Feature Selection Methods in Predixion Insight™ 4.0

Version 4.0.1

Predixion Software
15 Enterprise Road #300, Aliso Viejo, CA 92656
Phone: 949.373.4900
Website: http://www.predixionsoftware.com
Support: support@predixionsoftware.com

© 2015 Predixion Software. All rights reserved.
This document describes the feature selection algorithms introduced in Predixion Insight 4.0 (OLS, Lasso and Laplacian Score) and Predixion Insight 3.2 (Information Gain), including how each algorithm calculates the score of each column or feature to rank them. The algorithms in Predixion insight 4.0 are better suited for large numbers of columns or predictors than ones included in earlier versions of Predixion Insight.

**Preliminary Definitions**

The section describes the notation used in this document.

Assume $\mathbf{Y}$ and $\mathbf{X}$ are jointly distributed random variables taking values in $\mathbb{R}$ and $\mathbb{R}^p$. Linear regression may be viewed as a linear estimate of the function on $\mathbb{R}^p$.

$$f(x; \beta) = E(\mathbf{Y} | \mathbf{X} = x),$$

where $\beta$ is a tuning parameter (viz., hyper-parameter).

When the above random variables have sufficient sample observations

$$\{y_1, ..., y_n\} \subset \mathbb{R}$$
$$\{x_1, ..., x_n\} \subset \mathbb{R}^p,$$

the linear estimate of $f(x; \beta)$ may be reformulated in terms of case table $X$ and response $Y$ as

$$Y = f(X; \beta) = X\beta + \epsilon,$$

where

$$Y = (y_1, ..., y_n)^T \in \mathbb{R}^n$$
$$X = (x_1, ..., x_n)^T \in (\mathbb{R}^p)^n = \mathbb{R}^{p \times n}.$$

In terms of features or columns of $X$

$$X = (X_1, ..., X_p)$$
$$Y = \sum_{j=1}^{p} \beta_j X_j + \epsilon.$$

Assume the features and response have been normalized to mean 0 and sum of squares 1 and treat the dependence on the stochastic component $\epsilon$ as implicit.

**Feature Selection**

Feature selection may be defined as seeking the smallest set of features $X_{j_1}, ..., X_{j_k}$ that best describe the response, in terms of minimizing the error on new data for which the model was not explicitly trained, viz., minimizing generalization error.

The rationale for smallest is that generalization error tends to grow out of irrelevant or noisy features.

**Notation**

The canonical inner product or dot product is denoted
Due to the normalization assumption this is equivalent to correlation (and to covariance when n is large ~1000.)

Denote the $L_1$ and $L_2$ norms by

$$
\|x\|_1 = \sum_{i=1}^{n} |x_i|
$$

$$
\|x\|_2 = \sqrt{\langle x, x \rangle}.
$$

Orthogonal Least Squares (OLS)

This feature selection technique is sometimes called Forward Stepwise Selection, or Gram-Schmidt. From a purely feature selection perspective it is equivalent to Orthogonal Matching Pursuit or Forward Stagewise Regression – though the regression coefficients obtained are different. (cf. [1], section 3.3).

**OLS Algorithm**

For $k=1, 2, ...$ choose

$$
score(X_j) = |\langle X_j, Y \rangle| = \max_{1 \leq i \leq p} |\langle X_i, Y \rangle|.
$$

$$
Y \leftarrow Y - \langle X_j, Y \rangle X_j
$$

renormalize $Y$

Lasso

This feature selection technique is sometimes termed Basis Pursuit Denoising. Consider ($for \lambda > 0$)

$$
\mathcal{L}(\beta) = \frac{1}{2} \|Y - \beta X\|_2^2 + \lambda \|\beta\|_1
$$

Lasso may be interpreted as stepwise regression toward the goal
Feature Selection Methods in Predixion Insight™ 4.0

\[ \min_{\beta} \mathcal{L}(\beta). \]

Sketch:

- **Apply the gradient operator**
  \[ \nabla \mathcal{L}(\beta) = -\langle \nabla X \beta, Y - X \beta \rangle + \lambda \nabla \| \beta \|_1 = \langle X, X \beta - Y \rangle + \lambda \text{sgn}(\beta) \]

- **Set \( \nabla \mathcal{L}(\beta) = 0 \), thus**
  \[ \langle X, X \beta \rangle = \langle X, Y \rangle - \lambda \text{sgn}(\beta) \]

- **With j^{th} components**
  \[ \langle X_j, X \beta \rangle = \langle X_j, Y \rangle - \lambda \text{sgn}(\beta_j) \]
  \[ \langle X_j, X \beta_j + \sum_{k \neq j} X_k \beta_k \rangle = \langle X_j, Y \rangle - \lambda \text{sgn}(\beta_j) \]

- **Hence \( \| X_j \|_2 = 1 \) implies**
  \[ \beta_j = \langle X_j, R(\beta, j) \rangle - \lambda \text{sgn}(\beta_j) \]

- **Where \( R(\beta, j) \) is the partial residual**
  \[ R(\beta, j) = \langle X_j, Y - \sum_{k \neq j} X_k \beta_k \rangle \]

A computable representation of \( \beta_j \) is obtained by observing that

- \( \beta_j > 0 \iff R(\beta, j) > \lambda \)
- \( \beta_j < 0 \iff R(\beta, j) < -\lambda \)

Thus

\[ \beta_j = S(R(\beta, j), \lambda) \]

Where \( S(x, \lambda) \) is the soft threshold operator

\[ S(x, \lambda) = \begin{cases} 
  x - \lambda : x > \lambda \\
  x + \lambda : x < -\lambda \\
  0 : -\lambda \leq x \leq \lambda 
\end{cases} \]

**Lasso Stepwise Regression**

1. Let \( \lambda_{max} = \max_{1 \leq i \leq p} |\langle X_i, Y \rangle| \), then \( \beta_j = 0 \), \( \forall j \)
2. Choose \( \lambda_{min} = \lambda_K < \lambda_{K-1} < \lambda_{K-2} < \cdots < \lambda_0 = \lambda_{max} \)
   \[ \lambda_{min} = \delta \lambda_{max}, \lambda_{t+1} = \delta^{1/K} \lambda_t; \ e.g. \ K = 100, \delta = 0.001 \]
3. Then \( \tilde{\beta} = 0 \) is a solution for \( \lambda = \lambda_0 \), thus given a solution \( \tilde{\beta} \) that solves \( \lambda = \lambda_t \)
   solve for \( \lambda = \lambda_{t+1} \) by setting
   \[ \beta_j = S(R(\tilde{\beta}, j), \lambda_{t+1}) \]
and sequentially iterating $j=1,...,p$ while updating $\tilde{\beta}$ with the previous (step $j-1$) result until (logarithmic) convergence.

4. At early stages $\lambda_l \rightarrow \lambda_{l+1}$ the solution $\beta$ is sparse, in fact ideal steps between $\lambda$ which add but a single feature per stage are easily achieved (cf. LAR [1], section 3.4.4).

5. At a given stage $\lambda_l \rightarrow \lambda_{l+1}$ features may be removed, so it is important to find a stopping point at a given feature count before delineating a ranking of features (e.g. 5-fold cross-validation.)

6. $score(X_j) = |R(\beta, j)|$ where the Lasso stepwise regression has been stopped at its goal feature count.

Lasso can become sub-optimal in the presence of excessive correlation amongst features, so the implementation is actually of the Elastic Net (cf. [3]), which minimizes

$$\mathcal{E}(\beta) = \frac{1}{2} \|Y - \beta X\|_2^2 + \lambda \left( \alpha \|\beta\|_1 + \frac{1-\alpha}{2} \|\beta\|_2^2 \right).$$

The solution technique is identical to Lasso. The value of $\alpha = 0.925$ has been found to work quite well for feature selection.

Laplacian Score

Laplacian Score is fundamentally based on Laplacian Eigenmaps and Locality Preserving Projection. The basic idea of is to evaluate the features according to their locality preserving power (cf. [3]). Heuristically Laplacian score can also be viewed as related spectral clustering.

Form a weight matrix from the rows of the case table $X$

$$S = (S_{ij})$$

$$S_{ij} = \exp \left( -0.5 \|x_i - x_j\|_2^2 \right)$$

$$S_i = \sum_{j=1}^{n} S_{ij}$$

$$D = (D_{ij})$$

$$D_{ij} = \begin{cases} S_i : i = j \\ 0 : i \neq j \end{cases}$$

$$L = D - S$$

$$score(X_j) = 1 - \langle X_j, LX_j \rangle / \langle X_j, DX_j \rangle$$

In fact, $S_{ij}$ should be set to 0 when $x_i, x_j$ are not in the same k-nearest neighbors partition ($k=3$ seems to work well). (cf. [3])
Information Gain

The implementation of information gain is a filtering technique which uses certain classical statistical tests to rank features. Only a single ranking pass takes place with no forward selection, hence redundant features are likely to be chosen.

Note: The assumption that the case table (and response) is normalized, will be dropped.

Filtering Statistics

Correlation (viz., Pearson)

In a supervised context, \( (\text{viz.}, Y = f(X; \beta)) \):

\[
\text{score}(X_j) = \frac{|\langle X_j, Y \rangle|}{\|X_j\|_2 \|Y\|_2}.
\]

Variance Ratio (F statistic)

In a supervised context, form the conditional distribution \( X_j \mid Y \), then

\[
\text{score}(X_j) = 1 - \frac{\text{Var}(X_j \mid Y)}{\text{Var}(Y)}.
\]

When unsupervised (e.g. clustering), consider a continuous feature \( X_j \). Conditional upon \( X_j \) construct the quintile distributions \( X_j^1, X_j^2, X_j^3, X_j^4, X_j^5 \) and form the F statistics and score:

\[
F(X_j^k) = \frac{\text{Var}(X_j^k)}{\text{Var}(X_j)}
\]

\[
\text{score}(X_j) = 1 - \min_k F(X_j^k).
\]

Entropy + Chi-Squared Contingency Table

Assume \( X_j, Y \) are two Bernoulli distributions and let \( p = E(X_j \mid Y) \), then the Shannon entropy of \( X_j \mid Y \) as well as score, can be realized as:

\[
H(X_j \mid Y) = 4p(1 - p),
\]

\[
\text{score}(X_j) = 1 - H(X_j \mid Y).
\]

However, if the chi-squared contingency table test for the null hypothesis of \( X_j, Y \) independence, cannot be rejected at the 5% level, then

\[
\text{score}(X_j) = 0.
\]

When unsupervised, \( X_j \) Bernoulli:

\[
\text{score}(X_j) = 1 - H(X_j) = 1 - 4p(1 - p).
\]
Score of a nested key from a nested table

The binarization of a nested table creates a sequence of nested key columns, each of which is Bernoulli and typically quite sparse. Define the score of a nested key to simply be the Bernoulli probability p.

Filtering and Feature Scoring

1. Supervised
   a. $X_j$ continuous and Y continuous – use correlation scoring.
   b. $X_j$ continuous and Y discrete -- use correlation scoring of the discrete state value with the largest score.
   c. $X_j$ discrete and Y discrete – use entropy and chi-squared contingency table scoring of the discrete state value with the largest score.
   d. $X_j$ discrete and Y continuous – use variance ratio.
   e. $X_j$ nested key – use Bernoulli probability p.

2. Unsupervised
   a. $X_j$ continuous – use conditional quintile variance ratio scoring.
   b. $X_j$ discrete – use entropy scoring.
   c. $X_j$ nested key – use Bernoulli probability p.

