Γιαννακοπούλου Σταυρούλα - 3220027

Σταμαδιάνου Μαρία - 3220194

Αναφορά 2ης εργασίας Τεχνητής Νοημοσύνης:

Μέρος Α:

Από τους διαθέσιμους αλγορίθμους μάθησης, για κατάταξη κειμένων θετικών και αρνητικών κριτικών του imdb dataset, επιλέχθηκαν για υλοποίηση ο Naïve Bayes στην πολυμεταβλητή μορφή Bernoulli και η Λογιστική Παλινδρόμηση. Κάθε κείμενο παριστάνεται με ένα διάνυσμα ιδιοτήτων με τιμές 0 και 1 που δείχνει ποιες λέξεις ενός λεξιλογίου περιέχονται στο κείμενο που εξετάζεται κάθε φορά και παραλείπονται οι n πιο συχνές, οι k πιο σπάνιες ενώ επιλέγονται οι m με το υψηλότερο πληροφοριακό κέρδος. Από τα training data αντλούμε ένα ποσοστό της τάξης του 20% και τα χρησιμοποιούμε ως development data.

Bernoulli Naïve Bayes:

Κατόπιν διαφόρων δοκιμών για τιμές n,k,m καταλήξαμε πειραματικά στις τιμές n = 200, k = 50 και m = 3000 ως ένα set τιμών που δίνουν ικανοποιητικά αποτελέσματα και για τις δύο υλοποιήσεις. Επιλέξαμε να εξετάσουμε τα αρνητικά δεδομένα εκπαίδευσης και ανάπτυξης.

Logistic Regression:

Επιλέξαμε να εξετάσουμε τα αρνητικά δεδομένα εκπαίδευσης και ανάπτυξης και πειραματιστήκαμε με τιμές που θα δείτε σε αντίστοιχο πίνακα παρακάτω

Μέρος Β:

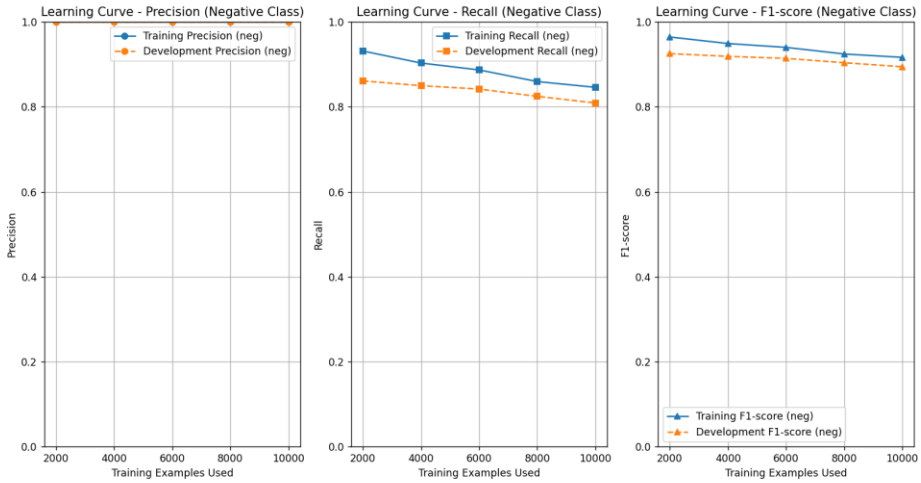
Για αρχή, θα συγκρίνουμε τη δική μας υλοποίηση του Naïve Bayes και την έτοιμη υλοποίηση της βιβλιοθήκης sklearn. Φροντίσαμε ώστε οι τιμές n,m,k να είναι κατά το δυνατό πιο κοντά σε αυτές που επιλέξαμε για να ελέγξουμε τη λειτουργία της δικής μας υλοποίησης για τον naïve bayes κώδικα.

Παρακάτω, παραθέτουμε τις καμπύλες και τους πίνακες που προκύπτουν σε κάθε περίπτωση:

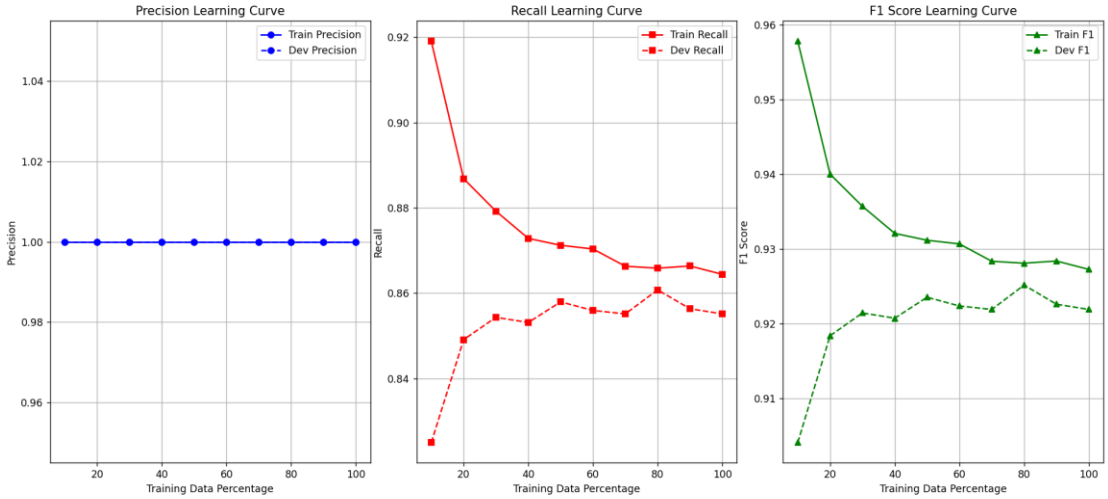
Για τον δικό μας κώδικα Naïve Bayes έχουμε πίνακα:

```
Enter n, k, m values: 200 50 3000
Test Set Evaluation:

              precision    recall  f1-score   support
-------------------------------------------------------
Negative       0.85      0.82      0.84      12500
Positive       0.83      0.86      0.84      12500

accuracy                           0.84      25000
macro avg      0.84      0.84      0.84      25000
weighted avg   0.84      0.84      0.84      25000
```

Και καμπύλες:



Για τον κώδικα με την έτοιμη υλοποίηση έχω τις καμπύλες:



Και τον πίνακα:

```
Test Set Classification Report:

              precision    recall  f1-score   support

    Negative       0.80      0.86      0.83     12500
    Positive       0.85      0.79      0.82     12500

    accuracy                           0.82     25000
   macro avg       0.83      0.82      0.82     25000
weighted avg       0.83      0.82      0.82     25000
```

Από τη σύγκριση των δύο διαγραμμάτων παρατηρούμε:

1. Precision
    - Και στις δυο υλοποιήσεις, η precision είναι σταθερή και πολύ κοντά στο 1.0 γεγονός που αποδεικνύει ότι και οι δυο έχουν μια σταθερότητα και ικανοποιητικά αποτελέσματα
2. Recall
    - Η έτοιμη υλοποίηση έχει recall που ξεκινά υψηλά (κοντά στο 0.92) και μειώνεται σταδιακά.
    - Στη χειροποίητη υλοποίηση, το recall ξεκινά επίσης από υψηλή τιμή και μειώνεται σταδιακά, όμως φαίνεται να ακολουθεί πιο σταθερή πορεία.
    - Και στις δύο περιπτώσεις, το recall μειώνεται με περισσότερα δεδομένα, κάτι που μπορεί να υποδηλώνει overfitting
3. F1-score
    - Η έτοιμη υλοποίηση ξεκινά από πολύ υψηλό F1-score (κοντά στο 0.96) και μειώνεται σταδιακά.
    - Η χειροποίητη υλοποίηση έχει επίσης υψηλό F1-score, αλλά δείχνει μικρότερη πτώση.
    - Το γεγονός ότι η χειροποίητη υλοποίηση έχει σταθερότερη συμπεριφορά μπορεί να σημαίνει ότι έχει καλύτερη γενίκευση στα validation δεδομένα.

Συμπέρασμα: Η έτοιμη υλοποίηση διατηρεί εξαιρετικά υψηλή precision και F1-score, αλλά το recall μειώνεται αισθητά. Η χειροποίητη υλοποίηση έχει πιο φυσιολογική καμπύλη μάθησης όμως.

Ακολουθούν πίνακες με σύγκριση τιμών μεταξύ της έτοιμης και της δικής μας υλοποίησης για τα αρνητικά δεδομένα εκπαίδευσης και τα αρνητικά δεδομένα ανάπτυξης:

Ανάλυση Negative

	Precision	**Recall**	**F1-score**
Χειροποίητη	0.85	0.82	0.84
Έτοιμη	0.80	0.86	0.83

- Η χειροποίητη υλοποίηση έχει καλύτερη precision.
- Η έτοιμη υλοποίηση έχει καλύτερη recall.
- Το F1-score είναι ελαφρώς καλύτερο στη χειροποίητη (0.84 έναντι 0.83).

Ανάλυση Positive

	Precision	**Recall**	**F1-score**
Χειροποίητη	0.83	0.86	0.84
Έτοιμη	0.85	0.79	0.82

- Η έτοιμη υλοποίηση έχει καλύτερη precision.
- Η χειροποίητη υλοποίηση έχει καλύτερη recall.
- Το F1-score είναι καλύτερο στη χειροποίητη.

Συνολικά Μέτρα

	Accuracy	**Macro Avg Precision**	**Macro Avg Recall**	**Macro Avg F1**
Χειροποίητη	0.84	0.84	0.84	0.84
Έτοιμη	0.82	0.83	0.82	0.82

- Η χειροποίητη υλοποίηση υπερτερεί σε όλα τα συνολικά μέτρα, με accuracy 0.84 αντί για 0.82 και macro averages 0.84 αντί για 0.82-0.83.

Συμπέρασμα:Η χειροποίητη υλοποίηση αποδίδει καλύτερα συνολικά, με υψηλότερο accuracy, macro avg precision, recall και F1-score. Από την άλλη, η έτοιμη υλοποίηση έχει υψηλότερη recall στην κατηγορία Negative, αλλά χάνει recall στην κατηγορία Positive, κάτι που οδηγεί σε χαμηλότερο F1-score.

Έπειτα, συγκρίνουμε τη δική μας υλοποίηση Logistic Regression με την έτοιμη υλοποίηση της βιβλιοθήκης sklearn. Φροντίσαμε ώστε οι τιμές n,m,k να είναι κατά το δυνατό πιο κοντά σε αυτές που επιλέξαμε για να ελέγξουμε τη λειτουργία της δικής μας υλοποίησης και προκύπτει ο ακόλουθος πίνακας δεδομένων:

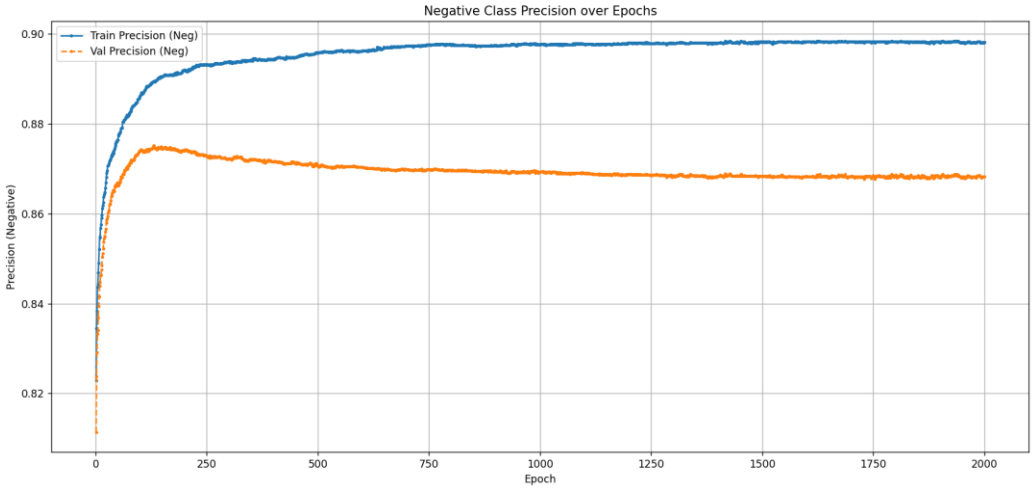
Values	*Precision for A) and B) Parts*
N=200, M=250, K=4000	A) 0.8625 B) 0.83
N=100, M=50, K=5000	A) 0.87 B) 0.84
N=200, M=50, K=5000	A) 0.87 B) 0.84
N=200, M=150, K=5000	A) 0.86 B) 0.84

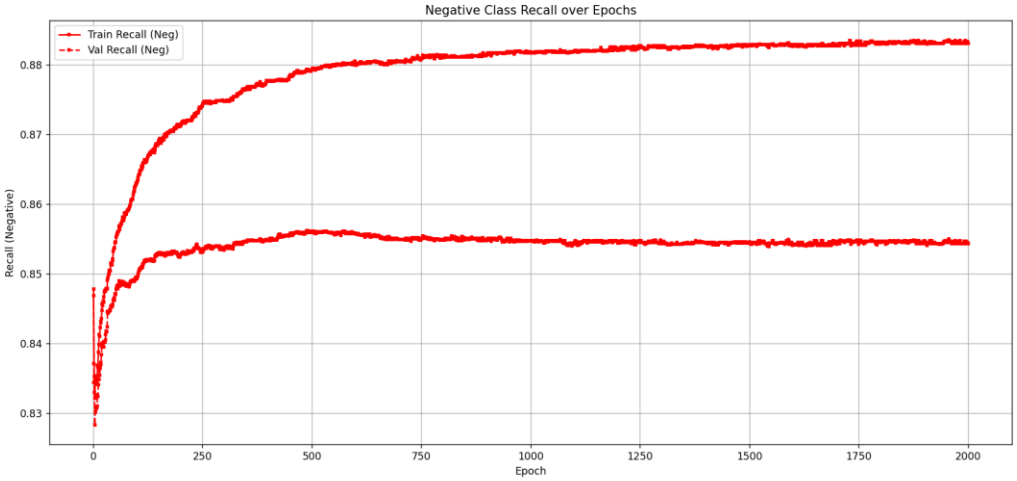
Όπως γίνεται εύκολα αντιληπτό από τον παραπάνω πίνακα, κατά την αναζήτηση τιμών για βέλτιστη απόδοση, κινηθήκαμε στις τιμές 4000 και 5000 για το m, κυρίως 200 για το n και με k=50 πήραμε τα καλύτερα αποτελέσματα και για τους δυο κώδικες. Μπορούμε λοιπόν εύκολα να συμπεράνουμε ότι παίρνουμε παρόμοιες αποδόσεις για τους δυο κώδικες όταν τους δίνουμε τις ίδιες τιμές.

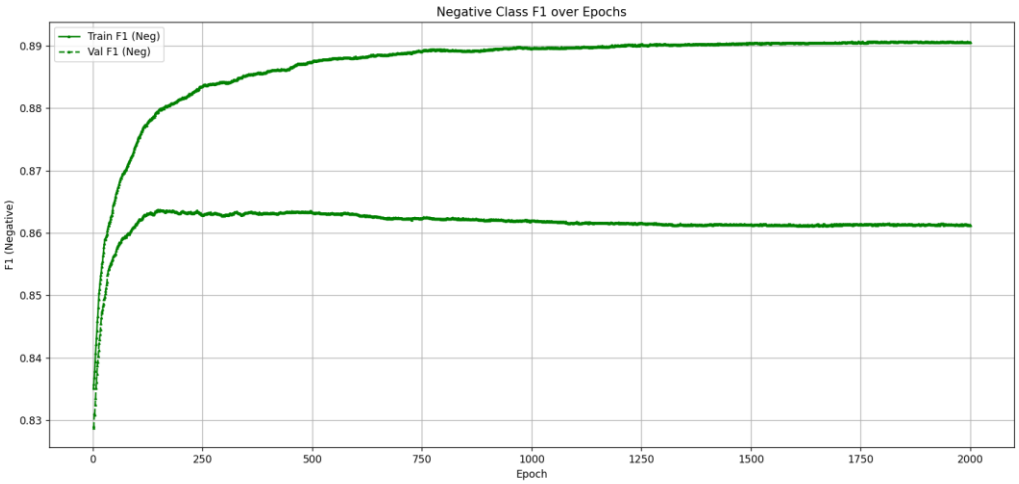
Παρακάτω βλέπουμε τον πίνακα της δικής μας υλοποίησης logistic regression:

```
Enter n, k, m values: 200 250 4000

              precision    recall  f1-score   support
-------------------------------------------------------
Negative       0.87      0.85    0.86     12500
Positive       0.86      0.87    0.86     12500

accuracy                           0.86     25000
macro avg      0.86      0.86    0.86     25000
weighted avg   0.86      0.86    0.86     25000
```

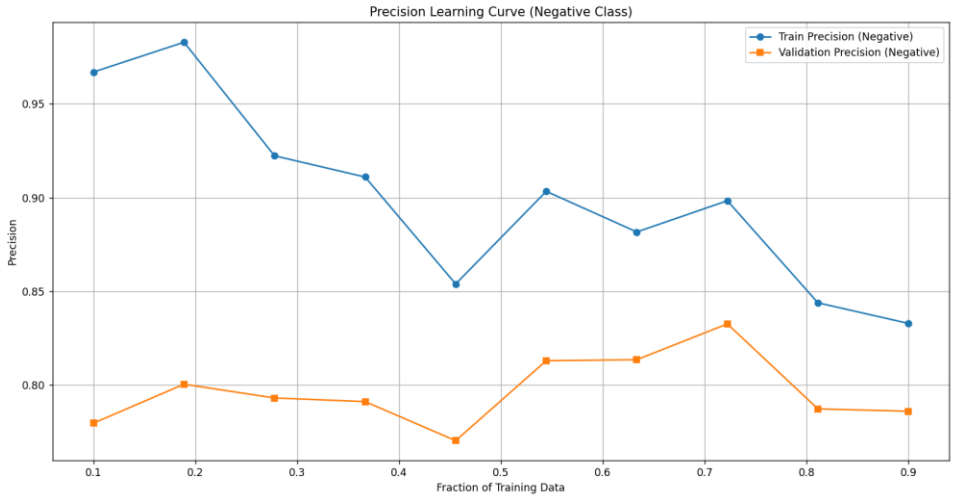
Και οι καμπύλες της δικής μας υλοποίησης logistic regression φαίνονται παρακάτω:

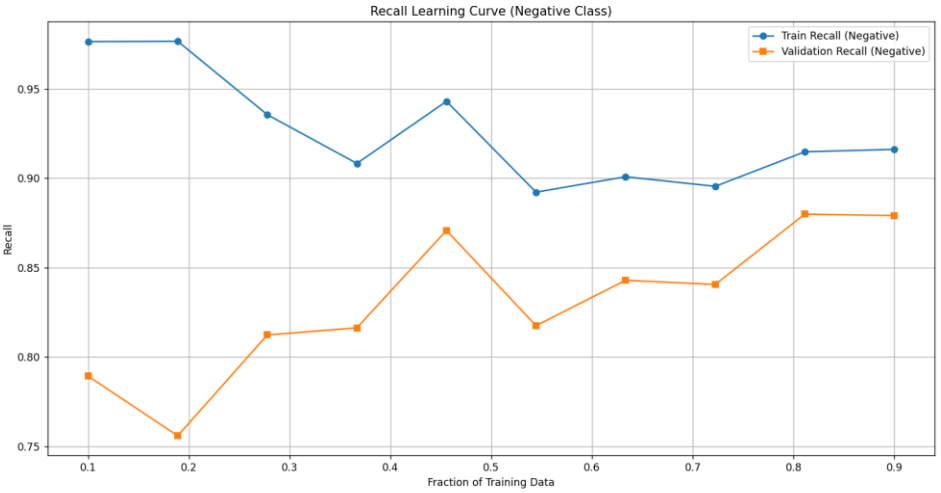


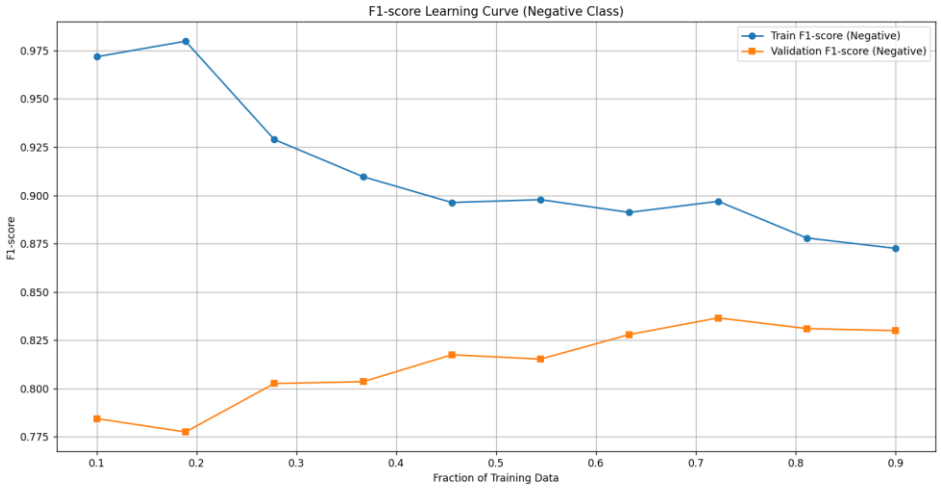




Για την έτοιμη υλοποίηση logistic regression έχουμε τις παρακάτω καμπύλες:







Ακολουθεί πίνακας για τον έτοιμο κώδικα logistic regression:

```
Enter n, k, m values: 200 250 4000
              precision    recall  f1-score   support
-----------------------------------------------------
Negative       0.82      0.85      0.84     12500
Positive       0.85      0.81      0.83     12500

accuracy                           0.83     25000
macro avg      0.83      0.83      0.83     25000
weighted avg   0.83      0.83      0.83     25000
```

Ανάλυση Negative

	Precision	**Recall**	**F1-score**
Έτοιμη	0.82	0.85	0.84
Χειροποίητη	0.87	0.85	0.86

- Η χειροποίητη υλοποίηση έχει καλύτερη precision.
- Η recall είναι ίδια (0.85).
- Η χειροποίητη έχει καλύτερο F1-score.

Ανάλυση Κατηγορίας: Positive

	Precision	**Recall**	**F1-score**
Έτοιμη	0.85	0.81	0.83
Χειροποίητη	0.86	0.87	0.86

- Η χειροποίητη υλοποίηση έχει ελαφρώς καλύτερη precision (0.86 έναντι 0.85).
- Η χειροποίητη υλοποίηση έχει σημαντικά καλύτερη recall (0.87 έναντι 0.81).
- Το F1-score είναι επίσης καλύτερο από αυτό της έτοιμης υλοποίησης.

Συνολικά Μέτρα

	Accuracy	**Macro Avg Precision**	**Macro Avg Recall**	**Macro Avg F1**
Έτοιμη	0.83	0.83	0.83	0.83
Χειροποίητη	0.86	0.86	0.86	0.86

- Η χειροποίητη υλοποίηση υπερτερεί σε όλα τα συνολικά μέτρα, με accuracy 0.86 αντί για 0.83 και macro averages 0.86 αντί για 0.83.

Συμπέρασμα:Η χειροποίητη υλοποίηση Logistic Regression αποδίδει καλύτερα συνολικά, με υψηλότερο accuracy, macro avg precision, recall και F1-score.Η έτοιμη υλοποίηση έχει χαμηλότερη recall στην κατηγορία Positive, γεγονός που οδηγεί σε χαμηλότερο F1-score.

Παρακάτω συγκρίνουμε τα διαγράμματα της δικής μας υλοποίησης logistic regression:

Precision της δικής μας υλοποίησης για αρνητικά δεδομένα εκπαίδευσης και ανάπτυξης

Παρατηρήσεις:

- Το training precision ξεκινά χαμηλά αλλά ανεβαίνει γρήγορα, πλησιάζοντας το 0.90 στις 2000 εποχές.
- Το validation precision σταθεροποιείται σε σχετικά υψηλή τιμή μετά από τις 500 εποχές.
- Φαίνεται μικρή απόσταση μεταξύ training και validation precision, γεγονός που δείχνει ότι υπάρχει κάποια διαφορά στην απόδοση μεταξύ των δύο συνόλων

Precision της έτοιμης υλοποίησης για αρνητικά δεδομένα εκπαίδευσης και ανάπτυξης

Παρατηρήσεις:

- Το training precision ξεκινά πολύ υψηλά (πάνω από 0.95) αλλά μειώνεται σταδιακά όσο αυξάνεται το μέγεθος των δεδομένων.
- Το validation precision είναι σημαντικά χαμηλότερο (γύρω στο 0.78-0.80) και παραμένει σχετικά σταθερό ανεξάρτητα από το μέγεθος των δεδομένων εκπαίδευσης.
- Υπάρχει μεγάλη διαφορά μεταξύ training και validation precision, κάτι που δείχνει πιθανό overfitting: το μοντέλο αποδίδει πολύ καλά στα training data αλλά όχι εξίσου καλά στα validation data.

Χαρακτηριστικό	Χειροποίητη Υλοποίηση	Έτοιμη Υλοποίηση
Training Precision	0.90	Από 0.95+ και πέφτει
Validation Precision	0.86	0.78-0.80 (αστάθμητο)

Η χειροποίητη υλοποίηση έχει καλύτερη γενίκευση, καθώς το training και validation precision είναι πιο κοντά ενώ η έτοιμη υλοποίηση παρουσιάζει έντονο overfitting, με πολύ υψηλό training precision αλλά σημαντικά χαμηλότερο validation precision.

Παρόμοια συμπεράσματα προκύπτουν αν εξετάσουμε τα αντίστοιχα διαγράμματα με καμπύλες μάθησης που παράγονται από τα αρνητικά δεδομένα εκπαίδευσης και ανάπτυξης για ανάκληση και f1.