This cheatsheet is the summary of content taken from UCSD courses website:

- CSE 190: Discrete Continuous Optimization
- CSE 258: Web Mining and Recommender System
- CSE 252-A: Probabilistic Reasoning and Decision-Making

Data Preparation

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

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

Linear Algebra Foundamentals

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)

Movitation

If X is a mxn 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.

Note in *Figure 1*, $v - (v \cdot n)n$ is the **residual** vector which is **orthogonal** to vector n.

Optimization

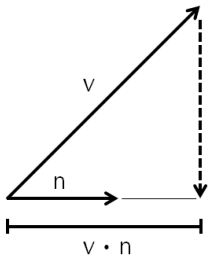

Figure 1: projection of v onto the dimension n

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 $y^T X = (y^T X)^T = yX^T$

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

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

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:

$$P_\theta(y|X, \theta) = \prod_i \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(y_i - x_i \cdot \theta)^2}{2\sigma^2}}$$

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

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.

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

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

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

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.

$$(\text{feature}_i, \text{feature}_j|\text{label}) = (\text{feature}_i|\text{label})(\text{feature}_j|\text{label})$$

Using Bayes Theorem:

$$p(\text{label}|\text{features}) = \frac{p(\text{label}) \prod_i p(\text{feature}_i|\text{label})}{p(\text{features})}$$

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

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

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(label|features) and p(-label|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.

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

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:

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

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
- "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

---

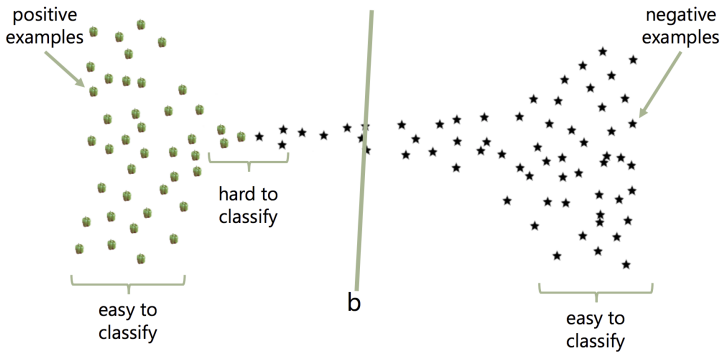

Figure 2: logistic regression for classification

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:

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

Support Vector

Because $\theta x - \alpha = 0$ is a hyperplane and θ is underdeterminated, 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.

Distance from Point to Line Suppose we are trying to calculate the distance between a line to a point.

$$\begin{aligned} \text{line} &: ax + by + c = 0 \\ \text{point} &: x_0, y_0 \end{aligned}$$

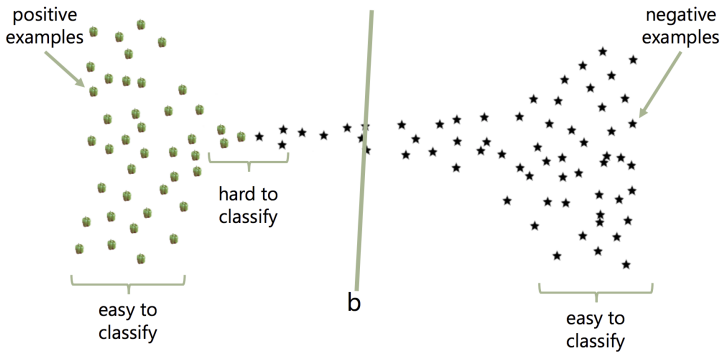

Figure 3: support vector machines use support vector to determin theta

Then the distance from the line to the point is:

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

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 $\frac{1}{2}||\theta||_2^2$.

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

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.

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

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

Disadvantages

More expensive to train compared to logistic regression and naive bayes

---

Evaluation of Classification

imbalanced data precision/recall

Principle Component Analysis
