Advanced Data Analysis

DATA 71200

Class 1

Course Description

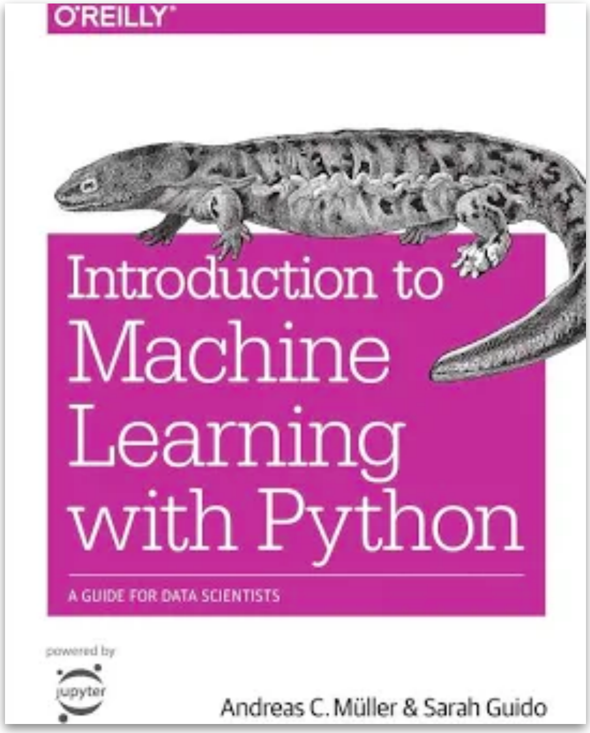
- This course will provide you with skills necessary to a**pply machine learning techniques to data**, and i**nterpret and communicate their results**.
- You will also begin to develop **intuitions** about when machine learning is an appropriate tool versus other statistical methods.
- This course will cover both **supervised methods** (e.g., k-nearest neighbors, naïve Bayes classifiers, decision trees, and support vector machines) and **unsupervised methods** (e.g., principal component analysis and k-means clustering).
  - The supervised methods will focus primarily on **“classic” machine learning techniques** where features are designed rather than learned, although we will briefly look at recent deep learning models with neural networks.
- This is an **applied machine learning class** that emphasizes the intuitions and know-how needed to get learning algorithms to work in practice, rather than mathematical derivations.
- The course will be taught in **Python**, primarily using the **scikit-learn** library.

Course Objectives

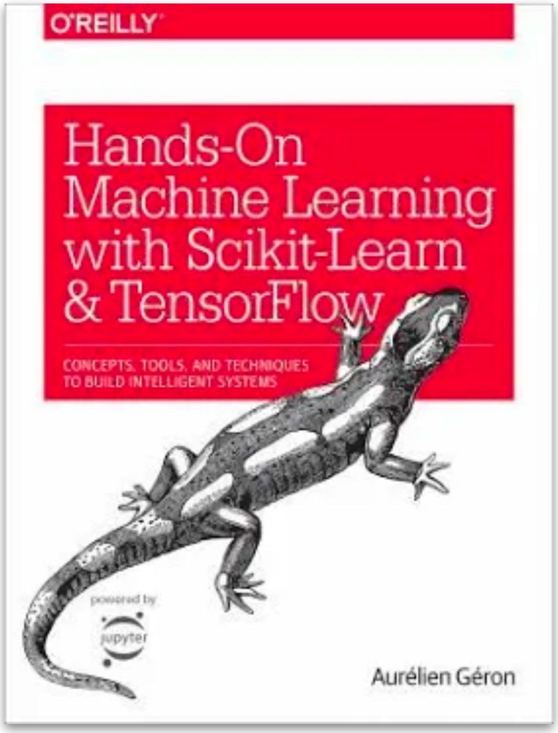
- By the end of the course, you will be able to
  - articulate the main assumptions underlying machine learning approaches
  - demonstrate the basic principles of dataset creation
  - articulate the importance of data representations
  - evaluate machine learning algorithms
  - articulate the difference between supervised and unsupervised learning
  - apply a range of supervised and unsupervised learning techniques

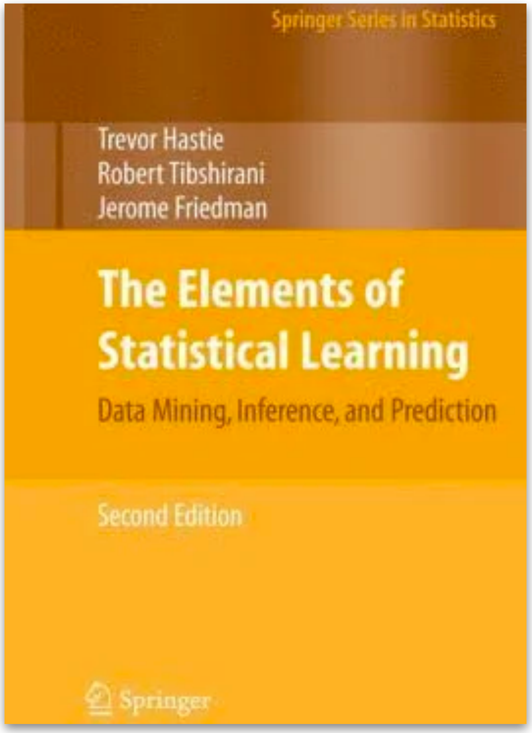
Textbooks

Required


O'REILLY
Introduction to Machine Learning with Python
A GUIDE FOR DATA SCIENTISTS
powered by jupyter
Andreas C. Müller & Sarah Guido

Recommended


O'REILLY
Hands-On Machine Learning with Scikit-Learn & TensorFlow
CONCEPTS, TOOLS, AND TECHNIQUES TO BUILD INTELLIGENT SYSTEMS
powered by jupyter
Aurélien Géron



Springer Series in Statistics

Trevor Hastie
Robert Tibshirani
Jerome Friedman

The Elements of Statistical Learning

Data Mining, Inference, and Prediction

Second Edition

Springer

Grade Breakdown

Class Participation	10%
Datacamp Assignments	25%
Project 1: Dataset creation	15%
Project 2: Supervised learning	15%
Project 3: Unsupervised learning	15%
Final Paper	20%

Grade Breakdown Details

- **Class Participation: 10%**
  - The participation grade is a combination of attendance (including arriving on time); attentiveness, engagement, and participation during class; and general preparedness for class discussions.
- **Datacamp Assignments: 25%**
  - These projects are hands-on activities designed to both provide coding background and reinforce the concepts covered in class.

Grade Breakdown Details

- **Project 1 (Dataset creation): 15%**
    - Curation and cleaning of a labeled data set that you will use for the supervised and unsupervised learning tasks in project 2 and 3. The dataset can built from existing data and should be stored in your GitHub repostiory.
- **Project 2 (Supervised learning): 15%**
    - Application of two supervised learning techniques on the dataset you created in Project 1. This assignment should be completed as a Jupyter notebook your GitHub repository.

Grade Breakdown Details

- **Project 3 (Unsupervised learning): 15%**
  - Application of two unsupervised learning techniques on the dataset you created in Project 1. This assignment should be completed as a Jupyter notebook your GitHub repository.
- **Final Paper: 20%**
  - A 5--8 page paper describing the work you did in projects 1--3 (your dataset and your supervised and unsupervised experiments). The paper should describe both what you did technically and what you learned from the relative performance of the machine learning approaches you applied to your dataset. This assignment should be posted as a PDF in your GitHub repository.

Course Schedule

29-Jan	Introduction
5-Feb	What is Machine Learning?
12-Feb	No Class
19-Feb	Getting Started with Machine Learning
26-Feb	Inspecting Data
4-Mar	Representing Data

Course Schedule

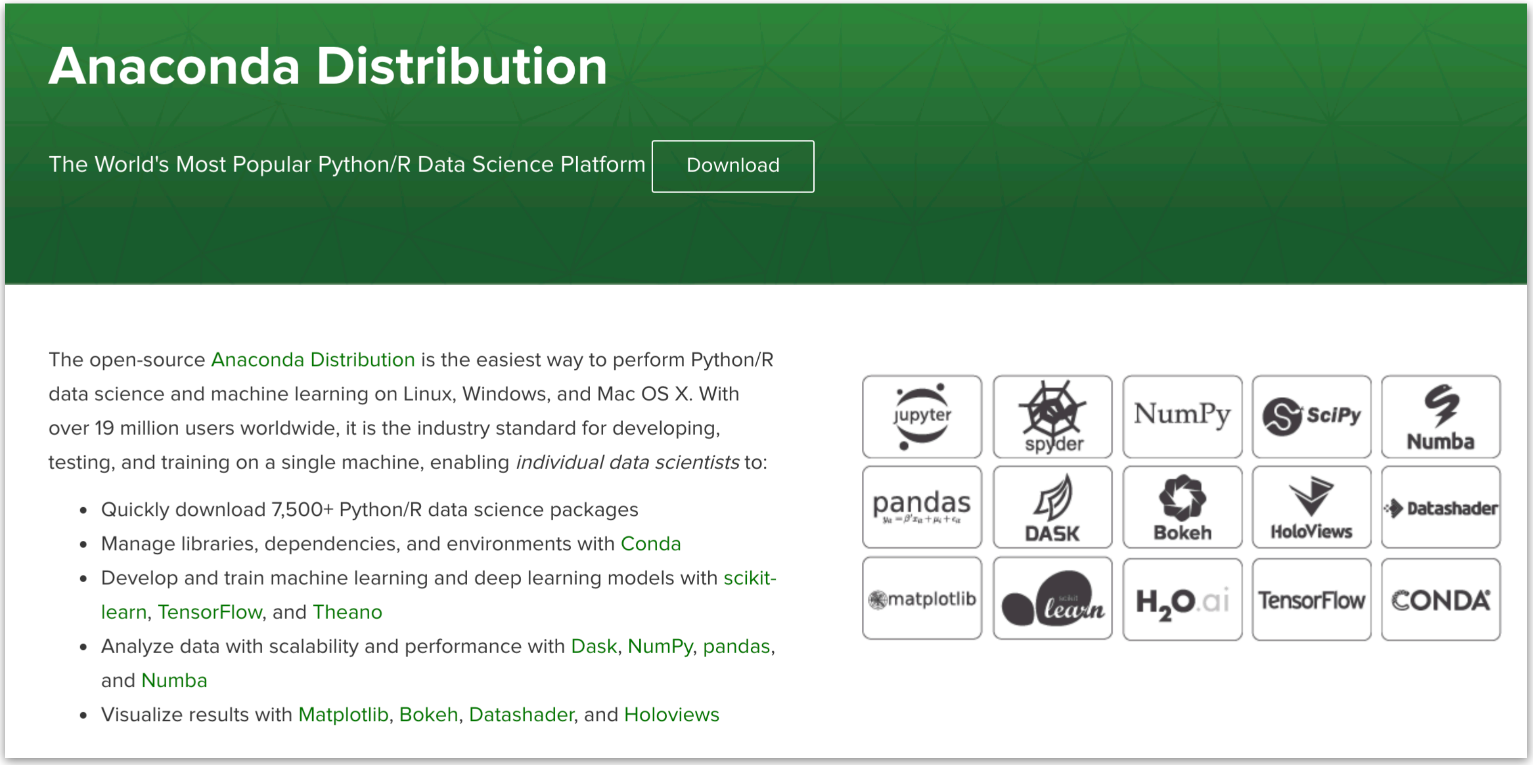
11-Mar	Evaluation Methods
18-Mar	Supervised Learning (k-Nearest Neighbors, Linear Models) – *Project 1 Due*
25-Mar	Supervised Learning (Naive Bayes Classifiers and Decision Trees)
1-Apr	Supervised Learning (Support Vector Machines and Uncertainty estimates from Classifiers)
7-Apr	Unsupervised Learning (Dimensionality Reduction & Feature Extraction, and Manifold Learning) - *Project 2 Due*

Course Schedule

8-Apr	No class
15-Apr	No class
22-Apr	Unsupervised Learning (Clustering)
29-Apr	Deep Learning
6-May	Ethics - *Project 3 Due*
13-May	Ethics
20-May	*Final Project Due*

Coding Environment

- **Python 3**
  - matplotlib, NumPy, Pandas, SciPy scikit learn
- **Jupytr notebooks**



Anaconda Distribution

The World's Most Popular Python/R Data Science Platform

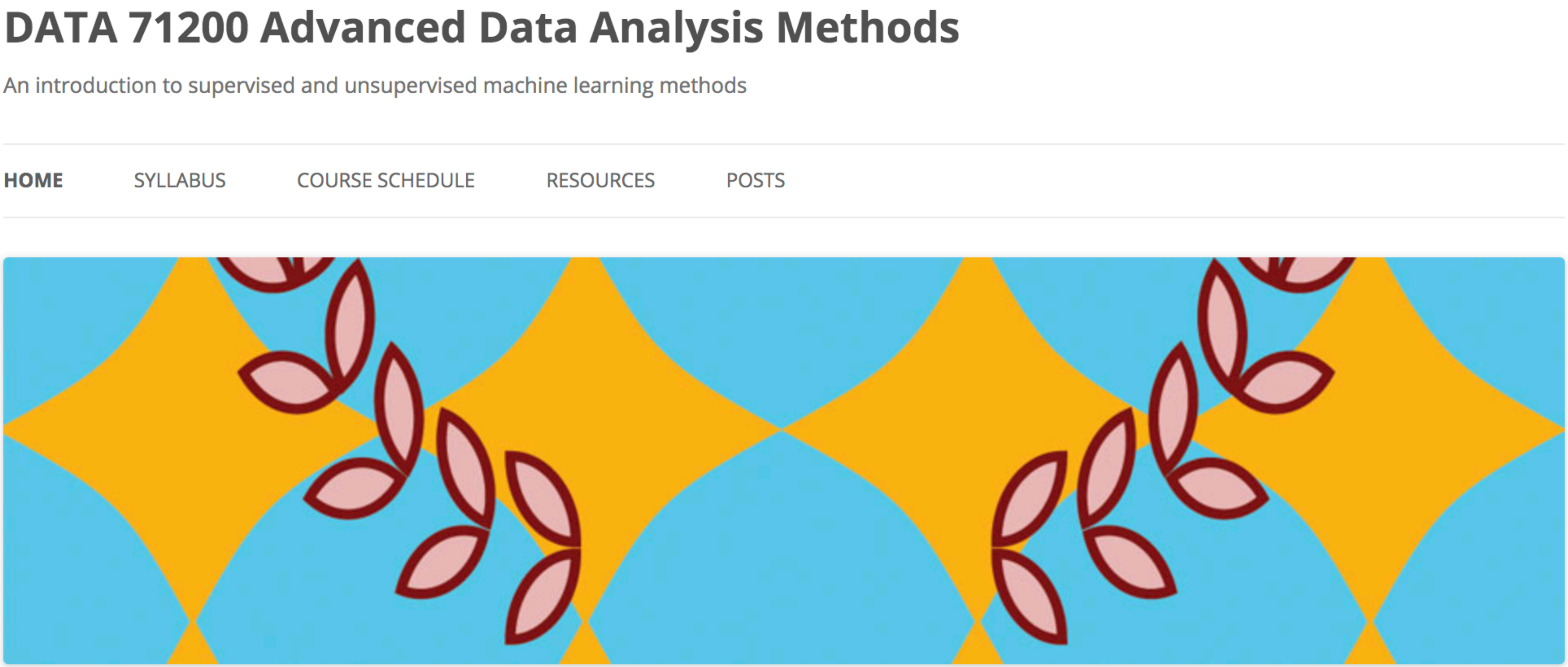
Download

The open-source Anaconda Distribution is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 19 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

- Quickly download 7,500+ Python/R data science packages
- Manage libraries, dependencies, and environments with Conda
- Develop and train machine learning and deep learning models with scikit-learn, TensorFlow, and Theano
- Analyze data with scalability and performance with Dask, NumPy, pandas, and Numba
- Visualize results with Matplotlib, Bokeh, Datashader, and Holoviews

Tutorial: https://machinelearningmastery.com/setup-python-environment-machine-learning-deep-learning-anaconda/

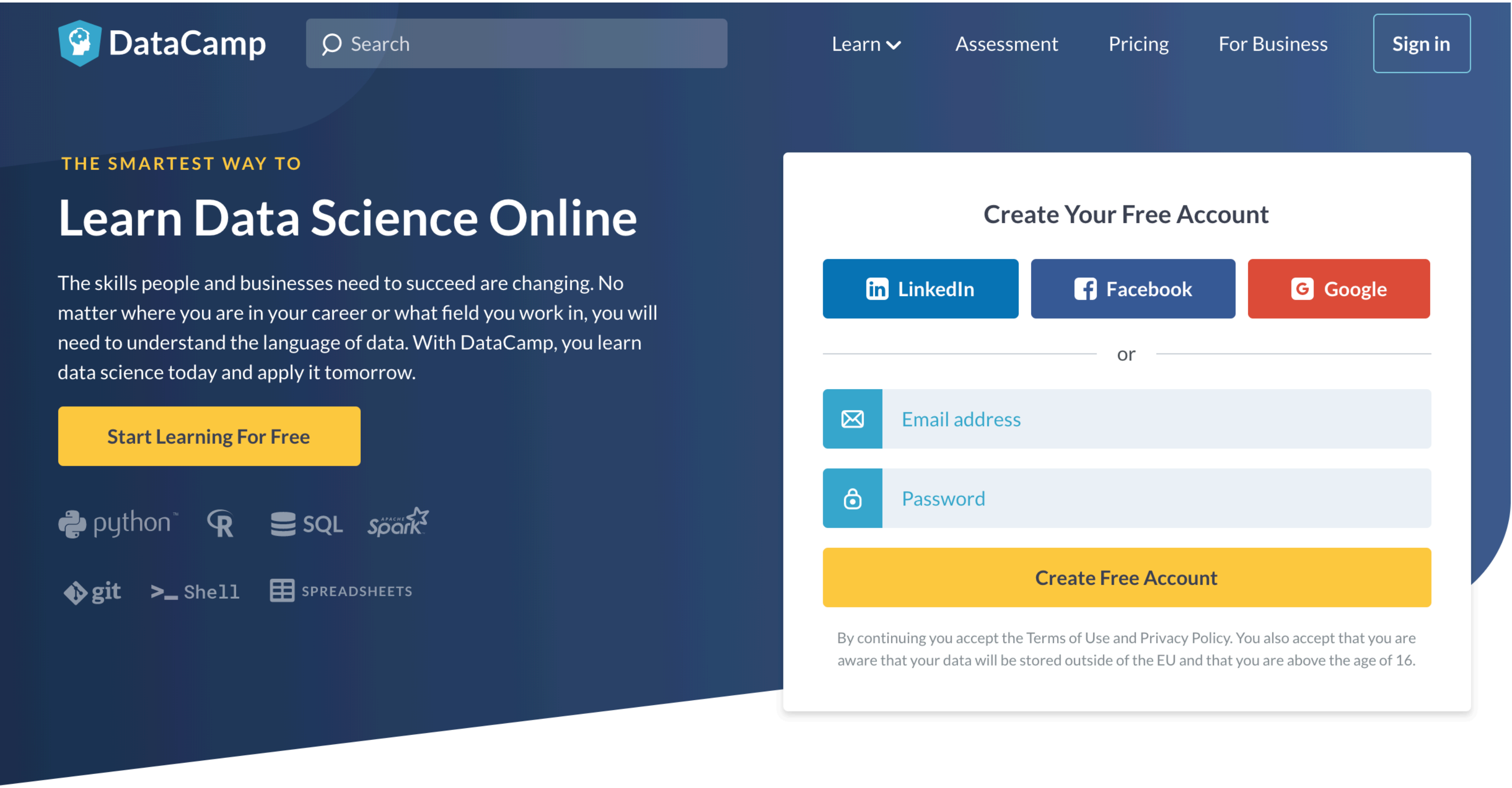
Class Website



DATA 71200 Advanced Data Analysis Methods

An introduction to supervised and unsupervised machine learning methods

Data Camp



THE SMARTEST WAY TO

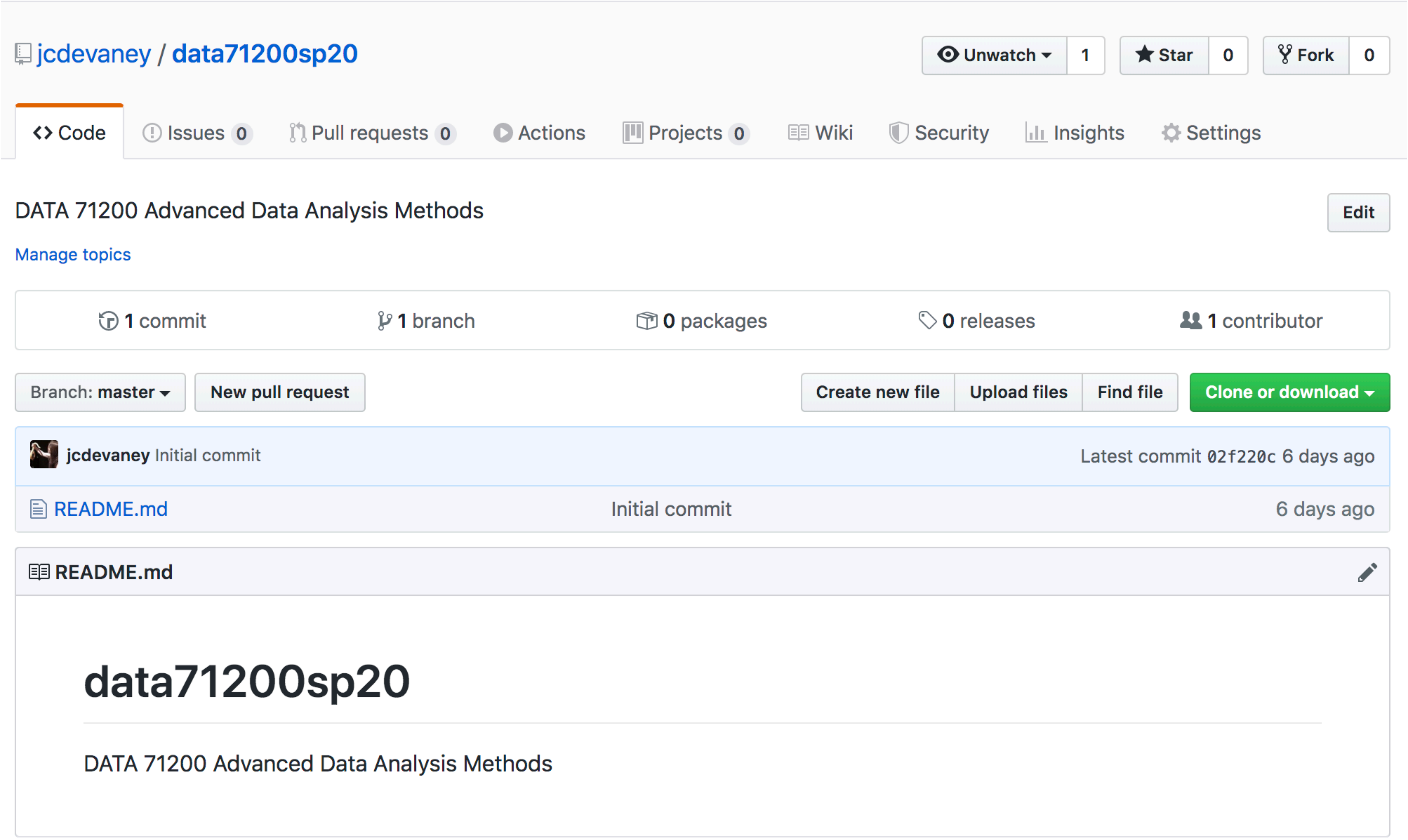
Learn Data Science Online

The skills people and businesses need to succeed are changing. No matter where you are in your career or what field you work in, you will need to understand the language of data. With DataCamp, you learn data science today and apply it tomorrow.

Create Your Free Account

By continuing you accept the Terms of Use and Privacy Policy. You also accept that you are aware that your data will be stored outside of the EU and that you are above the age of 16.

GitHub



DATA 71200 Advanced Data Analysis Methods

jcdevaney Initial commit — Latest commit 02f220c 6 days ago

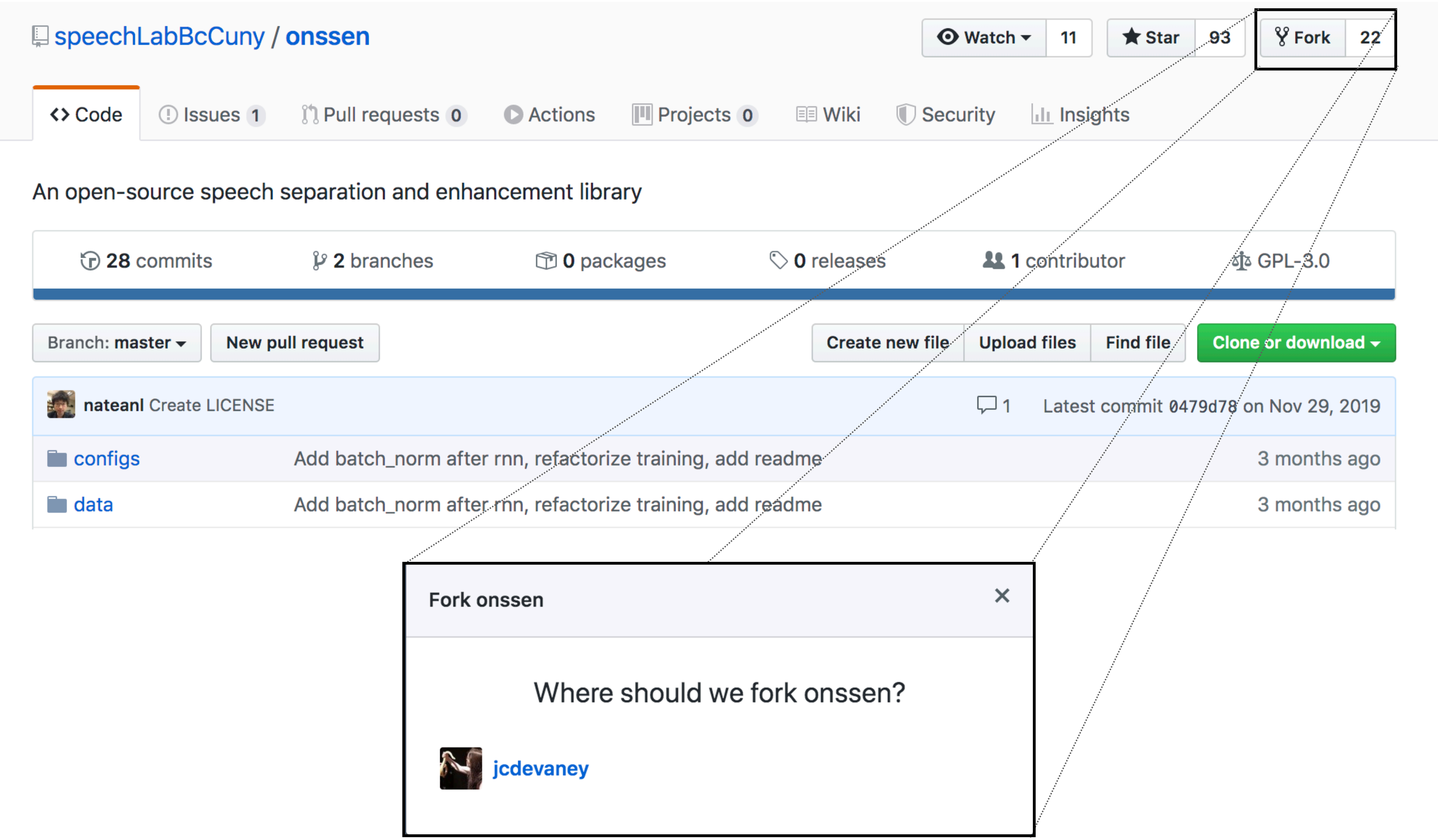
README.md	Initial commit	6 days ago

README.md

data71200sp20

DATA 71200 Advanced Data Analysis Methods

GitHub - Forking versus Cloning



speechLabBcCuny / onssen

An open-source speech separation and enhancement library

nateanl Create LICENSE 1 Latest commit 0479d78 on Nov 29, 2019

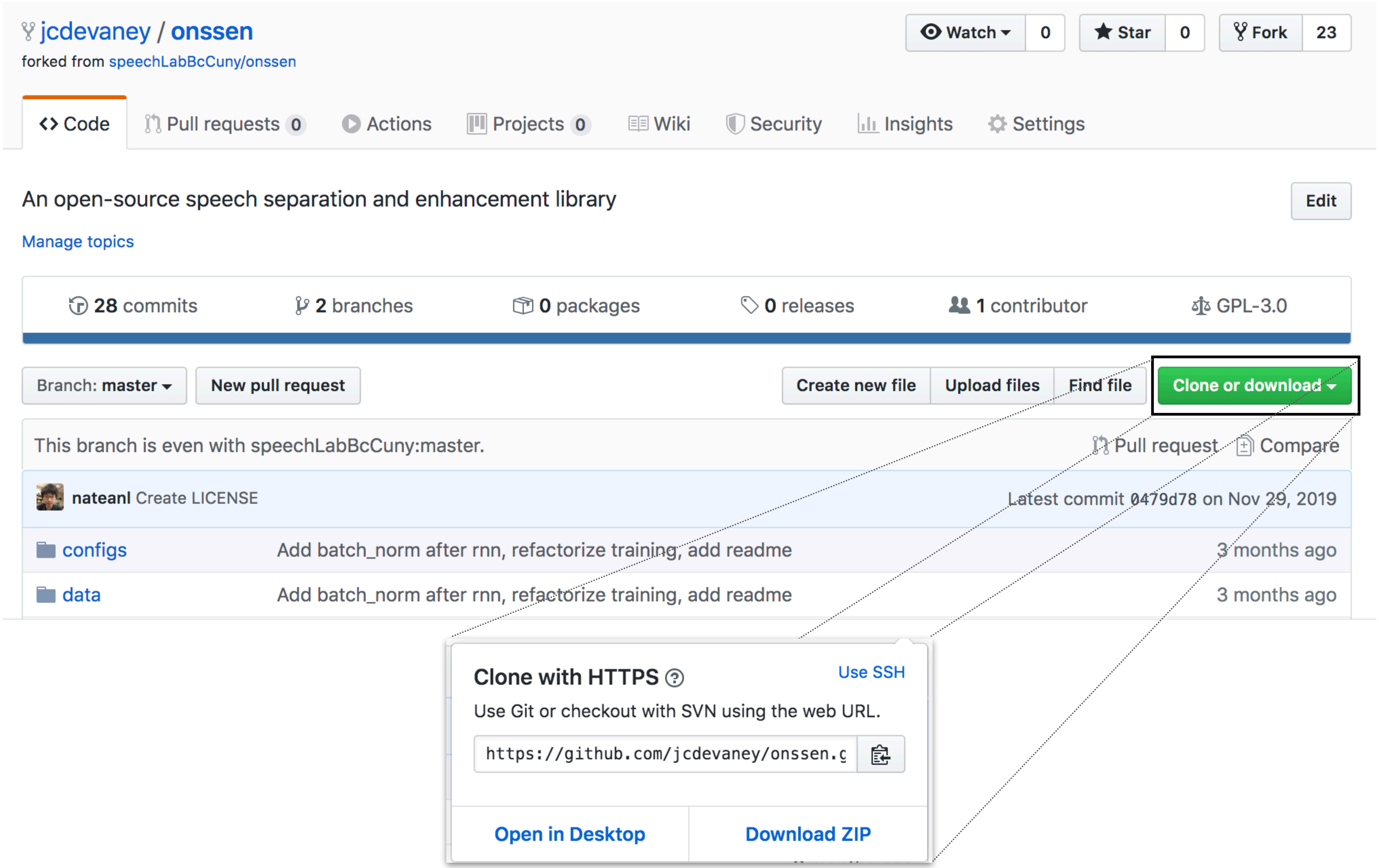
configs	Add batch_norm after rnn, refactorize training, add readme	3 months ago
data	Add batch_norm after rnn, refactorize training, add readme	3 months ago

Fork onssen

Where should we fork onssen?

jcdevaney

GitHub - Forking versus Cloning



forked from speechLabBcCuny/onssen

An open-source speech separation and enhancement library

This branch is even with speechLabBcCuny:master.

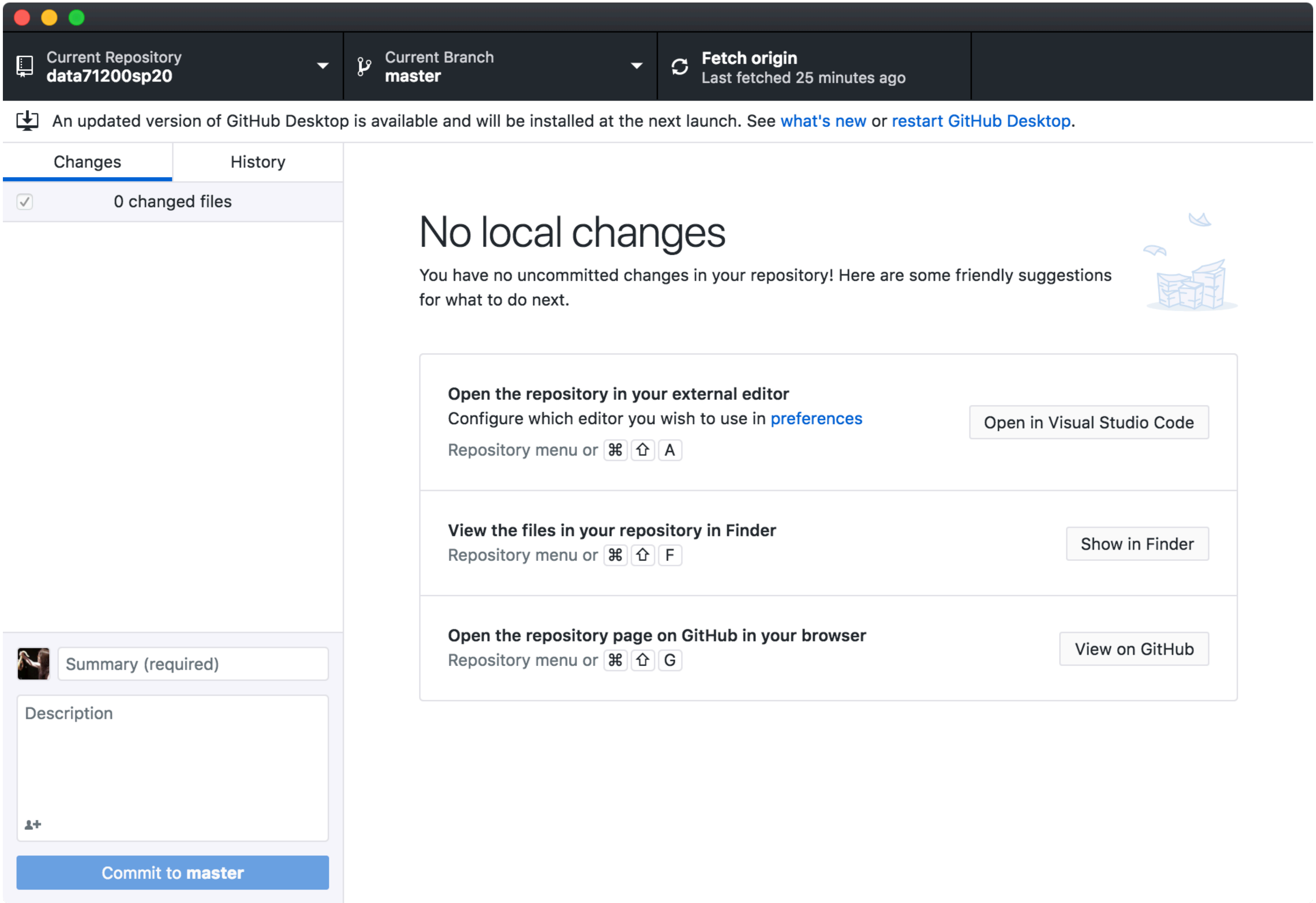
nateanl Create LICENSE		Latest commit 0479d78 on Nov 29, 2019
configs	Add batch_norm after rnn, refactorize training, add readme	3 months ago
data	Add batch_norm after rnn, refactorize training, add readme	3 months ago

Clone with HTTPS

Use Git or checkout with SVN using the web URL.

```
https://github.com/jcdevaney/onssen.g
```

GitHub Desktop



An updated version of GitHub Desktop is available and will be installed at the next launch. See what's new or restart GitHub Desktop.

No local changes

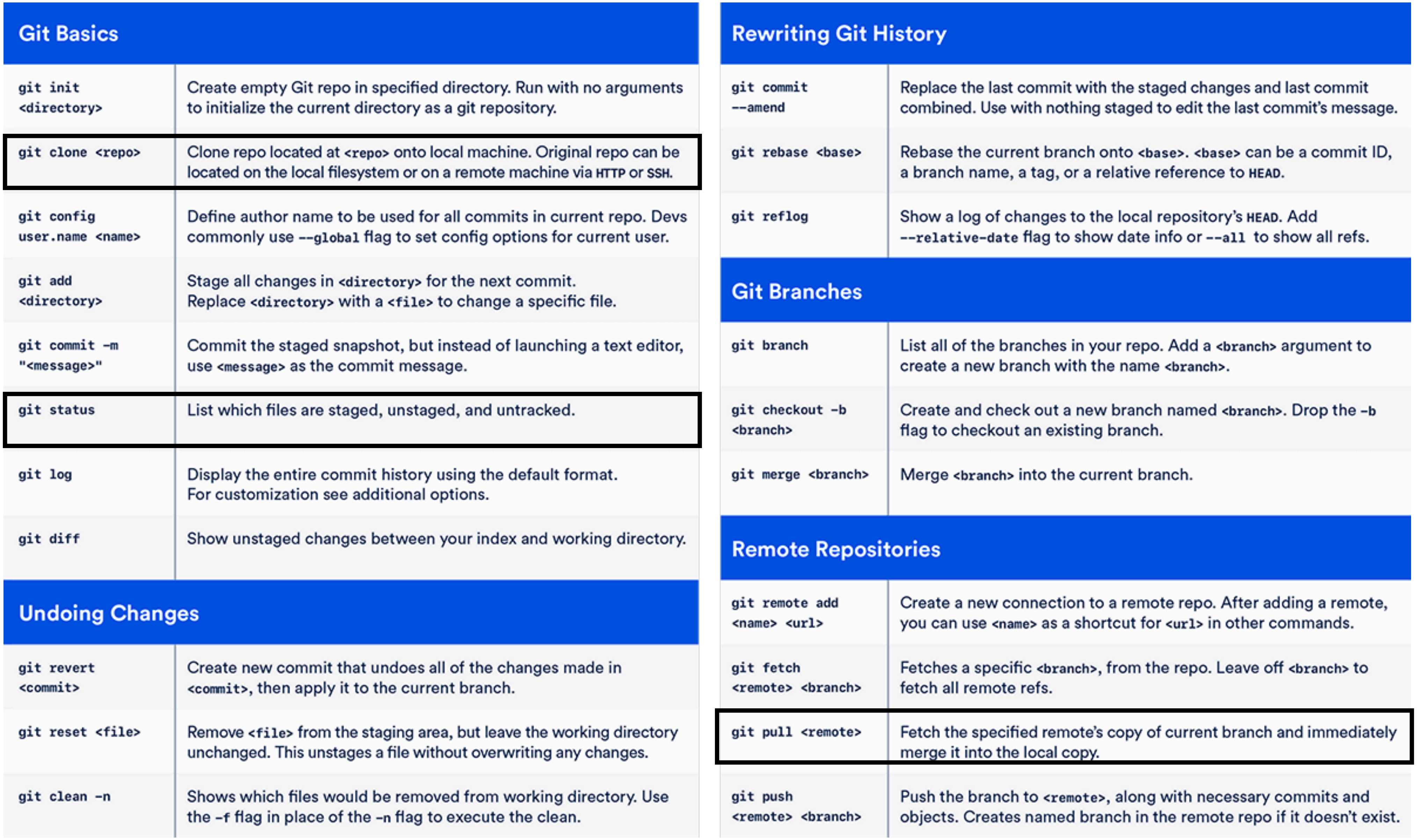
You have no uncommitted changes in your repository! Here are some friendly suggestions for what to do next.

Open the repository in your external editor
Configure which editor you wish to use in preferences
Repository menu or ⌘ ⇧ A

View the files in your repository in Finder
Repository menu or ⌘ ⇧ F

Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G

Git Command Line



Git Basics

`git init <directory>`	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.
`git clone <repo>`	Clone repo located at `<repo>` onto local machine. Original repo can be located on the local filesystem or on a remote machine via `HTTP` or `SSH`.
`git config user.name <name>`	Define author name to be used for all commits in current repo. Devs commonly use `--global` flag to set config options for current user.
`git add <directory>`	Stage all changes in `<directory>` for the next commit. Replace `<directory>` with a `<file>` to change a specific file.
`git commit -m "<message>"`	Commit the staged snapshot, but instead of launching a text editor, use `<message>` as the commit message.
`git status`	List which files are staged, unstaged, and untracked.
`git log`	Display the entire commit history using the default format. For customization see additional options.
`git diff`	Show unstaged changes between your index and working directory.

Undoing Changes

`git revert <commit>`	Create new commit that undoes all of the changes made in `<commit>`, then apply it to the current branch.
`git reset <file>`	Remove `<file>` from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes.
`git clean -n`	Shows which files would be removed from working directory. Use the `-f` flag in place of the `-n` flag to execute the clean.

Rewriting Git History

`git commit --amend`	Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.
`git rebase <base>`	Rebase the current branch onto `<base>`. `<base>` can be a commit ID, a branch name, a tag, or a relative reference to `HEAD`.
`git reflog`	Show a log of changes to the local repository's `HEAD`. Add `--relative-date` flag to show date info or `--all` to show all refs.

Git Branches

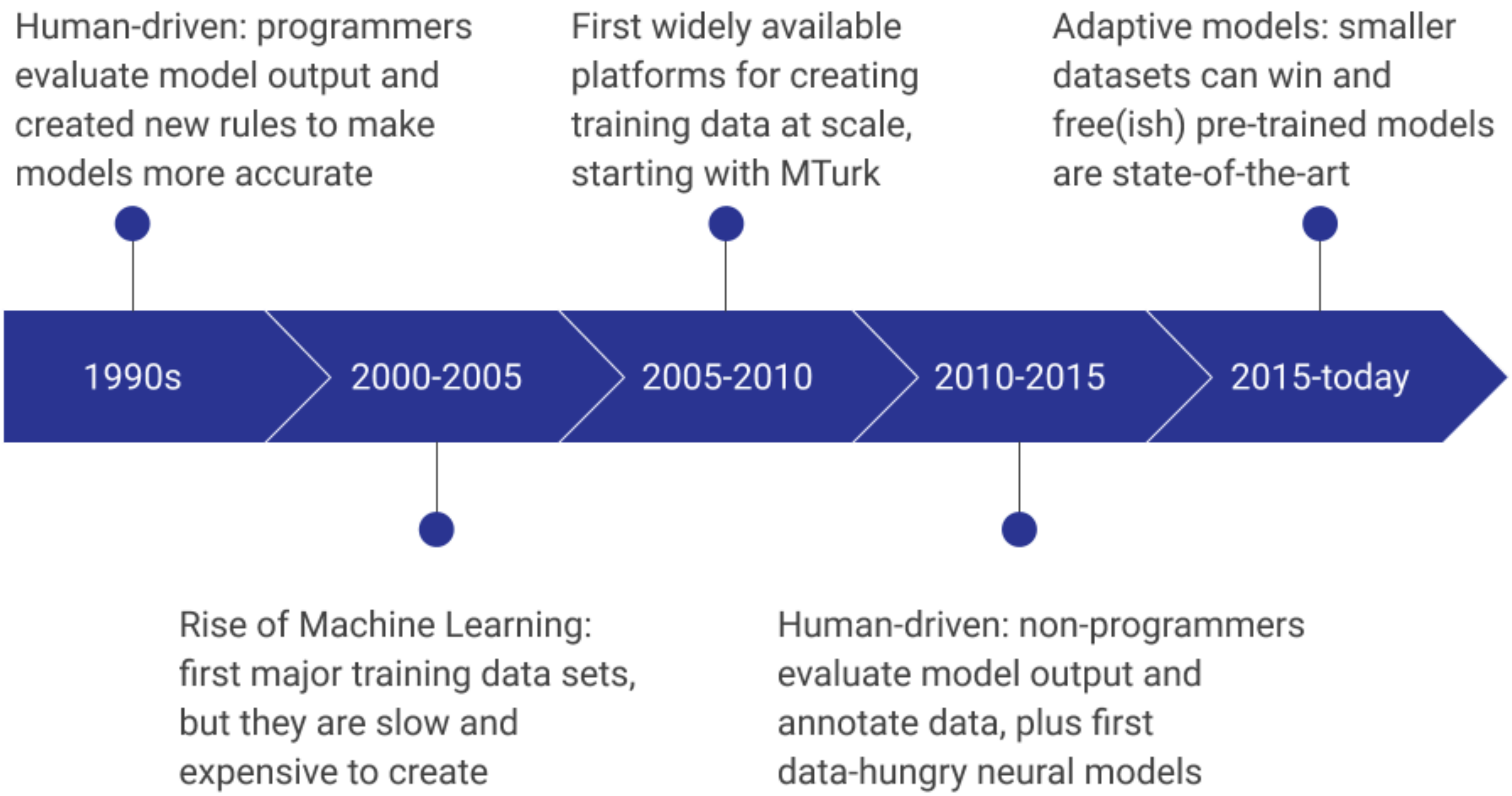
`git branch`	List all of the branches in your repo. Add a `<branch>` argument to create a new branch with the name `<branch>`.
`git checkout -b <branch>`	Create and check out a new branch named `<branch>`. Drop the `-b` flag to checkout an existing branch.
`git merge <branch>`	Merge `<branch>` into the current branch.

Remote Repositories

`git remote add <name> <url>`	Create a new connection to a remote repo. After adding a remote, you can use `<name>` as a shortcut for `<url>` in other commands.
`git fetch <remote> <branch>`	Fetches a specific `<branch>`, from the repo. Leave off `<branch>` to fetch all remote refs.
`git pull <remote>`	Fetch the specified remote's copy of current branch and immediately merge it into the local copy.
`git push <remote> <branch>`	Push the branch to `<remote>`, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet

Machine Learning



1990s

Human-driven: programmers evaluate model output and created new rules to make models more accurate

2000-2005

Rise of Machine Learning: first major training data sets, but they are slow and expensive to create

2005-2010

First widely available platforms for creating training data at scale, starting with MTurk

2010-2015

Human-driven: non-programmers evaluate model output and annotate data, plus first data-hungry neural models

2015-today

Adaptive models: smaller datasets can win and free(ish) pre-trained models are state-of-the-art

https://towardsdatascience.com/a-brief-history-of-training-data-9c513fc95b3e