Data Mining, Synthetic Data, Knowledge Discovery: Homework Report

Group: Akinola Lanre Olusegun, Boiardi Alessio, Baiguini Silvio

February 4, 2025

The purpose of this paper is to present the solutions to the questions assigned as homework to partial completion of the grade in the course Data Mining and Knowledge Discovery. Each section will be dedicated to answering the corresponding question, with each subsection dedicated to a different sub-question. The concise nature of the answers is supposed to reflect the requirement of "a concise report", as expressively requested in the homework assignment.

1 Technical terms definition

1.1 Maximum Likelihood

In the context of neural networks training, refers, in general, to the value that better models the likelihood of training data. Through a process called Maximum Likelihood Estimation, it is possible to find a value for the weights W such that the output value $f(X)$ of the network is the one that has the highest probability of matching the ground truth of the training data. Such probability is called the Maximum Likelihood. The estimation process is concerned with the following optimization objective:

$$\hat{\theta} = \arg\max_{\theta} F(y; \theta)$$

1.2 Maximum a Posteriori

Similarly to Maximum Likelihood Estimation, Maximum a Posteriori is an estimation process. It could be considered a step up from Maximum Likelihood, since not only does it try to maximize the probability that $f(X) = X^T W$ is equal to the training data, but also keeps into account the observed probability w_0 in $f(X) = X^T W + W_0$. Despite being generally considered more powerful than MLE, it can only be applied when certain assumptions about the weights can be made. Extending the Maximum Likelihood Estimation formula with the A Priori distribution G, the optimization objective of MAP becomes:

$$\hat{\theta} = \arg\max_{\theta} F(y; \theta) G(\theta)$$

1.3 Regularization

Regularization is a technique used in training that works by adding a penalty term to the "objective function". A penalty term generally means that large coefficients will be discouraged, which results in less complex models that are more tolerant to noise. This translates to a better resilience of training to overfitting, and makes regularization such as L2 norm (Ridge regression) an effective tool against it.

1.4 Examples

- Maximum Likelihood: let's consider a problem of classic linear regression. In this problem, a linear relation between the input values and the output values is assumed. This means that the neural network tries to predict $f(X) = X^T W$ where X is the matrix of input values and W is the matrix of weights. Let's assume that y_i (the ground truths for all datapoints) is Gaussian distributed. Its *pdf* is:

$$\frac{1}{\sqrt{2\pi\sigma^2}} exp\bigg(- \frac{(y_i - y)^2}{2\sigma^2} \bigg)$$

  Maximum Likelihood Estimation is concerned with maximizing the probability of the entire dataset. In this case, the log-likelihood:

$$logL(D, h) = -\frac{n}{2} ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - y)^2$$

- Maximum a Posteriori: let's take the same example of linear regression from the previous problem. This time, instead of the previous probability for the dataset $P(D|h)$, the a posteriori term is added, that represents prior beliefs about the network:

$$P(D|h)P(h)$$

  Assume that $P(h)$ is also Gaussian distributed with: $P(h) \sim \mathcal{N}(0, \tau^2)$. Adding the new probability to the equation of the log-likelihood in the previous example, the target function to maximize becomes then:

$$logL(D, h) = -\frac{n}{2} ln(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - y)^2 - \frac{1}{2\tau^2} ||w||^2$$

  The added term also acts as a regularizer, which is why it is said that Maximum A Posteriori aims to maximize the regularized log-likelihood of a loss function.

- Regularization: consider this time the sum of squared errors:

$$L(\mathbf{w}) = \sum_{i=0}^{n} |\mathbf{x}_i^T \mathbf{w} - y_i|^2$$

In the Ridge regression, using the L2-norm, the following regularizer term is added:

$$L(\mathbf{w}) + \Omega_2(\mathbf{w}) = \sum_{i=0}^{n} |\mathbf{x}_i^T \mathbf{w} - y_i|^2 + \lambda \sum_{j=1}^{m} |w_j|^2$$

The purpose of the chosen term Ω_2 is to increase the results of the regularized loss function so that big weights are discouraged. It effectively makes weights smaller, thus helping to prevent overfitting.

2 The kernel trick

The assignment requires to describe the reasons behind the kernel trick and the nature of the formula it comes from:

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) \tag{1}$$

In some classification problems, data might be difficult to divide in the original dimensions, so it is very common to pick a linear transformation φ that maps the features from the input space to a space of higher dimensions where the features are (hopefully) linearly separable. However, such a transformation might become unfeasible to compute when the dimensions of K are very large. The alternative is to build the kernel matrix by picking a *kernel function*, which job is compute similarity between features. The eq. (1) shows this *kernel trick* technique, where K_{ij}, instead of being computed from the dot product of the linear transformations of x_i and x_j, is computed by applying the kernel function k to their respective pairs of features.

3 CIFAR Exercise

We decided to dedicate a portion of the report to still talk about the CIFAR exercise. However, in order to keep the report concise and not too verbose, this part is rather brief.

3.1 Data Preparation

This part was straightforward: following the directions on the assignment, the data was loaded, normalized and flattened. Then, 80% of the training data was dedicated to training and the rest to validation.

3.2 Problem Analysis

A quick look at the dataset, including visualization of samples, revealed that the classes look well-balanced, with exactly 5.000 samples per class. The problem at hand is undoubtedly a multi-class classification task, for which accuracy was chosen to be the target.

3.3 Model Selection and Evaluation

The three models chosen for this task were SVM, Gaussian Naïve Bayes and ANN. As a bonus test, it was also decided to add both the linear kerneled Support Vector Machine and the Gaussian kerneled Support Vector Machine. The results of the experiment are:

Models	Accuracy (%)
ANN	44.48
Linear SVM	34.46
RBF SVM	49.45
Gaussian Naïve Bayes	29.39

These are the results of the most successful experiment. As can clearly be seen, the best model was the Gaussian kernel SVM, followed by the feed-forward model, then by the linear kernel SVM, with the Gaussian Naïve in last place. The training time for these models are:

Models	Training time (s)
ANN	X
Linear SVM	1184
RBF SVM	1066
Gaussian Naïve Bayes	3

The Naïve Bayes could be trained in just a handful of seconds, while the two SVM took slightly more than a quarter of an hour to finish. The feed-forward model took several hours, the reason being that a very high number of epochs was used. Arguably, an early-stopping mechanism could have been used, since the validation accuracy and loss displayed the tendency to plateau before the first 10 epochs, but due to the general lack of improvement and the overall well-established results, the group decided to move on and focus the efforts elsewhere. Nonetheless, the feed-forward still proved to be the slowest to train, even with a reduced number of epochs. No significant differences in how models predict categories was found in the interpretation of results.

3.4 Cross Validation

As per homework assignment, stratified K-fold cross validation was used. Here are the results for k=10:

Models	Mean accuracy (%)	Standard deviation (%)
ANN	41.83	1.25
Linear SVM	33.49	1.15
RBF SVM	48.83	1.51
Gaussian Naïve Bayes	28.00	0.84

The results are in line with the test accuracy, as the mean roughly reflects the outcome of the best experiment reported in the previous section. The most consistent across folds was definitely the Naïve Bayes, as can be seen from the relatively low standard deviation. Typically, cross-validation is more representative of a model's performance than a train/test split, because it takes into account the average of multiple experiments, ensuring that the results are as free as possible from noise (such as a coincidentally favorable partition of the test set).

4 Primal and dual problem of the Lagrange function

Starting from the following equation:

$$L(w, b, \alpha) = 1/2||\mathbf{w}||^2 - \sum \alpha_i[y_i(\mathbf{w} \cdot x_i + b) - 1] \tag{2}$$

the task is to find the solution for the primal and the dual problem. The primal problem is easy to identify: in the context of optimization, it simply refers to the task of finding the minimum of the loss function. In the case of this equation, it means trying to minimize the term $1/2||\mathbf{w}||^2$. Since in SVM we would like to have each datapoint be classified with a margin of at least 1, ensuring a certain degree of confidence, it means that there exists also an implicit constraint (enforced by the penalty term multiplied with the Lagrangian α) that $y_i(\mathbf{w} \cdot x_i + b) \geq 1 \; \forall \; i$. This leads to an interesting consequence: it is possible to formulate a derived problem, called the dual problem, by rewriting the constraints using properties of functions optimization. In this case, the derivative of the Lagrange loss function with respect to both weights and biases should be 0 in the optimal point.

$$\frac{\delta L}{\delta w} = w - \sum_{i=1}^{n} \alpha_i x_i y_i = 0 \tag{3}$$

$$\frac{\delta L}{\delta b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{4}$$

From the eq. (3) follows:

$$w = \sum_{i=1}^{n} \alpha_i x_i y_i \tag{5}$$

while from eq. (4):

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{6}$$

The newly obtained w in the squared norm of weights can be substituted in the eq. (2). The first term becomes:

$$\frac{1}{2}||w||^2 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \tag{7}$$

while the second term (the penalty) can be expanded:

$$\sum_{i=1}^{n} \alpha_i[y_i(\mathbf{w} \cdot x_i + b) - 1] = \sum_{i=1}^{n} \alpha_i y_i(w_i x_i) + \sum_{i=1}^{n} \alpha_i y_i b + \sum_{i=1}^{n} \alpha_i(-1) \quad (8)$$

From eq. (4), the second Sum is equal to 0. Substituting w, again from eq. (3), the result becomes:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{n} \alpha_i \quad (9)$$

Now substituting eq. (7) and eq. (9) into eq. (2):

$$L'(\alpha) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - (\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{n} \alpha_i) \quad (10)$$

Simplifying:

$$L'(\alpha) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) + \sum_{i=1}^{n} \alpha_i \quad (11)$$

This last expression denotes a new loss function, one with only parameter α, and is much simpler to maximize due to its limited number of parameters and their scalar nature.

5 Relations between concepts

5.1 Tanh and Sigmoid

Equation of the tanh:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (12)$$

Equation of the sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (13)$$

Despite looking slightly different, it is possible to identify a relation between these two functions by expressing tanh in terms of sigmoid. Let's first simplify eq. (12) by dividing each term by e^x:

$$tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad (14)$$

eq. (13) can also be rewritten to match the denominator of eq. (14):

$$\sigma(2x) = \frac{1}{1 + e^{-2x}} \quad (15)$$

From this point, it is trivial to show how eq. (14) can be rewritten in terms of eq. (15):

$$\begin{aligned} tanh(x) &= \frac{1}{1+e^{-2x}} - \frac{e^{-2x}}{1+e^{-2x}} \\ tanh(x) &= \sigma(2x)(1-e^{-2x}) \end{aligned} \tag{16}$$

5.2 Sigmoid and Softmax

Consider the eq. (13) for the sigmoid function. The softmax function is the following:

$$softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{17}$$

These two functions are used for different problems: while sigmoid outputs values in the boundary $[0, 1]$ for binary classification, the softmax computes a probability for each class in a set of n classes. This is the reason why the argument of the softmax is x_i. Mathematically, they could be similar in case the softmax is used for multiclass classification with only two classes:

$$softmax(x_1) = \frac{e^{x_1}}{e^{x_1}+e^{x_2}} = \frac{e^{x_1}}{e^{x_1}+e^{x_2}} \cdot \frac{e^{x_1}}{e^{x_1}} = \frac{1}{\frac{e^{x_1}+e^{x_2}}{e^{x_1}}} = \frac{1}{1+e^{x_2-x_1}} \tag{18}$$

In the special case that x_2 happened to be 0, the $softmax(x_1)$ would then be equal to eq. (13):

$$softmax(x_1) = \frac{1}{1+e^{-x_1}} = \sigma(x_1) \tag{19}$$

5.3 Naïve Bayes and linear models

There is no strict mathematical relationship between these two concepts in the traditional meaning. One could argue that under certain assumptions, Naïve Bayes follows a linear decision boundary, and this could be considered a sort of similarity to linear models. Specifically, Naïve Bayes computes the probability $P(x|y)$ under the assumption of independency $\forall x_i$:

$$P(x|y) = \Pi_{i=1}^{n} P(x_i|y) \tag{20}$$

By applying Bayes' rule and computing the log-likelihood of eq. (20):

$$ln(P(y|x)) = ln(P(y)) + \sum_{i=1}^{n} ln(P(x_i|y)) + C \tag{21}$$

Now, assuming that the probability $P(x_i|y)$ is Gaussian distributed, in a problem of binary classification we can first replace it by the corresponding formula for the log-odds:

$$ln(P(y|x)) = ln\left(\frac{P(y=1)}{P(y=0)}\right) + \sum_{i=1}^{n} ln\left(\frac{P(x_i|y=1)}{P(x_i|y=0)}\right) + C \tag{22}$$

From eq. (22), it is possible to derive the formula for the likelihood of one class:

$$P(y=1|x) = \sigma\left(ln\left(\frac{P(y=1)}{P(y=0)}\right) + \sum_{i=1}^{n} ln\left(\frac{P(x_i|y=1)}{P(x_i|y=0)}\right)\right) \tag{23}$$

Since the probability $P(x_i|y)$ belongs to the exponential family by assumption, its formula can be written as:

$$P(x_i|y) = h_i(x_i)exp(u_{iy}^T \phi_i(x_i) - A_i(u_{iy}))$$

Substituting this into eq. (23):

$$P(y=1|x) = \sigma\left(\sum_{i=1}^{n} w_i^T \phi_i(x_i) + b\right) \tag{24}$$

considering:

$$\begin{aligned} w_i &= u_{i1} - u_{i0} \\ b &= ln\frac{P(y=1)}{P(y=0)} - \sum_{i=1}^{N}(A_i(u_{i1}) - A_i(u_{i0})) \end{aligned} \tag{25}$$

eq. (24) shows a linear version of the Naïve Bayes under the specified assumptions.

5.4 Logistic regression and Bayes' rule

Consider a binary classification problem. We are interested in the posterior probability of the ground truth being $y = 1$ given some sample x. Starting from the Bayes' Rule:

$$P(y=1|x) = \frac{P(x|y=1)P(y=1)}{P(x)} \tag{26}$$

It is possible to derive the formula for the odds $O(y=1|x)$:

$$O(y=1|x) = \frac{P(y=1|x)}{P(y=0|x)} = \frac{P(x|y=1)P(y=1)}{P(x|y=0)P(y=0)} \tag{27}$$

Calculating the log-odds yields this result:

$$ln(O(y=1|x)) = ln\left(\frac{P(x|y=1)}{P(x|y=0)}\right) + ln\left(\frac{P(y=1)}{P(y=0)}\right) \tag{28}$$

Assume for this problem that eq. (28) follows a linear function of x:

$$ln(O(y=1|x)) = \beta x + \beta_0 \tag{29}$$

From the definition of logarithm, eq. (29) can be written as:

$$O(y=1|x) = e^{(\beta x + \beta_0)} \tag{30}$$

Turning the odds back into a probability using the inverse formula, and the definition of eq. (30) gives:

$$P(y=1|x) = \frac{O(y=1|x)}{1+O(y=1|x)} = \frac{e^{(\beta x+\beta_0)}}{1+e^{(\beta x+\beta_0)}} = \frac{1}{1+e^{-(\beta x+\beta_0)}} \tag{31}$$

eq. (31) represents the inverse logit function, the function used in logistic regression to calculate probability, showing how it can be connected to Bayes' Theorem.

5.5 Maximum likelihood and cross entropy

In maximum likelihood estimation, the likelihood of the data is defined as:

$$L(w) = \Pi_{i=1}^{n} P(y_i|x_i, w) \tag{32}$$

With the log-likelihood version of this function being:

$$logL(w) = \sum_{i=1}^{n} ln(P(y_i|x_i, w)) \tag{33}$$

The formula for Cross Entropy Loss in the context of multi-class classification, instead, is:

$$H(q) = -\frac{1}{N}\sum_{i=1}^{N} ln(q(x_i)) = -\frac{1}{N}\sum_{i=1}^{N} ln(P(y_i|x_i, w)) \tag{34}$$

q being the probability distribution function of choice. This means that there exist a relation of direct proportionality between eq. (33) and eq. (34):

$$H(q) = -\frac{1}{N} logL(w) \tag{35}$$