

## ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

### ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΑΚΑΔ. ΕΤΟΣ 2024-2025

#### Sentiment Analysis using SVM/Neural Network on IMDB Data

Κωνσταντίνος Ζαφειρόπουλος E19044

#### 1.Εισαγωγή

Στα πλαίσια της εργασίας του μαθήματος προηγμένα θέματα ανάλυσης δεδομένων ζητήθηκε να εκπαιδεύσουμε δύο μοντέλα ένα με Support Vector Machine και ένα Νευρωνικό Δίκτυο (Neural Network) με δεδομένα από το dataset IMDB\_Masters του IMDB. Το dataset περιλαμβάνει 100.000 αξιολογήσεις διαφόρων ταινιών και τις κατηγοριοποιεί σε τρεις κατηγορίες : pos (positive), neg(negative) και unsup για κάποιες οι οποίες δεν μπορούν να κατηγοριοποιηθούν. Εμείς πήραμε τα classified reviews εφαρμόσαμε μία διαδικασία preprocessing για να φέρουμε τις κριτικές στην κατάλληλη μορφή έτσι ώστε να μπορούμε να εκπαιδεύσουμε τα μοντέλα. Έπειτα μετατρέψαμε τις κριτικές σε vectors μέσω word2vec μοντέλου που εκπαιδεύσαμε και κατασκευάσαμε τα μοντέλα. Ελέγξαμε την επίδοση τους σε σχέση με τις διάφορες παραμέτρους μέχρι να έχουμε το επιθυμητό αποτέλεσμα, τα συγκρίναμε αναμεταξύ τους και τέλος κατασκευάσαμε ένα web application το οποίο αξιοποιεί τα ήδη εκπαιδευμένα μοντέλα για να εξάγει τα συναισθήματα από νέες κριτικές. Όλα τα μοντέλα, οι τεχνικές, οι βιβλιοθήκες και οι παράμετροι που επιλέχθηκαν θα επεξηγηθούν διεξοδικά παρακάτω όπως και ο source κώδικας ανά block.

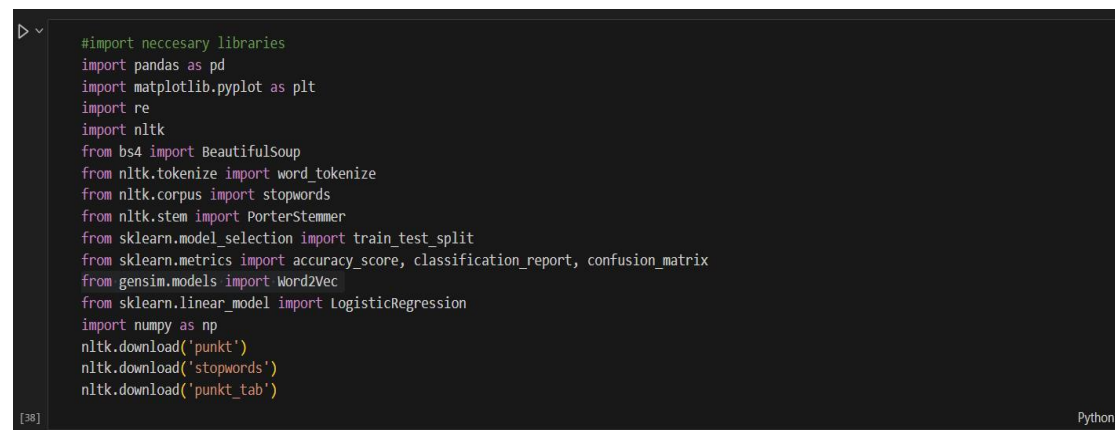
#### 2. Ορισμός προβλήματος

Το πρόβλημα που μας έχει ανατεθεί να λύσουμε είναι η διαχείριση, επεξεργασία και τροποποίηση των δεδομένων ώστε να έρθουν σε μία μορφή κατάλληλη για το σύστημα ώστε να μπορέσει να τα διαχειριστεί. Το δεύτερο σκέλος του προβλήματος είναι η εκπαίδευση των μοντέλων με αποτελεσματικό τρόπο και η επιλογή των σωστών τεχνικών ως προς την υλοποίηση. Τέλος, η μεταφορά

και αναπαράσταση των μοντέλων στο web app και η σωστή ταξινόμηση / classification των όποιων νέων κριτικών.

### 3.Παρουσίαση Προσέγγισης / Μοντέλου

Σε αυτό το σημείο θα αναλυθεί ο πηγαίος κώδικας της εργασίας κομμάτι κομμάτι καθώς και ακριβώς η λειτουργία κάθε block. Να σημειωθεί ότι το περιβάλλον που επιλέχθηκε για την ανάπτυξη του project είναι αρχείο jupyter notebook της python έτσι ώστε να μπορώ να ανιχνεύω εύκολα τα λάθη και να μην χρειάζεται κάθε φορά επανέλεγχος όλου του κώδικα. Ακολουθεί ο κώδικας



```
#import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import re
import nltk
from bs4 import BeautifulSoup
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from gensim.models import Word2Vec
from sklearn.linear_model import LogisticRegression
import numpy as np
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('punkt_tab')
```

[38] Python

Αρχικά, οι βιβλιοθήκες που κρίθηκαν απαραίτητες για την υλοποίηση του project. Η pandas για τη διαχείριση και μετατροπή των δεδομένων. Matplotlib για τα διαγράμματα που αναδεικνύουν την σχέση που επιτύχαμε όσον αφορά το training data και το test data. Τη βιβλιοθήκη regular expressions. Έπειτα, beautiful soup για την απαλοιφή κώδικα html και από την nltk την tokenize για τη διαχώριση των κριτικών σε λέξεις / tokens , την corpus όπου περιλαμβάνει πολύ χρησιμοποιημένες λέξεις διαφόρων γλωσσών που δεν προσφέρουν εννοιολογική αξία στην εκπαίδευση και τη stem η οποία επαναφέρει τις λέξεις στην ριζική τους. Από τη scikit learn χρησιμοποιήσαμε την train\_test\_split για τη διαχώριση σε δεδομένα εκπαίδευσης και δεδομένα ελέγχου , την metrics για τον έλεγχο των μετρικών αξιολόγησης της αξιοπιστίας των μοντέλων. Από την gensim το μοντέλο word2vec που χρειαστήκαμε για την μετατροπή των δεδομένων και τέλος τη numpy για τη διαχείριση αριθμητικών δεδομένων.

```
import os

# Check if the file exists
print(os.path.isfile('imdb_master.csv'))

True

data = pd.read_csv(r"C:\Users\i_zaf\OneDrive\Υπολογιστής\proigmena_themata\imdb_master.csv", encoding="latin-1", engine = 'python')

#checking i can see a review and the file has the format it is supposed to
data['review'][250]

"Take a clich  story and insert Steve Guttenberg.Need i say anymore?This truly is as bad as you would expect. Sheriff Tom Palmer(Guttenberg)"
```

Σε αυτό το σημείο ο κώδικας δεν είναι  λος ο κ δικας απαραίτητος για την λειτουργία της εφαρμογής. Παρόλα αυτά στο πρώτο block ελέγχω για την ύπαρξη του ζητούμενου αρχείου. Έπειτα διαβάζω το αρχείο imdb\_masters.csv χρησιμοποιώντας την εντολή `pd.read_csv` και τέλος ελέγχω  τι μπορώ να δω μία οποιαδήποτε κριτική και  τι το αρχείο έχει την κατάλληλη μορφή.

```
data['label'].value_counts()

label
unsup    50000
neg      25000
pos       25000
Name: count, dtype: int64

data = data[data['label'] != 'unsup']
```

Σε αυτό το σημείο ελέγχω πόσες κριτικές είναι κατηγοριοποιημένες και σε ποιες σχετικές κατηγορίες έχει γίνει η ταξινόμηση μετρώντας τα values της στήλης label για  λα τα διαφορετικά labels.

Η σημαντική παρατήρηση που γίνεται είναι  τι από τις 100.000 κριτικές μόνο οι 50.000 προσφέρουν αξία στην εκπαίδευση των μοντέλων αφού οι 50.000 που ανήκουν στην κατηγορία `unsup` δεν βοηθούν την εκπαίδευση των μοντέλων απλά την μπερδεύουν κατά κάποιο τρόπο. Οπότε με την επόμενη εντολή κάνουμε `drop`  λες τις κριτικές με το `label: unsup`.

```
print(data)
```

|       | Unnamed: 0 | type  | review  |
|-------|------------|-------|---|
| 0     | 0          | test  | Once again Mr. Costner has dragged out a movie... |
| 1     | 1          | test  | This is an example of why the majority of acti... |
| 2     | 2          | test  | First of all I hate those moronic rappers, who... |
| 3     | 3          | test  | Not even the Beatles could write songs everyon... |
| 4     | 4          | test  | Brass pictures (movies is not a fitting word f... |
| ...   | ...        | ...   | ...   |
| 49995 | 49995      | train | Seeing as the vote average was pretty low, and... |
| 49996 | 49996      | train | The plot had some wretched, unbelievable twist... |
| 49997 | 49997      | train | I am amazed at how this movie(and most others ... |
| 49998 | 49998      | train | A Christmas Together actually came before my t... |
| 49999 | 49999      | train | Working-class romantic drama from director Mar... |

|       | label | file        |
|-------|-------|-------------|
| 0     | neg   | 0_2.txt     |
| 1     | neg   | 10000_4.txt |
| 2     | neg   | 10001_1.txt |
| 3     | neg   | 10002_3.txt |
| 4     | neg   | 10003_3.txt |
| ...   | ...   | ...         |
| 49995 | pos   | 9998_9.txt  |
| 49996 | pos   | 9999_8.txt  |
| 49997 | pos   | 999_10.txt  |
| 49998 | pos   | 99_8.txt    |
| 49999 | pos   | 9_7.txt     |

[50000 rows x 5 columns]

```
#pos=1 neg=0  
data['label'] = data['label'].replace({'positive': 1, 'negative': 0})
```

Ελέγχω ότι έχουν πράγματι διαγραφεί οι κριτικές που δεν είναι κατηγοριοποιημένες. Έπειτα αντικαθιστώ την ετικέτα positive με 1 και με 0 την ετικέτα negative για την καλύτερη κατανόηση κατά τη διάρκεια της εκπαίδευσης.

```
#xrisomopoiw etoimpo set leksewn poy tha ginoun drop apo fixed library kai me to porterstemmer ferno tis lekseis  
stop_words = set(stopwords.words('english'))  
stemmer = PorterStemmer()  
from nltk.stem import WordNetLemmatizer  
from nltk.corpus import wordnet  
from nltk import pos_tag  
lemmatizer = WordNetLemmatizer()
```

```
def get_wordnet_pos(tag):  
    if tag.startswith('J'):  
        return wordnet.ADJ  
    elif tag.startswith('V'):  
        return wordnet.VERB  
    elif tag.startswith('N'):  
        return wordnet.NOUN  
    elif tag.startswith('R'):  
        return wordnet.ADV  
    else:  
        return wordnet.NOUN
```

Εδώ θέτω στα stopwords την αγγλική γλώσσα και έπειτα χρησιμοποιώ έτοιμη συνάρτηση που αναγνωρίζει το κομμάτι του λόγου που αναπαριστά κάθε λέξη πχ ουσιαστικό , ρήμα κλπ.

```
#drop axriston lekseon , ola peza , drop special xarakktiron , token ana leksi
def preprocess(review):
    review = BeautifulSoup(review, "html.parser").get_text()
    review = re.sub(r"[^a-zA-Z]", " ", review.lower())
    tokens = word_tokenize(review)
    filtered_tokens = [
        lemmatizer.lemmatize(word, get_wordnet_pos(tag))
        for word, tag in pos_tag(tokens)
        if word not in stop_words
    ]
    return filtered_tokens
```

Αυτή είναι η διαδικασία προεπεξεργασίας των δεδομένων. Όπου αξιοποιούμε όλες τις βιβλιοθήκες που κάναμε import νωρίτερα. Δηλαδή απαλοίφουμε τον κώδικα html (αν υπάρχει) , κάνουμε όλα τα γράμματα πεζά , χωρίζουμε τις κριτικές ανά λέξη και επαναφέρουμε όλες τις λέξεις στη ριζική τους μορφή και παίρνουμε το κομμάτι λόγου που αναπαριστούν. Επιστρέφουμε τις προεπεξεργασμένες κριτικές.

```
data['tokens'] = data['review'].apply(preprocess)

C:\Users\i_zaf\AppData\Local\Temp\ipykernel_30200\3389255198.py:3: MarkupResemblesLocatorWarning: The input looks more like a filename than m
review = BeautifulSoup(review, "html.parser").get_text()

#ekpaidefsi word2vec monteloy epeksigisi epilosis parametron sto documentation
w2v_model = Word2Vec(sentences=data['tokens'], vector_size=200, window=5, min_count=5, workers=4)

#metatropi ton kritikon se vectors kai diaxeirisi axrisimopoiiton leksewn
def vectorize_review(tokens, model, vector_size):
    vectors = [model.wv[word] for word in tokens if word in model.wv]
    if len(vectors) == 0:
        return np.zeros(vector_size)
    return np.mean(vectors, axis=0)

data['vector'] = data['tokens'].apply(lambda x: vectorize_review(x, w2v_model, 200))
```

Εδώ εφαρμόζουμε τη διαδικασία του preprocess σε όλες τις κριτικές. Έπειτα, αναθέτουμε παραμέτρους στο word2vec μοντέλο. Επέλεξα vector size = 200 διότι όταν δοκιμάστηκε το 100 χανόταν μεγάλο ποσοστό ευστοχίας στην ταξινόμηση και όταν δοκιμάστηκε 300 το training διαρκούσε πολύ περισσότερη ώρα. Η παράμετρος window αφορά το πόσες λέξεις πριν και μετά από μία λέξη συμπεριλαμβάνονται στην ‘πρόταση’ , το min count = 5 σημαίνει ότι όσες λέξεις συμπεριλαμβάνονται στο

σύνολο των κριτικών λιγότερο από 5 φορές κόβονται καθώς δεν προσφέρουν εννοιολογική αξία στην εκπαίδευση. Μετά, κατασκευάζουμε μία διαδικασία που μετατρέπει τις κριτικές σε διανύσματα η οποία έχει σαν παραμέτρους τις κριτικές, το μοντέλο που χρησιμοποιείται και το μέγεθος διανύσματος. Τέλος, πάμε και εφαρμόζουμε τη διαδικασία μετατροπής σε διάνυσμα / `vectorize_review` σε όλες τις κριτικές με τις παραμέτρους που έχουμε επιλέξει νωρίτερα για το `word2vec` μοντέλο.

```
#προετοιμασία για εκπαίδεψι
X = np.stack(data['vector'].values)
y = data['label']
```

Python

```
X_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Python

Εδώ γίνεται η σχετική προετοιμασία των δεδομένων για εκπαίδευση και ανατίθεται το 30% των δεδομένων στα δεδομένα ελέγχου.

```
#xrisi svm gia provlepsi
from sklearn.svm import LinearSVC
svm_model = LinearSVC()
# Train the model on training data
svm_model.fit(X_train, y_train)
# Predict on the test data
y_pred_svm = svm_model.predict(x_test)
```

```
#aksiologisi montelou
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print("SVM Test Accuracy: {:.2f}%".format(accuracy_svm * 100))
print("Confusion Matrix (SVM):\n", confusion_matrix(y_test, y_pred_svm))
print("Classification Report (SVM):\n", classification_report(y_test, y_pred_svm))
```

```
SVM Test Accuracy: 85.21%
Confusion Matrix (SVM):
[[6359 1198]
 [1021 6422]]
Classification Report (SVM):
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| neg          | 0.86      | 0.84   | 0.85     | 7557    |
| pos          | 0.84      | 0.86   | 0.85     | 7443    |
| accuracy     |           |        | 0.85     | 15000   |
| macro avg    | 0.85      | 0.85   | 0.85     | 15000   |
| weighted avg | 0.85      | 0.85   | 0.85     | 15000   |



Εδώ γίνεται η χρήση του svm για εκπαίδευση και πρόβλεψη και από κάτω παρατίθενται μερικές μετρικές αξιολόγησης του μοντέλου.

```
#neural network
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.optimizers import Adam

from keras import regularizers

nn_model = Sequential([
    Dense(128, input_dim=X_train.shape[1], activation='relu',
        kernel_regularizer=regularizers.l2(0.01)),
    Dropout(0.3),
    Dense(64, activation='relu',
        kernel_regularizer=regularizers.l2(0.01)),
    Dropout(0.3),
    Dense(1, activation='sigmoid')
])
```

Εδώ γίνεται η προετοιμασία του νευρωνικού δικτύου. Αρχικά import των απαραίτητων βιβλιοθηκών και έπειτα επιλογή παραμέτρων. Αρχικά δοκίμασα με 4 επίπεδα αλλά διαρκούσε πολύ ώρα χωρίς κάποιο πρόσθετο effect στην απόδοση. Επίσης dropout 0.3 είχε με διαφορά την καλύτερη επίδοση χωρίς να χαθεί accuracy και χωρίς να υπάρχει έντονο overfitting.

```
nn_model.compile(optimizer=Adam(learning_rate=0.0001), loss='binary_crossentropy', metrics=['accuracy'])

# ekpaidefsi
history = nn_model.fit(X_train, y_train, validation_data=(x_test, y_test), epochs=8, batch_size=100, verbose=1)
```

| Epoch     | 350/350 | Time        | accuracy | loss   | val_accuracy | val_loss |
|-----------|---------|-------------|----------|--------|--------------|----------|
| Epoch 1/8 | 350/350 | 4s 5ms/step | 0.8230   | 0.6182 | 0.8357       | 0.5387   |
| Epoch 2/8 | 350/350 | 2s 5ms/step | 0.8307   | 0.5382 | 0.8379       | 0.5028   |
| Epoch 3/8 | 350/350 | 2s 4ms/step | 0.8313   | 0.5078 | 0.8379       | 0.4828   |
| Epoch 4/8 | 350/350 | 2s 5ms/step | 0.8294   | 0.4914 | 0.8387       | 0.4688   |
| Epoch 5/8 | 350/350 | 1s 4ms/step | 0.8321   | 0.4805 | 0.8400       | 0.4596   |
| Epoch 6/8 | 350/350 | 1s 4ms/step | 0.8318   | 0.4732 | 0.8417       | 0.4524   |
| Epoch 7/8 | 350/350 | 1s 4ms/step | 0.8355   | 0.4628 | 0.8413       | 0.4474   |
| Epoch 8/8 | 350/350 | 1s 4ms/step | 0.8342   | 0.4578 | 0.8420       | 0.4418   |

Αφού μετατρέψουμε όλα τα δεδομένα σε αριθμητικά με χρήση label encoder. Αναθέτουμε τιμές στο learning rate = 0.0001 όπου παρατηρήθηκε ότι είχαμε την κατάλληλη πορεία της τιμής του accuracy κατά τη διάρκεια της εκπαίδευσης ενώ παράλληλα

αποφεύχθηκε το overfitting. Επίσης επιλέξαμε 8 εποχές στην εκπαίδευση του νευρωνικού και batch\_size = 100. Στο batch size είχα ξεκινήσει από πολύ μικρότερο αριθμό αλλά παρατήρησα ότι μέχρι το 100 όσο το ανέβαζα η εκπαίδευση γινόταν γρηγορότερη χωρίς tradeoff στο accuracy. Από κάτω φαίνονται οι μετρικές αξιολόγησης του νευρωνικού. Να σημειωθεί πως αυτό που μας ενδιαφέρει είναι το validation loss και accuracy κατά κύριο λόγο.

```
import pickle
import joblib
joblib.dump(data['vector'], 'vector.pkl')
joblib.dump(w2v_model, 'word2vec.pkl')

['word2vec.pkl']

joblib.dump(nn_model, 'neural.pkl')

['neural.pkl']
```

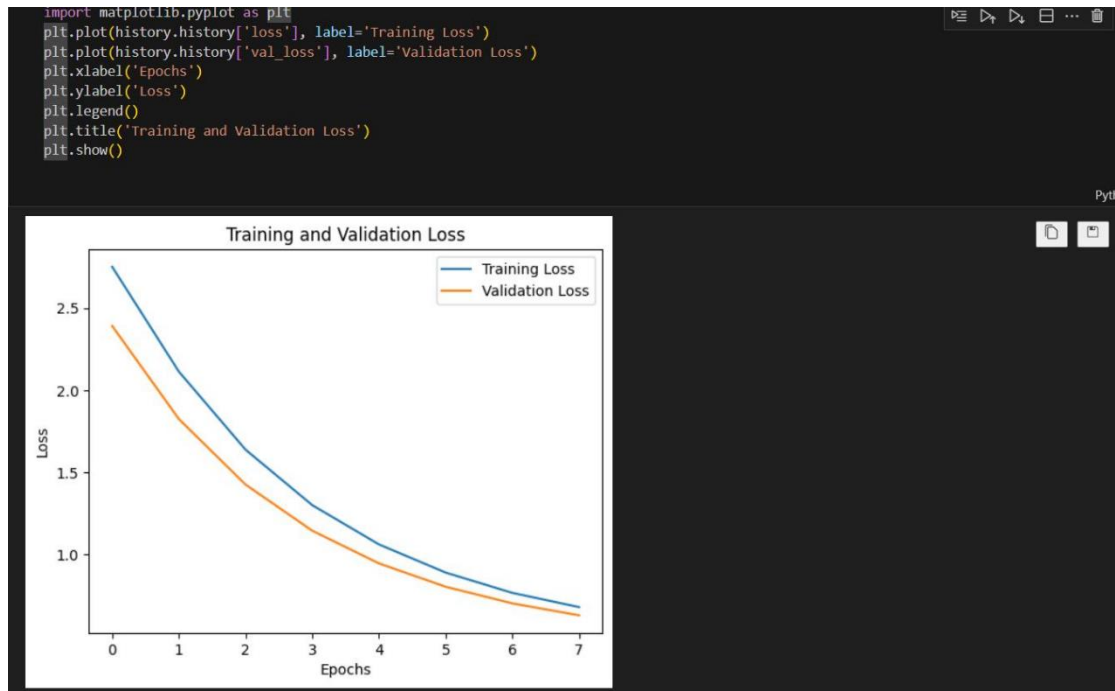
Εδώ μέσω του pickle κάνω dump τα εκπαιδευμένα μοντέλα που θα χρειαστώ στο web app.



Εδώ χρησιμοποιώ το history object που δημιούργησα νωρίτερα στο οποίο αποθήκευσα το ιστορικό εκπαίδευσης για να δω διαγραμματικά ότι έχουμε τη σχέση που θέλουμε ανάμεσα στο



training και το validation accuracy δηλαδή ότι όσο αυξάνεται το ένα τόσο αυξάνεται και το άλλο χωρίς αυξομειώσεις.



Έπειτα κάνω ένα διάγραμμα όπου φαίνεται και η σωστή συσχέτιση ανάμεσα στο training και validation loss όπου φαίνεται ότι δεν υπάρχει overfitting όσο προχωράνε τα epochs. Αν υπήρχαν αυξομειώσεις στο validation loss θα ήταν ένδειξη παρουσίας overfitting.

```
nn_loss, nn_accuracy = nn_model.evaluate(x_test, y_test, verbose=0)
print("Neural Network Test Accuracy: {:.2f}%".format(nn_accuracy * 100))

#sygkrisi modelon
print("\n--- Model Comparison ---")
print(f"SVM Test Accuracy: {accuracy_svm * 100:.2f}%")
print(f"Neural Network Test Accuracy: {nn_accuracy * 100:.2f}%")

--- Model Comparison ---
SVM Test Accuracy: 85.21%
Neural Network Test Accuracy: 83.37%
```

Εδώ βλέπουμε την τελική σύγκριση των μοντέλων όπου υπερισχύει το svm παρόλα αυτά στο web app επέλεξα τη χρήση του neural network.

```

# arxikopoiisi flask app
app = Flask(__name__)

# route homepage
@app.route('/')
def home():
    return render_template('index.html')

# route gia provlepsi
@app.route('/predict', methods=['POST'])
def predict():
    #pare tin kritiki apo to form
    review = request.form['review']

    preproc = preprocess(review)
    vectorized = vectorize_review(preproc, word2vec, 200)
    print('\nShape of Tensor:', tf.shape(vectorized).numpy())
    reshaped_tensor = tf.reshape(vectorized, [-1, 200])
    print('\nShape of Tensor NEW:', tf.shape(reshaped_tensor).numpy())

    prediction = model.predict(reshaped_tensor)
    print(prediction)
    # apokodikopoiise to apotelesma
    result = 'Good Review' if prediction[0] > 0.8 else 'Bad Review'

    # epestrepse to apotelesma
    return render_template('index.html', prediction_text=f'This is a {result}')

if __name__ == '__main__':
    app.run(debug=True)

```

Αυτός είναι όλος ο λειτουργικός κώδικας του αρχείου app.py για την υλοποίηση του web app. Παραπάνω γίνεται το import των βιβλιοθηκών και των ιδίων μεθόδων από τον κώδικα του παραδοτέου για τη διαχείριση της νέας κριτικής. Η διαδικασία είναι η ίδια ακριβώς με μία αλλαγή στα tensor.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <title>Review Sentiment Analysis</title>
</head>
<body>
    <h1>Review Sentiment Analysis</h1>
    <form action="/predict" method="post">
        <textarea name="review" rows="4" cols="50" placeholder="Enter your review here..."></textarea><br><br>
        <button type="submit">Submit</button>
    </form>
    {% if prediction_text %}
    <h3>{{ prediction_text }}</h3>
    {% endif %}
</body>
</html>

```

Τέλος , μία μικρή αναφορά στον html κώδικα που χρησιμοποιήθηκε για το web app στο παραδοτέο αρχείο υπάρχει και το css αρχείο του web application.

#### 4. Πειραματική Μελέτη

Στα πλαίσια της μελέτης θα παρατεθούν δύο απλά παραδείγματα χρήσης της εφαρμογής.

The image displays two screenshots of a web application titled "Review Sentiment Analysis".

The top screenshot shows a text input field containing the sentence "I did not like the movie". Below the input field is a blue "Submit" button. Below the button, the text "This is a Bad Review" is displayed.

The bottom screenshot shows a text input field containing the sentence "The movie was great". Below the input field is a blue "Submit" button. Below the button, the text "This is a Good Review" is displayed.

Στα παραπάνω παραδείγματα βλέπουμε την ορθή πρόβλεψη δύο απλών κριτικών μίας καλής και μίας κακής. Τα περισσότερα από αυτά που ζητούνται στα πλαίσια της πειραματικής μελέτης έχουν αναλυθεί και επεξηγηθεί παραπάνω στην ανάλυση του πηγαίου κώδικα όπως η παρουσίαση δεδομένων, η προεπεξεργασία, οι μετρικές αξιολόγησης και τα αποτελέσματα αλλά και ο σχολιασμός τους. Λίγα λόγια και παρακάτω στα συμπεράσματα. Εδώ παρατέθηκαν δύο παραδείγματα για να φανεί και η εικόνα του web app.

#### 5. Συμπεράσματα

\_\_\_ Το κύριο συμπέρασμα είναι ότι για το συγκεκριμένο παράδειγμα αποδίδουν και τα δύο μοντέλα αρκετά καλά με πολύ

μικρή διαφορά στην αποτελεσματικότητα και την ευστοχία και εκτός αυτού το νευρωνικό παρότι πιο σύνθετο αποδίδει εξαιρετικά γρήγορα στην εκπαίδευση του. Το SVM έχει μία υπεροχή του σχεδόν 2% στο συνολικό accuracy. Παρόλα αυτά και τα δύο κατά τη διάρκεια των δοκιμών απέδωσαν εξίσου καλά με τα αναμενόμενα αποτελέσματα και σωστές προβλέψεις ακόμη και σε πιο σύνθετες κριτικές και αυτός είναι και ο λόγος που επέλεξα το νευρωνικό για πρόβλεψη εντός της εφαρμογής.

## 6. Βιβλιογραφία

Μερικά notebooks που αναφέρθηκα όσον αφορά το sentiment analysis.

<https://www.analyticsvidhya.com/blog/2022/07/sentiment-analysis-using-python/>

<https://www.datacamp.com/tutorial/text-analytics-beginners-nltk>

<https://www.educative.io/answers/how-to-use-svm-for-sentiment-analysis>

<https://www.kaggle.com/code/atillasilva/sentiment-analysis-using-neural-network>