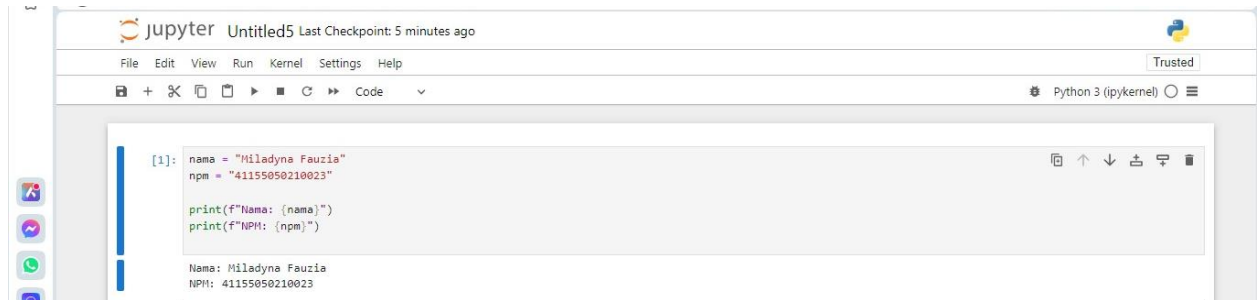


Nama : Miladyna Fauzia

Kelas : INF A1

NPM : 41155050210023

1. Tuliskan nama dan nomor NPM anda pada Jupiter Notebook.

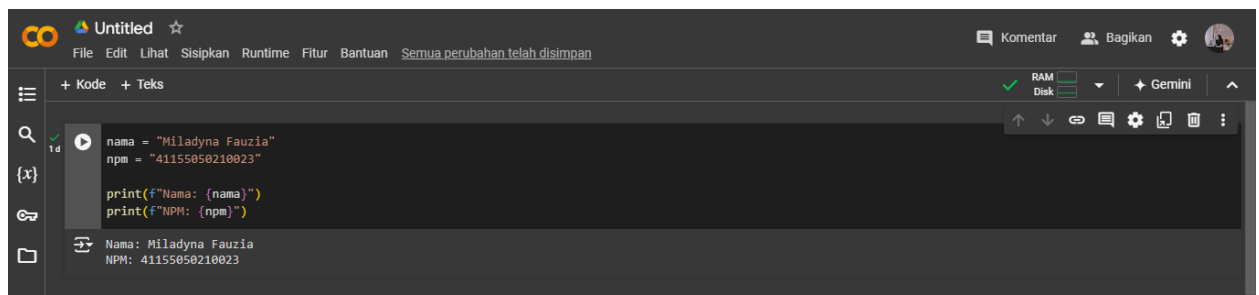


```
[1]: nama = "Miladyna Fauzia"
    npm = "41155050210023"

    print(f>Nama: {nama}")
    print(f"NPM: {npm}")
```

Nama: Miladyna Fauzia  
NPM: 41155050210023

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

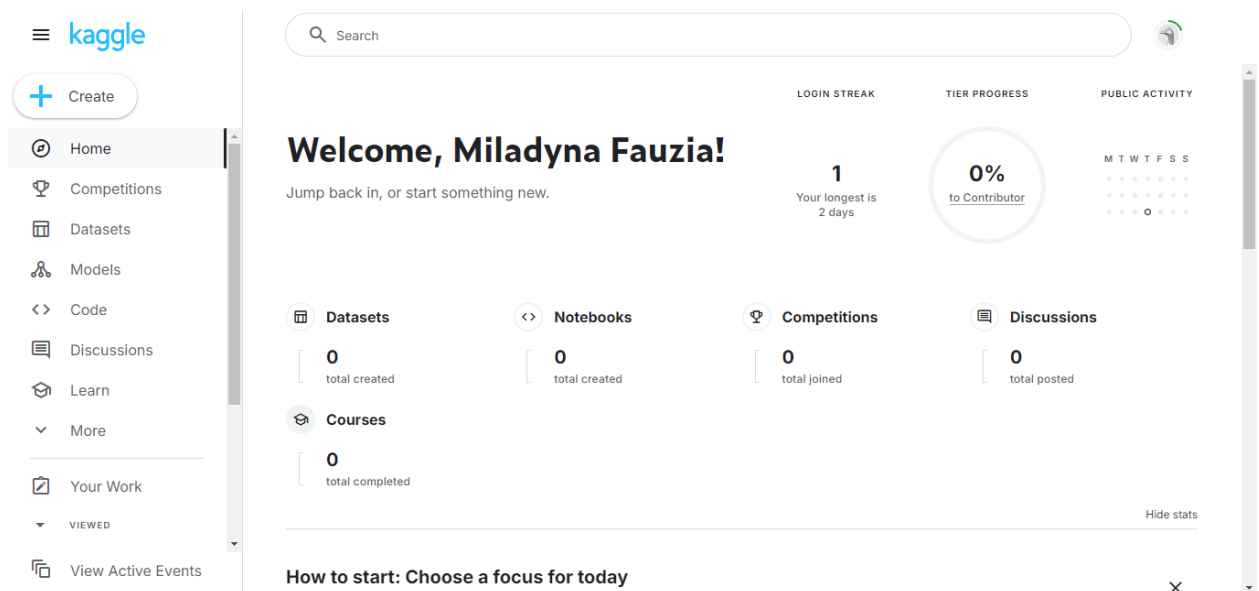


```
nama = "Miladyna Fauzia"
npm = "41155050210023"

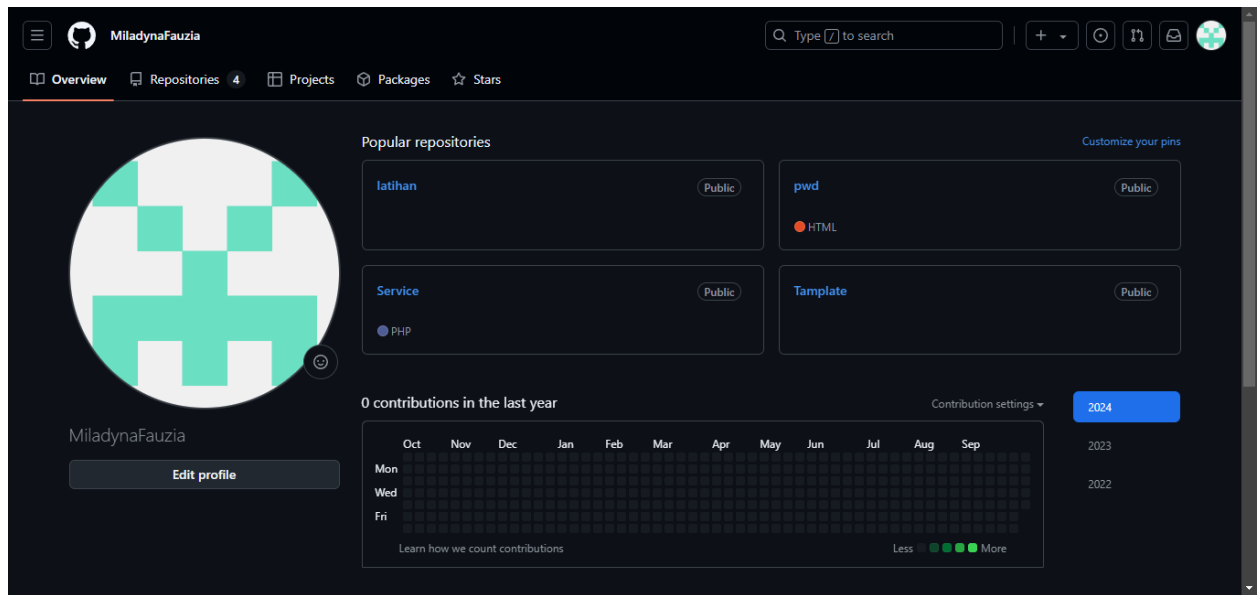
print(f>Nama: {nama}")
print(f"NPM: {npm}")
```

Nama: Miladyna Fauzia  
NPM: 41155050210023

3. Buatlah akun di <https://www.kaggle.com/>.

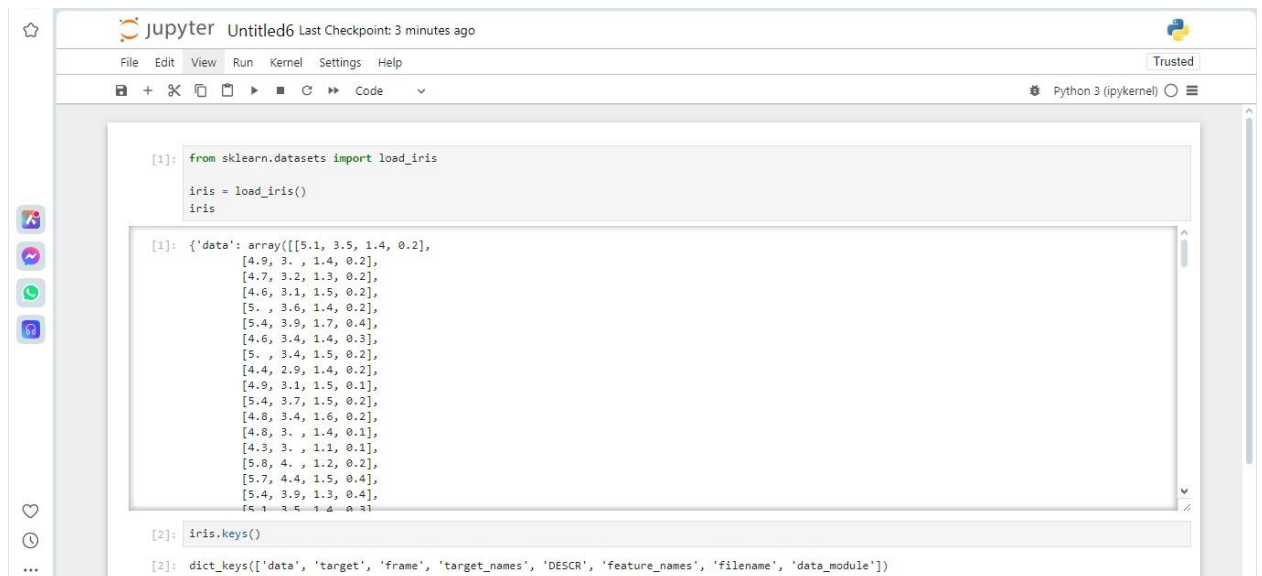


4. Buatlah akun di <https://github.com/> .

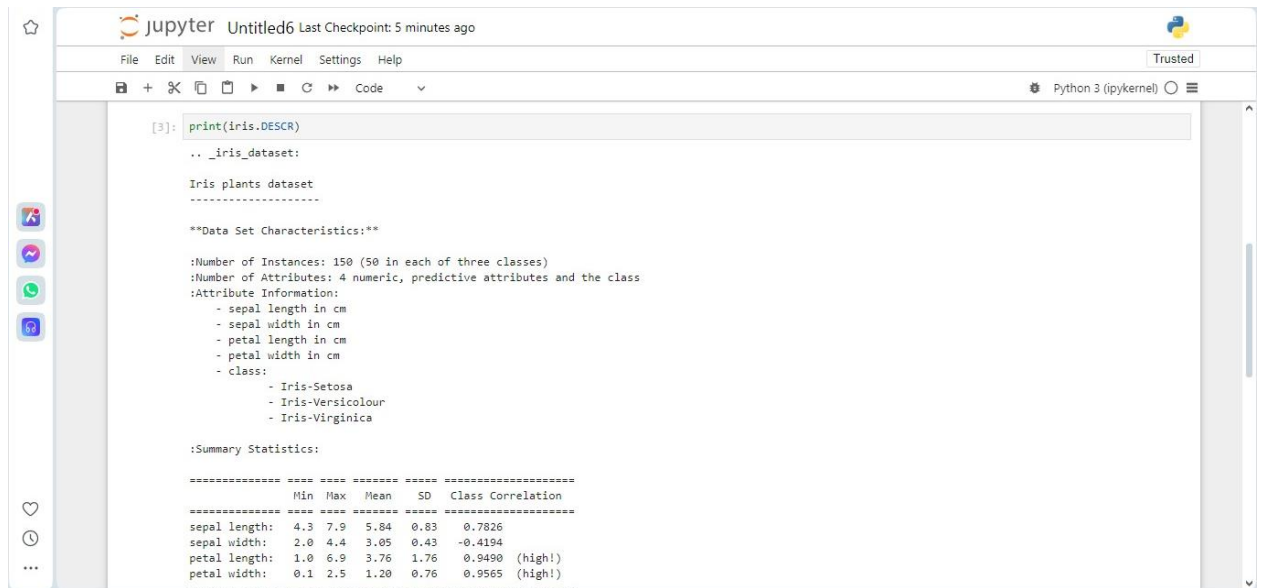


5. Lakukan praktek dari <https://youtu.be/mSO2hJln0OY?feature=shared> . Praktek tersebut yaitu:

#### 5.1. Load sample dataset



## 5.2. Metadata deskripsi dari sample dataset



```
[3]: print(iris.DESCR)

.. _iris_dataset:

Iris plants dataset
-----

**Data Set Characteristics:**

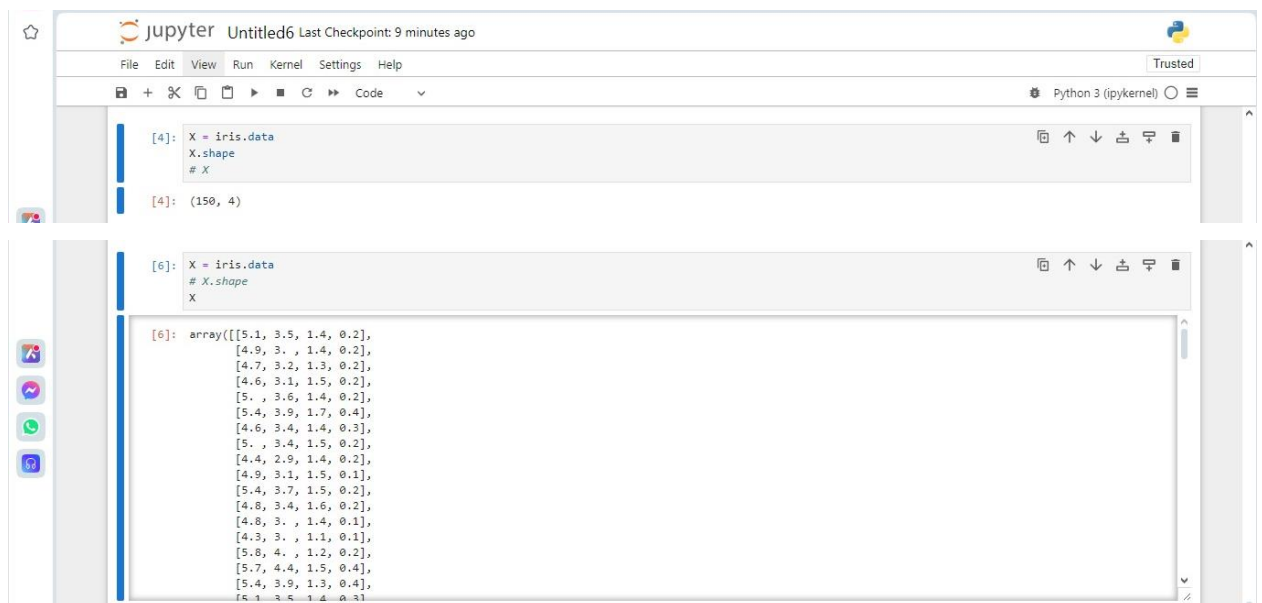
:Number of Instances: 150 (50 in each of three classes)
:Number of Attributes: 4 numeric, predictive attributes and the class
:Attribute Information:
 - sepal length in cm
 - sepal width in cm
 - petal length in cm
 - petal width in cm
 - class:
   - Iris-Setosa
   - Iris-Versicolour
   - Iris-Virginica

: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   3.05   0.43   -0.4194
petal length:  1.0   6.9   3.76   1.76    0.9490 (high!)
petal width:   0.1   2.5   1.20   0.76    0.9565 (high!)
=====
```

## 5.3. Explanatory & Response Variables | Features & Target

### Features



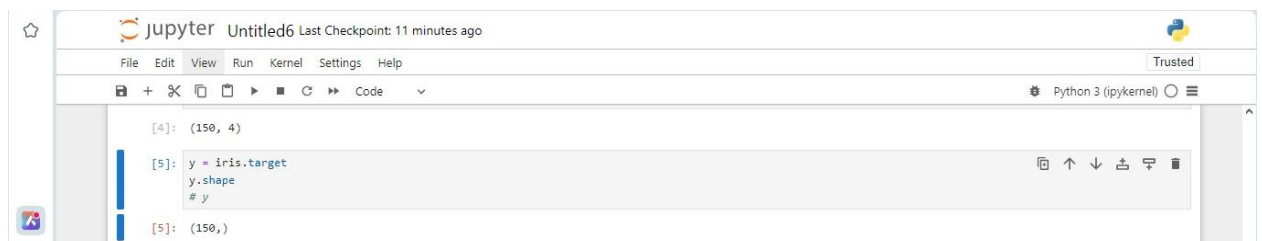
```
[4]: X = iris.data
     X.shape
     # X

[4]: (150, 4)

[6]: X = iris.data
     # X.shape
     X

[6]: 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, 0.3],
          [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. , 1.1, 0.1],
          [5.8, 4. , 1.2, 0.2],
          [5.7, 4.4, 1.5, 0.4],
          [5.4, 3.9, 1.3, 0.4],
          [5.1, 3.5, 1.4, 0.3]])
```

### Target



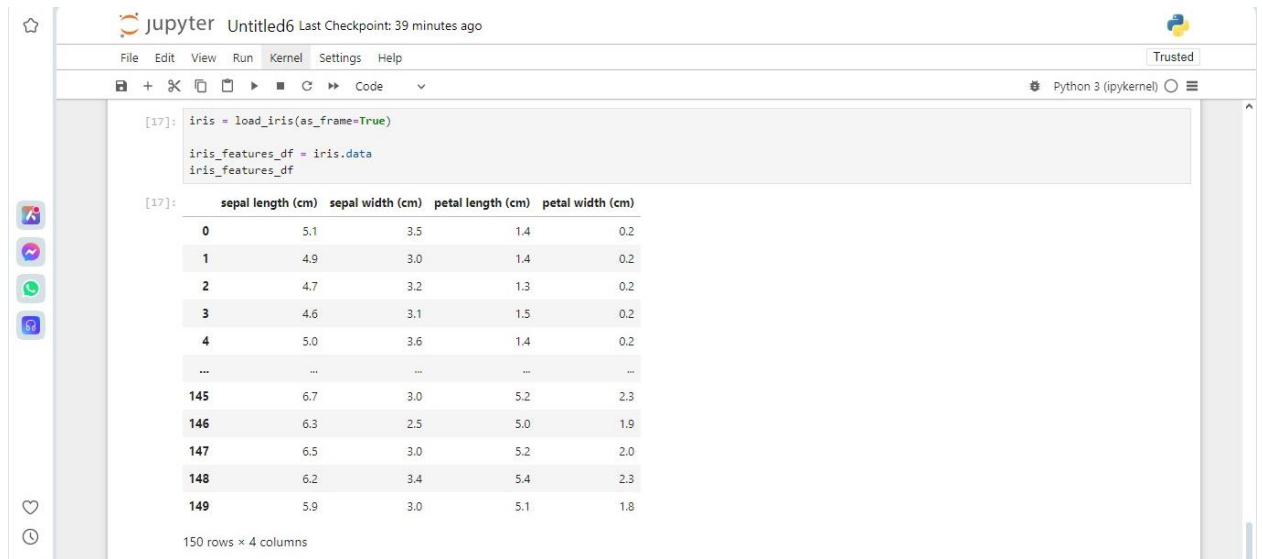
```
[4]: (150, 4)

[5]: y = iris.target
     y.shape
     # y

[5]: (150,)
```



## 5.7. Load sample dataset sebagai Pandas Data Frame



The screenshot shows a Jupyter Notebook interface with the following code and output:

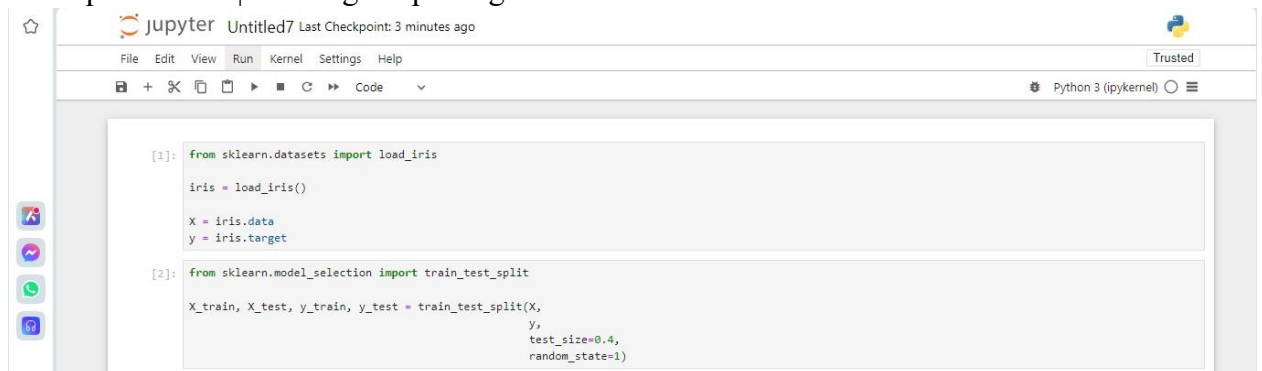
```
[17]: iris = load_iris(as_frame=True)
iris_features_df = iris.data
iris_features_df
```

	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. Lakukan praktek dari <https://youtu.be/tiREcHrtDLo?feature=shared> . Praktek tersebut yaitu:

### 6.1. Persiapan dataset | Loading & splitting dataset



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

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

[2]: 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




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

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

The output of the code is a `KNeighborsClassifier` object:

```
KNeighborsClassifier(n_neighbors=3)
```

### 6.3. Evaluasi model Machine Learning



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

```
[4]: from sklearn.metrics import accuracy_score

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

The output of the code is:

```
Accuracy: 0.9833333333333333
```

#### 6.4. Pemanfaatan trained model machine learning

```
[5]: data_baru = [[5, 5, 3, 2],
                 [2, 4, 3, 5]]

preds = model.predict(data_baru)
preds

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

[6]: 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

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

[7]: ['iris_classifier_knn.joblib']

[8]: production_model = joblib.load('iris_classifier_knn.joblib')
```

7. Lakukan praktek dari <https://youtu.be/smNnhEd26Ek?feature=shared> . Praktek tersebut yaitu:

##### 7.1. Persiapan sample dataset

```
Jupyter Untitled8 Last Checkpoint: 7 minutes ago

File Edit View Run Kernel Settings Help Trusted
Python 3 (ipykernel)

[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: binarization

```
Jupyter Untitled8 Last Checkpoint: 12 minutes ago

File Edit View Run Kernel Settings Help Trusted
Python 3 (ipykernel)

[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]])

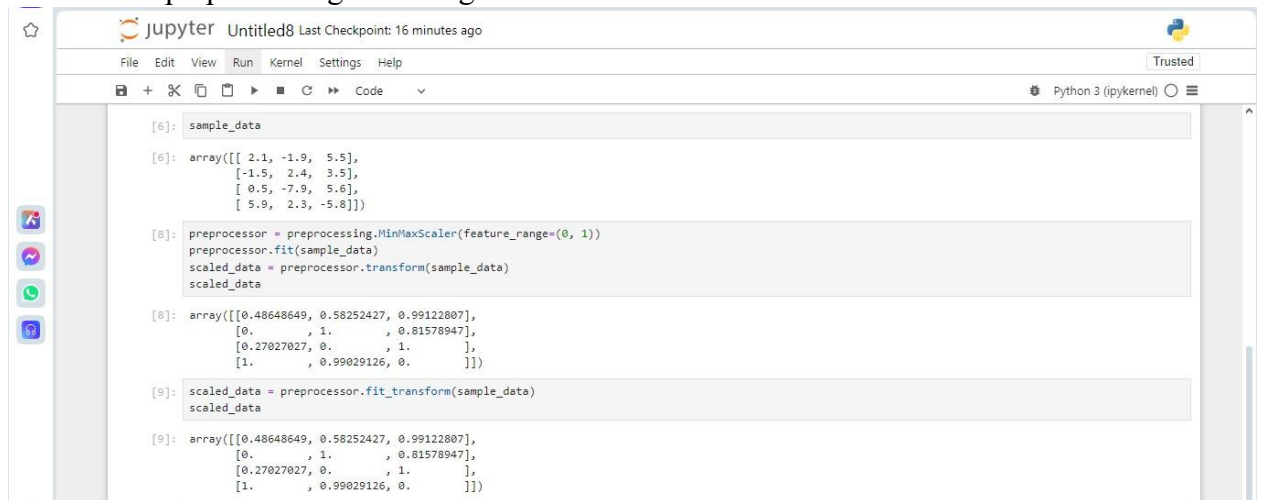
[5]: preprocessor = preprocessing.Binarizer(threshold=0.5)
     binarised_data = preprocessor.transform(sample_data)
     binarised_data

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

[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]])
```

### 7.3. Teknik data preprocessing 2: scaling



The image shows a Jupyter Notebook interface with the title 'Untitled8' and a last checkpoint of 16 minutes ago. The notebook is running Python 3 (ipykernel). The code in the notebook performs the following steps:

```
[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]])

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

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

[9]: 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: normalization



The image shows a Jupyter Notebook interface with the title 'Untitled8' and a last checkpoint of 22 minutes ago. The notebook is running Python 3 (ipykernel). The code in the notebook performs the following steps:

```
[10]: sample_data

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

[12]: array([[ 0.22105263, -0.2         ,  0.57894737],
          [-0.2027027 ,  0.32432432,  0.47297297],
          [ 0.03571429, -0.56428571,  0.4         ],
          [ 0.42142857,  0.16428571, -0.41428571]])
```