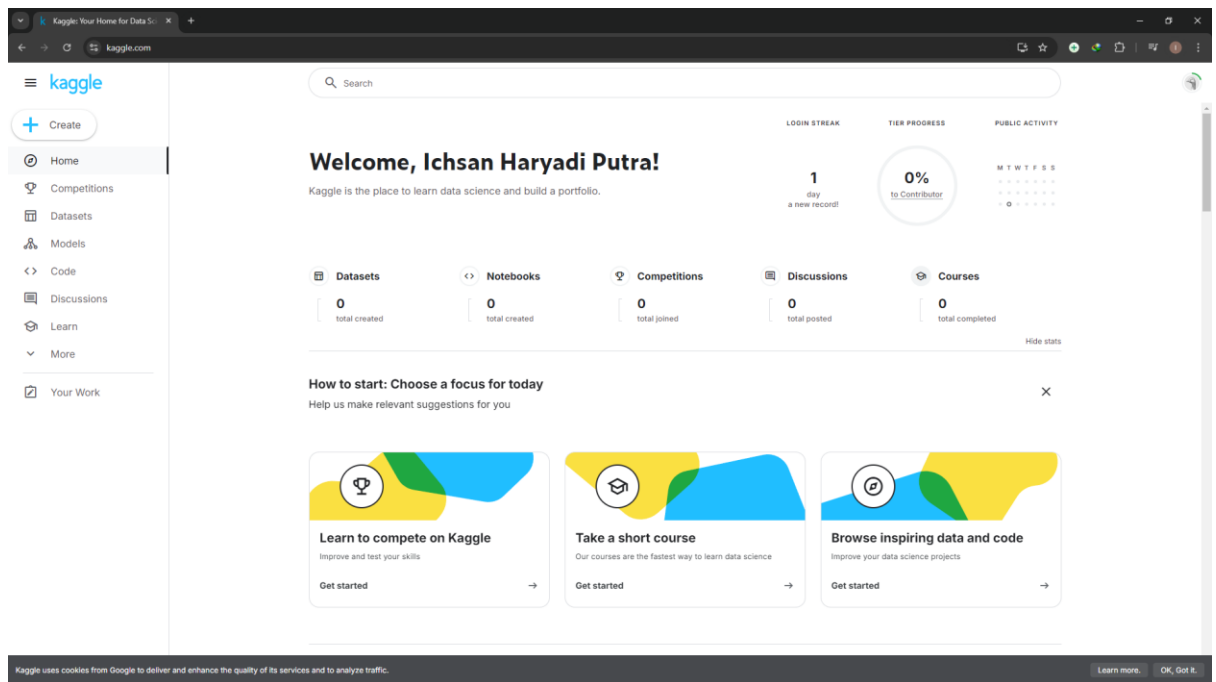
```
print('NAMA : Ichan Haryadi Putra' '\n NPM : 41155050210042')
```

The output of the code is displayed below the cell:

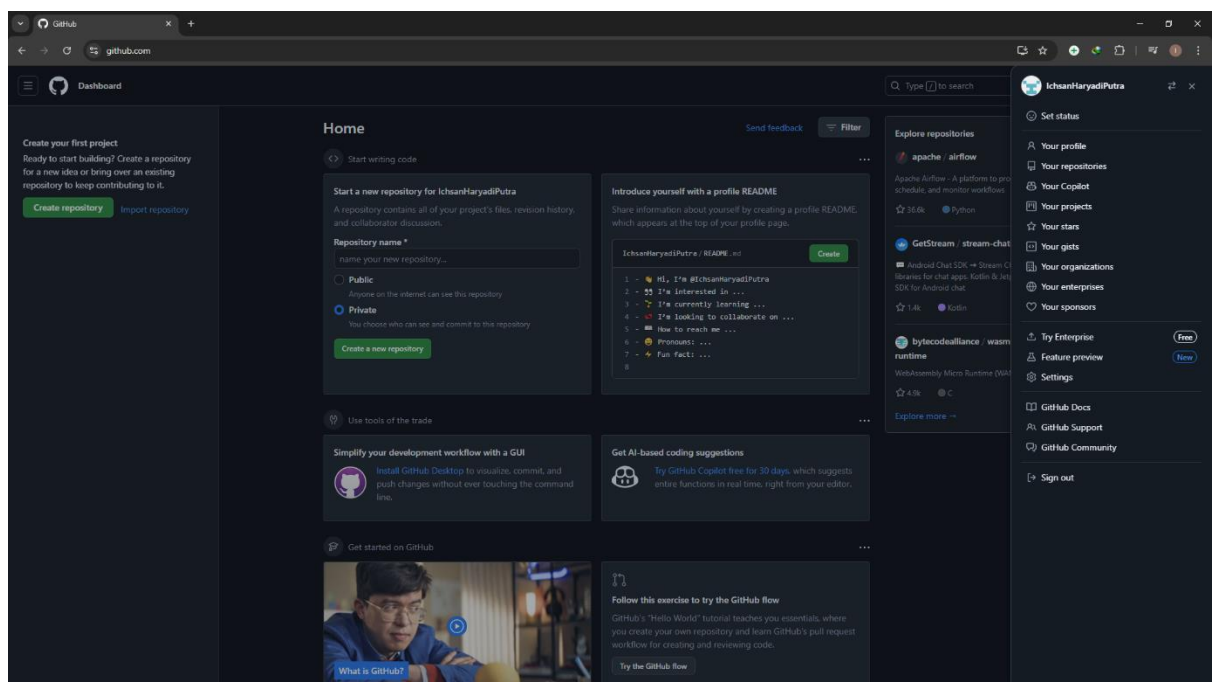
```
NAMA : Ichan Haryadi Putra
NPM : 41155050210042
```

The bottom of the notebook shows the status bar with the text "Connected to Python 3 Google Compute Engine backend".

3. Membuat akun Kaggle

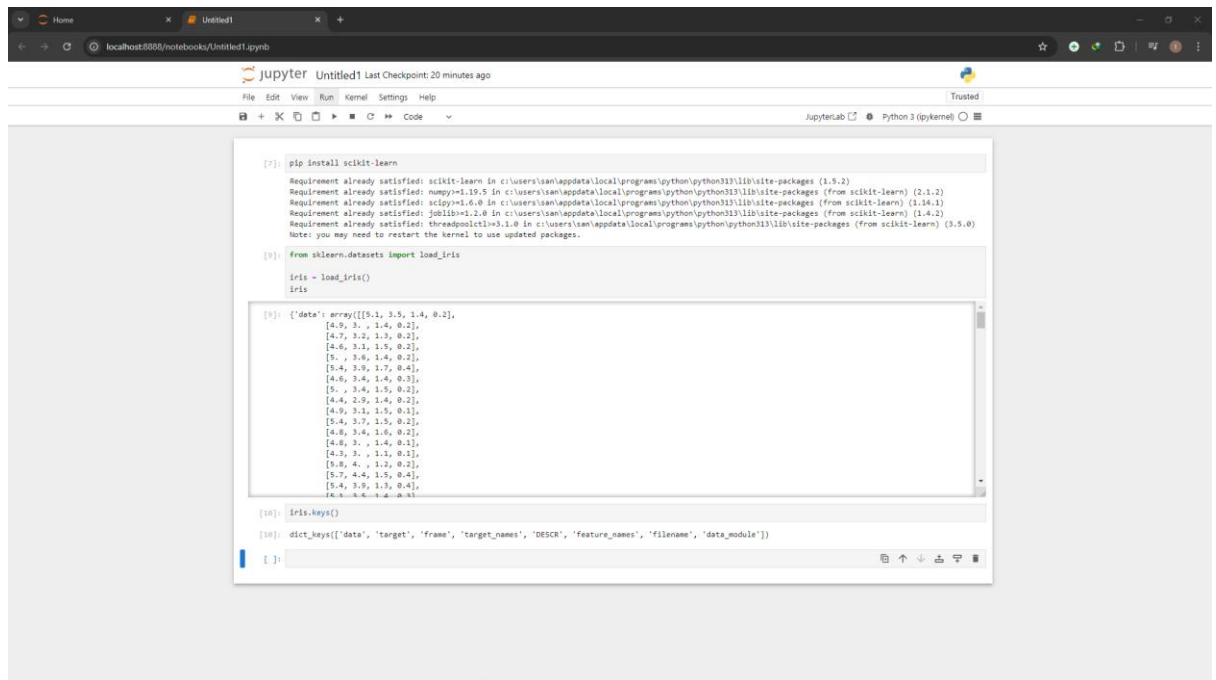


4. Membuat akun github



5. Praktek

- Load Sample dataset dari scikit-learn



The screenshot shows a Jupyter Notebook interface with a code cell containing the following Python code:

```
[7]: pip install scikit-learn

Requirement already satisfied: scikit-learn in c:\users\san\appdata\local\programs\python\python311\lib\site-packages (1.5.2)
Requirement already satisfied: numpy>=1.19.5 in c:\users\san\appdata\local\programs\python\python311\lib\site-packages (from scikit-learn) (2.1.2)
Requirement already satisfied: scipy>=1.6.0 in c:\users\san\appdata\local\programs\python\python311\lib\site-packages (from scikit-learn) (1.14.1)
Requirement already satisfied: joblib>=1.2.0 in c:\users\san\appdata\local\programs\python\python311\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\san\appdata\local\programs\python\python311\lib\site-packages (from scikit-learn) (3.5.0)
Note: you may need to restart the kernel to use updated packages.

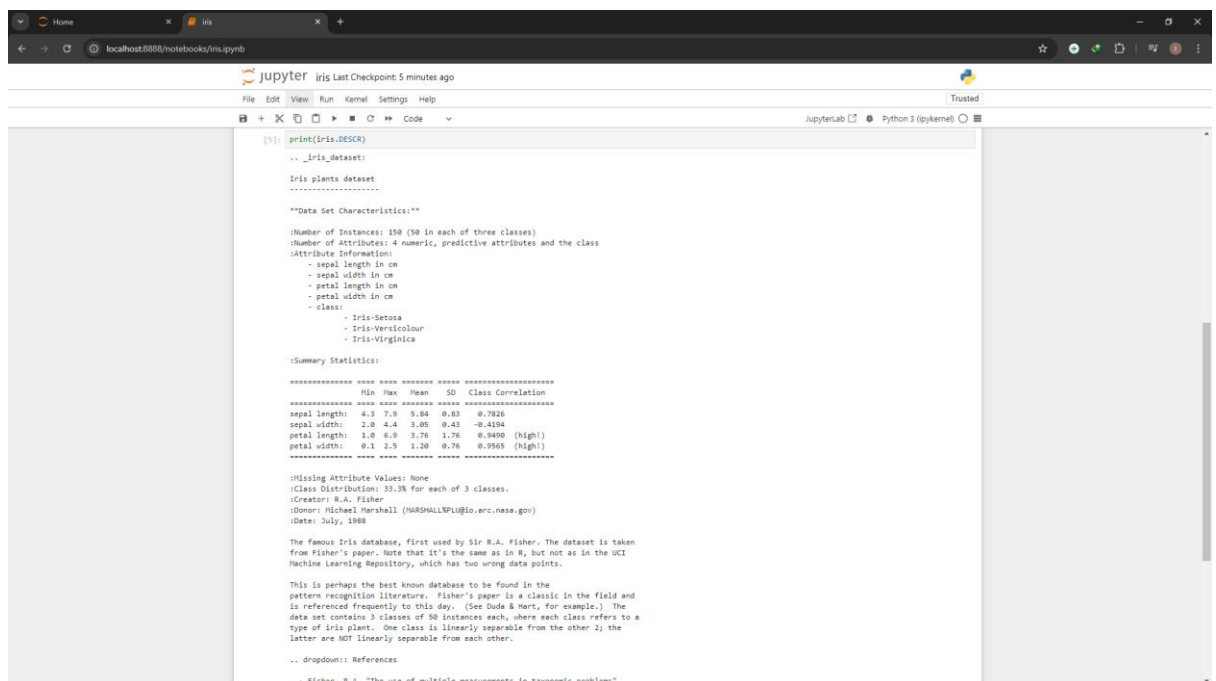
[9]: from sklearn.datasets import load_iris

iris = load_iris()
iris
```

The output of the code cell shows the Iris dataset as a dictionary with the following structure:

```
[9]: {'data': array([[5.1, 3.5, 1.4, 0.2],
[4.9, 3. , 1.4, 0.2],
[4.7, 3.2, 1.3, 0.2],
[4.6, 3.1, 1.5, 0.2],
[5. , 3.6, 1.4, 0.2],
[5.4, 3.9, 1.7, 0.4],
[4.6, 3.4, 1.4, 0.2],
[5. , 3.4, 1.5, 0.2],
[4.4, 2.9, 1.4, 0.2],
[4.9, 3.1, 1.5, 0.1],
[5.4, 3.7, 1.5, 0.2],
[4.8, 3.4, 1.6, 0.2],
[4.8, 3. , 1.4, 0.1],
[4.3, 3. , 1.5, 0.1],
[5.0, 4. , 1.2, 0.2],
[5.7, 4.4, 1.5, 0.4],
[5.4, 3.9, 1.5, 0.4],
[4.4, 3. , 1.4, 0.1]]),
'target': array([0, 1, 2]),
'frame': None,
'target_names': None,
'DESC': None,
'feature_names': None,
'filename': None,
'data_module': None})
```

- Metadata | Deskripsi dari sample dataset



The screenshot shows a Jupyter Notebook interface with a code cell containing the following Python code:

```
[5]: print(iris.DESC)

.. _iris_dataset:

Iris plants dataset
-----

**Data Set Characteristics:**

:Number of Instances: 150 (50 in each of three classes)
:Number of Attributes: 4 numeric, predictive attributes and the class
:Attribute Information:
  - sepal length in cm
  - sepal width in cm
  - petal length in cm
  - petal width in cm
  - class:
    - Iris-Setosa
    - Iris-Versicolour
    - Iris-Virginica

:Summary Statistics:

=====
      Min      Max      Mean      SD      Class Correlation
=====
sepal length:  4.3      7.9      5.84      0.83      0.7826
sepal width:   2.0      4.4      3.86      0.43      -0.4134
petal length:  1.0      6.9      3.76      1.76      0.9490 (high!)
petal width:   0.1      1.9      1.20      0.76      0.9965 (high!)
=====

:Missing Attribute Values: None
:Class Distribution: 33.3% for each of 3 classes.
:Creator: R.A. Fisher
:Donor: Michael Marshall (MARSHALLNPL@bio.unc-nash.gov)
:Date: July, 1988

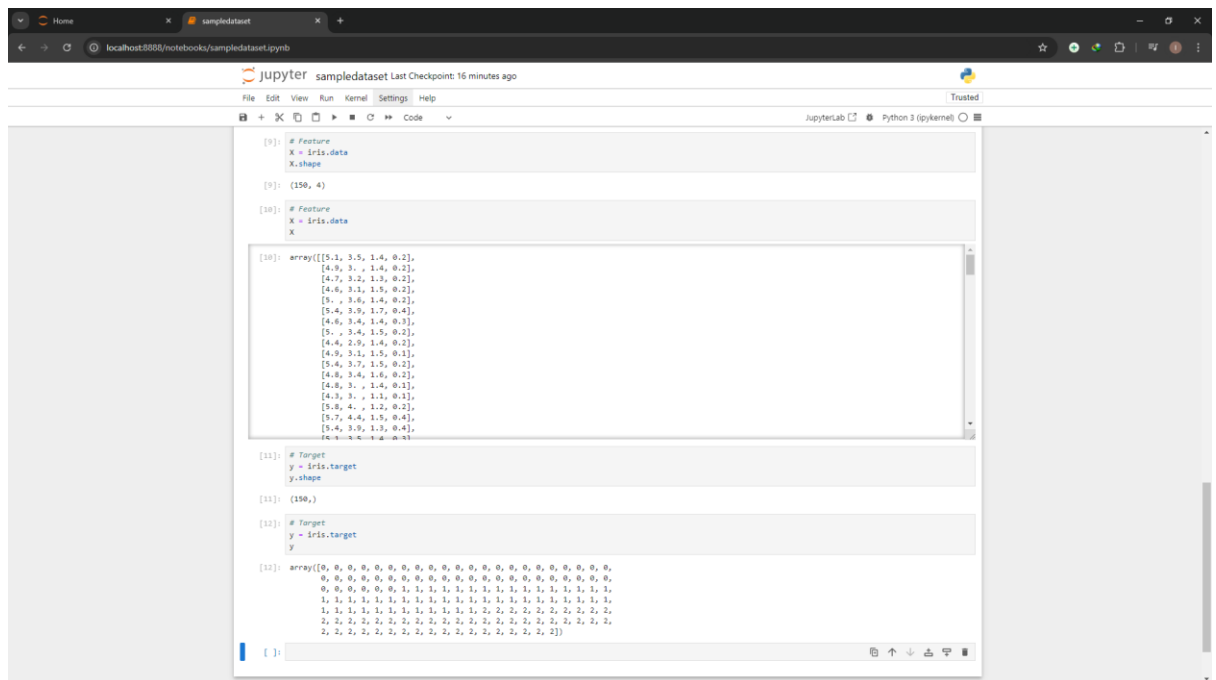
The Famous Iris database, first used by Sir R.A. Fisher. The dataset is taken
from Fisher's paper. Note that it's the same as in R, but not as in the UCI
Machine Learning Repository, which has two wrong data points.

This is perhaps the best known database to be found in the
pattern recognition literature. Fisher's paper is a classic in the field and
is referenced frequently to this day. (See Duda & Hart, for example.) The
data set contains 3 classes of 50 instances each, where each class refers to a
type of Iris plant. One class is linearly separable from the other 2; the
latter are NOT linearly separable from each other.

... dropdown:: References

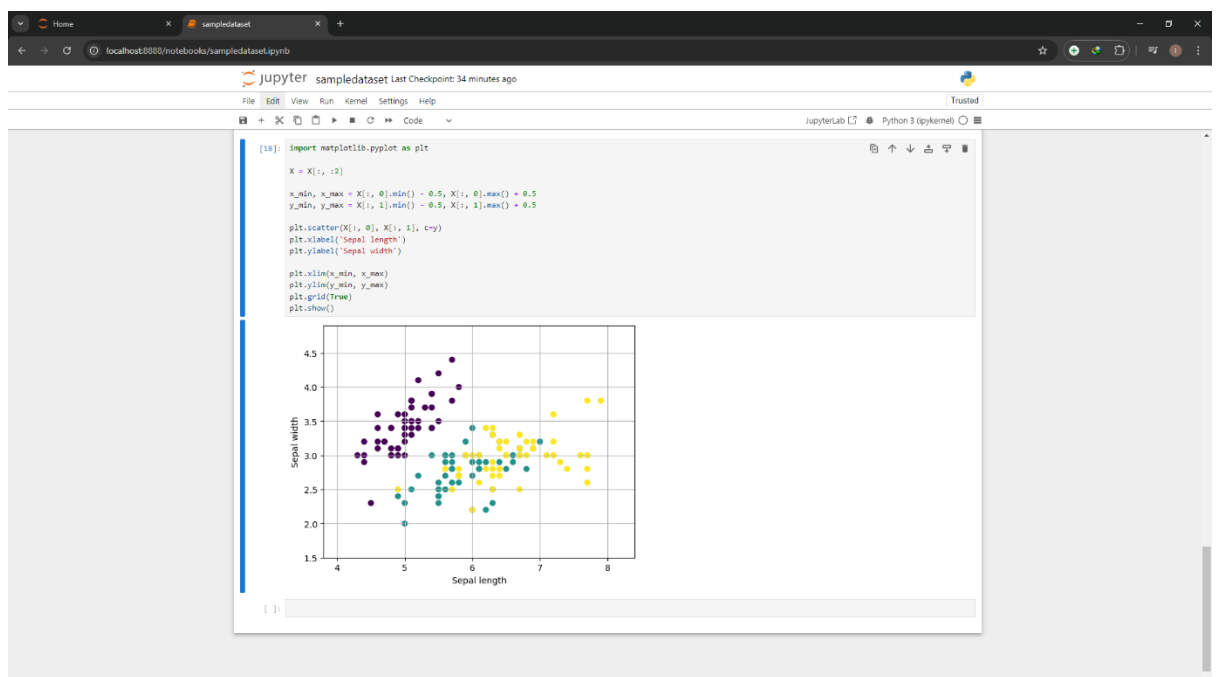
- Fisher, R.A. "The use of multiple measurements in taxonomic problems"
```

- Explanatory & Response Variables | Features & Target



- Feature & Target Names

- Visualisasi Data menggunakan matplotlib



- Training Set & Testing Set

```
[19]: 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,)
```

- Load sample dataset sebagai Pandas Data Frame

```
[21]: iris = load_iris(as_frame=True)
      iris_features_df = iris.data
      iris_features_df

[21]:
```

	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 x 4 columns

6. Praktek

- Persiapan dataset | Loading & splitting dataset

```
[6]: from sklearn.datasets import load_iris
      iris = load_iris()
      X = iris.data
      y = iris.target

[7]: 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)
```

- Training model Machine Learning

```
[8]: from sklearn.neighbors import KNeighborsClassifier
      model = KNeighborsClassifier(n_neighbors=3)
      model.fit(X_train, y_train)

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

- Evaluasi model Machine Learning

```
[9]: from sklearn.metrics import accuracy_score
      y_pred = model.predict(X_test)
      acc = accuracy_score(y_test, y_pred)
      print(f'Accuracy: {acc}')

      Accuracy: 0.9333333333333333
```

- Pemanfaatan trained model machine learning

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

[10]: array([1, 2])

[12]: pred_species = [iris.target_names[p] for p in preds]
      print('Hasil Prediksi: ', pred_species)
      Hasil Prediksi: [np.str_('versicolor'), np.str_('virginica')]
```

- Deploy model Machine Learning | Dumping dan Loading model Machine Learning

```
[17]: # Dumping Model
      import joblib
      joblib.dump(model, 'iris_classifier_knn.joblib')

[17]: ['iris_classifier_knn.joblib']

[18]: # Loading Model
      production_model = joblib.load('iris_classifier_knn.joblib')

[18]:
```

7. Praktek

- Persiapan sample dataset

```
[2]: 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

[2]: 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]])

[3]: sample_data.shape

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

- Teknik data preprocessing binarisation, scalling, normalisation

```
[1]: # Binarisation

[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.]])

[1]: # Scalling

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

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

[7]: scaled_data = preprocessor.fit_transform(sample_data)
      scaled_data

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

[1]: # Normalisation: Least Absolute Deviations

[8]: l1_normalised_data = preprocessing.normalize(sample_data, norm='l1')
      l1_normalised_data

[8]: array([[0.22185263, 0.2         , 0.57894737],
            [0.2027027 , 0.32432432, 0.47297297],
            [0.05714289, 0.56428571, 0.4         ],
            [0.42142857, 0.16428571, 0.41428571]])

[1]: # Normalisation: Least Squares

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

[9]: array([[0.33946114, 0.30711151, 0.88906489],
            [0.33325108, 0.53328169, 0.777858  ],
            [0.05156558, 0.81473612, 0.57753446],
            [0.68780934, 0.26784053, 0.6754239 ]])
```