Introduction to Machine Learning

Bayesian Regression

Varun Chandola

March 31, 2020

Outline

Contents

1 Linear Regression

1.1 Problem Formulation

- There is one scalar **target** variable y (instead of hidden)
- There is one vector **input** variable x
- Inductive bias:

$$y = \mathbf{w}^\top \mathbf{x}$$

Linear Regression Learning Task
Learn $\mathbf{w}$ given training examples, $\langle \mathbf{X}, \mathbf{y} \rangle$.

The training data is denoted as $\langle \mathbf{X}, \mathbf{y} \rangle$, where $\mathbf{X}$ is a $N \times D$ data matrix consisting of N data examples such that each data example is a D dimensional vector. $\mathbf{y}$ is a $N \times 1$ vector consisting of corresponding target values for the examples in $\mathbf{X}$.

- y is assumed to be normally distributed

$$y \sim \mathcal{N}(\mathbf{w}^\top \mathbf{x}, \sigma^2)$$

- or, equivalently:

$$y = \mathbf{w}^\top \mathbf{x} + \epsilon$$

  where $\epsilon \sim \mathcal{N}(0, \sigma^2)$
- y is a *linear combination* of the input variables
- *Given* $\mathbf{w}$ *and* σ^2*, one can find the probability distribution of* y *for a given* $\mathbf{x}$

1.2 Learning Parameters

- Find $\mathbf{w}$ and σ^2 that maximize the likelihood of training data

$$
\begin{aligned}
\widehat{\mathbf{w}}_{MLE} &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \\
\widehat{\sigma}^2_{MLE} &= \frac{1}{N}(\mathbf{y}-\mathbf{X}\mathbf{w})^\top(\mathbf{y}-\mathbf{X}\mathbf{w})
\end{aligned}
$$

The derivation of the MLE estimates can be done by maximizing the log-likelihood of the data set. The likelihood of the training data set is given by:

$$L(\mathbf{w}) = \prod_{i=1}^{N}\frac{1}{\sqrt{2\pi}\sigma}exp(-\frac{(y_i-\mathbf{w}^\top\mathbf{x_i})^2}{2\sigma^2})$$

The log-likelihood is given by:

$$LL(\mathbf{w}) = -\frac{1}{2}\log 2\pi - \log\sigma - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y_i-\mathbf{w}^\top\mathbf{x}_i)^2$$

This can be rewritten in matrix notation as:

$$LL(\mathbf{w}) = -\frac{1}{2}\log 2\pi - \log\sigma - \frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\mathbf{w})^\top(\mathbf{y}-\mathbf{X}\mathbf{w})$$

To maximize the log-likelihood, we first compute its derivative with respect to $\mathbf{w}$ and σ.

$$
\begin{aligned}
\frac{\partial LL(\mathbf{w})}{\partial\mathbf{w}} &= -\frac{1}{2\sigma^2}\frac{\partial}{\partial\mathbf{w}}(\mathbf{y}-\mathbf{X}\mathbf{w})^\top(\mathbf{y}-\mathbf{X}\mathbf{w}) \\
&= -\frac{1}{2\sigma^2}\frac{\partial}{\partial\mathbf{w}}(\mathbf{y}^\top\mathbf{y}+\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w}-2\mathbf{y}^\top\mathbf{X}\mathbf{w})
\end{aligned}
$$

Note that, we use the fact that $(\mathbf{X}\mathbf{w})^\top\mathbf{y} = \mathbf{y}^\top\mathbf{X}\mathbf{w}$, since both quantities are scalars and the transpose of a scalar is equal to itself. Continuing with the derivative:

$$\frac{\partial LL(\mathbf{w})}{\partial\mathbf{w}} = -\frac{1}{2\sigma^2}(2\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}-2\mathbf{y}^\top\mathbf{X})$$

Setting the derivative to 0, we get:

$$
\begin{aligned}
2\mathbf{w}^\top\mathbf{X}^\top\mathbf{X}-2\mathbf{y}^\top\mathbf{X} &= 0 \\
\mathbf{w}^\top\mathbf{X}^\top\mathbf{X} &= \mathbf{y}^\top\mathbf{X} \\
(\mathbf{X}^\top\mathbf{X})^\top\mathbf{w} &= \mathbf{X}^\top\mathbf{y} \text{ (Taking transpose both sides)} \\
(\mathbf{X}^\top\mathbf{X})\mathbf{w} &= \mathbf{X}^\top\mathbf{y} \\
\mathbf{w} &= (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}
\end{aligned}
$$

In a similar fashion, one can set the derivative to 0 with respect to σ and plug in the the optimal value of $\mathbf{w}$

1.3 Issues with Linear Regression

1. Not truly Bayesian
2. Susceptible to outliers
3. *Too simplistic* - Underfitting
4. No way to control overfitting
5. Unstable in presence of correlated input attributes
6. Gets "confused" by unnecessary attributes

2 Bayesian Linear Regression

3 Bayesian Regression

- "Penalize" large values of $\mathbf{w}$
- A zero-mean Gaussian prior

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|0, \tau^2 I)$$

- What is posterior of $\mathbf{w}$

$$p(\mathbf{w}|\mathcal{D}) \propto \prod_i \mathcal{N}(y_i|\mathbf{w}^\top\mathbf{x}_i, \sigma^2)p(\mathbf{w})$$

- Posterior is also Gaussian
- Regularized least squares estimate of $\mathbf{w}$

$$\arg\max_{\mathbf{w}} \sum_{i=1}^{N} log\mathcal{N}(y_i|\mathbf{w}^\top \mathbf{x}_i, \sigma^2) + \log \mathcal{N}(\mathbf{w}|0, \tau^2 I)$$

3.1 Estimating Bayesian Regression Parameters

- Prior for $\mathbf{w}$

$$\mathbf{w} \sim \mathcal{N}(\mathbf{w}|\mathbf{0}, \tau^2 \mathbf{I}_D)$$

- Posterior for $\mathbf{w}$

$$\begin{aligned} p(\mathbf{w}|\mathbf{y}, \mathbf{X}) &= \frac{p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{y}|\mathbf{X})} \\ &= \mathcal{N}(\bar{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X} + \frac{\sigma^2}{\tau^2}\mathbf{I}_N)^{-1}\mathbf{X}^\top \mathbf{y}, \sigma^2(\mathbf{X}^\top \mathbf{X} + \frac{\sigma^2}{\tau^2}\mathbf{I}_N)^{-1}) \end{aligned}$$

- Posterior distribution for $\mathbf{w}$ is also Gaussian
- What will be MAP estimate for $\mathbf{w}$?

The denominator term in the posterior above can be computed as the marginal likelihood of data by marginalizing $\mathbf{w}$:

$$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})d\mathbf{w}$$

One can compute the posterior for $\mathbf{w}$ as follows. We first show that the likelihood of $\mathbf{y}$, i.e., all target values in the training data, can be jointly modeled as a Gaussian as follows:

$$\begin{aligned} p(\mathbf{y}|\mathbf{X}, \mathbf{w}) &= \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(-\frac{1}{2\sigma^2}(y_i - \mathbf{w}\top \mathbf{x}_i)^2\right) \\ &= \frac{1}{(2\pi\sigma^2)^{N/2}} exp\left(-\frac{1}{2\sigma^2}|\mathbf{y} - \mathbf{X}\mathbf{w}|^2\right) \\ &= \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2 \mathbf{I}_N) \end{aligned}$$

Ignoring the denominator which does not depend on $\mathbf{w}$:

$$\begin{aligned} p(\mathbf{w}|\mathbf{y}, \mathbf{X}) &\propto exp(-\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w})) exp(-\frac{1}{2\tau^2}\mathbf{w}^\top \mathbf{w}) \\ &\propto exp(-\frac{1}{2}(\mathbf{w} - \bar{\mathbf{w}})^\top (\frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{X} + \frac{1}{\tau^2}\mathbf{I}_N)(\mathbf{w} - \bar{\mathbf{w}})) \end{aligned}$$

where $\bar{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X} + \frac{\sigma^2}{\tau^2}\mathbf{I}_N)^{-1}\mathbf{X}\mathbf{y}$.

3.2 Prediction with Bayesian Regression

- For a new $\mathbf{x}^*$, predict y^*
- Point estimate of y^*

$$y^* = \widehat{\mathbf{w}}_{MLE}^\top \mathbf{x}^*$$

- Treating y as a Gaussian random variable

$$p(y^*|\mathbf{x}^*) = \mathcal{N}(\widehat{\mathbf{w}}_{MLE}^\top \mathbf{x}^*, \widehat{\sigma}_{MLE}^2)$$

$$p(y^*|\mathbf{x}^*) = \mathcal{N}(\widehat{\mathbf{w}}_{MAP}^\top \mathbf{x}^*, \widehat{\sigma}_{MAP}^2)$$

- Treating y and $\mathbf{w}$ as random variables

$$p(y^*|\mathbf{x}^*) = \int p(y^*|\mathbf{x}^*, \mathbf{w})p(\mathbf{w}|\mathbf{X}, \mathbf{y})d\mathbf{w}$$

- This is also *Gaussian*!

4 Handling Outliers in Regression

- Linear regression training gets impacted by the presence of outliers
- The square term in the exponent of the Gaussian pdf is the culprit
  - Equivalent to the square term in the loss
- How to handle this (*Robust Regression*)?
- Probabilistic:

- Use a different distribution instead of Gaussian for $p(y|\mathbf{x})$
- Robust regression uses Laplace distribution

$$p(y|\mathbf{x}) \sim Laplace(\mathbf{w}^\top \mathbf{x}, b)$$

- Geometric:
  - *Least absolute deviations* instead of least squares

$$J(\mathbf{w}) = \sum_{i=1}^{N} |y_i - \mathbf{w}^\top \mathbf{x}|$$

5 Generative vs. Discriminative Classifiers

- Probabilistic classification task:

$$p(Y = benign|\mathbf{X} = \mathbf{x}), p(Y = malicious|\mathbf{X} = \mathbf{x})$$

- How do you estimate $p(y|\mathbf{x})$?

$$p(y|\mathbf{x}) = \frac{p(y, \mathbf{x})}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

- Two step approach - Estimate generative model and then posterior for y (Naïve Bayes)
- Solving a more general problem [2, 1]
- Why not directly model $p(y|\mathbf{x})$? - **Discriminative approach**
- Number of training examples needed to learn a PAC-learnable classifier $\propto$ *VC-dimension of the hypothesis space*
- VC-dimension of a probabilistic classifier $\propto$ Number of parameters [2] (or a small polynomial in the number of parameters)
- Number of parameters for $p(y, \mathbf{x})$ > Number of parameters for $p(y|\mathbf{x})$

Discriminative classifiers need lesser training examples to for PAC learning than generative classifiers

6 Bayesian Logistic Regression

- $y|\mathbf{x}$ is a *Bernoulli* distribution with parameter $\theta = sigmoid(\mathbf{w}^\top \mathbf{x})$
- When a new input $\mathbf{x}^*$ arrives, we toss a coin which has $sigmoid(\mathbf{w}^\top \mathbf{x}^*)$ as the probability of heads
- If outcome is heads, the predicted class is 1 else 0
- Learns a linear boundary

Learning Task for Logistic Regression
Given training examples $\langle \mathbf{x}_i, y_i \rangle_{i=1}^{D}$, learn $\mathbf{w}$

7 Logistic Regression

Bayesian Interpretation

- Directly model $p(y|\mathbf{x})$ $(y \in \{0, 1\})$
- $p(y|\mathbf{x}) \sim Bernoulli(\theta = sigmoid(\mathbf{w}^\top \mathbf{x}))$

Geometric Interpretation

- Use regression to predict discrete values
- *Squash* output to $[0, 1]$ using sigmoid function
- Output less than 0.5 is one class and greater than 0.5 is the other

8 Logistic Regression - Training

- MLE Approach
- Assume that $y \in \{0, 1\}$
- What is the likelihood for a bernoulli sample?
  - If $y_i = 1$, $p(y_i) = \theta_i = \frac{1}{1+exp(-\mathbf{w}^\top \mathbf{x}_i)}$

– If $y_i = 0$, $p(y_i) = 1 - \theta_i = \frac{1}{1+exp(\mathbf{w}^\top \mathbf{x}_i)}$

– In general, $p(y_i) = \theta_i^{y_i}(1-\theta_i)^{1-y_i}$

Log-likelihood

$$LL(\mathbf{w}) \quad = \quad \sum_{i=1}^{N} y_i \log \theta_i + (1-y_i)\log(1-\theta_i)$$

- No closed form solution for maximizing log-likelihood

To understand why there is no closed form solution for maximizing the log-likelihood, we first differentiate $LL(\mathbf{w})$ with respect to $\mathbf{w}$. We make use of the useful result for sigmoid:

$$\frac{d\theta_i}{d\mathbf{w}} = \theta_i(1-\theta_i)\mathbf{x}_i$$

Using this result we obtain:

$$\begin{aligned}
\frac{d}{d\mathbf{w}}LL(\mathbf{w}) &= \sum_{i=1}^{N} \frac{y_i}{\theta_i}\theta_i(1-\theta_i)\mathbf{x}_i - \frac{(1-y_i)}{1-\theta_i}\theta_i(1-\theta_i)\mathbf{x}_i \\
&= \sum_{i=1}^{N} (y_i(1-\theta_i) - (1-y_i)\theta_i)\mathbf{x}_i \\
&= \sum_{i=1}^{N} (y_i - \theta_i)\mathbf{x}_i
\end{aligned}$$

Obviously, given that θ_i is a non-linear function of $\mathbf{w}$, a closed form solution is not possible.

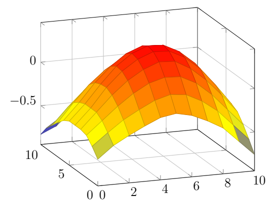
8.1 Using Gradient Descent for Learning Weights

- Compute gradient of LL with respect to $\mathbf{w}$
- A convex function of $\mathbf{w}$ with a unique global maximum

$$\frac{d}{d\mathbf{w}}LL(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \theta_i)\mathbf{x}_i$$

- Update rule:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta \frac{d}{d\mathbf{w}_k} LL(\mathbf{w}_k)$$



8.2 Using Newton's Method

- Setting η is sometimes *tricky*
- Too large – incorrect results
- Too small – slow convergence
- Another way to speed up convergence:

Newton's Method

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta \mathbf{H}_k^{-1} \frac{d}{d\mathbf{w}_k} LL(\mathbf{w}_k)$$

- Hessian or $\mathbf{H}$ is the second order derivative of the objective function
- Newton's method belong to the family of **second order optimization algorithms**
- For logistic regression, the Hessian is:

$$H = -\sum_i \theta_i(1-\theta_i)\mathbf{x}_i\mathbf{x}_i^\top$$

8.3 Regularization with Logistic Regression

- **Overfitting** is an issue, especially with large number of features
- Add a *Gaussian prior* $\sim \mathcal{N}(\mathbf{0}, \tau^2)$
- Easy to incorporate in the gradient descent based approach

$$LL'(\mathbf{w}) = LL(\mathbf{w}) - \frac{1}{2}\lambda\mathbf{w}^\top\mathbf{w}$$

$$\frac{d}{d\mathbf{w}}LL'(\mathbf{w}) = \frac{d}{d\mathbf{w}}LL(\mathbf{w}) - \lambda\mathbf{w}$$

$$H' = H - \lambda I$$

where I is the identity matrix.

8.4 Handling Multiple Classes

- $p(y|\mathbf{x}) \sim Multinoulli(\boldsymbol{\theta})$
- Multinoulli parameter vector $\boldsymbol{\theta}$ is defined as:

$$\theta_j = \frac{exp(\mathbf{w}_j^\top\mathbf{x})}{\sum_{k=1}^{C} exp(\mathbf{w}_k^\top\mathbf{x})}$$

- Multiclass logistic regression has C weight vectors to learn

8.5 Bayesian Logistic Regression

- How to get the posterior for $\mathbf{w}$?
- Not easy - *Why?*

Laplace Approximation

- We do not know what the true posterior distribution for $\mathbf{w}$ is.
- Is there a close-enough (approximate) Gaussian distribution?

One should note that we used a Gaussian prior for $\mathbf{w}$ which is not a conjugate prior for the Bernoulli distribution used in the logistic regression. In fact there is no convenient prior that may be used for logistic regression.

References

References

[1] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *NIPS*, pages 841–848. MIT Press, 2001.

[2] V. Vapnik. *Statistical learning theory.* Wiley, 1998.