Assignment 2 - Logistic Regression & Naive Bayes

Due by 11:59pm, Feb 24, 2023

Theory Questions (Full points: 40, each question 4 points)

1. Explain the importance of setting up learning rate in the gradient descent based methods.

Answer:

Learning rate is a hyperparameter in the gradient descent methods.
It means we can tune this rate so as to observe the effect as how fast or slow the algorithm will converge to the minumum loss value.
In other words, it controls the size of the steps taken in each iteration of the algorithm towards the minimum of the loss function. This makes it very crucial to get the right value for it.
As explained in my HW1 report, if the learning rate is too small, the algorithm may take too long to converge to the minimum, while if it is too large, it may overshoot and never converge or converge to a suboptimal solution.
An optimum learning rate value is chosen via experimentation and tuning.

Reference:

https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de#:~:text=In%20order%20for%20Gradient%20Descent,will%20skip%20the%20optimal%20solutic
(https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de#:~:text=In%20order%20for%20Gradient%20Descent,will%20skip%20the%20optimal%20solutic

1. What is the stochastic gradient descent? Why do we need stochastic gradient descent?

Answer:

In a traditional Gradient Descent, the entire training dataset is used to calculate the gradient. This process can be computationally expensive and time consuming.
Hence, the idea of Stochastic Gradient Descent (aka. SGD) came into picture which increments/updates the weights for each training sample.
It is a probabilistic approach to the normal gradient descent which makes it even more helpful when the dataset is quite large.

The benefits of using SGD are:
1) Faster convergence
2) Computationally efficient
3) Avoids to get stuck in local minima and tries to reach global minima.

Reference:

https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/ (https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/)

1. Explain the reasons to perform feature scaling.

Answer:

Feature scaling is an important process in data preprocessing. It helps in transforming the input features values to have similar scales. The reason to perform this is that the dataset can contain features which vary in scale and due to this the model's performance can be undesirable. Consider the example of dealing with real estate data where we have number of rooms as a feature and area of the house as another feature. While one will be a small number, the other will be a very high number compared to the first one. The units of measurement are created for human understanding, machine learning models may not understand those. Hence, for a machine learning algorithm to work efficiently, we should perform feature scaling. Another reason can be it will help in making the importance of a feature more evident.

1. What is the probabilistic generative model?

Answer:

Probabilistic generative models are statistical models used in machine learning to model the underlying distribution of the data. In the context of classification, probabilistic generative models are used to model the probability distributions of the input features for each class, and then use Bayes' rule to compute the probability of each class given a new input.

For example, suppose we have a dataset of images of handwritten digits, and we want to classify these digits. A probabilistic generative model would estimate the probability of each pixel being on as 1 or off as 0 for each class, and then use this probability to classify new image to its appropriate class.

Generative models have the advantage of being able to generate new data similar to the training data, and can be useful for tasks such as data synthesis and anomaly detection.

Reference:

https://medium.com/@jordi299/about-generative-and-discriminative-models-d8958b67ad32 (https://medium.com/@jordi299/about-generative-and-discriminative-models-d8958b67ad32)

1. Explain how we perform maximum likelihood.

Answer:

Maximum likelihood is a technique which is utilized to get an estimation of the parameters of a model by maximizing the likelihood function, which is the probability of observing the data given the parameters.

To perform maximum likelihood, the very first step is to define the likelihood function based on parameters we want to estimate. Once the likelihood function is defined, the next step is to find the values of the parameters that maximize it using an optimization algorithm such as gradient descent algorithm.

The maximum likelihood method is widely used in statistics and machine learning to estimate the parameters of models such as linear regression, logistic regression, and neural networks.

Reference:

https://machinelearningmastery.com/what-is-maximum-likelihood-estimation-in-machine-learning/ (https://machinelearningmastery.com/what-is-maximum-likelihood-estimation-in-machine-learning/)

1. Explain the reasons about using cross entropy loss in logistic regression.

Answer:

In logistic regression, the aim is to model the probability of an input belonging to a particular class. The predicted probabilities are then compared with the actual class labels to compute the loss during training. One of the most commonly used loss function in logistic regression is the cross entropy loss.

The cross entropy loss measures the dissimilarity between the predicted probability distribution and the true probability distribution of the classes. Cross-entropy loss is preferred for logistic regression because it puts more emphasis on the model's ability to correctly classify data with high confidence.

The formula for the cross-entropy loss is:

$H(p, q) = -\sum_{i=1}^{n} p(x_i) \log q(x_i)$

where $p(x_i)$ is the actual probability distribution and $q(x_i)$ is the predicted probability distribution.

The cross-entropy loss evaluates how closely the model's predicted distribution matches the actual distribution. In logistic regression, gradient descent can be employed to minimize this loss and once it is minimized, the model can better predict the correct classification of labels.

Reference:

https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e#:~:text=Cross%2Dentropy%20loss%20is%20used,cross%2Dentropy%20loss%20of%200 (https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e#:~:text=Cross%2Dentropy%20loss%20is%20used,cross%2Dentropy%20loss%20of%200)

1. Explain the differences between discriminative and generative model.

Answer:

Discriminative Models: - Discriminative models cannot handle incomplete data and require a dataset with no missing or faulty data. Discriminative models are generally more accurate than generative models, especially when the dataset is complete. Discriminative models are designed for discrimination tasks and are less computationally expensive than generative models. More suitable when the focus is on accuracy of classification.

Generative Models: - Generative models can handle datasets with missing values and also work on unlabeled data. However, they are less efficient and less accurate than discriminative models. Generative models are used for Naive Bayes, Bayesian learning, and Hidden Markov Models. They are more computationally expensive compared to discriminative models. More suitable when the focus is on understanding the probability distribution of the data.

Reference:

https://medium.com/@mlengineer/generative-and-discriminative-models-af5637a66a3 (https://medium.com/@mlengineer/generative-and-discriminative-models-af5637a66a3)

1. What is N-folds? Explain the reasons why we need N-folds.

Answer:

N-folds is a cross-validation technique used in machine learning to evaluate the performance of a model on a limited dataset. The basic idea behind N-folds is to split the dataset into N equal parts, called folds. The model is then trained N times, each time using a different fold as the validation set and the remaining folds as the training set.

The reason for using N-folds is to get a more accurate estimate of the model's performance by reducing the bias in the evaluation process. By using multiple folds, we can get a better sense of how well the model generalizes to new, unseen data, and can also detect any issues such as overfitting.

N-folds also helps to make efficient use of limited data by allowing us to train and evaluate the model on all available data, while also avoiding the risk of overfitting to a specific subset of data. It is a widely used technique in machine learning and is essential for developing accurate and robust models.

Reference:

https://www.quora.com/What-exactly-is-a-fold-in-machine-learning-For-example-what-is-a-fold-in-K-fold-cross-validation (https://www.quora.com/What-exactly-is-a-fold-in-machine-learning-For-example-what-is-a-fold-in-K-fold-cross-validation) https://machinelearningmastery.com/k-fold-cross-validation/ (https://machinelearningmastery.com/k-fold-cross-validation/)

1. Bishop's Book "Pattern Recognition and Machine learning" - Exercise 4.12

Answer:

$\frac{d\sigma}{da} = \sigma(1-\sigma)$ (Equation 4.88 in Bishop) [sigmoid function differentiation]

$\sigma(a) = \frac{1}{1+e^{-a}}$ (Equation 4.59 in Bishop) [sigmoid function]

To get our results, we have to differentiate (Equation 4.59) as following:

$\frac{d\sigma}{da} = \frac{e^{-a}}{(1+e^{-a})^2}$ [exponent differentiation and power rule differentiation]

$= \sigma(a)[\frac{e^{-a}}{1+e^{-a}}]$

$= \sigma(a)[\frac{1+e^{-a}}{1+e^{-a}} - \frac{1}{1+e^{-a}}]$

$= \sigma(a)(1-\sigma(a))$

1. Bishop's Book "Pattern Recognition and Machine learning" - Exercise 4.13

Answer:

$\frac{d\sigma}{da} = \sigma(1-\sigma)$ (Equation 4.88 in Bishop)

$E(w) = -lnp(t|w) = -\sum_{n=1}^{N}[t_n ln y_n + (1-t_n)ln(1-y_n)]$ (Equation 4.90 in Bishop)

$\nabla E(w) = \sum_{n=1}^{N}(y_n - t_n)\phi_n$ (Equation 4.91 in Bishop)

Computing the derivative of (Equation 4.90) with respect to y_n as follows:

$\frac{\partial E}{\partial y_n} = \frac{1-t_n}{1-y_n} - \frac{t_n}{y_n}$

$= \frac{y_n(1-t_n)-t_n(1-y_n)}{y_n(1-y_n)}$

$= \frac{y_n - y_n t_n - t_n + y_n t_n}{y_n(1-y_n)}$

$= \frac{y_n - t_n}{y_n(1-y_n)}$