ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

# Εφαρμογή τεχνικών μηχανικής μάθησης για την πρόβλεψη της κατανομής των επισκεπτών σε μεγάλες εκδηλώσεις, κάνοντας χρήση δεδομένων τοποθεσίας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αναστάσιος Μπερδελής

**Επιβλέπουσα :** Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Αθήνα, Ιανουάριος 2019

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗ Σ

# Εφαρμογή τεχνικών μηχανικής μάθησης για την πρόβλεψη της κατανομής του κόσμου σε μεγάλες εκδηλώσεις, κάνοντας χρήση δεδομένων τοποθεσίας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αναστάσιος Μπερδελής

**Επιβλέπουσα :** Θεοδώρα Βαρβαρίγου
Καθηγήτρια Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 14η Ιανουαρίου 2019.

| | | |
|---|---|---|
| ..... | ..... | ..... |
| Θ. Βαρβαρίγου | Σ. Παπαβασιλείου | Δ. Ασκούνης |
| Καθ. Ε.Μ.Π | Καθ. Ε.Μ.Π. | Καθ. Ε.Μ.Π. |

Αθήνα, Ιανουάριος 2019

.....
Αναστάσιος Δ. Μπερδελής

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

# ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

# 1. Εισαγωγή

Η επίλυση υπολογιστικών προβλημάτων μέσω αλγοριθμικών διαδικασιών αποτελεί βασικό κομμάτι της έρευνας, των επιστημών αλλά και της καθημερινότητας. Ένας αλγόριθμος είναι ουσιαστικά μία σειρά από εντολές οι οποίες με κατάλληλο συνδυασμό μετατρέπουν την είσοδο που δέχονται στην αντίστοιχη έξοδο, όπως απεικονίζεται στην Εικόνα 1.1.


Εικόνα 1.1. Μετατροπή εισόδου σε έξοδο
μέσω αλγοριθμικής διαδικασίας

Το πρόβλημα της ταξινόμησης ενός συνόλου τυχαίων αριθμών μπορεί να λυθεί εύκολα με την παραπάνω μέθοδο. Ο αλγόριθμος δέχεται σαν είσοδο τους αριθμούς που επιθυμούμε να ταξινομήσουμε και σαν έξοδο επιστρέφει τους αριθμούς αυτούς στη σωστή σειρά. Υπάρχουν στη διάθεσή μας διάφοροι αλγόριθμοι που επιλύουν προβλήματα ταξινόμησης και σε κάθε περίπτωση μπορούμε να διαλέξουμε αυτόν ο οποίος επιλύει βέλτιστα το πρόβλημά μας.

Ωστόσο, υπάρχουν προβλήματα για τα οποία δεν έχουν βρεθεί οι αντίστοιχοι αλγόριθμοι επίλυσης. Η πρόβλεψη της συμπεριφοράς των καταναλωτών, όπως και η αναγνώριση ανεπιθύμητης αλληλογραφίας είναι τέτοιου είδους προβλήματα. Για παράδειγμα στην περίπτωση της ανεπιθύμητης αλληλογραφίας, η είσοδος είναι ένα email και η έξοδος είναι μία ετικέτα ναι/όχι η οποία δείχνει εάν το περιεχόμενό του είναι ανεπιθύμητο ή όχι. Το τι θεωρείται όμως ανεπιθύμητο σε ένα email είναι σχετικό και αλλάζει από άνθρωπο σε άνθρωπο. Οπότε είναι δύσκολο να βρεθεί συγκεκριμένη μέθοδος που να μετατρέπει την είσοδο σε έξοδο.

Μπορεί λοιπόν να μην έχουμε την απαιτούμενη γνώση για να επιλύσουμε αλγοριθμικά τέτοια προβλήματα, έχουμε όμως χιλιάδες παραδείγματα έτσι ώστε ο υπολογιστής να καταφέρει να εξάγει ο ίδιος τον αλγόριθμο επίλυσής τους. Μπορούμε λοιπόν να χρησιμοποιήσουμε χιλιάδες email, κάποια από τα οποία είναι ανεπιθύμητα και κάποια όχι, έτσι ώστε με τη βοήθεια του υπολογιστή να βρούμε έναν αλγόριθμο ο οποίος θα επιλύει το πρόβλημα της ανεπιθύμητης αλληλογραφίας.

Σκοπός της μηχανικής μάθησης είναι να διαχειρίζεται τέτοιου είδους δεδομένα με στόχο την εξαγωγή αυτοματοποιημένων αλγοριθμικών διαδικασιών, οι οποίες να επιλύουν τα αντίστοιχα προβλήματα. Οι αλγόριθμοι βέβαια αυτοί μπορεί να μη λειτουργούν βέλτιστα σε όλες τις περιπτώσεις, αλλά μας δίνουν μία καλή και χρήσιμη εκτίμηση έτσι ώστε να κατανοήσουμε καλύτερα τη φύση των προβλημάτων, να εντοπίσουμε μοτίβα, και να κάνουμε προβλέψεις για μελλοντικά παρόμοια προβλήματα. Δεδομένου ότι τα δείγματα που θα έχουμε στο μέλλον δε θα είναι πολύ διαφορετικά από τα δεδομένα που έχουμε συλλέξει έως τώρα, οι προβλέψεις αυτές καθίστανται αρκετά ισχυρές.

Η εύρεση μοτίβων στις ανθρώπινες δραστηριότητες (pattern recognition), η πρόβλεψη των τιμών του χρηματιστηρίου (stock market prediction), η αναγνώριση ήχου και εικόνας (voice and image recognition) είναι μερικά από τα αμέτρητα προβλήματα που επιλύονται με τη χρήση της μηχανικής μάθησης.

Το σημαντικό όμως είναι ότι τα προβλήματα αυτά δε λύνονται μόνο βάσει στατιστικής μελέτης και παρατήρησης των αποτελεσμάτων. Τα προβλήματα αυτά ανήκουν στον ευρύτερο τομέα της τεχνητής νοημοσύνης. Ο υπολογιστής ουσιαστικά μαθαίνει από μόνος του τον τρόπο επίλυσής τους, και αυτό είναι που καθιστά τους αλγόριθμους που προκύπτουν ιδιαίτερα ισχυρούς.

Η εφαρμογή τεχνικών μηχανικής μάθησης σε μεγάλες βάσεις δεδομένων ονομάζεται εξόρυξη δεδομένων (data mining). Η λογική είναι ότι από την επεξεργασία ενός μεγάλου όγκου ακατέργαστων δεδομένων, κατασκευάζουμε απλούστερα μοντέλα, τα οποία περιέχουν τις απαραίτητες πληροφορίες για την επίλυση των προβλημάτων.

Για παράδειγμα, οι τράπεζες επεξεργάζονται τα δεδομένα που συλλέγουν, έτσι ώστε να μπορούν να ανιχνεύουν περιπτώσεις οικονομικής απάτης, καθώς και να προβλέπουν τις τιμές του χρηματιστηρίου. Στην ιατρική χρησιμοποιούνται μοντέλα για την ταχύτερη διάγνωση ασθενειών,ενώ στις τηλεπικοινωνίες αναλύονται μοτίβα κλήσεων για τη βελτιστοποίηση των δικτύων και την καλύτερη εξυπηρέτηση των πελατών. Στις διάφορες επιστήμες (φυσική, αστρονομία, βιολογία) τα δεδομένα που συλλέγονται βοηθούν τους επιστήμονες στην καλύτερη κατανόηση των φυσικών φαινομένων και στην έγκυρη πρόβλεψή τους. Είναι φανερό λοιπόν πως τεράστιοι όγκοι δεδομένων, αν αξιοποιηθούν σωστά, μπορούν να συμβάλλουν στην επίλυση σημαντικών προβλημάτων. [1]

# 2. Μηχανική Μάθηση - Τρόποι Μάθησης

## 2.1. Εισαγωγή

Σε όλες τις διεργασίες μηχανικής εκμάθησης στόχος είναι να βελτιώνεται κάθε φορά η απόδοσή του μοντέλου, έτσι ώστε να καταφέρει να επιλύει μελλοντικές διεργασίες. Οι τεχνικές που χρησιμοποιούμε για να επιτευχθεί αυτό στηρίζονται σε τέσσερις βασικούς παράγοντες:

- Ποιά κομμάτια/συστατικά του μοντέλου χρειάζονται βελτίωση.
- Τις έως τώρα γνώσεις που έχουμε.
- Με ποιον τρόπο αναπαρίστανται τα δεδομένα του προβλήματος, καθώς και τα συστατικά του μοντέλου.
- Με ποιον τρόπο γίνεται η αναπληροφόρηση (feedback), έτσι ώστε το μοντέλο να εντοπίσει τα λάθη του και "να μάθει από αυτά".

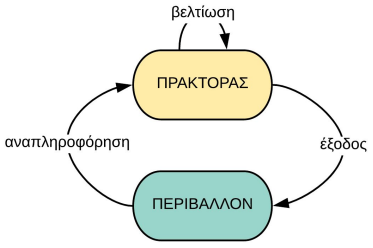
Βλέπουμε λοιπόν ότι μέσα από την αναπληροφόρηση το μοντέλο μαθαίνει και βελτιώνει κάθε φορά την απόδοσή του. Υπάρχουν τέσσερις τρόποι με τους οποίους γίνεται η αναπληροφόρηση, οι οποίοι και καθορίζουν τους τέσσερις βασικούς τρόπους μάθησης:

- **Μη επιτηρούμενη μάθηση (unsupervised learning)**
  Σε αυτήν την περίπτωση το μοντέλο αναγνωρίζει μοτίβα από μόνο του χωρίς να έχει σαφή αναπληροφόρηση. Το βασικότερο παράδειγμα μη επιτηρούμενης μάθησης είναι οι τεχνικές συσταδοποίησης (clustering), ο εντοπισμός δηλαδή πιθανών υποσυνόλων όταν έχουμε σαν είσοδο ένα ευρύτερο σύνολο παρατηρήσεων. Για παράδειγμα τα συστήματα συστάσεων (recommendation systems) χρησιμοποιούν τεχνικές συσταδοποίησης, αξιοποιώντας τις προτιμήσεις των καταναλωτών για την καλύτερη εξυπηρέτησή τους.

- **Ενισχυτική μάθηση (reinforcement learning)**
  Στην ενισχυτική μάθηση έχουμε τον πράκτορα και το περιβάλλον. Ο πράκτορας αλληλεπιδρά με το περιβάλλον και μαθαίνει μέσα από μία διαδικασία επιβράβευσης-ποινών (Εικόνα 1.2). Τέτοια μοντέλα χρησιμοποιούνται στην εκπαίδευση του υπολογιστή στα διάφορα ηλεκτρονικά παιχνίδια, έτσι ώστε αυτός σε κάθε περίπτωση να παίρνει τη σωστή απόφαση. Για παράδειγμα, σε μία παρτίδα σκάκι ο υπολογιστής μετά από κάθε νίκη επιβραβεύεται με πόντους, ενώ μετά από κάθε ήττα βαθμολογείται αρνητικά. Ωστόσο δεν επιβραβεύονται μεμονωμένα οι αποφάσεις του και εναπόκειται σε αυτόν να επιλέξει ποιες κινήσεις ήταν υπεύθυνες για τη νίκη και πρέπει να τις επαναλάβει στα επόμενα παιχνίδια.

- **Επιτηρούμενη μάθηση (supervised learning)**
  Σε αυτήν την κατηγορία το μοντέλο παρατηρεί ζευγάρια εισόδου-εξόδου και εξάγει συναρτήσεις/αλγοριθμικές διαδικασίες που αντιστοιχίζουν νέες εισόδους στις αντίστοιχες εξόδους. Το πρόβλημα του εντοπισμού ανεπιθύμητης αλληλογραφίας που αναφέρεται παραπάνω είναι παράδειγμα προβλήματος επιτηρούμενης μάθησης. Στην περίπτωση αυτή η είσοδος είναι η αλληλογραφία και η έξοδος το αν αυτή είναι ανεπιθύμητη ή όχι.

- **Ημι-επιτηρούμενη μάθηση (semi-supervised learning)**
  Στην ημι-επιτηρούμενη μάθηση έχουμε ορισμένα παραδείγματα για τα οποία είναι γνωστή η έξοδος και καλούμαστε να εργαστούμε πάνω σε ένα μεγάλο όγκο δεδομένων για τον οποίο η έξοδος δεν είναι γνωστή. Για παράδειγμα έστω ότι θέλουμε να κατηγοριοποιήσουμε τις ιστοσελίδες βάσει του περιεχομένου τους σε ενημερωτικές, εκπαιδευτικές και αθλητικές. Το να βάλουμε ετικέτα οι ίδιοι σε κάθε μία ιστοσελίδα ξεχωριστά θα ήταν αρκετά χρονοβόρο, δεδομένου του αριθμού των ιστοσελίδων που υπάρχουν. Έτσι χρησιμοποιούμε ένα μικρό σύνολο και αντιστοιχίζουμε κάθε στοιχείο του σε μία από τις κατηγορίες. Το μοντέλο μαθαίνει από μόνο του ποια είναι τα χαρακτηριστικά που είναι υπεύθυνα για την κατηγοριοποίηση, ενώ παράλληλα μπορεί να δημιουργήσει καινούριες κατηγορίες βάσει των παρατηρήσεών του. [2]



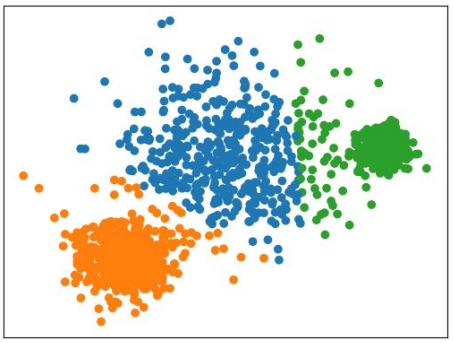

Εικόνα 1.2. Αλληλεπίδραση πράκτορα - περιβάλλοντος

## 2.2. Μη Επιτηρούμενη Μάθηση

Σε αντίθεση με τις άλλες τεχνικές μάθησης, στη μη επιτηρούμενη μάθηση (unsupervised learning), σκοπός των μοντέλων είναι να εξάγουν εξ' ολοκλήρου από μόνα του μία αλγοριθμική διαδικασία έτσι ώστε να αντιστοιχίσουν τις εισόδους που τους δίνονται στην έξοδο. Ουσιαστικά έχουμε ένα σύνολο από δεδομένα εισόδου, χωρίς να γνωρίζουμε κάτι για την έξοδο. Κατά συνέπεια δεν υπάρχει κάποιος απόλυτος τρόπος για να αξιολογήσουμε τα αποτελέσματα, αλλά μπορούμε να αντλήσουμε σημαντικές πληροφορίες παρατηρώντας τα. Σε αυτήν την κατηγορία χρησιμοποιούνται αρκετά γνωστά υπολογιστικά μοντέλα, όπως για παράδειγμα νευρωνικά δίκτυα, μαρκοβιανές αλυσίδες και τεχνικές συσταδοποίησης, τα οποία αναλύονται εκτενέστερα παρακάτω.

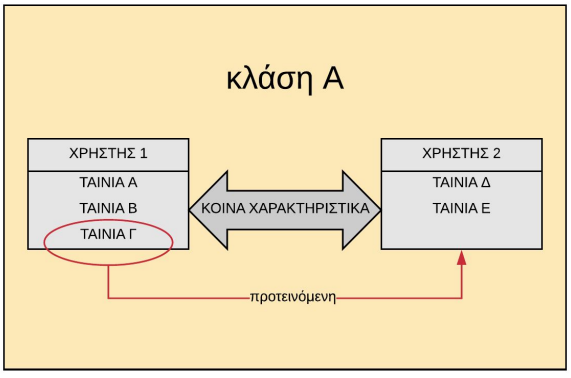
### 2.2.1. Συσταδοποίηση

Η τεχνική της συσταδοποίησης (clustering) έχει ως στόχο το διαχωρισμό των δεδομένων σε κλάσεις (υποομάδες), έτσι ώστε τα δεδομένα που βρίσκονται σε κάθε κλάση να έχουν περισσότερα κοινά μεταξύ τους, συγκριτικά με τα δεδομένα διαφορετικών υποομάδων.



Εικόνα 2.1. Συσταδοποίηση δεδομένων σε τρεις κλάσεις,
βάσει του αλγορίθμου K-means

Έστω ότι στο παράδειγμα με τα συστήματα συστάσεων που αναφέρθηκε προηγουμένως έχουμε έναν όγκο δεδομένων με χρήστες μιας ιστοσελίδας και τις αγαπημένες τους ταινίες. Στόχος μας είναι να προτείνουμε στους χρήστες αυτούς νέες ταινίες που πιθανόν να τους ενδιαφέρουν. Οι αλγόριθμοι συσταδοποίησης προσπαθούν να βρουν κοινά χαρακτηριστικά μεταξύ των χρηστών και να τους χωρίσουν σε κλάσεις βάσει των προτιμήσεών τους. Για παράδειγμα στην Εικόνα 2.1 έχουμε χωρίσει τους χρήστες σε τρεις κλάσεις βάσει των αγαπημένων τους ταινιών. Αυτό φυσικά δε σημαίνει ότι σε κάθε κλάση όλοι οι χρήστες έχουν δει τις ίδιες ακριβώς ταινίες, αλλά ότι υπάρχουν πάρα πολλά κοινά μεταξύ των ταινιών που έχουν δει. Για παράδειγμα στην πορτοκαλί κλάση θα μπορούσαν να βρίσκονται οι χρήστες που έχουν δει πολλές ταινίες τους ηθοποιού Α, στην μπλε κλάση οι χρήστες που βλέπουν αρκετές ταινίες θρίλερ και στην πράσινη κλάση οι χρήστες που βλέπουν πολλά ντοκιμαντέρ. Αυτό όμως δεν σημαίνει ότι όσοι βρίσκονται στην πορτοκαλί κλάση βλέπουν μόνο ταινίες του ηθοποιού Α, ή ότι όσοι βρίσκονται στην πράσινη κλάση βλέπουν μόνο ντοκιμαντέρ. Ο αλγόριθμος αντιλαμβάνεται ότι μεταξύ όλων των χρηστών, υπάρχουν κάποιοι που έχουν περισσότερα κοινά μεταξύ τους, παρά απ' ότι με τους υπόλοιπους. Με τον τρόπο αυτόν μπορούμε να προτείνουμε στους χρήστες της πορτοκαλί κλάσης ταινίες που έχουν δει άλλοι χρήστες της ίδιας κλάσης και ίσως τους ενδιαφέρουν.




Εικόνα 2.2. Παράδειγμα τρόπου λειτουργίας συστηματων
σύστασης ταινιών μέσα σε μία κλάση

Στην Εικόνα 2.2 φαίνεται πως στην κλάση Α υπάρχουν δύο χρήστες, που παρόλο που δεν έχουν δει καμία κοινή ταινία, ο αλγόριθμος έχει εντοπίσει ότι οι ταινίες που έχουν δει έχουν πολλά κοινά χαρακτηριστικά μεταξύ τους (π.χ. ηθοποιοί, είδος ταινίας, σκηνοθέτης). Το ιδιαίτερα σημαντικό στη

διαδικασία αυτή είναι ότι ο άνθρωπος δεν επεμβαίνει για να υποδείξει στο μοντέλο με ποιον τρόπο θα ταξινομήσει τους χρήστες και γι' αυτό οι αλγόριθμοι συσταδοποίησης ανήκουν στην κατηγορία της μη επιτηρούμενης μάθησης. [3]

### 2.2.1.1. Αλγόριθμος K-means

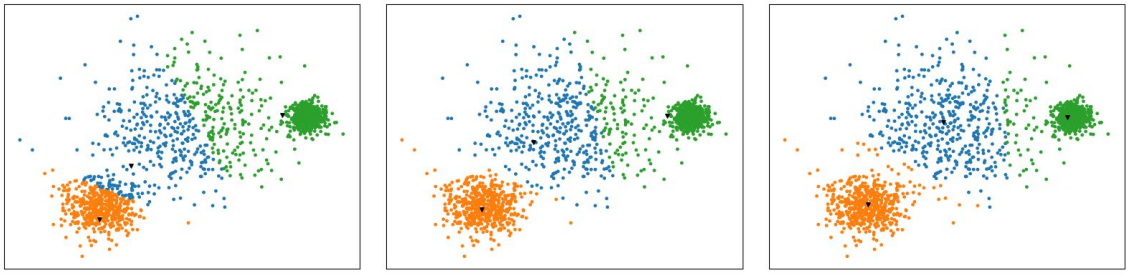
Έστω ότι έχουμε ένα σύνολο παρατηρήσεων $X = \{x_i\},\ i = 1, 2, ...., n$ και θέλουμε να το χωρίσουμε σε K υποομάδες (clusters), $C = \{c_k,\ k = 1, 2, ...., K\}$. Ο αλγόριθμος K-means εντοπίζει τις υποομάδες έτσι ώστε το τετραγωνικό σφάλμα μεταξύ της εμπειρικής μέσης τιμής (αντιπροσωπευτική μέση τιμή) της υποομάδας και της τιμής των παρατηρήσεων να ελαχιστοποιείται. Το τετραγωνικό σφάλμα μεταξύ της μέσης τιμής $\mu_k$ και των παρατηρήσεων $c_k$ μιας συστάδας ορίζεται από την παρακάτω σχέση.

$$J(c_k) = \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \quad (2.1)$$

Στόχος λοιπόν του αλγορίθμου K-means είναι η ελαχιστοποίηση του αθροίσματος των τετραγωνικών σφαλμάτων όλων των K υποομάδων,

$$J(C) = \sum_{k=1}^{K} \sum_{x_i \in c_k} \|x_i - \mu_k\|^2 \quad (2.2)$$

Η ελαχιστοποίηση της παραπάνω συνάρτησης είναι πρόβλημα δυσκολίας NP-hard, ακόμα και για $K = 2$. Γι' αυτό το λόγο θεωρούμε πως ο αλγόριθμος K-means συγκλίνει σε ένα τοπικό ελάχιστο, παρόλο που σύγχρονες μελέτες αποδεικνύουν ότι όταν υπάρχει σαφής διαχωρισμός μεταξύ των υποομάδων, ο αλγόριθμος μπορεί να συγκλίνει στη βέλτιστη-ελάχιστη τιμή της παραπάνω σχέσης. [4]



Εικόνα 2.3. Παράδειγμα σύγκλισης του αλγορίθμου
K-means μετά από 1,2 και 7 επαναλήψεις

Στην Εικόνα 2.3 παρατηρούμε πως συγκλίνει ο αλγόριθμος K-means για $K = 3$ μετά από επτά επαναλήψεις. Στην πρώτη επανάληψη επιλέγονται τυχαία τρία σημεία-κέντρα (αντιπροσωπευτικά του κάθε cluster) και κάθε παρατήρηση αντιστοιχίζεται σε ένα από αυτά βάσει της κοντινότερης απόστασης. Στη δεύτερη επανάληψη τα κέντρα ανανεώνονται έτσι ώστε να ελαχιστοποιείται η τιμή $J(C)$ του σφάλματος όπως αναφέρεται παραπάνω. Μετά από επτά επαναλήψεις ο αλγόριθμος έχει πλέον συγκλίνει και οι παρατηρήσεις μας έχουν ομαδοποιηθεί έτσι ώστε η τιμή του $J(C)$ να είναι πλέον η ελάχιστη δυνατή.

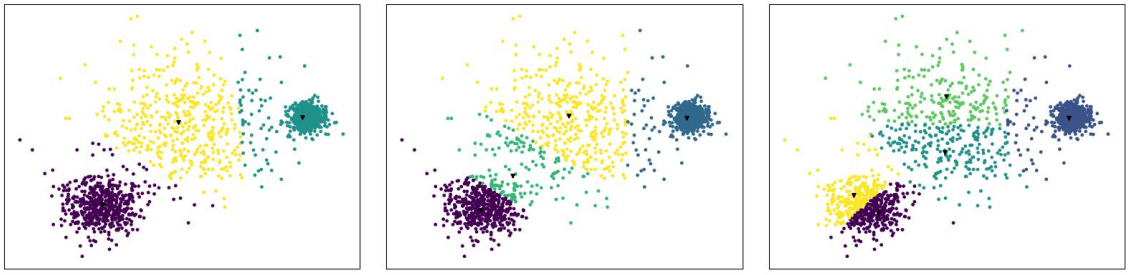
Στο παραπάνω παράδειγμα επιλέξαμε εκ των προτέρων τον αριθμό των clusters σε $K = 3$, δηλώνοντας έτσι ότι επιθυμούμε να χωρίσουμε τις παρατηρήσεις μας σε τρεις ομάδες. Η επιλογή του σωστού αριθμού των clusters είναι πολύ σημαντική αλλά και αρκετά περίπλοκη. Παρακάτω αναφέρονται κάποιες βασικές μέθοδοι, οι οποίες μας βοηθούν να εκτιμήσουμε ποιος είναι ο βέλτιστος αριθμός συστάδων σε κάθε πρόβλημα [5].

**A. Εμπειρικός κανόνας (Rule of thumb)**

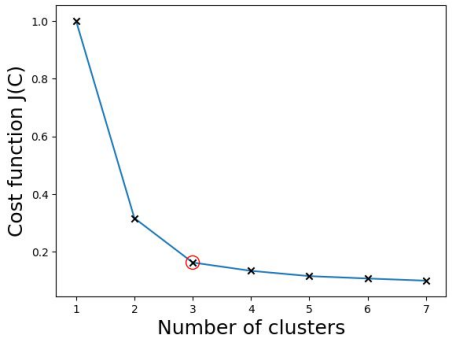
Είναι η πιο απλή μέθοδος και μπορεί να χρησιμοποιηθεί σε οποιαδήποτε δεδομένα. Ο αριθμός των clusters υπολογίζεται από τη σχέση $k \approx \sqrt{n/2}$, όπου $n$ είναι ο αριθμός των παρατηρήσεων.

**B. Μέθοδος του αγκώνα (Elbow method)**

Είναι η πιο διαδεδομένη μέθοδος για τον εντοπισμό του αριθμού των clusters και στηρίζεται στην παρατήρηση. Ξεκινάμε με $C = 2$ και αυξάνουμε τον αριθμό κατά ένα κάθε φορά, υπολογίζοντας τα clusters καθώς και την τιμή $J(C)$ του αθροίσματος των τετραγωνικών σφαλμάτων. Η λογική είναι πως όσο αυξάνεται ο αριθμός $C$ των clusters, η τιμή του $J(C)$ θα μειώνεται, καθώς όπως είναι φυσιολογικό στην περίπτωση που έχουμε πάρα πολλά clusters, η τιμή του σφάλματος σε κάθε cluster είναι πάρα πολύ μικρή (η τιμή κάθε παρατήρησης προσεγγίζει την αντιπροσωπευτική μέση τιμή της υποομάδας, σε κάθε υποομάδα). Όπως βλέπουμε όμως στην Εικόνα 2.5 παρόλο που η τιμή της συνάρτησης κόστους μειώνεται συνεχώς, ο ρυθμός με τον οποίο μειώνεται δεν είναι σταθερός. Ενώ δηλαδή παρατηρείται μεγάλη μείωση για τις μεταβολές από $C = 1$ σε $C = 2$, και από $C = 2$ σε $C = 3$, βλέπουμε ότι όσο αυξάνουμε τον αριθμό των clusters από $C = 4$ και έπειτα, δεν πετυχαίνουμε μεγάλες μεταβολές στην τιμή της $J(C)$. Η λογική πίσω από αυτό είναι ότι μετά από κάποια τιμή του $C$ η διακύμανση των τιμών των παρατηρήσεων σε κάθε cluster είναι ιδιαίτερα μικρή και κατ' επέκταση υπάρχουν πολλά clusters τα οποία βρίσκονται κοντά το ένα στο άλλο. Στην Εικόνα 2.5 παρατηρούμε πώς το σύνολο των παρατηρήσεών μας χωρίζεται σε συστάδες, βάσει του αριθμού $C$, που έχουμε επιλέξει. Διακρίνουμε ότι όσο αυξάνεται ο αριθμός των συστάδων, τα δεδομένα κάθε υποομάδας συνεχίζουν να ομαδοποιούνται με τέτοιο τρόπο, έτσι ώστε να ελαχιστοποιείται η συνάρτηση $J(C)$. Παρόλα αυτά, με τη μέθοδο του αγκώνα έχουμε υπολογίσει ότι ο βέλτιστος αριθμός συστάδων για το πρόβλημά μας είναι $C = 3$, όπως φαίνεται στην Εικόνα 2.6. Από τις εικόνες διακρίνουμε ότι ενώ για $C = 3$ το μωβ cluster φαίνεται να αποτελείται από αρκετά παρόμοια σημεία, για $C = 5$ πλέον τα σημεία αυτά (παρόλο που συνεχίζουν να έχουν πολλά κοινά μεταξύ τους) ανήκουν σε δύο διαφορετικά clusters.

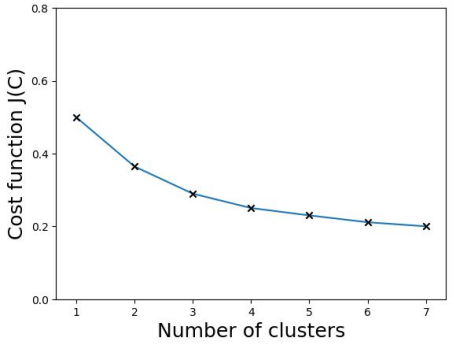


Εικόνα 2.4. Χρήση του αλγορίθμου K-means για C=3, C=4 και C=5




Εικόνα 2.5. Εντοπισμός του βέλτιστου αριθμού των clusters
με τη μέθοδο του αγκώνα

Η μέθοδος του αγκώνα χρησιμοποιείται ως ενδεικτική μέθοδος και δεν αποτελεί πάντα λύση στο πρόβλημα του εντοπισμού των αριθμό των clusters. Υπάρχουν περιπτώσεις όπου δεν είναι εύκολο να εντοπιστεί ο "αγκώνας" της καμπύλης διότι είτε δεν υπάρχει, είτε υπάρχουν περισσότεροι του ενός (Εικόνα 2.7).




Εικόνα 2.6. Παράδειγμα καμπύλης όπου είναι δύσκολος ο εντοπισμός
του βέλτιστου αριθμού των clusters

**Γ. Κριτήριο προσέγγισης της πληροφορίας (Information Criterion Approach)**

Σύμφωνα με τη μέθοδο αυτή, έχουμε ένα σύνολο από διαφορετικά μοντέλα (για το ίδιο πρόβλημα) και προσπαθούμε να εκτιμήσουμε την απόδοση καθενός εξ αυτών, σε σχέση με τα υπόλοιπα. Βασικό εργαλείο της μεθόδου αυτής είναι μία συνάρτηση πιθανοφάνειας (likelihood function), η οποία μας δείχνει κατά πόσο η τιμή μιας παραμέτρου είναι ικανοποιητική δεδομένων των παρατηρήσεών μας. Όσο αυξάνεται ο αριθμός των clusters σε ένα μοντέλο, αυξάνεται επίσης και η τιμή της παραπάνω συνάρτησης. Αν καλούμασταν για παράδειγμα να κατασκευάσουμε το μοντέλο με τις πιο ικανοποιητικές παραμέτρους για κάποιο πρόβλημα, έχοντας τη δυνατότητα να χρησιμοποιήσουμε όσα clusters επιθυμούμε, το μοντέλο αυτό θα αποτελούνταν από τόσα clusters όσες και οι παρατηρήσεις μας. Έτσι κάθε παρατήρηση θα ήταν αντιπροσωπευτική του cluster της και το τετραγωνικό σφάλμα θα ήταν ίσο με μηδέν. Αυτό φυσικά δε μας βοηθάει σε κάτι και δεν επιλύει το πρόβλημά μας. Στόχος λοιπόν της μεθόδου αυτής είναι να συνδυάσει τη συνάρτηση πιθανοφάνειας και τον αριθμό των παραμέτρων του μοντέλου, έτσι ώστε όσο αυξάνεται ο αριθμός των παραμέτρων να υπάρχει αφενός επιβράβευση (λόγω αύξησης της πιθανοφάνειας), αφετέρου ποινή. Δύο πολύ γνωστά κριτήρια είναι το κριτήριο του Akaike (Akaike Information Criterion) και το Μπεϋζιανό κριτήριο πληροφορίας (Bayesian Information Criterion). Παρακάτω, φαίνονται οι σχέσεις που ορίζουν τα δύο αυτά κριτήρια.

$$AIC = -2 \cdot ln(\widehat{L}) + 2 \cdot k \quad (2.2)$$
$$BIC = -2 \cdot ln(\widehat{L}) + ln(N) \cdot k \quad (2.3)$$

Όπου $\widehat{L}$ η μέγιστη τιμή της συνάρτησης πιθανοφάνειας του μοντέλου, $k$ ο αριθμός των εκτιμώμενων παραμέτρων και $N$ ο αριθμός των παρατηρήσεων.

**Δ. Επιλογή του αριθμού των clusters με τη χρήση του περιγράμματος (Silhouette)**

Στη μέθοδο αυτή χρησιμοποιούνται ορισμένοι δείκτες οι οποίοι συγκρίνουν τις αποστάσεις μεταξύ των σημείων εντός του ίδιου cluster, με τις αποστάσεις μεταξύ σημείων διαφορετικών clusters. Όσο μεγαλύτερη είναι η διαφορά, τόσο καλύτερα έχει γίνει η ομαδοποίηση των παρατηρήσεων. Ένας αρκετά ισορροπημένος συντελεστής, αναφορικά με την απόσταση μεταξύ των παρατηρήσεων, είναι το περίγραμμα πλάτους. Σύμφωνα με αυτό, συσχετίζεται η πυκνότητα εντός των clusters και η αποστασιοποίησή τους από τα υπόλοιπα. Η τιμή του περιγράμματος πλάτους $s(i)$ για κάθε παρατήρηση $i$ δίνεται από τη σχέση 2.4.

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))} \quad (2.4)$$

Όπου $a(i)$ η μέση απόσταση μεταξύ του σημείου $i$ και όλων των άλλων σημείων του cluster στο οποίο βρίσκεται το $i$, και $b(i)$ η ελάχιστη τιμή μεταξύ των μέσων αποστάσεων του σημείου $i$ από όλα τα άλλα σημεία των υπόλοιπων clusters. Το περίγραμμα πλάτους παίρνει τιμές μεταξύ $-1$ και $1$. Όταν η τιμή βρίσκεται κοντά στο μηδέν, σημαίνει ότι το σημείο θα μπορούσε να ενταχθεί και σε διαφορετικό cluster. Όταν η τιμή βρίσκεται κοντά στο $-1$ σημαίνει ότι το σημείο έχει ενταχθεί σε λάθος ομάδα, ενώ στην περίπτωση που όλες οι τιμές του περιγράμματος πλάτους βρίσκονται κοντά στο $1$ συμπεραίνουμε ότι έχει γίνει πολύ καλή ομαδοποίηση των παρατηρήσεών μας.

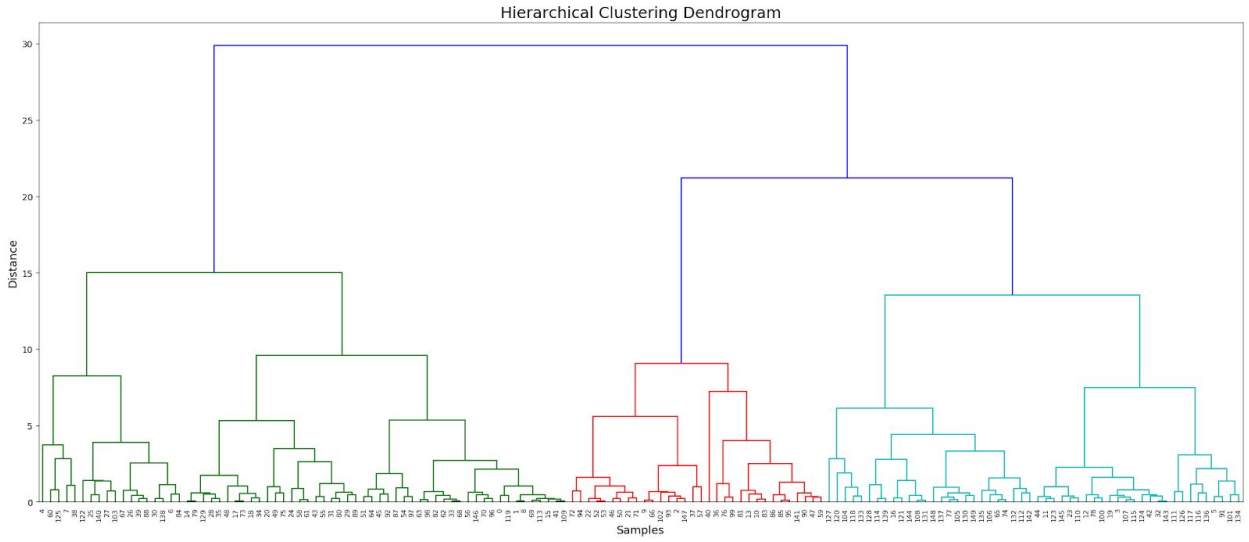
**Ε. Διασταύρωση-επικύρωση (Cross-validation)**

Οι R. Tibshirani και G. Walther [6], πρότειναν το 2005 μία διαφορετική προσέγγιση στον τρόπο με τον οποίο βρίσκουμε τον βέλτιστο αριθμό των clusters σε ένα πρόβλημα. Σύμφωνα με την προσέγγιση αυτή, το πρόβλημα της εύρεσης του αριθμού των clusters αντιμετωπίζεται σαν model selection πρόβλημα. Η μέθοδος αυτή είναι ευρέως γνωστή σε προβλήματα ταξινόμησης (classification), όπου η επιλογή του μοντέλου γίνεται βάσει της μείωσης του σφάλματος πρόβλεψης. Τα προβλήματα αυτά αναλύονται εκτενέστερα σε επόμενα κεφάλαια. Σύμφωνα λοιπόν με τη μέθοδο του cross validation, χωρίζουμε τα δεδομένα μας σε δύο ή και περισσότερα κομμάτια. Ένα κομμάτι χρησιμοποιείται για την ομαδοποίηση των δεδομένων (clustering), και τα υπόλοιπα για την επικύρωση, δηλαδή αξιολόγηση των αποτελεσμάτων. Η διαδικασία αυτή επαναλαμβάνεται περισσότερες φορές, επιλέγοντας κάθε φορά τυχαία τμήματα των δεδομένων μας για την ομαδοποίηση και την αξιολόγηση αντίστοιχα, έτσι ώστε να αντλήσουμε σημαντικά συμπεράσματα για τη σταθερότητα και την απόδοση του μοντέλου μας.

### 2.2.1.2. Μέθοδος Ιεραρχικής Συσταδοποίησης

Οι αλγόριθμοι ιεραρχικής συσταδοποιησης (hierarchical clustering) επιλύουν σε ένα βαθμό το

πρόβλημα εύρεσης του σωστού αριθμού των clusters που παρατηρήθηκε προηγουμένως. Οι T.Calinski και J. Harabasz [7] στηριζόμενοι στο γράφο-δέντρο ελαχίστων αποστάσεων του Florek [8] πρότειναν το 1974 έναν τρόπο με τον οποίο τα δεδομένα χωρίζονται σταδιακά σε ομάδες. Σύμφωνα με την πρότασή τους, θεωρούμε ότι τα δεδομένα μας ανήκουν σε ένα ευρύτερο cluster και σταδιακά διαιρούνται σε μικρότερα. Στην Εικόνα 2.7 γίνεται φανερό πώς από ένα ευρύτερο cluster σταδιακά καταλήγουμε σε μικρότερα, και στο τέλος έχουμε απλωμένα τα αρχικά μας δείγματα.




Εικόνα 2.7. Χωρισμός παρατηρήσεων σε clusters με τη μέθοδο
της ιεραρχικής ταξινόμησης

Στην ίδια εικόνα μπορούμε να διατυπώσουμε την παραπάνω παρατήρηση διαφορετικά, και να επικεντρωθούμε στην "προς τα πάνω" ομαδοποίηση των δεδομένων. Ουσιαστικά τα δεδομένα ομαδοποιούνται σιγά σιγά σε clusters, καταλήγοντας στο τέλος να ανήκουν όλα σε μία ευρύτερη ομάδα. Οι δύο αυτές προσεγγίσεις του προβλήματος ορίζονται ως εξής:

*Αθροιστική ομαδοποίηση (agglomerative clustering)*
Κάθε παρατήρηση αποτελεί ένα cluster. Βάσει των μεταξύ τους αποστάσεων, τα clusters σταδιακά ενώνονται, δημιουργώντας καινούριες ομάδες.

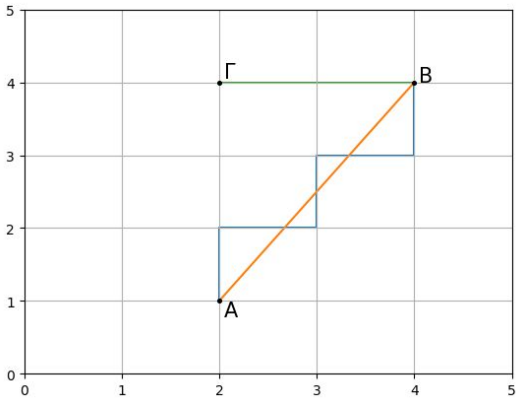
*Διαιρετική ομαδοποίηση (divisive clustering)*
Όλες οι παρατηρήσεις ανήκουν σε ένα ευρύτερο cluster. Σταδιακά, το cluster αυτό διαιρείται σε μικρότερες ομάδες. Η μέθοδος επαναλαμβάνεται έως ότου όλες οι παρατηρήσεις να αποτελούν ξεχωριστά clusters.

Σε κάθε περίπτωση, έπειτα από επισκόπηση των ομαδοποιημένων παρατηρήσεων (μετακινούμαστε στην ιεραρχία, αλλάζοντας κάθε φορά επίπεδο) μπορούμε να διακρίνουμε οι ίδιοι ποιος είναι ο βέλτιστος αριθμός των clusters για το πρόβλημά μας.

Όπως είδαμε λοιπόν οι αλγόριθμοι ιεραρχικής συσταδοποίησης ομαδοποιούν τις παρατηρήσεις (και στη συνέχεια τις υποομάδες που δημιουργήθηκαν) βάσει των μεταξύ τους αποστάσεων. Κάποιοι από τους κανόνες που χρησιμοποιούνται αναφέρονται παρακάτω.

*Απόσταση μεταξύ των παρατηρήσεων*

Βάσει της μετρικής αυτής, υπολογίζονται οι αποστάσεις μεταξύ των παρατηρήσεων, και στη συνέχεια προχωράμε στη διαδικασία της ομαδοποίησης.




Εικόνα 2.8. Σύγκριση ευκλείδειας απόστασης, με απόσταση Manhattan

Στην Εικόνα 2.8 εύκολα υπολογίζουμε ότι $\Gamma B = 2$. Η απόσταση όμως AB μπορεί να είναι είτε ίση μ $\sqrt{13}$, σύμφωνα με την ευκλείδεια απόσταση, είτε ίση με 5, σύμφωνα με την απόσταση Manhattan. Παρατηρούμε λοιπόν ότι η επιλογή της κατάλληλης μετρικής επηρεάζει σε μεγάλο βαθμό την ομαδοποίηση των παρατηρήσεών μας. Κάποιες βασικές μετρικές αναφέρονται στον Πίνακα 2.1.

| | |
|---|---|
| Ευκλείδεια απόσταση | $\|A-B\|_2=\sqrt{\sum_i \left(A_i-B_i\right)^2}$ |
| Τετραγωνική ευκλείδεια απόσταση | $\|A-B\|_2^2=\sum_i \left(A_i-B_i\right)^2$ |
| Απόσταση Manhattan | $\|A-B\|_1=\sum_i \|A_i-B_i\|$ |
| Απόσταση βάσει του συνημιτόνου | $\cos(\theta)=\frac{A\cdot B}{\|A\|\cdot\|B\|}$ |

Πινακας 2.1. Μετρικές υπολογισμού της απόστασης μεταξύ των παρατηρήσεων

*Κριτήρια ένωσης (linkage criteria)*

Τα κριτήρια αυτά απεικονίζουν με μορφή συνάρτησης την απόσταση μεταξύ δύο διαφορετικών ομάδων σε κάθε βήμα. Έτσι καθορίζεται πως οι υποομάδες είτε θα δημιουργήσουν μία ευρύτερη ομάδα (agglomerative clustering), είτε θα διασπαστούν σε μικρότερα clusters (divisive clustering). Στον Πίνακα 2.2 ορίζονται κάποια βασικά κριτήρια ένωσης.

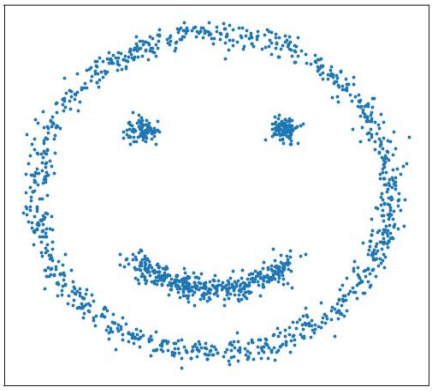
| | |
|---|---|
| Single linkage | $min\{d(a,b):a\in A,b\in B\}$ |
| Complete linkage | $max\{d(a,b):a\in A,b\in B\}$ |
| UPGMA | $\frac{1}{\|A-B\|}\cdot\sum_{a\in A}\sum_{b\in B}d(a,b)$ |

Πινακας 2.2. Κριτήρια ένωσης για ομαδοποίηση των clusters

Ένα ακόμα βασικό κριτήριο ένωσης είναι το κριτήριο του Ward, [9]. Σύμφωνα με αυτό, η απόσταση μεταξύ δύο clusters, A και B, καθορίζεται από την αύξηση στην τιμή του τετραγωνικού σφάλματος μετά την ένωσή τους. Στόχος του κριτηρίου είναι μετά την ένωση να ελαχιστοποιείται η τιμή αυτή.

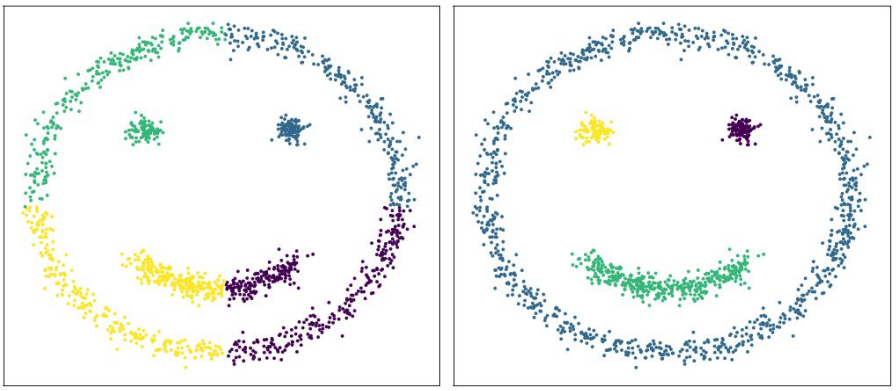
### 2.2.1.3. Αλγόριθμος DBSCAN

Όπως είδαμε παραπάνω, ο αλγόριθμος k-means και οι αλγόριθμοι ιεραρχικής συσταδοποίησης χρησιμοποιούν τις αποστάσεις μεταξύ των σημείων για να προχωρήσουν στην ομαδοποίησή τους. Το 1996 προτάθηκε ένας νέος τρόπος για να γίνεται η ομαδοποίηση των δεδομένων σε clusters [10], αυτή τη φορά βάσει της επί μέρους πυκνότητας των παρατηρήσεων. Έστω ότι έχουμε ένα σύνολο παρατηρήσεων τα οποία σχηματίζουν ένα πρόσωπο, Εικόνα 2.9. Στη συνέχεια εφαρμόζουμε τον αλγόριθμο k-means που είδαμε προηγουμένως για να χωρίσουμε τις παρατηρήσεις μας σε clusters. Επιλέγουμε $C = 4$ και παρατηρούμε τα αποτελέσματα.



Εικόνα 2.9. Απεικόνιση ενός συνόλου παρατηρήσεων
που σχηματίζουν εικόνα προσώπου

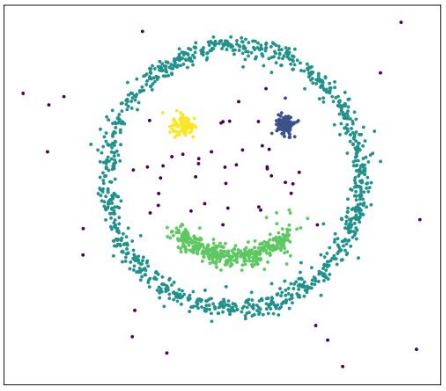
Βλέπουμε λοιπόν ότι ο αλγόριθμος k-means χώρισε τα αποτελέσματα σε τέσσερις ομάδες, βάσει της ευκλείδειας απόστασης μεταξύ των σημείων του, Εικόνα 2.10 (α). Θεωρητικά έχουμε ένα πολύ καλό αποτέλεσμα, έχοντας απόλυτη γεωγραφική ισορροπία ανάμεσα στα clusters. Είναι όμως ο διαχωρισμός αυτός πρακτικά αποτελεσματικός; Το ερώτημα αυτό απαντάται στην Εικόνα 2.10 (β). Παρατηρούμε τη διαφορετική προσέγγιση που ακολουθεί ο αλγόριθμος DBSCAN και το μέγεθος της σημασίας της. Σύμφωνα με αυτόν, χωρίς να παρέμβουμε επιλέγοντας τον αριθμό τον clusters, εντοπίζονται τέσσερις διαφορετικές ομάδες βάσει της πυκνότητας των σημείων του.



(α) (β)

Εικόνα 2.10. Χρήση αλγορίθμων k-means (α) και DBSCAN (β)
για την ομαδοποίηση των δεδομένων σε clusters

Στον αλγόριθμο αυτό επιλέγουμε ποια είναι η μέγιστη απόσταση μεταξύ δύο σημείων έτσι ώστε αυτά να θεωρηθούν στην ίδια ομάδα, καθώς και τον ελάχιστο αριθμό δειγμάτων που απαιτούνται έτσι ώστε ένα σημείο να θεωρηθεί πυρήνας (core point). Στην Εικόνα 2.10 (β) παρατηρούμε ότι έχουν εντοπιστεί τέσσερα σημεία πυρήνες, το περίγραμμα του προσώπου, τα δύο μάτια και το χαμόγελο, βάσει των αποστάσεων των επιμέρους παρατηρήσεων. Ένας ακόμα λόγος για τον οποίο είναι σημαντικός ο αλγόριθμος DBSCAN είναι ότι μπορούμε με τη χρήση του να εντοπίσουμε την ύπαρξη θορύβου στις παρατηρήσεις μας. Στο παραπάνω παράδειγμα, προσθέτουμε εκατό δείγματα τυχαίου θορύβου και χρησιμοποιούμε εκ νέου τον αλγόριθμο DBSCAN για να ομαδοποιήσουμε τις παρατηρήσεις μας, Εικόνα 2.11.



Εικόνα 2.11. Προσθήκη τυχαίου θορύβου στις παρατηρήσεις
και εκ νέου ομαδοποίησή τους με τη χρήση του αλγορίθμου DBSCAN

Παρατηρούμε ότι βάσει των κανόνων που αναφέραμε προηγουμένως, οι παρατηρήσεις μας έχουν ομαδοποιηθεί και πάλι στις τέσσερις βασικές κατηγορίες που μας ενδιαφέρουν. Αυτή τη φορά, ο αλγόριθμος δεν κατάφερε να εντοπίσει επαρκή αριθμό δειγμάτων έτσι ώστε να δημιουργήσει ένα καινούριο πυρήνα και κατ' επέκταση ομαδοποίησε τα σημεία που περισσεύουν, δείχνοντας έτσι πως πρόκειται για θόρυβο. Σε αντίθεση λοιπόν με τον αλγόριθμο k-means, όπου όλα τα σημεία πρέπει να αντιστοιχηθούν σε μία ομάδα, εδώ τα σημεία που δεν ανήκουν σε κάποιο cluster και ταυτόχρονα δεν μπορούν να δημιουργήσουν έναν καινούριο πυρήνα, ομαδοποιούνται μεταξύ τους.

## 2.3. Επιτηρούμενη Μάθηση

Σε αυτή την κατηγορία ανήκουν τα προβλήματα για τα οποία έχουμε ως δεδομένα ζευγάρια εισόδου-εξόδου, και αξιοποιώντας τα κατάλληλα προσπαθούμε να προβλέψουμε την έξοδο μελλοντικών καταστάσεων. Τα προβλήματα επιτηρούμενης μάθησης (supervised learning) χωρίζονται σε δύο μεγάλες κατηγορίες βάσει της φύσης της μεταβλητής εξόδου. Αν η έξοδος είναι διακριτή (χωρίζεται σε κατηγορίες) τότε έχουμε να κάνουμε με προβλήματα ταξινόμησης (classification). Αντιθέτως, αν η μεταβλητή εξόδου παίρνει συνεχόμενες τιμές, τότε έχουμε να κάνουμε με προβλήματα παλινδρόμησης (regression).

Τα δεδομένα μας αποτελούνται από τις παρατηρήσεις, κάποια χαρακτηριστικά τους και την τιμή της εξόδου. Τα χαρακτηριστικά αυτά, συνήθως είναι ακατέργαστα και χρειάζονται κάποια επεξεργασία έτσι ώστε να αξιοποιηθούν κατάλληλα. Η επεξεργασία αυτή γίνεται με τη *χρήση τεχνικών επιλογής και εξαγωγής χαρακτηριστικών (feature engineering)*. Οι τεχνικές αυτές στηρίζονται σε μεγάλο βαθμό στην παρατήρηση των χαρακτηριστικών και στην μετατροπή τους σε μεταβλητές χρήσιμες για τα μοντέλα προβλέψεων που θα κατασκευάσουμε.

### 2.3.1. Διαχωρισμός των Δεδομένων σε Ομάδες Εκπαίδευσης και Αξιολόγησης

Πρώτο βήμα για τη δημιουργία ενός μοντέλου πρόβλεψης, όπως είδαμε παραπάνω, είναι η αξιοποίηση των παρατηρήσεων που έχουμε (raw data) και η μετατροπή τους σε δεδομένα κατάλληλα προς επεξεργασία. Επόμενο βήμα, αφού έχουμε κατασκευάσει τη βάση δεδομένων μας είναι ο διαχωρισμός της σε δύο ομάδες: Τα δεδομένα εκπαίδευσης (training set) και τα δεδομένα αξιολόγησης (test set). Το training set είναι το κομμάτι πάνω στο οποίο βασίζεται η κατασκευή των μοντέλων πρόβλεψης, και το test set είναι το κομμάτι πάνω στο οποίο εξετάζεται η απόδοση των μοντέλων αυτών. Συνήθως το training set αποτελείται από το 60-70% των παρατηρήσεων και το test set από το υπόλοιπο 30-40% αντίστοιχα. Η επιλογή των δεδομένων για το κάθε set είναι τυχαία.

### 2.3.2. Παλινδρόμηση

Τα μοντέλα παλινδρόμησης χρησιμοποιούνται στα προβλήματα επιτηρούμενης μάθησης όπου η μεταβλητή εξόδου λαμβάνει συνεχόμενες τιμές. Στόχος των μοντέλων είναι η εύρεση μιας συνάρτησης *f,* η οποία να αντιστοιχίζει τα δεδομένα εισόδου $\boldsymbol{x}$ σε μία συνεχόμενη μεταβλητή εξόδου $\boldsymbol{y}$. Για παράδειγμα, με τη χρήση τεχνικών παλινδρόμησης μπορεί να αναλυθεί η σχέση μεταξύ εισοδήματος (ή και άλλων παραγόντων) και χρόνου εργασίας [11]. Παρακάτω θα αναλυθούν κάποια βασικά μοντέλα παλινδρόμησης.

#### 2.3.2.1. Γραμμική Παλινδρόμηση

Ένα μοντέλο γραμμικής παλινδρόμησης (linear regression) είναι ένας τρόπος να εκφράσουμε τα δύο βασικά συστατικά μιας στατιστικής σχέσης. [12]

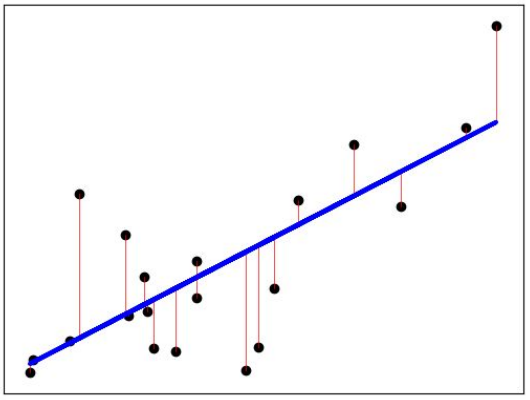
1. Την τάση της μεταβλητής εξόδου $Y$ (εξαρτημένη μεταβλητή) να σχετίζεται με την ανεξάρτητη μεταβλητή $X$ συστηματικά.
2. Τον τρόπο με τον οποίο οι παρατηρήσεις είναι διασκορπισμένες γύρω από την καμπύλη της

στατιστικής σχέσης.

Με τη μέθοδο των ελαχίστων τετραγώνων προσπαθούμε να υπολογίσουμε τους εκτιμητές (estimators), βάσει των οποίων προκύπτει η καμπύλη της παλινδρόμησης. Στην Εικόνα 2.12 έχουμε με μαύρο χρώμα τις παρατηρήσεις, με μπλε χρώμα την ευθεία της παλινδρόμησης και με κόκκινο χρώμα την απόσταση κάθε σημείου από την ευθεία αυτή. Οι κόκκινες ευθείες απεικονίζουν την απόκλιση της πραγματικής τιμής της εξαρτημένης μεταβλητής $Y_{true}$, από την τιμή της $Y$ που υπολογίζεται έπειτα από τη μέθοδο της παλινδρόμησης. Στόχος μας είναι η καμπύλη παλινδρόμησης που προκύπτει να ελαχιστοποιεί το άθροισμα όλων των αποκλίσεων. Στην παρακάτω σχεση οι εκτιμητές $\beta_0$ και $\beta_1$ υπολογίζονται έτσι ώστε να ελαχιστοποιείται η τιμή του Q.

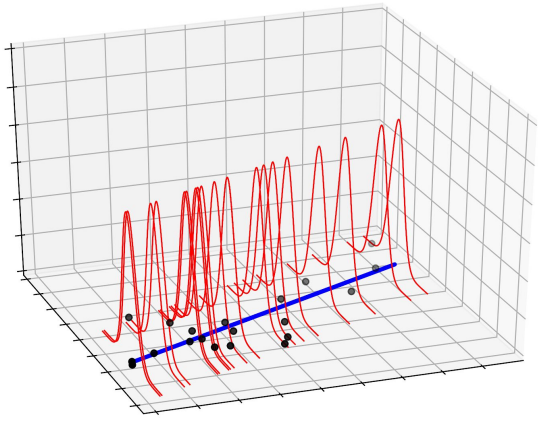
$$Q = \sum_{i=1}^{n} \left( Y_i - \beta_0 - \beta_1 X_i \right)^2 \quad (2.5)$$

Έτσι σχηματίζεται μία ευθεία γραμμή η οποία αντιπροσωπεύει το σύνολο των εξόδων των παρατηρήσεων.



Εικόνα 2.12. Ευθεία γραμμή που προκύπτει έπειτα από εφαρμογή της μεθόδου ελαχίστων τετραγώνων σε ένα σύνολο δεδομένων

Αυτό που μας δείχνει η καμπύλη παλινδρόμησης, είναι ότι σε κάθε σημείο της καμπύλης υπάρχει μία συνάρτηση πυκνότητας πιθανότητας με μέση τιμή την τιμή του σημείου και διακύμανση που καθορίζεται από την τιμή του σφάλματος, Εικόνα 2.13.



Εικόνα 2.13. Σχέση καμπύλης παλινδρόμησης με κατανομή πυκνότητας πιθανότητας

Η παραπάνω προσέγγιση είναι καθαρά στατιστική. Παρόλ' αυτά η χρήση της έχει μεγάλη σημασία στην δημιουργία μοντέλων πρόβλεψης. Έστω ότι έχουμε ένα σύνολο δεδομένων που αποτελείται από τις παρατηρήσεις μας και την τιμή της μεταβλητής εξόδου. Χωρίζουμε τις παρατηρήσεις μας σε δύο ομάδες (train set, test set) όπως αναφέραμε προηγουμένως. Με τη μέθοδο των ελαχίστων τετραγώνων φτιάχνουμε την καμπύλη παλινδρόμησης βάσει των δεδομένων που περιέχονται στο train set (προσαρμόζουμε την κλιση της ευθείας έτσι ώστε να αντιπροσωπεύει όσο το δυνατό καλύτερα το σύνολο των δεδομένων) και βάσει αυτής προβλέπουμε την έξοδο των παρατηρήσεων του test set. Έτσι εξετάζουμε την απόδοση του μοντέλου μας και το κατά πόσο μπορεί να προβλέψει την έξοδο μελλοντικών παρατηρήσεων. Στην περίπτωση που η απόδοση του μοντέλου μας είναι αρκετά υψηλή, πλέον θα έχουμε τη δυνατότητα να αντιστοιχίσουμε μελλοντικές εισόδους $X_i$, στην τιμή $Y_i$ που τους αναλογεί, στηριζόμενοι στην καμπύλη παλινδρόμησης που έχουμε δημιουργήσει.

Από τα παραπάνω παρατηρούμε ότι η σημασία των μοντέλων γραμμικής παλινδρόμησης είναι αρκετά μεγάλη, τόσο στην επιστήμη της στατιστικής, όσο και στην επιστήμη των υπολογιστών. Παρόλ' αυτά, κάποια βασικά μειονεκτήματά τους μας οδηγούν στη χρήση πιο πολύπλοκων μοντέλων.

**Η γραμμική παλινδρόμηση περιορίζεται σε γραμμικά προβλήματα.**
Από τη φύση της η γραμμική παλινδρόμηση κοιτάζει τις γραμμικές σχέσεις μεταξύ των εξαρτημένων και των ανεξάρτητων μεταβλητών. Συνεπώς υποθέτει ότι η σχέση τους μπορεί να απεικονιστεί με τη μορφή μιας ευθείας γραμμής. Αυτό φυσικά δεν είναι πάντα σωστό, καθώς στα περισσότερα προβλήματα οι ευθείες που συσχετίζουν τις μεταβλητές δεν είναι γνησίως μονότονες.

**Η γραμμική παλινδρόμηση εστιάζει μόνο στη μέση τιμή της εξαρτημένης μεταβλητής.**
Όπως η μέση τιμή δεν αντιπροσωπεύει πάντα πλήρως μία μεταβλητή, έτσι και η γραμμική παλινδρόμηση δεν αντιπροσωπεύει πλήρως τις σχέσεις μεταξύ των μεταβλητών.

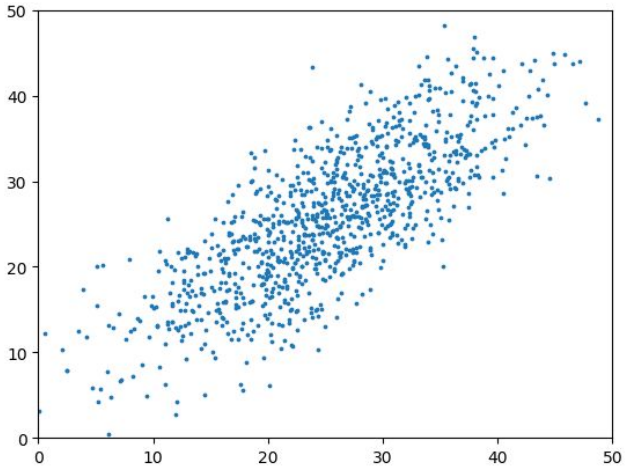
**Η γραμμική παλινδρόμηση επηρεάζεται από ακραίες τιμές παρατηρήσεων.**
Στα περισσότερα δεδομένα παρατηρούνται πολλές φορές ακραίες τιμές εξόδου σε ορισμένες παρατηρήσεις. Στην περίπτωση της γραμμικής παλινδρόμησης οι τιμές αυτές επηρεάζουν σε μεγάλο

βαθμό την κλίση της ευθείας. Οι παρατηρήσεις αυτές ονομάζονται παρατηρήσεις με ισχυρή επιρροή (influential observations) και η διαγραφή τους μπορεί να αυξήσει την απόδοση του μοντέλου μας. Αυτό συμβαίνει διότι δεν επιθυμούμε η καμπύλη μας να επηρεάζεται από τις ακραίες τιμές, αλλά από τιμές οι οποίες είναι αντιπροσωπευτικές του συνόλου που μελετάμε.

**Τα δεδομένα πρέπει να είναι ανεξάρτητα μεταξύ τους.**

Η γραμμική παλινδρόμηση υποθέτει ότι τα δεδομένα είναι ανεξάρτητα μεταξύ τους. Αυτό σημαίνει ότι η τιμή μιας ανεξάρτητης μεταβλητής δε θα έπρεπε να επηρεάζει την τιμή μιας άλλης. Αυτό φυσικά δεν είναι πάντα σωστό. Πολλές φορές οι ανεξάρτητες μεταβλητές σε ένα πρόβλημα συσχετίζονται, γεγονός που επιφέρει αρνητικές συνέπειες στα αποτελέσματά μας. Με την ανάλυση συσχέτισης (correlation analysis) μπορούμε να μετρήσουμε και να ερμηνεύσουμε το κατά πόσο η γραμμική ή μη γραμμική σχέση μεταξύ δύο συνεχόμενων μεταβλητών είναι ισχυρή. [13]

Στην Εικόνα 2.14 παρατηρούμε το διάγραμμα συσχέτισης μεταξύ δύο ανεξάρτητων μεταβλητών. Στο διάγραμμα αυτό φαίνεται πως η επηρεάζεται η τιμή της μιας μεταβλητής συναρτήσει της άλλης. Παρατηρούμε ότι τα στίγματα σχηματίζονται γύρω από την ευθεία $x = y$, γεγονός που μας οδηγεί στο συμπέρασμα ότι οι δύο αυτές μεταβλητές δεν είναι τελείως ανεξάρτητες μεταξύ τους. Συνεπώς η ταυτόχρονη χρήση τους στην εκτίμηση της ευθείας παλινδρόμησης θα οδηγούσε σε λανθασμένα αποτελέσματα.




Εικόνα 2.14. Διάγραμμα συσχέτισης μεταξύ δύο ανεξάρτητων μεταβλητών

### 2.3.2.2. Μηχανές Διανυσμάτων Υποστήριξης - Παλινδρόμηση

Όπως είδαμε παραπάνω, στόχος της γραμμικής παλινδρόμησης είναι η ελαχιστοποίηση του σφάλματος ανάμεσα στην προβλεπόμενη και την αναμενόμενη τιμή της εξόδου. Στην παλινδρόμηση διανυσμάτων υποστήριξης (support vector regression - SVR) θεωρούμε ένα κατώφλι σφάλματος $\varepsilon$, εντός του οποίου οι τιμές σφάλματος είναι αποδεκτές, και εκτιμούμε με τη χρήση του δείκτη $C > 0$ κατά πόσο "ανεχόμαστε" αποκλίσεις μεγαλύτερες του $\varepsilon$. [14]

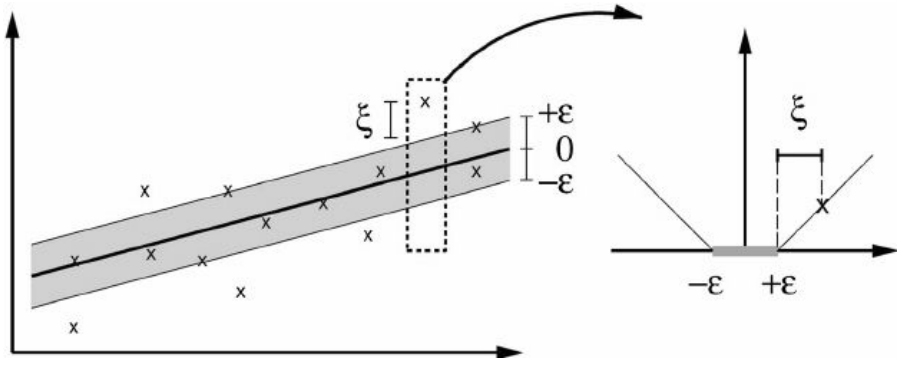
Στόχος είναι η ελαχιστοποίηση της συνάρτησης:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\left(\xi i + \xi_i^*\right) \quad (2.6)$$

η οποία υπόκειται στη σχέση:

$$\begin{cases} y_i - \langle w, x\rangle - b \le \varepsilon + \xi_i \\ \langle w, x\rangle + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \quad (2.7)$$

Στην παραπάνω σχέση ορίζονται οι δύο ευθείες της εικόνας 2.15, οι οποίες ορίζουν το γκρι πλαίσιο, εντός του οποίου η τιμή του σφάλματος είναι μικρότερη ή ίση της τιμής $\varepsilon$.




Εικόνα 2.15. Support Vector Regression [1]

Η συνάρτηση $\xi$ ορίζεται ως εξής:

$$|\xi|_\varepsilon := \begin{cases} 0 & , |\xi| \le \varepsilon \\ |\xi| - \varepsilon, & \varepsilon\iota\delta\acute{\alpha}\lambda\lambda\omega\varsigma \end{cases} \quad (2.8)$$

Τα SVMs (Support Vector Machines) έχουν ιδιαίτερη σημασία διότι χρησιμοποιούνται ευρέως σε προβλήματα ταξινόμησης όπως θα δούμε παρακάτω.

### 2.3.2.3. Μη Γραμμική Παλινδρόμηση

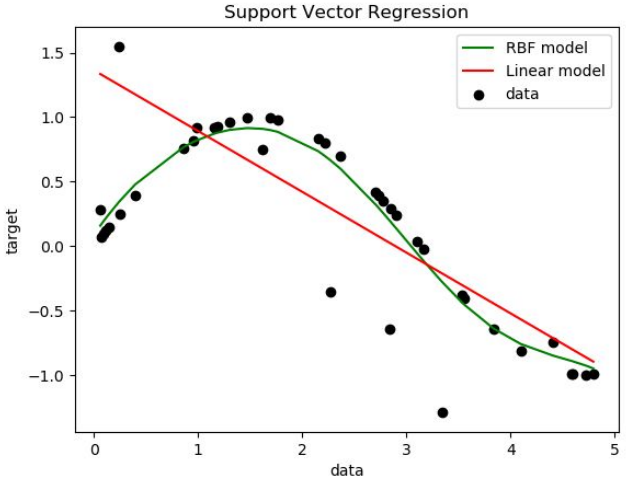
Τα προβλήματα παλινδρόμησης δεν είναι πάντοτε γραμμικά. Τις περισσότερες φορές η καμπύλη μας πρέπει να προσαρμοστεί κατάλληλα πάνω στα δεδομένα, έτσι ώστε να μπορέσουμε να φτιάξουμε ένα αρκετά καλό μοντέλο πρόβλεψης. Ο βασικός λόγος για τον οποίο κατασκευάζουμε μη γραμμικούς αλγορίθμους είναι για να εφαρμόσουμε γραμμικές μεθόδους, όχι στο επίπεδο των παρατηρήσεων, αλλά σε ένα επίπεδο $F$, το οποίο συνδέεται με το επίπεδο παρατηρήσεων με μία μη γραμμική

[1] [14]

αντιστοίχιση της μορφής:

$$\phi: \mathfrak{R}^{\mathbb{N}} \to F,\ x \to \phi(x) \quad (2.9)$$ [2]

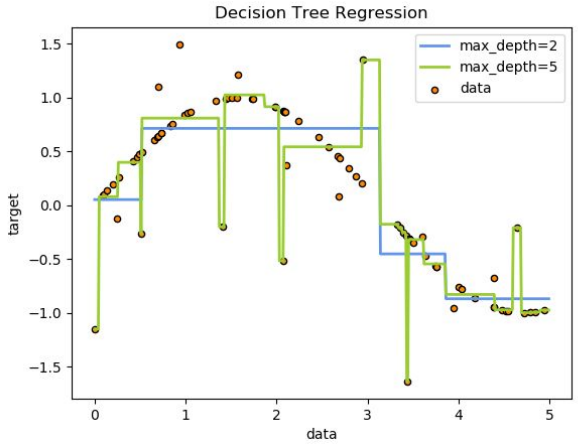
Στην Εικόνα 2.16 παρατηρούμε πως αντιμετωπίζεται το πρόβλημα της μη γραμμικότητας από τον αλγόριθμο SVR, συγκρίνοντας τα αποτελέσματα με τη γραμμική παλινδρόμηση.




Εικόνα 2.16. Σύγκριση SVR (RBF) με Linear Regression [3]

### 2.3.2.4. Παλινδρόμηση με Δέντρα Αποφάσεων

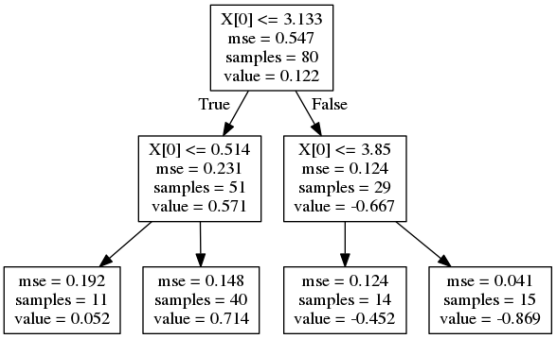
Ο βασικός αλγόριθμος πάνω στον οποίο στηρίζονται τα δέντρα αποφάσεων (decision tree regression) είναι ο ID3 και αναπτύχθηκε από τον J. R. Quinlan το 1983. Στόχος είναι μέσω άπληστης αναζήτησης, από πάνω προς τα κάτω, να χωρίσουμε τα δεδομένα μας σε ομοιογενείς υποομάδες (υποομάδες που περιέχουν παρόμοιες τιμές). [16]



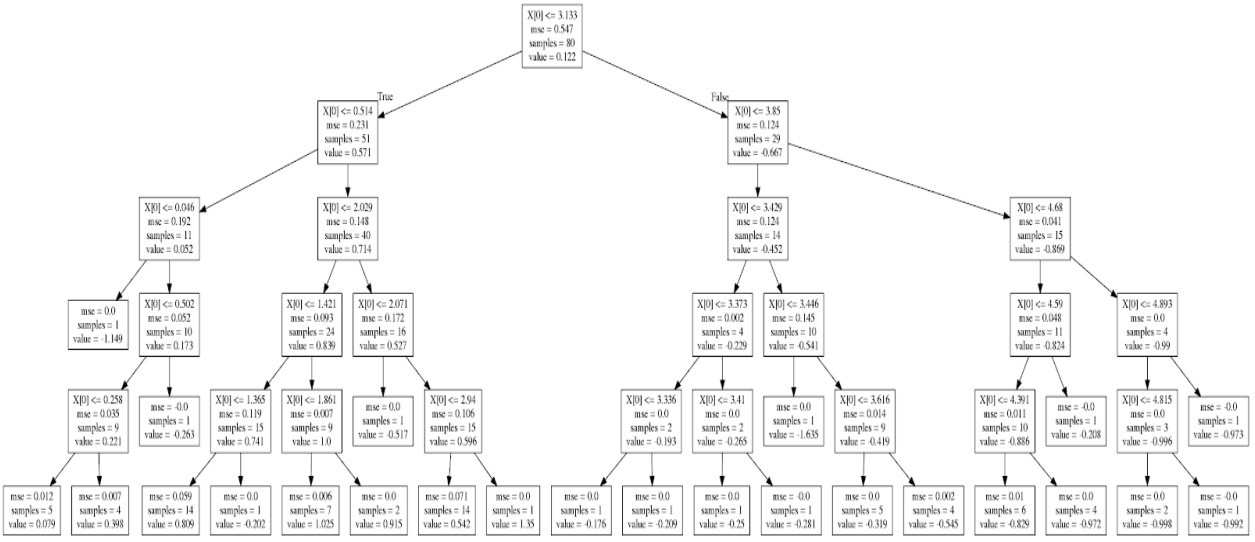

Εικόνα 2.17. Decision Tree Regression [4]

[2] [15]
[3] http://scikit-learn.org/stable/auto_examples/plot_kernel_ridge_regression.html
[4] http://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

Για να υπολογίσουμε την ομοιογένεια ενός συνόλου χρησιμοποιούμε την τυπική απόκλιση. Όταν το σύνολο είναι τελείως ομοιογενές, η τυπική απόκλιση είναι ίση με μηδέν. Στην Εικόνα 2.17 παρατηρούμε πως σχηματίζεται η καμπύλη παλινδρόμησης για δύο διαφορετικές περιπτώσεις δέντρων αποφάσεων. Στη μία περίπτωση (μπλε καμπύλη) έχει επιλεγεί μέγιστο βάθος ίσο με δύο, ενώ στην περίπτωση της πράσινης καμπύλης έχουμε μέγιστο βάθος ίσο με πέντε. Οι καμπύλες αυτές αντιπροσωπεύουν τα διαγράμματα των εικόνων 2.18 και 2.19 και αποτελούνται από ευθύγραμμα τμήματα παράλληλα στον οριζόντια άξοντα, αντιπροσωπευτικά της κάθε υποομάδας. Τα ευθύγραμμα αυτά τμήματα καθορίζονται από τη μέση (αντιπροσωπευτική) τιμή της κάθε υποομάδας, που προκύπτει μετά από κάθε επανάληψη του αλγορίθμου αναζήτησης.




Εικόνα 2.18. Decision Tree, max_depth = 2



Εικόνα 2.19. Decision Tree, max_depth = 5

Παρατηρούμε στην Εικόνα 2.16 ότι η καμπύλη έχει προσαρμοστεί σε μεγάλο βαθμό στα δεδομένα μας (overfitting), γεγονός που μπορεί να επιφέρει αρνητικά αποτελέσματα στις προβλέψεις μας.

Όπως και στην περίπτωση του SVM, τα δέντρα αποφάσεων χρησιμοποιούνται ευρέως σε προβλήματα ταξινόμησης. Ο αλγόριθμος Random Forest, που θα αναλυθεί παρακάτω, συνδυάζει πολλά δέντρα αποφάσεων έτσι ώστε να βγάλει πιο ισχυρές προβλέψεις. Στην περίπτωση της παλινδρόμησης, ο Random Forest λαμβάνει τις τιμές πολλών διαφορετικών δέντρων αποφάσεων και

επιστρέφει τη μέση τιμή τους.

### 2.3.3. Ταξινόμηση

Η έξοδος των προβλημάτων που μελετάμε δεν περιγράφεται πάντοτε με συνεχόμενη μεταβλητή. Πολλές φορές έχει διακριτές τιμές. Το πρόβλημα της ανεπιθύμητης αλληλογραφίας είναι χαρακτηριστικό παράδειγμα προβλήματος ταξινόμησης (classification). Η τιμή της εξόδου μπορεί να πάρει την τιμή "ΝΑΙ" ή "ΟΧΙ" ανάλογα με την κατηγορία στην οποία ανήκει η εκάστοτε αλληλογραφία. Η ταξινόμηση είναι η η αντίστοιχη διαδικασία της συσταδοποίησης (clustering) στα προβλήματα επιτηρούμενης μάθησης. Παρακάτω θα αναλυθούν κάποια βασικά μοντέλα ταξινόμησης.

#### 2.3.3.1. Λογιστική Παλινδρόμηση

Η λογιστική παλινδρόμηση (logistic regression) επεκτείνει τις τεχνικές της παλινδρόμησης σε περιπτώσεις που η έξοδος έχει διακριτή μορφή (χωρίζεται σε κατηγορίες), [17]. Στη γραμμική παλινδρόμηση, η αναμενόμενη (μέση) τιμή της εξόδου $Y$, δεδομένης της εισόδου $x$, $E(Y|x)$, εκφράζεται με τη μορφή μιας ευθείας γραμμής:

$$E(Y|x) = \beta_0 + \beta_1 x \quad (2.10)$$

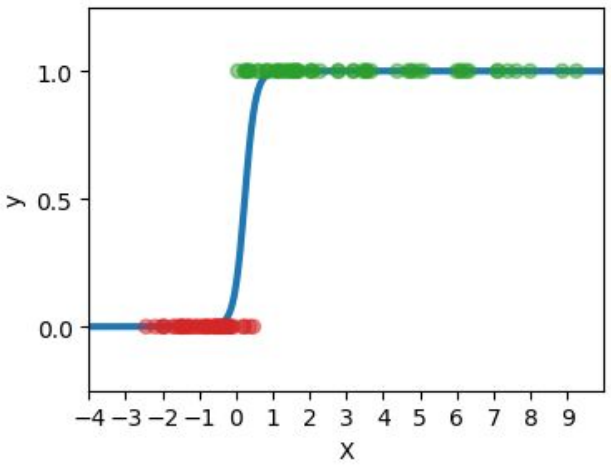
Θέτουμε $\pi(x) = E(Y|x)$ για να αναπαραστήσουμε την υποθετική μέση τιμή του $Y$, δεδομένης της τιμής του $x$, όταν κάνουμε χρήση της λογιστικής κατανομής. Η μορφή του μοντέλου λογιστικής παλινδρόμησης είναι η εξής:

$$\pi(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}} \quad (2.11)$$

Με λογαριθμική μετατροπή της παραπάνω σχέσης καταλήγουμε στην επόμενη σχέση, όπου παρατηρούμε ότι η συνάρτησή μας διαθέτει πολλές από τις ιδιότητες ενός γραμμικού μοντέλου παλινδρόμησης. [18]

$$g(x) = \ln\left[\frac{\pi(x)}{1-\pi(x)}\right] = \beta_0 + \beta_1 x \quad (2.12)$$

Στην Εικόνα 2.20 έχουμε κάποια δείγματα των οποίων η έξοδος παίρνει τις τιμές $0, 1$. Η καμπύλη της λογιστικής παλινδρόμησης (μπλε χρώμα) σχεδιάζεται με τέτοιο τρόπο, ώστε να υπάρχει σαφής διαχωρισμός μεταξύ των τιμών της εξόδου. Βάσει αυτού του μοντέλου μπορούμε πλέον να προβλέψουμε την έξοδο για μελλοντικές τιμές του $x$.



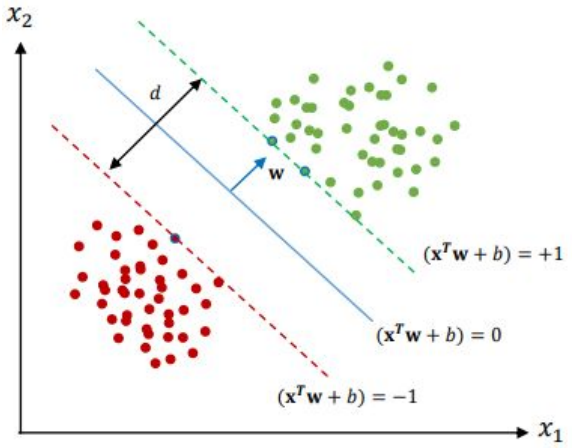

Εικόνα 2.20. Παράδειγμα λογιστικής παλινδρόμησης

Παρόλ' αυτά, σε αρκετές περιπτώσεις το μοντέλο της γραμμικής παλινδρόμησης αποτυγχάνει. Αυτο συμβαίνει για τρεις βασικούς λόγους.

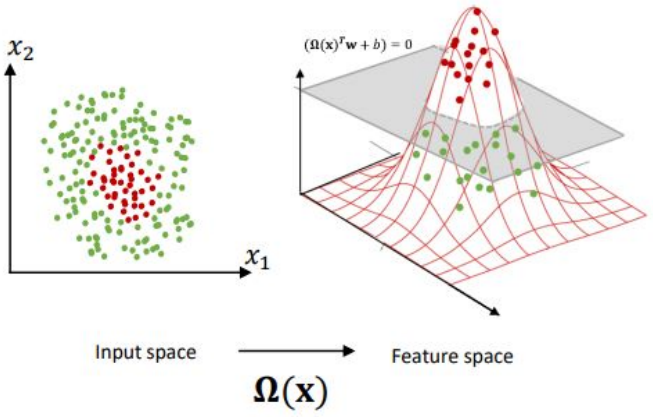
- Τα μοντέλα λογιστικής παλινδρόμησης δεν μπορούν να διαχειριστούν δεδομένα με μεγάλο αριθμό διαστάσεων (large feature space).
- Τα μοντέλα λογιστικής παλινδρόμησης δεν μπορούν να διαχειριστών μεγάλο αριθμό χαρακτηριστικών/κατηγορικών μεταβλητών.
- Η λογιστική παλινδρόμηση στηρίζεται σε μετασχηματισμούς για μη γραμμικά χαρακτηριστικά.

#### 2.3.3.2. Μηχανές Διανυσμάτων Υποστήριξης - Ταξινόμηση

Τα προβλήματα μη γραμμικότητας και χαρακτηριστικών μεγάλων διαστάσεων λύνονται σε μεγάλο βαθμό με τη χρήση των μηχανών διανυσμάτων υποστήριξης (support vector machines - SVM). Ο SVM χρησιμοποιεί τις πιο ακραίες περιπτώσεις ανάμεσα στις κλάσεις έτσι ώστε να φτιάξει το μοντέλο. Έστω για παράδειγμα ότι έχουμε σε μία εικόνα ένα ζώο και θέλουμε να ξεχωρίσουμε αν είναι γάτα ή σκύλος. Κατά την προσαρμογή των δεδομένων, ο SVM χρησιμοποιεί τις γάτες που μοιάζουν πολύ με σκύλους και τους σκύλους που μοιάζουν πολύ με γάτες για να ορίσει το φράγμα μεταξύ των δύο κλάσεων. Τα στοιχεία αυτά ονομάζονται διανύσματα υποστήριξης (support vectors).




Εικόνα 2.21. Χρήση SVM για ταξινόμηση παρατηρήσεων [5]




Εικόνα 2.22. Μετασχηματισμός δεδομένων σε επίπεδο με περισσότερες διαστάσεις, για επίλυση μη γραμμικών προβλημάτων [5]

Αν θεωρήσουμε ότι η Εικόνα 4.21 απεικονίζει το σύνολο των παρατηρήσεών μας, τότε τα κυκλωμένα (με μπλε χρώμα) στοιχεία θα ήταν τα διανύσματα υποστήριξης, αντιπροσωπευτικά της κάθε κλάσης.

Οι κλάσεις όμως στα προβλήματα ταξινόμησης δεν είναι πάντα γραμμικά διακρίσιμες. Στην Εικόνα 2.22 παρατηρούμε ένα πρόβλημα όπου οι δύο κλάσεις χωρίζονται με μη γραμμικό τρόπο. Η επίλυση του προβλήματος γίνεται με τη χρήση συναρτήσεων πυρήνα (kernel functions), οι οποίες μετασχηματίζουν το πρόβλημά μας σε καινούριες (περισσότερες) διαστάσεις, όπου πλέον το πρόβλημα μπορεί να λυθεί γραμμικά. Στην Εικόνα 2.22 φαίνεται πώς μπορεί να υπολογιστεί ένα επίπεδο, το οποίο χωρίζει τις δύο κλάσεις με γραμμικό πλέον τρόπο. Οι πιο διαδεδομένες συναρτήσεις πυρήνα είναι οι παρακάτω. [20]

[5] [19]

- Πολυωνυμικός πυρήνας (Polynomial Kernel)

$$k(x,x') = \langle x,x' \rangle^d \quad (2.13)$$

- Γκαουσιανός πυρήνας (RBF/Gaussian Kernel)

$$k(x,x') = exp\left(-\frac{\|x-x'\|^2}{2\sigma^2}\right) \quad (2.14)$$

- Σιγμοειδής πυρήνας (Sigmoid Kernel)

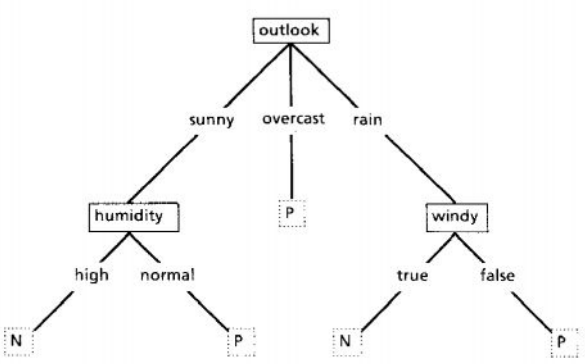
$$k(x,x') = \tanh(\kappa\langle x,x' \rangle + \Theta) \quad (2.15)$$

**2.3.3.3. Ταξινόμηση με Δέντρα Αποφάσεων**

Η χρήση των δέντρων αποφάσεων είναι πολύ διαδεδομένη στα προβλήματα ταξινόμησης. Κάθε παρατήρηση αποτελείται από ένα σύνολο χαρακτηριστικών και μία κατηγορική έξοδο. Στόχος των δέντρων αποφάσεων είναι ο διαχωρισμός των παρατηρήσεων βάσει του αλληλοσχετισμού των χαρακτηριστικών του. Στο παράδειγμα της Εικόνας 2.23 έχουμε ένα δέντρο αποφάσεων που σχετίζεται με τον καιρό μιας μέρας της εβδομάδας και τις κλάσεις *P (positive outcome)* και *N (negative outcome)* που χαρακτηρίζουν την κατάστασή του. [21]

Τα χαρακτηριστικά που χρησιμοποιούμε για την ταξινόμηση των παρατηρήσεων είναι οι συνθήκες του καιρού (ηλιοφάνεια, βροχή, συννεφιά), η θερμοκρασία, η υγρασία και ο αέρας. Στην Εικόνα 2.23 παρατηρούμε όμως ότι δεν έχουν χρησιμοποιηθεί όλα τα χαρακτηριστικά για να γίνει η κατηγοριοποίηση. Η λογική των δέντρων αποφάσεων είναι η κατηγοριοποίηση να γίνεται με τέτοιο τρόπο έτσι ώστε τα μοντέλα που κατασκευάζονται να μπορούν να γενικευθούν, δηλαδή να επιλύουν μελλοντικά προβλήματα, χωρίς όμως να είναι επηρεάζονται από μεμονωμένες παρατηρήσεις των έως τώρα δεδομένων.

Ο αλγόριθμος που χρησιμοποιείται είναι ο ID3, όπως αναφέρεται και για την περίπτωση της παλινδρόμησης, και η βασική του δομή είναι επαναληπτική. Κάθε φορά επιλέγεται ένα υποσύνολο του training set, που ονομάζεται παράθυρο, και δημιουργείται ένα δέντρο βάσει των παρατηρήσεων που περιέχονται σε αυτό. Όλες οι υπόλοιπες παρατηρήσεις του training set ταξινομούνται σύμφωνα με το δέντρο που κατασκευάστηκε. Αν η ταξινόμηση που προέκυψε είναι σωστή για όλες τις παρατηρήσεις ο αλγόριθμος τερματίζει. Σε περίπτωση που αυτό δε συνέβη, ένα υποσύνολο των παρατηρήσεων που δεν ταξινομήθηκαν σωστά προστίθεται στο παράθυρο και η διαδικασία συνεχίζεται.




Εικόνα 2.23. Δέντρο αποφάσεων προβλήματος ταξινόμησης

### 2.3.3.4. Ταξινόμηση με Τυχαίο Δάσος

Πολλές φορές, συνδυάζοντας απλούστερες μεθόδους προβλέψεων, μπορούμε να κατασκευάσουμε ισχυρότερα μοντέλα. Οι ensemble μέθοδοι, όπως ονομάζονται, μπορούν να προκύψουν με διάφορους τρόπους. Δύο από τους πιο γνωστούς αλγορίθμους είναι ο αλγόριθμος Boosting [22], και ο αλγόριθμος Bagging (Bootstrap aggregating) [23]. Ο αλγόριθμος Random Forest μοιάζει σε μεγάλο βαθμό με τον Bagging, με μόνη διαφορά ότι σε κάθε στάδιο διαχωρισμού επιλέγεται ένα υποσύνολο των χαρακτηριστικών της εισόδου.

Ένα Τυχαίο Δάσος (Random Forest) αποτελείται από μία συλλογή ταξινομητών δενδρικής δομής (tree structured classifiers) $\{h(x, \Theta_k),\ k = 1, ...\}$, όπου $\{\Theta_k\}$ είναι ανεξάρτητα ομοίως κατανεμημένα τυχαία διανύσματα, και κάθε δέντρο αποφασίζει για την κλάση της εισόδου $x$. Οι αποφάσεις όλων των δέντρων συμψηφίζονται, και στο τέλος επιλέγεται η δημοφιλέστερη απόφαση για την ταξινόμηση της εισόδου. [24]

Κάποιες βασικές παράμετροι του αλγορίθμου είναι ο ελάχιστος αριθμός δειγμάτων που πρέπει να υπάρχει σε κάθε τερματικό κόμβο (φύλλο του δέντρου), καθώς και το μέγιστο επιτρεπόμενο βάθος κάθε δέντρου. Η έννοια του βάθους του δέντρου απεικονίζεται στην Εικόνα 2.24. Όσο μικρότερη είναι η τιμή του, τόσο μειώνονται οι πιθανότητες να υπάρχει overfitting, δηλαδή το δέντρο να έχει προσαρμοστεί στις παρατηρήσεις του training set, και κατ' επέκταση να μην μπορεί να γενικεύσει σε περεταίρω δεδομένα. Τέλος, όσο περισσότερα δείγματα έχουμε σε έναν τερματικό κόμβο, τόσο μειώνονται οι πιθανότητες του να έχουμε μοντελοποιήσει τυχόν θόρυβο.

Εικόνα 2.24. Βάθος και ύψος σε ένα δέντρο αποφάσεων

# 3. Βαθιά Μάθηση

## 3.1. Εισαγωγή

Η τεχνολογίες μηχανικής μάθησης τροφοδοτούν πολλές πτυχές της σύγχρονης καθημερινότητας. Οι μηχανές αναζήτησης στο διαδίκτυο, τα συστήματα συστάσεων, τα smartphones, ακόμα και οι κάμερες χρησιμοποιούν τέτοιου είδους τεχνολογία. Παρόλ' αυτά, η ικανότητα των συμβατικών τεχνικών μηχανικής μάθησης είναι πολύ περιορισμένη όταν επεξεργάζονται φυσικά δεδομένα σε ακατέργαστη μορφή (raw data). Στο προηγούμενο κεφάλαιο δείξαμε πόσο σημαντική είναι η διαδικασία του feature engineering, δηλαδή της μετατροπής των ακατέργαστων δεδομένων σε χρήσιμες πληροφορίες για τα μοντέλα που κατασκευάζουμε. Η διαδικασία όμως του feature engineering είναι αρκετά απαιτητική από άποψη χρόνου εργασία, καθώς γίνεται χειροκίνητα από αναλυτές και data scientists, αναδεικνύοντας έτσι τις αδυναμίες των παραδοσιακών αλγορίθμων μάθησης. Η ανάγκη για αυτοματοποίηση τέτοιου είδους διαδικασιών μας οδήγησε στη δημιουργία ενός νέου τομέα που ονομάζεται *αναπαραστατική μάθηση (representation learning) [25]***.** Η αναπαραστατική μάθηση είναι ένα σύνολο μεθόδων που επιτρέπουν σε μία μηχανή να δεχτεί ακατέργαστα δεδομένα και να ανακαλύψει από μόνη της τον τρόπο με τον οποίο αυτά θα αναπαρίστανται έτσι ώστε να συμβάλλουν στην επίλυση του προβλήματος.

Οι μέθοδοι *βαθιάς μάθησης (deep learning)* είναι μέθοδοι αναπαραστατικής μάθησης με πολλαπλά επίπεδα αναπαράστασης [26]. Κάθε νέο επίπεδο δημιουργείται έπειτα από μετασχηματισμό του προηγούμενου μέσω απλών μη γραμμικών μεθόδων, και αποτελεί έναν καινούριο (πιο αφηρημένο) τρόπο αναπαράστασης των δεδομένων. Στα προβλήματα ταξινόμησης, τα υψηλότερα επίπεδα αναπαράστασης ενισχύουν πτυχές της εισόδου που είναι σημαντικές για το διαχωρισμό των παρατηρήσεων, και αντίστοιχα καταστέλλουν τα χαρακτηριστικά εκείνα που η σημασία τους δεν επηρεάζει σε μεγάλο βαθμό την κατάσταση της εξόδου.

Παρακάτω θα αναλύσουμε τη λειτουργία των νευρωνικών δικτύων, η σημασία των οποίων είναι ιδιαίτερα σημαντική, τόσο σε προβλήματα μηχανικής μάθησης, όσο και σε προβλήματα βαθιάς μάθησης.

## 3.2. Τεχνητά Νευρωνικά Δίκτυα

Τα *τεχνητά νευρωνικά δίκτυα (Artificial Neural Networks - ANN)* είναι μία προσπάθεια μοντελοποίησης της ικανότητας του ανθρώπινου νευρικού συστήματος να επεξεργάζεται πληροφορίες [27]**.**

Η χρήση τους γίνεται όλο και πιο συχνή τα τελευταία χρόνια καθώς, συνδυάζοντας διαφορετικούς αλγορίθμους μηχανικής μάθησης, μπορούν να επιλύσουν προβλήματα με αρκετά πολύπλοκες εισόδους. Κάποιες από τις βασικές εφαρμογές τους είναι οι εξής [28]:

- **Ιατρική**
  Ανάλυση κυττάρων για τον καρκίνο του μαστού, ανάλυση εγκεφαλογραφήματος/καρδιογραφήματος, μείωση των εξόδων στα νοσοκομεία.

- **Τηλεπικοινωνίες**
  Συμπίεση εικόνας και δεδομένων, άμεση μετάφραση ξένης γλώσσας.

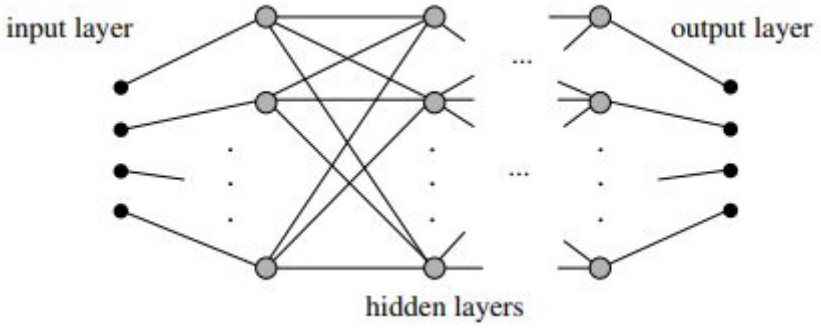
- **Οικονομία**
  Εκτίμηση ακινήτων, πρόβλεψη ισοτιμίας μεταξύ των νομισμάτων, παρακολούθηση στεγαστικών δανείων.

- **Ομιλία**
  Αναγνώριση φωνής, μετατροπή κειμένου σε ήχο.

- **Αυτοκινητοβιομηχανία**
  Συστήματα αυτόματης οδήγησης, συστήματα αυτόματου φρεναρίσματος, εντοπισμός σφαλμάτων.

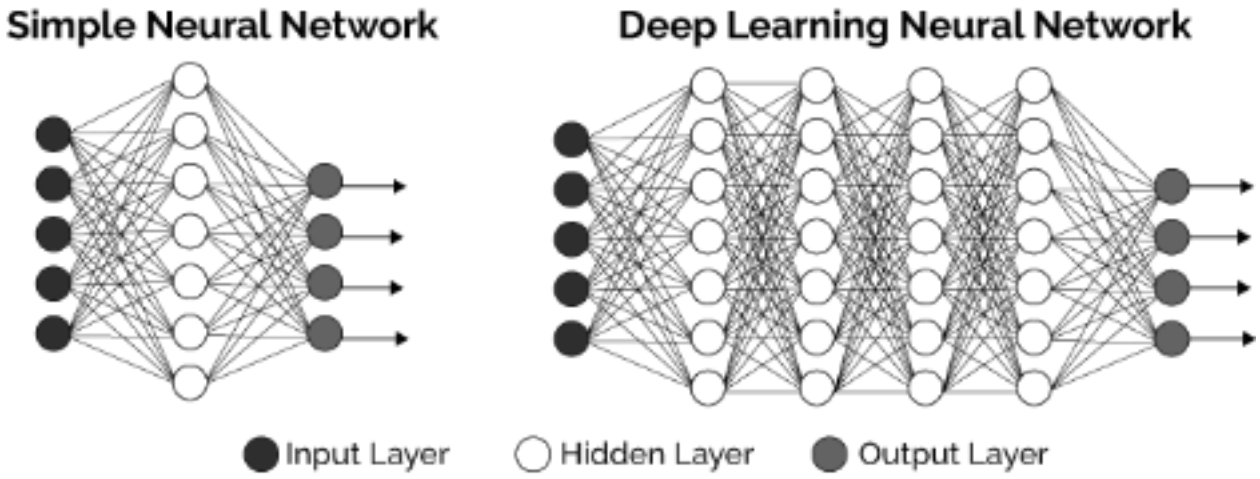
Η αρχιτεκτονική ενός δικτύου *(I, N, O, E)* αποτελείται από ένα σύνολο *I* θέσεων εισόδου, ένα σύνολο *N* υπολογιστικών μονάδων, ένα σύνολο *O* θέσεων εξόδου, και ένα σύνολο *E* σταθμισμένων και προσανατολισμένων ακμών. Οι ακμές έχουν τη μορφή διανύσματος *(u, v, w)*, όπου $u \in I \cup N$, $v \in N \cup O$, και $w \in \mathbb{R}$ το βάρος της κάθε ακμής.

Στις θέσεις εισόδου οι πληροφορίες διοχετεύονται στο δίκτυο και δεν εκτελείται κάποια υπολογιστική διαδικασία, στις θέσεις εξόδου λαμβάνουμε τα αποτελέσματα και στα *N* στοιχεία του δικτύου γίνονται όλοι οι απαραίτητοι υπολογισμοί. Επίσης όλες οι ακμές μεταξύ των στοιχείων του δικτύου (εισόδου, εξόδου, υπολογισμού) είναι σταθμισμένες. Στις στρωματοποιημένες αρχιτεκτονικές οι *N* υπολογιστικές μονάδες υποδιαιρούνται σε $\ell$ υποσύνολα, $N_1$, $N_2$, ....., $N_\ell$ , με τέτοιο τρόπο ώστε οι μονάδες του συνόλου $N_1$ να συνδέονται μόνο με τις μονάδες του συνόλου $N_2$, οι μονάδες του συνόλου $N_2$ μόνο με τις μονάδες του συνόλου $N_3$, κοκ. Οι θέσεις εισόδου συνδέονται μόνο με τις μονάδες τους συνόλου $N_1$, ενώ οι μονάδες εξόδου συνδέονται μόνο με τις μονάδες του συνόλου $N_\ell$. Στη συνηθισμένη ορολογία οι μονάδες του συνόλου $N_\ell$ αποτελούν τις μονάδες εξόδου του δικτύου, δηλαδή το *στρώμα εξόδου (output layer)*. Οι θέσεις εισόδου αποτελούν το *στρώμα εισόδου (input layer),* ενώ τα ενδιάμεσα στρώματα, τα οποία δε συνδέονται απευθείας είτε με την είσοδο, είτε με την έξοδο, ονομάζονται *κρυφά στρώματα (hidden layers)*. Η αρχιτεκτονική αυτή απεικονίζεται στην Εικόνα 3.1.




Εικόνα 3.1. Αρχιτεκτονική στρωμάτων νευρωνικού δικτύου

Το απλούστερο μοντέλο ενός ANN είναι ο *νευρώνας (perceptron)* [29], ο οποίος είναι ένα απλό προς τα εμπρός τροφοδοτούμενο (feedforward) δίκτυο που δεν περιέχει κανένα κρυφό στρώμα και λειτουργει σαν ένας γραμμικός-δυαδικός ταξινομητής. Ο όρος *προς τα εμπρός τροφοδοτούμενο* αναδεικνύει την από την είσοδο προς την έξοδο κατεύθυνση της ροής της πληροφορίας σε ένα νευρωνικό δίκτυο. Ένα ANN με πολλαπλά στρώματα μεταξύ της εισόδου και της εξόδου ονομάζεται *βαθύ νευρωνικό δίκτυο (deep neural network)*, Εικόνα 3.2. Παρακάτω θα αναλυθούν κάποια βασικά χαρακτηριστικά των τεχνητών νευρωνικών δικτύων, καθώς και ο τρόπος λειτουργίας τους. [30]




Εικόνα 3.2. Διαφορά μεταξύ simple και deep neural network [6]

### 3.2.1. Εκτίμηση της Συνάρτησης Πυκνότητας Πιθανότητας ή Εκτίμηση της Συνάρτησης Μάζας Πιθανότητας

Εισάγουμε την έννοια της *εκτίμησης της συνάρτησης πυκνότητας πιθανότητας (ΣΠΠ)*, η οποία χρειάζεται για την κατανόηση των συναρτήσεων κόστους παρακάτω. Σε αυτό το πρόβλημα, ο αλγόριθμος μηχανικής μάθησης προσπαθεί να μάθει μία συνάρτηση της μορφής $p_{model} : \mathbb{R}^n \rightarrow \mathbb{R}$, όπου η συνάρτηση $p_{\text{model}}(\mathbf{x})$ μπορεί να ερμηνευθεί είτε ως μια συνάρτηση πυκνότητας πιθανότητας (εάν το **x** παίρνει συνεχείς τιμές), είτε ως μια συνάρτηση μάζας πιθανότητας (εάν το **x** λαμβάνεις διακριτές τιμές). Για να συμβεί αυτό, ο αλγόριθμος θα πρέπει να κατανοήσει τη δομή των δεδομένων που δέχεται. Η εκτίμηση της ΣΠΠ μας βοηθάει να επιλύσουμε σημαντικά προβλήματα που αντιμετωπίζουμε κατά τη διάρκεια των προβλημάτων μηχανικής μάθησης. Ένα από αυτά είναι και το πρόβλημα των ελλειπουσών τιμών (missing values). Πολλές φορές στα δεδομένα μας απουσιάζουν κάποια χαρακτηριστικά (features) από ορισμένες παρατηρήσεις. Με τη χρήση της ΣΠΠ μπορούμε να φτιάξουμε μια συνάρτηση κατανομής πιθανότητας $p(\mathbf{x})$, και να τη χρησιμοποιήσουμε για να καλύψουμε τα κενά.

### 3.2.2. Συνάρτηση Κόστους

Στόχος του αλγορίθμου BP, όπως και των περισσότερων αλγορίθμων που στοχεύουν στη βελτιστοποίηση ενός μοντέλου, είναι η ελαχιστοποίησης μιας *συνάρτησης κόστους (loss function)*. Η συνάρτηση κόστους μας δείχνει πόσο καλά ανταποκρίνεται το μοντέλο μας, υπολογίζοντας την απόκλιση της προβλεπόμενης τιμής της εξόδου από την πραγματική. Παρακάτω ορίζονται τρεις βασικές συναρτήσεις κόστους:

[6] https://cdn-images-1.medium.com/max/800/1*r0fxAZRpRGapPnC4bniDiQ.png

- **Mean squared error (MSE)**

$$J(\boldsymbol{\theta}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim\hat{p}_{data}}(\boldsymbol{y}|\boldsymbol{x})\left\|\boldsymbol{y}-f(\boldsymbol{x};\boldsymbol{\theta})\right\|^2 + const \quad (3.1)$$

- **Mean absolute error (MAE)**

$$J(\boldsymbol{\theta}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim\hat{p}_{data}}(\boldsymbol{y}|\boldsymbol{x})\left\|\boldsymbol{y}-f(\boldsymbol{x};\boldsymbol{\theta})\right\|_1 + const \quad (3.2)$$

- **Cross-entropy**

$$J(\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim\hat{p}_{data}}\log p_{model}(\boldsymbol{y}|\boldsymbol{x}) \quad (3.3)$$

### 3.2.3. Αλγόριθμος Πίσω Διάδοσης

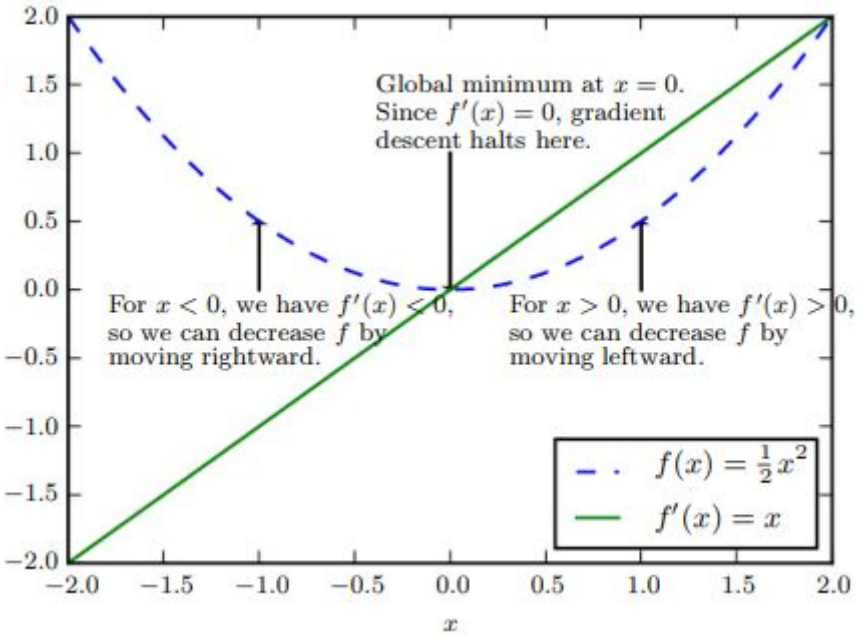
Ο *αλγόριθμος πίσω διάδοσης (backpropagation - BP)* είναι ένας από τους δημοφιλέστερους αλγορίθμους που χρησιμοποιούνται στα νευρωνικά δίκτυα [31]. Χρησιμοποιείται για τον υπολογισμό του gradient (ανάδελτα) που χρησιμεύει για τον υπολογισμό της τιμής του βάρους (w) κάθε ακμής. Στα νευρωνικά δίκτυα η διάδοση της πληροφορίας γίνεται προς τα εμπρός (forward propagation). Η μεταβλητή εισόδου $\boldsymbol{x}$ παρέχει στο δίκτυο την αρχική πληροφορία, η οποία συνεχίζει και διαδίδεται από στρώμα σε στρώμα έως ότου παραχθεί η έξοδος $\hat{\boldsymbol{y}}$. Κατά τη διαδικασία της εκπαίδευσης, και γνωρίζοντας πλέον την τιμή της μεταβλητής $\hat{\boldsymbol{y}}$, μπορούμε να υπολογίσουμε την τιμή $J(\theta)$ του κόστους. Με τη χρήση του αλγορίθμου BP μπορούμε να τροφοδοτήσουμε την πληροφορία αυτή προς τα πίσω στο δίκτυο, έτσι ώστε να υπολογίσουμε την τιμή $\nabla_{\theta}J(\theta)$, που μας ενδιαφέρει για την ανανέωση των τιμών του βάρους.

### 3.2.4. Αλγόριθμος σύγκλισης με ελάττωση της παραγώγου

Ο *αλγόριθμος σύγκλισης με ελάττωση της παραγώγου (gradient descent)* μας βοηθάει στην ελαχιστοποίηση της συνάρτησης κόστους. Όπως αναφέραμε παραπάνω, ο αλγόριθμος BP τροφοδοτεί την τιμή της συνάρτησης κόστους προς τα πίσω στο δίκτυο και στη συνέχεια υπολογίζουμε το gradient της τιμής αυτής. Στην Εικόνα 3.3 παρατηρούμε πως συνδέεται η έννοια της παραγώγου $f'(x)$ με την ελαχιστοποίηση της συνάρτησης $f(x)$. Στα νευρωνικά δίκτυα η καμπύλη αυτή προκύπτει από ένα διάνυσμα εισόδου $\boldsymbol{x}$, οπότε αντί της παραγώγου χρησιμοποιούμε το gradient $\nabla_{\mathbf{x}}f(\mathbf{x})$, δηλαδή το άθροισμα όλων των μερικών παραγώγων της $f$. Στόχος του gradient descent είναι να υπολογίσει ποια κατεύθυνση πρέπει να ακολουθήσουμε έτσι ώστε να φτάσουμε "βήμα βήμα" στο ελάχιστο της καμπύλης. Βάσει της κατεύθυνσης αυτής ανανεώνονται τα βάρη των ακμών σε ένα νευρωνικό δίκτυο. Το μέγεθος του βήματος καθορίζεται από την (θετική) τιμή μιας βαθμωτής συνάρτησης, που ονομάζεται ρυθμός εκμάθησης. Η παρακάτω σχέση μας δείχνει τη διαδικασία με την οποία ο gradient descent προσπαθεί να προσεγγίσει το ελάχιστο της καμπύλης.

$$\boldsymbol{x}' = \boldsymbol{x} - \varepsilon\nabla_{\boldsymbol{x}}f(\boldsymbol{x}) \quad (3.4)$$

Ο gradient descent συγκλίνει όταν κάθε στοιχείο του gradient είναι ίσο με μηδέν (στην πραγματικότητα πολύ κοντά στο μηδέν).




Εικόνα 3.3. Απεικόνιση λειτουργίας του αλγορίθμου gradient descent,
με τη χρήση της συνάρτησης παραγώγου [7]

### 3.2.5. Στοχαστικός αλγόριθμος σύγκλισης με ελάττωση της παραγώγου

Στην περίπτωση που τα δεδομένα που διαχειριζόμαστε είναι πάρα πολλά, η υπολογιστική πολυπλοκότητα του gradient descent αυξάνεται σε πολύ μεγάλο βαθμό. Για να αντιμετωπίσουμε το πρόβλημα αυτό χρησιμοποιούμε μία παραλλαγή του αλγορίθμου gradient descent, τον *στοχαστικό αλγόριθμο σύγκλισης με ελάττωση της παραγώγου (stochastic gradient descent - SGD)* [32]. Σύμφωνα με αυτόν, μετά από κάθε επανάληψη στη διαδικασία της εκπαίδευσης του δικτύου, υπολογίζουμε το gradient βάσει ενός τυχαίου παραδείγματος που έχουμε επιλέξει. Η στοχαστική διαδικασία στηρίζεται στο παράδειγμα που επιλέγουμε τυχαία σε κάθε επανάληψη και εκτιμούμε ότι η συμπεριφορά του gradient θα είναι αντίστοιχη της προηγούμενης περίπτωσης.
Αντίστοιχα υπάρχει ο αλγόριθμος mini-batch gradient descent, ο οποίος χρησιμοποιεί *n* τυχαία παραδείγματα σε κάθε επανάληψη για τον υπολογισμό του gradient.

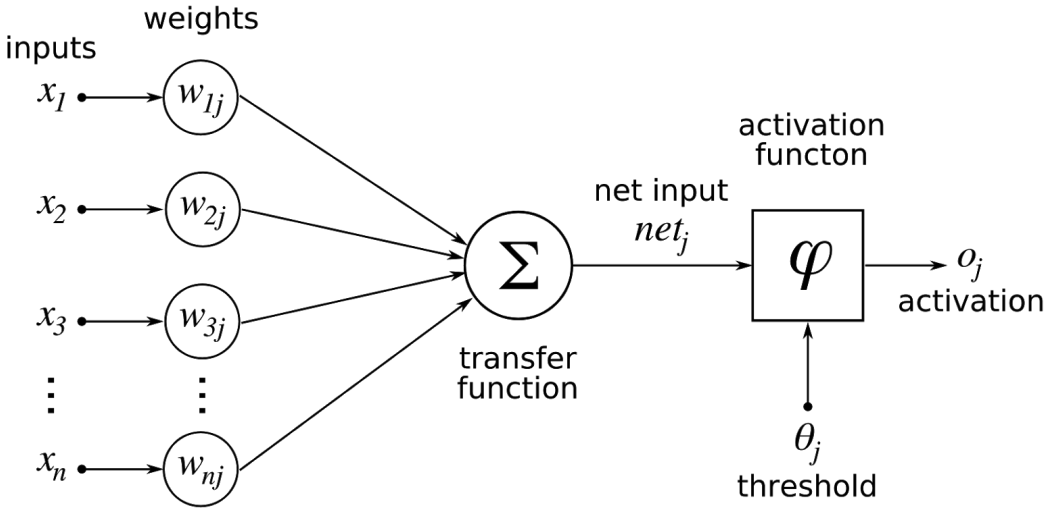
### 3.2.6. Ρυθμός Εκμάθησης

Ο *ρυθμός εκμάθησης (learning rate)* είναι μία έννοια που συνδυάζει την ελαχιστοποίηση της συνάρτησης κόστους με τη βελτιστοποίηση του μοντέλου που κατασκευάζουμε. Μας δείχνει σε τι βαθμό θα πρέπει να αλλάζει το δίκτυο τις παραμέτρους του για να πετύχει καλύτερα αποτελέσματα. Στις περισσότερες περιπτώσεις η τιμή του πρέπει να είναι αρκετά μικρή, έτσι ώστε το δίκτυο να συγκλίνει σε κάτι χρήσιμο, αλλά παράλληλα αρκετά μεγάλη έτσι ώστε να περιοριστεί ο χρόνος της σύγκλισης. Ο αλγόριθμος Adagrad προσαρμόζει το ρυθμό εκμάθησης στις παραμέτρους [33]. Οι παράμετροι που σχετίζονται με features που εμφανίζονται αραιά παρουσιάζουν μεγάλες μεταβολές στην τιμή τους, ενώ αντίθετα αν ένα feature εμφανίζεται πολύ συχνά, οι παράμετροι που σχετίζονται με αυτό παρουσιάζουν μικρές μεταβολές. Η μέθοδος Adadelta [34] είναι μία επέκταση του αλγορίθμου Adagrad, ενώ η μέθοδος Adam [35] χρησιμοποιείται ευρέως στα νευρωνικά δίκτυα, καθώς αξιοποιεί τη μέση τιμή προηγούμενων gradients.

[7] [30]

### 3.2.7. Συνάρτηση Ενεργοποίησης

Στην Εικόνα 5.1 είδαμε την αρχιτεκτονική στρωμάτων ενός νευρωνικού δικτύου, καθώς και τις ακμές που ενώνουν τους νευρώνες ανάμεσα στα διαφορετικά στρώματα. Στην Εικόνα 3.4 παρατηρούμε τη ροή της πληροφορίας σε ένα νευρώνα ενός δικτύου.



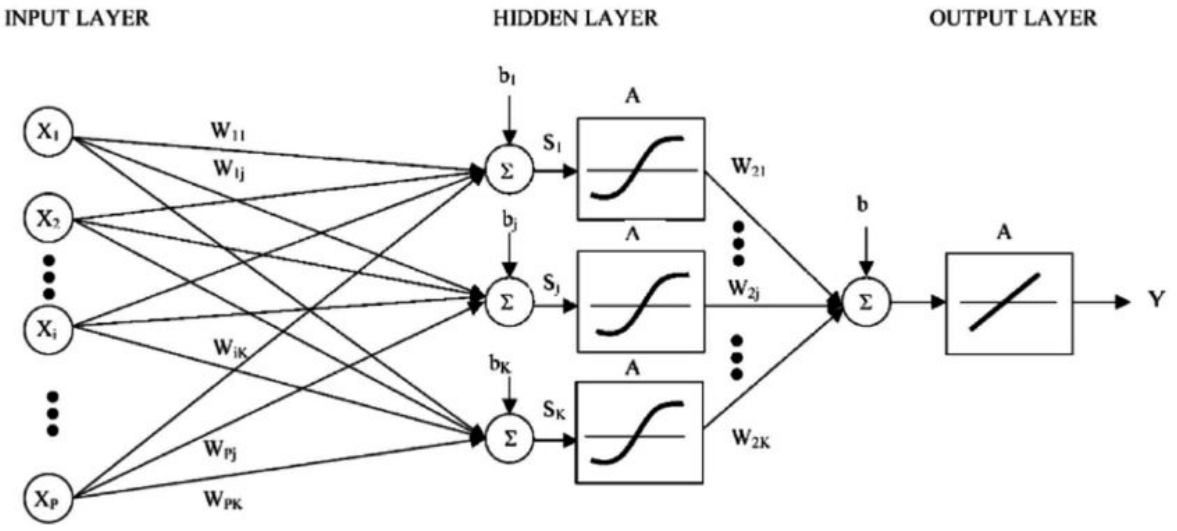

Εικόνα 3.4. Απεικόνιση ενός νευρώνα του δικτύου [8]

Ο τρόπος με τον οποίο οι νευρώνες επεξεργάζονται τα σήματα καθορίζεται από δύο συναρτήσεις [36]. Η *συνάρτηση ενεργοποίησης (activation function)* προσδιορίζει το συνολικό σήμα που λαμβάνεται σε ένα νευρώνα. Στο παράδειγμα της Εικόνας 5.2 η πληροφορία εισέρχεται στο δίκτυο από τις θέσεις εισόδου και καταλήγει σε ένα νευρώνα του δικτύου.. Στην περίπτωση που είχαμε $j$ νευρώνες (για $j$ = 1,..., K), με τη μορφή σημάτων $x_i$, ισχύς $w_{ij}$, η συνολική ενεργοποίηση $I_i(\mathbf{x})$ είναι της μορφής:

$$I_i(\boldsymbol{x}) = \sum_{j=0}^{K} w_{ij} \cdot x_j \quad (3.5)$$

όπου $w_{0j} = \theta$ είναι το κατώφλι (threshold) και $x_0 = 1$.

8 https://commons.wikimedia.org/wiki/File:ArtificialNeuronModel_english.png#/media/File:ArtificialNeuronModel_english.png




Εικόνα 3.5. Συνάρτηση ενεργοποίησης στο στρώμα εξόδου ενός δικτύου [9]

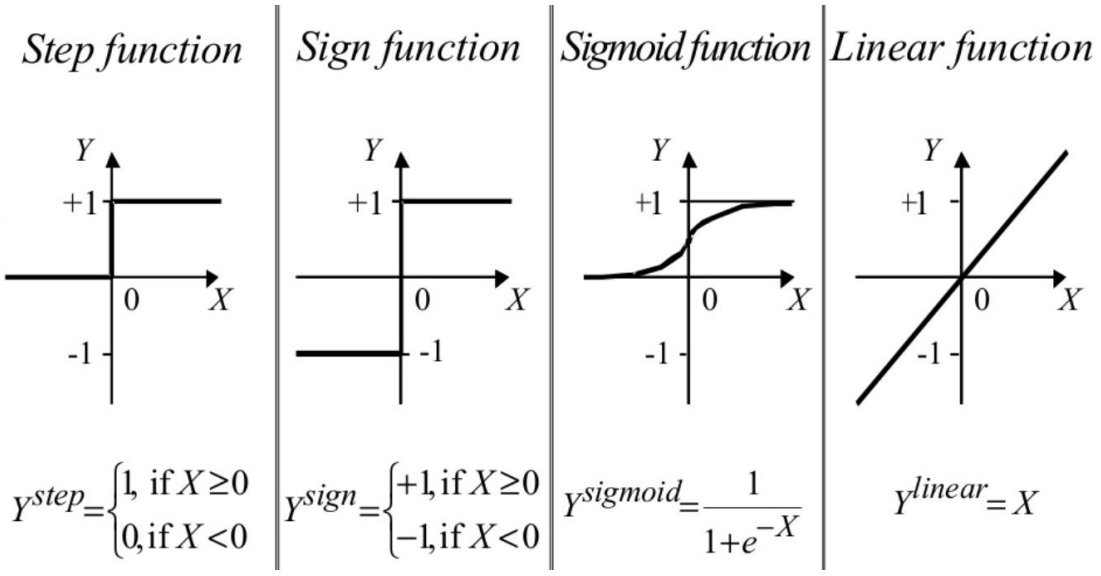
Στην Εικόνα 3.5 παρατηρούμε πως γενικεύεται η έννοια της συνολικής ενεργοποίησης σε ένα δίκτυο που έχουμε πολλούς νευρώνες. Στο στρώμα εξόδου για παράδειγμα, καταλήγουν όλες οι πληροφορίες που έχουν ενεργοποιηθεί προηγουμένως, λόγω των συναρτήσεων ενεργοποίησης στο κρυφό στρώμα.

Η τιμή της συνάρτησης ενεργοποίησης είναι συνήθως βαθμωτό μέγεθος και τα δεδομένα που δέχεται είναι διανύσματα. Η *συνάρτηση εξόδου* $o_j$ δέχεται ως είσοδο την τιμή αυτή και συνήθως χρησιμοποιείται για να την περιορίσει εντός ενός συγκεκριμένου εύρους τιμών (κανονικοποίηση).

Οι δύο αυτές συναρτήσεις καθορίζουν την τιμή του σήματος που εξέρχεται από ένα νευρώνα και μαζί αποτελούν τη *συνάρτηση μεταφοράς o(I(***x***)*. Για ορισμένες συναρτήσεις μεταφοράς δεν υπάρχει φυσικός διαχωρισμός μεταξύ της συνάρτησης ενεργοποίησης και της συνάρτησης εξόδου. Στα περισσότερα νευρωνικά δίκτυα οι συναρτήσεις μεταφοράς που χρησιμοποιούνται στο στρώμα εισόδου και εξόδου είναι γραμμικές, ενώ στα κρυφά στρώματα συνήθως χρησιμοποιούνται μη γραμμικές συναρτήσεις μεταφοράς. Κάποιες από τις πιο συνηθισμένες συναρτήσεις ενεργοποίησης είναι οι εξής:

- Βηματική συνάρτηση (Step function)
- Συνάρτηση προσήμου (Sign function)
- Γραμμική συνάρτηση (Linear function)
- Σιγμοειδής συνάρτηση (Sigmoid function)

[9] [37]




Εικόνα 3.6. Γραφική αναπαράσταση συναρτήσεων ενεργοποίησης [10]

### 3.2.8. Εκπαίδευση ενός Τεχνητού Νευρωνικού Δικτύου

Όλοι οι παραπάνω ορισμοί χρησιμεύουν στην κατανόηση της διαδικασίας εκπαίδευσης ενός νευρωνικού δικτύου. Όπως αναφέραμε προηγουμένως, η εκπαίδευση ενός ANN είναι μία επαναληπτική διαδικασία. Στην αρχή, τα βάρη όλων των ακμών λαμβάνουν μία τυχαία τιμή και υπολογίζεται η τιμή της εξόδου για όλες τις παρατηρήσεις. Οι συναρτήσεις ενεργοποίησης επιλέγουν ποιες ακμές ενεργοποιούνται κάθε φορά και ποιες όχι, ενώ καθορίζουν τη μορφή που λαμβάνει η πληροφορία $x_i$ σε κάθε στάδιο. Μετά από κάθε επανάληψη (epoch), υπολογίζεται η τιμή της συνάρτησης κόστους, η οποία στη συνέχεια διαδίδεται πίσω στο δίκτυο (με τη χρήση του αλγορίθμου BP) για τον υπολογισμό του gradient. Επιλέγοντας το μέγεθος του batch, αποφασίζουμε για το πόσα δείγματα θα χρησιμοποιηθούν για να γίνει ο υπολογισμός του gradient. Τέλος, βάσει της τιμής του gradient, καθώς και της τιμής του ρυθμού εκμάθησης, ανανεώνονται τα βάρη των ακμών και η διαδικασία επαναλαμβάνεται από την αρχή μέχρι οι τιμές των μεταβλητών εξόδου να συγκλίνουν. Η σύγκλιση αυτή δε γίνεται πάντα προς το επιθυμητό αποτέλεσμα και πολύ συχνά χρειάζεται εκ νέου παραμετροποίηση του δικτύου.

## 3.3. Ανατροφοδοτούμενα Νευρωνικά Δίκτυα

Όπως είδαμε παραπάνω, τα τεχνητά νευρωνικά δίκτυα χρησιμοποιούνται για τον υπολογισμό της εξόδου *y*, βάσει της εισόδου *x* που δέχονται και των παραμέτρων του δικτύου. Σε πολλά όμως προβλήματα, η είσοδος μπορεί να είναι μία ακολουθία παρατηρήσεων, όπου η τιμή της κάθε παρατήρησης εξαρτάται από την τιμή της προηγούμενης. Το κλασικό μοντέλο ANN που μελετήσαμε δεν μπορεί να αντιμετωπίσει τέτοιου είδους προβλήματα, καθώς τα παραδείγματα που του εισάγουμε δεν μπορούν με κάποιο τρόπο να αλληλεπιδράσουν. Σε τέτοιες περιπτώσεις χρησιμοποιούμε τα *ανατροφοδοτούμενα νευρωνικά δίκτυα (Recurrent Neural Networks - RNN)*. Τα RNN είναι μία κατηγορία τεχνητών νευρωνικών δικτύων όπου οι συνδέσεις μεταξύ των κόμβων (νευρώνων του δικτύου) σχηματίζουν έναν κατευθυνόμενο γράφο κατά μήκος μιας ακολουθίας.

[10] https://www.slideshare.net/HouwLiongThe/neural-networks-29512097

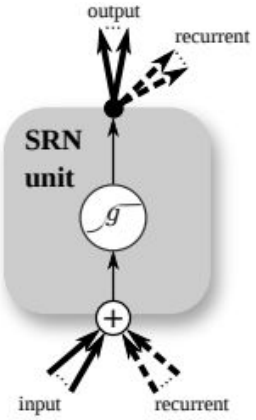
Σε ένα απλό RNN, δεδομένης μιας ακολουθίας εισόδου $\boldsymbol{x} = (x_1, \ldots, x_T)$, υπολογίζεται το κρυφό διάνυσμα $\boldsymbol{h} = (h_1, \ldots, h_T)$ και το διάνυσμα εξόδου $\boldsymbol{y} = (y_1, \ldots, y_T)$ επαναλαμβάνοντας κάθε φορά τις παρακάτω εξισώσεις από $t=1$ έως $t=T$. [38]

$$h_t = H\left( W_{xh} x_t + W_{hh} h_{t-1} + b_h \right) \quad (3.6)$$

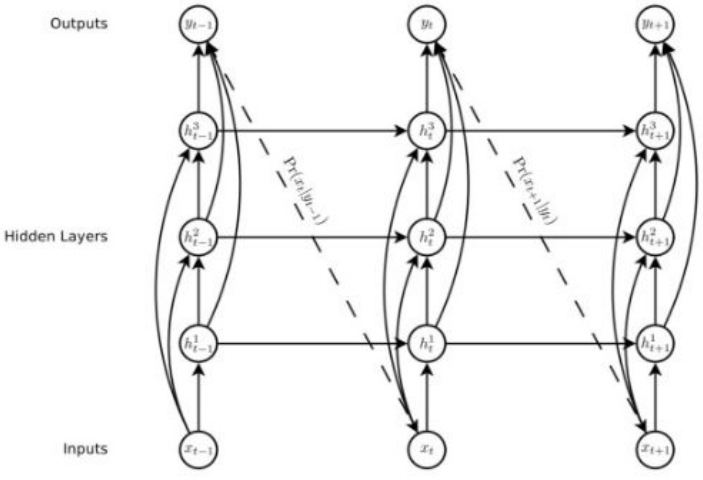
$$y_t = W_{hy} h_t + b_y \quad (3.7)$$

Όπου $W$, ο πίνακας που περιέχει τα αντίστοιχα βάρη (πχ. $W_{xh}$ είναι ο πίνακας εισόδου-κρυφών βαρών), το διάνυσμα $b$ είναι το διάνυσμα πόλωσης (πχ. $b_h$ είναι το διάνυσμα της κρυφής πόλωσης) και $H$ η συνάρτηση του κρυφού στρώματος.

Στην Εικόνα 3.7 παρατηρούμε την αρχιτεκτονική μιας μονάδας ενός απλού ανατροφοδοτούμενου δικτύου.




Εικόνα 3.7. Μονάδα απλού ανατροφοδοτούμενου δικτύου (simple recurrent network unit) [11]

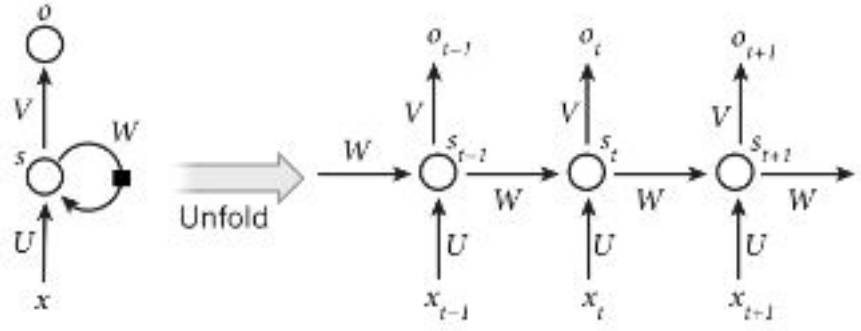



Εικόνα 3.8. Αρχιτεκτονική deep recurrent neural network. Οι κόμβοι αντιπροσωπεύουν τις υπολογιστικές μονάδες κάθε στρώματος, οι συνεχείς γραμμές τις ακμές με τα αντίστοιχα βάρη και οι διακεκομμένες γραμμές τις προβλέψεις [12]

[11] [39]

[12] https://adriancolyer.files.wordpress.com/2017/03/graves13-fig-1.jpeg?w=640

Η εκπαίδευση των RNN γίνεται όπως και προηγουμένως με τη χρήση μεθόδων πίσω διάδοσης. Σαν είσοδο εισάγουμε μία ακολουθία παρατηρήσεων και σε κάθε βήμα (time step) τα RNN διατηρούν στις υπολογιστικές μονάδες των κρυφών στρωμάτων ένα διάνυσμα κατάστασης (state vector), που περιέχει πληροφορίες σχετικά με όλα τα προηγούμενα στοιχεία της ακολουθίας.




Εικόνα 3.9. Ανατροφοδοτούμενο νευρωνικό δίκτυο σε απλή και ξεδιπλωμένη μορφή [13]

Στην Εικόνα 3.9 θεωρούμε δύο διαφορετικές μορφές ενός RNN, την απλή (αριστερά) και την ξεδιπλωμένη (δεξιά). Ο κόμβος *s* αντιπροσωπεύει το σύνολο των κρυφών υπολογιστικών μονάδων σε κάθε χρονική στιγμή *t*. Στην εικόνα αυτή μπορούμε να συμπεράνουμε ότι η πίσω διάδοση εκτελείται όπως και στα ANN, με μόνη διαφορά ότι στα RNN χρησιμοποιούμε μία παραλλαγή του αλγορίθμου BP, τον BPTT (Backpropagation through time), [40]. Μετα από κάθε επανάληψη του αλγορίθμου ανανεώνονται οι τιμές των πινάκων *(U, V, W),* που αντιστοιχούν στους πίνακες $(W_{xh}, W_{hy}, W_{hh})$ που ορίσαμε παραπάνω.

Στόχος λοιπόν του BPTT είναι η διάδοση του σφάλματος προς τα πίσω, για τον υπολογισμό του gradient. Παρόλο όμως που τα RNN φαίνεται να έχουν σπουδαία σημασία στην επίλυση προβλημάτων αναγνώρισης εικόνας, γραπτού λόγου και ήχου, το πρόβλημα της σταδιακής εξαφάνισης του gradient (*vanishing and exploding gradient)* ευθύνεται για τη μη συχνή χρήση τους. Σύμφωνα με αυτό, οι αναπαραστάσεις που σχετίζονται με το κοντινό παρελθόν επηρεάζουν σε μεγάλο βαθμό την έξοδο, ενώ οι αναπαραστάσεις που σχετίζονται με το μακρύ παρελθόν σταδιακά εξαφανίζονται και δεν παίζουν ρόλο στην έκβαση του αποτελέσματος. Παρακάτω θα δούμε πως επιδρά η εξαφάνιση του gradient στην τιμή των παραγώγων της συνάρτησης κόστους $C_t$ , σε μία στιγμή *t* σε ένα RNN με πίνακα βάρους *W*, και καταστάσεων *α* (*α* αντίστοιχο του *s* που είδαμε προηγουμένως).

$$\frac{\partial C_t}{\partial W} = \sum_{\tau \le t} \frac{\partial C_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W} = \sum_{\tau \le t} \frac{\partial C_t}{\partial a_t} \frac{\partial a_t}{\partial a_\tau} \frac{\partial a_\tau}{\partial W} \quad (3.8)$$

Έστω ότι βρισκόμαστε σε μία κατάσταση όπου το δίκτυο είναι σταθερά κλειδωμένο (robustly latched) και η παράγωγος της συνάρτησης κόστους δίνεται από την παραπάνω σχέση. Παρατηρούμε ότι για $\tau \ll t$,

$$\left| \frac{\partial C_t}{\partial a_t} \frac{\partial a_t}{\partial W} \right| \to 0 \quad (3.9)$$ [14]

[13] http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/
[14] [41]

Ο παραπάνω όρος είναι πολύ μικρός, συγκριτικά με τους όρους του αθροίσματος στους οποίους το *τ* βρίσκεται πολύ κοντά στο *t.* Αυτό πρακτικά σημαίνει ότι μία αλλαγή στον πίνακα βάρους θα επηρεάσει πολύ περισσότερο τις καταστάσεις του δικτύου κοντά στο *t.*

Η επίδραση της εξαφάνισης του gradient γίνεται αισθητή στα προβλήματα πρόβλεψης γραπτού λόγου. Για παράδειγμα έστω ότι έχουμε ένα κείμενο (μία ακολουθία από λέξεις) και κάθε φορά θέλουμε να προβλέψουμε ποια θα είναι η επόμενη λέξη που θα εισάγει ο χρήστης. Το δίκτυο λειτουργεί ως εξής:

$$\textit{Έχω γεννηθεί στην Ελλάδα. Η μητρική μου γλώσσα είναι τα } [\textit{Ελληνικά}]$$

Στόχος κάθε φορά είναι η πρόβλεψη της μητρικής γλώσσας του χρήστη, βάσει των πληροφοριών που εντοπίζονται στην ακολουθία των λέξεων. Έστω τώρα ότι ένας χρήστης εισάγει το παρακάτω κείμενο:

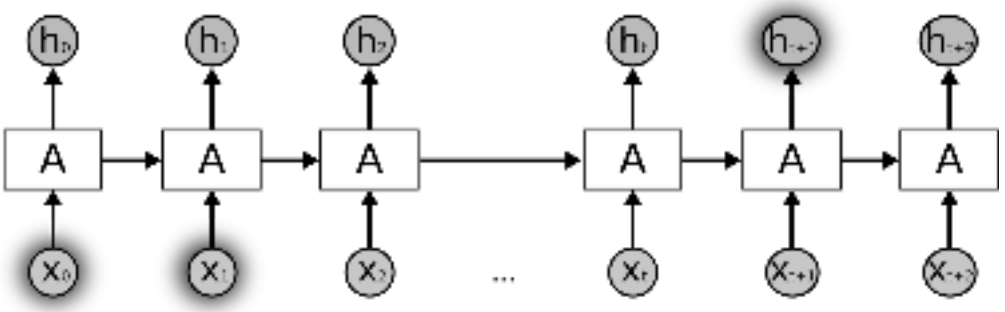
$$\textit{Έχω γεννηθεί στην Ελλάδα. Έχω ζήσει πολλά χρόνια στο Βέλγιο.}$$

$$\textit{Έχω ταξιδέψει στην Αγγλία και στη Γερμανία.}$$

$$\ldots\ldots$$

$$\textit{Η μητρική μου γλώσσα είναι τα } [\ldots]$$

Αυτή τη φορά το δίκτυο θα δυσκολευτεί να προβλέψει την επόμενη λέξη που θα εισάγει ο χρήστης, και κατ' επέκταση τη μητρική του γλώσσα. Αυτό συμβαίνει διότι η μνήμη του δικτύου εξασθενεί όσο εισέρχονται νέες πληροφορίες, με αποτέλεσμα οι πληροφορίες του μακρινού παρελθόντος να μην επηρεάζουν πλέον τις προβλέψεις που γίνονται. Το πρόβλημα αυτό αναπαρίσταται στην Εικόνα 3.10, όπου βλέπουμε ότι η κρυφή κατάσταση $h_{t+1}$ πλέον δε συνδέεται κάπως με τις εισόδους $x_0$ και $x_1$.




Εικόνα 3.10. Πρόβλημα διατήρησης μνήμης μακρινού παρελθόντος στα RNN [15]

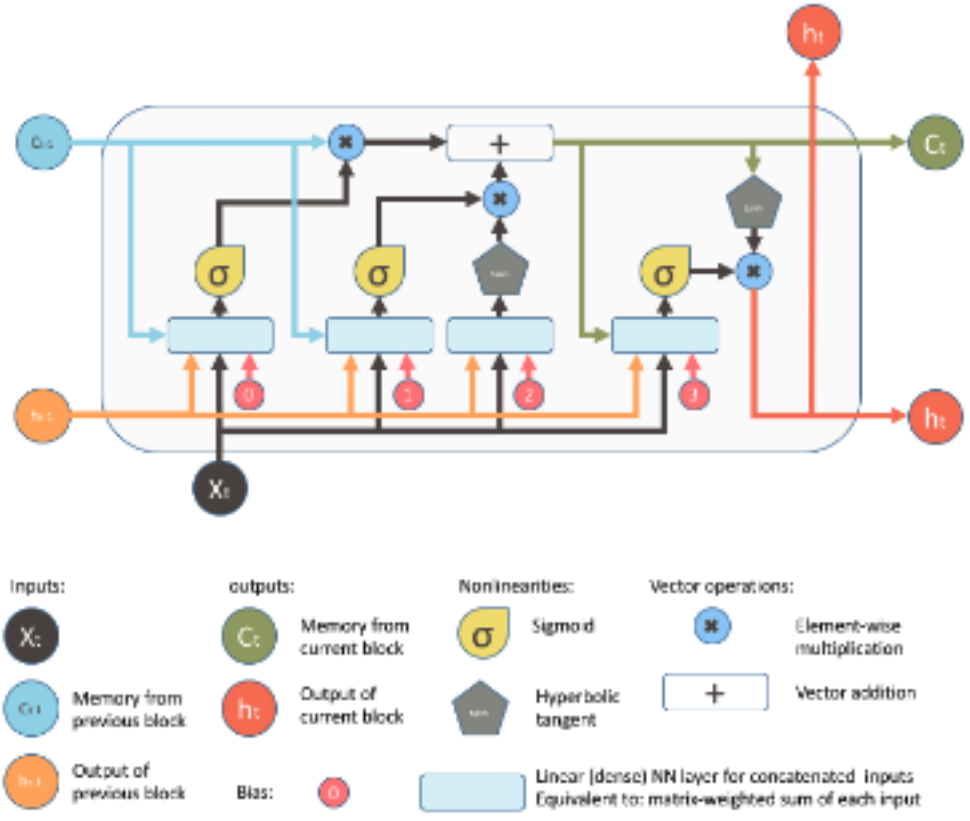
Η λύση στο πρόβλημα της εξαφάνισης του gradient δίνεται με τη χρήση ενός νέου, ανανεωμένου μοντέλου RNN, του LSTM.

[15] http://colah.github.io/posts/2015-08-Understanding-LSTMs

## 3.4. Δίκτυα Βραχείας και Μακράς Μνήμης

Τα δίκτυα βραχείας και μακράς μνήμης *(Long Short Term Memory - LSTM)* ανήκουν στην κατηγορία των ανατροφοδοτούμενων νευρωνικών δικτύων. Δέχονται δεδομένα με μορφή ακολουθίας, μπορούν να διαχειριστούν αναπαραστάσεις που σχετίζονται με το μακρινό παρελθόν και μπορούν να βελτιστοποιηθούν πολύ ευκολότερα απ' ότι τα απλά ανατροφοδοτούμενα δίκτυα. Για τους λόγους αυτούς, η χρήση τους είναι πολύ σημαντική στην επίλυση προβλημάτων, όπως η αναγνώριση γραφής, η μοντελοποίηση της γλώσσας και η μετάφρασή της και η ανάλυση οπτικοακουστικών δεδομένων. [39]

Η βασική ιδέα πίσω από την αρχιτεκτονική των LSTM είναι ένα κελί μνήμης, το οποίο μπορεί να διατηρεί την κατάστασή του με το πέρασμα του χρόνου, σε συνδυασμό με μη γραμμικές πύλες, οι οποίες ελέγχουν τη ροή της πληροφορίας από και προς το κελί αυτό. Οι σύγχρονες μελέτες προτείνουν διάφορες αλλαγές στη βασική αρχιτεκτονική των LSTM, που είχαν προτείνει οι Hochreiter και Schmidhuber [42], με στόχο την επίλυση όλο και πιο απαιτητικών προβλημάτων. Στην Εικόνα 3.11 φαίνεται ένα block του LSTM και στη συνέχεια αναλύεται ο τρόπος με τον οποίο λειτουργεί.




Εικόνα 3.11. Αρχιτεκτονική ενός LSTM block [16]

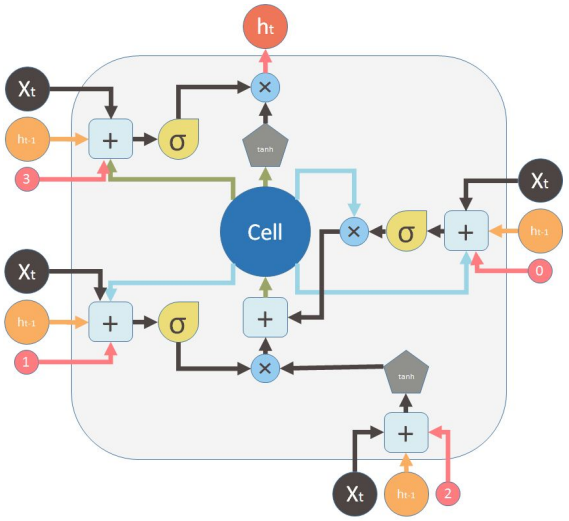
[16] https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

Σε κάθε block του LSTM υπάρχουν τρεις πύλες (gates), οι οποίες καθορίζουν μέσω της σιγμοειδούς συνάρτησης (*σ)* τι ποσοστό της πληροφορίας θα περάσει. Η συνάρτηση *σ* παίρνει τιμές ανάμεσα στο 0 και στο 1.

- **forget gate**
  Καθορίζει τι ποσοστό της προηγούμενης μνήμης θα περάσει στην τωρινή μνήμη. Ελέγχεται από ένα απλό ANN ενός στρώματος, του οποίου οι είσοδοι είναι η έξοδος $h_{t-1}$ και η μνήμη $C_{t-1}$ του προηγούμενου LSTM block, η είσοδος $x_t$ του τωρινού LSTM block και ένα διάνυσμα πόλωσης $b_0$. Η σιγμοειδής συνάρτηση χρησιμοποιείται ως συνάρτηση ενεργοποίησης του ANN, και η έξοδος πολλαπλασιάζεται με τη μνήμη $C_{t-1}$ του προηγούμενου block.

- **external input gate**
  Καθορίζει τι ποσοστό της νέας πληροφορίας θα περάσει στη νέα μνήμη. Μέσω ενός απλού ANN που δέχεται ως εισόδους την έξοδο $h_{t-1}$ του προηγούμενου block και την είσοδο $x_t$ του τωρινού block, και με συνάρτηση ενεργοποίησης την υπερβολική εφαπτομένη, δημιουργείται ένα νέο κομμάτι μνήμης βάσει της εισόδου $x_t$. Μία καινούρια πύλη (ANN αντίστοιχο του προηγουμένου) ελέγχει τι ποσοστό της νέας αυτής μνήμης θα προστεθεί στην τωρινή μνήμη του block που υπολογίστηκε παραπάνω. Συνεπώς, έχουμε κατασκευάσει πλέον το διάνυσμα της νέας μνήμης $C_t$, το οποίο προκύπτει από το συνδυασμό της προηγούμενης μνήμης και της εισόδου του LSTM.

- **output gate**
  Καθορίζει τι ποσοστό της τωρινής μνήμης περνάει στην έξοδο $h_t$. Ελέγχεται από ένα απλό ANN, του οποίο οι είσοδοι είναι η έξοδος $h_{t-1}$ του προηγούμενου block, η είσοδος $x_t$ του τωρινού block, η νέα μνήμη $C_t$ που υπολογίστηκε παραπάνω και ένα διάνυσμα πόλωσης $b_3$. Η πύλη αυτή καθορίζει τι ποσοστό της νέας μνήμης εξάγεται στην επόμενη υπολογιστική μονάδα του LSTM.




Εικόνα 3.12. Αρχιτεκτονική ενός LSTM block και κελί μνήμης [17]

[17] https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

Πολλές φορές στη βιβλιογραφία χρησιμοποιείται ο όρος *κελί μνήμης (memory cell) ενός LSTM*. Στην Εικόνα 3.11 είδαμε την αρχιτεκτονική ενός block LSTM, και αναφερθήκαμε στους όρους $C_t$ και $C_{t-1}$ που αντιπροσωπεύουν το περιεχόμενο της μνήμης σε κάθε χρονική στιγμή. Οι δύο αυτοί όροι συνοψίζονται με τη μορφή ενός κελιού μνήμης $C$ το οποίο θεωρούμε ότι περιέχει όλες τις πληροφορίες σχετικά με τη μνήμη ενός block. Η αντίστοιχη αρχιτεκτονική ενός block LSTM με το κελί μνήμης φαίνεται στην Εικόνα 3.12. Ένα απλό LSTM δίκτυο έχει τη μορφή της Εικόνας 3.10, όπου τα blocks συνδέονται σειριακά. Με αυτόν τον τρόπο το δίκτυο πλέον έχει τη δυνατότητα στην κατάσταση $h_{t+1}$ να περιέχει στη μνήμη πληροφορίες σχετικές με τις εισόδους $x_0$ και $x_1$.

# 4. Μεγάλες Εκδηλώσεις σε Έξυπνες Πόλεις

## 4.1. Έξυπνες Πόλεις

Μία *έξυπνη πόλη (smart city)* είναι μία αστική περιοχή, εντός της οποίας χρησιμοποιούνται αισθητήρες για τη συλλογή δεδομένων. Τα δεδομένα αυτά χρησιμοποιούνται έτσι ώστε να γίνεται καλύτερη αξιοποίηση των πόρων της πόλης, και κατ' επέκταση, καλύτερη εξυπηρέτηση του κοινού. Σε μία έξυπνη πόλη, πολλές συσκευές είναι συνδεδεμένες σε ένα IoT δίκτυο [43], έτσι ώστε να μπορούν να ανταλλάσουν πληροφορίες, τόσο με αυτό, όσο και μεταξύ τους. Για παράδειγμα, η έξυπνη πόλη του Κάνσας, περιλαμβάνει έξυπνο φωτισμό, διαδραστικά κιόσκια και περιοχές δωρεάν Wi-Fi. Μέσω μιας εφαρμογής, οι πολίτες έχουν πρόσβαση σε διάφορες πληροφορίες, όπως για παράδειγμα ελεύθερες θέσεις πάρκινγκ, ροή της κίνησης εντός της πόλης, ακόμα και σημεία του δρόμου όπου υπάρχει διάβαση των πεζών [44]. Στην Εικόνα 4.1 μπορούμε να δούμε κάποιες από τις έξυπνες υπηρεσίες, που εμφανίζονται σε μια έξυπνη πόλη.


Εικόνα 4.1 Έξυπνες υπηρεσίες σε μία έξυπνη πόλη [18]

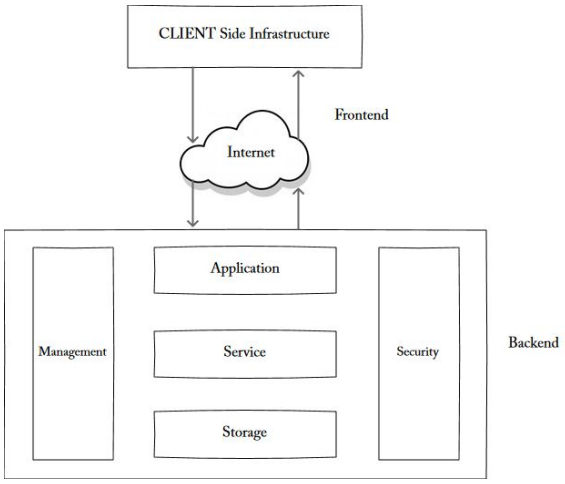
## 4.2. Μεγάλες Εκδηλώσεις

Οι *μεγάλες εκδηλώσεις (Large Events)* εντός των έξυπνων πόλεων, είναι ένα κεφάλαιο το οποίο απασχολεί σε μεγάλο βαθμό τους ερευνητές τα τελευταία χρόνια. Οι εκδηλώσεις αυτές, αφορούν τη συγκέντρωση κόσμου σε μία συγκεκριμένη περιοχή, με σκοπό την παρακολούθηση ενός μουσικού φεστιβάλ, ενός αθλητικού αγώνα, μίας παρέλασης, ή ακόμα και μιας ομιλίας. Οι διοργανωτές των εκδηλώσεων αυτών, είναι υπεύθυνοι για την ομαλή τους λειτουργία αλλά και για την ασφάλεια του κόσμου. Επίσης, είναι απαραίτητη η παροχή διαφόρων υπηρεσιών, για την καλύτερη εξυπηρέτηση του κοινού.

[18] http://scitechconnect.elsevier.com/elseviers-new-smart-cities-book-series-launched

## 4.3. Υπολογιστικό Νέφος - Υπολογιστική Ομίχλη

*Υπολογιστικό νέφος (cloud computing)* ονομάζεται η κατά παραγγελία παροχή υπολογιστικών (και όχι μόνο) πόρων, μέσω του διαδικτύου [45]. Στην εποχή μας, η ανάγκη για αποθήκευση όλο και περισσότερων πληροφοριών, μας οδηγεί στη δημιουργία απομακρυσμένων χώρων αποθήκευσης, όπου οι πληροφορίες διατηρούνται με ασφάλεια. Για παράδειγμα, οι διάφορες δημόσιες και ιδιωτικές υπηρεσίες, έχουν ανάγκη από συνεχή πρόσβαση σε πολύ μεγάλες βάσεις δεδομένων. Αν τα δεδομένα αυτά ήταν αποθηκευμένα σε κάποια τοπική μονάδα, το κόστος θα ήταν πολύ μεγάλο, ενώ σε περίπτωση βλάβης (πχ. διακοπή ρεύματος) η πρόσβαση σε αυτά θα ήταν αδύνατη. Για το λόγο αυτό, τα δεδομένα αποθηκεύονται με ασφάλεια στο cloud,όπου η πρόσβασή είναι εύκολη μέσω του διαδικτύου. Παρόλ' αυτά, στο cloud δε γίνεται μόνο αποθήκευση των δεδομένων. Καθώς ο όγκος των δεδομένων αυξάνεται, αυξάνεται και η επεξεργαστική ισχύς που χρειάζεται για την επεξεργασία τους. Πλέον, η επεξεργασία αυτή γίνεται απομακρυσμένα (εντός του cloud), οπότε απελευθερώνονται όλο και περισσότεροι πόροι από την πλευρά του πελάτη (client), για τη διαχείριση των τοπικών υπολογιστικών προβλημάτων. Στην Εικόνα 4.2 απεικονίζεται η σχέση μεταξύ του client και του cloud.

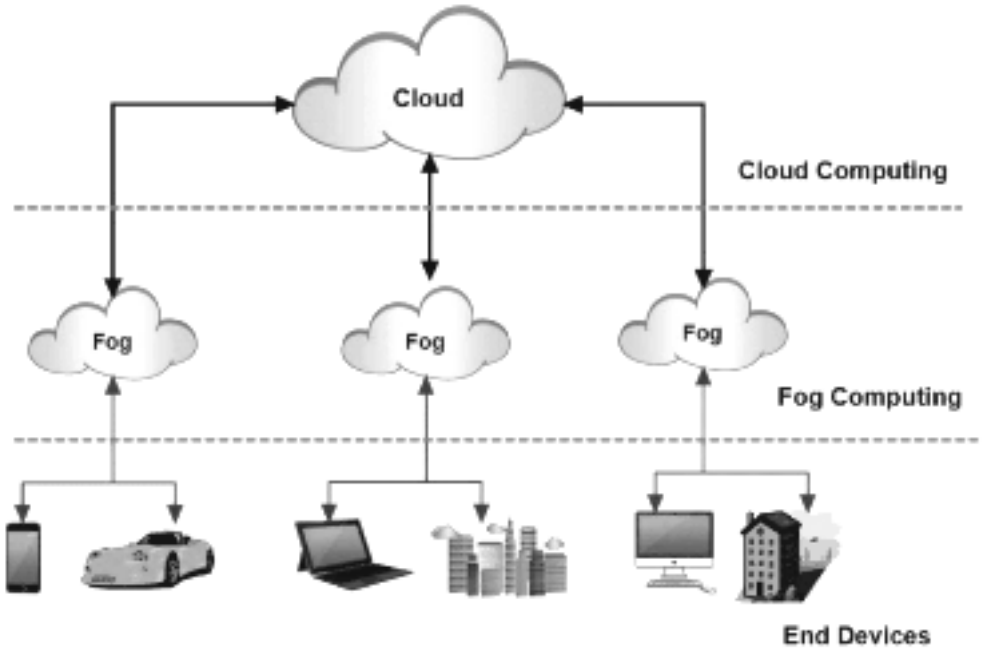



Εικόνα 4.2. Αρχιτεκτονική cloud computing [19]

Παρατηρούμε, ότι ο client ανταλλάζει πληροφορίες με το cloud (backend) μέσω του διαδικτύου. Το cloud είναι υπεύθυνο για την αποθήκευση, διαχείριση και προστασία όλων αυτών των πληροφοριών. Με τον όρο cloud computing, αναφερόμαστε στις αρχιτεκτονικές υπολογιστικές μονάδες/πόρους που περιέχονται εντός του cloud.

Πολλές φορές, η επεξεργασία των δεδομένων θέλουμε να γίνεται γρήγορα και με αποτελεσματικό τρόπο. Για το λόγο αυτό, εισάγουμε την έννοια της *υπολογιστικής ομίχλης (fog computing)*. Σύμφωνα με την αρχιτεκτονική αυτή, ορισμένες διεργασίες γίνονται στην άκρη (at the edge) [46] του δικτύου (τοπικά), σε μία έξυπνη συσκευή, ενώ άλλες διεργασίες ελέγχονται από το cloud. Με τον τρόπο αυτό, έχουμε ένα ενδιάμεσο στρώμα, ανάμεσα στις τοπικές συσκευές και το cloud, έτσι ώστε να πετυχαίνουμε αποτελεσματικότερη αποθήκευση, επεξεργασία και ανάλυση των δεδομένων. Συνεπώς, βάσει του φόρτου εργασίας, μπορούμε να κρίνουμε αν οι τοπικοί πόροι επαρκούν για την επεξεργασία των δεδομένων, ή αν η επεξεργασία αυτή πρέπει να γίνει στο cloud. Στην Εικόνα 4.3. απεικονίζεται η σχέση μεταξύ cloud και fog computing.

[19] https://www.w3schools.in/cloud-computing/cloud-computing-architecture/




Εικόνα 4.3. Cloud και fog computing [20]

## 4.4. Υλοποίηση Αρχιτεκτονικής Ομίχλης για Μεγάλες Εκδηλώσεις

Από τα παραπάνω, παρατηρούμε ότι οι αρχιτεκτονικές ομίχλης μπορούν να αξιοποιηθούν σε μεγάλο βαθμό εντός των smart cities. Πιο συγκεκριμένα, προτείνουμε μια fog αρχιτεκτονική για την υποστήριξη μεγάλων εκδηλώσεων [48]. Η εκδήλωση που μελετάμε αφορά ένα μουσικό φεστιβάλ στην Καρλσρούη της Γερμανίας. Η αρχιτεκτονική αυτή σχεδιάζεται, έτσι ώστε να περιέχει τις εξής ιδιότητες:

- αξιοπιστία και χαμηλό κόστος λειτουργιών
- το φόρτο εργασία κατανέμεται αυτομάτως στους πόρους, εύκολα, γρήγορα και με ασφάλεια

Όπως φαίνεται στην Εικόνα 4.4, η αρχιτεκτονική που υλοποιήσαμε απαρτίζεται από τις εξής μονάδες:

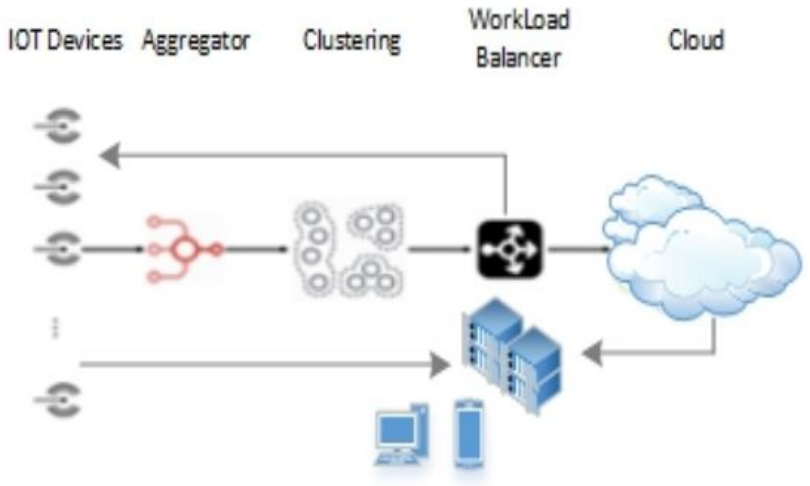
- IOT συσκευές
  Οι συσκευές αυτές τοποθετούνται σε όλη την έκταση του χώρου της εκδήλωσης που μελετάμε, και περιέχουν έναν αισθητήρα ισχύς σήματος (RSSI sensor), συνδεδεμένο σε ένα raspberry pi.

- Aggregator
  Στη μονάδα αυτή, συλλέγονται τα δεδομένα που λαμβάνονται από τους αισθητήρες, και φιλτράρονται, έτσι ώστε να διατηρηθεί η ανωνυμία της προέλευσής τους. Το κομμάτι αυτό είναι πολύ σημαντικό, καθώς μας επιτρέπει να συλλέγουμε δεδομένα, χωρίς να παραβιάζεται η ιδιωτικότητα των χρηστών.

- Clustering
  Στη μονάδα αυτή, γίνεται ομαδοποίηση των δεδομένων που προέρχονται από τον aggregator. Τα δεδομένα ομαδοποιούνται σε mini-batches, βάσει των ψευδοανώνυμων id που τους έχουν δοθεί.

[20] [47]

- WorkLoad Balancer
  Ο WorkLoad Balancer [49] αποτελεί την πιο σημαντική μονάδα της αρχιτεκτονικής. Κατανέμει το φόρτο εργασίας, τα δεδομένα, καθώς και τις διαθέσιμες υπηρεσίες, είτε στις τοπικές υπολογιστικές μονάδες, είτε στο cloud, βάσει της πρόβλεψης της κατανομής του κόσμου στο χώρο της εκδήλωσης. Η πρόβλεψη αυτή γίνεται με τη χρήση τεχνικών μηχανικής μάθησης [50], και παρουσιάζεται αναλυτικά στο επόμενο κεφάλαιο.




Εικόνα 4.4. Fog αρχιτεκτονική για μεγάλες εκδηλώσεις

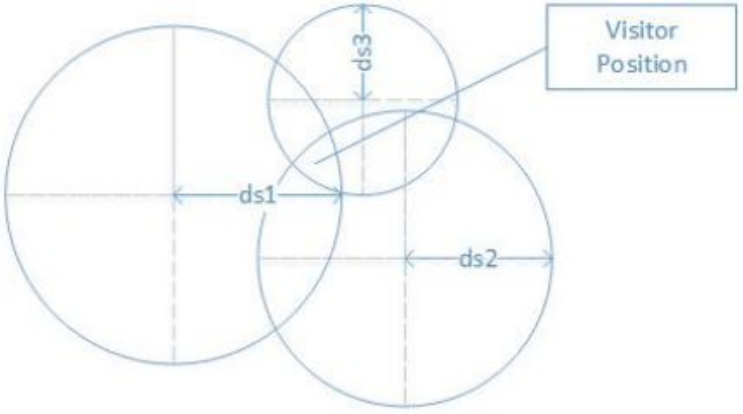
Οι υπηρεσίες που αναπτύσσονται δυναμικά στο fog, είναι η υπηρεσία Trilateration, καθώς και οι backend υπηρεσίες μιας εφαρμογής κινητής τηλεφωνίας, στην οποία είχαν πρόσβαση οι επισκέπτες του φεστιβάλ. Ο WorkLoad Balancer παρακολουθεί τη λειτουργία των υπηρεσιών αυτών, και βάσει της πρόβλεψης της κατανομής του κόσμου είτε θέτει σε λειτουργία, είτε απελευθερώνει τα εικονικά μηχανήματα που βρίσκονται στο Cloud [51], βάσει της παρακάτω εξίσωσης.

$$WB(us, vm) = \begin{cases} Request\ VM & us_+ \ \&\ vm_{high} \\ No\ Action & us_+ \ \&\ vm_{low} \\ No\ Action & us\_ \ \&\ vm_{high} \\ Free\ VM & us\_ \ \&\ vm_{low} \end{cases} \quad (4.1)$$

Στην παραπάνω σχέση, με $us_+$ συμβολίζουμε την αύξηση στη συγκέντρωση του κόσμου στο χώρο του φεστιβάλ, με $us\_$ τη μείωση, ενώ με $vm_{high}$ και $vm_{low}$ συμβολίζουμε το υψηλό και χαμηλό επίπεδο εκμετάλλευσης των υπολογιστικών μονάδων (είτε τοπικών, είτε του Cloud) αντίστοιχα. Η πολιτική, βάσει της οποίας γίνεται η κατανομή του φόρτου εργασίας, δίνει προτεραιότητα στην ανάπτυξη τοπικών διεργασιών (at the edge), καθώς έτσι μειώνεται ο χρόνος απόκρισης (latency), η σύνδεση μεταξύ των συσκευών είναι συμπαγής, διατηρείται η ιδιωτικότητα των δεδομένων, ενώ παράλληλα μειώνεται το οικονομικό κόστος που προκύπτει από τη χρήση των πόρων του Cloud. Συνεπώς, όταν το φόρτο εργασίας είναι μικρό, ο WorkLoad Balancer θεωρεί ότι η επεξεργαστική ισχύς των τοπικών

συσκευών στα άκρα του δικτύου (Edge devices), επαρκεί για τη διαχείρισή του.

Οι Edge devices έχουν δύο λειτουργίες στην παραπάνω fog αρχιτεκτονική. Αρχικά, δρουν σαν αισθητήρες, ανιχνεύοντας το σήμα που εκπέμπουν τα κινητά τηλέφωνα, και έπειτα διαθέτουν επεξεργαστική ισχύ και μνήμη για την επεξεργασία των δεδομένων. Οι IoT αισθητήρες βρίσκονται σε γνωστές συντεταγμένες, σε διάφορα σημεία εντός του χώρου του φεστιβάλ, και λειτουργούν ως παροχείς WiFi. Καταγράφουν την ταυτότητα των χρηστών με τη μορφή διευθύνσεων MAC, τα IoT ids, καθώς και την ισχύ των εκπεμπόμενων σημάτων (RSSI) σε dB. Βάσει της ισχύς αυτής, υπολογίζεται η απόσταση του χρήστη από τους αισθητήρες και με τη διαδικασία του Trilateration υπολογίζονται οι συντεταγμένες του. Βάσει της διαδικασίας αυτής, γνωρίζοντας την απόσταση ενός σημείου από τρία διαφορετικά - γνωστά σημεία, μπορούμε να υπολογίσουμε τις συντεταγμένες του.




Εικόνα 4.5. Εκτίμηση της θέσης του χρήστη με τη διαδικασία Trilateration

# 5. Πειραματικό Κομμάτι - Πρόβλεψη Κατανομής Πληθυσμού με Χρήση Τεχνικών Μηχανικής Μάθησης

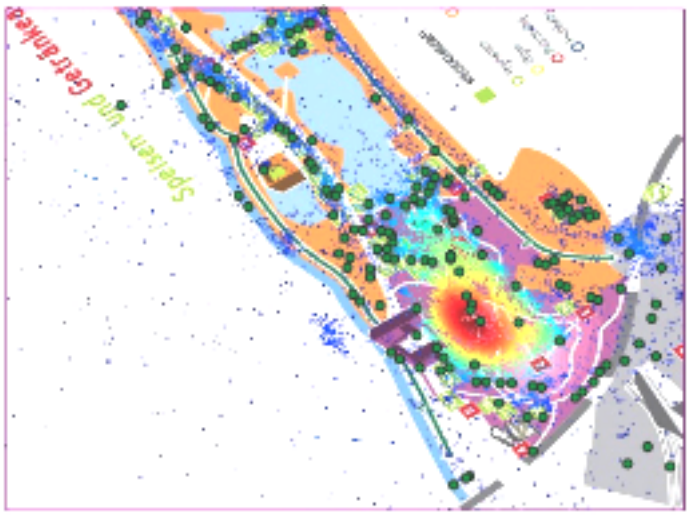
## 5.1. Περίληψη του Προβλήματος

Όπως αναφέραμε στην εισαγωγή, οι τεχνικές μηχανικής μάθησης έχουν ιδιαίτερη σημασία στη σύγχρονη καθημερινότητα. Η ικανότητα πρόβλεψης μελλοντικών καταστάσεων σε συνδυασμό με τη στατιστική ανάλυση των αποτελεσμάτων που προκύπτουν, μας προσφέρει την ικανότητα επίλυσης πολλών προβλημάτων, με στόχο την καλύτερη εξυπηρέτηση του κοινού. Μία ιδιαίτερη κατηγορία τέτοιου είδους προβλημάτων σχετίζονται με την ανάλυση δεδομένων, σε μέρη όπου παρατηρείται μεγάλος συνωστισμός κόσμου. Στα μεγάλα εμπορικά κέντρα, στα διάφορα φεστιβάλ, στα αεροδρόμια, ακόμα και σε δημόσιους χώρους, η μελέτη της συμπεριφοράς του κοινού αλλά και ο τρόπος με τον οποίο ο κόσμος κατανέμεται στο χώρο, έχει ιδιαίτερη σημασία τόσο για την καλύτερη εξυπηρέτησή του, όσο και για τη λήψη μέτρων ασφαλείας. Ένας μεγάλος κλάδος της τεχνητής νοημοσύνης ασχολείται με την αναγνώριση μοτίβων στις ανθρώπινες δραστηριότητες, τόσο μέσα από τα δεδομένα που λαμβάνονται από τις κάμερες ασφαλείας [52], όσο και από δεδομένα τοποθεσίας που λαμβάνονται μέσω των κινητών τηλεφώνων [53].

Στο πειραματικό μέρος αυτής της εργασίας θα μελετήσουμε πώς με τη χρήση δεδομένων χωρικών συντεταγμένων, μπορούμε να προβλέψουμε την κατανομή του κόσμου στο χώρο ενός φεστιβάλ. Το φεστιβάλ που μελετήσαμε είναι ανοιχτού χώρου, ονομάζεται *Das Fest* και λαμβάνει χώρα στην Καρλσρούη της Γερμανίας. Τα δεδομένα μας αφορούν τις χρονιές 2017 και 2018 και προέρχονται από μία υπηρεσία του Basmati [46], η οποία προσέφερε στο κοινό μία εφαρμογή κινητής τηλεφωνίας που περιείχε πληροφορίας σχετικά με το φεστιβάλ [54]. Κατά τη χρήση της εφαρμογής, η υπηρεσία κατέγραφε την τοποθεσία του χρήστη (geotag), την ταυτότητά του (με η μορφή ακολουθίας ψηφίων/χαρακτήρων), καθώς επίσης και τη χρονική στιγμή κατά την οποία λήφθηκε η πληροφορία αυτή. Τα δεδομένα ήταν σε ακατέργαστη μορφή, οπότε βασικός στόχος ήταν η μετατροπή τους σε μορφή κατάλληλη προς επεξεργασία. Επιπλέον, μας δόθηκαν οι συντεταγμένες του χώρου στον οποίο εκτείνεται το φεστιβάλ, αλλά και οι συντεταγμένες διαφόρων σημείων ενδιαφέροντος *(Points Of Interest - POIs)* εντός του χώρου αυτού, όπως για παράδειγμα οι εξέδρες των συναυλιακών χώρων, τα μικρά καταστήματα, οι καντίνες και οι αίθουσες ανάπαυσης. Στην Εικόνα 5.1 παρουσιάζεται ο χάρτης του φεστιβάλ, όπως δόθηκε στο κοινό, προσανατολισμένος όμως βάσει του παγκόσμιου χάρτη.

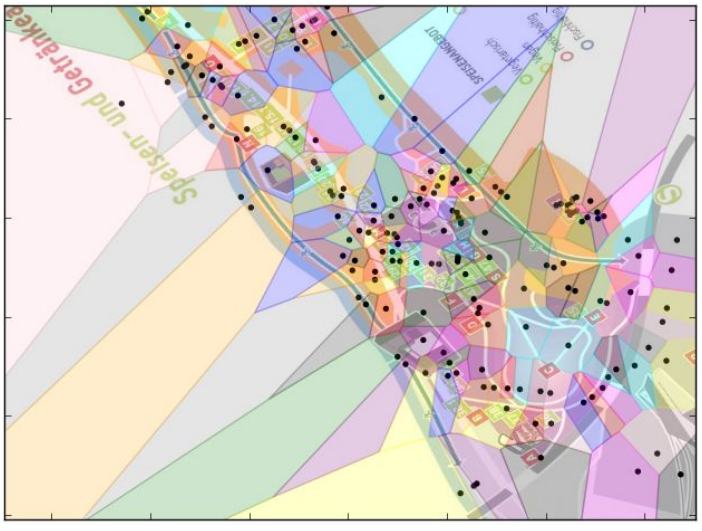
Εικόνα 5.1. Προσανατολισμένος χάρτης του φεστιβάλ

Η περίοδος που έλαβε χώρα το φεστιβάλ είναι μεταξύ 21 και 23 Ιουλίου (για το 2017). Όσες παρατηρήσεις ήταν εκτός αυτής της περιόδου αφαιρέθηκαν, καθώς επίσης αφαιρέθηκαν και όσα δεδομένα προέρχονταν από τοποθεσίες εκτός του φεστιβάλ.



Εικόνα 5.2. Γραφική απεικόνιση του χώρου του φεστιβάλ, των σημείων ενδιαφέροντος (πράσινο χρώμα) και του κοινού (users' heatmap)

Τα δεδομένα που είχαμε απεικονίζονται στην Εικόνα 5.2. Μπορούμε να δούμε σε ποια σημεία του φεστιβάλ παρατηρείται μεγαλύτερη συγκέντρωση κόσμου, καθώς και που βρίσκονται τα διάφορα σημεία ενδιαφέροντος. Στη συνέχεια, δημιουργήσαμε το διάγραμμα Voronoi βάσει των POIs πάνω στο χάρτη. Στην Εικόνα 5.3 παρατηρούμε τις περιοχές που προκύπτουν σύμφωνα με το διάγραμμα αυτό.

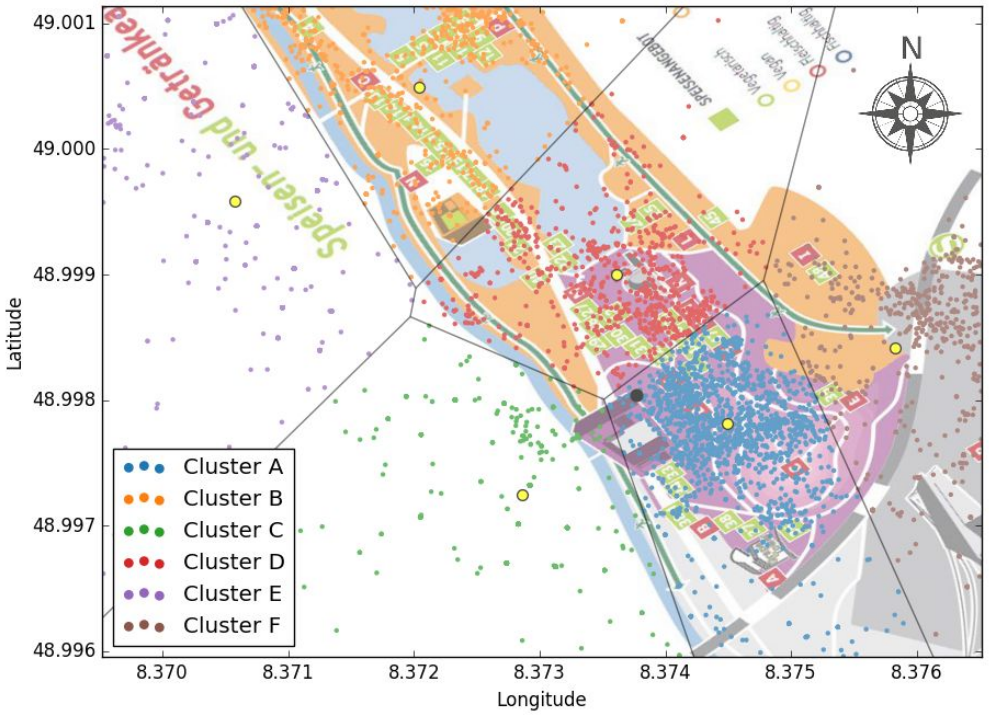


Εικόνα 5.3. Διάγραμμα Voronoi βάσει των POIs του φεστιβάλ. Στην εικόνα απεικονίζονται οι περιοχές ενδιαφέροντος

Κάθε περιοχή του χάρτη αντιστοιχίζεται σε καθένα από τα POIs. Κάθε σημείο των περιοχών αυτών βρίσκεται πιο κοντά στο POI της περιοχής, απ' ότι σε οποιοδήποτε άλλο POI του χάρτη. Έτσι μπορούμε να θεωρήσουμε με μαθηματικό τρόπο την περιοχή που καταλαμβάνει το κάθε POI. Η θεώρηση όμως αυτή δεν ανταποκρίνεται στην πραγματικότητα, καθώς δε λαμβάνει υπόψη το μέγεθος των POIs, καθώς και τη σημασία τους. Στην Εικόνα 5.3 ο συναυλιακός χώρος δεν αναπαρίσταται με σαφήνεια, καθώς τα διάφορα POIs γύρω από αυτόν θεωρούνται "ισάξιας γεωγραφικής σημασίας".

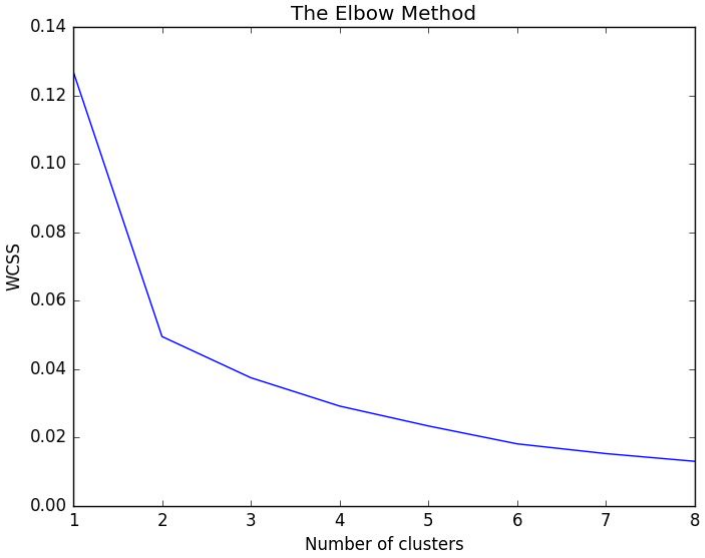
## 5.2. Συσταδοποίηση

Χωρίς να είναι εκ των προτέρων γνωστά τα όρια των περιοχών των POIs, και δεδομένου ότι το διάγραμμα Voronoi δε μας δίνει μία αντιπροσωπευτική αναπαράστασή τους, θα έπρεπε να οριστούν αυθαίρετα. Αυτό είναι πολύ δύσκολο για κάποιον ο οποίος δεν έχει επισκεφτεί το φεστιβάλ και δε γνωρίζει πως η εικόνα που έχουμε αντιστοιχίζεται στην πραγματικότητα. Παρόλ' αυτά, στην περίπτωση που είχαμε τη γνώση αυτή και επιλέγαμε να χωρίσουμε αυθαίρετα το χάρτη σε περιοχές, η επιλογή αυτή θα καθιστούσε τα συμπεράσματά μας υποκειμενικά. Αποφασίσαμε έτσι να λύσουμε το πρόβλημα του διαχωρισμού του χάρτη σε περιοχές ενδιαφέροντος *(Areas Of Interest - AOIs)* με τη χρήση τεχνικών συσταδοποιησης, βάσει του συνολικού κόσμου που παρακολούθησε το φεστιβάλ. Έτσι, τα AEIs αντιπροσωπεύουν τους χώρους στους οποίους κινείται ο κόσμος, και όχι τα σημεία ενδιαφέροντος του φεστιβάλ.




Εικόνα 5.4. Χάρτης του φεστιβάλ χωρισμένος σε clusters - AOIs, με τη χρήση του αλγορίθμου k-means. Με κίτρινο χρώμα απεικονίζονται τα κέντρα των AOIs

Στην Εικόνα 5.4 παρατηρούμε το χώρο του φεστιβάλ χωρισμένο σε έξι clusters, βάσει του αλγορίθμου k-means. Οι άξονες μας δείχνουν τις συντεταγμένες όλων των σημείων πάνω στο χάρτη. Ο αριθμός των clusters επιλέχθηκε με τη χρήση της μεθόδου του αγκώνα, όπως φαίνεται στην Εικόνα 5.5.




Εικόνα 5.5. Χρήση της μεθόδου του αγκώνα για την εύρεση του αριθμού των clusters

Παρόλο που τα AOIs αυτή τη φορά προκύπτουν από τα δεδομένα που είχαμε για τον κόσμο του φεστιβάλ, παρατηρούμε ότι έχουν ιδιαίτερη φυσική σημασία, αντίστοιχη των POIs.

- Cluster A: Κεντρική σκηνή του φεστιβάλ. Στην περιοχή αυτή βρίσκεται ο κόσμος που παρακολουθεί το φεστιβάλ.

- Cluster B: Στην περιοχή αυτή ανήκει η εξέδρα του DJ, ενώ γεωγραφικά αποτελεί το βορειότερο κομμάτι του χώρου του φεστιβάλ.

- Cluster E, C: Στις περιοχές αυτές ίσως ανήκει ο κόσμος που αναζητά την είσοδο του φεστιβάλ. Όπως προκύπτει από το χάρτη, οι περιοχές αυτές αν και φαίνεται να βρίσκονται εντός του χώρου του φεστιβάλ, είναι αρκετά αποκομμένες από όλα τα σημεία ενδιαφέροντος.

- Cluster D: Στην περιοχή αυτή ίσως ανήκει ένα κομμάτι του κόσμου που παρακολουθεί την κεντρική σκηνή του φεστιβάλ, ή και ο κόσμος που αποχωρεί από την περιοχή αυτή για να περιηγηθεί στον υπόλοιπο χώρο του φεστιβάλ. Επίσης, στην περιοχή αυτή βρίσκονται αρκετά μικρά καταστήματα, βάσει των δεδομένων που έχουμε για την υποδομή του φεστιβάλ.

- Cluster F: Είσοδος/Εξοδος φεστιβάλ. Στην περιοχή αυτή βρίσκεται ο κόσμος που είτε εισέρχεται στο φεστιβάλ, είτε εξέρχεται από αυτό.

Ο αλγόριθμος k-means επιλέχθηκε αντί των υπόλοιπων αλγορίθμων συσταδοποίησης, λόγω των συμπερασμάτων που παρουσιάζουν τα αποτελέσματά του. Ο αλγόριθμος dbscan, αν και προσεγγίζει το πρόβλημα με διαφορετικό και εξίσου ενδιαφέρων τρόπο, απαιτεί επιπλέον επεξεργασία των αποτελεσμάτων, έτσι ώστε να δημιουργηθούν τα τελικά clusters.

## 5.3. Επιλογή και εξαγωγή χαρακτηριστικών

Η επεξεργασία των δεδομένων είναι απαραίτητη προϋπόθεση για την κατασκευή μοντέλων πρόβλεψης. Στόχος της επεξεργασίας αυτής είναι η εξαγωγή χαρακτηριστικών, τα οποία θα λειτουργούν ως ανεξάρτητες μεταβλητές, προσδιορίζοντας την κατάσταση της εξαρτημένης μεταβλητής (της εξόδου). Το πρόβλημα της πρόβλεψης της κατανομής του πληθυσμού, αφορά χρονικές περιόδους και απαιτεί τον εκ των προτέρων προσδιορισμό τους. Αποφασίσαμε έτσι να χωρίσουμε την ημέρα σε περιόδους, οι οποίες μπορούν να προσφέρουν σαφή συμπεράσματα στη μελέτη μας, και να έχουν και ιδιαίτερη φυσική σημασία. Κάθε φορά χρησιμοποιούμε την κατάσταση μιας χρονικής περιόδου, για την πρόβλεψη της κατάστασης της επόμενης. Για παράδειγμα, χωρίζοντας την ημέρα σε περιόδους της μιας ώρας, γνωρίζοντας την κατανομή του κόσμου στα cluster για μια περίοδο, προσπαθούμε να προβλέψουμε την κατανομή της επόμενης περιόδου. Αυτή η προσέγγιση όμως δεν έχει ιδιαίτερη φυσική σημασία, καθώς σε διάστημα μιας ώρας οι μεταβολές στην κατανομή των clusters είναι αρκετές. Κατά συνέπεια, η τιμές που χρησιμοποιούμε για τις προβλέψεις μας, δεν είναι αντιπροσωπευτική της κατάστασης που επικρατεί στο χώρο του φεστιβάλ. Αντίθετα, χωρίζοντας την ημέρα σε περιόδους του ενός λεπτού, χάνεται η ουσία του προβλήματός μας. Οι μεταβολές είναι τόσο μικρές από περίοδο σε περίοδο, και ως αποτέλεσμα τα συμπεράσματα που απορρέουν είναι μικρής σημασίας. Επίσης, αυτό που αξίζει να σημειώσουμε είναι ότι κάθε χρήστης μπορεί να ανήκει σε ένα cluster/AOI σε κάθε χρονική περίοδο, έτσι ώστε τα αποτελέσματά μας να είναι αντικειμενικά. Κατά συνέπεια, κάθε φορά ο κάθε χρήστης προσμετράται βάσει της πρώτης του καταγραφής σε κάθε χρονική περίοδο, ενώ οι υπόλοιπες καταγραφές του για την ίδια χρονική περίοδο διαγράφονται. Παρόλο που υπήρχαν αρκετοί πιστοί χρήστες της εφαρμογής (χρήστες που χρησιμοποιούσαν συχνά την εφαρμογή), ο συνολικός αριθμός των χρηστών ήταν μικρός. Κατ' επέκταση, πρέπει να επιλέξουμε αρκετά μικρή περίοδο έτσι ώστε να αξιοποιήσουμε τους πιστούς χρήστες, αλλά ταυτόχρονα αρκετά μεγάλη περίοδο έτσι ώστε να έχουμε όσο το δυνατό περισσότερα δεδομένα. Επιλέγουμε λοιπόν χρονικές περιόδους διάρκειας 15 λεπτών, και υπολογίζουμε την κατανομή του πληθυσμού στα cluster σε κάθε μία από αυτές. Στον Πίνακα 5.1 απεικονίζονται τα δεδομένα που έχουμε έως τώρα:

| **A** | **B** | **C** | **D** | **E** | **F** | **Χρονική περίοδος** |
|---|---|---|---|---|---|---|
| 8 | 2 | 5 | 10 | 4 | 5 | 2017-07-21 15:00:00 |
| 10 | 8 | 5 | 9 | 7 | 10 | 2017-07-21 15:15:00 |
| 26 | 9 | 1 | 5 | 6 | 6 | 2017-07-21 15:30:00 |
| 19 | 5 | 5 | 5 | 7 | 5 | 2017-07-21 15:45:00 |
| 23 | 9 | 5 | 3 | 10 | 8 | 2017-07-21 16:00:00 |

Πίνακας 5.1. Πληθυσμός των cluster για τις αντίστοιχες χρονικές περιόδους

Σε κάθε παρατήρηση, η τιμή της περιόδου αναπαριστά τη χρονική στιγμή κατά την οποία ξεκινάει η περίοδος. Σύμφωνα με τον πίνακα αυτό, στο cluster A, τη χρονική περίοδο *2017-07-21 15:30:00 έως 2017-07-21 15:45:00,* καταγράφηκαν συνολικά 26 άτομα. Αντίστοιχα στο cluster B καταγράφηκαν 9 άτομα, κοκ. Το ένα άτομο που καταγράφεται στο cluster C την περίοδο αυτή, είναι ενδεικτικό της ποιότητας των δεδομένων που έχουμε στην κατοχή μας, καθώς τα δεδομένα αυτά αντιπροσωπεύουν τα χιλιάδες άτομα που εκείνη τη στιγμή βρίσκονταν στο χώρο του φεστιβάλ.

Στην προσπάθεια αναζήτησης περισσότερων χαρακτηριστικών, έτσι ώστε να έχουμε περισσότερο υλικό για τα μοντέλα μας, παρατηρούμε δύο σημαντικά χαρακτηριστικά του φεστιβάλ:

- Το φεστιβάλ είναι ανοιχτού χώρου.
- Το φεστιβάλ έχει περισσότερους του ενός συναυλιακούς χώρους.

Οι δύο παραπάνω παρατηρήσεις μας οδηγουν στα εξής συμπεράσματα:

- Τα καιρικά φαινόμενα που επικρατούν είναι πολύ πιθανό να επηρεάζουν τη συμπεριφορά του κοινού εντός του χώρου του φεστιβάλ. Για παράδειγμα, σε περίπτωση βροχής, ο κόσμος ίσως μεταφέρεται σε clusters που περιέχουν σκέπαστρα. Επίσης, ίσως μεγάλο κομμάτι του κοινού να αποχωρεί από το φεστιβάλ σε περίπτωση δυσμενών καιρικών συνθηκών.
- Το φεστιβάλ έχει μουσικό χαρακτήρα, συνεπώς η συμπεριφορά του κοινού ίσως εξαρτάται από τη δημοτικότητα των καλλιτεχνών. Για παράδειγμα, cluster που περιέχουν συναυλιακούς χώρους όπου παρίστανται δημοφιλείς καλλιτέχνες, ίσως προσελκύουν περισσότερο κόσμο.

Τα παραπάνω συμπεράσματα μας οδηγούν στην κατασκευή νέων χαρακτηριστικών των οποίων η σημασία είναι ιδιαίτερη. Οι πληροφορίες αντλούνται από ανοιχτές πηγές δεδομένων, όπου έχουμε εύκολη και συνεχή πρόσβαση. Μέσα από μία γερμανική ιστοσελίδα πρόγνωσης του καιρού [55], λάβαμε τις απαραίτητες πληροφορίες, για τον καιρό κατά τη διάρκεια του φεστιβάλ. Στον Πίνακα 5.2 φαίνεται ο τρόπος με τον οποίο αντιστοιχίζουμε τα δεδομένα του καιρού με τις χρονικές περιόδους.

| **Θερμοκρασία** | **Καιρικές συνθήκες** | **Χρονική περίοδος** |
|---|---|---|
| 28 °C | Αραιή συννεφιά | 2017-07-21 19:45:00 |
| 27 °C | Μερική ηλιοφάνεια | 2017-07-21 20:00:00 |
| 27 °C | Μερική ηλιοφάνεια | 2017-07-21 20:15:00 |
| 27 °C | Βροχές και καταιγίδες | 2017-07-21 20:30:00 |

Πίνακας 5.2. Δεδομένα θερμοκρασίας και καιρικών συνθηκών

Στη συνέχεια, κατασκευάσαμε το πρόγραμμα και για τις τρεις ημέρες του φεστιβάλ, όπως φαίνεται στον Πίνακα 5.3.

| **Καλλιτέχνης** | **Σκηνή** | **Ώρα έναρξης** |
|---|---|---|
| Donots | Hauptbuhne | 2017-07-21 17:30:00 |
| Jennifer Rostock | Hauptbuhne | 2017-07-21 19:10:00 |
| Sportfreunde Stiller | Hauptbuhne | 2017-07-21 21:00:00 |
| Meute | Hauptbuhne | 2017-07-21 23:00:00 |
| Mars of Illyricum | Feldbuhne | 2017-07-21 20:00:00 |
| Astronautalis | Feldbuhne | 2017-07-21 21:15:00 |
| Curse | Feldbuhne | 2017-07-21 22:30:00 |
| OstWest Brothers | DJ-Buhne | 2017-07-21 18:00:00 |
| DJ SiMa | DJ-Buhne | 2017-07-21 19:30:00 |
| Daniel Metzger | DJ-Buhne | 2017-07-21 22:00:00 |
| Le Filou | DJ-Buhne | 2017-07-21 23:00:00 |

Πίνακας 5.3. Πρόγραμμα συναυλιών για την πρώτη ημέρα του φεστιβάλ

| **Καλλιτέχνης** | **Επικαιρότητα** | **Δημοτικότητα** |
|---|---|---|
| Donots | 0.83026372013594 | Καθιερωμένος |
| Jennifer Rostock | 0.41923628207298 | Καθιερωμένος |
| Sportfreunde Stiller | 0.72061136139894 | Mainstream |
| Meute | 1.1417908685146 | Καθιερωμένος |
| Mars of Illyricum | -2 | Ανεύρετος |
| Astronautalis | -0.10170046226542 | Καθιερωμένος |

Πίνακας 5.4. Μετρικές καλλιτεχνών/συγκροτημάτων για τους έξι πρώτους καλλιτέχνες/συγκροτήματα του προγράμματος

Με τη χρήση κατάλληλου API [56], λάβαμε τις εξής πληροφορίες για τον κάθε καλλιτέχνη/συγκρότημα:

- Μία μετρική η οποία αντιπροσωπεύει κατά πόσο βρίσκεται αυτήν τη στιγμή στην επικαιρότητα. Η μετρική αυτή προκύπτει μέσα από τη συχνότητα εμφάνισης του καλλιτέχνη τόσο στα social media, όσο και στους ραδιοφωνικούς σταθμούς. Η μετρική αυτή λαμβάνει τιμές από -2 έως 2.

- Μία μετρική που αντιπροσωπεύει τη δημοτικότητά του. Η μετρική αυτή λαμβάνει κατηγορικές τιμές (πχ. καθιερωμένος, ανεύρετος, mainstream).

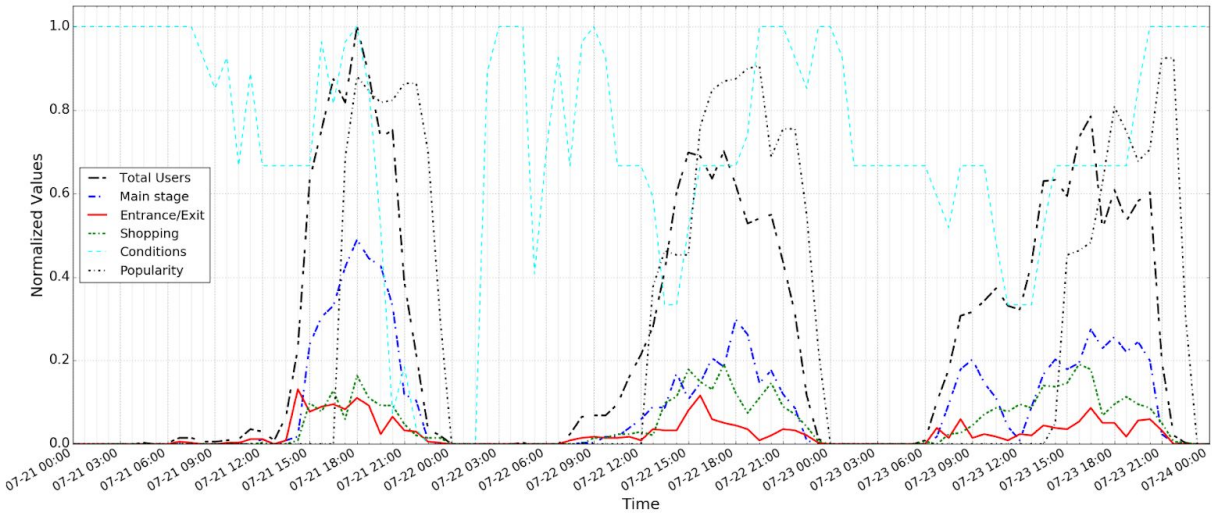
Στον Πίνακα 5.4 παρουσιάζονται τα δεδομένα αυτά για ορισμένους από τους καλλιτέχνες.

Στο σημείο αυτό, αν και φαίνεται να έχουμε αρκετά δεδομένα στη διάθεσή μας, απαραίτητη προϋπόθεση, προτού περάσουμε στο κομμάτι της κατασκευής των μοντέλων, είναι να μετατρέψουμε τα ακατέργαστα αυτά δεδομένα σε χαρακτηριστικά (ανεξάρτητες μεταβλητές). Αρχικά μετατρέπουμε όλες τις κατηγορικές μεταβλητές σε ποσοτικές, κλιμακωτής αξίας. Παίρνουμε όλες τις τιμές των καιρικών συνθηκών και τις αντιστοιχίζουμε σε αριθμούς από μηδέν έως τρία. Η τιμή μηδέν αντιστοιχίζεται στη χειρότερη καιρική κατάσταση (καταιγίδες), ενώ η τιμή τρία στην καλύτερη (ηλιοφάνεια). Με τον ίδιο τρόπο μετασχηματίζουμε τις τιμές της δημοτικότητας των συγκροτημάτων, καθώς και τις χρονικές περιόδους. Θεωρούμε ως μηδέν την πρώτη περίοδο της ημέρας, και αντικαθιστούμε τις υπόλοιπες περιόδους κατ' αντιστοιχία. Στη συνέχεια, αντιστοιχίζουμε τους καλλιτέχνες σε περιόδους, θεωρώντας ότι κάθε καλλιτέχνης βρίσκεται στη σκηνή το πολύ επτά περιόδους. Τέλος, αντιστοιχίζουμε τους καλλιτέχνες σε clusters, βάσει της σκηνής στην οποία εμφανίζονται. Λόγω του ότι δύο σκηνές βρίσκονται στο ίδιο cluster, αθροίζουμε τις μετρικές των καλλιτεχνών που εμφανίζονται ταυτόχρονα στο cluster αυτό, έτσι ώστε να δώσουμε περισσότερη βαρύτητα. Ο Πίνακας 5.5 αναπαριστά τα δεδομένα μας, όπως αυτά θα χρησιμοποιηθούν για την κατασκευή των μοντέλων πρόβλεψης. Για τις στήλες A/B artists, Temperature και Conditions έχει χρησιμοποιηθεί η κατάσταση της επόμενης χρονικής περιόδου, καθώς θέλουμε να μελετήσουμε πως η γνώση αυτή επηρεάζει τη συμπεριφορά του κόσμου εντός του φεστιβάλ.

| **A** | **B** | **C** | **D** | **E** | **F** | **A artists** | **B artists** | **Total users** | **Temperature** | **Conditions** | **Period index** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 17 | 2 | 5 | 6 | 2 | 1 | 0.807 | 0.608 | 33 | 22 | 2 | 75 |
| 12 | 2 | 6 | 6 | 4 | 1 | 0.678 | 0.608 | 31 | 22 | 2 | 76 |
| 19 | 3 | 7 | 5 | 3 | 2 | 0.678 | 0.575 | 39 | 22 | 2 | 77 |
| 18 | 2 | 12 | 4 | 9 | 1 | 0.678 | 0.795 | 46 | 22 | 2 | 78 |
| 12 | 0 | 7 | 5 | 5 | 3 | 0.678 | 0.795 | 32 | 22 | 2 | 79 |
| 11 | 1 | 8 | 4 | 1 | 3 | 0.678 | 0.795 | 28 | 21 | 3 | 80 |
| 14 | 0 | 8 | 3 | 4 | 3 | 0.678 | 0.795 | 32 | 21 | 3 | 81 |

Πίνακας 5.5. Δεδομένα με τα οποία τροφοδοτούμε τα μοντέλα πρόβλεψης.
Στον πίνακα παρουσιάζονται έξι χρονικές περίοδοι

Παρατηρούμε ότι συναυλίες έχουμε στα cluster A και B, ενώ επίσης προσθέτουμε και το συνολικό αριθμό του πληθυσμού του φεστιβάλ, σαν επιπλέον χαρακτηριστικό. Στη συνέχεια, χωρίζουμε την ημέρα σε περιόδους διάρκειας 45 λεπτών, έτσι ώστε να δημιουργήσουμε το διάγραμμα της Εικόνας 5.6. Στο διάγραμμα αυτό, μπορούμε να συγκρίνουμε την κατανομή του κόσμου στα AOIs, τις καιρικές συνθήκες, τη δημοτικότητα των καλλιτεχνών, και να παρατηρήσουμε πως οι τιμές των μεταβλητών αυτών κυμαίνονται εντός της ημέρας. Όλες οι τιμές είναι κανονικοποιημένες στο διάστημα [0,1]. Η διάρκεια των περιόδων (45 λεπτά), αυτή τη φορά μας δίνει μια ευρύτερη εικόνα, αναφορικά με τη σχέση μεταξύ των χαρακτηριστικών. Τέλος, κάθε καμπύλη αναπαριστά μία ημέρα του φεστιβάλ.




Εικόνα 5.6. Κατανομή του κόσμου στα AOIs και διακύμανση ανεξάρτητων μεταβλητών, σε σχέση με το χρόνο

Βλέποντας το διάγραμμα μπορούμε να βγάλουμε τα εξής συμπεράσματα:

- Την Κυριακή παρατηρούμε ότι υπάρχει προσέλευση του κόσμου στο χώρο του φεστιβάλ από νωρίς το πρωί. Αυτό ίσως οφείλεται στις δραστηριότητες που υπήρχαν εκείνη την ημέρα, και αφορούσαν τα μικρά παιδιά (σύμφωνα με το πρόγραμμα του φεστιβάλ).

- Από τις 12:00 έως τις 18:00 παρατηρούμε ότι ο κόσμος αυξάνεται, ενώ από τις 18:00 έως τις 00:00 ο κόσμος μειώνεται σταδιακά. Επίσης, όσο περνάει η ώρα, η δημοτικότητα του καλλιτέχνη επηρεάζει όλο και λιγότερο τη συγκέντρωση του κόσμου στο cluster της κεντρικής σκηνής.

- Η καμπύλη του cluster εισόδου/εξόδου και η καμπύλη της κεντρικής σκηνής, παρατηρούμε ότι συμπεριφέρονται με συμπληρωματικό τρόπο. Όταν η μία αυξάνεται, η άλλη μειώνεται.

- Ο καιρός επηρέασε την προσέλευση του κόσμου στο φεστιβάλ. Παρόλ' αυτά, η επίδρασή του φαίνεται να είναι μακροπρόθεσμη.

## 5.4. Κατασκευή Μοντέλων Πρόβλεψης - Ταξινόμηση

Εφόσον έχουμε μετατρέψει τα διαθέσιμα δεδομένα σε επιθυμητή μορφή, είμαστε έτοιμοι να δημιουργήσουμε μοντέλα πρόβλεψης. Αρχικά προσεγγίζουμε το πρόβλημα ως διεργασία ταξινόμησης. Δεδομένης της κατανομής του κόσμου σε μια χρονική περίοδο σε κάθε cluster, προσπαθούμε να προβλέψουμε την κατάστασή της την επόμενη χρονική περίοδο. Η κατάσταση αυτή λαμβάνει τις εξής τιμές:

- σταθερή
- αύξηση
- μείωση

Σαν είσοδο έχουμε μία χρονική περίοδο, και σαν έξοδο την επόμενη, όπως φαίνεται στον Πίνακα 5.6.

| **ΕΙΣΟΔΟΣ** | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **A** | **B** | **C** | **D** | **E** | **F** | **A pop** | **B pop** | **Total users** | **Temperature** | **Conditions** | **Period index** |
| 19 | 3 | 7 | 5 | 3 | 2 | 0.678 | 0.575 | 39 | 22 | 2 | 77 |
| | | | **ΈΞΟΔΟΣ** | | | | | | | | |
| | | | **A** | **B** | **C** | **D** | **E** | **F** | | | |
| | | | 18 | 2 | 12 | 4 | 9 | 1 | | | |

Πίνακας 5.6 Δεδομένα εισόδου και εξόδου

Μετασχηματίζουμε την έξοδο, έτσι ώστε να μετατρέψουμε το πρόβλημα σε διεργασία ταξινόμησης. Για να γίνει αυτό, θεωρούμε ότι αποκλίσεις εντός ενός ορίου θεωρούνται σταθερές. Το όριο αυτό καθορίζεται από ένα ποσοστό του συνολικού πληθυσμού του φεστιβάλ της εκάστοτε περιόδου (στην περίπτωσή μας 5%). Οπότε έχουμε $0.05 * 39 = 1.95$, και κατ' επέκταση η έξοδος παίρνει τη μορφή του Πίνακα 5.7.

| **ΈΞΟΔΟΣ** | | | | | |
|---|---|---|---|---|---|
| **A** | **B** | **C** | **D** | **E** | **F** |
| σταθερό | σταθερό | αύξηση | σταθερό | αύξηση | σταθερό |

Πίνακας 5.7. Δεδομένα εξόδου για διεργασία ταξινόμησης

Αρχικά, αντιμετωπίζουμε το πρόβλημα ως ταξινόμηση πολλών εξόδων (multioutput classification). Χρησιμοποιούμε τα δεδομένα μιας χρονικής περιόδου έτσι ώστε να προβλέψουμε την κατάσταση όλων των εξόδων ταυτόχρονα. Με τη χρήση της μετρικής *mutual information (MI) [57],* προσπαθούμε να εξετάσουμε τη σχέση μεταξύ της κάθε μεταβλητής εξόδου, με την κάθε ανεξάρτητη μεταβλητή. Στον Πίνακα 5.8 παρουσιάζουμε τα αποτελέσματά μας.

| features\output | **A** | **B** | **C** | **D** | **E** | **F** |
|---|---|---|---|---|---|---|
| **A** | **0.2193** | 0.0607 | 0.0275 | 0.1993 | 0.0571 | 0.1022 |
| **B** | 0.0637 | **0.1716** | 0.0137 | 0.1650 | 0.0280 | 0.0258 |
| **C** | 0.0837 | 0.0752 | **0.1805** | 0.0685 | 0.0245 | 0.0000 |
| **D** | 0.1682 | 0.0681 | 0.0122 | **0.2672** | 0.0000 | 0.0164 |
| **E** | 0.1950 | 0.0981 | 0.0057 | 0.1083 | **0.1981** | 0.0909 |
| **F** | 0.1437 | 0.0022 | 0.0184 | 0.0912 | 0.0786 | **0.1081** |
| **Apop** | 0.0269 | 0.0101 | 0.0000 | 0.0242 | 0.0995 | 0.0539 |
| **Bpop** | 0.0367 | 0.0185 | 0.0270 | 0.0206 | 0.0468 | 0.0083 |
| **Total users** | 0.2226 | 0.1471 | 0.1173 | 0.2108 | 0.1799 | 0.1823 |
| **Temperature** | 0.1015 | 0.0057 | 0.0378 | 0.1402 | 0.0927 | 0.0542 |
| **Conditions** | 0.0728 | 0.0401 | 0.0672 | 0.0000 | 0.0000 | 0.0063 |
| **Time index** | 0.1210 | 0.1421 | 0.0773 | 0.0808 | 0.1349 | 0.1092 |

Πίνακας 5.8. Πίνακα αμοιβαίας σχέσης (mutual information) μεταξύ
των ανεξάρτητων και των εξαρτημένων μεταβλητών

Παρατηρούμε ότι η έξοδος του κάθε cluster, εξαρτάται σε μεγάλο βαθμό από την αντίστοιχη τιμή της εισόδου. Πιο συγκεκριμένα, η τιμή του MI στο ζευγάρι (feature, output), (A,A) έχει την τιμή 0.2168. Όλες οι αντίστοιχες τιμές για τα υπόλοιπα cluster είναι αρκετά υψηλές, σε σχέση με τα υπόλοιπα ζεύγη (feature, output). Αυτό είναι ιδιαίτερα σημαντικό, καθώς φαίνεται ότι όντως η κατανομή του κόσμου σε μια χρονική περίοδο επηρεάζει την κατάσταση της εξόδου (δηλαδή, της επόμενης χρονικής περιόδου). Επίσης, παρατηρούμε ότι διαφορετικά χαρακτηριστικά επηρεάζουν με διαφορετικό τρόπο την έξοδο του κάθε cluster. Για παράδειγμα, η θερμοκρασία φαίνεται να επηρεάζει περισσότερο την κατάσταση του cluster A, σε σύγκριση με το cluster B. Κατ' επέκταση φτιάχνουμε διαφορετικούς συνδυασμούς χαρακτηριστικών, έτσι ώστε να εξετάσουμε πώς μεταβάλλεται η ακρίβεια των αποτελεσμάτων. Οι αλγόριθμοι που χρησιμοποιούμε για την ταξινόμηση είναι ο Random Forest (RF), ο K-Nearest Neighbors (KNN), ο Support Vector Classifier (SVC) και ο Naive Bayes (NB). Για τη δημιουργία ταξινομητών πολλών κλάσεων (multiclass classifiers), χρησιμοποιούμε *έναν-έναντι-όλων ταξινομητές (one vs all classifiers) [58].*

Οι διαφορετικοί συνδυασμοί χαρακτηριστικών που χρησιμοποιούμε κατά την προσαρμογή των ταξινομητών είναι η εξής:

- Πληθυσμός των cluster + time index [+time]
- Πληθυσμός των cluster + artists' popularity [+time_pops]
- Πληθυσμός των cluster + weather condition [+time_temp/cond]
- Πληθυσμός των cluster + artists' popularity + weather condition [+time_pops_temp/cond]

Με τη χρήση ενός standardscaler, μετασχηματίσαμε τα δεδομένα μας και εκπαιδεύσαμε κάθε μοντέλο πρόβλεψης τέσσερις φορές (βάσει των διαφορετικών συνδυασμών των χαρακτηριστικών) και χρησιμοποιήσαμε αναζήτηση πλέγματος (grid search) σε συνδυασμό με cross validation για την επιλογή του καλύτερου ταξινομητή. Σύμφωνα με την αναζήτηση πλέγματος, κάθε ταξινομητής προσαρμόζεται και εκπαιδεύεται με ένα σύνολο διαφορετικών συνδυασμών των παραμέτρων του, και βάσει της τεχνικής cross validation τα αποτελέσματα όλων των ταξινομητών συγκρίνονται και επιλέγεται ο καλύτερος. Ενδεικτικά, ο ταξινομητής Random Forest εκπαιδεύτηκε 330 φορές, κάθε φορά με διαφορετικό συνδυασμό παραμέτρων, για κάθε συνδυασμό χαρακτηριστικών ξεχωριστά. Στη συνέχεια, χρησιμοποιούμε five-fold-cross validation, για να συγκρίνουμε την ακρίβεια των ταξινομητών, και να επιλέξουμε τον καλύτερο. Στον Πίνακα 5.9 παρουσιάζονται τα αποτελέσματα.

| features\estimator | **mult_RF** | **mult_KNN** | **mult_SVC** | **mult_NB** |
|---|---|---|---|---|
| **clusters+time** | 0.6759 | 0.6806 | 0.6829 | 0.5718 |
| **clusters+time_pops** | 0.6944 | 0.6736 | 0.6759 | 0.5741 |
| **clusters+time_temp/cond** | 0.6898 | 0.6644 | 0.6782 | 0.5764 |
| **clusters+time_pops_temp/cond** | 0.6782 | 0.6389 | 0.6667 | 0.5741 |

Πίνακας 5.9. Πίνακα αποτελεσμάτων για ταξινομητές πολλών εξόδων

Παρατηρώντας τα αποτελέσματα, συμπεραίνουμε ότι η απόδοση των αλγορίθμων επηρεάζεται σε μικρό βαθμό από τα χαρακτηριστικά που χρησιμοποιούμε. Παρόλ' αυτά, ο ταξινομητής KNN παρουσιάζει καλύτερα αποτελέσματα όσο μειώνεται το πλήθος των χαρακτηριστικών. Αυτό δικαιολογείται, καθώς ο KNN θεωρείται *lazy classifier [59],* και κατ' επέκταση αποδίδει καλύτερα σε προβλήματα με λίγα χαρακτηριστικά. Ο RF και ο SVC έχουν σταθερή απόδοση, όπως επίσης και ο NB, παρόλο που ακρίβεια που πετυχαίνει ο τελευταίος είναι ιδιαίτερα μικρή.

Παρατηρούμε ότι με την παραπάνω μέθοδο, δηλαδή τη χρήση ταξινομητών πολλών εξόδων, μπορούμε να πετύχουμε ακρίβειες έως και 69.44%. Παρόλ' αυτά, ίσως η απόδοση κάποιων clusters να μειώνεται λόγω της φύσης των ταξινομητών. Για παράδειγμα, ίσως στο cluster A, να μπορέσουμε να πετύχουμε καλύτερη ακρίβεια, αν θεωρήσουμε την έξοδό του ξεχωριστή από τα υπόλοιπα clusters. Για το λόγο αυτό, επεκτείνουμε την έρευνα μας, προσπαθώντας να παρατηρήσουμε τη μεταβολή του πληθυσμού σε κάθε cluster ξεχωριστά. Αυτή τη φορά, εκπαιδεύσαμε κάθε μοντέλο πρόβλεψης 24

φορές (4 συνδυασμοί χαρακτηριστικών x 6 clusters), και υπολογίσαμε την ακρίβεια όλων των μοντέλων για κάθε περίπτωση ξεχωριστά. Όπως και προηγουμένως, χρησιμοποιήσαμε αναζήτηση πλέγματος, σε συνδυασμό με five-fold cross validation, για να συγκρίνουμε τα αποτελέσματά μας. Στον Πίνακα 5.10 παρατηρούμε τα αποτελέσματα κάθε μοντέλου, για κάθε διαφορετικό συνδυασμό χαρακτηριστικών, για κάθε cluster ξεχωριστά.

| model\cluster+features | **+time** | **+time_pops** | **+time_temp/cond** | **+time_pops_temp/cond** |
|---|---|---|---|---|
| **CLUSTER A** | | | | |
| **single_RF** | 0.6944 | 0.6944 | 0.6667 | 0.6806 |
| **single_KNN** | 0.7222 | 0.6944 | 0.6806 | 0.6111 |
| **single_SVC** | 0.6389 | 0.6111 | 0.6528 | 0.6111 |
| **single_NB** | 0.6250 | 0.6389 | 0.6250 | 0.6389 |
| **CLUSTER B** | | | | |
| **single_RF** | 0.6944 | 0.7361 | 0.6667 | 0.6944 |
| **single_KNN** | 0.7083 | 0.7222 | 0.6667 | 0.7083 |
| **single_SVC** | 0.6806 | 0.6944 | 0.6944 | 0.6944 |
| **single_NB** | 0.5556 | 0.5694 | 0.5694 | 0.5694 |
| **CLUSTER C** | | | | |
| **single_RF** | 0.7361 | 0.7222 | 0.7778 | 0.7361 |
| **single_KNN** | 0.7361 | 0.7222 | 0.7083 | 0.7083 |
| **single_SVC** | 0.7361 | 0.7361 | 0.7361 | 0.7361 |
| **single_NB** | 0.4583 | 0.4583 | 0.4583 | 0.4583 |
| **CLUSTER D** | | | | |
| **single_RF** | 0.6944 | 0.7083 | 0.6389 | 0.6389 |
| **single_KNN** | 0.6806 | 0.6667 | 0.6111 | 0.6528 |
| **single_SVC** | 0.6528 | 0.6528 | 0.6944 | 0.6944 |
| **single_NB** | 0.5972 | 0.5694 | 0.5972 | 0.5694 |

| **CLUSTER E** | | | | |
|---|---|---|---|---|
| **single_RF** | 0.6528 | 0.6667 | 0.6667 | 0.6944 |
| **single_KNN** | 0.6528 | 0.6667 | 0.6111 | 0.5833 |
| **single_SVC** | 0.6111 | 0.5833 | 0.5833 | 0.5833 |
| **single_NB** | 0.6111 | 0.6250 | 0.6389 | 0.6250 |
| **CLUSTER F** | | | | |
| **single_RF** | 0.6528 | 0.6667 | 0.6528 | 0.6250 |
| **single_KNN** | 0.6389 | 0.5972 | 0.6250 | 0.5833 |
| **single_SVC** | 0.6111 | 0.6111 | 0.6389 | 0.6389 |
| **single_NB** | 0.5833 | 0.5833 | 0.5694 | 0.5833 |

Πίνακας 5.10. Πίνακας ακρίβειας για ταξινομητές μιας εξόδου,
για όλους τους συνδυασμούς χαρακτηριστικών, για κάθε cluster ξεχωριστά

Για να έχουμε μία πιο σαφή εικόνα των αποτελεσμάτων, κατασκευάζουμε τον Πίνακα 5.11, ο οποίος περιέχει τη μέγιστη ακρίβεια που πετυχαίνει ο κάθε ταξινομητής σε κάθε cluster, άλλα και τη μέση μέγιστη ακρίβεια του κάθε ταξινομητή.

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** | **AVG** |
|---|---|---|---|---|---|---|---|
| **single_RF** | 0.6944 | 0.7361 | 0.7778 | 0.7083 | 0.6944 | 0.6667 | 0.713 |
| **single_KNN** | 0.7222 | 0.7222 | 0.7361 | 0.6806 | 0.6667 | 0.6389 | 0.6944 |
| **single_SVC** | 0.6528 | 0.6944 | 0.7361 | 0.6944 | 0.6111 | 0.6389 | 0.6713 |
| **single_NB** | 0.6389 | 0.5694 | 0.4583 | 0.5972 | 0.6389 | 0.5833 | 0.581 |

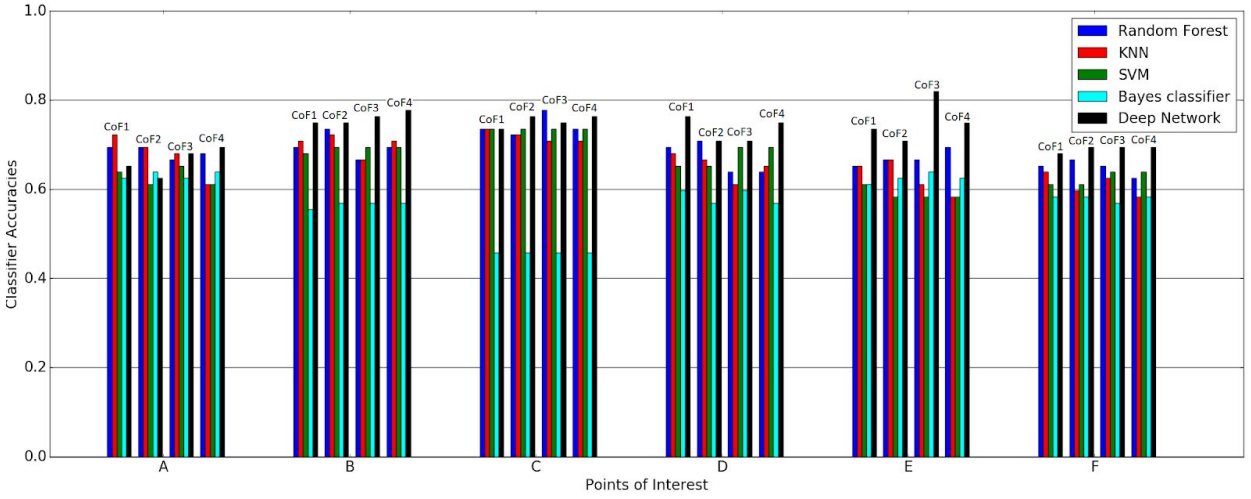
Πίνακας 5.11. Πίνακας μέγιστης ακρίβειας για κάθε cluster

Βλέπουμε ότι η ακρίβεια των προβλέψεών μας έχει βελτιωθεί αρκετά. Πλέον, χρησιμοποιώντας διαφορετικούς ταξινομητές και διαφορετικούς συνδυασμούς χαρακτηριστικών, μπορούμε να πετύχουμε ακρίβεια έως και 77.78%.

Συγκρίνοντας τα αποτελέσματα, παρατηρούμε ότι ο εκτιμητής Random Forest πετυχαίνει καλύτερη ακρίβεια, όταν συνδυάζει περισσότερα χαρακτηριστικά. Από την άλλη πλευρά, ο εκτιμητής KNN

αποδίδει καλύτερα όταν χρησιμοποιούνται μόνο οι χρονικές περίοδοι σαν επιπλέον χαρακτηριστικά. Ο SVC έχει σταθερή απόδοση, ανεξάρτητα των χαρακτηριστικών που χρησιμοποιούνται κατά την εκπαίδευσή του, αν και πολλές φορές τα αποτελέσματα παρουσιάζουν βελτίωση, όταν χρησιμοποιούμε περισσότερα χαρακτηριστικά. Επίσης, παρόλο που τα χαρακτηριστικά μας έχουν συνεχείς τιμές, ο εκτιμητής Naive Bayes είχε αρκετά χαμηλά ποσοστά επιτυχίας. Αυτό ήταν αναμενόμενο, καθώς τα χαρακτηριστικά δεν είναι ανεξάρτητα μεταξύ τους (το πλήθος του κόσμου σε ένα cluster εξαρτάται από το πλήθος του κόσμου στα υπόλοιπα clusters). Τέλος, παρατηρούμε ότι τα ποσοστά επιτυχίας διαφέρουν από cluster σε cluster. Αυτό έχει ιδιαίτερη σημασία, καθώς μπορούμε να συμπεράνουμε ότι η θέση στην οποία βρίσκεται το κάθε cluster, επηρεάζει την ακρίβεια των αποτελεσμάτων. Δηλαδή, μπορούμε πιο εύκολα να προβλέψουμε τη συμπεριφορά του κόσμου στην περιοχή της κεντρικής σκηνής και των μικρών καταστημάτων, σε σχέση με την περιοχή της εισόδου/εξόδου του φεστιβάλ.

Στην προσπάθειά μας για περαιτέρω βελτίωση των αποτελεσμάτων, προχωρήσαμε στην κατασκευή ενός νευρωνικού δικτύου, με τη χρήση τεχνικών βαθιάς μάθησης. Χρησιμοποιήσαμε όλους τους συνδυασμούς δεδομένων, κατασκευάζοντας δίκτυα πέντε στρωμάτων. Κάθε κρυφό στρώμα αποτελείται από εκατό νευρώνες, με υψηλό ρυθμό dropout, έτσι ώστε να αποφύγουμε το overfitting. Χρησιμοποιούνται η γραμμική συνάρτηση και η ReLu ως συναρτήσεις ενεργοποίησης, ενώ η συνάρτηση softmax χρησιμοποιείται για την εξαγωγή των αποτελεσμάτων στο στρώμα εξόδου. Για την επιλογή του καλύτερου μοντέλου κάναμε χρήση του γενετικού αλγορίθμου. Κατορθώσουμε να πετύχουμε αρκετά υψηλή ακρίβεια, σε πολλές περιπτώσεις. Στην Εικόνα 5.7 γίνεται σύγκριση των αποτελεσμάτων όλων των μοντέλων, για κάθε συνδυασμό χαρακτηριστικών (CoF).




Εικόνα 5.7. Σύγκριση των αποτελεσμάτων όλων των μοντέλων ταξινόμησης

## 5.5. Κατασκευή Μοντέλων Πρόβλεψης - Παλινδρόμηση

Έχοντας πλέον κατασκευάσει μοντέλα για την πρόβλεψη της κατάστασης του πλήθους του κόσμου την επόμενη χρονική στιγμή, προσπαθούμε αυτή τη φορά να προσεγγίσουμε το πρόβλημα ως

διεργασία παλινδρόμησης. Πιο συγκεκριμένα, γνωρίζοντας το πλήθος του κόσμου σε ένα cluster, μία δεδομένη χρονική περίοδο, προσπαθούμε να υπολογίσουμε το πλήθος του την επόμενη χρονική περίοδο. Για το λόγο αυτό χρησιμοποιούμε τους εκτιμητές Random Forest regressor (RFr), KNN regressor (KNNr), Support Vector Regressor (SVR) και Kernel Ridge regressor (KRr) [60]. Τα δεδομένα μας έχουν τη μορφή του Πίνακα 5.5. Αυτή τη φορά φτιάχνουμε μόνο εκτιμητές μιας εξόδου, έτσι ώστε να μπορέσουμε να μελετήσουμε όλους τους πιθανούς συνδυασμούς αποτελεσμάτων. Για τη σύγκριση των αποτελεσμάτων της αναζήτησης πλέγματος, χρησιμοποιούμε τη μέση τιμή του απόλυτου σφάλματος διαιρεμένη με τη μέση τιμή του πληθυσμού του κάθε cluster, βάσει της παρακάτω σχέσης.

$$error(\,cluster) \;=\; \frac{MAE(\,cluster)}{AVG(\,cluster's\ population)} \quad (5.1)$$

Με αυτόν τον τρόπο, έχουμε μια αντικειμενικότερη προσέγγιση της τιμής του σφάλματος, σε αντίθεση με τη χρήση πιο γνωστών συναρτήσεων κόστους *(MAE, MSE)*. Όπως και προηγουμένως, προτού εκπαιδεύσουμε τα μοντέλα μας, μετασχηματίζουμε με τη χρήση ενός standardscaler τα δεδομένα, και χρησιμοποιούμε five-fold cross validation για την εύρεση του καλύτερου εκτιμητή. Στον Πίνακα 5.12 παρουσιάζονται τα αποτελέσματα όλων των μοντέλων που κατασκευάστηκαν.

| model\cluster+features | **+time** | **+time_pops** | **+time_temp/cond** | **+time_pops_temp/cond** |
|---|---|---|---|---|
| **CLUSTER A** | | | | |
| **single_RFr** | 0.6184 | 0.6258 | 0.6277 | 0.6203 |
| **single_KNNr** | 0.5096 | 0.5026 | 0.5296 | 0.4507 |
| **single_SVR** | 0.5233 | 0.5263 | 0.5316 | 0.5304 |
| **single_KRr** | 0.4777 | 0.4858 | 0.5162 | 0.5090 |
| **CLUSTER B** | | | | |
| **single_RFr** | 0.3517 | 0.3528 | 0.3288 | 0.3134 |
| **single_KNNr** | 0.3627 | 0.3484 | 0.3481 | 0.3329 |
| **single_SVR** | 0.3518 | 0.3504 | 0.3498 | 0.3429 |
| **single_KRr** | 0.3277 | 0.3278 | 0.3257 | 0.3284 |
| **CLUSTER C** | | | | |
| **single_RFr** | 0.2475 | 0.2583 | 0.2436 | 0.2502 |
| **single_KNNr** | 0.2783 | 0.2702 | 0.2717 | 0.2630 |
| **single_SVR** | 0.2530 | 0.2580 | 0.2525 | 0.2571 |

| | | | | |
|---|---|---|---|---|
| **single_KRr** | 0.2538 | 0.2550 | 0.2555 | 0.2576 |
| **CLUSTER D** | | | | |
| **single_RFr** | 0.4582 | 0.4649 | 0.3986 | 0.4695 |
| **single_KNNr** | 0.4333 | 0.4221 | 0.4252 | 0.4239 |
| **single_SVR** | 0.4236 | 0.4200 | 0.4187 | 0.4170 |
| **single_KRr** | 0.4266 | 0.4265 | 0.4307 | 0.4288 |
| **CLUSTER E** | | | | |
| **single_RFr** | 0.4307 | 0.4247 | 0.4298 | 0.3847 |
| **single_KNNr** | 0.4038 | 0.4026 | 0.4017 | 0.4014 |
| **single_SVR** | 0.4289 | 0.4281 | 0.4240 | 0.4194 |
| **single_KRr** | 0.4278 | 0.4293 | 0.4147 | 0.4168 |
| **CLUSTER F** | | | | |
| **single_RFr** | 0.3499 | 0.3415 | 0.3803 | 0.3559 |
| **single_KNNr** | 0.3671 | 0.3528 | 0.3373 | 0.3303 |
| **single_SVR** | 0.3801 | 0.3841 | 0.3804 | 0.3792 |
| **single_KRr** | 0.3845 | 0.3826 | 0.3844 | 0.3781 |

Πίνακας 5.12. Πίνακας σφάλματος για όλα τα μοντέλα παλινδρόμησης, για όλους τους συνδυασμούς χαρακτηριστικών, για κάθε cluster ξεχωριστά

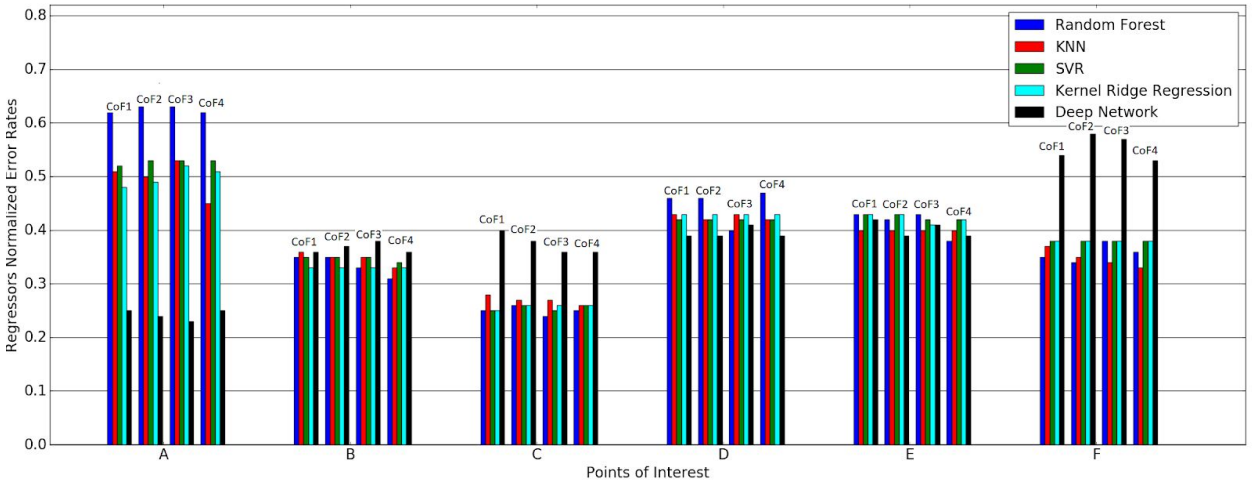
Όπως και στην περίπτωση της ταξινόμησης, για να έχουμε μια ευρύτερη εικόνα των αποτελεσμάτων, κατασκευάζουμε τον Πίνακα 5.13, που περιέχει το ελάχιστο σφάλμα κάθε εκτιμητή, για κάθε cluster.

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** | **AVG** |
|---|---|---|---|---|---|---|---|
| **single_RFr** | 0.6184 | 0.3134 | 0.2436 | 0.3986 | 0.3847 | 0.3415 | 0.3834 |
| **single_KNNr** | 0.4507 | 0.3329 | 0.2630 | 0.4221 | 0.4014 | 0.3303 | 0.3668 |
| **single_SVR** | 0.5233 | 0.3429 | 0.2525 | 0.4170 | 0.4194 | 0.3792 | 0.3891 |
| **single_KRr** | 0.4777 | 0.3257 | 0.2538 | 0.4265 | 0.4147 | 0.3781 | 0.3794 |

Πίνακας 5.13. Πίνακας ελάχιστου σφάλματος για κάθε cluster

Παρατηρούμε ότι η απόδοση των εκτιμητών Random Forest και KNN, επηρεάζεται σε σημαντικό βαθμό από τα χαρακτηριστικά που χρησιμοποιούνται κατά την εκπαίδευση των μοντέλων. Ο SVR και ο KR regressor, έχουν σταθερή απόδοση σε όλα τα cluster, ανεξάρτητα από το συνδυασμό των χαρακτηριστικών που χρησιμοποιήθηκε κατά την εκπαίδευσή τους. Για ακόμα μια φορά, παρατηρούμε ότι σε κάποια clusters το σφάλμα είναι αρκετά υψηλό, ενώ σε κάποια άλλα όχι. Για παράδειγμα, παρατηρούμε ότι η πρόβλεψη της κατανομής του κόσμου είναι δύσκολη στο cluster που περιέχει την κεντρική σκηνή. Αντιθέτως, μπορούμε να προβλέψουμε με μικρότερο σφάλμα το πλήθος των χρηστών στα clusters B και C.

Όπως και στην περίπτωση της ταξινόμησης, προσπαθήσαμε να βελτιώσουμε τα αποτελέσματά μας, χρησιμοποιώντας τεχνικές βαθιάς μάθησης. Ο εκτιμητής που κατασκευάσαμε αποτελείται από ένα νευρωνικό δίκτυο πολλών εξόδων, προβλέποντας την επόμενη κατάσταση όλων των clusters ταυτόχρονα. Χρησιμοποιεί τρία κρυφά στρώματα, αποτελούμενα από 128 νευρώνες το καθένα, τα οποία έχουν ως συνάρτηση ενεργοποίηση τη σιγμοειδή και τη γραμμική συνάρτηση, και ένα κρυφό στρώμα 72 νευρώνων, που χρησιμοποιεί ως συνάρτηση ενεργοποίησης τη ReLu. Τα αποτελέσματα όλων των μεθόδων που χρησιμοποιήσαμε, απεικονίζονται στην Εικόνα 5.8.




Εικόνα 5.8. Σύγκριση των αποτελεσμάτων όλων των μοντέλων παλινδρόμησης

## 5.6. Χρήση των Μοντέλων, για Πρόβλεψη της Κατανομής του Πληθυσμού του φεστιβάλ, τη χρονιά 2018

Σε αυτό το κομμάτι, προσπαθούμε να αξιολογήσουμε τους εκτιμητές που κατασκευάσαμε στις προηγούμενες ενότητες. Για να γίνει αυτό, εκπαιδεύουμε τους καλύτερους εκτιμητές κάθε αλγορίθμου με τα δεδομένα του φεστιβάλ για το έτος 2018. Οι καλύτεροι εκτιμητές προκύπτουν βάσει των αποτελεσμάτων, λαμβάνοντας υπόψη το συνδυασμό των χαρακτηριστικών και τις παραμέτρους των μοντέλων. Συνεπώς, αυτή τη φορά θα χρησιμοποιηθούν τέσσερις εκτιμητές (ένας για κάθε αλγόριθμο) για κάθε cluster του 2018. Στον Πίνακα 5.14 παρουσιάζεται ο συνδυασμός των

χαρακτηριστικών που προτιμήθηκε για κάθε συνδυασμό (cluster, μοντέλο).

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** |
|---|---|---|---|---|---|---|
| **single_RF** | +time | +time_pops | +time_temp/cond | +time_pops | +time_pops_temp/cond | +time_pops |
| **single_KNN** | +time | +time_pops | +time | +time | +time_pops | +time |
| **single_SVC** | +time_temp/cond | +time_pops_temp/cond | +time | +time_temp/cond | +time | +time_pops_temp/cond |
| **single_NB** | +time_pops_temp/cond | +time_pops_temp/cond | +time | +time | +time_temp/cond | +time |

Πίνακας 5.14. Συνδυασμός χαρακτηριστικών για κάθε ζεύγος (cluster, μοντέλο) - ταξινόμηση

Βάσει του παραπάνω πίνακα και των παραμέτρων που επιλέχθηκαν από την αναζήτηση πλέγματος για κάθε εκτιμητή, εκπαιδεύουμε εκ νέου τα μοντέλα μας με όλα τα δεδομένα του 2017, και προβλέπουμε την έξοδο για όλα τα δεδομένα του 2018. Παρουσιάζουμε την ακρίβεια των αποτελεσμάτων μας στον Πίνακα 5.15.

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** | **AVG** |
|---|---|---|---|---|---|---|---|
| **single_RF** | 0.6446 | 0.7526 | 0.5296 | 0.6481 | 0.6690 | 0.7038 | 0.6580 |
| **single_KNN** | 0.6272 | 0.7352 | 0.5923 | 0.6934 | 0.6690 | 0.6551 | 0.6620 |
| **single_SVC** | 0.5889 | 0.6969 | 0.5714 | 0.6620 | 0.6794 | 0.6376 | 0.6394 |
| **single_NB** | 0.6063 | 0.6760 | 0.5749 | 0.7247 | 0.6098 | 0.6516 | 0.6406 |

Πίνακας 5.15 Ακρίβεια προβλέψεων, για τα δεδομένα του 2018 - ταξινόμηση

Όπως και πριν, κατασκευάζουμε τους καλύτερους εκτιμητές παλινδρόμησης, και σε συνδυασμό με τα δεδομένα του Πίνακα 5.16 προβλέπουμε τις νέες κατανομές στα clusters. Ο Πίνακας 5.17 περιέχει τις τιμές σφάλματος που υπολογίστηκαν, βάσει των προβλέψεών μας και της εξίσωσης 5.1.

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** |
|---|---|---|---|---|---|---|
| **single_RFr** | +time | +time_pops_temp/cond | +time_temp/cond | +time_temp/cond | +time_pops_temp/cond | +time_pops |
| **single_KNNr** | +time_pops_temp/cond | +time_temp/cond | +time_pops_temp/cond | +time_pops | +time_pops_temp/cond | +time_pops_temp/cond |
| **single_SVR** | +time_temp/cond | +time_pops_temp/cond | +time_temp/cond | +time_pops_temp/cond | +time_pops_temp/cond | +time_pops_temp/cond |
| **single_KRr** | +time | +time | +time | +time_pops | +time_temp/cond | +time_pops_temp/cond |

Πίνακας 5.16. Συνδυασμός χαρακτηριστικών για κάθε ζεύγος
(cluster, μοντέλο) - παλινδρόμηση

| model\cluster | **A** | **B** | **C** | **D** | **E** | **F** | **AVG** |
|---|---|---|---|---|---|---|---|
| **single_RFr** | 0.4311 | 0.7432 | 0.6236 | 1.2637 | 1.0160 | 0.7071 | 0.7975 |
| **single_KNNr** | 0.5240 | 0.7352 | 0.6380 | 1.0607 | 1.0637 | 0.6669 | 0.7814 |
| **single_SVR** | 0.4207 | 0.6821 | 0.5527 | 0.8535 | 0.8087 | 0.6538 | 0.6619 |
| **single_KRr** | 0.4140 | 0.6669 | 0.5430 | 0.9129 | 0.9243 | 0.6609 | 0.6870 |

Πίνακας 5.17. Σφάλματα προβλέψεων, για τα δεδομένα του 2018 - παλινδρόμηση

Στον Πίνακα 5.15 παρατηρούμε ότι μέσα από τις τεχνικές μηχανικής μάθησης που εφαρμόσαμε, έχουμε πετύχει ακρίβεια έως και 75.26% στα δεδομένα του 2018. Συγκρίνοντας τα αποτελέσματά μας με αυτά του Πίνακα 5.11, διακρίνουμε ότι τα ποσοστά ακρίβειας διαφέρουν ανάμεσα στα clusters. Για παράδειγμα, προηγουμένως, στο cluster C είχαμε αρκετά υψηλή ακρίβεια, ενώ αυτή τη φορά η ακρίβεια των εκτιμητών έχει μειωθεί κατά 20%. Αυτό, αφενός οφείλεται στις λιγοστές παρατηρήσεις που είχαμε για τη χρονιά 2018, αφετέρου, ίσως οφείλεται στη διαφορετική δομή των εγκαταστάσεων του φεστιβάλ ανάμεσα στις δύο χρονιές.

Στο κομμάτι της παλινδρόμησης, παρατηρούμε ότι οι τιμές των σφαλμάτων είναι αρκετά υψηλές. Αυτό οφείλεται κυρίως στο πλήθος των δεδομένων που είχαμε στη διάθεσή μας. Όπως φαίνεται στην εξίσωση 5.1, η μέση τιμή του συνολικού πληθυσμού ενός cluster επηρεάζει αντιστρόφως ανάλογα την τιμή του σφάλματος. Λόγω του ότι η τιμή αυτή είναι αρκετά χαμηλή (~1) σε όλα σχεδόν τα clusters του 2018, η τιμή σφάλματος αυξάνεται.

## 5.7. Υλοποίηση

Η υλοποίηση του προβλήματος έγινε με τη χρήση της γλώσσας python[21]. Για το κομμάτι της μηχανικής μάθησης (clustering, μοντέλα πρόβλεψης), χρησιμοποιήθηκε η βιβλιοθήκη scikit-learn [61]. Η διαχείριση των δεδομένων, καθώς και ο μετασχηματισμός τους στην επιθυμητή μορφή, έγινε με τη χρήση των βιβλιοθηκών numpy [62], και pandas[62], [63]. Η κατασκευή των νευρωνικών δικτύων έγινε με τη χρήση της βιβλιοθήκης tensorflow [64].

Τα δεδομένα που έχουμε στη διάθεσή μας προέρχονται από ένα json αρχείο, που περιέχει εγγραφές (oid) αναφορικά με τα στοιχεία τοποθεσίας κάθε χρήστη (X, Y), την ταυτότητά του (ID), καθώς και την ώρα-ημερομηνία της εγγραφής. Τα στοιχεία που έχουμε στη διάθεσή μας είναι ανώνυμα, καθώς η ταυτότητα των χρηστών μας δίνεται με τη μορφή MAC διεύθυνσης. Κάνοντας χρήση του αρχείου *datapreprocessingTime.py* μετατρέπουμε το json αρχείο σε πίνακα, έτσι ώστε η πρόσβαση στα δεδομένα να είναι πιο εύκολη. Παράλληλα, κατά τη μετατροπή ελέγχουμε τις ημερομηνίες των εγγραφών, διαγράφοντας τις εγγραφές που πραγματοποιήθηκαν εκτός των ημερών του φεστιβάλ. Μετατρέψαμε τα δεδομένα ημερομηνίας και ώρας σε μορφή GMT και τα διαχωρίσαμε σε τρεις στήλες (Timestamp, Date, Time), καθώς αρχικά είχαν τη μορφή Epoch. Στον Πίνακα 5.18[22] φαίνονται οι 5 πρώτες εγγραφές που έχουμε στη διάθεσή μας.

| **$oid** | **ID** | **Epoch** | **Timestamp** | **Date** | **Time** | **X** | **Y** |
|---|---|---|---|---|---|---|---|
| 5954df20c1acdf0e8cc68f77 | f54a6682-732c-4fa4-b1b9-ec5b410a5de2 | 1500634396 | 2017-07-21 10:53:16 | 2017-07-21 | 10:53:16 | 20.1401708727474 | 46.2610297769029 |
| 5954df20c1acdf0e8cc68f78 | f54a6682-732c-4fa4-b1b9-ec5b410a5de2 | 1500634396 | 2017-07-21 10:53:16 | 2017-07-21 | 10:53:16 | 20.1401708727474 | 46.2610297769029 |
| 5954df20c1acdf0e8cc68f79 | f54a6682-732c-4fa4-b1b9-ec5b410a5de2 | 1500634396 | 2017-07-21 10:53:16 | 2017-07-21 | 10:53:16 | 20.1401708727474 | 46.2610297769029 |
| 5954df20c1acdf0e8cc68f7a | f54a6682-732c-4fa4-b1b9-ec5b410a5de2 | 1500634387 | 2017-07-21 10:53:07 | 2017-07-21 | 10:53:07 | 20.1401708727474 | 46.2610297769029 |
| 5954df20c1acdf0e8cc68f7b | f54a6682-732c-4fa4-b1b9-ec5b410a5de2 | 1500635146 | 2017-07-21 11:05:46 | 2017-07-21 | 11:05:46 | 20.1401708525204 | 46.2610297943781 |

Πίνακας 5.18 Πίνακας δεδομένων που προκύπτει από το αρχικό json αρχείο

[21] Στην ιστοσελίδα https://github.com/AnastasisB/Prediction-of-Visitors-Distribution-for-Large-Events βρίσκονται όλα τα αρχεία που χρησιμοποιήθηκαν στο πειραματικό κομμάτι της εργασίας.

[22] Καθόλη τη διάρκεια του κεφαλαίου της υλοποίησης, κάθε πίνακας περιέχει πέντε στοιχεία κάθε αρχείου δεδομένων (dataset). Όποτε γίνεται αναφορά στα δεδομένα ενός πίνακα, εννοείται αναφορά στο σύνολο όλων των δεδομένων του αντίστοιχου dataset.

Στη συνέχεια, χρησιμοποιούμε τα δεδομένα που έχουμε σχετικά με την τοποθεσία του φεστιβάλ, έτσι ώστε να κατασκευάσουμε το γεωγραφικό χώρο, στον οποίο πρέπει να βρίσκονται οι παρατηρήσεις μας. Στο αρχείο *defs.py,* η μέθοδος *polygonaki()* δέχεται ως είσοδο τα σημεία που ορίζουν το χώρο του φεστιβάλ. Έπειτα, τα σημεία αυτά ενώνονται, δημιουργώντας ένα πολύγωνο, το οποίο αναπαριστά τον εν λόγω χώρο. Πλέον, μπορούμε να ελέγξουμε ποιες εγγραφές βρίσκονται εκτός του πολυγώνου, και κατά συνέπεια να τις διαγράψουμε. Προκύπτει έτσι ο Πίνακας 5.19. Τα δεδομένα πλέον είναι φιλτραρισμένα βάσει της γεωγραφικής τοποθεσίας της κάθε εγγραφής, ενώ παράλληλα έχουν ταξινομηθεί βάσει ημερομηνίας και ώρας (πρώτη καταγραφή, έως τελευταία καταγραφή).

| **$oid** | **ID** | **Epoch** | **Timestamp** | **Date** | **Time** | **X** | **Y** |
|---|---|---|---|---|---|---|---|
| 597188dac1acdf13a82dd456 | 9e3a954d-550a-423c-b6da-42bfd61a7bbb | 1500612618 | 2017-07-21 04:50:18 | 2017-07-21 | 04:50:18 | 8.3716083 | 48.9980785 |
| 597248c2c1acdf13a82debfc | 9e3a954d-550a-423c-b6da-42bfd61a7bbb | 1500612618 | 2017-07-21 04:50:18 | 2017-07-21 | 04:50:18 | 8.3716083 | 48.9980785 |
| 5971f2e6c1acdf13a82dd677 | 9e3a954d-550a-423c-b6da-42bfd61a7bbb | 1500612618 | 2017-07-21 04:50:18 | 2017-07-21 | 04:50:18 | 8.3716083 | 48.9980785 |
| 59721b37c1acdf13a82dd9a7 | ba0c925f-13fd-4e7e-a021-8ff5de63049d | 1500617581 | 2017-07-21 06:13:01 | 2017-07-21 | 06:13:01 | 8.3731124 | 48.996683 |
| 59721ddbc1acdf13a82ddaa3 | ba0c925f-13fd-4e7e-a021-8ff5de63049d | 1500617581 | 2017-07-21 06:13:01 | 2017-07-21 | 06:13:01 | 8.3731124 | 48.996683 |

Πίνακας 5.19. Πίνακας δεδομένων που προκύπτει μετά από έλεγχο της τοποθεσίας των αρχικών δεδομένων

Επόμενο βήμα είναι η αντιστοίχιση της ώρας σε χρονικές περιόδους, καθώς και η αντιστοίχιση των πληροφοριών τοποθεσίας σε clusters. Οι παραπάνω διαδικασίες γίνονται με τη χρήση των μεθόδων *create()* και *assign_users_in_regions()* που βρίσκονται στα αρχεία *time_periods.py* και *defs_after.py* αντίστοιχα. Μπορούμε να καθορίσουμε τη διάρκεια των χρονικών περιόδων, τον αριθμό των clusters, καθώς και τον αλγόριθμο βάσει του οποίου γίνεται η συσταδοποίηση (Kmeans ή Meanshift). Εξετάζουμε κάθε εγγραφή του Πίνακα 5.19, και αντιστοιχίζουμε τα δεδομένα ώρας και τοποθεσίας, σε χρονικές περιόδους και clusters. Κατασκευάζεται έτσι ο Πίνακας 5.20 (έχουν επιλεχθεί χρονικές περίοδοι διάρκειας 15 λεπτών, και ο χώρος του φεστιβάλ έχει χωριστεί σε έξι clusters).

| $oid | ID | Epoch | Timestamp | Date | Time | X | Y | Period | Cluster |
|---|---|---|---|---|---|---|---|---|---|
| 597188dac1acdf13a82dd456 | 9e3a954d-550a-423c-b6da-42bfd61a7bbb | 1500612618 | 2017-07-21 04:50:18 | 2017-07-21 | 04:50:18 | 8.3716083 | 48.9980785 | 04:45:00 to 05:00:00 | C |
| 59721b37c1acdf13a82dd9a7 | ba0c925f-13fd-4e7e-a021-8ff5de63049d | 1500617581 | 2017-07-21 06:13:01 | 2017-07-21 | 06:13:01 | 8.3731124 | 48.996683 | 06:00:00 to 06:15:00 | C |
| 597218c1c1acdf13a82dd90f | 7fa011ad-03ae-4ea8-96a7-16d65ad0bbe0 | 1500621177 | 2017-07-21 07:12:57 | 2017-07-21 | 07:12:57 | 8.3737939 | 48.9975775 | 07:00:00 to 07:15:00 | B |
| 5971aad4c1acdf13a82dd4bb | 8817da41-ab4e-459f-adbd-6e52b9844599 | 1500621530 | 2017-07-21 07:18:50 | 2017-07-21 | 07:18:50 | 8.3762961 | 48.9992998 | 07:15:00 to 07:30:00 | E |
| 5971ab16c1acdf13a82dd4bd | d6a55f24-243f-4d43-93e5-3c1a2ae2e6e1 | 1500621596 | 2017-07-21 07:19:56 | 2017-07-21 | 07:19:56 | 8.37295537212706 | 48.9973861938249 | 07:15:00 to 07:30:00 | C |

Πίνακας 5.20. Αντιστοίχιση δεδομένων ώρας και τοποθεσίας, σε χρονικές περιόδους και clusters

Στο αρχείο *final_tune.py* η μέθοδος *combine_all.py* αξιοποιεί τις πληροφορίες του παραπάνω πίνακα, καθώς και τις πληροφορίες που έχουμε για τους καλλιτέχνες και τον καιρό, έτσι ώστε να κατασκευάσουμε το dataset με το οποίο τροφοδοτούμε τα μοντέλα πρόβλεψης. Τα δεδομένα open data βρίσκονται στα αρχεία *"weather_history.csv"* (πρόγνωση καιρού)*, artists_list.csv* (πρόγραμμα καλλιτεχνών), *metrs.xls* (μετρικές δημοτικότητας καλλιτεχνών), και έχουν τη μορφή των Πινάκων 5.2, 5.3 και 5.4 αντίστοιχα. Η μέθοδος *cluster_distribution_per_period()* του αρχείου *defs_after* υπολογίζει το πλήθος των παρατηρήσεων σε κάθε cluster, για κάθε χρονική περίοδο. Πλέον, κάθε γραμμή του πίνακα δεδομένων, αντιπροσωπεύει μία χρονική περίοδο κάθε ημέρας (Πίνακας 5.1.). Στη συνέχεια, οι καλλιτέχνες αντιστοιχίζονται σε clusters (σύμφωνα με τη σκηνή στην οποία εμφανίζονται), καθώς και σε χρονικές περιόδους (σύμφωνα με την ώρα έναρξης της συναυλίας). Τα καιρικά δεδομένα αντιστοιχίζονται επίσης σε χρονικές περιόδους. Όλα τα διατάξιμα δεδομένα μετατρέπονται σε ποσοτικά, για να αξιοποιηθούν στη συνέχεια από τα μοντέλα πρόβλεψης. Τέλος, συνδυάζοντας όλες τις παραπάνω πληροφορίες, κατασκευάζουμε τον Πίνακα 5.5.

Στο αρχείο *control_all.py* ελέγχουμε όλες τις παραμέτρους, βάσει των οποίων κατασκευάζεται το τελικό dataset. Επιλέγουμε τον αριθμό των clusters, τη διάρκεια των χρονικών περιόδων, καθώς και τα χαρακτηριστικά που μας ενδιαφέρουν (καιρικές συνθήκες, μετρικές καλλιτεχνών).

Στο φάκελο *PREDICTIONS* περιέχονται τα αρχεία σύμφωνα με τα οποία γίνονται οι προβλέψεις. Κάθε αρχείο εκτελείται μέσω του terminal, με την εντολή '*python "όνομα αρχείου" "όνομα φακέλου" -s "shift"'*. Η παράμετρος *-s* ελέγχει κατά πόσο μετακινούνται οι πληροφορίες των καιρικών συνθηκών και των μετρικών των συγκροτημάτων. Στην περίπτωση που επιλέγουμε *"-s 1",* κάθε φορά χρησιμοποιούμε την κατάσταση της επόμενης χρονικής περιόδου σα χαρακτηριστικό για τις προβλέψεις μας. Έτσι, εξετάζουμε κατά πόσο η γνώση αυτή (πρόγνωση), επηρεάζει τα αποτελέσματα των προβλέψεών μας. Ως *"όνομα φακέλου"* επιλέγουμε έναν από τους φακέλους εντός του *datasets.* Στους φακέλους αυτούς περιέχονται τα δεδομένα που έχουμε στη διάθεσή μας, για περιόδους διάρκειας 15 και 30 λεπτών.

Τέλος, ο φάκελος *"2018_evaluation"* περιέχει τα αρχεία που εκτελέστηκαν για να γίνει η αξιολόγηση των καλύτερων εκτιμητών. Οι παράμετροι κάθε εκτιμητή περιέχονται στα αρχεία *"best_params_class.yml"* και *"best_params_regression.yml"*.

## 5.8. Συμπεράσματα

Αντιμετωπίσαμε το πρόβλημα της κατανομής του κόσμου σε ένα φεστιβάλ, τόσο ως διεργασία ταξινόμησης, όσο και ως διεργασία παλινδρόμησης. Στην πρώτη περίπτωση, στόχος ήταν η πρόβλεψη της μεταβολής του πληθυσμού εντός των περιοχών ενδιαφέροντος (αύξηση, μείωση, σταθερή), ανάμεσα σε χρονικές περιόδους διάρκειας 15 λεπτών. Στη δεύτερη περίπτωση, προσπαθήσαμε να προβλέψουμε τη νέα κατανομή που προκύπτει, ανάμεσα στις ίδιες χρονικές περιόδους. Αρχικά, είδαμε πως μπορούμε να αξιοποιήσουμε δεδομένα από ανοιχτές πηγές (open data), στα οποία η πρόσβαση είναι εύκολη και άμεση, για την εξαγωγή χαρακτηριστικών. Έπειτα, εκπαιδεύσαμε εκτιμητές πολλών εξόδων, έτσι ώστε να προβλέπουμε την έξοδο σε όλες τις περιοχές ταυτόχρονα, ενώ στη συνέχεια κατασκευάσαμε μοναδικούς εκτιμητές, διαφορετικούς για κάθε περιοχή, προσπαθώντας να βελτιώσουμε την ακρίβεια των αποτελεσμάτων μας. Τέλος, χρησιμοποιώντας αναζήτηση πλέγματος, κατορθώσαμε να βρούμε τους καλύτερους εκτιμητές για κάθε μοντέλο, καθώς και τον καλύτερο συνδυασμό χαρακτηριστικών, για κάθε περιοχή ξεχωριστά.

Όπως αναφέραμε και στο κομμάτι της συσταδοποιησης στην αρχή του κεφαλαίου, η γνώση της κατανομής του χώρου του φεστιβάλ μπορεί να καθορίσει σε μεγάλο βαθμό τα αποτελέσματα των προβλέψεών μας. Με επισκόπηση του χώρου, μπορούμε να εισάγουμε νέα χαρακτηριστικά στα μοντέλα που κατασκευάζουμε, καθώς και να ορίσουμε προκαθορισμένα clusters, βάσει των προτιμήσεών μας.

Παρατηρούμε ότι συνδυάζοντας μη επιτηρούμενες (clustering) και επιτηρούμενες (μοντέλα πρόβλεψης) τεχνικές μηχανικής μάθησης, εξάγουμε σημαντικά συμπεράσματα, σχετικά με τη συμπεριφορά του κόσμου στις μεγάλες εκδηλώσεις. Μελετώντας τα αποτελέσματα, μπορούμε να προτείνουμε λύσεις στους διοργανωτές των φεστιβάλ, τόσο για τη λήψη μέτρων προστασίας, όσο και για την καλύτερη εξυπηρέτηση του κοινού.

Αξιοποιώντας τις πληροφορίες που μας δίνονται από τα μοντέλα πρόβλεψης, μπορούμε να βελτιώσουμε σημαντικά την ποιότητα ζωής στις έξυπνες πόλεις. Με τη χρήση αρχιτεκτονικών fog computing, η κατανομή του φόρτου εργασίας εντός της πόλης γίνεται δυναμικά, απελευθερώνοντας τους διαθέσιμους πόρους, μειώνοντας το κόστος εργασίας, ενώ παράλληλα η εξυπηρέτηση των χρηστών γίνεται άμεσα και με ευκολότερο τρόπο.

# Συντομογραφίες

| | | |
|---|---|---|
| ANN | Artificial Neural Network | Τεχνητό Νευρωνικό Δίκτυο |
| AOI/AoI | Area of Interest | Περιοχή Ενδιαφέροντος |
| BP | Backpropagation | Πίσω Διάδοση |
| DL | Deep Learning | Βαθιά Μάθηση |
| IoT | Internet of Things | Διαδίκτυο των Πραγμάτων |
| KNNr | K-Nearest Neighbors regressor | εκτιμητής KNN - παλινδρόμηση |
| KRr | Kernel Ridge regressor | εκτιμητής Kernel Ridge - παλινδρόμηση |
| LSTM | Long Short Term Memory | Νευρωνικά Δίκτυα Βραχείας και Μακράς Μνήμης |
| ML | Machine Learning | Μηχανική Μάθηση |
| mult_ | Multi output estimator | Εκτιμητής πολλών εξόδων |
| POI/PoI | Point of Interest | Σημείο Ενδιαφέροντος |
| RF | Random Forest | εκτιμητής RF |
| RFr | Random Forest regressor | εκτιμητής RF - παλινδρόμηση |
| RNN | Recurrent Neural Networks | Ανατροφοδοτούμενα Νευρωνικά Δίκτυα |
| single_ | Single output estimator | Εκτιμητής μιας εξόδου |
| SGD | Stochastic Gradient Descent | Στοχαστικός αλγόριθμος σύγκλισης με ελάττωση της παραγώγου |
| SVC | Support Vector Classifier | Ταξινομητής SVM |
| SVR | Support Vector Regressor | Εκτιμητής SVM για παλινδρόμηση |

# Βιβλιογραφικές Αναφορές


[1] E. Alpaydin, *Introduction to Machine Learning*. MIT Press, 2014.

[2] G. Brewka, "Artificial intelligence—a modern approach by Stuart Russell and Peter Norvig, Prentice Hall. Series in Artificial Intelligence, Englewood Cliffs, NJ," *Knowl. Eng. Rev.*, vol. 11, no. 01, p. 78, 1996.

[3] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2013.

[4] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[5] T. M. Kodinariya and P. R. Makwana, "Review on determining number of Cluster in K-Means Clustering," *Aquat. Microb. Ecol.*, vol. 1, no. 6, pp. 90–95, 2013.

[6] R. Tibshirani and G. Walther, "Cluster Validation by Prediction Strength," *J. Comput. Graph. Stat.*, vol. 14, no. 3, pp. 511–528, 2005.

[7] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics - Theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[8] K. Florek, J. Łukaszewicz, J. Perkal, H. Steinhaus, and S. Zubrzycki, "Sur la liaison et la division des points d'un ensemble fini," *Colloq. Math.*, vol. 2, no. 3–4, pp. 282–285, 1951.

[9] J. H. Ward, "Hierarchical Grouping to Optimize an Objective Function," *J. Am. Stat. Assoc.*, vol. 58, no. 301, p. 236, 1963.

[10] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, and Others, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Kdd*, 1996, vol. 96, pp. 226–231.

[11] S. Mehdikarimi, S. Norris, and C. Stalzer, "Regression Analysis of the Relationship between Income and Work," 2015.

[12] M. H. Kutner, *Applied Linear Statistical Models*. McGraw-Hill Education, 2005.

[13] K. H. Zou, K. Tuncali, and S. G. Silverman, "Correlation and simple linear regression," *Radiology*, vol. 227, no. 3, pp. 617–622, Jun. 2003.

[14] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat. Comput.*, vol. 14, no. 3, pp. 199–222, 2004.

[15] V. Roth and V. Steinhage, *Nonlinear Discriminant Analysis Using Kernel Functions*. 1999.

[16] W. Peng, J. Chen, and H. Zhou, "An implementation of ID3-decision tree learning algorithm," *From web. arch. usyd. edu. au/wpeng/DecisionTree2. pdf Retrieved date: May*, vol. 13, 2009.

[17] C. M. Dayton, "Logistic regression analysis," *Stat*, pp. 474–574, 1992.

[18] D. W. Hosmer and S. Lemeshow, *Applied Logistic Regression*. 2000.

[19] P. Davidson and A. M. Waas, "Probabilistic defect analysis of fiber reinforced composites using kriging and support vector machine based surrogates," *Compos. Struct.*, vol. 195, pp. 186–198, Jul. 2018.

[20] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[21] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[22] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *Ann. Stat.*, vol. 26, no. 5, pp. 1651–1686, 1998.

[23] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[24] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[25] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[26] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[27] R. Rojas, *Neural Networks: A Systematic Introduction*. Springer Science & Business Media,

2013.
[28] M. Hagan, H. Demuth, M. Beale, and O. De Jesus, *Neural Network Design (2nd Edition)*. 2014.
[29] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, pp. 386–408, Nov. 1958.
[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
[31] H. Leung and S. Haykin, "The complex backpropagation algorithm," *IEEE Trans. Signal Process.*, vol. 39, no. 9, pp. 2101–2104, 1991.
[32] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," in *Proceedings of COMPSTAT'2010*, 2010, pp. 177–186.
[33] J. Duchi, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *J. Mach. Learn. Res.*, vol. 12, no. Jul, pp. 2121–2159, 2011.
[34] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *arXiv [cs.LG]*, 22-Dec-2012.
[35] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv [cs.LG]*, 22-Dec-2014.
[36] W. Duch and N. Jankowski, "Survey of neural transfer functions," *Neural Computing Surveys*, vol. 2, no. 1, pp. 163–212, 1999.
[37] A. Vuckovic, D. Popovic, and V. Radivojevic, "Artificial neural network for detecting drowsiness from EEG recordings," in *6th Seminar on Neural Network Applications in Electrical Engineering*.
[38] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.
[39] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Trans Neural Netw Learn Syst*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[40] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.
[41] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, 1994.
[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
[43] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.
[44] "Beyond Traffic: The Vision for the Kansas City Smart City Challenge." [Online]. Available: https://cms.dot.gov/sites/dot.gov/files/docs/Kansas%20City%20Vision%20Narrative.pdf.
[45] A. Psychas *et al.*, "Cloud toolkit for Provider assessment, optimized Application Cloudification and deployment on IaaS," *Future Gener. Comput. Syst.*, 2018.
[46] J. Altmann *et al.*, "BASMATI: An Architecture for Managing Cloud and Edge Resources for Mobile Users," in *Lecture Notes in Computer Science*, 2017, pp. 56–66.
[47] H. F. Atlam, R. J. Walters, and G. B. Wills, "Fog Computing and the Internet of Things: A Review," vol. 2, no. 10, Apr. 2018.
[48] J. Violos, S. Pelekis, A. Berdelis, S. Tsanakas, K. Tserpes, T. Varvarigou, "'Predicting Visitor Distribution for Large Events in Smart Cities' In 1st International Workshop on Big data, cloud, and IoT technologies for smart cities, Kyoto, Japan 27 February, 2019."
[49] E. Carlini *et al.*, "BASMATI: Cloud Brokerage Across Borders for Mobile Users and Applications," in *Communications in Computer and Information Science*, 2018, pp. 181–186.
[50] J. Violos *et al.*, "User Behavior and Application Modeling in Decentralized Edge Cloud Infrastructures," in *Lecture Notes in Computer Science*, 2017, pp. 193–203.
[51] G. Z. Santoso *et al.*, "Dynamic Resource Selection in Cloud Service Broker," in *2017 International Conference on High Performance Computing & Simulation (HPCS)*, 2017.
[52] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine Recognition of Human

Activities: A Survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1473–1488, 2008.
[53] L. Garcia, G. Lansley, and B. Calnan, "Modelling Spatial Behaviour in Music Festivals Using Mobile Generated Data and Machine Learning," 2017.
[54] "DIE OFFIZIELLE DAS FEST-APP." [Online]. Available: http://www.dasfest.de/index.php?article_id=249&clang=0.
[55] "Weather in Karlsruhe," *https://www.timeanddate.com/weather/germany/karlsruhe*. .
[56] pwtempuser, "Next Big Sound," *ProgrammableWeb*, 27-Feb-2011. [Online]. Available: https://www.programmableweb.com/api/next-big-sound. [Accessed: 25-Nov-2018].
[57] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. John Wiley & Sons, 2012.
[58] R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification," *J. Mach. Learn. Res.*, vol. 5, no. Jan, pp. 101–141, 2004.
[59] G. Bontempi, M. Birattari, and H. Bersini, "Lazy learning for local modelling and control design," *Int. J. Control*, vol. 72, no. 7–8, pp. 643–658, 1999.
[60] M. Welling, "Kernel ridge regression," *Max Welling's Classnotes in Machine Learning*, pp. 1–3, 2013.
[61] R. Garreta and G. Moncecchi, *Learning scikit-learn: Machine Learning in Python*. Packt Publishing Ltd, 2013.
[62] T. Oliphant, *Guide to NumPy: 2nd Edition*. CreateSpace, 2015.
[63] W. McKinney and Others, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, 2010, vol. 445, pp. 51–56.
[64] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, and J. Dean, "Tensorflow: a system for large-scale machine learning," *OSDI*, 2016.