IST Austria: Data Science and Scientific Computing
Track Core Course 2016/17 – Segment 1: "Predictive Models"

Instructor: Christoph Lampert <chl@ist.ac.at>
TAs: Bor Kavcic <bor.kavcic@ist.ac.at>, Katharina Ölsböck <katharina.oelsboeck@ist.ac.at>,

Exercise Sheet 1

Literature

1) Read this paper: [L. Breiman, *"Statistical Modeling: The Two Cultures"*, Statistical Science 16 (3), 199–231 (2001)] as well as as the comments and replies following it at `http://projecteuclid.org/download/pdf_1/euclid.ss/1009213726`. Answer the following questions in one or two sentences each.

a) what does Breiman identify as the biggest problem of the "statistical community"?

b) how did the "statistical community" react to Breiman's claims?

c) what aligns better with your own interests: data modeling or algorithmic modeling? (or something else?)

d) if you were in charge of IST's Data Science program, how much of which aspects would you have taught?

Hands-On

The goal of this exercise is to get hands-on experience with regularization and model selection.

2) Reproduce the figures about *model selection* for ridge regression shown in the lecture:

- download the `diabetes.txt` dataset from the course website
  use the first 10 columns as features and the last column as target value
- repeat the following experiment 50 times with different random seeds
  - pick a random subset of 100 example for training and 100 examples for validation
  - a train least squares regression model without regularization (using matrix operations)
  - for each regularization parameters $\lambda \in \{2^{-20}, \dots, 2^{10}\}$
    - train a ridge regression model with regularization λ (using matrix operations)
  - evaluate the trained models on the validation set
- plot the results in a style that you find most appropriate/informative

More Hands-On

The goal of this exercise is to get hands-on experience in handling *text* data.

- Download the archive `http://www.ist.ac.at/~chl/courses/DSSC-TCC17/glove.6B.50d.zip` (66MB). It contains a 50-dimensional *word-to-vector* model in the following format: each row corresponds to one item (a word or a symbol), the item is specified in the first column, the remaining columns are 50 numeric vector entries. Columns are separated by spaces. Note: all words are lowercased.

- Write code that loads a file of this format and allows you to access it through a function that takes a word or symbol as input (as a string) and outputs the corresponding vector.

  Note 1: If your programming language of choice (ploc) has problems handling mixed text/numeric documents, write a script that transforms a file of the above format into two files, one containing the words/symbols and one containing the numeric vectors (in the same order). Then use these two files for the above task.

  Note 2: Beware that the file contain special characters, e.g. `'#"%&`, and non-latin characters, e.g. `äæå` in Unicode (UTF-8). If your ploc cannot handle these, write a script that preprocesses the text file by removing all lines with such entries.

- Compute all pairwise Euclidean distances between the following words: `frog`, `toad`, `iguana`, `cheesecake`.

- Write code that for a given word computes the 10 words closest to it. What are the outcomes for `frog` and `cheesecake`?

  Note: computing all distances in a loop might take long. Try to rely on vector or matrix operations as much as possible, and prefer existing routines for sorting or partial sorting over implementing the search for the 10 largest entries yourself.

Additional exercises (voluntary, to get more practice)

4) In the regression experiments above, create two validation sets instead of just one and plot the average absolute difference in error between both of them. Interpret your findings.

5) In the regression experiments above, also compute and plot the error estimate of from 10-fold cross-validation. For each repeat, identify the 'best' regularization constant according to its CV error and evaluate the corresponding error on the validation set. Interpret your findings.

6) The above text representation can also be used for finding *analogies* of the type *"A is to B, like C is to ?"*. For this, one computes the vectors $\phi(A)$, $\phi(B)$ and $\phi(C)$, searches the word in the dictionary with smallest distance to the vector $\phi(B) - \phi(A) + \phi(C)$.

- Compute the 5 best answers to the analogy "`prince` is to `princess`, like `boy` is to `?`"

- Compute the 5 best answers to the analogy "`england` is to `france`, like `london` is to `?`"

7) Repeat the text experiments using a different word-to-vector model, e.g. the 25-dimensional `http://www.ist.ac.at/~chl/courses/DSSC-TCC17/glove.twitter.27B.25d.zip` (105MB) or the 300-dimensional `http://nlp.stanford.edu/data/glove.840B.300d.zip` (2GB!). What do you observe?