Building a Prognostic Biomarker

Noah Simon and Richard Simon

July 28, 2016

Prognostic Biomarker for a Continuous Measure

On each of n patients measure

y_i - single continuous outcome

(eg. blood pressure, tumor growth)

$\mathbf{x}_i$ - p-vector of features

(eg. SNPs, gene expression values)

Want to model y_i by $\mathbf{x}_i$ to

- Predict high risk patients (give them additional care)
- Learn the underlying biology

An Oldie But a Goodie

Linear Regression:

We assume that

$$y_i = \beta_0 + x_i^\top \beta + \epsilon_i$$

Generally fit by solving:

$$\min_{\beta_0,\beta} \frac{1}{2} \sum \left(y_i - \beta_0 - x_i^\top \beta \right)^2$$

An Oldie But a Goodie

Diabetes data example

$n = 442$, $p = 10$

y_i is quantitative measure of disease progression (one year after baseline)

$\mathbf{x}_i$ includes age, sex, BMI, avg BP, and six serum measurements

An Oldie But a Goodie

Can use $\eta_i = \hat{\beta}_0 + \hat{\beta}^\top x_i$ to predict risk.

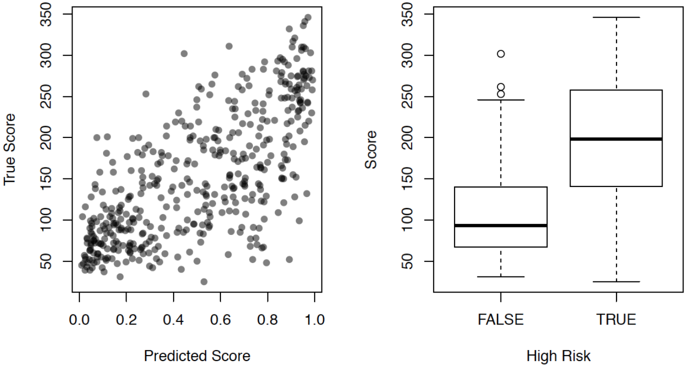
Or can stratify

$$\eta_i \geq \mathit{cutoff} \rightarrow \text{high risk}$$

$$\eta_i < \mathit{cutoff} \rightarrow \text{low risk}$$

Choosing the *cutoff* can be tricky

An Oldie But a Goodie



An Oldie But a Goodie

Two Issues

When $p \sim n$ (or $p > n$), estimate is highly variable

When true model is far from linear, estimate is inflexible

Working in High Dimensions

For $p \sim n$ we need an often reasonable assumption:

Most of the features are [conditionally] unrelated to response

---

More formally: In the model

$$y_i = \beta_0 + x_i^\top \beta + \epsilon_i$$

Most of the β_j are 0 (or very nearly 0).

Bet on Sparsity

Is this assumption reasonable?

Often

If not, statistical trickery generally will not help.

Either need more samples

Or more benchwork

Taking Advantage of Sparsity

How do we fit a sparse model?

Most obvious approach is:

$$\begin{aligned} \text{minimize}_{\beta_0,\beta} \quad & \sum_i \left(y_i - \beta_0 - x_i^\top \beta\right)^2 \\ \text{subject to} \quad & \#\left\{j \,|\beta_j \neq 0\right\} \leq d \end{aligned}$$

Unfortunately, this is computationally intractable.

Taking Advantage of Sparsity

How do we fit a sparse model?

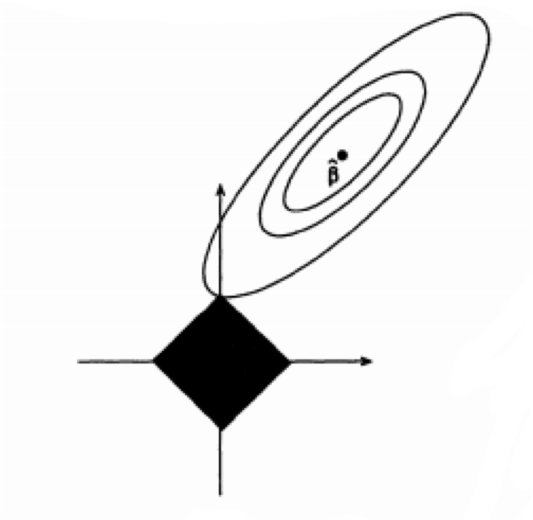
Instead we use the *Lasso*

$$\begin{aligned} &\text{minimize}_{\beta_0,\beta} \quad && \sum_i \left(y_i - \beta_0 - x_i^\top \beta\right)^2 \\ &\text{subject to} && \sum_j |\beta_j| \leq c \end{aligned}$$

This can be solved as fast as (or faster than) least squares.

Taking Advantage of Sparsity

How does this give us sparsity?



Decreasing c increases sparsity.

Taking Advantage of Sparsity

Equivalent Penalized Form

$$\text{minimize}_{\beta_0,\beta} \sum_i \left(y_i - \beta_0 - x_i^\top \beta\right)^2 + \lambda \sum_j |\beta_j|$$

c in *constrained form* $\longleftrightarrow$ λ in *penalized form*

Choosing λ

In practice everyone uses

Training/test validation

OR

Cross-validation

Training/Test Validation

1. Choose candidate λ-values: $\lambda_1, \ldots, \lambda_M$

2. Split observations into two groups: training and test

---

3. For each candidate $m \le M$

   3.1 Training Data $\rightarrow$ Build model with λ_m

   3.2 Test Data $\rightarrow$ Apply model to get "predictions"

---

4. Evaluate the predictions for each model choose the best.

Training/Test Validation

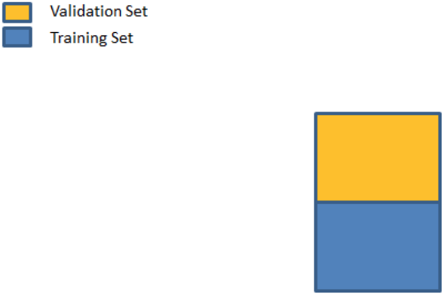
Evaluating each model

For each λ_m we have $\hat{y}_i^{(m)}$ $i = 1, \ldots, n_{test}$.

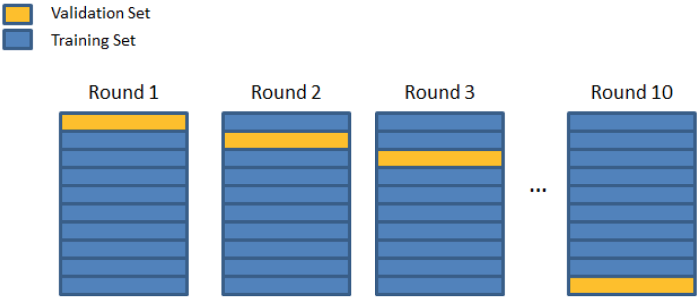
Simplest evaluation via mean-square-error:

$$\text{performance}_m = \sum_{i \in \text{test data}} \left(y_i - \hat{y}_i^{(m)} \right)^2$$

Training/Test Validation



Cross-Validation



Cross-Validation

1. Choose candidate λ-values: $\lambda_1, \ldots, \lambda_M$

2. Split observations into K folds:

---

3. For each candidate $m \leq M$, and each fold $k \leq K$

   3.1 Data (minus fold k) → Build model with λ_m

   3.2 Data (fold k) → Apply model to get "predictions"

---

4. Evaluate models (now on all data)

Making Linear Regression Less Linear

What if the relationship isn't linear?

$$\begin{aligned} y &= 3\sin(x) + \epsilon \\ y &= 2e^x + \epsilon \\ y &= 3x^2 + 2x + 1 + \epsilon \end{aligned}$$

If we know the functional form we can still use "linear regression"

Making Linear Regression Less Linear

$y = 3\sin(x) + \epsilon$:

$$\begin{pmatrix} x \end{pmatrix} \rightarrow \begin{pmatrix} sin(x) \end{pmatrix}$$

$y = 3x^2 + 2x + 1 + \epsilon$:

$$\begin{pmatrix} x \end{pmatrix} \rightarrow \left(\begin{array}{c|c} x & x^2 \end{array}\right)$$

Making Linear Regression Less Linear

What if we don't know the right functional form?

Use a flexible basis expansion:

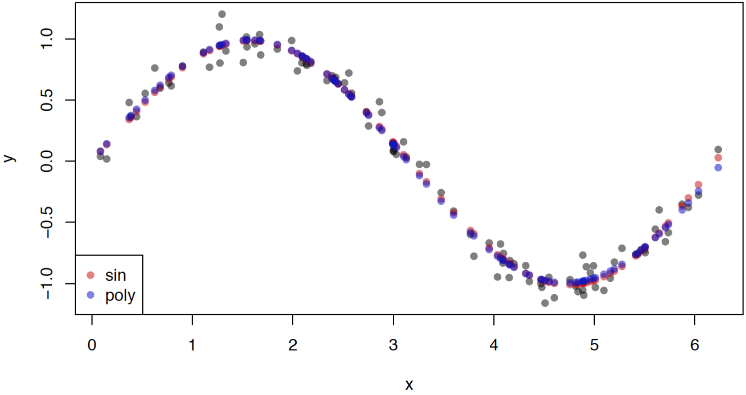
- polynomial basis

$$\left(\, x \,\right) \rightarrow \left(x \;\middle|\; x^2 \;\middle|\; \cdots \;\middle|\; x^k \right)$$

- hockey-stick (/spline) basis

$$\left(\, x \,\right) \rightarrow \left(x \;\middle|\; (x - t_1)_+ \;\middle|\; \cdots \;\middle|\; (x - t_k)_+ \right)$$

Making Linear Regression Less Linear



Making Linear Regression Less Linear

For high dimensional problems, expand each variable

$$\left(\begin{array}{c|c|c|c} x_1 & x_2 & \cdots & x_p \end{array}\right) \to \left(\begin{array}{ccc|ccc|c|ccc} x_1 & \cdots & x_1^k & x_2 & \cdots & x_2^k & \cdots & x_p & \cdots & x_p^k \end{array}\right)$$

and use the *Lasso* on this expanded problem.

---

k must be small ($\sim$ 5ish)

Spline basis generally outperforms *polynomial*

Prognostic Biomarker for a Binary Measure

On each of n patients measure

y_i - single binary outcome

(eg. Progression after a year, pCR)

$\mathbf{x}_i$ - p-vector of features

(eg. SNPs, gene expression values)

More common than continuous response

Logistic Regression

Relate it back to continuous methods:

For continuous response solve:

$$\text{minimize}_{\beta,\beta_0} \sum_i \left(y_i - \beta_0 - x_i^\top \beta \right)^2$$

One interpretation:

If

$$y_i | x_i \sim N\left(x_i^\top \beta, \sigma^2 \right)$$

then maximizing likelihood is equivalent to least squares.

Logistic Regression

Relate it back to continuous methods:

For Binary Response, consider

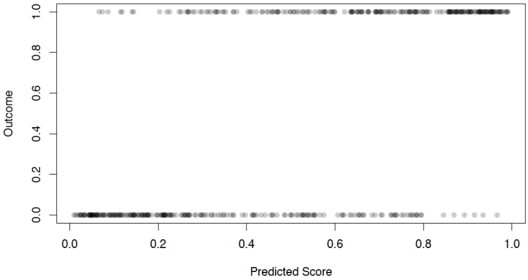
$$y_i|x_i \sim \text{ber}\left(p_i = \frac{e^{\beta_0 + x_i^\top \beta}}{1 + e^{\beta_0 + x_i^\top \beta}}\right)$$

Maximizing likelihood is equivalent to minimizing

$$\ell\left(\beta, \beta_0\right) = -\sum_i \left[y_i\left(\beta_0 + x_i^\top \beta\right) - \log\left(1 + e^{\beta_0 + x_i^\top \beta}\right)\right]$$

This is just logistic regression

Diabetes Example



Additionally:

166/221 test-positive vs 55/221 test-negative

patients with above median progression

Penalized Logistic Regression

As in least squares, we can induce sparsity:

$$\text{minimize}_{\beta,\beta_0} \, \ell\left(\beta, \beta_0\right) + \lambda \sum |\beta_j|$$

Penalized Logistic Regression

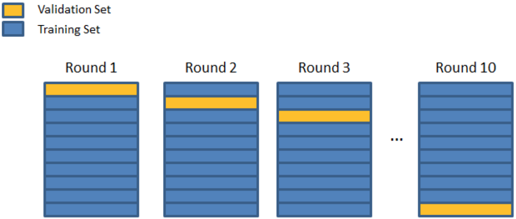
As in least squares, we can induce sparsity:

$$\text{minimize}_{\beta,\beta_0}\, \ell\,(\beta, \beta_0) + \lambda \sum |\beta_j|$$

Choosing λ is a bit tricky.

We need a measure of the performance of our model

Choosing λ



Using CV, we get

$$\hat{\eta}_i = \hat{\beta}_0^{train} + x_i^\top \hat{\beta}^{train}$$

Choosing λ

For each patient we have a CV score

$$\hat{\eta}_i = \hat{\beta}_0^{\,train} + x_i^\top \hat{\beta}^{train}$$

Now plug-in

$$\hat{p}_i = \frac{e^{\hat{\eta}_i}}{1 + e^{\hat{\eta}_i}}$$

And use Cross-Validated Predictive Likelihood as our measure

$$\prod_i \hat{p}_i^{y_i} (1 - \hat{p}_i)^{1-y_i}$$

(Some software equivalently uses the negative log-likelihood)

Choosing λ

Can also classify patients using $\hat{p}_i$:

$$\widehat{class}_i = \begin{cases} 1 & : \hat{p}_i \geq 0.5 \\ 0 & : \hat{p}_i < 0.5 \end{cases}$$

And use Cross-Validated Misclassification Rate as our measure

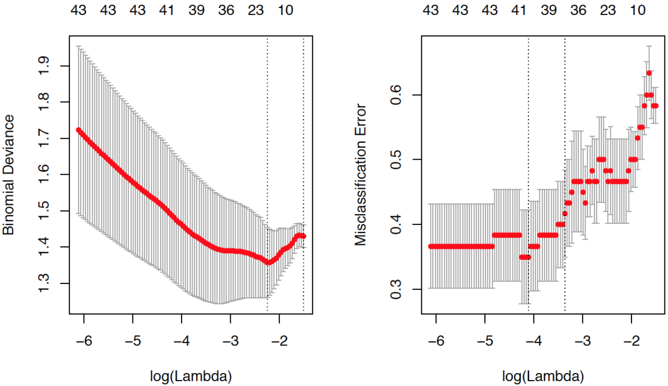
$$proportion\left(y_i \neq \widehat{class}_i\right)$$

Example

Prognostic Biomarker for pCR of HER2+ breast cancer patients on Herceptin + CT

- $n = 60$ patients, with 28 pCR
- Expression from $p = 5349$ genes with non-negligable variance
- GEO: GSE50948

Example


43 43 43 41 39 36 23 10
Binomial Deviance
1.3
1.4
1.5
1.6
1.7
1.8
1.9
−6
−5
−4
−3
−2
log(Lambda)
43 43 43 41 39 36 23 10
Misclassification Error
0.3
0.4
0.5
0.6
−6
−5
−4
−3
−2
log(Lambda)

Some Other Classifiers

Many other classification choices. Noteworthy high dimensional options:

Diagonal Linear Discriminant Analysis (DLDA)

Nearest Shrunken Centroid (PAM)

Support Vector Machine (SVM)

Find a *score* based on features: $x_i \rightarrow x_i^\top \beta$ indicating likelihood of each class

DLDA

Assumes:

Gaussian features within class With Pooled Diagonal Covariance:

$$(\mathbf{x_i}|y_i = 0) \sim \mathrm{N}(\mu_0, D) \qquad (\mathbf{x_i}|y_i = 1) \sim \mathrm{N}(\mu_1, D)$$

Estimate μ_1, μ_2, D by maximum likelihood.

Calculate Probability of each class using Bayes and plug-in.

Score is:

$$\eta_i = \hat{D}^{-1} \left(\hat{\mu}_1 - \hat{\mu}_0\right)^\top x_i$$

PAM

Assumes:

Gaussian features within class With Pooled Diagonal Covariance:

$$(\mathbf{x_i}|y_i = 0) \sim \mathrm{N}\left(\mu_0, D\right) \qquad (\mathbf{x_i}|y_i = 1) \sim \mathrm{N}\left(\mu_1, D\right)$$

Estimate μ_1, μ_2 with shrinkage!

$$\tilde{\mu}_{1j} = \tilde{\mu}_{2j} \quad \text{for most } j$$

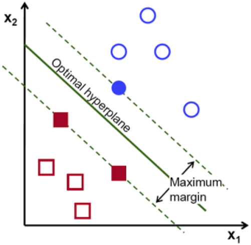
Score is:

$$\eta_i = \hat{D}^{-1}\left(\tilde{\mu}_1 - \tilde{\mu}_0\right)^\top x_i$$

SVM

No statistical model: just discriminant method.

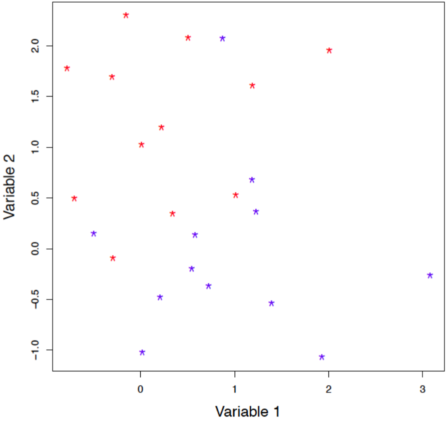
Finds the maximum-margin separating hyperplane



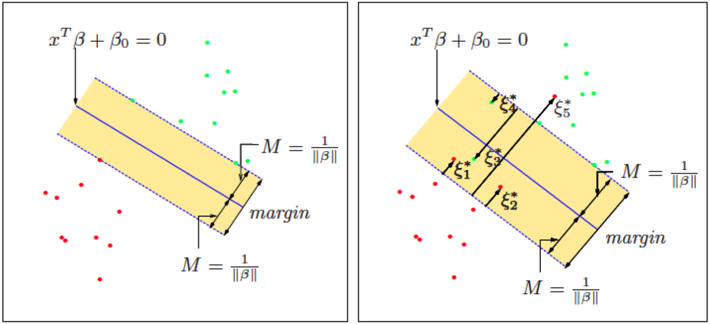
Score is (signed) distance from hyperplane

$$\eta_i = \beta^\top x_i + \beta_0$$

What if There is No Separating Hyperplane?



Support Vector Classifier: Allow for Violations



Support Vector Machine

- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels”.

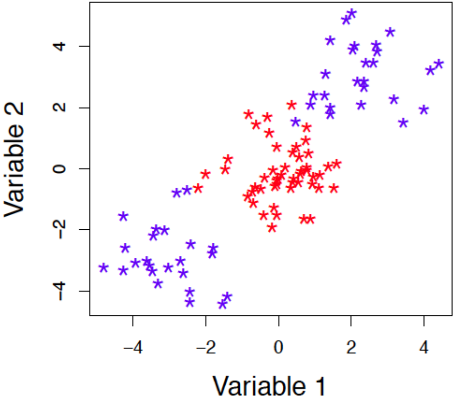
Support Vector Machine

- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels”.
- However, linear regression, logistic regression, and other classical statistical approaches can also be applied to non-linear functions of the variables.

Support Vector Machine

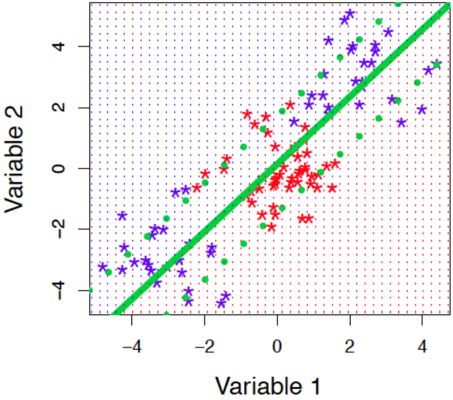
- The support vector machine is just like the support vector classifier, but it elegantly allows for non-linear expansions of the variables: “non-linear kernels”.
- However, linear regression, logistic regression, and other classical statistical approaches can also be applied to non-linear functions of the variables.
- For historical reasons, SVMs are more frequently used with non-linear expansions as compared to other statistical approaches.

Non-Linear Class Structure



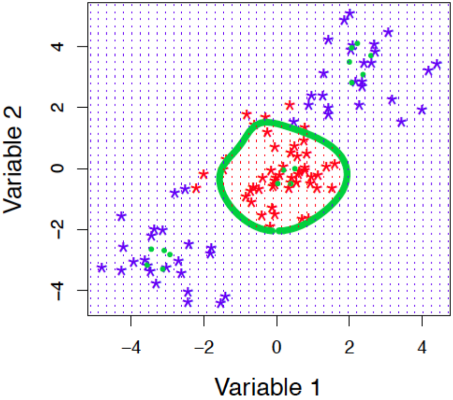
This will be hard for a linear classifier!

Try a Support Vector Classifier



Uh-oh!!

Support Vector Machine



Much Better.

Prognostic Biomarker for Survival Data

On each of n patients measure

(t_i, s_i) - time, censoring-status

(eg. Disease free survival)

$\mathbf{x}_i$ - p-vector of features

(eg. SNPs, gene expression values)

Prognostic Biomarker for Survival Data

Hazard is

> probability density of failure at time t given survival up to t.

Want to model the hazard as a function of covariates

Prognostic Biomarker for Survival Data

We will use *Cox Proportional Hazards Model*

Assumes hazard is

$$\lambda(t) = h(t)e^{x_i^\top \beta}$$

where

$h(t)$ is covariate-independent baseline hazard

$e^{x_i^\top \beta}$ is covariate-based tilt

Prognostic Biomarker for Survival Data

Considering Partial Likelihood — likelihood at only event times

$h(t)$ falls out.

$$L(\beta) = \prod_{i \in D} \frac{e^{x_i^\top \beta}}{\sum_{j \in R_i} e^{x_j^\top \beta}}$$

Maximizing this is equivalent to minimizing

$$\ell(\beta) = -\sum_{i \in D} \left[x_i^\top \beta - \log \left(\sum_{j \in R_i} e^{x_j^\top \beta} \right) \right]$$

Bells and Whistles

Similar additions as continuous/binary response

$$\text{minimize}_\beta \, \ell(\beta) + \lambda \sum |\beta_j|$$

Trickiest for choosing λ yet.

Straightforward CV approach

Find CV estimate for each patient

$$\hat{\eta}_i = x_i^\top \hat{\beta}^{train}$$

Calculate CV predictive partial likelihood

$$\prod_{i \in D} \frac{e^{\hat{\eta}_i}}{\sum_{j \in R_i} e^{\hat{\eta}_j}}$$

Not necessarily a great measure