Assignment # 5: Hugging Face AutoTrain
Course: CAP 6610 Applied Machine Learning
Professor: Dr. Oge Marques
Student Number: Z23596812
Student Name: Renee Raven
Term: Fall 2022

Hugging face has created an easy to use, intuitive AutoTrain platform for code-free construction of machine learning models. Drop down buttons allow users to create a new project and select a task. Users can select an existing dataset in the hugging face library or upload their data in select formats. By far, this my favorite no-code/low-code tool introduced during this class.

The limits for the free version include 3000 rows, 500 images, and 5 models per project. That still allows plenty of experimentation.

Project 1: Tabular data classification (binary) with scikit-learn/breast-cancer-Wisconsin dataset

Following the instructions for creating the first model, a tabular data classification model, flowed effortlessly. The models took less than 20 minutes to run. Five models were trained, the metrics examined for each model and the CO2 emissions (in grams) noted for each model. Generally, CO2 emissions correlate with training time, but not always. The best 2 models offered impressive metrics (accuracy of 0.9826 and F1 of 0.9762).

Table 1. Results for Model 1 of Project 1

	Model	Minutes	CO2(gms)	Loss	Accuracy	Recall	AUC	F1
Tabular Data Binary Classification	simplistic-wren	4	2.6165	0.0424	0.9826	0.953	1	0.9762
	fancy-rabbit	1	0.0006	0.075	0.983	0.953	0.999	0.976
	jumbo-skunk	10	0.0163	0.091	0.991	0.977	1	0.988
	negative-aardvark	17	6.5978	0.094	0.974	0.93	0.999	0.964
	bland-rabbit	1	0.0006	0.3	0.991	1	0.993	0.989

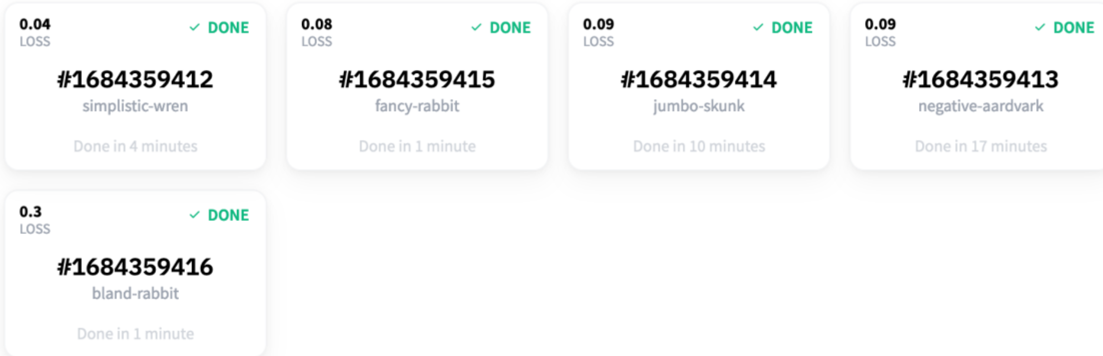
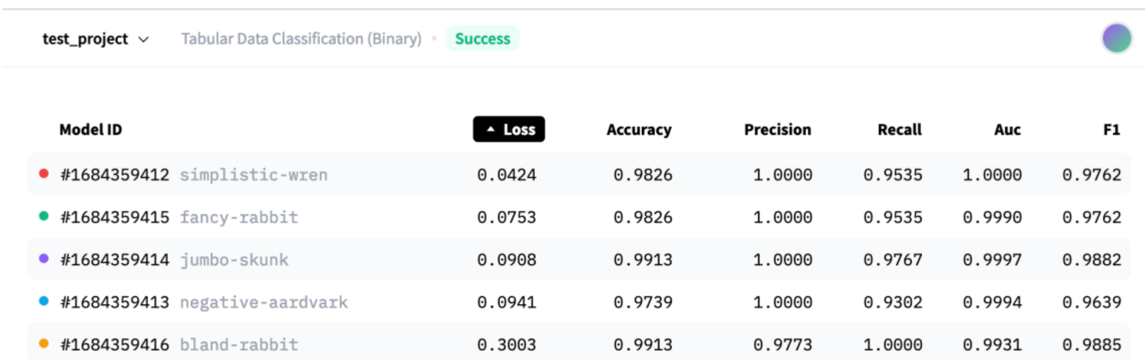


Figure 1. Results of Training for Project 1


test_project · Tabular Data Classification (Binary) · Success

Model ID	Loss	Accuracy	Precision	Recall	Auc	F1
#1684359412 simplistic-wren	0.0424	0.9826	1.0000	0.9535	1.0000	0.9762
#1684359415 fancy-rabbit	0.0753	0.9826	1.0000	0.9535	0.9990	0.9762
#1684359414 jumbo-skunk	0.0908	0.9913	1.0000	0.9767	0.9997	0.9882
#1684359413 negative-aardvark	0.0941	0.9739	1.0000	0.9302	0.9994	0.9639
#1684359416 bland-rabbit	0.3003	0.9913	0.9773	1.0000	0.9931	0.9885

Figure 2. Metrics for 5 Models from Project 1

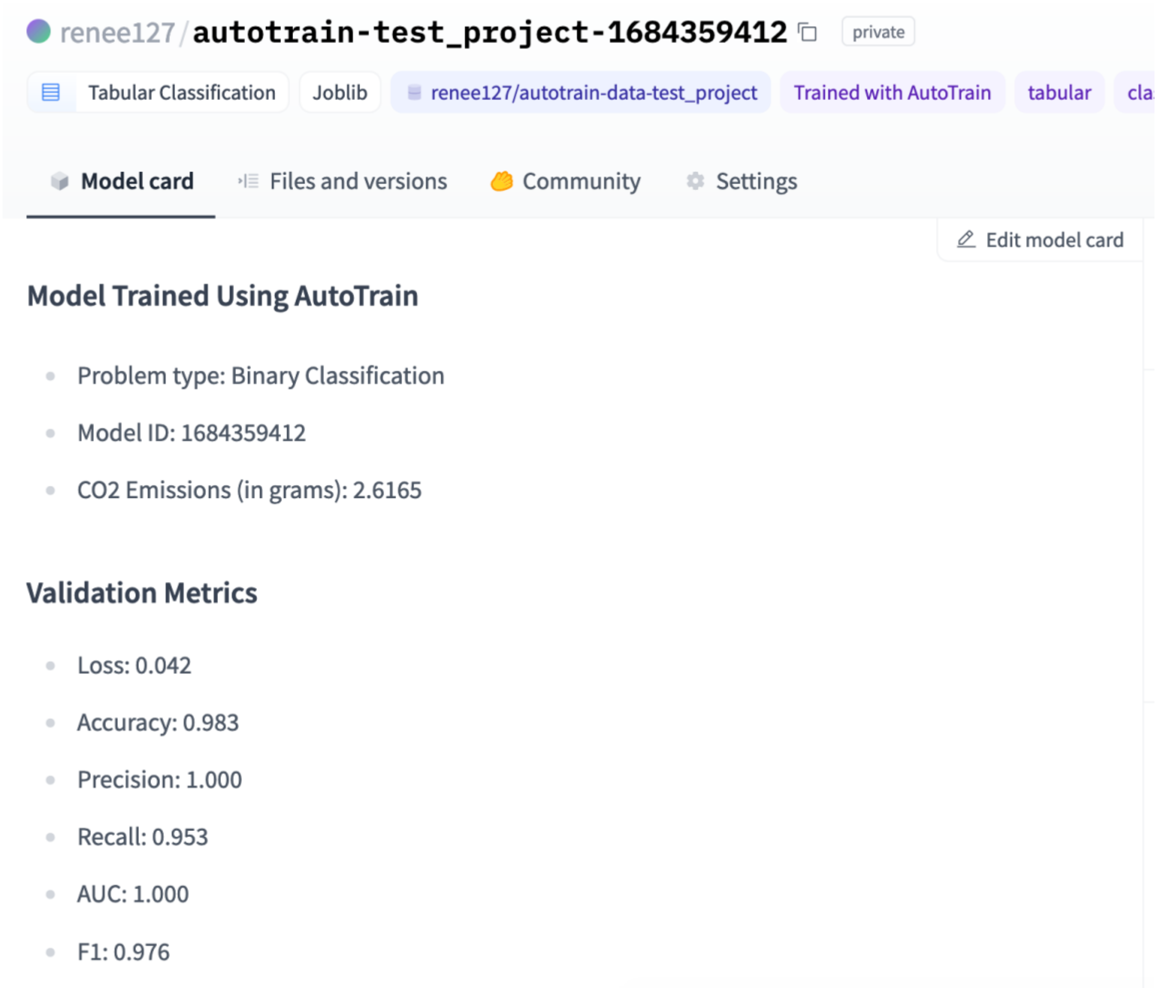


Figure 3. Details for Model 1 of Project 1

Project 2: Text Classification (Multi-class) with poem-sentiment dataset

I selected the poem-sentiment dataset because it has a relatively small number of rows and therefore would demand less energy use. Additionally, it offered a multi-class model (positive, negative, mixed, or no impact) and already is divided into train, validation, and test sets.

Dataset Structure

Data Instances

Example of one instance in the dataset.

```
{'id': 0, 'label': 2, 'verse_text': 'with pale blue berries. in these peaceful
shades--'}
```

Data Fields

- `id`: index of the example
- `verse_text`: The text of the poem verse
- `label`: The sentiment label. Here
  - 0 = negative
  - 1 = positive
  - 2 = no impact
  - 3 = mixed (both negative and positive)

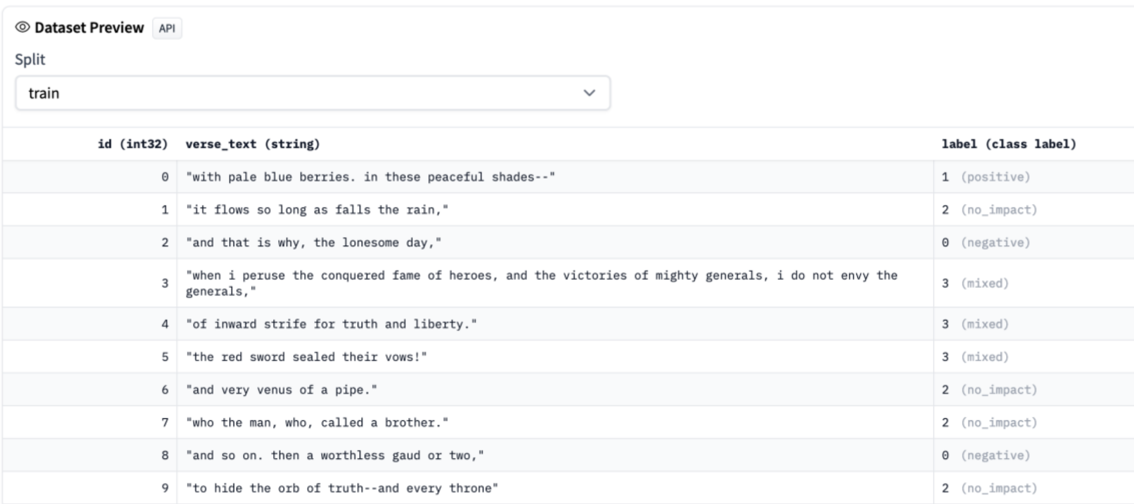
  > *"Note: The original dataset uses different label indices (negative = -1, no impact = 0, positive = 1)"*

Data Splits

The dataset is split into a `train`, `validation`, and `test` split with the following sizes:

	train	validation	test
Number of examples	892	105	104

Figure 4. Poem-sentiment Dataset Details


Dataset Preview API

Split

train

id (int32)	verse_text (string)	label (class label)
0	"with pale blue berries. in these peaceful shades--"	1 (positive)
1	"it flows so long as falls the rain,"	2 (no_impact)
2	"and that is why, the lonesome day,"	0 (negative)
3	"when i peruse the conquered fame of heroes, and the victories of mighty generals, i do not envy the generals,"	3 (mixed)
4	"of inward strife for truth and liberty."	3 (mixed)
5	"the red sword sealed their vows!"	3 (mixed)
6	"and very venus of a pipe."	2 (no_impact)
7	"who the man, who, called a brother."	2 (no_impact)
8	"and so on. then a worthless gaud or two,"	0 (negative)
9	"to hide the orb of truth--and every throne"	2 (no_impact)

Figure 5. Poem-sentiment Dataset Preview

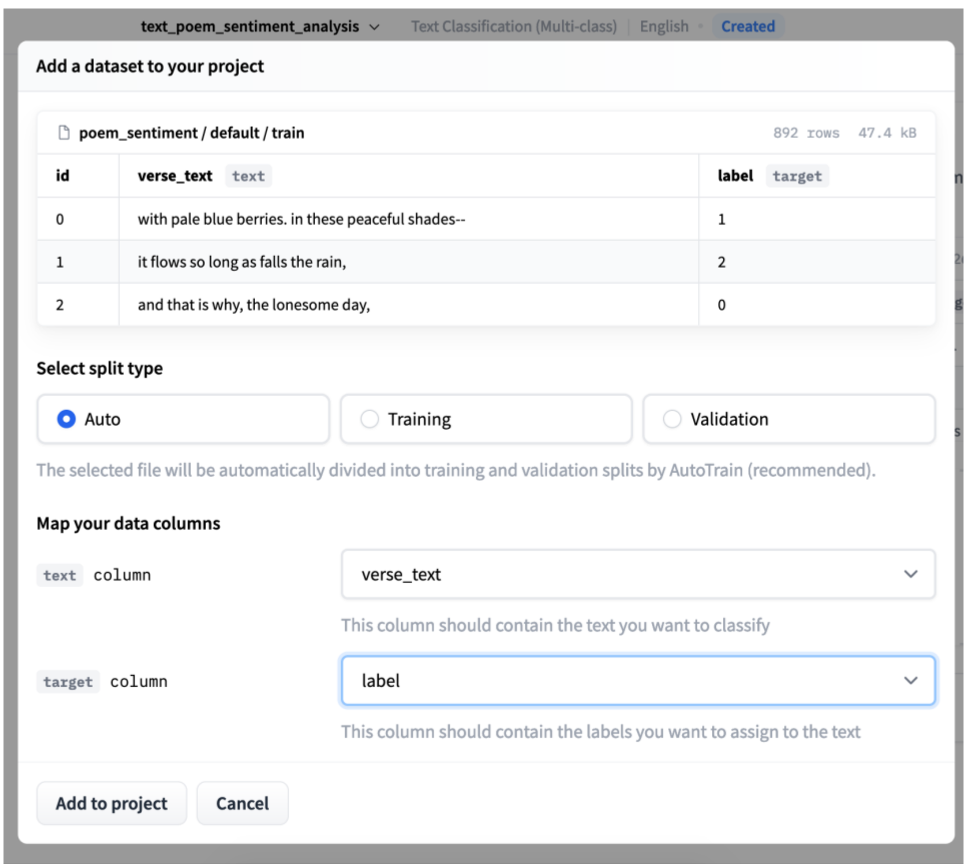


Figure 6. Project 2 Setup

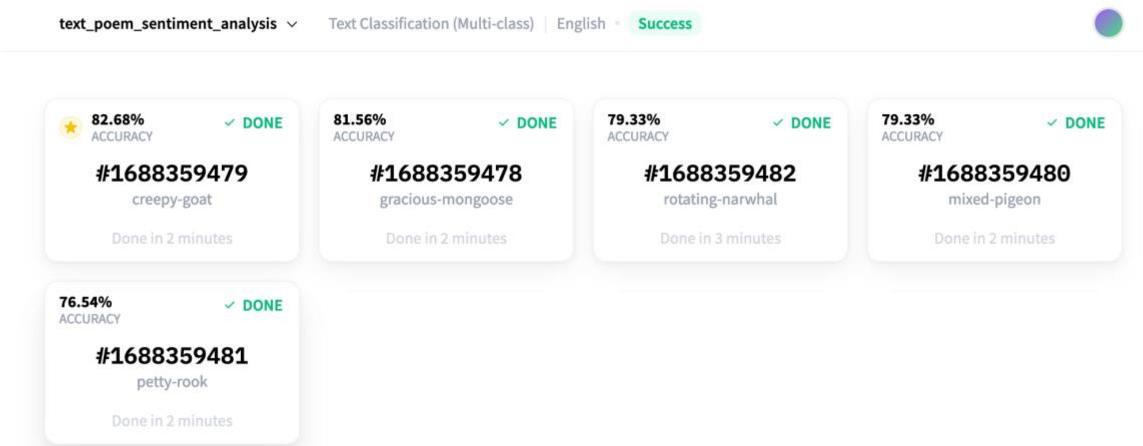


Figure 7. Project 2 Results of Training 5 Models

text_poem_sentiment_analysis | Text Classification (Multi-class) | English | Success

Model ID	Loss	Accuracy	Precision macro	Precision micro	Precision weighted	Recall macro	Recall micro	Recall weighted	F1 macro	F1 micro
#1688359479 creepy-goat	0.6692	0.8268	0.7765	0.8268	0.8276	0.6940	0.8268	0.8268	0.7101	0.8268
#1688359478 gracious-mongoose	0.6098	0.8156	0.8336	0.8156	0.8237	0.6283	0.8156	0.8156	0.6622	0.8156
#1688359482 rotating-narwhal	0.6644	0.7933	0.5624	0.7933	0.7482	0.5960	0.7933	0.7933	0.5785	0.7933
#1688359480 mixed-pigeon	0.6055	0.7933	0.5542	0.7933	0.7645	0.6181	0.7933	0.7933	0.5811	0.7933
#1688359481 petty-rook	0.6128	0.7654	0.5189	0.7654	0.7135	0.5309	0.7654	0.7654	0.5192	0.7654

Figure 8. Project 2 Metrics

CO_2 Emissions (in grams) creepy-goat: 0.0069
CO_2 Emissions (in grams) gracious-mongoose: 0.0066
CO_2 Emissions (in grams) rotating-narwhal: 1.4177
CO_2 Emissions (in grams) mixed-pigeon: 0.0032
CO_2 Emissions (in grams) petty-rook: 0.0024

Once again, the set up flowed easily once the task was selected and the dataset loaded. The CO_2 emissions were much lower overall and the minutes to train the 5 models ranged from 2 to 3 minutes. Accuracy ranged from 0.8268 to 0.7654 for this pretty complex task. I suspect sentiment analysis for poetry is harder than for reviews or tweets.

The hosted inference API allows classification of new text. I tested a few different poems against the best performing model. First, some lines from a haiku by Basho:

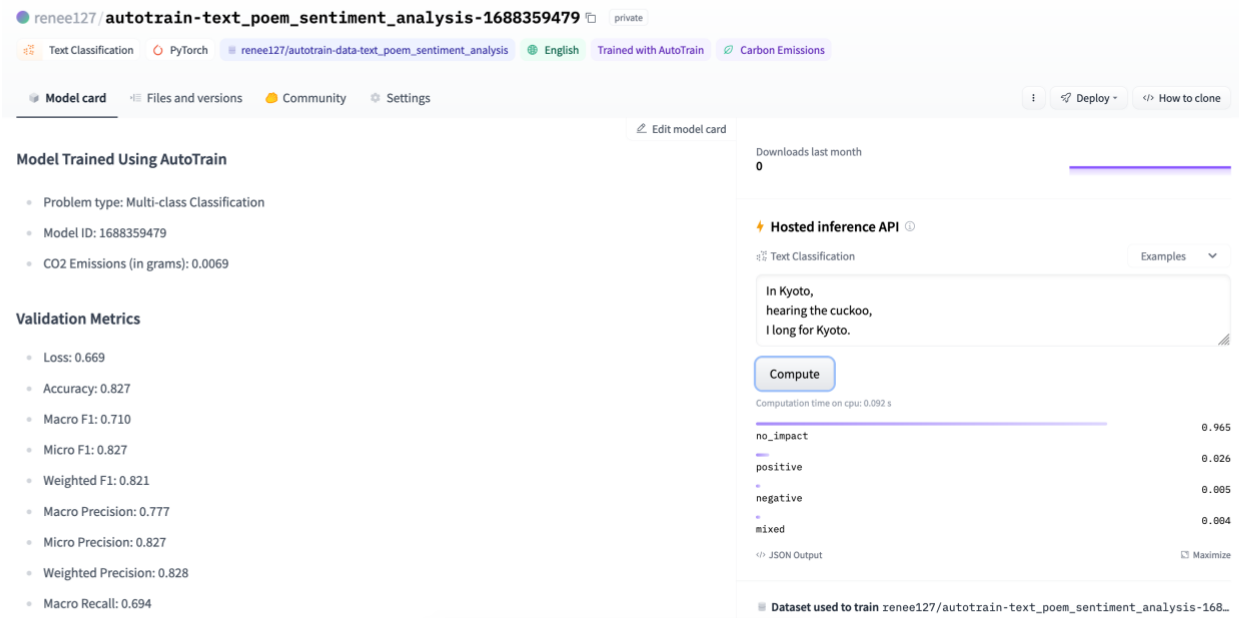


Figure 9. Testing "In Kyoto" by Basho

It wasn’t surprising, given the somewhat inscrutable nature of haikus, that the model was not able to determine the poem’s sentiment. Two poems with more obvious sentiments were tested: lines from Spirits of the Dead by Edgar Allen Poe (negative sentiment) and lines from Infant Joy by William Blake (positive sentiment).

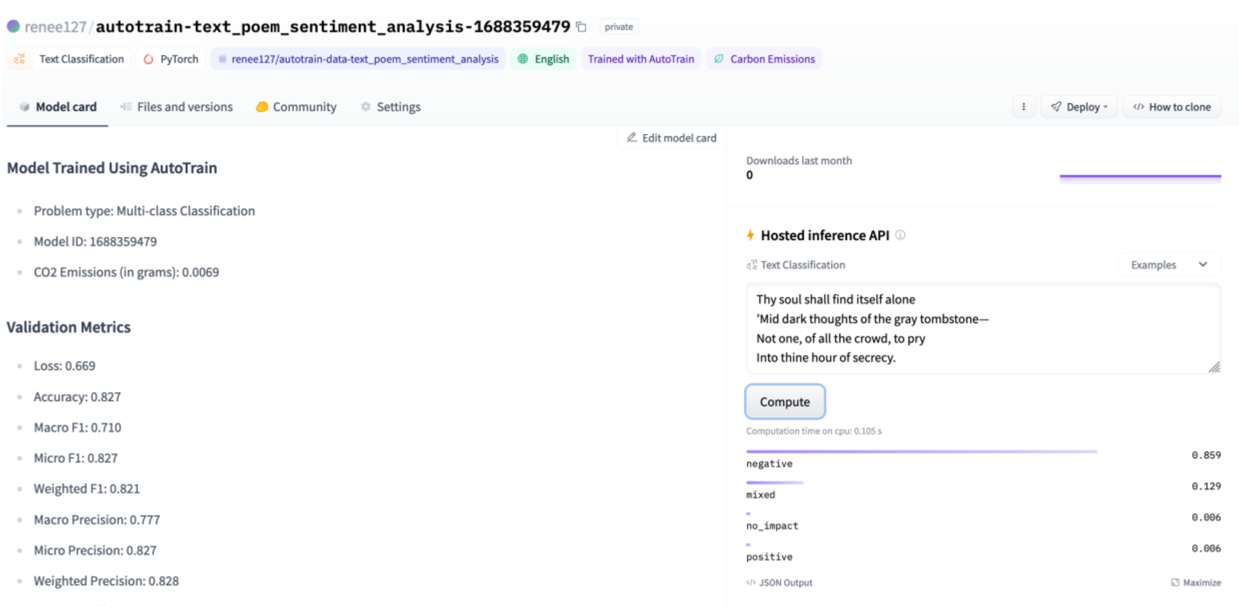


Figure 10. Testing Lines from Spirits of the Dead by Edgar Allen Poe

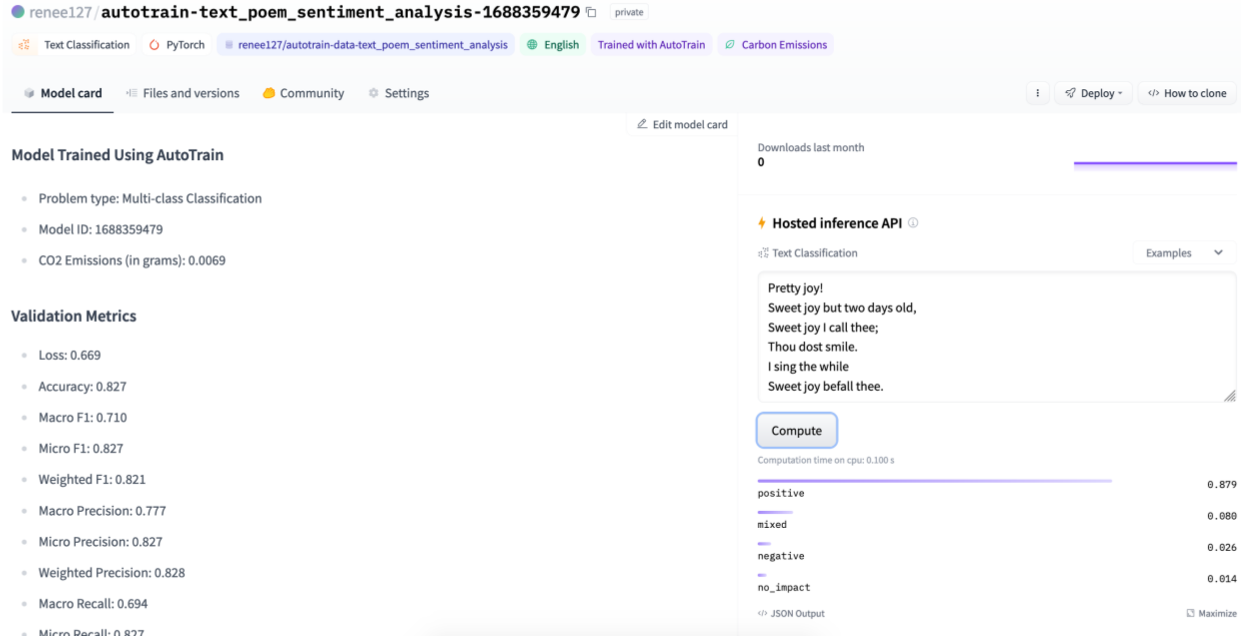


Figure 11. Testing Lines from Infant Joy from William Blake

Project 3: Vision Image Classification using Uploaded Indoor Scenes

I decided to use the images of indoor bus versus indoor subway, since they were such a challenge for Lobe in a previous assignment. Each category had 98 files. I used the prearranged folders option to upload the images.

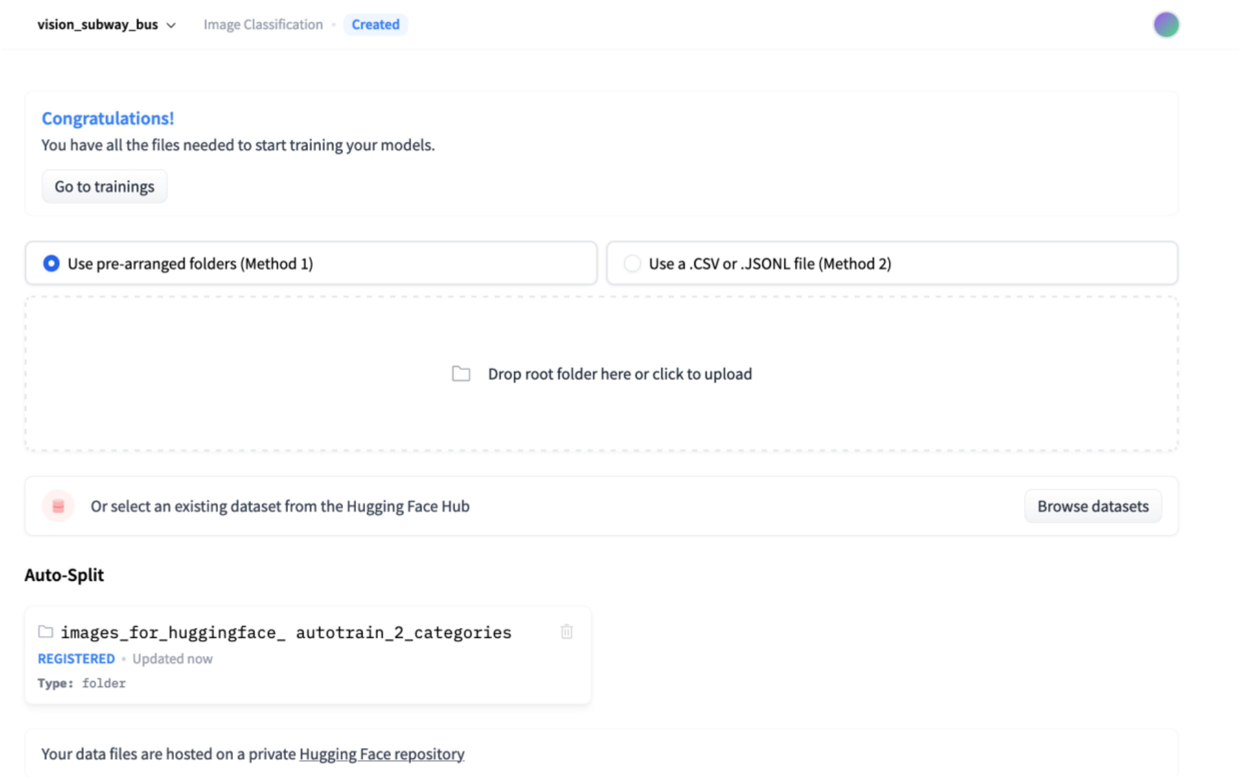


Figure 12. Project 3 Ready for Training

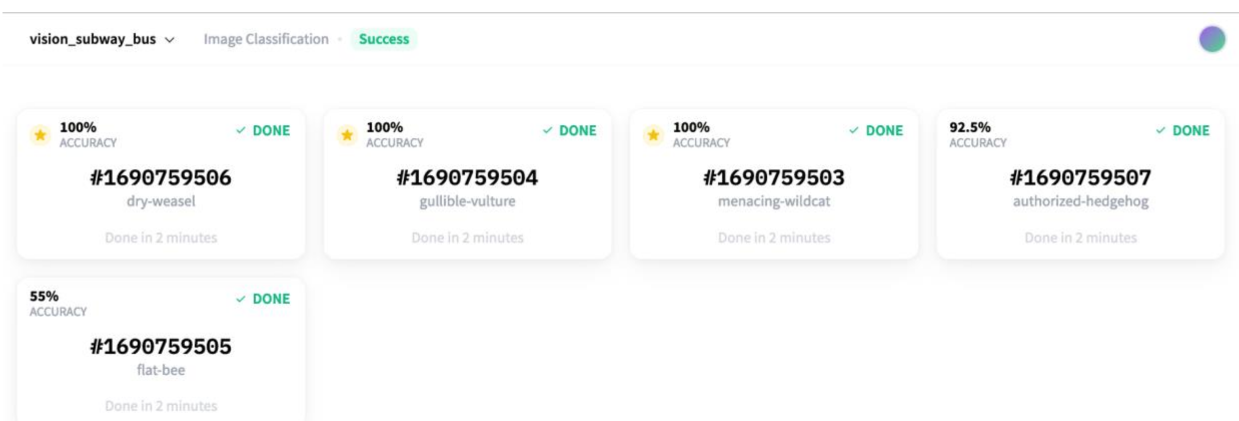
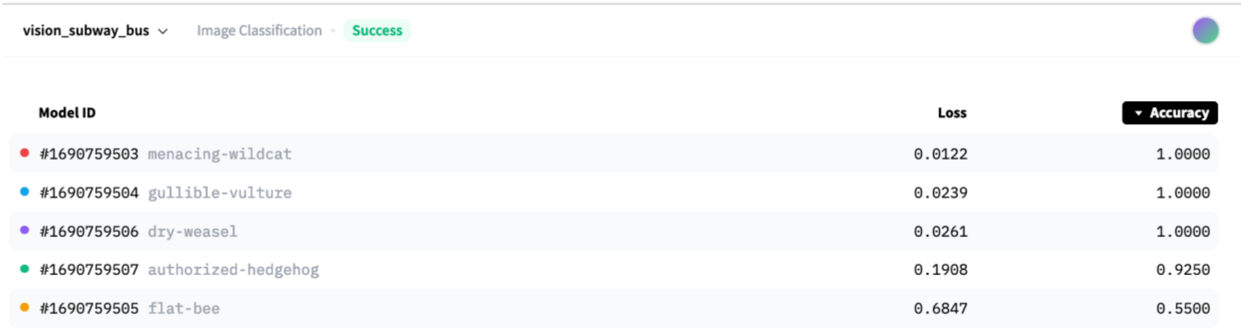


Figure 13. Training Results for Project 3



Model ID	Loss	Accuracy
#1690759503 menacing-wildcat	0.0122	1.0000
#1690759504 gullible-vulture	0.0239	1.0000
#1690759506 dry-weasel	0.0261	1.0000
#1690759507 authorized-hedgehog	0.1908	0.9250
#1690759505 flat-bee	0.6847	0.5500

Figure 14. Metrics for 5 Models from Project 3

Since the results were so good (perfect accuracy) and the training was so quick, I decided to challenge the model with a new collection of 6 categories of images with 70 images of each category.

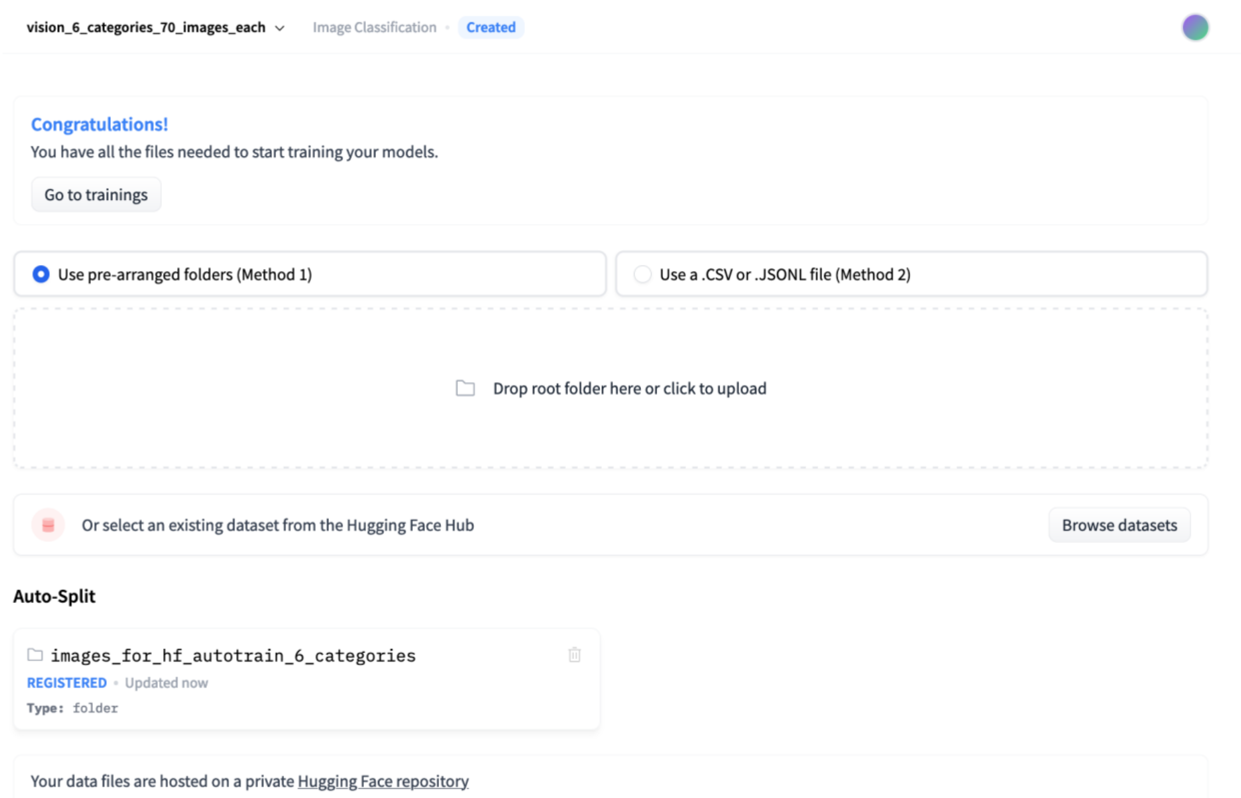


Figure 15. Modified Project 3 for 6 Categories

vision_6_categories_70_images_each · Image Classification · Success

ID		Loss	Accuracy	Precision macro	Precision micro	Precision weighted	Recall macro	Recall micro	Recall weighted	F1 macro	F1 micro	F1 weighted
1759542	virtual-woodcock	0.0559	0.9881	0.9889	0.9881	0.9889	0.9881	0.9881	0.9881	0.9881	0.9881	0.9881
1759540	superior-raccoon	0.1188	0.9881	0.9889	0.9881	0.9889	0.9881	0.9881	0.9881	0.9881	0.9881	0.9881
1759539	junior-shark	0.0811	0.9762	0.9778	0.9762	0.9778	0.9762	0.9762	0.9762	0.9762	0.9762	0.9762
1759543	red-dolphin	0.3920	0.8690	0.8688	0.8690	0.8688	0.8690	0.8690	0.8690	0.8648	0.8690	0.8648
1759541	noxious-buffalo	1.7472	0.2976	0.3291	0.2976	0.3291	0.2976	0.2976	0.2976	0.2057	0.2976	0.2057

Figure 16. Project 3 Metrics for 6 Categories

CO_2 Emissions (in grams): 1.8709
CO_2 Emissions (in grams): 2.5473
CO_2 Emissions (in grams): 0.0075
CO_2 Emissions (in grams): 2.2498
CO_2 Emissions (in grams): 0.7925

Even with 6 categories, the performance of the top 2 models is quite impressive I decided to test the model with previously unseen images in each of the 6 categories.

renee127/autotrain-vision_6_categories_70_images_each-1691759542 · like 0

Image Classification · PyTorch · renee127/autotrain-data-vision_6_categories_70_images_each · Trained with AutoTrain · vision · Carbon Emissions

Model card · Files and versions · Community · Settings · Deploy · How to clone

Edit model card

Model Trained Using AutoTrain

- Problem type: Multi-class Classification
- Model ID: 1691759542
- CO2 Emissions (in grams): 1.8709

Validation Metrics

- Loss: 0.056
- Accuracy: 0.988
- Macro F1: 0.988
- Micro F1: 0.988
- Weighted F1: 0.988
- Macro Precision: 0.989
- Micro Precision: 0.988
- Weighted Precision: 0.989
- Macro Recall: 0.988
- Micro Recall: 0.988

Downloads last month
0

Hosted inference API

Image Classification · Examples

Computation time on cpu: 0.188 s

airport_inside	0.999
inside_bus	0.000
auditorium	0.000
inside_subway	0.000
artstudio	0.000

Figure 17. Testing Project 3 with Inside Airport Image

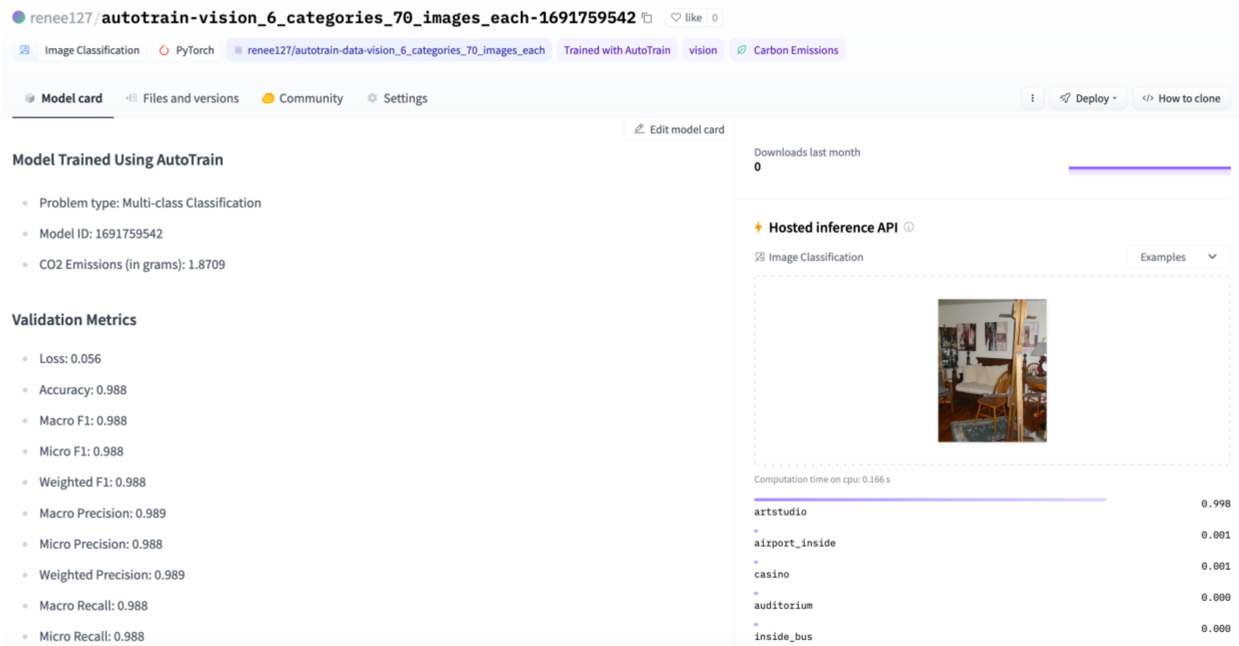


Figure 18. Testing Project 3 with Art Studio Image

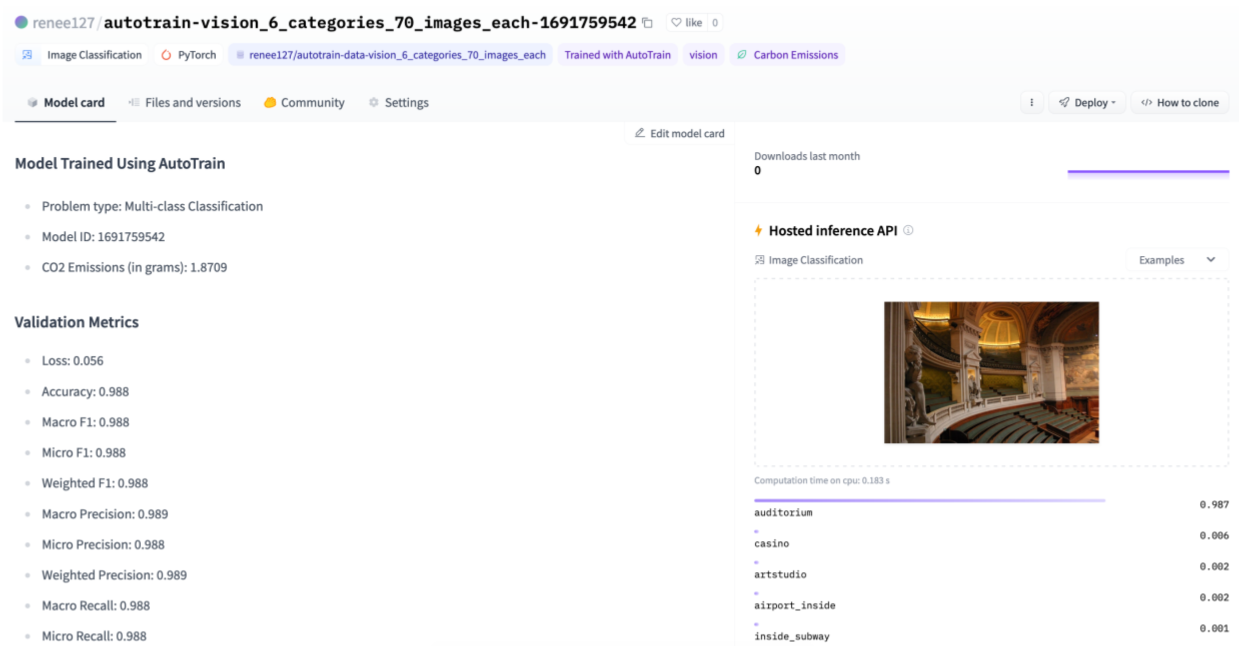


Figure 19. Testing Project 3 with Auditorium Image

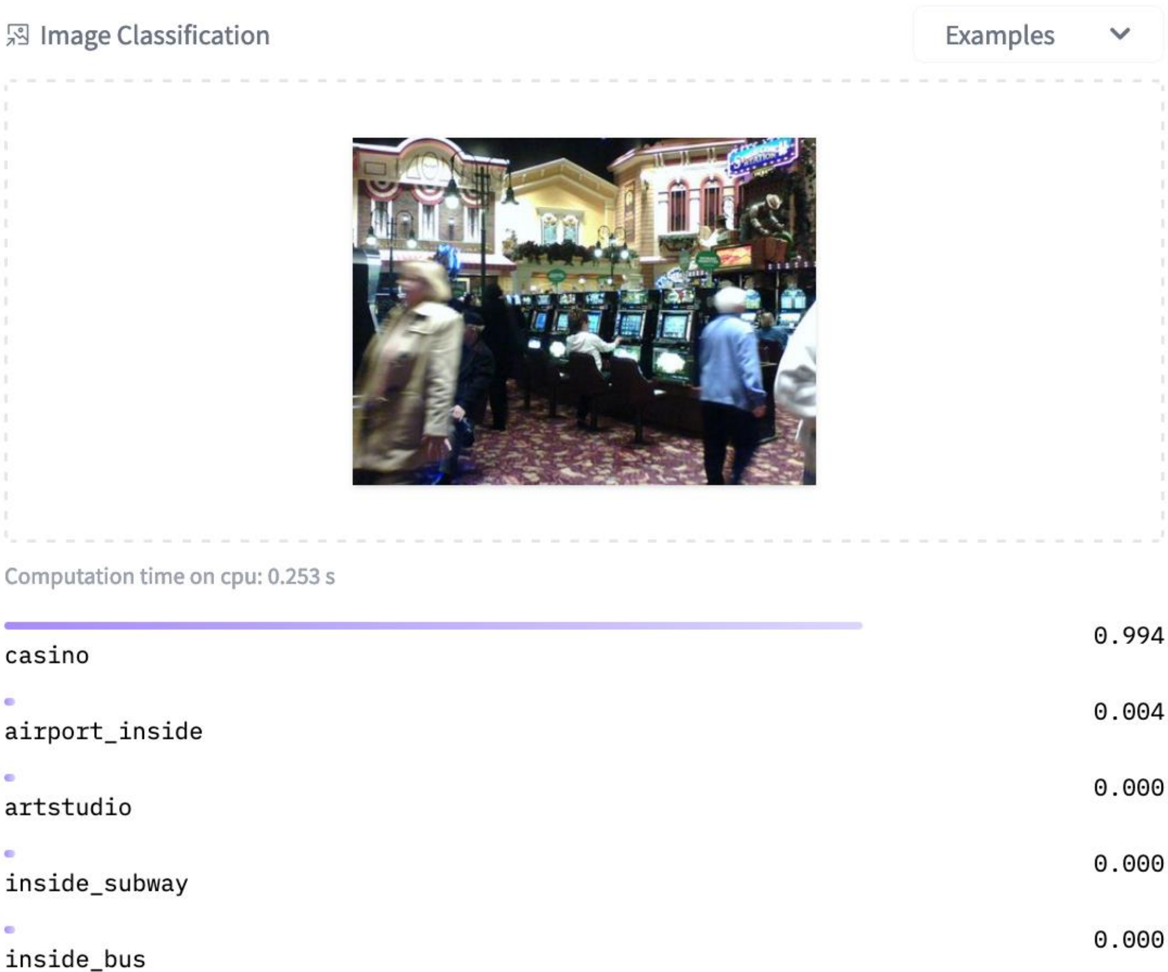


Figure 20. Testing Project 3 with Casino Image

Hosted inference API

Image Classification

Examples

Computation time on cpu: 0.394 s

inside_bus	0.996
inside_subway	0.003
airport_inside	0.000
casino	0.000
auditorium	0.000

Figure 21. Testing Project 3 with Inside Bus Image

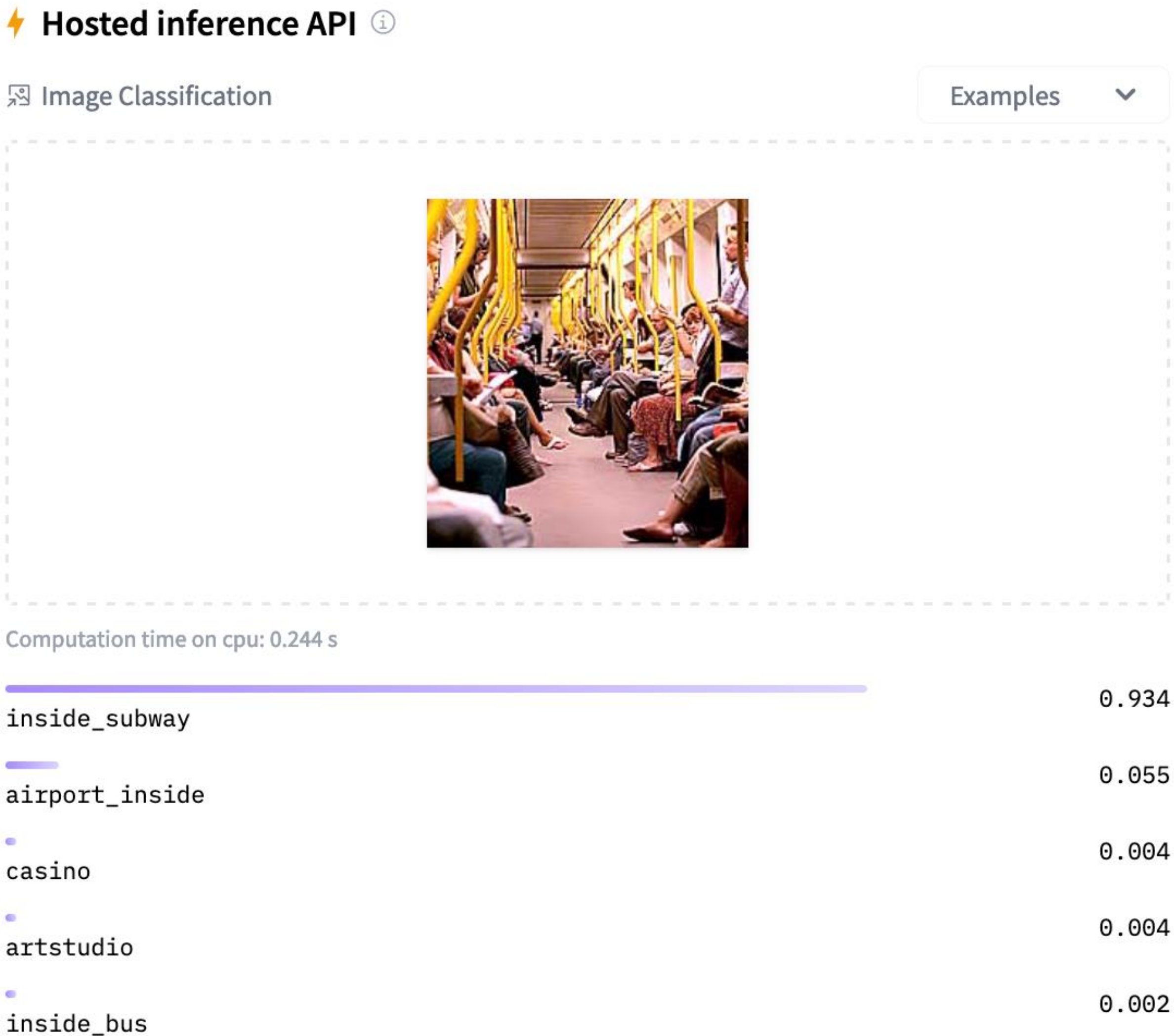


Figure 22. Testing Project 3 with Inside Subway Image

The top model accurately predicted the label of all 6 images, which is very impressive.

Overall Impressions

This is an easy to use and accurate (at least in my small experiments) tool. The ease of setting up, the whimsical names, and the accuracy are delightful! There are many options for different data tasks (though I think there will be many more in the future). I also liked the fact the information on models highlighted the CO2 emissions in grams. I believe there will be growing awareness of the high emissions of many ML applications.

Future Directions

I will continue to explore this tool and am looking forward to learning how to deploy the models that are created. Predictive hacks https://predictivehacks.com/get-started-with-hugging-face-auto-train/ appears to offer some tips for using tokens and colab. I would also like to learn how to clone the models to github.

Other things I plan to explore include how to use the question extraction task and how to use a task to gauge the similarity of 2 statements or questions as in this dataset: