

2. PET CLASSIFICATION AI-MODEL PROJECT IN JUPYTER NOTEBOOK

The image shows a Visual Studio Code editor window with a Jupyter Notebook file named 'PET_CLASSIFICATION.ipynb' open. The file path is 'C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb'. The notebook has two code cells. The first cell, labeled '[1]', is titled '# INSTALLING AND IMPORTING MODULES' and contains the following code:

```
# INSTALLING AND IMPORTING MODULES
#!pip install pandas
#!pip install matplotlib
#!pip install glob2
#!pip install os-sys
#!pip install numpy
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.image as image
import glob
import os
```

The second cell, labeled '[2]', is titled '# PROVIDING THE DATA SET TO IMAGES_FP' and contains the following code:

```
# PROVIDING THE DATA SET TO IMAGES_FP
images_fp = './images'
image_names = [os.path.basename(file) for file in glob.glob(os.path.join(images_fp, '*.jpg'))]
image_names
```

Below the second cell, a list of image names is displayed:

```
...
['Abyssinian 1.jpg',
'Abyssinian 10.jpg',
'Abyssinian 100.jpg',
'Abyssinian 101.jpg',
'Abyssinian 102.jpg',
'Abyssinian 103.jpg',
'Abyssinian 104.jpg',
'Abyssinian 105.jpg',
'Abyssinian 106.jpg',
'Abyssinian 107.jpg',
'Abyssinian 108.jpg',
'Abyssinian 109.jpg',
'Abyssinian 11.jpg',
'Abyssinian 110.jpg',
'Abyssinian 111.jpg',
'Abyssinian 112.jpg',
'Abyssinian 113.jpg',
```

The VS Code interface includes a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top bar shows the file path. The bottom status bar shows the file encoding as UTF-8 and the line number 14. A notification bubble in the bottom right corner asks for feedback.

[illegible]

```
File Edit Selection View Go Run Terminal Help
C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb > ...

+ Code + Markdown + Run All + Clear All Outputs + Outline ... Python 3.12.3

'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
'Abyssinian',
...
'beagle',
'beagle',
'beagle',
'beagle',
...]

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

# DEFINING FUNCTION FOR PETS
def label_encode(label):
    if label == 'abyssinian': return 0
    elif label == 'american bulldog': return 1
    elif label == 'american pit bull terrier': return 2
    elif label == 'basset hound': return 3
    elif label == 'Birman': return 4
    elif label == 'Bombay': return 5
    elif label == 'boxer': return 6
    elif label == 'chihuahua': return 7
    elif label == 'Egyptian mau': return 8
    elif label == 'german shorthaired': return 9
    elif label == 'newfoundland': return 10
    elif label == 'japanese chin': return 11
    elif label == 'english cocker spaniel': return 12
    elif label == 'Bengal': return 13
    elif label == 'pomeranian': return 14
    elif label == 'english setter': return 15
    elif label == 'wheaten terrier': return 16
```

```
File Edit Selection View Go Run Terminal Help
C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb > ...

+ Code + Markdown + Run All + Clear All Outputs + Outline ... Python 3.12.3

!pip install tensorflow
import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img, img_to_array

features = []
labels = []
IMAGE_SIZE = (224,224)
for name in image_names:
    label = ''.join(name.split('.')[::-1])
    label_encoded = label_encode(label)
    if label_encoded != None:
        img = load_img(os.path.join(images_fp, name))
        img = tf.image.resize_with_pad(img_to_array(img, dtype='uint8'), *IMAGE_SIZE).numpy().astype('uint8')
        image = np.array(img)
        features.append(image)
        labels.append(label_encoded)

features

[[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]]], dtype=object)]
```



```
File Edit Selection View Go Run Terminal Help
PET_CLASSIFICATION.ipynb
C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb > ...
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.12.3

...
1 2 3 4 5 6 7 9 10 11 12 13 14 15 16
0 True False False False False False False False False False False False False False
1 True False False False False False False False False False False False False False
2 True False False False False False False False False False False False False False
3 True False False False False False False False False False False False False False
4 True False False False False False False False False False False False False False
...
2995 False False False False False False False False False False False False False True
2996 False False False False False False False False False False False False False True
2997 False False False False False False False False False False False False False True
2998 False False False False False False False False False False False False False True
2999 False False False False False False False False False False False False False True

3000 rows x 15 columns

#pip install scikit-learn
from sklearn.model_selection import train_test_split

[12] Python

print("New features array shape:", features_array.shape)
print("Labels one-hot shape:", labels_one_hot.shape)

[13] Python

... New features array shape: (3000, 224, 224, 3)
Labels one-hot shape: (3000, 15)

#Trian = 70%, val = 30 % and test = 25%
x_train, x_test, y_train, y_test = train_test_split(features_array, labels_one_hot, test_size=0.3, random_state=38)

[14] Python
```

```
File Edit Selection View Go Run Terminal Help
PET_CLASSIFICATION.ipynb
C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb > ...
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.12.3

#training and validation sets 50%
x_train, x_val, y_train, y_val = train_test_split(x_train,y_train, test_size = 0.5, random_state=4)

[15] Python

from tensorflow.keras import layers, Input ,Model
from tensorflow.keras.models import Sequential
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input as pp_i
from tensorflow.keras.layers import RandomFlip , RandomRotation,Dense,Dropout
from tensorflow.keras.losses import CategoricalCrossentropy
from tensorflow.keras.optimizers import Adam

[16] Python

data_augmentation = Sequential([RandomFlip("horizontal_and_vertical"),RandomRotation(0.4)])
prediction_layers = Dense(15,activation = 'softmax')

[17] Python

resnet_model = ResNet50(include_top = False, pooling = 'avg', weights = 'imagenet')
resnet_model.trainable = False
preprocess_input = pp_i

[18] Python

#model Building
inputs = Input(shape = (224,224,3))
x = data_augmentation(inputs)
x = preprocess_input(x)
x = resnet_model(x, training = False)
x = Dropout(0.2)(x)
outputs = prediction_layers(x)
model = Model(inputs, outputs)

[19] Python
```

```
File Edit Selection View Go Run Terminal Help
C:\Users\sanju> OneDrive> Documents> pet_class> PET_CLASSIFICATION.ipynb> ...
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.12.3

model.compile(optimizer = Adam(), loss = CategoricalCrossentropy(), metrics = ['accuracy'])

model_history = model.fit(x=x_train, y=y_train, validation_data = (x_val, y_val), epochs = 15)

Epoch 1/15
33/33 76s 2s/step - accuracy: 0.2042 - loss: 2.7067 - val_accuracy: 0.7790 - val_loss: 0.8068
Epoch 2/15
33/33 68s 2s/step - accuracy: 0.6779 - loss: 1.0256 - val_accuracy: 0.8752 - val_loss: 0.4278
Epoch 3/15
33/33 67s 2s/step - accuracy: 0.7969 - loss: 0.6590 - val_accuracy: 0.9095 - val_loss: 0.3295
Epoch 4/15
33/33 67s 2s/step - accuracy: 0.8258 - loss: 0.5619 - val_accuracy: 0.9086 - val_loss: 0.2970
Epoch 5/15
33/33 68s 2s/step - accuracy: 0.8435 - loss: 0.4685 - val_accuracy: 0.9190 - val_loss: 0.2661
Epoch 6/15
33/33 67s 2s/step - accuracy: 0.8552 - loss: 0.4591 - val_accuracy: 0.9210 - val_loss: 0.2471
Epoch 7/15
33/33 67s 2s/step - accuracy: 0.8971 - loss: 0.3664 - val_accuracy: 0.9305 - val_loss: 0.2375
Epoch 8/15
33/33 67s 2s/step - accuracy: 0.8879 - loss: 0.3748 - val_accuracy: 0.9371 - val_loss: 0.2138
Epoch 9/15
33/33 67s 2s/step - accuracy: 0.9154 - loss: 0.2763 - val_accuracy: 0.9324 - val_loss: 0.2192
Epoch 10/15
33/33 69s 2s/step - accuracy: 0.9083 - loss: 0.2866 - val_accuracy: 0.9105 - val_loss: 0.2579
Epoch 11/15
33/33 67s 2s/step - accuracy: 0.9075 - loss: 0.2902 - val_accuracy: 0.9343 - val_loss: 0.2135
Epoch 12/15
33/33 67s 2s/step - accuracy: 0.9270 - loss: 0.2473 - val_accuracy: 0.9371 - val_loss: 0.1899
Epoch 13/15
...
Epoch 14/15
33/33 67s 2s/step - accuracy: 0.9331 - loss: 0.2308 - val_accuracy: 0.9343 - val_loss: 0.1928
Epoch 15/15
33/33 69s 2s/step - accuracy: 0.9317 - loss: 0.1907 - val_accuracy: 0.9248 - val_loss: 0.2232
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

```
File Edit Selection View Go Run Terminal Help
C:\Users\sanju> OneDrive> Documents> pet_class> PET_CLASSIFICATION.ipynb> ...
+ Code + Markdown | Run All | Clear All Outputs | Outline ... Python 3.12.3

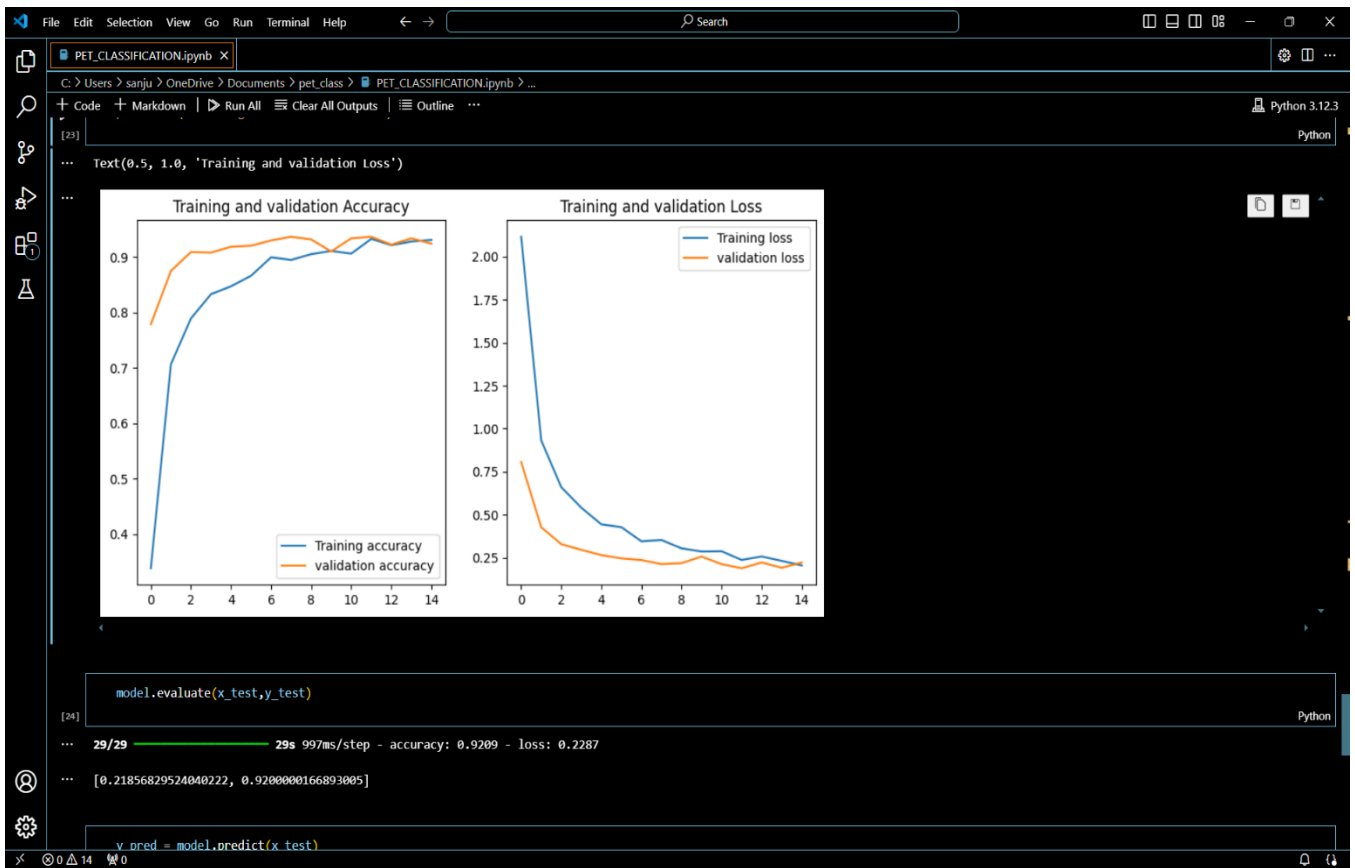
Epoch 10/15
33/33 67s 2s/step - accuracy: 0.9124 - loss: 0.2703 - val_accuracy: 0.9324 - val_loss: 0.2192
Epoch 11/15
33/33 67s 2s/step - accuracy: 0.9075 - loss: 0.2902 - val_accuracy: 0.9343 - val_loss: 0.2135
Epoch 12/15
33/33 67s 2s/step - accuracy: 0.9270 - loss: 0.2473 - val_accuracy: 0.9371 - val_loss: 0.1899
Epoch 13/15
...
Epoch 14/15
33/33 67s 2s/step - accuracy: 0.9331 - loss: 0.2308 - val_accuracy: 0.9343 - val_loss: 0.1928
Epoch 15/15
33/33 69s 2s/step - accuracy: 0.9317 - loss: 0.1907 - val_accuracy: 0.9248 - val_loss: 0.2232
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

acc = model_history.history['accuracy']
val_acc = model_history.history['val_accuracy']
loss = model_history.history['loss']
val_loss = model_history.history['val_loss']

epochs_range = range(15)
plt.figure(figsize = (10,5))
plt.subplot(1,2,1)
plt.plot(epochs_range, acc, label = 'Training accuracy')
plt.plot(epochs_range, val_acc, label = 'validation accuracy')
plt.legend(loc = 'lower right')
plt.title('Training and validation Accuracy')

plt.subplot(1,2,2)
plt.plot(epochs_range, loss, label = 'Training loss')
plt.plot(epochs_range, val_loss, label = 'validation loss')
plt.legend(loc = 'upper right')
plt.title('Training and validation Loss')

Text(0.5, 1.0, 'Training and validation Loss')
```



File Edit Selection View Go Run Terminal Help

Search

PET_CLASSIFICATION.ipynb

C:\Users> sanju > OneDrive > Documents > pet_class > PET_CLASSIFICATION.ipynb > ...

+ Code + Markdown | Run All | Clear All Outputs | Outline ...

Python 3.12.3

[25]

... 29/29 31s 1s/step

y_pred = model.predict(x_test)

[26]

print("THE RESULTED ARRAY",y_pred)

THE RESULTED ARRAY [[5.8490747e-07 2.2212479e-04 5.6363560e-06 ... 8.6754717e-08
1.2230744e-04 5.6569593e-06]
[1.5772545e-06 4.8678605e-05 3.1463216e-05 ... 2.1247447e-06
9.9392289e-01 9.6528063e-05]
[6.6566049e-06 1.2688189e-04 3.2823500e-05 ... 8.1706576e-06
4.3516457e-03 8.3286550e-06]
...
[5.5256791e-05 3.2410931e-06 3.4407592e-05 ... 5.6928932e-03
2.2819496e-04 1.2997835e-05]
[2.3663735e-05 7.4102136e-05 1.5124980e-03 ... 1.4927465e-05
4.3019753e-02 3.8745950e-04]
[2.5249118e-08 2.3912478e-06 6.7861174e-07 ... 2.4232571e-04
6.7483052e-05 1.0283243e-05]]

0 14 0