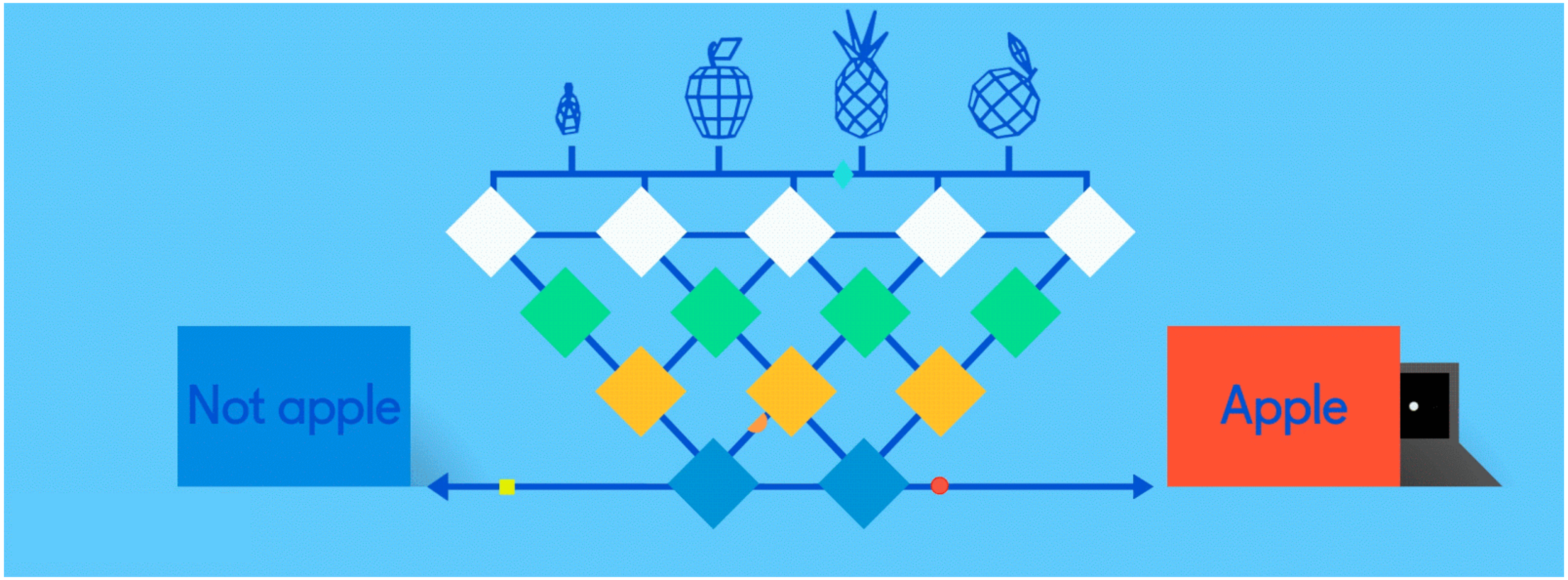
# Neural Networks for Machine Learning

## demonstrations



# Neural Network Architectures

Current focus on large networks with different “architectures” suited for different kinds of tasks

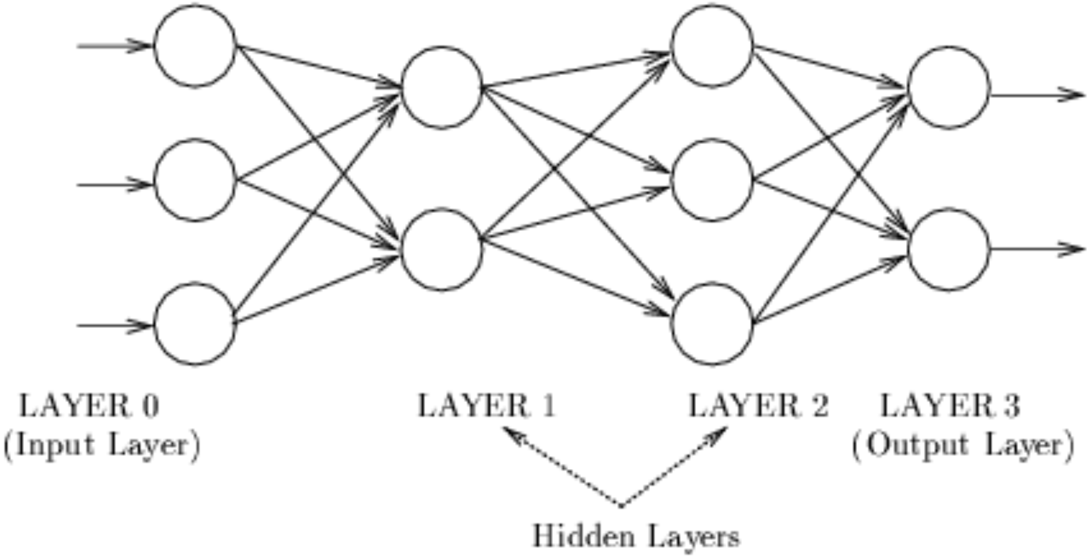
- Feedforward Neural Network
- CNN: Convolutional Neural Network
- RNN: Recurrent Neural Network
- LSTM: Long Short Term Memory
- GAN: Generative Adversarial Network

# Feedforward Neural Network

- Connections allowed from a node in layer *i* only to nodes in layer *i*+1

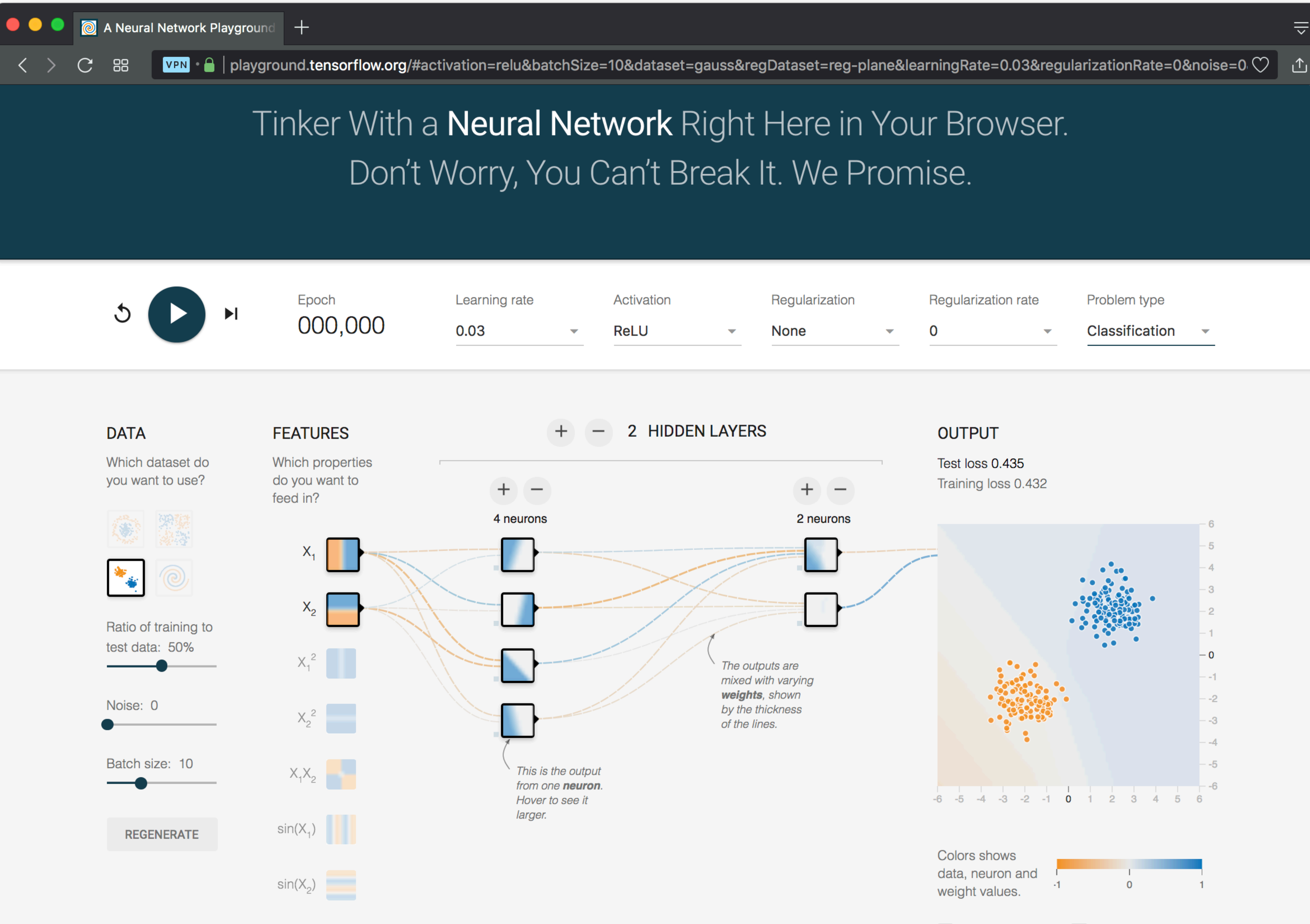
  i.e., no cycles or loops

- Simple, widely used architecture.



downstream nodes tend to successively abstract features from preceding layers

**HTTP://PLAYGROUND.TENSORFLOW.ORG/**



playground.tensorflow.org/#activation=relu&batchSize=10&dataset=gauss&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=0

Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.

Epoch 000,000

Learning rate 0.03

Activation ReLU

Regularization None

Regularization rate 0

Problem type Classification

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

Which properties do you want to feed in?

*This is the output from one* ***neuron****. Hover to see it larger.*

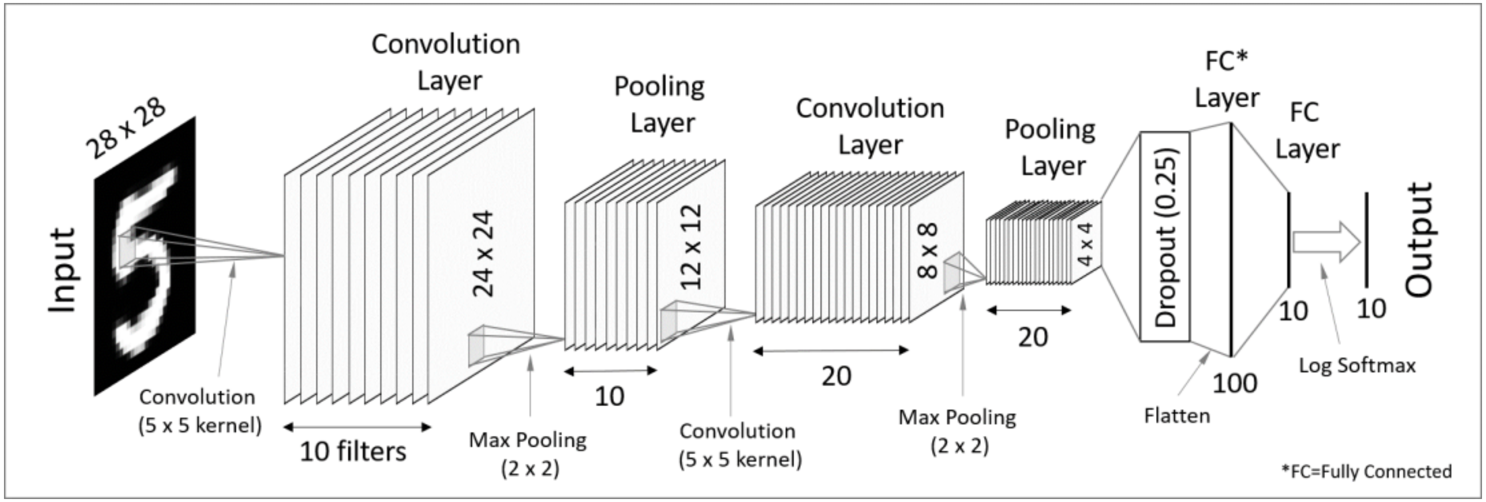
*The outputs are mixed with varying* ***weights****, shown by the thickness of the lines.*

Test loss 0.435
Training loss 0.432

Colors shows data, neuron and weight values.

**HTTP://PLAYGROUND.TENSORFLOW.ORG/**

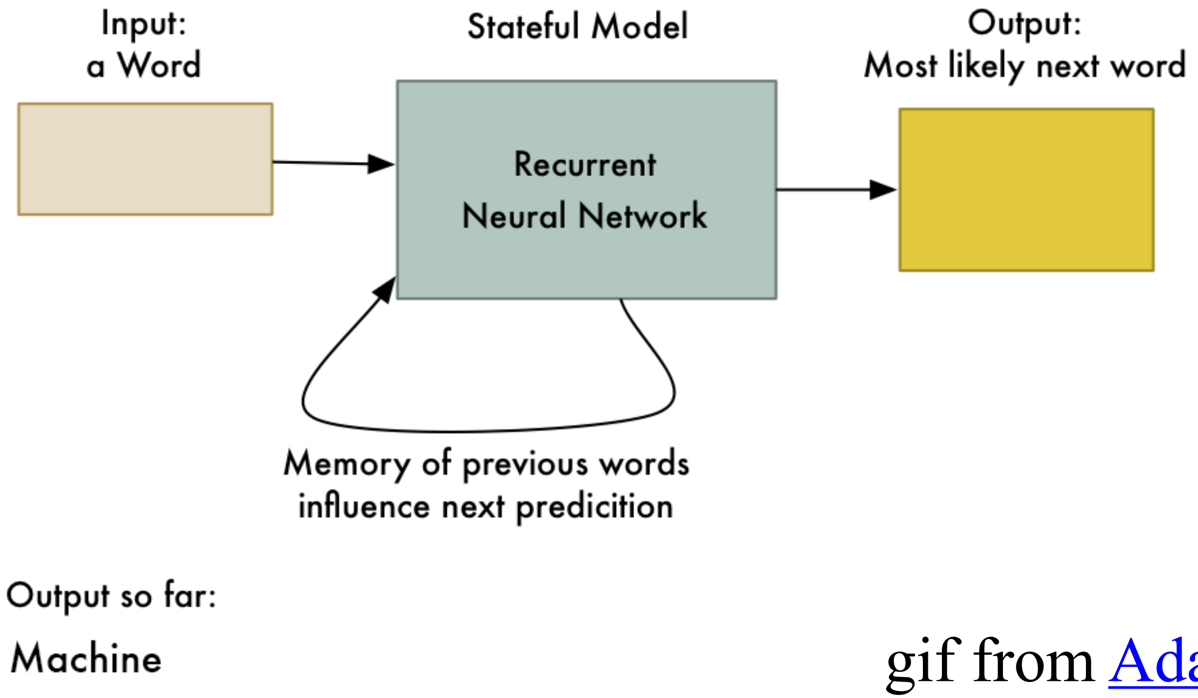
# CNN: Convolutional Neural Network



- Good for image processing: classification, object recognition, automobile lane tracking, etc.
- Classic demo: learn to recognize hand-written digits from MNIST data with 70K examples

# RNN: Recurrent Neural Networks

- Good for learning over sequences of data, e.g., a sentence orf words
- LSTM (Long Short Term Memory) a popular architecture



gif from Adam Geitgey

# Deep Learning Frameworks

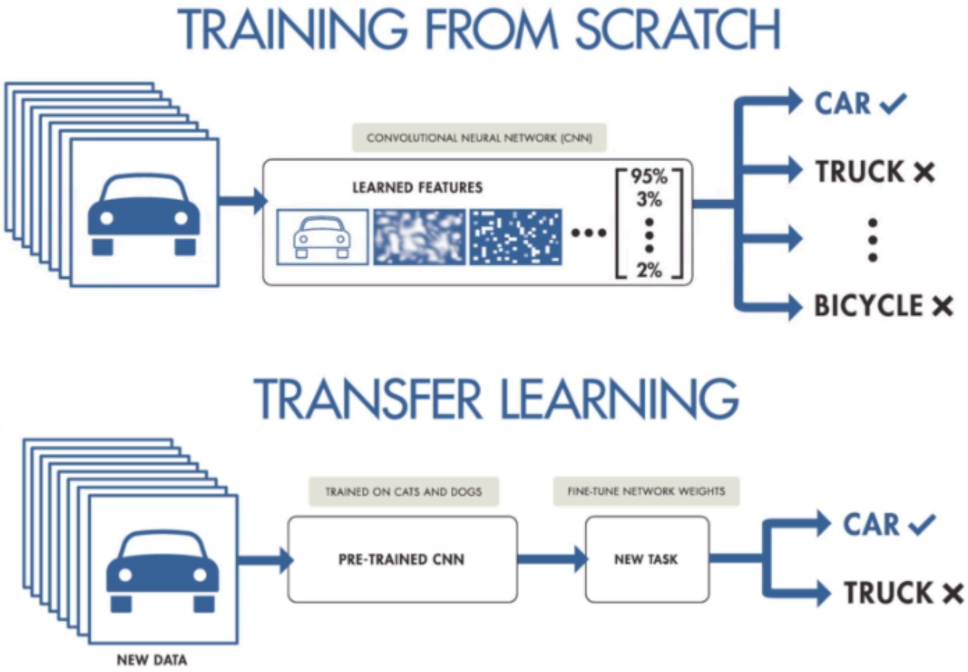
- Popular open source deep learning frameworks use Python at top-level; C++ in backend
  - TensorFlow (via Google)
  - PyTorch (via Facebook)
  - MxNet (Apache)
  - Caffe (Berkeley)
- Keras: popular API works with the first two and provides good support at architecture level

# Good at Transfer Learning

- Neural networks effective for transfer learning

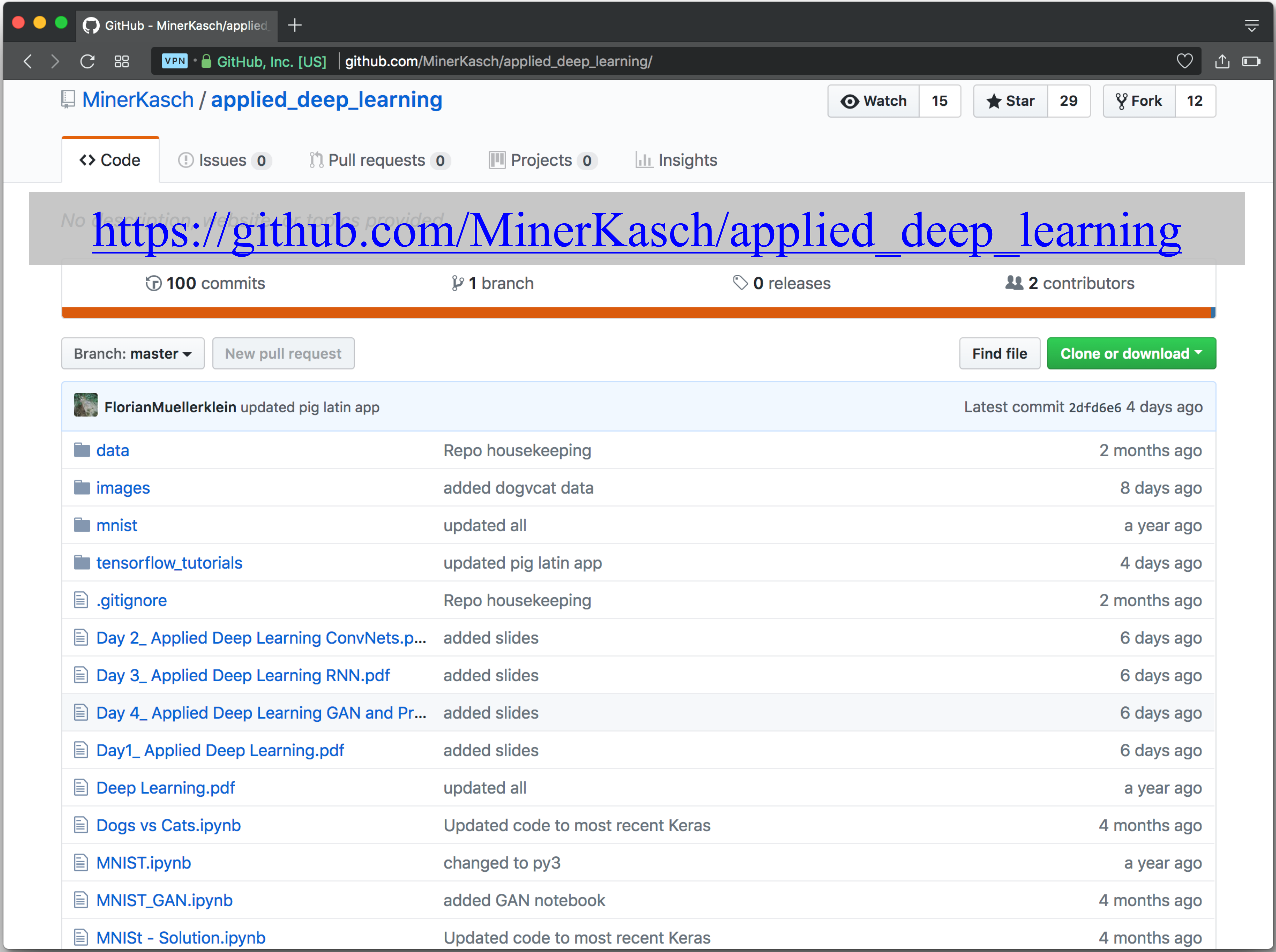
  Using parts of a model trained on a task as an initial model to train on a different task

- Particularly effective for image recognition



# Scikit-learn

- We’ll look at using sicikit-learn’s feed forward model on the iris dataset

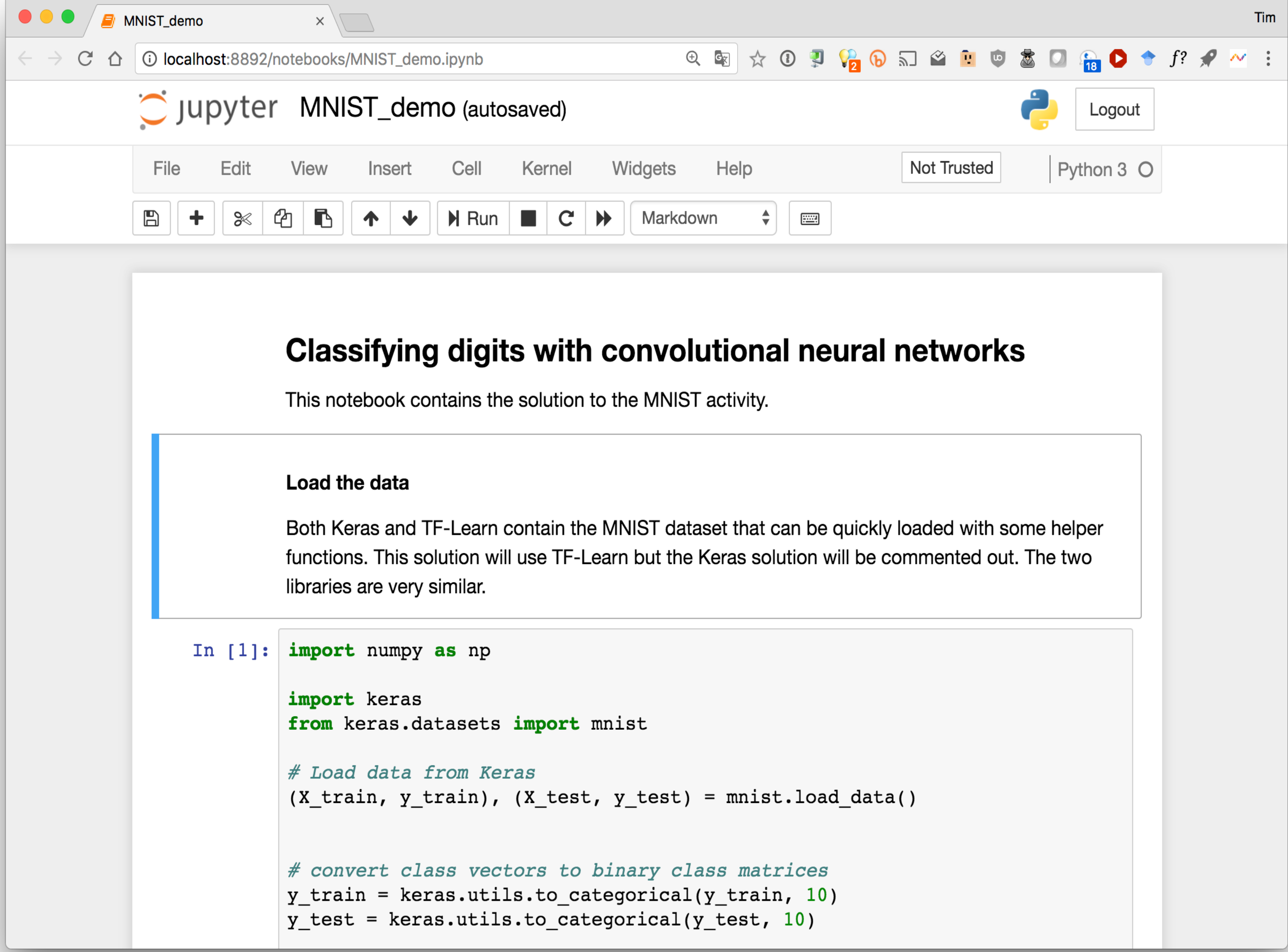


[https://github.com/MinerKasch/applied_deep_learning](https://github.com/MinerKasch/applied_deep_learning)

No description, website, or topics provided.

FlorianMuellerklein updated pig latin app Latest commit 2dfd6e6 4 days ago

| | | |
|---|---|---|
| data | Repo housekeeping | 2 months ago |
| images | added dogvcat data | 8 days ago |
| mnist | updated all | a year ago |
| tensorflow_tutorials | updated pig latin app | 4 days ago |
| .gitignore | Repo housekeeping | 2 months ago |
| Day 2_ Applied Deep Learning ConvNets.p... | added slides | 6 days ago |
| Day 3_ Applied Deep Learning RNN.pdf | added slides | 6 days ago |
| Day 4_ Applied Deep Learning GAN and Pr... | added slides | 6 days ago |
| Day1_ Applied Deep Learning.pdf | added slides | 6 days ago |
| Deep Learning.pdf | updated all | a year ago |
| Dogs vs Cats.ipynb | Updated code to most recent Keras | 4 months ago |
| MNIST.ipynb | changed to py3 | a year ago |
| MNIST_GAN.ipynb | added GAN notebook | 4 months ago |
| MNISt - Solution.ipynb | Updated code to most recent Keras | 4 months ago |



# Classifying digits with convolutional neural networks

This notebook contains the solution to the MNIST activity.

### Load the data

Both Keras and TF-Learn contain the MNIST dataset that can be quickly loaded with some helper functions. This solution will use TF-Learn but the Keras solution will be commented out. The two libraries are very similar.

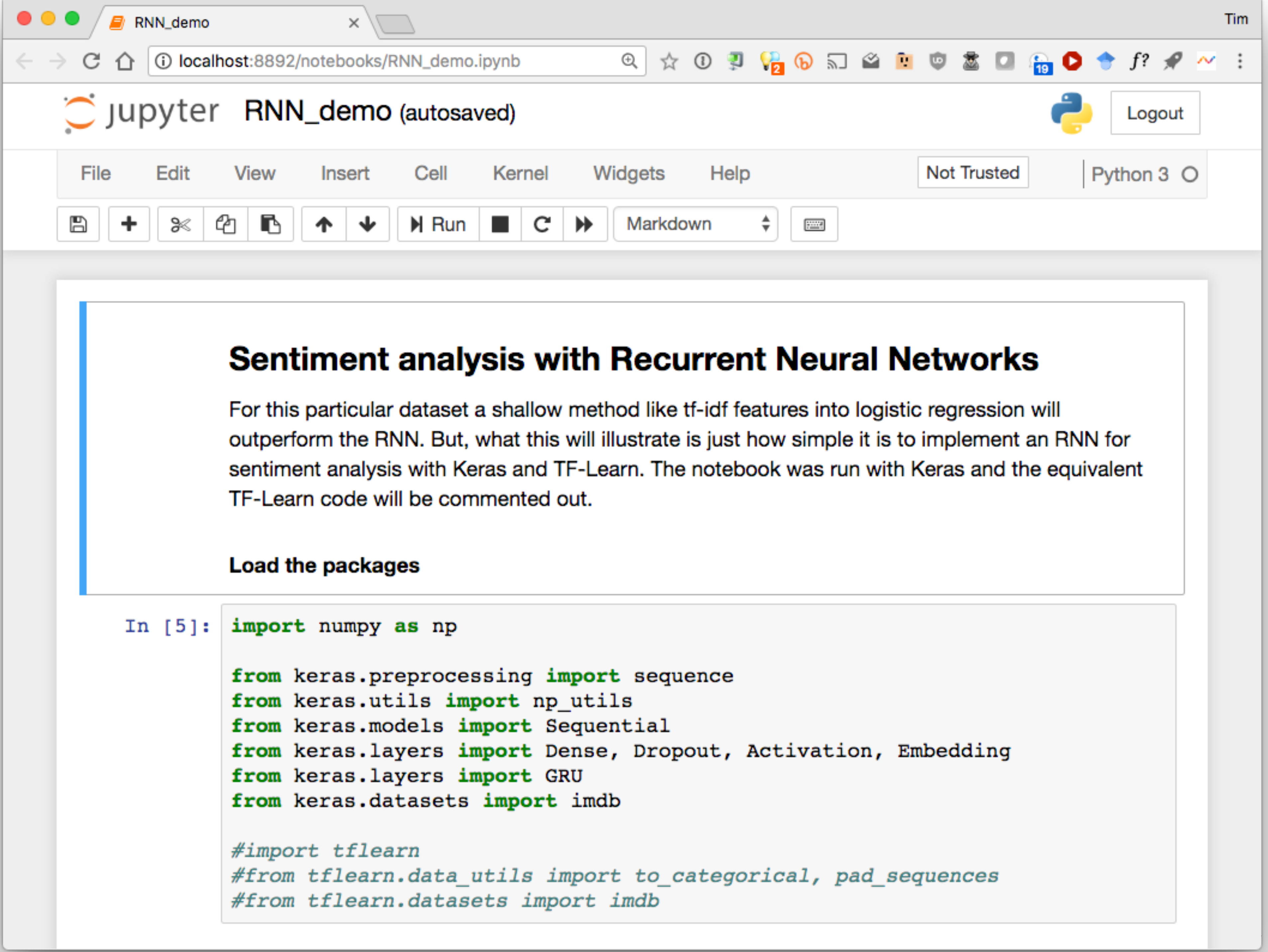
In [1]:

```
import numpy as np

import keras
from keras.datasets import mnist

# Load data from Keras
(X_train, y_train), (X_test, y_test) = mnist.load_data()


# convert class vectors to binary class matrices
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
```



## Sentiment analysis with Recurrent Neural Networks

For this particular dataset a shallow method like tf-idf features into logistic regression will outperform the RNN. But, what this will illustrate is just how simple it is to implement an RNN for sentiment analysis with Keras and TF-Learn. The notebook was run with Keras and the equivalent TF-Learn code will be commented out.

**Load the packages**

In [5]:

```
import numpy as np

from keras.preprocessing import sequence
from keras.utils import np_utils
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Embedding
from keras.layers import GRU
from keras.datasets import imdb

#import tflearn
#from tflearn.data_utils import to_categorical, pad_sequences
#from tflearn.datasets import imdb
```