CSCI-GA.2565-001 Machine Learning: Part 1

Chuanyang Jin

October 13, 2022

1 Linear Regression Model

Consider the data generating process as such: $\mathbf{x} \in \mathbb{R}^D$ is drawn from some unknown $p(\mathbf{x})$ and $y = w_1^{true} x_1 + \epsilon_y$, where $w_1^{true} \in \mathbb{R}$ and $\epsilon_y \sim \mathcal{N}(0, 1)$. This is unknown to us, as a result, we construct a linear model for y using all D features of $\mathbf{x}$, instead of just using x_1.

(A) Explain what the terms **model class** and **model misspecification** mean. Is our model correctly *specified* here? Why or why not?

Solution.

Model class means the set of possible predictive models based on your assumptions and learning algorithm.

Model misspecification means the model has a wrong form that poorly represents the data-generating process, so it's hard to approach the true distribution.

Our model is correctly specified here, since our model contains the true relationship (when the coefficients of all features except x_1 are zeros), and can represent the true distribution.

□

Let $\widehat{w_1}$ be the estimate of w_1^{true} using only x_1 and let $\widehat{w_1^{all}}$ be the estimate of w_1^{true} when using all of $\mathbf{x}$. We will study the effects of our model by analyzing the relationships between $\mathbb{E}[\widehat{w_1^{all}}]$ and $\mathbb{E}[\widehat{w_1}]$, as well as between $\text{Var}[\widehat{w_1^{all}}]$ and $\text{Var}[\widehat{w_1}]$. We do so empirically by running PyTorch simulations as follows:

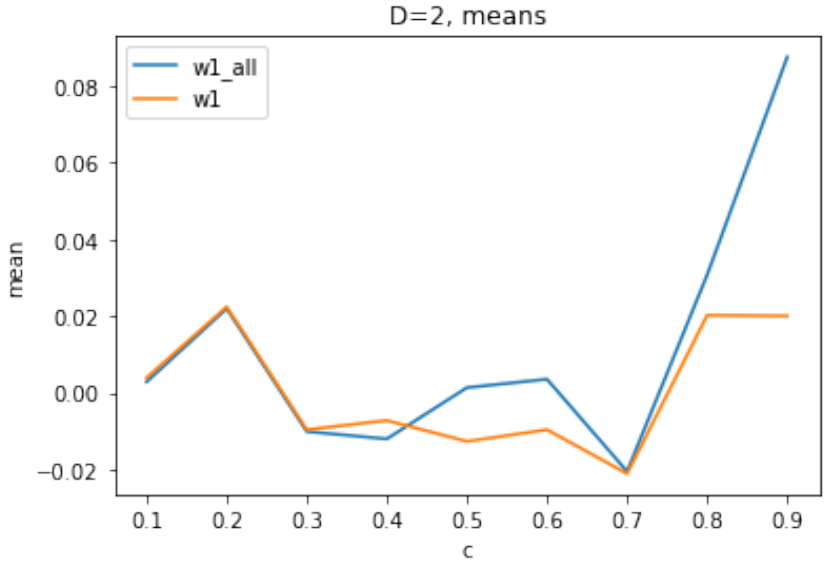
1. Pick any value of w_1^{true} you like as ground truth, e.g. with `torch.randn(1)`.
2. Write a function, taking D and c as input, that does the following: **(1)** Generate $N = 50$ samples of $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, where Σ is the $D \times D$ covariance matrix with all diagonal entries equal to $\sigma^2 = 1$ and all off-diagonal entries equal to c. **(2)** Compute y using the relationship above (note that y only depends on the first feature). This involves drawing N samples of noise $\epsilon_y \sim \mathcal{N}(0, 1)$. **(3)** Using our dataset of N samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, compute the least-squares solutions for $\widehat{w_1^{all}}$ (i.e. using all features and taking the first coefficient) and $\widehat{w_1}$ (i.e. using only one feature).
3. Write a function that performs Step 2 for $T = 100$ trials, i.e. each trial generates a new dataset to compute $\widehat{w_1^{all}}$ and $\widehat{w_1}$. (Note that w_1^{true} is constant throughout.) For each estimator, compute the mean and standard deviation of the T trials.
4. Perform Step 3 for each $c \in \{0.1, 0.2, 0.3, \ldots, 0.9\}$ and each $D \in \{2, 4, 8, 16, 32\}$. Separately plot the means and standard deviations as a function of c, using the same plot for both estimators. This means you should have 10 plots altogether: two plots (means and standard deviations) for each of the five choices of D. Each plot will contain two curves (the two estimators).

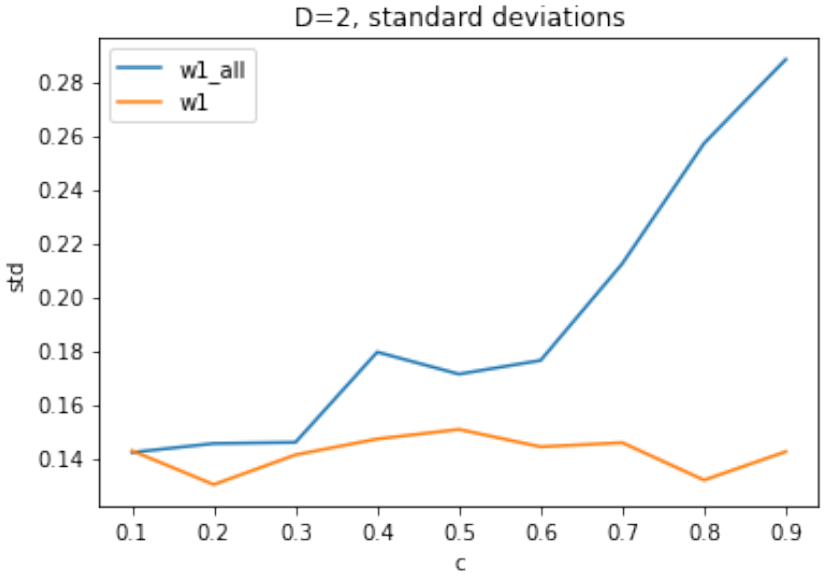
(B) What do you observe with respect to c and D? How do you explain your results? Show a few (not all) of your 10 generated plots to support your answer.

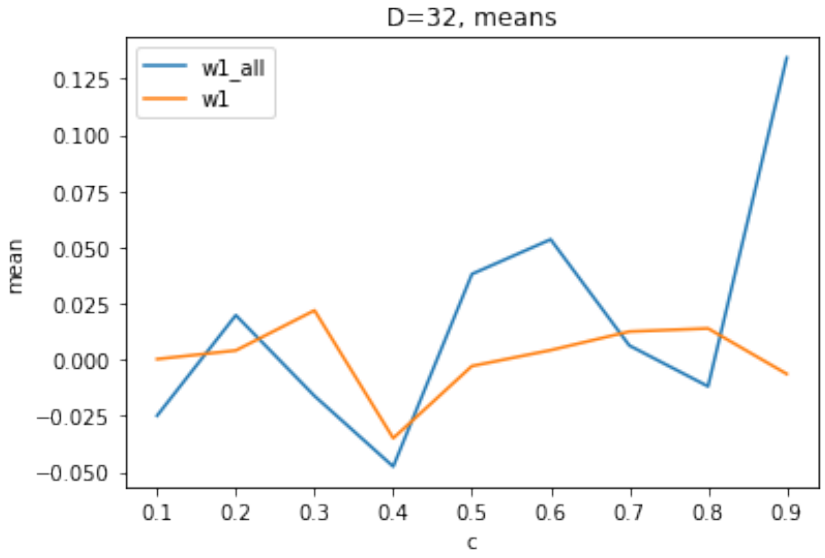
Solution.

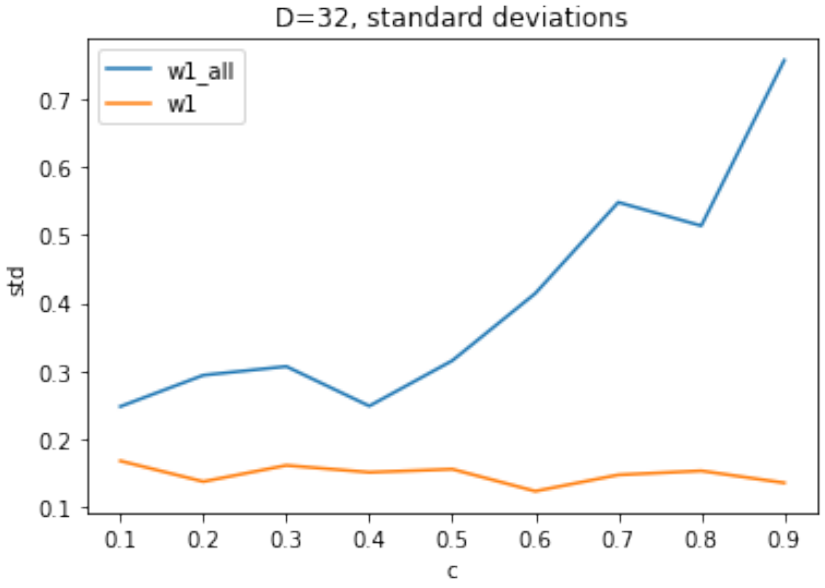
As c increases, the variance of $\widehat{w_1^{all}}$ increases, while the variance of $\widehat{w_1}$ remains small; the mean of $\widehat{w_1^{all}}$ fluctuates away from w_1^{true}, while the mean of $\widehat{w_1^{all}}$ fluctuates around w_1^{true}. This is because as c increases, the weights of other features are more and more related to the weight of x_1.

As D increases, the variance of $\widehat{w_1^{all}}$ increases, while the variance of $\widehat{w_1}$ remains small; the mean of $\widehat{w_1^{all}}$ fluctuates away from w_1^{true}, while the mean of $\widehat{w_1^{all}}$ fluctuates around w_1^{true}. This is because as D increases, more and more features will influence the first estimator.






D=32, means
w1_all
w1
mean
c
0.125
0.100
0.075
0.050
0.025
0.000
−0.025
−0.050
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9


D=32, standard deviations
w1_all
w1
std
c
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

□

2 Bayesian Linear Regression Model

Consider the data generating process as such: $x \sim \mathcal{N}(0,1)$ and $y = w_{true}x^2 + \epsilon$, where $w_{true} = 1.0$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, and $\sigma^2 = 1.0$. Again, this is unknown to us. We will model the data using Bayesian linear regression.

(A) Using PyTorch, simulate a dataset $\mathcal{D}_N = \{(x_i, y_i)\}_{i=1}^N$ for each $N \in \{10, 100, 1000, 10000\}$ according to the true data generating process above. For our Bayesian linear regression model, let us choose our prior as $w \sim N(0,1)$ and our likelihood as $y|w, x \sim N(wx, \sigma^2)$ for $\sigma^2 = 1.0$. Compute the mean and variance of the posterior $w|\mathcal{D}_N$ for each dataset. Does the posterior concentrate on w_{true}? Why or why not?

```
N: 10
mean: tensor(-1.0254) variance: tensor(0.1416)
N: 100
mean: tensor(-0.7935) variance: tensor(0.0101)
N: 1000
mean: tensor(0.2106) variance: tensor(0.0009)
N: 10000
mean: tensor(0.0225) variance: tensor(0.0001)
```

Solution.

The posterior does not concentrate on w_{true}. While $w_{true} = 1$, our posterior comes nearer and nearer to 0.
The reason is that our model is based on a wrong hypothesis, and tries to model the quadratic relationship with linear regression. □

(B) What would be challenging about our analysis in part (A) if we had picked a different prior, for example, Laplace or Gamma?

Solution.

When our prior is Gaussian, after we multiply $p(w)$ and $p(y|x, w)$, our posterior will still follow a Gaussian distribution. Then the distribution can be used to predict or viewed as a new prior when we observe more data.
However, multiplying two Laplaces or Gammas may not result in a Laplace or Gamma. Then the category of our posterior remain uncertain when we try to predict or observe more data. □

(C) Repeat part (A), except we use the basis set $\boldsymbol{\phi}(x) = [x, x^2]$ (instead of x itself) and perform 2D Bayesian linear regression. We choose our prior to be $\mathbf{w} \sim N(\mathbf{0}, \mathbf{I})$, where $\mathbf{I}$ is the 2×2 identity matrix, and our likelihood to be $y|\mathbf{w}, x \sim N(\mathbf{w}^\top \boldsymbol{\phi}(x), \sigma^2)$ for $\sigma^2 = 1.0$. Compute the mean and variance of the posterior $\mathbf{w}|\mathcal{D}_N$ for each dataset. What do you observe about the posterior as N changes? Why?

```
N: 10
mean: tensor([-0.0392,  0.5788])
variance: tensor([[0.1426, 0.0544],
        [0.0544, 0.0558]])
N: 100
mean: tensor([-0.1408,  1.0154])
variance: tensor([[0.0100, 0.0007],
        [0.0007, 0.0040]])
N: 1000
mean: tensor([-0.0328,  1.0190])
variance: tensor([[ 9.7950e-04, -1.5732e-05],
        [-1.5732e-05,  3.3155e-04]])
N: 10000
mean: tensor([-0.0047,  1.0016])
variance: tensor([[ 9.9907e-05, -1.4765e-06],
        [-1.4765e-06,  3.2912e-05]])
```

Solution.

As N grows larger, the posterior weight of x^2 concentrates on $w_{true} = 1$, and the posterior weight of x converges to 0.
It is approaching the true values, since we model the quadratic relationship in a quadratic way correctly.
The more data we observe, the more knowledge we will have. □

(D) Reflecting on your answers in parts (A) and (C), name one challenge that **cannot be solved** by using a Bayesian model (instead of a frequentist approach like standard linear regression).

Solution.

When the dataset is too small, the predictions of a Bayesian model will be largely influenced by the prior. We may not gain enough new knowledge to make the accurate prediction.
Moreover, a Bayesian model usually requires much more computation than a frequentist approach. □

(E) Reflecting on your answers in part (C) and in Question 1, name one challenge that **can be improved** by using a Bayesian model.

Hint: Both part (C) above and Question 1 involve doing linear regression with many correlated features. How do these two sets of findings relate? Does using a Bayesian approach affect the way the model treats correlated features?

Solution.

When there are many correlated features, a frequentist approach may have trouble concentrating on the true value (as in the situation of Question 1 when c is large). However, a Bayesian model can identify those correlation between features by maintaining an estimation of the covariance among the feature weights. Moreover, such patterns may be represented by a proper prior and likelihood. □

3 Class-Conditional Gaussian Generative Model

Consider a classification task where $\mathbf{x} \in \mathbb{R}^D$ and $y \in \{1, \ldots, K\}$. We observe the dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Let us construct a model for the joint distribution as

$$p_\theta(\mathbf{x}, y) = p_\theta(\mathbf{x}|y)p_\theta(y)$$

where θ denotes the set of all parameters of the model.

(A) Our model is known as a **class-conditional generative model**. What about the model makes it generative? What makes it class-conditional?

Solution.

The model is generative because it models the distribution of data, and one can sample new data from this distribution.
The model is class-conditional because it first estimates $p_\theta(\mathbf{x}, y)$ conditioned on each class. □

(B) For a given value of θ, how would you predict the label for a new test point $\mathbf{x}_\star$ using your model $p_\theta(\mathbf{x}, y)$?

Solution.

We predict the label $y_\star$ that maximizes the likelihood $p_\theta(y_\star|\mathbf{x}_\star)$. Thus

$$y_\star = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p_\theta(y = k|\mathbf{x}_\star) = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; \frac{p_\theta(\mathbf{x}_\star, y = k)}{p(\mathbf{x}_\star)} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; p_\theta(\mathbf{x}_\star, y = k)$$

□

Let us model y as a Categorical distribution Cat($\boldsymbol{\pi}$). Here $p_\theta(y = k) \triangleq \pi_k$, where $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]$ such that $\forall k$, $\pi_k \geq 0$ and $\sum_k \pi_k = 1$. You may leave $p_\theta(\mathbf{x}|y)$ unspecified for now.

(C) Write down an expression for the log-likelihood of the observed dataset $\mathcal{D}$.

Solution.

$$\begin{aligned}
\log p_\theta(\mathcal{D}) &= \log \prod_{i=1}^N p_\theta(\mathbf{x}_i, y_i) \\
&= \sum_{i=1}^N \log p(\mathbf{x}_i, y_i) \\
&= \sum_{i=1}^N \log p_\theta(\mathbf{x}_i|y_i)p_\theta(y_i) \\
&= \sum_{i=1}^N \log p_\theta(\mathbf{x}_i|y_i)\pi_{y_i} \\
&= \sum_{i=1}^N \log p_\theta(\mathbf{x}_i|y_i) + \sum_{i=1}^N \log \pi_{y_i}
\end{aligned}$$

□

(D) Derive an expression for the maximum likelihood estimator (MLE) for $\boldsymbol{\pi}$, which we will denote as $\widehat{\boldsymbol{\pi}}$. Make sure to account for the constraints on $\boldsymbol{\pi}$ using Lagrange multipliers.

Solution.

Constraints on $\boldsymbol{\pi}$: $\sum_k \pi_k = 1$.
The Lagrangian is

$$\log p_\theta(D) = \sum_{i=1}^{N} \log p_\theta(\mathbf{x}_i|y_i) + \sum_{i=1}^{N} \log \pi_{y_i} + \lambda(\sum_k \pi_k - 1)$$

For all i, we set the gradients with respect to π_i equal to 0, then

$$\frac{\log p_\theta(D)}{\pi_i} = \frac{\sum_{j=1}^{N} 1(y_j = i)}{\pi_i} + \lambda = 0$$

$$\pi_i = -\frac{\sum_{j=1}^{N} 1(y_j = i)}{\lambda}$$

We set the gradients with respect to λ equal to 0, then

$$\sum_k \pi_k - 1 = \sum_k \Big[-\frac{\sum_{j=1}^{N} 1(y_j = i)}{\lambda} \Big] - 1 = -\frac{N}{\lambda} - 1 = 0$$

$$\lambda = -N$$

So

$$\pi_i = \frac{\sum_{j=1}^{N} 1(y_j = i)}{N}$$

In other words, π_i equals to the ratio of the number of data with label i to the number of all data N. □

Let us further model $\mathbf{x}|y$ as (multivariate) Gaussian distributions $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k)$ for all K classes, where $\boldsymbol{\mu}_k \in \mathbb{R}^D$ and Σ_k is a $D \times D$ (positive semi-definite) covariance matrix. Assume that there are only $K = 2$ classes. This means that the total set of parameters are $\theta = \{\boldsymbol{\pi}, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \Sigma_1, \Sigma_2\}$.

Now, consider the case where the *true* data comes from this model. That is, $y \sim \text{Cat}(\boldsymbol{\pi}^{true})$ and $\mathbf{x}|y = k \sim \mathcal{N}(\boldsymbol{\mu}_k^{true}, \Sigma_k^{true})$ for all k. If we observe some of this data, we can then construct a *discriminative model* to predict y from $\mathbf{x}$, that is, we will learn a model for $p_{true}(y = k|\mathbf{x})$. Let us do so using **logistic regression**:

$$y|\mathbf{x} \sim \text{Bernoulli}(\sigma(\mathbf{w}^\top \mathbf{x}))$$

where $\mathbf{w}$ are the parameters of the model and $\sigma(z)$ is the logistic sigmoid:

$$\sigma(z) = \frac{1}{1 + \exp[-z]}$$

(E) Will logistic regression always be able to model the true data conditional $p_{true}(y = k|\mathbf{x})$? If so, why? If sometimes, when? And if there are any cases where logistic regression will not be able to model $p_{true}(y = k|\mathbf{x})$, are there any ways to fix it?

Solution.

The logistic regression will not always be able to model the true data conditional $p_{true}(y = k|\mathbf{x})$

$$p_{true}(y = 1|\mathbf{x}) = \frac{p_{true}(y = 1|\mathbf{x})p_{true}(y = 1)}{p_{true}(y = 1|\mathbf{x})p_{true}(y = 1) + p_{true}(y = 2|\mathbf{x})p_{true}(y = 2)}$$

$$= \frac{1}{1 + \dfrac{p_{true}(y = 2|\mathbf{x})\pi_2}{p_{true}(y = 1|\mathbf{x})\pi_1}}$$

where

$$\frac{p_{true}(y=2|\mathbf{x})\pi_2}{p_{true}(y=1|\mathbf{x})\pi_1} = \frac{\pi_2}{\pi_1} \cdot \frac{\det(2\pi\Sigma_2)^{-\frac{1}{2}} \exp[-\frac{1}{2}(\mathbf{x}-\mu_2)^T\Sigma_2^{-1}(\mathbf{x}-\mu_2)]}{\det(2\pi\Sigma_1)^{-\frac{1}{2}} \exp[-\frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma_1^{-1}(\mathbf{x}-\mu_1)}$$

$$= \exp\left[\frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma_1^{-1}(\mathbf{x}-\mu_1) - \frac{1}{2}(\mathbf{x}-\mu_2)^T\Sigma_2^{-1}(\mathbf{x}-\mu_2) + \log\sqrt{\frac{\pi_2^2\det\Sigma_1}{\pi_1^2\det\Sigma_2}}\right]$$

If we define $f_\theta(\mathbf{x}) := \frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma_1^{-1}(\mathbf{x}-\mu_1) - \frac{1}{2}(\mathbf{x}-\mu_2)^T\Sigma_2^{-1}(\mathbf{x}-\mu_2) + \log\sqrt{\frac{\pi_2^2\det\Sigma_1}{\pi_1^2\det\Sigma_2}}$, then

$$p_{true}(y=1|\mathbf{x}) = \frac{1}{1+\exp[-f_\theta(\mathbf{x})]}$$

The logistic regression is in the form

$$p(y=1|\mathbf{x}) = \sigma(z) = \frac{1}{1+\exp[-z]} = \frac{1}{1+\exp[-w^T\mathbf{x}]}$$

Since the true $f_\theta(\mathbf{x})$ is a quadratic function of $\mathbf{x}$, while we model it with a linear function of $\mathbf{x}$, the logistic regression will not always be able to model the true data.
It sometimes can, when the true data makes $f_\theta(\mathbf{x})$ be linear, that is when $\Sigma_1 = \Sigma_2 = \Sigma$,

$$f_\theta(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1) - \frac{1}{2}(\mathbf{x}-\mu_2)^T\Sigma^{-1}(\mathbf{x}-\mu_2) + \log\sqrt{\frac{\pi_2^2\det\Sigma}{\pi_1^2\det\Sigma}}$$

$$= \frac{1}{2}(-\mathbf{x}^T\Sigma^{-1}\mu_1 - \mu_1^T\Sigma^{-1}\mathbf{x} + \mu_1^T\Sigma^{-1}\mu_1 + \mathbf{x}^T\Sigma^{-1}\mu_2 + \mu_2^T\Sigma^{-1}\mathbf{x} - \mu_2^T\Sigma^{-1}\mu_2) + \log\frac{\pi_2}{\pi_1}$$

$$= \frac{1}{2}(\Sigma^{-1}\mu_2 + \mu_2^T\Sigma^{-1} - \Sigma^{-1}\mu_1 - \mu_1^T\Sigma^{-1})\mathbf{x} + \frac{1}{2}(\mu_1^T\Sigma^{-1}\mu_1 - \mu_2^T\Sigma^{-1}\mu_2) + \log\frac{\pi_2}{\pi_1}$$

To fix the logistic regression to be able to model all $p_{true}(y=k|\mathbf{x})$, we can introduce basis functions to represent terms such as x_1x^2, x_1x_2, or we can replace the $w^T\mathbf{x}$ in our model with $\mathbf{x}^T\mathbf{W}\mathbf{x}$ where $\mathbf{W}$ is a $D \times D$ parameter matrix. □

4 Poisson Generalized Linear Model

Consider a classification task where $\mathbf{x} \in \mathbb{R}^D$ and $y \in \mathbb{N} = \{0, 1, 2, 3, \dots\}$, noting that the support of y is the unbounded set of natural numbers. We have an observed dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Let us also assume that the number of features, D, is larger than the number of examples, N. We will model this data using a Poisson Generalized Linear Model (GLM). Let θ denote the linear coefficients of the model.

(A) Write down the log-likelihood function of the Poisson GLM.

Solution.

For Poisson distribution, $p(y = k|\mathbf{x}, \boldsymbol{\theta}) = e^{-\lambda}\frac{\lambda^k}{k!} = \exp[\log\frac{\lambda^k}{k!} - \lambda] = \exp[k\log\lambda - \lambda - \log k!]$.
The log-likelihood function is given by $p(y = k|\mathbf{x}, \boldsymbol{\theta}) = k\log\lambda - \log k! - \lambda$,
where $\lambda = e^{\boldsymbol{\theta}^T\mathbf{x}}$.
So $p(y = k|\mathbf{x}, \boldsymbol{\theta}) = k\boldsymbol{\theta}^T\mathbf{x} - e^{\boldsymbol{\theta}^T\mathbf{x}} - \log k!$.
So $l_{\boldsymbol{\theta}} = \sum_{i=1}^N \left(y_i\boldsymbol{\theta}^T x_i - e^{\boldsymbol{\theta}^T x_i} - \log y_i!\right)$. □

(B) Given a test point $\mathbf{x}_\star$ and some estimate of the model parameter $\hat{\boldsymbol{\theta}}$, how do you make a prediction $\hat{y}_\star$?

Solution.

Given a test point $\mathbf{x}_\star$ and some estimate of the model parameter $\hat{\boldsymbol{\theta}}$, we can predict with maximum likelihood estimation, by choosing the class k that maximizes the log-likelihood function $p(y = k|\mathbf{x}_\star, \hat{\boldsymbol{\theta}})$.
In other words,

$$\hat{y}_\star = \underset{k\in\mathbb{N}}{\operatorname{argmax}}\, p(y = k|\mathbf{x}_\star, \hat{\boldsymbol{\theta}}) = [e^{\boldsymbol{\theta}^T\mathbf{x}}]$$

□

(C) Now suppose that the parameter $\hat{\boldsymbol{\theta}}$ of the Poisson GLM is estimated using ℓ_2-regularized maximum likelihood estimation. If the test point $\mathbf{x}_\star$ is *orthogonal* to the subspace generated by the training data, what is the distribution $\hat{y}_\star|\mathbf{x}_\star$ predicted by the Poisson GLM model? Prove your answer.

Solution.

Maximizing the ℓ_2-regularized likelihood is equivalent to find

$$\min_{\boldsymbol{\theta}} -l_{\boldsymbol{\theta}}(\mathcal{D}) + \gamma||\boldsymbol{\theta}||_2^2 = \min_{\boldsymbol{\theta}} \sum_{i=1}^N \left(e^{\boldsymbol{\theta}^T x_i} - y_i\boldsymbol{\theta}^T x_i + \log y_i!\right) + \gamma\boldsymbol{\theta}^T\boldsymbol{\theta}$$

We set its derivative w.r.t. $\boldsymbol{\theta}$ equal to $\mathbf{0}$, so

$$\sum_{i=1}^N \left(e^{\boldsymbol{\theta}^T x_i}x_i^T - y_i x_i^T\right) + 2\gamma\boldsymbol{\theta}^T = \mathbf{0}$$

Since the test point $\mathbf{x}_\star$ is orthogonal to the subspace generated by the training data, we have

$$\mathbf{0} = \mathbf{0}^T\mathbf{x}_\star = \sum_{i=1}^N \left(e^{\boldsymbol{\theta}^T x_i} - y_i\right)x_i^T\mathbf{x}_\star + 2\gamma\boldsymbol{\theta}^T\mathbf{x}_\star = 2\gamma\boldsymbol{\theta}^T\mathbf{x}_\star$$

Therefore

$$\boldsymbol{\theta}^T\mathbf{x}_\star = 0$$

The distribution will be

$$p(\hat{y}_\star = k|\mathbf{x}_\star) = \exp(k\boldsymbol{\theta}^T\mathbf{x}_\star - \boldsymbol{\theta}^T\mathbf{x}_\star - \log k!) = \exp(-1 - \log k!) = \frac{1^k e^{-1}}{k!}$$

which is a Poisson distribution with $\lambda = 1$. □

(D) From your answer to part (C), motivate ℓ_1-regularization when the number of features, D, is larger than the number of examples, N.

Solution.

When the number of features, D, is larger than the number of examples, N, a test point $\mathbf{x}_\star$ will likely to be orthogonal to the space generated by the training data. If so, according to my answer to part (C), the distribution $\hat{y}_\star|\mathbf{x}_\star$ predicted by the Poisson GLM model using ℓ_2-regularization will become independent to $\mathbf{x}_\star$. So the model will be very insensitive.
We can mitigate the problem by using ℓ_1-regularization instead. Maximizing the ℓ_1-regularized likelihood is equivalent to find

$$\min_{\boldsymbol{\theta}} -l_{\boldsymbol{\theta}}(\mathcal{D}) + \gamma||\boldsymbol{\theta}||_1 = \min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \left(e^{\boldsymbol{\theta}^T x_i} - y_i \boldsymbol{\theta}^T x_i + \log y_i!\right) + \gamma\boldsymbol{\theta}$$

Now given a test point $\mathbf{x}_\star$, we can avoid the problem of getting a constant distribution.
Moreover, ℓ_1-regularization generally gives more sparse results and helps in feature selection by eliminating the features that are not important. So it is ideal when the number of features, D, is larger than the number of examples, N. □