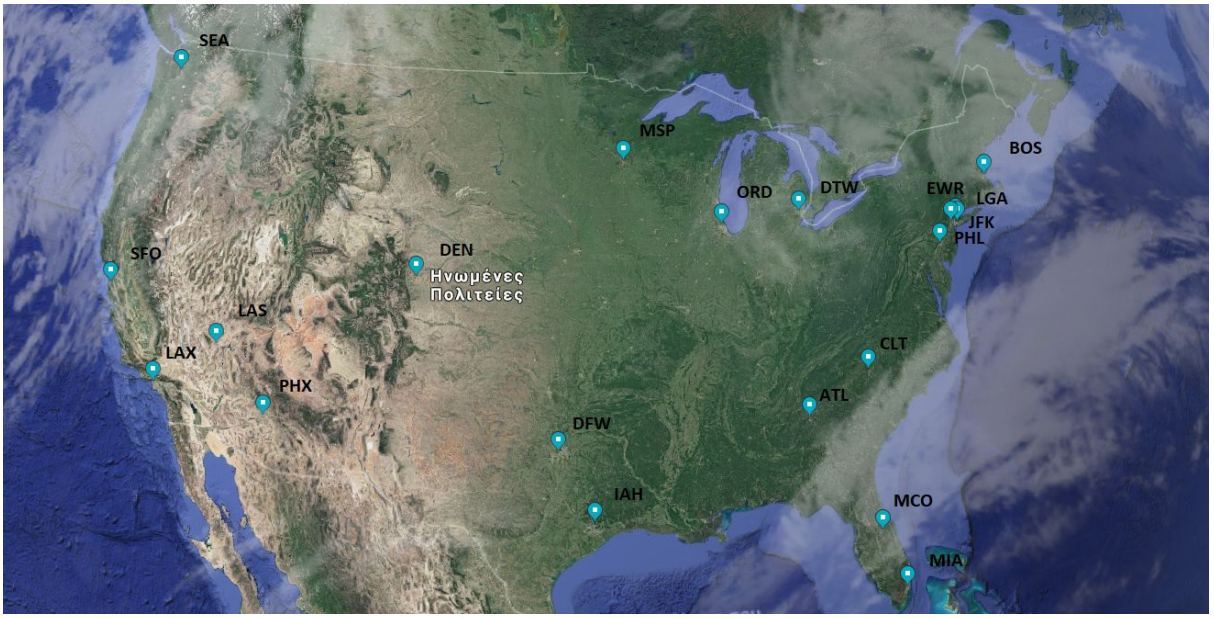
Αναφορά Εργασίας στο μάθημα
"Εξόρυξη Γνώσης από Βάσεις Δεδομένων και τον Παγκόσμιο Ιστό"

Πέτρος - Σώζων Δημητρακόπουλος - Α.Μ 3150034
Σταύρος Μαρκόπουλος - Α.Μ 3150098
Νίκος Καβαλέγας - Α.Μ 6130034

Εισαγωγή

Η εργασία αφορούσε την δημιουργία ενός ταξινομητή (classifier) για την πρόβλεψη της κατηγοριοποίησης πτήσεων μεταξύ 20 αεροδρομίων των ΗΠΑ, σύμφωνα με τον αριθμό των επιβατών τους.

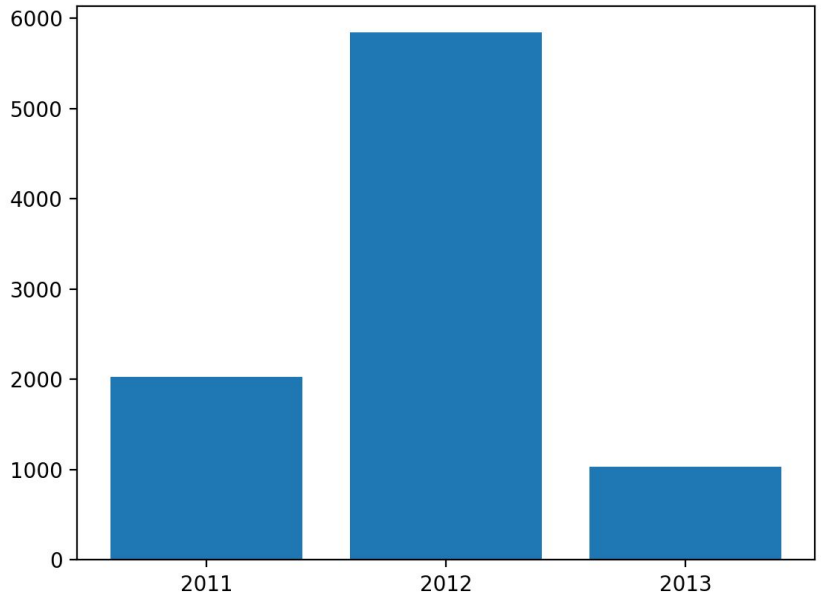


Πιό συγκεκριμένα ζητούνταν για κάθε πτήση να να προβλεφθεί η μεταβλητή **PAX** που σχετίζεται με τον αριθμό των επιβατών και μπορούσε να παίρνει ακέραιες τιμές από 0-7.

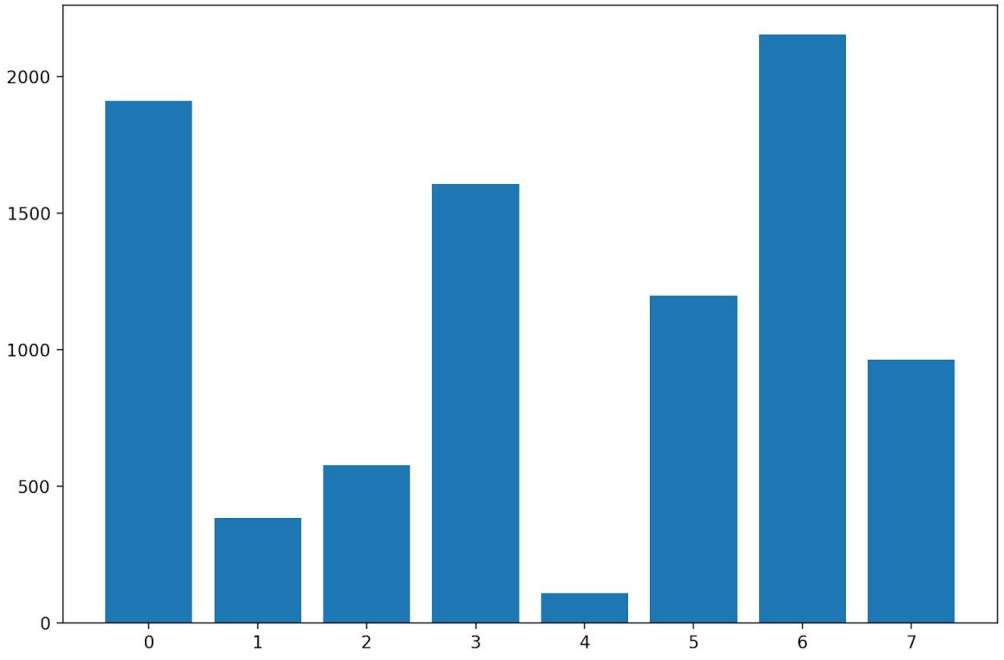
Ιδιαιτερότητες των δεδομένων

Γενικά ο όγκος των δεδομένων ήταν **μικρός** (8899 εγγραφές για εκπαίδευση)**,** γεγονός το οποίο δυσκόλεψε αρκετά την δημιουργία και την ακρίβεια του ταξινομητή**.** Ενδιαφέρον βέβαια έχει το γεγονός ότι από τα features (στοιχεία / χαρακτηριστικά) που δόθηκαν μπορούσαν εύκολα να προκύψουν πολύ χρήσιμα νέα features που μας βοήθησαν αρκετά.

Ξεκινήσαμε προσπαθώντας να “ανακαλύψουμε” ιδιαιτερότητες και ανωμαλίες που μπορεί να έχουν τα δεδομένα οπότε αρχικά “σχεδιάσαμε” (πάντα με την βοήθεια της python και των βιβλιοθηκών pandas και matplotlib) κάποια γραφήματα όπως τα παρακάτω:



(Πλήθος πτήσεων που δίνονται / Έτος)

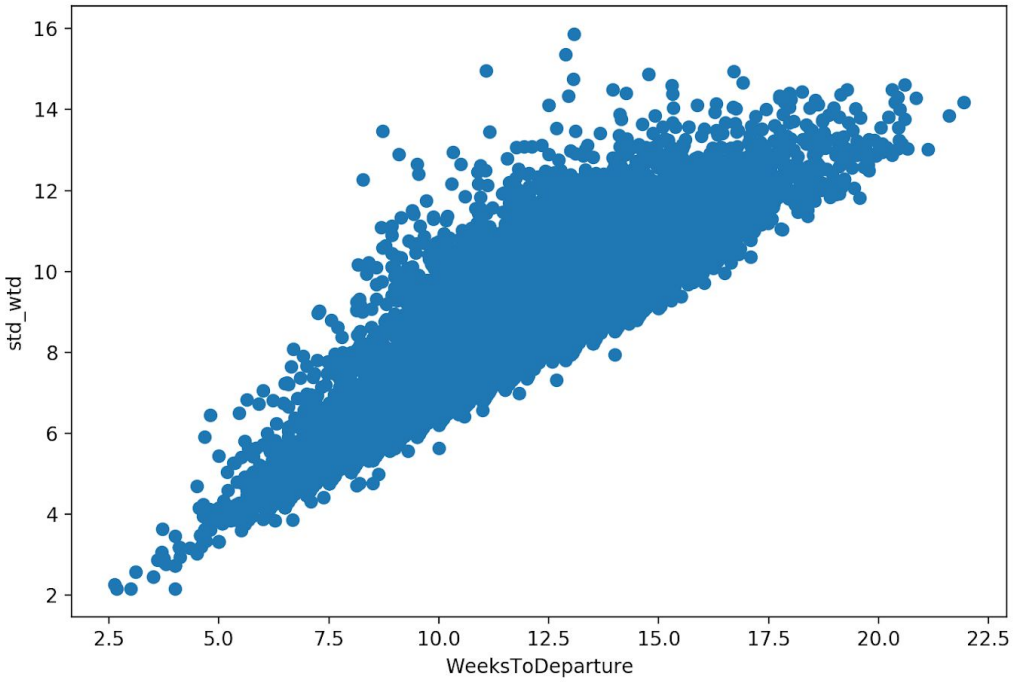


(Κατανομή PAX στις πτήσεις που δίνονται)

Και από τα 2 παραπάνω γραφήματα δίνονται αρκετά σημαντικές πληροφορίες. Από το πρώτο γράφημα καταλαβαίνουμε ότι τα δεδομένα που μας δίνονται δεν είναι ομοιόμορφα κατανεμημένα στα 3 έτη, “ψαχνοντας” περ’ εταίρω γιατί μπορεί να συμβαίνει αυτό βρήκαμε ότι για το **2012** έχουμε δεδομένα **για όλους τους μήνες του έτους**, ενώ για το **2011** έχουμε μόνο **για τους 4 τελευταίους** και για το **2013 μόνο για τους 3 πρώτους**. Από το 2ο γράφημα καταλαβαίνουμε ότι οι **κατηγορίες που σχετίζονται με τον αριθμό των επιβατών** επίσης **δεν είναι ομοιόμορφα κατανεμημένες**.
Διαισθητικά σκεφτήκαμε ότι είναι λογικό οι πτήσεις που γίνονται το Σαββατοκύριακο να έχουν περισσότερους επιβάτες από τις πτήσεις που πραγματοποιούνται στις εργάσιμες ημέρες της εβδομάδα οπότε αυτό μας βοήθησε να δημιουργήσουμε το νέο feature “**weekday**” που είναι μία ακέραια μεταβλητή που δείχνει ποιά μέρα της εβδομάδας πραγματοποιήθηκε η πτήση (θα γίνει εκτενέστερη αναφορά σε όλα τα features που δημιουργήσαμε, κάποια τα οποία βοήθησαν αρκετά, κάποια άλλα λιγότερο και κάποια άλλα καθόλου).
Αυτές είναι κάποιες ενδεικτικές μόνο παρατηρήσεις για την μορφή και τις ιδιαιτερότητες των δεδομένων από τις οποίες ξεκινήσαμε.

Features και Feature Engineering

Από τις στήλες που δίνονται λίγο ως πολύ όλες συμβάλουν με τον τρόπο τους στην επίτευξη καλύτερης κατηγοριοποίησης. Εξαίρεση αποτελεί η στήλη **WeeksToDeparture** η οποία στις περισσότερες (αν όχι σε όλες) τις δοκιμές που κάναμε έριχνε αισθητά την απόδοση του μοντέλου. Αυτό εξηγείται από το γεγονός ότι για πολλές εγγραφές είναι πολύ μεγάλη η τυπική απόκλιση αυτής της τιμής (**std_wtd**). Επιλέξαμε συνειδητά να αφήσουμε αυτά τα 2 features (WeeksToDeparture και std_wtd) εκτός του μοντέλου μας. Το παρακάτω γράφημα απεικονίζει αρκετά καλά γιατί δεν βοηθάει το συγκεκριμένο feature.



Από την στήλη **DateOfDeparture** (day/month/year) εξάγουμε επιπλέον τα εξής:

- "**weekday**" (αριθμός ημέρας της εβδομάδας. 0 για την Δευτέρα , 6 για την Κυριακή)
- "**week**" (αριθμός εβδομάδα μέσα στην οποία πραγματοποιείται η πτήση),
- "**days_to_nye**" (πλήθος ημερών μέχρι την πρωτοχρονιά) αυτό το feature το δημιουργήσαμε σκεπτόμενοι ότι κοντά στις γιορτές των Χριστουγέννων προς το τέλος του έτους , είναι πιθανότερο να ταξιδεύει μεγαλύτερος αριθμός επιβατών. Δίνει μία πολύ καλή εκτίμηση για το πόσο "κοντά" στις γιορτές των Χριστουγέννων βρισκόμαστε ημερολογιακά.
- "**dateDistance**" Η "απόσταση" της ημέρας που έγινε η πτήση από μία σταθερή ημερομηνία που ορίσαμε εμείς (1/1/2010). Αυτό προέκυψε μετά από την σκέψη ότι μπορεί συγκεκριμένα γεγονότα (π.χ καιρικά) να επηρεάζουν τις πτήσεις για λίγες πολύ συγκεκριμένες μέρες (π.χ για ένα 2ήμερο / 3ήμερο) που είναι άσχετα με την εποχή, την ημέρα της εβδομάδας ή τον μήνα.

  Αντίστοιχες προσπάθειες με το days_to_nye κάναμε και για τις γιορτές **Thanksgiving** και **Independence day** χωρίς όμως να διαπιστώσουμε κάποια αισθητή βελτίωση στην απόδοση του συστήματος.

Οι στήλες departure και arrival αναφέρονται στα ίδια αεροδρόμια με τις **CityDeparture** και **CityArrival**. Ωστόσο, οι departure και arrival παίρνουν 20 μοναδικές τιμές, ενώ οι **CityDeparture** και **CityArrival** παίρνουν 19. Αυτό οφείλεται στα αεροδρόμια JFK και LGA που είναι και τα δύο στην Νέα Υόρκη.

Όσον αφορά τις συντεταγμένες των αεροδρομίων αναχώρησης και άφιξης, υπολογίζουμε την μεταξύ τους γεωγραφική απόσταση. Έτσι προέκυψε το feature “**distance**” που φαίνεται να είναι ένα από τα σημαντικότερα για το μοντέλο μας. Το συγκεκριμένο feature το υπολογίζουμε για κάθε εγγραφή με την βοήθεια της βιβλιοθήκης **geopy.**

Φανταστήκαμε ότι κάποια δρομολόγια έχουν περισσότερους επιβάτες από κάποια άλλα, (Π.χ Νέα Υόρκη - Σαν Φρανσίσκο) επομένως στην στήλη “**route”** ενώνουμε για κάθε πτήση το Arrival με το Departure και του αποδίδουμε έναν μοναδικό κωδικό, παίρνοντας 20*20=400 συνδυασμούς δρομολογίων από αεροδρόμιο σε αεροδρόμιο. Εδώ αξίζει να σημειώσουμε ότι (προφανώς) δεν υπάρχουν όλα, και τα 400 δρομολόγια στην πραγματικότητα. Από τους 400 συνδυασμούς που προκύπτουν, πραγματοποιούνται μόνο τα **124 δρομολόγια** (είναι απόλυτα λογικό μιας και δεν γίνονται αεροπορικά δρομολόγια μεταξύ μικρών ή/και κοντινών επαρχιακών πόλεων).

Προεπεξεργασία και Αναπαράσταση Δεδομένων

Αρχικά παρατηρούμε ότι από τα δεδομένα μας δεν λείπει καμία τιμή, επομένως δεν χρειάζεται να μαντέψουμε κάποιες άγνωστες τιμές.

Χρησιμοποιούμε LabelEncoder για να μετατρέψουμε τα αλφαριθμητικά σε αριθμητικές τιμές για να δουλέψουν σωστά οι αλγόριθμοι ταξινόμησης. Έτσι οι κωδικοί των αεροδρομίων μετατρέπονται σε ακέραιους αριθμούς, οι οποίοι όμως δεν έχουν κάποια αριθμητική σχέση μεταξύ τους, είναι απλώς διαφορετικές κατηγορίες αλλα αναπαριστώνται ως αριθμοί.

Για τον λόγο αυτό χρησιμοποιούμε one-hot encoding (ή dummies), δηλαδή μία κατηγορική συνηθως μεταβλητή μετατρέπεται σε όσες στήλες είναι και οι διακριτές τιμές που μπορεί να πάρει, οι οποίες παίρνουν όλες την τιμή 0 για μία εγγραφή ενώ παίρνει 1 μόνο η στήλη που αφορά την τιμή που έχει η συγκεκριμένη εγγραφή στο συγκεκριμένο γνώρισμα.
Αυτή η επιλογή φάνηκε να βοηθάει αρκετά την απόδοση του συστήματός μας.

Επειδή οι αλγόριθμοι που χρησιμοποιούμε δεν υπολογίζουν κάποια απόσταση ανάμεσα στα δεδομένα, κρίναμε σωστό να μην κάνουμε κάποια κανονικοποίηση στις τιμές.

Μέθοδοι που χρησιμοποιήσαμε στο μοντέλο μας

Το μοντέλο που αναπτύξαμε βασίζεται σε μεθόδους **ensembling**, δηλαδή στην χρήση πολλών διαφορετικών αλγορίθμων μηχανικής μάθησης προκειμένου να επιτύχουμε μεγαλύτερη ακρίβεια στην πρόβλεψη της κατηγορίας. Συγκεκριμενα χρησιμοποιήσαμε την κλάση **VotingClassifier** του πακέτου scikit-learn η οποία επιτρέπει την συμμετοχή πολλών αλγορίθμων - ταξινομητών οι οποίοι αφού εκπαιδευτούν πάνω στα δεδομένα εκπαίδευσης “ψηφίζουν” για την κλάση στην οποία πιστεύουν ότι ανήκει η κάθε εγγραφή. Έτσι επιτυγχάνεται ένα αρκετά βελτιωμένο αποτέλεσμα σε σύγκριση με το να χρησιμοποιούσαμε

έναν μόνο αλγόριθμο κατηγοριοποίησης. Οι ταξινομητές που χρησιμοποιήσαμε στο μοντέλο μας είναι κυρίως τα **Random Forests,** το **Gradient Boosting Classifier,** ο **BaggingClassifier με Decision Tree** (πρόκειται και αυτό για μέθοδο ensemble)**,** ο **AdaBoost και** ο **XGBClassifier.**

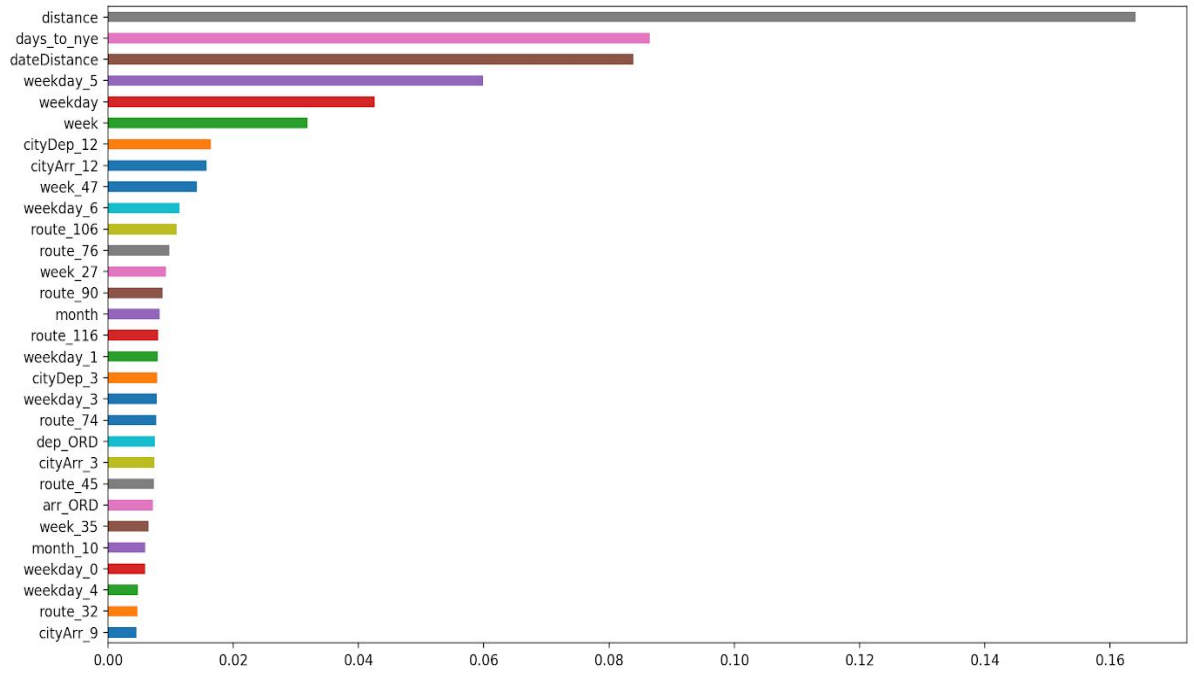
Είναι σημαντικό να αναφέρουμε ότι κάθε ένας από αυτούς τους αλγορίθμους έχει πολλές παραμέτρους τις οποίες έπρεπε να ρυθμίσουμε προκειμένου να καταλήξουμε στο βέλτιστο σύνολο παραμέτρων για τον κάθε ένα. Σε αυτό το σημείο βοήθησε αρκετά η κλάση **GridSearchCV** του πακέτου scikit-learn**.** Πρόκειται για μία κλάση που εκπαιδεύει έναν ταξινομητή πολλές φορές διαλέγοντας κάθε φορά διαφορετικό σύνολο παραμέτρων από ένα υπερσύνολο πολλών διαφορετικών παραμέτρων που του έχουμε αρχικά εισάγει. Αφού εκπαιδεύσει αρκετές φορές το μοντέλο και υπολογίσει την αξιοπιστία του για κάθε σύνολο παραμέτρων, μας ενημερώνει για το βέλτιστο σύνολο παραμέτρων. Δυστυχώς πρόκειται για πολύ χρονοβόρα διαδικασία οπότε δεν μπορέσαμε να την κάνουμε για όλους τους αλγορίθμους και για όλα τα δυνατά υπερσύνολα παραμέτρων.

Feature Importance

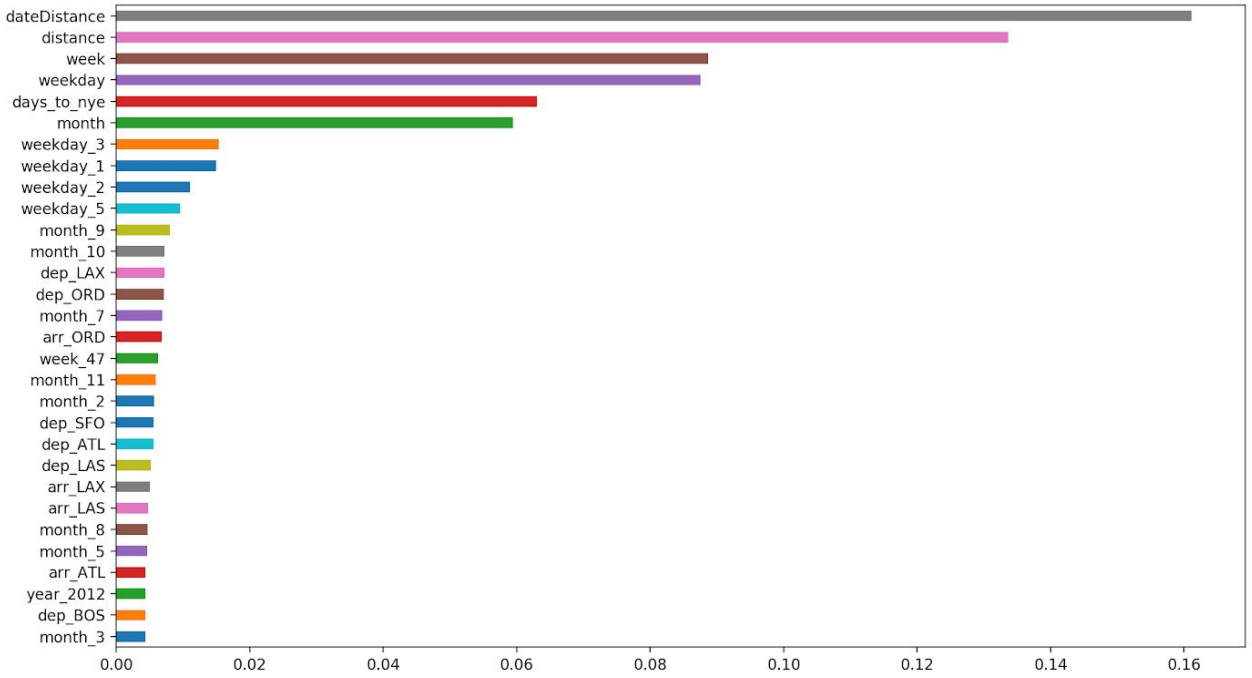
Για πολλούς από τους αλγορίθμους που χρησιμοποιούμε τα χαρακτηριστικά που συμμετέχουν στην εκπαίδευση δεν είναι εξίσου σημαντικά και δεν συνεισφέρουν το ίδιο στην σωστή ταξινόμηση των δεδομένων μας. Παρακάτω φαίνεται ενδεικτικά η σημαντικότητα των **30** πιό σημαντικών χαρακτηριστικών για τους αλγόριθμους των **Random Forests** και **XGB** αντίστοιχα. Παρατηρούμε ότι για διαφορετικούς αλγορίθμους μπορεί να διαφέρει το σύνολο των πιό σημαντικών features. Αυτό είναι πολύ χρήσιμο γιατί σημαίνει ότι με την χρήση πολλών διαφορετικών αλγορίθμων “μπαίνουν” κάθε φορά στο μοντέλο διαφορετικά “είδη” σφαλμάτων οπότε δεν συγκεντρώνεται κάποιο συγκεκριμένο είδος σφάλματος (π.χ από ένα μόνο feature).

Βάρη Κλάσεων - Class Weights

Όπως αναφέραμε και στην πρώτη ενότητα παρατηρήσαμε ότι οι κλάσεις στα δεδομένα εκπαίδευσης που μας δόθηκαν δεν είναι ομοιόμορφα κατανεμημένες. Για αυτό τον λόγο, με την βοήθεια της κλάσης **compute_class_weight** του scikit-learn υπολογίσαμε το “βάρος” της κάθε κλάσης το οποίο ύστερα δώσαμε ως παράμετρο σε κάποιους (όσους το δέχονταν) από τους ταξινομητές μας προκειμένου να τους βοηθήσουμε στην βελτίωση της απόδοσής τους.



(Σημαντικότητα χαρακτηριστικών για τα Random Forests)



(Σημαντικότητα χαρακτηριστικών για τον XGB)

Μέθοδοι / features που δοκιμάσαμε αλλά δεν μας οδήγησαν προς την σωστή κατεύθυνση

Αφιερώσαμε πάρα πολύ χρόνο στην εργασία πολλές φορές χωρίς να έχουμε το προσδοκώμενο αποτέλεσμα. Μερικές φορές μάλιστα οι αλλαγές που πιστεύαμε ότι θα βελτίωναν την απόδοση του συστήματος, την έριχναν ραγδαία οπότε δεν θα μπορούσαμε να μην αναφερθούμε σε αυτό.

Features που δημιουργήσαμε και δεν βοήθησαν

Όπως αναφέραμε και παραπάνω η απόσταση από συγκεκριμένες γιορτές όπως **Independence Day, Haloween, Thanksgiving** κλπ δεν έδωσαν κάτι σημαντικό την επίδοση. Ακόμα το feature "**times"** το οποίο είναι το πλήθος των φορών που απαντάται ένα συγκεκριμένο δρομολόγιο (route) στα δεδομένα μας (η συχνότητα του δρομολογίου) δεν βοήθησε επίσης. Σε ότι αφορά τα features που προέρχονται από το "**DateOfDeparture"**, το feature που δεν βοηθάει το σύστημα είναι το "**Day**" (ημερολογιακή ημέρα του μήνα) και είναι λογικό αφού μάλλον φαίνεται να μην έχει μεγάλη σημασία η ακριβής ημερομηνία μιας πτήσης για τον αριθμό των επιβατών δεδομένου ότι δεν πρόκειται για κάποια "ειδική" ημερομηνία (π.χ πρωτοχρονιά).

Αλγόριθμοι που χρησιμοποιήσαμε και δεν βοήθησαν

Ο αλγόριθμος που χρησιμοποιήσαμε και καμία φορά δεν βοήθησε δραστικά προς θετική κατεύθυνση είναι τα Νευρωνικά Δίκτυα και συγκεκριμένα η κλάση **MLPClassifier** του scikit-learn. Δοκιμάσαμε αρκετούς διαφορετικούς συνδυασμούς κρυφών επιπέδων και παραμέτρων αλλά ποτέ δεν κατάφερε να μας δώσει απόδοση μεγαλύτερη του 30% οπότε και αποφασίσαμε να μην το βάλουμε στο τελικό ensemble που φτιάξαμε. Όσες φορές δοκιμάσαμε να το βάλουμε έριχνε σημαντικά την συνολική απόδοση. Αυτό μάλλον το αποδίδουμε στον **μικρό όγκο των δεδομένων.** Τα δεδομένα φάινονταν πολύ λίγα για να εκπαιδεύσουν σωστά ένα νευρωνικό δίκτυο και οι άλλοι αλγόριθμοι που χρησιμοποιήσαμε φαίνονται μάλλον καταλληλότεροι.

Συνολική Απόδοση του συστήματος

Τελικά η απόδοση του συστήματός μας κυμαίνεται περίπου στο **60% σύμφωνα με την μετρική f1-score (**στο kaggle φαίνεται να "τερματίζουμε" με **59,88%** μιας και πολλές φορές υπήρξαν μεγάλες αποκλίσεις, **μέχρι και 4%** από το score που πετυχαίναμε τοπικά σε αντίστοιχο όγκο δεδομένων**).** Τοπικά αρκετές φορές το σύστημά μας πέτυχε score ακόμα και μεγαλύτερο του 60%.

Χρόνος εκπαίδευσης

Ανάλογα τον υπολογιστή και το configuration από αυτά που είχαμε στην διάθεσή μας, το μοντέλο εκπαιδευόταν σε περίπου 5-7 λεπτά.