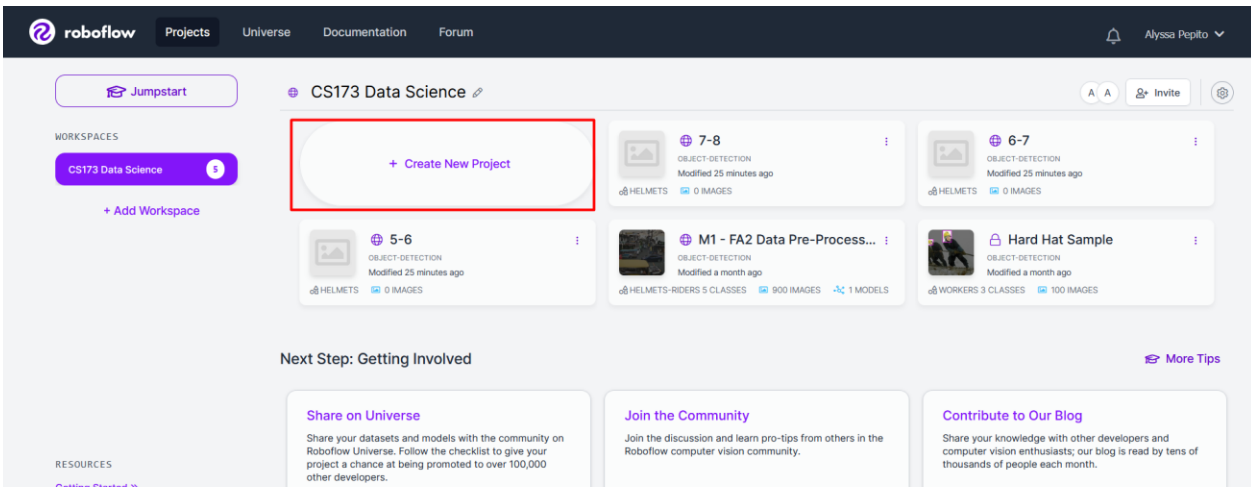
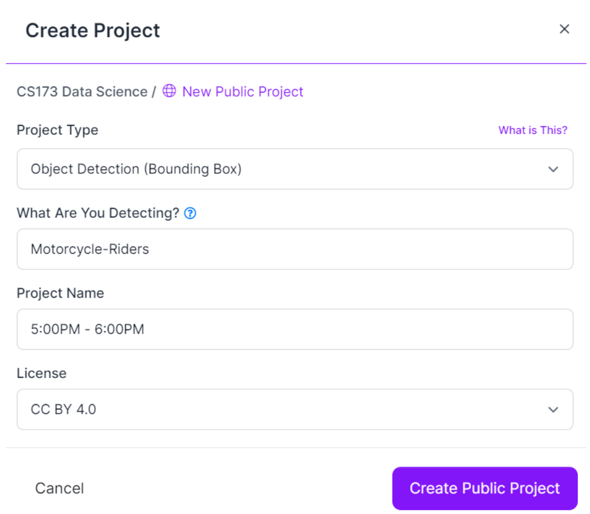
M2 – FA1 (Ground Truth): Process Documentation

I. Creating a Project for Each Time Scenario

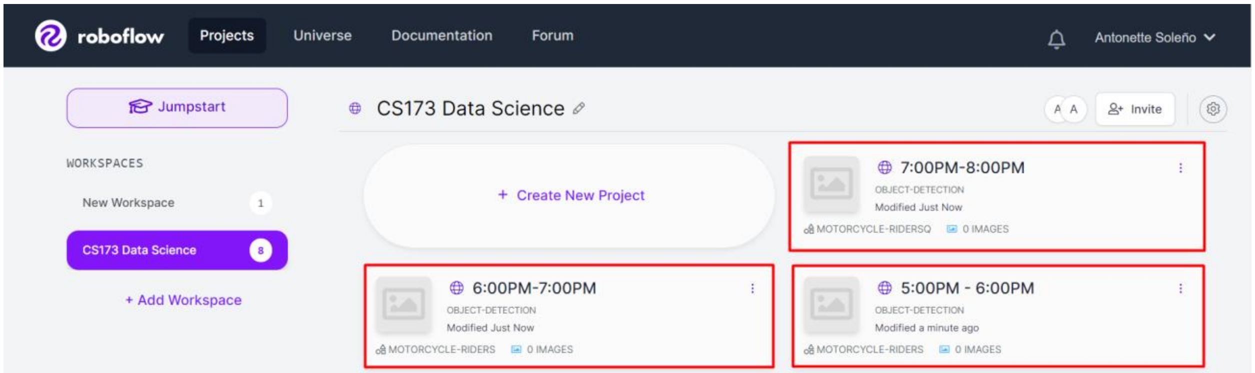
A separate project for each time scenario must be created to fulfill one of the main objectives of this project: to produce three confusion matrices for each YOLOv5 scale at different time scenarios. To do this, a project was created for each time scenario by clicking on + *Create New Project*.



A pop-up window should appear, prompting the user to complete the necessary details about the project they want to create. This step was repeated two more times to create a project for the 6:00PM-7:00PM and 7:00PM-8:00PM time scenarios.

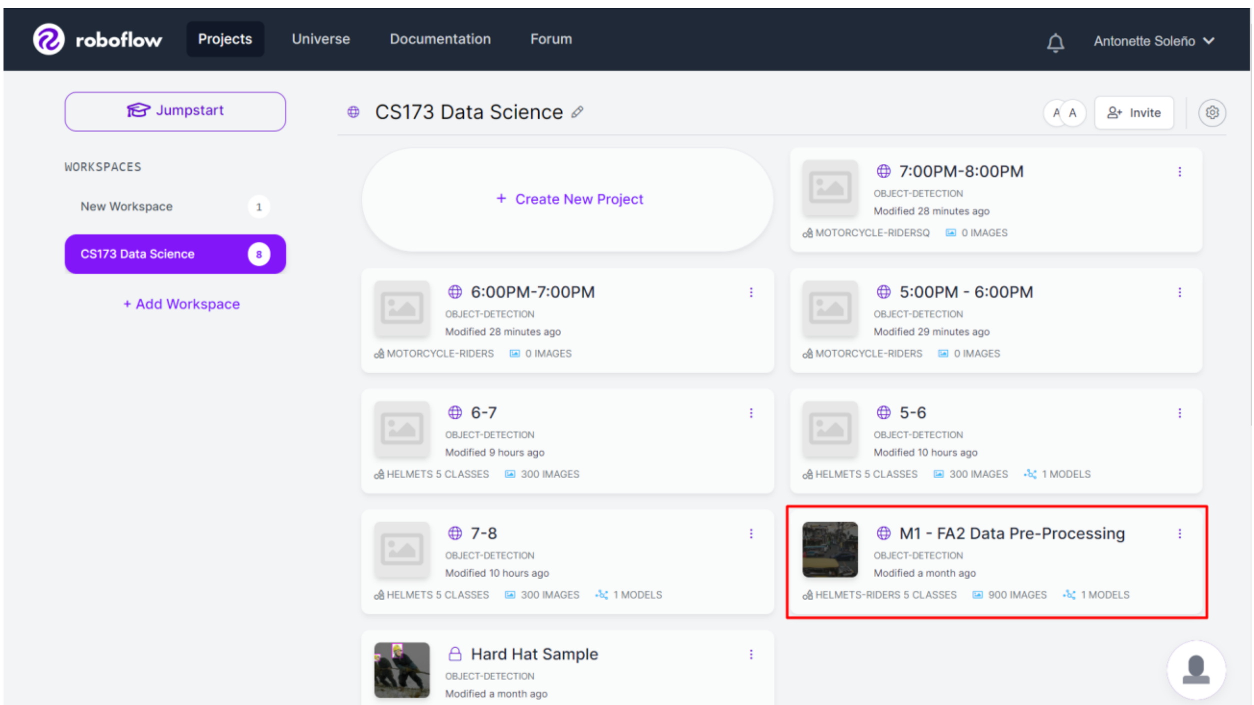


After creating these projects, the Roboflow workspace should look the figure below.

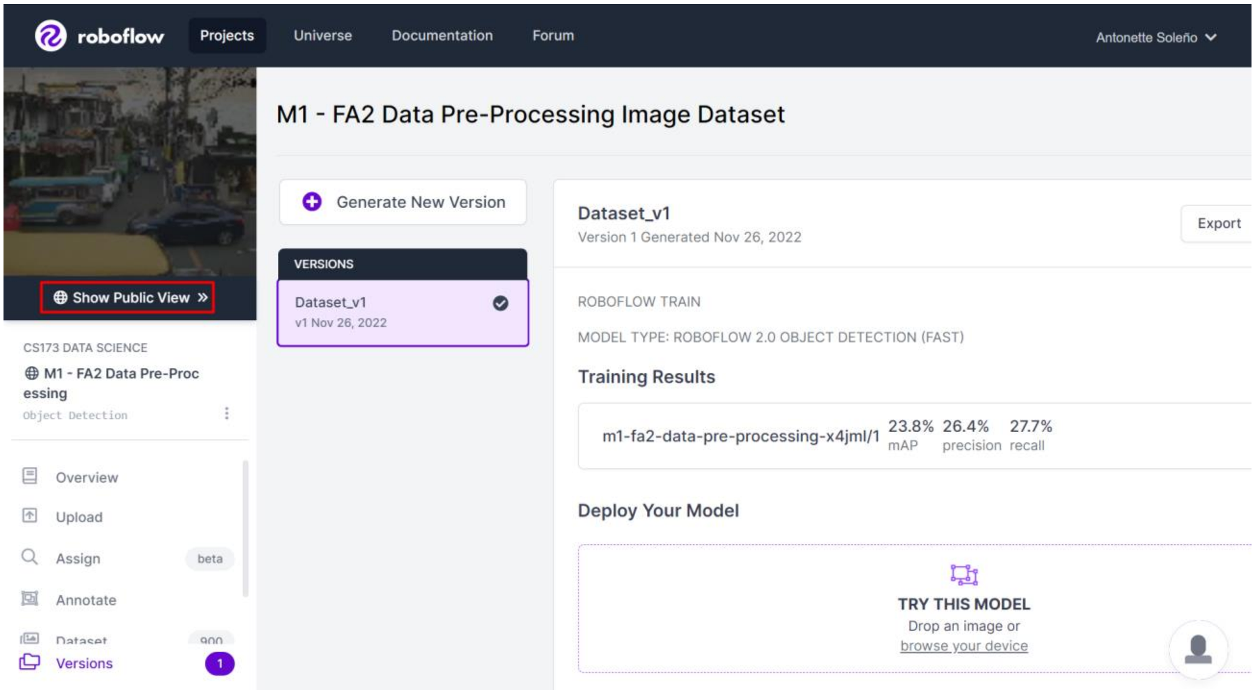


II. Cloning the Original Dataset

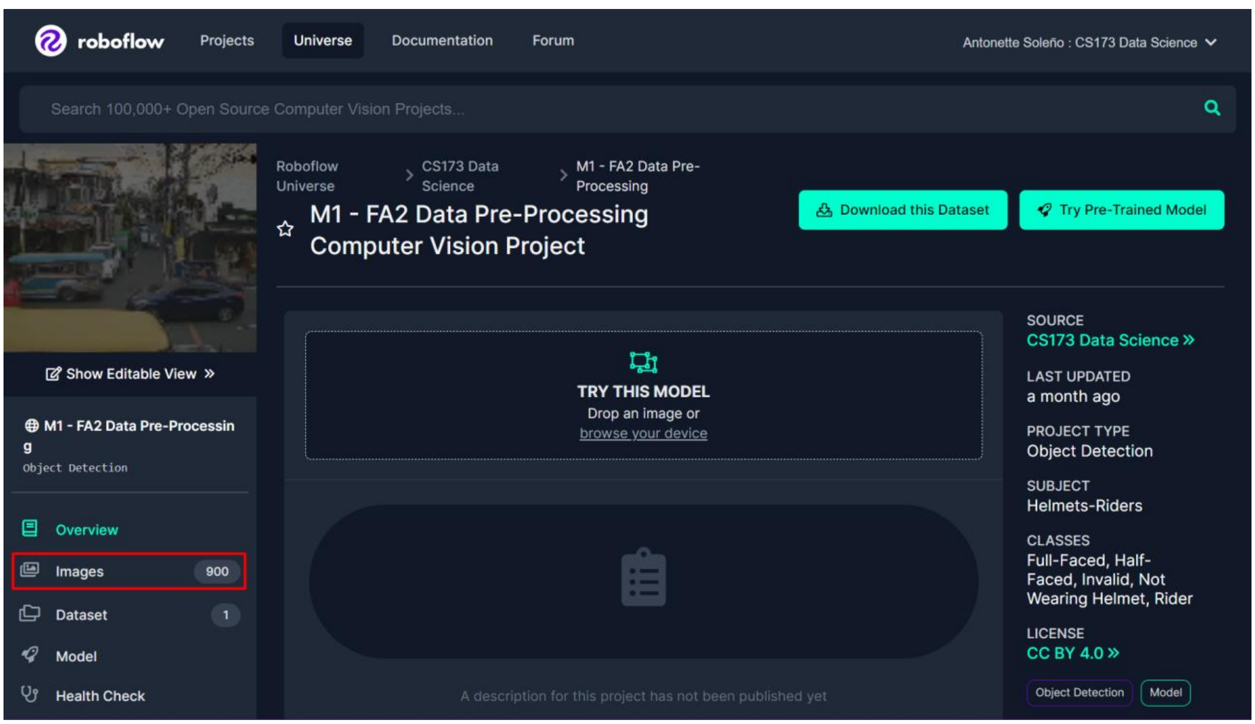
Dividing the original dataset for each time scenario requires cloning portions of the dataset. To do this, navigate to the project *M1 – FA2 Data Pre-Processing* within the Workspace and select it.



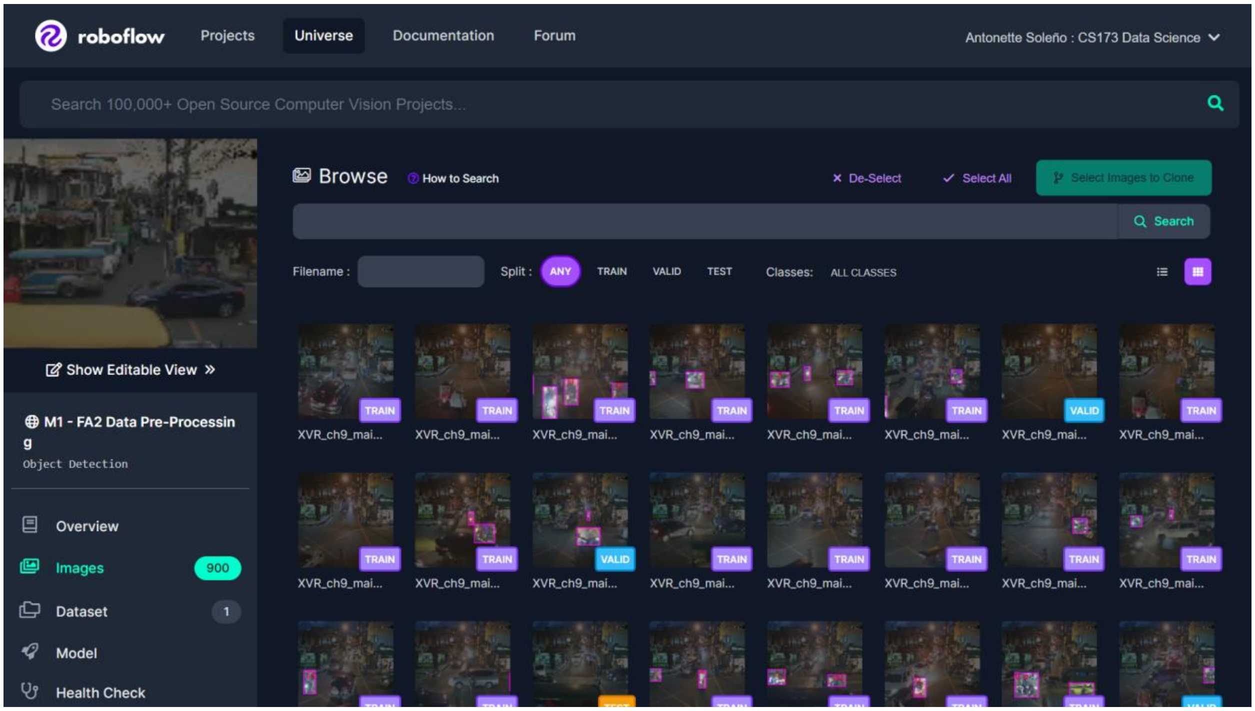
The figure below should appear after clicking on the project. Next, select *Show Public View >>* to access the Public View of the project.



Once the Public View of the dataset has been accessed, access the project's dataset by selecting the *Images* option on the left menu, as indicated below.



Selecting the *Images* option should generate the following window.



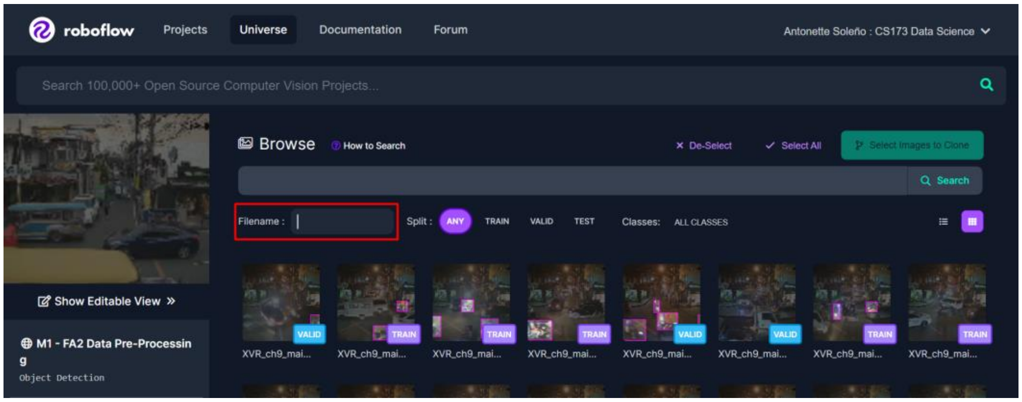
To segregate the dataset into the required time scenarios, the three, original videos of the dataset were first accessed to identify which files were of the 5:00PM-6:00PM, 6:00PM-7:00PM, 7:00PM-8:00PM time scenarios.

The following are the file names with their corresponding time scenarios.

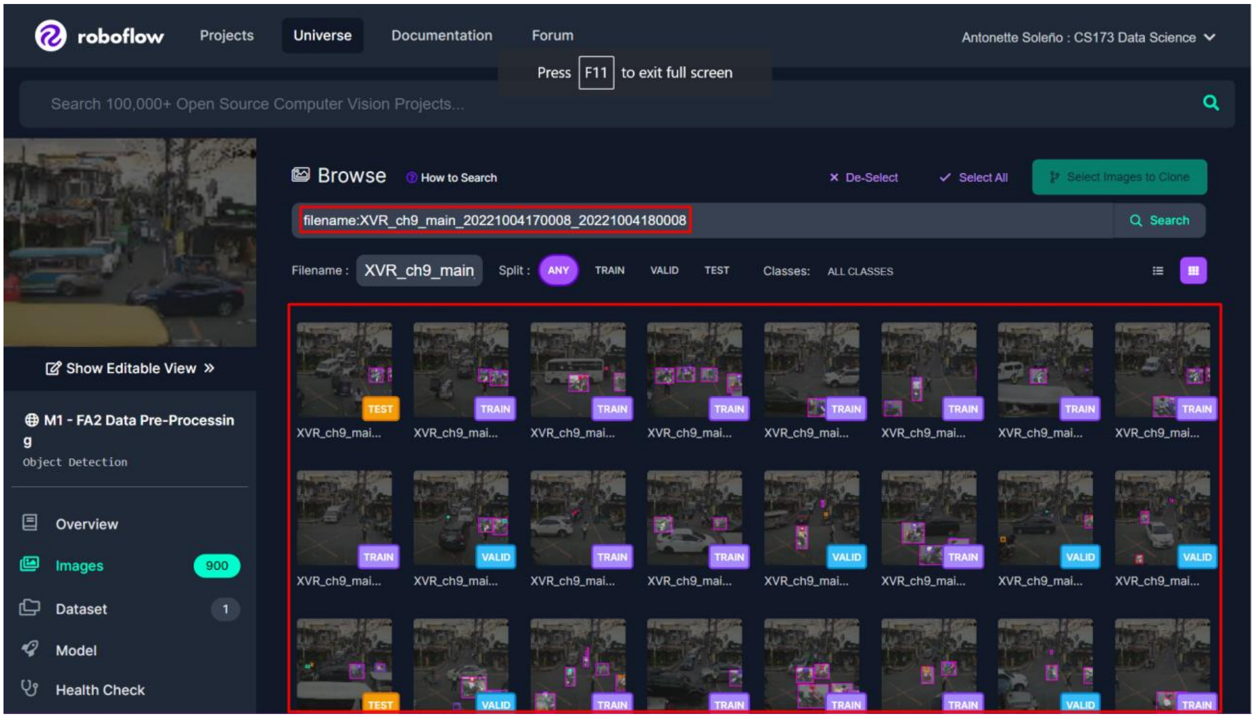
1. 5:00PM-6:00PM – *XVR_ch9_main_20221004170008_20221004180008*
2. 6:00PM-7:00PM – *XVR_ch9_main_20221004180008_20221004190008*
3. 7:00PM-8:00PM – *XVR_ch9_main_20221004190008_20221004200000*

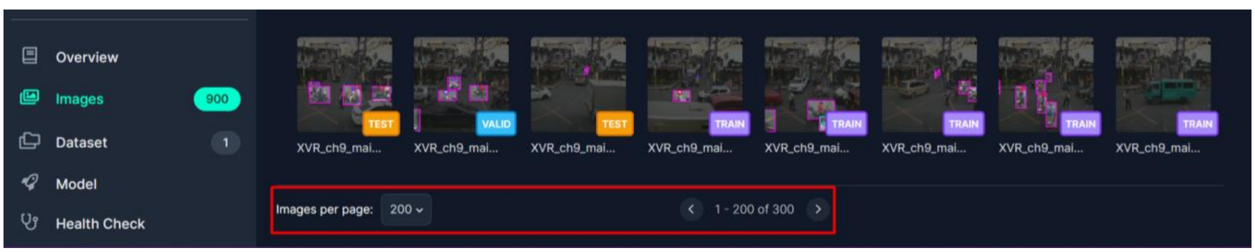
The above file names could then be used to identify which images from the Roboflow workspace are from the specified time scenario.

For example, to find which images are under the 5:00PM-6:00PM time period, enter the file name *XVR_ch9_main_20221004170008_20221004180008* in the text field highlighted below.

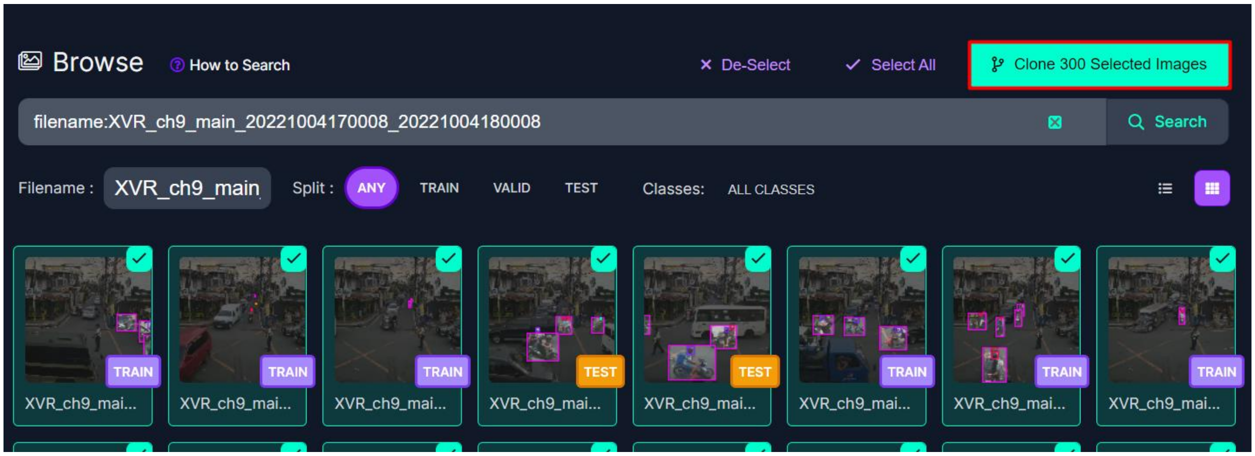


The following results should appear, with images of the specified file name only appearing.

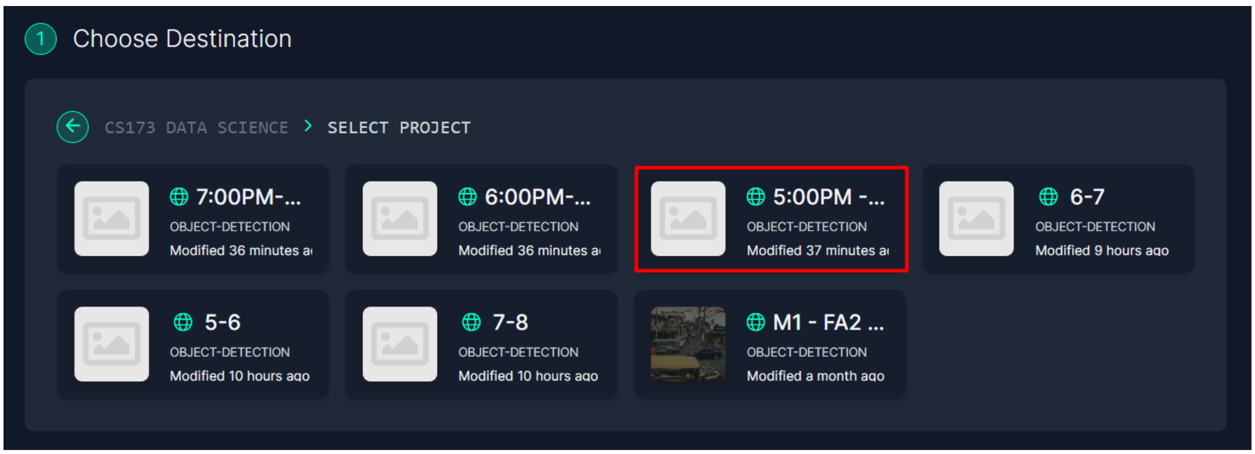




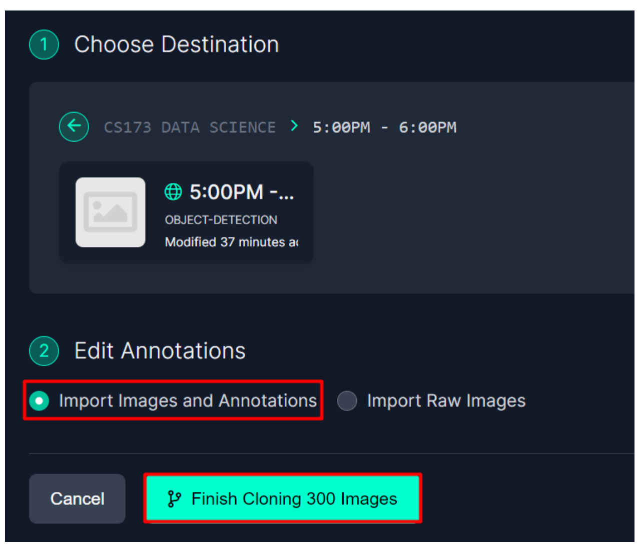
Now that the images have been identified and segregated according to their time period, they are now ready to be cloned. To do this, select "Select All" until all 300 images have been selected and click *Clone 300 Selected Images* to begin the cloning process.

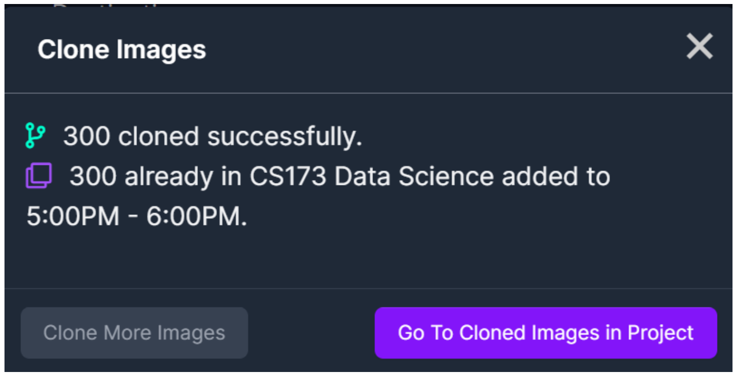


Since the images being cloned are from the 5:00PM-6:00PM time period, select the appropriate project as the clones' destination.

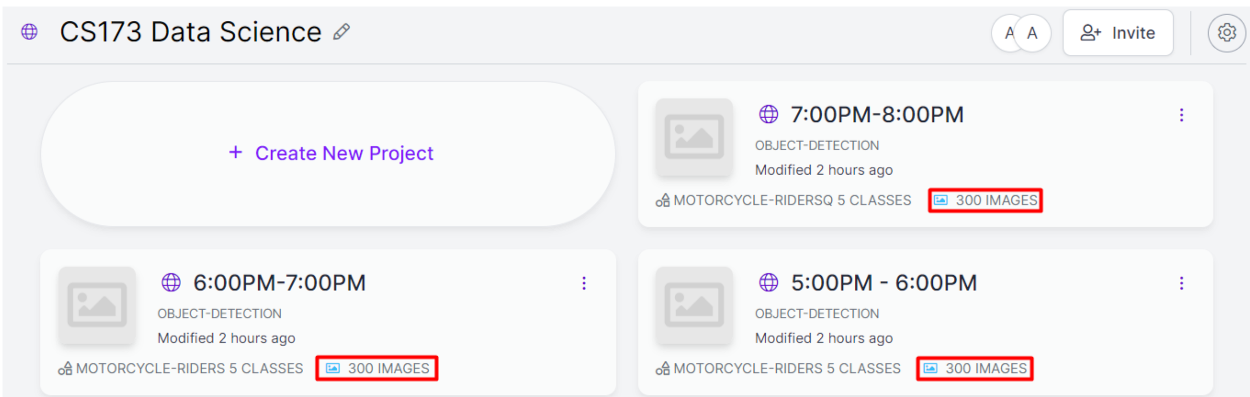


Then, leave the default selection to *Import Images and Annotations* and click the button *Finish Cloning 300 Images.*



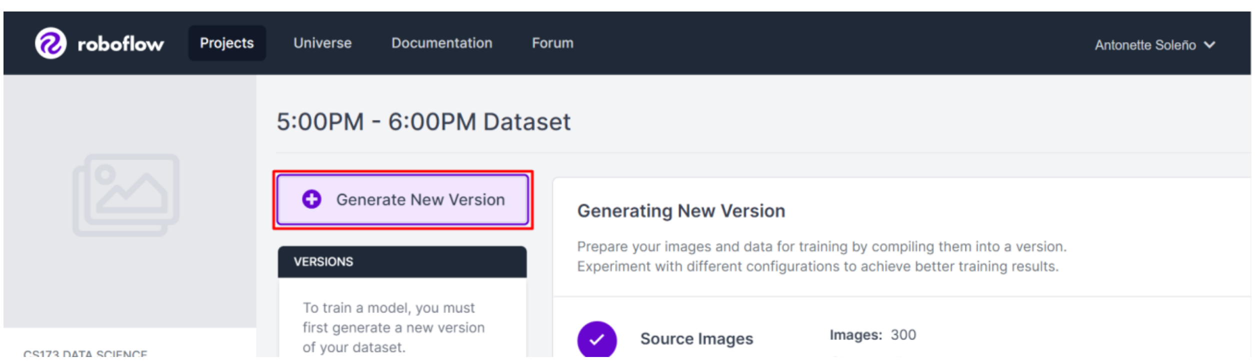


These same steps should be repeated for the two other time scenarios, 6:00PM-7:00PM and 7:00PM-8:00PM.

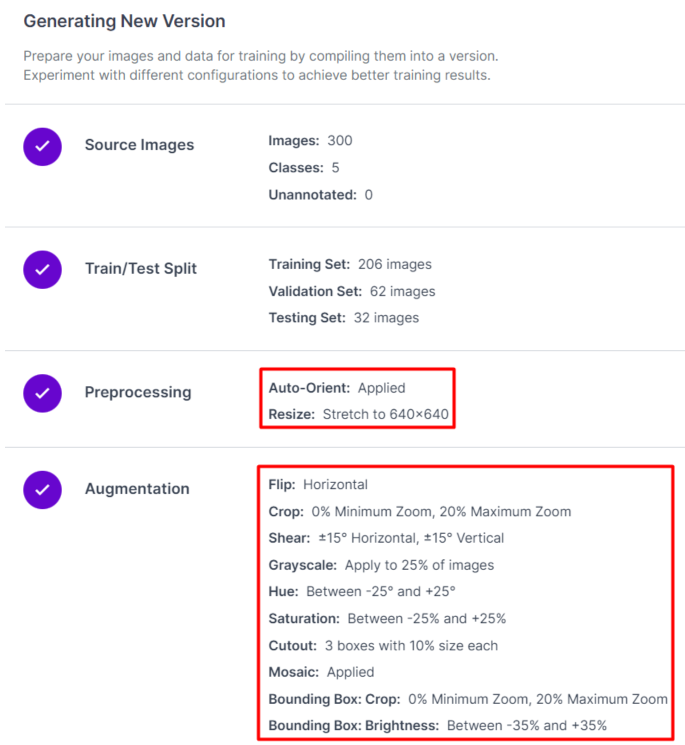


III. Generating YOLOv5 Versions of Each Time Scenario

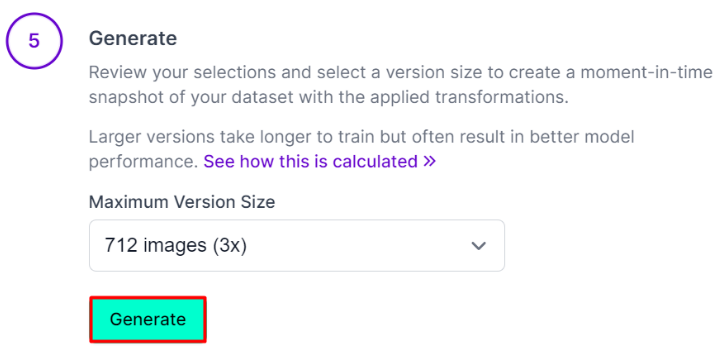
The next step after segregating the dataset according to their time scenarios is to generate a new version of the dataset. (This step will be repeated three times for each time scenario.) To generate a new version of the dataset, select which project you would like to work on first, then select *Generate New Version*.



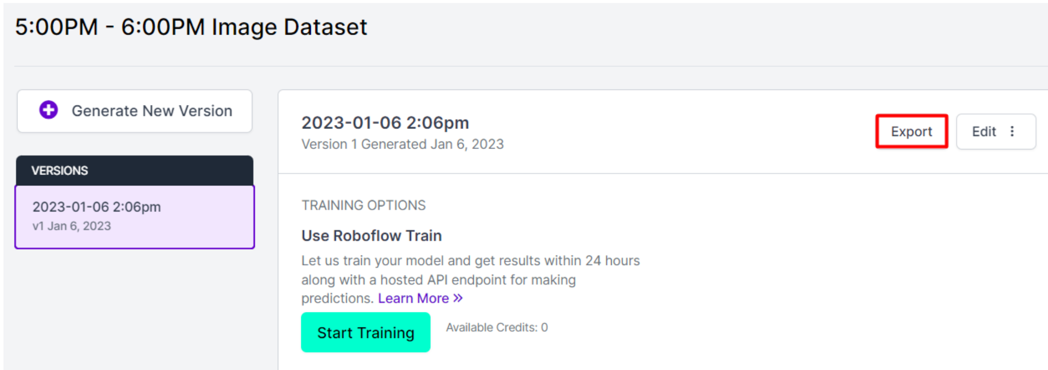
Generating a new version of the dataset will require the user to select the preprocessing and augmentation modifications they would like to make to their new dataset. In this project, the following features were selected for the preprocessing and augmentation features.



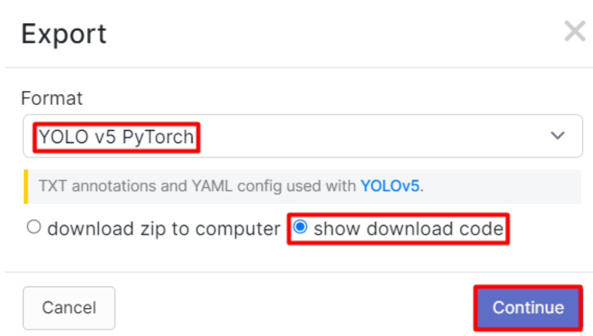
The final step is to generate the new dataset by selecting the *Generate* button. (Leave the default settings as is.)



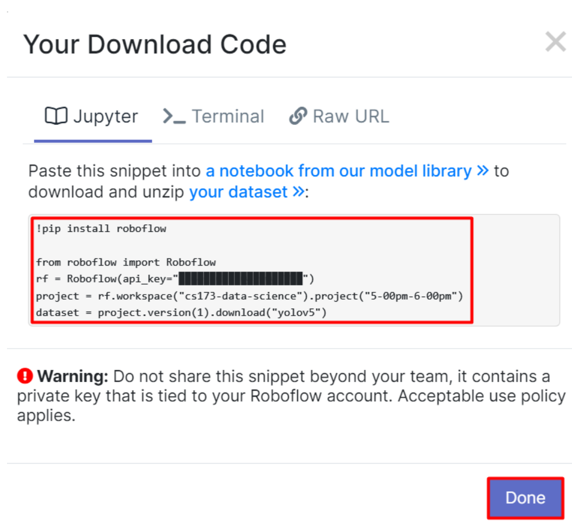
After the new dataset has finished generating, select the *Export* button to begin training the model in the YOLOv5 format.



In the pop-up window that will appear after selecting the *Export* button, select the option *YOLO v5 PyTorch* in the drop-down menu and the option to *show download code*. Select the *Continue* button to proceed.



The download code for the 5:00PM-6:00PM project should appear, as indicated below.



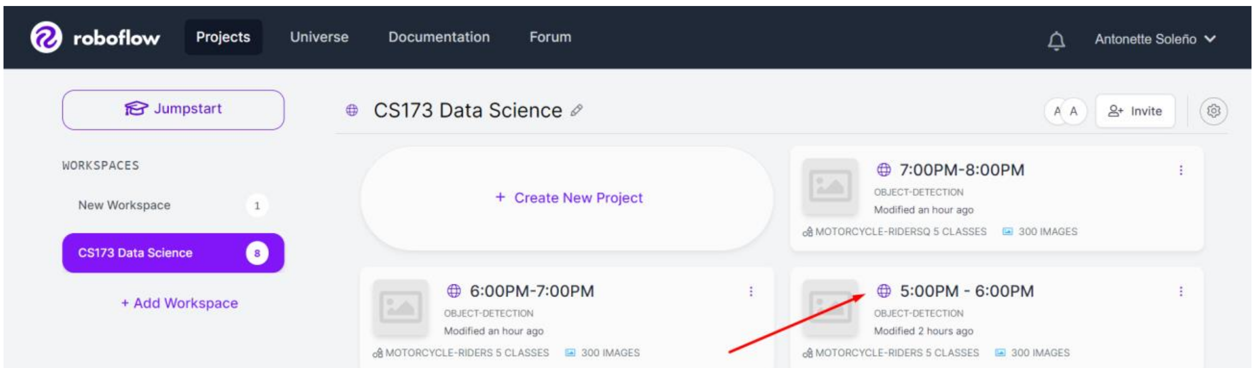
These same steps should be repeated for the two other time scenarios, 6:00PM-7:00PM and 7:00PM-8:00PM.

IV. Ground Truth of Test Dataset

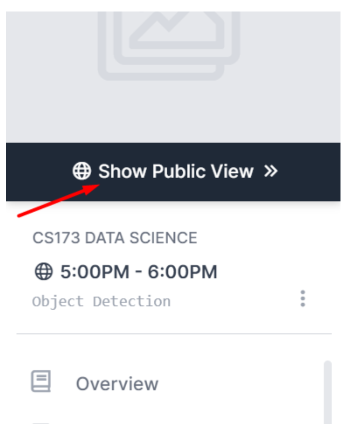
The next step after generating three versions of the original dataset in terms of their corresponding time scenarios is to check the ground truth of the versions' test datasets by counting how many motorcycle riders using and not using full-faced and half-faced helmets (invalid) and not wearing helmets there are. These results will be compared (validated) to the model output that was accomplished in the previous assignments.

To expedite the process of manually counting all classes within the test dataset, we used Excel Spreadsheets to generate the total of all classes per time scenario. To do this, the details of each annotated image must be accessed in order to know the number of classes per image.

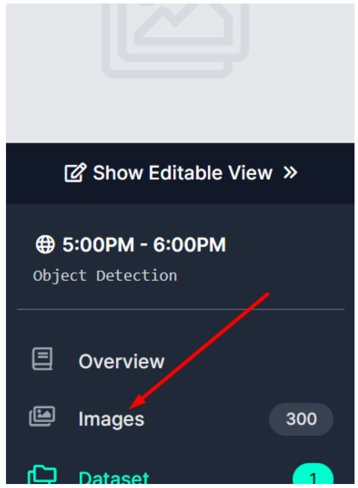
The example that will be used to demonstrate the manual counting of classes is the 5:00PM – 6:00PM time scenario project. First, select the appropriate project in the *CS173 Data Science* Workspace.



Then, select *Show Public View* on the left menu to access the Public View of the dataset.

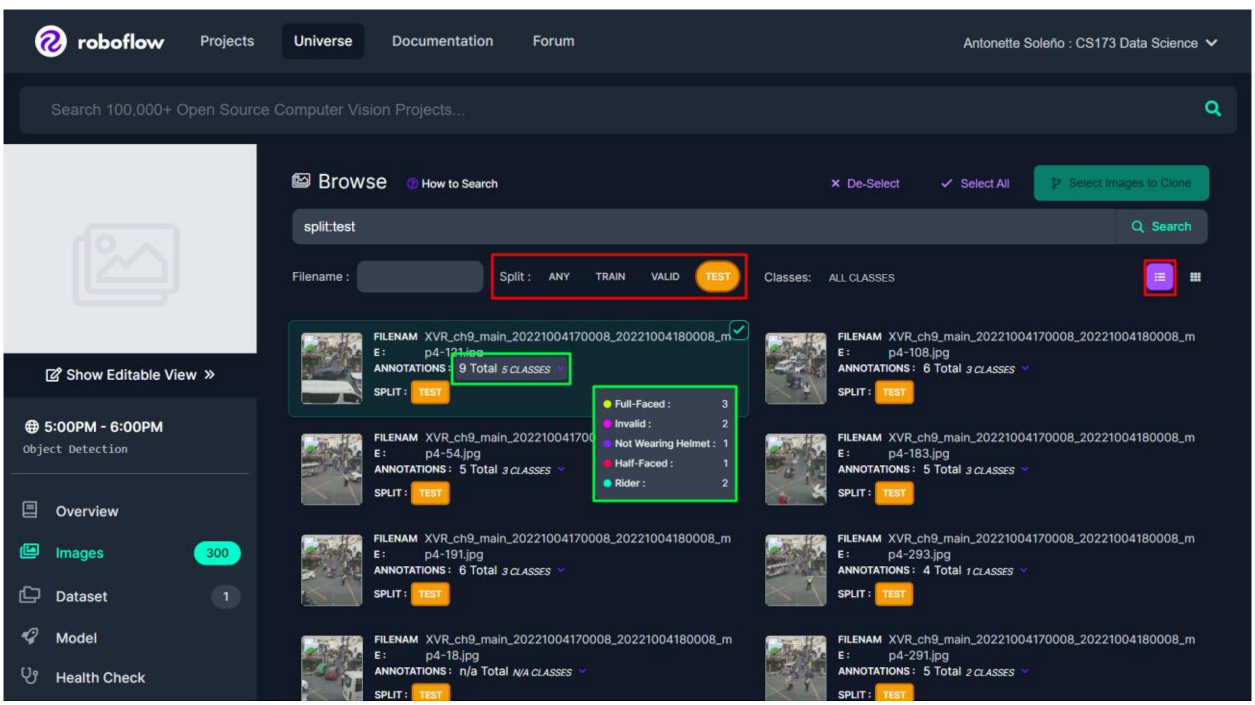


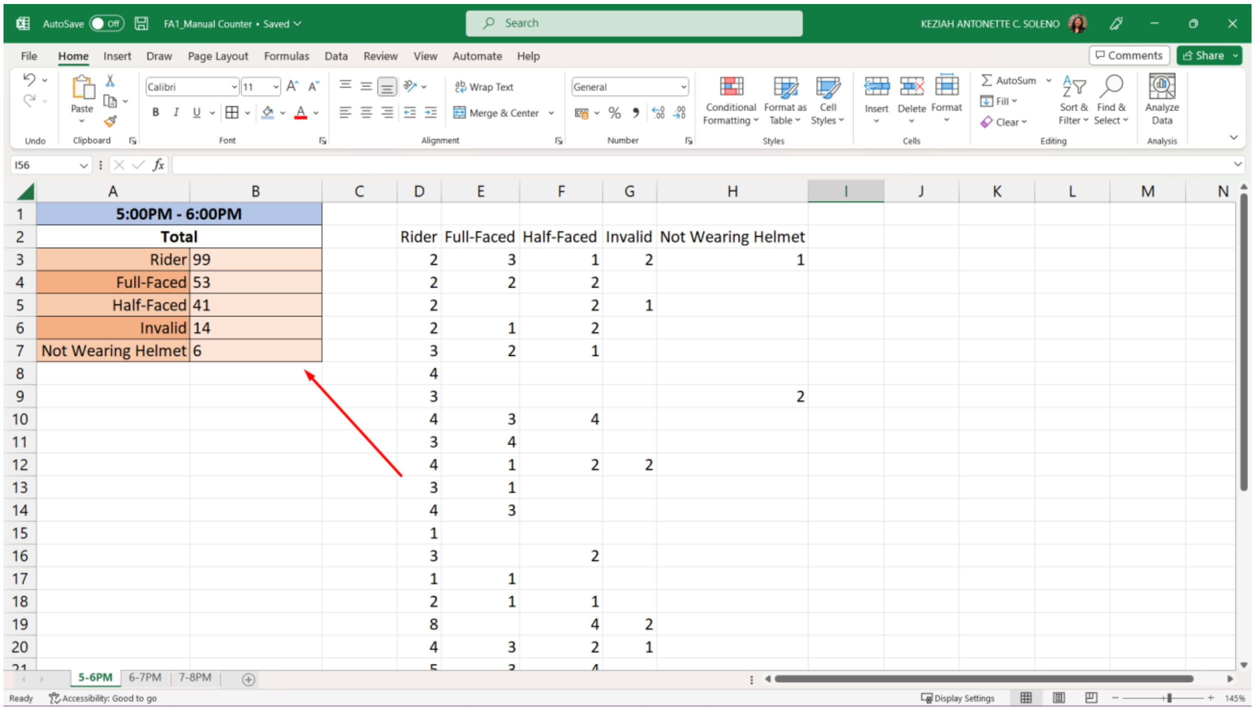
On the Public View menu on the left, select the *Images* option. Then, select the *TEST* option on the *Split :* filter to see the Test images that were generated by Roboflow and enable the list view, as indicated in the succeeding figure (highlighted in red on the upper right corner).



The information that appears (highlighted in green) when the cursor hovers over the classes drop-down will be used to generate the sum of each class per time scenario using the SUM() function of Excel, as indicated in the succeeding figure.

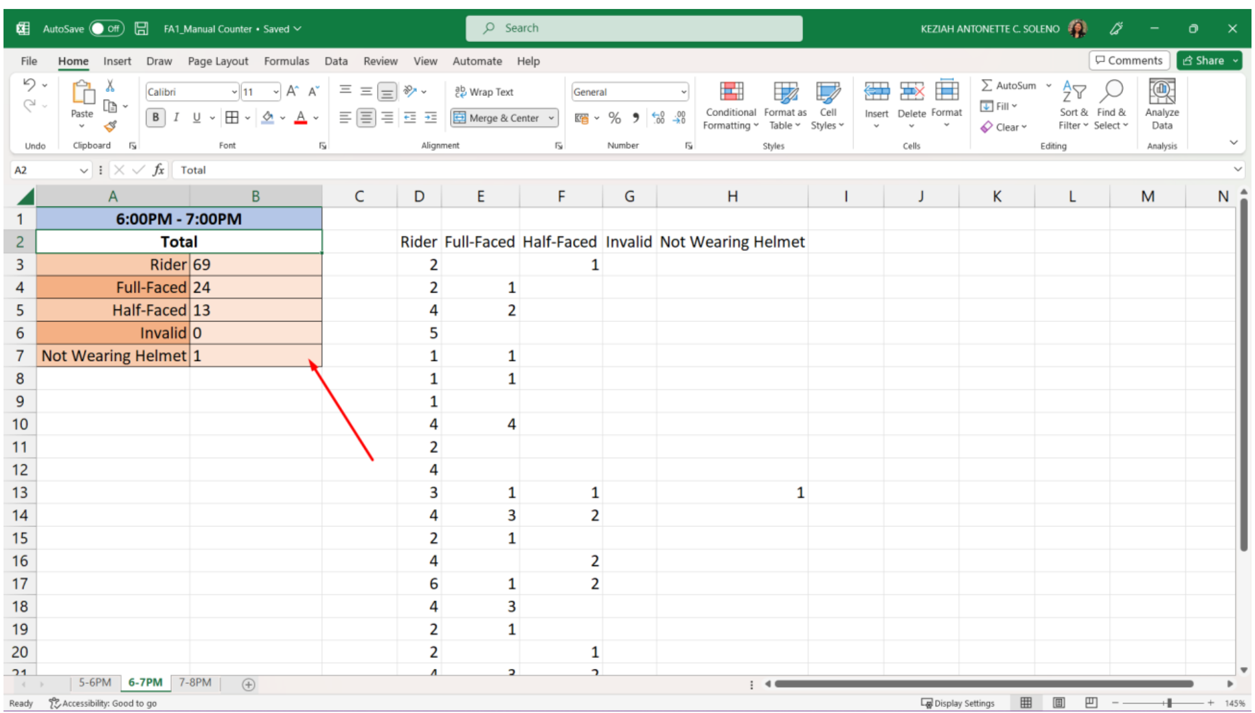
Note: This process was repeated twice, one for each of the remaining time scenarios.





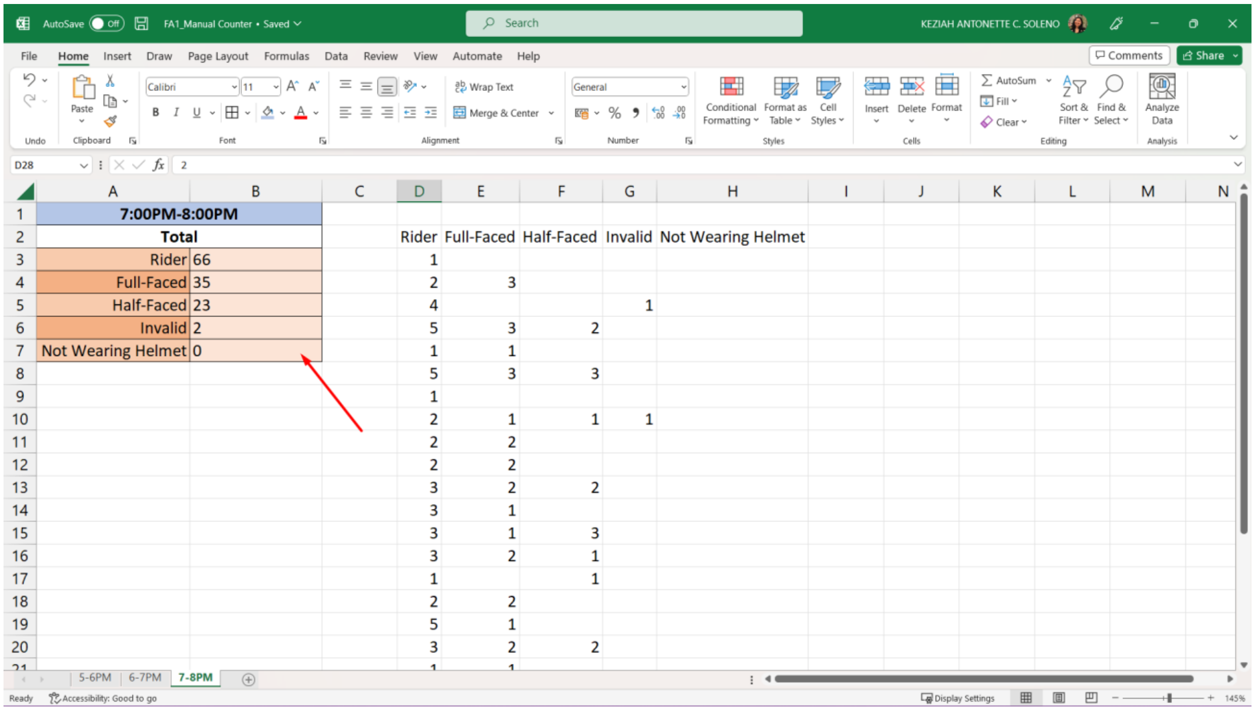
5:00PM - 6:00PM	
Total	
Rider	99
Full-Faced	53
Half-Faced	41
Invalid	14
Not Wearing Helmet	6

Rider	Full-Faced	Half-Faced	Invalid	Not Wearing Helmet
2	3	1	2	1
2	2	2		
2		2	1	
2	1	2		
3	2	1		
4				
3				2
4	3	4		
3	4			
4	1	2	2	
3	1			
4	3			
1				
3		2		
1	1			
2	1	1		
8		4	2	
4	3	2	1	



6:00PM - 7:00PM	
Total	
Rider	69
Full-Faced	24
Half-Faced	13
Invalid	0
Not Wearing Helmet	1

Rider	Full-Faced	Half-Faced	Invalid	Not Wearing Helmet
2		1		
2	1			
4	2			
5				
1	1			
1	1			
1				
4	4			
2				
4				
3	1	1		1
4	3	2		
2	1			
4		2		
6	1	2		
4	3			
2	1			
2		1		



The following tables contain the manually-counted sum (Ground Truth) of each class present in each time scenario.

5:00PM - 6:00PM	
Total	
Rider	99
Full-Faced	53
Half-Faced	41
Invalid	14
Not Wearing Helmet	6

6:00PM - 7:00PM	
Total	
Rider	69
Full-Faced	24
Half-Faced	13
Invalid	0
Not Wearing Helmet	1

7:00PM-8:00PM	
Total	
Rider	66
Full-Faced	35
Half-Faced	23
Invalid	2
Not Wearing Helmet	0

V. Libraries and Dependencies Installation and Environment Set-up

The code that was used in the previous assignment (M1 – SA: YOLOv5 Modeling) will be used to produce the YOLOv5 model output of the three time scenarios. Therefore, the same libraries and dependencies will be installed, as indicated in the following code snippets and their corresponding outputs.

```
#clone forked YOLOv5 from Ultralytics YOLOv5 Repository
!git clone https://github.com/ultralytics/yolov5  # clone repo
%cd yolov5
%pip install -qr requirements.txt # install dependencies
%pip install -q roboflow # install roboflow

import torch
import os
from IPython.display import Image, clear_output  # to display images

print(f"Setup complete. Using torch {torch.__version__} ({torch.cuda.g
et_device_properties(0).name if torch.cuda.is_available() else 'CPU'})
")
```

Output:

```
Cloning into 'yolov5'...
remote: Enumerating objects: 14927, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 14927 (delta 7), reused 13 (delta 3), pack-reused 14908
Receiving objects: 100% (14927/14927), 14.01 MiB | 17.34 MiB/s, done.
Resolving deltas: 100% (10246/10246), done.
/content/yolov5
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 184.0/184.0 KB 15.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.7/62.7 KB 8.7 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 71.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.7/45.7 KB 5.3 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 138.5/138.5 KB 17.2 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.8/67.8 KB 9.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.5/54.5 KB 6.8 MB/s eta 0:00:00
  Building wheel for wget (setup.py) ... done
Setup complete. Using torch 1.13.0+cu116 (Tesla T4)
```

Running the code snippet below should result in a text message prompting the programmer to retrieve an API KEY from the provided link in its output.

```
from roboflow import Roboflow #import roboflow for our datasets
rf = Roboflow(model_format="yolov5", notebook="ultralytics")
```

```
[2] from roboflow import Roboflow #import roboflow for our datasets
    rf = Roboflow(model_format="yolov5", notebook="ultralytics")

    upload and label your dataset, and get an API KEY here: https://app.roboflow.com/?model=yolov5&ref=ultralytics
```

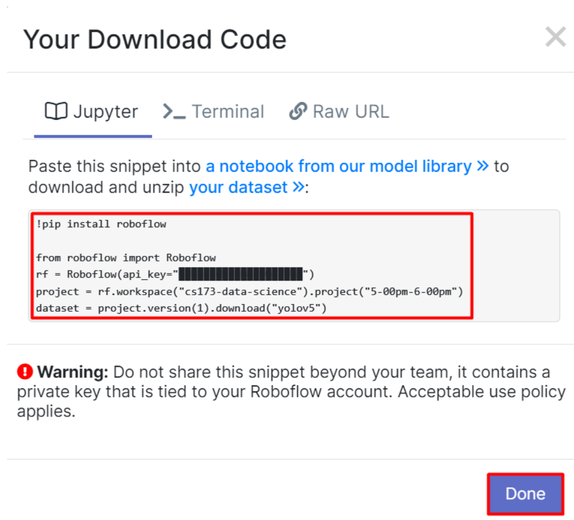
Note: Since a download code has already been created and saved for each time scenario project, there will be no need to follow the link to retrieve an API key. If you still do not have an API key, follow the instructions under the section **III. Generating YOLOv5 Versions of Each Time Scenario**.

Then, to set up the environment, run the code snippet below.

```
# set up environment
os.environ["DATASET_DIRECTORY"] = "/content/datasets"
```

VI. Downloading the New Datasets using Roboflow's Download Code

To download and unzip the new datasets, paste the download code that Roboflow previously provided for each time scenario dataset on a new Python cell and run it.



Running the following code snippet will download and unzip the dataset (5:00PM-6:00PM).

```
# 5:00-6:00
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="[redacted]")
project = rf.workspace("cs173-data-science").project("5-00pm-6-00pm")
dataset = project.version(1).download("yolov5")
```

Output:

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: roboflow in /usr/local/lib/python3.8/dist-packages (0.2.22)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.8/dist-packages (from roboflow) (3.2.2)
Requirement already satisfied: chardet==4.0.0 in /usr/local/lib/python3.8/dist-packages (from roboflow) (4.0.0)
Requirement already satisfied: cycler==0.10.0 in /usr/local/lib/python3.8/dist-packages (from roboflow) (0.10.0)
Requirement already satisfied: requests in /usr/local/lib/python3.8/dist-packages (from roboflow) (2.25.1)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.8/dist-packages (from roboflow) (0.21.0)
Requirement already satisfied: pyparsing==2.4.7 in /usr/local/lib/python3.8/dist-packages (from roboflow) (2.4.7)
Requirement already satisfied: requests-toolbelt in /usr/local/lib/python3.8/dist-packages (from roboflow) (0.10.1)
Requirement already satisfied: six in /usr/local/lib/python3.8/dist-packages (from roboflow) (1.15.0)
Requirement already satisfied: glob2 in /usr/local/lib/python3.8/dist-packages (from roboflow) (0.7)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.8/dist-packages (from roboflow) (1.4.4)
Requirement already satisfied: opencv-python-headless>=4.5.1.48 in /usr/local/lib/python3.8/dist-packages (from roboflow) (4.6.0.66)
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.8/dist-packages (from roboflow) (4.64.1)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.8/dist-packages (from roboflow) (6.0)
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.8/dist-packages (from roboflow) (7.1.2)
Requirement already satisfied: certifi==2022.12.7 in /usr/local/lib/python3.8/dist-packages (from roboflow) (2022.12.7)
Requirement already satisfied: wget in /usr/local/lib/python3.8/dist-packages (from roboflow) (3.2)
Requirement already satisfied: urllib3==1.26.6 in /usr/local/lib/python3.8/dist-packages (from roboflow) (1.26.6)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.8/dist-packages (from roboflow) (1.21.6)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.8/dist-packages (from roboflow) (2.8.2)
Requirement already satisfied: idna==2.10 in /usr/local/lib/python3.8/dist-packages (from roboflow) (2.10)
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in /content/datasets/5:00PM---6:00PM-1 to yolov5pytorch: 100% [44609948 / 44609948] bytes
Extracting Dataset Version Zip to /content/datasets/5:00PM---6:00PM-1 in yolov5pytorch:: 100%|██████████| 1432/1432 [00:00<00:00, 2224.08it/s]
```

VII. Training the Model per Scale and Time Scenario

After all the dependencies have been installed and environments have been set up, the YOLOv5 model will be trained using the *train.py* file from the Ultralytics repository that was cloned earlier. Since the code from the previous assignment is being recycled, the same hyperparameters will also be used for this assignment.

1. Image size = 640 pixels
2. Batch size = 16
3. Epochs = 50
4. YOLOv5 Version = YOLOv5**s*** (Small)

Note: *The model version scale will vary among small, medium, and large, given that the outputs of the three version scales of each time scenario must be provided. Since the process for each scale model will be the same for other scale models, only one YOLOv5 model will be discussed in this section.

To apply the hyperparameters previously specified, the code snippet below was run, producing results as seen in the succeeding screenshots.

```
!python train.py --img 640 --batch 16 --epochs 50 --
data {dataset.location}/data.yaml --weights yolov5s.pt --
name yolov5s_results --cache
```

```
50 epochs completed in 0.191 hours.
Optimizer stripped from runs/train/yolov5s_results4/weights/last.pt, 14.5MB
Optimizer stripped from runs/train/yolov5s_results4/weights/best.pt, 14.5MB

Validating runs/train/yolov5s_results4/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.15it/s]
                   all         62        436      0.733      0.283       0.29      0.141
           Full-Faced         62        109      0.392      0.404      0.361      0.113
           Half-Faced         62         86      0.385       0.35      0.276     0.0839
              Invalid         62         22          1          0     0.0179    0.00713
    Not Wearing Helmet         62         26          1          0    0.00342    0.00157
                 Rider         62        193      0.885      0.663       0.79      0.499
Results saved to runs/train/yolov5s_results4
```

5:00PM-6:00PM (Small)

```
50 epochs completed in 0.245 hours.
Optimizer stripped from runs/train/yolov5m_results/weights/last.pt, 42.2MB
Optimizer stripped from runs/train/yolov5m_results/weights/best.pt, 42.2MB

Validating runs/train/yolov5m_results/weights/best.pt...
Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.08it/s]
                   all         62        436      0.748      0.318      0.328      0.163
           Full-Faced         62        109      0.583      0.505      0.504      0.153
           Half-Faced         62         86      0.287      0.349      0.236     0.0884
              Invalid         62         22          1          0     0.0586     0.0223
    Not Wearing Helmet         62         26          1          0    0.00623     0.0026
                 Rider         62        193      0.871      0.738      0.836       0.55
Results saved to runs/train/yolov5m_results
```

5:00PM-6:00PM (Medium)

```
50 epochs completed in 0.363 hours.
Optimizer stripped from runs/train/yolov5l_results/weights/last.pt, 92.9MB
Optimizer stripped from runs/train/yolov5l_results/weights/best.pt, 92.9MB

Validating runs/train/yolov5l_results/weights/best.pt...
Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.10s/it]
                   all         62        436      0.775      0.348      0.359      0.176
           Full-Faced         62        109      0.624      0.532      0.604      0.207
           Half-Faced         62         86      0.348      0.496      0.322      0.103
              Invalid         62         22          1          0     0.0546     0.0235
    Not Wearing Helmet         62         26          1          0     0.0113    0.00303
                 Rider         62        193      0.902      0.712      0.801      0.545
Results saved to runs/train/yolov5l_results
```

5:00PM-6:00PM (Large)

```
50 epochs completed in 0.165 hours.
Optimizer stripped from runs/train/yolov5s_results/weights/last.pt, 14.5MB
Optimizer stripped from runs/train/yolov5s_results/weights/best.pt, 14.5MB

Validating runs/train/yolov5s_results/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.75it/s]
                   all         56        158      0.851      0.335      0.407      0.196
           Full-Faced         56         32      0.927      0.399      0.568       0.22
           Half-Faced         56         18      0.443      0.556      0.531      0.235
              Invalid         56          1          1          0     0.0149    0.00555
    Not Wearing Helmet         56          1          1          0      0.111     0.0553
                 Rider         56        106      0.884      0.722      0.809      0.465
Results saved to runs/train/yolov5s_results
```

6:00PM-7:00PM (Small)

```
50 epochs completed in 0.246 hours.
Optimizer stripped from runs/train/yolov5m_results/weights/last.pt, 42.2MB
Optimizer stripped from runs/train/yolov5m_results/weights/best.pt, 42.2MB

Validating runs/train/yolov5m_results/weights/best.pt...
Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.37it/s]
                   all         56        158      0.863       0.37      0.648      0.359
            Full-Faced         56         32      0.663        0.5      0.514      0.215
            Half-Faced         56         18      0.786      0.614      0.704      0.328
               Invalid         56          1          1          0      0.199     0.0796
    Not Wearing Helmet         56          1          1          0      0.995      0.697
                 Rider         56        106      0.867      0.735      0.827      0.475
Results saved to runs/train/yolov5m_results
```

6:00PM-7:00PM (Medium)

```
50 epochs completed in 0.403 hours.
Optimizer stripped from runs/train/yolov5l_results/weights/last.pt, 92.9MB
Optimizer stripped from runs/train/yolov5l_results/weights/best.pt, 92.9MB

Validating runs/train/yolov5l_results/weights/best.pt...
Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.33it/s]
                   all         56        158      0.635       0.59      0.781      0.358
            Full-Faced         56         32       0.52      0.594      0.541      0.221
            Half-Faced         56         18      0.389      0.556      0.534      0.275
               Invalid         56          1      0.418          1      0.995      0.497
    Not Wearing Helmet         56          1          1          0      0.995      0.298
                 Rider         56        106       0.85      0.802      0.839      0.499
Results saved to runs/train/yolov5l_results
```

6:00PM-7:00PM (Large)

```
50 epochs completed in 0.162 hours.
Optimizer stripped from runs/train/yolov5s_results/weights/last.pt, 14.5MB
Optimizer stripped from runs/train/yolov5s_results/weights/best.pt, 14.5MB

Validating runs/train/yolov5s_results/weights/best.pt...
Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.62it/s]
                   all         62        208      0.711      0.319      0.302      0.145
            Full-Faced         62         37      0.413      0.432       0.33      0.114
            Half-Faced         62         54      0.371      0.426      0.329     0.0947
               Invalid         62          4          1          0     0.0591     0.0256
    Not Wearing Helmet         62          2          1          0          0          0
                 Rider         62        111       0.77      0.739      0.793      0.491
Results saved to runs/train/yolov5s_results
```

7:00PM-8:00PM (Small)

```
50 epochs completed in 0.251 hours.
Optimizer stripped from runs/train/yolov5m_results/weights/last.pt, 42.2MB
Optimizer stripped from runs/train/yolov5m_results/weights/best.pt, 42.2MB

Validating runs/train/yolov5m_results/weights/best.pt...
Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.06s/it]
                   all         62        208      0.765       0.33      0.333      0.167
            Full-Faced         62         37      0.466      0.541      0.376        0.1
            Half-Faced         62         54      0.508      0.333      0.354      0.117
               Invalid         62          4          1          0     0.0967     0.0694
    Not Wearing Helmet         62          2          1          0          0          0
                 Rider         62        111      0.851      0.775      0.839      0.548
Results saved to runs/train/yolov5m_results
```

7:00PM-8:00PM (Medium)

```
50 epochs completed in 0.391 hours.
Optimizer stripped from runs/train/yolov5l_results/weights/last.pt, 92.9MB
Optimizer stripped from runs/train/yolov5l_results/weights/best.pt, 92.9MB

Validating runs/train/yolov5l_results/weights/best.pt...
Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:01<00:00,  1.15it/s]
                   all         62        208      0.634      0.519      0.473      0.213
            Full-Faced         62         37      0.464      0.676      0.525      0.161
            Half-Faced         62         54        0.3      0.635      0.538       0.16
               Invalid         62          4      0.618      0.427      0.438      0.181
    Not Wearing Helmet         62          2          1          0    0.00268   0.000803
                 Rider         62        111      0.788      0.856      0.861      0.561
Results saved to runs/train/yolov5l_results
```

7:00PM-8:00PM (Large)

VIII. TensorBoard

To portray visualizations of the training process results, TensorBoard was used. This feature will create graphic visualizations of the data through graphs and confusion matrices. To launch TensorBoard, the following code was run.

```
%load_ext tensorboard
%tensorboard --logdir runs
```

Running the code to launch TensorBoard will produce the visualizations required in this assignment: confusion matrices. (These graphs can be viewed in a separate document here.

IX. Results

This section contains the real precision and recall results after validating each model.

```
val: data=/content/datasets/5:00PM---6:00PM-1/data.yaml, weights=['./runs/train/yolov5s_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
val: Scanning /content/datasets/5:00PM---6:00PM-1/valid/labels.cache... 62 images, 0 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.47s/it]
                   all         62        436      0.732      0.282      0.289       0.14
            Full-Faced         62        109      0.383      0.394      0.352      0.108
            Half-Faced         62         86       0.39       0.35       0.28     0.0836
               Invalid         62         22          1          0     0.0179    0.00717
    Not Wearing Helmet         62         26          1          0    0.00349    0.00159
                 Rider         62        193      0.885      0.663       0.79      0.498
Speed: 0.3ms pre-process, 11.5ms inference, 5.3ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp4
```

5:00PM-6:00PM (Small)

```
val: data=/content/datasets/5:00PM---6:00PM-1/data.yaml, weights=['./runs/train/yolov5m_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
val: Scanning /content/datasets/5:00PM---6:00PM-1/valid/labels.cache... 62 images, 0 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:03<00:00,  1.61s/it]
                   all         62        436      0.759      0.311      0.329      0.163
            Full-Faced         62        109      0.619      0.495      0.505      0.152
            Half-Faced         62         86      0.301      0.326      0.238     0.0894
               Invalid         62         22          1          0     0.0596     0.0222
    Not Wearing Helmet         62         26          1          0    0.00632    0.00262
                 Rider         62        193      0.876      0.732      0.836      0.548
Speed: 1.4ms pre-process, 18.0ms inference, 2.7ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

5:00PM-6:00PM (Medium)

```
val: data=/content/datasets/5:00PM---6:00PM-1/data.yaml, weights=['./runs/train/yolov5l_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
val: Scanning /content/datasets/5:00PM---6:00PM-1/valid/labels.cache... 62 images, 0 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:03<00:00,  1.88s/it]
                   all         62        436      0.772      0.344      0.355      0.176
            Full-Faced         62        109      0.624      0.532      0.604      0.207
            Half-Faced         62         86      0.335      0.474      0.302     0.0999
               Invalid         62         22          1          0     0.0545     0.0227
    Not Wearing Helmet         62         26          1          0     0.0109    0.00289
                 Rider         62        193      0.902      0.713      0.801      0.545
Speed: 0.3ms pre-process, 29.4ms inference, 3.5ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

5:00PM-6:00PM (Large)

```
val: data=/content/datasets/6:00PM-7:00PM-1/data.yaml, weights=['./runs/train/yolov5s_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
val: Scanning /content/datasets/6:00PM-7:00PM-1/valid/labels.cache... 56 images, 6 backgrounds, 0 corrupt: 100% 56/56 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.20s/it]
                   all         56        158      0.851      0.336      0.408      0.196
            Full-Faced         56         32      0.927      0.399      0.569      0.218
            Half-Faced         56         18      0.444      0.556      0.535      0.233
               Invalid         56          1          1          0     0.0149    0.00557
    Not Wearing Helmet         56          1          1          0      0.111     0.0553
                 Rider         56        106      0.885      0.724      0.809      0.467
Speed: 0.3ms pre-process, 12.3ms inference, 3.0ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

6:00PM-7:00PM (Small)

```
val: data=/content/datasets/6:00PM-7:00PM-1/data.yaml, weights=['./runs/train/yolov5m_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
val: Scanning /content/datasets/6:00PM-7:00PM-1/valid/labels.cache... 56 images, 6 backgrounds, 0 corrupt: 100% 56/56 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.32s/it]
                   all         56        158      0.871      0.376      0.656       0.36
            Full-Faced         56         32      0.704      0.531      0.555      0.221
            Half-Faced         56         18      0.786      0.614      0.705      0.326
               Invalid         56          1          1          0      0.199     0.0796
    Not Wearing Helmet         56          1          1          0      0.995      0.697
                 Rider         56        106      0.866      0.735      0.827      0.476
Speed: 0.2ms pre-process, 21.3ms inference, 3.4ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

6:00PM-7:00PM (Medium)

```
val: data=/content/datasets/6:00PM-7:00PM-1/data.yaml, weights=['./runs/train/yolov5l_results/weights/best.pt'], batch_size=32, imgsz=6
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
val: Scanning /content/datasets/6:00PM-7:00PM-1/valid/labels.cache... 56 images, 6 backgrounds, 0 corrupt: 100% 56/56 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:03<00:00,  1.92s/it]
                   all         56        158       0.63       0.59      0.781      0.359
             Full-Faced         56         32      0.508      0.594       0.54      0.221
             Half-Faced         56         18      0.388      0.556      0.535      0.276
               Invalid         56          1      0.403          1      0.995      0.497
     Not Wearing Helmet         56          1          1          0      0.995      0.298
                 Rider         56        106       0.85      0.801      0.842      0.502
Speed: 0.3ms pre-process, 33.6ms inference, 3.1ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

6:00PM-7:00PM (Large)

```
val: data=/content/datasets/7:00PM-8:00PM-1/data.yaml, weights=['./runs/train/yolov5s_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 157 layers, 7023610 parameters, 0 gradients, 15.8 GFLOPs
val: Scanning /content/datasets/7:00PM-8:00PM-1/valid/labels.cache... 62 images, 11 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:02<00:00,  1.19s/it]
                   all         62        208      0.714      0.319      0.303      0.145
             Full-Faced         62         37      0.425      0.432       0.33      0.111
             Half-Faced         62         54      0.378      0.426      0.334     0.0927
               Invalid         62          4          1          0     0.0596     0.0257
     Not Wearing Helmet         62          2          1          0          0          0
                 Rider         62        111      0.769      0.739      0.794      0.494
Speed: 0.2ms pre-process, 9.7ms inference, 2.7ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

7:00PM-8:00PM (Small)

```
val: data=/content/datasets/7:00PM-8:00PM-1/data.yaml, weights=['./runs/train/yolov5m_results/weights/best.pt'], batch_size=32, imgsz=64
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 212 layers, 20869098 parameters, 0 gradients, 47.9 GFLOPs
val: Scanning /content/datasets/7:00PM-8:00PM-1/valid/labels.cache... 62 images, 11 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:03<00:00,  1.51s/it]
                   all         62        208      0.749       0.34      0.328      0.166
             Full-Faced         62         37      0.442      0.557      0.376      0.102
             Half-Faced         62         54      0.461       0.37      0.326      0.112
               Invalid         62          4          1          0     0.0967     0.0695
     Not Wearing Helmet         62          2          1          0          0          0
                 Rider         62        111      0.843      0.775      0.839      0.547
Speed: 0.2ms pre-process, 19.7ms inference, 3.5ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

7:00PM-8:00PM (Medium)

```
val: data=/content/datasets/7:00PM-8:00PM-1/data.yaml, weights=['./runs/train/yolov5l_results/weights/best.pt'], batch_size=32, imgsz=640,
YOLOv5 🚀 v7.0-56-gc0ca1d2 Python-3.8.16 torch-1.13.0+cu116 CUDA:0 (Tesla T4, 15110MiB)

Fusing layers...
Model summary: 267 layers, 46129818 parameters, 0 gradients, 107.7 GFLOPs
val: Scanning /content/datasets/7:00PM-8:00PM-1/valid/labels.cache... 62 images, 11 backgrounds, 0 corrupt: 100% 62/62 [00:00<?, ?it/s]
                 Class     Images  Instances          P          R      mAP50   mAP50-95: 100% 2/2 [00:03<00:00,  1.79s/it]
                   all         62        208      0.634      0.514      0.473      0.213
             Full-Faced         62         37      0.473      0.676      0.525      0.159
             Half-Faced         62         54      0.304       0.63      0.538      0.163
               Invalid         62          4      0.606      0.408      0.438      0.181
     Not Wearing Helmet         62          2          1          0    0.00266   0.000797
                 Rider         62        111      0.788      0.856      0.861      0.561
Speed: 0.2ms pre-process, 33.4ms inference, 3.3ms NMS per image at shape (32, 3, 640, 640)
Results saved to runs/val/exp
```

7:00PM-8:00PM (Large)

X. Validation of Model Output vs. Ground Truth of Test Dataset

The main objective of this assignment is to compare the results produced by manually counting the number of present classes per time scenario and the validation results of each time scenario in small, medium, and large scaling. After the original dataset had been segregated into three divisions: 5:00PM-6:00PM, 6:00PM-7:00PM, 7:00PM-8:00PM, each class (Rider, Full-Faced, Half-Faced, Invalid, and Not Wearing Helmet) was manually calculated to produce each of their totals. These calculated sums per hour can be seen in the tables in the section **IV. Ground Truth of Test Dataset**. On the other hand, the validation results of the YOLOv5 model output can be viewed in the figures under the previous section.

IX. Results.

The actual (manually-counted) results or ground truth of the 5:00PM – 6:00PM scenario says that there are 99 riders, 53 riders wearing full-faced helmets, 41 riders wearing half-faced helmets, 14 riders wearing invalid helmets, and 6 riders not wearing helmets. The validation results of the 5:00PM – 6:00PM hour state that there were 193 riders, 109 riders wearing full-faced helmets, 86 riders wearing half-faced helmets, 22 riders wearing invalid helmets, and 28 riders not wearing helmets. One observation between the ground truth and validation values is that there is a small difference between the number of riders identified by the model and the programmers (ground truth). This similarity between values is further validated by a precision rate of 88.50%, implying that the model used was that close to identifying each rider from the dataset correctly. A high precision rate for this class makes sense given that most of the entities that were identified in the dataset were riders. However, the rest of the other classes do not portray a similar likeness with each other's values. This result is understandable, given that the rest of the other classes either rarely appeared in the images or objects were too blurry to identify as a class.

As for the results for both instances of the 6:00PM – 7:00PM hour, the similarity between them is more promising than the previous hour, given that the value of the Not Wearing Helmet class is the same for both (100% precision rate) and the Half-Faced and Full-Faced classes have a small difference between their values implying that the model fulfilled a proper job of identifying objects.

The last hour, 7:00PM – 8:00PM, only portrays one similarity between values: Full-Faced class. Although its precision rate is not optimal, their values are similar given that there is only a small difference between them. This observation can lead to the conclusion that more Full-Faced classes were identified in the original dataset and that the model was able to learn how to identify these objects over time.

Although the results may not be optimal, one can say that the YOLOv5 model is still effective in identifying objects within a given dataset, as long as many instances of the classes are provided to the model. Furthermore, increasing the number of instances to be observed by the YOLOv5 model will enable more accurate and better results in the future.

References

C3.ai. (2022, March 15). *Ground Truth*. C3 AI. https://c3.ai/glossary/machine-learning/ground-truth/

Ground Truth in Machine Learning: Process & Key Challenges. (2023, January 4). Datagen. https://datagen.tech/guides/data-training/ground-truth/

Nelson, J. (2022, November 1). *How to Create a Train Test Split*. Roboflow Blog. https://blog.roboflow.com/train-test-split-with-roboflow/