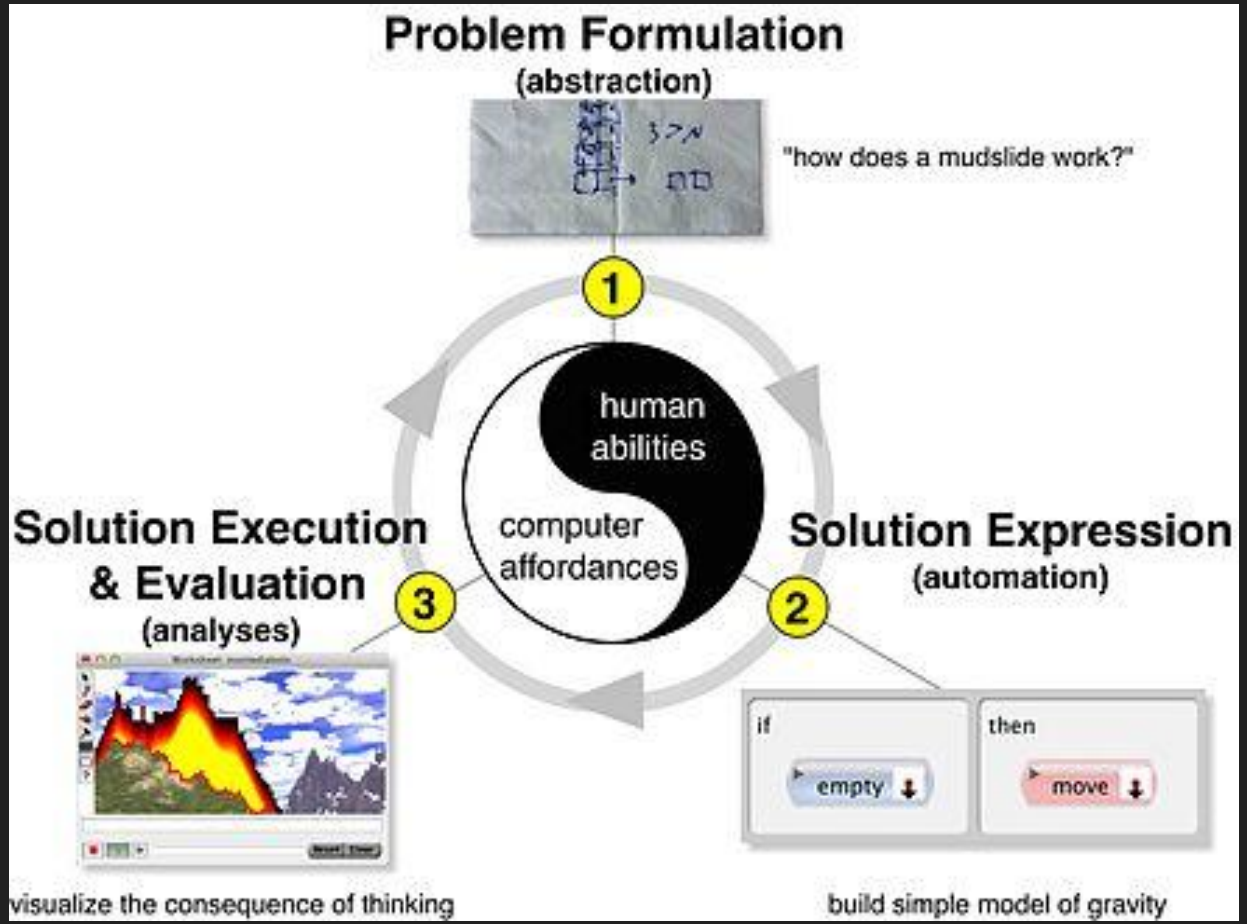
# 人工智能知识分享（二）

以TensorFlow为例

# Computational thinking



AI＞ML＞DL＞CNN

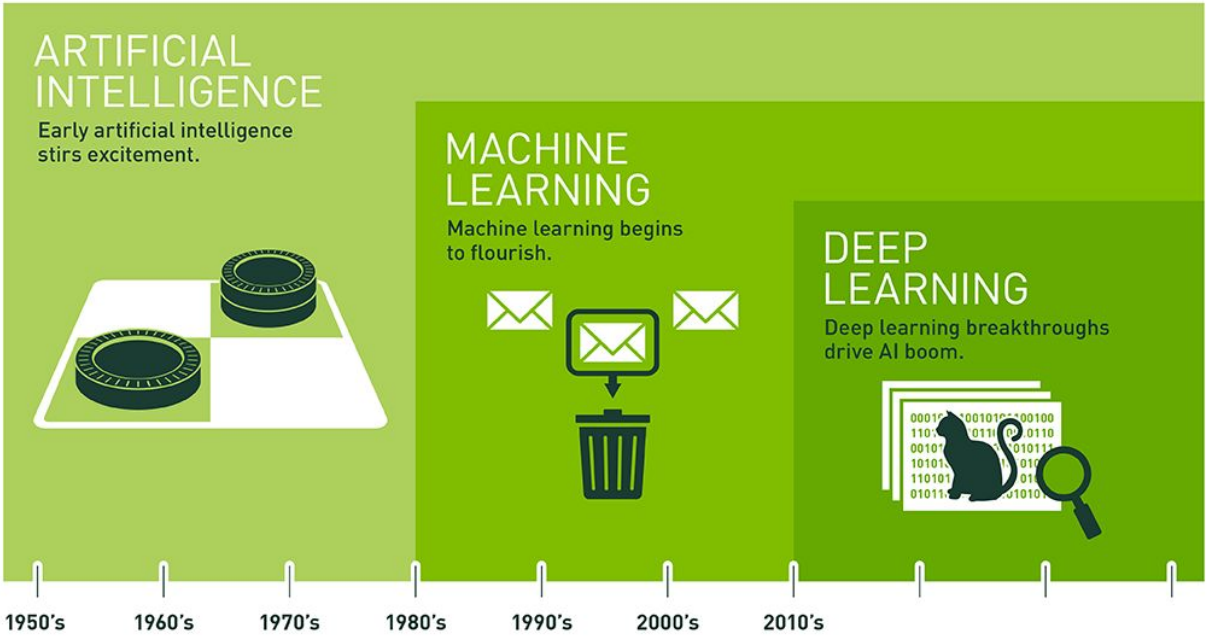
AI　人工智能 1950

ML 机器学习　1980

DL　深度学习　2010

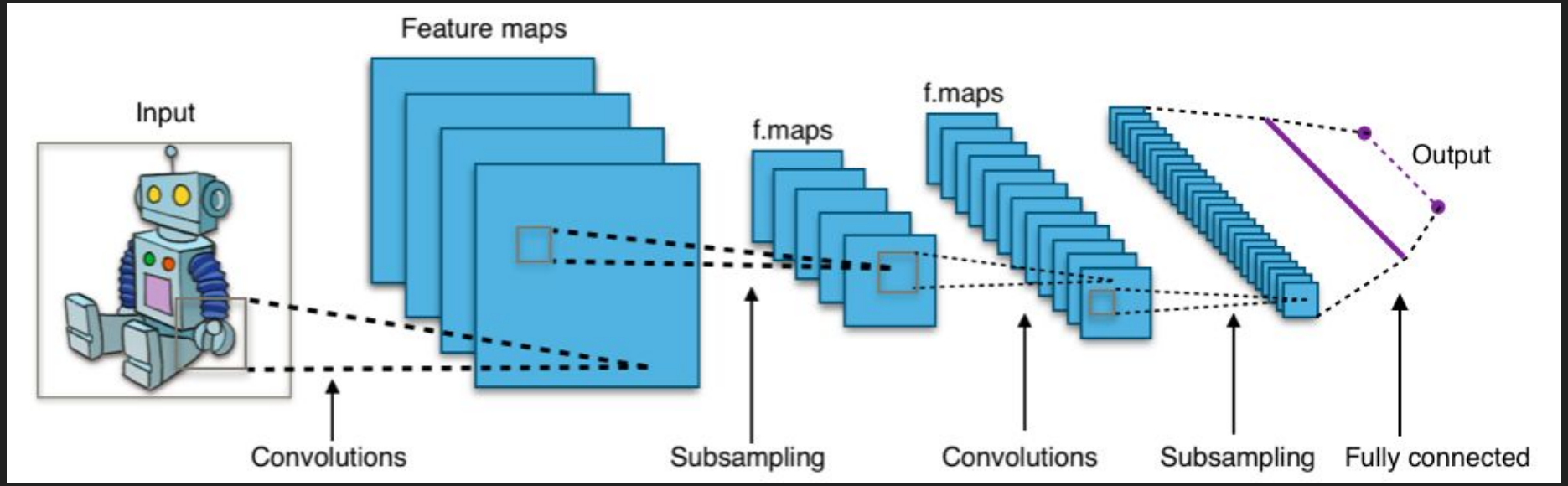
CNN　卷积神经网络

RNN　循环神经网络




Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

# CNN


Input
Feature maps
f.maps
f.maps
Output
Convolutions
Subsampling
Convolutions
Subsampling
Fully connected

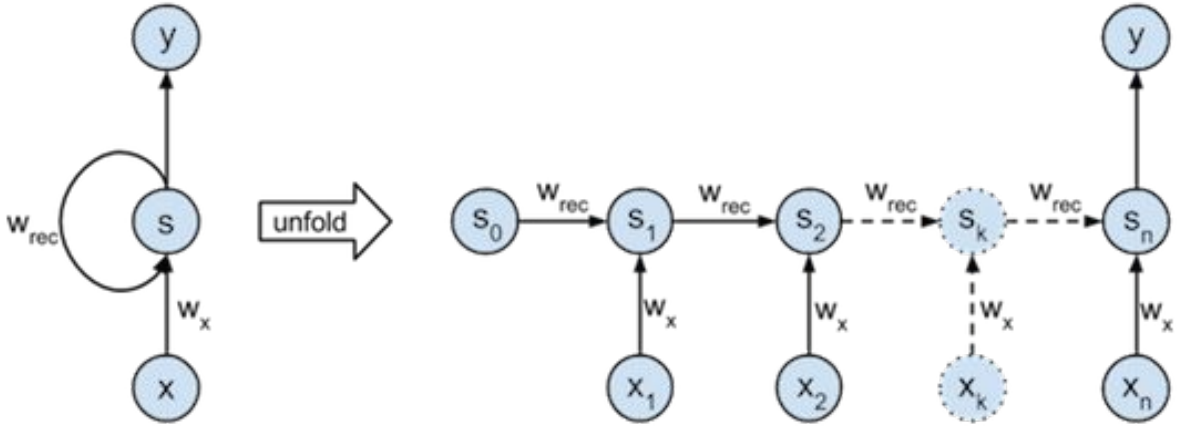
# RNN（exm1）

问题：计数

输入x 为二进制数据中的1bit

输出y为数据中1的个数

RNN展开等价于 (n+1)层神经网络，共享参数.

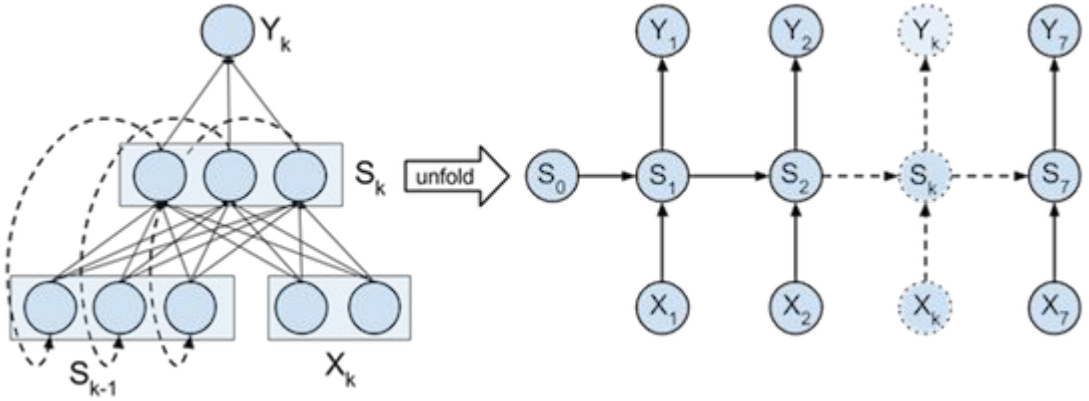



$$S_k = f(S_{k-1} * W_{rec} + X_k * W_x)$$

# RNN 应用

handwriting recognition

speech recognition




$$S_k = f(S_{k-1} * W_{rec} + X_k * W_x)$$

# RNN 代码

分析讨论

可视化

# Activation Functions

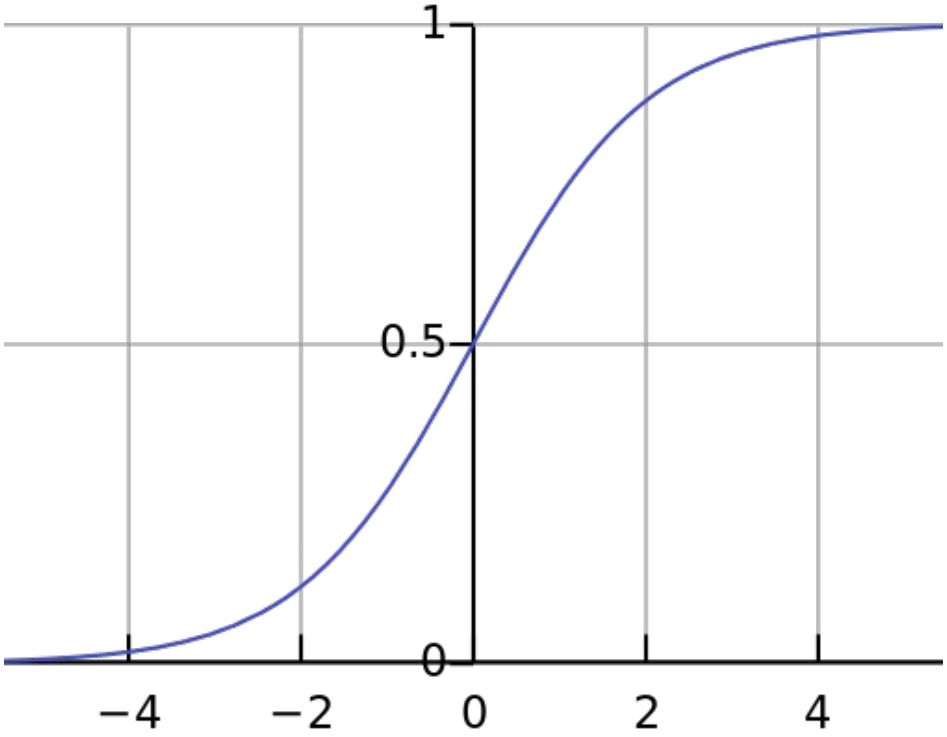
nonlinearities (sigmoid, tanh, elu, softplus, and softsign),

continuous but not everywhere differentiable functions (relu, relu6, crelu relu_x),

random regularization (dropout)

**tf.sigmoid(x, name=None)**

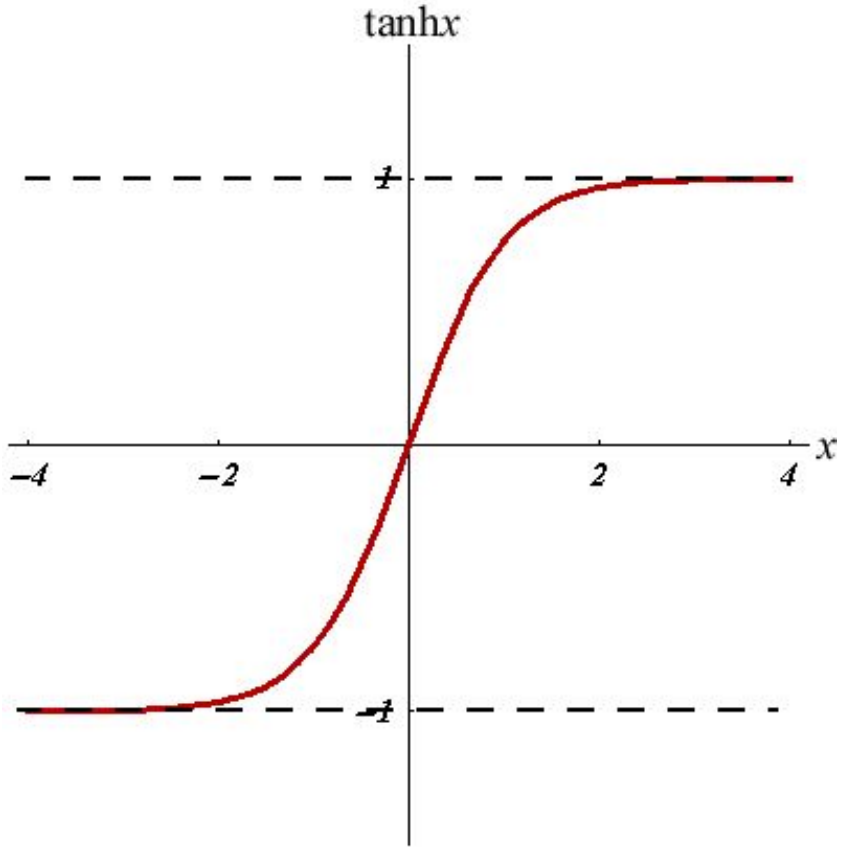
A logistic function or logistic curve is a common "S" shape (sigmoid curve)




$$S(t) = \frac{1}{1 + e^{-t}}.$$

**tf.tanh(x, name=None)**

Computes hyperbolic tangent of x element-wise

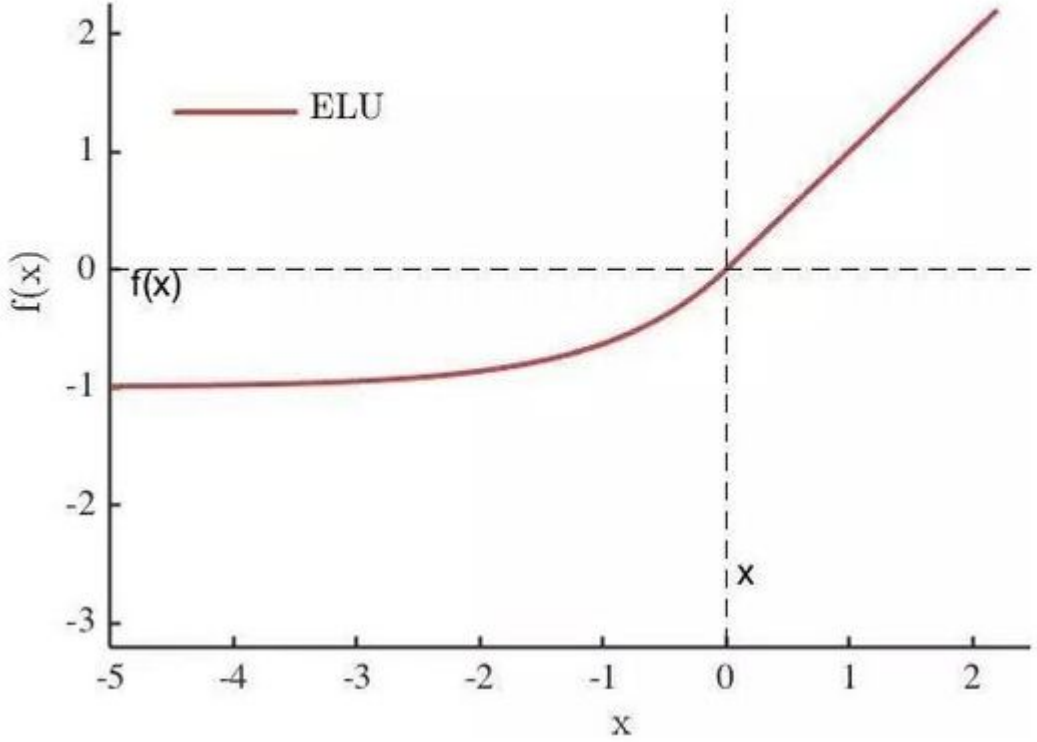



$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

**tf.elu(x, name=None)**

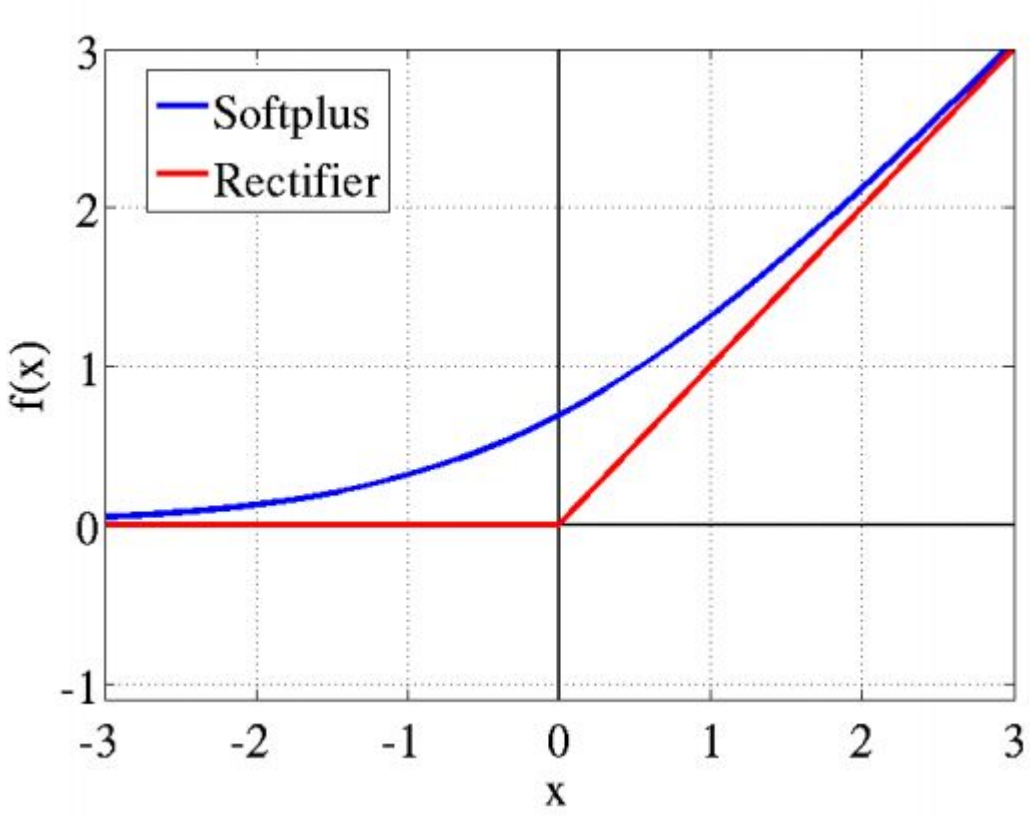
Exponential Linear Unit

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \leq 0 \end{cases}$$



**tf.softplus(x, name=None)**
**tf.**relu**(x, name=None)**

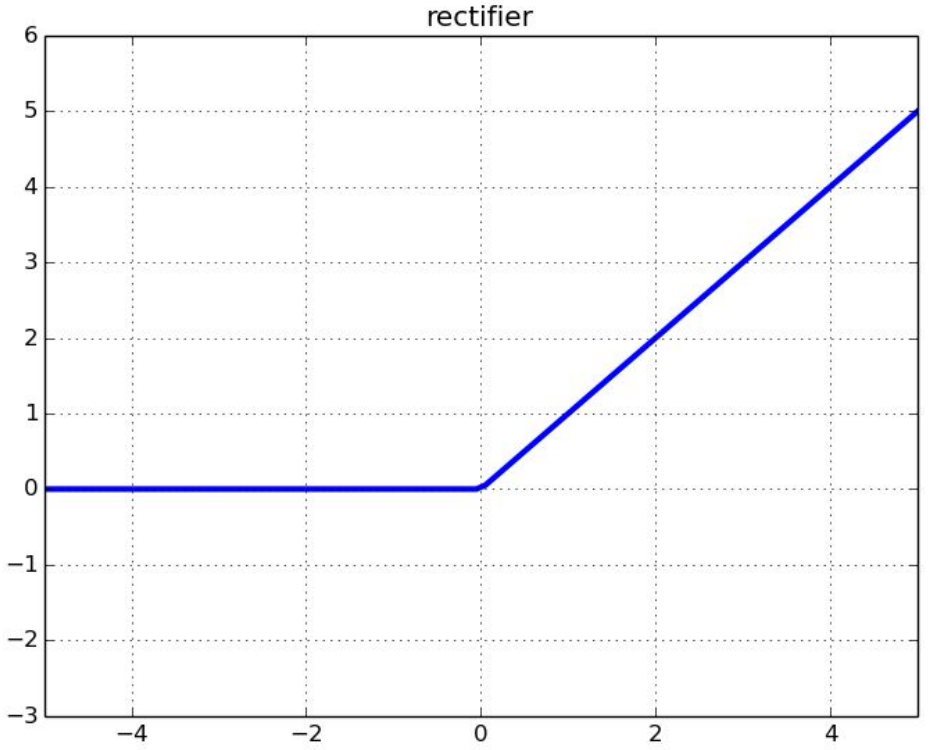
Computes




$$f(x) = \ln(1 + e^x)$$

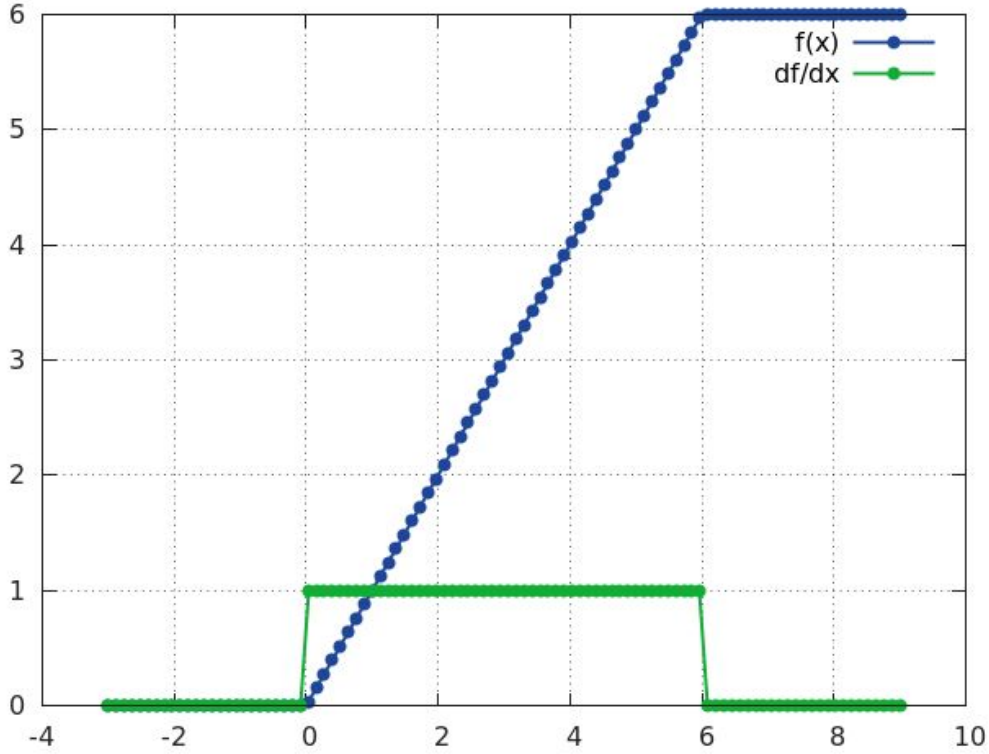
**tf.**relu**(x, name=None)**

max(x, 0)

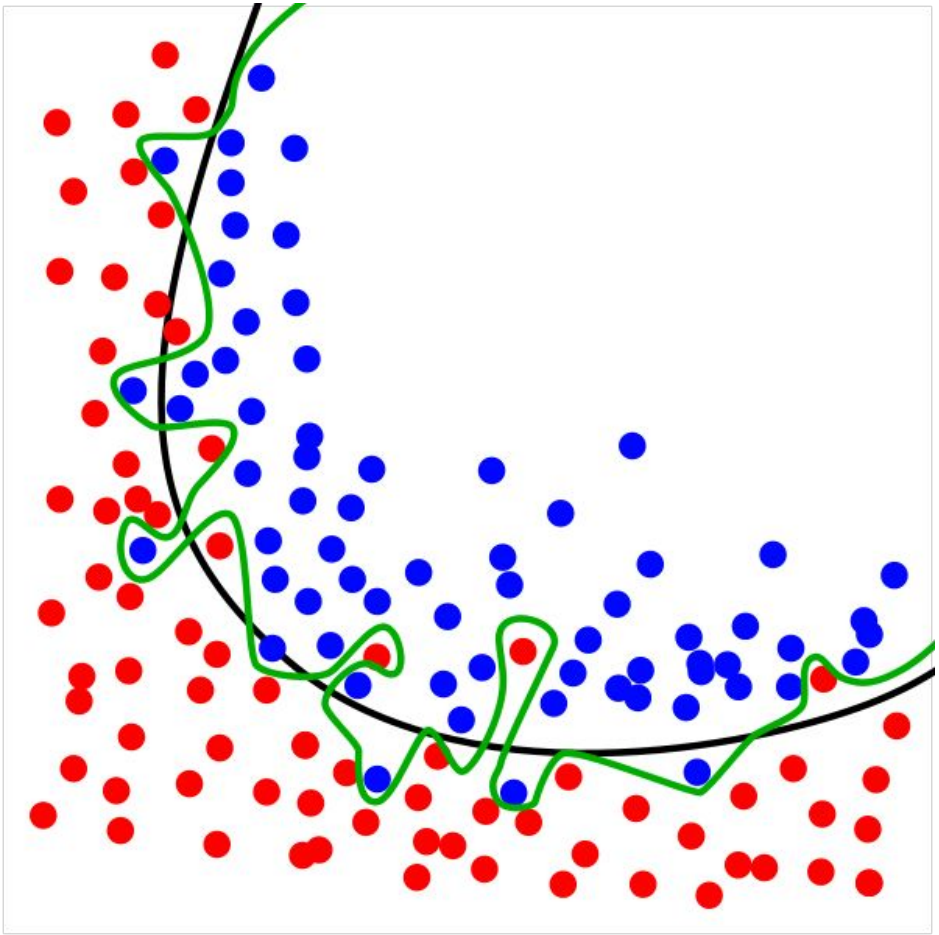


**tf.**relu6**(x, name=None)**

min(max(x, 0), 6)



tf.nn.dropout(Wx_plus_b, keep_prob)



# softmax

probability distribution

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \qquad \text{for } j = 1, \cdots, K.$$

$$P(y = j|\mathbf{x}) = \frac{e^{\mathbf{x}^\mathsf{T}\mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\mathsf{T}\mathbf{w}_k}}$$

# Activation Functions

特性：非线性+稀疏性

选择标准：试凑法

# cross_entropy 函数详解

它表示2个函数或概率分布的差异性:差异越大则相对熵越大，差异越小则相对熵越小，特别地，若2者相同则熵为0。

是两个概率分布P和Q差别的度量。使用基于Q的编码来编码来自P的样本平均所需的额外的位元数。

$$D_{\mathrm{KL}}(P\|Q) = \sum_i P(i)\,\log\frac{P(i)}{Q(i)}.$$

$$D_{\mathrm{KL}}(P\|Q) = \int_{-\infty}^{\infty} p(x)\,\log\frac{p(x)}{q(x)}\,\mathrm{d}x,$$

# 几个相关的概念

cross_entropy

maximum likelihood estimator

广义线性模型GLM(General Linear Model)

指数分布族（The exponential family）=大多数概率分布都可以表示成指数分布族形式

广义线性模型正是将指数分布族中的所有成员都作为线性模型的扩展，通过各种非线性的连接函数将线性函数映射到其他空间，从而大大扩大了线性模型可解决的问题

# Optimizer

class tf.train.GradientDescentOptimizer
class tf.train.AdadeltaOptimizer
class tf.train.AdagradOptimizer
class tf.train.AdagradDAOptimizer
class tf.train.MomentumOptimizer
class tf.train.AdamOptimizer
class tf.train.FtrlOptimizer
class tf.train.RMSPropOptimizer

http://sebastianruder.com/optimizing-gradient-descent/

# Gradient　Descent（exam1）

loss函数　$f(x)=x^4-3x^3+2$

梯度　　$f'(x)=4x^3-9x^2$

公式　　$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$$

# 代码

```
x_old = 0 ; x_new = 6 ; gamma = 0.01 ; precision = 0.00001

def df(x):
    y = 4 * x**3 - 9 * x**2
    return y

while abs(x_new - x_old) > precision:
    x_old = x_new
    x_new += -gamma * df(x_old)

print("The local minimum occurs at ", +x_new)
```

# 梯度下降法
# 多参数问题（exam2）

代码

可视化分析

# 太监吗？（务实和务虚的关系）

离开产品谈情怀 、谈理论、谈方法、谈梦想

好比**太监**谈高潮

# 学习资源

http://www.deeplearningbook.org/ 深度学习的斯坦福三位教授最新的书

http://sebastianruder.com/optimizing-gradient-descent/ 详细介绍优化算法