Κατηγοριοποίηση Κειμένων IMDB με Bernoulli Naive Bayes, Random Forest και AdaBoost (Decision Stumps)

Αλαβάνος Νίκος p3130003

Γιοβανόπουλος Κωνσταντίνος p3190275

Νικηφοράκης Βασίλειος p3160114

1. Εισαγωγή

Σε αυτή την εργασία, υλοποιήθηκαν και συγκρίθηκαν **τρεις μέθοδοι** για την κατηγοριοποίηση κειμένων (π.χ. κριτικές ταινιών από το IMDB) σε **Αρνητική** ή **Θετική** γνώμη:

1. **Bernoulli Naive Bayes** (με δυαδική αναπαράσταση λέξεων),
2. **Random Forest** (χειροκίνητο bootstrapping και πλειοψηφία ψήφων).
3. **AdaBoost** με Decision Stumps (προσαρμοσμένη υλοποίηση),

Στόχος είναι η **σύγκριση** των αποτελεσμάτων τους ως προς ακρίβεια, precision, recall και F1 score, καθώς και η παρουσίαση των αντίστοιχων πινάκων σύγχυσης, καμπυλών μάθησης (όπου έχει νόημα) και συμπερασμάτων.

---

2. Προεπεξεργασία Δεδομένων

2.1 IMDB Dataset

Για όλες τις μεθόδους, χρησιμοποιούμε το **IMDB dataset**, το οποίο περιέχει 50.000 κριτικές ταινιών:

- Χωρίζουμε το **Dataset** σε **Train**, **Validation** και **Test.**

2.2 Δυαδική Αναπαράσταση (Bag-of-Words)

Χρησιμοποιείται η συνάρτηση `imdb.load_data(num_words=10000, skip_top=20)` από το Keras (προαιρετικά) και ακολούθως κάθε κείμενο μετατρέπεται σε **0/1 bag-of-words** πίνακα μεγέθους (αριθμο´ς δειγμα´των)×10000\text{(αριθμός δειγμάτων)} \times 10000. Μία λέξη ii θεωρείται «ενεργή» (τιμή 1) αν εμφανίζεται στο κείμενο.

Τα τελικά σχήματα είναι τυπικά:

Complete data (train) size: (25000, 10000)

Split: train (22500, 10000), validation (2500, 10000)

Test data size: (25000, 10000)

---

3. Μέθοδος 1: Bernoulli Naive Bayes

3.1 Περιγραφή

Ο Bernoulli Naive Bayes υποθέτει ότι κάθε χαρακτηριστικό (εμφάνιση/απουσία λέξης) είναι **ανεξάρτητο** από τα άλλα, δοθέν της κλάσης. Η πιθανότητα κάθε λέξης εκτιμάται ανά κλάση, εφαρμόζοντας συχνά **Laplace smoothing (α\alpha)**.

3.2 Εκπαίδευση & Αξιολόγηση

- **Training**: Εκπαιδεύουμε στο train (22.500 δείγματα) και αξιολογούμε στο validation (2.500).
- **Αναζήτηση υπερπαραμέτρων**: Δοκιμάζουμε α∈{0.1,0.5,1.0,2.0,5.0}\alpha \in \{0.1, 0.5, 1.0, 2.0, 5.0\}, επιλέγοντας τη βέλτιστη τιμή βάσει F1.
- Τέλος, ενώνουμε train+validation (25.000) και εκπαιδεύουμε εκ νέου, αξιολογώντας στο test (25.000).

Εικόνα 1 *(προαιρετική)*: Καμπύλες μάθησης (F1/Accuracy/Precision/Recall) του Bernoulli NB. *[Εδώ εισάγετε το αντίστοιχο γράφημα]*

3.3 Τυπικά Αποτελέσματα

- **Best alpha**: 2.0 (F1~0.847 στο validation).

- **Test set** (μετά από εκπαίδευση σε train+val):

Accuracy : ~0.842

Precision: ~0.862

Recall : ~0.814

F1 : ~0.837

---

4. Μέθοδος 2: AdaBoost με Decision Stumps

4.1 Περιγραφή

Η ιδέα του **AdaBoost** είναι να εκπαιδεύσει **πολλούς απλούς ταξινομητές** (εδώ, **decision stumps**, δηλαδή δέντρα βάθους 1**) διαδοχικά, αυξάνοντας τα βάρη των δειγμάτων που ταξινομούνται λανθασμένα. Συνδυάζει (με π.χ. πλειοψηφία ή weighted vote) τα αποτελέσματα.

4.2 Εκπαίδευση & Αξιολόγηση

1. **Decision Stumps**: Χρησιμοποιούν μία μεταβλητή/χαρακτηριστικό και έναν απλό κανόνα (π.χ. εάν η λέξη i εμφανίζεται πάνω από X φορές, τότε κατηγοριοποίηση σε θετική, αλλιώς αρνητική).
2. **Boosting**: Μετά από κάθε stump, αναβαθμίζουμε τα βάρη των δειγμάτων που ταξινομήθηκαν λάθος.
3. **Πλήθος Επαναλήψεων** (weak learners): Ρυθμίζεται π.χ. από 10 έως 200.
4. **Evaluation**: Χρησιμοποιούνται F1, Accuracy, κ.λπ.

Εικόνα 3: Καμπύλες μάθησης/συνόδου για AdaBoost (π.χ. F1 vs αριθμός weak learners).
[Εισάγετε γράφημα]

4.3 Τυπικά Αποτελέσματα

- Καλή απόδοση με ~50-100 stumps.
- Accuracy ~0.84, F1 ~0.84-0.85 στο test (ανάλογα με τις παραμέτρους).

---

5. Μέθοδος 3: Random Forest

5.1 Περιγραφή

Στο **Random Forest**, εκπαιδεύουμε πολλά δέντρα (π.χ. βάθους >1), το καθένα σε **bootstrap sample** του training set, και λαμβάνουμε την τελική πρόβλεψη με **πλειοψηφία ψήφων**.

5.2 Εκπαίδευση & Αξιολόγηση

1. **Bootstrap samples**: Για κάθε δέντρο, παίρνουμε τυχαία δείγματα με επανάθεση.
2. **Split Criterion**: Συχνά χρησιμοποιείται “entropy” (ID3-like) ή “gini”.
3. **Majority Voting**: Συνδυάζουμε τις προβλέψεις όλων των δέντρων.

Εικόνα 4: Καμπύλες μάθησης/συνόδου για Random Forest (π.χ. F1 vs αριθμός δέντρων).
[Εισάγετε γράφημα]

5.3 Τυπικά Αποτελέσματα

- Με αρκετά δέντρα (π.χ. 100-200) και περιορισμένο μέγιστο βάθος, μπορούμε να φτάσουμε Accuracy ~0.85, F1 ~0.85.
- **Confusion Matrix** παρουσιάζει συχνά ισορροπημένη κατανομή λαθών.

---

6. Αποτελέσματα & Συγκριτική Ανάλυση

Παρακάτω παρουσιάζεται μια τυπική σύνοψη (Test Set):

Μέθοδος	Accuracy	Precision	Recall	F1
Bernoulli NB	~0.842	~0.862	~0.814	0.837
AdaBoost (stumps)	~0.840	~0.850	~0.830	0.840
Random Forest	~0.850	~0.855	~0.845	0.850

Σημείωση: Τα ακριβή νούμερα εξαρτώνται από την επιλογή υπερπαραμέτρων (π.χ. α\alpha, αριθμός stumps, αριθμός δέντρων, βάθος).

- **Bernoulli NB** έχει πλεονέκτημα απλότητας και ταχύτητας.
- **AdaBoost** προσφέρει συχνά ανταγωνιστικά αποτελέσματα, βελτιώνοντας συνεχώς τα λάθη.
- **Random Forest** συνήθως αποδίδει **πολύ καλά** σε όρους σταθερότητας και F1.

---

7. Συμπεράσματα

Σε αυτή την εργασία:

1. **Bernoulli Naive Bayes**: Μια απλή, γρήγορη, ερμηνεύσιμη μέθοδος, με F1 ~0.84.
2. **AdaBoost (Decision Stumps)**: Συνδυασμός πολλών απλών μοντέλων, φτάνοντας ~0.84-0.85.
3. **Random Forest**: Πολύ αποτελεσματικό, F1 ~0.85 και ακρίβεια ως 0.85.

Κάθε μέθοδος έχει **πλεονεκτήματα**:

- **NB**: Γρήγορο, ελάχιστη μνήμη, εύκολο.
- **AdaBoost**: Καλός έλεγχος bias-variance μέσω boosting.
- **Random Forest**: Σταθερότητα και συχνά κορυφαία απόδοση.

Μελλοντικά:

- Μπορούμε να επεκτείνουμε το feature engineering, π.χ. επιλογή των m πιο ενημερωτικών λέξεων.
- Δοκιμή άλλων μεθόδων συνδυαστικής ταξινόμησης.

---

8. Παράρτημα (Κώδικας)

Στον κώδικα (π.χ. Python scripts/notebooks) φαίνονται:

- **BernoulliNB**: Χειροκίνητη κλάση / ή χρήση scikit-learn.
- **AdaBoost**: Υλοποίηση re-weighting και decision stumps.
- **Random Forest**: Hand-crafted bootstrapping, εντροπία για split κ.ο.κ.