Link Prediction in Social Networks

Κωνσταντίνος Δημητρίου
Τμήμα Πληροφορικής
Πανεπιστήμιο Κύπρου
constandinosdemetriou@gmail.com

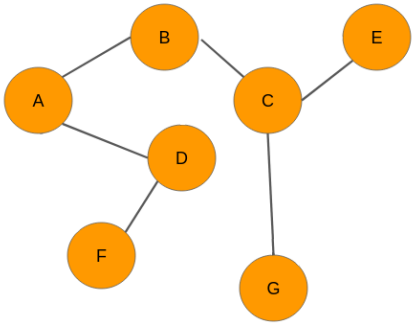
***Abstract* – Αναρωτηθήκατε ποτέ από ποιον μπορεί να προέλθει το επόμενο αίτημα φιλίας στο facebook; Τι θα λέγατε εάν υπήρχε κάποιος τρόπος να το προβλέψουμε; Τα κοινωνικά δίκτυα όπως είναι το facebook μπορούν να αναπαρασταθούν με γραφήματα στα οποία οι χρήστες αντιπροσωπεύουν τους κόμβους και οι σχέσεις μεταξύ των χρηστών τις πλευρές. Το link prediction είναι ένα από τα σημαντικότερα ερευνητικά θέματα στην περιοχή των γραφημάτων και των δικτύων. Ο στόχος του link prediction είναι ο εντοπισμός ζευγών κόμβων που θα σχηματίσουν έναν σύνδεσμο στο μέλλον. Στην παρούσα εργασία χρησιμοποιήσαμε την τοπολογία μη-κατευθυνόμενων γραφημάτων από κοινωνικά δίκτυα για να εξάγουμε features με τα οποία εκπαιδεύσαμε αλγόριθμους classification supervised learning.**

Key words – Link prediction, Social networks, Undirect graph, Features, Jaccard's coefficient, Adamic/Adar, Preferential attachment, Clustering coefficient, Correlation Analysis, Machine learning, Supervised learning, Accuracy, Cross validation, Grid search.

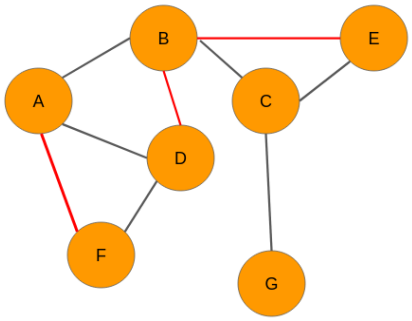
1. Introduction

Το link prediction έχει πολλές χρήσεις σε πραγματικές εφαρμογές. Μερικές σημαντικές περιπτώσεις της χρήσης του link prediction είναι: α) η πρόβλεψη ποιοι πελάτες είναι πιθανό να αγοράσουν ποια προϊόντα σε διαδικτυακές αγορές όπως το Amazon. Επίσης, μπορεί να βοηθήσει σε καλύτερο recommendation των προϊόντων β) η πρόταση αλληλεπιδράσεων ή συνεργασιών μεταξύ των εργαζομένων σε έναν οργανισμό γ) εξαγωγή πληροφοριών ζωτικής σημασίας από τρομοκρατικά δίκτυα.

Στη θεωρία γραφημάτων, link prediction είναι το πρόβλημα της πρόβλεψης της ύπαρξης μελλοντικής σύνδεσης μεταξύ δύο οντοτήτων σε ένα δίκτυο. Το Σχήμα 1 παρουσιάζει ένα μη-κατευθυνόμενο γράφημα σε ένα δεδομένο χρόνο t. Το Σχήμα 2 παρουσιάζει το ίδιο γράφημα σε χρόνο t+n. Παρατηρούμε ότι σε μελλοντικό χρόνο t+n προστέθηκαν οι 3 πλευρές μεταξύ των κόμβων οι οποίες απεικονίζονται με κόκκινο χρώμα. Στόχος μας στην παρούσα εργασία είναι να προβλέψουμε αυτές τις πλευρές που θα προστεθούν στο γράφημα.



Σχήμα 1: Παράδειγμα γραφήματος σε χρόνο *t*



Σχήμα 2: Παράδειγμα γραφήματος σε χρόνο *t+n*

Εάν με κάποιο τρόπο μπορούσαμε να αντιπροσωπεύσουμε ένα γράφημα με τη μορφή ενός δομημένου dataset με ένα σύνολο από features, τότε θα μπορούσαμε να χρησιμοποιήσουμε αλγορίθμους machine learning για να προβλέψουμε το σχηματισμό συνδέσμων μεταξύ των μη συνδεδεμένων ζευγών κόμβων του γραφήματος.

2. Related Work

2.1. The Link-Prediction Problem for Social Networks

Οι Liben-Nowell και Kleinberg [1] έκαναν μια επισκόπηση των similarity-based μεθόδων για link prediction χρησιμοποιώντας ένα δίκτυο co-authorship. Συγκρίναν τη σχετική αποτελεσματικότητα των μετρικών που σχετίζονται με similarity για το δίκτυο. Το άρθρο ταξινομεί τις μετρικές σε δύο κατηγορίες: (1) μέθοδοι που βασίζονται σε γειτνιάσεις κόμβων όπως Common Neighbors, Jaccard's Coefficient, Adamic/Adar και Preferential attachment, (2) μέθοδοι που βασίζονται στο σύνολο όλων των διαδρομών μεταξύ δύο κόμβων όπως Katz, PageRank και SimRank. Η συγκεκριμένη δουλεία απέδειξε ότι δεν υπάρχει ξεκάθαρος νικητής μεταξύ αυτών των μεθόδων και η επίδοση εξαρτάται από το dataset αλλά ένα σημαντικός αριθμός τεχνικών έχει πολύ καλές αποδόσεις.

2.2. Link Prediction using Supervised Learning

Οι Hasan et al. [2] εξέτασαν το link prediction σε co-authorship δίκτυα για τη βιολογία και την πληροφορική χρησιμοποιώντας supervised learning. Οι συγγραφείς παρήγαγαν διάφορα proximity features για κάθε κόμβο (πχ keyword match count), aggregated features (πχ sum of papers, sum of neighbors, sum of keyword count, sum of classification code) και topological features (πχ shortest distance). Επίσης, συγκρίναν την απόδοση διαφορετικών μοντέλων, όπως SVM, K-nearest neighbors, Naive Bayes και στα δύο dataset και διαπίστωσαν ότι το SVM έχει την υψηλότερη ακρίβεια στα δύο σύνολα δεδομένων τα οποία χρησιμοποίησαν. Παρατήρησαν ότι παρά την ικανοποιητική ακρίβεια που πέτυχαν τα μοντέλα, οι μέθοδοι supervised learning μπορεί να είναι δύσκολο να επεκταθούν σε μεγάλα γραφήματα καθώς ο χρόνος εκπαίδευσης αυξάνεται.

2.3. Graph-based Features for Supervised Link Prediction

Οι Cukierski et al.[3] αντιμετώπισαν την πρόκληση του IJCNN Social Network για να διαχωρίσει τις πραγματικές πλευρές από τις ψεύτικες από ένα σύνολο 8960 πλευρών δειγματοληψίας από ένα ανώνυμο, κατευθυνόμενο γράφημα που απεικονίζει ένα υποσύνολο σχέσεων στο Flickr. Για εξαγωγή features, το μοντέλο αναπτύσσει μια μεγάλη ποικιλία τεχνικών, συμπεριλαμβανομένης της εξαγωγής τοπικών υπογραφημάτων που σχετίζονται με τους εν λόγω κόμβους, χρησιμοποιώντας SVD, kNN, EdgeRank κλπ καθώς και παραδοσιακές μετρικές similarity όπως Common Neighbors, Jaccard κλπ. Τέλος, εκτελούν repeated classification χρησιμοποιώντας τις posterior πιθανότητες από τα Random Forest.

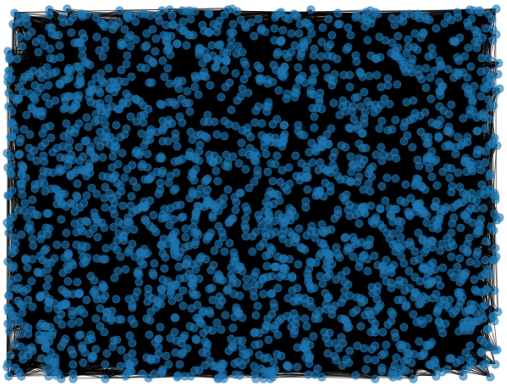
3. Data

3.1. Origin Data

Στην παρούσα εργασία χρησιμοποιηθήκαν datasets με μη-κατευθυνόμενα γραφήματα από τις βάσεις δεδομένων SNAP και KONECT. Επιλέχθηκαν γραφήματα που αφορούν κοινωνικά δίκτυα και συγκεκριμένα: α) *hamsterster*: αυτό το δίκτυο περιέχει φιλίες και οικογενειακούς συνδέσμους μεταξύ των χρηστών του website hamsterster.com με τους κόμβους να αναπαριστούν τους χρήστες και τις πλευρές τη σχέση φιλίας μεταξύ των χρηστών, β) *twitch*: είναι δίκτυο χρηστών του Twitch που αποτελείται από gamers οι οποίοι που μεταδίδουν stream σε μια συγκεκριμένη γλώσσα με τους κόμβους να είναι οι ίδιοι οι χρήστες και οι πλευρές να είναι οι φιλίες μεταξύ τους, γ) *github*: είναι ένα μεγάλο δίκτυο από developers όπου οι κόμβοι είναι οι developers που έχουν αστέρι σε τουλάχιστον 10 repositories και πλευρές είναι αμοιβαίες σχέσεις μεταξύ τους, δ) *deezer*: είναι μια υπηρεσία music streaming όπου οι κόμβοι αντιπροσωπεύουν τους χρήστες και οι πλευρές είναι οι αμοιβαίες φιλίες, ε) *erdos*: ένα τυχαίο γράφημα. Ο Πίνακας 1 παρουσιάζει τις βασικές πληροφορίες για τα datasets όπως ο αριθμός των κόμβων, ο αριθμός των πλευρών και το average degree τους.

Dataset	Number of nodes	Number of edges	Average degree
hamsterster	2426	16631	13.71
twitch	9498	153138	32.25
github	37700	289003	15.33
deezer	54573	498202	18.26
facebook	63731	817035	25.64
erdos	9967	30000	6.02

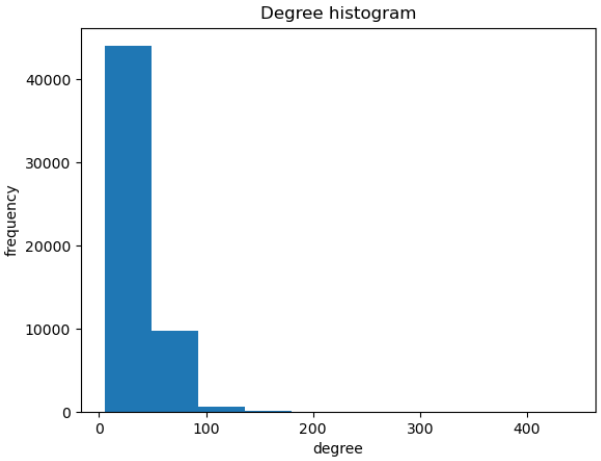
Πίνακας 1: Πληροφορίες για τα datasets



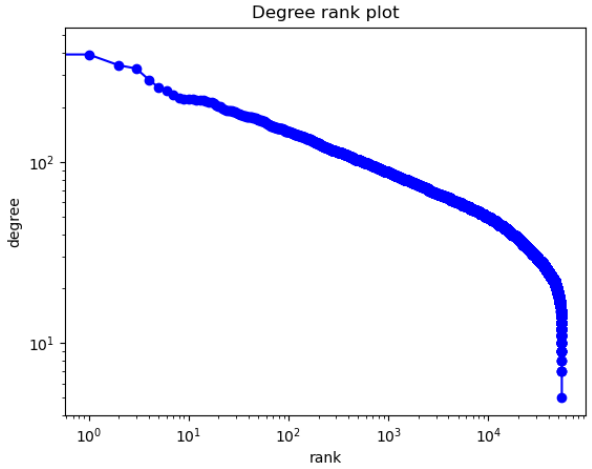
Σχήμα 3: Γράφημα για το δίκτυο *hamsterster*

3.2. Graph Analysis

Αρχικά, αναλύσαμε την κατανομή του degree στο γράφο όπως φαίνεται στο Σχήμα 4 και στο Σχήμα 5. Όπως φαίνεται για το γράφο του *deezer* οι περισσότεροι κόμβοι έχουν degree μικρότερο του 100 και πολύ λίγοι έχουν degree μεγαλύτερο του 100. Αυτό μας οδηγεί στο συμπέρασμα ότι το δίκτυο ακολουθεί κατανομή heavy-tailed. Στις κατανομές heavy-tailed υπάρχουν πολλοί κόμβοι με μικρό degree και λίγοι κόμβοι με μεγάλο degree. Αυτό συμβαίνει και στα περισσότερα dataset με πραγματικά δεδομένα. Όσον αφορά τα datasets που επιλέχθηκαν σε αυτή την εργασία δημιουργήθηκαν οι αντίστοιχες γραφικές παραστάσεις και παρατηρήσαμε ότι ακολουθούν κατανομές heavy-tailed.



Σχήμα 4: Degree histogram για το δίκτυο *deezer*



Σχήμα 5: Degree distribution για το δίκτυο *deezer*

3.3. Dataset Construction

Όπως ήδη έχουμε αναφέρει για να λύσουμε το πρόβλημα του link prediction θα χρησιμοποιήσουμε supervised leaning. Κατά συνέπεια θα πρέπει να εκπαιδεύσουμε τα μοντέλα μας με features τόσο από πλευρές που υπάρχουν στο γράφημα (positive edges) όσο και με features από πλευρές που δεν υπάρχουν στο γράφημα (negative edges). Αυτό εξ υπακούει ότι θα πρέπει στα υφιστάμενα dataset να προσθέσουμε πλευρές που δεν υπάρχουν (negative edges) στο γράφο. Στην παρούσα εργασία αποφασίσαμε να προσθέσουμε τυχαίες πλευρές που δεν υπάρχουν στο γράφο (negative edges), ο αριθμός των οποίων να ισούται με τον αριθμό των πλευρών που υπάρχουν στο γράφο (positive edges). Επίσης,

χρησιμοποιήσαμε το heuristic «για να προστεθεί μια negative edge θα πρέπει το shortest path μεταξύ των κόμβων που συνδέεται να είναι μεγαλύτερο του 2». Η λογική για αυτό το heuristic είναι ότι με αυτόν τον τρόπο ίσως καταφέρουμε τα features που θα δημιουργηθούν για τις negative edges να διαφοροποιούνται από αυτά των positive edges και να γίνει πιο εύκολα το classification. Ο Πίνακας 2 παρουσιάζει ένα υποσύνολο του dataset που θα δημιουργηθεί με βάση το γράφο παρουσιάζεται στην Εικόνα 1.

Edge	Positive / Negative	Target
A-B	Positive	1
A-D	Positive	1
B-C	Positive	1
A-E	Negative	0
A-G	Negative	0
B-F	Negative	0
…	…	…

Πίνακας 2: Παράδειγμα δημιουργίας dataset με positive και negative edges με βάση το γράφο που παρουσιάζεται στην Εικόνα 1

4. Features

4.1. Feature Extraction

Σε αυτή την ενότητα θα παρουσιάσουμε μια σειρά από μεθόδους οι οποίες θα εξάγουν features με βάση την τοπολογία ενός δεδομένου γράφου. Όλοι οι μέθοδοι δημιουργούν ένα *score(x,y)* για όλες τις πλευρές <*x*,*y*> που έχουμε στο dataset όπου *x* και *y* είναι κόμβοι στο γράφο. Για έναν κόμβο *x*, ας ορίσουμε *Γ(x)* το σύνολο των γειτόνων του *x* στο γράφημα.

4.1.1. Jaccard's coefficient

Ο Jaccard's coefficient είναι ένας συντελεστής που συχνά χρησιμοποιείται για μέτρηση της ομοιότητας σε information retrieval. Μετρά την πιθανότητα και ο *x* και ο *y* να έχουν ένα χαρακτηριστικό *f*, για ένα τυχαία επιλεγμένο χαρακτηριστικό *f* που έχει είτε ο *x* είτε ο *y*.

$$score(x,y) = \frac{\Gamma(x) \cap \Gamma(y)}{\Gamma(x) \cup \Gamma(y)}$$

4.1.2. Adamic Adar

Η μετρική Adamic Adar χρησιμοποιείται για link prediction στα κοινωνική δίκτυα και ανάλογα με τον αριθμό των κοινών συνδέσμων μεταξύ δύο κόμβων. Ορίζεται ως το άθροισμα του αντίστροφου λογαριθμικού degree centrality των γειτόνων που μοιράζονται οι δύο κόμβοι. Ο ορισμός βασίζεται στην ιδέα ότι κοινά στοιχεία με πολύ μεγάλες γειτονιές είναι λιγότερο σημαντικά κατά την πρόβλεψη σύνδεσης μεταξύ δύο κόμβων σε σύγκριση με στοιχεία που μοιράζονται μεταξύ μικρού αριθμού κόμβων.

$$score(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log |\Gamma(z)|}$$

4.1.3. Preferential attachment

Η βασική προϋπόθεση στη μετρική preferential attachment είναι η πιθανότητα μια νέα πλευρά να έχει τον κόμβο *x* ως τελικό σημείο είναι ανάλογη με το |Γ(x)| δηλαδή τον τρέχοντα αριθμό γειτόνων του *x*. Κατά συνέπεια η πιθανότητα ύπαρξης πλευράς μεταξύ των *x* και *y* συσχετίζεται με το γινόμενο του αριθμού των γειτόνων του *x* και *y*.

$$score(x,y) = |\Gamma(x)| \times |\Gamma(y)|$$

4.1.4. Clustering coefficient

Ένας clustering coefficient είναι μια μετρική του βαθμού στον οποίο οι κόμβοι σε ένα γράφημα τείνουν να ομαδοποιούνται μαζί. Τα στοιχεία δείχνουν ότι στα περισσότερα δίκτυα πραγματικού κόσμου, και ιδίως στα κοινωνικά δίκτυα, οι κόμβοι τείνουν να δημιουργούν σφιχτά δεμένες ομάδες που χαρακτηρίζονται από σχετικά υψηλή πυκνότητα δεσμών. Αυτή η πιθανότητα τείνει να είναι μεγαλύτερη από τη μέση πιθανότητα ενός δεσμού που δημιουργείται τυχαία μεταξύ δύο κόμβων. Το clustering για έναν κόμβο z δίνεται από την εξίσωση:

$$c(z) = \frac{2T(z)}{|\Gamma(z)| \times (|\Gamma(z)| - 1)}$$

όπου $T(z)$ είναι ο αριθμός των τριγώνων μέσω του κόμβου *z*.

$$score(x,y) = c(x) \times c(y)$$

4.2. Correlation Analysis

Το correlation είναι μια διμερής ανάλυση που μετρά τη δύναμη της συσχέτισης μεταξύ δύο μεταβλητών και της κατεύθυνσης της σχέσης. Όσον αφορά την ισχύ της σχέσης, η τιμή του συντελεστή συσχέτισης κυμαίνεται μεταξύ +1 και -1. Η τιμή ± 1 δείχνει τον τέλειο βαθμό συσχέτισης μεταξύ των δύο μεταβλητών. Καθώς η τιμή του συντελεστή συσχέτισης πηγαίνει προς το 0, η σχέση μεταξύ των δύο μεταβλητών είναι πιο αδύναμη. Η κατεύθυνση της σχέσης υποδεικνύεται από το σύμβολο του συντελεστή. Το σύμβολο + υποδηλώνει θετική σχέση δηλαδή ευθέως ανάλογη σχέση στην οποία με τη αύξηση της μια μεταβλητής αυξάνεται και η άλλη μεταβλητή. Αντίθετα το σύμβολο - υποδηλώνει αρνητική σχέση δηλαδή αντιστρόφως ανάλογη σχέση στην οποία με τη αύξηση της μια μεταβλητής μειώνεται η άλλη. Το Σχήμα 6 παρουσιάζει το Correlation Analysis μεταξύ των features που προέκυψαν για το δίκτυο του twitch. Στην παρούσα εργασία δεν παρατηρήσαμε ξεκάθαρες ισχυρές συσχετίσεις κάποιου feature με το class για όλα τα δίκτυα γι' αυτό αποφασίσαμε να χρησιμοποιήσουμε αλγορίθμους machine learning οι οποίοι θα χρησιμοποιούν όλα τα feature και θα κτίσουν τα μοντέλα πρόβλεψης αυτόματα.

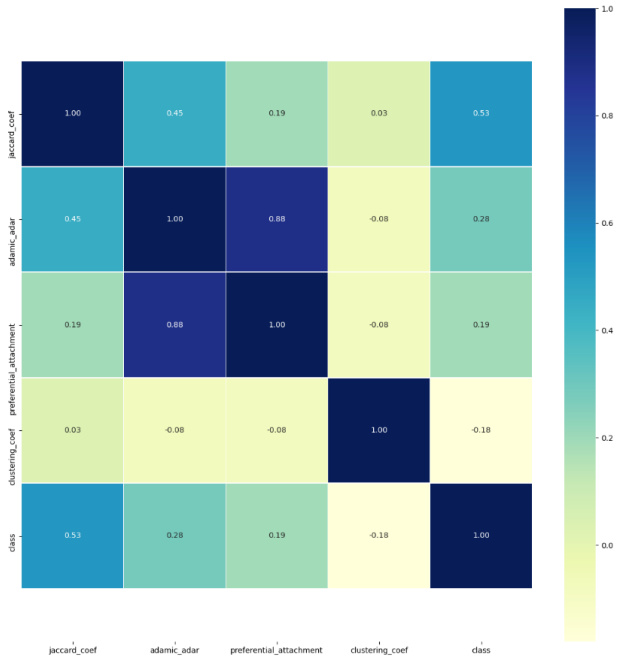
4.3. Standardize features

Για να βοηθήσουμε τα μοντέλα να μάθουν καλύτερα κανονικοποίησαμε τα features αφαιρώντας τον μέσο όρο και διαιρώντας με την τυπική απόκλιση. Το κανονικοποιημένο score ενός δείγματος *x* υπολογίζεται ως εξής:

$$z = \frac{x - u}{s}$$

όπου *u* είναι ο μέσος όρος των δειγμάτων και *s* είναι η τυπική απόκλιση των δειγμάτων. Η κανονικοποίηση εκτελείται ανεξάρτητα σε κάθε feature υπολογίζοντας τα σχετικά

στατιστικά στοιχεία για τα δείγματα. Στη συνέχεια αποθηκεύονται ο μέσος όρος και η τυπική απόκλιση για τα δεδομένα και χρησιμοποιούνται μεταγενέστερα στο μετασχηματισμό. Η κανονικοποίηση ενός συνόλου δεδομένων είναι μια κοινή απαίτηση για πολλά μοντέλα machine learning. Ενδέχεται τα μοντέλα να συμπεριφέρονται άσχημα εάν τα μεμονωμένα features δεν είναι κανονικοποιημένα.



Σχήμα 6: Correlation Analysis μεταξύ των features που προέκυψαν για το δίκτυο του twitch

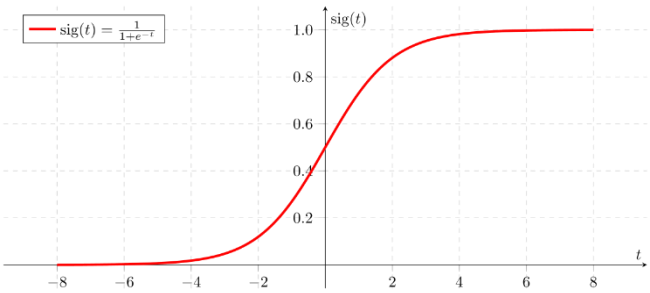
5. Supervised binary classification

5.1. Logistic Regression

Ο αλγόριθμος Logistic Regression είναι ένας αλγόριθμος supervised learning που χρησιμοποιείται για να λύσει προβλήματα classification. Ο αλγόριθμος Logistic Regression χρησιμοποιεί μια sigmoid function η οποία δίνεται από τη σχέση:

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

Η υπόθεση του Logistic Regression περιορίζει τη συνάρτηση μεταξύ του 0 και του 1. Θα χρησιμοποιούμε το sigmoid function για να χαρτογραφήσουμε τις προβλέψεις στις πιθανότητες.



Σχήμα 7: Γραφική παράσταση sigmoid function

Όταν χρησιμοποιούμε Linear Regression η σχέση για την υπόθεση είναι:

$$h\Theta(x) = \beta_0 + \beta_1 X$$

Στο Logistic Regression αναμένουμε ότι η υπόθεση μας θα μας δώσει τιμές μεταξύ του 0 και του 1 οπότε:

$$h\Theta(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

Το cost function αναπαριστά τον optimization στόχο δηλαδή θέλουμε να δημιουργούμε ένα cost function και να το ελαχιστοποιούμε ώστε να μπορέσουμε να αναπτύξουμε ένα ακριβές μοντέλο με ελάχιστο error. Το cost function δίνεται από τη σχέση:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

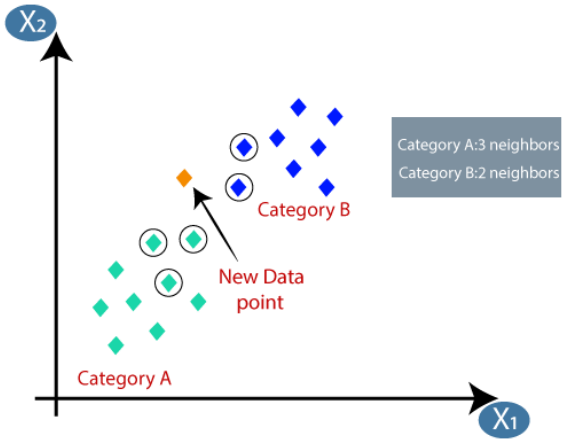
όπου το $y^{(i)}$ είναι το predicted output και $h_\theta(x^{(i)})$ είναι η υπόθεση function. Χρησιμοποιούμε τη μέθοδο του Gradient Descent για να βρούμε την ελάχιστη τιμή του cost.

5.2. k-Nearest Neighbors (kNN)

Ο αλγόριθμος k-Nearest Neighbors είναι ένας απλός αλγόριθμος supervised learning που χρησιμοποιείται για να λύσει προβλήματα classification ή regression. Ο αλγόριθμος kNN υποθέτει ότι παρόμοια πράγματα βρίσκονται κοντά το ένα στο άλλο. Ο kNN χρησιμοποιεί την ιδέα του similarity (μερικές φορές ονομάζεται και distance, proximity, ή closeness) για να υπολογίσουμε την απόσταση μεταξύ σημείων σε ένα γράφημα. Υπάρχουν πολλές μέθοδοι υπολογισμού της απόστασης και ένας τρόπος μπορεί να είναι προτιμητέος από κάποιον άλλο τρόπο ανάλογα με το πρόβλημα που επιλύουμε. Τέτοιες μέθοδοι είναι euclidian, minkowski και manhattan distance. Για κάθε νέο query υπολόγιζει το distance του με όλα τα vectors που υπάρχουν στο train set. Στη συνέχεια με βάση το distance επιλέγει τους k κοντινότερους γείτονες. Τέλος, κατατάσσει το query στην κατηγορία της πλειοψηφίας των γειτόνων. Το Σχήμα 8 παρουσιάζει την εισαγωγή ενός νέου query το οποίο θα ταξινομηθεί με βάση τον αλγόριθμο nearest neighborhoods με k = 5. Από τους 5 κοντινότερους γείτονες οι 3 ταξινομούνται στην κατηγορία A και οι άλλοι 2 στην κατηγορία B με αποτέλεσμα το query να ταξινομηθεί στην κατηγορία A στην οποία ανήκει και η πλειοψηφία των γειτόνων.

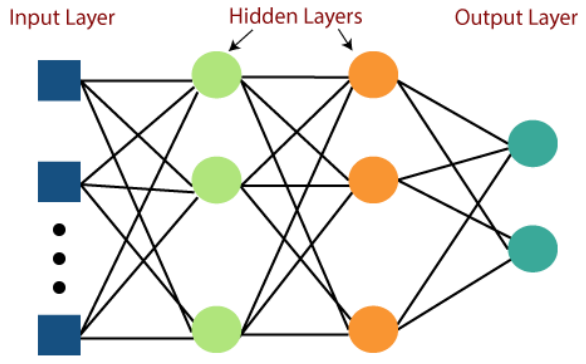
5.3. Multilayer Perceptron (MLP)

Ο αλγόριθμος Multilayer Perceptron Classifier είναι ένας αλγόριθμος supervised learning ο οποίος βασίζει τη λειτουργία του στα Τεχνητά Νευρωνικά Δίκτυα. Ο αλγόριθμος εκπαιδεύεται με ένα σύνολο εισόδων και τις επιθυμητές τους εξόδους και δυνητικά μπορεί να μάθει μια κατά προσέγγιση μη γραμμική συνάρτηση για classification ή regression. Η αρχιτεκτονική του δικτύου αποτελείται από ένα επίπεδο εισόδου (input layer), ένα ή περισσότερα μη γραμμικά επίπεδα τα οποία ονομάζονται κρυφά επίπεδα (hidden layer) και τέλος ένα επίπεδο εξόδου (output layer).



Σχήμα 8: Παράδειγμα εισαγωγής νέου query το οποίο θα ταξινομηθεί με τον αλγόριθμο 5-nearest neighborhoods

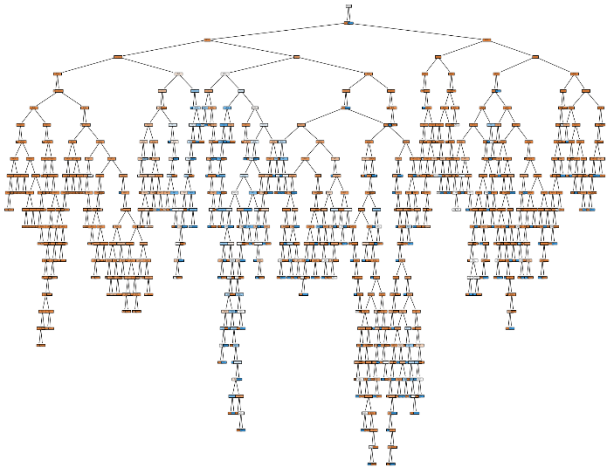
Το επίπεδο εισόδου δεν είναι ενεργό καθώς απλά διοχετεύει τα χαρακτηριστικά (features) στους νευρώνες του επόμενου επιπέδου. Οι νευρώνες συνδέονται μεταξύ τους με τις συνάψεις. Κάθε σύναψη χαρακτηρίζεται από ένα βάρος (weight) η τιμή του οποίου μεταβάλλεται κατά τη διάρκεια της εκπαίδευσης. Κάθε νευρώνας σε κάθε ενεργό επίπεδο μετατρέπει τις τιμές που λαμβάνει από το προηγούμενο επίπεδο σε ένα weighted sum το οποίο διοχετεύεται σε μια μη γραμμική συνάρτηση ενεργοποίησης (πχ sigmoid function ή hyperbolic tan function). Για προβλήματα ταξινόμησης όπως είναι αυτά που έχουμε να λύσουμε σε αυτή την εργασία ο αλγόριθμος multilayer perceptron χρησιμοποιεί τη μέθοδο ανάστροφης μετάδοσης σφάλματος (backpropagation) κατά την εκπαίδευση. Αυτή η μέθοδος βασίζεται στην προσπάθεια να προσαρμόσουμε τα βάρη έτσι ώστε να ελαχιστοποιήσουμε τη τετραγωνική συνάρτηση σφάλματος. Αυτό επιτυγχάνεται με τη μέθοδο Gradient Descent οπόταν και η αλλαγή των βαρών είναι ανάλογη ως προς το αρνητικό της κλίσης της συνάρτησης του σφάλματος σε ένα σημείο δηλαδή είναι ανάλογη με το αρνητικό της παραγώγου της συνάρτησης του σφάλματος ως προς τα βάρη. Εν τέλη, ο αλγόριθμος δε χρειάζεται περισσότερα από τρία ενεργά επίπεδα για να λύσει οποιοδήποτε μη γραμμικά διαχωρίσιμο πρόβλημα εφόσον μπορεί να σχηματίσει οποιαδήποτε κυρτή επιφάνεια (με 1 κρυφό επίπεδο) ή οποιαδήποτε αυθαίρετη περιοχή (με 2 κρυφά επίπεδα). Το Σχήμα 9 παρουσιάζει μια γραφική αναπαράσταση ενός Multilayer Perceptron δίκτυο.



Σχήμα 9: Γραφική αναπαράσταση ενός Multilayer Perceptron δίκτυο

5.4. Decision Tree

Ο αλγόριθμος Decision Tree Classifier είναι ένας αλγόριθμος supervised learning που χρησιμοποιείται για να λύσει προβλήματα classification ή regression. Τα δέντρα αποφάσεων αναφέρονται σε ένα ιεραρχικό μοντέλο αποφάσεων καθώς και των συνέπειών τους. Ο στόχος, είναι να δημιουργηθούν μοντέλα τα οποία να προβλέπουν την τιμή μιας εισόδου δεδομένων με τη μάθηση απλών κανόνων απόφασης που προκύπτουν από τα χαρακτηριστικά (features) των δεδομένων έτσι ώστε σε άγνωστα δεδομένα να ακολουθήσει τη στρατηγική με την οποία έχει τη μεγαλύτερη πιθανότητα για να επιτύχει το στόχο του. Για προβλήματα ταξινόμησης όπως είναι αυτά που έχουμε να λύσουμε σε αυτή την εργασία ένα δέντρο απόφασης αναφέρεται ως ένα δέντρο ταξινόμησης. Το δέντρο αποφάσεων αποτελείται από κόμβους που σχηματίζουν ένα δέντρο με ρίζα. Σε κάθε φύλλο έχει αντιστοιχηθεί μία κατηγορία η οποία αναπαριστά την κατάλληλη έξοδο. Τα γεγονότα ταξινομούνται με βάση τη διαδρομή από τη ρίζα του δέντρου σε ένα φύλλο, σύμφωνα με τα αποτελέσματα των δοκιμών κατά μήκος της διαδρομής. Ο τρόπος με τον οποίο κτίζεται το δέντρο είναι: (1) ορίζεται ως ρίζα ο κόμβος με το καλύτερο χαρακτηριστικό από τα παραδείγματα (2) εάν για μια τιμή αυτού του χαρακτηριστικού για όλα τα παραδείγματα που έχουν αυτή την τιμή η έξοδος είναι ίδια τότε δημιούργησε φύλλο με αυτή την έξοδο (3) διαφορετικά δημιούργησε ένα υποδέντρο αναδρομικά επιλέγοντας το αμέσως επόμενο καλύτερο χαρακτηριστικό. Το καλύτερο χαρακτηριστικό είναι εκείνο που με τη διάσπαση των παραδειγμάτων οδηγεί σε όσο το δυνατό μεγαλύτερη μείωση της εντροπίας. Το Σχήμα 10 παρουσιάζει το decision tree που δημιουργήθηκε μετά την εκπαίδευση για το δίκτυο hamsterster.

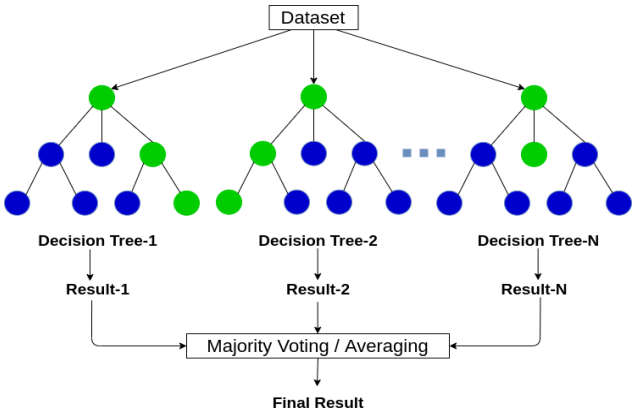


Σχήμα 10: Το decision tree που δημιουργήθηκε μετά την εκπαίδευση για το δίκτυο hamsterster

5.5. Random Forest

Τα Random Forest είναι ένας αλγόριθμος supervised learning που χρησιμοποιείται για να λύσει προβλήματα classification ή regression. Όπως υποδηλώνει το όνομά του, αποτελούνται από ένα μεγάλο αριθμό μεμονωμένων decision trees που λειτουργούν ως σύνολο. Κάθε μεμονωμένο δέντρο στο Random Forest παράγει μια πρόβλεψη για ένα class και το class με τις περισσότερες ψήφους γίνεται η πρόβλεψη του μοντέλου μας. Ο λόγος για τον οποίο τα Random Forest

λειτουργούν τόσο καλά είναι ότι τα δέντρα προστατεύουν το ένα το άλλο από τα ατομικά τους λάθη (αρκεί να μην κάνουν λάθος όλα στην ίδια κατεύθυνση). Το Random Forest προσθέτει επιπλέον τυχαιότητα στο μοντέλο, ενώ κτίζει τα δέντρα. Αντί να αναζητά το πιο σημαντικό χαρακτηριστικό όταν διαχωρίζει έναν κόμβο, αναζητά το καλύτερο χαρακτηριστικό σε ένα τυχαίο υποσύνολο χαρακτηριστικών. Αυτό οδηγεί σε μια μεγάλη ποικιλία που γενικά οδηγεί σε ένα καλύτερο μοντέλο. Το Σχήμα 11 παρουσιάζει τον τρόπο λειτουργίας των Random Forest.



Σχήμα 11: Τρόπος λειτουργίας Random Forest

5.6. Gaussian Naive Bayes

Το Naive Bayes είναι ένας αλγόριθμος supervised learning που χρησιμοποιείται για να λύσει προβλήματα classification. Ο αλγόριθμος βασίζεται στη θεωρία των πιθανοτήτων. Ας ξεκινήσουμε με το εξής ερώτημα «Δεδομένου ενός σημείου *x*, ποια είναι η πιθανότητα του *x* να ανήκει σε μια κλάση *c*;». Ο Naive Bayes classifier προσπαθεί να υπολογίσει αυτές τις πιθανότητες απευθείας. Επομένως, δεδομένου ενός σημείου *x*, θέλουμε να υπολογίσουμε τη *p(c | x)* για όλες τις κλάσεις *c* και η έξοδος είναι το *c* με τη μεγαλύτερη πιθανότητα. Αυτό μπορεί να γραφτεί ως εξής:

$$prediction(x) = \arg\max p(c \mid x)$$

όπου το $\max p(c \mid x)$ επιστρέφει τη μέγιστη πιθανότητα ενώ το $\arg\max p(c \mid x)$ επιστέφει το c με τη ψηλότερη πιθανότητα. Μπορούμε να υπολογίσουμε το *p(c | x)*, με το θεώρημα του Bayes:

$$p(c \mid x) = \frac{p(x \mid c) \cdot p(c)}{p(x)} = \frac{p(x \mid c) \cdot p(c)}{\sum_c p(x \mid c) \cdot p(c)}$$

Πως υπολογίζουμε το *p(x | c)* και *p(c)*; Αυτό είναι το θέμα της εκπαίδευσης του Bayes classifier. Ο απλούστερος τρόπος για υπολογίσουμε το *p(c)* είναι να υπολογίσουμε τις σχετικές συχνότητες των κλάσεων και να τις χρησιμοποιήσουμε σαν πιθανότητες. Για να υπολογίσουμε τη πιθανότητα *p(x | c)* θα χρειαστεί να κάνουμε τη naive υπόθεση ότι τα features *x1, x2* είναι στοχαστικά ανεξάρτητα δεδομένου του *c*.

$$p(x_1, x_2 \mid c) = p(x_1 \mid c) \cdot p(x_2 \mid c)$$

Από αυτό το μέρος προέρχεται η naive προσέγγιση του Bayes επειδή αυτή η εξίσωση δεν ισχύει γενικά. Για να υπολογίσουμε τη πιθανότητα *p(x | c)* θα χρησιμοποιήσουμε τη Gaussian κατανομή η οποία δίνεται από τη σχέση:

$$p(x_i \mid c_j) = \frac{1}{\sqrt{2\pi\sigma_{i,j}^2}} e^{-\frac{1}{2}\left(\frac{x_i - \mu_{i,j}}{\sigma_{i,j}}\right)^2} \text{ for } i = 1, 2 \text{ and } j = 1, 2, 3$$

όπου $\mu_{i,j}$ είναι ο μέσος και $\sigma_{i,j}$ είναι το standard deviation.

6. Evaluation

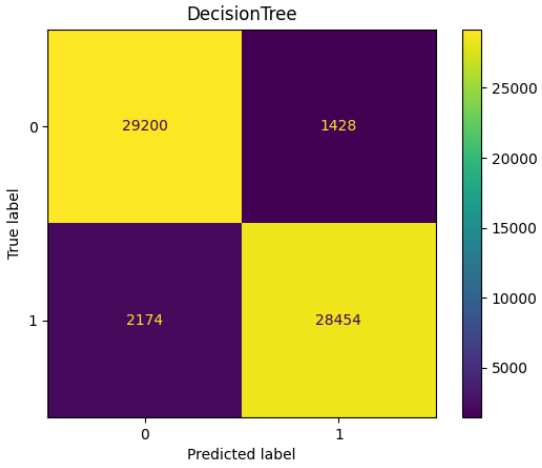
6.1. Accuracy

Για την αξιολόγηση της επίδοσης των αλγορίθμων machine learning χρησιμοποιήθηκε η μετρική του *accuracy* σε προβλέψεις με άγνωστα δεδομένα δηλαδή δεδομένα που δεν χρησιμοποιήθηκαν κατά την εκπαίδευση.

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

Ένα *true positive (TP)* είναι το αποτέλεσμα όπου το μοντέλο προβλέπει σωστά μια positive edge. Ομοίως, ένα *true negative (TN)* είναι ένα αποτέλεσμα όπου το μοντέλο προβλέπει σωστά μια negative edge. Αντίθετα, ένα *false positive (FP)* είναι ένα αποτέλεσμα όπου το μοντέλο προβλέπει εσφαλμένα μια positive edge. Αντίστοιχα, ένα *false negative (FN)* είναι το αποτέλεσμα όπου το μοντέλο προβλέπει εσφαλμένα μια negative edge. Το Σχήμα 12 παρουσιάζει ένα παράδειγμα *confusion matrix*. Οι πιο πάνω πληροφορίες συνήθως βρίσκονται σε ένα *confusion matrix* το οποίο έχει τη μορφή:

$$confusion\ matrix = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix}$$



Σχήμα 12: Confusion matrix για τις προβλέψεις του αλγορίθμου decision tree στο δίκτυο twitch

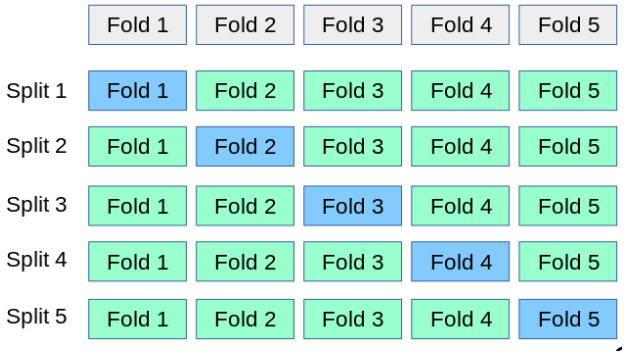
6.2. Cross validation

Το cross validation είναι μια μέθοδος που χρησιμοποιείται για την αξιολόγηση της αποτελεσματικότητας machine learning μοντέλων. Συνήθως χρησιμοποιείται για σύγκριση και αξιολόγηση διάφορων μοντέλων machine learning και για επιλογή του καλύτερου. Είναι επίσης μια διαδικασία επανα-δειγματοληψίας που χρησιμοποιείται για την αξιολόγηση ενός μοντέλου εάν έχουμε περιορισμένα δεδομένα, βοηθώντας μας να βεβαιωθούμε ότι το μοντέλο

δεν θα πάθει overfitting. Έχει μια μόνο παράμετρο, το k, που αναφέρεται στον αριθμό ομάδων (folds) στις οποίες θα διαιρεθεί ένα σύνολο δεδομένων. Γι' αυτό η διαδικασία αυτή συχνά ονομάζεται k-fold cross-validation. Η διαδικασία που ακολουθεί είναι η εξής:

1. Σπάζουμε το dataset σε k folds (ομάδες).
2. Επιλέγουμε ένα fold και το χρησιμοποιούμε σαν test set.
3. Χρησιμοποιούμε τα υπόλοιπα k-1 folds σαν train set και εκπαιδεύουμε το μοντέλο.
4. Υπολογίζουμε το accuracy του εκπαιδευμένου μοντέλου στο test set.
5. Επαναλαμβάνουμε αυτήν τη διαδικασία έως ότου κάθε k-fold χρησιμοποιηθεί σαν test set.
6. Τέλος, υπολογείζουμε το average accuracy (μέση ακρίβεια) και το standard deviation (τυπική απόκλιση). Αυτές θα είναι οι μετρήσεις για την απόδοση του μοντέλου.

Αυτό σημαίνει ότι κάθε ομάδα έχει την ευκαιρία να χρησιμοποιηθεί σαν test set 1 φορά και να χρησιμοποιηθεί για την εκπαίδευση του μοντέλου k-1 φορές. Το Σχήμα 13 παρουσιάζει το διαχωρισμό που θα γίνει στα δεδομένα στην περίπτωση 5-fold cross validation.



Σχήμα 13: Εικονική αναπαράσταση 5-fold cross validation

6.3. Grid Search

Ένα μοντέλο machine learning έχει πολλές παραμέτρους που δεν εκπαιδεύονται από το train set. Αυτές οι παράμετροι είναι πολύ σημαντικές εφόσον ελέγχουν την ακρίβεια του μοντέλου. Για παράδειγμα, ο ρυθμός μάθησης (learning rate) ενός νευρικού δικτύου είναι παράμετρος επειδή ορίζεται ρητά πριν την εκπαίδευση. Από την άλλη, τα βάρη (weights) ενός νευρικού δικτύου δεν είναι παράμετρος επειδή εκπαιδεύονται από το train set. Το grid search είναι μια τεχνική που επιχειρεί να βρει τις βέλτιστες τιμές των παραμέτρων. Είναι μια εξαντλητική αναζήτηση που πραγματοποιείται σε συγκεκριμένες τιμές παραμέτρων ενός μοντέλου. Πολλές φορές το grid search χρησιμοποιεί k-fold cross validation για να βρει τις βέλτιστες παραμέτρους. Ο Πίνακας 3 παρουσιάζει τις τιμές των παραμέτρων που εξερευνήθηκαν κατά τη διάρκεια του Grid Search για κάθε μοντέλο machine learning.

6.4. ROC curve

Μια ROC curve (receiver operating characteristic curve) είναι μια γραφική παράσταση που απεικονίζει τη διαγνωστική ικανότητα ενός συστήματος δυαδικού classifier σε διάφορα thresholds. Η ROC είναι μια καμπύλη

Μοντέλο	**Παράμετρος**	**Τιμές**
Logistic Regression	solver	newton-cg, lbfgs, liblinear
	max_iter	100, 500, 1000
k-NN	n_neighbors	5, 10, 15, 20
	metric	euclidean, minkowski, manhattan
MLP	activation	tanh, relu
	learning_rate	constant, invscaling, adaptive
	max_iter	200, 500, 1000
Decision Tree	criterion	gini, entropy
	splitter	best, random
	max_features	auto, sqrt, log2
Random Forest	n_estimators	100, 500, 1000
	criterion	gini, entropy
	max_features	auto, sqrt, log2
Gaussian NB		

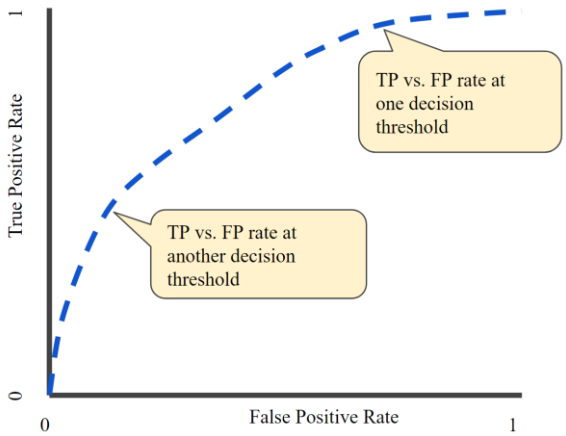
Πίνακας 3: Παράμετροι που διερευνήθηκαν στο Grid Search

πιθανότητας και το AUC (Area Under the Curve) αναπαριστά την ικανότητα του μοντέλου να διαχωρίζει τις κλάσεις. Όσο πιο ψηλό είναι το AUC τόσο καλύτερα το μοντέλο διαχωρίζει τις negative edges από τις positive edges. Η καμπύλη ROC αναπαρίσταται με το *true positive rate* να βρίσκεται στον y-άξονα και το *false positive rate* στο x-άξονα.

$$true\ possitive\ rate = \frac{TP}{TP + FN}$$

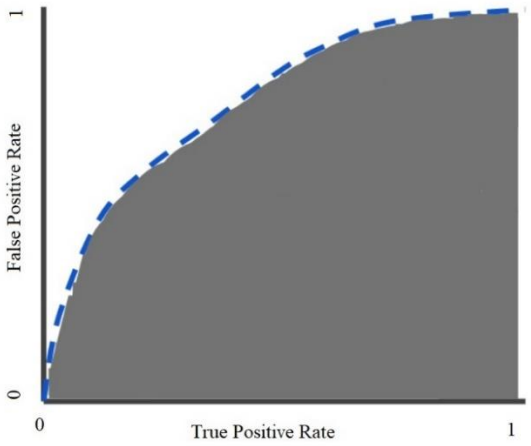
$$false\ possitive\ rate = \frac{FP}{FP + TN}$$

Μια καμπύλη ROC απεικονίζει το *true positive rate* εναντίον του *false positive rate* εναντίον σε διαφορετικά classification thresholds. Χαμηλότερα classification thresholds κατατάσσουν περισσότερα στοιχεία ως θετικά, αυξάνοντας έτσι τόσο τα *false positive* όσο και *τα true positive*. Το παρακάτω Σχήμα 14 δείχνει μια τυπική καμπύλη ROC.



Σχήμα 14: True positive rate και False positive rate σε διαφορετικά classification thresholds

Το AUC μετρά το εμβαδόν κάτω από την καμπύλη ROC από το (0,0) μέχρι το (1,1). Το Η AUC παρέχει ένα συνολικό μέτρο απόδοσης σε όλα τα πιθανά classification thresholds. Όσο μεγαλύτερο είναι το AUC τόσο πιο ψηλή είναι η απόδοση του μοντέλου με μέγιστη τιμή το 100% που σημαίνει ότι το μοντέλο ήταν ορθό σε όλες τις προβλέψεις του. Αντίστοιχα, Όσο μικρότερο είναι το AUC τόσο πιο χαμηλή είναι η απόδοση του μοντέλου με ελάχιστη τιμή το 0% που σημαίνει ότι το μοντέλο ήταν λανθασμένο σε όλες τις προβλέψεις του. Το Σχήμα 15 παρουσιάζει με γκρίζο χρώμα το AUC.



Σχήμα 15: Area under the ROC Curve

7. Experimental Results

7.1. Grid Search Results

Το πρώτο βήμα στην πειραματική αξιολόγηση των μοντέλων μας ήταν να εντοπίσουμε τις βέλτιστες παραμέτρους για τα μοντέλα. Όπως ήδη έχουμε αναφέρει το grid search είναι μια τεχνική που επιχειρεί να βρει τις βέλτιστες τιμές των παραμέτρων. Ο Πίνακας 4 παρουσιάζει το accuracy που προκύπτει σε κάθε συνδυασμό παραμέτρων για το μοντέλο k-NN στο δίκτυο deezer. Παρατηρούμε ότι διαφορετικές τιμές παραμέτρων προσδίδουν και διαφορετικό accuracy στο μοντέλο. Γενικά παρατηρήσαμε, ότι οι τιμές του accuracy των διαφόρων παραμέτρων που εξετάζονται στο grid search είναι κοντά η μια στην άλλη. Αυτό το γεγονός πιθανόν να οφείλεται ότι στην παρούσα εργασία εξετάσαμε μόνο ένα μικρό εύρος παραμέτρων λόγω του περιορισμένου χρόνου και υπολογιστικών πόρων που είχαμε στη διάθεση μας. Ο Πίνακας 5 παρουσιάζει τις βέλτιστες παραμέτρους που επέλεξε το grid search για το κάθε μοντέλο στο δίκτυο deezer.

metric	n_neighbors	Accuracy
euclidean	5	86.63%
euclidean	10	87.87%
euclidean	15	87.93%
euclidean	20	88.11%
minkowski	5	86.63%
minkowski	10	87.87%
minkowski	15	87.93%
minkowski	20	88.11%
manhattan	5	86.63%
manhattan	10	87.87%
manhattan	15	87.92%
manhattan	20	**88.12%**

Πίνακας 4: Το accuracy που προκύπτει σε κάθε συνδυασμό παραμέτρων για το μοντέλο k-NN στο δίκτυο deezer

Model	Best Parameters
Logistic Regression	'max_iter': 100, 'solver': 'newton-cg'
kNN	'metric': 'manhattan', 'n_neighbors': 20
MLP	'activation': 'relu', 'learning_rate': 'constant', 'max_iter': 200
Decision Tree	'criterion': 'gini', 'max_features': 'auto', 'splitter': 'best'
Random Forest	'criterion': 'gini', 'max_features': 'auto', 'n_estimators': 1000

Πίνακας 5: Οι βέλτιστες παράμετροι που επέλεξε το grid search για το κάθε μοντέλο στο δίκτυο deezer

7.2. Cross Validation Accuracy

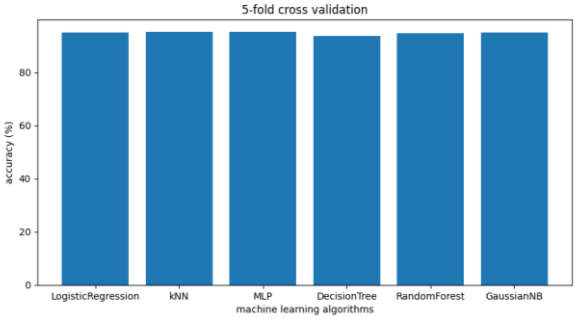
Στη συνέχεια χρησιμοποιήσαμε τις βέλτιστες παραμέτρους που προέκυψαν από το grid search για να εκπαιδεύσουμε τα μοντέλα και να αξιολογήσουμε το accuracy και το standard deviation που προκύπτουν από τη διαδικασία του 5-fold cross validation. Οι Πίνακες 6 και 7 παρουσιάζουν το μέσο accuracy και το standard deviation αντίστοιχα για κάθε μοντέλο σε κάθε dataset. Παρατηρούμε ότι το accuracy διαφέρει από μοντέλο σε μοντέλο και από dataset σε dataset. Για τα γραφήματα που σχετίζονται με κοινωνικά δίκτυα (hamsterster, twitch, github, deezer, facebook) παρουσιάζουν accuracy από 85.96 % έως 98.18 %. Το γεγονός αυτό μας αφήνει απόλυτα ικανοποιημένους εφόσον μπορούμε να πούμε ότι στα γραφήματα κοινωνικών δικτύων προβλέπουμε τις πλευρές μεταξύ των κόμβων που «κρύψαμε» με μεγάλη ακρίβεια (από 85.96 % έως 98.18 % ανάλογα με το dataset). Αντίθετα, σε τυχαίο γράφημα (erdos) παρουσιάζουν accuracy από 50.18 % έως 61.46 %. Είναι αναμενόμενο ότι σε ένα γράφημα όπου οι πλευρές μεταξύ των κόμβων τοποθετούνται πιθανοτικά χωρίς κάποια συσχέτιση δεν μπορεί να εφαρμοστεί σε ένα πρόβλημα link prediction, εξίσου και τα χαμηλά accuracy.Ένα άλλο σημείο που θα πρέπει να λάβουμε υπόψιν μας είναι το standard deviation που προκύπτει από το 5-fold cross validation. Ιδανικά θέλουμε να έχουμε μικρό standard deviation διότι έτσι μπορούμε να πούμε ότι δεν υπάρχουν μεγάλες αποκλίσεις μεταξύ των accuracy στα διάφορα folds δεδομένων που εξετάζονται. Πράγματι στην περίπτωση μας όλα τα standard deviation που προκύπτουν είναι μικρότερα του 1 % γεγονός που μας κάνει να εμπιστευθούμε ακόμη περισσότερο τα αποτελέσματα μας. Το Σχήμα 16 και 17 παρουσιάζουν το μέσο accuracy και το standard deviation αντίστοιχα για το κάθε μοντέλο στο δίκτυο facebook.

Dataset	Logist Regre	kNN	MLP	Decisi Tree	Rando Forest	Gausi NB
hamster	97.91 %	**98.18 %**	98.15 %	97.76 %	98.09 %	97.94 %
twitch	95.53 %	95.60 %	**95.69 %**	94.11 %	95.02 %	95.52 %
github	90.34 %	92.29 %	**92.34 %**	91.36 %	91.79 %	90.26 %
deezer	87.80 %	88.13 %	**88.21 %**	85.93 %	86.79 %	87.47 %
faceboo k	94.92 %	95.11 %	**95.12 %**	93.63 %	94.56 %	94.89 %
erdos	60.91 %	58.43 %	**61.46 %**	61.31 %	61.32 %	50.18 %

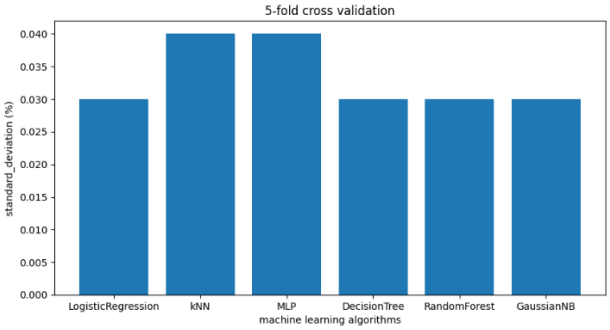
Πίνακας 6: Μέσο accuracy για κάθε μοντέλο σε κάθε δίκτυο

Dataset	Logist Regre	kNN	MLP	Decisi Tree	Rando Forest	Gausi NB
hamster	0.29 %	0.27 %	0.29 %	0.25 %	**0.23 %**	0.25 %
twitch	0.07 %	0.07 %	0.10 %	0.10 %	0.08 %	**0.06 %**
github	0.08 %	**0.07 %**	0.08 %	**0.07 %**	0.08 %	0.09 %
deezer	0.08 %	**0.04 %**	0.06 %	0.06 %	0.05 %	0.05 %
facebook	**0.03 %**	0.04 %	0.04 %	**0.03 %**	**0.03 %**	**0.03 %**
erdos	0.21 %	0.90 %	0.23 %	**0.12 %**	**0.12 %**	0.43 %

Πίνακας 7: Standard deviation για κάθε μοντέλο σε κάθε δίκτυο



Σχήμα 16: Μέσο accuracy για κάθε μοντέλο στο δίκτυο facebook



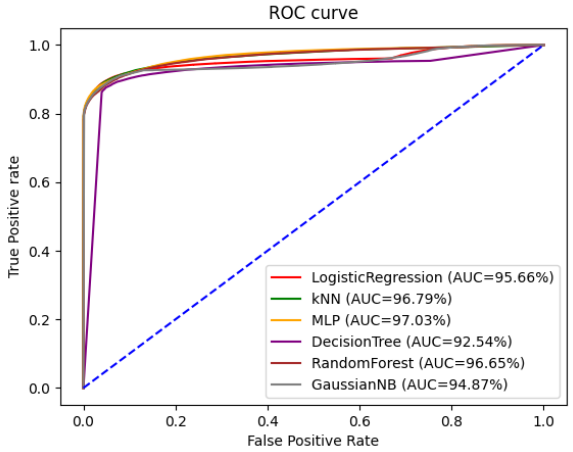
Σχήμα 17: Standard deviation για κάθε μοντέλο στο δίκτυο facebook

7.3. Area Under the curve

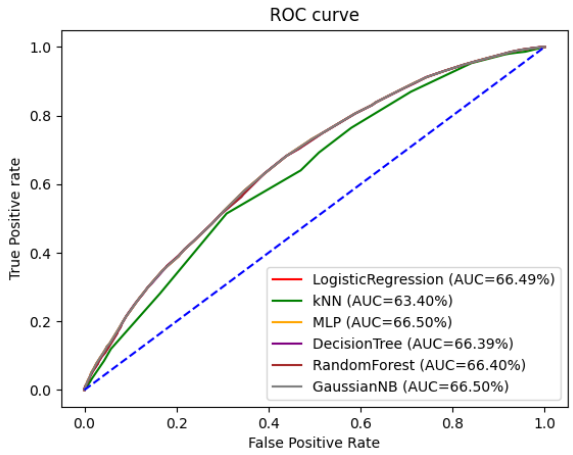
Μια τελευταία τεχνική που αποφασίσαμε να χρησιμοποιήσουμε για να αξιολογήσουμε τα μοντέλα μας ήταν το AUC (Area Under the Curve) στις ROC γραφικές παραστάσεις. Όπως ήδη αναφέραμε μια ROC curve είναι μια γραφική παράσταση που απεικονίζει τη διαγνωστική ικανότητα ενός συστήματος δυαδικού classifier. Όσο μεγαλύτερο είναι το εμβαδόν που σχηματίζεται κάτω από την καμπύλη τόσο καλύτερα το μοντέλο διαχωρίζει τις negative edges από τις positive edges. Ο Πίνακας 8 παρουσιάζει τα ACU που προκύπτουν για κάθε μοντέλο σε κάθε δίκτυο. Τα AUC στα γραφήματα που σχετίζονται με τα κοινωνικά δίκτυα κυμαίνεται από 86.78 % έως και 99.49 % ενώ στο τυχαίο γράφημα από 63.40 % έως 66.50 %. Όπως φαίνεται και από τα Σχήματα 18 και 19 στα γραφήματα κοινωνικών δικτύων τα μοντέλα στα κοινωνικά δίκτυα έχουν ψηλό AUC ενώ στο τυχαίο γράφημα το AUC είναι πιο χαμηλό και το ROC προσεγγίζει ευθεία γραμμή. Πάλι μπορούμε να συμπεράνουμε ότι οι τεχνικές που αναπτύξαμε δουλεύουν καλά στα γραφήματα με κοινωνικά δίκτυα εφόσον το AUC είναι αρκετά ψηλό.

Dataset	Logist Regre	kNN	MLP	Decisi Tree	Rando Forest	Gausi NB
hamster	99.32 %	99.00 %	99.43 %	98.42 %	**99.49 %**	99.19 %
twitch	97.93 %	98.03 %	**98.47 %**	93.79 %	97.89 %	97.78 %
github	95.66 %	96.79 %	**97.03 %**	92.54 %	96.65 %	94.87 %
deezer	92.70 %	92.71 %	**93.38 %**	86.78 %	92.03 %	89.15 %
facebook	96.90 %	97.16 %	**97.62 %**	93.56 %	97.05 %	96.41 %
erdos	66.49 %	63.40 %	**66.50 %**	66.39 %	66.40 %	**66.50 %**

Πίνακας 8: ACU που προκύπτουν για κάθε μοντέλο σε κάθε δίκτυο



Σχήμα 18: ROC curve για το δίκτυο github



Σχήμα 19: ROC curve για το erdos γράφημα

8. Conclusion

Στην παρούσα εργασία προσπαθήσαμε να επιλύσουμε το πρόβλημα του link prediction που είναι ένα από τα σημαντικότερα ερευνητικά θέματα στην περιοχή των γραφημάτων και των δικτύων. Παρά τον μεγάλο αριθμό μεθόδων που έχουν αναπτυχθεί το πρόβλημα του link prediction παραμένει ένα δύσκολο πρόβλημα. Ο στόχος του link prediction είναι ο εντοπισμός ζευγών κόμβων που θα σχηματίσουν έναν σύνδεσμο στο μέλλον. Χρησιμοποιήσαμε την τοπολογία μη-κατευθυνόμενων γραφημάτων από κοινωνικά δίκτυα για να εξάγουμε features με τα οποία εκπαιδεύσαμε αλγόριθμους machine learning.

Μέσω αυτής της εργασία καταφέραμε να δείξουμε ότι το link prediction πρόβλημα μπορεί να λυθεί με ψηλή ακρίβεια στις προβλέψεις μας χρησιμοποιόντας τους αλγόριθμος supervised machine learning: Logistic Regression, kNN, MLP, Decision Tree, Random Forest, Gaussian NB. Τα αποτελέσματα διαφέρουν από γράφημα σε γράφημα έτσι οδηγούμαστε στο συμπέρασμα ότι δεν υπάρχει πάντα «συνταγή» την οποία εάν ακολουθήσουμε θα πάρουμε τα βέλτιστα αποτελέσματα. Στα περισσότερα γραφήματα που εξετάσαμε παρατηρήσαμε ότι ο αλγόριθμος που βασίζεται σε νευρωνικά δίκτυα MPL «νικάει» στις περισσότερες περιπτώσεις. Ο αλγόριθμος MPL είναι και ο πιο βιολογικά

ρεαλιστικός εφόσον προσπαθεί να μιμηθεί τον τρόπο με τον οποίο μαθαίνουν οι νευρώνες στον εγκέφαλο. Αντίθετα, ο αλγόριθμος Gaussian NB είχε συνήθως τις χειρότερες επιδόσεις ωστόσο τα αποτελέσματα του όσον και τον άλλον αλγορίθμων που εξετάσαμε μας αφήνουν απόλυτα ικανοποιημένους.

Ένα μειονέκτημα της μεθόδου που ακολουθήσαμε είναι ο μεγάλος χρόνος εκπαίδευσης που απαιτείται για τα μοντέλα. Ο χρόνος εκπαίδευσης των μοντέλων αυξάνεται όσο αυξάνεται και το μέγεθος του δικτύου. Έτσι, στην περίπτωση μας το μικρότερο δίκτυο που ήταν το hamsterster εκπαιδεύτηκε σε σχετικά μικρό χρόνο ενώ το μεγαλύτερο δίκτυο που ήταν το facebook ο χρόνος εκπαίδευσης ήταν μεγάλος. Επίσης, είναι καλό κατά την εκτέλεση να χρησιμοποιούνται πολλοί CPUs έτσι ώστε να επιταχύνεται η εκπαίδευση. Τέλος, η όταν χρησιμοποιούνται αλγόριθμοι όπως Decision Tree και Random Forest απαιτείται πολλή μνήμη εφόσον αυτοί οι αλγόριθμοι κτίζουν δέντρα τα οποία διατηρούνται στη μνήμη.

9. Future Work

Σίγουρα θα ήταν καλό κατά την διαδικασία του grid search να εξερευνηθεί ένα μεγαλύτερο εύρος παραμέτρων, έτσι ώστε να βεβαιωθούμε ότι έχουμε εντοπίσει τις βέλτιστες παραμέτρους οι οποίες θα μας προσφέρουν και το μέγιστο accuracy. Επίσης, αντί να παράξουμε manually τα features από τον γράφο με τη μεθόδους που εξηγήσαμε πιο πάνω, θα μπορούσαμε να χρησιμοποιήσουμε τη μέθοδο του node embedding για να παράξουμε με έναν αυτοματοποιημένο τρόπο τα features. Τέλος, θα μπορούσαμε να δοκιμάσουμε να λύσουμε το πρόβλημα με διαφορετικές μεθόδους για link prediction όπως είναι τα random walks και το matrix factorization.

Bibliography

[1] D. Liben-Nowell and J. Kleinberg, "The Link-Prediction Problem for Social Networks," Journal of the American Society for Information Science and Technology, 2007.

[2] M. Al Hasan, V. Chaoji, S. Salem and M. Zaki, "Link Prediction using Supervised Learning," SDM06: workshop on link analysis, counter-terrorism and security, 2006.

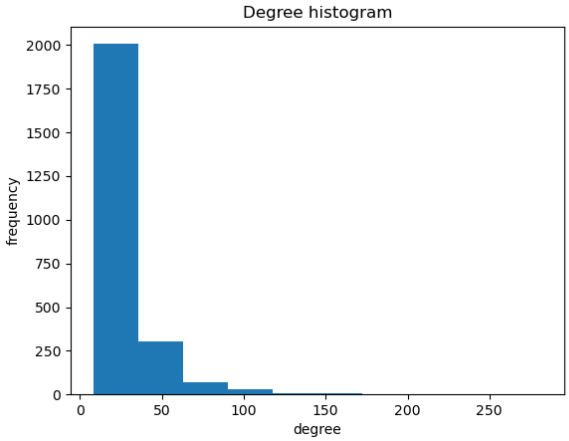
[3] W. Cukierski,, B. Hamner and B. Yang, "Graph-based Features for Supervised Link Prediction," International Joint Conference on Neural Networks, 2011.

[4] "Machine Learning in Python," scikit-learn, [Online]. Available: https://scikit-learn.org/stable/.

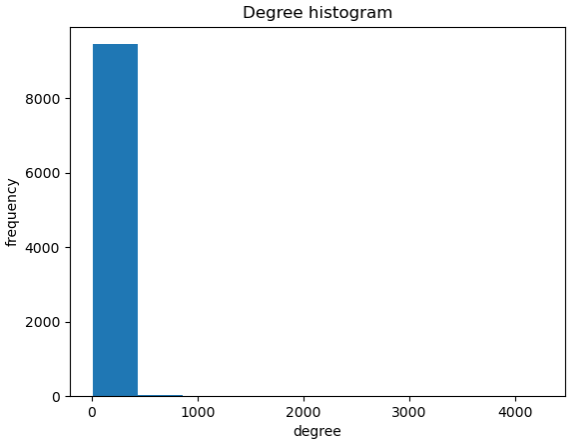
[5] "Towards Sata Science," [Online]. Available: https://towardsdatascience.com/.

ΠΑΡΑΡΤΗΜΑ – Επιπρόσθετες Γραφικές Παραστάσεις

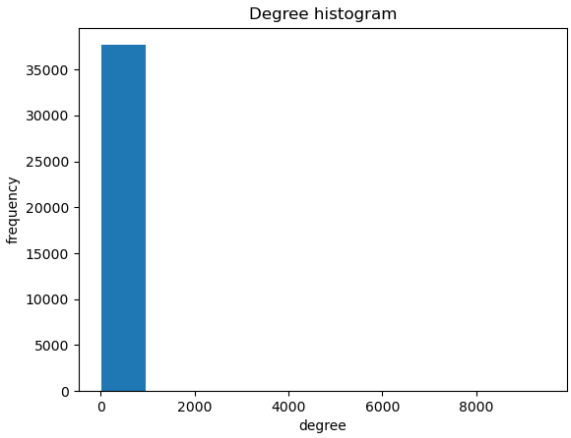
Degree Histograms:



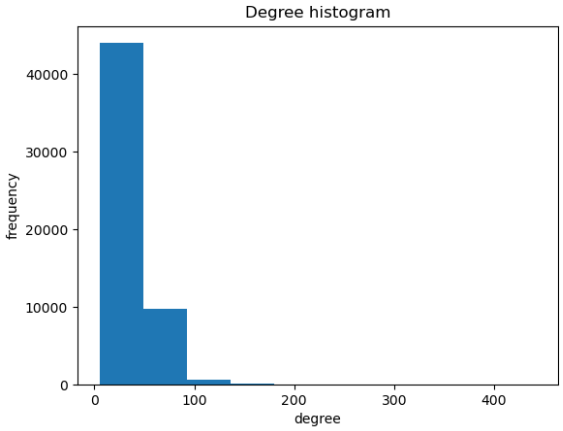
Σχήμα Π1a: Degree Histogram για το δίκτυο hamasterster



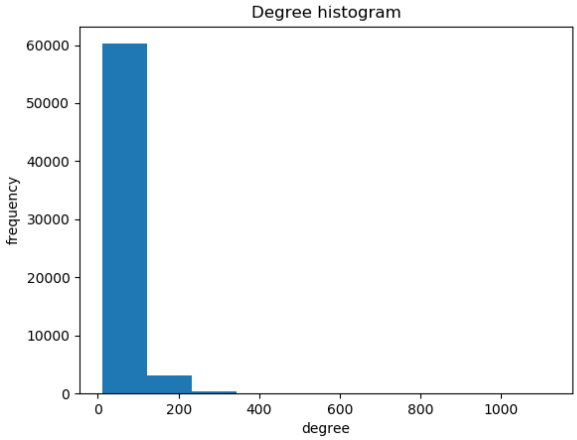
Σχήμα Π1b: Degree Histogram για το δίκτυο twitch



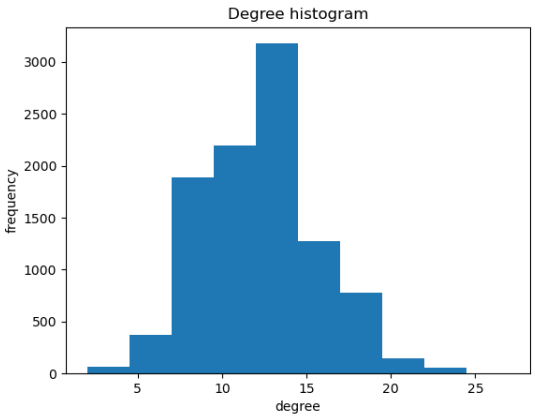
Σχήμα Π1c: Degree Histogram για το δίκτυο github



Σχήμα Π1d: Degree Histogram για το δίκτυο deezer

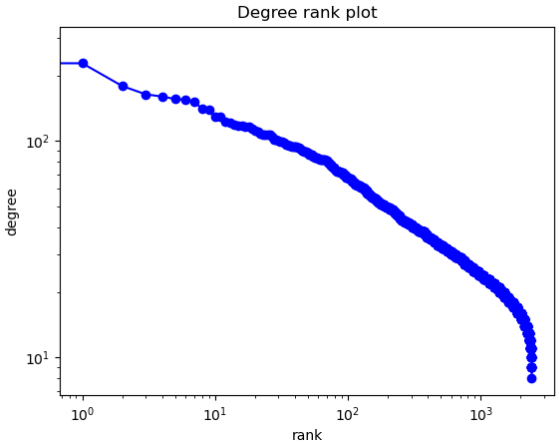


Σχήμα Π1e: Degree Histogram για το δίκτυο facebook

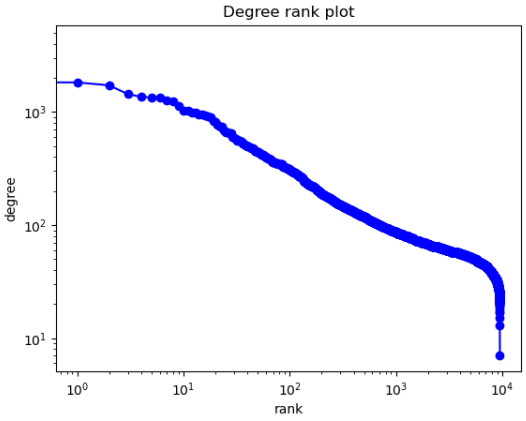


Σχήμα Π1f: Degree Histogram για το erdos γράφημα

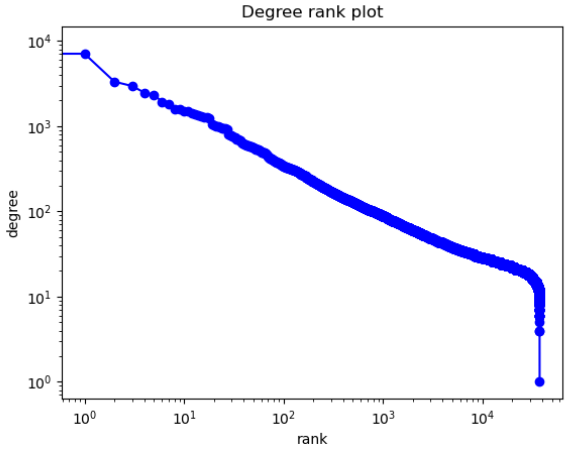
Degree Rank:



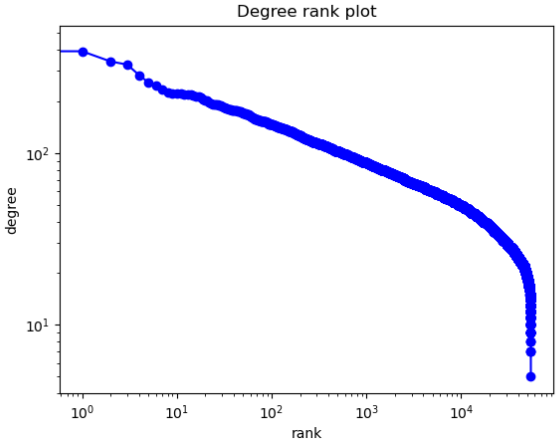
Σχήμα Π2a: Degree Rank για το δίκτυο hamasterster



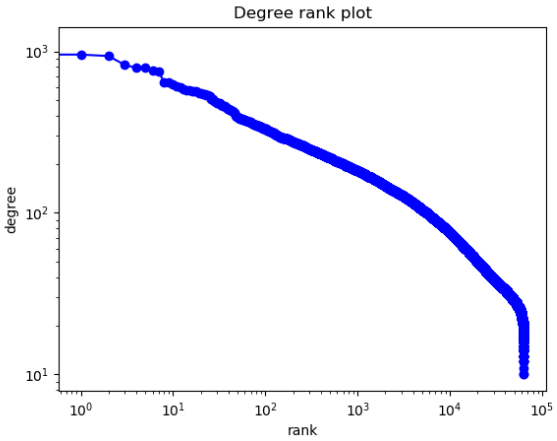
Σχήμα Π2b: Degree Rank για το δίκτυο twitch



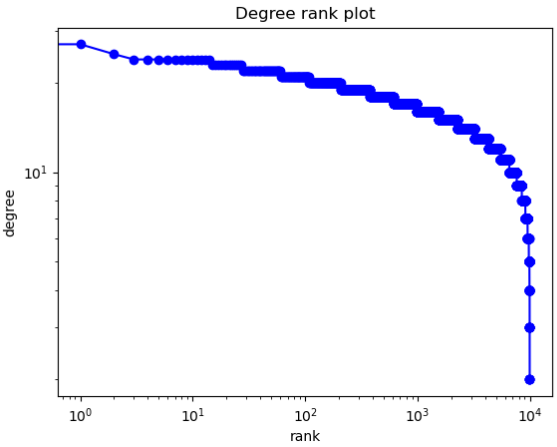
Σχήμα Π2c: Degree Rank για το δίκτυο github



Σχήμα Π2d: Degree Rank για το δίκτυο deezer

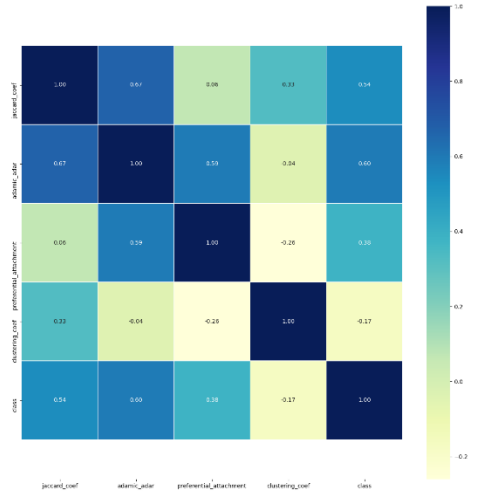


Σχήμα Π2e: Degree Rank για το δίκτυο facebook

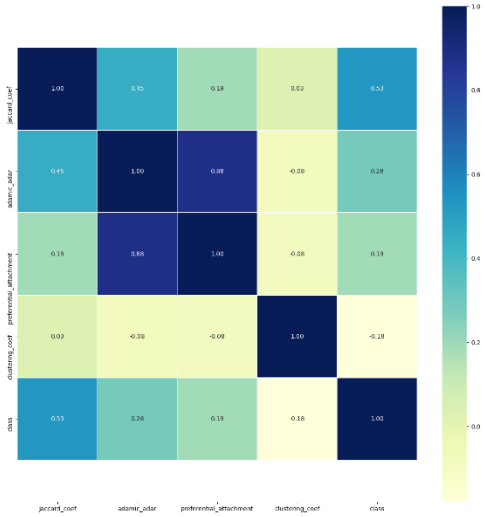


Σχήμα Π2f: Degree Rank για το erdos γράφημα

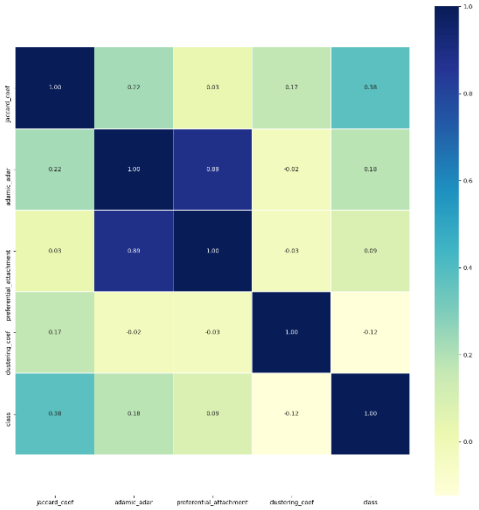
Correlation Analysis:



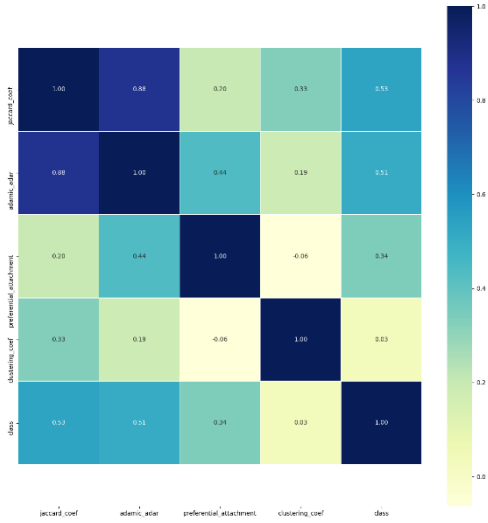
Σχήμα Π3a: Degree Rank για το δίκτυο hamasterster



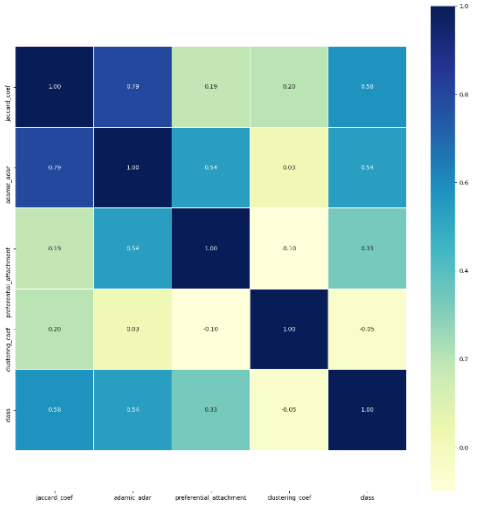
Σχήμα Π3b: Degree Rank για το δίκτυο twitch



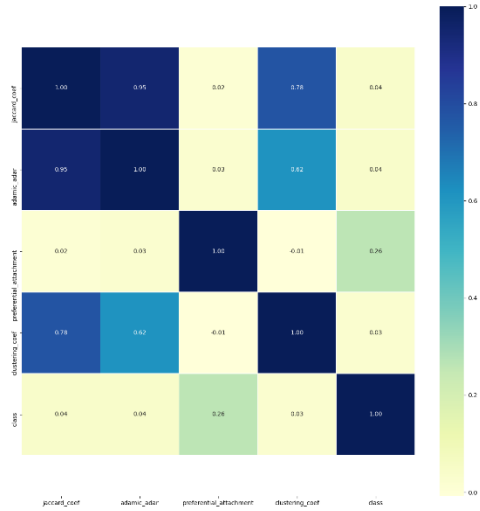
Σχήμα Π3c: Correlation Analysis για το δίκτυο github



Σχήμα Π3d: Correlation Analysis για το δίκτυο deezer

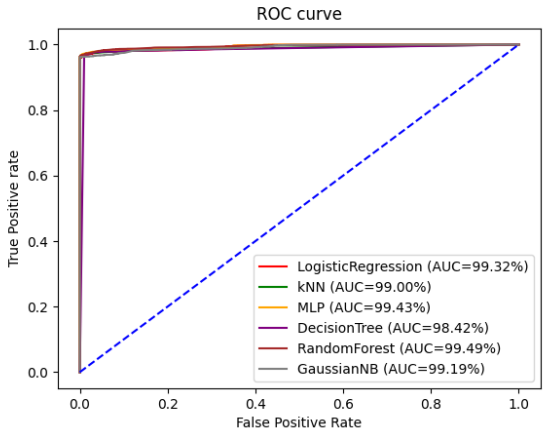


Σχήμα Π3e: Correlation Analysis για το δίκτυο facebook

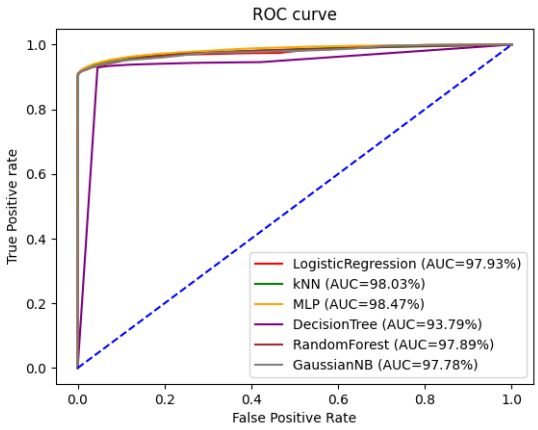


Σχήμα Π3f: Correlation Analysis για το erdos γράφημα

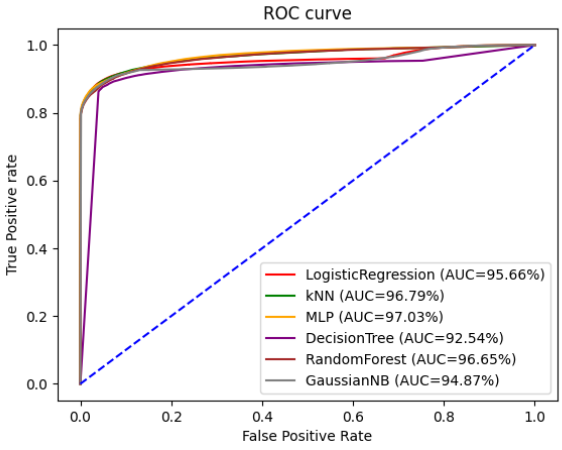
ROC Curve:



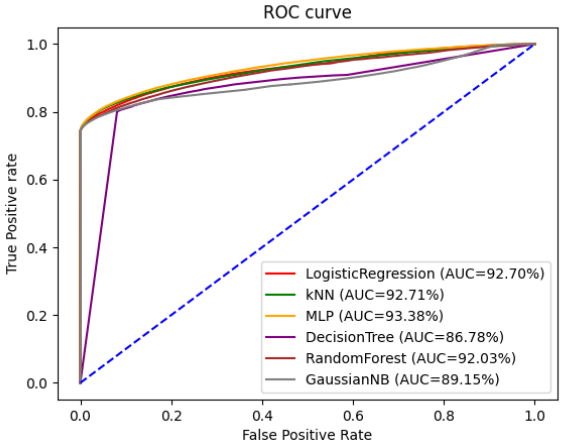
Σχήμα Π4a: ROC Curve για το δίκτυο hamasterster



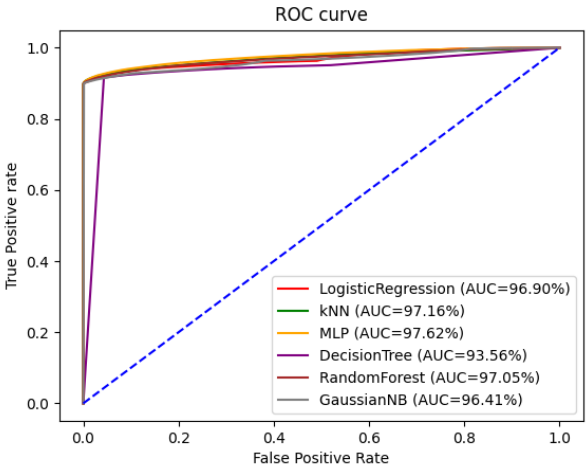
Σχήμα Π4b: ROC Curve για το δίκτυο twitch



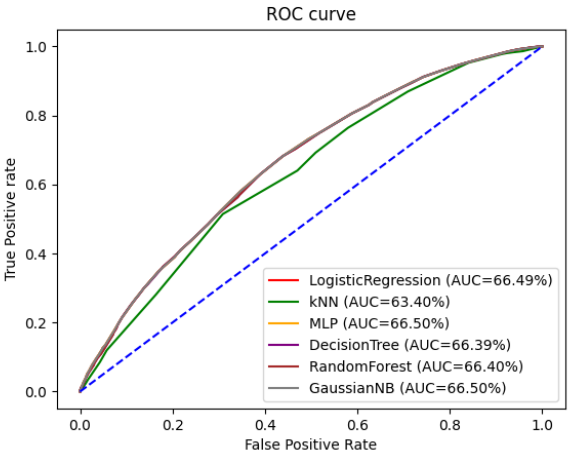
Σχήμα Π4c: ROC Curve για το δίκτυο github



Σχήμα Π4d: ROC Curve για το δίκτυο deezer



Σχήμα Π4e: ROC Curve για το δίκτυο facebook



Σχήμα Π4f: ROC Curve για το erdos γράφημα