7. Search and Variable Selection

Data mining entails many kinds of search. Some searches are easier than others.

A relatively easy kind of search is univariate. For example, one might want to find an appropriate bandwidth h for a smoother or the appropriate degree for fitting a polynomial regression.

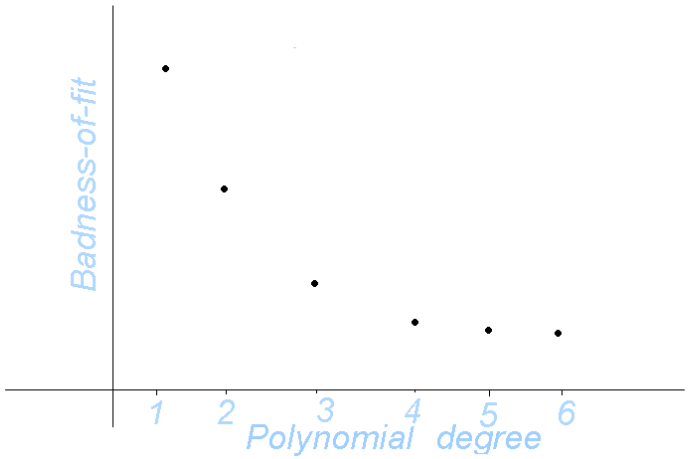
Search becomes harder in multivariate cases, such as finding an appropriate set of knots for spline smoothing or a set of weights in a neural network.

Even more difficult is combinatorial search, where there is no natural Euclidean space. This occurs in variable selection (a/k/a feature selection).

The hardest search is list search, where there is no structure in the problem at all. This occurs, for example, when there is a list of possible models, each of which is largely unrelated to the others, and one wants to find the best model for the data.

7.1 Univariate Search

In univariate search, one good strategy is to plot some criterion (e.g., goodness-of-fit, predictive mean squared error) against the index (e.g., degree in polynomial regression) and look for a knee in the curve.



If the criterion is monotonic in the index but graphical solution is difficult, try Fibonnacci search. If the criterion is not monotonic in the index, then other kinds of search (e.g., random restart hill-climbing, simulated annealing) should be considered.

7.2 Multivariate Search

For multivariate search, there are dozens of smart algorithms, and the best choice depends upon the nature of the response surface.

If the surface has a single bump, then steepest ascent works very well. If the surface has many bumps, then random restart combined with steepest ascent is a strategy that enables one to make probability statements about the chance that one has found a global maximum.

The Nelder-Mead algorithm is simple to program and very robust. It sets a simplex in the space $\mathbb{R}^k$ and passes the worst vertex through the center to the opposite side. (Nelder and Mead; 1965, it Computer Journal, **7**, 308-313.)

Hybrid methods combine features from standard operations research algorithms and allow one to learn about the surface as one seeks the maximum.

7.3 Variable Selection

For combinatorial search, one wants to take advantage of whatever structure the problem enjoys.

For example, variable selection is a key problem in data mining. Often one has very large p and one wants (needs) to discard those that have no or little predictive value. If one looked at all possible subsets of variables, there are 2^p cases to consider. Something smarter is needed.

The 2^p possible models can be identified with the 2^p vertices of the unit hypercube in $\mathbb{R}^p$. The $(0, 0, \ldots, 0)$ vertex corresponds to the model with all variables excluded, whereas the $(1, 1, \ldots, 1)$ model is the regression on all variables. From this perspective, a clever search of the hypercube would be an attractive way to find a good regression model.

7.3.1 Gray Codes

A Hamiltonian circuit of the unit hypercube is a traversal that reaches each vertex exactly once. There are many possible Hamiltonian circuits—the exact number is not known.

¿From the standpoint of model search, one wants a Hamiltonian circuit that has desirable properties of symmetry, treating all vertices in the same way.

The Gray code is a procedure for listing the vertices of the hypercube in such a way that there is no repetition, each vertex is one edge from the previous vertex, and all vertices in a neighborhood are explored before moving on to a new neighborhood. Wilf (1989; *Combinatorial Algorithms: An Update*, SIAM) describes the mathematical theory and properties of the Gray code system.

To explain the Gray code algorithm, consider the case of four explanatory variables, or the unit hypercube in $\mathbb{R}^4$. The table shows the rank of the vertex in the Gray code traversal, the binary digit representation of the rank, and the bit string of the visited vertex on the hypercube.

Gray code vertex rank, binary rank, and vertex string.

0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

The Gray code has subtle balance. For example, it can be generated by reflection and recursion. Let L_p be the list of all possible binary bit strings of length p, arranged in Gray code order. Then generate the first half of L_{p+1} by writing a zero in front of each element in the list L_p. For the second half of L_{p+1}, write L_p in reverse order, and then prefix each element with a one. By concatenating the lists, one obtains L_{p+1}.

Suppose one prefixes each Gray code string with an infinite number of zeroes. This makes it possible to consider the numbers corresponding to the Gray code strings as an infinite series:

$$0, 1, 3, 2, 6, 7, 5, 4, 12, 13, 15, 14, 10, 11, 9, 8, \ldots$$

Note that each number in the sequence is relatively close to its neighbors. Yuen (1974; *IEEE Transactions on Information Theory*, **20**, 688) shows that two strings in the Gray code whose Hamming distance is at least d must have ranks that differ by at least $[2d/3]$ (here $[\cdot]$ is the nearest-integer function), and this provides the greatest possible separation. This means that the traversal explores models locally and exhaustively, rather than swooping back after a distant excursion.

¿From our perspective, the key point from Yuen's theorem is that if starts at an arbitrary model, then goes a large number of steps in the Gray code traversal, one ends up at a vertex corresponding to a model that is very different from the starting point. This property suggests that by taking every dth step, for d large, and then testing the corresponding model, one is performing a thorough search of the set of possible models.

To implement this search strategy one needs to be able to efficiently generate the Gray code vertex for step m. Wilf gives a theorem that does this.

Theorem: Let $m = \sum a_i 2^i$ be the representation of integer m in binary notation. Let $\ldots, b_3 b_2 b_1 b_0$ be the string for the vertex of rank m in the Gray code. Then

$$b_i = a_i + a_i + 1 \ (\mathbf{mod}\ 2)$$

for $i = 0, 1, 2, \ldots$.

To use this ranking theorem to efficiently explore a set of models, suppose one decides to examine only 100 models and then infer the final fit.

If there are p explanatory variables, one takes $d = [2^p/100]$, and then finds the Gray code vertex sequences of rank $d, 2d, \ldots, 100d$.

Each sequence defines a particular set of variables that may be included or excluded in the regression. In practice, one would examine the 100 model fitting results, probably in terms of the square root of the adjusted R^2, and then home in on the region of the cube that provides good explanation.

This enables one to quickly identify the vertex or bit string corresponding to a set of variables that provides good explanation. One might make additional Gray code searches in the region of the best results from the first search, and iterate to find the final model.

7.3.3 Experimental Design Selection

Another approach to variable selection uses ideas from experimental design. The method is due to Clyde (1999; *Bayesian Statistics 6*, 157-185, Oxford University Press).

To implement this approach, one views each explanatory variable as a factor in an experimental design. All factors have two levels, corresponding to whether or not the explanatory variable is included in the model. This enables one to perform a 2^{p-k} fractional factorial experiment in which one fits a multiple regression model to the included variables and records some measure of goodness-of-fit.

Obviously, one takes k to be sufficiently large that it is possible to perform the computations in a reasonable amount of time and also to limit the effect of multiple testing.

One uses analysis of variance to examine which factors and factor combinations have a significant influence on the observations. Significant main effects correspond to explanatory variables that contribute on their own. Significant interaction terms correspond to subsets of variables whose joint inclusion in the model provides explanation.

In multiple linear regression, then these results are implicit in significance tests on the coefficients. But if one using one of the nonparametric regression techniques poplular in data mining (e.g., MARS, PPR, neural nets), this helps find influential variables.

Based on the results of the designed experiment, one can ultimately find and fit the model that includes all variables corresponding to significant main effects or interactions. And the factorial design reduces the penalty one pays for multiple testing, as compared to exhaustive search or other less-efficient searches.

Possible measures of goodness-of-fit include:

- R^2, the proportion of variance in the observations that is explained by the model;
- Adjusted R^2, the proportion of variance in the observations that is explained by the model, but with an adjustment to account for the number of variables in the model;
- Mallows' C_p, a measure of predictive accuracy that takes account of the number of terms in the model.
- MISE, the mean integrated squared error of the fitted model over a given region (often the hyperrectangle defined by the minimum and maximum values taken by each explanatory variable used in the model.
- The square root of the adjusted R^2, since this transformation appears to stabilize the variance and thereby supports use of analysis of variance and response surface methodology in the model search.

Weisberg (1985, *Applied Linear Regression*, 185-190, Wiley) discusses the first three. Scott (1992, *Multivariate Density Estimation*, chapter 2.4, Wiley) discusses MISE.

7.4 List Search

With list search, there is no exploitable structure that links the elements of the list, and the list is usually so long that exhaustive search is infeasible.

There is not much that one can do. If one tests entries on the list at random, then one can try some of the following:

- Estimate the proportion of list entries that give results above some threshold.
- Use some modeling to estimate the maximum value on the list from a random sample of list entries.
- Estimate the probability that further search will discover a new maximum within a fixed amount of time.

A strategy invented by computer scientists (Maron and Moore, 1997, *Artificial Intelligence Review*, **11** 193-225) is to **race** the testing.

One does pairwise comparisons of models. At first, one fits only a small random fraction of the data (say a random 1%) to each model on the list. Usually this is sufficent to discover which model is best and one discards the other.

If that small fraction does not distinguish the models, then one fits another small fraction. Only very rarely is it necessary to fit all mor most of the data to select the better model.

Racing is an easy way to extend one's search capability by about 100-fold.

7.5 Bayesian Methods

Suppose there are p explanatory variables and one considers a model of the form:

$$\mathcal{M}_i : \boldsymbol{Y} = \boldsymbol{X}_i^* \boldsymbol{\beta}_i^* + \boldsymbol{\epsilon}$$

where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2 \boldsymbol{I})$ and $i = 1, \ldots, 2^p$, indexing all possible choices of inclusion or exclusion among the p explanatory variables.

Assume there is an intercept: each $\boldsymbol{\beta}_i^* = (\beta_0, \boldsymbol{\beta}_i')$, and thus let $\boldsymbol{X}_i$ be the submatrix of $\boldsymbol{X}_i^*$ corresponding to $\boldsymbol{\beta}_i$, and let the length of $\boldsymbol{\beta}_i$ be $d(i) + 1$. Denote the density corresponding to model $\mathcal{M}_i$ by $f_i(\boldsymbol{y} | \boldsymbol{\beta}_i^*, \sigma^2)$.

For this model, standard choices for the prior on the parameters in $\mathcal{M}_i$ include:

- The g-priors. Here:

$$\pi_i^g(\boldsymbol{\beta}_i, \sigma^2) = \frac{1}{\sigma^2} N(\boldsymbol{\beta}_i | \mathbf{0}, cn\sigma^2 (\boldsymbol{X}'_i \boldsymbol{X}_i)^{-1})$$

  where c is fixed (typically to 1, or estimated through empirical Bayes).

- Zellner-Siow priors. The intercept model has prior $\pi(\beta_0, \sigma^2) = 1/\sigma^2$, while the prior for all other $\mathcal{M}_i$ is:

$$\begin{aligned} \pi_i^{ZS}(\boldsymbol{\beta}_i, \sigma^2) &= \pi_i^g(\boldsymbol{\beta}_i, \sigma^2 | c) \\ \pi_i^{ZS}(c) &\sim \textbf{InverseGamma}(c | 0.5, 0.5). \end{aligned}$$

The **marginal density** under $\mathcal{M}_i$ is

$$m_i(\boldsymbol{Y}) = \int f_i(\boldsymbol{y} | \boldsymbol{\beta}_i, \sigma^2) \pi(\boldsymbol{\beta}_i, \sigma^2) d\boldsymbol{\beta}_i d\sigma^2.$$

If all models $\mathcal{M}_i$ have equal prior probability, then the posterior probability of a model is

$$P(\mathcal{M}_i|\boldsymbol{Y}) = \frac{m_i(\boldsymbol{y})}{\sum_k m_k(\boldsymbol{Y})}.$$

Recall the posterior inclusion probability for the ith variable:

$$\begin{aligned} q_i &= P(\beta_i \in \textbf{ correct model } |\boldsymbol{Y}) \\ &= \sum_k P(\mathcal{M}_k|\boldsymbol{Y}) I_{\beta_i \in \mathcal{M}_k}. \end{aligned}$$

The **median probability model** is the model consisting of all and only those variables whose posterior inclusion probability is at least 1/2.

Usually, the median probability model is superior to the maximum posterior probability model for prediction. (But the Bayesian model average is better than both, if one can use an ensemble method.)

Theorem: (Barbieri and Berger, 2004, Annals of Statistics. Consider a sequence of nested linear models. If

- prediction is wanted at "future covariates like the past",
- the posterior mean under $\mathcal{M}_i$ satisfies $\tilde{\boldsymbol{\beta}}_i = b\hat{\boldsymbol{\beta}}_i$, where $\hat{\boldsymbol{\beta}}_i$ is the OLS estimate,

then the best single model for prediction under squared error loss is the median probability model.

The second condition is satisfied if one uses either noninformative priors for the model parameters or if one uses g-type priors with the same constant $c > 0$ for each model (and any prior on σ^2).

Example: Polynomial regression. Here $\mathcal{M}_i$ is

$$y = \sum_{j=0}^{i} \beta_j x^j + \epsilon.$$

Model	0	1	2	3	4	5	6
$\mathrm{P}(\mathcal{M}_i \mid \boldsymbol{Y})$	≈ 0	.06	.22	.29	.38	.05	≈ 0

Covariate j	0	1	2	3	4	5	6
P(x^j is in model$\mid \boldsymbol{Y}$)	≈ 1	≈ 1	.94	.72	.33	.05	≈ 0

The correct model was $\mathcal{M}_3$, which is the median probability model. But the MAP model is $\mathcal{M}_4$.

For variable selection, the model space is so large that one needs to use "guided search". One finds some reasonably good models, and then searches more intensively in their neighborhoods.

At any stage, and letting $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}$ denote the models previously visited:

- Define the current estimated posterior probabilities of models (assuming equal prior probabilities) as

$$\hat{P}(\mathcal{M}_i|\boldsymbol{Y}) = \frac{m_i(\boldsymbol{Y})}{\sum_{\ell=1}^{k} m_\ell(\boldsymbol{Y})}.$$

- Define the current estimated variable inclusion probabilities (estimated over all posterior probabilities that variables are in the model) as

$$\begin{aligned} q_j &= \hat{P}[(\beta_j \in \textbf{ correct model } |\boldsymbol{Y}) \\ &= \sum_{\ell=1}^{k} \hat{P}[\mathcal{M}_\ell|\boldsymbol{Y}) I_{\beta_i \in \mathcal{M}_j}. \end{aligned}$$

1. At iteration k, compute the current posterior model and inclusion probability estimates $\hat{P}(\mathcal{M}_i|\boldsymbol{y})$ and q_j.

2. Return to one of the $k-1$ (distinct) previous models already visited, in proportion to their estimated probabilities $\hat{P}(\mathcal{M}_i|\boldsymbol{Y})$.

3. Add/remove a variable with probability 1/2.

   - if adding a variable, choose the jth variable with probability $\propto \frac{q_j+c}{1-q_j+c}$
   - if removing a variable, choose the jth with probability $\propto \frac{1-q_j+c}{q_j+c}$.

   Here c is a tuning parameter that keeps q_j away from zero or one; $c = .01$ works well.

4. if the obtained in step 3

   - has already been visited, return to step 2.
   - is new, update $\hat{P}(\mathcal{M}_i|\boldsymbol{Y})$ and q_j and go to step 2.

Example: Recall the ozone pollution data used in the ACE model by Breiman and Friedman (1985). Berger and Barbieri (200?) refit it.

The ozone data has 178 observations and 10 covariates. Berger and Barbieri considered linear models with an intercept and all possible main effects, quadratic terms, and two-way product interactions, for a total of 65 covariates and $2^{65} \approx 10^{19}$ models.

They used the g-priors and Zellner-Siow priors, for computational reasons.

They implemented the algorithm at different starting points (random restart) and found essentially no difference in the results. They made 5,000,000 iterations and saved only the best 65,536 models.

No model had appreciable posterior probability; the largest was 0.0025.

The following (partial) table shows the estimated posterior probabilities for variable inclusion in the ozone problem:

variable	g-prior	ZSN prior
x1	.86	.94
x2	.05	.06
x3	.03	.03
x4	.99	.99
x5	.20	.31
x6	.19	.35
x7	.20	.21
x8	.96	.97
x1*x1	.99	.99
x1*x2	.57	.73
x1*x7	.08	.14
x6*x8	.78	.86

Note that the same principle can be used for any search criterion and for any model structure—it doesn't have to be Bayesian.

- Choose the model features (variables, graphical nodes, structure) you want to use to drive the search.
- Choose the criterion for defining a good model (e.g., the AIC).
- Convert the criterion to pseudo-probabilities for the models, e.g.,

$$\hat{P}(\mathcal{M}_i|\boldsymbol{Y}) \propto \exp(\mathbf{AIC}_i/2).$$

- Define the feature inclusion probabilities as

$$\begin{aligned} q_j &= \hat{P}[\ \mathbf{feature}_j \in\ \mathbf{correct\ model}\ |\boldsymbol{Y}] \\ &= \sum_{i=1}^{k} \hat{P}(\mathcal{M}_j|Y) I_{\ \mathbf{feature}_{j \in \mathcal{M}_i}}. \end{aligned}$$

- Apply the search algorithm.

Observations on high-dimensional search strategies:

- Effective search in large, non-orthogonal situations requires guidance; revisiting high probability models and changing variables in their neighborhood according to posterior inclusion probabilities appears to be effective.
- There may be no model with significant posterior probability (e.g., the largest for the ozone data was 0.0025). And there may be many thousands with roughly comparable posterior probability.
- In that case it is more important to discover interpretable insight, such as the posterior probabilities for inclusion of each of the variables.