ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

Δημιουργία συνόλου δεδομένων άρθρων και ταξινόμηση κειμένου για διαφημιστικούς σκοπούς

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Φωτεινή Μαρία Κωστοπούλου

Επιβλέπων: Κωνσταντίνος Κουσουρής, Επίκουρος
Καθηγητής Ε.Μ.Π

Αθήνα, Σεπτέμβριος 2020

NATIONAL TECHNICAL UNIVERSITY OF ATHENS

SCHOOL OF APPLIED MATHEMATICAL AND PHYSICAL SCIENCES

Blog Dataset creation and multi-class Text Classification for Advertisement purposes

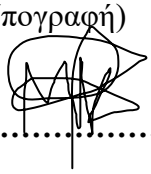
DIPLOMA THESIS

Foteini Maria Kostopoulou

Professor: Konstantinos Kousouris

Athens, September 2020

(Υπογραφή)



……

Φωτεινή Μαρία Κωστοπούλου

Διπλωματούχος Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών Ε.Μ.Π.

Περίληψη

Το αντικείμενο της παρούσας διπλωματικής εργασίας σχετίζεται με τον τομέα της Επεξεργασίας Φυσικής Γλώσσας προσανατολισμένο στις εφαρμογές της διαδικτυακής διαφήμισης. Αναλυτικότερα η επεξεργασία κειμένου είναι αναγκαίο εργαλείο για την αυτόματη προσθήκη διαφημίσεων σε δωρεάν ιστοσελίδες του παγκόσμιου ιστού, οι οποίες διαφημίσεις βασίζονται στα ενδιαφέροντα του κάθε χρήστη έπειτα από την ανάλυση που γίνεται σε προηγούμενους ιστότοπους που έχει επισκεφθεί.

Στο πρώτο σκέλος της εργασίας δημιουργείται ένα σύνολο δεδομένων από άρθρα στην αγγλική γλώσσα, τα οποία συλλέγονται από τις ιστοσελίδες Wikipedia και Twingly. Ακολούθως, το δεύτερο σκέλος της εργασίας απαρτίζει η ταξινόμηση των κειμένων αυτών. Το σετ ταξινόμησης προέρχεται από την ταξινομία IAB της ομώνυμης αμερικανικής εταιρείας και αποτελείται από 169 κλάσεις. Κάθε κλάση χαρακτηρίζει μια κατηγορία διαφήμισης. Τα κείμενα αφού συλλεχθούν περνούν από το στάδιο της προεπεξεργασίας κειμένου, κατά το οποίο αφαιρούνται περιττά στοιχεία (πχ.αριθμοί, σημεία στίξης) και το κείμενο μετασχηματίζεται σε μια πιο απλή μορφή. Ως επόμενο βήμα πριν την ταξινόμηση τα κείμενα χρειάζεται να αναπαρασταθούν διανυσματικά ώστε να είναι κατανοητά σαν είσοδος στον ταξινομητή. Για την διανυσματική αναπαράστασή τους επιλέχθηκε ένα NN εν ονόματι Doc2Vec το οποίο είναι ικανό να αναγνωρίζει σημασιολογικές και συντακτικές πληροφορίες. Το Doc2Vec θα βελτιστοποιηθεί χρησιμοποιώντας τον βασικό ταξινομητή Bernoulli Naive Bayes. Για την ταξινόμηση των κειμένων θα χρησιμοποιηθούν τα μοντέλα Λογιστικής Παλινδρόμησης, Μηχανών διανυσμάτων Υποστήριξης και Πολυστρωματικού Perceptron, αφότου βελτιστοποιηθούν. Οι επιλεγμένοι ταξινομητές εν τέλει θα αξιολογηθούν με κριτήρια την μεγαλύτερη δυνατή ακρίβεια στις προβλέψεις τους και ταυτόχρονα την καλύτερη γενίκευσή που παρουσιάζουν ως μοντέλα.

Λέξεις Κλειδιά: Επεξεργασία Φυσικής Γλώσσας, Σύνολο Δεδομένων, Προεπεξεργασία Κειμένου, Διανυσματική Αναπαράσταση Κειμένου, Doc2Vec, Bernoulli Naive Bayes, Λογιστική Παλινδρόμηση, Μηχανές Διανυσμάτων Υποστήριξης, Πολυστρωματικός Perceptron, Πίνακας Σύγχυσης

Abstract

This thesis surveys Neural Network (NN) models from the perspective of Natural Language Processing (NLP) when it is applied for digital advertising purposes. Automatic text analysis is a very powerful tool that is used in online advertising to recognize users' interests.

The first part of the thesis covers the creation of a dataset comprised of blogs which were collected by websites Wikipedia and Twingly. The second part of the project refers to text categorization. For this goal, the chosen category set is IAB taxonomy which is developed by the homonymous American business specialized in industry standards for the online advertising industry. In order to build the NLP model, firstly is essential to perform text preprocessing techniques like text tokenization and stop-words removal so as to bring blogs to a simpler form and achieve dimensionality reduction. Next, texts need to get numerically represented so that they are comprehensible inputs for the classifier. For the text vectorization a NN is used called Doc2Vec, which model is additionally able to understand semantic and syntactic relationships. Doc2Vec gets optimized using the Bernoulli Naive Bayes algorithm. For the final step of text classification, the models that were selected and optimized are Logistic Regression, SVM and Multilayer Perceptron. All models were in the end evaluated according to the highest accuracy of predictions they can perform and how well they can generalize.

Key words: Natural Language Processing, Text Preprocessing, Text Vectorization, Doc2Vec, Bernoulli Naive Bayes, Logistic Regression, SVM, Multilayer Perceptron, Confusion Matrix

Ευχαριστίες

Πριν την παρουσίαση και ανάλυση των αποτελεσμάτων της παρούσας διπλωματικής εργασίας, αισθάνομαι την ανάγκη να ευχαριστήσω ορισμένους ανθρώπους τους οποίους γνώρισα και συνεργάστηκα μαζί τους.

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ.Κ.Κουσουρή για τη δυνατότητα ανάληψης της επικείμενης διπλωματικής εργασίας, την καθοδήγηση και την υποστήριξή του κατά τη διάρκεια της εκπόνησής της και την προθυμία του να συζητήσει μαζί μου real-world εφαρμογές που έχουν διάφορα κομμάτια της εργασίας.

Ακόμη θα ήθελα να ευχαριστήσω την εταιρεία XPLAIN και ιδιαίτερα τον κ. Κ.Καρακώστα για το θέμα που μου εμπιστεύθηκε στο πολύ ενδιαφέρον επιστημονικό πεδίο της Επεξεργασίας Φυσικής Γλώσσας, καθώς και την πολύτιμη βοήθεια που παρείχε με την παραχώρηση υπολογιστικών πόρων και τις πολύτιμες συμβουλές στο κομμάτι της δημιουργίας του συνόλου δεδομένων.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου για την αγάπη, την διαρκή συμπαράσταση και υπομονή και την υπέρμετρη στήριξή τους, όλα αυτά τα χρόνια.

Περιεχόμενα

Εισαγωγή

Επεξεργασία Φυσικής Γλώσσας (Natural Language Processing - NLP)

Το αντικείμενο της Επεξεργασίας Φυσικής Γλώσσας (ΕΦΓ) έχει εισχωρήσει στην ζωή μας με πολλές καθημερινές εφαρμογές, με ποιο χαρακτηριστικά παραδείγματα την αυτόματη μετάφραση από μια γλώσσα σε μια άλλη, την ταξινόμηση των email σε ανεπιθύμητα και μη, μηχανές αναζήτησης στο διαδίκτυο κτλ.

Η ΕΦΓ είναι ένα επιστημονικό πεδίο που συνδυάζει την επιστήμη των υπολογιστών (Computer Science), την Γλωσσολογία και την τεχνητή νοημοσύνη (Artificial Intelligence) με στόχο την πλήρη αποκρυπτογράφηση και κατανόηση της φυσικής γλώσσας από τον υπολογιστή με τον τρόπο που την αντιλαμβάνεται ο άνθρωπος. Πιο αναλυτικά, για την σχεδίαση της ΕΦΓ οι επιστήμες της γλωσσολογίας, ψυχολογίας και νευροεπιστήμης χρησιμεύουν ως πρώτο βήμα σε θεωρητικό πλαίσιο στο σχηματισμό κανόνων για την δομή και λειτουργία της φυσικής γλώσσας, όπως πχ. συντακτικοί και γραμματικοί κανόνες, και την κατανόηση του τρόπου με τον οποίο ο ανθρώπινος εγκέφαλος μαθαίνει και χρησιμοποιεί την φυσική γλώσσα. Επόμενο βήμα της σχεδίασης της ΕΦΣ είναι η μοντελοποίηση της φυσικής γλώσσας με χρήση προγραμματιστικών και στατιστικών μοντέλων με σκοπό την επίλυση του εκάστοτε προβλήματος.

Ο τομέας της ΕΦΓ περιλαμβάνει πολλούς υποκλάδους οι οποίοι με την σειρά τους έχουν εξελιχθεί σε αυτόνομες επιστήμες. Έννοιες όπως η Γλωσσική Μοντελοποίηση (Language Modeling), η Εξαγωγή πληροφορίας (information extraction), η Εξόρυξη δεδομένων (Data Mining), και η Ανάκτηση Πληροφορίας (Information Retrieval) στελεχώνουν το ευρύ πεδίο της ΕΦΓ. Η παρούσα εργασία θα ασχοληθεί με τον κλάδο της Κατηγοριοποίησης Κειμένων (Text Categorization) για να επιτύχει την ταξινόμηση κειμένων ενός συνόλου δεδομένων (dataset) βάσει του περιεχομένου τους.

Εφαρμογές και παραδείγματα της Επεξεργασίας Φυσικής Γλώσσας σήμερα

Αναλυτικότερα εφαρμογές της ΕΦΓ σήμερα :

- Μηχανική μετάφραση: Η αυτόματη μετάφραση κειμένου από μία ανθρώπινη γλώσσα σε μία άλλη.
- Ανάλυση λόγου : Η αναγνώριση και η κατηγοριοποίηση των γλωσσικών πράξεων σε ένα κομμάτι κειμένου.
- Αυτόματη αναγνώριση ομιλίας: Η αυτόματη μετατροπή του προφερόμενου ανθρώπινου λόγου σε κείμενο από υπολογιστές.
- Αυτόματη ερωταπόκριση: Η αναζήτηση μας σωστής απάντησης σε μία δεδομένη ερώτηση διατυπωμένη σε ανθρώπινη γλώσσα.
- Αυτόματη μορφολογική τεμαχιοποίηση: Η κατάτμηση των λέξεων στα μορφήματά της καθώς και η αναγνώριση και κατηγοριοποίηση αυτών των μορφημάτων.

- Αυτόματη περίληψη: Η παραγωγή μιας αναγνώσιμης περίληψης ενός κειμένου.
- Εξαγωγή πληροφοριών: Η ανάκτηση πληροφοριών από μη δομημένα ή ημιδομημένα δεδομένα (τυπικά κείμενα γραμμένα σε φυσική γλώσσα, ιστοσελίδες κ.α.)
- Επίλυση σχέσεων συναναφοράς : Η αναζήτηση του πλήθους των λέξεων (αναφορών) που αναφέρονται στα ίδια υποκείμενα (οντότητες) σε μία δεδομένη πρόταση ή ένα μεγαλύτερο κομμάτι κειμένου.
- Επισήμανση των μερών του λόγου: Ο αυτόματος καθορισμός των μερών του λόγου σε μία δεδομένη πρόταση και η επίλυση μας συντακτικής αμφισημίας.
- Οπτική αναγνώριση χαρακτήρων: Ο προσδιορισμός του αντίστοιχου κειμένου από μία δεδομένη εικόνα που αναπαριστά κάποιο τυπωμένο κείμενο.
- Παραγωγή φυσικής γλώσσας: Η μετατροπή των πληροφοριών από υπολογιστικές βάσεις δεδομένων σε αναγνώσιμο φυσικό λόγο.
- Σύνθεση ομιλίας: Η αυτόματη, τεχνητή παραγωγή του ανθρώπινου λόγου από μας υπολογιστές.
- Συντακτική ανάλυση: Ο αυτόματος καθορισμός του συντακτικού δέντρου μας δεδομένης πρότασης και η επίλυση των οποιοδήποτε συντακτικών αμφισημιών.

Κάποια εύκολα ευδιάκριτα παραδείγματα της ΕΦΓ στην καθημερινή μας ζωή είναι τα ακόλουθα :

- Autocomplete, Predictive typing εφαρμογές που συμπληρώνουν αυτόματα το υπόλοιπο μιας λέξης ή φράσης.
- Spell checkers, Grammar checkers εφαρμογές που ελέγχουν την ορθογραφία στα κείμενα, γραμματικά λάθη .
- Αυτόματη ταξινόμηση των e-mail σε ανεπιθύμητα και μη.
- Η αυτόματη μετατροπή του προφερόμενου ανθρώπινου λόγου σε βίντεο σε κείμενο σε μορφή υποτίτλων από υπολογιστές.
- Εφαρμογές που επιτρέπουν σε άτομα με ειδικές ανάγκες να επικοινωνήσουν, για παράδειγμα εφαρμογή που επιτρέπει σε κουφούς να επικοινωνήσουν με ανθρώπους που δεν γνωρίζουν την νοηματική.
- Εξαγωγή πληροφοριών από την υπηρεσία Δίωξης Ηλεκτρονικού Εγκλήματος με σκοπό την πρόληψη, την έρευνα και την καταστολή εγκλημάτων ή αντικοινωνικών συμπεριφορών, που διαπράττονται μέσω του διαδικτύου ή άλλων μέσων ηλεκτρονικής επικοινωνίας.

Επεξεργασία Φυσικής Γλώσσας στο χώρο της Διαφήμισης

Βασική έμπνευση για την εργασία αυτή υπήρξε η σπουδαιότητα που κατέχει ο τομέας της ΕΦΓ στον παγκόσμιο ιστό. Η προσθήκη διαφημιστικών μηνυμάτων αποτελεί βασική και στις περισσότερες περιπτώσεις μοναδική εισροή κέρδους για πληθώρα ιστοτόπων δωρεάν πρόσβασης. Η μόνη εναλλακτική λύση στην ελεύθερη πρόσβαση είναι η υποβολή χρηματικού αντιτίμου από τις διαδικτυακές εταιρίες, πράγμα δυσάρεστο για τους χρήστες του Ίντερνετ. Στο σημείο αυτό λοιπόν έρχεται και συμβάλει ο τομέας της ΕΦΓ με την αυτόματη προβολή διαφημίσεων στο διαδίκτυο.

Πώς ακριβώς γίνεται αυτό; Για κάθε ιστοσελίδα που επισκέπτεται ένας χρήστης γίνεται αυτόματη επεξεργασία του κειμένου που περιέχει και κατηγοριοποίηση του περιεχομένου του σύμφωνα με τεχνικές της ΕΦΓ, έτσι ώστε στην επόμενη ιστοσελίδα που θα επισκεφθεί η διαφήμιση που θα προβληθεί να είναι στα ενδιαφέροντά του. Αν για παράδειγμα ένας χρήστης έχει επισκεφθεί μια σελίδα αναφορικά με ταξιδιωτικούς προορισμούς, τότε στην επόμενη σελίδα είναι πολύ πιθανό να εμφανιστούν διαφημίσεις σχετικά με ταξίδια κ.ο.κ.

Reading an article about travel? You'll probably see ads for travel pop on your screen shortly after. Check out a musician's website? You might get some ads for music popping up soon too.

IAB taxonomy

Για την κατηγοριοποίηση των διαδικτυακών κειμένων πολύ χρήσιμο εργαλείο είναι το IAB taxonomy. Ο όρος IAB προέρχεται από την συντομογραφία Interactive Advertising Bureau και αποτελεί μια αμερικάνικη επιχείρηση η οποία αποτελείται από περισσότερες από 650 επιχειρήσεις των media, brands και τεχνολογίας και είναι υπεύθυνη για την έκδοση και πώληση ψηφιακών διαφημιστικών καμπανιών. Η επιχείρηση αυτή επιπλέον κατέχει το ερευνητικό κέντρο IAB Tech Lab το οποίο δραστηριοποιείται στην δημιουργία και συντήρηση προδιαγραφών και εφαρμογών στην ψηφιακή διαφήμιση.

Μια από τις παροχές που προσφέρει η εταιρία IAB είναι μια ταξινομία εν ονόματι IAB taxonomy η οποία θα αποτελέσει το σετ εκπαίδευσης για την ταξινόμηση κειμένων της παρούσας εργασίας. Το σετ εμπεριέχει κατηγορίες προϊόντων προς διαφήμιση και διακλαδώνεται σε 4 επίπεδα (Tiers). Ο αριθμός κάθε Tier υποδηλώνει τον βαθμό εμβάθυνσης συγκριτικά με το Tier 1 το οποίο αποτελεί την γενικότερη κατηγορία. Για παράδειγμα , αν Tier 1 αποτελεί την κατηγορία “Books & Literature” τότε το Tier 2 θα είναι η υποκατηγορία του “Books & Literature>Fantasy Books”.

Το πρωτότυπο IAB taxonomy αποτελείται από 23 Tier 1 και 372 υποκατηγορίες, για τον σκοπό της παρούσας εργασίας αυτές συγχωνεύτηκαν σε 169 και παρουσιάζονται ακολούθως.

	Category		
1	Automotive>Auto Body Styles & Types	88	News and Politics>News
2	Automotive>Auto Buying and Selling	89	News and Politics>Politics
3	Automotive>Auto Parts & Repair	90	Personal Finance>Insurance>Health Insurance
4	Automotive>Auto Safety	91	Personal Finance>Insurance>Home Insurance
5	Automotive>Auto Shows	92	Personal Finance>Insurance>Life Insurance
6	Automotive>Auto Technology	93	Personal Finance>Insurance>Motor Insurance
7	Automotive>Generic Automotive	94	Personal Finance>Insurance>Pet Insurance
8	Automotive>Motorcycles	95	Personal Finance>Insurance>Travel Insurance
9	Automotive>Road-Side Assistance	96	Personal Finance>Retail Banking and Finance
10	Books and Literature>Biographies	97	Pets
11	Books and Literature>Children's Literature	98	Pop Culture>Celebrities
12	Books and Literature>Comics and Graphic Novels	99	Real Estate>Personal Real Estate
13	Books and Literature>Cookbooks	100	Real Estate>Professional Real Estate
14	Books and Literature>Fiction	101	Religion & Spirituality>Astrology
15	Books and Literature>Poetry	102	Religion & Spirituality>Religion
16	Books and Literature>Travel Books	103	Science>Biology and Genetics
17	Careers>Apprenticeships & Vocational Training	104	Science>Environment
18	Careers>Other Careers	105	Science>Geology and Geography

19	Education>Adult Education	106	Science>Physics Chemistry and Space
20	Education>College Education	107	Shopping>Children's Games and Toys
21	Education>Homework and Study	108	Shopping>Discounts and Promos
22	Education>Online Education	109	Shopping>Grocery and Household Shopping
23	Education>School	110	Sports>Auto Racing
24	Events and Attractions>Amusement and Theme Parks	111	Sports>Basketball
25	Events and Attractions>Casinos & Gambling	112	Sports>Cycling
26	Events and Attractions>Concerts & Music Events	113	Sports>Disabled Sports
27	Events and Attractions>Fan Conventions	114	Sports>Extreme Sports
28	Events and Attractions>Fashion Events	115	Sports>Football
29	Events and Attractions>Museums & Galleries	116	Sports>Gymnastics and Field
30	Events and Attractions>Nature Attractions	117	Sports>Martial Arts
31	Events and Attractions>Nightclubs	118	Sports>Niche Sports
32	Family and Relationships>Family Related Events	119	Sports>Sailing
33	Family and Relationships>Parenting	120	Sports>Skiing
34	Family and Relationships>Single Life	121	Sports>Swimming and Water Polo
35	Fine Art>Dance	122	Sports>Tennis and Racket Sports
36	Fine Art>Digital Arts and Design	123	Sports>Volleyball
37	Fine Art>Modern Art	124	Style & Fashion>Beauty>Hair Care
38	Fine Art>Theater & Opera	125	Style & Fashion>Beauty>Makeup and Accessories
39	Food & Drink>Alcoholic Beverages	126	Style & Fashion>Beauty>Nail Care
40	Food & Drink>Cooking	127	Style & Fashion>Beauty>Perfume and Fragrance
41	Food & Drink>Desserts and Baking	128	Style & Fashion>Beauty>Skin Care
42	Food & Drink>Dining Out	129	Style & Fashion>Generic Style & Fashion
43	Food & Drink>Non-Alcoholic Beverages	130	Style & Fashion>Men's Fashion>Men's Clothing and Accessories
44	Food & Drink>Vegan/Vegetarian Diets	131	Style & Fashion>Men's Fashion>Men's Shoes and Footwear
45	Healthy Living>Children's Health	132	Style & Fashion>Personal Care
46	Healthy Living>Fitness and Exercise	133	Style & Fashion>Street Style
47	Healthy Living>Weight Loss	134	Style & Fashion>Women's Fashion>Women's Clothing and Accessories
48	Healthy Living>Wellness and Nutrition	135	Style & Fashion>Women's Fashion>Women's Shoes and Footwear

49	Hobbies & Interests>Arts and Crafts	136	Technology & Computing>Artificial Intelligence
50	Hobbies & Interests>Collecting	137	Technology & Computing>Computing>Computer Software and Applications
51	Hobbies & Interests>Games and Puzzles	138	Technology & Computing>Computing>Computers
52	Hobbies & Interests>Generic Hobbies	139	Technology & Computing>Computing>Generic Computing
53	Hobbies & Interests>Magic and Illusion	140	Technology & Computing>Computing>Information and Network Security
54	Hobbies & Interests>Model Toys	141	Technology & Computing>Computing>Internet of Things
55	Hobbies & Interests>Musical Instruments	142	Technology & Computing>Computing>Programming Languages
56	Hobbies & Interests>Photography	143	Technology & Computing>Consumer Electronics>Home Entertainment and Electronics
57	Home & Garden>Home Appliances	144	Technology & Computing>Consumer Electronics>Smart Devices
58	Home & Garden>Home Entertaining	145	Technology & Computing>Robotics
59	Home & Garden>Home Improvement	146	Technology & Computing>Virtual Reality
60	Home & Garden>Home Security	147	Television>Animation TV
61	Home & Garden>Landscaping	148	Television>Children's TV
62	Home & Garden>Smart Home	149	Television>Comedy Drama and SciFi TV
63	Medical Health>Cosmetic Medical Services	150	Television>Music TV
64	Medical Health>Diseases and Conditions>Allergies	151	Television>Reality and Lifestyle TV
65	Medical Health>Diseases and Conditions>Cancer	152	Television>Sports TV
66	Medical Health>Diseases and Conditions>Generic Conditions	153	Travel>Means of Transportation
67	Medical Health>Diseases and Conditions>Heart and Cardiovascular Diseases	154	Travel>Travel Accessories
68	Medical Health>Diseases and Conditions>Injuries	155	Travel>Travel Destinations
69	Medical Health>Diseases and Conditions>Lung and Respiratory Health	156	Travel>Travel Type>Budget Travel
70	Medical Health>Diseases and Conditions>Mental Health	157	Travel>Travel Type>Hotels and Motels
71	Medical Health>Diseases and Conditions>OTC	158	Travel>Travel Type>Road Trips
72	Medical Health>Medical Tests	159	Travel>Travel Type>Travel Reasons
73	Medical Health>Pharmaceutical Drugs	160	Video Gaming>eSports
74	Medical Health>Surgery	161	Video Gaming>Mobile Games

75	Medical Health>Vaccines	162	Video Gaming>Video Game Genres>Action-Adventure Video Games
76	Movies>Action Movies	163	Video Gaming>Video Game Genres>Educational Video Games
77	Movies>Animation Movies	164	Video Gaming>Video Game Genres>Exercise and Fitness Video Games
78	Movies>Drama Comedy and Romance Movies	165	Video Gaming>Video Game Genres>Music and Party Video Games
79	Movies>Family and Children Movies	166	Video Gaming>Video Game Genres>RPG & MMOs
80	Movies>Special Interests Movies	167	Video Gaming>Video Game Genres>Simulation Video Games
81	Music and Audio>Classical Music	168	Video Gaming>Video Game Genres>Sports Video Games
82	Music and Audio>Country Music	169	Video Gaming>Video Game Genres>Strategy Video Games
83	Music and Audio>Dance and Electronic Music		
84	Music and Audio>Generic Radio		
85	Music and Audio>R&B/Soul/Funk and Hip Hop Music		
86	Music and Audio>Rock Music		
87	Music and Audio>Special Interests Music		

Θεωρητικό Υπόβαθρο

Προεπεξεργασία κειμένου (Text Preprocessing)

Ένα κείμενο αποτελεί ένα σύνολο λεκτικών αντικειμένων με ιδιομορφίες στους γραμματικούς και συντακτικούς κανόνες. Στην ηλεκτρονική μορφή ένα κείμενο αναπαρίσταται ως μια σειρά από χαρακτήρες. Όταν όμως επέρχεται το στάδιο της επεξεργασίας, τα κείμενα περιέχουν 'θόρυβο' είτε υπό μορφή λέξεων με χαμηλή πληροφοριακή αξία (επιρρήματα, σύνδεσμοι, αντωνυμίες) είτε με μορφή χαρακτήρων που δεν προσθέτουν πληροφορία όπως αριθμοί, emoticons κτλ. Κάθε κείμενο πριν μετατραπεί σε διάνυσμα και ταξινομηθεί από κάποιο αλγόριθμο ομαδοποίησης πρέπει προηγουμένως να έχει υποβληθεί σε μια διαδικασία προεπεξεργασίας και μετασχηματισμού. Στην συγκεκριμένη εφαρμογή τα βήματα προεπεξεργασίας που χρειάστηκαν είναι τα ακόλουθα:

1. Αναγνώριση Γλώσσας Κειμένων (Language Identification)

Πρωταρχικό βήμα για την προεπεξεργασία του συνόλου κειμένων είναι η βεβαίωση ότι όλα τα κείμενα είναι γραμμένα στην αγγλική γλώσσα. Για την εκτέλεση του χρησιμοποιήθηκε η βιβλιοθήκη της Python FastText. Η βιβλιοθήκη αυτή δημιουργήθηκε από το εργαστήριο Facebook's AI Research (FAIR) και χρησιμοποιείται στην εκπαίδευση απεικόνισης και ταξινόμησης κειμένων σε πάνω από 294 διαθέσιμες γλώσσες. Χρησιμοποιώντας λοιπόν το πακέτο fasttext της Python φιλτραρίστηκαν και επιλέχτηκαν όσα άρθρα κατείχαν την ένδειξη "en" και είναι επομένως γραμμένα στην αγγλική.

2. Καθαρισμός Κειμένων -Text Cleaning

- Μετατροπή κεφαλαίων χαρακτήρων κάθε λέξης σε μικρούς (lowercasing). Σκοπός του βήματος αυτού είναι η ύπαρξη ομοιομορφίας εντός των κειμένων και η αποφυγή επανάληψης λέξεων που αποθηκεύονται ως διαφορετικές λόγω της διαφοράς του πρώτου γράμματός τους, όπως για παράδειγμα σε περιπτώσεις που βρίσκονται στην αρχή προτάσεων.

The capital of Greece is Athens. ➡ the capital of greece is athens.

- Αφαίρεση σημείων στίξης (πχ. ! . , ^ / - κτλ.) , για την αφαίρεση χαρακτήρων που προσθέτουν θόρυβο και την αποφυγή επανάληψης λέξεων οι οποίες παρουσιάζονται ως διαφορετικές, έχοντας σαν επιπρόσθετο χαρακτήρα, κάποιο σημείο στίξης. Οι σύνθετες λέξεις οι οποίες περιέχουν τους χαρακτήρες '−' ή '_', συνενώνονται σε έναν όρο αγνοώντας τους χαρακτήρες αυτούς (π.χ. "long-term"→longterm"). Το ίδιο συμβαίνει και με τις εμφανίσεις άλλων συμβόλων μέσα σε λέξεις.

- Αφαίρεση αριθμών.

we’re		we re
road-trip	→	road trip
e-mail		email

- Αφαίρεση επιπρόσθετων χαρακτήρων κενού.

You should always check your oil, water and tyres /n before taking your car on a long trip.

↓

You should always check your oil, water and tyres before taking your car on a long trip.

- Αφαίρεση hashtag, email, url.

3. Αφαίρεση συνηθισμένων λέξεων (stop-words)

Ως ‘stop-words’ αποκαλούνται οι λέξεις με μικρό σημασιολογικό περιεχόμενο που εμφανίζονται με σχετικά μεγάλη συχνότητα. Αυτές οι λέξεις αφαιρούνται διότι λειτουργούν βοηθητικά στο κείμενο, ώστε να συνδέονται συντακτικά οι έννοιες που ο άνθρωπος θέλει να παραθέσει. Συνήθως πρόκειται για μέρη του λόγου όπως τα άρθρα, οι αντωνυμίες, οι σύνδεσμοι κ.α. Με την αφαίρεσή τους μειώνεται το συνολικό λεξιλόγιο του συνόλου δεδομένων και αποτρέπεται η δημιουργία περιττών χαρακτηριστικών προς ανάλυση.

the, a, an, is, are, was ,were, with, by, from, has, have, that, to, by, for, of, by, from, at, on, in , about, as, will, would, etc.

Παραδείγματα stop-word στην αγγλική γλώσσα.

Για την διπλωματική η λίστα των stop-word προέκυψε από την συγχώνευση των ήδη διαθέσιμων βιβλιοθηκών της Python ntlk, spacy και sklearn.

Επιπλέον αφαιρούνται οι όροι που το μήκος τους είναι πολύ μικρό, συγκεκριμένα που το μέγεθος τους είναι μικρότερο από 2 γράμματα καθώς πιθανόν αφορούν συνδέσμους ή λέξεις με «μικρό» νόημα που δεν υπήρχαν στην λίστα των stop-words.

4. Διαχωρισμός των λέξεων – Tokenization

Στο βήμα αυτό κάθε κείμενο μετατρέπεται από ένα σύνολο χαρακτήρων σε ένα σύνολο από λεκτικές μονάδες (tokens), δηλαδή λέξεις. Το βήμα αυτό είναι απαραίτητο στην επεξεργασία κειμένων, καθώς τα βάρη των λεκτικών μονάδων, που υπολογίζονται από την συχνότητα εμφάνισης των λέξεων σε κάθε κείμενο, θα αποτελέσουν τα συστατικά στοιχεία στην διανυσματοποίηση των

κειμένων. Για τον λόγο αυτό, είναι σημαντικό πριν την διαδικασία κατάτμησης κειμένου να έχει προηγηθεί ο καθαρισμός των κειμένων από σημεία στίξης, αριθμούς, stop-words κτλ.

5. Λημματοποίηση (Lemmatization)

Στο βήμα αυτό σε κάθε κείμενο οι λέξεις μετατρέπονται στην λέξη-ρίζα τους. Συχνά σε κείμενα συμβαίνει να συναντήσουμε την ίδια λέξη, πολλές φορές, αλλά σε διαφορετική μορφή, δηλαδή σε διαφορετικό γένος, πτώση, αριθμό για τα ουσιαστικά και διαφορετικό χρόνο, έγκλιση, αριθμό, πρόσωπο για τα ρήματα. Η λημματοποίηση είναι το κλειδί στην επίλυση αυτού του προβλήματος γιατί ομαδοποιεί τις φόρμες λέξεων μαζί, κάτω από μια ενιαία λέξη-ρίζα (το «λήμμα») και με τον τρόπο αυτό μας επιτρέπει να αναγνωρίσουμε σε όλες αυτές τις περιπτώσεις ότι πρόκειται για την ίδια λέξη.

Ως εκ τούτου η λημματοποίηση μας βοηθά να μετρήσουμε σωστά την συχνότητα εμφάνισης της κάθε λέξης σε κάθε κείμενο και έτσι να αξιολογήσουμε την σημαντικότητά της. Αν για παράδειγμα συναντήσουμε τη λέξη cook με συχνότητα 4, την λέξη cooking με συχνότητα 5 και τη λέξη cooker με 3, τότε όλες αυτές οι λέξεις ανάγονται στην λέξη- ρίζα τους cook με συνολική συχνότητα 12.

Για την εφαρμογή του βήματος αυτού στην εργασία θα χρησιμοποιήσαμε το μοντέλο WordNetLemmatizer, το οποίο παρέχεται από την βιβλιοθήκη της Python NLTK. Το μοντέλο αυτό έχει προεκπαιδευτεί σε δεδομένα της αγγλικής γλώσσας και έχει την δυνατότητα να αναγνωρίζει το μέρος του λόγου (ρήμα, ουσιαστικό, επίθετο, επίρρημα) που αποτελεί κάθε λέξη και να επιλέγει το κατάλληλο λήμμα.

TF-IDF

Ο όρος TF-IDF είναι συντόμευση του term frequency–inverse document frequency (συχνότητα όρου-αντίστροφη συχνότητα εγγράφων). Το μοντέλο TF-IDF μετρά την συχνότητα εμφάνισης κάθε λέξης στο κάθε κείμενο στο πλαίσιο ενός συνόλου κειμένου (corpus).

Η συχνότητα όρου **tf** ορίζεται ως ο λόγος του αριθμού εμφάνισης μιας λέξης-όρου t σε ένα κείμενο d προς το σύνολο των λέξεων του κειμένου.

Η συχνότητα όρου **idf** ορίζεται ως ο λογάριθμος του λόγου του συνόλου των κειμένων |D| προς το σύνολο των κειμένων στα οποία εμφανίζεται η λέξη-όρος t.

Συγκεκριμένα η συχνότητα tfidf ενός όρου υπολογίζεται ως εξής:

- TF (t,d) $= \frac{count(t,d)}{\sum_k count(k,d)}$
- IDF (t,d,D) = $\log \frac{|D|}{\{d:t \in d\}}$
- TF-IDF (t,d,D) = TF(t,d) × IDF (t,d,D)

Για παράδειγμα, έστω ένα μικρό σύνολο κειμένων προτάσεων :

corpus = ["you were born with potential",
"you were born with goodness and trust",
"you were born with ideals and dreams",
"you were born with greatness",
"you were born with wings",
"you are not meant for crawling, so don't",
"you have wings",
"learn to use them and fly"]

Σκοπός είναι να υπολογιστούν οι όροι tf-idf για τις λέξεις “potential” και “wings” που ανήκουν στα κείμενα “you were born with potential” και “you were born with wings” :

	you	were	born	with	potential
Doc 0	1	1	1	1	1

Total number of words
5

	you	were	born	with	wings
Doc 4	1	1	1	1	1

Total number of words
5

Οι λέξεις αυτές συνολικά στο corpus εμφανίζονται όπως δείχνει και ο ακόλουθος πίνακας :

Count of Words Vocabulary words Index & Sentence	0 and	1 are	2 born	3 crawlin	4 don't	5 dreams	6 fly	7 for	8 goodne	9 greatness	10 have	11 ideals	12 learn	13 meant	14 not	15 potential	16 so	17 them	18 to	19 trust	20 use	21 were	22 wings	23 with	24 you
0 you were born with potential			1													1						1		1	1
1 you were born with goodness and trust	1		1						1											1		1		1	1
2 you were born with ideals and dreams	1		1			1						1										1		1	1
3 you were born with greatness			1								1											1		1	1
4 you were born with wings			1																			1	1	1	1
5 you are not meant for crawling, so don't		1		1	1			1						1	1		1								1
6 you have wings											1												1		1
7 learn to use them and fly	1						1						1					1	1		1				
# Documents with the word	3	1	5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	5	2	5	7

Οι tf όροι των λέξεων είναι :

- Tf της λέξης “potential” στο κείμενο "you were born with potential" = 1
- Tf της λέξης “wings” στο κείμενο "you were born with wings"= 1
- Tf της λέξης “wings” στο κείμενο "you have wings" = 1

Αντίστοιχα για τον υπολογισμό των όρων idf όπως διαφοροποιείται η εξίσωση σύμφωνα με την βιβλιοθήκη Sklearn της Python μετρούνται τα εξής απαραίτητα στοιχεία για την εξίσωση του idf στοιχεία :

- Συνολικός αριθμός κειμένων = 8
- Αριθμός των κειμένων που περιέχουν την λέξη “potential” = 1
- Αριθμός των κειμένων που περιέχουν την λέξη “wing” = 2

$$idf(potential) = \ln(\frac{1+8}{1+1}) + 1 = \ln(\frac{9}{2}) + 1 = 2.504077$$

$$idf(wings) = \ln(\frac{1+8}{2+1}) + 1 = \ln(\frac{9}{3}) + 1 = 2.098612$$

Ως τελικό βήμα για τον υπολογισμό των όρων tf-idf πολλαπλασιάζουμε τους υπολογισμένους tf, idf μεταξύ τους :

	you	were	born	with	potential
Doc 0	1	1	1	1	1
	x	x	x	x	x
	you	were	born	with	potential
Doc 0	1.117783	1.405465	1.405465	1.405465	2.504077

Και ο τελικός πίνακας με τους υπολογισμένους tf-idf όρους :

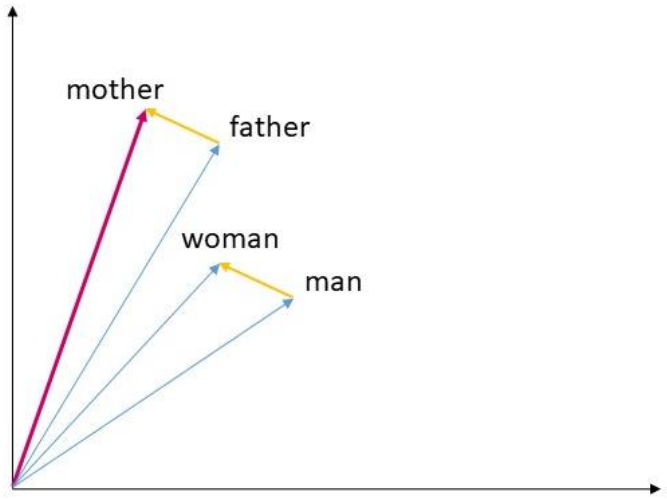
Multiply TF and IDF Vocabulary Words Index & Sentence	and	are	born	crawlin	don	dreams	fly	for	goodne	greatness	have	ideals	learn	meant	not	potential	so	them	to	trust	use	were	wings	with	you
0 you were born with potential	0	0	1.4055	0	0	0	0	0	0	0	0	0	0	0	0	2.504077	0	0	0	0	0	1.4055	0	1.4055	1.1178
1 you were born with goodness and trust	1.81093	0	1.4055	0	0	0	0	0	2.5041	0	0	0	0	0	0	0	0	0	0	2.5041	0	1.4055	0	1.4055	1.1178
2 you were born with ideals and dreams	1.81093	0	1.4055	0	0	2.5041	0	0	0	0	0	2.5041	0	0	0	0	0	0	0	0	0	1.4055	0	1.4055	1.1178
3 you were born with greatness	0	0	1.4055	0	0	0	0	0	0	2.504077	0	0	0	0	0	0	0	0	0	0	0	1.4055	0	1.4055	1.1178
4 you were born with wings	0	0	1.4055	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.4055	2.098612	1.4055	1.1178
5 you are not meant for crawling, so don't	0	2.5041	0	2.5041	2.50408	0	0	2.50408	0	0	0	0	0	2.50408	2.5041	0	2.50408	0	0	0	0	0	0	0	1.1178
6 you have wings	0	0	0	0	0	0	0	0	0	0	2.5041	0	0	0	0	0	0	0	0	0	0	0	2.098612	0	1.1178
7 learn to use them and fly	1.81093	0	0	0	0	0	2.5041	0	0	0	0	0	2.5041	0	0	0	0	2.5041	2.5041	0	2.5041	0	0	0	0

Όπως αναμενόταν όσο χαμηλότερη είναι η τιμή του όρου tf-idf τόσο πιο συχνά εμφανίζεται η δεδομένη λέξη στο σύνολο των δεδομένων, ενώ όσο μεγαλύτερος είναι αντίστοιχα τόσο πιο σπάνια είναι.

Doc2Vec

Ο αλγόριθμος Doc2Vec είναι ένα τυπικό νευρωνικό δίκτυο (ΝΔ) δύο επιπέδων που χρησιμοποιείται για την διανυσματική αναπαράσταση κειμένων και το οποίο κατασκευάστηκε από μια ομάδα ερευνητών με επικεφαλής τον Tomas Mikolov στη Google. Η αρχιτεκτονική του είναι επέκταση του αλγορίθμου Word2Vec για το λόγο αυτό θα γίνει μια σύντομη περιγραφή του.

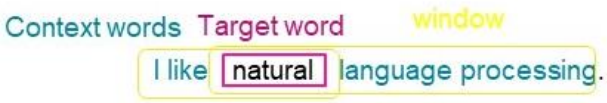
Ο αλγόριθμος Word2Vec όπως μαρτυρεί και η ονομασία του παράγει διανυσματικές αναπαραστάσεις λέξεων. Δέχεται ως είσοδο ένα μεγάλο σύνολο λέξεων και ως έξοδο δημιουργεί ένα διανυσματικό χώρο μεταβλητών διαστάσεων, όπου για κάθε λέξη δημιουργείται και ένα ξεχωριστό διάνυσμα. Με τον τρόπο αυτό σε αντίθεση με άλλους αλγορίθμους που αναπαριστούν την κάθε λέξη ως μια συχνότητα, ο αλγόριθμος αυτός μπορεί να αναγνωρίσει σημασιολογικές και συντακτικές πληροφορίες, οι οποίες μπορούν να αναπαραχθούν με τη χρήση αλγεβρικών πράξεων μεταξύ των παραγόμενων διανυσμάτων. Για παράδειγμα για να βρούμε την σχέση των λέξεων mother και father μέσω των πράξεων των διανυσμάτων father – man + woman = mother. Στο μοντέλο Doc2Vec αυτού του είδους η διανυσματοποιήση επεκτείνεται σε επίπεδο κειμένων.



Αναπαράσταση παραδείγματος αλγεβρικών πράξεων μεταξύ διανυσμάτων λέξεων.

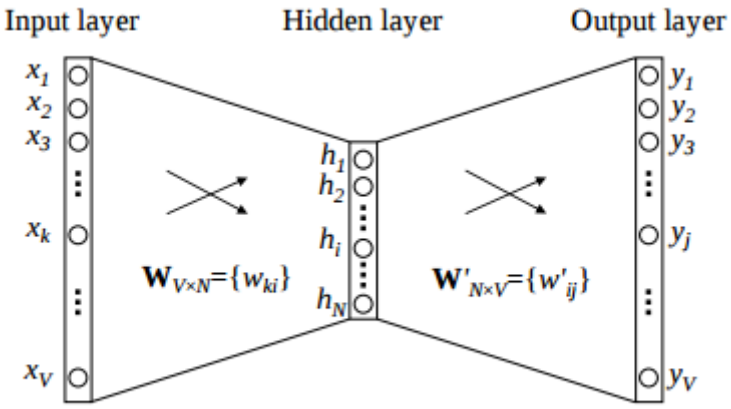
Στον διανυσματικό χώρο αυτό οι σχέσεις των λέξεων υπολογίζονται με την χρήση του εσωτερικού γινομένου. Με τον τρόπο αυτό για λέξεις σημασιολογικά όμοιες το συνημίτονο της γωνίας που σχηματίζεται μεταξύ των διανυσμάτων, που αναπαριστούν τις λέξεις αυτές, τείνει στη μονάδα. Ειδάλλως, το συνημίτονο τείνει στο μηδέν.

Η φιλοσοφία του μοντέλου Word2Vec βασίζεται στην ύπαρξη λέξεων σημασιολογικού πλαισίου (context words) και των λέξεων στόχου (target word), όπου ο αριθμός τους ποικίλλει ανάλογα με τον τύπο αρχιτεκτονικής του αλγορίθμου και μπορεί να οριστεί στον αλγόριθμο μέσω της μεταβλητής του παραθύρου αναζήτησης (context window). Ουσιαστικά, το παράθυρο λέξεων είναι μια ακολουθία συγκεκριμένου αριθμού λέξεων (ο αριθμός λέξεων, δηλαδή το μέγεθος του παραθύρου, είναι υπερ-παράμετρος του μοντέλου).



Αρχικά θα αναλυθεί η απλούστερη αρχιτεκτονική που προτείνεται από την έρευνα του Mikolov, που αντιστοιχεί στην ύπαρξη μιας λέξης σημασιολογικού πλαισίου και μιας λέξης στόχου με στόχο την μάθηση σχέσεων μεταξύ ζεύγους λέξεων (bigram model).

Η αρχιτεκτονική του Word2Vec είναι ένα μοντέλο MLP 3 επιπέδων (layers). Το πλήθος των νευρώνων εισόδου ισοδυναμεί με αυτό των νευρώνων εξόδου και αντιστοιχεί στο μέγεθος του λεξιλογίου της συλλογής κειμένων για εκπαίδευση (corpus). Συγκεκριμένα με τον όρο λεξιλόγιο εννοείται ο συνολικός αριθμός των μοναδικών λέξεων. Οι κρυφοί νευρώνες είναι γραμμικοί και το πλήθος τους ισούται με την διάσταση των διανυσμάτων λέξεων που προκύπτουν, που συνήθως είναι της τάξης των μερικών εκατοντάδων.



Αρχιτεκτονική μοντέλου Word2Vec

Αν λοιπόν το μέγεθος του λεξιλογίου είναι V και η διάσταση των word vectors N, τότε οι συνδέσεις μεταξύ του επιπέδου εισόδου και του κρυφού επιπέδου αναπαρίστανται από έναν πίνακα βαρών W διάστασης VxN, όπου κάθε γραμμή v^T_W του αντιστοιχεί σε μια λέξη του λεξιλογίου. Η είσοδος στο δίκτυο κωδικοποιείται χρησιμοποιώντας την αναπαράσταση «1 από V» ή αλλιώς one-hot. Επομένως αν η εισαγόμενη λέξη στο μοντέλο είναι η μια λέξη σημασιολογικού πλαισίου k, όπου $x_k=1$ και $x_{k'}=0$ για$k' \neq k$

$$h = W^T x = v^T_{WI}$$

Ομοίως για τις συνδέσεις κρυφού επιπέδου και επιπέδου εξόδου υπάρχει ο πίνακας W' διάστασης NxV, στον οποίο κάθε στήλη v'_{wj} αντιπροσωπεύει κάθε λέξη του λεξιλογίου. Χρησιμοποιώντας τα βάρη του πίνακα W' υπολογίζεται το σκόρ u_j για κάθε λέξη του λεξιλογίου ως υποψήφια λέξη στόχου,

$$u_j = {v'_{wj}}^T * h$$

Προκειμένου η έξοδος του μοντέλου να συμβολίζει την πιθανότητα κάθε λέξης του λεξιλογίου να αποτελεί την επόμενη λέξη της λέξης εισόδου (context word) , χρειάζεται το συνολικό άθροισμα εξόδων στο επίπεδο εξόδου να ισούται με 1. Αυτό επιτυγχάνεται με τη συνάρτηση softmax :

$$p(w_j|w_I) = y_j = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})}$$

Δοσμένου λοιπόν ενός πλήθους λέξεων προς εκπαίδευση $w_1, w_2, .., w_V$, η αντικειμενική συνάρτηση του μοντέλου Word2Vec είναι η μεγιστοποίηση της εξαρτημένης πιθανότητας $p(w_O|w_I)$ να εμφανιστεί δηλαδή η λέξη w_O ως έξοδος δοσμένης της λέξης εισόδου w_I. Η μεγιστοποίηση αυτή προκύπτει από την μεγιστοποίηση της μέσης λογαριθμικής πιθανοφάνειας (average log probability) ως ακολούθως:

$$\max p(w_O|w_I) = \max y_{j*}$$
$$= \max \log y_{j*}$$
$$= uj* - \log \sum_{j'=1}^{V} \exp(u_{j'}) = -E$$

,όπου $E = - \log p(w_O|w_I)$ είναι η συνάρτηση κόστους (loss function) του μοντέλου με σκοπό να ελαχιστοποιηθεί και j* αντιστοιχεί στην αρίθμηση (index) της εξόδου της πραγματικής λέξης.

Τα βάρη στους πίνακες W και W' ανανεώνονται σύμφωνα με τον αλγόριθμο της στοχαστικής κατάβασης δυναμικού (Stochastic Gradient Descent–SGD) με χρήση του κανόνα της οπισθοδιάδοσης σφάλματος (backpropagation) και με τον τρόπο αυτό επιτυγχάνεται η μάθηση του αλγορίθμου Word2Vec δημιουργώντας διαφορετικά ζευγάρια context-στόχου από όλο το διαθέσιμο λεξιλόγιο (corpus) και υπολογίζοντας τις διανυσματικές τους αναπαραστάσεις.

Υπάρχουν 2 κύριοι αλγόριθμοι μάθησης στον Word2vec για την μάθηση σχέσεων μεταξύ ζεύγους λέξεων:

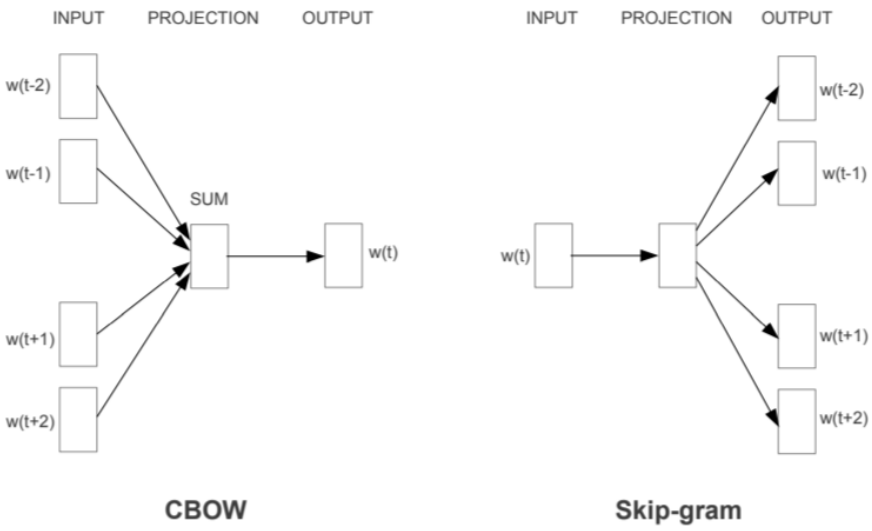
- Ο αλγόριθμος Continuous Bag of Words (CBOW)
- Ο αλγόριθμος Skip-gram

CBOW

Στο μοντέλο CBOW, οι λέξεις σημασιολογικού πλαισίου είναι πολλαπλές ενώ η λέξη στόχου είναι μία. Αν δηλαδή επιθυμούμε να προβλέψουμε τη λέξη στόχου w(t), τότε η είσοδος του μοντέλου μας είναι w(t-2), w(t-1), w(t+1), w(t+2) (οι προηγούμενες και επόμενες λέξεις από τη λέξη στόχου). Η έξοδος του νευρωνικού δικτύου τότε θα είναι w(t). Άρα, σκοπός του CBOW μοντέλου είναι να προβλέπει μία λέξη βάσει του περιεχομένου.

Skip-gram

Αντίθετα με την αρχιτεκτονική CBOW , η αρχιτεκτονική Skip-gram ακολουθεί την αντίθετη προσέγγιση και είσοδος του μοντέλου είναι μία σημασιολογική λέξη w(t), ενώ έξοδος του είναι οι λέξεις w(t-2), w(t-1), w(t+1), w(t+2). Ουσιαστικά δηλαδή σκοπός του μοντέλου είναι να προβλέπει το περιεχόμενο βάσει μίας λέξης.



Αρχιτεκτονικές μοντέλων CBOW και Skip-gram.

Doc2Vec

Ο αλγόριθμος Doc2Vec είναι μια επέκταση του Word2Vec και χρησιμοποιείται για την εκμάθηση διανυσματικών απεικονίσεων ολόκληρων κειμένων. Η ειδοποιός διαφορά του με το μοντέλο Word2Vec είναι η ύπαρξη του διανύσματος παραγράφου (Paragraph Vector – PV) το οποίο αναπαριστά ένα κείμενο και συνεισφέρει στην είσοδο του αλγορίθμου για την πρόβλεψη της λέξης στόχου σύμφωνα με το context που έχει ληφθεί από το κείμενο.

Κάθε κείμενο του dataset απεικονίζεται από ένα μοναδικό διάνυσμα το οποίο αναπαρίσταται από μια στήλη στον πίνακα D. Ομοίως με το μοντέλο Word2Vec κάθε λέξη του λεξιλογίου αναπαρίσταται ως διάνυσμα σε κάθε στήλη του πίνακα W. Στον αλγόριθμο Doc2Vec, οι δυο πίνακες D και W αθροίζονται ή συνδέονται σειριακά ανάλογα με την επιλογή του χρήστη και το αποτέλεσμα δίνεται ως είσοδος για την πρόβλεψη της λέξης στόχου.

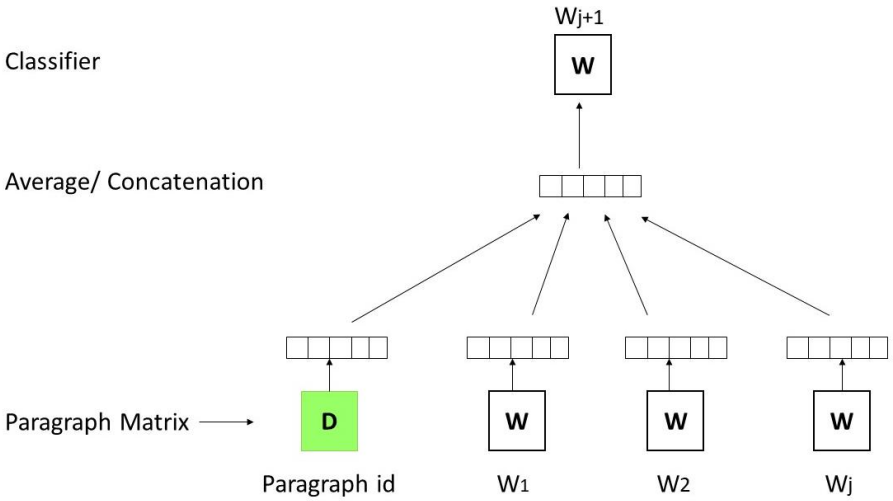
Ο αλγόριθμος Doc2Vec διακρίνεται σε δυο αρχιτεκτονικές : PV-DM, DBOW.

PV-DM

Η αρχιτεκτονική του PV-DM (Paragraph Vector – Distributed Memory) στηρίζεται σε αυτή του CBOW. Το PV-DM εκπαιδεύεται με σκοπό να μάθει μια διανυσματική παράσταση $\vec{p}$ για κάθε κείμενο, ώστε η $\vec{p}$ να επιτρέπει την πρόβλεψη της επόμενης λέξης (w_{j+1}) σε ένα παράθυρο αναζήτησης (context window) $w_i,...,w_j$ της παραγράφου, έχοντας ως είσοδο τη συνένωση ή το μέσο άθροισμα της $\vec{p}$ και των διανυσμάτων των λέξεων του παραθύρου.

Το διάνυσμα παραγράφου υπολογιστικά δεν διαφέρει από το διάνυσμα λέξης, έχει την χρησιμότητα όμως να λειτουργεί σαν “μνήμη” η οποία συγκρατεί την εκάστοτε λέξη στόχου που

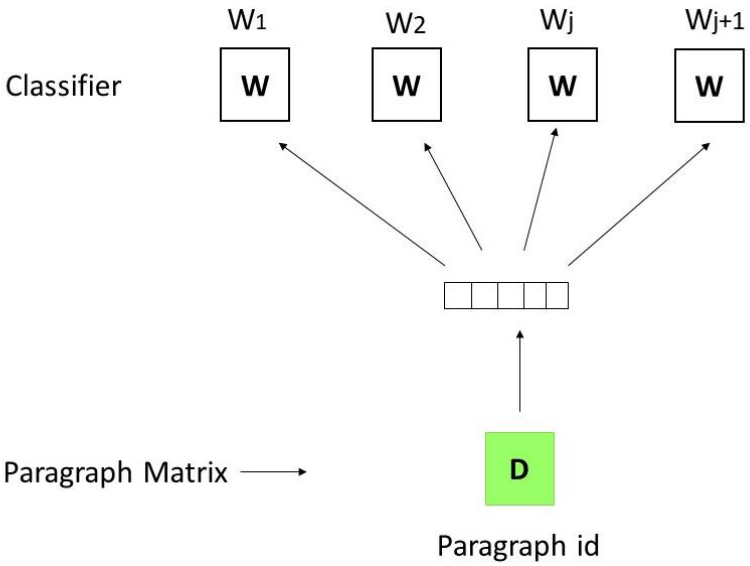
λείπει από το εκάστοτε σημασιολογικό πλαίσιο προς εκπαίδευση, εξ ου και η ονομασία «Distributed Memory».



Αρχιτεκτονική μοντέλου PV-DM.

DBOW

Η αρχιτεκτονική του DBOW μοιάζει με αυτή του Skip-gram. Δέχεται σαν είσοδο ένα διάνυσμα παραγράφου – στήλη του πίνακα D με σκοπό να ταξινομήσει κάθε λέξη που ανήκει σ αυτό. Συγκεκριμένα σε κάθε επανάληψη του αλγορίθμου της στοχαστικής κατάβασης δυναμικού για κάθε παράθυρο αναζήτησης από το εκάστοτε διάνυσμα παραγράφου του πίνακα D, επιλέγεται σαν είσοδος τυχαία μια λέξη από το παράθυρο και ακολουθεί η ταξινόμησή του.



Αρχιτεκτονική μοντέλου DBOW..

Οπισθοδιάδοση σφάλματος (Backpropagation)

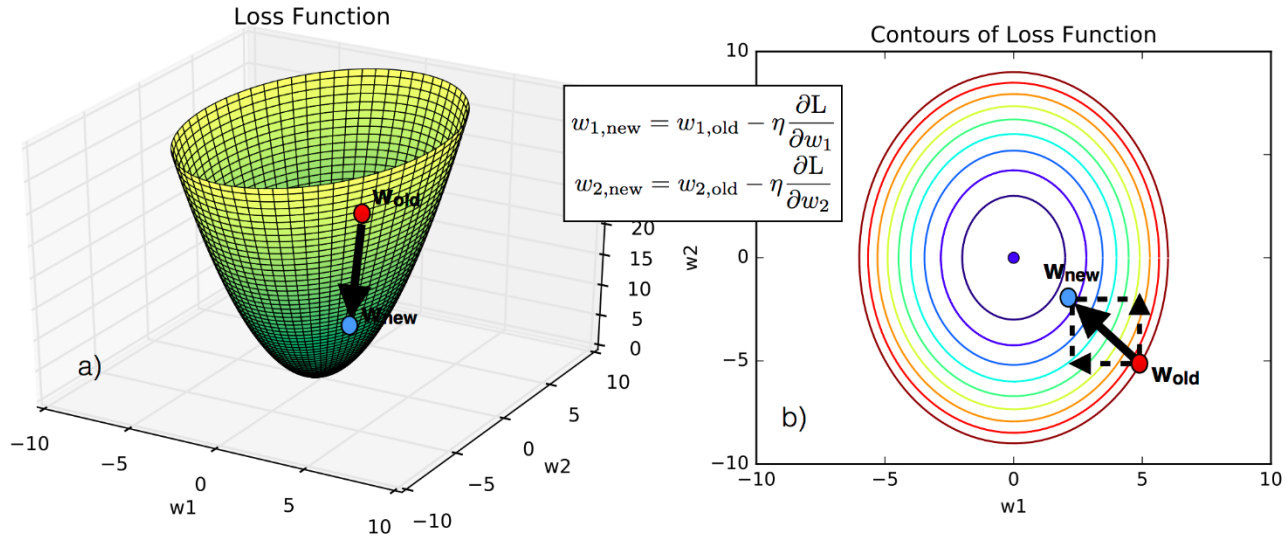
Από την στιγμή που έχει οριστεί η αρχιτεκτονική του ΝΔ δηλαδή έχουν επιλεγεί ο αριθμός των επιπέδων, ο αριθμός των νευρώνων ανά επίπεδο και οι συναρτήσεις ενεργοποίησης σε κάθε επίπεδο, επόμενο βήμα αποτελεί η εκπαίδευσή του. Η εκπαίδευση του ΝΔ έχει ως στόχο τη βελτιστοποίηση των βαρών του δικτύου έτσι ώστε να ελαχιστοποιείται η συνάρτηση κόστους J και γίνεται με χρήση του αλγορίθμου της οπίσθιας διάδοσης (backpropagation).

Ο συνήθης τρόπος για να μάθει το ΝΔ τις βέλτιστες παραμέτρους είναι με την χρήση ενός αλγόριθμου βελτιστοποίησης (optimizer), τον **αλγόριθμο της κατάβασης δυναμικού/κλίσης (Gradient Descent – GD)**. Η λειτουργία του GD βασίζεται στον υπολογισμό της μερικής παραγώγου (gradient) του κριτηρίου J πρώτα ως προς τα βάρη του επιπέδου εξόδου και μετά χρησιμοποιώντας τον κανόνα της αλυσίδας και ως προς τα βάρη των προηγούμενων επιπέδων, για την εύρεση ενός τοπικού ελάχιστου. Ο κανόνας που αξιοποιείται για την ανανέωση των βαρών είναι ο ακόλουθος:

$$w_{new} = w_{old} - n\nabla J(w)$$

(όπου w το διάνυσμα βαρών ενός νευρώνα και n μια μικρή θετική παράμετρος γνωστή ως ρυθμός μάθησης – learning rate).

Σχηματικά ο κανόνας GD μπορεί να αποδοθεί ως ακολούθως με το παράδειγμα δυο μεταβλητών προς βελτιστοποίηση w_1 ,w_2. Υπολογίζοντας τις δυο μερικές παραγώγους $\partial L/\partial w_1$, $\partial L/\partial w_2$ και χρησιμοποιώντας τον κανόνα παρατηρούμε σε πόσες επαναλήψεις θα γίνει η ελαχιστοποίηση της συνάρτησης κόστους (μπλε σημείο):



Σχηματική απεικόνιση ελαχιστοποίησης συνάρτησης κόστους με τη μέθοδο SGD.

Ο αλγόριθμος backpropgation διεξάγεται συνδυαστικά με τον κανόνα GD σε 2 φάσεις:

- Φάση εμπρόσθιας διάδοσης (forward propagation): Το διάνυσμα εισόδου **x** εφαρμόζεται στο επίπεδο εισόδου και αφού γίνουν οι υπολογισμοί σε κάθε επίπεδο, παράγεται τελικά το διάνυσμα εξόδου **y**.

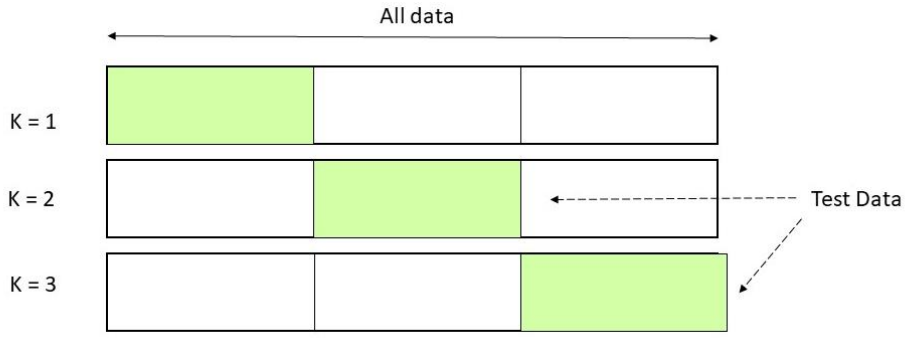
- Φάση ανάστροφης διάδοσης (backward propagation): Αφού υπολογιστεί το διάνυσμα εξόδου **y** που αντιστοιχεί στο διάνυσμα εισόδου **x** , εφαρμόζεται ο κανόνας GD για την ανανέωση των βαρών.

Η κυκλική παρουσίαση όλων των εισόδων **x** στο δίκτυο κατά την εκπαίδευση ονομάζεται εποχή (epoch). Ο αριθμός των εποχών θα αποτελέσει σ αυτήν την εργασία τον παράγοντα για τον τερματισμό της εκπαίδευσης του Doc2Vec.

Διασταυρωμένη επικύρωση Κ-τμημάτων (K-fold cross-validation (K-CV)

Στην στατιστική μέθοδο διασταυρωμένης επικύρωσης Κ-τμημάτων το σύνολο δεδομένων χωρίζεται σε Κ ισόποσα και ξένα μεταξύ τους υποσύνολα D_1, D_2, ... , D_K (folds). Ο αριθμός των υποσυνόλων Κ καθορίζει και των αριθμό των επαναλήψεων. Σε κάθε επανάληψη i=1,2,...,Κ το ΝΔ εκπαιδεύεται θεωρώντας ως σύνολο εκπαίδευσης τα Κ-1 υποσύνολα ($D-D_i$) και υπολογίζεται το γενικό σφάλμα γενίκευσης θεωρώντας το υποσύνολο D_i ως σύνολο ελέγχου. Στο τέλος του αλγορίθμου το σφάλμα γενίκευσης εκτιμάται ως τον μέσο όρο των επιμέρους σφαλμάτων της κάθε επανάληψης και έτσι παράγεται ένας μέσος όρος προβλέψεων.

Στην εφαρμογή της παρούσας εργασίας εφαρμόστηκε η τεχνική Κ-CV για 3 επαναλήψεις. Για την υλοποίηση του K-Fold Validation



Παράδειγμα διασταυρωμένης επικύρωσης με 3 folds.

Multi-variate BernoulliNB

Ο αλγόριθμος BernoulliNB ανήκει στην οικογένεια των αλγορίθμων Naive Bayes και ταξινομεί ένα κείμενο σε μια κλάση, η οποία έχει την μεγαλύτερη πιθανότητα, χρησιμοποιώντας τον κανόνα Bayes ο οποίος για μια δεδομένη κλάση y και τα εξαρτημένα διανύσματα x_1 έως x_v δίνεται :

$$P(y \mid x_1,\ldots, x_v) = \frac{P(y)P(x_1,\ldots,x_v \mid y)}{P(x_1,\ldots,x_v)}$$

Το μοντέλο καλείται «απλοϊκό» (Naive) διότι στηρίζεται στην ανεξαρτησία μεταξύ των μεταβλητών, το οποίο σημαίνει :

$$P(x_i \mid y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_v) = P(x_i \mid y)$$

όπου για όλα τα i απλοποιείται στην μορφή :

$$P(y \mid x_1,\ldots, x_v) = \frac{P(y)\prod_{i=1}^{v} P(x_i \mid y)}{P(x_1,\ldots,x_v)}$$

Και από την στιγμή που η πιθανότητα P(y| x1,.., xν) μένει σταθερή για τα δεδομένα που εισάγονται, χρησιμοποιείται ο ακόλουθος κανόνας ταξινόμησης:

$$P(y \mid x_1,\ldots, x_v) \propto P(y) \prod_{i=1}^{v} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_{y} P(y) \prod_{i=1}^{v} P(x_i \mid y)$$

Με τον τελευταίο όρο να υποδηλώνει την σχετική συχνότητα της κλάσης y που προκύπτει από την μεγιστοποίηση των posteriori πιθανοτήτων P(y) και $P(x_i|y)$.

Όπως συμβαίνει στους ταξινομητές κειμένων (text classifiers), ο BernoulliNB αντιμετωπίζει κάθε κείμενο ως ένα σύνολο λέξεων (bag of words) απ' το οποίο αναγνωρίζει μόνο πόσες λέξεις περιέχονται στο κείμενο χωρίς να καταλαβαίνει την σειρά των λέξεων (word order). Με αυτήν την λογική κάθε κείμενο στο μοντέλο αναπαρίσταται ως ένα στοιχείο-διάνυσμα (Feature Vector – FV) με διάσταση |V| ίση με του λεξιλογίου του συνόλου των δεδομένων και αποτελείται από διωνυμικά στοιχεία που υποδηλώνουν την ύπαρξη (ως 1) ή μη (ως 0) της κάθε λέξης του λεξιλογίου.

Αν λοιπόν θεωρήσουμε $P(w_t|C)$ την πιθανότητα η λέξη w_t να ανήκει στο κείμενο D_i της κλάσης C και $1\text{-}P(w_t|C)$ την πιθανότητα η λέξη w_t να μην ανήκει στο κείμενο D_i αυτής της κλάσης, χρησιμοποιώντας την αναφερθείσα πιθανολογική θεωρία του Bayes, ο BernoulliNB αποτυπώνεται στην εξίσωση:

$$P(D_i \mid C) \sim P(b_i \mid C) = \prod_{t=1}^{|V|}[\, b_{it}P(w_t|C) + (1 - b_{it})\big(1 - P(w_t|C)\big)]$$

Με τον τρόπο υπολογίζεται η πιθανότητα κάθε κειμένου να ανήκει στην αντίστοιχη κλάση ως το σύνολο των πιθανοτήτων των λέξεων που ανήκουν στο κείμενο να ανήκουν στην αντίστοιχη κλάση. Το γινόμενο της εξίσωσης περνά απ' όλες τις λέξεις του λεξιλογίου, και η μεταβλητή bit γίνεται 1 αν η λέξη ανήκει στο κείμενο και η υπολογιζόμενη πιθανότητα είναι η $P(w_t|C)$, και 0 αν η λέξη δεν ανήκει με υπολογιζόμενη πιθανότητα $1-P(w_t|C)$.

Λογιστική Παλινδρόμηση (Logistic Regression)

Ο αλγόριθμος λογιστικής παλινδρόμησης (Logistic Regression) παρότι η ονομασία του παραπέμπει σε παλινδρόμηση, αποτελεί μια γενίκευση της γραμμικής παλινδρόμησης που χρησιμοποιείται για ταξινόμηση. Στην απλούστερη μορφή του είναι ένα μοντέλο που χρησιμοποιεί μια εξαρτημένη δίτιμη μεταβλητή Y και μια ανεξάρτητη μεταβλητή X και παράγει μια γραμμική εξίσωση που περιλαμβάνει μια σταθερά (b_0) και το συντελεστή της παλινδρόμησης (b_1) για την ανεξάρτητη μεταβλητή (X). Η γραμμική αυτή εξίσωση ισούται με το φυσικό λογάριθμο των συμπληρωματικών πιθανοτήτων (odds) του γεγονότος, γνωστή και ως σχέση Logit και γράφεται ως εξής:

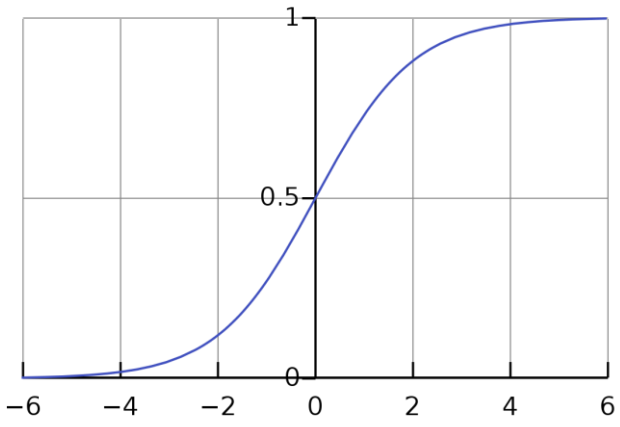
$$\text{Log}\left(\frac{P}{1-P}\right) = b_0 + b_1 X$$

Όπου P είναι η πιθανότητα ταξινόμησης στην κλάση, b_0 είναι η σταθερά του μοντέλου και ρυθμίζει το ύψος της κλίσης της γραμμής παλινδρόμησης, και b_1 είναι ο συντελεστής παλινδρόμησης που δείχνει τον βαθμό επίδρασης της ανεξάρτητης μεταβλητής στην εξαρτημένη. Πιο αναλυτικά, η θετική τιμή του συντελεστή σημαίνει ότι η μεταβλητή X αυξάνει την πιθανότητά να γίνει ταξινόμηση στην κλάση, ενώ η αρνητική τιμή του σημαίνει ότι η μεταβλητή X μειώνει την πιθανότητα αυτή. Αντίστοιχα, η υψηλή τιμή του συντελεστή σημαίνει ότι η ανεξάρτητη μεταβλητή επηρεάζει πολύ ισχυρά την πιθανότητα να συμβεί η ταξινόμηση ή όχι, ενώ η χαμηλή τιμή του δηλώνει τη μικρή επίδραση της ανεξάρτητης μεταβλητής στην πιθανότητα αυτή.

Λύνοντας την παραπάνω εξίσωση ως προς την πιθανότητα P καταλήγουμε στην εξής σχέση :

$$\text{P} = \frac{1}{1 + e^{-(b_0 + b_1 X)}}$$

Η παραπάνω συνάρτηση καλείται σιγμοειδής και αποτελεί την συνάρτηση ενεργοποίησης του LR ταξινομητή. Η ονομασία της οφείλεται στην καμπύλη μορφής S που σχηματίζει όπως παρακάτω :



Σιγμοειδής καμπύλη.

Η παραπάνω σιγμοειδής καμπύλη εμφανίζει τα εξής χαρακτηριστικά :

- Είναι είτε μονότονα αύξουσα συνάρτηση είτε μονότονα φθίνουσα
- Εμφανίζει μια σχεδόν γραμμική συμπεριφορά στην περιοχή [0.2, 0.8]
- Πλησιάζει το 0 και 1 στις ακραίες τιμές της εμβέλειας της ανεξάρτητης μεταβλητής X

Στον LR αλγόριθμο οι συντελεστές παλινδρόμησης εκτιμώνται με τη βοήθεια της μεθόδου της μέγιστης πιθανοφάνειας (Maximum Likelihood Estimate – MLE). Γνωρίζοντας λοιπόν ότι εξαρτημένη μεταβλητή Y ακολουθεί τη διωνυμική κατανομή Bernoulli, όπου $P(Y_i =1) = \pi_i$ για την μία κλάση και $P(Y_i =1) = 1 - \pi_i$, η συνάρτηση πυκνότητας πιθανότητας είναι :

$$f(Yi) = \pi_i^{Y_i}(1-\pi_i)^{1-Y_i} \quad \text{για } Yi = 0,1 \text{ και } i=1,2,\ldots,n$$

και εφόσον οι παρατηρήσεις είναι ανεξάρτητες η από κοινού συνάρτηση πιθανότητας είναι :

$$g(Y1, \ldots, Yn) = \prod_{i=1}^{n} f(Y_i) = \prod_{i=1}^{n} \pi_i^{Y_i}(1-\pi_i)^{1-Y_i}$$

Λογαριθμώντας την πάνω σχέση υπολογίζουμε την λογαριθμική συνάρτηση της μέγιστης πιθανοφάνειας :

$$\ln L(b_0,b_1) = \sum_{i=1}^{n} Y_i (b_0 + b_1 X) + \sum_{i=1}^{n} \ln\left(1 - \frac{e^{(b_0+b_1X)}}{1+e^{(b_0+b_1X)}}\right)$$

$$= \sum_{i=1}^{n} Y_i (b_0 + b_1 X) + \sum_{i=1}^{n} \ln(1 + e^{(b_0+b_1X)})^{-1}$$

$$= \sum_{i=1}^{n} Y_i (b_0 + b_1 X) - \sum_{i=1}^{n} \ln(1 + e^{(b_0+b_1X)})$$

$$= - J(b_0,b_1)$$

Καταλήγοντας στην πάνω σχέση της συνάρτησης κόστους J συμπεραίνουμε πως δεν μπορεί να επιλυθεί αναλυτικά και δεν υπάρχουν λύσεις κλειστής μορφής για τις τιμές των b0, b1 που θα την ελαχιστοποιούσαν. Για το λόγο αυτό η ελαχιστοποίηση της συνάρτησης κόστους επιλύεται προσεγγιστικά και χρησιμοποιούνται επαναληπτικές αριθμητικές μέθοδοι για την εύρεση των εκτιμητών bo και b1.

Για διευκόλυνση, αναπαριστούμε ξανά την συνάρτηση κόστους με τα διανύσματα:

$$\varphi = \begin{pmatrix} x \\ 1 \end{pmatrix} \in R^{d+1} \ , \theta = \begin{pmatrix} w \\ b \end{pmatrix} \in R^{d+1} \quad \text{για } x \in R^{d+1}$$

$$\rightarrow \min_{\theta} J(\theta) = \sum_{i=1}^{n} [-y_i \varphi_i^T \theta + log(1 + e^{\varphi_i^T \theta})]$$

Στα πλαίσια της διπλωματικής θα εφαρμοστεί η επαναληπτική μέθοδος Newton η οποία χρησιμοποιεί τη σειρά Taylor για να προσεγγίσει το μη γραμμικό μοντέλο με γραμμικούς όρους. Στην απλούστερη περίπτωση μιας λείας συνάρτησης f που περιέχει μια μεταβλητή b, το ανάπτυγμα Taylor δεύτερου βαθμού δίνεται ως γνωστόν από την σχέση :

$$f(b) = f(b^{(0)}) + (b - b^{(0)}) f'(b^{(0)}) + (b - b^{(0)})^2 \frac{f''(b^{(0)})}{2!}$$

Για την ελαχιστοποίηση του δεξιού μέλους μηδενίζουμε την παράγωγο της συνάρτησης σε ένα σημείο b(1)

$$0 = f'(b^{(0)}) + 2(b^{(1)} - b^{(0)}) \frac{f''(b^{(0)})}{2!}$$

$$\rightarrow \ b^{(1)} = b^{(0)} - \frac{f'(b^{(0)})}{f''(b^{(0)})}$$

Σύμφωνα με την σχέση η μεταβλητή $b^{(1)}$ θα είναι μικρότερη της αρχικής μεταβλητής $b^{(0)}$, και με την φιλοσοφία αυτή εφαρμόζεται n αριθμός επαναλήψεων για την εύρεση της βέλτιστης τιμής που ελαχιστοποιεί την συνάρτηση.

$$b^{(n+1)} = b^{(n)} - \frac{f'(b^{(n)})}{f''(b^{(n)})}$$

Στην περίπτωση που η συνάρτηση f διαθέτει πολλές παραμέτρους η προαναφερθείσα σχέση παίρνει την μορφή

$$b^{(n+1)} = b^{(n)} - H^{-1}(b^{(n)}) \nabla f(b^{(n)})$$

όπου H ο Hessian πίνακας της συνάρτησης και ∇ η απόκλισή της. Επιστρέφοντας στην συνάρτηση κόστους της λογιστικής παλινδρόμησης, οι ποσότητες αυτές υπολογίζονται όπως παρακάτω:

- $\nabla J(\theta) = -\sum_i^n y_i \varphi_i + \sum_{i=1}^n \varphi_i \frac{e^{\varphi_i^T \theta}}{1+e^{\varphi_i^T \theta}} = -\sum_{i=1}^n (y_i - p_i)\varphi_i$, για $p_i = \frac{e^{\varphi_i^T \theta}}{1+e^{\varphi_i^T \theta}}$

- $H = \frac{\partial^2 J(\theta)}{\partial \theta \, \partial \theta^T} = \sum_{i=1}^{n} p_i(1 - p_i)\varphi_i \, \varphi_i^T$

Στην περίπτωση που ο αριθμός των δεδομένων εκπαίδευσης n είναι πολύ μικρός συγκριτικά με τον αριθμό d των feature variables ο αλγόριθμος LR συνηθίζει να εμφανίζει το φαινόμενο της υπερ-εφαρμογής (over-fitting) κάνοντας ανεπιτυχείς προβλέψεις σε δεδομένα που δεν έχει εκπαιδευτεί. Το φαινόμενο over-fitting παρουσιάζεται όταν το εκπαιδευόμενο μοντέλο έχει πολλές feature variables με σχετικά μεγάλα βάρη b τα οποία προστίθενται στην μέση συνάρτηση κόστους. Για να αντισταθμιστούν τα μεγάλα βάρη μια πρακτική λύση είναι η κανονικοποίηση (regularization). Στην κατηγορία l2 κανονικοποίησης στην μέση συνάρτηση κόστους προστίθενται ένας ακόμα όρος που αποκαλείται ρυθμιστής (regularizer) και ο οποίος ισούται με :

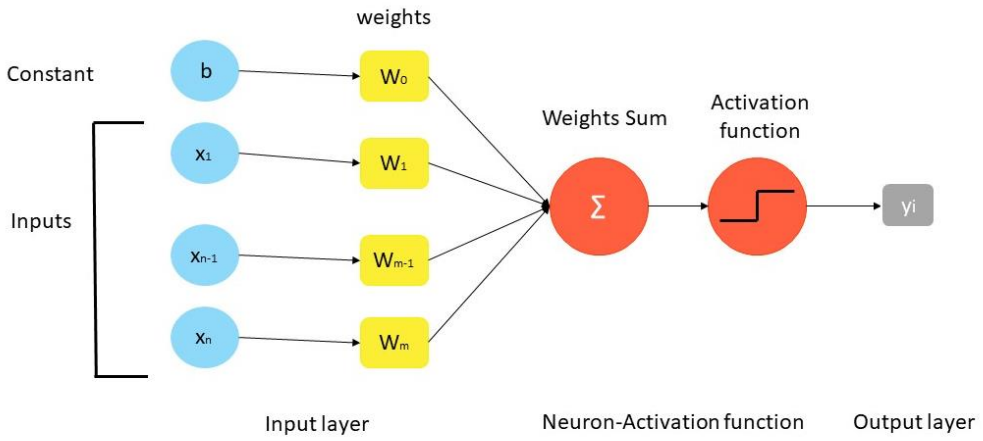
$$L2 = \frac{\lambda}{2} \|b\|^2 = \frac{\lambda}{2} \sum_{i=1}^{n} b_i$$

Για την εκτέλεση του αλγορίθμου της λογιστικής παλινδρόμησης χρησιμοποιείται το πακέτο της Python LogisticRegression της βιβλιοθήκης sklearn. Η multiclass ταξινόμηση συμβαίνει με την μέθοδο OneVsRest ή OneVsAll. Σύμφωνα με την μέθοδο αυτή η ταξινόμηση επιμερίζεται n ξεχωριστές υπο-ταξινομήσεις ίσες με τον αριθμό των κλάσεων. Σε κάθε i υπο-ταξινόμηση τα δεδομένα κατηγοριοποιούνται σύμφωνα με την πιθανότητα να ανήκουν στην κλάση i ή όχι, δηλαδή στις υπόλοιπες. Υπενθυμίζεται ότι στην περίπτωση της multiclass ταξινόμησης κάνουμε την παραδοχή ότι κάθε δεδομένο μπορεί να ανήκει σε μια μόνο κλάση. Βασιζόμενοι σε αυτήν την παραδοχή, το σύνολο δεδομένων κατά την εκτέλεση κάθε i υπο-ταξινόμησης χωρίζεται σε ένα θετικό και ένα αρνητικό σύνολο δεδομένων, έτσι ώστε όσα δεδομένα ανήκουν στην κλάση i να ανήκουν στο θετικό σύνολο και όσα δεν ανήκουν αντίστοιχα στο αρνητικό.

Multilayer perceptron

Perceptron

Αρχικά θα αναλύσουμε την λειτουργία της απλούστερης μορφής perceptron που αποτελείται από έναν νευρώνα. Οι τιμές εισόδου $x_1,x_2..,x_n$ εισέρχονται στο μοντέλο στο στρώμα εισόδου (Input Layer) και έπειτα επιβαρύνονται μέσω κάποιων αρχικοποιημένων βαρών $w_1,w_2,..,w_n$. Τα βάρη είναι πραγματικές τιμές και η φυσική σημασία τους είναι να ενημερώνουν το μοντέλο με το βαθμό «σημαντικότητας» της κάθε τιμής εισόδου στην τιμή εξόδου. Στη συνέχεια κάθε εσωτερικό γινόμενο w_ix_i μεταξύ κάθε τιμής εισόδου με το αντίστοιχο βάρος της οδηγείται στον νευρώνα στον οποίο υπολογίζεται το άθροισμά τους. Έπειτα το βεβαρυμμένο άθροισμα περνά στην συνάρτηση ενεργοποίησης (activation function) της οποίας η τελική τιμή θα καθορίσει αν θα ενεργοποιηθεί ή όχι ο νευρώνας, δηλαδή αν θα σταλεί ή όχι η υπολογισμένη τιμή στο στρώμα εξόδου (Output Layer). Να σημειωθεί ότι η τιμή b ("bias") στο στρώμα εισόδου είναι ένας σταθερός όρος ο οποίος δεν εξαρτάται από καμία τιμή εισόδου και η φυσική σημασίας της είναι να μετατοπίζει την θέση του συνόρου απόφασης που ορίζει η συνάρτηση ενεργοποίησης.

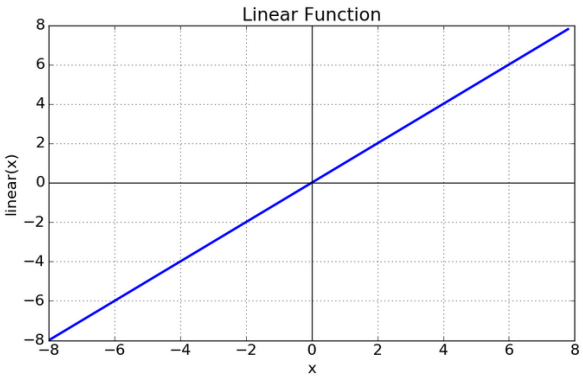


Αρχιτεκτονική μοντέλου MLP.

Όπως αναφέρθηκε και προηγουμένως, η συνάρτηση ενεργοποίησης αποτελεί πολύ σημαντικό στοιχείο του νευρωνικού δικτύου καθώς καθορίζει αν θα παραχθεί ή όχι έξοδος από τον νευρώνα. Υπάρχουν πολλά είδη συναρτήσεων ενεργοποίησης χωρίς να υπάρχει κάποιος συγκεκριμένος περιορισμός για την επιλογή τους. Οι βασικές συναρτήσεις ενεργοποίησης είναι οι ακόλουθες :

- **Γραμμική Συνάρτηση (Linear or Identity function)**

$$\mathbf{f(x) = x}$$



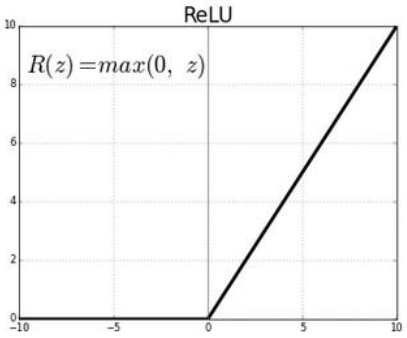
Γραμμική συνάρτηση.

Η γραμμική συνάρτηση f(x) = x είναι η πιο απλή συνάρτηση ενεργοποίησης. Το σύνολο τιμών της είναι (-inf,+inf) και επιστρέφει το ίδιο το βεβαρυμμένο άθροισμα του νευρώνα. Το γεγονός αυτό όμως εμφανίζει πρόβλημα στην εκπαίδευση του νευρωνικού δικτύου όταν χρησιμοποιείται η μέθοδος βελτιστοποίησης απότομης καθόδου (Gradient Descent) καθώς η κλίση της παραμένει σταθερή, ίση με την μονάδα με αποτέλεσμα και ανεξάρτητη από τις τιμές εισόδου x.

Ακόμη, η σταθερή κλίση της γραμμικής συνάρτησης την καθιστά πλήρως προβληματική στην περίπτωση εκπαίδευσης νευρωνικού δικτύου με πολλά στρώματα με πολλούς νευρώνες που συνδέονται μεταξύ τους. Στην περίπτωση αυτή για κάθε στρώμα που ενεργοποιείται η γραμμική συνάρτηση το βεβαρυμμένο άθροισμα οδηγείται ως είσοδος στο επόμενο στρώμα κ.ο.κ. Με τον τρόπο αυτό όμως στο τελευταίο στρώμα εξόδου καταλήγει το βεβαρυμμένο άθροισμα των τιμών εισόδου καθιστώντας τα ενδιάμεσα στρώματα εν τέλει αχρείαστα. Ως εκ τούτου η γραμμική συνάρτηση αποφεύγεται να χρησιμοποιείται σε κρυφά στρώματα.

- **Ανορθωτική Γραμμική Συνάρτηση (Rectifier – ReLU function):**

$$f(x) = \max(0,x)$$



Ανορθωτική Γραμμική Συνάρτηση

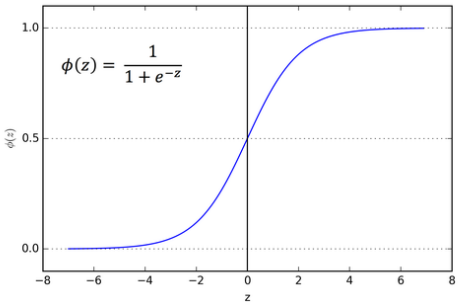
Η ανορθωτική γραμμική συνάρτηση έχει σύνολο τιμών [0,+inf) και μηδενίζει το βεβαρυμμένο άθροισμα του νευρώνα όταν αυτό έχει τιμή μικρότερη του μηδενός και συμπεριφέρεται ως γραμμική όταν αυτό είναι μεγαλύτερο του μηδενός επιστρέφοντας την ίδια την τιμή του. Η συμπεριφορά της

αυτή γίνεται προβληματική στην εκπαίδευση του νευρωνικού δικτύου στην μέθοδο βελτιστοποίησης απότομης καθόδου καθώς η κλίση της συνάρτησης είναι μηδενική στην περίπτωση που το βεβαρυμμένο άθροισμα είναι αρνητικό. Αυτό έχει σαν αποτέλεσμα οι συγκεκριμένοι νευρώνες να μην ενεργοποιούνται και το νευρωνικό δίκτυο να μην μπορεί να διαχειριστεί αρνητικές τιμές εισόδου καθώς οι τιμές των βαρών δεν ανανεώνονται και να παγιδεύονται σε ένα τοπικό ελάχιστο, φαινόμενο γνωστό και ως “dying relu”.

Ωστόσο η ανορθωτική γραμμική συνάρτηση επιλέγεται συχνά λόγω της γρήγορης σύγκλισης που εμφανίζει.

- **Σιγμοειδής (Sigmoid)**

$$\boldsymbol{f(x) = \frac{1}{1 + e^{-x}}}$$

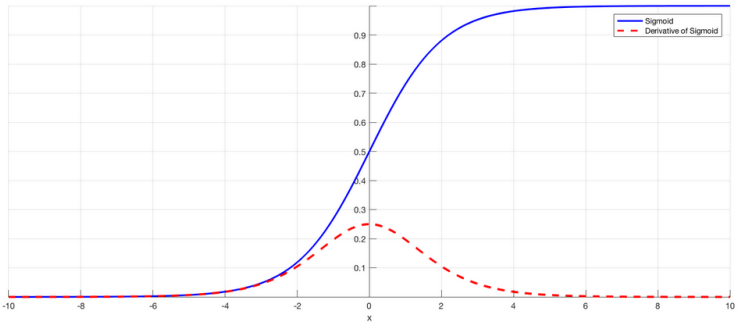


Σιγμοειδής συνάρτηση.

Η σιγμοειδής συνάρτηση έχει σύνολο τιμών [0,1] και λόγω της καμπυλότητας που παρουσιάζει η χρήση της ενδείκνυται σε προβλήματα ταξινόμησης καθώς σε σύγκριση με την συνάρτηση ενεργοποίησης Relu δέχεται και ενδιάμεσες τιμές πέρα από τις 0 και 1. Ενώ η συγκεκριμένη συνάρτηση είναι παραγωγίσιμη και επιτρέπει την εύρεση της κλίσης στο βήμα της καθοδικής .. για ορισμένες περιπτώσεις υπάρχει ο κίνδυνος να συμβεί το φαινόμενο της εξαφανιζόμενης κλίσης (" Vanishing Gradient").

Το φαινόμενο αυτό εμφανίζεται σε περιπτώσεις που οι τιμές εισόδου x είναι είτε πολύ μεγάλες είτε πολύ μικρές με αποτέλεσμα η κλίση της σιγμοειδούς συνάρτησης να παίρνει πολύ χαμηλές τιμές ή σχεδόν μηδενικές. Το συμβάν αυτό επηρεάζει αρνητικά την ανανέωση το βαρών κατά την εκπαίδευση του νευρωνικού δικτύου οδηγώντας είτε σε μια πολύ αργή εκπαίδευση είτε στην "άρνηση" του δικτύου να εκπαιδευτεί περισσότερο.

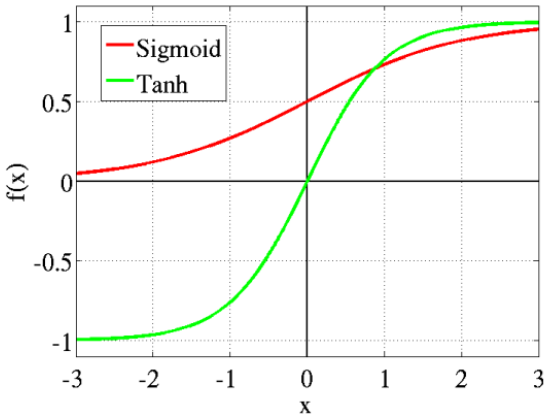
Στο παρακάτω διάγραμμα παρατίθενται η σιγμοειδής συνάρτηση μαζί με την συνάρτηση της κλίσης της απ' όπου είναι φανερό ότι για ακραίες τιμές εισόδου x η κλίση συγκλίνει στο μηδέν:



Σιγμοειδής συνάρτηση με συνεχή γραμμή και η κλίση της με διακεκομμένη.

- **Υπερβολική Εφαπτομένη (Tanh)**

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$



Υπερβολική και σιγμοειδής συνάρτηση.

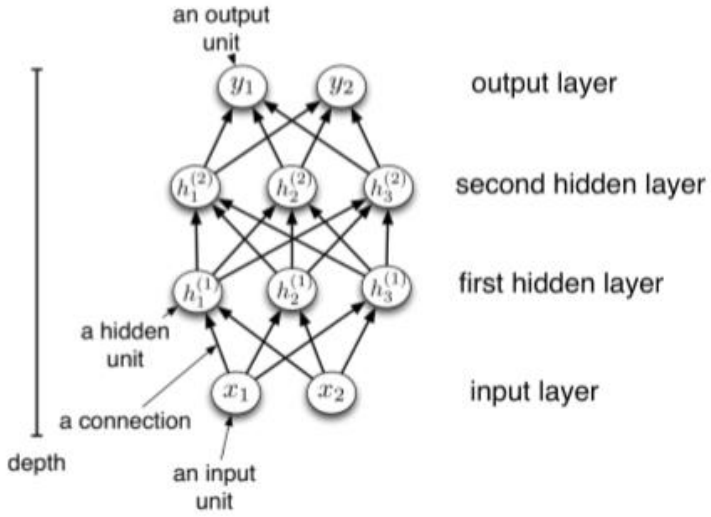
Η συνάρτηση της υπερβολικής εφαπτομένης έχει πεδίο τιμών [-1,1] και για τους ίδιους λόγους με την σιγμοειδή συνάρτηση μπορεί να εκφράσει πιθανότητες και χρησιμοποιείται σε προβλήματα ταξινόμησης. Ωστόσο συγκριτικά με την σιγμοειδή συνάρτηση λόγω της κεντραρισμένης στο μηδέν θέσης των τιμών εξόδου της y διαχειρίζεται αποτελεσματικότερα "ισχυρά" αρνητικές, ουδέτερες και "ισχυρά" θετικές τιμές εισόδου x. Τέλος είναι ομοίως με την σιγμοειδή επιρρεπής στο φαινόμενο της εξαφανιζόμενης κλίσης.

Πολυστρωματικά Perceptron (Multilayer Perceptron)

Τα πολυστρωματικά perceptron (Multilayer Perceptron, MLP) είναι νευρωνικά δίκτυα προσοτροφοδότησης (Feedforward NN) και αποτελούνται από πολλούς perceptron οι οποίοι σχηματίζουν τα στρώματα (layers). Το πρώτο στρώμα ενός MLP καλείται στρώμα εισόδου (Input Layer) και ο αριθμός των νευρώνων ισοδυναμεί με τις τιμές εισόδου $x_1, x_2, \dots, x_n$ του μοντέλου. Τα επόμενα στρώματα αποκαλούνται κρυφά στρώματα (Hidden Layers) και ο αριθμός τους όσο και ο αριθμός των νευρώνων τους ποικίλει ανάλογα με την αρχιτεκτονική του MLP που επιθυμεί ο χρήστης. Ο αριθμός των κρυφών στρωμάτων σε ένα NN ταυτόχρονα αποκαλείται «βάθος» (depth) του NN και ο αριθμός των perceptron πλάτος (width). Το τελευταίο στρώμα, το στρώμα εξόδου (Output Layer) εξάγει το αποτέλεσμα του μοντέλου και ο αριθμός των νευρώνων του ποικίλει ανάλογα με το πρόβλημα που τίθενται από τον χρήστη. Στα προβλήματα ταξινόμησης ο αριθμός τους αντιστοιχεί με τον αριθμό των κλάσεων, επομένως στο πείραμα αυτό θα είναι 169.

Τέλος, οι perceptron κάθε στρώματος συνδέονται με εκείνους του επόμενου στρώματος, με τη χρήση συνάψεων-ακμών, στην οποία αντιστοιχεί και ένα βάρος. Σε κάθε perceptron εκτελείται η λειτουργία που περιεγράφηκε στην προηγούμενη παράγραφο.

Παρακάτω απεικονίζεται η αρχιτεκτονική ενός MLP με δυο κρυφά επίπεδα :



Αρχιτεκτονική NN.

Feedfoward Μέθοδος

Αυτό είναι το πρώτο βήμα για την εκπαίδευση του MLP, στο οποίο με αφετηρία το στρώμα εισόδου υπολογίζονται οι ενεργοποιήσεις κάθε νευρώνα σε κάθε διαδοχικό κρυφό στρώμα μέχρι το τελικό στρώμα εξόδου. Θα αναλύσουμε την περίπτωση της αρχιτεκτονικής ενός MLP με δυο κρυφά στρώματα. Οι ενεργοποιήσεις α_j των νευρώνων που σχηματίζονται στο πρώτο κρυφό στρώμα

ισοδυναμούν με m γραμμικούς συνδυασμούς των τιμών εισόδου $x_1, x_2, \dots, x_n$ με τα βάρη $w_1, w_2, .., w_m$ των συνάψεων-ακμών μεταξύ του στρώματος εισόδου και του πρώτου κρυφού στρώματος ως ακολούθως:

$$a_j = \sum_{i=1}^{n} w_{ji}^{(1)} x_i + w_{j0}^{(1)}$$

για j = 1,2,…,m

όπου w_{j0} είναι η τιμή bias και ο εκθέτης (1) υποδηλώνει ότι οι παράμετροι ανήκουν στο πρώτο κρυφό στρώμα.

Έπειτα κάθε ενεργοποίηση μετατρέπεται με χρήση της συνάρτησης ενεργοποίησης που αναλύθηκε προηγουμένως σε τιμή εξόδου του νευρώνα σύμφωνα με τον ακόλουθο τύπο:

$$z_j = h(a_j)$$

,όπου h η συνάρτηση ενεργοποίησης.

Τα βήματα αυτά επαναλαμβάνονται και για το δεύτερο κρυφό στρώμα όπου οι ενεργοποιήσεις κάθε νευρώνα του υπολογίζονται :

$$a_k = \sum_{i=1}^{m} w_{ji}^{(2)} z_i + w_{j0}^{(2)} \quad \text{για k} = 1,2,\dots,m$$

Εν τέλει οι τιμές εξόδου του νευρωνικού δικτύου υπολογίζονται χρησιμοποιώντας την συνάρτηση ενεργοποίησης f :

$$y = f\left(\sum_{j=0}^{m} w_{kj}^{(2)}\, h\left(\sum_{i=0}^{n} w_{ji}^{(1)} x_i\right)\right)$$

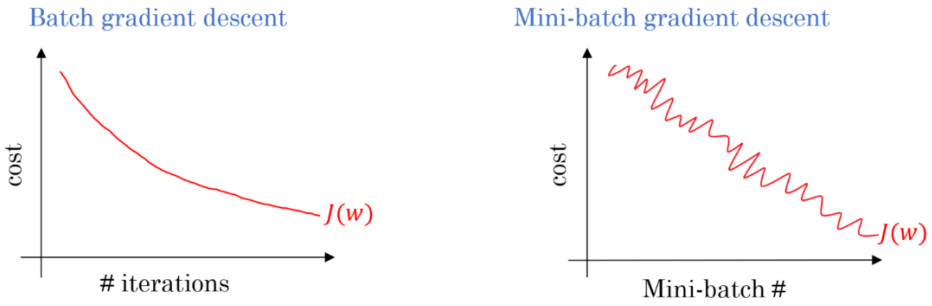
Στοχαστική κατάβαση Δυναμικού/Κλίσης (Stochastic Gradient Descent –SGD)

Στον αλγόριθμό κατάβασης κλίσης (GD) στον υπολογισμό της συνολικής απώλειας για την ανανέωση των βαρών χρησιμοποιείται όλο το σύνολο δεδομένων εκπαίδευσης. Το γεγονός αυτό όμως καθιστά την εκτέλεση μιας εποχής πολύ αργή με αντίκτυπο στην εξαντλητική ταυτόχρονα χρήση του CPU του υπολογιστή. Ως προτεινόμενη λύση στο πρόβλημα αυτό αποτελεί ο αλγόριθμος της στοχαστικής κατάβασης κλίσης, ως υποπερίπτωση του γενικού αλγορίθμου GD, στον οποίο για τον υπολογισμό της κλίσης σε κάθε εποχή χρησιμοποιείται ένα ή λίγα δείγματα εκπαίδευσης (mini batch SGD). Παρόλο που ο SGD μπορεί να χρειάζεται περισσότερες επαναλήψεις για να την

σύγκλισή του σε τοπικό ή ολικό ελάχιστο επειδή δεν χρησιμοποιούνται όλα τα δεδομένα εκπαίδευσης όπως στον GD, έχει αποδειχθεί ότι είναι αποδοτικός και γρηγορότερος στην εκπαίδευση του νευρωνικού δικτύου.

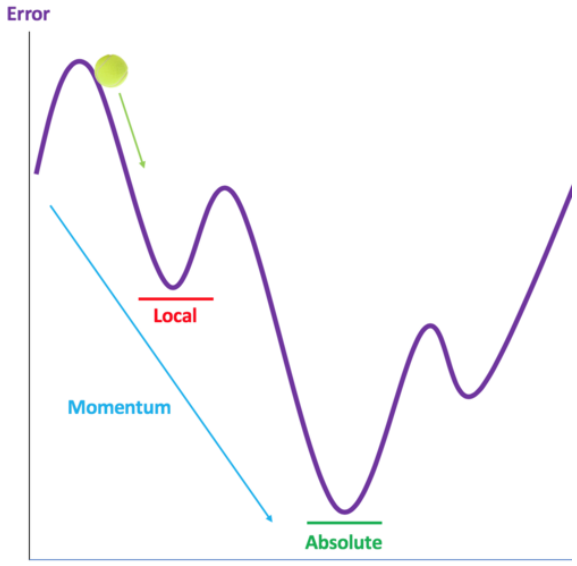
Ορμή (momentum)

Ο αλγόριθμος SGD σε αντίθεση με τον γενικό GD αντιμετωπίζει καλύτερα τον διαχωρισμό του τοπικού ελαχίστου με το ολικό ελάχιστο, χάρη στις μικρές ταλαντώσεις που συμβαίνουν στα βήματα εκπαίδευσης (learning rates) όπως απεικονίζονται στο παρακάτω σχήμα :



Εκμάθηση με τον αλγόριθμο GD στα αριστερά και τον SGD στα αριστερά.

Ωστόσο εξαιτίας του θόρυβου που σχηματίζουν οι μικρές αυτές ταλαντώσεις ο SGD συχνά δεν ανταποκρίνεται όταν στο σχήμα της συνάρτησης κόστους εμφανίζονται «χαράδρες», δηλαδή σε περιοχές όπου η επιφάνεια καμπυλώνεται πολύ πιο απότομα σε μία διάσταση από ότι σε μία άλλη, και με αποτέλεσμα να «κολλάει» σε τοπικό ελάχιστο (local minimum problem). Παρακάτω φαίνεται σχηματικά πάνω στην καμπύλη πώς μοιάζουν αυτές οι περιοχές:

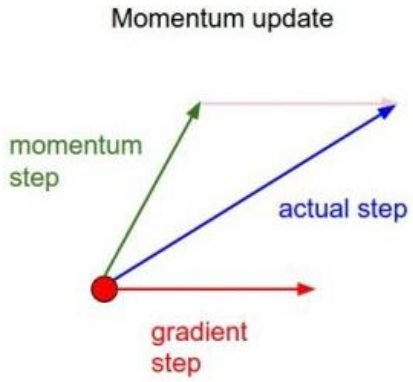


Στο σημείο αυτό η μέθοδος ορμή (momentum) βοηθά στην μείωση αυτών των ταλαντώσεων-θορύβων και στην επιτάχυνση του 'SGD' στη σχετική κατεύθυνση. Για την υλοποίηση αυτής την μεθόδου προστίθενται στην εξίσωση ενημέρωσης της κλίσης του SGD ένας ακόμα όρος, το εσωτερικό γινόμενο του σταθερού αριθμού γ (momentum) με το διάνυσμα ενημέρωσης του προηγούμενου βήματος χρόνου:

$$v_t = \eta \nabla L(w) + \gamma v_{t-1}$$

$$w = w - v_{t-1}$$

όπου vt το τρέχον διάνυσμα ενήμερωσης και vt-1 του προηγούμενου βήματος χρόνου, w η παράμετρος προς βελτιστοποίηση και γ η ορμή (momentum) που παίρνει τιμές στο σύνολο [0,1]. Όσο πιο κοντά στο 1 τόσο περισσότερο απαλύνονται οι ταλαντώσεις-θόρυβοι. Η ενημέρωση της κλίσης γίνεται εν τέλει από την πρόσθεση του βήματος από την μέθοδο της ορμής και του βήματος του κλασικού SGD :



Η μέθοδος της ορμής μοιάζει με μια σφαίρα που αφήνεται να πέσει από έναν λόφο. Καθώς κυλάει προς τα κάτω συσσωρεύει την ορμή και κατεβαίνει ταχύτερα μέχρι να φτάσει στην τελική της ταχύτητα αν υπάρξει αντίσταση από τον αέρα. Αντίστοιχα και στις ενημερώσεις των παραμέτρων η ορμή αυξάνεται για τις διαστάσεις των οποίων οι κλίσεις δείχνουν προς τις ίδιες κατευθύνσεις ενώ αντίθετα σε διαστάσεις όπου οι κλίσεις αλλάζουν κατευθύνσεις οι ενημερώσεις μειώνονται. Ως εκ τούτου μειώνονται οι ταλαντώσεις-θόρυβοι και επιτυγχάνεται ταχύτερη σύγκλιση.

Μέθοδος Adam

Ο αλγόριθμος βελτιστοποίησης Adam (Adaptive Moment Estimation) χρησιμοποιείται για την ανανέωσης των βαρών του NN και συνδυάζει την προσαρμογή του ρυθμού μάθησης η του SGD, με έναν όρο ορμής (momentum). Πιο αναλυτικά ο αλγόριθμος αυτός εκτιμά τις αναμενόμενες τιμές της $1^{ης}$ (g_t) και $2^{ης}$ (g_t^2) ροπής αδράνειας της κλίσης με τη βοήθεια δυο κινούμενων μέσων όρων, του m_t για την g_t και του v_t για την g_t^2. Ο ψευδοκώδικας που περιγράφει την μεταβολή των παραμέτρων στο βήμα t είναι ο ακόλουθος :

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

,όπου β1,β2 οι ρυθμοί αποσύνθεσης (decay rates).

Οι παράμετροι $\widehat{m_t}$, $\widehat{v_t}$ είναι οι διορθωμένες εκτιμήσεις με σκοπό τη διόρθωση του σφάλματος πόλωσης που προκύπτει από την αρχικοποίηση των στο μηδέν.

Οι δημιουργοί του 'Adam' προτείνουν τις προκαθορισμένες τιμές 0.9, 0.999 και 10-8 για το β1, το β2 και το ε αντίστοιχα.

Εκπαίδευση του MLP

Τα βήματα που χρησιμοποιεί ένα NN για να παράγει προβλέψεις απαριθμούνται ως ακολούθως :

1. Πρώτα καθορίζεται η αρχιτεκτονική του νευρωνικού δικτύου, δηλαδή από πόσα κρυφά στρώματα θα αποτελείται και πόσους νευρώνες θα περιέχει το κάθε κρυφό στρώμα.
2. Τροφοδοτείται το NN με τα δείγματα εκπαίδευσης και αρχικοποιούνται οι τιμές όλων των βαρών με τυχαίο τρόπο και πολύ μικρά μεγέθη.
3. Χρησιμοποιείται η μέθοδος Feedfoward Propagation έτσι ώστε να ενεργοποιηθούν οι νευρώνες του MLP και να παραχθούν οι πρώτες προβλέψεις.
4. Συγκρίνονται οι προβλέψεις που παράχθηκαν από το προηγούμενο βήμα με τις πραγματικές τιμές y του σετ αξιολόγησης και υπολογίζεται συνάρτηση κόστους.
5. Χρησιμοποιείται η μέθοδος BackPropagation έτσι ώστε να ανανεωθούν τα βάρη που ελαχιστοποιούν την συνάρτηση κόστους σύμφωνα με την μέθοδο βελτιστοποίησης που έχει επιλεχθεί.

Μόλις εφαρμοστούν όλα τα βήματα εκμάθησης σε όλα τα δεδομένα τότε έχει συμπληρωθεί μια εποχή του NN, η οποία ταυτόχρονα είναι και μία από τις υπερπαραμέτρους του. Η εκπαίδευση του NN εκπληρώνεται μόλις ολοκληρωθεί ο αριθμός των εποχών που έχει ορίσει ο χρήστης.

Ως αναφορά το πρώτο βήμα δεν υπάρχει κάποιος αποδεδειγμένος κανόνας για την επιλογή της αρχιτεκτονικής του NN, ωστόσο υπάρχουν κάποιοι εμπειρικοί κανόνες που συχνά φαίνονται χρήσιμοι :

- Είναι προτιμότερο να ξεκινάμε με μια απλή αρχιτεκτονική με ένα κρυφό στρώμα και μικρό αριθμό νευρώνων και να προσθέτουμε σταδιακά κρίνοντας από την βελτίωση ή όχι των μετρικών απόδοσης του NN.
- Η αρχιτεκτονική του NN επηρεάζεται από το μέγεθος του συνόλου εκπαίδευσης. Για το λόγο αυτό προκειμένου να αποφευχθεί το φαινόμενο της υπερεκπαίδευσης προτιμάται για μικρά σύνολα εκπαίδευσης να χρησιμοποιούνται απλούστερες αρχιτεκτονικές και για μεγάλα πολυπλοκότερες.
- Για την επιλογή του αριθμού των νευρώνων στα κρυφά στρώματα δοκιμάζουμε το μέσο όρο των νευρώνων του προηγούμενου και του επόμενου στρώματος.

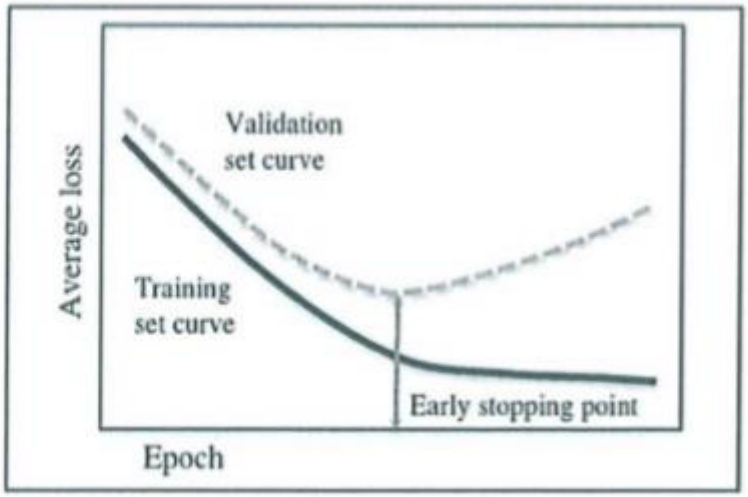
Υπερεκπαίδευση Νευρωνικού Δικτύου

Τα NN είναι συχνά ευαίσθητα στο φαινόμενο της υπερεκπαίδευσης, γνωστό και ως overfitting. Το φαινόμενο αυτό συμβαίνει όταν κατά την διάρκεια της εκπαίδευσης του το NN «εξειδικευτεί» υπερβολικά στο δείγμα δεδομένων που του δόθηκε με αποτέλεσμα να μην είναι ικανό να γενικεύσει σε διαφορετικά άγνωστα δείγματα.

Για τον εντοπισμό του overfitting το σύνολο δεδομένων χωρίζεται σε δυο υποσύνολα , το σύνολο που χρησιμοποιείται για την εκπαίδευση του NN (training set) και το σύνολο που χρησιμοποιείται για την επαλήθευσή του (test set). Όταν η απόδοση-ακρίβεια του μοντέλου στο training set είναι πολύ καλύτερη από αυτή του test set, τότε συμβαίνει το φαινόμενο overfitting.

Για να αποφευχθεί το overfitting από την βιβλιογραφία συνίσταται το NN να έχει όσο το δυνατόν απλούστερη αρχιτεκτονική και να δέχεται μεγάλο αριθμό δεδομένων εκπαίδευσης. Ωστόσο κατά το βήμα της βελτιστοποίησης του αλγορίθμου του NN είναι πιθανό κάποιες υπερ-παράμετροι να οδηγούν επίσης σε overfitting.

Μια υπερ-παράμετρος που επηρεάζει την ικανότητα του NN στην γενίκευση (generalization) είναι ο αριθμός των εποχών, στην περίπτωση που είναι πολύ μεγάλος. Αυτό συμβαίνει γιατί το NN όταν έχει πολλές εποχές συνεχίζει να εκπαιδεύεται στο δοσμένο training set ακόμα και όταν έχει συγκλείσει σε κάποια τιμή με αποτέλεσμα να συνεχίζουν να ανανεώνονται τα βάρη και το NN να προσαρμόζεται ή «εξειδικεύεται» στο training set. Για να αποφευχθεί η επιλογή πολλών εποχών, βοηθάει η γραφική αναπαράσταση της συνάρτησης κόστους συναρτήσει των εποχών για το training set και test set, ώστε να επιλεχθεί ο αριθμός των εποχών στο σημείο που η συνάρτηση κόστους έχει και χαμηλή τιμή και οι αποδόσεις στα training set και test set είναι κοντινές. Το σημείο επιλογής του βέλτιστου αριθμού εποχών είναι το «early stopping point» όπως στο σχήμα:



Γραφική αναπαράσταση της συνάρτησης κόστους συναρτήσει των εποχών

Support Vector Machines (SVM)

Οι μηχανές διανυσμάτων υποστήριξης (Support Vector Machines - SVM) αποτελούν ταξινομητές οι οποίοι κατασκευάζουν μια γραμμική ή μια μη γραμμική διαχωριστική επιφάνεια στο χώρο εισόδου του συνόλου δεδομένων (επιφάνεια δηλαδή που χωρίζει στις κλάσεις προτύπων). Σκοπός του αλγορίθμου SVM είναι να βρει εκείνο το υπερεπίπεδο, το οποίο θα έχει την μέγιστη δυνατή απόσταση από τις παρατηρήσεις κάθε κλάσης, δηλαδή το μέγιστο περιθώριο (margin) διαχωρισμού μεταξύ των κλάσεων.

Γραμμικά Διαχωρίσιμα Δεδομένα

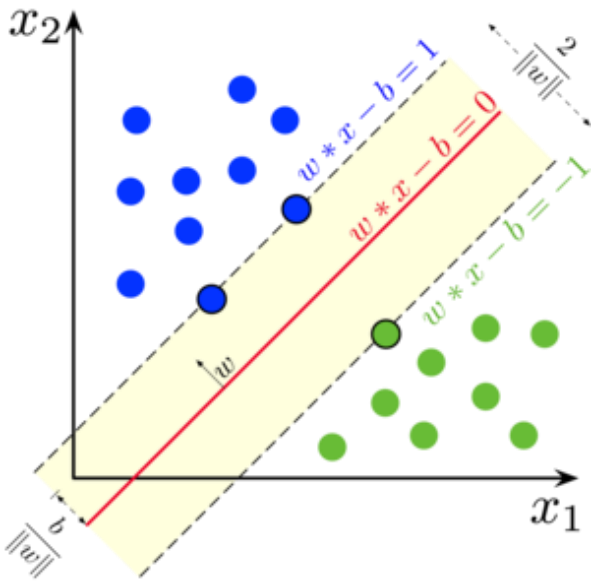
Το πιο απλό μοντέλο SVM, είναι ο αποκαλούμενος ταξινομητής μέγιστου περιθωρίου (maximal margin classifier). Ο συγκεκριμένος ταξινομητής έχει εφαρμογή μόνο σε γραμμικά διαχωρίσιμα δεδομένα.

Αρχικά θα αναλύσουμε το απλό πρόβλημα ταξινόμησης των δεδομένων σε δυο κλάσεις (έστω θετική και αρνητική) σε έναν διανυσματικό χώρο 2 διαστάσεων. Ορίζουμε τα δεδομένα εκπαίδευσης ως $[x_i,y_i]$, για i= 1,2,…,n και $y_i \in \{-1,1\}$, $x_i \in R^d$. Η λειτουργία του SVM περιγράφεται ως ακολούθως είναι το εξής:

Αρχικά βρίσκουμε μία οποιαδήποτε επιφάνεια η οποία να διαχωρίζει γραμμικά τα σημεία στις δύο κλάσεις, η οποία είναι της μορφής :

$$g(x) = w^T x + b = 0$$

, όπου το w είναι κάθετο στο υπερεπίπεδο και b/||w|| αποτελεί την κάθετη απόσταση από το υπερεπίπεδο μέχρι το σημείο αναφοράς, με ||w|| να αποτελεί την ευκλείδεια νόρμα του w.



Παράδειγμα ταξινόμησης μοντέλου SVM.

Τα διανύσματα στήριξης (support vectors) είναι τα δεδομένα εκπαίδευσης τα οποία βρίσκονται πλησιέστερα προς τη διαχωριστική επιφάνειας ορίζοντας μια ζώνη διαχωρισμού. Ο στόχος του μοντέλου SVM είναι να προσανατολίσει την διαχωριστική επιφάνεια κατά τέτοιο τρόπο ώστε να μεγιστοποιείται η ζώνη διαχωρισμού και για το λόγο αυτό τα διανύσματα στήριξης είναι τα μόνα που θα καθορίσουν την τελική ταξινόμηση των κλάσεων. Το περιθώριο που σχηματίζεται από την απόσταση κάθε κλάσης με την διαχωριστική επιφάνια σχηματικά αποτυπώνεται ως $\frac{1}{\|w\|}$.

Για να υλοποιηθεί η ταξινόμηση στις δυο κλάσεις (θετική και αρνητική) όλα τα δεδομένα εκπαίδευσης θα πρέπει να τηρούν τους ακόλουθους περιορισμούς :

- $w^T x + b \geq 1$ για κάθε x που ανήκει στην θετική κλάση
- $w^T x + b \leq -1$ για κάθε x που ανήκει στην αρνητική κλάση

,οι οποίοι μπορούν να συνδυαστούν σε μια ανισότητα

$$y_i(w^T x_i + b) - 1 \geq 0 \text{ για i=1,2,…,n}$$

και από την τελευταία σχέση ορίζεται η συνάρτηση κόστους του μοντέλου

$$J(w, b) = \frac{1}{2}\|w\|^2$$

Προφανώς επειδή η ελαχιστοποίηση της ευκλείδειας νόρμας οδηγεί στην μεγιστοποίηση του περιθωρίου, σκοπός είναι η ελαχιστοποίηση της συνάρτησης κόστους.

Σε αυτό το σημείο λόγω της ύπαρξης περιορισμών θα προχωρήσουμε σε μια αναδιατύπωση του προβλήματος με χρήση των Lagrangians. Επομένως οι παραπάνω περιορισμοί θα αντικατασταθούν από τους πολλαπλασιαστές Lagrange και με τον τρόπο αυτό τα δεδομένα εκπαίδευσης θα εμφανιστούν μόνο με τη μορφή γινομένων μεταξύ διανυσμάτων. Εισάγουμε τους θετικούς πολλαπλασιαστές Lagrange αi,i = 1,...,n ένα για κάθε περιορισμό. Σύμφωνα με την θεωρία για να σχηματιστεί η Lagrangian, οι ανισότητες της μορφής $x \geq 0$ πολλαπλασιάζονται με θετικούς πολλαπλασιαστές Lagrange και αφαιρούνται από την αντικειμενική συνάρτηση (συνάρτηση κόστους), ενώ οι ισότητες δεν δέχονται κάποια αλλαγή. Επομένως η συνάρτηση κόστους αναδιατυπώνεται ως εξής :

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} a_i y_i (x_i w + b) + \sum_{i=1}^{n} a_i$$

Από την αναδιατυπωμένη εξίσωση συμπεραίνουμε ότι πλέον η ελαχιστοποίηση γίνεται ως προς τις παραμέτρους w και b με την προϋπόθεση για τους συντελεστές Lagrange ότι είναι θετικοί. Διαφορίζοντας λοιπόν την LP ως προς το w και το b, και θέτοντας τις παραγώγους ίσες με το μηδέν ¨

$$\frac{\partial L_P}{\partial w} = 0 \ \rightarrow w = \sum_{i=1}^{n} a_i y_i x_i$$

$$\frac{\partial L_P}{\partial b} = 0 \ \rightarrow \sum_{i=1}^{n} a_i y_i = 0$$

Αντικαθιστώντας στην συνάρτηση κόστους LP αναδιατυπώνουμε μια νέα μορφή η οποία πλέον εξαρτάται μόνο από τις μεταβλητές α :

$$L_D = \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i,j}^{n} a_i a_j y_i y_j x_i x_j$$

$$= \sum_{i=1}^{n} a_i - \frac{1}{2} \sum_{i,j}^{n} a_i H_{i,j} a_j$$

, όπου $H_{i,j} = y_i y_j x_i x_j$

$$= \sum_{i=1}^{n} a_i - \frac{1}{2} \alpha^T H \alpha$$

Η τελική μορφή της συνάρτησης κόστους L_D απαιτεί μόνο τον υπολογισμό του γινομένου όλων των διανυσμάτων εισόδου και η τελική λύση του μοντέλου έγκειται στην μεγιστοποίησή της:

$$\max[\sum_{i=1}^{n} a_i - \frac{1}{2} a^T H \, a]$$

Από την επίλυσή της υπολογίζεται το α και από την επίλυση της L_P το w. Αντικαθιστώντας τις επιλυμένες L_D και L_P εξισώσεις στον αρχικό περιορισμό του προβλήματος και υπολογίζοντας το μέσο όρο όλων των δεδομένων εκπαίδευσης βρίσκουμε το b.

Με τον τρόπο αυτό έχουν πλέον βρεθεί οι μεταβλητές w και b που ορίζουν το βέλτιστο διαχωριστικό προσανατολισμό της διαχωριστικής επιφάνειας.

Μη γραμμικά διαχωρίσιμα Δεδομένα

Στην περίπτωση που έχουμε δεδομένα εκπαίδευσης τα οποία δεν είναι πλήρως γραμμικά διαχωρίσιμα, τότε ο αλγόριθμος προσπαθεί να επιτεύξει μεγαλύτερο περιθώριο (margin) στο οποίο εσκεμμένα κατηγοριοποιούνται λάθος κάποια σημεία . Αυτό στην πράξη σημαίνει ότι χαλαρώνονται οι προηγούμενοι περιορισμοί με την εισαγωγή μιας θετικής μεταβλητής χαλάρωσης (slack variable) ξ_i για να επιτρέπουν ελαφρώς μη ταξινομημένα σημεία. Η μεταβλητή ξ_i για i=1,2,…,n επιτρέπει σε κάθε σημείο να τοποθετηθεί μέσα στο σύνορο ή σε λάθος κατηγορία :

- $x_i w + b \geq 1 - \xi_i$ για κάθε x που ανήκει στην θετική κλάση
- $x_i w + b \leq -1 + \xi_i$ για κάθε x που ανήκει στην αρνητική κλάση

, όπου $\xi_i \geq 0 \quad \forall i$

Αυτές οι εξισώσεις συνδυάζονται ως εξής :

$$y_i(x_i w + b) - 1 + \xi_i \geq 0 \quad \text{, όπου } \xi_i \geq 0 \quad \forall i$$

Αυτή η μέθοδος του αλγορίθμου SVM αποκαλείται «μαλακό» SVM περιθώριο (soft margin SVM). Ωστόσο για να αποτραπεί η υπερβολική χρήση των λάθος τοποθετημένων σημείων, ορίζεται η σταθερά C ως «ποινή» η οποία θέτει τους όρους για τη μεγιστοποίηση του soft margin και την ελαχιστοποίηση των λάθος κατηγοριοποιημένων σημείων. Λαμβάνοντας αυτά υπόψη, η αντικειμενική συνάρτηση που παρουσιάστηκε στην περίπτωση των γραμμικών διαχωριζομένων δεδομένων προσαρμόζεται στην ακόλουθη μορφή :

$$min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i$$

όπου όσο μεγαλύτερες είναι οι τιμές του C, τόσο περισσότερο βάρος δίνεται στην αποφυγή των εσφαλμένων κατηγοριοποιήσεων, στις οποίες τα δύο σημεία που βρίσκονται κοντά στη διαχωριστική επιφάνεια επηρεάζουν άμεσα τον προσανατολισμό της και την μετατοπίζουν πολύ κοντά στα υπόλοιπα σημεία. Η αναδιατύπωση της Lagrangian :

$$L_P = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} a_i [y_i(x_i w + b) - 1 + \xi_i] - \sum_{i=1}^{n} \mu_i \xi_i$$

,όπου $\mu_i \geq 0$ οι πολλαπλασιαστές Lagrange για την επιβολή της θετικότητας της μεταβλητής ξ_i.

Διαφορίζοντας την ως προς τα w, b και ξ_i και θέτοντας τις παραγώγους ίσες με το μηδέν :

- $\frac{\partial L_P}{\partial w} = 0 \ \rightarrow w = \sum_{i=1}^{n} a_i y_i x_i$
- $\frac{\partial L_P}{\partial b} = 0 \ \rightarrow \sum_{i=1}^{n} a_i y_i = 0$
- $\frac{\partial L_P}{\partial \xi_i} = 0 \ \rightarrow \ C = \alpha_\iota + \mu_i$

Ομοίως με προηγουμένως αντικαθιστώντας τις δυο πρώτες παραγώγους στην L_P καταλήγουμε στην L_D, με τον πρόσθετο περιορισμό που προκύπτει από την τρίτη παράγωγο ότι για $\mu_i \geq 0$ συνεπάγεται ότι $a \leq C$. Ομοίως χρειάζεται να λύσουμε την μεγιστοποίηση :

$$\max[\sum_{i=1}^{n} a_i - \frac{1}{2} a^T H \, a]$$

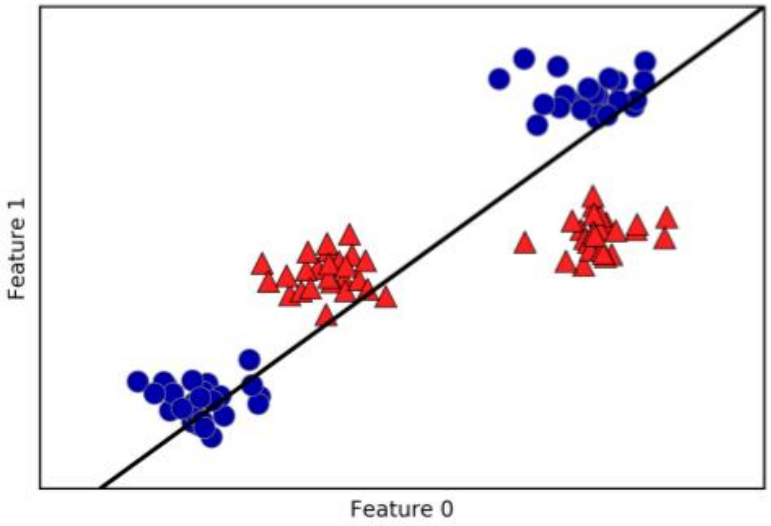
Η παράμετρος b υπολογίζεται με τον ίδιο τρόπο όπως στην περίπτωση των γραμμικά διαχωριζόμενων δεδομένων.

Με την συνάρτηση κόστους L_D όπως αναφέραμε και στην περίπτωση των γραμμικά διαχωρίσιμων δεδομένων, πλέον η επίλυση του μοντέλου βασίζεται μόνο στα στοιχεία των εσωτερικών γινομένων των διανυσμάτων εισόδου $x_i * x_j$, τα οποία έχουν χαρτογραφηθεί σε ένα ευκλείδειο Η χώρο με την μορφή $\Phi:\Phi:R^D \rightarrow H$.

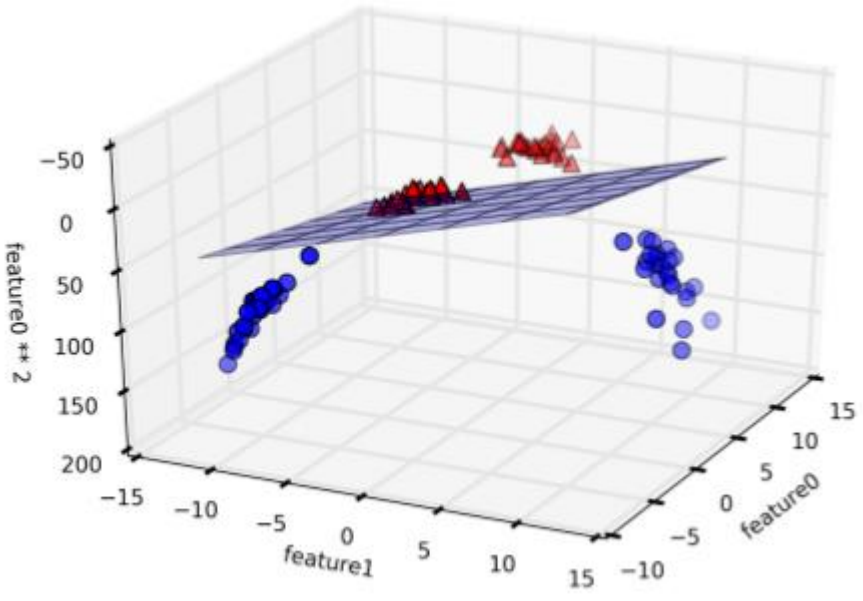
Τα γραμμικά μοντέλα ωστόσο μπορεί να αποφανθούν περιοριστικά σε χώρους μικρών διαστάσεων καθώς η διαχωριστική γραμμή ή υπερεπίπεδο δεν είναι αρκετά ευέλικτα στην λήψη απόφασης.

Για τον λόγο αυτό στην περίπτωση των μη γραμμικά διαχωρίσιμων δεδομένων γίνεται η χρήση ενός «πυρήνα» (Kernel) $K(x_i, x_j) = \Phi(x_i)\Phi(x_j)$, φαινόμενο γνωστό και ως «*kernel trick*», η οποία επιτρέπει να υπολογιστούν τα εσωτερικά γινόμενα σε μη γραμμικό χώρο. Η μέθοδος αυτή είναι μια επέκταση του γραμμικού μοντέλου στην οποία προστίθενται μη γραμμικά στοιχεία στην αναπαράσταση των δεδομένων όπως πολυώνυμα μεγάλου βαθμού των δεδομένων εισόδου κ.α.

Για παράδειγμα, έστω η περίπτωση ταξινόμησης ενός γραμμικού ταξινομητή σε δυσδιάστατο χώρο δυο κλάσεων (feature_0, feature_1). Όπως φαίνεται στο σχήμα ο γραμμικός ταξινομητής αδυνατεί να διαχωρίσει επιτυχώς τα δεδομένα :



Ως λύση υπολογίζεται το βαθμωτό γινόμενο δευτέρου βαθμού (feature_1^2) των στοιχείων της δεύτερης κλάσης και προστίθενται στην αναπαράσταση των δεδομένων. Η αναπαράσταση των δεδομένων εισόδου επεκτείνεται στον τρισδιάστατο χώρο της μορφής (feature_0, feature_1, feature_1^2). Στην νέα αναπαράσταση είναι πλέον δυνατή η ταξινόμηση των κλάσεων με χρήση του γραμμικού ταξινομητή, όπως φαίνεται και στο παρακάτω σχήμα:



Με την μέθοδο «kernel trick» λοιπόν, μέσω του υπολογισμού της απόστασης (βαθμωτών γινομένων) των δεδομένων γίνεται η εκμάθηση του ταξινομητή σε έναν χώρο αναπαράστασης υψηλότερης διάστασης χωρίς να χρειαστεί να γινεί ο μετασχηματισμός στην αναπαράσταση αυτή (ο οποίος είναι

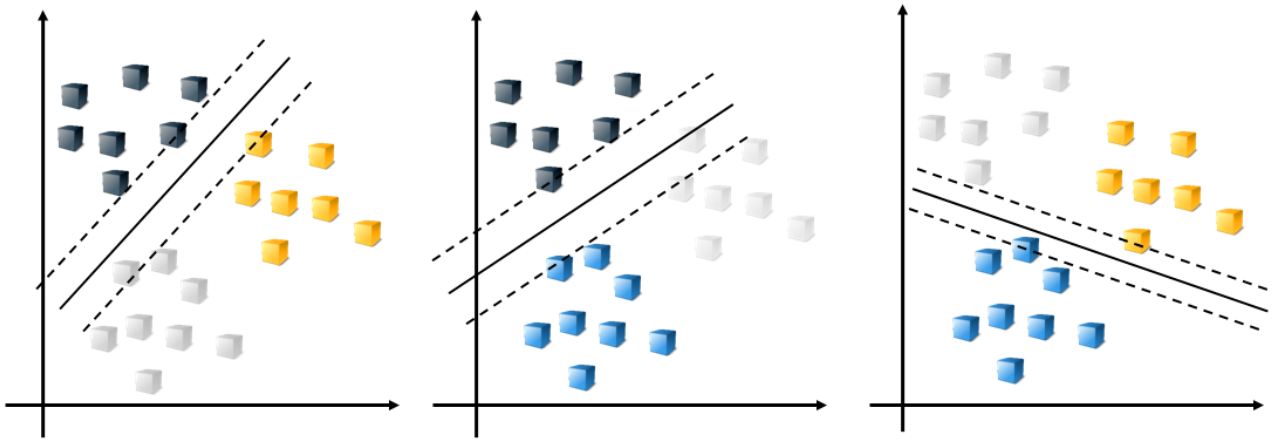
υπολογιστικά δαπανηρός). Κάποια παραδείγματα «πυρήνων» που υπολογίζουν την απόσταση μεταξύ δεδομένων είναι τα ακόλουθα :

- Πολυωνυμικός (Polynomial) : $K(x_i, x_j) = (x_i{}^T x_j + 1)^q$
- Ακτινωτής βάσης (radial basis function) : $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{\sigma^2})$
- Υπερβολική εφαπτομένη (hyperbolic tangent) : $K(x_i, x_j) = \tanh(\beta x_i{}^T x_j + \gamma)$

Δεν υπάρχει γενικός κανόνας για την επιλογή kernel, καθώς όλα εξαρτώνται από τα δεδομένα. Συνήθως αρχικά δοκιμάζεται ένας γραμμικός kernel και μετά εξετάζεται κάποιος πιο σύνθετος.

Multi-class SVM

Ομοίως με την περίπτωση του αλγορίθμου Logistic Regression, η multiclass ταξινόμηση που παρέχει το πακέτο Support vector machines (SVMs) της βιβλιοθήκης Sklearn γίνεται με την μέθοδο One-Versus-Rest ή One-Versus-All. Η μέθοδος αυτή κατασκευάζει κ πλήθος δυαδικών ταξινομητών SVM ίσο με τα πλήθος των κλάσεων. Σε κάθε επανάληψη i γίνεται η ταξινόμηση της κλάσης i (στην οποία τα δεδομένα ταξινομούνται ως θετικά) ενάντια σε όλες τις υπόλοιπες κλάσεις όπως φαίνεται και στο παρακάτω σχήμα.



Μετρικές αξιολόγησης των μοντέλων

Πίνακας Σύγχυσης (Confusion Matrix)

Ο πίνακας σύγχυσης αποτελεί ένα χρήσιμο εργαλείο για την αξιολόγηση ενός αλγορίθμου ως προς την απόδοσή του στην ταξινόμηση κειμένων. Ο πίνακας αυτός είναι πολύ χρήσιμος καθώς μας πληροφορεί σχετικά με το αν κάποιες κλάσεις έχουν τη τάση να συγχέονται με άλλες κλάσεις κατά την ταξινόμηση. Κάθε γραμμή του αντιπροσωπεύει δείγματα της πραγματικής κλάσης, ενώ κάθε στήλη της προβλεπόμενης. Συγκεκριμένα πρόκειται για έναν MxM πίνακα, όπου το (i,j) στοιχείο του ισούται με το πλήθος των σημείων (στην περίπτωση πλήθος κειμένων) που, ενώ προέρχονται από την κλάση i, καταχωρούνται στην κλάση j. Στην παρούσα εργασία εφόσον οι κλάσεις ταξινόμησής είναι 169 ο πίνακας σύγχυσης θα αποτελεί έναν 169x169 πίνακα. Στην απλούστερη περίπτωση που M=2 η μορφή του πίνακα είναι η ακόλουθη :

		Predicted class	
	Data class	positive	negative
Actual class	positive	TP	FN
	negative	FP	TN

Όπως δηλαδή υποδεικνύει ο πίνακας σύγχυσης στην περίπτωση ενός δυαδικού ταξινομητή και ενός δείγματος, παράγονται τέσσερα διαφορετικά αποτελέσματα:

- TP (True Positive), ο αριθμός δειγμάτων που ταξινομήθηκαν επιτυχώς ως θετικά
- FN (False Negative), ο αριθμός δειγμάτων που ταξινομήθηκαν επιτυχώς ως αρνητικά
- FN (False Negative), ο αριθμός δειγμάτων που ταξινομήθηκαν εσφαλμένα ως αρνητικά
- FP (False Positive), ο αριθμός δειγμάτων που ταξινομήθηκαν εσφαλμένα ως θετικά

Με τον τρόπο αυτόν πληροφορούμαστε όχι μόνο για ποιες κλάσεις μπορεί να συγχέονται μεταξύ τους αλλά και για τους τύπους των λαθών. Βάσει του πίνακα σύγχυσης προκύπτουν οι παρακάτω μετρικές αξιολόγησης:

1. Μετρική Accuracy

Η μετρική accuracy είναι συνήθης για την αξιολόγηση των μοντέλων μηχανικής μάθησης και δηλώνει το ποσοστό του συνολικού αριθμού των σωστών προβλέψεων και ο αντίστοιχος τύπος της είναι:

$$\text{accuracy} = \frac{TN+TP}{TN+TP+FN+FP}$$

2. Μετρική Precision

Η μετρική precision δηλώνει το ποσοστό των θετικών παραδειγμάτων που ταξινομήθηκαν σωστά σε σχέση με το συνολικό πλήθος των δεδομένων που ταξινομήθηκαν ως θετικά από τον ταξινομητή και υπολογίζεται με βάση τον τύπο:

$$\text{precision} = \frac{TP}{TP+FP}$$

Από την εξίσωση συμπεραίνουμε, όσο αυξάνεται το precision, τόσο μειώνεται ο αριθμός των FP, δηλαδή των που δεν ταξινομούνται σωστά.

3. Μετρική Recall

Η μετρική recall δηλώνει το ποσοστό των θετικών παραδειγμάτων που ταξινομήθηκαν σωστά σε σχέση με το συνολικό πλήθος των δεδομένων που είναι πραγματικά θετικά και υπολογίζεται με βάση τον τύπο:

$$\text{recall} = \frac{TP}{TP+FN}$$

Για τη μετρική recall σε ένα πρόβλημα κατηγοριοποίησης ισχύει ότι όσο μεγαλύτερη είναι η τιμή της και συγκλίνει στο 1 τόσο περισσότερα δεδομένα έχουν σωστά αντιστοιχηθεί στην κλάση A , ωστόσο απουσιάζει η πληροφορία για το ποσοστό των στοιχείων που θεωρήθηκε λανθασμένα ότι ανήκουν επίσης στην κλάση A.

4. F-score

Οι μετρικές precision και recall δεν είναι αρκετές ώστε να περιγράψουν την αποτελεσματικότητα ενός ταξινομητή, καθώς η καλή απόδοση της μίας δε σημαίνει ότι θα συμβαίνει το ίδιο και με της άλλης. Ως λύση του προβλήματος αυτού, προτείνεται μία τέταρτη μετρική, η F-score, η οποία ορίζεται ως ο αρμονικός μέσος των μετρικών precision και recall, και ο τύπος που την περιγράφει είναι ο ακόλουθος :

$$\text{F-score} = \frac{2*precision*recall}{precision+recall}$$

Όταν η τιμή της μετρικής F-measure προσεγγίζει τη μονάδα, τότε έχει επιτευχθεί ένας καλός συνδυασμός των precision και recall.

Στην περίπτωση μιας multi-class ταξινόμησης όπως συμβαίνει στο πείραμα, για k πλήθος κλάσεων οι αναφερθείσες μετρικές προσαρμόζονται στις μορφές micro και macro. Στον υπολογισμό των macro μετρικών όλες οι κλάσεις αντιμετωπίζονται ισότιμα, αντίθετα οι micro μετρικές υπολογίζονται ξεχωριστά για κάθε κλάση μόνο από τα κείμενα που περιέχονται σ' αυτήν.Ως εκ τούτου συχνά οι micro μετρικές δίνουν πληροφορίες για την λοξότητα (skewness) και ισορροπία (balance) εντός κάθε κλάσης. Αναλυτικά δίνονται οι εξισώσεις :

1. **Μετρική Accuracy**

$$\text{Accuracy} = \frac{\sum_{i=1}^{k} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}}{k}$$

2. **Μετρική Precision**

- $\text{Precision\_macro} = \frac{\sum_{i=1}^{k} TP_i}{TP_i + FP_i}$
- $\text{Precision\_micro} = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k} (TP_i + FP_i)}$

3. **Μετρική Recall**

- $\text{Recall\_macro} = \frac{\sum_{i=1}^{k} TP_i}{TP_i + FN_i}$
- $\text{Recall\_micro} = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k} (TP_i + FN_i)}$

4. **F-score**

- $\text{F-score\_macro} = \frac{2 * precision\_macro * recall\_macro}{precision\_macro + recall\_macro}$
- $\text{F-score\_micro} = \frac{2 * precision\_micro * recall\_micro}{precision\_micro + recall\_micro}$

RMSE (Root Mean Squared Error), ρίζα μέσου τετραγωνικού σφάλματος (ΜΣΤ).

Αν $\hat{y}_i$ είναι η πρόβλεψη της ταξινόμησης του i-οστού στοιχείου και y_i η πραγματική τιμή του, τότε το ΜΤΣ δίνεται από την ακόλουθη σχέση :

$$\mathrm{mse}\,(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} (y_i - \hat{y}_i)^2.$$

Το ΜΣΤ ερμηνεύεται φυσικά ως την «κατά μέσο όρο» τετραγωνική απόκλιση του των δυο ποσοτήτων $\hat{y}_i$, y_i και μάλιστα εφόσον αυτές έχουν την ιδιότητα της αμεροληψίας, το ΜΤΣ συμπίπτει με τη διασπορά τους. Στην περίπτωση αριθμητικών υπολογισμών, προτιμάται η ρίζα του ΜΣΤ αντί της τιμής του ΜΣΤ και αυτό γιατί είναι άμεσα συγκρίσιμη και είναι στην ίδια τάξη μεγέθους με τις συγκρινόμενες ποσότητες.

Βελτιστοποίηση υπερ-παραμέτρων

Πολύ σημαντικό βήμα πριν την τελική ταξινόμηση των δεδομένων είναι η βελτιστοποίηση των υπερ-παραμέτρων κάθε αλγορίθμου. Ως υπερ-παράμετροι ορίζονται εσωτερικοί παράμετροι ενός αλγορίθμου που αφορούν τον σχεδιασμό του και μπορεί για παράδειγμα να σχετίζονται με αποφάσεις μοντελοποίησης, ή με τον αριθμό των μονάδων και των στρωμάτων σε ένα ΝΔ ή να είναι παράμετροι κανονικοποίησης (regularization) κατά την εκπαίδευση του αλγορίθμου.

Πιο αναλυτικά, ένα προβλεπτικό μοντέλο της μορφής $y = f(x;\lambda)$ αποτελείται από τα εξής χαρακτηριστικά :

- Ένα σύνολο υπερ-παραμέτρων $\lambda = [\lambda_1,\lambda_2,..,\lambda_v]$.
- Έναν αλγόριθμο μάθησης για να υλοποιηθεί η βελτιστοποίηση των υπερ-παραμέτρων χρησιμοποιώντας τα δεδομένα εκπαίδευσης.
- Ένα μοντέλο πρόβλεψης της μορφής $y = f(x)$, το οποίο παράγεται από τον αλγόριθμο μάθησης.
- Μία συνάρτηση κόστους L για να γίνει η αξιολόγηση του μοντέλου πρόβλεψης ως αποτέλεσμα της σύγκρισης των προβλεπόμενων δεδομένων με το σετ ελέγχου.

Σύμφωνα με όσα αναπτύχθηκαν η βελτιστοποίηση των υπερ-παραμέτρων μπορεί να αποτυπωθεί με την παρακάτω εξίσωση για έναν αλγόριθμο Α:

$$\lambda^* = \text{argmin } L(A_\lambda, D_{train}, D_{test})$$

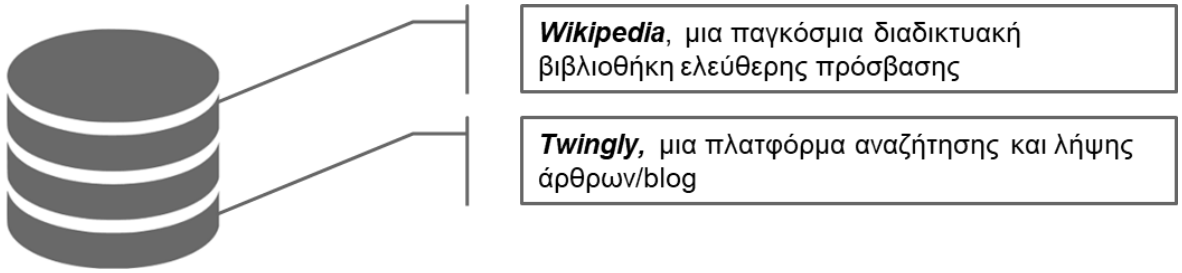
, όπου D_{train}, D_{test} τα σύνολα εκπαίδευσης και ελέγχου αντίστοιχα.

Επομένως κατά την εξίσωση, ο στόχος στην εκπαίδευση ενός μοντέλου είναι να βρεθεί το σύνολο των βέλτιστων υπερ-παραμέτρων λ^* το οποίο ελαχιστοποιεί τη συνάρτηση κόστους L στο αντίστοιχο σύνολο δεδομένων.

Συλλογή Δεδομένων

Πηγές δεδομένων

Οι πηγές στις οποίες βασίστηκε η συλλογή των άρθρων για την δημιουργια του συνόλου δεδομένων είναι οι ιστοσελίδες Wikipedia και Twingly.



Wikipedia

Η Βικιπαίδεια (Wikipedia) είναι μια παγκόσμια, πολύγλωσση, διαδικτυακή εγκυκλοπαίδεια ελεύθερης πρόσβασης, που δημιουργήθηκε από τον Jimmy Wales το 2001. Η σύστασή της βασίζεται στην αρχή του wiki, δηλαδή καθένας μπορεί διαδικτυακά να δημοσιεύσει άμεσα ή να τροποποιήσει περιεχόμενο, με την προϋπόθεση να συμφωνεί τους βασικούς κανόνες που καθορίζονται από την κοινότητα, π.χ. την επαληθευσιμότητα του περιεχομένου και την αποδοχή των λημμάτων. Επιπλέον η Βικιπαίδεια ενδείκνυται για την εργασία αυτήν χάρη στην εύκολη χρήση της, καθώς όλα τα άρθρα είναι κατηγοριοποιημένα σύμφωνα με μια αρχιτεκτονική κατά την οποία το περιεχόμενο κάθε άρθρου είναι άμεσα συνδεδεμένο με τον τίτλο του.

Twingly

Το Twingly API είναι μια πλατφόρμα αναζήτησης άρθρων/blog με , που ανήκει στην ομώνυμη σουηδική εταιρεία.

Βήματα για την δημιουργία συνόλου δεδομένων (Dataset)

1. Πρώτο βήμα για την δημιουργία συνόλου δεδομένων (dataset) αποτέλεσε η συλλογή άρθρων από την Wikipedia με χρήση της ομώνυμης βιβλιοθήκης της Python. Από τα άρθρα της Wikipedia επιλέχτηκαν μόνο οι περιλήψεις με σκοπό να αποφευχθεί η θεματική αλληλοεπικάλυψη ή απόκλιση των επιμέρους τμημάτων τους. Στο βήμα αυτό, συλλέχθηκαν 20 άρθρα για κάθε κατηγορία.

2. Μετά την συλλογή των πρώτων κειμένων εφαρμόστηκε ο αλγόριθμος TF-IDF αφού προηγουμένως είχαν αφαιρεθεί τα stop-words. Απ' τον αλγόριθμο αποσπάστηκαν τα TF κάθε λέξη; έτσι ώστε να εντοπιστούν οι λέξεις κλειδιά (keywords) της εκάστοτε κατηγορίας.

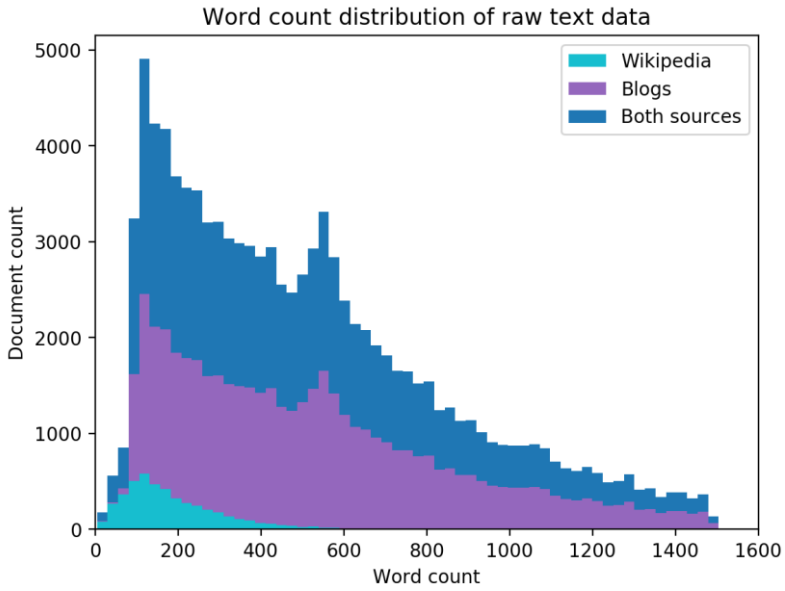
3. Χρήση των αποκτηθέντων keyword ως είσοδο στην μηχανή αναζήτησης Twingly και αποθήκευση των αποτελεσμάτων σε μορφή csv.

4. Εισαγωγή των αποτελεσμάτων του Twingly σε έναν βασικό αλγόριθμο Naive Bayes, συγκεκριμένα τον BernoulliNb, ούτως ώστε να ταξινομηθούν 30 κείμενα σε κάθε κατηγορία των οποίων η επίβλεψη θα επιβεβαιωθεί από ανθρώπινη επίβλεψη. Επανάληψη του βήματος 2 για εύρεση επιπλέον keyword.

Τα βήματα 3 και 4 επαναλαμβάνονται κυκλικά μέχρι να ολοκληρωθεί η επιθυμητή διάσταση του dataset των 300 κειμένων ανά κατηγορία, που εν τέλει συνολικά ισοδυναμούν με 51.087 άρθρα.

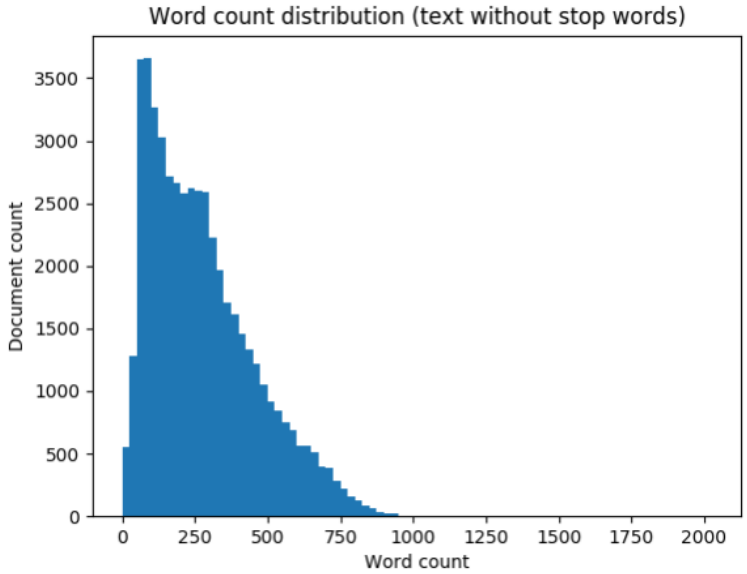
Στατιστικά στοιχεία του συνόλου δεδομένων

Όπως εμφανίζει και παρακάτω το ιστόγραμμα κατανομής του συνόλου των λέξεων των κειμένων του συνόλου δεδομένων, συλλέχτηκαν κείμενα με επιτρεπτό όριο από 90 έως 1500 λέξεις. Το 90% του συνόλου των δεδομένων αποτελείται από άρθρα του Twingly και αντίστοιχα 10% της Wikipedia. Η πλειοψηφία των κειμένων περιέχει λέξεις μεταξύ 100-200 και 500-700 τα οποία ποσά προκύπτουν κυρίως από τυχαιότητα στην παροχή άρθρων από την ιστοσελίδα Twingly.



Κατανομή συνόλου λέξεων στα κείμενα.

Παρακάτω εμφανίζεται η ίδια κατανομή των λέξεων όταν αφαιρεθούν τα stop words :



Κατανομή συνόλου λέξεων στα κείμενα μετά την αφαίρεση τετριμμένων λέξεων.

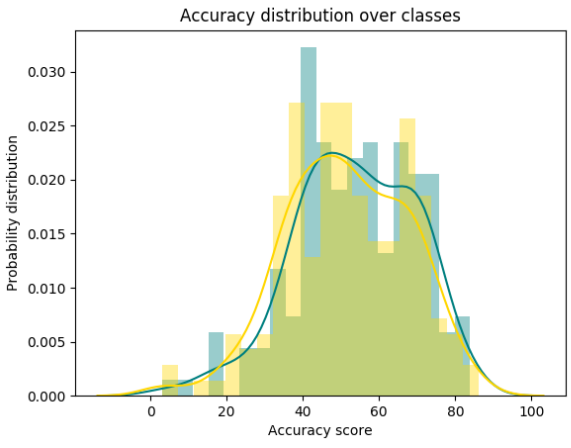
Παρουσίαση πειραματικών αποτελεσμάτων

Baseline Performance

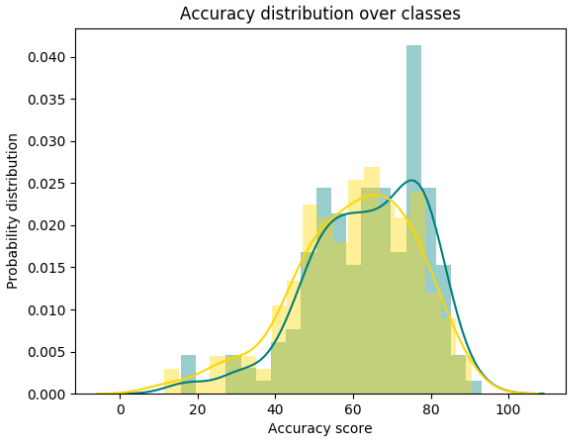
Ο BernoulliNB ο οποίος βασίζεται στο νόμο του Naive Bayes επιλέχτηκε χάρη στην απλότητα του για την υλοποίηση του baseline performance.

Χρησιμοποιώντας λοιπόν το πακέτο BernoulliNB() της βιβλιοθήκης της Python sklearn με τις προεπιλεγμένες τιμές των υπερ-παραμέτρων για τον ταξινομητή και το πακέτο Doc2Vec της βιβλιοθήκης genism, αρχικά προσπαθήσαμε να επιλέξουμε το μοντέλο Doc2Vec με την υψηλότερη ακρίβεια και λιγότερη υπερπροσαρμογή. Το σύνολο δεδομένων σε αυτό το βήμα χωρίστηκε σε 50% σύνολο εκπαίδευσης και 50% σύνολο ελέγχου. Οι επιλογές των μοντέλων Doc2Vec είναι οι ακόλουθες :

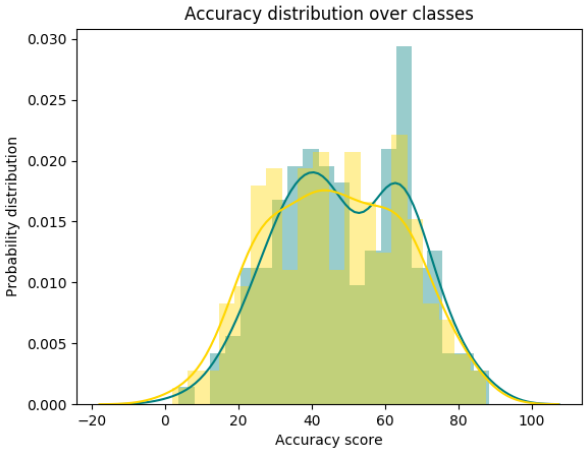
- **PV-DM** (Paragraph Vector – Distributed Memory)
- **PV – DBOW** (Paragraph Vector – Distribited Bag of Words) στο οποίο έγινε εκπαίδευση των διανυσμάτων λέξεων με το μοντέλο Word2Vec skip-gram κατά την εκπαίδευση των διανυσμάτων παραγράφων και διατηρήθηκαν έτσι καθ' όλη τη διάρκεια της εκπαίδευσης
- **PV – DBOW** (Paragraph Vector – Distribited Bag of Words) στο οποίο έγινε τυχαία αρχικοποίηση των διανυσμάτων λέξεων κατά την εκπαίδευση των διανυσμάτων παραγράφων και διατηρήθηκαν έτσι καθ' όλη τη διάρκεια της εκπαίδευσης



Α. Μοντέλο PV-DM



Β. Μοντέλο PV-DBOW με εκπαίδευση διανυσμάτων λέξεων



Γ. Μοντέλο PV-DBOW χωρίς εκπαίδευση διανυσμάτων λέξεων

	PV-DM	***PV-DBOW με εκπαίδευση διανυσμάτων λέξεων***	***PV-DBOW χωρίς εκπαίδευση διανυσμάτων λέξεων***
Train score	53.59	63.82	49.03
Test score	51.54	60.35	46.50

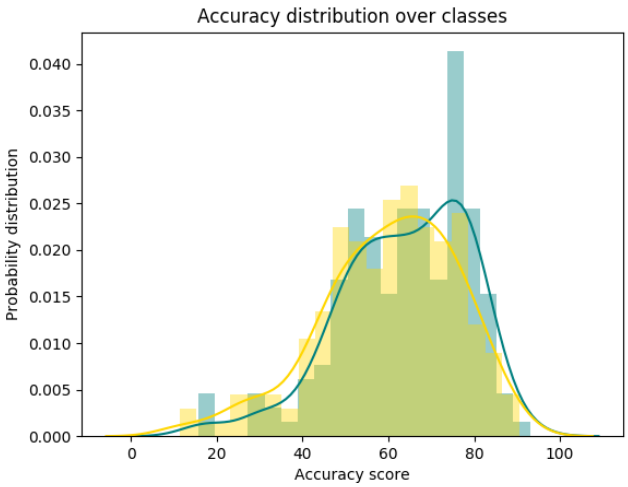
Για να επιλεχθεί το βέλτιστο μοντέλο, αρχικά δημιουργήθηκαν τα παραπάνω ιστογράμματα που δείχνουν την κατανομή της μετρικής accuracy όπως αυτή προκύπτει από τον Πίνακα Σύγχυσης για το σύνολο εκπαίδευσης με μπλε χρώμα και ελέγχου με κίτρινο.

Για την σύγκριση των ιστογραμμάτων παρατηρούμε την απόκλιση στα κέντρα τους, τη διαφορά στα σχήματά τους, το εύρος τους και την ύπαρξη ή μη απομακρυσμένων σημείων.

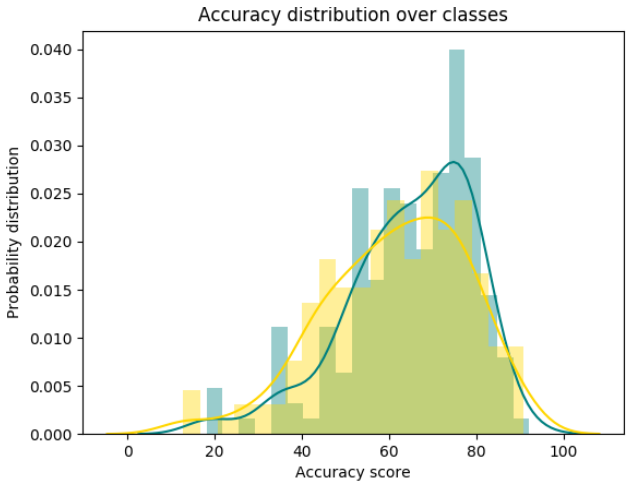
Όπως παρατηρούμε τα ιστογράμματα του μοντέλου PV-DBOW με εκπαίδευση διανυσμάτων λέξεων είναι μετατοπισμένα περισσότερο προς δεξιά δηλαδή προς μεγαλύτερες τιμές ακρίβειας και έχουν ταυτόχρονα μικρή απόκλιση μεταξύ τους. Το ίδιο επιβεβαιώνει και ο πίνακας με τις μέσες τιμές της μετρικής για κάθε μοντέλο. Ως εκ τούτου, επιλέγεται το μοντέλο PV-DBOW με εκπαίδευση διανυσμάτων λέξεων για την διανυσματική αναπαράσταση των κειμένων του συνόλου δεδομένων.

Εύρεση αποδοτικότερου χωρισμού συνόλου δεδομένων

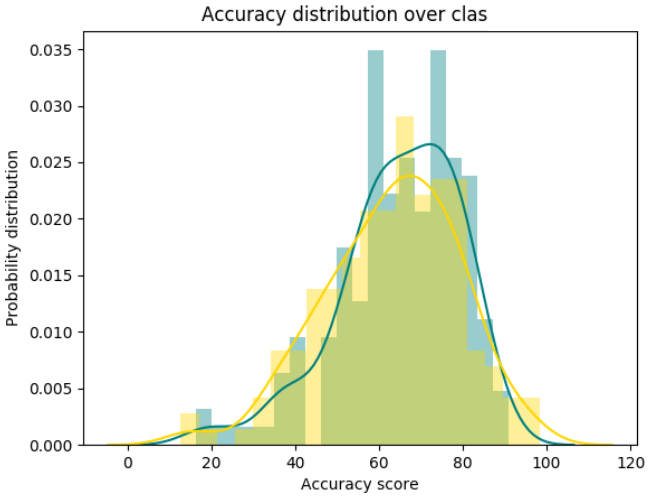
Εφόσον επιλέχτηκε το αποδοτικότερο μοντέλο Doc2Vec στο προηγούμενο βήμα, ψάχνουμε να δούμε αν κάποια άλλη αναλογία συνόλου εκπαίδευσης-συνόλου ελέγχου δίνει καλύτερα αποτελέσματα. Επιπλέον της αναλογίας 50%-50% συνόλου εκπαίδευσης-συνόλου ελέγχου που είδαμε προηγουμένως, θα δοκιμαστούν και οι αναλογίες 70%-30% και 80%-20% :



50% - 50%



70% - 30%



80% - 20%

	50%-50%	***70%-30%***	***80%-20%***
Train score	63.82	64.78	64.88
Test score	60.35	61.59	63.03

Από τα διαγράμματα και τον πίνακα με τις μέσες τιμές ακρίβειας παρατηρούμε ότι συγκεκριμένα η αναλογία 80%-20% έχει και την υψηλότερη απόδοση και το φαινόμενο υπερ-προσαρμογής είναι λιγότερο έντονο, επομένως είναι και αυτή που επιλέγεται.

Βελτιστοποίηση μοντέλου Doc2Vec

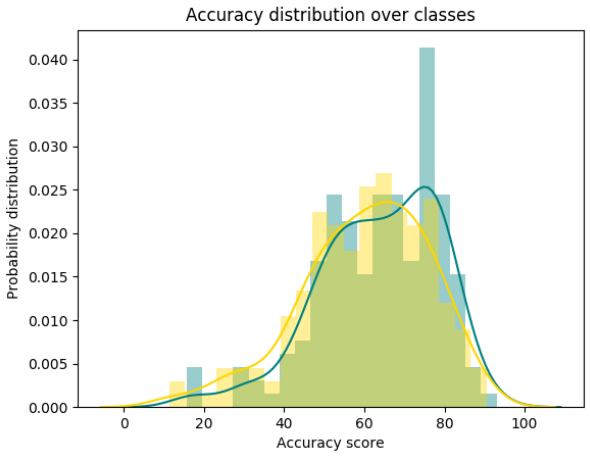
Εφόσον στο προηγούμενο βήμα επιλέχτηκε το μοντέλο PV-DBOW με εκπαίδευση διανυσμάτων λέξεων, επόμενο είναι να βελτιστοποιηθούν οι υπερ-παράμετροι του μοντέλου. Συγκεκριμένα οι υπερ-παράμετροι που επιλέχθηκαν προς βελτιστοποίηση είναι οι ακόλουθες :

- "**epochs**" : αριθμός εποχών εκπαίδευσης,
- "**window**" : μέγεθος παραθύρου,
- "**vector size**" : διαστάσεις διανυσμάτων παραγράφων και λέξεων,

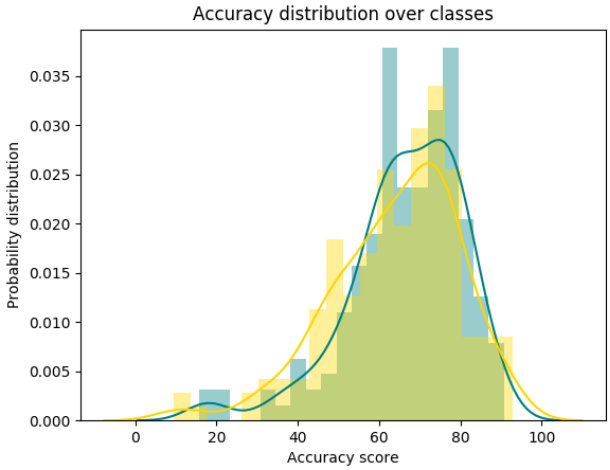
Οι προεπιλεγμένες τιμές των υπερ-παραμέτρων οι οποίες χρησιμοποιήθηκαν και στο προηγούμενο βήμα είναι :

Default Doc2Vec hyper-parameters	
window	**5**
vector size	**100**
Epochs	**5**

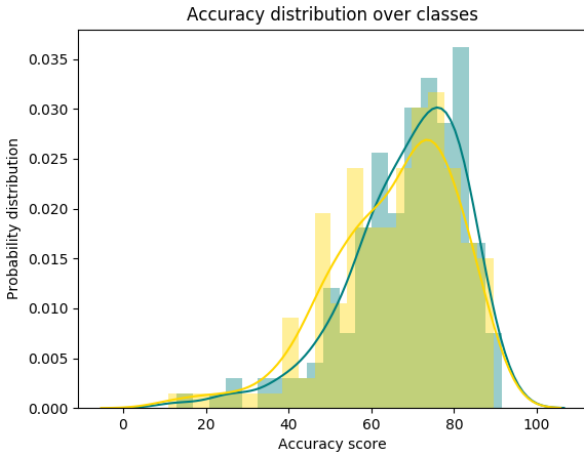
Αρχικά πειραματιστήκαμε με μεγαλύτερες τιμές της υπερ-παραμέτρου window.



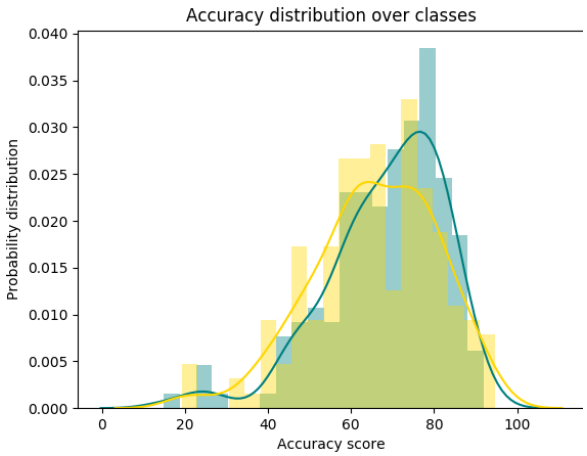
Window = 5



Window = 10



Window = 15

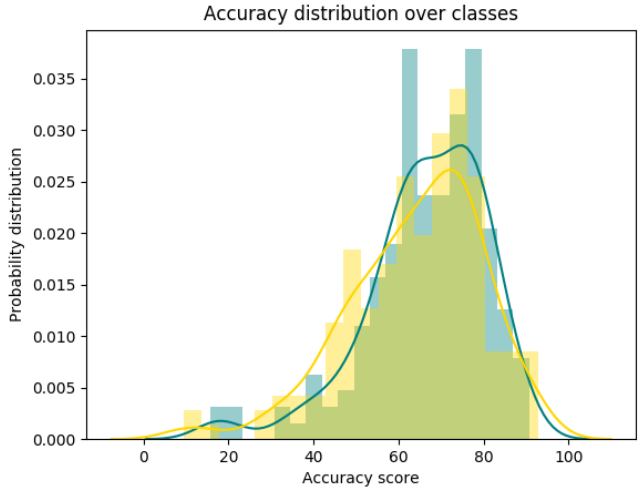


Window = 20

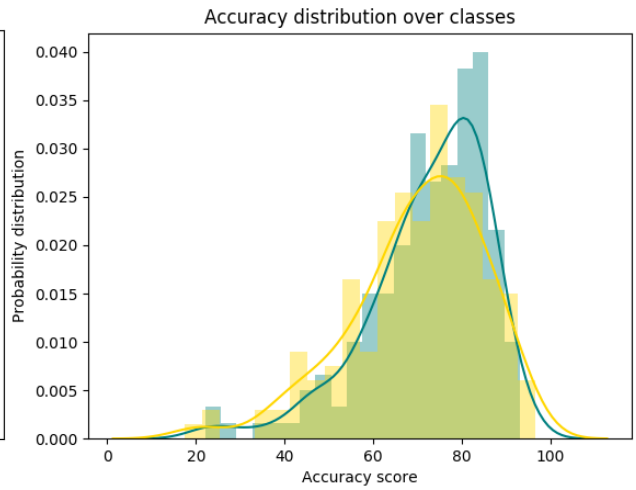
	Window = 5	**Window = 10**	**Window = 15**	**Window = 20**
Train score	63.82	66.65	68.59	68.68
Test score	60.35	64.11	65.80	65.57

Ομοίως συγκρίνοντας τα ιστογράμματα που προκύπτουν από το σύνολο εκπαίδευσης και ελέγχου και τις αντίστοιχες μέσες τιμές ακριβείας, παρατηρούμε ότι η υπερ-παράμετρος window με τιμή δέκα έχει υψηλότερες μέσες τιμές από την default τιμή και ταυτόχρονα γενικεύει και καλύτερα. Όσο η τιμή της υπερ-παραμέτρου αυξάνεται ταυτόχρονα αυξάνονται και οι μέσες τιμές ακρίβειας του συνόλου εκπαίδευσης και ελέγχου ωστόσο μεγαλώνει και η απόκλιση μεταξύ τους οδηγώντας σε μεγαλύτερο ποσοστό υπερπροσαρμογής. Για το λόγο αυτό τελικώς επιλέγεται η τιμή δέκα.

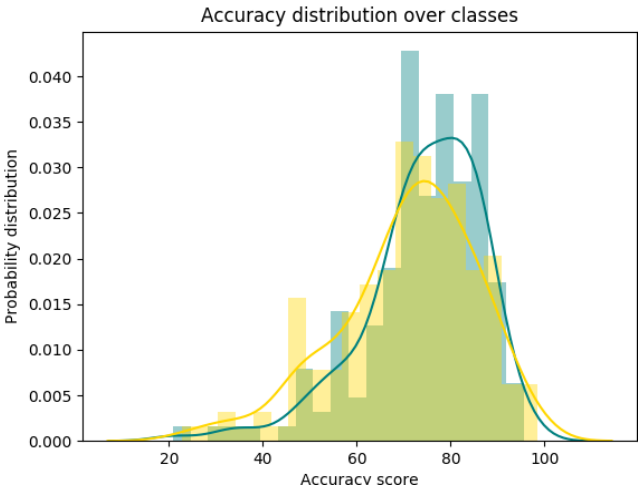
Επόμενη υπερ-παράμετρος για βελτιστοποίηση επιλέγεται η “vector size”:



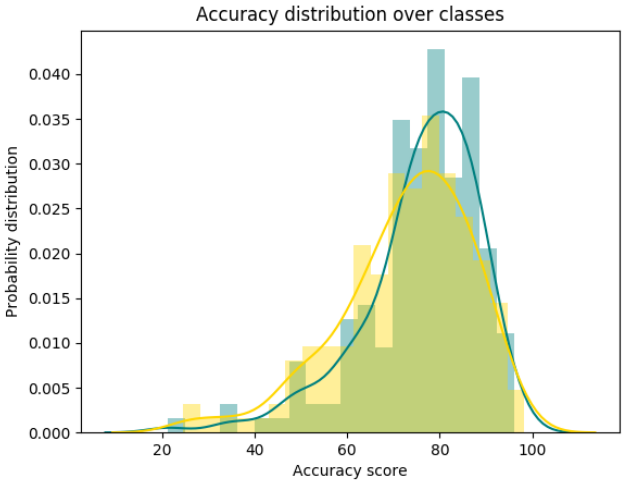
Vector size =100



Vector size =200



Vector size =300

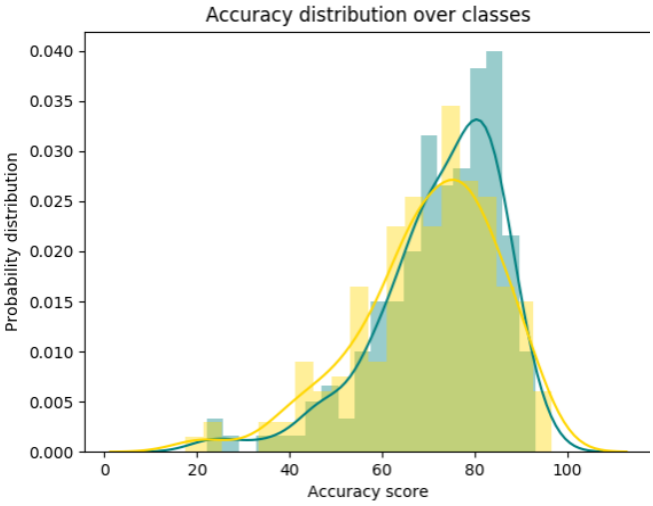


Vector size =500

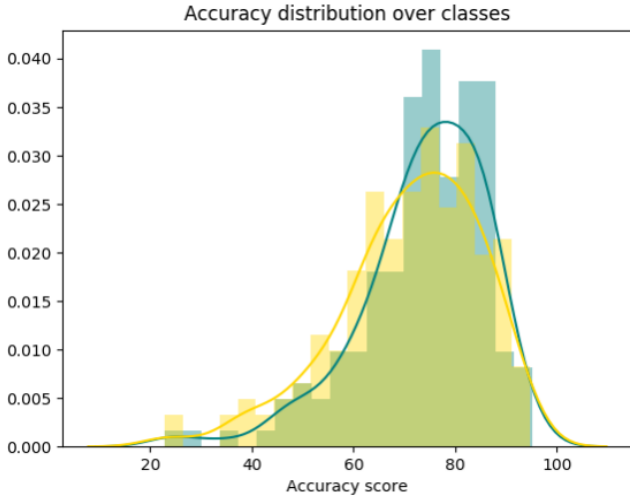
	Vector size= 100	**Vector size= 200**	**Vector size= 300**	**Vector size= 500**
Train score	66.65	72.53	74.27	76.02
Test score	64.11	70.01	70.87	72.72

Χρησιμοποιώντας τα ίδια κριτήρια όπως προηγουμένως επιλέγεται η τιμή διακόσια.

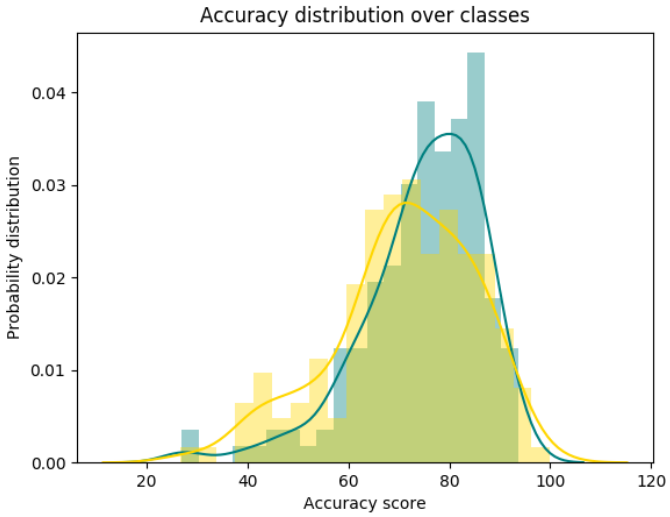
Τέλος, βελτιστοποιείται ο αριθμός των εποχών :



epochs = 5



epochs = 7



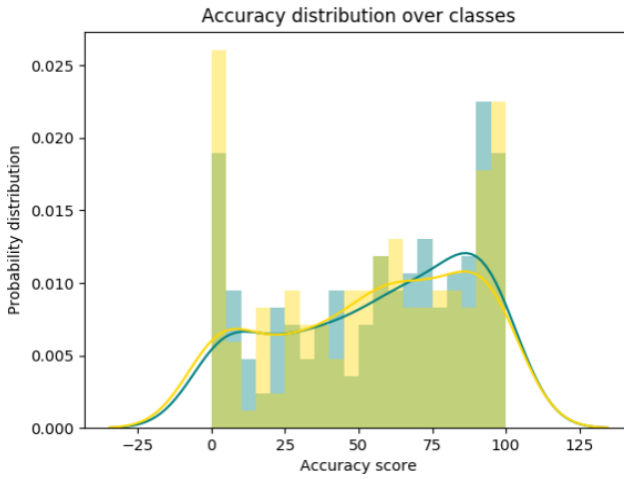
epochs = 10

	Epochs = 5	**Epochs = 7**	**Epochs = 10**
Train accuracy	72.53	73.98	74.80
Test accuracy	70.01	70.48	71.13

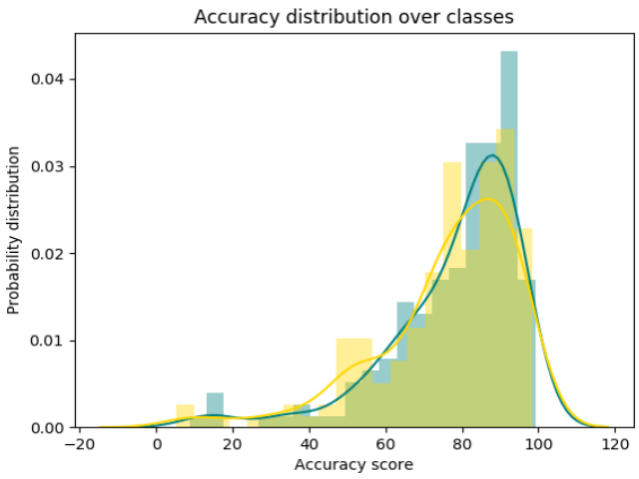
Παρατηρούμε ότι όσο αυξάνει ο αριθμός των εποχών το μοντέλο αρχίζει να πάσχει εντονότερα από υπερ-προσασμογή καθώς τα σημεία των δεδομένων εκπαίδευσης και ελέγχου απέχουν περισσότερο μεταξύ τους. Για το λόγο αυτό θα παραμείνουμε στο σύνολο των πέντε εποχών όπως αρχικά.

Βελτιστοποίηση αλγορίθμου LR

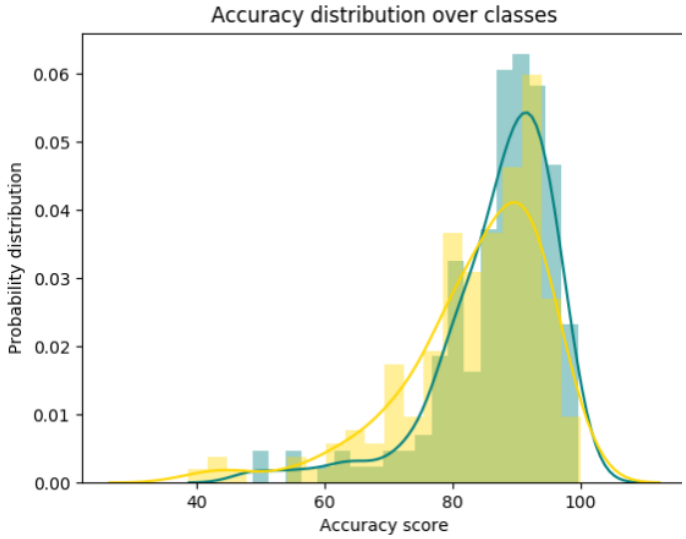
Ως υπερ-παράμετρο για την βελτιστοποίηση του αλγορίθμου LR επιλέχθηκε η παράμετρος κανονικοποίησης C :



C = 0.001



C = 0.01



C = 0.1

	C = 0.001	C = 0.01	C = 0.1
Train accuracy	**55.52**	**80.14**	**87.09**
Test Accuracy	**57.60**	**77.76**	**83.51**
RMSE	**5.48**	**6.13**	**6.36**

Παρατηρούμε ότι όσο αυξάνεται η τιμή στην παράμετρο C τόσο αυξάνεται το φαινόμενο της υπερ-προσαρμογής με συνέπεια να επιλέξουμε την τιμή 0.01.

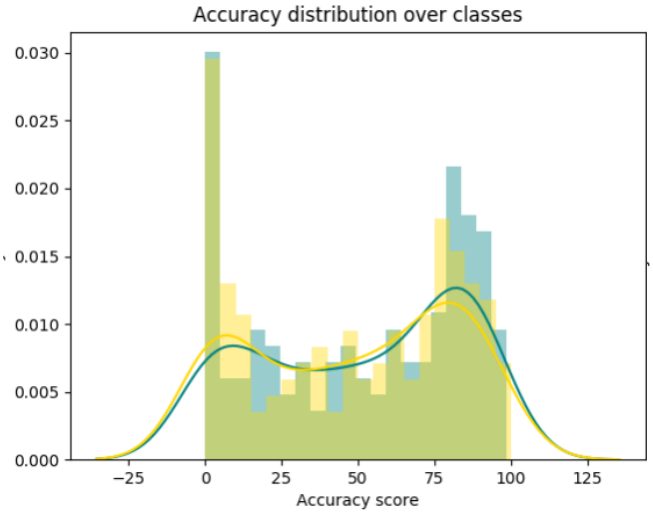
Βελτιστοποίηση αλγορίθμου SVM

Στο πακέτο SVC της βιβλιοθήκης sklearn ο προεπιλεγμένος τύπος πυρήνα είναι γκαουσσιανής ή ακτινικής βάσης, δηλαδή :

$$K(X,Z) = \exp(\gamma \|X - Z\|^2) \ \ ,\text{για} \ \ \gamma > 0$$

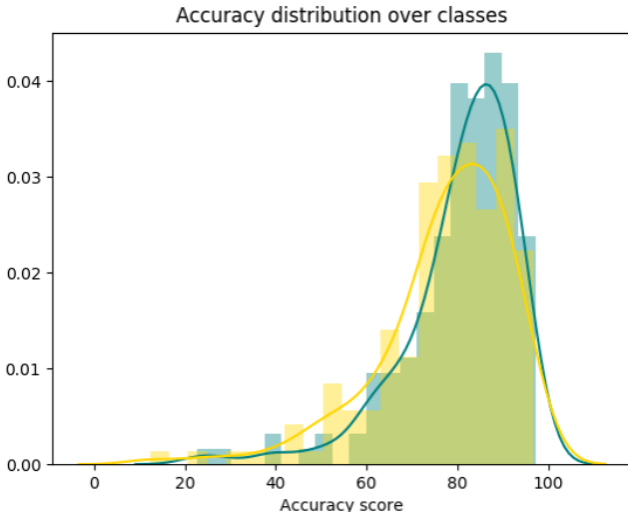
Η παράμετρος γ λειτουργεί σαν την αντίστροφο της τυπικής απόκλισης στην σύγκριση ομοιότητας δεδομένων εκπαίδευσης. Μεγάλες τιμές της γ δείχνουν μικρή διακύμανση στην ομοιότητα μεταξύ δεδομένων εκπαίδευσης ακόμα και αν αυτά βρίσκονται «μακριά» στον διανυσματικό χώρο και αντίστοιχα μικρές τιμές της γ δείχνουν μεγάλη διακύμανση. Η παράμετρος C λειτουργεί ως μια παράμετρος κανονικοποίησης, η οποία ρυθμίζει την ανοχή στο ποσοστό λάθος ταξινομήσεων για τον ορισμό του περιθωρίου του μοντέλου.

Για τις τιμές 0,1 και 1 στην υπερ-παράμετρο C δοκιμάστηκαν διάφοροι συνδυασμοί με τιμές της υπερ-παραμέτρου γ όπως δείχνουν παρακάτω τα διαγράμματα :



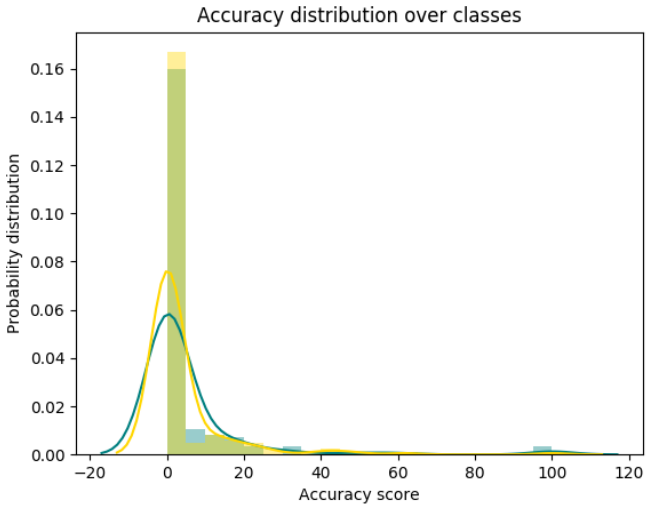
C=0.1

gamma=0.01



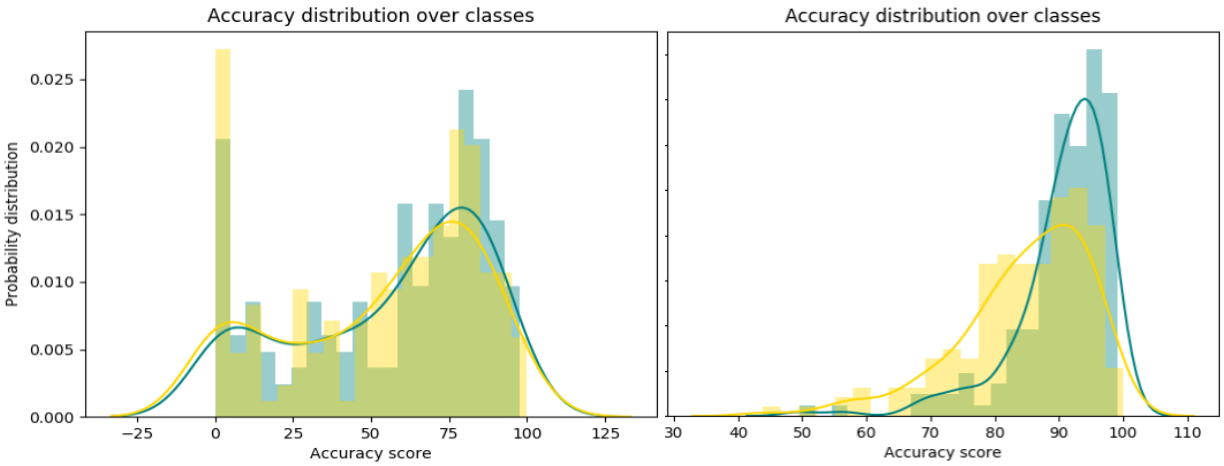
C=0.1

gamma=0.1



C=0.1

gamma=1

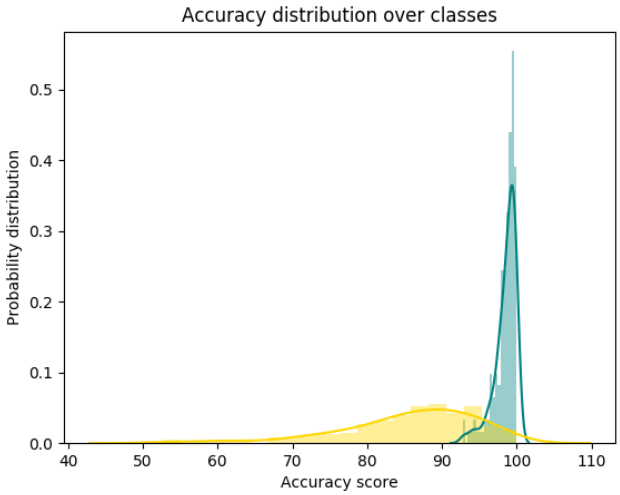


C=1

gamma=0.001

C=1

gamma=0.01



C=1

gamma=0.1

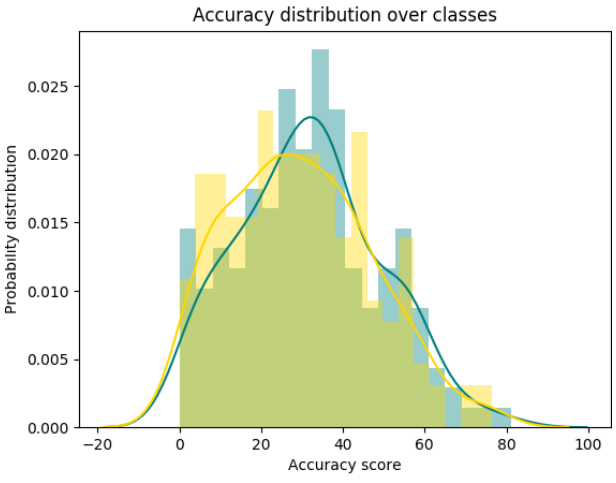
	C = 0.1 γ =0.01	**C = 0.1 γ =0.1**	**C = 0.1 γ =1**	**C = 1 γ =0.001**	**C = 1 γ =0.01**	**C = 1 γ =0.1**
Train accuracy	52.23	80.85	5.44	57.01	90.58	98.54
Test accuracy	49.69	77.37	4.24	54.63	84.46	85.75
RMSE	5.77	6.97	8.18	5.88	6.63	10.78

Παρατηρούμε ότι όσο μικραίνει η τιμή της υπερ-παραμέτρου C τόσο μικρότερη είναι και η απόκλιση στις κατανομές ακριβείας μεταξύ του συνόλου εκπαίδευσης και ελέγχου. Αυτό συμβαίνει διότι για μικρότερες τιμές του C το μοντέλο κατά την βελτιστοποίησή του θα αναζητήσει ένα μεγαλύτερο περιθώριο που διαχωρίζει το υπερεπίπεδο καθιστώντας το πιο ικανό να γενικεύει. Για το λόγο αυτό επιλέγεται η τιμή 0.1. Όσον αναφορά την υπερ-παράμετρο γ παρατηρούμε ότι για μεγαλύτερες τιμές το μοντέλο προσπαθεί να κάνει ακριβώς fit στα δεδομένα εκπαίδευσης με συνέπεια να οδηγείται σε υπερ-προσαρμογή. Για το λόγο αυτό για την δεδομένη τιμή 0.1 της υπερ-παραμέτρου C επιλέγεται η τιμή 0.1 και για την τιμή γ.

Βελτιστοποίηση αλγορίθμου MLP

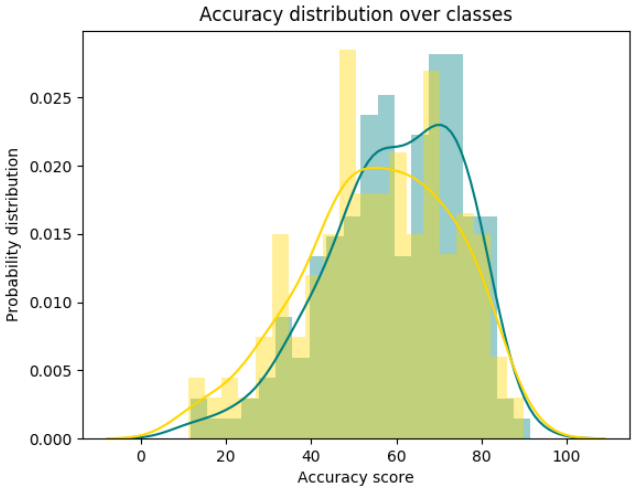
Στο πακέτο MLP της βιβλιοθήκης sklearn ο προεπιλεγμένος τύπος συνάρτησης ενεργοποίησης είναι η συνάρτηση “relu”. Για την βελτιστοποίηση του μοντέλου αρχικά θα αναζητήσουμε την αποδοτικότερη αρχιτεκτονική του NN με υπερ-παραμέτρους τον αριθμό των κρυφών στρωμάτων και τον αριθμό των κόμβων σε καθένα από αυτά.

Ως πρώτο βήμα ορίσαμε ένα NN με ένα κρυφό στρώμα και για διαφορετικό αριθμό κόμβων καταλήξαμε στα ακόλουθα αποτελέσματα με προεπιλεγμένο αριθμό εποχών 200 :



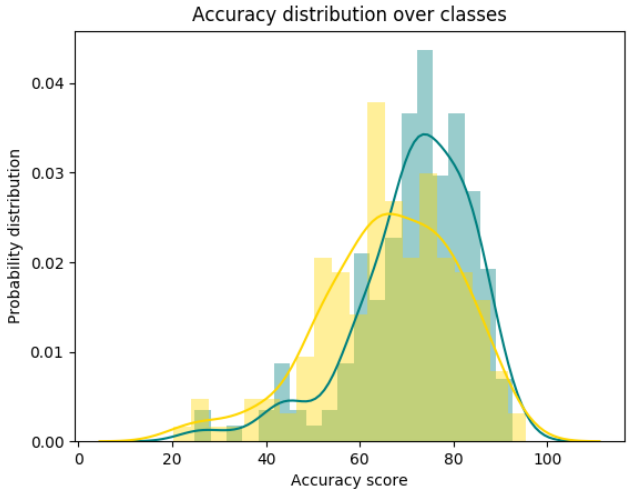
1 hidden layer

5 nodes



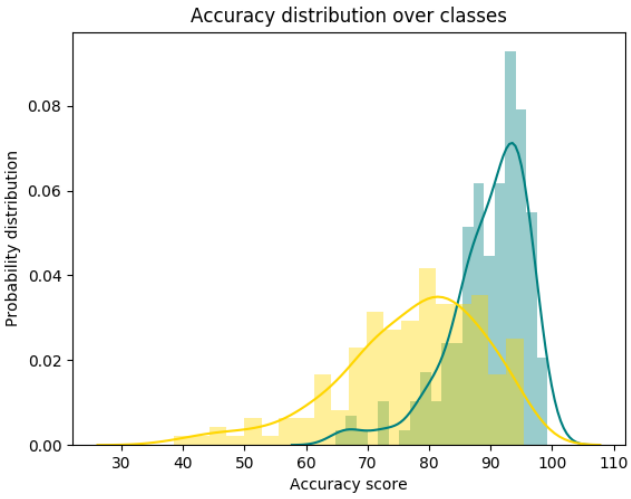
1 hidden layer

10 nodes



1 hidden layer

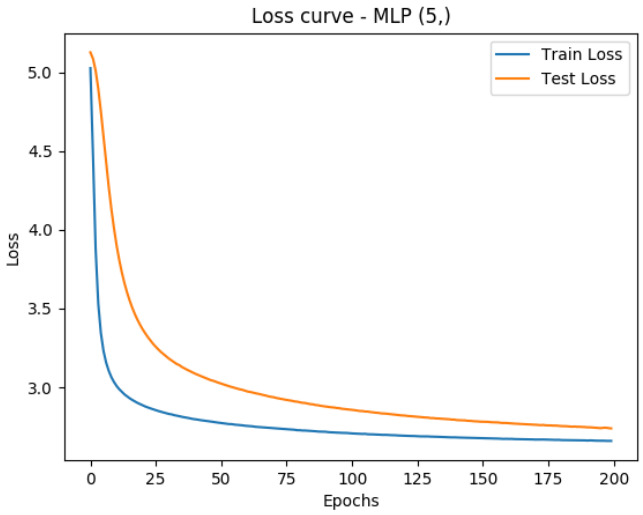
15 nodes



1 hidden layer

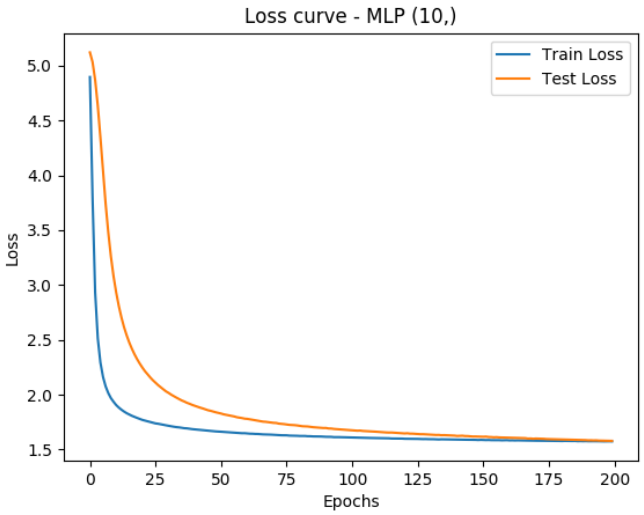
50 nodes

Και ακόλουθα οι γραφικές παραστάσεις συνάρτησης κόστους για κάθε αρχιτεκτονική :



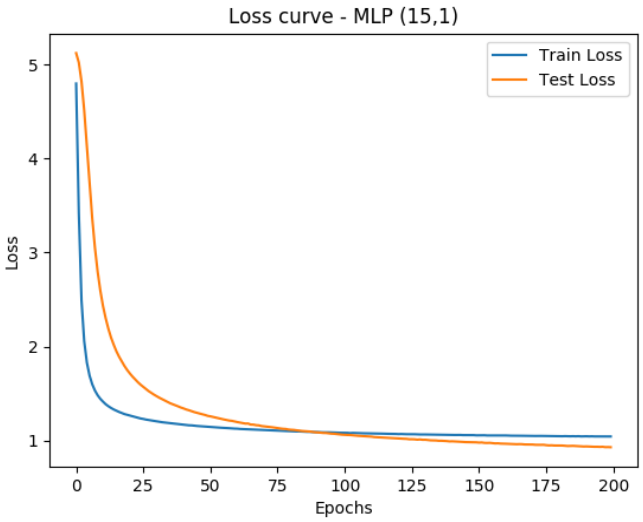
1 hidden layer

5 nodes



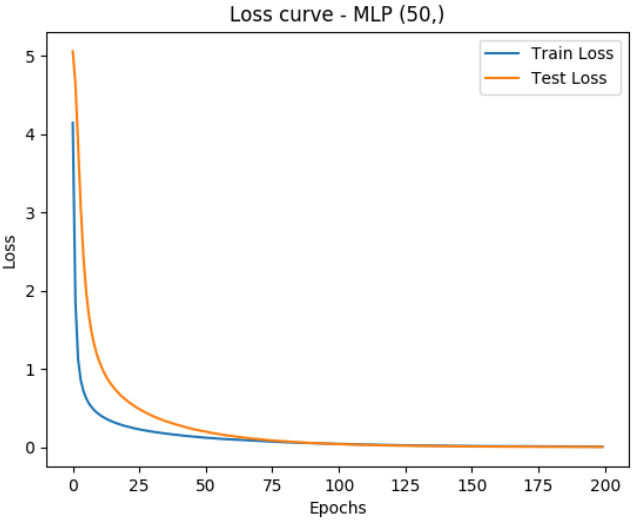
1 hidden layer

10 nodes



1 hidden layer

15 nodes



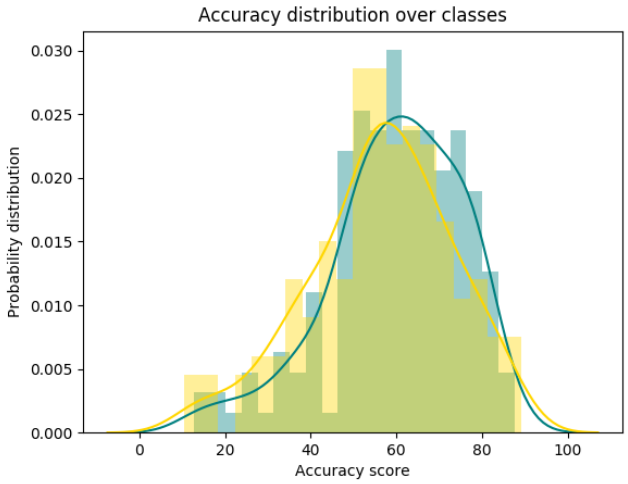
1 hidden layer

50 nodes

	1 hl 5 nodes	**1 hl 10 nodes**	**1 hl 15 nodes**	**1 hl 50 nodes**
Train accuracy	31.90	59.74	71.58	89.65
Test accuracy	29.68	55.69	66.35	77.23
RMSE	6.30	8.08	8.25	14.38

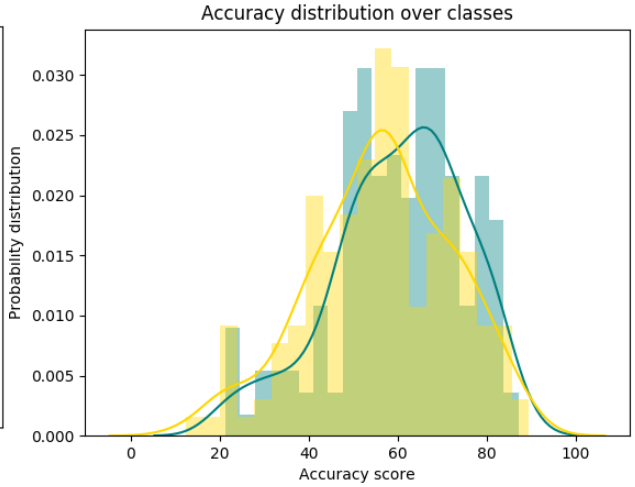
Παρατηρούμε ότι το NN είναι πιο «ευαίσθητο» στο φαινόμενο της υπερ-προσαρμογής συγκριτικά με τα μοντέλα LR και SVM καθώς εμφανίζει μεγαλύτερη απόκλιση μεταξύ του συνόλου εκπαίδευσης και συνόλου ελέγχου και ακόμα παρουσιάζει χαμηλότερη απόδοση . Μεταξύ των δοκιμών που έγιναν για διαφορετικούς αριθμούς κόμβων επιλέγεται ο αριθμός 15 καθώς είναι η μόνη αρχιτεκτονική του NN που εμφανίζει τη μεγαλύτερη δυνατή ακρίβεια κρατώντας παράλληλα τη γενίκευση του μοντέλου. Ακόμη από την γραφική παράσταση της συνάρτησης κόστους συναρτήσει των εποχών παρατηρούμε ότι η συνάρτηση κόστους για την δεδομένη αρχιτεκτονική για 200 εποχές έχει καταφέρει να σταθεροποιηθεί σε μια ελάχιστη τιμή και για το σύνολο εκπαίδευσης και το σύνολο ελέγχου.

Ως επόμενο βήμα είναι να δοκιμάσουμε την αρχιτεκτονική του NN με δυο κρυφά στρώματα :



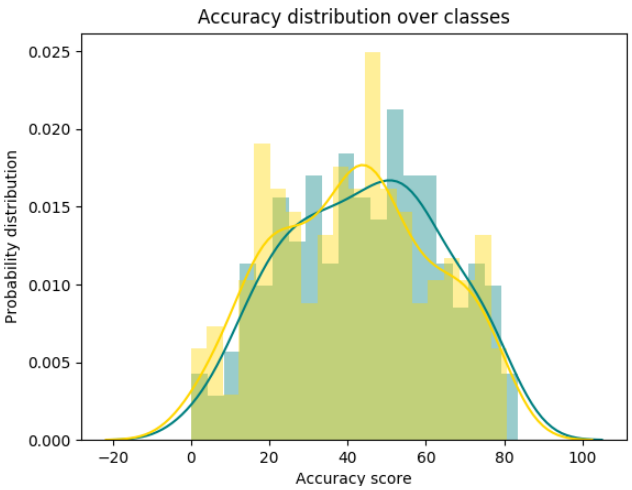
2 hidden layers

10 nodes, 10 nodes



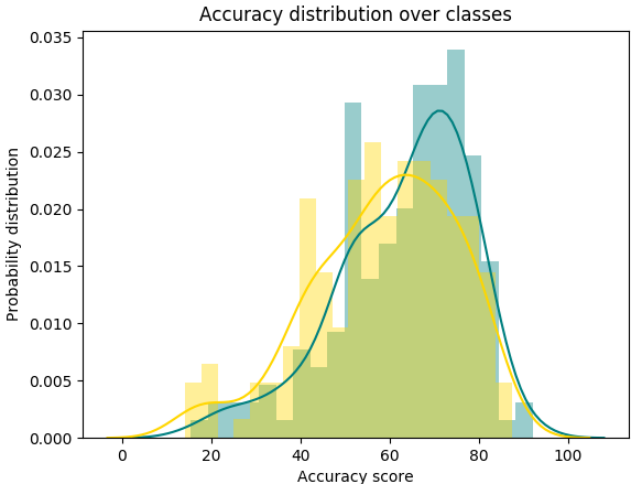
2 hidden layers

10 nodes, 15 nodes



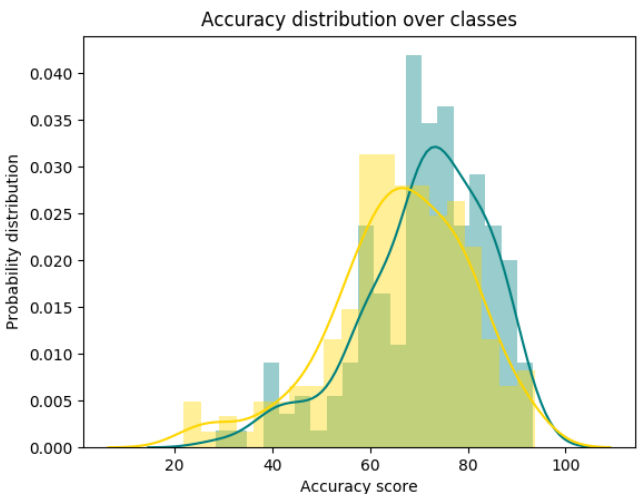
2 hidden layers

15 nodes, 5 nodes



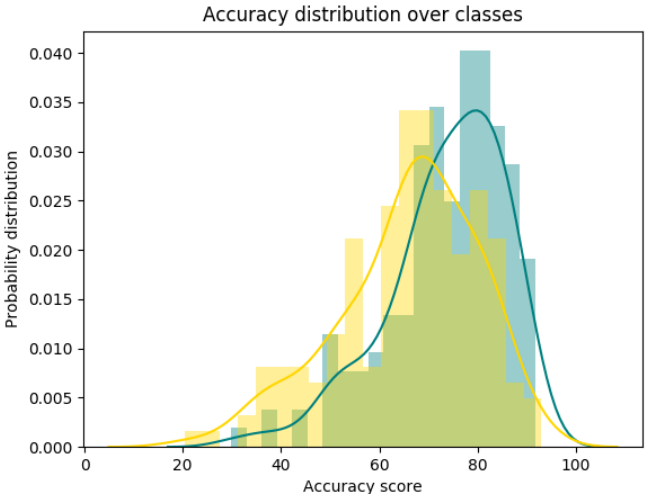
2 hidden layers

15 nodes, 10 nodes



2 hidden layers

15 nodes, 15 nodes

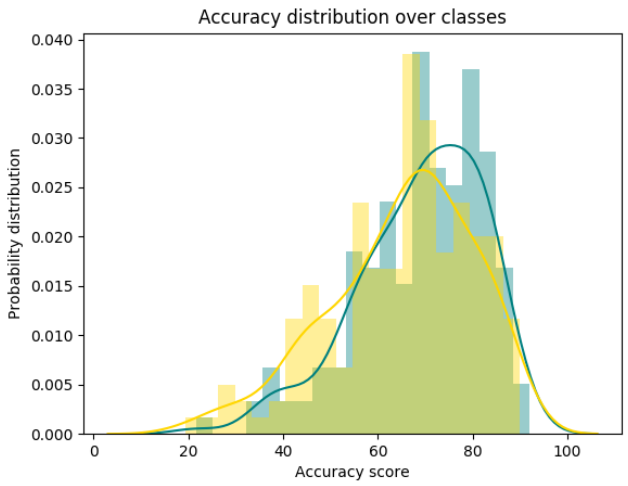


2 hidden layers

15 nodes, 50 nodes

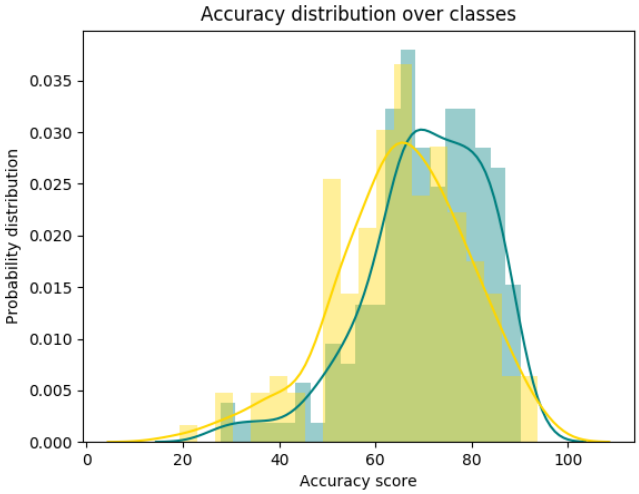
Όπως συμπεραίνουμε από τα παραπάνω διαγράμματα για μεγαλύτερες τιμές βάθους στην αρχιτεκτονική του NN το μοντέλο πάσχει πιο έντονα από υπερ-προσασμογή και η απόδοση του είναι ακόμα χαμηλότερη με αποτέλεσμα να καταλήξουμε και πάλι στην αρχική αρχιτεκτονική με ένα κρυφό στρώμα και 15 κόμβους.

Ως τελικό βήμα για την βελτιστοποίηση του NN είναι για την επιλεγμένη αρχιτεκτονική του ενός κρυφού στρώματος με 15 κόμβους να δοκιμαστούν διαφορετικοί αριθμοί εποχών :



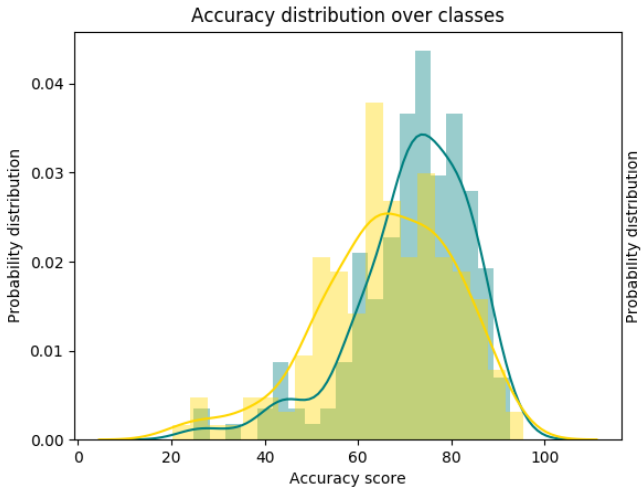
50 epochs

1 hidden layer, 15 nodes



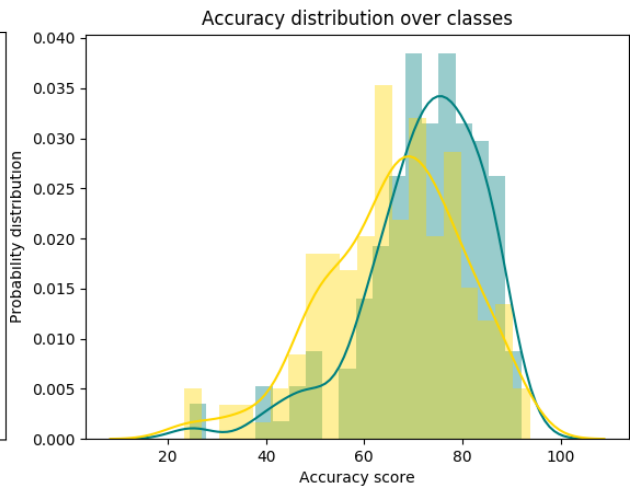
100 epochs

1 hidden layer, 15 nodes



200 epochs

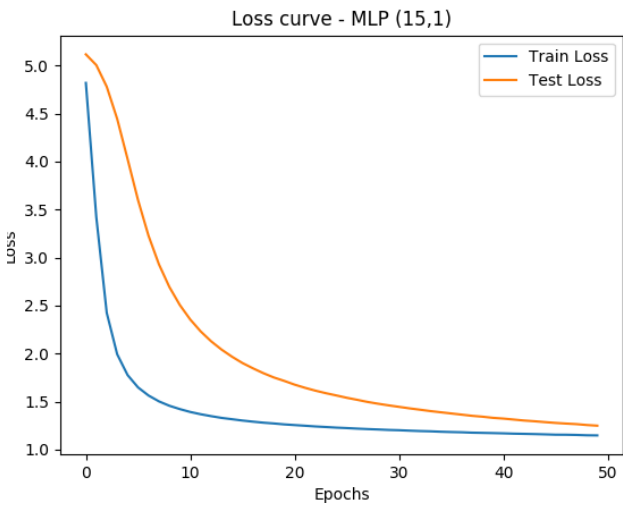
1 hidden layer, 15 nodes



500 epochs

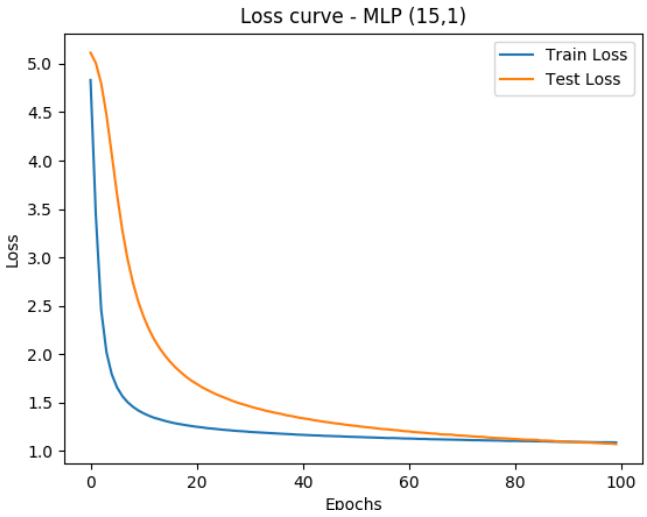
1 hidden layer, 15 nodes

	50 epochs 1 hl, 15 nodes	**100 epochs 1 hl, 15 nodes**	**200 epochs 1 hl, 15 nodes**	**500 epochs 1 hl, 15 nodes**
Train accuracy	69.08	70.54	71.58	72.25
Test accuracy	65.13	66.22	66.35	66.21
RMSE	8.28	8.63	8.25	8.82



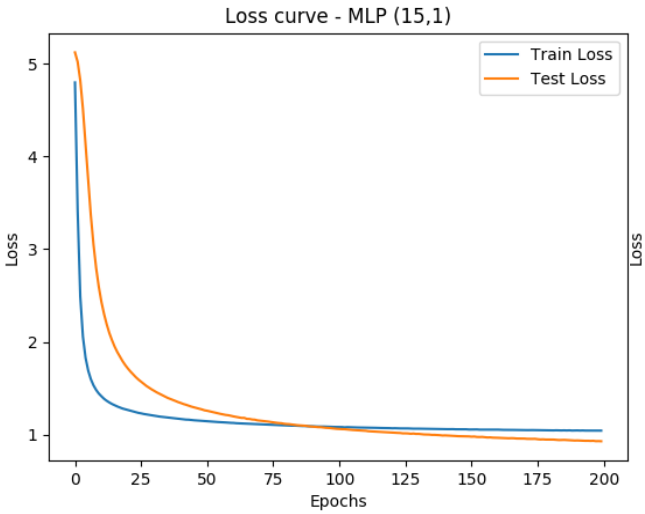
50 epochs

1 hidden layer, 15 nodes



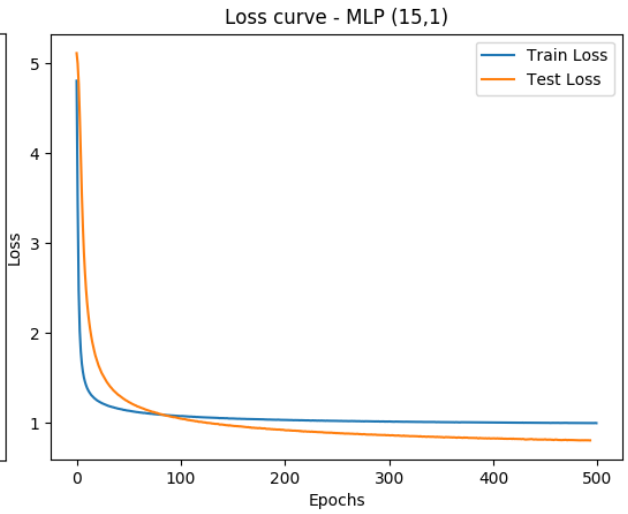
100 epochs

1 hidden layer, 15 nodes



200 epochs

1 hidden layer, 15 nodes



500 epochs

1 hidden layer, 15 nodes

Όσο αυξάνεται ο αριθμός των εποχών τόσο πιο έντονο γίνεται το φαινόμενο της υπερ-προσαρμογής όπως αναμενόταν. Αντίθετα για λιγότερες εποχές όπως το παράδειγμα των 50, το μοντέλο δείχνει να έχει την μικρότερη απόκλιση μεταξύ συνόλου εκπαίδευσης και ελέγχου αλλά ταυτόχρονα έχει και την χαμηλότερη απόδοση. Το μοντέλο με τις 100 εποχές αν και στην αναπαράσταση των ιστογραμμάτων δείχνει να έχει λιγότερες αποκλίσεις μεταξύ συνόλου εκπαίδευσης και ελέγχου από ότι το μοντέλο με τις 200 εποχές, παρατηρούμε ότι στην γραφική αναπαράσταση της συνάρτησης κόστους το σύνολο ελέγχου δεν σταθεροποιείται τόσο ομαλά όσο στην περίπτωση των 200 εποχών. Για το λόγο αυτό η τελική επιλογή του μοντέλου επιμένει να είναι ένα NN με ένα κρυφό στρώμα με 15 κόμβους και 200 εποχές.

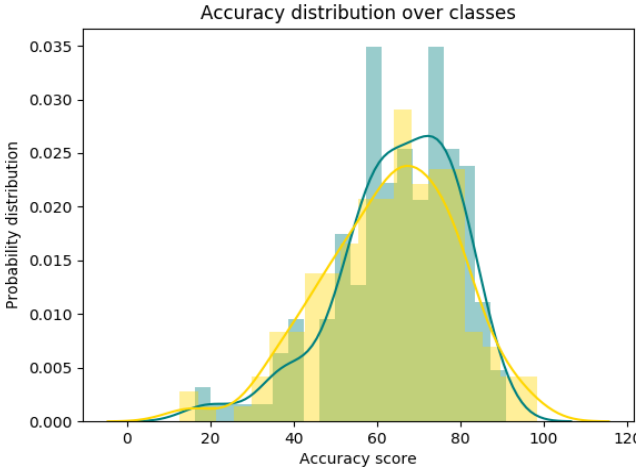
Συμπεράσματα

Αρχικά στο πρώτο στάδιο της δημιουργίας του συνόλου δεδομένων διαπιστώσαμε ότι είναι μια αρκετά «επίπονη» και χρονοβόρα διαδικασία, καθώς σε κάποιες κατηγορίες πιο σπάνιου περιεχομένου (πχ. *«Personal Finance>Insurance>Pet Insurance», «Shopping>Grocery and Household Shopping»*) ήταν πιο δύσκολη η εύρεση αρκετών λέξεων κλειδιών ή τα αντίστοιχα ευρεθέντα κείμενα ήταν πολύ μικρά, με αποτέλεσμα να γίνονται περισσότερες επαναλήψεις των βημάτων συλλογής δεδομένων.

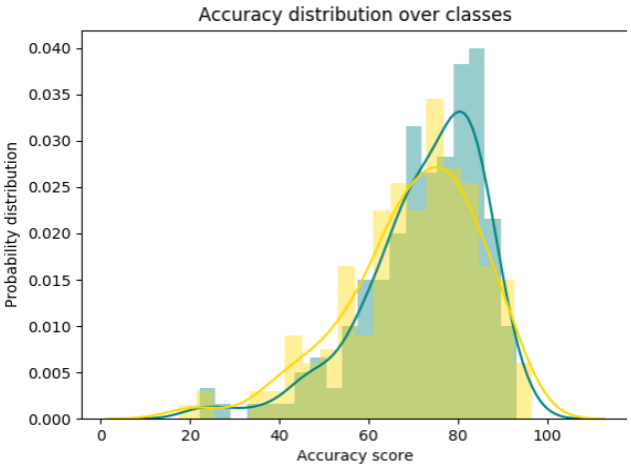
Στο δεύτερο στάδιο της εργασίας, για την επιλογή του καλύτερου μοντέλου Doc2Vec παρατηρήσαμε ότι το PV-DBOW με τυχαία αρχικοποίηση των διανυσμάτων λέξεων είχε την χειρότερη συνολικά απόδοση. Μια εξήγηση στο φαινόμενο αυτό σύμφωνα με την βιβλιογραφία μπορεί να βασίζεται στην αντικειμενική συνάρτηση του μοντέλου Doc2Vec η οποία αποσκοπεί στην μεγιστοποίηση του εσωτερικού γινομένου του κάθε διανύσματος παραγράφου με τα διανύσματα λέξεων των συστατικών του λέξεων. Επομένως τελική επιλογή ήταν το μοντέλο PV-DBOW στο οποίο εκπαιδεύονται τα διανύσματα λέξεων διότι είχε την υψηλότερη απόδοση και ταυτόχρονα έδειχνε καλή γενίκευση.

Έπειτα παρατηρήσαμε ότι για διαφορετικές αναλογίες συνόλου εκπαίδευσης-συνόλου ελέγχου επηρεάζεται εντονότερα η ικανότητα του μοντέλου να γενικεύει, με αποτέλεσμα να επιλεγεί το ποσοστό 80-20. Στην βελτιστοποίηση του μοντέλου Doc2Vec διαπιστώσαμε ότι για μεγαλύτερο αριθμό εποχών το φαινόμενο της υπερ-προσαρμογής εντεινόταν οπότε παραμείναμε στην αρχική τιμή των πέντε εποχών. Για τις υπόλοιπες δύο υπερπαραμέτρους ωστόσο επιλέχθηκαν νέες τιμές.

Ακολούθως φαίνεται το επιλεγμένο μοντέλο Doc2Vec πριν και μετά την βελτιστοποίησή του :



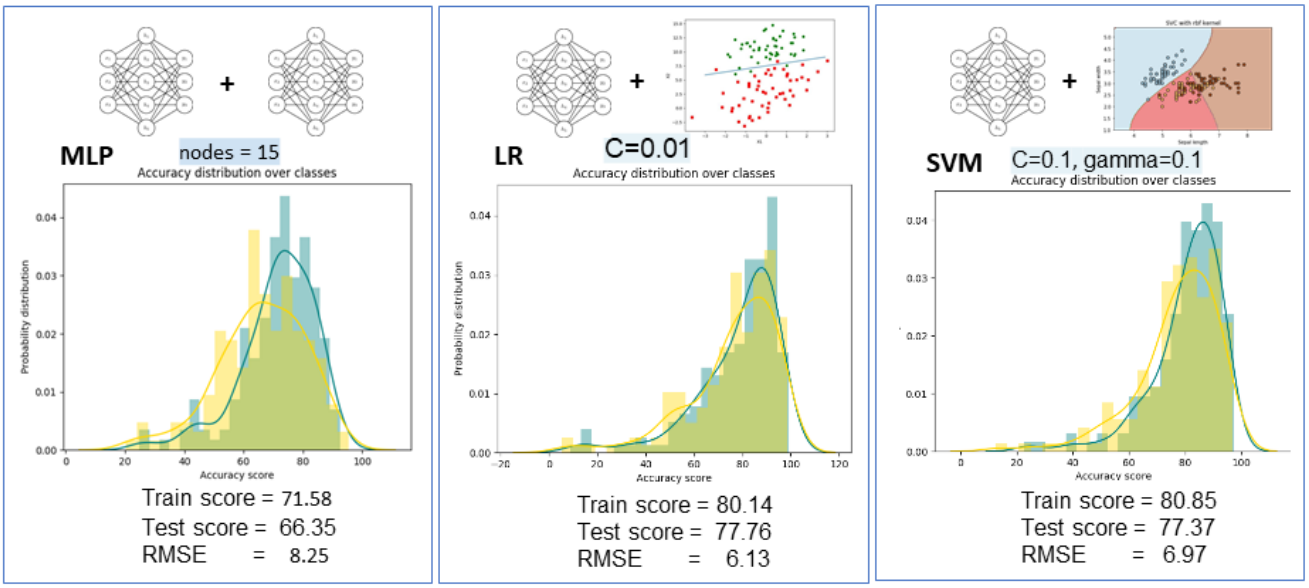
Πριν την βελτιστοποίηση
Train score = 64.88,
Test score = 63.03



Μετά την βελτιστοποίηση
Train score = 72.53
Test score = 70.01

Για την ταξινόμηση κειμένου διαπιστώσαμε ότι ο συνδυασμός NN για την διανυσματική αναπαράσταση κειμένων με γραμμικό ταξινομητή ,στην περίπτωση μας η Λογιστική Παλινδρόμηση,

ήταν ο αποδοτικότερος. Δεύτερος σε σειρά έρχεται ο συνδυασμός του Doc2Vec με τον ταξινομητή SVM χρησιμοποιώντας μη γραμμική διαχωριστική επιφάνεια. Η απόκλιση των μέσων τιμών της ακρίβειας για τα σύνολα εκπαίδευσης και ελέγχου του μοντέλου SVM με το μοντέλο LR είναι συγκριτικά μικρά ωστόσο παρατηρώντας τα ιστογράμματα το μοντέλο LR δείχνει καλύτερη γενίκευση. Τέλος, ο συνδυασμός Doc2Vec με άλλο NN, δηλαδή το μοντέλο MLP, ήταν ο χαμηλότερος σε απόδοση και ταυτόχρονα ο πιο ευαίσθητος στο φαινόμενο της υπερπροσαρμογής.



Μελλοντικές επεκτάσεις

Εφόσον μεγάλο κομμάτι της διπλωματικής αποτέλεσε η δημιουργία του συνόλου δεδομένων, το τελικό αποτέλεσμα της εργασίας θα μπορούσε να βελτιωθεί και να επεκταθεί περαιτέρω ως προς δυο κατευθύνσεις.

- **Διατηρώντας το παρόν σύνολο δεδομένων ως έχει.**

  Αυτού του είδους η κατεύθυνση έχει μια πιο "real-world" προσέγγιση καθώς δεν πρέπει να ξεχνάμε ότι το 90% του συνόλου των δεδομένων προέρχεται από την ιστοσελίδα Twingly. Εν αντιθέσει με τα άρθρα της Wikipedia που το περιεχόμενο των άρθρων είναι κυρίως ορισμοί και ο λόγος είναι επίσημος, αρκετά από τα άρθρα του Twingly προέρχονται από blogs χρηστών του διαδικτύου στα οποία γίνεται χρήση πιο καθημερινής γλώσσας και εμπεριέχονται λογοπαίγνια και αργκό.

  Υπό αυτή την κατεύθυνση λοιπόν, η εργασία θα μπορούσε να επεκταθεί αρχικά με αναλυτικότερη βελτιστοποίηση του αλγορίθμου Doc2Vec. Για παράδειγμα θα μπορούσε να βελτιστοποιηθεί η υπερ-παράμετρος "min_count" σύμφωνα με την οποία το μοντέλο αγνοεί όσες λέξεις εμφανίζονται λιγότερες φορές από την τιμή της. Επιπλέον, θα μπορούσαν να εκπαιδευτούν πέρα από μονές λέξεις διανύσματα και ζευγάρια λέξεων γνωστά και ως bigrams. Τέλος ιδιαίτερη σημασία για την επέκταση της παρούσας εργασίας αποτελεί ταυτόχρονα και η αναλυτικότερη βελτιστοποίηση των επιλεγμένων ταξινομητών, επιλέγοντας περισσότερες υπερ-παραμέτρους όπως τον «solver» οι οποίες μπορούν να αποφέρουν καλύτερα αποτελέσματα.

- **Βελτιώνοντας το παρών σύνολο δεδομένων.**

  Για την βελτίωση του παρόντος συνόλου δεδομένων, αρχικά θα ανατρέξουμε στον πίνακα σύγχυσης του μοντέλου με την υψηλότερη απόδοση και καλύτερη γενίκευση έτσι ώστε να αναζητήσουμε σε αυτόν τα μη διαγώνια στοιχεία με το υψηλότερο σκορ. Αυτά τα στοιχεία του πίνακα δηλώνουν λανθασμένες προβλέψεις που έγιναν μεταξύ των κλάσεων που αφορά το στοιχείο. Για τις κλάσεις αυτές θα αναζητηθούν τα «προβληματικά» κείμενα που προκαλούν αυτή τη σύγχυση και στις περιπτώσεις που κρίνεται ότι ένα κείμενο θα ταίριαζε περισσότερο σε μια άλλη κλάση να γίνει ανακατάταξή του. Αν μετά το βήμα αυτό δεν βελτιωθεί σημαντικά η απόδοση, θα μπορούσαν να συλλεγούν επιπλέον κείμενα για τις προβληματικές περιπτώσεις. Αυτή η λύση έχει μια πιο εκπαιδευτική προσέγγιση καθώς θα προσπαθήσουμε να συλλέξουμε κείμενα με πιο σαφή περιεχόμενο και πιο απλή γλώσσα, με αποτέλεσμα να γίνουν πιο ακριβείς προβλέψεις από τους ταξινομητές.

  Τέλος σημαντικό είναι να δοθεί περισσότερη προσοχή στο κομμάτι της προεπεξεργασίας κειμένου. Από την βιβλιογραφία συνιστάται ότι το κομμάτι αυτό μπορεί να καλύψει χρονικά έως και το ένα τέταρτο μιας εργασίας ταξινόμησης κειμένων καθιστώντας το πολύ σημαντικό στοιχείο για την καλή απόδοση του μοντέλου. Επομένως για να επιτευχθεί μια ποιοτικότερη ταξινόμηση θα μπορούσαν για παράδειγμα να εξεταστούν αναλυτικότερα τα stop-words και να αφαιρεθούν και άλλα όπως και να αφαιρεθούν συχνές φράσεις που δεν προσθέτουν νόημα στο κείμενο όπως «follow my page», «source link» κ.α.

Βιβλιογραφία

[1] Ingrid Grønlie Guren, August 2, 2015 Content Categorization for Contextual Advertising Using Wikipedia

[2] V. Srividhya, R. Anitha, 2010 Evaluating Preprocessing Techniques in Text Categor ization

[3] Ammar Ismael Kadhim, Yu-N Cheah, and Nurul Hashimah Ahamed,2014 Text Document Preprocessing and Dimension Reduction Techniques for Text Document Clusterin

[4] Matthew J. Dennyy Arthur Spirling,2016 Text Preprocessing For Unsupervised Learning: Why It Matters, When It Misleads, And What To Do About It

[5] Marina Sokolova, Guy Lapalme, 2009 A systematic analysis of performance measures for classification tasks

[6] Ying Liu, Han Tong Loh, Kamal Youcef-Toumi, and Shu Beng Tor, Handling of Imbalanced Data in Text

Classification: Category-Based Term Weights

[7] Jey Han Lau and Timothy Baldwin, 2014 An Empirical Evaluation of doc2vec with

Practical Insights into Document Embedding Generation

[8] David Meyer, 2016 How exactly does word2vec work?

[9] Tomas Mikolov, Distributed Representations of Sentences and Documents

[10] Xin Rong, word2vec Parameter Learning Explained

[11] Hiroshi Shimodaira,2015 Text Classification using Naive Bayes

[12] Andrew McCallum, A Comparison of Event Models for Naive Bayes Text Classi_cation

[13] Kwangmoo Koh Seung-Jean Kim Stephen Boy, 2007 An Interior-Point Method for Large-Scale l1-Regularized Logistic Regression

[14] Tran Dong, Modeling Conditional Probabilities

[15] Gilles Gasso,2019 Logistic regression

[16] Sebastian Ruder, An overview of gradient descent optimization
algorithms

[17] Diederik P. Kingma Jimmy Lei Ba,2015 ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION

[18] Kein Lin Du M. N. S. Swammy,2014 Neural Networks and
Statistical Learning

[19] Roger Grosse, Multilayer Perceptrons

[20] Leo Weissbart Stjepan Picek and Lejla Batina, On the Performance of Multilayer Perceptron in Pro_ling Side-channel Analysis

[21] Haim Sompolinsky,2013 The Perceptron

[22] Johan A. K. Suykens Marco Signoretto Andreas Argyriou,2015 Regularization Optimization Kernels and Support Vector Machines

[23] Ingo Steinwart Andreas Christmann, Support Vector Machines

[24] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin,2007 A Practical Guide to Support Vector Classification

[25] Chih-Wei Hsu and Chih-Jen Lin, A Comparison of Methods for Multi-class Support Vector Machines

[26] S. Sathiya Keerthi Chih-Jen Lin, Asymptotic Behaviors of Support Vector Machines with Gaussian Kernel

[27] *Andreas C. Müller Sarah Guido,2016 Introduction to Machine Learning with Python*

[28] https://el.wikipedia.org/wiki/Επεξεργασία_φυσικής_γλώσσας

[29]https://towardsdatascience.com/a-gentle-introduction-to-calculating-the-tf-idf-values-9e391f8a13e5

[30] https://radimrehurek.com/gensim/models/deprecated/doc2vec.html

[31] https://radimrehurek.com/gensim/models/doc2vec.html#module-gensim.models.doc2vec

[32] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[33] https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

[34] https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html

[35] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[36] https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/

[37] https://machinelearningmastery.com/learzning-curves-for-diagnosing-machine-learning-model-performance/

[38] https://towardsdatascience.com/useful-plots-to-diagnose-your-neural-network-521907fa2f45

[39] https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics

[40] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html#sklearn.metrics.confusion_matrix

Παράρτημα

Αποθήκευση δεδομένων

Για την αποθήκευση του συνόλου των δεδομένων όπως και σε επόμενα βήματα χρησιμοποιείται η βιβλιοθήκη της Python, Pandas. Η Pandas είναι μια πολύ εύχρηστη βιβλιοθήκη λογισμικού ανοιχτού κώδικα για τον χειρισμό και την ανάλυση δεδομένων.

- **Πλαίσιο Δεδομένων (Dataframe)**

Το Dataframe είναι μια δισδιάστατη δομή δεδομένων της βιβλιοθήκης Pandas στην οποία τα δεδομένα ευθυγραμμίζονται σε μορφή πίνακα σε γραμμές και στήλες. Στις γραμμές αποθηκεύονται τα δεδομένα και οι τίτλοι των στηλών που επιλέγονται από τον χρήστη υποδηλώνουν την κατηγορία των αντίστοιχων δεδομένων. Στην παρούσα εργασία ως πρώτη στήλη του Dataframe με τίτλο "DBID" επιλέγεται το αναγνωριστικό (id) του κείμενο, το οποίο είναι μοναδικό για κάθε κείμενο. Έπειτα προστίθενται οι στήλες "Text" ,"Title", "Category" που όπως φανερώνουν και οι τίτλοι εμπεριέχουν το κείμενο, τον τίτλο του κειμένου και την κατηγορία στην οποία ταξινομείται το κείμενο.

- **Csv file**

Για την αποθήκευση του Dataframe χρησιμοποιήθηκε η μορφή αρχείων CSV (Comma-Seperated Values). Στα τελικά σύνολα δεδομένων που έχουν τη μορφή CSV οι διαφορετικοί τύποι δεδομένων είναι διαχωρισμένοι με κόμμα (,). Ο λόγος επιλογής αυτής της μορφής αρχείων είναι η απλότητά της και η ευκολία στην αποθήκευση και προσπέλαση των δεδομένων με χρήση των εντολών save_to_csv("file anme") και read_csv("file name") της βιβλιοθήκης Pandas.

Προεπεξεργασία κειμένου (Text Preprocessing)

Στα βήματα της προεπεξεργασίας κειμένου χρησιμοποιήθηκε η βιβλιοθήκη regex της Python και δημιουργήθηκαν οι ακόλουθες συναρτήσεις.

Καθαρισμός Κειμένων -Text Cleaning

- Μετατροπή κεφαλαίων χαρακτήρων κάθε λέξης σε μικρούς (lowercasing).

```
2  import re
3
4  def lowercase_text(text):
5      text = text.lower()
```

- Αφαίρεση σημείων στίξης (πχ. ! . , ^ / - κτλ.)

```
2  import re
3
4  def remove_unacceptable_characters(string, regex=None):
5
6      if regex is None:
7          regex = re.compile(r'[^\w\s\d,!@#$%^&*()_=+{};:\<\>\[\]\.\/\?\'\"\\|`]', re.UNICODE)
8      return regex.sub(' ', string)
9
```

Συνάρτηση στην Python για την αφαίρεση σημείων στήξης.

- Αφαίρεση αριθμών.

```
4  def remove_numbers(string):
5
6      return ''.join(i for i in string if not i.isdigit())
```

Συνάρτηση στην Python για την αφαίρεση αριθμών.

- Αφαίρεση επιπρόσθετων χαρακτήρων κενού.

```
2  import re
3
4  def remove_multiple_whitespaces(string, regex=None):
5
6      if regex is None:
7          regex = re.compile(r'\s+', re.UNICODE)
8      return regex.sub(' ', string)
```

Συνάρτηση στην Python για την αφαίρεση επιπρόσθετων χαρακτήρων κενού.

- Αφαίρεση hashtag, email, url.

```
2  import re
3
4  def remove_emails(string, regex=None):
5
6      if regex is None:
7          regex = re.compile(r'[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,4})', flags=re.IGNORECASE)
8      return regex.sub(' ', string)
9
10 def remove_hashtags(string, regex=None):
11
12     if regex is None:
13         regex = re.compile(r'#\w+', flags=re.IGNORECASE)
14     return regex.sub(' ', string)
15
16
17 def remove_urls(string, regex=None):
18
19     if regex is None:
20         regex = re.compile(r'(((https?\:\/\/)|(www.))+((\w+\.)*))\S*[+\s\ ]*', flags=re.UNICODE)
21     return regex.sub(' ', string)
```

Συναρτήσεις στην Python για την αφαίρεση σημείων hashtag, email, url .

- **Διαχωρισμός των λέξεων – Tokenization**

Στο βήμα αυτό κάθε κείμενο μετατρέπεται από ένα σύνολο χαρακτήρων σε ένα σύνολο από λεκτικές μονάδες (tokens), δηλαδή λέξεις.

- **Λημματοποίηση (Lemmatization)**

Στο βήμα αυτό σε κάθε κείμενο οι λέξεις μετατρέπονται στην λέξη-ρίζα τους.

Τα βήματα του διαχωρισμού των λέξεων και της λημματοποίησης υλοποιήθηκαν με την χρήση της ακόλουθης συνάρτησης :

```
import re

def tokenize(text, SW=None):
    base_res = [ WordNetLemmatizer.lemmatize(word) for word in nltk_word_tokenize(text) if len(word) > 2]
    if SW is None: return [_ for _ in base_res if len(_)>2]
    else: return [_ for _ in base_res if len(_)>2 if _ not in SW]
```