Custom Vision SDK

Image classification

In [5]:
```
from azure.cognitiveservices.vision.customvision.training import CustomVisionTrainingClient
from azure.cognitiveservices.vision.customvision.prediction import CustomVisionPredictionClient
from azure.cognitiveservices.vision.customvision.training.models import ImageFileCreateBatch, ImageFileCreateEnt
from msrest.authentication import ApiKeyCredentials
import time
```

In [ ]:
```

```

In [6]:
```
# Replace with valid values
ENDPOINT = "https://asdfasfasv.cognitiveservices.azure.com/"
training_key = "4d4a0c2f1ecf45edb26f822df3c3d67e"
prediction_key = "2d07822655d84e9889700511270b7a68"
prediction_resource_id = "/subscriptions/f468ceaa-a610-4b88-9742-2b3e8f4ef76c/resourceGroups/Day2/providers/Micr
```

In [7]:
```
# Authenticate client

credentials = ApiKeyCredentials(in_headers={"Training-key": training_key})
trainer = CustomVisionTrainingClient(ENDPOINT, credentials)
prediction_credentials = ApiKeyCredentials(in_headers={"Prediction-key": prediction_key})
predictor = CustomVisionPredictionClient(ENDPOINT, prediction_credentials)
```

In [8]:
```
# Classifier

publish_iteration_name = "classifyModel"

credentials = ApiKeyCredentials(in_headers={"Training-key": training_key})
trainer = CustomVisionTrainingClient(ENDPOINT, credentials)

# Create a new project
print ("Creating project...")
project = trainer.create_project("My New Project")
```

```
Creating project...
```

In [9]:
```
# Make two tags in the new project
cat_tag = trainer.create_tag(project.id, "cat")
dog_tag = trainer.create_tag(project.id, "dog")
```

In [10]:
```
# Update this with the path to where you downloaded the images.
import glob
import cv2
base_image_location = "C:/Users/Jaswanth Reddy/Desktop/Image dataset/api_cat_dog/"

print("Adding images...")

image_list = []

for image_num in range(1, 26):
    file_name = "cat_{}.jpg".format(image_num)
    with open(base_image_location + "cat/" + file_name, "rb") as image_contents:
        image_list.append(ImageFileCreateEntry(name=file_name, contents=image_contents.read(), tag_ids=[cat_tag.

for image_num in range(1, 26):
    file_name = "dog_{}.jpg".format(image_num)
    with open(base_image_location + "dog/" + file_name, "rb") as image_contents:
        image_list.append(ImageFileCreateEntry(name=file_name, contents=image_contents.read(), tag_ids=[dog_tag.

upload_result = trainer.create_images_from_files(project.id, ImageFileCreateBatch(images=image_list))
if not upload_result.is_batch_successful:
    print("Image batch upload failed.")
    for image in upload_result.images:
        print("Image status: ", image.status)
    exit(-1)
```

```
Adding images...
```

In [11]:
```
print ("Training...")
iteration = trainer.train_project(project.id)
while (iteration.status != "Completed"):
    iteration = trainer.get_iteration(project.id, iteration.id)
    print ("Training status: " + iteration.status)
    time.sleep(1)

# The iteration is now trained. Publish it to the project endpoint
trainer.publish_iteration(project.id, iteration.id, publish_iteration_name, prediction_resource_id)
print ("Done!")
```

```
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training

Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
Training status: Training
```

In [12]:
```
# Now there is a trained endpoint that can be used to make a prediction
prediction_credentials = ApiKeyCredentials(in_headers={"Prediction-key": prediction_key})
predictor = CustomVisionPredictionClient(ENDPOINT, prediction_credentials)

with open(r"C:\Users\Jaswanth Reddy\Desktop\Image dataset\cat_dog\train\cat\cat.2981.jpg", "rb") as image_conter
    results = predictor.classify_image(
        project.id, publish_iteration_name, image_contents.read())

    # Display the results.
    for prediction in results.predictions:
        print("\t" + prediction.tag_name +
              ": {0:.2f}%".format(prediction.probability * 100))
```

```
        cat: 61.15%
        dog: 40.10%
```

In [26]:
```
# Now there is a trained endpoint that can be used to make a prediction
prediction_credentials = ApiKeyCredentials(in_headers={"Prediction-key": prediction_key})
predictor = CustomVisionPredictionClient(ENDPOINT, prediction_credentials)

with open(r"C:\Users\Jaswanth Reddy\Desktop\Image dataset\api_cat_dog\dog\dog_24.jpg", "rb") as image_contents:
    results = predictor.classify_image(
        project.id, publish_iteration_name, image_contents.read())

    # Display the results.
    for prediction in results.predictions:
        print("\t" + prediction.tag_name +
              ": {0:.2f}%".format(prediction.probability * 100))
```

```
        dog: 56.18%
        cat: 41.91%
```

In [ ]: