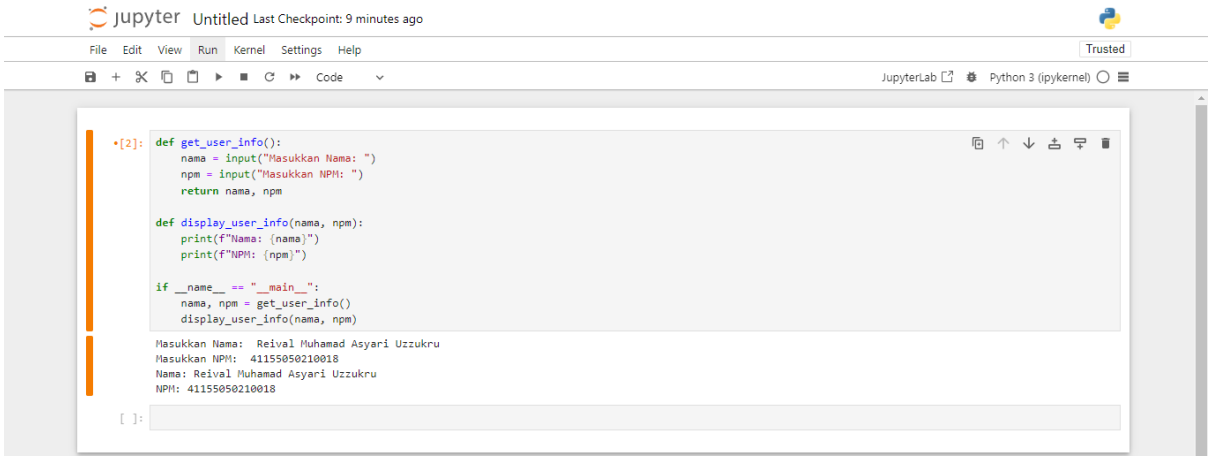
Nama : Reival Muhamad Asyari Uzzukru

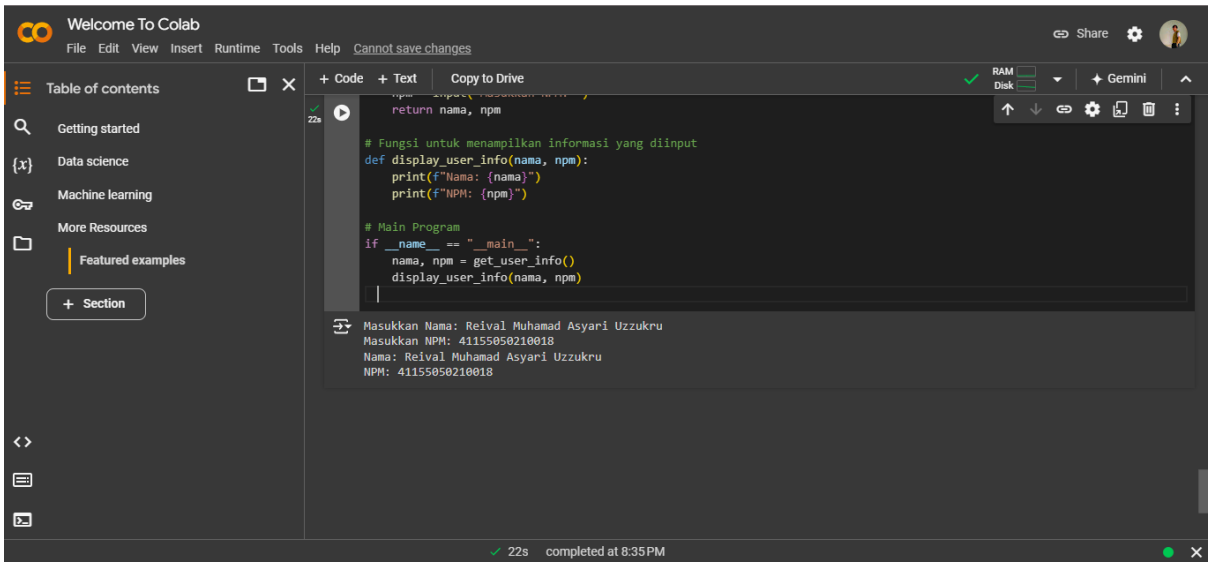
NPM : 41155050210018

Kelas : A1

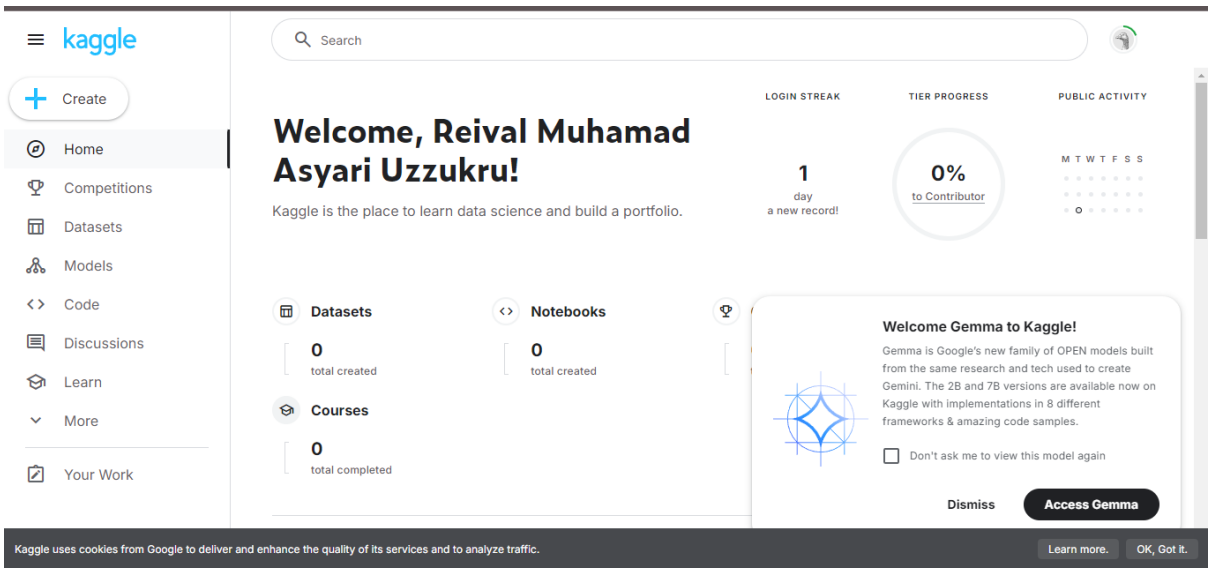
1. Instal Jupiter notebook dan library NumPy, SciPy, Pandas, Matplotlib, Seaborn, Scikit-learn. Serta membuat nama dan npm pada Jupiter notebook



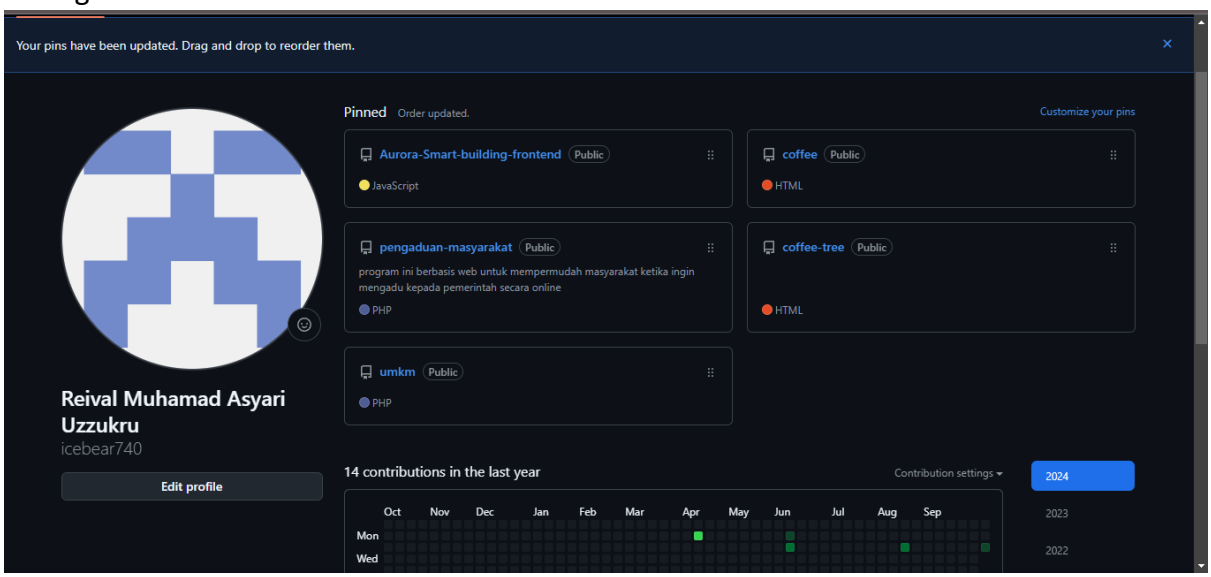
2. Menggunakan google collab



3. Membuat akun kagle



4. akun github



5. praktek

- load sample dataset

```
[1]: from sklearn.datasets import load_iris
     iris = load_iris()
     print(iris)
```

```
{'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, 0.3],
       [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],
```

```
[2]: iris.keys()
```

```
[2]: dict_keys(['data', 'target', 'frame', 'target_names', 'DESCR', 'feature_names', 'filename', 'data_module'])
```

```
[ ]:
```

- 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!)
```

- Explanatory & Response Variables | Features & Target

```
[4]: x = iris.data
     x.shape
```

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

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

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

```
[ ]:
```

- Feature & Target Names

[8]:

```
feature_names = iris.feature_names
feature_names
```

[8]:

```
['sepal length (cm)',
 'sepal width (cm)',
 'petal length (cm)',
 'petal width (cm)']
```

[9]:

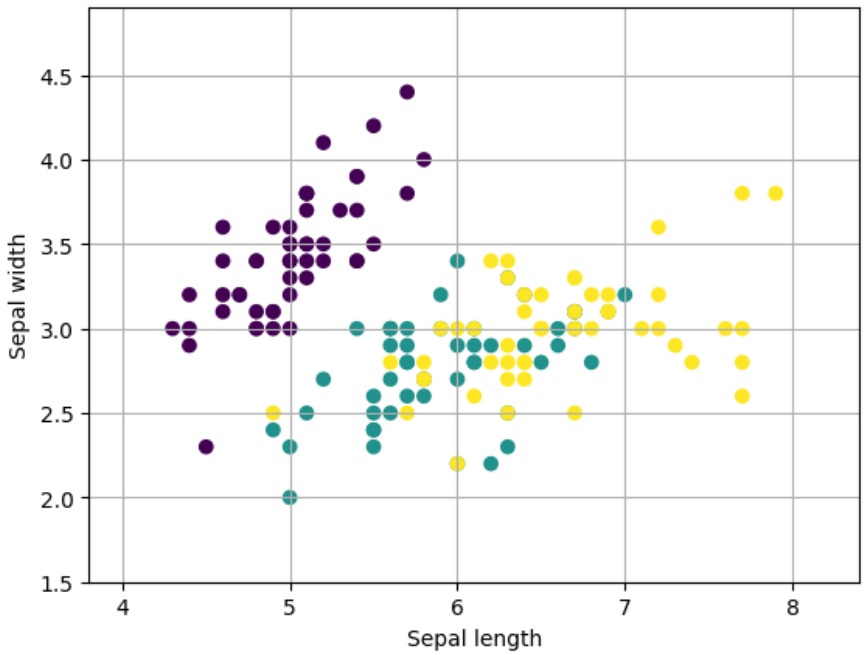
```
target_names = iris.target_names
target_names
```

[9]:

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

- Visualisasi Data

```
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()
```



- Training Set & Testing Set

[20]:
```
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: {X_train.shape} ')
print(f'y test: {X_test.shape} ')
```

```
X train: (105, 4)
X test: (45, 4)
y train: (105, 4)
y test: (45, 4)
```

- 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 × 4 columns

6. Praktek

- Persiapan dataset | Loading & splitting dataset

[5]:

```
from sklearn.datasets import load_iris
iris = load_iris()
X = iris.data
y = iris.target
```

[6]:

```
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

```
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
```

- Pemanfaatan trained model machine learning

[11]:

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

[11]:

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

[12]:

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

```
hasil prediksi: ['versicolor', 'virginica']
```

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

[13]:

```
import joblib
joblib.dump(model, 'iris_classifier_knn.joblib')
```

[13]:

```
['iris_classifier_knn.joblib']
```

[14]:

```
production_model = joblib.load('iris_classifier_knn.joblib')
```

[ ]:

7. Praktek
   - Persiapan sample dataset

[15]:

```
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
```

[15]:

```
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]])
```

[17]:

```
sample_data.shape
```

[17]:

```
(4, 3)
```

   - Teknik data preprocessing 1: binarization

```
sample_data
```

[18]:

```
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]])
```

[20]:

```
preprocessor = preprocessing.Binarizer(threshold=0.5)
binarised_data = preprocessor.transform(sample_data)
binarised_data
```

[20]:

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

- Teknik data preprocessing 2: scaling

[22]:

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

[22]:

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

[23]:

```
scaled_data = preprocessor.fit_transform(sample_data)
scaled_data
```

[23]:

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

- Teknik data preprocessing 3: normalisation

[24]:

```
l1_normalised_data = preprocessing.normalize(sample_data, norm='l1')
l1_normalised_data
```

[24]:

```
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]])
```

[25]:

```
l2_normalised_data = preprocessing.normalize(sample_data, norm='l2')
l2_normalised_data
```

[25]:

```
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 ]])
```