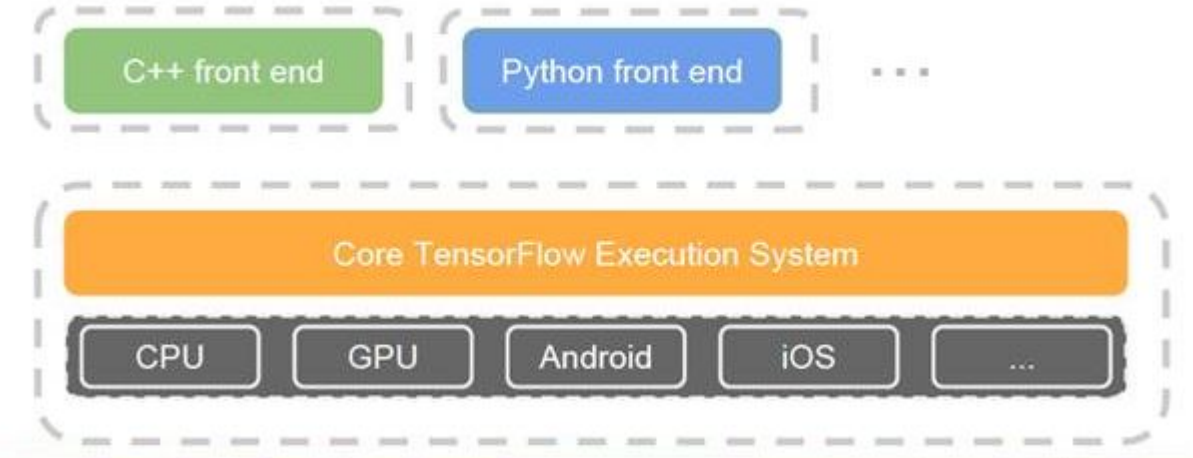
TensorFlow

Machine Learning for Everyone

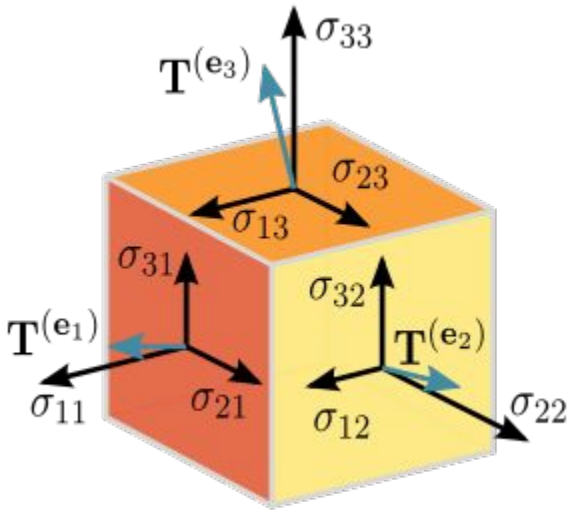
TensorFlow

- Open source software library for numerical computation using data flow graphs
  - nodes - mathematical operations
  - edges - multidimensional data arrays
- Python, C++, Go bindings
- GPU support
- Multi platform
- Released 9th November 2015

Architecture



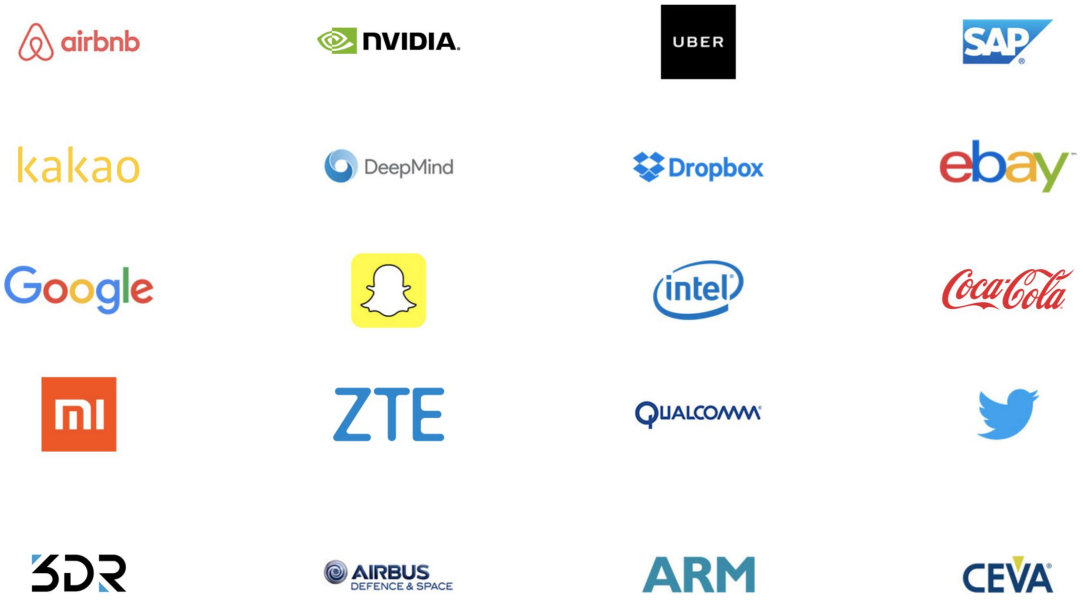
Tensor


σ_{33}
$\mathbf{T}^{(\mathbf{e}_3)}$
σ_{23}
σ_{13}
σ_{31}
$\mathbf{T}^{(\mathbf{e}_1)}$
σ_{11}
σ_{21}
σ_{32}
$\mathbf{T}^{(\mathbf{e}_2)}$
σ_{12}
σ_{22}

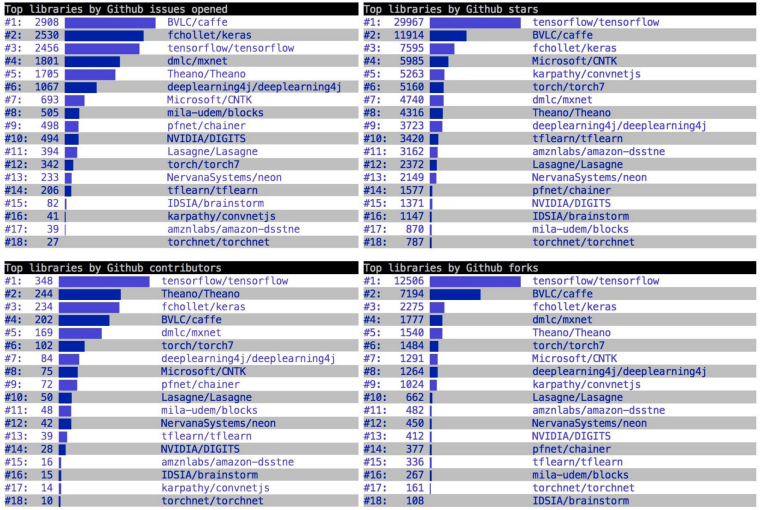
Google Product

- Take advantage of Big Data
- Machine Learning in:
  - Image search
  - Translation
  - Recommendations
  - … nearly all other Google products

Notable users



Popularity among ML/DS tools



Library	Rank	Overall	Github	Stack Overflow	Google Results
tensorflow	1	10.8676777173	4.25282914794	4.371905768	2.24294280139
keras	2	1.92768682345	0.613405340454	0.830444013135	0.483837469861
caffe	3	1.85536658344	1.00172325244	0.301598379669	0.552044951334
theano	4	0.757142065184	-0.156657475854	0.361637072631	0.552162468406
pytorch	5	0.481418742361	-0.198079135346	-0.30225967424	0.981757551946
sonnet	6	0.427865682184	-0.326074511957	-0.361634296039	1.11557449018
mxnet	7	0.0987996914674	0.121327235453	-0.306328604959	0.283801060973
torch	8	0.00559731666893	-0.153332101969	-0.00824393023136	0.167173348869
cntk	9	-0.0205203098963	0.0965088202554	-0.282173869559	0.165144739407
dlib	10	-0.599823512154	-0.39578194316	-0.223382454956	0.0193408859617
caffe2	11	-0.671062928351	-0.274071118159	-0.359648165565	-0.0373436446266
chainer	12	-0.70151841136	-0.400397905813	-0.234603397931	-0.0665171076164
paddlepaddle	13	-0.833003782881	-0.267123408237	-0.366884083295	-0.198996291348
deeplearning4j	14	-0.893319117931	-0.0575131634759	-0.321347169592	-0.514458784863

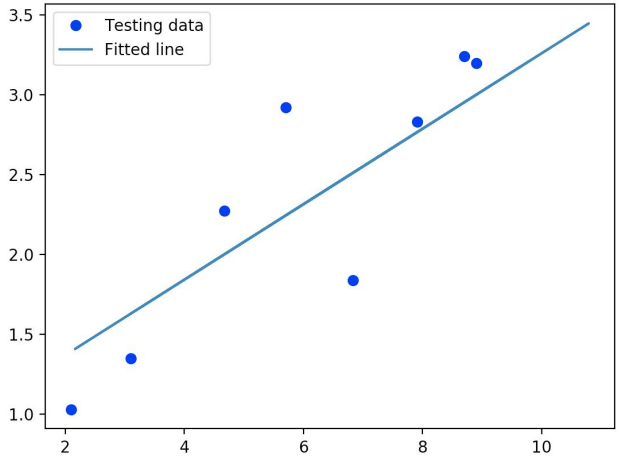
Rule them all

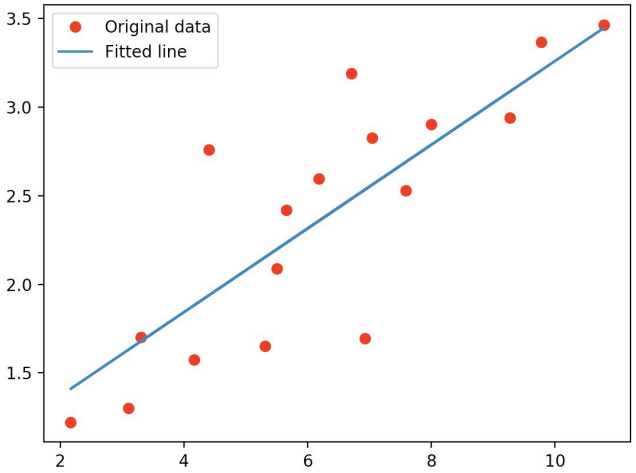
Who uses ML?

- Researchers
- Data Scientists
- Developers

Same toolset to collaborate with each other

Linear Regression





Linear Regression

```
# Parameters

learning_rate = 0.01
training_epochs = 1000
display_step = 50

train_X = numpy.asarray([3.3,4.4,5.5,6.71,6.93,4.168,9.779,6.182,7.59,2.167,
                         7.042,10.791,5.313,7.997,5.654,9.27,3.1])
train_Y = numpy.asarray([1.7,2.76,2.09,3.19,1.694,1.573,3.366,2.596,2.53,1.221,
                         2.827,3.465,1.65,2.904,2.42,2.94,1.3])
n_samples = train_X.shape[0]

# tf Graph Input
X = tf.placeholder("float")
Y = tf.placeholder("float")

# Set model weights
W = tf.Variable(rng.randn(), name="weight")
b = tf.Variable(rng.randn(), name="bias")

# Construct a linear model
pred = tf.add(tf.multiply(X, W), b)

# Mean squared error
cost = tf.reduce_sum(tf.pow(pred-Y, 2))/(2*n_samples)
# Gradient descent
#  Note, minimize() knows to modify W and b because Variable objects are trainable=True by default
optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cost)

# Initialize the variables (i.e. assign their default value)
init = tf.global_variables_initializer()
```

```
# Start training
with tf.Session() as sess:

    # Run the initializer
    sess.run(init)

    # Fit all training data
    for epoch in range(training_epochs):
        for (x, y) in zip(train_X, train_Y):
            sess.run(optimizer, feed_dict={X: x, Y: y})

        # Display logs per epoch step
        if (epoch+1) % display_step == 0:
            c = sess.run(cost, feed_dict={X: train_X, Y:train_Y})
            print("Epoch:", '%04d' % (epoch+1), "cost=", "{:.9f}".format(c), \
                "W=", sess.run(W), "b=", sess.run(b))

    print("Optimization Finished!")
    training_cost = sess.run(cost, feed_dict={X: train_X, Y: train_Y})
    print("Training cost=", training_cost, "W=", sess.run(W), "b=", sess.run(b), '\n')

    # Graphic display
    plt.plot(train_X, train_Y, 'ro', label='Original data')
    plt.plot(train_X, sess.run(W) * train_X + sess.run(b), label='Fitted line')
    plt.legend()
    plt.show()
```

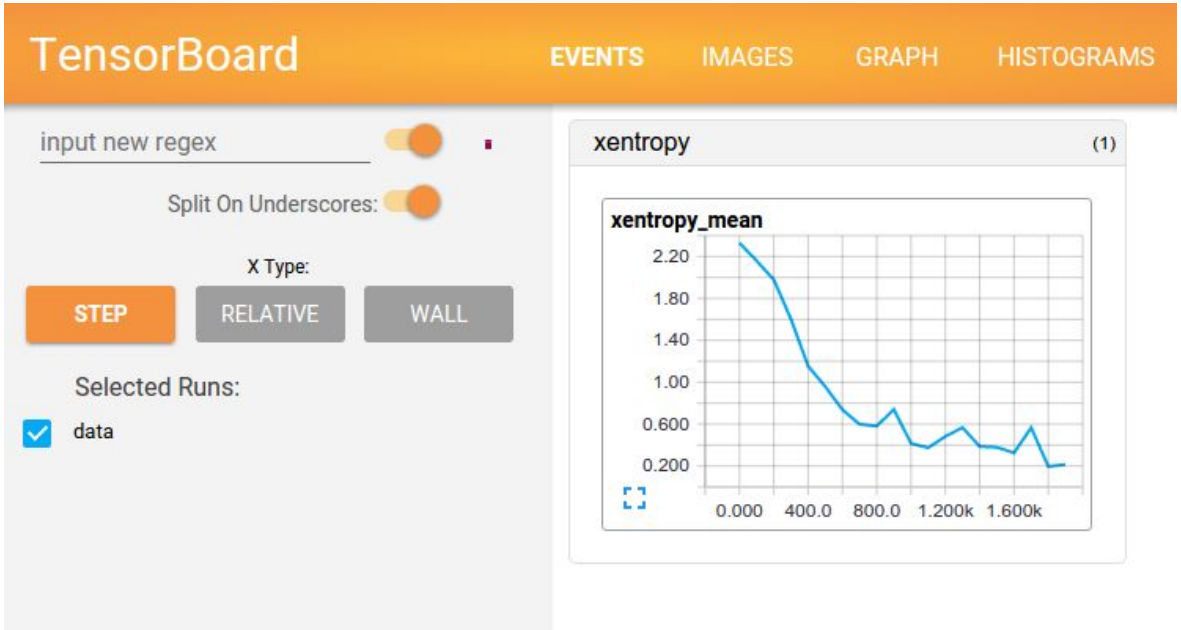
Keras

- Open source neural network library written in Python
- Backends: MXNet, Deeplearning4j, Tensorflow, CNTK or Theano
- In 2017, Google's TensorFlow team decided to support Keras in TensorFlow's core library

TensorBoard

- Training a network is often complex and confusing
- TensorBoard is a set of visualization tools which helps:
  - Optimize
  - Debug
  - Understand
  - Present results
- Graphs, plots, metrics and other stuff in the browser (localhost)

TensorBoard



Thoughts

- Effective and fast (written in C++) library for ML
- High level API
- Good for experimenting and production ready
- Community and big companies
- Learning materials

Dark side

- API level may be too low for many use cases
- There are faster solutions (Torch, Caffe, Cognitive Services)

Resources and useful links

- https://www.tensorflow.org/
- https://www.youtube.com/watch?v=2FmcHiLCwTU
- https://github.com/aymericdamien/TensorFlow-Examples
- https://www.datacamp.com/community/tutorials/tensorflow-tutorial