Data Mining

Part 4. Prediction

4.8. Credibility of a Predictor

Fall 2009

Instructor: Dr. Masoud Yaghini

Outline

- Training and Testing Data Sets
- Predicting Performance
- Cross Validation
- Comparing Data Mining Methods
- Evaluating Numeric Prediction
- References

Training and Testing Data Sets

Evaluation: the key to success

- Error on the training data is *not* a good indicator of performance on future data
  - Otherwise 1NN would be the optimum classifier!
- Simple solution that can be used:
  - Split data into training and test set
- Statistical reliability of estimated differences in performance (-> significance tests)

Issues in evaluation

- Choice of performance measure:
  - Number of correct classifications
    - e.g. decision tree
  - Accuracy of probability estimates
    - e.g. in Naïve Bayesian Classification
  - Error in numeric predictions
    - e.g. in regression analysis

Training and Testing

- Natural performance measure for classification models: *error rate*
  - **Success**: instance's class is predicted correctly
  - **Error**: instance's class is predicted incorrectly
  - **Error rate**: proportion of errors made over the whole set of instances
- Resubstitution error*:* error rate obtained from training data

Training and testing

- **Test set**: independent instances that have played no part in formation of predictor
  - Assumption: both training data and test data are representative samples of the underlying problem
- Test and training data may differ in nature
  - Example: classifiers built using customer data from two different towns *A* and *B*
    - To estimate performance of classifier from town *A* in completely **new town**, test it on data from *B*

Note on parameter tuning

- It is important that the test data is not used in any way to create the classifier
- Some learning schemes operate in two stages:
  - **Stage 1**: build the basic structure
  - **Stage 2:** optimize parameter settings
- The test data can't be used for parameter tuning!

Note on parameter tuning

- Proper procedure uses three sets:
  - **Training data**: is used to build the basic structure
  - **Validation data**: is used to optimize parameters or to select a particular method
  - **Test data**: is used to calculate the error rate of the final method

Making the most of the data

- Once evaluation is complete, **all the data** can be used to build the final classifier
- Generally,
  - The larger the training data the better the classifier
  - The larger the test data the more accurate the error estimate
- **Holdout procedure**: method of splitting original data into training and test set
  - Ideally both training set *and* test set should be large!

Predicting Performance

Predicting performance

- Assume the estimated error rate is 25%.
- How close is this to the true error rate?
- To answer these questions, we need some statistical reasoning.
  - Depends on the amount of test data

Confidence intervals

- Suppose p is success rate, that out of N trials, S are successes: thus the observed success rate is $f = S/N$
- We can say: p lies within a certain specified interval with a certain specified confidence
- Example: S=750 successes in N=1000 trials
  - Estimated success rate: 75%
  - How close is this to true success rate p?
    - Answer: with 80% confidence p in [73.2,76.7]
- Another example: S=75 and N=100
  - Estimated success rate: 75%
  - With 80% confidence p in [69.1,80.1]

Cross Validation

Holdout Estimation

- **Holdout method**
  - The **holdout method** reserves a certain amount for testing and uses the remainder for training
  - Usually: **one third** for testing, the rest for training
- **Problem**: the samples might not be representative
  - Example: class might be missing in the test data
- **Stratified Holdout Method**
  - Ensures that each class is represented with approximately equal proportions in both subsets

Repeated holdout method

- **Repeated holdout method**
  - Holdout estimate can be made more reliable by repeating the process with different subsamples
  - In each iteration, a certain proportion is randomly selected for training
  - The error rates on the different iterations are averaged to yield an overall error rate
  - This is called the **repeated holdout method**
- **Still not optimum**: the different test sets overlap
  - Can we prevent overlapping?

Cross Validation

- **Cross-validation method**
  - **Cross-validation** avoids overlapping test sets
  - **First step**: split data into *k* subsets of equal size
  - **Second step**: use each subset in turn for testing, the remainder for training
  - Called **k-fold cross-validation**
  - Often the subsets are stratified before the cross-validation is performed
  - The error estimates are averaged to yield an overall error estimate

More on cross-validation

- **Standard method for evaluation**:
  - stratified ten-fold cross-validation
- Why ten?
  - Extensive experiments have shown that this is the best choice to get an accurate estimate
  - There is also some theoretical evidence for this
- Stratification reduces the estimate's variance
- **Repeated stratified cross-validation**
  - Even better
  - E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

Comparing Data Mining Methods

Comparing data mining methods

- **Frequent question**: which of two learning methods performs better?
  - this is domain dependent!
  - **Obvious way**: compare 10-fold CV estimates
- How about, when a new learning algorithm is proposed?
  - Need to show that a particular method works really better

Comparing data mining methods

- Want to show that method A is better than method B in a particular domain
  - For a given amount of training data
  - On average, across all possible training sets
- Let's assume we have an infinite amount of data from the domain:
  - Sample infinitely many dataset of specified size
  - Obtain cross-validation estimate on each dataset for each method
  - Check if mean accuracy for method A is better than mean accuracy for method B

Paired t-test

- In practice we have limited data and a limited number of estimates for computing the mean
- *Student's t-test* tells whether the means of two samples are significantly different
- In our case the **samples are cross-validation** estimates for different datasets from the domain
- Use a *paired* t-test because the individual samples are paired
  - The same CV is applied twice

Evaluating Numeric Prediction

Evaluating numeric prediction

- Strategies: independent test set, cross-validation, significance tests, etc.
- Difference: error measures
- Actual target values: $a_1\ a_2 \ldots a_n$
- Predicted target values: $p_1\ p_2 \ldots\ p_n$
- Most popular measure: **mean-squared error**

$$\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n}$$

  - Easy to manipulate mathematically

Other measures

- The **root mean-squared error**:

$$\sqrt{\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{n}}$$

- The **mean absolute error**:

$$\frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{n}$$

  - is less sensitive to outliers than the mean-squared error:

Improvement on the mean

- How much does the scheme improve on simply predicting the average?

- The *relative squared error* is:

$$\frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(a_1 - \bar{a})^2 + \ldots + (a_n - \bar{a})^2}, \text{ where } \bar{a} = \frac{1}{n}\sum_i a_i$$

- The *relative absolute error* is:

$$\frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{|a_1 - \bar{a}| + \ldots + |a_n - \bar{a}|}$$

Correlation coefficient

- Measures the *statistical correlation* between the predicted values and the actual values

$$\frac{S_{PA}}{\sqrt{S_P S_A}}, \text{ where } S_{PA} = \frac{\sum_i (p_i - \overline{p})(a_i - \overline{a})}{n-1},$$

$$S_p = \frac{\sum_i (p_i - \overline{p})^2}{n-1}, \text{ and } S_A = \frac{\sum_i (a_i - \overline{a})^2}{n-1}$$

- Scale independent, between –1 and +1
- Good performance leads to large values!

Which measure?

- Best to look at all of them
- Often it doesn't matter
- Example: Performance measures for four numeric prediction models

	A	B	C	D
root mean-squared error	67.8	91.7	63.3	57.4
mean absolute error	41.3	38.5	33.4	29.2
root relative squared error	42.2%	57.2%	39.4%	35.8%
relative absolute error	43.1%	40.1%	34.8%	30.4%
correlation coefficient	0.88	0.88	0.89	0.91

References

References

- I. H. Witten and E. Frank, **Data Mining: Practical Machine Learning Tools and Techniques**, 2nd Edition, Elsevier Inc., 2005. (Chapter 5)

The end