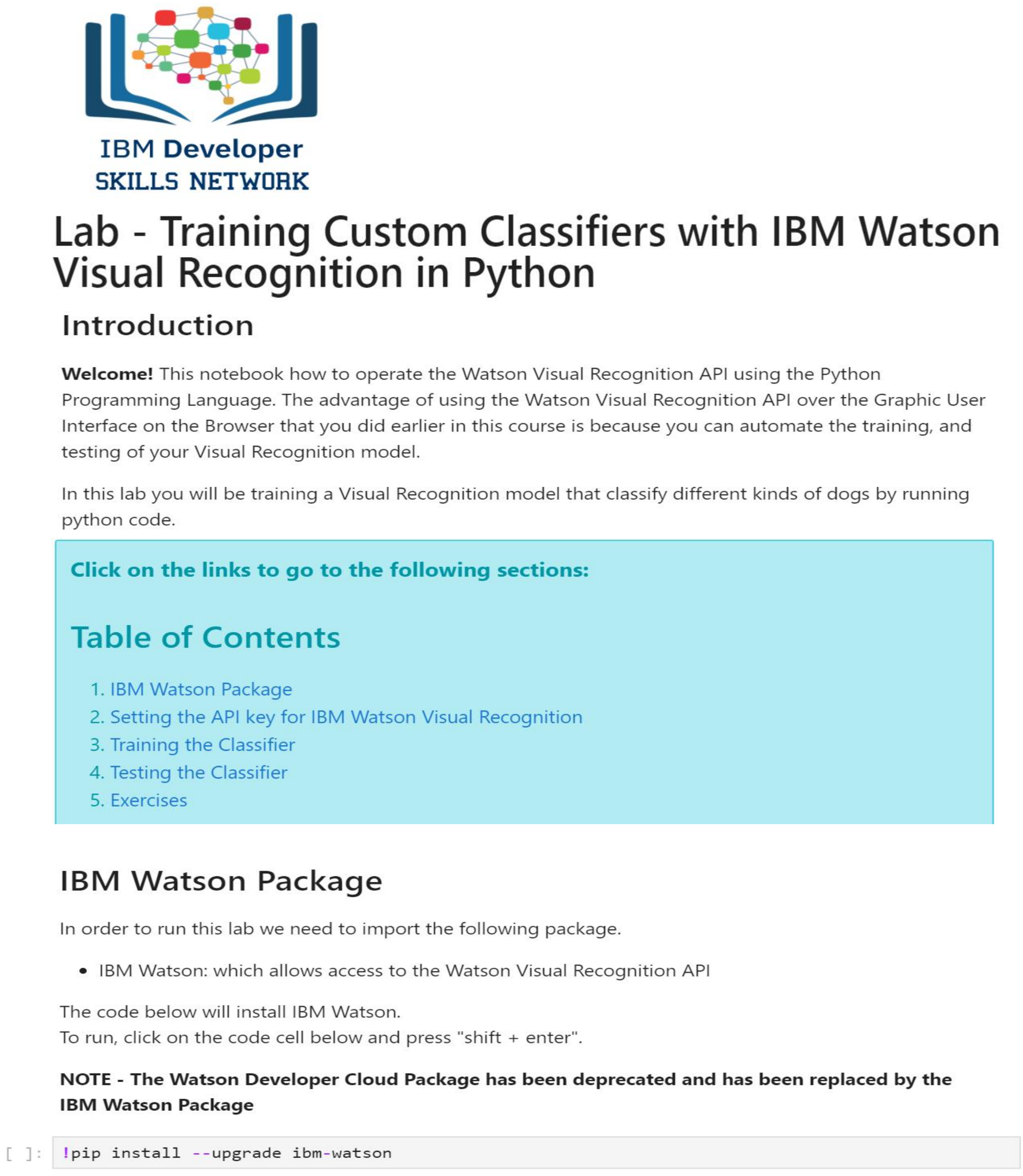
LAB: Training Custom Classifiers with IBM Watson

https://www.coursera.org/learn/introduction-computer-vision-watson-opencv/ungradedLti/yLHdc/lab-training-custom-classifiers-with-ibm-watson

In this lab, you will use Python to Train and Test your Visual Recognition Classifier.

Ce cours utilise l'outil d'un tiers, LAB: Training Custom Classifiers with IBM Watson, pour améliorer votre expérience d'apprentissage. L'outil référence des informations de base comme votre nom, votre adresse e-mail et votre ID Coursera.



IBM Developer
SKILLS NETWORK

Lab - Training Custom Classifiers with IBM Watson Visual Recognition in Python

Introduction

Welcome! This notebook how to operate the Watson Visual Recognition API using the Python Programming Language. The advantage of using the Watson Visual Recognition API over the Graphic User Interface on the Browser that you did earlier in this course is because you can automate the training, and testing of your Visual Recognition model.

In this lab you will be training a Visual Recognition model that classify different kinds of dogs by running python code.

Click on the links to go to the following sections:

Table of Contents

IBM Watson Package

In order to run this lab we need to import the following package.

- IBM Watson: which allows access to the Watson Visual Recognition API

The code below will install IBM Watson.
To run, click on the code cell below and press "shift + enter".

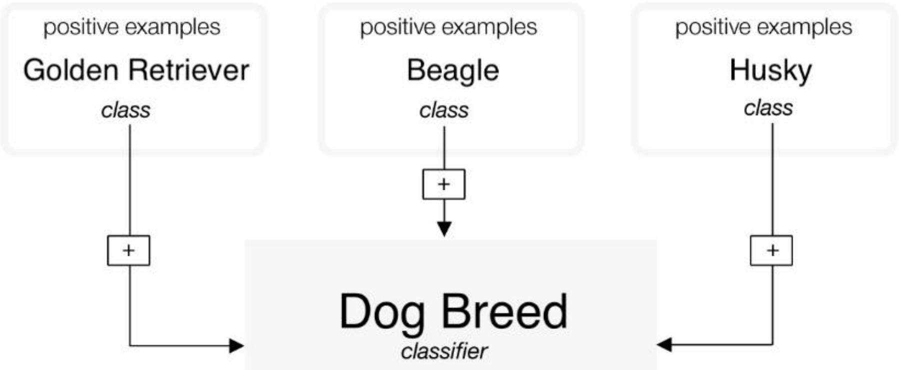
NOTE - The Watson Developer Cloud Package has been deprecated and has been replaced by the IBM Watson Package

```
[ ]: !pip install --upgrade ibm-watson
```

Code : !pip install --upgrade ibm-watson

Goal of this lab:

In this lab, we will be creating a completely new image classifier using training images. We will train a custom classifier to identify between three different dog breeds (Golden Retriever, Beagle and Husky).



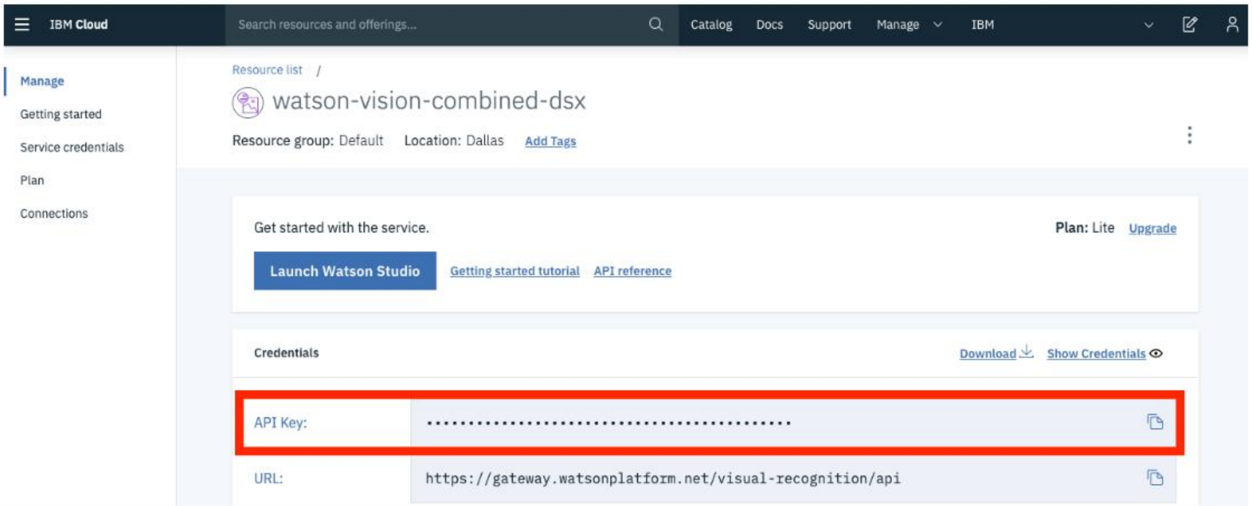
Setting the API key for IBM Watson Visual Recognition

In order for you to use the IBM Watson Visual Recognition API, you will need the API key of the Visual Recognition instance that you have created in the previous sections.

Log into your IBM Cloud Account with the following link.

https://cloud.ibm.com

1. Click on **Services**
2. Under Services, click on your Watson Visual Recognition Instance
3. Copy the **API Key** and past it in the code cell below



watson-vision-combined-dsx

Resource group: Default Location: Dallas

Get started with the service.

Plan: Lite

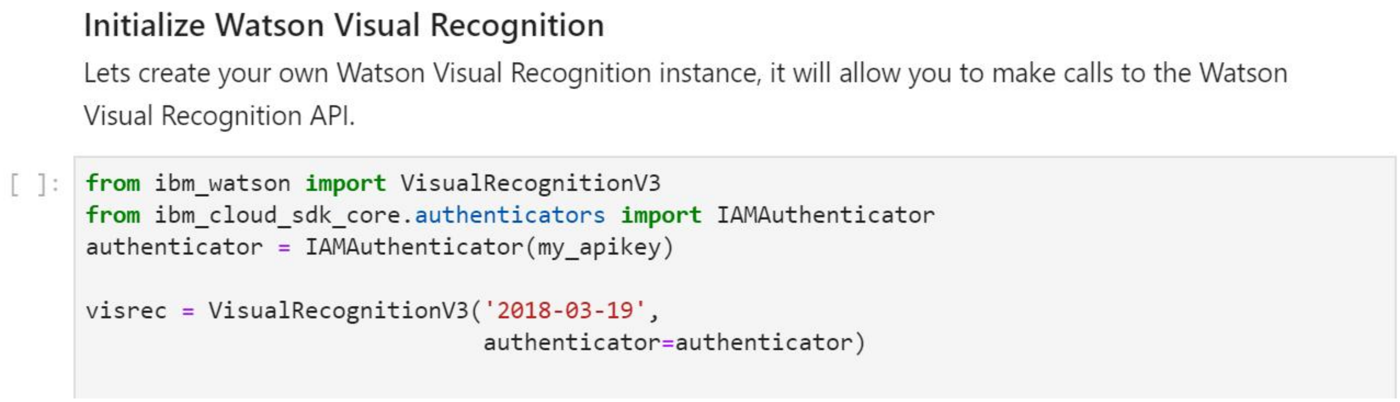
Credentials

API Key: ··

URL: https://gateway.watsonplatform.net/visual-recognition/api

```
[ ]: # Paste your API key for IBM Watson Visual Recognition below:
     my_apikey = 'Paste_your_API_key'
```

CODE: # Paste your API key for IBM Watson Visual Recognition below:
my_apikey = 'Paste_your_API_key'



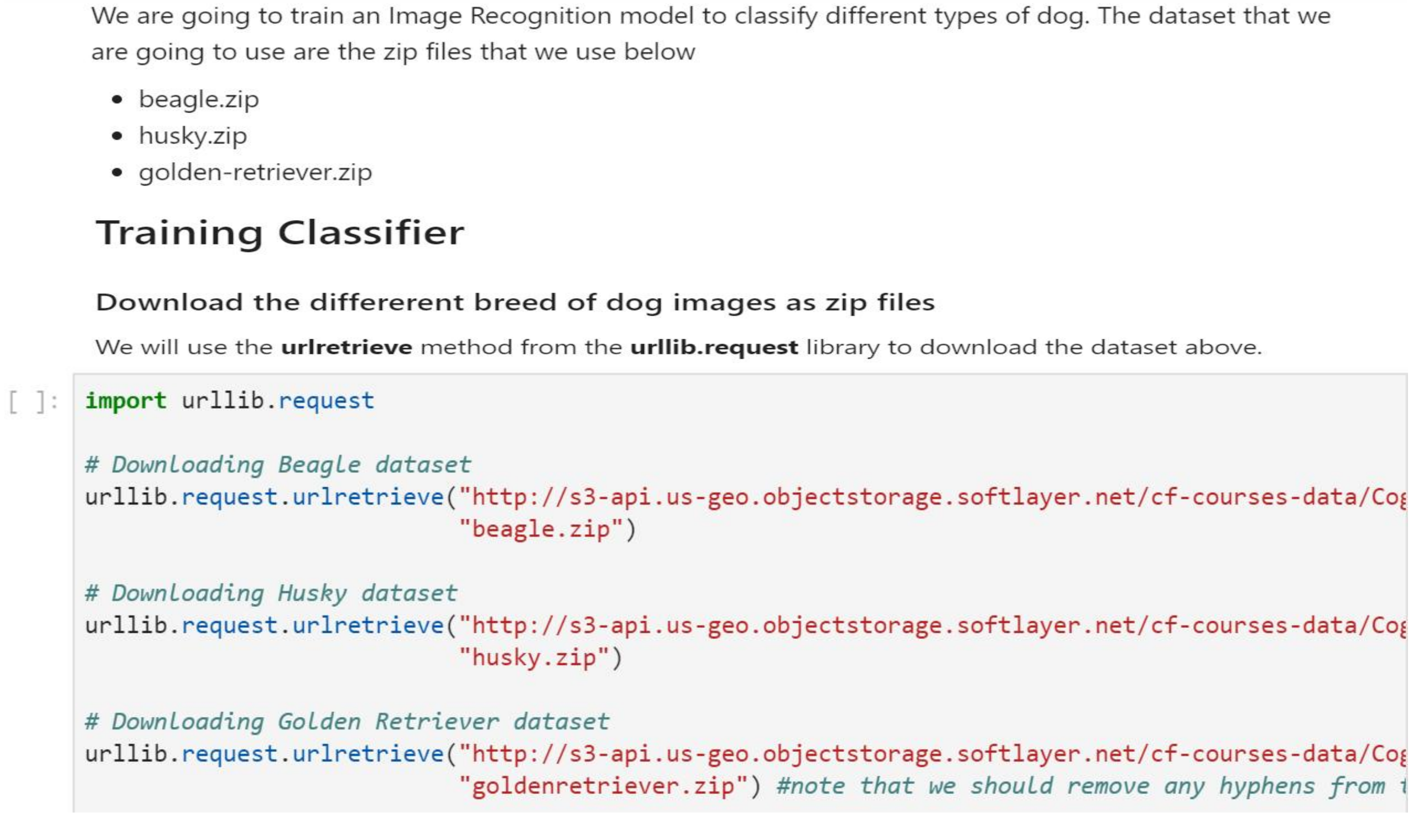
Initialize Watson Visual Recognition

Lets create your own Watson Visual Recognition instance, it will allow you to make calls to the Watson Visual Recognition API.

```
[ ]: from ibm_watson import VisualRecognitionV3
     from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
     authenticator = IAMAuthenticator(my_apikey)

     visrec = VisualRecognitionV3('2018-03-19',
                                  authenticator=authenticator)
```

CODE:

```
from ibm_watson import VisualRecognitionV3
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
authenticator = IAMAuthenticator(my_apikey)

visrec = VisualRecognitionV3('2018-03-19',
                              authenticator=authenticator)
```



We are going to train an Image Recognition model to classify different types of dog. The dataset that we are going to use are the zip files that we use below

- beagle.zip
- husky.zip
- golden-retriever.zip

Training Classifier

Download the differerent breed of dog images as zip files

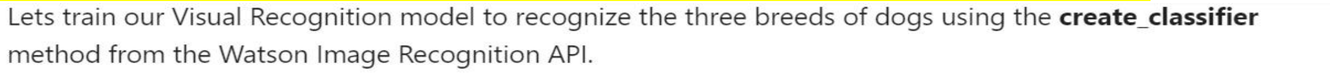
We will use the **urlretrieve** method from the **urllib.request** library to download the dataset above.

```
[ ]: import urllib.request

     # Downloading Beagle dataset
     urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cog
                                "beagle.zip")

     # Downloading Husky dataset
     urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cog
                                "husky.zip")

     # Downloading Golden Retriever dataset
     urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cog
                                "goldenretriever.zip") #note that we should remove any hyphens from t
```

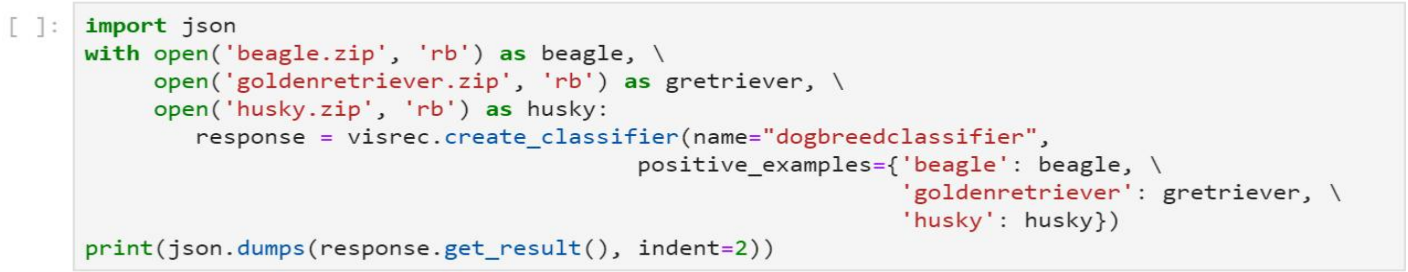
CODE:

```
import urllib.request

# Downloading Beagle dataset
urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Beagle.zip",
                           "beagle.zip")

# Downloading Husky dataset
```

```
urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Husky.zip",
                           "husky.zip")

# Downloading Golden Retriever dataset
urllib.request.urlretrieve("http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/GoldenRetriever.zip",
                           "goldenretriever.zip") #note that we should remove any hyphens from the zip file name
```



Lets train our Visual Recognition model to recognize the three breeds of dogs using the **create_classifier** method from the Watson Image Recognition API.



```
[ ]: import json
     with open('beagle.zip', 'rb') as beagle, \
          open('goldenretriever.zip', 'rb') as gretriever, \
          open('husky.zip', 'rb') as husky:
             response = visrec.create_classifier(name="dogbreedclassifier",
                                                 positive_examples={'beagle': beagle, \
                                                                    'goldenretriever': gretriever, \
                                                                    'husky': husky})
     print(json.dumps(response.get_result(), indent=2))
```

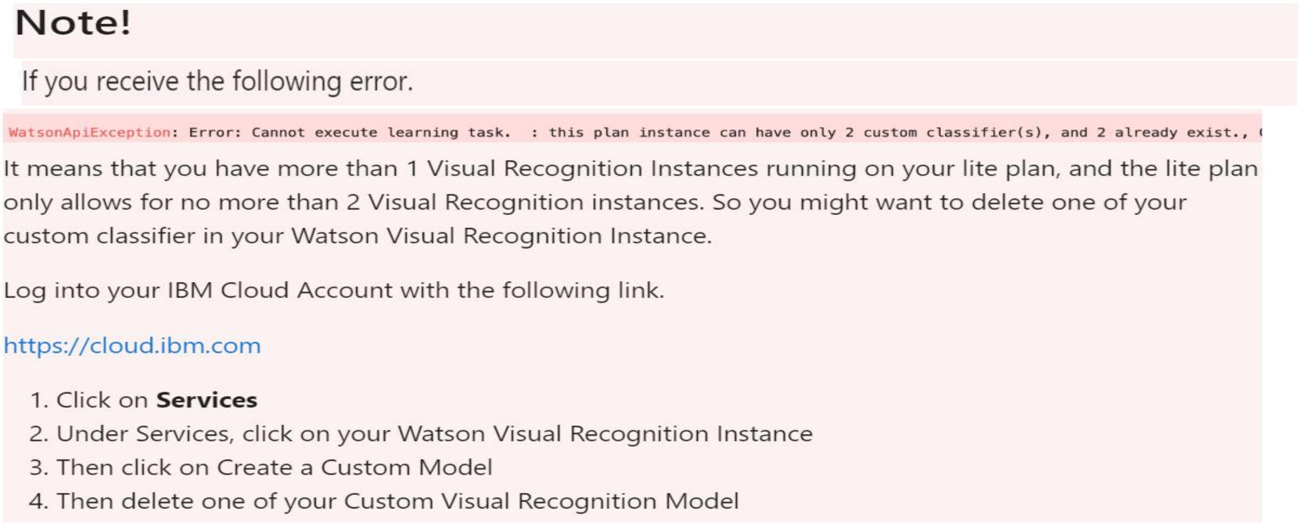
```
[ ]: #lets grab the classifier id
     classifier_id = response.get_result()["classifier_id"]
     classifier_id
```

CODE 1:

```
import json
with open('beagle.zip', 'rb') as beagle, \
     open('goldenretriever.zip', 'rb') as gretriever, \
     open('husky.zip', 'rb') as husky:
        response = visrec.create_classifier(name="dogbreedclassifier",
                                            positive_examples={'beagle': beagle, \
                                                               'goldenretriever': gretriever, \
                                                               'husky': husky})
print(json.dumps(response.get_result(), indent=2))
```

CODE 2:

```
#lets grab the classifier id
classifier_id = response.get_result()["classifier_id"]
classifier_id
```



Note!

If you receive the following error.

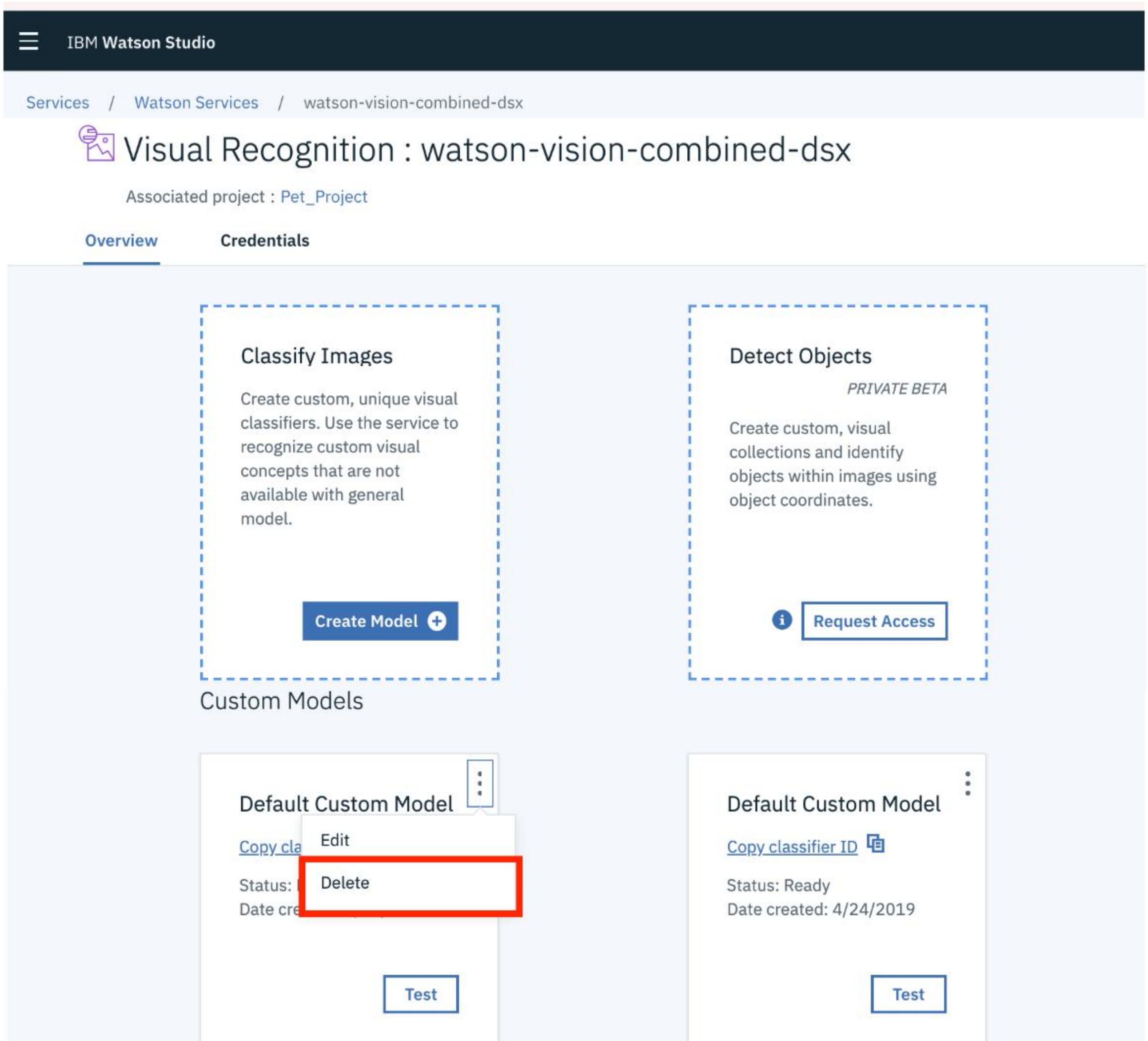
```
WatsonApiException: Error: Cannot execute learning task.  : this plan instance can have only 2 custom classifier(s), and 2 already exist.,
```

It means that you have more than 1 Visual Recognition Instances running on your lite plan, and the lite plan only allows for no more than 2 Visual Recognition instances. So you might want to delete one of your custom classifier in your Watson Visual Recognition Instance.

Log into your IBM Cloud Account with the following link.

https://cloud.ibm.com

1. Click on **Services**
2. Under Services, click on your Watson Visual Recognition Instance
3. Then click on Create a Custom Model
4. Then delete one of your Custom Visual Recognition Model



Visual Recognition : watson-vision-combined-dsx

Associated project : Pet_Project

Classify Images

Create custom, unique visual classifiers. Use the service to recognize custom visual concepts that are not available with general model.

Detect Objects

PRIVATE BETA

Create custom, visual collections and identify objects within images using object coordinates.

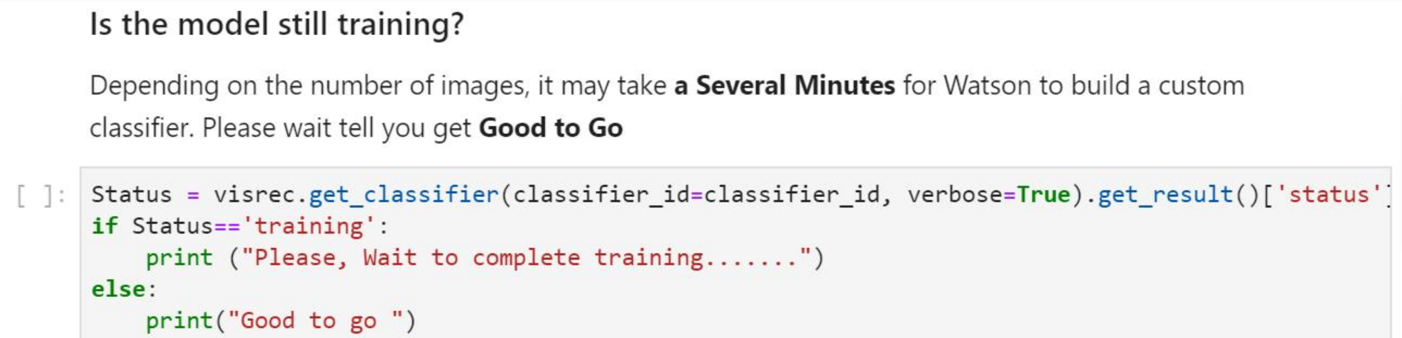
Custom Models

Default Custom Model

Status:
Date cre

Default Custom Model

Status: Ready
Date created: 4/24/2019



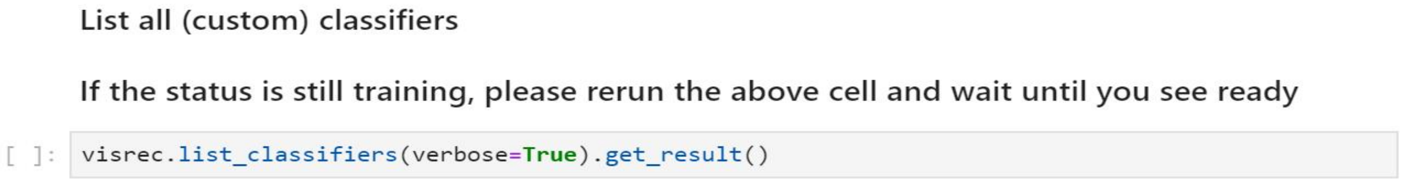
Is the model still training?

Depending on the number of images, it may take **a Several Minutes** for Watson to build a custom classifier. Please wait tell you get **Good to Go**

```
[ ]: Status = visrec.get_classifier(classifier_id=classifier_id, verbose=True).get_result()['status'
     if Status=='training':
         print ("Please, Wait to complete training.......")
     else:
         print("Good to go ")
```

CODE:

```
Status = visrec.get_classifier(classifier_id=classifier_id, verbose=True).get_result()['status']
if Status=='training':
    print ("Please, Wait to complete training.......")
else:
    print("Good to go ")
```

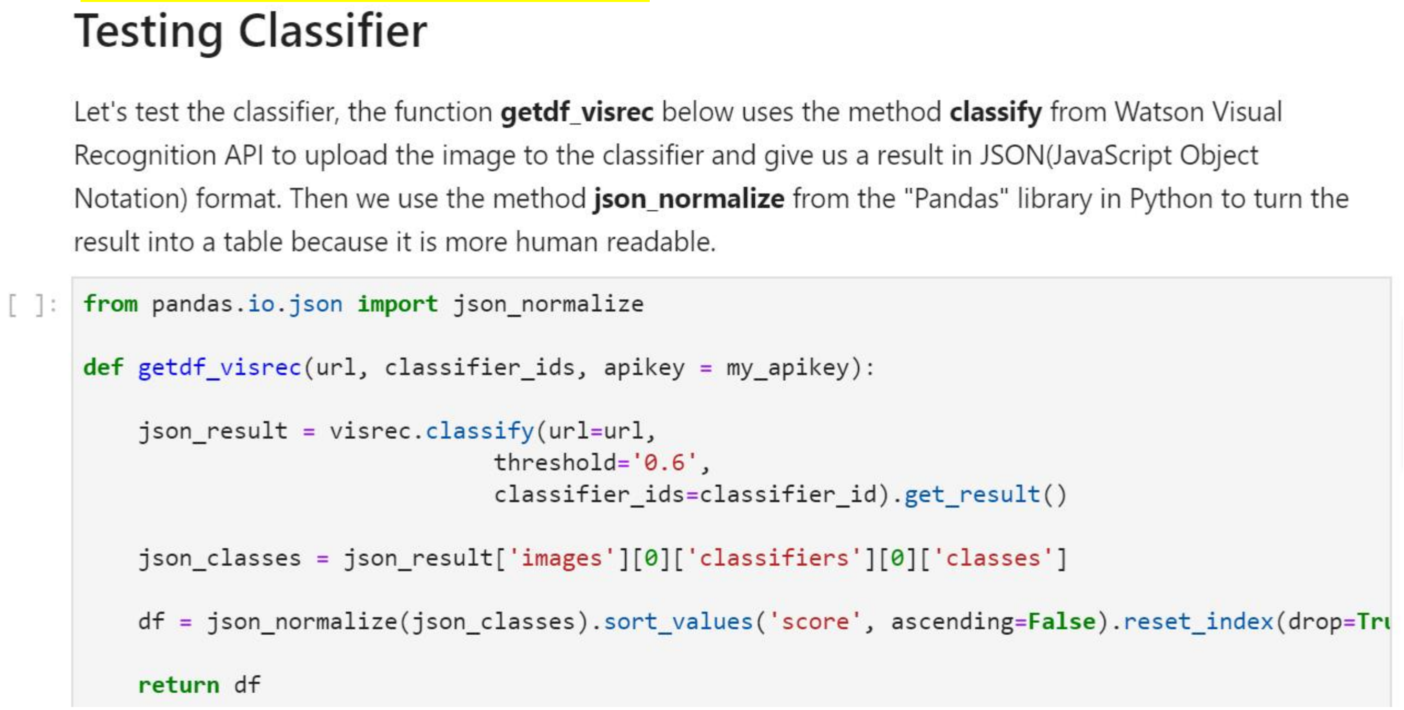


List all (custom) classifiers

If the status is still training, please rerun the above cell and wait until you see ready

```
[ ]: visrec.list_classifiers(verbose=True).get_result()
```

CODE: visrec.list_classifiers(verbose=True).get_result()



Testing Classifier

Let's test the classifier, the function **getdf_visrec** below uses the method **classify** from Watson Visual Recognition API to upload the image to the classifier and give us a result in JSON(JavaScript Object Notation) format. Then we use the method **json_normalize** from the "Pandas" library in Python to turn the result into a table because it is more human readable.

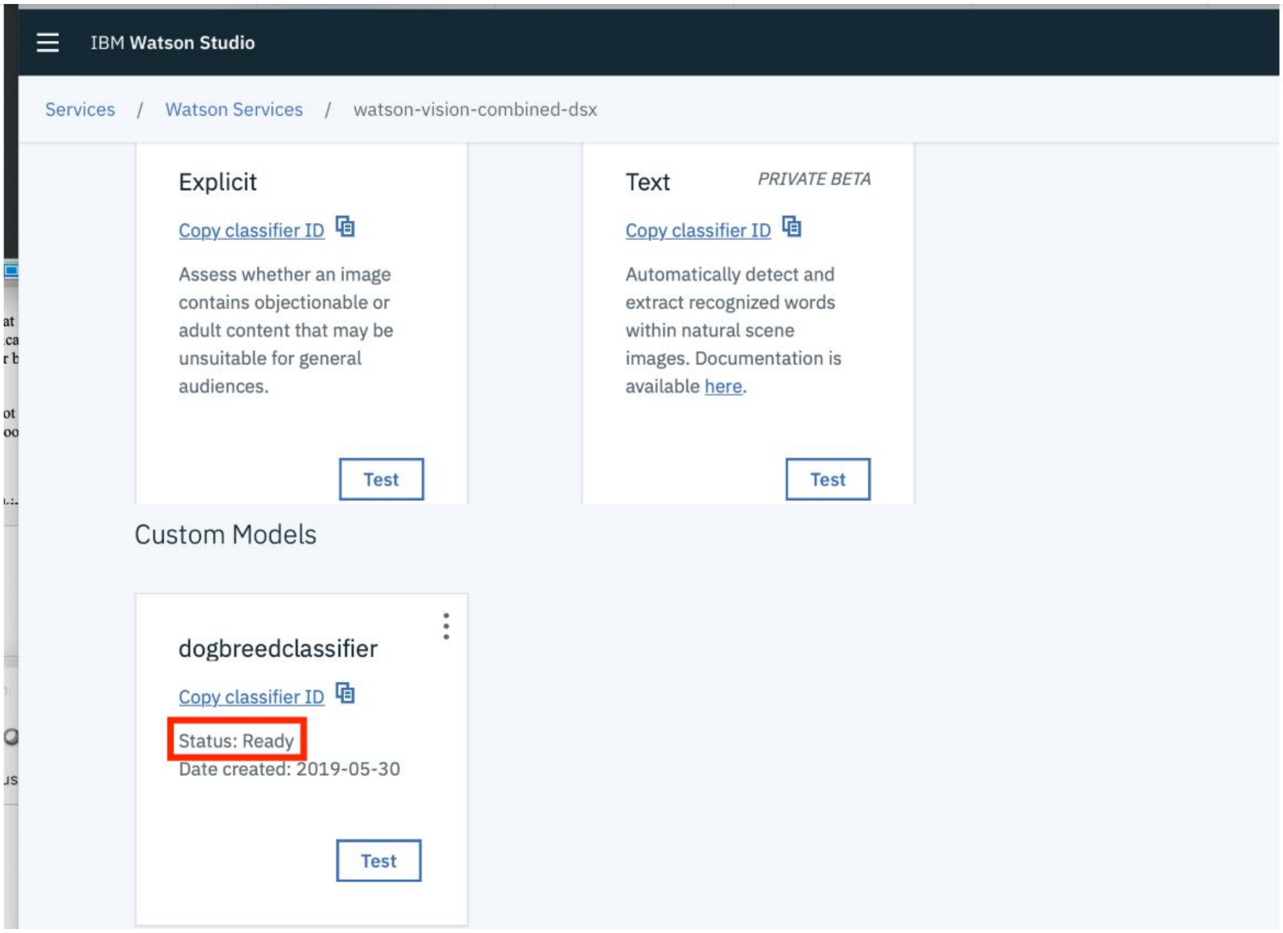
```
[ ]: from pandas.io.json import json_normalize

     def getdf_visrec(url, classifier_ids, apikey = my_apikey):

         json_result = visrec.classify(url=url,
                                    threshold='0.6',
                                    classifier_ids=classifier_id).get_result()

         json_classes = json_result['images'][0]['classifiers'][0]['classes']

         df = json_normalize(json_classes).sort_values('score', ascending=False).reset_index(drop=Tru

         return df
```

CODE:

```
from pandas.io.json import json_normalize

def getdf_visrec(url, classifier_ids, apikey = my_apikey):

    json_result = visrec.classify(url=url,
                           threshold='0.6',
                           classifier_ids=classifier_id).get_result()

    json_classes = json_result['images'][0]['classifiers'][0]['classes']

    df = json_normalize(json_classes).sort_values('score', ascending=False).reset_index(drop=True)

    return df
```

Let's test our Visual Recognition model on this picture of Golden Retriever

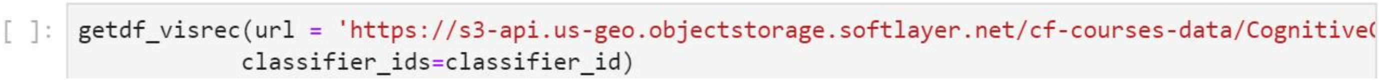
Please wait for your Custom Model to finish training before you upload your test image to your Custom Classifier, **you might get an error if your model is still training and you run the function below.**



Assess whether an image contains objectionable or adult content that may be unsuitable for general audiences.

Automatically detect and extract recognized words within natural scene images. Documentation is available here.

Status: Ready
Date created: 2019-05-30



```
[ ]: getdf_visrec(url = 'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/Cognitive
                classifier_ids=classifier_id)
```

CODE:

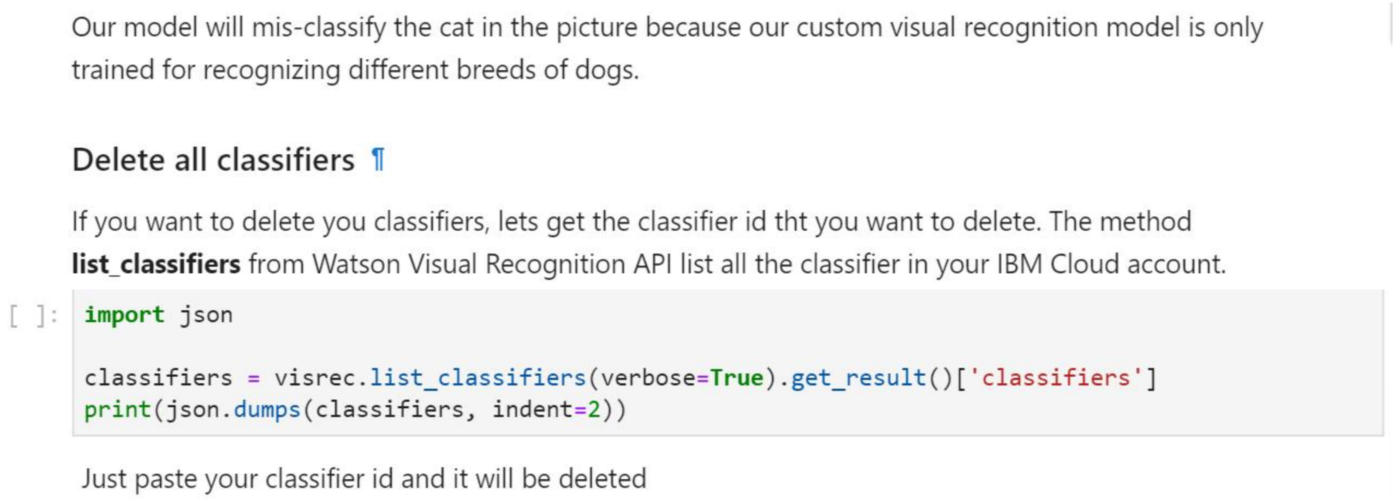
```
getdf_visrec(url = 'https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/GoldenRetriever1_stacked.jpg',
        classifier_ids=classifier_id)
```

Lets test our Visual Recognition model on this picture of cat

```
[ ]: getdf_visrec(url = 'http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveCl
                  classifier_ids=classifier_id)
```

CODE;

```
getdf_visrec(url = 'http://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/cat-2083492_960_720.jpg',
        classifier_ids=classifier_id)
```



Our model will mis-classify the cat in the picture because our custom visual recognition model is only trained for recognizing different breeds of dogs.

Delete all classifiers ¶

If you want to delete you classifiers, lets get the classifier id tht you want to delete. The method **list_classifiers** from Watson Visual Recognition API list all the classifier in your IBM Cloud account.

```
[ ]: import json

     classifiers = visrec.list_classifiers(verbose=True).get_result()['classifiers']
     print(json.dumps(classifiers, indent=2))
```

Just paste your classifier id and it will be deleted

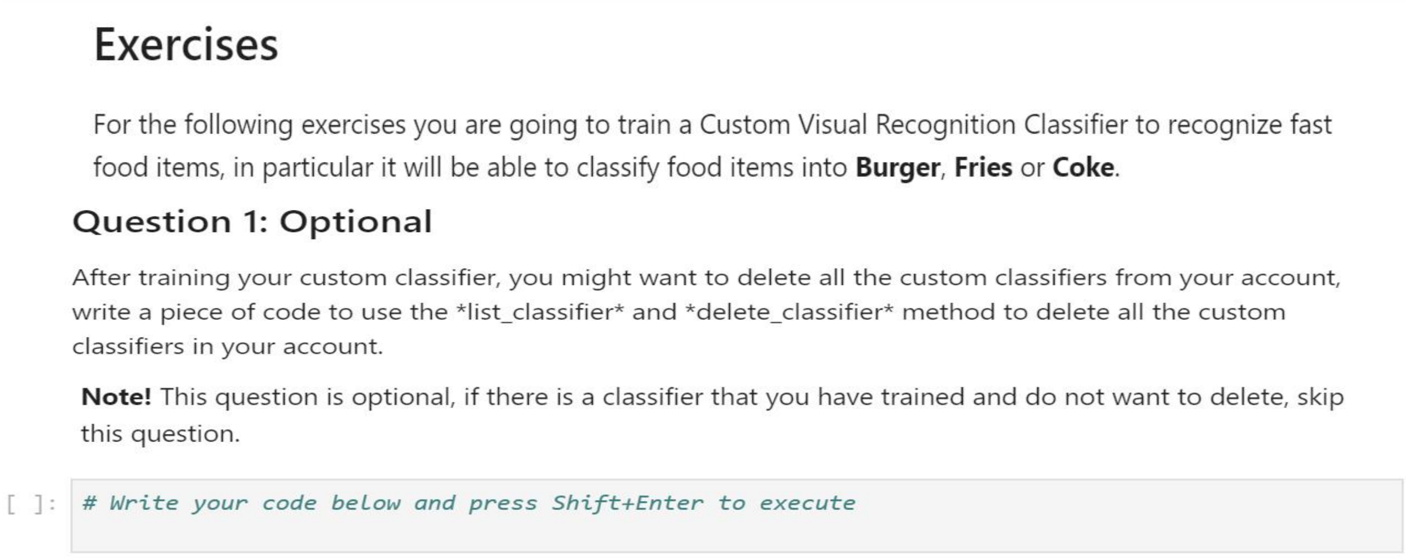
CODE;

```
import json

classifiers = visrec.list_classifiers(verbose=True).get_result()['classifiers']
print(json.dumps(classifiers, indent=2))
```

```
[ ]: mycid = '' #the classifier id you want to delete
     visrec.delete_classifier(classifier_id = mycid)
```

CODE:

```
mycid = '' #the classifier id you want to delete
visrec.delete_classifier(classifier_id = mycid)
```



Exercises

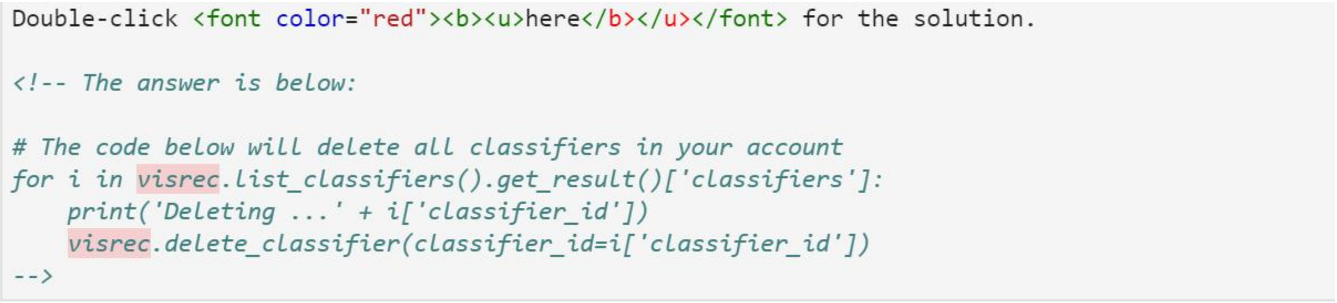
For the following exercises you are going to train a Custom Visual Recognition Classifier to recognize fast food items, in particular it will be able to classify food items into **Burger**, **Fries** or **Coke**.

Question 1: Optional

After training your custom classifier, you might want to delete all the custom classifiers from your account, write a piece of code to use the *list_classifier* and *delete_classifier* method to delete all the custom classifiers in your account.

Note! This question is optional, if there is a classifier that you have trained and do not want to delete, skip this question.

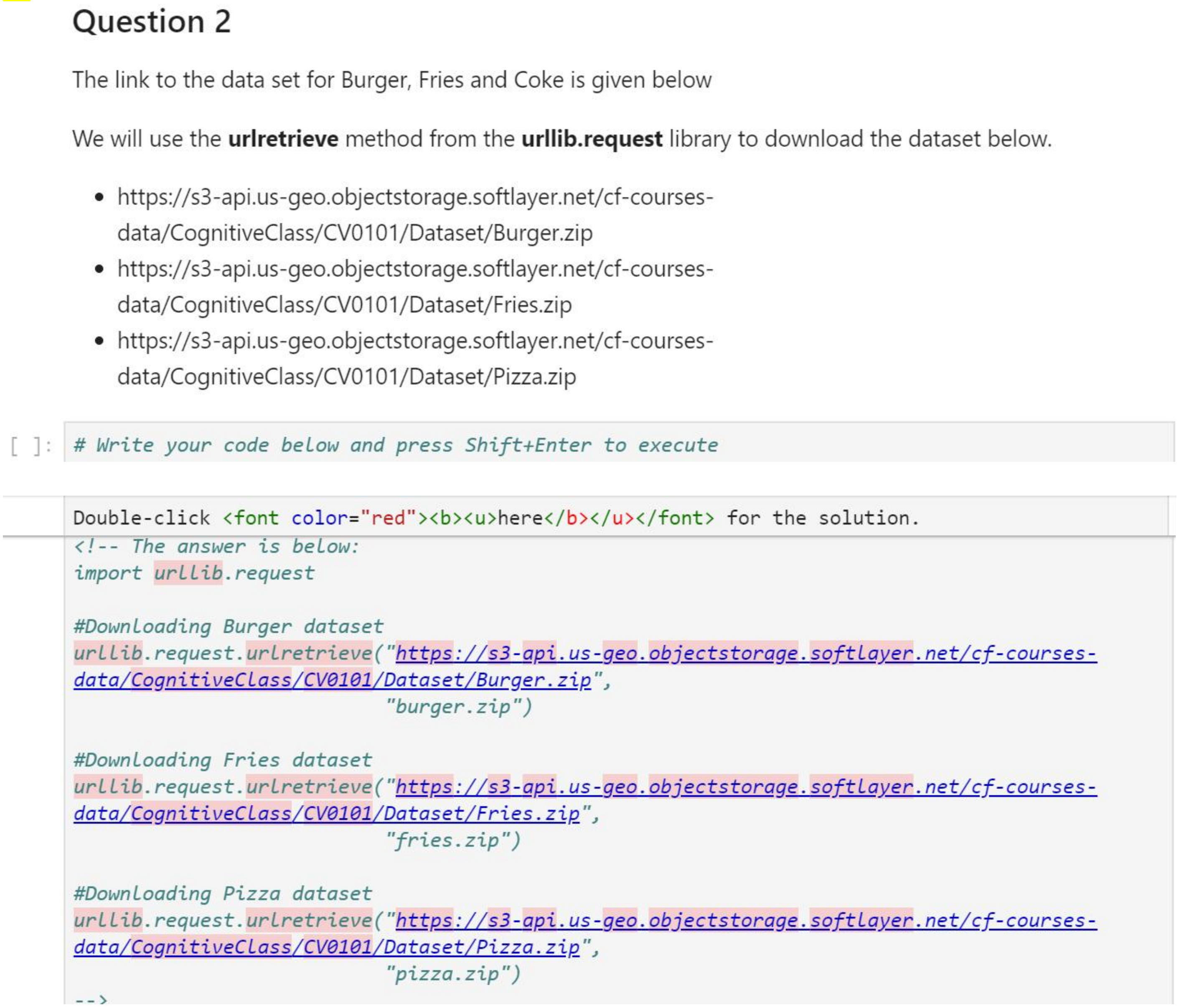
```
[ ]: # Write your code below and press Shift+Enter to execute
```



```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:

# The code below will delete all classifiers in your account
for i in visrec.list_classifiers().get_result()['classifiers']:
    print('Deleting ...' + i['classifier_id'])
    visrec.delete_classifier(classifier_id=i['classifier_id'])
-->
```

CODE :

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:

# The code below will delete all classifiers in your account
for i in visrec.list_classifiers().get_result()['classifiers']:
    print('Deleting ...' + i['classifier_id'])
    visrec.delete_classifier(classifier_id=i['classifier_id'])
-->
```



Question 2

The link to the data set for Burger, Fries and Coke is given below

We will use the **urlretrieve** method from the **urllib.request** library to download the dataset below.

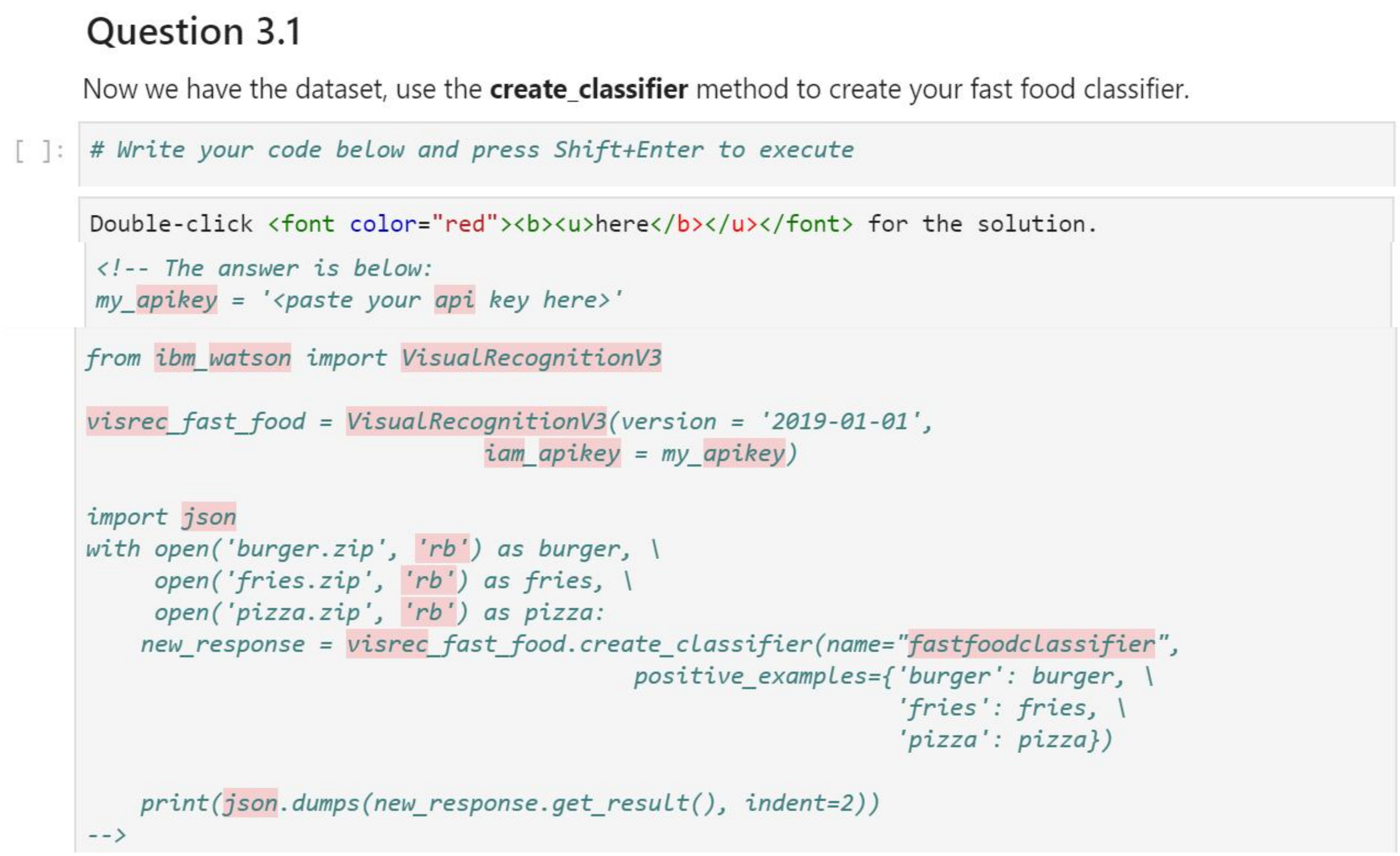
- https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Burger.zip
- https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Fries.zip
- https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Pizza.zip

```
# Write your code below and press Shift+Enter to execute
```

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.
<!-- The answer is below:
import urllib.request

#Downloading Burger dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Burger.zip",
                           "burger.zip")

#Downloading Fries dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Fries.zip",
                           "fries.zip")

#Downloading Pizza dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Pizza.zip",
                           "pizza.zip")
-->
```

CODE :

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
import urllib.request

#Downloading Burger dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Burger.zip",
                           "burger.zip")

#Downloading Fries dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Fries.zip",
                           "fries.zip")

#Downloading Pizza dataset
urllib.request.urlretrieve("https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/CV0101/Dataset/Pizza.zip",
                           "pizza.zip")
-->
```



Question 3.1

Now we have the dataset, use the **create_classifier** method to create your fast food classifier.

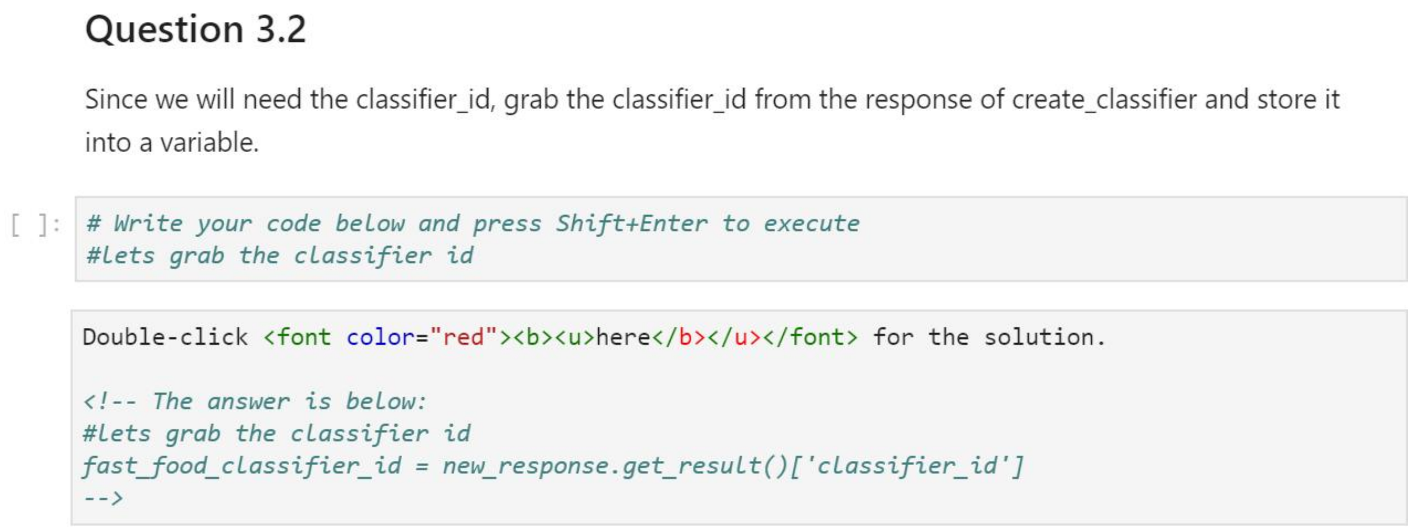
```
[ ]: # Write your code below and press Shift+Enter to execute
```

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.
<!-- The answer is below:
my_apikey = '<paste your api key here>'

from ibm_watson import VisualRecognitionV3

visrec_fast_food = VisualRecognitionV3(version = '2019-01-01',
                                       iam_apikey = my_apikey)

import json
with open('burger.zip', 'rb') as burger, \
     open('fries.zip', 'rb') as fries, \
     open('pizza.zip', 'rb') as pizza:
    new_response = visrec_fast_food.create_classifier(name="fastfoodclassifier",
                                                      positive_examples={'burger': burger, \
                                                                         'fries': fries, \
                                                                         'pizza': pizza})

    print(json.dumps(new_response.get_result(), indent=2))
-->
```

CODE ;

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
my_apikey = '<paste your api key here>'
```

```
from ibm_watson import VisualRecognitionV3

visrec_fast_food = VisualRecognitionV3(version = '2019-01-01',
                         iam_apikey = my_apikey)

import json
with open('burger.zip', 'rb') as burger, \
    open('fries.zip', 'rb') as fries, \
    open('pizza.zip', 'rb') as pizza:
   new_response = visrec_fast_food.create_classifier(name="fastfoodclassifier",
                                positive_examples={'burger': burger, \
                                               'fries': fries, \
                                               'pizza': pizza})

   print(json.dumps(new_response.get_result(), indent=2))
-->
```



Question 3.2

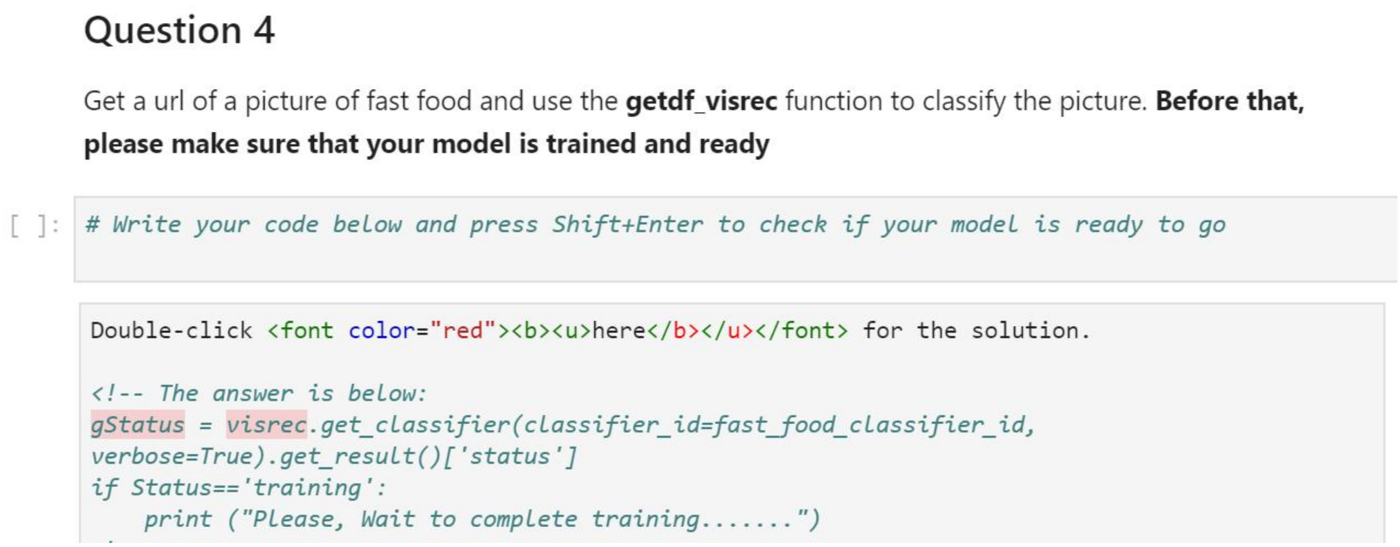
Since we will need the classifier_id, grab the classifier_id from the response of create_classifier and store it into a variable.

```
[ ]: # Write your code below and press Shift+Enter to execute
     #lets grab the classifier id
```

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
#lets grab the classifier id
fast_food_classifier_id = new_response.get_result()['classifier_id']
-->
```

CODE :

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
#lets grab the classifier id
fast_food_classifier_id = new_response.get_result()['classifier_id']
-->
```



Question 4

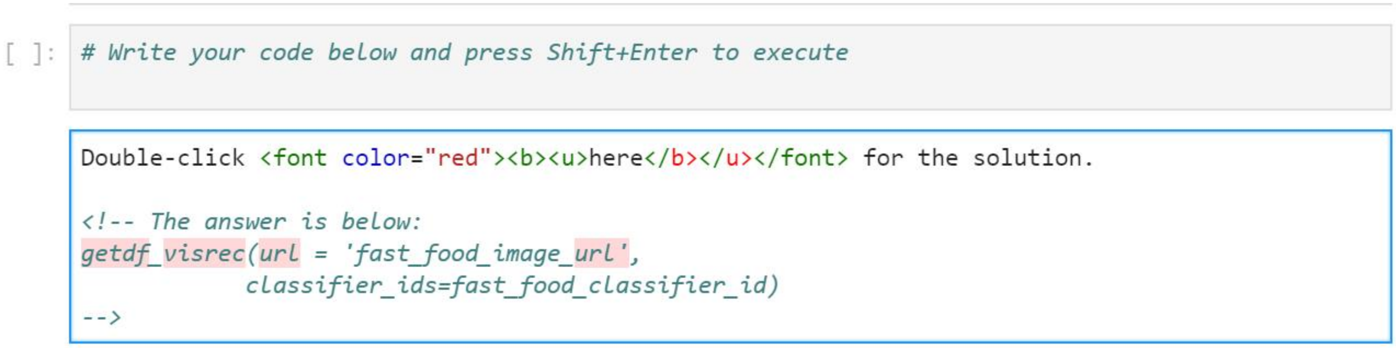
Get a url of a picture of fast food and use the **getdf_visrec** function to classify the picture. **Before that, please make sure that your model is trained and ready**

```
[ ]: # Write your code below and press Shift+Enter to check if your model is ready to go
```

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
gStatus = visrec.get_classifier(classifier_id=fast_food_classifier_id,
verbose=True).get_result()['status']
if Status=='training':
    print ("Please, Wait to complete training.......")
```

CODE ;

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
gStatus = visrec.get_classifier(classifier_id=fast_food_classifier_id, verbose=True).get_result()['status']
if Status=='training':
    print ("Please, Wait to complete training.......")
else:
    print("Good to go ")
-->
```

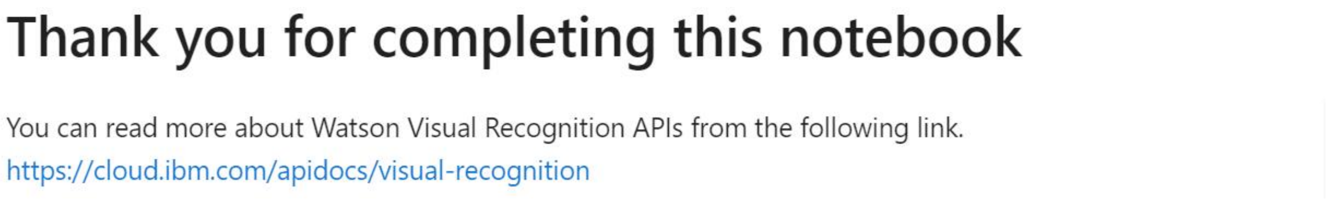


```
[ ]: # Write your code below and press Shift+Enter to execute
```

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
getdf_visrec(url = 'fast_food_image_url',
            classifier_ids=fast_food_classifier_id)
-->
```

CODE ;

```
Double-click <font color="red"><b><u>here</b></u></font> for the solution.

<!-- The answer is below:
getdf_visrec(url = 'fast_food_image_url',
        classifier_ids=fast_food_classifier_id)
-->
```



Thank you for completing this notebook

You can read more about Watson Visual Recognition APIs from the following link.
https://cloud.ibm.com/apidocs/visual-recognition