Apprendimento Automatico

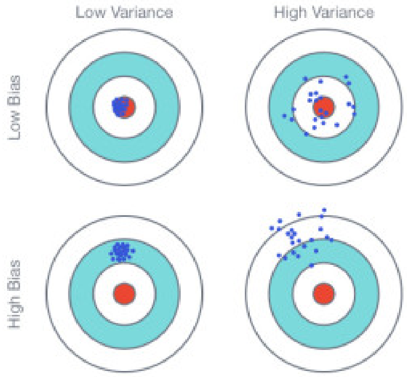
Practical Issues

Fabio Aiolli

4 Dicembre 2017

Bias and Variance

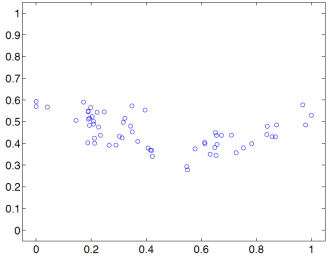
- The BIAS measures the *distortion* of an estimate
- The VARIANCE measures the *dispersion* of an estimate



Some practical issues

Underfitting/Overfitting and learning parameters

- Suppose we have some data (60 points) that we want to fit a curve to



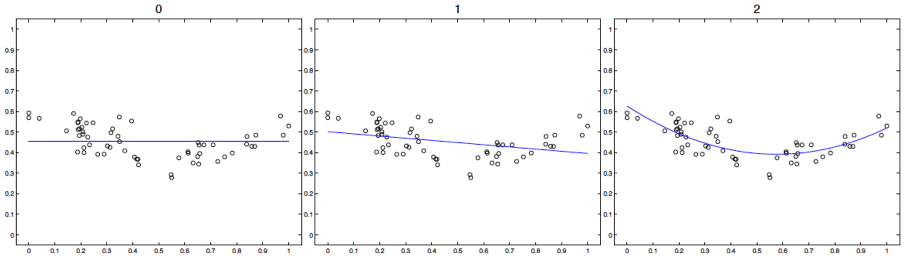
- Let fit a polynomial, of the form

$$y = w_0 + w_1 x + w_2 x^2 + \ldots + w_p x^p$$

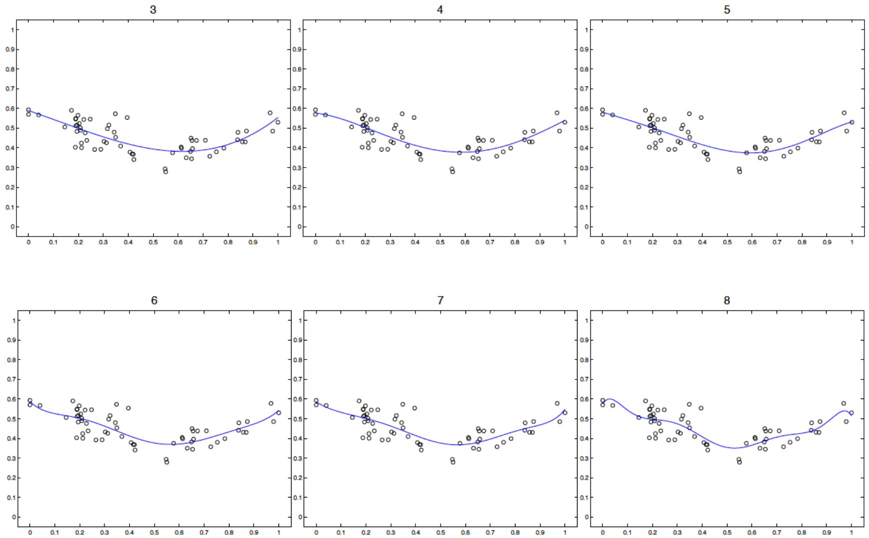
Some practical issues

Underfitting/Overfitting and learning parameters

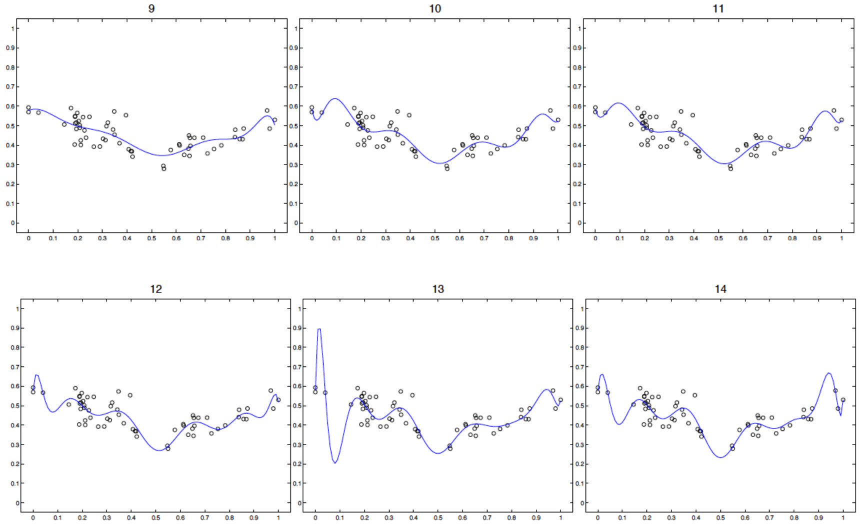
- How to choose p ? (Hypothesis Space)
- For various p, we can find and plot the best polynomial, in terms of minimizing the Empirical Error (Mean Squared Error in this case)
- Here are the solutions for different values of p



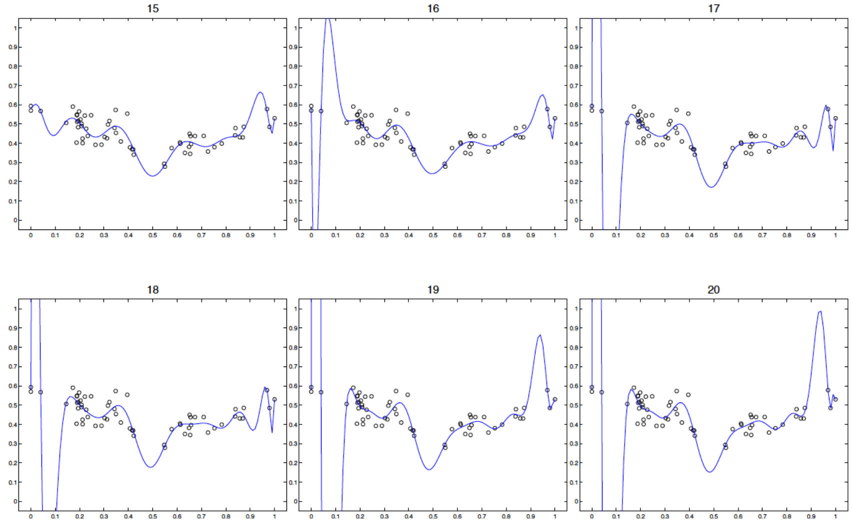
Some practical issues



Some practical issues



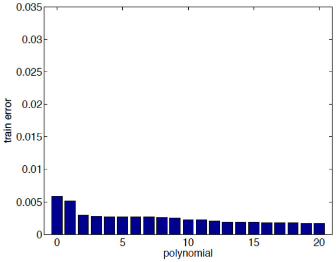
Some practical issues

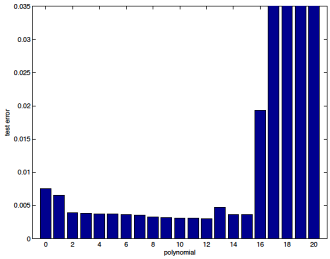


Some practical issues

Underfitting/Overfitting and learning parameters

- Here is a summary of the Empirical Error ... and the Empirical Error over some new TEST data (100,000 extra points) from the same distribution, as a function of p:





Some practical issues

Underfitting/Overfitting and learning parameters

- For very low p, the model is very simple, and so cannot capture the full complexities of the data (Underfitting! also called **bias**)
- For very high p, the model is complex, and so tends to overfit to spurious properties of the data (Overfitting! also called **variance**)

Unfortunately we cannot use the test set to pick up the right value of p!

$\boxed{\text{PRACTICAL PROBLEM: how can we use the training set to set } p \text{ ?}}$

Model Selection and Hold-out

We can hold out some of our original training data

Hold-out procedure

1. A small subset of Tr, called the validation set (or hold-out set), denoted Va, is identified
2. A classifier/regressor is learnt using examples in $Tr - Va$
3. Step 2 is performed with different values of the parameter(s) (in our example, p), and tested against the hold-out sample

In an operational setting, after parameter optimization, one typically re-trains the classifier on the entire training corpus, in order to boost effectiveness (debatable step!)

It is possible to show that the evaluation performed in Step 2 gives an unbiased estimate of the error performed by a classifier learnt with the same parameter(s) and with training set of cardinality $|Tr| - |Va| < |Tr|$

K-fold Cross Validation

An alternative approach to model selection (and evaluation) is the K-fold cross-validation method

K-fold CV procedure

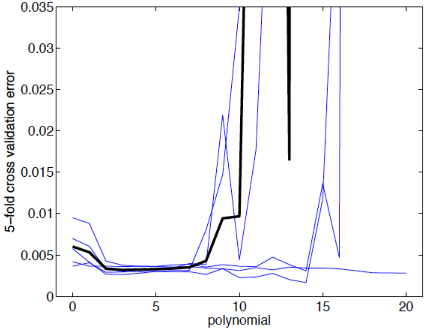
1. K different classifiers/regressors $h_1, h_2, \dots, h_k$ are built by partitioning the initial corpus Tr into k disjoint sets $Va_1, \dots, Va_k$ and then iteratively applying the Hold-out approach on the k-pairs $(Tr_i = Tr - Va_i, Va_i)$
2. Final error is obtained by individually computing the errors of $h_1, \dots, h_k$, and then averaging the individual results

The above procedure is repeated for different values of the parameter(s) and the setting (model) with smaller final error is selected

The special case $k = |Tr|$ of k-fold cross-validation is called **leave-one-out** cross-validation

Back to our example

- Let's apply 5-fold CV



- Minimum error reached for $p = 3$, rather than the optimal $p = 12$
- Clearly, cross validation is no substitute for a large test set. However, if we only have a limited training set, it is often the best option available.

Back to our example

Why cross-validation selected a simpler model than optimal ?

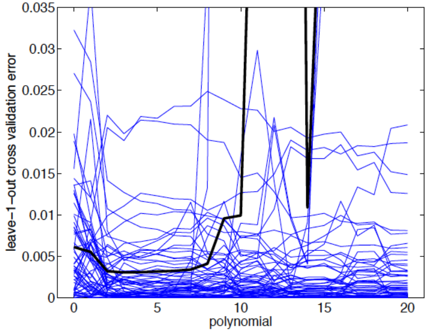
- Notice that with 60 points, 5-fold cross validation effectively tries to pick the polynomial that makes the best bias-variance tradeoff for 48 ($60 * \frac{4}{5}$) points
- With 10-fold cross validation, it would instead try to pick the best polynomial for 54 ($60 * \frac{9}{10}$) points

Thus, cross validation biases towards simpler models

- **leave-one-out** cross-validation reduces this tendency to the minimum possible by doing 60-fold cross validation

Back to our example

- So let's try **leave-one-out** cross-validation



- We still get $p = 3$!
- Cross validation is a good technique, but it doesn't work miracles: there is only so much information in a small dataset.

Analysis of Cross Validation

- What happens varying k?
- For highest k's we have larger training sets, hence less bias! Smaller validation sets, hence more variance!
- For lower k's we have smaller training sets, hence more bias! Larger validation sets, hence less variance!

In general...

Almost all learning algorithms have (hyper)parameters!

- Support Vector Machines: C, type of kernel (polynomial, RBF, etc.), kernel parameter (degree of polynomial, width of RFB, etc.)
- Neural Networks: nonlinear/linear neurons, number of hidden units, η, other learning parameters we have not discussed (e.g., momentum μ)

Hold-out or Cross-Validation can be used to select the "optimal" values for the (hyper)parameters (i.e., select the "optimal" model).

After model selection, the training set is used to evaluate the goodness of the selected model

Evaluation for unbalanced data - Beyond accuracy

Classification accuracy:

- Very common in ML,
- Proportion of correct decisions,
- Not appropriate when the number of positive examples is much lower than the number of negative examples (or viceversa)
- Precision, Recall and F_1 are better in these cases!

Contingency table

	Relevant	Not Relevant
Retrieved	True Positive (TP)	False Positive (FP)
Not Retrieved	False Negative (FN)	True Negative (TN)

$$\pi = \frac{\mathsf{TP}}{\mathsf{TP}+\mathsf{FP}} \qquad \rho = \frac{\mathsf{TP}}{\mathsf{TP}+\mathsf{FN}}$$

why not using the accuracy $\alpha = \frac{\mathsf{TP}+\mathsf{TN}}{\mathsf{TP}+\mathsf{TN}+\mathsf{FP}+\mathsf{FN}}$?

Effectiveness for Unbalanced Classification

If relevance is assumed to be binary-valued, effectiveness is typically measured as a combination of

- Precision is the “degree of soundness” of the system:
  $P(\text{RELEVANT}|\text{RETURNED})$
- Recall is the “degree of completeness” of the system:
  $P(\text{RETURNED}|\text{RELEVANT})$

F Measure

How can one trade-off between precision and recall?

F-measure (weighted harmonic mean of the precision and the recall)

$$F_\beta = \frac{(1+\beta^2)\pi\rho}{\beta^2\pi + \rho}$$

$\beta < 1$ emphasizes precision

$$\beta = 1 \;\Rightarrow\; F_1 = 2\frac{\pi\rho}{\pi + \rho}$$

From Binary to Multiclass

Multiclass classification means a classification task with more than two classes; e.g., classify a set of images of fruits which may be oranges, apples, or pears. Multiclass classification makes the assumption that each sample is assigned to one and only one label: a fruit can be either an apple or a pear but not both at the same time.

How can the multiclass problem be reduced to a set of binary problems?

One vs Rest

The one-vs-rest strategy, is implemented in `OneVsRestClassifier` in `sklearn`. The strategy consists in fitting one classifier per class.

For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency (only `n_classes` classifiers are needed), one advantage of this approach is its interpretability. Since each class is represented by one and only one classifier, it is possible to gain knowledge about the class by inspecting its corresponding classifier. This is the most commonly used strategy and is a fair default choice.

One vs One

The one-vs-one strategy, is implemented in `OneVsOneClassifier` in `sklearn`. The strategy consists in fitting one classifier for each pair of classes.

At prediction time, the class which received the most votes is selected.

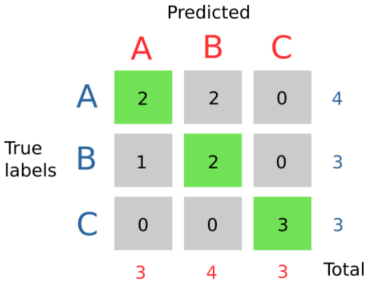
Since it requires to fit `n_classes * (n_classes - 1) / 2` classifiers, this method is usually slower than one-vs-the-rest. However, this method may be advantageous for algorithms such as kernel algorithms which don't scale well with n_samples. This is because each individual learning problem only involves a small subset of the data whereas, with one-vs-the-rest, the complete dataset is used `n_classes` times.

Evaluation in Multiclass classification

A very intuitive way to show the results of a multiclass predictor is using the confusion matrix.

Example:

```
Prediction :   A B A B C C C A B
True labels:   A A A A C C C B B
```



		Predicted			
		A	B	C	
True labels	A	2	2	0	4
	B	1	2	0	3
	C	0	0	3	3
		3	4	3	Total

Precision and Recall for multiclass problems

Precision can be calculated separately for each class. For each row, we take the number on the diagonal, and divide it by the sum of all the elements in the column.

```
prec_A: 2/3 = 0.67
prec_B: 2/4 = 0.50
prec_C: 3/3 = 1.00
```

Recall can be calculated separately for each class. It is the value on the diagonal, divided by the sum of the values in the row.

```
recall_A: 2/4 = 0.50
recall_B: 2/3 = 0.67
recall_C: 3/3 = 1.00
```

Micro and Macro Averaging

To extract a single number from the class precision, recall or F1-score, it is common to adopt two different kinds of average.

The first, is to compute the score separately for each class and then taking the average value this is called macro averaging. This kind of averaging is useful when the dataset is unbalanced and we want to put the same emphasis on all the classes.

$$\frac{r_A + r_B + r_C}{3} = 0.7233$$

The second is to calculate the measure from the grand total of the numerator and denominator, this is called micro averaging and is useful when you want to bias your classes towards the most populated class. For our recall example:

$$\frac{2+2+3}{4+3+3} = 0.636$$