**Αναφορά Εργασίας**

**Κατάταξη Κειμένων IMDB με Bernoulli Naive Bayes & BiLSTM**

Σε αυτή την εργασία, εξετάζουμε την πρόβλεψη της **συναισθηματικής πόλωσης** (θετικό/αρνητικό) κειμένων από το **IMDB dataset**. Το IMDB διαθέτει 50.000 κριτικές ταινιών: 25.000 για εκπαίδευση και 25.000 για αξιολόγηση (test). Εμείς χωρίζουμε το train περαιτέρω σε **train** (22.500) και **validation** (2.500). Στο τέλος, αξιολογούμε στο **test**.

Η εργασία χωρίζεται σε **τρία** μέρη:

1. **Μέρος Α**: Υλοποίηση Custom Bernoulli Naive Bayes, μελέτη καμπυλών μάθησης.
2. **Μέρος Β**: Αναζήτηση υπερπαραμέτρων (Laplace smoothing alpha), σύγκριση με scikit-learn, τελική αξιολόγηση στο test set.
3. **Μέρος Γ**: BiLSTM RNN με προ-εκπαιδευμένα embeddings (Word2Vec), εκπαίδευση και αξιολόγηση.

**Μέρος Α: Custom Bernoulli Naive Bayes & Καμπύλες Μάθησης**

**1. Προεπεξεργασία Δεδομένων**

* Χρησιμοποιούμε τη συνάρτηση:
* (x\_train, y\_train), (x\_test, y\_test) = imdb.load\_data(num\_words=10000, skip\_top=20)
* Μετατρέπουμε κάθε κριτική σε **0/1 Bag-of-Words** (μέγεθος 10000). Έτσι, αποκτούμε πίνακα διαστάσεων (25000, 10000) για τα δεδομένα εκπαίδευσης.
* Χωρίζουμε το train (25.000) σε **train (22.500)** και **validation (2.500)**:
* Complete data size: (25000, 10000) (25000,)
* Training data size: (22500, 10000) (22500,)
* Validation data size: (2500, 10000) (2500,)
* Test data size: (25000, 10000) (25000,)

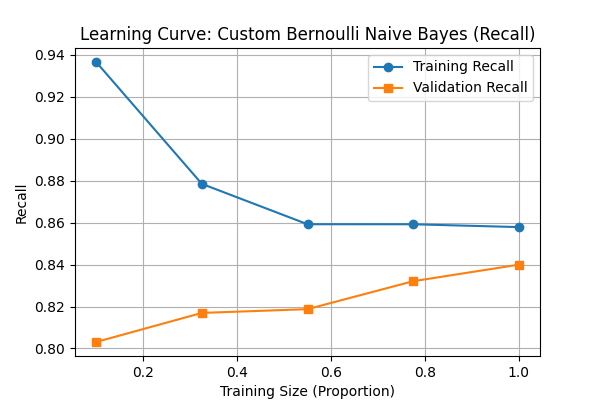
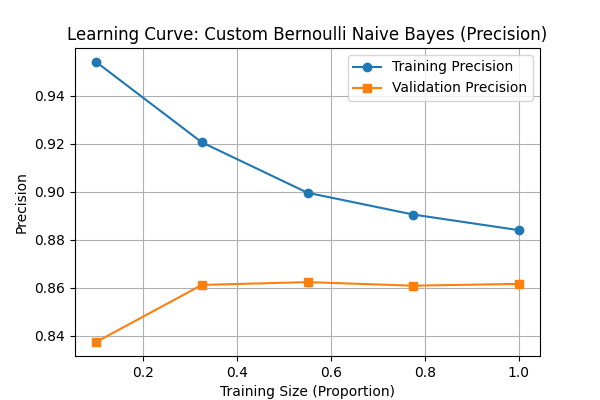
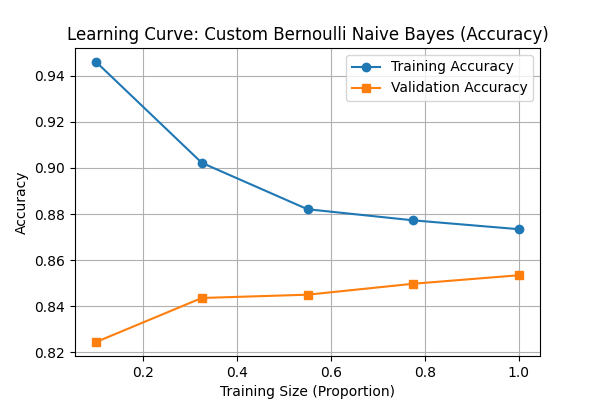
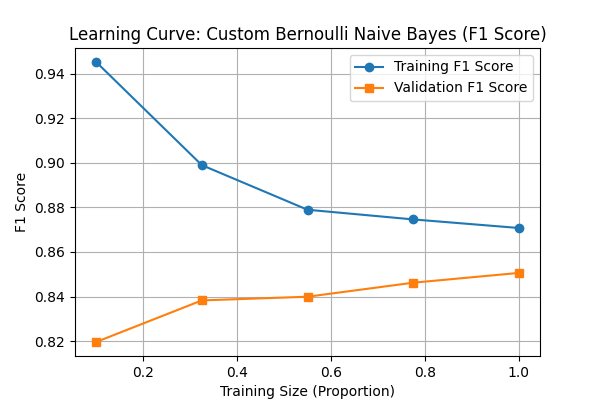
**2. Υλοποίηση Custom Bernoulli Naive Bayes**

Υλοποιείται κλάση BernoulliNaiveBayes:

1. **fit(X, y)**:
   * Υπολογίζει πιθανότητες εμφάνισης χαρακτηριστικών (λέξεων) ανά κλάση, με **Laplace smoothing** α\alpha.
   * Αποθηκεύει λογαριθμικές πιθανότητες.
2. **predict(X)**:
   * Για κάθε δείγμα, υπολογίζει το joint log-likelihood και επιλέγει την κλάση με το μέγιστο.

**3. Καμπύλες Μάθησης (Learning Curves)**

Για να ελέγξουμε τη συμπεριφορά μάθησης του Bernoulli NB, σχεδιάσαμε καμπύλες μάθησης για **F1, Accuracy, Precision, Recall**, συναρτήσει του ποσοστού εκπαίδευσης. Παραθέτουμε ενδεικτικά αποτελέσματα:

****  
  
Οι καμπύλες δείχνουν ότι το training score ξεκινά υψηλά (λίγα δεδομένα) και μειώνεται όσο αυξάνονται τα δεδομένα, ενώ το validation score ανεβαίνει/σταθεροποιείται, καταλήγοντας σε σχετικά καλή απόδοση.

**Μέρος Β: Αναζήτηση alpha, Σύγκριση με Scikit-Learn & Τελική Αξιολόγηση**

**1. Αρχική Εκπαίδευση σε Train, Αξιολόγηση σε Validation**

Με α=1.0\alpha=1.0 αρχικά:

Custom Bernoulli Naive Bayes Evaluation on Validation Set:

Accuracy: 0.8392

Precision: 0.8544

Recall: 0.8375

F1 Score: 0.8459

**Classification Report (Validation)**:

precision recall f1-score support

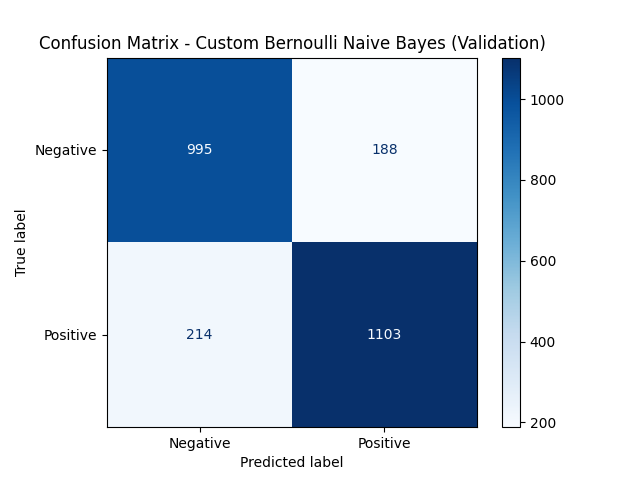
Negative 0.82 0.84 0.83 1183

Positive 0.85 0.84 0.85 1317

accuracy 0.84 2500

macro avg 0.84 0.84 0.84 2500

weighted avg 0.84 0.84 0.84 2500

**  
2. Αναζήτηση beta alpha**

Δοκιμάζουμε α∈{0.1,0.5,1.0,2.0,5.0}\alpha\in\{0.1, 0.5, 1.0, 2.0, 5.0\}:

Alpha: 0.1, Accuracy: 0.8396, Precision: 0.8550, Recall: 0.8375, F1: 0.8462

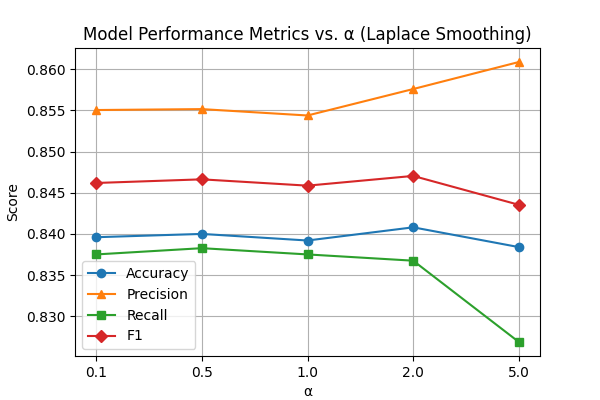
Alpha: 0.5, Accuracy: 0.8400, Precision: 0.8552, Recall: 0.8383, F1: 0.8466

Alpha: 1.0, Accuracy: 0.8392, Precision: 0.8544, Recall: 0.8375, F1: 0.8459

Alpha: 2.0, Accuracy: 0.8408, Precision: 0.8576, Recall: 0.8368, F1: 0.8470

Alpha: 5.0, Accuracy: 0.8384, Precision: 0.8609, Recall: 0.8269, F1: 0.8435

Καλύτερη τιμή: **α=2.0\alpha=2.0** (F1=0.8470).

**  
3. Σύγκριση με Scikit-Learn BernoulliNB**

Ορίζοντας BernoulliNB(alpha=2.0), έχουμε:

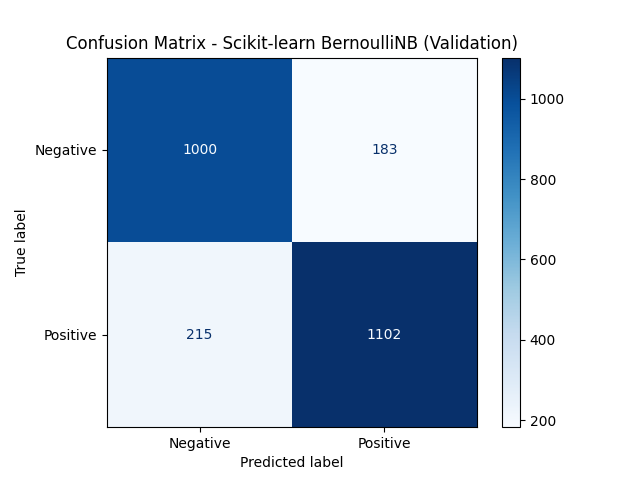
Scikit-learn BernoulliNB Evaluation on Validation Set (alpha=2.0):

Accuracy : 0.8408

Precision: 0.8576

Recall : 0.8368

F1 Score : 0.8470

**  
4. Τελική Εκπαίδευση (Train+Val) και Αξιολόγηση σε Test Set**

Συγχωνεύουμε train+val και εκπαιδεύουμε εκ νέου, με α=2.0\alpha=2.0:

Test metrics (Custom NB):

Accuracy : 0.8418

Precision: 0.8621

Recall : 0.8138

F1 Score : 0.8373

**Classification Report (Test)**:

precision recall f1-score support

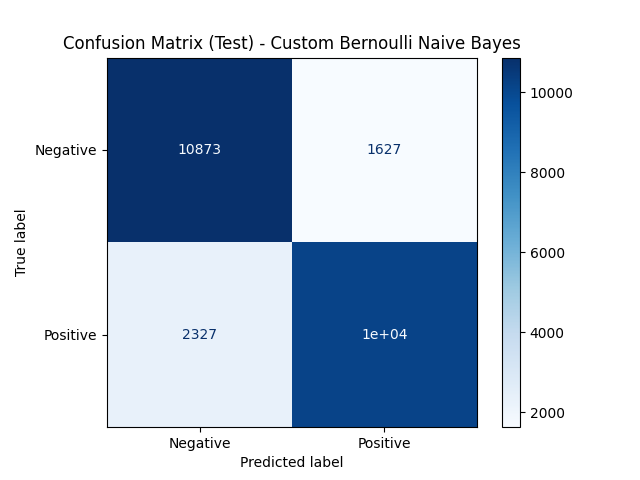
Negative 0.82 0.87 0.85 12500

Positive 0.86 0.81 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



**Μέρος Γ: BiLSTM RNN με Προ-εκπαιδευμένα Word2Vec Embeddings**

**1. Προεπεξεργασία ως Ακολουθίες**

* Χρησιμοποιούμε ξανά τα δεδομένα IMDB.
* Χωρίζουμε σε train/val/test (22500/2500/25000).
* Εφαρμόζουμε pad\_sequences(..., maxlen=200) για σταθερό μήκος.

**2. Εκπαίδευση με Word2Vec**

* Κατεβάζουμε τα **GoogleNews-vectors-negative300**.
* Φτιάχνουμε embedding\_matrix 10000x300, τοποθετούμε το καθένα στο nn.Embedding, βάζοντας requires\_grad=False.
* Χρησιμοποιούμε 2 LSTM layers (bidirectional), hidden\_dim=64, optimizer Adam, BCELoss.

**3. Αποτελέσματα**

Από τα logs:

Epoch 1/10 - Train Loss: 0.4331, Val Loss: 0.4086

Epoch 5/10 - Train Loss: 0.2526, Val Loss: 0.3371

Epoch 10/10 - Train Loss: 0.1060, Val Loss: 0.4505

Training complete. Best validation loss: 0.3371

Στο test set:

[BiLSTM] Test Set Metrics:

Accuracy : 0.8558

Precision: 0.8286

Recall : 0.8973

F1 Score : 0.8616

**Classification Report (Test)**:

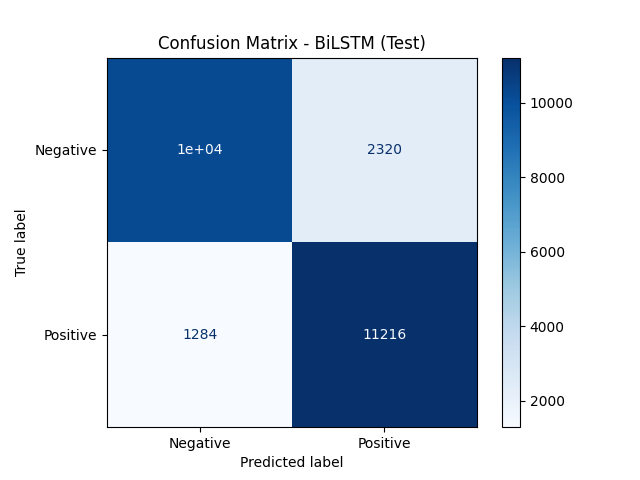
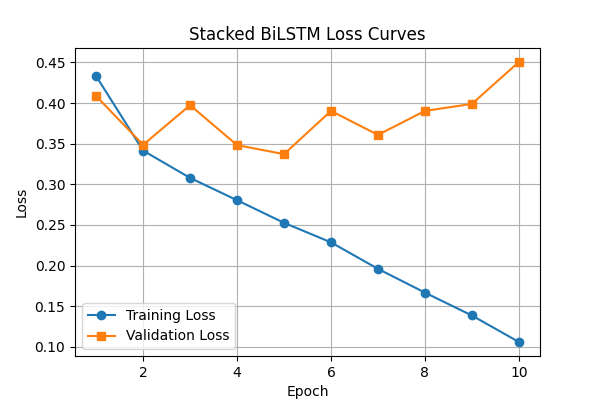
precision recall f1-score support

Negative 0.89 0.81 0.85 12500

Positive 0.83 0.90 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  
  


**Τελική Σύγκριση & Συμπεράσματα**

Συνοψίζουμε τα F1 scores στο test set:

* **Custom NB (best alpha=2.0)**: F1 ~ 0.8373
* **BiLSTM**: F1 ~ 0.8616

Το BiLSTM αποδίδει καλύτερα, αξιοποιώντας την πληροφορία ακολουθιών και τα προ-εκπαιδευμένα embeddings. Ωστόσο, το Naive Bayes είναι ένα απλό, γρήγορο και ερμηνεύσιμο μοντέλο, με αρκετά καλή απόδοση.

**Απαντήσεις στα Ερωτήματα της Εργασίας**

1. **Καμπύλες μάθησης (Α)**: Παρουσιάζονται για F1/Accuracy/Precision/Recall. Φαίνεται πώς αυξάνεται η απόδοση στο validation με περισσότερο train data.
2. **Υπερπαράμετροι**: α=2.0\alpha=2.0 στο Naive Bayes, hidden\_dim=64/2 LSTM layers/Word2Vec embeddings στο BiLSTM.
3. **Αξιολόγηση στο Test**: Confusion matrices & classification reports για NB και BiLSTM.
4. **Τελική σύγκριση**: Το BiLSTM υπερισχύει (~0.862 vs. ~0.837 F1).