Data Mining with R

Yuwu Chen

HPC @ LSU

Some materials are borrowed from the EXST 7142/7152 data mining courses by Dr. Bin Li at Statistics Dept.

Outline

- Statistical data mining
  - Introduction to statistical learning
  - Review neural network basics
- Neural network with R
- Deep learning with R

What is Statistical Learning

- Statistical learning is the science of learning from the data using statistical methods.
- Statistical learning plays a key role in data mining, artificial intelligence and machine learning.
    - Predict the price of a stock in 6 months from now, on the basis of company performance measures and economic data
    - Predict whether a patient, hospitalized due to a heart attack, will have a second attack based on patient's demographic, diet and clinical measurements.
    - Given a collection of text documents, we want to organize them according to their content similarities

Supervised vs. Unsupervised Learning

- Supervised learning is where both the predictors, $\mathtt{X}_\mathtt{i}$'s, and the response, $\mathtt{Y}_\mathtt{i}$, are observed (e.g. regression / classification)

- In unsupervised learning, only $\mathtt{X}_\mathtt{i}$ are observed (e.g. clustering / market basket analysis)

Estimating Unknown Function f

- Suppose we observe Y_i and $\mathbf{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})$ for $i = 1, 2\ldots, n$
- We believe that there is a relationship between Y and at least one of the X's. So we model the relationship as

$$Y_i = f(\mathbf{X_i}) + \varepsilon_i \quad \text{with} \quad E\{\varepsilon_i\}=0$$

where f is an unknown function and ε is a random error

Why do we estimate f?

- Two main reasons: prediction and inference.
  - Make accurate predictions of Y based on a new value of X.
  - Which particular predictors actually affect the response?
  - Is the relationship positive or negative?
  - Is the relationship a simple linear one or is it more complicated etc.?

How do we estimate `f`?

- Use the training data $\{(X_1, Y_1), (X_2, Y_2), \ldots, (Xn, Yn)\}$ and a statistical method to estimate `f`.
- Two groups of statistical learning methods:
  - Parametric methods
  - Non-parametric methods

Bias Variance Tradeoff

- Two competing forces govern the choice of learning method, i.e. **bias** and **variance.**
- Bias refers to the error that is introduced by modeling a real life problem (which is usually extremely complicated) by a much simpler problem.
  - For example, linear regression assumes that there is a linear relationship between Y and X, which is unlikely in real life.
  - In general, the more flexible/complex a method is, the less bias it will have
- Variance refers to how much your estimate for f would change by if you had a different (test) dataset.
  - Generally, the more flexible/complex a method, the more variance it will have.

Bias Variance Tradeoff

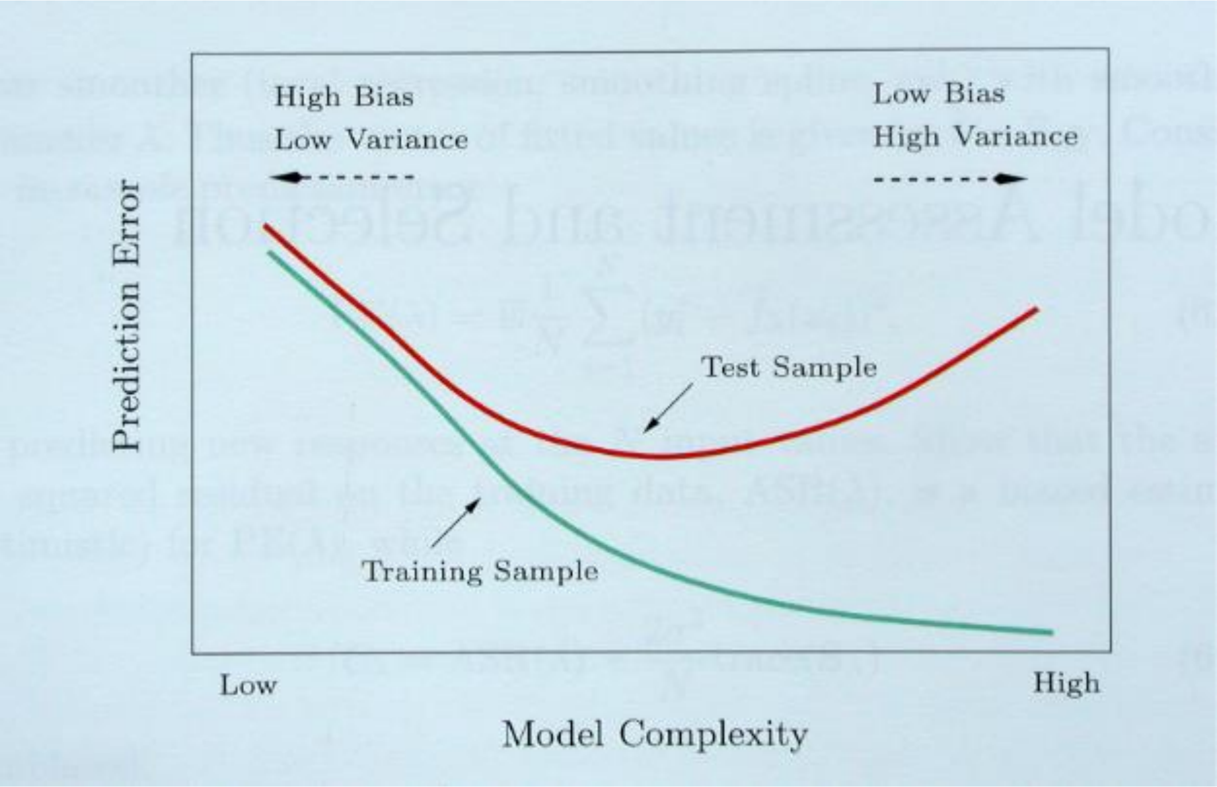


Figure from EOSL 2001

A Cautionary Note

- George Box, a famous statistician and son-in-law of Ronald A. Fisher, once said:

  “All models are wrong, but some are useful.”
- In practice, there is really NO true model but a good model.
- A good model should achieve at least one of the following:
  - an interpretable model that can be explained by some known facts or knowledge;
  - reveals some unknown truth or relationship among the variables or observations;
  - a model with accurate prediction on new samples.

Training Set and Test Set

- Dataset could be randomly split into two parts: training set and test set.

```
> set.seed(1) #set random seed reproducible
> indx <- sample(1:1995,size=1995,replace=F)
> forbes.train <- forbes2[indx[1:1600],]
> forbes.test <- forbes2[indx[1601:1995],]
```

Outline

- Statistical data mining
  - – Introduction to statistical learning
  - – Review neural network basics
- Neural network with R
- Deep learning with R

Biological Inspiration

- Idea: To make the computer more robust, intelligent and learn
- Model our computer software and/or hardware after the brain

Neurons in the Brain

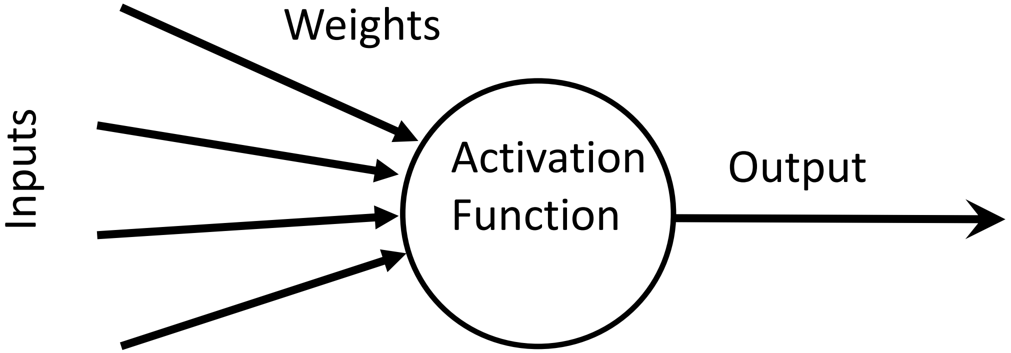
- Although heterogeneous, at a low level the brain is composed of neurons:
  - a neuron receives input from other neurons (generally thousands) from its synapses
  - inputs are approximately summed
  - when the input exceeds a threshold, the neuron sends an electrical spike that travels from the body, down the axon, to the next neuron(s)
- Brain learns:
  - altering strength between neurons;
  - creating/deleting connections.

Neural Network History

- History traces back to the 1950’s but became popular in the 80’s with work by Rumelhart, Hinton and Mclelland.
  - A General Framework for Parallel Distributed Processing: explorations in the microstructure of cognition and foundation for all connectionist models
- Peaked in the 1990s. Today has hundreds of variants.
- Numerous applications:
  - Handwriting, face, speech recognition
  - Vehicles that drive themselves
  - Marketing, risk management and medicine.

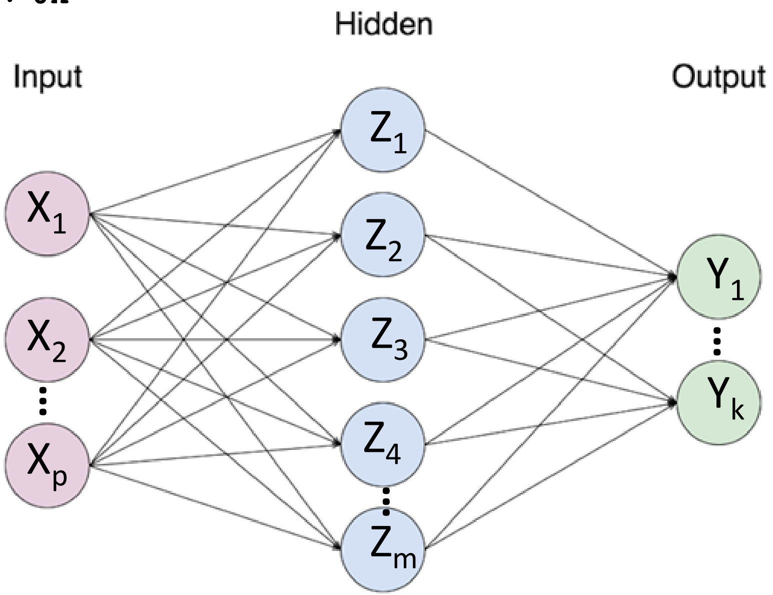
Artificial Neuron

- Like real neurons, artificial neurons basically consist of:
  - inputs (like synapses), which are multiplied by weights (strength of the respective signals), and then computed by an activation function, which determines the activation of the neuron.
  - The output function computes the output of the artificial neuron
- Neural networks combine artificial neurons in order to process information.



Single Hidden Layer Neural Networks

- Inputs: $\mathbf{X}_1, \mathbf{X}_2,\ldots,\mathbf{X}_p$.
- Hidden layer: $\mathbf{Z}_1, \mathbf{Z}_2,\ldots,\mathbf{Z}_m$. Each $\mathbf{Z}_m$ is a modelled as a function of linear combination of inpu
- Outputs: $\mathbf{Y}_1,\mathbf{Y}_2,\ldots\mathbf{Y}_K$. Each $\mathbf{Y}_K$ is a modelled as a function of linear combination of Z_m's.
  - For regression: $\mathbf{K}$=1;
  - K-class classification: one for each class.
- An additional bias unit feed into every unit of hidden and output layers. Captures the intercepts $\boldsymbol{\alpha}_{0m}$ and $\boldsymbol{\beta}_{0k}$ in the model.



Mathematical Model for Artificial Neural Networks

- Derived features Z_m (called hidden unit or hidden node) in the hidden layer is a function of linear combination of inputs

$$Z_m = \sigma(\alpha_{0m} + \alpha_m^T \boldsymbol{X}), m = 1, \ldots, M$$

  - one of the common activations functions σ(v) is the sigmoid: $\sigma(v) = 1/(1+e^{-v})$
  - Many other activation functions
- T_k: a linear combination of Z_m

$$T_k = \beta_{0k} + \beta_k^T \boldsymbol{Z}, k = 1, \ldots, K$$

- The output function:

$$f_k(X) = g_k(\boldsymbol{T}), k = 1, \ldots, K$$

  - For regression, use identity function g(T)=T
  - For classification, use softmax function:

$$g_k(T) = \frac{e^{T_k}}{\sum_{l=1}^{K} e^{T_k}}$$

Fitting Neural Networks

- A single hidden layer neural networks has unknown parameters, called weights. Denote θ as complete set of weights which consists of

$$\{\alpha_{0m}, \alpha_m; m = 1,2,\dots,M\} \qquad \text{M(p+1) weights}$$
$$\{\beta_{0k}, \beta_k; \quad k = 1,2,\dots,K\} \qquad \text{K(m+1) weights}$$

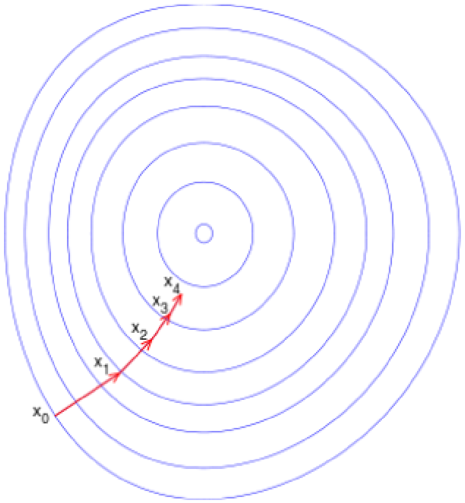
- For regression, use SSE as measure of fit: $R(\theta) = \sum_k \sum_i \left(y_{ik} - f_k(x_i)\right)^2$
- For classification, use cross-entropy:$R(\theta) = -\sum_i \sum_k y_{ik} \log f_k(x_i)$.

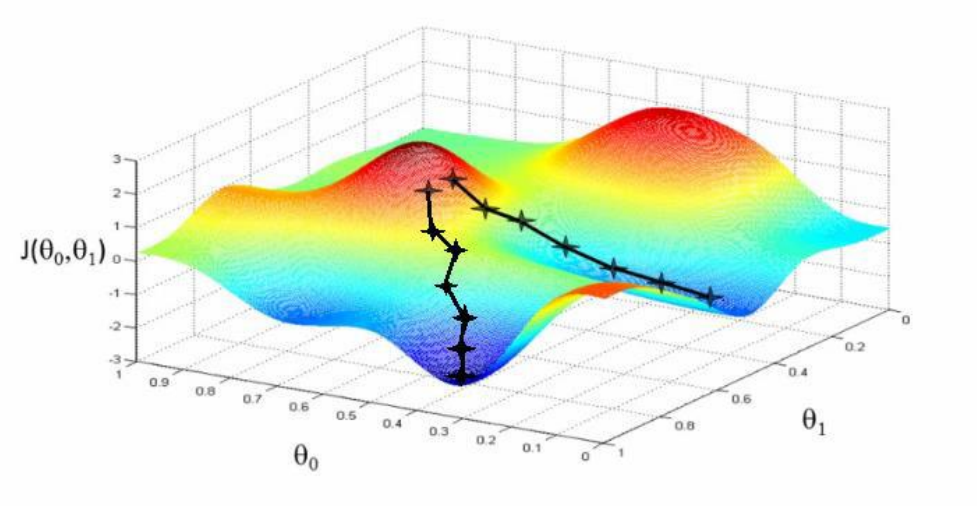
Fitting Neural Networks

- The generic approach to minimizing R(θ) is by **gradient descent**, known as back-propagation, which is simple and local nature.
- It iteratively updates the estimate of weights by a two-pass procedure:
  1. **forward pass**: fixed current weight and predict $\widehat{f_k}(x_i)$
  2. **backward pass**: compute the gradients and update the weights
- **Training** can be carried online or on a parallel architecture computer since back-propagation can be very slow.
- Other choices to minimizing R(θ) are conjugate gradients and variable metric methods with faster convergence.

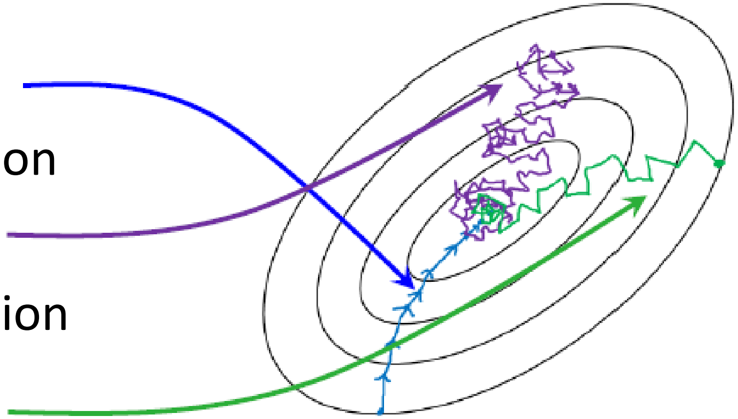
Theory: Gradient Descent

- Gradient descent is a first order iterative optimization algorithm. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or of the approximate gradient) of the function at the current point.





Mini-batch Gradient Descent



- Batch gradient descent:
  - Use all examples in each iteration
- Stochastic gradient descent:
  - Use one example in each iteration
- Mini-batch gradient descent:
  - Splits the training dataset into small batches (size b) that are used to calculate model error and update model coefficients.
- In the neural network terminology:
  - one *epoch* consists of one full training cycle on the training set.
  - Using all your batches once is 1 *epoch*. If you have 10 epochs it mean that you will use all your data 10 times (split in batches).

Training Neural Networks

- Model tends to be over-parametrized (Why?).
- Initial values of weights: random values near zero
- Regularization to avoid overfitting
  - **early stopping**: use a validation set to determine when to stop
  - **weight decay:** add a penalty to loss: $R(\theta) + \lambda\left(\sum \beta_{km}^2 + \sum \alpha_{ml}^2\right)$
  - **weight elimination**: use another penalty:

$$R(\theta) + \lambda\left(\sum \frac{\beta_{km}^2}{1 + \beta_{km}^2} + \sum \frac{\alpha_{ml}^2}{\alpha_{ml}^2}\right)$$

- **Scaling** of the inputs has a large effect on quality of the solution. Standardize all inputs to mean zero and variance one.

Other Usage of Neural Networks

- Radial Basis Function (RBF) architecture is especially suited for clustering. Special units try to catch prototypes for each cluster. Variants like probabilistic neural networks are meant for classification tasks.

- In unsupervised learning the neural network itself tries to find patterns in the data without response Y.

Outline

- Statistical data mining
  - Introduction to statistical learning
  - Review neural network basics
- Neural network with R
- Deep learning with R

Build a Neural Network From Scratch???

- **it is possible but not necessary…**
  - will need define your own functions to load the dataset, set variables, choose the activation function, set cost function, set backward propagation etc..

```
backward_prop <-
function(X,Y,activation_ftn,weights1,weights2,activation1,activation2)
{
  m <- ncol(Y)
  derivative2 <- activation2-Y
  dweights2 <<- (derivative2 %*% t(activation1)) / m
  dbias2 <<- rowSums(derivative2) / m
  upd <- derivative_function(activation_ftn,activation1)
  derivative1 <- t(weights2) %*% derivative2 * upd
  dweights1 <<- (derivative1 %*% t(X)) / m
  dbias1 <<- rowSums(derivative1) / m
}
```

  - if you are developing your own algorithm, you may need to write your own code.

Code from R deep learning essentials by Mark Hodett and Joshua F. Wiley

Neural Network Packages in R

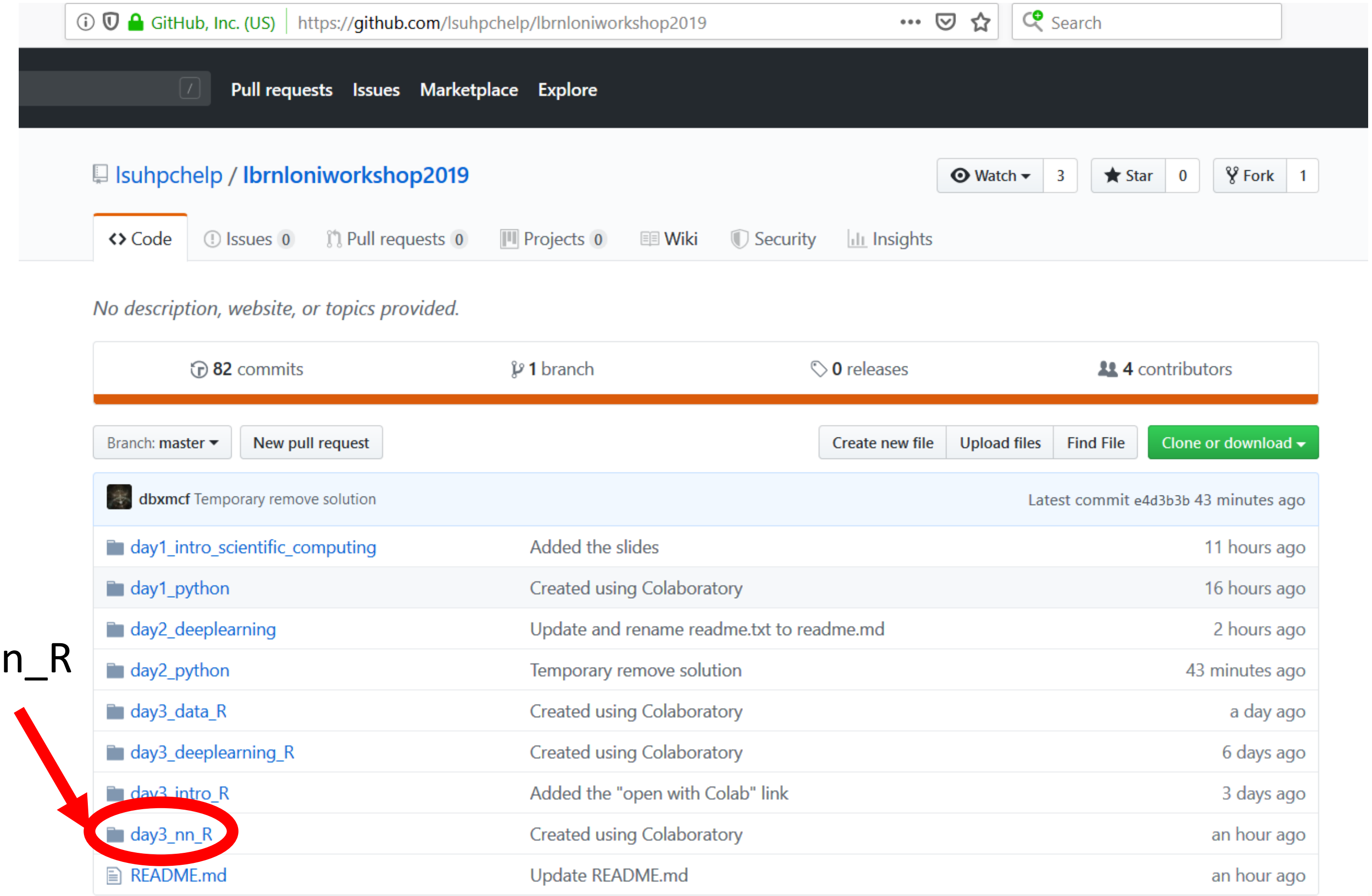
- **nnet** is the R standard neural network packages. It implements a multi-layer perception with one hidden layer. For weight adjustment, it does not use back-propagation, but a general quasi-Newton optimization procedure, the BFGS algorithm.

- **neuralnet** implements the standard back-propagation. It allows multiple hidden layers and units. It provides functions to visualize the fitted networks.

- **RSNNS** is a comprehensive neural network packages in R. It includes several types of NNs with different tasks (can do unsupervised tasks) and has many options for the main functions.

- https://github.com/lsuhpchelp/lbrnloniworkshop2019

day3_nn_R



lsuhpchelp / lbrnloniworkshop2019

No description, website, or topics provided.

82 commits | 1 branch | 0 releases | 4 contributors

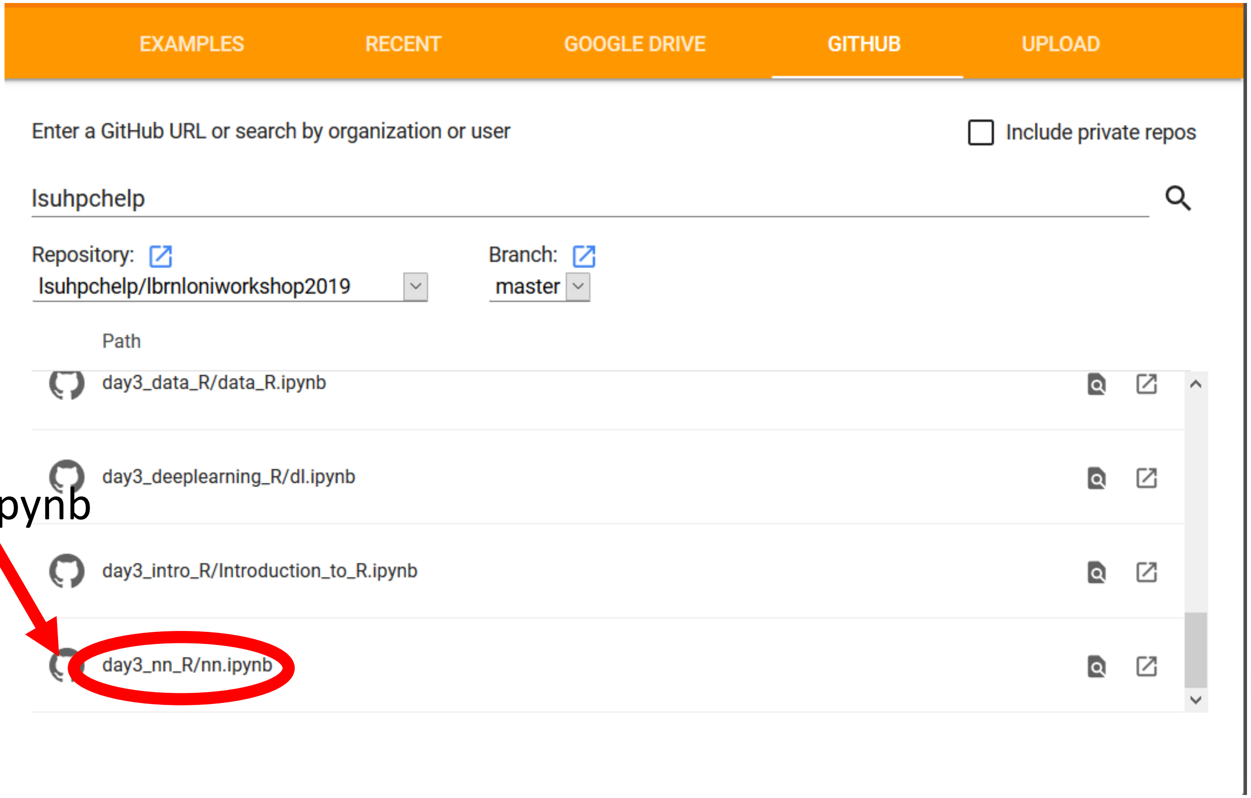
dbxmcf Temporary remove solution — Latest commit e4d3b3b 43 minutes ago

day1_intro_scientific_computing	Added the slides	11 hours ago
day1_python	Created using Colaboratory	16 hours ago
day2_deeplearning	Update and rename readme.txt to readme.md	2 hours ago
day2_python	Temporary remove solution	43 minutes ago
day3_data_R	Created using Colaboratory	a day ago
day3_deeplearning_R	Created using Colaboratory	6 days ago
day3_intro_R	Added the "open with Colab" link	3 days ago
day3_nn_R	Created using Colaboratory	an hour ago
README.md	Update README.md	an hour ago

If you cannot open data_R.ipynb and get “Sorry, something went wrong. Reload?”

Try this link:
https://colab.research.google.com/github/lsuhpchelp

day3_nn_R/nn.ipynb



Enter a GitHub URL or search by organization or user

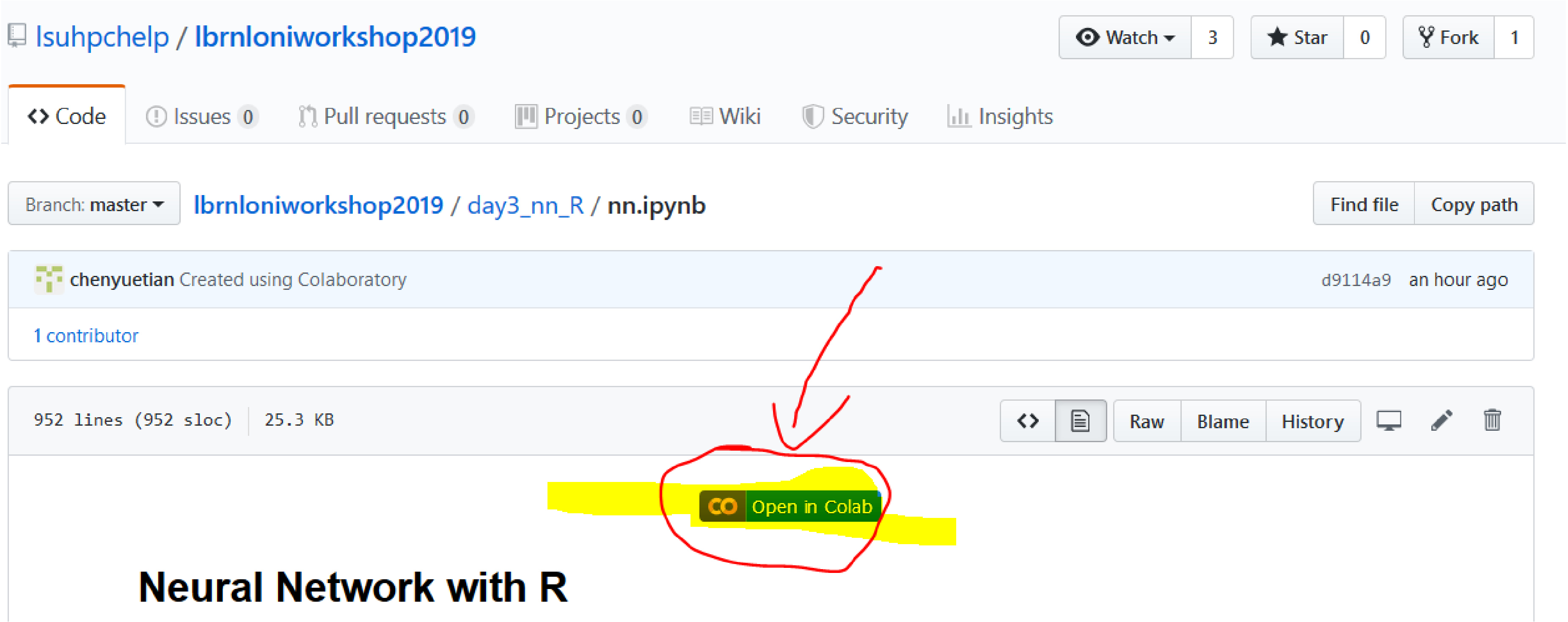
□ Include private repos

lsuhpchelp

Repository: lsuhpchelp/lbrnloniworkshop2019

Branch: master

Path
- day3_data_R/data_R.ipynb
- day3_deeplearning_R/dl.ipynb
- day3_intro_R/Introduction_to_R.ipynb
- day3_nn_R/nn.ipynb



lsuhpchelp / lbrnloniworkshop2019

Branch: master

lbrnloniworkshop2019 / day3_nn_R / nn.ipynb

chenyuetian Created using Colaboratory d9114a9 an hour ago

1 contributor

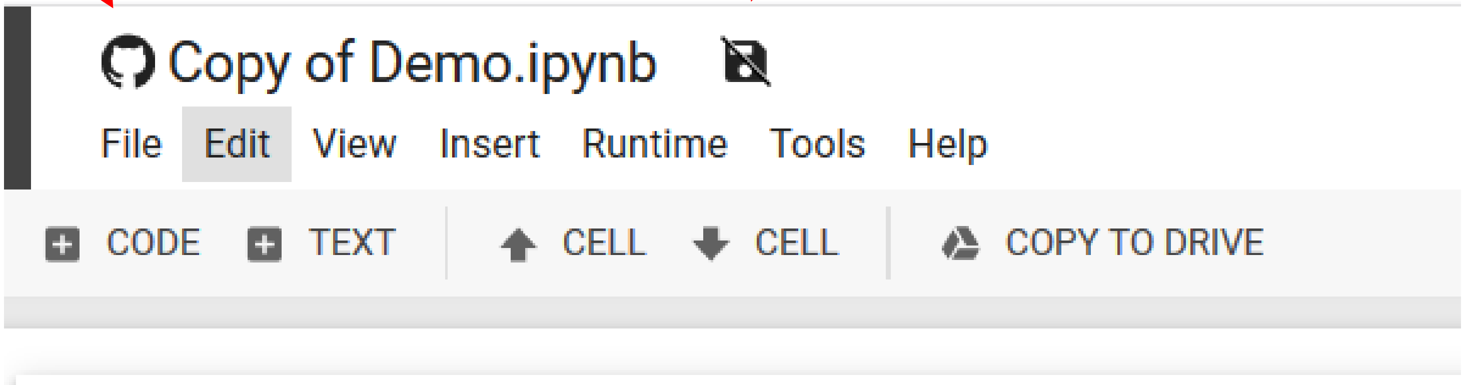
952 lines (952 sloc) 25.3 KB

CO Open in Colab

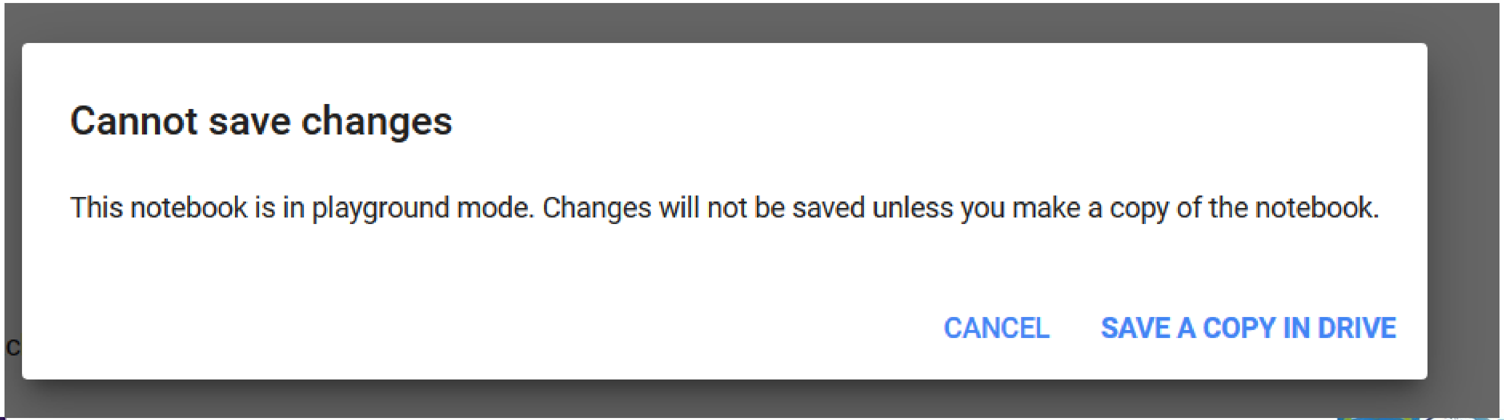
Neural Network with R

Github

Playground mode

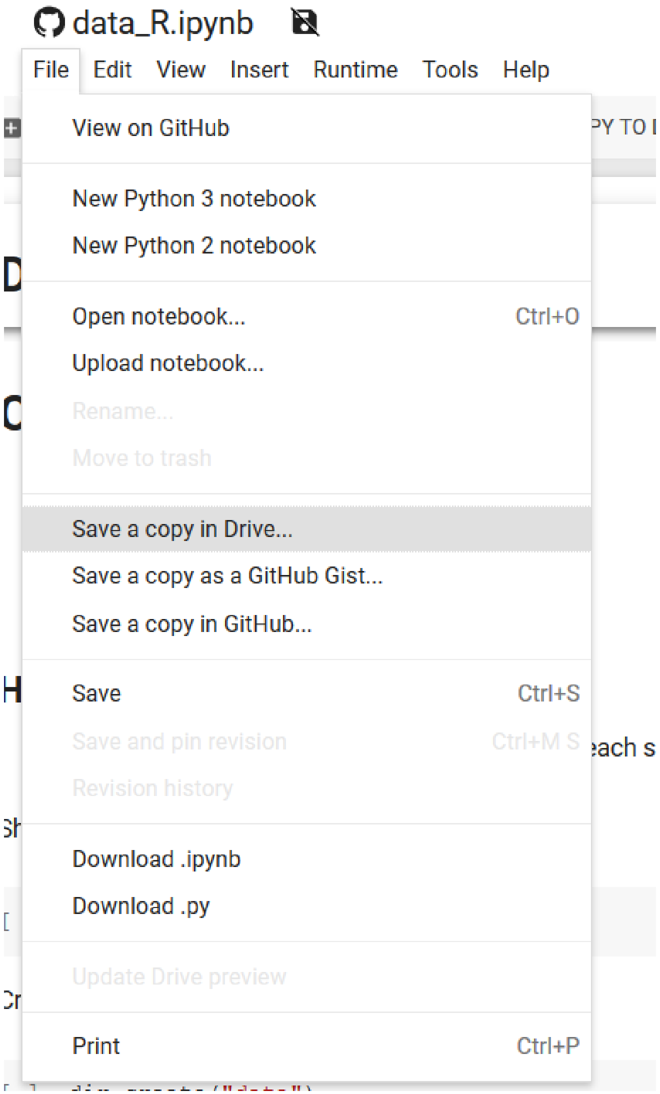


Cannot “File” -> “Save”:

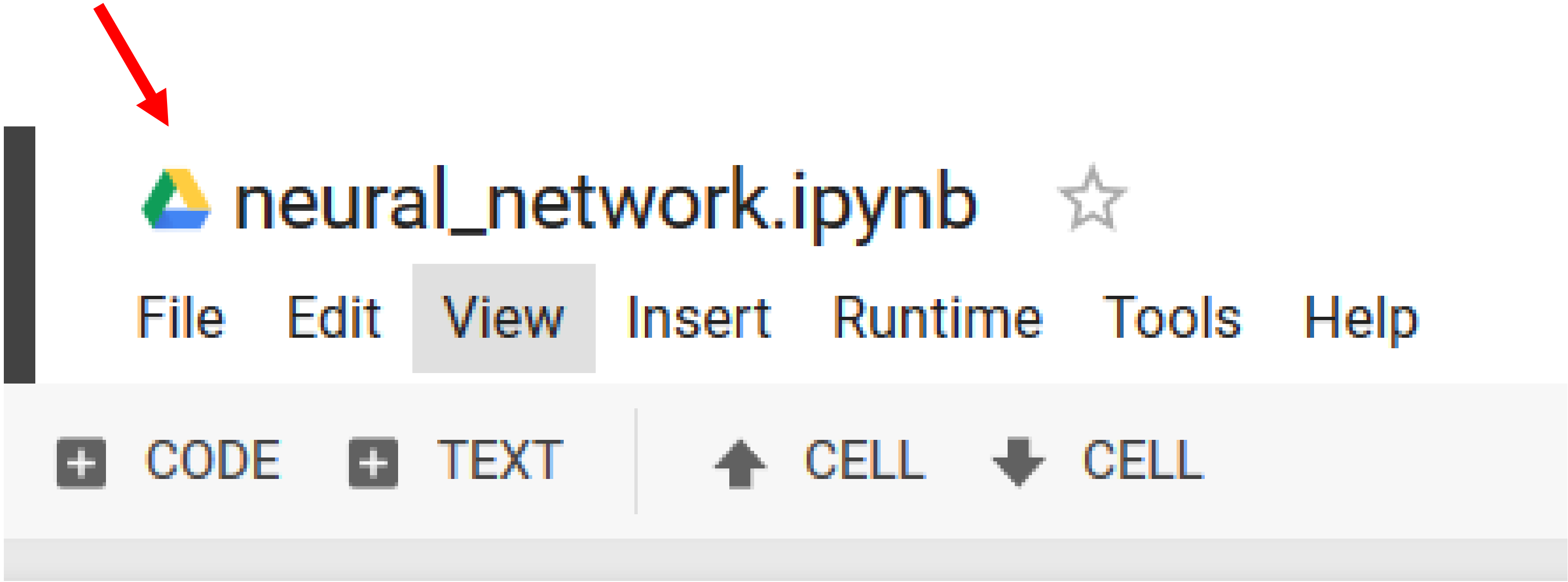

Cannot save changes

This notebook is in playground mode. Changes will not be saved unless you make a copy of the notebook.

Save a copy in Drive…



Google Drive



Outline

- Statistical data mining
  - Introduction to statistical learning
  - Review neural network basics
- Neural network with R
- Deep learning with R

Deep Neural Networks

- Deep Learning is the branch of Machine Learning based on **Deep Neural Networks** (DNNs, i.e., neural networks composed of more than 1 hidden layer).

- **Convolutional Neural Networks** (CNNs) are one of the most popular DNN architectures, most commonly applied to analyzing visual imagery.

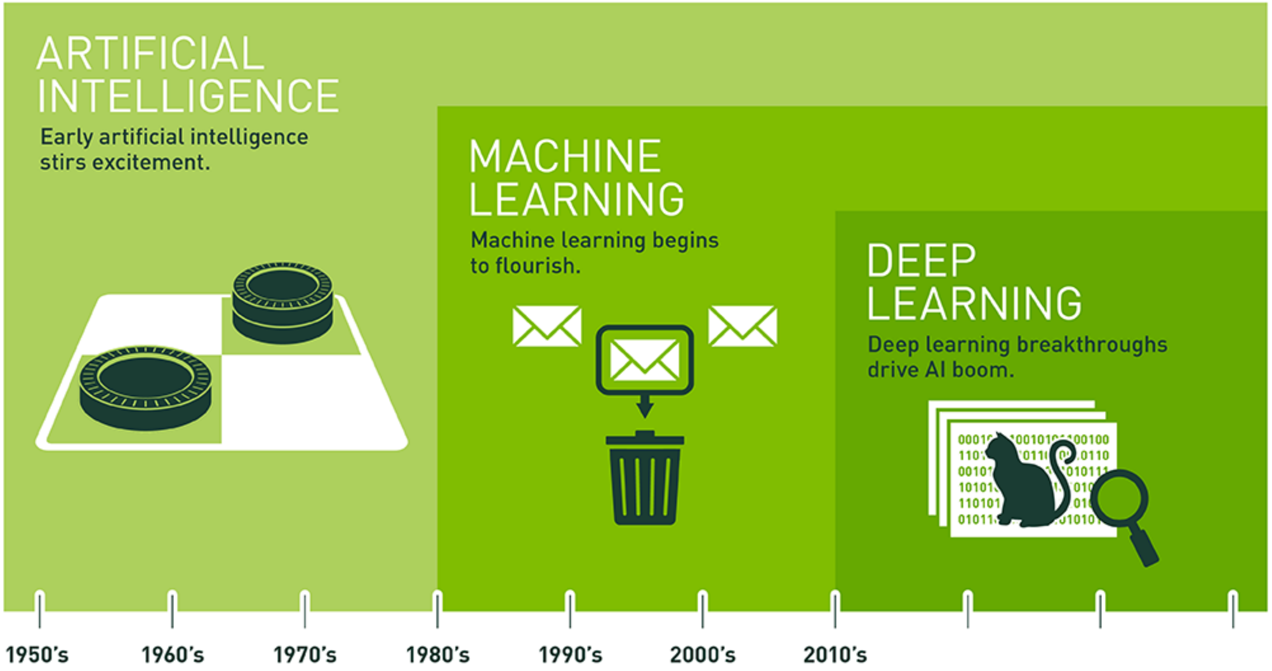
From R deep learning essentials by Mark Hodett and Joshua F. Wiley

Myth or Truth?

- Artificial intelligence means deep learning and replaces all other techniques

Myth or Truth?

- Artificial intelligence means deep learning and replaces all other techniques


ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.
MACHINE LEARNING
Machine learning begins to flourish.
DEEP LEARNING
Deep learning breakthroughs drive AI boom.
1950's
1960's
1970's
1980's
1990's
2000's
2010's

Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Myth or Truth?

- Deep learning requires a PhD-level understanding of mathematics

Myth or Truth?

- Deep learning requires a PhD-level understanding of mathematics
- Deep learning does not require a deep understanding of mathematics unless your interest is in researching new deep learning algorithms and specialized architectures.

From R deep learning essentials by Mark Hodett and Joshua F. Wiley

Myth or Truth?

- Deep learning is hard to train, almost an art form

Myth or Truth?

- Deep learning is hard to train, almost an art form

- Yes deep learning is hard to train and some training method is still a ongoing research.

- But it is not an art form. It does require practice, but the same problems occur over and over again. Even better, there is often a prescribed fix for that problem, for example, if your model is overfitting, add regularization.

From R deep learning essentials by Mark Hodett and Joshua F. Wiley

Myth or Truth?

- Deep learning needs GPUs

Myth or Truth?

- Deep learning needs GPUs

- Deep learning models can run on CPUs, the truth is that any real deep learning work requires excellent GPU(s).
- This does not mean that you need to go out and purchase one, as you can use HPC or cloud-computing to train your models.
- The Google Colab we are using in this workshop is a free Jupyter Notebook environment that runs entirely in the cloud.

From R deep learning essentials by Mark Hodett and Joshua F. Wiley

Myth or Truth?

- R is not a good option for deep learning.

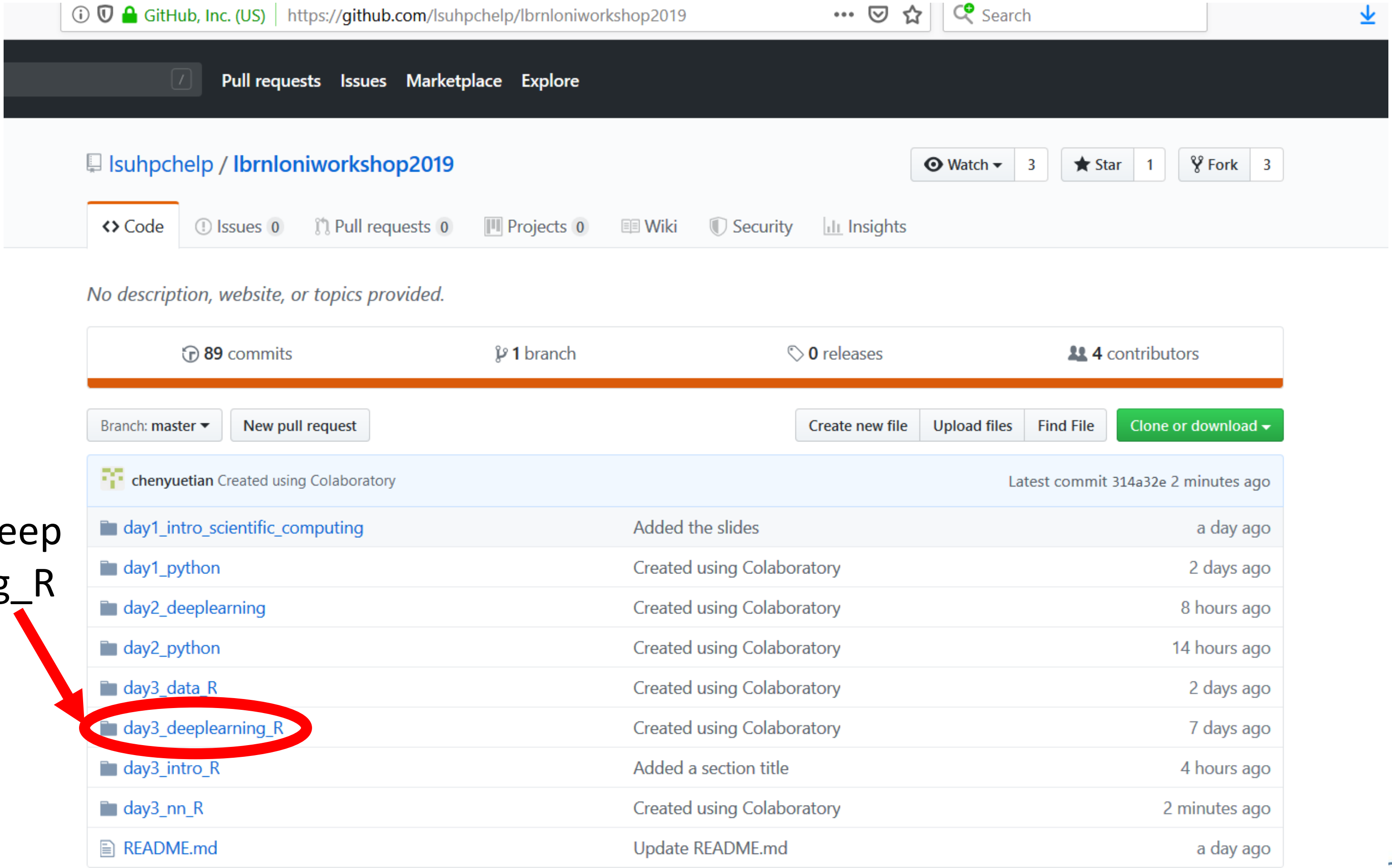
Popular Deep Learning R packages

Package	
keras	high-level interface for neural networks, with a focus on enabling fast experimentation.
tensorflow	a lower-level interface that provides full access to the TensorFlow computational graph.
mxnet	MXNet is supported by public cloud providers including Amazon Web Services (AWS) and Microsoft Azure. Amazon has chosen MXNet as its deep learning framework of choice at AWS.
h2o	provides similar advanced features as the keras package
caret	General training package, even can train certain keras or mxnet model

Traditional
R packages

From R deep learning essentials by Mark Hodett and Joshua F. Wiley

- https://github.com/lsuhpchelp/lbrnloniworkshop2019



GitHub, Inc. (US) https://github.com/lsuhpchelp/lbrnloniworkshop2019

lsuhpchelp / lbrnloniworkshop2019

Watch 3 Star 1 Fork 3

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Security | Insights

No description, website, or topics provided.

89 commits | 1 branch | 0 releases | 4 contributors

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

chenyuetian Created using Colaboratory — Latest commit 314a32e 2 minutes ago

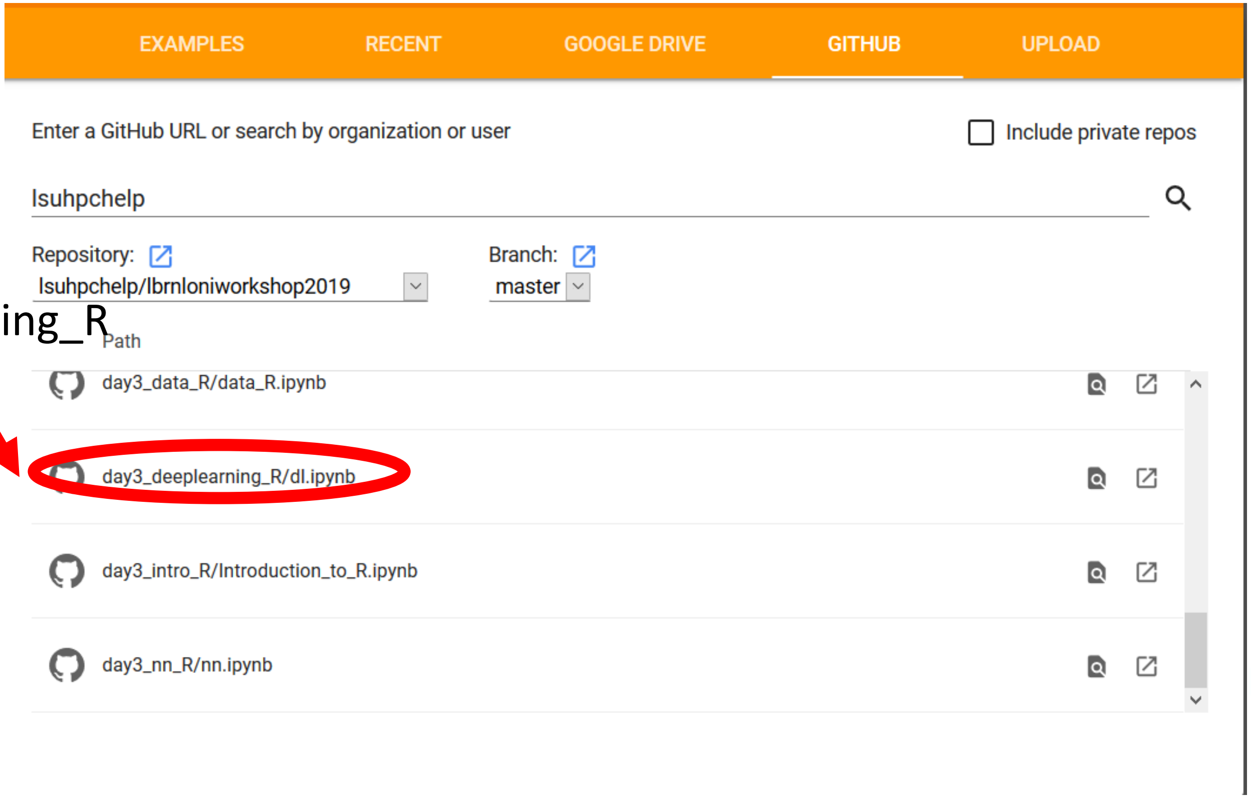
day1_intro_scientific_computing	Added the slides	a day ago
day1_python	Created using Colaboratory	2 days ago
day2_deeplearning	Created using Colaboratory	8 hours ago
day2_python	Created using Colaboratory	14 hours ago
day3_data_R	Created using Colaboratory	2 days ago
day3_deeplearning_R	Created using Colaboratory	7 days ago
day3_intro_R	Added a section title	4 hours ago
day3_nn_R	Created using Colaboratory	2 minutes ago
README.md	Update README.md	a day ago

day3_deep learning_R

If you cannot open data_R.ipynb and get “Sorry, something went wrong. Reload?”

Try this link:
https://colab.research.google.com/github/lsuhpchelp

day3_deeplearning_R /dl.ipynb



Enter a GitHub URL or search by organization or user

□ Include private repos

lsuhpchelp

Repository: lsuhpchelp/lbrnloniworkshop2019

Branch: master

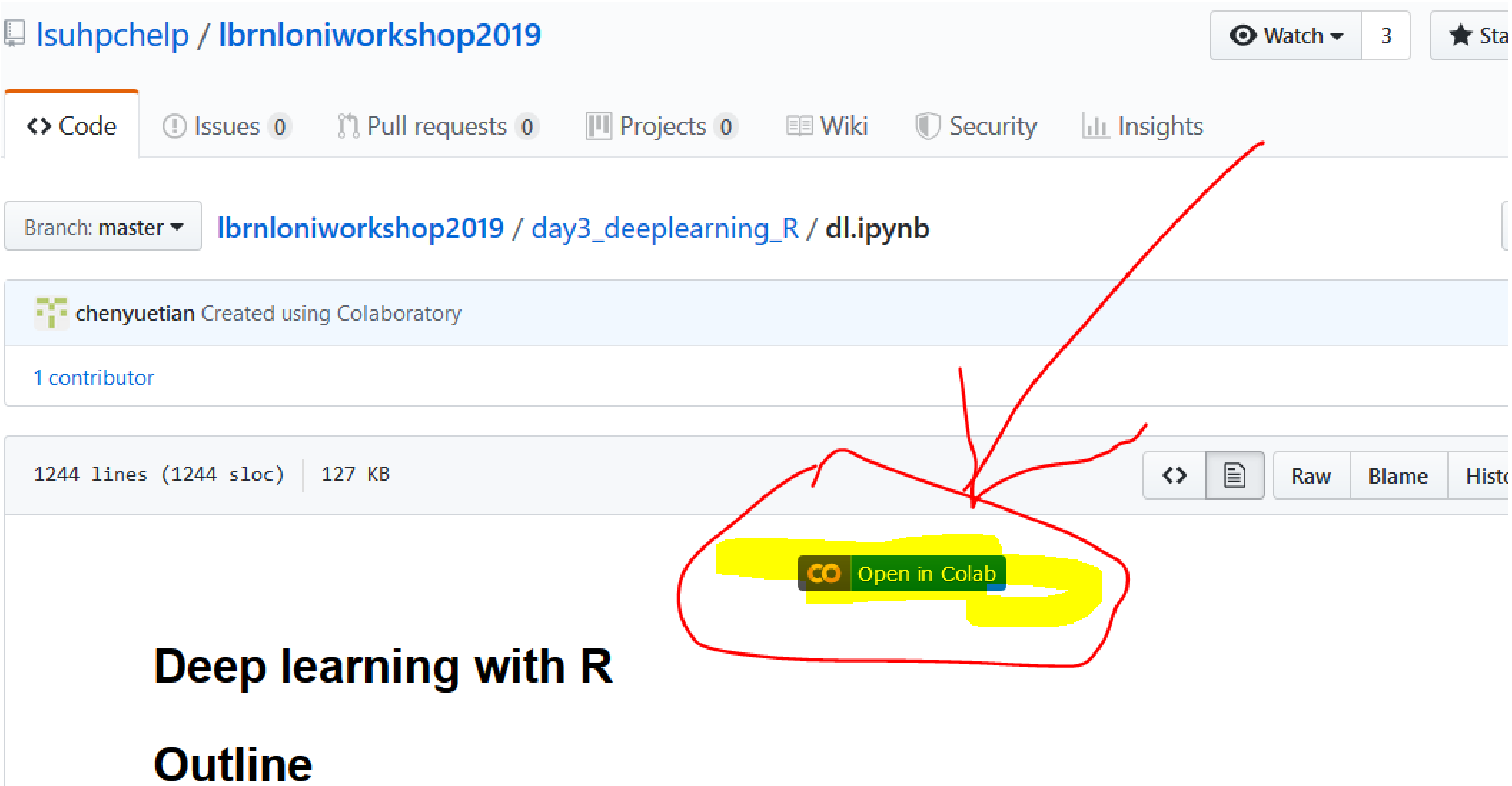
Path

day3_data_R/data_R.ipynb

day3_deeplearning_R/dl.ipynb

day3_intro_R/Introduction_to_R.ipynb

day3_nn_R/nn.ipynb



Deep learning with R

Outline

Getting Help

- User Guides
  - LSU HPC: http://www.hpc.lsu.edu/docs/guides.php#hpc
  - LONI:http://www.hpc.lsu.edu/docs/guides.php#loni
- Documentation: http://www.hpc.lsu.edu/docs
- Contact us
  - Email ticket system: sys-help@loni.org
  - Telephone Help Desk: 225-578-0900