APKBUS

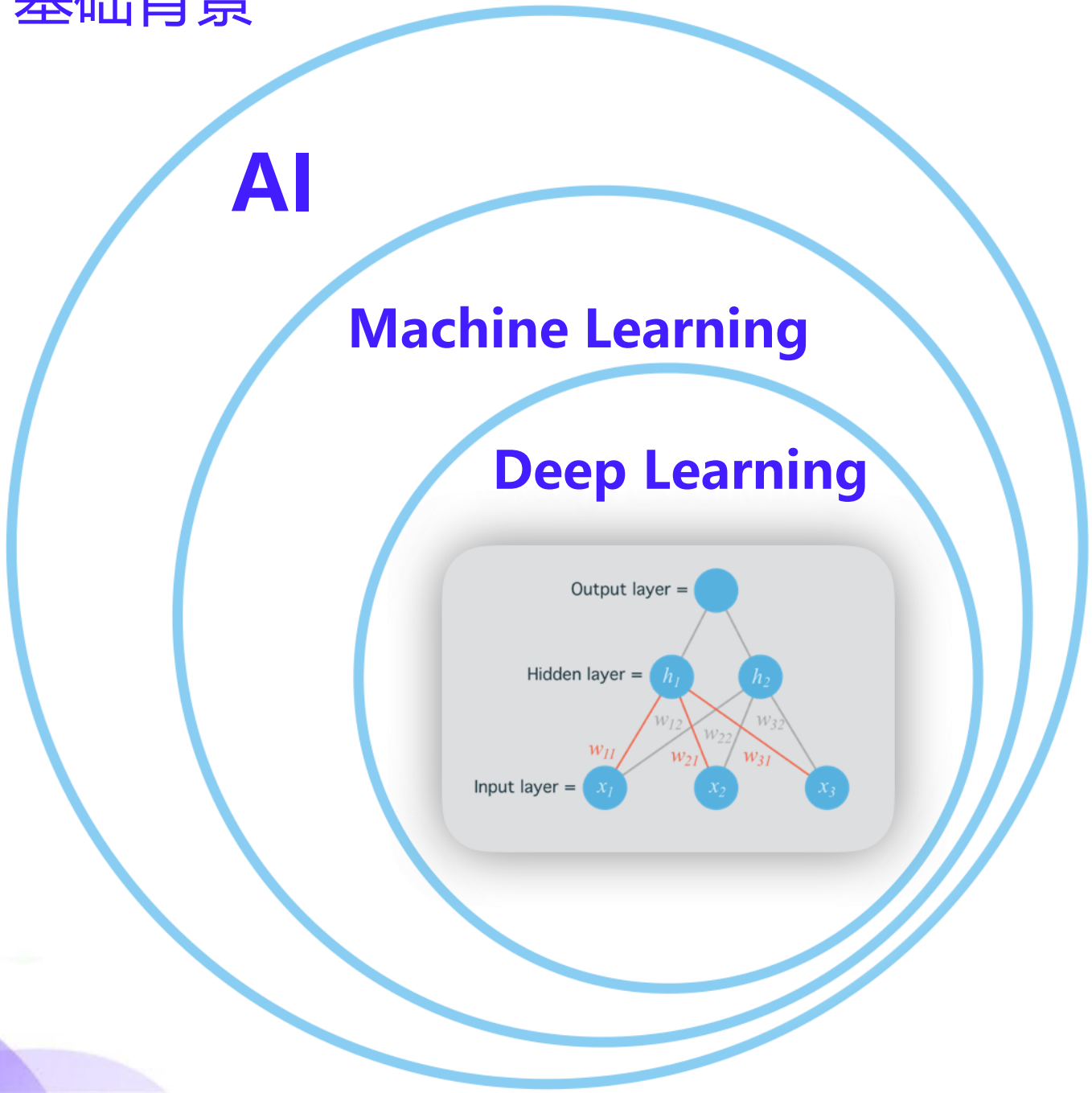
AI Workshop

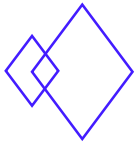
江骏

APKBUS

1. AI 基础背景 & TensorFlow 简介
2. 电商领域的应用
3. 开发环境

AI 基础背景


AI
Machine Learning
Deep Learning
Output layer =
Hidden layer =
Input layer =
h_1
h_2
x_1
x_2
x_3
w_{11}
w_{12}
w_{21}
w_{22}
w_{31}
w_{32}



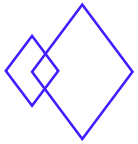
TensorFlow 的由来

TensorFlow 的上一代产品 DistBelief

- 解决了数据量 > GPU 最大内存的问题
- paper 中提到最大的 model，达到了
  1.7 billion parameters, utilizing 81 machines, delivering a 12x speedup.
- 它的缺点：
  - 擅长图像识别，但对其它机器学习 model 适用性差，
    而且不支持 mobile
  - 维护大规模系统成为负担，缺乏抽象。

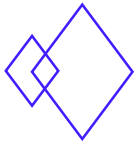
TensorFlow 多种设备

TensorFlow 生态



为什么选择 TensorFlow?

- 从开源的角度，社区活跃
- 从框架的角度，涉及的面越来越全 (TFF, TFP, TFA)
- 从工程的角度，已经有比较丰富的工程选择 (TFX, kubeflow)
- 从公司、团队的角度，有利于技术积累



电商场景的应用

搜索推荐	营销	安全风控	业务指标
点击率	红包	异常行为检测	配送时长
GMV	商家代金券	网络攻击	订单量
			预测用户数量
			...

Weight

Bias

Activation Function

Loss

What is machine learning?

Learning Rate

Gradient Descent

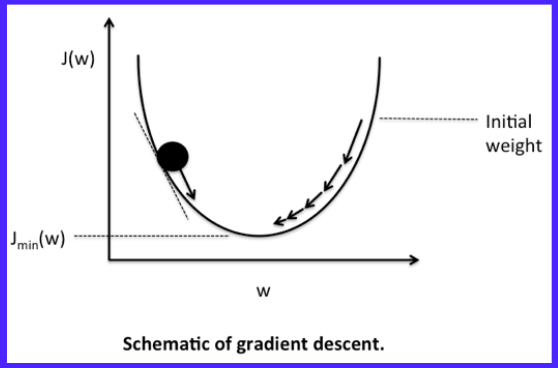
Hidden Layer

Machine Learning 基本原理

神经元 neural

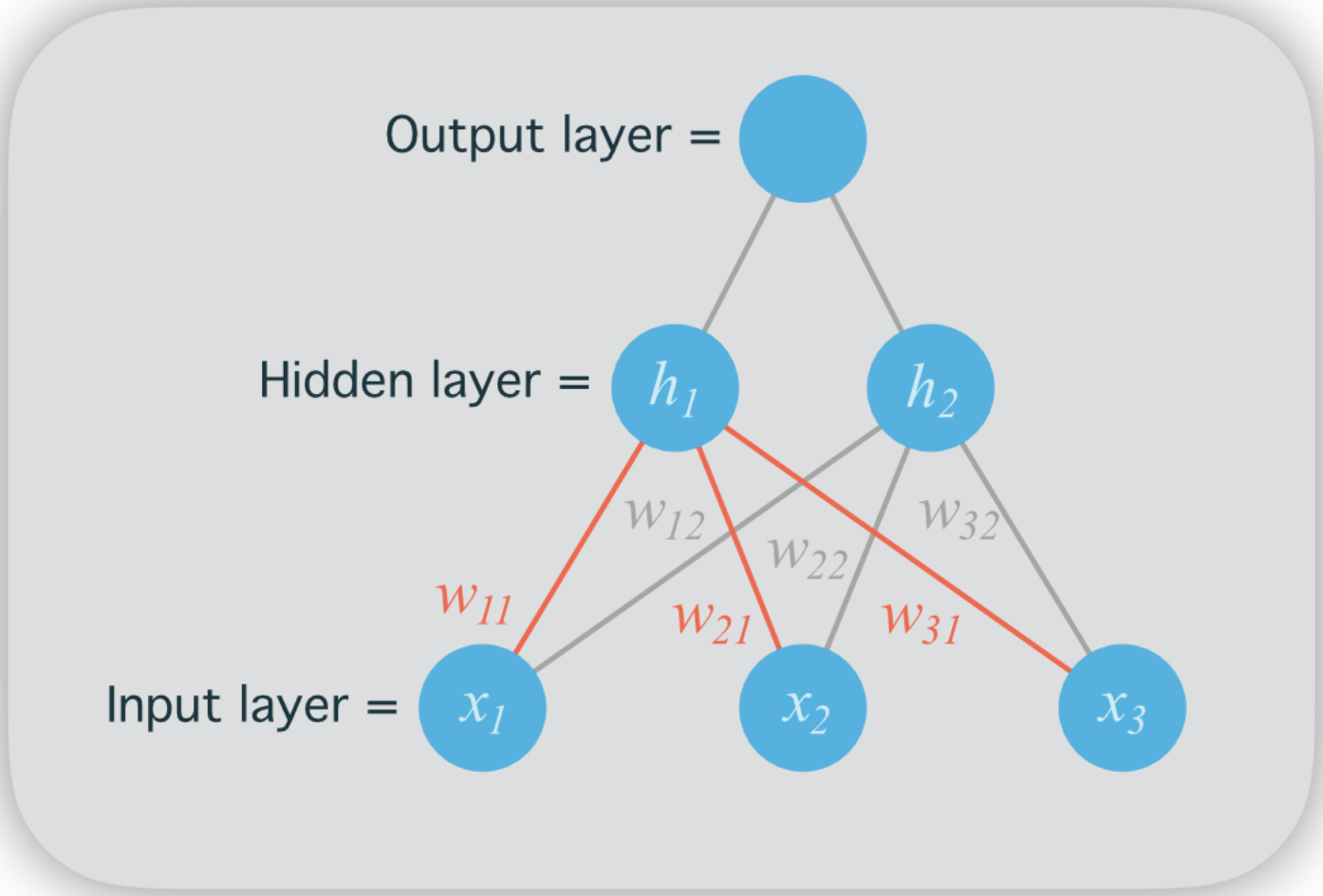
- 输入 x
- 计算 $w^T x + b$ ，把结果代入 activation function，得到输出值 $\bar{y}$

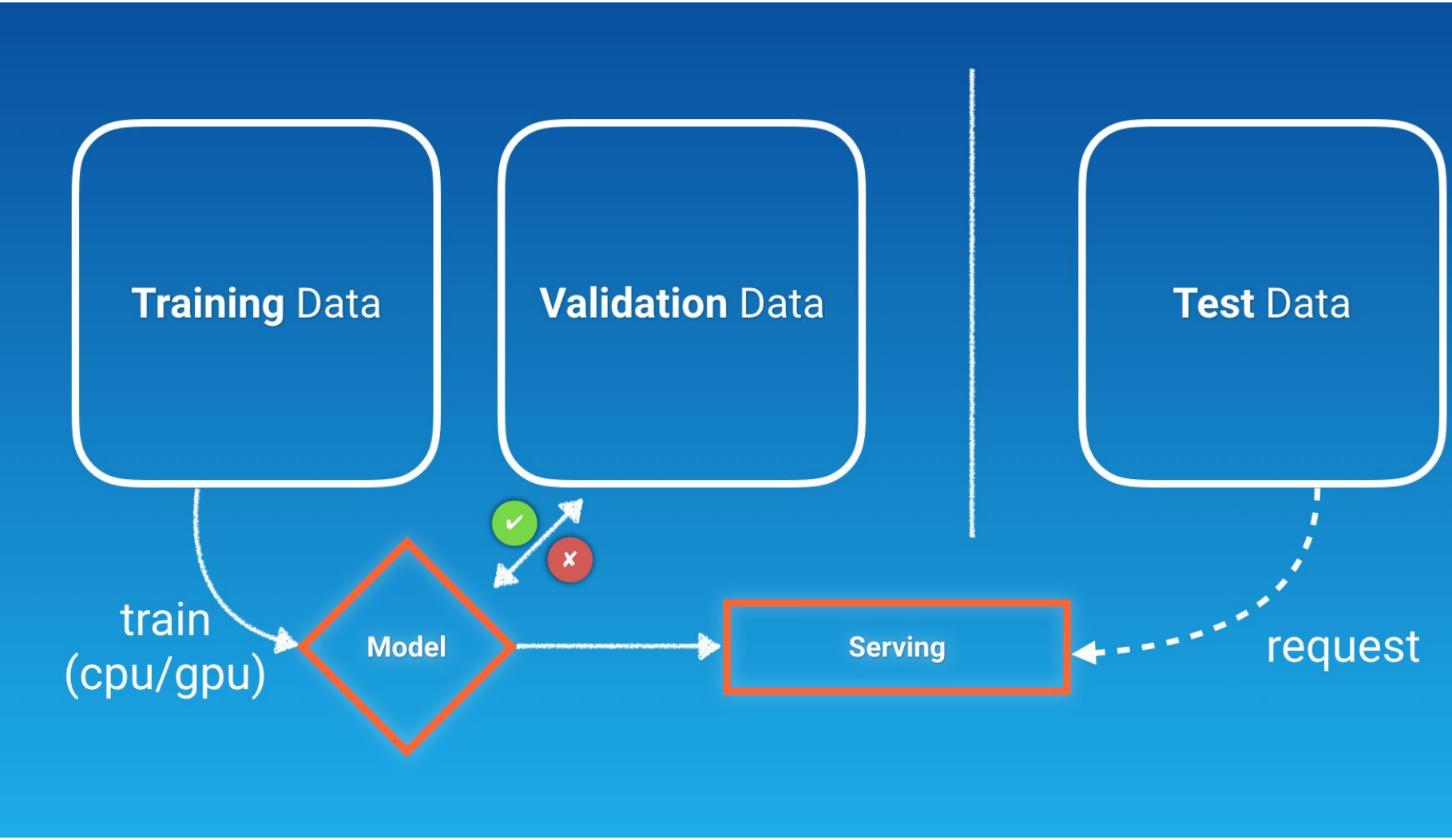
- 预测得准吗？Loss function 来衡量。比如，用它们的距离 $loss = (y - \bar{y})^2$
- 目标：让 loss 值越小越好
- 用 Optimizer 调整 weight 和 bias
  （比如用 Gradient Descent）

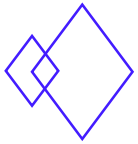


Schematic of gradient descent.

Deep Learning



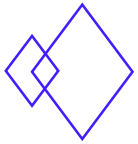

Training Data
Validation Data
Test Data
train
(cpu/gpu)
Model
Serving
request



Python modules

```
In [1]: import sys

In [2]: sys.path
Out[2]:
['/Users/ohmystack/.pyenv/versions/3.6.8/envs/py3.6/bin',
 '/Users/ohmystack/.pyenv/versions/3.6.8/lib/python36.zip',
 '/Users/ohmystack/.pyenv/versions/3.6.8/lib/python3.6',
 '/Users/ohmystack/.pyenv/versions/3.6.8/lib/python3.6/lib-dynload',
 '',
 '/Users/ohmystack/.pyenv/versions/3.6.8/envs/py3.6/lib/python3.6/site-packages',
 '/Users/ohmystack/.pyenv/versions/3.6.8/envs/py3.6/lib/python3.6/site-packages/IPython/extensions',
 '/Users/ohmystack/.ipython']
```

环境变量：PYTHONPATH

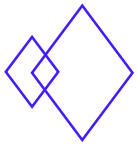


Virtualenv

pip install virtualenv

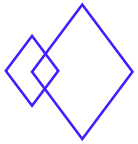
virtualenv –p python3 dev

source dev/bin/activate



Pyenv

```
brew install readline xz zlib
brew install pyenv pyenv-virtualenv

pyenv install --list
pyenv install 3.6.8
CPPFLAGS="-I$(brew --prefix zlib)/include" pyenv install 3.6.8

pyenv global 3.6.8

pyenv virtualenv
# List
pyenv virtualenvs
# Create
pyenv virtualenv  py3
# Activate
pyenv activate py3
```



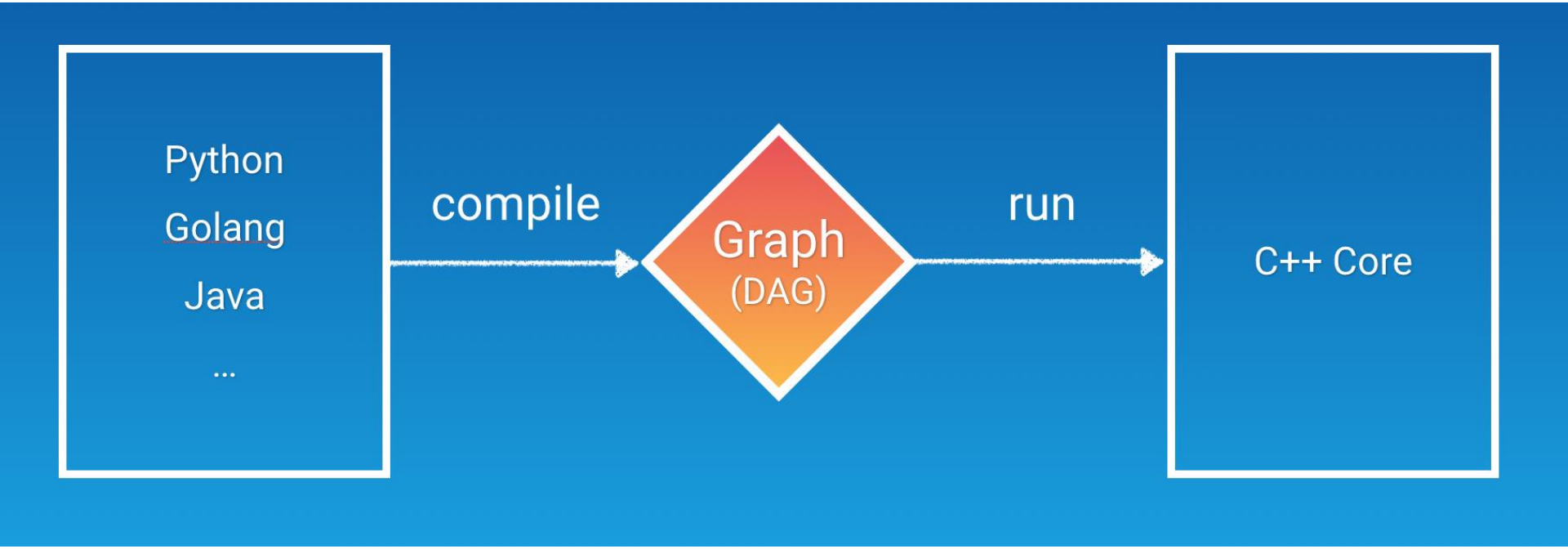
Install TensorFlow

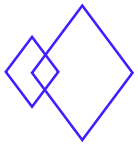
https://www.tensorflow.org/install/pip

Jupyter Notebook

```
pip install ipython ipykernel

# For Python2
pip install 'ipython<6.0'
pip install 'ipykernel<5.0' jupyter_client

ipython kernelspec list

python -m ipykernel install \
  --user --name xxx-env \
  --display-name "Python (xxx-env)"

jupyter notebook
```

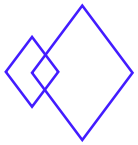
Graph Mode



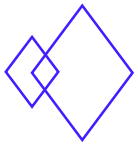


Eager Mode

```
import tensorflow as tf
x = [[2.]]
m = tf.matmul(x, x)

# The 1x1 matrix [[4.]]
```



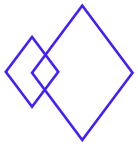
```
a = tf.constant(12)
counter = 0
while not tf.equal(a, 1):
  if tf.equal(a % 2, 0):
    a = a / 2
  else:
    a = 3 * a + 1
  print(a)
```



Gradients

```
def square(x):
  return tf.multiply(x, x)

grad = tfe.gradients_function(square)

print(square(3.))     # [9.]
print(grad(3.))       # [6.]
```

```
gradgrad = tfe.gradients_function(lambda x: grad(x)[0])

print(gradgrad(3.))  # [2.]
```



```
X = tf.constant(X)
y = tf.constant(y)

a = tf.get_variable('a', dtype=tf.float32, shape=[], initializer=tf.zeros_initializer)
b = tf.get_variable('b', dtype=tf.float32, shape=[], initializer=tf.zeros_initializer)
variables = [a, b]

num_epoch = 10000
optimizer = tf.train.GradientDescentOptimizer(learning_rate=1e-3)
for e in range(num_epoch):
    # 使用tf.GradientTape()记录损失函数的梯度信息
    with tf.GradientTape() as tape:
        y_pred = a * X + b
        loss = 0.5 * tf.reduce_sum(tf.square(y_pred - y))
    # TensorFlow自动计算损失函数关于自变量（模型参数）的梯度
    grads = tape.gradient(loss, variables)
    # TensorFlow自动根据梯度更新参数
    optimizer.apply_gradients(grads_and_vars=zip(grads, variables))
```

https://developers.google.com/machine-learning/crash-course/representation/feature-engineering