Car Insurance

Havránek, Pokorný, Tomášek

Outline

- Data overview
- Horizontal approach + Decision tree/forests
- Vertical (column) approach + Neural networks
- SVM

Data overview

- Customers
    - Viewed policies
    - Bought policy

Horizontal approach

Štěpán Havránek

Horizontal approach

- Horizontal approach
  - Model: decision trees
  - Input: customer info
  - Output: bought policy
- Last time results:
  - 50 - 80% mean validation error per one output attribute
    - By C4.5 alg

Horizontal approach

- Overfitted trees pruning
  - Simplified the models
  - Mean validation error decreased
    - 30 - 65% per one output attribute
- Wider input data
  - Customer info + one viewed policy
  - Better results:
    - 10 - 45% MVE

Horizontal approach

- Observation: Output attributes aren't independent
  - New trees deciding subsets of the output attributes
    - All the possible subsets
      - => Out of memory exception
    - Only manually picked subsets
      - Singletons, pairs, 6 of 7, all
    - Mean validation error much more better than independent products
      - 15 - 80%
      - Only observed values
        - Running time > 12h

Horizontal approach

- With the subsets we have more than one tree for each output attribute
  - Save them to the forests
  - Voting model
    - By validation score
      - More progressive
      - Less progressive
  - Results with less wide input data (running time)
    - More progressive
      - Total overfit on the best tree => useless results
    - Less progressive
      - Variant results
      - Not good in official rating (~ 0.2)
      - Good for others

Column approach

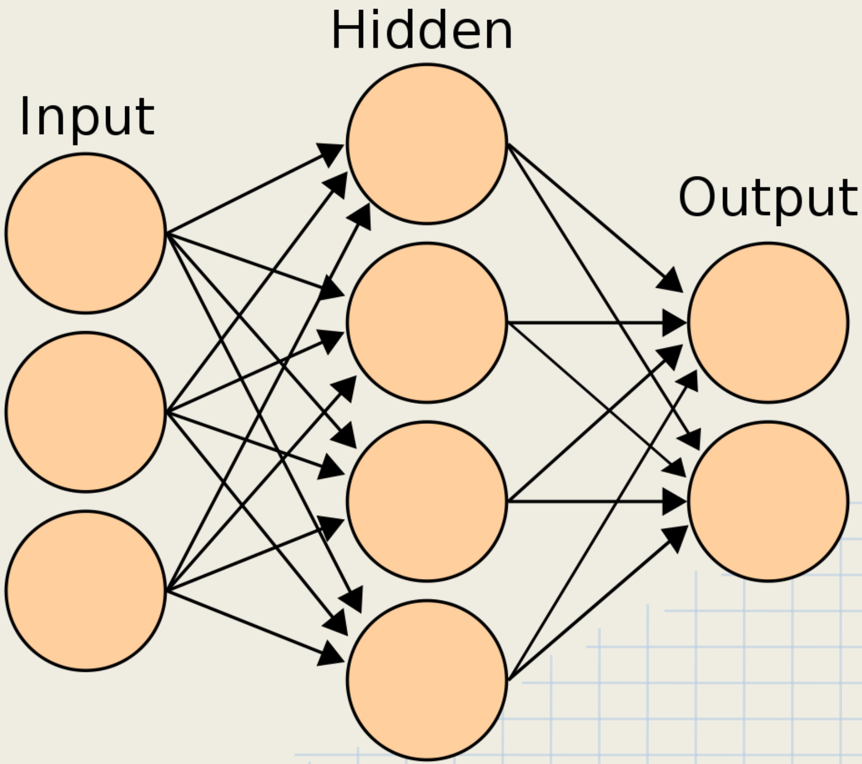
Bc. Jan Tomášek

Column approach + N-net

- Separated Neural network for each Class
- Neural network inputs
  - last three entries in class
    - as characteristic vector
  - car age
    - original range 0-80
      - cropped to 0-30
  - car value
    - ‘a’ - ‘i’
      - scaled to (0-1)
- Levenberg-Marquardt backpropagation

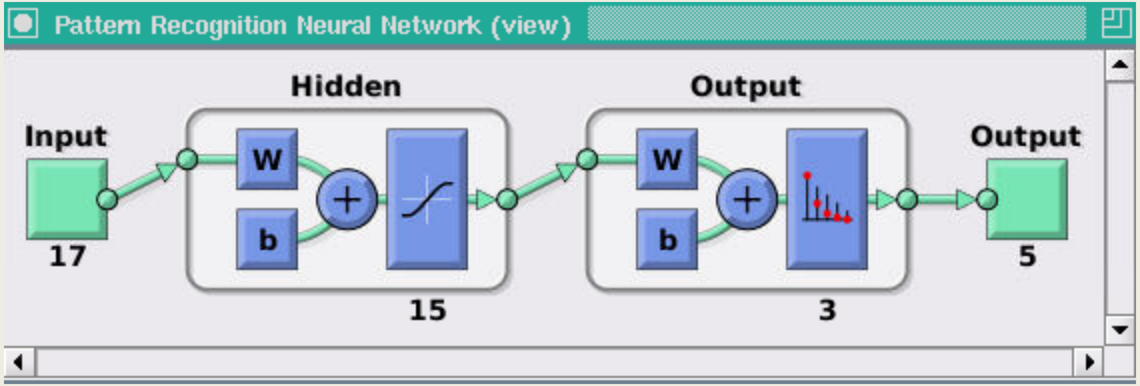
Neural network - basics

- Oriented graph
  - weights
  - bias
- (+) strong in finding linear dependence
- (-) black box



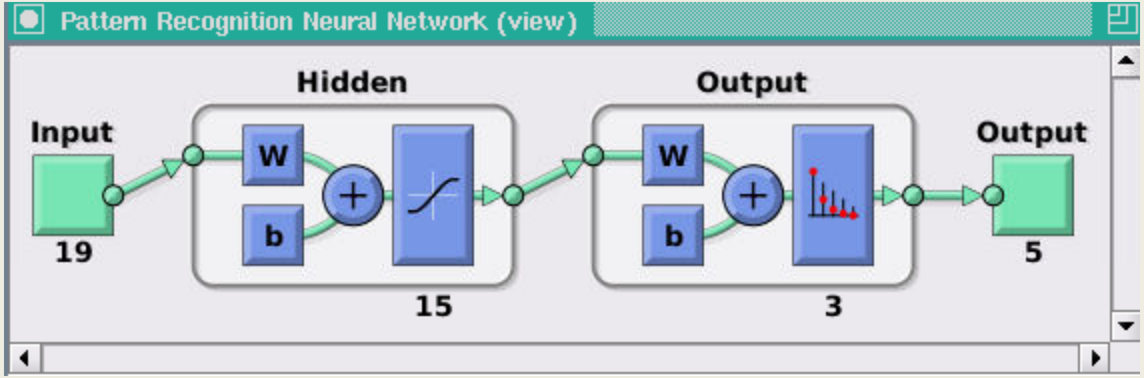
Neural network 1

- Neural network structure
  - one hidden layer with 15 neurons
- result
  - ~ 0.529



Neural Network 2

- added more features
  - risk factor
  - cost
- result
  - ~0.535



Neural network 3

- Deep network
- More networks
  - 20 networks
  - up to 3 layers
  - more different networks
  - bizarre topologies
- result
  - ~0.528
  - probably overfit

Neural network 4

- Manual selection of good networks
- result
  - ~0.537
- total 8 networks

Neural network results

- Best solution(0.53745) about the same as last quoted benchmark(0.53793)
  - nn probably learns to use last :)
- **best solution of our group**
- possible improvement
  - Adaptive Boosting
    - lear something else where last is failing
  - combine with last quoted (how?), combine with
- neural network problem
  - very fast convergence to local optimum
    - 10 iterations
    - almost no improvement after

scikit-learn SVC’s

Michal Pokorný

scikit-learn SVC’s

- Support vector machine classifiers
  - SVM’s find the optimal separating hyperplane in a high-dimensional example space
  - Kernel function can be interpreted as “searching for similar browsing histories”
  - Basic SVM’s just give a binary result, so N of them are trained for more-than-binary attributes
  - Can be tweaked to give probabilities, not just predictions
    - (Some simple approaches don’t guarantee that.)
- Simplification: suggest the most likely plan the user actually looked at
- So, let’s score every browsed plan
  - “Naive Bayes assumption”: take one SVC for every attribute and multiply together their probability results

scikit-learn SVC’s

- Tested feature ideas:
  - “Static features”: day of the week, group size, homeowner, car age+value, risk factor, oldest/youngest age, married couple?, cost
  - Histogram of browsed attribute values
    - “We saw A=0 5 times, A=1 2 times, ...”
  - Most commonly browsed plan
  - Last browsed plan, first browsed plan
- Unfortunately, no combinations improved upon the trivial solution :(
  - (The best reached accuracy equals that of the benchmark model.)
- Overfitting?
  - Not too likely, since changing regularization parameters of the SVC’s didn’t help

scikit-learn SVC's

But…

This leaderboard is calculated on approximately 30% of the test data.
The final results will be based on the other 70%, so the final standings may be different.

See someone using multiple accounts?
Let us know.

#	Δ1w	Team Name * in the money	Score	Entries	Last Submission UTC (Best – Last Submission)
1	↑2	Magic Learner *	0.54571	392	Sun, 18 May 2014 23:49:54 (-30.6h)
2	↓1	Owen *	0.54571	71	Mon, 19 May 2014 00:55:50 (-0h)
3	↓1	Finite State Insurance Machines *	0.54565	219	Sun, 18 May 2014 23:37:23 (-3d)
4	-	Alessandro & BreakfastPirate	0.54535	259	Mon, 19 May 2014 00:53:42
5	↑14	dynamic24	0.54481	226	Sun, 18 May 2014 22:19:45
6		User Error Structure	0.54463	105	[illegible]

…, while the trivial solution gives 0.53739.

scikit-learn SVC's

- Since neither us, nor actual machine learning professionals found any significantly better solutions, the data probably just isn't there.
    - Maybe more features could help.
    - Website analytics could give us visit lengths, tracking of user activity, etc. - those are some ideas of useful predictors of plans or plan features the user might be interested in.

- Finally, please accept an apology for my physical absence - this is the result of a planning/scheduling problem concerning my student duties.

Q & A