Evaluation
and Credibility

How much should we believe in what was learned?

Branko Kavšek
branko.kavsek@upr.si

University of Primorska

Faculty of Mathematics, Natural Sceinces and Information Technologies

Outline

- Introduction
- Classification with Train, Test, and Validation sets
  - Handling Unbalanced Data
  - Parameter Tuning
- Cross-validation
- Comparing Data Mining Schemes

Introduction

- How predictive is the model we learned?
- Error on the training data is ***not*** a good indicator of performance on future data
  - ☐ ***Q: Why?***
  - ☐ A: Because new data will probably not be **exactly** the same as the training data!
- Overfitting – fitting the training data too precisely – usually leads to poor results on new data

© Branko Kavšek, 2023/24

Evaluation issues

- Possible evaluation measures:
  - ☐ Classification Accuracy
  - ☐ Total cost/benefit – when different errors involve different costs
  - ☐ Lift and ROC curves
  - ☐ Error in numeric predictions
- How reliable are the predicted results?

Classifier error rate

- Natural performance measure for classification problems: ***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
- *Training set error rate:* is way too optimistic!
 - □ you can find patterns even in random data

Evaluation on "LARGE" data, 1

If many (thousands) of examples are available, including several hundred examples from each class, then how can we evaluate our classifier method?

© Branko Kavšek, 2023/24
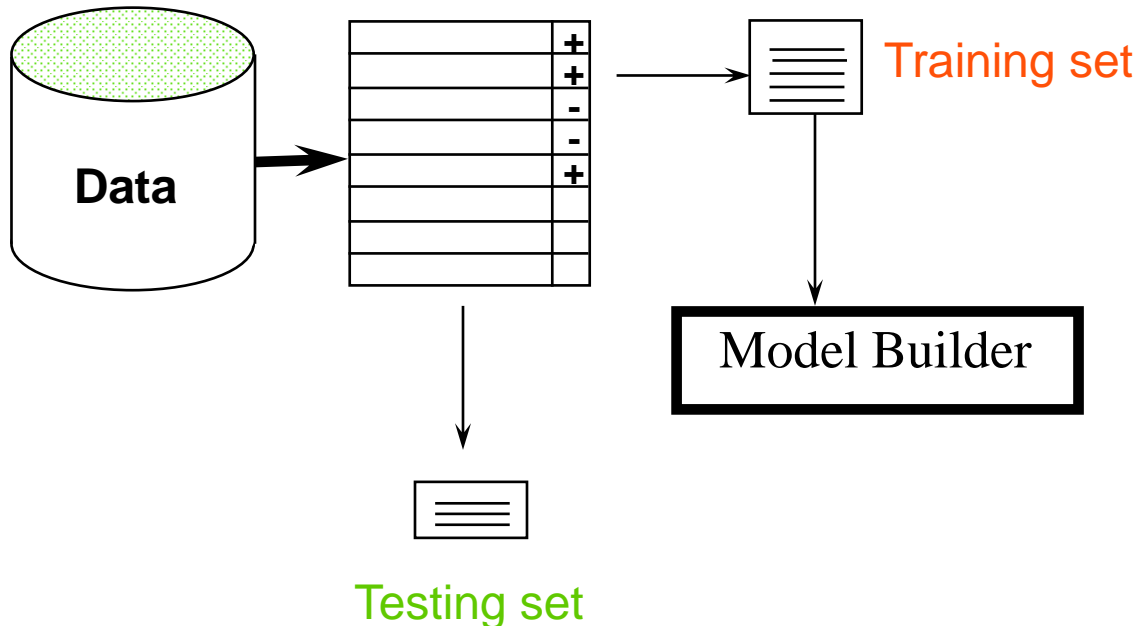
Evaluation on "LARGE" data, 2

- A simple evaluation is sufficient
  - □ Randomly split data into training and test sets (usually 2/3 for train, 1/3 for test)
- Build a classifier using the *train* set and evaluate it using the *test* set.

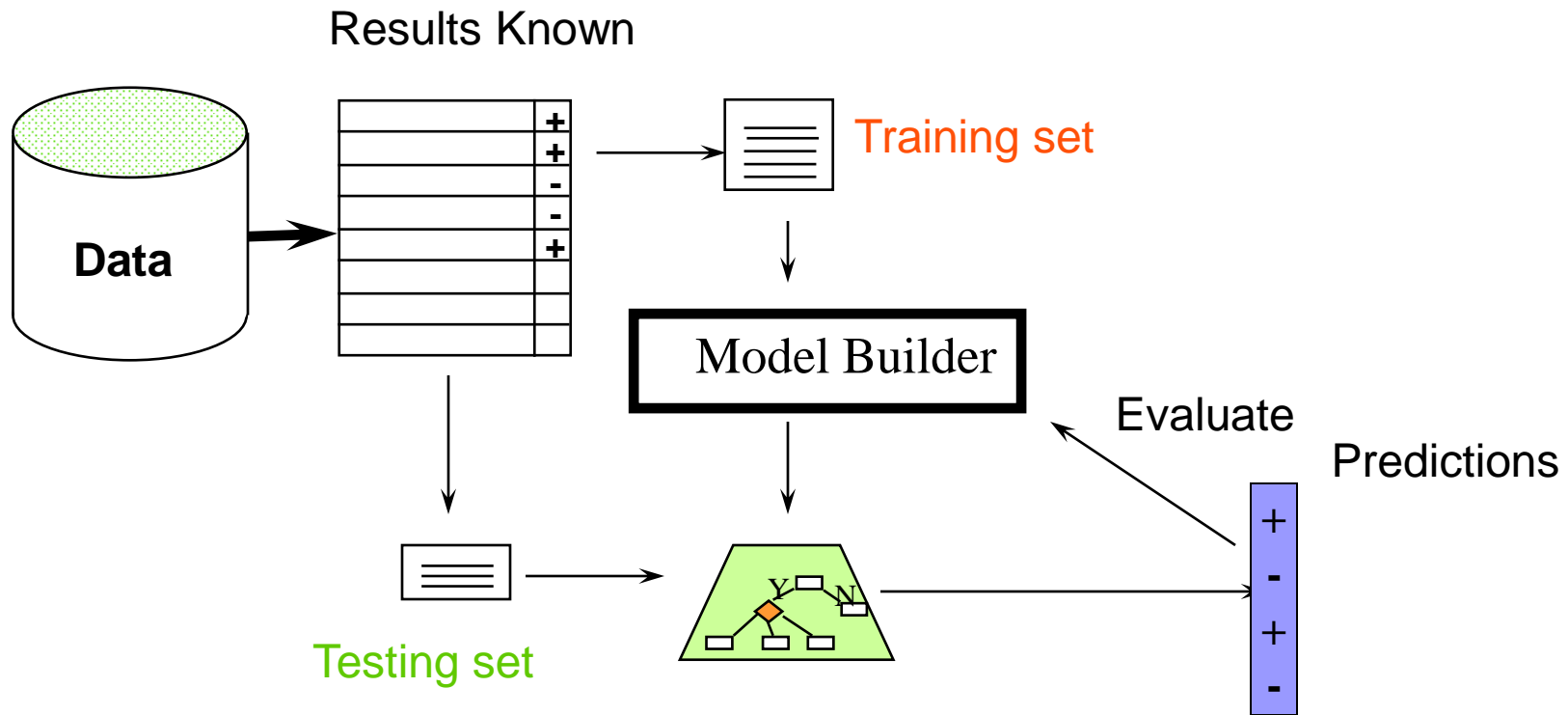# Classification Step 1: Split data into train and test sets

THE PAST

Results Known

Data

Training set

Testing set
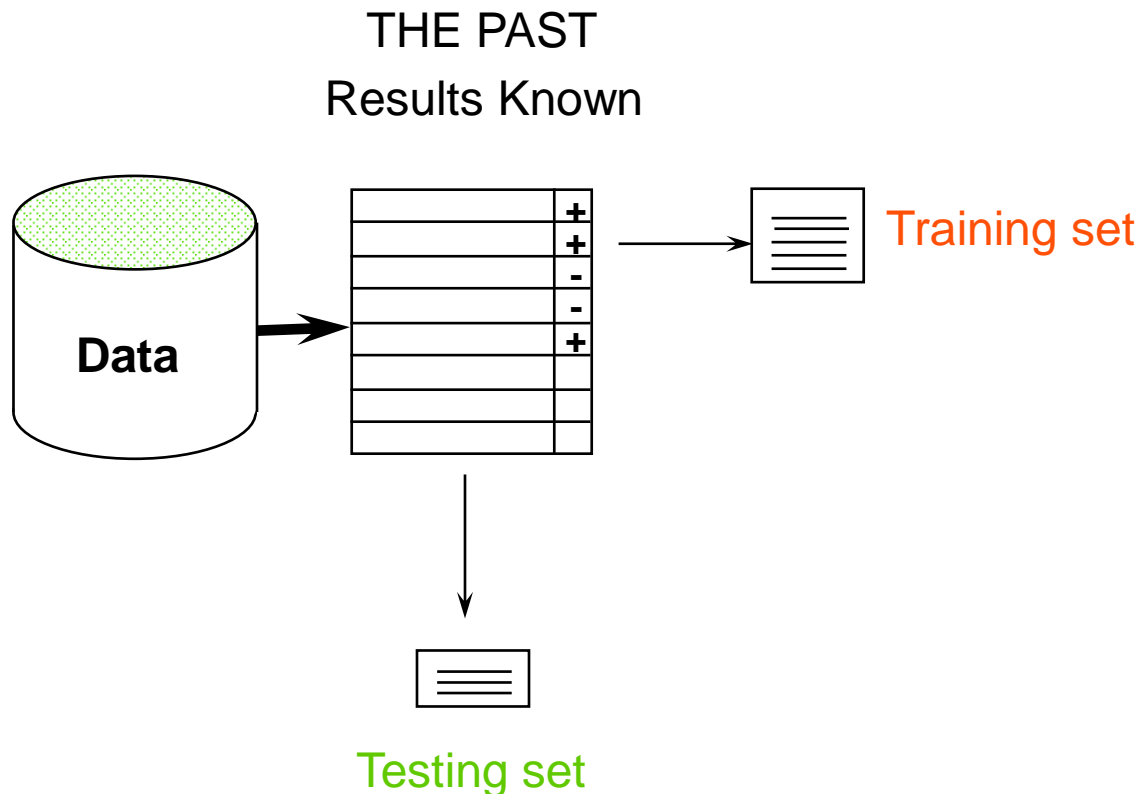
Classification Step 2: Build a model on a training set

THE PAST

Results Known

Data

+
+
-
-
+

Training set

Model Builder

Testing set

Classification Step 3: Evaluate on test set (Re-train?)

Results Known

Data

+
+
-
-
+

Training set

Model Builder

Testing set

Evaluate

Predictions

+
-
+
-

Unbalanced data

- Sometimes, classes have very unequal frequency
  - Attrition prediction: 97% stay, 3% attrite (in a month)
  - medical diagnosis: 90% healthy, 10% disease
  - eCommerce: 99% don't buy, 1% buy
  - Security: >99.99% of Americans are not terrorists
- Similar situation with multiple classes
- Majority class classifier can be 97% correct, but useless

© Branko Kavšek, 2023/24

Handling unbalanced data – how?

If we have two classes that are very unbalanced, then how can we evaluate our classifier method?

© Branko Kavšek, 2023/24

Balancing unbalanced data, 1

- With two classes, a good approach is to build **BALANCED** train and test sets, and train model on a balanced set
  - ☐ randomly select desired number of minority class instances
  - ☐ add equal number of randomly selected majority class
- How do we generalize "balancing" to multiple classes?

Balancing unbalanced data, 2

- Generalize "balancing" to multiple classes
  - □ Ensure that each class is represented with approximately equal proportions in train and test

© Branko Kavšek, 2023/24

A 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: builds the basic structure
    - ☐ Stage 2: optimizes parameter settings

- The test data can't be used for parameter tuning!

- Proper procedure uses three sets: **training data, validation data, and test data**
    - ☐ Validation data is used to optimize parameters

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

Classification: Train, Validation, Test split
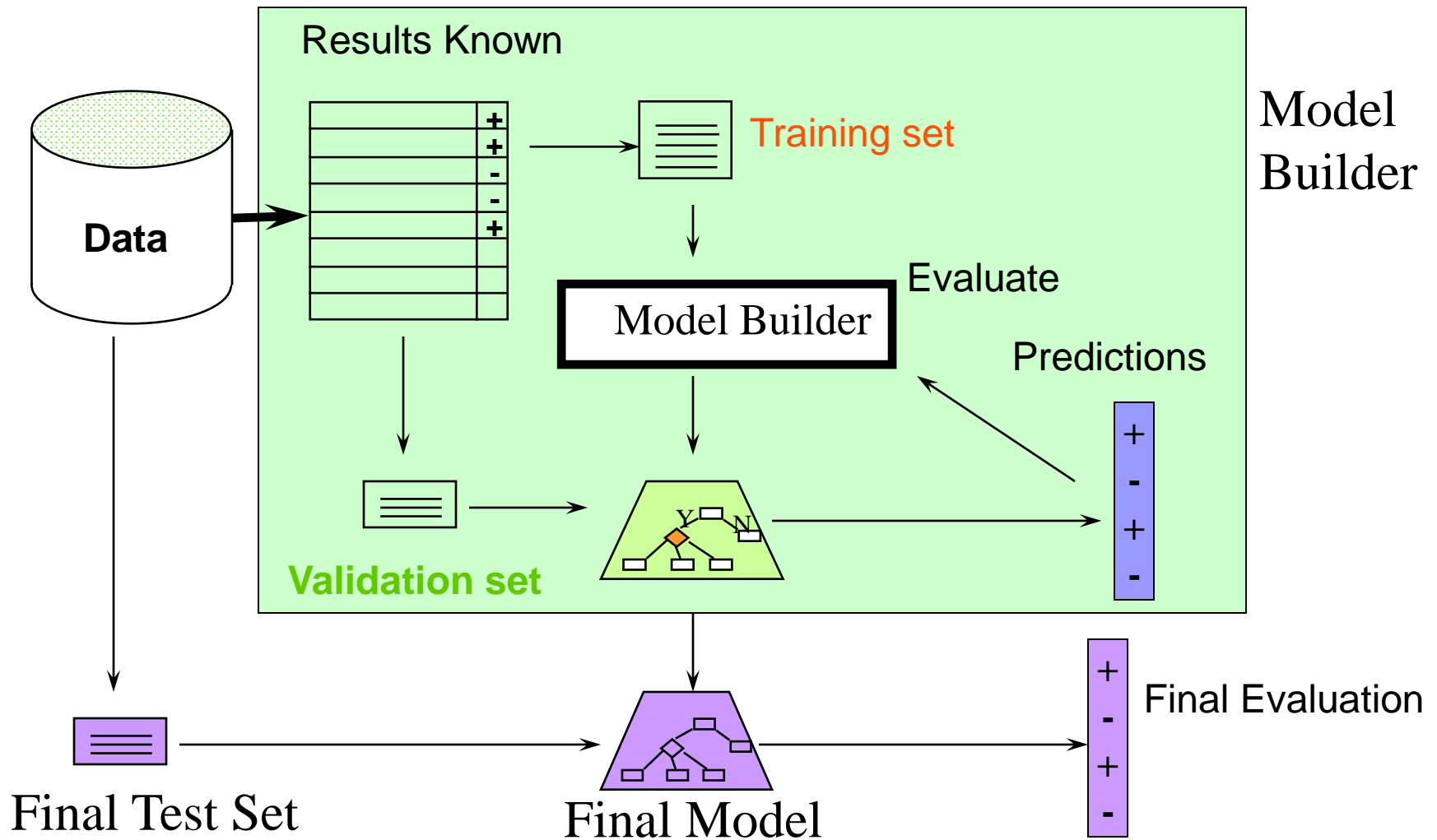
*Predicting performance

- Assume the estimated error rate is 25%. How close is this to the true error rate?
  - □ Depends on the amount of test data
- Prediction is just like tossing a biased (!) coin
  - □ "Head" is a "success", "tail" is an "error"
- In statistics, a succession of independent events like this is called a Bernoulli process
- Statistical theory provides us with confidence intervals for the true underlying proportion!

*Confidence intervals

- 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 $p \in [69.1, 80.1]$

*Mean and variance

- Mean and variance for a Bernoulli trial: *p, p*(1–*p*)

- Expected success rate *f=S/N*

- Mean and variance for *f* : *p, p* (1–*p*)/*N*

- For large enough *N, f* follows a Normal distribution

- c% confidence interval [–*z* ≤ *X* ≤ *z*] for random variable with 0 mean is given by:

$$\Pr[-z \leq X \leq z] = c$$

- With a symmetric distribution:

$$\Pr[-z \leq X \leq z] = 1 - 2 \times \Pr[X \geq z]$$

*Confidence limits

- Confidence limits for the normal distribution with 0 mean and a variance of 1:
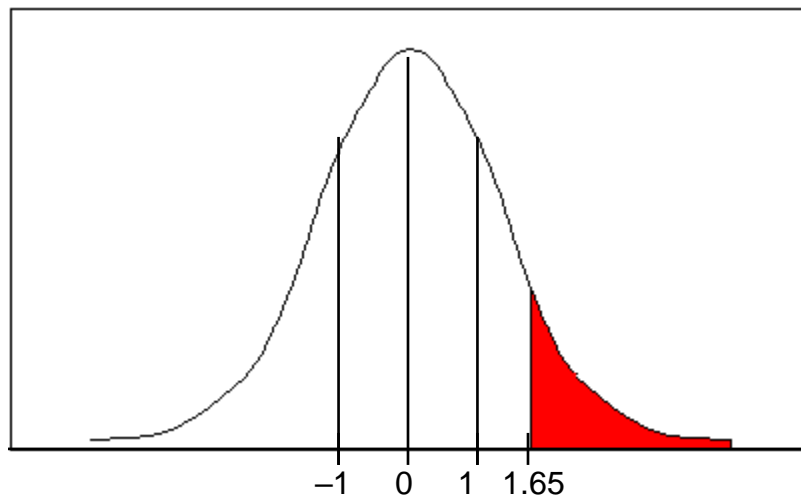
Pr[$X \geq z$]	z
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84
40%	0.25

−1 0 1 1.65

- Thus:

$$\Pr[-1.65 \leq X \leq 1.65] = 90\%$$

- To use this we have to reduce our random variable f to have 0 mean and unit variance

*Transforming *f*

- **Transformed value for *f* :**

$$\frac{f - p}{\sqrt{p(1-p)/N}}$$

(i.e. subtract the mean and divide by the *standard deviation*)

- **Resulting equation:**

$$\Pr\left[-z \le \frac{f - p}{\sqrt{p(1-p)/N}} \le z\right] = c$$

- **Solving for *p* :**

$$p = \left(f + \frac{z^2}{2N} \pm z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}\right) \Big/ \left(1 + \frac{z^2}{N}\right)$$

*Examples

- $f = 75\%$, $N = 1000$, $c = 80\%$ (so that $z = 1.28$): $p \in [0.732, 0.767]$

- $f = 75\%$, $N = 100$, $c = 80\%$ (so that $z = 1.28$): $p \in [0.691, 0.801]$

- Note that normal distribution assumption is only valid for large N (i.e. $N > 100$)
- $f = 75\%$, $N = 10$, $c = 80\%$ (so that $z = 1.28$): $p \in [0.549, 0.881]$
  (should be taken with a grain of salt)

Evaluation on "small" data, 1

- The *holdout* method reserves a certain amount for testing and uses the remainder for training
  - ☐ Usually: one third for testing, the rest for training
- For "unbalanced" datasets, samples might not be representative
  - ☐ Few or none instances of some classes
- *Stratified sample:*
  advanced version of balancing the data
  - ☐ Make sure that each class is represented with approximately equal proportions in both subsets

Evaluation on "small" data, 2

- What if we have a small data set?
- The chosen 2/3 for training may not be representative.
- The chosen 1/3 for testing may not be representative.

© Branko Kavšek, 2023/24

Repeated holdout method, 1

- 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 (possibly with stratification)

  - The error rates on the different iterations are averaged to yield an overall error rate

- This is called the *repeated holdout* method

Repeated holdout method, 2

- Still not optimum: the different test sets overlap.

- Can we prevent overlapping?

Cross-validation

- *Cross-validation* avoids overlapping test sets
  - □ First step: data is split into *k* subsets of equal size
  - □ Second step: each subset in turn is used for testing and the remainder for training
- This is 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

Cross-validation example:

— Break up data into groups of the same size

— Hold aside one group for testing and use the rest to build the model

— Test

— Repeat
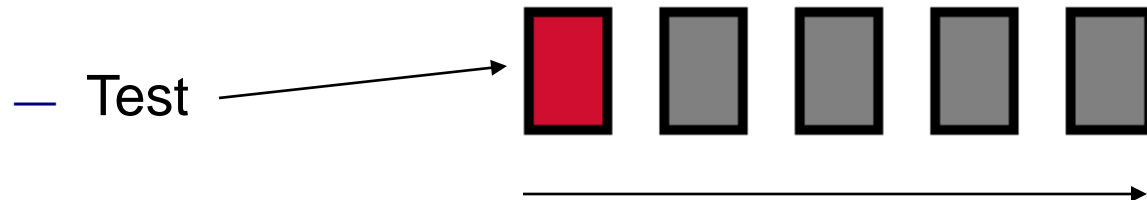
More on cross-validation

- 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

- Stratification reduces the estimate's variance

- Even better: repeated stratified cross-validation
  - ☐ E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)

Leave-One-Out cross-validation

- Leave-One-Out:
  a particular form of cross-validation:
  - ☐ Set number of folds to number of training instances
  - ☐ I.e., for *n* training instances, build classifier *n* times
- Makes best use of the data
- Involves no random subsampling
- Very computationally expensive

Leave-One-Out-CV and stratification

- Disadvantage of Leave-One-Out-CV: stratification is not possible

    - It *guarantees* a non-stratified sample because there is only one instance in the test set!

- Extreme example:
  random dataset split equally into two classes

    - Best inducer predicts majority class

    - 50% accuracy on fresh data

    - Leave-One-Out-CV estimate is 100% error!

*The bootstrap

- CV uses sampling *without replacement*
  - □ The same instance, once selected, can not be selected again for a particular training/test set

- The *bootstrap* uses sampling *with replacement* to form the training set
  - □ Sample a dataset of *n* instances *n* times *with replacement* to form a new dataset of *n* instances
  - □ Use this data as the training set
  - □ Use the instances from the original dataset that don't occur in the new training set for testing

BS/201   3 Strand Also Availible
                                     $24.95

2 Strand Boot Strap
with concho $19.95

*The 0.632 bootstrap

- Also called the *0.632 bootstrap*
  - A particular instance has a probability of 1–1/n of *not* being picked
  - Thus its probability of ending up in the test data is:

$$\left(1-\frac{1}{n}\right)^{n} \approx e^{-1} = 0.368$$

  - This means the training data will contain approximately 63.2% of the instances

*Estimating error with the bootstrap

- The error estimate on the test data will be very pessimistic

  - Trained on just ~63% of the instances

- Therefore, combine it with the resubstitution error:

$$err = 0.632 \cdot e_{\text{test instances}} + 0.368 \cdot e_{\text{training instances}}$$

- The resubstitution error gets less weight than the error on the test data

- Repeat process several times with different replacement samples; average the results

*More on the bootstrap

- Probably the best way of estimating performance for very small datasets

- However, it has some problems
  - ☐ Consider the random dataset from above
  - ☐ A perfect memorizer will achieve
    0% resubstitution error and
    ~50% error on test data
  - ☐ Bootstrap estimate for this classifier:
    $$err = 0.632 \cdot 50\% + 0.368 \cdot 0\% = 31.6\%$$
  - ☐ True expected error: 50%

Comparing data mining schemes

- Frequent situation: we want to know which one of two learning schemes performs better
  - Note: this is domain dependent!
- Obvious way: compare 10-fold CV estimates
  - Problem: variance in estimate
- Variance can be reduced using repeated CV
- However, we still don't know whether the results are reliable

Significance tests

- Significance tests tell us how confident we can be that there really is a difference
- *Null hypothesis*: there is no "real" difference
- *Alternative hypothesis*: there is a difference
- A significance test measures how much evidence there is in favor of rejecting the null hypothesis
- Let's say we are using 10 times 10-fold CV
- Then we want to know whether the two means of the 10 CV estimates are significantly different
  - □ *Student's paired t-test* tells us whether the means of two samples are significantly different

*Paired t-test

- *Student's t-test* tells whether the means of two samples are significantly different

- Take individual samples from the set of all possible cross-validation estimates

- Use a *paired* t-test because the individual samples are paired
  - ☐ The same CV is applied twice

William Gosset

Born: 1876 in Canterbury; Died: 1937 in Beaconsfield, England. Obtained a post as a chemist in the Guinness brewery in Dublin in 1899. Invented the t-test to handle small samples for quality control in brewing. Wrote under the name "Student".

© Branko Kavšek, 2023/24

*Distribution of the means

- $x_1\ x_2\ \dots\ x_k$ and $y_1\ y_2\ \dots\ y_k$ are the 2*k* samples for a *k*-fold CV

- m_x and m_y are the means

- With enough samples, the mean of a set of independent samples is normally distributed $\dfrac{m_x - \mu}{\sqrt{\sigma_x^2 / k}}$

- Estimated variances of the means are σ_x^2/k and σ_y^2/k

- If μ_x and μ_y are the true means then $\dfrac{m_x - \mu_x}{\sqrt{\sigma_x^2 / k}}\quad \dfrac{m_y - \mu_y}{\sqrt{\sigma_y^2 / k}}$

  are *approximately* normally distributed with mean 0 and variance 1

*Student's distribution

- With small samples (*k* < 100) the mean follows *Student's distribution with k–1 degrees of freedom*

- Confidence limits:

9 degrees of freedom

Pr[$X \geq z$]	z
0.1%	4.30
0.5%	3.25
1%	2.82
5%	1.83
10%	1.38
20%	0.88

normal distribution

Pr[$X \geq z$]	z
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84

*Distribution of the differences

- Let $m_d = m_x - m_y$

- The difference of the means (m_d) also has a Student's distribution with k–1 degrees of freedom

- Let σ_d^2 be the variance of the difference

- The standardized version of m_d is called the t-statistic:
$$t = \frac{m_d}{\sqrt{\sigma_d^2 / k}}$$

- We use t to perform the t-test

*Performing the test

1. Fix a significance level α

   □ If a difference is significant at the α% level, there is a (100-α)% chance that there really is a difference

2. Divide the significance level by two because the test is two-tailed

   □ I.e. the true difference can be +ve or –ve

3. Look up the value for z that corresponds to $\alpha/2$

4. If $t \leq -z$ or $t \geq z$ then the difference is significant

   □ I.e. the null hypothesis can be rejected

Unpaired observations

- ■ If the CV estimates are from different randomizations, they are no longer paired
  - ☐ (or maybe we used *k*-fold CV for one scheme, and *j*-fold CV for the other one)

- ■ Then we have to use an *un* paired t-test with min(*k*, *j*) – 1 degrees of freedom

- ■ The *t*-statistic becomes:

$$t = \frac{m_d}{\sqrt{\sigma_d^2 / k}} \quad \Longrightarrow \quad t = \frac{m_x - m_y}{\sqrt{\dfrac{\sigma_x^2}{k} + \dfrac{\sigma_y^2}{j}}}$$

*Interpreting the result

- All our cross-validation estimates are based on the same dataset

- Hence the test only tells us whether a *complete k*-fold CV for this dataset would show a difference

  - Complete *k*-fold CV generates all possible partitions of the data into *k* folds and averages the results

- Ideally, should use a different dataset sample for each of the *k*-fold CV estimates used in the test to judge performance across different training sets

T-statistic many uses

- Looking ahead, we will come back to use of T-statistic for gene filtering in later modules

*Predicting probabilities

- Performance measure so far: success rate

- Also called *0-1 loss function*:

$$\sum_i \begin{cases} 0 & \text{if prediction is correct} \\ 1 & \text{if prediction is incorrect} \end{cases}$$

- Most classifiers produces class probabilities

- Depending on the application, we might want to check the accuracy of the probability estimates

- 0-1 loss is not the right thing to use in those cases

*Quadratic loss function

- $p_1 \dots p_k$ are probability estimates for an instance

- *c* is the index of the instance's actual class

- $a_1 \dots a_k = 0$, except for a_c which is 1

- *Quadratic loss* is: $$\sum_j (p_j - a_j)^2 = \sum_{j \neq c} p_j^2 + (1 - p_c)^2$$

- Want to minimize $$E\left[\sum_j (p_j - a_j)^2\right]$$

- Can show that this is minimized when $p_j = p_j^*$, the true probabilities

*Informational loss function

- The informational loss function is $-\log(p_c)$,
  where c is the index of the instance's actual class

- Number of bits required to communicate the actual class

- Let p_1^* … p_k^* be the true class probabilities

- Then the expected value for the loss function is:

$$-p_1^* \log_2 p_1 - ... - p_k^* \log_2 p_k$$

- Justification: minimized when $p_j = p_j^*$

- Difficulty: *zero-frequency problem*

*Discussion

- ### Which loss function to choose?

  - ☐ Both encourage honesty

  - ☐ Quadratic loss function takes into account all class probability estimates for an instance

  - ☐ Informational loss focuses only on the probability estimate for the actual class

  - ☐ Quadratic loss is bounded:
    it can never exceed 2 $\longrightarrow$ $1 + \sum_j p_j^2$

  - ☐ Informational loss can be infinite

- ### Informational loss is related to *MDL principle*

© Branko Kavšek, 2023/24

Evaluation Summary:

- Use Train, Test, Validation sets for "LARGE" data

- Balance "un-balanced" data

- Use Cross-validation for small data

- Don't use test data for parameter tuning – use separate validation data

- Most Important: Avoid Overfitting

© Branko Kavšek, 2023/24