Topics in Deep Learning

Team-17

DROWSINESS DETECTION IN DRIVERS

G. Sai Kruthi	PES2UG21CS179
M V Sanjay	PES2UG21CS295
Nama Sai Pranav	PES2UG21CS317
Patel Saketh Kumar Reddy	PES2UG21CS369

PROBLEM STATEMENT

Drowsy driving can significantly affect driving performance and overall road safety. Statistically, the main causes are decreased alertness and attention of the drivers. The combination of deep learning and computer-vision algorithm applications has been proven to be one of the most effective approaches for the detection of drowsiness

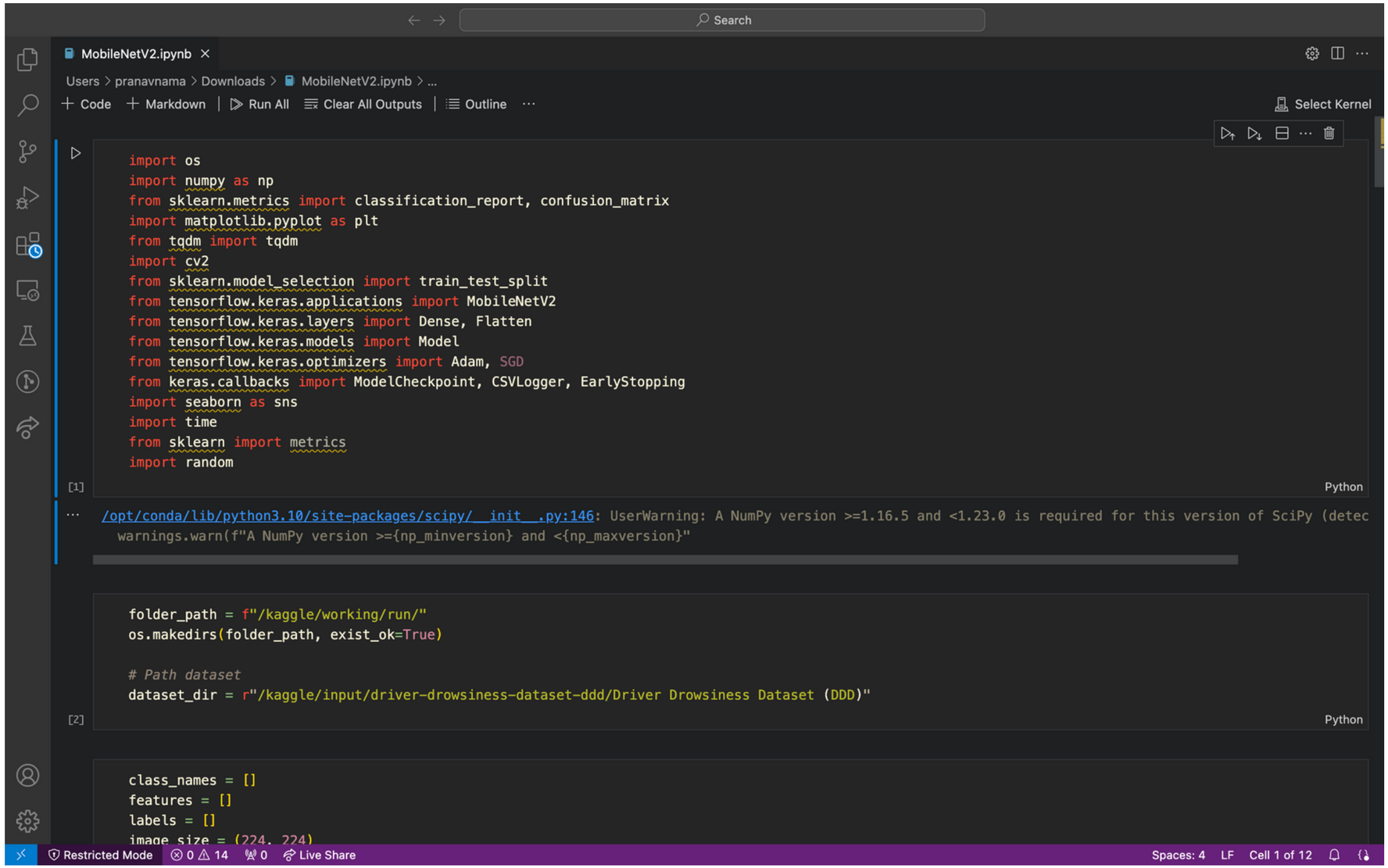
Drowsy & non-drowsy dataset

- Our dataset contains two kinds of images which includes drowsy and non- drowsy pictures of drivers. It was created by collecting pictures from different newspaper articles and various other platforms. It is saved in the local storage with name as drowsy and non-drowsy

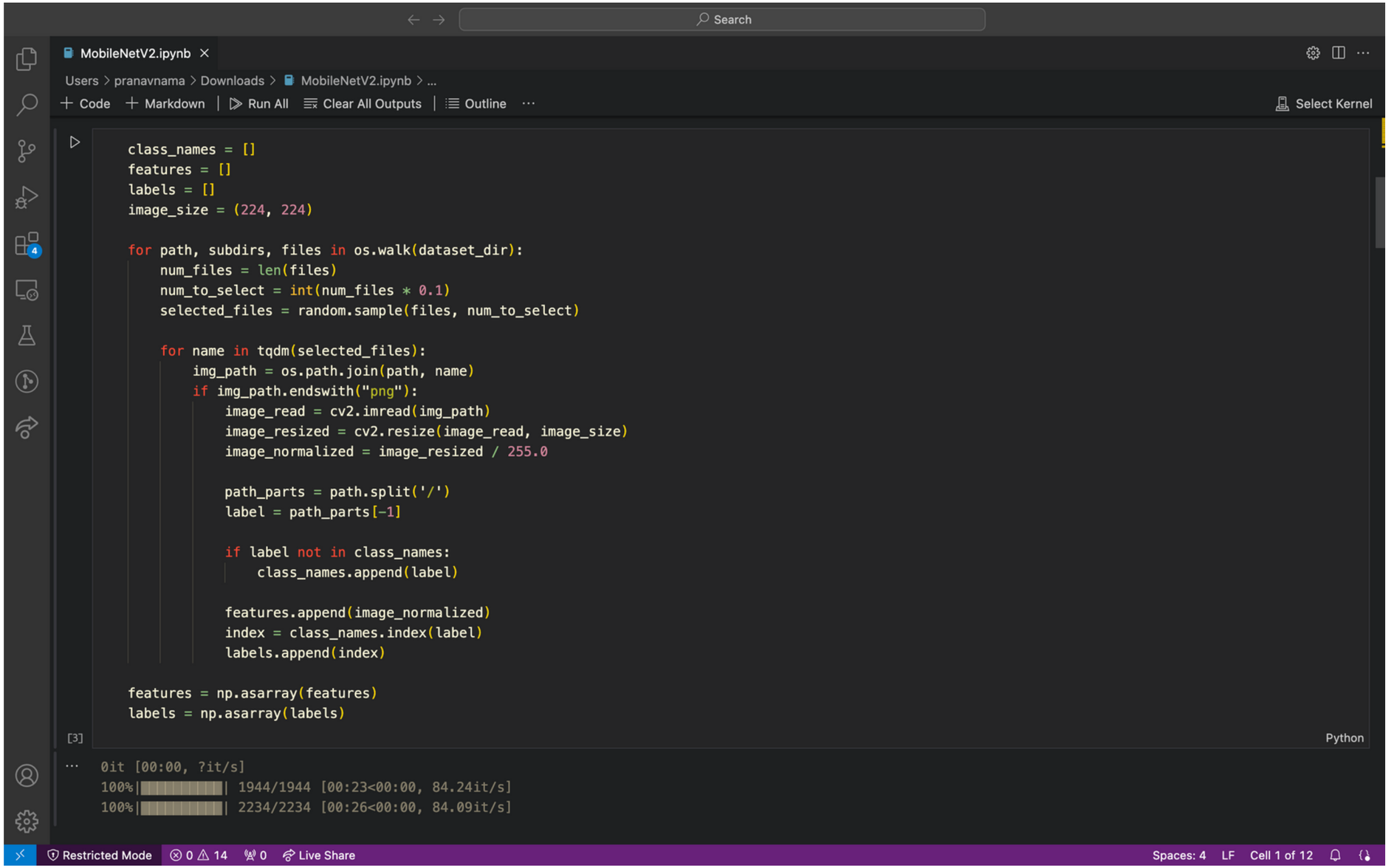
Dataset

- **Tools used for dataset creation:** selenium, beautifulsoup, pandas, scrapy
- **Classes:** Drowsy and Non-drowsy
- **Link to the dataset scrapped:** https://www.kaggle.com/datasets/ismailnasri20/driver-drowsiness-dataset-ddd

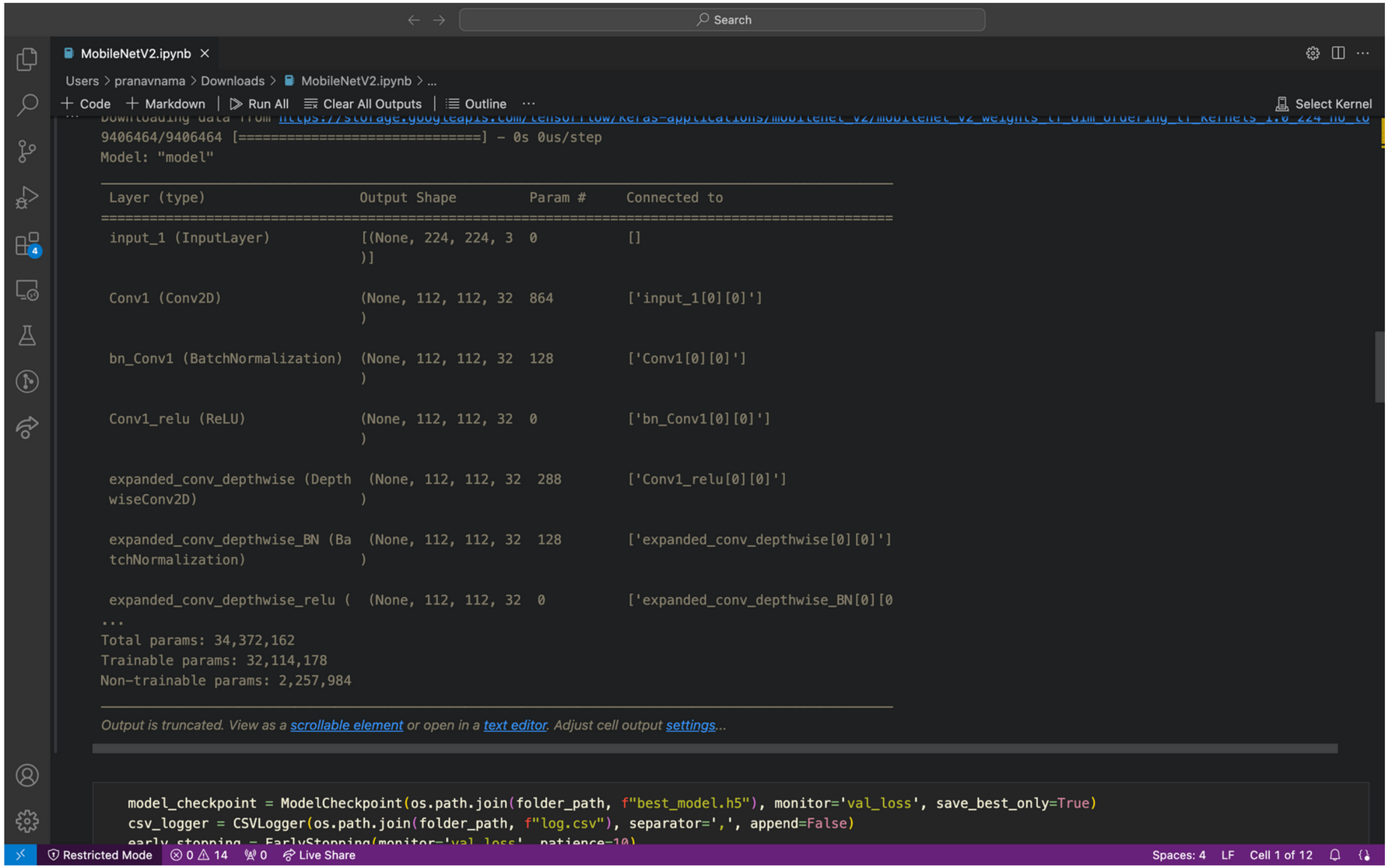
Code


Search
MobileNetV2.ipynb
Users > pranavnama > Downloads > MobileNetV2.ipynb > ...
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
import os
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
from tqdm import tqdm
import cv2
from sklearn.model_selection import train_test_split
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam, SGD
from keras.callbacks import ModelCheckpoint, CSVLogger, EarlyStopping
import seaborn as sns
import time
from sklearn import metrics
import random
[1]
Python
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detec
warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
folder_path = f"/kaggle/working/run/"
os.makedirs(folder_path, exist_ok=True)
# Path dataset
dataset_dir = r"/kaggle/input/driver-drowsiness-dataset-ddd/Driver Drowsiness Dataset (DDD)"
[2]
Python
class_names = []
features = []
labels = []
image_size = (224, 224)
Restricted Mode
0 14
0
Live Share
Spaces: 4
LF
Cell 1 of 12

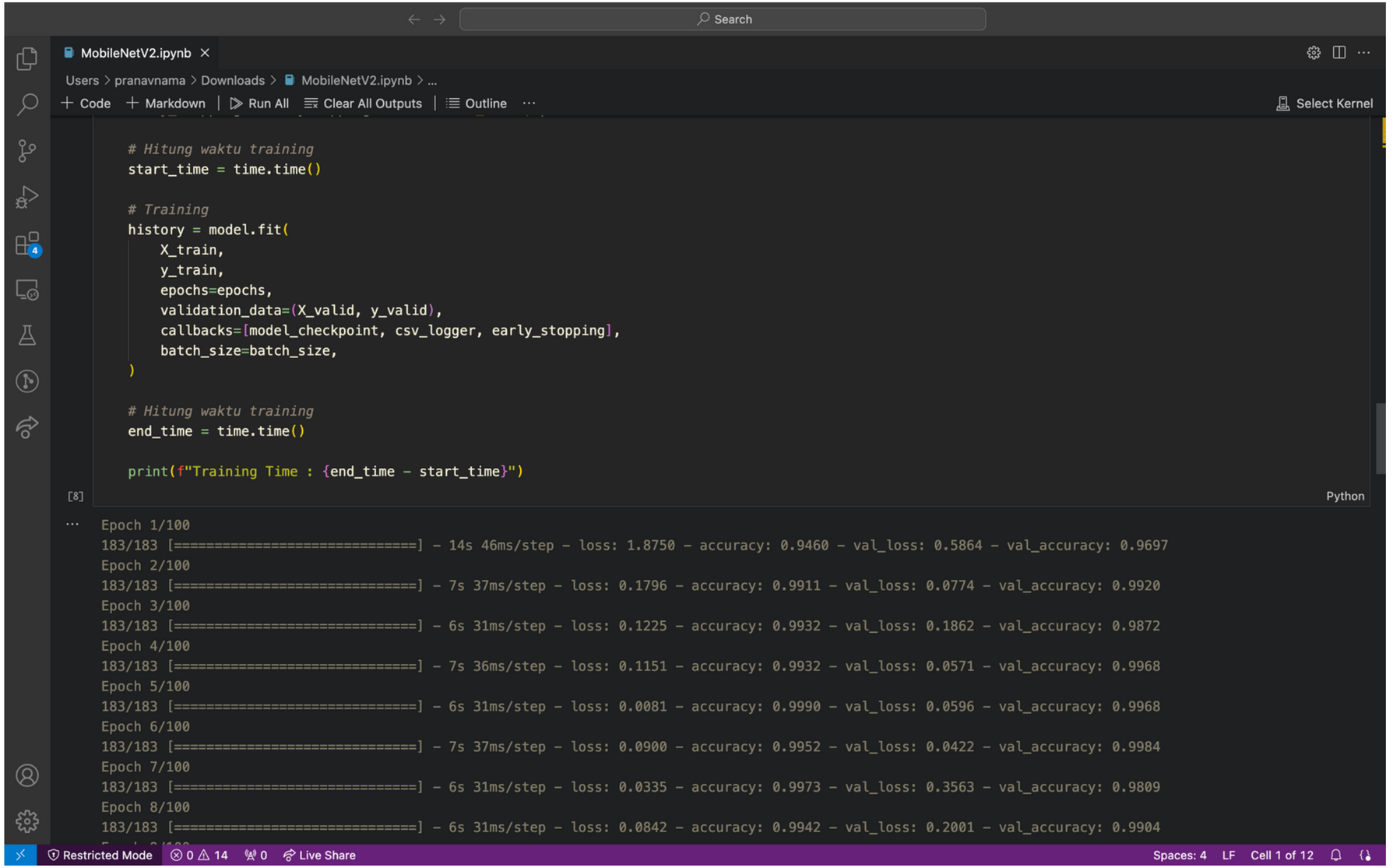
Code


Search
MobileNetV2.ipynb
Users > pranavnama > Downloads > MobileNetV2.ipynb > ...
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
class_names = []
features = []
labels = []
image_size = (224, 224)
for path, subdirs, files in os.walk(dataset_dir):
num_files = len(files)
num_to_select = int(num_files * 0.1)
selected_files = random.sample(files, num_to_select)
for name in tqdm(selected_files):
img_path = os.path.join(path, name)
if img_path.endswith("png"):
image_read = cv2.imread(img_path)
image_resized = cv2.resize(image_read, image_size)
image_normalized = image_resized / 255.0
path_parts = path.split('/')
label = path_parts[-1]
if label not in class_names:
class_names.append(label)
features.append(image_normalized)
index = class_names.index(label)
labels.append(index)
features = np.asarray(features)
labels = np.asarray(labels)
[3]
Python
0it [00:00, ?it/s]
100%|| 1944/1944 [00:23<00:00, 84.24it/s]
100%|| 2234/2234 [00:26<00:00, 84.09it/s]
Restricted Mode
0 14
0
Live Share
Spaces: 4
LF
Cell 1 of 12

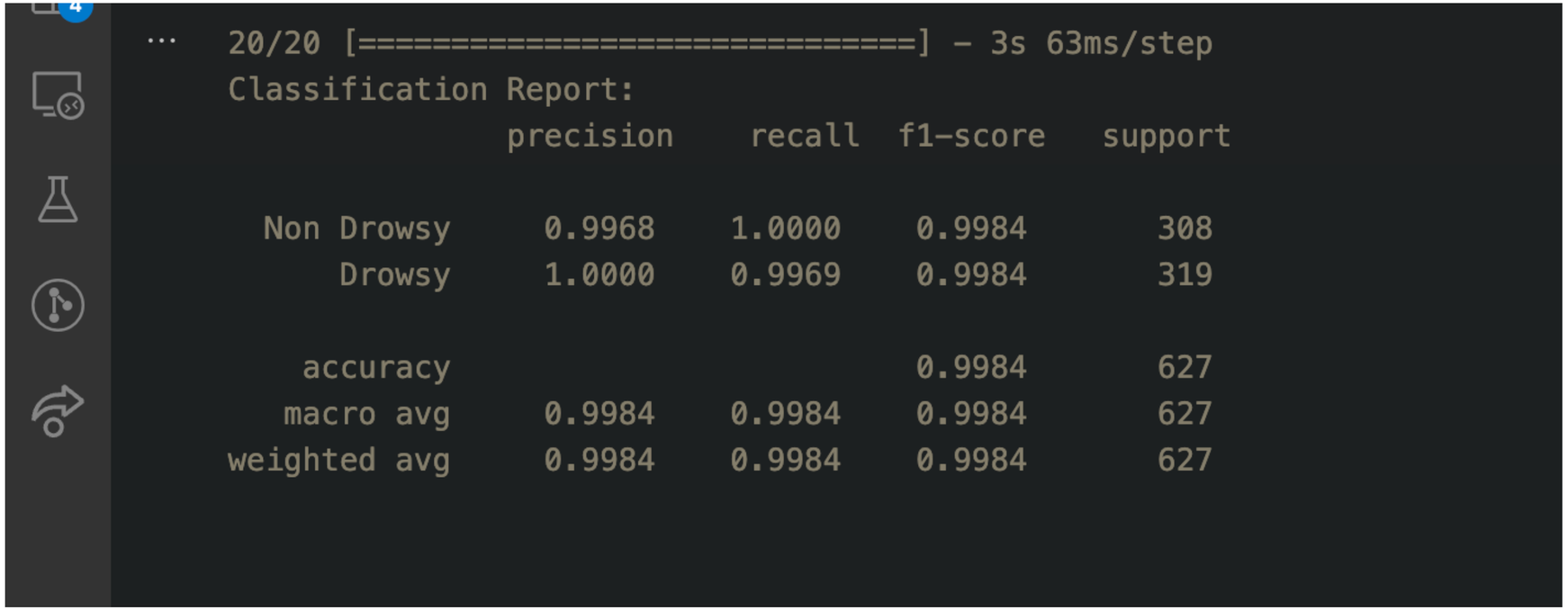
Code


Search
MobileNetV2.ipynb
Users > pranavnama > Downloads > MobileNetV2.ipynb > ...
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
9406464/9406464 [==============================] - 0s 0us/step
Model: "model"
Layer (type) Output Shape Param # Connected to
input_1 (InputLayer) [(None, 224, 224, 3 0 []
)]
Conv1 (Conv2D) (None, 112, 112, 32 864 ['input_1[0][0]']
)
bn_Conv1 (BatchNormalization) (None, 112, 112, 32 128 ['Conv1[0][0]']
)
Conv1_relu (ReLU) (None, 112, 112, 32 0 ['bn_Conv1[0][0]']
)
expanded_conv_depthwise (Depth (None, 112, 112, 32 288 ['Conv1_relu[0][0]']
wiseConv2D))
expanded_conv_depthwise_BN (Ba (None, 112, 112, 32 128 ['expanded_conv_depthwise[0][0]']
tchNormalization))
expanded_conv_depthwise_relu ((None, 112, 112, 32 0 ['expanded_conv_depthwise_BN[0][0
...
Total params: 34,372,162
Trainable params: 32,114,178
Non-trainable params: 2,257,984
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
model_checkpoint = ModelCheckpoint(os.path.join(folder_path, f"best_model.h5"), monitor='val_loss', save_best_only=True)
csv_logger = CSVLogger(os.path.join(folder_path, f"log.csv"), separator=',', append=False)
Restricted Mode
0 14
0
Live Share
Spaces: 4
LF
Cell 1 of 12

Code


Search
MobileNetV2.ipynb
Users > pranavnama > Downloads > MobileNetV2.ipynb > ...
Code
Markdown
Run All
Clear All Outputs
Outline
Select Kernel
# Hitung waktu training
start_time = time.time()
# Training
history = model.fit(
X_train,
y_train,
epochs=epochs,
validation_data=(X_valid, y_valid),
callbacks=[model_checkpoint, csv_logger, early_stopping],
batch_size=batch_size,
)
# Hitung waktu training
end_time = time.time()
print(f"Training Time : {end_time - start_time}")
[8]
Python
Epoch 1/100
183/183 [==========] - 14s 46ms/step - loss: 1.8750 - accuracy: 0.9460 - val_loss: 0.5864 - val_accuracy: 0.9697
Epoch 2/100
183/183 [==========] - 7s 37ms/step - loss: 0.1796 - accuracy: 0.9911 - val_loss: 0.0774 - val_accuracy: 0.9920
Epoch 3/100
183/183 [==========] - 6s 31ms/step - loss: 0.1225 - accuracy: 0.9932 - val_loss: 0.1862 - val_accuracy: 0.9872
Epoch 4/100
183/183 [==========] - 7s 36ms/step - loss: 0.1151 - accuracy: 0.9932 - val_loss: 0.0571 - val_accuracy: 0.9968
Epoch 5/100
183/183 [==========] - 6s 31ms/step - loss: 0.0081 - accuracy: 0.9990 - val_loss: 0.0596 - val_accuracy: 0.9968
Epoch 6/100
183/183 [==========] - 7s 37ms/step - loss: 0.0900 - accuracy: 0.9952 - val_loss: 0.0422 - val_accuracy: 0.9984
Epoch 7/100
183/183 [==========] - 6s 31ms/step - loss: 0.0335 - accuracy: 0.9973 - val_loss: 0.3563 - val_accuracy: 0.9809
Epoch 8/100
183/183 [==========] - 6s 31ms/step - loss: 0.0842 - accuracy: 0.9942 - val_loss: 0.2001 - val_accuracy: 0.9904
Restricted Mode
0 14
0
Live Share
Spaces: 4
LF
Cell 1 of 12

Classification Report

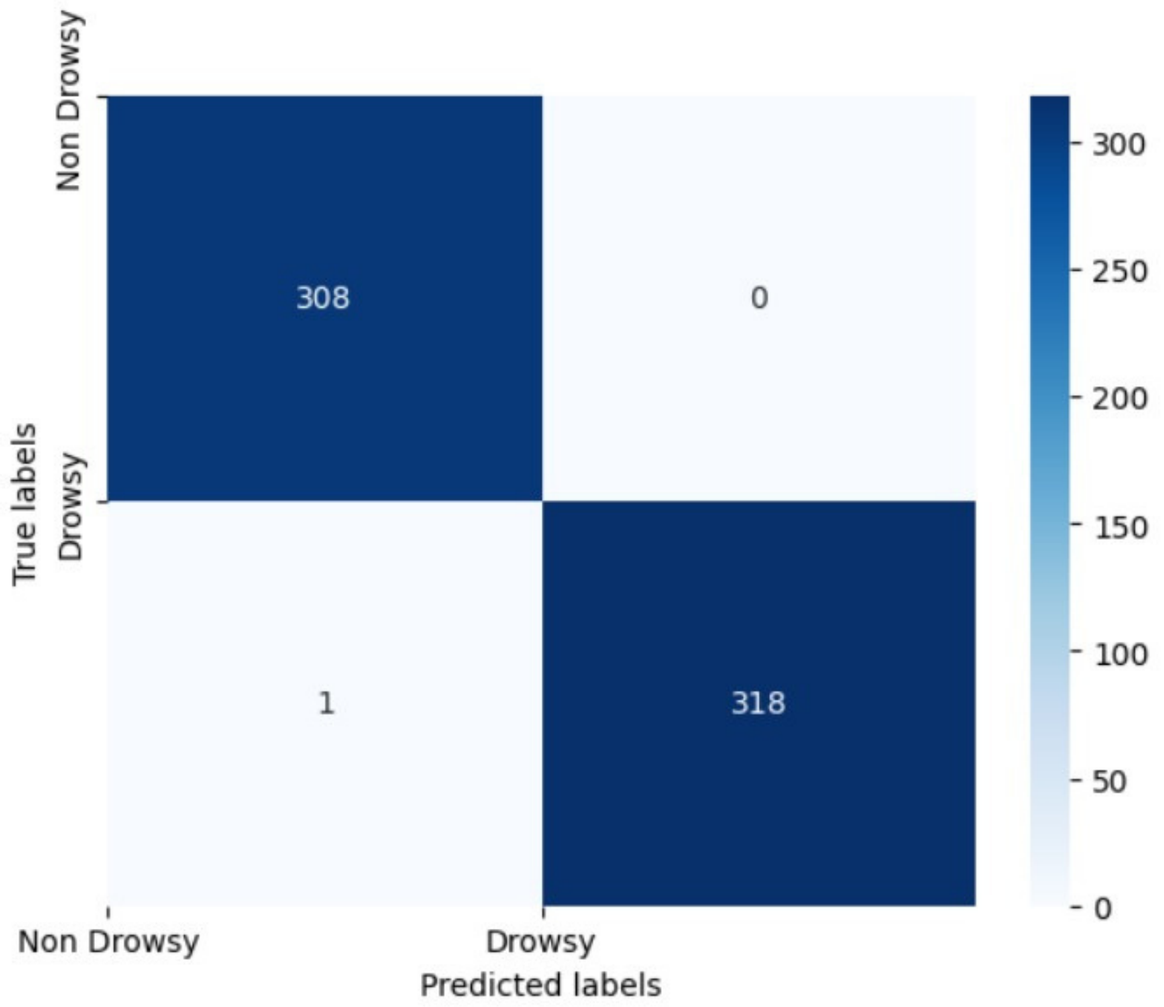


```
20/20 [==============================] - 3s 63ms/step
Classification Report:
                precision    recall  f1-score   support

    Non Drowsy     0.9968    1.0000    0.9984       308
        Drowsy     1.0000    0.9969    0.9984       319

      accuracy                         0.9984       627
     macro avg     0.9984    0.9984    0.9984       627
  weighted avg     0.9984    0.9984    0.9984       627
```

Possible Outcomes



Approach

We have implemented a transfer learning approach using the MobileNetV2 architecture for image classification tasks. It first prepares the data by resizing and normalizing images, then splits them into training, validation, and test sets. The model is defined by loading the pre-trained MobileNetV2 without its top layer and adding custom fully connected layers for classification. The training process fine-tunes the model on the training data, validating it with the validation set, and saving the best performing model. Visualization plots track training and validation loss and accuracy. Finally, the model is evaluated on the test set, generating a classification report and a confusion matrix for performance assessment.

Novelty

The novelty of this methodology lies in its efficient utilization of transfer learning with the MobileNetV2 architecture for image classification tasks. By leveraging a pre-trained model, specifically designed for mobile and embedded vision applications, the approach benefits from its ability to capture a wide range of visual features.

Number of Data Points

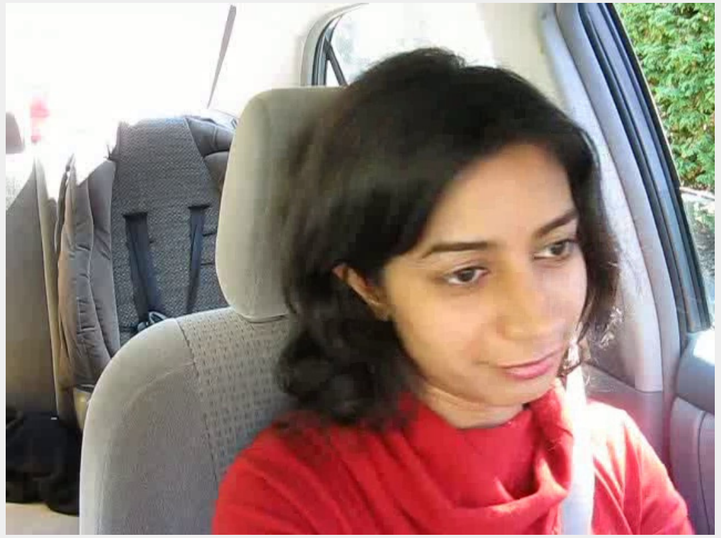
- DriverDrowsinessDetection comprises 2234 drowsy images 1940 and non-drowsy), each depicting a specific instance of driver’s face from various angles. This ensures a sufficiently diverse set of samples for training and evaluation

Ground Truth Values

- Categorisations accompanying images indicate different facial expression/ patterns.
- They serve as reference standards for training and evaluating machine learning models.
- Provide accurate information about the specific instance depicted in the images.

DROWSY IMAGES

NON-DROWSY IMAGES



THANK YOU