ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ – ΑΠΟΘΗΚΕΣ ΚΑΙ ΕΞΟΡΙΞΗ ΔΕΔΟΜΕΝΩΝ

ΑΠΑΛΛΑΚΤΙΚΗ ΕΡΓΑΣΙΑ

Ονοματεπώνυμο: Τασιόπουλος Μανώλης

ΑΜ : Π2015046

Διδάσκοντες : Θέμης Έξαρχος – Κάτια Κερμανίδου

Αριθμός σελίδων : 18

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΕΡΙΛΙΨΗ

Η μηχανική μάθηση αποτελεί μεγάλο κομμάτι της επιστήμης των υπολογιστών, όσο και της καθημερινότητάς μας. Ο τρόπος με τον οποίο ένα ηλεκτρονικός υπολογιστής έχει την ικανότητα να προβλέψει αποτελέσματα για προβλήματα του πραγματικού κόσμου, είναι με τη χρήση αλγορίθμων μηχανικής μάθησης. Στην εργασία αυτή θα αναλύσουμε ένα σετ δεδομένων, θα εφαρμόσουμε σε αυτό διάφορες μεθόδους προ επεξεργασίας και θα εκπαιδεύσουμε τέσσερεις αλγορίθμους ταξινόμησης. Στο τέλος θα συγκρίνουμε τις επιδώσεις τους για το ίδιο σετ δεδομένων, καθώς επίσης και το πόσο σωστά η όχι ταξινομήσανε τα δεδομένα.

ΕΙΣΑΓΩΓΗ

ΕΙΣΑΓΩΓΗ ΣΤΟΝ ΕΡΕΥΝΗΤΙΚΟ ΧΩΡΟ ΤΗΣ ΕΡΓΑΣΙΑΣ

Η μηχανική μάθηση είναι ένα κομμάτι της επιστήμης των υπολογιστών, το οποίο είναι σχετικά καινούργιο στον κλάδο αυτό. Με τον όρο data mining εννοείται η εξόρυξη δεδομένων, δηλαδή η ανακάλυψη γνώσης και συσχετισμών από ένα σετ δεδομένων (data set).

Σκοπός είναι να συλλέξουμε πληροφορίες ή πρότυπα από τα δεδομένα, με τη χρήση αλγορίθμων ομαδοποίησης, κατηγοριοποίησης, στατιστικής, τεχνητής νοημοσύνης και μηχανικής μάθησης. Βασική επιδίωξη της εξόρυξης δεδομένων είναι η πληροφορία που θα εξαχθεί, καθώς και τα πρότυπα που θα προκύψουν να είναι κατανοητά προς τον άνθρωπο. Η απαίτηση αυτή είναι απαραίτητη, προκειμένου ο αλγόριθμος να εξάγει όσο πιο σωστά αποτελέσματα γίνεται.

Μερικές από τις βασικότερες διεργασίες στη εξόρυξη δεδομένων είναι η:

- Ταξινόμηση (Classification)
- Ομαδοποίηση / συσταδοποίηση (Clustering)
- Εύρεση κανόνων συσχέτισης (Association Rule Discovery)
- Εύρεση διαδοχικών / σειριακών προτύπων (Sequential Pattern Discovery)
- Παλινδρόμηση (Regression)
- Ανίχνευση Αποκλίσεων (Deviation Detection)

ΒΑΣΙΚΕΣ ΠΡΟΣΕΓΓΙΣΕΙΣ ΕΠΙΛΥΣΗΣ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Το αντικείμενο που πραγματεύεται η εργασία αυτή έχει να κάνει συγκεκριμένα με αλγορίθμους ταξινόμησης, καθώς και με την σύγκριση των αποδόσεών τους. Για την υλοποίηση του σκοπού αυτού χρησιμοποιήθηκαν τέσσερεις αλγόριθμοι ταξινόμησης με το ίδιο σετ δεδομένων.

Το σετ δεδομένων (data set) που επιλέχθηκε περιέχει δεδομένα πελατών από διάφορες τράπεζες. Ο βασικός σκοπός ήταν η ταξινόμηση των πελατών με βάση την παραμονή τους ή όχι στην εκάστοτε τράπεζα.

Χρησιμοποιήθηκαν διάφορα εργαλεία για την επίτευξη του σκοπού αυτού, όπως το jupyter notebook και διάφορες βιβλιοθήκες της γλώσσας python.

Για την επίτευξη του σκοπού αυτού ακολουθήθηκαν οι παρακάτω μεθοδολογίες:

1. Εισαγωγή του σετ δεδομένων (.csv)
2. Ανάλυση του σετ δεδομένων και κατανόηση των χαρακτηριστικών τους
3. Προ επεξεργασία των δεδομένων
4. Προσαρμογή των δεδομένων για τους αλγορίθμους
5. Εκτέλεση των αλγορίθμων ταξινόμησης με το σετ δεδομένων
6. Σύγκριση των αποτελεσμάτων του καθενός

ΣΥΝΕΙΣΦΟΡΑ ΤΗΣ ΕΡΓΑΣΙΑΣ

Με την εργασία αυτή θα μελετήσουμε τέσσερεις αλγόριθμους ταξινόμησης. Εκτελώντας τις τεχνικές μάθησης που προσφέρονται από τη βιβλιοθήκη sklearn της γλώσσας προγραμματισμού python, θα μελετήσουμε την αποδοτικότητά τους. Σκοπός μας είναι να βρούμε ποιος από τους τέσσερεις αλγορίθμους, προσφέρει τη μεγαλύτερη απόδοση και παράλληλα να εντοπίσουμε σε ποια σημεία υστερούν οι άλλοι.

Ο τρόπος με τον οποίο θα καταλήξουμε στον αλγόριθμο με τη μεγαλύτερη απόδοση είναι με τη χρήση της αποδοτικότητας αλλά και με τη χρήση πινάκων σύγχυσης (confusion matrix) για την εύρεση των λανθασμένων ταξινομήσεων.

ΑΛΓΟΡΙΘΜΟΙ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ

Στο σημείο αυτό θα αναλυθούν οι αλγόριθμοι ταξινόμησης που χρησιμοποιήθηκαν για τη διεξαγωγή των αποτελεσμάτων. Κάθε αλγόριθμος που επιλέχθηκε εκτελέσθηκε με το ίδιο σετ δεδομένων προκειμένου να γίνει αντικειμενική μελέτη.

SVM (SUPPORT VECTOR MACHINE) [1]

Στη μηχανική μάθηση, τα support vector machines (SVMs) εποπτεύονται μοντέλα μάθησης με συνδεδεμένους αλγορίθμους μάθησης οι οποίοι αναλύουν δεδομένα και αναγνωρίζουν πρότυπα, που χρησιμοποιούνται για την ταξινόμηση και την ανάλυση παλινδρόμησης.

Η βασική SVM παίρνει ένα σύνολο δεδομένων εισόδου και προβλέπει, για κάθε δεδομένη είσοδο, ποια από τις δύο πιθανές καταστάσεις αποτελεί την έξοδο, καθιστώντας το ένα

μηγραμμικό δυαδικό ταξινομητή. Λαμβάνοντας υπόψη ένα σύνολο παραδειγμάτων εκπαίδευσης, ένας αλγόριθμος SVM χτίζει ένα μοντέλο που εκχωρεί νέα δείγματα σε μία από τις καταστάσεις.

Ένα μοντέλο SVM είναι μια αναπαράσταση των δειγμάτων ως σημεία στο χώρο, τα οποία χαρτογραφούνται έτσι ώστε τα δείγματα διαφορετικών κατηγοριών να μπορούν να διαχωρίζονται όσο το δυνατόν καλύτερα. Εκτός από την εκτέλεση γραμμικής ταξινόμησης, τα SVMs μπορούν να εκτελέσουν το ίδιο αποτελεσματικά και την μη-γραμμική ταξινόμηση.

KNN (K-NEAREST NEIGHBOR) [2]

Ο αλγόριθμος k-nearest-neighbors είναι μια μη παραμετρική μέθοδος που χρησιμοποιείται για ταξινόμηση. Οι είσοδοι αποτελούνται από τα k κοντινότερα παραδείγματα στο χώρο. Η έξοδος, ωστόσο, εξαρτάται από την χρήση του k-NN, αν γίνεται για ταξινόμηση.

Στην ταξινόμηση, η έξοδος του αλγόριθμου είναι μέλος μιας κατηγορίας. Ένα αντικείμενο κατηγοριοποιείται από την πλειοψηφία των ψήφων των γειτόνων του, με το αντικείμενο να ανατίθεται στην κλάση που είναι περισσότερο κοινή στους k κοντινότερους γείτονες (με το k να είναι ένας θετικός ακέραιος, συνήθως μικρός αριθμός). Αν k = 1, τότε το αντικείμενο ανατίθεται, απλά, στην κλάση του πιο κοντινού γείτονα.

RANDOM FOREST [3]

Τα random forests ή random decision forests είναι μια συνδυαστική μέθοδος μάθησης για ταξινόμηση, τα οποία λειτουργούν με την δημιουργία ενός πλήθους decision trees. Κατά την φάση της εκπαίδευσης παράγουν την κλάση που εμφανίζεται πιο συχνά στην ταξινόμηση του κάθε δέντρου. Τα random forests αντιμετωπίζουν και διορθώνουν σε μεγάλο βαθμό το πρόβλημα των decision trees κατά το οποίο υπάρχει overfitting του σετ εκπαίδευσης.

NAIVE BAYES [4]

Ο Naive Bayes ταξινομητής προσφέρει μια απλή προσέγγιση στα προβλήματα, όπου στόχος μας είναι να προβλέψουμε επακριβώς την κατηγορία-κλάση των στιγμιότυπων δοκιμής. Για την επίτευξη της πρόβλεψης χρησιμοποιεί ταξινομημένα στιγμιότυπα εκπαίδευσης που περιλαμβάνουν την πληροφορία της κλάσης που ανήκουν. Βασίζεται σε δυο σημαντικές υποθέσεις.

- Ότι κάθε χαρακτηριστικό των στιγμιότυπων είναι στοχαστικά ανεξάρτητα των υπολοίπων, δεδομένης της κλάσης
- Ότι δεν υπάρχουν άλλα κρυφά χαρακτηριστικά που να επηρεάζουν την διαδικασία της πρόβλεψης.

ΜΕΘΟΔΟΛΟΓΙΚΗ ΔΙΑΔΙΚΑΣΙΑ

Σε αυτό το κεφάλαιο θα αναλυθεί το σετ δεδομένων που χρησιμοποιήθηκαν. Θα αναλυθούν δηλαδή τα χαρακτηριστικά που το αποτελούν, ποιες είναι οι συσχετίσεις που δημιουργούνται μεταξύ τους, καθώς και η προ επεξεργασία που εφαρμόστηκε. Επιπλέον θα αναλυθούν και οι αλγόριθμοι ταξινόμησης μαζί με την εφαρμογή τους στο σετ δεδομένων. Τέλος θα παρουσιασθούν τα αποτελέσματα που προέκυψαν.

ΠΕΡΙΓΡΑΦΗ ΔΕΔΟΜΕΝΩΝ

Όπως αναφέρθηκε και στην εισαγωγή, το σετ δεδομένων που χρησιμοποιήθηκε περιέχει στοιχεία πελατών από διάφορες τράπεζες. Στόχος είναι να εντοπίσουμε εάν ένας πελάτης αποχωρεί από την τράπεζα (κλείνοντας το λογαριασμό του), ή εάν συνεχίζει να χρησιμοποιεί τις υπηρεσίες της.

Το data set μας είναι ένα αρχείο τύπου (.csv), οπότε μπορούμε πολύ εύκολα να το επεξεργαστούμε με τη χρήση της βιβλιοθήκης pandas της python. Περιέχει 10.000 γραμμές ,από τις οποίες η κάθε μία περιέχει 14 χαρακτηριστικά που αντιστοιχούν σε κάθε πελάτη από μία τράπεζα.

Η είσοδος των δεδομένων γίνεται με την παρακάτω μέθοδο

```
import numpy as np #linear agebra
import pandas as pd #data processing for our CSV file
import warnings

warnings.filterwarnings("ignore")

data = pd.read_csv("bank_data.csv")
```

Πρώτού προβούμε στην προ επεξεργασία, θα χρειαστεί πρώτα να κατανοήσουμε τα δεδομένα του data set

RowNumb	Customer	Surname	CreditSco	Geograph	Gender	Age	Tenure	Balance	NumOfPr	HasCrCard	IsActiveM	Estimated	Exited
1	15634602	Hargrave	619	France	Female	42	2	0	1	1	1	101348.9	1
2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.6	0
3	15619304	Onio	502	France	Female	42	8	159660.8	3	1	0	113931.6	1
4	15701354	Boni	699	France	Female	39	1	0	2	0	0	93826.63	0
5	15737888	Mitchell	850	Spain	Female	43	2	125510.8	1	1	1	79084.1	0
6	15574012	Chu	645	Spain	Male	44	8	113755.8	2	1	0	149756.7	1

Το δεδομένο που θέλουμε να εξετάσουμε ονομάζεται exited αποτελεί ένα binary variable (δυαδική μεταβλητή). Πρόκειται δηλαδή για μία μεταβλητή της οποίας οι τιμές είναι αληθείς (true) ή ψευδείς (false). Αλλιώς κυμαίνονται από τις τιμές 0 – 1.

ΠΕΡΙΓΡΑΦΗ ΠΡΟΕΠΕΞΕΡΓΑΣΙΑΣ

Προκειμένου οι αλγόριθμοι να έχουν μεγαλύτερη αποδοτικότητα, θα πρέπει το data set να υποστεί κάποιες επεξεργασίες. Η διαδικασία αυτή ονομάζεται προ επεξεργασία (preprocessing) και εφαρμόζεται πριν τη χρήση κάποιου αλγόριθμου. Στη συνέχεια θα πρέπει να διαχωρίσουμε το data set σε Ζητούμενα (y) και δεδομένα (x).

ΔΗΜΙΟΥΡΓΙΑ HEAT-MAP ΔΕΔΟΜΕΝΩΝ

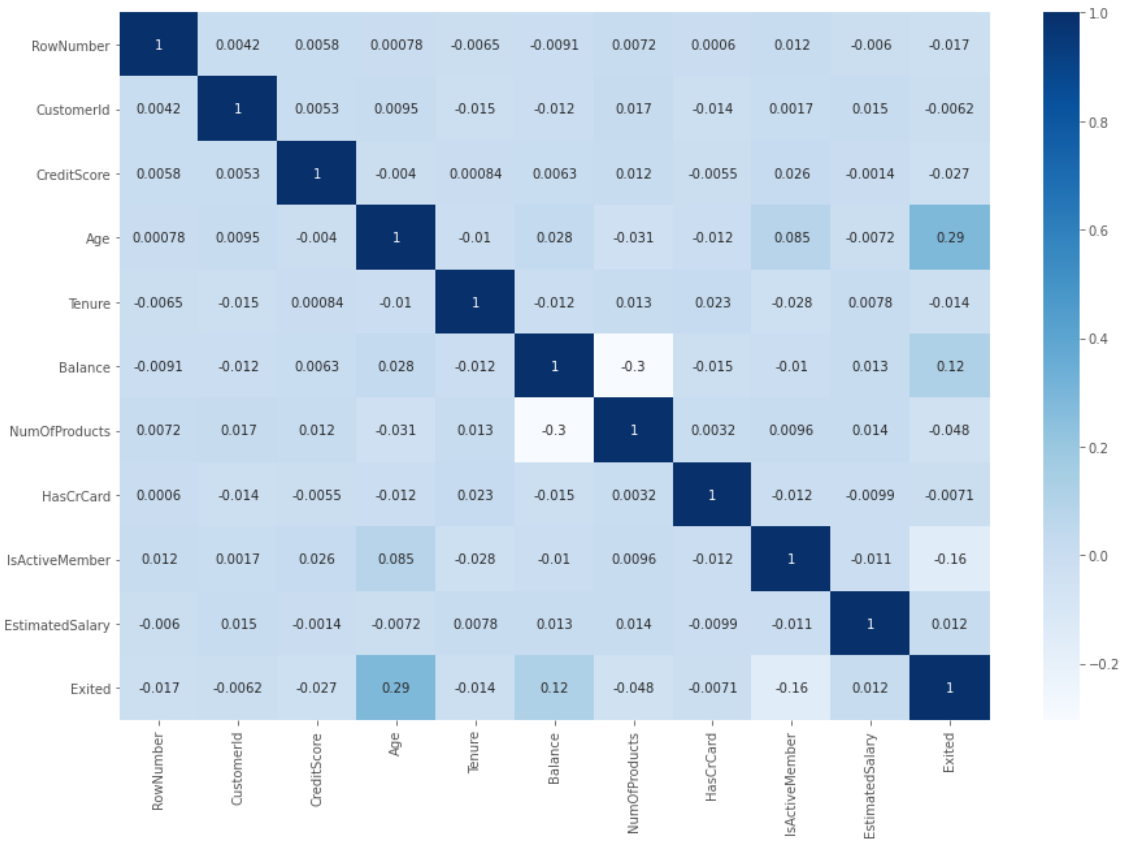
Προτού προχωρήσουμε στις μεθόδους προ επεξεργασίας, μία καλή τεχνική για να εντοπίσουμε διάφορες συσχετίσεις στα δεδομένα είναι να δημιουργήσουμε ένα heatmap. Πρόκειται για έναν πίνακα ο οποίος οπτικοποιεί το πόσο ένα δεδομένο συσχετίζεται με ένα άλλο.

Για τη δημιουργία του χρησιμοποιήθηκαν οι βιβλιοθήκες της python matplotlib και seaborn. Τα χαρακτηριστικά που διακρίνονται στον παρακάτω πίνακα είναι τα ακόλουθα:

- Το heatmap εμφανίζει μερικές χρήσιμες συσχετίσεις, όπως για παράδειγμα η στήλη 'Age' συσχετίζεται θετικά με τη στήλη (positive related) 'Exited'

- Τα στοιχεία των οποίων το χρώμα είναι πιο σκούρο συσχετίζονται περισσότερο μεταξύ του

- Αντίθετα τα στοιχεία με πιο ανοιχτό χρώμα δε συσχετίζονται τόσο πολύ. Για παράδειγμα η στήλη 'NumOfProducts' δε συσχετίζεται με τη στήλη 'Balance'



ΑΧΡΕΙΑΣΤΕΣ ΣΤΗΛΕΣ ΚΑΙ ΔΙΑΓΡΑΦΗ

Για την επίλυση του προβλήματος, θα διαγράψουμε κάποια χαρακτηριστικά που δε χρειάζονται. Αυτά είναι τα παρακάτω:

- RowNumber
- CustomerId
- Surname
- Geography

Ο λόγος για τον οποίο τα διαγράφουμε, είναι διότι περιέχουν πληροφορίες οι οποίες δε συσχετίζονται με το ζητούμενό μας που είναι το στοιχείο Exited.

```
data.drop(['RowNumber', 'CustomerId', 'Surname', 'Geography'], axis = 1, inplace = True)
```

axis = 1: επιλογή του άξονα με τις στήλες

inplace = True: Διαγραφή χωρίς εκ νέου ανάθεσης

ΑΛΛΑΓΗ ΤΩΝ ΤΙΜΩΝ ΣΕ 0 ΚΑΙ 1

Στο data set το χαρακτηριστικό του γένους περιέχει αλφαριθμητικά. Για να το απλοποιήσουμε αυτό θα μετατρέψουμε τα Female/Male σε 0/1

```
data.Gender = [1 if each == 'Male' else 0 for each in data.Gender]
```

```
data.head()
```

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
0	619	0	42	2	0.00	1	1	1	101348.88	1
1	608	0	41	1	83807.86	1	0	1	112542.58	0
2	502	0	42	8	159660.80	3	1	0	113931.57	1
3	699	0	39	1	0.00	2	0	0	93826.63	0
4	850	0	43	2	125510.82	1	1	1	79084.10	0

Όπως διακρίνουμε τα δεόμενα στη στήλη Gender έχουν γίνει 0.
Άρα αυτό σημαίνει ότι οι πέντε πρώτες γραμμές περιέχουν πελάτες γένους θηλυκού.

ΔΙΑΧΩΡΙΣΜΟΣ ΔΕΔΟΜΕΝΩΝ

Σε αυτό το σημείο θα διαχωρίσουμε το ζητούμενό μας από το data set, έτσι ώστε να δημιουργήσουμε το τις μεταβλητές x_data και y όπου περιέχουν τα δεδομένα και τις απαντήσεις αντίστοιχα

```
y = data.Exited.values
```

```
x_data = data.drop(['Exited'], axis = 1)
```

Επιπλέον θα ήταν καλό να οπτικοποιήσουμε τον πίνακα y έτσι ώστε να έχουμε καλύτερη άποψη πάνω γύρω από το ζητούμενο. Όπως και με το hetamap, θα οπτικοποιήσουμε τον πίνακα με τη χρήση των βιβλιοθηκών matplotlib και seaborn.

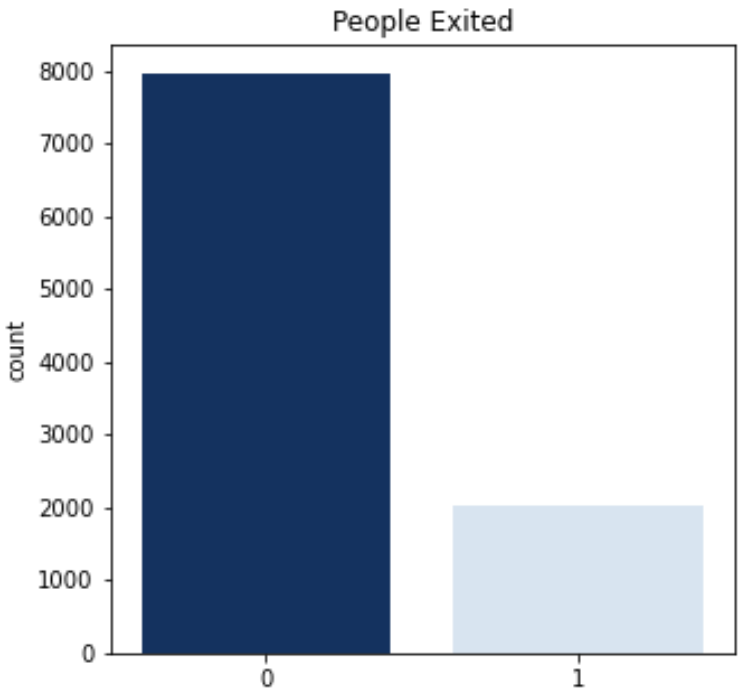
ΚΑΝΟΝΙΚΟΠΟΙΗΣΗ (NORMALIZATION)

Εάν εμφανίσουμε τον πίνακα x_data θα με τη μέθοδο describe() της βιβλιοθήκης pandas, εντοπίζουμε πως υπάρχουν αριθμοί μεγάλης κλίμακας στο x_data.

```
x_data.describe()
```

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000
mean	650.528800	0.545700	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881
std	96.653299	0.497932	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818
min	350.000000	0.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000
25%	584.000000	0.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000
50%	652.000000	1.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000
75%	718.000000	1.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500
max	850.000000	1.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000

Θα πρέπει να κανονικοποιήσουμε τις τιμές αυτές σε 0 και 1. Η κανονικοποίηση που εφαρμόστηκε στα δεδομένα έγινε με τον παρακάτω τύπο:



```
x = (x_data - np.min(x_data)) / (np.max(x_data) - np.min(x_data))
x.head()
```

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	0.538	0.0	0.324324	0.2	0.000000	0.000000	1.0	1.0	0.506735
1	0.516	0.0	0.310811	0.1	0.334031	0.000000	0.0	1.0	0.562709
2	0.304	0.0	0.324324	0.8	0.636357	0.666667	1.0	0.0	0.569654
3	0.698	0.0	0.283784	0.1	0.000000	0.333333	0.0	0.0	0.469120
4	1.000	0.0	0.337838	0.2	0.500246	0.000000	1.0	1.0	0.395400

Όπως βλέπουμε οι τιμές έχουν συμπιεστεί σε τιμές μεταξύ του 0 και του 1. Αυτό το κάναμε προκειμένου η υπολογιστική ισχύς που θα χρειαστεί για την ανάλυση του από τους αλγορίθμους να είναι μικρότερη.

ΠΡΟΣΑΡΜΟΓΗ ΤΩΝ ΔΕΔΟΜΕΝΩΝ ΓΙΑ ΕΚΠΑΔΕΥΣΗ

Σε αυτό το σημείο Θα διαιρέσουμε τα δεδομένα σε (x_train y_train) και σε (x_test y_test). Στη συνέχεια θα εκπαιδεύουμε τους αλγορίθμους με τα δεδομένα αυτά και θα προσπαθήσουν να προβλέψουν τα test δεδομένα. Τέλος θα συγκρίνουμε τις προβλέψεις (y_pred) με τα test δεδομένα.

Ο τρόπος με τον οποίο θα χωρίσουμε τα δεδομένα είναι με την παρακάτω συνάρτηση.

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.10, random_state=7)
```

- x, y: Το data set
- test_size = 0.10: Μέγεθος των test μεταβλητών
- random_state = 7: Τυχαίος αριθμός ως seed

Με τον τρόπο αυτό χωρίσαμε τα δεδομένα μας όπως απεικονίζονται παρακάτω.

```
x_train shape:  (9000, 9)
y_train shape:  (9000,)
x_test shape:  (1000, 9)
y_test shape:  (1000,)
```

ΠΕΙΡΑΜΑΤΑ ΜΑΘΗΣΗΣ

Στην ενότητα αυτή θα αναλύσουμε την εκτέλεση του κάθε αλγορίθμου, καθώς και την κλήση τους από τη βιβλιοθήκη sklearn της python. Στις παρακάτω εικόνες εμφανίζονται τόσο τα ορίσματα που δέχεται ο κάθε αλγόριθμος, όσο και οι τρόποι με τους οποίους το μοντέλο εκπαιδεύεται και στη συνέχεια προβλέπει. Η εκπαίδευση γίνεται με τα δεδομένα που περιέχουν το x_train και y_train, ενώ η πρόβλεψη γίνεται στα δεδομένα του x_test.

SUPPORT VECTOR MACHINE (SVM) ΑΛΓΟΡΙΘΜΟΣ

```
from sklearn.svm import SVC

svm = SVC(random_state = 2) #SVM model with random_state = 2

svm.fit(x_train, y_train) #Train the model

y_pred_svm = svm.predict(x_test) #Predict using x_test
```

K-NEAREST NEIGHBOR (KNN) ΑΛΓΟΡΙΘΜΟΣ

```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors = 13)#KNN model with k = 13

knn.fit(x_train, y_train) #Training the model

y_pred_knn = knn.predict(x_test) #Predict using x_test
```

RANDOM FOREST ΑΛΓΟΡΙΘΜΟΣ

```
from sklearn.ensemble import RandomForestClassifier

rand_for = RandomForestClassifier(n_estimators=100, random_state=3) #random forest model

rand_for.fit(x_train, y_train) #Training the model

y_pred_rand_for = rand_for.predict(x_test) #Predict using x_test
```

NAIVE BAYES ΑΛΟΡΙΘΜΟΣ

```
from sklearn.naive_bayes import GaussianNB

naiv_b = GaussianNB() #Naive Bayes model

naiv_b.fit(x_train, y_train) #Training the model

y_pred_naiv_b = naiv_b.predict(x_test) #Predict using x_test
```

ΑΝΑΛΥΣΗ ΚΑΙ ΑΞΙΟΛΟΓΗΣΗ

Τέλος θα συγκρίνουμε τα μοντέλα που εκπαιδεύσαμε. Για το κάθε μοντέλο με την ολοκλήρωση της εκπαίδευσής του, εξάγομε το ποσοστό ακρίβειας που είχε στο συγκεκριμένο σετ δεδομένων. Επιπλέον δημιουργήσαμε έναν confusion matrix για τον κάθε αλγόριθμο, ώστε να παρατηρήσουμε σε ποια σημεία έκανε λάθος ταξινόμηση.

ΣΥΓΚΡΙΣΗ ΤΩΝ ΑΠΟΔΟΣΕΩΝ

Στον παρακάτω πίνακα εμφανίζονται οι αποδώσεις που είχαν οι αλγόριθμοι:

```
svm_accuracy              0.867
random_forest_accuracy    0.862
knn_accuracy              0.839
naive_bayes_accuracy      0.836
dtype: float64
```

Όπως μπορούμε να δούμε και παραπάνω ο αλγόριθμος SVM προβλέπει με απόδοση 0.87

Μία ακόμη μέθοδος απεικόνισης των αποτελεσμάτων που πραγματοποιήθηκε είναι η σύγκριση των αποτελεσμάτων που προέβλεψε ο κάθε αλγόριθμος, με τα πραγματικά αποτελέσματα του πίνακα y_test, για κάθε στοιχείο του πίνακα αυτού.

```
y_predict_test_data = {'y_test': y_test,
                       'svm_prediction': y_pred_svm,
                       'knn_prediction': y_pred_knn,
                       'random_forest_prediction': y_pred_rand_for,
                       'naive_bayes_prediction': y_pred_naiv_b}

array_predictions_test = pd.DataFrame(data = y_predict_test_data)
array_predictions_test.T
```

Με άλλα λόγια θα κατασκευάσουμε ένα DataFrame με τη χρήση της βιβλιοθήκης pandas, για την καλύτερη προβολή των αποτελεσμάτων.

	0	1	2	3	4	5	6	7	8	9	...	990	991	992	993	994	995	996	997	998	999
y_test	1	0	0	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	0	0
svm_prediction	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	0	0
knn_prediction	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
random_forest_prediction	1	0	0	1	0	0	0	0	0	1	...	0	0	0	0	0	0	0	1	0	0
naive_bayes_prediction	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	0	0

5 rows × 1000 columns

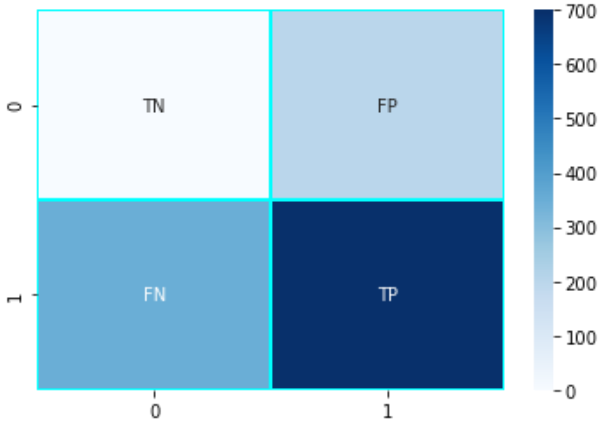
Ένα παράδειγμα με το οποίο μπορούμε να διατυπώσουμε τον παραπάνω πίνακα είναι το εξής. Για το στοιχείο μηδέν η πραγματική τιμή του y_test είναι 1, δηλαδή ο πελάτης θα αποχωρήσει από την συγκεκριμένη τράπεζα. Για το ίδιο στοιχείο, όλοι οι αλγόριθμοι επέστρεψαν την τιμή μηδέν, εκτός από τον random forest, πράγμα που τον καθιστά τον μόνο σωστό για το στοιχείο αυτό.

ΣΥΓΚΡΙΣΗ ΤΩΝ CONFUSION MATRIXES

Τέλος για την κατανόηση του ποιος αλγόριθμος κατάφερε να προβλέψει σωστά ή όχι τις τιμές του y_test, συγκρίνουμε τους Confusion Matrixes μεταξύ τους. Κάθε πίνακας περιέχει το παρακάτω layout:

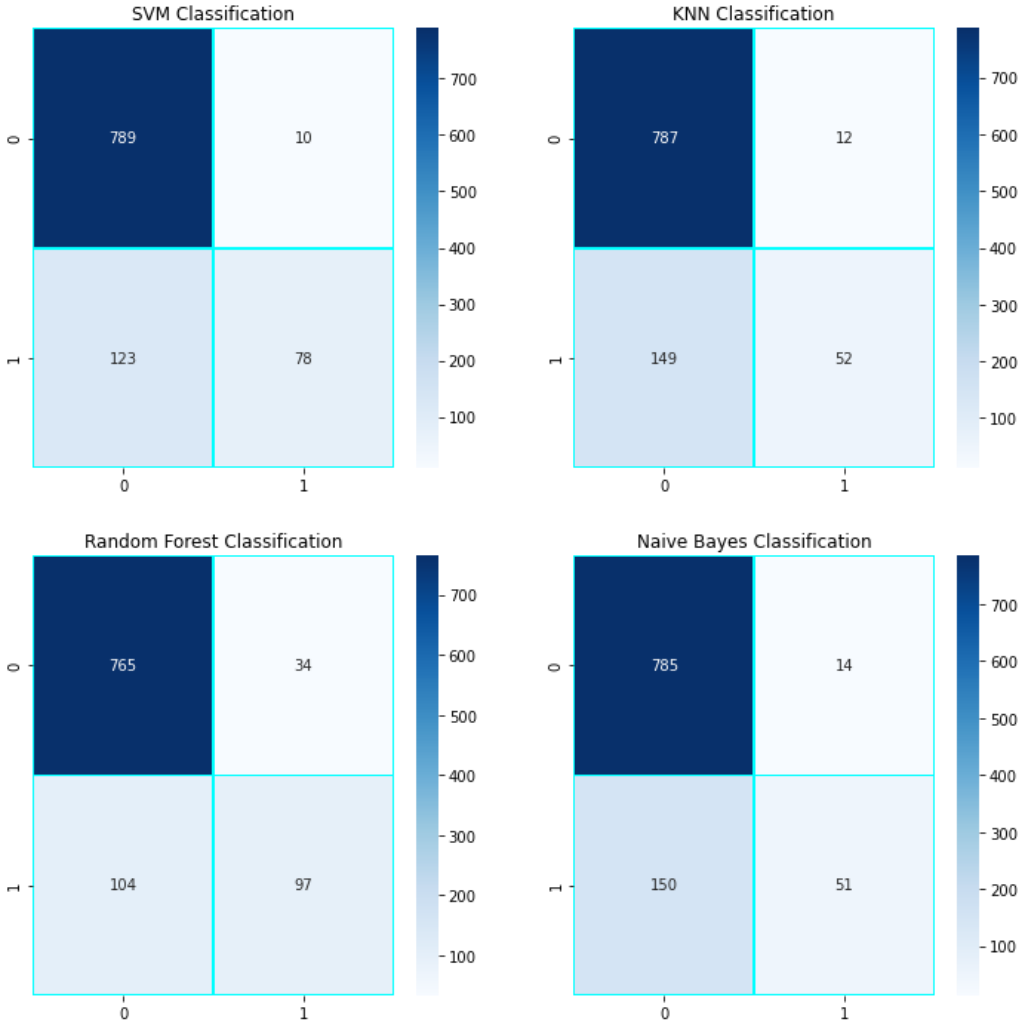
Σωστές προβλέψεις

- TN (True Negatives): Τιμές που είναι αρνητικές και ο αλγόριθμος τις ταξινόμησε σωστά
- TP (True Positives): Τιμές που είναι θετικές και ο αλγόριθμος τις ταξινόμησε σωστά



Λανθασμένες προβλέψεις

- FN (False Negative): Τιμές που είναι αρνητικές και ο αλγόριθμος δεν ταξινόμησε σωστά
- FP (False Positive): Τιμές που είναι θετικές και ο αλγόριθμος δεν ταξινόμησε σωστά



Στους παραπάνω πίνακες μπορούμε να διακρίνουμε τις εξής πληροφορίες:

- Αλγόριθμος SVM
  - TN = 789
  - TP = 78
- Αλγόριθμος KNN
  - TN = 787
  - TP = 52
- Αλγόριθμος Random Forest
  - TN = 765
  - TP = 97
- Αλγόριθμος Naïve Bayes
  - TN = 785
  - TP = 51

ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΡΟΤΑΣΕΙΣ ΓΙΑ ΜΕΛΛΟΝΤΙΚΕΣ ΒΕΛΤΙΩΣΕΙΣ

Ολοκληρώνοντας την εργασία αυτή μας δόθηκε η ευκαιρία να ασχοληθούμε με τις μεθόδους μηχανικής, όπως επίσης και την προ επεξεργασία των δεδομένων. Έγιναν επίσης αντιληπτοί αρκετοί από τους τρόπους εύρεσης συσχετίσεων στο σετ των δεδομένων, για την καλύτερη προ επεξεργασία αλλά και την κατανόηση των χαρακτηριστικών. Επιπλέον είδαμε τους τρόπους σύγκρισης αλγορίθμων μάθησης και με ποιόν τρόπο θα μπορούσε να εφαρμοστεί σε ένα πραγματικό πρόβλημα.

Μερικές μελλοντικές βελτιώσεις θα ήταν η προσθήκη περισσότερων αλγορίθμων. Επιπλέον θα μπορούσε το απλό πρόγραμμα αυτό να υλοποιηθεί με τέτοιο τρόπο ώστε να δέχεται αρκετά data sets και στη συνέχεια να εμφανίζει στο χρήστη ποιος είναι ο καλύτερος αλγόριθμος για το εκάστοτε data set.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Suykens, J. A. K., & Vandewalle, J. (1999). Neural Processing Letters, 9(3), 293–300.

[2] Dudani, S. A. (1976). The Distance-Weighted k-Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics, SMC-6(4), 325–327

[3] Pal, M. (2005). Random forest classifier for remote sensing classification. International Journal of Remote Sensing, 26(1), 217–222.

[4] Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In Machine Learning: ECML-98 (pp. 4–15).

[5] Jupyter Notebook - https://jupyter.org/

[6] Sklearn - https://scikit-learn.org/stable/