Machine Learning 2024 Spring

HW2: Classification

109511286 蔡佩蓉

<u>Methodology</u>

Training data $D \equiv \{(x_1, t_1), (x_2, t_2), \ldots, (x_N, t_N)\}$, where t_n's are represented in the 1-of-K format (one hot encoding).

Probabilistic Generative Model

In generative models, we compute (infer) the class-conditional densities $p(x|C_k)$ for each class C_k, as well as the class priors $p(C_k)$, and then use them to compute posterior class probabilities $p(C_k|x)$ through Bayes' theorem.

$$p(C_k|x) = \frac{p(x, C_k)}{\sum_{i=1}^{K} p(x, C_i)} = \frac{p(x|C_k)p(C_k)}{\sum_{i=1}^{K} p(x|C_i)p(C_i)} = \frac{p(x|C_k)p(C_k)}{p(x)}$$

Using the posterior (class) probabilities, we can determine class membership for each new input x. Such classifiers are called Bayesian Classifiers. This approach explicitly or implicitly models the distribution of inputs as well as outputs.

Class prior:

$$p(C_k) = \frac{N_k}{\sum_j N_j} = \frac{N_k}{N}$$

Class-conditional probability distributions (Gaussian distribution with the same covariance matrix but different mean vectors):

$$p(x|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x - \mu_k)^T \Sigma^{-1} (x - \mu_k)\right\}$$

where

$$\mu_k = \frac{1}{N_k} \sum_{x_n \in C_k} x_n$$

and

$$\Sigma = \frac{1}{N} \sum_{k=1}^{K} N_k S_k$$

where

$$S_k = \frac{1}{N_k} \sum_{x_n \in C_k} (x_n - \mu_k)(x_n - \mu_k)^T$$

Probabilistic Discriminative Model

In the discriminative model, we directly model the conditional probability $p(C_i|x)$ without explicitly modeling the joint distribution. Here, we define the conditional probability as:

$$p(C_i|x) = y_k(x) = \frac{\exp\big(a_k(x) - a_1(x)\big)}{\sum_{j=1}^{K} \exp\Big(a_j(x) - a_1(x)\Big)}$$

where

$$a_k(x) = w_k^T \phi(x)$$

where

$$\phi(x) = [\phi_0(x), \phi_1(x), \dots, \phi_{M-1}(x)]^T$$

$$\phi_0(x) = 1, \phi_0(x) = x_1, \phi_0(x) = x_2$$

x_1: the offensive value of one data point

x_2: the defensive value of one data point

and

$$w_k = \left[w_{k,0}, w_{k,1}, \dots, w_{k,M-1}\right]^T$$

Define

$$W = [w_1^T, w_2^T, \dots, w_K^T]^T$$

Gradient Descent method

$$W^{(new)} = W^{(old)} - \eta \nabla E\big(W^{(old)}\big)$$

where

$$\nabla E(W) = \left[\left(\nabla_{w_1} E(W)\right)^T, \left(\nabla_{w_2} E(W)\right)^T, \dots, \left(\nabla_{w_K} E(W)\right)^T\right]^T$$

$$\nabla_{w_j} E(W) = \sum_{n=1}^{N} \big(y_j(x_n; w_j) - t_{nj}\big)\phi(x_n)$$

****Extra (**updating the weights using Adagrad):

$$\forall d: W_d^{(new)} = W_d^{(old)} - \frac{\eta}{\sqrt{z_d + \epsilon}} \nabla E\left(W_d^{(old)}\right)$$

where $\forall d: z_d \leftarrow z_d + \nabla E\left(W_d^{(old)}\right)^2$

Newton-Raphson method

$$W^{(new)} = W^{(old)} - H^{-1}\nabla E\left(W^{(old)}\right)$$

where Hessian matrix, $H = \nabla\nabla E(W)$

where

$$\nabla_{w_k}\nabla_{w_j}E(W) = \sum_{n=1}^{N} y_k(x_n; w_k)\left(I_{kj} - y_j\left(x_n; w_j\right)\right)\phi(x_n)\phi(x_n)^T$$

In theory, since $\nabla_{w_k}\nabla_{w_j}E(W) = \nabla_{w_j}\nabla_{w_k}E(W)$, H is symmetric matrix.

Confusion Matrix and Accuracy

$$\text{Confusion Matrix} = \begin{bmatrix} T_0 & F_{01} & F_{02} & F_{03} \\ F_{10} & T_1 & F_{12} & F_{13} \\ F_{20} & F_{21} & T_2 & F_{23} \\ F_{30} & F_{31} & F_{32} & T_3 \end{bmatrix}$$

where

T_i: The number of samples in class i that are actually predicted as i by the model.

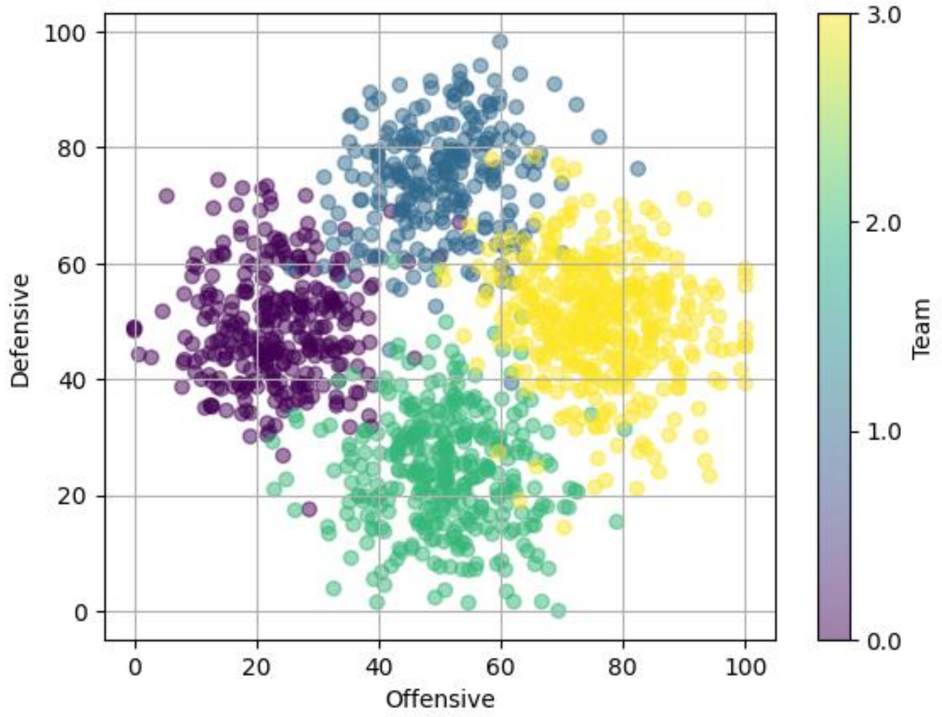
F_{ij}: The number of samples in class i that are actually predicted as j by the model.

$$\text{Accuracy} = \frac{\sum_{i=0}^{3} T_i}{\sum_{i=0}^{3}\left(T_i + \sum_{i=0}^{3} F_{ij}\right)} \times 100\%$$

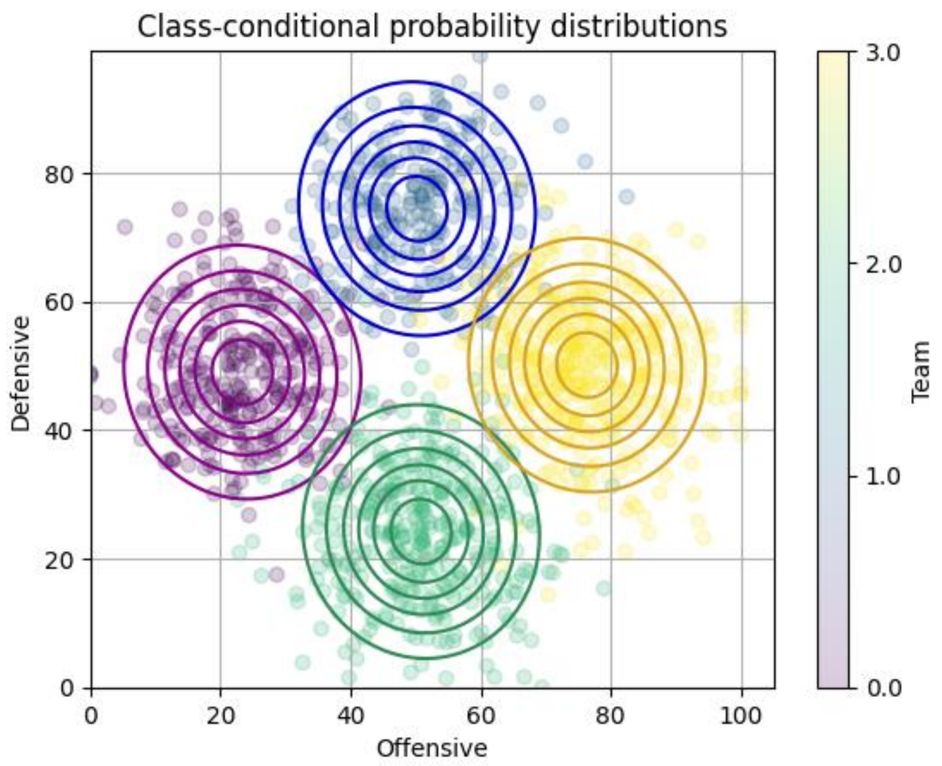
Cross-entropy error function

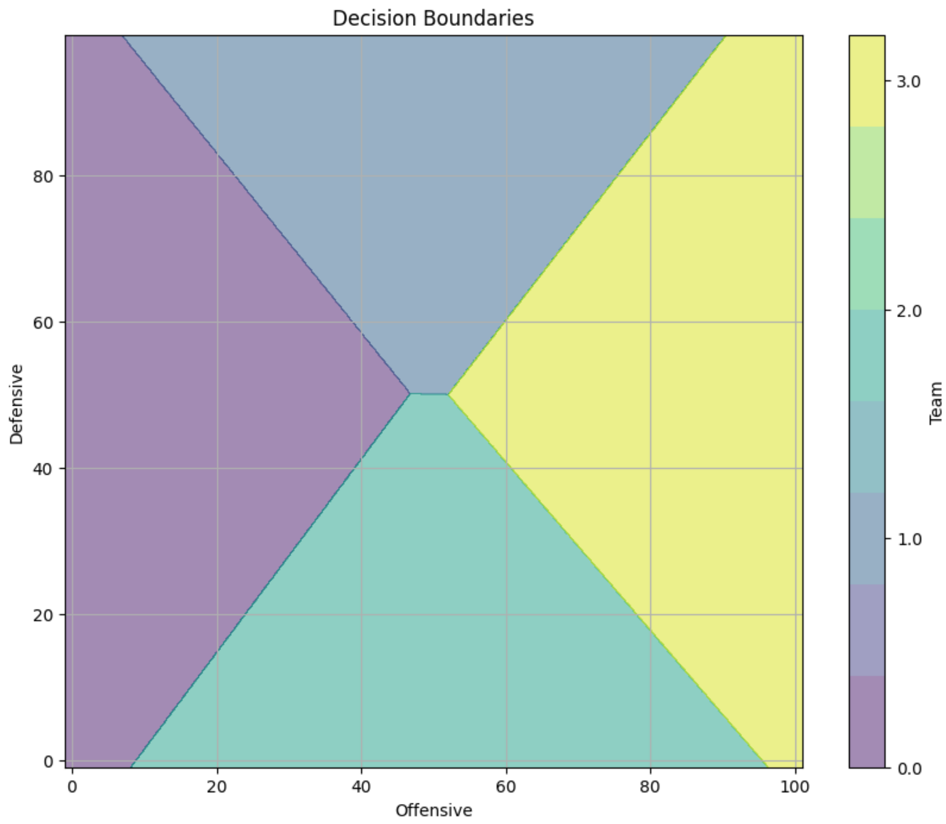
$$-E(w_1, w_2, \dots, w_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_k(x_n; w_k)$$

Part I

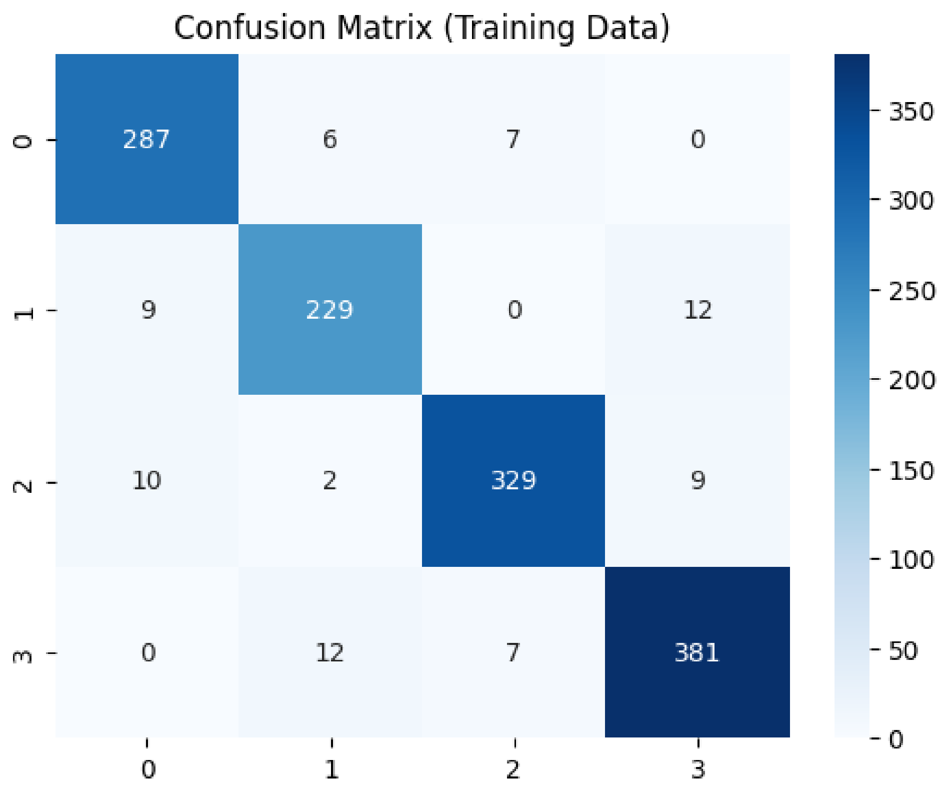


Generative Model

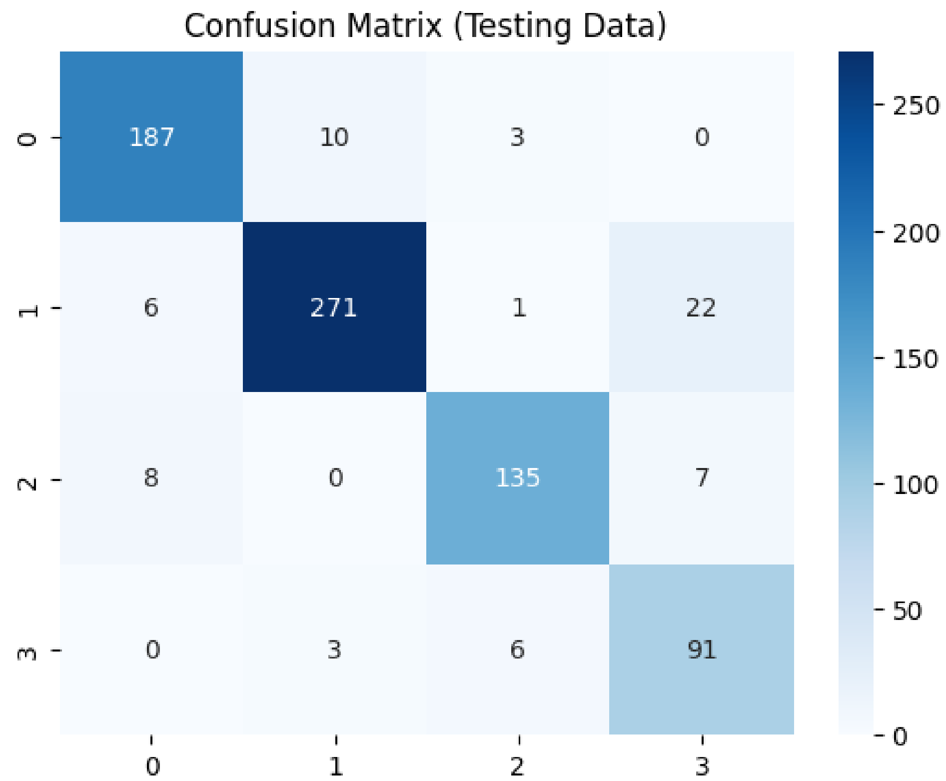




Confusion Matrix



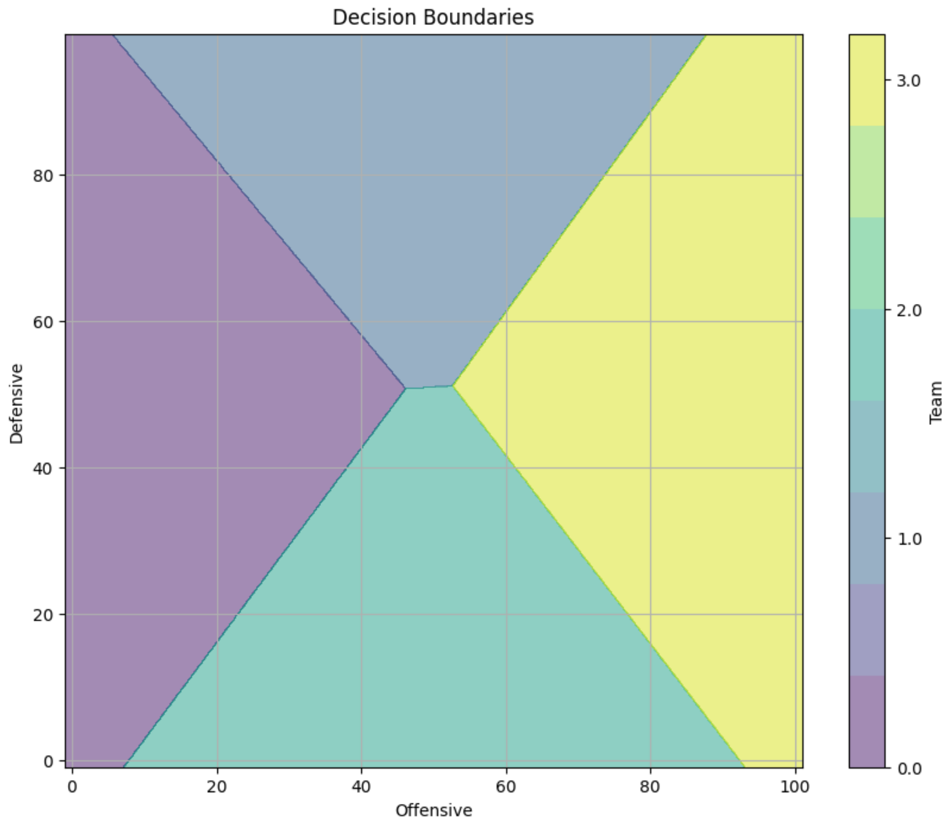
`Accuracy (Training Data): 94.3076923076923`



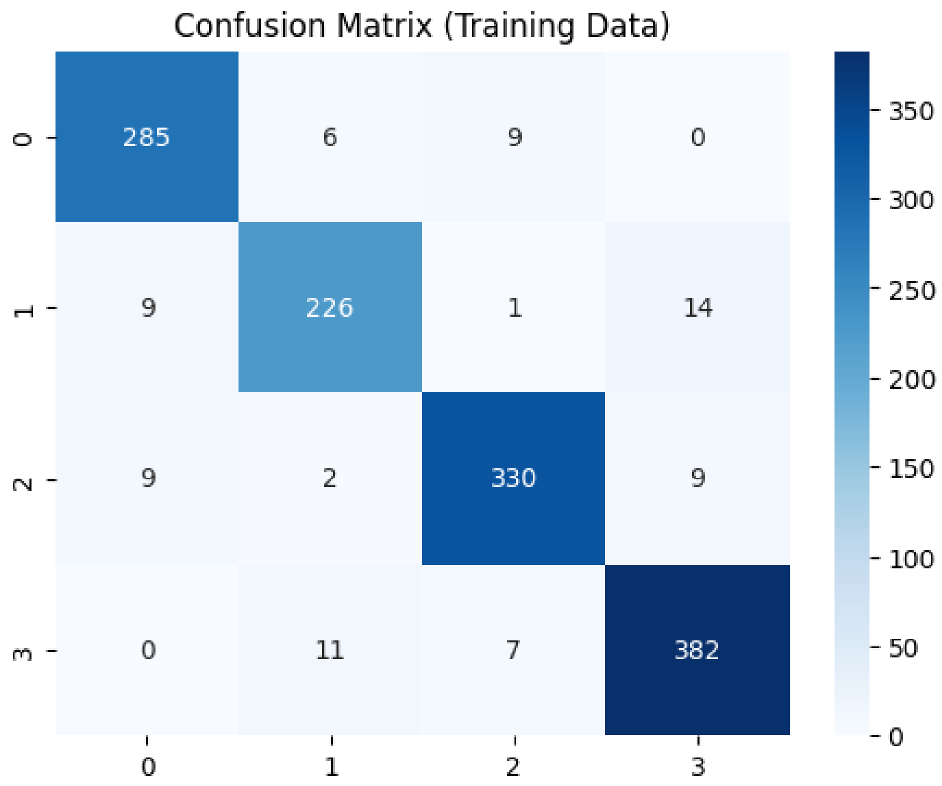
`Accuracy (Testing Data): 91.2`

Discriminative Model

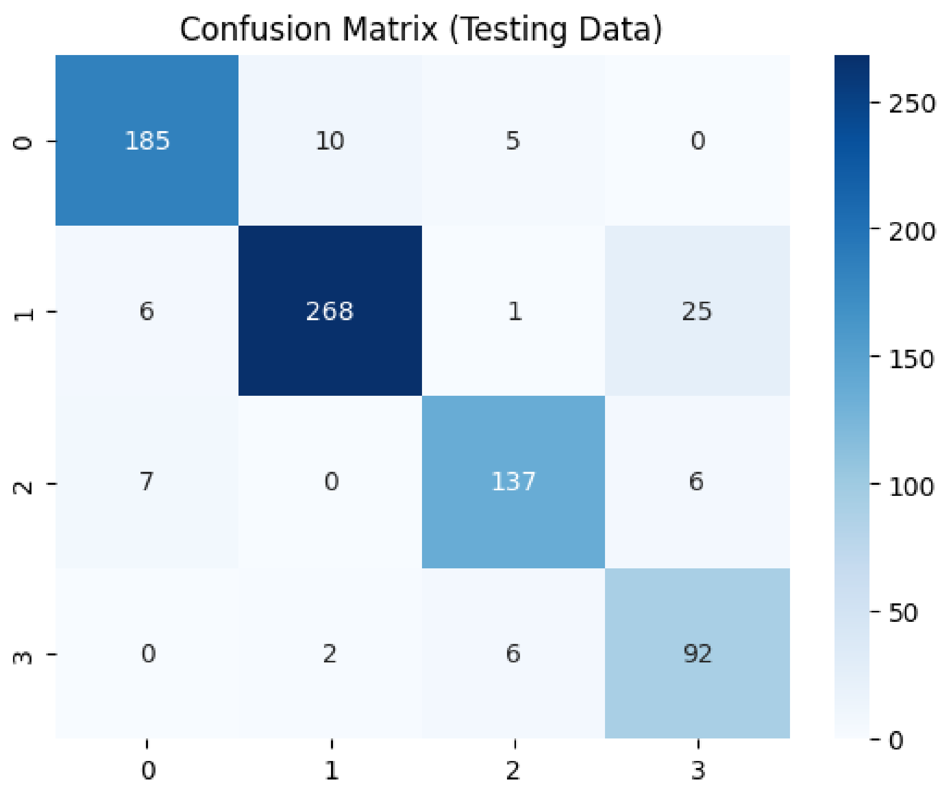
Gradient Descent method



Confusion Matrix

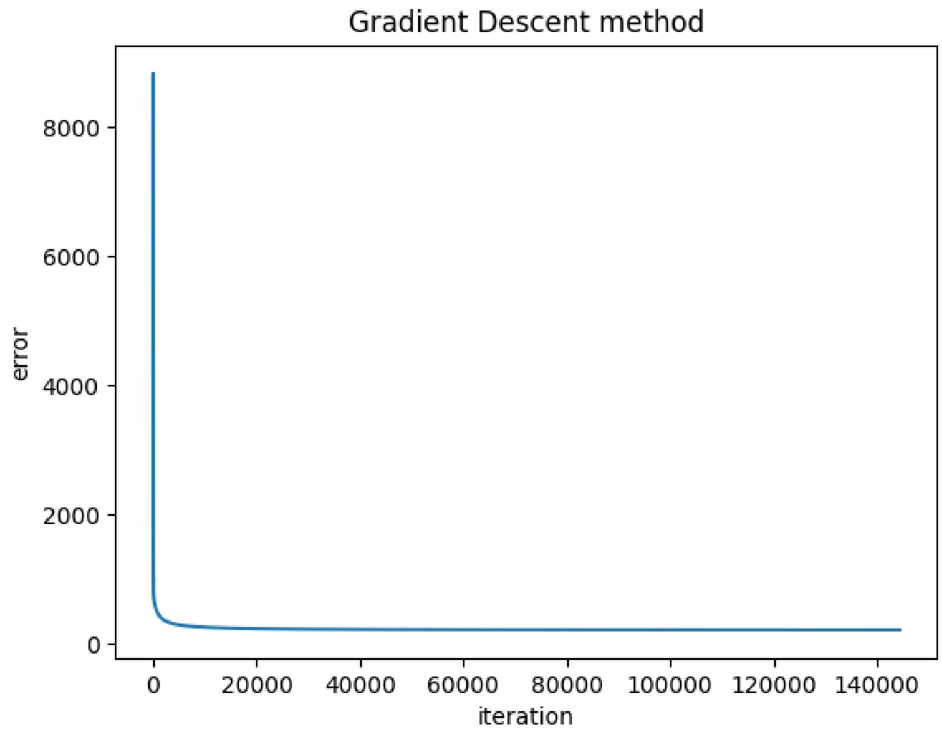


Accuracy (Training Data): 94.07692307692308

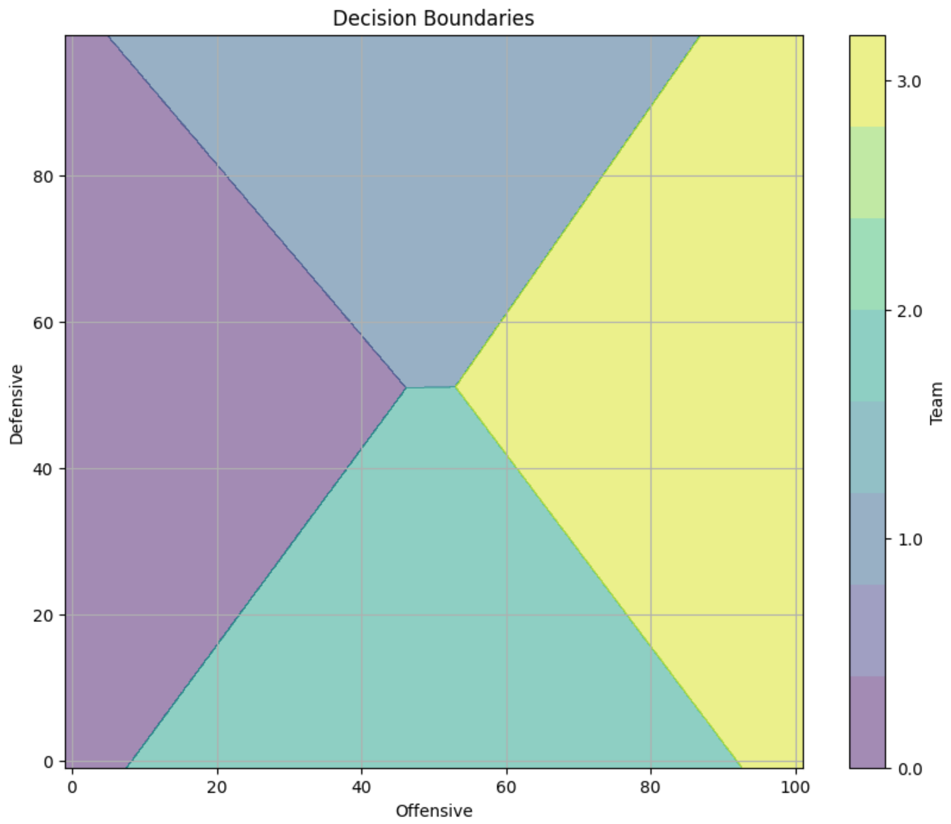


Accuracy (Testing Data): 90.93333333333334

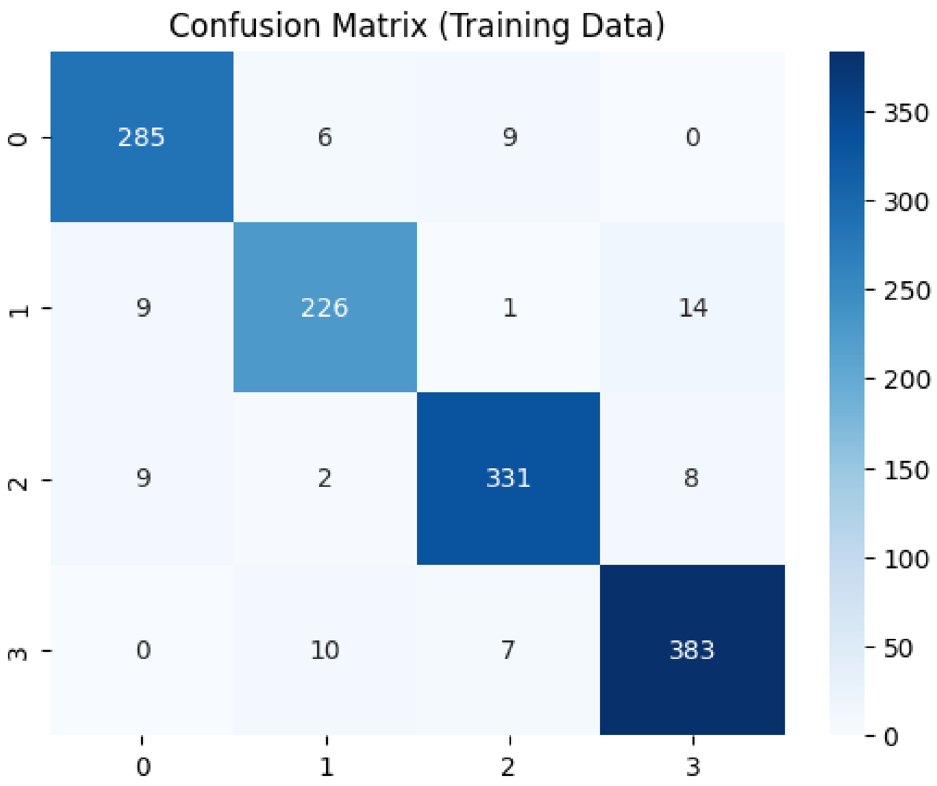
Cross-entropy error function



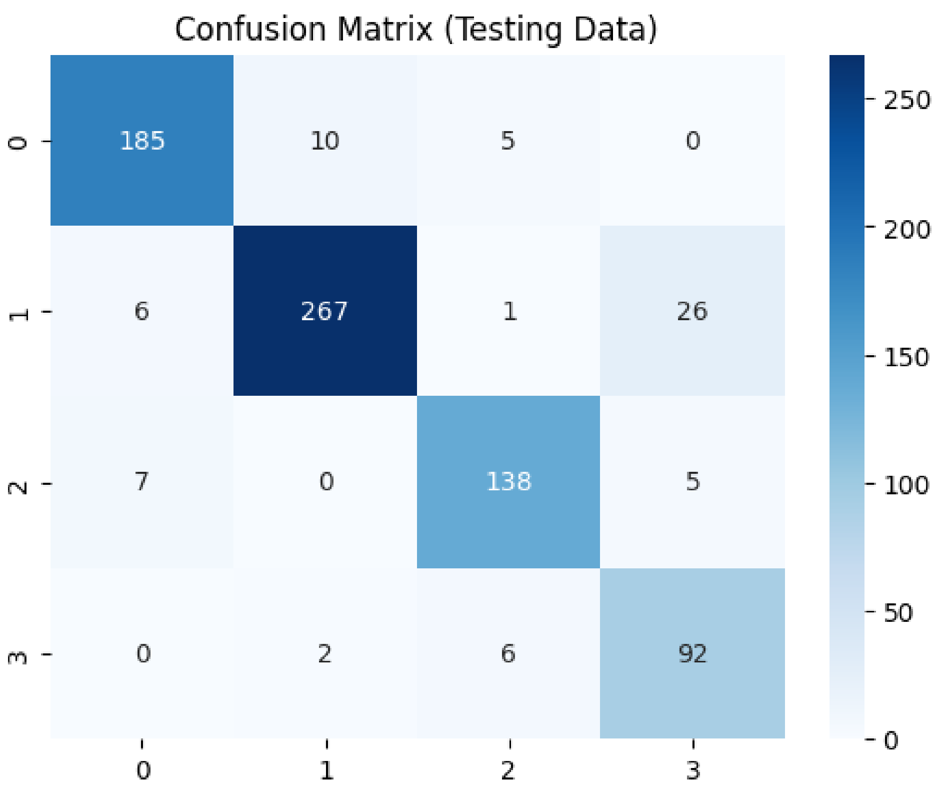
Newton-Raphson method



Confusion Matrix

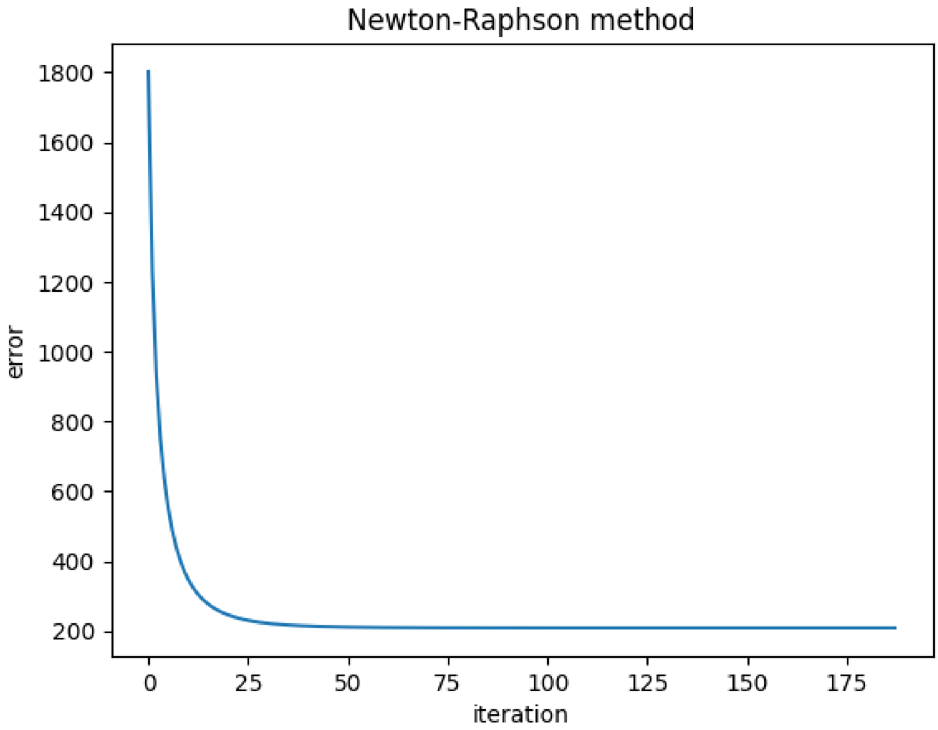


Accuracy (Training Data): 94.23076923076923

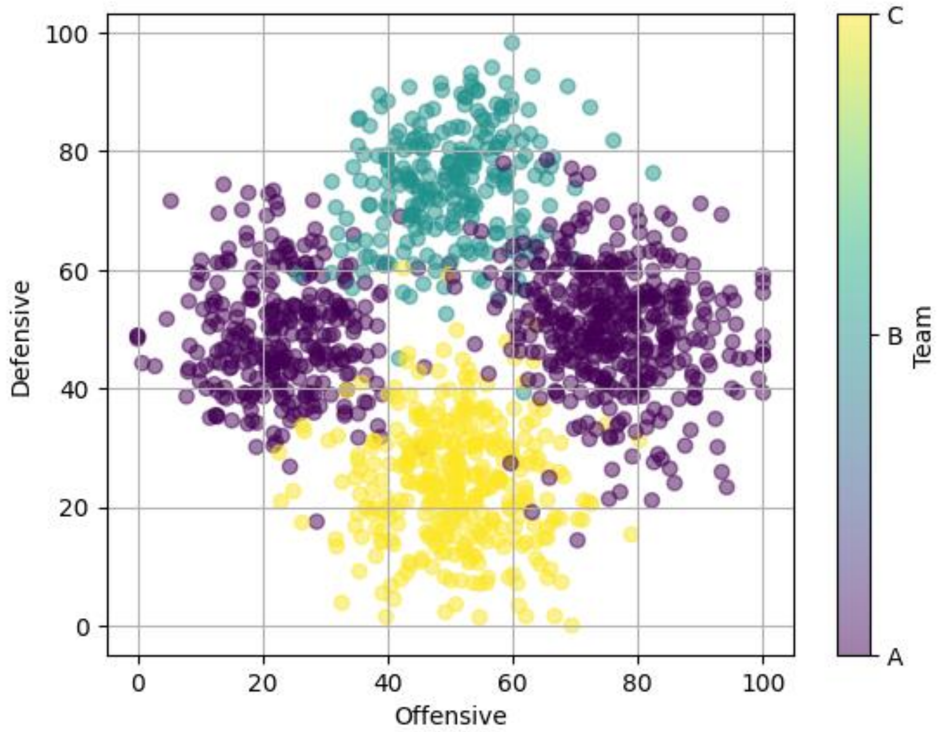


Accuracy (Testing Data): 90.93333333333334

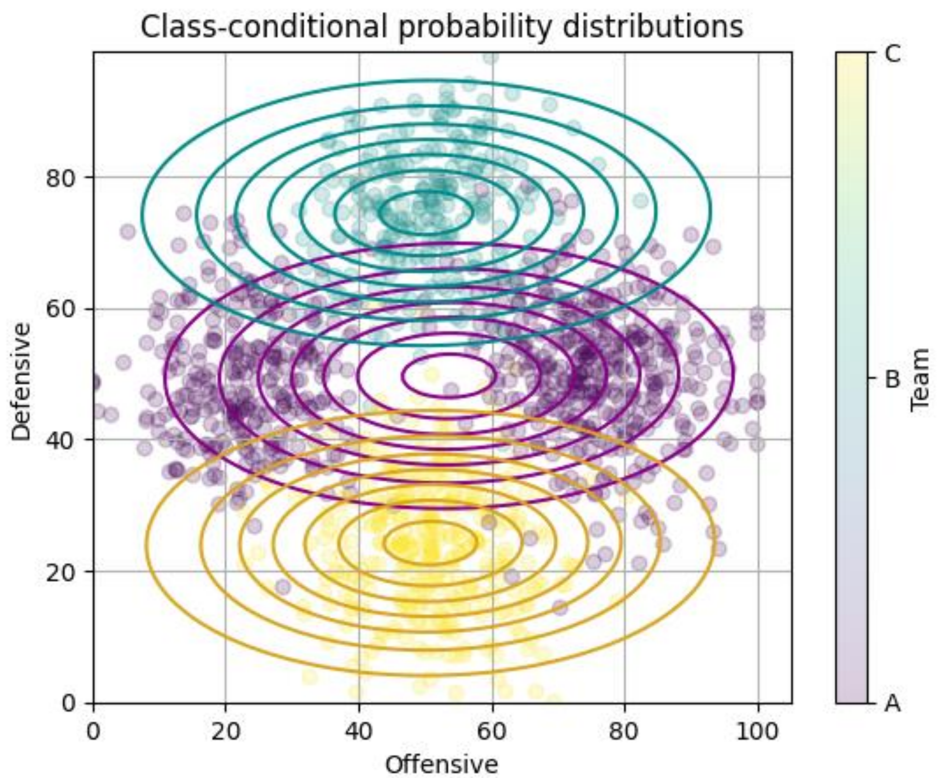
Cross-entropy error function

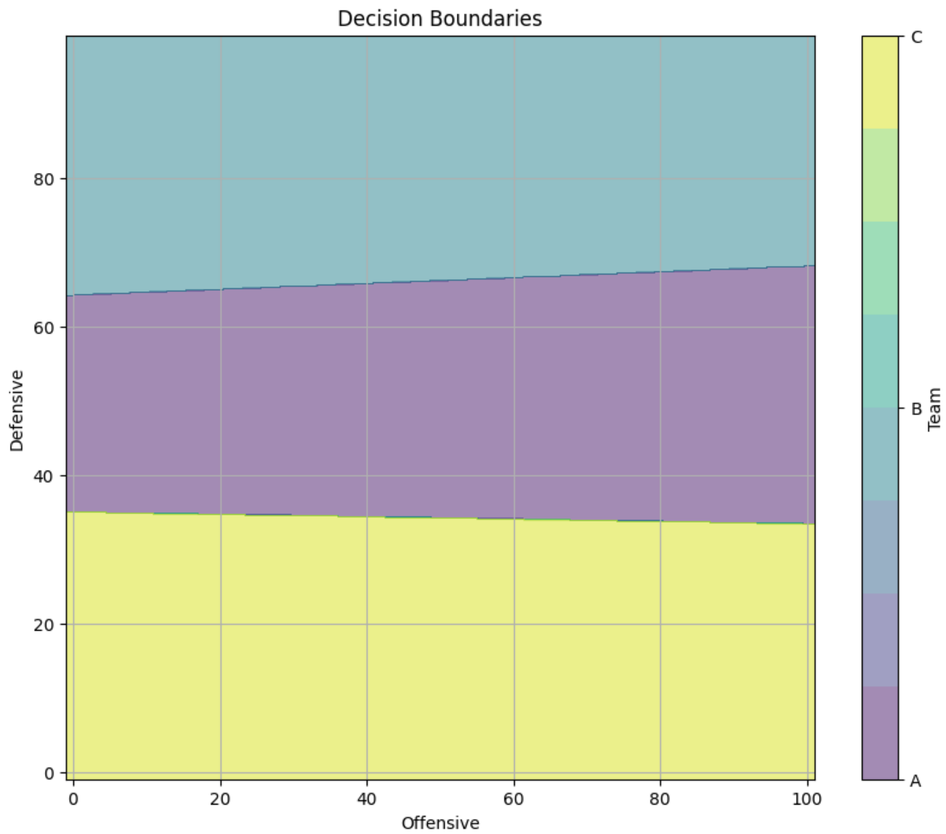


Part II

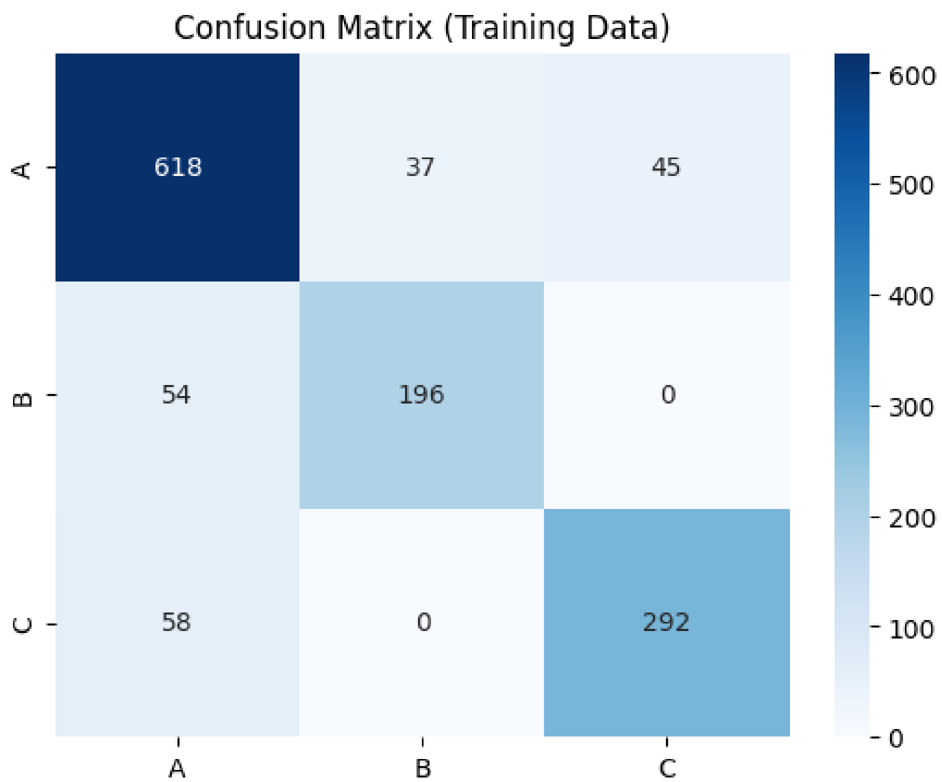


Generative Model

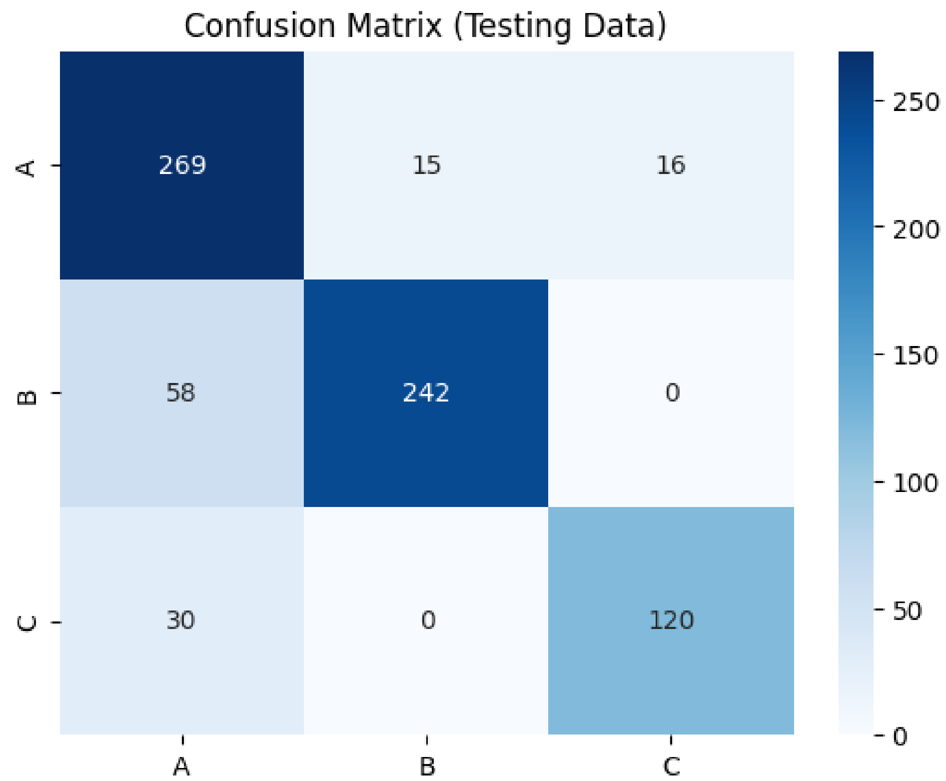




Confusion Matrix



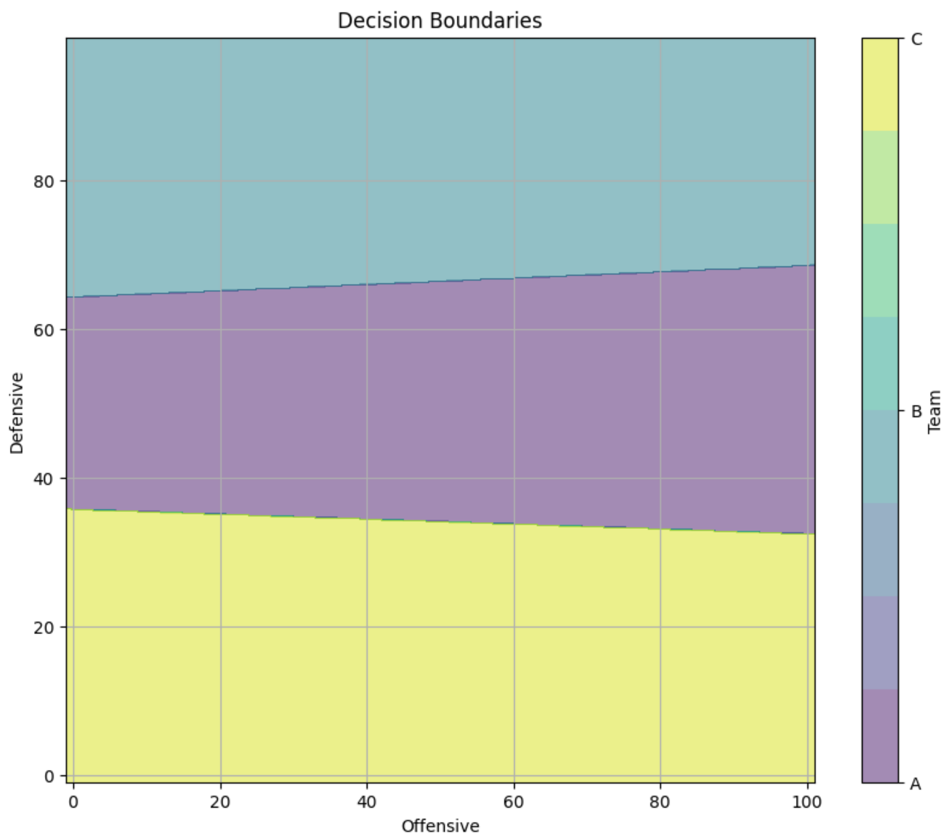
Accuracy (Training Data): 85.07692307692308



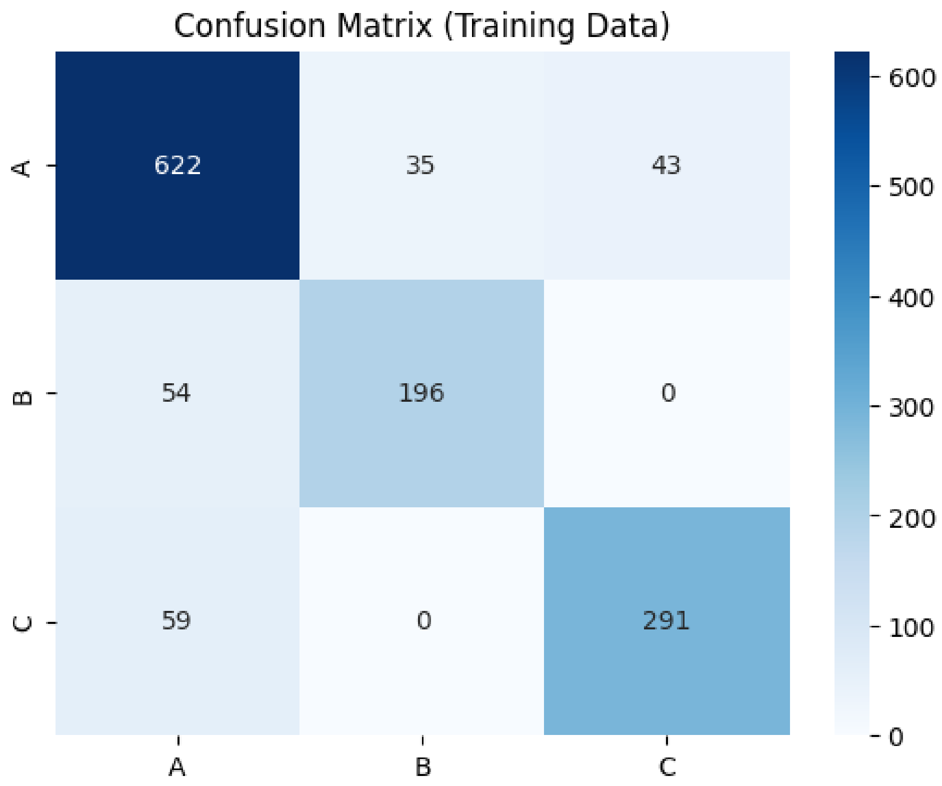

```
Accuracy (Testing Data): 84.13333333333334
```

Discriminative Model

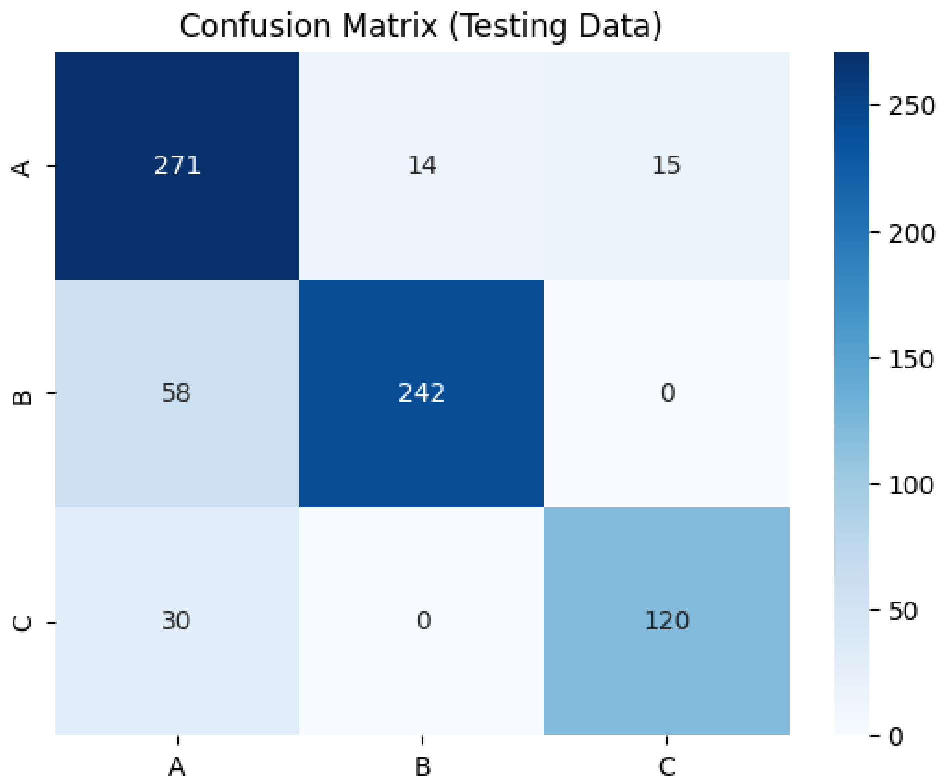
Gradient Descent method



Confusion Matrix

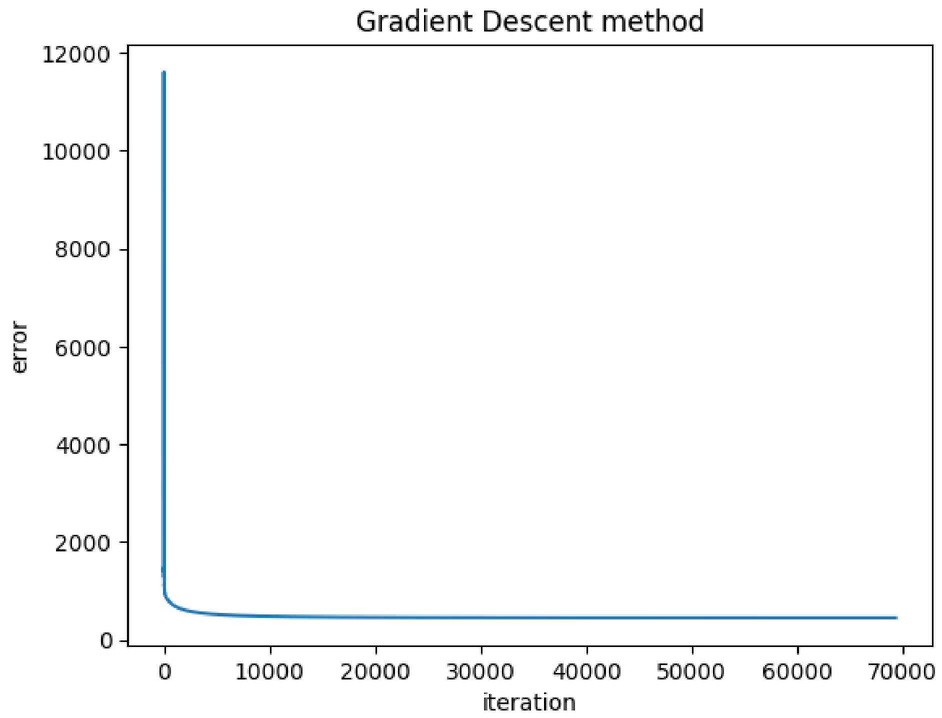


`Accuracy (Training Data): 85.3076923076923`

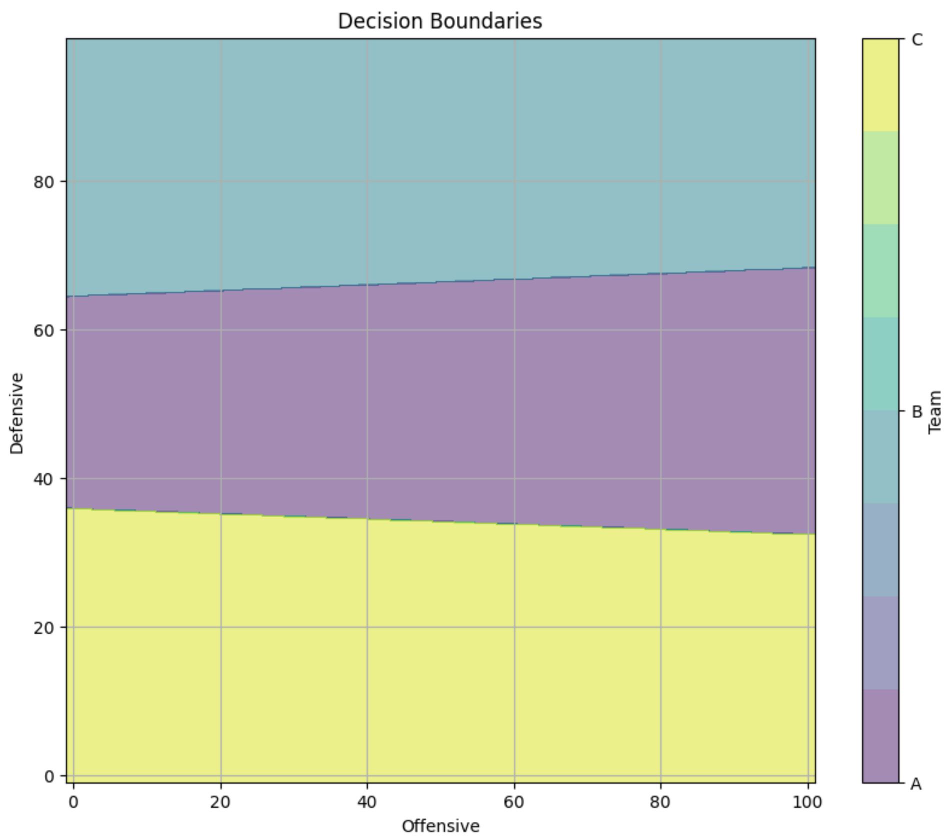


`Accuracy (Testing Data): 84.4`

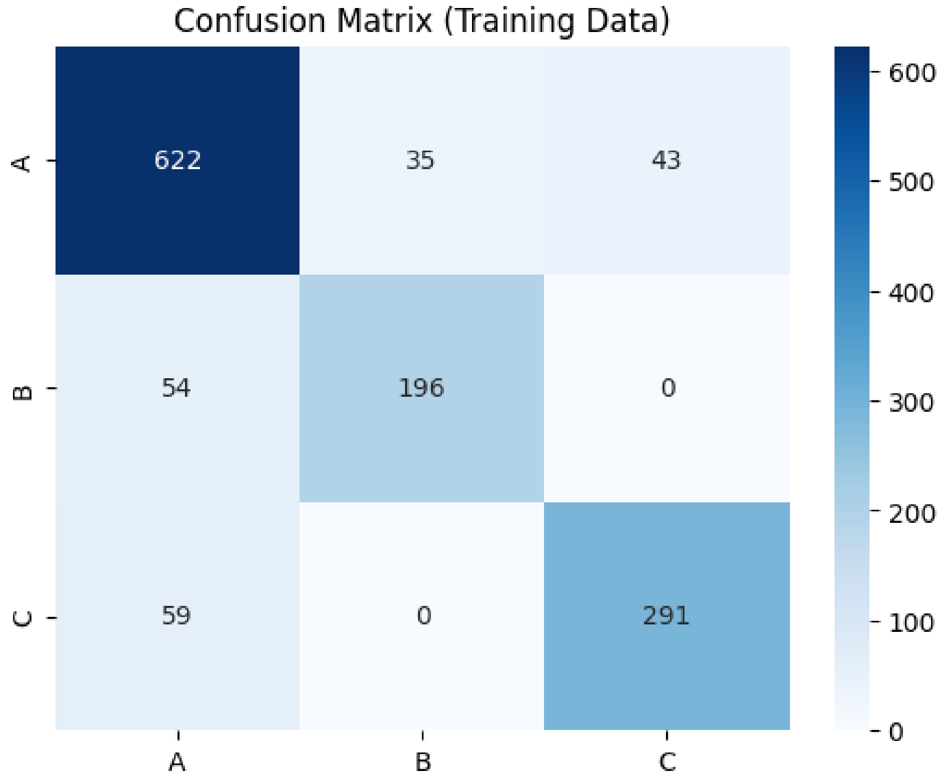
Cross-entropy error function



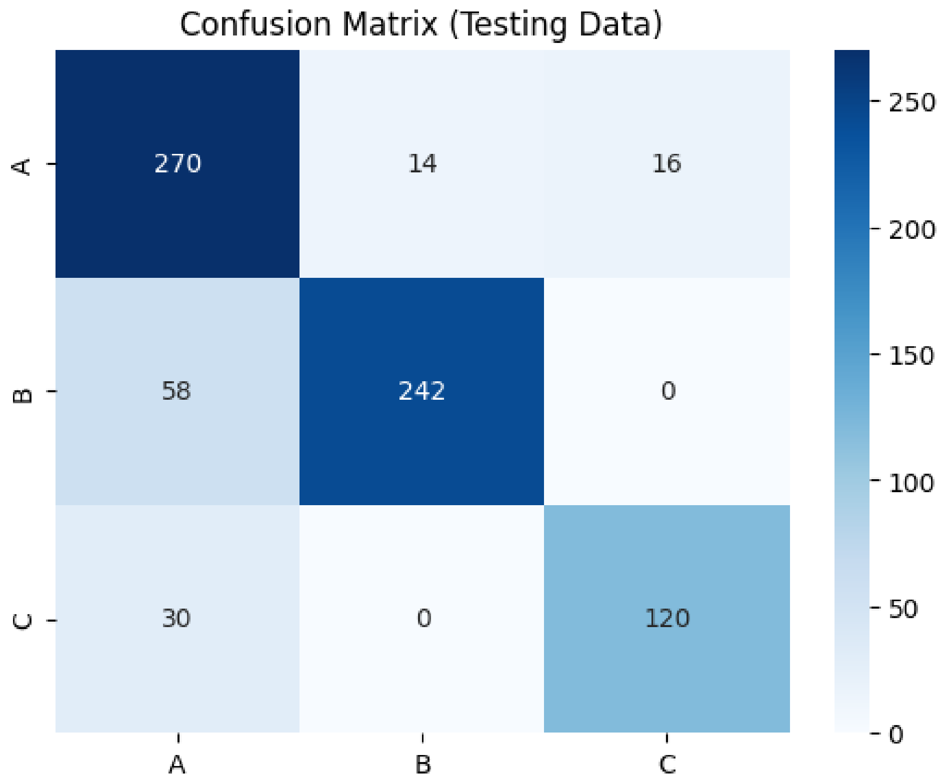
Newton-Raphson method



Confusion Matrix

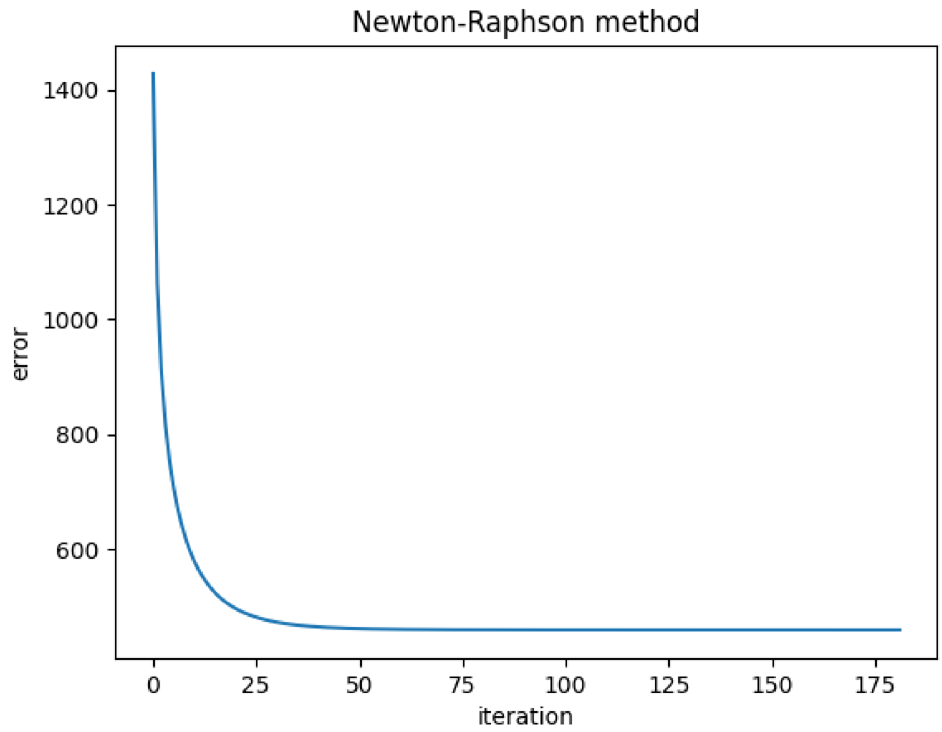


`Accuracy (Training Data): 85.3076923076923`



`Accuracy (Testing Data): 84.26666666666667`

Cross-entropy error function



Discussion

Aspect	Generative model	Discriminative model
Objective	Learn the joint probability distribution $p(x, C_k)$ then evaluate $p(C_k\|x)$	Directly learn the conditional probability distribution $p(C_k\|x)$
Focus	Models how data is generated from each class	Models the decision boundary between classes
Data Assumption	Requires assumptions about the data distribution	Does not require assumptions about the data distribution

Aspect	Gradient Descent method	Newton-Raphson method
Type	Iterative Optimization Method	
Update Rule	Adjusts parameters in the direction of steepest descent (negative gradient)	Uses second-order derivative information to adjust parameters
Convergence	Slower	Faster (due to use of second-order information)
Complexity	Simple to implement	More complex due to the computation of the Hessian matrix

In Part I, the generative model achieves slightly higher accuracy than the discriminative model on both the training and testing datasets. Generative models make assumptions about the underlying distribution of the data and model the joint probability of features and labels, which might be beneficial when the assumptions hold true. In this case, assuming Gaussian distributions might have captured the data's structure effectively. Discriminative models, on the other hand, directly model the decision boundary between classes. They don't make assumptions about the data distribution and can sometimes lead to better performance, especially with complex or high-dimensional data.

In Part II, the discriminative model achieves slightly higher accuracy than the generative model on both the training and testing datasets. Part II involves a different classification task where the teams are grouped based on overall performance ratings rather than being classified individually. Discriminative models might be more effective in capturing complex decision boundaries and relationships between features and labels, especially when the classes are not easily separable. The discriminative model might be better suited for this task because it directly models the decision boundary between the classes without making assumptions about the underlying data distribution.

The accuracy of both models dropped slightly in Part II compared to Part I. This could be due to the change in the distribution of data points after modifying the team labels (size of data points for class A [Team 0 and 3 in Part I] is much larger than the size of others (class B and class C)).