Model Assessment and Selection

Dr. Lourdes Peña-Castillo
Departments of Computer Science and Biology
Memorial University of Newfoundland

Why to assess generalization performance?

- As we have seen, the generalization performance of a learning method correlates with its prediction capability on independent test data
- Assessing this generalization performance is critical to choose a good learning method and/or model for a specific task, and gives us an estimate of how good the chosen model is

Different Types of Error

Assume we have a loss function $L(Y, f'(X)$ for measuring errors between Y and a prediction model $f'(X)$ that has been estimated from a training set τ.

Generalization error Err_τ $= E[L(Y, f'(X))|\tau]$ where X and Y are drawn randomly from their joint distribution (population), τ is fixed, and Err_τ referes to the error for this specific training set.

Expected prediction error Err $= E[L(Y, f'(X))] = E[\mathrm{Err}_\tau]$. Err averages over everything that is random, including the randomness in the training set that produced f'.

Training error $\overline{err} = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f'(x_i))$ is the average loss over the training sample.

Typical loss functions

For regression:

$$L(Y, f'(X)) = \begin{cases} (Y - f'(X))^2 \text{ squared error} \\ |Y - f'(X)| \text{ absolute error.} \end{cases}$$

For classification:

$$L(Y, f'(X)) = I(Y \neq \hat{Y})) \text{ 0 - 1 loss.}$$

How to estimate Err?

- We want to know the expected prediction error (or expected test error) Err of our model *f'*
- But training error is not a good estimate of Err
- Training error consistently decreases as the model becomes more complex and is able to adapt to more complicated underlying structures
  - decrease in bias, increase in variance
- A model with zero training error is overfit to the training data and typically generalizes poorly

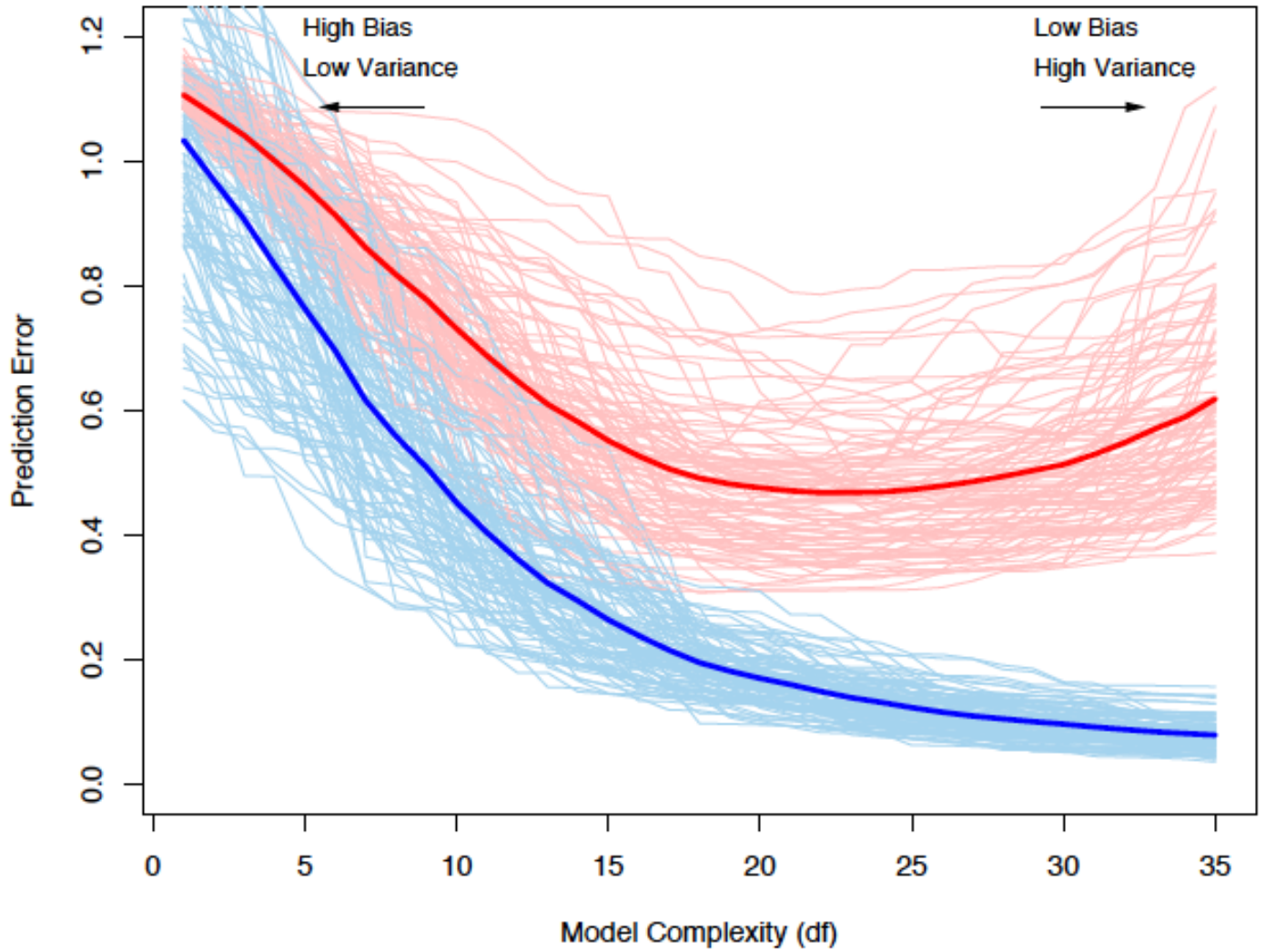


FIGURE 7.1. *Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error* $\overline{\text{err}}$*, while the light red curves show the conditional test error* $\text{Err}_{\mathcal{T}}$ *for* 100 *training sets of size* 50 *each, as the model complexity is increased. The solid curves show the expected test error* Err *and the expected training error* $\text{E}[\overline{\text{err}}]$*.*

Fig. from *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman, 2009.

Why we want to estimate Err?

- Model selection
  - Goal: estimate the performance of different models to choose the best one
  - Learning methods have tuning parameters α that vary the complexity of the model and we want to find the value of α that minimizes the average test error.
- Model assessment
  - Goal: estimate the test error of a chosen final model (on new data)

How to estimate Err?

- Validation Set Approach
  - Divide the available set of observations into three parts: a training set, a validation set and a test set.
    - Fit the model on the training set
    - Evaluate the performance of alternative models using the validation set (model selection)
    - Assess the generalization error of the final chosen model using the test set (model assessment)

Important: Test set is only used at the end of the analysis!

Fig. from *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman, 2009.

How to estimate Err?

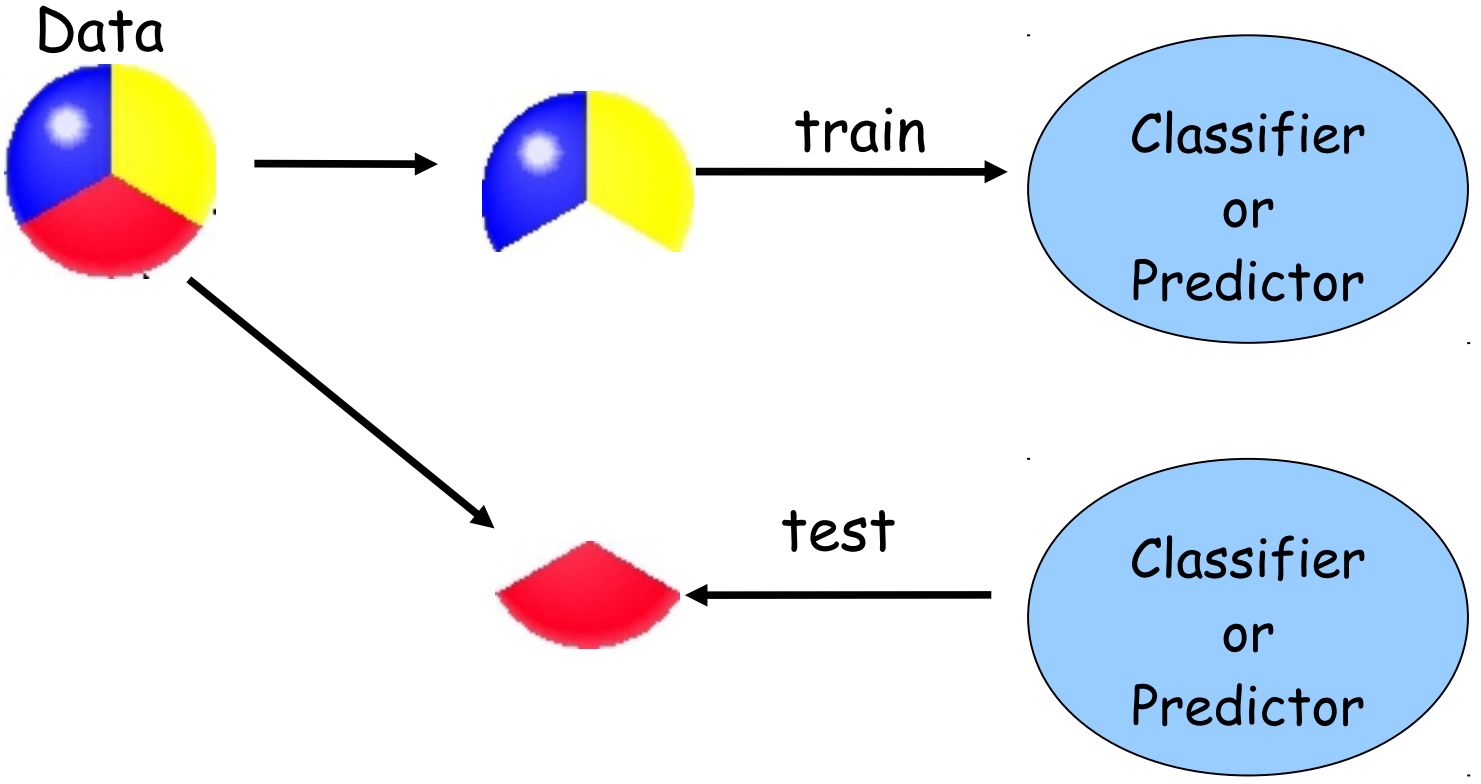
- Validation Set Approach Drawbacks:
  - Estimation of the test error can be highly variable depending on which observations are included in each set
  - Since learning methods tend to perform worse on fewer observations, this approach may tend to overestimate the test error rate for the model fit on the entire data set

Important: Test set is only used at the end of the analysis! Otherwise, we will underestimate the true test error.

Fig. from *The Elements of Statistical Learning* by Hastie, Tibshirani and Friedman, 2009.

How to estimate Err?

- K-Fold Cross-Validation
  - Simplest and most widely used method for estimating the expected prediction error Err
  - It works by splitting the data into *K* approximately equal-sized parts, and using some parts to fit the model and a different part to test it.



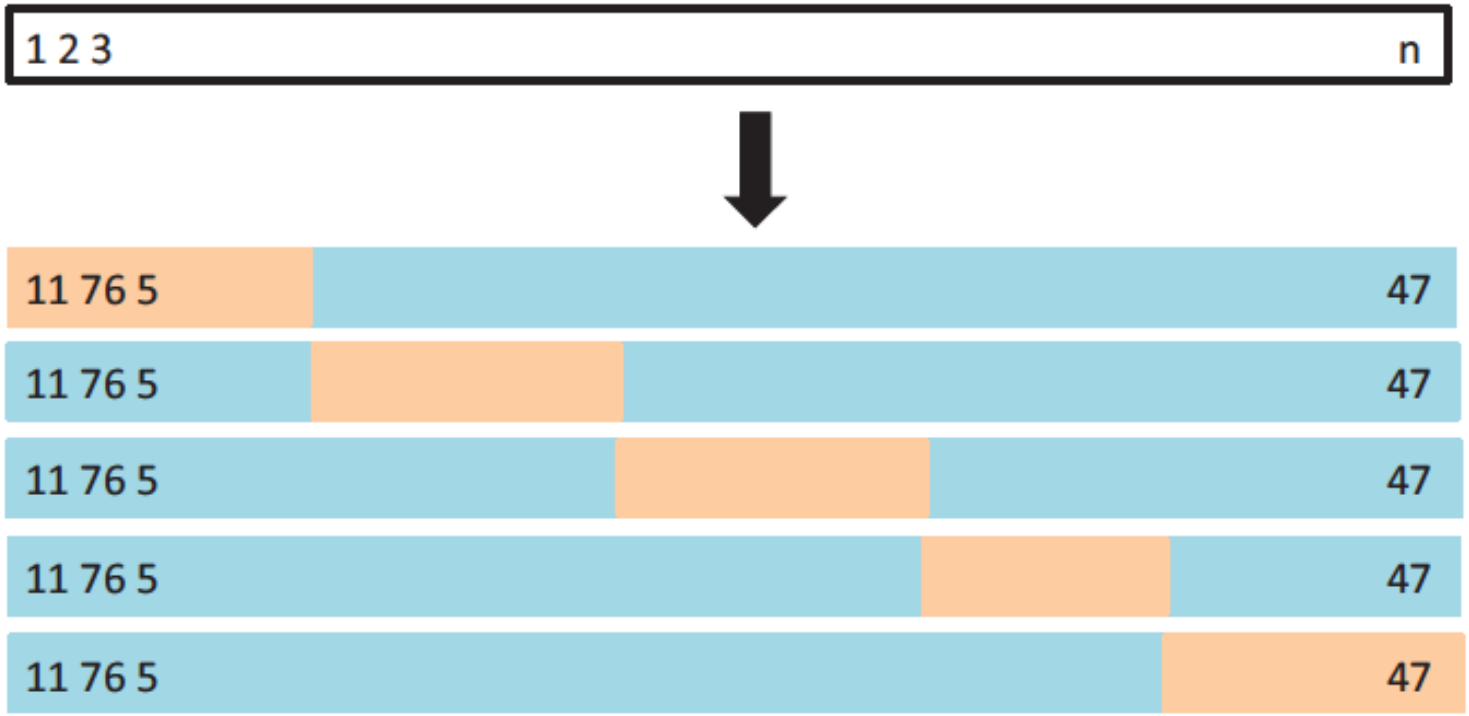


FIGURE 5.5. *A schematic display of 5-fold CV. A set of n observations is randomly split into five non-overlapping groups. Each of these fifths acts as a validation set (shown in beige), and the remainder as a training set (shown in blue). The test error is estimated by averaging the five resulting MSE estimates.*

Fig. from *An Introduction to Statistical Learning* by James et al, 2013.

Let $\kappa : 1, \ldots, N \to 1, \ldots, K$ be an indexing function that indicates the partition to which instance i is allocated by randomization. $f'^{-k}(x)$ denotes the fitted function generated with the kth part of the data removed. Then the cross-validation estimate of Err is

$$\mathrm{CV}(f') = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f'^{-\kappa(i)}(x_i)).$$

The case $K = N$ is called **leave-one-out** cross-validation (LOOCV). In this case $\kappa(i) = i$ and for the ith instance the fit is computed using all the data except the ith.

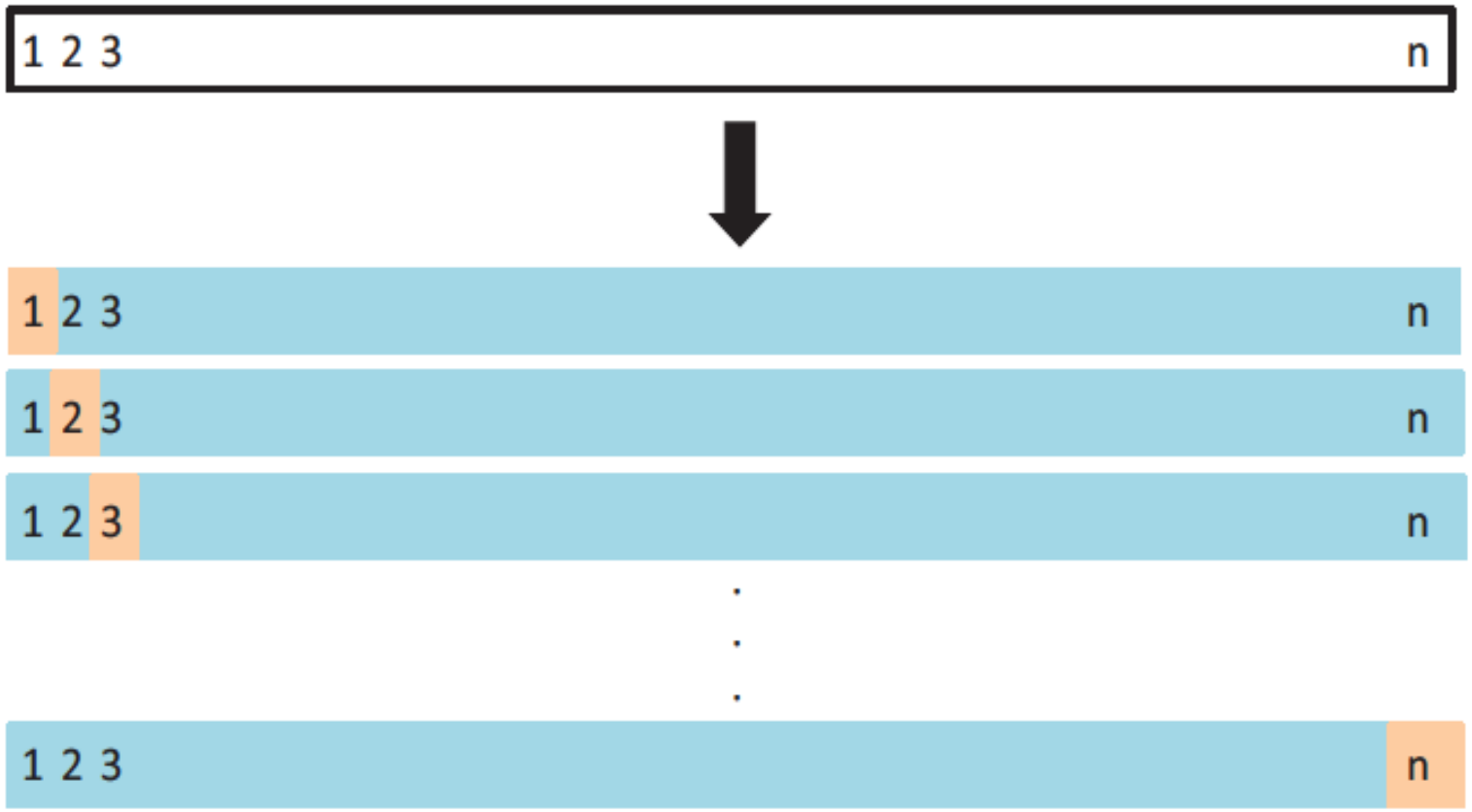


FIGURE 5.3. *A schematic display of LOOCV. A set of n data points is repeatedly split into a training set (shown in blue) containing all but one observation, and a validation set that contains only that observation (shown in beige). The test error is then estimated by averaging the n resulting MSE's. The first training set contains all but observation 1, the second training set contains all but observation 2, and so forth.*

Fig. from *An Introduction to Statistical Learning* by James et al, 2013.

Given a set of models $f(x, \alpha)$ indexed by a tuning parameter α, $f'^{-k}(x, \alpha)$ denotes the αth model generated with the kth part of the data removed. Then for this set of models we define

$$\mathrm{CV}(f', \alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f'^{-\kappa(i)}(x_i, \alpha)).$$

The function $\mathrm{CV}(f', \alpha)$ estimates the test error curve, and we find the tuning parameter $\hat{\alpha}$ that minimizes the test error. The final chosen model is $f(x, \hat{\alpha})$, which is then used to fit all the data.

Which value of *k*?

- LOOCV requires fitting the learning method N times, which may be computationally expensive.
- The test error estimate of LOOCV has lower bias than *k*-fold CV (because it uses more data), but it has higher variance than *k*-fold CV
- *k*=5 or *k*=10 are used as these values have been shown empirically to yield test error estimates that suffer neither from high bias nor from very high variance

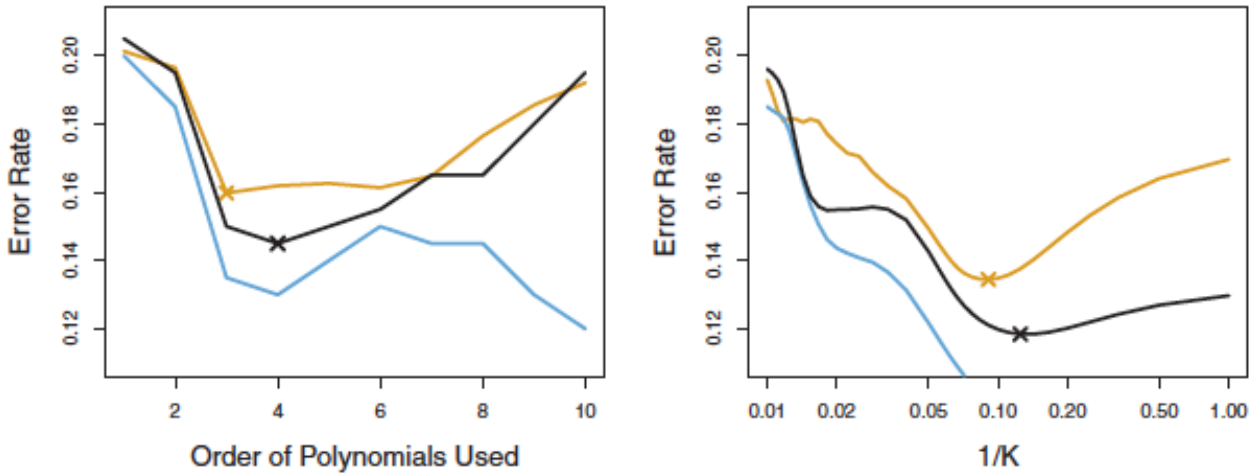


FIGURE 5.8. *Test error (brown), training error (blue), and* 10*-fold CV error (black) on the two-dimensional classification data displayed in Figure 5.7.* Left: *Logistic regression using polynomial functions of the predictors. The order of the polynomials used is displayed on the x-axis.* Right: *The KNN classifier with different values of K, the number of neighbors used in the KNN classifier.*

Fig. from *An Introduction to Statistical Learning* by James et al, 2013.

Which value of *k*?

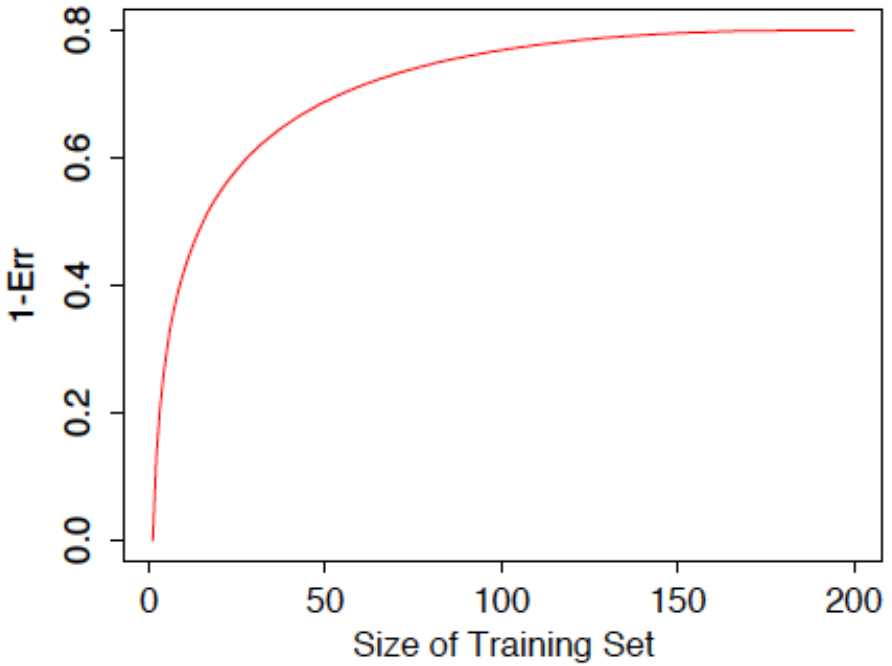


FIGURE 7.8. *Hypothetical learning curve for a classifier on a given task: a plot of* $1 - \text{Err}$ *versus the size of the training set* N*. With a dataset of* 200 *observations,* 5*-fold cross-validation would use training sets of size* 160*, which would behave much like the full set. However, with a dataset of* 50 *observations fivefold cross-validation would use training sets of size* 40*, and this would result in a considerable overestimate of prediction error.*

CV algorithms for model selection

Algorithm 1: parameter tuning with repeated grid-search cross-validation

We have a dataset D which consists of N realisations $(Y, X_1, X_2,..., X_P)$ of one output variable Y and variables $X_1, X_2,..., X_P$. We have at our disposal a regression or classification model building method F with a tuning parameter vector α. We create a grid of K points $\alpha_1, \alpha_2,..., \alpha_K$ and wish to find the optimal value among them. Model F predicts either categories for classification or numbers for regression. We have a loss function loss() as a measure of goodness of fit.

1. Repeat the following process *Nexp* times.
   a. Divide the dataset D pseudo-randomly into V folds
   b. For *I* from 1 to V
      i. Define set L as the dataset D without the *I*-th fold
      ii. Define set T as the *I*-th fold of the dataset D
      iii. For *k* from 1 to K
         1. Build a statistical model $f^k = f(L; \alpha^k)$
         2. Apply f^k on T and store the predictions.
   c. For each α value calculate the goodness of fit (loss()) for all elements in D.
2. For each α value calculate the mean of the *Nexp* calculations of losses.
3. Let α' be the α value for which the average loss is minimal. If there are multiple α values for which the average loss is minimal, then α' is the one with the lowest model complexity.
4. Select α' as the optimal cross-validatory choice for tuning parameter and select statistical model $f' = f(D; \alpha')$ as the optimal cross-validatory chosen model.

Algorithm 3: repeated grid-search cross-validation for variable selection and parameter tuning

1. Repeat the following process *Nexp* times.
   a. Divide the dataset D pseudo-randomly into V folds
   b. For *I* from 1 to V
      i. Define set L as the dataset D without the *I*-th fold
      ii. Define set T as the *I*-th fold of the dataset D
      iii. For *p* from 1 to P
         1. L′ = S(L; *p*); Define set L′ as set L with only *p* selected variables.
         2. Define T′ as set T with only *p* selected variables as in L′.
         3. For *k* from 1 to K
            a. Build a statistical model f′ = f(L′; α^k)
            b. Apply f′ on T′ and store predictions.
   c. For each point in the grid (n, α) calculate loss() for all elements of D.
2. For each point in the grid (n, α) calculate average loss.
3. Define the pair (p′, α′) with minimal average loss as the optimal pair of number of selected variables and parameter values.
4. D′ = S(D; p′); define set D′ as D with only p′ selected predictor variables.
5. Select statistical model f′ = f(D′; α′) as the optimal model.

Note

- Only *unsupervised* (i.e., not involving Y) selection or filtering steps can be done prior to CV
  - For example, removing attributes with zero variance

Model Assessment

- Ideally one would have separate data for assessing the quality of one's model (test data).
- Both training and test data need to be sufficiently large and diverse so that they are representative
- If one has test data then CV only has to be done for model selection

CV algorithm for model selection and assessment
(estimation of prediction error)

Algorithm 2: repeated stratified nested cross-validation

1. Cross-validation protocol *P* is to use *Nexp1* repeated V1-fold cross-validation with a grid of K points $\alpha_1, \alpha_2, \ldots, \alpha_K$. Designate by M the model chosen by application of the cross-validation protocol *P*.

2. Repeat the following process *Nexp2* times.

   a. Stratify the output variable Y.

   b. Divide the dataset D pseudo-randomly into V2 folds making sure that each fold contains the same proportion of each of the Y strata.

   c. For *I* from 1 to V2

      i. Define set L as the dataset D without the *I*-th fold

      ii. Define set T as the *I*-th fold of the dataset D

      iii. Apply the cross-validation protocol to select model f', i.e. use *Nexp1* repeated V1-fold cross-validation with a grid of K points $\alpha_1, \alpha_2, \ldots, \alpha_K$ to find an optimal cross-validatory chosen model f' on dataset L.

      iv. Apply f' on T

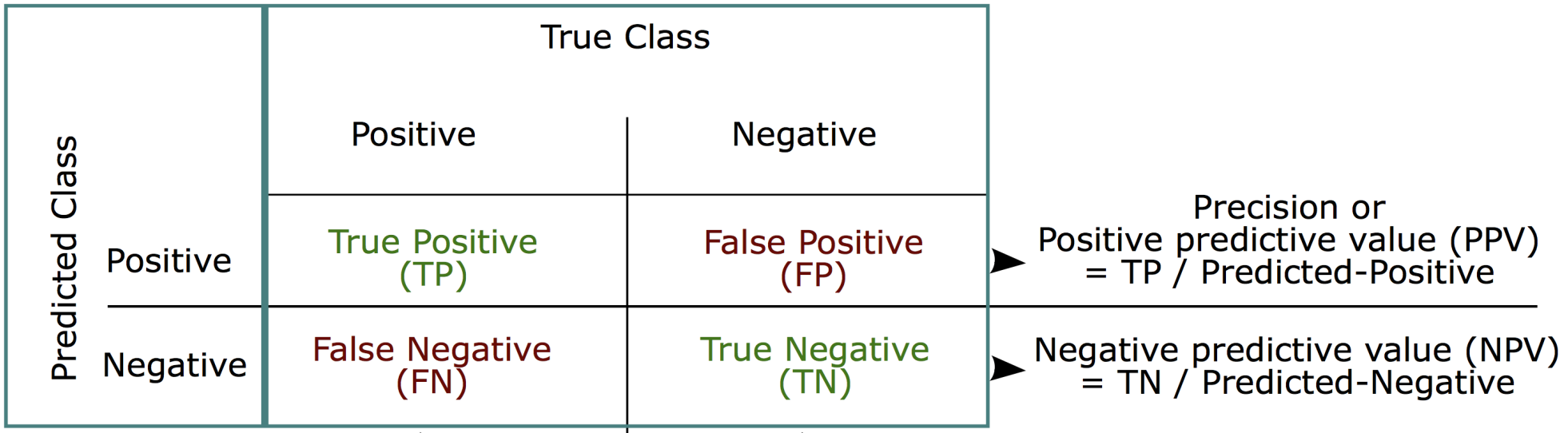
   d. Calculate loss() for all elements of D. We refer to it as the nested cross-validation error.

3. The interval between the minimum and maximum of *Nexp2* nested cross-validation errors is the P-estimated interval of the large-sample error of model M. The mean of *Nexp2* nested cross-validation errors is the P-estimate of the large-sample error of the model M.

Remember

- Validation set was used to choose the best model and it is part of the training set
- Never use the test set for training

Performance Metrics for Classification



Accuracy = (TP + TN) / (P+ N)

F_1 score = 2TP / (2TP + FP + FN)

FDR = FP / P

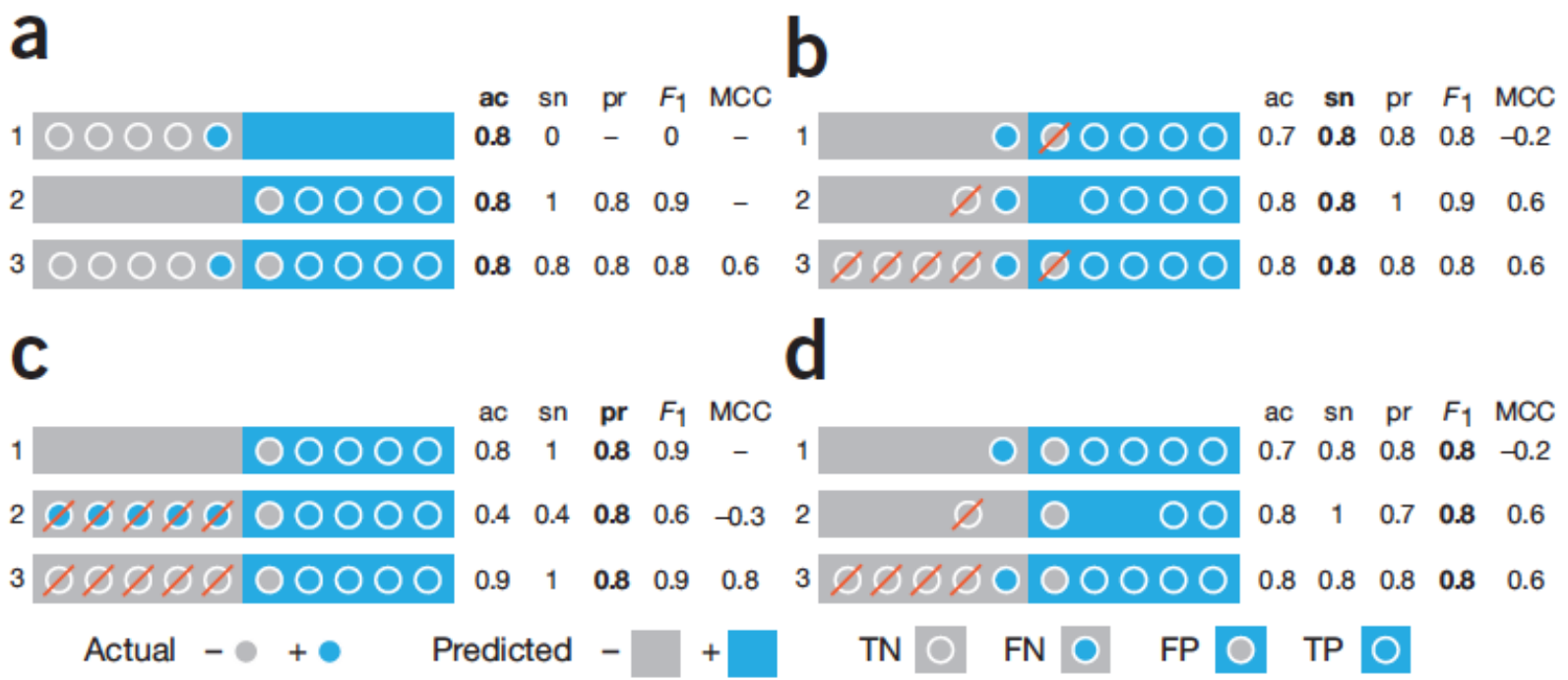


Figure 2 | The same value of a metric can correspond to very different classifier performance. (**a**–**d**) Each panel shows three different classification scenarios with a table of corresponding values of accuracy (ac), sensitivity (sn), precision (pr), F_1 score (F_1) and Matthews correlation coefficient (MCC). Scenarios in a group have the same value (0.8) for the metric in bold in each table: (**a**) accuracy, (**b**) sensitivity (recall), (**c**) precision and (**d**) F_1 score. In each panel, those observations that do not contribute to the corresponding metric are struck through with a red line. The color-coding is the same as in **Figure 1**; for example, blue circles (cases known to be positive) on a gray background (predicted to be negative) are FNs.

Lever, J., Krzywinski, M. & Altman, N. Nat. Methods 13, 541–542 (2016).

Graphical Representation of Performance

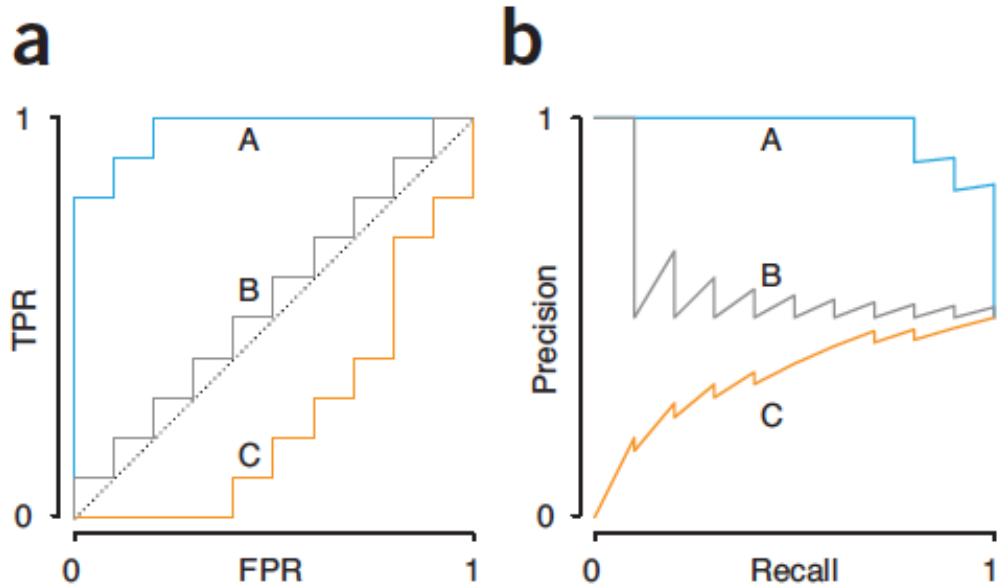


Figure 3 | Graphical evaluation of classifiers. (**a**,**b**) Findings obtained with the (**a**) ROC, which plots the true positive rate (TPR) versus the false positive rate (FPR), and (**b**) PR curves. In both panels, curves depict classifiers that are (A) good, (B) similar to random classification and (C) worse than random. The expected performance of a random classifier is shown by the dotted line in **a**. The equivalent for the PR curve depends on the class balance and is not shown.

- https://www.youtube.com/watch?v=OAl6eAyP-yo

Lever, J., Krzywinski, M. & Altman, N. Nat. Methods 13, 541–542 (2016).

Precision – Recall (PR) curve

- For datasets with a high imbalance between the classes, the AUC may give an overly optimistic view of an algorithm's performance
- For those cases, it is better to use the precision – recall curve

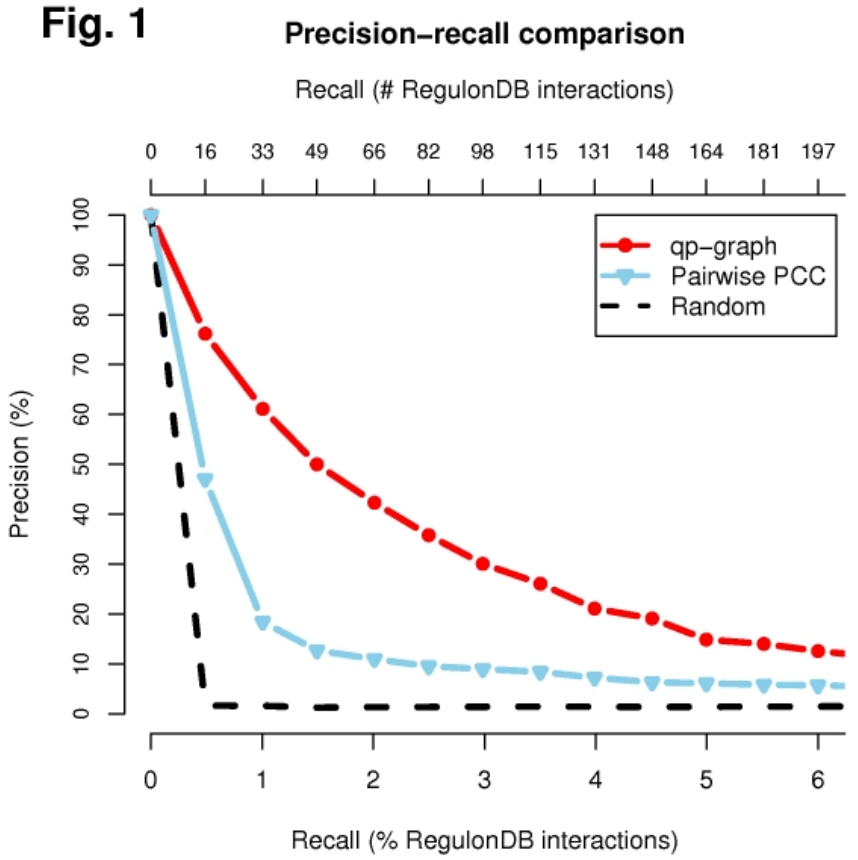


Figure from http://functionalgenomics.upf.edu

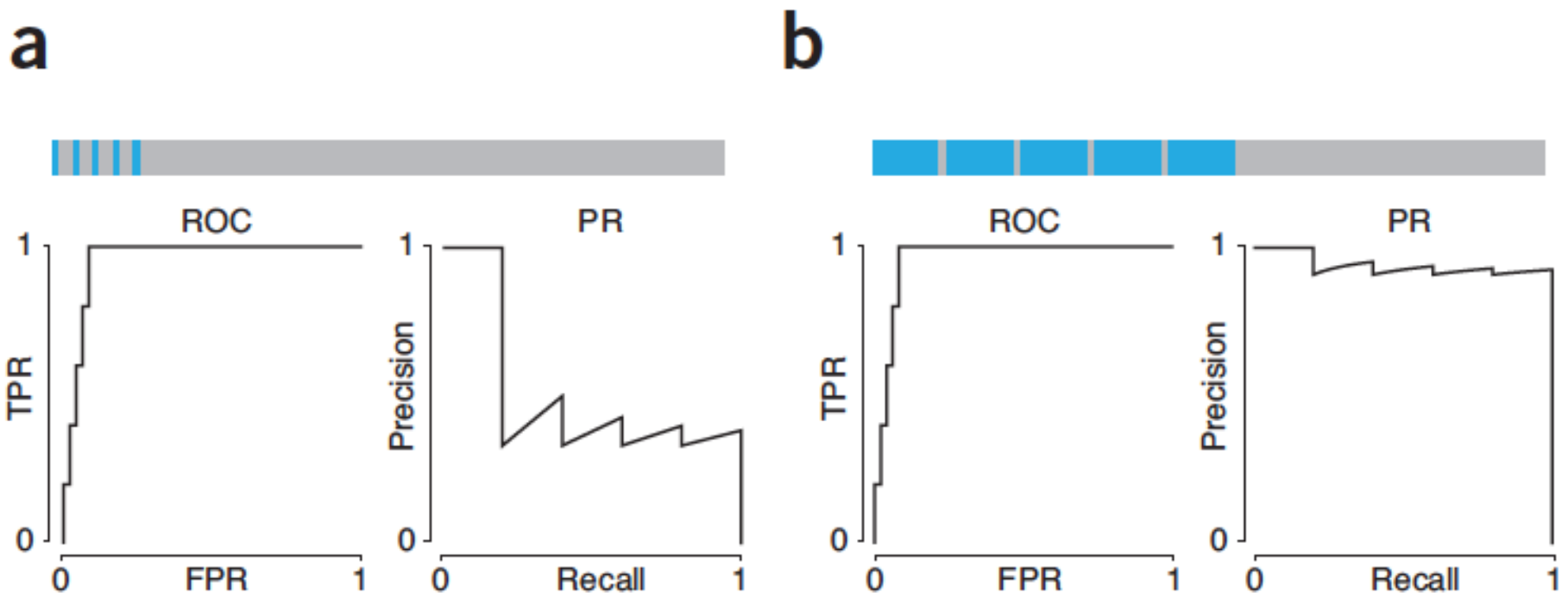


Figure 4 | Graphical representation of classifier performance avoids setting an exact threshold on results but may be insensitive to important aspects of the data. (**a**,**b**) ROC and PR curves for two data sets with very different class balances: (**a**) 5% positive and (**b**) 50% positive observations. For each panel, observations are shown as vertical lines (top), of which 5% or 50% are positive (blue).

Lever, J., Krzywinski, M. & Altman, N. Nat. Methods 13, 541–542 (2016).

Classification Metrics

- Area under the ROC curve (AUC) and the Area under the PR curve (AUPRC) are frequently used as performance metrics and are recommended for classifiers using a confidence threshold
- No single performance metric can distinguish all the strengths and weaknesses of a classifier

Other issues to consider

- Class imbalance
- Intended use of the classifier
- Selection of negative instances

A sample case on assessing predictive performance

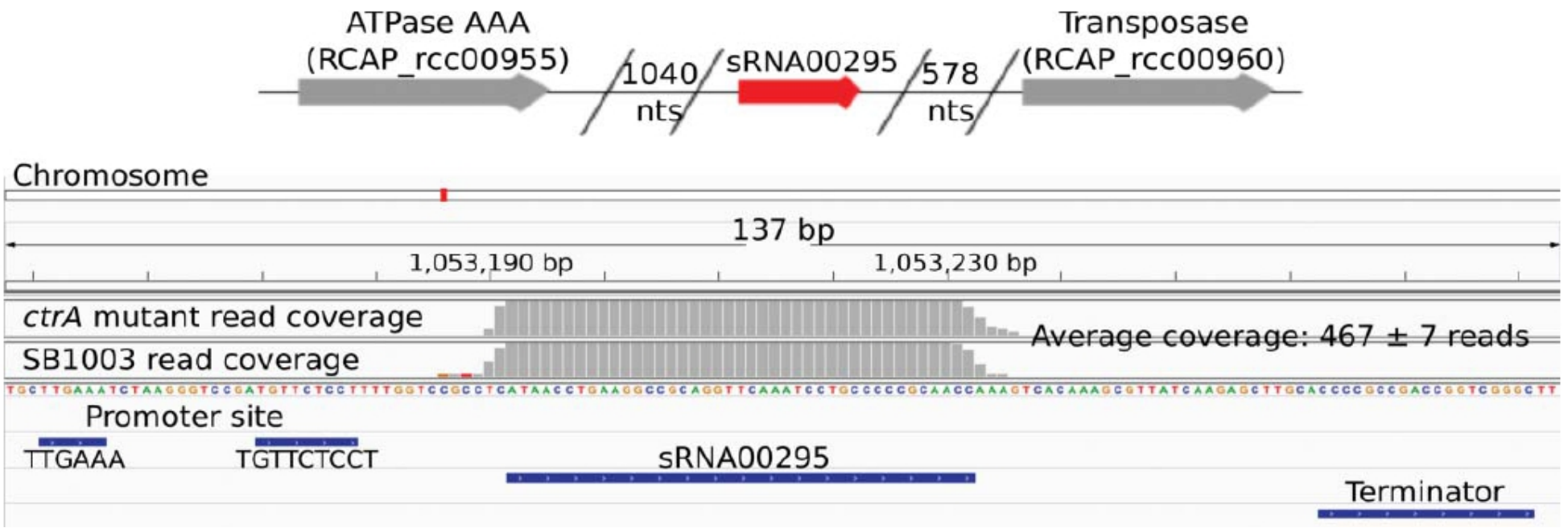
- Bacterial small RNAs (sRNAs) are regulatory non-coding RNAs between 50 and 250 nucleotides long
- sRNAs are involved in the regulation of numerous cellular processes:
  - inhibition/activation of translation
  - degradation/stabilization of mRNA
  - stress responses
  - pathogenicity
  - antibiotic resistance
- Low sequence conservation between bacterial species
- Identified through high-throughput sequencing and validated by wet lab experiments (e.g., Northern blot)

A sample case on assessing predictive performance

- Learning task:
  - to distinguish whether a genomic sequence contains an sRNA or not.
  - For example, does the following sequence encode an sRNA or not?
    CGCCCTATAAACGGGTAATTATACTGACACGGGCGAAGGGGAATTTC
    CTCTCCGCCCGTGCATTCATCTAGGGGCAATTTAAAAAAGA
- To train classifiers for computational prediction of sRNAs, negative instances need to be generated
  - Approach 1: shuffled genome sequences (artificial sequences)
  - Approach 2: random genome sequences (real sequences)

A sample case on assessing predictive performance

- Attributes used:
  - Sequence-based features
    - Frequency of di-nucleotides, tri-nucleotides, etc
  - Genomic-context features



Grüll, et al (2017): Genome-wide identification and characterization of small RNAs in Rhodobacter capsulatus and identification of small RNAs affected by loss of the response regulator CtrA, RNA Biology.

A sample case on assessing predictive performance

Sequence-based features, artificial sequences as negative instances, training data from a single bacterium.

Machine learning method	Best feature sets	Vector length	P(+): N(−)	Sensitivity (%)	Specificity (%)	Accuracy (%)	PPV (%)	MCC	F1 score (%)	AUC
SVM	Tri-nucleotide composition	**64**	**1:2**	**85.11**	**91.58**	**88.35**	**91.21**	**0.77**	**88.06**	**0.937**
Multilayer perceptron	Tri-nucleotide composition	64	1:2	81.87	89.56	85.71	88.69	0.71	85.14	0.908
Random forest	Tri-nucleotide composition	64	1:2	66.48	95.88	81.18	94.16	0.68	77.94	0.927

Table 3. Performance comparison of different machine learning methods. Optimal parameter sets were used for respective methods.

Method	SLT2 Sensitivity (%)	SLT2 Specificity (%)	SLT2 Accuracy (%)
QRNA	59.00	71.00	65.00
Alifoldz	42.00	87.00	64.50
MSARi	2.00	100.00	51.00
zMFold	90.00	49.00	69.50
RNAz2	27.00	98.00	62.50
dynalign	28.00	86.00	57.00
vsFold	25.00	88.00	56.50
Arnedo *et al.*	67.00	78.00	72.50
Proposed	**85.11**	**91.58**	**88.35**

Table 4. Performance measures of the individual methods on SLT2 dataset.

Barman, R. K. et al. An improved method for identification of small non-coding RNAs in bacteria using support vector machine. Sci. Rep. 7, 46070 (2017).

Genomic context features, random genome sequences as negative instances, training data from various bacteria

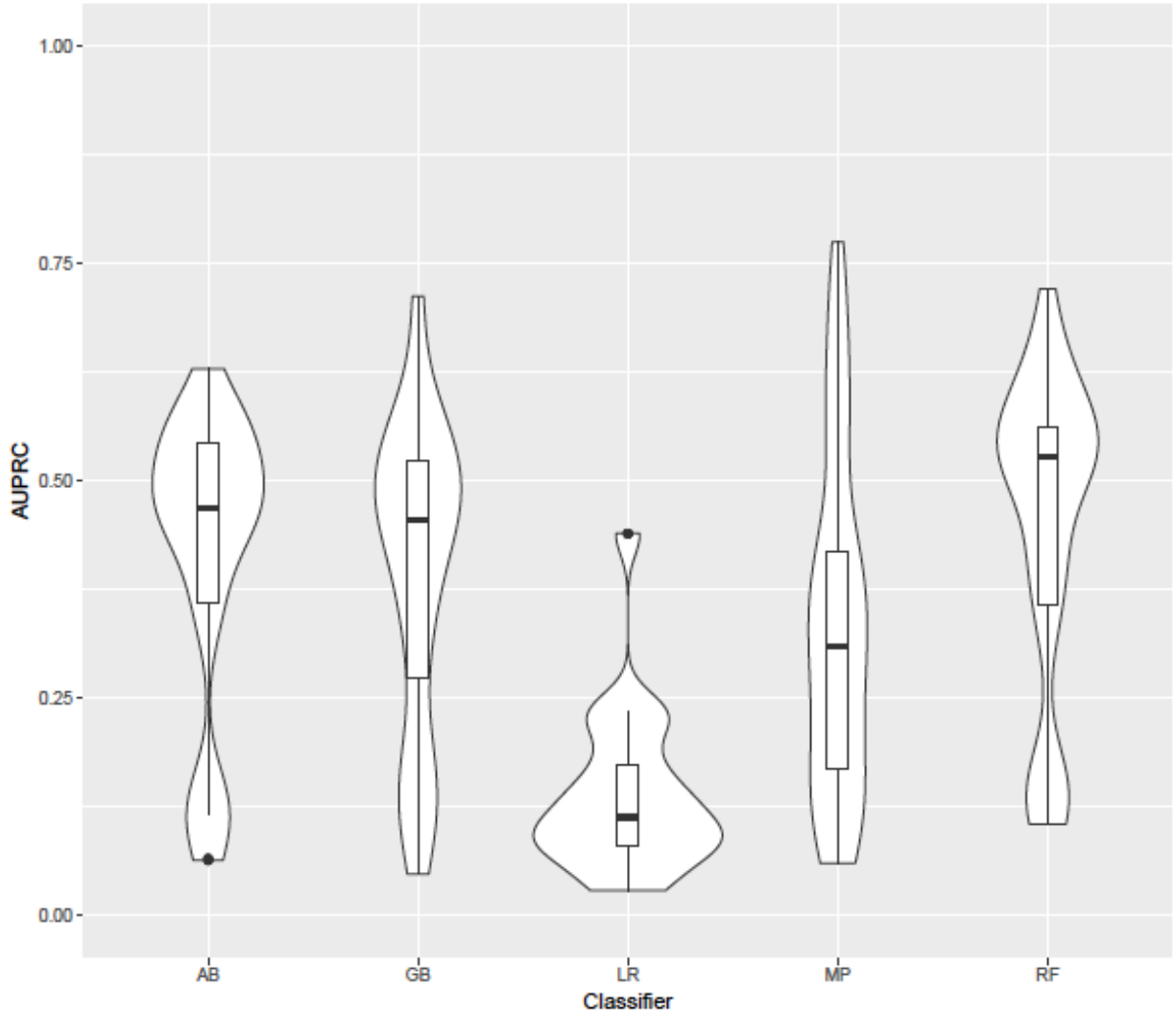


Figure 3 Distribution of AUPRC values per classifier. Violin plots illustrate the distribution of AUPRC values for all models obtained with each classifier. Inside the distribution shape a box indicates the range from the 25 percentile to 75 percentile of the precision values. AB, Adaptive Boosting, GB, Gradient Boosting; LR, Logistic Regression; MP, Multilayer Perceptron; RF, Random Forest.

Eppenhof EJJ, and Peña-Castillo L. 2019. Prioritizing bona fide bacterial small RNAs with machine learning classifiers. PeerJ 7:e6304

Genomic context features, random genome sequences as negative instances, training data from various bacteria

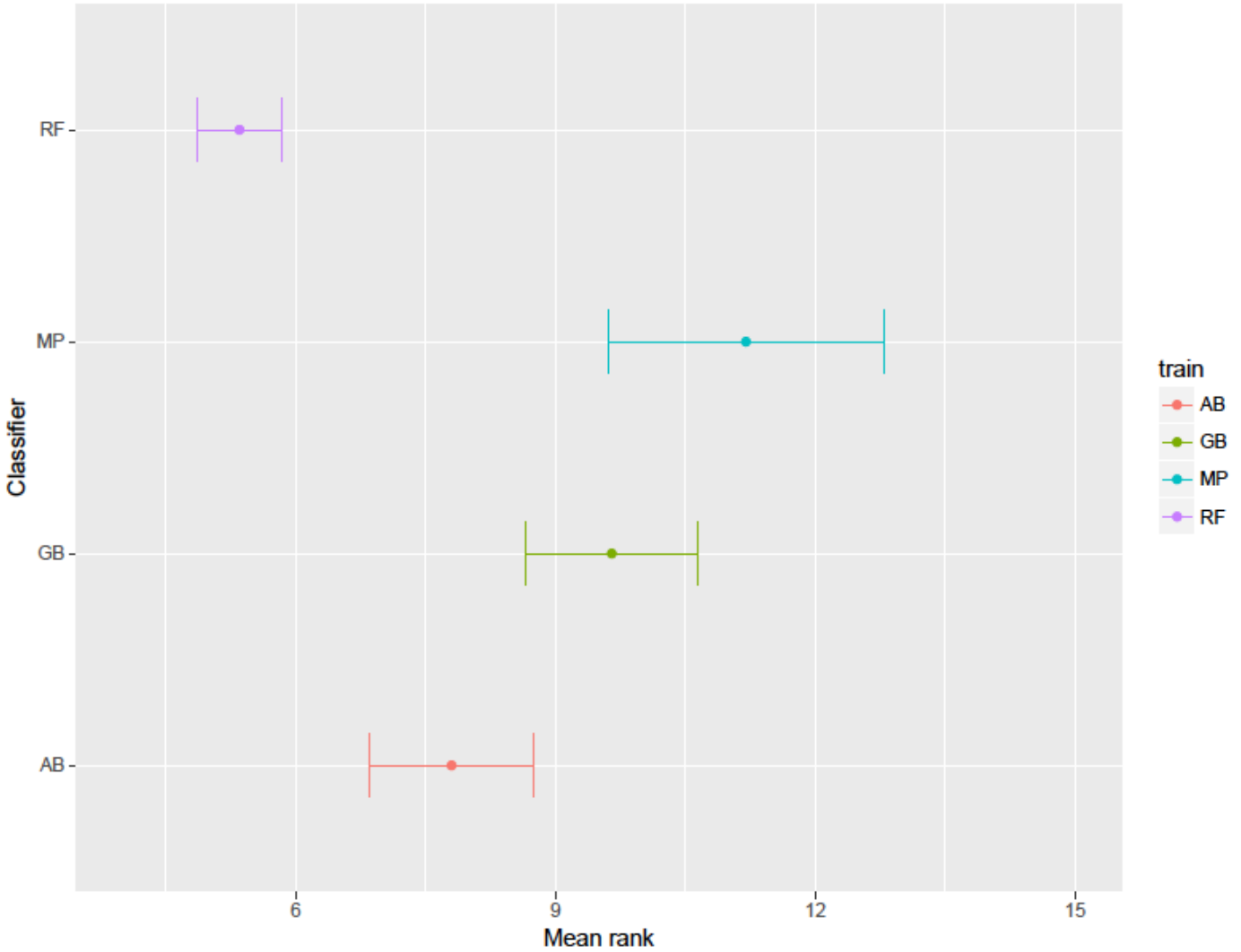


Figure 5 Mean rank per classifier. The dot represents the mean rank and bars represent standard error. Lower ranks indicate better performance in terms of AUPRC. Classifiers are indicated by AB, Adaptive Boosting; GB, Gradient Boosting; MP, Multilayer Perceptron; RF, Random Forest.

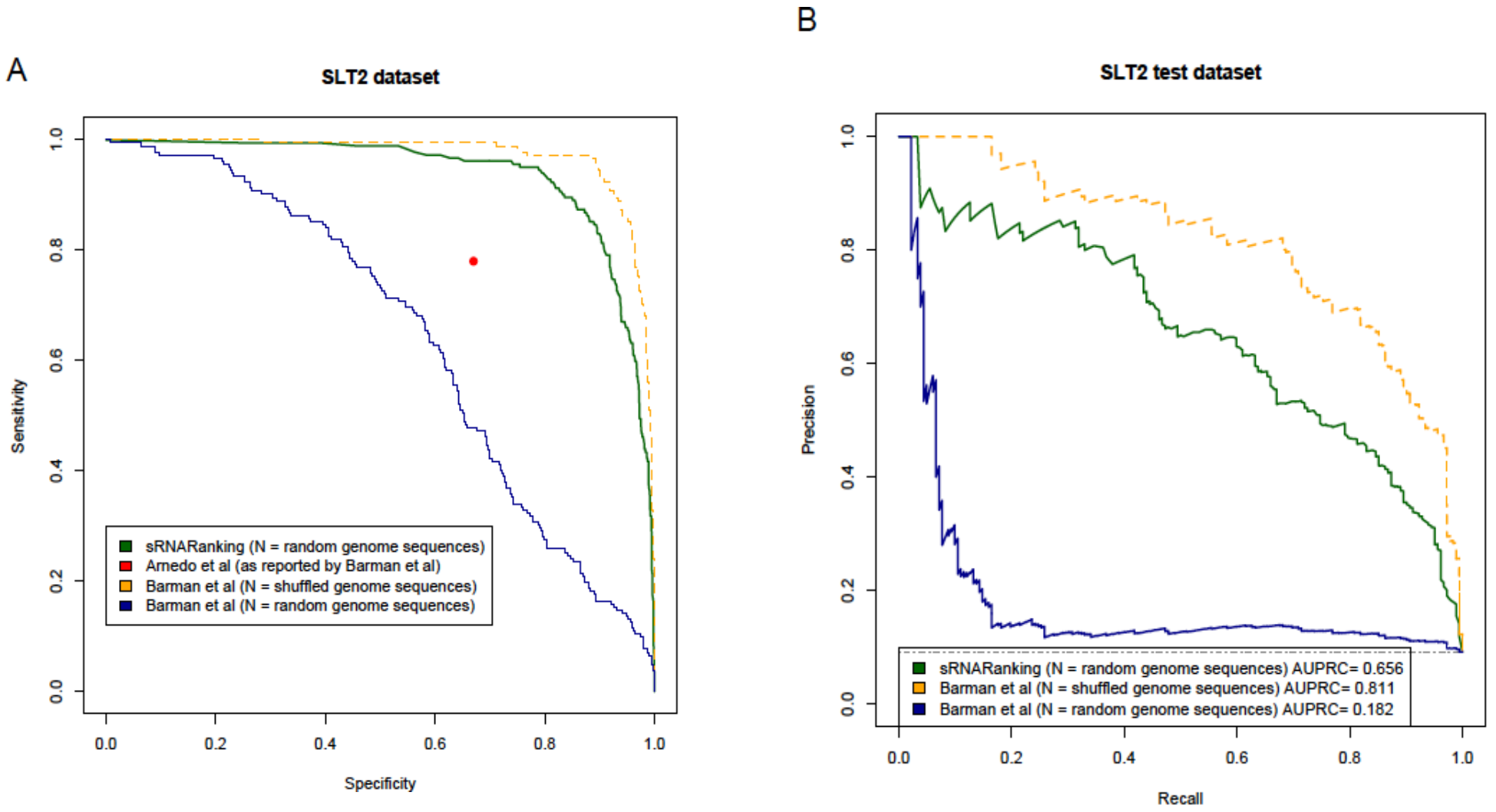


Figure 9 Methods performance on the SLT2 dataset. Barman et al.'s SVM method performance is reported on Barman et al.'s test set with shuffled sequences (artificial sequences) as negative instances and on our test set with random genomic sequences (real genomic sequences) as negative instances. (A) Sensitivity-Specificity curve. Arnedo et al.'s approach reported sensitivity and specificity is indicated with a red dot. (B) Precision-Recall curve. The horizontal dashed line indicates the performance of a random classifier.

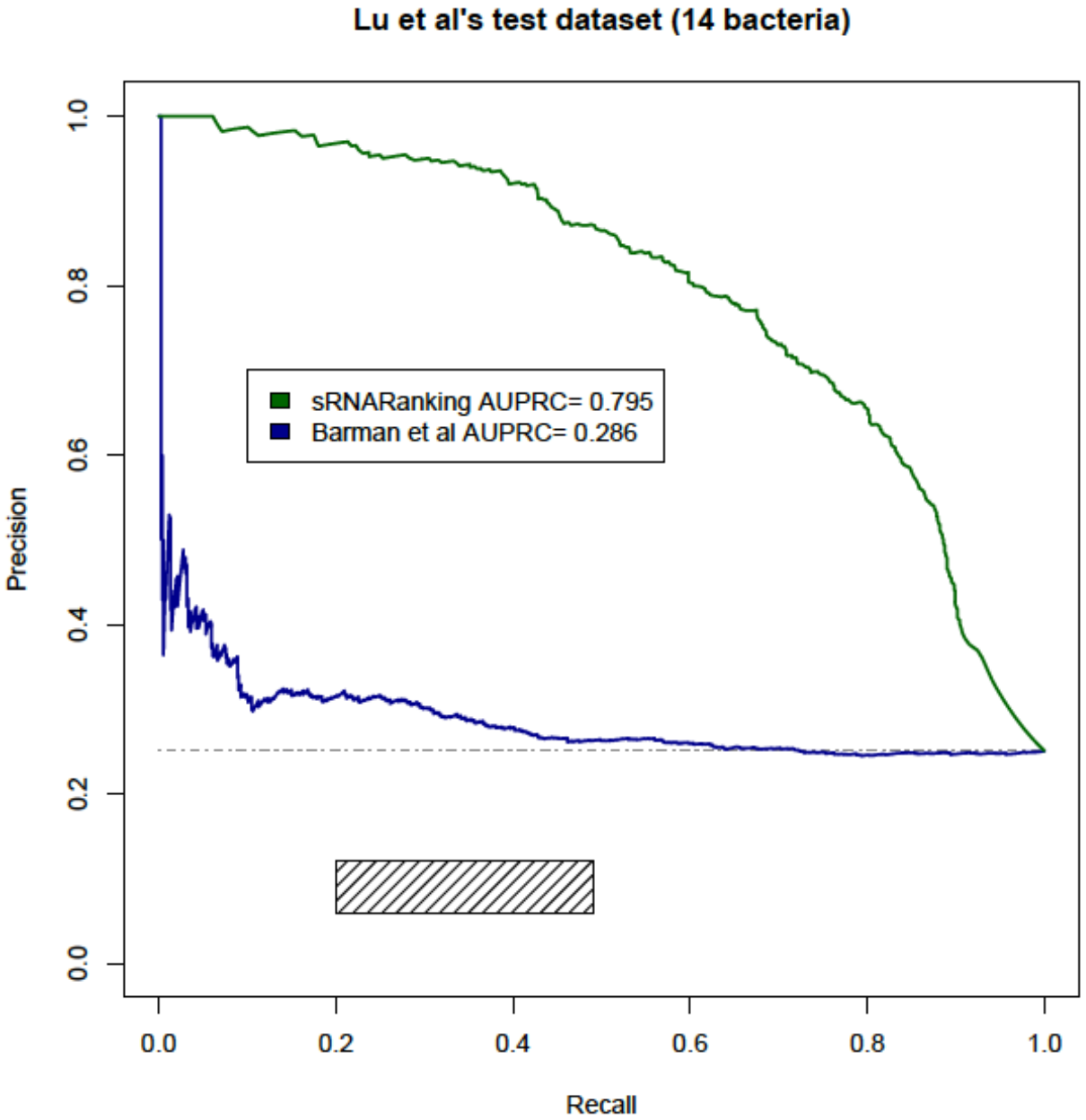


Figure 8 PRC of sRNARanking and Barman et al.'s SVM method performance on Lu et al.'s multi-species dataset. The four approaches assessed by Lu et al. achieved recall of 0.20 to 0.49 with precision of 0.06 to 0.12. The corresponding area in the PRC is indicated by the rectangle. The horizontal dashed line indicates the performance of a random classifier.

By now, you should be able to...

- explain why we need to assess generalization performance
- define generalization error, expected predicted error, and training error
- mention typical loss functions for regression and classification
- explain why the training error is not a good estimate of the test error
- describe the goal of model selection and model assessment
- understand the validation set approach and its drawbacks
- describe k-fold cross-validation (CV) and be able to implement it
- avoid common pitfalls when performing model selection
- calculate and interpret various performance metrics for classification
- obtain and interpret graphical representations of classification performance
- appreciate that predictive performance depends on multiple factors
- be aware of what classification metric show and hide