

**Εξόρυξη Γνώσης από Βάσεις Δεδομένων και τον Παγκόσμιο Ιστό**

**The case of flight passengers prediction**

**Team name on Kaggle:***V and K*

**Ονοματεπώνυμο ΑΜ**

*Άγγελος Βανακάρης 3150006*

*Κωνσταντίνος Κουμπανάκης 3150200*

Contents

[Εισαγωγή 2](#_Toc534492469)

[Προεπεξεργασία δεδομένων 2](#_Toc534492470)

[Μοντέλα 3](#_Toc534492471)

[ΚΝΝ 3](#_Toc534492472)

[Logistic Regression 4](#_Toc534492473)

[Decision Tree Classifier 4](#_Toc534492474)

[Random Forest Classifier 4](#_Toc534492475)

[Multi Layer Perceptron Classifier 4](#_Toc534492476)

# Εισαγωγή

Στην εργασία αυτή σκοπός μας ήταν αρχικά να κατανοήσουμε τη δομή των δεδομένων που μας δόθηκαν, τους ορθούς τρόπους κατηγοριοποίησης των παραδειγμάτων, καθώς και τις διαφορές ανάμεσα στα μοντέλα κατηγοριοποίησης. Τα μοντέλα που χρησιμοποιήσαμε καθώς και η προεπεξεργασία των δεδομένων που κάναμε είναι υλοποιημένα σε γλώσσα ***Python*** *(έκδοση 3.6)* στο περιβάλλον ***Jupyter Notebook*** της πλατφόρμας ***Anaconda***.

Για να μπορέσουμε να εφαρμόσουμε και να αξιολογήσουμε τα μοντέλα κατηγοριοποίησης διαχωρίσαμε τα δεδομένα εκπαίδευσης σε 2 ομάδες δεδομένων, εκ των οποίων τα δεδομένα της πρώτης τα χρησιμοποιήσαμε ως δεδομένα εκπαίδευσης *(train data)* των μοντέλων και αυτά της δεύτερης ως δεδομένα δοκιμής *(test data)*. Αυτό το πετύχαμε με τη χρήση της συνάρτησης **train\_test\_split** της βιβλιοθήκης **model\_selection** που περιέχει η **scikit-learn**, η οποία χρησιμοποιείται για πολλές εφαρμογές μηχανικής μάθησης.

# Προεπεξεργασία δεδομένων

Το πρώτο στάδιο της προεπεξεργασίας των δεδομένων ήταν η μελέτη των μεταβλητών των δεδομένων καθώς και των πιθανών συσχετίσεων που υπάρχουν ανάμεσα σε αυτές.

Αρχικά, παρατηρήσαμε ότι κάποιες μεταβλητές είναι κατηγορικές, επομένως δεν μπορούν να χρησιμοποιηθούν κατευθείαν από τα μοντέλα που χρησιμοποιούν αριθμητικούς υπολογισμούς.  
Πιο συγκεκριμένα, οι μεταβλητές **Departure**, **CityDeparture**, **Arrival** και **CityArrival**.

Επομένως, το πρώτο μας βήμα ήταν η μετατροπή των μεταβλητών αυτών σε αριθμητικές. Αυτό το πετύχαμε με τη χρήση της κλάσης **LabelEncoder** της βιβλιοθήκης **scikit-learn**. Η μετατροπή έγινε με τη χρήση των συναρτήσεων **fit[[1]](#footnote-1)** και **transform[[2]](#footnote-2)** της κλάσης **LabelEncoder** για τα *train data* και *test data*.

To επόμενο βήμα για την βαθύτερη κατανόηση των δεδομένων ως προς τις συσχετίσεις τους αλλά και τις τιμές που μπορούν να έχουν, κρίναμε απαραίτητο να οπτικοποιήσουμε τα δεδομένα μας. Αυτό μας έδωσε κυρίως την δυνατότητα να εντοπίσουμε τυχόν συσχετίσεις μεταξύ των μεταβλητών ανά δύο καθώς και να συμπαιράνουμε το εύρος τιμών που μπορούν να έχει η κάθε μια μεταβλητή. Η οπτικοποίηση των δεδομένων μας έγινε με την χρήση της βιβλιοθήκης **matplotlib**,η οποία περιέχει την κλάση **pyplot** η οποία και χρησημοποιήθηκε για το σκοπό αυτό.

Στην συνέχεια, αποφασίσαμε να παράξουμε νέα χαρακτηριστικά, τα οποία θα πηγάζουν από το συνδιασμό 2 ή περισσότερων μεταβλητών. Η πρώτη μας ενέργεια ήταν να διαχωρίσουμε την ημέρα *(ημέρα της εβδομάδας με τη βοήθεια της συνάρτησης* ***weekday*** *της* ***datetime****)*, το μήνα, την εποχή και το έτος κάθε πτήσης στους πίνακες των δεδομένων εκπάιδευσης *train data* και *test data*. Τη μεταβλητή εποχή *(****Season****)* φροντίσαμε να την κρατάμε με αριθμητική τιμή, καθώς η αρχική συνάρτηση της μετατροπής που φτιάξαμε επέστρεφε αλφαριθμητικές τιμές *(“Winter”, “Summer”, etc)*. Με τον τρόπο αυτό δημιουργήσαμε 4 νέα χαρακτηριστικά τα οποία και προσθέσαμε στους πίνακες αυτούς.

Επιπλέον, θεώρησαμε πως θα είχε ιδιαίτερη σημασία να γνωρίζουμε αν μια συγκεκριμένη πτήση συμβαίνει σε περίοδο διακοπών. Στην περίπτωση αυτή, θα αναμέναμε να επηρεάζεται ο αριθμός των επιβατών μιας πτήσης και ενδεχομένως να αυξηθεί και συνεπώς θα ήταν μια πληροφορία αρκετά χρήσιμη για το μοντέλο το οποίο θα καταλήγαμε να χρησημοποιήσουμε. Για τον λόγο αυτό δημιουργήσαμε την συνάρητηση **isHoliday**, η οποία δέχεται μια ημερομηνία τύπου Date ή String σε format ημερομηνίας και επιστρέφει 1 αν πρόκειται για ημερομηνία σε περίοδο διακοπών και 0 διαφορετικά. Ο έλεγχος αυτός έγινε με τη βοήθεια της βιβλιοθήκης **holiday**, η οποία περιέχει λίστες με τις ημερομηνίες γιορτών για αρκετές χώρες. Εμείς διαλέξαμε τις Ηνωμένες Πολιτείες της Αμερικής, καθώς σύμφωνα με μία σύντομη έρευνα που κάναμε παρατηρήσαμε πως τα αεροδρόμια που εμπεριέχονται στα δεδομένα βρίσκονται στη χώρα αυτή. Έτσι, δημιουργήσαμε μια νέα μεταβλητή σε κάθε πίνακα, την **Holiday** με τιμές 1 ή 0 ανάλογα με το αν η ημερομηνία της εκάστοτε πτήσης είναι σε περίοδο διακοπών ή όχι.

Επιπρόσθετα, μέσω των μεταβλητών Longitude και Latitude των αεροδρομίων απογείωσης και προσγείωσης των πτήσεων υπολογίσαμε την απόσταση ανάμεσά τους, δημιουργώντας έτσι την μεταβλητή **Distance**. Η μεταβλητή αυτή μας έδωσε επίσης τη δυνατότητα να εξάγουμε σαν επιπλέον πληροφορία το χρόνο της κάθε πτήσης *(προσεγγιστικά, χρησιμοποιώντας 600 χιλμ/ώρα σαν μέση ταχύτητα ενός αεροπλάνου, την οποία διαλέξαμε βάσει των πληροφοριών για τις πτήσεις που βρήκαμε στο διαδίκτυο)*. Συνεπώς, δημιουργήσαμε μια ακόμα μεταβλητή την **ETA** *(Estimated Time of Arrival)*. Με την ίδια λογική και ακολουθώντας τα αντίστοιχα βήματα, δημιουργήσαμε τη μεταβλητή **cost** *(εκτιμώμενο κόστος μίας πτήσης με βάση την απόσταση)*.

Τέλος, δημιουργήσαμε μία μεταβλητή **std\_mean\_combined**, η οποία προκύπτει από τη διαίρεση των μεταβλητών **WeeksToDeparture** και **std\_wtd**. Παρατηρήσαμε ότι τα μοντέλα δίνουν καλύτερα αποτελέσματα με τη μεταβλητή αυτήν.

# Μοντέλα

## ΚΝΝ

Έπειτα από την προ επεξεργασία των δεδομένων μας, αποφασίσαμε να αρχίσουμε να διερευνούμε τα μοντέλα που ήταν κατάλληλα για να μας δώσουν την καλύτερη δυνατή πρόβλεψη. Ένα από αυτά ήταν ο ταξινομητής **KNN** *(K Nearest Neighbors)*. Δίχως κάποια αλλαγή των παραμέτρων του ταξινομητή αλλά αλλάζοντας συνεχώς τη σταθερά k, τα αρχικά score είχαν ένα εύρος τιμών από ***0.25*** έως ***0.32***. Καθώς το score πρόβλεψης στα δεδομένα δοκιμής *(test data)* ήταν σε χαμηλά επίπεδα, ξεκινήσαμε να εφαρμόζουμε τα βήματα της προ επεξεργασίας των δεδομένων που αναφέρθηκαν προηγουμένως , βλέποντας μια πρώτη αύξηση του score πρόβλεψης σε επίπεδα της τάξης του ***0.1***. Ένας ακόμη λόγος για τον οποίο καταφέραμε να βελτιώσουμε το score του μοντέλου και να φτάσει από ***0.40*** έως ***0.42***, ήταν η αλλαγή των παραμέτρων του ταξινομητή KNN. Μεταβάλαμε κυρίως τον αλγόριθμο που χρησιμοποιεί ο ταξινομητής και τον ορίσαμε σε ***‘kd\_tree’***, καθώς η δεύτερη επιλογή που μας δινόταν, η ***‘ball\_tree’***, δεν έδωσε ιδιαίτερα καλό score. Επίσης, μεταβάλαμε και το συνολικό αριθμό φύλλων *(leaf size)* που μπορεί να έχει το δένδρο που κατασκευάζει ο αλγόριθμος. Πειραματικά και έπειτα από αρκετές προσπάθειες ορίσαμε το *leaf\_size = 45*. Σε αυτό το σημείο καταλήξαμε στο συμπέρασμα ότι ο KNN, με τις ενέργειες που είχαμε κάνει έως τότε, δεν θα μπορούσε να μας δώσει κάποιο καλύτερο score πρόβλεψης από ***0.40*** *-* ***0.42***.

## Logistic Regression

Ο ταξινομητής ***Logistic Regression*** της βιβλιοθήκης **sklearn.linear\_model** αποτέλεσε το δεύτερο ταξινομητή που χρησιμοποιήσαμε στο πρόβλημα μας. Δυστυχώς, το score πρόβλεψης δεν ξεπέρασε το ***0.25***. Ως ένα κομμάτι της προεπεξεργασίας των δεδομένων μας ήταν η κανονικοποίηση των τιμών όλων των μεταβλητών μας στο διάστημα [0, 1] με την βοήθεια του **MinMaxScaler** της βιβλιοθήκης **sklearn.preprocessing**, κάτι το οποίοδεν βοήθησε στο να αυξήσουμε το score πρόβλεψης αυτού του μοντέλου. Μέσα από πειραματικές αλλαγές των παραμέτρων του ταξινομητή λάβαμε κάποια καλύτερα αποτελέσματα, όμως οι βελτιώσεις δεν ήταν σημαντικές.

## Decision Tree Classifier

Ένα από τα μοντέλα ταξινόμησης το οποίο λειτούργησε αρκετά καλά σε σχέση με τα προηγούμενα ήταν τα **Δέντρα Απόφασης** *(Decision Trees)*της βιβλιοθήκης **sklearn.tree**. Έχοντας ήδη εφαρμόσει την προ επεξεργασία των δεδομένων που αναφέρθηκε στην προηγούμενη ενότητα, τα Decision Trees έδωσαν απευθείας, ένα score της τάξεως του ***0.43***. Οποιαδήποτε πειραματική αλλαγή των παραμέτρων του ταξινομητή, όπως **max\_depth** του δέντρου που κατασκευάζει ο ταξινομητής, αλλαγή των τυχαίων καταστάσεων **random\_state** *(και άρα την δημιουργία νέων δέντρων σε κάθε δοκιμή)*, δεν επέφερε σημαντική βελτίωση του score.

## Random Forest Classifier

Ο ταξινομητής **Random Forest** της βιβλιοθήκης **sklearn.ensemble**, ήταν αυτός ο οποίος μας έδωσε ένα από τα καλύτερα score πρόβλεψης. Eξ’ αρχής, με τα δεδομένα μας να είναι πλήρως προ επεξεργασμένα, ο αλγόριθμος έδωσε score της τάξεως του ***0.45***. Ο τρόπος λειτουργίας των Random Forests είναι ο εξής: Κατασκευάζουν πολλά, τυχαία Decision Trees για να προβλέψουν τμήματα του αρχικού dataset και έτσι μπορούν να αποφασίσουν για τον καλύτερο συνδυασμό δέντρων. Μία από τις βασικές μεταβολές που κάναμε ήταν η μεταβολή της παραμέτρου **n\_estimators**. Η παράμετρος αυτή μας επιτρέπει να αποφασίζουμε πόσα Δέντρα Απόφασης θα κατασκευάσει το μοντέλο προβλέψεων. Έπειτα από αρκετές προσπάθειες ορίσαμε την παράμετρο αυτή σε 32 και αλλάξαμε μετά από διάφορες δοκιμές την τυχαία κατάσταση σε 7 *(random\_state = 7),* ώστε κάθε φορά να δημιουργείται το ίδιο μοντέλο για να έχουμε ένα πιο σταθερό αποτέλεσμα. Με τις αλλαγές αυτές το μοντέλο έφερε score περίπου ***0.50***.

## Multi Layer Perceptron Classifier

Ο ταξινομητής **Multi Layer Perceptron** της βιβλιοθήκης **sklearn.neural\_network**, ήταν αυτός ο οποίος μας έδωσε το καλύτερο score πρόβλεψης. Ο ταξινομητής αυτός υλοποιεί ένα νευρωνικό δίκτυο και δίνει τη δυνατότητα στους χρήστες του να τροποποιήσουν τη μορφή του και τον τρόπο εκπαίδευσης, καθώς μπορούν να αλλάξουν την αρχιτεκτονική των κρυφών επιπέδων, το βήμα της εκπαίδευσης *(learning\_rate)*, το πλήθος των εποχών *(max\_iter)*, την τυχαία κατάσταση *(random\_state)* που θα χρησιμοποιηθεί για να αρχικοποιηθούν τα βάρη των συνδέσεων του νευρωνικού δικτύου, τον τρόπο εκπαίδευσης *(solver, εμείς επιλέξαμε τη μέθοδο ‘stochastic gradient descent’)*, καθώς και τη συνάρτηση ενεργοποίησης που χρησιμοποιούν οι νευρώνες *(activation, εμείς χρησιμοποιήσαμε τη ‘logistic’)*. Ο ταξινομητής αυτός χρειάζεται αρκετές πειραματικές αλλαγές για να δώσει καλά αποτελέσματα στο score, αλλά μπορεί με σωστή παραμετροποίηση να φτάσει πολύ υψηλό score. Ένα χαρακτηριστικό που δείχνει τη δύναμη του μοντέλου και γενικότερα των νευρωνικών δικτύων, αλλά και ένα συχνό πρόβλημα που εμφανίζουν αυτά τα μοντέλα μηχανικής μάθησης είναι η υπερεκπαίδευση *(****overfitting****)*. Όταν συμβαίνει αυτό, το μοντέλο δίνει πολύ καλά αποτελέσματα στα δεδομένα εκπαίδευσης *(train data)*, όμως δεν δίνει το ίδιο καλά αποτελέσματα σε άγνωστα δεδομένα. Ενδεικτικά, μερικά μοντέλα νευρωνικών δικτύων έδιναν στα δεδομένα εκπαίδευσης score ίσο με ***0.64***, ενώ στα άγνωστα δεδομένα μόλις ***0.52***. Το πρόβλημα αυτό εμφανίζεται επειδή το μοντέλο προσπαθεί με κάθε τρόπο να προσαρμοστεί στα δεδομένα εκπαίδευσης, τα οποία αποτελούν ένα υποσύνολο των πραγματικών δεδομένων. Έτσι, το μοντέλο κατασκευάζει τα κριτήριά του έτσι ώστε να κατηγοριοποιούν σωστά τα δεδομένα που ανήκουν στο σύνολο αυτό και όχι στο υπερσύνολο των πραγματικών δεδομένων επομένως και των πειραματικών. Ένας τρόπος αντιμετώπισης του παραπάνω προβλήματος είναι η απενεργοποίηση ορισμένων συνδέσεων του νευρωνικού δικτύου που φαίνεται να μην επηρεάζονται πολύ κατά την εκπαίδευση *(έχουν σταθεροποιηθεί τα βάρη τους, επομένως δε συμβάλλουν στο σφάλμα της εκπαίδευσης)*, δίνοντας έτσι τη δυνατότητα στις υπόλοιπες συνδέσεις να εκπαιδευτούν καλύτερα (*διαισθητικά, να δυναμώσουν και εκείνες το ίδιο)*. Έχοντας πλέον ένα πιο ισορροπημένο νευρωνικό δίκτυο *(με σχετικά σταθερές τιμές σε όλα τα βάρη των συνδέσεων)*, μπορούμε να υποθέσουμε πως το μοντέλο μας δίνει αρκετή έμφαση σε όλες τις μεταβλητές ενός παραδείγματος και λαμβάνει υπ’ όψιν τη συνολική εικόνα αυτού, καθιστώντας το πιο «αντικειμενικό» και επομένως πιο ικανό να κατηγοριοποιήσει ένα παράδειγμα που δε βρίσκεται στο σύνολο των δεδομένων εκαπαίδευσης.

1. *Η συνάρτηση αυτή παίρνει ως όρισμα μία μεταβλητή και αναλύει τις κατηγορικές τιμές της. Στη συνέχεια κατασκευάζει μία λίστα στην οποία αντιστοιχεί κάθε τέτοια τιμή με μία αριθμητική τιμή.* [↑](#footnote-ref-1)
2. *Η συνάρτηση αυτή παίρνει ως όρισμα μία στήλη και μετατρέπει κάθε κατηγορική μεταβλητή που περιέχει στην αντίστοιχη αριθμητική τιμή της με τη βοήθεια της λίστας που έχει δημιουργήσει η συνάρτηση* ***fit****.* [↑](#footnote-ref-2)