ΑΛΓΟΡΙΘΜΟΙ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΓΙΑ ΤΟ ΦΙΛΤΡΑΡΙΣΜΑ ΤΩΝ SPAM E-MAIL

Η Διπλωματική Εργασία
παρουσιάστηκε ενώπιον
του Διδακτικού Προσωπικού του
Πανεπιστημίου Αιγαίου

των
Κανάρη Ιωάννη
Κανάρη Κωνσταντίνου
ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2005

Η ΤΡΙΜΕΛΗΣ ΕΠΙΤΡΟΠΗ ΔΙΔΑΣΚΟΝΤΩΝ ΕΠΙΚΥΡΩΝΕΙ
ΤΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
ΤΩΝ
ΚΑΝΑΡΗ ΙΩΑΝΝΗ
ΚΑΝΑΡΗ ΚΩΝΣΤΑΝΤΙΝΟΥ

ΣΤΑΜΑΤΑΤΟΣ ΕΥΣΤΑΘΙΟΣ, Επιβλέπων
20 Οκτωβρίου 2005
Τμήμα Μηχανικών Πληροφοριακών και
Επικοινωνιακών Συστημάτων

ΚΑΡΑΧΑΛΙΟΣ ΝΙΚΟΛΑΟΣ, Μέλος
Τμήμα Μαθηματικών

ΦΕΛΟΥΖΗΣ ΕΥΑΓΓΕΛΟΣ, Μέλος
Τμήμα Μαθηματικών

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ
ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2005

ΠΕΡΙΛΗΨΗ

Το θέμα της εργασίας είναι η κατηγοριοποίηση των email σε spam και non spam ή αλλιώς ham (αθώα email) και πιο συγκεκριμένα οι αλγόριθμοι μηχανικής μάθησης (machine learning algorithms) για το σκοπό αυτό. Στις επόμενες ενότητες γίνεται αναφορά σε αρκετούς γνωστούς αλγόριθμους αλλά ασχολούμαστε εκτενέστερα με τον αλγόριθμο Support Vector Machines (S.V.M.) με τον οποίο έχουμε κάνει μετρήσεις που αφορούν την αποτελεσματικότητά του σε σύγκριση με τον αλγόριθμο του Bayes όπως υλοποιήθηκε στην εργασία An Evaluation of Naive Bayesian Anti-Spam Filtering [1]. Στο κύριο μέρος της εργασίας γίνεται μια ανάλυση της απαραίτητης θεωρίας για την προσέγγιση προβλημάτων κατηγοριοποίησης και εν συνεχεία καταγράφονται οι μετρήσεις που έγιναν μαζί με τον απαραίτητο σχολιασμό και τις συγκρίσεις.

ΕΥΧΑΡΙΣΤΙΕΣ - ΑΦΙΕΡΩΣΕΙΣ

Ευχαριστούμε θερμά και αφιερώνουμε την εργασία αυτή στους γονείς μας που μας στήριξαν σε κάθε μας βήμα και είναι πάντα δίπλα μας.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ

1. ΕΙΣΑΓΩΓΗ

Τι είναι όμως τα spam και γιατί εδώ και αρκετά χρόνια αποτελούν αντικείμενο μελέτης από ερευνητές ανά τον κόσμο;

Τα spam ή αλλιώς Unsolicited Commercial Email (U.C.E.) ή Unsolicited Bulk Email (U.B.E) δεν είναι τίποτε άλλο από emails τα οποία έχουν διαφημιστικό χαρακτήρα και έχουν ως σκοπό την προώθηση κυρίως κάποιων προϊόντων που δεν μπορούν να διαφημιστούν (τουλάχιστον όχι στο βαθμό που διαφημίζονται άλλα προϊόντα) από τα συνηθισμένα μέσα μαζικής ενημέρωσης λόγω της λογοκρισίας που έχει εφαρμοστεί από τη συντριπτική πλειοψηφία των κρατών[20]. Τα προϊόντα αυτά μπορεί να είναι από χάπια αδυνατίσματος - αμφιβόλου ποιότητας και προέλευσης φυσικά - μέχρι και πορνογραφικό υλικό και τέτοια συναφή. Είναι λοιπόν προφανές το πόσο ενοχλητικά είναι στους χρήστες του διαδικτύου όχι μόνο για το πιθανό τους περιεχόμενο αλλά και για την ποσότητα με την οποία εισέρχονται στα ηλεκτρονικά γραμματοκιβώτιά τους. Εδώ πρέπει να σημειωθεί ότι το 75 % περίπου των email που διακινούνται στο διαδίκτυο είναι spam γεγονός το οποίο δείχνει το μέγεθος του προβλήματος που δημιουργείται. Βέβαια υπάρχουν στην αγορά αλλά και δωρεάν στο διαδίκτυο πολλά φίλτρα για την καταπολέμηση των spam τα οποία όμως δεν είναι πανάκεια γιατί υπάρχει η πιθανότητα και κάποια spam να εισέλθουν σε ένα ηλεκτρονικό γραμματοκιβώτιο αλλά και κάποια αθώα email να μη φτάσουν ποτέ στον παραλήπτη τους. Πιο αναλυτικά ο χρήστης πρέπει να ξοδεύει το χρόνο του να σβήνει ανεπιθύμητα email και σε περίπτωση που χρησιμοποιήσει κάποιο φίλτρο μπορεί να χάσει κάποιο αθώο email το οποίο να έχει ιδιαίτερη σημασία γι' αυτόν. Τέλος πολλά φίλτρα που χρησιμοποιούνται σε μεγάλες επιχειρήσεις ή κρατικούς οργανισμούς μπορεί να εμποδίζουν ακόμη περισσότερα αθώα email να εισέρχονται στο ηλεκτρονικό γραμματοκιβώτιό τους προκαλώντας έτσι την απώλεια σημαντικών πληροφοριών οικονομικής πολλές φορές αξίας.

Το φιλτράρισμα των spam γίνεται με πολλούς τρόπους. Τέσσερις από αυτούς είναι οι εξής[18][24] :

1.1 Real Time Blackhole Lists

Οι διακομιστές (servers) οι οποίοι είτε στέλνουν spam emails είτε απλώς τα αναμεταδίδουν είναι Simple Mail Transfer Protocol (SMTP) servers. Real Time Blackhole Lists ονομάζονται οι Domain Name Servers (DNS) οι οποίοι περιέχουν τις

διευθύνσεις IP (Internet Protocol Address) των SMTP servers οι οποίοι θεωρούνται ύποπτοι για αναμεταδότες spam emails. Το φιλτράρισμα με τη χρήση Real Time Blackhole List γίνεται με τρεις τρόπους. Πρώτον στα email που θεωρούνται spam μπαίνει μια ηλεκτρονική ετικέτα από τον email provider που δηλώνει ότι το email είναι spam και γίνεται η αποστολή του κανονικά στους παραλήπτες του οι οποίοι όμως γνωρίζουν πλέον αν είναι η όχι spam. Δεύτερον ο email provider μπλοκάρει κατευθείαν το ύποπτο email από τις συγκεκριμένες IP διευθύνσεις και τρίτον μπορεί να μπλοκάρει απευθείας κάθε δικτυακή κίνηση από και προς τις συγκεκριμένες διευθύνσεις. Σε γενικές γραμμές θεωρείται επιτυχημένο γιατί έχει μειώσει κατά πολύ την κίνηση των spam. Ένα μειονέκτημα που έχει όμως είναι ότι στην περίπτωση που ένα spam περάσει τον έλεγχο πρέπει ο χρήστης να δει εάν είναι spam και να το αναφέρει. Επίσης υπάρχει ο κίνδυνος ένας SMTP server ο οποίος όμως δε στέλνει spam να καταχωρηθεί κατά λάθος σε κάποιο Real Time Blackhole List, από μία απλή απροσεξία ενός χρήστη ο οποίος αποθήκευσε στα junk email, ένα email που έστειλε αυτός ο server[22][23].

1.2 Challenged Based Authentication

Αυτή η μέθοδος δουλεύει ως εξής: Κάθε email που λαμβάνεται πρέπει πρώτα να έχει γίνει η αναγνώριση του αποστολέα. Όταν ένα νέο email λαμβάνεταιο αποστολέας ελέγχεται με τη χρήση μιας βάσης δεδομένων γνωστών διευθύνσεων. Εάν ο αποστολέας είναι γνωστός τότε το email παραδίδεται αμέσως στον παραλήπτη του. Σε αντίθετη περίπτωση στέλνεται μια αίτηση στον αποστολέα η οποία ζητάει απάντηση έτσι ώστε να διαπιστωθεί η ύπαρξη του καθώς η πληθώρα των spam στέλνονται από ψεύτικες διευθύνσεις. Αν λοιπόν απαντήσει και γίνει δεκτός από το σύστημα τότε τα email που θα στέλνει θα γίνονται αυτομάτως αποδεκτά. Τα προγράμματα ASK(Active Spam Killer)[26], TMDA και Qconfirm είναι μερικά παραδείγματα εφαρμογών της μεθόδου Challenge Based Authentication με κάποιες διαφοροποιήσεις φυσικά.

1.3 Distributed Antispam Networks

Είναι μια τεχνική η οποία τοποθετεί ένα είδος ηλεκτρονικής υπογραφής σε κάθε εισερχόμενο email. Αυτές οι υπογραφές αγνοούν μικρές αλλαγές στο κείμενο

των email έτσι ώστε με μικροδιαφορές στο σώμα του spam ή στην κεφαλίδα του να παράγονται παρόμοιες υπογραφές. Στη συνέχεια μια κεντρική βάση δεδομένων εκτελεί έλεγχο για την υπογραφή που παράγεται από το εισερχόμενο email. Αν βρεθεί η ίδια ή μια παρόμοια το email χαρακτηρίζεται σα spam και απορρίπτεται. Οι χρήστες είναι υπεύθυνοι για την αναφορά των spam στη βάση δεδομένων με τη βοήθεια ενός ειδικού λογαριασμού στον οποίο προωθούνται. Μόλις ένα spam βρεθεί και δηλωθεί από ένα χρήστη όλοι οι υπόλοιποι θα προστατευθούν από αυτό το spam αλλά και από παρόμοια με αυτό. Αυτή η μέθοδος δουλεύει αρκετά καλά αφού οι πιθανότητες ένα αθώο email να χαρακτηριστεί spam είναι πολύ μικρές και οι πιθανότητες ένα spam email να πιαστεί είναι πολύ καλές. Το κύριο μειονέκτημά του όμως είναι ότι αν ένα τελείως διαφορετικό spam εισέλθει δεν θα ανιχνευτεί και θα πρέπει να γίνει πάλι η ίδια διαδικασία από το χρήστη. Αν λάβει κανείς υπ' όψη το πλήθος των spam που μπορεί να στέλνονται καθημερινά τότε είναι σαφές ότι χρειάζονται πολλοί περισσότεροι του ενός χρήστη για να αναφέρουν τα spam έτσι ώστε να δουλεύει ικανοποιητικά το σύστημα. Τα Vipul's Razor και Distributed Checksum Clearinghouse (DCC) είναι μερικές από τις εφαρμογές που χρησιμοποιούν αυτή τη λογική.

1.4 Content Based

Οι εφαρμογές που ανήκουν σε αυτήν την κατηγορία ανιχνεύουν τα spam ελέγχοντας το περιεχόμενο το εισερχόμενων email. Είναι υλοποιήσεις των αλγορίθμων μηχανικής μάθησης (machine learning) για την κατηγοριοποίηση κειμένου και κατ' επέκταση των email. Τα spam assassin και Bogofilter είναι μόνο μερικές από αυτές τις εφαρμογές. Η λογική των content based αλγορίθμων είναι η εξής: Όταν εισέρχεται ένα email αναλύεται πρώτα σε κάποια συγκεκριμένα δείγματα που είναι συνήθως οι λέξεις ή οι φράσεις. Αυτά τα δείγματα εξετάζονται με βάση κάποια χαρακτηριστικά τους όπως για παράδειγμα η συχνότητα εμφάνισής τους και ανάλογα με τον αλγόριθμο κρίνεται κατά πόσο το δείγμα είναι χαρακτηριστικό των spam ή των ham. Ενδεχομένως κάποια εξίσου συχνά εμφανιζόμενα δείγματα σε spam και ham να μην είναι καλά κριτήρια για το διαχωρισμό αυτόν οπότε και δε λαμβάνονται υπ' όψη. Όσα όμως κριθούν κατάλληλα γι' αυτή τη διαδικασία δίνουν ένα συνολικό score στο email το οποίο αν περνάει μία συγκεκριμένη τιμή (threshold) χαρακτηρίζεται σα spam και αν όχι σαν ham. Αναλυτικότερη περιγραφή αυτών των

αλγορίθμων δίνεται στην επόμενη ενότητα. Ας δούμε τώρα σε γενικές γραμμές τι είναι το machine learning που αναφέραμε προηγουμένως.

Το machine learning (μηχανική μάθηση) είναι ένας τομέας της τεχνητής νοημοσύνης ο οποίος ασχολείται με την ανάπτυξη τεχνικών με σκοπό να επιτρέπουν στον ηλεκτρονικό υπολογιστή να μαθαίνει. Πιο συγκεκριμένα είναι μια μέθοδος δημιουργίας προγραμμάτων για τον ηλεκτρονικό υπολογιστή μέσα από την ανάλυση κάποιων συνόλων από δεδομένα. Επίσης σχετίζεται σε μεγάλο βαθμό με τη στατιστική αφού και τα δύο αυτά επιστημονικά πεδία χρησιμοποιούνται για τη μελέτη των δεδομένων, αλλά σε αντίθεση με τη στατιστική το machine learning ασχολείται με την πολυπλοκότητα των υπολογιστικών εφαρμογών. Αξίζει εδώ να σημειωθεί ότι το machine learning έχει μία πληθώρα εφαρμογών η οποία περιλαμβάνει μηχανές αναζήτησης, ιατρικές γνωματεύσεις, ανίχνευση απάτης σχετιζόμενες με πιστωτικές κάρτες, ανάλυση της αγοράς μετοχών, κατηγοριοποίηση αλληλουχιών DNA, αναγνώριση φωνής και γραφικού χαρακτήρα κλπ. Τώρα στην περίπτωση των spam τα σύνολα δεδομένων που χρησιμοποιούνται είναι κατηγοριοποιημένα εξ' αρχής email δηλαδή spam και ham οπότε στην ουσία το πρόγραμμα που θα κάνει την ανάλυση των email θα κατανοήσει πως είναι τα spam πως είναι τα ham και θα μάθει να τα ξεχωρίζει.

2. ΠΡΟΗΓΟΥΜΕΝΕΣ ΠΡΟΣΕΓΓΙΣΕΙΣ

Σε αυτήν την ενότητα παραθέτουμε μερικούς από τους πιο γνωστούς αλγόριθμους για την κατηγοριοποίηση των email σε spam και non spam καθώς επίσης και κάποιες εργασίες στις οποίες υλοποιούνται με τα αποτελέσματά τους. Οι αλγόριθμοι αυτοί είναι οι εξής : Bayesian, SVM (Support Vector Machines), Rocchio, Find Similar και Suffix Tree. Πριν προχωρήσουμε όμως με την ανάλυση των αλγορίθμων θα πρέπει να θέσουμε μία κοινή βάση πάνω στην οποία δουλεύουν, δηλαδή ποια βήματα ακολουθούνται πριν εφαρμοστεί στην ουσία ο εκάστοτε αλγόριθμος. Το πρώτο βήμα είναι να επιλέξουμε με τι είδους δεδομένα θα εργαστούμε όπως για παράδειγμα λέξεις, φράσεις, n-γράμματα (αλληλουχίες γραμμάτων) αλλά και άλλα δείγματα. Το επόμενο βήμα είναι να εξάγουμε τα χαρακτηριστικά που θέλουμε από τα επιλεγμένα δείγματα. Βέβαια κάποιοι αλγόριθμοι ενδέχεται να θέτουν κάποιους περιορισμούς ως προς αυτά τα

χαρακτηριστικά γεγονός το οποίο ξεφεύγει από το σκοπό αυτού του κειμένου. Τα χαρακτηριστικά που χρησιμοποιούνται συχνότερα είναι τα εξής[6]:

1. TF (Term Frequency) – Συχνότητα εμφάνισης δείγματος: Είναι ο λόγος του πλήθους των εμφανίσεων του δείγματος δια του πλήθους των συνολικών εμφανίσεων των δειγμάτων μέσα στο email.

2. TF-IDF (Term Frequency * Inverse Document Frequency): Είναι το γινόμενο της συχνότητας εμφάνισης του δείγματος με την αντίστροφη συχνότητα εμφάνισης (IDF) σε όλα τα email.

3. Binary representation – Δυαδική εκπροσώπηση: Είναι απλά ένας δείκτης ο οποίος φανερώνει αν ένα δείγμα υπάρχει μέσα σε ένα email ή όχι. Αν υπάρχει σημειώνεται μία μονάδα αν δεν υπάρχει σημειώνεται ένα μηδενικό.

2.1 Bayesian Method

Ένας από τους πιο δημοφιλής αλγόριθμους για την κατηγοριοποίηση των email σε spam και non spam είναι ο αλγόριθμος του Thomas Bayes (1702-1761)[15]20]. Οι εφαρμογές που ενσωματώνουν ή στηρίζονται σε αυτόν τον αλγόριθμο είναι πάρα πολλές γεγονός που αποδεικνύει την παραπάνω πρόταση. Ο αλγόριθμος του Bayes βασίζεται στο γνωστό θεώρημά του που δημοσιεύτηκε το 1763 και έδινε τον τύπο υπολογισμού της δεσμευμένης πιθανότητας, δηλαδή την πιθανότητα ενός ενδεχομένου με δεδομένο το ότι ένα άλλο ενδεχόμενο έχει συμβεί[10].
Ένα φίλτρο που εφαρμόζει αυτόν τον αλγόριθμο πρέπει να ικανοποιεί τα παρακάτω:

1. Το φίλτρο πρέπει να έχει τη δυνατότητα να διαβάζει μια συλλογή από πολλά email (spam και non spam) και στη συνέχεια να τα απαριθμεί.

2.Πρέπει να διαβάζει τη συχνότητα κάθε λέξης ή όποιου άλλου δείγματος έχουμε επιλέξει που εμφανίζεται στη συλλογή και να αποθηκεύεται μαζί με τη συχνότητα του.

3.Στην περίπτωση που το δείγμα είναι λέξη είναι όπως έχει αποδειχθεί και σε προηγούμενα πειράματα προτιμότερο να εφαρμόζεται η διαδικασία **word stemming** και η διαδικασία **stop word removal**[1][9][25]. Στη διαδικασία word stemming κάποιες παράγωγες λέξεις πχ. ran , run αποθηκεύονται μόνο μία φορά σαν run. Στη διαδικασία stop word removal αφαιρούνται τα stop words τα οποία είναι λέξεις που χρησιμοποιούνται συχνότερα από όλες και δεν είναι καλό δείγμα ούτε για spam αλλά ούτε για non spam όπως για παράδειγμα οι λέξεις and, of, from, the κλπ. Εάν δεν είναι λέξεις και είναι για παράδειγμα ακολουθίες χαρακτήρων μήκους n τα λεγόμενα n-grams δηλαδή τότε απλά θέτουμε μια συχνότητα εμφάνισης των n-grams κάτω από την οποία τα αντίστοιχα n-grams να αφαιρούνται.

4.Το φίλτρο πρέπει να εφαρμόζει μια μορφή της εξίσωσης του Bayes για να μετράει την πιθανότητα ενός email να είναι spam. Η εξίσωση του Bayes σε απλουστευμένη μορφή δηλαδή για δύο ενδεχόμενα A και B (spam και non spam) με x να είναι η πιθανότητα ένα spam email να περιέχει μία λέξη ή ενός non spam να περιέχει την ίδια λέξη αντίστοιχα είναι η παρακάτω:

$$P(A|x) = \frac{P(x \mid A) * P(A)}{(P(x \mid A) * P(A)) + (P(x \mid B) * P(B))}$$

5.Το φίλτρο πρέπει να έχει τη δυνατότητα να διαβάζει για δοκιμαστικούς σκοπούς ένα σύνολο αποτελούμενο από non spam και από spam email και να αξιολογεί κάθε email ξεχωριστά για το αν είναι spam ή όχι.

6.Τα spam και non spam email που δοκιμάσαμε δε θα πρέπει να ανήκουν στη συλλογή από την οποία λάβαμε τις συχνότητες των δειγμάτων.

7.Στόχος είναι φυσικά αν είναι δυνατόν όλα τα spam email να ανιχνευτούν ως spam και να μην ανιχνευτεί κανένα “αθώο” email ως spam.

Με άλλα λόγια η βασική ιδέα είναι να τροφοδοτούμε το φίλτρο που εφαρμόζει αυτόν τον αλγόριθμο με όσο το δυνατόν περισσότερα email spam και non spam (ham) έτσι ώστε να εκπαιδευτεί στο τι να περιμένει σε ένα spam ή ένα ham.

Επιλέγουμε λοιπόν τον τύπο των δειγμάτων-δεδομένων με τον οποίο έχουμε επιλέξει να δουλεύουμε δηλαδή λέξεις, φράσεις, n-grams κ.ο.κ και στη συνέχεια καταγράφουμε το δείγμα υπολογίζοντας συγχρόνως τη συχνότητα εμφάνισής του. Αυτό εφαρμόζεται φυσικά και για τις δύο συλλογές email δηλαδή spam και ham. Με αυτόν τον τρόπο προκύπτουν δύο τεράστιες λίστες δειγμάτων μαζί με τις συχνότητές τους. Επίσης χρησιμοποιώντας τον αριθμό των emails σε κάθε συλλογή η πιθανότητα μία λέξη να προδίδει ότι το email είναι "spam" μπορεί να υπολογιστεί με τη βοήθεια της εξίσωσης του Bayes που προαναφέραμε. Ωστόσο γεννιέται το ερώτημα τι πιθανότητα να δώσει κανείς σε ένα δείγμα το οποίο εμφανίζεται στη μία συλλογή και δεν εμφανίζεται στην άλλη. Εδώ ο σχεδιαστής του φίλτρου μπορεί να πειραματιστεί ελεύθερα με τιμές που θα δίνουν καλά συνολικά αποτελέσματα όταν δοκιμάζονται emails, που λογικά θα κυμαίνονται μεταξύ 0.99 και 0.01.

Άλλος ένας παράγοντας που πρέπει να σκεφτεί κανείς είναι το τι συμβαίνει όταν ένα δείγμα ενός email δεν έχει εμφανιστεί προηγουμένως σε καμία από τις δύο συλλογές. Φυσιολογικά και στην περίπτωση των spam αλλά και των ham ένα άγνωστο δείγμα είναι προφανώς αθώο οπότε μπορεί να του δοθεί μια τιμή για την πιθανότητά του της τάξης του 0.4 ή μια παραπλήσια. Ο λόγος για αυτήν την επιλογή θα εξηγηθεί παρακάτω.

Αναφορικά τώρα με τα δείγματα τα οποία δεν εμφανίζονται σε καμία από τις δύο συλλογές, αν για παράδειγμα ένα από αυτά αρχίζει να εμφανίζεται σε spam τότε όσο κάθε email σκανάρεται τα δείγματά του αυτά ελέγχονται για την εισαγωγή του ή μη στη συλλογή. Αυτό είναι μάλιστα ένα από τα κύρια πλεονεκτήματα στη σχεδίαση ενός φίλτρου που εφαρμόζει τον αλγόριθμο του Bayes, η ικανότητά του να μαθαίνει. Όσο εισάγονται νέα email, είτε είναι spam είτε όχι τα δείγματα του νέου email προστίθενται στη σχετική συλλογή με τις συχνότητες των αναγνωρισμένων δειγμάτων του να προσαυξάνονται. Με αυτόν τον τρόπο τα στατιστικά φίλτρα μαθαίνουν συνεχώς νέες λέξεις και τις προσθέτουν στη βάση δεδομένων των δειγμάτων και των αντίστοιχών τους πιθανοτήτων.

Με εφόδιο πλέον έναν πίνακα με πιθανότητες για εκατοντάδες χιλιάδες δείγματα είναι κανείς έτοιμος να λάβει emails και να τα αξιολογήσει για το αν είναι spam. Όταν ένα email λαμβάνεται το φίλτρο το κατακερματίζει σε δείγματα με τον

ίδιο τρόπο που χωρίστηκαν οι δύο συλλογές. Κάθε δείγμα ενός email έχει την πιθανότητα να καταγραφεί στα spam. Με μία λίστα από πιθανότητες κάθε δείγματος
Η συνολική πιθανότητα ότι ένα ολόκληρο email είναι spam μπορεί να βρεθεί παίρνοντας τις δεκαπέντε πιο ακραίες πιθανότητες από το 0.5 που είναι το ουδέτερο σημείο προς κάθε κατεύθυνση κι έτσι δικαιολογούμε και την επιλογή της τιμής 0.4 που κάναμε παραπάνω όντας πολύ κοντά στο 0.5. Χρησιμοποιώντας αυτήν την τεχνική παίρνουμε τα πιο ακραία παραδείγματα δειγμάτων, αυτά δηλαδή που είναι πιο πιθανό να είναι "λέξεις-κλειδιά" για spam και αυτά που είναι πιο πιθανό να είναι "λέξεις-κλειδιά" για ham. Στη συνέχεια χρησιμοποιείται η παρακάτω εξίσωση:

$$P(Spam)=\frac{P(w1\mid Spam)...P(w15\mid Spam)}{(P(w1\mid Spam)...P(w15\mid Spam))+(P(w1\mid NonSpam)...P(w15\mid NonSpam))}$$

Ο τελικός υπολογισμός για τις πιθανότητες των email ξεχωριστά

Η τιμή της παραπάνω συνάρτησης που απαιτείται συνήθως για το χαρακτηρισμό ενός email ως spam (το λεγόμενο threshold) είναι το 0.9. Αυτό μπορεί να φαίνεται πολύ υψηλό αλλά όταν γίνονται δοκιμές με στατιστικά φίλτρα τα πιο πολλά αθώα email έχουν εξαιρετικά χαμηλή πιθανότητα περίπου 0.00000001. Ομοίως τα spam έχουν συνήθως πιθανότητα 1 ή πολύ κοντά σε αυτό.

Παλαιότερα χρησιμοποιούνταν όλες οι πιθανότητες όλων των δειγμάτων για την αξιολόγηση του email αλλά οι spammers εισήγαγαν παραγράφους με αθώο περιεχόμενο κρυμμένο μέσα στον κώδικα html ο οποίος δε φαίνεται στον αναγνώστη αλλά παρ' όλα αυτά αλλοιώνει τα στατιστικά και επισκιάζει τις πιθανότητες των spam κλειδιών.

2.2 Naive Bayes

Η πιο γνωστή ίσως παραλλαγή του αλγόριθμου του Bayes είναι η μέθοδος **Naive Bayes**[1][5][9][18][19][21]. Έστω $X = (X_1, X_2,...,X_d)$ το διάνυσμα των όρων για ένα τυχαίο email όπου **d** είναι το πλήθος των διαφορετικών δειγμάτων στα εκπαιδευτικά δεδομένα. Έστω Y ο αντίστοιχος χαρακτηρισμός (είναι ή όχι spam) των email. Η μέθοδος αυτή έχει ως σκοπό την κατασκευή ενός μοντέλου για το :

$$\Pr(Y=1|X_1=x_1,\ldots,X_d=x_d).$$

Από το θεώρημα του Bayes έχουμε :

$$\Pr(Y=1|X_1=x_1,\ldots,X_d=x_d)=\frac{\Pr(Y=1)*\Pr(X_1=x_1,\ldots,X_d=x_d\mid Y=1)}{\Pr(X_1=x_1,\ldots,X_d=x_d)} \qquad (1)$$

ή διαφορετικά στην κλίμακα των λογαριθμικών πιθανοτήτων

$$\log\frac{\Pr(Y=1|X_1=x_1,\ldots,X_d=x_d)}{\Pr(Y=0|X_1=x_1,\ldots,X_d=x_d)}=\log\frac{\Pr(Y=1)}{\Pr(Y=0)}+\log\frac{\Pr(X_1=x_1,\ldots,X_d=x_d\mid Y=1)}{\Pr(X_1=x_1,\ldots,X_d=x_d\mid Y=0)}$$

(2)

Η κλίμακα αυτή αποφεύγει τη σταθερά ομαλοποίησης στον παρονομαστή του δεύτερου μέλους της σχέσης (1).

Ο πρώτος όρος στο δεξί μέλος της σχέσης (2) εμπλέκει την προηγούμενη πιθανότητα ενός email να είναι spam. Ο δεύτερος όρος στο δεξί μέλος της σχέσης (2) εμπλέκει δύο δεσμευμένες πιθανότητες, συγκεκριμένα τη δεσμευμένη πιθανότητα ενός διανύσματος ενός όρου, δεδομένου ότι το μήνυμα του διανύσματος του όρου είναι spam και τη δεσμευμένη πιθανότητα ενός διανύσματος όρου, δεδομένου ότι το μήνυμα του διανύσματος αυτού δεν είναι spam. Αυτό είναι λίγο προβληματικό διότι εμπλέκει τη διανομή των από κοινού πιθανοτήτων των d τυχαίων μεταβλητών, $X_1,\ldots,X_d$ όπου το d είναι αρκετά μεγάλο. Η βασική υπόθεση του μοντέλου Naive Bayes είναι ότι αυτές οι τυχαίες μεταβλητές είναι κατά συνθήκη ανεξάρτητες δεδομένου Y. Δηλαδή:

$$\Pr(X_1=x_1,\ldots,X_d=x_d\mid Y=1)=\prod_{i=1}^{d}\Pr(X_i=x_i\mid Y=1)$$

και

$$\Pr(X_1=x_1,\ldots,X_d=x_d\mid Y=0)=\prod_{i=1}^{d}\Pr(X_i=x_i\mid Y=0)$$

το οποίο μας δίνει:

$$\log\frac{\Pr(Y=1\mid X_1=x_1,\ldots,X_d=x_d)}{\Pr(Y=0\mid X_1=x_1,\ldots,X_d=x_d)}=\log\frac{\Pr(Y=1)}{\Pr(Y=0)}+\sum_{i=1}^{d}\log\frac{\Pr(X_i=x_i\mid Y=1)}{\Pr(X_i=x_i\mid Y=0)}.$$

(3)

Αυτή η υπόθεση ανεξαρτησίας είναι απίθανο να αντικατοπτρίζει την πραγματικότητα.

Ωστόσο μας παρέχει μια δραστική μείωση στον αριθμό των διακριτών πιθανοτήτων που πρέπει να υπολογίσουμε από το εκπαιδευτικό σύνολο και μάλιστα συχνά αποδίδει καλά στην πράξη.

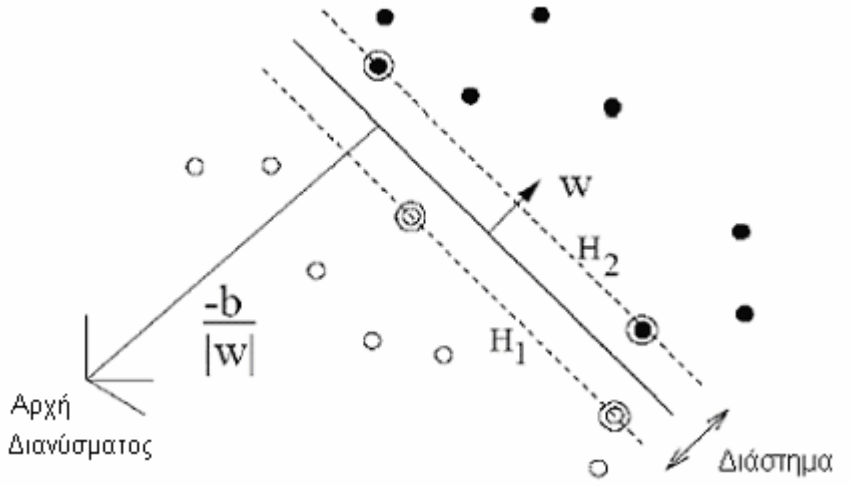
Μία πολύ γνωστή εφαρμογή του αλγόριθμου του Bayes είναι το **CRM114**[3] [20] το οποίο είναι στην ουσία μια γλώσσα προγραμματισμού που φτιάχνει τέτοιου είδους φίλτρα. Αυτά μάλιστα αποτρέπουν τα spam που ενσωματώνουν word generators τα οποία είτε γεμίζουν το email με λέξεις, είτε αλλοιώνουν κάποιες λέξεις όπως για παράδειγμα η λέξη FREE γίνεται F.R.E.E. έτσι ώστε να πέσει το ποσοστό πάνω από το οποίο ένα email είναι spam το λεγόμενο threshold[4]. Επίσης εκτός από λέξεις παίρνει και φράσεις και εφαρμόζει πάνω τους τον αλγόριθμο του Bayes.Αυτό σημαίνει ότι αν για παράδειγμα υπάρχουν σε ένα email οι λέξεις Dear Sir οι οποίες από μόνες τους δεν προδίδουν ότι το email είναι spam σαν φράση συναντάται πολύ συχνά σε spam οπότε κι ανιχνεύεται.

2.3 Support Vector Machines (SVM)

Ο αλγόριθμος SVM είναι ο αλγόριθμος ο οποίος επιλέξαμε να δοκιμάσουμε συγκρίνοντάς τον με τον αλγόριθμο Bayes. Οι λόγοι για την επιλογή αυτή σχετίζονται κυρίως με την αποτελεσματικότητα και την ταχύτητα του αλγόριθμου. Ας τα πάρουμε όμως τα πράγματα από την αρχή. Τι είναι ο αλγόριθμος SVM;

Είναι μια τεχνική που χρησιμοποιείται για κατηγοριοποίηση δεδομένων γενικότερα και εφαρμόζεται με πολύ μεγάλη επιτυχία στην κατηγοριοποίηση των αρχείων κειμένου και κατ' επέκταση των email την οποία αφορά το θέμα αυτής της εργασίας. Δημιουργήθηκε από τους Vapnik και Chervonenkis το 1992 και συγκαταλέγεται ανάμεσα στους πιο αποδοτικούς κατηγοριοποιητές καθώς έχει μια μοναδική ικανότητα να χειρίζεται ιδιαίτερα μεγάλα σύνολα χαρακτήρων όπως για παράδειγμα μεγάλα σε όγκο είδη κειμένου. Ο SVM αλγόριθμος λειτουργεί ως εξής: χαρτογραφεί το δοθέν εκπαιδευτικό σύνολο-στη περίπτωση που εξετάζουμε ένα σύνολο από email spam και μη- σε ένα πιθανό πολυδιάστατο χώρο διανυσμάτων και προσπαθεί να εντοπίσει σε αυτό το χώρο ένα πεδίο το οποίο να διαχωρίζει τα θετικά από τα αρνητικά παραδείγματα. Έχοντας βρει ένα τέτοιο πεδίο ο αλγόριθμος μπορεί να προβλέψει την κατηγοριοποίηση ενός αχαρακτήριστου παραδείγματος χαρτογραφώντας το στον χώρο που περιέχει τα χαρακτηριστικά και ψάχνοντας σε

ποια πλευρά του διαχωριστικού πεδίου βρίσκεται. Πώς όμως διαλέγουμε το διαχωριστικό πεδίο τη στιγμή που υπάρχουν πολλά υποψήφια ; Ο SVM αλγόριθμος επιλέγει το πεδίο που διατηρεί το μεγαλύτερο διάστημα μεταξύ οποιουδήποτε σημείου στο εκπαιδευτικό σύνολο.



<u>Σχήμα 2.3.1</u> Γραμμικώς διαχωρισμένα πεδία για τη ευδιάκριτη γραμμική περίπτωση.

Πιο αναλυτικά όλα τα διανύσματα εισόδου μπορούν να χωριστούν από τα πεδία H_1 και H_2. Κάποια διανύσματα της περιοχής του χώρου της μίας κατηγορίας είναι πιο κοντά στην περιοχή του χώρου μιας άλλης κατηγορίας. Τα διανύσματα αυτά που βρίσκονται στο πεδίο H_1 και στο πεδίο H_2 ονομάζονται support vectors (διανύσματα υποστήριξης) και είναι κυκλωμένα στο παραπάνω σχήμα.

Ο στόχος του αλγόριθμου είναι να επιλέξει ένα διαχωριστικό πεδίο $(w \cdot x_i + b) = 0$ το οποίο μεγιστοποιεί το διάστημα μεταξύ του H_1 $(w \cdot x_i + b) = -1$ και του H_2 $(w \cdot x_i + b) = 1$.

Αυτό υλοποιείται ως εξής : υποθέτουμε ότι όλα τα εκπαιδευτικά δεδομένα ικανοποιούν τους παρακάτω περιορισμούς :

$$y_i \, (w \cdot x_i + b) \geq 1$$

όπου y_i είναι η αντίστοιχη ζητούμενη τιμή. Αν $y_i = 1$ τότε αυτό σημαίνει ότι το x_i ανήκει στην κατηγορία 1 και αν $y_i = -1$ τότε το x_i ανήκει στην κατηγορία 2.

Για ένα πεδίο $(w \cdot x_i + b) = 0$, η απόσταση από το πεδίο στην αρχή του διανύσματος x_i είναι $|b| / \|w\|$.Επομένως η απόσταση του H_1 από την αρχή του

διανύσματος είναι $|b|+1/\|w\|$ και αντίστοιχα η απόσταση του H_2 από την αρχή του διανύσματος είναι $|b|-1/\|w\|$.Άρα το διάστημα μεταξύ του H_1 και του H_2 είναι $2/\|w\|$ οπότε και μπορούμε να βρούμε ένα ζεύγος πεδίων που θα δίνει το μέγιστο διάστημα ελαχιστοποιώντας την ποσότητα $\|w\|^2$ λαμβάνοντας υπ' όψη τους περιορισμούς που αναφέραμε προηγουμένως. Το πρόβλημα αυτό το αποκαλούμε πρωτεύον πρόβλημα.

Τώρα θα μεταβούμε σε ένα Lagrange-ιανό σχηματισμό του προβλήματος. Δοθέντων των θετικών πολλαπλασιαστών Lagrange α_i για κάθε περιορισμό ανισότητας ο Lagrange-ιανός αυτός σχηματισμός υλοποιείται ως εξής:

$$\mathrm{Lp} = \frac{1}{2}\|w\|^2 - \sum_i a_i\left[y_i(w\cdot x_i - b) - 1\right]$$

Κατόπιν πρέπει να ελαχιστοποιήσουμε την ποσότητα **Lp** ως προς τα **w** και **b**, και συγχρόνως να απαιτήσουμε οι παράγωγοι του **Lp** ως προς όλα τα α_i να εξαφανιστούν. Αυτό είναι ισοδύναμο με το να λύσουμε το παρακάτω δυαδικό πρόβλημα.

Απαιτώντας το διάνυσμα βαθμίδας (gradient) του **Lp** ως προς τα **w** και **b** να εξαφανιστεί δίνουμε τις συνθήκες:

$$\mathrm{w}(\alpha) = \sum_i a_i y_i x_i$$

$$\sum_i a_i y_i = 0$$

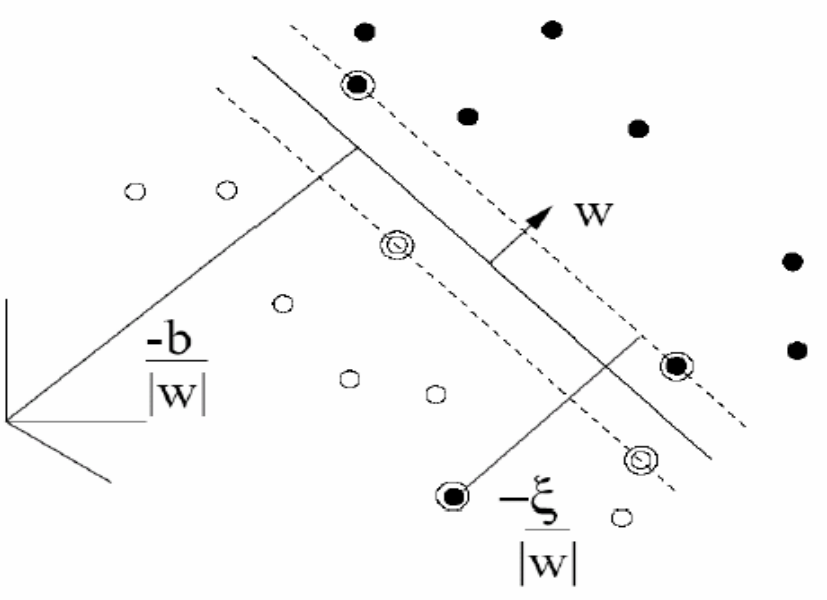
Μπορούμε να τις αντικαταστήσουμε στην εξίσωση

$$\mathrm{Lp} = \frac{1}{2}\|w\|^2 - \sum_i a_i\left[y_i(w\cdot x_i - b) - 1\right]$$

και να πάρουμε την εξής σχέση:

$$\mathrm{L_D} = \sum_i a_i - \frac{1}{2} w(a)\cdot w(a)$$

Η εκπαίδευση με αυτόν τον αλγόριθμο (για την ευδιάκριτη, γραμμική περίπτωση) ωστόσο ισοδυναμεί με τη μεγιστοποίηση της ποσότητας **Lᴅ** ως προς τα α_i λαμβάνοντας υπ'όψη τους περιορισμούς $\sum_i a_i y_i = 0$ και $\alpha_i \geq 0$.



<u>Σχήμα 2.3.2</u> Γραμμικώς διαχωρισμένα πεδία για τη μη ευδιάκριτη γραμμική περίπτωση.

Για τη μη ευδιάκριτη περίπτωση όπως φαίνεται στο παραπάνω σχήμα θα χαλαρώσουμε λίγο τους περιορισμούς $y_i\ (w\cdot x_i+b) \geq 1$ εισάγοντας θετικές μεταβλητές ξ_i, i = 1,2,…,I στους περιορισμούς οι οποίοι γίνονται

$$y_i(w \cdot x_i+b) \geq 1- \xi_i, \xi_i \geq 0, \forall i$$

Ωστόσο για να εμφανιστεί ένα σφάλμα στη μη ευδιάκριτη περίπτωση το αντίστοιχο ξ_i πρέπει να είναι μεγαλύτερο του ενός και οπότε το $\sum_i \xi_i$ είναι το άνω όριο του αριθμού των εκπαιδευτικών λαθών. Ένας φυσικός τρόπος να αναθέσουμε ένα επιπλέον κόστος για σφάλματα είναι να αλλάξουμε την αντικειμενική συνάρτηση έτσι ώστε να ελαχιστοποιείται από $\|w\|^2/2$ σε $\|w\|^2/2+C\sum_i \xi_i$ όπου **C** είναι μια παράμετρος η οποία επιλέγετε από τον χρήστη. Ένα μεγάλο **C** αντιστοιχεί στην ανάθεση μεγαλύτερης ποινής στα σφάλματα. Όμοια με την ευδιάκριτη περίπτωση είναι επίσης ένα δευτερεύον προγραμματιστικό πρόβλημα και μπορούμε να το λύσουμε μεγιστοποιώντας τη δυαδική μορφή :

$$L_D = \sum_i a_i - \frac{1}{2} w(a) \cdot w(a)$$

με περιορισμούς :

$$0 \le \boldsymbol{\alpha}_i \le \mathbf{C}$$

και

$$\sum_i a_i y_i = 0$$

δεδομένης της σχέσης

$$w(\alpha)=\sum_i^{Ns} a_i y_i x_i$$

όπου **Ns** είναι ο αριθμός των support vectors. Ωστόσο η μόνη διαφορά από τη βέλτιστη περίπτωση πεδίων είναι το ότι το $\boldsymbol{\alpha}_i$ έχει πλέον ένα άνω όριο το **C.**

Οι μέθοδοι μπορούν να γενικευτούν στην περίπτωση που η συνάρτηση απόφασης (decision function) είναι μια μη γραμμική συνάρτηση των δεδομένων. Ας υποθέσουμε ότι πρώτα χαρτογραφούμε τα δεδομένα σε κάποιο άλλο πολυδιάστατο χώρο **H,** χρησιμοποιώντας ένα μη γραμμικό μετασχηματισμό $\mathbf{Z}_i = \boldsymbol{\Phi}(\mathbf{X}_i)$. Τότε το ίδιο πρόβλημα μπορεί να μορφοποιηθεί σε ένα πολυδιάστατο χώρο. Τα εσωτερικά γινόμενα της μορφής $\boldsymbol{\Phi}(\mathbf{X}_i) \cdot \boldsymbol{\Phi}(\mathbf{X}_j)$ θα χρησιμοποιούνται για την εκπαίδευση του SVM. Συνήθως η συνάρτηση **Φ** θεωρείται άγνωστη ενώ αντιθέτως ορίζεται μία συνάρτηση πυρήνα $\mathbf{k(x,}\hat{x}\mathbf{)} = \boldsymbol{\Phi}\mathbf{(x)}\cdot\boldsymbol{\Phi}(\hat{\mathrm{x}})$**.**

Πολλοί γνωστοί πυρήνες περιλαμβάνουν:

Πολυωνυμικές συναρτήσεις :

$$\mathbf{K(X,Y)} = (1+\mathbf{X}\cdot\mathbf{Y})^d$$

Συναρτήσεις Radial Basis(RBF):
Radial Basis Functions

$$\mathbf{K(X,Y)} = \mathbf{exp}(-\|\mathrm{X}-\mathrm{Y}\|^2 / 2\sigma^2)$$

Sigmoidal:

$$\mathbf{K(X,Y)} = \mathbf{tanh}(\mathbf{k}_1\mathbf{X}\cdot\mathbf{Y} + \mathbf{k}_2)$$

Για την επίλυση του δυαδικού προβλήματος εμείς χρησιμοποιήσαμε τον αλγόριθμο SMO (Sequential Minimal Optimization) του Platt, ο οποίος χρησιμοποιεί ένα σύνολο μεγέθους δύο των διανυσμάτων σαν σύνολο εργασίας και βελτιστοποιεί τα αντίστοιχα $\boldsymbol{\alpha}_i$ ενώ παγώνει τα υπόλοιπα. Το SMO είναι αρκετά γρήγορο και εφαρμόζεται από το πρόγραμμα WEKA το οποίο χρησιμοποιήσαμε για να

εκτιμήσουμε τις επιδόσεις του αλγόριθμου SVM για το φιλτράρισμα των spam emails[2][5][6][11][12][13][21].

2.4 Rocchio Method

Αυτή η μέθοδος χρησιμοποιεί ομαλοποιημένη TF-IDF εκπροσώπηση των εκπαιδευτικών διανυσμάτων. Ένα πρωτότυπο διάνυσμα w σχηματίζεται ως εξής:

$$\mathrm{w} = \frac{1}{N_{spam}} \sum_{i \in spam} x_i - \beta \frac{1}{N_{nonspam}} \sum_{i \in nonspam} x_i$$

όπου το N αντιπροσωπεύει το πλήθος των email που κατηγοριοποιούνται ως spam ή non spam. Τα στοιχεία του πρωτότυπου διανύσματος τα οποία είναι αρνητικά μηδενίζονται και τότε το w ομαλοποιείται σε μήκος μονάδας. Η κατηγοριοποίηση εκτελείται με βάση το εσωτερικό γινόμενο του πρωτότυπου διανύσματος και του υποψήφιου δοκιμαστικού διανύσματος. Αυτά που έχουν μεγάλα θετικά εσωτερικά γινόμενα είναι spam και αυτά τα οποία έχουν μεγάλες αρνητικές τιμές είναι non spam.Σε αντίθεση με άλλους αλγόριθμους δεν υπάρχει κάποια φυσική «ιδανική τιμή» για το εσωτερικό γινόμενο. Αυτό σημαίνει ότι ο αλγόριθμος δε μας λέει για ποιες τιμές πάνω από μία «ιδανική τιμή» του εσωτερικού γινομένου πρέπει να κατηγοριοποιούμε το email σαν spam.Αυτή η τιμή πρέπει να αποκομίζεται αντιστοιχώντας τα αποτελέσματα των εσωτερικών γινομένων του πρωτότυπου διανύσματος που προαναφέραμε με όλα τα εκπαιδευτικά διανύσματα και βρίσκοντας ποια τιμή ελαχιστοποιεί το σφάλμα εκπαίδευσης.

Σε αυτό το σημείο τονίζουμε ότι δεν πρέπει να χρησιμοποιηθούν τα δοκιμαστικά διανύσματα για να βρούμε αυτήν την τιμή. Ομοίως η ιδανική τιμή του β δε πρέπει να εξαχθεί από το δοκιμαστικό σύνολο (test set) αλλά από το εκπαιδευτικό σύνολο (training set) και μάλιστα είναι αυτό ακριβώς το β που ελαχιστοποιεί το εκπαιδευτικό σφάλμα.. Το πλεονέκτημα του αλγόριθμου Rocchio είναι η ταχύτητά του στην εκπαίδευση και στη δοκιμή. Το μειονέκτημα είναι ότι πρέπει κανείς να ψάξει μόνος του αυτήν την «ιδανική τιμή» και το ιδανικό β στο εκπαιδευτικό σύνολο το οποίο απαιτεί επιπλέον χρόνο εκπαίδευσης και δεν γενικεύεται –απαραίτητα- καλά στο δοκιμαστικό σύνολο(test set)[6].

2.5 Find Similar (Εύρεση Ομοίων)

Η μέθοδος Find Similar είναι μια παραλλαγή του αλγόριθμου Rocchio για ανατροφοδότηση σχέσεων, η οποία είναι μια δημοφιλής μέθοδος για την επέκταση των ερωτημάτων των χρηστών στη βάση των σχεσιακών κρίσεων. Στη διατύπωση του Rocchio το βάρος που αποδίδεται σε έναν όρο είναι ένας συνδυασμός του βάρους του σε ένα αρχικό ερώτημα και των σχετικών και άσχετων εγγράφων που έχουν κριθεί.

$$\mathrm{x}_j = \alpha \cdot \mathrm{x}_{q,j} + \beta \cdot \frac{\sum_{i \in rel} x_{i,j}}{n_r} + \gamma \cdot \frac{\sum_{i \in non-rel} x_{i.j}}{N - n_r}$$

Οι παράμετροι α, β και γ ελέγχουν τη σχετική σημασία του αρχικού διανύσματος, τα θετικά παραδείγματα και τα αρνητικά παραδείγματα. Στο γενικό πλαίσιο της κατηγοριοποίησης κειμένου δεν υπάρχει αρχικό ερώτημα, οπότε α = 0. Επίσης θέτουμε γ = 0 για να μπορούμε εύκολα να χρησιμοποιήσουμε διαθέσιμο κώδικα. Ωστόσο έτσι όπως εφαρμόζεται συνήθως η μέθοδος Find Similar το βάρος του κάθε όρου είναι απλώς ο μέσος όρος των βαρών σε θετικά παραδείγματα της κατηγορίας.

Δεν υπάρχει σαφής ελαχιστοποίηση σφάλματος η οποία να εμπλέκεται στον υπολογισμό των βαρών Find Similar. Παρ'όλα αυτά δεν υπάρχει αυτό που λέμε χρόνος εκμάθησης παρά μόνο για το σύνολο των βαρών από θετικά παραδείγματα της κάθε κατηγορίας. Τα δοκιμαστικά παραδείγματα κατηγοριοποιούνται συγκρίνοντάς τα με τους μέσους όρους των βαρών από θετικά παραδείγματα της κάθε κατηγορίας χρησιμοποιώντας το μέτρο ομοιότητας Jaccard[27]:

$$\mathbf{T}(\mathrm{x},\mathrm{y}) = \frac{(x \cdot y)}{\|x\|^2 + \|y\|^2 - (x \cdot y)}$$

όπου (x · y) είναι το εσωτερικό γινόμενο των διανυσμάτων x και y.

Αν η μέτρηση υπερβεί μία ιδανική τιμή (threshold) το αντικείμενο κατηγοριοποιείται αναλόγως[5].

2.6 Suffix Tree Method

Η μέθοδος suffix trees που σημαίνει δέντρα προσφυμάτων(= γράμμα ή λέξη που προστίθεται στη ρίζα μιας λέξης και πριν από την κατάληξη) είναι μία τεχνική για αποθήκευση δεδομένων και γρήγορης αναζήτησης τα οποία χρησιμοποιούνται συχνά σε τομείς όπως υπολογιστική βιολογία για εφαρμογές όπως η αντιστοίχηση ακολουθιών που εφαρμόζεται σε ακολουθίες DNA. Για να εφαρμοστεί η μέθοδος suffix trees στην κατηγοριοποίηση κειμένου και κατ 'επέκταση email ακολουθείται η παρακάτω διαδικασία:

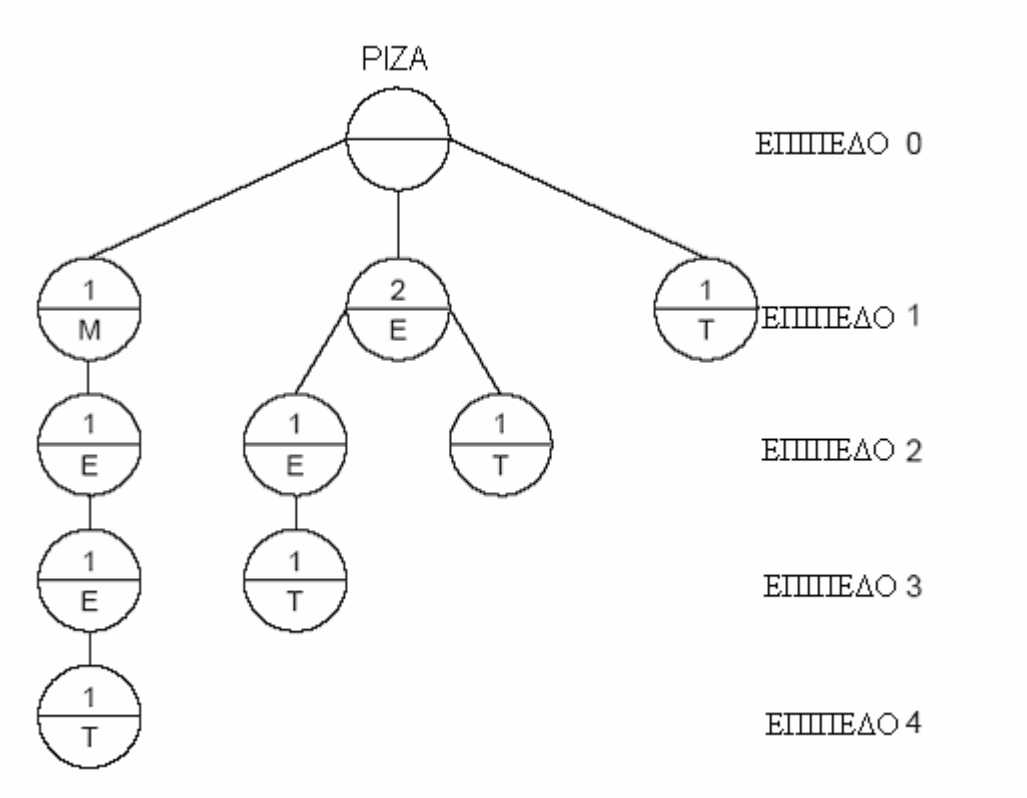
Παίρνουμε ένα σύνολο **D** με αρχεία (στην περίπτωσή μας email) τα οποία γνωρίζουμε ότι ανήκουν σε μία κατηγορία $\mathbf{C}_j$ ενός συνόλου κατηγοριών **C** και φτιάχνουμε ένα δέντρο για κάθε κατηγορία. Η μέθοδος βέβαια εφαρμόζεται και για πολλές κατηγορίες αλλά στην περίπτωση της κατηγοριοποίησης των emails οι κατηγορίες είναι δύο : spam και non spam. Καθένα από τα δέντρα λέμε ότι εκπροσωπεί μία κατηγορία γι 'αυτό και το κάθε δέντρο που δημιουργείται από μία κατηγορία ονομάζεται δέντρο κατηγορίας (class tree).

Όταν έχουμε ένα νέο αρχείο $\mathbf{d}_n$, το κατατάσσουμε λαμβάνοντας υπ'όψην καθένα από τα δέντρα κατηγοριών. Η κατηγορία με το μεγαλύτερο πλήθος αντιστοιχίσεων ακολουθιών καλείται και κατηγορία του αρχείου. Ωστόσο η μεγαλύτερη πρόκληση που εμφανίζεται είναι η ανάπτυξη μιας επαρκούς και κατάλληλης μεθόδου για τη για την αντιστοίχιση των ακολουθιών σε κάθε δέντρο κατηγορίας. Πώς όμως κατασκευάζονται αυτά τα δέντρα;

Ας υποθέσουμε ότι θέλουμε να κατασκευάσουμε ένα δέντρο προσφυμάτων από την ακολουθία **S**= "MEET".Η ακολουθία έχει τέσσερα προσφύματα: $\mathbf{S}_1$="MEET",$\mathbf{S}_2$ ="EET", $\mathbf{S}_3$ = "ET" and $\mathbf{S}_4$ = "T".

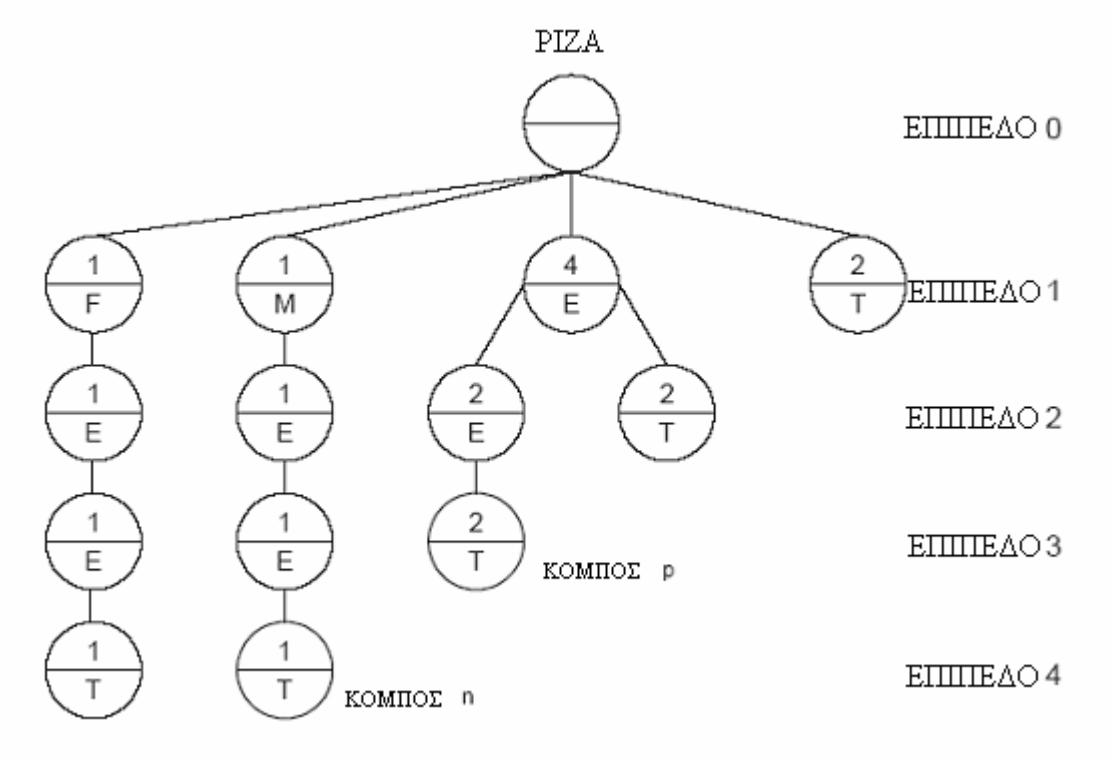
Ξεκινάμε από τη ρίζα του δέντρου και δημιουργούμε ένα θυγατρικό κόμπο για τον πρώτο χαρακτήρα του προσφύματος $\mathbf{S}_1$. Κατόπιν κατεβαίνουμε το δέντρο μέχρι τον νεοσύστατο κόμπο και δημιουργούμε ένα νέο θυγατρικό κόμπο για τον επόμενο χαρακτήρα στο πρόσφυμα και επαναλαμβάνουμε την διαδικασία για κάθε ένα χαρακτήρα στο πρόσφυμα. Σε κάθε κόμπο δημιουργούμε μόνο ένα θυγατρικό κόμπο αν κανένας από τους ήδη δημιουργημένους θυγατρικούς δεν αντιπροσωπεύει

το χαρακτήρα με τον οποίο ασχολούμαστε ως εκείνο το σημείο. Όταν εισάγουμε όλα τα προσφύματα το δέντρο που προκύπτει μοιάζει με αυτό του παρακάτω σχήματος.



<u>Σχήμα 2.6.1</u> ***Ένα δέντρο προσφυμάτων μετά το πέρας της εισαγωγής της ακολουθίας χαρακτήρων (string) "MEET".***

Κάθε κόμπος χαρακτηρίζεται από το χαρακτήρα τον οποίο αντιπροσωπεύει και τη συχνότητά του. Η θέση του κόμπου επίσης αντιπροσωπεύει τη θέση του χαρακτήρα στο πρόσφυμα έτσι ώστε να μπορούμε να έχουμε πολλούς κόμπους χαρακτηριζόμενους από τον ίδιο χαρακτήρα αλλά κάθε θυγατρικός κόμπος κάθε κόμπου (συμπεριλαμβανομένης και της ρίζας) θα μπορεί να «κουβαλάει» μία ετικέτα χαρακτήρα η οποία να είναι μοναδική ανάμεσα στους ομοίους του. Αν στη συνέχεια εισάγουμε το string $\mathbf{T}_1$ = "FEET" στο δέντρο του παραπάνω σχήματος τότε προκύπτει το επόμενο σχήμα.:



<u>Σχήμα 2.6.2</u> Ένα δέντρο προσφυμάτων μετά το πέρας της εισαγωγής των ακολουθιών χαρακτήρων (strings) "MEET" και "FEET".

Το νέο δέντρο είναι σχεδόν ίδιο σε δομή με το προηγούμενο επειδή τα προσφύματα των δύο string είναι ίδια αν εξαιρέσουμε το $\mathbf{T}_1$ = "FEET" αφού όπως είπαμε και προηγουμένως πρέπει να δημιουργούμε ένα νέο κόμπο μόνο όταν δεν υπάρχει ήδη. Στην περίπτωση που υπάρχει από πριν το μόνο που πρέπει να κάνουμε είναι να προσαυξήσουμε τη συχνότητα του.

Ωστόσο όσο συνεχίζουμε να εισάγουμε περισσότερα strings στο δέντρο, το πλήθος των κόμπων στο δέντρο αυξάνεται μόνο στην περίπτωση που το νέο string περιέχει substrings (= υπακολουθίες χαρακτήρων) που δεν έχουν συναντηθεί νωρίτερα. Αυτό έχει ως αποτέλεσμα το εξής.: Δεδομένου ένα σταθερό αλφάβητο και έναν περιορισμό στο μήκος των strings που λαμβάνουμε υπ'όψη υπάρχει και περιορισμός στο μέγεθος του δέντρου. Στην πράξη θα περίμενε κανείς ότι για τις περισσότερες κατηγορίες, όσο συνεχίζουμε να προσθέτουμε strings στο δέντρο κατηγορίας, το δέντρο θα αυξάνεται σε μέγεθος με φθίνοντα ρυθμό και πολύ πιθανόν σε κάποιο σημείο να σταθεροποιείται.

Για κάθε string **S** συμβολίζουμε τον i-οστό χαρακτήρα του **S** είτε με **S**[i], είτε με **S**i, το πρόσφυμα του **S** το οποίο αρχίζει από τον i-οστό χαρακτήρα με **S**(i) και το string από τον i-οστό έως τον j-οστό χαρακτήρα με **S**(i,j).

Κάθε κόμπος **n** χαρακτηρισμένος από ένα χαρακτήρα **c** είναι μοναδικά αναγνωρίσιμος από το μονοπάτι από τη ρίζα στο **n**. Για παράδειγμα ας πάρουμε το δέντρο στο δεύτερο σχήμα. Υπάρχουν πολλοί κόμποι που χαρακτηρίζονται με "Τ", αλλά μπορούμε να διακρίνουμε μεταξύ του κόμπου **n** = ("Τ" δεδομένου του "ΜΕΕ") = (Τ\ΜΕΕ) και **p** = ("Τ" δεδομένου του "ΕΕ") = (Τ\ΕΕ). Αυτοί οι κόμποι χαρακτηρίζονται **n** και **p** στο δεύτερο σχήμα. Θα λέμε ότι το μονοπάτι του **n** είναι $\vec{P}_n$ = "ΜΕΕ" και το μονοπάτι του **p** είναι $\vec{P}_p$ = "ΕΕ". Επιπλέον η συχνότητα του **n** είναι 1 όπου η συχνότητα του **p** είναι 2 και λέγοντας ότι το **n** έχει συχνότητα ίση με 1 είναι ισοδύναμο με το λέμε ότι η συχνότητα του "Τ" δεδομένου "ΜΕΕ" είναι 1 και ομοίως ισχύει και για το **p**.

Αν πούμε ότι ο κόμπος της ρίζας **r** είναι στο μηδενικό επίπεδο του δέντρου τότε όλοι οι θυγατρικοί κόμποι του **r** είναι στο πρώτο επίπεδο. Γενικεύοντας την παραπάνω πρόταση, μπορούμε να πούμε ότι το επίπεδο του εκάστοτε κόμπου στο δέντρο είναι ένα συν στο πλήθος των γραμμάτων στο μονοπάτι. Για παράδειγμα το επίπεδο του **n** είναι level(n)= 4 και level(p)= 3.

Το σύνολο των γραμμάτων που σχηματίζουν το πρώτο επίπεδο ενός δέντρου είναι το αλφάβητο που σημαίνει ότι όλοι οι κόμποι του δέντρου χαρακτηρίζονται με ένα από αυτά τα γράμματα. Για παράδειγμα αν εξετάσουμε ξανά το δέντρο στο δεύτερο σχήμα τα γράμματα του πρώτου του επιπέδου είναι το σύνολο Σ = {m,e,t,f} και όλοι οι κόμποι του δέντρου χαρακτηρίζονται με ένα από αυτά.

Έστω τώρα ότι ορίζουμε μία κατηγορία C, η οποία περιέχει δύο strings (τα οποία μπορούμε να τα θεωρήσουμε σαν αρχεία κειμένου π. χ. email), **S** = "ΜΕΕΤ" και **T** = "FEET". Τότε μπορούμε να αναφερόμαστε στο δέντρο του δεύτερου σχήματος ως το δέντρο κατηγορίας του **C** το οποίο συμβολίζουμε με **T** .

Το μέγεθος του δέντρου, | **T** | είναι το πλήθος των κόμπων που έχει όσα είναι δηλαδή και τα μοναδικά substrings του **C**. Για παράδειγμα στην περίπτωση του δεύτερου σχήματος έχουμε :

UC = uniqueSubstrings(C) = {meet, mee, me, m, eet, ee, e, et, t, feet, fee, fe, f}

|**UC**| = |uniqueSubstrings(C)| = 13

| **T** | = numberOfNodes(**T**) = 13

Το πλήθος εντούτοις όλων των εμφανίσεων των substrings χαρακτήρων του **C,** μπορεί να καλείται το πλήθος των δειγμάτων των substrings της κατηγορίας **C**:

$$\text{AC} = \text{allSubStrings(C)} = \begin{Bmatrix} meet, mee, me, m, eet, ee, e, et, e, t, \\ feet, fee, fe, f, eet, ee, e, et, e, t \end{Bmatrix}$$

Εδώ μπορεί κανείς να παρατηρήσει ότι τα τέσσερα "e" στο σύνολο είναι στην ουσία τα substrings χαρακτήρων **S**(1,1), **S**(2,2), **T**(1,1), **T**(2,2).

Επίσης όπως κάθε κόμπος στο δέντρο **T** αντιπροσωπεύει ένα string στο **UC** το μέγεθος της κατηγορίας **AC** είναι ίσο με το άθροισμα των συχνοτήτων των κόμπων στο δέντρο **T** .

$$|\mathbf{AC}| = |\text{allSubstrings(C)}| = \text{sumOfFrequencies}(\mathbf{T}) = 20.$$

Με ένα παρόμοιο τρόπο το δέντρο προσφυμάτων μας επιτρέπει να διαβάζει τις άλλες συχνότητες πολύ γρήγορα και εύκολα. Για παράδειγμα αν θέλουμε να μάθουμε το πλήθος των χαρακτήρων στην κατηγορία **C** μπορούμε να αθροίσουμε τις συχνότητες των κόμπων του πρώτου επιπέδου του δέντρου. Αν θέλουμε πάλι να μάθουμε το πλήθος των strings με μήκος δύο μπορούμε να αθροίσουμε τις συχνότητες των κόμπων του δεύτερου επιπέδου κ.ο.κ..

Αυτό επίσης μας επιτρέπει να υπολογίζουμε πολύ εύκολα τις πιθανότητες των strings οποιουδήποτε μήκους (μέχρι βέβαια το βάθος του δέντρου) ή οποιουδήποτε κόμπου στο δέντρο. Για παράδειγμα μπορούμε να πούμε με βάση το δέντρο στο δεύτερο σχήμα ότι η πιθανότητα ενός substring **u** μήκους δύο με τιμή **u** = "ee", δεδομένης της κατηγορίας **C** είναι η συχνότητα του κόμπου **n** = (E|E) διαιρούμενο με το άθροισμα των συχνοτήτων όλων των κόμπων επιπέδου δύο στο δέντρο **T** :

$$\textbf{estimatedProbabilityOfString(u)} = \hat{p}_s(s_u) = \frac{f(u)}{\sum_{i \in N_u} f(i)}$$

όπου $\mathbf{N}_u$ είναι το σύνολο όλων των κόμπων στο ίδιο επίπεδο με το **u**.

Ομοίως μπορεί κανείς να υπολογίσει τη δεσμευμένη πιθανότητα του **u** ως τη συχνότητα του **u** διαιρούμενη με το άθροισμα των συχνοτήτων όλων των θυγατρικών του μητρικού του **u** :

$$\textbf{estimatedProbabilityOfChar(u)} = \hat{p}_c(c_u) = \frac{f(u)}{\sum_{i \in n_u} f(i)}$$

όπου $\mathbf{n_u}$ είναι το σύνολο όλων των θυγατρικών του μητρικού του **u**.

Όπως είπαμε και νωρίτερα στο θέμα που μας αφορά σε αυτήν εδώ την εργασία είναι η κατηγοριοποίηση των emails οπότε έχουμε μόνο δύο κατηγορίες spam και non spam.Έτσι λοιπόν υπολογίζουμε τις αντιστοιχίσεις των strings στις κατηγορίες non spam(ham) και spam και παίρνουμε το λόγο hsr = hamscore/spamscore. Αν τώρα αυτός ο λόγος είναι πάνω από μία συγκεκριμένη τιμή το λεγόμενο threshold δηλαδή το email κατηγοριοποιείται ως ham και αν είναι κάτω από αυτήν την τιμή ως spam[7].

3. ΤΟ ΠΕΙΡΑΜΑ

Αφού μιλήσαμε γενικότερα για όλους τους αλγόριθμους ας περάσουμε τώρα στο πείραμά μας. Σε αυτό το κεφάλαιο θα επεξηγήσουμε με λεπτομέρειες την διαδικασία που ακολουθήθηκε προκειμένου να βγουν τα αποτελέσματά μας και με ποιο τρόπο τα απεικονίζουμε.

Όλα τα πειράματά μας έγιναν πάνω στο σύνολο spam και ham mails που ονομάζεται Ling-Spam corpus. Το corpus αυτό έχει προέλθει από ένα forum συζητήσεων σχετικά με θέματα γλωσσολογικά γιατί έτσι μόνο θα μπορούσαμε να βρούμε μεγάλο αριθμό ham emails χωρίς να υπάρξει πρόβλημα προσβολής του ιδιωτικού απορρήτου καθώς τα legitimate emails θεωρούνται προσωπικά δεδομένα. Στην περίπτωση των Ling-mails δεν υπάρχει τέτοιο θέμα αφού είναι δημοσιοποιημένα εξ' αρχής. Το corpus αυτό αποτελείται από :

- 2412 μηνύματα γλωσσομάθειας (Linguist messages) που περιέχουν μόνο το θέμα και το κυρίως σώμα του μηνύματος και

- 481 διαφημιστικά μηνύματα από την ίδια πηγή που πάλι περιέχουν μόνο το θέμα και το κυρίως σώμα του μηνύματος.

Τα spam αποτελούν το 16.6% του συνολικού corpus ποσοστό κοντινό σε διάφορες υλοποιήσεις παρόμοιων πειραμάτων.

Κάναμε δοκιμές με μήκος ν-γραμμάτων 2,3,4 και 5 με τον αλγόριθμο SVM και αντιπαραθέτουμε τα αποτελέσματά μας με τις δοκιμές της εργασίας του κυρίου Ίωνα Ανδρουτσόπουλου : An Evaluation of Naive Bayesian Anti-Spam Filtering [1] που έγιναν με τον αλγόριθμο Bayes όπου όμως είχαν χρησιμοποιηθεί λέξεις. Οι δικές μας μετρήσεις έγιναν στο corpus όπως είναι στην αρχική του μορφή (Bare) χωρίς να έχει περάσει δηλαδή από τα στάδια Lemmatizer και Stop Word List γιατί έτσι θα χάναμε ίσως πολύτιμη πληροφορία.

3.1 Πλήθος Χαρακτηριστικών

Στις μέχρι τώρα εργασίες ως χαρακτηριστικά έχουν χρησιμοποιηθεί λέξεις ή ομάδες λέξεων (φράσεις). Η προσέγγιση με ν-γράμματα δεν είχε ακολουθηθεί μέχρι τώρα λόγω του ότι θεωρείται πολύ ασύμφορη.

Πράγματι με την χρήση λέξεων ο αριθμός των χαρακτηριστικών κυμαίνεται σε χαμηλά επίπεδα αφού ακόμα και με πλήθος 300 ή 600 χαρακτηριστικών τα αποτελέσματα είναι ικανοποιητικά. Τα ν-γράμματα όμως είναι σαφώς περισσότερα και έτσι έπρεπε να φτάσουμε σε όσο μεγαλύτερο αριθμό μπορούσαμε. Το μηχάνημα στο οποίο δουλέψαμε ήταν ένας AMD AthlonXP 3000+ με 1024 MB RAM και με λειτουργικό τα Windows XP Professional. Με τον εξοπλισμό που είχαμε στην διάθεσή μας μπορέσαμε να φτάσουμε μέχρι τον «μικρό» αριθμό των 6000 περίπου ν-γραμμάτων. Οι έρευνες όμως σχετικά με τον αλγόριθμο SVM έχουν δείξει πως λειτουργεί καλά με όσο το δυνατόν περισσότερα ή ακόμα και όλα τα διαθέσιμα στοιχεία που μπορούμε να εξάγουμε από το σύνολο των δειγμάτων μας. Αυτό όμως είναι υπολογιστικά πολύ ασύμφορο έως αδύνατο. Και θα εξηγήσουμε γιατί.

Τα διαφορετικά ν-γράμματα που μπορεί να εμφανιστούν υπολογίζονται από τον τύπο των διατάξεων με επανατοποθέτηση. Εάν λοιπόν N είναι ο αριθμός των πιθανών χαρακτήρων και k το μήκος του ν-γράμματος το πλήθος τους υπολογίζεται από το τύπο N^k. Υπολογίσαμε τον συνολικό αριθμό χαρακτήρων μας N στους 66

(26 γράμματα, 10 νούμερα και 30 σημεία στίξης και άλλοι χαρακτήρες) Έχουμε λοιπόν :

Πλήθος συνδυασμών ανά μήκος ν-γραμμάτων

Μήκος (k)	**2**	**3**	**4**	**5**
Συνολικοί συνδυασμοί	4.356	287.496	18.974.736	1.252.332.576

Πίνακας 3.1.1

Το πλήθος των επιλεγμένων ν-γραμμάτων στις δοκιμές μας σε κάθε κατηγορία είναι μικρό σε σχέση με τους συνολικούς συνδυασμούς που μας δίνει ο τύπος των διατάξεων. Συγκεκριμένα στα 2-γράμματα πήραμε τα 1322 πιο συχνά εμφανιζόμενα, στα 3-γράμματα τα 3990, στα 4-γράμματα τα 3999 και στα 5-γράμματα τα 5978 πιο συχνά εμφανιζόμενα. Προφανώς με την πρώτη ματιά τα δείγματά μας φαίνονται να είναι πολύ μικρά. Στην πραγματικότητα όμως στην φυσική γλώσσα δεν συναντώνται όλοι αυτοί οι διαφορετικοί συνδυασμοί. Υπολογίσαμε λοιπόν το πλήθος όλων των διαφορετικών ν-γραμμάτων για ν = 2 και ν = 3 που περιέχει το dataset για να ερευνήσουμε πόσο απέχουν τα δείγματά μας από αυτό. Βρήκαμε πως υπάρχουν 1478 2-γράμματα σε όλο το dataset δηλαδή το 33.93% του αριθμού των δυνατών συνδυασμών και 20270 διαφορετικά 3-γράμματα, δηλαδή το 7.05% των αντίστοιχων δυνατών συνδυασμών. Ακόμα και έτσι όμως για να επεξεργαστούμε όλα τα 3-γράμματα θα χρειαζόμασταν ένα πολύ ισχυρό μηχάνημα με μεγάλα ποσά μνήμης πράγμα πολύ δύσκολο να υλοποιηθεί ακόμα και για επιχειρήσεις πόσο μάλλον από ιδιώτες. Άρα θεωρήθηκε ως μη εφαρμόσιμη λύση.

Βλέπουμε λοιπόν ότι τα 2-γράμματα που επιλέξαμε αποτελούν το 89.44% του πραγματικού πλήθους 2-γραμμάτων και τα 3-γράμματα το 19.68% του αντίστοιχου πλήθους 3-γραμμάτων. Το ποσοστό αυτό θα συνεχίζει να πέφτει όσο αυξάνεται το μήκος των ν-γραμμάτων, αλλά παρατηρήσαμε πως το ποσοστό επιτυχίας μας δεν φθίνει. Αντιθέτως μεγαλώνει. Άρα μπορούμε να υποθέσουμε πως ο «χαμένος» αριθμός των ν-γραμμάτων που δεν παίρνουμε υπ' όψιν μας δεν επηρεάζουν σοβαρά την ακρίβεια των μετρήσεων και ίσως μάλιστα να την μείωναν δυσκολεύοντας την σωστή χαρτογράφηση των δειγμάτων[2][6].

3.2 Πρόβλημα Κατηγοριοποίησης

Στα πειράματα σχετικά με την αναγνώριση spam και ham υπάρχουν δύο ειδών λάθη. Να χαρακτηριστεί ένα spam ως ham και θα το συμβολίζουμε S → L (Spam → Legitimate) και να χαρακτηριστεί ένα ham ως spam που θα το συμβολίζουμε L → S (Legitimate → Spam). Γενικότερα θεωρείται πολύ πιο σοβαρό λάθος να χαρακτηριστεί ένα ham ως spam και γι' αυτό θα θέσουμε ένα βάρος λ που θα χαρακτηρίζει την σοβαρότητα του λάθους. Έτσι το λάθος L → S είναι λ φορές πιο σοβαρό από το S → L δηλαδή κάθε ham που μπλοκάρεται από το φίλτρο μας θα αντιστοιχεί σε λ spam που πέρασαν το φίλτρο. Οι τιμές του λ θα είναι : λ = 1, λ = 9 και λ = 999 σε συμφωνία με την αντίστοιχη εργασία του κυρίου Ανδρουτσόπουλου. Οι διαφορετικές τιμές του λ αντιπροσωπεύουν τρία διαφορετικά σενάρια :

λ = 999 Το χαρακτηρισθέν ως spam email σβήνεται μόνιμα από το mailbox χωρίς την συμμετοχή κανενός χρήστη.

λ = 9 Το χαρακτηρισθέν ως spam email μπλοκάρεται και στέλνεται μια αίτηση στον αποστολέα του να ξαναπροσπαθήσει να το στείλει με διαφορετική μορφή.

λ = 1 Το χαρακτηρισθέν ως spam email εμφανίζεται στο Inbox του παραλήπτη με την σημείωση πως είναι πιθανόν spam και ακολούθως ενημερώνεται και ο αποστολέας.

Εάν λοιπόν συμβολίσουμε ως $n_{L\to S}$ και $n_{S\to L}$ τους αριθμούς των L → S και S → L λαθών και ως $n_{L\to L}$ και $n_{S\to S}$ τους αριθμούς των σωστά εξακριβωμένων ham και spam μηνυμάτων τότε οι τιμές Spam Recall (SR) και Spam Precision (SP) είναι :

$$SR = \frac{n_{S\to S}}{n_{S\to S} + n_{S\to L}} \qquad SP = \frac{n_{S\to S}}{n_{S\to S} + n_{L\to S}}$$

Δηλαδή ως Spam Recall αναφέρεται η αναλογία των σωστά αναγνωρισμένων spam προς το συνολικό αριθμό spam και ως Spam Precision η αναλογία των σωστά

αναγνωρισμένων spam προς τον αριθμό των χαρακτηρισμένων ως spam μηνυμάτων. Εμείς την αναλογία αυτή θα την εκφράζουμε σαν ποσοστό επί τοις εκατό στους πίνακές μας.

Στις εργασίες ταξινόμησης δύο συχνά χρησιμοποιούμενες τιμές αξιολόγησης των αποτελεσμάτων είναι η ακρίβεια και ο ρυθμός σφάλματος ή αλλιώς Accuracy (Acc) και Error rate (Err = 1 - Acc) αντιστοίχως που δίνονται από τους τύπους :

$$Acc = \frac{n_{L\to L} + n_{S\to S}}{N_L + N_S} \qquad Err = \frac{n_{L\to S} + n_{S\to L}}{N_L + N_S}$$

Όπου N_L και N_S οι αντίστοιχοι αριθμοί των ham και spam mails του corpus.

Εμείς όμως δεν παίρνουμε με το ίδιο βάρος τις επιτυχίες και τις αποτυχίες του φίλτρου μας συνεπώς πρέπει να συνυπολογίσουμε και το βάρος λ στους παραπάνω τύπους και έτσι έχουμε τα Weighted Accuracy και Weighted Error Rate :

$$WAcc = \frac{\lambda \cdot n_{L\to L} + n_{S\to S}}{\lambda \cdot N_L + N_S} \qquad WErr = \frac{\lambda \cdot n_{L\to S} + n_{S\to L}}{\lambda \cdot N_L + N_S}$$

Πρέπει όμως για να πάρουμε μια σωστή εικόνα των αποτελεσμάτων μας με βάση αυτούς τους τύπους να συγκρίνουμε τις τιμές τους με μια «βάση» ας το πούμε έτσι ώστε να μην μπερδευόμαστε με πολύ ψηλές τιμές Accuracy ή πολύ χαμηλές τιμές Error rate. Θα υποθέσουμε λοιπόν πως μετράμε τις δύο αυτές τιμές χωρίς την ύπαρξη φίλτρου οπότε όλα τα ham θα περάσουν στο γραμματοκιβώτιό μας (σωστά) όπως επίσης και όλα τα spam (λάθος). Οι τιμές αυτές θα είναι :

$$WAcc^b = \frac{\lambda \cdot N_L}{\lambda \cdot N_L + N_S} \qquad WErr^b = \frac{N_S}{\lambda \cdot N_L + N_S}$$

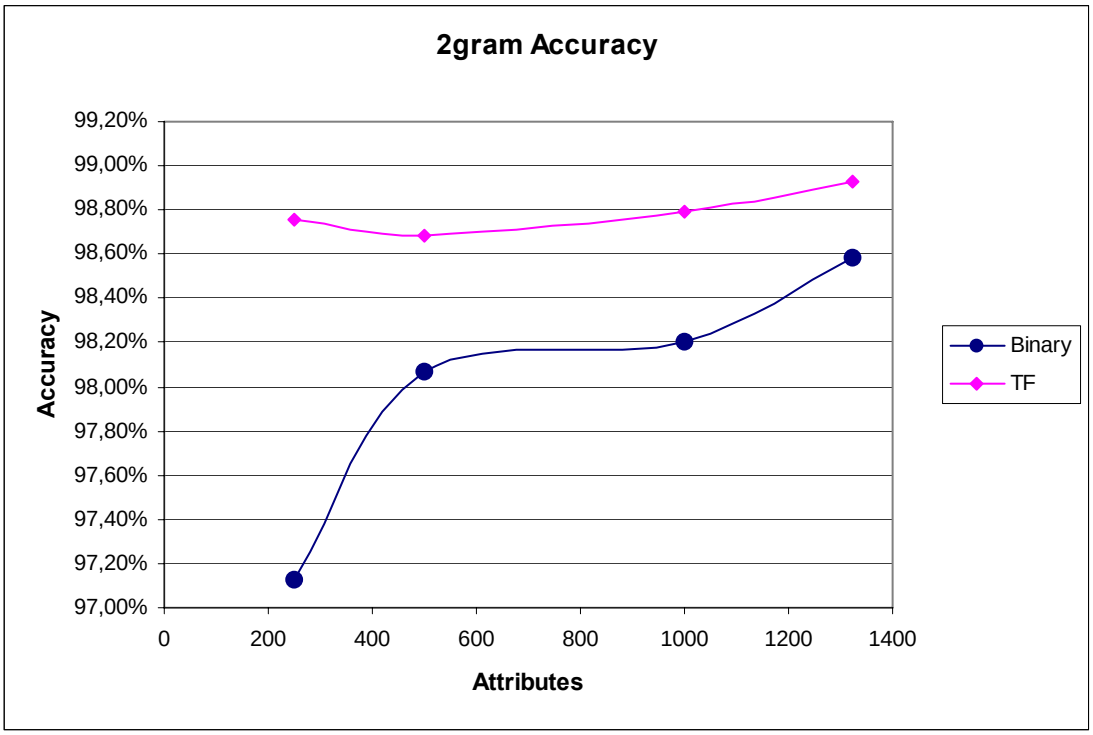
Έτσι λοιπόν εισάγουμε την έννοια Συνολική Αναλογία Κόστους ή Total Cost Ratio (TCR) με τύπο :

$$TCR = \frac{WErr^b}{WErr} = \frac{N_S}{\lambda \cdot n_{L \to S} + n_{S \to L}}$$

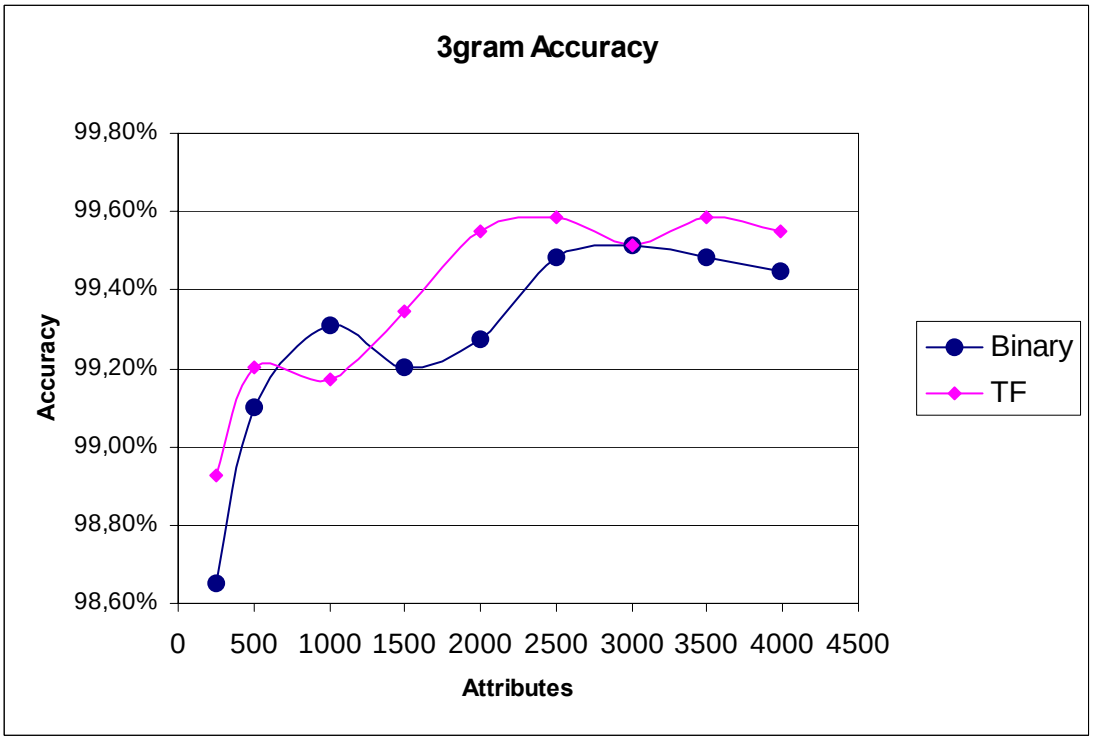
Μεγαλύτερες τιμές του TCR σημαίνουν καλύτερη απόδοση του φίλτρου και όταν TCR < 1 το να μην χρησιμοποιείς το φίλτρο είναι καλύτερα. Εάν το TCR είναι ανάλογο του χρόνου τότε μετράει πόση ώρα χρειάζεται για να διαγραφούν χειροκίνητα όλα τα spam mails που θα έρθουν χωρίς φίλτρο (N_S) συγκρινόμενο με τον χρόνο που χρειάζεται για να διαγραφούν τα spam που θα περάσουν το φίλτρο ($n_{S \to L}$) συν την ώρα που θα χρειαστεί να ξεμπλοκαριστούν τα κακώς φραγμένα ham mails ($\lambda \cdot n_{L \to S}$).

Ας δούμε όμως αρχικά τα γενικότερα αποτελέσματα των μετρήσεών μας σε απόλυτο ποσοστό επιτυχίας δηλαδή το ποσοστό επιτυχίας αναγνώρισης συνολικά των ham και spam emails. Ξεκινάμε από τον μέγιστο αριθμό ν-γραμμάτων (attributes) που μας επέστρεψε το πρόγραμμά μας και είναι τα ν-γράμματα με το μεγαλύτερο ποσοστό εμφάνισης. Ο αριθμός των χαρακτηριστικών (attributes) μειώνεται με βήμα 500 με την βοήθεια του φίλτρου Infogain. Το Infogain είναι ένα φίλτρο αξιολόγησης των attributes που παίρνει υπ' όψιν του την συνολική συμμετοχή του καθενός στην κατηγοριοποίηση των δειγμάτων και τα ταξινομεί με φθίνουσα σειρά σημαντικότητας αλλά όχι απαραίτητα συχνότητας εμφάνισης. Αν δηλαδή ένα ν-γραμμα εμφανιστεί λίγες σχετικά φορές συνολικά αλλά εμφανιστεί μόνο σε spam email θεωρείται σημαντικότερο στοιχείο για την κατηγοριοποίηση από κάποιο άλλο που θα εμφανιστεί περισσότερες φορές αλλά θα έχει ισάξια εμφάνιση και στις δύο κατηγορίες και συνεπώς δεν θα βοηθήσει το έργο της κατηγοριοποίησης.

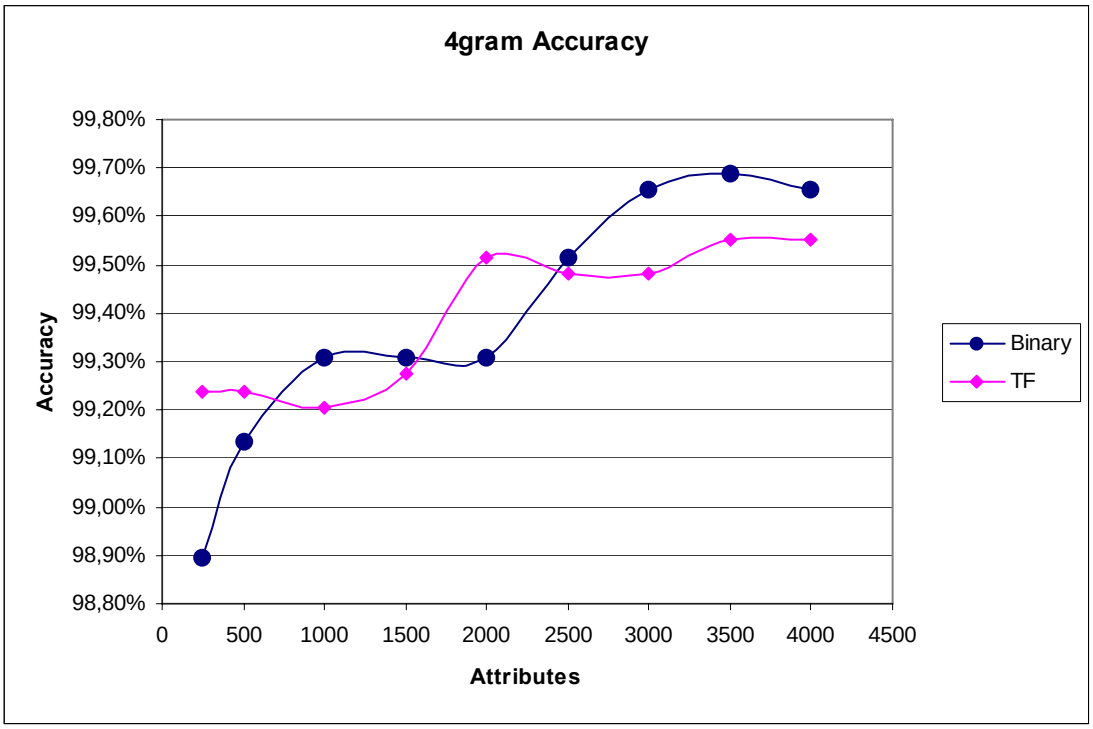
Αρχικά θα κοιτάξουμε την γενικότερη αποτελεσματικότητα του φίλτρου μας απεικονίζοντας την ακρίβεια του φίλτρου (δηλαδή επιτυχίες σε spam και ham μαζί). Τα διαγράμματα με το Accuracy ανά ν-γραμμα και σε συνάρτηση με τα επιλεγμένα attributes και την μέθοδο εκπροσώπησης είναι :



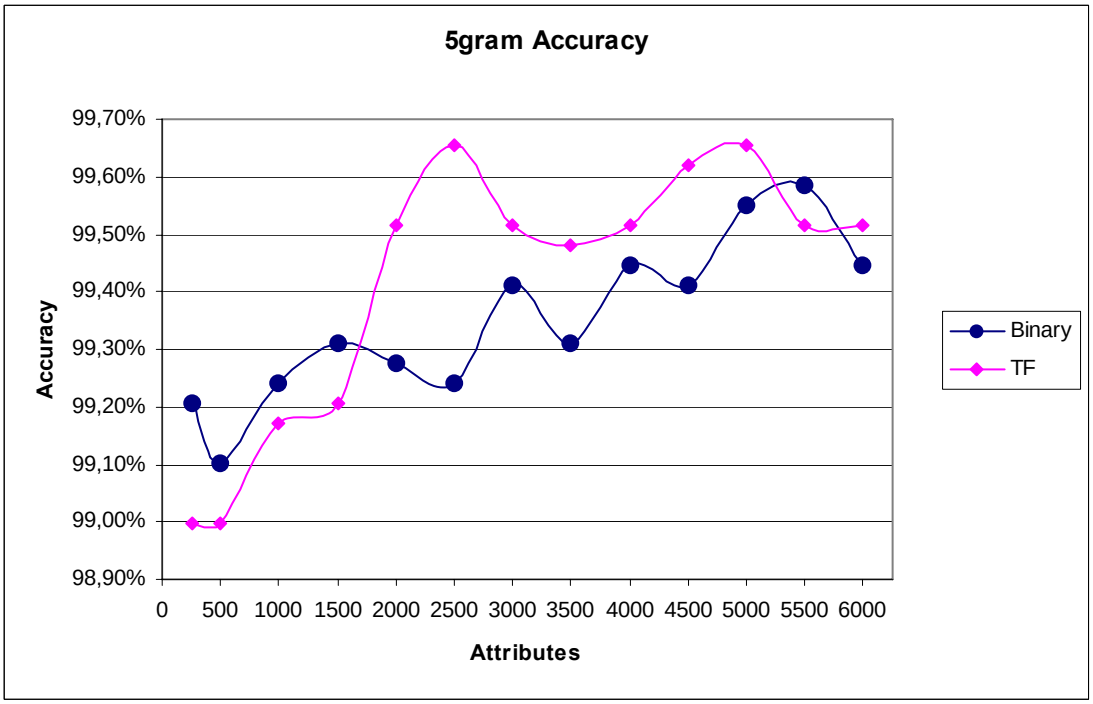
Διάγραμμα 3.2.1



Διάγραμμα 3.2.2



Διάγραμμα 3.2.3



Διάγραμμα 3.2.4

Με την πρώτη ματιά βλέπουμε πως τα αποτελέσματα που μας έδωσαν τα 2-γράμματα είναι απογοητευτικά σε σχέση με τα υπόλοιπα ν-γράμματα οπότε θα τα αποκλείσουμε από την περαιτέρω εξέτασή μας. Στα υπόλοιπα παρατηρούμε πως σε άλλες περιπτώσεις υπερτερεί η μέθοδος εκπροσώπησης των όρων με βάση την

συχνότητα εμφάνισής τους από την αντίστοιχη δυαδική και σε άλλες περιπτώσεις το αντίθετο. Συγκεκριμένα παρατηρούμε τις παρακάτω τιμές ως πιο επιτυχείς ανά ν-γραμμα :

N-gram Length	Attributes	Success
3-gram Binary	3000	99.5161%
3-gram TF	**2500**	**99.5852%**
4-gram Binary	**3500**	**99.6889%**
4-gram TF	3500	99.5506%
5-gram Binary	5500	99.5853%
5-gram TF	**2500**	**99.6543%**

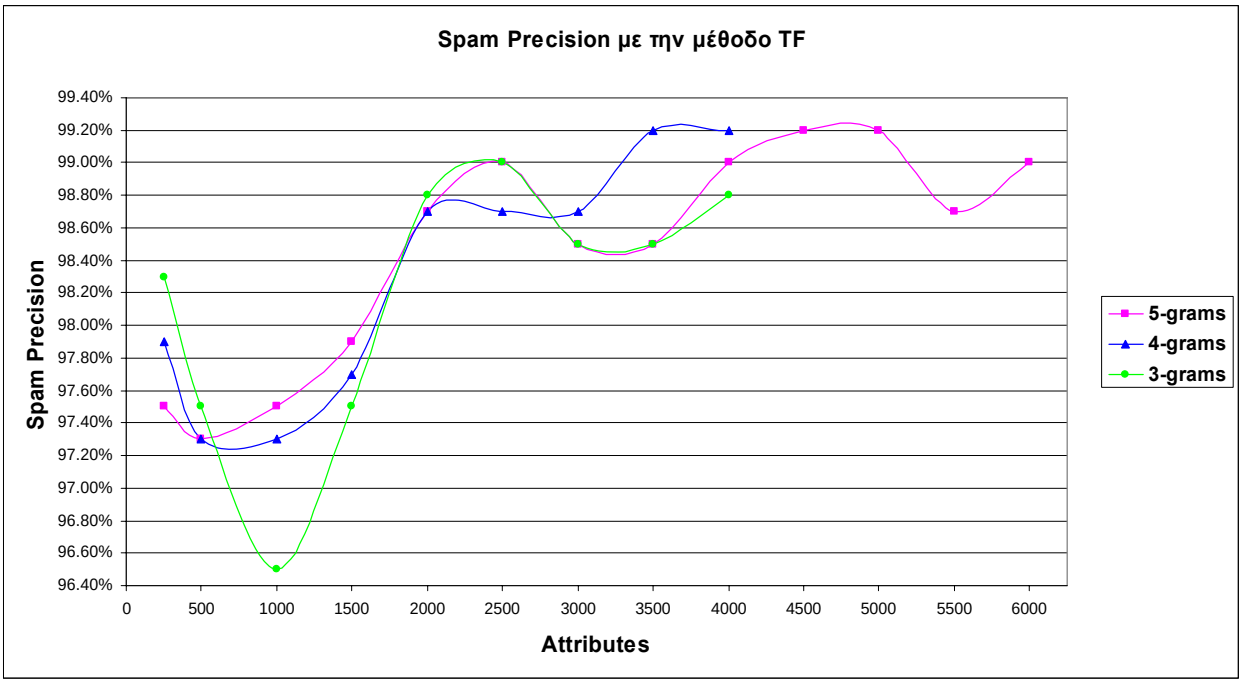
Πίνακας 3.2.1

Οι «νικητές» κάθε κατηγορίας είναι τα 3-γράμματα σε εκπροσώπηση συχνότητας στα 2500 στοιχεία, τα 4-γράμματα σε δυαδική εκπροσώπηση στα 3500 στοιχεία και τα 5-γράμματα σε εκπροσώπηση συχνότητας εμφάνισης στα 2500 στοιχεία. Συνολικά την καλύτερη την έχουν τα 4-γράμματα με μόλις 2 ham emails χαρακτηρισμένα ως spam (σε σύνολο 2412 ham emails) και 7 spam emails χαρακτηρισμένα ως ham (σε σύνολο 481 spam emails). Αν και δεν φτάσαμε ποτέ στο 100% επιτυχία στην αναγνώριση των ham (που είναι και το σημαντικότερο τελικά), πιστεύουμε πως το ποσοστό λάθους μας είναι μηδαμινό (μόλις 0.082%). Παρόλα αυτά δεν έχουμε βγάλει ακόμα κάποια χρήσιμα συμπεράσματα σχετικά με την σωστή μέθοδο εκπροσώπησης, ούτε τον ακριβή αριθμό των ν-γραμμάτων που πρέπει να επιλέξουμε ώστε να βγάζουμε σωστά αποτελέσματα. Το μόνο που μπορούμε να παρατηρήσουμε στους πίνακές μας είναι πως τα ποσοστά μας παρουσιάζουν απότομες και συχνά μεγάλες πτώσεις μόλις το πλήθος των attributes πέσει κάτω από το πλήθος των emails κατά ένα ποσοστό. Έτσι εξηγείται και η χαμηλή απόδοση των 2-γραμμάτων που ακόμα και ο συνολικός τους αριθμός είναι ο μισός του αριθμού των email.

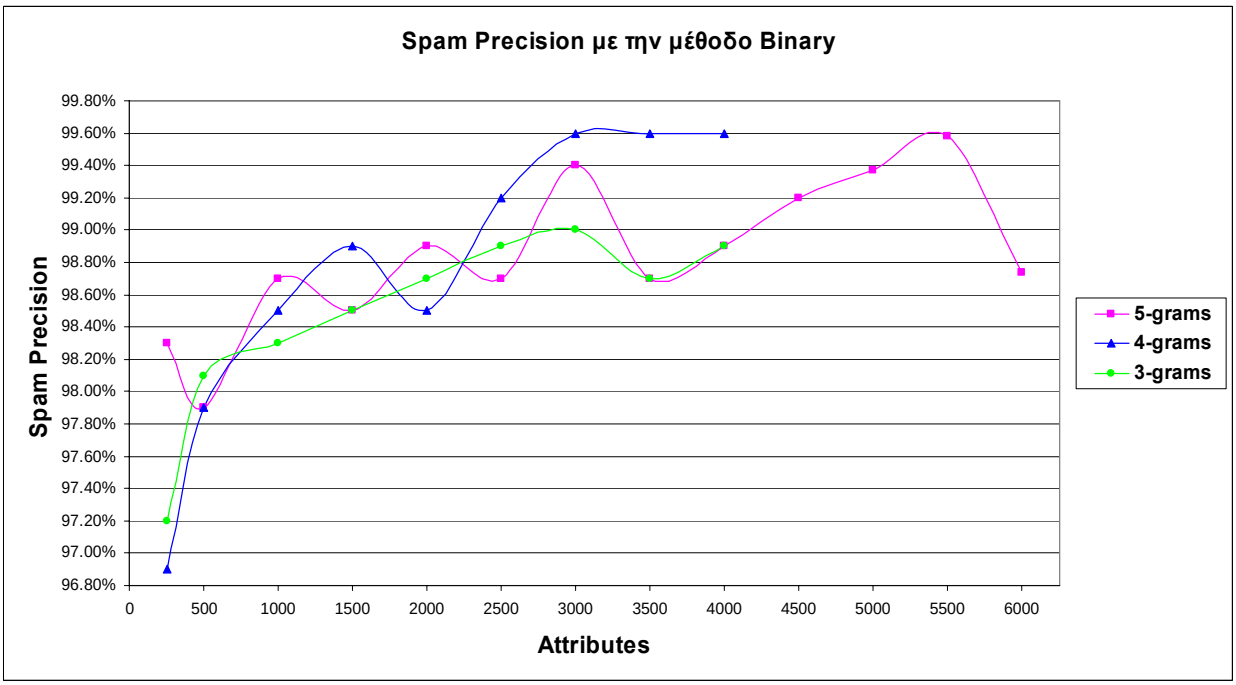
3.3 Spam Recall – Precision

Στη συνέχεια θα εξετάσουμε την αποτελεσματικότητα του φίλτρου μας με τη βοήθεια της έννοιας Spam Precision. Τα διαγράμματα απεικόνισης του spam

precision συναρτήσει του πλήθους των ν-γραμμάτων αλλά και των μεθόδων TF και Binary είναι οι παρακάτω:



Διάγραμμα 3.3.1

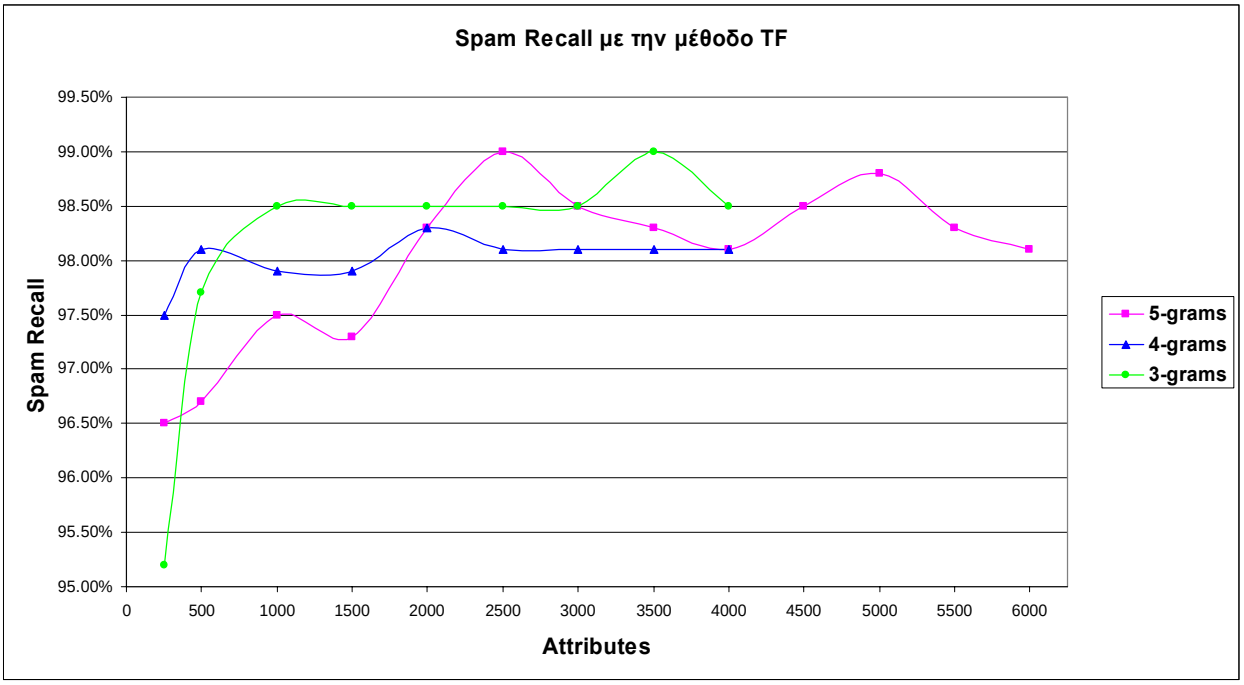


Διάγραμμα 3.3.2

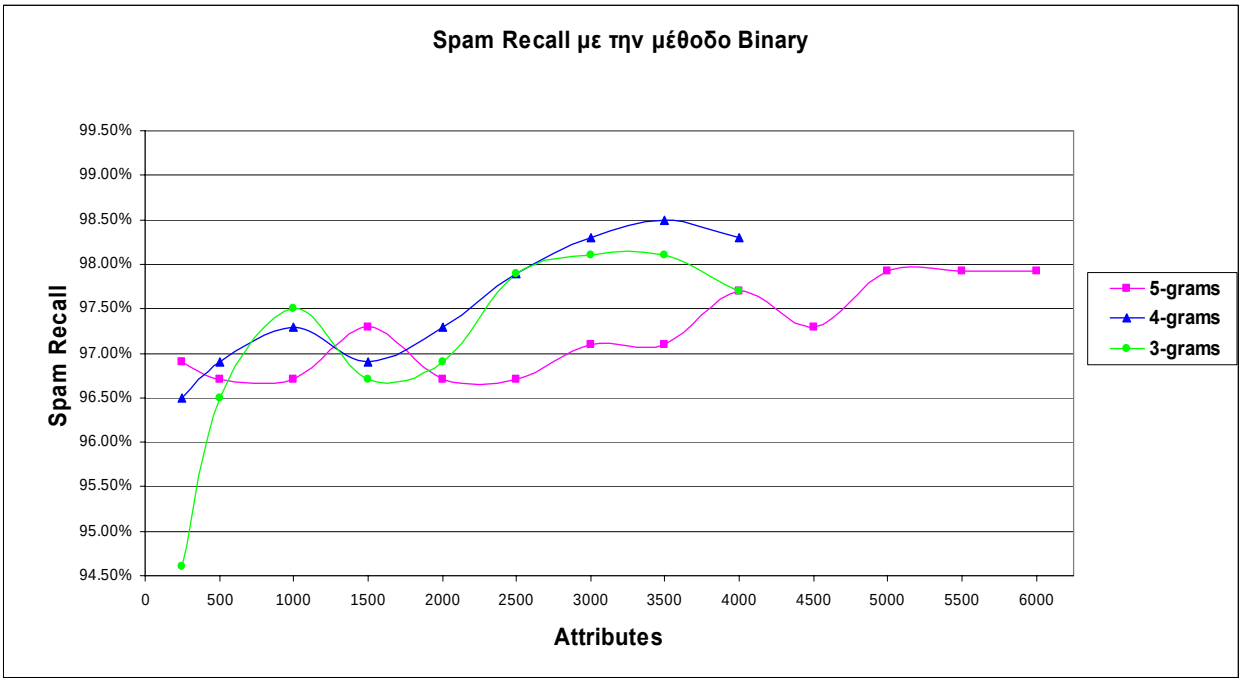
Η πρώτη παρατήρηση που μπορεί να κάνει κανείς είναι ότι το spam precision και με τις δύο μεθόδους κυμαίνεται σε πολύ ψηλά επίπεδα όταν τα attributes είναι

αρκετά δηλαδή περισσότερα από τον αριθμό των emails. Επίσης βλέπουμε πως τις καλύτερες αποδόσεις τις πετυχαίνουμε με την μέθοδο Binary, η οποία έχει μικρότερες αλλά περισσότερες μεταβολές από την μέθοδο TF και μάλιστα φαίνεται να σταθεροποιείται στο μεγάλο πλήθος των ν-γραμμάτων τουλάχιστον στα 4-γράμματα και τα 5-γράμματα. Στα 3-γράμματα εδώ έχει ελάχιστες μεταβολές ενώ σε όλα τα ν-γράμματα πέφτει δραματικά όταν το πλήθος των ν-γραμμάτων είναι πολύ μικρό, προφανώς για τους ίδιους λόγους για τους οποίους παρουσιάζει πτώση και η απόλυτη επιτυχία που ερευνήσαμε παραπάνω. Με τη μέθοδο TF τέλος το spam precision φτάνει στη χαμηλότερη τιμή που βρήκαμε στις δοκιμές και συγκεκριμένα στα 3-γράμματα.

Για να πάρουμε μια πιο ολοκληρωμένη εικόνα ας εξετάσουμε και τις επιδόσεις του φίλτρου και στο Spam Recall :



Διάγραμμα 3.3.3



Διάγραμμα 3.3.4

Τώρα λοιπόν βλέπουμε πως έχουμε αρκετά καλές επιδόσεις και στο Spam Recall που σε συνδυασμό με τις επιδόσεις του Spam Precision μας δείχνουν πως εκτός από καλό ποσοστό σωστά αναγνωρισμένων spam emails σε σχέση με τα συνολικά spam-αναγνωρισμένα έχουμε και καλό ποσοστό και σε σχέση με τα πραγματικά spam. Πιο απλά μπορούμε και πετυχαίνουμε χαμηλά ποσοστά λανθασμένης αναγνώρισης email και για τις δύο κατηγορίες. Εντύπωση μας κάνει η μεγάλη σταθερότητα των 3-γραμμάτων στον δείκτη Spam Recall στο TF από τα 4000 attributes μέχρι και τα 1000 με μοναδική εξαίρεση τα 3500 όπου πετυχαίνει και τη μέγιστη απόδοση ανάμεσα στα ν-γράμματα. Αυτό σε συνδυασμό με τις χαμηλές επιδόσεις των 3-γραμμάτων στον δείκτη Spam Precision δηλώνει πως τα 3-γράμματα είναι πιο επιρρεπή στο L → S σφάλμα όσο μειώνονται τα attributes και γι' αυτό μειώνεται δραματικά και η συνολική τους απόδοση.

3.4 Weighted Accuracy – Total Cost Ratio

Παρακάτω θα παραθέσουμε και με αριθμούς τις καλύτερες τιμές που πετύχαμε σε συνάρτηση με το βάρος λ και θα τις συγκρίνουμε με τις αντίστοιχες του κυρίου Ανδρουτσόπουλου. Στη συνέχεια θα εξετάσουμε και γραφικά τα

αποτελέσματα για να έχουμε μια καλύτερη εικόνα. Να υπενθυμίσουμε πως τα αποτελέσματα όσον αφορά τον αλγόριθμο του Bayes έχουν γίνει χρησιμοποιώντας ως attributes λέξεις οπότε είναι φυσικό να έχει καλές επιδόσεις σε μικρά πλήθη σε αντίθεση με τα ν-γράμματα που χρησιμοποιήσαμε εμείς.

Filter Configuration	λ	No. of Attrib.	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W. Acc.	TCR
(a)Bare	1	50	81.10%	96.85%	96.408%	83.374%	4.63
(b)Stop-List	1	50	82.35%	97.13%	96.649%	83.374%	4.96
(c)Lemmatizer	1	100	82.35%	99.02%	96.926%	83.374%	5.41
(d)Lemm+Stop list	1	100	82.78%	99.49%	97.064%	83.374%	5.66
(a)Bare	9	200	76.94%	99.46%	99.419%	97.832%	3.73
(b)Stop-List	9	200	76.11%	99.47%	99.401%	97.832%	3.62
(c)Lemmatizer	9	100	77.57%	99.45%	99.432%	97.832%	3.82
(d)Lemm+Stop list	9	100	78.41%	99.47%	99.450%	97.832%	3.94
(a)Bare	999	200	73.82%	99.43%	99.912%	99.980%	0.23
(b)Stop-List	999	200	73.40%	99.43%	99.912%	99.980%	0.23
(c)Lemmatizer	999	300	63.67%	100%	99.993%	99.980%	2.86
(d)Lemm+Stop list	999	300	63.05%	100%	99.993%	99.980%	2.86

Πίνακας 3.4.1

Τα παραπάνω αποτελέσματα έχουν γίνει στο ίδιο corpus χρησιμοποιώντας την δυαδική μόνο εκπροσώπηση και εφαρμόζοντας τον αλγόριθμο του Bayes. Παρακάτω βλέπετε τα αποτελέσματα μας ανάλυσής μας με τον αλγόριθμο του SVM (Support Vector Machines) όπου παρουσιάζουμε τα καλύτερα αποτελέσματά μας ανά ν-γραμμα και βάρος λ. Οι αναλύσεις μας έγιναν μόνο στο Bare δείγμα του corpus χωρίς την χρήση περαιτέρω φίλτρων. Για πιο αναλυτικά αποτελέσματα δείτε το Παράρτημα Β’ στο τέλος μας εργασίας μας.

5-γράμματα (TF)

Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5000	1	98.80%	99.20%	99.65%	83.37%	48.10
4500	1	98.50%	99.20%	99.62%	83.37%	43.73
2500	1	99.00%	99.00%	99.65%	83.37%	48.10
5000	9	98.80%	99.20%	99.81%	97.83%	11.45
4500	9	98.50%	99.20%	99.81%	97.83%	11.18
6000	999	98.10%	99.00%	99.79%	99.98%	0.10
5000	999	98.80%	99.20%	99.83%	99.98%	0.12
4500	999	98.50%	99.20%	99.83%	99.98%	0.12
2500	999	99.00%	99.00%	99.79%	99.98%	0.10

Πίνακας 3.4.2

4-γράμματα (TF)

Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4000	1	98.10%	99.20%	99.55%	83.37%	37.00
3500	1	98.10%	99.20%	99.55%	83.37%	37.00
4000	9	98.10%	99.20%	99.80%	97.83%	10.69
3500	9	98.10%	99.20%	99.80%	97.83%	10.69
4000	999	98.10%	99.20%	99.83%	99.98%	0.12
3500	999	98.10%	99.20%	99.83%	99.98%	0.12

Πίνακας 3.4.3

3-γράμματα (TF)

Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3500	1	99.00%	98.50%	99.59%	83.37%	40.08
2500	1	98.50%	99.00%	99.59%	83.37%	40.08
3500	9	99.00%	98.50%	99.77%	97.83%	9.25
2500	9	98.50%	99.00%	99.77%	97.83%	9.25
3500	999	99.00%	98.50%	99.79%	99.98%	0.10
2500	999	98.50%	99.00%	99.79%	99.98%	0.10

Πίνακας 3.4.4

5-γράμματα (Binary)

Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
6000	1	97.92%	98.74%	99.45%	83.37%	30.06
5500	1	97.92%	99.58%	99.59%	83.37%	40.08
5000	1	97.92%	99.37%	99.55%	83.37%	37
4000	1	97.70%	98.90%	99.45%	83.37%	30.06
5500	9	99.92%	99.59%	99.87%	97.83%	17.18
3000	9	97.10%	99.40%	99.82%	97.83%	11.73
5500	999	99.92%	99.59%	99.92%	99.98%	0.24
5000	999	99.88%	99.59%	99.88%	99.98%	0.16
4500	999	97.30%	99.20%	99.83%	99.98%	0.12
3000	999	97.10%	99.40%	99.88%	99.98%	0.16

Πίνακας 3.4.5

4-γράμματα (Binary)

Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4000	1	98.30%	99.60%	99.65%	83.37%	48.1
3500	1	98.50%	99.60%	99.69%	83.37%	53.44
3000	1	98.30%	99.60%	99.65%	83.37%	48.1
4000	9	98.30%	99.60%	99.88%	97.83%	18.5
3500	9	98.50%	99.60%	99.89%	97.83%	19.24
3000	9	98.30%	99.60%	99.88%	97.83%	18.5
4000	999	98.30%	99.60%	99.92%	99.98%	0.24
3500	999	98.50%	99.60%	99.92%	99.98%	0.24
3000	999	98.30%	99.60%	99.92%	99.98%	0.24

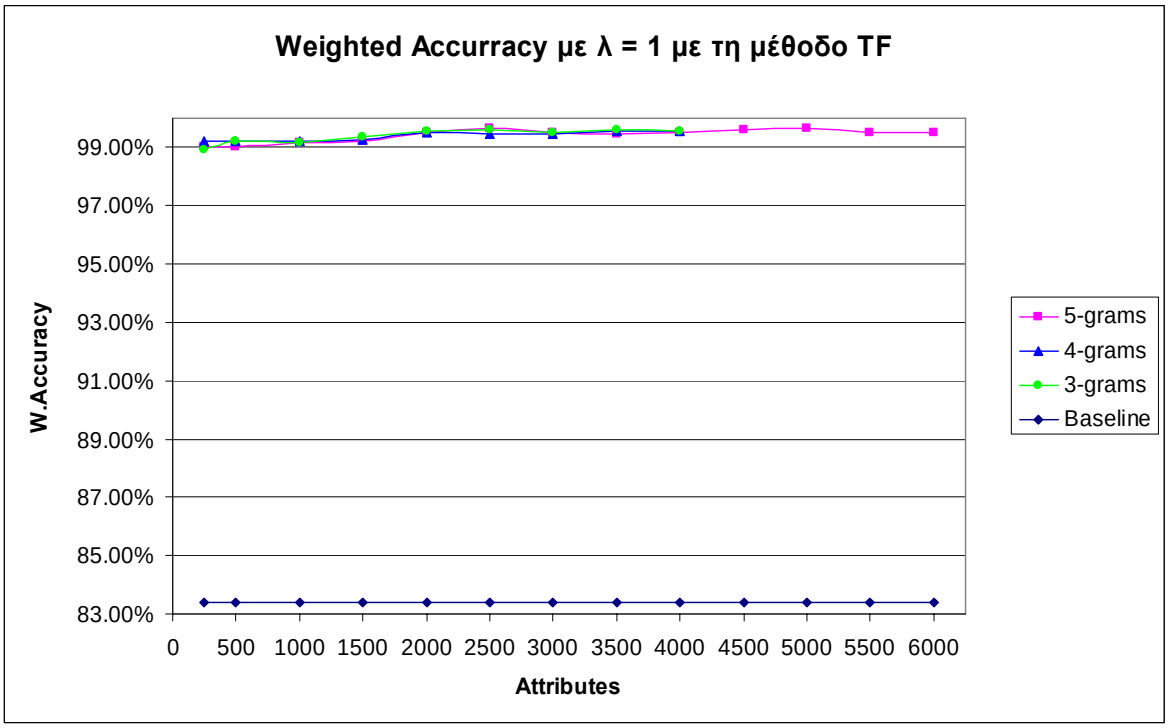
Πίνακας 3.4.6

3-γράμματα (Binary)

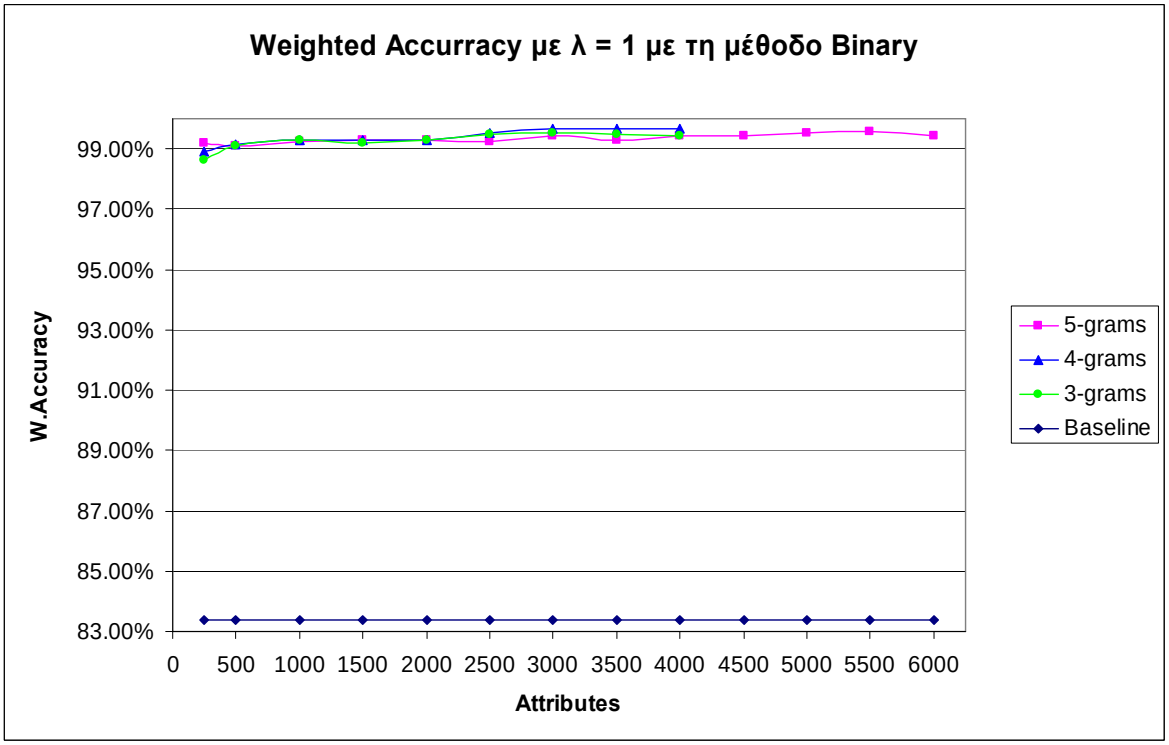
Αριθμός ν-γραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3500	1	98.10%	98.70%	99.48%	83.37%	32.07
3000	1	98.10%	99.00%	99.52%	83.37%	34.36
2500	1	97.90%	98.90%	99.48%	83.37%	32.07
4000	9	97.70%	98.90%	99.75%	97.83%	8.59
3000	9	98.10%	99.00%	99.76%	97.83%	8.91
2500	9	97.90%	98.90%	99.75%	97.83%	8.75
4000	999	97.70%	98.90%	99.79%	99.98%	0.1
3000	999	98.10%	99.00%	99.79%	99.98%	0.1
2500	999	97.90%	98.90%	99.79%	99.98%	0.1

Πίνακας 3.4.7

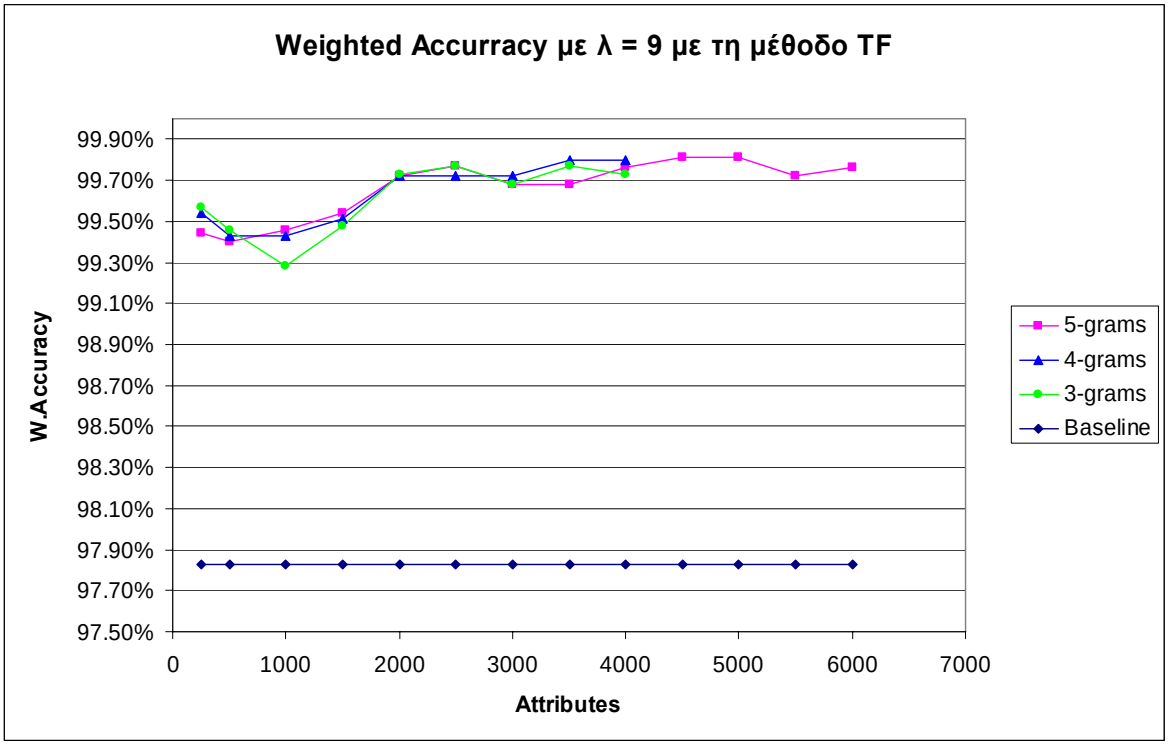
Τώρα με τη βοήθεια των διαγραμμάτων απεικόνισης των weighted accuracy συναρτήσει του πλήθους των ν-γραμμάτων και του λ θα επιχειρήσουμε να κάνουμε μία ακόμη αξιολόγηση της αποτελεσματικότητας του φίλτρου μας.



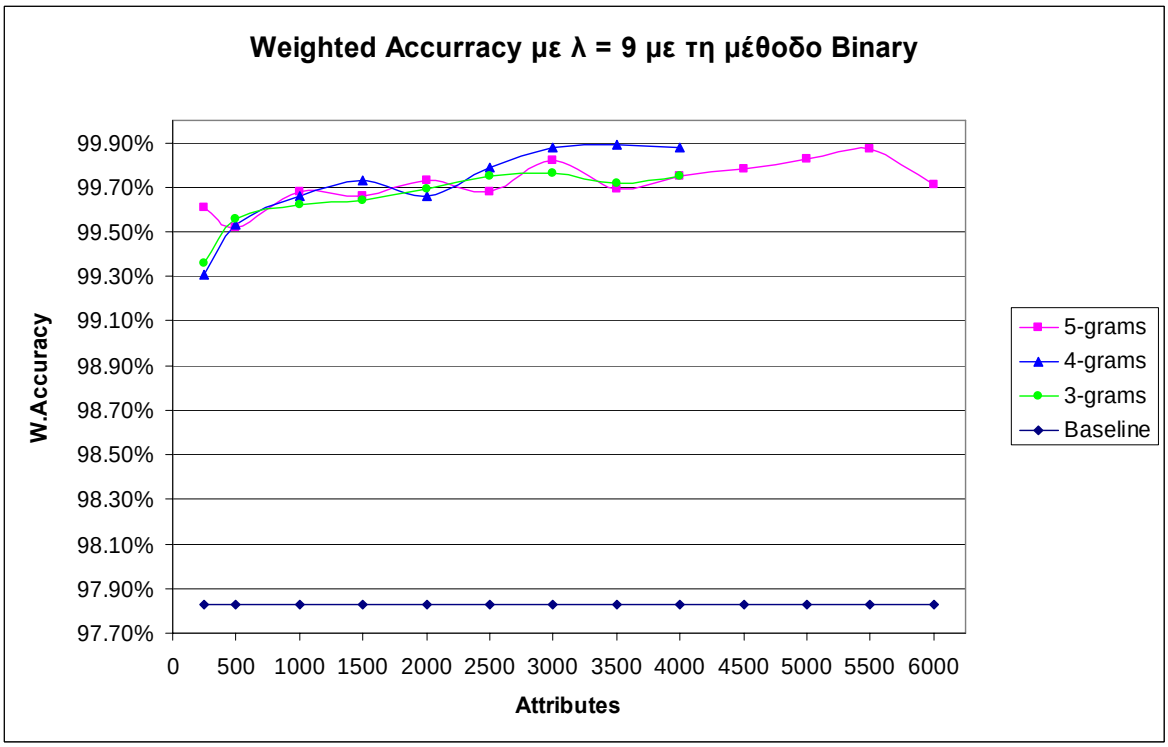
Διάγραμμα 3.4.1



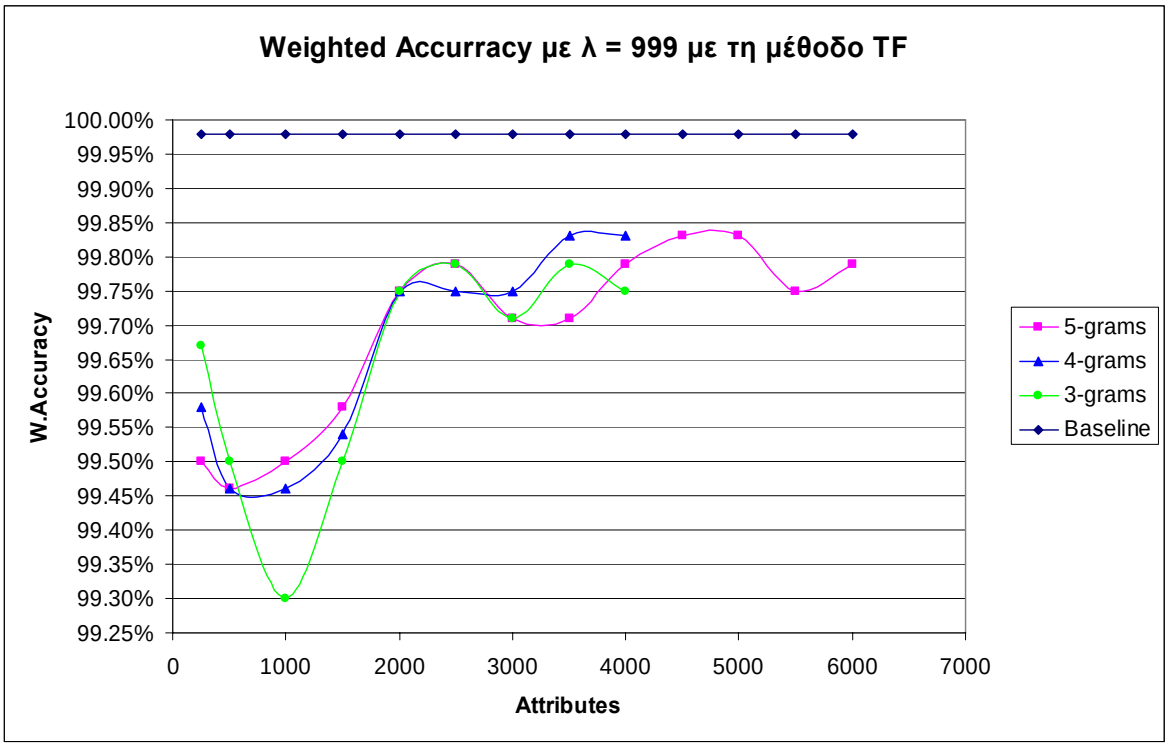
Διάγραμμα 3.4.2



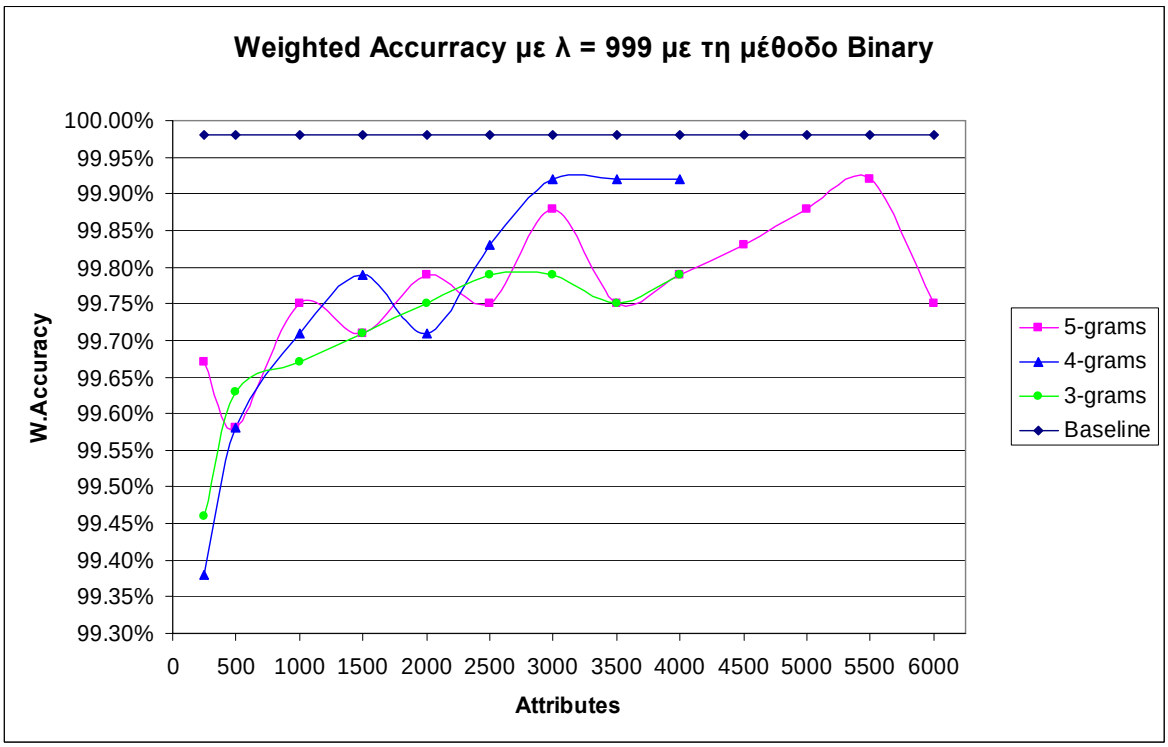
Διάγραμμα 3.4.3



Διάγραμμα 3.4.4



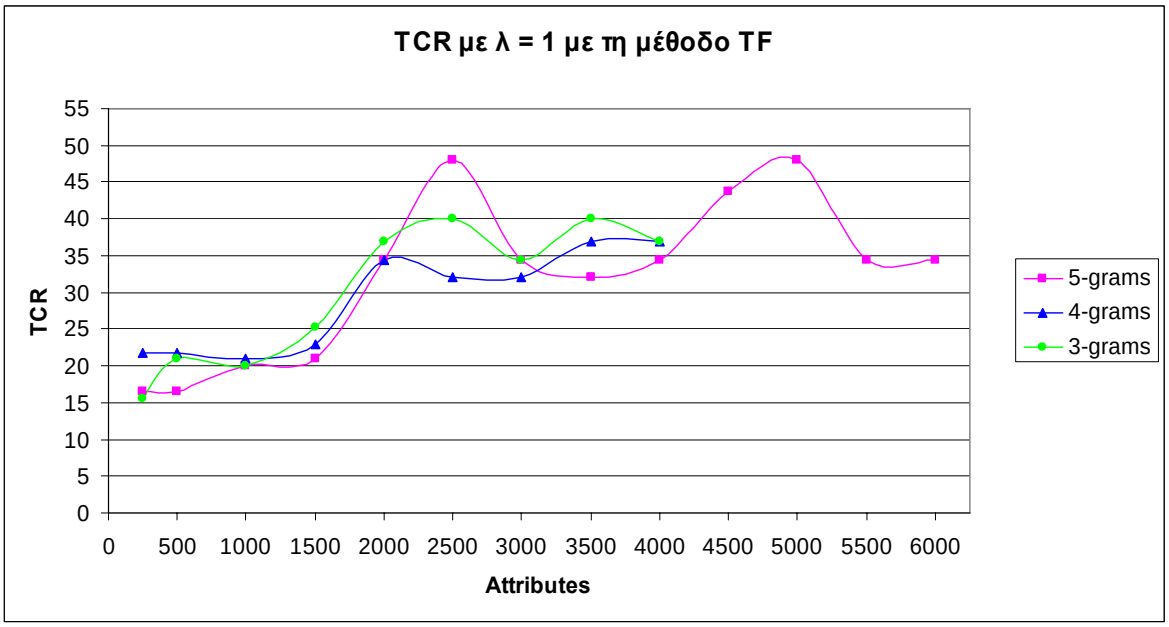
Διάγραμμα 3.4.5



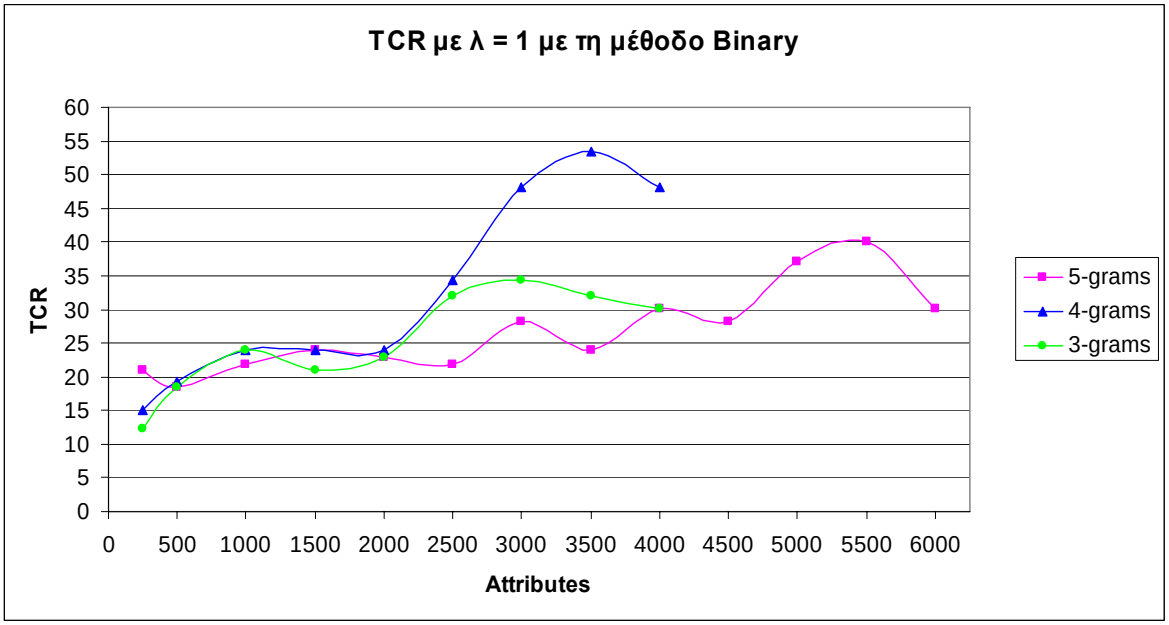
Διάγραμμα 3.4.6

Και στην περίπτωση του weighted accuracy όπως μπορεί κανείς να παρατηρήσει από τα διαγράμματα τα αποτελέσματα των μετρήσεων κυμαίνονται σε πολύ υψηλά επίπεδα. Ιδιαίτερα στο σενάριο του λ = 1 το weighted accuracy είναι κατά πολύ μεγαλύτερο του baseline. Ωστόσο όταν λ = 999 είναι ή μοναδική περίπτωση που το weighted accuracy πέφτει κάτω από το baseline έστω και ελαφρώς. Επίσης για άλλη μια φορά τα 3-γράμματα έχουν συνολικά τις χαμηλότερες αποδόσεις από όλα τα υπόλοιπα ν-γράμματα και με τις δύο μεθόδους Binary και TF. Άξιο απορίας είναι ακόμα το γεγονός ότι ενώ το weighted accuracy στα 4-γράμματα και στα 5-γράμματα έχει το ίδιο απότομες μεταβολές με όλες τις μεθόδους εκπροσώπησης στα 3-γράμματα υπάρχει αισθητή διαφορά μεταξύ των μεθόδων αφού με Binary παρουσιάζει πάλι εξαιρετική σταθερότητα κάτι που δε συμβαίνει με TF. Μια τελευταία παρατήρηση είναι το ότι τις χαμηλότερες αποδόσεις με τη μέθοδο Binary για όλες τις τιμές του λ που πήραμε πετυχαίνουν τα 3-γράμματα ενώ με τη μέθοδο TF τα 4-γράμματα πράγμα που συμβαίνει όπως ήταν αναμενόμενο και από προηγούμενες παραγράφους, όταν το πλήθος των ν-γραμμάτων φτάνει σε πολύ χαμηλά επίπεδα.

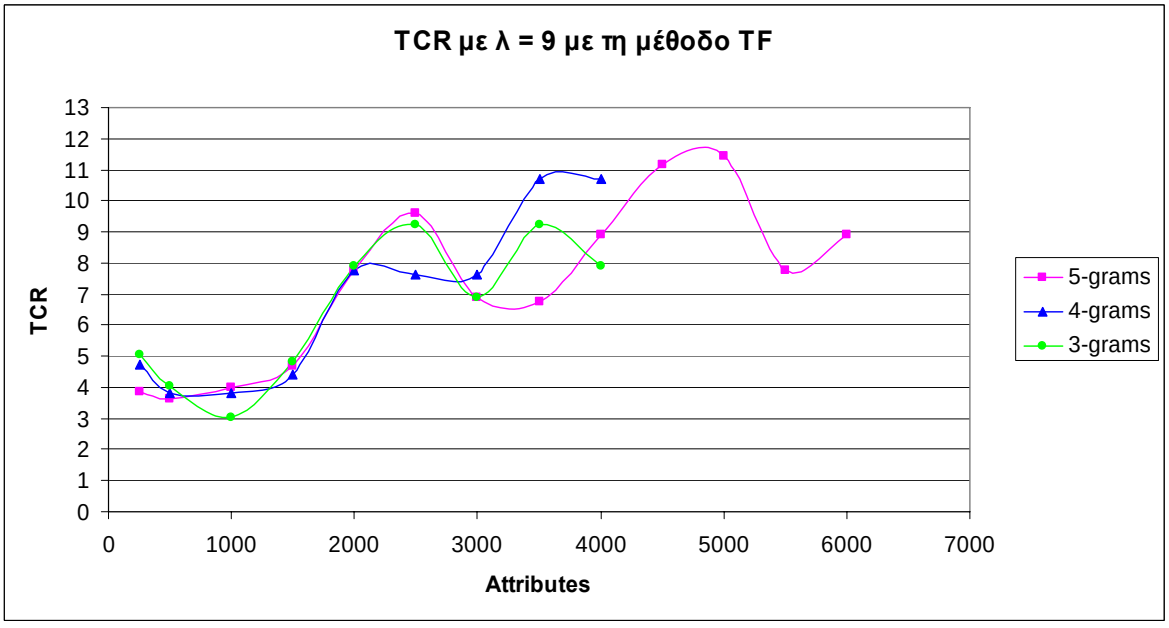
Ολοκληρώνοντας τώρα τη μελέτη μας αυτή θα εξετάσουμε τα διαγράμματα απεικόνισης του Total Cost Ratio σε συνάρτηση με το πλήθος των ν-γραμμάτων τις τιμές του λ και των μεθόδων εκπροσώπησης.



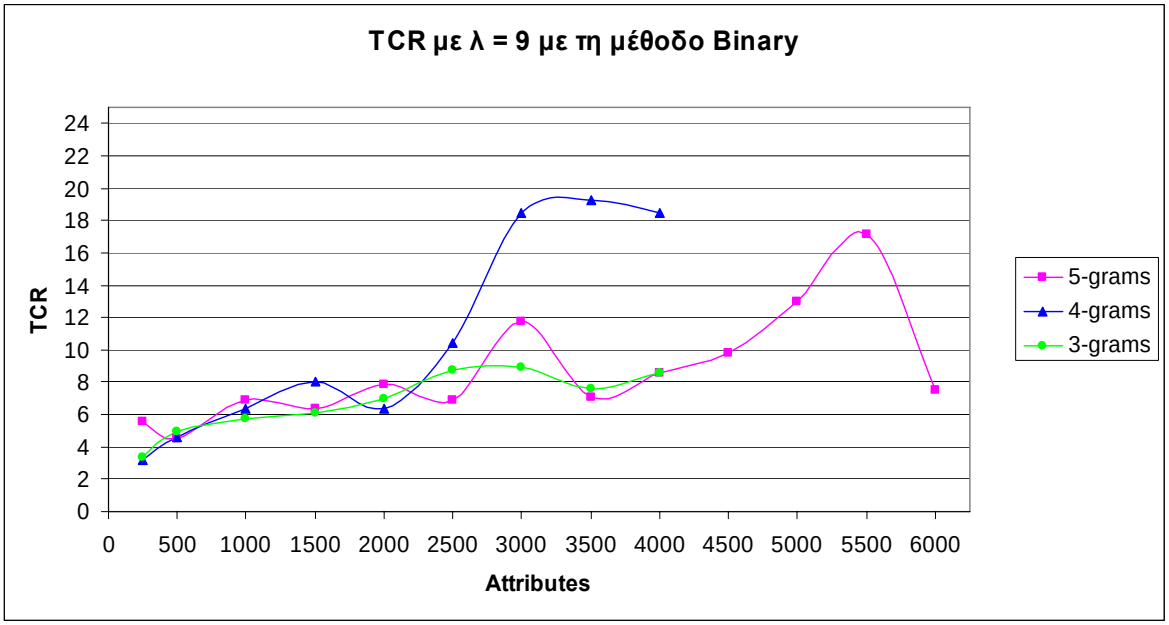
Διάγραμμα 3.4.7



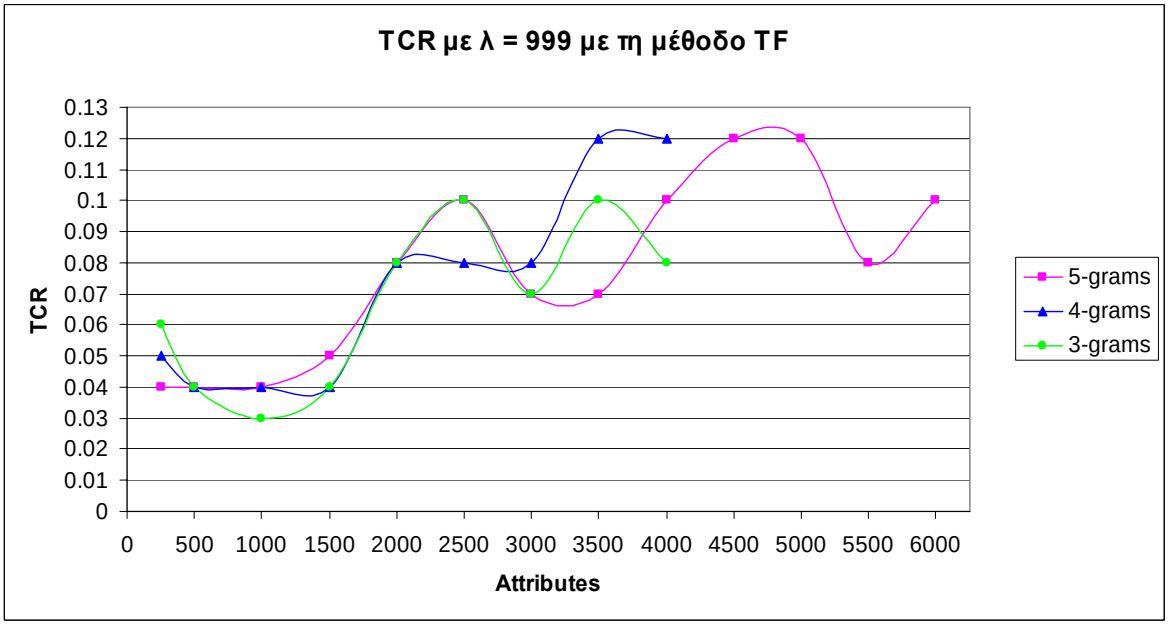
Διάγραμμα 3.4.8



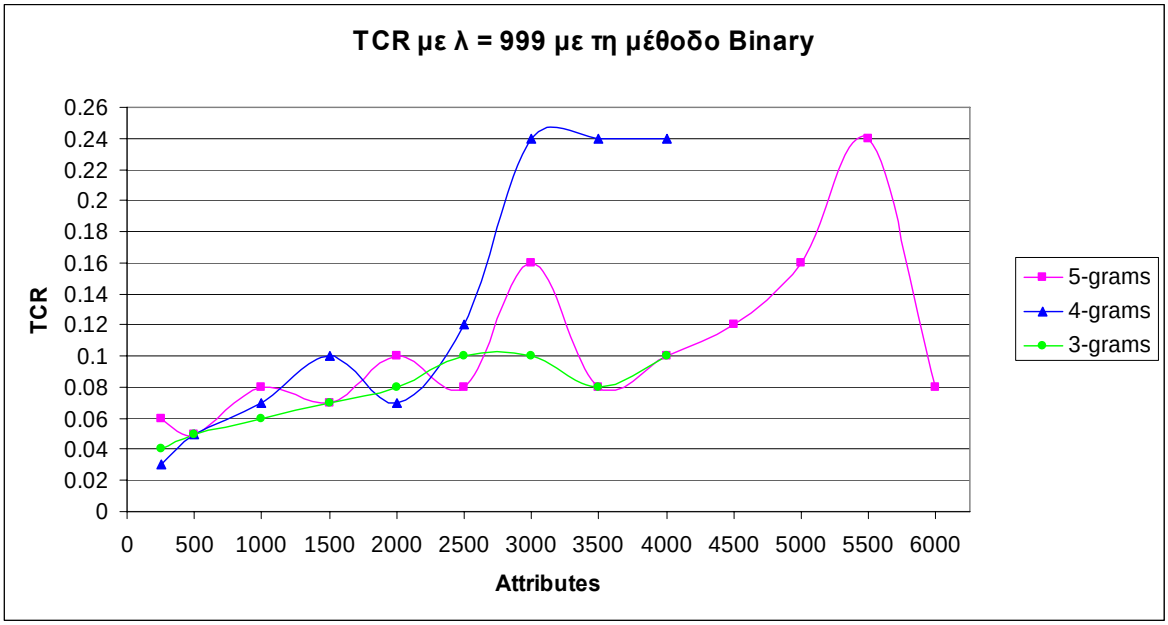
Διάγραμμα 3.4.9



Διάγραμμα 3.4.10



Διάγραμμα 3.4.11



Διάγραμμα 3.4.12

Στα διαγράμματα του Total Cost Ratio οι τιμές των μετρήσεων ποικίλουν ανάλογα με τις τιμές του λ. Πιο αναλυτικά όταν μιλάμε για το σενάριο για το οποίο η τιμή του λ είναι 1 το TCR πετυχαίνει τις υψηλότερες τιμές και γενικότερα κυμαίνεται σε πολύ καλύτερα επίπεδα σε σχέση με τα άλλα δύο σενάρια. Είναι αξιοσημείωτο το γεγονός ότι η μέγιστη τιμή που παίρνει το TCR είναι 53.44 με τη μέθοδο binary στα 3500 4-γράμματα, αριθμός που ξεπερνάει κατά πολύ τις αντίστοιχες μετρήσεις του TCR στην εργασία του κυρίου Ανδρουτσόπουλου. Αυτό οφείλεται κυρίως στα μεγάλα ποσοστά Spam Recall που πετυχαίνει το SVM σε σύγκριση με το Bayes, καθώς δεν πέφτει ποτέ κάτω από το 94.60% (στα 250 3-γράμματα Binary) ενώ το Bayes σημειώνει 63.06% σαν χαμηλότερη τιμή (στις 300 λέξεις με χρήση Lemmatizer και Stop-word List). Όπως αναφέραμε και προηγουμένως ένα φίλτρο για να είναι εφαρμόσιμο πρέπει το Total Cost Ratio να είναι μεγαλύτερο από τη μονάδα. Όλες οι μετρήσεις που έγιναν για λ = 1 και λ = 9 σε Binary αλλά και σε TF είναι κατά πολύ μεγαλύτερες της μονάδας γεγονός το οποίο σημαίνει ότι το φίλτρο μας σε αυτά τα σενάρια είναι πολύ αποτελεσματικό. Δυστυχώς όμως το ίδιο δε συμβαίνει και με την περίπτωση λ = 999 πράγμα αναμενόμενο αφού έστω και ένα ham να χαρακτηριστεί ως spam ο παρονομαστής του TCR ξεπερνάει τον αριθμητή. Συγκεκριμένα στον αριθμητή έχουμε το πλήθος των spam τα οποία είναι συνολικά 481 που είναι μικρότερο του 999 το οποίο βρίσκεται στον παρονομαστή πολλαπλασιασμένο με τον αριθμό των μπλοκαρισμένων ham. Βέβαια στην πράξη το

σενάριο λ = 999 δεν εφαρμόζεται γιατί είναι εξαιρετικά δύσκολο να βρεθεί ο ακριβής αριθμός των attributes έτσι ώστε κανένα ham να μην μπλοκαριστεί. Το τελευταίο συμπέρασμα έρχεται σε συμφωνία και με την εργασία του κυρίου Ανδρουτσόπουλου στην οποία μάλιστα βρέθηκε μόνο ένα πλήθος attributes στο οποίο όλα τα ham χαρακτηρίζονταν σωστά οπότε και το TCR βρισκόταν μεγαλύτερο της μονάδας. Μια επιπλέον παρατήρηση είναι το γεγονός ότι τα διαγράμματα του TCR έχουν πολλές ομοιότητες με αυτά του weighted accuracy καθώς στα 3-γράμματα με τη μέθοδο Binary υπάρχει εξαιρετική σταθερότητα σε όλα τα σενάρια του λ και οι μετρήσεις συνολικά είναι χαμηλότερες από τα υπόλοιπα ν-γράμματα. Επίσης οι χαμηλότερες τιμές συναντώνται στα 3-γράμματα με δύο εξαιρέσεις στα 4-γράμματα με τη μέθοδο binary για λ = 9 και λ = 999. Τέλος παρατηρούμε ότι όσο μικραίνει το πλήθος των attributes το TCR φθίνει, κάτι το οποίο είναι λογικό αφού η αποτελεσματικότητα του SVM μικραίνει όσο μικρότερο είναι το attribute set.

4. ΣΥΜΠΕΡΑΣΜΑΤΑ

Ολοκληρώνοντας την έρευνά μας πάνω στην μέθοδο κατηγοριοποίησης email καταλήξαμε σε κάποια συμπεράσματα που αφορούν τόσο τον ίδιο τον αλγόριθμο όσο και γενικότερα τα στατιστικά στοιχεία των email. Ξεκινώντας από τα στατιστικά στοιχεία μπορούμε να πούμε πως η εικόνα που τελικά αποκομίσαμε είναι αρκετά θετική αφού μπορέσαμε με την χρήση ν-γραμμάτων να φτάσουμε σε πολύ καλά αποτελέσματα. Αυτό που παρατηρήσαμε είναι πως καλές επιδόσεις είχαν τα 4-γράμματα και τα 5-γράμματα, δηλαδή ν-γράμματα με αρκετά μεγάλο μήκος ώστε να μπορούν να απεικονίσουν αρκετές από τις συχνά χρησιμοποιούμενες λέξεις, σε αντίθεση με τα 2-γράμματα και τα 3-γράμματα που το μήκος τους επιτρέπει την απεικόνιση μόνο κάποιων άρθρων και συνδέσμων (to, and, or, if κτλ.). Συνεπώς χρησιμοποιώντας ας πούμε 4-γράμματα πετυχαίνουμε μια αρκετά καλή προσέγγιση ως προς τα άλλα φίλτρα. Το κυριότερο όμως πλεονέκτημα της ανάλυσης με ν-γράμματα είναι η σωστή στατιστική καταγραφή των αλλοιωμένων λέξεων. Όπως αναφέραμε και νωρίτερα στην παρούσα εργασία, πολλοί κατασκευαστές spam email για να παρακάμψουν τα φίλτρα που εξετάζουν τα εισερχόμενα μηνύματα χρησιμοποιώντας λέξεις ως στοιχεία, παραποιούσαν κάποιες λέξεις που παρουσιάζουν υψηλό ποσοστό εμφάνισης σε spam email (π.χ. η λέξη free → f.r.e.e.). Έτσι η προσέγγιση “bag of words” αποτύγχανε να την αναγνωρίσει αφού οι τελείες

την ανάγκαζαν να σπάσει την λέξη f.r.e.e. σε 4 λέξεις : ‘f’, ’r’, ’e’, ’e’ γεγονός που μείωνε κατά πολύ τις δυνατότητες του φίλτρου. Τα ν-γράμματα δεν έχουν αυτό το πρόβλημα. Το μόνο μειονέκτημά τους σε σχέση με τις άλλες μεθόδους είναι το μεγάλο πλήθος τους πράγμα που κάνει δύσκολη την εις βάθος μελέτη τους σε όλους τους δυνατούς συνδυασμούς. Τα πρόσφατα όμως επιτεύγματα στον τομέα της επεξεργαστικής ισχύος μας επιτρέπει να κάνουμε τα πρώτα βήματα προς αυτή την κατεύθυνση και να ευελπιστούμε πως σε μερικά χρόνια ακόμα και οι προσιτοί οικιακοί υπολογιστές θα μπορούν να αναλάβουν τέτοιου μεγέθους εργασίες.

Τι παρατηρήσαμε όμως σχετικά με τις συχνότητες εμφάνισης κάποιων ν-γραμμάτων και τι μας λένε για την κατηγορία του email που τα περιέχει; Αναφέραμε και νωρίτερα πως τα αθώα email του corpus που χρησιμοποιήσαμε ήταν linguist messages δηλαδή δημοσιευμένα email γλωσσολογικού περιεχομένου. Δεν μας κάνει λοιπόν εντύπωση πως τα πιο συχνά εμφανιζόμενα ν-γράμματα σε ham ήταν τα “ingu”, “uist”, “ngui” τα οποία περιέχονται στη λέξη linguist, καθώς επίσης και τα “lang”, “angu”, “ngua”, “guag”, “uage” τα οποία περιέχονται στη λέξη language. Συγκεκριμένα το ν-γραμμα “ingu” εμφανίζεται σε παραπάνω από τα μισά ham και σε μόνο 5 spam. Αυτό μας οδηγεί στο συμπέρασμα ότι το συγκεκριμένο ν-γραμμα είναι από τα πιο χαρακτηριστικά ν-γράμματα των ham όσον αφορά το υπό εξέταση σύνολο μηνυμάτων. Δεν είναι όμως κανόνας για όλα τα ham emails φυσικά αφού σε πραγματικές συνθήκες το περιεχόμενό τους ποικίλλει. Από την άλλη, τα ν-γράμματα “ ! !”, “free”, “remo” και “ www” είναι μερικά μόνο από τα ν-γράμματα που βρέθηκαν πιο συχνά σε spam παρά σε ham. Ειδικά για το ν-γραμμα “ www” έχουμε παρατηρήσει ότι περιέχεται σε αρκετά ham μεταξύ των οποίων και τα 2 ham που αναγνωρίστηκαν ως spam στην καλύτερή μας μέτρηση στα 3500 4-γράμματα με τη μέθοδο Binary. Αυτά τα ham έχουν μικρό μέγεθος και περιέχουν στο κείμενό τους ένα ή περισσότερα links για κάποιες σελίδες στο internet όπως συμβαίνει και με πολλά spam γι’ αυτό το λόγο το φίλτρο τα χαρακτήρισε ως spam και τα ταξινόμησε λάθος. Τα ham με αυτά τα χαρακτηριστικά καλούνται hard ham. Αντίστοιχα τα spam email που ονομάζονται hard spam περιέχουν πολύ μεγάλο κείμενο και σχεδόν εξαλείφουν τις πιθανότητες εμφάνισης των λέξεων που τα χαρακτηρίζουν και έτσι ξεγελούν το φίλτρο. Αυτό είναι φυσικά φαινόμενο που συναντάται σε όλα τα είδη φίλτρων που βασίζονται στο περιεχόμενο.

Κλείνοντας με λίγα λόγια για τον αλγόριθμο SVM μπορούμε να πούμε πως είναι ό,τι καλύτερο έχει παρουσιαστεί τα τελευταία χρόνια στον τομέα

κατηγοριοποίησης κειμένου γενικά αφού παρουσιάζει εξαιρετικά αποτελέσματα σε πολλούς τομείς [5][21]. Συνεπώς απλά μένει να δοκιμάσουμε περισσότερα πειράματα σε υλοποιήσεις του αλγόριθμου αυτού που ίσως να είναι πιο αποδοτικές από την έκδοση του Platt's SMO με την οποία έγιναν τα πειράματά μας στο πρόγραμμα WEKA.

5. ΜΕΛΛΟΝΤΙΚΕΣ ΒΕΛΤΙΩΣΕΙΣ

Όπως αναφέραμε και σε προηγούμενη ενότητα ο αλγόριθμος SVM δουλεύει καλύτερα με όσο το δυνατόν περισσότερα αν είναι δυνατόν και όλα τα attributes του dataset. Αυτό βέβαια δεν ήταν εφικτό με τον εξοπλισμό που είχαμε στη διάθεσή μας. Ωστόσο πιστεύουμε πως με ένα πιο ισχυρό μηχάνημα ή μια πιο γρήγορη υλοποίηση του SVM το πλήθος των 10000 attributes ίσως και παραπάνω να είναι εφικτό να υποστεί επεξεργασία έτσι ώστε να μας δώσει ακόμα καλύτερα αποτελέσματα. Η λογική που κρύβεται πίσω από το μέγιστο δυνατό πλήθος attributes έχει να κάνει με το πιο είναι τελικά αυτό που λέμε «σημαντικό» attribute για την ταξινόμηση. Στην εργασία μας επιλέγαμε αρχικά ένα πλήθος ν-γραμμάτων τα οποία παρουσίαζαν τη μεγαλύτερη συνολική εμφάνιση στο dataset. Αυτό δε σημαίνει απαραίτητα ότι αυτά είναι και τα πιο σημαντικά και αυτό φαίνεται από το γεγονός ότι πετυχαίναμε καλύτερα αποτελέσματα και με λιγότερα attributes από το μέγιστο πλήθος. Είναι προφανές λοιπόν ότι τα attributes που αποκλείονταν κάθε φορά από τις μετρήσεις δεν ήταν τόσο σημαντικά τελικά κι ας είχαν μεγαλύτερη συνολική εμφάνιση από κάποια άλλα που δεν αποκλείστηκαν. Έτσι λοιπόν συμπεραίνουμε ότι μεγαλώνοντας το μέγιστο πλήθος attributes μπορεί να βρεθούν ν-γράμματα τα οποία έχουν μικρότερη συνολική εμφάνιση αλλά μεγαλύτερη σημασία στην ταξινόμηση. Το πρόβλημα πάντως που παραμένει με αυτήν την προσέγγιση είναι το αν θα μπορέσουμε κάποια στιγμή να εξετάσουμε συνολικά όλα τα πιθανά εμφανιζόμενα ν-γράμματα και μάλιστα με προσιτό εξοπλισμό και σε πεπερασμένο (λογικό) χρόνο. Τότε θα έχουμε εξαντλήσει τις δυνατότητες του αλγορίθμου και θα μπορούμε να μιλάμε με βεβαιότητα για το ακριβές πλήθος attributes και κυρίως ποια είναι αυτά που μας δίνουν το καλύτερο αποτέλεσμα κάθε φορά.

ΠΑΡΑΡΤΗΜΑ Α' : SPAM ANALYZER

Για την διεξαγωγή των πειραμάτων μας έπρεπε να περάσουμε τα αρχεία των email από επεξεργασία ώστε να δημιουργήσουμε ειδικά αρχεία με τις πληροφορίες που μας χρειάζονται για να εφαρμόσουμε τον αλγόριθμο. Ειδικότερα στην περίπτωσή μας έπρεπε να φτιάξουμε ένα πρόγραμμα που θα έπαιρνε όλα τα αρχεία του Dataset, δηλαδή τα spam και τα ham και θα έβρισκε όλα τα πιθανά ν-γράμματα με τα ποσοστά εμφάνισης τους.

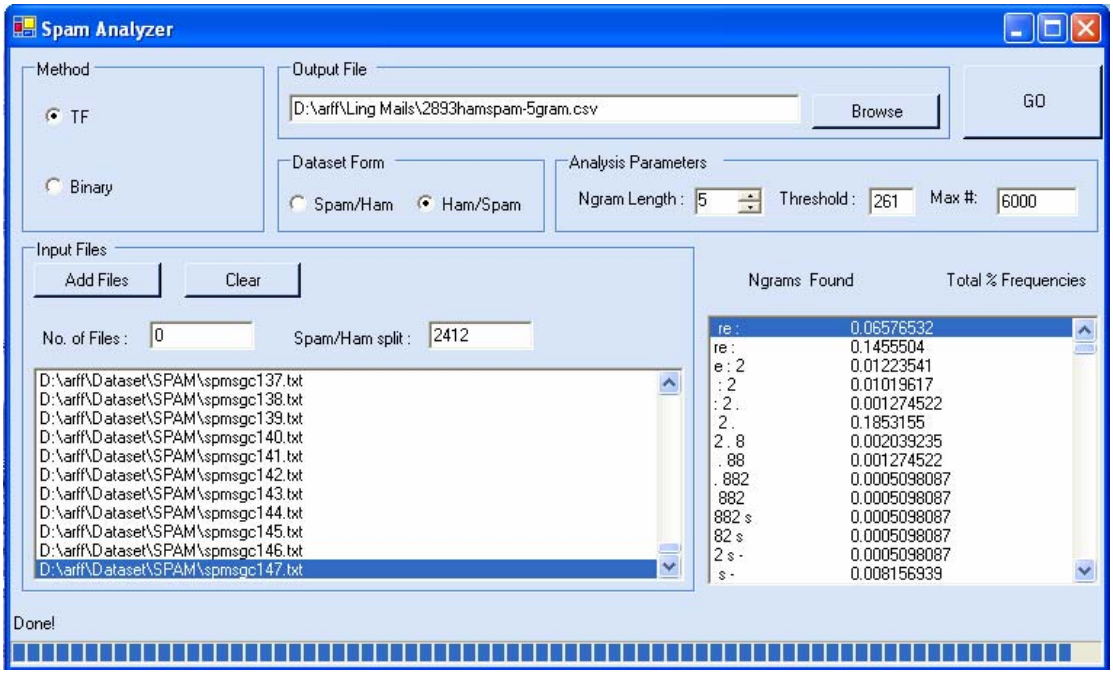
Διαλέξαμε δύο διαφορετικούς τρόπους αναπαράστασης των δεδομένων. Την συχνότητα εμφάνισης όρων (TF – Term frequency) και την δυαδική εκπροσώπηση (Binary Representation). Στην πρώτη μετράμε τις εμφανίσεις ενός ν-γράμματος σε ένα συγκεκριμένο mail και υπολογίζουμε το ποσοστό εμφάνισής του με τον τύπο :

(αριθμός εμφάνισης / συνολικά ν-γράμματα email) * 100%

Για την δυαδική μέθοδο απλά μας ενδιέφερε αν ένα ν-γραμμα εμφανιζόταν οπότε το συμβολίζαμε με το 1 ή αλλιώς με το 0.

Η διαδικασία που ακολουθήσαμε είναι η εξής: Σε κάθε email από την αρχή ως το τέλος του παίρναμε διαδοχικά ν-γράμματα με βήμα ενός χαρακτήρα και τα καταγράφαμε σε μια λίστα. Σε μία δεύτερη λίστα καταγράφαμε κατ' αντιστοιχία το πλήθος των εμφανίσεων κάθε ν-γράμματος στο τρέχον email. Κάθε φορά το επιλεγμένο ν-γραμμα το συγκρίναμε με τα στοιχεία της υπάρχουσας λίστας και αν υπήρχε ήδη απλά αυξάναμε τον αριθμό εμφανίσεών του στην αντίστοιχη θέση της δεύτερης λίστας. Αλλιώς το προσθέταμε στο τέλος της λίστας και του θέταμε αριθμό εμφάνισης 1. Η λίστα αυτή με τα ν-γράμματα έμενε ανέπαφη μετά την επεξεργασία του κάθε email, αλλά μηδενίζονταν στην δεύτερη λίστα οι μετρητές εμφάνισης έτσι ώστε να αναλυθεί το νέο email εξ αρχής.

Ας εξηγήσουμε τώρα πιο διεξοδικά το interface του προγράμματος.



Εικόνα 1

Στην εικόνα 1 βλέπουμε το πρόγραμμά μας αφού έχει τελειώσει την ανάλυση του Dataset για 5-γράμματα. Πάνω αριστερά υπάρχει ο επιλογέας της μεθόδου που θέλουμε να ακολουθηθεί για την ανάλυση των mail (TF ή Binary).

Στο πλαίσιο Dataset Form ορίζουμε την σειρά με την οποία έχουν δοθεί τα προς εξέταση email, δηλαδή αν έχουμε βάλει πρώτα τα spam ή τα ham.

Πιο δεξιά βρίσκονται τα Analysis Parameters που αποτελούνται από τρεις παραμέτρους : Το μήκος του ν-γράμματος (Ngram Length), το κάτω όριο επιλογής (Threshold) και το μέγιστο πλήθος στοιχείων που θέλουμε να κρατήσουμε μετά την ανάλυση (Max #). Τα τελευταία δύο μας βοηθούν να ορίσουμε το τελικό μέγεθος του Dataset file καθώς αν δεν κάναμε κάποια επιλογή εκεί θα γινόταν τεράστιο και δεν θα μπορούσαμε να το επεξεργαστούμε. Ενδεικτικά αναφέρουμε πως το πλήθος των διαφορετικών 5-γραμμάτων που βρέθηκαν μετά την ανάλυση του δείγματός μας έφταναν τις 350.000! Για να κάνουμε λοιπόν την διαλογή μας χρησιμοποιούμε το κάτω όριο και το μέγιστο αριθμό σε συνδυασμό ως εξής : Κάθε φορά που τελειώνει η ανάλυση ενός email οι αριθμοί εμφάνισης κάθε στοιχείου αθροίζονται σε έναν πίνακα. Συνεπώς με το πέρας της διαδικασίας έχουμε τις συνολικές εμφανίσεις κάθε ν–γράμματος σε όλα τα email. Ξεκινάμε την επιλογή των συχνότερων με threshold = 3 δηλαδή επιλέγουμε αυτά που έχουν εμφανιστεί τουλάχιστον τρεις φορές. Αυτό είναι συνήθης τακτική που χρησιμοποιείται γιατί θεωρούμε πως οι λέξεις ή τα ν-γράμματα που εμφανίζονται λιγότερο από 3 φορές είναι λάθος γραμμένα. Αν αυτά

είναι περισσότερα από το μέγιστο πλήθος στοιχείων τότε αυξάνουμε το threshold κατά 1 και ξανακάνουμε διαλογή μέχρις ότου φτάσουμε στον επιθυμητό αριθμό. Στο παράδειγμά μας π.χ. για να επιλέξουμε τα 6000 πιο συχνά εμφανιζόμενα 5-γράμματα έπρεπε το threshold να πάρει την τιμή 261.

Στο πλαίσιο Input Files υπάρχει η λίστα με τα email που θέλουμε να αναλύσουμε. Τα εισάγουμε με το κουμπί Add Files και καθαρίζουμε την λίστα με το Clear. Είναι σημαντικό να μην ξεχάσουμε να γράψουμε τον αριθμό Spam/Ham Split που δείχνει στο πρόγραμμα σε ποιο αριθμό email τελειώνουν τα spam ή τα ham (ανάλογα με το Dataset Form που έχουμε επιλέξει) και συνεχίζουν τα αντίστοιχα ham ή spam. Με αυτό δίνουμε την εντολή στο πρόγραμμα να αλλάξει την τιμή του τελευταίου attribute με το όνομα class από 0 (ham), σε 1 (spam) ή το αντίστροφο.

Τέλος έχουμε και το πλαίσιο εξόδου αρχείου όπου ορίζουμε το όνομα του αρχείου με τα τελικά αποτελέσματα που θα αναλυθούν. Το Format που επιλέξαμε είναι το .CSV δηλαδή Comma Separated Values όπου γράφουμε όλα τα ν-γράμματα σε μια γραμμή χωριζόμενα με κόμματα και από κάτω τους πάλι χωριζόμενα με κόμματα όλα τα ποσοστά εμφάνισής τους σε email ανά γραμμή, με ακρίβεια τεσσάρων δεκαδικών ψηφίων. Μια ενδεικτική εικόνα ενός τέτοιου αρχείου με 2-γράμματα είναι :

```
' r','re','e ',' :',': ',' 2','2 ',' .','. ',' 8','88','82',class

0.5687,1.0427,3.981,0.7583,0.7583,0.2844,0.4739,1.2322,1.2322,0.1896
,0.0948,0.0948,0
0.3868,0.5803,3.0948,0.3868,0.3868,0,0,1.1605,1.1605,0,0,0,1
0.7353,2.2059,2.9412,0,0,0.7353,1.4706,4.4118,4.4118,0.7353,0.7353,0
,1
```

Σχήμα 1

Στο τέλος κάθε γραμμής προσθέτουμε ένα τελευταίο attribute με το όνομα class το οποίο χαρακτηρίζει το κάθε mail σαν spam (1) ή ham (0). Αυτό είναι που χρησιμοποιείται από το πρόγραμμα που εκτελεί τον αλγόριθμο SVM με στόχο να χαρτογραφήσει στο επίπεδο τις συντεταγμένες των spam και των ham για να μπορεί να κάνει classification.

Πέρα όμως από την στατιστική ανάλυση των mails το πρόγραμμα μας έπρεπε να έχει και κάποια επιπλέον «χαρακτηριστικά» που θα το διευκόλυναν στην ανάλυση. Το dataset που χρησιμοποιήσαμε εδώ για την ανάλυσή μας ήταν το μοναδικό που διατίθεται στο δίκτυο σε «καθαρή» μορφή, δηλαδή μόνο με το Subject και το Body. Τα περισσότερα spam και ham corpus που κυκλοφορούν είναι σε ακατέργαστη μορφή δηλαδή περιέχουν μέσα εκτός από το κυρίως κείμενο και πολλές άλλες πληροφορίες εν μέρει άχρηστες όπως στοιχεία αποστολέα και παραλήπτη, ημερομηνίες αποστολής, χαρακτήρες εκτύπωσης (change line, tab, line feed κ.α.), HTML κώδικα, εντολές που αφορούν τον email client ακόμα και ολόκληρα συνημμένα αρχεία κωδικοποιημένα. Όλα αυτά στην παρούσα εργασία θεωρήθηκαν ως άχρηστες πληροφορίες γι' αυτό και έχουμε ενσωματώσει στο πρόγραμμα μας συναρτήσεις για τον καθαρισμό έως ένα βαθμό των email καθώς σε περίπτωση που δεν τα περιορίζαμε οι συνδυασμοί των ν-γραμμάτων εκτοξεύονταν σε αστρονομικά νούμερα (550-600 χιλιάδες 5-γράμματα) που έκαναν το έργο της στατιστικής ανάλυσης πάρα πολύ χρονοβόρο. Σε μελλοντικές αναπτύξεις του προγράμματος μας σκοπεύουμε την περαιτέρω βελτίωση των φίλτρων καθαρισμού ώστε να μπορούμε να επεξεργαζόμαστε κάθε μορφή email απομονώνοντας ακριβώς τα τμήματα που χρειαζόμαστε.

Μετά το πέρας της ανάλυσης στο κάτω δεξί μέρος της εφαρμογής εμφανίζονται όλα τα διαφορετικά ν-γράμματα που εντοπίστηκαν κατά την ανάλυση με τα ποσοστά εμφάνισής τους σε σχέση με τον συνολικό αριθμό τους. Αυτά είναι απλά βοηθητικά για τον ερευνητή και δεν παίρνουν μέρος στην τελική επεξεργασία της ταξινόμησης.

ΠΑΡΑΡΤΗΜΑ Β' : Αναλυτικοί πίνακες αποτελεσμάτων

Για τους αναγνώστες που θα ήθελαν μια πιο αναλυτική αναφορά στα αποτελέσματα των πειραμάτων μας παραθέτουμε τους πίνακες τους οποίους δημιουργήσαμε και που αποτέλεσαν την πηγή των σχεδιαγραμμάτων του κεφαλαίου 3.

TF

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	999	98.10%	99.00%	99.79%	99.98%	0.10
	5500	999	98.30%	98.70%	99.75%	99.98%	0.08
	5000	999	98.80%	**99.20%**	**99.83%**	99.98%	**0.12**
	4500	999	98.50%	**99.20%**	**99.83%**	99.98%	**0.12**
	4000	999	98.10%	99.00%	99.79%	99.98%	0.10
	3500	999	98.30%	98.50%	99.71%	99.98%	0.07
	3000	999	98.50%	98.50%	99.71%	99.98%	0.07
	2500	999	**99.00%**	99.00%	99.79%	99.98%	0.10
	2000	999	98.30%	98.70%	99.75%	99.98%	0.08
	1500	999	97.30%	97.90%	99.58%	99.98%	0.05
	1000	999	97.50%	97.50%	99.50%	99.98%	0.04
	500	999	96.70%	97.30%	99.46%	99.98%	0.04
	250	999	96.50%	97.50%	99.50%	99.98%	0.04

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	999	98.10%	**99.20%**	**99.83%**	99.98%	**0.12**
	3500	999	98.10%	**99.20%**	**99.83%**	99.98%	**0.12**
	3000	999	98.10%	98.70%	99.75%	99.98%	0.08
	2500	999	98.10%	98.70%	99.75%	99.98%	0.08
	2000	999	**98.30%**	98.70%	99.75%	99.98%	0.08
	1500	999	97.90%	97.70%	99.54%	99.98%	0.04
	1000	999	97.90%	97.30%	99.46%	99.98%	0.04
	500	999	98.10%	97.30%	99.46%	99.98%	0.04
	250	999	97.50%	97.90%	99.58%	99.98%	0.05

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	999	98.50%	98.80%	99.75%	99.98%	0.08
	3500	999	**99.00%**	98.50%	**99.79%**	99.98%	**0.10**
	3000	999	98.50%	98.50%	99.71%	99.98%	0.07
	2500	999	98.50%	**99.00%**	**99.79%**	99.98%	**0.10**
	2000	999	98.50%	98.80%	99.75%	99.98%	0.08
	1500	999	98.50%	97.50%	99.50%	99.98%	0.04
	1000	999	98.50%	96.50%	99.30%	99.98%	0.03
	500	999	97.70%	97.50%	99.50%	99.98%	0.04
	250	999	95.20%	98.30%	99.67%	99.98%	0.06

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	9	98.10%	99.00%	99.76%	97.83%	8.91
	5500	9	98.30%	98.70%	99.72%	97.83%	7.76
	5000	9	98.80%	**99.20%**	**99.81%**	97.83%	**11.45**
	4500	9	98.50%	**99.20%**	**99.81%**	97.83%	11.18
	4000	9	98.10%	99.00%	99.76%	97.83%	8.91
	3500	9	98.30%	98.50%	99.68%	97.83%	6.77
	3000	9	98.50%	98.50%	99.68%	97.83%	6.87
	2500	9	**99.00%**	99.00%	99.77%	97.83%	9.62
	2000	9	98.30%	98.70%	99.72%	97.83%	7.76
	1500	9	97.30%	97.90%	99.54%	97.83%	4.67
	1000	9	97.50%	97.50%	99.46%	97.83%	4.01
	500	9	96.70%	97.30%	99.40%	97.83%	3.62
	250	9	96.50%	97.50%	99.44%	97.83%	3.85

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	9	98.10%	**99.20%**	**99.80%**	97.83%	**10.69**
	3500	9	98.10%	**99.20%**	**99.80%**	97.83%	**10.69**
	3000	9	98.10%	98.70%	99.72%	97.83%	7.63
	2500	9	98.10%	98.70%	99.72%	97.83%	7.63
	2000	9	**98.30%**	98.70%	99.72%	97.83%	7.76
	1500	9	97.90%	97.70%	99.51%	97.83%	4.41
	1000	9	97.90%	97.30%	99.43%	97.83%	3.79
	500	9	98.10%	97.30%	99.43%	97.83%	3.82
	250	9	97.50%	97.90%	99.54%	97.83%	4.72

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	9	98.50%	98.80%	99.73%	97.83%	7.89
	3500	9	**99.00%**	98.50%	**99.77%**	97.83%	**9.25**
	3000	9	98.50%	98.50%	99.68%	97.83%	6.87
	2500	9	98.50%	**99.00%**	**99.77%**	97.83%	**9.25**
	2000	9	98.50%	98.80%	99.73%	97.83%	7.89
	1500	9	98.50%	97.50%	99.48%	97.83%	4.81
	1000	9	98.50%	96.50%	99.28%	97.83%	3.01
	500	9	97.70%	97.50%	99.46%	97.83%	4.04
	250	9	95.20%	98.30%	99.57%	97.83%	5.06

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	1	98.10%	99.00%	99.52%	83.37%	34.36
	5500	1	98.30%	98.70%	99.52%	83.37%	34.36
	5000	1	98.80%	**99.20%**	**99.65%**	83.37%	**48.10**
	4500	1	98.50%	**99.20%**	99.62%	83.37%	43.73
	4000	1	98.10%	99.00%	99.52%	83.37%	34.36
	3500	1	98.30%	98.50%	99.48%	83.37%	32.07
	3000	1	98.50%	98.50%	99.52%	83.37%	34.36
	2500	1	**99.00%**	99.00%	**99.65%**	83.37%	**48.10**
	2000	1	98.30%	98.70%	99.52%	83.37%	34.36
	1500	1	97.30%	97.90%	99.20%	83.37%	20.91
	1000	1	97.50%	97.50%	99.17%	83.37%	20.04
	500	1	96.70%	97.30%	99.00%	83.37%	16.59
	250	1	96.50%	97.50%	99.00%	83.37%	16.59

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	1	98.10%	**99.20%**	**99.55%**	83.37%	**37.00**
	3500	1	98.10%	**99.20%**	**99.55%**	83.37%	**37.00**
	3000	1	98.10%	98.70%	99.48%	83.37%	32.07
	2500	1	98.10%	98.70%	99.48%	83.37%	32.07
	2000	1	**98.30%**	98.70%	99.52%	83.37%	34.36
	1500	1	97.90%	97.70%	99.27%	83.37%	22.90
	1000	1	97.90%	97.30%	99.20%	83.37%	20.91
	500	1	98.10%	97.30%	99.24%	83.37%	21.86
	250	1	97.50%	97.90%	99.24%	83.37%	21.86

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	1	98.50%	98.80%	99.55%	83.37%	37.00
	3500	1	**99.00%**	98.50%	**99.59%**	83.37%	**40.08**
	3000	1	98.50%	98.50%	99.52%	83.37%	34.36
	2500	1	98.50%	**99.00%**	**99.59%**	83.37%	**40.08**
	2000	1	98.50%	98.80%	99.55%	83.37%	37.00
	1500	1	98.50%	97.50%	99.34%	83.37%	25.32
	1000	1	98.50%	96.50%	99.17%	83.37%	20.04
	500	1	97.70%	97.50%	99.20%	83.37%	20.91
	250	1	95.20%	98.30%	98.93	83.37%	15.52

BINARY

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	999	**97.92%**	98.74%	99.75%	99.98%	0.08
	5500	999	**97.92%**	**99.58%**	**99.92%**	99.98%	**0.24**
	5000	999	**97.92%**	99.37%	99.88%	99.98%	0.16
	4500	999	97.30%	99.20%	99.83%	99.98%	0.12
	4000	999	97.70%	98.90%	99.79%	99.98%	0.10
	3500	999	97.10%	98.70%	99.75%	99.98%	0.08
	3000	999	97.10%	99.40%	99.88%	99.98%	0.16
	2500	999	96.70%	98.70%	99.75%	99.98%	0.08
	2000	999	96.70%	98.90%	99.79%	99.98%	0.10
	1500	999	97.30%	98.50%	99.71%	99.98%	0.07
	1000	999	96.70%	98.70%	99.75%	99.98%	0.08
	500	999	96.70%	97.90%	99.58%	99.98%	0.05
	250	999	96.90%	98.30%	99.67%	99.98%	0.06

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	999	98.30%	**99.60%**	**99.92%**	99.98%	**0.24**
	3500	999	**98.50%**	**99.60%**	**99.92%**	99.98%	**0.24**
	3000	999	98.30%	**99.60%**	**99.92%**	99.98%	**0.24**
	2500	999	97.90%	99.20%	99.83%	99.98%	0.12
	2000	999	97.30%	98.50%	99.71%	99.98%	0.07
	1500	999	96.90%	98.90%	99.79%	99.98%	0.10
	1000	999	97.30%	98.50%	99.71%	99.98%	0.07
	500	999	96.90%	97.90%	99.58%	99.98%	0.05
	250	999	96.50%	96.90%	99.38%	99.98%	0.03

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	999	97.70%	98.90%	**99.79%**	99.98%	**0.10**
	3500	999	**98.10%**	98.70%	99.75%	99.98%	0.08
	3000	999	**98.10%**	**99.00%**	**99.79%**	99.98%	**0.10**
	2500	999	97.90%	98.90%	**99.79%**	99.98%	**0.10**
	2000	999	96.90%	98.70%	99.75%	99.98%	0.08
	1500	999	96.70%	98.50%	99.71%	99.98%	0.07
	1000	999	97.50%	98.30%	99.67%	99.98%	0.06
	500	999	96.50%	98.10%	99.63%	99.98%	0.05
	250	999	94.60%	97.20%	99.46%	99.98%	0.04

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	9	97.92%	98.74%	99.71%	97.83%	7.52
	5500	9	97.92%	99.58%	99.87%	97.83%	17.18
	5000	9	97.92%	99.37%	99.83%	97.83%	13
	4500	9	97.30%	99.20%	99.78%	97.83%	9.82
	4000	9	**97.70%**	98.90%	99.75%	97.83%	8.59
	3500	9	97.10%	98.70%	99.69%	97.83%	7.07
	3000	9	97.10%	**99.40%**	**99.82%**	97.83%	**11.73**
	2500	9	96.70%	98.70%	99.68%	97.83%	6.87
	2000	9	96.70%	98.90%	99.73%	97.83%	7.89
	1500	9	97.30%	98.50%	99.66%	97.83%	6.33
	1000	9	96.70%	98.70%	99.68%	97.83%	6.87
	500	9	96.70%	97.90%	99.52%	97.83%	4.54
	250	9	96.90%	98.30%	99.61%	97.83%	5.53

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	9	98.30%	**99.60%**	99.88%	97.83%	18.50
	3500	9	**98.50%**	**99.60%**	**99.89%**	97.83%	**19.24**
	3000	9	98.30%	**99.60%**	99.88%	97.83%	18.50
	2500	9	97.90%	99.20%	99.79%	97.83%	10.46
	2000	9	97.30%	98.50%	99.66%	97.83%	6.33
	1500	9	96.90%	98.90%	99.73%	97.83%	8.02
	1000	9	97.30%	98.50%	99.66%	97.83%	6.33
	500	9	96.90%	97.90%	99.53%	97.83%	4.58
	250	9	96.50%	96.90%	99.31%	97.83%	3.16

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	9	97.70%	98.90%	99.75%	97.83%	8.59
	3500	9	**98.10%**	98.70%	99.72%	97.83%	7.63
	3000	9	**98.10%**	**99.00%**	**99.76%**	97.83%	**8.91**
	2500	9	97.90%	98.90%	99.75%	97.83%	8.75
	2000	9	96.90%	98.70%	99.69%	97.83%	6.97
	1500	9	96.70%	98.50%	99.64%	97.83%	6.09
	1000	9	97.50%	98.30%	99.62%	97.83%	5.73
	500	9	96.50%	98.10%	99.56%	97.83%	4.91
	250	9	94.60%	97.20%	99.36%	97.83%	3.36

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
5γράμματα	6000	1	97.92%	98.74%	99.45%	83.37%	30.06
	5500	1	97.92%	99.58%	99.59%	83.37%	40.08
	5000	1	97.92%	99.37%	99.55%	83.37%	37
	4500	1	97.30%	99.20%	99.41%	83.37%	28.29
	4000	1	**97.70%**	98.90%	**99.45%**	83.37%	30.06
	3500	1	97.10%	98.70%	99.31%	83.37%	24.05
	3000	1	97.10%	**99.40%**	99.41%	83.37%	28.29
	2500	1	96.70%	98.70%	99.24%	83.37%	21.86
	2000	1	96.70%	98.90%	99.27%	83.37%	22.90
	1500	1	97.30%	98.50%	99.31%	83.37%	24.05
	1000	1	96.70%	98.70%	99.24%	83.37%	21.86
	500	1	96.70%	97.90%	99.10%	83.37%	18.50
	250	1	96.90%	98.30%	99.20%	83.37%	20.91

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
4γράμματα	4000	1	98.30%	**99.60%**	99.65%	83.37%	48.10
	3500	1	**98.50%**	**99.60%**	**99.69%**	83.37%	**53.44**
	3000	1	98.30%	**99.60%**	99.65%	83.37%	48.10
	2500	1	97.90%	99.20%	99.52%	83.37%	34.36
	2000	1	97.30%	98.50%	99.31%	83.37%	24.05
	1500	1	96.90%	98.90%	99.31%	83.37%	24.05
	1000	1	97.30%	98.50%	99.31%	83.37%	24.05
	500	1	96.90%	97.90%	99.14%	83.37%	19.24
	250	1	96.50%	96.90%	98.89%	83.37%	15.03

	Αριθμός Νγραμμάτων	λ	Spam Recall	Spam Precision	Weighted Accuracy	Baseline W.Accuracy	TCR
3γράμματα	4000	1	97.70%	98.90%	99.45%	83.37%	30.06
	3500	1	**98.10%**	98.70%	99.48%	83.37%	32.07
	3000	1	**98.10%**	**99.00%**	**99.52%**	83.37%	**34.36**
	2500	1	97.90%	98.90%	99.48%	83.37%	32.07
	2000	1	96.90%	98.70%	99.27%	83.37%	22.90
	1500	1	96.70%	98.50%	99.20%	83.37%	20.91
	1000	1	97.50%	98.30%	99.31%	83.37%	24.05
	500	1	96.50%	98.10%	99.10%	83.37%	18.50
	250	1	94.60%	97.20%	98.65%	83.37%	12.33

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Ion Androutsopoulos, John Koutsias, Konstantinos V. Chandrinos, George Paliouras and Constantine D. Spyropoulos **An Evaluation of Naive Bayesian Anti-Spam Filtering**, CoRR cs.CL/0006013: (2000)

[2] Jiang Tao **SVM FOR SPAM FILTERING,** 2004 www.ntu.edu.sg/home/aswduch/Teaching/Assign-2/JiangTao.pdf

[3] William S. Yerazunis **Sparse Binary Polynomial Hashing and the CRM114 Discriminator,** 2003 Spam Conference

[4] Annie Christian and Andrea Pound **Link-Following for Spam Detection :** http://www.owlnet.rice.edu/~ap/comp527/final_project/spam-paper.pdf December 3 2004

[5] Susan Dumais, John Platt, David Heckerman, Mehran Sahami **Inductive Learning Algorithms and Representations for Text Categorization** CIKM 1998: 148-155

[6] Harris Drucker, Donghui Wu, Vladimir N. Vapnik **Support Vector Machines For Spam Categorization** IEEE Transactions on neural networks, vol. 10, No. 5 September 1999

[7] Rajesh M.Pampapathi, Boris Mirkin, Mark Levene **A Suffix Tree Approach to Text Categorization Applied to Spam Filtering** UKCI05 5-7 September 2005

[8] Doina Caragea, Adrian Silvescu and Vasant Honavar **Incremental and Distributed Learning with Support Vector Machines** AAAI/IAAI 2000: 1067

[9] David Madigan **Statistics and the War on Spam,** 2003 www.stat.rutgers.edu/~madigan/PAPERS/sagtu.pdf

[10] Bayes Theorem http://www.bookrags.com/sciences/mathematics/bayess-theorem-wom.html

[11] SVM*light* Support Vector Machine http://svmlight.joachims.org/

[12] Scott Selikoff **The SVM-Tree Algorithm: A New Method for Handling Multi-Class SVMs** http://scott.selikoff.net/papers/CS678_-_Final_Report.pdf May 12 2003

[13] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin **A Practical Guide to Support Vector Classification,** 2003 www.csie.ntu.edu.tw/~cjlin/papers/**guide**/**guide**.pdf

[14] Tom Fawcett **"In vivo" spam filtering: A challenge problem for data mining** CoRR cs.AI/0405007: (2004)

[15] T. Lauwers, R. Schlender, D. Villa **A Bayesian Approach to Spam Filtering** www.andrew.cmu.edu/user/tlauwers/Bayes.pdf January 30 2004

[16] Konstantin Tretyakov **Machine Learning Techniques in Spam Filtering** Data Mining Problem-oriented Seminar, MTAT.03.177, May 2004, pp. 60-79.

[17] Shabbir Ahmed and Farzana Mithun **Word Stemming to Enhance Spam Filtering ,** 2004 http://www.ceas.cc/papers-2004/167.pdf

[18] Flavio D. Garcia, Jaap-Henk Hoepman, Jeroen van Nieuwenhuizen **Spam Filter Analysis** SEC 2004: 395-410

[19] Johan Hovold **Naive Bayes Spam Filtering Using Word Position Attributes** 15th Nodalida, Joensuu, May 20-21, 2005 http://www.ceas.cc/papers-2005/144.pdf

[20] Nicholas Crisp **Bayesian Spam Filtering** http://www.doc.ic.ac.uk/lab/labsrc_area/msc/projects/Archive02-03/Ex_CONV_PROJECT/nrc02/report.pdf September 2003

[21] Ion Androutsopoulos, Georgios Paliouras, Eirinaios Michelakis **Learning to Filter Unsolicited Commercial E-Mail,** October 2004 www.aueb.gr/users/ion/docs/TR2004_updated.pdf

[22] Shining Light Into the Realtime Blackhole List http://www.oreillynet.com/pub/a/network/2000/06/09/magazine/rbl.html

[23] Using Real-Time Blackhole Lists For Filtering Email http://www.outlookexchange.com/Articles/JohnYoung/article1.asp

[24] Background of spam filtering http://www.usenix.org/events/usenix03/tech/freenix03/full_papers/paganini/paganini_html/node2.html

[25] Ray Ellington **A Look At Spam Filtering And How To Avoid Getting Your Innocent Email Caught In Its Trap** July 20 2004 http://www.giac.org/certified_professionals/practicals/gsec/4015.php

[26] Marco Paganini: **ASK: Active Spam Killer** USENIX Annual Technical Conference, FREENIX Track 2003: 51-62

[27] Distance and Similarity Measures http://www.vias.org/tmdatanaleng/cc_distance_meas.html