Scuola di Ingegneria
Corso di Laurea triennale in Ingegneria Informatica

Intelligenza Artificiale

Esame 12/06/2023

Fouepe Dylan
6333669

1-Introduzione

La classificazione binaria si riferisce a quei compiti di classificazione che hanno due class label. In genere, i compiti di classificazione binaria prevedono una classe che rappresenta lo stato normale e un'altra che rappresenta lo stato anormale. Ad esempio, se vogliamo creare un rilevatore di spam in e-mail, "non spam" è lo stato normale e "spam" è lo stato anormale. Un altro esempio è "cancro non rilevato", lo stato normale di un compito che prevede un test medico, e "cancro rilevato", lo stato anormale. Alla classe per lo stato normale viene assegnata l'etichetta 0, mentre alla classe con lo stato anormale viene assegnata l'etichetta 1. Un algoritmo che può essere utilizzato per la classificazione binaria è il multi-layer perceptron con l'algoritmo di backpropagation per il calcolo del gradiente.
In questo esperimento si è implementato l'architettura del modello e applicato il metodo del gradiente stocastico per l'aggiornamento dei pesi del modello. Le classi 3 e 8 del dataset Fashion-MINIST sono utilizzate per addestrare e il testare il modello.

3-Dataset

Fashion-MNIST è un dataset di immagini di articoli di Zalando, composto da un training set di 60.000 esempi e da un test set di 10.000 esempi. Ogni esempio è un'immagine in scala di grigi 28x28, associata a un'etichetta di 10 classi.
Ogni esempio di training e test è assegnato a una delle seguenti etichette:

Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

4-Codice

Il codice è scritto in Python. I seguenti passi sintetizzano il lavoro che svolge:

1. Carica i dati di training dei file ***train-labels-idx1-ubyte.gz*** in X_train e ***train-images-idx3-ubyte.gz*** in y_train e di test dei file ***t10k-images-idx3-ubyte.gz*** in X_test e ***t10k--labels-idx1-ubyte.gz*** in y_test. Quindi:
    - X_train: contiene gli esempi del campione di training.
    - y_train: contiene la classe di appartenenza di ogni campione di training.
    - X_test: contiene gli esempi del campione di test.
    - y_test: contiene la classe di appartenenza di ogni campione di test.
2. preparare i dati
3. selezionare le classe da addestrare
4. costruito il modello
5. creato il batch con di dati da addestrare
6. addestrato il modello
7. visualizzato i valori della loss per ogni iterazione

5-Loss rispetto numero iteration e grandezza del batch

batch_size è un iper-parametro che deve settato all'inizio nella variabile omonima

5-1 batch_size = 10

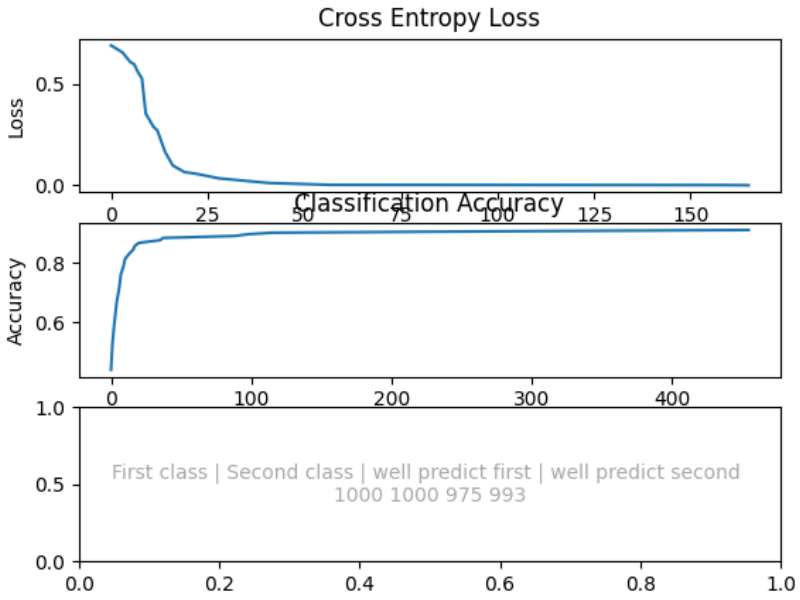

Figura 1: Loss sul training set in funzione del numero di iterazione

5-2 batch_size = 25

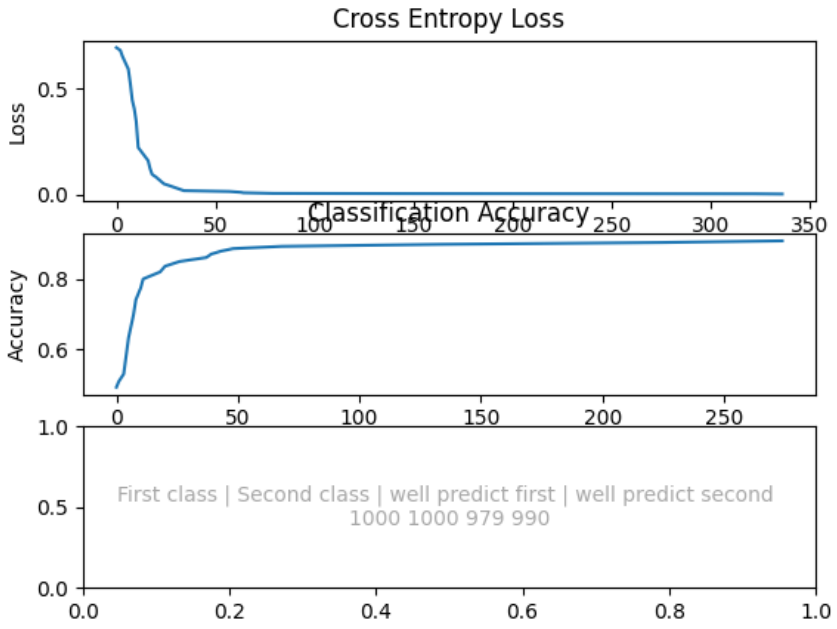

Figura 2: Loss sul training set in funzione del numero di iterazione

5-3 batch_size = 50

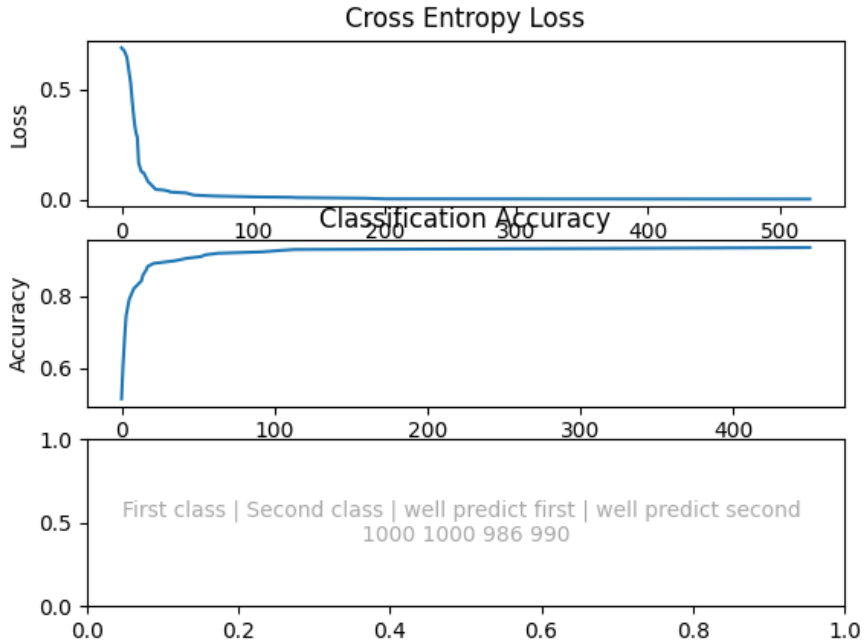

Figura 3: Loss sul training set in funzione del numero di iterazione

6-Utilizzo del codice

Il file principale contenente il codice è main.py. Per eseguirlo si deve scaricare il dataset e salvare i file in una cartella con nome ‘data’ nella stessa directory di ‘main.py’. Dopo avere eseguito il codice viene visualizzato il grafo che valuta la loss e l’accuratezza sul validation set in funzione del numero di iterazione e il numero di esempi per ogni classe nel test set con quelli classificati correttamente dal modello e salvati i risultati nel file ‘train_loss.pkl’ e test_accuracy.pkl.