Πρόβλεψη καιρικών φαινομένων με χρήση αλγορίθμων Μηχανικής Μάθησης

Πανεπιστήμιο Πατρών

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ και Πληροφορικής

Βακαλόπουλος Δημήτρης

ΑΜ : 1059564

Επιβλέπων:	Σιούτας Σπυρίδων	Καθηγητής
Συνεπιβλέποντες:	Μακρής Χρήστος	Αναπληρωτής Καθηγητής
	Τσίχλας Κωνσταντίνος	Επίκουρος Καθηγητής

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Σπυρίδωνα Σιούτα για την συνεχή βοήθεια που μου παρείχε καθ' όλη τη διάρκεια της εκπόνησης της διπλωματικής μου εργασίας. Επίσης ευχαριστώ την οικογένεια μου για την οικονομική και ψυχολογική υποστήριξη που μου παρείχαν καθ' όλη την διάρκεια των σπουδών. Τέλος θα ήθελα να ευχαριστήσω τους φίλους μου για την ψυχολογική υποστήριξη που μου παρείχαν.

Περίληψη

Η διπλωματική εργασία αφορά την διαδικασία πρόβλεψης καιρικών φαινομένων, μέσω συστημάτων μηχανικής μάθησης. Στο πρώτο τμήμα της εργασίας γίνεται μια εισαγωγή στην πρόβλεψη των καιρικών φαινομένων, μέσω των βασικών εννοιών, των τεχνικών, των μεθόδων και των εργαλείων που χρησιμοποιούνται για τις προβλέψεις. Στη συνέχεια αναλύονται τα νευρωνικά δίκτυα και η μηχανική μάθηση ως εργαλεία τα οποία είναι στη διάθεση των πιο σύγχρονων τεχνολογιών αιχμής και χρησιμοποιούνται στην καθημερινότητα, όπως και στην πρόγνωση του καιρού. Κατά τη λειτουργία τους, τα παραπάνω εργαλεία χρησιμοποιούν μεθόδους ταξινόμησης και ομαδοποίησης οι οποίες αναλύονται στην παρούσα διπλωματική. Αφού παρουσιαστούν εκτενώς τα παραπάνω, θα χρησιμοποιηθεί ένα database που περιέχει δεδομένα για την πρόβλεψη καιρού. Το database συνθέτουν δεδομένα που αφορούν τη βροχόπτωση, τη θερμοκρασία, την πίεση στην ατμόσφαιρα, την κατεύθυνση και την ταχύτητα των ανέμων. Τα δεδομένα εισάγονται σε RNN δίκτυο και σε διαδικασίες ομαδοποίησης και ταξινόμησης. Συγκεκριμένα, σε KNN, Regression, Random Forest, Decision Tree, SVM και Naïve Bayes για την διαδικασία Classification, K-Means και Agglomerative για τη διαδικασία Clustering. Τέλος, γίνεται πρόβλεψη με ένα LSTM μοντέλο. Κατά το πρακτικό τμήμα της διπλωματικής (κεφάλαιο 7) εφαρμόζονται τα στοιχεία που έχουν παρατεθεί στο θεωρητικό τμήμα της (κεφάλαια 1-6) και καταλήγουμε στη διεξαγωγή συμπερασμάτων (κεφάλαιο 8).

Λέξεις – Κλειδιά : Εξόρυξη δεδομένων, τεχνικές ταξινόμησης, τεχνικές ομαδοποίησης συστάδων, Μετεωρολογική πρόβλεψη, Μηχανική Μάθηση

Abstract

The current dissertation concerns the process of forecasting weather phenomena, through machine learning systems. In the first part of the dissertation, an introduction to weather forecasting is given, through the basic concepts, techniques, methods and tools used for forecasting. Neural networks and machine learning are then analyzed as tools that are at the disposal of the most modern cutting-edge technologies and are used in everyday life, as well as in weather forecasting. During their operation process, the above tools use classification and grouping methods which are analyzed in this dissertation. After the above theoretical issues have been extensively presented, a database containing data for weather forecasting will be used. The database consists of data concerning precipitation, temperature, atmospheric pressure, wind direction and speed. The data is fed into an RNN network and into clustering and classification processes. Specifically, in KNN, Regression, Random Forest, Decision Tree, SVM and Naïve Bayes for the Classification process, K-Means and Agglomerative for the Clustering process. Finally, prediction is made with an LSTM model.During the practical part of the dissertation (chapter 7) the elements listed in the theoretical part (chapters 1-6) are applied and we end up drawing conclusions (chapter 8).

Keywords: Data mining, Clustering, Classification, Weather Forecast, Machine Learning

Πίνακας περιεχομένων

Κατάλογος Εικόνων

Κατάλογος Πινάκων

1. Εισαγωγή

Η Μετεωρολογία εντάσσεται στις ατμοσφαιρικές επιστήμες και κυρίως ασχολείται με την πρόγνωση του καιρού. Η μελέτη της μετεωρολογίας δεν είναι σύγχρονη επιστήμη καθώς έχει τις ρίζες της στην αρχαιότητα. Όμως, τα πιο βασικά τεχνολογικά επιτεύγματα που ώθησαν τη μετεωρολογία, επιτεύχθηκαν το 18ο αιώνα. Αντίστοιχα, το 19ο αιώνα σημειώθηκε σημαντική πρόοδος μετά τη δημιουργία δικτύων μέσω των οποίων ο καιρός μπορούσε να παρατηρηθεί σε μεγάλες περιοχές. Ως τότε, οι μετεωρολογικές προβλέψεις βασίζονταν κυρίως σε στοιχεία του παρελθόντος.

Μετά από τεχνολογικά επιτεύγματα μέσω ηλεκτρονικών υπολογιστών κατά δεύτερο μισό του 20ου αιώνα, σημειώθηκαν σημαντικά προχωρήματα στην πρόγνωση του καιρού. Επίσης, σημαντικός κλάδος της πρόγνωσης καιρού είναι η θαλάσσια πρόγνωση καιρού που σχετίζεται με την ασφάλεια στη θάλασσα και τις ακτές. Η μετεωρολογία και η υδρολογία απαρτίζουν την υδρομετεωρολογία κατά την οποία εξετάζονται οι αλληλεπιδράσεις μεταξύ της ατμόσφαιρας της Γης και των ωκεανών της, που αποτελούν μέρος ενός συζευγμένου συστήματος ωκεανού-ατμόσφαιρας. Η μετεωρολογία χρησιμοποιείται για διαφορετικούς τομείς όπως ο στρατός, η παραγωγή ενέργειας, οι μεταφορές, η γεωργία και οι κατασκευές.

Τα μετεωρολογικά φαινόμενα είναι παρατηρήσιμα καιρικά φαινόμενα που εξηγούνται από την μετεωρολογία. Περιγράφονται και μεταφράζονται σε ποσοτικούς όρους από τις μεταβλητές της ατμόσφαιρας της Γης: θερμοκρασία, πίεση και κατεύθυνση ανέμων, υγρασία, ροή μάζας και οι διακυμάνσεις και οι αλληλεπιδράσεις αυτών των μεταβλητών.

2. Πρόβλεψη καιρικών φαινομένων

Η πρόγνωση καιρού με αριθμητικούς όρους είναι μια διαδικασία η οποία χρησιμοποιεί μαθηματικά μοντέλα λαμβάνοντας στοιχεία που υπάρχουν στην ατμόσφαιρα και στους ωκεανούς και βασίζεται στις υπάρχουσες καιρικές συνθήκες. Η πρώτη προσπάθεια για την πρόβλεψη του καιρού έγινε στη δεκαετία του 1920, ενώ η πρώτη αριθμητική προσπάθεια η οποία είχε ως αποτέλεσμα αρκετά ρεαλιστικά αποτελέσματα με προσομοίωση σε ηλεκτρονικούς υπολογιστές, διεξήχθη τη δεκαετία του 1950. Σήμερα, διάφορα παγκόσμια και περιφερειακά μοντέλα λειτουργούν σε διαφορετικές χώρες διεθνώς τον κόσμο, χρησιμοποιώντας τις τρέχουσες καιρικές παρατηρήσεις που αναμεταδίδονται από ραδιοβολίδες ή μετεωρολογικούς δορυφόρους ως εισροές στα μοντέλα τα οποία υπάρχουν. Τα δύο πιο γνωστά συστήματα σε παγκόσμιο επίπεδο είναι το Αμερικάνικο **Παγκόσμιο Σύστημα Προβλέψεων** της Εθνικής Μετεωρολογικής Υπηρεσίας (GFS), και το **Ευρωπαϊκό Κέντρο Πρόγνωσης Καιρού Μεσαίου Εύρους** (ECMWF).

Το Παγκόσμιο Σύστημα Πρόβλεψης (GFS) είναι ένα μοντέλο πρόγνωσης καιρού των Εθνικών Κέντρων Περιβαλλοντικής Πρόβλεψης (NCEP) που παράγει δεδομένα για δεκάδες ατμοσφαιρικές και εδαφικές μεταβλητές, συμπεριλαμβανομένων των θερμοκρασιών, των ανέμων, της βροχόπτωσης, της υγρασίας του εδάφους και της συγκέντρωσης του όζοντος στην ατμόσφαιρα. Το σύστημα συνδυάζει τέσσερα ξεχωριστά μοντέλα (ατμόσφαιρα, μοντέλο ωκεανού, μοντέλο ξηράς/εδάφους και θαλάσσιο πάγο) που συνεργάζονται για να απεικονίσουν με ακρίβεια τις καιρικές συνθήκες. Οι προβλέψεις του είναι παγκόσμιες και βασίζονται σε βασική οριζόντια ανάλυση 28 χιλιομέτρων μεταξύ των σημείων ενός πλέγματος ανάλυσης. Η χρονική ανάλυση καλύπτει την ανάλυση και τις προβλέψεις έως τις 16 ημέρες. Η οριζόντια ανάλυση διεξάγεται στα 70 χιλιόμετρα μεταξύ των σημείων του πλέγματος για προβλέψεις μεταξύ μιας και δύο εβδομάδων.

Το Ευρωπαϊκό Κέντρο Προγνώσεων Καιρού Μεσαίου Εύρους (ECMWF) έχει μόνιμη λειτουργία και παράγει αριθμητικές προβλέψεις καιρού σε παγκόσμιο επίπεδο καθώς και άλλα δεδομένα για τα κράτη μέλη και τα συνεργαζόμενα κράτη της Ευρωπαϊκής Ένωσης. Διαθέτει τεράστιες εγκαταστάσεις υπολογιστών και υπερυπολογιστών καθώς και αρχεία μετεωρολογικών δεδομένων σε παγκόσμιο επίπεδο. Ταυτόχρονα, παρέχει προηγμένη εκπαίδευση και βοήθειας στο Παγκόσμιο Μετεωρολογικό Οργανισμό για την εφαρμογή των προγραμμάτων του. Το Ευρωπαϊκό Κέντρο Προγνώσεων Καιρού Μεσαίου Εύρους συμμετέχει στο Παρατηρητήριο της Γής - Copernicus, του διαστημικού προγράμματος της Ευρωπαϊκής Ένωσης, για την παροχή ποιοτικών πληροφοριών σχετικά με την κλιματική αλλαγή την ατμοσφαιρική σύνθεση, τις πλημμύρες και τον κίνδυνο πυρκαγιών. Ταυτόχρονα, το ECMWF μέσω της πρωτοβουλίας “Destination Earth” της Ευρωπαϊκή Ένωση, αναπτύσσει πρωτότυπα ψηφιακές αναπαραστάσεις της Γης.

2.1. Βασικές υπηρεσίες πρόβλεψης καιρικών φαινομένων

Στο συγκεκριμένο τμήμα του κεφαλαίου γίνεται μια εισαγωγή για τις βασικές υπηρεσίες που καλύπτουν την πρόβλεψη καιρικών φαινομένων ανά τη γη. Πολλές από αυτές τις υπηρεσίες θα αναλυθούν παρακάτω ως προς τα εργαλεία ή τις μεθόδους που χρησιμοποιούν. Συνεπώς, τα κυριότερα μετεωρολογικά κέντρα είναι τα εξής:

1. ECMWF (European Center for Medium-Range Weather Forecasts), Ευρωπαϊκό Κέντρο Προγνώσεων Καιρού Μεσαίου Εύρους

2. GFS (Global Forecast System), Παγκόσμιο Σύστημα Προβλέψεων το οποίο λειτουργεί από το εθνικό κέντρο μετεωρολογικών προβλέψεων των ΗΠΑ

3. ICON (Icosahedral Non-hydrostatic model), Εικοσαεδρικό μη υδροστατικό μοντέλο, της Γερμανικής εθνικής μετεωρολογικής υπηρεσίας

4. ARPEGE (Action de Recherche Petite Echelle Grande Echelle), η Ερευνητική Δράση Μικρής & Μεγάλης Κλίμακας η οποία αποτελεί μια συνεργασία μεταξύ του Γαλλικού κέντρου μετεωρολογίας, του ινστιτούτου Meteo-France και του Ευρωπαϊκού Κέντρου Προγνώσεων Καιρού Μεσαίου Εύρους

5. UM-UKMO (Unified Model, United Kingdom Meteorological Office), το Ενιαίο μοντέλο, της Μετεωρολογική Υπηρεσία Ηνωμένου Βασιλείου

6. ACCESS-G (Australian Community Climate and Earth-System Simulator-Global model), Ο Προσομοιωτής Συστήματος Κλίματος και Γης, της Αυστραλίας

7. GSM-JMA (Global Spectral Model-Japan Meteorological Agency) της Ιαπωνίας

8. GEM (Global Environmental Multiscale model, Environment and Climate Change Canada) του Καναδά

9. RHMC (Hydrometeorological Center of Russia), το Υδρο-μετεωρολογικό Κέντρο της Ρωσίας

Τα παραπάνω κέντρα – υπηρεσίες δεν είναι τα μοναδικά σε όλο τον κόσμο. Αντιθέτως υπάρχουν και δευτερεύοντα πιο μικρά, τοπικά, εθνικά κέντρα και υπηρεσίες μετεωρολογικών προβλέψεων σε διεθνές επίπεδο που όμως σε κάποιο βαθμό συνεργάζονται με τα παραπάνω για τις γρηγορότερες και εγκυρότερες προβλέψεις τους.

2.2.Γενικές τεχνικές πρόβλεψης καιρικών φαινομένων

Σε αυτό το τμήμα του κεφαλαίου αναφέρονται σε γενικές γραμμές οι τεχνικές που χρησιμοποιούν τα μεγαλύτερα κέντρα και υπηρεσίες μετεωρολογικών προβλέψεων. Για την κάθε τεχνική που αναφέρεται θα αναλυθούν στη συνέχεια οι μέθοδοι και τα εργαλεία των προβλέψεων.

Η μέθοδος της διατήρησης:

Πρόκειται για μια πολύ απλή μέθοδος πρόβλεψης η οποία βασίζεται σε τρέχουσες συνθήκες για να προβλέψει τις επόμενες. Αυτή μπορεί να είναι μια ασφαλής μέθοδος μετεωρολογικής πρόβλεψης όταν ο καιρός βρίσκεται σε σταθερή κατάσταση, όπως κατά τη διάρκεια μιας θερινής περιόδου σε τροπικές περιοχές. Όμως η μέθοδος εξαρτάται σε σημαντικό βαθμό από τη στατικότητα του καιρού. Συνεπώς, σε ασταθείς καταστάσεις , αυτή η μέθοδος πρόβλεψης γίνεται ανακριβής. Παρ' όλα αυτά, μπορεί να αποδειχθεί χρήσιμη σε προβλέψεις μικρής και μακράς εμβέλειας.

Η χρήση βαρόμετρου

Πρόκειται για μια μέθοδο που χρησιμοποιείται από τα τέλη του 19^{ου} αιώνα και βασίζεται σε μετρήσεις της βαρομετρικής πίεσης και της τάσης πίεσης (η μεταβολή της πίεσης με την πάροδο του χρόνου). Οι αλλαγές στη μεταβολή της πίεσης (και ειδικά οι μεγαλύτερες από 3,5 hPa - 2,6 mmHg) σημαίνουν και ισχυρότερες μετεωρολογικές διακυμάνσεις. Παράλληλα, η ταχύτερη πτώση της πίεσης μεταφράζεται σε ισχυρή πιθανότητα βροχής και οι ταχύτερες αυξήσεις της πίεσης μεταφράζονται συνήθως σε βελτίωση των καιρικών συνθηκών, όπως ο καθαρισμός του ουρανού.

Η παρατήρηση του ουρανού

Η παρατήρηση της κατάστασης του ουρανού είναι μαζί με την πίεση, από τις πιο σημαντικές παραμέτρους που χρησιμοποιούνται για την πρόγνωση του καιρού σε ορεινές περιοχές. Η αύξηση της νεφοκάλυψης ή η εισβολή σε υψηλότερο κατάστρωμα

νεφών είναι ενδεικτική βροχής στο εγγύς μέλλον. Τα ψηλά λεπτά σύννεφα μπορούν να δημιουργήσουν αντανακλάσεις σαν φωτοστέφανα γύρω από τον ήλιο, γεγονός που υποδηλώνει μια προσέγγιση ενός θερμού μετώπου και υψηλής πιθανότητας βροχής (Eskow, 1983). Η πρωινή ομίχλη προμηνύει πιο σταθερές μετεωρολογικά συνθήκες, καθώς οι βροχοπτώσεις προηγούνται από άνεμο ή σύννεφα που εμποδίζουν τη δημιουργία ομίχλης. Η προσέγγιση μιας σειράς καταιγίδων θα μπορούσε να υποδηλώνει τον ερχομό ενός ψυχρού μετώπου. Ο ουρανός χωρίς σύννεφα είναι ένδειξη καλού καιρού για το εγγύς μέλλον. Ο Wilson (n.d.) αναφέρει ότι η χρήση της κάλυψης του ουρανού στην πρόβλεψη του καιρού οδήγησε σε διάφορες καιρικές γνώσεις κατά τη διάρκεια των αιώνων.

Η άμεση εκτίμηση πρόβλεψης - Nowcasting

Η πρόγνωση του καιρού τις επόμενες έξι ώρες αναφέρεται συχνά ως nowcasting (U.S. National Weather Service). Εντός αυτού του χρονικού εύρους, είναι δυνατό να προβλεφθούν μικρότερα καιρικά φαινόμενα, όπως μεμονωμένες βροχές και καταιγίδες με λογική ακρίβεια, καθώς και άλλα χαρακτηριστικά πολύ μικρά για να επιλυθούν από ένα σύνθετο μοντέλο υπολογιστή. Ένας μετεωρολόγος με τα πιο πρόσφατα δεδομένα ραντάρ, δορυφόρου και παρατήρησης θα είναι σε θέση να κάνει μια καλύτερη ανάλυση των χαρακτηριστικών μικρής κλίμακας που υπάρχουν και έτσι θα είναι σε θέση να κάνει μια πιο ακριβή πρόβλεψη για ορισμένο αμέσως επόμενο διάστημα. Ωστόσο, υπάρχουν πλέον εξειδικευμένα συστήματα που χρησιμοποιούν τέτοια δεδομένα και το αριθμητικό μοντέλο μεσοκλίμακας για να κάνουν καλύτερη παρέκταση, συμπεριλαμβανομένης της εξέλιξης στο χρόνο. Η γνωστή διαδικτυακή πλατφόρμα Accuweather είναι γνωστή για τη λειτουργία Minute-Cast, που είναι μια πρόβλεψη βροχοπτώσεων λεπτό προς λεπτό για τις επόμενες δύο ώρες.

Χρήση μοντέλων πρόβλεψης

Η εποχή που ο ανθρώπινος παράγοντας ήταν αποκλειστικός υπεύθυνος για τη δημιουργία ολόκληρης της μετεωρολογικής πρόγνωσης με βάση τις διαθέσιμες παρατηρήσεις έχει παρέλθει. Σήμερα, η ανθρώπινη συμβολή περιορίζεται γενικά στην επιλογή ενός μοντέλου που βασίζεται σε διάφορες παραμέτρους, όπως οι

προκαταλήψεις και η απόδοση του μοντέλου. Η χρήση μοντέλων πρόβλεψης, καθώς και τμημάτων συνόλου των διαφόρων μοντέλων, μπορεί να βοηθήσει στη μείωση των σφαλμάτων πρόβλεψης. Ωστόσο, ανεξάρτητα από το πόσο μικρό γίνεται το μέσο σφάλμα με κάθε μεμονωμένο σύστημα, μεγάλα σφάλματα σε οποιοδήποτε συγκεκριμένο τμήμα καθοδήγησης εξακολουθούν να είναι δυνατά σε οποιαδήποτε εκτέλεση μοντέλου. Ο στόχος της μετεωρολογικής πρόβλεψης είναι η κατανοητή ερμηνεία στο ευρύ κοινό. Ακόμα και οι μικρές επιδράσεις προσθέτουν σημαντικές πληροφορίες για το μοντέλο που επεξεργάζεται και ερμηνεύει την τελική πρόβλεψη. Η ολοένα και αυξανόμενη ακρίβεια των μοντέλων πρόβλεψης σημαίνει ότι η χρησιμότητα του ανθρώπινου παράγοντα σταδιακά μειώνεται και ενδεχομένως να φτάσουμε σε μια εποχή που η ανθρώπινη παρέμβαση να μην χρειάζεται καθόλου.

Η αναλογική τεχνική

Η αναλογική τεχνική αποτελεί έναν πολύπλοκο τρόπο για να διεξαχθεί μια πρόβλεψη, που απαιτεί από τον μετεωρολόγο να θυμάται ένα προηγούμενο καιρικό γεγονός που αναμένεται να αναπαραχθεί από ένα επερχόμενο γεγονός. Αυτό που καθιστά δύσκολη τη χρήση αυτής της τεχνικής είναι ότι σπάνια υπάρχει ένα τέλεια αναλογική αναπαραγωγή ενός συμβάντος στο μέλλον. Παρά τη δυσκολία της, παραμένει μια χρήσιμη μέθοδος παρατήρησης για ορισμένα φαινόμενα όπως η βροχόπτωση πάνω από κενά δεδομένων όπως οι ωκεανοί και η πρόβλεψη των ποσοτήτων βροχοπτώσεων και της κατανομής στο μέλλον, φαινόμενα που τείνουν να επαναλαμβάνονται αναλογικά.

2.3. Μέθοδοι πρόβλεψης καιρικών φαινομένων

Η πρόβλεψη καιρικών φαινομένων μπορεί να πραγματοποιηθεί μέσω πληθώρας μεθόδων. Οι πιο βασικές μέθοδοι πρόβλεψης είναι η χρησιμοποίηση δεδομένων παρατήρησης, οι επιφανειακές παρατηρήσεις, οι παρατηρήσεις της ανώτερης ατμόσφαιρας, οι παρατηρήσεις ADMAR, τα δεδομένα από δορυφόρους και η αφομοίωση δεδομένων (Schulze, 2007).

Η χρησιμοποίηση **δεδομένων παρατήρησης** αφορά απλές και συμβατικές επιφανειακές παρατηρήσεις καθώς και παρατηρήσεις που έχουν ληφθεί στα ανώτερα αερικά στρώματα, για τα οποία χρησιμοποιούνται πιο προηγμένες τεχνολογίες, όπως το σύστημα παρατήρησης AMDAR, το οποίο θα αναλυθεί στη συνέχεια.

Οι **επιφανειακές παρατηρήσεις** είναι το πιο βασικό τμήμα του παγκόσμιου δικτύου παρατήρησης παρέχοντας δεδομένα επιφανειακής θερμοκρασίας, υγρασίας, ανέμου και πίεσης από συμβατικούς και αυτόματους μετεωρολογικούς σταθμούς. Είναι ικανές να παρέχουν προγνώσεις καιρού 72 ωρών σε παγκόσμιο επίπεδο, οι οποίες είναι σημαντικά ακριβείς. Για την κατανόηση της τεχνολογικής εξέλιξης αρκεί να αναλογιστούμε οτι ακρίβεια των σημερινών προβλέψεων στις 72 ώρες θεωρείται στα ίδια επίπεδα με την πρόβλεψη 24 ωρών του 1980. Η παραπάνω πρόοδος συντελέστηκε μέσω της κατανόησης των δυναμικών και φυσικών διεργασιών στην ατμόσφαιρα σε συνδυασμό με νέες, βελτιωμένες τεχνικές μοντελοποίησης, αυξημένη διαθεσιμότητα δεδομένων τηλεπισκόπησης από μετεωρολογικούς δορυφόρους και πρόοδο σε τεχνικές αφομοίωσης δεδομένων που ενσωματώνουν μετεωρολογικές παρατηρήσεις καιρού σε μοντέλα πρόβλεψης. Σε αυτή την τεχνολογική εξέλιξη συνέβαλλαν καθοριστικά και οι αυξημένες υπολογιστικές ικανότητες, οι οποίες θα αναλυθούν στη συνέχεια, όπως και τα εργαλεία που χρησιμοποιούνται σε αυτή την μέθοδο πρόβλεψης.

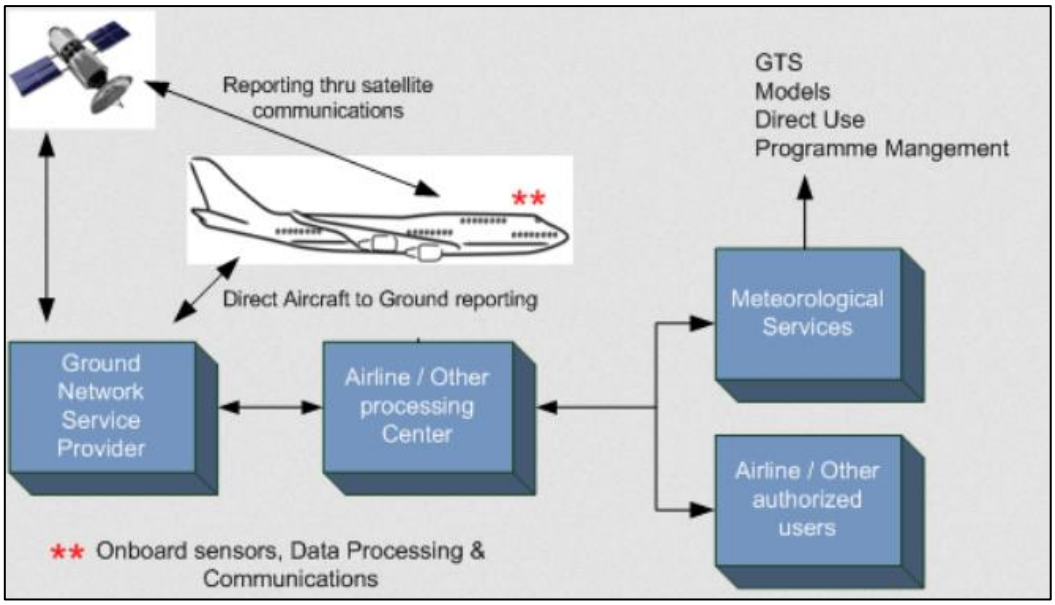
Οι **παρατηρήσεις στα ανώτερα επίπεδα αέρα** διεξάγονται για τον προσδιορισμό της τρισδιάστατης δομής της ατμόσφαιρας και πραγματοποιούνται με τη χρήση

ραδιοφωνικών ηχείων. Η παραπάνω διαδικασία γίνεται μέσω ενός οργάνου το οποίο είναι προσαρτημένο σε ένα μετεωρολογικό μπαλόνι με υδρογόνο, το οποίο λαμβάνει μετρήσεις της ατμοσφαιρικής πίεσης, της θερμοκρασίας και της υγρασίας σε διάφορα στρώματα της ατμόσφαιρας. Οι κινήσεις του μπαλονιού παρακολουθούνται χρησιμοποιώντας ένα παγκόσμιο σύστημα πλοήγησης για να ληφθεί η κατεύθυνση και η ταχύτητα του ανέμου στα διαφορετικά επίπεδα. Οι παρατηρήσεις με την πάροδο του χρόνου δείχνουν ότι τα δεδομένα για τα ανώτερα στρώματα του αέρα εμφανίζουν σημαντικές διαφορές τα τελευταία 50 χρόνια. Το 1961 το κέντρο της παρατήρησης ήταν το βόρειο ημισφαίριο: τα στοιχεία του Ευρωπαϊκού Κέντρου Μεσοπρόθεσμων Μετεωρολογικών δείχνουν το 97% των παρατηρήσεων αφορούσαν το βόρειο ημισφαίριο. Ο Schulze (2007) επισημαίνει ότι τα οφέλη από τη χρήση δορυφορικών δεδομένων στο νότιο ημισφαίριο είναι πολλά και η συνολική χρησιμότητα των προβλέψεων είναι μεγαλύτερη από μια πρόβλεψη μονάχα εστιασμένη από το βόρειο τμήμα του ισημερινού. Η παλιά στρατηγική θεωρείται δεν έπαιρνε υπόψη τις τρέχουσες ποιοτικές προβλέψεις στο νότο που πλησιάζουν εκείνες στο βορρά αλλά δεν είναι ίδιες. Αυτές οι παρατηρήσεις ακόμα και σήμερα παραμένουν σημαντικές και αξιοποιούνται ιδιαίτερα σε χερσαία τμήματα όπου τα δεδομένα ακτινοβολίας από δορυφόρους είναι αναξιόπιστα.

Το σύστημα **ADMAR** είναι ένα πρόγραμμα αναμετάδοσης μετεωρολογικών δεδομένων αεροσκαφών και αποτελεί πλέον ένα οικονομικά αποδοτικό μέσο για τη χρήση εμπορικών αεροσκαφών για τη λήψη ουσιαστικά περισσότερων δεδομένων στην κατακόρυφη διάσταση και σε δύσβατες περιοχές δεδομένων. Το AMDAR συλλέγει και διανέμει μετεωρολογικά δεδομένα όπως η ανάλυσης θερμοκρασίας αέρα, ταχύτητας και κατεύθυνσης ανέμου σε αεροδρόμια, τακτικές αναφορές μετεωρολογικών μεταβλητών από συμβατικά αεροπλάνα, ακριβείς μετρήσεις συντεταγμένων, μετρήσεις αναταράξεων στον αέρα, δεδομένα υδρατμών ή υγρασίας. Το πρόγραμμα αυτό εισήχθη από τον Παγκόσμιο Μετεωρολογικό Οργανισμό σχετικά πρόσφατα (περίπου το 1996) και έχει συναντήσει μεγάλη επιτυχία. Κατά την έναρξη της λειτουργίας του το πρόγραμμα καθημερινά έδινε 9.000 αναφορές και μέσα σε 8 χρόνια έφτασε τις 28.000 αναφορές. Το 2007 ο Παγκόσμιος Μετεωρολογικός Οργανισμός αναφέρει ότι το πρόγραμμα μπορεί να διεξάγει ακόμη και 700.000 αναφορές καθημερινά (Κογκολη, 2018). Το πρόγραμμα αυτό εφαρμόζεται ολοένα και

σε περισσότερες περιοχές του κόσμου και για την εκτέλεσή του χρησιμοποιείται ένας μεγάλος αριθμός αεροσκαφών. Για την πραγματοποίηση των μετρήσεων, ορισμένες αεροπορικές εταιρίες (πχ South African Airways) συνεργάζονται με τον Παγκόσμιο Μετεωρολογικό Οργανισμό (WMO) που τις παρέχουν με το λογισμικό ADMAR. Σήμερα ο WMO συνεργάζεται με περισσότερες από 40 αεροπορικές εταιρείες σε παγκόσμιο επίπεδο.

Εικόνα 1: Λειτουργία συστήματος ADMAR



(Πηγή: Παγκόσμιος Μετεωρολογικός Οργανισμός)

Αναφορικά με τα **δεδομένα από δορυφόρους**, υπάρχει σημαντική αύξηση στην ποσότητα και την ποιότητα των παρατηρήσεων μέσω δορυφόρων τα τελευταία χρόνια. Τα δεδομένα από δορυφόρους είναι πηγή δεδομένων που μέσω ακτινοβολίας μεταφέρονται και χρησιμοποιούνται απευθείας σε προσομοιώσεις και προγράμματα για την ανάλυσή τους. Η ηλιακή ενέργεια έχει καθοριστική επίδραση σε όλες τις διεργασίες στην ατμόσφαιρα. Σε αυτά τα πλαίσια, η αναπαράσταση του ισοζυγίου ενέργειας μεταξύ της Γης και ατμόσφαιρας είναι απαραίτητα ακριβής για την πρόγνωση του καιρού αλλά και γενικότερα για την κλιματική έρευνα. Ο Schulze (2007) αναφέρει πως η εξάρτηση των μετεωρολογικών υπηρεσιών και οργανισμών από δορυφορικά δεδομένα είναι ισχυρή, πιο έντονη από το παρελθόν καθώς η

χρησιμοποίησή τους έχει επεκταθεί και σε άλλες διαδικασίες όπως η πρόβλεψη των καιρικών συνθηκών.

Τέλος, η διαδικασία **αφομοίωσης δεδομένων** (data assimilation) είναι ένα σημαντικό συστατικό που χρησιμοποιείται στην πρόγνωση των μετεωρολογικών φαινομένων. Όπως ο εκτίμησης Bengtsson (1991) αναφέρει, κατά την απαρχή των διαδικασιών της αριθμητικής μετεωρολογικής πρόβλεψης, η βασική αρχή που δίεπε την ανάλυση ήταν η βέλτιστη παρεμβολή και η ανάλυση τελικά ήταν μια στάθμιση μεταξύ παρατηρήσεων και πεδίων αρχικής εκτίμησης. Όμως, η μέθοδος αυτή εμφάνισε κάποιες αδυναμίες όπως η μη αποτελεσματική αντιμετώπιση των αυξημένων ποσοτήτων μη-συμβατικών δεδομένων από όργανα τηλεπισκόπησης. Σταδιακά η μέθοδος αυτή βελτιώθηκε με την εισαγωγή της 3D μεταβλητής στη διαδικασία αφομοίωσης και της 4D μεταβλητής εκτίμησης (Bengtsson, 1991). Ίσως η πιο σημαντική βελτίωση στην ακρίβεια πρόβλεψης αριθμητικών μοντέλων κατά την πρόσφατη ιστορία αφορά αλλαγές στον τρόπο με τον οποίο παρατηρούνται κάθετες ηχητικές παρατηρήσεις από δορυφόρους που χρησιμοποιήθηκαν για την αφομοίωση δεδομένων (Schulze, 2007). Συγκεκριμένα, η μετατροπή των ακτινοβολιών που καταμετρώνται από δορυφόρους σε θερμοκρασία και υγρασία, αντικαταστάθηκε με την απευθείας χρησιμοποίησή τους σε διαδικασίες αφομοίωσης δεδομένων. Αυτή η αλλαγή εφαρμόστηκε άμεσα και στο Παγκόσμιο Σύστημα Παρατήρησης, για τη βελτίωση των διαστημικών αποστολών. Με τον τρόπο αυτό επιτράπηκε η πιο άμεση χρήση των δεδομένων από τα αρμόδια όργανα με αποτέλεσμα μεγαλύτερη ανάλυση χρόνου και χώρου.

2.4. Τα εργαλεία της πρόβλεψης καιρικών φαινομένων

Σε αυτό το τμήμα του κεφαλαίου παρουσιάζονται τα πιο βασικά εργαλεία που χρησιμοποιούνται στην πρόβλεψη καιρικών φαινομένων. Τα παρακάτω εργαλεία αξιοποιούνται για την παρακολούθηση των συνθηκών της ατμόσφαιρας που επηρεάζουν τα καιρικά φαινόμενα, μέθοδος που χρησιμοποιούνταν πάντα με διαφορετικά όμως εργαλεία κατά την πάροδο του χρόνου λόγω της ανάπτυξης της τεχνολογίας. Τα πιο προηγμένα μέσα που χρησιμοποιούνται στην εποχή μας αξιοποιούν πιο αποτελεσματικά τον εξοπλισμό για να συλλέξουν και να μεταφράσουν περισσότερα δεδομένα που υπάρχουν στην ατμόσφαιρα. Το αποτέλεσμα είναι να επιτυγχάνεται πιο γρήγορη και πιο έγκυρη πρόβλεψη των καιρικών φαινομένων.

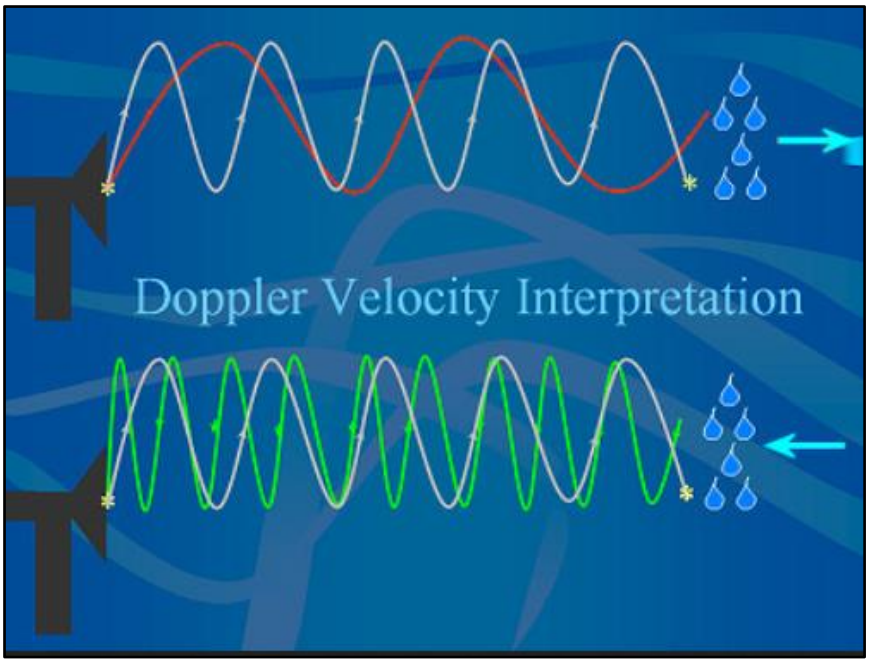
Το ραντάρ **Doppler** επιτρέπει στις μετεωρολογικές υπηρεσίες την παρατήρηση ισχυρών καταιγίδων. Το ραντάρ έχει ονομαστεί από τον Αυστριακό μετεωρολόγο Christian Doppler ο οποίος είναι ο εφευρέτης του. Τέτοια ραντάρ τοποθετούνται σε καθορισμένες αποστάσεις ώστε να καλύπτουν πλήρως μεγάλες εκτάσεις για την πρόβλεψη φαινομένων σε ολόκληρες περιοχές. Για παράδειγμα, η Εθνική Μετεωρολογική Υπηρεσία των ΗΠΑ έχει στη διάθεσή της 159 ραντάρ Doppler τα οποία εκτείνονται σε όλη την έκτασή της. Τα ραντάρ αυτά ανιχνεύουν τις βροχοπτώσεις και στοιχεία που τις αφορούν όπως ένταση, διάρκεια κ.α., την κίνηση των σύννεφων, την ένταση των ανεμοστρόβιλων, την ένταση και την κατεύθυνση όλων των ανέμων.

Το ραντάρ Doppler λειτουργεί με ένα ανακλώμενο το οποίο εκπέμπεται και στη συνέχεια λαμβάνεται από το ραντάρ κατά τη διάρκεια μιας ορισμένης περιόδου, η οποία έχει οριστεί ως περίοδος ακρόασης. Το ραντάρ συνδέεται με ηλεκτρονικούς υπολογιστές οι οποίοι χρησιμοποιούνται για να αναλύσουν ισχύ του ανακλώμενου σήματος, τον χρόνο που αυτό χρειάστηκε για να επιστρέψει, κ.α. Αυτή η διαδικασία εκπομπής, ανάκλασης, ακρόασης και στη συνέχεια εκπομπής ενός επόμενου σήματος πραγματοποιείται σε εξαιρετικά υψηλές χρονικές ταχύτητες, έως και περίπου 1300 φορές κάθε δευτερόλεπτο (National Weather Service, n.d.).

Τα ραντάρ Doppler έχουν την ικανότητα να ανιχνεύουν την μετατροπή της φάσης του παλμού της ενέργειας. Το σήμα που επιστρέφει στο ραντάρ αλλάζει φάση με βάση την κίνηση της βροχής ή άλλα φαινόμενα. Ο παλμός ενέργειας από το ραντάρ χτυπάει ένα

αντικείμενο και αντανακλάται πίσω προς το ραντάρ, το οποίο μέσω ηλεκτρονικών υπολογιστών μετράει την αλλαγή φάσης του παλμού που ανακλάται και στη συνέχεια τη μετατρέπει σε ταχύτητα από/προς το ραντάρ. Αυτές οι πληροφορίες χρησιμοποιούνται για την εκτίμηση της ταχύτητας που κινούνται τα αντικείμενα (π.χ. σύννεφα), οι άνεμοι, οι καταιγίδες, οι ανεμοστρόβιλοι κλπ.

Εικόνα 2: Λειτουργία Doppler



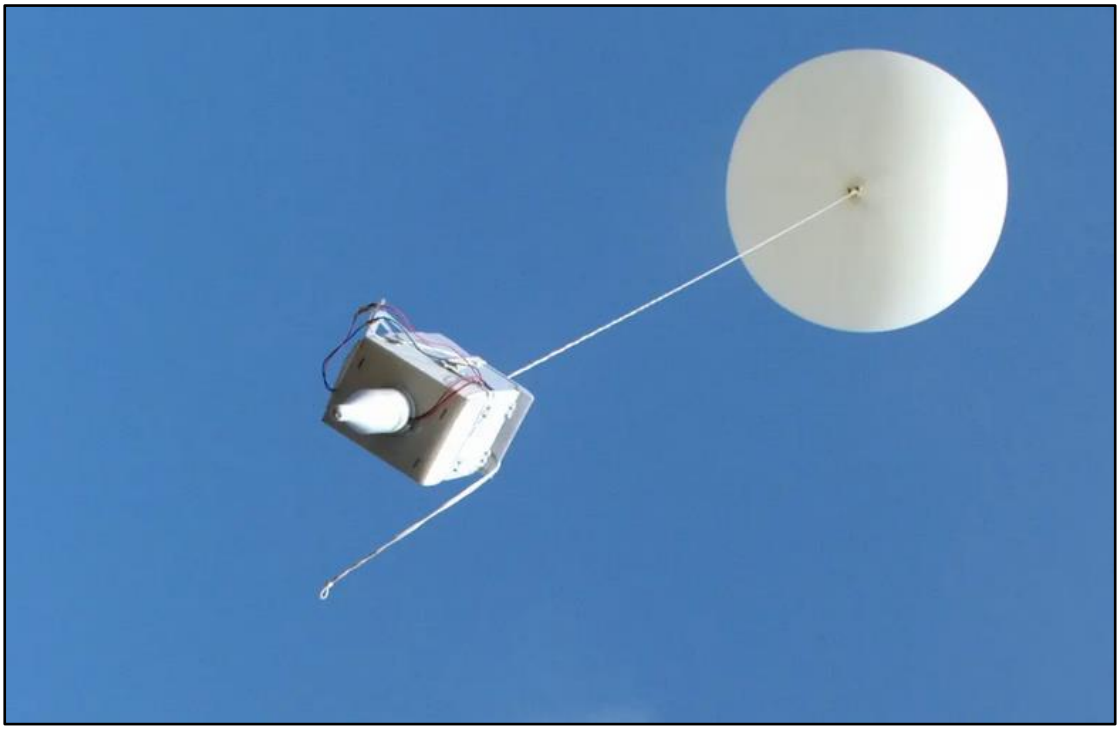
(Πηγή : weather.gov)

Οι **μετεωρολογικοί δορυφόροι** είναι μη επανδρωμένοι δορυφόροι που παρακολουθούν τη γη από το διάστημα και συλλέγουν δεδομένα που επεξεργάζονται και αναλύουν οι επιστημονικές ομάδες των μετεωρολογικών υπηρεσιών. Υπάρχουν τρείς τύποι μετεωρολογικοί δορυφόρων. Η πρώτη κατηγορία είναι οι πολικοί, οι οποίοι βρίσκονται σε τροχιά γύρω από τη γη κοντά στην επιφάνειά της και λαμβάνουν έξι/επτά αρκετά λεπτομερείς εικόνες της γης ανά ημέρα. Η δεύτερη κατηγορία δορυφόρων ονομάζεται γεωστατικοί, οι οποίοι έχουν σταθερή θέση σε μεγάλο ύψος από αυτή και λαμβάνουν εικόνες από ολόκληρη τη γη όσο περιστρέφεται, αρκετά συχνά κάθε 30 δευτερόλεπτα. Η τρίτη κατηγορία μετεωρολογικών δορυφόρων είναι οι βαθέως διαστήματος που είναι στραμμένοι προς τον ήλιο για να παρακολουθούν ισχυρές καταιγίδες σε αυτόν και γενικότερα τον καιρό στο διάστημα. Στην εποχή μας,

ένα μεγάλο δίκτυο από σύγχρονους μετεωρολογικούς δορυφόρους λειτουργεί κάτω από την επίβλεψη της χωρών που λειτουργούν μετεωρολογικές υπηρεσίες όπως η Ευρώπη, οι ΗΠΑ, η Ρωσία, η Κίνα, η Ιαπωνία και η Ινδία. Σε πολλές από αυτές τις περιπτώσεις των δορυφόρων, οι εικόνες, οι φωτογραφίες που λαμβάνουν οι δορυφόροι, ανεβαίνουν αυτόματα στο διαδίκτυο σε ιστότοπους που είναι προσβάσιμοι για το ευρύ κοινό και έτσι οι ενδιαφερόμενοι μπορούν να δουν απευθείας εικόνα από αυτούς τους δορυφόρους και να μάθουν πληροφορίες για τα καιρικά φαινόμενα.

Οι **ραδιοβολίδες** (**Radiosondes**) είναι ένα όργανο το οποίο κάνει μετρήσεις απομακρυσμένα (τηλεμετρήσεις). Λειτουργεί με μπαταρίες και μεταφέρεται στην ατμόσφαιρα μέσω μετεωρολογικού μπαλονιού. Προσμετρά διάφορες ατμοσφαιρικές παραμέτρους και τις μεταδίδει μέσω ραδιοφωνικών σημάτων σε έναν επίγειο δέκτη. Οι σύγχρονοι ραδιοφωνικοί σταθμοί μετρούν ή υπολογίζουν το υψόμετρο, την πίεση, την θερμοκρασία, την υγρασία, την ταχύτητα και την κατεύθυνση ανέμου, διάφορες μετρήσεις κοσμικής ακτίνας σε μεγάλο υψόμετρο και την γεωγραφική θέση (πλάτος/μήκος). Αυτού του είδους οι ραδιοβολίδες που μετρούν τη συγκέντρωση του όζοντος είναι γνωστοί ως όζονες (Gleason, 2008). Οι όζονες λειτουργούν σε ραδιοσυχνότητα 403 MHz ή 1680 MHz. Οι περισσότεροι ραδιοφωνικοί σταθμοί που στέλνουν σήμα οι όζονες, έχουν ανακλαστήρες ραντάρ. Ένας ακόμη τρόπος λειτουργίας των όζονων είναι η απελευθέρωσή τους από αεροπλάνα (dropsonde) αντί για την απελευθέρωσή τους επίγεια. Οι όζοντες είναι μια βασική πηγή μετεωρολογικών δεδομένων και εκατοντάδες εκτοξεύονται σε όλο τον κόσμο καθημερινά. Κατά τη λειτουργία τους οι όζοντες συνδέονται με μετεωρολογικά μπαλόνια τουλάχιστον δύο φορές τη μέρα. Οι ΗΠΑ διαθέτουν μετεωρολογικά μπαλόνια σε 92 τοποθεσίες. Κατά τη διαδρομή του, ο όζοντας βρίσκεται στην ανώτερη στρατόσφαιρα από όπου γίνεται η συλλογή δεδομένων τα οποία στέλνει πίσω. Στις περιπτώσεις έντονων καιρικών φαινομένων, συνήθως εκτοξεύονται συχνότερα μετεωρολογικά μπαλόνια για τη συλλογή πρόσθετων και πιο ακριβή δεδομένων σχετικά με τα επερχόμενα φαινόμενα. Στη συνέχεια παρατίθεται μια εικόνα – παράδειγμα, για την πληρέστερη κατανόηση της ραδιοβολίδας.

Εικόνα 3 : Παράδειγμα radiosondes



(Πηγή : guardian.com)

Οι **μετεωρολογικοί σημαντήρες** που βρίσκονται στην επιφάνεια της θάλασσα, αποτελούν όργανα που συλλέγουν μετεωρολογικά και ωκεανογραφικά στις θάλασσες και συγκεκριμένα στους ωκεανούς του κόσμου. Οι μετεωρολογικοί σημαντήρες δεν είναι πρόσφατο εργαλείο αφού η χρήση τους χρονολογείται από το 1951, ενώ οι παρασυρόμενοι μετεωρολογικοί σημαντήρες χρησιμοποιούνται από το 1979. Για τη λειτουργία τους συνδέονται με τον πυθμένα του ωκεανού και χρησιμοποιούν είτε αλυσίδες, νάιλον ή πλευστό πολυπροπυλένιο. Με την πάροδο της χρήσης των μετεωρολογικών πλοίων, έχουν αναλάβει πιο πρωταρχικό ρόλο στη μέτρηση των συνθηκών στην ανοιχτή θάλασσα από τη δεκαετία του 1970. Κατά τη διάρκεια των δεκαετιών του 1980 και του 1990, ένα δίκτυο μετεωρολογικών σημαντήρων στον Ειρηνικό Ωκεανό βοήθησε στη μελέτη της ταλάντωσης φαινομένου Ελ Νίνιο. Οι μετεωρολογικοί σημαντήρες που είναι αγκυροβολημένοι κυμαίνονται από 1,5–12 μέτρα (5–40 ft) σε διάμετρο, ενώ οι μετεωρολογικοί σημαντήρες που δεν είναι (παρασυρόμενοι) είναι μικρότεροι, με διαμέτρους 30–40 εκατοστών (12–16 in). Οι τελευταίοι, είναι η κυρίαρχη μορφή σημαντήρων σε τεράστιο αριθμό (1250 σημαντήρες διάσπαρτοι ανά τον κόσμο). Τα δεδομένα ανέμου από μετεωρολογικούς

σημαντήρες έχουν μικρότερο σφάλμα από αυτά συλλέγονταν από μετεωρολογικά πλοία. Διαφορές παρατηρήθηκαν στις τιμές των μετρήσεων της θερμοκρασίας της επιφάνειας της θάλασσας. Επίσης ένα ο όγκος των πλοίων ήταν καταλυτικός για την λανθασμένη μέτρηση της θερμοκρασίας αφού το πλοίο κατά την έλευσή του συντελούσε στην αύξηση της θερμοκρασίας των υδάτων και άρα σε λανθασμένη πρόβλεψη.

Εικόνα 4: Ενδεικτικός μετεωρολογικός σημαντήρας

(Πηγή : Poseidon System)

Τα **αυτοματοποιημένα συστήματα παρατήρησης επιφανειών** (automated surface observing systems - ASOS) είναι συστήματα που παρακολουθούν συνεχώς την επιφάνεια της γης και της τις καιρικές συνθήκες. Το ASOS είναι μια πρωτοβουλία της Εθνικής Μετεωρολογικής Υπηρεσίας, της Ομοσπονδιακής Υπηρεσίας Αεροπορίας και του Υπουργείου Άμυνας των ΗΠΑ. Είναι το βασικό δίκτυο παρατήρησης καιρού επιφανείας στις ΗΠΑ. Το ASOS έχει σχεδιαστεί για να μπορεί να υποστηρίζει

δραστηριότητες πρόγνωσης καιρού και αεροπορικές επιχειρήσεις και ταυτόχρονα, να υποστηρίζει τις ανάγκες των μετεωρολογικών, υδρολογικών και κλιματολογικών ερευνητικών κοινοτήτων. Στις ΗΠΑ υπάρχουν περισσότεροι από 900 σταθμοί οι οποίοι συλλέγουν συνεχώς δεδομένα που αφορούν τις συνθήκες του ουρανού, την ορατότητα της επιφάνειας, τις βροχοπτώσεις, τη θερμοκρασία και τον άνεμο. Οι σταθμοί αυτοί συνεργάζονται με συστήματα τα οποία συλλέγουν πρόσθετα δεδομένα θερμοκρασίας, χιονόπτωσης και βροχοπτώσεων. Τα δεδομένα παρατήρησης που συλλέγει το ASOS και τα βοηθητικά συστήματα είναι απαραίτητα για τη βελτίωση των προβλέψεων και των προειδοποιήσεων για επικίνδυνα καιρικά φαινόμενα.

Εικόνα 5: Παράδειγμα ASOS

(Πηγή: National Weather Service)

Οι υπερ-υπολογιστές (supercomputers) που χρησιμοποιούνται στα συστήματα πρόβλεψης καιρού θεωρούνται η ραχοκοκαλιά της σύγχρονης πρόβλεψης. Με υπολογιστική ικανότητα 5,78 petaflop έχουν τη δυνατότητα επεξεργασίας τετρά-δισεκατομμύρια υπολογισμούς ανά δευτερόλεπτο. Δηλαδή, οι υπερυπολογιστές αυτοί είναι σχεδόν 6 εκατομμύρια φορές πιο ισχυροί από τον μέσο υπολογιστή που έχει ένας μέσος άνθρωπος. Τα δεδομένα παρατήρησης που συλλέγονται από ραντάρ doppler, ραδιοφωνικούς ήχους, δορυφόρους καιρού, σημαδούρες και άλλα όργανα

τροφοδοτούνται σε υπολογιστικά μοντέλα αριθμητικής πρόγνωσης. Τα μοντέλα χρησιμοποιούν εξισώσεις, μαζί με νέα και προηγούμενα δεδομένα καιρού, για να παρέχουν βοηθητικές πληροφορίες για την εγκυρότερη πρόβλεψη των μετεωρολόγων. Στην πραγματικότητα, οι υπερυπολογιστές δεν είναι ξεχωριστό εργαλείο πρόβλεψης καιρικών φαινομένων αλλά είναι ένα εργαλείο που συνεργάζεται με όλα τα υπόλοιπα. Από τα προηγούμενα αντιλαμβανόμαστε ότι η λειτουργία τους είναι απαραίτητη για την χρήση όλων των προηγούμενων συστημάτων.

Τέλος, το **AWIPS** (**Advanced Weather Information Processing System**) είναι ένα τεχνολογικά προηγμένο λειτουργικό σύστημα επεξεργασίας που χρησιμοποιείται από τους υπερ-υπολογιστές για να τρέξει τα δεδομένα και να τα παρουσιάσει σε γραφικές παραστάσεις που είναι εύκολα κατανοητές. Στο AWIPS συνδυάζονται τα δεδομένα που συλλέχθηκαν από όλα τα προηγούμενα εργαλεία, εισάγονται σε σχετικούς αλγορίθμους και με το χειρισμό των μετεωρολόγων μπορούν να αναλυθούν ως δεδομένα. Με αυτό τον τρόπο οι μετεωρολόγοι εκδίδουν προβλέψεις, χρονοδιαγράμματα καιρικών φαινομένων, προειδοποιήσεις για επικίνδυνες καταστάσεις, κ.α.. Αυτό το σύστημα χρησιμοποιείται από τους υπερυπολογιστές για την επεξεργασία δεδομένων από ραντάρ doppler, ραδιοφωνικούς ήχους, δορυφόρους καιρού, ASOS και άλλες πηγές χρησιμοποιώντας μοντέλα και μεθόδους πρόβλεψης. Αφού οι μετεωρολόγοι προετοιμάσουν τις προβλέψεις, το AWIPS δημιουργεί γραφικά καιρού και επισημαίνει τους χρόνους επικινδυνότητας και τις σχετικές προειδοποιήσεις για τον καιρό. Όλα αυτά βοηθούν τους μετεωρολόγους να δημιουργούν πιο έγκυρες και ακριβείς προβλέψεις και σε μεγάλες χρονικές ταχύτητες.

3. Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα (neural networks) αποτελούν εδώ και χρόνια μια επιστήμη αιχμής που αναπτύσσεται διεθνώς. Σημαντικές ανακαλύψεις καθώς και εφαρμογές της διευκόλυνσης της καθημερινότητας, έχουν βασιστεί σε εφαρμογές των νευρωνικών δικτύων που ήδη υπάρχουν γύρω μας. Οι πρώτες ιδέες ξεκίνησαν από τους βιολογικούς οργανισμούς και ιδιαίτερα από το νευρικό σύστημα του ανθρώπου, αλλά η μελέτη και η χρήση τους έχει προχωρήσει πολύ περισσότερο Σήμερα τα νευρωνικά δίκτυα χρησιμοποιούνται για να λύσουν κάθε είδους προβλήματα μέσω ηλεκτρονικού υπολογιστή. Η φιλοσοφία τους όμως είναι λίγο διαφορετική από τον κλασικό τρόπο με τον οποίο δουλεύουν οι υπολογιστές: Τα νευρωνικά δίκτυα λειτουργούν με παρόμοιο τρόπο σκέψης με τον εγκέφαλο του ανθρώπου, δηλαδή με έναν μαθηματικό και εφαρμοσμένο τρόπο σκέψης. Το νευρωνικό δίκτυο μαθαίνει, εκπαιδεύεται, απομνημονεύει αριθμητικές τιμές, ξεχνάει και θυμάται με την πάροδο του χρόνου. Παράλληλα, χρησιμοποιούν περίπλοκα μαθηματικά εργαλεία, κυρίως από τη μαθηματική ανάλυση για τους υπολογισμούς που διεξάγουν.

Σκοπός της εφαρμογής των ΤΝΔ είναι να εκτιμηθεί αν και κατά πόσο αυτά μπορούν να έχουν την παρόμοια λειτουργία με τον άνθρωπο μέσα από παρόμοιες ακολουθίες υπολογισμών. Δηλαδή, αν μπορούν να μεταδώσουν πληροφορίες σχετικά με τα ερεθίσματα που λαμβάνουν και να μετασχηματίσουν ερεθίσματα σε μια απόκριση ανάλογη με αυτή που εμείς παράγουμε όταν διαβάζουμε μια λέξη δυνατά, ή αναγνωρίζουμε ένα πρόσωπο από μία νέα γωνία, ή εξάγουμε ένα συμπέρασμα. Τα Τεχνητά Νευρωνικά Δίκτυα (ΤΝΔ) είναι προγράμματα για ηλεκτρονικούς υπολογιστές που προσομοιάζουν την οργάνωση και τη λειτουργία των βιολογικών νευρώνων. Ένα συνηθισμένο – τυπικό μοντέλο συγκροτείται από διάφορα επίπεδα στα οποία πραγματοποιείται η επεξεργασία. Η μονάδα επεξεργασίας μπορεί να θεωρηθεί ως ένας πραγματικός νευρώνας, ή ομάδα νευρώνων. Κάθε μονάδα αθροίζει πληροφορία από άλλες μονάδες, εκτελεί ένα απλό υπολογισμό σε αυτό το άθροισμα και περνά το αποτέλεσμα σε άλλες μονάδες. Η επίδραση μιας μονάδας σε μια άλλη εξαρτάται από τη βαρύτητα της μεταξύ τους διασύνδεσης. Τα βασικά στοιχεία αυτού του μοντέλου είναι τα εξής:

1. Ένα σύνολο από συνάψεις (synapses) ή συνδέσεις (connections), κάθε μια από τις οποίες χαρακτηρίζεται από ένα «βάρος» (weight) (πολλές φορές καλείται και συναπτικό βάρος – synaptic weight). Συγκεκριμένα, ένα σήμα xi στην είσοδο της σύναψης i πολλαπλασιάζεται με το αντίστοιχο συναπτικό βάρος wi
2. Έναν αθροιστή (summer – summing junction) για την πρόσθεση των «ζυγισμένων» σημάτων εισόδου. Μια συνάρτηση ενεργοποίησης κατωφλίου (ή απλά η συνάρτηση ενεργοποίησης, επίσης γνωστή ως συνάρτηση σύνθλιψης) οδηγεί σε σήμα εξόδου μόνο όταν ένα σήμα εισόδου που υπερβαίνει μια συγκεκριμένη τιμή κατωφλίου έρχεται ως είσοδος.
3. Ένα εξωτερικά εφαρμοζόμενο threshold, που διαφέρει ανάμεσα στα δίκτυα που οδηγεί σε σήμα εξόδου μόνο όταν ένα σήμα εισόδου που υπερβαίνει μια συγκεκριμένη τιμή threshold ως είσοδο.
4. Μια συνάρτηση ενεργοποίησης (activation function) g() για την παραγωγή της εξόδου του νευρώνα. Η συνάρτηση ενεργοποίησης μπορεί να είναι γραμμική ή μη-γραμμική. Σκοπός της μη γραμμικής συνάρτησης είναι να εξασφαλίζει ότι το νευρώνιο αποκρίνεται εντός συγκεκριμένου πεδίου τιμών. Αυτό γίνεται και στα βιολογικά νευρώνια: το ανθρώπινο αυτί για να αισθανθεί ένα ήχο ως διπλάσιας έντασης, πρέπει το πλάτος του ήχου να αυξηθεί περίπου δέκα φορές.

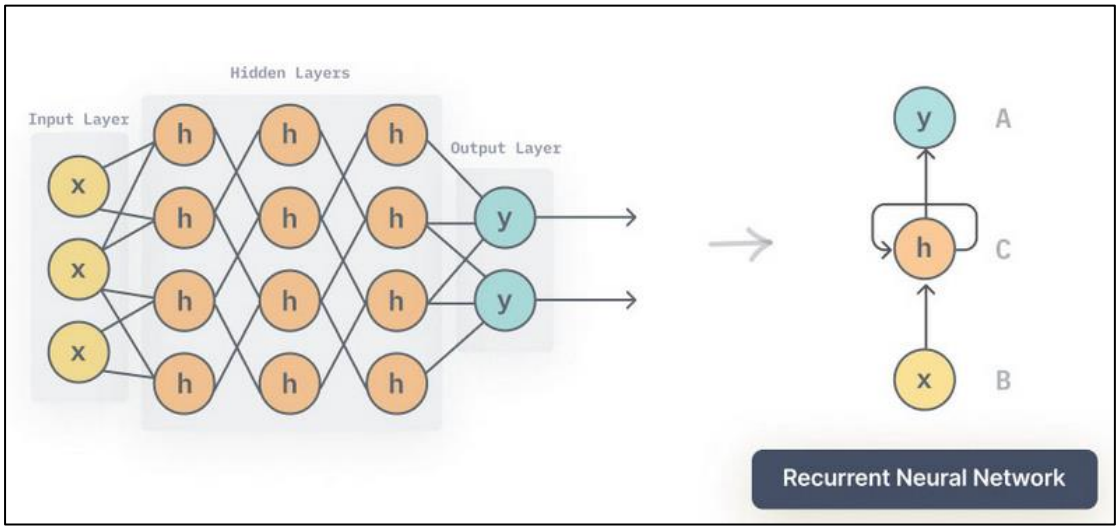
Η ιδιότητα, η οποία είναι πρωτεύουσας σημασίας για ένα νευρωνικό δίκτυο είναι η ικανότητα του δικτύου να μαθαίνει από το περιβάλλον του και να βελτιώνει την απόδοσή του μέσω της εκπαίδευσης. Η βελτίωση στην απόδοση λαμβάνει χώρα πάνω στον χρόνο σε συμφωνία με κάποιο προκαθορισμένο μέτρο. Ένα νευρωνικό δίκτυο μαθαίνει για το περιβάλλον του μέσω μιας αλληλεπιδραστικής διαδικασίας από ρυθμίσεις που εφαρμόζονται στα επίπεδα των συναπτικών τους βαρών και των μεροληψιών. Ιδανικά, το δίκτυο γίνεται πιο γνωστικό για το περιβάλλον του μετά από κάθε επανάληψη της διαδικασίας εκπαίδευσης. Υπάρχουν πάρα πολλές δραστηριότητες που συνδέονται με την έννοια της εκπαίδευσης που δικαιολογεί τον ορισμό της με ένα ακριβή τρόπο. Επιπλέον, η διαδικασία της εκπαίδευσης είναι ένα θέμα σκοπιάς, το οποίο κάνει ακόμη δυσκολότερη τη συμφωνία για έναν ακριβή ορισμό του όρου. Για παράδειγμα, η εκπαίδευση από την σκοπιά των ψυχολόγων είναι αρκετά διαφορετική από την εκπαίδευση που πραγματοποιείται σε μια αίθουσα. Ένας από τους ορισμούς που έχουν δοθεί για την εκπαίδευση των νευρωνικών δικτύων και χρησιμοποιείται ευρέως ακόμη και σήμερα είναι ο εξής: Η εκπαίδευση (training) είναι

μια διαδικασία από την οποία οι ελεύθερες παράμετροι ενός νευρωνικού δικτύου προσαρμόζονται μέσω μιας διαδικασίας διέγερσης από το περιβάλλον στο οποίο είναι εμπεδωμένο το δίκτυο (Mendel και McLaren, 1970). Ο τύπος της εκπαίδευσης καθορίζεται από τον τρόπο με τον οποίο πραγματοποιούνται αλλαγές στις παραμέτρους.

3.1. Επαναλαμβανόμενο Νευρωνικό Δίκτυο (RNN)

Ένα επαναλαμβανόμενο νευρωνικό δίκτυο (RNN) είναι μια κατηγορία ΤΝΔ όπου οι συνδέσεις μεταξύ κόμβων μπορούν να δημιουργήσουν έναν κύκλο, επιτρέποντας στην έξοδο από ορισμένους κόμβους να επηρεάσουν την επακόλουθη είσοδο στους ίδιους κόμβους. Αυτό του επιτρέπει να επιδεικνύει χρονική δυναμική συμπεριφορά. Τα RNN μπορούν να χρησιμοποιήσουν την μνήμη τους για να επεξεργαστούν ακολουθίες μεταβλητού μήκους εισόδων (Abiodun et al., 2018). Αυτό τις καθιστά εφαρμόσιμες σε εργασίες όπως η αναγνώριση γραφής και η αναγνώριση ομιλίας (Graves et al., 2008; Sak et al., 2014) Τα επαναλαμβανόμενα νευρωνικά δίκτυα θεωρητικά είναι ολοκληρωμένα δίκτυα τύπου Turing και μπορούν να εκτελούν προγράμματα για την επεξεργασία ακολουθιών εισόδων αυθαίρετα.[7]

Εικόνα 6: Παράδειγμα Επαναλαμβανόμενου Νευρωνικού Δικτύου



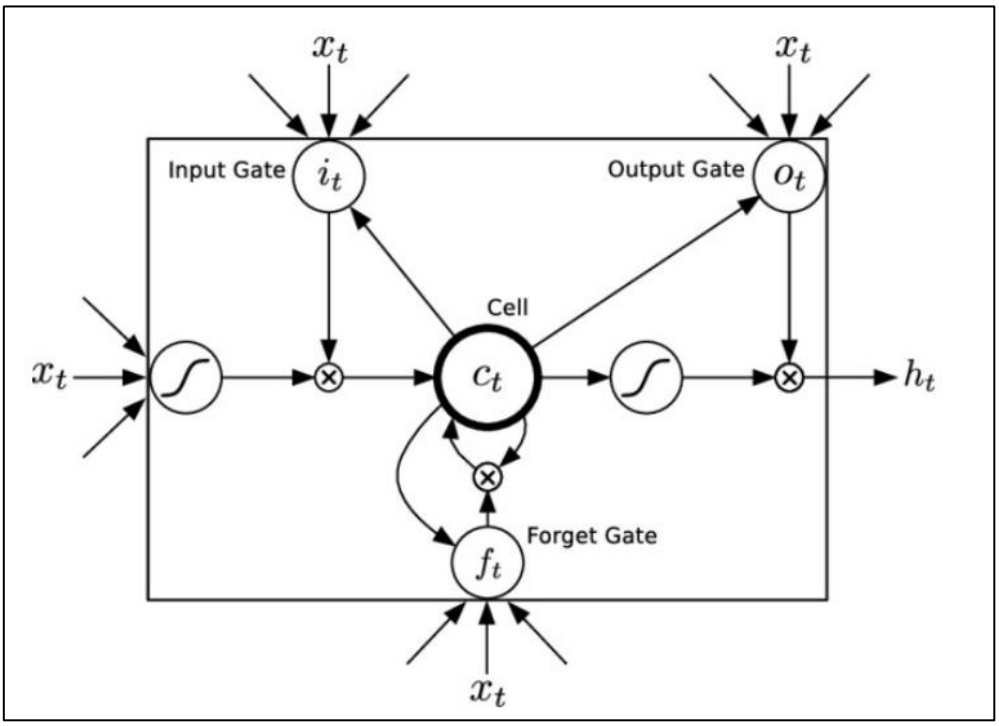
(Πηγή: v7labs.com)

Η ονομασία του δικτύου χρησιμοποιείται για να αναφέρεται στην κατηγορία των δικτύων με άπειρη παλμική απόκριση, ενώ άλλες κατηγορίες δικτύων όπως τα Συνελικτικά και τα Βραχυπρόθεσμης μνήμης χαρακτηρίζονται από την απόκριση πεπερασμένων παλμών. Συνεπώς, γίνεται αντιληπτό ότι ο χρόνος είναι ένας σημαντικός παράγοντας για τη συμπεριφορά των δικτύων. Ένα επαναλαμβανόμενο δίκτυο πεπερασμένου παλμού είναι ένα κατευθυνόμενο μη-κυκλικό γράφημα (Εικόνα 6) που μπορεί να αναπτύσσεται και να αντικατασταθεί με ένα νευρωνικό δίκτυο το οποίο κινείται προς μια κατεύθυνση αυστηρά, ενώ ένα επαναλαμβανόμενο δίκτυο άπειρων παλμών είναι ένα κατευθυνόμενο κυκλικό γράφημα που δεν μπορεί να αναπτυχθεί περεταίρω. Τόσο τα δίκτυα πεπερασμένων παλμών όσο και τα επαναλαμβανόμενα δίκτυα άπειρων παλμών μπορούν να έχουν πρόσθετες αποθηκευμένες καταστάσεις με τρόπο τέτοιο ώστε το δίκτυο να ελέγχει άμεσα την αποθήκευση. Ο χώρος αποθήκευσης μπορεί επίσης να αντικατασταθεί από άλλο δίκτυο ή γράφημα εάν ενσωματώνει χρονικές καθυστερήσεις ή έχει βρόχους ανάδρασης.

3.2.Νευρωνικά Δίκτυα Μακράς Βραχύχρονης Μνήμης (LSTM)

Τα δίκτυα μακράς βραχυπρόθεσμης μνήμης (LSTM) είναι κατηγορία ΤΝΔ που χρησιμοποιείται στο πεδίο της τεχνητής νοημοσύνης και της βαθιάς μάθησης. Εφευρέθηκαν από τους Hochreiter και Schmidhuber (1997). Διαθέτουν συνδέσεις ανάδρασης ώστε να μπορεί να επεξεργαστεί όχι μόνο μεμονωμένα σημεία δεδομένων, αλλά και ολόκληρες ακολουθίες δεδομένων. Το όνομα του χρησιμοποιείται για να δείξει ότι το δίκτυο έχει ταυτόχρονα μακροπρόθεσμη και βραχυπρόθεσμη μνήμη. Τα βάρη σύνδεσης και το δυναμικό (bias) στο δίκτυο αλλάζουν μία φορά ανά στάδιο, ανάλογα με το πώς οι φυσιολογικές αλλαγές στις συναπτικές δυνάμεις αποθηκεύουν μακροπρόθεσμες μνήμες. Η Elman (1990) αναφέρει ότι τα μοτίβα ενεργοποίησης στο δίκτυο αλλάζουν μία φορά ανά χρονικό βήμα, ανάλογα με την αποθήκευση βραχυπρόθεσμων μνημών βάση των αλλαγών στα μοτιβα ηλεκτρικής πυροδότησης. Η αρχιτεκτονική δόμηση των δικτύων LSTM έχει στόχο να παρέχει μια βραχυπρόθεσμη μνήμη για το δίκτυο που μπορεί να διαρκέσει χιλιάδες χρονικά βήματα, άρα η βραχυπρόθεσμη μεταλλάσσεται σε μακροπρόθεσμη μνήμη. Στις μέρες μας, τα δίκτυα αυτά βρίσκουν εφαρμογή σε αναγνώριση φωνής, ομιλίας, γραφής, κειμένων , κ.α.

Εικόνα 7: Παράδειγμα Δικτύου Μακράς Βραχυπρόθεσμης Μνήμης



(Πηγή: Graves et al., 2013)

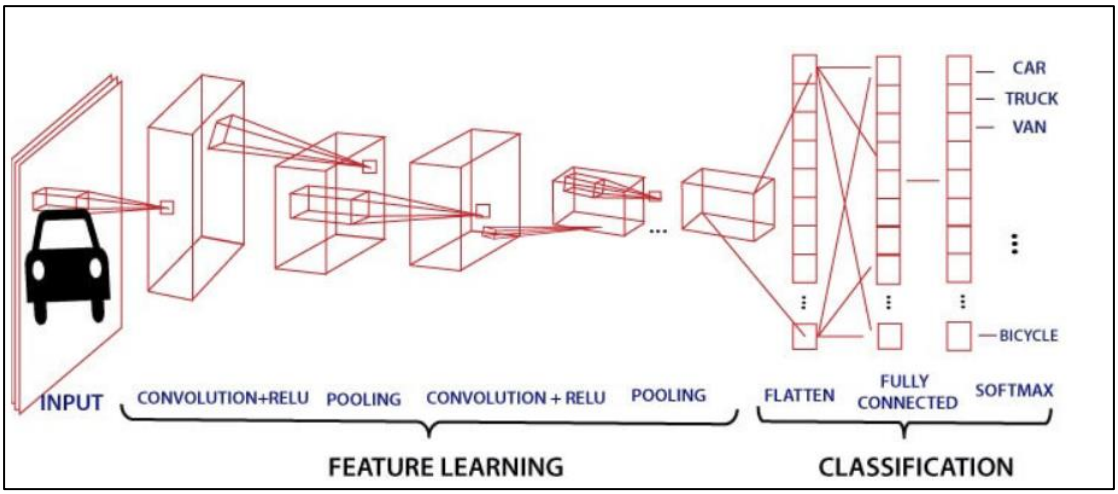
3.3.Συνελικτικό Νευρωνικό Δίκτυο (CNN)

Τα συνελικτικά δίκτυα είναι ακόμα μια κατηγορία ΤΝΔ στη βαθιά μάθηση (deep learning), που χρησιμοποιείται κυρίως για την ανάλυση εικόνας. Η μορφή τους προσομοιάζει τις βιολογικές διεργασίες (Hubel and Wiesel, 1970) στο γεγονός ότι ο τρόπος που συνδέονται οι νευρώνες είναι παρόμοιος με την οργάνωση του οπτικού φλοιού των ζώων. Βασίζονται στην αρχιτεκτονική κοινόχρηστου βάρους των πυρήνων συνέλιξης ή των φίλτρων που ολισθαίνουν κατά μήκος των χαρακτηριστικών εισόδου και παρέχουν αποκρίσεις ισοδύναμης μετάφρασης, γνωστές ως χάρτες χαρακτηριστικών. Τα συνελικτικά δίκτυα είναι φτιαγμένα για να εκμεταλλεύονται τις δομές χώρου ή χρόνου στα δεδομένα εισόδου. Διαθέτουν εφαρμογές οι οποίες χρησιμεύον στην αναγνώριση και την ταξινόμηση εικόνας και βίντεο, συστήματα διαδικασίας λήψης αποφάσεων, , ανάλυση ιατρικών εικόνων, επεξεργασία γλώσσας και διαλέκτου, σύνδεση εγκεφάλου με υπολογιστές και οικονομικές χρονοσειρές.

Στα CNN ο κάθε νευρώνας στο κάθε στρώμα συνδέεται με όλους τους νευρώνες στο επόμενο στρώμα (Goodfellow, 2016). Αυτό το πλέγμα από νευρώνες και η "πλήρης συνδεσιμότητα" αυτών των δικτύων καθιστά τα συνελικτικά δίκτυα επιρρεπή στην υπερπροσαρμογή δεδομένων, γεγονός που θα πρέπει ο χρήστης να προσέξει. Τυπικές μέθοδοι τακτοποίησης ή αποτροπής της υπερβολικής προσαρμογής περιλαμβάνουν: τιμωρία παραμέτρων κατά τη διάρκεια της προπόνησης (όπως μείωση βάρους) ή περικοπή κάποιας συνδεσιμότητας (παραλείψεις συνδέσεων, εγκατάλειψη κ.λπ.) Τα CNN υιοθετούν μια διαφορετική προσέγγιση προς την ταξινόμηση: εκμεταλλεύονται το ιεραρχικό μοτίβο στα δεδομένα και συναρμολογούν μοτίβα αυξανόμενης πολυπλοκότητας χρησιμοποιώντας μικρότερα και απλούστερα μοτίβα ανάγλυφα στα φίλτρα τους. Συνεπώς, στην κλίμακα συνδεσιμότητας - πολυπλοκότητας, τα συνελικτικά βρίσκονται στην κατώτερη θέση.

Τα συνελικτικά δίκτυα χρησιμοποιούν σχετικά λίγη προ-επεξεργασία σε σύγκριση με άλλους αλγόριθμους ταξινόμησης εικόνων. Αυτό σημαίνει ότι το δίκτυο μαθαίνει να βελτιστοποιεί τα φίλτρα (ή τους πυρήνες) μέσω της αυτοματοποιημένης εκμάθησης, ενώ στους παραδοσιακούς αλγόριθμους αυτά τα φίλτρα είναι κατασκευασμένα από τον χρήστη (Zaki and Meira, 2020). Αυτή η ανεξαρτησία από την προηγούμενη γνώση και την ανθρώπινη παρέμβαση στην εξαγωγή χαρακτηριστικών είναι ένα σημαντικό πλεονέκτημα. Η παρακάτω εικόνα απεικονίζει τη λειτουργία ενός συνελικτικού νευρωνικού δικτύου που μετατρέπει μια εικόνα σε πολυδιάστατου πίνακες και σταδιακά την αναγνωρίζει.

Εικόνα 8: Συνελικτικό Δίκτυο



(Πηγή: javapoint.com)

4. Μηχανική Μάθηση

Η Μηχανική μάθηση είναι τμήμα της έννοιας που ονομάζεται τεχνητής νοημοσύνης. Πρόκειται για ένα εκτενές πεδίο μελέτης της κατανόηση και της δημιουργία μεθόδων που «μαθαίνουν», δηλαδή μεθόδους που αξιοποιούν δεδομένα για τη βελτίωση της απόδοσης σε κάποιο σύνολο εργασιών.

Η μηχανική μάθηση εκτός από το πλαίσιο της παρούσας εργασίας, δηλαδή τη μετεωρολογία, εφαρμόζεται και σε διαφορετικούς κλάδους. Στην αεροπορία εφαρμόζεται για τη δημιουργία αυτομάτων συστημάτων πλοηγήσεων υψηλής απόδοσης, προσομοιωτές πτήσεων, για συστήματα αυτομάτου ελέγχου αεροπλάνων και για την ανίχνευση πιθανών βλαβών. Τα συστήματα αυτόματης πλοήγησης έχουν εφαρμογή και στην αυτοκίνηση εκτός από την αεροπορία. Για τους κινητήρες, τα συστήματα αυτά χρησιμοποιούνται για να αναλύσουν τις εισροές των αισθητήρων ενός κινητήρα. Ουσιαστικά το σύστημα ελέγχει τις παραμέτρους με τις οποίες λειτουργεί ο κινητήρας, προκειμένου να επιτευχθεί ένας στόχος (π.χ. ένα σύστημα που στοχεύει στην ελαχιστοποίηση κατανάλωσης καυσίμων).

Στον τραπεζικό τομέα η μηχανική μάθηση εφαρμόζεται για την ανάγνωση επιταγών και άλλων εγγράφων, για οικονομικές αναλύσεις, για χρηματιστηριακές προβλέψεις, και για συστήματα που αξιολογούν τη δανειοληπτική δυνατότητα των χρηστών. Ειδικότερα για τις χρηματιστηριακές προβλέψεις η μηχανική μάθηση χρησιμοποιείται για να αναλύσει τις διακυμάνσεις των τιμών των μετοχών και των συναφών δεικτών.

Στον τομέα της εθνικής ασφάλειας, συστήματα μηχανικής μάθησης δημιουργούν την πλοήγηση οπλικού εξοπλισμού, την ανίχνευση πιθανών στόχων, το σύστημα αισθητήρων, ραντάρ και σόναρ, την ψηφιακή επεξεργασία σημάτων, δεδομένων και κατά την αναγνώριση σημάτων και εικόνων. Στον τομέα της ηλεκτρονικής, χρησιμοποιούνται για την πρόβλεψη μιας ακολουθίας κωδικών, για τη μορφοποίηση και τη διάγνωση βλαβών κυκλωμάτων και τη μηχανική όραση.

Στη βιομηχανία, συστήματα μηχανικής μάθησης ελέγχουν τις διεργασίες, χρησιμοποιούνται για τη διάγνωση βλαβών και άλλες συναφείς λειτουργίες. Στην Ιατρική, ένα τέτοιο σύστημα μπορεί να παρακολουθήσει δεδομένα και ενδείξεις όπως η καρδιακή συχνότητα, να προβεί σε ανάλυση καρκινικών κυττάρων, σε ανάλυση ηλεκτροεγκεφαλογραφήματος και ηλεκτροκαρδιογραφήματος, να παρακολουθεί τα

επίπεδα των διαφόρων ουσιών στο αίμα, ο ρυθμός της αναπνοή, κ.α. Το σύστημα παρακολουθεί τον ασθενή και σε περίπτωση ανάγκης θα αναγνωρίσει την ιατρική κατάσταση, ώστε να χορηγηθεί η κατάλληλη θεραπεία. Σε έρευνες που σχετίζονται με τη γεωλογία, όπως ο εντοπισμός πετρελαίου, φυσικού αερίου ή άλλων κοιτασμάτων χρησιμοποιούνται συστήματα μηχανικής μάθησης.

Στη ρομποτική για τον έλεγχο της κίνησης και το σύστημα όρασης του ρομπότ. Τέλος, στις τηλεπικοινωνίας, η μηχανική μάθηση συμπιέζει την εικόνα και τα δεδομένα, παρέχει αυτοματοποιημένες υπηρεσίες πληροφοριών, χρησιμοποιείται και στα αυτόματα συστήματα πληρωμών.

Η βασική ιδιότητα για ένα δίκτυο σε όλες τις παραπάνω εφαρμογές και τομείς, είναι η μαθησιακή του ικανότητα από το περιβάλλον του, καθώς και η ικανότητά του να βελτιώνει την απόδοσή του μέσω αυτής της μαθησιακής διαδικασίας. Η βελτίωση στην απόδοση του δικτύου πραγματοποιείται μέσω μας διαδικασίας η οποία είναι προγραμματισμένη από έναν χρήστη, διαδικασία ορισμένου χρόνου όπου το δίκτυο εκπαιδεύεται για να μπορεί να αλληλοεπιδρά με το περιβάλλον του. Σκοπός αυτής της μαθησιακής διαδικασίας είναι το δίκτυο να γίνεται ολοένα και περισσότερο γνωστικό για το περιβάλλον μέσω της επανάληψης της διαδικασίας. Για την ολοκλήρωση αυτής της διαδικασίας υπάρχει πληθώρα δραστηριοτήτων που έχουν τη δυνατότητα να εκπαιδεύουν. Η εκπαίδευση είναι μια διαδικασία από την οποία οι παράμετροι ενός δικτύου προσαρμόζονται από το περιβάλλον, το οποίο είναι προκαθορισμένο για το δίκτυο. Η κάθε αλλαγή των παραμέτρων εκπαίδευσης καθορίζει και τον τύπο της εκπαίδευσης του δικτύου. Οι δύο βασικές μορφές εκπαίδευσης είναι η εκπαίδευση με επίβλεψη και η εκπαίδευση χωρίς επίβλεψη.

4.1 Εκπαίδευση με επίβλεψη

Το σύστημα εκπαιδεύεται με τα δεδομένα εισόδου και εξόδου. Τα πρώτα είναι αυτά που εισάγει ο χρήστης στην αρχή της διαδικασίας και το σύστημα έχει την ικανότητα να αναπαράγει τα δεύτερα, τα δεδομένα εξόδου ως αποτέλεσμα της διαδικασίας. Συγκεκριμένα, ο στόχος της εκπαίδευσης είναι η ελαχιστοποίηση των σφαλμάτων και η προσαρμογή των τιμών έτσι ώστε όταν στο δίκτυο διαβιβάζεται η σωστή είσοδος, να αναπαράγει τη σωστή έξοδο. Στην εκπαίδευση με επίβλεψη (supervised learning), θεωρείται ότι ο χρήστης που εισάγει τα δεδομένα έχει ήδη γνώση του περιβάλλοντος

στο οποίο αλληλοεπιδρά το δίκτυο (Mohri et al., 2012). Τη γνώση αυτή τη χρησιμοποιεί για να εισάγει τα κατάλληλα δεδομένα εισόδου και εξόδου. Ταυτόχρονα, το περιβάλλον είναι άγνωστο στο δίκτυο. Στη συνέχεια χρήστης και δίκτυο αλληλοεπιδρούν με το περιβάλλον μέσω ενός παραδείγματος εκπαίδευσης όπου ο δάσκαλος είναι παρέχει στο δίκτυο τις επιθυμητές απαντήσεις για το μαθησιακό παράδειγμα. Στόχος του χρήστη είναι να εκπαιδευτεί το δίκτυο με τρόπο ώστε να παρέχει τις επιθυμητές απαντήσεις για κάθε ενδεχόμενο. Οι παράμετροι του δικτύου ρυθμίζονται σε σχέση με την εκπαιδευτική διαδικασία και με τη διαδικασία του σφάλματος, κατά την οποία ο χρήστης ανατροφοδοτεί το σύστημα. Με αυτό τον τρόπο, ο χρήστης μεταφέρει τις γνώσεις του στο δίκτυο μέσω της μαθησιακής διαδικασίας, όσο το δυνατό πληρέστερα. Κατά την ολοκλήρωση αυτής της διαδικασίας ο χρήστης είναι πλέον περιττός και το δίκτυο μπορεί να συνεχίσει να αλληλοεπιδρά με το περιβάλλον χωρίς την υποστήριξή του. Το σύστημα εκπαιδεύεται να επιλύει προβλήματα μέσω έξι βημάτων. Σε αυτή τη διαδικασία ο χρήστης έχει συνεχώς την εποπτεία και η παρέμβασή του απαιτείται για την επιτυχημένη ολοκλήρωσή της (Alpaydin, 2020).

1. Προσδιορισμός των δεδομένων εκπαίδευσης όπου αποφασίζεται ο τύπος των δεδομένων που θα χρησιμοποιηθούν για την εκπαίδευση του συστήματος.
2. Στη συνέχεια ο χρήστης εισάγει δεδομένα για στα οποία ο αλγόριθμός κάνει δοκιμές. Στο στάδιο αυτά θα πρέπει να εισαχθούν αντιπροσωπευτικά δεδομένα έτσι ώστε να υπάρξει η κατάλληλη εκπαίδευση με μετρήσιμα δεδομένα
3. Προσδιορισμός της συνάρτησης εκμάθησης του αλγορίθμου, φάση η οποία εξαρτάται σε μεγάλο βαθμό από τον τρόπο που εισάγονται τα δεδομένα εισόδου. Συνήθως, τα δεδομένα εισόδου μετατρέπονται σε ένα διάνυσμα, το οποίο περιγράφει τα δεδομένα επαρκώς για την ακρίβεια της πρόβλεψης.
4. Προσδιορισμός της δομής της συνάρτησης εκμάθησης και προσδιορισμός δομής ρου αλγορίθμου.
5. Εκτέλεση μαθησιακού αλγόριθμου όπου ο χρήστης καθορίζει τις παραμέτρους, οι οποίες βελτιώνουν την απόδοση του συστήματος.
6. Αξιολόγηση της διαδικασίας, μετά την προσαρμογή όλων των παραμέτρων και μετά την εκπαίδευση. Η απόδοση της συνάρτησης που προκύπτει υπόκειται σε ένα σύνολο δοκιμών για να μετρηθεί.

4.2 Εκπαίδευση χωρίς επίβλεψη

Η Εκπαίδευση χωρίς επίβλεψη είναι η εναλλακτική διαδικασία που μπορεί να εκπαιδευτεί ένα σύστημα/αλγόριθμος, όπου παρέχονται μονάχα δεδομένα / διανύσματα εισόδου τα οποία ομαδοποιούνται με βάση τα κοινά τους χαρακτηριστικά. Ο στόχος αυτής της κατηγορίας είναι να ελαχιστοποιηθούν τα σφάλματα κατά τα δεδομένα εξόδου. Βασική υπόθεση της διαδικασίας αυτής είναι ότι το σύστημα αποκτά μιμητική ικανότητα και διαμορφώνει μια πραγματικότητα που λειτουργεί εντός της. Σε αντίθεση με την εκπαίδευση με επίβλεψη ο χρήστης δεν επισημαίνει τα δεδομένα με ενδείξεις, ο ίδιος ο αλγόριθμος το πραγματοποιεί. Σε γενικές γραμμές, αυτή η εναλλακτική χρησιμοποιείται για λύσεις περισσότερο περίπλοκες, ακόμα και για αυτές τις οποίες ο χρήστης δεν μπορεί να παρέχει καθοδήγηση στον αλγόριθμο. Η εκπαιδευτική διαδικασία χωρίς επίβλεψη διακρίνεται σε εκπαίδευση με και χωρίς ενίσχυση.

Η διαδικασία με ενίσχυση (reinforcement learning) εκτελείται μέσω της συνεχής αλληλεπίδρασης με το περιβάλλον. Η διαδικασία αυτή περιλαμβάνει την εκπαίδευση ενός συστήματος να υιοθετεί συγκεκριμένες συμπεριφορές, τροφοδοτούμενο με στοιχεία από το περιβάλλον, στο οποίο λειτουργεί (Kaelbling et al., 1996). Η διαδικασία αυτή έχει στόχο να μεγιστοποιήσει το κέρδος σε σχέση με τις δράσεις του εντός του περιβάλλοντός του. Σε αυτή τη διαδικασία το σύστημα μαθαίνει αποκλειστικά από το περιβάλλον και εκπαιδεύεται μόνο του στην αντιμετώπιση διαφόρων καταστάσεων, επεξεργαζόμενο ελάχιστα δεδομένα και πληροφορίες. Από το περιβάλλον λαμβάνει και την «ανταμοιβή» του, η οποία είναι και το μοναδικό στοιχείο που λαμβάνει ως απάντηση για την ανατροφοδότησή του. Σκοπός του συστήματος είναι να μεγιστοποιήσει την ανταμοιβή του. Από τα παραπάνω γίνεται αντιληπτό ότι σε αυτή τη διαδικασία απουσιάζει ο χρήστης ο οποίος θα επιλέξει τη βέλτιστη επιλογή για το σύστημα κατά τη λήψη αποφάσεων. Αντιθέτως, κατά τη διαδικασία χωρίς ενίσχυση (reinforcement learning) απουσιάζει ο εξωτερικός χρήστης ο οποίος επιβλέπει τη διαδικασία. Το δίκτυο επεξεργάζεται τα δεδομένα εισόδου και αναπτύσσει την ικανότητα να σχηματίσει εσωτερικές αναπαραστάσεις για την κωδικοποίηση χαρακτηριστικών της εισόδου. Για να εκτελεστεί εκπαίδευση χωρίς επίβλεψη και χωρίς ενίσχυση πρέπει να χρησιμοποιηθεί ένας ανταγωνιστικός εκπαιδευτικός κανόνας που έχει την ικανότητα να ανταγωνίζεται τον κανόνα εκπαίδευσης και με ικανότητα να αποκρίνεται σε χαρακτηριστικά που περιέχονται στα δεδομένα εισόδου.

5. Μέθοδοι και Τεχνικές Clasification

Η ταξινόμηση είναι μια επεξεργασία φυσικής γλώσσας που εκτελούν αλγόριθμοι μηχανικής μάθησης. Πρόκειται για μια διαδικασία αναγνώρισης, κατανόησης και ομαδοποίησης ιδεών και αντικειμένων σε προκαθορισμένες κατηγορίες - υπό-ομάδες. Για την διαδικασία της ταξινόμησης τα προγράμματα μηχανικής μάθησης χρησιμοποιούν κατηγοριοποιημένα σύνολα δεδομένων εκπαίδευσης και ένα εύρος αλγορίθμων για να ταξινομήσουν μελλοντικά σύνολα σε κατηγορίες. Στη μηχανική μάθηση, οι αλγόριθμοι ταξινόμησης χρησιμοποιούν τα δεδομένα εισόδου για να προβλέψουν την κατηγοριοποίηση των μελλοντικών δεδομένων σε προκαθορισμένες κατηγορίες και την πιθανότητα τους. Ένα παράδειγμα τέτοιας λειτουργίας είναι η καταχώρηση και το φιλτράρισμα των e-mail ενός χρήστη σε βασικά, φόρουμ, κοινωνικά, προσφορές, spam, κλπ. Μερικοί από τους πιο ευρέως χρησιμοποιούμενους αλγόριθμους ταξινόμησης είναι ο K-Nearest Neighbors (KNN), η Γραμμική και η Λογιστική Παλινδρόμηση (Linear / Logistic Regression), το Random Forest ή Decision Forest / Tree, ο Support Vector Machines και ο Naive Bayes.

5.1. K-Nearest Neighbors (KNN)

Η μέθοδος K-Nearest Neighbors (k-NN) είναι ένας μη-παραμετρικός αλγόριθμος αναγνώρισης προτύπων που χρησιμοποιεί σύνολα δεδομένων εκμάθησης για να βρει τις πιο σχετικά κοντινές ομάδες σε μελλοντικά παραδείγματα. Όταν το KNN χρησιμοποιείται στην ταξινόμηση, υπολογίζεται η τοποθέτηση δεδομένων στην κατηγορία του πλησιέστερου γειτονικού. Αν k = 1, τότε θα τοποθετηθεί στην κλάση πλησιέστερη του 1. Το K ταξινομείται από μια διαδικασία εύρεσης των πλησιέστερων. Συνοπτικά, ο αλγόριθμος εκτελεί τα παρακάτω βήματα (Bijalwan et al., 2014):

1. Δημιουργεί διάνυσμα για κάθε έγγραφο στο δοκιμαστικό σύνολο.
2. Δημιουργεί το διάνυσμα του κέντρου για κάθε τάξη.
3. Υπολογίζει την ομοιότητα μεταξύ κάθε διανύσματος εγγράφου και διανύσματος κλάσης.
4. Το έγγραφο ανήκει στην κατηγορία για την οποία η ομοιότητα είναι μέγιστη.

Ο Meersman (2003) αναφέρει ότι για την βελτίωση της ακρίβειας ταξινόμησης μπορούν να εφαρμοστούν δύο διαφορετικές μέθοδοι εντός του αλγορίθμου στο μοντέλο που ταξινομεί με KNN. Η πρώτη μέθοδος είναι η αφαίρεση των αντιπροσωπευτικών από το μοντέλο που καλύπτει μόνο μερικές ακολουθίες δεδομένων και τις σχετικές ακολουθίες δεδομένων που καλύπτονται από αυτούς τους αντιπροσώπους από το σύνολο δεδομένων εκπαίδευσης και στη συνέχεια να ανακατασκευάζεται το μοντέλο από το αναθεωρημένο σύνολο δεδομένων εκπαίδευσης. Η δεύτερη μέθοδος αφορά μια τροποποίηση του βήματος 3 που παρουσιάστηκε παραπάνω στον αλγόριθμο κατασκευής του μοντέλου για να επιτρέψει σε κάθε μεγαλύτερη τοπική κάλυψη r (ονομάζεται βαθμός ανοχής σφάλματος) ακολουθίες δεδομένων με διαφορετικές κατηγορίες στην πιο πλειοψηφική κατηγορία σε αυτήν τη γειτονιά. Αυτή η τροποποίηση ενσωματώνει τις εργασίες «κλαδέματος» στη διαδικασία κατασκευής μοντέλου (Meersman, 2003). Ένα αρνητικό σημείο για τον αλγόριθμο που αναφέρεται συχνά στη βιβλιογραφία, είναι το γεγονός ότι απαιτεί την αποθήκευση ολόκληρου του σετ εκπαίδευσης η οποία ενδεχομένως να είναι αρκετά δαπανηρή όταν αυτό το σετ είναι μεγάλο, ερευνητές προσπάθησαν να απαλλαγούν από το σχετικό πλεονασμό του σετ εκπαίδευσης για να μετριάσουν την επίδραση αυτού του προβλήματος (Alpaydin, 1997; Hart, 1968).

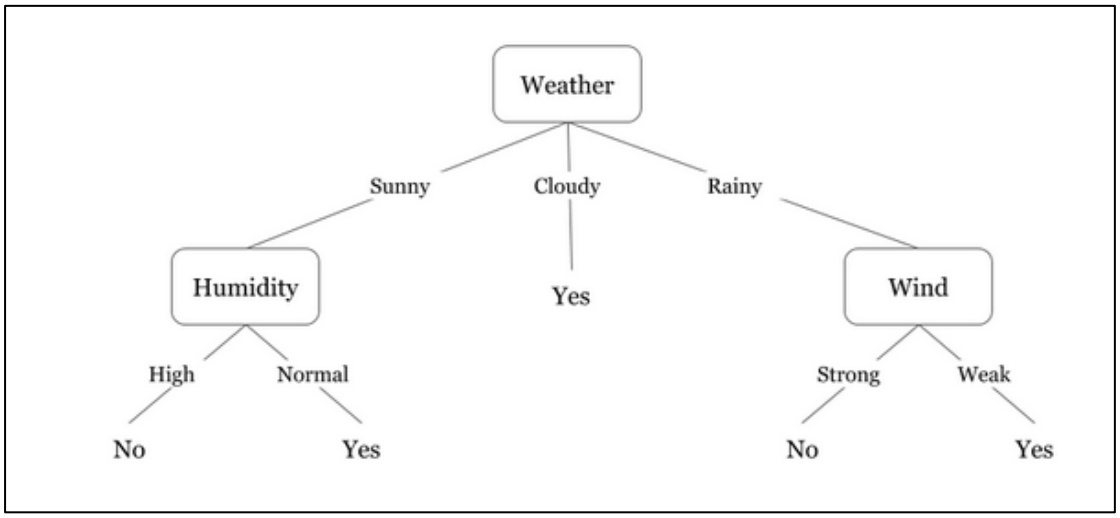
5.2. Γραμμική και η Λογιστική Παλινδρόμηση (Linear / Logistic Regression)

Η παλινδρόμηση είναι ίσως η πιο γνωστή και κατανοητή μέθοδος για έναν αλγόριθμο, λόγω του ότι είναι γνωστή από τη στατιστική. Στη στατιστική, η παλινδρόμηση είναι μια μέθοδος που χρησιμοποιείται για να προσδιορίσει τη δύναμη και τον χαρακτήρα της σχέσης μεταξύ μιας εξαρτημένης μεταβλητής (που συνήθως υποδηλώνεται με Y) και μιας σειράς άλλων μεταβλητών (γνωστές ως ανεξάρτητες μεταβλητές). Τέτοιες μέθοδοι χρησιμοποιούνται ευρέως στα χρηματοοικονομικά, στις επενδύσεις και σε άλλους κλάδους για την δημιουργία προβλέψεων με κάποια δεδομένα μεταβλητών.

5.3.Random Forest (Decision Forest) και Decision Tree

Το δέντρο αποφάσεων είναι ένας εποπτευόμενος αλγόριθμος μάθησης που είναι ιδανικός για προβλήματα ταξινόμησης, καθώς μπορεί να παραγγέλνει τάξεις σε ένα ακριβές επίπεδο. Προσομοιάζει τη δομή ενός δέντρου και λειτουργεί σαν ένα διάγραμμα ροής, διαχωρίζοντας τα σημεία δεδομένων σε δύο παρόμοιες κατηγορίες κάθε φορά από τον «κορμό του δέντρου» στα «κλαδιά», στα «φύλλα», όπου οι κατηγορίες γίνονται πιο όμοιες. Αυτό δημιουργεί κατηγορίες εντός κατηγοριών, επιτρέποντας την οργανική ταξινόμηση με περιορισμένη ανθρώπινη επίβλεψη.

Εικόνα 9: Δέντρο αποφάσεων



(Πηγή: Hackerearth.com)

5.4.Support Vector Machines

Η μέθοδος Support Vector Machines είναι ένα εποπτευόμενο μοντέλο μηχανικής μάθησης που χρησιμοποιεί αλγόριθμους ταξινόμησης για προβλήματα ταξινόμησης ανάμεσα σε δύο ομάδες. Αφού ανατεθούν στο μοντέλο SVM σύνολα δεδομένων εκπαίδευσης με ενδείξεις - ετικέτες για κάθε κατηγορία, ο αλγόριθμος είναι ικανός να κατηγοριοποιήσει το νέο κείμενο. Σε σύγκριση με πιο σύγχρονους αλγόριθμους (π.χ. νευρωνικά δίκτυα) οι SVM διαθέτουν δύο βασικά πλεονεκτήματα: υψηλότερη ταχύτητα και καλύτερη απόδοση με περιορισμένο αριθμό δειγμάτων. Αυτά τους καθιστούν πολύ κατάλληλους για προβλήματα ταξινόμησης κειμένου, όπου είναι σύνηθες να υπάρχει πρόσβαση σε ένα σύνολο δεδομένων με μερικές χιλιάδες δείγματα

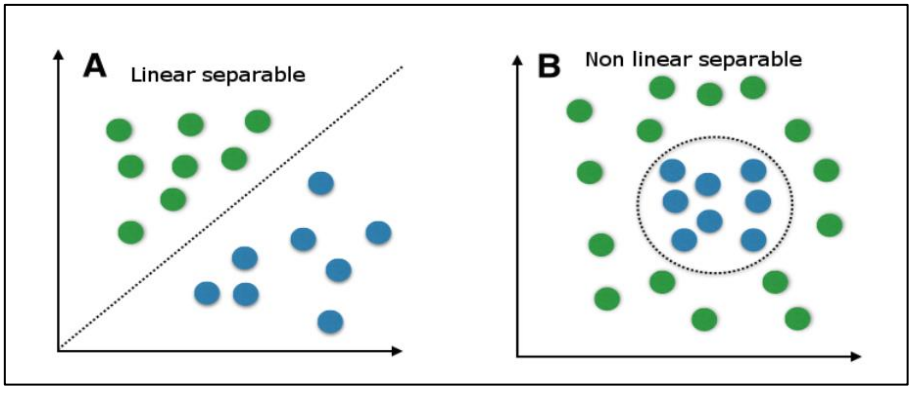
με ενδείξεις. Οι θεμελιωτές των SVM θεωρούνται οι Vapnik and Chervonenkis (1982) και Cortes and Vapnik (1995). Ο υπολογισμός του της ταξινόμησης SVM ισοδυναμεί με ελαχιστοποίηση μιας συνάρτηση τύπου:

Εικόνα 10: SVM συνάρτηση

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i - b)\right)\right] + \lambda\|\mathbf{w}\|^2.$$

Οι αλγόριθμοι ταξινόμησης SVM μπορεί να είναι δύο τύπων: Γραμμικοί και Μη-Γραμμικοί ταξιμομιτές. Ο γραμμικός χρησιμοποιείται για δεδομένα που μπορούν να διαχωριστούν γραμμικά, γεγονός που σημαίνει ότι εάν ένα σύνολο δεδομένων μπορεί να ταξινομηθεί σε δύο κατηγορίες χρησιμοποιώντας μια ευθεία γραμμή, τότε αυτά τα δεδομένα ονομάζονται γραμμικά διαχωρίσιμα δεδομένα. Ο μη-γραμμικός χρησιμοποιείται για μη γραμμικά διαχωρισμένα δεδομένα, γεγονός που σημαίνει ότι εάν ένα σύνολο δεδομένων δεν μπορεί να ταξινομηθεί χρησιμοποιώντας μια ευθεία γραμμή, τότε αυτά τα δεδομένα ονομάζονται μη γραμμικά δεδομένα. Παρακάτω παρατίθενται γραμμικοί και μη – γραμμικοί αλγόριθμοι ταξινόμησης SVM**.** Ο στόχος του SVM είναι να δημιουργήσει την καλύτερη γραμμή ή όριο απόφασης που διαχωρίζει το χώρο n-διαστάσεων σε κλάσεις, ώστε να μπορούν οι χρήστες εύκολα να τοποθετήσουν το νέο σημείο δεδομένων στη σωστή κατηγορία σε επόμενη φάση. Αυτό το όριο καλύτερης απόφασης ονομάζεται υπερεπίπεδο - hyperplane (Shao et al., 2014).

Εικόνα 11: Παράδειγμα γραμμικού και μη γραμμικού αλγόριθμου SVM



(Πηγή: medium.com)

5.5.Naive Bayes

Η μέθοδος Naive Bayes υπολογίζει την πιθανότητα ένα σημείο δεδομένων ανήκει σε μια συγκεκριμένη κατηγορία ή όχι. Στην ανάλυση κειμένου, μπορεί να χρησιμοποιηθεί για να κατηγοριοποιήσει λέξεις ή φράσεις ως ανήκουν σε μια προκαθορισμένη «ετικέτα» που χρησιμοποιείται για την ταξινόμηση. Για να αποφασίσει ο αλγόριθμος αν η φράση – λέξη ανήκει στην ετικέτα, χρειάζεται να υπολογίσει την παρακάτω πιθανότητα. Συνοπτικά, η πιθανότητα του Α, αν το Β είναι αληθές, ισούται με την πιθανότητα του Β, αν το Α είναι αληθές, πολλαπλασιάζει την πιθανότητα το Α να είναι αληθές, διαιρούμενο με την πιθανότητα να είναι αληθές το Β:

Εικόνα 12: Naïve Bayes συνάρτηση

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Ο αλγόριθμος εκτελεί τέσσερις ενέργειες ως βήματα: Ελέγχει τη λέξη-κλειδί στο έγγραφο δοκιμής και την αποθηκεύει. Υπολογίζει τη συχνότητα θετικής και αρνητικής απάντησης (ναι και όχι) για κάθε λέξη-κλειδί στο έγγραφο δοκιμής. Υπολογίζει την πιθανότητας κάθε λέξης κλειδιού του εγγράφου δοκιμής. Ταξινομεί το Εγγράφου Δοκιμής σε διάφορες κατηγορίες με βάση την πιθανότητα που υπολογίστηκε παραπάνω. Ένα πλεονέκτημα της μεθόδου ταξινόμησης Naive-Bayes είναι ότι έχει χαμηλές απαιτήσεις σε αριθμό δεδομένων εκπαίδευσης για την εκτίμηση των παραμέτρων που είναι απαραίτητες για τη διαδικασία ταξινόμησης.

6. Μέθοδοι και τεχνικές Clustering

Η ανάλυση clustering (ομαδοποίηση ή ανάλυση συστάδων) είναι μια έννοια που έχει τις ρίζες της στην ανθρωπολογία από τους Driver και Kroeber (1932) και εισήχθη στην ψυχολογία από τον Zubin (1938) (Καλτσάς, 2015). Έπειτα χρησιμοποιήθηκε από τον Cattell (1943) για την ταξινόμηση της θεωρίας χαρακτηριστικών στην ψυχολογία της προσωπικότητας. Η ανάλυση συστάδων είναι η διαδικασία της ομαδοποίησης ενός συνόλου αντικειμένων με τρόπο ώστε τα αντικείμενα της ίδιας ομάδας (που ονομάζεται σύμπλεγμα) να είναι πιο παρόμοια μεταξύ τους παρά με αυτά σε άλλες ομάδες. Αποτελεί βασική ιδιότητα της διερευνητικής ανάλυσης δεδομένων και παράλληλα μια κοινή τεχνική για στατιστική ανάλυση δεδομένων, που χρησιμοποιείται σε πολλούς τομείς, όπως η αναγνώριση προτύπων, η ανάλυση εικόνας, η ανάκτηση πληροφοριών, η βιοπληροφορική, η συμπίεση δεδομένων, τα γραφικά υπολογιστών και η μηχανική μάθηση. Η ανάλυση συστάδων δεν είναι ένας συγκεκριμένος αλγόριθμος, αλλά η γενική εργασία που πρέπει να επιλυθεί. Μπορεί να επιτευχθεί με διάφορους αλγόριθμους που διαφέρουν σημαντικά ως προς την κατανόησή τους για το τι αποτελεί ένα σύμπλεγμα και τον τρόπο αποτελεσματικής εύρεσης τους. Η ομαδοποίηση των αντικειμένων μπορεί να διατυπωθεί ως ένα πρόβλημα βελτιστοποίησης πολλαπλών στόχων. Ο κατάλληλος αλγόριθμος ομαδοποίησης και οι ρυθμίσεις παραμέτρων εξαρτώνται από το μεμονωμένο σύνολο δεδομένων και την προβλεπόμενη χρήση των αποτελεσμάτων. Τέτοιες παραμέτρους αποτελούν η συνάρτηση απόστασης προς χρήση, το όριο πυκνότητας ή ο αριθμός των αναμενόμενων συμπλεγμάτων.

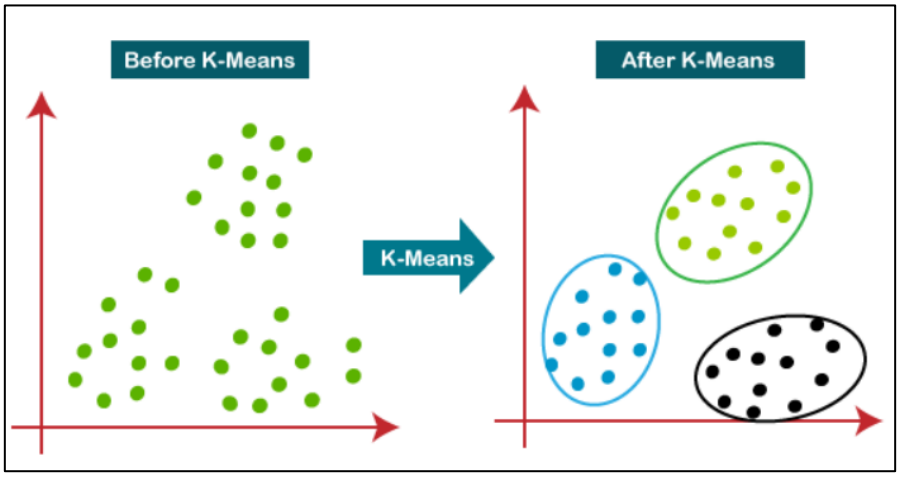
Οι τεχνικές ανάλυσης συστάδων (clustering) δεν είναι μια αυτόματη εργασία, αλλά μια διαδικασία που επαναλαμβάνεται για την ανακάλυψη γνώσης ή διαδραστικής βελτιστοποίησης πολλαπλών στόχων, διαδικασία συνεχών δοκιμών και ενδεχόμενων αποτυχιών. Συχνά η διαδικασία απαιτεί να τροποποιηθεί η προ-επεξεργασία δεδομένων και τις παραμέτρους του μοντέλου μέχρι το αποτέλεσμα να επιτύχει τις επιθυμητές ιδιότητες. Ετυμολογικά, υπάρχουν και άλλες αναφορές στο clustering εκτός από τον όρο ομαδοποίηση – ανάλυση συστάδων, όπως «αυτόματη ταξινόμηση», «αριθμητική ταξινόμηση», «βοτρυολογία», «τυπολογική ανάλυση» και «ανίχνευση μιας κοινότητας». Οι λεπτές διαφορές των παραπάνω αφορούν τη χρήση των αποτελεσμάτων: ενώ στην εξόρυξη δεδομένων (data mining), στο κέντρο του ενδιαφέροντος βρίσκονται οι ομάδες που προκύπτουν, στην αυτόματη ταξινόμηση στο

κέντρο του ενδιαφέροντος βρίσκεται διακριτική δύναμη. Η πραγματικότητα δείχνει ότι υπάρχουν διαφορετικοί τύποι αλγορίθμων που κάνουν ομαδοποίηση με διαφορετικές μεθόδους: με βάση την πυκνότητα (density-based), με βάση τη διανομή (distribution-based), με βάση την κεντρική δόμησή τους (centroid-based) και με βάση την ιεραρχία (hierarchical-based). Στη συνέχεια παρατίθενται παραδείγματα τεχνικών clustering που απαντώνται συνήθως στην πρακτική.

6.1. K-Means

Ο αλγόριθμος ομαδοποίησης K-means (αλγόριθμός επίπεδης ομαδοποίησης ή πλησιέστερος κεντροειδικός ταξινομητής) υπολογίζει τις προβλέψεις με βάση την κεντρική δόμησή τους (centroids) και επαναλαμβάνει μέχρι να βρεθεί το βέλτιστο. Είναι πιθανώς γνωστό πόσες συστάδες υπάρχουν. Ο αριθμός των συστάδων που βρέθηκαν από δεδομένα με τη μέθοδο συμβολίζεται με το γράμμα «K» στο K-means. Πρόκειται για μέθοδο «κβαντοποίησης» διανυσμάτων, αρχικά από την επεξεργασία σήματος, που στοχεύει να χωρίσει τις παρατηρήσεις (N) σε συμπλέγματα (k) στα οποία κάθε παρατήρηση ανήκει στο σύμπλεγμα με τον πλησιέστερο μέσο όρο (κέντρα συστάδων), που έχει την ιδιότητα του πρωτότυπου στο σύμπλεγμα. Ο χώρος ομαδοποιείται σε δεδομένων σε κελιά - τμήματα ενός επιπέδου σε περιοχές κοντά σε καθεμία από ένα δεδομένο σύνολο αντικειμένων.

Εικόνα 13: Παράδειγμα εφαρμογής ομαδοποίησης K-Means



(Πηγή: Javapoint.com)

Η ομαδοποίηση k-means ελαχιστοποιεί τις διακυμάνσεις εντός του συμπλέγματος, αλλά όχι τις κανονικές αποστάσεις (Ευκλείδειες), το οποίο θα ήταν το πιο σύνθετο πρόβλημα τύπου Weber (απαίτηση εύρεσης σημείου που ελαχιστοποιεί το άθροισμα του κόστους μεταφοράς από αυτό το σημείο σε n σημεία προορισμού, όπου διαφορετικά σημεία προορισμού συνδέονται με διαφορετικό κόστος ανά μονάδα απόστασης). Αν και το πρόβλημα είναι υπολογιστικά δύσκολο, αλγόριθμοι εύρεσης που είναι αποτελεσματικοί συγκλίνουν γρήγορα σε μια βέλτιστη λύση. Τέτοιοι αλγόριθμοι είναι συνήθως παρόμοιοι με τον αλγόριθμο προσδοκίας-μεγιστοποίησης για μείγματα Gaussian κατανομών μέσω μιας προσέγγισης επαναληπτικής βελτίωσης η οποία χρησιμοποιείται τόσο από τη μοντελοποίηση k-means όσο και από τη μοντελοποίηση μειγμάτων Gauss. Αμφότεροι χρησιμοποιούν συστάδες για να μοντελοποιήσουν τα δεδομένα. Ωστόσο, η ομαδοποίηση k-means τείνει να βρίσκει συστάδες συγκρίσιμης χωρικής έκτασης, ενώ το μοντέλο του μείγματος Gauss δίνει τη δυνατότητα στις συστάδες να έχουν διαφορετικά σχήματα. Ο μη εποπτευόμενος αλγόριθμος k-means έχει μια χαλαρή σχέση με τον ταξινομητή k-πλησιέστερου γείτονα, μια δημοφιλή τεχνική εποπτευόμενης μηχανικής μάθησης για ταξινόμηση που συχνά συγχέεται με το k-means λόγω του ονόματος. Η εφαρμογή του ταξινομητή 1-πλησιέστερου γείτονα, στα κέντρα συμπλέγματος που λαμβάνονται από το k-means ταξινομεί νέα δεδομένα στις υπάρχουσες συστάδες.

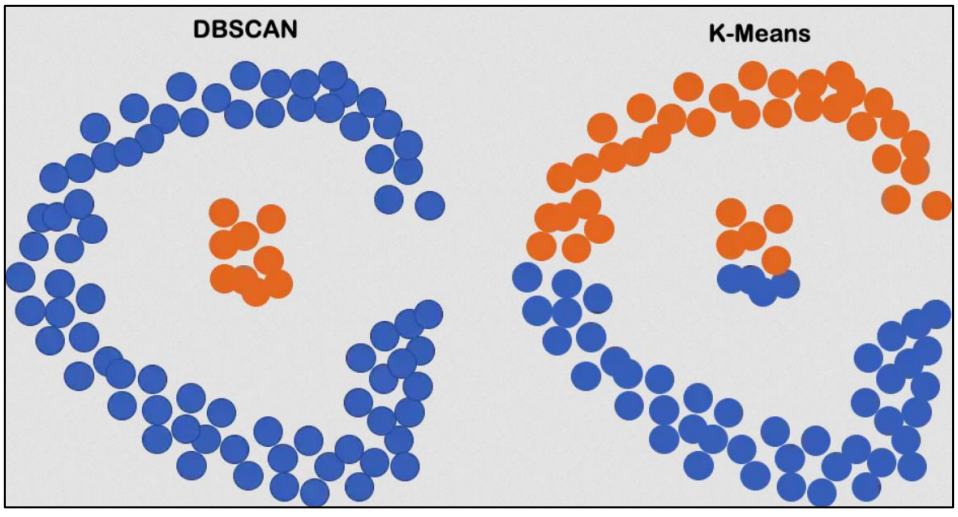
6.2. DBSCAN

Το DBSCAN (Density-based spatial clustering of applications with noise) είναι μια χωρική ομαδοποίηση εφαρμογών με θόρυβο με βάση την πυκνότητα. Είναι σε θέση να βρει συστάδες αυθαίρετου σχήματος και συστάδες με θόρυβο (δηλαδή ακραίες τιμές). Η κύρια ιδέα πίσω από το DBSCAN είναι ότι ένα σημείο ανήκει σε ένα σύμπλεγμα εάν είναι κοντά σε πολλά σημεία από αυτό το σύμπλεγμα. Είναι ένας μη παραμετρικός αλγόριθμος ομαδοποίησης που βασίζεται στην πυκνότητα. Ο αλγόριθμος ομαδοποιεί σημεία που είναι στενά συνδεδεμένα μεταξύ τους σε ένα σύνολο δεδομένων στο χώρο. Επίσης, επισημαίνει ως ακραία τα σημεία που βρίσκονται μόνα τους σε περιοχές με χαμηλή πυκνότητα. Ο DBSCAN είναι ένας συνηθισμένος στην πρακτική και τη βιβλιογραφία αλγόριθμος ομαδοποίησης. Αλγόριθμοι DBSCAN βρίσκονται εντός διαφόρων προγραμμάτων όπως το Μαθηματικό πρόγραμμα Apache Commons, το

προγραμμα εξόρυξης δεδομένων ELKI, το MATLAB, το πρόγραμμα μηχανικής μάθησης mlpack, κ.α.

Οι Shubert et al. (2017) αναφέρουν λόγους για τους οποίους ο αλγόριθμος αυτός είναι ακόμη χρήσιμος και αποτελεί καλύτερη εναλλακτική έναντι των υπολοίπων. Στα πλεονεκτήματά του συγκαταλέγονται η μη απαίτηση καθαρισμού συστάδων (σε αντίθεση με το K-MEANS), η δυνατότητα εύρεσης συμπλεγμάτων αυθαίρετου σχήματος, η ανθεκτικότητα στις ακραίες τιμές, η απαίτηση μονάχα δύο παραμέτρων και η μη ευαισθησία στη σειρά των σημείων στη βάση δεδομένων και η ευκολία χρήσης του για βάσεις δεδομένων που μπορούν να επιταχύνουν ερωτήματα περιοχής. Αντιθέτως, στα μειονεκτήματα συγκαταλέγονται το γεγονός ότι τα οριακά σημεία που είναι ταιριαστά από περισσότερα από μία συστάδες μπορούν να αποτελούν μέρος αυτών, ανάλογα με τη σειρά επεξεργασίας των δεδομένων. Επίσης, η ποιότητα του εξαρτάται από τη μέτρηση της απόστασης, συνήθως με την Ευκλείδεια απόσταση που όμως για δεδομένα υψηλών διαστάσεων, δεν είναι η βέλτιστη λύση. Ταυτόχρονα, το DBSCAN δεν μπορεί να ομαδοποιήσει καλά σύνολα δεδομένων με μεγάλες διαφορές στις πυκνότητες και σε περίπτωση που δεδομένα και κλίμακα δεν είναι κατανοητά, η επιλογή ενός σημαντικού ορίου απόστασης ε ενδεχομένως να είναι δύσκολη.

Εικόνα 14: Παράδειγμα ομαδοποίησης DBSCAN και σύγκριση με K-Means



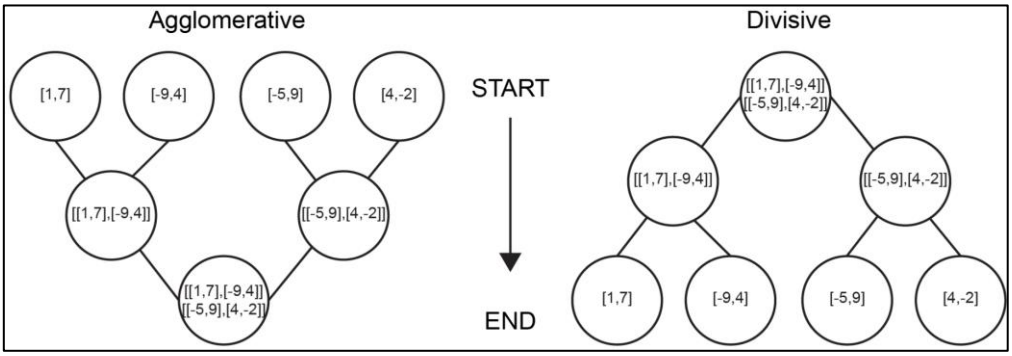
(Πηγή: towardsdatascience.com)

6.3. Agglomerative Clustering

Η συγκεντρωτική (ή αθροιστική) ομαδοποίηση (Agglomerative Clustering) είναι μια ομαδοποιητική διαδικασία από κάτω προς τα πάνω, στην οποία κάθε σημείο (αντικείμενο) δεδομένων είναι αρχικά ένα δικό του σύμπλεγμα και καθώς προχωράει προς τα πάνω στην ιεραρχία, συνδυάζονται περισσότερα ζεύγη συστάδων για να σχηματίσουν ένα ενιαίο σύμπλεγμα (κόμβος). Αυτή η διαδικασία επαναλαμβάνεται μέχρι όλα τα σημεία που βρίσκονται στο σύνολο να είναι μέλη ενός μόνο μεγάλου συμπλέγματος, το οποίο ονομάζεται ρίζα. Η αντίστροφη της συγκεντρωτικής είναι η διαιρετική ομαδοποίηση, (DIANA - Divise Analysis) και λειτουργεί από πάνω προς τα κάτω. Ξεκινάει με τη ρίζα, στην οποία όλα τα αντικείμενα περιλαμβάνονται σε ένα ενιαίο σύμπλεγμα. Σε κάθε βήμα της επανάληψης, το πιο ετερογενές σύμπλεγμα χωρίζεται σε δύο. Η διαδικασία επαναλαμβάνεται μέχρι όλα τα αντικείμενα να βρίσκονται στο δικό τους σύμπλεγμα. Η διαδικασία της αθροιστικής ομαδοποίησης εκτελείται με τέσσερα απλά βήματα και παρουσιάζεται στον παρακάτω πίνακα.

1. Η προετοιμασία των δεδομένων
2. Ο υπολογισμός πληροφοριών ομοιότητας και διαφοράς μεταξύ κάθε ζεύγους αντικειμένων στο σύνολο των δεδομένων.
3. Η χρήση της συνάρτησης σύνδεσης για την ομαδοποίηση αντικειμένων σε ιεραρχικό δέντρο συμπλέγματος, με βάση τις πληροφορίες απόστασης που δημιουργούνται στο βήμα 1. Με τη χρήση της συνάρτησης σύνδεσης, τα αντικείμενα/συστάδες που βρίσκονται σε κοντινή απόσταση συνδέονται.
4. Ο καθορισμός των σημείων που θα τμηματοποιήσει το ιεραρχικό δέντρο σε συστάδες.

Εικόνα 15: Παράδειγμα ομαδοποίησης Agglomerative και Divise



(Πηγή: Packt Subscription)

7. Πρακτικό Τμήμα Διπλωματικής

Σε αυτό το τμήμα της διπλωματικής παρουσιάζεται το πρακτικό τμήμα, η αναφορά της πρώτης και δεύτερης φάσης υλοποίησης και όλα τα δεδομένα που προέκυψαν από την υλοποίηση. Αναλυτικά ο κώδικας παρατίθεται στο παράρτημα της εργασίας.

7.1. Αναφορά πρώτης φάσης υλοποίησης

7.1.1. Γλώσσα και περιβάλλον υλοποίησης

Για την υλοποίηση του πρακτικού μέρους της διπλωματικής εργασίας χρησιμοποιήθηκε η γλώσσα Python και το περιβάλλον υλοποίησης είναι το Spyder IDE. Σημαντικές βιβλιοθήκες που μας έχουν βοηθήσει μέχρι στιγμής στην υλοποίηση της πειραματικής διαδικασίας:

- csv
- pandas
- numpy
- matplotlib
- sklearn

7.1.2. DATASET

Το dataset το οποίο έχει χρησιμοποιηθεί μέχρι στιγμής ελήφθη από τον παρακάτω σύνδεσμο:
https://data.hellenicdataservice.gr/dataset/d3b0d446-aaba-49a8-acce-e7c6f6f5d3b5
Το dataset αυτό περιέχει .csv αρχεία τα οποία προέρχονται από διάφορους μετεωρολογικούς σταθμούς της χώρας και καλύπτουν σε χρονικό εύρος τα έτη 2010 έως 2019. Κάθε csv αρχείο περιέχει τις παρακάτω στήλες:

- date
- mean_temp
- max_temp
- min_temp
- mean_hum
- max_hum

- min_hum
- mean_pres
- max_pres
- min_pres
- rain
- mean_speed_wind
- dir_wind
- max_gust_wind

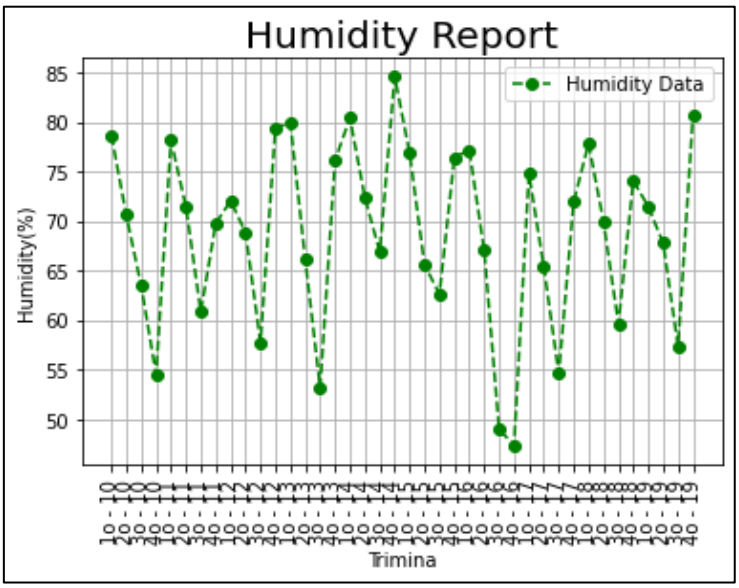
Οι στήλες τις οποίες αποφασίσαμε να χρησιμοποιήσουμε για την πραγματοποίηση της πειραματικής διαδικασίας είναι οι εξής:

- mean_temp
- mean_pres
- mean_hum
- rain
- mean_speed_wind
- dir_wind

7.1.3. Οπτικοποίηση Δεδομένων

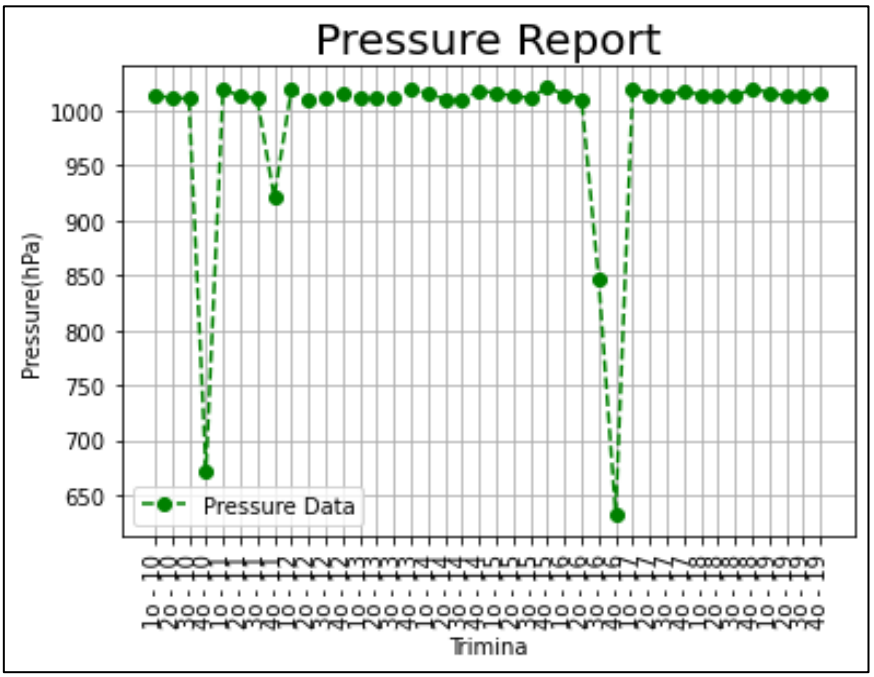
Για το humidity (παρουσιάζουμε τα συγκεντρωτικά δεδομένα ανά τρίμηνο):

Εικόνα 16: Humidity Report



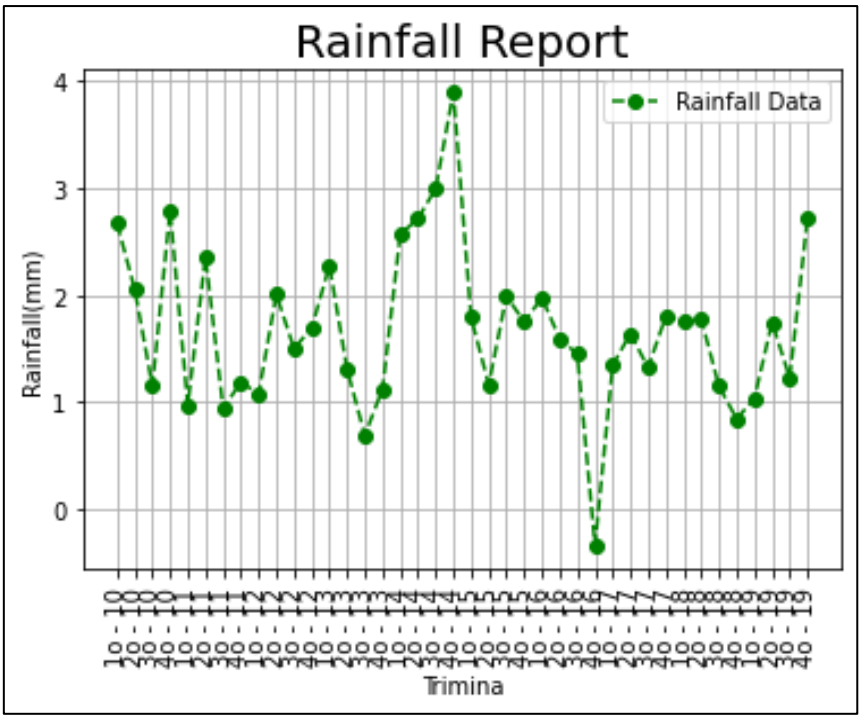
Για το pressure (παρουσιάζουμε τα συγκεντρωτικά δεδομένα ανά τρίμηνο):

Εικόνα 17: Pressure Report



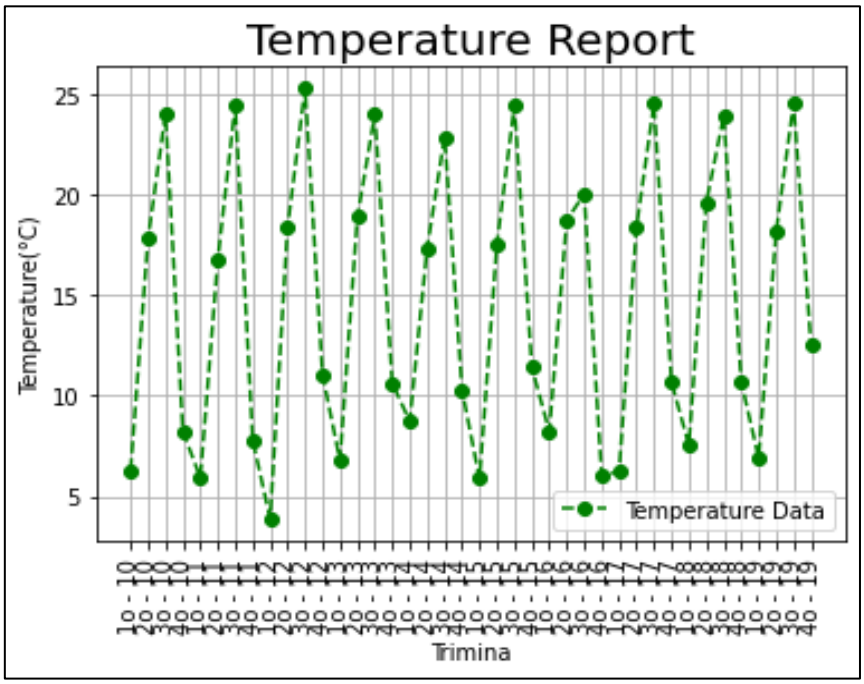
Για το rainfall (παρουσιάζουμε τα συγκεντρωτικά δεδομένα ανά τρίμηνο):

Εικόνα 18:Rainfall Report



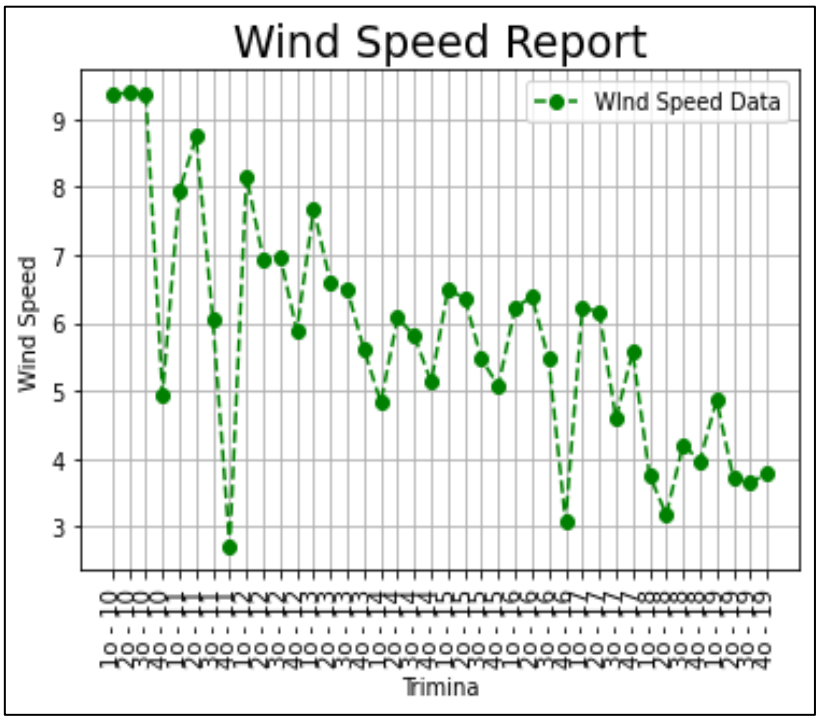
Για το temperature (παρουσιάζουμε τα συγκεντρωτικά δεδομένα ανά τρίμηνο):

Εικόνα 19:Temperature Report



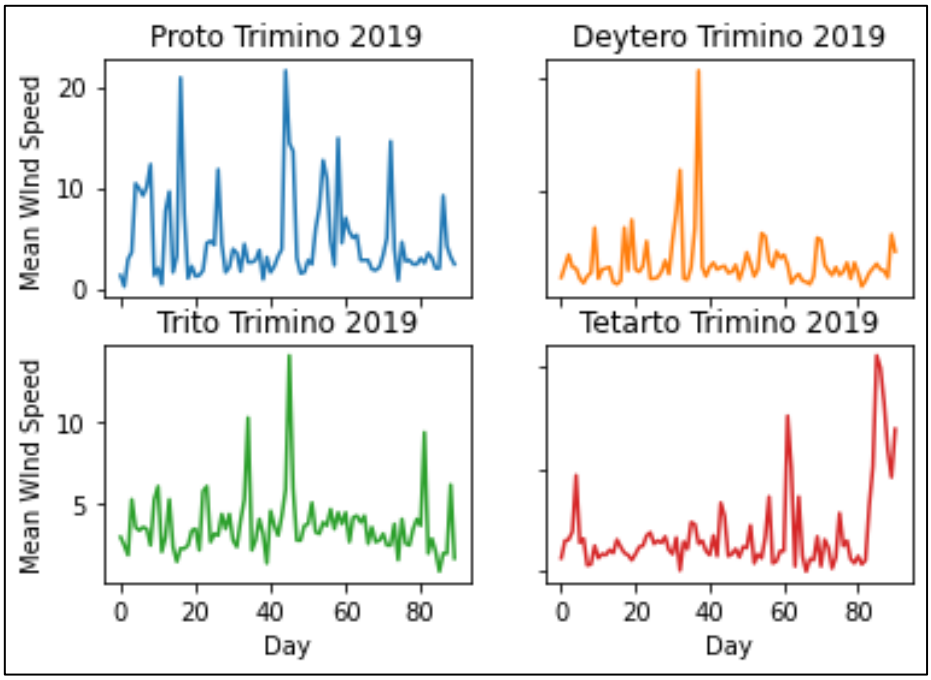
Για το wind speed (παρουσιάζουμε τα συγκεντρωτικά δεδομένα ανά τρίμηνο):

Εικόνα 20:Wind Speed Report



Επίσης για όλα τα παραπάνω χαρακτηριστικά, έχουμε παράξει για όλες τις χρονιές γραφήματα όπως το παρακάτω τα οποία μας βοηθούν στην αναλυτικότερη οπτικοποίηση των δεδομένων:

Εικόνα 21:Χρονικά γραφήματα



7.1.4. Ελλιπείς τιμές

Από την παραπάνω πειραματική διαδικασία διαπιστώσαμε ότι σε μερικές περιπτώσεις αντιμετωπίζουμε το πρόβλημα έλλειψης τιμών σε κάποιες στήλες κάποιων εγγραφών. Οι ελλιπείς τιμές δημιουργούν πρόβλημα στη λειτουργία των αλγορίθμων και θα έπρεπε να βρεθεί μια λύση για την ορθότερη λειτουργία των αλγορίθμων και άρα της πρόβλεψης. Για να δοθεί μια λύση στο παραπάνω πρόβλημα αποφασίσαμε να εφαρμόσουμε την παρακάτω μέθοδο απόδοσης του μέσου όρου στις ελλείπεις τιμές. Εναλλακτικά, εάν είναι επιθυμητό, μπορούν να χρησιμοποιηθούν και εναλλακτικές μέθοδοι για την συμπλήρωση των τιμών που λείπουν (πχ πρόβλεψη με χρήση νευρωνικού δικτύου).

7.1.5. Μοντέλο που χρησιμοποιήθηκε

Τελικά, αποφασίσαμε να χρησιμοποιήσουμε ένα **RNN** δίκτυο ως βασικό μοντέλο για πρόβλεψη των τιμών των διαφόρων χαρακτηριστικών (εγκαταλείφθηκε η ιδέα του CNN δικτύου, καθώς διαπιστώσαμε ότι δεν ανταποκρίνεται στις ανάγκες μας).

Όσο αφορά την πειραματική διαδικασία, στόχος μας είναι να μπορούμε να προβλέψουμε κάθε ένα από τα παρακάτω χαρακτηριστικά, με χρήση του RNN δικτύου:

- rain
- humidity
- temperature
- pressure
- wind_speed
- wind_direction

Αρχικά, έγινε αξιολόγηση της πειραματικής διαδικασίας, χρησιμοποιώντας το LOSS. Στους πίνακες που παρουσιάζονται στην συνέχεια, δείχνουμε για την πρόβλεψη του κάθε χαρακτηριστικού, ποια είναι τα χαρακτηριστικά τα οποία λάβαμε υπόψη και ποια είναι η τιμή του loss που πέτυχαμε. Στη συνέχεια παρουσιάζονται τα αποτελέσματα της πειραματικής διαδικασίας στην πρώτη φάση υλοποίησης, για όλες τις τιμές των παραμέτρων που παρουσιαστηκαν.

7.1.6. Αποτελέσματα πειραματικής διαδικασίας

Πίνακας 1: Πρόβλεψη Βροχής

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_DIR_WIND FINAL_MEAN_HUM	22.75
FINAL_DIR_WIND FINAL_MEAN_TEMP	26.4
FINAL_DIR_WIND FINAL_MEAN_PRES	26.83
FINAL_DIR_WIND FINAL_MEAN_SPEED	26.78
FINAL_MEAN_HUM FINAL_MEAN_TEMP	24.23
FINAL_MEAN_HUM FINAL_MEAN_SPEED	23.71
FINAL_MEAN_HUM FINAL_MEAN_PRESS	26.83
FINAL_MEAN_PRESS FINAL_MEAN_SPEED	26.84
FINAL_MEAN_PRESS FINAL_MEAN_TEMP	26.61
FINAL_MEAN_TEMP FINAL_MEAN_SPEED	26.69
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_PRES	25.71
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_SPEED	23.68
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP	21.96
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP FINAL_MEAN_SPEED	25.02
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP FINAL_MEAN_PRES	26.69

Πίνακας 2: Πρόβλεψη υγρασίας

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_RAIN FINAL_MEAN_SPEED	813.12
FINAL_RAIN FINAL_MEAN_PRES	2254.305
FINAL_RAIN FINAL_DIR_WIND	1001.706
FINAL_RAIN FINAL_MEAN_TEMP	884.58
FINAL_MEAN_TEMP FINAL_MEAN_SPEED	1014.32
FINAL_MEAN_TEMP FINAL_MEAN_PRES	2284.36
FINAL_MEAN_TEMP FINAL_DIR_WIND	1002.81
FINAL_MEAN_PRES FINAL_DIR_WIND	883.29
FINAL_MEAN_PRES FINAL_MEAN_SPEED	941.43
FINAL_MEAN_SPEED FINAL_DIR_WIND	834.05
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_PRES	2628.65
FINAL_RAIN FINAL_MEAN_SPEED FINAL_DIR_WIND	814
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP	669
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP FINAL_DIR_WIND	909.15
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP FINAL_MEAN_PRES	2558.303

Πίνακας 3: Πρόβλεψη πίεσης

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_MEAN_TEMP, FINAL_DIR_WIND	937351.625
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	941353.875
FINAL_MEAN_TEMP, FINAL_RAIN	936037.875
FINAL_MEAN_TEMP, FINAL_MEAN_SPEED	940016.0
FINAL_DIR_WIND, FINAL_MEAN_HUM	945729.81
FINAL_DIR_WIND, FINAL_RAIN	939404.3125
FINAL_DIR_WIND, FINAL_MEAN_SPEED	938650.3125
FINAL_MEAN_HUM, FINAL_RAIN	939900.625
FINAL_MEAN_HUM, FINAL_MEAN_SPEED	938955.0
FINAL_RAIN, FINAL_MEAN_SPEED	942287.1875

Πίνακας 4: Πρόβλεψη ταχύτητας ανέμου

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_RAIN, FINAL_MEAN_TEMP	16.66
FINAL_RAIN, FINAL_MEAN_PRES	16.98
FINAL_RAIN, FINAL_DIR_WIND	15.08
FINAL_RAIN, FINAL_MEAN_HUM	16.47
FINAL_MEAN_TEMP, FINAL_MEAN_PRES	16.98
FINAL_MEAN_TEMP, FINAL_DIR_WIND	14.81
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	15.45
FINAL_MEAN_PRES, FINAL_DIR_WIND	15.06
FINAL_MEAN_PRES, FINAL_MEAN_HUM	16.43
FINAL_DIR_WIND, FINAL_MEAN_HUM	14.92
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM	14.01
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_PRES	14.96
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_RAIN	14.67
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM, FINAL_RAIN	14.48
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM, FINAL_MEAN_PRESS	16.98

Πίνακας 5: Πρόβλεψη κατεύθυνσης ανέμου

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	26716.18359375
FINAL_MEAN_TEMP, FINAL_MEAN_PRES	30722.734375
FINAL_MEAN_TEMP, FINAL_RAIN	27942.083984375
FINAL_MEAN_TEMP, FINAL_MEAN_SPEED	26950.923
FINAL_MEAN_HUM, FINAL_MEAN_PRES	30936.806
FINAL_MEAN_HUM, FINAL_RAIN	27171.921875
FINAL_MEAN_HUM, FINAL_MEAN_SPEED	27601.537
FINAL_MEAN_PRES, FINAL_RAIN	27452.46
FINAL_MEAN_PRES, FINAL_MEAN_SPEED	26785.98
FINAL_RAIN, FINAL_MEAN_SPEED	26573.17

Πίνακας 6:Πρόβλεψη θερμοκρασίας

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_DIR_WIND, FINAL_MEAN_HUM	46.76
FINAL_DIR_WIND, FINAL_RAIN	63.56
FINAL_DIR_WIND, FINAL_MEAN_PRES	63.84
FINAL_DIR_WIND, FINAL_MEAN_SPEED	61
FINAL_MEAN_PRES, FINAL_MEAN_HUM	47.7
FINAL_MEAN_PRES, FINAL_RAIN	63.29
FINAL_MEAN_PRES, FINAL_MEAN_SPEED	60.55
FINAL_MEAN_SPEED, FINAL_RAIN	63.57
FINAL_MEAN_SPEED, FINAL_MEAN_HUM	41.52
FINAL_RAIN, FINAL_MEAN_HUM	44.61
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN	40.06
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_MEAN_PRESS	62.41
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_DIR_WIND	48.1
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN, FINAL_DIR_WIND	58.13
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN, FINAL_MEAN_PRES	63.82

7.2. Αναφορά δεύτερης φάσης υλοποίησης

7.2.1. Διαδικασία Clustering

Στο συγκεκριμένο μέρος της υλοποίησης πραγματοποιήσαμε clustering των δεδομένων, χρησιμοποιώντας τους 2 παρακάτω αλγορίθμους συσταδοποίησης: K-Means και Agglomerative. Στο συγκεκριμένο σημείο, θα πρέπει να αναφερθεί ότι εξετάστηκαν και άλλοι αλγόριθμοι συσταδοποίησης, οι οποίοι όμως δεν παρουσίασαν αξιόλογη συμπεριφορά. Έχοντας πραγματοποιήσει clustering με κάθε έναν από τους 2 παραπάνω αλγορίθμους, αποφασίσαμε να χρησιμοποιήσουμε ως τεχνική πρόβλεψης τιμής ενός χαρακτηριστικού την απόδοση του μέσου όρου του χαρακτηριστικού της συστάδας στην οποία ανήκει η εκάστοτε εγγραφή. Επίσης να αναφερθεί ότι για την αξιολόγηση των αλγορίθμων συσταδοποίησης χρησιμοποιήθηκε η μετρική silhouette. Παραθέτουμε τα πειραματικά αποτελέσματα τα οποία προέκυψαν (χρησιμοποιήθηκαν 2 clusters):

Πίνακας 7: Αλγόριθμος K-Means

Χαρακτηριστικό	**Cluster 0**	**Cluster 1**
Average temperature	15.774	14.39
Average humidity	71.96	68.83
Average pressure	1014.75	1013.96
Average speed wind	4.88	7.28
Average rain	2.02	1.46

Πίνακας 8: Αλγόριθμος Agglomerative

Χαρακτηριστικό	**Cluster 1**	**Cluster 0**
Average temperature	15.774	14.387
Average humidity	71.96	68.826
Average pressure	1014.748	1013.957
Average speed wind	4.88	7.277
Average rain	2.016	1.458

7.2.2. Χρήση Κλασσικών Μοντέλων Clasification

Στο συγκεκριμένο σημείο της υλοποίησης, προσπαθήσαμε να χρησιμοποιήσουμε κλασσικά μοντέλα πραγματοποίησης classification για την πρόβλεψη των χαρακτηριστικών. Οι μέθοδοι που χρησιμοποιήθηκαν είναι οι

- Random Forest,
- KNN,
- Logistic Regression,
- Decision Tree,
- SVM
- Naive Bayes.

Οι μέθοδοι αυτοί αξιολογήθηκαν ως προς την μετρική του precision. Στην συνέχεια παραθέτουμε τα πειραματικά αποτελέσματα για την πρόβλεψη του κάθε χαρακτηριστικού:

Πίνακας 9: Χρήση Random Forest

Χαρακτηριστικό	**Precision**
rain	0.722
temperature	0.04
speed wind	0.05
pressure	0.04
wind direction	0.334
humidity	0.03

Πίνακας 10: Χρήση KNN

Χαρακτηριστικό	**Precision**		
	K=2	**K=3**	**K=4**
rain	0.67	0.7	0.7
temperature	0.02	0.03	0.027
speed wind	0.07	0.05	0.056
pressure	0.03	0.02	0.034
wind direction	0.28	0.25	0.261
humidity	0.03	0.03	0.031

Πίνακας 11:Χρήση Logistic Regression

Χαρακτηριστικό	Precision
rain	0.7
temperature	0.03
speed wind	0.04
pressure	0.03
wind direction	0.28
humidity	0.03

Πίνακας 12: Χρήση Decision Tree

Χαρακτηριστικό	Precision
rain	0.6
temperature	0.02
speed wind	0.06
pressure	0.03
wind direction	0.25
humidity	0.04

Πίνακας 13:Χρήση SVM

Χαρακτηριστικό	Precision
rain	0.71
temperature	0.04
speed wind	0.04
pressure	0.03
wind direction	0.33
humidity	0.03

Πίνακας 14:Χρήση Naïve Bayes

Χαρακτηριστικό	**Precision**
rain	0.69
temperature	0.03
speed wind	0.06
pressure	0.02
wind direction	0.35
humidity	0.03

Παρατηρούμε ότι σε όλες τις περιπτώσεις μεθόδων που χρησιμοποιήθηκαν, ικανοποιητικά αποτελέσματα λάβαμε στην περίπτωση πρόβλεψης του rain (της τάξης του 0.7), ενώ στις υπόλοιπες περιπτώσεις χαρακτηριστικών τα αποτελέσματα δεν ήταν ικανοποιητικά. Επομένως οι συγκεκριμένες μέθοδοι classification θα μπορούσαν να χρησιμοποιηθούν για την πρόβλεψη της βροχόπτωσης.

7.2.3. Πρόβλεψη με LSTM

Στο τρίτο μέρος της δεύτερης φάσης της υλοποίησης, υλοποιήθηκε πρόβλεψη τιμών χρησιμοποιώντας LSTM νευρωνικό δίκτυο. Η πειραματική διαδικασία που ακολουθήθηκε είναι παρόμοια με αυτή του RNN, έτσι ώστε να μπορέσουν να γίνουν και οι απαραίτητες συγκρίσεις. Στην συνέχεια, παραθέτουμε τα αποτελέσματα της πειραματικής διαδικασίας:

Πίνακας 15: Πρόβλεψη βροχής

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_DIR_WIND FINAL_MEAN_HUM	23.05
FINAL_DIR_WIND FINAL_MEAN_TEMP	26.53
FINAL_DIR_WIND FINAL_MEAN_PRES	26.9
FINAL_DIR_WIND FINAL_MEAN_SPEED	26.91
FINAL_MEAN_HUM FINAL_MEAN_TEMP	22.106
FINAL_MEAN_HUM FINAL_MEAN_SPEED	22.824
FINAL_MEAN_HUM FINAL_MEAN_PRESS	26.904
FINAL_MEAN_PRESS FINAL_MEAN_SPEED	26.86
FINAL_MEAN_PRESS FINAL_MEAN_TEMP	26.615
FINAL_MEAN_TEMP FINAL_MEAN_SPEED	26.477
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_PRES	26.93
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_SPEED	22.103
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP	21.718
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP FINAL_MEAN_SPEED	21.192
FINAL_DIR_WIND FINAL_MEAN_HUM FINAL_MEAN_TEMP FINAL_MEAN_PRES	23.117

Πίνακας 16:Πρόβλεψη υγρασίας

ΣΥΝΔΥΑΣΜΟΙ	**LOSS**
FINAL_RAIN FINAL_MEAN_SPEED	149.714
FINAL_RAIN FINAL_MEAN_PRES	190.266
FINAL_RAIN FINAL_DIR_WIND	138.807
FINAL_RAIN FINAL_MEAN_TEMP	93.309
FINAL_MEAN_TEMP FINAL_MEAN_SPEED	112.057
FINAL_MEAN_TEMP FINAL_MEAN_PRES	190.324
FINAL_MEAN_TEMP FINAL_DIR_WIND	190.504
FINAL_MEAN_PRES FINAL_DIR_WIND	190.616
FINAL_MEAN_PRES FINAL_MEAN_SPEED	180.993
FINAL_MEAN_SPEED FINAL_DIR_WIND FINAL_RAIN	190.96
FINAL_MEAN_SPEED FINAL_MEAN_PRES	143.067
FINAL_RAIN FINAL_MEAN_SPEED FINAL_DIR_WIND	190.45
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP	116.737
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP FINAL_DIR_WIND	87.632
FINAL_RAIN FINAL_MEAN_SPEED FINAL_MEAN_TEMP FINAL_MEAN_PRES	125.938

Πίνακας 17:Πρόβλεψη ατμοσφαιρικής πίεσης

ΣΥΝΔΥΑΣΜΟΙ	**LOSS**
FINAL_MEAN_TEMP, FINAL_DIR_WIND	47.421
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	51.224
FINAL_MEAN_TEMP, FINAL_RAIN	47.361
FINAL_MEAN_TEMP, FINAL_MEAN_SPEED	47.256
FINAL_DIR_WIND, FINAL_MEAN_HUM	177.953
FINAL_DIR_WIND, FINAL_RAIN	47.38
FINAL_DIR_WIND, FINAL_MEAN_SPEED	46.98
FINAL_MEAN_HUM, FINAL_RAIN	47.27
FINAL_MEAN_HUM, FINAL_MEAN_SPEED	47.456
FINAL_RAIN, FINAL_MEAN_SPEED	47.465

Πίνακας 18:Πρόβλεψη ταχύτητας ανέμου

ΣΥΝΔΥΑΣΜΟΙ	**LOSS**
FINAL_RAIN, FINAL_MEAN_TEMP	16.669
FINAL_RAIN, FINAL_MEAN_PRES	17.058
FINAL_RAIN, FINAL_DIR_WIND	15.109
FINAL_RAIN, FINAL_MEAN_HUM	16.427
FINAL_MEAN_TEMP, FINAL_MEAN_PRES	16.883
FINAL_MEAN_TEMP, FINAL_DIR_WIND	14.804
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	15.66
FINAL_MEAN_PRES, FINAL_DIR_WIND	15.287
FINAL_MEAN_PRES, FINAL_MEAN_HUM	16.614
FINAL_DIR_WIND, FINAL_MEAN_HUM	14.782
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM	13.847
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_PRES	14.911
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_RAIN	14.505
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM, FINAL_RAIN	13.704
FINAL_MEAN_TEMP, FINAL_DIR_WIND, FINAL_MEAN_HUM, FINAL_MEAN_PRESS	14.412

Πίνακας 19:Πρόβλεψη κατεύθυνσης ανέμου

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_MEAN_TEMP, FINAL_MEAN_HUM	13335.487
FINAL_MEAN_TEMP, FINAL_MEAN_PRES	13338.082
FINAL_MEAN_TEMP, FINAL_RAIN	13033.622
FINAL_MEAN_TEMP, FINAL_MEAN_SPEED	11509.185
FINAL_MEAN_HUM, FINAL_MEAN_PRES	13387.442
FINAL_MEAN_HUM, FINAL_RAIN	13324.188
FINAL_MEAN_HUM, FINAL_MEAN_SPEED	11908.295
FINAL_MEAN_PRES, FINAL_RAIN	13338.082
FINAL_MEAN_PRES, FINAL_MEAN_SPEED	11998.876
FINAL_RAIN, FINAL_MEAN_SPEED	11874.652

Πίνακας 20:Πρόβλεψη θερμοκρασίας

ΣΥΝΔΥΑΣΜΟΙ	LOSS
FINAL_DIR_WIND, FINAL_MEAN_HUM	42.8
FINAL_DIR_WIND, FINAL_RAIN	61.98
FINAL_DIR_WIND, FINAL_MEAN_PRES	63.99
FINAL_DIR_WIND, FINAL_MEAN_SPEED	58.39
FINAL_MEAN_PRES, FINAL_MEAN_HUM	44.96
FINAL_MEAN_PRES, FINAL_RAIN	63.062
FINAL_MEAN_PRES, FINAL_MEAN_SPEED	59.79
FINAL_MEAN_SPEED, FINAL_RAIN	59.65
FINAL_MEAN_SPEED, FINAL_MEAN_HUM	40.71
FINAL_RAIN, FINAL_MEAN_HUM	43.26
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN	40.22
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_MEAN_PRESS	63.97
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_DIR_WIND	41.23
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN, FINAL_DIR_WIND	41.78
FINAL_MEAN_SPEED, FINAL_MEAN_HUM, FINAL_RAIN, FINAL_MEAN_PRES	40.38

8. Συμπεράσματα & Προτάσεις

Κατά την πρώτη φάση υλοποίησης παρατηρήθηκε ότι στις περιπτώσεις πρόβλεψης της θερμοκρασίας, της ταχύτητας του ανέμου και της βροχόπτωσης, επιτεύχθηκαν ικανοποιητικές τιμές όσο αφορά το loss. Αντιθέτως, στις υπόλοιπες περιπτώσεις τα αποτελέσματα δεν ήταν καθόλου ικανοποιητικά. Επίσης, για την πρόβλεψη της θερμοκρασίας ο συνδυασμός που πέτυχε το χαμηλότερο loss ήταν ο:

- mean_speed_wind
- mean_humidity
- rain

Για την πρόβλεψη του rain, ο συνδυασμός που πέτυχε το χαμηλότερο loss ήταν ο:

- dir_wind
- mean_humidity
- mean_temperature

Για την πρόβλεψη του wind_speed, ο συνδυασμός με τα καλύτερα αποτελέσματα ήταν ο:

- mean_temp
- mean_humidity
- dir_wind

Για τη δεύτερη φάση, παρατηρείται ότι στην περίπτωση χρήσης του LSTM λάβαμε πολύ καλύτερα αποτελέσματα όσο αφορά το loss σε σχέση με το RNN. Επομένως καταλήγουμε στο συμπέρασμα ότι προτιμάται η χρήση του LSTM δικτύου για την πρόβλεψη των καιρικών φαινομένων.

Στα πλαίσια της συγκεκριμένης διπλωματικής εργασίας, χρησιμοποιήθηκε πληθώρα τεχνικών, προκειμένου να γίνει πρόβλεψη των τιμών διάφορων μετεωρολογικών δεδομένων. Πιο συγκεκριμένα:

- Εφαρμόσαμε RNN δίκτυο, στην περίπτωση του οποίου διαπιστώσαμε ικανοποιητικά αποτελέσματα όσο αφορά το loss για την πρόβλεψη

συγκεκριμένων χαρακτηριστικών, λαμβάνοντας υπόψη συγκεκριμένους συνδυασμούς χαρακτηριστικών.

- Εφαρμόσαμε LSTM δίκτυο, όπου και λάβαμε καλύτερα αποτελέσματα σε σχέση με την περίπτωση του RNN.
- Εφαρμόσαμε αλγορίθμους συσταδοποίησης για πρόβλεψη των τιμών των χαρακτηριστικών με την μέθοδο απόδοσης της μέσης τιμής της συστάδας.
- Τέλος, εφαρμόσαμε κλασσικά μοντέλα κατηγοριοποίησης, όπου και διαπιστώσαμε ότι μπορούν να λειτουργήσουν αποτελεσματικά για την περίπτωση πρόβλεψης της βροχόπτωσης.

Σαν μελλοντικές κατευθύνσεις προτείνουμε:

- Χρήση επιπλέον αλγορίθμων και μοντέλων, έτσι ώστε να προκύψουν περισσότερα συγκριτικά αποτελέσματα
- Χρήση ογκωδέστερου dataset όσο αφορά και το πλήθος των χαρακτηριστικών αλλά και το πλήθος των εγγραφών
- Χρήση περισσότερων μετρικών αξιολόγησης

Ελληνική Βιβλιογραφία

Καλτσάς, Δ. (2015). *Εξόρυξη Δεδομένων σε δεδομένα διαδικτυακής κίνησης και ροής χτυπημάτων* (Master's thesis, Πανεπιστήμιο Πειραιώς).

Κογκόλη, Μ. (2018) Μοντέλα Πρόγνωσης του Καιρού και Επαλήθευση τους, με Αναφορά στα Βασικά υπό Παρατήρηση Φαινόμενα και Μεγέθη. Διπλωματική Εργασία, Πανεπιστήμιο Θεσσαλίας, Σχολή Θετικών Επιστημών.

Ξενόγλωσση Βιβλιογραφία

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, *4*(11), e00938.

Alpaydin, E. (1997). Voting over multiple condensed nearest neighbors. In *Lazy learning* (pp. 115-132). Springer, Dordrecht.

Alpaydin, E. (2020). *Introduction to machine learning*. MIT press.

Bengtsson L. (1991). Advances and prospects in numerical weather prediction. *Quarterly Journal of the Royal Meteorological Society*, Vol. 117, pp. 855–902.

Bijalwan, V., Kumar, V., Kumari, P., & Pascual, J. (2014). KNN based machine learning approach for text and document mining. *International Journal of Database Theory and Application*, *7*(1), 61-70.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179-211.

Eskow, D. (1983). "Make Your Own Weather Forecasts". *Popular Mechanics*. Vol. 159, no. 3. p. 148.

Gleason, K.L. (2008). "Ozonesonde" National Oceanic and Atmospheric Administration.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2008). A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5), 855-868.

Graves, A., Mohamed, A. R., & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645-6649). IEEE.

Hart, P. (1968). The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, *14*(3), 515-516.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8), 1735-1780.

Hubel, D. H., & Wiesel, T. N. (1970). The period of susceptibility to the physiological effects of unilateral eye closure in kittens. *The Journal of physiology*, 206(2), 419-436.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. Journal of artificial intelligence research, 4, 237-285.

Sak, H., Senior, A. W., & Beaufays, F. (2014). *Long short-term memory recurrent neural network architectures for large scale acoustic modeling.*

Shao, Y. H., Chen, W. J., & Deng, N. Y. (2014). Nonparallel hyperplane support vector machine for binary classification problems. *Information Sciences*, *263*, 22-35.

Schulze, G.C. (2007) Atmospheric observations and numerical weather prediction : SAEON review. *South African Journal of Science*, Volume 103, Issue 7 https://journals.co.za/doi/epdf/10.10520/EJC96696

Meersman, R. (2003). *On the Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003: Proceedings* (Vol. 1). Springer Science & Business Media.

Mendel, J. M., & McLaren, R. W. (1970). 8 reinforcement-learning control and pattern recognition systems. *Mathematics in Science and Engineering* , 66, 287-318.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

National Weather Service, (n.d.) Using and Understanding Doppler Radar https://www.weather.gov/mkx/using-radar

Wilson, J. (n.d.) "Skywatch: Signs of the Weather". https://archive.ph/20130106041039/http://wilstar.com/skywatch.htm

Vapnik, V. N., & Chervonenkis, A. Y. (1982). Necessary and sufficient conditions for the uniform convergence of means to their expectations. *Theory of Probability & Its Applications*, *26*(3), 532-553.

Zaki, M. J., & Meira Jr, W. (2020). *Data mining and machine learning: Fundamental concepts and algorithms*. Cambridge University Press.

Παράρτημα Κώδικα

Κώδικας test

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
data = pd.read_csv('test.csv')
date = data['date']
mean_temp = data['mean_temp']
date = date.to_numpy()
mean_temp = mean_temp.to_numpy()
mytemps = []
mysum = []
mycount = []
trimina = ['1o - 10','2o - 10', '3o - 10', '4o - 10',

        '1o - 11','2o - 11', '3o - 11', '4o - 11',

        '1o - 12','2o - 12', '3o - 12', '4o - 12',

        '1o - 13','2o - 13', '3o - 13', '4o - 13',

        '1o - 14','2o - 14', '3o - 14', '4o - 14',

        '1o - 15','2o - 15', '3o - 15', '4o - 15',

        '1o - 16','2o - 16', '3o - 16', '4o - 16',

        '1o - 17','2o - 17', '3o - 17', '4o - 17',
```

```
        '1o - 18','2o - 18', '3o - 18', '4o - 18',

        '1o - 19','2o - 19', '3o - 19', '4o - 19']
for i in range(len(mean_temp)):
    if mean_temp[i]=='---':
        mean_temp[i]='-1'
for i in range(40):
    mysum.append(0)
    mycount.append(0)
for i in range(len(mean_temp)):
        myyear = str(date[i][0:4])
        if(myyear=='2010'):
            if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                    mysum[0] = mysum[0] + float(mean_temp[i])
                    mycount[0] = mycount[0]+1
        elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
            mysum[1] = mysum[1] + float(mean_temp[i])
                    mycount[1] = mycount[1]+1

        elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
            mysum[2] = mysum[2] + float(mean_temp[i])
                    mycount[2] = mycount[2]+1
                else:
                    mysum[3] = mysum[3] + float(mean_temp[i])
                    mycount[3] = mycount[3]+1
                elif(myyear=='2011'):
            if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                    mysum[4] = mysum[4] + float(mean_temp[i])
                    mycount[4] = mycount[4]+1
            elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                    mysum[5] = mysum[5] + float(mean_temp[i])
                    mycount[5] = mycount[5]+1
```

```
elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                mysum[6] = mysum[6] + float(mean_temp[i])
                mycount[6] = mycount[6]+1
                else:
                mysum[7] = mysum[7] + float(mean_temp[i])
                mycount[7] = mycount[7]+1
elif(myyear=='2012'):
    if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                mysum[8] = mysum[8] + float(mean_temp[i])
                mycount[8] = mycount[8]+1
    elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                mysum[9] = mysum[9] + float(mean_temp[i])
                mycount[9] = mycount[9]+1
        elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                mysum[10] = mysum[10] + float(mean_temp[i])
                mycount[10] = mycount[10]+1
else:
                mysum[11] = mysum[11] + float(mean_temp[i])
                mycount[11] = mycount[11]+1
        elif(myyear=='2013'):
    if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                mysum[12] = mysum[12] + float(mean_temp[i])
                mycount[12] = mycount[12]+1
elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                mysum[13] = mysum[13] + float(mean_temp[i])
                mycount[13] = mycount[13]+1
        elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                mysum[14] = mysum[14] + float(mean_temp[i])
                mycount[14] = mycount[14]+1
```

```
    else:
                mysum[15] = mysum[15] + float(mean_temp[i])
                mycount[15] = mycount[15]+1
    elif(myyear=='2014'):
       if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                mysum[16] = mysum[16] + float(mean_temp[i])
                mycount[16] = mycount[16]+1
          elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                mysum[17] = mysum[17] + float(mean_temp[i])
                mycount[17] = mycount[17]+1
          elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                mysum[18] = mysum[18] + float(mean_temp[i])
                mycount[18] = mycount[18]+1
               else:
                mysum[19] = mysum[19] + float(mean_temp[i])
                mycount[19] = mycount[19]+1
elif(myyear=='2015'):
       if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                mysum[20] = mysum[20] + float(mean_temp[i])
                mycount[20] = mycount[20]+1
    elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                mysum[21] = mysum[21] + float(mean_temp[i])
                mycount[21] = mycount[21]+1
          elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                mysum[22] = mysum[22] + float(mean_temp[i])
                mycount[22] = mycount[22]+1
                else:
                mysum[23] = mysum[23] + float(mean_temp[i])
                mycount[23] = mycount[23]+1
    elif(myyear=='2016'):
       if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                mysum[24] = mysum[24] + float(mean_temp[i])
```

```
            mycount[24] = mycount[24]+1
      elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
            mysum[25] = mysum[25] + float(mean_temp[i])
            mycount[25] = mycount[25]+1
       elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
            mysum[26] = mysum[26] + float(mean_temp[i])
            mycount[26] = mycount[26]+1
            else:
    mysum[27] = mysum[27] + float(mean_temp[i])
            mycount[27] = mycount[27]+1
       elif(myyear=='2017'):
    if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
            mysum[28] = mysum[28] + float(mean_temp[i])
            mycount[28] = mycount[28]+1
      elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
            mysum[29] = mysum[29] + float(mean_temp[i])
            mycount[29] = mycount[29]+1
       elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
            mysum[30] = mysum[30] + float(mean_temp[i])
            mycount[30] = mycount[30]+1
            else:
            mysum[31] = mysum[31] + float(mean_temp[i])
            mycount[31] = mycount[31]+1
      elif(myyear=='2018'):
    if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
            mysum[32] = mysum[32] + float(mean_temp[i])
            mycount[32] = mycount[32]+1
       elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
            mysum[33] = mysum[33] + float(mean_temp[i])
            mycount[33] = mycount[33]+1
       elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
```

```
            mysum[34] = mysum[34] + float(mean_temp[i])
                    mycount[34] = mycount[34]+1
                else:
                    mysum[35] = mysum[35] + float(mean_temp[i])
                    mycount[35] = mycount[35]+1
                elif(myyear=='2019'):
            if (date[i][5:7]=='01' or date[i][5:7]=='02' or date[i][5:7]=='03'):
                    mysum[36] = mysum[36] + float(mean_temp[i])
                    mycount[36] = mycount[36]+1
                elif (date[i][5:7]=='04' or date[i][5:7]=='05' or date[i][5:7]=='06'):
                    mysum[37] = mysum[37] + float(mean_temp[i])
                    mycount[37] = mycount[37]+1
                elif (date[i][5:7]=='07' or date[i][5:7]=='08' or date[i][5:7]=='09'):
                    mysum[38] = mysum[38] + float(mean_temp[i])
                    mycount[38] = mycount[38]+1
                   else:
                    mysum[39] = mysum[39] + float(mean_temp[i])
                    mycount[39] = mycount[39]+1
                   for i in range(40):
    mytemps.append(mysum[i]/mycount[i])
          plt.plot(trimina, mytemps, color = 'g', linestyle = 'dashed',
        marker = 'o',label = "Temperature Data")
  plt.xticks(rotation = 90)
plt.xlabel('Trimina')
plt.ylabel('Temperature(°C)')
plt.title('Temperature Report', fontsize = 20)
plt.grid()
plt.legend()
plt.show()
```

Κώδικας RNN

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
data = pd.read_csv('test.csv')
temp = data['mean_temp']
mean_temp = temp.to_numpy()
sum_temp = 0.0
count_temp=0
final_mean_temp=[]
for x in mean_temp:
    if x!='---':
        sum_temp=sum_temp+float(x)
        count_temp=count_temp + 1
        avg_temp=sum_temp/count_temp
for x in mean_temp:
    if x == '---':
        final_mean_temp.append(avg_temp)
    else:
        final_mean_temp.append(float(x))
        hum = data['mean_hum']
mean_hum = hum.to_numpy()
sum_hum = 0.0
count_hum=0
final_mean_hum=[]
for x in mean_hum:
    if x!='---':
        sum_hum=sum_hum+float(x)
```

```
        count_hum=count_hum + 1
       avg_hum=sum_hum/count_hum
for x in mean_hum:
    if x == '---':
        final_mean_hum.append(avg_hum)
    else:
        final_mean_hum.append(float(x))
    pres = data['mean_pres']
mean_pres = pres.to_numpy()
sum_pres = 0.0
count_pres=0
final_mean_pres=[]
for x in mean_pres:
    if x!='---':
        sum_pres=sum_pres+float(x)
        count_pres=count_pres + 1
        avg_pres=sum_pres/count_pres
for x in mean_pres:
    if x == '---':
        final_mean_pres.append(avg_pres)
    else:
        final_mean_pres.append(float(x))
rain = data['rain']
rain = rain.to_numpy()
sum_rain = 0.0
count_rain=0
final_rain=[]
for x in rain:
    if x!='---':
        sum_rain=sum_rain+float(x)
        count_rain=count_rain + 1
```

```
avg_rain=sum_rain/count_rain
for x in rain:
    if x == '---':
        final_rain.append(avg_rain)
    else:
        final_rain.append(float(x))
mean_speed = data['mean_speed_wind']
mean_speed = mean_speed.to_numpy()
sum_speed = 0.0
count_speed=0
final_mean_speed=[]
for x in mean_speed:
    if x!='---':
        sum_speed=sum_speed+float(x)
        count_speed=count_speed + 1
       avg_speed=sum_speed/count_speed
for x in mean_speed:
    if x == '---':
        final_mean_speed.append(avg_speed)
    else:
        final_mean_speed.append(float(x))
dir_wind = data['dir_wind']
dir_wind = dir_wind.to_numpy()
sum_dir_wind = 0.0
count_dir_wind=0
for i in range(len(dir_wind)):
        if(dir_wind[i]=='---'):
        dir_wind[i]=float('-1')
           elif(dir_wind[i]=='E'):
        dir_wind[i]=float('90')
            elif(dir_wind[i]=='ENE'):
        dir_wind[i]=float('67.5')
```

```
      elif(dir_wind[i]=='ESE'):
        dir_wind[i]=float('112.5')
#
    elif(dir_wind[i]=='N'):
        dir_wind[i]=float('0')
#
    elif(dir_wind[i]=='NE'):
        dir_wind[i]=float('45')
#
    elif(dir_wind[i]=='NNE'):
        dir_wind[i]=float('22.5')
#
    elif(dir_wind[i]=='NNW'):
        dir_wind[i]=float('337.5')
#
    elif(dir_wind[i]=='NW'):
        dir_wind[i]=float('315')
#
    elif(dir_wind[i]=='S'):
        dir_wind[i]=float('180')
#
    elif(dir_wind[i]=='SE'):
        dir_wind[i]=float('135')
#
    elif(dir_wind[i]=='SSE'):
        dir_wind[i]=float('157.5')
#
    elif(dir_wind[i]=='SSW'):
        dir_wind[i]=float('202.5')
#
    elif(dir_wind[i]=='SW'):
        dir_wind[i]=float('225')
```

```
#
    elif(dir_wind[i]=='W'):
        dir_wind[i]=float('270')
#
    elif(dir_wind[i]=='WNW'):
       dir_wind[i]=float('292.5')
#
    elif(dir_wind[i]=='WSW'):
        dir_wind[i]=float('247.5')
       final_dir_wind=[]
for x in dir_wind:
     if x!='---':
         sum_dir_wind=sum_dir_wind+float(x)
         count_dir_wind=count_dir_wind + 1

avg_dir_wind=sum_dir_wind/count_dir_wind
for x in dir_wind:
     if x == '---':
         final_dir_wind.append(avg_speed)
     else:
         final_dir_wind.append(float(x))
         #enopoihsh twn dedomenwn
final_data = []

for i in range(len(final_rain)):
    temp_list = []
    temp_list.append(final_dir_wind[i])
    temp_list.append(final_mean_pres[i])
    temp_list.append(final_mean_speed[i])
    temp_list.append(final_mean_temp[i])
    temp_list.append(final_rain[i])
```

```
    temp_list.append(final_mean_hum[i])
    final_data.append(temp_list)
  final_data = np.array(final_data)
#kanonikopoihsh twn dedomenwn
#normalized = preprocessing.normalize(final_data)
#input data
X =  final_data[:,:5]
#output data
y = final_data[:,5]
#kanoume expand to dimension kai sto X kai sto y
#etsi oste na mporei na einai apodeikti i eisodos sto RNN
X = np.expand_dims(X, axis=2)
y = np.expand_dims(y, axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
random_state=42)
from keras.models import Sequential
from keras.layers import Dense, SimpleRNN
model = Sequential()
model.add(SimpleRNN(units=64, input_shape=(None, 1),  activation="sigmoid"))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='rmsprop')
history = model.fit(X_train,y_train, epochs=20, batch_size=64, verbose=2)
trainScore = model.evaluate(X_train,y_train, verbose=0)
print(trainScore)
provlepseis= model.predict(X_test)
```

Κώδικας LSTM

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
data = pd.read_csv('test.csv')
temp = data['mean_temp']
mean_temp = temp.to_numpy()
sum_temp = 0.0
count_temp=0
final_mean_temp=[]
for x in mean_temp:
    if x!='---':
        sum_temp=sum_temp+float(x)
        count_temp=count_temp + 1
       avg_temp=sum_temp/count_temp
for x in mean_temp:
    if x == '---':
        final_mean_temp.append(avg_temp)
    else:
        final_mean_temp.append(float(x))
hum = data['mean_hum']
mean_hum = hum.to_numpy()
sum_hum = 0.0
count_hum=0
final_mean_hum=[]
for x in mean_hum:
    if x!='---':
        sum_hum=sum_hum+float(x)
```

```
        count_hum=count_hum + 1
        avg_hum=sum_hum/count_hum
for x in mean_hum:
    if x == '---':
        final_mean_hum.append(avg_hum)
    else:
        final_mean_hum.append(float(x))
pres = data['mean_pres']
mean_pres = pres.to_numpy()
sum_pres = 0.0
count_pres=0
final_mean_pres=[]
for x in mean_pres:
    if x!='---':
        sum_pres=sum_pres+float(x)
        count_pres=count_pres + 1
        avg_pres=sum_pres/count_pres
for x in mean_pres:
    if x == '---':
        final_mean_pres.append(avg_pres)
    else:
        final_mean_pres.append(float(x))
rain = data['rain']
rain = rain.to_numpy()
sum_rain = 0.0
count_rain=0
final_rain=[]
for x in rain:
    if x!='---':
        sum_rain=sum_rain+float(x)
        count_rain=count_rain + 1
        avg_rain=sum_rain/count_rain
```

```
for x in rain:
    if x == '---':
        final_rain.append(avg_rain)
    else:
        final_rain.append(float(x))
        mean_speed = data['mean_speed_wind']


mean_speed = mean_speed.to_numpy()
sum_speed = 0.0
count_speed=0
final_mean_speed=[]
for x in mean_speed:
    if x!='---':
        sum_speed=sum_speed+float(x)
        count_speed=count_speed + 1
       avg_speed=sum_speed/count_speed
for x in mean_speed:
    if x == '---':
        final_mean_speed.append(avg_speed)
    else:
        final_mean_speed.append(float(x))
dir_wind = data['dir_wind']
dir_wind = dir_wind.to_numpy()
sum_dir_wind = 0.0
count_dir_wind=0
for i in range(len(dir_wind)):
        if(dir_wind[i]=='---'):
        dir_wind[i]=float('-1')
           elif(dir_wind[i]=='E'):
        dir_wind[i]=float('90')
            elif(dir_wind[i]=='ENE'):
        dir_wind[i]=float('67.5')
```

```
      elif(dir_wind[i]=='ESE'):
       dir_wind[i]=float('112.5')
#
   elif(dir_wind[i]=='N'):
       dir_wind[i]=float('0')
#
   elif(dir_wind[i]=='NE'):
       dir_wind[i]=float('45')
#
   elif(dir_wind[i]=='NNE'):
       dir_wind[i]=float('22.5')
#
   elif(dir_wind[i]=='NNW'):
       dir_wind[i]=float('337.5')
#
   elif(dir_wind[i]=='NW'):
       dir_wind[i]=float('315')
#
   elif(dir_wind[i]=='S'):
       dir_wind[i]=float('180')
#
   elif(dir_wind[i]=='SE'):
       dir_wind[i]=float('135')
#
   elif(dir_wind[i]=='SSE'):
       dir_wind[i]=float('157.5')
#
   elif(dir_wind[i]=='SSW'):
       dir_wind[i]=float('202.5')
#
   elif(dir_wind[i]=='SW'):
       dir_wind[i]=float('225')
```

```
#
    elif(dir_wind[i]=='W'):
        dir_wind[i]=float('270')
#
    elif(dir_wind[i]=='WNW'):
       dir_wind[i]=float('292.5')
#
    elif(dir_wind[i]=='WSW'):
        dir_wind[i]=float('247.5')
     final_dir_wind=[]
for x in dir_wind:
     if x!='---':
         sum_dir_wind=sum_dir_wind+float(x)
         count_dir_wind=count_dir_wind + 1

avg_dir_wind=sum_dir_wind/count_dir_wind
for x in dir_wind:
     if x == '---':
         final_dir_wind.append(avg_speed)
     else:
         final_dir_wind.append(float(x))

#enopoihsh twn dedomenwn
final_data = []

for i in range(len(final_rain)):
    temp_list = []
        temp_list.append(final_mean_speed[i])
    temp_list.append(final_rain[i])
    temp_list.append(final_mean_pres[i])
    temp_list.append(final_dir_wind[i])
    temp_list.append(final_mean_temp[i])
```

```
        temp_list.append(final_mean_hum[i])
    final_data.append(temp_list)
    final_data = np.array(final_data)
#input data
X =  final_data[:,:5]
#output data
y = final_data[:,5]
#kanoume expand to dimension kai sto X kai sto y
#etsi oste na mporei na einai apodeikti i eisodos sto LSTM
X = np.expand_dims(X, axis=2)
y = np.expand_dims(y, axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
random_state=42)
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
import tensorflow as tf
batch_size = 1
model=Sequential()
model.add(LSTM(64, batch_input_shape=(batch_size,5,1), stateful=True))
model.add(Dense(1))
model.compile(loss='mean_squared_error', optimizer='adam')
history = model.fit(X_train,y_train, epochs=10, batch_size=batch_size, verbose=2,
shuffle=True)
provlepseis= model.predict(X_test)
```

Κώδικας Clustering

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.cluster import KMeans
from sklearn.cluster import DBSCAN
from sklearn.cluster import AgglomerativeClustering
from sklearn import metrics
from sklearn.cluster import SpectralClustering
from sklearn.cluster import OPTICS
data = pd.read_csv('test.csv')
temp = data['mean_temp']
mean_temp = temp.to_numpy()
sum_temp = 0.0
count_temp=0
final_mean_temp=[]
for x in mean_temp:
    if x!='---':
        sum_temp=sum_temp+float(x)
        count_temp=count_temp + 1
avg_temp=sum_temp/count_temp
for x in mean_temp:
    if x == '---':
        final_mean_temp.append(avg_temp)
    else:
        final_mean_temp.append(float(x))
    hum = data['mean_hum']
```

```
mean_hum = hum.to_numpy()
sum_hum = 0.0
count_hum=0
final_mean_hum=[]
for x in mean_hum:
    if x!='---':
        sum_hum=sum_hum+float(x)
        count_hum=count_hum + 1
avg_hum=sum_hum/count_hum
for x in mean_hum:
    if x == '---':
        final_mean_hum.append(avg_hum)
    else:
        final_mean_hum.append(float(x))
    pres = data['mean_pres']
mean_pres = pres.to_numpy()
sum_pres = 0.0
count_pres=0
final_mean_pres=[]
for x in mean_pres:
    if x!='---':
        sum_pres=sum_pres+float(x)
        count_pres=count_pres + 1
       avg_pres=sum_pres/count_pres
for x in mean_pres:
    if x == '---':
        final_mean_pres.append(avg_pres)
    else:
        final_mean_pres.append(float(x))
rain = data['rain']
rain = rain.to_numpy()
sum_rain = 0.0
```

```
count_rain=0
final_rain=[]
for x in rain:
    if x!='---':
        sum_rain=sum_rain+float(x)
        count_rain=count_rain + 1
avg_rain=sum_rain/count_rain
for x in rain:
    if x == '---':
        final_rain.append(avg_rain)
    else:
        final_rain.append(float(x))
mean_speed = data['mean_speed_wind']
mean_speed = mean_speed.to_numpy()
sum_speed = 0.0
count_speed=0
final_mean_speed=[]
for x in mean_speed:
    if x!='---':
        sum_speed=sum_speed+float(x)
        count_speed=count_speed + 1
avg_speed=sum_speed/count_speed
for x in mean_speed:
    if x == '---':
        final_mean_speed.append(avg_speed)
    else:
        final_mean_speed.append(float(x))
dir_wind = data['dir_wind']
dir_wind = dir_wind.to_numpy()
sum_dir_wind = 0.0
count_dir_wind=0
```

```
for i in range(len(dir_wind)):
        if(dir_wind[i]=='---'):
        dir_wind[i]=float('-1')
          elif(dir_wind[i]=='E'):
        dir_wind[i]=float('90')
           elif(dir_wind[i]=='ENE'):
        dir_wind[i]=float('67.5')

    elif(dir_wind[i]=='ESE'):
        dir_wind[i]=float('112.5')
#
    elif(dir_wind[i]=='N'):
        dir_wind[i]=float('0')
#
    elif(dir_wind[i]=='NE'):
        dir_wind[i]=float('45')
#
    elif(dir_wind[i]=='NNE'):
        dir_wind[i]=float('22.5')
#
    elif(dir_wind[i]=='NNW'):
        dir_wind[i]=float('337.5')
#
    elif(dir_wind[i]=='NW'):
        dir_wind[i]=float('315')
#
    elif(dir_wind[i]=='S'):
        dir_wind[i]=float('180')
#
    elif(dir_wind[i]=='SE'):
        dir_wind[i]=float('135')
#
```

```
    elif(dir_wind[i]=='SSE'):
        dir_wind[i]=float('157.5')
#
    elif(dir_wind[i]=='SSW'):
        dir_wind[i]=float('202.5')
#
    elif(dir_wind[i]=='SW'):
        dir_wind[i]=float('225')
#
    elif(dir_wind[i]=='W'):
        dir_wind[i]=float('270')
#
    elif(dir_wind[i]=='WNW'):
       dir_wind[i]=float('292.5')
#
    elif(dir_wind[i]=='WSW'):
        dir_wind[i]=float('247.5')
final_dir_wind=[]
for x in dir_wind:
    if x!='---':
        sum_dir_wind=sum_dir_wind+float(x)
        count_dir_wind=count_dir_wind + 1
avg_dir_wind=sum_dir_wind/count_dir_wind

for x in dir_wind:
    if x == '---':
        final_dir_wind.append(avg_speed)
    else:
        final_dir_wind.append(float(x))
#enopoihsh twn dedomenwn
```

```
final_data = []
for i in range(len(final_rain)):
    temp_list = []
    temp_list.append(final_dir_wind[i])
    temp_list.append(final_mean_hum[i])
    temp_list.append(final_mean_pres[i])
    temp_list.append(final_mean_speed[i])
    temp_list.append(final_mean_temp[i])
    temp_list.append(final_rain[i])
    final_data.append(temp_list)
    final_data = np.array(final_data)
X=final_data
#efarmozw kmeans algorithmo
kmeans = KMeans(n_clusters=2, random_state=0).fit(X)

etiketes1=kmeans.labels_

#efarmozw DBSCAN algorithmo den mas kanei giati oi etiketes exoun arnitikes times
#dbscan = DBSCAN(eps=0.2, min_samples=2).fit(X)
#etiketes2=dbscan.labels_
#efarmozw agglomerative algorithmo
agglomer = AgglomerativeClustering().fit(X)
etiketes3=agglomer.labels_
#efarmozw spectral clustering
#spectClust =
SpectralClustering(n_clusters=2,assign_labels='discretize',random_state=0).fit(X)
#etiketes4=spectClust.labels_
#efarmozw OPTICS de mas kanei
#optics = OPTICS(min_samples=2).fit(X)
#etiketes5=optics.labels_
#ypologizw tin metriki silhouette
#print(metrics.silhouette_score(X,etiketes1,metric='euclidean'))
```

```
#print(metrics.silhouette_score(X,etiketes3,metric='euclidean'))
#simplironw tis thermokrasies xrisimopoiontas tis times tou kmeans
sum_temper1=0.0
count_temper1=0
for i in range(len(final_data)):
  if etiketes1[i]==0:
        sum_temper1=sum_temper1+float(final_data[i][4])
        count_temper1=count_temper1+1
average_temper1=sum_temper1/count_temper1
 sum_temper2=0.0
count_temper2=0
for i in range(len(final_data)):
  if etiketes1[i]==1:
        sum_temper2=sum_temper2+float(final_data[i][4])
        count_temper2=count_temper2+1
average_temper2=sum_temper2/count_temper2
#simplironw tis thermokrasies xrisimopoiontas tis times tou Agglomerative
sum_temper3=0.0
count_temper3=0
for i in range(len(final_data)):
  if etiketes3[i]==0:
        sum_temper3=sum_temper3+float(final_data[i][4])
        count_temper3=count_temper3+1
average_temper3=sum_temper3/count_temper3
sum_temper4=0.0
count_temper4=0
for i in range(len(final_data)):
  if etiketes3[i]==1:
        sum_temper4=sum_temper4+float(final_data[i][4])
        count_temper4=count_temper4+1
average_temper4=sum_temper4/count_temper4
```

```
#simplironw ta humidities xrisimopoiontas tis times tou kmeans
sum_humidity1=0.0
count_humidity1=0
for i in range(len(final_data)):
    if etiketes1[i]==0:
        sum_humidity1=sum_humidity1+float(final_data[i][1])
        count_humidity1=count_humidity1+1
average_humidity1=sum_humidity1/count_humidity1
 sum_humidity2=0.0
count_humidity2=0
for i in range(len(final_data)):
    if etiketes1[i]==1:
        sum_humidity2=sum_humidity2+float(final_data[i][1])
        count_humidity2=count_humidity2+1
average_humidity2=sum_humidity2/count_humidity2
     #simplironw ta humidities xrisimopoiontas tis times tou Agglomerative
sum_humidity3=0.0
count_humidity3=0
for i in range(len(final_data)):
   if etiketes3[i]==0:
            sum_humidity3=sum_humidity3+float(final_data[i][1])
            count_humidity3=count_humidity3+1
average_humidity3=sum_humidity3/count_humidity3
sum_humidity4=0.0
count_humidity4=0
for i in range(len(final_data)):
   if etiketes3[i]==1:
            sum_humidity4=sum_humidity4+float(final_data[i][1])
            count_humidity4=count_humidity4+1
average_humidity4=sum_humidity4/count_humidity4
#simplironw ta pressures xrisimopoiontas tis times tou kmeans
sum_pres1=0.0
```

```
count_pres1=0
for i in range(len(final_data)):
  if etiketes1[i]==0:
        sum_pres1=sum_pres1+float(final_data[i][2])
        count_pres1=count_pres1+1
average_pres1=sum_pres1/count_pres1
 sum_pres2=0.0
count_pres2=0
for i in range(len(final_data)):
  if etiketes1[i]==1:
        sum_pres2=sum_pres2+float(final_data[i][2])
        count_pres2=count_pres2+1
average_pres2=sum_pres2/count_pres2
#simplironw ta pressures xrisimopoiontas tis times tou Agglomerative
sum_pres3=0.0
count_pres3=0
for i in range(len(final_data)):
  if etiketes3[i]==0:
        sum_pres3=sum_pres3+float(final_data[i][2])
        count_pres3=count_pres3+1
average_pres3=sum_pres3/count_pres3
sum_pres4=0.0
count_pres4=0
for i in range(len(final_data)):
  if etiketes3[i]==1:
        sum_pres4=sum_pres4+float(final_data[i][2])
        count_pres4=count_pres4+1
average_pres4=sum_pres4/count_pres4
#simplironw ta speed wind xrisimopoiontas tis times tou kmeans
sum_speed1=0.0
count_speed1=0
```

```
for i in range(len(final_data)):
  if etiketes1[i]==0:
        sum_speed1=sum_speed1+float(final_data[i][3])
        count_speed1=count_speed1+1
average_speed1=sum_speed1/count_speed1
 sum_speed2=0.0
count_speed2=0
 for i in range(len(final_data)):
  if etiketes1[i]==1:
        sum_speed2=sum_speed2+float(final_data[i][3])
        count_speed2=count_speed2+1
average_speed2=sum_speed2/count_speed2

#simplironw ta speed wind xrisimopoiontas tis times tou Agglomerative

sum_speed3=0.0

count_speed3=0
for i in range(len(final_data)):
  if etiketes3[i]==0:
        sum_speed3=sum_speed3+float(final_data[i][3])
        count_speed3=count_speed3+1
average_speed3=sum_speed3/count_speed3
sum_speed4=0.0
count_speed4=0
for i in range(len(final_data)):
  if etiketes3[i]==1:
        sum_speed4=sum_speed4+float(final_data[i][3])
        count_speed4=count_speed4+1

average_speed4=sum_speed4/count_speed4
```

```
#simplironw ta rains xrisimopoiontas tis times tou kmeans
sum_rain1=0.0
count_rain1=0
for i in range(len(final_data)):
  if etiketes1[i]==0:
          sum_rain1=sum_rain1+float(final_data[i][5])
          count_rain1=count_rain1+1

average_rain1=sum_rain1/count_rain1
 sum_rain2=0.0
count_rain2=0
 for i in range(len(final_data)):
  if etiketes1[i]==1:
          sum_rain2=sum_rain2+float(final_data[i][5])
          count_rain2=count_rain2+1
average_rain2=sum_rain2/count_rain2
#simplironw rains xrisimopoiontas tis times tou Agglomerative
sum_rain3=0.0
count_rain3=0
for i in range(len(final_data)):
  if etiketes3[i]==0:
          sum_rain3=sum_rain3+float(final_data[i][5])
          count_rain3=count_rain3+1
average_rain3=sum_rain3/count_rain3
sum_rain4=0.0
count_rain4=0
for i in range(len(final_data)):
  if etiketes3[i]==1:
          sum_rain4=sum_rain4+float(final_data[i][5])
          count_rain4=count_rain4+1
         average_rain4=sum_rain4/count_rain4
```

Κώδικας Classification

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn import preprocessing
from sklearn import utils
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
data = pd.read_csv('test.csv')
temp = data['mean_temp']
mean_temp = temp.to_numpy()
sum_temp = 0.0
count_temp=0
final_mean_temp=[]
for x in mean_temp:
    if x!='---':
        sum_temp=sum_temp+float(x)
```

```
        count_temp=count_temp + 1
avg_temp=sum_temp/count_temp
for x in mean_temp:
    if x == '---':
        final_mean_temp.append(avg_temp)
    else:
        final_mean_temp.append(float(x))
    hum = data['mean_hum']
mean_hum = hum.to_numpy()
sum_hum = 0.0
count_hum=0
final_mean_hum=[]
for x in mean_hum:
    if x!='---':
        sum_hum=sum_hum+float(x)
        count_hum=count_hum + 1
        avg_hum=sum_hum/count_hum
for x in mean_hum:
    if x == '---':
        final_mean_hum.append(avg_hum)
    else:
        final_mean_hum.append(float(x))
pres = data['mean_pres']
mean_pres = pres.to_numpy()
sum_pres = 0.0
count_pres=0
final_mean_pres=[]
for x in mean_pres:
    if x!='---':
        sum_pres=sum_pres+float(x)
        count_pres=count_pres + 1
```

```
avg_pres=sum_pres/count_pres
for x in mean_pres:
    if x == '---':
        final_mean_pres.append(avg_pres)
    else:
        final_mean_pres.append(float(x))
rain = data['rain']
rain = rain.to_numpy()
sum_rain = 0.0
count_rain=0
final_rain=[]
for x in rain:
    if x!='---':
        sum_rain=sum_rain+float(x)
        count_rain=count_rain + 1
        avg_rain=sum_rain/count_rain
for x in rain:
    if x == '---':
        final_rain.append(avg_rain)
    else:
        final_rain.append(float(x))
            mean_speed = data['mean_speed_wind']
mean_speed = mean_speed.to_numpy()
sum_speed = 0.0
count_speed=0
final_mean_speed=[]
for x in mean_speed:
    if x!='---':
        sum_speed=sum_speed+float(x)
        count_speed=count_speed + 1
        avg_speed=sum_speed/count_speed
for x in mean_speed:
```

```
    if x == '---':
        final_mean_speed.append(avg_speed)
    else:
        final_mean_speed.append(float(x))
dir_wind = data['dir_wind']
dir_wind = dir_wind.to_numpy()
sum_dir_wind = 0.0
count_dir_wind=0
for i in range(len(dir_wind)):
        if(dir_wind[i]=='---'):
        dir_wind[i]=float('-1')
           elif(dir_wind[i]=='E'):
        dir_wind[i]=float('90')
            elif(dir_wind[i]=='ENE'):
        dir_wind[i]=float('67.5')
       elif(dir_wind[i]=='ESE'):
        dir_wind[i]=float('112.5')
#
    elif(dir_wind[i]=='N'):
        dir_wind[i]=float('0')
#
    elif(dir_wind[i]=='NE'):
        dir_wind[i]=float('45')
#
    elif(dir_wind[i]=='NNE'):
        dir_wind[i]=float('22.5')
#
    elif(dir_wind[i]=='NNW'):
        dir_wind[i]=float('337.5')
#
    elif(dir_wind[i]=='NW'):
        dir_wind[i]=float('315')
```

```
#
    elif(dir_wind[i]=='S'):
        dir_wind[i]=float('180')
#
    elif(dir_wind[i]=='SE'):
        dir_wind[i]=float('135')
#
    elif(dir_wind[i]=='SSE'):
        dir_wind[i]=float('157.5')
#
    elif(dir_wind[i]=='SSW'):
        dir_wind[i]=float('202.5')
#
    elif(dir_wind[i]=='SW'):
        dir_wind[i]=float('225')
#
    elif(dir_wind[i]=='W'):
        dir_wind[i]=float('270')
#
    elif(dir_wind[i]=='WNW'):
       dir_wind[i]=float('292.5')
#
    elif(dir_wind[i]=='WSW'):
        dir_wind[i]=float('247.5')
        final_dir_wind=[]
for x in dir_wind:
    if x!='---':
        sum_dir_wind=sum_dir_wind+float(x)
        count_dir_wind=count_dir_wind + 1
        avg_dir_wind=sum_dir_wind/count_dir_wind
for x in dir_wind:
    if x == '---':
```

```
            final_dir_wind.append(avg_speed)
        else:
            final_dir_wind.append(float(x))
        #enopoihsh twn dedomenwn
final_data = []

for i in range(len(final_rain)):
    temp_list = []
    temp_list.append(final_rain[i])
    temp_list.append(final_mean_temp[i])
    temp_list.append(final_mean_speed[i])
    temp_list.append(final_mean_pres[i])
    temp_list.append(final_dir_wind[i])
    temp_list.append(final_mean_hum[i])
    final_data.append(temp_list)
final_data = np.array(final_data)
#efarmozoume random forest
#input data
X = final_data[:,:5]
#output data
y = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
y = lab_enc.fit_transform ( y )
#diaspasi se train set kai test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
#dimiourgoume to montelo toy RandomForestClassifier
clf = RandomForestClassifier(max_depth=2, random_state=0)
#ekpaideysi
clf.fit(X_train, y_train)
#provlepsi
y_pred=clf.predict(X_test)
```

```
#ypologizoume to precision
#precision = precision_score(y_test, y_pred, average = 'micro')
#print('Precision: %.3f' % precision)
#efarmozoume knn
#input data
X1 = final_data[:,:5]
#output data
y1 = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
y1 = lab_enc.fit_transform ( y1 )
#diaspasi se train set kai test set
X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.20)
#dimiourgoume to montelo tou knn
neigh = KNeighborsClassifier(n_neighbors=5)
#ekpaideysi
neigh.fit(X1_train, y1_train)
#provlepsi
y1_pred=neigh.predict(X1_test)
#ypologizoume to precision
#precision = precision_score(y1_test, y1_pred, average = 'micro')
#print('Precision: %.3f' % precision)
#efarmozoume logistic regression
#input data
X2 = final_data[:,:5]
#output data
y2 = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
y2 = lab_enc.fit_transform ( y2 )
#diaspasi se train set kai test set
X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2, test_size=0.20)
#dimiourgoume to montelo tou logistic regression
clf2 = LogisticRegression(random_state=0)
```

```
#ekpaideysi
clf2.fit(X2_train,y2_train)
#provlepsi
y2_pred=clf2.predict(X2_test)
#ypologizoume to precision
#precision = precision_score(y2_test, y2_pred, average = 'micro')
#print('Precision: %.3f' % precision)
#efarmozoume decision tree
#input data
X3 = final_data[:,:5]
#output data
y3 = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
y3 = lab_enc.fit_transform ( y3 )
#diaspasi se train set kai test set
X3_train, X3_test, y3_train, y3_test = train_test_split(X3, y3, test_size=0.20)
#dimiourgoume to montelo tou decision tree
clf3 = DecisionTreeClassifier(random_state=0)
#ekpaideysi
clf3.fit(X3_train,y3_train)
#provlepsi
y3_pred=clf3.predict(X3_test)
#ypologizoume to precision
#precision = precision_score(y3_test, y3_pred, average = 'micro')
#print('Precision: %.3f' % precision)
#efarmozoume SVM
#input data
X4 = final_data[:,:5]
#output data
y4 = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
```

```
y4 = lab_enc.fit_transform ( y4 )
#diaspasi se train set kai test set
X4_train, X4_test, y4_train, y4_test = train_test_split(X4, y4, test_size=0.20)
#dimiourgoume to montelo tou SVM
clf4 = make_pipeline(StandardScaler(), SVC(gamma='auto'))
#ekpaideysi
clf4.fit(X4_train,y4_train)
#provlepsi
y4_pred=clf4.predict(X4_test)
#ypologizoume to precision
#precision = precision_score(y4_test, y4_pred, average = 'micro')
#print('Precision: %.3f' % precision)
#efarmozoume naive bayes
#input data
X5 = final_data[:,:5]
#output data
y5 = final_data[:,5]
lab_enc = preprocessing.LabelEncoder()
y5 = lab_enc.fit_transform ( y5 )
#diaspasi se train set kai test set
X5_train, X5_test, y5_train, y5_test = train_test_split(X5, y5, test_size=0.20)
#dimiourgoume to montelo tou Naive Bayes
gnb = GaussianNB()
#ekpaideysi
gnb.fit(X5_train,y5_train)
#provlepsi
y5_pred=gnb.predict(X5_test)
#ypologizoume to precision
precision = precision_score(y4_test, y4_pred, average = 'micro')
print('Precision: %.3f' % precision)
```