Support Vector Machin:

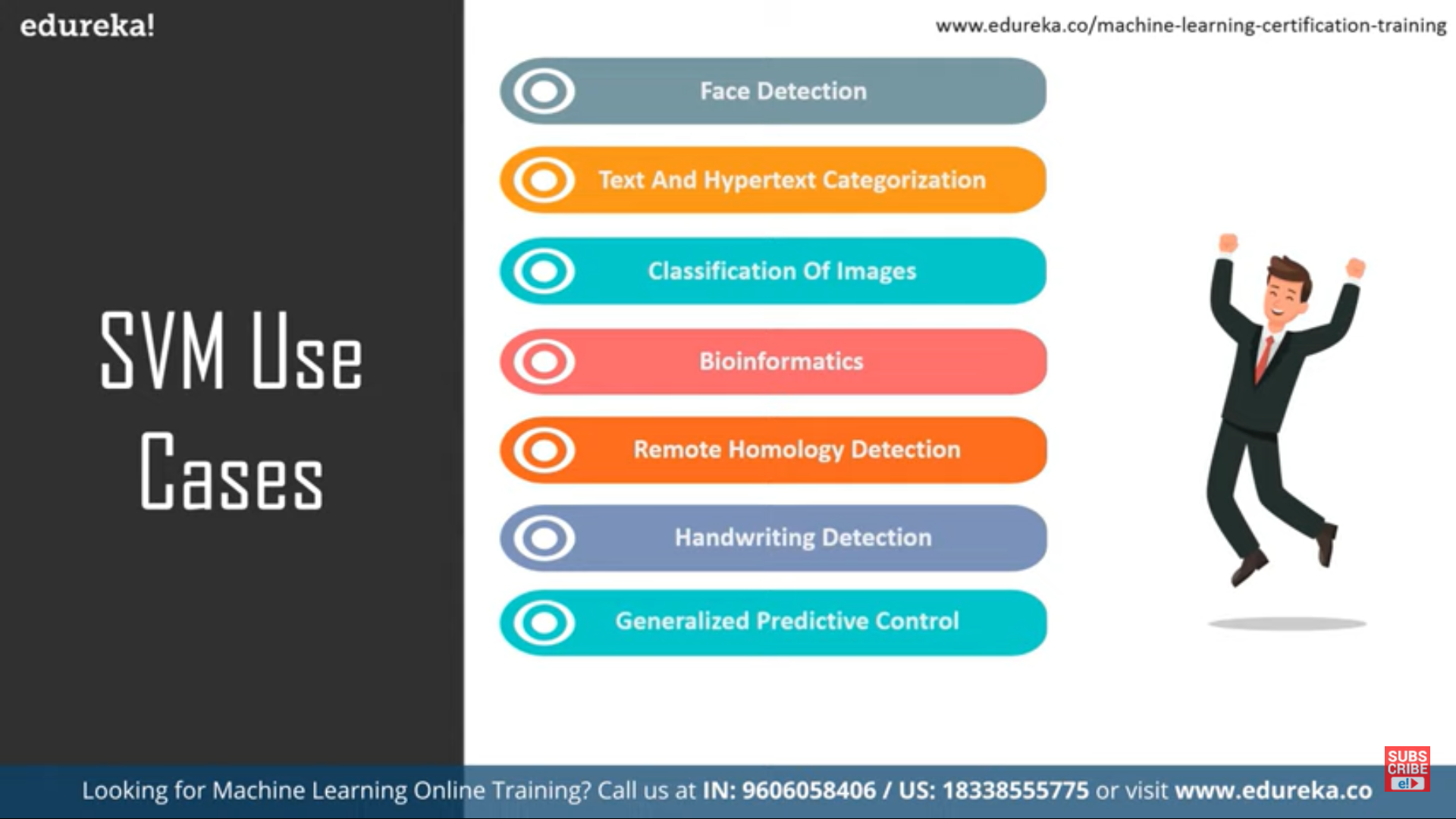
* Used for segregating the given data in the best possible way i.e. classification, regression, out layer detection.
* It is a discriminative classifier designed by***separative hyperplane***.
* It can also perform non-linear classification.
* **A:** It is effective in high dimensional spaces and in cases where number of dimensions greater than the number of the samples.
* **A:** Memory efficient since uses subset of training points in the function and is versatile.
* **DA:** If number of features is much larger than the number of the samples, we have to avoid over fitting in choosing the kernel function (regularisation term is crucial).
* **DA:** do not directly provide probability estimates (have to be calculated using 5 fold cross validation.
* SVM Kernels: used to add more number of dimensions to a lower dimension space (uses kernel trick to do so). Different types of kernels:

Linear: normal dot product of each pair of input values.

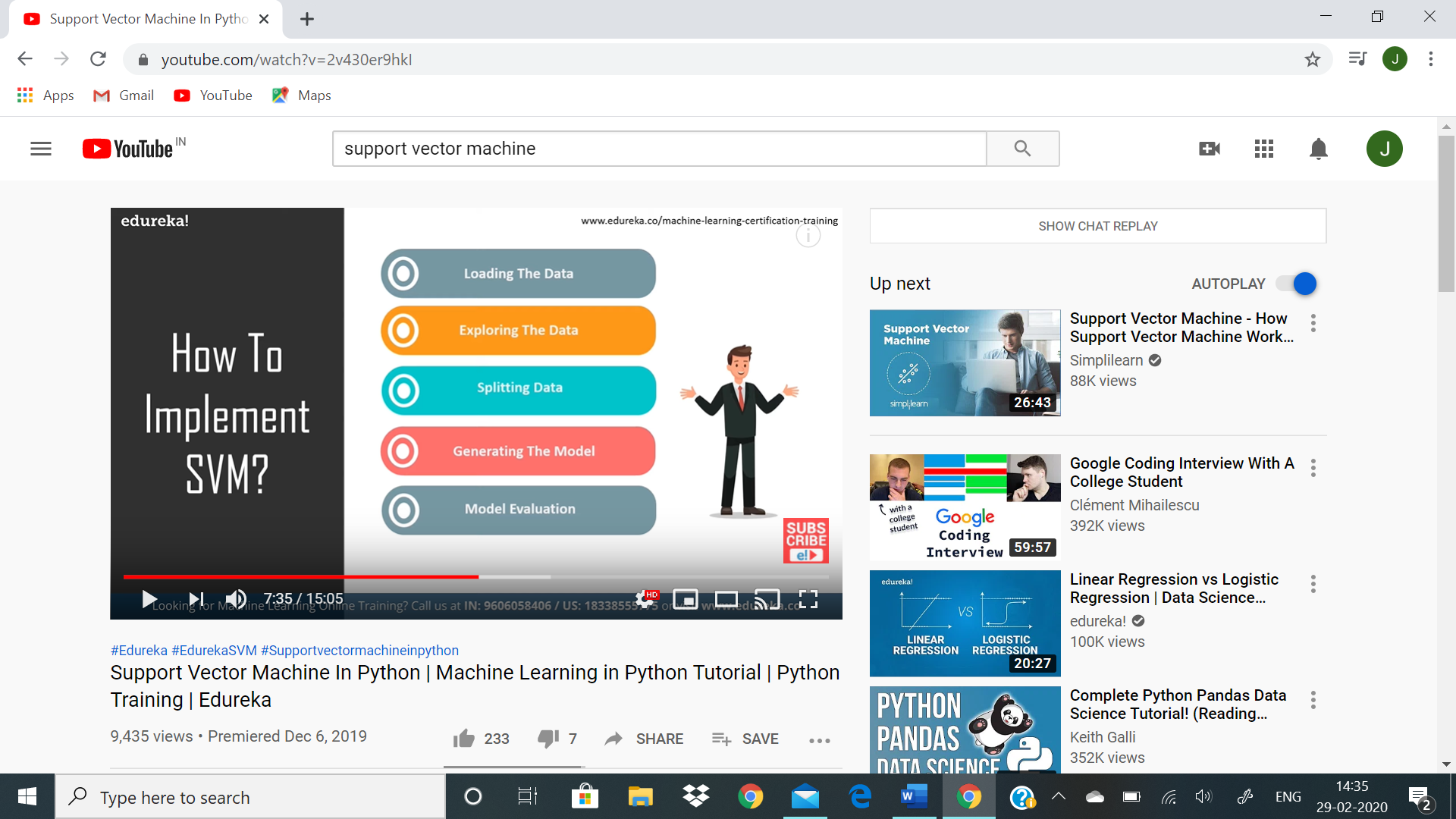
Radial Based Function: Commonly used since it can map the space in infinite dimensions.

Polynomial: Generalised for of linear kernel can also distinguish non linearity such as curved.

* Used cases:



* How to Implement:



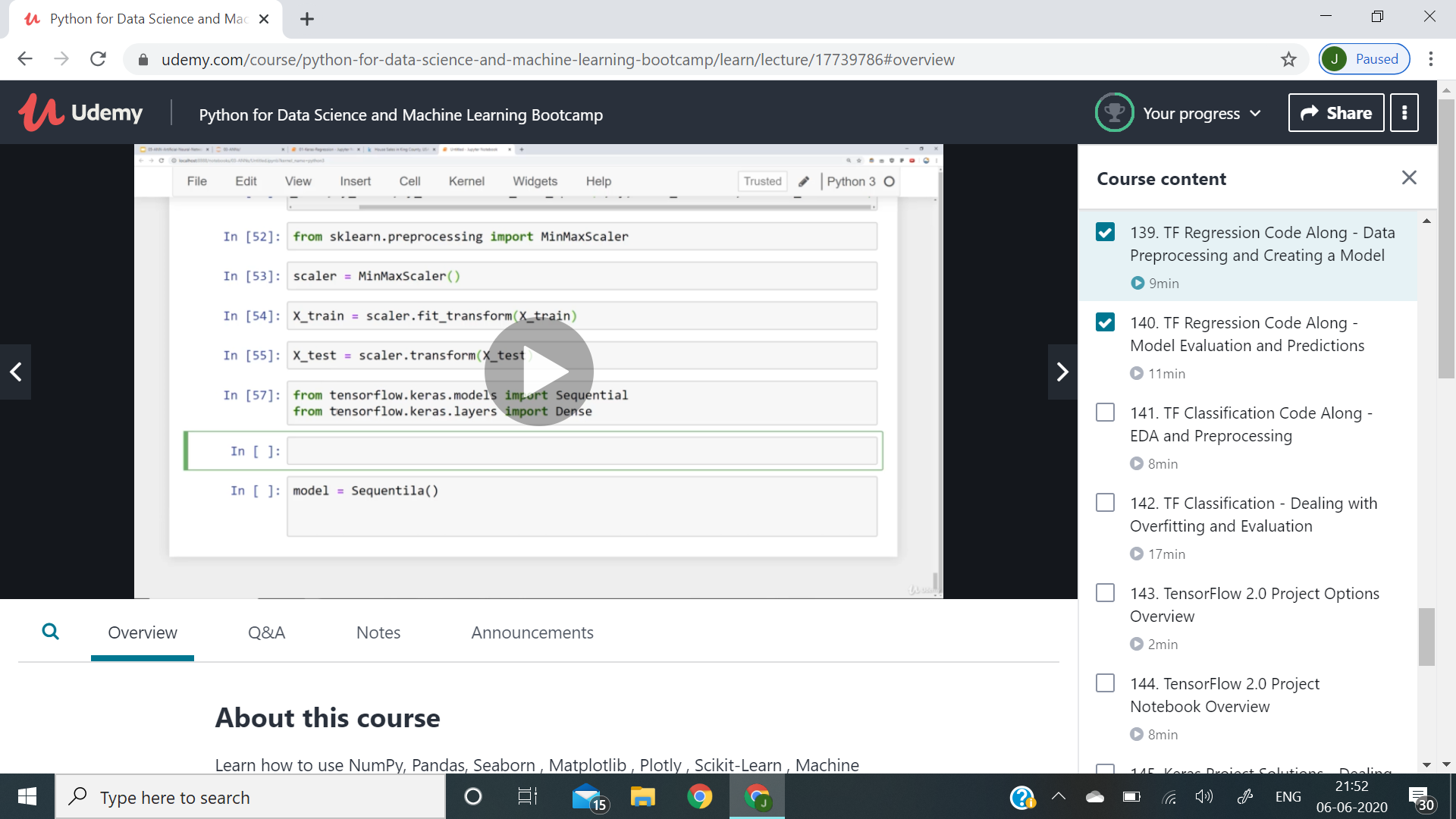
* PACKAGES INSTALLED etc:

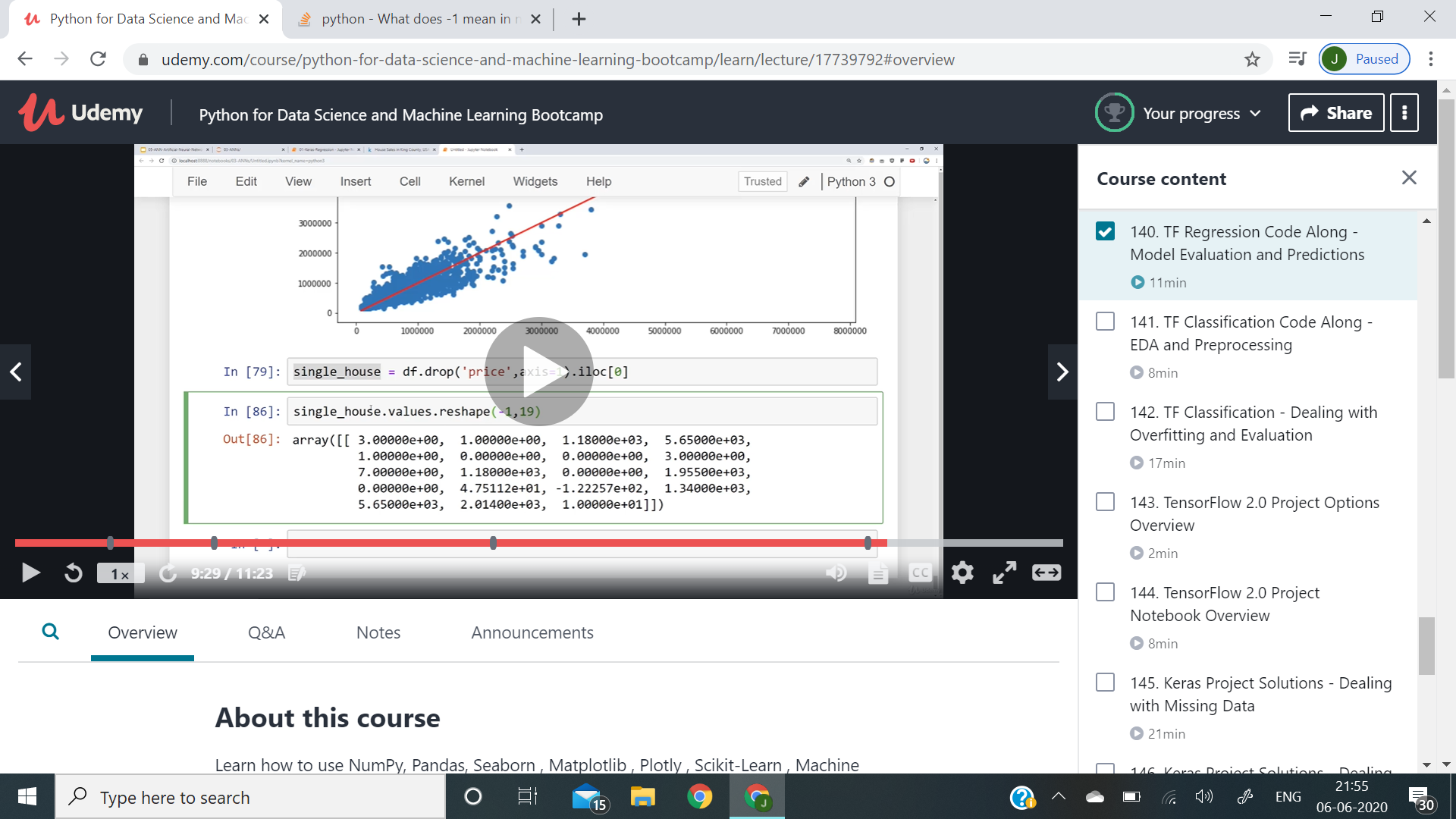
Yfinance:2

DataReader:2

Chart Studio:2

Cufflinks:2





<body>

<div class="container">

<form method="POST">

<h1>Please sign up here!</h1>

<label for="F\_name">First Name:</label>

<input type="text" name="F\_name" value="">{{form.F\_name.as\_p}}</input><br>

<label for="L\_name">Last Name:</label>

<input type="text" name="L\_name" value="">{{form.L\_name.as\_p}}</input><br>

<label for="Email">Email:</label>

<input type="text" name="Email" value="">{{form.Email.as\_p}}</input><br>

{% csrf\_token %}

<input type="submit" class="btn btn-primary" value="Submit">

</form>

</div>

</body>

# Definition for a binary tree node.

# class TreeNode:

# def \_\_init\_\_(self, val=0, left=None, right=None):

# self.val = val

# self.left = left

# self.right = right

master=[]

class Solution:

def levelOrderBottom(self, root: TreeNode) -> List[List[int]]:

global master

slave=[]

p=root

if root is None:

return master

elif root.left==None and root.right==None:

slave.append(root.val)

master.insert(0,slave)

else:

slave.append(root.val)

master.insert(0,slave)

self.levelOrderBottom(root.left)

self.levelOrderBottom(root.right)

return master

<https://github.com/sersavn/coursera-python-for-everybody-specialization>

<https://github.com/diskandarnerd/deeplearningcourseraspecialization/blob/master/Building_your_Deep_Neural_Network_Step_by_Step_v8a.ipynb>

<https://github.com/viritaromero/Plant-diseases-classifier>

50GRE\_COVID-efd21

<https://applicants.bairesdev.com/home/openings>

Conversion to h5 format :-

<https://realpython.com/storing-images-in-python/>

import h5py

def store\_single\_hdf5(image, image\_id, label):

""" Stores a single image to an HDF5 file.

Parameters:

---------------

image image array, (32, 32, 3) to be stored

image\_id integer unique ID for image

label image label

"""

# Create a new HDF5 file

file = h5py.File(hdf5\_dir / f"{image\_id}.h5", "w")

# Create a dataset in the file

dataset = file.create\_dataset(

"image", np.shape(image), h5py.h5t.STD\_U8BE, data=image

)

meta\_set = file.create\_dataset(

"meta", np.shape(label), h5py.h5t.STD\_U8BE, data=label

)

file.close()

How to host a website:

<https://www.wpbeginner.com/beginners-guide/how-to-host-a-website/>

<https://www.namecheap.com/hosting/how-to-host-website/>

