

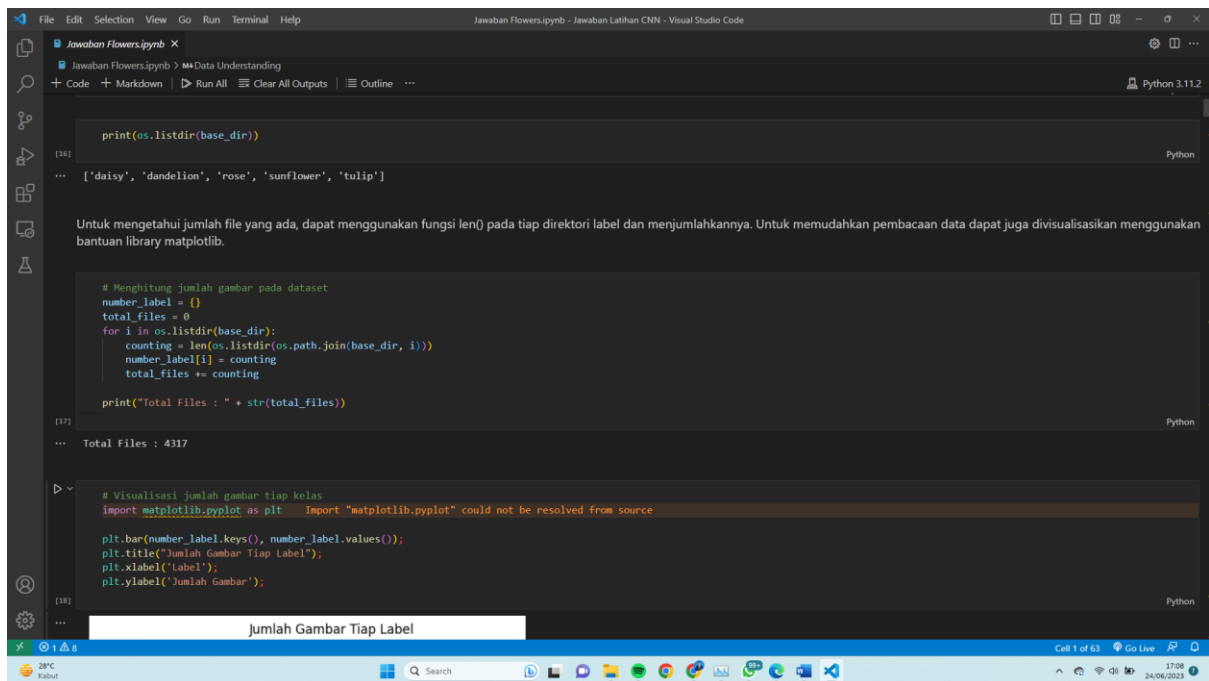
Nama : Ichsan Nur Rachmad Yusuf

NIM : 1207070056

Kelas : B – TKK

Metode Convolutional Neural Network (CNN) adalah salah satu algoritma dalam bidang pembelajaran mesin yang digunakan untuk memproses data berstruktur, terutama data gambar dan video. CNN secara khusus dirancang untuk mengenali pola dan fitur dalam data visual dengan memanfaatkan operasi konvolusi.

LATIHAN CNN



The screenshot shows a Visual Studio Code editor window titled 'Jawaban Flowers.ipynb - Jawaban Latihan CNN - Visual Studio Code'. The editor displays a Python script with the following code:

```
print(os.listdir(base_dir))

... ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']

Untuk mengetahui jumlah file yang ada, dapat menggunakan fungsi len() pada tiap direktori label dan menjumlahkannya. Untuk memudahkan pembacaan data dapat juga divisualisasikan menggunakan bantuan library matplotlib.

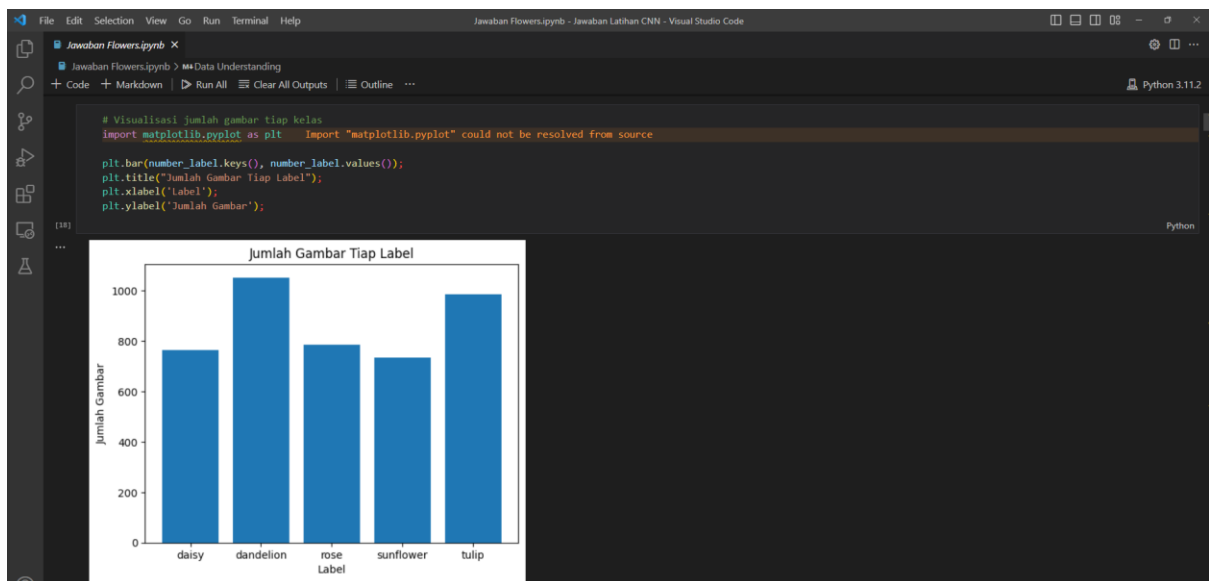
# Menghitung jumlah gambar pada dataset
number_label = {}
total_files = 0
for i in os.listdir(base_dir):
    counting = len(os.listdir(os.path.join(base_dir, i)))
    number_label[i] = counting
    total_files += counting

print("Total Files : " + str(total_files))

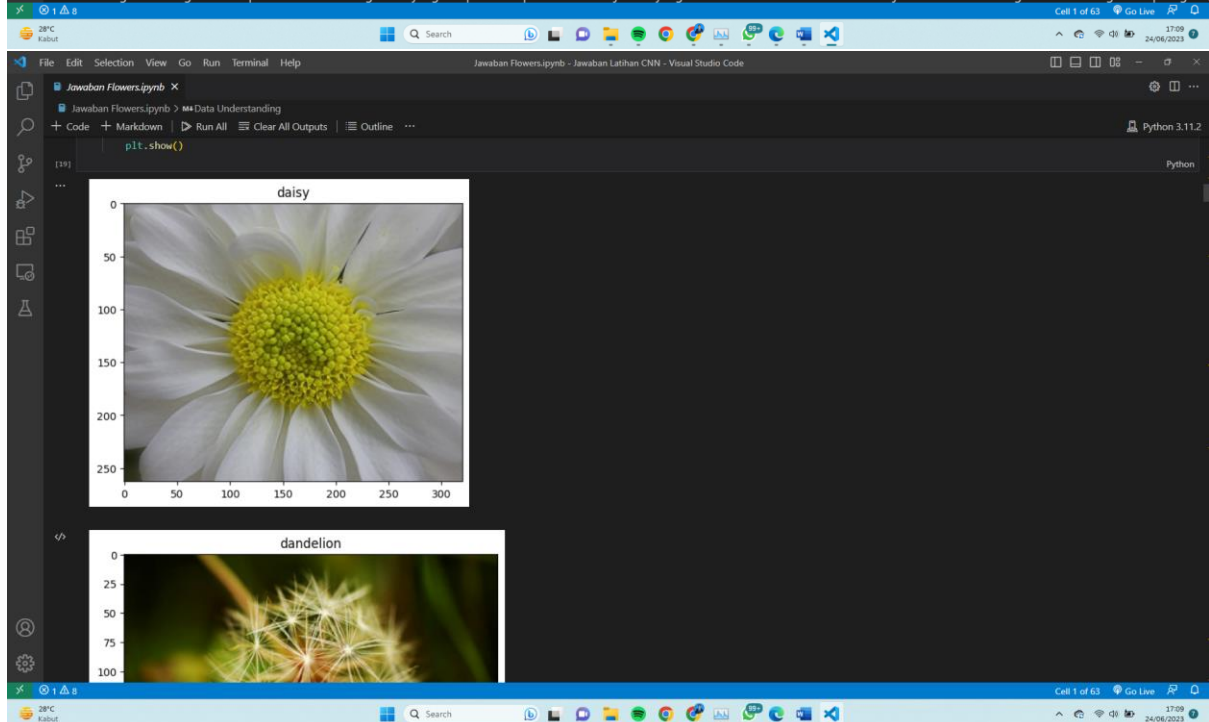
... Total Files : 4317

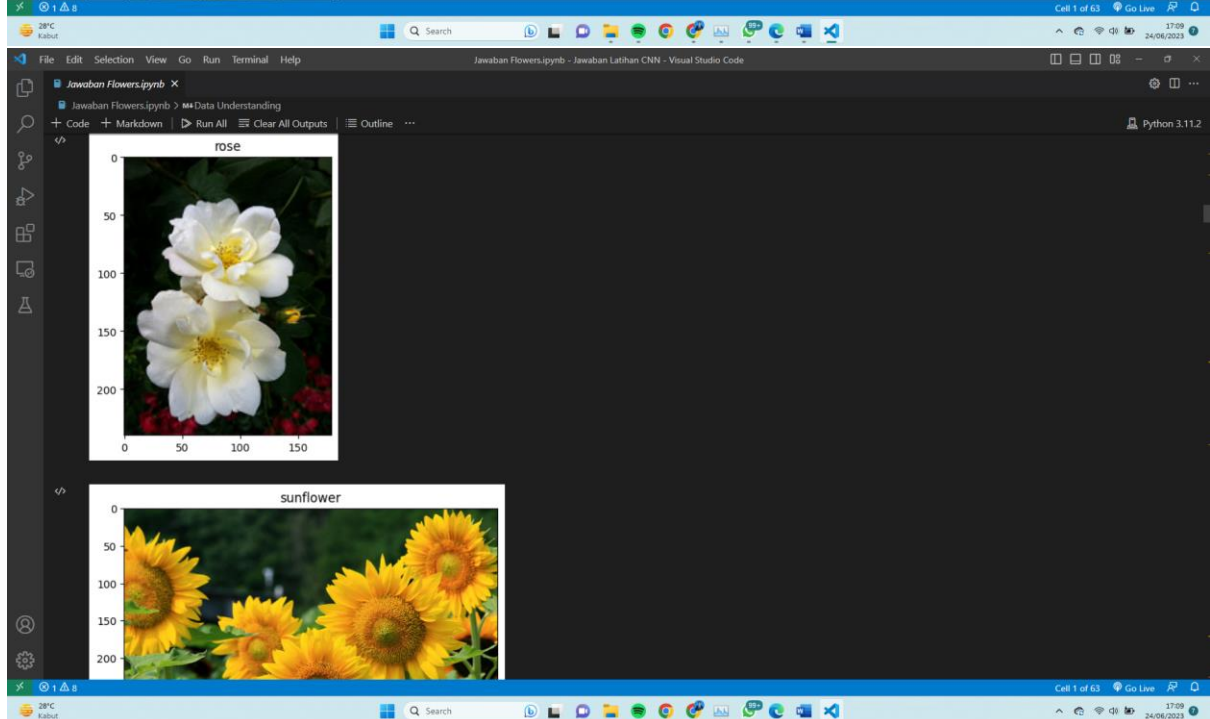
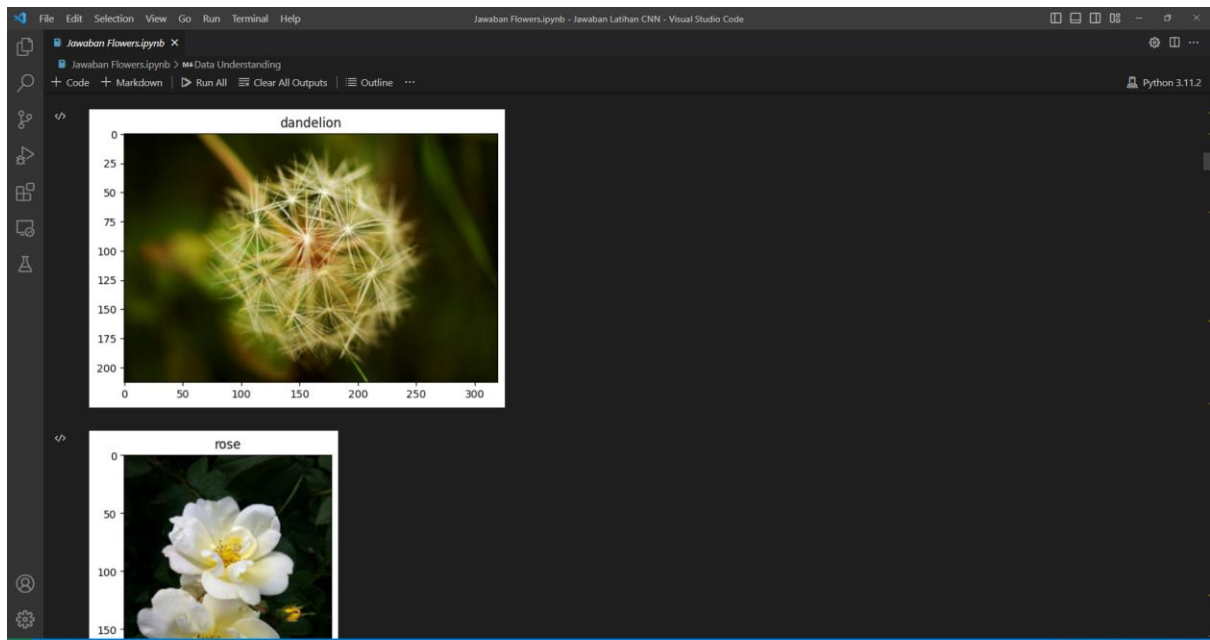
# Visualisasi jumlah gambar tiap kelas
import matplotlib.pyplot as plt
plt.bar(number_label.keys(), number_label.values())
plt.title("Jumlah Gambar Tiap Label")
plt.xlabel('Label')
plt.ylabel('Jumlah Gambar')
```

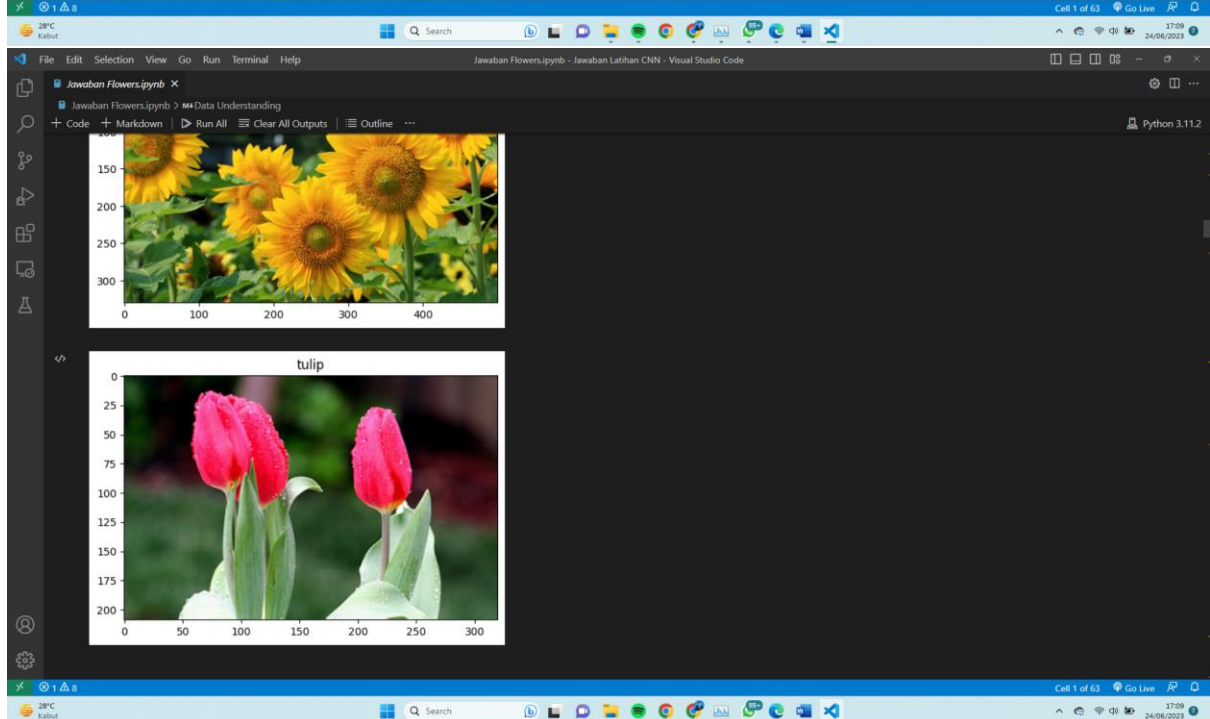
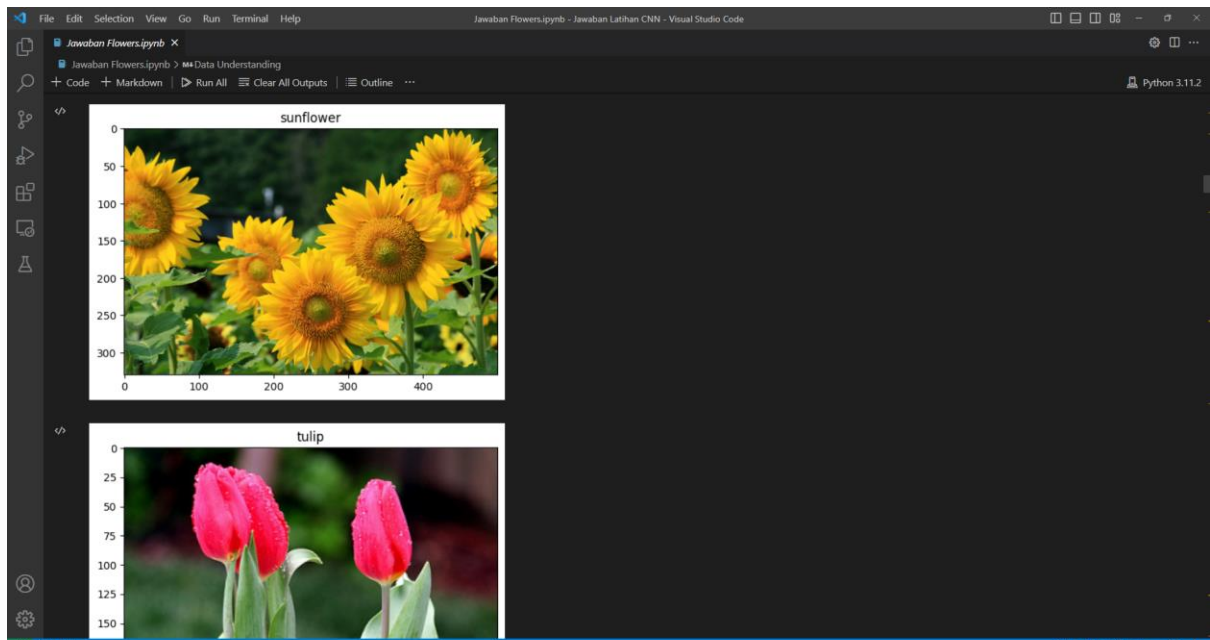
The output of the script shows the list of directories: ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip'] and the total number of files: 4317. The visualization part of the script is commented out, but the title 'Jumlah Gambar Tiap Label' is visible in the output area.

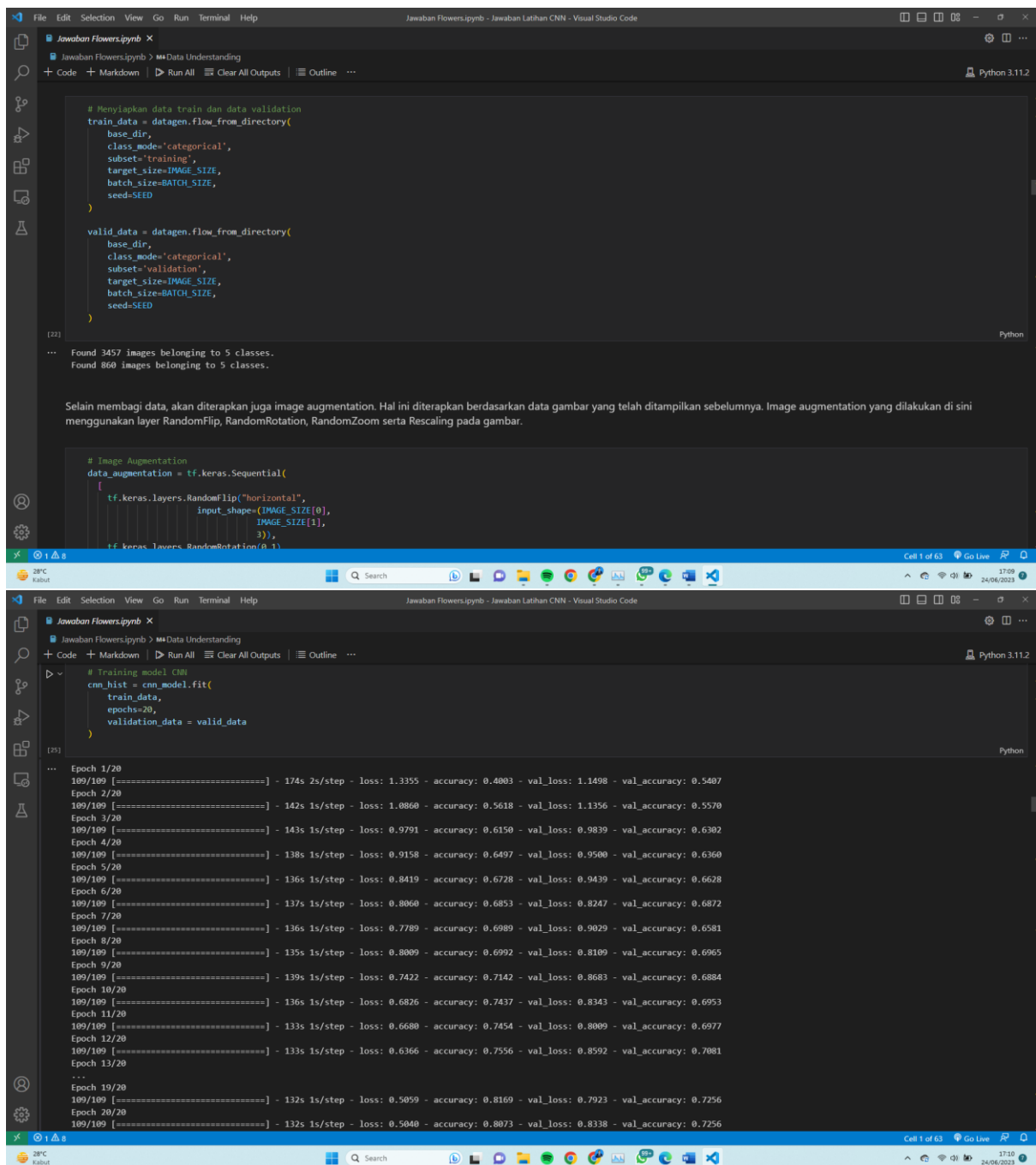


Dari diagram batang di atas dapat diketahui bahwa gambar yang ada pada setiap label memiliki jumlah yang berbeda-beda. Semua label memiliki jumlah lebih dari 700 gambar. Jumlah gambar paling









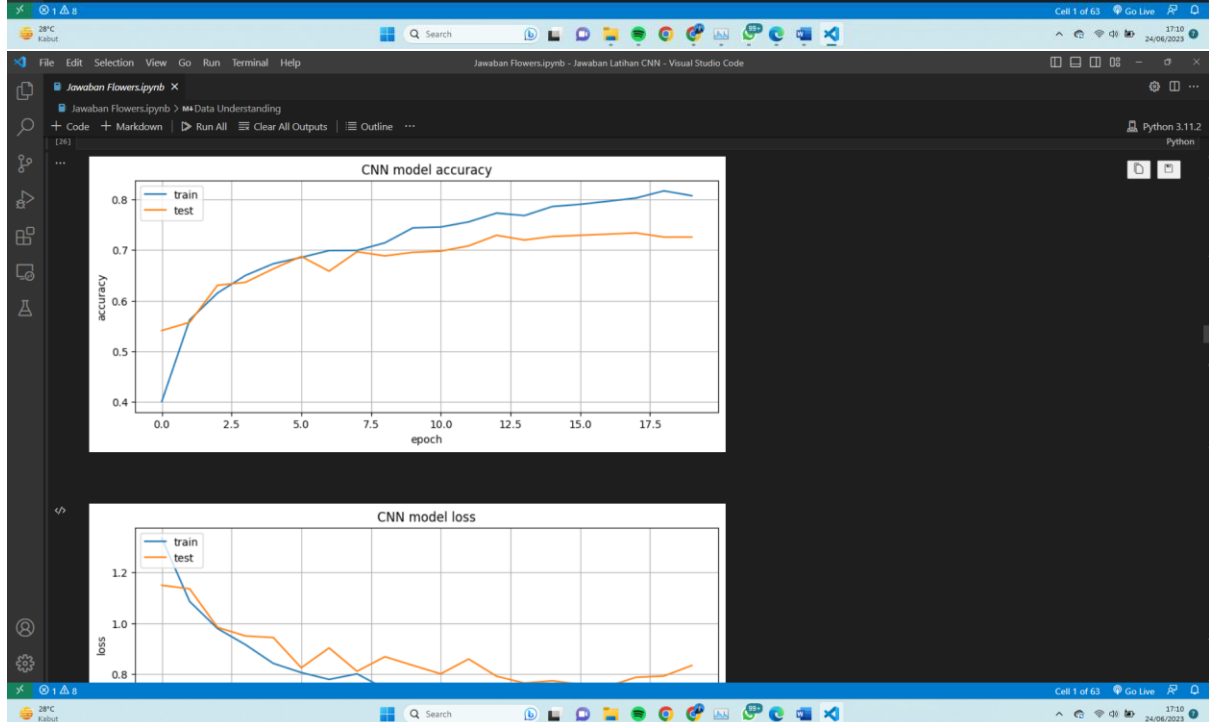
```
File Edit Selection View Go Run Terminal Help
Jawaban Flowers.ipynb - Jawaban Latihan CNN - Visual Studio Code

Jawaban Flowers.ipynb x
Jawaban Flowers.ipynb > MA Data Understanding
+ Code + Markdown ▶ Run All Clear All Outputs Outline ... Python 3.11.2

train_data,
epochs=20,
validation_data = valid_data

[25]
Python

... Epoch 1/20
109/109 [=====] - 174s 2s/step - loss: 1.3355 - accuracy: 0.4003 - val_loss: 1.1498 - val_accuracy: 0.5407
Epoch 2/20
109/109 [=====] - 142s 1s/step - loss: 1.0860 - accuracy: 0.5618 - val_loss: 1.1356 - val_accuracy: 0.5570
Epoch 3/20
109/109 [=====] - 143s 1s/step - loss: 0.9791 - accuracy: 0.6150 - val_loss: 0.9839 - val_accuracy: 0.6302
Epoch 4/20
109/109 [=====] - 138s 1s/step - loss: 0.9158 - accuracy: 0.6497 - val_loss: 0.9500 - val_accuracy: 0.6360
Epoch 5/20
109/109 [=====] - 136s 1s/step - loss: 0.8419 - accuracy: 0.6728 - val_loss: 0.9439 - val_accuracy: 0.6628
Epoch 6/20
109/109 [=====] - 137s 1s/step - loss: 0.8060 - accuracy: 0.6853 - val_loss: 0.8247 - val_accuracy: 0.6872
Epoch 7/20
109/109 [=====] - 136s 1s/step - loss: 0.7789 - accuracy: 0.6989 - val_loss: 0.9029 - val_accuracy: 0.6581
Epoch 8/20
109/109 [=====] - 135s 1s/step - loss: 0.8009 - accuracy: 0.6992 - val_loss: 0.8109 - val_accuracy: 0.6965
Epoch 9/20
109/109 [=====] - 139s 1s/step - loss: 0.7422 - accuracy: 0.7142 - val_loss: 0.8683 - val_accuracy: 0.6884
Epoch 10/20
109/109 [=====] - 136s 1s/step - loss: 0.6826 - accuracy: 0.7437 - val_loss: 0.8343 - val_accuracy: 0.6953
Epoch 11/20
109/109 [=====] - 133s 1s/step - loss: 0.6680 - accuracy: 0.7454 - val_loss: 0.8009 - val_accuracy: 0.6977
Epoch 12/20
109/109 [=====] - 133s 1s/step - loss: 0.6366 - accuracy: 0.7556 - val_loss: 0.8592 - val_accuracy: 0.7081
Epoch 13/20
...
Epoch 19/20
109/109 [=====] - 132s 1s/step - loss: 0.5059 - accuracy: 0.8169 - val_loss: 0.7923 - val_accuracy: 0.7256
Epoch 20/20
109/109 [=====] - 132s 1s/step - loss: 0.5040 - accuracy: 0.8073 - val_loss: 0.8338 - val_accuracy: 0.7256
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.
```



Visual Studio Code interface showing a Jupyter Notebook titled "Jawaban Flowers.ipynb" with two plots and Python code for transfer learning using VGG16.

The top plot shows accuracy vs epoch, and the bottom plot shows CNN model loss vs epoch.

The code in the notebook includes:

```
import tensorflow as tf
from tensorflow.keras.applications.vgg16 import VGG16

# Loading VGG16 model
base_vgg_model = VGG16(weights="imagenet", include_top=False, input_shape=(IMAGE_SIZE[0], IMAGE_SIZE[1], 3))
base_vgg_model.trainable = False

# Preprocessing Input
vgg_preprocess = tf.keras.applications.vgg16.preprocess_input
train_data.preprocessing_function = vgg_preprocess

# Transfer learning dengan VGG16
vgg_model = tf.keras.models.Sequential([
    data_augmentation,
    base_vgg_model,
    tf.keras.layers.Dropout(0.7),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(5, activation='softmax')
])

# Compiling model
vgg_model.compile(
```

The bottom plot shows the following data (approximate values):

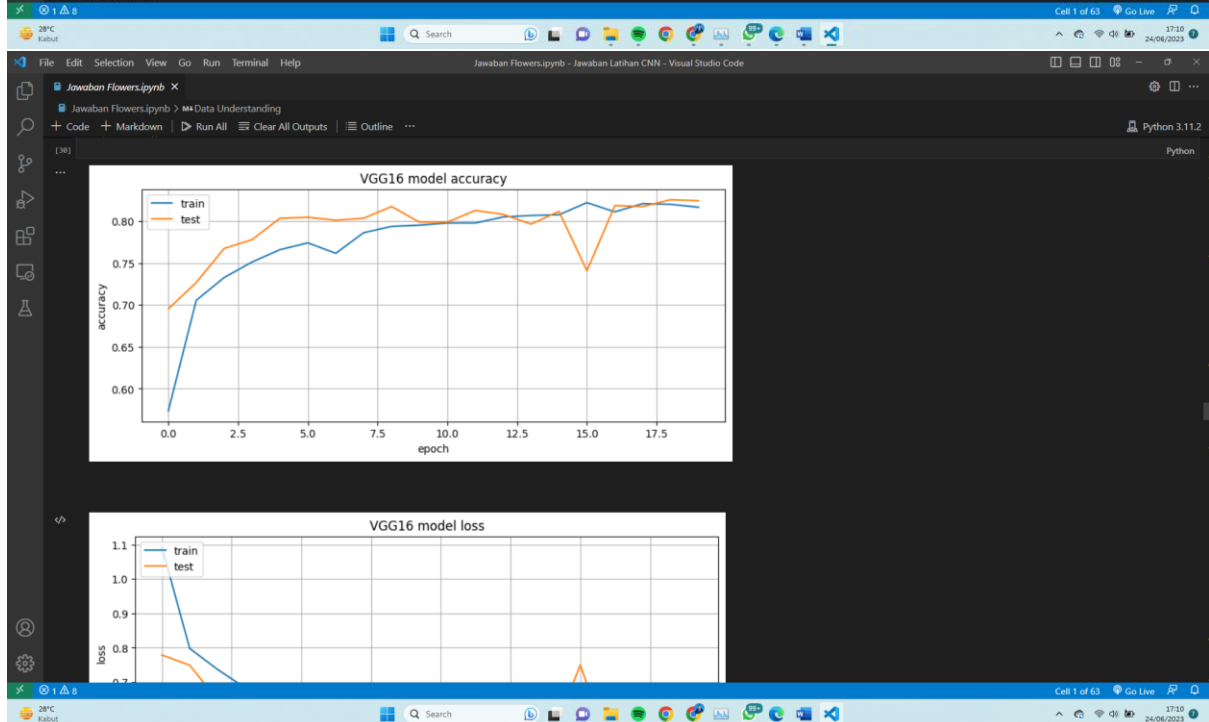
epoch	train loss	test loss
0.0	1.2	1.1
2.5	0.9	0.95
5.0	0.8	0.85
7.5	0.75	0.85
10.0	0.7	0.8
12.5	0.65	0.78
15.0	0.6	0.78
17.5	0.55	0.8

```
File Edit Selection View Go Run Terminal Help
Jawaban Flowers.ipynb - Jawaban Latihan CNN - Visual Studio Code

Jawaban Flowers.ipynb x
Jawaban Flowers.ipynb > Data Understanding
+ Code + Markdown ▶ Run All Clear All Outputs Outline ... Python 3.11.2

[29]
...
Epoch 1/20
109/109 [=====] - 347s 3s/step - loss: 1.0934 - accuracy: 0.5736 - val_loss: 0.7790 - val_accuracy: 0.6953
Epoch 2/20
109/109 [=====] - 330s 3s/step - loss: 0.7978 - accuracy: 0.7055 - val_loss: 0.7491 - val_accuracy: 0.7267
Epoch 3/20
109/109 [=====] - 329s 3s/step - loss: 0.7366 - accuracy: 0.7327 - val_loss: 0.6477 - val_accuracy: 0.7674
Epoch 4/20
109/109 [=====] - 326s 3s/step - loss: 0.6825 - accuracy: 0.7512 - val_loss: 0.6384 - val_accuracy: 0.7779
Epoch 5/20
109/109 [=====] - 324s 3s/step - loss: 0.6407 - accuracy: 0.7660 - val_loss: 0.5443 - val_accuracy: 0.8035
Epoch 6/20
109/109 [=====] - 325s 3s/step - loss: 0.6222 - accuracy: 0.7741 - val_loss: 0.5496 - val_accuracy: 0.8047
Epoch 7/20
109/109 [=====] - 325s 3s/step - loss: 0.6311 - accuracy: 0.7616 - val_loss: 0.5486 - val_accuracy: 0.8012
Epoch 8/20
109/109 [=====] - 324s 3s/step - loss: 0.5725 - accuracy: 0.7862 - val_loss: 0.5428 - val_accuracy: 0.8035
Epoch 9/20
109/109 [=====] - 383s 4s/step - loss: 0.5391 - accuracy: 0.7938 - val_loss: 0.5341 - val_accuracy: 0.8174
Epoch 10/20
109/109 [=====] - 1099s 10s/step - loss: 0.5496 - accuracy: 0.7952 - val_loss: 0.5612 - val_accuracy: 0.7988
Epoch 11/20
109/109 [=====] - 816s 7s/step - loss: 0.5386 - accuracy: 0.7978 - val_loss: 0.5337 - val_accuracy: 0.7988
Epoch 12/20
109/109 [=====] - 636s 6s/step - loss: 0.5344 - accuracy: 0.7978 - val_loss: 0.5194 - val_accuracy: 0.8128
Epoch 13/20
...
Epoch 19/20
109/109 [=====] - 486s 4s/step - loss: 0.4791 - accuracy: 0.8201 - val_loss: 0.4841 - val_accuracy: 0.8256
Epoch 20/20
109/109 [=====] - 593s 5s/step - loss: 0.4959 - accuracy: 0.8166 - val_loss: 0.5047 - val_accuracy: 0.8244
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.

Evaluasi Model
```



Visual Studio Code interface showing a Jupyter Notebook titled "Jawaban Flowers.ipynb" with two plots and Python code.

Plot 1: Accuracy vs Epoch

epoch	accuracy
0.0	0.58
1.0	0.70
2.0	0.70
3.0	0.70
4.0	0.70
5.0	0.70
6.0	0.70
7.0	0.70
8.0	0.70
9.0	0.70
10.0	0.70
11.0	0.70
12.0	0.70
13.0	0.70
14.0	0.70
15.0	0.70
16.0	0.70
17.0	0.70

Plot 2: VGG16 model loss vs Epoch

epoch	train loss	test loss
0.0	1.0	0.8
1.0	0.8	0.75
2.0	0.7	0.65
3.0	0.65	0.6
4.0	0.6	0.55
5.0	0.55	0.55
6.0	0.55	0.55
7.0	0.55	0.55
8.0	0.55	0.55
9.0	0.55	0.55
10.0	0.55	0.55
11.0	0.55	0.55
12.0	0.55	0.55
13.0	0.55	0.55
14.0	0.55	0.55
15.0	0.55	0.75
16.0	0.55	0.55
17.0	0.55	0.55

Python Code:

```
# Loading ResNet50 model
base_resnet_model = ResNet50(include_top=False,
                              input_shape=(IMAGE_SIZE[0], IMAGE_SIZE[1], 3),
                              pooling='max', classes=5,
                              weights='imagenet')

base_resnet_model.trainable = False

train_data.preprocessing_function = tf.keras.applications.resnet50.preprocess_input

# Transfer learning ResNet50
resnet_model = tf.keras.models.Sequential([
    data_augmentation,
    base_resnet_model,
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(5, activation='softmax')
])

# Compiling model
resnet_model.compile(
    loss='categorical_crossentropy',
    optimizer=tf.keras.optimizers.Adam(),
    metrics=['accuracy']
)
```

Download data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5
94765736/94765736 [=====] - 32s @us/step

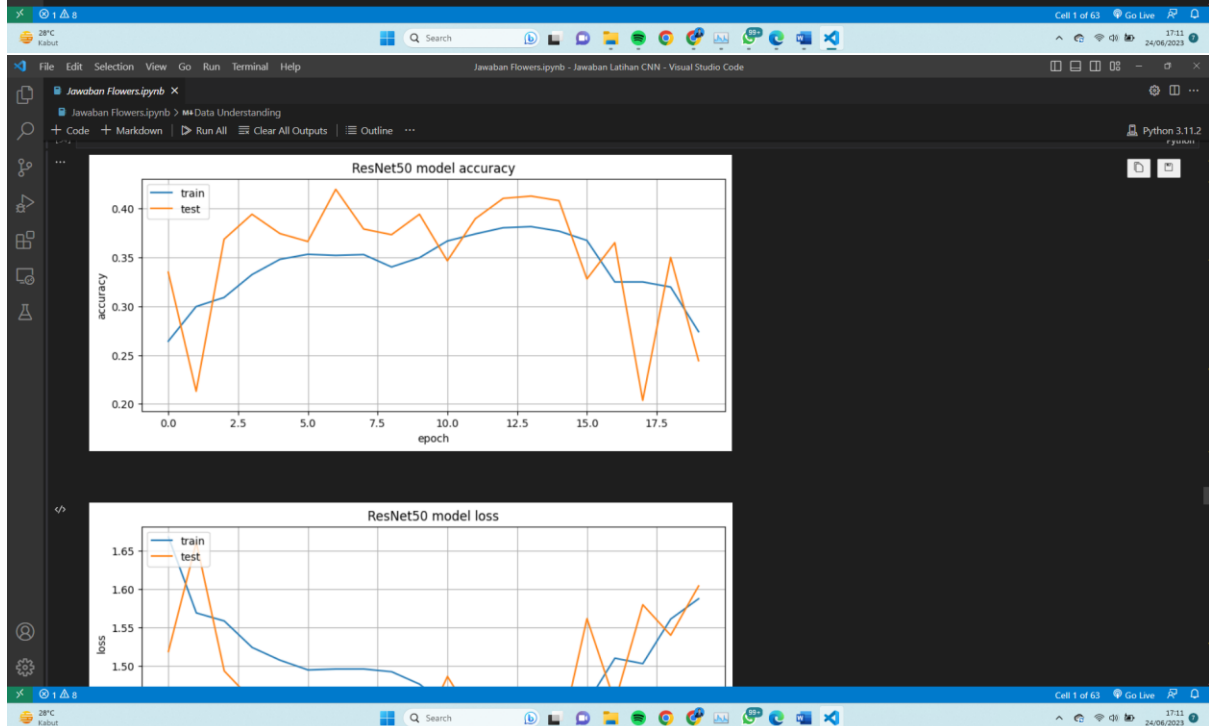
Cell 1 of 63 | Go Live | Python 3.11.2

```
File Edit Selection View Go Run Terminal Help
Jawaban Flowers.ipynb - Jawaban Latihan CNN - Visual Studio Code

Jawaban Flowers.ipynb x
Jawaban Flowers.ipynb > MA Data Understanding
+ Code + Markdown + Run All + Clear All Outputs + Outline ... Python 3.11.2

[33]
...
Epoch 1/20
109/109 [=====] - 288s 3s/step - loss: 1.6675 - accuracy: 0.2641 - val_loss: 1.5189 - val_accuracy: 0.3349
Epoch 2/20
109/109 [=====] - 321s 3s/step - loss: 1.5689 - accuracy: 0.2997 - val_loss: 1.6607 - val_accuracy: 0.2128
Epoch 3/20
109/109 [=====] - 297s 3s/step - loss: 1.5585 - accuracy: 0.3089 - val_loss: 1.4936 - val_accuracy: 0.3686
Epoch 4/20
109/109 [=====] - 266s 2s/step - loss: 1.5241 - accuracy: 0.3324 - val_loss: 1.4557 - val_accuracy: 0.3942
Epoch 5/20
109/109 [=====] - 254s 2s/step - loss: 1.5075 - accuracy: 0.3480 - val_loss: 1.4708 - val_accuracy: 0.3744
Epoch 6/20
109/109 [=====] - 258s 2s/step - loss: 1.4949 - accuracy: 0.3532 - val_loss: 1.4640 - val_accuracy: 0.3663
Epoch 7/20
109/109 [=====] - 246s 2s/step - loss: 1.4961 - accuracy: 0.3520 - val_loss: 1.4153 - val_accuracy: 0.4198
Epoch 8/20
109/109 [=====] - 251s 2s/step - loss: 1.4961 - accuracy: 0.3529 - val_loss: 1.4655 - val_accuracy: 0.3791
Epoch 9/20
109/109 [=====] - 247s 2s/step - loss: 1.4926 - accuracy: 0.3402 - val_loss: 1.4582 - val_accuracy: 0.3733
Epoch 10/20
109/109 [=====] - 247s 2s/step - loss: 1.4705 - accuracy: 0.3497 - val_loss: 1.4097 - val_accuracy: 0.3942
Epoch 11/20
109/109 [=====] - 246s 2s/step - loss: 1.4476 - accuracy: 0.3668 - val_loss: 1.4864 - val_accuracy: 0.3465
Epoch 12/20
109/109 [=====] - 246s 2s/step - loss: 1.4649 - accuracy: 0.3740 - val_loss: 1.4275 - val_accuracy: 0.3895
Epoch 13/20
...
Epoch 19/20
109/109 [=====] - 246s 2s/step - loss: 1.5610 - accuracy: 0.3196 - val_loss: 1.5399 - val_accuracy: 0.3580
Epoch 20/20
109/109 [=====] - 248s 2s/step - loss: 1.5873 - accuracy: 0.2739 - val_loss: 1.6039 - val_accuracy: 0.2442
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.

Evaluasi Model
```



Visual Studio Code interface showing two plots and Python code for a CNN model.

Top Plot: Accuracy vs Epoch

epoch	acc (blue)	acc (orange)
0.0	0.27	0.27
1.0	0.29	0.22
2.0	0.30	0.28
3.0	0.30	0.28
4.0	0.30	0.28
5.0	0.30	0.28
6.0	0.30	0.28
7.0	0.30	0.28
8.0	0.30	0.28
9.0	0.30	0.28
10.0	0.30	0.28
11.0	0.30	0.28
12.0	0.30	0.28
13.0	0.30	0.28
14.0	0.30	0.28
15.0	0.30	0.28
16.0	0.30	0.28
17.0	0.30	0.28
17.5	0.28	0.25

Bottom Plot: ResNet50 model loss

epoch	train loss (blue)	test loss (orange)
0.0	1.55	1.55
1.0	1.52	1.52
2.0	1.50	1.50
3.0	1.48	1.48
4.0	1.47	1.47
5.0	1.47	1.47
6.0	1.47	1.47
7.0	1.47	1.47
8.0	1.47	1.47
9.0	1.47	1.47
10.0	1.47	1.47
11.0	1.47	1.47
12.0	1.47	1.47
13.0	1.47	1.47
14.0	1.47	1.47
15.0	1.47	1.47
16.0	1.47	1.47
17.0	1.47	1.47
17.5	1.47	1.47

Python Code:

```
Memuat Model DenseNet201

# Loading DenseNet201 model
base_densenet_model = tf.keras.applications.DenseNet201(include_top=False,
                                                         weights='imagenet',
                                                         input_shape=(IMAGE_SIZE[0], IMAGE_SIZE[1], 3),
                                                         pooling='max')
base_densenet_model.trainable=False
train_data.preprocessing_function = tf.keras.applications.densenet.preprocess_input

[35] Python

... Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet201_weights_tf_dim_ordering_tf_kernels_notop.h5
74836368/74836368 [=====] - 24s 0us/step

# Transfer learning DenseNet201
densenet_model = tf.keras.models.Sequential([
    data_augmentation,
    base_densenet_model,
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(5, activation='softmax')
])

# Compiling model
densenet_model.compile(
    loss='categorical_crossentropy',
    optimizer=tf.keras.optimizers.Adam(),
    metrics=['accuracy']
)

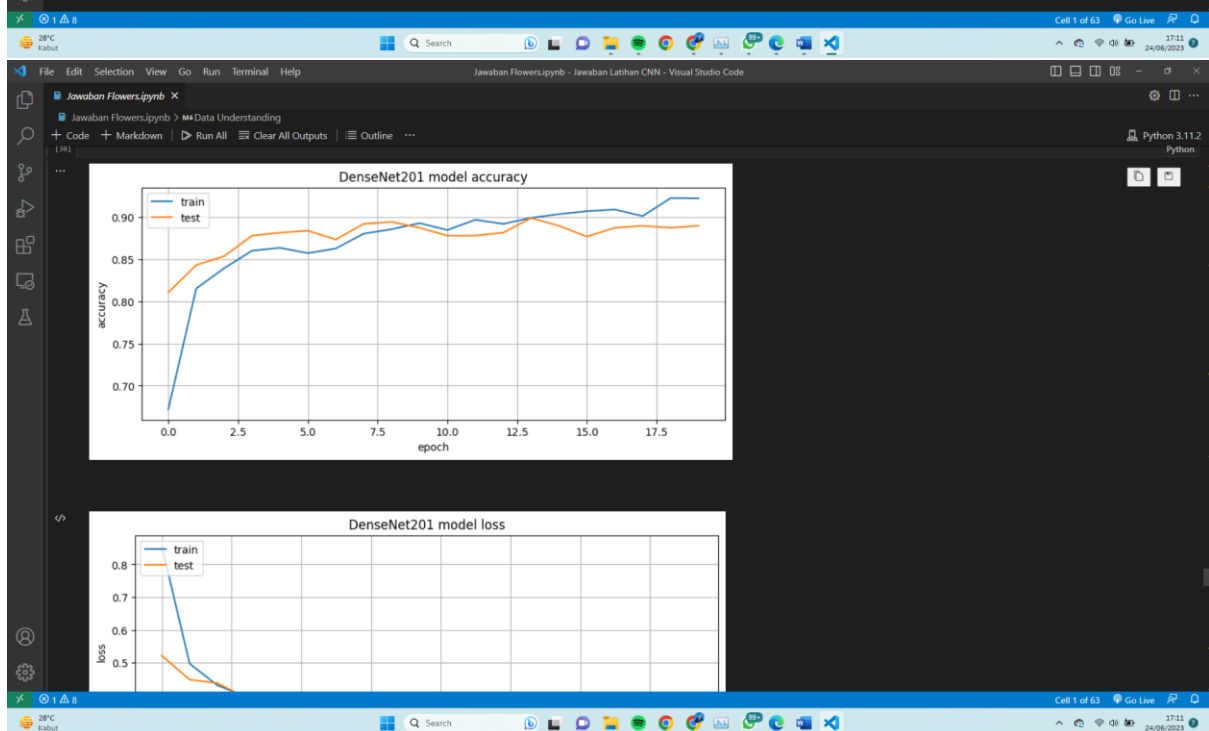
[36] Python
```

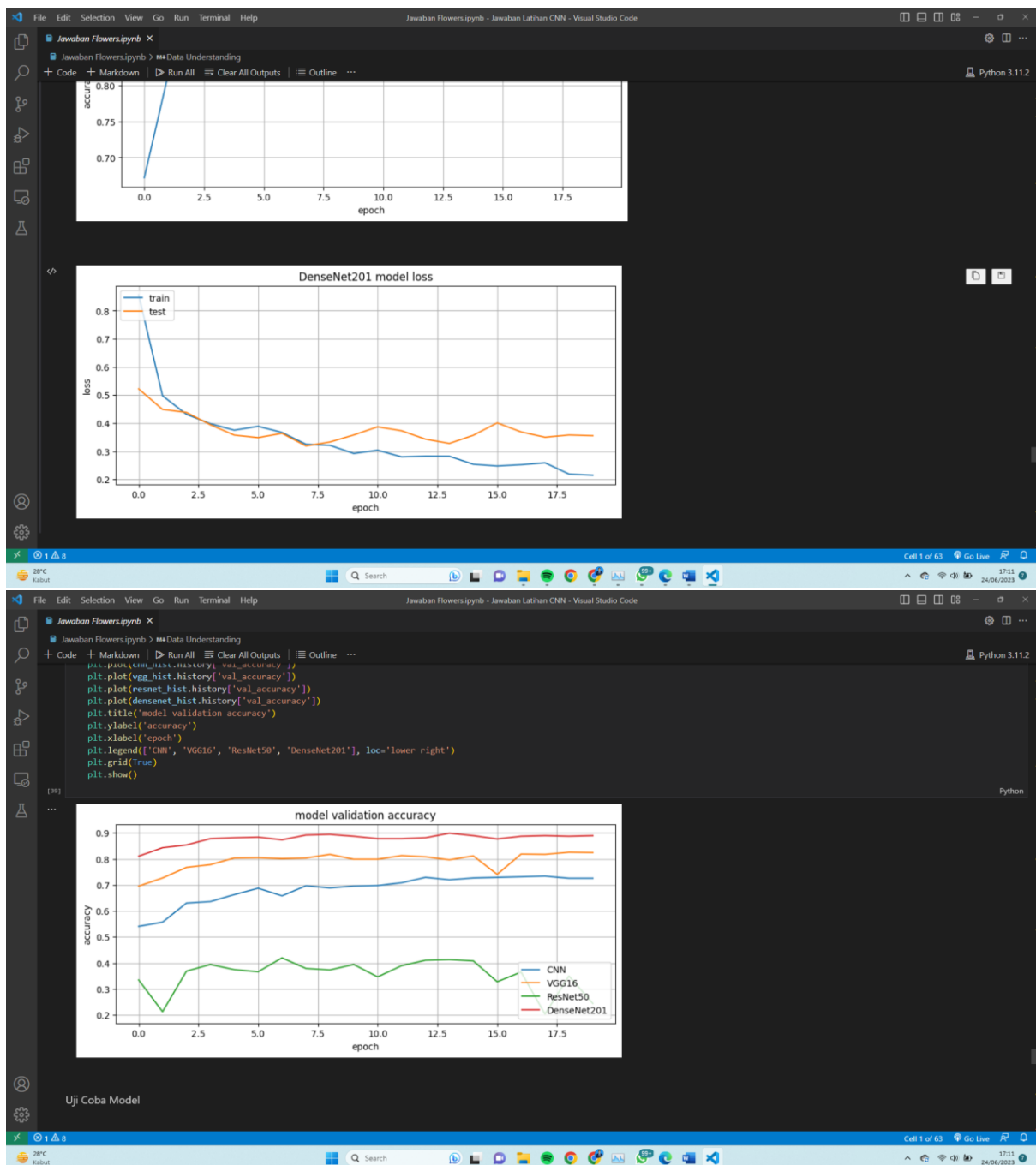
```
File Edit Selection View Go Run Terminal Help
Jawaban Flowers.ipynb - Jawaban Latihan CNN - Visual Studio Code

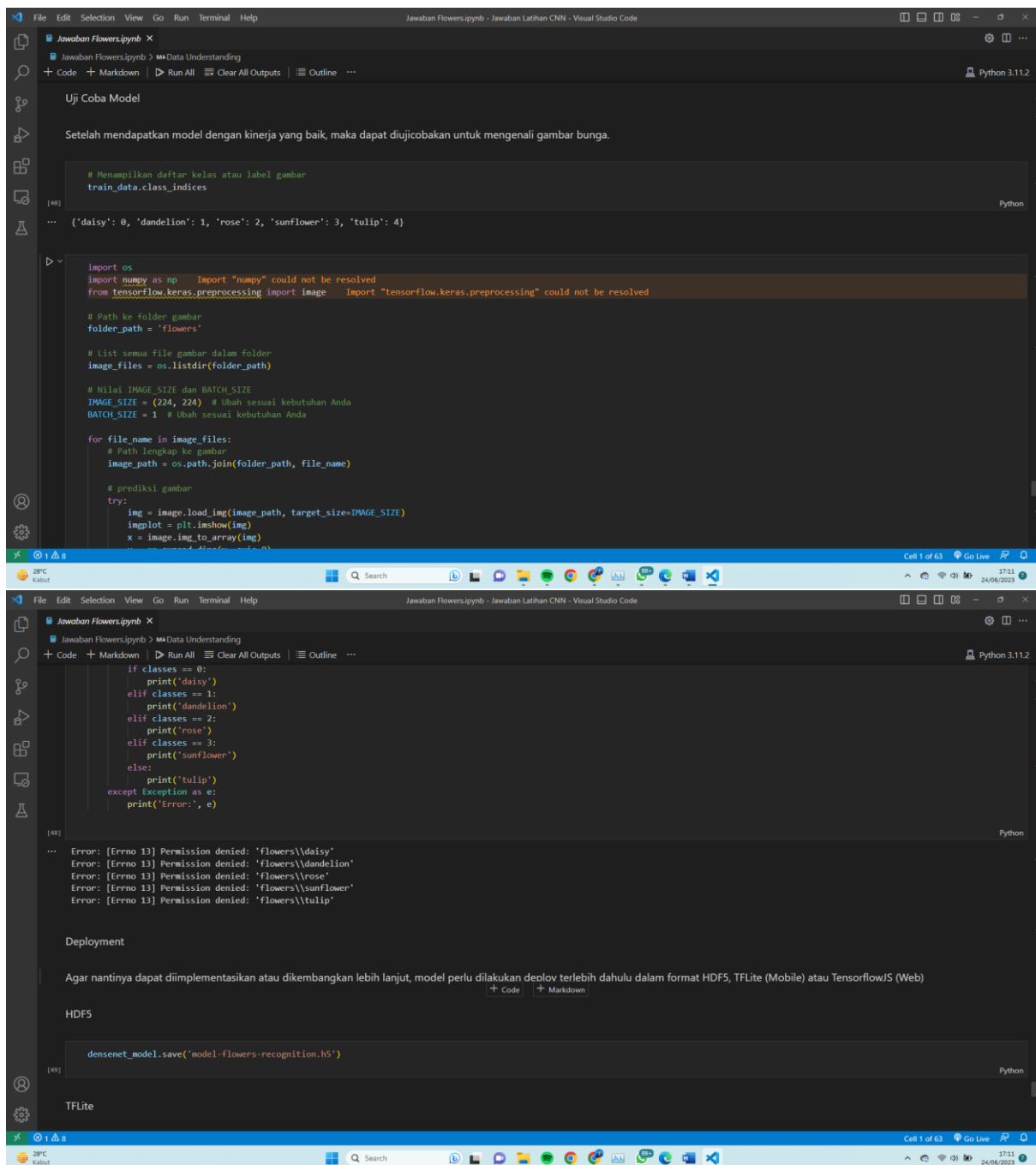
Jawaban Flowers.ipynb x
Jawaban Flowers.ipynb > MA Data Understanding
+ Code + Markdown ▶ Run All Clear All Outputs Outline ... Python 3.11.2

[37]
...
Epoch 1/20
109/109 [=====] - 472s 4s/step - loss: 0.8566 - accuracy: 0.6720 - val_loss: 0.5217 - val_accuracy: 0.8185
Epoch 2/20
109/109 [=====] - 438s 4s/step - loss: 0.4972 - accuracy: 0.8152 - val_loss: 0.4486 - val_accuracy: 0.8430
Epoch 3/20
109/109 [=====] - 443s 4s/step - loss: 0.4311 - accuracy: 0.8392 - val_loss: 0.4381 - val_accuracy: 0.8535
Epoch 4/20
109/109 [=====] - 440s 4s/step - loss: 0.3980 - accuracy: 0.8600 - val_loss: 0.3941 - val_accuracy: 0.8779
Epoch 5/20
109/109 [=====] - 431s 4s/step - loss: 0.3748 - accuracy: 0.8635 - val_loss: 0.3573 - val_accuracy: 0.8814
Epoch 6/20
109/109 [=====] - 442s 4s/step - loss: 0.3886 - accuracy: 0.8571 - val_loss: 0.3484 - val_accuracy: 0.8837
Epoch 7/20
109/109 [=====] - 435s 4s/step - loss: 0.3667 - accuracy: 0.8626 - val_loss: 0.3636 - val_accuracy: 0.8733
Epoch 8/20
109/109 [=====] - 432s 4s/step - loss: 0.3249 - accuracy: 0.8802 - val_loss: 0.3188 - val_accuracy: 0.8919
Epoch 9/20
109/109 [=====] - 1153s 11s/step - loss: 0.3211 - accuracy: 0.8854 - val_loss: 0.3328 - val_accuracy: 0.8942
Epoch 10/20
109/109 [=====] - 1108s 10s/step - loss: 0.2919 - accuracy: 0.8927 - val_loss: 0.3578 - val_accuracy: 0.8872
Epoch 11/20
109/109 [=====] - 1018s 9s/step - loss: 0.3035 - accuracy: 0.8846 - val_loss: 0.3867 - val_accuracy: 0.8779
Epoch 12/20
109/109 [=====] - 1205s 11s/step - loss: 0.2800 - accuracy: 0.8967 - val_loss: 0.3728 - val_accuracy: 0.8779
Epoch 13/20
...
Epoch 19/20
109/109 [=====] - 760s 7s/step - loss: 0.2189 - accuracy: 0.9225 - val_loss: 0.3577 - val_accuracy: 0.8872
Epoch 20/20
109/109 [=====] - 445s 4s/step - loss: 0.2147 - accuracy: 0.9222 - val_loss: 0.3553 - val_accuracy: 0.8895
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.

Evaluasi Model
```







Link Github : https://github.com/IchsanNurRachmadY/PCD_Pertemuan-11.git