Draft Version

MACHINE LEARNING YEARNING

Technical Strategy for AI Engineers, In the Era of Deep Learning

ANDREW NG

Machine Learning Yearning is a
deeplearning.ai project.

偏差和方差

20 偏差和方差

假设你的训练集，开发集和测试集都来自同一分布。那么你会觉得获取更多的训练数据就可以提高性能，对吗？

尽管更多的数据是无害的，但它并不是总会像我们所期望的那样有用。获取更多的数据需要耗费很多时间。所以，你需要什么什么时候该增加数据量，什么时候不该增加。

机器学习中有两个主要的错误来源：偏差和方差。理解它们有助于你觉得是否添加数据，以及其它提高性能的策略，这将会很好的利用你的时间。

假设你正在构建一个错误率为 5%的猫咪识别器。目前，你的训练集错误率为 15%，并且你的开发集错误率为 16%，在这种情况下，添加数据可能不会有太大的帮助。你应该关注其它的办法。实际上，在你的训练集上添加更多的样本只会让你的算法难以在训练集上做的更好。（后面的章节我会解释原因）

如果你在训练集上的错误率为 15%（85%的准确率），但是你的目标是 5%的错误率（95%的准确率），那么第一个要解决的问题是提高算法在训练集上的性能。你的开发/测试集上的性能通常比训练集差。所以，如果算法在见过的样本上得到了 85%的准确率，那么是不可能在没见过的样本上得到 95%的准确率的。

假设如上述你的算法在开发集上有 16%的错误率（84%的准确率）。我们将这 16%的错误分为两部分：

- 首先，算法在训练集上的错误率。在本例中，它是 15%。我们非正式的认为这是算法的**偏差(bias)**。
- 其次，算法在开发（或测试）集上比训练集差多少。在本例中，开发集比训练集差 1%。我们非正式的认为这是算法的**方差(Variance)**[1]。

学习算法的一些改变能解决错误的第一个组成部分——偏差，并且提高算法在训练集上的性能；一些改变能解决第二个组成部分——方差，并帮助算法从训练集到开发/测试集上得到更好的泛化[2] 。为了选择最有希望的改变，了解这两组错误中哪个更值得去解决是非常有用的。
培养你对于偏差和方差的感觉可以帮你在优化算法上有非常大的帮助。

1 统计领域有更多关于偏差和方差的正式定义，我们不必担心。粗略地说，当你有一个非常大的训练集时，偏差就是你算法在训练集上的错误率。方差是与此设置中的训练集相比，你在测试集上差多少。当你的误差衡量是均方差（mean squared error）时，你可以写下指定这两个量的公式，然后证明：Total Error = Bais + Variance。但是为了决定如何在 ML 问题上取得进展，这里了解偏差与方差即可。
2 这里还有一些通过对系统架构做出大的改变的方法，能够同时减少偏差和方差。但是这些方法做起来比较难。

21 关于偏差和方差的实例

思考我们的猫咪分类器。一个理想的分类器（如：人为分类）会在这个任务中有着完美的表现。

假设你的算法表现如下：

- 训练集上的误差 = 1%
- 开发集上的误差 = 11%

这里存在什么问题呢？根据前面章节的定义我们估计它的偏差为 1%，方差为 10%(=11%-1%)。因此，它的方差很大，分类器在训练集上的错误率很低，但是它不能很好的泛化到开发集上，这被称为**过拟合(overfitting)**。

现在我们思考下面这个情况:

- 训练集上的误差 = 15%
- 开发集上的误差 = 16%

我们估计偏差为 15%，方差为 1%。这个分类器适用于训练集，虽然存在 15%的误差，但是它在开发集上的误差比在训练集上的误差要高。这个分类器存在着**高偏差，低方差**，我们认为这个算法是**欠拟合(underfitting)**。

现在我们再思考这种情况:

- 训练集上的误差 = 15%
- 开发集上的误差 = 30%

我们估计偏差为 15%，方差也为 15%。这个分类器**同时具有高偏差和高方差**：它在训练集上的效果不是很好，因此产生了高偏差，然后它在开发集上的效果表现的更差，因此出现高方差，所以这个分类器是同时过拟合/欠拟合的，难以用术语表示。

最后思考这种情况:

- 训练集上的误差 = 0.5%
- 开发集上的误差 = 1%

这个分类器的效果就很好，它的偏差和方差都很低。如果是这种情况，祝贺你！

22 对比最优误差率

在我们的猫咪识别实例中，这个“想法”的错误率指的是——最优分类器的错误率接近 0%，就像一个人可以很轻松的识别它。而且随时可以进行识别，我们希望机器也可以做到这点。

还有一些问题是比较困难的。例如：假设你建立了一个语音识别系统，并且发现有 14%的音频杂音非常多，即使一个人也很难听出音频中在说什么。在这种情况下，这个“最优的”语音识别系统的误差大约为 14%。

假设在这个语音识别系统中，你的算法效果如下：

- 在训练集上的误差 = 15%
- 在开发集上的误差 = 30%

在训练集上的效果接君最优误差 14%。因此，在偏差和训练集上面进行改进是不会取得太大的效果的。然而这个算法并不适用于开发集；因此，由于方差的原因，在这里有很大的改进空间。

这个例子于上一章节的第三个例子类似，它有在训练集上有 15%的误差，在开发集上有 30%的误差。如果最优分类器的误差接近于 0%的话，则训练集上有 15%的误差改进空间非常大，减少偏差是非常有效的。但是如果最优错误率约为 14%，那么近乎相同的训练集的数据告诉我们我们分类器是很难提高的。

对于最优错误率远大于 0%的问题，这里有一个关于算法错误的更详细的分类。我们继续使用上面的语音识别示例，可以按如下方式分解在开发集上的 30%误差。（在测试集上也可以类似进行错误分析）

- **最优误差率 (“不可避免的偏差”)**: 14%. 假设我们认为，即使世界上最好的语言我们仍会有 14%的误差，我们可以把这个看作为不可避免的部分。
- **可避免的偏差**：1%.由训练集上的误差于最优误差的差值计算得到。[3]
- **方差**：15%.训练集与开发集上误差的区别。

由我们之前的定义，我们定义这两者关系如下：[4]

偏差 = 最优误差（不可避免的偏差） + 可避免的偏差

这个可避免的偏差反映了你算法的在训练集上与最优分类器直接的差别。

方差的定义与之前的定义一样，从理论上讲，我们可以通过对大量训练集的训练，将方差减少到接近 0%的水平。因此，如果数据量足够大，所有的方差都是可避免的，反之不可避免。

再思考一个例子：假设最佳错误率为 14%，我们有如下：

- 在训练集上的误差 = 15%
- 在开发集上的误差 = 16%

在前一章，我们把它叫做高偏差分类器，我们现在可以知道，可避免的误差为 1%，误差在 1%左右，因此，这个算法已经做的很好了，几乎没有改进空间，它与最佳错误率相差不超过 2%。

我们从这些例子中可以看出，知道这个误差率有助于指导我们进行下一步工作。在统计中，最优错误率也称为**贝叶斯误差率（Bayes error rate）**或贝叶斯率。

我们怎么知道最优错误率是多少？对于人类擅长的任务，例如识别图片或视频剪辑，人类可以给结果打标签，然后计算出准确率。这里将给出最优错误率的一个评估准则。如果你在做一个人类做起来比较困难的事情（如：预测推荐电影，或者广告给一个用户）这就很难给出最优的错误率。

在“与人类性能比较[Comparing to Human-Level Performance]（33-35 章节）”中，我们将详细讨论它。将学习算法的性能与人类的水平进行比较。

在最后的几章中，你会了解如何通过查看训练集和开发集的错误率来评估可避免/不可避免的偏差和方差。下一章将讨论如何利用这样的分析来区分偏差和一些其它的减小偏差的技术。根据你当前项目的情况，是高偏差（可避免的）还是高方差，你应该使用不同的优化方法。加油！

---

[3] 如果是负数，那么在训练集上你做得比最佳错误率更好。这意味着你过拟合了，此时你应该关注如何减少方差，而不是进一步的减少偏差。

[4] 这些定义是用于你算法改进的一些见解。这不同于统计学上定义的偏差和方差。从技术上讲，我们这里定义的偏差为，我在这里定义的“我们认为存在的错误”；而“可避免的偏差”应该是我们认为算法的错误率大于最佳错误率。