Evaluating model (Validation Techniques)

1. Resubstitution

所有的数据都被用于训练.

无法评估模型对于没有出现过的数据的效果.

2. Hold out validation 比.

将数据集划分为2个互斥的集，即训练集与测试集。如按照 6:4, 7:3, 8:2 的比例划分.

1) 划分形成的2个集中类别不平衡或数据分布不一致 ⇒ 分层采样 (stratified sampling)

2) 一次划分形成引入误差 ⇒ 重复多次划分，取多次结果的平均.

3) 每次都会有部分数据没有被用于训练，导致信息丢失.

3. K-Fold cross-validation. 比.

数据被划分为k个相同大小的子集，每次使用k-1个集作为训练集，剩下的一个作为测试集.

这样就重复k次训练，最终为误差为取k次训练的平均.

1) 为了保证数据一致性，则对k个集采用分层采样方法.

2) k的取值通常为10.

3) k个集的划分同样有随机性，因此重复划分. ⇒ p次k折交叉验证.

4. Leave-One-Out Cross-Validation (LOOCV).

K-Fold cross validation 的特殊情况，此时 $k = N$，每次只有一个样本作为测试集.

5. Random subsampling

每次从整个数据集中采样 n 个数据作为测试集，剩下的数据作为训练集.

6. Bootstrapping

前面提出的算法都是在训练集比D小的情况，会引入因训练规模不同而导致的估计偏差。

给定包含m个样本的数据集D，每次从D中有放回地采样m个样本组成训练集D_1。

重复这个步骤m次，即可得到m个包含m个样本的训练集。每次采样剩下的数据样本组成对应的测试集：

Completed dataset: $\{x_1, x_2, x_3, x_4, x_5\}$

1: $\{x_1, x_2, x_4, x_4, x_3\}$ $\{x_5\}$

2: $\{x_4, x_5, x_5, x_1, x_1\}$ $\{x_2, x_3\}$

⋮

一个样本在m次采样中始终不被采样到的概率是：$(1-\frac{1}{m})^m$；有：

$$\lim_{m\to\infty}(1-\frac{1}{m})^m \Rightarrow \frac{1}{e}\approx 0.368.$$

适用于数据集较小，难以有效划分训练/测试集时很有用。

Evaluation Metrics

classification

$$prevalence = \frac{n_{pos}}{n}$$

数据集中正样本的比例.

1. Accuracy

$$accuracy = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(f(x_i) = y_i)$$

不适用于类别不平衡情况，无法知道每个类别的情况.

2. Confusion Matrix, Precision, Recall, F-measure.

(binary)

	Predict positive	Predict negative.
Actually positive	True positive (TP)	False negative (FN)
Actually negative	False positive (FP)	True negative (TN)

$$precision = \frac{TP}{TP + FP}$$ ⇒ 预测为正类的样本中有多少确实是正类. (正类的precision)

$$recall = \frac{TP}{TP + FN}$$ ⇒ 所有正类样本有多少被预测正确. (正类的recall)

$$sensitivity = recall = \frac{TP}{n_p}$$ ⇒ 正类的准确率, true positive rate

$$specificity = \frac{TN}{n_n}$$ ⇒ 负类的准确率, true negative rate

$$type\text{-}I\ error = false\ positive\ rate = \frac{FP}{TN + FP} = 1 - specificity$$ ⇒ 把假的预测为真的.

$$type\text{-}II\ error = false\ negative\ rate = \frac{FN}{TP + FN} = 1 - sensitivity$$

balanced accuracy $= \frac{1}{2}$(sensitivity + specificity)

$= \frac{1}{2}$(recall-pos + recall-neg)

$= \frac{1}{2}$(acc-pos + acc-neg)

每个类别正确率的平均.

~~F-measure $= \frac{(\beta^2+1)\ P\times R}{(\beta^2\times(P)+R)}$~~

F-measure

$$F_\beta = \frac{(1+\beta^2)\ P\times R}{\beta^2\cdot P + R}$$

⇒ P和R的调和平均。$\beta>1$时, recall更重要.

$\beta<1$时, precision更重要

$$F_1 = \frac{2\cdot P\cdot R}{P+R}$$

⇒ precision和recall同样重要

balanced accuracy 和 F-measure 适用于类别不平衡问题.

1) $N \gg P$时, recall-neg一定总是很高, 因此F_1更适合.

2) $P \gg N$时, recall-pos和pre-pos都很高, 因此balanced accuracy更适合.

Average Precision (AP)

precision只考虑了返回结果的准确率, 却没有考虑结果的顺序。在排序相关应用中(搜索), 顺序是很重要的.

1) 按照样本预测为正类别的概率从大到小排序.

2) 以每个位置作为划分点, 计算precision.

3) 求P = 所有precision的平均.

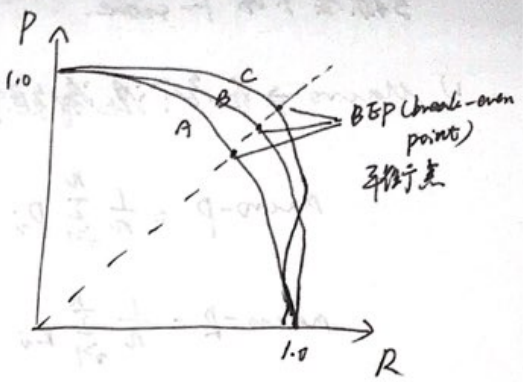
4) mAP (mean average precision) = 多次查询的AP值的平均.

P-R Curve

1) 将样本预测为正类的概率从大到小排序

2) 分别以每个位置作为划分点，得到对应的P. R.

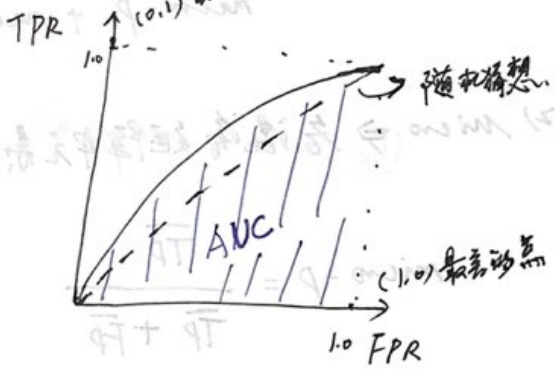
3) BEP越大，说明性能越好.



ROC-Curve.

$$\begin{cases} TPR = \dfrac{TP}{TP+FN} & \Rightarrow \text{true positive rate} \\ FPR = \dfrac{FP}{FP+TN} & \Rightarrow \text{false positive rate} \end{cases}$$

AUC越大，性能越好.



多标签下的 F-score.

1) macro ⇒ 在各个混淆矩阵上分别计算 P 和 R，有：

$$\text{macro-}P = \frac{1}{n}\sum_{i=1}^{n} P_i$$

$$\text{macro-}R = \frac{1}{n}\sum_{i=1}^{n} R_i$$

$$\text{macro-}F_1 = \frac{2 \times \text{macro-}P \times \text{macro-}R}{\text{macro-}P + \text{macro-}R}$$

2) micro ⇒ 各混淆矩阵中元素对平均，再算一次 P.R.

$$\text{micro-}P = \frac{\overline{TP}}{\overline{TP} + \overline{FP}}$$

$$\text{micro-}R = \frac{\overline{TP}}{\overline{TP} + \overline{FN}}$$

$$\text{micro-}F_1 = \frac{2 \times \text{micro-}P * \text{micro-}R}{\text{micro-}P + \text{micro-}R}.$$

Log Loss. (logistic regression loss / cross-entropy loss)

(熵?)

~~binary: $L = -\log P_r(y|p)$~~

1° binary: $L = -[y \log p + (1-y)\log(1-p)]$ → 样本属于类别1的概率.

$$L = \frac{1}{N}\sum_{i=0}^{N} L_i = \frac{1}{N}\sum_{i=0}^{N} -[y_i \log p_i + (1-y_i)\log(1-p_i)]$$

likelihood $= p^y \cdot (1-p)^{(1-y)}$ $\xrightarrow{-\log}$ $-[y\log p + (1-y)\log(1-p)]$

2° multiclass

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{i,k}\log p_{i,k}$$

likelihood $= \prod_{i=1}^{N}\prod_{k=1}^{K} P_{i,k}^{y_{i,k}}$ $\xrightarrow{-\log}$ $-\sum_{i=1}^{N}\sum_{i=1}^{K} y_{ik}\log p_{ik}$.

y_i是一个 1-of-k binary indicator (one-hot encoding)

Motivation 比模型预测有时间的预测结果. 所以accuracy, AV-ROC, AV-PRC都是一样的.

Gain $= y\,p(x) + (1-y)(1-p(x))$

~~熵: $H = -\sum_i P_i \log P_i$~~

~~Log loss~~ $= -[y\log p + (1-y)\log(1-p)]$.

↓

只有当结果为负例正确时会有 $y\log p$. 因此 Log loss 对预测错的结果惩罚非常严重.