



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

Σχολή Θετικών Επιστημών και Τεχνολογίας

**Μεταπτυχιακό Πρόγραμμα Εξειδίκευσης στα
Πληροφοριακά Συστήματα**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

**Τεχνολογίες εξόρυξης επιχειρησιακών δεδομένων με χρήση
προγραμματιστικών εργαλείων**

Οικονομίδης Κωνσταντίνος

Επιβλέπουσα: Πάσχου Μερσίνη

ΠΑΤΡΑ
ΙΟΥΛΙΟΣ, 2021

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή («συγγραφέας/δημιουργός») που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.



Τεχνολογίες εξόρυξης επιχειρησιακών δεδομένων με χρήση προγραμματιστικών εργαλείων

Κωνσταντίνος Οικονομίδης

Επιτροπή Επίβλεψης Διπλωματικής Εργασίας

Επιβλέπουσα:
Μερσίνη Πάσχου
Μέλος ΣΕΠ, Ελληνικό Ανοικτό
Πανεπιστήμιο

Συν-Επιβλέπων Καθηγητής:
Δημήτριος Καραπιτέρης
Μεταδιδακτορικός
ερευνητής ΕΑΠ

Αθήνα, Ιούλιος, 2021

*Θα ήθελα να εκφράσω τις ευχαριστίες μου στην επιβλέπουσα της εργασίας, καθηγήτρια κυρία
Μερσίνη Πάσχου, καθώς και τον καθηγητή κύριο Σακκόπουλο Ευάγγελο, για τη σημαντική
καθοδήγηση, υποστήριξη αλλά και για την ενθάρρυνση τους ώστε να πραγματοποιηθεί αυτή η
εργασία.*

*Επίσης, θέλω να ευχαριστήσω την οικογένεια μου για την υποστήριξη τους σε όλη τη διάρκεια
των σπουδών μου στο ΕΑΠ.*

Περίληψη

Η παρούσα διπλωματική ασχολείται με την έννοια της εξόρυξης δεδομένων με προγραμματιστικά εργαλεία. Αρχικά παρουσιάζεται το θεωρητικό υπόβαθρό της, οι εφαρμογές της σε διάφορους τομείς της επιστήμης και της οικονομίας καθώς και η ευρεία χρήση που έχει στον επαγγελματικό αθλητισμό. Παρουσιάζονται τα προγραμματιστικά εργαλεία που χρησιμοποιούνται, όπως ο Microsoft SQL Server, το Machine Learning Services και το Azure Data Studio της Microsoft και επιχειρείται μια θεωρητική περιγραφή τριών γνωστών αλγορίθμων που χρησιμοποιούνται στην εξόρυξη δεδομένων, του αλγορίθμου συσταδοποίησης K-means, του αλγορίθμου Γραμμικής Παλινδρόμησης και του αλγορίθμου Λογιστικής Παλινδρόμησης.

Παρουσιάζονται τρία παραδείγματα εφαρμογής της εξόρυξης δεδομένων που έχουν αναπτυχθεί από τη Microsoft. Το πρώτο αφορά σύστημα ηλεκτρονικών αγορών προϊόντων λιανικής για την κατηγοριοποίηση των πελατών με τη μέθοδο της συσταδοποίησης, το δεύτερο αφορά σύστημα κρατήσεων/ενοικιάσεων τουριστικών υπηρεσιών με σκοπό την πρόβλεψη των μελλοντικών ενοικιάσεων με τη μέθοδο της γραμμικής παλινδρόμησης ενώ το τρίτο παράδειγμα αφορά σύστημα υπηρεσιών ταξί όπου μελετάται η πιθανότητα πληρωμής φιλοδωρήματος με τη μέθοδο της λογιστικής παλινδρόμησης.

Στη συνέχεια με χρήση πραγματικών στατιστικών δεδομένων μέτρησης της απόδοσης παικτών και ομάδων του αμερικανικού πρωταθλήματος επαγγελματικού μπάσκετ (National Basketball Association – NBA) σε διάρκεια πολλών ετών, επιχειρείται η εφαρμογή των αλγορίθμων που μελετήθηκαν, για την ανακάλυψη συσχετίσεων μεταξύ των στατιστικών και των αποτελεσμάτων. Αρχικά μελετάται και παρουσιάζεται η κατηγοριοποίηση των παικτών του πρωταθλήματος σε συστάδες ανάλογα με τα διαφορετικά χαρακτηριστικά τους, με χρήση του αλγορίθμου συσταδοποίησης K-means. Στη συνέχεια επιχειρείται η εφαρμογή του αλγορίθμου γραμμικής παλινδρόμησης για την πρόβλεψη του MVP της κανονικής διάρκειας του πρωταθλήματος για κάθε σεζόν.

Στο τελευταίο κεφάλαιο αναπτύσσονται συμπεράσματα και σχόλια που προέκυψαν κατά τη διαδικασία της εφαρμογής των αλγορίθμων και της επεξεργασίας των δεδομένων καθώς και σκέψεις για πιθανές μελλοντικές χρήσεις και εφαρμογές.

Abstract

This Thesis is about Data Mining with use of Programming Tools. It begins with a presentation of the theoretical background of Data Mining, its several applications in science, economy and in professional sports. We present the Programming Tools that we are going to use, like Microsoft SQL Server, Machine Learning Services and Microsoft's Azure Data Studio and 3 largely used algorithms in data mining, the Microsoft K-means Clustering Algorithm, the Microsoft Linear Regression Algorithm and Microsoft Logistic Regression Algorithm.

Secondly we present 3 examples of data mining algorithms application, developed by Microsoft. The first example is about a retail e-shop and how we can categorize its customers in clusters using the K-means Clustering Algorithm. The second example is about a ski rental company and the aim is to predict its future bookings with the Linear Regression Algorithm. The last example deals with a Taxi rental company and is about predicting the possibility of tip payment for each trip, with the use of Logistic Regression Algorithm.

Thirdly, with the use of real data from teams and players of the National Basketball Association (NBA) through several seasons, we try to apply the algorithms that are presented above, to discover relations between individual or team statistics and results.

Firstly, using the individual performance data and the K-means Clustering Algorithm, we categorize all the players in clusters, depending on the different playing style and thus the position of each player.

Secondly, through the use of the individual statistics of each player and the Linear Regression Algorithm, we predict the Regular Season MVP for each season.

In the last chapter we present the conclusions and discuss issues that emerged while dealing with data cleaning and pre-processing and while applying the Data Mining Algorithms, and we suggest some thoughts and ideas for future applications.

Περιεχόμενα

Περίληψη	5
Abstract	6
Περιεχόμενα.....	7
Κατάλογος Εικόνων / Σχημάτων	11
Κατάλογος Πινάκων	12
1. Εισαγωγή.....	13
1.1. Στόχος	13
1.2. Δομή της εργασίας.....	14
2. Θεωρητικό υπόβαθρο.....	15
2.1. Η έννοια της Εξόρυξης Δεδομένων.	15
2.2. Συχνές εφαρμογές της εξόρυξης δεδομένων.	16
2.2.1. Οικονομική ανάλυση.....	16
2.2.2. Τομέας τηλεπικοινωνιών.....	17
2.2.3. Ανίχνευση παραβιάσεων	17
2.2.4. Λιανικό εμπόριο	17
2.2.5. Ανώτατη Εκπαίδευση.....	18
2.2.6. Βιομηχανία ενέργειας.....	18
2.2.7. Εξόρυξη Χωρικών Δεδομένων.....	19
2.2.8. Ανάλυση Δεδομένων στη Βιολογία	19
2.2.9. Άλλες επιστημονικές εφαρμογές.....	19
2.2.10. Κατασκευαστικός Κλάδος	20
2.2.11. Εγκληματολογία.....	20
2.2.12. Αντιτρομοκρατική.....	20
2.3. Εξόρυξη Δεδομένων στον Επαγγελματικό Αθλητισμό	21
2.4. Εργαλεία Εξόρυξης Δεδομένων Microsoft Data Mining Tools.	23
2.4.1. Εισαγωγική περιγραφή.....	23
2.4.2. Microsoft SQL Server	23
2.4.3. SQL Server' s Machine Learning Services	24
2.4.4. Azure Data Studio	24
2.5. Παρουσίαση Αλγορίθμου Συσταδοποίησης - Microsoft Clustering Algorithm 25	
2.5.1. Παράδειγμα	26

2.5.2.	Πως δουλεύει ο αλγόριθμος	26
2.5.3.	Απαιτούμενα δεδομένα για τα Μοντέλα Συσταδοποίησης	27
2.6.	Παρουσίαση Αλγορίθμου Λογιστικής Παλινδρόμησης - Microsoft Logistic Regression Algorithm	28
2.6.1.	Παράδειγμα	28
2.6.2.	Πως δουλεύει ο αλγόριθμος	29
2.6.3.	Απαιτούμενα δεδομένα για τα μοντέλα Λογιστικής Παλινδρόμησης	29
2.7.	Παρουσίαση Αλγορίθμου Γραμμικής Παλινδρόμησης - Microsoft Linear Regression Algorithm	30
2.7.1.	Παράδειγμα	31
2.7.2.	Πως δουλεύει ο αλγόριθμος	31
2.7.3.	Απαιτούμενα δεδομένα για τα μοντέλα Γραμμικής Παλινδρόμησης	32
3.	Εκτέλεση παραδειγμάτων	33
3.1.	Εισαγωγή.....	33
3.2.	«Predict ski rental with linear regression with SQL machine learning»	33
3.2.1.	Παρουσίαση προβλήματος.....	33
3.2.2.	Σύνδεση με τη Βάση Δεδομένων	33
3.2.3.	Προετοιμασία των δεδομένων	35
3.2.4.	«Εκπαίδευση» των δεδομένων.....	36
3.2.5.	Δημιουργία αποθηκευμένης διαδικασίας (Stored Procedure) που παράγει το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model)	38
3.2.6.	Αποθήκευση του Linear Regression Model σε ένα νέο πίνακα της Βάσης Δεδομένων	39
3.2.7.	Δημιουργία αποθηκευμένης διαδικασίας που κάνει προβλέψεις για τα δεδομένα	39
3.3.	«Categorizing customers using k-means clustering with SQL machine learning»	40
3.3.1.	Παρουσίαση προβλήματος.....	40
3.3.2.	Σύνδεση με τη Βάση Δεδομένων	40
3.3.3.	Προετοιμασία των δεδομένων	41
3.3.4.	Δημιουργία μοντέλου συσταδοποίησης (clustering).....	43
3.3.5.	Ενσωμάτωση του Clustering Model με SQL Machine Learning.....	45
3.4.	«Predict NYC taxi fares with binary classification».....	47
3.4.1.	Παρουσίαση προβλήματος.....	47
3.4.2.	Σύνδεση με τη Βάση Δεδομένων και προεπισκόπηση των δεδομένων	47

3.4.3.	Εξερεύνηση και οπτικοποίηση των δεδομένων	48
3.4.4.	Δημιουργία γραφημάτων με χρήση της Python μέσω Transact-SQL.....	49
3.4.5.	Calculate trip Distance	51
3.4.6.	Εκπαίδευση και αποθήκευση μοντέλου σε Python με χρήση T-SQL.....	52
3.4.7.	Διαχωρισμός των δεδομένων σε υποσύνολα εκπαίδευσης (training set) και ελέγχου (testing set).....	52
3.4.8.	Κατασκευή μοντέλου λογιστικής παλινδρόμησης.....	52
3.4.9.	Προβλέψεις με χρήση της Python εντός αποθηκευμένης διαδικασίας	54
3.4.10.	Δημιουργία προβλέψεων με χρήση των μοντέλων	57
4.	Εφαρμογή των μεθόδων εξόρυξης δεδομένων στο αμερικάνικο επαγγελματικό μπάσκετ (NBA).....	58
4.1.	Εισαγωγή.....	58
4.2.	Δημιουργία βάσης δεδομένων και ανάλυση πινάκων.	58
4.3.	Κατηγοριοποίηση παικτών NBA σε συστάδες με χρήση του αλγορίθμου συσταδοποίησης K-means	61
4.3.1.	Παρουσίαση προβλήματος.....	61
4.3.2.	Προεπεξεργασία των δεδομένων	62
4.3.3.	Προσδιορισμός επιθυμητών δεδομένων και εισαγωγή σε dataframe του Pandas	64
4.3.4.	Δημιουργία μοντέλου συσταδοποίησης (clustering).....	67
4.3.5.	Ενσωμάτωση του Clustering Model με SQL Machine Learning.....	70
4.4.	Πρόβλεψη MVP του NBA με χρήση του αλγορίθμου γραμμικής παλινδρόμησης - Linear Regression Algorithm	76
4.4.1.	Παρουσίαση προβλήματος.....	76
4.4.2.	Εισαγωγή πινάκων	77
4.4.3.	Προεπεξεργασία των δεδομένων	79
4.4.4.	«Εκπαίδευση» των δεδομένων	80
4.4.5.	Δημιουργία αποθηκευμένης διαδικασίας (Stored Procedure) που παράγει το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model)	83
4.4.6.	Αποθήκευση του Linear Regression Model σε ένα νέο πίνακα της Βάσης Δεδομένων	85
4.4.7.	Δημιουργία αποθηκευμένης διαδικασίας που κάνει προβλέψεις για τα δεδομένα	86
5.	Συμπεράσματα	91
5.1.	Σωστή επιλογή συνόλου δεδομένων για την επιθυμητή ανάλυση.....	91

5.1.1.	Προσδιορισμός κατάλληλων δεδομένων για το σκοπό της ανάλυσης	91
5.1.2.	Εύρεση συνόλων δεδομένων από διαθέσιμες πηγές	91
5.2.	Σημασία προεπεξεργασίας δεδομένων	92
5.2.1.	Καθαρισμός συνόλου δεδομένων.....	92
5.2.2.	Μετατροπή δεδομένων στην επιθυμητή μορφή.....	92
5.2.3.	Εντοπισμός λογικών λαθών	92
5.3.	Αξία της καλής γνώσης του προς ανάλυση αντικειμένου	93
5.3.1.	Περίπτωση «Έντι Τζόνσον».....	93
5.4.	Περαιτέρω ενέργειες για τη συνέχεια της ανάλυσης	94
5.4.1.	Συσταδοποίηση παικτών με τον αλγόριθμο K-means clustering Algorithm	94
5.4.2.	Πρόβλεψη MVP με τον Linear Regression Algorithm	94
	Βιβλιογραφία	95

Κατάλογος Εικόνων / Σχημάτων

Εικόνα 1 - Συσταδοποίηση δεδομένων	25
Εικόνα 2 - Διάγραμμα Διασποράς	26
Εικόνα 3 - Γραμμική απεικόνιση δεδομένων	30
Εικόνα 4 - Εμφάνιση πινάκων που περιέχει η Β.Δ.	34
Εικόνα 5 - Δεδομένα του πίνακα dbo.rental_data	34
Εικόνα 6 - Εμφάνιση πινάκων που περιέχει η Β.Δ.	41
Εικόνα 7 - Γραφική παράσταση Elbow method for KMeans clustering	44
Εικόνα 8 - Εμφάνιση πινάκων και συναρτήσεων της Β.Δ.	47
Εικόνα 9 - Γράφημα με χρήση της Python μέσω Transact-SQL	50
Εικόνα 10 - Πιθανότητα πληρωμής φιλοδωρήματος - βιβλιοθήκη scikit-learn	55
Εικόνα 11 - Πιθανότητα πληρωμής φιλοδωρήματος - βιβλιοθήκη revoscalepy	56
Εικόνα 12 - Οι 5 θέσεις παικτών στο άθλημα του μπάσκετ	61
Εικόνα 13 - Γραφική παράσταση Elbow method for KMeans clustering	68
Εικόνα 14 - Βραβείο MVP κανονικής σεζόν NBA	76
Εικόνα 15 - Αποθηκευμένη διαδικασία generate_MVP_py_model	85

Κατάλογος Πινάκων

Πίνακας 1 - Περιεχόμενα Πίνακα py_rental_predictions	39
Πίνακας 2 - Περιεχόμενα Πίνακα dbo.customer	41
Πίνακας 3 - Περιεχόμενα Πίνακα customer_data.....	43
Πίνακας 4 - Πλήθος δεδομένων ανά cluster	44
Πίνακας 5 - Μέσες τιμές ανά cluster	45
Πίνακας 6 - Περιεχόμενα Πίνακα py_customer_clusters	46
Πίνακας 7 - Emails πελατών που ανήκουν στη συστάδα 0	46
Πίνακας 8 - Περιεχόμενα Πίνακα Seasons_Stats	59
Πίνακας 9 - Στήλες πίνακα Seasons_Stats.....	60
Πίνακας 10 - Πρώτοι σκόρερ όλων των εποχών	60
Πίνακας 11 - Τιμές NULL που εμπεριέχονται στον πίνακα Seasons_Stats	63
Πίνακας 12 - Συνολικά παιχνίδια παίκτη Lou Williams τη σεζόν 2016-2017	64
Πίνακας 13 - Περιεχόμενα του Πίνακα Player_Stats	67
Πίνακας 14 - Πλήθος δεδομένων ανά cluster	69
Πίνακας 15 - Μέσες τιμές ανά cluster	69
Πίνακας 16 - Εμφάνιση συστάδας για κάθε παίκτη	72
Πίνακας 17 - Παίκτες ανά σεζόν που ανήκουν στη συστάδα μηδέν	73
Πίνακας 18 - Περιεχόμενα Πίνακα MVP	78
Πίνακας 19 - Αρχική και φιλτραρισμένη λίστα στηλών.....	80
Πίνακας 20 - Training και test σύνολο δεδομένων.....	82
Πίνακας 21 - Τιμές πρόβλεψης και σφάλμα	83
Πίνακας 22 - Αποθηκευμένο μοντέλο στον πίνακα MVP_py_models	86
Πίνακας 23 - Σύγκριση τιμών πρόβλεψης με πραγματικές	89
Πίνακας 24 - Σφάλμα, προβλέψεις και πραγματικές τιμές.....	90

1. Εισαγωγή

1.1. Στόχος

Στόχος της διπλωματικής εργασίας είναι να μελετηθεί η έννοια της Εξόρυξης Δεδομένων και η χρήση της για την εξαγωγή γνώσης από μεγάλα σύνολα δεδομένων. Μελετώνται γνωστοί αλγόριθμοι εξόρυξης δεδομένων και επιχειρείται εφαρμογή τους με εργαλεία που έχουν αναπτυχθεί για τη διαχείριση Βάσεων Δεδομένων και την εξόρυξη δεδομένων με χρήση τεχνικών Μηχανικής Μάθησης.

Μελετώνται τρία παραδείγματα εφαρμογής της εξόρυξης δεδομένων που έχουν αναπτυχθεί από τη Microsoft με χρήση του SQL Server και του Azure Data Studio. Το πρώτο παράδειγμα αφορά σύστημα ηλεκτρονικών αγορών προϊόντων λιανικής και επιχειρείται κατηγοριοποίηση των πελατών της εταιρείας με τη μέθοδο της συσταδοποίησης για τη βελτίωση της αποτελεσματικότητας των προωθητικών ενεργειών της εταιρείας. Το δεύτερο παράδειγμα αφορά σύστημα κρατήσεων/ενοικιάσεων τουριστικών υπηρεσιών και επιχειρείται η πρόβλεψη των μελλοντικών ενοικιάσεων με τη μέθοδο της γραμμικής παλινδρόμησης. Τέλος το τρίτο παράδειγμα αφορά σύστημα υπηρεσιών ταξί και μελετάται η πιθανότητα πληρωμής φιλοδωρήματος με τη μέθοδο της λογιστικής παλινδρόμησης.

Στη συνέχεια επιχειρείται η εφαρμογή των εργαλείων και των μεθόδων της εξόρυξης δεδομένων στον τομέα του αθλητισμού και συγκεκριμένα στο αμερικανικό επαγγελματικό μπάσκετ (National Basketball Association – NBA). Με χρήση στατιστικών δεδομένων μέτρησης της απόδοσης παικτών και ομάδων του NBA σε διάρκεια πολλών ετών, επιχειρείται η εφαρμογή των αλγορίθμων που μελετήθηκαν για την ανακάλυψη συσχετίσεων μεταξύ των στατιστικών και των ατομικών και ομαδικών αποτελεσμάτων και δημιουργία αποθηκευμένων διαδικασιών για την πρόβλεψη αυτών βάσει παραμέτρων που εισάγονται από το χρήστη.

1.2. Δομή της εργασίας

Στο πρώτο κεφάλαιο επιχειρείται μια εισαγωγή στο αντικείμενο, το σκοπό και τη δομή της εργασίας.

Στο δεύτερο κεφάλαιο παρουσιάζεται το θεωρητικό υπόβαθρο της εξόρυξης δεδομένων, των εργαλείων και των αλγορίθμων που θα χρησιμοποιηθούν στη συνέχεια.

Στο τρίτο κεφάλαιο παρουσιάζονται ορισμένα παραδείγματα που αναπτύχθηκαν με χρήση αλγορίθμων μηχανικής μάθησης από τη Microsoft με το εργαλείο Azure Data Studio.

Στο τέταρτο κεφάλαιο επιχειρείται εφαρμογή των τεχνικών και των αλγορίθμων που περιγράφονται νωρίτερα, σε πραγματικά δεδομένα του επαγγελματικού αθλητισμού.

Στο πέμπτο κεφάλαιο αξιολογούνται τα αποτελέσματα και παρατίθενται σχόλια και συμπεράσματα που προέκυψαν από τη διαδικασία της δοκιμής των αλγορίθμων.

2. Θεωρητικό υπόβαθρο

2.1. Η έννοια της Εξόρυξης Δεδομένων.

Η Εξόρυξη Δεδομένων (data mining) είναι η διαδικασία εξαγωγής γνώσης από μεγάλα σύνολα δεδομένων μέσω της ανακάλυψης μοτίβων σε αυτά με χρήση μεθόδων που συνδυάζουν την επιστήμη της Στατιστικής, τα Συστήματα Διαχείρισης Βάσεων Δεδομένων (Database Management Systems - DBMS) και τις τεχνικές της Μηχανικής Μάθησης (Machine Learning). Η εξόρυξη δεδομένων αποτελεί διεπιστημονικό υποπεδίο της Επιστήμης των Υπολογιστών και της Στατιστικής με απώτερο σκοπό την εξαγωγή πληροφορίας από ένα σύνολο δεδομένων με έξυπνες μεθόδους και τη μετατροπή της σε κατανοητή μορφή για περαιτέρω χρήση. Η εξόρυξη δεδομένων είναι στάδιο της διαδικασίας «Ανακάλυψης Γνώσης σε Βάσεις Δεδομένων» («Knowledge Discovery in Databases» - KDD). [1]

Τα βασικά βήματα που συνήθως περιλαμβάνει η διαδικασία της εξόρυξης δεδομένων είναι:

- **Κατανόηση του επιχειρηματικού στόχου:** Πλήρης και ουσιαστική κατανόηση των παραμέτρων του project για το οποίο θα χρησιμοποιηθεί η εξόρυξη. Μεταξύ των παραμέτρων είναι η τρέχουσα κατάσταση της επιχείρησης, ο βασικός στόχος του συγκεκριμένου project και τα κριτήρια βάσει των οποίων η διαδικασία θα θεωρηθεί επιτυχής.
- **Κατανόηση των δεδομένων:** Προσδιορισμός των απαιτούμενων δεδομένων για την επίλυση του προβλήματος και συλλογή τους από όλες τις διαθέσιμες πηγές.
- **Προεπεξεργασία των δεδομένων:** Προετοιμασία των δεδομένων ώστε να βρεθούν σε κατάλληλη μορφή για να απαντήσουν στο επιχειρηματικό πρόβλημα, διόρθωση τυχόν ζητημάτων ποιότητας των δεδομένων όπως κενές ή διπλές εγγραφές.
- **Κατασκευή μοντέλων:** Χρήση αλγορίθμων για τον εντοπισμό προτύπων εντός των δεδομένων.

- **Αξιολόγηση αποτελεσμάτων:** Εκτίμηση της ποιότητας των αποτελεσμάτων της εκτέλεσης των μοντέλων και της συμβολής τους στην επίτευξη του επιχειρηματικού στόχου. Συχνά αυτό το βήμα έχει επαναληπτικό χαρακτήρα, μέχρι να βρεθεί ο καταλληλότερος αλγόριθμος που δίνει τα χρησιμότερα αποτελέσματα.
- **Deployment:** Η διαδικασία της μετατροπής των αποτελεσμάτων σε κατανοητή και χρηστική μορφή για τους αυτούς που θα τα χρησιμοποιήσουν για τη λήψη αποφάσεων. [2]

2.2. Συχνές εφαρμογές της εξόρυξης δεδομένων.

Η εξόρυξη δεδομένων χρησιμοποιείται σε πολλούς τομείς των επιχειρήσεων και της έρευνας, μεταξύ των οποίων οι πωλήσεις και το μάρκετινγκ, η ανάπτυξη προϊόντων, η υγεία και η εκπαίδευση. Όταν χρησιμοποιείται σωστά η εξόρυξη δεδομένων μπορεί να παρέχει ένα σημαντικό πλεονέκτημα έναντι του ανταγωνισμού δίνοντας τη δυνατότητα καλύτερης γνώσης των χαρακτηριστικών του πελατολογίου, ανάπτυξης αποτελεσματικών στρατηγικών μάρκετινγκ, αύξηση των εσόδων και μείωση του κόστους. [2]

2.2.1. Οικονομική ανάλυση

Ο χρηματοοικονομικός και τραπεζικός κλάδος χρειάζονται και χρησιμοποιούν δεδομένα υψηλής ποιότητας και αξιοπιστίας. Στην αγορά δανεισμού, οικονομικά και προσωπικά δεδομένα των πελατών χρησιμοποιούνται για διάφορους σκοπούς, όπως η πρόβλεψη της αποπληρωμής των δανείων και ο προσδιορισμός της πιστωτικής αξιοπιστίας. Η εξόρυξη δεδομένων βοηθά στην αποδοτικότερη διαχείριση αυτών των εργασιών.

Οι τεχνικές ταξινόμησης διευκολύνουν το διαχωρισμό των κρίσιμων παραγόντων που επηρεάζουν τη λήψη αποφάσεων των πελατών από τους παράγοντες που δεν έχουν αντίστοιχη επιρροή. Επιπλέον, οι τεχνικές συσταδοποίησης πολλών διαστάσεων επιτρέπουν τον εντοπισμό των πελατών με παρόμοια συμπεριφορά όσον αφορά τις

αποπληρωμές. Η εξόρυξη και η ανάλυση δεδομένων μπορούν επίσης να βοηθήσουν στον εντοπισμό του ξεπλύματος χρήματος και άλλων οικονομικών εγκλημάτων. [3]

2.2.2. Τομέας τηλεπικοινωνιών

Πρόκειται για έναν ταχύτατα αναπτυσσόμενο τομέα, ιδιαίτερα μετά την εξάπλωση του Internet. Η εξόρυξη δεδομένων μπορεί να βελτιώσει την ποιότητα των παρεχόμενων υπηρεσιών, δίνοντας έτσι συγκριτικό πλεονέκτημα έναντι του ανταγωνισμού στους μεγάλους παρόχους του κλάδου.

Η ανάλυση προτύπων χωρικών και χρονικών δεδομένων μπορεί να παίζει τεράστιο ρόλο στις υπηρεσίες κινητής τηλεφωνίας καθώς και στις υπηρεσίες διαδικτυακών εφαρμογών πληροφορικής. Τεχνικές όπως η εύρεση ακραίων τιμών (outlier analysis) μπορούν να βοηθήσουν στον εντοπισμό απάτης από τους χρήστες. Επίσης τα εργαλεία ηλεκτρονικής αναλυτικής επεξεργασίας (OLAP) και οπτικοποίησης βοηθούν στη σύγκριση πληροφοριών σχετικά με τη συμπεριφορά των χρηστών, το κέρδος, την κίνηση των δεδομένων, την υπερφόρτωση των συστημάτων κ.ά. [3]

2.2.3. Ανίχνευση παραβιάσεων

Η παγκόσμια συνδεσιμότητα της εποχής μας έχει εγείρει πολλά ζητήματα ασφάλειας πληροφοριών και συστημάτων. Οι διαδικτυακές υπηρεσίες αντιμετωπίζουν απειλές και απόπειρες παραβίασης της αξιοπιστίας και της ακεραιότητάς τους. Ως εκ τούτου η ανίχνευση τέτοιων ενεργειών αποτελεί κρίσιμο τομέα της εξόρυξης δεδομένων.

Μέσω της ανάλυσης συσχέτισης, των τεχνικών συνάθροισης, οπτικοποίησης και εργαλεία σύνταξης ερωτημάτων μπορεί να ανιχνεύσει αποτελεσματικά διακυμάνσεις και ανωμαλίες σε σχέση με την κανονική συμπεριφορά. [3]

2.2.4. Λιανικό εμπόριο

Ο τομέας του οργανωμένου λιανικού εμπορίου χειρίζεται μεγάλες ποσότητες δεδομένων που αφορούν τις πωλήσεις, το ιστορικό συναλλαγών, την παράδοση προϊόντων, τις καταναλωτικές συνήθειες και τις λοιπές υπηρεσίες των πελατών. Με

την ανάπτυξη του ηλεκτρονικού εμπορίου οι βάσεις δεδομένων του λιανικού εμπορίου έχουν γίνει ακόμη μεγαλύτερες.

Στο σύγχρονο λιανικό εμπόριο, οι αποθήκες δεδομένων σχεδιάζονται και κατασκευάζονται ώστε να λαμβάνουν το μέγιστο όφελος από την εξόρυξη δεδομένων. Οι ανάλυση δεδομένων πολλών διαστάσεων βοηθά στον χειρισμό δεδομένων που αφορούν διαφορετικούς πελάτες, προϊόντα, περιοχές και time-zones. Οι επιχειρήσεις του ηλεκτρονικού εμπορίου μπορούν επιπλέον να προτείνουν προϊόντα για να αυξήσουν τα κέρδη των πωλήσεων και στη συνέχεια να αξιολογήσουν την αποτελεσματικότητα των προωθητικών τους ενεργειών. Επομένως, από τον εντοπισμό προτύπων στις αγοραστικές συνήθειες μέχρι τη βελτίωση της εξυπηρέτησης και ικανοποίησης του πελάτη, η εξόρυξη δεδομένων προσφέρει νέες δυνατότητες στον συγκεκριμένο κλάδο. [3]

2.2.5. Ανώτατη Εκπαίδευση

Καθώς τα ποσοστά του πληθυσμού που λαμβάνουν ανώτατη εκπαίδευση αυξάνονται παγκοσμίως, τα πανεπιστημιακά ιδρύματα αναζητούν καινοτόμες λύσεις ώστε να εξυπηρετήσουν τις αυξανόμενες ανάγκες. Τα ιδρύματα χρησιμοποιούν την εξόρυξη δεδομένων για να προβλέψουν ποιοι φοιτητές πρόκειται να εγγραφούν στα προγράμματά τους, ποιοι θα χρειαστούν συμπληρωματική βοήθεια για να την ολοκλήρωση των σπουδών, βελτιώνοντας έτσι τη συνολική διαχείριση των ακαδημαϊκών προγραμμάτων.

Επιπλέον, η πρόγνωση της μετέπειτα σταδιοδρομίας των φοιτητών και η παρουσίαση των δεδομένων, διευκολύνεται με αποτελεσματικότερη ανάλυση. Με αυτό τον τρόπο, οι τεχνικές της εξόρυξης δεδομένων βοηθούν στην ανακάλυψη κρυμμένων προτύπων σε μεγάλου μεγέθους βάσεις δεδομένων των ακαδημαϊκών ιδρυμάτων. [3]

2.2.6. Βιομηχανία ενέργειας

Τα μαζικά δεδομένα (Big Data) σήμερα είναι διαθέσιμα και στον τομέα της ενέργειας, πράγμα που δημιουργεί την ανάγκη για τεχνικές εξόρυξης δεδομένων. Τα Δένδρα Απόφασης (Decision Trees) και οι Μηχανές Διανυσμάτων Υποστήριξης (Support

Vector Machines - SVM) είναι από τις πιο δημοφιλείς προσεγγίσεις στη βιομηχανία της ενέργειας, παρέχοντας εφαρμόσιμες λύσεις για τη λήψη αποφάσεων και τη διαχείριση. Επιπρόσθετα, η εξόρυξη δεδομένων μπορεί να επιτύχει επίσης κέρδη μέσω της πρόβλεψης της παραγωγής και της κατανάλωσης ενέργειας και της διαμόρφωσης των τιμών της. [3]

2.2.7. Εξόρυξη Χωρικών Δεδομένων

Τα Γεωγραφικά Συστήματα Πληροφοριών (Geographic Information Systems – GIS) και αρκετές άλλες εφαρμογές πλοήγησης χρησιμοποιούν την εξόρυξη δεδομένων για τη διασφάλιση ζωτικής σημασίας πληροφορίες και την κατανόηση τυχόν επιπλοκών. Η νέα αυτή τάση περιλαμβάνει την εξαγωγή γεωγραφικών, περιβαλλοντικών και αστρονομικών δεδομένων, μεταξύ των οποίων είναι ακόμη και φωτογραφίες του έξω-διαστήματος. Κατά κανόνα η εξόρυξη χωρικών δεδομένων αποκαλύπτει πτυχές όπως η απόσταση και η τοπολογία. [3]

2.2.8. Ανάλυση Δεδομένων στη Βιολογία

Οι τεχνικές Ανάλυσης Δεδομένων στη Βιολογία είναι συχνές στη γονιδιακή, την πρωτεϊνική και τη βιοϊατρική έρευνα. Από τον προσδιορισμό της συμπεριφοράς των ασθενών και την πρόβλεψη των επισκέψεων, μέχρι τον εντοπισμό θεραπείας για την ασθένειά τους, οι τεχνικές της ανάλυσης δεδομένων προσφέρουν πολλά πλεονεκτήματα.

Κάποιες από τις εφαρμογές της εξόρυξης δεδομένων στον τομέα της Βιοπληροφορικής είναι η σημασιολογική χαρτογράφηση των δεδομένων, η ανάλυση συσχέτισης και διαδρομών, η οπτικοποίηση, η ανακάλυψη δομικών προτύπων και η ανάλυση γενετικών δικτύων και πρωτεϊνικών διαδρομών. [3]

2.2.9. Άλλες επιστημονικές εφαρμογές

Η γρήγορη αριθμητική προσομοίωση σε επιστημονικά πεδία όπως η Χημική Μηχανική, η Μηχανική Ρευστών, η προσομοίωση κλίματος και οικοσυστήματος

παράγουν τεράστιο όγκο δεδομένων. Η εξόρυξη δεδομένων παρέχει δυνατότητες όπως οι αποθήκες δεδομένων, η προεπεξεργασία των δεδομένων, η οπτικοποίηση, η εξόρυξη μέσω γραφημάτων κ.ά.. [3]

2.2.10. Κατασκευαστικός Κλάδος

Ο σχεδιασμός προϊόντων ή/και κτιρίων χρησιμοποιεί την εξόρυξη δεδομένων για να εξάγει συσχετίσεις μεταξύ της μελέτης και του τελικού προϊόντος. Επιπλέον, οι μέθοδοι της εξόρυξης δεδομένων μπορούν να χρησιμεύσουν και στην πρόβλεψη του τελικού κόστους καθώς και στον απαιτούμενο χρόνο για την υλοποίηση του σχεδιασμού. [3]

2.2.11. Εγκληματολογία

Εργασίες εξόρυξης δεδομένων χρησιμοποιούνται επίσης στην Εγκληματολογία, η οποία μελετά τα χαρακτηριστικά των εγκληματικών πράξεων. Αρχικά οι γραπτές αναφορές μετατρέπονται από την αρχική μορφή κειμένου σε επεξεργασμένα σύνολα λέξεων – κλειδιών. Στη συνέχεια με την ανακάλυψη προτύπων σε τεράστια σύνολα δεδομένων γίνεται εντοπισμός μηχανισμών του εγκλήματος. [3]

2.2.12. Αντιτρομοκρατική

Εξεζητημένοι μαθηματικοί αλγόριθμοι μπορούν να υποδείξουν ποια μονάδα είναι κατάλληλη να αναλάβει την περαιτέρω διερεύνηση κάποιας υπόθεσης της αντιτρομοκρατικής υπηρεσίας. Η εξόρυξη δεδομένων μπορεί να βοηθήσει ακόμη και σε ζητήματα αστυνομικής οργάνωσης όπως τον καθορισμό του τρόπου κατανομής των δυνάμεων και της καθοδήγησης των ερευνών στα σύνορα μιας χώρας. [3]

2.3. Εξόρυξη Δεδομένων στον Επαγγελματικό Αθλητισμό

Σε όλα τα πεδία του αθλητισμού υπάρχει τεράστιος όγκος δεδομένων. Τα δεδομένα αυτά μπορεί να αφορούν ατομικές και ομαδικές επιδόσεις, προπονητικές ή διοικητικές αποφάσεις και διάφορα άλλα γεγονότα που συμβαίνουν στη διάρκεια των αγώνων. Περισσότερο από τον τρόπο που θα γίνει η συλλογή των δεδομένων, το ζητούμενο είναι το ποια δεδομένα έχει αξία να συλλεχθούν και το πώς αυτά θα χρησιμοποιηθούν. Βρίσκοντας τους σωστούς τρόπους να ανακαλύψουν λογική και να εξάγουν γνώση από τα δεδομένα, οι αθλητικοί οργανισμοί μπορούν να εξασφαλίσουν ανταγωνιστικό πλεονέκτημα έναντι των αντιπάλων τους. Αυτή η αναζήτηση της γνώσης μέσω των δεδομένων έχει εφαρμογή σε όλα τα επίπεδα ενός οργανισμού, από τη βελτίωση των επιδόσεων των αθλητών κατά τους αγώνες, μέσω της ανάλυσης των βίντεο των αγώνων τους, μέχρι τη στατιστική ανάλυση και τις τεχνικές πρόβλεψης από τα τμήματα scouting για τον εντοπισμό των ταλέντων που θα έχουν τη θετικότερη επίδραση στον οργανισμό. Η εξόρυξη δεδομένων εξελίσσεται ραγδαία σε αναπόσπαστο κομμάτι της διαδικασίας λήψης αποφάσεων στον αθλητισμό, με τους προπονητές και μάνατζερ των οργανισμών να χρησιμοποιούν τις τεχνικές μηχανικής μάθησης και προσομοίωσης για να καταστρώσουν τις βέλτιστες στρατηγικές για τις επερχόμενες διοργανώσεις.

Το πρώτο μέρος του προβλήματος είναι ο προσδιορισμός των μετρούμενων δεδομένων της απόδοσης. Πολλά από τα μετρούμενα δεδομένα μπορούν να οδηγήσουν σε εσφαλμένη προσέγγιση και παρόλο που ο τελικός στόχος κάθε ομάδας ή αθλητή είναι να πετύχει μεγαλύτερο σκορ από τον αντίπαλο, η απόδοση της ομάδας ή του αθλητή δε θα πρέπει να μετράται μονοδιάστατα βάσει των πόντων που πετυχαίνει και μόνον αυτών. Οι σύγχρονες προσεγγίσεις εισάγουν νέα εξειδικευμένα στατιστικά δεδομένα τα οποία είναι προσανατολισμένα σε κάθε άθλημα ανάλογα με τα ιδιαίτερα χαρακτηριστικά του.

Το δεύτερο μέρος του προβλήματος είναι ο εντοπισμός προτύπων ή μοτίβων στα δεδομένα. Τα πρότυπα αυτά μπορούν να περιλαμβάνουν τις ανοδικές ή πτωτικές τάσεις κάποιου μεγέθους, να προσδιορίζουν την πιθανότητα επερχόμενων τραυματισμών μέσω της παρακολούθησης της απόδοσης στις προπονήσεις ή να κάνουν προβλέψεις βασισμένες στην προϊστορία των ομάδων και των αθλητών.

Οι επαγγελματικοί αθλητικοί οργανισμοί συχνά είναι επιχειρήσεις πολλών εκατομμυρίων και κάθε απόφαση που λαμβάνεται να έχει κόστος ή κέρδος επίσης εκατομμυρίων. Με τέτοια ποσά να διακυβεύονται σε κάθε κίνηση, γίνεται σαφές ότι μία λάθος απόφαση μπορεί να κρατήσει έναν οργανισμό πολύ πίσω στην εξέλιξη και την πρόοδό του. Ως εκ τούτου με τόσα πολλά ρίσκα και την κρισιμότητα της λήψης ορθών αποφάσεων, η αθλητική βιομηχανία αποτελεί ελκυστικό πεδίο για τις εφαρμογές εξόρυξης δεδομένων.

2.4. Εργαλεία Εξόρυξης Δεδομένων Microsoft Data Mining Tools.

2.4.1. Εισαγωγική περιγραφή

Κατά την εκπόνησή της παρούσας εργασίας μελετήθηκαν Αλγόριθμοι Εξόρυξης Δεδομένων μέσω της λειτουργίας Machine Learning Services του Microsoft SQL Server. Ως επεξεργαστής κώδικα (code editor) χρησιμοποιήθηκε το λογισμικό Azure Data Studio.

2.4.2. Microsoft SQL Server

Ο Microsoft SQL Server είναι ένα Σύστημα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων (Relational Database Management System - RDBMS), το οποίο έχει αναπτυχθεί από τη Microsoft. Είναι ένα λογισμικό με πρωταρχική λειτουργία την αποθήκευση και ανάκτηση των δεδομένων μίας βάσης από άλλες εφαρμογές λογισμικού, οι οποίες μπορούν να τρέχουν είτε στον ίδιο είτε σε διαφορετικό υπολογιστή που βρίσκεται στο ίδιο δίκτυο, είτε τοπικό είτε στο Internet. [4]

Ο Microsoft SQL Server υποστηρίζει μεγάλη ποικιλία εφαρμογών επεξεργασίας συναλλαγών, επιχειρηματικής ευφυίας και ανάλυσης δεδομένων σε εταιρικά πληροφοριακά περιβάλλοντα. Ο Microsoft SQL Server είναι ένα από τα τρία πιο διαδεδομένα εργαλεία επεξεργασίας βάσεων δεδομένων, μαζί με την Oracle Database και το DB2 της IBM.

Όπως και τα άλλα Συστήματα Διαχείρισης Σχεσιακών Βάσεων Δεδομένων, ο SQL Server έχει κατασκευαστεί και λειτουργεί πάνω στην SQL, μια προτυποποιημένη γλώσσα προγραμματισμού που χρησιμοποιείται από τους διαχειριστές βάσεων δεδομένων (database administrators – DBAs) και άλλες ειδικότητες του τομέα της Πληροφορικής, για τη διαχείριση των βάσεων και τη δημιουργία ερωτημάτων (queries) πάνω στα δεδομένα που αυτές περιέχουν. Ο SQL Server είναι άρρηκτα συνδεδεμένος με την Transact – SQL (T-SQL), μια υλοποίηση της SQL που περιέχει ένα σύνολο επιπρόσθετων προγραμματιστικών εργαλείων. [5]

2.4.3. SQL Server's Machine Learning Services

Το Machine Learning Services είναι μία λειτουργία του SQL Server που δίνει τη δυνατότητα να εκτελεστούν τμήματα κώδικα (scripts) σε γλώσσα προγραμματισμού Python και R πάνω σε σχεσιακά δεδομένα. Ο χρήστης μπορεί να χρησιμοποιήσει πακέτα και πλαίσια λογισμικού ανοικτού κώδικα καθώς και πακέτα της Python και της R για προβλεπτική ανάλυση (predictive analytics) και μηχανική μάθηση (Machine Learning). Τα scripts εκτελούνται εντός της Βάσης Δεδομένων χωρίς μεταφορά δεδομένων έξω από τον SQL Server ή στο διαδίκτυο. [6]

2.4.4. Azure Data Studio

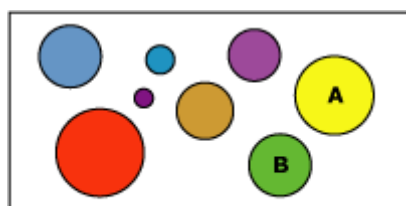
Το Azure Data Studio είναι ένα ανεξάρτητο πλατφόρμας λογισμικό Βάσεων Δεδομένων για επαγγελματίες που χρησιμοποιούν είτε πλατφόρμες εγκατεστημένες τοπικά στον υπολογιστή τους (on-premises platforms) είτε cloud υπηρεσίες, σε περιβάλλον Windows, macOS και Linux.

Το Azure Data Studio προσφέρει στο χρήστη την εμπειρία ενός σύγχρονου επεξεργαστή κώδικα (code editor) με τη βοήθεια εργαλείων όπως η αυτόματη συμπλήρωση κώδικα (IntelliSense), ο χειρισμός αποσπασμάτων κώδικα (code snippets), source control integration και ενσωματωμένη τερματική μηχανή (terminal). Είναι προσανατολισμένο στο χρήστη μιας πλατφόρμας δεδομένων, παρέχοντας ενσωματωμένη δυνατότητα γραφικής απεικόνισης δεδομένων που προκύπτουν ως αποτελέσματα ερωτημάτων σε Βάσεις Δεδομένων (queries) καθώς και επεξεργάσιμα και προσαρμόσιμα από το χρήστη διαγράμματα. [7]

2.5. Παρουσίαση Αλγορίθμου Συσταδοποίησης - Microsoft Clustering Algorithm

Ο Microsoft Clustering Algorithm είναι ένας αλγόριθμος κατακερματισμού ή συσταδοποίησης ο οποίος διατρέχει επαναληπτικά τις εγγραφές ενός σετ δεδομένων ώστε να τα ομαδοποιήσει σε συστάδες που περιέχουν παρόμοια χαρακτηριστικά. Αυτές οι ομαδοποιήσεις είναι χρήσιμες για την εξερεύνηση των δεδομένων, την εύρεση τυχόν ανωμαλιών και την πρόβλεψη.

Τα μοντέλα συσταδοποίησης εντοπίζουν συσχετίσεις σε ένα σετ δεδομένων οι οποίες δε θα εξάγονταν λογικά από την απλή παρατήρηση. Για παράδειγμα θα μπορούσαμε εύκολα να μαντέψουμε ότι οι άνθρωποι που μεταβαίνουν στις δουλειές τους με ποδήλατο δε μένουν κατά κανόνα μακριά από το χώρο εργασίας τους. Ο αλγόριθμος αντίθετα μπορεί να βρει άλλα χαρακτηριστικά των ποδηλατών που δεν είναι τόσο προφανή. Στο ακόλουθο διάγραμμα, η συστάδα A αναπαριστά δεδομένα για ανθρώπους που πηγαίνουν με το αυτοκίνητο στη δουλειά τους ενώ η συστάδα B αναπαριστά δεδομένα για εκείνους που πηγαίνουν με ποδήλατο.



A = Commuters who drive to work
B = Commuters who bicycle to work

Εικόνα 1 - Συσταδοποίηση δεδομένων

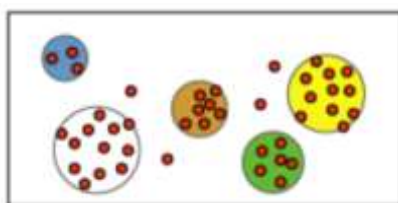
Ο Clustering Algorithm διαφέρει από άλλους αλγορίθμους όπως ο Microsoft Decision Tree Algorithm στο ότι δε χρειάζεται να οριστεί προβλέψιμη στήλη για να κατασκευαστεί το μοντέλο συσταδοποίησης. Ο Clustering Algorithm εκπαιδεύει το μοντέλο αυστηρά μέσω των συσχετίσεων που υπάρχουν στα δεδομένα και μέσω των συστάδων που αναγνωρίζει.

2.5.1. Παράδειγμα

Έστω ένα σύνολο ανθρώπων οι οποίοι μοιράζονται παρόμοια δημογραφικά χαρακτηριστικά και αγοράζουν παρόμοια προϊόντα από την εταιρεία Adventure Works. Αυτό το σύνολο ανθρώπων αποτελεί μία συστάδα δεδομένων. Παραπάνω από μία τέτοιες συστάδες ενδέχεται να υπάρχουν σε μία Βάση Δεδομένων. Παρατηρώντας τις στήλες που αποτελούν μία συστάδα, γίνεται πιο ξεκάθαρο το πώς οι εγγραφές συσχετίζονται μεταξύ τους.

2.5.2. Πως δουλεύει ο αλγόριθμος

Ο Microsoft Clustering Algorithm αρχικά αναγνωρίζει συσχετίσεις σε ένα σύνολο δεδομένων και δημιουργεί μία σειρά από συστάδες με βάση αυτές τις συσχετίσεις. Το διάγραμμα διασποράς (scatter plot) είναι ένας καλός τρόπος να απεικονιστεί οπτικά το πώς ο αλγόριθμος ομαδοποιεί τα δεδομένα, όπως φαίνεται στο ακόλουθο διάγραμμα. Το διάγραμμα διασποράς αναπαριστά όλες τις περιπτώσεις του συνόλου δεδομένων και κάθε περίπτωση είναι ένα σημείο στο γράφημα. Οι συστάδες ομαδοποιούν σημεία στο γράφημα και έτσι απεικονίζουν τις συσχετίσεις που αναγνωρίζει ο αλγόριθμος.



Εικόνα 2 - Διάγραμμα Διασποράς

Αφού ορίσει τις αρχικές συστάδες, ο αλγόριθμος υπολογίζει το πόσο καλά οι συστάδες αναπαριστούν τις ομαδοποιήσεις των σημείων και στη συνέχεια προσπαθεί να επαναπροσδιορίσει τις ομαδοποιήσεις ώστε να δημιουργήσει συστάδες που να αναπαριστούν καλύτερα τα δεδομένα. Ο αλγόριθμος επαναλαμβάνει τη διαδικασία έως ότου να μη μπορεί να βελτιώσει περαιτέρω τα αποτελέσματα με νέο ορισμό των συστάδων.

Μπορούμε να προσαρμόσουμε τον τρόπο που δουλεύει ο αλγόριθμος επιλέγοντας μια συγκεκριμένη τεχνική συσταδοποίησης, περιορίζοντας τον αριθμό των συστάδων ή αλλάζοντας την ελάχιστη υποστήριξη (support) που απαιτείται για τη δημιουργία μιας συστάδας. Ο Microsoft Clustering Algorithm περιλαμβάνει 2 δημοφιλείς μεθόδους συσταδοποίησης: την K-means συσταδοποίηση και την Expectation Maximization μέθοδο.

2.5.3. Απαιτούμενα δεδομένα για τα Μοντέλα Συσταδοποίησης

Όταν προετοιμάζουμε τα δεδομένα για χρήση στην εκπαίδευση ενός μοντέλου συσταδοποίησης, πρέπει να κατανοούμε τις απαιτήσεις για το συγκεκριμένο αλγόριθμο, όπως την ποσότητα των δεδομένων που χρειάζονται και το πώς τα δεδομένα χρησιμοποιούνται. Οι απαιτήσεις για ένα μοντέλο συσταδοποίησης είναι:

- **Μία στήλη μοναδικού κλειδιού** Κάθε μοντέλο πρέπει να περιέχει μία αριθμητική ή αλφαριθμητική στήλη η οποία ορίζει μοναδικά κάθε εγγραφή. Συνδυαστικά κλειδιά δεν επιτρέπονται.
- **Στήλες εισαγωγής δεδομένων** Κάθε μοντέλο πρέπει να περιέχει τουλάχιστον μία στήλη εισαγωγής δεδομένων η οποία περιέχει τις τιμές που χρησιμοποιούνται για τη δημιουργία των συστάδων. Μπορούμε να έχουμε όσες στήλες εισαγωγής δεδομένων επιθυμούμε, αλλά ανάλογα με το πλήθος των τιμών κάθε στήλης, η προσθήκη νέων στηλών μπορεί να αυξήσει τον απαιτούμενο χρόνο για την εκπαίδευση του μοντέλου.
- **Προαιρετική προς πρόβλεψη στήλη** Ο αλγόριθμος δε χρειάζεται προβλέψιμη στήλη για τη δημιουργία του μοντέλου, μπορούμε όμως να προσθέσουμε μια σχεδόν κάθε τύπου δεδομένων. Οι τιμές της προβλέψιμης στήλης μπορούν να αντιμετωπιστούν ως δεδομένα εισόδου του μοντέλου συσταδοποίησης ή μπορούμε να ορίσουμε ότι η στήλη θα χρησιμοποιηθεί μόνο για πρόβλεψη. Για παράδειγμα, αν θέλουμε να προβλέψουμε το εισόδημα ενός πελάτη με συσταδοποίηση βάσει δημογραφικών δεδομένων όπως η περιοχή ή η ηλικία, θα ορίζαμε το εισόδημα ως **PredictOnly** και θα προσθέταμε όλες τις άλλες στήλες, όπως περιοχή και ηλικία, ως δεδομένα εισόδου.

2.6. Παρουσίαση Αλγορίθμου Λογιστικής Παλινδρόμησης - Microsoft Logistic Regression Algorithm

Η Λογιστική Παλινδρόμηση είναι μια ευρέως διαδεδομένη τεχνική της επιστήμης της Στατιστικής που χρησιμοποιείται για την ανάλυση δυαδικών δεδομένων.

Υπάρχουν πολλές εφαρμογές της λογιστικής παλινδρόμησης στη στατιστική έρευνα, οι οποίες χρησιμοποιούν διαφορετικές τεχνικές μάθησης. Ο Microsoft Logistic Regression Algorithm μπορεί να θεωρηθεί και ως μία υποπερίπτωση του Microsoft Neural Network Algorithm καθώς χρησιμοποιεί παρόμοιες μεθόδους αλλά είναι πιο εύκολη η εκπαίδευση των δεδομένων.

Ένα πλεονέκτημα της Λογιστικής Παλινδρόμησης είναι ότι ο αλγόριθμος είναι εξαιρετικά ευέλικτος καθώς λαμβάνει οποιασδήποτε μορφής δεδομένα εισόδου και υποστηρίζει διαφορετικές εργασίες ανάλυσης:

- Μπορεί να χρησιμοποιήσει δημογραφικά στοιχεία για να προβλέψει αποτελέσματα όπως για παράδειγμα η πιθανότητα εμφάνισης μιας συγκεκριμένης ασθένειας.
- Μπορεί να εξερευνά και να αξιολογεί τους παράγοντες που συνεισφέρουν σ' ένα αποτέλεσμα. Για παράδειγμα να εντοπίσει εκείνες τις παραμέτρους που επηρεάζουν έναν πελάτη στο να επαναλάβει την επίσκεψή του σε ένα κατάστημα.
- Μπορεί να ταξινομεί έγγραφα, e-mail ή άλλα αντικείμενα με βάση διάφορα χαρακτηριστικά τους.

2.6.1. Παράδειγμα

Ας θεωρήσουμε ένα σύνολο ανθρώπων οι οποίοι μοιράζονται κοινά δημογραφικά χαρακτηριστικά και οι οποίοι αγοράζουν προϊόντα από την εταιρεία Adventure Works. Ταξινομώντας τα δεδομένα με βάση κάποιο συγκεκριμένο χαρακτηριστικό, όπως για παράδειγμα η αγορά ενός συγκεκριμένου προϊόντος, μπορούμε να συμπεράνουμε πως

τα διάφορα δημογραφικά χαρακτηριστικά ενός ατόμου σχετίζονται με την πιθανότητα να αγοράσει το συγκεκριμένο προϊόν.

2.6.2. Πως δουλεύει ο αλγόριθμος

Η Λογιστική Παλινδρόμηση είναι μία ευρέως γνωστή στατιστική μέθοδος για τον προσδιορισμό της συνεισφοράς πολλαπλών παραγόντων σε κάποιο αποτέλεσμα. Η εφαρμογή της στον Αλγόριθμο της Microsoft χρησιμοποιεί ένα κατάλληλα προσαρμοσμένο Νευρωνικό Δίκτυο για να προσδιορίσει τις συσχετίσεις μεταξύ των παραγόντων εισόδου και των αποτελεσμάτων στην έξοδο. Μετριέται η επιρροή κάθε παράγοντα και προσδιορίζονται τα βάρη της επιρροής όλων των παραγόντων στο τελικό μοντέλο. Ο όρος «Λογιστική Παλινδρόμηση» προκύπτει από το γεγονός ότι η καμπύλη των δεδομένων «συμπιέζεται» με τη μέθοδο του λογιστικού μετασχηματισμού, με σκοπό την ελαχιστοποίηση της επιρροής των ακραίων τιμών.

2.6.3. Απαιτούμενα δεδομένα για τα μοντέλα Λογιστικής Παλινδρόμησης

Όταν προετοιμάζουμε τα δεδομένα για χρήση στην εκπαίδευση ενός μοντέλου λογιστικής παλινδρόμησης, πρέπει να κατανοούμε τις απαιτήσεις για το συγκεκριμένο αλγόριθμο, όπως την ποσότητα των δεδομένων που χρειάζονται και το πώς τα δεδομένα χρησιμοποιούνται. Οι απαιτήσεις για ένα μοντέλο συσταδοποίησης είναι:

- **Μία στήλη μοναδικού κλειδιού** Κάθε μοντέλο πρέπει να περιέχει μία αριθμητική ή αλφαριθμητική στήλη η οποία ορίζει μοναδικά κάθε εγγραφή. Συνδυαστικά κλειδιά δεν επιτρέπονται.
- **Στήλες εισόδου δεδομένων** Κάθε μοντέλο πρέπει να περιέχει τουλάχιστον μία στήλη εισόδου η οποία θα περιέχει τις τιμές που θα χρησιμοποιηθούν ως παράγοντες κατά την ανάλυση. Μπορούμε να έχουμε όσες στήλες εισόδου επιθυμούμε, όμως ανάλογα με το πλήθος των τιμών κάθε στήλης, η προσθήκη επιπλέον στηλών μπορεί να αυξήσει τον απαιτούμενο χρόνο για την εκπαίδευση του μοντέλου.
- **Προς πρόβλεψη στήλη** Ο αλγόριθμος χρειάζεται τουλάχιστον μία προς πρόβλεψη στήλη, οποιουδήποτε τύπου δεδομένων, συμπεριλαμβανομένων και των συνεχών αριθμητικών τιμών. Οι τιμές της προς πρόβλεψης στήλης μπορούν

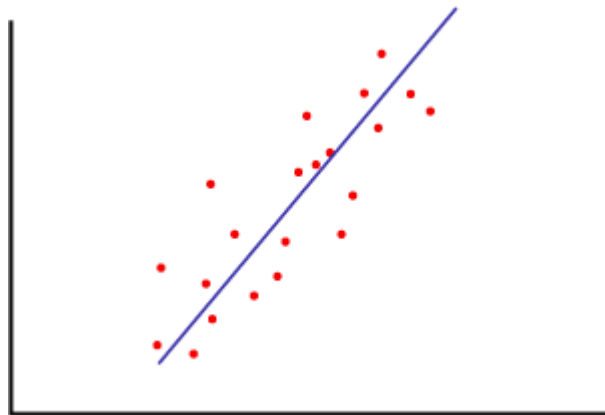
επίσης είτε να χρησιμοποιηθούν και σαν δεδομένα εισόδου στο μοντέλο είτε μπορούμε να ορίσουμε ότι θα χρησιμοποιηθούν μόνο για προβλέψεις.

[9]

2.7. Παρουσίαση Αλγορίθμου Γραμμικής Παλινδρόμησης - Microsoft Linear Regression Algorithm

Ο Microsoft Linear Regression Algorithm είναι μια παραλλαγή του Microsoft Decision Tree Algorithm που μας βοηθά να υπολογίσουμε μια γραμμική συσχέτιση μεταξύ μιας εξαρτημένης και μιας ανεξάρτητης μεταβλητής και στη συνέχεια να χρησιμοποιήσουμε αυτή τη συσχέτιση για προβλέψεις.

Η συσχέτιση παίρνει τη μορφή εξίσωσης της ευθείας που αντιπροσωπεύει καλύτερα μια σειρά δεδομένων. Για παράδειγμα η ευθεία στο ακόλουθο διάγραμμα είναι η καλύτερη δυνατή γραμμική απεικόνιση των δεδομένων.



Εικόνα 3 - Γραμμική απεικόνιση δεδομένων

Σε κάθε σημείο-δεδομένο στο διάγραμμα αντιστοιχεί ένα σφάλμα που σχετίζεται με την απόσταση του σημείου από την ευθεία. Οι συντελεστές a και b της εξίσωσης παλινδρόμησης προσδιορίζουν την κλίση και τη θέση της ευθείας παλινδρόμησης. Μπορούμε να βρούμε την εξίσωση παλινδρόμησης διορθώνοντας τα a και b έως ότου το άθροισμα των σφαλμάτων που αντιστοιχούν σε όλα τα σημεία φτάσει στην ελάχιστη τιμή του.

Υπάρχουν και άλλα είδη παλινδρόμησης τα οποία χρησιμοποιούν πολλαπλές μεταβλητές, καθώς και μη γραμμικές μέθοδοι παλινδρόμησης. Ωστόσο, η γραμμική παλινδρόμηση είναι μια χρήσιμη και ευρέως γνωστή μέθοδος για την μοντελοποίηση της αντίδρασης μιας αλλαγής σε κάποιον παράγοντα.

2.7.1. Παράδειγμα

Μπορούμε να χρησιμοποιήσουμε τη γραμμική παλινδρόμηση για να ορίσουμε μια συσχέτιση μεταξύ 2 συνεχών στηλών. Για παράδειγμα μπορούμε να χρησιμοποιήσουμε τη γραμμική παλινδρόμηση για να υπολογίσουμε την καμπύλη δεδομένων πωλήσεων ή παραγωγής. Μπορούμε επίσης να χρησιμοποιήσουμε τη γραμμική παλινδρόμηση σαν εργαλείο στην ανάπτυξη πιο πολύπλοκων μοντέλων εξόρυξης δεδομένων για να προσδιορίσουμε τις συσχετίσεις μεταξύ των στηλών.

Παρόλο που υπάρχουν πολλοί τρόποι να υπολογίσουμε τη γραμμική παλινδρόμηση χωρίς τη χρήση εργαλείων εξόρυξης δεδομένων, το πλεονέκτημα της χρήσης του Microsoft Linear Regression Algorithm για αυτό το πρόβλημα είναι ότι υπολογίζονται και ελέγχονται αυτόματα όλες οι πιθανές συσχετίσεις μεταξύ των μεταβλητών. Δε χρειάζεται να επιλέξουμε μέθοδο υπολογισμού, όπως των ελαχίστων τετραγώνων. Ωστόσο, η γραμμική παλινδρόμηση ενδέχεται να υπεραπλουστεύσει τις συσχετίσεις σε σενάρια όπου πολλαπλοί παράγοντες επηρεάζουν το αποτέλεσμα.

2.7.2. Πως δουλεύει ο αλγόριθμος

Ο Microsoft Linear Regression Algorithm είναι μία παραλλαγή του Microsoft Decision Tree Algorithm. Όταν επιλέγουμε τον Microsoft Linear Regression Algorithm, καλείται μία ειδική περίπτωση του Microsoft Decision Tree Algorithm με παραμέτρους που περιορίζουν τη συμπεριφορά του αλγορίθμου και απαιτούν συγκεκριμένο τύπο δεδομένων εισόδου. Επιπλέον, σε ένα μοντέλο γραμμικής παλινδρόμησης, ολόκληρο το σύνολο των δεδομένων χρησιμοποιείται για τον υπολογισμό των συσχετίσεων στο αρχικό πέρασμα, ενώ ένα κλασσικό Decision Tree Model διαχωρίζει τα δεδομένα συνεχώς σε μικρότερα υποσύνολα ή δέντρα.

2.7.3. Απαιτούμενα δεδομένα για τα μοντέλα Γραμμικής Παλινδρόμησης

Όταν προετοιμάζουμε τα δεδομένα για χρήση στην εκπαίδευση ενός μοντέλου γραμμικής παλινδρόμησης, πρέπει να κατανοούμε τις απαιτήσεις για το συγκεκριμένο αλγόριθμο, όπως την ποσότητα των δεδομένων που χρειάζονται και το πώς τα δεδομένα χρησιμοποιούνται.

Οι απαιτήσεις για ένα μοντέλο συσταδοποίησης είναι ως ακολούθως:

- **Μία στήλη μοναδικού κλειδιού** Κάθε μοντέλο πρέπει να περιέχει μία αριθμητική ή αλφαριθμητική στήλη η οποία ορίζει μοναδικά κάθε εγγραφή. Συνδυαστικά κλειδιά δεν επιτρέπονται.
- **Προβλέψιμη στήλη** Ο αλγόριθμος χρειάζεται τουλάχιστον μία προβλέψιμη στήλη. Μπορούμε να συμπεριλάβουμε πολλαπλά προβλέψιμα χαρακτηριστικά σ' ένα μοντέλο, θα πρέπει όμως τα ο τύπος δεδομένων των προβλέψιμων χαρακτηριστικών να είναι συνεχή αριθμητικά. Δε μπορούμε να χρησιμοποιήσουμε προβλέψιμα χαρακτηριστικά με τύπο δεδομένων ημερομηνίας και ώρας ακόμη και αν η αρχική αποθήκευση των δεδομένων είναι αριθμητική.
- **Στήλες εισόδου δεδομένων** Οι στήλες εισόδου πρέπει να περιέχουν συνεχή αριθμητικά δεδομένα και να τους έχει ανατεθεί ο κατάλληλος τύπος δεδομένων.

3. Εκτέλεση παραδειγμάτων

3.1. Εισαγωγή

Εκτελέστηκαν και μελετήθηκαν τα παραδείγματα: «*Predict ski rental with linear regression with SQL machine learning*» [11], «*Categorizing customers using k-means clustering with SQL machine learning*» [12] και «*Predict NYC taxi fares with binary classification*» [13] που έχουν αναπτυχθεί από τη Microsoft. Παρακάτω επιχειρείται μια περιγραφή των βημάτων που ακολουθήθηκαν για τα 3 παραδείγματα.

3.2. «Predict ski rental with linear regression with SQL machine learning»

3.2.1. Παρουσίαση προβλήματος

Στο συγκεκριμένο παράδειγμα χρησιμοποιούνται τα δεδομένα μιας εταιρείας ενοικίασης εξοπλισμού σκι. Αναπτύσσεται και χρησιμοποιείται ένα μοντέλο γραμμικής παλινδρόμησης (linear regression) για την πρόβλεψη των μελλοντικών ενοικιάσεων. Σκοπός της ανάλυσης είναι η καλύτερη οργάνωση και προετοιμασία του εξοπλισμού, του προσωπικού και των εγκαταστάσεων της εταιρείας, ανάλογα με τις αναμενόμενες μελλοντικές ενοικιάσεις.

3.2.2. Σύνδεση με τη Βάση Δεδομένων

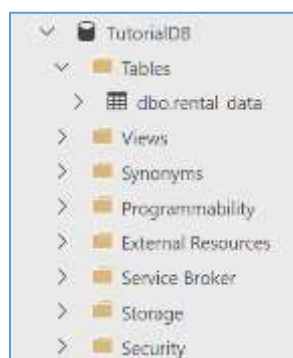
Αρχικά εισάγονται στο Azure Data Studio οι κάτωθι βιβλιοθήκες της Python:

- pandas
- pyodbc
- sklearn

οι οποίες περιέχουν τις συναρτήσεις που θα χρησιμοποιηθούν κατά την ανάλυση. Η φόρτωση των πακέτων – βιβλιοθηκών γίνεται μέσω της επιλογής Manage Packages που εμφανίζεται όταν ανοίξουμε ένα Notebook σε Python.

Στη συνέχεια εισάγεται η Βάση Δεδομένων που θα χρησιμοποιηθεί στο συγκεκριμένο παράδειγμα και βρίσκεται αποθηκευμένη στον υπολογιστή ως backup αρχείο Βάσεων Δεδομένων με όνομα TutorialDB.bak. Η εισαγωγή γίνεται μέσω της επιλογής Restore του Azure Data Studio, αλλάζοντας την επιλογή του πεδίου «Restore from:» από Database σε Backup File και κατόπιν αναζήτηση και επιλογή του αρχείου .bak από το φάκελο στον οποίο είναι αποθηκευμένο.

Με δεξί κλικ πάνω στην κατηγορία Databases στο βοηθητικό παράθυρο του Azure επιλέγεται η ανανέωση της λίστας των Β.Δ. (Refresh) ώστε να ελεγχθεί ότι η Β.Δ έχει όντως εισαχθεί σωστά στο λογισμικό. Κατόπιν με επιλογή στο αριστερό βελάκι της TutorialDB.bak ανοίγει νέο μενού που εμφανίζει τα περιεχόμενα της Β.Δ.. Στο φάκελο με όνομα Tables μπορούμε να δούμε τους πίνακες που εμπεριέχονται στη Β.Δ.:



Εικόνα 4 - Εμφάνιση πινάκων που περιέχει η Β.Δ.

Παρατηρούμε ότι η Β.Δ. TutorialDB.bak περιλαμβάνει μόνο έναν πίνακα με όνομα dbo.rental_data. Ανοίγοντας ένα Notebook σε SQL και εκτελώντας ένα ερώτημα (query) μπορούμε να δούμε τα δεδομένα του πίνακα dbo.rental_data:

	Year	Month	Day	RentalCount	WeekDay	Holiday	Snow	FHoliday	FSnow	FWeekDay
1	2014	1	20	445	2	1	0	1	0	2
2	2014	2	13	40	5	0	0	0	0	5
3	2013	3	10	456	1	0	0	0	0	1
4	2014	3	31	38	2	0	0	0	0	2
5	2014	4	24	23	5	0	0	0	0	5

Εικόνα 5 - Δεδομένα του πίνακα dbo.rental_data

3.2.3. Προετοιμασία των δεδομένων

Σε αυτό το βήμα γίνεται επεξεργασία των δεδομένων της ΒΔ μέσω της Python. Αρχικά φορτώνονται τα δεδομένα σε ένα data frame του pandas και στη συνέχεια γίνεται μια προεργασία τους αφαιρώντας κάποιες στήλες.

Ξεκινάμε ανοίγοντας ένα Notebook σε Python και εισάγουμε τις βιβλιοθήκες που περιέχουν τις συναρτήσεις οι οποίες θα μας χρειαστούν στο συγκεκριμένο παράδειγμα.

Εισάγεται μία μεταβλητή με όνομα **df** η οποία είναι τύπου πλαίσιο δεδομένων (data frame) και στην οποία θα αποθηκευτούν τα δεδομένα της database. Για τη φόρτωση των δεδομένων στη df χρησιμοποιείται η συνάρτηση **read_sql** (**sql=** , **con=**) της βιβλιοθήκης pandas. Η read_sql παίρνει τα κάτωθι δύο ορίσματα:

con: μεταβλητή τύπου string η οποία αποτελεί τη «σύνδεση» με τη database από την οποία θα αντληθούν τα δεδομένα καθώς και με τον server στον οποίο αυτή βρίσκεται. Εντός της con, εκτός του Server και της Database, ορίζονται και άλλες παράμετροι απαραίτητες ή μη, κατά περίπτωση.

sql: μεταβλητή τύπου string που περιλαμβάνει ένα ερώτημα (query) σε SQL το οποίο επιστρέφει μόνο τα δεδομένα στα οποία επιθυμούμε να δουλέψουμε.

Εισάγεται μία μεταβλητή τύπου λίστας με όνομα **columns** στην οποία αποθηκεύονται τα ονόματα των στηλών του data frame που δημιουργήσαμε. Η εισαγωγή των δεδομένων γίνεται με χρήση της συνάρτησης του pandas **df.columns.tolist** όπου **df** είναι το data frame, **columns** συνάρτηση που επιστρέφει τα ονόματα των στηλών του **df** και **tolist** η εντολή που αποθηκεύει τα δεδομένα σε μορφή λίστας. Στη συνέχεια φιλτράρονται τα περιεχόμενα της λίστας **columns** ώστε να αφαιρεθεί η στήλη «Year».

Για επαλήθευση των περιεχομένων της λίστας τυπώνουμε σε κάθε βήμα τη μεταβλητή **columns** και λαμβάνουμε τα εξής αποτελέσματα:

```
Αρχική λίστα 'columns':  ['Year', 'Month', 'Day',  
                          'Rentalcount', 'Weekday', 'Holiday', 'Snow']  
Φιλτραρισμένη λίστα 'columns':  ['Month', 'Day',  
                                  'Rentalcount', 'Weekday', 'Holiday', 'Snow']
```

3.2.4. «Εκπαίδευση» των δεδομένων

Σε αυτό το στάδιο εκπαιδεύουμε το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model) και γίνονται προβλέψεις για τα δεδομένα που μας ενδιαφέρουν με χρήση του μοντέλου.

Εισάγεται μία μεταβλητή τύπου string με όνομα **target** στην οποία αποθηκεύεται το όνομα της στήλης «**Rentalcount**» για την οποία επιθυμούμε να κάνουμε προβλέψεις.

Εισάγεται μεταβλητή με όνομα **train** στην οποία θα αποθηκευτεί το σύνολο των προς εκπαίδευση δεδομένων (**training set**). Μέσω της συνάρτησης `df.sample` επιλέγεται τυχαίο δείγμα των συνολικών δεδομένων του data frame που μας ενδιαφέρει. Η **sample** λαμβάνει τα κάτωθι δύο ορίσματα:

frac: μεταβλητή τύπου float η οποία εκφράζει το ποσοστό των συνολικών δεδομένων που επιλέγονται και λαμβάνει δεκαδικές τιμές από 0 έως και 1. Στο συγκεκριμένο παράδειγμα επιλέγεται η τιμή 0,8 που σημαίνει ότι το training set θα περιέχει το 80% του συνολικού data set.

random_state: μεταβλητή τύπου int που ορίζει ότι το δείγμα που επιλέγεται από το σύνολο των δεδομένων θα είναι σε κάθε εκτέλεση το ίδιο, εφόσον δεν αλλάζει η τιμή του random_state. Εάν δεν οριστεί καμία τιμή, τότε σε κάθε εκτέλεση η εντολή επιλέγει διαφορετικό δείγμα με τυχαίο τρόπο. Εφόσον οριστεί κάποια ακέραια τιμή, τότε σε κάθε εκτέλεση λαμβάνεται το ίδιο δείγμα που έχει αντιστοιχηθεί στη συγκεκριμένη τιμή κατά την πρώτη εκτέλεση, για την εξασφάλιση της επαναληψιμότητας της διαδικασίας.

Εισάγεται μεταβλητή με όνομα **test** στην οποία θα αποθηκευτεί το σύνολο των υπόλοιπων δεδομένων (**testing set**) που δεν επιλέχθηκαν για το training set.

Για επαλήθευση και καλύτερη εποπτεία τυπώνονται, μέσω της συνάρτησης `shape`, οι διαστάσεις των δύο υποσυνόλων training set και testing set καθώς και του αρχικού df και λαμβάνουμε τα εξής αποτελέσματα:

```
Training set shape: (362, 7)
Testing set shape: (91, 7)
df set shape: (453, 7)
```

Εισάγεται η μεταβλητή **lin_model**, στην οποία αποθηκεύεται η κλάση (class) **LinearRegression()** της βιβλιοθήκης Scikit-Learn και στην οποία θα αποθηκευτεί το μοντέλο γραμμικής παλινδρόμησης που θα χρησιμοποιηθεί για την ανάλυση των δεδομένων και καλείται η μέθοδος **fit** της κλάσης LinearRegression() για την εισαγωγή των επιθυμητών δεδομένων. Η μέθοδος fit λαμβάνει τις κάτωθι παραμέτρους **X** και **Y**:

X: το σύνολο των προς εκπαίδευση δεδομένων (training data set)

Y: η προς πρόβλεψη στήλη (target values)

Εκτελείται η μέθοδος **predict** της κλάσης (class) **LinearRegression()**, η οποία λαμβάνει ως όρισμα τις προς πρόβλεψη στήλες του test set και επιστρέφει τις προβλεφθείσες τιμές, οι οποίες αποθηκεύονται στη μεταβλητή **lin_predictions**.

Στη συνέχεια καλείται η συνάρτηση **mean_squared_error** που υπολογίζει το σφάλμα μεταξύ των αποτελεσμάτων της πρόβλεψης σε σχέση με τις πραγματικές τιμές του test set.

Εκτυπώνοντας τις τιμές της πρόβλεψης καθώς και το σφάλμα που προκύπτει λαμβάνουμε τα κάτωθι αποτελέσματα:

```
Predictions: [ 40.  38. 240.  39. 514.  48. 297.  25. 507.
 24.  30.  54.  40.  26.
  30.  34.  42. 390. 336.  37.  22.  35.  55. 350. 252. 370.
499.  48.
  37. 494.  46.  25. 312. 390.  35.  35. 421.  39. 176.  21.
33. 452.
  34.  28.  37. 260.  49. 577. 312.  24.  24. 390.  34.  64.
26.  32.
  33. 358. 348.  25.  35.  48.  39.  44.  58.  24. 350. 651.
38. 468.
  26.  42. 310. 709. 155.  26. 648. 617.  26. 846. 729.  44.
432.  25.
  39.  28. 325.  46.  36.  50.  63.]
Computed error: 1.4410460238917318e-26
```

3.2.5. Δημιουργία αποθηκευμένης διαδικασίας (Stored Procedure) που παράγει το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model)

Σε αυτό το στάδιο ενσωματώνουμε το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model) που αναπτύξαμε σε Python σε μια Βάση Δεδομένων του SQL Server με χρήση του Machine Learning Services.

- Δημιουργούμε μία αποθηκευμένη διαδικασία (stored procedure) η οποία κατασκευάζει το μοντέλο μηχανικής μάθησης (machine learning model).
- Αποθηκεύουμε το μοντέλο σε έναν πίνακα της βάσης δεδομένων.
- Δημιουργούμε μία αποθηκευμένη διαδικασία η οποία κάνει προβλέψεις με χρήση του μοντέλου.
- Εκτελούμε το μοντέλο με καινούρια δεδομένα.

Η διαδικασία που θα δημιουργήσουμε θα έχει όνομα `generate_rental_py_model`. Αρχικά με τη συνάρτηση `DROP` της SQL διαγράφεται τυχόν προηγούμενη αποθηκευμένη διαδικασία με το ίδιο όνομα και στη συνέχεια με τη συνάρτηση `CREATE` δημιουργείται η διαδικασία.

Το πρώτο βήμα είναι να εκτελεστεί η εντολή `sp_execute_external_script` μέσω της οποίας γίνεται η σύνδεση της SQL με τον κώδικα που περιγράφει τη διαδικασία. Η εντολή δέχεται τις κάτωθι παραμέτρους:

@language: η γλώσσα Python

@script: ο κώδικας σε Python που αναπτύχθηκε στο στάδιο της εκπαίδευσης των δεδομένων

@input_data_1: τα δεδομένα για τα οποία επιθυμούμε να γίνουν προβλέψεις, τα οποία εισάγονται μέσω ενός ερωτήματος της SQL

@input_data_1_name: το όνομα με το οποίο θα αποθηκευτούν τα δεδομένα

@params: το μοντέλο που παράγεται

@trained_model: το μοντέλο που παράγεται

3.2.6. Αποθήκευση του Linear Regression Model σε ένα νέο πίνακα της Βάσης Δεδομένων

Ο πίνακας που θα δημιουργήσουμε εντός της βάσης δεδομένων για την αποθήκευση του μοντέλου θα έχει όνομα `rental_py_models`. Αρχικά με τη συνάρτηση `DROP` της `SQL` διαγράφεται τυχόν προηγούμενος αποθηκευμένος πίνακας με το ίδιο όνομα και στη συνέχεια με τη συνάρτηση `CREATE` δημιουργείται ο νέος πίνακας.

Αποθηκεύουμε το μοντέλο στο νέο πίνακα ως αντικείμενο δυαδικών δεδομένων (binary object) με όνομα `linear_model`.

3.2.7. Δημιουργία αποθηκευμένης διαδικασίας που κάνει προβλέψεις για τα δεδομένα

Στο τελευταίο στάδιο του παραδείγματος δημιουργούμε μια αποθηκευμένη διαδικασία με όνομα `py_predict_rentalcount` η οποία κάνει προβλέψεις με χρήση του εκπαιδευμένου μοντέλου και ενός νέου συνόλου δεδομένων.

Δημιουργούμε έναν πίνακα με όνομα `py_rental_predictions` για την αποθήκευση των προβλέψεων.

Εκτελούμε την αποθηκευμένη διαδικασία για να προβλέψουμε τις τιμές της στήλης `rental_count`.

Για να δούμε τις προβλέψεις εμφανίζουμε μέσω ενός `query` όλα τα περιεχόμενα περιεχόμενα του πίνακα `py_rental_predictions` και λαμβάνουμε τις κάτωθι τιμές:

	RentalCount_Predicted	RentalCount_Actual	Month	Day	WeekDay	Snow	Holiday	Year
1	42	42	2	11	4	0	0	2015
2	360	360	3	29	1	0	0	2015
3	20	20	4	22	4	0	0	2015
4	42	42	3	6	6	0	0	2015
5	405	405	2	28	7	1	0	2015
6	38	38	1	12	2	1	0	2015
7	327	327	1	24	7	0	0	2015

Πίνακας 1 - Περιεχόμενα Πίνακα `py_rental_predictions`

3.3. «Categorizing customers using k-means clustering with SQL machine learning»

3.3.1. Παρουσίαση προβλήματος

Στο συγκεκριμένο παράδειγμα χρησιμοποιούνται τα δεδομένα μιας εταιρείας λιανικού εμπορίου. Αναπτύσσεται και χρησιμοποιείται ένα μοντέλο συσταδοποίησης (Clustering) με τον αλγόριθμο K-means για την κατηγοριοποίηση των πελατών της εταιρείας. Σκοπός της ανάλυσης είναι η βελτίωση της αποτελεσματικότητας των προωθητικών ενεργειών της εταιρείας, μέσω της στόχευσής τους σε συγκεκριμένες ομάδες πελατών.

3.3.2. Σύνδεση με τη Βάση Δεδομένων

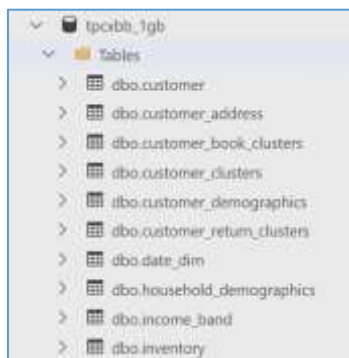
Αρχικά εισάγονται στο Azure Data Studio οι κάτωθι βιβλιοθήκες της Python:

- matplotlib
- pandas
- pyodbc
- scipy
- sklearn

οι οποίες περιέχουν τις συναρτήσεις που θα χρησιμοποιηθούν κατά την ανάλυση. Η φόρτωση των πακέτων – βιβλιοθηκών γίνεται μέσω της επιλογής Manage Packages που εμφανίζεται όταν ανοίξουμε ένα Notebook σε Python. Σημειώνεται ότι για τις βιβλιοθήκες pandas, pyodbc και sklearn δεν απαιτείται εκ νέου εισαγωγή τους καθώς έχουν φορτωθεί στο προηγούμενο παράδειγμα.

Στη συνέχεια εισάγεται η Βάση Δεδομένων που θα χρησιμοποιηθεί στο συγκεκριμένο παράδειγμα και βρίσκεται αποθηκευμένη στον υπολογιστή ως backup αρχείο Βάσεων Δεδομένων με όνομα trcxbb_1gb.bak, όπως και στο προηγούμενο παράδειγμα.

Στον φάκελο με όνομα Tables μπορούμε να δούμε τους πίνακες που εμπεριέχονται στη Β.Δ.. Στην επόμενη εικόνα απεικονίζονται οι 10 πρώτοι πίνακες της Β.Δ.:



Εικόνα 6 - Εμφάνιση πινάκων που περιέχει η Β.Δ.

Παρατηρούμε ότι η Β.Δ. tpcxb1_1gb.bak περιλαμβάνει πολλαπλούς πίνακες. Ανοίγοντας ένα Notebook σε SQL και εκτελώντας το κάτωθι ερώτημα (query) μπορούμε να δούμε τα δεδομένα του πίνακα dbo.customer:

	c_customer_sk	c_customer_id	c_current_demo_sk	c_current_hdemo_sk	c_current_addr_sk	c_first_shipt
1	0	AAAAAAAAAAAAAAAA	1824793	3203	2555	28776
2	1	AAAAAAAAAAAAAAAAB	830976	2600	9191	94858
3	2	AAAAAAAAAAAAAAAAC	335540	5527	14091	31783
4	3	AAAAAAAAAAAAAAAAD	1056662	1978	19122	8741

Πίνακας 2 - Περιεχόμενα Πίνακα dbo.customer

3.3.3. Προετοιμασία των δεδομένων

Σε αυτό το βήμα γίνεται επεξεργασία των δεδομένων της ΒΔ με χρήση της γλώσσας προγραμματισμού Python. Αρχικά διαχωρίζονται οι πελάτες (customers) σε διαφορετικές διαστάσεις και φορτώνονται τα δεδομένα σε ένα data frame του pandas και στη συνέχεια γίνεται μια προεργασία τους αφαιρώντας κάποιες στήλες.

Ξεκινάμε ανοίγοντας ένα Notebook σε Python και εισάγουμε τις βιβλιοθήκες pyodbc, matplotlib, numpy, pandas, scipy και sklearn που περιέχουν τις συναρτήσεις οι οποίες θα μας χρειαστούν στο συγκεκριμένο παράδειγμα.

Μέσω της συνάρτησης `read_sql(sql=, con=)` της βιβλιοθήκης pandas, όπως και στο προηγούμενο παράδειγμα, γίνεται η σύνδεση της Python με τη Β.Δ. tpcxb1_1gb.bak.

Αρχικά δημιουργούμε ένα νέο πίνακα με όνομα **customer_data**, στον οποίο θα αποθηκευτούν τα δεδομένα που θα προκύψουν από τους πίνακες **store_sales** και **store_returns** της Β.Δ., με την εκτέλεση του κάτωθι query σε SQL.

Από τον πίνακα **store_sales** με ομαδοποίηση ανά πελάτη (GROUP BY **ss_customer_sk**) καταμετρώνται το πλήθος των αγορών (**ss_ticket_number**) και των προϊόντων που αγοράστηκαν (**ss_item_sk**) με την εντολή COUNT και αποθηκεύονται ως **orders_count** και **orders_items** αντίστοιχα, ενώ τα χρηματικά ποσά (**ss_net_paid**) των αγορών κάθε πελάτη αθροίζονται με την εντολή SUM και αποθηκεύονται ως **orders_money**.

Από τον πίνακα **store_returns** και πάλι με ομαδοποίηση ανά πελάτη (GROUP BY **sr_customer_sk**) καταμετρώνται το πλήθος των επιστροφών (**sr_ticket_number**) και των προϊόντων που επιστράφησαν (**sr_item_sk**) με την εντολή COUNT και αποθηκεύονται ως **returns_count** και **returns_items** αντίστοιχα, ενώ τα χρηματικά ποσά (**sr_net_paid**) των επιστροφών κάθε πελάτη αθροίζονται με την εντολή SUM και αποθηκεύονται ως **returns_money**.

Τα αποτελέσματα συνενώνονται με την εντολή LEFT OUTER JOIN ανά κοινό πελάτη με τη συνθήκη **ss_customer_sk = sr_customer_sk** και στη συνέχεια διαιρούνται τα δεδομένα των επιστροφών, **returns_count**, **returns_items** και **returns_money**, με τα αντίστοιχα δεδομένα των αγορών, **orders_count**, **orders_items** και **orders_money**, ώστε να προκύψουν τα ποσοστά επιστροφών/αγορών, τα οποία αποθηκεύονται στις κάτωθι στήλες:

- **orderRatio** = ποσοστό επιστροφών (συνολικός αριθμός αγορών που επιστράφηκαν μερικώς ή ολικώς προς το συνολικό αριθμό αγορών)
- **itemsRatio** = ποσοστό επιστραφέντων προϊόντων (συνολικός αριθμός προϊόντων που επιστράφηκαν προς το συνολικό αριθμό προϊόντων που αγοράστηκαν)
- **monetaryRatio** = ποσοστό επιστραφέντων χρηματικών ποσών (συνολικό χρηματικό ποσό επιστροφών προς το συνολικό χρηματικό ποσό αγορών)
- **frequency** = συχνότητα επιστροφών

Για λόγους εποπτείας, τυπώνονται οι 5 πρώτες σειρές του πίνακα **customer_data**:

Data frame:	customer	orderRatio	itemsRatio	monetaryRatio	frequency
0	29727	0.000000	0.000000	0.000000	0
1	97643	0.068182	0.078176	0.037034	3
2	57247	0.000000	0.000000	0.000000	0
3	32549	0.086957	0.068657	0.031281	4
4	2040	0.000000	0.000000	0.000000	0

Πίνακας 3 - Περιεχόμενα Πίνακα customer_data

3.3.4. Δημιουργία μοντέλου συσταδοποίησης (clustering)

Στο επόμενο βήμα κατασκευάζουμε ένα μοντέλο συσταδοποίησης σε Python. Προσδιορίζεται ο αριθμός των συστάδων (clusters), εκτελείται η συσταδοποίηση και αναλύονται τα αποτελέσματα.

Για τη συσταδοποίηση των δεδομένων πελατών (customers) χρησιμοποιείται ο αλγόριθμος K-Means clustering, που είναι ένας από τους απλούστερους και γνωστότερους τρόπους ομαδοποίησης δεδομένων.

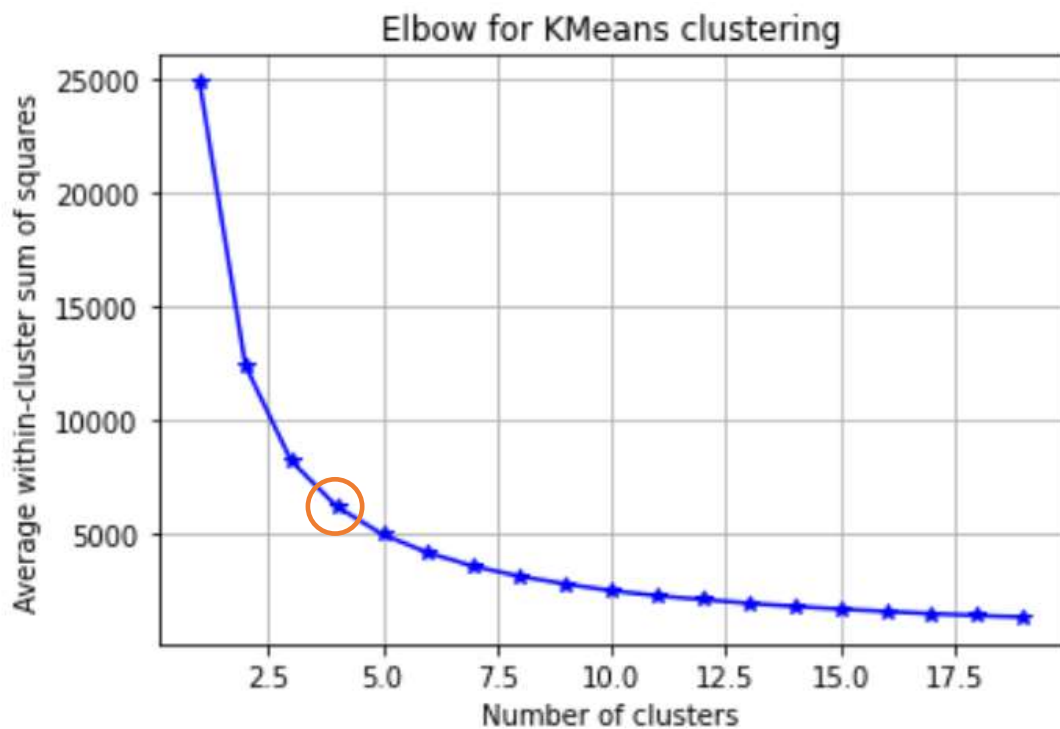
Ο αλγόριθμος λαμβάνει 2 ορίσματα εισόδου: Τα προς ανάλυση δεδομένα και έναν προκαθορισμένο αριθμό «K» που εκφράζει τον αριθμό των συστάδων που θα δημιουργηθούν. Το αποτέλεσμα είναι K συστάδες με τα δεδομένα εισόδου κατανεμημένα σε αυτές τις συστάδες.

Ο σκοπός του αλγορίθμου K-Means είναι να ομαδοποιήσει τα δεδομένα σε K συστάδες έτσι ώστε όλα τα δεδομένα της ίδιας συστάδας να είναι όσο περισσότερο όμοια μεταξύ τους και όσο περισσότερο διαφορετικά από τα δεδομένα των άλλων συστάδων γίνεται.

Για τον προσδιορισμό του αριθμού των συστάδων που θα χρησιμοποιήσει ο αλγόριθμος, χρησιμοποιούμε μια γραφική απεικόνιση των αθροισμάτων των τετραγώνων της Ευκλείδειας απόστασης των δεδομένων από τα κέντρα βάρους κάθε συστάδας.

Για τη δημιουργία της γραφικής παράστασης, εισάγεται η μεταβλητή **cdata** στην οποία αποθηκεύεται ο πίνακας customer_data που δημιουργήσαμε στο προηγούμενο βήμα. Για τις τιμές του K ορίζεται αρχικά ένα εύρος από 1 έως 19. Μέσω της κλάσης **cluster.KMeans** της βιβλιοθήκης sklearn καλείται η μέθοδος **fit** η οποία με όρισμα τα δεδομένα του πίνακα cdata εκτελεί τη συσταδοποίηση για κάθε τιμή του K.

Ο προσδιορισμός του αριθμού των συστάδων γίνεται οπτικά με χρήση της «Elbow method», κατά την οποία επιλέγεται ως καταλληλότερος αριθμός συστάδων η τιμή που βρίσκεται εγγύτερα στο «σπάσιμο» ή στον «αγκώνα» της καμπύλης της γραφικής παράστασης.



Εικόνα 7 - Γραφική παράσταση Elbow method for KMeans clustering

Με βάση τη γραφική παράσταση που προκύπτει στο συγκεκριμένο παράδειγμα παρατηρούμε ότι η τιμή του k που βρίσκεται εγγύτερα στον «αγκώνα» της καμπύλης είναι το $k = 4$, επομένως επιλέγεται να ομαδοποιηθούν τα δεδομένα σε 4 συστάδες.

Στη συνέχεια εκτελείται εκ νέου η μέθοδος **fit** της κλάσης **cluster.KMeans** για $k = 4$.

Τυπώνουμε τα μεγέθη των 4 clusters, δηλαδή το πλήθος των δεδομένων που κατανεμήθηκαν σε κάθε συστάδα:

```
Cluster0 (n=31675) :
Cluster1 (n=4989) :
Cluster2 (n=1) :
Cluster3 (n=671) :
```

Πίνακας 4 - Πλήθος δεδομένων ανά cluster

Και τέλος τυπώνουμε τις μέσες τιμές, ανά συστάδα, των στηλών που εκφράζουν τα ποσοστά αγορών/επιστροφών (orderRatio, itemsRatio, monetaryRatio, frequency).

cluster	customer	orderRatio	itemsRatio	monetaryRatio	frequency
0	50854.809882	0.000000	0.000000	0.000000	0.000000
1	51332.535779	0.721604	0.453365	0.307721	1.097815
2	57044.000000	1.000000	2.000000	108.719154	1.000000
3	48516.023845	0.136277	0.078346	0.044497	4.271237

Πίνακας 5 - Μέσες τιμές ανά cluster

Από τη μελέτη των αποτελεσμάτων, μπορούν να βγουν κάποια πρώτα συμπεράσματα όπως για παράδειγμα ότι η συστάδα 0 φαίνεται να αφορά πελάτες που είναι ανενεργοί, ενώ η συστάδα 3 αφορά πελάτες που επιστρέφουν συχνότερα τα προϊόντα που αγοράζουν.

Τέτοιου είδους συμπεράσματα θα μπορούσαν να οδηγήσουν για παράδειγμα μια εταιρεία να επικεντρώσει τις προσπάθειες προώθησης των προϊόντων της στο ανενεργό μέχρι σήμερα γκρουπ πελατών της συστάδας 0.

3.3.5. Ενσωμάτωση του Clustering Model με SQL Machine Learning

Για να έχουμε τη δυνατότητα να εκτελούμε τη συσταδοποίηση ανά πάσα στιγμή, καθώς νέοι πελάτες προστίθενται συνεχώς στη Β.Δ., θα πρέπει να μπορούμε να καλούμε το τμήμα κώδικα (script) της Python από οποιαδήποτε εφαρμογή. Για να το επιτύχουμε αυτό ενσωματώνουμε το Python script ως αποθηκευμένη διαδικασία της SQL.

Σε αυτό το στάδιο θα δημιουργηθεί η αποθηκευμένη διαδικασία η οποία θα δημιουργεί το μοντέλο, θα εκτελεστεί η συσταδοποίηση εντός του server και θα αξιοποιηθούν οι πληροφορίες που εξάγονται από τα αποτελέσματα.

Αφού δημιουργήσαμε την αποθηκευμένη διαδικασία, θα δημιουργήσουμε ένα νέο πίνακα με όνομα **py_customer_clusters** εντός της Β.Δ., για την αποθήκευση των αποτελεσμάτων της συσταδοποίησης.

Κατόπιν εκτελούμε την αποθηκευμένη διαδικασία, αποθηκεύουμε τα αποτελέσματα στο νέο πίνακα `py_customer_clusters` και εκτυπώνουμε τα περιεχόμενά του για επαλήθευση.

	Customer	OrderRatio	itemsRatio	monetaryRatio	frequency	cluster
1	29727	0	0	0	0	0
2	97643	0,0681818	0,0781759	0,037034	3	3
3	57247	0	0	0	0	0
4	32549	0,0869565	0,0686567	0,031281	4	3
5	2040	0	0	0	0	0

Πίνακας 6 - Περιεχόμενα Πίνακα `py_customer_clusters`

Καθώς αποθηκεύσαμε τη διαδικασία συσταδοποίησης εντός της Β.Δ., μπορεί να εκτελέσει συσταδοποίηση των δεδομένων πελατών που είναι αποθηκευμένα εντός της ίδιας Β.Δ.. Η διαδικασία μπορεί να εκτελεστεί κάθε φορά που ενημερώνονται τα δεδομένα των πελατών και να χρησιμοποιηθούν τα επικαιροποιημένα αποτελέσματα της συσταδοποίησης.

Ας υποθέσουμε ότι θέλουμε να στείλουμε προωθητικά emails στους πελάτες της συστάδας 0, που αφορά τους ανενεργούς πελάτες. Ο ακόλουθος πίνακας εμφανίζει τις ηλεκτρονικές διευθύνσεις των πελατών της συστάδας 0.

	c_email_address	c_customer_sk
1	Juan.Ogden@dcemail.com ...	6429
2	Anna.Ayers@vfemail.net ...	6671
3	Olga.Mcintire@deadaddress.co...	7100
4	John.Munoz@mailvault.com ...	7216
5	Kevin.Jacobs@hush.ai ...	7318
6	Wanda.Pimentel@usa.com ...	7441
7	Philip.Daniels@lpemail.com ...	7478

Πίνακας 7 - Emails πελατών που ανήκουν στη συστάδα 0

3.4. «Predict NYC taxi fares with binary classification»

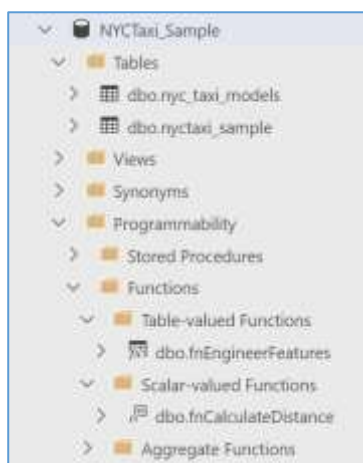
3.4.1. Παρουσίαση προβλήματος

Στο συγκεκριμένο παράδειγμα χρησιμοποιούνται τα δεδομένα μιας εταιρείας υπηρεσιών ταξί και πολυτελών αυτοκινήτων της Νέας Υόρκης. Αναπτύσσεται και χρησιμοποιείται ένα μοντέλο δυαδικής ταξινόμησης (binary classification) για την κατηγοριοποίηση των διαδρομών των οχημάτων της εταιρείας. Σκοπός της ανάλυσης είναι η πρόβλεψη της πιθανότητας πληρωμής φιλοδωρήματος από τον πελάτη, ανάλογα με την ώρα της ημέρας, το μήκος της διαδρομής και το σημείο αφετηρίας.

3.4.2. Σύνδεση με τη Βάση Δεδομένων και προεπισκόπηση των δεδομένων

Αρχικά εισάγεται η Βάση Δεδομένων που θα χρησιμοποιηθεί στο συγκεκριμένο παράδειγμα και βρίσκεται αποθηκευμένη στον υπολογιστή ως backup αρχείο Βάσεων Δεδομένων με όνομα NYCTaxi_Sample.bak, όπως και στα προηγούμενα παραδείγματα.

Στο φάκελο με όνομα Tables μπορούμε να δούμε τους πίνακες που εμπεριέχονται στη Β.Δ., ενώ στο φάκελο Programmability -> Functions μπορούμε να δούμε τις περιεχόμενες συναρτήσεις:



Εικόνα 8 - Εμφάνιση πινάκων και συναρτήσεων της Β.Δ.

3.4.3. Εξερεύνηση και οπτικοποίηση των δεδομένων

Αρχικά έχουμε τα αναγνωριστικά των οχημάτων:

Η στήλη **medallion** περιέχει το ID number κάθε οχήματος

Η στήλη **hack_license** περιέχει τον αριθμό αδείας κάθε οδηγού, ανώνυμα.

Κάθε εγγραφή διαδρομής περιλαμβάνει το σημείο και το χρόνο έναρξης και λήξης της διαδρομής καθώς και τη διανυόμενη απόσταση.

Κάθε εγγραφή χρέωσης περιλαμβάνει πληροφορίες πληρωμής, όπως η μέθοδος πληρωμής, το συνολικό ποσό και το ποσό του φιλοδώρηματος.

Οι τελευταίες 3 στήλες μπορούν να χρησιμοποιηθούν για διάφορες εργασίες μηχανικής μάθησης. Η στήλη **tip_amount** περιέχει συνεχείς αριθμητικές τιμές και μπορεί να χρησιμοποιηθεί σαν **label** στήλη για ανάλυση παλινδρόμησης. Η στήλη **tipped** περιέχει μόνο yes/no τιμές και μπορεί να χρησιμοποιηθεί για δυαδική ταξινόμηση. Η στήλη **tip_class** περιέχει πολλαπλές κατηγορίες για τα φιλοδωρήματα ανάλογα με το ποσό και μπορεί να χρησιμοποιηθεί για ανάλυση ταξινόμησης πολλαπλών κατηγοριών.

Οι τιμές που χρησιμοποιούνται στις στήλες **tipped** και **tip_class** είναι βασισμένες στη στήλη **tip_amount** με βάση τους κάτωθι κανόνες:

- Η στήλη **tipped** λαμβάνει τιμές 0 και 1:

Αν ισχύει $\text{tip_amount} > 0$, τότε $\text{tipped} = 1$, διαφορετικά $\text{tipped} = 0$

- Η στήλη **tip_class** λαμβάνει τιμές από 0 έως 4:

Κατηγορία 0: $\text{tip_amount} = 0\$$

Κατηγορία 1: $0\$ < \text{tip_amount} \leq 5\$$

Κατηγορία 2: $5\$ < \text{tip_amount} \leq 10\$$

Κατηγορία 3: $10\$ < \text{tip_amount} \leq 20\$$

Κατηγορία 4: $\text{tip_amount} > 20\$$

3.4.4. Δημιουργία γραφημάτων με χρήση της Python μέσω Transact-SQL.

Η ανάπτυξη λύσεων της επιστήμης των δεδομένων περιλαμβάνει εκτεταμένη εξερεύνηση και απεικόνιση των δεδομένων. Καθώς η απεικόνιση είναι τόσο σημαντικό εργαλείο για την κατανόηση της κατανομής των δεδομένων, η γλώσσα Python παρέχει αρκετές βιβλιοθήκες για την απεικόνιση. Η βιβλιοθήκη matplotlib είναι από τις δημοφιλείς για απεικόνιση δεδομένων και περιλαμβάνει συναρτήσεις για τη δημιουργία ιστογραμμάτων, γραφημάτων διασποράς (scatter plots) , θηκογραμμάτων (box plots) κ.ά..

Η Python δίνει τη δυνατότητα δημιουργίας γραφήματος μέσω αποθηκευμένων διαδικασιών και αποθήκευσής τους ως αντικείμενα της Python με τύπο δεδομένων **varbinary** και κατόπιν εγγραφή σε αρχείο που μπορεί να μεταφερθεί και να χρησιμοποιηθεί και από άλλη πλατφόρμα.

Η stored_procedure επιστρέφει ένα σειριακό αντικείμενο γραφήματος της Python σαν σειρά δυαδικών δεδομένων. Για την απεικόνιση των δεδομένων σε γράφημα χρησιμοποιούμε κώδικα σε Python ο οποίος επιστρέφει ένα αρχείο εικόνας το οποίο μπορούμε να αποθηκεύσουμε.

Δημιουργούμε την αποθηκευμένη διαδικασία PyPlotMatplotlib:

- Η μεταβλητή @query λαμβάνει ως τιμή τη συμβολοσειρά SELECT tipped FROM nyc taxi_sample και χρησιμοποιείται ως όρισμα της μεταβλητής @input_data_1 ώστε να περαστεί στην Python το ερώτημα που επιστρέφει τα επιθυμητά αποτελέσματα.
- Ο κώδικας σε Python δημιουργεί το ιστόγραμμα και το διάγραμμα διασποράς με χρήση των αντικειμένων εικόνας της βιβλιοθήκης matplotlib τα οποία στη συνέχεια μετατρέπονται σε σειριακά αντικείμενα μέσω της βιβλιοθήκης pickle.
- Το γραφικό αντικείμενο της Python μετατρέπεται σε πλαίσιο δεδομένων της βιβλιοθήκης Pandas.

Στη συνέχεια εκτελούμε την αποθηκευμένη διαδικασία χωρίς ορίσματα ώστε να δημιουργήσουμε το γράφημα.

```
USE NYCTaxi_Sample  
EXEC [dbo].[PyPlotMatplotlib]
```

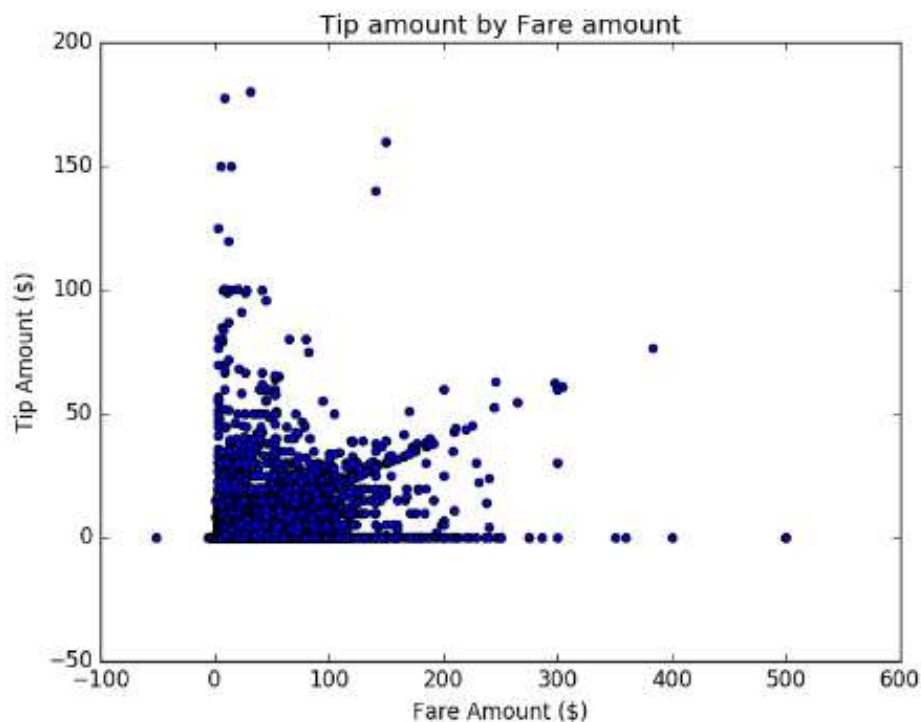
Τα αποτελέσματα εμφανίζονται στην κάτωθι μορφή:

```
plot  
0xFFD8FFE000104A4649...  
0xFFD8FFE000104A4649...  
0xFFD8FFE000104A4649...  
0xFFD8FFE000104A4649...
```

Από έναν Python client μπορούμε να συνδεθούμε στον SQL Server όπου δημιουργήθηκαν τα δυαδικά γραφικά αντικείμενα και να τα δούμε ως γραφικές απεικονίσεις. Εάν η σύνδεση είναι επιτυχής βλέπουμε το κάτωθι μήνυμα:

```
The plots are saved in directory: xxxx
```

Το αρχείο εξόδου έχει δημιουργηθεί στο φάκελο της Python. Πηγαίνοντας στο φάκελο και ανοίγοντας το αρχείο βλέπουμε το κάτωθι γράφημα:



Εικόνα 9 - Γράφημα με χρήση της Python μέσω Transact-SQL

3.4.5. Calculate trip Distance

Σε αυτό το στάδιο κατασκευάζουμε μία συνάρτηση για τον υπολογισμό της απόστασης κάθε διαδρομής και στη συνέχεια μία δεύτερη συνάρτηση για να σώσουμε τα αποτελέσματα αυτής.

Οι τιμές των αποστάσεων που υπάρχουν στα αρχικά δεδομένα βασίζονται στις δηλωθείσες αποστάσεις και δεν αντικατοπτρίζουν απαραίτητα τη γεωγραφική ή τη διανυθείσα απόσταση. Ως εκ τούτου πρέπει να υπολογίσουμε την απευθείας απόσταση μεταξύ των σημείων αφετηρίας και προορισμού, με χρήση των διαθέσιμων γεωγραφικών συντεταγμένων στο αρχικό σετ δεδομένων NYC Taxi. Αυτό θα γίνει με χρήση της εξίσωσης Haversine εντός μιας συνάρτησης της T-SQL.

Αρχικά κατασκευάζουμε τη συνάρτηση **fnCalculateDistance** για να υπολογίσουμε την απόσταση με χρήση της εξίσωσης Haversine και στη συνέχεια τη συνάρτηση **fnEngineerFeatures** για να δημιουργήσουμε έναν πίνακα που θα περιέχει τα αποτελέσματα.

Η συνάρτηση δέχεται ως είσοδο δύο τιμές, το γεωγραφικό πλάτος και μήκος στα σημεία αφετηρίας και προορισμού μιας διαδρομής και μέσω της εξίσωσης Haversine μετατρέπει τις γεωγραφικές συντεταγμένες σε ακτίνια και επιστρέφει μία τιμή που εκφράζει την απευθείας απόσταση σε μίλια μεταξύ των δύο σημείων.

Στη συνέχεια δημιουργούμε τη συνάρτηση **fnEngineerFeatures** η οποία λαμβάνει δεδομένα τύπου πίνακα ως είσοδο, δηλαδή με πολλαπλές στήλες και επιστρέφει αντίστοιχα πολλαπλές στήλες αποτελεσμάτων. Σκοπός της συνάρτησης είναι να δημιουργήσει ένα σύνολο αποτελεσμάτων για χρήση στην κατασκευή ενός μοντέλου. Λειτουργεί με κλήση της προηγούμενης συνάρτησης, **fnCalculateDistance**, ώστε να υπολογίσει τις απευθείας αποστάσεις μεταξύ των σημείων αφετηρίας και προορισμού.

Για την επαλήθευση της ορθής λειτουργίας της συνάρτησης μπορούμε να τη χρησιμοποιήσουμε για τον υπολογισμό της γεωγραφικής απόστασης των διαδρομών που εμφανίζουν μηδενική μετρούμενη απόσταση αλλά έχουν διαφορετικά σημεία αφετηρίας και προορισμού.

Όπως διαπιστώνουμε από τα αποτελέσματα, η δηλωθείσα απόσταση δε συμπίπτει πάντοτε με τη γεωγραφική.

3.4.6. Εκπαίδευση και αποθήκευση μοντέλου σε Python με χρήση T-SQL

Σε αυτό το στάδιο θα δημιουργήσουμε και θα εκπαιδεύσουμε ένα μοντέλο μηχανικής μάθησης μέσω αποθηκευμένων διαδικασιών στην SQL, με χρήση των βιβλιοθηκών scikit-learn και revoscalepy της Python και στη συνέχεια θα αποθηκεύσουμε το μοντέλο σε έναν πίνακα της SQL.

3.4.7. Διαχωρισμός των δεδομένων σε υποσύνολα εκπαίδευσης (training set) και ελέγχου (testing set)

Δημιουργούμε μία αποθηκευμένη διαδικασία με όνομα **PyTrainTestSplit** για το διαχωρισμό των δεδομένων του πίνακα `nyctaxi_sample` σε δύο υποσύνολα, το `nyctaxi_sample_training` και το `nyctaxi_sample_testing`.

Για το διαχωρισμό των δεδομένων κατά περίπτωση ανάλογα με τις ανάγκες της ανάλυσης, κατά την κλήση της συνάρτησης πληκτρολογούμε έναν ακέραιο αριθμό που εκφράζει το ποσοστό των αρχικών δεδομένων που θα τοποθετηθούν στο training set. Για παράδειγμα στην ακόλουθη εκτέλεση θα τοποθετηθεί στο training set το 60% του αρχικού συνόλου δεδομένων.

3.4.8. Κατασκευή μοντέλου λογιστικής παλινδρόμησης

Αφού έχει γίνει η προεπεξεργασία των δεδομένων, μπορούν να χρησιμοποιηθούν για την εκπαίδευση του μοντέλου, με κλήση μιας stored procedure, που τρέχει κώδικα σε Python και η οποία λαμβάνει ως είσοδο τον πίνακα που περιέχει τα δεδομένα εκπαίδευσης.

Θα δημιουργήσουμε δύο μοντέλα δυαδικής ταξινόμησης για την πρόβλεψη της πιθανότητας πληρωμής φιλοδωρήματος από τον πελάτη:

Το πρώτο μοντέλο θα δημιουργηθεί με χρήση της βιβλιοθήκης scikit-learn μέσω της stored procedure **PyTrainScikit**.

Το δεύτερο μοντέλο θα δημιουργηθεί με χρήση της βιβλιοθήκης revoscalepy μέσω της stored procedure **TrainTipPredictionModelRxPy**.

Η κάθε αποθηκευμένη διαδικασία χρησιμοποιεί τα δεδομένα εισόδου και δημιουργεί ένα μοντέλο λογιστικής παλινδρόμησης.

Για να είναι ευκολότερη η επανεκπαίδευση του μοντέλου με νέα δεδομένα, αποθηκεύουμε την κλήση της `sp_execute_external_script` εντός άλλης stored procedure την οποία εισάγουμε στο νέο training data set ως παράμετρο.

Αρχικά εκτελούμε τον κώδικα σε SQL και δημιουργούμε την stored procedure **PyTrainScikit** και στη συνέχεια εισάγουμε το εκπαιδευμένο μοντέλο στον πίνακα `nyc_taxi_models`.

Για επαλήθευση ανοίγουμε τον πίνακα `nyc_taxi_models` και παρατηρούμε ότι μία νέα σειρά έχει προστεθεί, η οποία στη στήλη `model` περιέχει το μοντέλο `SciKit_model`.

Στη συνέχεια εκτελούμε τον κώδικα σε SQL και δημιουργούμε την stored procedure `TrainTipPredictionModelRxPy`.

Το query σε SQL μέσω της εντολής `SELECT` εφαρμόζει τη συνάρτηση `fnCalculateDistance` για τον υπολογισμό της απευθείας απόστασης μεταξύ σημείων αφετηρίας και προορισμού. Τα αποτελέσματα αποθηκεύονται στη μεταβλητή εισόδου `InputDataset` της Python.

Η δυαδική μεταβλητή `tipped` χρησιμοποιείται ως όνομα της στήλης των αποτελεσμάτων και το μοντέλο χρησιμοποιεί τις στήλες `passenger_count`, `trip_distance`, `trip_time_in_secs`, and `direct_distance`.

Το εκπαιδευμένο μοντέλο αποθηκεύεται στη μεταβλητή `logitObj` της Python. Προσθέτοντας την εντολή `OUTPUT` της T-SQL μπορούμε να προσθέσουμε τη μεταβλητή σαν έξοδο της αποθηκευμένης διαδικασίας. Στο επόμενο βήμα η μεταβλητή θα χρησιμοποιηθεί για την εισαγωγή του δυαδικού κώδικα του μοντέλου στη Β.Δ. `nyc_taxi_models`. Με αυτό τον τρόπο διευκολύνεται η αποθήκευση και η επαναχρησιμοποίηση του μοντέλου.

Στη συνέχεια εισάγουμε το εκπαιδευμένο μοντέλο στον πίνακα `nyc_taxi_models`. Για επαλήθευση ανοίγουμε τον πίνακα `nyc_taxi_models` και παρατηρούμε ότι μία νέα σειρά έχει προστεθεί, η οποία στη στήλη `model` περιέχει το μοντέλο `TrainTipPredictionModelRxPy`.

3.4.9. Προβλέψεις με χρήση της Python εντός αποθηκευμένης διαδικασίας Run predictions using Python embedded in a stored procedure

Σε αυτό το στάδιο παρουσιάζονται δύο μέθοδοι δημιουργίας προβλέψεων με βάση ένα μοντέλο σε Python: η μέθοδος **batch scoring** («παράλληλη») και η μέθοδος **scoring row by row** («σειρά – σειρά»).

Batch scoring: Για την εισαγωγή πολλαπλών εγγραφών ως είσοδο στο μοντέλο χρησιμοποιούμε μια εντολή SELECT επί της αποθηκευμένης διαδικασίας. Το αποτέλεσμα είναι ένας πίνακας που αντιστοιχεί στις περιπτώσεις που ορίσαμε με το query.

Individual scoring: Χρησιμοποιούμε μια μοναδική παράμετρο ως είσοδο και λαμβάνουμε ως αποτέλεσμα μία μοναδική γραμμή ή τιμή.

Οι πρώτες δύο αποθηκευμένες διαδικασίες που δημιουργούνται στη συνέχεια απεικονίζουν τη βασική σύνταξη για την ενσωμάτωση της πρόβλεψης με χρήση Python, εντός της αποθηκευμένης διαδικασίας. Και οι δύο stored procedures λαμβάνουν έναν πίνακα ως είσοδο.

Το όνομα του επιθυμητού ανά περίπτωση μοντέλου παρέχεται ως παράμετρος εισόδου στην αποθηκευμένη διαδικασία, μέσω του σειριακού μοντέλου που βρίσκεται αποθηκευμένο στον πίνακα nyc_taxi_models.table το οποίο επιστρέφεται μέσω της εντολής SELECT στην αποθηκευμένη διαδικασία. Το σειριακό μοντέλο αποθηκεύεται στη μεταβλητή mod της Python για περαιτέρω επεξεργασία.

Οι νέες περιπτώσεις που πρέπει να μελετηθούν λαμβάνονται από το εισαγωγικό query @input_data_1 και καθώς γίνεται η ανάγνωσή του, οι εγγραφές αποθηκεύονται στο πλαίσιο δεδομένων InputDataSet.

Και οι δύο αποθηκευμένες διαδικασίες χρησιμοποιούν συναρτήσεις της βιβλιοθήκης sklearn για να υπολογίσουν ένα μέτρο ακρίβειας με όνομα AUC (area under curve).

Αρχικά δημιουργούμε την αποθηκευμένη διαδικασία **PredictTipSciKitPy** μέσω της T-SQL με χρήση της βιβλιοθήκης scikit-learn.

Τα δεδομένα εισόδου εισάγονται στη συνάρτηση predict_proba του μοντέλου Λογιστικής Παλινδρόμησης. Η συνάρτηση predict_proba (probArray =

`mod.predict_proba(X)` επιστρέφει ένα δεκαδικό αριθμό που εκφράζει την πιθανότητα να δοθεί φιλοδώρημα, ανεξάρτητα από το ποσό.

Στη συνέχεια η αποθηκευμένη διαδικασία **PredictTipRxPy** χρησιμοποιεί τα ίδια δεδομένα εισόδου και επιστρέφει ίδιου τύπου αποτελέσματα, με χρήση όμως της βιβλιοθήκης `renvoscalepy`.

Οι αποθηκευμένες διαδικασίες **PredictTipSciKitPy** και **PredictTipRxPy** λαμβάνουν δύο παραμέτρους:

1. Το `query` που επιστρέφει από το αρχικό σύνολο δεδομένων τα επιθυμητά δεδομένα για την πρόβλεψη.
2. Το όνομα του εκπαιδευμένου μοντέλου.

Εισάγοντας αυτά τα δύο ορίσματα στην αποθηκευμένη διαδικασία, μπορούμε να επιλέξουμε ένα συγκεκριμένο μοντέλο ή να αλλάξουμε τα δεδομένα που θα χρησιμοποιήσουμε για την πρόβλεψη.

Για να χρησιμοποιήσουμε το μοντέλο της βιβλιοθήκης `scikit-learn` για πρόβλεψη, καλούμε την αποθηκευμένη διαδικασία `PredictTipSciKitPy`, εισάγοντας το όνομα του επιθυμητού μοντέλου και το κατάλληλο `query` ως ορίσματα.

Η αποθηκευμένη διαδικασία επιστρέφει τις πιθανότητες πληρωμής φιλοδωρήματος για κάθε διαδρομή που εισήχθη στο μοντέλο με το συγκεκριμένο `query`.

Τα αποτελέσματα που λαμβάνουμε είναι ένας πίνακας με μία στήλη «score» που εκφράζει την πιθανότητα πληρωμής φιλοδωρήματος:

	Score
1	0,5077273971197059
2	0,5077328269866626
3	0,5077314035677836
4	0,5077358664088818
5	0,5077291607769071
6	0,5077287887287634
7	0,507728468117952
8	0,5077309721943294
9	0,507727077710949

Εικόνα 10 - Πιθανότητα πληρωμής φιλοδωρήματος - βιβλιοθήκη `scikit-learn`

Για να χρησιμοποιήσουμε το μοντέλο της βιβλιοθήκης *revoscalepy* για πρόβλεψη, καλούμε την αποθηκευμένη διαδικασία *PredictTipRxPy*, εισάγοντας το όνομα του επιθυμητού μοντέλου και το κατάλληλο *query* ως ορίσματα.

Ομοίως με την προηγούμενη περίπτωση, λαμβάνουμε τα αποτελέσματα σε μία στήλη όπου περιέχονται οι πιθανότητες («score») πληρωμής φιλοδωρήματος.

	Score
1	0,5085393211698432
2	0,5085447302790985
3	0,5085432547085337
4	0,508547740823516
5	0,5085411099689545
6	0,5085407148046122
7	0,5085404167645767
8	0,5085429044574837
9	0,5085390512334118
10	0,5085416804720436

Εικόνα 11 - Πιθανότητα πληρωμής φιλοδωρήματος - βιβλιοθήκη revoscalepy

Single-row scoring: Κάποιες φορές αντί για ομαδικές προβλέψεις επιθυμούμε να εισάγουμε μία μεμονωμένη περίπτωση, λαμβάνοντας τιμές από μία εφαρμογή και επιστρέφοντας ένα μοναδικό αποτέλεσμα βασισμένο σ' αυτές τις τιμές. Για παράδειγμα, μπορούμε να καλέσουμε την αποθηκευμένη διαδικασία μέσω ενός φύλλου Excel ή μιας web εφαρμογής και να εισάγουμε τις επιθυμητές τιμές.

Η αποθηκευμένη διαδικασία *PredictTipSingleModeSciKitPy* έχει σχεδιαστεί για προβλέψεις βασισμένες σε μία μοναδική γραμμή των δεδομένων, με χρήση του μοντέλου που αναπτύχθηκε με τη βιβλιοθήκη *scikit-learn*.

Η αποθηκευμένη διαδικασία *PredictTipSingleModeRxPy* έχει σχεδιαστεί για προβλέψεις βασισμένες σε μία μοναδική γραμμή των δεδομένων, με χρήση του μοντέλου που αναπτύχθηκε με τη βιβλιοθήκη *revoscalepy*.

3.4.10. Δημιουργία προβλέψεων με χρήση των μοντέλων

Μετά τη δημιουργία των αποθηκευμένων διαδικασιών είναι εύκολο να κάνουμε προβλέψεις με βάση όποιο μοντέλο επιθυμούμε. Ανοίγουμε ένα νέο παράθυρο δημιουργίας ερωτημάτων (Query window) και εισάγουμε τις επιθυμητές τιμές των παραμέτρων για τις στήλες που χρησιμοποιεί το μοντέλο. Για το εξεταζόμενο παράδειγμα οι 7 απαιτούμενες στήλες είναι οι κάτωθι:

passenger_count, trip_distance, trip_time_in_secs, pickup_latitude, pickup_longitude,
dropoff_latitude, dropoff_longitude

Μπορούμε να κάνουμε πρόβλεψη βάσει του μοντέλου που χρησιμοποιεί τη βιβλιοθήκη revoscalepy και του μοντέλου που χρησιμοποιεί τη βιβλιοθήκη scikit-learn.

Το αποτέλεσμα που λαμβάνουμε από τις 2 διαδικασίες, είναι η πιθανότητα πληρωμής φιλοδωρήματος για τη διαδρομή με τις συγκεκριμένες παραμέτρους που εισήχθησαν.

4. Εφαρμογή των μεθόδων εξόρυξης δεδομένων στο αμερικάνικο επαγγελματικό μπάσκετ (NBA)

4.1. Εισαγωγή

Βασιζόμενοι σε δεδομένα του επαγγελματικού αμερικανικού μπάσκετ (National Basketball Association – NBA) από πηγές όπως το <https://www.basketball-reference.com> θα επιχειρήσουμε μία εφαρμογή των μεθόδων, των εργαλείων και των αλγορίθμων που μελετήθηκαν στα προηγούμενα κεφάλαια για τον εντοπισμό προτύπων στα στατιστικά δεδομένα παικτών και ομάδων και τη δημιουργία προβλέψεων σχετικά με τα ατομικά και ομαδικά αποτελέσματα, με τη λογική των αποθηκευμένων διαδικασιών (stored procedures).

4.2. Δημιουργία βάσης δεδομένων και ανάλυση πινάκων.

Αρχικά μέσω της ιστοσελίδας <https://www.kaggle.com> επιλέγεται το σύνολο δεδομένων Season_Stats, που περιλαμβάνει στατιστικά δεδομένα απόδοσης των παικτών του NBA σε αρχικό σύνολο 68 αγωνιστικών σεζόν, από τη σεζόν 1949-1950 έως και τη σεζόν 2016-2017.

Μέσω του Microsoft SQL Server και του Azure Data Studio δημιουργούμε μία βάση δεδομένων με όνομα NBA.bak η οποία περιέχει σε αυτή τη φάση έναν μόνο πίνακα με όνομα Season_Stats, που αντιστοιχεί στο dataset που κατεβάσαμε.

Ανοίγοντας ένα query σε SQL μπορούμε να κάνουμε μία προεπισκόπηση των δεδομένων:

```
SELECT * FROM Seasons_Stats
```

	F1	Year	Player	Pos	Age	Tm	G	GS	MP	PER	TS%	3PAr	FT%	ORB%	DRB%	TRB%
1	18	1958	Charlie Black	F-C	28	AND	29	NULL	NULL	NULL	0,326	NULL	0,296	NULL	NULL	NULL
2	19	1958	Frankie Brian	G	26	AND	64	NULL	NULL	NULL	0,415	NULL	0,422	NULL	NULL	NULL
3	31	1958	Jake Carter	F-C	25	AND	11	NULL	NULL	NULL	0,454	NULL	0,9	NULL	NULL	NULL
4	37	1958	Bill Closs	SF	28	AND	64	NULL	NULL	NULL	0,372	NULL	0,288	NULL	NULL	NULL
5	73	1958	Frank Gates	G	29	AND	64	NULL	NULL	NULL	0,322	NULL	0,244	NULL	NULL	NULL
6	93	1958	John Hargis	G-F	29	AND	68	NULL	NULL	NULL	0,479	NULL	0,584	NULL	NULL	NULL
7	115	1958	Ralph Johnson	PG	28	AND	35	NULL	NULL	NULL	0,364	NULL	0,195	NULL	NULL	NULL
8	126	1958	Walt Kirk	PG	25	AND	26	NULL	NULL	NULL	0,368	NULL	0,664	NULL	NULL	NULL
9	133	1958	Milo Komenich	F-C	29	AND	64	NULL	NULL	NULL	0,326	NULL	0,29	NULL	NULL	NULL
10	175	1958	Murray Mitchell	C	26	AND	2	NULL	NULL	NULL	0,333	NULL	0	NULL	NULL	NULL

AST%	STL%	BLK%	TOV%	USG%	blan1	ovS	ovS	WS	WS/48	blan2	OBPM	DBPM	BPM	VOBP	FG	FGA	FG%	3P	3PA	3P%
NULL	NULL	NULL	NULL	NULL	NULL	-1,5	2,8	1,3	NULL	NULL	NULL	NULL	NULL	NULL	181	378	0,267	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	3,6	5,6	5,2	NULL	NULL	NULL	NULL	NULL	NULL	368	1156	0,318	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	0,2	0,5	0,7	NULL	NULL	NULL	NULL	NULL	NULL	18	38	0,333	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	-0,5	4,9	4,4	NULL	NULL	NULL	NULL	NULL	NULL	283	898	0,315	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	-1,7	3	1,3	NULL	NULL	NULL	NULL	NULL	NULL	113	482	0,281	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	4,8	3,6	8,4	NULL	NULL	NULL	NULL	NULL	NULL	223	558	0,405	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	-0,4	2,8	2,4	NULL	NULL	NULL	NULL	NULL	NULL	133	426	0,312	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	0	1,8	1,3	NULL	NULL	NULL	NULL	NULL	NULL	31	125	0,248	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	-3,7	5,2	1,4	NULL	NULL	NULL	NULL	NULL	NULL	244	861	0,283	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	0	0	0	NULL	NULL	NULL	NULL	NULL	NULL	1	3	0,333	NULL	NULL	NULL

2P	2PA	2P%	eFG%	FT	FTA	FT%	ORB	DRB	TRB	AST	STL	BLK	TOV	PF	PTS
181	378	0,267	0,267	77	112	0,688	NULL	NULL	NULL	88	NULL	NULL	NULL	133	279
368	1156	0,318	0,318	482	488	0,824	NULL	NULL	NULL	189	NULL	NULL	NULL	192	1138
18	38	0,333	0,333	18	27	0,667	NULL	NULL	NULL	8	NULL	NULL	NULL	32	38
283	898	0,315	0,315	186	259	0,718	NULL	NULL	NULL	168	NULL	NULL	NULL	198	752
113	482	0,281	0,281	61	98	0,622	NULL	NULL	NULL	91	NULL	NULL	NULL	147	287
223	558	0,405	0,405	197	277	0,711	NULL	NULL	NULL	182	NULL	NULL	NULL	178	643
133	426	0,312	0,312	71	83	0,855	NULL	NULL	NULL	104	NULL	NULL	NULL	112	337
31	125	0,248	0,248	57	83	0,687	NULL	NULL	NULL	43	NULL	NULL	NULL	63	119
244	861	0,283	0,283	146	258	0,584	NULL	NULL	NULL	124	NULL	NULL	NULL	246	634
1	3	0,333	0,333	0	0	NULL	NULL	NULL	NULL	2	NULL	NULL	NULL	1	2

Πίνακας 8 - Περιεχόμενα Πίνακα Seasons_Stats

Μία πρώτη παρατήρηση που εύκολα μπορεί να γίνει αφορά τις πολλές τιμές Null των δεδομένων. Ο λόγος είναι ότι καθώς τα αποτελέσματα στην αρχική τους μορφή εμφανίζονται με χρονολογική σειρά ξεκινώντας από την παλαιότερη σεζόν, υπάρχουν στατιστικά δεδομένα που δεν καταγράφονταν κατά τη διάρκεια εκείνων των αρχικών σεζόν. Επομένως γίνεται εύκολα κατανοητό ότι εάν η ανάλυση απαιτεί τα συγκεκριμένα στατιστικά, τότε θα πρέπει να επιλεγούν μόνο οι σεζόν για τις οποίες υπάρχει καταγραφή, δηλαδή οι τιμές δεν είναι Null.

Για να λάβουμε τα ονόματα των στηλών σε μορφή λίστας, κάνουμε μία αρχική δοκιμή σύνδεσης στη βάση δεδομένων μέσω της γλώσσας Python, ομοίως με τα παραδείγματα που περιγράφηκαν νωρίτερα.

```
import pyodbc

conn_str = pyodbc.connect(
    'DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-
69EC11S\SQLEXPRESS;DATABASE=NBA;Trusted_Connection=yes')
query_str = 'SELECT * FROM dbo.Seasons_Stats'
df = pandas.read_sql(sql=query_str, con=conn_str)

columns = df.columns.tolist()
print("columns: ", columns)
```

```
columns: ['F1', 'Year', 'Player', 'Pos', 'Age', 'Tm', 'G', 'GS', 'MP',
'PER', 'TS%', '3PAr', 'FTr', 'ORB%', 'DRB%', 'TRB%', 'AST%', 'STL%', 'BLK%',
'TOV%', 'USG%', 'blanl', 'OWS', 'DWS', 'WS', 'WS/48', 'blank2', 'OBPM',
'DBPM', 'BPM', 'VORP', 'FG', 'FGA', 'FG%', '3P', '3PA', '3P%', '2P', '2PA',
'2P%', 'eFG%', 'FT', 'FTA', 'FT%', 'ORB', 'DRB', 'TRB', 'AST', 'STL', 'BLK',
'TOV', 'PF', 'PTS']
```

Πίνακας 9 - Στήλες πίνακα Seasons_Stats

Για επισκόπηση των δεδομένων εκτελούμε δοκιμαστικά, εντός της Python, διάφορα queries όπως το κάτωθι που αφορά τους πρώτους σκόρερ όλων των εποχών:

```
#Return all time scoring leaders
query_PTS = 'SELECT Player, SUM(PTS) as TOTAL_CAREER_PTS FROM dbo.Seasons_Stats GRO
UP BY Player ORDER BY TOTAL_CAREER_PTS DESC'
df_PTS = pandas.read_sql(sql=query_PTS, con=conn_str)
print(df_PTS)
```

	Player	TOTAL_CAREER_PTS
0	Kareem Abdul-Jabbar*	38387.0
1	Karl Malone*	36928.0
2	Wilt Chamberlain*	33953.0
3	Kobe Bryant	33643.0
4	Michael Jordan*	32292.0
...
3917	J.J. O'Brien	0.0
3918	Jim Caldwell	0.0
3919	Guy Rucker	0.0
3920	Danuel House	0.0
3921	None	NaN

Πίνακας 10 - Πρώτοι σκόρερ όλων των εποχών

4.3. Κατηγοριοποίηση παικτών NBA σε συστάδες με χρήση του αλγορίθμου συσταδοποίησης K-means



Εικόνα 12 - Οι 5 θέσεις παικτών στο άθλημα του μπάσκετ

4.3.1. Παρουσίαση προβλήματος

Μία ομάδα μπάσκετ αποτελείται από διαφορετικού τύπου παίκτες που συνεισφέρουν με διάφορους τρόπους στο παιχνίδι της. Υπάρχουν παίκτες που κρατούν τη μπάλα στην κατοχή τους για αρκετή ώρα και οργανώνουν το παιχνίδι της ομάδας μοιράζοντας πολλές πάσες, παίκτες που έχουν έφεση στο σκοράρισμα, οι οποίοι εκτελούν και πετυχαίνουν τα περισσότερα σουτ, παίκτες με έφεση στο αμυντικό παιχνίδι κ.ο.κ.. Κάθε ομάδα προσπαθεί να διαθέτει έναν συνδυασμό παικτών με διαφορετικά χαρακτηριστικά, για να καλύψει κατά τον καλύτερο τρόπο τις ανάγκες ενός αγώνα, αναλόγως πάντα με τον τρόπο παιχνιδιού της.

Στο παρόν κεφάλαιο αναπτύσσεται και χρησιμοποιείται ένα μοντέλο συσταδοποίησης (Clustering) με τον αλγόριθμο K-means για την κατηγοριοποίηση των παικτών με κοινά αγωνιστικά χαρακτηριστικά σε συστάδες, ανάλογα με τις επιδόσεις τους σε συγκεκριμένες στατιστικές κατηγορίες. Σκοπός της ανάλυσης είναι η γρήγορη και εύκολη αναγνώριση της θέσης στην οποία αγωνίζεται και του τρόπου παιχνιδιού κάθε παίκτη, ανάλογα με τη συστάδα στην οποία ανήκει.

Η ανάλυση θα βασιστεί στις κάτωθι γενικές στατιστικές κατηγορίες για κάθε παίκτη, για το σύνολο των ετών που αγωνίστηκε εντός του διαθέσιμου συνόλου δεδομένων:

- Πόντοι (Points, στήλη PTS). Οι συνολικοί πόντοι που πετυχαίνει ένας παίκτης με κάθε δυνατό τρόπο (ελεύθερες βολές, σουτ 2 πόντων και σουτ 3 πόντων).
- Τελικές πάσες (Assists, στήλη AST). Οι πάσες που δίνει ένας παίκτης των οποίων ο άμεσος αποδέκτης σκοράρει με σουτ 2 ή 3 πόντων.
- Ριμπάουντ (Rebounds, στήλη TRB). Η απόκτηση της μπάλας από τον πρώτο παίκτη μετά από χαμένο σουτ ή βολή.
- Κλεψίματα (Steals). Η απόκτηση της κατοχής της μπάλας από τον αντίπαλο.
- Κοψίματα (Blocks). Το σταμάτημα της πορείας της μπάλας προς το καλάθι, κατά την προσπάθεια του αντιπάλου για σουτ.
- Ελεύθερες βολές (Free throw). Σουτ αξίας 1 πόντου που εκτελείται σε νεκρό χρόνο και χωρίς αντίπαλο, από σταθερή θέση, όταν ένας παίκτης κερδίσει φάουλ.
- Σουτ 2 πόντων (2 point shot). Σουτ αξίας 2 πόντων που γίνεται σε κανονική ροή αγώνα, από οποιαδήποτε θέση εντός της γραμμής 3 πόντων.
- Σουτ 3 πόντων (3 point shot). Σουτ αξίας 3 πόντων που γίνεται σε κανονική ροή αγώνα, από οποιαδήποτε θέση εκτός της γραμμής 3 πόντων.

4.3.2. Προεπεξεργασία των δεδομένων

Αρχικά επιχειρείται καθαρισμός των δεδομένων από κάποιους επιπλέον χαρακτήρες όπως το σύμβολο ‘ * ‘, που υπάρχει δίπλα στο όνομα κάποιων παικτών για να υποδηλώσει ότι ο συγκεκριμένος παίκτης έχει μπει στο Basketball Hall of Fame, τιμητική διάκριση που απονέμεται στους πιο επιδραστικούς παίκτες μετά το τέλος της καριέρας τους.

```
UPDATE Seasons_Stats  
SET Player = REPLACE(Player, '*', '')
```

Στη συνέχεια εκτελώντας το κάτωθι query σε SQL διαπιστώνεται ότι υπάρχουν πολλές εγγραφές με τιμή NULL στο όνομα παίκτη και σε όλες τις στήλες.

```
SELECT *  
FROM Seasons_Stats  
WHERE Player IS NULL
```

	F1	Year	Player	Pos	Age	Tm	G	GS	MP	PER	TS%	3PA%	FTr
1	312	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
2	487	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	618	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	779	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
5	911	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
6	1021	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
7	1128	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
8	1236	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Πίνακας 11 - Τιμές NULL που εμπεριέχονται στον πίνακα Seasons_Stats

Διαγράφουμε τις κενές εγγραφές με χρήση του κάτωθι query:

```
DELETE FROM Seasons_Stats
WHERE Player IS NULL
```

Στη συνέχεια διαγράφουμε τις στήλες blan1 και blank2 οι οποίες είναι κενές:

```
ALTER TABLE Seasons_Stats
DROP COLUMN blan1, blank2
```

Από τη μελέτη των δεδομένων παρατηρούμε ότι στη στήλη Tm που εκφράζει την ομάδα για την οποία αγωνίζεται κάθε στιγμή ο κάθε παίκτης, υπάρχει η εγγραφή “TOT”, που σύμφωνα με το Basketball Reference είναι συντομογραφία του Total και χρησιμοποιείται για τις περιπτώσεις που ένας παίκτης αγωνίστηκε για περισσότερες από μία ομάδες την ίδια σεζόν. Βρίσκοντας ποιοι παίκτες έχουν την τιμή “TOT” σε κάποιες από τις σεζόν που αγωνίστηκαν, μπορούμε να εκτελέσουμε ένα query για κάποιον από αυτούς και να δούμε όλες τις ομάδες που αγωνίστηκε σε μία συγκεκριμένη χρονιά. Για παράδειγμα εκτελείται το κάτωθι query για τον παίκτη Lou Williams και τη σεζόν 2016-2017, ώστε να βρούμε τα συνολικά παιχνίδια που αγωνίστηκε σε εκείνη τη σεζόν, για κάθε ομάδα.

```
SELECT F1, Year, Player, G, Tm
FROM Seasons_Stats
WHERE Year = 2017 AND Player = 'Lou Williams'
```


	F1	Year	Player	G	Tm
1	24667	2017	Lou Williams	81	TOT
2	24669	2017	Lou Williams	23	HOU
3	24668	2017	Lou Williams	58	LAL

Πίνακας 12 - Συνολικά παιχνίδια παίκτη Lou Williams τη σεζόν 2016-2017

Παρατηρούμε ότι η εγγραφή που αντιστοιχεί στην τιμή “TOT” περιλαμβάνει το άθροισμα των άλλων 2 επιμέρους εγγραφών, που αντιστοιχούν σε κάθε μία από τις ομάδες που αγωνίστηκε τη συγκεκριμένη σεζόν ο παίκτης. Επομένως τα στατιστικά κάθε παίκτη που ανήκει σε αυτή την κατηγορία είναι καταχωρημένα εις διπλούν στον πίνακα, μία φορά για κάθε ομάδα που αγωνίστηκε και μία φορά για το σύνολο της κάθε σεζόν.

Ως εκ τούτου κρίνεται απαραίτητο να αφαιρεθούν από τον πίνακα Seasons_Stats όλες οι εγγραφές με τιμή “TOT” στη στήλη Tm, εφόσον όλα τα περιεχόμενα σε αυτές δεδομένα υπάρχουν ήδη μοιρασμένα σε άλλες εγγραφές.

```
DELETE FROM Seasons_Stats WHERE Tm = 'TOT';
```

4.3.3. Προσδιορισμός επιθυμητών δεδομένων και εισαγωγή σε dataframe του Pandas

Αρχικά πρέπει να επιλεγούν τα δεδομένα για τα οποία υπάρχει καταγραφή όλων των ζητούμενων στατιστικών για την ανάλυση και ως εκ τούτου οι τιμές των αντίστοιχων στηλών δεν είναι NULL. Υπάρχουν στατιστικές κατηγορίες για τις οποίες δεν υπήρχε καταγραφή από την πρώτη σεζόν, όπως τα ριμπάουντ που καταγράφονται για πρώτη φορά τη σεζόν 1950-1951, τα κλεψίματα και τα κοψίματα τη σεζόν 1973-1974, ενώ το σουτ 3 πόντων θεσπίστηκε για πρώτη φορά τη σεζόν 1979-1980. Ως εκ τούτου επιλέγεται να χρησιμοποιηθούν για την ανάλυση μόνο οι εγγραφές που αφορούν τη σεζόν 1979-1980 και τις μεταγενέστερες.

Προς διευκόλυνση της χρήσης δημιουργούμε έναν νέο πίνακα με όνομα **Stats**, ο οποίος προκύπτει από το αρχικό σύνολο δεδομένων του πίνακα **Seasons_Stats** αφαιρώντας τις εγγραφές που αντιστοιχούν στις προγενέστερες της 1979-1980 σεζόν.

```
SELECT * INTO Stats
FROM Seasons_Stats
WHERE Year > 1979
```

Στη συνέχεια με χρήση της γλώσσας προγραμματισμού Python προσδιορίζονται οι ζητούμενες διαστάσεις των δεδομένων που θα χρειαστούμε για την ανάλυση και φορτώνονται τα δεδομένα σε ένα data frame του pandas.

Ξεκινάμε ανοίγοντας ένα Notebook σε Python και εισάγουμε τις βιβλιοθήκες που περιέχουν τις συναρτήσεις οι οποίες θα μας χρειαστούν για την ανάλυση:

```
# Load packages.
import pyodbc
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from scipy.spatial import distance as sci_distance
from sklearn import cluster as sk_cluster
```

Αρχικά δημιουργούμε ένα νέο πίνακα με όνομα **Player_Stats**, στον οποίο θα αποθηκευτούν τα δεδομένα που θα προκύψουν από τον πίνακα **Stats** της Β.Δ., με την εκτέλεση του κάτωθι query της SQL, που βρίσκεται ενσωματωμένο στον κώδικα σε Python.

Οι επιθυμητές για την ανάλυση διαστάσεις των δεδομένων για κάθε παίκτη και για κάθε σεζόν του, που περιλαμβάνονται στον πίνακα **Stats**, διαμορφώνονται ως εξής:

- **PPG** (Points Per Game) = Πόντοι ανά παιχνίδι. Προκύπτει διαιρώντας τους συνολικούς πόντους (στήλη PTS) κάθε παίκτη με τα συνολικά παιχνίδια (στήλη G) στα οποία συμμετείχε.
- **APG** (Assists Per Game) = Τελικές πάσες (ασίστ) ανά παιχνίδι. Προκύπτει διαιρώντας τις συνολικές ασίστ (στήλη AST) κάθε παίκτη με τα συνολικά παιχνίδια (στήλη G) στα οποία συμμετείχε.

- **RPG** (Rebounds Per Game) = Ριμπάουντ ανά παιχνίδι. Προκύπτει διαιρώντας τα συνολικά ριμπάουντ (στήλη TRB) κάθε παίκτη με τα συνολικά παιχνίδια (στήλη G) στα οποία συμμετείχε. Σημειώνεται ότι στα δεδομένα υπάρχει καταγραφή και ξεχωριστά για αμυντικά (DRB) και επιθετικά (ORB) ριμπάουντ, όμως επιλέχθηκε να χρησιμοποιηθούν για την ανάλυση μόνο τα συνολικά.
- **SPG** (Steals Per Game) = Κλεψίματα ανά παιχνίδι. Προκύπτει διαιρώντας τα συνολικά κλεψίματα (στήλη STL) κάθε παίκτη με τα συνολικά παιχνίδια (στήλη G) στα οποία συμμετείχε.
- **BPG** (Blocks Per Game) = Κοψίματα ανά παιχνίδι. Προκύπτει διαιρώντας τα συνολικά κοψίματα (στήλη BLK) κάθε παίκτη με τα συνολικά παιχνίδια (στήλη G) στα οποία συμμετείχε.
- **FTP** (Free Throw Percentage) = Ποσοστό επιτυχημένων ελεύθερων βολών. Προκύπτει διαιρώντας τις συνολικές βολές (στήλη FT) που έχει πετύχει μέσα στη χρονιά ο κάθε παίκτης με τις συνολικές βολές που εκτέλεσε (στήλη FTA).
- **2PP** (2 Points Percentage) = Ποσοστό σουτ 2 πόντων. Προκύπτει διαιρώντας τα συνολικά σουτ 2 πόντων (στήλη 2P) που έχει πετύχει μέσα στη χρονιά ο κάθε παίκτης με τα συνολικά σουτ 2 πόντων που εκτέλεσε (στήλη 2PA).
- **3PP** (3 Points Percentage) = Ποσοστό σουτ 3 πόντων. Προκύπτει διαιρώντας τα συνολικά σουτ 3 πόντων (στήλη 3P) που έχει πετύχει μέσα στη χρονιά ο κάθε παίκτης με τα συνολικά σουτ 3 πόντων που εκτέλεσε (στήλη 3PA).

Παρακάτω φαίνεται ο κώδικας σε Python που εκτελεί τη διαδικασία που περιγράφηκε:

```
# Connection string to connect to SQL Server named instance.
conn_str = pyodbc.connect(
    'DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-
69EC11S\SQLEXPRESS;DATABASE=NBA;Trusted_Connection=yes')

input_query = ''' SELECT F1,
ROUND(COALESCE(PTS/NULLIF(1.0*G, 0),0), 1) AS PPG,
ROUND(COALESCE(AST/NULLIF(1.0*G, 0),0), 1) AS APG,
ROUND(COALESCE(TRB/NULLIF(1.0*G, 0),0), 1) AS RPG,
ROUND(COALESCE(STL/NULLIF(1.0*G, 0),0), 1) AS SPG,
ROUND(COALESCE(BLK/NULLIF(1.0*G, 0),0), 1) AS BPG,
ROUND(COALESCE(FT/NULLIF(1.0*FTA, 0),0), 1) AS [FTP],
ROUND(COALESCE([2P]/NULLIF(1.0*[2PA], 0),0), 1) AS [2PP],
ROUND(COALESCE([3P]/NULLIF(1.0*[3PA], 0),0), 1) AS [3PP]
```

```
FROM Stats
'''

# Define the columns we wish to import.
column_info = {
    "F1": {"type": "integer"},
    "PPG": {"type": "integer"},
    "APG": {"type": "integer"},
    "RPG": {"type": "integer"},
    "SPG": {"type": "integer"},
    "BPG": {"type": "integer"},
    "FTP": {"type": "integer"},
    "2PP": {"type": "integer"},
    "3PP": {"type": "integer"},
}

Player_Stats = pd.read_sql(input_query, conn_str)
```

Για λόγους εποπτείας, τυπώνονται οι 5 πρώτες σειρές του πίνακα Player_Stats:

```
print("Data frame:", Player_Stats.head(n=5))
```

Data frame:	F1	PPG	APG	RPG	SPG	BPG	FTP	2PP	3PP
0	5727.0	24.8	4.5	10.8	1.0	3.4	0.8	0.6	0.0
1	5728.0	5.4	1.3	2.9	0.5	0.2	0.7	0.5	0.0
2	5729.0	14.9	4.3	8.1	1.4	0.7	0.8	0.5	0.0
3	5730.0	14.1	8.4	2.5	1.3	0.1	0.8	0.5	0.2
4	5731.0	3.3	1.5	4.4	0.5	0.6	0.6	0.5	0.0

Πίνακας 13 - Περιεχόμενα του Πίνακα Player_Stats

4.3.4. Δημιουργία μοντέλου συσταδοποίησης (clustering)

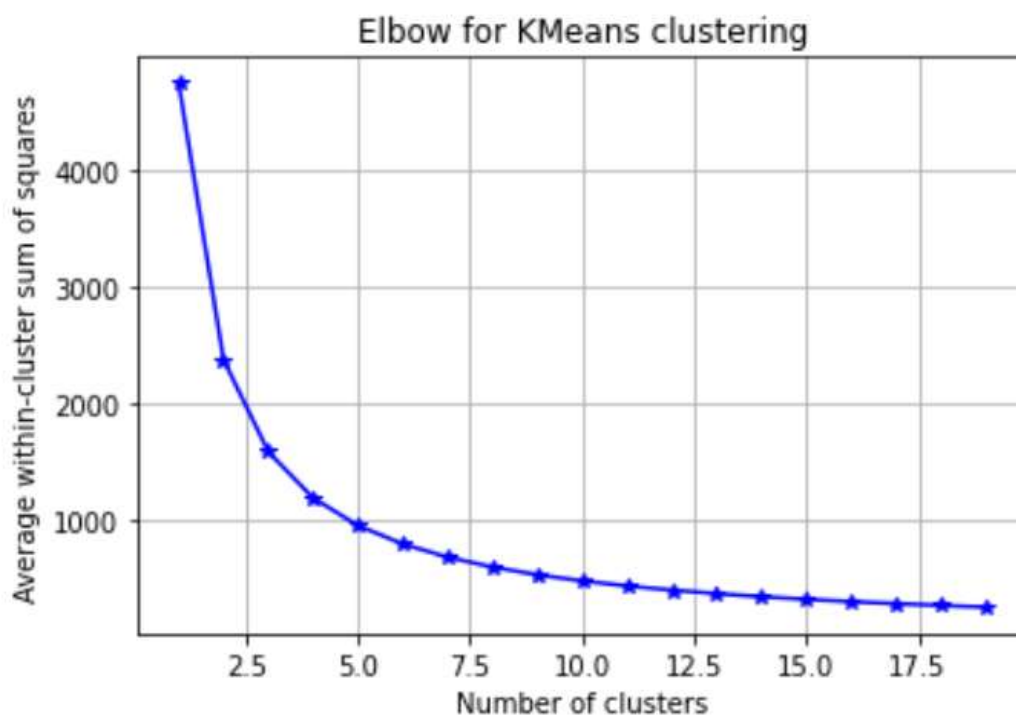
Στο επόμενο βήμα κατασκευάζουμε το μοντέλο συσταδοποίησης σε Python με τον αλγόριθμο K-Means clustering.

Για τη δημιουργία της γραφικής παράστασης, εισάγεται η μεταβλητή **Pdata** στην οποία αποθηκεύεται ο πίνακας Player_Stats που δημιουργήσαμε στο προηγούμενο βήμα. Για τις τιμές του K ορίζεται αρχικά ένα εύρος από 1 έως 19. Μέσω της κλάσης **cluster.KMeans** της βιβλιοθήκης sklearn καλείται η μέθοδος **fit** η οποία με όρισμα τα δεδομένα του πίνακα Pdata εκτελεί τη συσταδοποίηση για κάθε τιμή του K.

```
## Determine number of clusters using the Elbow method
Pdata = Player_Stats
K = range(1, 20)
```

```
KM = (sk_cluster.KMeans(n_clusters=k).fit(Pdata) for k in K)
centroids = (k.cluster_centers_ for k in KM)

D_k = (sci_distance.cdist(Pdata, cent, 'euclidean') for cent in centroids)
dist = (np.min(D, axis=1) for D in D_k)
avgWithinSS = [sum(d) / Pdata.shape[0] for d in dist]
plt.plot(K, avgWithinSS, 'b*-')
plt.grid(True)
plt.xlabel('Number of clusters')
plt.ylabel('Average within-cluster sum of squares')
plt.title('Elbow for KMeans clustering')
plt.show()
```



Εικόνα 13 - Γραφική παράσταση Elbow method for KMeans clustering

Με βάση τη γραφική παράσταση και με χρήση της «Elbow method» ως τιμή του k που βρίσκεται εγγύτερα στον «αγκώνα» της καμπύλης θα μπορούσε να θεωρηθεί το $k = 4$, επομένως επιλέγεται να ομαδοποιηθούν τα δεδομένα σε 4 συστάδες.

Στη συνέχεια εκτελείται εκ νέου η μέθοδος **fit** της κλάσης **cluster.KMeans** για $k = 4$.

```
## Perform clustering using Kmeans

# It looks like k=4 is a good number to use based on the elbow graph.
n_clusters = 4
```

```
means_cluster = sk_cluster.KMeans(n_clusters=n_clusters, random_state=111)
columns = ["APG", "RPG", "SPG", "BPG", "FTP", "2PP", "3PP"]
est = means_cluster.fit(Player_Stats[columns])
clusters = est.labels_
Player_Stats['cluster'] = clusters
```

Τυπώνουμε τα μεγέθη των 4 clusters, δηλαδή το πλήθος των δεδομένων που κατανεμήθηκαν σε κάθε συστάδα:

```
# For each cluster, count the members.
for c in range(n_clusters):
    cluster_members=Player_Stats[Player_Stats['cluster'] == c][:]
    print('Cluster{}(n={}):'.format(c, len(cluster_members)))
    print('-'* 17)
```

```
Cluster0 (n=5132) :
-----
Cluster1 (n=9182) :
-----
Cluster2 (n=2470) :
-----
Cluster3 (n=2143) :
```

Πίνακας 14 - Πλήθος δεδομένων ανά cluster

Και τέλος τυπώνουμε τις μέσες τιμές, ανά συστάδα, των στηλών που εκφράζουν τις ζητούμενες διαστάσεις των δεδομένων.

```
print(Player_Stats.groupby(['cluster']).mean())
```

cluster	F1	PPG	APG	RPG	SPG	BPG	\
0	15379.002728	9.326773	1.416816	4.583807	0.707385	0.576072	
1	15427.675452	4.227783	1.024766	1.623993	0.404182	0.169277	
2	14683.152227	13.790850	5.581984	3.342955	1.304939	0.241336	
3	14482.259449	14.741297	2.115539	8.842977	0.904760	1.206393	
cluster	FTP	2PP	3PP				
0	0.710659	0.482288	0.186107				
1	0.655772	0.424483	0.199292				
2	0.792429	0.470283	0.304089				
3	0.712179	0.507140	0.148110				

Πίνακας 15 - Μέσες τιμές ανά cluster

Από τη μελέτη των αποτελεσμάτων, μπορούν να βγουν κάποια πρώτα συμπεράσματα:

Η συστάδα 3 φαίνεται να περιλαμβάνει παίκτες που παίζουν πιο κοντά στο καλάθι, καθώς έχουν καλύτερες επιδόσεις σε κατηγορίες που ιστορικά χαρακτηρίζουν παίκτες με αυτό το προφίλ: είναι 1^{οι} στα ριμπάουντ (RPG), στα κοψίματα (BPG) και στα ποσοστά ευστοχίας στα σουτ 2 πόντων (2PP), ενώ είναι τελευταίοι στα ποσοστά ευστοχίας στα σουτ 3 πόντων (3PP).

Η συστάδα 2 αντίθετα φαίνεται να αφορά παίκτες που αγωνίζονται πιο μακριά από το καλάθι, καθώς είναι 1^{οι} στις τελικές πάσες (APG), στα κλεψίματα (SPG) και στα ποσοστά ευστοχίας στα σουτ 3 πόντων (3PP).

Η συστάδα 1 μοιάζει να αφορά παίκτες με σχετικά μικρή συνεισφορά στο παιχνίδι της ομάδας τους, καθώς είναι τελευταίοι σχεδόν σε όλες τις στατιστικές κατηγορίες.

Τέλος, η συστάδα 0 περιλαμβάνει παίκτες με ενδιάμεσα χαρακτηριστικά, πιθανώς και καλούς αμυντικούς, καθώς είναι 2^{οι} στα ριμπάουντ (RPG), στα κοψίματα (BPG) αλλά και στα ποσοστά ευστοχίας στα σουτ 2 πόντων (2PP).

Τέτοιου είδους συμπεράσματα θα μπορούσαν να βοηθήσουν μια ομάδα να κάνει πιο στοχευμένη έρευνα, γύρω από παίκτες που ενδιαφέρεται να αποκτήσει, για να καλύψει συγκεκριμένες ανάγκες του παιχνιδιού της.

4.3.5. Ενσωμάτωση του Clustering Model με SQL Machine Learning

Στη συνέχεια ενσωματώνουμε το Python script ως αποθηκευμένη διαδικασία της SQL.

Σε αυτό το στάδιο θα δημιουργηθεί η αποθηκευμένη διαδικασία η οποία θα δημιουργεί το μοντέλο, θα εκτελεστεί η συσταδοποίηση εντός του server και θα αξιοποιηθούν οι πληροφορίες που εξάγονται από τα αποτελέσματα.

```
USE [NBA]
GO
DROP PROCEDURE IF EXISTS [py_generate_players_clusters]
GO
CREATE procedure [dbo].[py_generate_players_clusters]
AS
BEGIN
    DECLARE
```

```
-- Input query to generate the players metrics
@input_query NVARCHAR(MAX) = N'
SELECT F1,
ROUND(COALESCE(PTS/NULLIF(1.0*G, 0),0), 1) AS PPG,
ROUND(COALESCE(AST/NULLIF(1.0*G, 0),0), 1) AS APG,
ROUND(COALESCE(TRB/NULLIF(1.0*G, 0),0), 1) AS RPG,
ROUND(COALESCE(STL/NULLIF(1.0*G, 0),0), 1) AS SPG,
ROUND(COALESCE(BLK/NULLIF(1.0*G, 0),0), 1) AS BPG,
ROUND(COALESCE(FT/NULLIF(1.0*FTA, 0),0), 1) AS [FTP],
ROUND(COALESCE([2P]/NULLIF(1.0*[2PA], 0),0), 1) AS [2PP],
ROUND(COALESCE([3P]/NULLIF(1.0*[3PA], 0),0), 1) AS [3PP]
FROM Stats
'

EXEC sp_execute_external_script
    @language = N'Python'
    , @script = N'

import pandas as pd
from sklearn.cluster import KMeans

#get data from input query
Player_Stats = my_input_data

#We concluded in step 2 in the tutorial that 4 would be a good number of clusters
n_clusters = 4

#Perform clustering
est = KMeans(n_clusters=n_clusters, random_state=111).fit(Player_Stats[["APG", "RPG", "SPG", "BPG", "FTP", "2PP", "3PP"]])
clusters = est.labels_
Player_Stats["cluster"] = clusters

OutputDataSet = Player_Stats
'

    , @input_data_1 = @input_query
    , @input_data_1_name = N'my_input_data'
        with result sets (("F1" int, "PPG" float, "APG" float, "RPG" float, "SPG" float, "BPG" float, "FTP" float, "2PP" float, "3PP" float, "cluster" float));
END;
GO
```

Αφού δημιουργήσαμε την αποθηκευμένη διαδικασία, εκτελούμε τον κάτωθι κώδικα για να δημιουργήσουμε ένα νέο πίνακα με όνομα **py_players_clusters** εντός της Β.Δ., για την αποθήκευση των αποτελεσμάτων της συσταδοποίησης.

```
--Create a table to store the predictions in

DROP TABLE IF EXISTS [dbo].[py_players_clusters];
```

```
GO
CREATE TABLE [dbo].[py_players_clusters] (
    [F1] [bigint] NULL
, [PPG] [float] NULL
, [APG] [float] NULL
, [RPG] [float] NULL
, [SPG] [float] NULL
, [BPG] [float] NULL
, [FTP] [float] NULL
, [2PP] [float] NULL
, [3PP] [float] NULL
, [cluster] [int] NULL
,
) ON [PRIMARY]
GO
```

Κατόπιν εκτελούμε την αποθηκευμένη διαδικασία, αποθηκεύουμε τα αποτελέσματα στο νέο πίνακα `py_players_clusters` και εκτυπώνουμε τα περιεχόμενά του για επαλήθευση.

```
--Execute the clustering and insert results into table
INSERT INTO py_players_clusters
EXEC [dbo].[py_generate_players_clusters];
-- Select contents of the table to verify it works
SELECT * FROM py_players_clusters;
```

Τα περιεχόμενά του νέου πίνακα φαίνονται παρακάτω:

	F1	PPG	APG	RPG	SPG	BPG	FTP	2PP	3PP	cluster
1	5727	24,8	4,5	10,8	1	3,4	0,8	0,6	0	0
2	5728	5,4	1,3	2,9	0,5	0,2	0,7	0,5	0	2
3	5729	14,9	4,3	8,1	1,4	0,7	0,8	0,5	0	0
4	5730	14,1	8,4	2,5	1,3	0,1	0,8	0,5	0,2	3
5	5731	3,3	1,5	4,4	0,5	0,6	0,6	0,5	0	1
6	5732	1,9	1,3	1,4	0,3	0,2	0,4	0,4	1	2
7	5733	4,7	0,4	2,9	0,3	0,8	0,7	0,5	0	2
8	5734	15,6	1,9	7,8	1,1	0,4	0,8	0,5	0,3	0
9	5735	11,8	3,6	5,9	1,1	0,6	0,7	0,5	0,3	1
10	5736	3,2	0,9	3,9	0,3	0,6	0,5	0,4	0	1

Πίνακας 16 - Εμφάνιση συστάδας για κάθε παίκτη

Καθώς αποθηκεύσαμε τη διαδικασία συσταδοποίησης εντός της Β.Δ., αυτή μπορεί να εκτελέσει συσταδοποίηση των δεδομένων παικτών που είναι αποθηκευμένα εντός της

ίδιας Β.Δ.. Η διαδικασία μπορεί να εκτελεστεί κάθε φορά που ενημερώνονται τα δεδομένα των παικτών και να χρησιμοποιηθούν τα επικαιροποιημένα αποτελέσματα της συσταδοποίησης.

Ας υποθέσουμε ότι θέλουμε να βρούμε τη θέση κάθε παίκτη της συστάδας 0 σε κάθε χρονιά. Ο ακόλουθος κώδικας επιστρέφει τις θέσεις των παικτών της συστάδας 0:

```
--Get Position of every player in cluster 0  
SELECT S.Player, S.Year, S.Tm, S.Pos  
FROM Seasons_Stats S  
JOIN  
py_players_clusters as C  
ON C.F1 = S.F1  
WHERE C.cluster = 0  
ORDER BY S.Player, S.Year
```

	Player	Year	Tm	Pos
1	A.C. Green	1987	LAL	PF
2	A.C. Green	1988	LAL	PF
3	A.C. Green	1989	LAL	PF
4	A.C. Green	1990	LAL	PF
5	A.C. Green	1992	LAL	PF
6	A.C. Green	1993	LAL	PF
7	A.C. Green	1994	PHO	PF
8	A.C. Green	1995	PHO	SF
9	A.C. Green	1997	DAL	PF
10	A.C. Green	1998	DAL	PF
11	Aaron Williams	2001	NJN	PF
12	Adonal Foyle	2001	GSW	C
13	Adrian Dantley	1980	UTA	SF
14	Al Harrington	2005	ATL	SF
15	Al Harrington	2006	ATL	PF

Πίνακας 17 - Παίκτες ανά σεζόν που ανήκουν στη συστάδα μηδέν

Αναζητώντας τις θέσεις των παικτών κάθε συστάδας, επαληθεύουμε ότι είναι ορθή η λογική της συσταδοποίησης που έγινε.

Εκτελώντας το κάτωθι query λαμβάνουμε τις θέσεις των παικτών που ανήκουν στη συστάδα 3:

```
SELECT L.Pos, COUNT(L.Pos) AS num
FROM
  (SELECT S.F1, S.Pos
   FROM Seasons_Stats S
   JOIN
     py_players_clusters as C
   ON C.F1 = S.F1
   WHERE C.cluster = 3) L
GROUP BY L.Pos
ORDER BY num DESC
```

	Pos	num
1	C	912
2	PF	867
3	SF	213
4	SG	21
5	PG	2

Παρατηρούμε ότι όντως η συντριπτική πλειοψηφία των μελών της συστάδας 3 ανήκει στη θέση “C” που εκφράζει τους Centers δηλαδή τους παίκτες που παίζουν πιο κοντά στο καλάθι. Δεύτεροι περισσότεροι, επίσης με μεγάλο ποσοστό είναι οι “PF” (Power Forwards) που παίζουν επίσης πιο συχνά κοντά στο καλάθι. Ακολουθούν με μικρότερη συμμετοχή οι SF – Small Forwards, ενώ στις θέσεις που κατά κανόνα παίζουν μακριά από το καλάθι (SG – Shooting Guards, PG – Point Guards) παρατηρούμε ότι το ποσοστό εμφάνισης είναι αμελητέο.

Εκτελώντας το κάτωθι query λαμβάνουμε τις θέσεις των παικτών που ανήκουν στη συστάδα 2:

```
SELECT L.Pos, COUNT(L.Pos) AS num
FROM
  (SELECT S.F1, S.Pos
   FROM Seasons_Stats S
   JOIN
     py_players_clusters as C
   ON C.F1 = S.F1
   WHERE C.cluster = 2) L
GROUP BY L.Pos
ORDER BY num DESC
```

	Pos	num
1	PG	1530
2	SG	573
3	SF	165
4	PF	16
5	C	3

Στη συστάδα 2 παρατηρούμε ότι όπως αναμέναμε τα αποτελέσματα είναι αντίστροφα της συστάδας 3, με τα ποσοστά των PG και SG να είναι συντριπτικά μεγαλύτερα, των SF να είναι πιο χαμηλά και των PF και C να είναι αμελητέα.

Εκτελώντας το κάτωθι query λαμβάνουμε τις θέσεις των παικτών που ανήκουν στη συστάδα 1:

```
SELECT L.Pos, COUNT(L.Pos) AS num
FROM
  (SELECT S.F1, S.Pos
   FROM Seasons_Stats S
   JOIN
     py_players_clusters as C
   ON C.F1 = S.F1
   WHERE C.cluster = 1) L
GROUP BY L.Pos
ORDER BY num DESC
```

	Pos	num
1	SG	2220
2	PG	1820
3	SF	1632
4	PF	1334
5	C	1266

Στη συστάδα 1, των παικτών με μικρή συνεισφορά στο παιχνίδι, παρατηρούμε ότι τα ποσοστά είναι μοιρασμένα, κάτι λογικό καθώς το δείγμα στατιστικών δεδομένων τέτοιων παικτών είναι συνήθως μικρό, λόγω λίγων σεζόν, αγώνων αλλά και χρόνου συμμετοχής ανά αγώνα και ως εκ τούτου δεν παράγει αξιόπιστα αποτελέσματα σχετικά με τον εντοπισμό κάποιου προτύπου. Επιπλέον παρατηρούμε ότι πρόκειται για τη συστάδα με το μεγαλύτερο πλήθος παικτών συνολικά, γεγονός επίσης λογικό καθώς είναι πολλοί περισσότεροι οι παίκτες που έπαιζαν λίγο και είχαν μικρή συνεισφορά από εκείνους που είχαν πρωταγωνιστικό ρόλο και έπαιζαν περισσότερα χρόνια.

Τέλος εκτελώντας το κάτωθι query λαμβάνουμε τις θέσεις των παικτών που ανήκουν στη συστάδα 0, που αφορά παίκτες με ενδιάμεσα χαρακτηριστικά που αγωνίζονται ίσως πιο κοντά στο καλάθι:

```
SELECT L.Pos, COUNT(L.Pos) AS num
FROM
  (SELECT S.F1, S.Pos
   FROM Seasons_Stats S
   JOIN
     py_players_clusters as C
   ON C.F1 = S.F1
   WHERE C.cluster = 0) L
GROUP BY L.Pos
ORDER BY num DESC
```

	Pos	num
1	PF	1421
2	C	1322
3	SF	1317
4	SG	613
5	PG	44

4.4. Πρόβλεψη MVP του NBA με χρήση του αλγορίθμου γραμμικής παλινδρόμησης - Linear Regression Algorithm



Εικόνα 14 - Βραβείο MVP κανονικής σεζόν NBA

4.4.1. Παρουσίαση προβλήματος

Με την ολοκλήρωση της κανονικής διάρκειας κάθε σεζόν (Regular Season) του πρωταθλήματος στο αμερικάνικο επαγγελματικό μπάσκετ (NBA), ψηφίζεται ο πολυτιμότερος παίκτης (MVP) της χρονιάς. Έως τη σεζόν 1979-1980 για την ανάδειξη του MVP ψήφιζαν οι παίκτες του πρωταθλήματος ενώ από τη σεζόν 1980-1981 και μετά ψηφίζει ένα σύνολο δημοσιογράφων και αναλυτών. Τέλος από τη σεζόν 2010-2011 στη διαδικασία προστέθηκε και μία ψήφος από τους φιλάθλους μέσω online ψηφοφορίας.

Κάθε μέλος με δικαίωμα ψήφου καλείται να επιλέξει τους 5 πολυτιμότερους κατά την κρίση του παίκτες, ιεραρχικά από την 1^η έως την 5^η θέση, μέσω ενός συστήματος πόντων. Κάθε ψήφος που λαμβάνει ένας παίκτης για την 1^η θέση έχει αξία 10 πόντων, για τη 2^η θέση αξία 7 πόντων, για την 3^η θέση 5 πόντων, για την 4^η θέση 3 πόντων και για την 5^η θέση αξία 1 πόντου. Ο παίκτης που θα συγκεντρώσει τους περισσότερους συνολικά πόντους ανακηρύσσεται MVP της κανονικής διάρκειας του πρωταθλήματος (Regular Season MVP).

Στο παρόν κεφάλαιο αναπτύσσεται και χρησιμοποιείται ένα μοντέλο Γραμμικής Παλινδρόμησης με χρήση του αλγορίθμου Linear Regression της Microsoft, για την

πρόβλεψη των ψήφων που λαμβάνει ένας παίκτης στην ψηφοφορία για την ανάδειξη του MVP, ανάλογα με τις επιδόσεις του σε συγκεκριμένες στατιστικές κατηγορίες.

4.4.2. Εισαγωγή πινάκων

Αρχικά μέσω της ιστοσελίδας <https://www.kaggle.com> επιλέγεται το σύνολο δεδομένων **mvp_votings**, που περιλαμβάνει στατιστικά δεδομένα απόδοσης των παικτών του NBA που έλαβαν έστω και 1 ψήφο στην ψηφοφορία για MVP, για το διάστημα από τη σεζόν 1980-1981 έως και τη σεζόν 2017-2018.

Μέσω του Microsoft SQL Server και του Azure Data Studio εισάγουμε στη βάση δεδομένων με όνομα NBA.bak, την οποία δημιουργήσαμε στο κεφάλαιο 4.3, τον πίνακα **MVP**.

Ανοίγοντας ένα query σε SQL μπορούμε να κάνουμε μία προεπισκόπηση των δεδομένων:

```
SELECT * FROM MVP
```

	num	PER	BPM	season	player	win_pct	votes_first	points_won	points_max	award_share	G
1	0	25,1	0	1981	Julius Erving	0,75609	28	454	690	0,658	82
2	1	19,9	5,1	1981	Larry Bird	0,75609	20	423	690	0,613	82
3	2	25,5	5,3	1981	Kareem Abdul-Jabbar	0,65853	8	286	690	0,414	80
4	3	25,1	3,7	1981	Moses Malone	0,4878	8	180	690	0,261	80
5	4	22,9	1,6	1981	George Gervin	0,63414	1	83	690	0,12	82
6	5	22	5,9	1981	Marques Johnson	0,7317	1	73	690	0,106	76
7	6	25,2	5	1981	Robert Parish	0,75609	0	53	690	0,077	82
8	7	17	2	1981	Dennis Johnson	0,69512	0	50	690	0,072	79
9	8	14,3	-1,3	1981	Tiny Archibald	0,75609	0	32	690	0,046	80
1..	9	18	0,8	1981	Janaal Wilkes	0,65853	0	19	690	0,028	81
1..	10	25,7	9,2	1981	Magic Johnson	0,65853	0	18	690	0,026	37
1..	11	24,3	4,6	1981	Adrian Dantley	0,34146	1	15	690	0,022	80
1..	12	17,3	-1,1	1981	Phil Ford	0,4878	1	15	690	0,022	66
1..	13	19,9	3,3	1981	Bernard King	0,4756	0	12	690	0,017	81
1..	14	14,9	-2,8	1981	Kelvin Ransey	0,54878	1	11	690	0,016	80

MPG	PPG	RPG	APG	SPG	BPG	FPG	_3PP	FTP	MS	MS48	fga	fg3a	fta	ts_pct	usg_pct
35	24,6	8	4,4	2,1	1,8	0,521	0,222	0,787	13,8	0,231	18,6	0,2	6,5	0,572	28,4
39,5	21,2	10,9	5,5	2	0,8	0,478	0,27	0,853	10,8	0,16	16,3	0,9	4	0,528	24,3
37,2	25,2	10,3	3,4	0,7	2,9	0,574	0	0,766	14,3	0,23	18,2	0	6,9	0,616	26,3
40,6	27,8	14,8	1,8	1	1,9	0,522	0,333	0,757	13,7	0,202	19,3	0	10,1	0,585	27,6
33,7	27,1	5,1	3,2	1,1	0,7	0,492	0,257	0,826	10,5	0,182	21,1	0,4	7,6	0,555	32,3
33,4	20,3	6,8	4,6	1,5	0,5	0,552	0	0,706	11,2	0,211	15,2	0,1	5	0,583	23,3
28	18,9	9,5	1,8	1	2,6	0,545	0	0,71	10,9	0,228	14,2	0	4,8	0,579	27,1
33,1	18,8	4,6	3,7	1,7	0,8	0,436	0,216	0,62	8,4	0,154	15,4	0,6	6,3	0,516	24,6
35,3	13,8	2,2	7,7	0,9	0,2	0,499	0	0,816	6,9	0,118	9,6	0,1	5,2	0,582	17,5
37,4	22,6	5,4	2,9	1,5	0,4	0,526	0,077	0,758	8,5	0,135	18,5	0,2	4,1	0,556	24,5
37,1	21,6	8,6	8,6	3,4	0,7	0,532	0,176	0,76	6,4	0,225	15,9	0,5	6,1	0,582	24,3
42,7	30,7	6,4	4	1,4	0,2	0,559	0,288	0,886	13,6	0,191	20,3	0,1	9,8	0,622	28,4
34,7	17,5	1,9	8,8	1,5	0,1	0,478	0,306	0,831	5,2	0,11	13,4	0,5	5,4	0,553	23,3
36	21,9	6,8	3,5	0,9	0,4	0,588	0,333	0,703	9,1	0,15	15,4	0,1	5,4	0,617	23,2
30,4	15,2	2,4	6,9	1,1	0,1	0,452	0,097	0,749	2,8	0,056	14,5	0,4	2,7	0,484	24,3

Πίνακας 18 - Περιεχόμενα Πίνακα MVP

Στα δεδομένα που θα χρησιμοποιηθούν υπάρχουν 4 στήλες που σχετίζονται με τη βαθμολογία του παίκτη ως κάτωθι:

- votes_first: Οι ψήφοι για την 1^η θέση, δηλαδή αξίας 10 πόντων, που έλαβε ο παίκτης.
- points_won: Οι συνολικοί πόντοι που συγκέντρωσε ο παίκτης, όπως προκύπτουν ως άθροισμα των πόντων κάθε ψήφου που έλαβε για τις θέσεις 1 έως 5.
- points_max: Το μέγιστο άθροισμα των πόντων που μπορεί να συγκεντρώσει ένας παίκτης, εάν δηλαδή ψηφιστεί από όλα τα μέλη για την 1^η θέση.
- award_share: Το ποσοστό των συνολικών πόντων που συγκέντρωσε ένας παίκτης σε σχέση με το μέγιστο δυνατό, όπως προκύπτει από το λόγο points_won/points_max.

Καθώς ο αριθμός των μελών με δικαίωμα ψήφου δεν είναι σταθερός ανά τα χρόνια, κρίνεται ότι το πιο αντιπροσωπευτικό μέγεθος για τη μέτρηση της επιτυχίας κάθε παίκτη στην ψηφοφορία για την ανάδειξη του MVP είναι η στήλη **award_share**.

4.4.3. Προεπεξεργασία των δεδομένων

Σε αυτό το βήμα γίνεται επεξεργασία των δεδομένων της ΒΔ μέσω της Python. Αρχικά φορτώνονται τα δεδομένα σε ένα data frame του pandas και στη συνέχεια γίνεται μια προεργασία τους αφαιρώντας τις στήλες που δεν επιθυμούμε να χρησιμοποιηθούν από το μοντέλο για την πρόβλεψη.

Αφαιρούνται οι στήλες "num", "season" και "player" καθώς δεν αποτελούν μετρήσιμα μεγέθη αλλά ετικέτες των εγγραφών, καθώς και οι στήλες "votes_first", "points_won" και "points_max" οι οποίες όπως προαναφέρθηκε δεν μας είναι χρήσιμες για την πρόβλεψη, εφόσον η προς πρόβλεψη στήλη ορίσαμε ότι θα είναι η στήλη **award_share**.

Ξεκινάμε ανοίγοντας ένα Notebook σε Python και ξεκινάμε εισάγοντας τις βιβλιοθήκες που περιέχουν τις συναρτήσεις οι οποίες θα μας χρειαστούν στο συγκεκριμένο παράδειγμα:

```
import pyodbc
import pandas
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

Εισάγεται μία μεταβλητή με όνομα **df** η οποία είναι τύπου πλαίσιο δεδομένων (data frame) και στην οποία θα αποθηκευτούν τα δεδομένα της database. Για τη φόρτωση των δεδομένων στη df χρησιμοποιείται η συνάρτηση **read_sql** (**sql=** , **con=**) της βιβλιοθήκης pandas.

Ο κώδικας για την περίπτωση του παραδείγματος διαμορφώνεται ως εξής:

```
# Connection string to your SQL Server instance

conn_str = pyodbc.connect(
    'DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-
69EC11S\SQLEXPRESS;DATABASE=NBA;Trusted_Connection=yes')

query_str = 'SELECT * FROM dbo.MVP'

df = pandas.read_sql(sql=query_str, con=conn_str)
```


Εισάγεται μία μεταβλητή τύπου λίστας με όνομα **columns** στην οποία αποθηκεύονται τα ονόματα των στηλών του data frame που δημιουργήσαμε. Η εισαγωγή των δεδομένων γίνεται με χρήση της συνάρτησης του pandas **df.columns.tolist** όπου **df** είναι το data frame, **columns** συνάρτηση που επιστρέφει τα ονόματα των στηλών του **df** και **tolist** η εντολή που αποθηκεύει τα δεδομένα σε μορφή λίστας. Στη συνέχεια φιλτράρονται τα περιεχόμενα της λίστας **columns** ώστε να αφαιρεθεί η στήλη «Year», ως εξής:

```
# Get all the columns from the dataframe.
columns = df.columns.tolist()
print("Αρχική λίστα 'columns': ", columns)

# Filter the columns to remove ones we don't want to use in the training
columns = [c for c in columns if c not in ["num", "season", "player",
'votes_first', 'points_won', 'points_max']]
print("\nΦιλτραρισμένη λίστα 'columns': ", columns)
```

Για επαλήθευση των περιεχομένων της λίστας τυπώνουμε σε κάθε βήμα τη μεταβλητή **columns** και λαμβάνουμε τα εξής αποτελέσματα:

```
Αρχική λίστα 'columns': ['num', 'PER', 'BPM', 'season',
'player', 'win_pct', 'votes_first', 'points_won',
'points_max', 'award_share', 'G', 'MPG', 'PPG', 'RPG', 'APG',
'SPG', 'BPG', 'FGP', '_3PP', 'FTP', 'WS', 'WS48', 'fga',
'fg3a', 'fta', 'ts_pct', 'usg_pct']
Φιλτραρισμένη λίστα 'columns': ['PER', 'BPM', 'win_pct',
'award_share', 'G', 'MPG', 'PPG', 'RPG', 'APG', 'SPG', 'BPG',
'FGP', '_3PP', 'FTP', 'WS', 'WS48', 'fga', 'fg3a', 'fta',
'ts_pct', 'usg_pct']
```

Πίνακας 19 - Αρχική και φιλτραρισμένη λίστα στηλών

4.4.4. «Εκπαίδευση» των δεδομένων

Σε αυτό το στάδιο εκπαιδεύουμε το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model) και γίνονται προβλέψεις για τα δεδομένα που μας ενδιαφέρουν με χρήση του μοντέλου.

Εισάγεται μία μεταβλητή τύπου string με όνομα **target** στην οποία αποθηκεύεται το όνομα της στήλης «**award_share**» για την οποία επιθυμούμε να κάνουμε προβλέψεις.

Εισάγεται μεταβλητή με όνομα **train** στην οποία θα αποθηκευτεί το σύνολο των προς εκπαίδευση δεδομένων (**training set**). Μέσω της συνάρτησης `df.sample` επιλέγεται τυχαίο δείγμα των συνολικών δεδομένων του data frame που μας ενδιαφέρει. Η **sample** λαμβάνει τα κάτωθι δύο ορίσματα:

frac: μεταβλητή τύπου float η οποία εκφράζει το ποσοστό των συνολικών δεδομένων που επιλέγονται και λαμβάνει δεκαδικές τιμές από 0 έως και 1. Στο συγκεκριμένο παράδειγμα επιλέγεται η τιμή 0,8 που σημαίνει ότι το training set θα περιέχει το 80% του συνολικού data set.

random_state: μεταβλητή τύπου int που ορίζει ότι το δείγμα που επιλέγεται από το σύνολο των δεδομένων θα είναι σε κάθε εκτέλεση το ίδιο, εφόσον δεν αλλάζει η τιμή του random_state. Εάν δεν οριστεί καμία τιμή, τότε σε κάθε εκτέλεση η εντολή επιλέγει διαφορετικό δείγμα με τυχαίο τρόπο. Εφόσον οριστεί κάποια ακέραια τιμή, τότε σε κάθε εκτέλεση λαμβάνεται το ίδιο δείγμα που έχει αντιστοιχηθεί στη συγκεκριμένη τιμή κατά την πρώτη εκτέλεση, για την εξασφάλιση της επαναληψιμότητας της διαδικασίας.

Εισάγεται μεταβλητή με όνομα **test** θα αποθηκευτεί το σύνολο των υπόλοιπων δεδομένων (**testing set**) που δεν επιλέχθηκαν για το training set.

```
## Store the variable we'll be predicting on.  
target = "award_share"  
  
# Generate the training set. Set random_state to be able to replica  
te results.  
train = df.sample(frac=0.8, random_state=1)  
  
# Select anything not in the training set and put it in the testing  
set.  
test = df.loc[~df.index.isin(train.index)]  
  
# Print the shapes of both sets.  
print("Training set shape:", train.shape)  
print("Testing set shape:", test.shape)  
print("df set shape:", df.shape)
```

Για επαλήθευση και καλύτερη εποπτεία τυπώνονται, μέσω της συνάρτησης `shape`, οι διαστάσεις των δύο υποσυνόλων training set και testing set καθώς και του αρχικού df και λαμβάνουμε τα εξής αποτελέσματα:

```
Training set shape: (510, 27)
Testing set shape: (127, 27)
df set shape: (637, 27)
```

Πίνακας 20 - Training και test σύνολο δεδομένων

Εισάγεται η μεταβλητή **lin_model**, στην οποία αποθηκεύεται η κλάση (class) **LinearRegression()** της βιβλιοθήκης Scikit-Learn και στην οποία θα αποθηκευτεί το μοντέλο γραμμικής παλινδρόμησης που θα χρησιμοποιηθεί για την ανάλυση των δεδομένων και καλείται η μέθοδος **fit** της κλάσης **LinearRegression()** για την εισαγωγή των επιθυμητών δεδομένων. Η μέθοδος **fit** λαμβάνει τις κάτωθι παραμέτρους **X** και **Y**:

X: το σύνολο των προς εκπαίδευση δεδομένων (training data set)

Y: η προς πρόβλεψη στήλη (target values)

```
# Initialize the model class.
lin_model = LinearRegression()
print("lin_model:", lin_model)
# Fit the model to the training data.
lin_model.fit(train[columns], train[target])
```

Εκτελείται η μέθοδος **predict** της κλάσης (class) **LinearRegression()**, η οποία λαμβάνει ως όρισμα τις προς πρόβλεψη στήλες του test set και επιστρέφει τις προβλεφθείσες τιμές, οι οποίες αποθηκεύονται στη μεταβλητή **lin_predictions**.

Στη συνέχεια καλείται η συνάρτηση **mean_squared_error** που υπολογίζει το σφάλμα μεταξύ των αποτελεσμάτων της πρόβλεψης σε σχέση με τις πραγματικές τιμές του test set.

```
# Generate our predictions for the test set.
lin_predictions = lin_model.predict(test[columns])
print("Predictions:", lin_predictions)
# Compute error between our test predictions and the actual values.
lin_mse = mean_squared_error(lin_predictions, test[target])
print("Computed error:", lin_mse)
```

Εκτυπώνοντας τις τιμές της πρόβλεψης καθώς και το σφάλμα που προκύπτει λαμβάνουμε τα κάτωθι αποτελέσματα:

```
Predictions: [0.072 0.026 0.016 0.007 0.006 0.004 0.004 0.146
0.017 0.04 0.005 0.005
0.003 0.001 0.092 0.05 0.279 0.027 0.004 0.004 0.145 0.036
0.012 0.001
0.001 0.831 0.136 0.001 0.704 0.026 0.021 0.004 0.613 0.111
0.01 0.005
0.03 0.026 0.01 0.027 0.003 0.055 0.001 0.88 0.723 0.001
0.001 0.001
0.001 0.079 0.032 0.022 0.009 0.007 0.211 0.007 0.857 0.004
0.003 0.002
0.001 0.001 0.061 0.006 0.003 0.002 0.033 0.001 0.001 0.149
0.093 0.42
0.006 0.004 0.001 0.31 0.029 0.017 0.359 0.07 0.145 0.002
0.001 0.386
0.007 0.001 0.001 0.882 0.785 0.404 0.024 0.005 0.001 0.348
0.048 0.02
0.003 0.969 0.562 0.027 0.002 0.001 0.98 0.389 0.431 0.157
0.02 0.002
0.002 0.888 0.318 0.048 0.239 0.152 0.001 0.068 0.036 0.021
0.02 0.006
0.003 0.425 0.095 0.002 0.001 0.082 0.065]
Computed error: 8.723896818407452e-30
```

Πίνακας 21 - Τιμές πρόβλεψης και σφάλμα

4.4.5. Δημιουργία αποθηκευμένης διαδικασίας (Stored Procedure) που παράγει το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model)

Σε αυτό το στάδιο ενσωματώνουμε το Μοντέλο Γραμμικής Παλινδρόμησης (Linear Regression Model) που αναπτύξαμε σε Python σε μια Βάση Δεδομένων του SQL Server με χρήση του Machine Learning Services.

- Δημιουργούμε μία αποθηκευμένη διαδικασία (stored procedure) η οποία κατασκευάζει το μοντέλο μηχανικής μάθησης (machine learning model).
- Αποθηκεύουμε το μοντέλο σε έναν πίνακα της βάσης δεδομένων.
- Δημιουργούμε μία αποθηκευμένη διαδικασία η οποία κάνει προβλέψεις με χρήση του μοντέλου.
- Εκτελούμε το μοντέλο με καινούρια δεδομένα.

Η αποθηκευμένη διαδικασία που θα δημιουργήσουμε θα έχει όνομα **generate_MVP_py_model**. Αρχικά με τη συνάρτηση DROP της SQL διαγράφεται

τυχόν προηγούμενη αποθηκευμένη διαδικασία με το ίδιο όνομα και στη συνέχεια με τη συνάρτηση CREATE δημιουργείται η διαδικασία.

Το πρώτο βήμα είναι να εκτελεστεί η εντολή `sp_execute_external_script` μέσω της οποίας γίνεται η σύνδεση της SQL με τον κώδικα που περιγράφει τη διαδικασία. Η εντολή δέχεται τις κάτωθι παραμέτρους:

@language: η γλώσσα Python

@script: ο κώδικας σε Python που αναπτύχθηκε στο στάδιο της εκπαίδευσης των δεδομένων

@input_data_1: τα δεδομένα για τα οποία επιθυμούμε να γίνουν προβλέψεις, τα οποία εισάγοντας μέσω ενός ερωτήματος της SQL

@input_data_1_name: το όνομα με το οποίο θα αποθηκευτούν τα δεδομένα

@params: το μοντέλο που παράγεται

@trained_model: το μοντέλο που παράγεται

```
--  
Stored procedure that trains and generates a Python model using the  
MVP data and a linear regression algorithm  
DROP PROCEDURE IF EXISTS generate_MVP_py_model;  
go  
CREATE PROCEDURE generate_MVP_py_model (@trained_model varbinary(max)  
) OUTPUT)  
AS  
BEGIN  
    EXECUTE sp_execute_external_script  
        @language = N'Python'  
        , @script = N'  
from sklearn.linear_model import LinearRegression  
import pickle  
  
df = MVP_train_data  
  
# Get all the columns from the dataframe.  
columns = df.columns.tolist()  
print(columns)  
  
# Store the variable well be predicting on.  
target = "award_share"  
  
# Initialize the model class.
```

```
lin_model = LinearRegression()

# Fit the model to the training data.
lin_model.fit(df[columns], df[target])

# Before saving the model to the DB table, convert it to a binary object
trained_model = pickle.dumps(lin_model)'

, @input_data_1 = N'select "award_share", "PER", "BPM", "win_pct", "G", "MPG", "PPG", "RPG", "APG", "SPG", "BPG", "FGP", "_3PP", "FTP", "WS", "WS48", "fga", "fg3a", "fta", "ts_pct", "usg_pct" from dbo.MVP'
, @input_data_1_name = N'MVP_train_data'
, @params = N'@trained_model varbinary(max) OUTPUT'
, @trained_model = @trained_model OUTPUT;
END;
GO
```

Από το μενού του Azure Data Studio μπορούμε να επαληθεύσουμε ότι η αποθηκευμένη διαδικασία **generate_MVP_py_model** δημιουργήθηκε πράγματι στο φάκελο Programmability / Stored Procedures της βάσης δεδομένων:



Εικόνα 15 - Αποθηκευμένη διαδικασία generate_MVP_py_model

4.4.6. Αποθήκευση του Linear Regression Model σε ένα νέο πίνακα της Βάσης Δεδομένων

Ο πίνακας που θα δημιουργήσουμε εντός της βάσης δεδομένων για την αποθήκευση του μοντέλου θα έχει όνομα MVP_py_models. Αρχικά με τη συνάρτηση DROP της SQL διαγράφεται τυχόν προηγούμενος αποθηκευμένος πίνακας με το ίδιο όνομα και στη συνέχεια με τη συνάρτηση CREATE δημιουργείται ο νέος πίνακας:

```
DROP TABLE IF EXISTS dbo.MVP_py_models;
```

```
GO
CREATE TABLE dbo.MVP_py_models (
    model_name VARCHAR(30) NOT NULL DEFAULT('default model') PRIMARY
    KEY,
    model VARBINARY(MAX) NOT NULL
);
GO
```

Αποθηκεύουμε το μοντέλο στο νέο πίνακα ως αντικείμενο δυαδικών δεδομένων (binary object) με όνομα linear_model.

```
DECLARE @model VARBINARY(MAX);
EXEC generate_MVP_py_model @model OUTPUT;

INSERT INTO MVP_py_models (model_name, model) VALUES('linear_model',
@model);
```

Εκτελώντας το κάτωθι query επαληθεύουμε ότι πράγματι το μοντέλο έχει αποθηκευτεί στον πίνακα MVP_py_models:

```
SELECT * FROM dbo.MVP_py_models
```

	model_name	model
1	linear_model	0x800363736B6C6561726E2E6C69...

Πίνακας 22 - Αποθηκευμένο μοντέλο στον πίνακα MVP_py_models

4.4.7. Δημιουργία αποθηκευμένης διαδικασίας που κάνει προβλέψεις για τα δεδομένα

Στο τελευταίο στάδιο του παραδείγματος δημιουργούμε μια αποθηκευμένη διαδικασία με όνομα **py_predict_award_share** η οποία κάνει προβλέψεις με χρήση του εκπαιδευμένου μοντέλου και ενός νέου συνόλου δεδομένων.

```
-- 1. CREATE PROCEDURE py_predict_award_share
DROP PROCEDURE IF EXISTS py_predict_award_share;
GO
```

```
CREATE PROCEDURE py_predict_award_share (@model varchar(100))
AS
BEGIN
    DECLARE @py_model varbinary(max) = (select model from MVP_py_model
s where model_name = @model);
    EXEC sp_execute_external_script
        @language = N'Python',
        @script = N'
# Import the scikit-learn function to compute error.
from sklearn.metrics import mean_squared_error
import pickle
import pandas as pd
MVP_model = pickle.loads(py_model)
df = MVP_score_data
# Get all the columns from the dataframe.
columns = df.columns.tolist()
# variable we will be predicting on.
target = "award_share"
# Generate our predictions for the test set.
lin_predictions = MVP_model.predict(df[columns])
print(lin_predictions)
# Compute error between our test predictions and the actual values.
lin_mse = mean_squared_error(lin_predictions, df[target])
#print(lin_mse)
predictions_df = pd.DataFrame(lin_predictions)
OutputDataSet = pd.concat([predictions_df, df["award_share"], df["PER"], df["BPM"], df["win_pct"], df["G"], df["MPG"], df["PPG"], df["RPG"], df["APG"], df["SPG"], df["BPG"], df["FGP"], df["_3PP"], df["FTP"], df["WS"], df["WS48"], df["fga"], df["fg3a"], df["fta"], df["ts_pct"], df["usg_pct"]], axis=1)

, @input_data_1 = N'select "award_share", "PER", "BPM", "win_pct", "G", "MPG", "PPG", "RPG", "APG", "SPG", "BPG", "FGP", "_3PP", "FTP", "WS", "WS48", "fga", "fg3a", "fta", "ts_pct", "usg_pct" from dbo.MVP'
, @input_data_1_name = MVP_score_data
, @params = N'@py_model varbinary(max)'
, @py_model = @py_model
with result sets (("award_share_Predicted" float, " award_share " float, "PER" float, "BPM" float, "win_pct" float, "G" smallint, "MPG" float, "PPG" float, "RPG" float, "APG" float, "SPG" float, "BPG" float, "FGP" float, "_3PP" float, "FTP" float, "WS" float, "WS48" float, "fga" float, "fg3a" float, "fta" float, "ts_pct" float, "usg_pct" float));
END;
GO
```

Δημιουργούμε έναν πίνακα με όνομα py_MVP_predictions για την αποθήκευση των προβλέψεων.

```
DROP TABLE IF EXISTS [dbo].[py_MVP_predictions];
GO
--Create a table to store the predictions in
CREATE TABLE [dbo].[py_MVP_predictions](
  [award_share_Predicted] [float] NULL,
  [award_share_Actual] [float] NULL,
  [PER] [float] NULL,
  [BPM] [float] NULL,
  [win_pct] [float] NULL,
  [G] [smallint] NULL,
  [MPG] [float] NULL,
  [PPG] [float] NULL,
  [RPG] [float] NULL,
  [APG] [float] NULL,
  [SPG] [float] NULL,
  [BPG] [float] NULL,
  [FGP] [float] NULL,
  [_3PP] [float] NULL,
  [FTP] [float] NULL,
  [WS] [float] NULL,
  [WS48] [float] NULL,
  [fga] [float] NULL,
  [fg3a] [float] NULL,
  [fta] [float] NULL,
  [ts_pct] [float] NULL,
  [usg_pct] [float] NULL
) ON [PRIMARY]
GO
```

Εκτελούμε την αποθηκευμένη διαδικασία για να προβλέψουμε τις τιμές της στήλης award_share.

```
TRUNCATE TABLE py_MVP_predictions;
--Insert the results of the predictions for test set into a table
INSERT INTO py_MVP_predictions
EXEC py_predict_award_share 'linear_model';
```

Για να δούμε τις προβλέψεις εμφανίζουμε μέσω ενός query όλα τα περιεχόμενα του πίνακα py_MVP_predictions και λαμβάνουμε τις κάτωθι τιμές:

```
-- Select contents of the table
```



```
SELECT ROUND(award_share_Predicted, 3) award_share_Predicted, award_
share_Actual, PER, BPM, win_pct, G, MPG, PPG, RPG, APG, SPG, BPG, FG
P, _3PP, FTP, WS, WS48,
fga, fg3a, fta, ts_pct, usg_pct
FROM py_MVP_predictions;
```

	award_share_Predicted	award_share_Actual	PER	BPM	win_pct	G	MPG	PPG	RPG
1	0,658	0,658	25,1	8	0,75609	82	35	24,6	8
2	0,613	0,613	19,9	5,1	0,75609	82	39,5	21,2	10,9
3	0,414	0,414	25,5	5,3	0,65853	80	37,2	26,2	10,3
4	0,261	0,261	25,1	3,7	0,4878	80	40,6	27,8	14,8
5	0,12	0,12	22,9	1,6	0,63414	82	33,7	27,1	5,1
6	0,106	0,106	22	5,9	0,7317	76	33,4	20,3	6,8
7	0,077	0,077	25,2	5	0,75609	82	28	18,9	9,5
8	0,072	0,072	17	2	0,69512	79	33,1	18,8	4,6
9	0,046	0,046	14,3	-1,3	0,75609	80	35,3	13,8	2,2
10	0,028	0,028	18	0,8	0,65853	81	37,4	22,6	5,4
11	0,026	0,026	25,7	9,2	0,65853	37	37,1	21,6	8,6
12	0,022	0,022	24,3	4,6	0,34146	80	42,7	30,7	6,4
13	0,022	0,022	17,3	-1,1	0,4878	66	34,7	17,5	1,9
14	0,017	0,017	19,9	3,3	0,4756	81	36	21,9	6,8
15	0,016	0,016	14,9	-2,8	0,54878	80	30,4	15,2	2,4
16	0,016	0,016	18	1,5	0,41463	82	35,6	18,7	10,4

Πίνακας 23 - Σύγκριση τιμών πρόβλεψης με πραγματικές

Με το κάτωθι query, εμφανίζοντας όλα τα δεκαδικά ψηφία της στήλης award_share_Predicted και υπολογίζοντας το σφάλμα, δηλαδή τη διαφορά, της πρόβλεψης με τα πραγματικά δεδομένα, μπορούμε να έχουμε καλύτερη εποπτεία του αποτελέσματος:

```
SELECT (award_share_Actual - award_share_Predicted) error, award_sh
are_Predicted, award_share_Actual
FROM py_MVP_predictions
ORDER BY error DESC
```

	error	award_share_Predicted	award_share_Actual
1	2,2898349882893854E-15	0,11699999999999772	0,117
2	2,220446049250313E-15	0,9039999999999978	0,904
3	2,220446049250313E-15	0,7349999999999978	0,735
4	1,887379141862766E-15	0,18899999999999811	0,189
5	1,887379141862766E-15	0,8729999999999981	0,873
6	1,887379141862766E-15	0,7349999999999981	0,735
7	1,6974269212433057E-15	0,0009999999999983026	0,001
8	1,6930901125533637E-15	0,22799999999999832	0,228
9	1,6653345369377348E-15	0,40399999999999836	0,404
10	1,6653345369377348E-15	0,49499999999999833	0,495
11	1,6653345369377348E-15	0,26999999999999835	0,27
12	1,6653345369377348E-15	0,9859999999999983	0,986
13	1,6653345369377348E-15	0,9999999999999983	1
14	1,6306400674181987E-15	0,04999999999999837	0,05
15	1,6167622796103842E-15	0,021999999999998382	0,022
16	1,5994150448506161E-15	0,0009999999999984006	0,001

Πίνακας 24 - Σφάλμα, προβλέψεις και πραγματικές τιμές

5. Συμπεράσματα

5.1. Σωστή επιλογή συνόλου δεδομένων για την επιθυμητή ανάλυση

5.1.1. Προσδιορισμός κατάλληλων δεδομένων για το σκοπό της ανάλυσης

Κατά την εκπόνηση της εργασίας, διαπιστώθηκε στην πράξη η μεγάλη σημασία του προσδιορισμού των κατάλληλων δεδομένων για την εκάστοτε ανάλυση που επιθυμούμε να κάνουμε. Τα δεδομένα που θα επιλεγούν θα πρέπει να έχουν σχέση αιτίου – αιτιατού με τα προς πρόβλεψη αποτελέσματα, ώστε να μπορούν να εξαχθούν ορθά συμπεράσματα σχετικά με την επιρροή τους σε αυτά.

Για παράδειγμα, στην περίπτωση πρόβλεψης του MVP του NBA που παρουσιάστηκε, δε θα είχε νόημα να συμπεριληφθούν στην ανάλυση δημογραφικά δεδομένα των παικτών, όπως π.χ. η Πολιτεία καταγωγής τους, καθώς η σχέση αυτών με το ζητούμενο προς πρόβλεψη μέγεθος είναι εντελώς τυχαία.

Αντίστοιχα, στην περίπτωση της συσταδοποίησης των παικτών ανάλογα με τα αγωνιστικά τους χαρακτηριστικά, έπρεπε να επιλεγούν δεδομένα που διαφοροποιούνται ανάλογα με το τρόπο παιχνιδιού (κοντά ή μακριά από το καλάθι) κάθε παίκτη, ώστε να επηρεάζουν τον κατακερματισμό τους σε συστάδες.

5.1.2. Εύρεση συνόλων δεδομένων από διαθέσιμες πηγές

Παράλληλα με τον ορισμό των επιθυμητών δεδομένων, εξίσου σημαντική διαδικασία είναι και η εύρεση από διαθέσιμες πηγές των κατάλληλων συνόλων δεδομένων. Αυτή η διαδικασία συχνά είναι χρονοβόρα καθώς απαιτεί αρκετή έρευνα μέσα από πληθώρα διαφορετικών πηγών για τον εντοπισμό ενός πλήρους και ορθού συνόλου δεδομένων που θα εξυπηρετήσει καλύτερα την ανάλυση απαιτώντας ταυτόχρονα τη λιγότερη δυνατή προεπεξεργασία.

5.2. Σημασία προεπεξεργασίας δεδομένων

Όπως συχνά αναφέρεται στη θεωρία της εξόρυξης δεδομένων, το βασικότερο ίσως και πιο χρονοβόρο στάδιο της διαδικασίας αποτελεί η προεπεξεργασία των δεδομένων ώστε να φτάσουν στην κατάλληλη μορφή για τη χρήση τους με τα εκάστοτε μοντέλα, γεγονός που επαληθεύθηκε και στην πράξη, κατά την εκπόνηση της εργασίας.

5.2.1. Καθαρισμός συνόλου δεδομένων

Αρχικά εντοπίζονται και διαγράφονται τυχόν περιττοί χαρακτήρες που ενδεχομένως στο αρχικό σύνολο δηλώνουν κάποια ιδιότητα των δεδομένων αλλά δε χρειάζονται για την ανάλυση που επιθυμούμε να κάνουμε.

Εντοπίζονται και διαγράφονται τυχόν null τιμές και κενές στήλες, που υπάρχουν στα δεδομένα πιθανώς λόγω λάθους.

5.2.2. Μετατροπή δεδομένων στην επιθυμητή μορφή

Ακολουθεί μετατροπή των δεδομένων του αρχικού συνόλου, το οποίο συχνά πρόκειται για ένα αρχείο excel ή csv, στον κατάλληλο τύπο δεδομένων (string, integer, float κλπ) για τις διάφορες συναρτήσεις και μοντέλα που θα τα χρησιμοποιήσουν.

5.2.3. Εντοπισμός λογικών λαθών

Στη συνέχεια εντοπίζονται τυχόν μη λογικές τιμές στα δεδομένα και είτε διαμορφώνονται κατάλληλα εάν αυτό είναι δυνατόν είτε διαγράφονται εάν οφείλονται σε λάθος καταγραφή ή καταχώρηση. Πρόκειται για χρονοβόρα διαδικασία που απαιτεί γνώση του αντικειμένου που περιγράφουν τα δεδομένα και ενδελεχή έρευνα και δοκιμές για τον εντοπισμό των λογικών λαθών.

5.3. Αξία της καλής γνώσης του προς ανάλυση αντικειμένου

Κατά τη διαδικασία εντοπισμού του κατάλληλου συνόλου δεδομένων και της προεπεξεργασίας του, διαπιστώθηκε η μεγάλη σημασία και κρισιμότητα της καλής γνώσης της φυσικής έννοιας των δεδομένων που εξετάζονται καθώς και της προϋστορίας του αντικειμένου το οποίο περιγράφουν. Στις περιπτώσεις που αυτό είναι εφικτό επιταχύνεται σημαντικά η εύρεση των κατάλληλων δεδομένων και εξασφαλίζεται με μεγαλύτερη ασφάλεια η ορθότητά τους.

5.3.1. Περίπτωση «Έντι Τζόνσον»

Κατά την αρχική μελέτη των δεδομένων και για να ελεγχθεί η ορθότητα τους εκτελέστηκαν διάφορα ερωτήματα σε SQL ώστε να λάβουμε συγκεντρωτικές τιμές όπως ποιοι παίκτες έχουν πετύχει τους περισσότερους πόντους στο σύνολο της καριέρας τους. Από τα αποτελέσματα διαπιστώθηκε ότι εντός της 1^{ης} δεκάδας βρισκόταν το όνομα «Έντι Τζόνσον». Λόγω προσωπικής ενασχόλησης με το άθλημα του μπάσκετ και τα ιστορικά στατιστικά του, υπήρχε η γνώση των 10 πρώτων σκόρερ όλων των εποχών στο NBA. Παράλληλα επειδή τυχαίνει ο συγκεκριμένος παίκτης να είναι γνωστός και από τη θητεία του στο ελληνικό πρωτάθλημα τη δεκαετία του 1990, η εμφάνισή του στην 1^η δεκάδα του NBA, θεωρήθηκε παράδοξη. Αυτή η παρατήρηση οδήγησε σε περαιτέρω μελέτη των δεδομένων από την οποία προέκυψε το συμπέρασμα ότι υπήρξαν 2 διαφορετικοί παίκτες με το όνομα Έντι Τζόνσον, χωρίς όμως στο αρχικό σύνολο δεδομένων να διαχωρίζονται με κάποιον τρόπο οι εγγραφές που τους αφορούν. Ως εκ τούτου κατά την ομαδοποίηση κατά όνομα παίκτη, οι εγγραφές αυτές αντιμετωπίζονταν από την SQL ως να αφορούσαν τον ίδιο παίκτη, με αποτέλεσμα οι πόντοι και των δύο να αθροίζονται και έτσι το τελικό σύνολο να φτάνει σε επίπεδα που τον/τους κατέτασσαν στην 1^η δεκάδα όλων των εποχών.

Από τα παραπάνω ουσιαστικά εντοπίστηκε ότι η στήλη με τα ονόματα των παικτών δε θα μπορούσε να λειτουργήσει ως στήλη μοναδικού κλειδιού στην περίπτωση που επιθυμούσαμε να δουλέψουμε με δεδομένα ομαδοποιημένα ανά παίκτη ανά τις σεζόν, καθώς διαπιστώθηκε ότι υπήρχαν συνωνυμίες μεταξύ παικτών χωρίς να διαχωρίζονται σαφώς.

Έτσι λοιπόν ένα τεχνικό πρόβλημα της Βάσης Δεδομένων που μελετούσαμε και επιχειρούσαμε να δημιουργήσουμε, έγινε ορατό μόνο χάρη στην ειδική αυτή γνώση περί των συγκεκριμένων στατιστικών, αποδεικνύοντας έτσι τη μεγάλη σημασία της και τη βοήθεια που αποτελεί για τον αναλυτή αν και εφόσον είναι διαθέσιμη.

5.4. Περαιτέρω ενέργειες για τη συνέχεια της ανάλυσης

5.4.1. Συσταδοποίηση παικτών με τον αλγόριθμο K-means clustering

Algorithm

Για την κατηγοριοποίηση των παικτών ανάλογα με τα διαφορετικά χαρακτηριστικά του τρόπου παιχνιδιού τους, παρουσιάστηκε η απλή περίπτωση όπου από το αρχικό σύνολο δεδομένων και με βάση κάποια βασικά στατιστικά δεδομένα γίνεται ένας αρχικός διαχωρισμός. Η ανάλυση μπορεί στη συνέχεια να εξειδικευτεί, αναλόγως με τον εκάστοτε στόχο της ομάδας και να βασιστεί σε πιο συγκεκριμένα χαρακτηριστικά ή να αφορά μέρος του συνόλου των παικτών.

Για παράδειγμα στην ανάλυση που έγινε, η συστάδα που αφορά παίκτες με μικρή συνεισφορά στο παιχνίδι μπορεί να αφαιρεθεί, καθώς οι παίκτες που ανήκουν σε αυτή μπορεί να μην ενδιαφέρουν την έρευνα μιας ομάδας και να επαναληφθεί η διαδικασία της συσταδοποίησης για το υπόλοιπο σύνολο, ορίζοντας εκ νέου τις ζητούμενες συστάδες καθώς και τις επιθυμητές διαστάσεις του μοντέλου.

5.4.2. Πρόβλεψη MVP με τον Linear Regression Algorithm

Κατά τη διαδικασία της κατασκευής μοντέλου γραμμικής παλινδρόμησης για την πρόβλεψη του MVP, χρησιμοποιήθηκαν τα στατιστικά δεδομένα της απόδοσής του μέσα στη χρονιά. Η ανάλυση κ εδώ μπορεί να επεκταθεί ή να εξειδικευτεί ώστε να ανακαλυφθούν συσχετίσεις και με άλλα διαθέσιμα δεδομένα.

Για παράδειγμα μπορούν να χρησιμοποιηθούν βιομετρικά δεδομένα των παικτών, ώστε να εξαχθεί κάποια συσχέτιση των σωματικών τους χαρακτηριστικών με την πιθανότητα να διακριθούν κατά την ψηφοφορία για την ανάδειξη του MVP.

Βιβλιογραφία

- [1] Wikipedia, '*Data mining*'. Στο: https://en.wikipedia.org/wiki/Data_mining (προσπελάστηκε στις 2/3/2021).
- [2] Talend, '*What is Data Mining?*'. Στο: <https://www.talend.com/resources/what-is-data-mining/> (προσπελάστηκε στις 3/3/2021).
- [3] Sharma, R., '*12 Most Useful Data Mining Applications of 2021*'. Στο: <https://www.upgrad.com/blog/12-most-useful-data-mining-applications-of-2020/> (προσπελάστηκε στις 10/3/2021).
- [4] Wikipedia, '*Microsoft SQL Server*'. Στο: https://el.wikipedia.org/wiki/Microsoft_SQL_Server (προσπελάστηκε στις 6/3/2021).
- [5] Hughes A., Stedman C., '*Microsoft SQL Server*'. Στο: <https://searchdatamanagement.techtarget.com/definition/SQL-Server> (προσπελάστηκε στις 20/3/2021).
- [6] Microsoft, '*What is SQL Server Machine Learning Services with Python and R?*'. Στο: <https://docs.microsoft.com/en-us/sql/machine-learning/sql-server-machine-learning-services?view=sql-server-ver15> (προσπελάστηκε στις 22/3/2021).
- [7] Microsoft, '*What is Azure Data Studio?*'. Στο: <https://docs.microsoft.com/en-us/sql/azure-data-studio/what-is-azure-data-studio?view=sql-server-ver15> (προσπελάστηκε στις 22/3/2021).
- [8] Microsoft, '*Microsoft Clustering Algorithm*'. Στο: <https://docs.microsoft.com/en-us/analysis-services/data-mining/microsoft-clustering-algorithm?redirectedfrom=MSDN&view=asallproducts-allversions&viewFallbackFrom=sql-server-ver15> (προσπελάστηκε στις 44170).
- [9] Microsoft, '*Microsoft Logistic Regression Algorithm*'. Στο: <https://docs.microsoft.com/en-us/analysis-services/data-mining/microsoft-logistic-regression-algorithm?redirectedfrom=MSDN&view=asallproducts-allversions&viewFallbackFrom=sql-server-ver15> (προσπελάστηκε στις 44171).
- [10] Microsoft, '*Microsoft Linear Regression Algorithm*'. Στο: <https://docs.microsoft.com/en-us/analysis-services/data-mining/microsoft-linear-regression-algorithm?redirectedfrom=MSDN&view=asallproducts-allversions&viewFallbackFrom=sql-server-ver15> (προσπελάστηκε στις 44172).

[11] Microsoft, '*Python tutorial: Predict ski rental with linear regression with SQL machine learning*'. Στο: <https://docs.microsoft.com/en-us/sql/machine-learning/tutorials/python-ski-rental-linear-regression?view=sql-server-ver15>

(προσπελάστηκε στις 17/1/2021).

[12] Microsoft, '*Python tutorial: Categorizing customers using k-means clustering with SQL machine learning*'. Στο: <https://docs.microsoft.com/en-us/sql/machine-learning/tutorials/python-clustering-model?view=sql-server-ver15>

(προσπελάστηκε στις 17/1/2021).

[13] Microsoft, '*Python tutorial: Predict NYC taxi fares with binary classification*'.

Στο: <https://docs.microsoft.com/en-us/sql/machine-learning/tutorials/python-taxi-classification-introduction?view=sql-server-ver15> (προσπελάστηκε στις 17/1/2021).