

Install greta



Figure 15.1: The greta logo. See <https://greta-stats.org/> for more information on greta.

As of April 2019, there are version mismatches between **greta** 0.3.0 available via CRAN (i.e. `install.packages("greta")`) and more recent versions of TensorFlow (i.e. `TensorFlow >= 1.12`); the installation of **greta** is not as simple as it once was. The below instructions will help you set up your computer environment to get **greta** running. They do not assume any previous installation of Python and only assume you have a recent version of R/RStudio (R-version 3.4.X or higher) and a machine capable of running TensorFlow.

15.1 Software Stack for Using greta

greta makes Bayesian inference scalable, yet elegantly simple. For scalability, it uses TensorFlow as its numerical computation engine. For simplicity, the **greta** models are written in **R**, sparing us from learning an additional language. While this enables scaleable Bayesian data analysis inside the R-ecosystem, we do need to step outside of R/RStudio for installation of the Google TensorFlow *software stack***.

TensorFlow System Requirements:
 * Ubuntu 16.04 or later (64-bit) *
 macOS 10.12.6 (Sierra) or later (64-bit) (no GPU support) * Windows 7 or later (64-bit) (Python 3 only) *
 Raspbian 9.0 or later



** A software stack is a collection of programs, applications, components and tools that work together to get a result.
 Figure 15.2: The April 2019 software stack possibilities. The stack on the left is the a stable one. The stack on the right is what you would get installing more recent versions of each component; i.e. a falling stack that does not work.

Figure 15.2 shows two potential software stacks. Both include four key components:

1. **Python:** TensorFlow is a Python library and hence, needs a Python implementation installed on its host system in order to run.
2. **TensorFlow:** This is the heart of being able to do fast numerical computing.
3. **TensorFlow Probability:** An additional library built on top of TensorFlow specializing in probabilistic inference at scale.
4. **greta:** A very slick interface between R and TensorFlow. This interface gives us the power of TensorFlow without the complexity overhead of learning another language.

As I write this, only the stack on the left is servicable. The stack on the right represent more recent software versions - which you might get following the easiest default install instructions - but they do not play well with one another. Simply stated, these other versions will not work together. So before leveraging **greta** we will have to do some system configuration and installation to get everything working. I anticipate that this will be made easier in the near future, but for now, this is a robust install process that should work regardless of your system. That being said, you do need a 64-bit computer with about 10GB of free hard drive space.**

**** If you do not meet the minimum system requirements or decide to abandon the installation process detailed below,** you can access greta capabilities using RStudio in the cloud. Simply navigate to <https://rstudio.cloud/project/103813> and use the browser-based version of RStudio with **greta** already installed - i.e. you can start with `library(greta)` and have no need to install anything beyond that.

15.2 The Super Short Installation Instructions

For more detailed instructions with some explanation, skip this section. If you want the super short-version, here it is:

1. Install 64-bit Miniconda from <https://conda.io/miniconda.html> to get Python if you do not have an Anaconda installation already.
2. Open Anaconda prompt (Terminal on Mac) and execute the following commands:
 - i) `conda create -n r-tensorflow python=3.6 tensorflow=1.11 pyyaml requests Pillow pip numpy=1.15`
 - ii) `conda activate r-tensorflow`
 - iii) `pip install tensorflow-probability==0.4.0`
3. Open R/RStudio and execute the following commands:
 - i) `install.packages("greta")`
 - ii) `library(greta)`

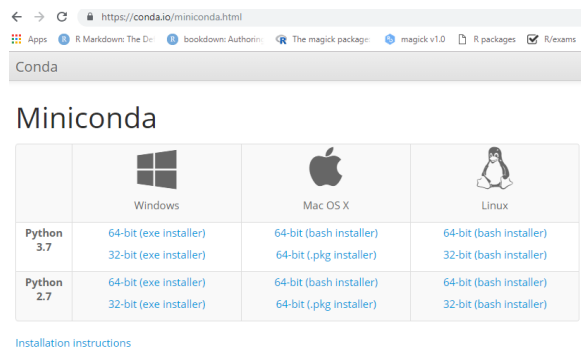
If all of the above works without error, then you are ready for **greta**.

15.3 The More Detailed Installation Instructions

Explanations for all of the steps above and additional details are given in the below instructions.

15.3.1 Installing Miniconda

Assuming you do not have **conda** capabilities already installed on your system, the best way to get an easily manageable Python distribution on your system is to use Miniconda. Figure 15.3 shows the installation page available at <https://conda.io/miniconda.html>. Navigate to that page and download the Python 3.X 64-bit installer for your operating system. While it probably seems like you are downloading Python 3.7 (or a more recent release), we will ultimately downgrade the Python version that TensorFlow will rely on.



As I write this, the most recent version of Python is 3.7, but Tensorflow 1.11 does not play nice with Python 3.7. So even though it might look like you are downloading this incompatible version, switching Python versions is easily done post-installation.

Figure 15.3: The Miniconda download site.

Once downloaded, **Windows** users can double-click the file and follow the prompts - accept all defaults and install Miniconda. After successful install, you should have the **Anaconda Prompt** accessible via the Start Menu (Windows OS):

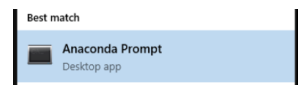


Figure 15.4: The Anaconda Prompt application will open a command prompt that can access the conda package manager for Python (see <https://conda.io/miniconda.html> for more information).

MAC users should complete the Miniconda installation with these steps (alternative installation instruction may be found here <https://conda.io/projects/conda/en/latest/user-guide/install/macos.html>):

1. Open a Terminal window. If you don't know how to do this click * **Applications -> Utilities -> Terminal***
2. Within Terminal "change directories into the folder where your Miniconda download was placed and execute the following:
 - i) `cd ~/Downloads`

The name of the Miniconda may be different, but you just need to run `chmod` in order to make the script executable. Look for the name in your downloads folder. Alternatively, right click on your script and chose Properties -> Permissions -> Allow executing file as program, leaves you with the same result as the `chmod` command in terminal.

- ii) `chmod +x Miniconda2-latest-MacOSX-x86_64.sh`
 - iii) `./Miniconda2-latest-MacOSX-x86_64.sh`
3. Once the installation has begun, accept the license terms and the default installation location. Say yes when prompted whether or not the installer should prepend the Miniconda install location to your PATH.
 4. After installing Miniconda, close your current terminal and open another in order to activate the installation. You should have access to a new command within the terminal, namely the `conda` command. You can test this by verifying the below command can be run without error:

```
conda list
```

Regardless of operating system, once accessed, the Anaconda command screen is awaiting your input:



Figure 15.5: The Anaconda Prompt application will open a command prompt that can access the conda package manager for Python (see <https://conda.io/miniconda.html> for more information).

15.3.2 Create *r-tensorflow* Environment Using Conda

The first thing you will do is create a conda environment - a repository for a software stack where versions of various software packages are carefully maintained and coordinated (Conda does this coordination for you). Execute the following line at the Anaconda (Terminal) command line to create an environment named `r-tensorflow` that includes `Python 3.6`, `tensorflow 1.11`, a local version of `pip`, and some other required Python packages. `**`: :

```
conda create -n r-tensorflow python=3.6 tensorflow=1.11 pyyaml requests Pillow pip numpy=1.15
```

You will most likely be asked whether you wish to proceed, you should enter `y` and then press `ENTER`. This step creates the `r-tensorflow` environment where we will maintain our software stack.

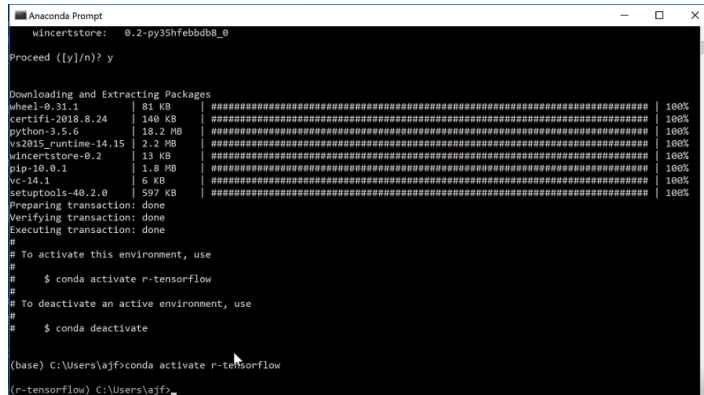
15.3.3 Activating the *r-tensorflow* Environment

To make sure our subsequent package installations are done in the `r-tensorflow` environment (as opposed to some default global environment), we **activate** it:

```
conda activate r-tensorflow
```

After executing the appropriate command from above, the command prompt will now have the environment name (`r-tensorflow`) in parentheses:

`**` `pip` is a supplemental Python package manager that we will use in addition to Conda because the version of `tensorflow-probability` we need is not available via Conda



```

Anaconda Prompt
wincertstore: 0.2-py35hfebbd8_0

Proceed ([y]/n)? y

Downloading and Extracting Packages
wheel-0.31.1 | 81 KB | ##### | 100%
certifi-2018.8.24 | 140 KB | ##### | 100%
python-3.5.6 | 18.2 MB | ##### | 100%
vc2015_runtime-14.15 | 2.2 KB | ##### | 100%
wincertstore-0.2 | 13 KB | ##### | 100%
pip-10.0.1 | 1.8 MB | ##### | 100%
vc-14.1 | 6 KB | ##### | 100%
setuptools-40.2.0 | 597 KB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

#
# To activate this environment, use
#
# $ conda activate r-tensorflow
#
# To deactivate an active environment, use
#
# $ conda deactivate

(base) C:\Users\ajf>conda activate r-tensorflow
(r-tensorflow) C:\Users\ajf>

```

Figure 15.6: Anaconda command prompt with the r-tensorflow environment activated.

15.3.4 Install tensorflow-probability

tensorflow-probability is built on top of TensorFlow. To get the right version of tensorflow-probability, we use the pip package manager because it is not available through Conda at this time. Please note that Conda is the better package manager for installing tensorflow - it installs a faster implementation (see <https://towardsdatascience.com/stop-installing-tensorflow-using-pip-for-performance-sake-5854f9d9eb0c>). Hence, using pip we install tensorflow-probability only using the following line at the command prompt:

```
pip install tensorflow-probability==0.4
```

Again, you will most likely be asked whether you wish to proceed, you should enter y and then press ENTER (note: do this whenever prompted - I will not be explicit about this below). Verify that tensorflow-probability 0.4.0 is installed in the r-tensorflow environment by typing conda list and then pressing ENTER.

```

tensorflow      1.11.0
tensorflow-base  1.11.0
tensorflow-probability 0.4.0

```

Figure 15.7: Anaconda command prompt with the r-tensorflow environment activated.

You can close the Anaconda command prompt - we will not use it again as our TensorFlow environment is setup and complete. Next, we open RStudio to install greta.

15.3.5 Install greta

From within RStudio, execute the following lines:

```

install.packages("remotes")
remotes::install_github("rich-iannone/Diagrammer") # need dev version for plate notation
remotes::install_github("flyaflya/causact") # update causact package used in book
install.packages("greta")
library(greta)

```

The first three lines install some prerequisites for the **greta** package. The fourth line installs the **greta** package and does not need to run again. The last line makes package functionality available within your **R** session and as long as you do **NOT** see any error messages related to **tensorflow** or **tensorflow-probability** you are ready for Bayesian inference made possible by **greta**. Messages regarding **masked objects** can be safely ignored.

15.4 Testing the Install

Just to make sure everything works, run the following within **R**:

```
library(greta)
library(causact)
## simulate Bernoulli data with 72% prob of success
y <- rbern(n = 100, prob = 0.72) #data

## create prior for theta - prob of succes
theta = uniform(0,1) #prior

## specify likelihood of data
distribution(y) <- bernoulli(prob = theta) #likelihood

## create model - list parameters of interest
m <- model(theta)

## get representative sample of posterior distribution
draws <- mcmc(m)
## below line requires bayesplot package if you
## get an error running the below then install.packages("bayesplot")
bayesplot::mcmc_trace(draws)
## if you see plots that look like fuzzy caterpillars
## then your install was successful
```

If it runs, congratulations - you can now harness the power of **greta**!!

15.4.1 Extra Hints

Sometimes **RStudio** can't seem to find your **r-tensorflow** environment. You will get messages insicating **tensorflow** is not installed. Try restarting your **R** session and running these lines (be sure your path in the second line points to the **r-tensorflow** folder on your computer):

```
reticulate::use_condaenv("r-tensorflow")  
reticulate::use_python("c:/Users/<USERNAME>/Miniconda3/envs/r-tensorflow/", required = TRUE)
```

where you replace the path to the environment with your own.