# 1. 我的模型-自编码器

一、立项依据

本文旨在针对不平衡分类提出新的机器学习分类模型，传统的不平衡分类中通常有两种方法：数据层面和算法层面。但本文希望提出在不修改样本的情况下，针对不平衡分类提出比较好的方法。

二、国内外研究现状和发展趋势

目前针对不平衡分类的情况下，由于少数类样本的分类代价更高，所以产生了数据层面的过采样和欠采样方法，SMOTE[1]方法是典型的过采样方法，有学者则提出了遗传算法和采样结合的算法[2]，遗传算法不仅可以用于数据采样[2]、特征选择[3]，甚至可以作为集成算法里的分类器选择等方法。

1．国内外研究现状

2．研究现状存在的核心问题和本项目的研究设想

1）研究现状存在的核心问题

2）本项目的研究设想

本模型主要是希望直接针对样本进行分类，能够在比较原始的情况下对数据进行处理，希望无视不平衡状态而达到比较好的分类效果。

三、拟解决的关键科学问题和主要研究内容

1. 研究内容

采用自编码器对某类样本进行建模，取得重建模型，根据未知样本的重建误差进行投票区分类别，整个模型的优势在于不需要加入或者减少样本数量，并且是无监督学习。

2. 拟解决的关键问题

由于自编码器的特性和本模型的设计特性，可能在训练时容易造成过拟合，针对这个问题的解决方案目前有：

1. 加入dropout
2. 采用噪声自编码器[4]
3. 采用其他神经网络，比如DBN[5]
4. 加入不同的激活函数等抽取不同的特征和重建模型[6], [7]

综合目前的情况来看，本文在加入了噪声训练的情况下，没有同等条件下有监督微调的两层自编码器的分类效果好，目前原因不明，单独采用加权投票机制的效果并不明显，且未能找到有效的调参技巧。

## Tricks

属性值较多时，需要加入dropout，而属性值较少时，则需要加入noise。

用生成器产生样本时，不能超过生成器的训练集大小，否则生成样本质量会降低。

程序中生成的图片，横坐标为输入训练集的μ,纵坐标为σ，从测试结果中可以看出，当μ和σ位于（0,0）并且均匀分布时，能够得到比较好的测试结果。猜测原因是：采样过程中产生的μ和σ是（0,0）对称的，所以当采样和原始分布的概率重合时，能得到最佳的生成样本质量。

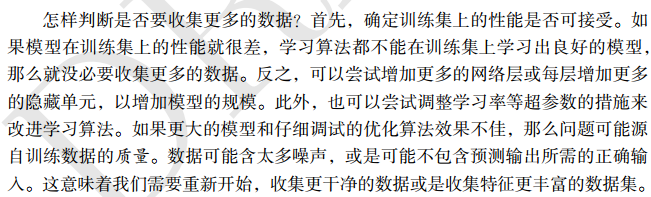
接下来的问题在于如何根据样本本身的分布情况，比如根据样本密度等，进行采样，以获得更好的生成样本。

答：这个问题尝试用改变生成样本的μ和σ，用训练样本的μ和σ的均值来代替生成的正态分布的参数，但从ionosphere数据集结果来看，效果还没有标准正态分布好。

### 问题多多？

对于重建分类器模型来说，目前的训练集效果也不是很好，很少有达到百分百的情况，针对此种情况，是否考虑采用有监督模型，需要试验过后进行确认。

答：针对模型的选择情况，在训练集上的效果好则证明模型有可取之处，如果训练集上的效果也很差的话，则没有考虑的必要。



**为何生成样本对别的分类器效果不好？**

**是否考虑使用GAN，由于这个是经过分类器进行验证的生成样本，因此有理由认为其生成样本的质量更加好。**

从实验结果本身来看，贝叶斯和本模型有相反的分类效果，基本处于一个高而另一个低的情况，所以是否能考虑用集成方法进行学习？

我需要改进一下我的SDAE，使其成为真正的SDAE。

答：已经完成。但无fine-tune

Yeast数据集的隐层分布明显不是呈正态分布状，本方法只针对能够形成正态分布的数据集有比较好的生成效果，因为vae的隐层设置假设其为正态分布，训练集F1值最高能到0.9，但是针对别的小型数据集，效果都不是很好。针对这种问题，我觉得可以考虑用传统方法来进行处理，比如adaboost或者bagging之类的，对于神经网络可能不适用。

针对样本的数量和特征数，对模型的大小和层数进行进一步的微调，对于正类样本较少的情况下，主动减少模型层数和隐层节点数，观察实验效果。

答：对模型的激活函数进行了实验验证，对于各个数据集来说，饱和类的激活函数能得到相对比较稳定的性能，但是不一定是最好的。对于正类样本减少模型层数和隐层节点数（减少模型容量）有助于提升模型性能。

### 针对vae的生成样本质量提出更好的方法，这期间需要看文献来解决，之后考虑GAN，再考虑有监督。或者可以换下顺序，主要还是生成样本的质量问题。

答：如何评价生成样本质量，1.GAN的思路是能够欺骗分类器，我们的想法是希望更加真实的生成样本可以提高分类器模型的效果，并且真实分布和生成数据分布尽量拟合（这个可以参考KL散度实现）。

文献[8]分析表明，GAN能够生成比较真实的样本，其原因不在于针对数据和模型的概率分布比率的散度进行优化，而在于其对抗过程，而vae中则是采用了两个分布的低维潜在变量的下界来进行优化，因此生成的样本可能没有GAN生成的样本具有典型性，而目前则产生了对抗性的vae等（目前未涉及）

分类器的训练误差有时会比测试误差还大，交叉验证效果不稳定，生成样本分布无法控制。

交叉验证结果不稳定：一方面是因为神经网络的训练本身是有多局部极小值的，另一方面则是因为过采样本身是一个上限很高的算法。

## 接下来的计划

1. 模型本身的改进方法

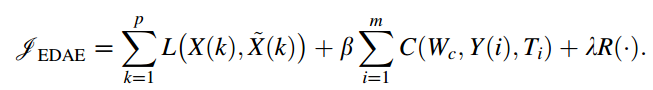
调整模型的层数，将其改进为可调节层数的噪声叠加自编码器，并能输入隐层节点等。

答：该要求已经完成，能输入隐层节点数，可调节层数，该SDAE采用的是随机抽取batch，但缺少整体微调。修改成顺序获取batch并且加入微调

1. 模型的loss函数优化

针对模型的任务相关性不强的问题，本文[9]提出在AE的训练loss中加入有监督分量，即在模型中添加一个softmax层，并在训练中根据其分类结果，调整自编码器的参数。

传统的训练过程是逐层训练，最后添加一个softmax层，而本文[9]则在训练过程中直接添加了softmax层，并且将其分类结果直接添加入loss函数。





1. 关于生成少数类样本的方法

[GAN的一些知乎简介](https://www.zhihu.com/question/52602529/answer/155743699) [关于GAN的基本简介](https://www.leiphone.com/news/201704/fcG0rTSZWqgI31eY.html) [变分自编码器](http://blog.csdn.net/jackytintin/article/details/53641885)  [GAN的另一个简介](https://www.leiphone.com/news/201701/Kq6FvnjgbKK8Lh8N.html)

[GAN的入门代码](https://www.leiphone.com/news/201704/b8w2VNuvTV2CERMP.html)

变分自编码器 利用神经网络生成样本，假设样本的最后表现形式由某些潜在的变量决定，设其为z，并对其先验分布和后验分布用神经网络进行模拟。

最开始z假设要求为高斯分布，假设较强，在[10]中针对该种问题，提出了用模糊假设的方式模拟z的分布，即对z的分布采用参数化方法，而不是简单的高斯分布，结果显示，针对一些问题，能够获得比原始的vae更好的结果。

1. 对多数类进行下采样，去掉离群点可能会有助于提高分类器的准确度，但对提高正类样本准确度没有帮助，主动下采样方法，消除远离决策面的负类样本[11]。
2. 有监督模型

本文[12]中采用了SMOTE+有监督自编码器的架构，首先对数据采取SMOTE方式进行过采样至数据集平衡，后采用AE对数据进行重建，最后加入softmax层进行分类。结果显示该模型对提高不平衡分类效果有一定的好处，并在UCI数据集上的auc值取得了比较高的测试结果。

针对模型的任务相关性不强的问题，本文[9]提出在AE的训练loss中加入有监督分量，即在模型中添加一个softmax层，并在训练中根据其分类结果，调整自编码器的参数。传统的训练过程是逐层训练，最后添加一个softmax层，而本文[9]则在训练过程中直接添加了softmax层，并且将其分类结果直接添加入loss函数。

## 如何查找数据

[各类数据库查询网站](https://sanwen8.cn/p/4a7Ms1J.html) [科研数据吧](https://tieba.baidu.com/f?kw=%E7%A7%91%E7%A0%94%E6%95%B0%E6%8D%AE)

<http://www.gems-system.org/> lung cancer, SRBCT, 可用

[Colon cancer 下载地址](http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html) 可用

[常用数据库下载地址](http://www.xmarks.com/site/similar/2/datam.i2r.a-star.edu.sg/datasets/krbd/?created=all)

[肯特岗数据集](http://leo.ugr.es/elvira/DBCRepository/) [另一个数据集下载处](http://featureselection.asu.edu/datasets.php)

根据论文中给出的链接、名字（名字不是很准，可以多换一些试试、缩写等 或者多添加一些关键字dataset等）、样本数目和特征数目（加上会提高查询效率）、

## 概率密度函数等

[一些概率密度函数和定义等](http://www.itl.nist.gov/div898/handbook/eda/section3/eda362.htm)

## 期刊顶会

期刊

PAMI

PR

Neural network

Bioinformatics

Plos one

顶会

AAAI 人工智能领域

CVPR 机器视觉和模式识别

ACL 自然语言处理

NIPS 深度学习

ICLR 深度学习

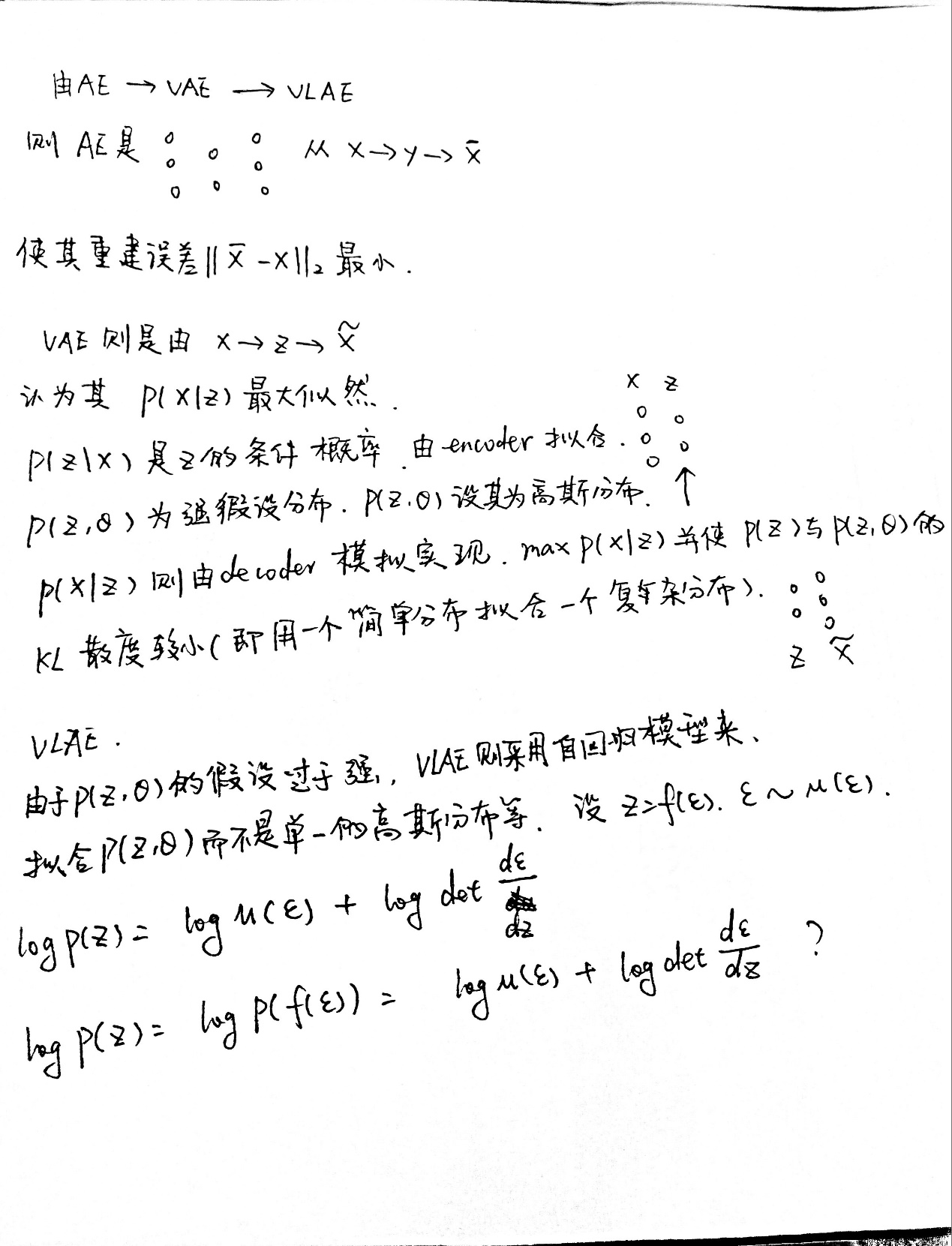
## 一些资料

[lecun关于无监督学习的蛋糕](https://mp.weixin.qq.com/s?__biz=MzI3MTA0MTk1MA==&mid=2652000161&idx=1&sn=2b44f2590464b13a9cbccb7f2544d12f&chksm=f1213b50c656b246d24a9dac15fb49d292a703ce0c1fbbffc91cf69eb504914eb9082bc38f7f&mpshare=1&scene=23&srcid=0701jr3AWJ2ExrJZKIOyJT1G#rd) （概述了目前深度学习的应用原因，主要在于可以层次性的表达物体特征，并对GAN的可行性提出了很大的赞许。）

[panda等数据可视化工具](http://python.jobbole.com/87136/)

闲着无聊可以搜一搜“堪称神器”关键字，有挺多东西挺好玩的。

[vae编码](http://blog.csdn.net/u014749291/article/details/56675805)



[人际交往中的期望值管理能力、阈值自控意识、应对主观时空扭曲的能力](http://www.xinli001.com/info/100370347?from=shouye)

英语写作咨询1308607911 QQ

## 参考文献

[1] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2011.

[2] S. García and F. Herrera, “Evolutionary Undersampling for Classification with Imbalanced Datasets: Proposals and Taxonomy,” *Evolutionary Computation*, vol. 17, no. 3, pp. 275–306, 1993.

[3] M. Yousef, M. D. Saçar Demirci, W. Khalifa, and J. Allmer, “Feature Selection Has a Large Impact on One-Class Classification Accuracy for MicroRNAs in Plants,” *Advances in Bioinformatics*, vol. 2016, pp. 1–6, 2016.

[4] M. M. A. Rahhal, Y. Bazi, H. Alhichri, N. Alajlan, F. Melgani, and R. R. Yager, “Deep learning approach for active classification of electrocardiogram signals,” *Information Sciences*, vol. 345, no. 1, pp. 340–354, 2016.

[5] M. Hayat, M. Bennamoun, and S. An, “Learning Non-linear Reconstruction Models for Image Set Classification,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 37, no. 4, pp. 713–727, 2015.

[6] W. W. Y. Ng, G. Zeng, J. Zhang, D. S. Yeung, and W. Pedrycz, “Dual autoencoders features for imbalance classification problem,” *Pattern Recognition*, vol. 60, pp. 875–889, 2016.

[7] 王坤峰, 苟超, 段艳杰, 林懿伦, 郑心湖, and 王飞跃, “生成式对抗网络GAN的研究进展与展望,” 自动化学报, vol. 43, no. 3, 2017.

[8] I. Goodfellow, “NIPS 2016 Tutorial: Generative Adversarial Networks,” 2016.

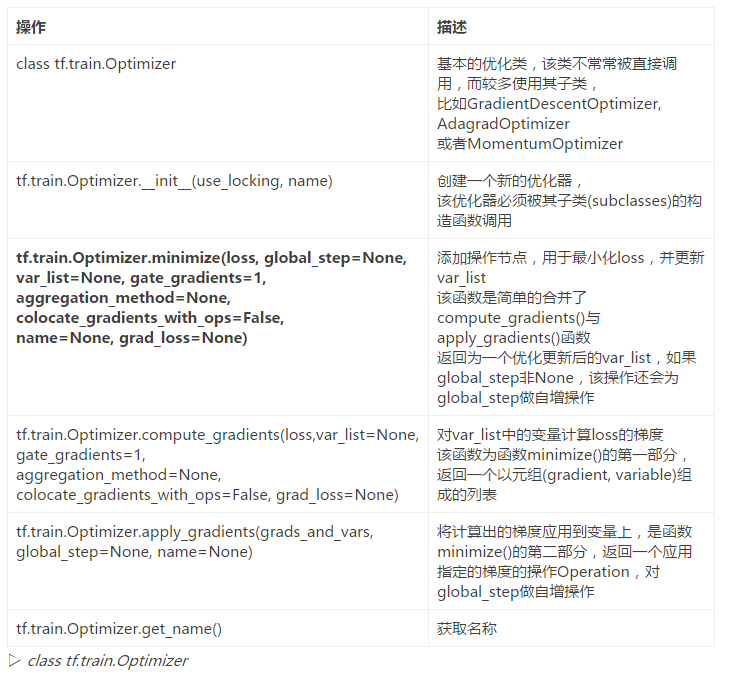
[9] Y. Sun, H. Mao, Y. Sang, and Z. Yi, “Explicit guiding auto-encoders for learning meaningful representation,” *Neural Computing & Applications*, vol. 28, no. 3, pp. 429–436, 2017.

[10] X. Chen *et al.*, “Variational Lossy Autoencoder,” 2016.

[11] 杨泽平, “基于神经网络的不平衡数据分类方法研究,” 华东理工大学, 2015.

[12] 张成刚, 宋佳智, 姜静清, and 裴志利, “一种改进的降噪自编码神经网络不平衡数据分类算法,” 计算机应用研究, no. 5, pp. 1329–1332, 2017.

## 优化器算法



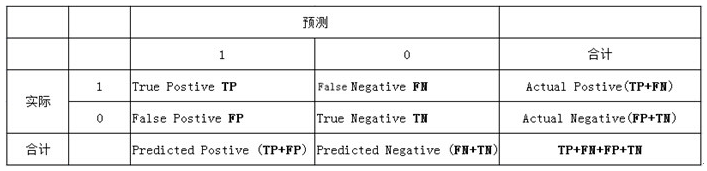
| **操作** | **描述** |
| --- | --- |
| **class tf.train.GradientDescentOptimizer** | 使用梯度下降算法的Optimizer |
| tf.train.GradientDescentOptimizer.\_\_init\_\_(learning\_rate,  use\_locking=False, name=’GradientDescent’) | 构建一个新的梯度下降优化器(Optimizer) |
| **class tf.train.AdadeltaOptimizer** | 使用[Adadelta算法](http://arxiv.org/abs/1212.5701)的Optimizer |
| tf.train.AdadeltaOptimizer.\_\_init\_\_(learning\_rate=0.001,  rho=0.95, epsilon=1e-08,  use\_locking=False, name=’Adadelta’) | 创建Adadelta优化器 |
| **class tf.train.AdagradOptimizer** | 使用[Adagrad算法](http://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf)的Optimizer |
| tf.train.AdagradOptimizer.\_\_init\_\_(learning\_rate,  initial\_accumulator\_value=0.1,  use\_locking=False, name=’Adagrad’) | 创建Adagrad优化器 |
| class tf.train.MomentumOptimizer | 使用[Momentum](http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf)算法的Optimizer |
| tf.train.MomentumOptimizer.\_\_init\_\_(learning\_rate,  momentum, use\_locking=False,  name=’Momentum’, use\_nesterov=False) | 创建momentum优化器 momentum：动量，一个tensor或者浮点值 |
| **class tf.train.AdamOptimizer** | 使用[Adam 算法](http://arxiv.org/abs/1412.6980)的Optimizer  （推荐使用） |
| tf.train.AdamOptimizer.\_\_init\_\_(learning\_rate=0.001, beta1=0.9, beta2=0.999, epsilon=1e-08, use\_locking=False, name=’Adam’) | 创建Adam优化器 |
| class tf.train.FtrlOptimizer | 使用[FTRL 算法](https://www.eecs.tufts.edu/~dsculley/papers/ad-click-prediction.pdf)的Optimizer |
| tf.train.FtrlOptimizer.\_\_init\_\_(learning\_rate,  learning\_rate\_power=-0.5,  initial\_accumulator\_value=0.1,  l1\_regularization\_strength=0.0,  l2\_regularization\_strength=0.0, use\_locking=False, name=’Ftrl’) | 创建FTRL算法优化器 |
| class tf.train.RMSPropOptimizer | 使用[RMSProp算法](http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf)的Optimizer |
| tf.train.RMSPropOptimizer.\_\_init\_\_(learning\_rate,  decay=0.9, momentum=0.0, epsilon=1e-10,  use\_locking=False, name=’RMSProp’) | 创建RMSProp算法优化器 |

具体描述参见这篇：

[基本优化算法对比](http://blog.csdn.net/u012759136/article/details/52302426) [TensorFlow具体实现](http://blog.csdn.net/lenbow/article/details/52218551)

## 分类器评价准则

1. 混淆矩阵（confusion matrix）



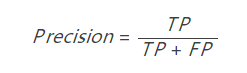
(1)真正类率(True Positive Rate , TPR)【灵敏度(sensitivity)】：TPR = TP /(TP + FN) ，即正样本预测结果数/正样本实际数

(2)假负类率(False Negative Rate , FNR) ：FNR = FN /(TP + FN) ，即被预测为负的正样本结果数/正样本实际数

(3)假正类率(False Positive Rate , FPR) ：FPR = FP /(FP + TN) ，即被预测为正的负样本结果数/负样本实际数

(4)真负类率(True Negative Rate , TNR)【特指度(specificity)】：TNR = TN /(TN + FP) ，即负样本预测结果数/负样本实际数

Recall  表明分类器对正类的区分能力

Precision  表明模型对负类样本的区分能力

F1值则表明了两类的分类能力，且相对比较均衡

ROC 曲线

当测试集中的正负样本的分布变化的时候，ROC曲线能够保持不变。

四、 总体研究方案

1. 学术思路

2. 技术途径

3. 可行性分析

1）学术思想可行性

a、问题提出的可行性：

b、解决问题方案的可行性：

## 技术研究类项目申请

1. 立项依据

目的和意义（类似文献中的背景）

阐述目前存在的问题，以及其他学者解决了部分问题，但还存在着别的问题等，对问题进行总结，并对其意义进行肯定

1. 国内外研究现状（针对题目所提到的方法进行阐述，即解决问题的方法）

比如如果我要写基于深度学习的不平衡样本分类，则应该针对不平衡样本分类方法进行总结和阐述，并标明深度学习用于该问题的解决方案。

不平衡分类大问题：分成集成、代价敏感、深度学习等

总结

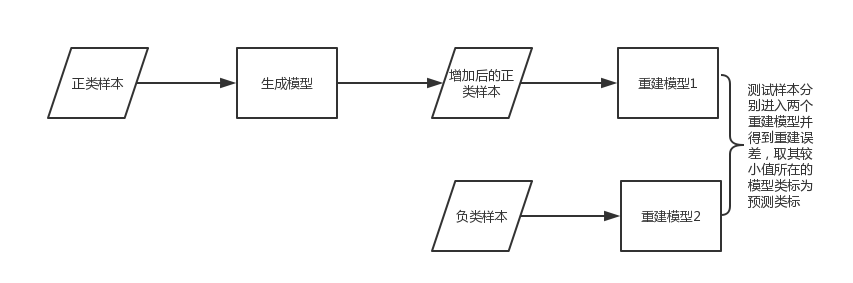
1. 核心问题和研究设想

目前的方法中存在的问题等（没有较好的普适性、生成算法仅仅是数据层面进行简单的加权计算SMOTE等）

利用生成样本提高分类性能包含两个方面：生成样本和提高性能

1. 学术思路和技术途径

其中学术思路比较宽泛，而技术途径则需要细节的实现，比如基本框架及实现细节等，比如GAN和vae等

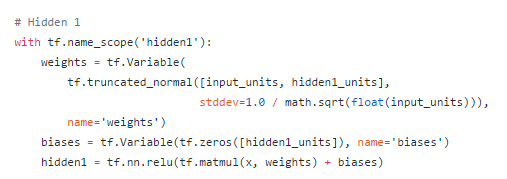


# 2.TensorFlow技术

## 1. 关于命名空间的问题

[关于TensorFlow的的命名空间的问题](http://blog.csdn.net/lqfarmer/article/details/69566182)

在TensorFlow中，由于存在并行和共享数据的问题，因此采用了命名空间的结构，这样可以把变量限制在某个空间下，不同的空间可以用相同的变量名等



这个只是解释了如何在不同的层中采用相同的变量名称，但是还不知道如何重复使用代码。

2.关于TensorFlow如何使用

GitHub上的很多开源项目，代码较标准

[TensorFlow\_examples](https://github.com/tobegit3hub/tensorflow_examples/blob/master/mlp.py)

[TensorFlow核心使用技巧](https://github.com/tobegit3hub/deep_recommend_system)

[TensorFlow的变量共享和复用](http://blog.csdn.net/xwd18280820053/article/details/70808583)

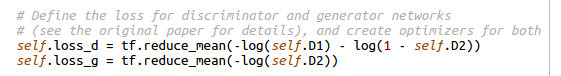
tf.name\_scope：

tf.variable\_scope: 通常和tf.get\_variable 一起使用

tf.variable: 运用该函数会产生新变量，且不允许同一个名字定义两次，即程序不能运行两次。

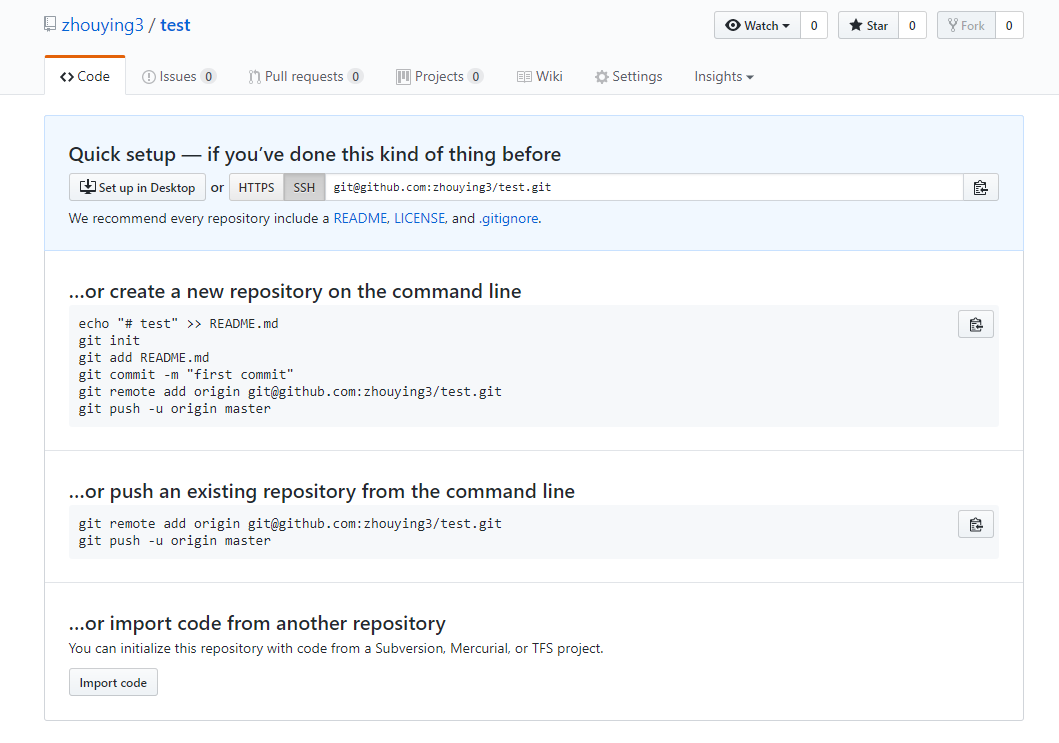
tf.get\_variable: 这个函数相当于从变量中读取数据，不会产生新数据，且能重复使用。

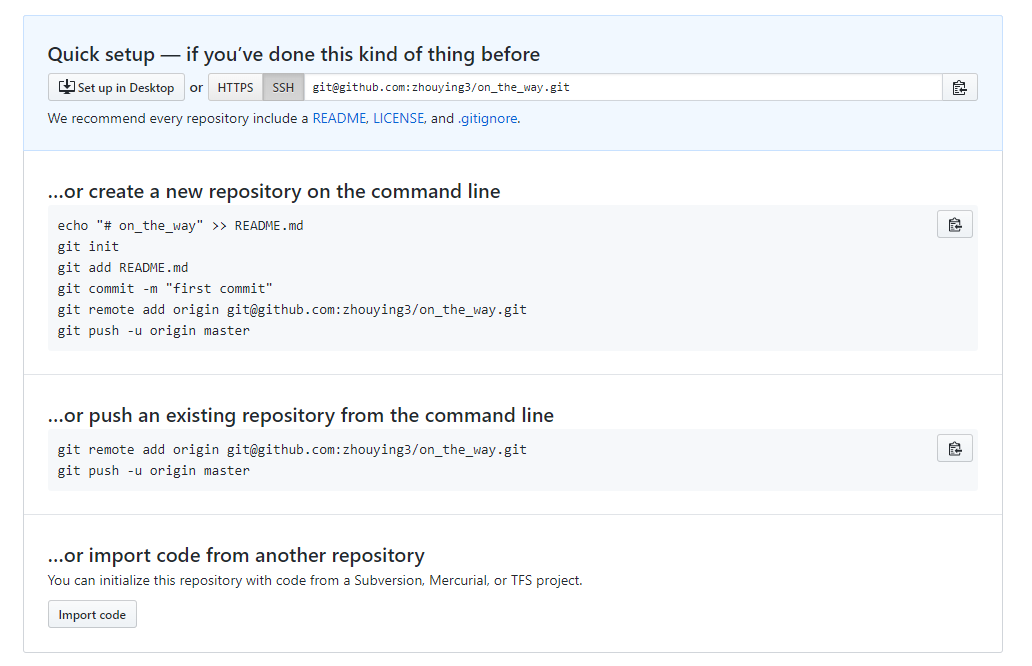
如果想要达到重复利用变量的效果, 我们就要使用 tf.variable\_scope(), 并搭配 tf.get\_variable() 这种方式产生和提取变量. 不像 tf.Variable() 每次都会产生新的变量, tf.get\_variable() 如果遇到了同样名字的变量时, 它会单纯的提取这个同样名字的变量(避免产生新变量). 而在重复使用的时候, 一定要在代码中强调 scope.reuse\_variables(), 否则系统将会报错, 以为你只是单纯的不小心重复使用到了一个变量.

GAN基本架构：训练D 网络结构：单隐层mlp loss 

训练G 网络结构：单隐层mlp loss

# 关于GitHub





git@github.com:zhouying3/on\_the\_way.git