MO435 — Probabilistic Machine Learning
INSTITUTO DE COMPUTAÇÃO — UNICAMP

Project 1

João Victor da Silva Guerra and Leonardo Alves de Melo *

Abstract

In this project, we implemented two approaches of machine learning classifiers: generative and discriminative. For the generative model, we created two datasets with three classes and evaluated the performance of our model based on them. We also described the model derivation to obtain our model. On the other hand, for the discriminative model, we created one dataset with two classes and evaluated the performance of our model based on them. Further, we evaluated the effects of the learning rate in the gradient descent algorithm and the sample size in the Monte Carlo approximation. Finally, we sucessfully implemented both models with good performance metrics, such as accuracy, precision and recall.

1 Introduction

Classification plays an essential role in the field of Machine Learning techniques. The main goal is to predict classes (y) of a set of data points (x), modelled by a mapping function ($f(x)$). A powerful approach to tackle classification problems are to model the conditional probability distribution ($p(y = C_i|x)$) in an inference stage and use it to classify new data across a set of classes ($y = \{C_1, ..., C_n\}$). When classes have already been assigned to the input data, the classification belongs to the category of supervised learning. In this scenario, a sample of data from the input data is used for adjusting the parameters of the classifier model [1]–[3].

There are two approaches to determine the conditional probability distribution, a generative and a discriminative. The first approach models the class-conditional density ($p(x|C_i)$) and the class prior ($p(C_i)$), which applies the Bayes' theorem to model the conditional probability, as described in Eq.(1). The latter approach models the conditional probability distribution directly, optimizing a parametric model, as described in Eq.(2). Both approaches adjust their model's parameters from training data.

$$p(C_i|x) = \frac{p(x, C_i)}{p(x)} = \frac{p(x|C_i)p(C_i)}{\sum_j p(x|C_j)p(C_j)} \tag{1}$$

$$p(C_i|x, W) = \frac{\exp(w_i^T x)}{\sum_j \exp(w_j^T x)} \tag{2}$$

2 Probabilistic generative model

2.1 Data

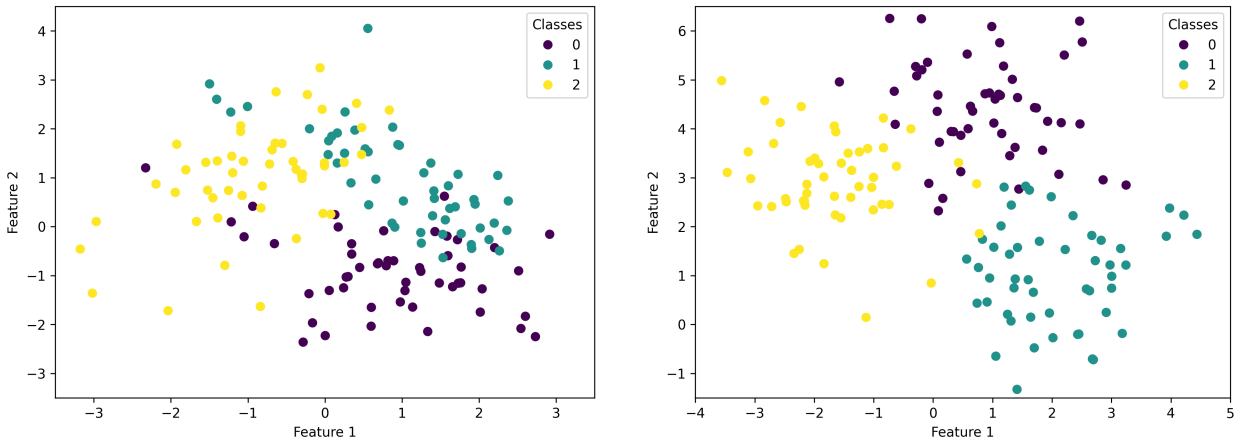
For our experiments, we created two datasets composed by 150 data points divided equally in three classes, named as A and B (Fig. 1). Each data point has two features (Feature 1 and Feature 2). In addition, we split each dataset in two partitions of 75 data points, i. e. training set and test set, with each class equally distributed.

2.2 Model

Here, we shall derive Eq.(1) for our three-class classification problem. First of all, a parametric form must be adopted for the class-conditional densities, then we assume a Gaussian distribution ($\mathcal{N}$) with shared covariance matrix (Σ), as described in Eq.(3).

$$p(x|C_i) \sim \mathcal{N}(x|\mu_i, \Sigma) = \frac{1}{\sqrt{(2\pi)^{(D/2)}|\Sigma|}} \exp\left\{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)\right\} \tag{3}$$

* 117410 and 156188. j117410@dac.unicamp.br and leonardo.alves.melo.1995@gmail.com.



(a) Dataset A

(b) Dataset B

Figure 1. Scatter plot of datasets by class

where μ_i is the mean vector and D is the number of features.

Once we assumed a parametric form for the class-conditional density, we shall determine its parameters (μ_i, Σ) and the class prior ($p(C_i)$), using the maximum likelihood solution. Consider our training set $\{x_n, t_n\}$, where $n = [0, 75)$, $t_n = 0$ denotes class C_0, $t_n = 1$ denotes class C_1 and $t_n = 2$ denotes class C_2. The class prior is given by $p(C_i) = \pi_i$. Hence, the likelihood function is

$$p(x_n, t_n|\pi_i) = \prod_{n=0}^{74} \prod_{i=0}^{2} \{\mathcal{N}(x_n|\mu_i, \Sigma)\pi_i\}^{t_{ni}} \tag{4}$$

and applying the logarithm, we get the log-likelihood function as

$$\ln p(x_n, t_n|\pi_i) = \sum_{n=0}^{74} \sum_{i=0}^{2} t_{ni} \{\ln \mathcal{N}(x_n|\mu_i, \Sigma) + \ln \pi_i\}. \tag{5}$$

2.2.1 Prior class probability

We shall maximize the log-likelihood with respect to π_i but ensuring the constraint $\sum_i \pi_i = 1$. For this, we introduce a Lagrange multiplier λ in Eq.(5) as

$$\ln p(x_n, t_n|\pi_i) + \lambda \left(\sum_{i=0}^{2} \pi_i = 1 \right) \tag{6}$$

and differentiate in relation to π_i and set the result to zero, yields

$$\sum_{n=0}^{74} \frac{t_{ni}}{\pi_i} + \lambda = \frac{N_i}{\pi_i} + \lambda = 0. \tag{7}$$

Summing Eq.(7) over the three classes, we obtaing that $\lambda = -N$, and using this in Eq.(7), we obtain

$$\pi_i = \frac{N_i}{N}. \tag{8}$$

2.2.2 Class-conditional densities

Now we must consider the Gaussian distribution (Eq.(3)) in the log-likelihood function as

$$\ln p(x_n, t_n|\pi_i) = -\frac{1}{2} \sum_{n=0}^{74} \sum_{i=0}^{2} t_{ni} \left[\ln |\Sigma| + (x_n - \mu_i)^T \Sigma^{-1} (x_n - \mu_i) + \ln \pi_i \right] \tag{9}$$

where $\ln \pi_i$ is constant and independent of μ_i and Σ.

Hence, we shall maximize the log-likelihood with respect to μ_i, setting the derivate in relation to μ_i to zero, we obtain

$$\sum_{n=0}^{74} t_{ni}(x_n - \mu_i) = 0 \implies \mu_i = \frac{1}{N_i}\sum_{n=0}^{74} t_{ni} x_n \tag{10}$$

Finally, we shall maximize the log-likelihood with respect to Σ. Choosing the terms dependent on Σ, we can rearrange it as

$$\begin{aligned} &-\frac{N}{2}\ln|\Sigma| - \frac{1}{2}\sum_{n=0}^{74}\sum_{i=0}^{2} t_{ni}\left[(x_n-\mu_i)^T\Sigma^{-1}(x_n-\mu_i)\right] \\ &= -\frac{N}{2}\ln|\Sigma| - \frac{N}{2} tr\left(\Sigma^{-1}\frac{1}{N}\sum_{n=0}^{74}\sum_{i=0}^{2} t_{ni}\left[(x_n-\mu_i)(x_n-\mu_i)^T\right]\right) \\ &= -\frac{N}{2}\ln|\Sigma| - \frac{N}{2} tr\left(\Sigma^{-1}S\right). \end{aligned} \tag{11}$$

where we define

$$S = \sum_{i=0}^{2}\frac{N_i}{N}S_i \tag{12}$$

$$S_i = \frac{1}{N_i}\sum_{n=0}^{74} t_{ni}\left[(x_n-\mu_i)(x_n-\mu_i)^T\right] \tag{13}$$

Then, differentiating Eq.(11) in relation to Σ and equating to zero, we obtain that $\Sigma = S$.

2.2.3 Results

With our training sets, we adjusted the parameters μ_i, Σ and prior probabilities (π_i) for datasets A and B, which are shown in Table 1. Then, we replace the parameters in the Eq.(1), and obtained the following expression for the conditional probability of each class for each data point

$$p(C_i|x) = \frac{\mathcal{N}(x|\mu_i,\Sigma)\pi_i}{\mathcal{N}(x|\mu_0,\Sigma)\pi_0 + \mathcal{N}(x|\mu_1,\Sigma)\pi_1 + \mathcal{N}(x|\mu_2,\Sigma)\pi_2}. \tag{14}$$

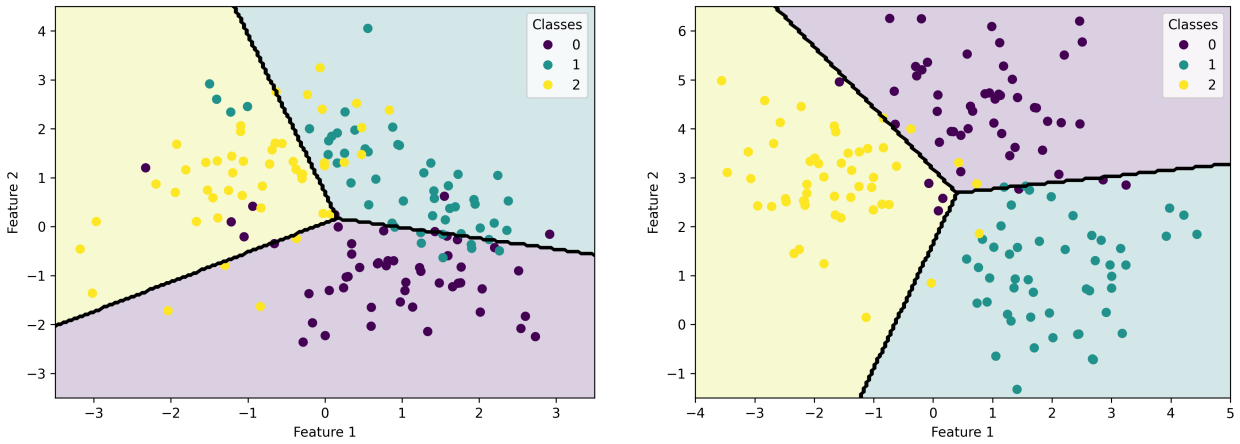
Table 1. Parameters adjusted through maximum likelihood solution for datasets A and B.

Dataset	π_0	π_1	π_2	μ_0	μ_1	μ_2	Σ
A	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\begin{bmatrix} 0.76 \\ -0.93 \end{bmatrix}$	$\begin{bmatrix} 0.86 \\ 0.99 \end{bmatrix}$	$\begin{bmatrix} -1.03 \\ 0.93 \end{bmatrix}$	$\begin{bmatrix} 1.20 & -0.23 \\ -0.23 & 0.85 \end{bmatrix}$
B	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	$\begin{bmatrix} 0.83 \\ 4.45 \end{bmatrix}$	$\begin{bmatrix} 2.03 \\ 1.19 \end{bmatrix}$	$\begin{bmatrix} -1.70 \\ 3.00 \end{bmatrix}$	$\begin{bmatrix} 1.24 & -0.24 \\ -0.24 & 1.05 \end{bmatrix}$

Given any data point x, we calculate the conditional probability for the classes 0, 1 and 2. Hence, the data point is predicted to the class with the higher conditional probability, as described in Eq.(15).

$$\hat{C} = \underset{C_i}{\operatorname{argmax}}\, p(C_i|x) \tag{15}$$

Finally, we ploted a scatter plot with all data points (training and test set) colored by the actual class and the background colored by the predicted class with the linear decision boundaries (Fig. 2).



(a) Dataset A **(b)** Dataset B

Figure 2. Scatter plot of datasets colored by class with contour plots colored by predicted class.

2.2.4 Performance evaluation

We used three metrics to evaluate our probabilistic generative model: Accuracy (Eq.(16)), Precision (Eq.(17)) and Recall (Eq.(18)).

$$Accuracy(\%) = \frac{TP}{P} \times 100\% \tag{16}$$

$$Precision(\%) = \frac{TP}{TP+FP} \times 100\% \tag{17}$$

$$Recall(\%) = \frac{TP}{TP+FN} \times 100\% \tag{18}$$

where TP is the number of true positives, P is the number of positives, FP is the number of false positives and FN is the number of false positives.

The accuracy, precision and recall of our training and test sets are presented for dataset A and B in Table 2 and Table 3, respectively.

Table 2. Performance evaluation of dataset A.

	Dataset A (%)					
	Training set			**Test set**		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	79	84	84	73	71	85
1		76	76		70	73
2		76	76		82	61

Table 3. Performance evaluation of dataset B.

	Dataset B (%)					
	Training set			**Test set**		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	88	91	80	95	92	92
1		86	96		96	100
2		88	88		96	92

Our probabilistic generative model showed a reliable prediction performance, shown in our evaluation metrics. However, our model had some limitation when dealing with outliers. The Gaussian distribution fitted to the

class-conditional probabilities are sensible to outliers, because the maximum likelihood estimation of a Guassian is not robust, as discussed by [1]. Then, analyzing Fig. 2, we suggest that misclassified points could be outliers in their respective class. Probably a distribution robust to outliers, e. g. Laplace distribution, would yield a higher accuracy. Furthermore, the Gaussian with shared covariance matrix determine linear decision boundaries; however, we could have adopted individual covariances for each class, which would yield a quadratic decision boundary, what in some cases would improve our prediction capacity.

3 Discriminative model

3.1 Data

For the second experiment, we created a dataset composed by 100 data points divided equally in two classes, named as C (Fig. 3). The dataset was splited in the same way of the first experiment, with 50% points in the training set and the remaining in the test set.

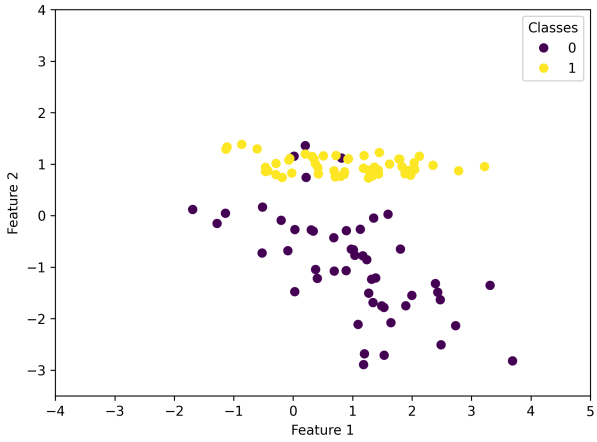


Figure 3. Scatter plot of dataset C by class

3.2 Model

Here, we shall implement a logistic regression to our two-class classification problem. We implemented a gradient descent algorithm to optimize the parameters of the logistic model and the Monte Carlo sampling to approximate it based on S samples.

3.2.1 Gradient Descent

The gradient descent formula to optimize the parameter w in a function ℓ is given by Eq.(19), where α is the learning rate, that is, the size of each step. To obtain the function ℓ, we must first define the hypothesis for the logistic regression, which is given by Eq.(20). Intuitively, we can define for y being one by Eq.(21) and being zero by Eq.(22).

$$w_{i+1} = w_i - \alpha \frac{\delta \ell(w_i)}{\delta w_i} \tag{19}$$

$$h(x) = \frac{1}{1 + e^{-W^T x}} \tag{20}$$

$$p(y_i = 1 | x_i; W) = h(x_i) \tag{21}$$

$$p(y_i = 0 | x_i; W) = 1 - h(x_i) \tag{22}$$

We can use a Bernoulli distribution to join Eq.(21) and Eq.(22) to find the Eq.(23). We can see that for $y = 1$ the term $h(x_i)^{y_i}$ becomes $h(x_i)$ and $(1-h(x_i))^{1-y_i}$ becomes 1, resulting in $h(x_i)$, and if $y = 0$ the term $h(x_i)^{y_i}$ becomes 1 and the $(1-h(x_i))^{1-y_i}$ becomes $(1-h(x_i))$, resulting in $(1-h(x_i))$. We can obtain the likelihood from this equation obtaining (24).

$$p(y_i|x_i;W) = h(x_i)^{y_i}(1-h(x_i))^{1-y_i} \tag{23}$$

$$L(W) = \prod_{i=1}^{m} h(x_i)^{y_i}(1-h(x_i))^{1-y_i} \tag{24}$$

With likelihood equation, we obtain the log-likelihood (Eq.(25)) from it.

$$\ell(W) = \sum_{i=1}^{m} y_i \log h(x_i) + (1-y_i)\log(1-h(x_i)) \tag{25}$$

Hence, we have to maximize the log-likelihood to optimize the parameters. To do so, we derive Eq.(25) in terms of the parameters W, obtaining the Eq.(26).

$$\frac{\delta \ell(W)}{\delta W} = (Y - h(X))X \tag{26}$$

Now, we return to Eq.(19) and replace the derivative, finally obtaining the Eq.(27), in which we can optimize W.

$$w_{i+1} = w_i - \alpha(Y - h(X))X \tag{27}$$

Finally, we will evaluate the effects of different learning rates on the performance of gradient descent algorithm. We apply learning rates ranging from $1e-7$ to 34.9.

3.2.2 Prediction with Monte Carlo

Knowing that the posterior is intractable, to obtain the uncertainty, we will use Monte Carlo, given the Eq.(28). We will get the normal using W as average and $W/10$ as covariance.

$$p(y = 1|x, D) \approx \frac{1}{S}\sum_{s=1}^{S} h(x) \tag{28}$$

We will evaluate the effects of the sample size on the performance of the Monte Carlo sampling. We apply the following sample sizes: 100, 1000, 10000 and 100000

3.2.3 Results

First, we ploted the loss function versus the learning rate (Fig. 4). We observed that, for a small learning rate, i.e. between $1e-7$ and $1e-5$, the loss function does not converge in a reasonable time. Between $1e-5$ and 10, the loss function decreases constantly, which represents a good choice of learning rate, and between 10 and 34.9, the loss function diverges. Based on this, we chose the values of $1e-7$, $1e-3$ and 34.9 to be the values of learning rate to be tested, using an example of each part of the curve.

We plotted all results of the test set with the varying learning rates and samples sizes, as previously mentioned, in the Fig. 5. We noted that the linear decision boundary was incorrectly fitted for a very small learning rate, sucessfully fitted for a middle and large learning rates, with some small differences between them, which was expected. In addition, the performance metrics, such as accuracy, precision and recall, are showed in the Tables 4 to 15. As shown, the most part of the configurations predicted the classes sucessfully for a middle and large learning rates, and incorrectly for a very small learning rate, achieving reasonable metrics.

We also plotted the histograms of W for bias, Feature 1 and Feature 2 are shown in Figs. 6, 7 and 8, respectively. As observed, with a small number of samples, the histogram does not resembles a Gaussian distribution, which is expected. Then, with a increasing number of samples, the histogram also increases its resemblance to a Gaussian distribution. Therefore, the sample size S yield good approximations if it is able to properly describe the real distribution of W.

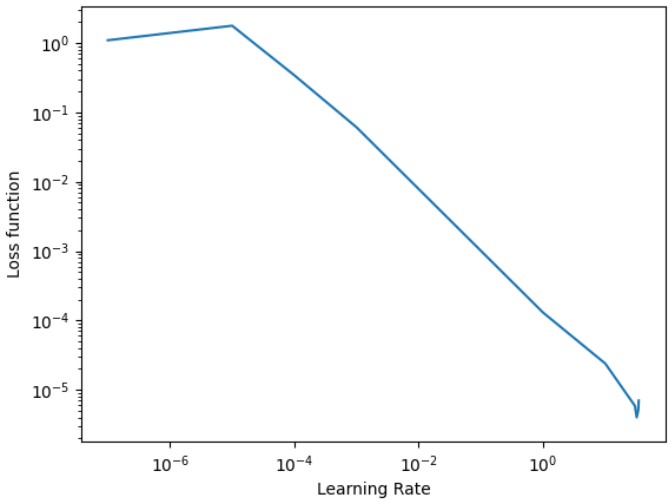
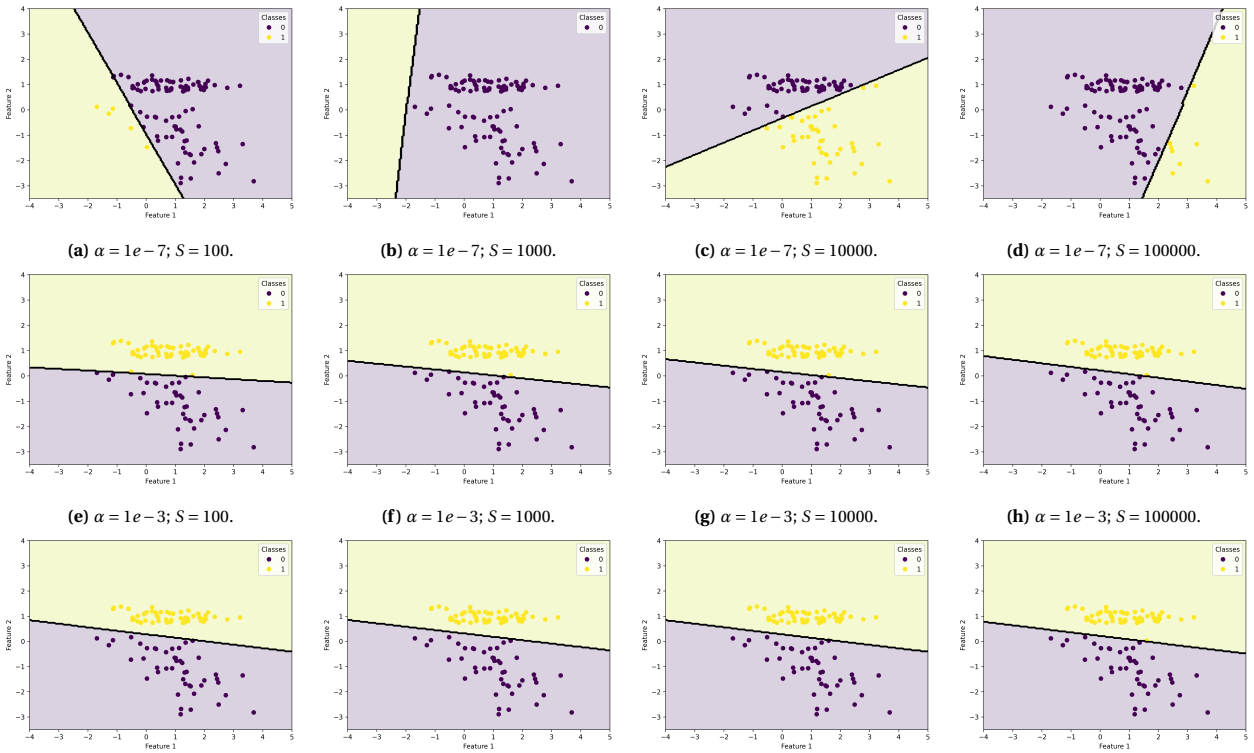


Figure 4. Loss function with different learning rates.

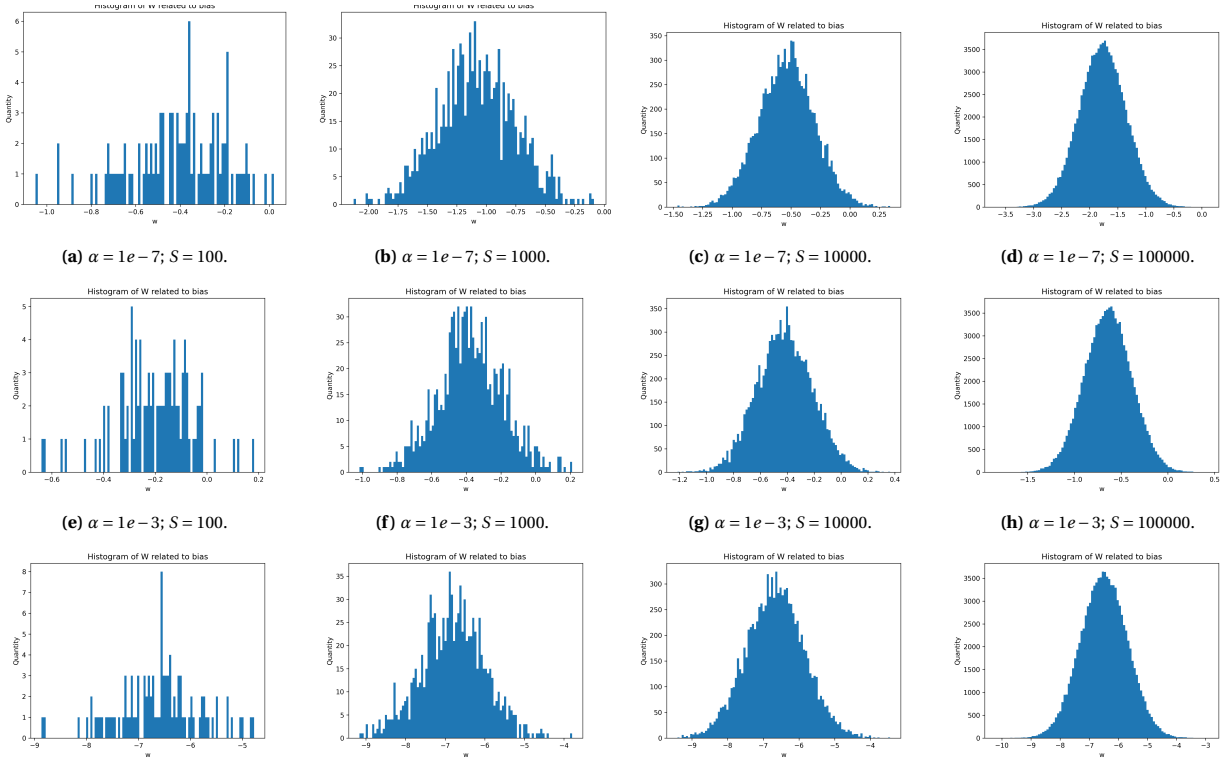


(a) $\alpha = 1e-7$; $S = 100$. **(b)** $\alpha = 1e-7$; $S = 1000$. **(c)** $\alpha = 1e-7$; $S = 10000$. **(d)** $\alpha = 1e-7$; $S = 100000$.

(e) $\alpha = 1e-3$; $S = 100$. **(f)** $\alpha = 1e-3$; $S = 1000$. **(g)** $\alpha = 1e-3$; $S = 10000$. **(h)** $\alpha = 1e-3$; $S = 100000$.

(i) $\alpha = 34.9$; $S = 100$. **(j)** $\alpha = 34.9$; $S = 1000$. **(k)** $\alpha = 34.9$; $S = 10000$. **(l)** $\alpha = 34.9$; $S = 100000$.

Figure 5. Scatter plot of the predictions over the test set with a linear boundary decision, using different learning rates (α) and sample sizes (S) for Monte Carlo approximation.

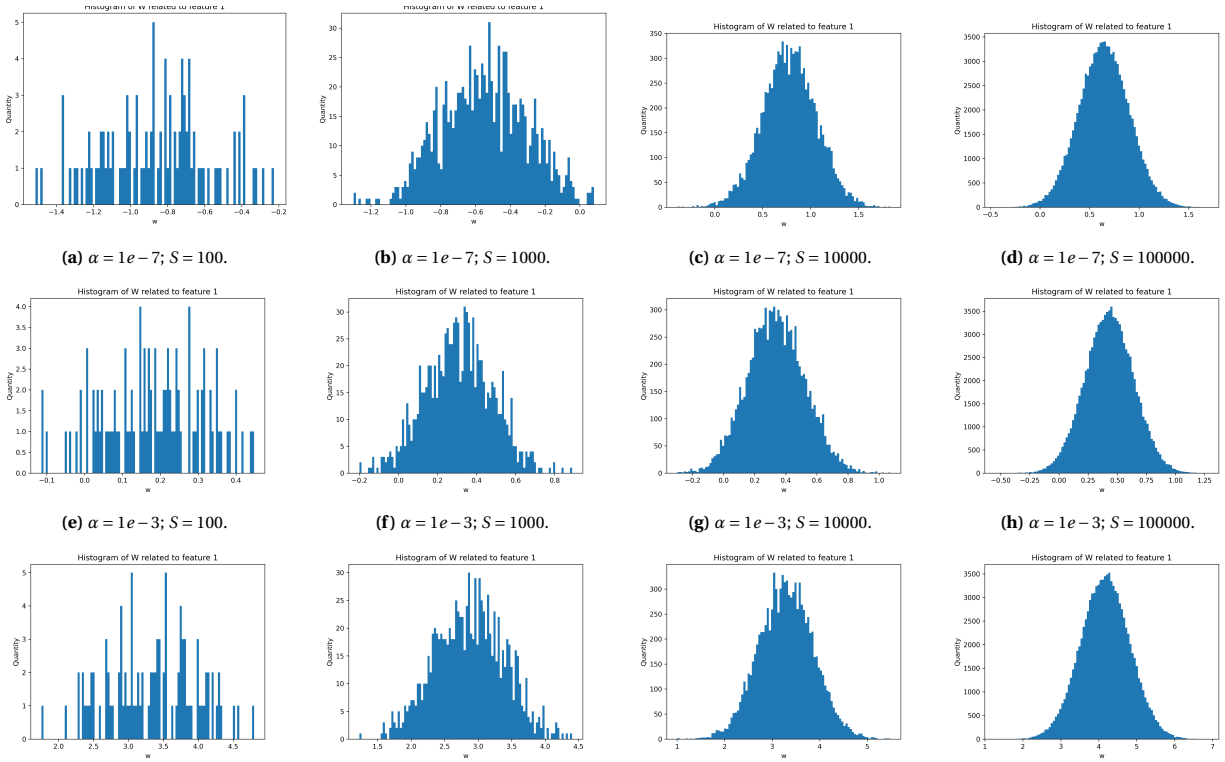


(a) $\alpha = 1e-7$; $S = 100$. **(b)** $\alpha = 1e-7$; $S = 1000$. **(c)** $\alpha = 1e-7$; $S = 10000$. **(d)** $\alpha = 1e-7$; $S = 100000$.

(e) $\alpha = 1e-3$; $S = 100$. **(f)** $\alpha = 1e-3$; $S = 1000$. **(g)** $\alpha = 1e-3$; $S = 10000$. **(h)** $\alpha = 1e-3$; $S = 100000$.

(i) $\alpha = 34.9$; $S = 100$. **(j)** $\alpha = 34.9$; $S = 1000$. **(k)** $\alpha = 34.9$; $S = 10000$. **(l)** $\alpha = 34.9$; $S = 100000$.

Figure 6. Histogram of W for bias with differents learning rates (α) and sample sizes (S) for Monte Carlo approximation.

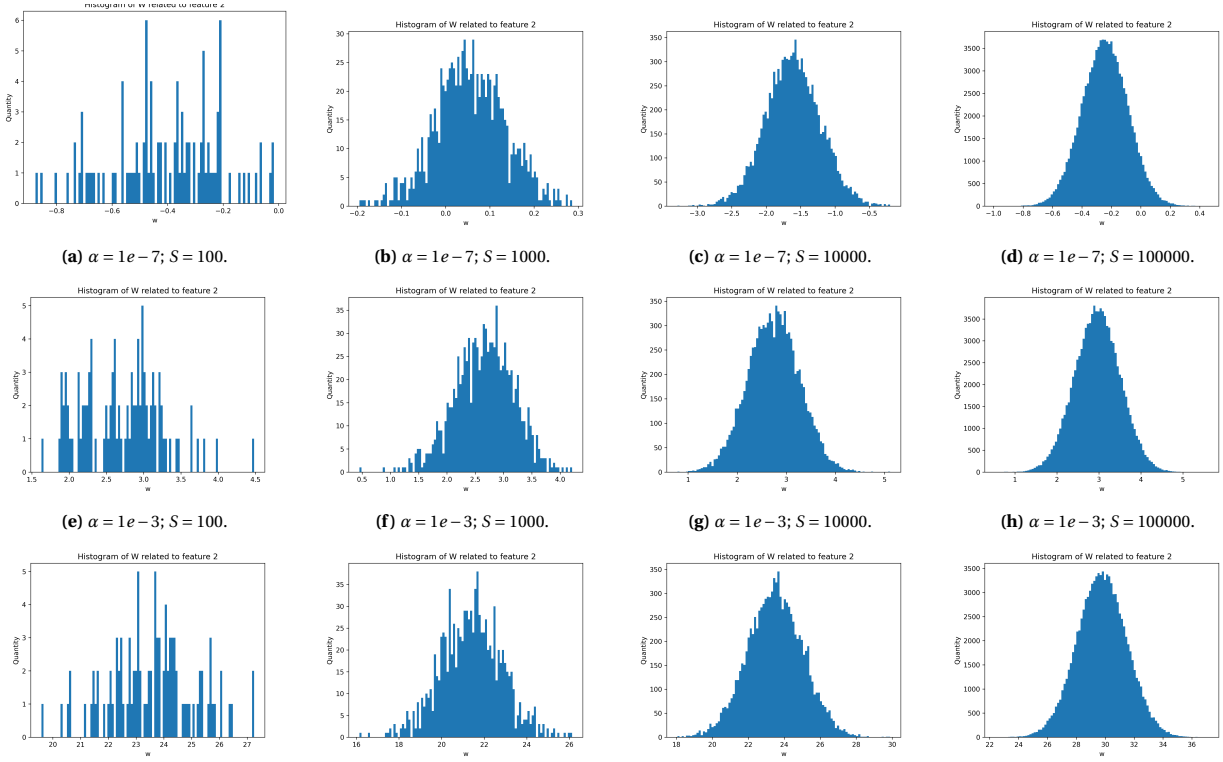


(a) $\alpha = 1e-7$; $S = 100$. **(b)** $\alpha = 1e-7$; $S = 1000$. **(c)** $\alpha = 1e-7$; $S = 10000$. **(d)** $\alpha = 1e-7$; $S = 100000$.

(e) $\alpha = 1e-3$; $S = 100$. **(f)** $\alpha = 1e-3$; $S = 1000$. **(g)** $\alpha = 1e-3$; $S = 10000$. **(h)** $\alpha = 1e-3$; $S = 100000$.

(i) $\alpha = 34.9$; $S = 100$. **(j)** $\alpha = 34.9$; $S = 1000$. **(k)** $\alpha = 34.9$; $S = 10000$. **(l)** $\alpha = 34.9$; $S = 100000$.

Figure 7. Histogram of W for feature 1 with differents learning rates (α) and sample sizes (S) for Monte Carlo approximation.



(a) $\alpha = 1e-7$; $S = 100$. **(b)** $\alpha = 1e-7$; $S = 1000$. **(c)** $\alpha = 1e-7$; $S = 10000$. **(d)** $\alpha = 1e-7$; $S = 100000$.

(e) $\alpha = 1e-3$; $S = 100$. **(f)** $\alpha = 1e-3$; $S = 1000$. **(g)** $\alpha = 1e-3$; $S = 10000$. **(h)** $\alpha = 1e-3$; $S = 100000$.

(i) $\alpha = 34.9$; $S = 100$. **(j)** $\alpha = 34.9$; $S = 1000$. **(k)** $\alpha = 34.9$; $S = 10000$. **(l)** $\alpha = 34.9$; $S = 100000$.

Figure 8. Histogram of W for feature 2 with differents learning rates (α) and sample sizes (S) for Monte Carlo approximation.

Table 4. Performance evaluation of discriminative model for learning rate $1e-07$ and 100 samples

	Dataset C (%)					
	Training set			**Testing set**		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	42.0	45.65	84.0	48.0	48.97	96.0
1		0.0	0.0		0.0	0.0

4 Conclusion

In this project, we implemented two types of machine learning algorithms to predict classes: generative and discriminative. In the probabilistic generative model, we fitted three linear decision boundary for three classes in each dataset. We achieved an accuracy of 73% and 95% in the test set of dataset A and B, respectively. In the discriminative model, we applied a gradient descent algorithm to maximize likelihood function of a logistic regression, and used Monte Carlo sampling in the prediction step, which for good choices of learning rate, achieved a reasonable performance, with high values of accuracy, precision and recall, and for a bad choice of learning rate, those metrics achieved bad results, what was expected. In addition, both models yield linear decision boundaries to classify data between classes. Further, the training procedures are different; the generative approach maximize the joint log-likelihood and the discriminative maximize the conditional log-likelihood. However, logistic regression should performe better than the probabilistic generative model, when Gaussian assumptions are not applicable, e. g., with outlier data.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st ed. Springer, 2006, ISBN: 0-387-31073-8.

[2] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012, pp. 27–33.

Table 5. Performance evaluation of discriminative model for learning rate 1e-07 and 1000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	50.0	50.0	100.0	50.0	50.0	100.0
1		0	0.0		0	0.0

Table 6. Performance evaluation of discriminative model for learning rate 1e-07 and 10000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	10.0	14.28	16.0	14	20.0	24.0
1		4.54	4.0		5.0	4.0

[3] R. Sathya and A. Abraham, "Comparison of supervised and unsupervised learning algorithms for pattern classification," *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, no. 2, 2013. DOI: 10.14569/ijarai.2013.020206.

Table 7. Performance evaluation of discriminative model for learning rate 1e-07 and 100000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	42.0	45.45	80.0	46.0	47.91	92.0
1		16.66	4.0		0.0	0.0

Table 8. Performance evaluation of discriminative model for learning rate 0.001 and 100 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	88.0	100.0	76.0
1		100.0	100.0		80.64	100.0

Table 9. Performance evaluation of discriminative model for learning rate 0.001 and 1000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	90.0	100.0	80.0
1		100.0	100.0		83.33	100.0

Table 10. Performance evaluation of discriminative model for learning rate 0.001 and 10000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	90.0	100.0	80.0
1		100.0	100.0		83.33	100.0

Table 11. Performance evaluation of discriminative model for learning rate 0.001 and 100000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	90.0	100.0	80.0
1		100.0	100.0		83.333	100.0

Table 12. Performance evaluation of discriminative model for learning rate 34.9 and 100 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	92.0	100.0	84.0
1		100.0	100.0		86.20	100.0

Table 13. Performance evaluation of discriminative model for learning rate 34.9 and 1000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	92.0	100.0	84.0
1		100.0	100.0		86.20	100.0

Table 14. Performance evaluation of discriminative model for learning rate 34.9 and 10000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	92.0	100.0	84.0
1		100.0	100.0		86.20	100.0

Table 15. Performance evaluation of discriminative model for learning rate 34.9 and 100000 samples

	Dataset C (%)					
	Training set			Testing set		
Classes	**Accuracy**	**Precision**	**Recall**	**Accuracy**	**Precision**	**Recall**
0	100.0	100.0	100.0	90.0	100.0	80.0
1		100.0	100.0		83.33	100.0