Course 3：结构化机器学习项目（Structuring Machine Learning Projects）

3.1 ML strategy

ML strategy

当最初得到一个深度神经网络模型时，希望从很多方面来对它进行优化，例如：
Collect more data
Collect more diverse training set
Train algorithm longer with gradient descent
Try Adam instead of gradient descent
Try bigger network
Try smaller network
Try dropout
Add L2 regularization
Network architecture: Activation functions, #hidden units…

Orthogonalization

Orthogonalization 的核心在于每次调试一个参数只会影响模型的某一个性能
机器学习监督式学习模型大致分成四个独立的“功能”：

- Fit training set well on cost function ，优化训练集可以通过使用更复杂 NN，使用 Adam 等优化算法来实现
- Fit dev set well on cost function，优化验证集可以通过正则化，采用更多训练样本来实现
- Fit test set well on cost function，优化测试集可以通过使用更多的验证集样本来实现
- Performs well in real world，提升实际应用模型可以通过更换验证集，使用新的 cost function 来实现

每一种“功能”对应不同的调节方法，是正交的
early stopping 在模型功能调试中并不推荐使用。因为 early stopping 在提升验证集性能的同时降低了训练集的性能。即 early stopping 同时影响两个“功能”，不具有独立性、正交性

3.2 Setting Up your Goal

Single number evaluation metric

准确率与召回率（Precision & Recall）
准确率和召回率是广泛用于信息检索和统计学分类领域的两个度量值,用来评价结果的质量。其中精度是检索出相关文档数与检索出的文档总数的比率，衡量的是检索系统的查准率；召

回率是指检索出的相关文档数和文档库中所有的相关文档数的比率，衡量的是检索系统的查全率。
一般来说，Precision 就是检索出来的条目（比如：文档、网页等）有多少是准确的，Recall 就是所有准确的条目有多少被检索出来了。
正确率、召回率和 F 值是在鱼龙混杂的环境中，选出目标的重要评价指标。不妨看看这些指标的定义先：

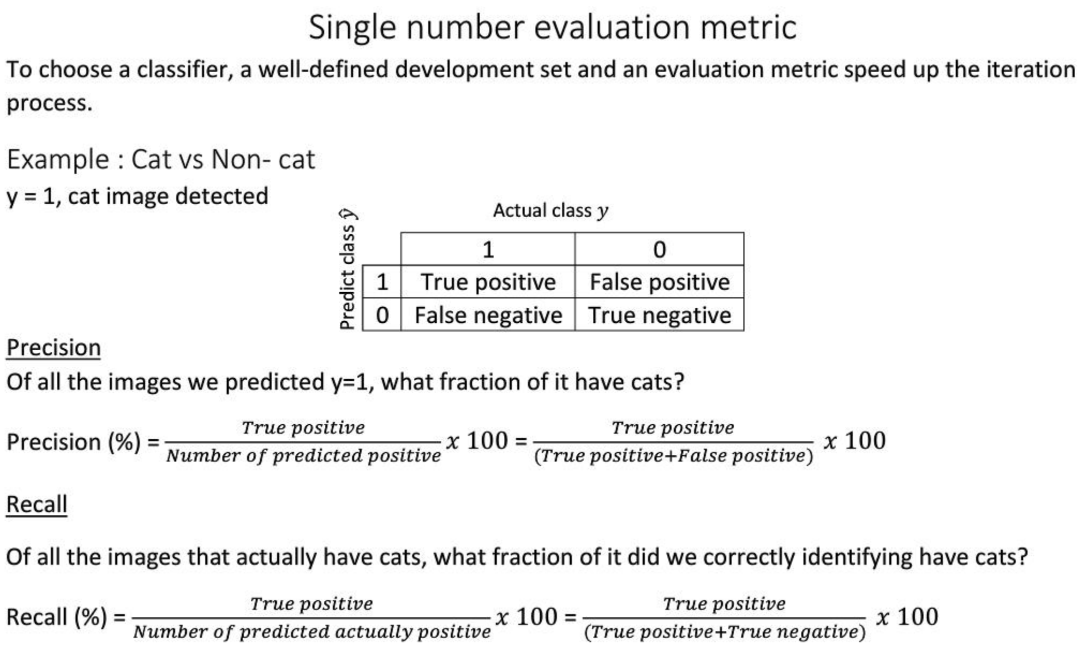
1. 正确率 = 提取出的正确信息条数 / 提取出的信息条数
2. 召回率 = 提取出的正确信息条数 / 样本中的信息条数

两者取值在 0 和 1 之间，数值越接近 1，查准率或查全率就越高。

3. F 值 = 正确率 * 召回率 * 2 / (正确率 + 召回率) （F 值即为正确率和召回率的调和平均值）

不妨举这样一个例子：某池塘有 1400 条鲤鱼，300 只虾，300 只鳖。现在以捕鲤鱼为目的。撒一大网，逮着了 700 条鲤鱼，200 只虾，100 只鳖。那么，这些指标分别如下：
正确率 = 700 / (700 + 200 + 100) = 70%
召回率 = 700 / 1400 = 50%
F 值 = 70% * 50% * 2 / (70% + 50%) = 58.3%
不妨看看如果把池子里的所有的鲤鱼、虾和鳖都一网打尽，这些指标又有何变化：
正确率 = 1400 / (1400 + 300 + 300) = 70%
召回率 = 1400 / 1400 = 100%

F 值 = 70% * 100% * 2 / (70% + 100%) = 82.35%



Single number evaluation metric

To choose a classifier, a well-defined development set and an evaluation metric speed up the iteration process.

Example : Cat vs Non- cat

y = 1, cat image detected

		Actual class y	
Predict class $\hat{y}$		1	0
	1	True positive	False positive
	0	False negative	True negative

<u>Precision</u>

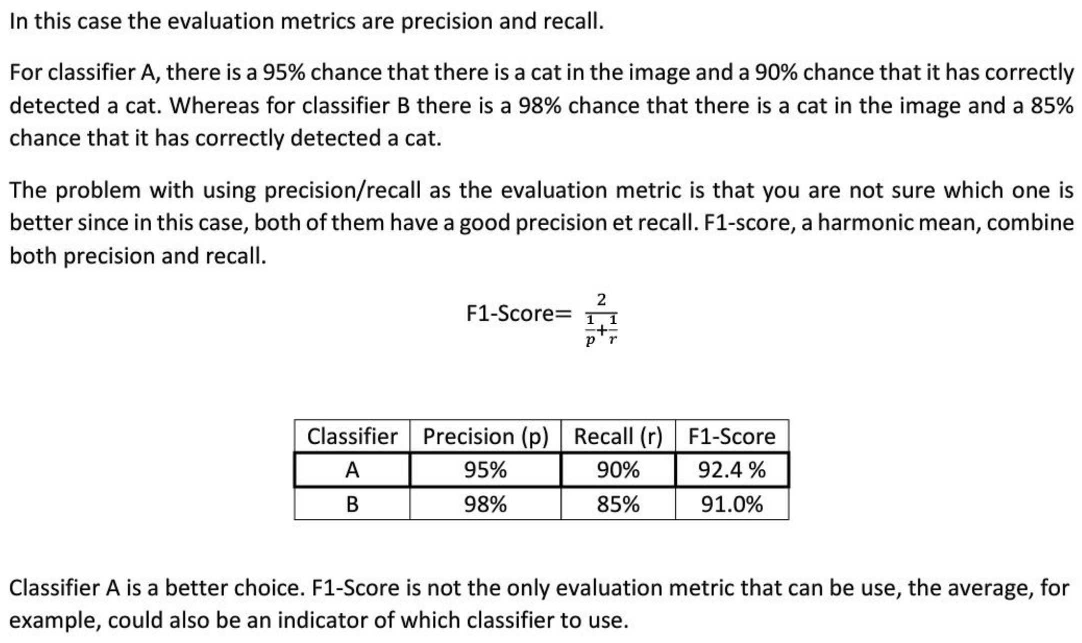
Of all the images we predicted y=1, what fraction of it have cats?

$$\text{Precision (\%)} = \frac{True\ positive}{Number\ of\ predicted\ positive}\ x\ 100 = \frac{True\ positive}{(True\ positive + False\ positive)}\ x\ 100$$

<u>Recall</u>

Of all the images that actually have cats, what fraction of it did we correctly identifying have cats?

$$\text{Recall (\%)} = \frac{True\ positive}{Number\ of\ predicted\ actually\ positive}\ x\ 100 = \frac{True\ positive}{(True\ positive + True\ negative)}\ x\ 100$$


In this case the evaluation metrics are precision and recall.

For classifier A, there is a 95% chance that there is a cat in the image and a 90% chance that it has correctly detected a cat. Whereas for classifier B there is a 98% chance that there is a cat in the image and a 85% chance that it has correctly detected a cat.

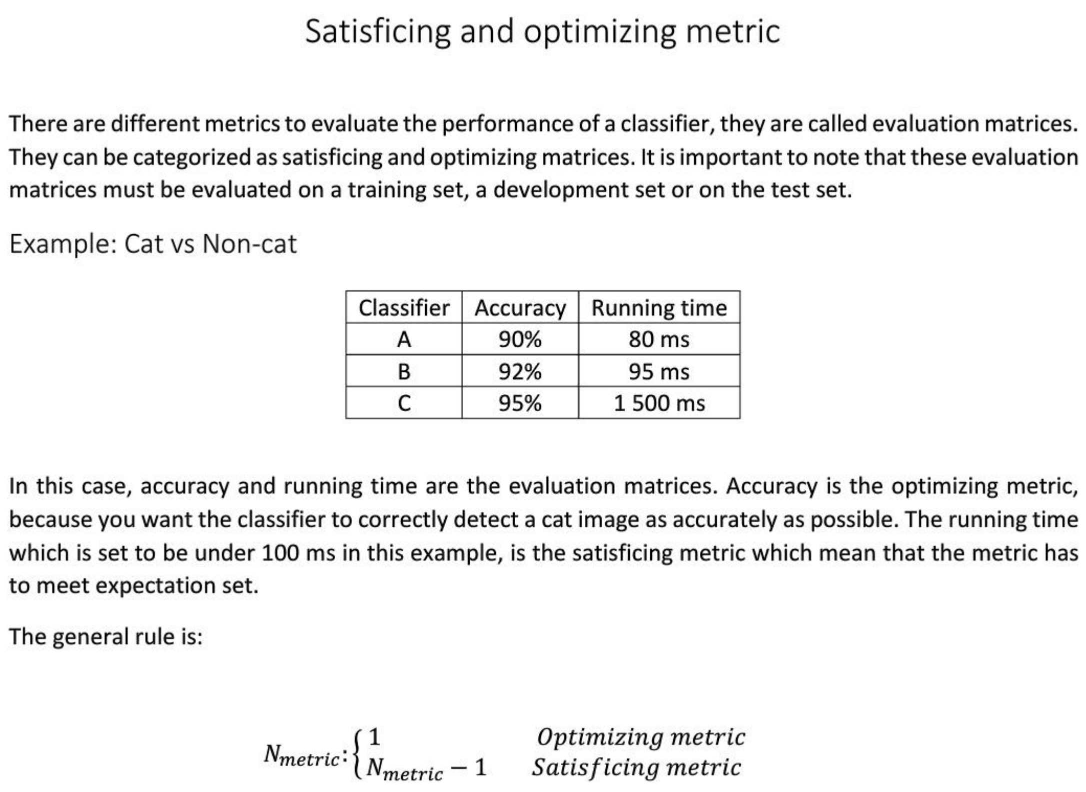
The problem with using precision/recall as the evaluation metric is that you are not sure which one is better since in this case, both of them have a good precision et recall. F1-score, a harmonic mean, combine both precision and recall.

$$\text{F1-Score} = \frac{2}{\frac{1}{p}+\frac{1}{r}}$$

Classifier	Precision (p)	Recall (r)	F1-Score
A	95%	90%	92.4 %
B	98%	85%	91.0%

Classifier A is a better choice. F1-Score is not the only evaluation metric that can be use, the average, for example, could also be an indicator of which classifier to use.

Satisficing and optimizing metrics


Satisficing and optimizing metric

There are different metrics to evaluate the performance of a classifier, they are called evaluation matrices. They can be categorized as satisficing and optimizing matrices. It is important to note that these evaluation matrices must be evaluated on a training set, a development set or on the test set.

Example: Cat vs Non-cat

Classifier	Accuracy	Running time
A	90%	80 ms
B	92%	95 ms
C	95%	1 500 ms

In this case, accuracy and running time are the evaluation matrices. Accuracy is the optimizing metric, because you want the classifier to correctly detect a cat image as accurately as possible. The running time which is set to be under 100 ms in this example, is the satisficing metric which mean that the metric has to meet expectation set.

The general rule is:

$$N_{metric}:\begin{cases}1 & Optimizing\ metric\\ N_{metric}-1 & Satisficing\ metric\end{cases}$$

Accuracy 和 Running time 这两个性能不太合适综合成单值评价指标。可以将 Accuracy 作为优化指标（Optimizing metic），Running time 作为满意指标（Satisficing metic）。给 Running time 设定一个阈值，在其满足阈值的情况下，选择 Accuracy 最大的模型。如果设定 Running time 必须在 100ms 以内，模型 C 不满足阈值条件，剔除；模型 B 相比较模型 A 而言，Accuracy 更高，性能更好

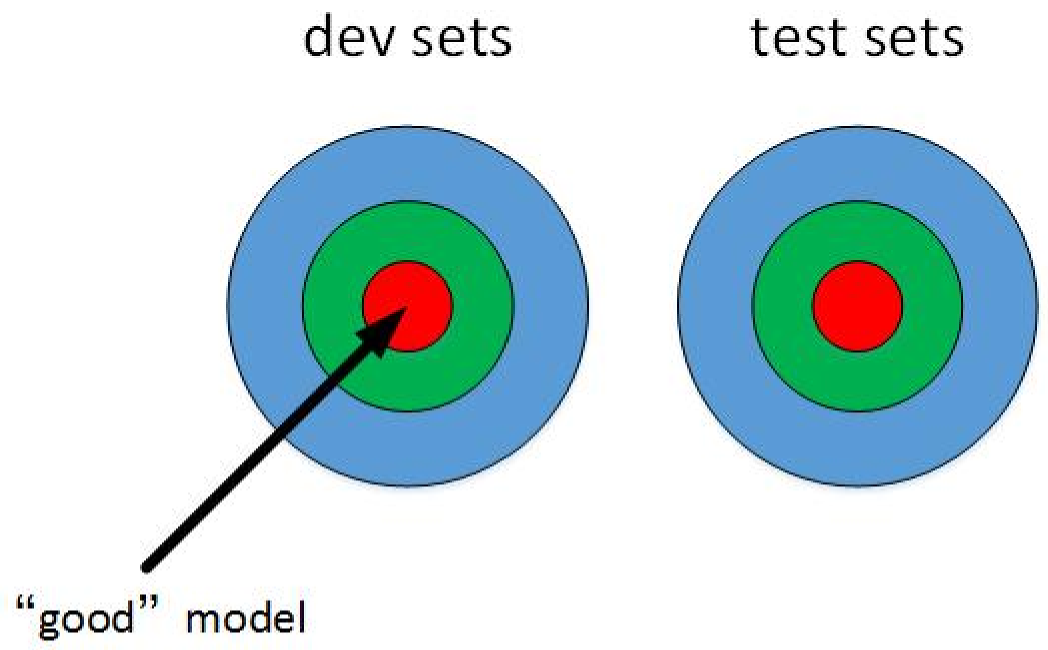
如果要考虑 N 个指标，则选择一个指标为优化指标，其他 N-1 个指标都是满足指标：

Train/dev/test distributions

训练、开发、测试集选择设置的一些规则和意见：

- ✓ 训练、开发、测试集的设置会对产品带来非常大的影响；
- ✓ 在选择开发集和测试集时要使二者来自同一分布，且从所有数据中随机选取；
- ✓ 所选择的开发集和测试集中的数据，要与未来想要或者能够得到的数据类似，即模型数据和未来数据要具有相似性；
- ✓ 设置的测试集只要足够大，使其能够在过拟合的系统中给出高方差的结果就可以，也许 10000 左右的数目足够；
- ✓ 设置开发集只要足够使其能够检测不同算法、不同模型之间的优劣差异就可以，百万大数据中 1%的大小就足够；

尽量保证 dev sets 和 test sets 来源于同一分布且都反映了实际样本的情况。如果 dev sets 和 test sets 不来自同一分布，从 dev sets 上选择的“最佳”模型往往不能够在 test sets 上表现得很好。好比在 dev sets 上找到最接近一个靶的靶心的箭，但是 test sets 提供的靶心却远远偏离 dev sets 上的靶心，结果肯定无法射中 test sets 上的靶心位置



- 样本数量不多（小于一万）的时候，通常将 Train/dev/test sets 的比例设为 60%/20%/20%
- 没有 dev sets 的情况下，Train/test sets 的比例设为 70%/30%
- 样本数量很大（百万级别）的时候，通常将相应的比例设为 98%/1%/1%或者 99%/1%

When to change dev/test sets and metrics
假设有两个猫的图片的分类器:

- 评估指标：分类错误率
- 算法 A: 3% 错误率
- 算法 B： 5% 错误率

初始的评价标准是错误率， A 更好一些。实际使用时发现算法 A 会通过一些色情图片，但是 B 没有。从用户的角度来说，更 倾向选择 B 模型，虽然 B 的错误率高一些。这时候需要改变之前只使用错误率作为评价标准，考虑新的情况进行改变。如增 加色情图片的权重，增加其代价
假设开始的评估指标如下:

$$\text{Error} = \frac{1}{m_{\text{dev}}} \sum_{i=1}^{m_{\text{dev}}} I\{y_{\text{pred}}^{(i)} \neq y^{(i)}\}$$

该评估指标对色情图片和非色情图片一视同仁
修改的方法，在其中加入权重 $w^{(i)}$:

$$\text{Error} = \frac{1}{\sum w^{(i)}} \sum_{i=1}^{m_{\text{dev}}} w^{(i)} I\{y_{\text{pred}}^{(i)} \neq y^{(i)}\}$$

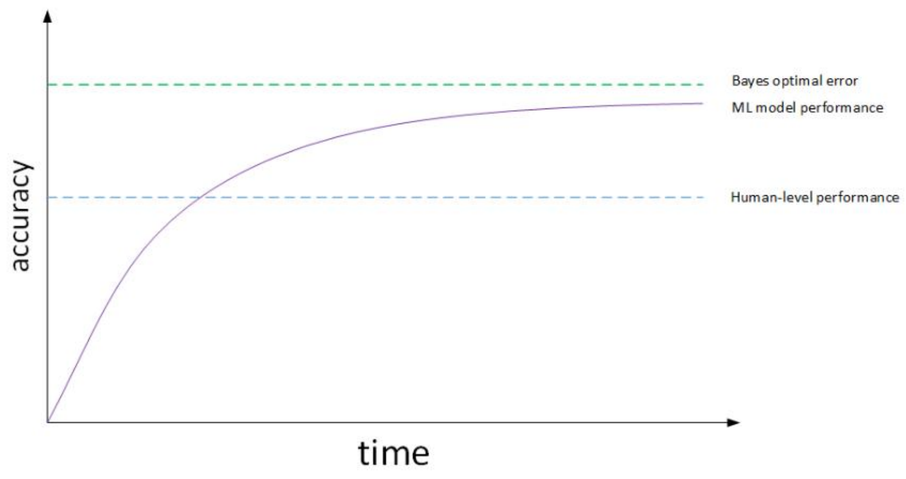
$$w^{(i)} = \begin{cases} 1, & x^{(i)} \text{ is non} - \text{porn} \\ 10 \text{ or } 100, & x^{(i)} \text{ is porn} \end{cases}$$

通过设置权重，当算法将色情图片分类为猫时，误差项会快速变大
概括来说，机器学习可分为两个过程:

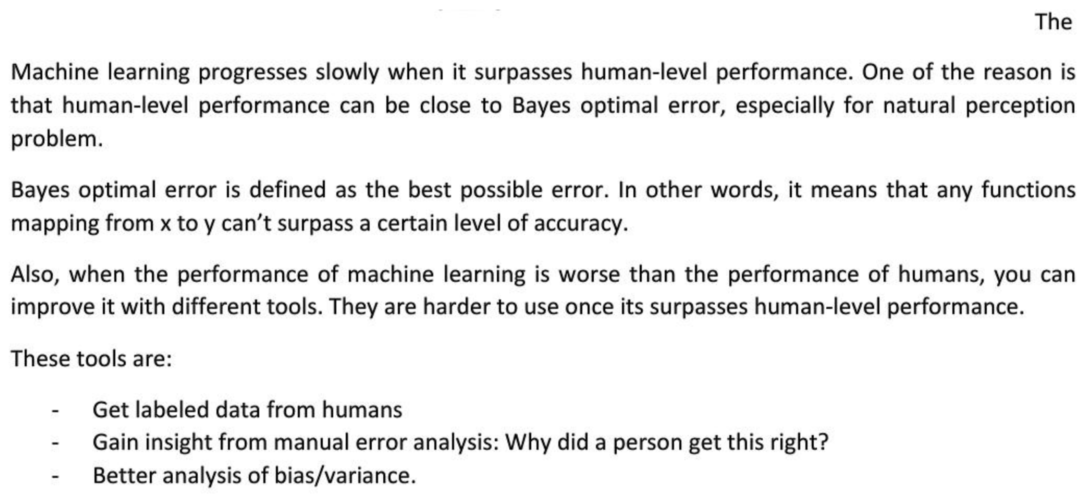
- Define a metric to evaluate classifiers
- How to do well on this metric
  第一步是找靶心，第二步是通过训练，射中靶心。但是在训练的过程中可能会根据实际情况改变算法模型的评价标准，进 行动态调整，如果评估指标无法正确评估算法的排名，则需要重新定义一个新的评估指标

3.3 Comparing to Human-level Performance

Why human-level performance?



贝叶斯最优错误率（Bayes optimal error），理论上任何模型都不能超过它，即没有任何办法设计出一个到的函数，让它能够超过一定的准确度，bayes optimal error 代表了最佳表现。



The

Machine learning progresses slowly when it surpasses human-level performance. One of the reason is that human-level performance can be close to Bayes optimal error, especially for natural perception problem.

Bayes optimal error is defined as the best possible error. In other words, it means that any functions mapping from x to y can’t surpass a certain level of accuracy.

Also, when the performance of machine learning is worse than the performance of humans, you can improve it with different tools. They are harder to use once its surpasses human-level performance.

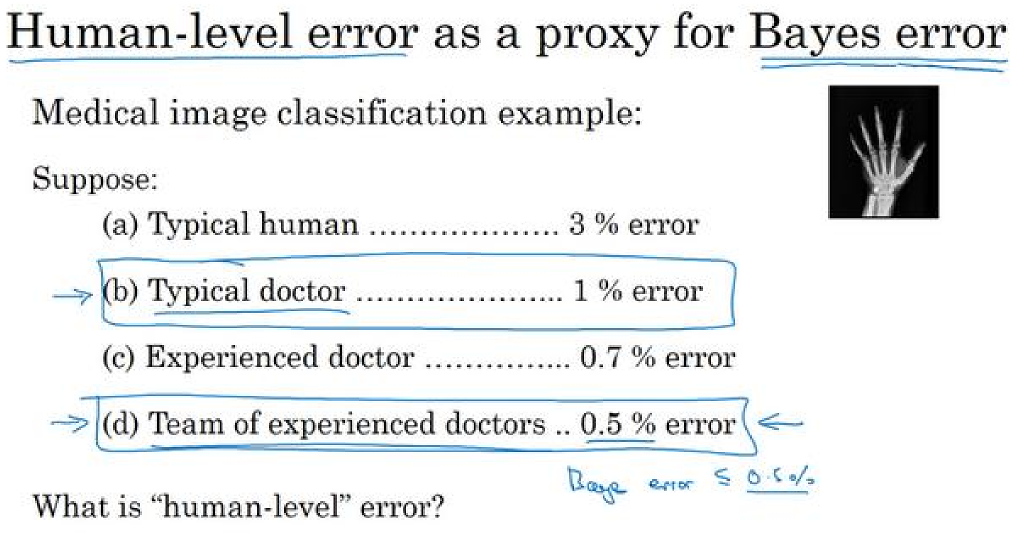
These tools are:

- Get labeled data from humans
- Gain insight from manual error analysis: Why did a person get this right?
- Better analysis of bias/variance.

可避免偏差（Avoidable bias）

贝叶斯错误率或者对贝叶斯错误率的估计和训练错误率之间的差值称为可避免偏差
理论上是不可能超过贝叶斯错误率的，除非过拟合
训练错误率和开发错误率之前的差值，说明算法在方差问题上还有多少改善空间

Understanding human-level performance



Human-level error as a proxy for Bayes error

Medical image classification example:

Suppose:

(a) Typical human .... 3 % error

→ (b) Typical doctor .... 1 % error

(c) Experienced doctor .... 0.7 % error

→ (d) Team of experienced doctors .. 0.5 % error ←

Bayes error ≤ 0.5%

What is "human-level" error?

在减小误诊率的背景下，人类水平误差在这种情形下应定义为：0.5% error。但是实际应用中，不同人可能选择的 human-level performance 基准是不同的，这会带来一些影响

如果在为了部署系统或者做研究分析的背景下，也许超过一名普通医生即可，即人类水平误差在这种情形下应定义为：1% error

假如该模型 training error 为 0.7%，dev error 为 0.8%。如果选择 Team of experienced doctors，即 human-level error 为 0.5%，则 bias 比 variance 更加突出。如果选择 Experienced doctor，即 human-level error 为 0.7%，则 variance 更加突出。选择什么样的 human-level error，有时候会影响 bias 和 variance 值的相对变化。当然这种情况一般只会在模型表现很好，接近 bayes optimal error 的时候出现。越接近 bayes optimal error，模型越难继续优化，因为这时候的 human-level performance 可能是比较模糊难以准确定义的。

Scenario A

In this case, the choice of human-level performance doesn't have an impact. The avoidable bias is between 4%-4.5% and the variance is 1%. Therefore, the focus should be on bias reduction technique.

Scenario B

In this case, the choice of human-level performance doesn't have an impact. The avoidable bias is between 0%-0.5% and the variance is 4%. Therefore, the focus should be on variance reduction technique.

Scenario C

In this case, the estimate for Bayes error has to be 0.5% since you can't go lower than the human-level performance otherwise the training set is overfitting. Also, the avoidable bias is 0.2% and the variance is 0.1%. Therefore, the focus should be on bias reduction technique.

Summary of bias/variance with human-level performance

- Human - level error – proxy for Bayes error
- If the difference between human-level error and the training error is bigger than the difference between the training error and the development error. The focus should be on bias reduction technique
- If the difference between training error and the development error is bigger than the difference between the human-level error and the training error. The focus should be on variance reduction technique

Improving your model performance

提高机器学习模型性能主要要解决两个问题：avoidable bias 和 variance。training error 与 human-level error 之间的差值反映的是 avoidable bias，dev error 与 training error 之间的差值反映的是 variance

基本假设：

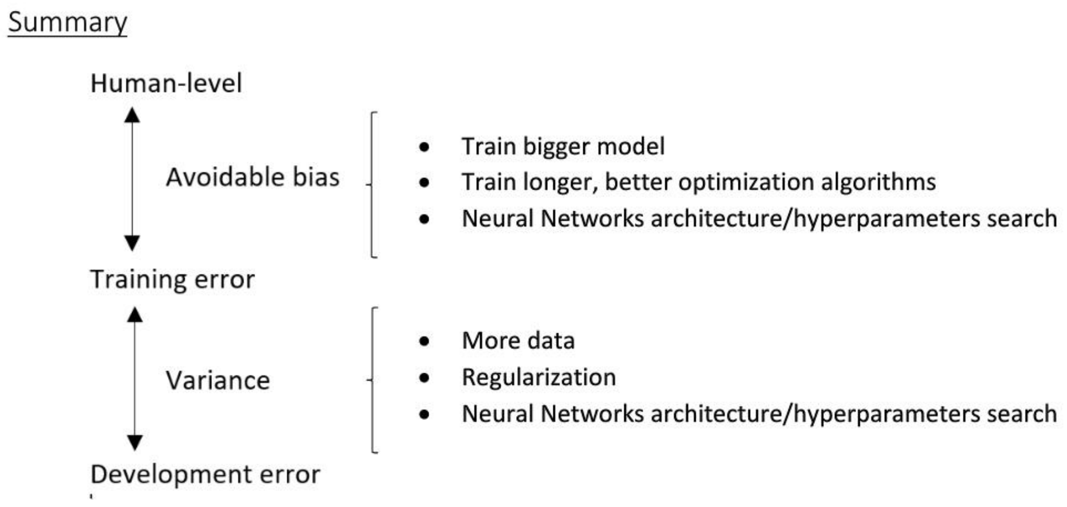
- 模型在训练集上有很好的表现；
- 模型推广到开发和测试集啥也有很好的表现

减少可避免偏差

- 训练更大的模型
- 训练更长时间、训练更好的优化算法（Momentum、RMSprop、Adam）
- 寻找更好的网络架构（RNN、CNN）、寻找更好的超参数

减少方差

- 收集更多的数据
- 正则化（L2、dropout、数据增强）
- 寻找更好的网络架构（RNN、CNN）、寻找更好的超参数


Summary
Human-level
Avoidable bias
Train bigger model
Train longer, better optimization algorithms
Neural Networks architecture/hyperparameters search
Training error
Variance
More data
Regularization
Neural Networks architecture/hyperparameters search
Development error