Chapter 1

Regression Diagnostics

1.1 Coefficient of determination

1.1.1 R^2 statistic

$$\begin{aligned} FVU(f) = \frac{MSE(f)}{Var(y)} \quad &\leftarrow \quad \text{fraction of variance unexplained} \\ R^2 = 1 - FVU(f) = 1 - \frac{MSE(f)}{Var(y)} & \\ R^2 = 0 \quad &\rightarrow \quad \text{Trivial predictor} \\ R^2 = 1 \quad &\rightarrow \quad \text{Perfect predictor} \end{aligned}$$

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

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

Solve:

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

This expression can only be satisfied if φ_j and λ_j are an $\boxed{\text{eigenvectors/eigenvalues}}$ 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)
```