University of Athens
Department of Informatics and Telecommunications

Deep Learning for NLP

Student name: *Βασιλική Τσαντήλα*

Course: *Artificial Intelligence II*
Semester: *Spring Semester 2025*

Περιεχόμενα

1. Abstract

Με χρήση της μεθόδου TF-IDF και του Logistic Regression, στόχος μας είναι η ανάπτυξη ενός sentiment classifier για ταξινόμηση συναισθήματος σε δοσμένο dataset από tweets.

2. Data processing and analysis

2.1. Pre-processing

Εκτυπώνουμε μερικά από τα δεδομένα μας:

```
0       @whoisralphie dude I'm so bummed ur leaving!
1       oh my god, a severed foot was found in a wheely bin in cobham!!! where
        they found is literally minutes from my house! feel sick now!
2       I end up "dog dialing" sometimes. What's dog dialing, u ask? My dogs
        will walk across my phone & end up calling someone.
3       @_rachelx meeeee toooooo!
4       I was hoping I could stay home and work today, but looks like I have to
        make another trip into town
148383 just love the jonas brothers its too bad i will never get to see them
       tears
148384 another day gone by....time is moving so fast...
148385 fuck college, i'm just gonna marry rich. : fuck college, i'm just gonna
       marry rich.
148386 ZOMGZ NEW SONG FTW. remember that night. <3
148387 http://twitpic.com/7mwrd - Arby's took down their Roastburger coupon.
       But i found the image in my browser cache...so...cheap lunch ...
```

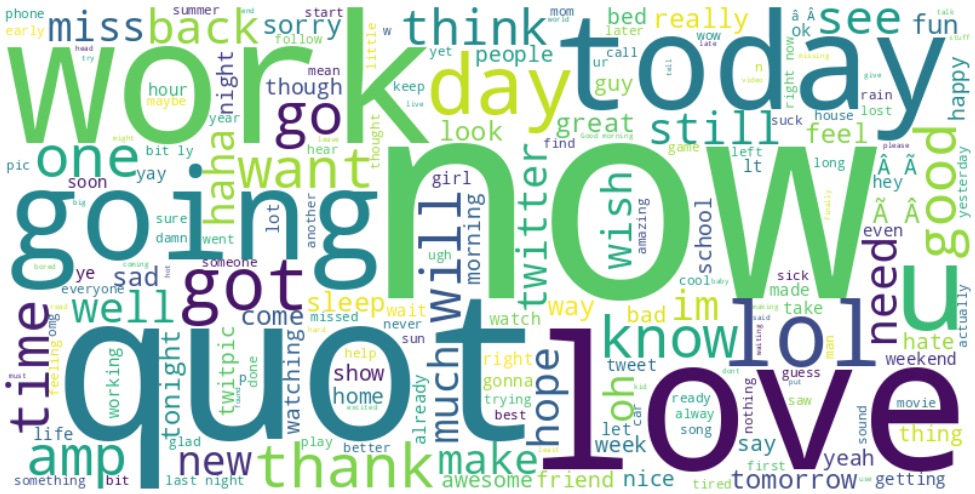
Αναλύοντας τα δεδομένα μάς σκεφτόμαστε, ότι θα είναι χρήσιμο να μετατρέπουμε κάθε κεφαλαίο γράμμα σε πεζό, καθώς θέλουμε οι λέξεις π.χ. 'Good' και 'good' το μοντέλο να τις καταλάβει ως την ίδια λέξη (my_lower).
Επιπλέον, βλέπουμε πως οι προτάσεις μας αποτελούνται από stopwords, λέξεις που δεν προσδίδουν κάποια χρησιμότητα στο μοντέλο στην προσπάθειά του να ερμηνεύσει αν η πρόταση έχει θετικό ή αρνητικό πρόσημο (συναισθηματικά) (my_stopword).
Σκεφτόμαστε, πως μπορούμε να δοκιμάσουμε να αφαιρέσουμε τα σημεία στίξης, και να δούμε αν αυτό θα επιφέρει κάποια βελτίωση ή όχι στο μοντέλο (my_unpunct).
Συνειδητοποιούμε ότι το ίδιο ρήμα εμφανίζεται στο dataset σε διαφορετικές μορφές. Έτσι, σκεφτόμαστε ότι θα είναι χρήσιμο κάθε ρήμα να το μετατρέπουμε στη βασική του μορφή (my_lemmatize).
Αντιλαμβανόμαστε ότι στην εν λόγω άσκηση δεν υπάρχει κάποια απαίτηση για ανωνυμία και προστασία των προσωπικών δεδομένων, επομένως επιλέγουμε να αφήσουμε τυχόν usernames, e-mails κλπ ως έχουν, χωρίς κάποια επιπλέον επεξεργασία.
Σημειώνουμε, ότι η παραπάνω επεξεργασία εφαρμόζεται αφού έχει γίνει πρώτα tokenized το κείμενο (my_split).

Εφαρμόζοντας τις παραπάνω συναρτήσεις προ-επεξεργασίας, παρατηρούμε ότι πλέον τα δεδομένα μας έχουν την εξής μορφή:

```
0       whoisralphie dude  im bummed ur leaving
1       oh god severed foot foun wheely bin cobham found literally minute house
        feel sick now
2       end quotdog dialingquot sumtimes whats dog dialing u askmy dog walk
        across
        phone amp end calling someone aka  quotdog dialingquot
3       rachelx meeeee toooooo
4       hoping could stay home work today look like make another trip town
148383 love jonas brother  tooo bad wil never get see  tear
148384 another day gone bytime moving fast
148385 fuck college im gonna marry rich  fuck college im gonna marry rich
        httpbitlymgic4
148386 zomgz new song ftw  remember night lt3
148387 httptwitpiccom7mwrd  arbys took roastburger coupon     found image
        browser cachesocheap lunch
```

2.2. Analysis

Παράγουμε 2 word clouds πάνω στο X_train: το 1ο word cloud παράγεται πριν την εφαρμογή του preprocessing στο X_train, ενώ το 2ο word cloud παράγεται μετά την εφαρμογή του preprocessing στο X_train.

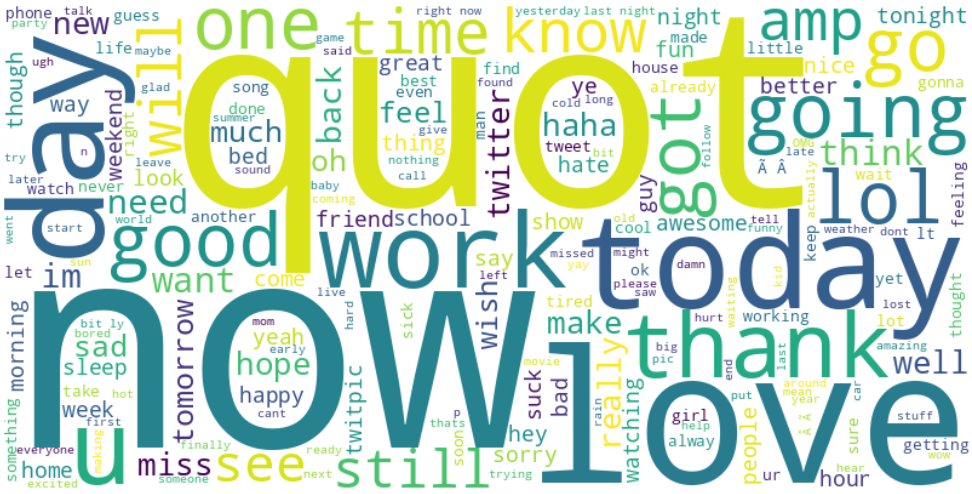


Σχήμα 1: [X_train] Before preprocessing

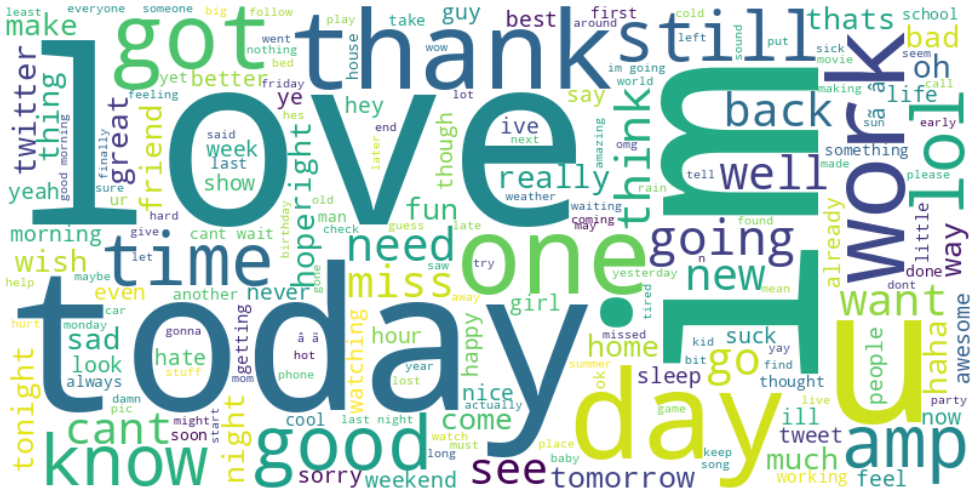
Παρατηρούμε ότι λέξεις ελάχιστης σημασίας (όπως τα stopwords: now, quot) εμφανίζονται με μεγάλη συχνότητα στο *X_train-before-preprocessing.png*. Βλέπουμε όμως ότι στο *X_train-after-preprocessing.png*, οι προαναφερθείσες λέξεις δεν εμφανίζονται πλέον στο wordcloud, αλλά εμφανίζονται λέξεις μείζονας σημασίας για το μοντέλο (όπως love, lol, thank, good), το οποίο είναι και το επιθυμητό.

Με τον ίδιο τρόπο παράγουμε wordclouds για το X_val και X_test. Παρατηρούμε ότι και εδώ εφαρμόζονται οι ίδιες παρατηρήσεις με το X_train.

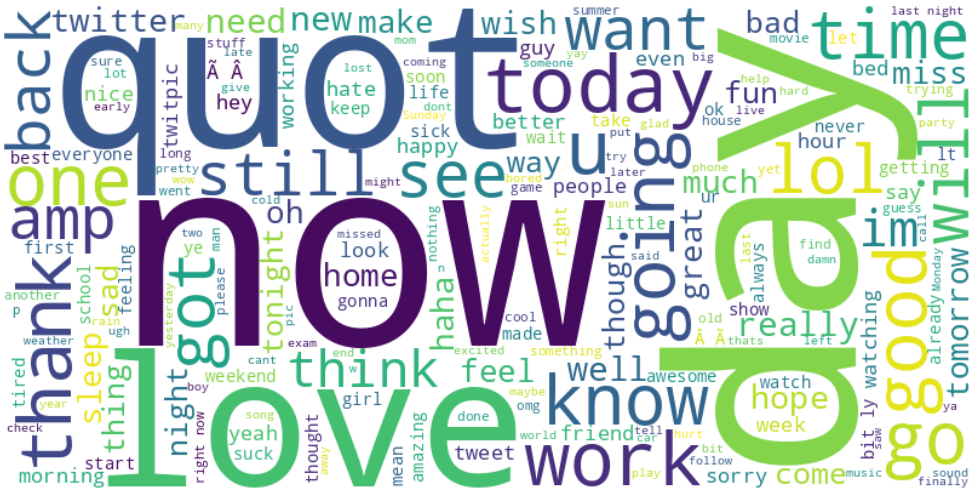
Σχήμα 2: [X_train] After preprocessing



Σχήμα 3: [X_val] Before preprocessing



Σχήμα 4: [X_val] After preprocessing



Σχήμα 5: [X_test] Before preprocessing

Σχήμα 6: [X_test] After preprocessing

2.3. Data partitioning for train, test and validation

Τα δεδομένα μας ήταν ήδη από την εκφώνηση χωρισμένα σε train, validation και test set, συνεπώς δεν υπέστη κάποιο διαφορετικό διαχωρισμό από εμάς.

2.4. Vectorization

Ο vectorizer που χρησιμοποιούμε στην παρούσα εργασία, όπως αυτός ζητείται από την εκφώνηση, είναι εκείνος που για την μετατροπή κειμένου σε αριθμούς εφαρμόζει την TF-IDF φόρμουλα. Η TF-IDF φόρμουλα αποτελείται από δύο δομικά στοιχεία: τον TF (Term Frequency) και τον IDF (Inverse Document Frequency), οι μαθηματικοί τύποι των οποίων είναι:

$$\mathrm{TF}(t,d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

$$\text{IDF}(t) = \log\left(\frac{\text{Total number of documents in corpus}}{\text{Number of documents containing term } t}\right)$$

$$\text{TF-IDF}(t, d) = \text{TF(t,d) x IDF(t)}$$

Σημειώνουμε ότι η TF-IDF φόρμουλα δίνει μεγαλύτερη έμφαση σε λέξεις που εμφανίζονται πιο σπάνια σε ολόκληρο το dataset, αλλά είναι πιο συχνές ανά tweet.

3. Algorithms and Experiments

3.1. Experiments

Ο τρόπος που δουλέψαμε είναι ο εξής: Αρχικά ελέγξαμε αν κάθε συνάρτηση από αυτές που έχουμε ορίσει για το preprocessing βοηθούν στο accuracy του μοντέλου ή όχι, χωρίς να έχουμε πειράξει κάποια παράμετρο του vectorizer ή του Logistic Regression. Στη συνέχεια, μελετήσαμε τις παραμέτρους που προσφέρουν ο vectorizer και το Logistic Regression. Ορίσαμε πιθανά εύρη τιμών για τις παραμέτρους του vectorizer και του logistic regression που κρίναμε ότι είναι χρήσιμο να τροποποιηθούν. Τέλος, τρέξαμε την GridSearchCV για αυτοματοποιημένη εύρεση των βέλτιστων τιμών των παραμέτρων αυτών.

3.1.1 Table of trials

- Το preprocessing βοηθά σε συνδυασμό με έναν default vectorizer και ένα default logistic regression;

Preprocessing	Accuracy
disabled	0.7875
enabled all	0.7511
disabled stopword	0.7782
disabled lemmatize	0.7562
disabled stopword, lemmatize	0.7829
disabled stopword, lemmatize, unpuct	0.7908

Table 1: TfidfVectorizer(lowercase=False) & LogisticRegression(max_iter = 300)

Παρατηρούμε ότι η μοναδική συνάρτηση στο preprocessing που βοηθά να πετύχουμε καλύτερο accuracy από όταν το preprocessing είναι εντελώς απενεργοποιημένο, είναι η συνάρτηση my_lower. Συνεπώς, αποφασίζουμε από εδώ και στο εξής, η συνάρτηση my_lower να είναι η μοναδική που είναι ενεργοποιημένη.

- Ποιες παραμέτρους του vectorizer πρέπει να πειράξουμε;

Μετά από μελέτη των διαθέσιμων παραμέτρων του TfidfVectorizer, καταλήγουμε ότι οι παράμετροι που θα πειράξουμε είναι οι εξής:

⋆ lowercase = False, καθώς το lowercase το κάνουμε ήδη εμείς στο preprocessing.

⋆ max_features: [None, 1000, 50000], καθώς θέλουμε να ελέγξουμε αν το μοντέλο θα αποδώσει καλύτερα αν το λεξιλόγιό του αποτελείται από λιγότερες σε πλήθος λέξεις, από όσες υπάρχουν συνολικά στο dataset (model, focus on the important stuff!). Συγκεκριμένα, του λέμε το λεξιλόγιό του να αποτελείται από τις top <max_features> πιο συχνές λέξεις του dataset.

⋆ min_df: np.linspace(0.000005, 0.00002, 2). Στο ίδιο πνεύμα με την παράμετρο max_features, θέλουμε το μοντέλο, καθώς δημιουργεί το λεξιλόγιό του να απορρίπτει λέξεις με συχνότητα εμφάνισης στο dataset μικρότερη του <min_df>.

⋆ ngram_range: [(1,2), (1,3)] Σκεφτόμαστε ότι ενδεχομένως να είναι χρήσιμο να προσθέσουμε επιπλέον στο λεξιλόγιο του μοντέλου σύνολα διαδοχικών λέξεων που αποτελούνται από 2 ή 3 λέξεις.

- Ποιες παραμέτρους του LogisticRegression πρέπει να πειράξουμε;

Μελετούμε τις διαθέσιμες παραμέτρους του LogisticRegression, και καταλήγουμε ότι οι παράμετροι που θα πειράξουμε είναι οι εξής:

⋆ C: [1, 10]. Ταυτίζεται με 1/λ, όπου λ είναι η παράμετρος κανονικοποίησης.

⋆ solver: ['saga', 'lbfgs']. Δοκιμάζουμε 2 από τους 6 διαθέσιμους solvers.

⋆ max_iter: [100, 300]. Προσπαθούμε το πλήθος των iterations να είναι επαρκές, ώστε ο solver να μπορεί να κάνει converge.

Για την εύρεση της βέλτιστης τιμής για κάθε παράμετρο τόσο για τον vectorizer όσο και για το LogisticRegression, τρέχουμε την GridSearchCV η οποία θα δοκιμάσει όλους τους συνδυασμούς τιμών των παραμέτρων. Ο τρόπος που θα αποφασίσει αν είναι καλό το μοντέλο ή όχι, είναι μέσω cross validation και εύρεση του average accuracy από τα 5 folds. Τρέχοντας, λοιπόν την GridSearchCV με εύρος τιμών για κάθε παράμετρο τις τιμές που αναφέραμε στα δύο προηγούμενα bullet points, βρίσκουμε ότι οι καλύτερες τιμές είναι:
vectorizer__max_features: None
vectorizer__min_df: 5e-06
vectorizer__ngram_range: (1, 3)
logisticregression__C: 10
logisticregression__max_iter: 100
logisticregression__solver: saga

3.2. Optimization techniques

Είχαμε πολλές επιλογές, ωστόσο επικεντρωθήκαμε στα εξής:

- Χρησιμοποιήσαμε την GridSearchCV, προκειμένου να μην δοκιμάζουμε με το χέρι κάθε συνδυασμό τιμών παραμέτρων.
- Χρησιμοποιούμε cross validation, το οποίο και αυτό αυτοματοποιείται μέσω της GridSearchCV.

- Για να βρούμε ποιο εύρος τιμών έχει νόημα να δώσουμε στην παράμετρο max_features, φροντίζουμε μέσω της συνάρτησης get_feature_names_out του vectorizer, να δούμε πόσα features υπάρχουν με max_features = None για το X_train μας. Έπειτα, φροντίζουμε οι τιμές που θα δώσουμε στο max_features να είναι μικρότερες ή ίσες από το πλήθος των features που μάς επέστρεψε η get_feature_names_out.
  Ακριβώς το ίδιο σκεπτικό ακολουθήσαμε και για την εύρεση των τιμών του min_df.

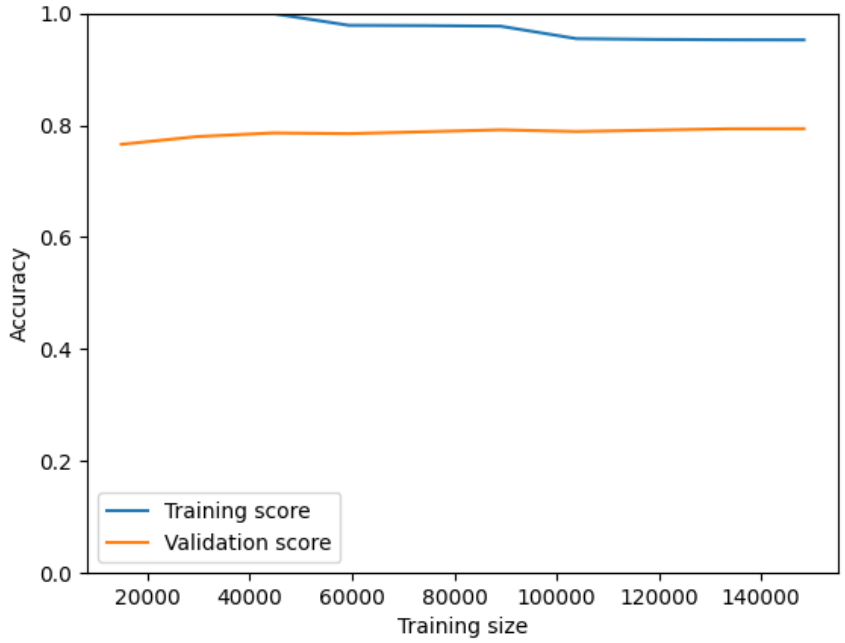
3.3. Evaluation

Έχοντας τρέξει τον vectorizer και το logistic regression με τις καλύτερες παραμέτρους που βρήκαμε στο τελευταίο bullet point του 3.1.1, βρίσκουμε ότι το accuracy που παίρνουμε στο validation set είναι: 0.803849 στα 42396 samples. Βρίσκουμε, δηλαδή, ότι το μοντέλο μας πετυχαίνει σωστά το 80% των samples μας.

3.3.1 Learning curve

Έχοντας κατά νου την τιμή του accuracy του μοντέλου στο validation set, είναι χρήσιμο να μελετήσουμε την καμπύλη μάθησης του μοντέλου (learning curve), προκειμένου να ελέγξουμε αν το μοντέλο μας κάνει overfit, underfit, ή τίποτα από τα δύο.

Μελετώντας την καμπύλη, όπως αυτή φαίνεται στο παρακάτω σχήμα, βλέπουμε ότι υπάρχει τάση σύγκλισης και κατάληξη σε σταθεροποίηση. Γεγονός που σημαίνει ότι το μοντέλο μας δεν πάσχει από underfitting ή overfitting.

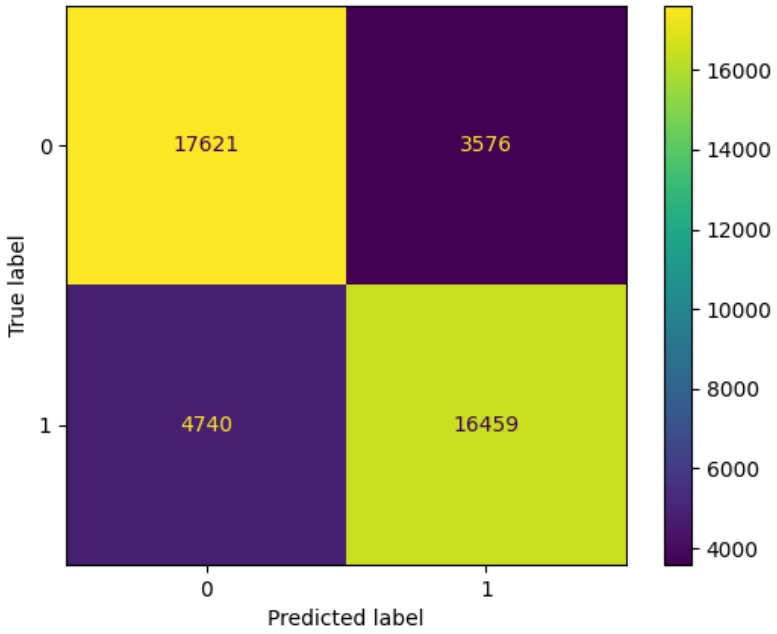


Σχήμα 7: Learning curve for Logistic Regression

3.3.2 Confusion matrix

Χρήσιμο στη μελέτη του μοντέλου μας είναι και ο σχεδιασμός ενός confusion matrix, όπως αυτός φαίνεται στο παρακάτω σχήμα. Σημειώνουμε ότι το πάνω αριστερά τετραγωνάκι του

πίνακα μάς δείχνει το πλήθος των samples με χαρακτηρισμό True-Negative, το πάνω δεξιά: False Positive, το κάτω αριστερά: False Negative, και το κάτω δεξιά: True Positive. 'True' σημαίνει ότι η τιμή του label έγινε predicted σωστά, ενώ 'False' σημαίνει εσφαλμένο prediction.



Σχήμα 8: [X_val] Confusion matrix

Έχοντας πλέον τον confusion matrix, μπορούμε να υπολογίσουμε τις τιμές των μετρικών που φαίνονται στο παρακάτω πινακάκι προκειμένου να 'μετρήσουμε' την ποιότητα του τελικού μας μοντέλου.

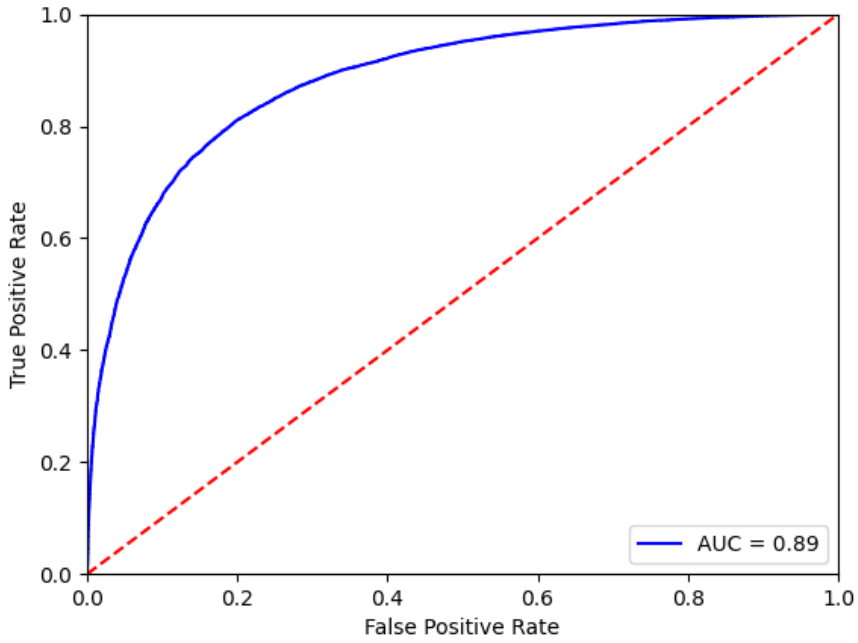
Class	Precision	Recall	F1-score	Support
0	0.79	0.83	0.81	21197
1	0.82	0.78	0.80	21199

Table 2: [X_val] Scores

Βλέπουμε ότι το precision μας είναι αρκετά υψηλό και για τις 2 κλάσεις, κάτι το οποίο σημαίνει ότι το πλήθος των False Positives είναι χαμηλό. Αναφορικά με το recall βλέπουμε ότι είναι και αυτό αρκετά υψηλό και για τις 2 κλάσεις, κάτι το οποίο σημαίνει ότι το πλήθος των False Negatives είναι χαμηλό. Η εικόνα του συνδυασμού των προαναφερθέντων μετρικών αποτυπώνεται στο F1-score, δίνοντας και σε αυτό αρκετά υψηλή τιμή.

3.3.3 ROC curve

Ακολουθεί η Reciever Operating Characteristic Curve (ROC) του μοντέλου. Παρατηρούμε ότι η Area Under the Curve (AUC) είναι σημαντικά μεγαλύτερη από 0.5 (τιμή που αποδίδεται σε ένα μοντέλο που τυχαία δίνει τιμές στα predictions του). Η τιμή της AUC μας πλησιάζει το 1, καθιστώντας το μοντέλο μας αρκετά καλό.



Σχήμα 9: [X_val] ROC curve

4. Results and Overall Analysis

Το τελικό μας μοντέλο είναι αρκετά καλό. Βρήκαμε, ότι το preprocessing (εκτός της μετατροπής σε lowercase) δεν βοήθησε ανοδικά στην απόδοση του μοντέλου μας. Με τη βοήθεια της GridSearchCV καταφέραμε να βρούμε τις τιμές παραμέτρων για τον vectorizer και το logistic regression που θα ήταν πιο επωφελείς σε εμάς. Ο σχεδιασμός διαγραμμάτων μάς βοήθησε να καταλάβουμε αν το μοντέλο μας πάσχει από overfitting ή underfitting, αν και τελικά δεν έπασχε από κάποιο από τα δύο.
Αξίζει να σημειωθεί ότι το accuracy που πετύχαμε στο test set με το τελικό μας μοντέλο είναι ίσο με 0.80135.