Assessing model performance

Machine Learning 2019

Cesare Alippi, Jürgen Schmidhuber, Michael Wand, Paulo Rauber

TAs: Róbert Csordás, Krsto Proroković, Xingdong Zuo, Francesco Faccio, Louis Kirsch

Quality assessment of the solution

«How good is your «good»?»

Recall: the statistical learning approach

The structural risk

$$\bar{V}(\theta) = \int L(y, f(\theta, x))\, p_{x,y} dxy$$

$$\theta^o = \arg\min_{\theta \in \Theta} \bar{V}(\theta)$$

The empirical risk

$$V_N(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(\theta, x_i))$$

$$\hat{\theta} = \arg\min_{\theta \in \Theta} V_N(\theta)$$

Recall: the statistical learning approach

It is defined as the cardinality of the largest set of points that the algorithm can shatter.
It describes the capability of the function in classifying data into different groups

In the classification case, by considering the empirical risk as the ratio of correct classification on the training set and defined d_{VC} as the Vapnik Chervonenkis dimension, we have

$$\Pr\left(\bar{V}(\hat{\theta}) \leq V_N(\hat{\theta}) + \sqrt{\frac{1}{N}\left[d_{VC}\left(log(\frac{2N}{d_{VC}})+1\right) - log(\frac{\delta}{4})\right]}\right) \geq 1-\delta$$

The above holds when $d_{VC} << N$

d_{VC} measures the expressive power of the model family

Recall: the statistical learning approach

The Empirical mean converges to its expectation uniformely as N goes to infinity and for each element of an arbitrarily selected finite sequence of parameter estimates (finite number of models). From the Hoeffding inequality we have

When the random variables L(y,f(theta,x)) are bounded in [0,1]: 2e^(-2Nt^2)

$$\Pr\left(\sup_{L\in A}|V_N(L(\theta))-E_x[L(\theta,x)]|>\varepsilon\right)\leq 2me^{-2N\varepsilon^2}$$

$$A=\{L(\hat{\theta}_1,x),\cdots,L(\hat{\theta}_m,x)\}$$

The traditional statistical approach

The UCEM property can also hold for a family of function with infinite models

$$A = \{L(\theta, x), \theta \in \Theta\}$$

provided that the Pollard dimension of family A is finite (the dimension extends the VC one to the real case).

For deep learning (feedforward networks) the VC-dim is O(d log d), d being the number of weights

How to estimate performance $\bar{V}(\hat{\theta})$

- *Apparent Error Rate* (AER),or *resubstitution,* or *training error*

- The method uses the empirical risk to estimate the structural one
- Set Z_N is used to both infer the model and estimate its accuracy performance
  - The AER is a strongly optimistically biased estimate unless N is very large

How to estimate performance $\bar{V}(\hat{\theta})$

- *Sample Partitioning* (SP), or *crossvalidation*
- It estimates the generalization error $\bar{V}(\hat{\theta})$ on a virgin –independent- dataset
- Sets S_D and S_E are generated by *randomly* splitting Z_N into two disjoint subsets.
  - S_D is used to learn the model and
  - S_E to estimate its performance.
- SP can be considered to be a good unbiased estimate if S_E is large enough

How to estimate performance $\bar{V}(\hat{\theta})$

- k-*fold Crossvalidation* (kCV):

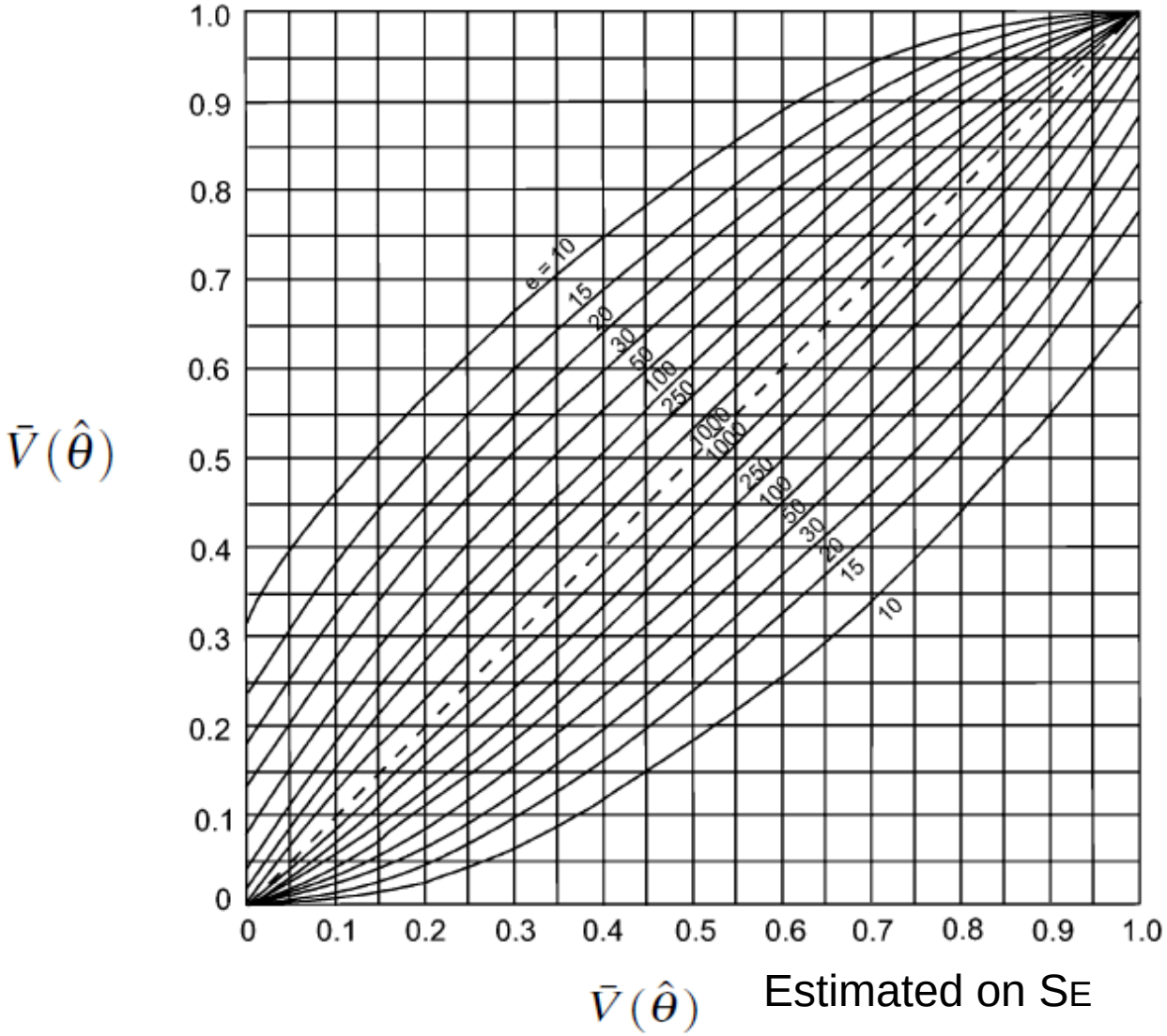
- Z_N is randomly split into k disjoint subsets of equal size. For each subset the remaining k − 1 subsets are merged to form S_D and the reserved subset is used as S_E .
- The resulting k estimates are averaged.
- A common choice for k is a number between 5 and10
- This estimate is a generalization of LOO.

How to estimate performance $\bar{V}(\hat{\theta})$

- *Leave-One-Out* (LOO):

- S_E contains one pattern of Z_N, and S_D contains the remaining N − 1 patterns.
- The procedure is iterated N times by holding out each pattern in Z_N; the resulting N estimates are averaged.
  - Estimates from each fold are highly correlated; the estimate has large variance

Comments on the finiteness of the test set



A classification example: binomial law, confidence 0.95, crossvalidation method

How to evaluate the *quality* of the estimate

Consider a data set Z_n obtained by extracting n i.i.d. samples $x_1, \ldots, x_n$ from random variable x defined over X, i.e., $Z_n = \{x_1 \ldots, x_n\}$ and construct the estimator $\Phi_n = \Phi(Z_n)$. We are interested in providing an indication of the quality ζ of Φ_n, e.g., we wish to provide a confidence interval for Φ_n.

Clearly, the ideal framework would recommend to carry out the following procedure

1. Extract m independent data sets of cardinality n from X so as to generate datasets $Z_n^1, \ldots, Z_n^m$;
2. Evaluate, in correspondence of the generic i-th data set Z_n^i the estimator $\Phi_n^i = \Phi(Z_n^i)$. Repeat this procedure for all $i = 1, \ldots, m$.;
3. Estimate the quality $\zeta\left(\Phi_n^1, \ldots, \Phi_n^m\right)$ of the estimator Φ_n based on the m realizations $\Phi_n^i = \Phi(Z_n^i), i = 1, \ldots, m$.

Having few data: The Bootstrap method

Unfortunately, the above framework is mostly theoretical: if we have m independent datasets Z_n we should use all nm data to provide a better estimate. This means that in practical applications we have only a dataset but, at the same time, we are interested in evaluating the quality ζ of the estimator Φ_n.

- In the bootstrap method data sets $Z_n^i,\ i = 1, \ldots, m$ are extracted with replacement from Z_n

Algorithm 15: The bootstrap algorithm

```
i = 0;
while i < m do
    Extract n samples with replacement from Z_n and insert them in Z_n^i;
    Compute Φ_n^i = Φ(Z_n^i);
    i = i+1;
end
Evaluate the assessment ζ(Φ_n^1, ..., Φ_n^m) of the quality of the estimator Φ_n.
```

- Bootstrap underestimates the test error $\bar{V}(\hat{\theta})$

Model validity

« Which model provides better performance at task?»

A test on residuals: the regression problem

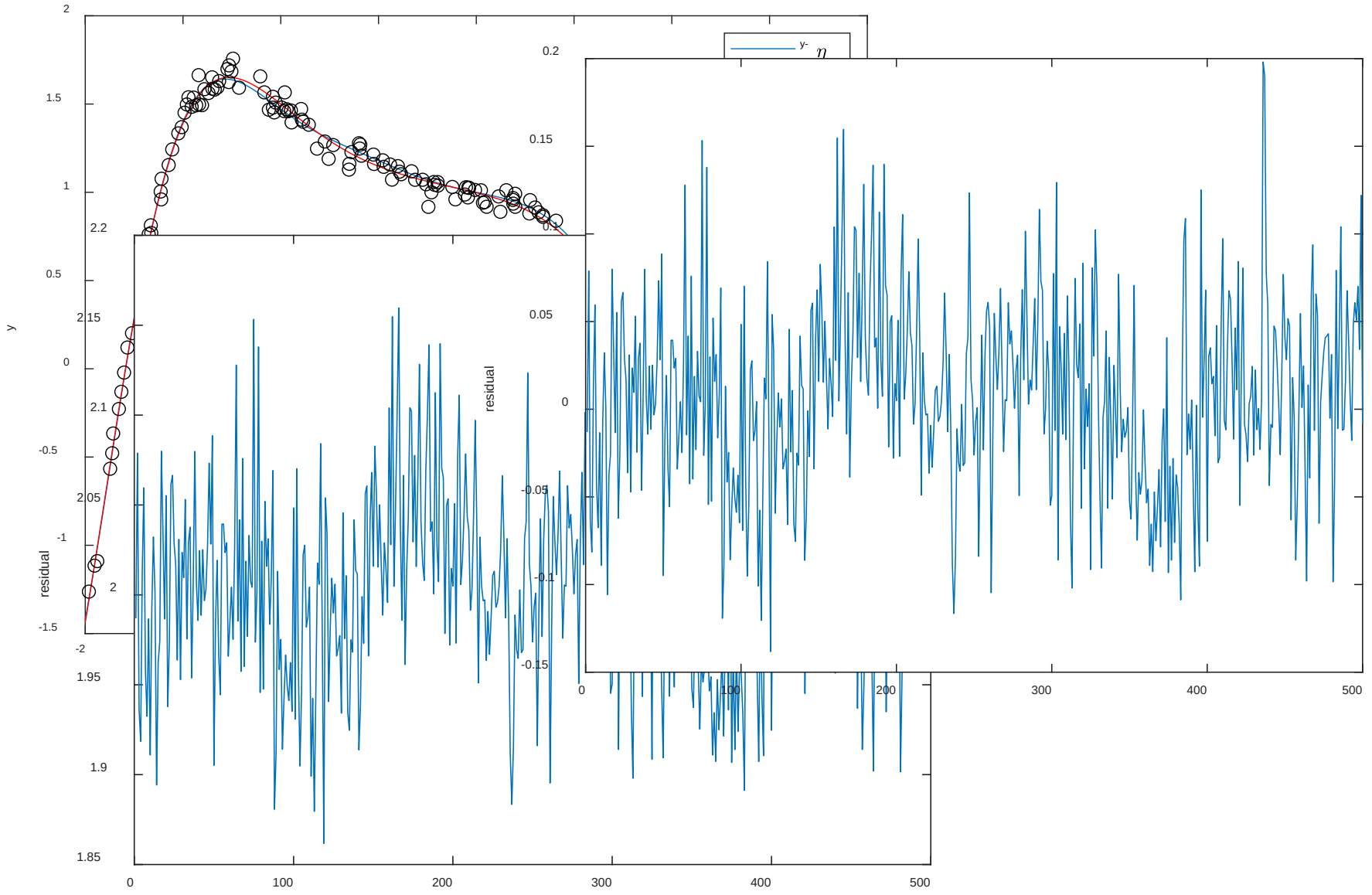
- In regression problems a first quality test requires to inspect the residual errors of the test set

$$\varepsilon_i = y(x_i) - f(\hat{\theta}, x_i), (x_i, y_i) \in S_E$$

- The residual error has to be characterized by a null expectation (unbiased)

$$E[\varepsilon] = 0$$

- Otherwise, by removing the bias (offset) we improve right away the model performance


y
residual
y-
η

A test on residuals

- Verification of hypothesis $E[\varepsilon] = 0$ requires a test on the sample mean:

$$H_0 : E[\varepsilon] = 0$$

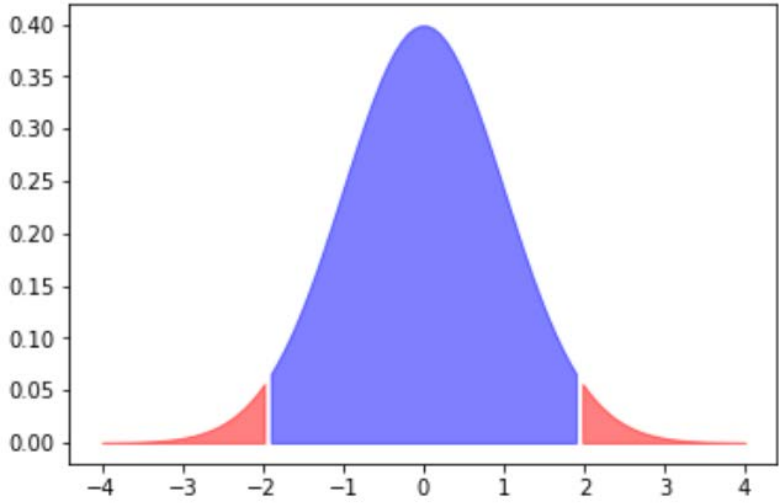
$$H_1 : E[\varepsilon] \neq 0$$

- Under the null hypothesis, the Central Limit Theorem grants

$$T = \frac{\overline{\varepsilon}}{\sqrt{s^2/N}} \sim N(0,1)$$

$$\overline{\varepsilon} = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_i$$

$$s^2 = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_i^2$$



- If T is outside the 95% confidence interval (-1.96, 1.96), then reject the null hypothesis $H_0 : E[\varepsilon] = 0$

Is model A better than model B?

- Consider two (possibly distinct) samples

$$(x_1^a, y_1^a), (x_2^a, y_2^a), \ldots, (x_{N_a}^a, y_{N_a}^a) \quad (x_1^b, y_1^b), (x_2^b, y_2^b), \ldots, (x_{N_b}^b, y_{N_b}^a)$$

- Assume both models have no bias, i.e., $E[\varepsilon_a] = E[\varepsilon_b] = 0$
  - otherwise subtract the estimated bias

$$\overline{\varepsilon_k} = \frac{\sum_{i=1}^{N_k} \varepsilon_{k,i}}{N_k} \quad \text{from} \quad f_k(x_i^k)$$

- Design a hypothesis test on the variance

$$H_0 : \mathrm{Var}[\varepsilon_a] = \mathrm{Var}[\varepsilon_b] \qquad H_1 : \mathrm{Var}[\varepsilon_a] \neq \mathrm{Var}[\varepsilon_b]$$

- If models are different then, the model with the smaller variance is preferable.

Is model A better than model B?

- Test

$$H_0 : \mathrm{Var}[\varepsilon_a] = \mathrm{Var}[\varepsilon_b]$$
$$H_1 : \mathrm{Var}[\varepsilon_a] \neq \mathrm{Var}[\varepsilon_b]$$

- Evaluate squared residuals

$$e_{k,i} = \left(y_i^k - f_k(x_i^k)\right)^2, \quad i = 1, \ldots, N_k, \quad k = a, b$$

- Under the CLT compute the statistic

$$T = \frac{\overline{e_a} - \overline{e_b}}{\sqrt{\frac{s_a^2}{N_a} + \frac{s_b^2}{N_b}}} \sim N(0, 1)$$

$$\overline{e_k} = \frac{\sum_{i=1}^{N_k} e_{k,i}}{N_k}$$

$$s_k^2 = \frac{\sum_{i=1}^{N_k} \left(e_{k,i} - \overline{e_k}\right)^2}{N_k - 1}$$

- If T is outside the 95% confidence interval (-1.96, 1.96), then reject H_0 and select the model with the smaller variance.

And for classification?

Computations are similar to those used for regression

Sample means are the ratios of correct classifications (classification accuracy)

$$\overline{e_k} = \frac{\sum_{i=1}^{N_k} e_{k,i}}{N_k} \qquad k = a, b$$

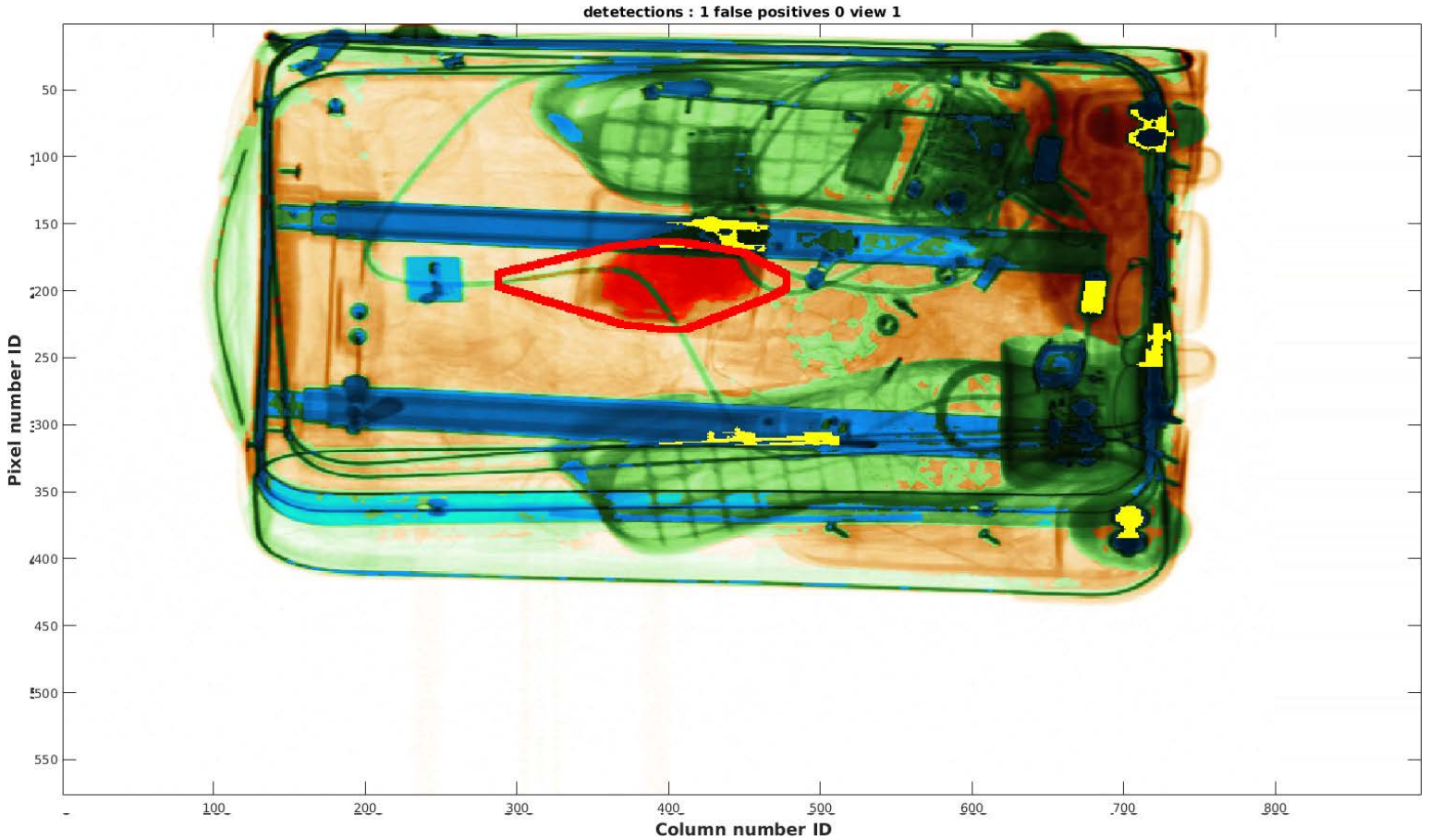
- For regression we had $$s_k^2 = \frac{\sum_{i=1}^{N_k} \left(e_{k,i} - \overline{e_k}\right)^2}{N_k - 1}$$

- For classification we have $$s_k^2 = N_k \cdot \overline{e_k}(1 - \overline{e_k})$$

Assessing anomaly detection methods

An anomaly detector is a classifier that processes data streams looking for anomalies

- Example: explosive detection



Assessing anomaly detection methods

The analysis holds for classifiers but the anomaly case eases the explanation

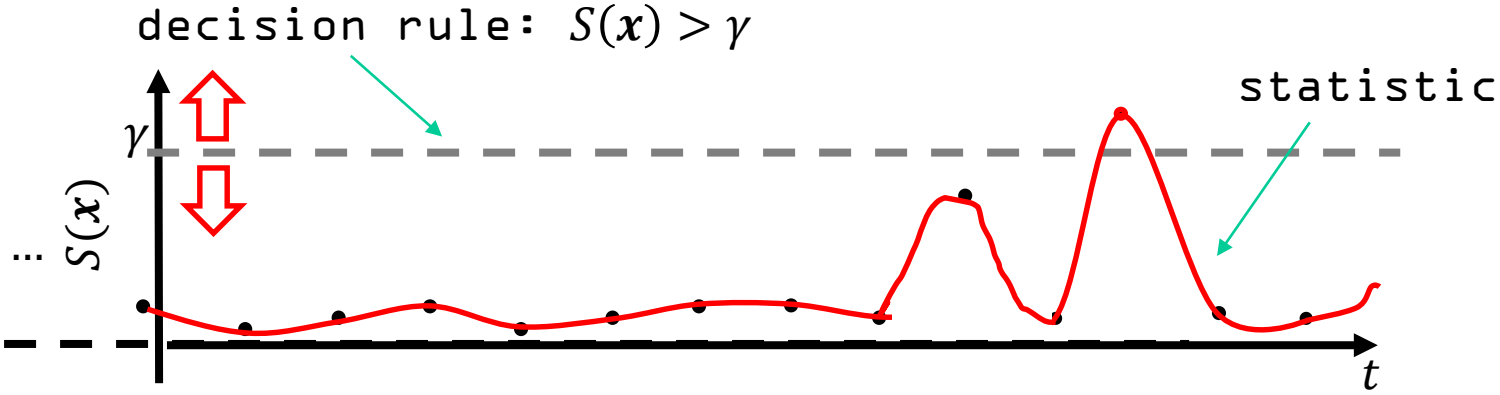
- True positive rate: $TPR = \frac{\#\{\text{anomalies correctly detected}\}}{\#\{\text{anomalies}\}}$
- False positive rate: $FPR = \frac{\#\{\text{normal samples detected } as\ anomalies\}}{\#\{\text{normal samples}\}}$

You have probably also heard of

- False negative rate (or miss-rate): $FNR = 1 - TPR$
- True negative rate (or specificity): $TNR = 1 - FPR$
- **Precision** on anomalies: $\frac{\#\{\text{anomalies correctly detected}\}}{\#\{\text{detections}\}}$
- **Recall** on anomalies (or sensitivity, hit-rate): TPR

Assessing anomaly detection methods

- We always trade-off TPR and FPR (and derived quantities) through some algorithm parameters (hyperparameters).
- For instance consider hyperparameter γ applied to statistic/method *S(x)*.
- By tuning γ we achieve different detection performance (e.g. more true positive, more false positives)



Assessing anomaly detection methods

- To correctly assess the method performance it is necessary to consider at least **two indicators**
  - TPR, FPR or
  - Accuracy and F1 score that combine TPR and FPR

$$\text{Accuracy} = \frac{\#\{\text{anomalies detected}\} + \#\{\text{normal samples not detected}\}}{\#\{\text{samples}\}}$$

$$\text{F1 score} = \frac{2\#\{\text{anomalies detected}\}}{\#\{\text{detections}\} + \#\{\text{anomalies}\}}$$

Accuracy and F1 score equal 1 when we have “ideal detecting methods”, i.e., the method detects all anomalies without false positives

Assessing anomaly detection methods

When comparing two different methods we have to make sure that both have been configured in their best conditions

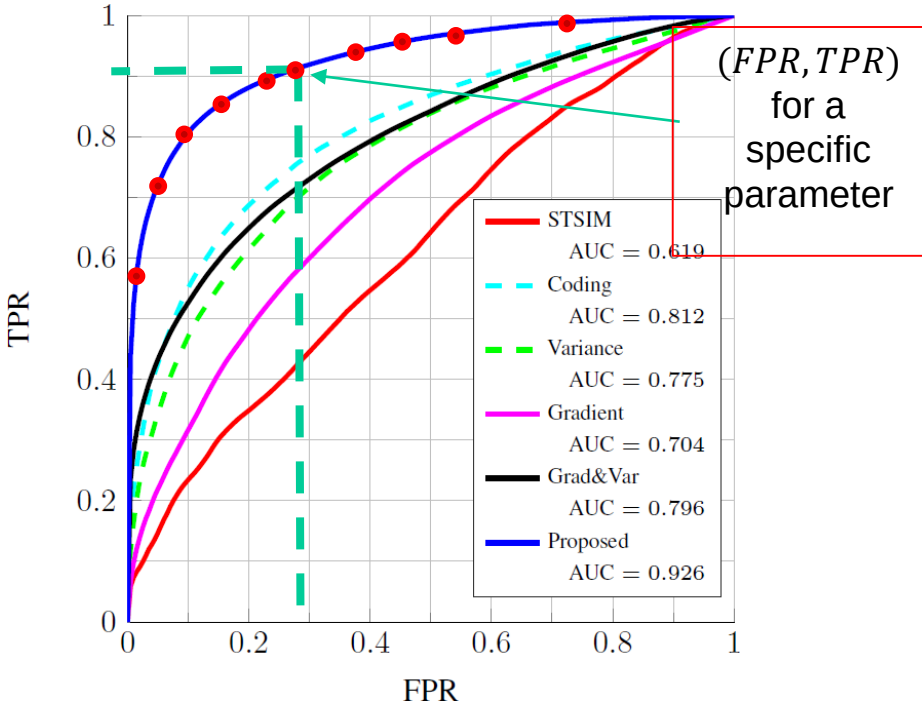
By exploring a large number of configurations we design the **Receiver Operating Characteristic** (ROC) curve

The ideal detector would achieve:

- $FPR = 0\%$,
- $TPR = 100\%$

The larger the **Area Under the Curve** (AUC) for a method, the better (the method)

The best hyperparameter is the one closest to point (0,1)



Loss functions for classifiers: Logistic regression

- Linear classifiers neither provide a bounded output nor a probabilistic interpretation of it
- Logistic regression aims at training the network parameters so that the sigmoidal output is supported by a probabilistic framework
- Define

$$p = Pr(y_i = 1|x) = \frac{1}{1+e^{-x^T\theta}} = \frac{e^{x^T\theta}}{1+e^{x^T\theta}}$$

$$1-p = Pr(y_i = 0|x) = \frac{1}{1+e^{x^T\theta}}$$

Logit function:

$$\ln\frac{p}{1-p} = x^T\theta$$

Logistic regression

- The probabilities can be cast in a canonical form

$$Pr(y_i|x) = p^{y_i}(1-p)^{1-y_1}$$

- And for n training samples

$$Pr(y_i|x_1, x_2, \cdots x_n) = \prod_{i-1}^{n} p^{y_i}(1-p)^{1-y_i}$$

- The log-likelihood becomes

$$L(\theta) = \ln\left(Pr(y_i|x_1, x_2, \cdots x_n)\right)$$

Logistic regression

- From which

$$L(\theta) = \sum_{i=1}^{n} y_i \ln p + (1 - y_i) \ln(1 - p)$$

- and

$$\hat{\theta} = \arg\max_{\theta \in \Theta} L(\theta)$$

- i.e.,

$$\hat{\theta} = \arg\max_{\theta \in \Theta} y_i x_i^T \theta - \ln(1 + e^{x_i^T \theta})$$

- The estimate can be found with a gradient ascent procedure

Loss function for classifiers

- Square error

$$L(y(x), f(\theta, x)) = (y(x) - f(\theta, x))^2$$

- Cross entropy (binary classes)

$$L(y(x), f(\theta, x)) = -[y(x) \log f(\theta, x) + (1 - y(x)) \log(1 - f(\theta, x))]$$

- Cross entropy (multi classes)

$$L(y(x), f(\theta, x)) = -\sum_{c=1}^{M} y_c(x) \log f_c(\theta, x)$$