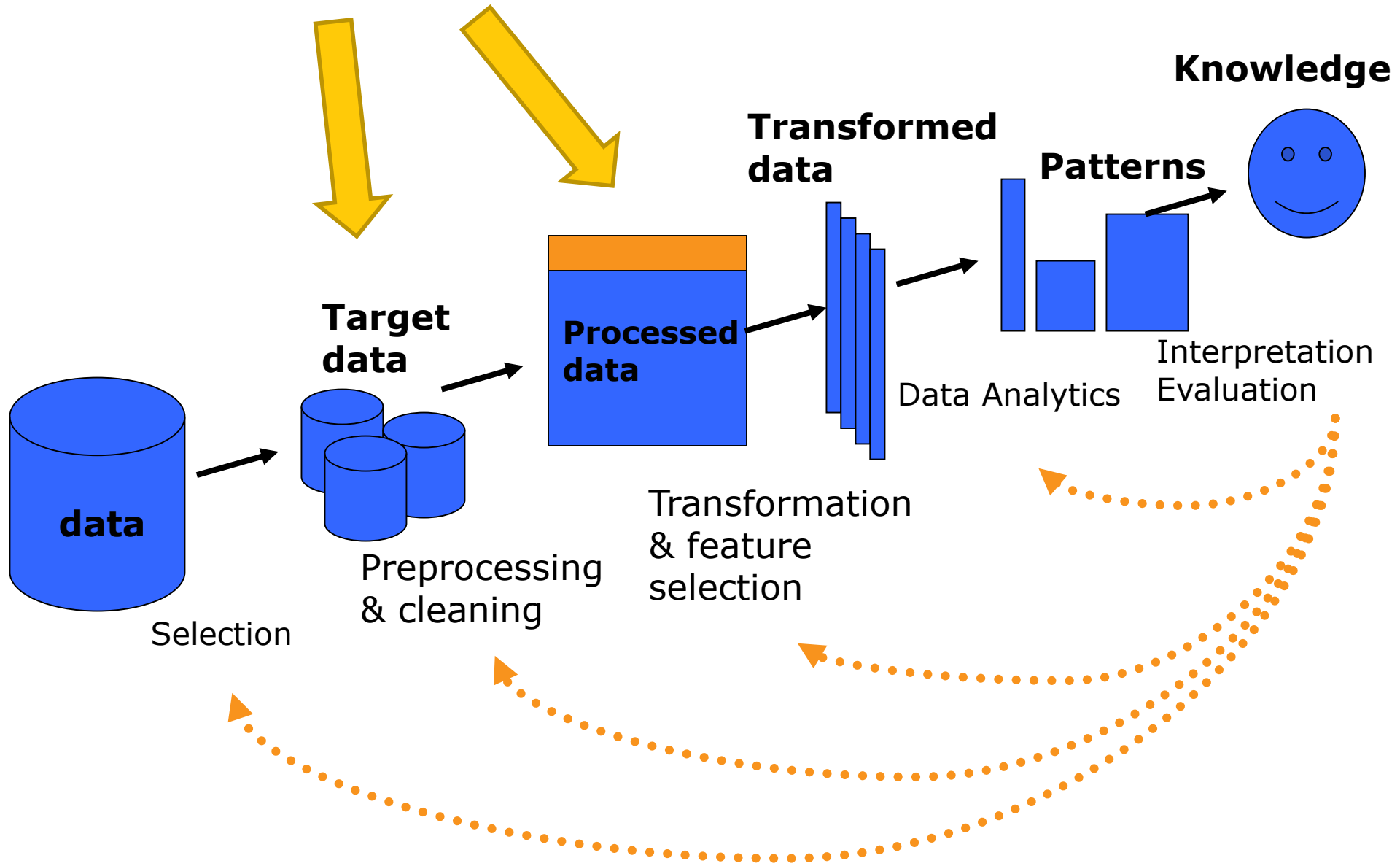
FEATURE SELECTION

盧信銘
Department of Information Management,
National Taiwan University

Outline

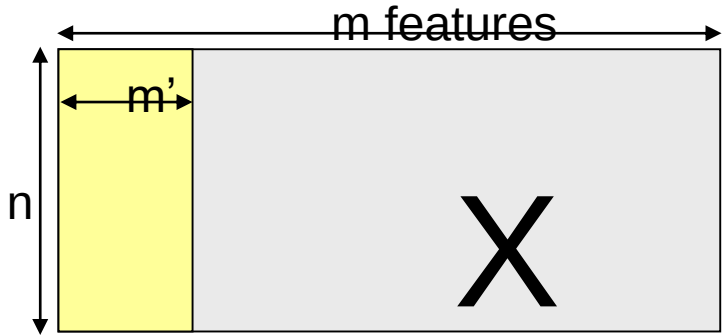
- Overview of Feature Selection
- Filtering approach
- Wrapper approach
- Embedded methods

Feature Selection



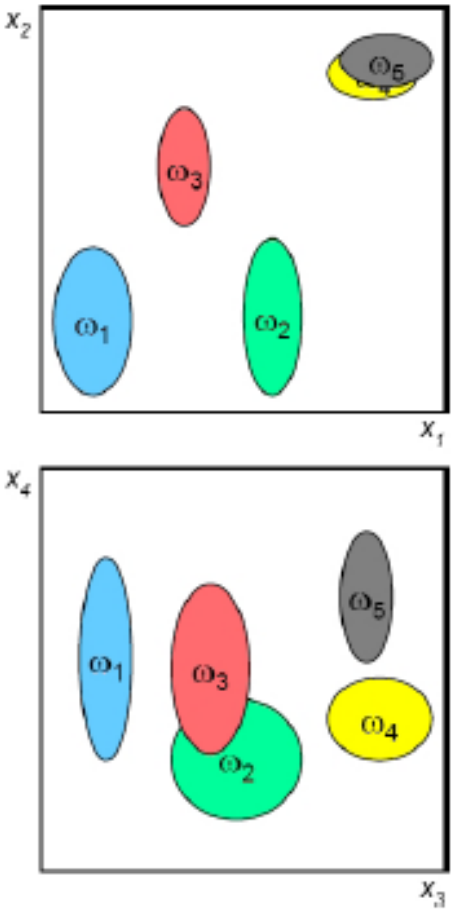
Feature Selection

- Goal: In the presence of **millions of features/attributes/inputs/variables**, select the most relevant ones.
- Why do we perform feature selection?
  - Make using a particular classifier feasible
    - Some classifiers can't deal with 100,000 of features
  - Reduce training time
    - Training time for some methods is quadratic or worse in the number of features (e.g., logistic regression)
  - Simpler model that may be easier to interpret



Feature Selection (Example)

- Consider a classification problem with four features (x_1, x_2, x_3, x_4)
- Five classes: $\omega_1, \omega_2, \omega_3, \omega_4, \omega_5$
- Goal: select the two best features individually
  - Any reasonable objective J will rank the features
  - J(x1) > J(x2) = J(x3) > J(x4)
  - Thus, features chosen [x1,x2] or [x1,x3].
  - However, x4 is the only feature that provides *complementary* information to x1



Do we Really need Feature Selection?

- Doing feature selection is a more “traditional approach” to machine learning.
- Effective → Improve model performance (speed, prediction performance) with relatively low investment in computational resource.
- However, most modern models have integrated feature selection into the learning process

Do we Really need Feature Selection? (Cont'd.)

- Meaning: just throw in everything, and the model will take care of selecting features for you.
- Still, you need to perform a minimal level of feature selection so that the model can be trained efficiently.
- **Conclusion**: in most scenarios, minimal feature selection + powerful models.

- You should not do it the other way around (running with big feature selection procedure + simple model) unless you know what you are doing.

Feature Selection

- Filtering approach: ranks features or feature subsets based on some criteria (that is not related to the classifier considered).
  - …using univariate methods: consider one variable at a time
  - …using multivariate methods: consider more than one variables at a time

- Wrapper approach: uses a model (e.g. a classifier) to assess (many) features or feature subsets.
- Embedding approach:

  uses a classifier to build a (single) model with a subset of features that are internally selected.
  [e.g., adopt L1 regularization]

Regression vs. Classification

- We can perform feature selection for regression and classification problems.
- The basic idea is the same.
- Can do filtering, wrapper, and embedded approaches.
- However, details are different.

Procedure for Filtering (Single Variable)

- Given a training set T1 and tuning set T2
- Given a filtering approach g(y,x)
  - g(y,x) takes (outcome y, predictor x) and returns a score
  - For a predictor x_i
    - compute $g_i = g(y, x_i)$ using T1
  - Sort g_i and select the K most informative predictors

- Train model M1 on T1 using selected K predictors
- Test M1 on T2
- Repeat using different K.
- Select the best K, named K*
- For production model, combine T1 and T2 to apply the filtering approach and select K* feature to train the model.

Filtering (Single Variable)

- For Classification Problem
  - Document frequency and feature variance
  - IG – information gain
  - MI – point-wise mutual information
  - CHI – Chi-squared statistic
  - mRMR – Minimum Redundancy Maximum Relevance
- For Regression Problem
  - Correlation Coefficient (or its t-value) [Equivalent to running simple regression using each variable only.]
- Similarity-based Methods
  - Laplician score
  - Fisher score

Document Frequency and Feature Variance

- Other things being equal, features with higher variance is preferred.
  - You do not want a feature that is the same across all data points.
- For continuous-valued features, this criterion is less useful because we often “standardize” continuous-valued features so that all features have unit variance and zero means.
- However, for dummy variables (binary-valued features), this is a useful criterion.
- Consider the standard “bag of words” representation.

Document Frequency and Feature Variance (Cont'd.)

- Represent a corpus (a set of document) as a matrix.
- Each row represent a document
- Each column represent the appearance of a word.
- Each cell is either 0 or 1.
- For the column i, the variance is $p_i(1-p_i)$, where p_i is the probability of having 1.
- For text data, p_i is usually small, and is less than 0.5.
- As a result, more frequent words has higher p_i ➔ Higher variance.

Document Frequency and Feature Variance (Cont'd.)

- Conclusion: Select words with higher frequency because frequent words have higher variance.

- Note: "Tradition" text mining approach typically remove "stop words" before doing frequency count.
- This approach works better for "classic text" such as news articles.
- You may not want to do stop word removal if the characteristic of text is not so "regular."

Mutual Information (a.k.a Information Gain)

- We have talked about mutual information:
  $H(x) = -\sum_x p(x) \log_2 p(x)$,
- In general $H(x) = E[-\log_2 p(x)]$
- Also the conditional entropy:
  $H(y|x) = -\int\int p(y,x) \ln p(y|x)\, dydx$
- For discrete random variables,
  $H(y|x) = -\sum_x p(x)\, p(y|x) \ln p(y|x)$
- Mutual Information (or Information Gain)
  $IG(y|x) = H(Y) - H(Y|x)$

Information Gain (for Feature Selection)

Definition:

$$G(t) = H(c) - H(c|t)$$

$$= Entropy\ of\ outcomes - Entropy\ of\ outcomes\ given\ t$$

If t is a binary feature:

$$G(t) = -\sum_{i=1}^{m} P_r(c_i) \log P_r(c_i)$$

$$+P_r(t) \sum_{i=1}^{m} P_r(c_i|t) \log P_r(c_i|t)$$

$$+P_r(\bar{t}) \sum_{i=1}^{m} P_r(c_i|\bar{t}) \log P_r(c_i|\bar{t})$$

More discussion will follow…

Information Gain: Entropy

- Suppose X takes *n* values, V_1, V_2,… V_n, and

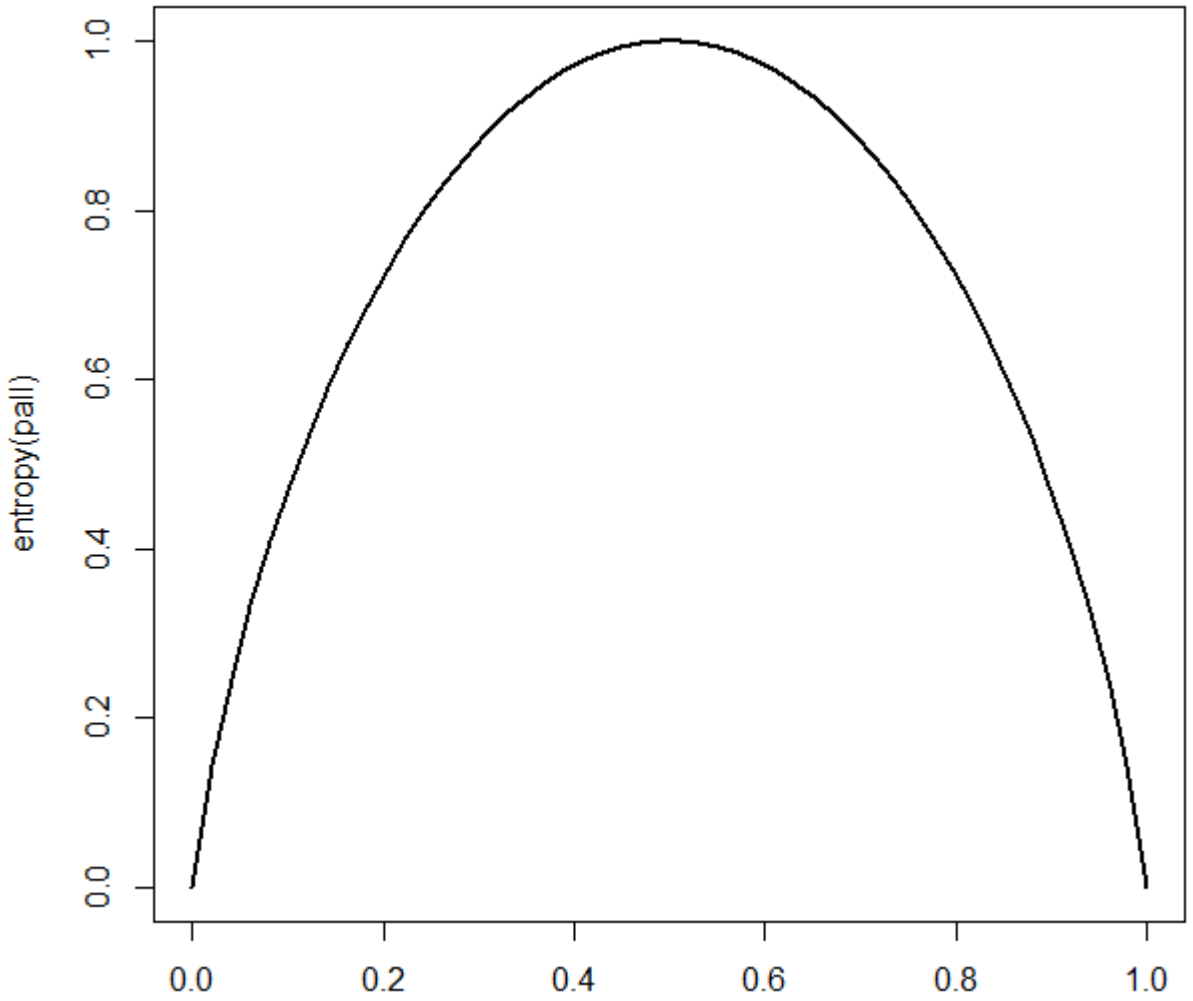
$$P(X=V_1)=p_1,\ P(X=V_2)=p_2,\ \ldots\ P(X=V_n)=p_n$$

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \ldots\ p_n \log_2 p_n$$

$$= -\sum_{i=1}^{n} p_i \log_2 (p_i)$$

- $H(X)$ = the entropy of X

Example: Two-class Entropy



- X-axis: probability of having positive outcome.
- Y-axis: entropy.
- Maximum of entropy is 1, with prob = 0.5

High, Low Entropy

- "High Entropy"
  - X is from a uniform like distribution
  - Flat histogram
  - Values sampled from it are less predictable

- "Low Entropy"
  - X is from a varied (peaks and valleys) distribution
  - Histogram has many lows and highs
  - Values sampled from it are more predictable

Specific Conditional Entropy, H(Y|X=v)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
IM	Yes
Math	No
Math	No
IM	Yes
History	No
Math	Yes

- I have input X and want to predict Y
- From data we estimate probabilities

  P(LikeG = Yes) = 0.5

  P(Major=Math & LikeG=No) = 0.25

  P(Major=Math) = 0.5

  P(Major=History & LikeG=Yes) = 0
- Note: You can compute the following

  H(X) = 1.5

  H(Y) = 1

Specific Conditional Entropy, H(Y|X=v)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
IM	Yes
Math	No
Math	No
IM	Yes
History	No
Math	Yes

- Definition of Specific Conditional Entropy
- H(Y|X=v) = **entropy of *Y* among only those records in which *X* has value *v***
- Example:

  H(Y|X=Math) = 1

  H(Y|X=History) = 0

  H(Y|X=IM) = 0

Conditional Entropy, H(Y|X)

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
IM	Yes
Math	No
Math	No
IM	Yes
History	No
Math	Yes

- Definition of Conditional Entropy

H(Y|X) = **the average conditional entropy of** *Y*

$$= \Sigma_i P(X=v_i)\, H(Y|X=v_i)$$

- Example:

v_i	$P(X=v_i)$	$H(Y|X=v_i)$
Math	0.5	1
History	0.25	0
IM	0.25	0

H(Y|X) = 0.5*1+0.25*0+0.25*0 = 0.5

Information Gain

X = College Major

Y = Likes "Gladiator"

X	Y
Math	Yes
History	No
IM	Yes
Math	No
Math	No
IM	Yes
History	No
Math	Yes

- Definition of Information Gain
- IG(Y|X) = **I must transmit** *Y.*

  How many bits on average would it save me if both ends of the line knew X?

$$IG(Y|X) = H(Y) - H(Y|X)$$

- Example:

  H(Y) = 1

  H(Y|X) = 0.5

  Thus:

  IG(Y|X) = 1 – 0.5 = 0.5

Pointwise Mutual Information (PMI)

- An information-theoretically motivated measure for discovering interesting co-occurrence is *pointwise mutual information* (Church et al. 1989, 1991; Hindle 1990).
- It is roughly a measure of how much one feature (g) tells us about the outcome (class c).
  - $I(g,c) = \log_2 \frac{P(g \cap c)}{P(g)P(c)}$
- Zero if *g* and *c* are independent.
  - $I(g,c) = \log_2 \frac{P(g \cap c)}{P(g)P(c)} = \log_2 \frac{P(g)P(c)}{P(g)P(c)} = 0$

Pointwise Mutual Information (Cont'd.)

- PMI can be interpreted as the improvement of probability for class *c* after we have known feature *g*.
- $I(g,c) = \log_2 \frac{P(g \cap c)}{P(g)P(c)} = \log_2 \frac{P(g|c)P(c)}{P(g)p(c)}$
- $= \log_2 \frac{p(g|c)}{p(g)} = -\log_2 \mathrm{P(g)} - \left(-\log_2 \mathrm{P(g|c)}\right)$

- Can be think of the log of "Lift" in association rule mining
- Example: *P*(*c*)=1/8, and *P*(*c*|g)=1,
  - I(g,c) = -(-3)-0 = 3

Pointwise Mutual Information (Cont'd.)

- PMI, in many cases, is not a good measure.
- For features with an similar conditional probability $P(g|c)$, **rare features usually have artificially high PMI**
- Undesired to assigned higher score to low frequency features.
  - We prefer to assign a higher score to frequent terms.
- Use frequency threshold to exclude rare features when applying PMI

Pointwise Mutual Information (Cont'd.)

- Applying PMI to the case of more than 2 outcome class needs some modification
- Average of individual class:

$$I_{avg}(g,c) = \sum_{i=1}^{M} P(c_i) I(g,c_i)$$

- Maximum of individual class:

$$I\_\max(g,c) = \max_{i=1}^{M} I(g,c_i)$$

- **Unless you are at a very special situation, we usually do not recommend use PMI to do feature selection.**

χ^2 (Chi-Squared) Feature Selection

- In statistics, the χ^2 test is applied **to test the independence of two random variables**.
- Independence of *A* and *B*:
  - $P(AB) = P(A)P(B)$, or $P(A|B) = P(A)$ and $P(B|A) = P(B)$.
- In feature selection, the two random variables are
  - Occurrence of the predictor: g=0/1, absent or present.
  - Occurrence of the outcome class c=0/1, not about *c* or about *c*.

observed frequency of (e_g & e_c) in *D*

expected frequency of (e_t & e_c) in *D* assuming that ***g* and *c* are independent**!!

$$\chi^2(g,c) = \sum_{g\in\{0,1\}} \sum_{c\in\{0,1\}} \frac{(N_{g,c} - E_{g,c})^2}{E_{g,c}}$$

Chi-Squared Feature Selection (Cont'd.)

- How to calculate the expected frequencies?

number of data points

		Feature g: *present*	Feature g: *absent*	
class *c*	*YES*	49	141	190
	NO	27,652	774,106	801758
		27701	774247	801948

$$E_{g=1,c=1} = N \times P(g = 1 \; and \; c = 1)$$
$$= \mathrm{N} \times P(g = 1)P(c = 1)$$

independence assumption

$$= 801948 * \frac{27701}{801948} * \frac{190}{801948} \approx 6.6$$

Chi-Squared Feature Selection (Cont'd.)

$$\chi^2(g,c) = \sum_{g\in\{0,1\}} \sum_{c\in\{0,1\}} \frac{(N_{g,c} - E_{g,c})^2}{E_{g,c}}$$

- χ^2 is a measure of how much expected counts *E* and observed counts *N* deviate from each other.
  - A high value of χ^2 indicates that the hypothesis of independence is incorrect.
- Meaning: the feature and the outcome are dependent
  - It may be the case that the absent of feature g is associated with the appearance of class c.

Chi-Squared Feature Selection (Cont'd.)

- Chi-squared can be directly extended to outcome class >=3

Outcome Class	Feature g Present	Feature g absent
Class 1	A	D
Class 2	B	E
Class 3	C	F

- A similar equation can be used to compute Chi-squared statistics
- Another approach is to convert to 3 binary classes (not recommended)
  - Apply binary Chi-squared statistics
  - Take average (weighted by probability) or maximum

Maximum-relevance-minimal-redundancy (mRMR)

- A general setting for feature selection is to find a “best” subset of features.
- The “best” subset can be defined by maximizing posterior probability when a classification or regression model is involved.
- What should be do if there is no “model” involved?

- Consider the case that we are given a set of selected features S.
- How do we select the next feature f_i to be include in S?

Hanchuan Peng, Fuhui Long and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226-1238, Aug. 2005.

Maximum-relevance-minimal-redundancy (mRMR)

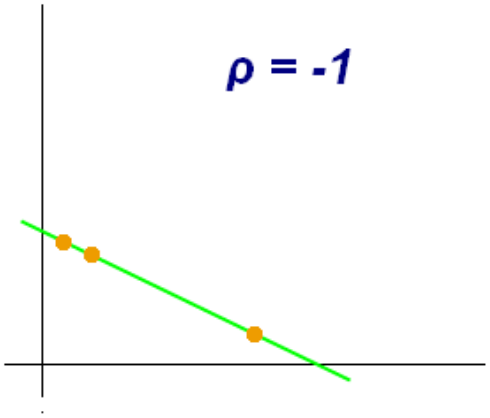
- Consider the case that we are given a set of selected features S.
- How do we select the next feature f_k to be include in S?
- We should try to
- Maximize its correlation with class label Y: $I(f_k;\ Y)$. $I(\cdot)$ is the mutual information between f_k and Y.
- Minimize its redundancy w.r.t. selected features in S: $\sum_{f_j \in S} I(f_j; f_k)$.
- $score(f_k) = I(f_k, Y) - \frac{1}{|S|}\sum_{f_j \in S} I(f_k; f_j)$
- ➔ Forward selection procedure.

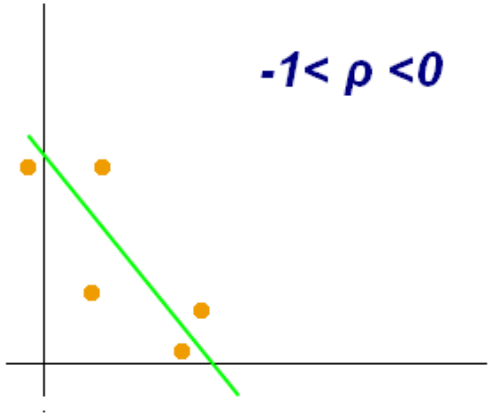
Correlation Coefficient

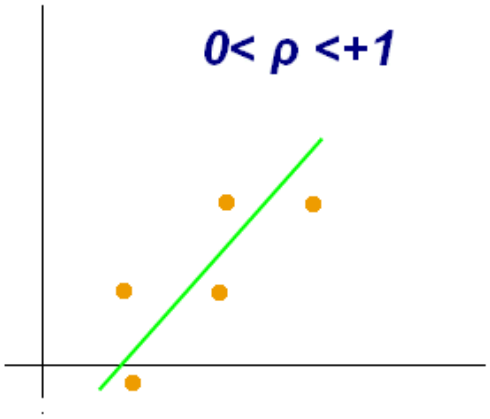
- Pearson product-moment correlation coefficient is a measure of the linear correlation (dependence) between two variables *X* and *Y*,
  - Giving a value between +1 and −1 (inclusive)
  - 1 is total positive correlation, 0 is no correlation, and −1 is total negative correlation.
- Given a dataset: $\mathrm{r} = r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})}}$
- t statistics:

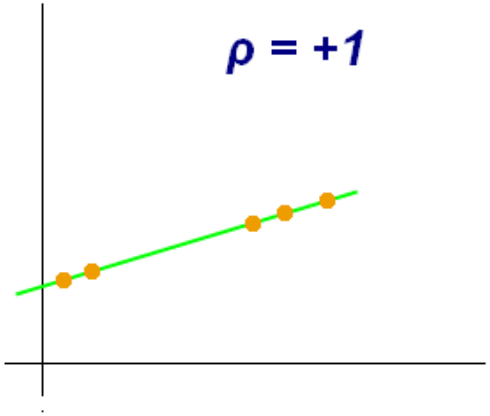
$$t = r\sqrt{\frac{n-2}{1-r^2}}$$

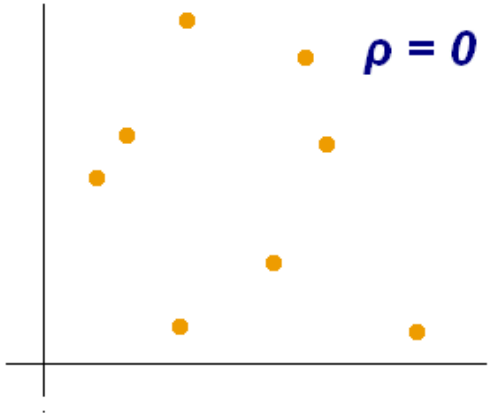
Correlation Coefficient (Cont'd.)



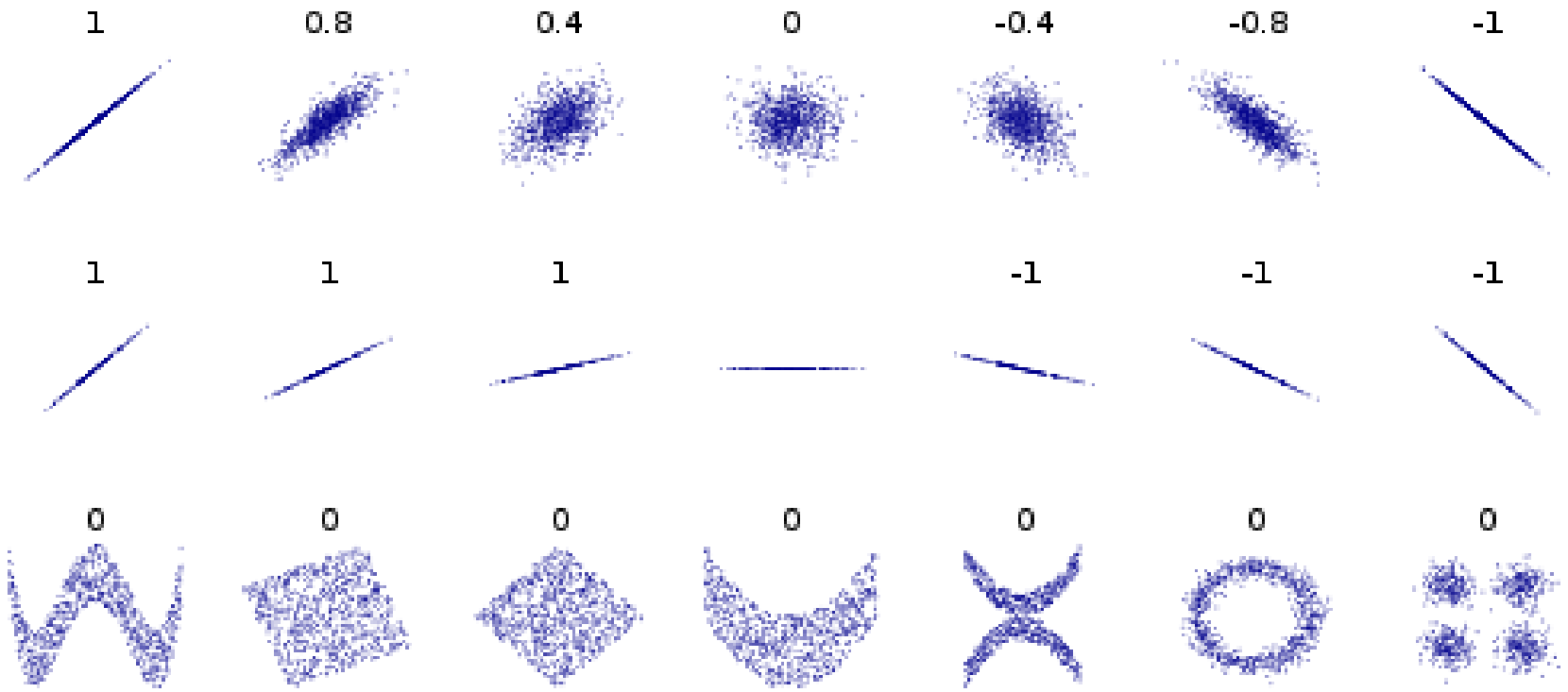








Correlation Coefficient (Cont'd.)



Feature Selection with Correlation Coefficient

- Compute r_{y,x_i} for each feature x_i
- Rank Feature by
  - Absolute value of r_{y,x_i}
  - Absolute t-value of r_{y,x_i}
- When rank by absolute t-value, correlation coefficient is equivalent to regressing y on x_i and rank by absolute t-value.

Similarity-Based Approach

- Pairwise data similarity often emerges naturally. For example, we can define the similarity of two pictures using a RBF kernel.

- $s_{ij} = \exp\left\{-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right\}$
- For class-labeled data, we can define similarity by

- $s_{ij} = \begin{cases} \frac{1}{n_l} & if\ y_i = y_j = l \\ 0 & otherwise \end{cases}$

Laplacian Score (Unsupervised Learning)

- Intuition: Locality preserving power.
- Data preparation.
- Construct a similar matrix S via the RBF kernel with preselected bandwidth.
- Set $S_{ij} = 0$ if i and j are not close. Two points are close if i is among the k nearest neighbors of j, or j is among the k nearest neighbors of i (k is often set to 10% of total data points).
- A good feature should (1) have similar values for similar data points (2) have large variance. Thus we minimize (r: feature index):
- $L_r = \frac{\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij}}{Var(\boldsymbol{f}_r)}$

Laplacian Score (Cont'd.)

- A good feature should (1) have similar values for similar data points (2) have large variance. Thus we minimize:
- $L_r = \frac{\sum_{ij}(f_{ri}-f_{rj})^2 S_{ij}}{Var(\boldsymbol{f}_r)}$
- $Var(\boldsymbol{f}_r)$ is the estimated variance of the r-th feature.
- By minimizing $\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij}$, we prefer those features respecting the pre-defined graph structure as defined in S.
- For a good feature, the bigger S_{ij}, the smaller $(f_{ri} - f_{rj})$.

Laplacian Score Details

- $\sum_{ij}(f_{ri} - f_{rj})^2 S_{ij} = \sum_{ij}(f_{ri}^2 + f_{rj}^2 - 2f_{ri}f_{rj})S_{ij}$
- $= 2\sum_{ij} f_{ri}^2 S_{ij} - 2\sum_{ij} f_{ri} S_{ij} f_{rj} = 2\boldsymbol{f}_r^T D\boldsymbol{f}_r - 2\boldsymbol{f}_r^T S\boldsymbol{f}_r$
- $= 2\boldsymbol{f}_r^T L\boldsymbol{f}_r,$
- where D is a diagonal matrix that has $D_{ii} = S_{ii}$, and
- $L = D - S$

Laplacian Score Details (Cont'd.)

- The other term is $Var(\boldsymbol{f}_r)$, which is defined to be the weighted data variance: $Var(\boldsymbol{f}_r) = \sum_i (f_{ri} - \mu_r)^2 D_{ii}$.
- In most cases, $D_{ii} = 1$, which means that all data points are equally important.
- $\mu_r = \sum_i (f_{ri} \frac{D_{ii}}{\sum_j D_{jj}}) = \frac{1}{\sum_j D_{jj}} \sum_i f_{ri} D_{ii} = \frac{\boldsymbol{f}_r^T D \mathbf{1}}{\mathbf{1}^T D \mathbf{1}}$
- Define the de-meaned feature vector as: $\tilde{f}_r = f_r - \frac{\boldsymbol{f}_r^T D \mathbf{1}}{\mathbf{1}^T D \mathbf{1}} \mathbf{1}$
- $Var(\boldsymbol{f}_r) = \sum_i \tilde{f}_{ri}^2 D_{ii} = \tilde{\boldsymbol{f}}_r^T D \tilde{\boldsymbol{f}}_r$
- Also, note that $\boldsymbol{f}_r^T L \boldsymbol{f}_r = \tilde{\boldsymbol{f}}_r^T L \tilde{\boldsymbol{f}}_r$

Laplacian Score Algorithm

- Set k and t to reasonable numbers.
- Compute the similarity matrix S using $S_{ij} = \exp\left\{-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{t}\right\}$ and truncate those that are not close (in the sense of k).
- For the r-th feature, define $f_r = [f_{r1}\ f_{r2}\ \ldots, f_{rn}]^T$,
- Compute $\tilde{f}_r = f_r - \frac{\boldsymbol{f}_r^T D\mathbf{1}}{\mathbf{1}^T D\mathbf{1}}\mathbf{1}$
- Compute $L_r = \frac{\tilde{\boldsymbol{f}}_r^T L\tilde{\boldsymbol{f}}_r}{\tilde{\boldsymbol{f}}_r^T D\tilde{\boldsymbol{f}}_r}$.
- Lower L_r are better.

Hints

- Document frequency is simply yet effective (for text classification problem).
- Information gain and Chi-squared methods are good choices.
- Do not use pointwise mutual information.
- You can typically remove more than 50% of features without strong effect on prediction performance.
- **Remove just enough features so that you can train your models efficiently.**
- If you need aggressive feature selection, consider mRMR.
- For unsupervised learning problem, consider Laplician score.

Wrapper Approach

- Uses a classifier to assess (many) features or feature subsets.
- Best subset selection: select the “best” subset from all combinations of features.
- Impractical to do subset selection for large p (# of features)
  - Why?
- Another issue: we are guilty if we conducted extensive search using the training data.
  - Good training performance does not necessarily lead to good testing performance
  - Susceptible to the overfitting issue.
- For these reasons, we often conduct “stepwise selection”
  - Forward selection
  - Backward selection

Validation Dataset

- When conducting feature selection using wrapper approach, you may or may not need a validation dataset.

- If you are using model selection criteria such as adjusted R2, AIC, BIC, or model evidence, you do not need to use a validation dataset.
- However, if you are simply selecting features based on prediction performance, then a validation dataset is needed.

Forward Stepwise Selection (Regression)

- Begins with a model containing no features (predictors)
- Add predictors to the model one-at-a-time
  - Select the variable that gives the greatest additional improvement to the fit
- Until all predictors are in the model

Forward Stepwise Selection (Regression)

1. Let M_0 denote the null model (no predictors)
2. For $k = 0, 1, 2, \ldots, p-1$:
   1. Consider all $p - k$ models that augment the predictors in M_k with one additional predictor.
   2. Choose the best among the $p - k$ models, and call it M_{k+1}. Here best is defined as having smallest residual sum of square (RSS) or highest R^2.
3. Select a single best model among $M_0, M_1, \ldots, M_p$ using cross validated prediction error or other model selection criteria (e.g. adjusted R^2, AIC, BIC)

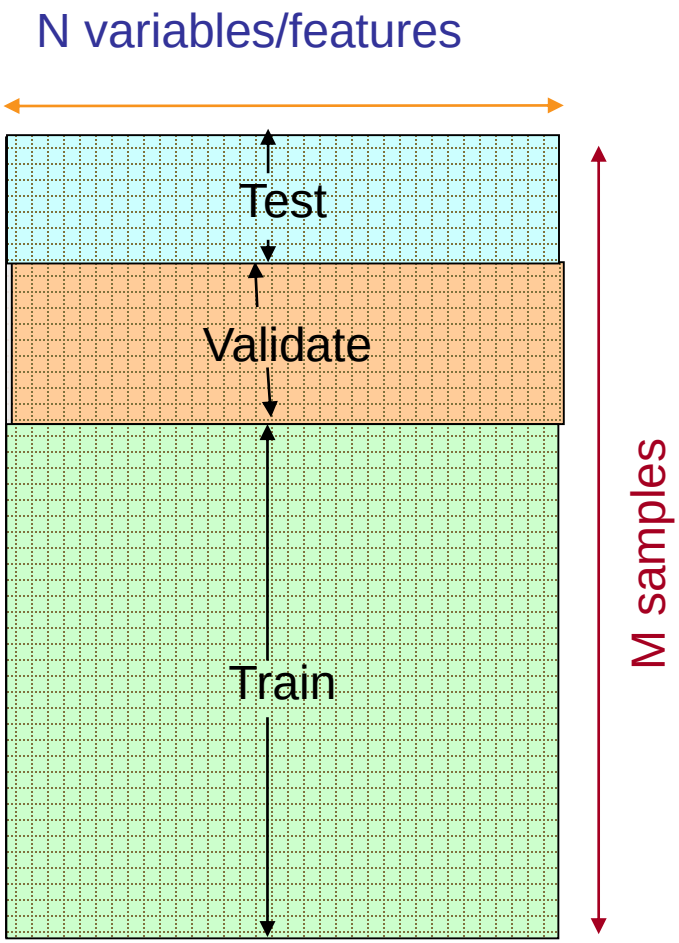
Backward Stepwise Selection (Regression)

- Start with the full model: a regression model that contains all predictors
- Iteratively removes the least useful predictor one-at-a time.
- Need to have large-enough sample (n>p) in order to estimate a full model.

Backward Stepwise Selection (Regression)

1. Let M_p denote the full model containing all p predictors
2. For $k = p, p-1, \ldots, 1$:
   1. Consider all k models that contain all but one of the predictors in M_k.
   2. Choose the best among these k models, and call it M_{k-1}. Here the best is defined as having smallest training residual sum of square (RSS) of highest R^2.
3. Select a single best model among $M_0, M_1, \ldots, M_p$ using cross validated prediction error or other model selection criteria (e.g. adjusted R^2, AIC, BIC)

Feature Selection: feature subset assessment (wrapper)



Split data into 3 sets:
training, validation, and test set.

1) For each feature subset, train predictor on training data.

2) Select the feature subset, which performs best on validation data.
   - Repeat and average if you want to reduce variance (cross-validation).

3) Test on test data.

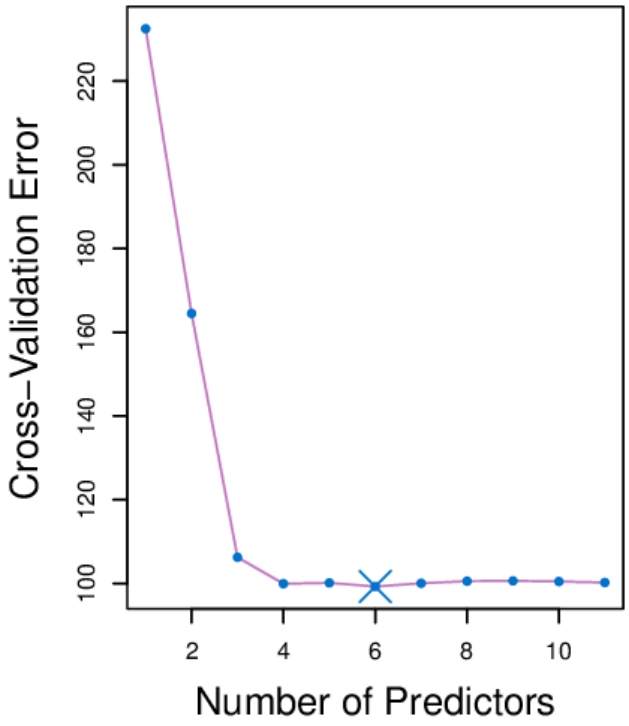
Danger of over-fitting with intensive search!

Choosing the Optimal Model

- Models with more features will always have smaller training error (RSS) and higher training R^2.
- We wish to choose a model with low test error, not a model with low training error.
- Need to perform stepwise selection within the train, validate, test framework.

Credit Data Example

- Two commonly used model selection rules.
- Rule 1 (Minimal error): Select the model with the lowest cross-validation error.
- Rule 2 (One Standard Error):
  (a) Compute the standard error for each model setting
  (b) select the smallest model for which the estimated test error is within one standard error of the lowest point on the curve. (why?)

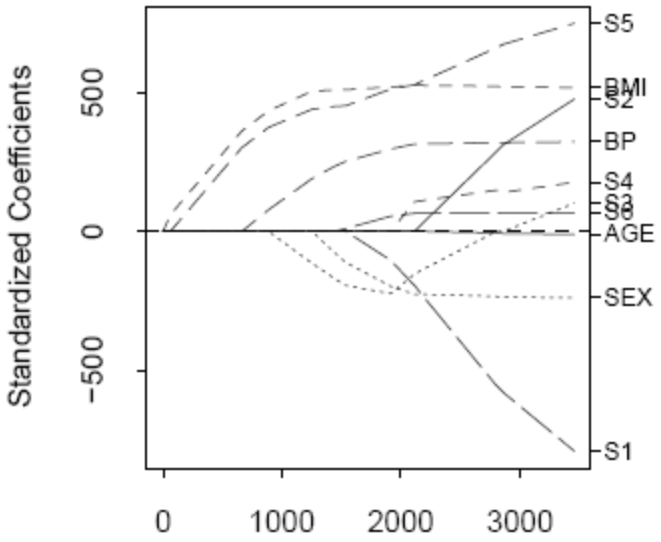


Embedded Methods: L1 and L2 Regularization

l_1 penalty: $y \sim Model(X\beta) + \lambda \sum |\beta_i|$ (lasso)

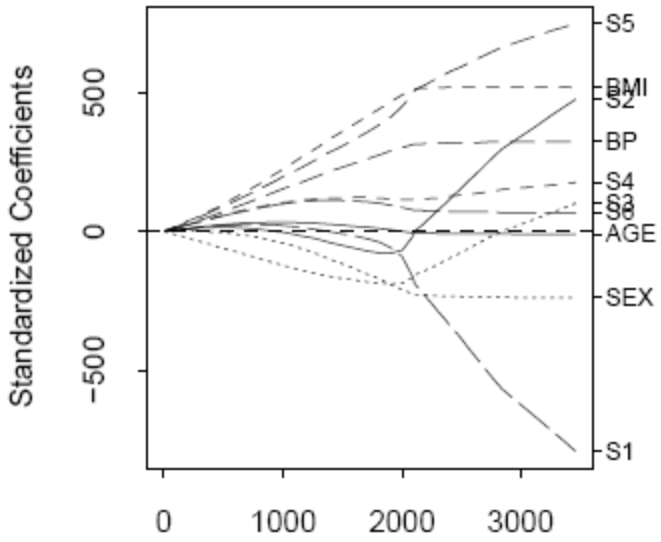
l_2 penalty: $y \sim Model(X\beta) + \lambda \sum \beta_i^2$ (ridge regression)

LASSO



sum(|beta|)

Ridge Regression



sum(|beta|)