(/)

LOGOUT

OPEN HACK ENVIRONMENT OVERVIEW

OPEN HACK GUIDE PROVIDE FEEDBACK

1 2 3
4 5 6

Mark Complete

Challenge 1: Establishing Base Camp

Background

Before working on a computer vision solution, the data science team at Adventure Works wants to standarize on a collaborative development environment.

After provisioning the development environment, you can use it to explore the images that Adventure Works has collected, and prepare them for use in training a machine learning model.

Challenge

(/)

LOGOUT

This challenge consists of three tasks:

1. Provision a Data Science Virtual Machine in Azure
2. Explore the JupyterHub Environment
3. Prepare Image Data for Machine Learning

Each task includes some detailed requirements and hints to help you. Additionally, there's a **References** section at the bottom of this page with links to useful resources.

1. Provision a Data Science Virtual Machine in Azure

In this OpenHack, you will use the Azure-based Data Science Virtual Machine (DSVM) as a shared development environment for your team. This virtual machine image includes essential data science tools, including the Jupyterhub notebook environment; in which you will create and run Python code.

To set up the environment, one team member should sign into the Azure portal (https://portal.azure.com) (check with your coach to determine the Azure login credentials you should use), and create a single Azure Data Science Virtual Machine (DSVM) for the team. The following DSVM configuration has been found to work well, and is the recommended environment for this OpenHack:

- DSVM Image: *Data Science Virtual Machine for Linux (Ubuntu)*
- Region: *(Ask your coach for the appropriate region for your OpenHack)*
- Size: *NC6* Standard (GPU Family)
- Authentication type: *Password*
- Username: *(Specify a **lowercase** user name of your choice)*
- Password: *(Specify a complex password)*

(/)

After the DSVM has been created, connect to Jupyterhub and log in using the username and password you specified when provisioning the DSVM.

Hints

- To conserve subscription resources, create ***one*** DSVM for the team.
- When provisioning the DSVM, specify a **lowercase** user name and be sure to choose **Password** as the authentication type. All team members will be able to use these credentials to connect to Jupyterhub on this virtual machine - for information about using Jupyterhub, see this video (https://www.youtube.com/watch?v=4b1G9pQC3KM) or this document (https://docs.microsoft.com/azure/machine-learning/data-science-virtual-machine/linux-dsvm-walkthrough/?wt.mc_id=OH-ML-ComputerVision#jupyterhub).
- Jupyterhub is at **https://*your.dsvm.ip.address*:8000**
- To get to the Jupyterhub, you must click through the non-private connection warnings in browser - this is expected behavior.
- If Jupyterhub takes a while to load, click the **jupyter** logo to open the folder tree page.

See the **References** section below for more guidance and help.

2. Explore the JupyterHub Environment

In the Jupyterhub folder tree, note that there are already folders and notebooks that you can use to learn about various data science frameworks and technologies. Then create a new folder named **openhack** in which to store the code and other files you will use in this hack.

(/)

LOGOUT

After you've created your folder, create a new terminal session, which should open with the working directory set to your home folder (*username*@*dsvm*:/data/*username*). The terminal shell is a useful way to enter operating system (OS) commands.

Enter the following command to change the current directory to the folder you created:

```
cd notebooks/openhack
```

Now enter the following command to download a notebook file to this folder:

```
curl -O https://computervisionhack.blob.core.windows.net/challengefiles/
DataPrep.ipynb
```

After the file has been downloaded, switch back to the tab containing the folder tree, browse to your **openhack** folder, and verify that the **DataPrep.ipynb** notebook has been downloaded.

Hints

- Use the **New** drop-down button to create new folders, notebooks, terminal sessions, and other items.
- When you create a new folder, it is initially named ***Untitled Folder***. To rename it, select the checkbox to its left and then click **Rename** at the top of the **Files** tab.

3. Prepare Image Data for Machine Learning

Open the **DataPrep.ipynb** notebook and examine the notes and code it contains. Run each code cell, and review the output. The code in the notebook:

1. Downloads and extracts a folder hierarchy of image files that you will use in subsequent challenges.

(/)

2. Displays the first image in each folder - each folder represents a category or *class* of product image.
3. Standardizes the images so that they are a common format and size.

LOGOUT

> **Note**: In this challenge, the code has been provided for you to enable you to get familiar with the Jupyter notebook environment. However, you should take the time to review the code and ensure you understand it, because in later challenges you will need to write your own code to perform similar tasks!

Hints

- The **os** Python module (https://docs.python.org/3.6/tutorial/stdlib.html#operating-system-interface) includes functions for interacting with the file system.
- The **matplotlib** Python library (https://matplotlib.org/2.0.2/index.html) provides functions for plotting visualizations and images.
- To ensure that plots are displayed in a notebook, you must run the following *magic* command before creating the first plot:

```
%matplotlib inline
```

- Images are essentially just numeric arrays. In the case of color images, they are three-dimensional arrays that contain a two-dimensional array of pixels for each color channel. For example, a 128x128 Jpeg image is represented as three 128x128 pixel arrays (one each for the red, green, and blue color channels). The Python *NumPy* library (https://docs.scipy.org/doc/numpy/reference/arrays.html)

(/)

LOGOUT

provides a great way to work with multidimensional arrays. For example, you can use:

  - `numpy.array(my_img)` to exlicitly convert an image object to a numpy array.
  - `my_array.shape` to determine the size of the array dimensions - an image has three dimensions (height, width, and channels)

- There are several Python libraries for working with images, as noted in the **References** section. You can use whatever combination of these packages works best to process your images, and rely on the **numpy** array data type as an intermediary format.
- The *PIL* library (https://pillow.readthedocs.io) uses a native format for images, but you can easily convert PIL images to numpy arrays using the `numpy.array()` function, and you can convert a numpy array to a PIL Image object by using the `Image.fromarray()` function. You can also convert PIL images between image formats (for example, from a 4-channel PNG to a 3-channel JPG) using the `my_img.convert()` function.
- To open a file as a PIL Image object, use the `Image.open()` function. To save a PIL image as a file, use the `my_img.save()` function.
- A common strategy to resize an image while maintaining its aspect ratio is to:

  1. Scale the image so that its largest dimension (height or width) is set to the target size for that dimension. You can use the PIL `my_image.thumbnail()` method to accomplish this.
  2. Create a new image of the required size and shape with an appropriate background color. You can use the PIL `Image.new()` function to accomplish this.

(/)

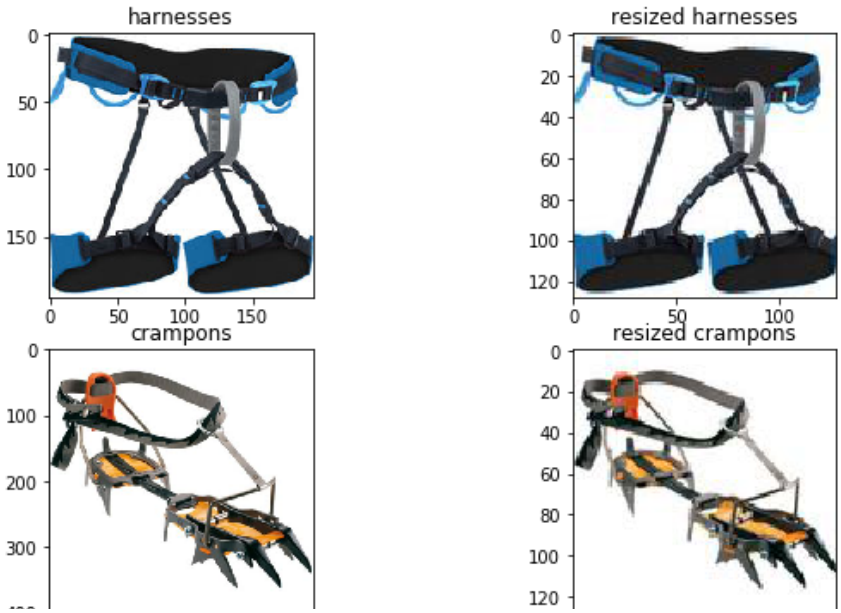
3. Paste the rescaled image into the center of the new background image. You can use the PIL `my_bg_img.paste()` function to accomplish this.

LOGOUT

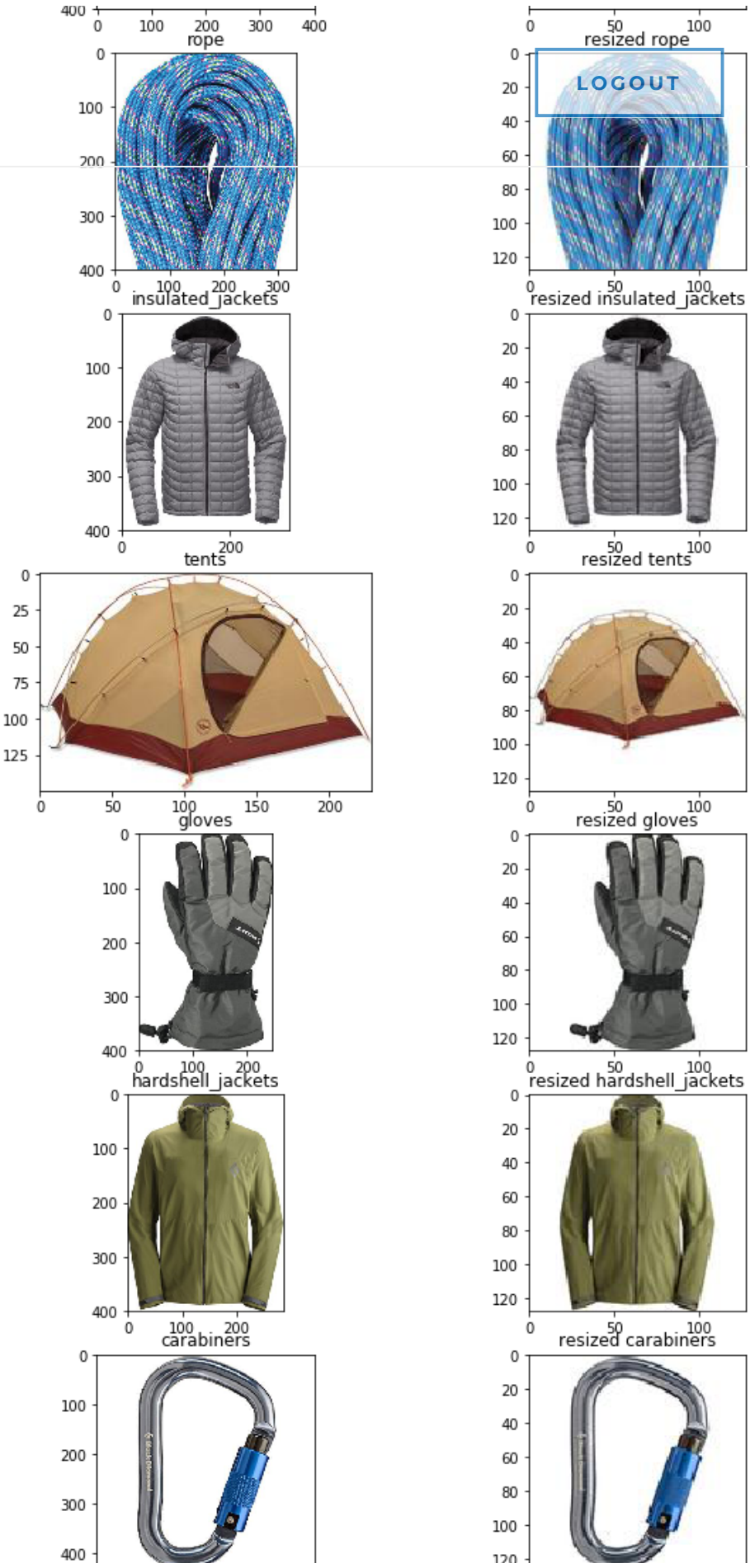
- When using *matplotlib* (https://matplotlib.org/2.0.2/users/image_tutorial.html) to plot multiple images in a grid format, create a figure and add a subplot for each image by using the `my_figure.add_subplot()` function. The parameters for this function are:
  - The total number of *rows* in the grid.
  - The total number of *columns* in the grid.
  - The *ordinal position* of this subplot in the grid (starting with 1 in the top-left cell).

Success Criteria

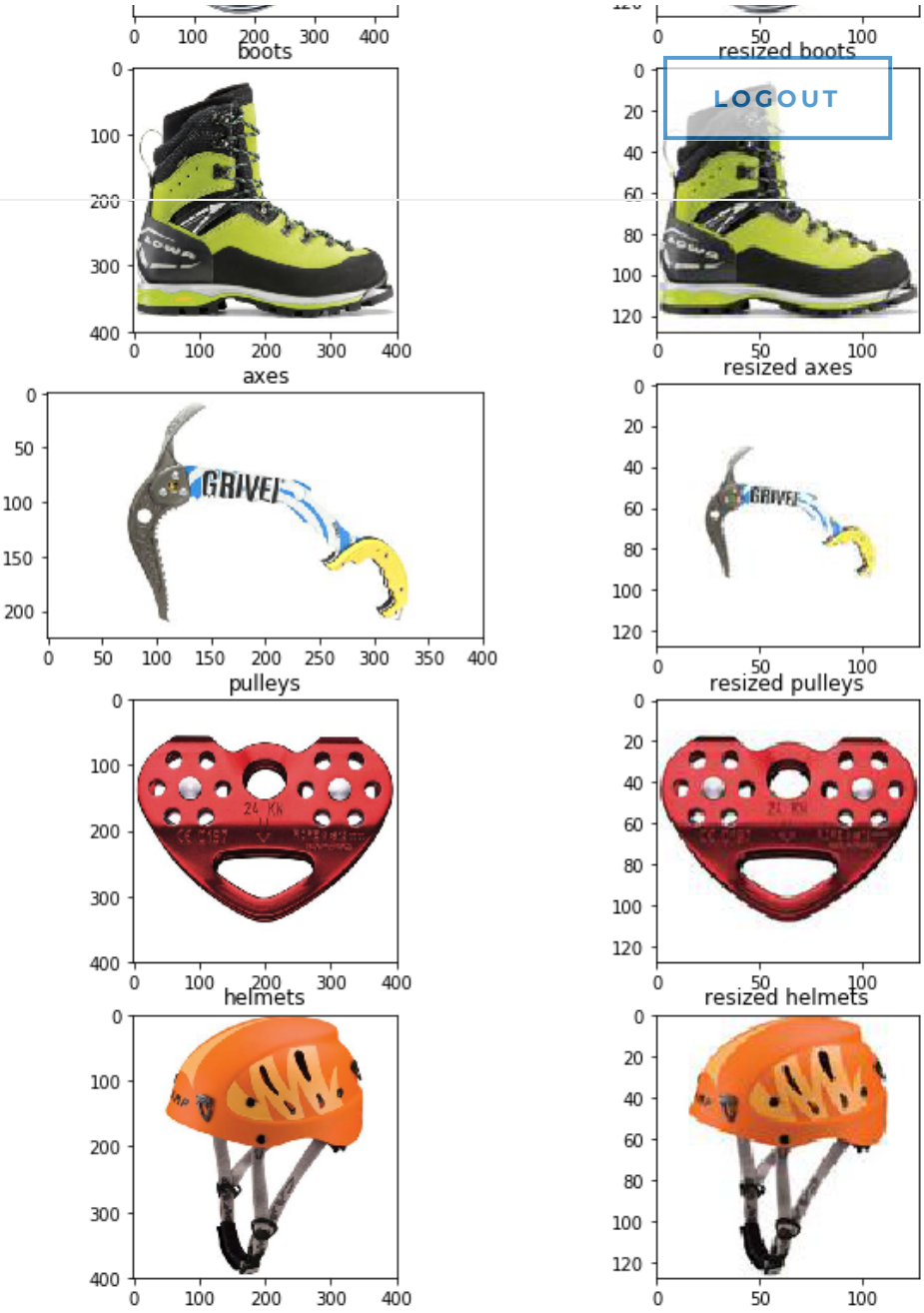
To complete this challenge successfully, you must run the code in the **DataPrep.ipynb** notebook in the Jupyterhub environment hosted by your DSVM instance. The final code cell in the notebook should display the original and resized version of the first image in each folder, similar to the following:



(/)


rope
resized rope
LOGOUT
insulated_jackets
resized insulated_jackets
tents
resized tents
gloves
resized gloves
hardshell_jackets
resized hardshell_jackets
carabiners
resized carabiners

(/)



Gear Images

References

Data Science Virtual Machine

- Provisioning a Ubuntu DSVM (https://docs.microsoft.com/en-us/azure/machine-learning/data-science-virtual-machine/dsvm-ubuntu-intro)

(/)

LOGOUT

Jupyterhub and Notebooks

- Jupyterhub in the DSVM (https://docs.microsoft.com/en-us/azure/machine-learning/data-science-virtual-machine/linux-dsvm-walkthrough#jupyterhub)
- Jupyterhub documentation (https://jupyterhub.readthedocs.io/en/stable/)
- Jupyter Notebooks documentation (https://jupyter-notebook.readthedocs.io/en/latest/)

Python Fundamentals

- Python 3.6 documentation (https://docs.python.org/3.6/)
- NumPy User Guide (https://docs.scipy.org/doc/numpy/user/index.html)
- The Python Debugger (https://docs.python.org/3.6/library/pdb.html)

Image Processing

- Using *matplotlib* for image I/O and plotting (https://matplotlib.org/2.0.2/users/image_tutorial.html)
- Using the *PIL Image* module for I/O and more (https://pillow.readthedocs.io/en/5.3.x/reference/Image.html)
- Using the *PIL ImageOps* module for image manipulation (http://pillow.readthedocs.io/en/5.3.x/reference/ImageOps.html)
- Using *NumPy* for image I/O (https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.io.html)
- Using *NumPy* for image manipulation/processing/visualization (http://www.scipy-lectures.org/advanced/image_processing/)
- *OpenCV* - another library for image processing (https://opencv.org/) (*Note: In **opencv** images are read in a*

(/)

*BGR format, whereas **matplotlib** reads and expects images as RGB. Conversion information can be found here (https://www.scivision.co/numpy-image-bgr-to-rgb/).)*

LOGOUT