Minimzing MSE for Linear Models

BACK TO. 2-5 NOTES PAGE 2 Complexity of solving equations or finding matrix inverse is $O(Nd^2 + d^3)$

Minimizing MSE for Non-Linear Equations

$MSE(\theta)$ is concave so you can use gradient descent:

$$\theta^{new} = \theta^{current} - stepSize * gradient(MSE)$$

Stochastic Gradient Descent can also be used, faster but noisier

Probabilistic Interpretation of Regression

$p(y|x)$: for fixed x, there is variation in y 2 types of variation: - Measurement noise - Unobserved Variables 2 sources of variability: $p(y|x)$: variability in y given x $p(x)$: distribution of input data in the space We have a joint distribution: $p(x,y) = p(y|x)p(x)$ and we learn $p(y|x)$

Modeling Framework

$y_x = E[y|x] + e$ where
y_x: what we observe
$E[y|x]$: what we try to learn with $f(x,\theta)$
e: unpredictable error term

Simple Model:

$p(y|x) = N(f(x,\theta), \sigma^2)$ where $f(x,\theta) = \theta^T x$

Conditional Likelihood for Regression

$$L(\theta) = \prod p(y_i|x_i, \theta)$$

As an example, we can use a Gaussian model for $p(y|x,\theta)$.
This will give us

$$p(y|x,\theta) = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{1}{2\sigma^2}(y - f(x,\theta))^2)$$

This equation comes from the fact that $E[y|x] = f(x, \theta)$. After doing some algebra

$$logL(\theta) \propto -MSE(\theta)$$

Thus maximizing log likelihood is the same as minimizing MSE. This gives us a useful framework to go beyond.

Bayesian View of Regression

Posterior Density on θ is:

$$p(\theta|D_x, D_y) \propto p(D_x, D_y|\theta)p(\theta) = p(D_y|D_x, \theta)p(D_x|\theta)p(\theta)$$

$$p(\theta|D_x, D_y) \propto p(D_y|D_x, \theta)p(\theta)$$

This is because we do not model $p(D_x|\theta)$

TODO: BACK TO GAUSSIAN ERROR MODEL IN 2-19 NOTES

Properties of Minimizing MSE

Because of this fact:

$$MSE(\theta) = \int\int (y - f(x;\theta))^2 p(x, y)\, dx\, dy$$

It eventually (**TODO: make sure you can derive this. 2-19 notes**) holds that the optimal value is when $f(x;\theta) = E[y_x]$.
We are limited because
Bias: $f(x, \theta)$ might not be able to exactly approximate $E[y_x]$
Variance: Even if so, only have finite data to learn $E[y_x]$
Tradeoff exists between complex model with low bias but high variance and
simple model with high bias and low variance

Bias-Variance Tradeoff

TODO: DERIVE THIS FROM 2-19 NOTES Eventually we have $MSE_x = \sigma_{yx}^2 + Bias^2 + Variance$
This leads to the fundamental bias-variance tradeoff

- simple models with few parameters have high bias, but low variance
- complex models with many parameters have low bias, but high variance

Logistic Regression Classifier

Regression but $y_i \in 0, 1$ and $p(y|x)$ is the logistic function

MAKE SURE TO UNDERSTAND WHERE LOGISTIC FUNCTION COMES FROM

MAKE SURE TO REMEMBER FORMULAS AND IDEA FROM 2-24 NOTES

BACK TO MULTI-CLASS LOGISTIC REGRESSION AND CONNECTION TO FEED-FORWARD NEURAL NETWORKS ON 2-26 NOTES

Generative Approach to Classification

We model $p(x|c)$ instead of $p(c|x)$. Models are generative when we model the distribution of both x's and c's.

$$L(\theta) = \prod p(x_i|c_i, \theta)p(c_i)$$

Key Points:

- learn how x_i values are distributed for each class, so θ_k is set of parameters for class k model.
- learn p(c=k)
- possibly decomposes into k optimization problems
- it is optimal if distributional assumptions are correct
- predict using bayes rule
$$p(c = k|x, \theta) \propto p(x|c = k, \theta_k)p(c = k)$$
or we could be Bayesian and average over the θ values.

Weaknesses of Gaussian model for each class:

- sensitive to Gaussian assumption
- scales poorly as d increases. for high dimensions, we can assume covariance matrices are diagonal.

Naive Bayes Model:
If x_i are binary vectors, we can use a Naive Bayes model for each class. Parameters are $\theta_k = \{\theta_{k1}, ..., \theta_{kd}\}$ where θ_{kj} is Bernoulli probability that $x_{ij} = 1$
*FIRST ORDER MARKOV MODEL EXAMPLE WILL NOT BE ON THE FINAL**

Discriminant Functions

To make a decision about mostly class, we can compute $argmax_k \, p(c = k|x)$
Using Bayes rule, this is $argmax_k \, p(x|c = k)p(c = k)$
This is equal to $argmax_k \, log(p(x|c = k)) + log(p(c = k))$
All of these can be used as discriminant functions $g_k(x)$

For 2 class case, decision boundary is when $g(x) = g_1(x) - g_2(x) = 0$

BACK TO MULTIVARIATE GAUSSIAN DISCRIMINANT FUNCTION FROM 2-26

CONTINUE STUDYING AT 3-3 NOTES