Διπλωματική Εργασία

Τίτλος: «Εξαγωγή γνώσης από ροές κυκλοφορίας σε Δίκτυα Οριζόμενα από Λογισμικό, με τη χρήση τεχνικών εξόρυξης δεδομένων»

« **Knowledge extraction from traffic flows in Software Defined Networks, based on Data Mining Techniques**»

Επιβλέπουσα Καθηγήτρια: Μαργαρίτη Σπυριδούλα

Άρτα, Μάρτιος 2024

Δήλωση μη λογοκλοπής

Δηλώνω υπεύθυνα και γνωρίζοντας τις κυρώσεις του Ν. 2121/1993 περί Πνευματικής Ιδιοκτησίας, ότι η παρούσα μεταπτυχιακή εργασία είναι εκ ολοκλήρου αποτέλεσμα δικής μου ερευνητικής εργασίας, δεν αποτελεί προϊόν αντιγραφής ούτε προέρχεται από ανάθεση σε τρίτους. Όλες οι πηγές που χρησιμοποιήθηκαν (κάθε είδους, μορφής και προέλευσης) για τη συγγραφή της περιλαμβάνονται στη βιβλιογραφία.

Επίθετο, Όνομα

Ευαγγελία Κιούση

Υπογραφή

Ευαγγελία Κιούση

Εξαγωγή γνώσης από ροές κυκλοφορίας σε Δίκτυα Οριζόμενα από Λογισμικό, με τη χρήση τεχνικών εξόρυξης δεδομένων

Ευαγγελία Κιούση

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Επιβλέπων καθηγητής

   Όνομα Επίθετο, Μαργαρίτη Σπυριδούλα

   τίτλος, βαθμίδα

2. Μέλος επιτροπής

   Όνομα Επίθετο, Τσούλος Ιωάννης

   τίτλος, βαθμίδα

3. Μέλος επιτροπής

   Όνομα Επίθετο, Στεργίου Ελευθέριος

   τίτλος, βαθμίδα

Ευχαριστίες

Μέσα από την παρούσα διπλωματική εργασία θα ήθελα να ευχαριστήσω θερμά την επιβλέπουσα καθηγήτριά μου κα. Σπυριδούλα Μαργαρίτη για την εμπιστοσύνη που έδειξε στο πρόσωπο μου από την πρώτη στιγμή, για την καθοδήγηση και την υποστήριξη της όλο αυτό το διάστημα.

Θερμές ευχαριστίες στην κόρη μου Κατερίνα και στον σύζυγό μου Λάμπρο για την κατανόηση που επέδειξαν καθόλη την περίοδο των σπουδών μου.

Περίληψη

Στην παρούσα εργασία γίνεται μια εκτενής βιβλιογραφική επισκόπηση στην οποία περιλαμβάνονται μελέτες οι οποίες περιγράφουν τη λογική των SDN δικτύων, του πρωτοκόλλου Open Flow, και του Traffic Engineering. Επίσης, περιγράφεται η διαχείριση των ροών κυκλοφορίας μέσα από την βαθιά μάθηση στα SDN δίκτυα, η αρχιτεκτονική των δικτύων SDN, οι τεχνικές εξόρυξης δεδομένων, η λειτουργία της πλατφόρμας Apache Spark με τα βασικότερα χαρακτηριστικά της όπως: το Resilient distributed dataset, το Directed acrylic graph, τα data frames και τα datasets, το Machine learning library και η χρήση της γλώσσας python. Ακόμη, περιγράφονται οι αλγόριθμοι και οι τεχνικές εξόρυξης δεδομένων όπως η ταξινόμηση, η συσταδοποίηση, η παλινδρόμηση, οι κανόνες σύνδεσης, τα νευρωνικά δίκτυα και οι εφαρμογές εξόρυξης δεδομένων. Ακολουθεί, η μεθοδολογία που χρησιμοποιήθηκε για την πειραματική προσέγγιση περιγράφοντας τα εργαλεία που χρησιμοποιήθηκαν για την υλοποίηση καθώς και την επικύρωση των πειραματικών μετρήσεων αναδεικνύοντας στην ουσία τον αλγόριθμο που δείχνει τις καλύτερες μετρήσεις για την πρόβλεψη. Ακολουθεί η πειραματική προσέγγιση και οι πειραματικές μετρήσεις, αρχικά με την περιγραφή των δεδομένων που περιλαμβάνεται στο αρχείο του συνόλου δεδομένων με ονομασία SDN_Traffic.csv και στην συνέχεια η ανάπτυξη του μοντέλου πρόβλεψης εξόρυξης δεδομένων μέσα από την χρήση τριών αλγορίθμων μάθησης όπως ο Naïve Bayes, ο Linear Regressionκαι ο Logistic Regression παραθέτοντας τις ανάλογες μετρήσεις.

Λέξεις Κλειδιά: *Ροές Δεδομένων, Big Data, SDN, Τεχνικές Εξόρυξης Δεδομένων*

Abstract

In this paper, an extensive literature review is carried out which includes studies that describe the logic of SDN networks, the Open Flow protocol, and Traffic Engineering. It also describes the management of traffic flows through deep learning in SDN networks, the architecture of SDN networks, data mining techniques, the operation of the ApacheSpark platform with its main features such as Resilient distributed dataset, Directed acrylic graph, data frames and datasets, Machine learning library and the use of the python language. Also, data mining algorithms and techniques such as classification, clustering, regression, connection rules, neural networks, and data mining applications are described. Next, the methodology used for the experimental approach describing the tools used for the implementation as well as the validation of the experimental measurements highlighting essentially the algorithm that shows the best measurements for the prediction The experimental approach and the experimental measurements follow, initially with the description of the dataset included, named SDN_Traffic.csv file and then the development of the data mining prediction model through the use of three learning algorithms such as Naïve Bayes, Linear Regression and Logistic Regression, listing the corresponding measurements.

Keywords: *Data Streams, Big Data, SDN, Data Mining Techniques*

Ακρωνύμια

QoS-Quality of Service

QoE-Quality of Experience

SDN-Software Defined Network

GPU-Graphic Processing Unit

DPI-Deep Packet Inspection

API-Application Programming Interface

OF-Open Flow

TE-Traffic Engineering

IoT-Internet of Things

DL-Deep Learning

IP-Internet Protocol

RNN-Recurrent Neural Network

LSTM-GRU-Long Short Term Memory-Gated Recurrent Unit

CNN-Convolutional Neural Network

DBN-Deep Belief Network

DoS-Denial of Service Attack

AI-Artificial Intelligence

DAG-Directed Acyclic Graph

MLib-Machine Learning Library

RDD-Resilient Distributed Dataset

Πίνακας Περιεχομένων

Μέρος Α' : Θεωρητική Προσέγγιση

Κεφάλαιο 1ο. Εισαγωγή

Η λήψη αποφάσεων αποτελεί σημαντική ευθύνη για τη διοίκηση ενός φορέα. Η περιοχή της Επιστήμης των Δεδομένων αποτελεί πρόσφορο έδαφος για την εξαγωγή γνώσης από μεγάλους όγκους δεδομένων. Για να τελεσφορήσει όμως απαιτείται χρήση αλγορίθμων μηχανικής μάθησης και στατιστικής προκειμένου να εξαχθεί γνώση και προβλέψεις. Ωστόσο, οι συνεχείς μεταβολές που αφορούν την διαχείριση των δεδομένων οδήγησαν στην ανάπτυξη νέων τεχνολογιών, αλγορίθμων μηχανικής μάθησης, αρχιτεκτονικών και άλλων συστημάτων ώστε να βελτιωθεί η ταχύτητα προκειμένου να υπάρξει η δυνατότητα επεξεργασίας και διαχείρισης τεράστιου όγκου δεδομένων (big data). Οι τεχνικές που αναπτύχθηκαν εξαιτίας αυτής της αναγκαιότητας βασίζονται στην βελτίωση της πρόβλεψης και της λήψης αποφάσεων.

1.1 Η Σημαντικότητα της Ταξινόμησης στην Ροή ενός Δικτύου

Η ταξινόμηση της δικτυακής κυκλοφορίας των δεδομένων χρησιμοποιούνται ευρέως για την αποτελεσματικότερη διαχείριση δικτύων, την παρακολούθηση των λειτουργιών του, την εκτίμηση της ποιότητας των παρεχόμενων υπηρεσιών, τη λήψη αποφάσεων για το σχεδιασμό δικτύων, την πρόβλεψη των μοτίβων μελλοντικής κυκλοφορίας και την βελτίωση της ποιότητας υπηρεσιών (Yan & Yuan, 2018).

Η ακριβής ταξινόμηση της κυκλοφορίας είναι απαραίτητη για την αντιμετώπιση ζητημάτων QoS (Quality of Service) και για εργασίες παρακολούθησης ασφάλειας.

1.1.1 QoS (Quality of Service)

Για την ανάπτυξη ενός κατάλληλου και αποτελεσματικού επιπέδου QoS είναι απαραίτητη η υιοθέτηση μιας σωστής στρατηγικής. Γενικότερα, η υιοθέτηση ενός καλού μοντέλου που θα προβαίνει σε χρεώσεις αναλογικά των πόρων που χρησιμοποιούνται εξασφαλίζει τη διαφάνεια αφού εξαλείφει τις υποχρεώσεις των πελατών (Farad, 2015). Οι ISP, προβαίνουν στην ανάπτυξη αποτελεσματικών και κερδοφόρων μοντέλων κυκλοφορίας. Οι περισσότερες από τις τεχνικές που προτείνονται είναι αποτελεσματικές αφού οι καταναλωτές δίκαια υφίστανται την χρέωση του QoS. Βέβαια, καμία λύση QoS δεν έχει ικανοποιήσει επαρκώς τις ανάγκες των πελατών. Για τον λόγο αυτό, θα πρέπει να υπάρχουν οι κατάλληλες λύσεις QoS, οι οποίες πρέπει να εφαρμοστούν αν λάβει κανείς υπόψη του την αποτελεσματικότητα σε οικονομικό, τεχνικό και κοινωνικό επίπεδο. Η τεχνική αποτελεσματικότητα αναφέρεται στο κόστος που έχει να κάνει με την τεχνολογία και τα ανάλογα συστήματα. Η οικονομική αποτελεσματικότητα αφορά τις συνέπειες που επιφέρει το μοντέλο για την αξιοποίηση του δικτύου. Ένα καλό μοντέλο πρέπει να εφαρμόζεται με βάση τη συνέπεια και τη διαφάνεια. Το κόστος της εφαρμογής του QoS, είναι σημαντικό και δεν πρέπει να υπερβαίνει τα έσοδα που έχουν δημιουργηθεί από αυτό. Θέματα σταθερότητας και συνέπειας του δικτύου θα πρέπει να ληφθούν υπόψη κατά την εφαρμογή του QoS. Έτσι, μια προγραμματισμένη ταξινόμηση κυκλοφορίας θα πρέπει να ενσωματωθεί στο μοντέλο που βασίζεται στο QoS. Η κατηγοριοποίηση της κυκλοφορίας αποτελεί μια σημαντική λύση. Τα μοτίβα κυκλοφορίας σε ένα σύστημα ISP μπορούν να ανιχνευθούν μέσα από την ταξινόμησης της κυκλοφορίας.

Ακόμη, η ταξινόμηση της κυκλοφορίας, μπορεί να χρησιμοποιηθεί για να προσδιοριστούν οι κατηγορίες εφαρμογών που χρησιμοποιούνται από τους πελάτες σε μια συγκεκριμένη στιγμή. Οι εν λόγω πληροφορίες μπορούν να ανακτηθούν από το δίκτυο χωρίς να παραβιάζονται οι νόμοι περί απορρήτου που ρυθμίζουν την χρήση του

διαδικτύου. Η ταξινόμηση της κυκλοφορίας είναι σημαντική καθώς διευκολύνει την χρήση ενός μοντέλου που βασίζεται στην κατηγορία το οποίο είναι δίκαιο για τους πελάτες διασφαλίζοντας τη βιωσιμότητα. Σε αυτό το μοντέλο, οι ISP είναι σε θέση να ανακτήσουν το κόστος παράδοσης του QoS, χρεώνοντας με βάση τις ανάγκες και τις υπηρεσίες που λαμβάνουν (Nguyen et al., 2009). Η ταξινόμηση της κυκλοφορίας σε πραγματικό χρόνο διευκολύνει την ανάπτυξη αυτοματοποιημένων αρχιτεκτονικών QoS. Το γεγονός αυτό οδηγεί σε μια αποτελεσματική διαμόρφωση που σχετίζεται με τις ανάγκες QoS ανάμεσα στις εφαρμογές ~~του~~ και του διαδικτύου

1.1.2 Σύστημα Ανίχνευσης Εισβολών

Εκτός από την επίλυση ζητημάτων QoS για τους ISP, είναι η συνδρομή στους φορείς στο να μπορούν να διακρίνουν μη φυσιολογικές συμπεριφορές για να προστατεύσουν τους πελάτες τους από τις απειλές. Τα τελευταία δέκα χρόνια έχουν αυξηθεί τα τρωτά σημεία αλλά και οι επιθέσεις μέσω του διαδικτύου. Έτσι, η αύξηση της συνδεσιμότητας με το διαδίκτυο και των εταιρικών συστημάτων για τον έλεγχο εθνικών υποδομών που είναι ζωτικής σημασίας, έχει επεκτείνει την ικανότητα των ξένων να παραβιάζουν την ασφάλεια. Για την αντιμετώπιση ενός αυξανόμενου αριθμού επιθέσεων και απειλών η ταξινόμηση της κυκλοφορίας του δικτύου έχει διαμορφωθεί ως σύστημα ανίχνευσης εισβολής και αποτελεί σημαντικό εργαλείο ασφάλειας για τη διαχείριση του κινδύνου και της συνολικής αρχιτεκτονικής ασφαλείας. Το σύστημα ανίχνευσης εισβολής χρησιμοποιείται ως γραμμή άμυνας για να εντοπιστούν ύποπτες και κακόβουλες δραστηριότητες στην κυκλοφορία του δικτύου. Προβαίνει σε συλλογή και ανάλυση πληροφοριών από διάφορες πηγές και μόλις εντοπίσουν την επίθεση ενημερώνει το δίκτυο ώστε να απαντήσει κατάλληλα. Έτσι, μια ακριβής προσέγγιση ταξινόμησης του δικτύου παίζει σημαντικό ρόλο στην υποστήριξη του δικτύου και στους χειριστές για να προστατεύσουν τα δίκτυά τους από απειλές και επιθέσεις (Farad, 2015).

Πολλά ήταν τα σχήματα ταξινόμησης της κυκλοφορίας του δικτύου τα οποία έχουν ερευνηθεί σε βάθος την τελευταία δεκαετία. Μια από τις προσεγγίσεις που

θεωρήθηκαν επιτυχής ήταν η χρήση των σταθερών αριθμών θυρών[1]. Ωστόσο, η τρέχουσα γενιά εφαρμογών προσπαθεί να αποκρύψει την επισκεψιμότητα κάνοντας χρήση δυναμικής θύρας. Επίσης, εφαρμογές στις οποίες οι αριθμοί των θυρών δεν είναι γνωστοί δεν μπορούν φυσικά να αναγνωριστούν εκ των προτέρων. Άλλες προσεγγίσεις έχουν την βάση τους στην επιθεώρηση του περιεχομένου των πακέτων (Moore & Papagiannaki, 2005). Το περιεχόμενο του φορτίου που θεωρείται ωφέλιμο πρέπει να ελεγχθεί για να διαπιστωθεί αν περιέχονται σε αυτό υπογραφές εφαρμογών. Τα όποια χαρακτηριστικά εξάγονται από τα δεδομένα κυκλοφορίας και στη συνέχεια συγκρίνονται με τις γνωστές υπογραφές εφαρμογών που παρέχονται από ειδικούς. Ωστόσο, οι προσεγγίσεις αυτές ενέχουν μειονεκτήματα και πλεονεκτήματα. Δεν έχουν τη δυνατότητα αναγνώρισης επιθέσεων για εφαρμογές που δεν έχουν υπογραφές. Το γεγονός αυτό, σημαίνει ότι οι προσεγγίσεις αυτές πρέπει να διατηρούν μια ενημερωμένη λίστα υπογραφών. Ωστόσο, αυτό είναι πρόβλημα από μόνο του αφού οι νέες εφαρμογές δέχονται επιθέσεις καθημερινά και η εκ βάθων επιθεώρηση των πακέτων είναι ένα δύσκολο έργο καθώς απαιτείται χρόνος επεξεργασίας και μνήμης (Auld et al., 2007). Άλλες προσεγγίσεις εστιάζουν στην αποτελεσματική μηχανική μάθηση. Αυτό προϋποθέτει ότι οι εφαρμογές κατά κύριο λόγο αποστέλλουν δεδομένα με ένα είδος μοτίβου το οποίο χρησιμοποιείται και ως μέσο ταξινόμησης. των συνδέσεων από διαφορετικές κατηγορίες κυκλοφορίας. Για να εξαχθούν τέτοια μοτίβα, μόνο κεφαλίδες TCP/IP χρειάζονται για την παρατήρηση στατιστικών ροής, όπως το μέσο μέγεθος πακέτου, το μήκος ροής και τον συνολικό αριθμό των πακέτων. Αυτό επιτρέπει τις τεχνικές ταξινόμησης αφού με αυτόν τον τρόπο υπάρχει αρκετή πληροφόρηση (Auld et al., 2007).

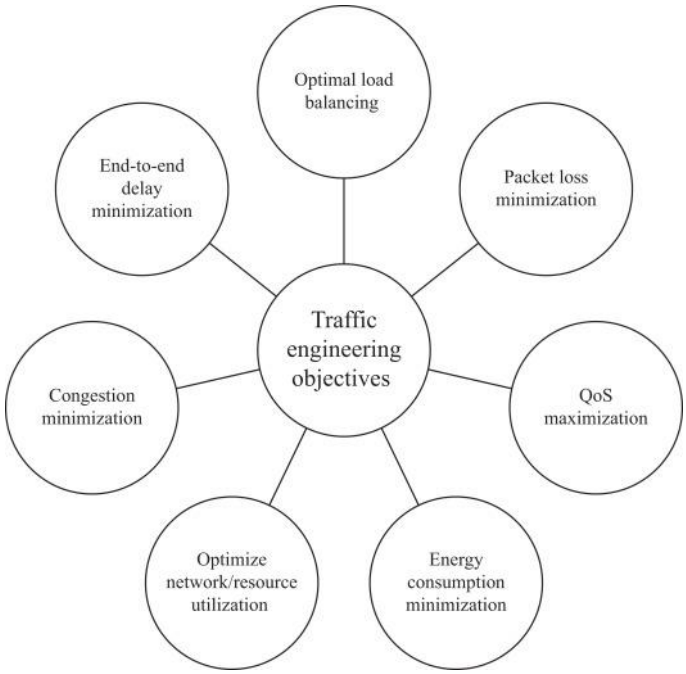
1.Ορισμός της θύρας: *Μια θύρα δικτύου είναι εικονικό σημείο σύνδεσης που επιτρέπει στις συσκευές να επικοινωνούν μεταξύ τους.*

Κάθε θύρα αναγνωρίζεται από έναν μοναδικό αριθμό, γνωστό ως αριθμός θύρας, ο οποίος χρησιμοποιείται για να κατευθύνει την

κυκλοφορία δικτύου σε μια συγκεκριμένη υπηρεσία σε μια συσκευή δικτύου. Οι θύρες είναι κρίσιμες για τη λειτουργία της επικοινωνίας δικτύου, καθώς επιτρέπουν την ταυτόχρονη εκτέλεση πολλαπλών υπηρεσιών σε μία μόνο συσκευή.(https://portalcripto.com.br/)

1.1.3 Traffic Engineering (TE) σε SDN

Το Traffic Engineering (TE) κατέχει σημαντικό ρόλο στη βελτίωση της απόδοσης του δικτύου καταγράφοντας όλη την κυκλοφορία σε πραγματικό χρόνο, κάνοντας προβλέψεις για εκείνη ενώ ταυτόχρονα σχεδιάζει τους μηχανισμούς δρομολόγησης με σκοπό να βελτιωθούν οι πόροι του δικτύου (Akyildiz et al., 2016; Shu et al., 2016). Η μηχανική κυκλοφορίας σε ένα δίκτυο SDN, τελεί τις δραστηριότητές του, οι οποίες διακρίνονται σε: διαχείριση ροής, σε σφάλμα και σε ανοχή, σε ενημέρωση τοπολογίας, και στην τελική φάση σε ανάλυση και σε χαρακτηρισμό της κυκλοφορίας (Εικόνα 1.)



Εικόνα 1.Δραστηριότητες Κυκλοφοριακής Μηχανικής (Science direct.com).

Η μηχανική κυκλοφορίας δίκτυα μεγάλης κλίμακας χρησιμοποιεί πολλές φορές διάφορους τύπους δεδομένων. Η ταξινόμηση αυτών των έργων βοηθά σημαντικά στην ολοκλήρωση τους. Έχει μεγάλη αξία, αφού η ταξινόμηση της μηχανικής κυκλοφορίας κάνει χρήση δεδομένων όπου αναλαμβάνει να τα οργανώσει και να τα καθαρίσει με σκοπό την απόκτηση ενός μορφοποιημένου συνόλου δεδομένων στο οποίο θα χρησιμοποιήσει την μέση και την ελάχιστη τιμή αλλά και άλλους δείκτες προκειμένου να δημιουργήσει χαρακτηριστικά μέσα από αλγορίθμους ομαδοποίησης. Ωστόσο, η μηχανική κυκλοφορίας βοηθά στο να κατανοηθούν οι ρυθμίσεις των παραμέτρων του μοντέλου (Yang et al., 2023).

1.3 Αντικείμενο της Εργασίας

Οι περιορισμοί στο εύρος ζώνης, η καθυστέρηση, η εμπειρία χρήστη και η ασφάλεια έχουν αυξηθεί με την αύξηση του αριθμού συσκευών που συνδέονται στο διαδίκτυο. Για τη διευκόλυνση των λειτουργιών διαχείρισης του δικτύου, την ταξινόμηση της προτείνεται η χρήση της μηχανικής μάθησης.

Η παρούσα εργασία, πραγματεύεται τη διαχείριση ενός συνόλου δεδομένων το οποίο περιλαμβάνει στοιχεία για την κίνηση ενός δικτύου SDN. Ειδικότερα, γίνεται χρήση τριών αλγορίθμων μηχανικής μάθησης (Naïve Bayes, Linear Regression, Logistic Regression) για ταξινόμηση της κυκλοφορίας σε δίκτυα οριζόμενα από το λογισμικό, παραθέτοντας τα αποτελέσματά τους ανά δύο κλάσεις και ανά έξι κλάσεις, παρουσιάζοντας και συγκρίνοντας τις αντίστοιχες προβλέψεις.

1.2 Σκοπός της εργασίας

Η αποτελεσματική παρακολούθηση και διαχείριση της δικτυακής κυκλοφορίας δεδομένων απαιτεί την χρήση αποτελεσματικών και μεγάλης ακρίβειας τεχνικών κατηγοριοποίησης της προσβλέποντας στη βελτίωση της απόδοσης του δικτύου. Η

ανάγκη αυτή επεκτείνεται από δύο παράγοντες: α) τον τεράστιο όγκο διαδικτυακής κυκλοφορίας και β) τον μεγάλο αριθμό των νέων αναδυόμενων εφαρμογών που διεκδικούν σημαντικό μερίδιο κίνησης.

Σκοπός της παρούσας εργασίας, είναι η δημιουργία προβλέψεων δεδομένων της κυκλοφορίας του δικτύου SDN με τεχνικές εξόρυξης δεδομένων μέσα από την χρήση αλγορίθμων μηχανικής μάθησης που στοχεύει στο να περιγράψει την πρόβλεψη, την ορθότητα και την ακρίβεια των δεδομένων του δικτύου.

1.4Δομή της Εργασίας

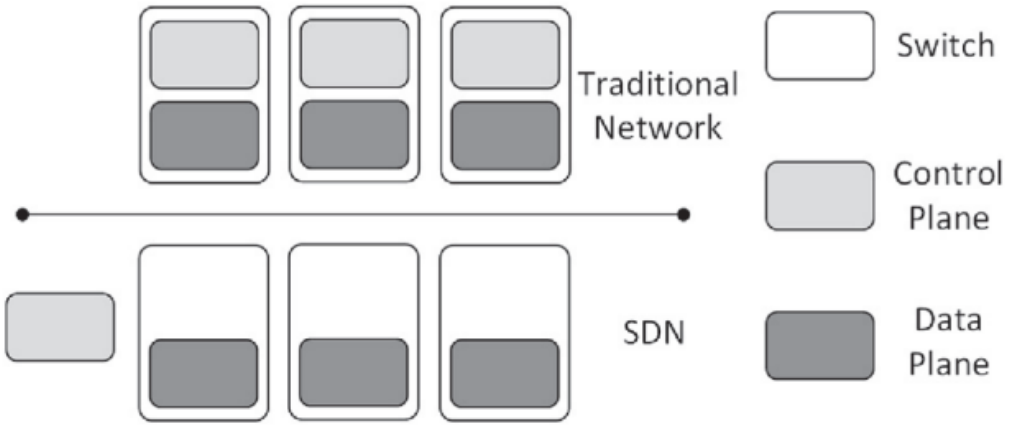
Η παρούσα εργασία αποτελείται από τέσσερα κεφάλαια. Στο πρώτο κεφάλαιο, παρουσιάζεται η εισαγωγή, η σημαντικότητα της ταξινόμησης στην ροή ενός δικτύου, QoS, η μηχανική κυκλοφορίας στα SDN δίκτυα όπως επίσης ο σκοπός και το αντικείμενο της έρευνας. Στο δεύτερο κεφάλαιο, γίνεται μια εκτενής βιβλιογραφική επισκόπηση στην οποία παρουσιάζονται η εξόρυξη δεδομένων, τα δίκτυα SDN, η αρχιτεκτονική τους, το πρωτόκολλο Open Flow, οι τεχνικές ταξινόμησης και σχετικές εργασίες. Στη συνέχεια, στο τρίτο κεφάλαιο, παρουσιάζεται η μεθοδολογία της υλοποίησης των πειραματικών μετρήσεων παρουσιάζοντας τα εργαλεία που επιλέχθηκαν για την υλοποίηση. Τέλος στο τέταρτο κεφάλαιο, παρουσιάζονται οι πειραματικές μετρήσεις όπου παρουσιάζονται τα δεδομένα και στην συνέχεια γίνεται η ανάπτυξη του μοντέλου προβλέψεων εξόρυξης δεδομένων μέσα από τους αλγορίθμους μηχανικής μάθησης Naïve Bayes, Logistic Regression και Linear Regression. Στην παρούσα εργασία, περιγράφεται η υλοποίηση προβλέψεων με βάση τα δεδομένα κίνησης ενός δικτύου SDN με την χρήση τριών αλγορίθμων μηχανικής μάθησης με σκοπό την ανάδειξη της καλύτερης πρόβλεψης στο δίκτυο. Ο λόγος που επιλέχθηκαν οι αλγόριθμοι μηχανικής μάθησης ήταν για λόγους σύγκρισης και ανάδειξης του αλγορίθμου που δίνει την μεγαλύτερη ακρίβεια.

Κεφάλαιο 2ο. Θεωρητικό Υπόβαθρο – Σχετικές Εργασίες

2.1 SDN-Software Defined Network

Η δικτύωση που ορίζει το λογισμικό των SDN στην ουσία διαχωρίζει το επίπεδο ελέγχου από το επίπεδο δεδομένων. Οι συσκευές του δικτύου όπως: οι διακόπτες και οι δρομολογητές, έχουν προωθητική λειτουργία τηρώντας όμως όλους τους κανόνες προώθησης των πινάκων ροής. Το επίπεδο ελέγχου που εκτελείται σε άλλη συσκευή έχει υπό την ευθύνη του την διαχείριση τους πίνακες ροής όλων των συσκευών. Το γεγονός αυτό, δίνει την δυνατότητα δυναμικής διαμόρφωσης των συσκευών προσφέροντας με αυτόν τον τρόπο μια συνολική εικόνα του δικτύου. Στην Εικόνα 2, απεικονίζεται το επίπεδο ελέγχου και το επίπεδο δεδομένων τόσο σε ένα παραδοσιακό δίκτυο όσο και σε ένα δίκτυο SDN. Συγκριτικά με τα παραδοσιακά δίκτυα, το δίκτυο SDN υπερτερεί καθώς προσφέρει (Queiroz et al., 2019):

1. Καθολικό έλεγχο. Ο ελεγκτής του δικτύου έχει μια συνολική άποψη της τοπολογίας και της κατάστασης του δικτύου όπως επίσης και των απαιτήσεων των εφαρμογών,
2. Προγραμματισμό και ευελιξία. Το επίπεδο των δεδομένων χαρακτηρίζεται ως δυναμικό και μπορεί να προγραμματιστεί βελτιώνοντας έτσι την κατανομή των πόρων του δικτύου,
3. Είναι «ανοικτό» καθώς επιτρέπει την επικοινωνία ανάμεσα στον ελεγκτή και στις συσκευές προώθησης. Να σημειωθεί ότι οι συσκευές είναι ανεξάρτητες από τους προμηθευτές τους.



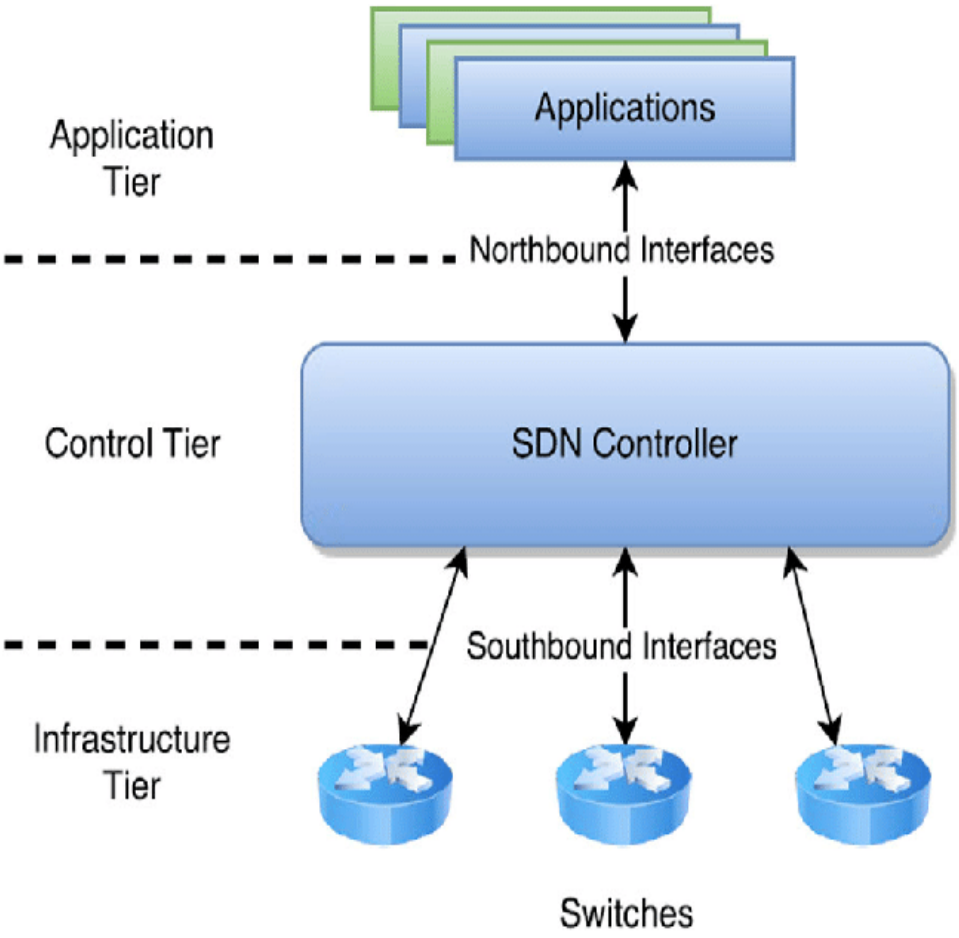
Εικόνα 5.. Απεικόνιση υψηλού επιπέδου και ανάπτυξης επιπέδου ελέγχου και δεδομένων (Akyildiz et al., 2014).

2.2.1 Αρχιτεκτονική των SDN

Το δίκτυο SDN αποτελείται από τρία επίπεδα:

1. Επίπεδο δεδομένων,
2. Επίπεδο ελέγχου,
3. Επίπεδο εφαρμογής,

Η αρχιτεκτονική του δικτύου για κάθε επίπεδο παρουσιάζεται στην παρακάτω Εικόνα 6.



Εικόνα 6. Αρχιτεκτονική SDN (EURASIP, 2018)

Η αρχιτεκτονική των SDN ορίζει το επίπεδο δεδομένων ως το χαμηλότερο επίπεδο το οποίο αποτελείται από στοιχεία προώθησης του δικτύου ανάμεσα στα οποία συγκαταλέγονται οι διακόπτες και οι λογικοί διακόπτες. Οι διακόπτες λογισμικού είναι κατά βάση εικονικοί διακόπτες που βασίζονται σε κοινή λειτουργία συστημάτων όπως το Linux. Υλοποιήσεις που αφορούν σε εικονικούς διακόπτες είναι οι Indigo, Open Switch και Pantou (Xie et al, 2018).

Το κύριο έργο των διακοπτών όσο αφορά το επίπεδο δεδομένων στοχεύει στην τροποποίηση των πακέτων βασισμένο πάντα στους κανόνες ροής που λαμβάνει ο

ελεγκτής στο επίπεδο ελέγχου. Ο SDN ελεγκτής αποτελεί και το κύριο συστατικό της αρχιτεκτονικής των SDN δικτύων. Στην ουσία πρόκειται για τον κεντρικό ελεγκτή μέσω του οποίου παρέχεται δυναμικός προγραμματισμός στους πόρους του δικτύου, κρατώντας ενημερωμένους πλήρως τους κανόνες ροής καθιστώντας έτσι την διαχείριση του δικτύου ευέλικτη. Μέσα από το πρωτόκολλο Open flow το επίπεδο ελέγχου επικοινωνεί με τις συσκευές δικτύου. Οι διακόπτες είναι υπεύθυνοι για την διαδρομή προώθησης των ροών. Οι ροές περιέχουν χαρακτηριστικά όπως η IP πηγής, IP προορισμού και τα δεδομένα κεφαλίδας. Εμπεριέχονται ακόμη χαρακτηριστικά όπως η διάρκεια, ο αριθμός των bytesκ.α. Ο ελεγκτής των SDN παρουσιάζει τη συνολική εικόνα του δικτύου όπου μπορεί να γίνει η ταξινόμηση της κυκλοφορίας. Το εισερχόμενο πακέτο βασίζεται σε ένα σύνολο κανόνων (Hsieh et al., 2018).

2.2.2Open Flow

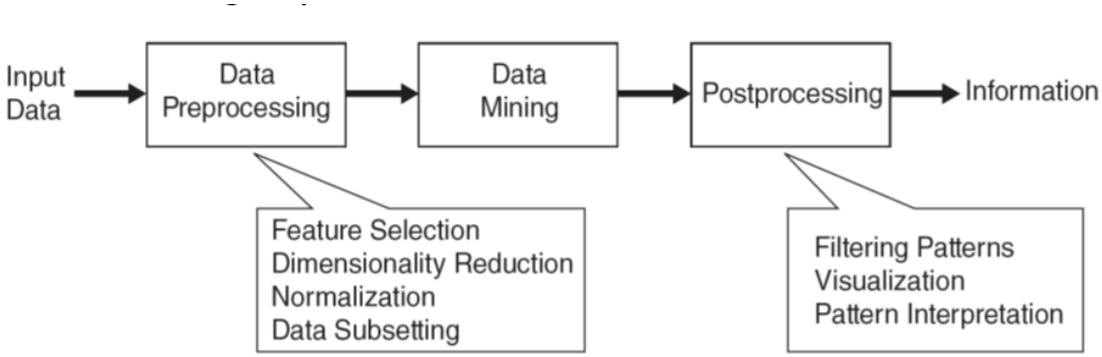
Για τις εφαρμογές SDN, το πρωτόκολλο που επιτρέπει την επικοινωνία ανάμεσα στο επίπεδο ελέγχου και στο επίπεδο δεδομένων είναι το Open flow (OF) (Open flow Switch Specifica, 2014). Το Open flow είναι το πρωτόκολλο κατά βάση, ορίζει μια σειρά μετρητών οι οποίοι οφείλουν να εμπεριέχονται σε μια συσκευή OF. Οι μετρητές είναι υπεύθυνοι για την κυκλοφορία που συντελείται στην συσκευή (Volpato et al., 2018).

2.2Εξόρυξη Δεδομένων(Data Mining)

Η εξόρυξη δεδομένων είναι μια διαδικασία εξαγωγής σημαντικών πληροφοριών η οποία θεωρείται επίσης και ως μια διαδικασία ανακάλυψης γνώσης από την οποία τα δεδομένα υφίστανται ανάλυση (Bharati, 2010).

Η εξόρυξη δεδομένων είναι μια τεχνολογία που συνδυάζει παραδοσιακές μεθόδους ανάλυσης δεδομένων με εξελιγμένους αλγόριθμους για την επεξεργασία μεγάλου όγκου δεδομένων. Έχει ανοίξει συναρπαστικές ευκαιρίες για εξερεύνηση και ανάλυση νέων τύπων δεδομένων και για την ανάλυση παλαιών τύπων δεδομένων με

νέους τρόπους. Οι τεχνικές εξόρυξης δεδομένων μπορούν να χρησιμοποιηθούν για να υποστηρίξουν ένα ευρύ φάσμα επιχειρήσεων εφαρμογές νοημοσύνης όπως προφίλ πελατών, στοχευμένο μάρκετινγκ, ροή εργασιών διαχείριση, διάταξη καταστήματος και ανίχνευση απάτης.

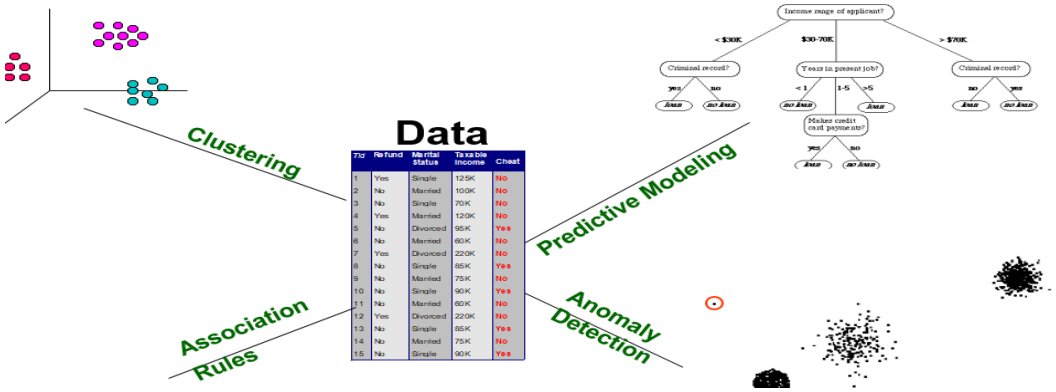


Εικόνα3. Data Mining (Tan et al., 2020)

Να σημειωθεί ότι υπήρξε μεγάλη αύξηση του όγκου των δεδομένων σε εμπορικές και επιστημονικές βάσεις εξαιτίας της προόδου της τεχνολογίας στην παραγωγή και στη συλλογή των δεδομένων ωστόσο, τα δεδομένα που συλλέγονται έχουν αξία όταν εξυπηρετούν τον σκοπό που συλλέγονται. Οι τεχνικές εξόρυξης δεδομένων στοχεύουν στο να ανευρεθούν όλα εκείνα τα μοτίβα τα οποία μέχρι πρότινος ήταν άγνωστα. Όταν αυτά εντοπιστούν μπορούν να χρησιμοποιηθούν στο να ληφθούν οι απαραίτητες αποφάσεις που θα οδηγήσουν στην ανάπτυξη των επιχειρήσεων. Οι τεχνικές αυτές περιλαμβάνουν τρία στάδια (Bharati, 2010):

1. Εξερεύνηση
2. Αναγνώριση Προτύπων
3. Ανάπτυξη

Στο πρώτο στάδιο αυτό της εξερεύνησης, τα δεδομένα υφίστανται καθαρισμό και μετατρέπονται σε άλλη μορφή. Εν συνεχεία, γίνεται προσδιορισμός των σημαντικών μεταβλητών και καθορίζεται η φύση των μεταβλητών δεδομένου του προβλήματος που εξετάζεται. Στο δεύτερο στάδιο, εκείνο της αναγνώρισης των προτύπων όπου μόλις διερευνηθούν, τελειοποιηθούν και οριστούν τα δεδομένα για τις συγκεκριμένες μεταβλητές ακολουθεί ο σχηματισμός της αναγνώρισης προτύπων, και στην συνέχεια γίνεται ο προσδιορισμός και η επιλογή των μοτίβων με σκοπό την καλύτερη πρόβλεψη. Στο τρίτο και τελευταίο στάδιο, τα μοτίβα αναπτύσσονται για το επιθυμητό αποτέλεσμα (Bharati,2010).



Εικόνα 4. Data Mining Tasks (Tan et al., 2020)

Ο ελεγκτής SDN παρέχει την πλήρη προβολή του δικτύου και τη συλλογή κίνησης. Έτσι, η ευφυΐα στην ταξινόμηση της κυκλοφορίας εκτελείται από τον κεντρικό ελεγκτή. Με βάση την εφαρμογή, οι τεχνικές μάθησης μπορεί να ποικίλλουν για να ανταποκριθούν στις απαιτήσεις της κατάστασης στην οποία βρίσκεται το δίκτυο.

2.3Τεχνικές Εξόρυξης Δεδομένων

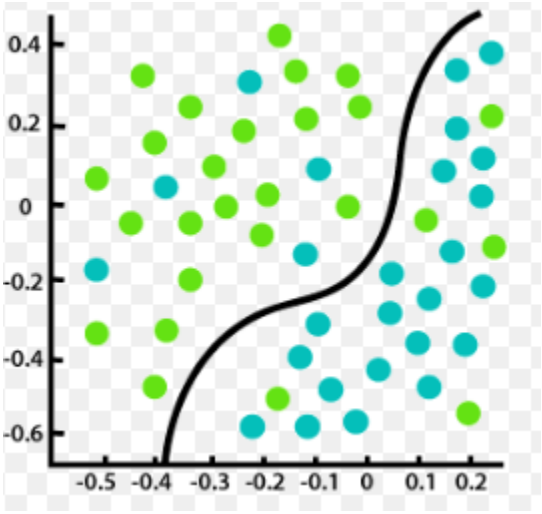
Υπάρχουν διάφοροι αλγόριθμοι και τεχνικές εξόρυξης δεδομένων που χρησιμοποιούνται για την εξόρυξη της γνώσης από τις βάσεις δεδομένων. Αυτές είναι οι ακόλουθες (Bharati, 2010).:

- Κατηγοριοποίηση (Classification),
- Συσταδοποίηση (Clustering),
- Παλινδρόμηση (Regression),
- Τεχνητή νοημοσύνη (Artificial intelligence),
- Νευρωνικά δίκτυα (Neuron networks),
- Κανόνες συσχέτισης (Association rules),
- Δέντρα αποφάσεων (Decision Trees),
- Γενετικός Αλγόριθμος (Genetic Algorithm),
- Πλησιέστερος Γείτονας (Nearest Neighbor),
- Naïve Bayes,
- Linearκαι Logistic Regression.

2.3.1Ταξινόμηση (Classification)

Η ταξινόμηση αποτελεί την πιο συχνά χρησιμοποιούμενη μέθοδο εξόρυξης δεδομένων μέσα από την οποία γίνεται χρήση ενός συνόλου προταξινομημένων δεδομένων για να αναπτυχθεί ένα μοντέλο που θα ταξινομεί τις εγγραφές. Ειδικότερα, οι εφαρμογές που ανιχνεύουν την απάτη και τον κίνδυνο θεωρούνται ιδανικές για ανάλυση. Οι εφαρμογές ανίχνευσης απάτης και πιστωτικού κινδύνου είναι ιδιαίτερα κατάλληλες για αυτόν τον τύπο ανάλυσης. Αυτή η προσέγγιση χρησιμοποιείται συχνά από τα δέντρα αποφάσεων και από τους αλγόριθμους ταξινόμησης και έχουν την βάση τους στα Νευρωνικά δίκτυα. Ωστόσο, μια διαδικασία ταξινόμησης δεδομένων εμπεριέχει την μάθηση και την ταξινόμηση. Στον τομέα της μάθησης, τα δεδομένα εκπαίδευσης υπόκεινται σε ανάλυση με έναν αλγόριθμο ταξινόμησης. Τα δεδομένα ταξινόμησης δοκιμών χρησιμοποιούνται κατά κύριο λόγο ώστε να εκτιμηθεί η ακρίβεια που διέπει τους κανόνες ταξινόμησης. Αν οι κανόνες είναι αποδεκτοί τότε εφαρμόζονται σε ποικιλία δεδομένων. Για παράδειγμα σε μια εφαρμογή ανίχνευσης απάτης θα εμπεριέχονταν

πλήρη αρχεία τόσο εκείνα που περιλαμβάνουν τις δόλιες όσο και τις έγκυρες δραστηριότητες με βάση την καταγραφή.



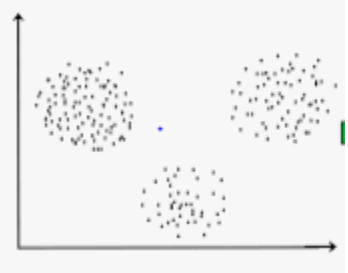
Εικόνα 7. Classification – Ταξινόμηση (Javapoint.com))

Από τους πιο γνωστούς ταξινομητές της μηχανικής μάθησης είναι ο Naïve Bayes. Είναι ένας εποπτευόμενος αλγόριθμος μηχανικής μάθησης ο οποίος χρησιμοποιείται για εργασίες ταξινόμησης. Επιδιώκει να μοντελοποιήσει την κατανομή εισροών μιας συγκεκριμένης τάξης ή μιας κατηγορίας. Ένας αλγόριθμος εκπαίδευσης του ταξινομητή κάνει χρήση των προταξινομημένων δεδομένων προκειμένου να προσδιοριστούν όλες εκείνες οι παράμετροι που είναι αναγκαίες για να υπάρξει σωστή διάκριση. Εν συνεχεία, ο αλγόριθμος προβαίνει σε κωδικοποίηση των παραμέτρων σε ένα μοντέλο που ορίζεται ως ταξινομητής. Οι τύποι των μοντέλων ταξινόμησης περιλαμβάνουν (Bharati, 2010):

1. Ταξινόμηση με επαγωγή δέντρου αποφάσεων
2. Ταξινόμηση Bayes
3. Νευρωνικά δίκτυα
4. Υποστήριξη Vector Machines
5. Ταξινόμηση βάση ενώσεων

2.3.2Συσταδοποίηση (Clustering)

Η Συσταδοποίηση μπορεί να χαρακτηριστεί και ως η αναγνώριση παρόμοιων κατηγοριών αντικειμένων με βάση κάποιο μέτρο ομοιότητας. Τα στοιχεία που παρουσιάζουν κάποια ομοιότητα κατατάσσονται στην ίδια ομάδα. Τα στοιχεία των συστάδων πρέπει να είναι όμοια ή να σχετίζονται και διαφορετικά μη σχετιζόμενα από στοιχεία άλλων συστάδων. Τάξεις ή ομάδες με εννοιολογική σημασία των αντικειμένων που μοιράζονται κοινά χαρακτηριστικά, παίζουν σημαντικό ρόλο στο πώς οι άνθρωποι αναλύουν και περιγράφουν τον κόσμο (Tan et al., 2020). Μέσα από τη χρήση τεχνικών συσταδοποίησης υπάρχει η δυνατότητα προσδιορισμού πυκνών και αραιών περιοχών στον χώρο των αντικειμένων ανακαλύπτοντας έτσι το συνολικό μοτίβο κατανομής και συσχετίσεων των χαρακτηριστικών που διέπουν τα δεδομένα.



Εικόνα 8. Clustering-Συσταδοποίηση (Geeks forGeeks)

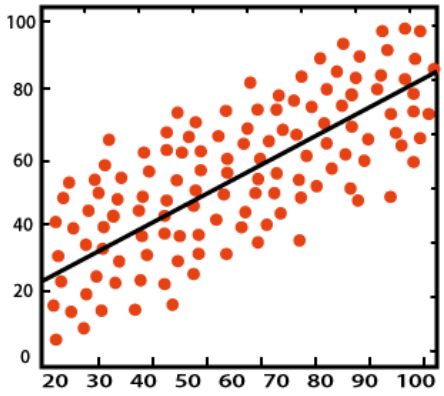
Η συγκεκριμένη τεχνική χρησιμοποιείται για τη διάκριση ομάδων και κατηγοριών αντικειμένων αλλά θεωρείται δαπανηρή. Συνεπώς, η Συσταδοποίηση χρησιμοποιείται ως προσέγγιση προεπεξεργασίας τόσο για την επιλογή όσο και για την ταξινόμηση ενός υποσυνόλου χαρακτηριστικών. Στους τύπους μεθόδων συσταδοποίησης συγκαταλέγονται (Bharati, 2010):

1. Μέθοδος κατάτμησης,
2. Ιεραρχικές συσσωματωτικές (διαιρετικές) μέθοδοι,

3. Μέθοδοι που βασίζονται στην πυκνότητα,
4. Μέθοδοι που βασίζονται σε πλέγμα,
5. Μέθοδοι που βασίζονται σε μοντέλα.

2.3.3 Παλινδρόμηση (Regression)

Η τεχνική της παλινδρόμησης μπορεί να χρησιμοποιηθεί για προβλέψεις. Μέσα από την ανάλυση που προσφέρει η συγκεκριμένη τεχνική δίνεται η δυνατότητα μοντελοποίησης της σχέσης ανάμεσα σε μια ή περισσότερες ανεξάρτητες και εξαρτημένες μεταβλητές.



Εικόνα 9. Regression –Οπισθοδρόμηση (Geeks for Geek)

Στην εξόρυξη δεδομένων, οι ανεξάρτητες μεταβλητές αποτελούν χαρακτηριστικά τα οποία είναι ήδη γνωστά και οι μεταβλητές απόκρισης αποτελούν το ζητούμενο πρόβλεψης. Ωστόσο, πολλά από τα προβλήματα στον πραγματικό κόσμο δεν είναι απλώς προβλέψεις (όγκος πωλήσεων, αποθέματα, κ.α) (Bharati, 2010). Για παράδειγμα η γραμμική παλινδρόμηση (linear regression) χρησιμοποιείται για την πρόβλεψη της τιμής μιας μεταβλητής με βάση την τιμή μιας άλλης μεταβλητής. Η μεταβλητή που θέλουμε να προβλέψουμε ονομάζεται εξαρτημένη μεταβλητή και η μεταβλητή που χρησιμοποιείται για την πρόβλεψη της άλλης μεταβλητής ονομάζεται ανεξάρτητη. Ενώ η λογιστική

παλινδρόμηση αφορά την διαδικασία μοντελοποίησης της πιθανότητας ενός διακριτού αποτελέσματος με βάση την μεταβλητή εισόδου. Η πιο κοινή λογιστική παλινδρόμηση διαμορφώνει δύο τιμές yes or no ή true ή false.

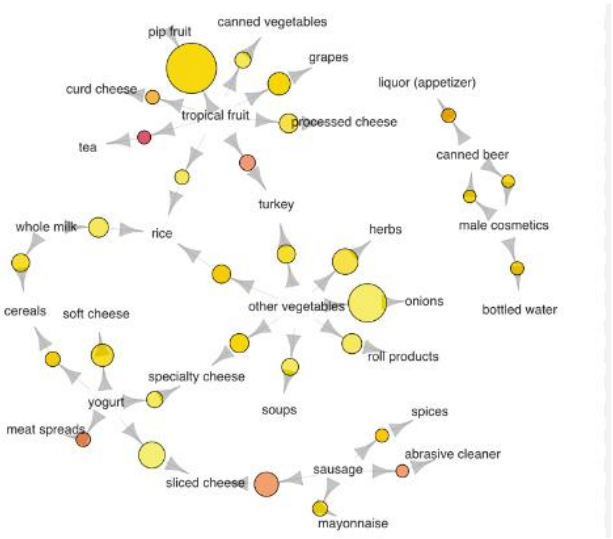
Η πρόβλεψη σε χαρακτηριστικά όπως οι τιμές αποτελούν δύσκολο πεδίο πρόβλεψης λόγω του ότι αποτελούν παράγοντες σύνθετων καταστάσεων. Εξαιτίας αυτού, σύνθετες τεχνικές όπως η λογιστική παλινδρόμηση, τα δέντρα αποφάσεων και τα νευρωνικά δίκτυα, θεωρούνται απαραίτητες για τέτοιου είδους προβλέψεις. Στις μεθόδους παλινδρόμησης συγκαταλέγονται τα παρακάτω (Bharati, 2010):

1. Γραμμική Παλινδρόμηση (Linear Regression),
2. Πολυμεταβλητή Γραμμική Παλινδρόμηση,
3. Μη γραμμική παλινδρόμηση,
4. Πολυμεταβλητή μη γραμμική παλινδρόμηση.

2.3.4Κανόνες σύνδεσης (Association rules)

Οι κανόνες συσχέτισης αφορούν συνήθως ευρήματα σε ένα σύνολο στοιχείων ενός μεγάλου συνόλου δεδομένων. Η εν λόγω τεχνική αποτελεί μια σημαντικά βοηθητική μέθοδο προκειμένου οι επιχειρήσεις να προβούν στην λήψη αποφάσεων σε θέματα σχεδιασμού καταλόγων, cross marketing και ανάλυση συμπεριφοράς των πελατών. Βέβαια, οι κανόνες συσχέτισης οφείλουν να δημιουργούν κανόνες που να παρέχουν ασφάλεια και σιγουριά. Στους κανόνες συσχέτισης περιλαμβάνονται(Bharati, 2010).:

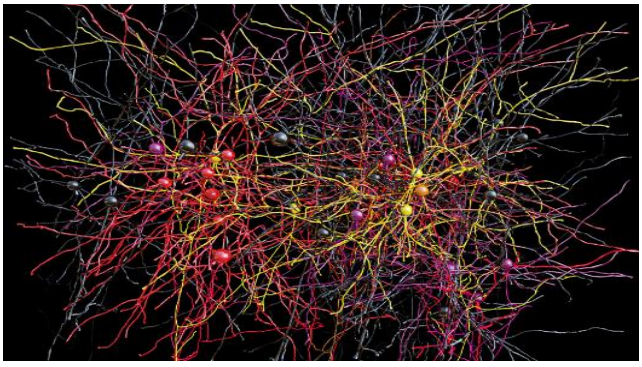
1. Κανόνας συσχέτισης πολλαπλών επιπέδων,
2. Πολυδιάστατος κανόνας συσχέτισης,
3. Κανόνας ποσοτικής συσχέτισης.



Εικόνα 10. Association Rules Κανόνες Συσχέτισης (Kdnuggets.com)

2.3.5Νευρωνικάδίκτυα (Neuron Networks)

Ένα νευρωνικό δίκτυο αποτελεί ένα σύνολο από συνδεδεμένες μονάδες εισόδου και εξόδου. Κάθε συσκευή έχει και ένα βάρος (weight). Στην φάση της εκμάθησης το νευρωνικό δίκτυο μαθαίνει μέσα από την προσαρμογή των βαρών για να μπορεί να προβλέψει την σωστή σειρά των ετικετών στις πλειάδες εισόδου.



Εικόνα 11. Neuron Network-Νευρωνικά Δίκτυα (Weired.com)

Τα νευρωνικά δίκτυα έχουν την ικανότητα να αντλούν νόημα από περίπλοκα στοιχεία και να εξάγουν μοτίβα και τάσεις που πολλές φορές δεν είναι αντιληπτά/ες στους περισσότερους ανθρώπους αλλά και στις τεχνικές υπολογιστών. Ωστόσο,

θεωρούνται κατάλληλα για συνεχείς τιμές εισόδου/εξόδου. Στους τύπους των νευρωνικών δικτύων συγκαταλέγονται(Bharati, 2010).:

1. Πίσω διάδοση.

2.3.6Εφαρμογές Εξόρυξης δεδομένων

Η εξόρυξη δεδομένων αποτελεί μια σχετικά νέα τεχνολογία. Ωστόσο, χρησιμοποιείται σε εφαρμογές στη βιομηχανική παραγωγή. Σε αυτές συγκαταλέγονται τα καταστήματα λιανικής πώλησης, τα νοσοκομεία, οι τράπεζες και οι ασφαλιστικές εταιρείες. Πολλές εξ αυτών, συνδυάζουν την εξόρυξη δεδομένων με εργαλεία που είναι χρήσιμα για την εξαγωγή στατιστικών στοιχείων και την αναγνώριση προτύπων. Βέβαια, η εξόρυξη δεδομένων δύναται να χρησιμοποιηθεί και για την εύρεση μοτίβων και συνδέσεων που σε άλλες περιπτώσεις θα ήταν δύσκολη η ανεύρεση τους. Είναι μια ευρέως γνωστή και δημοφιλής τεχνολογία καθώς επιτρέπει σε όλους όσους εμπλέκονται να μάθουν περισσότερα και να λάβουν καλύτερες αποφάσεις (Bharati, 2010).

2.4 Σχετικές Εργασίες

Η ταξινόμηση της κυκλοφορίας με ακρίβεια είναι σημαντική σε δραστηριότητες του δικτύου όπως η παρακολούθηση της ασφάλειας, η μηχανική της κυκλοφορίας, η ανίχνευση σφαλμάτων, η λογιστική χρήση του δικτύου, η τιμολόγηση και η παροχή της διαφοροποίησης της ποιότητας των υπηρεσιών QoS των παραμέτρων των υπηρεσιών του δικτύου (Raikar et al., 2020).

Οι τεχνικές ταξινόμησης της κυκλοφορίας που χρησιμοποιούνταν ήταν οι αριθμοί των θυρών και η βαθιά επιθεώρηση των πακέτων (DPI-Deep packet inspection). Στην πρώτη τεχνική, οι δυναμικές θύρες χρησιμοποιούνταν για την ανάπτυξη των εφαρμογών όπου δηλαδή δεν μπορούσε να γίνει ταξινόμηση βάση εφαρμοσμένης θύρας. Στην δεύτερη τεχνική, στην βαθιά επιθεώρηση του πακέτου πραγματοποιείται αναγνώριση μοτίβου για την εφαρμογή. Οι κανονικές εκφράσεις χρησιμοποιούνται για την

αναγνώριση προτύπων. Η ραγδαία αύξηση των εφαρμογών δημιούργησε πολλά προβλήματα στην ταξινόμηση με βάση τα πρότυπα καθώς δεν μπορούσε να γίνει ενημέρωση του μοτίβου για μεγάλο αριθμό εφαρμογών. Ωστόσο, η κρυπτογραφημένη κίνηση δεν μπορεί να ταξινομηθεί χρησιμοποιώντας τον μηχανισμό DPI (Raikar et al., 2020). Η αρχιτεκτονική της ταξινόμησης κυκλοφορίας των δικτύων εμπεριέχει τα εξής: τον ταξινομητή, την ταυτότητα και το ωφέλιμο φορτίο. Η αξιολόγηση και η ανάλυση της απόδοσης μετράτε μέσα από τον χρόνο που αποκρίνονται οι συσκευές(Ng, Hayes, Seah, 2015).

Οι Tsoulos et al., (2021), προτείνουν ένα γενετικό εργαλείο προγραμματισμού Gen class, για την ταξινόμηση των δεδομένων που βασίζεται στην τεχνική grammatical evolution και έχει σχεδιαστεί για να εκμεταλλεύεται πολυπύρηνα υπολογιστικά συστήματα χρησιμοποιώντας το την βιβλιοθήκη Open MP Το εργαλείο κατασκευάζει προγράμματα ταξινόμησης σε μια γλώσσα προγραμματισμού τύπου C για την ταξινόμηση των δεδομένων εισόδου παράγοντας απλούς κανόνες και μετά τον τερματισμό το εργαλείο παράγει τους κανόνες ταξινόμησης σε αρχεία C και Python . Η τεχνική Genclass συγκρίθηκε με άλλες τεχνικές ταξινόμησης δεδομένων όπως η BFGS (χρησιμοποιείται ευρέως για την επίλυση μη-γραμμικών προβλημάτων βελτιστοποίησης χωρίς περιορισμούς είναι μια μέθοδος εύρεσης των συντελεστών βαρύτητας των συνάψεων w με ελαχιστοποίηση του σφάλματος από παραδείγματα. Η μέθοδος κατασκευάζει μια προσέγγιση του Εσσιανού (Hessian) πίνακα των δεύτερων παραγώγων για τη συνάρτηση ελαχιστοποίησης, την τεχνική RBF (μπορεί να εκτιμήσει τις πυκνότητες υπό όρους κατηγορίας σε προβλήματα ταξινόμησης. που σε κάθε του έξοδο υπολογίζει την κατανομή των προτύπων μιας κατηγορίας) και την τεχνική MLP (χρησιμοποιεί μια εποπτευόμενη τεχνική μάθησης που ονομάζεται backpropagation για την εκπαίδευση του δικτύου, είναι μια τροποποίηση του τυπικού γραμμικού perceptron και μπορεί να διακρίνει δεδομένα που δεν είναι γραμμικά) δείχνοντας την υπεροχή της αποδίδοντας πολύ ελπιδοφόρα αποτελέσματα.

Οι Jamuna et al., (2013), εξετάζουν εποπτευόμενες τεχνικές μάθησης όπως δέντρα αποφάσεων, Naïve Bayes και SVM αλλά και μοντέλα μάθησης χωρίς επίβλεψη

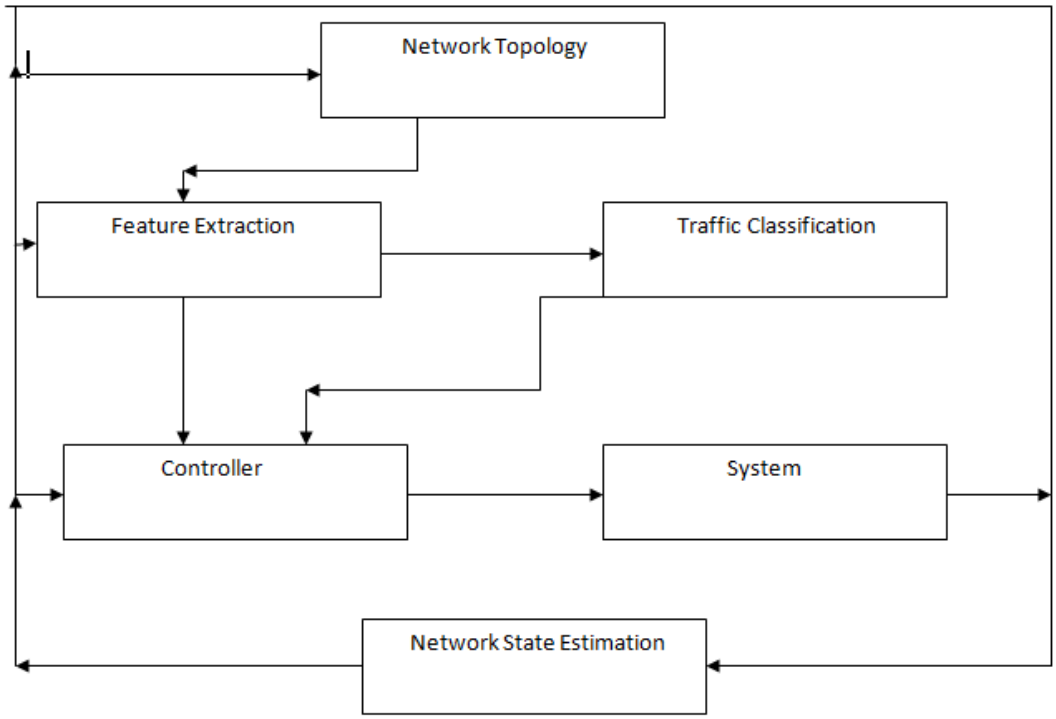
όπως (K-means, DBSCAN, Auto class). Οι Yan et al., (2018), προτείνουν την αρχιτεκτονική της ταξινόμησης κυκλοφορίας σε ένα δίκτυο SDN που βασίζεται σε ένα μηχανισμό ομαδοποίησης. Η ταξινόμηση της κυκλοφορίας πραγματοποιείται όταν λαμβάνεται υπόψη το δίκτυο που δημιουργήθηκε με την τεχνολογία SDN. Οι Li et al., (2014), παρουσιάζουν ένα σενάριο ενός δικτύου που αφορά πανεπιστημιούπολη στο οποίο έχει ρυθμιστεί η πλατφόρμα SDN για την ταξινόμηση της κυκλοφορίας.

Οι τεχνικές αυτές δεν παρείχαν την απαιτούμενη ακρίβεια κι αυτό λόγω της μεγάλης ποσότητας συσκευών και ροών που εντοπίζονταν στο δίκτυο. Η ολοκλήρωση των SDN δικτύων και η εισαγωγή τεχνολογίας της μηχανικής μάθησης έρχονται να άρουν τις παραπάνω δυσκολίες. Η εφαρμογή τους στην ουσία προβαίνει στην ταξινόμηση της κυκλοφορίας των δεδομένων και έχει την βάση τους σε πλατφόρμες δικτύου που ορίζονται από ανάλογα λογισμικά. Τα αποτυπώματα της κυκλοφορίας του δικτύου που δημιουργούνται κατασκευάζουν ροές οι οποίες στην συνέχεια αποστέλλονται για τις σχετικές προβλέψεις. Ακόμη, τα δεδομένα που παρέχουν την ανάλυση κίνησης σε ένα δίκτυο προσφέρουν εξαιρετικά χρήσιμα στοιχεία στον διαχειριστή του δικτύου για να προβεί στην λήψη αποφάσεων με σκοπό την καλύτερη κατανομή των πόρων του. Με την συνδρομή των τεχνολογιών όπως το Fog computing, το Cloud computing και το IoT, η διαδικασία κατανομής των πόρων γίνεται ευκολότερη καθώς είναι αυξανόμενη η ζήτηση για την διαχείριση των πόρων. Οι χρήστες του εκάστοτε δικτύου επιθυμούν την γρήγορη παράδοση υπηρεσιών, την επεκτασιμότητα, και την ποιότητα εμπειρίας (QoE). Ο πάροχος υπηρεσιών στοχεύει στην μεγιστοποίηση των πόρων μέσα από διάφορους μηχανισμούς QoE ελαχιστοποιώντας το λειτουργικό κόστος. Η κυκλοφορία του δικτύου στα υπάρχοντα συστήματα σε θέματα πολυπλοκότητας της ταξινόμησης βασίζεται σε κώδικα, ενώ στην μηχανική μάθηση η πολυπλοκότητα εντοπίζεται στον κώδικα και στα ίχνη του δικτύου που εφαρμόστηκε ο αλγόριθμος. Ένα από τα βασικά στοιχεία των αλγορίθμων της μηχανικής μάθησης είναι η ικανότητα της μάθησης που προσφέρει. Η εν λόγω ικανότητα προέκυψε από την διαδικασία της τελειοποίησης. Χαρακτηριστικά όπως η διάρκεια, το μήκος του πακέτου, και ο χρόνος ενδιάμεσης άφιξης του πακέτου αποτελούν κάποιες από τις κατηγορίες

κυκλοφορίας του δικτύου. Οι αλγόριθμοι μηχανικής μάθησης χρησιμεύουν στο να προσδιοριστούν τα μοτίβα στα δεδομένα του δικτύου που καταγράφονται. Ποικιλία εφαρμογών όπως βίντεο, μεταφορά αρχείων, και e-mail αποτελούν μεγάλη πρόκληση. Ωστόσο, για να υπάρξει η καλύτερη χρήση του δικτύου απαραίτητη προϋπόθεση είναι η ύπαρξη εργαλείου για την αυξανόμενη κυκλοφορία του δικτύου. Η νοημοσύνη πρέπει να συμπεριληφθεί στις συσκευές δικτύωσης προκειμένου να υπάρξει η ανάλογη ευκολία σε θέματα οργάνωσης, βελτιστοποίησης, συντήρησης και διαχείρισης. Βέβαια, η κατανεμημένη φύση του συστήματος και η μηχανική εκμάθηση των συσκευών είναι δύσκολο έργο. Οι πλατφόρμες SDN δίνουν αυτές τις ευκαιρίες. Η συμπερίληψη νοημοσύνης σε μια συσκευή δικτύου δίνει λύση. Η υλοποίηση όμως υποδομών όπως η GPU και η επεξεργασία δεδομένων όπως το Spark και το Hadoop σε συνδυασμό με βιβλιοθήκες μηχανικής μάθησης δίνει την δυνατότητα της εισαγωγής νοημοσύνης σε συσκευές δικτύωσης. Στο παραδοσιακό μοντέλο δικτύωσης η διαχείριση των συσκευών δικτύων γίνεται με την χρήση της διεπαφής γραμμής εντολών και σενάρια. Η πρόοδος που έχει συντελεστεί στα SDN δίκτυα είναι ότι η διαχείριση γίνεται με ημι-αυτοματισμό του δικτύου ενώ στην μηχανική μάθηση τα μοτίβα κυκλοφορίας βελτιώνουν τον έλεγχο του δικτύου μεγιστοποιώντας την διακίνηση (Raikar et al., 2020).

Η αρχιτεκτονική των δικτύων SDN, καθορίζεται από το λογισμικό και η χρήση του αφορά την ταξινόμηση της κυκλοφορίας του δικτύου η οποία δημιουργείται από την συσκευή του πελάτη. Στα δίκτυα SDN επικρατεί διαχωρισμός στο επίπεδο δεδομένων και στο επίπεδο ελέγχου. Το Openflow είναι πρωτόκολλο το οποίο αποτελεί την διεπαφή ανάμεσα στην συσκευή και στον ελεγκτή. Οι διεπαφές χρησιμοποιούνται για την επικοινωνία ανάμεσα στις εφαρμογές του δικτύου και τον ελεγκτή. Η τοπολογία του δικτύου αποστέλλεται στον ελεγκτή για να παρθούν αποφάσεις και να γίνει η διαχείριση των συσκευών του δικτύου. Ο μηχανισμός ελέγχου σε συνδυασμό με την ανάλυση δεδομένων θεωρείται ο κατάλληλος για να παρέχει όλες εκείνες τις εντολές ελέγχου ώστε να υπάρξει δυναμική διαχείριση του δικτύου. Από την άλλη πλευρά ένας μηχανισμός ανάδρασης περιλαμβάνει τις ανάλογες ενημερώσεις για την τοπολογία του δικτύου (αποτυχία συνδέσμου). Έτσι, οι πληροφορίες του δικτύου που προκύπτουν θεωρούνται

σημαντικές για την λήψη αποφάσεων, την κατασκευή αλλά και τον χειρισμό των συσκευών που υπάρχουν σε ένα δίκτυο (Εικόνα 1.) (Jamuna &Edwards, 2013).

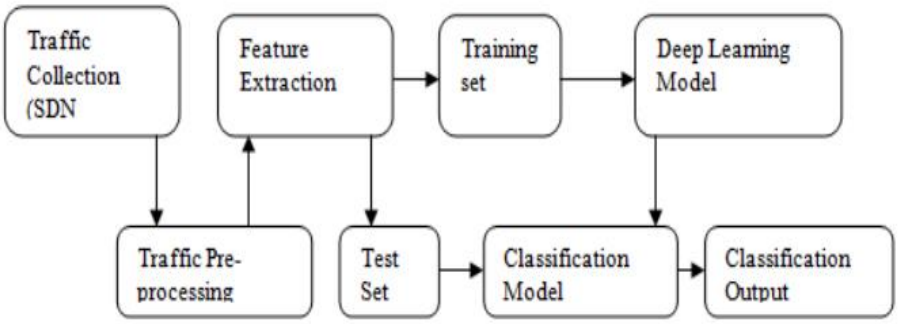


Εικόνα 2. Πλαίσιο για την ταξινόμηση της κυκλοφορίας σε δίκτυα καθορισμένα από λογισμικό (Jamuna&Edwards, 2013)

Η ροή εργασίας για την ταξινόμηση της κυκλοφορίας περιλαμβάνει τη συλλογή δεδομένων, την προεπεξεργασία, την επισήμανση, την κατασκευή, την επικύρωση και την πρόβλεψη του μοντέλου. (Mulla et. al.,2019).

Σύμφωνα με τους Yu et al., (2018), οι παράμετροι QoS είναι επίσης σημαντικά κριτήρια για την ταξινόμηση της κυκλοφορίας σε ένα δίκτυο SDN. Οι Xie et al., (2019), θεωρούν ότι η βελτιστοποίηση μιας διαδρομής σε ένα δίκτυο SDN περιλαμβάνει την

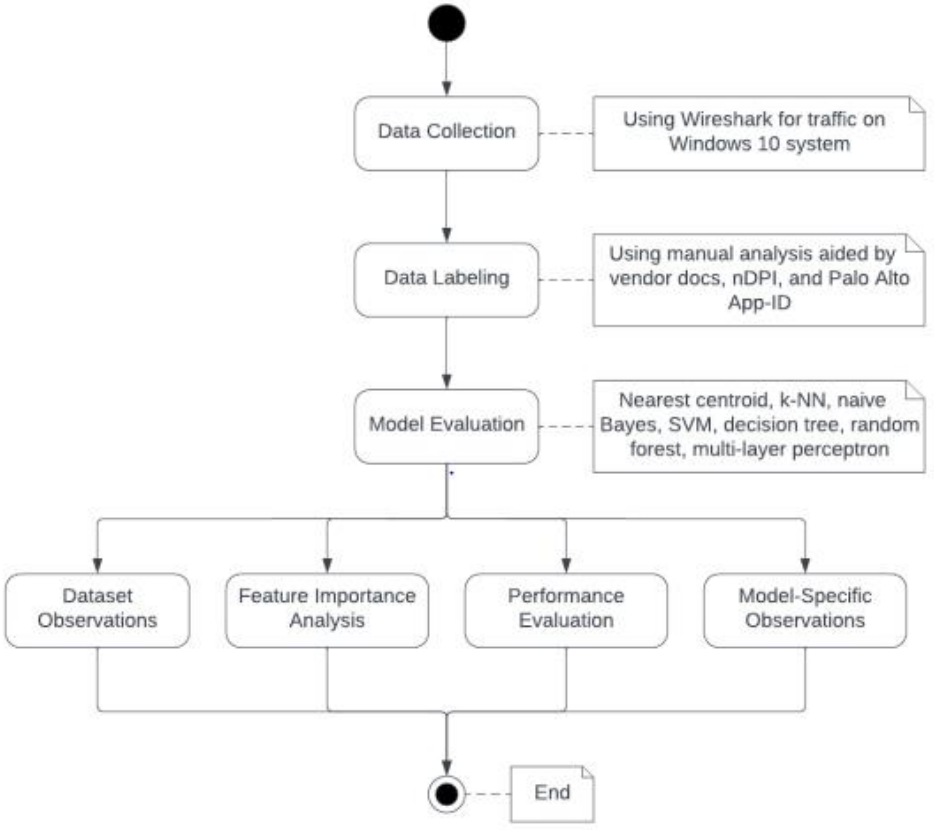
υιοθέτηση εννοιών μηχανικής μάθησης με τις οποίες θα υπάρξει καλύτερη διαχείριση πόρων και ταξινόμηση της κυκλοφορίας.



Εικόνα 7..Μοντέλο Εικονικής Ταξινόμησης (Xie et al., 2018)

Κεφάλαιο 3ο. Μεθοδολογία και χρήσιμα εργαλεία

Στην επιστήμη των δεδομένων και τη μηχανική μάθηση, υπάρχει μια ευρεία διαδικασία που πρέπει να εκτελεστεί να δημιουργήσει αποτελεσματικά ένα μοντέλο μηχανικής μάθησης που μπορεί να ταξινομήσει με ακρίβεια τα δεδομένα. Αυτή η διαδικασία, όπως θα παρουσιαστεί παρακάτω, περιλαμβάνει συλλογή δεδομένων, προετοιμασία δεδομένων, μοντελοποίηση και μοντέλο εκτίμηση. Περιλαμβάνεται ένα διάγραμμα ροής που δείχνει την ευρεία διαδικασία αυτής της εργασίας από την αρχή μέχρι το τέλος παρακάτω.



Διάγραμμα 1. Επισκόπηση της Διαδικασίας

3.1Μεθοδολογία

- **Δεδομένα :**Τα δεδομένα που χρησιμοποιήθηκαν προήλθαν από ένα dataset που διατίθεται από την πλατφόρμα Kaggle [2]και περιλαμβάνει στοιχεία για την κίνηση ενός δικτύου SDN.
- **Τεχνικές:** οι τεχνικές που χρησιμοποιήθηκαν αφορούν την τεχνική της κατηγοριοποίησης και εφαρμόστηκαν σε τρεις αλγορίθμους μηχανικής μάθησης Naïve Bayes, Logistic Regression και Linear Regression
- **Εργαλεία**: το εργαλείο υλοποίησης που χρησιμοποιήθηκε για την ανάπτυξη των πειραματικών μετρήσεων είναι το Google Colab στο οποίο με την χρήση του Jupiter Notebook δημιουργήθηκε ο κώδικας σε Python και για τους τρεις αλγορίθμους μηχανικής μάθησης. Επίσης, στα εργαλεία ανάπτυξης συμπεριλήφθηκαν η javaκαι το ApacheSpark.
- **Αποτελέσματα /συμπέρασμα**: από την υλοποίηση αναμένεται ποιος από τους τρεις αλγορίθμους και ποια προσέγγιση δίνει την καλύτερη πρόβλεψη για τα δεδομένα κίνησης του δικτύου.

Θα αναπτυχθεί ένα μοντέλο προβλέψεων που θα μελετά τις ροές της κίνησης ενός δικτύου SDN κάνοντας χρήση μεθοδολογιών επεξεργασίας δεδομένων μέσα από σύγχρονες αλγοριθμικές και στατιστικές τεχνικές. Επίσης, θα χρησιμοποιηθούν ειδικές τεχνικές οι οποίες είναι αρκετά δημοφιλής τα τελευταία χρόνια οι οποίες επεξεργάζονται μεγάλα δεδομένα προκειμένου να εξαχθούν μοντέλα πρόβλεψης και λήψης αποφάσεων. Ωστόσο να σημειωθεί ότι θα πραγματοποιηθεί αποτελεσματική και επιστημονικά άρτια παρουσίαση και σύνοψη πολύπλοκων δεδομένων και μοντέλων, και βασικές γνώσεις σε κάποια πεδία εφαρμογής. Η εφαρμογή θα υλοποιηθεί σε υπολογιστή, μέσα από την χρήση ενός διαδικτυακού και δυναμικού εργαλείου Google collaborator με προγραμματιστικό μοντέλο του Apache Spark σε γλώσσα Python.

2.βλέπε:https://www.kaggle.com/datasets/jsrojas/ip-network-traffic-flows-labeled-with-87-apps/discussion/277698

3.2 Εργαλεία

Για την υλοποίηση του προβλεπτικού μοντέλου ήταν θεμιτή η χρήση ενός δυναμικού εργαλείου το οποίο λειτουργεί συγκεντρωτικά. Για τον λόγο αυτό, επιλέχθηκε και χρησιμοποιήθηκε το εργαλείο της Google Collaboratory. Πρόκειται για ένα δυναμικό περιβάλλον web με πολλές δυνατότητες για υλοποίηση σχετικών εφαρμογών. Μέσω αυτού του εργαλείου έγινε εγκατάσταση της Java και του ApacheSpark. Έπειτα, με την χρήση σημειωματάριου της εφαρμογής αναπτύχθηκε ο αντίστοιχος κώδικας και για τους τρεις αλγορίθμους μηχανικής μάθησης για την ανάπτυξη του μοντέλου σε γλώσσα Python. Το δυναμικό αυτό περιβάλλον δίνει την δυνατότητα στον χρήστη να τρέξει ξεχωριστά την κάθε υλοποίηση για την παρουσίαση των αποτελεσμάτων. Η επιλογή των παραπάνω εργαλείων έγινε με σκοπό την συγκέντρωση τους σε μια πλατφόρμα για λόγους όπως η ευκολία χρήσης και ανάπτυξης, για λόγους αποτελεσματικότητας και για λόγους παροχής ενός πλούσιου συνόλου εργασιών. Δεν παρουσιάστηκαν σφάλματα και μειονεκτήματα στα εργαλεία που χρησιμοποιήθηκαν κατά την φάση της υλοποίησης.

3.2.1Google Colaboratory

Το Google Colaboratory είναι μια υπηρεσία Cloud που έχει τη βάση της στο σημειωματάριο της Jupyter. Αποτελεί ένα διαδεδομένο εργαλείο στον τομέα της μηχανικής μάθησης αφού προσφέρει πλήρη διαμορφωμένο περιβάλλον, χρόνο εκτέλεσης και εκμάθηση. Το σημειωματάριο του Jupyter που αποτελεί και την κύρια τεχνολογία στην οποία βασίζεται το Colab είναι ένα εργαλείο ανοικτού κώδικα που βασίζεται σε προγράμματα περιήγησης με ενσωματωμένες γλώσσες, βιβλιοθήκες και εργαλεία για οπτικοποίηση. Το σημειωματάριο τουJupyterμπορεί να λειτουργήσει είτε τοπικά είτε στο cloud. Κάθε έγγραφο, αποτελείται από κελιά τα οποία περιέχουν μια γλώσσα σεναρίου ή κώδικα σήμανσης και η έξοδος είναι ενσωματωμένη στο έγγραφο. Στα τυπικά αποτελέσματα περιλαμβάνονται κείμενο, πίνακες και γραφικά. Η τεχνολογία αυτή πραγματικά διευκολύνει την εκπόνηση εργασιών και την κοινή χρήση καθώς τα

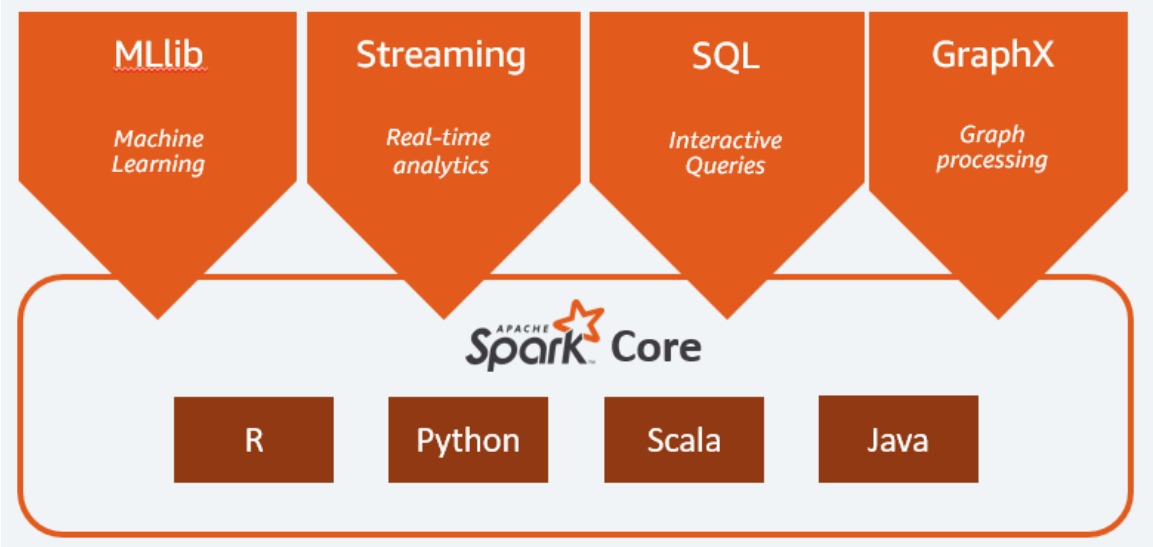
πειράματα παρουσιάζονται με αυτοτελή τρόπο. Μέσα από το Colab οι χρήστες μπορούν να εργαστούν στο ίδιο σημειωματάριο. Προσφέρει προρυθμισμένους χρόνους εκτέλεσης σε Python 2,3 και τις ανάλογες βιβλιοθήκες μηχανικής μάθησης και τεχνητής νοημοσύνης όπως οι Tensor Flow, Kerasκαι Matplotlib. Η υποδομή του Google Colabφιλοξενείται στην πλατφόρμα του Google Cloud (Carneiro et al, 2018).

3.2.2 Η ΠλατφόρμαApacheSpark

Το ApacheSpark είναι ένα πλαίσιο ανοικτού κώδικα που παρέχει μια γρήγορη μηχανή επεξεργασίας δεδομένων που υποστηρίζει εφαρμογές μηχανικής μάθησης και τεχνητής νοημοσύνης. Η πλατφόρμα έχει την υποστήριξη της μεγαλύτερης κοινότητας σε ζητήματα ανοικτού κώδικα και μεγάλων δεδομένων. Το ApacheSpark είναι μια μηχανή επεξεργασίας για δεδομένα ανοικτού κώδικα μεγάλου συνόλου δεδομένων. Ο σχεδιασμός της προσφέρει μεγάλη υπολογιστική ταχύτητα, προγραμματισμό και επεκτασιμότητα, δυνατότητες που απαιτούνται για τα Big data και είναι ιδανική για εφαρμογές ροών δεδομένων, γραφημάτων, μηχανικής μάθησης και τεχνητής νοημοσύνης (AI) (Lee et al., 2016).

Η μηχανή ανάλυσης που υποστηρίζει το Spark προσφέρει επεξεργασία δεδομένων 10 ως 100 φορές πιο γρήγορα από τις εναλλακτικές πλατφόρμες. Η λειτουργία της έγκειται στον καταμερισμό εργασιών επεξεργασίας μεγάλων συμπλεγμάτων υπολογιστών έχοντας ενσωματωμένη την ανοχή σε σφάλματα. Εμπεριέχει ακόμη, και API (Application Programming Interface) για γλώσσες προγραμματισμού τα οποία είναι αρκετά δημοφιλή στον κόσμο των προγραμματιστών και γενικότερα της ανάλυσης δεδομένων περιλαμβάνοντας τις γλώσσες Scala, Python, Rκαι Java. Η πλατφόρμα ApacheSparkσυχνά συγκρίνεται με την πλατφόρμα ApacheHadoop και ειδικότερα με τηνMapReduceη οποία αποτελεί στοιχείο επεξεργασίας δεδομένων της Hadoop. Στις διαφορές του Sparkκαι του MapReduce είναι η πρώτη προβαίνει στην επεξεργασία και στην διατήρηση των δεδομένων στην μνήμη για τα επόμενα βήματα χωρίς εγγραφή ή ανάγνωση στον δίσκο. Το γεγονός αυτό οδηγεί

σε μεγαλύτερες ταχύτητες επεξεργασίας. Η πλατφόρμα ApacheSpark αναπτύχθηκε το 2009, από το Πανεπιστήμιο Berkeley. Μέχρι σήμερα, την πλατφόρμα την διατηρεί το ApacheFoundationSoftware καθώς διαθέτει την μεγαλύτερη κοινότητα σε θέματα ανοικτού κώδικα σε μεγάλα δεδομένα. Ωστόσο, αποτελεί βασικό στοιχείο εμπορικών εφαρμογών για μεγάλα δεδομένα.



Εικόνα 12..Apache SPARK (ibm.com)

3.2.3 Η Λειτουργία της Πλατφόρμας ApacheSpark

Η αρχιτεκτονική της Apache Spark, είναι ιεραρχική (master/slave). Ο κύριος κόμβος που έχει τον έλεγχο του διαχειριστή συμπλέγματος είναι ο Spark Driver, ο οποίος διαχειρίζεται και τους slave κόμβους παραδίδοντας τα αποτελέσματα των δεδομένων στον πελάτη της εφαρμογής.

Αν λάβει κανείς υπόψη του τον κώδικα της εφαρμογής, το Spark Driver προβαίνει στην δημιουργία του Spark content το οποίο συνεργάζεται με τον διαχειριστή του συμπλέγματος, που ονομάζεται ως Spark Stand alone Cluster Manager ή ακόμα και με άλλους διαχειριστές όπως το Hadoop, Kubernetes, Yan, Mesos κ.α τόσο για την διανομή όσο και για την παρακολούθηση της εκτέλεσης στους κόμβους. Εκτός αυτού,

δημιουργεί τα Resilient Distributed Datasets (RDDs) τα οποία είναι και το κλειδί για την ταχύτητα επεξεργασίας της Spark(Ope).

3.2.4 Resilient Distributed Dataset (RDD)

Τα RDDs είναι κατά βάση συλλογές από στοιχεία τα οποία έχουν ανοχή στα σφάλματα και έχουν τη δυνατότητα διανομής σε πολλούς κόμβους ενός συμπλέγματος για παράλληλη εργασία. Αποτελούν μια θεμελιώδης δομή στο ApacheSpark. Έτσι, το Spark, προβαίνει στην φόρτωση δεδομένων με κύρια αναφορά σε μια πηγή δεδομένων παραλληλίζοντας την ήδη υπάρχουσα συλλογή Spark Content σε ένα RDD για επεξεργασία. Με την φόρτωση των δεδομένων, το Spark προβαίνει σε ενέργειες του RDD στην μνήμη για την ταχύτητα του Spark. Το Spark, αποθηκεύει τα δεδομένα στην μνήμη εκτός αν υπάρχει εξάντληση της μνήμης του συστήματος ή διαγραφή των στοιχείων από τον χρήστη. Κάθε σύνολο δεδομένων χωρίζεται σε λογικά διαμερίσματα τα οποία υπολογίζονται σε διάφορους κόμβους του συμπλέγματος. Οι χρήστες του, έχουν την δυνατότητα εκτέλεσης δύο λειτουργιών, του μετασχηματισμού και των δράσεων. Όσο αφορά τους μετασχηματισμούς αφορούν σε λειτουργίες που εφαρμόζονται για να δημιουργηθεί ενός συνόλου RDD, ενώ οι ενέργειες χρησιμοποιούνται για εντολές που δίνονται στο Spark προκειμένου να εφαρμοστούν οι υπολογισμοί και η μεταβίβαση του αποτελέσματος ξανά στο πρόγραμμα οδήγησης (Lee et al., 2016).

3.2.5 Directed Acyclic Graph (DAG)

Σε αντιπαραβολή με την εκτέλεση των δύο σταδίων που παρουσιάστηκαν στην προηγούμενη παράγραφο αναφορικά με το MapReduce, το Spark δημιουργεί ένα DAG, για τον προγραμματισμό των εργασιών και την λειτουργία των κόμβων του συμπλέγματος. Κατά την εκτέλεση των ενεργειών και του μετασχηματισμού του Sparkκατά την διαδικασία εκτέλεσης ο DAG δίνει μια ευελιξία στην αποτελεσματικότητα οργανώνοντας τους κόμβους στο σύμπλεγμα. Η παρακολούθηση αυτών των εργασιών δίνει την δυνατότητα της παρακολούθησης της ανοχής σφαλμάτων

και της παρακολούθησης των εργασιών αφού εφαρμόζει ξανά τις καταγεγραμμένες λειτουργίες στα δεδομένα της πρότερης κατάστασης (Lee et al., 2016).

3.2.6 Data Frames-Datasets

Το Spark εκτός των άλλων διαχειρίζεται και άλλους τύπους δεδομένων όπως τα datasets και τα data frames. Πιο συγκεκριμένα τα data frames αποτελούν κοινές διεπαφές προγραμματισμού δομημένων εφαρμογών (API)αντιπροσωπεύοντας πίνακες δεδομένων σε γραμμές και στήλες. Εξαιτίας της δημοφιλίας που έχει η βιβλιοθήκη της Μηχανικής Μάθησης MLib, τα data frames διαδραματίζουν τον ρόλο του βασικού API. Πρακτικά αυτό σημαίνει ότι με την χρήση του API MLib τα data frames χαρακτηρίζονται από ομοιομορφία στις γλώσσες προγραμματισμού (Java, R, Scala, Python).

Στον αντίποδα τα datasets, αποτελούν επέκταση των data frames παρέχοντας μια ασφαλή διεπαφή προγραμματισμού. Τα σύνολα δεδομένων είναι μια συλλογή αντικειμένων σε αντίθεση με τα data frames. Το Spark SQL δίνει την δυνατότητα αναζήτησης των δεδομένων από τα data frames και τους χώρους αποθήκευσης δεδομένων SQL όπως το Apache Hive. Τα ερωτήματα του Spark SQL, επιστρέφουν ένα data frame ή ένα σύνολο δεδομένων όταν εκείνα εκτελούνται σε άλλη γλώσσα (Lee et al., 2016).

3.2.7Apache Spark MLib

Μία από τις κρίσιμες δυνατότητες του Apache Spark είναι οι δυνατότητες μηχανικής μάθησης που είναι διαθέσιμες στο Spark MLlib. Το Apache Spark MLlib παρέχει μια ολοκληρωμένη λύση για ταξινόμηση και παλινδρόμηση, συνεργατικό φιλτράρισμα, ομαδοποίηση, κατανεμημένη γραμμική άλγεβρα, δέντρα απόφασης, τυχαία δάση, δέντρα ενισχυμένα με κλίση, συχνή εξόρυξη προτύπων, μετρήσεις αξιολόγησης και στατιστικές. Οι δυνατότητες του MLlib, σε συνδυασμό με τους διάφορους τύπους δεδομένων που μπορεί να χειριστεί το Spark, κάνουν το Apache Spark ένα απαραίτητο εργαλείο Big Data (Lee et al., 2016).

3.2.8 Python –ApacheSpark

Η Python είναι από τις γλώσσες προγραμματισμού που μπορεί να χρησιμοποιηθεί με το ApacheSpark. Όλοι όσοι ασχολούνται με την επιστήμη των δεδομένων προτιμούν την πλατφόρμα του Spark, καθώς προσφέρει πολλά πλεονεκτήματα συγκριτικά με άλλα εργαλεία Big Data. Όντως, η γλώσσα Python είναι αρκετά δημοφιλής καθώς φαίνεται να είναι πολύ αποτελεσματική τόσο στην αξιολόγηση όσο και στην αντιμετώπιση μεγάλων συνόλων δεδομένων μηχανικής μάθησης. Με την χρήση της γλώσσας Python όλοι οι προγραμματιστές μπορούν να κάνουν πολλά περισσότερα καθώς περιλαμβάνει ευκολότερη διεπαφή. Επίσης, είναι πιο εύκολη στην σύνταξη και στην εκμάθηση καθώς είναι μια γλώσσα υψηλού επιπέδου. Παρέχει αναγνωσιμότητα και προτιμάται περισσότερο στην υλοποίηση αλγορίθμων μηχανικής μάθησης (Lee et al., 2016).

3.9 Επικύρωση πειραματικών μετρήσεων

Η επικύρωση των πειραματικών μετρήσεων έρχεται μέσα από την αξιολόγηση των ταξινομητών και περιλαμβάνει βαθμολόγηση με βάση:

- **τη συνολική ακρίβεια,**
- **τη μέση ανάκληση, την ακρίβεια και τη βαθμολογία F1 και τη μέση σταθμισμένη ακρίβεια, ανάκληση και βαθμολογία F1.**

Η ακρίβεια-precision (ονομάζεται επίσης θετική προγνωστική τιμή) είναι το κλάσμα των σχετικών περιπτώσεων μεταξύ των ανακτημένων περιπτώσεων Η ανάκληση-Recall (γνωστή και ως ευαισθησία) είναι το κλάσμα των σχετικών περιπτώσεων που ανακτήθηκαν. Η βαθμολογία F1 είναι μια μέτρηση αξιολόγησης μηχανικής μάθησης που μετρά την ακρίβεια ενός μοντέλου. Συνδυάζει τις βαθμολογίες ακρίβειας και ανάκλησης ενός μοντέλου. Η μέτρηση ακρίβειας υπολογίζει πόσες φορές ένα μοντέλο έκανε σωστή πρόβλεψη σε ολόκληρο το σύνολο δεδομένων. Το μέσο τετράγωνο σφάλμα (MSE) ή το μέσο τετράγωνο

απόκλιση (MSD) ενός εκτιμητή (μιας διαδικασίας για την εκτίμηση μιας μη παρατηρούμενης ποσότητας) μετρά τον μέσο όρο των τετραγώνων των σφαλμάτων δηλαδή η μέση τετραγωνική διαφορά μεταξύ των εκτιμώμενων τιμών και της πραγματικής τιμής. Το μέσο απόλυτο σφάλμα (MAE), είναι ένα μέτρο σφαλμάτων μεταξύ ζευγαρωμένων παρατηρήσεων που εκφράζουν το ίδιο φαινόμενο. Το RMSD ενός δείγματος είναι ο τετραγωνικός μέσος όρος των διαφορών μεταξύ των παρατηρούμενων και των προβλεπόμενων τιμών. Ο συντελεστής προσδιορισμού, που συμβολίζεται με R2 είναι η αναλογία της διακύμανσης στην εξαρτημένη μεταβλητή που είναι προβλέψιμη από τις ανεξάρτητες μεταβλητές.

Μέρος Β' Πειραματική Προσέγγιση

Κεφάλαιο 4ο Πειραματικές Μετρήσεις

Σε αυτό το κεφάλαιο παρουσιάζεται η περίπτωση μελέτης ενός συνόλου δεδομένων που είναι αποθηκευμένα σε αρχείο .csv και περιλαμβάνει την κίνηση των ροών των δεδομένων σε ένα SDN δίκτυο. Σκοπός του κεφαλαίου είναι να αναδειχτεί η προσέγγιση που θα εφαρμοστεί μέσα από την χρήση αλγορίθμων μηχανικής μάθησης ώστε να ανιχνευτεί η ταξινόμηση των δεδομένων σε τέτοιους τύπους δικτύων.

4.1 Περιγραφή των δεδομένων

Σε αυτήν την ενότητα παρουσιάζονται τα σύνολα των δεδομένων (datasets) που εμπεριέχουν όλη την κίνηση των δεδομένων ενός SDN δικτύου τα οποία θα χρησιμοποιηθούν εφαρμόζοντας τεχνικές μηχανικής μάθησης για την εξόρυξη των δεδομένων με στόχο την επιθυμητή γνώση.

Στις παρακάτω ενότητες περιγράφεται το διαθέσιμο αρχείο που θα χρησιμοποιηθεί για την εξόρυξη των δεδομένων καθώς και η γραμμογράφησή του. Το μέγεθος του αρχείου περιέχει 4.234 εγγραφές. Μπορεί να φαντάζει μικρό σε μέγεθος όμως οι τεχνικές που θα χρησιμοποιηθούν κάνοντας χρήση του ApacheSparkκαι της γλώσσας Python είναι εφάμιλλή των τεχνικών που χρησιμοποιούνται και σε μεγαλύτερα datasets.

Στον Πίνακα 1, που ακολουθεί παρουσιάζεται η γραμμογράφηση του αρχείου συνόλου δεδομένων SDN_traffic.csv.

Κατηγορία	**Περιγραφή**	**Τύπος Δεδομένων**
Id_flow	Αναγνωριστικό ροής	Χαρακτήρας
nw_src	Πηγή δικτύου(IP address)	Αριθμός
tp_src	Πηγή TCP (port)	Αριθμός
nw_dst	Προορισμός δικτύου	Αριθμός
tp_dst	Προορισμός TCP	Αριθμός
nw_proto	Πρωτόκολλο δικτύου	Αριθμός
forward_pc	Προώθηση διαδικασίας επικοινωνίας	Αριθμός
forward_bc	Προώθηση βασικής επικοινωνίας	Αριθμός
forward_pl	Προώθηση εμπρός φορτίου	Αριθμός
forward_piat	Προώθηση φυσικής πληροφόρησης	Αριθμός
forward_pps	Προώθηση πακέτων/δευτερόλεπτο	Αριθμός
forward_bps	Προώθηση των bits per second	Αριθμός
forward_pl_mean	Προώθηση μέσου όρου εμπρός φορτίου	Αριθμός
forward_piat_mean	Προώθηση μέσου όρου φυσικής πληροφόρησής	Αριθμός
forward_pps_mean	Προώθηση μέσου όρου πακέτων/δευτερόλεπτο	Αριθμός
forward_bps_mean	Προώθηση μέσου όρου των bits per second	Αριθμός
forward_pl_var	Προώθηση αξίας εμπρός φορτίου	Αριθμός
forward_piat_var	Προώθηση αξίας φυσικής πληροφόρησης	Αριθμός
forward_pps_var	Προώθηση αξίας πακέτων/δευτερόλεπτο	Αριθμός
forward_bps_var	Προώθηση αξίας τωνbits per second	Αριθμός
forward_pl_q1	Προώθηση εμπρός φορτίου-Ουρά1	Αριθμός
forward_pl_q3	Προώθηση εμπρός φορτίου-Ουρά3	Αριθμός
forward_piat_q1	Προώθηση φυσικής πληροφόρησης-Ουρά 1	Αριθμός
forward_piat_q3	Προώθηση φυσικής πληροφόρησης-Ουρά 3	Αριθμός

forward_pl_max	Προώθηση μέγιστου εμπρός φορτίου	Αριθμός
forward_pl_min	Προώθηση ελάχιστου εμπρός φορτίου	Αριθμός
forward_piat_max	Προώθηση μέγιστης φυσικής πληροφόρησης	Αριθμός
forward_piat_min	Προώθηση ελάχιστης φυσικής πληροφόρησης	Αριθμός
forward_pps_max	Προώθηση μέγιστων πακέτων/δευτερόλεπτο	Αριθμός
forward_pps_min	Προώθηση ελάχιστων πακέτων/δευτερόλεπτο	Αριθμός
forward_bps_max	Προώθηση μέγιστου bits per second	Αριθμός
forward_bps_min	Προώθηση ελάχιστου bits per seconds	Αριθμός
forward duration	Προώθηση διάρκειας	Αριθμός
forward_size_packets	Προώθηση μεγέθους πακέτων	Αριθμός
forward_size_bytes	Προώθηση μεγέθους bytes	Αριθμός
reverse_pc	Αντιστροφή διαδικασίας επικοινωνίας	Αριθμός
reverse_bc	Αντιστροφή βασικής επικοινωνίας	Αριθμός
reverse_pl	Αντιστροφή εμπρός φορτίου	Αριθμός
reverse_piat	Αντιστροφή φυσικής πληροφόρησης	Αριθμός
reverse_pps	Αντιστροφή πακέτων/δευτερόλεπτο	Αριθμός
reverse_bps	Αντιστροφή των bits per seconds	Αριθμός
reverse_pl_mean	Αντιστροφή μέσου όρου εμπρός φορτίου	Αριθμός
reverse_piat_mean	Αντιστροφή μέσου όρου φυσικής πληροφόρησης	Αριθμός
reverse_pps_mean	Αντιστροφή μέσου όρου πακέτων/δευτερόλεπτο	Αριθμός
reverse_bps_mean	Αντιστροφή μέσου όρου bites per second	Αριθμός
reverse_pl_var	Αντιστροφή αξίας εμπρός φορτίου	Αριθμός
reverse_piat_var	Αντιστροφή αξίας φυσικής πληροφόρησης	Αριθμός
reverse_pps_var	Αντιστροφή αξίας πακέτων/δευτερόλεπτο	Αριθμός
reverse_bps_var	Αντιστροφή της αξίας bits per second	Αριθμός
reverse_pl_q1	Αντιστροφή εμπρός φορτίου-Ουρά 1	Αριθμός
reverse_pl_q3	Αντιστροφή εμπρός φορτίου –Ουρά 3	Αριθμός
reverse_piat_q1	Αντιστροφή φυσικής πληροφόρησης-Ουρά 1	Αριθμός

reverse_piat_q3	Αντιστροφή φυσικής πληροφόρησης-Ουρά 3	Αριθμός
reverse_pl_max	Αντιστροφή μέγιστου εμπρός φορτίου	Αριθμός
reverse_pl_min	Αντιστροφή ελάχιστου εμπρός φορτίου	Αριθμός
reverse_piat_max	Αντιστροφή μέγιστης φυσικής πληροφόρησης	Αριθμός
reverse_piat_min	Αντιστροφή ελάχιστης φυσικής πληροφόρησης	Αριθμός
reverse_pps_max	Αντιστροφή μέγιστων πακέτων/δευτερόλεπτο	Αριθμός
reverse_pps_min	Αντιστροφή ελάχιστων πακέτων/δευτερόλεπτο	Αριθμός
reverse_bps_max	Αντιστροφή μέγιστου bits per second	Αριθμός
reverse_bps_min	Αντιστροφή ελάχιστου bits per second	Αριθμός
reverse_duration	Διάρκεια Αντιστροφής	Αριθμός
reverse_size_packets	Αντιστροφή μεγέθους πακέτων	Αριθμός
reverse_size_bytes	Αντιστροφή μεγέθους bytes	Αριθμός
Category	Κατηγορία	Χαρακτήρας

Εικόνα 13. Γραμμογράφηση Αρχείου SDN-Traffic

Ωστόσο, παρατηρείται συχνά στα μεγάλα σύνολα δεδομένων οι τιμές που ενέχουν τα χαρακτηριστικά τους (attributes) να είναι ελλιπής ή να μην αντιστοιχούν σε αναμενόμενες τιμές. Συνήθως αυτό οφείλεται στις διάφορες μορφές των δεδομένων όπου ενδεχομένως να προέρχονται από διαφορετικά συστήματα αφενός, αφετέρου τα δεδομένα που εμπεριέχουν να έχουν υποστεί κάποιας μορφής επεξεργασίας από ανθρώπινο χέρι. Το γεγονός αυτό προϋποθέτει των καθαρισμό των διαθέσιμων δεδομένων, διαδικασία που προηγείται της επεξεργασίας. Βέβαια, για να επιτευχθεί επιτυχής καθαρισμός των δεδομένων είναι αναγκαίο:

- Να αντικατασταθούν οι τιμές των χαρακτηριστικών με κάποια σταθερά για τα δεδομένα εκείνα τα οποία δεν διαθέτουν κάποια τιμή ακόμα και για εκείνα που διαθέτουν συγκεκριμένες τιμές,

- Παράλειψη των δεδομένων για τα οποία είτε υπάρχει κενή τιμή σε κάποιο χαρακτηριστικό ή δεν έχει σημασία περαιτέρω ερμήνευσης,
- Εύρεση του μέσου όρου των χαρακτηριστικών και στην συνέχεια αντικατάσταση των τιμών που λείπουν με εκείνη του μέσου όρου,
- Εφαρμογή αλγορίθμων κατηγοριοποίησης για την ανίχνευση των λανθασμένων τιμών ώστε να αφαιρεθούν από το σύνολο των δεδομένων.

4.3 Παρουσίαση Προβλέψεων Με Τεχνικές Εξόρυξης Δεδομένων

4.3.1 Αλγόριθμος Naïve Bayes

Σε αυτή την ενότητα παρουσιάζεται η υλοποίηση του αλγορίθμου κατηγοριοποίησης Naïve Bayes στο αρχείο SDN_traffic.csv προκειμένου να αναλυθεί ώστε να εξαχθεί γνώση και να απεικονιστεί η όποια πρόβλεψη κατηγοριοποίησης του συνόλου δεδομένων που εξετάζεται αρχικά με τις έξι κλάσεις και στην συνέχεια με δύο κλάσεις.

Στην εικόνα 4.3.1, περιγράφεται η πρόβλεψη που δίνει ο Naïve Bayes με την προσέγγιση των έξι κλάσεων. Ενδεικτικά για τις πέντε πρώτες γραμμές του συνόλου δεδομένων οι προβλέψεις παίρνουν τις τιμές [0.0….3.0]

features	label	rawPrediction	probability	prediction
(262144,[493,7698...	4.0	[-1848.7753135227...	[8.30192455940876...	3.0
(262144,[4087,526...	1.0	[-2241.8777629162...	[1.0,2.5406825645...	0.0
(262144,[6524,715...	1.0	[-2162.3996921069...	[1.0,1.7383716917...	0.0
(262144,[2292,625...	1.0	[-1394.5642996798...	[1.0,4.8687893284...	0.0
(262144,[4758,825...	1.0	[-1077.5558480647...	[1.0,4.2323114490...	0.0
(262144,[12524,43...	6.0	[-244.37367107730...	[0.00161151642308...	5.0
(262144,[30483,76...	4.0	[-1950.4976360053...	[5.83932218669548...	3.0
(262144,[37135,57...	4.0	[-2259.4950407170...	[2.0E-323,0.0,0.0...	3.0
(262144,[41070,54...	5.0	[-4297.0336609326...	[6.36510439578388...	4.0
(262144,[493,2374...	4.0	[-2014.5785339574...	[4.07872394328699...	3.0
(262144,[4731,110...	3.0	[-2357.0603601263...	[5.05941446164458...	2.0
(262144,[12034,12...	1.0	[-3048.7596950131...	[1.0,4.3665343202...	0.0
(262144,[12524,19...	4.0	[-2823.9933118440...	[6.44547657452727...	3.0

Εικόνα 4.3.1Naive Bayes 6-classes

Στην Εικόνα 4.3.2 περιγράφονται οι τιμές του Naïve Bayes σχετικά με το accuracy, το precision, το recall και το F1. Οι τιμές για έξι κλάσεις κυμαίνονται από [0.015……0.027]

```
Naive Bayes Metrics
Accuracy for Naive Bayes = 0.015267175572519083
Precision for Naive Bayes = 0.23811862329646963
Recall for Naive Bayes = 0.015267175572519083
F1 for Naive Bayes =  0.027650704594137168
```

Εικόνα 4.3.2 Naive Bayes metrics

Στην Εικόνα 4.3.3 , παρουσιάζονται οι προβλέψεις του Naïve Bayes για την προσέγγιση με δύο κλάσεις. Οι τιμές των προβλέψεων κυμαίνονται από [0.00…..1.00]

Print Naive Bays dataframe for 2 Classes

features	label	rawPrediction	probability	prediction
(262144,[493,7698...	0.0	[-1374.8627046137...	[1.0,1.5526782549...	0.0
(262144,[4087,526...	1.0	[-2490.4259327255...	[1.11411299379054...	1.0
(262144,[6524,715...	1.0	[-2566.7024297428...	[2.53069938146481...	1.0
(262144,[2292,625...	1.0	[-1836.4841177041...	[1.17362636186518...	1.0
(262144,[4758,825...	1.0	[-1567.8466595762...	[1.15279000143891...	1.0
(262144,[12524,43...	0.0	[-234.79263886494...	[0.99993173425618...	0.0
(262144,[30483,76...	0.0	[-1495.4803931858...	[1.0,2.5021798410...	0.0
(262144,[37135,57...	0.0	[-1581.7014419048...	[1.0,4.2847145573...	0.0
(262144,[41070,54...	0.0	[-4114.5491672264...	[1.0,5.8150740958...	0.0
(262144,[493,2374...	0.0	[-1548.7627381752...	[1.0,4.9221937135...	0.0
(262144,[4731,110...	0.0	[-2231.8962446084...	[1.0,4.4324728894...	0.0
(262144,[12034,12...	1.0	[-3145.4702179322...	[9.70567248057976...	1.0
(262144,[12524,19...	0.0	[-2356.1891147714...	[1.0,7.0037460703...	0.0

Εικόνα 4.3.3 Naive Bayes predictions 2-classes

Στην Εικόνα 4.3.4 περιγράφονται οι μετρήσεις ου Naïve Bayes για το accuracy, recall, precision, f1 με τις τιμές να κυμαίνονται από [0.919…..0.918]

```
Naive Bayes classifier Metrics for 2 Classes (WWW or non-WWW)
Accuracy for Naive Bayes classifier = 0.9190839694656489
Precision for Naive Bayes classifier = 0.9189765407140271
Recall for Naive Bayes classifier = 0.919083969465649
F1 for Naive Bayes classifier =  0.9188882364454883
```

Εικόνα 4.3.4 Naive Bayes metircs-2 classes

4.3.2 Αλγόριθμος Logistic Regression

Σε αυτήν την ενότητα παρουσιάζεται η υλοποίηση του αλγορίθμου κατηγοριοποίησης Logistic Regression στο αρχείο SDN_traffic.csv προκειμένου να αναλυθεί ώστε να εξαχθεί γνώση και να απεικονιστεί η όποια πρόβλεψη κατηγοριοποίησης του συνόλου δεδομένων που εξετάζεται αρχικά με τις έξι κλάσεις και στην συνέχεια με δύο κλάσεις.

Στην Εικόνα 4.3.5 περιγράφονται οι προβλέψεις του αλγορίθμου Logistic Regression με την προσέγγιση των έξι κλάσεων καταγράφοντας τιμές από [1.0…4.0]

```
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  4.0|[-9.3078630500768...|[2.05540575815717...|       4.0|
|(262144,[4087,526...|  1.0|[-9.3078536508630...|[3.23334643470935...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078429662581...|[2.11369810870502...|       1.0|
|(262144,[2292,625...|  1.0|[-9.3078567423627...|[1.22204133141243...|       1.0|
|(262144,[4758,825...|  1.0|[-9.3078565356683...|[1.78461855870915...|       1.0|
|(262144,[12524,43...|  6.0|[-9.3078780508099...|[1.15859619395442...|       6.0|
|(262144,[30483,76...|  4.0|[-9.3078682182321...|[4.43757079568026...|       4.0|
|(262144,[37135,57...|  4.0|[-9.3078589155435...|[2.69312373335182...|       4.0|
|(262144,[41070,54...|  5.0|[-9.3078105381218...|[1.84711552285391...|       6.0|
|(262144,[493,2374...|  4.0|[-9.3078618568025...|[2.91330387376916...|       4.0|
|(262144,[4731,110...|  3.0|[-9.3078436392892...|[2.18709366216611...|       6.0|
|(262144,[12034,12...|  1.0|[-9.3078470847790...|[1.37325607739731...|       1.0|
|(262144,[12524,19...|  4.0|[-9.3078547484739...|[9.69904610692073...|       4.0|
|(262144,[61509,76...|  1.0|[-9.3078532055441...|[3.25231619621963...|       1.0|
|(262144,[3616,263...|  1.0|[-9.3078400023085...|[6.29494264461110...|       1.0|
|(262144,[3774,610...|  1.0|[-9.3078578680097...|[1.23672243531380...|       1.0|
|(262144,[4758,825...|  1.0|[-9.3078565356683...|[1.78461855870915...|       1.0|
|(262144,[5710,125...|  1.0|[-9.3078533092818...|[3.70536619074687...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078572266494...|[1.18658204579589...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078575698008...|[1.17557070855986...|       1.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 20 rows
```

Εικόνα 4.3.5 .Logistic Regression predictions- 6 classes

Στην Εικόνα 4.3.6 περιγράφονται οι μετρήσεις του αλγόριθμου Logistic Regression για το accuracy, recall, precision, F1 με τιμές από [0.79…….0.88]

```
Logistic Regression Metrics
Accuracy for Logistic Regression = 0.8412213740458016
Precision for Logistic Regression = 0.8082027197321239
Recall for Logistic Regression = 0.8412213740458016
F1 for Logistic Regression =  0.7957827232475606
```

Εικόνα 4.3.6 .Logistic Regressionmetrics-6 classes

Στην Εικόνα 4.3.7 περιγράφονται οι προβλέψεις του αλγόριθμου Logistic Regression με την προσέγγιση των δύο κλάσεων με τιμές [0.0….1.0].

Print Logistic Regression dataframe

features	label	rawPrediction	probability	prediction
(262144,[493,7698...	0.0	[2.97348583548864...	[0.95136182933893...	0.0
(262144,[4087,526...	1.0	[-2.4461484094555...	[0.07972066518319...	1.0
(262144,[6524,715...	1.0	[-3.0725219956060...	[0.04425503324684...	1.0
(262144,[2292,625...	1.0	[-3.8936873768245...	[0.01996343803458...	1.0
(262144,[4758,825...	1.0	[-3.4085075786648...	[0.03203063722598...	1.0
(262144,[12524,43...	0.0	[1.53532047281826...	[0.82278342967203...	0.0
(262144,[30483,76...	0.0	[2.81068367880512...	[0.94325042696653...	0.0
(262144,[37135,57...	0.0	[3.38434312232355...	[0.96721161978953...	0.0
(262144,[41070,54...	0.0	[1.54295157913317...	[0.82389338839687...	0.0
(262144,[493,2374...	0.0	[3.11770286032205...	[0.95761709302308...	0.0
(262144,[4731,110...	0.0	[1.65159317737120...	[0.83910625665687...	0.0
(262144,[12034,12...	1.0	[0.07716894364691...	[0.51928266776375...	0.0
(262144,[12524,19...	0.0	[2.83590639927652...	[0.94458557771491...	0.0
(262144,[61509,76...	1.0	[-2.4924909249386...	[0.07638627354122...	1.0

Εικόνα 4.3.7. Logistic Regression predictions with 2 classes

Στην Εικόνα 4.3.8 περιγράφονται οι μετρήσεις του αλγόριθμου Logistic Regression για το accuracy, recall, precision, F1 με τιμές από [0.920…….0.921]

```
Logistic Regression Metrics
Accuracy for Logistic Regression = 0.9206106870229007
Precision for Logistic Regression = 0.9252016816768205
Recall for Logistic Regression = 0.9206106870229007
F1 for Logistic Regression =  0.9210916048110697
```

Εικόνα 4.3.8 . Logistic Regression metrics with 2 classes

4.3.3Αλγόριθμος Linear Regression

Σε αυτήν την παράγραφο περιγράφεται η υλοποίηση του αλγορίθμου κατηγοριοποίησης Linear Regression του data set για να αναλυθεί ώστε να εξαχθεί γνώση και να απεικονιστεί η όποια πρόβλεψη κατηγοριοποίησης του συνόλου δεδομένων που εξετάζεται αρχικά με τις έξι κλάσεις και στην συνέχεια με δύο κλάσεις.

Στην Εικόνα 4.3.9 περιγράφονται οι προβλέψεις του αλγορίθμου Linear Regression με την προσέγγιση των έξι κλάσεων καταγράφοντας τιμές από [0.95…3.8]

features	label	prediction
(262144,[493,7698...	4.0	3.864791626125427
(262144,[4087,526...	1.0	1.9184876920447194
(262144,[6524,715...	1.0	1.5803268240273916
(262144,[2292,625...	1.0	0.954790074722017
(262144,[4758,825...	1.0	1.0063165801686988
(262144,[12524,43...	6.0	5.090234803746513
(262144,[30483,76...	4.0	3.9913273390483615
(262144,[37135,57...	4.0	3.9871401619147653
(262144,[41070,54...	5.0	4.072119469897814
(262144,[493,2374...	4.0	3.9393887732096964
(262144,[4731,110...	3.0	4.058254222111514
(262144,[12034,12...	1.0	3.07146167963999
(262144,[12524,19...	4.0	3.497616335245621
(262144,[61509,76...	1.0	1.4363418623737396
(262144,[3616,263...	1.0	2.0707440358521474
(262144,[3774,610...	1.0	1.0075947891257706

Εικόνα 4.3.9. .Linear Regression predictions-6 classes

Στην Εικόνα 4.3.10 περιγράφονται οι μετρήσεις του αλγορίθμου Linear Regression με την προσέγγιση των έξι κλάσεων καταγράφοντας τιμές των Mean square error, mean absolute error, RMSE, R2 από [0.10…..0.94]

```
Linear Regression Metrics
meanSquaredError: 0.198257
meanAbsoluteError: 0.105969
RMSE: 0.445260
r2: 0.949836
```

Εικόνα 4.3.10 .Linear Regression Metrics-6 classes

Στην Εικόνα 4.3.11 περιγράφονται οι προβλέψεις του αλγορίθμου Linear Regression με την προσέγγιση των έξι κλάσεων καταγράφοντας τιμές από [0.95…5.0]

```
Print Linear Regression dataframe
+--------------------+-----+------------------+
|            features|label|        prediction|
+--------------------+-----+------------------+
|(262144,[493,7698...|  0.0|3.8647916255932286|
|(262144,[4087,526...|  1.0|1.9184876924089738|
|(262144,[6524,715...|  1.0|1.5803268243875714|
|(262144,[2292,625...|  1.0|0.9547900746924358|
|(262144,[4758,825...|  1.0|1.0063165801371166|
|(262144,[12524,43...|  0.0| 5.090234804888531|
|(262144,[30483,76...|  0.0|3.9913273390335555|
|(262144,[37135,57...|  0.0|3.9871401619532456|
|(262144,[41070,54...|  0.0| 4.072119469645542|
|(262144,[493,2374...|  0.0| 3.939388773086295|
|(262144,[4731,110...|  0.0| 4.058254222130187|
|(262144,[12034,12...|  1.0|3.0714616799580083|
|(262144,[12524,19...|  0.0|3.4976163356807635|
|(262144,[61509,76...|  1.0|1.4363418625840487|
|(262144,[3616,263...|  1.0| 2.070744035406924|
|(262144,[3774,610...|  1.0|1.0075947894206165|
```

Εικόνα 4.3.11.Linear Regression predictions-2 classes

Στην Εικόνα 4.3.12 περιγράφονται οι μετρήσεις του αλγορίθμου Linear Regression με την προσέγγιση των δύο κλάσεων καταγράφοντας τιμές των Mean square error, mean absolute error, RMSE, R2 από [0.10…..0.94]

```
Linear Regression Metrics
meanSquaredError: 0.198257
meanAbsoluteError: 0.105969
RMSE: 0.445260
r2: 0.949836
```

Εικόνα 4.312.Linear Regression metrics 2- classes

4.5Αποτελέσματα

Στην παρούσα εργασία χρησιμοποιήθηκε το dataset όπου μελετήθηκε η ροή κίνησης των δεδομένων του. Στο πειραματικό μέρος της εργασίας έγινε μια προσπάθεια

υλοποίησης ενός προβλεπτικού μοντέλου με την χρήση των αλγορίθμων Naïves Bayes, Linear Regressionκαι Logistic Regression. Χρησιμοποιήθηκαν οι έξι κλάσεις WWW, P2P, DNS, VOIP, FTP,ICMP και στην συνέχεια οι δύο κλάσεις WWW or non WWW. Και στις τρεις υλοποιήσεις παρατηρήθηκε ότι οι δύο κλάσεις έδωσαν μεγαλύτερη ακρίβεια στις προβλέψεις.

Στον παρακάτω πίνακα παρουσιάζονται αναλυτικά οι προβλέψεις και για τις τρεις υλοποιήσεις:

Πίνακας 1. Προσέγγιση αλγορίθμων με έξι κλάσεις

Naïve Bayes		**Logistic Regression**		**Linear Regression**	
Accuracy	0,015	**Accuracy**	0,84	**Mean squared error**	0,19
Precision	0,23	**Precision**	0,80	**Mean absolute error**	0,10
Recall	0,015	**Recall**	0,84	**RMSE**	0,44
F1	0,027	**F1**	0,79	**R2**	0,94

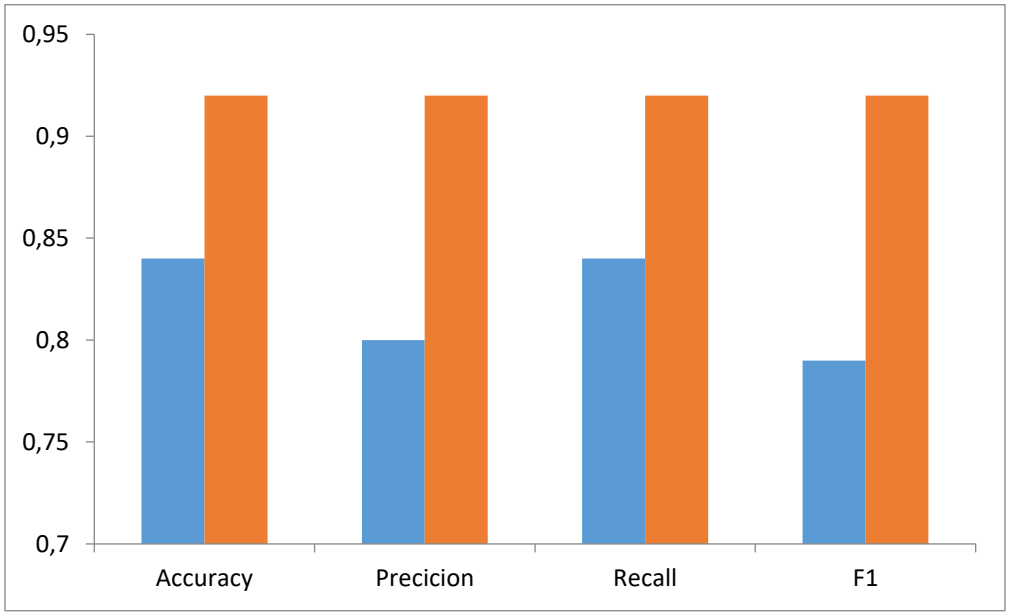
Πίνακας 2. Προσέγγιση αλγορίθμων με δύο κλάσεις

Naïve Bayes		**Logistic Regression**		**Linear Regression**	
Accuracy	0,91	**Accuracy**	0,92	**Mean squared**	0,19

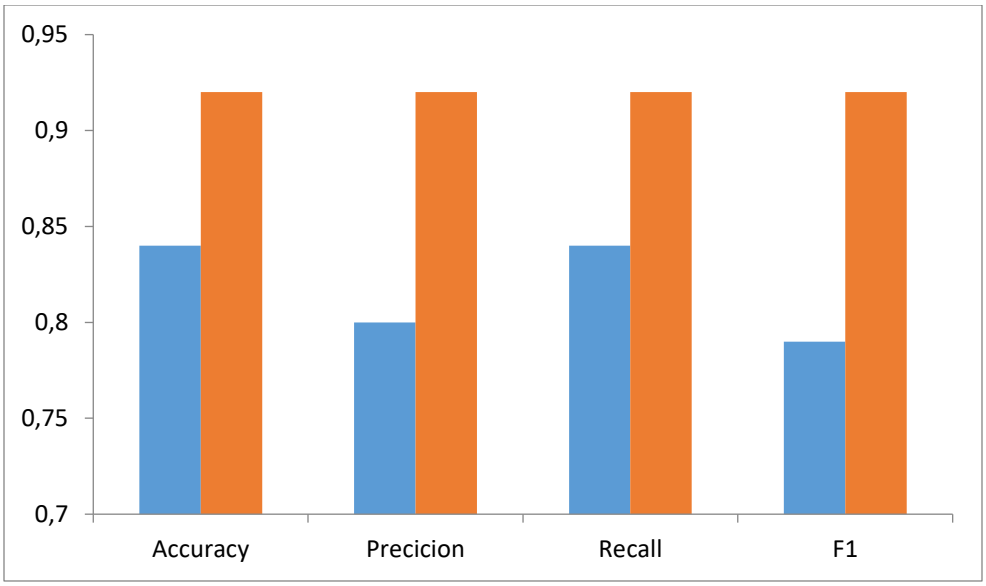
				error	
Precision	0,91	**Precision**	0,92	**Mean absolute error**	0,10
Recall	0,91	**Recall**	0,92	**RMSE**	0,44
F1	0,91	**F1**	0,92	**R2**	0,94

Στα παρακάτω γραφήματα παρουσιάζονται διαγραμματικά η σύγκριση των προσεγγίσεων των έξι και δύο κλάσεων με τον Naïve Bayes Classifier, Logistic και Linear Regression τεχνικών-μετρήσεων.

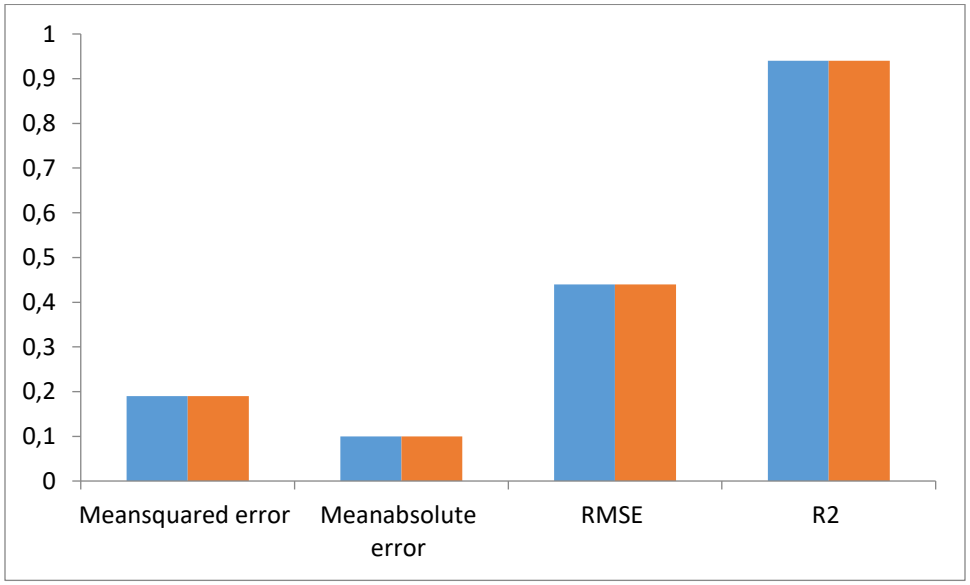
Γράφημα 1. Σύγκριση της Προσέγγισης των έξι/δύο κλάσεων (Naïve Bayes Classifier)



Γράφημα 2. Σύγκριση της Προσέγγισης των έξι/δύο κλάσεων (Logistic Regression)



Γράφημα 3. Σύγκριση της Προσέγγισης των έξι/δύο κλάσεων (Linear Regression)



Linear Regression with six/two classes

5. Συμπεράσματα

Η μεγάλη ανάγκη για εξόρυξη δεδομένων οδηγεί σε μεγάλες αλλαγές και στην υιοθέτηση νέων τεχνολογιών με σκοπό την συγκέντρωση μεγάλου αριθμού δεδομένων. Βασική προτεραιότητα είναι η πληροφορία που αναμένεται να εξαχθεί και τα μοντέλα που θα προκύψουν που θα έχουν μια δομή που θα είναι πλήρως κατανοητή και αντιληπτή για την λήψη αποφάσεων. Ο αναλυτής δεδομένων έχει μεγάλη ευθύνη στο να εντοπίσει τα λάθη που προκύπτουν από τα μεγάλα δεδομένα.

Με το παρόν εγχείρημα έγινε μια προσπάθεια ανάπτυξης ενός μοντέλου προβλέψεων μέσα από την πολύτιμη συμβολή των τεχνικών μηχανικής μάθησης ώστε να εξαχθούν οι προβλέψεις που απαιτούνταν σε ένα σύνολο δεδομένων εκτιμώντας τις ροές ενός δικτύου SDN. Το γεγονός αυτό οδήγησε στην χρήση των αλγορίθμων μηχανικής μάθησης στο συγκεκριμένο dataset ώστε να εξαχθούν με ακρίβεια οι πληροφορίες που απαιτούνταν. Πιο συγκεκριμένα, υλοποιήθηκε μια προσέγγιση έξι και δύο κλάσεων με κύρια στήλη για την ταξινόμηση την στήλη του Category ώστε να δημιουργηθεί ένα σύνολο προβλέψεων με την συνδρομή των αλγορίθμων μηχανικής μάθησης Naïve Bayes, Linear Regression και Logistic Regression. Οι προβλέψεις του Naïve Bayes επικεντρώθηκαν στην μέτρηση του accuracy, του precision, του recall και του F1. Στην προσέγγιση των έξι κλάσεων οι τιμές του για το dataset ήταν για το accuracy 0,015, για το precision 0,23, για το recall 0,015 και για το F1 0,027. Οι αντίστοιχες τιμές του Naïve Bayes για την προσέγγιση των δύο κλάσεων ήταν 0,91 και για τις τέσσερις μετρήσεις. Αντίστοιχα, μέσα από την υλοποίηση του αλγορίθμου Logistic Regression ήταν για το accuracy 0,84, για το precision 0,80, για το recall 0,84 και για το F1 0,79. Στην υλοποίηση των δύο κλάσεων οι αντίστοιχες μετρήσεις διαμορφώθηκαν στο 0,92. Τέλος, ο τρίτος αλγόριθμος μηχανικής μάθησης με την προσέγγιση των έξι κλάσεων έδωσε για την μέτρηση των Meansquared error 0,19 Meanabsolute error 0,10 RMSE 0,44, R2

0,94. Οι ίδιες τιμές διαμορφώθηκαν και για την προσέγγιση με δύο κλάσεις. Συνεπώς, από την υλοποίηση διαπιστώνεται ότι η προσέγγιση των δύο κλάσεων δίνει καλύτερες προβλέψεις και για τους τρεις αλγορίθμους που χρησιμοποιήθηκαν.

6.Προτάσεις για μελλοντική Έρευνα

Στην παρούσα έρευνα παρουσιάστηκαν μέθοδοι ταξινόμησης για ένα σύνολο δεδομένων SDN δικτύων. Οι μέθοδοι που χρησιμοποιήθηκαν έχουν ευρεία χρήση και πολλούς τομείς εφαρμογής. Ως μελλοντική έρευνα, στην εξόρυξη δεδομένων και στις τεχνικές εκπαίδευσης και ταξινόμησης συνόλου δεδομένων θα μπορούσε να παρουσιαστεί ένα λογισμικό που εφαρμόζει ταξινομήσεις δεδομένων και θα βασίζεται στην τεχνική grammatical evolution όπου οι μελλοντικές εκδόσεις του λογισμικού θα περιλαμβάνουν μια σειρά βελτιώσεων όπως εισαγωγή από διάφορες μορφές (CSV, Json κ.λπ.), χρήση κανόνων διακοπής βελτίωσης για τον γενετικό αλγόριθμο ή ακόμη και πρόσθετες μορφές εξόδου.

Βιβλιογραφία

Akyildiz, I.F., Lee, A., Wang, P., Luo, M., Chou, W., (2014). A roadmap for traffic engineering in SDN-Open Flow networks. Comput. Network. 71, 1–30.

Akyildiz, I.F., Lee, A., Wang, P., Luo, M., Chou, W., Jun. (2016). Research challenges for Traffic engineering in software defined networks. IEEE Network 30 (3), 52–58.

Ahmad, I., Namal, S., Ylianttila, M, andGurtov, A. (2015). Security in software defined networks: A survey," IEEE Communications Surveys & Tutorials, vol. 17, pp. 2317-2346

Bharati M. Ramageri. (2010). Data Mining Techniques and Applications Indian Journal of Computer Science and Engineering, Vol. 1 No. 4 301-305,Modern Institute of Information Technology and Research, Department of Computer Application, Yamunanagar, Nigdi Pune, Maharashtra, India-411044

Buczak, A.L., Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. IEEE Commun. Surv. Tutorials 2016, 18, 1153–1176.

Farhady, H., Lee, H., Nakao, A., Apr. (2015). Software-defined networking: a survey. In: Computer Networks, vol. 81. Elsevier B.V., pp. 79–95

Hsieh, C., Weng,N, Wei.., W. (2018). Scalable Many-Field Packet Classification for Traffic Steering in SDN Switches, IEEE Transactions on Network and Service Management, vol. 16, pp. 348-361

Jamuna, A., Ewards. V. (2013). Survey of Traffic Classification using Machine Learning. International journal of advanced research in computer science 4.4

Kruczkowski, M., Niewiadomska-Szynkiewicz, E.; Kozakiewicz, A. (2015). FP-tree and SVM for Malicious Web Campaign Detection. In Intelligent Information and Database Systems. ACIIDS 2015; Nguyen, N., Trawiński, B., Kosala, R., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switherland, 2015; Volume 9012, pp. 193–201.

Kruczkowski, M., Niewiadomska-Szynkiewicz, E. (2014). Comparative study of supervised learning methods for malware analysis. *J. Telecommun. Inf. Technol.* 2014, *4*, 24–33.

Lee, S., Yoon, C., Shin, S. (2016). The Smaller, the Shrewder. In Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization; ACM: New York, NY, USA, 2016; pp. 23–28.

Lee, D., Damji, J. (2016). Apache Spark Key Terms, Explained. Available at: https://www.databricks.com/blog/2016/06/22/apache-spark-key-terms-explained.html [Access at 26/11/2023]

Mendiola, A, Astorga, J, Jacob, E, and Higuero, M. (2016). A survey on the contributions of software-defined networking to traffic engineering, IEEE Communications Surveys & Tutorials, vol. 19, pp. 918-953, 2016.

Mestres, A. Rodriguez-Natal, J. Carner, P. BarletRos, E. Alarcón, M. Solé, et al., (2017). Knowledge-defined networking, ACM SIGCOMM Computer Communication Review, vol. 47, pp. 2-10

Mulla, M. M., Raikar, M. M., Meghana, M. K., Shetti, N. S., & Madhu, R. K. (2019). Load Balancing for Software-Defined Networks. In Emerging Research in Electronics, Computer Science and Technology (pp. 235-244). Springer, Singapore.

Ng, B., M. Hayes, and W. K. G. Seah. (2015). developing a traffic classification platform for enterprise networks with SDN: Experiences lessons learned. IFIP NETWORKING Conference IEEE, 2015.

P. C. da Rocha Fonseca, Mota, S. (2017). A survey on fault management in software-defined networks," IEEE Communications Surveys & Tutorials, vol. 19, pp. 2284-2321.

Raikar, M.M., Meena, S.M., Mulla, M.M., Shetti, N.S. and Karanandi, M., 2020. Data traffic classification in software defined networks (SDN) using supervised-learning. *Procedia Computer Science*, *171*, pp.2750-2759.

Sung, Y., Sharma, P., Lopez, E., Park, J., (2016). FS-Open Security: a taxonomic modeling of security threats in SDN for future sustainable computing," Sustainability, vol. 8, p. 919

Queiroz, W., Capretz, M., Dantas, M., (2019). An approach for SDN traffic monitoring based on big data techniquesa Department of Electrical and Computer Engineering, Western University, London ON N6A 5B9 Canada, Department of Computer Science, Federal University of Juiz de Fora, Juiz de Fora MG 36036-330 Brazil

Selvi, T., Thamiselvan, R., (2020). Deep Learning Based Traffic Classification In Software Defined Networking –A Survey international journal of scientific & technology research volume 9, issue 02, February 2020

Tan. Steinbach, Karpatne, Kumar. (2020). Introduction to Data Mining (2nd Edition)- Lecture Notes for Chapter 1

Tang, T. A.., Mhamdi, L., D. McLernon, Zaidi,S. A. R., Ghogho,M. (2018). Deep recurrent neural network for intrusion detection in sdn-based networks, 4th IEEE Conference on Network Softwarization and Workshops (NetSoft), 2018, pp. 202-206

Tsoulos., I, Anastasopoulos., N, Tzallas., A. (2021). GenClass: A parallel tool for data classification based on Grammatical Evolution, SoftwareX 16 (2021) 100830,

Volpato, F., Castro, M.B., Dantas, M.A.R., (2019). OFQuality: quality of service Management module for software-defined networking. Int. J. Grid Comput. Util. IJGUC 10 (2), 187–198.

Wickboldt, J, W. P. De Jesus, Isolani, P. H.., Both, C. B., Rochol, J., Granville, L. Z. (2015).Software-defined networking: management requirements and challenges, IEEE Communications Magazine, vol. 53, pp. 278-285.

Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, et al., (2018). A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges," IEEE Communications Surveys & Tutorials, vol. 21, pp. 393-430, 2018

Xie. J. et al., (2019). A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges, in IEEE Communications Surveys & Tutorials, vol. 21, no. 1, pp. 393-430, First quarter

Yan, Q., Yu, F., Gong,Q,Li,J. (2015). Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," IEEE Communications Surveys & Tutorials, vol. 18, pp. 602-622.

Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks," IEEE Access, vol. 5, pp. 21954-21961

Yu, Changhe, et al. "QoS-aware Traffic Classification Architecture Using Machine Learning and Deep Packet Inspection in SDNs." Procedia computer science 131 (2018): 1209-1216.

Ηλεκτρονικές Πηγές

https://www.kdnuggets.com/2016/04/association-rules-apriori-algorithm-tutorial.html

https://www.geeksforgeeks.org/clustering-in-machine-learning/

https://www.javatpoint.com/regression-vs-classification-in-machine-learning

https://www.wired.com/2016/03/took-neuroscientists-ten-years-map-tiny-slice-brain/

Παράρτημα Ι: Υλοποίηση Μοντέλου Πρόβλεψης Ροών Δικτύου SDN

Install Apache Spark, Java, NLP

```
[1]  # # innstall java
     # !apt-get install openjdk-8-jdk-headless -qq > /dev/null

     # # # install spark (change the version number if needed)
     # !wget -q https://archive.apache.org/dist/spark/spark-3.0.0/spark-3.0.0-bin-hadoop3.2.tgz

     # # # unzip the spark file to the current folder
     # !tar xf spark-3.0.0-bin-hadoop3.2.tgz

     # # # set your spark folder to your system path environment.
     # import os
     # os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
     # os.environ["SPARK_HOME"] = "/content/spark-3.0.0-bin-hadoop3.2"


     # # #install findspark using pip
     # !pip install -q findspark
```

```
[ ]  # Check if installed properly
     import findspark
     findspark.init()
     from pyspark.sql import SparkSession
     spark = SparkSession.builder.master("local[*]").getOrCreate()
```

```
[ ]  pip install spark-nlp==2.4.2
```

```
Collecting spark-nlp==2.4.2
  Downloading spark_nlp-2.4.2-py2.py3-none-any.whl (108 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 108.4/108.4 kB 2.9 MB/s eta 0:00:00
Installing collected packages: spark-nlp
Successfully installed spark-nlp-2.4.2
```

```
[ ]  import sparknlp

     spark = sparknlp.start()

     print("Spark NLP version: ", sparknlp.version())
     print("Apache Spark version: ", spark.version)
```

```
Spark NLP version:  2.4.2
Apache Spark version:  3.0.0
```

```
[ ]  # Import dataset
     import pandas as pd

     # dataset: SDN_traffic.csv
     df = pd.read_csv('./SDN_traffic.csv')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4234 entries, 0 to 4233
Data columns (total 66 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id_flow               4234 non-null   object
 1   nw_src                4234 non-null   object
 2   tp_src                4234 non-null   int64
 3   nw_dst                4234 non-null   object
 4   tp_dst                4234 non-null   int64
 5   nw_proto              4234 non-null   int64
 6   forward_pc            4234 non-null   int64
 7   forward_bc            4234 non-null   int64
 8   forward_pl            4234 non-null   float64
 9   forward_piat          4234 non-null   float64
 10  forward_pps           4234 non-null   float64
 11  forward_bps           4234 non-null   float64
 12  forward_pl_mean       4234 non-null   float64
 13  forward_piat_mean     4234 non-null   float64
 14  forward_pps_mean      4234 non-null   float64
 15  forward_bps_mean      4234 non-null   float64
 16  forward_pl_var        4234 non-null   float64
 17  forward_piat_var      4234 non-null   float64
 18  forward_pps_var       4234 non-null   float64
 19  forward_bps_var       4234 non-null   object
 20  forward_pl_q1         4234 non-null   object
 21  forward_pl_q3         4234 non-null   float64
```

```
[ ] df.head()
```

	id_flow	nw_src	tp_src	nw_dst	tp_dst	nw_proto	forward_pc	forward_bc	forward_p
0	b2bb77a570fcfa9325eb9e51b6116d2a	172.16.25.104	41402	34.107.221.82	80	6	5	300	60.0
1	f07977b0d1d6645c4fe1e9efea080ff3	172.16.25.104	41406	34.107.221.82	80	6	5	300	60.0
2	e4026ba9b6c1957516e92bdd0d04878f	172.16.25.104	38232	52.84.77.43	443	6	3	198	66.0
3	e2d747932e41500b1463fe8ae4299ecb	172.16.25.104	38234	52.84.77.43	443	6	3	198	66.0
4	56325703391225ad65e013e7a2b02fac	172.16.25.104	60166	52.32.34.32	443	6	4	265	66.2

5 rows × 66 columns

```
[ ]  # find category unique value
     df['category'].unique()
```

```
array(['WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P', '9643368', '12395988',
       '5244492', '5670522', '4875234', '7757508', '10810338', '12119844',
       '10833306', '11629662', '11207586', '7436682', '11731500',
       '5989962', '8869554', '5645982', '12876270'], dtype=object)
```

```
[ ]  df.columns
```

```
Index(['id_flow', 'nw_src', 'tp_src', 'nw_dst', 'tp_dst', 'nw_proto',
       'forward_pc', 'forward_bc', 'forward_pl', 'forward_piat', 'forward_pps',
       'forward_bps', 'forward_pl_mean', 'forward_piat_mean',
       'forward_pps_mean', 'forward_bps_mean', 'forward_pl_var',
       'forward_piat_var', 'forward_pps_var', 'forward_bps_var',
       'forward_pl_q1', 'forward_pl_q3', 'forward_piat_q1', 'forward_piat_q3',
       'forward_pl_max', 'forward_pl_min', 'forward_piat_max',
       'forward_piat_min', 'forward_pps_max', 'forward_pps_min',
       'forward_bps_max', 'forward_bps_min', 'forward_duration',
       'forward_size_packets', 'forward_size_bytes', 'reverse_pc',
       'reverse_bc', 'reverse_pl', 'reverse_piat', 'reverse_pps',
       'reverse_bps', 'reverse_pl_mean', 'reverse_piat_mean',
       'reverse_pps_mean', 'reverse_bps_mean', 'reverse_pl_var',
       'reverse_piat_var', 'reverse_pps_var', 'reverse_bps_var',
       'reverse_pl_q1', 'reverse_pl_q3', 'reverse_piat_q1', 'reverse_piat_q3',
       'reverse_pl_max', 'reverse_pl_min', 'reverse_piat_max',
       'reverse_piat_min', 'reverse_pps_max', 'reverse_pps_min',
       'reverse_bps_max', 'reverse_bps_min', 'reverse_duration',
       'reverse_size_packets', 'reverse_size_bytes', 'category',
       'Unnamed: 65'],
      dtype='object')
```

```
[ ]  # imports

     from pyspark.ml.evaluation import MulticlassClassificationEvaluator
     from pyspark.sql import Row, SparkSession
     from pyspark.sql.types import DoubleType, StringType, StructField, StructType
     from pyspark.ml.classification import LogisticRegressionModel, LogisticRegression, NaiveBayes, NaiveBayesModel, RandomFo
     from pyspark.ml.feature import HashingTF, IDF, Tokenizer, VectorAssembler, StringIndexer, VectorIndexer, OneHotEncoder,
     from pyspark.sql.functions import *
```

```
[ ]  # import findspark

     # findspark.init()

     # from pyspark.sql import SparkSession


     # class SDN:
     #     def main():

     #         ss = SparkSession.builder.master("local[*]").appName("SDN_traffic").getOrCreate()
     #         currentDir = os.getcwd()  # get the current directory
     #         inputFile = "./SDN_traffic.csv"

     #         print("Reading from input file: " + inputFile)
     #         # Read the contents of the csv file in a dataframe
     #         OriginalDF = ss.read.option("header", "true").csv(inputFile)
     #         print("Original Dataframe Schema\n")
     #         OriginalDF.printSchema()
```

```
[ ]  # # run main
     # if __name__ == "__main__":
     #     import sys
     #     import os

     #     SDN.main()
```

```
import findspark

findspark.init()

from pyspark.sql import SparkSession

ss = SparkSession.builder.master("local[*]").appName("SDN_traffic").getOrCreate()
currentDir = os.getcwd()  # get the current directory
inputFile = "./SDN_traffic.csv"

print("Reading from input file: " + inputFile)
# Read the contents of the csv file in a dataframe
OriginalDF = ss.read.option("header", "true").csv(inputFile)
print("Original Dataframe Schema\n")
OriginalDF.printSchema()
```

```
OriginalDF.printSchema()
```

```
Reading from input file: ./SDN_traffic.csv
Original Dataframe Schema

root
 |-- id_flow: string (nullable = true)
 |-- nw_src: string (nullable = true)
 |-- tp_src: string (nullable = true)
 |-- nw_dst: string (nullable = true)
 |-- tp_dst: string (nullable = true)
 |-- nw_proto: string (nullable = true)
 |-- forward_pc: string (nullable = true)
 |-- forward_bc: string (nullable = true)
 |-- forward_pl: string (nullable = true)
 |-- forward_piat: string (nullable = true)
 |-- forward_pps: string (nullable = true)
 |-- forward_bps: string (nullable = true)
 |-- forward_pl_mean: string (nullable = true)
 |-- forward_piat_mean: string (nullable = true)
 |-- forward_pps_mean: string (nullable = true)
 |-- forward_bps_mean: string (nullable = true)
 |-- forward_pl_var: string (nullable = true)
 |-- forward_piat_var: string (nullable = true)
 |-- forward_pps_var: string (nullable = true)
 |-- forward_bps_var: string (nullable = true)
 |-- forward_pl_q1: string (nullable = true)
 |-- forward_pl_q3: string (nullable = true)
```

```
# get the size of the original dataset
datasetSize = OriginalDF.count()
print("Original dataset size:" + str(datasetSize) + "\n")
```

```
Original dataset size:4234
```

Preprocessing

Σε αυτήν την ενότητα παρουσιάζεται η επιλογή των στηλών από τις έξι αρχικά που επιλέχθηκαν και στην συνέχεια ανά δύο παρουσιάζοντας και συγκρίνοντας τις προβλέψεις.

```
# Find records which are DNS
print("Dataset including DNS cases")
dnsCatDf = OriginalDF.filter(col("category")=="DNS")
dnsCatDf.show()
print("Number of selected DNS cases: " + str(dnsCatDf.count()) + "\n")
```

```
# Find records which are DNS
print("Dataset including DNS cases")
dnsCatDf = OriginalDF.filter(col("category")=="DNS")
dnsCatDf.show()
print("Number of selected DNS cases: " + str(dnsCatDf.count()) + "\n")
```

```
Dataset including DNS cases
+--------------------+-------------+------+-----------+------+--------+----------+----------+----------+------------+
|             id_flow|       nw_src|tp_src|     nw_dst|tp_dst|nw_proto|forward_pc|forward_bc|forward_pl|forward_piat|
+--------------------+-------------+------+-----------+------+--------+----------+----------+----------+------------+
|21f54165c25ac3b7e...|172.16.25.104| 50051|172.16.25.1|    53|      17|         0|         0|         0|           0|
|166af8ade1a6674f4...|172.16.25.104| 38848|172.16.25.1|    53|      17|         0|         0|         0|           0|
|b49ad598220236ba0...|172.16.25.104| 52352|172.16.25.1|    53|      17|         0|         0|         0|           0|
|3dbc74829dae30157...|172.16.25.104| 33554|172.16.25.1|    53|      17|         0|         0|         0|           0|
|a8989063676901166...|172.16.25.104| 33754|172.16.25.1|    53|      17|         0|         0|         0|           0|
|9c957b37d25135d2a...|172.16.25.104| 52138|172.16.25.1|    53|      17|         0|         0|         0|           0|
|1c197338994b7a3e8...|172.16.25.104| 53855|172.16.25.1|    53|      17|         0|         0|         0|           0|
|3f7f9afeb0108056a...|172.16.25.104| 41872|172.16.25.1|    53|      17|         0|         0|         0|           0|
|da183f6b5505605dc...|172.16.25.104| 42765|172.16.25.1|    53|      17|         0|         0|         0|           0|
|e61facc661d34e2bc...|172.16.25.104| 44821|172.16.25.1|    53|      17|         0|         0|         0|           0|
|c28ef67b237a37190...|172.16.25.104| 54059|172.16.25.1|    53|      17|         0|         0|         0|           0|
|0528234317514abfb...|172.16.25.104| 47077|172.16.25.1|    53|      17|         0|         0|         0|           0|
|81cf1b95bc982eb0f...|172.16.25.104| 35154|172.16.25.1|    53|      17|         0|         0|         0|           0|
|390403eae71a6a4b1...|172.16.25.104| 43583|172.16.25.1|    53|      17|         0|         0|         0|           0|
|70f214cc1af961c29...|172.16.25.104| 33386|172.16.25.1|    53|      17|         0|         0|         0|           0|
|ef700527181e5a9fc...|172.16.25.104| 44598|172.16.25.1|    53|      17|         0|         0|         0|           0|
|a5bddbf93462016c5...|172.16.25.104| 49721|172.16.25.1|    53|      17|         1|        69|        69|           5|
|ca05f677cf9bc133b...|172.16.25.104| 37496|172.16.25.1|    53|      17|         0|         0|         0|           0|
|2981a0f1d239b1f38...|172.16.25.104| 45838|172.16.25.1|    53|      17|         0|         0|         0|           0|
```

```
# Find records which are WWW
print("Dataset including WWW cases")
wwwCatDf = OriginalDF.filter(col("category")=="WWW")
wwwCatDf.show()
print("Number of selected WWW cases: " + str(wwwCatDf.count()) + "\n")
```

```
Dataset including WWW cases
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|             id_flow|       nw_src|tp_src|       nw_dst|tp_dst|nw_proto|forward_pc|forward_bc| forward_pl|forward_piat|
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|b2bb77a570fcfa932...|172.16.25.104| 41402|34.107.221.82|    80|       6|         5|       300|        60|           6|
|f07977b0d1d6645c4...|172.16.25.104| 41406|34.107.221.82|    80|       6|         5|       300|        60|           6|
|e4026ba9b6c195751...|172.16.25.104| 38232|  52.84.77.43|   443|       6|         3|       198|        66|          10|
|e2d747932e41500b1...|172.16.25.104| 38234|  52.84.77.43|   443|       6|         3|       198|        66|          10|
|56325703391225ad6...|172.16.25.104| 60166|  52.32.34.32|   443|       6|         4|       265|     66.25|         7.5|
|7c470792f8382287f...|172.16.25.104| 47356|  192.16.58.8|    80|       6|         4|       240|        60|         7.5|
|3faf9ae90a8890572...|172.16.25.104| 33978|  54200217149|   443|       6|         6|       385|64.16666667|           5|
|91bea0a2e15fe369e...|172.16.25.104| 51860|99.84.160.120|   443|       6|         4|       258|      64.5|         7.5|
|d10ec90ad4a9fef70...|172.16.25.104| 51040|172.217.28.67|    80|       6|         3|       180|        60|          10|
|b87f199764d4c424c...|172.16.25.104| 46978| 52.43.72.100|   443|       6|         8|       505|     63125|        3875|
|3dac57bcaed702bd8...|172.16.25.104| 47376|  192.16.58.8|    80|       6|         3|       180|        60| 10.33333333|
|e8f50f435d41f8965...|172.16.25.104| 35918|  35186227140|   443|       6|         3|       198|        66|           0|
|55599654f34b1616d...|172.16.25.104| 49304| 34.95.71.207|   443|       6|         2|       138|        69|           4|
|0d9ed7e0869d544f4...|172.16.25.104| 55924|  52.84.77.50|   443|       6|         3|       198|        66| 1.333333333|
|20e1dd0f4e323337c...|172.16.25.104| 49596|   192.0.73.2|   443|       6|         1|        60|        60|           3|
|651c60d03d21a7356...|172.16.25.104| 42270|172.217.30.42|   443|       6|         3|       198|        66|           1|
|0caf4ccba8991a6d5...|172.16.25.104| 41152|186.192.81.31|   443|       6|         1|        60|        60|           1|
|41361e8b4df740059...|172.16.25.104| 45994|186.192.81.62|   443|       6|         1|        60|        60|           2|
```

```
# Find records which are VOIP
print("Dataset including VOIP cases")
voipCatDf = OriginalDF.filter(col("category")=="VOIP")
voipCatDf.show()
print("Number of selected VOIP cases: " + str(voipCatDf.count()) + "\n")
```

```
Dataset including VOIP cases
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|             id_flow|       nw_src|tp_src|       nw_dst|tp_dst|nw_proto|forward_pc|forward_bc|forward_pl|forward_piat|
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|3e1b9c4e47f0e19dc...|172.16.25.101| 41418|172.16.25.102| 10001|      17|        30|      4740|       158|           1|
|b25f9ffece0d0d390...|172.16.25.102| 54463|172.16.25.103| 10001|      17|       885|    139830|       158| 0.035028249|
|132d438ba4d4c1589...|172.16.25.103| 45486|172.16.25.101| 10001|      17|       511|     80738|       158| 0.062622309|
|f453f64c9e1ad89a1...|172.16.25.101| 59321|172.16.25.102| 10001|      17|       991|    156578|       158| 0.032290616|
|70f5e386901ac2d83...|172.16.25.102| 46934|172.16.25.103| 10001|      17|       180|     28440|       158| 0.177777778|
|d58fee04c801174c5...|172.16.25.103| 38783|172.16.25.101| 10001|      17|       717|    113286|       158| 0.044630404|
|60be69610b44bfc3d...|172.16.25.101| 42088|172.16.25.102| 10001|      17|      1560|    246480|       158| 0.021153846|
|9588ff0ff105bcc69...|172.16.25.102| 46425|172.16.25.103| 10001|      17|       673|    106334|       158| 0.049034175|
|7c814f281099cc984...|172.16.25.103| 51526|172.16.25.101| 10001|      17|      1277|    201766|       158| 0.025841817|
|f938f84e5720096d7...|172.16.25.101| 58837|172.16.25.102| 10001|      17|      1239|    195762|       158| 0.026634383|
|2488d80284cde90a4...|172.16.25.103| 41029|172.16.25.101| 10001|      17|       127|     20066|       158| 0.251968504|
|763100559b1314c6f...|172.16.25.101| 51374|172.16.25.102| 10001|      17|      1345|    212510|       158| 0.023048327|
|a882aced2d7c95a9e...|172.16.25.102| 45054|172.16.25.103| 10001|      17|      1120|    176960|       158| 0.028571429|
|cbb8a5f41a8ea67ef...|172.16.25.102| 55226|172.16.25.103| 10001|      17|       169|     26702|       158| 0.183431953|
|f269c873920831070...|172.16.25.103| 49076|172.16.25.101| 10001|      17|      1177|    185966|       158| 0.026338148|
|d6671408e79479af5...|172.16.25.103| 50531|172.16.25.101| 10001|      17|        97|     15326|       158| 0.329896907|
|f4773be2e0161d5dc...|172.16.25.102| 50612|172.16.25.103| 10001|      17|       431|     68098|       158| 0.069605568|
|292ff3bb113f9bd9b...|172.16.25.101| 60751|172.16.25.102| 10001|      17|       756|    119448|       158| 0.041005291|
```

```
# Find records which are ICMP
print("Dataset including ICMP cases")
icmpCatDf = OriginalDF.filter(col("category")=="ICMP")
icmpCatDf.show()
print("Number of selected ICMP cases: " + str(icmpCatDf.count()) + "\n")
```

```
Dataset including ICMP cases
+--------------------+-------------+------+--------------+------+--------+----------+----------+----------+------------+
|             id_flow|       nw_src|tp_src|        nw_dst|tp_dst|nw_proto|forward_pc|forward_bc|forward_pl|forward_piat|
+--------------------+-------------+------+--------------+------+--------+----------+----------+----------+------------+
|79b4ce73fff937617...|172.16.25.101|     0| 137.226.34.46|     0|       1|        18|      1764|        98| 1.666666667|
|a8aceb52569621cd6...|172.16.25.102|     0| 137.226.34.46|     0|       1|        16|      1568|        98|      1.9375|
|be408346a754fff68...|172.16.25.103|     0| 137.226.34.46|     0|       1|         7|       686|        98|           2|
|ba39d5642eafea7f2...|172.16.25.102|     0|    141.76.2.4|     0|       1|        10|       980|        98|           3|
|255aa4fe8555d7276...|172.16.25.101|     0|    141.76.2.4|     0|       1|         1|        98|        98|          30|
|ae8500b892d7c1925...|172.16.25.103|     0|    141.76.2.4|     0|       1|        15|      1470|        98|           2|
|e041ba5cd7ec0b481...|172.16.25.102|     0|   93187162100|     0|       1|         8|       784|        98|        3.75|
|d35a19a165f659f13...|172.16.25.103|     0|   93187162100|     0|       1|        12|      1176|        98| 1.666666667|
|8add6b04093e73c34...|172.16.25.102|     0|150.203.164.37|     0|       1|         9|       882|        98| 3.333333333|
|d1622188fc16fa169...|172.16.25.101|     0|   93187162100|     0|       1|        27|      2646|        98| 1.148148148|
|4273f9faa049e06a7...|172.16.25.103|     0|150.203.164.37|     0|       1|         4|       392|        98|        7.75|
|8ae6bd041d046140e...|172.16.25.103|     0|213.129.232.18|     0|       1|        29|      2842|        98| 1.034482759|
|a756848563a7f0d52...|172.16.25.103|     0|195.234.45.114|     0|       1|         3|       294|        98| 10.33333333|
|1b83bffce22b84e8d...|172.16.25.102|     0|213.129.232.18|     0|       1|         9|       882|        98| 3.333333333|
|19fe1417959831a7e...|172.16.25.103|     0| 82.209.230.71|     0|       1|        26|      2548|        98| 1.192307692|
|a2c75086440c8cd77...|172.16.25.103|     0|  200.236.31.3|     0|       1|         4|       392|        98|         6.5|
|e17e812379d2f7bb9...|172.16.25.101|     0|150.203.164.37|     0|       1|        14|      1372|        98| 1.571428571|
|21c8f46dca2c1ba7c...|172.16.25.102|     0|195.234.45.114|     0|       1|        19|      1862|        98| 1.631578947|
```

```
# Find records which are FTP
print("Dataset including FTP cases")
ftpCatDf = OriginalDF.filter(col("category")=="FTP")
ftpCatDf.show()
print("Number of selected FTP cases: " + str(ftpCatDf.count()) + "\n")
```

```
Dataset including FTP cases
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|             id_flow|       nw_src|tp_src|       nw_dst|tp_dst|nw_proto|forward_pc|forward_bc| forward_pl|forward_piat|
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
|aa5e8ee6b0a93d124...|172.16.25.101| 46133|172.16.25.129|    20|       6|     54419| 127344669|2340.077344|  0.00058803|
|b7cf3037c2d770743...|172.16.25.103| 58023|172.16.25.129|    20|       6|     88849| 229796591|2586.372283| 0.000337652|
|d7b6c2ae757eb2546...|172.16.25.103| 45457|172.16.25.129|    20|       6|     41716|  77845116|1866.073353| 0.000767092|
|c37a4dfa628d83850...|172.16.25.101| 33289|172.16.25.129|    20|       6|     15436|  30038234|1945.985618|  0.00213786|
|853750fcf1af33a7c...|172.16.25.103| 40471|172.16.25.129|    20|       6|    113107| 211775688|1872.348201| 0.000291759|
|6ae86cde5bf7b712a...|172.16.25.103| 45339|172.16.25.129|    20|       6|     33105|  61724536| 1864.50796| 0.000996828|
|bae523140ef86db98...|172.16.25.103| 39639|172.16.25.129|    20|       6|     44435|  87319206|1965.099719| 0.000720153|
|199079a2d88b91de5...|172.16.25.101| 37855|172.16.25.129|    20|       6|    128289| 265758222|2071.558918| 0.000257232|
|fc5bd3ebcb8e9c444...|172.16.25.102| 52807|172.16.25.129|    20|       6|         2|       132|        66|         5.5|
|0c71c080538993036...|172.16.25.103| 52411|172.16.25.129|    20|       6|     74659| 145670494|1951.144457|  0.00044201|
|226206d4fc3e62fc7...|172.16.25.103| 55007|172.16.25.129|    20|       6|    101239| 198667563|1962.361965| 0.000325961|
|1e20062fc8c2694f3...|172.16.25.101| 56205|172.16.25.129|    20|       6|    126819| 237528756|1872.974523| 0.000252328|
|4e5044697b3f037ec...|172.16.25.103| 35467|172.16.25.129|    20|       6|     63426| 110237592|1738.050516| 0.000504525|
|0830567499e495132...|172.16.25.103| 46089|172.16.25.129|    20|       6|    110489| 224223911|2029.377685| 0.000280571|
|ddea4d814ecda4daa...|172.16.25.101| 55487|172.16.25.129|    20|       6|     28870|  55238427|1913.350433| 0.001073779|
|a5879e47914b9d5a8...|172.16.25.102| 50911|172.16.25.129|    20|       6|         2|       132|        66|           5|
|48b6f19229d2eb67e...|172.16.25.103| 48593|172.16.25.129|    20|       6|     13093|  25557213| 1951.97533| 0.002367677|
|651c5ac232c0e2746...|172.16.25.102| 47613|172.16.25.129|    20|       6|         2|       132|        66|        11.5|
+--------------------+-------------+------+-------------+------+--------+----------+----------+----------+------------+
```

```
# Find records which are P2P
print("Dataset including P2P cases")
p2pCatDf = OriginalDF.filter(col("category")=="P2P")
p2pCatDf.show()
print("Number of selected P2P cases: " + str(p2pCatDf.count()) + "\n")
```

```
Dataset including P2P cases
+--------------------+--------------+------+--------------+------+--------+----------+----------+----------+------------+
|             id_flow|        nw_src|tp_src|       nw_dst|tp_dst|nw_proto|forward_pc|forward_bc|forward_pl|forward_piat|
+--------------------+--------------+------+--------------+------+--------+----------+----------+----------+------------+
|bb58263461eb5a5e0...|    36.91.54.93| 20949|172.16.25.104| 42737|       6|         0|         0|         0|           0|
|8ff73550cc6f9d30b...| 172.93.177.52| 38860|172.16.25.104| 35885|       6|         0|         0|         0|           0|
|dafe4cec8b14ad1c3...|     37.3.94.67| 17586|172.16.25.104| 53565|       6|         1|        60|        60|          31|
|ca499186539058202...| 62.44.138.147| 65133|172.16.25.104| 33881|       6|         1|        60|        60|          32|
|4ca73e2e3783c0cc2...|197.210.64.107| 48156|172.16.25.104| 43073|       6|         2|       120|        60|        15.5|
|47907fca5fe95d6c3...|197.210.64.223| 39619|172.16.25.104| 38215|       6|         2|       120|        60|        15.5|
|45a48ea542e0f1282...|136.158.33.112| 65057|172.16.25.104| 33851|       6|         0|         0|         0|           0|
|486abd084c7d33550...| 111.68.98.139| 60486|172.16.25.104| 52809|       6|         0|         0|         0|           0|
|471ec0f08889f6e14...|  46.204.17.58| 14554|172.16.25.104| 47501|       6|         0|         0|         0|           0|
|2b478bb4e6f5b0d28...|180.191.194.77| 41405|172.16.25.104| 55795|       6|         0|         0|         0|           0|
|eedc4516a40975cf8...|   1.70238E+11| 10310|172.16.25.104| 45093|       6|         0|         0|         0|           0|
|e28d01be5e66bf80d...| 172.16.25.104| 54315|103.219.236.5|    43|       6|         2|       148|        74|        15.5|
|9a9a3c7dd6cbd6c9e...| 49.150.98.207| 56369|172.16.25.104| 45337|       6|         0|         0|         0|           0|
|e62eafaed341fc039...| 180.243.0.192| 45267|172.16.25.104| 55547|       6|         0|         0|         0|           0|
|6c23db37aaffcb318...|   77.219.2.82| 50545|172.16.25.104| 44893|       6|         1|        60|        60|          31|
|a544f1aa0cef8bbef...| 103.247.51.26|  6881|172.16.25.104| 58731|       6|         0|         0|         0|           0|
|0d6b42ddcc518e67a...|189.163.68.159| 35259|172.16.25.104| 44763|       6|         0|         0|         0|           0|
|11cab7c95759385af...| 39.115.187.92| 26075|172.16.25.104| 50671|       6|         2|       120|        60|        15.5|
+--------------------+--------------+------+--------------+------+--------+----------+----------+----------+------------+
```

```
# 'WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P'

# Sample dataset including all ('WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P') categories
# from the original dataset

print("Sample dataset including all ('WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P') cases from the original dataset")
SampleDF = dnsCatDf.union(wwwCatDf)
SampleDF = SampleDF.union(voipCatDf)
SampleDF = SampleDF.union(icmpCatDf)
SampleDF = SampleDF.union(ftpCatDf)
SampleDF = SampleDF.union(p2pCatDf).distinct()
SampleDF.show()
print("Sample dataset size: " + str(SampleDF.count()) + "\n")
```

```
Sample dataset including all ('WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P') cases from the original dataset
+--------------------+--------------+------+--------------+------+--------+----------+----------+----------+------------+
|             id_flow|        nw_src|tp_src|        nw_dst|tp_dst|nw_proto|forward_pc|forward_bc|forward_pl|forward_piat|
+--------------------+--------------+------+--------------+------+--------+----------+----------+----------+------------+
|9b6cbed197d543da6...| 172.16.25.104| 54398| 52.84.77.115|   443|       6|         1|        60|        60|           0|
|810ec88309aed7281...| 172.16.25.104| 34368|142.250.0.189|   443|       6|         2|       120|        60|         3.5|
|d2bc1a8aaefc9635d...| 172.16.25.104| 47318|23.38.154.216|   443|       6|         4|       258|      64.5|         8.5|
|2e2131ecd29558c5a...| 172.16.25.104| 38280| 189.26.4.160|    80|       6|         3|       180|        60| 10.33333333|
|b0d5776173fb209ce...| 172.16.25.104| 49114|52.46.136.126|   443|       6|         1|        60|        60|          30|
|f9cfea9e23e8d6ef5...| 172.16.25.104| 37702|  208.94.3.16|   443|       6|         2|       145|      72.5|        15.5|
|d79ea8074728bf85e...| 172.16.25.104| 58360|  192.16.58.8|    80|       6|         4|       240|        60|        7.75|
|af125ba3a64467cf1...| 172.16.25.104| 53212|  69164206232|   443|       6|         3|       198|        66| 2.333333333|
|56804cc2c62ffdf95...| 172.16.25.104| 49348|52.46.145.112|   443|       6|         1|        60|        60|          31|
|d4d9b3a85566df315...| 172.16.25.104| 45050| 172.217.28.3|   443|       6|         2|       138|        69|         3.5|
```

```
## DF columns
# Index(['id_flow', 'nw_src', 'tp_src', 'nw_dst', 'tp_dst', 'nw_proto',
#        'forward_pc', 'forward_bc', 'forward_pl', 'forward_piat', 'forward_pps',
#        'forward_bps', 'forward_pl_mean', 'forward_piat_mean',
#        'forward_pps_mean', 'forward_bps_mean', 'forward_pl_var',
#        'forward_piat_var', 'forward_pps_var', 'forward_bps_var',
#        'forward_pl_q1', 'forward_pl_q3', 'forward_piat_q1', 'forward_piat_q3',
#        'forward_pl_max', 'forward_pl_min', 'forward_piat_max',
#        'forward_piat_min', 'forward_pps_max', 'forward_pps_min',
#        'forward_bps_max', 'forward_bps_min', 'forward_duration',
#        'forward_size_packets', 'forward_size_bytes', 'reverse_pc',
#        'reverse_bc', 'reverse_pl', 'reverse_piat', 'reverse_pps',
#        'reverse_bps', 'reverse_pl_mean', 'reverse_piat_mean',
#        'reverse_pps_mean', 'reverse_bps_mean', 'reverse_pl_var',
#        'reverse_piat_var', 'reverse_pps_var', 'reverse_bps_var',
#        'reverse_pl_q1', 'reverse_pl_q3', 'reverse_piat_q1', 'reverse_piat_q3',
#        'reverse_pl_max', 'reverse_pl_min', 'reverse_piat_max',
#        'reverse_piat_min', 'reverse_pps_max', 'reverse_pps_min',
#        'reverse_bps_max', 'reverse_bps_min', 'reverse_duration',
#        'reverse_size_packets', 'reverse_size_bytes', 'category',
#        'Unnamed: 65'],
#       dtype='object')
```

```
from pyspark.sql.functions import array, col

print("Transformed Dataframe")
TransformedDF = SampleDF.select(
    array(
        col("forward_pc"), col("forward_bc"), col("forward_pl"), col("forward_piat"), col("forward_pps"),
        col("forward_bps"), col("forward_pl_mean"), col("forward_piat_mean"), col("forward_pps_mean"),
        col("forward_bps_mean"), col("forward_pl_var"), col("forward_piat_var"), col("forward_pps_var"),
        col("forward_bps_var"), col("forward_pl_q1"), col("forward_pl_q3"), col("forward_piat_q1"),
        col("forward_piat_q3"), col("forward_pl_max"), col("forward_pl_min"), col("forward_piat_max"),
        col("forward_piat_min"), col("forward_pps_max"), col("forward_pps_min"), col("forward_bps_max"),
        col("forward_bps_min"), col("forward_duration"), col("forward_size_packets"), col("forward_size_bytes"),
        col("reverse_pc"), col("reverse_bc"), col("reverse_pl"), col("reverse_piat"), col("reverse_pps"),
        col("reverse_bps"), col("reverse_pl_mean"), col("reverse_piat_mean"), col("reverse_pps_mean"),
        col("reverse_bps_mean"), col("reverse_pl_var"), col("reverse_piat_var"), col("reverse_pps_var"),
        col("reverse_bps_var"), col("reverse_pl_q1"), col("reverse_pl_q3"), col("reverse_piat_q1"),
        col("reverse_piat_q3"), col("reverse_pl_max"), col("reverse_pl_min"), col("reverse_piat_max"),
        col("reverse_piat_min"), col("reverse_pps_max"), col("reverse_pps_min"), col("reverse_bps_max"),
        col("reverse_bps_min"), col("reverse_duration"), col("reverse_size_packets"), col("reverse_size_bytes")
    ).alias("rCriteria"),
    col("category")
)
print("Transformed Dataframe Schema" + "\n")
TransformedDF.printSchema()
```

```
Transformed Dataframe
Transformed Dataframe Schema
```

```
    )
    print("Transformed Dataframe Schema" + "\n")
    TransformedDF.printSchema()
```

```
Transformed Dataframe
Transformed Dataframe Schema

root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- category: string (nullable = true)
```

```
# Rename the columns of the dataframe
print("Rename columns of transformed Dataset")
newColumnNames = ["rCriteria", "rLabel"]
renamedDF = TransformedDF.toDF(*newColumnNames)
print("Renamed dataframe schema")
renamedDF.printSchema()
```

```
Rename columns of transformed Dataset
Renamed dataframe schema
root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- rLabel: string (nullable = true)
```

```
print("Return rows of the renamed datafram")
renamedDF.select("rCriteria", "rLabel").take(20)
```

```
Return rows of the renamed datafram
[Row(rCriteria=['1', '60', '60', '0', '0', '0', '149.95', '6475', '0.087803379', '26.75701997', '20132.56', '60.5275',
'0.009619096', '1401.795358', '60', '253.25', '0', '7.5', '376.5', '0', '21', '0', '0.258064516', '0', '97.16129032',
'0', '84', '14', '4145', '1', '60', '60', '0', '0', '0', '176.7', '6475', '0.087803379', '29.70184332', '24622335',
'60.5275', '0.009619096', '1438.537156', '60', '356.25', '0', '7.5', '374.25', '0', '21', '0', '0.258064516', '0',
'96.58064516', '0', '84', '14', '4572'], rLabel='WWW'),
 Row(rCriteria=['2', '120', '60', '3.5', '0.285714286', '17.14285714', '92.15064103', '2.772435897', '0.861955965',
'159.878469', '4463.555771', '6.188414283', '1.988095251', '97189.96248', '60', '119.3028846', '0.783653846', '3375',
'214', '0', '7.5', '0', '4', '0', '856', '0', '81', '52', '9066', '1', '27', '27', '7', '0.142857143', '3.857142857',
'76.17084079', '3.37395647', '1.068926011', '178.0387353', '3423.563345', '9.186843921', '3.728492412', '130454.1892',
'35.25', '93.57692308', '0.716457961', '6', '183.255814', '0', '7.75', '0', '5375', '0', '985', '0', '81', '62',
'9459'], rLabel='WWW'),
 Row(rCriteria=['4', '258', '64.5', '8.5', '0.117647059', '7.588235294', '61125', '9.541666667', '0.106826537',
'6.541945187', '3.796875', '1.713541667', '0.00021793', '1.008954667', '60', '61125', '8375', '10.75', '64.5', '60',
'11', '8', '125', '0.090909091', '7.588235294', '5.454545455', '131', '14', '858', '5', '300', '60', '6.8',
'0.147058824', '8.823529412', '60', '9.783333333', '0.106366979', '6.382018717', '0', '2.985277778', '0.000553287',
'1.99183395', '60', '60', '9.7', '10.75', '60', '60', '11', '6.8', '0.147058824', '0.090909091', '8.823529412',
'5.454545455', '131', '14', '840'], rLabel='WWW'),
 Row(rCriteria=['3', '180', '60', '10.33333333', '0.096774194', '5.806451613', '60', '10.5', '0.095262097',
'5.715725806', '0', '0.027777778', '2.29E-06', '0.008231172', '60', '60', '10.33333333', '10.66666667', '60', '60',
'10.66666667', '10.33333333', '0.096774194', '0.09375', '5.806451613', '5625', '126', '12', '720', '3', '180', '60',
'10.33333333', '0.096774194', '5.806451613', '60', '10.5', '0.095262097', '5.715725806', '0', '0.027777778', '2.29E-06'
'0.008231172', '60', '60', '10.33333333', '10.66666667', '60', '60', '10.66666667', '10.33333333', '0.096774194',
'0.09375', '5.806451613', '5625', '126', '12', '720'], rLabel='WWW'),
 Row(rCriteria=['1', '60', '60', '30', '0.033333333', '2', '60', '20.16666667', '0.065053763', '3.903225806', '0',
```

```
# Set the number of partitions
print("Set the number of partitions = 4")
sdnDF = renamedDF
sdnDF.repartition(4)
```

```
Set the number of partitions = 4
DataFrame[rCriteria: array<string>, rLabel: string]
```

```
print("Create tf-idf features")
hashingTF = HashingTF().setInputCol("rCriteria").setOutputCol("rRawFeatures")#.setNumFeatures(20000)
featurizedDF = hashingTF.transform(sdnDF)
print("Print Schema of featurized dataset")
featurizedDF.printSchema()
```

```
Create tf-idf features
Print Schema of featurized dataset
root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- rLabel: string (nullable = true)
 |-- rRawFeatures: vector (nullable = true)
```

```
idf = IDF().setInputCol("rRawFeatures").setOutputCol("rFeatures")
idfM = idf.fit(featurizedDF)
completeDF = idfM.transform(featurizedDF)
print("Complete dataframe schema")
completeDF.printSchema()
```

```
Complete dataframe schema
root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- rLabel: string (nullable = true)
 |-- rRawFeatures: vector (nullable = true)
 |-- rFeatures: vector (nullable = true)
```

```
print("Print rows of complete dataframe")
completeDF.select("rFeatures","rCriteria").take(10)
```

```
Print rows of complete dataframe
[Row(rFeatures=SparseVector(262144, {6945: 7.654, 19292: 11.4161, 43180: 3.4084, 46045: 4.9556, 49603: 7.654, 64880:
7.654, 76985: 2.0835, 86989: 7.654, 87466: 6.3992, 89213: 7.654, 92651: 1.7871, 94346: 15.3079, 124721: 7.2485, 127421:
10.3381, 143325: 15.3079, 143357: 7.654, 169743: 15.3079, 181220: 7.654, 191169: 7.654, 200139: 7.654, 200436: 7.654,
211743: 15.3079, 218248: 4.2137, 219367: 7.654, 220669: 7.654, 230061: 7.654, 245224: 7.654, 252840: 7.654}), rCriteria=
['1', '60', '60', '0', '0', '0', '149.95', '6475', '0.087803379', '26.75701997', '20132.56', '60.5275', '0.009619096',
'1401.795358', '60', '253.25', '0', '7.5', '376.5', '0', '21', '0', '0.258064516', '0', '97.16129032', '0', '84', '14',
'4145', '1', '60', '60', '0', '0', '0', '176.7', '6475', '0.087803379', '29.70184332', '24622335', '60.5275',
'0.009619096', '1438.537156', '60', '356.25', '0', '7.5', '374.25', '0', '21', '0', '0.258064516', '0', '96.58064516',
'0', '84', '14', '4572']),
 Row(rFeatures=SparseVector(262144, {12524: 0.9569, 23392: 7.654, 26163: 7.2485, 27185: 7.654, 29734: 7.2485, 31077:
11.0278, 33204: 7.654, 36308: 7.654, 38016: 5.7081, 43180: 1.1361, 49783: 2.4555, 55690: 4.113, 64443: 7.654, 64631:
7.654, 66033: 7.654, 71117: 7.654, 72385: 7.654, 76985: 1.0417, 78861: 7.654, 81849: 7.654, 92651: 0.8936, 98929:
5.7822, 112073: 7.654, 118002: 7.654, 121035: 4.9459, 121661: 3.752, 124674: 1.4585, 131089: 5.5745, 151686: 7.654,
157432: 7.654, 165373: 2.6534, 167694: 2.1226, 179490: 7.654, 183660: 7.654, 190344: 6.1897, 193638: 6.0445, 194802:
1.2422, 199487: 7.654, 199849: 7.654, 203178: 5.3514, 207344: 7.654, 208989: 7.654, 211890: 7.654, 214833: 2.6601,
218248: 2.1068, 226069: 7.654, 233992: 7.654, 253058: 2.492}), rCriteria=['2', '120', '60', '3.5', '0.285714286',
'17.14285714', '92.15064103', '2.772435897', '0.861955965', '159.878469', '4463.555771', '6.188414283', '1.988095251',
'97189.96248', '60', '119.3028846', '0.783653846', '3375', '214', '0', '7.5', '0', '4', '0', '856', '0', '81', '52',
'9066', '1', '27', '27', '7', '0.142857143', '3.857142857', '76.17084079', '3.37395647', '1.068926011', '178.0387353',
'3423.563345', '9.186843921', '3.728492412', '130454.1892', '35.25', '93.57692308', '0.716457961', '6', '183.255814',
'0', '7.75', '0', '5375', '0', '985', '0', '81', '62', '9459']),
 Row(rFeatures=SparseVector(262144, {4255: 6.5554, 8254: 2.0555, 18521: 3.3499, 22682: 9.0809, 24379: 4.7918, 27134:
6.7377, 43180: 4.5445, 46045: 4.9556, 57207: 12.8024, 58225: 6.5554, 76985: 0.1302, 80944: 5.5745, 86892: 12.5353,
87316: 5.0513, 101638: 6.7377, 103775: 5.234, 104358: 6.5554, 105223: 9.1259, 106236: 11.5643, 113912: 5.4567, 114315:
6.4012, 118632: 7.3863, 153795: 6.7377, 157430: 3.2351, 160483: 8.1689, 167536: 13.4754, 188638: 6.7377, 194802: 1.2422,
```

```
# labels:'WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P'
# mapping www, dns, voip, icmp, ftp, p2p to 1,2,3,4,5,6

udf_toDouble = udf( lambda s: 1.0 if s=='WWW' else 2.0 if s=='DNS' else 3.0 if s=='VOIP' else 4.0 if s=='ICMP' else 5.0 )
print("Print schema of new dataframe")
newDF = completeDF.select("rCriteria", "rRawFeatures", "rFeatures",
"rLabel", (udf_toDouble("rLabel").alias("rrLabel")).cast('double'))
newDF.printSchema()
```

```
Print schema of new dataframe
root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- rRawFeatures: vector (nullable = true)
 |-- rFeatures: vector (nullable = true)
 |-- rLabel: string (nullable = true)
 |-- rrLabel: double (nullable = true)
```

```
print("Print rows of complete dataframe")
newDF.select("rLabel", "rrLabel").take(20)
```

```
Print rows of complete dataframe
[Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='VOIP', rrLabel=3.0),
 Row(rLabel='ICMP', rrLabel=4.0),
 Row(rLabel='ICMP', rrLabel=4.0),
 Row(rLabel='P2P', rrLabel=6.0),
 Row(rLabel='P2P', rrLabel=6.0),
 Row(rLabel='P2P', rrLabel=6.0),
 Row(rLabel='P2P', rrLabel=6.0),
 Row(rLabel='P2P', rrLabel=6.0),
 Row(rLabel='P2P', rrLabel=6.0)]
```

CLASSIFICATION

```
newDF.show()
```

```
+--------------------+--------------------+--------------------+------+-------+
|           rCriteria|        rRawFeatures|           rFeatures|rLabel|rrLabel|
+--------------------+--------------------+--------------------+------+-------+
|[1, 60, 60, 0, 0,...|(262144,[6945,192...|(262144,[6945,192...|   WWW|    1.0|
|[2, 120, 60, 3.5,...|(262144,[12524,23...|(262144,[12524,23...|   WWW|    1.0|
|[4, 258, 64.5, 8....|(262144,[4255,825...|(262144,[4255,825...|   WWW|    1.0|
|[3, 180, 60, 10.3...|(262144,[6821,193...|(262144,[6821,193...|   WWW|    1.0|
|[1, 60, 60, 30, 0...|(262144,[12524,18...|(262144,[12524,18...|   WWW|    1.0|
|[2, 145, 72.5, 15...|(262144,[8254,125...|(262144,[8254,125...|   WWW|    1.0|
|[4, 240, 60, 7.75...|(262144,[9268,185...|(262144,[9268,185...|   WWW|    1.0|
|[3, 198, 66, 2.33...|(262144,[6524,715...|(262144,[6524,715...|   WWW|    1.0|
|[1, 60, 60, 31, 0...|(262144,[8254,125...|(262144,[8254,125...|   WWW|    1.0|
|[2, 138, 69, 3.5,...|(262144,[6257,988...|(262144,[6257,988...|   WWW|    1.0|
|[5, 350, 70, 0.8,...|(262144,[4087,526...|(262144,[4087,526...|   WWW|    1.0|
|[926, 146308, 158...|(262144,[6704,586...|(262144,[6704,586...|  VOIP|    3.0|
|[12, 1176, 98, 1....|(262144,[58921,76...|(262144,[58921,76...|  ICMP|    4.0|
|[9, 882, 98, 3.33...|(262144,[493,7698...|(262144,[493,7698...|  ICMP|    4.0|
|[0, 0, 0, 0, 0, 0...|(262144,[8254,769...|(262144,[8254,769...|   P2P|    6.0|
|[1, 60, 60, 32, 0...|(262144,[43180,75...|(262144,[43180,75...|   P2P|    6.0|
|[2, 120, 60, 16.5...|(262144,[9882,125...|(262144,[9882,125...|   P2P|    6.0|
```

```
df = newDF.select("rFeatures", "rrLabel").withColumnRenamed("rFeatures", "features").withColu
print("Show Dataframe with columns <features> and <label>"+"\n")
df.show()
```

```
Show Dataframe with columns <features> and <label>

+--------------------+-----+
|            features|label|
+--------------------+-----+
|(262144,[6945,192...|  1.0|
|(262144,[12524,23...|  1.0|
|(262144,[4255,825...|  1.0|
|(262144,[6821,193...|  1.0|
|(262144,[12524,18...|  1.0|
|(262144,[8254,125...|  1.0|
|(262144,[9268,185...|  1.0|
|(262144,[6524,715...|  1.0|
|(262144,[8254,125...|  1.0|
|(262144,[6257,988...|  1.0|
|(262144,[4087,526...|  1.0|
|(262144,[6704,586...|  3.0|
|(262144,[58921,76...|  4.0|
|(262144,[493,7698...|  4.0|
|(262144,[8254,769...|  6.0|
|(262144,[43180,75...|  6.0|
|(262144,[9882,125...|  6.0|
|(262144,[12524,76...|  6.0|
|(262144,[12524,43...|  6.0|
|(262144,[43180,76...|  6.0|
```

```
print("Split the data into training and test sets (30% held out for testing)")
trainingData, testData = df.randomSplit([0.7, 0.3], seed=1234)
```

```
Split the data into training and test sets (30% held out for testing)
```

NaïveBayes

Naive Bayes Classifier

```
[ ] from pyspark.ml.classification import NaiveBayes
    naivebayes = NaiveBayes(featuresCol="features", labelCol="label")

[ ] from pyspark.ml.tuning import ParamGridBuilder
    param_grid = ParamGridBuilder().\
        addGrid(naivebayes.smoothing, [0, 1, 2, 4]).\
        build()

[ ] from pyspark.ml.evaluation import MulticlassClassificationEvaluator
    evaluator = MulticlassClassificationEvaluator()

[ ] from pyspark.ml.tuning import CrossValidator
    crossvalidator = CrossValidator(estimator=naivebayes, estimatorParamMaps=param_grid, evaluator=evaluator)

[ ] crossvalidation_mode = crossvalidator.fit(trainingData)

[ ] print("Naive Bayes classifier Predictions")
```

```
print("Naive Bayes classifier Predictions")

print("Print Naive Bays dataframe predictions test")
pred_test = crossvalidation_mode.transform(testData)
pred_test.show(5)
```

```
Naive Bayes classifier Predictions
Print Naive Bays dataframe predictions test
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  4.0|[-1848.7753135227...|[8.30192455940876...|       3.0|
|(262144,[4087,526...|  1.0|[-2241.8777629162...|[1.0,2.5406825645...|       0.0|
|(262144,[6524,715...|  1.0|[-2162.3996921069...|[1.0,1.7383716917...|       0.0|
|(262144,[2292,625...|  1.0|[-1394.5642996798...|[1.0,4.8687893284...|       0.0|
|(262144,[4758,825...|  1.0|[-1077.5558480647...|[1.0,4.2323114490...|       0.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 5 rows
```

```
# Best model from cross validation
print("The parameter smoothing has best value:",
      crossvalidation_mode.bestModel._java_obj.getSmoothing())
```

```
The parameter smoothing has best value: 1.0
```

```
print("Naive Bayes Metrics")
Evaluator_Accuracy = evaluator.setMetricName("accuracy")
Accuracy_NB = Evaluator_Accuracy.evaluate(pred_test)
print("Accuracy for Naive Bayes =", Accuracy_NB)

Evaluator_Precision = evaluator.setMetricName("weightedPrecision")
Precision_NB = Evaluator_Precision.evaluate(pred_test)
print("Precision for Naive Bayes =", Precision_NB)

Evaluator_Recall = evaluator.setMetricName("weightedRecall")
Recall_NB = Evaluator_Recall.evaluate(pred_test)
print("Recall for Naive Bayes =", Recall_NB)

Evaluator_F1 = evaluator.setMetricName("f1")
F1_NB = Evaluator_F1.evaluate(pred_test)
print("F1 for Naive Bayes = ", F1_NB)
```

```
## Not a good NaiveBayes classifier for our case with 6 classes
# from pyspark.ml.classification import NaiveBayes

# # Classification example using Naive Bayes Classifier
# NBmodel = NaiveBayes(modelType="multinomial").fit(trainingData)
# print("Naive Bayes classifier Predictions")
# predictionsNB = NBmodel.transform(testData)
# print("Schema of Naive Bayes model dataframe")
# predictionsNB.printSchema()

# print("Print Naive Bays dataframe")
# predictionsNB.show()
```

```
Naive Bayes Metrics
Accuracy for Naive Bayes = 0.015267175572519083
Precision for Naive Bayes = 0.23811862329646963
Recall for Naive Bayes = 0.015267175572519083
F1 for Naive Bayes =  0.027650704594137168
```

```
Naive Bayes classifier Predictions
Schema of Naive Bayes model dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- rawPrediction: vector (nullable = true)
 |-- probability: vector (nullable = true)
 |-- prediction: double (nullable = false)

Print Naive Bays dataframe
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  4.0|[-1848.7753135227...|[8.30192455940876...|       3.0|
|(262144,[4087,526...|  1.0|[-2241.8777629162...|[1.0,2.5406825645...|       0.0|
|(262144,[6524,715...|  1.0|[-2162.3996921069...|[1.0,1.7383716917...|       0.0|
|(262144,[2292,625...|  1.0|[-1394.5642996798...|[1.0,4.8687893284...|       0.0|
|(262144,[4758,825...|  1.0|[-1077.5558480647...|[1.0,4.2323114490...|       0.0|
|(262144,[12524,43...|  6.0|[-244.37367107730...|[0.00161151642308...|       5.0|
|(262144,[30483,76...|  4.0|[-1950.4976360053...|[5.83932218669548...|       3.0|
|(262144,[37135,57...|  4.0|[-2259.4950407170...|[2.0E-323,0.0,0.0...|       3.0|
|(262144,[41070,54...|  5.0|[-4297.0336609326...|[6.36510439578388...|       4.0|
|(262144,[493,2374...|  4.0|[-2014.5785339574...|[4.07872394328699...|       3.0|
|(262144,[4731,110...|  3.0|[-2357.0603601263...|[5.05941446164458...|       2.0|
|(262144,[12034,12...|  1.0|[-3048.7596950131...|[1.0,4.3665343202...|       0.0|
|(262144,[12524,19...|  4.0|[-2823.9933118440...|[6.44547657452727...|       3.0|
```

Naive Bayes Classifier 2 classes (www or non-WWW)

```
# labels:'WWW', 'DNS', 'VOIP', 'ICMP', 'FTP', 'P2P'
# mapping www, dns, voip, icmp, ftp, p2p to 1, 0 (all the non WWW cases)

udf_toDouble_c2 = udf( lambda s: 1.0 if s=='WWW' else 0.0 )
print("Print schema of new dataframe")
newDF_c2 = completeDF.select("rCriteria", "rRawFeatures", "rFeatures",
"rLabel", (udf_toDouble_c2("rLabel").alias("rrLabel")).cast('double'))
newDF_c2.printSchema()
```

```
Print schema of new dataframe
root
 |-- rCriteria: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- rRawFeatures: vector (nullable = true)
 |-- rFeatures: vector (nullable = true)
 |-- rLabel: string (nullable = true)
 |-- rrLabel: double (nullable = true)
```

```
print("Print rows of complete dataframe")
newDF_c2.select("rLabel", "rrLabel").take(20)
```

```
Print rows of complete dataframe
[Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='WWW', rrLabel=1.0),
 Row(rLabel='VOIP', rrLabel=0.0),
 Row(rLabel='ICMP', rrLabel=0.0),
 Row(rLabel='ICMP', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0),
 Row(rLabel='P2P', rrLabel=0.0)]
```

```
df_c2 = newDF_c2.select("rFeatures", "rrLabel").withColumnRenamed("rFeatures", "features").wi
print("Show Dataframe with columns <features> and <label>"+"\n")
df_c2.show()
```

```
Show Dataframe with columns <features> and <label>

+--------------------+-----+
|            features|label|
+--------------------+-----+
|(262144,[6945,192...|  1.0|
|(262144,[12524,23...|  1.0|
|(262144,[4255,825...|  1.0|
|(262144,[6821,193...|  1.0|
|(262144,[12524,18...|  1.0|
|(262144,[8254,125...|  1.0|
|(262144,[9268,185...|  1.0|
|(262144,[6524,715...|  1.0|
|(262144,[8254,125...|  1.0|
|(262144,[6257,988...|  1.0|
|(262144,[4087,526...|  1.0|
|(262144,[6704,586...|  0.0|
|(262144,[58921,76...|  0.0|
|(262144,[493,7698...|  0.0|
|(262144,[8254,769...|  0.0|
|(262144,[43180,75...|  0.0|
|(262144,[9882,125...|  0.0|
|(262144,[12524,76...|  0.0|
|(262144,[12524,43...|  0.0|
|(262144,[43180,76...|  0.0|
```

```
print("Split the data into training and test sets (30% held out for testing)")
trainingData_c2, testData_c2 = df_c2.randomSplit([0.7, 0.3], seed=1234)
```

```
Split the data into training and test sets (30% held out for testing)
```

```
print("Print Naive Bays dataframe for 2 Classes")
predictionsNB_c2.show()
```

```
Naive Bayes classifier Predictions for 2 Classes
Schema of Naive Bayes model dataframe for 2 Classes
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- rawPrediction: vector (nullable = true)
 |-- probability: vector (nullable = true)
 |-- prediction: double (nullable = false)

Print Naive Bays dataframe for 2 Classes
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  0.0|[-1374.8627046137...|[1.0,1.5526782549...|       0.0|
|(262144,[4087,526...|  1.0|[-2490.4259327255...|[1.11411299379054...|       1.0|
|(262144,[6524,715...|  1.0|[-2566.7024297428...|[2.53069938146481...|       1.0|
|(262144,[2292,625...|  1.0|[-1836.4841177041...|[1.17362636186518...|       1.0|
|(262144,[4758,825...|  1.0|[-1567.8466595762...|[1.15279000143891...|       1.0|
|(262144,[12524,43...|  0.0|[-234.79263886494...|[0.99993173425618...|       0.0|
|(262144,[30483,76...|  0.0|[-1495.4803931858...|[1.0,2.5021798410...|       0.0|
|(262144,[37135,57...|  0.0|[-1581.7014419048...|[1.0,4.2847145573...|       0.0|
|(262144,[41070,54...|  0.0|[-4114.5491672264...|[1.0,5.8150740958...|       0.0|
|(262144,[493,2374...|  0.0|[-1548.7627381752...|[1.0,4.9221937135...|       0.0|
|(262144,[4731,110...|  0.0|[-2231.8962446084...|[1.0,4.4324728894...|       0.0|
|(262144,[12034,12...|  1.0|[-3145.4702179322...|[9.70567248057976...|       1.0|
|(262144,[12524,19...|  0.0|[-2356.1891147714...|[1.0,7.0037460703...|       0.0|
```

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

evaluator = MulticlassClassificationEvaluator() \
    .setLabelCol("label") \
    .setPredictionCol("prediction")
```

```
print("Naive Bayes classifier Metrics for 2 Classes (WWW or non-WWW)")
Evaluator_Accuracy = evaluator.setMetricName("accuracy")
Accuracy_NB = Evaluator_Accuracy.evaluate(predictionsNB_c2)
print("Accuracy for Naive Bayes classifier =", Accuracy_NB)

Evaluator_Precision = evaluator.setMetricName("weightedPrecision")
Precision_NB = Evaluator_Precision.evaluate(predictionsNB_c2)
print("Precision for Naive Bayes classifier =", Precision_NB)

Evaluator_Recall = evaluator.setMetricName("weightedRecall")
Recall_NB = Evaluator_Recall.evaluate(predictionsNB_c2)
print("Recall for Naive Bayes classifier =", Recall_NB)

Evaluator_F1 = evaluator.setMetricName("f1")
F1_NB = Evaluator_F1.evaluate(predictionsNB_c2)
print("F1 for Naive Bayes classifier = ", F1_NB)
```

```
Naive Bayes classifier Metrics for 2 Classes (WWW or non-WWW)
Accuracy for Naive Bayes classifier = 0.9190839694656489
Precision for Naive Bayes classifier = 0.9189765407140271
Recall for Naive Bayes classifier = 0.919083969465649
F1 for Naive Bayes classifier =  0.9188882364454883
```

Logistic Regression

```
from pyspark.ml.classification import LogisticRegression

# Classification example using Logistic Regression Classifier
LRmodel = LogisticRegression() \
    .setMaxIter(10000) \
    .setRegParam(0.1) \
    .setElasticNetParam(0.0) \
    .fit(trainingData)

predictionsLR = LRmodel.transform(testData)
print("Schema of Logistic Regression dataframe")
predictionsLR.printSchema()

print("Print Logistic Regression dataframe")
predictionsLR.show()
```

```
Schema of Logistic Regression dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- rawPrediction: vector (nullable = true)
 |-- probability: vector (nullable = true)
 |-- prediction: double (nullable = false)
```

```
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  4.0|[-9.3078630500768...|[2.05540575815717...|       4.0|
|(262144,[4087,526...|  1.0|[-9.3078536508630...|[3.23334643470935...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078429662581...|[2.11369810870502...|       1.0|
|(262144,[2292,625...|  1.0|[-9.3078567423627...|[1.22204133141243...|       1.0|
|(262144,[4758,825...|  1.0|[-9.3078565356683...|[1.78461855870915...|       1.0|
|(262144,[12524,43...|  6.0|[-9.3078780508099...|[1.15859619395442...|       6.0|
|(262144,[30483,76...|  4.0|[-9.3078682182321...|[4.43757079568026...|       4.0|
|(262144,[37135,57...|  4.0|[-9.3078589155435...|[2.69312373335182...|       4.0|
|(262144,[41070,54...|  5.0|[-9.3078105381218...|[1.84711552285391...|       6.0|
|(262144,[493,2374...|  4.0|[-9.3078618568025...|[2.91330387376916...|       4.0|
|(262144,[4731,110...|  3.0|[-9.3078436392892...|[2.18709366216611...|       6.0|
|(262144,[12034,12...|  1.0|[-9.3078470847790...|[1.37325607739731...|       1.0|
|(262144,[12524,19...|  4.0|[-9.3078547484739...|[9.69904610692073...|       4.0|
|(262144,[61509,76...|  1.0|[-9.3078532055441...|[3.25231619621963...|       1.0|
|(262144,[3616,263...|  1.0|[-9.3078400023085...|[6.29494264461110...|       1.0|
|(262144,[3774,610...|  1.0|[-9.3078578680097...|[1.23672243531380...|       1.0|
|(262144,[4758,825...|  1.0|[-9.3078565356683...|[1.78461855870915...|       1.0|
|(262144,[5710,125...|  1.0|[-9.3078533092818...|[3.70536619074687...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078572266494...|[1.18658204579589...|       1.0|
|(262144,[6524,715...|  1.0|[-9.3078575698008...|[1.17557070855986...|       1.0|
+--------------------+-----+--------------------+--------------------+----------+
only showing top 20 rows
```

```
[ ]  from pyspark.ml.evaluation import MulticlassClassificationEvaluator

     evaluator = MulticlassClassificationEvaluator() \
         .setLabelCol("label") \
         .setPredictionCol("prediction")
```

```
print("Logistic Regression Metrics")
Evaluator_Accuracy = evaluator.setMetricName("accuracy")
Accuracy_LR = Evaluator_Accuracy.evaluate(predictionsLR)
print("Accuracy for Logistic Regression =", Accuracy_LR)

Evaluator_Precision = evaluator.setMetricName("weightedPrecision")
Precision_LR = Evaluator_Precision.evaluate(predictionsLR)
print("Precision for Logistic Regression =", Precision_LR)

Evaluator_Recall = evaluator.setMetricName("weightedRecall")
Recall_LR = Evaluator_Recall.evaluate(predictionsLR)
print("Recall for Logistic Regression =", Recall_LR)

Evaluator_F1 = evaluator.setMetricName("f1")
F1_LR = Evaluator_F1.evaluate(predictionsLR)
print("F1 for Logistic Regression = ", F1_LR)
```

```
Logistic Regression Metrics
Accuracy for Logistic Regression = 0.8412213740458016
Precision for Logistic Regression = 0.8082027197321239
Recall for Logistic Regression = 0.8412213740458016
F1 for Logistic Regression =  0.7957827232475606
```

```
from pyspark.ml.classification import LogisticRegression

# # Classification example using Logistic Regression Classifier
# LRmodel_c2 = LogisticRegression() \
#     .setMaxIter(10000) \
#     .setRegParam(0.1) \
#     .setElasticNetParam(0.0) \
#     .fit(trainingData_c2)

predictionsLR_c2 = LRmodel_c2.transform(testData_c2)
print("Schema of Logistic Regression dataframe")
predictionsLR_c2.printSchema()

print("Print Logistic Regression dataframe")
predictionsLR_c2.show()
```

```
Schema of Logistic Regression dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- rawPrediction: vector (nullable = true)
 |-- probability: vector (nullable = true)
 |-- prediction: double (nullable = false)
```

```
Print Logistic Regression dataframe
+--------------------+-----+--------------------+--------------------+----------+
|            features|label|       rawPrediction|         probability|prediction|
+--------------------+-----+--------------------+--------------------+----------+
|(262144,[493,7698...|  0.0|[2.97348583548864...|[0.95136182933893...|       0.0|
|(262144,[4087,526...|  1.0|[-2.4461484094555...|[0.07972066518319...|       1.0|
|(262144,[6524,715...|  1.0|[-3.0725219956060...|[0.04425503324684...|       1.0|
|(262144,[2292,625...|  1.0|[-3.8936873768245...|[0.01996343803458...|       1.0|
|(262144,[4758,825...|  1.0|[-3.4085075786648...|[0.03203063722598...|       1.0|
|(262144,[12524,43...|  0.0|[1.53532047281826...|[0.82278342967203...|       0.0|
|(262144,[30483,76...|  0.0|[2.81068367880512...|[0.94325042696653...|       0.0|
|(262144,[37135,57...|  0.0|[3.38434312232355...|[0.96721161978953...|       0.0|
|(262144,[41070,54...|  0.0|[1.54295157913317...|[0.82389338839687...|       0.0|
|(262144,[493,2374...|  0.0|[3.11770286032205...|[0.95761709302308...|       0.0|
|(262144,[4731,110...|  0.0|[1.65159317737120...|[0.83910625665687...|       0.0|
|(262144,[12034,12...|  1.0|[0.07716894364691...|[0.51928266776375...|       0.0|
|(262144,[12524,19...|  0.0|[2.83590639927652...|[0.94458557771491...|       0.0|
|(262144,[61509,76...|  1.0|[-2.4924909249386...|[0.07638627354122...|       1.0|
```

```
print("Logistic Regression Metrics")
Evaluator_Accuracy_c2 = evaluator_c2.setMetricName("accuracy")
Accuracy_LR_c2 = Evaluator_Accuracy_c2.evaluate(predictionsLR_c2)
print("Accuracy for Logistic Regression =", Accuracy_LR_c2)

Evaluator_Precision_c2 = evaluator_c2.setMetricName("weightedPrecision")
Precision_LR_c2 = Evaluator_Precision_c2.evaluate(predictionsLR_c2)
print("Precision for Logistic Regression =", Precision_LR_c2)

Evaluator_Recall_c2 = evaluator_c2.setMetricName("weightedRecall")
Recall_LR_c2 = Evaluator_Recall_c2.evaluate(predictionsLR_c2)
print("Recall for Logistic Regression =", Recall_LR_c2)

Evaluator_F1_c2 = evaluator_c2.setMetricName("f1")
F1_LR_c2 = Evaluator_F1_c2.evaluate(predictionsLR_c2)
print("F1 for Logistic Regression = ", F1_LR_c2)
```

```
Logistic Regression Metrics
Accuracy for Logistic Regression = 0.9206106870229007
Precision for Logistic Regression = 0.9252016816768205
Recall for Logistic Regression = 0.9206106870229007
F1 for Logistic Regression =  0.9210916048110697
```

Linear Regression

```
from pyspark.ml.regression import LinearRegression

# # Classification example using Linear Regression Classifier
# LiRModel = (
#     LinearRegression()
#     .setMaxIter(10000)
#     .setRegParam(0.1)
#     .setElasticNetParam(0.0)
#     .fit(trainingData)
# )

predictionsLiR = LiRModel.transform(testData)
print("Schema of Linear Regression dataframe")
predictionsLiR.printSchema()

print("Print Linear Regression dataframe")
predictionsLiR.show()
```

```
Schema of Linear Regression dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- prediction: double (nullable = false)
```

```
Schema of Linear Regression dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- prediction: double (nullable = false)

Print Linear Regression dataframe
+--------------------+-----+------------------+
|            features|label|        prediction|
+--------------------+-----+------------------+
|(262144,[493,7698...|  4.0| 3.864791626125427|
|(262144,[4087,526...|  1.0|1.9184876920447194|
|(262144,[6524,715...|  1.0|1.5803268240273916|
|(262144,[2292,625...|  1.0| 0.954790074722017|
|(262144,[4758,825...|  1.0|1.0063165801686988|
|(262144,[12524,43...|  6.0| 5.090234803746513|
|(262144,[30483,76...|  4.0|3.9913273390483615|
|(262144,[37135,57...|  4.0|3.9871401619147653|
|(262144,[41070,54...|  5.0| 4.072119469897814|
|(262144,[493,2374...|  4.0|3.9393887732096964|
|(262144,[4731,110...|  3.0| 4.058254222111514|
|(262144,[12034,12...|  1.0|  3.07146167963999|
|(262144,[12524,19...|  4.0| 3.497616335245621|
|(262144,[61509,76...|  1.0|1.4363418623737396|
|(262144,[3616,263...|  1.0|2.0707440358521474|
|(262144,[3774,610...|  1.0|1.0075947891257706|
```

```
# https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.regression.LinearR
# https://stats.stackexchange.com/questions/142873/how-to-determine-the-accuracy-of-regression-which-measure-should-be-use

# Model's coefficients and intercept
print("Coefficients: " + str(LiRModel.coefficients))
print("Intercept: " + str(LiRModel.intercept))


# Summarize the model over the training set and print out some metrics
trainingSummary = LiRModel.summary
print("\n\nnumIterations: %d" % trainingSummary.totalIterations)
print("objectiveHistory: %s" % str(trainingSummary.objectiveHistory))
trainingSummary.residuals.show()
```

```
Coefficients: (262144,[2,5,21,39,60,76,79,80,86,93,111,116,132,156,161,164,181,185,209,216,247,248,251,266,274,276,286,310
Intercept: 3.703613616939355
```

```
numIterations: 98
objectiveHistory: [0.4998280013759888, 0.18346150434800726, 0.09688586524327557, 0.07130072570633779, 0.05154934442304637,
+--------------------+
|           residuals|
+--------------------+
|-0.00369400549909...|
|-0.00477740322736...|
|-0.01098309884043...|
|-0.01580913937105...|
|-0.00641246318881...|
|0.002143579028309528|
|-0.00602648731569...|
| 0.10708747331287949|
|-0.00562095338649...|
|0.025536772607845037|
|0.002420536108287852|
|-0.00394630385062...|
|-0.09432876043258531|
|  1.7495713893795708|
|   0.023501991878911|
|  0.6024569614457995|
|  1.7908592513932815|
```

```
print("Linear Regression Metrics")
print("meanSquaredError: %f" % (trainingSummary.meanSquaredError))
print("meanAbsoluteError: %f" % (trainingSummary.meanAbsoluteError))
print("RMSE: %f" % trainingSummary.rootMeanSquaredError)
print("r2: %f" % trainingSummary.r2)
```

```
Linear Regression Metrics
meanSquaredError: 0.198257
meanAbsoluteError: 0.105969
RMSE: 0.445260
r2: 0.949836
```

```
from pyspark.ml.regression import LinearRegression

# Classification example using Linear Regression Classifier
LiRModel_c2 = (
    LinearRegression()
    .setMaxIter(10000)
    .setRegParam(0.1)
    .setElasticNetParam(0.0)
    .fit(trainingData)
)

predictionsLiR_c2 = LiRModel_c2.transform(testData_c2)
print("Schema of Linear Regression dataframe")
predictionsLiR_c2.printSchema()

print("Print Linear Regression dataframe")
predictionsLiR_c2.show()
```

```
Schema of Linear Regression dataframe
root
 |-- features: vector (nullable = true)
 |-- label: double (nullable = true)
 |-- prediction: double (nullable = false)
```

```
Print Linear Regression dataframe
+--------------------+-----+------------------+
|            features|label|        prediction|
+--------------------+-----+------------------+
|(262144,[493,7698...|  0.0|3.8647916255932286|
|(262144,[4087,526...|  1.0|1.9184876924089738|
|(262144,[6524,715...|  1.0|1.5803268243875714|
|(262144,[2292,625...|  1.0|0.9547900746924358|
|(262144,[4758,825...|  1.0|1.0063165801371166|
|(262144,[12524,43...|  0.0| 5.090234804888531|
|(262144,[30483,76...|  0.0|3.9913273390335555|
|(262144,[37135,57...|  0.0|3.9871401619532456|
|(262144,[41070,54...|  0.0| 4.072119469645542|
|(262144,[493,2374...|  0.0| 3.939388773086295|
|(262144,[4731,110...|  0.0| 4.058254222130187|
|(262144,[12034,12...|  1.0|3.0714616799580083|
|(262144,[12524,19...|  0.0|3.4976163356807635|
|(262144,[61509,76...|  1.0|1.4363418625840487|
|(262144,[3616,263...|  1.0| 2.070744035406924|
|(262144,[3774,610...|  1.0|1.0075947894206165|
```

```
print("Linear Regression Metrics")
print("meanSquaredError: %f" % (trainingSummary_c2.meanSquaredError))
print("meanAbsoluteError: %f" % (trainingSummary_c2.meanAbsoluteError))
print("RMSE: %f" % trainingSummary_c2.rootMeanSquaredError)
print("r2: %f" % trainingSummary_c2.r2)
```

```
Linear Regression Metrics
meanSquaredError: 0.198257
meanAbsoluteError: 0.105969
RMSE: 0.445260
r2: 0.949836
```