ML/DL常考知识点

笔记本:	机器学习		
创建时间:	2019/6/21 14:06	**更新时间:**	2020/5/30 11:28
作者:	beyourselfwb@163.com		
URL:	http://www.sohu.com/a/250971703_787107		

回答问题思路:
1、是什么?
2、适用场景?
3、核心原理?
4、跟其他对比?

常用的机器学习算法:
1、线性回归　（回归）　Lasso 回归、岭回归
2、逻辑回归　（分类）
3、决策树算法　（分类）　见下文
4、SVM 支持向量机　（分类）
5、NB 朴素贝叶斯　（分类）　见下文
6、RF 随机森林　（分类）

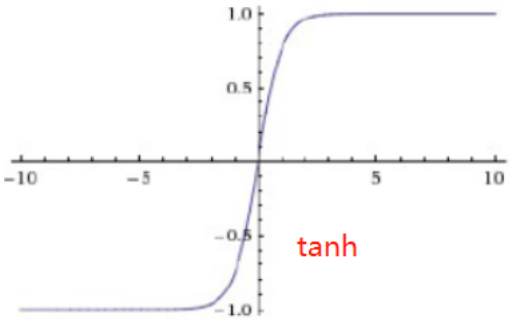
7、K-means　（聚类）

8、Gradient Boosting 和 AdaBoost算法

参考: https://blog.csdn.net/u012942818/article/details/74055224
https://github.com/imhuay/Algorithm_Interview_Notes-Chinese/blob/a274dcee72324519e043c639f254a8596a10b912/A-%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0/A-%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0%E7%AE%97%E6%B3%95.md#adab%E7%AE%97%E6%B3%95

8、Gradient Boosting 和 AdaBoost算法
AdaBoost是一种集成学习算法，以分类任务为例，基本思想是将多个分类器组合成一个强分类器。
两个核心点：（1）开始时，每个样本的权值是一样的，AdaBoost 的做法是提高上一轮弱分类器错误分类样本的权值，同时降低那些被正确分类样本的权值。（2）AdaBoost 采取加权表决的方法（加法模型）。具体的，AdaBoost 会加大分类误差率小的基学习器的权值，使其在表决中起到更大的作用，同时减小分类误差率大的基学习器的权值。

激活函数
常见的有：ReLU、sigmoid、tanh, softmax



tanh

Softmax	Sigmoid
$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$	$S(x) = \frac{1}{1+e^{-x}}$

sigmoid导数：y(1-y)

ReLu相比sigmoid的优势：
1、避免梯度消失
2、减缓过拟合
3、加速计算

BN(Batch Normalization)批标准化，主要作用：

- 加速网络的训练（缓解梯度消失，支持更大的学习率）
- 防止过拟合
- 降低了参数初始化的要求

LSTM面试：
https://blog.csdn.net/behboyhiex/article/details/81328510
https://www.jianshu.com/p/d6714b732927

朴素贝叶斯（生成模型）：
https://blog.csdn.net/jingyi130705008/article/details/79464740
http://www.sohu.com/a/250971703_787107
前提假设：相互独立；每个特征同等重要
优点：数据较少时任然有效，可以处理多分类
缺点：需要知道先验概率；对输入数据的表达形式很敏感

补充：
条件概率：P(X|Y) 表示在Y发生的条件下，X发生的概率
先验概率：一般是独立事件发生的概率，P(A)，P(B)，如P(垃圾邮件)，
特点----事件发生前预判概率（由历史数据统计或常识）

我收到一封邮件，在不进行浏览的条件下，根据经验我猜测20%的可能是垃圾邮件。

后验概率（反向条件概率）--- 即贝叶斯公式：

举例：10个男生，10个女生，夏天，男生打伞有1个，女生打伞有9个，

设事件打伞为 Y， 男生事件为 A， 女生为 B
那么 求一个女生打伞的概率（即在已知是女生的条件下，打伞的概率）：

$$P(\mathrm{Y}\mid\mathrm{B})=\frac{P(\mathrm{YB})}{P(\mathrm{B})}=\frac{\frac{9}{20}}{\frac{10}{20}}=\frac{9}{10}$$

这叫正向条件概率：

那么反向条件概率（贝叶斯公式）：已经打伞，求是女生的概率

$$p(\mathrm{B}=女\mid\mathrm{Y}=打伞)=\frac{p(\mathrm{Y}=打伞\mid\mathrm{B}=女)p(\mathrm{B}=女)}{p(\mathrm{Y}=打伞)}$$

由先验概率算出。

朴素贝叶斯的底层原理，比如说，如何选参数，如何训练模型，如何做分类?

贝叶斯分类实战:
二分类过程描述：遍历所有文档，得到三个概率向量---P(xi|C0)、P(xi|C1)、Pab(即侮辱性类别/总文档数)
来一份新的文档，分别计算 P(C0|w) 和 P(C1|w)的大小，概率大的为分类结果
p(c|w) = p(w|c) p(c) / p(w)
没有训练过程，只有统计的过程

问题1：计算概率乘积时可能会下溢出，（有点梯度消失那种意思）。
解决方案：取对数，ln(a * b) = ln(a) + ln(b)

决策树核心:
完整代码:
https://github.com/apachecn/AiLearning/blob/326edc5a5b207e66a06c2777a8c17d65dc3
https://gist.github.com/wbbeyourself/d53a3ea904921e02210d4457732c856d
（1）**信息熵**
参考 吴军 《数学之美》 P60页
信息量 单位bit ： 信息量就等于不确定性的多少
信息熵(Information Entropy) 单位bit：平均信息量（信息量的数学期望） 虽然同样是2MB的电子书，信息量也是不一样的，2MB只是平均值

信息熵数值大于0

变量的不确定性越大，熵也就越大，默认都是以2为底的对数函数。
计算参考代码:
https://gist.github.com/wbbeyourself/84bbc8ded8aedbb2c094491fe38b892c
代码解释:
https://github.com/apachecn/AiLearning/blob/master/docs/ml/3.%E5%86%B3%E7%AD%

$$H = -(p_1 \log p_1 + p_2 \log p_2 + \ldots + p_{32} \log p_{32})$$

$$H(X) = -\sum_{x \in X} P(x) \log P(x)$$

X：随机变量，{x1, x2, ···, x32}是 X 可能的取值；

$\sum P(x_i) = 1$ 概率之和等于 1

$-\log p(x_i)$ xi 的信息量

H(x)：信息熵，即信息量的平均值，信息量的期望

（2）信息增益(Information Gain)：选择最好的特征，划分数据集
如果按照特征i划分后的所有子数据集总体的熵的减少幅度最大（无序度减少），那么选择特征i的信息增益最大。
参考代码：
https://gist.github.com/wbbeyourself/460df44dc0cfa6516f434efe4f7eee58

$$IG = H(X) - \sum_{X' \in subX} p(X') H(X')$$

每次根据一个特征划分出子数据集（subX）的时候，总体信息熵会变小，所以 IG>0;
X'：一个划分出的子数据集;
P(X')：X'的数量在 subX 中占比
H(X')：子数据集 X'的信息熵

决策树的困难：
参考： https://www.cnblogs.com/hxyue/p/5841573.html
（1）属性值连续：分段离散二分
（2）样本值缺失：计算时不考虑那些样本缺失的数据（或者直接剔除）；确定划分的属性后，若在该属性上值缺失，如何确定样本归属？？
（3）多变量决策？？？

决策树的优化：剪枝

CBOW和skip-gram区别：
CBOW用周围的词预测中间的词，SG正好相反。
适用场景：都是用来预训练词向量的，后者用的更多些，因为1->n需要更强的预测能力
优化：取消隐藏层、使用分层softmax代替标准softmax，减少计算量
参考： https://www.cnblogs.com/pinard/p/7243513.html

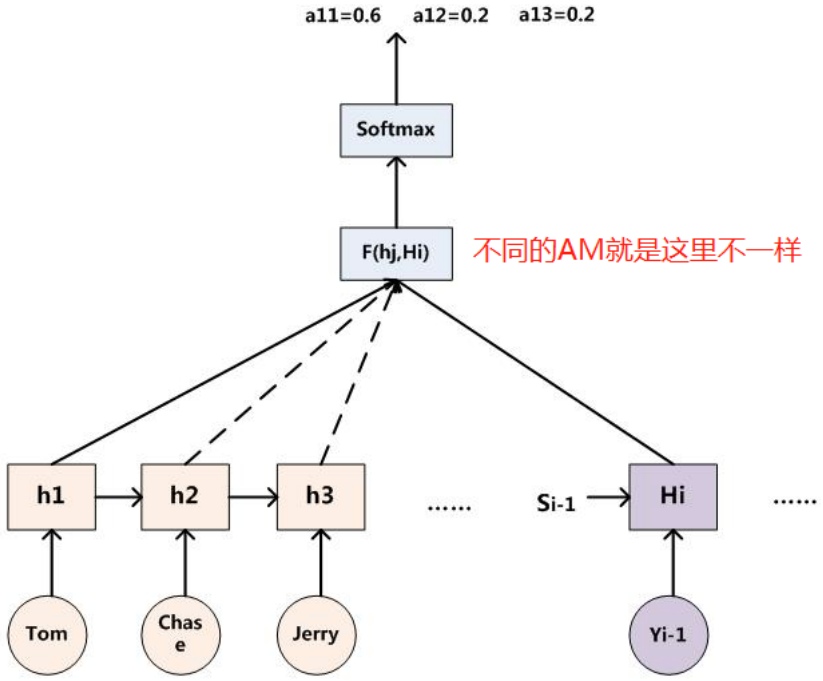
- 对比 N-gram 神经语言模型的网络结构
  - 【输入层】前者使用的是 `w` 的前 `n-1` 个词，后者使用 `w` 两边的词
    > 这是后者词向量的性能优于前者的主要原因
  - 【投影层】前者通过拼接，后者通过**累加求和**
  - 【隐藏层】后者无隐藏层
  - 【输出层】前者为线性结构，后者为树形结构
- 模型改进
  - 从对比中可以看出，CBOW 模型的主要改进都是为了**减少计算量**——取消隐藏层、使用**层Softmax**代替基本 Softmax

梯度消失：现象、原因、解决办法
三种解决办法：
（1）修改激活函数。
（2）用BN。
（3）把传统的循环神经网络，换成GRU网络。

CRF原理：
是什么：
应用场景：
核心原理：
对比：

fasttext 原理：
fasttext是一个快速文本分类算法，其模型架构和word2vec中的CBOW(连续词袋模型)，
但fasttext预测标签而CBOW预测的是中间词。
优化：N-gram(语序特征)、分层softmax(用霍夫曼树代替标准softmax，复杂度从N降低到logn)
参考： https://blog.csdn.net/qq_16633405/article/details/80578431

Attention Model关键问题：
(1) 注意力分配概率分布值是如何计算出来的，即I love you如何在翻译"爱"的时候，计算出"love"对应的系数更大的呢?
下面展示的就是常说的Soft Attention Model，也叫单词对齐模型，也可以理解为影响力模型。



上一篇讲AM模型的科普文介绍了Soft Attention Model，所谓Soft，意思是在求注意力分配概率分布的时候，对于输入句子X中任意一个单词都给出个概率，是个概率分布。那么相对Soft，就有相应的Hard Attention Model，提出Hard版本就是一种模型创新。既然Soft是给每个单词都赋予一个单词对齐概率，那么如果不这样做，直接从输入句子里面找到某个特定的单词，然后把目标句子单词和这个单词对齐，而其它输入句子中的单词硬性地认为对齐概率为0，这就是Hard Attention Model的思想。
参考： https://blog.csdn.net/malefactor/article/details/50583474

了解BERT吗?