

This cheatsheet is the summary of content taken from UCSD course:
CSE 258: Web Mining and Recommender System

General Modeling Knowledge

Learning Methods

Supervised Learning aims to directly model the relationship between input and output variables, so that the output variables can be predicted accurately given the input

Unsupervised Learning approaches find patterns, relationships and structure in data, but **are not** optimized to solve a particular predictive task. Most of the Unsupervised Machine Learning Methods are data reconstruction or data encoding algorithms.

Data Encodings

There are three types of features: **real-value feature**, **categorical feature** and **random feature**

Categorical feature is especially tricky to deal with because its possible values are discrete rather than continuous. We cannot simply fit a line across categorical data, in which case sequence doesn't matter.

- Ordinal Data (Temporal Data): Encode it as (1, 2, 3, .. n)
- Unordinal Data: Use **One-Hot Encoding** [0, 1, 0, ..., 0]

Mathematic Formula for One-Hot Encoding in Regression

$$c = \theta_0 + \theta_1[\text{is male}] + \theta_2[\text{is female}]$$

Model Selection

(**Train** | **Validation** | **Test**) split is important for determining various aspect of a predictive function.

- Training set: used to **optimize the model's parameters** : choose θ
- Test set: used to report how well we expect the model to perform on unseen data: only used **once**

- Validation set: used to **tune** any model parameters that are not directly optimized: choose λ

Theorems

- The training error **increases** as λ increases
- The validation and test error are at least as large as the training error (assuming infinitely large random partitions)
- The validation/test error will usually have a “sweet” spot between under- and over-fitting

Imbalanced Dataset

When implementing classifiers, sometimes there are far **fewer** positive examples than negative examples we may want to assign additional weight to negative instances.

Definition:

- True Positive (TP): label is true, prediction is true (correct predictions)
- True Negative (TN): label is false, prediction is false
- False Positive (FP): label is false, prediction is true (incorrect predictions)
- False Negative (FN): label is true, prediction is false

Reference to Binary Classification Performance Cheat Sheet *Figure 1*

Machine Learning Algorithms

Linear Regression

Formula

$$X\theta = y$$

- X is matrix of features (data)
- θ is unknowns (which features are relevant)
- y is the vector of outputs (labels)

Motivation

If X is a $m \times n$ matrix and $m \geq n$ (number of observations \geq number of predictive features), then X is an **inconsistent** matrix. Since X is inconsistent with y not exists in the subspace of X , we need to project y onto the span of θ .

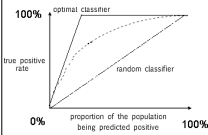
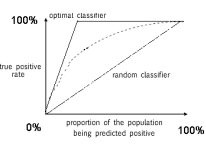
Binary classification performances measure cheat sheet Damien François - v1.1 - 2009 (damien.francois@uclouvain.be)																			
Confusion matrix for two possible outcomes p (positive) and n (negative) <div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: right; padding-right: 10px;">Actual</div> <table border="1" style="border-collapse: collapse; text-align: center;"> <tr> <td></td><td>p</td><td>n</td><td>Total</td></tr> <tr> <td>Predicted p'</td><td>true positive p</td><td>false positive p</td><td>p</td></tr> <tr> <td>Predicted n'</td><td>false negative n</td><td>true negative n</td><td>n</td></tr> <tr> <td>total</td><td>P'</td><td>N'</td><td>N</td></tr> </table> </div> Classification accuracy $(TP + TN) / (TP + TN + FP + FN)$ Error rate $(FP + FN) / (TP + TN + FP + FN)$			p	n	Total	Predicted p'	true positive p	false positive p	p	Predicted n'	false negative n	true negative n	n	total	P'	N'	N	True positive rate: proportion of actual positives which are predicted positive $TP / (TP + FN)$ True negative rate: proportion of actual negative which are predicted negative $TN / (TN + FP)$ Positive likelihood: likelihood that a predicted positive is an actual positive $sensitivity / (1 - specificity)$ Negative likelihood: likelihood that a predicted negative is an actual negative $(1 - sensitivity) / specificity$	
	p	n	Total																
Predicted p'	true positive p	false positive p	p																
Predicted n'	false negative n	true negative n	n																
total	P'	N'	N																
Paired criteria Precision: (or Positive predictive value) proportion of predicted positives which are actual positive $TP / (TP + FP)$ Recall: proportion of actual positives which are predicted positive $TP / (TP + FN)$ Sensitivity: proportion of actual positives which are predicted positive $TP / (TP + FN)$ Specificity: proportion of actual negative which are predicted negative $TN / (TN + FP)$		Combined criteria BCR: Balanced Classification Rate $\frac{1}{2} (TP / (TP + FN) + TN / (TN + FP))$ BER: Balanced Error Rate, or HTER: Half Total Error Rate: $1 - BCR$ F-measure harmonic mean between precision and recall $2 \cdot (precision \cdot recall) / (precision + recall)$ F_1-measure weighted harmonic mean between precision and recall $(1 + F^2) \cdot TP / ((1 + F^2) \cdot TP + F^2 \cdot FN + FP)$ The harmonic mean between specificity and sensitivity is also often used and sometimes referred to as F-measure.																	
		Youden's index: arithmetic mean between sensitivity and specificity $sensitivity - (1 - specificity)$ Matthews correlation correlation between the actual and predicted $((TP + FN)(TP + TN)(TP + FP)(TN + FN))^{1/2}$ comprised between -1 and 1 Discriminant power normalised likelihood index $\sqrt{3} / \sqrt{J}$ $(\log(sensitivity / (1 - specificity)) + \log(specificity / (1 - sensitivity)))$ < 1 = poor, > 3 = good, fair otherwise	(Cumulative) Lift chart plot of the true positive rate as a function of the proportion of the population being predicted positive, controlled by some classifier parameter (e.g. a threshold) 																
		Graphical tools ROC curve receiver operating characteristic curve : 2-D curve parametrized by one parameter of the classification algorithm, e.g. some threshold in the « true positive rate / false positive rate » space 	Relationships $sensitivity = recall = true\ positive\ rate$ $specificity = true\ negative\ rate$ $BCR = \frac{1}{2} \cdot (sensitivity + specificity)$ $BCR = 2 \cdot Youden's\ index - 1$ $F-measure = F_1-measure$ $Accuracy = 1 - error\ rate$ References Sokolova, M. and Lapalme, G. 2009. A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. 45, 4 (Jul. 2009), 427-437. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7 (2006) 1-30																

Figure 1: Link to Binary Classification Performances Measure Cheat Sheet

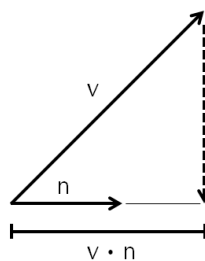


Figure 2: projection of v onto the dimension n

Note in *Figure 2*, $v - (v \cdot n)n$ is the **residual** vector which is **orthogonal** to vector n .

Optimization

Find θ such that $\|X\theta - y\|_2^2$ is minimized. The reason we use the residual norm of 2 is for convenient differentiation of the function.

$$\begin{aligned}\|X\theta - y\|_2^2 &= (X\theta - y)^T (X\theta - y) \\ &= \theta^T X^T X \theta - y^T X \theta - \theta^T X^T y + y^T y\end{aligned}$$

Differentiate the cost function:

$$\frac{\partial}{\partial \theta} f = 2X^T X \theta - y^T X - X^T y = 0$$

Because $y^T X$ is scalar, which implies $y^T X = (y^T X)^T = y X^T$

$$X^T X \theta = y^T X$$

$$\theta = (X^T X)^{-1} y^T X$$

MSE Mean-squared error (Numeric Explanation)

$$\frac{1}{N} \|y - X\theta\|_2^2 = \frac{1}{N} \sum_{i=1}^N (y_i - X_i \cdot \theta)^2$$

Motivation behind MSE is the assumption that label (truth) = prediction + error, where error is distributed with a normal distribution of $N(0, \sigma)$. This results in the prediction function as:

$$\begin{aligned}P_\theta(y|X, \theta) &= \prod_i \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y_i - x_i \cdot \theta)^2}{2\sigma^2}} \\ \max_\theta P_\theta(y|X, \theta) &= \max_\theta \prod_i e^{-(y_i - x_i \cdot \theta)^2} \\ &= \min_\theta \sum_i (y_i - x_i \cdot \theta)^2\end{aligned}$$

Coefficient of determination (R^2 statistic): Regression Diagnostics

FVU, fraction of variance unexplained. If $FVU(f) = 1$, f is **trivial** predictor. If $FVU(f) = 0$, f is **perfect** predictor.

$$FVU(f) = \frac{MSE(f)}{Var(y)}$$

If $R^2 = 0$, f is **trivial** predictor. If $R^2 = 1$, f is **perfect** predictor.

$$R^2 = 1 - FVU(f) = 1 - \frac{MSE(f)}{Var(y)}$$

Regularization

Regularization is the process of penalizing model complexity during training.

Naive Bayes

Naive Bayes is a classification algorithms that assumes that features are **conditionally independent** given the label.

$$(feature_i, feature_j | label) = (feature_i | label)(feature_j | label)$$

Using Bayes Theorem:

$$p(label | features) = \frac{p(label) \prod_i p(feature_i | label)}{p(features)}$$

Assume label is binary result, and the probability of **not label** is represented as $p(-label | features)$

$$\frac{p(label | features)}{p(-label | features)} = \frac{p(label) \prod_i p(features_i | label)}{p(-label) \prod_i p(features_i | -label)}$$

Advantages

- Easiest to implement, most efficient to “train”
- If we have a process that generates feature that are independent given the label, it's a very sensible idea

Disadvantage of Naive Bayes

Because both features encode essentially the same information when calculating $p(\text{label}|\text{features})$ and $p(\text{-label}|\text{features})$, we will end up **double-counting** their effect.

Logistic Regression

Logistic regression is essentially a linear regression being mapped into a $[0, 1]$ dimension space using a **Sigmoid** function. The classifier form is:

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta > 0 \\ 0 & \text{otherwise} \end{cases}$$

Motivation

With y_i is either 0 or 1, we need to have a linear regression that separate the $P(Y = 0)$ as far away as possible from $P(Y = 1)$, which is to maximize the total probability of $P(Y = 0|X, \theta)$ in addition to the total probability of $P(Y = 1|X, \theta)$. Note there is **no double counting** as it occurs in Naive Bayes.

$$\text{argmax}_{\theta} = \prod_{y_i=1} P_{\theta}(y|X) \prod_{y_i=0} (1 - P_{\theta}(y|X)) = \prod_{y_i=1} \sigma(X_i \theta) \prod_{y_i=0} (1 - \sigma(X_i \theta))$$

Sigmoid Function

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

Optimization

- Take logarithm
- Subtract regularizer
- Compute gradient
- Solve using gradient ascent

$$L_{\theta}(y|X) = \prod_{y_i=1} p_{\theta}(y_i|X_i) \prod_{y_i=0} (1 - p_{\theta}(y_i|X_i))$$

Take the logarithm of the original Loss function for computational convenience:

$$\begin{aligned}
l_\theta(y|X) &= \sum_{y_i=1} \log \sigma(X_i \theta) + \sum_{y_i=0} \log(1 - \sigma(X_i \theta)) - \lambda \|\theta\|_2^2 \\
&= \sum_{y_i=1} \log\left(\frac{1}{1 + e^{-X_i \theta}}\right) + \sum_{y_i=0} \log\left(\frac{e^{-X_i \theta}}{1 + e^{-X_i \theta}}\right) - \lambda \|\theta\|_2^2 \\
&= \sum_{y_i=1} -\log(1 + e^{-X_i \theta}) + \sum_{y_i=0} -\log(1 + e^{-X_i \theta}) + \sum_{y_i=0} -X_i \theta - \lambda \|\theta\|_2^2 \\
&= \sum_{y_i} -\log(1 + e^{-X_i \theta}) + \sum_{y_i=0} -X_i \theta - \lambda \|\theta\|_2^2
\end{aligned}$$

Taking the Differentiation of the loss function:

$$\begin{aligned}
\frac{\partial l}{\partial \theta_k} &= \sum_i \frac{x_{ik} e^{-X_i \theta}}{1 + e^{-X_i \theta}} + \sum_{y_i=0} -x_{ik} - 2\lambda \theta_k \\
&= \sum_i x_{ik} (1 - \sigma(X_i \theta)) - \sum_{y_i=0} x_{ik} - 2\lambda \theta_k
\end{aligned}$$

Perform Gradient Descent:

$$\theta_k := \theta_k - \alpha \frac{d}{d\theta_k} l(\theta)$$

Generalization

We can generalize logistic regression for **binary** classification into **multiclass** classification by training a binary predictor for each class. In the event that **multiclass** predictions are inconsistent, choose the one with the highest confidence.

Advantages

Fixes the “double counting” problem present in naive Bayes

Disadvantages

- Logistic regressors **don't** optimize the **number of “mistakes”**. It is simply trying to separate the difference between the general label groups $Y = 0$ and $Y = 1$.
- No special attention is paid to the “difficult” instances (which can be inconsistent in classifying edge cases laying on top of the boundary) - every instance influence the model

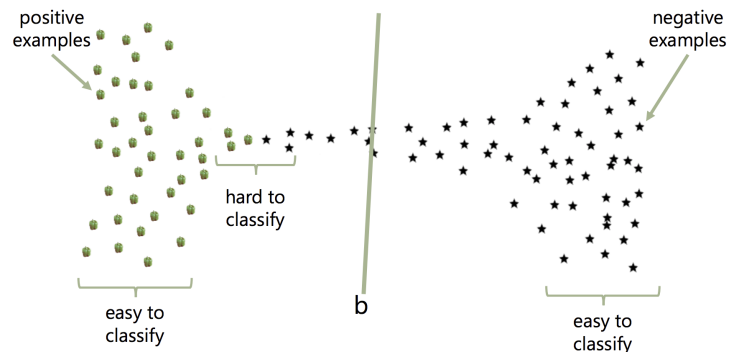


Figure 3: logistic regression for classification

- “easy” instances that has high confidence can affect the model in a bad way
- The model is more expensive to train compared to naive bayes

Support Vector Machine

Motivation

The intuition behind Support Vector Machines (SVMs) is to train a classifier that focuses on the **difficult** examples by minimizing the **misclassification error**. Our classifier form is:

$$y_i = \begin{cases} 1 & \text{if } X_i \cdot \theta - \alpha > 0 \\ -1 & \text{otherwise} \end{cases}$$

We want to minimize the number of misclassifications:

$$\operatorname{argmin}_{\theta} \sum_i \delta(y_i(X_i \cdot \theta - \alpha) \geq 0)$$

Support Vector

Because $\theta x - \alpha = 0$ is a hyperplane and θ is underdetermined, there are many solutions to θ . In this case, we need regularization to enforce the best solution of θ . Here we will choose the classifier that maximizes the distance to the nearest point.

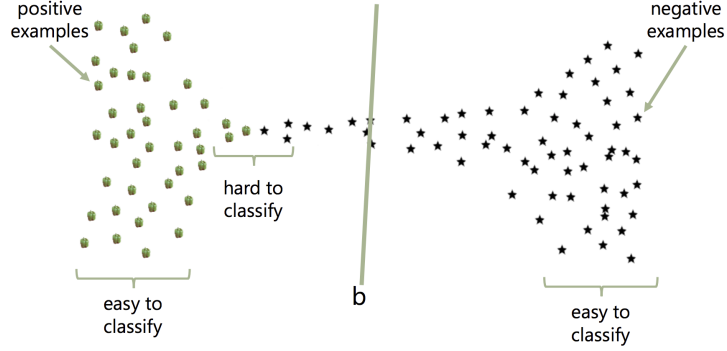


Figure 4: support vector machines use support vector to determin theta

Distance from Point to Line

Suppose we are trying to calculate the distance between a line to a point.

$$\text{line : } ax + by + c = 0$$

$$\text{point : } x_0, y_0$$

Then the distance from the line to the point is:

$$d(\text{line, point}) = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}}$$

Apply the same principle to the classifier, for all x_i , the distance between x_i to $\theta X - \alpha = 0$ should be greater or equal to the distance from support vector $\theta X - \alpha = 1$ or $\theta X - \alpha = -1$ to $\theta X - \alpha = 0$

$$\frac{(\theta X_i - \alpha)y_i}{\|\theta\|_2} \geq \frac{1}{\|\theta\|_2}$$

Optimization

Because the distance between the center regression line and the two support vectors is $\frac{1}{\|\theta\|}$ and we want to maximize the distance $\frac{2}{\|\theta\|}$, for mathematic convenience, we can solve the same problem by minimize $\frac{1}{2}\|\theta\|_2^2$.

$$\operatorname{argmin}_{(\theta, \alpha)} \frac{1}{2}\|\theta\|_2^2 \quad \text{s.t. } \forall_i y_i(\theta \cdot X_i - \alpha) \geq 1$$

This is known as a **quadratic program** (QP) and can be solved using **standard** techniques.

Soft Margin

If the data is not completely separable or there is a better classification by sacrificing several misclassifications, we can use **soft margin SVM** to treat this issue. By introducing ξ , we are able to penalize points that end up on the wrong side of the support vectors.

$$\operatorname{argmin}_{\theta, \alpha, \xi_i > 0} \frac{1}{2} \|\theta\|_2^2 + \sum_i \xi_i \quad \text{s.t. } \forall_i y_i (\theta \cdot X_i - \alpha) \geq 1 - \xi_i$$

Advantages

Non-probabilistic: optimizes the classification error rather than the likelihood

Disadvantages

More expensive to train compared to logistic regression and naive bayes

Principle Component Analysis

Motivation

- Denoise the data that is dense in certine dimensions and sparse in other dimension. (Dimensionality Reduction)

Assume $X \in R^{M \times N}$, $\varphi \in R^{M \times M}$, and φ is basis function such that when X is rotated ($Y = \varphi X$)

- $\varphi_i \cdot \varphi_i = 1$, $\varphi_i^{-1} = \varphi_i^T$
- $\varphi_i \cdot \varphi_j = 0$ if $i \neq j$
- Dimension with highest variance is y_0
- Dimension with 2^{nd} highest variance is y_1 , etc.

For a single data point: $y = \varphi x$ $x = \varphi^{-1} y = \varphi^T y$

$$\begin{aligned} x &= \varphi_1 y_1 + \varphi_2 y_2 + \cdots + \varphi_M y_M \\ &= \sum_{j=1}^M \varphi_j y_j \end{aligned}$$

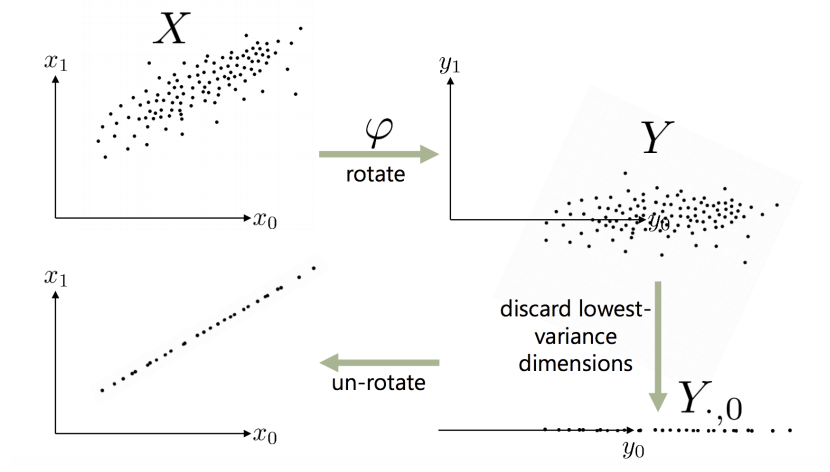


Figure 5: rotate data points across different dimensional space

Discarding the additional dimension ($K+1$ to N) by setting y_j where $K+1 \leq j < N$ to a constant b_j (**Data Reconstruction**)

$$x \simeq \sum_{j=1}^K \varphi_j y_j + \sum_{j=K+1}^M \varphi_j b_j$$

Optimization

minimize **MSE** to get the “best” overall reconstruction.

$$\begin{aligned} \min_{\varphi, b} \frac{1}{N} \sum_y \left\| \sum_{j=1}^K \varphi_j y_j + \sum_{j=K+1}^M \varphi_j b_j - \varphi^T y \right\|_2^2 &= \frac{1}{N} \sum_y \left\| \sum_{j=K+1}^M \varphi_j (b_j - y_j) \right\|_2^2 \\ &= \frac{1}{N} \sum_y \sum_{i=K+1}^M \sum_{j=K+1}^M (y_i - b_i)^T \varphi_i^T \varphi_j (y_j - b_j) \\ &= \frac{1}{N} \sum_y \sum_{j=K+1}^M (y_j - b_j)^2 \end{aligned}$$

Set $b_j = \bar{y}_j$ as the average of j^{th} dimension:

$$\min_{\varphi} \frac{1}{N} \sum_y \sum_{j=K+1}^M (y_j - \bar{y}_j)^2$$

(subject to φ **orthonormal**)

Expand in terms of X

$$\min \frac{1}{N} \sum_{j=K+1}^M \varphi_j (X - \bar{X})(X - \bar{X})^T \varphi_j^T$$

Regularization

We want to minimize **MSE** with subject to the constrain ($\varphi_i \cdot \varphi_i = 1 \ \forall i, i \geq N$)

Apply Lagrange Multipliers

$$\min \frac{1}{N} \sum_{j=K+1}^M \varphi_j \text{Cov}(X) \varphi_j^T - \lambda(\varphi_j \cdot \varphi_j - 1)$$

Solve

$$\frac{\partial}{\partial \varphi_j} \sum_{j=K+1}^M \varphi_j \text{Cov}(X) \varphi_j^T - \lambda(\varphi_j \cdot \varphi_j - 1) = 0$$

$$\text{Cov}(X) \varphi_j^T = \lambda_j \varphi_j^T$$

$$A \varphi^T = \lambda \varphi^T$$

- This expression can only be satisfied if φ_j and λ_j are an **eigenvectors/eigenvalues** of the covariance matrix
 - So to minimize the original expression we'd discard φ_j 's corresponding to the smallest eigenvalues
-