CSE 847 Mid Term

Submitted by: Ritam Guha (MSU ID: guharita)

Date: March 15, 2021

---

1. You have a new box containing 8 apples and 4 oranges and an old box containing 10 apples and 2 oranges. You select a box at random with equal probability, select an item out of that box at random with equal probability, and find it is an apple. Using Bayes' theorem to find the probability that the apple came from the old box? (10 points).

   Response:

   The fruit distribution between the old and the new box is shown below:

Box Type	Probability of Selection	Apples	Oranges
New	0.5	8	4
Old	0.5	10	2

   The probability of selecting one of the boxes is 0.5 because it is selected with an equal probability. We need to find the probability denoted by: $P(b = Old|f = Apple)$

   According to Bayes' Theorem,

   $$P(b = Old|f = Apple) = \frac{P(f = Apple|b = Old)P(b = Old)}{P(f = Apple)}$$

   The required probability values are computed below:

   $P(b = Old) = 0.5$

   $P(f = Apple|b = Old) = \frac{10}{12}$

$$
\begin{aligned}
P(f = Apple) &= P(f = Apple|b = Old)P(Old) + P(f = Apple|b = New)P(New) \\
&= \frac{10}{12} \times 0.5 + \frac{8}{12} \times 0.5 \\
&= \frac{5}{12} + \frac{4}{12} \\
&= \frac{9}{12}
\end{aligned}
$$

$$
P(b = Old|f = Apple) = \frac{\frac{10}{12} \times 0.5}{\frac{9}{12}} = \frac{\frac{5}{12}}{\frac{9}{12}} = \frac{5}{9} = 0.56 \text{ (approx.)}
$$

The probability that the apple came from of the old box is: $\frac{5}{9}$ or 0.56 (approx.) **(Ans)**

---

2. Describe in details the three distinct approaches (probabilistic generative models, probabilistic discriminative models, and discriminate function) for modeling. What is the advantage of using discriminative models, as compared to generative models? (10 points).

Response:

The classification problem can be divided into two distinct steps: **Inference** and **Decision**. Inference stage is used to learn probability models from the training data while decision stage is used to take the final decision on a new sample based on the computed probability values. There are three different approaches which can be used in the inference stage. In the following section, each of the three apporaches are described in detail:

Probabilistic Generative Model:

In order to model the posterior probability of the class label given a new sample $P(C_k|x)$, this kind of approach computes the class conditioned density $P(x|C_k)$ for each class and the prior class probabilities $P(C_k)$ of the classes separately. Then Bayes' Theorem is used to compute the posterior probability as follows:

$$P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)}$$

where $P(x)$ is the normalization factor which can be computed as: $P(x) = \sum_k P(x|C_k)P(C_k)$. The numerator can also be written as: $P(x, C_k)$ which is the joint probability of the sample and the classes. Thus the posterior proability can also be computed using the joint probabilities followed by normalization. After the probability caluclations are done, a decision about sample x is taken based on the probability values.

From the procedure of computing the posterior probability we can see that $P(x|C_k)$ is computed as an intermediary step. So, generative models can be used to sample synthetic data points for different classes from the conditional probability distribution which is a pretty useful feature. But due to so much computation, generative models are typically slower.

Example: Naive Bayes classifier.

Probabilistic Discriminative Model:

These models try to directly determine the posterioir probability of the class label given the new sample $P(C_k|x)$. Then a decision is taken for the sample based on this probability distribution in the decision stage.

Example: Logistic Regression.

Discriminate Function:

Instead of using any probability flavoured inference step, it tries to find a function f that directly maps the input sample to one of the class labels. A simple example maybe for a binary classification problem if $f(x) > 0$, we may assign x to class C_1, else C_2. These approaches try to model a function from the training data and then the functional values of the samples are used to make decisions. So, here both the inference and decision stages are combined into a single learning problem.

Example: Support Vector Machines, Decision Tree.

Advantage of using Discriminative model over Generative model:

In Generative models, to be able to compute the posterior probability, we need to model the class conditioned densities and prior probabilities. In order to model the class conditioned densities to some reasonable accruacy, it needs a lot of training data which is not available always. Computing so many distributions makes it quite demanding in terms of processing. Moreover, if the only goal of the task is to make classification decisions, then i**t leads to wastage of computational resources and demands too much data**.

On the other hand, Discriminative models directly models the posterior probability which makes i**t computationally less expensive and ideal when only classification decisions are required**.

---

3. Consider two data sets D_1 and D_2 , each consisting of scalar measurements x_i , $i = 1, \ldots, N_1$ for D_1 and x_j , $j = 1, \ldots, N_2$ for D_2 . Assume that each set of measurements comes from a Gaussian distribution. The two Gaussian distributions share a common variance σ^2 and the mean μ_2 of the Gaussian for data set D_2 is known to be twice the value of the mean μ_1 for the first data set D_1. μ_1, μ_2, and σ^2 are all assumed unknown.

   (a) Define the log-likelihood for this problem. (10 points)

   (b) Derive the maximum likelihood estimators for the unknown parameters. (10 points).

   Response:

   (a) In order to find the log-likelihood for the two data, we need to find $P(D_1)$ and $P(D_2)$.

$$
\begin{aligned}
P(D_1) &= \prod_{i=1}^{N_1} P(x_i) \text{[Assuming the data points to be independent and identically distributed (i.i.d)]} \\
&= \prod_{i=1}^{N_1} \mathcal{N}(x_i|\mu_1, \sigma^2) \text{[As they come from Gaussian distribution]} \\
&= \prod_{i=1}^{N_1} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x_i-\mu_1)^2}
\end{aligned}
$$

Taking ln on both sides:

$$
\begin{aligned}
\ln P(D_1) &= \sum_{i=1}^{N_1} [\ln(\frac{1}{\sqrt{2\pi\sigma^2}}) + \ln(e^{-\frac{1}{2\sigma^2}(x_i-\mu_1)^2})] \\
&= \sum_{i=1}^{N_1} [\ln((2\pi\sigma^2)^{-\frac{1}{2}}) - \frac{1}{2\sigma^2}(x_i-\mu_1)^2] \\
&= \sum_{i=1}^{N_1} [-\frac{1}{2}(\ln(2\pi\sigma^2)) - \frac{1}{2\sigma^2}(x_i-\mu_1)^2] \\
&= \sum_{i=1}^{N_1} [-\frac{1}{2}\ln(2\pi) - \frac{1}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}(x_i-\mu_1)^2] \\
&= -\frac{N_1}{2}\ln(2\pi) - \frac{N_1}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{N_1}(x_i-\mu_1)^2
\end{aligned}
$$

In a similar way, we can get $\ln P(D_2)$ as:

$$
\ln P(D_2) = -\frac{N_2}{2}\ln(2\pi) - \frac{N_2}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=1}^{N_2}(x_j-\mu_2)^2
$$

The log-likelihood of the problem is given by: $\ln P(D_1, D_2)$. Assuming D_1 and D_2 to be i.i.ds we get:

$$\ln P(D_1, D_2) = \ln P(D_1)P(D_2)$$
$$= \ln P(D_1) + \ln P(D_2)$$
$$= -\frac{N_1 + N_2}{2}\ln(2\pi) - \frac{N_1 + N_2}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}(\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - \mu_2)^2)$$

According to the question, $\mu_2 = 2\mu_1$

So, the final log-likelihood expression becomes:

$$\ln P(D_1, D_2) = -\frac{N_1 + N_2}{2}\ln(2\pi) - \frac{N_1 + N_2}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}(\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1)^2)$$

OR

$$\ln P(D_1, D_2) = -\frac{N_1 + N_2}{2}\ln(2\pi) - \frac{N_1 + N_2}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}(\sum_{i=1}^{N_1}(x_i - \frac{\mu_2}{2})^2 + \sum_{j=1}^{N_2}(x_j - \mu_2)^2)$$

(b) As this is a convex function, the maximum likelihood estimation can be simply derived by setting the derivatives of the function with respect to the unknown parameters as 0.

Setting the derivative w.r.t μ_1 equal to 0:

$$\frac{\partial \ln P(D_1, D_2)}{\partial \mu_1} = 0$$
$$\Rightarrow -\frac{1}{2\sigma^2}[\sum_{i=1}^{N_1} 2(x_i - \mu_1)(-1) + \sum_{j=1}^{N_2} 2(x_j - 2\mu_1)(-2)] = 0$$
$$\Rightarrow \sum_{i=1}^{N_1}(\mu_1 - x_i) + \sum_{j=1}^{N_2}(4\mu_1 - 2x_j) = 0$$
$$\Rightarrow N_1\mu_1 - \sum_{i=1}^{N_1} x_i + 4N_2\mu_1 - 2\sum_{j=1}^{N_2} x_j = 0$$
$$\Rightarrow (N_1 + 4N_2)\mu_1 = \sum_{i=1}^{N_1} x_i + 2\sum_{j=1}^{N_2} x_j$$
$$\Rightarrow \mu_1 = \frac{\sum_{i=1}^{N_1} x_i + 2\sum_{j=1}^{N_2} x_j}{N_1 + 4N_2}$$

$$\mu_1(ML) = \frac{\sum_{i=1}^{N_1} x_i + 2\sum_{j=1}^{N_2} x_j}{N_1 + 4N_2}$$

Setting the derivative w.r.t μ_2 equal to 0:

$$\frac{\partial \ln P(D_1, D_2)}{\partial \mu_2} = 0$$
$$-\frac{1}{2\sigma^2}[\sum_{i=1}^{N_1} 2(x_i - \frac{\mu_2}{2})(-\frac{1}{2}) + \sum_{j=1}^{N_2} 2(x_j - \mu_2)(-1)] = 0$$
$$\Rightarrow \sum_{i=1}^{N_1}(\frac{\mu_2}{2} - x_i) + \sum_{j=1}^{N_2}(2\mu_2 - 2x_j) = 0$$
$$\Rightarrow N_1\frac{\mu_2}{2} - \sum_{i=1}^{N_1} x_i + 2N_2\mu_2 - 2\sum_{j=1}^{N_2} x_j = 0$$
$$\Rightarrow (\frac{N_1}{2} + 2N_2)\mu_2 = \sum_{i=1}^{N_1} x_i + 2\sum_{j=1}^{N_2} x_j$$
$$\Rightarrow (N_1 + 4N_2)\mu_2 = 2\sum_{i=1}^{N_1} x_i + 4\sum_{j=1}^{N_2} x_j$$
$$\Rightarrow \mu_2 = \frac{2\sum_{i=1}^{N_1} x_i + 4\sum_{j=1}^{N_2} x_j}{N_1 + 4N_2}$$

$$\mu_2(ML) = \frac{2\sum_{i=1}^{N_1} x_i + 4\sum_{j=1}^{N_2} x_j}{N_1 + 4N_2} = 2\mu_1(ML)$$

Setting the derivative w.r.t σ^2 equal to 0:

$$\frac{\partial \ln P(D_1, D_2)}{\partial \sigma^2} = 0$$
$$\Rightarrow -\frac{N_1 + N_2}{2}\frac{1}{\sigma^2} - \frac{1}{2}(\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1)^2)(-\frac{1}{\sigma^4}) = 0$$
$$\Rightarrow -\frac{N_1 + N_2}{2}\sigma^2 + \frac{1}{2}(\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1)^2) = 0$$

$$\Rightarrow (N_1 + N_2)\sigma^2 = (\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1)^2)$$

$$\Rightarrow \sigma^2 = \frac{(\sum_{i=1}^{N_1}(x_i - \mu_1)^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1)^2)}{N_1 + N_2}$$

‘

$$\sigma^2(ML) = \frac{(\sum_{i=1}^{N_1}(x_i - \mu_1(ML))^2 + \sum_{j=1}^{N_2}(x_j - 2\mu_1(ML))^2)}{N_1 + N_2}$$

Thus the maximum likelihood estimators are derived.

---

4. Recall that in linear regression the objective function is

   $E(w) = \frac{1}{2}||Xw - t||_2^2$

   and its first-order derivative is $\nabla E(w) = X^T Xw - X^T t$. It is well-known that one condition for a function to be convex is its second-order derivative is positive semi-definite. Please

   (a) Compute the second-order derivative $\nabla^2 E(w)$ (i.e., the derivative of $\nabla E(w)$). (10 points)

   (b) Show that the second-order derivative is positive semi-definite. (10 points)

   Response:

   (a) The first order derivative is given by:

   $\nabla E(w)$

   $= X^T Xw - X^T t$

   The second order derivative is computed below:

   $\nabla_w^2 E(w)$

   $= \nabla_w \nabla_w E(w)$

   $= \nabla_w (X^T Xw - X^T t)$

   $= \nabla_w (X^T Xw) - \nabla_w (X^T t)$

   $= (X^T X)^T - 0$ [As $\frac{\partial A^T x}{\partial x} = A$]

   $= X^T X$

So, the second order derivative is: $X^T X$

(b) A matrix $A \in \mathbb{R}^{n \times n}$ is said to be a positive semi-definite (PSD) matrix if for any non-zero vector $v \in \mathbb{R}^n$, the following condition holds:

$v^T A v \geq 0$

If we replace A with the second order derivative, we get:

$v^T X^T X v$

$= (Xv)^T (Xv)$

$= ||Xv||_2^2$

Xv can be treated as the linear combination of the columns of X. The multiplication will give us a vector. So, $||Xv||_2$ is the l_2 norm of a vector which is always ≥ 0 and the square of that is also greater than or equal to 0. $v^T X^T X v \geq 0$. Thus from the definition of a PSD matrix, The second order derivative is a PSD matrix.

---

5. Generalization is one of the most important questions in machine learning.

   (a) What is generalization performance? (10 points)

   (b) How does generalization performance relate to regularization and model complexity?(10 points).

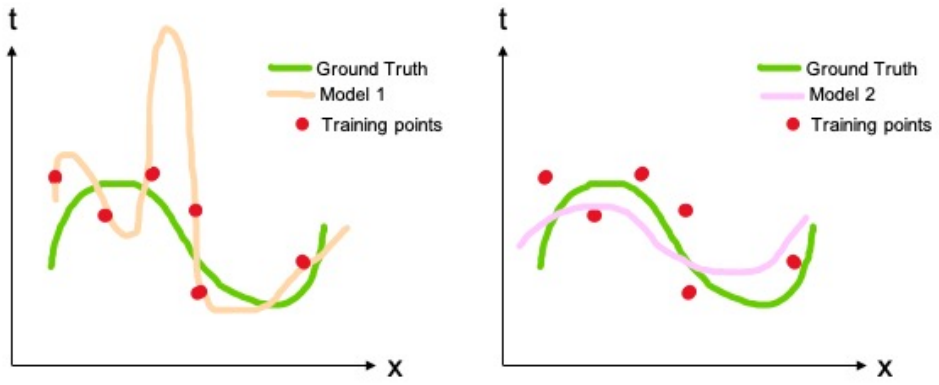
Response:

(a) Generalization for a machine learning model generally refers to how easily it can get adapted to unseen data. Unseen data means the data which was not exposed to the model during the training phase. The goal of generalization is to get as close to the ground truth as possible.

A model showing great performace on training data may not necessarily mean it is generalized or it will show the same level of preformance in testing data. When a model gets overfitted to the training data, it gives excellent results for the training data but it fails to capture the generalized trends of the data. So, it may not provide good outcomes for test or unseen data.

A generalized model may not give best performance for the training data but if we consider training data and unseen data combinedly, it will provide much better overall performance.

The following figure makes it easier to understand.:



The Model 1 in the figure gets overfitted to the training points, so it gives great results for the training samples but it fails to capture the trend in the data and it is nowhere close to the ground truth structure. On the other hand, Model 2 does not get overfitted to the training data. It starts following the ground truth. Although it does not give as good results as Model 1 on training data, it gives better result if we consider unseen data from the distribution of the ground truth. So, Model 2 is more generalized than Model 1.

(b) Generalization is a difficult task to achieve for a model. If we have a complex model and too many degrees of freedom, the model can easily get overfitted to the training data which is not desired and it goes away from generalization. So, as model complexity increases, the generalization improves till some extent but after that the model gets overfitted and starts losing generalization. The overfitting issue can be spotted from the huge values of the adjustable parameters. Regularization is a way to avoid the unwanted overfitting.

Regularization adds a function of the adjustable parameters to the loss function which encourages the values to reduce in order to avoid overfitting. A general loss function for regularized square loss formulation can be repsrented as:

$$min_w \frac{1}{2}||\Phi w - t||^2 + \frac{\lambda}{2}||w||_q^q$$

The addition of the term $\frac{\lambda}{2}||w||_q^q$ influences the values of the adjustable parameter (here w) to reduce which in turn reduces the possibility of the model getting overfitted. Thus using regularization we can use complex models to fit properly on training data without overfitting which leads to generalization. So, we can say that increasing model complexity reduces the possibility of generalization after some extent, but regularization encourages generalization by reducing the values of the adjustable parameters.

---

6. When dealing with overfitting, we have introduced various regularization approaches (e.g., l_1-norm regularization, l_2-norm regularization, etc.) Especially, we discussed in class that l_1-norm can induce sparsity in the solution

   $min_w f(w) + \lambda||w||_1$

   (a) Please explain why sparsity is useful in linear models? (10 points)

   (b) Recall that we can transform a regularized problem as a constrained problem. Please use the constrained problem:

   $min_w f(w)$, s.t.$||w||_1 \leq z$

   to illustrate why do we have sparsity using l_1-norm? (10 points)

Response:

(a) Sparsity basically refers to the number of zeros present in the model weights. It is particularly useful in linear models for the following reasons:

- **Dimension Reduction:** From sparsity, we can find the features which has no influence in the output of the model. So, it is a way to identify the set of irrelevant and redundant features in the design matrix. By removing the redundant features from the design matrix, the processing speed of linear models improves which is a huge advantage. Thus sparsity helps to perform dimension reduction.
- **Noise Removal:** Using the concept of sparsity, the more noisy portions of the data

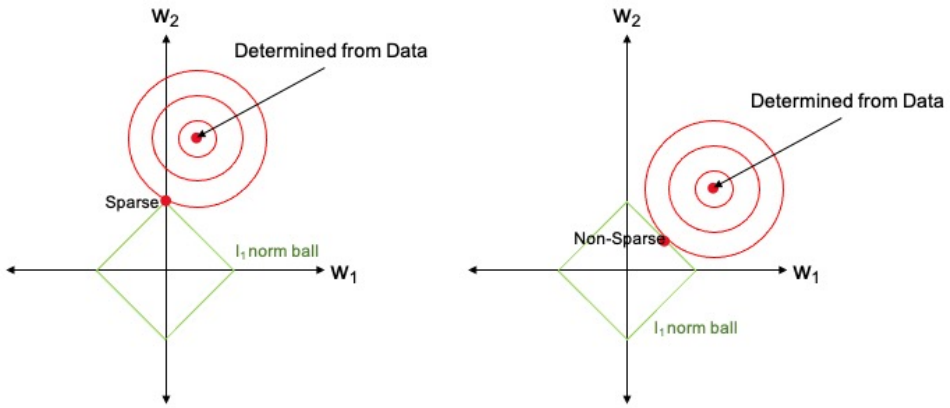
can be removed. Dense models can easily get overfitted to the noise present in the data. But sparse models use less noisy portions of the data to formulate the underlying model. This makes them susceptible to noisy data.

- **Increased Interpretability:** Sparse models are more interpretable than dense models. As the number of features grow, it becomes harder and harder to reason how the output of the model is related to all the inputs. But sparse models use smaller number of features in the data to make the decision which makes it more interpretable to the machine learning practioners.

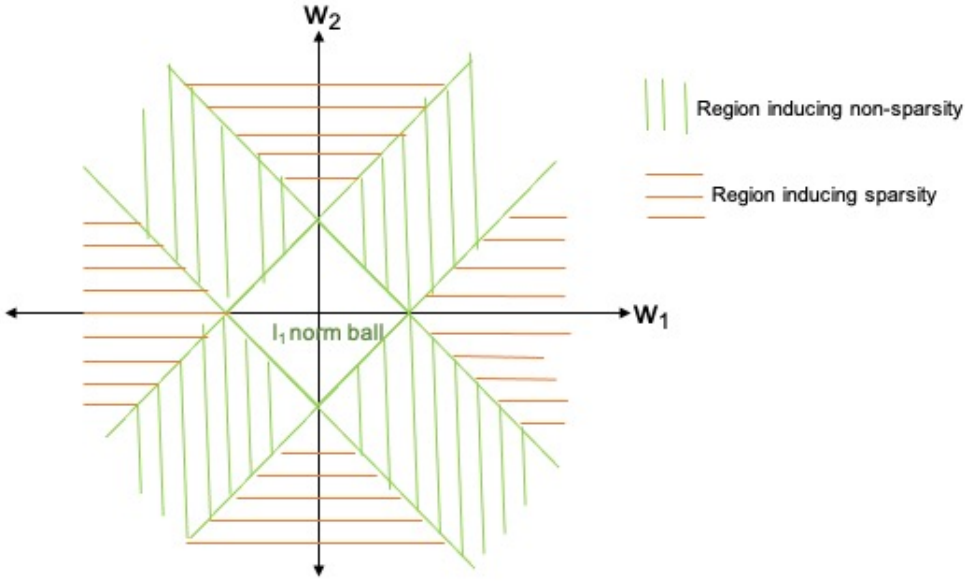
(b) The l_1 regularized version of the loss function of linear regression is represented as:

$$min_w f(w) + \lambda ||w||_1$$

The same problem can be represented as a constrained problem with the following formulation:
$min_w f(w)$ s.t. $||w||_1 \leq z$ where z defines the boundary of the feasible space. The unregularized version of the loss function gives us the value of w as: $w = \Phi(\Phi^T\Phi)^{-1}\Phi^T t$
After careful observation, we can see that the most optimal solution of w is produced by the design matrix or the data itself. But it leads to overfitting on the training data most of time. In order to get rid of that problem, regularization is introduced. So, now we are not selecting the most optimal w in the unbounded search space. In fact, we are restricting the w with a constraint i.e. $||w||_1 \leq z$. If we draw the contour of the loss function in 2D, we will get the following figure:



The weights corresponding to the minimum value of the loss function is derived directly from the data and it lies on the red points in the images. The l_1 norm ball represnts the constrained boundary of the problem. The feasible weights lie inside the norm ball. So, in order to get the most optimal feasible model weights, contours of the loss function are drawn centering the red points. The contours intersect the norm balls in some point and the corresponding model weights then become the solution to the constrained optimization problem. From the images we can see that depending on the most optimal weights (or the solution of the unconstrained counterpart), the solution maybe sparse or non-sparse. In the left scenario, the optimal model is non-sparse while for the right scenario, it is sparse. From careful observation, we can arrive at the following figure:



In the Figure, we can see if the solution to the unconstrained optimization problem lies in the yellow shaded region, it induces sparsity in the model and if it falls in the green shaded region, it leads to non-sparsity or dense models. So, the level of sparsity in the graph can be quantified by: $\frac{\text{Area of yellow shaded region}}{\mathbb{R}^2}$

Thus we can see the situations when l_1 norm may lead to sparsity based on the equivalent unconstrained optimization problem.