****

# Session 6: Putting it all Together

## Why This Meetup?

* To learn about all the infrastructure that goes into setting up our machines to be able to do distributed data processing and machine learning in the cloud
* To learn data processing in Scala and implement machine learning models with Tensorflow

## Introduction

Hey, thanks to coming to the final session of this series! Today we’ll be putting together the pieces from all the last meetups. Don’t worry if you haven’t been to any of the past ones though, this document will help guide you through everything. The only requirement for being here is **having an AWS account setup** and being excited to learn about really awesome things!

We’re going to be throwing a lot of things at you today some of which including:

* Spark: distributed data processing and preparation
* AWS Elastic Computing 2: cloud computing
* Zeppelin Notebook: developing with spark and PySpark
* Kubernetes: quick and easy deployment of software
* TensorFlow: developing neural network models

Don’t worry if some of those terms are unfamiliar to you, that’s why you’re here! If you have questions or get stuck throw your hand up and make intense eye contact with someone in a DeepLearni.ng shirt, we’re here to help you!

## The Setup

### 1. Setting up our Virtual Machine

Before we can start building our models we are going to need some computing power. Let’s set up a virtual machine to help us out with that. Even if you have some version of Ubuntu already on your computer we still require that you get a VM up.

1. Go to <https://www.ubuntu.com/download/server> and download the first link, Ubuntu Server 16.04.3 LTS
2. Open up VirtualBox
3. Click ‘New’ at the top left of the screen
4. Type ‘ubuntu’ as the name, select ‘Linux’ as the type and ‘Ubuntu (64-bit)’ as the version
5. Type ‘512’ for memory and click ‘Continue’
6. Click ‘Continue’ a few time and then click ‘Create’
7. Right click on the VM you just created and click ‘Settings’
8. Click on the ‘Storage’ tab and in the left bar click the little CD icon with a plus sign that’s on the same line as ‘Controller: IDE’ then select the .iso file you downloaded in step 1A
9. Double click on your VM to start it up.
10. Select your Language and select ‘Install Ubuntu Server’
11. Keep clicking through all of the default settings. If you get caught in a loop between 2 pages select ‘Yes’ on the page with that option
12. When you hit ‘[ ! ] Software Selection’ scroll down with your keyboard and by hitting the spacebar select ‘OpenSSH server’. Leave the default one selected and click ‘Continue’
13. Install the Grub boot loader and keep clicking through the default options
14. You should now be able to log in to your virtual machine

### 2. Preparing our VM

Now that we have our virtual machine we want to get the files from gitlab and run some shell scripts we have built to make set-up easier.

1. Type `sudo apt-get update`
2. Type `sudo apt-get install git`
3. Type `git clone <https://gitlab.com/deeplearni.ng/practical-learnings.git>`

* You will be asked to generate a personal access token. Create one at <https://gitlab.com/profile/personal_access_tokens>. Once you get the code retype step C and enter it as your password(remember that you can’t paste in your VM so enter it manually).

1. Type ‘cd practical-learnings/session6’

### 3. Preparing for AWS

We have pre-built some shell scripts to make things easier. We will need to configure some things with our AWS account and run the script.

1. Go to <https://console.aws.amazon.com/iam/home#/security_credential>
2. Click ‘Access Keys (Access Key ID and Secret Access Key)’
3. If you don’t already have an Access Key Create one.
4. Save the downloaded file.
5. In the terminal on your machine type ‘cat <PATH-TO-FILE-DOWNLOADED-IN-3D>
6. In your virtual machine type ‘sh prepare.sh’
7. Type in the Access Keys and the Secret Key when prompted
8. Region is `us-east-1`
9. Default output is `json`

### 3. Running and Scaling on AWS

Another pre-built script called ‘deploy.sh’ will be deploying our stuff to AWS. We’re using something called Ansible in a lot of these scripts which is a tool that automates a lot of the devops that we would otherwise have to do ourselves.

1. Type ‘sh deploy.sh’
2. Go to <https://console.aws.amazon.com/ec2>
3. On the left sidebar you will see ‘Auto Scaling Groups’ under ‘Auto Scaling’, click it
4. Select your Auto Scaling Group and click ‘Edit’ in the Details tab below
5. Set the Desired and Max fields to 1.

* **This is where you start getting charged. Make sure to stick around because towards the end we’ll make sure that you kill everything you make now so that you don’t get charged for it.**

### 5. Looking up our Instances

Now we’re going to create instances with the master and worker templates that we set up above.

1. Go to the Instances (left bar) and click “Instances”
2. Wait for a bit to make sure that the that your instances pop up
3. Click the “meetup-session-6-k8s-master” instances and select it. Copy the IPv4 Public IP Address from the “Details” tab below

### 6. SSHing into our Instances

Let’s go into the instance that we just created.

1. Open your Virtual Machine terminal again
2. Type `chmod 600 files/id\_meetup`
3. Type `ssh -i files/id\_meetup ubuntu@<IP-ADDRESS-COPIED-IN-STEP-5D>`

### 7. Deploying the Kubernetes Dashboard

Let’s deploy the dashboard so we can access our notebook!

1. Become root by typing `sudo su`
2. Type `export KUBECONFIG=/etc/kubernetes/admin.conf`

* You will have to type this in every time you log in to the machine
* If this file does not exist, your kubernetes cluster might not be ready yet. Check again in a minute or so

1. Type `kubectl create -f /etc/zeppelin/kubernetes/canal.yaml`

* This, like the previous meetup, will install a pod network into your cluster

1. Type `kubectl create -f /etc/zeppelin/kubernetes/dashboard.yaml`

Note:

* It may take a while before kubernetes is ready. You can type `kubectl get nodes` to see if kubernetes has started. If it has not started, you will see at least one node with the state ”NotReady”.
* Once you have done step 7C the nodes states will change to “Ready”.
* It may take a while for the dashboard to become ready. In order to see the status of everything in kubernetes you will have to type: `kubectl get po --all-namespaces -o wide`. When it’s ready the status of every item should be “Running”

### 8. Deploying our Spark/TensorFlow Web Application

1. Type `kubectl create -f /etc/zeppelin/kubernetes/zeppelin.yaml`
2. Type `kubectl create -f /etc/zeppelin/kubernetes/spark-master.yaml`
3. Type `kubectl create -f /etc/zeppelin/kubernetes/spark-slave.yaml`
4. You can now type `kubectl get pods -o wide` and copy the any row from the NODE column. Paste this into the searchbar in EC2 under instances. Click on the instance that it gives you and copy the IPv4 Public IP and paste it into your broswer like this <IPv4-Public-IP>:30001 to get to the Zepplin Notebook or <IPv4-Public-IP>:30000 to get to the Kubernetes Dashboard.

* If you aren’t able to follow this step and access the zeppelin application, you can try both the IPv4-Public-IP from the master and worker instances. One of them is likely to work

## 

## The Notebooks

### 9. Load the Dataset into Zepplin

1. Go to your local terminal and type `git clone <https://gitlab.com/deeplearni.ng/practical-learnings.git>`
2. Click ‘Import Notebook` on your Zepplin application
3. Choose to import as JSON and find the 2 notebooks in `practical-learnings/session6/notebooks`

10. The Dataset

The dataset is taken from the Kaggle competition: *Acquire Valued Shoppers Challenge.* The data file can be found at */data-sets/transactions\_mid.csv*. It is a subset of the original dataset, which is too large for the purpose of this meetup. That said, this small dataset allows you to perform all the necessary pre-processing and modeling steps that you will need on a real-world, much larger dataset.

In this dataset, we are given the shopping records of some customers over the course of at least a year. We would like to predict what the customer is likely to buy next given his/her past shopping records.

11. Prepare Your Data

1. The data preparation, or pre-processing, steps are included in the notebook *Session6-Data\_Preperation*. Click to open it.
2. Click the blue “Save” button if it asks for interpreter binding.
3. Just click “Run all paragraphs” located at the left of the top toolbar and then “OK”, and the data-preprocessing will be done for you.

(Note that it will take a while to initialize Spark on your first run. The blue bar under each cell indicates progress.)

1. Before moving on, let’s walk through the notebook and see what’s happening
   1. This notebook contains a mixture of Scala and Python environments. Cells starting with %spark are Scala code, and those with %pyspark are Python code.
   2. First step (cell 3) is to load the data from the file */data-sets/transactions\_mid.csv*. The data is loaded as a Spark SQL data frame.
   3. Then we get an RDD (Resilient Distributed Dataset) from the data frame, and cast each column to desired data types (cell 4).
   4. Cell 5-6 eliminates infrequent items which appear less than 10 times. We assume that they are not useful for prediction due to the lack of data.
   5. Cell 7-12 further adjust the data format through Spark manipulations and convert to Java RDD
   6. Cell 13-14 prepares the data for Python. Notice the Zeppelin Context variable z, through which we transfer data between languages.
   7. The rest of the code turns the RDD data into Numpy format. The data is ready now for the next file.

12. Build Models and Do Prediction

1. Now that we have data in the desired format, we can finally start building ML models and do cool stuff. Click the “Notebook” drop down menu at the top and choose *Session6-Retail Prediction*.
2. Click the blue “Save” button if it asks for interpreter binding.
3. Click “Run all paragraphs” located at the left of the top toolbar and then “OK”. The training process will take a couple minutes.
4. Again, let’s take a closer look at the code.
   1. First of all (and it is very important), you need to **collect** the data from the RDD that you prepared in the last file. A common mistake is to omit the collecting step, and later find your data to be empty.
   2. We then set the model hyperparameters. Hyperparameters here include:

* Learning rate: how “fast” you do gradient descent
* Epochs: number of complete training cycles
* Batch size: number of rows in dataset on which gradient is computed at each time
* Hidden layer size: the size of the hidden layer

Here we generate two distinct sets of hyperparameters, and mimic a simple parallelized hyperparameter search process. In reality, the choice of hyperparameters directly determines how well your model performs, and your search space is much larger.

* 1. We also define some helper functions that split the dataset into model inputs and targets. We will not go into the details of this.
  2. We want to predict which item a customer will by next given his/her past orders. We build and train a neural network with one hidden layer (validation\_and\_model()) and compare the validation performance with a naive benchmark model (validation\_and\_naive\_model()).
  3. The prediction results (validation loss) for both models are displayed at the end of the notebook.

## The Cleanup

### 13. Type this to not throw away money

**If you don’t do this step AWS could end up charging you a whole lot of money. Let’s take down our instances now.**

1. Go back to your virtualBoxVM
2. Type `exit` 2 times
3. Type ‘sh cleanup.sh’

### 14. Go home, we love you