Recommender Sys & Web Mining

Xiaojian Jin
xij049@ucsd.edu

February 12, 2017

Contents

Chapter 1

Regression Diagnostics

1.1 Coefficient of determination

1.1.1 R^2 statistic

$$FVU(f) = \frac{MSE(f)}{Var(y)} \quad \leftarrow \quad \text{fraction of variance unexplained}$$

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

$$R^2 = 0 \quad \rightarrow \quad \text{Trivial predictor}$$

$$R^2 = 1 \quad \rightarrow \quad \text{Perfect predictor}$$

1.1.2 Overfitting

When a model performs well on training data but doesn't generalize, we are said to be overfitting.

$$X\theta = y, \quad \theta \leftarrow \text{hypothesis}$$

A "simple" model is one where θ

- has few non-zero parameters
- is almost uniform

1.1.3 Regulariztion

Regularization is the process of penalizing model complexity during training

$$\arg\min \theta = \frac{1}{N}\|y - X\theta\|_2^2 + \lambda\|\theta\|_2^2$$

Optimizing the (regularized) model

$$\frac{\partial f}{\partial \theta_k} = \frac{1}{N}\sum_i -2x_{ik}(y_i - x_i\theta) + 2\lambda\theta$$

1.1.4 Model Selection

How to select which model is best? We need a **third** dataset.
A **validation set** is constructed to "tune" the model's parameters that are not directly optimized.
Some

- The training error increases as λ increases
- The validation and test error are at least as large as the training error
- The validation/test error will usually have a "sweet spot" between under- and over-fitting

1.2 Summary

1. Regression can be cast in terms of maximizing a likelihood
2. Gradient descent for model optimization
   - Initialize θ at random
   - While (not converge) do $\theta := \theta - \alpha f'(x)$
3. Regularization is the process of penalizing model complexity during training
4. Regularization Pipeline
   - Training
   - Test
   - Validation

Chapter 2

Supervised learning - Classification

2.1 Naive Bayes

Naive Bayes assumes that features are conditionally independent given the label

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

The denominator doesn't matter, because we really just care about

$$P(label|features) \quad \textbf{V.S.} \quad P(not \quad label|features)$$

2.2 Logistic Regression

sigmoid function: $\sigma(t) = \frac{1}{1+e^{-t}}$. **Training**:

$$\begin{aligned}
L_\theta(y|X) &= \prod_{y_i=1} P_\theta(y_i|X_i) \prod_{y_i=0} (1 - P_\theta(y_i|X_i)) \\
\log L_\theta(y|X) &= \sum_i -\log(1 + e^{-X_i\cdot\theta}) + \sum_{y_i=0} -X_i \cdot \theta - \lambda\|\theta\|_2^2 \\
\frac{\partial \log L_\theta(y|X)}{\partial \theta_k} &= \sum_i \frac{x_{ik}e^{-X_i\cdot\theta}}{1 + e^{-X_i\theta}} + \sum_{y_i=0} -x_{ik} - 2\lambda\theta_k \\
&= \sum_i x_{ik}\left[1 - \sigma(X_i \cdot \theta)\right] + + \sum_{y_i=0} -x_{ik} - 2\lambda\theta_k
\end{aligned}$$

2.2.1 Multiclass classification

The most common way to generalize binary classification (output in {0,1}) to multiclass classification (output in {1...N}) is simply to a binary predictor for each class.

2.3 Support Vector Machines

Distance from a point to a line

$$\begin{aligned} ax + by + c &= 0 \\ d(line, p_0) &= \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \\ \theta x - \alpha = 0 \Rightarrow & \frac{|\theta x - \alpha|}{\|\theta\|_2} \\ \frac{|\theta x - \alpha|}{\|\theta\|_2} y \quad &\geq \quad 1 \end{aligned}$$

Want the margin $\frac{1}{\|\theta\|}$ to be as wide as possible, while penalizing points on the wrong side. Soft-margin formulation:

$$\arg \min_{\theta, \alpha, \epsilon_i > 0} \frac{1}{2} \|\theta\|_2^2 + \epsilon$$

Try to optimize the **misclassfication error** rather than maximize a probability.

2.4 Evaluate Classifiers

classification accuracy = correct prediction / #predictions = (TP+TN) /(TP+TN+FP+FN)
Error rate = incorrect prediction / #predictions =(FP+FN) /(TP+TN+FP+FN)
True positive rate (TPR) = TP / (TP + FN)
True negative rate(TNR) = TN / (TN + FP)
Balanced Error Rate (BER) = 1/2 (FPR +FNR) = 1 - 0.5*(TPR + TNR)
=1/2 FOR A RANDOM/NAIVE classifier, 0 for a perfect classifier.
How to optimize a balanced error measure:

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

Then,

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

2.4.1 ranking

The classifiers we've seen can associate scores with each prediction

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

(harmonic mean of precision and recall)

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

(weighted in case precision is more important (low beta), or recall is more importrant (high beta))

2.5 Case study

2.5.1 Using Regression to Predict Content Popularity of Reddit

Chapter 3

Dimensionality Reduction

A1: A (sparse) vector including all movies Raw data issues:

1. Incredibly high-dimensional
2. Missing values
3. Not stable/robust to changing data
4. Redundant dimensions

A2: Describe my preferences using a **low-dimensional** vector. Represent each node/user in terms of the communities they belong to
Goal: take **high-dimensional** data, and describe it compactly using a small number of dimensions.
Assumption: Data lies (approximately) on some **low-dimensional** space.

Why dimensionality reduction? Unsupervised learning

To understand the important features of a dataset and the process which generated the data itself. The models we learn will prove useful when it comes to solving predictive tasks later.

3.1 Principal Component Analysis

- To select a few important features
- To compress the data by ignoring components which aren't meaningful.

For a single data point: $y = \underbrace{\varphi x}_{\text{compress}}, x = \underbrace{\varphi^{-1} y = \varphi^T y}_{\text{decompress}}$

$$x = \varphi_1 y_1 + \varphi_y \psi_2 + \ldots + \varphi_M y_M = \underbrace{\sum_{j=1}^{M} \varphi_j y_j}_{\text{"complete" reconstruction}}$$

$$x_i \simeq \underbrace{\sum_{j=1}^{k} \varphi_j y_j + \sum_{j=k+1}^{M} \varphi_j b_j}_{\text{approximate reconstruction}}$$

It should minimize the **MSE**:

$$\begin{aligned}\min_{\varphi,b}\frac{1}{N}\left\|\sum_{j=1}^{k}\varphi_j y_j+\sum_{j=k+1}^{M}\varphi_j b_j-\varphi^T y\right\|_2^2 \\ =\frac{1}{N}\sum_{y}\left\|\sum_{j+1}^{M}\varphi_j(y_j-b_j)\right\|_2^2 \\ =\sum_{y}\sum_{j=k+1}^{M}(y_i-b_i)^2 \\ \text{where } b_j=\bar{y_j}\end{aligned}$$

MSE is equal to the variance in the discarded dimensions.
Expand in terms of X

$$\begin{aligned}\sum_{j=k+1}^{M}\sum_{i}\left[\varphi_j(X-\bar{X})\right]^2&=\frac{1}{N}\sum_{j=k+1}^{M}\varphi_j(X-\bar{X})^T(X-\bar{X})\varphi_j^T \\ &=\frac{1}{N}\sum_{j=k+1}^{M}\varphi_j\text{cov}(X)\varphi_j^T\end{aligned}$$

Solve:

$$\frac{\partial}{\partial\varphi_j}\frac{1}{N}\sum_{j=k+1}^{M}\varphi_j\text{cov}(X)\varphi_j^T-\lambda_j(\varphi_j\varphi_j^T-1)=0$$

This expression can only be satisfied if φ_j and λ_j are an eigenvectors/eigenvalues of the covariance matrix.

3.2 K-means Clustering

1. Input is still a matrix of features
2. Output is a list of cluster "centroids"
3. From this we can describe each point in X by its cluster membership

Given feature(X) our goal is to choose k centroids (C) and cluster assignments (Y) so that the reconstruction error is minimized.

$$\text{reconstrunction error}=\sum_i\|X_i-C_{y_i}\|_2^2$$

This is an NP-Hard optimization problem. Use Greedy algorithm:

```
Initializing C (e.g. at random)
Do
        Assign each X_i to its nearest centroid
        Update each centroid to be the mean of points assigned to it
While (assignments change between iterations)
```

3.2.1 Hierarchical clustering

Hierarchical (agglomerative) clustering works by gradually fusing clusters whose points are closet together.

```
Assign every point to its own cluster:
Clusters = [[1],[2],[3],...,[N]]
while len(Clusters) > 1:
        Compute the center of each cluster
        Combine the two clusters with the nearest centers
```

If we keep track of the order in which clusters were merged, we can build a "hierarchical" of clusters. Splitting the dendrogram at different points defines cluster "levels" from which we can build our feature representation

3.2.2 How to choose K in K-means?

Keep adding dimensions until add more no longer decreases the reconstruction error significantly. Group sets of points based on their connectivity

3.3 Community Detection

Points are defined by their relationships to each other.
Question: How can we compactly represent the set of relationships in a graph? **Answer:** By representing the nodes in terms of the communities they belong to. How should a "community" be defined?

- Members should be connected
- Few edges between communities
- Cliqueishenses
- Dense inside, few edges outside

3.3.1 Connected Components

Define communities in terms of sets of nodes which are reachable from each other.
If a and b belong to be strongly connected component then there must be path from $a \to b$ and a path from $b \to a$
A weekly connected component is a set of nodes that would be strongly connected, if the graph were undirected.

3.3.2 Graph Cuts

What if the separation between communities isn't so clear?
Cut the network into two partitions such that the number of edges crossed by the cut is minimal. Solution will be degenerate - we need additional constraints.

$$\text{Ratio Cut}(C) = \frac{1}{|C|} \sum_{c \in C} \frac{\text{cut}(c, \bar{c})}{|c|}$$

where C is Proposed set of communities,
$\text{cut}(c, \bar{c})$ is # of edges that separate c from the rest of the network,
$|c|$ is the size of this community.

But if $|c|$ is relatively small (=1), this method may be wrong. Maybe rather than counting all nodes equally in a community, we should give additional weight to "influential", or high-degree nodes.

$$\text{Normalized Cut}(C) = \frac{1}{|C|} \sum_{c \in C} \frac{\text{cut}(c, \bar{c})}{\sum \text{degrees in } c}$$

where ($\sum$ degrees in c) means nodes of high degree will have more influence in the denominator.

3.3.3 Clique percolation

How can we define an algorithm that handles all three types of community (disjoint/overlapping/nested)? Clique percolation is one such algorithm, that discovers communities based on their "cloqueishness"

```
Given a clique size K
Initialize every K–clique as its own community
While (two communities I and J have a (K–1)–clique in common):
        Merge I and J into a single community
```

3.3.4 Network modularity

Null model:
Edges are equally likely between any pair of nodes, regardless of community structure.

$$e_k k = \frac{\#\text{edges with both endpoints in community k}}{\#\text{edges}}$$

$$a_k = \frac{\#\text{edge endpoints in community k}}{\#\text{edge endpoints}}$$

$$Q = \sum_{k=1}^{K} (e_k k - a_k^2)$$

where $e_k k$ is the fraction of edges in k and a_k is the fraction that we would expect if edges were allocated randomly. Q is in $[-\frac{1}{2}, 1]$.
Algorithm: Choose communities so that the deviation from the null model is maximized

$$Q = \sum_{k=1}^{K} (e_k k - a_k^2)$$

$$\arg \max_{\text{communities}} Q(\text{communities})$$

3.3.5 Summary

- Community detection aims to summarize the structure in networks. (as opposed to clustering which aims to summarize feature dimensions)
- Communities can be defined in various ways. depending on the type of network in question

Chapter 4

Recommender Systems

The goal of recommender systems is

- To help people discover new content
- To help us find the content we were already looking for
- To personalize user experiences in response to user feedback
- To recommend incredible products that are relevant to out interests
- To identify things that we like
- to model people's preferences, opinions and behavior

4.1 Recommending things to people

We are treating user and movie features as though they are **independent**

$$f(\text{user features, movie features}) =< \phi(\text{user features}, \theta_{\text{user}} > + < \phi(\text{movie features}, \theta_{\text{movie}} >$$

But these predictors should not be independent. **Recommender Systems** go beyond the methods we've seen so fat by trying to model the relationships between people and items they're evaluating.

4.2 Defining similarity between users & items

We can measure the similarity between two users in terms of the items they purchased !
We can measure the similarity between two items in terms of the users who purchased them!

I_u = set of items purchased by user u
U_u = set of users who purchased item i

$$R = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix}$$

$R_u =$ binary representation of items purchased by u
$R_{\cdot,i} =$ binary representation if users who purchased i
$I_u = \{i|R_{u,i} = 1\} U_i = \{u|R_{u,i} = 1\}$

4.2.1 Euclidean distance

Between two items i, j (similarly defined between two users)

$$|U_i \backslash U_j| + |U_j \backslash U_i| = \|R_i - R_j\|$$

This method favors small sets, even if they have few element in common

4.2.2 Jaccard similarity

$$\text{Jaccard(A,B)} = \frac{|A \bigcap B|}{|A \bigcup B|}$$

Maximum of 1 if the two users purchased exactly the same set of items
Minimum of 0 if the two users purchased completely disjoint sets of items

4.2.3 Cosine similarity

We have opinions in addition to purchases.

$$R = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix}$$
$$\rightarrow \begin{pmatrix} -1 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & -1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & -1 \end{pmatrix}$$

"-1" means bought and liked, "0" means didn't buy and "1" means bought and hated
$\cos(\theta) = 1 \rightarrow \theta = 0$ Rated by the same users, and they all agree
$\cos(\theta) = -1 \rightarrow \theta = 180$ Rated by the same users, but they completely disagree about it.
$\cos(\theta) = 0 \rightarrow \theta = 90$ Rated by different sets of users.

4.2.4 Pearson correlation

What if we have numerical ratings (rather than just thumbs-up/down)?

$$R = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 1 \\ \vdots & & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix}$$
$$\rightarrow \begin{pmatrix} 4 & 0 & \cdots & 2 \\ 0 & 0 & \cdots & 3 \\ \vdots & & \ddots & \vdots \\ 5 & 0 & \cdots & 1 \end{pmatrix}$$

We wouldn't want 1-star ratings to be parallel to 5-star ratings. So we can subtract the average ratings and positive for above-average ratings

$$Sim(u, v) = \frac{\sum_{i \in I_u \bigcap I_v} (R_{u,i} - \bar{R}_u)(R_{v,i} - \bar{R}_v)}{\sqrt{\sum_{i \in I_u \bigcap I_v} (R_{u,i} - \bar{R}_u)^2 \sum_{i \in I_u \bigcap I_v} (R_{v,i} - \bar{R}_v))^2}}$$

Compared to the cosine similarity

$$Sim(u,v) = \frac{\sum_{i \in I_u \bigcap I_v}(R_{u,i})(R_{v,i})}{\sqrt{\sum_{i \in I_u \bigcap I_v}(R_{u,i})^2 \sum_{i \in I_u \bigcap I_v}(R_{v,i}))^2}} = \frac{\sum_u \cdot \sum_v}{\|I_u\| \|I_v\|}$$

4.2.5 Collaborative filtering in practice

How does amazon generate their recommendations?
Give a product i. Let U_i be the set of users who viewed it.
Rank products according to