



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ  
ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Ηλεκτρικών Βιομηχανικών Διατάξεων και Συστημάτων Αποφάσεων  
Εργαστήριο Συστημάτων Αποφάσεων και Διοίκησης

Διερεύνηση αιτιωδών σχέσεων στην ελληνική  
αγορά ηλεκτρικής ενέργειας με τη χρήση μεθόδων  
μηχανικής μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Δημητρίου Πανουρή

Επιβλέπων: Ιωάννης Ψαρράς  
Καθηγητής

Αθήνα, Οκτώβριος 2014





ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ  
ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Ηλεκτρικών Βιομηχανικών Διατάξεων και Συστημάτων Αποφάσεων  
Εργαστήριο Συστημάτων Αποφάσεων και Διοίκησης

**Διερεύνηση αιτιωδών σχέσεων στην ελληνική  
αγορά ηλεκτρικής ενέργειας με τη χρήση μεθόδων  
μηχανικής μάθησης**

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

**Δημητρίου Πανουρή**

**Επιβλέπων:** Ιωάννης Ψαρράς  
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29/10/2014.

.....  
Ιωάννης  
Ψαρράς  
Καθηγητής

.....  
Δημήτριος Ασκούνης  
Αναπληρωτής  
Καθηγητής

.....  
Βασίλειος  
Ασημακόπουλος  
Καθηγητής

Αθήνα, Οκτώβριος 2014

.....  
**Δημήτριος Πανουρής**  
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ  
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ  
ΥΠΟΛΟΓΙΣΤΩΝ

Τομέας Ηλεκτρικών Βιομηχανικών Διατάξεων και Συστημάτων Αποφάσεων  
Εργαστήριο Συστημάτων Αποφάσεων και Διοίκησης

©2014 Δημήτριος Πανουρής  
Creative Commons Attribution 4.0 International License  
This work is licensed under the Creative Commons Attribution 4.0 International  
License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>





*The best way to predict the future is to invent it.*  
*Theodore Hook*



# Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή κ. Ιωάννη Ψαρρά για την εμπιστοσύνη που μου έδειξε όλο το διάστημα της γνωριμίας μου μαζί του. Η δυνατότητα που μου έδωσε να γίνω βοηθός στο μάθημα "Παιγνια Αποφάσεων" ήταν καθοριστικής σημασίας για μένα και για την εξέλιξή μου στο πολυτεχνείο. Το να περατώσω την διπλωματική μου στο Εργαστήριο Συστημάτων Αποφάσεων και Διοίκησης ήταν ένα φυσικό επακόλουθο.

Στα πλαίσια της διπλωματικής θα ήθελα να ευχαριστήσω τον κ. Σωτήρη Παπαδέλη για το εξαίρετο θέμα πάνω στο οποίο επικεντρώθηκε η έρευνά μας όπως και για τις συμβολές και την καθοδήγησή του καθ' όλη την διάρκεια της διπλωματικής. Επίσης τον ευχαριστώ θερμά για την ελευθερία που μου έδωσε στο να επιλέξω τα εργαλεία και τον τρόπο εργασίας που έγινε η ανάλυση.

Τέλος ευχαριστώ τους φίλους μου, τη σύντροφό μου και την οικογένειά μου για την αγάπη και υποστήριξη τους όλο αυτό τον καιρό.



# Περίληψη

Η μηχανική μάθηση είναι μία περιοχή της επιστήμης των υπολογιστών που ασχολείται με την κατασκευή και μελέτη συστημάτων που μπορούν μέσω αυτοματοποιημένης ανάλυσης να μάθουν από τα δεδομένα από τα οποία τροφοδοτούνται.

Στην παρούσα διπλωματική εξετάζουμε τρεις αλγορίθμους του κλάδου της μηχανικής μάθησης, Random Forest, MARS και CART, και με βάση αυτούς κάνουμε μία ανάλυση στη χονδρική ελληνική αγορά ηλεκτρικής ενέργειας. Πιο συγκεκριμένα, δημιουργούμε μοντέλα με βάση χαρακτηριστικά της ελληνικής αγοράς και κάνουμε προβλέψεις του SMP, το οποίο αποτελεί την τιμή της ηλεκτρικής ενέργειας στη χονδρεμπορική αγορά. Ταυτόχρονα μελετάμε τις σχέσεις μεταξύ των μεταβλητών στον καθορισμό της τιμής όπως και τις σημαντικότητές τους και πώς αυτές επηρεάζονται. Ιδιαίτερο βάρος δίνεται στο φυσικό αέριο και στη σχέση του με το SMP.

Στο κεφάλαιο 1 κάνουμε μία εισαγωγή και εξηγούμε την ορολογία που θα χρησιμοποιηθεί. Στα κεφάλαια 2 έως 4 γίνεται μία θεωρητική ανάλυση των αλγορίθμων που χρησιμοποιούνται. Στο κεφάλαιο 5 μελετάμε την ελληνική χονδρική αγορά ηλεκτρικής ενέργειας, τα χαρακτηριστικά της και τις σχέσεις μεταξύ τους. Στο κεφάλαιο 6 γίνεται η προεπεξεργασία των δεδομένων. Στο κεφάλαιο 7 γίνεται ανάλυση σε ολόκληρη την χρονοσειρά με τυχαία δείγματα ώστε να δούμε την ικανότητα πρόβλεψης των αλγορίθμων όπως και τις σημαντικότητες των μεταβλητών στον καθορισμό της τιμής. Στο κεφάλαιο 8 γίνεται ανάλυση της χρονοσειράς μέσω lagged μεταβλητών ώστε να βελτιστοποιήσουμε τα μοντέλα πρόβλεψης. Επίσης εξετάζονται οι σχέσεις μεταξύ των μεταβλητών μέσω partial dependence plots. Στο κεφάλαιο 9 παραθέτουμε τα συμπεράσματα της εργασίας.

Για την ανάλυση χρησιμοποιήθηκε η γλώσσα Python σε περιβάλλον Anaconda με εκτεταμένη χρήση των βιβλιοθηκών scikit-learn, py-earth, numpy, pandas, matplotlib. Η συγγραφή της διπλωματικής έγινε σε MiKTeX στο περιβάλλον Texmaker.

## Λέξεις Κλειδιά

Μηχανική μάθηση, CART, Random Forests, MARS, System Marginal Price, ελληνική αγορά ηλεκτρικής ενέργειας, φυσικό αέριο



# Abstract

Machine learning is a subfield of computer science that deals with the construction and study of algorithms that can learn from data by building an adaptive model based on inputs.

In this thesis we examine three machine learning algorithms, Random Forest, MARS and CART, and we perform an analysis on the Greek wholesale electricity market. More specifically, based on features of the Greek electricity market, we make projections of the System Marginal Price, which is the price of electricity on the Greek wholesale market. Moreover, we study the most important features that determine the price and the relationships between them. Particular emphasis is placed on natural gas and its relation to the SMP.

In chapter 1, which is introductory, we explain the terminology that will be used in later chapters. In chapters 2-4 we provide a theoretical analysis of the algorithms. In chapter 5 we study the Greek wholesale power market, its features and the relationships between them. In chapter 6 we preprocess the data. In chapter 7 we present an analysis with random samples across the entire series, in order to evaluate the predictive power of the algorithms as well as the most important features that define the price. In chapter 8 we analyze the time series using lagged variables in order to optimize the predictive models. We also examine the relationships between the most important features via partial dependence plots. In Chapter 9 we present the conclusions of this dissertation.

For the analysis we used the Python programming language and the Anaconda IDE. We made extensive use of the packages scikit-learn, py-earth, numpy, pandas, matplotlib. The thesis was written in MiKTeX in the environment Texmaker.

## Keywords

Machine learning, CART, Random Forests, MARS, System Marginal Price, Greek wholesale electricity market, natural gas



# Περιεχόμενα

Ευχαριστίες	1
Περίληψη	3
Abstract	5
Περιεχόμενα	9
Κατάλογος Σχημάτων	11
<b>I Θεωρητικό Υπόβαθρο</b>	<b>13</b>
<b>1 Εισαγωγή</b>	<b>15</b>
1.1 Τί είναι η μηχανική μάθηση . . . . .	15
1.2 Ορολογία . . . . .	16
1.3 Διαδικασία για επιβλεπόμενη μάθηση . . . . .	17
<b>2 CART</b>	<b>23</b>
2.1 Εισαγωγή . . . . .	23
2.2 Κέρδος πληροφορίας . . . . .	25
2.3 Αλγόριθμος διαχωρισμού . . . . .	27
2.4 Cost Complexity Pruning . . . . .	29
2.5 Cross Validation . . . . .	30
2.6 Παρατηρήσεις . . . . .	30
<b>3 MARS</b>	<b>33</b>
3.1 Εισαγωγή . . . . .	33
3.2 MARS και Normal Regression . . . . .	33
3.3 Περιγραφή της μεθόδου . . . . .	35
3.4 Generalized Cross Validation . . . . .	37
3.5 Ο αλγόριθμος MARS συνοπτικά . . . . .	38
3.6 Παρατηρήσεις . . . . .	39

<b>4 Random Forests</b>	<b>41</b>
4.1 Εισαγωγή . . . . .	41
4.2 Περιγραφή της μεθόδου . . . . .	41
4.3 Αλγόριθμος Random Forests . . . . .	44
4.4 Out of Bag Samples . . . . .	45
4.5 Feature Importances . . . . .	45
4.6 Random Forests and Overfitting . . . . .	46
<b>II Ανάλυση αγοράς ηλεκτρικής ενέργειας</b>	<b>47</b>
<b>5 Αγορά Ηλεκτρικής Ενέργειας</b>	<b>49</b>
5.1 Εισαγωγή . . . . .	49
5.2 Variable Cost Recovery Mechanism και Capacity Adequacy Mechanism . . . . .	49
5.3 ΑΠΕ και καμπύλη προσφοράς . . . . .	51
5.4 Παράμετροι που οδηγούν τις τιμές . . . . .	52
5.5 Φυσικό Αέριο . . . . .	52
5.6 Αιτιώδεις σχέσεις μεταξύ μεταβλητών . . . . .	53
<b>6 Προεπεξεργασία δεδομένων</b>	<b>55</b>
6.1 Δεδομένα . . . . .	55
6.1.1 SMP . . . . .	55
6.1.2 Χαρακτηριστικά . . . . .	56
6.2 Προεπεξεργασία . . . . .	57
6.2.1 Δημιουργία τελικού αρχείου δεδομένων . . . . .	57
6.2.2 Χειρισμός missing values . . . . .	58
6.2.3 Train and Validation Sets . . . . .	59
<b>7 Ανάλυση δεδομένων και αλγορίθμων</b>	<b>61</b>
7.1 Εισαγωγή . . . . .	61
7.2 Multicollinearity . . . . .	61
7.3 Σφάλματα . . . . .	62
7.4 Random Forests . . . . .	63
7.4.1 Εκτίμηση σφάλματος . . . . .	63
7.4.2 Cross Validation . . . . .	64
7.4.3 Πραγματικές - εκτιμηθείσες τιμές SMP . . . . .	67
7.4.4 Feature Importances . . . . .	68
7.5 CART (Regression Trees) . . . . .	69
7.5.1 Εκτίμηση σφάλματος . . . . .	69
7.5.2 Cross Validation . . . . .	71
7.5.3 Πραγματικές - εκτιμηθείσες τιμές SMP . . . . .	72
7.5.4 Feature Importances . . . . .	72
7.5.5 Τελικό Δέντρο . . . . .	74

<b>ΠΕΡΙΕΧΟΜΕΝΑ</b>	<b>9</b>
7.6 MARS . . . . .	76
7.6.1 Forward Pass . . . . .	76
7.6.2 Pruning Pass . . . . .	77
7.6.3 Τελικό μοντέλο . . . . .	78
7.6.4 Πραγματικές - εκτιμηθείσες τιμές SMP . . . . .	78
<b>8 Ανάλυση Χρονοσειράς</b>	<b>81</b>
8.1 Εισαγωγή . . . . .	81
8.2 Residuals και lagged values . . . . .	81
8.3 Επιλογή lagged μεταβλητών . . . . .	84
8.4 Ανάλυση Χρονοσειράς . . . . .	89
8.4.1 Διαφορετικά train και validation sets σε όλη την χρονοσειρά	89
8.4.2 Διαφορετικά train sets - ίδιο test set . . . . .	92
8.4.3 Μικρά train-test sets σε ολη την χρονοσειρά . . . . .	92
8.5 Partial Dependence Plots . . . . .	97
<b>9 Συμπεράσματα - Προοπτικές</b>	<b>99</b>
<b>Βιβλιογραφία</b>	<b>101</b>
<b>Ιστότοποι</b>	<b>105</b>



# Κατάλογος Συνημάτων

1.1	Cross Validation και Residual error . . . . .	20
2.1	Παράδειγμα recursive partitioning με δύο predictors . . . . .	24
2.2	Περιοχές στο επίπεδο $X_1, X_2$ μετά από recursive partition . . . . .	24
2.3	Δυαδικό δέντρο μετά από recursive partition . . . . .	25
2.4	Κέρδος πληροφορίας σε διαχριτή κατανομή. (a) Αρχικό σύνολο δεδομένων $S$ . (b) Μετά από οριζόντιο διαχωρισμό. (c) Μετά από κατακόρυφο διαχωρισμό. . . . .	26
3.1	MARS και Linear Regression . . . . .	34
3.2	Οι συναρτήσεις βάσης $(x - t)_+$ και $(t - x)_+$ του MARS . . . . .	35
3.3	Παράδειγμα συνάρτησης ως γινόμενο συναρτήσεων βάσης . . . . .	37
4.1	Εκπαίδευση ενός δυαδικού δέντρου . . . . .	42
4.2	Εξόδος ως το σύνολο (ensemble) εκπαίδευσης . . . . .	43
5.1	Χρηματικές ροές στην ελληνική αγορά ηλεκτρικής ενέργειας . . . . .	50
5.2	Καμπύλη προσφοράς ελληνικής αγοράς ηλεκτρικής ενέργειας . . . . .	51
5.3	Αιτιώδεις σχέσεις στην ελληνική αγορά ηλεκτρικής ενέργειας . . . . .	54
6.1	SMP . . . . .	56
6.2	Χαρακτηριστικά μεταβλητών συστήματος . . . . .	58
7.1	Random Forest: Accuracy . . . . .	65
7.2	Random Forest: Mean absolute error . . . . .	66
7.3	Random Forest: Mean squared error . . . . .	66
7.4	Random Forest: Διαφορά real-expected τιμών του SMP . . . . .	67
7.5	Random Forest: Feature Importances . . . . .	68
7.6	Tree Regressor: Accuracy . . . . .	70
7.7	Tree Regressor: Mean absolute error . . . . .	70
7.8	Tree Regressor: Mean squared error . . . . .	71
7.9	Tree Regressor: Διαφορά real-expected τιμών του SMP . . . . .	72
7.10	Tree Regressor: Feature Importances . . . . .	73
7.11	Δέντρο αλγορίθμου Tree Regressor - βάθος 4 . . . . .	75

7.12 MARS: Forward Pass . . . . .	76
7.13 MARS: Pruning Pass . . . . .	77
7.14 MARS: Πραγματικές και εκτιμηθείσες τιμές του SMP . . . . .	78
7.15 MARS: Earth Model . . . . .	79
8.1 Residuals Example . . . . .	82
8.2 Χρονοσειρά SMP . . . . .	82
8.3 Variance for different lagged values . . . . .	85
8.4 Cross Validation for different lagged values . . . . .	86
8.5 MAE for different lagged values . . . . .	86
8.6 Residuals for different lagged values . . . . .	87
8.7 Random Forest - no lag . . . . .	88
8.8 Random Forest - lag έως t-2 . . . . .	88
8.9 MARS - lag έως t-2 . . . . .	89
8.10 Ανάλυση δεδομένων 1 . . . . .	90
8.11 Variance - Ανάλυση 1η . . . . .	91
8.12 MAE - Ανάλυση 1η . . . . .	91
8.13 Ανάλυση δεδομένων 2 . . . . .	92
8.14 Variance - Ανάλυση 2η . . . . .	93
8.15 Random Forest και MARS - Real - Predicted . . . . .	93
8.16 Ανάλυση δεδομένων 3 . . . . .	94
8.17 Variance - Ανάλυση 3η . . . . .	95
8.18 MAE - Ανάλυση 3η . . . . .	95
8.19 MARS Στιγμιότυπο - Ανάλυση 3η . . . . .	96
8.20 Random Forest και MARS - Real - Predicted Ανάλυση 3 . . . . .	96
8.21 Partial Dependence - Target Features . . . . .	97
8.22 Partial Dependence of SMP on load_forecast and ngas . . . . .	98

# Μέρος Ι

## Θεωρητικό Υπόβαθρο



# Κεφάλαιο 1

## Εισαγωγή

### 1.1 Τί είναι η μηχανική μάθηση

Η μηχανική μάθηση ή machine learning αφορά μεθόδους με τις οποίες ένας υπολογιστής μπορεί να αποκτήσει γνώση για ένα σύνολο δεδομένων μετά από ανάλυση τους. Σε κάθε πρόβλημα το οποίο έχει τα εξής χαρακτηριστικά, η μηχανική μάθηση μπορεί να μας βοηθήσει να εξάγουμε χρήσιμα συμπεράσματα [5]:

- Το πρόβλημα παρουσιάζει ένα μοτίβο
- Δεν μπορούμε να περιγράψουμε το μοτίβο αυτό με μαθηματικό τρόπο
- Έχουμε δεδομένα για το πρόβλημα

Πολλές φορές μία απλή επιμεώρηση των δεδομένων δεν μπορεί να μας δώσει σαφείς απαντήσεις για το πρόβλημά μας, ούτε μπορούμε να βρούμε μία συνάρτηση που να παράγει πάντα το σωστό αποτέλεσμα, παρά το γεγονός ότι παρατηρούμε να κυριαρχεί ένα συγκεκριμένο πρότυπο στην έξοδο του συστήματος. Σε αυτό το σημείο η μηχανική μάθηση παίζει έναν χρίσμα ρόλο, αφού μπορεί να μας παράξει ένα μοντέλο για το σύστημά μας, χωρίς παράλληλα να του έχουμε ορίσει εμείς ποιό μοντέλο είναι αυτό, ούτε ποιά θα είναι η τελική συνάρτηση. Το μοντέλο παράγεται μέσα από διάφορους αλγορίθμους και περιγράφει το σύστημα με έναν ικανοποιητικό τρόπο, με αποτέλεσμα να μας δίνεται η δυνατότητα να εξάγουμε χρήσιμα συμπεράσματα τα οποία θα είχαμε μεγαλύτερη δυσκολία να παράξουμε με άλλες μεθόδους.

Η μηχανική μάθηση χρησιμοποιείται από στατιστικολόγους, μηχανικούς, επιστήμονες υπολογιστών έως επιχειρηματίες ή πολιτικούς. Συγχεκριμένα για τον κλάδο της επιχειρηματικότητας, η μηχανική μάθηση μπορεί να αποβεί καθοριστικής σημασίας για μία εταιρία, καθώς δημιουργεί αξία, αναλύοντας δεδομένα του κλάδου στον οποίο η εταιρία εδρεύει, βελτιώνοντας την λήψη αποφάσεων και ελαχιστοποιώντας τους κινδύνους. Επίσης, επιτρέπει στην επιχείρηση να προσαρμόσει με ακρίβεια τα προϊόντα ή τις υπηρεσίες της, ώστε να ανταποκρίνονται στις ανάγκες των πελατών της και, ως εκ τούτου, είναι ζωτικής σημασίας για την ανταγωνιστική θέση της εταιρίας στην αγορά.

## 1.2 Ορολογία

Σε κάθε σύστημα που θα εφαρμόσουμε μεθόδους μηχανικής μάθησης έχουμε ένα σύνολο εισόδων (inputs) οι οποίες επηρεάζουν μία ή περισσότερες εξόδους (outputs). Ο στόχος είναι να χρησιμοποιήσουμε τις εισόδους ώστε να προβλέψουμε τις τιμές των εξόδων. Η διαδικασία αυτή ονομάζεται επιβλεπόμενη μάθηση (supervised learning). Οι είσοδοι είναι οι ανεξάρτητες μεταβλητές του συστήματος και συναντώνται ως predictors ενώ οι έξοδοι είναι οι εξαρτημένες μεταβλητές ή responses. Σε περίπτωση που δεν έχουμε δεδομένη έξοδο, τότε το πρόβλημα είναι μη επιβλεπόμενο και χρησιμοποιούμε μεθόδους μη επιβλεπόμενης μάθησης (unsupervised learning).

Στους αλγορίθμους για επιβλεπόμενη μάθηση περιέχονται τα δέντρα απόφασης (decision trees), οι μέθοδοι bagging, boosting και random forest, η λογιστική παλινδρόμιση, όπως επίσης τα νευρωνικά δίκτυα και οι μηχανές διανυσματικής υποστήριξης (Support Vector Machines, SVM). Στους αλγορίθμους για μη επιβλεπόμενη μάθηση περιέχονται οι μέθοδοι ομαδοποίησης (clustering), blind signal separation όπως και κρυφά Μαρκοβιανά μοντέλα (hidden Markov models).

Η έξοδος διαιφέρει ανάλογα με το αν το αποτέλεσμα είναι ποσοτικό (quantitative) ή ποιοτικό (qualitative). Λέμε πως η έξοδος παίρνει ποσοτικές τιμές, όταν οι τιμές αυτές μπορούν να είναι συνεχής σε ένα φάσμα αριθμών ή να είναι διατεταγμένες. Οι τιμές είναι ποιοτικές όταν παίρνουν μη διατεταγμένες μεταβλητές και μπορούμε να τις κατηγοριοποιήσουμε σε κλάσεις. Έτσι, τα μοντέλα πρόβλεψης χωρίζονται σε δύο κατηγορίες: παλινδρόμησης (regression) όταν θέλουμε να προβλέψουμε ποσοτικές μεταβλητές και ταξινόμησης (classification) όταν θέλουμε να προβλέψουμε ποιοτικές.

Οι ποιοτικές μεταβλητές μπορούν να αναπαρασταθούν από κωδικούς. Για παράδειγμα αν έχουμε μόνο δύο κλάσεις (ή κατηγορίες) όπως 'επιτυχία' ή 'αποτυχία' μπορούμε να τις αναπαραστήσουμε με δυαδικό bit 0 ή 1. Οι κωδικοί αυτοί ονομάζονται και 'στόχοι' (targets). Όταν έχουμε περισσότερες από δύο κατηγορίες πολλοί εναλλακτικοί τρόποι είναι διαθέσιμοι. Ο πιο διαδεδομένος τρόπος είναι μέσω ψευδομεταβλητής (dummy variable). Μέσω αυτού μία ποιοτική μεταβλητή  $K$ -κλάσης μπορεί να αναπαρασταθεί από έναν πίνακα με  $K$ -bits όπου μόνο ένα είναι 'ανοιχτό' κάθε στιγμή [4].

Η είσοδος θα αναπαρίσταται από την μεταβλητή  $X$ . Αν το  $X$  είναι διάνυσμα, τότε μπορούμε να έχουμε πρόσβαση στα στοιχεία του  $X_j$ . Η κάθε μεταβλητή  $x_j$  θα έχει  $p$ -χαρακτηριστικά (features). Επομένως μπορούμε να αναπαραστήσουμε την είσοδο μας ως έναν πίνακα  $N \times p$ . Η απόκριση θα συμβολίζεται με  $Y$ . Οι τιμές που παρατηρήθηκαν θα συμβολίζονται με μικρό γράμμα ως  $x_i$ .

Για παράδειγμα, έστω ότι έχουμε το εξής πρόβλημα classification: Έχουμε κάποια δεδομένα επιβατών του τιτανικού, όπως όνομα, επώνυμο, ημερομηνία γέννησης, φύλο, passenger class, και άλλες πληροφορίες, και έχουμε και ως δεδομένο αν επέζησαν ή όχι. Τότε, η είσοδος μας θα ήταν τα  $N$ -δεδομένα επιβατών, όπου κάθε επιβάτης θα είχε  $p$ -χαρακτηριστικά, σαν αυτά που αναφέρθηκαν πιο πριν. Η έξοδος θα είχε  $N$  δεδομένα με δύο κλάσεις: επέζησε (0) ή πέθανε (1).

Έστω τώρα ότι μας έρχονται νέα δεδομένα επιβατών, για τα οποία δεν γνωρίζουμε

αν ο επιβάτης επέζησε ή όχι. Τότε μπορούμε μέσω μεθόδων μηχανικής μάθησης να προβλέψουμε σε ποιά κλάση θα ανήκει το νέο δεδομένο που μας ήρθε (1 ή 0) με μία καλή προσέγγιση. Επίσης, μπορούμε να βρούμε ποιά χαρακτηριστικά είναι πιο σημαντικά στο να καθορίσουν αν ο επιβάτης επέζησε ή όχι, αφού κάποια χαρακτηριστικά θα παίζουν ένα σημαντικότερο ρόλο σε σχέση με κάποια άλλα (για παράδειγμα το φύλο ή το passenger class μπορεί να είναι πιο σημαντικά από το όνομα του επιβάτη στον καθορισμό της τελικής εξόδου). Το παράδειγμα με τον τιτανικό είναι ένα από τα βασικά προβλήματα που μπορεί να λύσει κάποιος που ξεκινάει με το machine learning ώστε να εξασκηθεί (6).

Μπορούμε τώρα να ορίσουμε με έναν πολύ γενικό τρόπο την μηχανική μάθηση: Έχοντας δεδομένα για ένα διάνυσμα εισόδου X, κάνε μία καλή πρόβλεψη για την έξοδο Y, μέσω κάποιων κανόνων πρόβλεψης. Για την κατασκευή των κανόνων είναι απαραίτητα κάποια δεδομένα ( $x_i, y_i$ ), τα οποία ονομάζονται δεδομένα εκμάθησης (trainining data). Η ακρίβεια του μοντέλου θα είναι:

$$\text{Accuracy} = \frac{\text{Number of Correct Classifications}}{\text{Total Number of Test Cases}} \quad (1.1)$$

### 1.3 Διαδικασία για επιβλεπόμενη μάθηση

Στην παρούσα διπλωματική θα ασχοληθούμε με μειόδους επιβλεπόμενης μάθησης. Τα βήματα που ακολουθούμε είναι τα παρακάτω (5):

#### 1. Προετοιμασία Δεδομένων

Σε έναν πίνακα δεδομένων X, κάθε γραμμή του πίνακα θα αντιστοιχεί σε μία παρατήρηση ενώ κάθε στήλη του πίνακα αντιστοιχεί σε έναν predictor. Σημεία χωρίς τιμές (missing values) αναπαρίστανται και αυτά διότι ύστερα θα αντιμετωπιστούν με ειδικό τρόπο. Σε αυτό το στάδιο προετοιμάζεται το σετ εκπαίδευσης από το οποίο θα γίνει εξαγωγή της ζητούμενης συνάρτησης.

#### 2. Επιλογή Αλγορίθμου

Η επιλογή μπορεί να γίνει ανάλογα με τα ιδιαίτερα χαρακτηριστικά του κάθε αλγορίθμου και τον σκοπό που θέλουμε να πετύχουμε. Κάποια βασικά χαρακτηριστικά είναι:

- Ακρίβεια προβλέψεων
- Ταχύτητα εκτέλεσης αλγορίθμου
- Χρήση της μνήμης
- Ευκολία ερμηνείας αλγορίθμου

Στον πίνακα 1.1 παρουσιάζονται χαρακτηριστικά δημοφιλών αλγορίθμων επιβλεπόμενης μάθησης.

Αλγόριθμος	Ακρίβεια	Ταχύτητα	Χρήση Μνήμης
Trees	Εξαρτάται	Γρήγορη	Χαμηλή
SVM	Υψηλή	Εξαρτάται	Εξαρτάται
Naive Bayes	Χαμηλή	Εξαρτάται	Εξαρτάται
Nearest Neighbor	Εξαρτάται	Γρήγορη	Χαμηλή

Πίνακας 1.1: Σύγκριση χαρακτηριστικών αλγορίθμων επιβλεπόμενης μάθησης

### 3. Ταιριασμα με μοντέλο

Ανάλογα με τον αλγόριθμο που επιλέγουμε, έχουμε και αντίστοιχη συνάρτηση ταιριάσματος.

### 4. Επιλογή μεθόδου επικύρωσης

Οι τρεις κύριες μέθοδοι που εξετάζουν την ακρίβεια του τελικού μοντέλου είναι:

#### Resubstitution error

Το ποσοστό σφάλματος (resubstitution error) είναι μία εκτίμηση του σφάλματος βασιζόμενη στις διαφορές μεταξύ προβλεψθέντων και πραγματικών τιμών σε ένα μοντέλο μάθησης. Η απόλυτη τιμή κάθε σφάλματος προσμετράται και τελικά η διάμεσος των τιμών αυτών μας δίνει το απόλυτο υπολειμματικό σφάλμα (absolute residual error) (2).

Τα δεδομένα εκπαίδευσης χρησιμοποιούνται τα ίδια για εκμάθηση οπότε οποιαδήποτε εκτίμηση της απόδοσης θα μετριέται αισιόδοξα. Ως αποτέλεσμα, το σφάλμα αυτό μπορεί να χρησιμοποιηθεί ως άνω όριο σφάλματος σε δοκιμαστικά δεδομένα. Επίσης αν το σφάλμα αυτό είναι γενικά μικρό, έχουμε μία ένδειξη πως το σύστημα μας έχει καλή απόδοση αλλά δεν μπορεί να λειτουργήσει πιο γενικευμένα, ενώ αν το σφάλμα είναι μεγαλύτερο, το σύστημα μπορεί να παρουσιάζει μεγαλύτερη διακύμανση ως προς την λειτουργία του.

#### Cross Validation error

Όπως ειπώθηκε και πριν, το πρόβλημα με το ποσοστό σφάλματος είναι πως δεν μπορεί να δώσει καλές ενδείξεις για την επίδοση ενός συστήματος μάθησης όταν του ζητείται να κάνει προβλέψεις πάνω σε δεδομένα που δεν έχει ήδη δει. Ένας τρόπος να αντιμετωπίσουμε το πρόβλημα αυτό είναι να μην χρησιμοποιήσουμε ολόκληρο το σύνολο δεδομένων στη διαδικασία μάθησης. Αυτό μπορεί να πραγματοποιηθεί αφαιρώντας δεδομένα πριν την έναρξη της διαδικασίας και παίρνοντας τα δεδομένα αυτά ως εισόδους για να τεστάρουμε την απόδοση του συστήματος, συγχρίνοντας την έξοδο του μοντέλου με την πραγματική (1). Με αυτό τον τρόπο έχουμε ένα 'test model' κατά την φάση εκπαίδευσης του συστήματος το οποίο μας δίνει μία καλή ιδέα για το πώς το σύστημα θα συμπεριφερθεί όταν θα έρθουν δεδομένα από ένα άλλο ανεξάρτητο σύνολο [6].

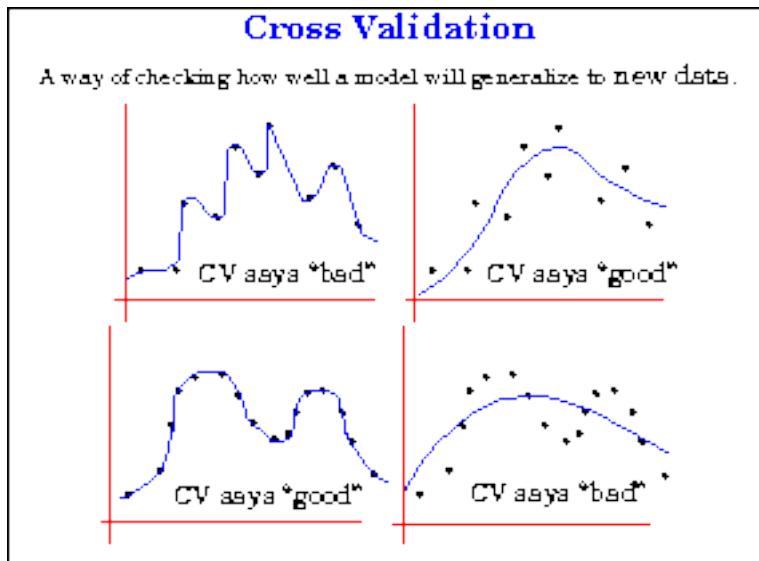
Η μέθοδος holdout είναι η απλούστερη μέθοδος για cross validation. Τα δεδομένα χωρίζονται σε δύο σύνολα, το σύνολο εκπαίδευσης (training set) και το σύνολο ελέγχου (testing set). Η συνάρτηση παράγεται χρησιμοποιώντας μόνο το σύνολο εκπαίδευσης και ύστερα χρησιμοποιείται για να προβλέψει τις τιμές εξόδου του συνόλου ελέγχου, χωρίς όμως να τις γνωρίζει. Για την αξιολόγηση του μοντέλου χρησιμοποιούμε το απόλυτο υπολειμματικό σφάλμα. Η μέθοδος holdout παράγει συνήθως καλύτερα αποτελέσματα χωρίς να αυξάνει τον χρόνο εκτέλεσης αλλά από την άλλη μεριά η συνάρτηση εξόδου εξαρτάται σε μεγάλο βαθμό από ποιά σημεία θα πάρουμε στο σύνολο εκπαίδευσης και ποιά στο σύνολο ελέγχου, οπότε μπορούμε να έχουμε μεγάλες αποκλίσεις στην συμπεριφορά της συνάρτησης σε διαφορετικές εκτελέσεις του αλγορίθμου [3].

Μία βελτίωση της μεθόδου holdout είναι το K-fold cross validation στο οποίο το σύνολο δεδομένων μας διαιρείται σε k υποσύνολα στα οποία εφαρμόζουμε την μέθοδο holdout k φορές. Κάθε φορά, ένα από τα k υποσύνολα χρησιμοποιείται σαν σύνολο ελέγχου ενώ τα υπόλοιπα k-1 υποσύνολα ως σύνολο μάθησης. Έστερα υπολογίζεται το μέσο σφάλμα ανάμεσα στις k δοκιμές. Με την παραπάνω διαδικασία δεν έχει τόση σημασία ο τρόπος με τον οποίο χωρίζουμε το σύνολο δεδομένων μας, αφού κάθε δεδομένο θα χρησιμοποιηθεί ως δεδομένο μάθησης k-1 φορές και ως ελέγχου ακριβώς μία φορά. Ως αποτέλεσμα, η απόκλιση μειώνει όσο το k αυξάνει. Το πρόβλημα με αυτή την μέθοδο είναι πως ο αλγόριθμος πρέπει να τρέξει k φορές οπότε θα πρέπει να κάνει k-υπολογισμούς για να παράξει αποτέλεσμα.

Στην περίπτωση που πάρουμε τον αλγόριθμο K-fold cross validation και τον εφαρμόζουμε N φορές, όσα δηλαδή είναι τα δεδομένα μας, τότε θα έχουμε τον αλγόριθμο Leave-one-out cross validation (LOO). Τα αποτελέσματα που παράγει η διαδικασία αυτή είναι πολύ ικανοποιητικά, αλλά υπολογιστικά ακριβά. Έχουν αναπτυχθεί τεχνικές με βάρη που μπορούν να κάνουν τους αλγορίθμους αυτούς προσιτούς από υπολογιστική άποψη [9].

Με το cross validation μπορούμε να μειώσουμε προβλήματα που ίσως προκύψουν στην έξοδο του συστήματος. Ένα παράδειγμα είναι το overfitting, το οποίο συμβαίνει όταν το μοντέλο μας περιγράφει κάποιον τυχαίο θόρυβο ή σφάλμα αντί να παράγει την επιιστροφή έξοδο. Αυτό μπορεί να συμβεί όταν το μέγεθος του συνόλου εκπαίδευσης είναι πολύ μικρό ή όταν ο αριθμός των παραμέτρων του μοντέλου είναι πολύ μεγάλος. Η διαδικασία μάθησης βελτιστοποιεί τις παραμέτρους του μοντέλου ώστε να το κάνει να ταιριάζει στα δεδομένα του συνόλου μάθησης όσο καλύτερα γίνεται [30].

Στο Σχήμα 1.1 έχουμε ένα παράδειγμα στο οποίο βλέπουμε την ισχύ του cross validation έναντι του residual error. Στα δύο πάνω γραφήματα τα δεδομένα έχουν θόρυβο και το cross validation προτείνει μία συνάρτηση που έχει υποστεί εξομάλυνση ενώ στα κάτω γραφήματα ο θόρυβος δεν υπάρχει και το cross validation θα δημιουργήσει μία συνάρτηση με μικρή εξομάλυνση.



Σχήμα 1.1: Cross Validation και Residual error

### Out of bag error

Η τεχνική αυτή χρησιμοποιείται με μεθόδους bagging ή άλλιως bootstrap aggregation σε δέντρα όπως random forests και όταν παρουσιαστεί αναλυτικά σε επόμενο κεφάλαιο. Η λογική της μεθόδου είναι πως μπορούμε να υπολογίσουμε το σφάλμα κατά την εκτέλεση της μεθόδου και όχι ξεχωριστά. Κάθε δέντρο δημιουργείται χρησιμοποιώντας ένα διαφορετικό δείγμα από τα δεδομένα, και περίπου το  $\frac{1}{3}$  από αυτά δεν χρησιμοποιείται για την κατασκευή του k δέντρου. Τα δεδομένα που μένουν εκτός χρησιμοποιούνται για να πάρουμε ένα σύνολο δοκιμών κατάταξης (classification test set). Έστω στο τέλος της εκτέλεσης ότι έχουμε j την κλάση που πήρε τις περισσότερες ψήφους κάθε φορά που η περίπτωση ή ήταν εκτός δείγματος. Η μέση τιμή του ποσοστού των φορών που η τάξη j δεν είναι ίση με την πραγματική τάξη ή σε όλες τις δυνατές περιπτώσεις είναι η εκτίμηση σφάλματος OOB. Η τιμή αυτή έχει αποδειχτεί αμερόληπτη σε πολλές δοκιμές [7].

### 5. Εξέταση και ενημέρωση του μοντέλου

Επόμενο βήμα είναι η βελτίωση του μοντέλου ώστε να πετύχουμε μεγαλύτερη ακρίβεια στα αποτελέσματά μας ή καλύτερη ταχύτητα εκτέλεσης ή χρησιμοποίηση λιγότερης μνήμης. Ένας τρόπος να το πετύχουμε αυτό είναι να αλλάξουμε τις παραμέτρους προσαρμογής ώστε να πάρουμε ένα πιο ακριβές μοντέλο. Για παράδειγμα μπορούμε να πάρουμε άνισα κόστη ταξινόμησης ή να αλλάξουμε το βάθος του δέντρου με το οποίο θα παίρνουμε τις μετρήσεις ή τον τρόπο κλαδεμάτος του δέντρου (prunning) ή τον τρόπο με τον οποίο θα διαχωρίζουμε το δείγμα μας.

**6. Χρησιμοποίηση του μοντέλου για προβλέψεις**

Σε αυτό το σημείο μπορούμε να χρησιμοποιήσουμε το μοντέλο που δημιουργήσαμε πάνω σε δείγματα χωρίς τιμές εξόδου και να παράγουμε προβλέψεις και αποτελέσματα που είναι ακριβή και με μικρό σφάλμα. Επιπλέον, είμαστε σε θέση να δοκιμάσουμε διαφορετικούς αλγορίθμους πάνω στο ίδιο δείγμα, να παρατηρήσουμε την έξοδό ώστε να έχουμε καλύτερη συνολική εικόνα και να παράγουμε πιο ακριβή συμπεράσματα.



# Κεφάλαιο 2

## CART

### 2.1 Εισαγωγή

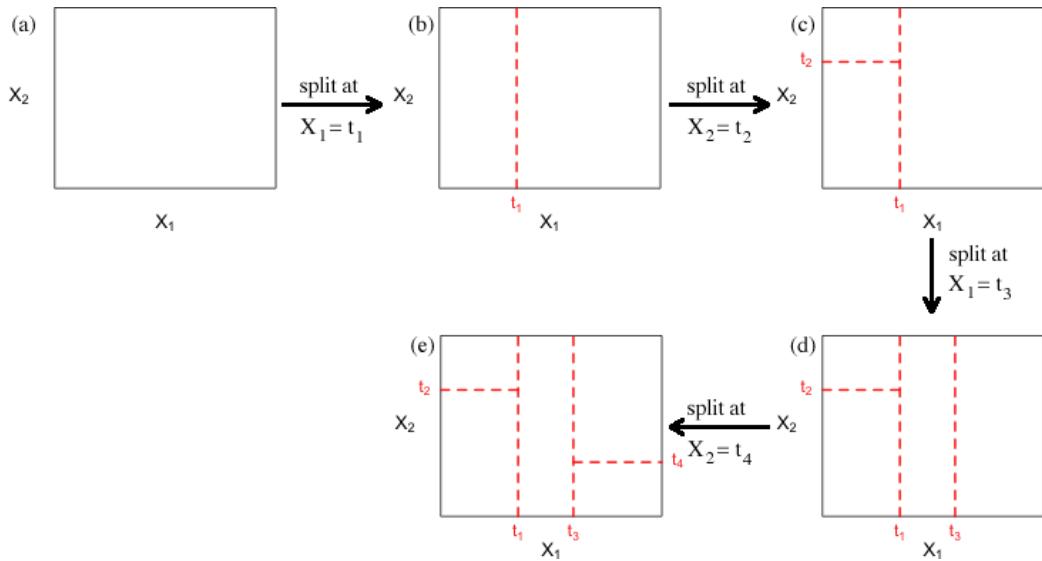
Τα μοντέλα CART (Classification & Regression Trees) είναι από τις πιο δημοφιλείς μεθόδους για tree-based regression και classification. Τα μοντέλα αυτά χρησιμοποιούν αναδρομική διχοτόμηση (binary recursive partitioning) ώστε να διαχωρίσουν τα δεδομένα σε υποσύνολα, έτσι ώστε οι καταγραφές εντός των υποσυνόλων να είναι πιο ομοιογενείς μεταξύ τους συγχρινόμενες με το πώς θα ήταν στο ίδιο υποσύνολο [29].

Σε κάθε βήμα της διαδικασίας επιλέγουμε μία συγκεκριμένη μεταβλητή και ένα σημείο διαχωρισμού και ύστερα διαχωρίζουμε τα δεδομένα μας ή ένα σύνολο αυτών σε δύο μέρη. Αυτό επιτυγχάνεται επιλέγοντας ένα σύνολο προς διαίρεση και εξετάζοντας όλες τις πιθανές μεταβλητές και όλα τα πιθανά σημεία διαχωρισμού αυτών των μεταβλητών. Έστερα επιλέγουμε τον συνδυασμό μεταβλητής - σημείου διαχωρισμού ο οποίος βελτιστοποιεί κάποιο χριτήριο και με βάση τον συνδυασμό αυτό διαχωρίζουμε το σύνολο σε δύο μέρη και επαναλαμβάνουμε την διαδικασία ανδρομικά. Το σύνηθες χριτήριο για ένα δέντρο παλινδρόμησης είναι το υπολειπόμενο άθροισμα τετραγώνων (RSS) ενώ για ένα δέντρο ταξινόμησης είναι ο δείκτης Gini (8).

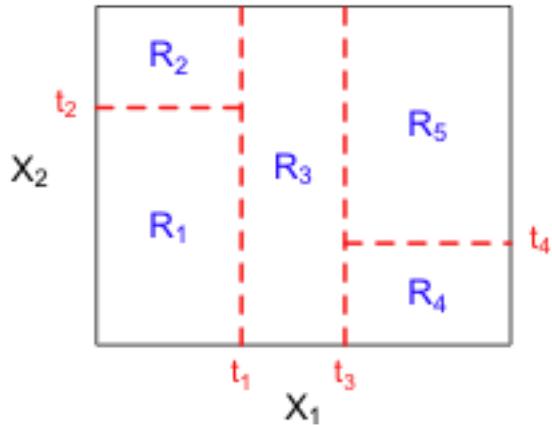
Έστω για παράδειγμα ένα πρόβλημα παλινδρόμησης με συνεχής απόχριση  $Y$  και εισόδους  $X_1$  και  $X_2$ . Αρχικά χωρίζουμε το σύνολο σε δύο υποσύνολα στο σημείο  $X_1 = t_1$ . Έστερα η περιοχή  $X_1 \leq t_1$  χωρίζεται στο  $X_2 = t_2$  και η περιοχή  $X_1 > t_1$  στο σημείο  $X_1 = t_3$ . Τέλος, η περιοχή  $X_1 > t_3$  χωρίζεται στο σημείο  $X_2 = t_4$ . Η διαδικασία δίνεται σχηματικά στο σχήμα 2.1. Το αποτέλεσμα αυτής είναι ο διαχωρισμός του συνόλου σε πέντε περιοχές  $R_1, R_2, \dots, R_5$  όπως φαίνεται στο σχήμα 2.2. Η τιμή της απόχρισης είναι η μέση τιμή κάθε μίας εκ των πέντε περιοχών,  $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_5$ .

Επομένως σε περιοχή  $R_m$  το μοντέλο παλινδρόμησης προβλέπει την έξοδο  $Y$  μέσω σταθεράς  $c_m$  σύμφωνα με την σχέση:

$$\hat{f}(x) = \sum_{m=1}^5 c_m I \{(X_1, X_2) \in R_m\} \quad (2.1)$$

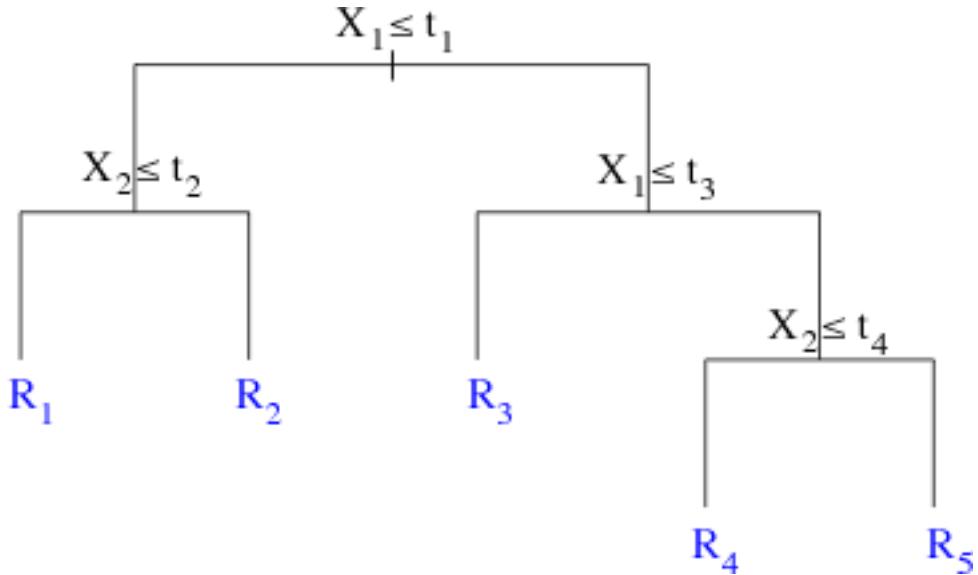


Σχήμα 2.1: Παράδειγμα recursive partitioning με δύο predictors



Σχήμα 2.2: Περιοχές στο επίπεδο  $X_1, X_2$  μετά από recursive partition

Το  $I$  είναι ένας δείκτης που δείχνει αν η παρατήρηση έγκειται σε κάποια δεδομένη ορθογώνια περιοχή. Επειδή μία παρατήρηση μπορεί να ανήκει κάθε φορά σε μόνο μία από τις πέντε περιφέρειες, τέσσερις από τους πέντε όρους του αιθροίσματος θα είναι μηδενικοί. Ως αποτέλεσμα παίρνουμε το  $\bar{y}$  της περιοχής που ανήκει το συγκεκριμένο σημείο.

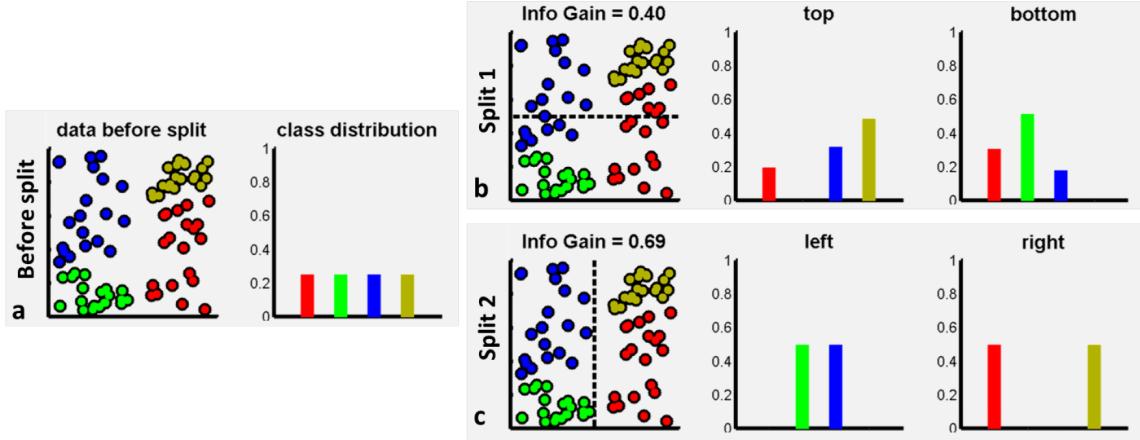


Σχήμα 2.3: Δυαδικό δέντρο μετά από recursive partition

Το ίδιο μοντέλο μπορεί να αναπαρασταθεί ως δέντρο, αφού σε κάθε αναδρομική κλήση μπορούμε να διαχωρίσουμε μόνο περιοχές που έχουν διαχωριστεί από πριν. Οι παρατηρήσεις που ικανοποιούν την συνθήκη βρίσκονται στον αριστερό κλάδο και οι υπόλοιπες στον δεξιό. Η κάθε περιοχή  $R_i$  είναι το φύλλο του δέντρου και μπορεί να βρεθεί ακολουθώντας τους κλάδους με τις αντίστοιχες συνθήκες. Τα σημεία διαχωρισμού ονομάζονται splits. Το βασικό πλεονέκτημα των δυαδικών δέντρων είναι η ευκολία ερμηνίας τους ακόμη και αν έχουμε ως είσοδο περισσότερες μεταβλητές [12]. Μία παρουσίαση του δέντρου του προηγούμενου παραδείγματος φαίνεται στην εικόνα 2.3.

## 2.2 Κέρδος πληροφορίας

Πριν μιλήσουμε για τον αλγόριθμο διαχωρισμού, είναι σημαντικό να εξοικειωθούμε με τις έννοιες της εντροπίας και του κέρδους πληροφορίας (information gain), οι οποίες χρησιμοποιούνται συχνά στην στατιστική και στη θεωρία πληροφορίας. Το σχήμα 2.4 μας δείχνει ένα σύνολο δεδομένων στον 2D χώρο. Διαφορετικά χρώματα υπονοούν διαφορετικές κλάσεις. Στο σχήμα 2.4a η κατανομή των κλάσεων είναι ομοιόμορφη διότι έχουμε ακριβώς τον ίδιο αριθμό σημείων σε κάθε κλάση. Αν διαιρέσουμε τα δεδομένα μας οριζόντια όπως στο σχήμα 2.4b τότε παράγουμε δύο σύνολα δεδομένων. Κάθε υποσύνολο που δημιουργείται έχει χαμηλότερη εντροπία, αφού έχουμε περισσότερες πληροφορίες σε μικρότερα ιστογράμματα.



Σχήμα 2.4: Κέρδος πληροφορίας σε διαχριτή κατανομή. (a) Αρχικό σύνολο δεδομένων  $S$ . (b) Μετά από οριζόντιο διαχωρισμό. (c) Μετά από κατακόρυφο διαχωρισμό.

Το κέρδος πληροφορίας που έχουμε με το να διαιρέσουμε τα δεδομένα είναι [17]:

$$I = H(S) - \sum_{i \in \{1,2\}} \frac{|S^i|}{|S|} H(S^i) \quad (2.2)$$

με την εντροπία Shannon να ορίζεται ως (4):

$$H(S) = - \sum_{c \in C} p(c) \log(p(c)) \quad (2.3)$$

Τα  $S^i$  υποδηλώνουν υποσύνολα του συνόλου δεδομένων, τα οποία έχουν δημιουργηθεί μετά από split και έχουν φτάσει σε κάποιο κόμβο  $i$  ακολουθώντας κάποιες διακλαδώσεις του δέντρου που προκύπτει ύστερα από τα split.

Στο παράδειγμά μας, ένας οριζόντιο split δεν διαχωρίζει τόσο καλά τα δεδομένα μας, και έχει ως αποτέλεσμα ένα κέρδος  $I = 0.4$ . Αν όμως διαιρέσουμε τα δεδομένα κάθετα, όπως στο σχήμα 2.4c, τότε έχουμε πολύ καλύτερα αποτελέσματα με μικρότερη εντροπία και μεγαλύτερο κέρδος πληροφορίας ( $I = 0.69$ ). Αυτό συμβαίνει διότι με τον κάθετο διαχωρισμό έχουμε παράξει δύο υποσύνολα που το καθένα έχει μόνο 2 κλάσεις, ενώ στην προηγούμενη περίπτωση αυτό δεν κατέστη δυνατό. Με αυτό το παράδειγμα, μπορούμε να δούμε πώς μπορούμε να χρησιμοποιήσουμε το information gain ώστε να παράξουμε την διάσπαση που θα μας δώσει τα καλύτερα αποτελέσματα. Η έννοια του κέρδους πληροφορίας αποτελεί και την βάση της μεθόδου random forest την οποία θα αναλύσουμε στο κεφάλαιο 4.

## 2.3 Αλγόριθμος διαχωρισμού

Τα βήματα που ακολουθούμε κατά την κατασκευή του δέντρου είναι τα ίδια σε περίπτωση παλινδρόμησης ή ταξινόμησης με διαφοροποιήσεις σε συγκεκριμένα σημεία που αφορούν τον τρόπο επιλογής των μεταβλητών όπως και το κριτήριο διαχωρισμού. Ο αλγόριθμος έχει ως εξής:

**Βήμα 1: Επιλογή μεταβλητής και σημείου διαχωρισμού**

**Regression:** Σε περίπτωση ποσοτικής μεταβλητής  $m$  με διακριτές τιμές  $\xi_1, \xi_2, \dots, \xi_m$  εξετάζουμε κάθε μεταβλητή ξεχωριστά. Για κάθε μεταβλητή δημιουργούνται δύο σύνολα τιμών:  $\{x \leq \xi_i\}$  και  $\{x > \xi_i\}$ ,  $i = 1, 2, \dots, m$ . Διαιρέσεις συμβαίνουν μόνο μεταξύ διαδοχικών τιμών των δεδομένων. Επομένως εξετάζονται  $m - 1$  χωρίσματα.

**Classification:** Σε περίπτωση ποιοτικής μεταβλητής με  $m$  κατηγορίες πρέπει να εξετάσουμε όλους τους δυνατούς τρόπους ανάθεσης των κατηγοριών σε δύο σύνολα. Τα σημεία διαχωρισμού μπορεί να βρίσκονται οπουδήποτε και μία κατηγορία μπορεί να ανατεθεί σε οποιοδήποτε σύνολο. Υπάρχουν  $2^{m-1} - 1$  τρόποι με τους οποίους μπορεί να συμβεί αυτό.

**Βήμα 2: Υπολογισμός κριτηρίου διαχωρισμού κάθε πιθανής διαιρεσης**

**Regression:** Έστω για συνεχή έξοδο  $y$  ότι διαλέγουμε την μεταβλητή  $X_j$  και το σημείο διαχωρισμού  $s$  οπότε παίρνουμε τα εξής δύο σύνολα:

$$R_1(j, s) = \{x | x_j \leq s\}, R_2(j, s) = \{x | x_j > s\} \quad (2.4)$$

Για συνεχείς μεταβλητές η σύνηθης μέθοδος που χρησιμοποιείται είναι το υπολειπόμενο άθροισμα τετραγώνων (residual sum of squares - RSS). Πριν διαιρέσουμε τα δεδομένα, υπολογίζουμε το συνολικό υπολειπόμενο άθροισμα τετραγώνων χρησιμοποιώντας την μέση τιμή όλων των αποχρίσεων [10].

$$RSS_0 = \sum (y_i - \bar{y})^2 \quad (2.5)$$

Για τις παρατηρήσεις που βρίσκονται στην ίδια ομάδα μετράμε την μέση τιμή της απόχρισης. Τότε το RSS θα είναι:

$$RSS(split) = RSS_1 + RSS_2 = \sum_{R_1} (y_i - \bar{y}_1)^2 + \sum_{R_2} (y_i - \bar{y}_2)^2 \quad (2.6)$$

Η μεταβλητή  $X_j$  και το σημείο διαχωρισμού  $s$  που θα λάβουμε θα ικανοποιούν:

$$\max[RSS_0 - RSS(split)] \quad (2.7)$$

**Classification:** Εδώ έχουμε περισσότερα κριτήρια διαχωρισμού. Έστω  $y$  μία κατηγορηματική μεταβλητή με  $m$  κατηγορίες και έστω:

$$\begin{aligned} n_{ik} &= \text{αριθμός παρατηρήσεων τύπου } k \text{ στον κόμβο } i \\ p_{ik} &= \text{ποσοστό παρατηρήσεων τύπου } k \text{ στον κόμβο } i \end{aligned}$$

Τότε μπορούμε να χρησιμοποιήσουμε κάποιο από τα επόμενα κριτήρια:

1. Deviance:

$$D_i = -2 \sum_k n_{ik} \log p_{ik} \quad (2.8)$$

2. Cross-entropy:

$$D_i = -2 \sum_k p_{ik} \log p_{ik} \quad (2.9)$$

3. Gini index:

$$D_i = 1 - \sum_k p_{ik}^2 \quad (2.10)$$

4. Misclassification error:

$$D_i = 1 - p_{ik(i)} \quad (2.11)$$

όπου  $k(i)$  είναι η κατηγορία στον κόμβο  $i$  με τον μεγαλύτερο αριθμό παρατηρήσεων.

Το σφάλμα για κάποιο διαχωρισμό ενός συνόλου είναι το άνθροισμα των σφαλμάτων πάνω σε όλους τους κόμβους:  $\sum_i D_i$ . Για κάθε ένα από τα παραπάνω κριτήρια το σφάλμα ελαχιστοποιείται όταν όλες οι παρατηρήσεις για κάποιο κόμβο είναι του ίδιου τύπου. Στόχος του δέντρου είναι να διαχωρίσει τα σύνολα με τέτοιο τρόπο ώστε τα τελικά σύνολα να είναι πιο αμιγή σε σχέση με πριν τον διαχωρισμό [13].

Οι μέθοδοι cross-entropy, deviance και gini index λόγω του ότι είναι διαφορίσιμες είναι πιο προσιτές για αριθμητική βελτιστοποίηση. Επιπρόσθετα, είναι πιο ευαίσθητες σε μεταβολές πιθανοτήτων των κόμβων σε σχέση με το missclassification rate.

### Βήμα 3: Επιπλέον διαχωρισμός των συνόλων

Έχοντας διαχωρίσει βέλτιστα το σύνολό μας στο προηγούμενο στάδιο, επόμενο βήμα είναι να διαχωρίσουμε καθένα από τα σύνολα που δημιουργήθηκαν αναδρομικά, λαμβάνοντας υπόψη όλους τους δυνατούς κόμβους προς διαίρεση. Είμαστε υποχρεωμένοι να διαιρέσουμε ανάμεσα στα σύνολα που δημιουργήθηκαν από το προηγούμενο στάδιο ώστε τελικά να παράγουμε το δέντρο μας.

### Βήμα 4: Περάτωση της διαδικασίας

Μέσω του επόμενου κριτηρίου, σε κάποιο σημείο σταματάμε τις επιπλέον διαιρέσεις και δίνουμε τέλος στον αλγόριθμο παίρνοντας τα τελικά μας αποτελέσματα.

## 2.4 Cost Complexity Pruning

Ο αλγόριθμος που περιγράψαμε είναι άπληστος και βρίσκει το βέλτιστο σημείο διαχωρισμού για το αμέσως επόμενο βήμα χωρίς όμως να λαμβάνει υπόψη δέντρα επόμενων βημάτων και την επίδοσή τους. Για παράδειγμα, όταν μπορούσαμε να είχαμε ένα σημείο διαχωρισμού που να μην δίνει πολύ καλό σφάλμα μετά την διαίρεση, αλλά σε κάποιο επόμενο βήμα να δίνει ένα άλλο σημείο διαχωρισμού με σημαντικά μειωμένο σφάλμα και πολύ καλύτερη επίδοση. Επομένως μία στρατηγική του να κάνουμε διαίρεση μόνο αν η μείωση του συνολικού σφάλματος ξεπερνά κάποιο κατώφλι ε δεν είναι καλή, αφού μπορεί σε κάποιο επόμενο βήμα να έχουμε ένα πολύ καλό split που να επιφέρει σημαντικές βελτιώσεις ακρίβειας.

Η στρατηγική που προτιμάται είναι να μεγαλώσουμε ένα πολύ μεγάλο δέντρο  $T_0$  σταματώντας την διαδικασία διαίρεσης μόνο όταν ένας ελάχιστος αριθμός κόμβων επιτευχθεί και ύστερα να κλαδέψουμε το δέντρο σύμφωνα με το κριτήριο cost complexity.

Έστω υποδέντρο  $T \subset T_0$  το οποίο παράγεται αν κλαδέψουμε το  $T_0$ . Έστω τα φύλλα του δέντρου  $n$ , όπου κάθε κόμβος  $n$  ανήκει στην περιοχή  $R_m$ . Έστω  $|T|$  ο αριθμός των φύλλων του δέντρου  $T$ .

Έστω ότι μεγαλώνουμε ένα πολύ μεγάλο δέντρο με  $n$  φύλλα. Σύμφωνα με τον αλγόριθμο που περιγράφτηκε νωρίτερα, σε κάθε στάδιο βρίσκουμε το βέλτιστο δέντρο, οπότε αν κλαδέψουμε το δέντρο μπορούμε να δημιουργήσουμε βέλτιστο δέντρο μικρότερου μεγέθους. Έστω  $T$  το τρέχον δέντρο και  $|T|$  το πλήθος των σημείων διαχωρισμού του δέντρου. Το κριτήριο cost complexity ορίζεται ως εξής:

$$CC(T) = \sum_{\text{terminal nodes}} D_i + \lambda |T| \quad (2.12)$$

Το  $D_i$  είναι το κριτήριο RSS για regression ή ένα από τα κριτήρια που αναφέρθηκαν για classification σε κάποιο κόμβο  $i$ . Η παράμετρος  $\lambda \geq 0$  είναι μία παράμετρος συντονισμού (tuning parameter or penalty term). Όταν αλλάζουμε αυτή την παράμετρο διαλέγουμε διαφορετικού μεγέθους δέντρα από την ακολουθία μας από τα βέλτιστα δέντρα. Μεγάλες τιμές της παραμέτρου έχουν ως αποτέλεσμα μικρότερα δέντρα, και αντίστροφα όσο μικραίνει το  $\lambda$ . Για  $\lambda = 0$  έχουμε το αρχικό μας δέντρο.

Το  $\lambda$  παίζει έναν παρόμοιο ρόλο όπως η μεταβλητή  $K$  στο Akaike information criterion (AIC), το οποίο μετρά την σχετική ποιότητα ενός στατιστικού μοντέλου για κάποιο σύνολο δεδομένων και παρέχει έναν τρόπο επιλογής του τελικού μοντέλου ανάλογα με το fitting και την πολυπλοκότητα που θέλουμε να πετύχουμε:

$$AIC = -2 \log L + 2K \quad (2.13)$$

Για κάθε τιμή της παραμέτρου  $\lambda$  μπορούμε να δείξουμε πως υπάρχει μοναδικό ελάχιστο υποδέντρο  $T_\lambda$  τέτοιο ώστε να ελαχιστοποιεί την 2.12. Για να βρούμε το  $T_\lambda$  χρησιμοποιούμε το λεγόμενο weakest link pruning: κλαδεύουμε διαδοχικά τον κόμβο που παράγει την μικρότερη αύξηση στο  $\sum_n D_i$  μέχρι να φτάσουμε στην ρίζα. Αυτή η διαδικασία μας δίνει μία αλληλουχία από υποδέντρα τα οποία περιέχουν το  $T_\lambda$  [27].

## 2.5 Cross Validation

Εκτίμηση της παραμέτρου  $\lambda$  μπορεί να επιτευχθεί με k-fold cross validation (συνήθως five ή tenfold). Με αυτό τον τρόπο μπορούμε να καθορίσουμε πότε να σταματήσουμε να κλαδεύουμε το δέντρο.

Για κάθε βέλτιστο δέντρο δεδομένου μεγέθους  $k$  για κάθε διακριτή τιμή  $CC(T)$  κάνουμε k-fold cross validation. Για να το πετύχουμε αυτό, διαιρούμε τυχαία τα δεδομένα μας  $D$  σε  $k$ -subsets ώστε να ισχύει:

$$D = D^{(1)} \cup D^{(2)} \cup \dots \cup D^{(k)}$$

Κάθε φορά αφήνουμε έξω ένα από τα  $k$  υποσύνολα  $D_i$  και εκτελούμε τον αλγόριθμο για τα  $k - 1$  υποσύνολα παίρνοντας προβλέψεις  $\bar{y}_i$  ή  $\hat{r}_{ij}$  ανάλογα με το αν έχουμε regression ή classification tree. Τα δεδομένα που μένουν έξω χρησιμοποιούνται για να υπολογίσουν το σφάλμα. Αυτή η διαδικασία γίνεται  $k$ -φορές για κάθε υποδέντρο και το σχετικό μέσο σφάλμα  $xerror$  (relative average cross-validation error) είναι:

$$xerror = \frac{\frac{1}{k} \sum_k RSS(k - foldtree)}{RSS(nulltree)} \quad (2.14)$$

Η διαδικασία μπορεί να σταματήσει με κάποιον από τους ακόλουθους δύο κανόνες:

**Κανόνας 1:** Διάλεξε την τιμή  $CC(T)$  που παράγει το δέντρο το οποίο ελαχιστοποιεί την 2.14.

**Κανόνας 2:** Έστω  $xstd$  η τυπική απόκλιση του δέντρου με το ελάχιστο  $xerror$ . Τότε διάλεξε την πρώτη (μεγαλύτερη) τιμή  $CC(T)$  τέτοια ώστε:

$$xerror < \min(xerror) + xstd \quad (2.15)$$

## 2.6 Παρατηρήσεις

Τα δέντρα δεν είναι χρήσιμα για μικρά σύνολα δεδομένων, διότι εκεί για να επιτύχουμε οποιαδήποτε πρόοδο πρέπει να κάνουμε ισχυρές παραδοχές ενώ τα δέντρα δεν κάνουν ποτέ υποθέσεις σχετικές με τα δεδομένα μας. Από την άλλη πλευρά είναι πολύ χρήσιμα σε μεγάλα σύνολα δεδομένων με πολλές μεταβλητές προς επιλογή αφού μπορούν να βρουν πολύπλοκες δομές οι οποίες δεν μπορούν να ανιχνευθούν με τα συνήθη μοντέλα παλινδρόμησης. Ως αποτέλεσμα, μπορούν να μας δώσουν πολύ καλές προβλέψεις. Είναι σημαντικό να σημειώσουμε πως λόγω του τρόπου δημιουργίας του, το τελικό μας δέντρο δεν είναι αναγκαία το βέλτιστο ακόμη και αν όλα τα υποδέντρα του έχουν δημιουργεί με το βέλτιστο τρόπο και με εφαρμογή μεθόδων pruning όπως η μέθοδος cost complexity που περιγράφηκε πριν.

Σε σχέση με τα κλασσικά παραμετρικά μοντέλα, τα δέντρα μπορούν να χρησιμοποιηθούν για να προτείνουν πιθανές αλληλεπιδράσεις μεταξύ μεταβλητών. Για παραδειγμα, όταν δύο διαφορετικές μεταβλητές χρησιμοποιούνται για να διαιρέσουν το δέντρο σε διαφορετικά σημεία στον ίδιο κλάδο, αυτό θα μπορούσε να ερμηνευτεί ως

μία αλληλεπίδραση μεταξύ αυτών των δύο μεταβλητών. Επιπρόσθετα, αν μία μεταβλητή εμφανίζεται πολλές φορές κατά μήκος του ίδιου κλάδου, αυτό υποδηλώνει πως μπορεί να υπάρχει μία μη γραμμική σχέση μεταξύ αυτής της μεταβλητής και της εξόδου του συστήματος. Τα δέντρα όμως μπορούσαμε ακόμη να τα χρησιμοποιήσουμε ώστε να δούμε αν υπάρχει κάποια πρόσθετη δομή που ίσως να έχει αγνοηθεί από τα κλασσικά παραμετρικά μοντέλα [19].

Τα δέντρα έχουν γνωρίσει αρκετές βελτιώσεις. Πρόσφατη προσοχή έχει δοθεί στις λεγόμενες ensemble methods, οι οποίες περιλαμβάνουν μεθόδους όπως boosting, bagging και random forests. Η ιδέα πίσω από αυτές τις μεθόδους είναι να δημιουργήσουμε δέντρα σε μικρότερα δείγματα δεδομένων, και να πάρουμε τον μέσο όρο των προβλέψεων αυτών έπειτα από διαδοχικές επαναλήψεις [20].



# Κεφάλαιο 3

## MARS

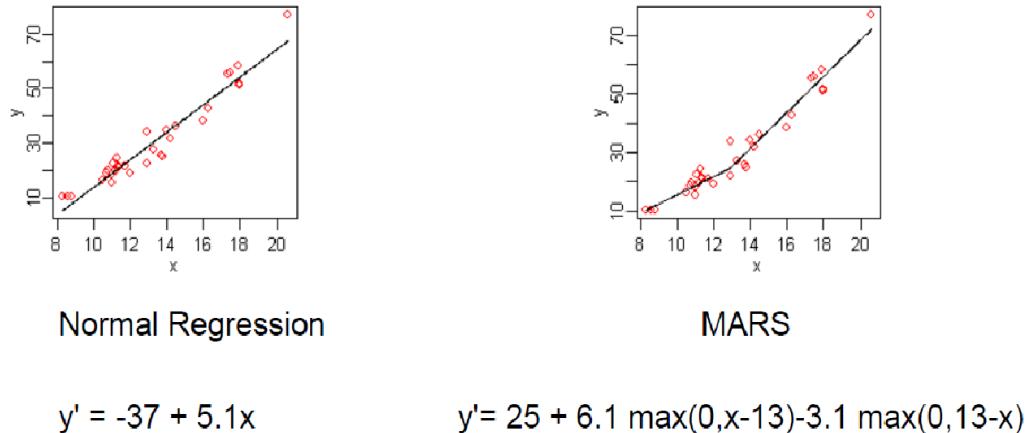
### 3.1 Εισαγωγή

Το MARS (Multivariate Adaptive Regression Splines) είναι μία προσαρμοστική τεχνική παλινδρόμησης και είναι κατάλληλη για υψηλών διαστάσεων προβλήματα τα οποία περιέχουν μεγάλο αριθμό εισόδων. Το MARS μπορεί να θεωρηθεί ως μία γενίκευση της τεχνικής της γραμμικής προοδευτικής παλινδρόμησης για μη γραμμική μοντελοποίηση ή ως μία τροποποίηση της μεθόδου CART ώστε να βελτιώσει τις επιδόσεις της τελευταίας στη ρύθμιση της παλινδρόμησης (7).

Το MARS δημιουργήθηκε από τον Jerome Friedman το 1991. Ο όρος 'MARS' είναι εμπορικό σήμα της εταιρίας Salford Systems ενώ για open source implementations φέρει το όνομα 'earth'. Θεωρείται απλούστερο σε σχέση με άλλα μοντέλα όπως random forests ή neural networks. Αρχικά θα εισάγουμε την τεχνική MARS και ύστερα θα κάνουμε την σύνδεση με άλλα μοντέλα όπως το CART.

### 3.2 MARS και Normal Regression

Στο σχήμα 3.1 βλέπουμε μία σύγκριση συναρτήσεων που έχουν παραχθεί μετά από εφαρμογή ενός απλού μοντέλου MARS και ενός μοντέλου γραμμικής παλινδρόμησης πάνω στα ίδια δεδομένα. Αρχικά παρατηρούμε πως στο μοντέλο της παλινδρόμησης έχουμε μία γραμμική συνάρτηση η οποία έχει χρησιμοποιηθεί πάνω στο σύνολο των δεδομένων και αφορά όλο το εύρος τους. Αντίθετα, στο μοντέλο MARS έχουμε χωρίσει τα δεδομένα μας σε περιοχές και έχουμε βρει μία γραμμική συνάρτηση για κάθε περιοχή, η οποία περιγράφει καλύτερα τα δεδομένα μας πάνω στην συγκεκριμένη περιοχή σε σχέση με το μοντέλο της παλινδρόμησης. Με αυτό τον τρόπο πετυχαίνουμε καλύτερα και πιο ακριβή αποτελέσματα.



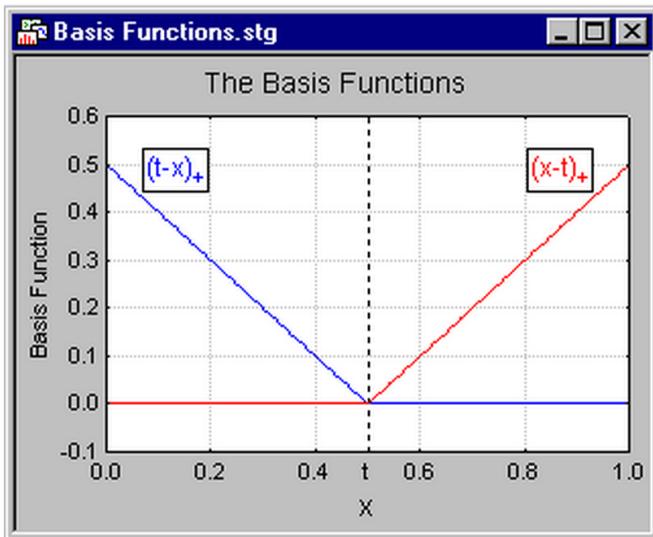
Σχήμα 3.1: MARS και Linear Regression

Τα χαρακτηριστικά ενός γραμμικού παραμετρικού μοντέλου (Global Parametric Model) όπως για παράδειγμα η γραμμική παλινδρόμηση είναι τα παρακάτω (12):

- Υψηλή ταχύτητα υπολογισμών αλλά με περιορισμένη ευελιξία
- Ακριβής μόνο αν το συγκεκριμένο μοντέλο έχει την σωστή λειτουργική δομή (για παράδειγμα παρουσιάζει γραμμικότητα)
- Λειτουργεί καλά σε μικρά σύνολα δεδομένων
- Όλα τα δεδομένα επιδρούν συνολικά στην κατασκευή του τελικού μοντέλου

Ένα μη παραμετρικό μοντέλο σε αντίθεση με πριν δημιουργεί την τελική συνάρτηση τοπικά και όχι συνολικά. Τα χαρακτηριστικά του είναι τα εξής:

- Προσδιορίζει μία μικρή περιοχή των δεδομένων
- Συνοψίζει πώς οι μεταβλητές συμπεριφέρονται στην συγκεκριμένη περιοχή
- Αναπτύσσει ένα ξεχωριστό μοντέλο σε κάθε περιοχή
- Έχει την δυνατότητα να επιβάλλει περιορισμούς συνέχειας



Σχήμα 3.2: Οι συναρτήσεις βάσης  $(x - t)_+$  και  $(t - x)_+$  του MARS

### 3.3 Περιγραφή της μεθόδου

Το MARS είναι μία μη παραμετρική διαδικασία παλινδρόμησης η οποία δεν κάνει καμία υπόθεση σχετικά με την υποκείμενη σχέση μεταξύ των εξαρτημένων και των ανεξάρτητων μεταβλητών του συστήματος. Αντίθετα οι συντελεστές και οι συναρτήσεις παράγονται μέσα από τα δεδομένα μας. Υπό μία έννοια, η μέθοδος βασίζεται στην τεχνική 'διαιρεί και βασίλευε' (Divide and Conquer) η οποία διαχωρίζει τον χώρο εισόδου σε περιοχές, κάθε μία με την δική της εξίσωση παλινδρόμησης. Το γεγονός αυτό καθιστά το MARS κατάλληλο για προβλήματα με υψηλότερες διαστάσεις εισόδου (δηλαδή με περισσότερες από 2 μεταβλητές) όπου άλλες κλασσικές τεχνικές δεν ψαρουσιάζουν καλή συμπεριφορά (9).

Το MARS χρησιμοποιεί τμηματικά γραμμικές συναρτήσεις της μορφής  $(x - t)_+$  και  $(t - x)_+$  οι οποίες ονομάζονται 'συναρτήσεις βάσης'. Το '+' σημαίνει ότι λαμβάνουμε υπ' όψη μας μόνο το θετικό μέρος των συναρτήσεων, αλλιώς θεωρούμε την συνάρτηση μηδενική. Έχουμε επομένως [22]:

$$(x - t)_+ = \begin{cases} x - t, & \text{αν } x > t, \\ 0, & \text{διαφορετικά} \end{cases}, \quad (t - x)_+ = \begin{cases} t - x, & \text{αν } x < t, \\ 0, & \text{διαφορετικά} \end{cases}$$

Στο σχήμα 3.2 έχουμε ένα παράδειγμα των συναρτήσεων  $(x - 0.5)_+$  και  $(0.5 - x)_+$ .

Οι συναρτήσεις είναι τυμηματικά γραφμικές, με κόμπο (knot) στο σημείο  $t$  και ονομάζονται linear splines. Οι δύο συναρτήσεις  $(x - t)_+$  και  $(t - x)_+$  αποτελούν ένα reflected pair. Η ιδέα είναι να δημιουργήσουμε reflected pairs για κάθε είσοδο  $X_j$  με knots σε κάθε παρατηρούμενη τιμή  $x_{ij}$  της συγκεκριμένης εισόδου. Επομένως αν πάρουμε συνολικά τις συναρτήσεις βάσεις για κάθε είσοδο θα έχουμε:

$$C = \{(X_j - t)_+, (t - X_j)_+\}_{\substack{t \in \{x_{1j}, x_{2j}, \dots, x_{Nj}\} \\ j = 1, 2, \dots, p}} \quad (3.1)$$

Αν κάθε τιμή εισόδου είναι διαφορετική τότε θα έχουμε συνολικά  $2Np$  συναρτήσεις βάσης. Παρότι κάθε συνάρτηση βάσης εξαρτάται μόνο από ένα συγκεκριμένο  $X_j$ , για παράδειγμα  $h(X) = (X_j - t)_+$ , θεωρείται συνάρτηση ολόκληρου του χώρου εισόδου  $\mathbb{R}^p$ .

Οι συναρτήσεις βάσης του συνόλου  $C$  μαζί με το γινόμενο τους συνδυάζονται για να παράγουν προβλέψεις δεδομένων των εισόδων. Η γενική συνάρτηση του μοντέλου MARS είναι [21]:

$$f(X) = \beta_0 + \sum_{m=1}^M \beta_m h_m(X) \quad (3.2)$$

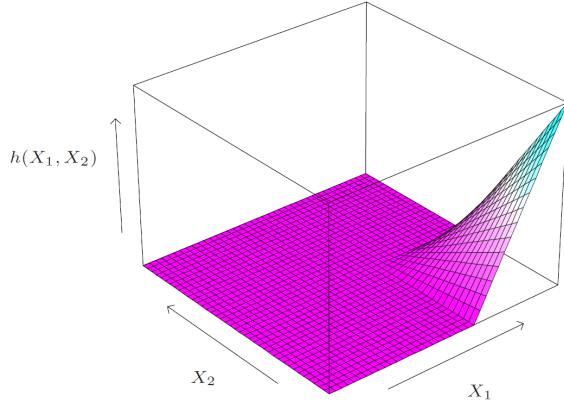
όπου κάθε  $h_m(X)$  είναι μία συνάρτηση στο  $C$ , ή το γινόμενο δύο ή περισσότερων τέτοιων συναρτήσεων.

Δεδομένων των  $h_m$ , οι συντελεστές  $\beta_m$  υπολογίζονται με την ελαχιστοποίηση του υπολοιπού αιθρίσματος τετραγώνων όπως στην κλασική γραφμική παλινδρόμηση. Για την κατασκευή των συναρτήσεων  $h_m(X)$  ξεκινούμε από την σταθερή συνάρτηση  $h_0(X) = 1$  και με όλες τις συναρτήσεις στο  $C$  ως υποψήφιες. Σε κάθε στάδιο θεωρούμε ως ένα καινούριο ζευγάρι συναρτήσεων το γινόμενο μιας συνάρτησης  $h_m$  στο σύνολο  $M$  με ένα από τα reflected pairs στο  $C$ . Προσθέτουμε στο μοντέλο  $M$  τον όρο της μορφής:

$$\hat{\beta}_{M+1} h_l(X) \cdot (X_j - t)_+ + \hat{\beta}_{M+2} h_l(X) \cdot (t - X_j)_+, h_l \in M$$

ο οποίος πετυχαίνει την μεγαλύτερη μείωση σφάλματος. Οι όροι  $\hat{\beta}_{M+1}$  και  $\hat{\beta}_{M+2}$  υπολογίζονται από τα ελάχιστα τετράγωνα μαζί με τους υπόλοιπους  $M+1$  όρους του μοντέλου. Υστερα τα καλύτερα γινόμενα μπαίνουν στο μοντέλο  $M$  και η διαδικασία συνεχίζεται μέχρι το μοντέλο  $M$  να έχει κάποιο μέγιστο αριθμό όρων.

Για παράδειγμα, στο πρώτο στάδιο εισάγουμε στο μοντέλο μία συνάρτηση της μορφής  $\beta_1(X_j - t)_+ + \beta_2(t - X_j)_+; t \in \{x_{ij}\}$  εφόσον το γινόμενο με την σταθερή συνάρτηση  $h_0(X) = 1$  μας δίνει την ίδια συνάρτηση. Έστω η βέλτιστη επιλογή σύμφωνα με τα προηγούμενα πως είναι η  $\beta_1(X_2 - x_{72})_+ + \beta_2(x_{72} - X_2)_+$ .



Σχήμα 3.3: Παράδειγμα συνάρτησης ως γινόμενο συναρτήσεων βάσης

Αυτό το ζευγάρι συναρτήσεων μπαίνει στο σύνολο  $M$  και στο επόμενο βήμα θεωρούμε ένα ζευγάρι της μορφής:

$$h_m(X) \cdot (X_j - t)_+ \quad \text{και} \quad h_m(X) \cdot (t - X_j)_+, \quad t \in \{x_{ij}\}$$

όπου για το  $h_m$  έχουμε τις εξής επιλογές:

$$\begin{aligned} h_0(X) &= 1, \\ h_1(X) &= (X_2 - x_{72})_+, \\ h_2(X) &= (x_{72} - X_2)_+ \end{aligned}$$

Η τρίτη επιλογή δημιουργεί συναρτήσεις όπως  $\eta(X_1 - x_{51})_+ \cdot (x_{72} - X_2)_+$  που φαίνεται στο σχήμα 3.3.

## 3.4 Generalized Cross Validation

Στο τέλος της διαδικασίας έχουμε ένα μεγάλο μοντέλο της μορφής (3.2). Το μοντέλο αυτό στις περισσότερες περιπτώσεις κάνει overfit τα δεδομένα, οπότε ακολουθείται μία αντίστροφη διαδικασία διαγραφής (παρόμοια με την διαδικασία κλαδεμάτων που είδαμε στα classification trees) κατά την οποία σε κάθε βήμα ο όρος ο οποίος δημιουργεί την μικρότερη αύξηση του υπολειπόμενου τετραγωνικού σφάλματος διαγράφεται από το μοντέλο. Ως αποτέλεσμα παράγεται ένα εκτιμώμενο βέλτιστο μοντέλο  $\hat{f}_\lambda$  για μέγεθος (πλήθος όρων)  $\lambda$ . Για την εύρεση της βέλτιστης τιμής  $\lambda$  θα μπορούσαμε να χρησιμοποιήσουμε cross-validation, αλλά για λόγους υπολογιστικού κόστους το MARS χρησιμοποιεί το χριτήριο generalized cross-validation.

To generalized cross-validation ορίζεται ως:

$$GCV(\lambda) = \frac{\sum_{i=1}^N (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2} \quad (3.3)$$

Η τιμή  $M(\lambda)$  είναι ο αριθμός των παραμέτρων του μοντέλου σε ισχύ: αυτό αντιπροσωπεύει τόσο τον αριθμό των όρων του μοντέλου όπως και τον αριθμό των παραμέτρων που χρησιμοποιούνται για την επιλογή των βέλτιστων θέσεων των knots. Κάποιες μαθηματικές προσομοιώσεις υποδεικνύουν πως χρειάζονται περίπου τρεις παράμετροι για να επιλέξουν ένα knot σε μία τμηματικά γραμμική παλινδρόμηση. Επομένως αν υπάρχουν  $r$  ανεξάρτητες γραμμικές συναρτήσεις βάσης στο μοντέλο, και  $K$  knots έχουν επιλεγεί κατά την εκτέλεση της διαδικασίας, θα έχουμε:  $M(\lambda) = r + cK$  όπου  $c = 3$ . Σύμφωνα με αυτό, επιλέγουμε το μοντέλο που κατά την αντίστροφη διαδικασία ελαχιστοποιεί το  $GCV(\lambda)$ .

### 3.5 Ο αλγόριθμος MARS συνοπτικά

Όπως αναφέρθηκε και πριν το MARS είναι μία διαδικασία δύο σταδίων η οποία εφαρμόζεται διαδοχικά μέχρι να παραχθεί ένα επαρκές μοντέλο. Στο πρώτο στάδιο δημιουργούμε το μοντέλο προσθέτοντας συναρτήσεις βάσης και αυξάνοντας την πολυπλοκότητά του μέχρι να φτάσουμε σε ένα συγκεκριμένο σημείο πολυπλοκότητας. Στο δεύτερο στάδιο, ξεκινούμε 'προς τα πίσω' και αφαιρούμε τις λιγότερο σημαντικές συναρτήσεις σύμφωνα με το κριτήριο Generalized Cross Validation. Ο αλγόριθμος έχει ως εξής:

1. Ξεκίνα με το απλούστερο μοντέλο που περιέχει μόνο την σταθερή συνάρτηση βάσης
2. Αναζήτησε στον χώρο των συναρτήσεων βάσης και για κάθε μεταβλητή και για όλα τα δυνατά knots εισήγαγε στο μοντέλο τις συναρτήσεις που μεγιστοποιούν κάποιο κριτήριο ταιριάσματος (goodness of fit) όπως για παράδειγμα ελαχιστοπόίηση σφάλματος πρόβλεψης
3. Εφάρμοσε αναδρομικά το βήμα 2 έως ότου το μοντέλο αποκτήσει μία προκαθορισμένη μέγιστη πολυπλοκότητα
4. Κλάδεψε το δέντρο διαγράφοντας τις βασικές συναρτήσεις που συμβάλλουν το λιγότερο (least squares) στο fitting του μοντέλου ώστε να παραχθεί το τελικό μοντέλο.

## 3.6 Παρατηρήσεις

Το MARS έχει γίνει πολύ δημοφιλές για την δημιουργία μοντέλων πρόβλεψης για 'δύσκολα' προβλήματα εξόρυξης δεδομένων, στα οποία οι predictors δεν παρουσιάζουν απλές ή μονότονες σχέσεις με τις εξαρτημένες μεταβλητές. Εναλλακτικά μοντέλα αποτελούν τα CHAID, CART, όπως και τα νευρωνικά δίκτυα (Neural Networks) [25]. Λόγω του συγκεκριμένου τρόπου με τον οποίο το MARS διαλέγει predictors (συναρτήσεις βάσης), γενικά συμπεριφέρεται καλά σε περιπτώσεις όπου μοντέλα όπως το CART συμπεριφέρονται καλά, δηλαδή όπου ιεραρχικά διαμημένες διαδοχικές διασπάσεις στις μεταβλητές δίνουν ακριβείς προβλέψεις. Στην πραγματικότητα, αντί να θεωρούμε το MARS ως μία γενίκευση της πολλαπλής παλινδρόμησης, μπορούμε να αναλογιστούμε το MARS ως μία γενίκευση των regression trees όπου τα 'σκληρά' binary splits μπορούν να αντικατασταθούν από 'ομαλές' συναρτήσεις βάσης [26].

Μία σημαντική ιδιότητα των συναρτήσεων βάσης είναι η ικανότητά τους να λειτουργούν σε τοπικό επίπεδο. Όταν οι συναρτήσεις βάσεις πολλαπλασιάζονται μεταξύ τους όπως στο σχήμα 3.3 τότε το αποτέλεσμα είναι μη μηδενικό μόνο σε ένα μικρό μέρος του επιπέδου στο οποίο και οι δύο συναρτήσεις είναι μη μηδενικές. Ως αποτέλεσμα η επιφάνεια στην οποία γίνεται η παλινδρομηση χτίζεται τοπικά μέσω μη μηδενικών στοιχείων όποτε αυτά χρειάζονται. Η χρήση άλλων συναρτήσεων βάσης όπως των πολυωνυμικών παράγει μη μηδενικό γινόμενο και αλλού στον χώρο, οπότε δεν θα λειτουργούσε καλά.

Ένα δεύτερο σημαντικό πλεονέκτημα των συναρτήσεων βάσης αφορά το υπολογιστικό κόστος. Έστω το γινόμενο μίας συνάρτησης στο  $M$  με καθένα από τα  $N$  reflected pairs για είσοδο  $X_j$ . Αυτό φαίνεται να απαιτεί το ταίριασμα  $N$  γραμμικών μοντέλων παλινδρόμησης μονής εισόδου καθένα από τα οποία χρησιμοποιεί  $O(N)$  υπολογισμούς με αποτέλεσμα να έχουμε πολυπλοκότητα  $O(N^2)$ . Σε αυτό το σημείο μπορούμε να εκμεταλλευτούμε την μορφή των συναρτήσεων βάσης. Αρχικά ταιριάζουμε το ζεύγος με το δεξιότερο knot. Καθώς το knot πηγαίνει διαδοχικά μία θέση αριστερά, οι συναρτήσεις βάσεις θα διαφέρουν κατά μηδέν στο αριστερό τμήμα του επιπέδου και κατά μία σταυθερά στο δεξί τμήμα. Επομένως σε κάθε βήμα μπορούμε να ενημερώσουμε το μοντέλο σε  $O(1)$  οπότε μπορούμε να δοκιμάσουμε κάθε πιθανό knot με  $O(N)$  υπολογισμούς.

Το μοντέλο MARS κατασκευάζεται ιεραρχικά, αφού οι νέες συναρτήσεις δημιουργούνται από γινόμενα με άλλες συναρτήσεις οι οποίες έχουν δημιουργηθεί από γινόμενα προηγούμενων βημάτων και βρίσκονται ήδη στο μοντέλο. Για παράδειγμα ένα γινόμενο με 4 συναρτήσεις μπορεί να γίνει μόνο αν το γινόμενο των τριών εκ των συναρτήσεων βρίσκεται ήδη στο μοντέλο. Επομένως μία αλληλεπίδραση υψηλής τάξης θα υπάρχει μόνο αν τα μικρότερα μέρη αυτής υπάρχουν ήδη στο μοντέλο. Με αυτό τον τρόπο αποφεύγουμε να αναζητούμε εκθετικές λύσεις στο μοντέλο μας. Ακόμη, έχουμε ένα περιορισμό κατά τον σχηματισμό του μοντέλου. Κάθε είσοδος μπορεί να εμφανιστεί μέχρι μία φορά στο γινόμενο. Αυτό εμποδίζει τον σχηματισμό μεγαλύτερης τάξης δυνάμεων εισόδου, ενώ με τις συναρτήσεις βάσης που είναι τμηματικά γραμμικές μπορούμε να δημιουργήσουμε καλή προσέγγιση με πιο σταυθερό τρόπο [8].

Παρά το γεγονός πως το MARS αρχικά είχε δημιουργηθεί για επίλυση προβλημάτων regression, μπορεί εύκολα να χρησιμοποιηθεί και για προβλήματα classification λόγω της δυνατότητας του μοντέλου να χειρίζεται πολλαπλές μεταβλητές εξόδου. Αρχικά κωδικοποιούμε τις κλάσεις εξόδου σε πολλαπλούς δείκτες (multiple indicator variables). Για παράδειγμα, θα μπορούσαμε να κωδικοποιήσουμε ως '1' την παρατήρηση η οποία ανήκει στην κλάση  $k$  και ως '0' την παρατήρηση που δεν ανήκει στην κλάση  $k$ . Έστερα εφαρμόζουμε τον αλγόριθμο MARS πάνω στο μοντέλο ο οποίος καθορίζει ένα κοινό σύνολο συναρτήσεων βάσης για τις εισόδους μας αλλά με διαφορετικούς συντελεστές για κάθε μεταβλητή και παίρνουμε ως αποτέλεσμα κάποιες προβλέψεις με συνεχείς τιμές ή scores. Τέλος, αναθέτουμε στη κλάση που πέτυχε το μεγαλύτερο score την πρόβλεψή μας. Σημειώνουμε εδώ πως αυτή η εφαρμογή θα δώσει classifications με ευριστικό τρόπο και θα λειτουργήσει καλά στην πράξη, όμως δεν βασίζεται σε κάποιο στατιστικό μοντέλο για εξαγωγή πιθανοτήτων που με βάση αυτές θα εφαρμόζαμε το classification.

## Κεφάλαιο 4

# Random Forests

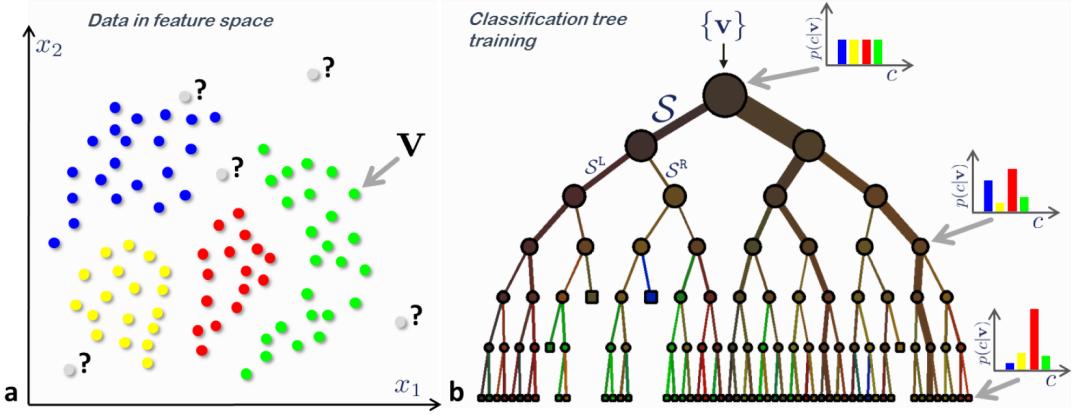
### 4.1 Εισαγωγή

Τα random forests αποτελούν μία μέθοδο μάθησης που μπορεί να χρησιμοποιηθεί τόσο για προβλήματα τάξινόμησης όσο και για προβλήματα παλινδρόμησης. Λειτουργούν με το να κατασκευάζουν ένα πλήθος από δέντρα απόφασης σε δείγμα του συνόλου των δεδομένων κατά τη φάση εκπαίδευσης του μοντέλου και ύστερα να συνυπολογίζουν όλα τα δέντρα για να καθορίσουν την τελική έξοδο. Αποτελούν μία τροποποίηση της μεθόδου bagging, η οποία παίρνει πολλά αμερόληπτα μοντέλα με θόρυβο και βρίσκει την μέση τιμή αυτών, μειώνοντας την διακύμανση της εξόδου.

Τα δέντρα απόφασης έχουν τις ιδιότητες που γνωρίσαμε στο κεφάλαιο 2 και είναι ιδανικοί υποψήφιοι για μεθόδους bagging διότι μπορούν να συλλάβουν πολύπλοκες αλληλεπιδράσεις μεταξύ των δεδομένων, ενώ αν μεγαλώσουν αρκετά βαθιά, έχουν σχετικά χαμηλή μεροληψία. Επιπρόσθετα, λόγω του θορύβου που έχουν, ο μέσος όρος τους αποτελεί έναν καλό δείκτη της πραγματικής εξόδου. Κάθε δέντρο που παράγεται λέμε πως είναι identically distributed, δηλαδή είναι ανεξάρτητο από τα άλλα δέντρα και παρέχει την ίδια κατανομή πιθανότητας ως προς την τελική έξοδο. Με αυτό τον τρόπο, ο μέσος όρος όλων των δέντρων παρέχει την ίδια μεροληψία με αυτή που παρέχει ένα δέντρο από μόνο του, οπότε πετυχαίνουμε βελτίωση μέσω μείωσης της διακύμανσης [14].

### 4.2 Περιγραφή της μεθόδου

Η διαδικασία που ακολουθείται είναι αρκετά απλή ως προς την κατανόησή της. Πριν από κάθε split διαλέγουμε τυχαία κάποιες μεταβλητές εισόδου ως υποψήφιες για διαχωρισμό. Έστερα, από αυτές τις μεταβλητές επιλέγουμε το split point που θα μας δώσει το μεγαλύτερο κέρδος πληροφορίας (όπως ειπώθηκε στο κεφάλαιο 2.2) και την μεταβλητή στην οποία θα κάνουμε τον διαχωρισμό. Έπειτα κάνουμε το split και παράγουμε δύο κόμβους - παιδιά οι οποίοι μας δίνουν την βέλτιστη πληροφορία. Η διαδικασία αυτή επαναλαμβάνεται αναδρομικά για κάθε δέντρο και για πολλά διαφορε-

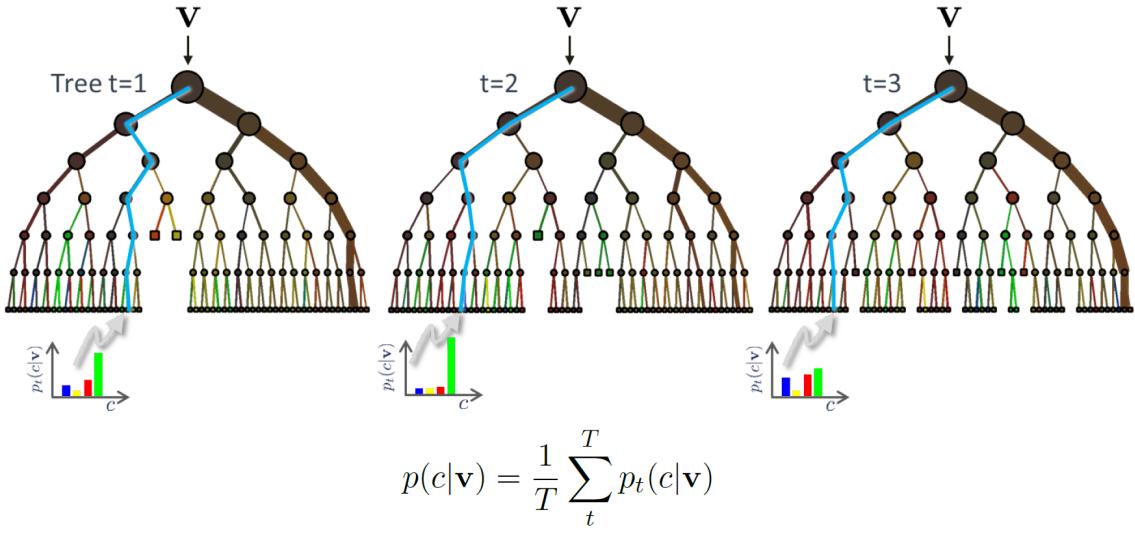


Σχήμα 4.1: Εκπαίδευση ενός δυαδικού δέντρου

τικά τυχαία δέντρα. Στο τέλος η έξοδός μας βασίζεται πάνω στο σύνολο των δέντρων και αποτελεί τον μέσο όρο τους.

Στο σχήμα 4.1 βλέπουμε ένα παράδειγμα εκπαίδευσης ενός random classification tree. Αρχικά διαλέγουμε τυχαία κάποια δεδομένα από το σύνολο εκπαίδευσης, και ύστερα  $V$ -πλήθος δεδομένων χρησιμοποιούνται ώστε να βελτιστοποιήσουν τις παραμέτρους του δέντρου. Τα split points που μας δίνουν το καλύτερο κέρδος πληροφορίας χρησιμοποιούνται για να σπάσουμε τους κόμβους του δέντρου σε νέους κόμβους  $S^L$  και  $S^R$ . Παρατηρούμε πως όσο πηγαίνουμε από τον κόμβο προς τα φύλλα η εντροπία της κατανομής μειώνει, δηλαδή αυξάνεται η εμπιστοσύνη ως προς την κατανομή των μεταβλητών σε κλάσεις. Στο παράδειγμα η μεταβλητή έχει πολύ μεγάλη πιθανότητα να ανήκει στην κλάση με κόκκινο χρώμα, αν ακολουθήσουμε την διαδρομή από την ρίζα στο φύλλο  $C$ . Στην αρχική μας ρίζα όλες οι μεταβλητές έχουν την ίδια κατανομή πιθανότητας.

Στο σχήμα 4.2 έχουμε ένα παράδειγμα δημιουργίας εξόδου για κάποιο από τα δεδομένα μας, έστω  $v$ . Το κάθε τυχαίο δέντρο παράγει προβλέψεις κατηγοροποίησης για το δεδομένο, οι οποίες έχουν προέλθει έστερα από διαδοχικές διαιρέσεις. Οι προβλέψεις είναι διαφορετικές για κάθε δέντρο και τα δέντρα εκπαιδεύονται ξεχωριστά (και πιθανόν παράλληλα). Έστερα κατά την φάση της δοκιμής του μοντέλου, το σημείο  $v$  ωθείται ταυτόχρονα σε όλα τα δέντρα (ξεκινώντας από την ρίζα) μέχρι να φτάσει στο κατάλληλο φύλλο όπου θα λάβει μία εκτίμηση. Στο παράδειγμα, το δέντρο  $t = 2$  παράγει την πιο σίγουρη πρόβλεψη ότι το σημείο θα ανήκει στην κλάση με πράσινο χρώμα, ενώ το δέντρο  $t = 3$  έχει μία πιο ομοιόμορφη κατανομή πιθανοτήτων. Η κάθε πρόβλεψη για το φύλλο θα είναι  $p_t(c|v)$ .



Σχήμα 4.2: Έξοδος ως το σύνολο (ensemble) εκπαίδευσης

Η τελική μας πρόβλεψη για το σημείο όταν χρησιμοποιεί το σύνολο των τυχαίων δέντρων (ensemble) και όταν είναι απλά ο μέσος όρος των προβλέψεων για το σημείο αυτό, επομένως [15]:

$$p(c|\mathbf{v}) = \frac{1}{T} \sum_t^T p_t(c|\mathbf{v}) \quad (4.1)$$

Ένα ερώτημα που προκύπτει είναι γιατί να χρησιμοποιούμε τον μέσο όρο των τυχαίων δέντρων και όχι κάποιο άλλο κριτήριο για την τελική μας έξοδο. Διαισθητικά, ο μέσος όρος ενός μεγάλου αριθμού από τυχαία δέντρα το καθένα από τα οποία μας δίνει μία πρόβλεψη πάνω σε κάποιο σημείο όταν παράγει καλύτερα αποτελέσματα από την πρόβλεψη ενός τυχαίου δέντρου, αφού όταν λαμβάνει υπόψη πολλές διαφορετικές περιπτώσεις που καταλήγουν στο ίδιο σημείο. Ο ensemble μέσος όρος των τυχαίων δέντρων που προέκυψεν από μικρά υποσύνολα των δεδομένων είναι καλύτερος και από το να κάνουμε split ολόκληρο το σύνολο (όπως έγινε στο CART) για τον ίδιο λόγο με πριν, αφού τα αποτελέσματα που μπορεί να μας δώσει σταθεροποιούνται γύρω από κάποιο σημείο χωρίς μεγάλη διακύμανση ενώ αν κάνουμε την ίδια διαδικασία σε ολόκληρο το σύνολο είναι πολύ πιθανό το split που όπως και η έξοδος να μην δίνουν τόσο καλά αποτελέσματα. Πιο τεχνικά, όπως ειπώθηκε και πριν, τα τυχαία δέντρα είναι identically distributed. Ο μέσος όρος  $B$  identically distributed μεταβλητών έχει διακύμανση ίση με:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \quad (4.2)$$

Καθώς το  $B$  αυξάνει, ο δεύτερος όρος εξαφανίζεται αλλά ο πρώτος μένει, επομένως όσο αυξάνει το σύνολο των δεδομένων μας τόσο λιγότερο μειώνει η διακύμανση. Επίσης, όσο πιο συσχετισμένα είναι τα δέντρα μεταξύ τους τόσο περιορίζονται τα οφέλη του μέσου όρου. Η ιδέα στα random forests είναι να μειώσουμε την συσχέτιση ανάμεσα στα δέντρα, χωρίς να μειώσουμε πολύ την διακύμανση. Αυτό πετυχαίνεται μέσω της τυχαίας επιλογής των μεταβλητών. Ειδικότερα πριν από κάθε split διαλέγουμε  $m \leq p$  από τις μεταβλητές εισόδου τυχαία ως υποψήφιες προς διαχωρισμό. Τυπικές τιμές για  $m$  είναι  $\sqrt{p}$  έως και 1.

Όταν  $B$  τέτοια δέντρα  $\{T(x; \Theta_b)\}_1^B$  δημιουργηθούν, τότε ο μέσος όρος των δέντρων αυτών θα είναι η έξοδός μας. Η μείωση του  $m$  θα μείωνε την συσχέτιση οποιουδήποτε ζεύγους δέντρων στο σύνολο των τυχαίων μας δέντων, οπότε θα μείωνε και την διακύμανση στον μέσο όρο. Αυτός είναι και ο λόγος που τα μοντέλα CART δεν παράγουν τόσο καλά αποτελέσματα όσο τα random forests: λαμβάνουν υπόψη ένα πολύ μεγάλο σύνολο δεδομένων και όχι μικρότερα σύνολα, οπότε έχουν και μεγαλύτερη διακύμανση στην έξοδο και τα αποτελέσματα δεν είναι πάντα τόσο καλά. Επομένως στα random forests, λόγω της μικρής συσχέτισης των δέντρων μεταξύ τους, μέσω του μέσου όρου μπορούμε να πετύχουμε πολύ καλά αποτελέσματα λόγω μείωσης της διακύμανσης.

### 4.3 Αλγόριθμος Random Forests

Σύμφωνα με τα παραπάνω, μπορούμε να ορίσουμε τον αγλόριθμο για random forests ως εξής [4]:

Έστω το σύνολο με  $N$  δεδομένα  $\{(X_1, Y_1), (X_2, Y_2), \dots, (X_N, Y_N)\}$  με  $X_i$  predictors (μεταβλητές εισόδου) με  $p$ -χαρακτηριστικά και  $Y_i$  responses (μεταβλητές εξόδου) (κεφάλαιο 1.2). Έστω επίσης  $B$  το σύνολο των δέντρων που θα δημιουργηθούν.

1. Για  $b = 1$  έως  $B$ :

- (α') Διάλεξε ένα bootstrap δείγμα  $\mathbf{Z}^*$  μεγέθους  $N$  από τα δεδομένα εκπαίδευσης.
- (β') Ανέπτυξε ένα τυχαίο δέντρο  $T_b$  στα δείγμα που πήρες, με το να επαναλάβεις αναδρομικά τα παρακάτω βήματα για κάθε τερματικό κόμβο του δέντρου, έως ότου φτάσεις σε κάποιο ελάχιστο μέγεθος κόμβων  $n_{\min}$ .
  - i. Διάλεξε  $m$  τυχαία χαρακτηριστικά από τα  $p$ -χαρακτηριστικά.
  - ii. Επέλεξε την καλύτερη μεταβλητή/σημείο διαχωρισμού από τα  $m$ .
  - iii. Διαίρεσε τον κόμβο σε δύο κόμβους παιδιά.

2. Η έξοδος είναι το σύνολο (ensemble) των δέντρων  $\{T_b\}_1^B$ .

Για να κάνουμε μία πρόβλεψη στο νέο σημείο  $x$ :

*Regression:*

$$\hat{f}_{rf}^B(x) = \frac{1}{B} \sum_{b=1}^B T(x; \Theta_b) \quad (4.3)$$

*Classification:* Έστω  $\hat{C}_b(x)$  η πρόβλεψη για την κλάση του  $b$ -δέντρου. Τότε:

$$\hat{C}_{rf}^B(x) = \text{majority vote } \left\{ \hat{C}_b(x) \right\}_1^B \quad (4.4)$$

Με τον όρο bootstraping εννοούμε την επιλογή ενός τυχαίου υποσυνόλου από το σύνολο δεδομένων για εκπαίδευση και ύστερα την δοκιμή του μοντέλου που έχει εκπαιδευτεί στο υποσύνολο που δεν επλέχθηκε ώστε να παραχθούν διάφορα συμπεράσματα σχετικά με την ακρίβεια του μοντέλου (για παράδειγμα προκατάληψη (bias), διακύμανση, διαστήματα εμπιστοσύνης, σφάλμα πρόβλεψης) [2].

## 4.4 Out of Bag Samples

Ένα σημαντικό χαρακτηριστικό του αλγορίθμου είναι ότι χρησιμοποιεί δείγματα out of bag (OOB), τα οποία ορίζονται ως εξής:

Για κάθε παρατήρηση  $z_i = (x_i, y_i)$  δημιουργησε έναν random forest predictor μέσω του μέσου όρου των δέντρων που αντιστοιχούν στα δείγματα στα οποία δεν βρίσκεται ο  $z_i$ .

Μια εκτίμηση σφάλματος OOB είναι σχεδόν ταυτόσημη με αυτή που λαμβάνουμε με ένα N-fold Cross Validation. Σε αντίθεση όμως με άλλους μη γραμμικούς εκτιμητές, μπορούμε καθώς τροφοδοτούμε το δέντρο σε μία ακολουθία να εκτελούμε παράλληλα cross validation. Μόλις το σφάλμα OOB σταθεροποιηθεί, η εκπαίδευση μπορεί να σταματήσει.

## 4.5 Feature Importances

Ένα πολύ σημαντικό χαρακτηριστικό του αλγορίθμου Random Forest όπως και του αλγορίθμου CART είναι η σημαντικότητα των predictors. Σε κάθε διαχωρισμό και σε κάθε δέντρο ξεχωριστά, η επιπλέον βελτίωση στην απόδοση του αλγορίθμου μέσω του συγκεχριμένου predictor και σύμφωνα με το κριτήριο διαχωρισμού είναι το μέτρο σημαντικότητας της μεταβλητής, και συσσωρεύεται για όλα τα δέντρα στο δάσος και ξεχωριστά για κάθε μεταβλητή. Στον αλγόριθμο Random Forest λόγω του κριτηρίου διαχωρισμού, η πιθανότητα όλες οι μεταβλητές να έχουν ρόλο στο τελικό δέντρο, ακόμα και μικρό, είναι πολύ αυξημένη, ειδικά σε σχέση με άλλες μεθόδους όπως το gradient boosting.

Επιπρόσθετα, στον αλγόριθμο Random Forest μπορούμε μέσω των OOB samples να δημιουργήσουμε ένα μοντέλο σημαντικότητας που μετρά την προβλεπτική ικανότητα της κάθε μεταβλητής. Όταν το δέντρο  $b$  μεγαλώνει, μέσω των OOB samples μπορούμε να μετρήσουμε την ακρίβεια πρόβλεψης. Έπειτα οι τιμές για την μεταβλητή  $j$  μετατίθενται τυχαία στα οοθέτη στα δείγματα και η ακρίβεια υπολογίζεται ξανά. Η μέση τιμή της μείωσης της ακρίβειας ως αποτέλεσμα των μετανέσεων σε όλα τα δέντρα είναι ένας δείκτης της σημαντικότητας της μεταβλητής  $j$  στο random forest.

## 4.6 Random Forests and Overfitting

Όταν στο δείγμα έχουμε μικρό αριθμό σημαντικών μεταβλητών τότε ο αλγόριθμος Random Forest εμφανίζει χειρότερη επίδοση κάθε φορά που αυξάνουμε τον αριθμό των μεταβλητών που δίνουν θόρυβο στην έξοδο. Όταν ο αριθμός των σημαντικών μεταβλητών αυξάνεται, έχουμε ισχυρή επίδοση ακόμα και με πολλές μεταβλητές με θόρυβο. Για παράδειγμα, με 6 σημαντικές και 100 μεταβλητές με θόρυβο, η πιθανότητα μία σημαντική μεταβλητή να επιλεγεί σε οποιοδήποτε split είναι 0.46, θεωρώντας  $m = \sqrt{6 + 100} \approx 10$ .

Ένα ακόμα σημαντικό γεγονός είναι πως ο αλγόριθμος δεν δημιουργεί overfitting στα δεδομένα, ακόμα και σε αύξηση των δέντρων  $B$ . Όπως και στο bagging, το Random Forest προσεγγίζει την προσδοκία:

$$\hat{f}_{rf}(x) = E_{\Theta} T(x; \Theta) = \lim_{B \rightarrow \infty} \hat{f}(x)_{rf}^B \quad (4.5)$$

με μέσο όρο των  $B$  δέντρων στην κατανομή  $\Theta$ . Η κατανομή για πολύ υψηλές τιμές μπορεί να κάνει overfit στα δεδομένα, δημιουργώντας ένα πολύ πλούσιο μοντέλο το οποίο να δίνει επιπλέον πληροφορία που δεν χρησιμεύει. Όμως η αύξηση της επίδοσης ενός μοντέλου με ελεγχόμενο βάθος δέντρων στον αλγόριθμο Random Forest είναι μηδαμινή [4]. Την επίδοση δέντρων με διαφορετικά βάθη στους αλγορίθμους CART και Random Forest μελετάμε στο κεφάλαιο 7. Πράγματι εκεί παρατηρούμε πως για μεγάλα βάθη η επίδοση του αλγορίθμου Random Forest είναι σταθερή, σε αντίθεση με τον αλγόριθμο CART που μπορεί να κάνει overfit των δεδομένων.

## Μέρος II

Ανάλυση αγοράς ηλεκτρικής  
ενέργειας



## Κεφάλαιο 5

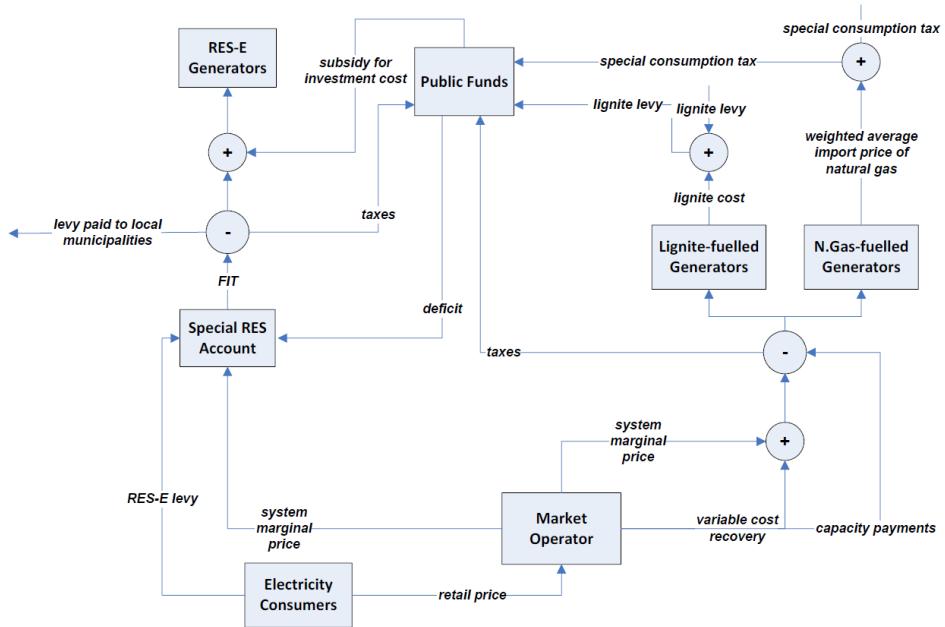
# Αγορά Ηλεκτρικής Ενέργειας

### 5.1 Εισαγωγή

Στην παρούσα εργασία θα ασχοληθούμε με την αγορά ηλεκτρικής ενέργειας και τις σχέσεις μεταξύ των παραγόντων που την καθορίζουν. Τα στοιχεία που παραθέτουμε βρίσκονται με περισσότερη λεπτομέρεια στο paper Analysis of the efficiency of RES του Σωτήρη Παπαδέλη που δημοσιεύτηκε στο Apraise στο article με τίτλο Assessment of Policy Impacts on Sustainability in Europe [1]. Οι κύριες χρηματικές ροές στην ελληνική αγορά ηλεκτρικής ενέργειας φαίνονται στο σχήμα 5.1. Η τιμή της χονδρεμπορικής αγοράς ηλεκτρικής ενέργειας (system marginal price - SMP) είναι αποτέλεσμα του Day-Ahead Scheduling (DAS) process και έχει έναν εξαίρετο ρόλο στον καθορισμό σημαντικού μέρους της ηλεκτρικής ενέργειας που παράγεται στην αγορά. Οι προσφορές στο DAS σε περίπτωση που δεν ικανοποιηθούν από την παραγωγική μονάδα την εκμέτουν, με αποτέλεσμα να υπάρχει πρόστιμο για την μη ισορροπία του συστήματος στην μορφή ex-post imbalance price. Ο Ανεξάρτητος Διαχειριστής Μεταφοράς Ηλεκτρικής Ενέργειας (ΑΔΜΗΕ) καθορίζει τα οριακά αυτά κόστη (System Imbalances Marginal Price (SIMP)).

### 5.2 Variable Cost Recovery Mechanism και Capacity Adequacy Mechanism

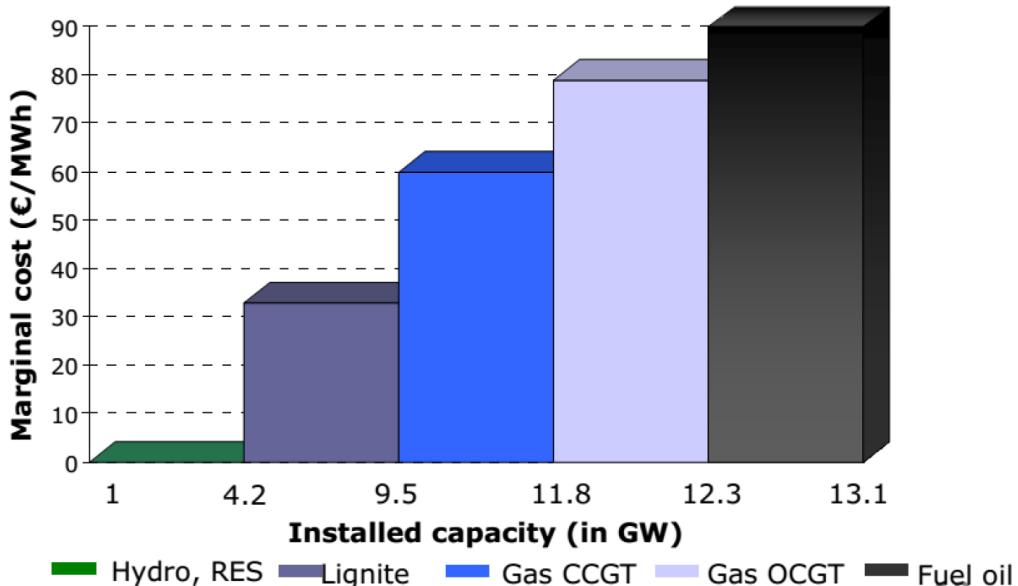
Από την 1/5/2008 έχει τεθεί σε ισχύ ένας μηχανισμός ανάκτησης μεταβλητού κόστους (Variable Cost Recovery Mechanism) ο οποίος δίνει πληρωμές σε μονάδες παραγωγής ίσες με 10% του μεταβλητού τους κόστους σε περίπτωση που δεν μπορούν να το επιτύχουν μέσω των εσόδων από την αγορά ενέργειας. Ο μηχανισμός αυτός λειτουργεί σε συνδυασμό με το γεγονός πως οι μη ανανεώσιμες μονάδες παραγωγής μπορούν να προσφέρουν έως 30% της δυναμικότητάς τους σε τιμές χαμηλότερες του μεταβλητού τους κόστους εφόσον η μέση τιμή που προσφέρουν την ενέργεια στην αγορά είναι ίση ή μεγαλύτερη του μέσου μεταβλητού τους κόστους.



Σχήμα 5.1: Χρηματικές ροές στην ελληνική αγορά ηλεκτρικής ενέργειας

Ο μηχανισμός επάρκειας ισχύος (Capacity Adequacy Mechanism) προσφέρει σε μονάδες παραγωγής μη ανανεώσιμων πηγών ενέργειας πληρωμές μέσω των οποίων οι μονάδες είναι σε θέση να ανακτήσουν μέρος των πάγιων εξόδων τους. Κάθε παραγωγός παίρνει για τα επόμενα πέντε έτη αξιοπιστίας έναν αριθμό από αποδεικτικά διαθεσιμότητας ισχύος (Capacity Availability Tickets), που το κάθε ένα είναι έγκυρο για ένα έτος αξιοπιστίας. Ο συνολικός αριθμός αποδεικτικών για κάθε μονάδα είναι ίσος με την καθαρή χωρητικότητα της μονάδας. Κάθε έτος αξιοπιστίας ο ΑΔΜΗΕ υπολογίζει την διαθέσιμη χωρητικότητα της κάθε μονάδας με βάση το Demand Equivalent Forced Outage Rate (EFORd) και κατανέμει τους πόρους ισόποσα στα αποδεικτικά διαθεσιμότητας. Επομένως κάθε αποδεικτικό έχει αξία χωρητικότητας 1-EFORd. Κάθε παραγωγός μπορεί να συνάψει σύμβαση με την ΑΔΜΗΕ ώστε να λαμβάνει ένα ποσό ίσο με την διαθέσιμη χωρητικότητα του αποδεικτικού πολλαπλασιασμένο με μία ποινή μη συμμόρφωσης  $P^{NCP}$  για τις ώρες που η μονάδα έχει δηλωθεί διαθέσιμη στην αγορά. Μία προσέγγιση της τιμής που λαμβάνει μία μονάδα  $j$  είναι:

$$P^{NCP} \cdot (1 - EFORd_j)^2 \quad (5.1)$$



Σχήμα 5.2: Καμπύλη προσφοράς ελληνικής αγοράς ηλεκτρικής ενέργειας

### 5.3 ΑΠΕ και καμπύλη προσφοράς

Σύμφωνα με το ισχύον κανονιστικό πλαίσιο, όλη η παραγωγή από ΑΠΕ πρέπει να τροφοδοτείται στο δίκτυο ηλεκτρικής ενέργειας. Οι μονάδες συμβατικής παραγωγής (λιγνίτης, φυσικό αέριο, υδροηλεκτρικά) καλύπτουν το υπόλοιπο φορτίο. Ως αποτέλεσμα, η παραγωγή από ΑΠΕ μπορεί να θεωρηθεί ως αρνητική ζήτηση που επηρεάζει τα επίπεδα του υπολειμματικού φορτίου. Ένας τρόπος για να κατανοήσουμε την υποκατάσταση παραγωγής από ΑΠΕ είναι να χρησιμοποιήσουμε την έννοια του οριακού κόστους, σύμφωνα με την οποία οι γεννήτριες έχουν μπει σε μια στοίβα παραγωγής (η λεγόμενη σειρά προτεραιότητας), που χυμαίνεται από το χαμηλότερο στο υψηλότερο οριακό κόστος. Η στοίβα της παραγωγής είναι τότε η συνάρτηση προσφοράς στο επίπεδο της αγοράς και χρησιμοποιείται για τον προσδιορισμό του επιπέδου λειτουργίας (δηλαδή της ποσότητας της παραγόμενης ενέργειας) από κάθε γεννήτρια για κάθε δεδομένο επίπεδο ζήτησης. Όσο υψηλότερο είναι το φορτίο τόσο περισσότερο προς τα δεξιά μετατοπίζεται η τομή της ζήτησης με την καμπύλη προσφοράς. Στο σχήμα 5.2 απεικονίζεται η καμπύλη προσφοράς στην ελληνική αγορά ηλεκτρικής ενέργειας. Πολύ σημαντικό να αναφερθεί πως η ποσότητα της ηλεκτρικής ενέργειας που παράγεται πρέπει να είναι πάντα ίση με την ζητούμενη ποσότητα συν τις επιπλέον ζημίες. Επομένως, μια αύξηση στην παραγωγή από ΑΠΕ πρέπει να οδηγήσει σε ισόποση μείωση της παραγωγής από μη ανανεώσιμες πηγές ενέργειας.

## 5.4 Παράμετροι που οδηγούν τις τιμές

Το σύνολο των θεμελιωδών παραμέτρων που οδηγούν τις τιμές της ηλεκτρικής ενέργειας περιλαμβάνει:

- Η ζήτηση της ηλεκτρικής ενέργειας. Όσο υψηλότερο είναι το φορτίο τόσο περισσότερο μετατοπίζεται προς τα δεξιά η τομή της καμπύλης ζήτησης με την καμπύλη προσφοράς. Όσο η τομή μετατοπίζεται προς τα δεξιά, τόσο αυξάνει το οριακό κόστος για την εξυπηρέτηση της ζήτησης.
- Η παραγωγή από ΑΠΕ η οποία θεωρείται αρνητική ζήτηση.
- Οι τιμές των καυσίμων οι οποίες επηρεάζουν το μεταβλητό κόστος των σταθμών παραγωγής. Οι τιμές του λιγνίτη έχουν μικρότερη επιρροή από ότι του φυσικού αερίου διότι:
  1. Είναι γενικά λιγότερο ευμετάβλητες από τις τιμές του φυσικού αερίου
  2. Κατά κύριο λόγο επηρεάζουν συνήθως το χαμηλότερο επίπεδο τμήμα της καμπύλης προσφοράς.
- Η διαθεσιμότητα των πόρων. Η μη διαθεσιμότητα των καυταρών τεχνολογιών βασικού φορτίου, όπως ο λιγνίτης και τα πυρηνικά, μεταφέρουν την καμπύλη προσφοράς προς τα αριστερά, το οποίο ισοδυναμεί με τη διατήρηση της καμπύλης ως σημείο αναφοράς και την αύξηση του φορτίου κατά το ίδιο ποσό. Από την άλλη πλευρά η μη διαθεσιμότητα των τεχνολογιών που εξυπηρετούν τα φορτία βάσης και αιχμής όπως είναι το φυσικό αέριο, αντανακλάται κυρίως στο περιθώριο χωρητικότητας της αγοράς. Το περιθώριο αυτό υπολογίζεται ως το φορτίο διαιρεμένο με την συνολική διαθέσιμη χωρητικότητα.

## 5.5 Φυσικό Αέριο

Σε μια αγορά όπου το μέσο φορτίο είναι σταθερό θα θέλαμε να βρειθεί μια σταθερή σχέση μεταξύ του ηλεκτρισμού και των τιμών των καυσίμων. Οι Jong και Schneider (2009) μελέτησαν αγορές φυσικού αερίου σε σχέση με την αγορά ανταλλαγής ηλεκτρικής ενέργειας του Amsterdam, δεδομένου ότι οι αγορές αυτές συνδέονται στενά λόγω φυσικής μεταφοράς. Το μοντέλο που ανέπτυξαν δείχνει πως το φυσικό αέριο και οι τιμές ηλεκτρικής ενέργειας έχουν σχέση συνολοκλήρωσης σε μακροπρόθεσμο επίπεδο τιμών [16].

Οι Bosco et al. (2010) εξετάζουν τις αλληλεξαρτήσεις που υπάρχουν στις τιμές ηλεκτρικής ενέργειας σε έξι μεγάλες ευρωπαϊκές χώρες. Τα αποτελέσματα της ανάλυσης τους αποκαλύπτουν την παρουσία τεσσάρων ολοκληρωμένων αγορών της κεντρικής Ευρώπης (Γαλλία, Γερμανία, Κάτω Χώρες, Αυστρία). Η τάση που υπάρχει σε αυτές τις τέσσερις αγορές ηλεκτρικής ενέργειας φαίνεται να είναι κοινή και για τις τιμές του φυσικού αερίου [11].

Σε άλλη έρευνα, οι Ferklingstad et al. (2010) βρίσκουν ότι οι τιμές του φυσικού αερίου έχουν ισχυρή συσχέτιση με τις τιμές της ηλεκτρικής ενέργειας ενώ ο άνθρωπος και το πετρέλαιο είναι λιγότερο σημαντικές μεταβλητές στον καθορισμό της τιμής [18].

Οι Furio και Chulia (2012) χρησιμοποιούν το μοντέλο VECM για να διερευνήσουν τους αιτιώδεις δεσμούς μεταξύ της ηλεκτρικής ενέργειας της Ισπανίας, του αργού πετρελαίου και του φυσικού αερίου ένα μήνα πριν τις προθεσμιακές τιμές. Τα ευρήματά τους αποκαλύπτουν ότι το αργό πετρέλαιο και το φυσικό αέριο έχουν έναν εξέχοντα ρόλο στην διαδικασία διαμόρφωσης των τιμών της ηλεκτρικής ενέργειας της ισπανικής αγοράς [23].

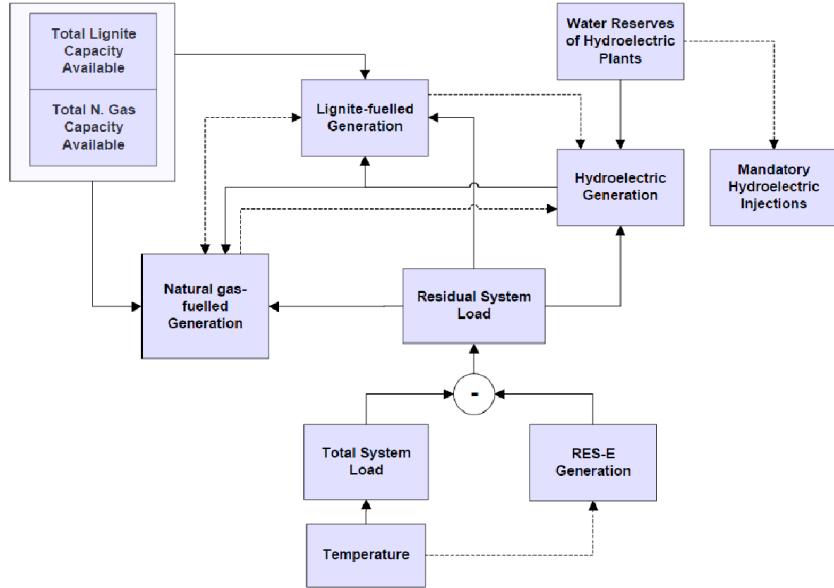
Στην παρούσα διπλωματική θα γίνει μία αναζήτηση για το αν υπάρχει αντίστοιχη αιτιώδης σχέση μεταξύ του φυσικού αερίου και των τιμών της ηλεκτρικής ενέργειας.

## 5.6 Αιτιώδεις σχέσεις μεταξύ μεταβλητών

Προκειμένου να διαμορφωθούν οι σχέσεις των μεταβλητών υπό μελέτη, αρχικά πρέπει να αποφασιστεί ποιοί θα είναι οι predictors που θα χρησιμοποιηθούν στην ανάλυση. Το σύνολο των μεταβλητών που θα βρεθούν θα πρέπει να έχει τις ακόλουθες ιδιότητες:

1. Οι μεταβλητές είναι εξωγενείς υπό την έννοια ότι δεν υπάρχει αντίστροφη αιτιώδης συνάρτεια από την εξαρτημένη μεταβλητή σε έναν ή περισσότερους predictors.
2. Χρησιμοποιούνται όλες οι σχετικές με το μοντέλο μεταβλητές. Αυτό γίνεται διότι αν είχαμε παραλήψεις μεταβλητών τότε το σφάλμα του μοντέλου θα περιείχε τους παράγοντες που είχαν παραληφθεί και που απαιτούνται για να εξηγήσουν τη διακύμανση της εξαρτημένης μεταβλητής, οπότε και το μοντέλο δεν θα ήταν τόσο ακριβές. Αυτό θα οδηγούσε σε μια συσχέτιση μεταξύ των παλινδρόμησης και του σφάλματος και, ως εκ τούτου, σε ψευδή αποτελέσματα.
3. Με αντίστοιχο τρόπο, οι περιττές επεξηγηματικές μεταβλητές που προσθέτουν θόρυβο στο σύστημα αποκλείονται. Επιπλέον, η συμπερίληψη πολλαπλών συσχετισμένων μεταβλητών στο μοντέλο θα κάνει τους επιμέρους συντελεστές της κάθε μεταβλητής να υπολογίζονται με λανθασμένο τρόπο.

Οι αιτιώδεις σχέσεις για την ελληνική αγορά ηλεκτρικής ενέργειας παρουσιάζονται στο σχήμα 5.3. Τα συμπαγή βέλη αντιπροσωπεύουν σχέσεις με βέβαιη αιτιώδη συνάρτεια, ενώ τα διακεκομμένα σχέσεις που είναι εύλογο ότι υπάρχουν, αλλά προϋπονθέτουν έλεγχο για αυτό.



Σχήμα 5.3: Αιτιώδεις σχέσεις στην ελληνική αγορά ηλεκτρικής ενέργειας

Ο όρος αιτιώδης συνάφεια χρησιμοποιείται για να δείξει οτι:

*Αν το σύνολο  $z_t$  περιέχει μια σειρά από προσεκτικά επιλεγμένες επεξηγηματικές μεταβλητές μπορούμε να κάνουμε μία πρόβλεψη με βάση τις lagged μεταβλητές της εξόδου  $Y_t$  όπως και του  $Z_t$ . Αν η πρόβλεψη αυτή μπορεί να βελτιωθεί μέσω των lagged values της μεταβλητής  $W_t$  (η οποία περιέχει πληροφορίες για την έξοδο  $Y_{t+1}$  υπάρχει μία αιτιώδης σχέση (Granger causal flow) μεταξύ των  $W_t$  και  $Y_t$  ( $W_t \rightarrow Y_t$ ).*

Θα πρέπει να αναμένουμε ότι η ροή Granger μεταξύ των ποσοτήτων που παράγονται, της παραγωγής από ΑΠΕ, του συνολικού φορτίου και της συνολικής διαθέσιμης χωρητικότητας, έχει μια τέτοια κατεύθυνση ώστε οι μεταβλητές μπορούν να θεωρηθούν ως εξωγενείς. Ειδικά σε σχέση με το φορτίο, οι καταναλωτές δεν ανταποκρίνονται άμεσα στις τιμές χονδρικής, δεδομένου ότι αγοράζουν ηλεκτρική ενέργεια σε σταθερές τιμές που αλλάζουν μόνο μετά από πολλά χρόνια. Κατά συνέπεια, για τις περιόδους όπου οι καταναλωτές αντιμετωπίζουν μια σταθερή τιμή για την ηλεκτρική ενέργεια, οι αλλαγές στη ζήτηση οδηγούνται από εξωγενείς δυνάμεις, οι οποίες δεν περιλαμβάνουν την τιμή χονδρικής ή παράγοντες που την επηρεάζουν, όπως η ποσότητα παραγωγής ηλεκτρικής ενέργειας ανά τύπο καυσίμου. Από την άλλη πλευρά, η αιτιώδης σχέση μεταξύ του φυσικού αερίου και της παραγωγή λιγνίτη, καθώς και μεταξύ αυτών και της υδροηλεκτρικής παραγωγής, θα πρέπει να εξεταστεί λεπτομερέστερα.

# Κεφάλαιο 6

## Προεπεξεργασία δεδομένων

### 6.1 Δεδομένα

Κατά την ανάλυση χρησιμοποιήθηκαν δεδομένα από την αγορά ηλεκτρικής ενέργειας, ώστε να παραχθεί το τελικό μοντέλο. Τα δεδομένα ήταν σε μορφή comma-separated values(csv). Ολόκληρη η ανάλυση και οι αλγόριθμοι που χρησιμοποιήθηκαν έγιναν στην γλώσσα python. Το SMP είναι η έξοδος για που θέλουμε να προβλέψουμε με βάση τα χαρακτηριστικά X που θα έχουμε ως είσοδο.

Στα παραδείγματα που θα ακολουθήσουν παρουσιάζονται snippets από κώδικα και όχι ολόκληρος ο κώδικας ώστε να γίνουν κατανοητές εύκολα οι έννοιες και να μην χαθεί ο αναγνώστης. Η μελέτη έγινε στην γλώσσα προγραμματισμού python με χρήση βελτιστοποιημένων βιβλιοθηκών για machine learning οι οποίες χρησιμοποιούνται από ερευνητές αλλά και μεγάλες εταιρίες του χώρου (11).

#### 6.1.1 SMP

Τιμή της ηλεκτρικής ενέργειας στη χονδρεμπορική αγορά στο διάστημα από 1/1/2008 έως 30/9/2013. Η πρώτη στήλη περιέχει ημερομηνίες ενώ οι υπόλοιπες 24 στήλες περιέχουν την τιμή της ηλεκτρικής ενέργειας για κάθε ώρα της ημέρας. Να σημειωθεί εδώ πως όλα τα χαρακτηριστικά που θα εξεταστούν στην ανάλυση αφορούν ημερήσιες τιμές στο διάστημα από 1/1/2009 έως 30/9/2013 επομένως το SMP θα έχει περισσότερες γραμμές από ότι στο τελικό αρχείο.

```
1 import pandas as pd
2 import datetime
3 path = "C:/Users/Dimitris/Desktop/diploma/data/"
4 df = pd.read_csv(path + "smp.csv", sep=',', converters={0:str2date})
5 df = df.set_index('Unnamed: 0')
6 runfile('C:/Users/Dimitris/Desktop/diploma/data/data2.py')
7 df.describe()
```

	1	2	3	4	5	6	7	8	9	10	...	15	16	17	18	19	20	21	22	23	24
<b>count</b>	2091.	2091.	2091.	2090.	2091.	2091.	2091.	2091.	2091.	2091.	...	2091.	2091.	2091.	2091.	2091.	2091.	2091.	2091.	2091.	2087.
	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000	00000
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
<b>mean</b>	50.70	44.04	40.22	36.45	33.63	34.31	38.14	47.10	56.78	62.30	...	58.79	57.01	57.68	60.55	64.44	66.68	67.56	65.90	61.31	54.05
	0774	7787	7556	6466	5078	5511	3203	2684	5546	5871		4626	7913	7209	4020	1801	5509	2447	8218	7047	2737
<b>std</b>	21.83	18.26	17.03	16.17	15.75	15.06	16.89	21.05	24.61	25.93	...	26.28	26.40	26.49	27.07	26.69	26.14	25.73	24.67	23.70	21.98
	1653	3886	6625	8099	1681	9106	9938	6612	9226	1790		9371	9496	2007	5303	9740	0864	8833	4212	6659	2860
<b>min</b>	0.000	0.000	0.000	0.000	0.000	0.000	-0.00	0.000	0.000	0.000	...	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
	000	000	000	000	000	000	1000	000	000	000		000	000	000	000	000	000	000	000	000	000
<b>25%</b>	35.48	33.01	31.83	30.50	29.80	29.98	30.70	34.03	36.98	38.97	...	37.85	36.86	37.04	38.13	40.02	41.91	43.01	42.51	39.74	37.16
	0000	1000	6000	4000	2095	1996	6000	0000	2500	6500		4500	8000	1500	0500	0500	6000	2500	7000	2500	5500
<b>50%</b>	39.87	37.15	36.01	34.64	33.53	33.64	35.38	38.48	47.87	59.56	...	55.35	49.39	49.39	54.73	60.85	63.82	64.70	63.43	57.93	43.76
	2000	8000	3000	9000	2310	6339	9000	2000	5000	1000		0000	2000	1000	3000	5000	0000	0000	2000	1000	5000
<b>75%</b>	63.64	49.44	40.60	38.03	36.98	37.23	38.85	60.41	76.88	84.24	...	80.84	78.02	80.16	83.28	85.51	87.32	87.97	87.08	83.12	68.16
	2239	5244	1000	6000	0500	8000	6500	4500	4373	5500		2000	1378	4776	4000	9500	0629	4500	3500	9425	5000
<b>max</b>	149.8	119.8	109.6	111.7	110.0	110.0	110.0	113.7	149.8	150.0	...	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	150.0	143.0
	00000	67073	42441	20596	00000	00000	00000	04002	50000	00000		00000	00000	00000	00000	00000	00000	00000	00000	00000	00000

8 rows × 24 columns

### Σχήμα 6.1: SMP

Έχουμε 24 στήλες δεδομένων και 2091 γραμμές. Όλες οι στήλες περιέχουν floating point αριθμούς μη αρνητικούς. Παρατηρούμε πως η μέση τιμή του SMP κυμαίνεται κοντά στα 50€ παρουσιάζοντας κάποιες διακυμάνσεις μέσα στην ημέρα. Τις πρώτες ώρες είναι μικρότερη κοντά στα 40€ ενώ αυξάνει τις επόμενες. Η τυπική απόκλιση είναι στα 25€ με μέγιστες τιμές να μην ξεπερνούν τα 150€. Οι ελάχιστες τιμές του SMP είναι μηδενικές, και αυτό οφείλεται στο γεγονός ότι έχουμε missing values στο δείγμα μας τα οποία ωστόσο εξαφανίσουμε κατά την προεπεξεργασία. Κατά την ανάλυση ωστόσο έλαβαμε τις μέσες ημερήσιες τιμές του SMP ώστε να εξάγουμε τα κατάλληλα συμπεράσματα.

#### 6.1.2 Χαρακτηριστικά

Τα χαρακτηριστικά με βάση τα οποία ωστόσο γίνεται η ανάλυσή μας παρουσιάζονται παρακάτω. Σημειώνεται πως στην ανάλυση μας απασχολούν τα συνολικά ημερήσια μεγέθη και όχι τα ωριαία.

**availability** Δυναμικότητα μονάδων παραγωγής λιγνίτη κάθε μέρα σε MW.

**exports** Συνολική ωριαία εξαγωγή ηλεκτρικής ενέργειας σε MW από την Ελλάδα προς τους γείτονές της.

**hydrogen** Συνολική ωριαία παραγωγή υδροηλεκτρικών εργοστασίων σε MW.

**imports** Συνολική ωριαία εισαγωγή ηλεκτρικής ενέργειας σε MW της Ελλάδας.

**lignite** Συνολική ωριαία παραγωγή λιγνίτη σε MW.

**load\_forecast** Συνολική ωριαία ζήτηση.

**ngas** Συνολική ωριαία παραγωγή φυσικού αερίου σε MW.

**res\_forecast** Ωριαία πρόγνωση παραγωγής από ανανεώσιμες πηγές ενέργειας σε MW (αρνητική ζήτηση).

**waters** Συνολική ωριαία παραγωγή από τα υποχρεωτικά νερά σε MW.

**waip** Μηνιαία τιμή φυσικού αερίου.

## 6.2 Προεπεξεργασία

### 6.2.1 Δημιουργία τελικού αρχείου δεδομένων

Αρχικά θα δημιουργήσουμε ένα .csv αρχείο το οποίο θα περιέχει τις συνολικές ημερήσιες τιμές των χαρακτηριστικών όπως και του SMP. Για να το πετύχουμε αυτό αρκεί να πάρουμε το άθροισμα των τιμών κάθε σειράς του αρχείου και να δημιουργήσουμε μία νέα στήλη που να περιέχει τα αντίστοιχα χαρακτηριστικά. Έστερα αυτή την στήλη την αποθηκεύουμε σε ένα νέο αρχείο το οποίο θα είναι το αρχείο με το οποίο θα γίνει η ανάλυση. Στην περίπτωση του SMP χρειαζόμαστε τον μέσο όρο των χαρακτηριστικών κάθε σειράς ενώ στην περίπτωση του waip θα έχουμε την ίδια τιμή σε ολόκληρο τον μήνα. Η υλοποίηση είναι πολύ εύκολη. Έστω για παράδειγμα ότι δημιουργούμε ένα νέο αρχείο out.csv το οποίο θα περιέχει την νέα στήλη για μία τιμή X και έστω ότι έχουμε δύο στηλες στο αρχικό αρχείο, την στήλη a και την στήλη b. Τότε:

- 1 df ['X'] = df['a'] + df['b']
- 2 out['X'] = df['X']

Δημιουργούμε με παρόμοιο τρόπο το τελικό αρχείο το οποίο στην πρώτη στήλη περιέχει τις τιμές του SMP και στις επόμενες τα χαρακτηριστικά οπότε καταλήγουμε να έχουμε το μοντέλο με απόκριση SMP με p-στοιχεία όπου το κάθε στοιχείο είναι μία μέρα μεταξύ 1/1/2008 και 30/9/2013 (σύνολο 1795 στοιχεία) και είσοδο τον πίνακα Y με NxP στοιχεία όπου N ο αριθμός των χαρακτηριστικών, δηλαδή 10. Φορτώνουμε το αρχείο και παίρνουμε τα χαρακτηριστικά που φαίνονται στο σχήμα 6.2.

Το SMP παρουσιάζει την συμπεριφορά που αναφέρθηκε στην προηγούμενη παράγραφο. Παρατηρούμε πως όλες οι μεταβλητές έχουν τιμές που λείπουν. Η συνολική δυναμικότητα που μας δίνουν οι μονάδες κάθε μέρα έχει μικρή τυπική απόκλιση, το οποίο σημαίνει πως η δυναμικότητα των εργοστασίων παραμένει σχεδόν σταθερή στη διάρκεια των ετών που μελετάμε.

	smp	availability	exports	hydrogen	imports	lignite	load_forecast	ngas	res_forecast	waters	waip
count	1722.000000	1714.000000	1609.000000	1739.000000	1609.000000	1770.000000	1722.000000	1770.000000	1722.000000	1787.000000	1765.000000
mean	48.846296	9357.179697	9466.071806	13712.56190	17846.98550	74387.74671	142407.9060	29541.63838	8465.615563	12228.17711	56.728669
std	15.415013	688.455803	4267.527126	7770.084824	5335.078831	12425.90604	17861.02290	10779.41590	6960.201075	8563.366811	13.532449
min	10.237917	4885.000000	0.000000	1704.581273	3305.000000	25740.02385	93970.88638	1806.691500	-16119.0000	1479.000000	31.530000
25%	37.352052	8942.250000	6779.714000	8183.995774	14030.00000	66321.38966	130556.7792	21099.24350	3432.500000	6020.000000	46.690000
50%	45.978292	9366.000000	9818.952000	11546.36398	17673.00000	74519.99670	140456.3680	29267.39500	5985.000000	10067.00000	57.140000
75%	57.808823	9844.000000	12536.76100	16637.78024	21182.85200	83534.41387	153111.5145	36403.79425	11451.50000	14976.50000	68.920000
max	123.773333	10640.00000	20209.88000	41777.50472	36521.41500	103852.4078	199396.0503	65469.85300	37066.00000	43922.00000	78.920000

Σχήμα 6.2: Χαρακτηριστικά μεταβλητών συστήματος

### 6.2.2 Χειρισμός missing values

Για να μπορούν οι αλγόριθμοι που περιγράψαμε στο πρώτο μέρος της εργασίας να δουλέψουν, πρέπει να μην έχουμε τιμές που να λείπουν από τα δεδομένα μας. Υπάρχουν διάφοροι τρόποι χειρισμού μεταβλητών που λείπουν [28].

1. Απόρριψη των δεδομένων που λείπουν. Είναι μία στρατηγική που χρησιμοποιείται συχνά από τους ερευνητές, ιδιαίτερα αν τα δεδομένα που λείπουν είναι τυχαία κατανεμημένα. Στην πράξη, η απόρριψη είναι κατάλληλη όταν είναι εφικτό να αφρηθούμε να κάνουμε μία πρόβλεψη σε συγκεκριμένα instances του δείγματός μας. Στην περίπτωσή μας απαιτούνται προβλέψεις για όλα τα instances των δεδομένων μας.
2. Απόκτηση των δεδομένων που λείπουν. Υπάρχουν περιπτώσεις που η απόκτηση των δεδομένων είναι εφικτή.
3. Εκτίμηση της τιμής που λείπει (imputation). Είτε μία τιμή που λείπει αντικαθίσταται από μία τιμή είτε η κατανομή των πιθανών τιμών που λείπουν εκτιμάται και δημιουργείται ένα μοντέλο που συνδυάζει προβλέψεις πιθανολογικά. Υπάρχουν διάφορα μοντέλα για imputation.
  - (α') (Predictive) Value Imputation (PVI): Τα missing values αντικαθίσταται από εκτιμώμενες τιμές πριν τις βάλουμε στο μοντέλο. Οι σύνηθες περιπτώσεις είναι ο μέσος όρος, η διάμεσος ή η πιο συχνή τιμή.

- (β') Distribution based Imputation (DBI) Δεδομένης μίας διανομής γύρω από μία τιμή, μπορούν να γίνουν διάφορες προβλέψεις γύρω από την τιμή αυτή και η κάθε πρόβλεψη να έχει ένα βάρος, και το αποτέλεσμα να είναι ο συνδυασμός των προβλέψεων αυτών. Παράδειγμα είναι ο αλγόριθμος C4.5.
- (γ') Unique value imputation: Αντί να γίνεται αντικατάσταση όλων των μεταβλητών με την ίδια τιμή, υπάρχουν μοντέλα που προσδιορίζουν μοναδικά τις τιμές που μπορεί να προκύψουν. Αυτά τα μοντέλα ψεωφούνται καλύτερα όταν η μεταβλητή που λείπει εξαρτάται περισσότερο από την τιμή των μεταβλητών της κλάσης που ανήκει η τιμή.

Η βιβλιοθήκη sklearn έχει διάφορους αλγόριθμους για machine learning και είναι η κύρια βιβλιοθήκη που χρησιμοποιείται στην ανάλυσή μας. Έχει επίσης βιβλιοθήκες για preprocessing δεδομένων. Η κλάση Imputer κάνει πρεδικτιες αλυες ψηφιατριών στα μισσινγ αλυες, σύμφωνα με κάποιο κριτήριο. Το default κριτήριο είναι ο μέσος όρος των τιμών, πράγμα που χρησιμοποιείται εδώ. Επομένως με τις ακόλουθες εντολές δίνουμε τιμές στα missing values και ταυτόχρονα τα φέρνουμε σε μορφή πίνακα ώστε να μπορούν να διαβαστούν από αλγορίθμους που θα χρησιμοποιηθούν.

```

1 from sklearn.preprocessing import Imputer
2 imp = Imputer(missing_values='NaN', strategy='mean', axis=0, verbose=0)
3 imp.fit(df)
4 train_imp = imp.transform(df)
```

### 6.2.3 Train and Validation Sets

Επόμενο βήμα είναι να χωρίσουμε τα δεδομένα που έχουμε σε δύο σετ: μάθησης (training) και επικύρωσης (validation). Στο πρώτο οι αλγόριθμοι θα πάρουν τα δεδομένα και την αντίστοιχη έξοδο και θα δημιουργήσουν patterns που ουσιαστικά θα είναι τα δέντρα εκμάθησης, και στο δεύτερο θα κάνουν προβλέψεις με βάση το μοντέλο που θα έχουν δημιουργήσει. Το παραπάνω γίνεται εύκολα με την βοήθεια της συνάρτησης train\_test\_split της sklearn.

```

1 from sklearn.cross_validation import train_test_split as ts
2 dataset = train_imp[:,1:]
3 smp = train_imp[:,0]
4 dataset_train, dataset_test, smp_train, smp_test= ts(dataset,smp, test_size=0.33)
```

Στο παραπάνω παράδειγμα έχουμε δημιουργήσει δύο πίνακες, όπου ο πρώτος περιέχει μόνο το SMP και ο δεύτερος τα χαρακτηριστικά και ύστερα δημιουργούμε δύο σετ από το καθένα. Με την παράμετρο random state μπορούμε να δημιουργήσουμε αρχεία που θα λαμβάνουν τις μεταβλητές τους με τυχαίο τρόπο, όπως και έγινε στην ανάλυση. Επίσης το δείγμα για ανάλυση είναι 67% ενώ το αντίστοιχο για επικύρωση 33% ώστε να πετύχουμε καλύτερο fit του μοντέλου.



# Κεφάλαιο 7

## Ανάλυση δεδομένων και αλγορίθμων

### 7.1 Εισαγωγή

Στο κεφάλαιο αυτό θα μελετήσουμε την συμπεριφορά των αλγορίθμων σε ολόχληρη την χρονοσειρά με τυχαία training και test sets. Αυτό γίνεται ώστε να κατανοήσουμε την λειτουργία των αλγορίθμων και την υπολογιστική τους ικανότητα στις βέλτιστες συνθήκες. Λόγω των τυχαίων training και validation sets, η χρονοσειρά που δημιουργείται είναι ομαλή, και μπορεί να βοηθήσει στις προβλέψεις. Με αυτό τον τρόπο περιμένουμε καλύτερα σκορ από ότι σε πραγματικές συνθήκες, αλλά την ίδια στιγμή μπορούμε να μελετήσουμε τις σημαντικότητες των μεταβλητών σε ολόχληρο το δείγμα. Επομένως μετά το τέλος του κεφαλαίου θα γνωρίζουμε σε ένα πολύ καλό βαθμό τις δυνατότητες του κάθε μοντέλου όπως και το πόσο σημαντική είναι η κάθε μεταβλητή στον καθορισμό της εξόδου.

### 7.2 Multicollinearity

Σε αυτό το σημείο οφείλουμε να αναφέρουμε πως οι σημαντικότητες των μεταβλητών δεν μπορούν να προσεγγίσουν εύκολα την πραγματικότητα και αυτό οφείλεται σε οικονομικούς και πολιτικούς λόγους, στην κατάσταση της χώρας κατά την περίοδο αναφοράς όπως επίσης και στην αντίστοιχη νομοθεσία. Επίσης οι μεταβλητές μπορεί να παρουσιάζουν υψηλή σημαντικότητα λόγω του multicollinearity, δηλαδή να επηρεάζονται από κάποιον άλλο παράγοντα (όπως από αυτούς που αναφέραμε πριν) ή ακόμη και από άλλες μεταβλητές του ίδιου δείγματος. Στην βιβλιογραφία collinearity μεταξύ δύο predictors μπορεί να συμβεί όταν υπάρχει μία σχέση μεταξύ τους, και αν αυτή η σχέση είναι γραφική, δηλαδή της μορφής:  $X_{2i} = \lambda_0 + \lambda_1 X_{1i}$  τότε λέμε πως οι μεταβλητές είναι perfectly collinear. Αν τώρα κάποια μεταβλητή εμφανίζει εξάρτηση από πολλές διαφορετικές μεταβλητές, τότε έχουμε multicollinearity. Έχουμε perfect

multicollinearity αν η εξάρτηση είναι γραμμική, δηλαδή της μορφής:

$$\lambda_0 + \lambda_1 X_{1i} + \lambda_2 X_{2i} + \cdots + \lambda_k X_{ki} = 0 \quad (7.1)$$

Για να έχουμε μία εποπτική εικόνα της σημαντικότητας του κάθε predictor θα δημιουργήσουμε κάποια bar charts. Αυτό όμως γίνεται ώστε να κατανοήσουμε τα μοντέλα και την δύναμη των μεταβλητών σε αυτά. Κατά την ανάλυση της χρονοσειράς στο κεφάλαιο 8 θα κάνουμε partial dependence plots τα οποία είναι πιο ισχυρά και δείχνουν τον πραγματικό βαθμό της σχέσης κάθε μεταβλητής με το τελικό μας αποτέλεσμα.

### 7.3 Σφάλματα

Στο μοντέλο έχουμε regression διότι η έξοδός μας παίρνει τιμές συνεχείς επομένως θα χρησιμοποιήσουμε τις αντίστοιχες βιβλιοθήκες. Τα βήματα που ακολουθούνται για όλους τους αλγορίθμους είναι:

1. Δημιουργία του μοντέλου. Στην συγκεκριμένη περίπτωση θα χρησιμοποιήσουμε εργαλεία και βιβλιοθήκες της scikit-learn
2. Εισαγωγή του μοντέλου στο training set
3. Προβλέψεις στο validation set
4. Εξαγωγή συμπερασμάτων

Για την ανάλυση σφάλματος θα χρησιμοποιήσουμε τα παραχώτω κριτήρια.

$$\text{explained\_variance}(y, \hat{y}) = 1 - \frac{\text{Var}\{y - \hat{y}\}}{\text{Var}\{y\}} \quad (7.2)$$

$$MAE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} |y_i - \hat{y}_i| \quad (7.3)$$

$$MSE(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} (y_i - \hat{y}_i)^2 \quad (7.4)$$

$$R^2(y, \bar{y}) = 1 - \frac{\sum_{i=0}^{n_{samples}-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{n_{samples}-1} (y_i - \bar{y}_i)^2}, \bar{y} = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} y_i \quad (7.5)$$

## 7.4 Random Forests

### 7.4.1 Εκτίμηση σφάλματος

Η ανάλυση θα ξεκινήσει από τον αλγόριθμο Random Forests.

```

1 from sklearn.ensemble import RandomForestRegressor
2 rf = RandomForestRegressor(n_estimators=1000,oob_score=True)
3 rf.fit(dataset_train, smp_train)
4 rf.get_params()
5 y1 = rf.predict(dataset_test)
```

Το πακέτο RandomForestRegressor έχει ενσωματωμένες κάποιες μεθόδους οι οποίες μας δείχνουν την ακρίβεια του συστήματός μας όπως επίσης και εκτίμηση για το oob error.

```

1 print("Mean accuracy,training set = %f"%(rf.score(dataset_train, smp_train)))
2 print("Mean accuracy,validation set = %f"%(rf.score(dataset_test, smp_test)))
3 print 'OOB score: %.2f\n' % rf.oob_score_
```

Ως έξοδο λαμβάνουμε:

Mean accuracy,training set = 0.966240  
 Mean accuracy,validation set = 0.739425  
 OOB score: 0.75

Παρατηρούμε πως ο αλγόριθμος έχει ένα πολύ καλό σκορ, το οποίο δείχνει πως πράγματι μπορεί να υποθέσει σωστά την έξοδό μας, δηλαδή στην περίπτωσή μας το SMP με βάση τα χαρακτηριστικά που του δίνουμε. Αυτό φαίνεται από το mean accuracy,validation set όπως και από το oob score, τα οποία μας λένε πως μπορούμε με ακρίβεια κοντά στο 75% να υποθέσουμε σωστά την έξοδο. Η αντίστοιχη ακρίβεια στο σετ εκμάθησης είναι πολύ υψηλότερη, κοντά στο 97%.

Υπολογίζουμε τιμές των σφαλμάτων.

```

1 from sklearn import metrics as skm
2 print("Explained variance = %f" %(skm.explained_variance_score(smp_test, y1)))
3 print("Mean absolute error = %f" %(skm.mean_absolute_error(smp_test, y1)))
4 print("Mean squared error = %f" %(skm.mean_squared_error(smp_test, y1)))
5 print("R2_score = %f" %(skm.r2_score(smp_test, y1)))
```

Έξοδος:

Explained variance = 0.739508  
 Mean absolute error = 5.456595  
 Mean squared error = 58.660423  
 R2\_score = 0.739425

Έχουμε παρόμοια χαρακτηριστικά για explained variance και  $R^2$  score ενώ το mean absolute error μας δείχνει πόσο κοντά είναι οι τιμές που προβλέψθηκαν στις πραγματικές. Το μέσο σφάλμα είναι γύρω στις 5.5 μονάδες από την πραγματική τιμή, το οποίο σημαίνει πως το μοντέλο μπορεί να προβλέψει σε πολύ καλό ποσοστό την τελική τιμή. Το mean squared error μετρά τον μέσο όρο των τετραγώνων των σφαλμάτων και λειτουργεί ως 'τυπική απόκλιση' των σφαλμάτων.

#### 7.4.2 Cross Validation

Η μέτρηση του cross validation είναι σημαντική διότι υπάρχει πάντα ο κίνδυνος για overfitting των δεδομένων μας, ακόμη και με διαφορετικά training και test sets, διότι μπορεί οι παράμετροι του μοντέλου να ρυθμιστούν με τρόπο τέτοιο ώστε ο εκτιμητής να λειτουργεί καλύτερα από ότι στην πραγματικότητα με τυχαία νέα δεδομένα. Η γνώση για το τεστ μπορεί να διαφρεύσει στο μοντέλο και τελικά τα αποτελέσματα που παράγουμε να μην είναι τα πραγματικά. Η λύση έρχεται με το να θεωρήσουμε ένα άλλο σύνολο του μοντέλου μας ως σύνολο επικύρωσης, να τρέξουμε τα αποτελέσματα του συνόλου εκπαίδευσης εκεί και ύστερα να τρέξουμε το μοντέλο στο test set. Για παράδειγμα με το 5-fold cross validation δημιουργούμε 5 τυχαία υποσύνολα στο σετ εκπαίδευσης, τρέχουμε τον αλγόριθμο σε 4 από αυτά και το επικυρώνουμε στο τελευταίο, και αυτό γίνεται 5 φορές.

```

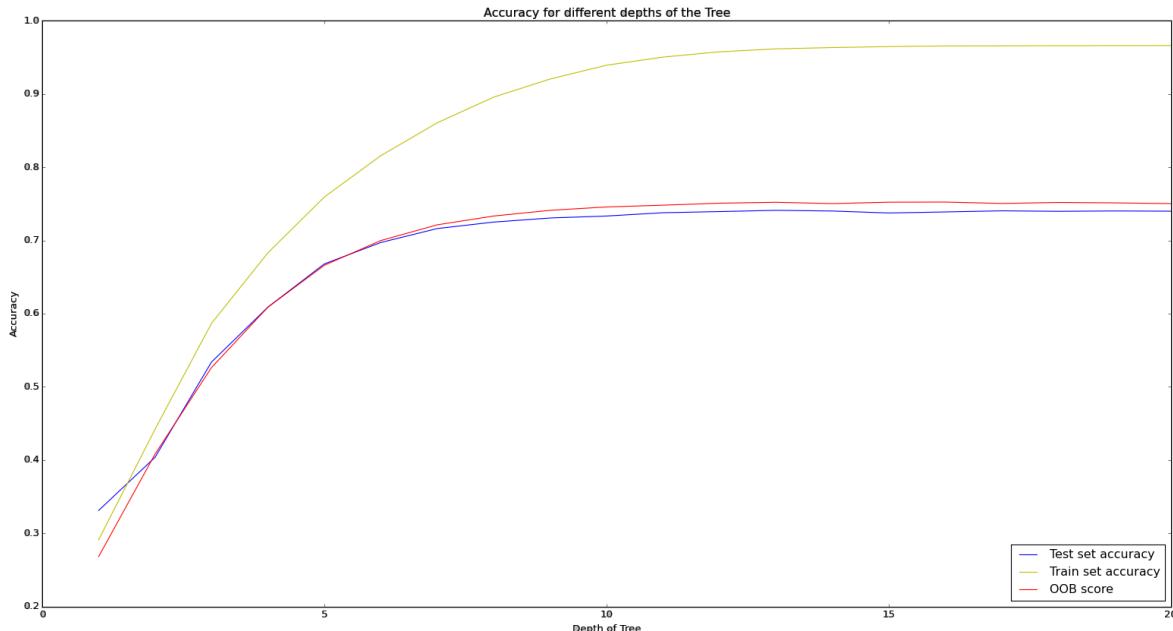
1 from sklearn import cross_validation
2 scores = cross_validation.cross_val_score(rf, dataset_train, smp_train, cv=5)
3 print 'Cross Validation Scores:'
4 print scores
5 print "Cross Validation Accuracy: %.2f (+/- %.2f)" % (scores.mean(), scores.std())

```

Έξοδος για 5-fold Cross Validation:  
Cross Validation Scores:  
[0.75343825 0.70602166 0.76413408 0.74446292 0.72598341]  
Cross Validation Accuracy: 0.74 (+/- 0.01)

Έξοδος για 10-fold Cross Validation:  
Cross Validation Scores:  
[0.786871 0.66804677 0.68724677 0.76931317 0.7993393 0.78431832  
0.71387939 0.76512308 0.75068604 0.68754537]  
Cross Validation Accuracy: 0.74 (+/- 0.02)

Παρατηρούμε πως η απόδοση του αλγορίθμου είναι παρόμοια με την απόδοση που είδαμε στο test set το οποίο σημαίνει πως το σύστημά μας είναι συνεπές και ότι μπορεί να ανταπεξέλθει με παρόμοιο τρόπο και να παράγει το ίδιο καλά συμπεράσματα σε πραγματικές συνθήκες με νέα δεδομένα.

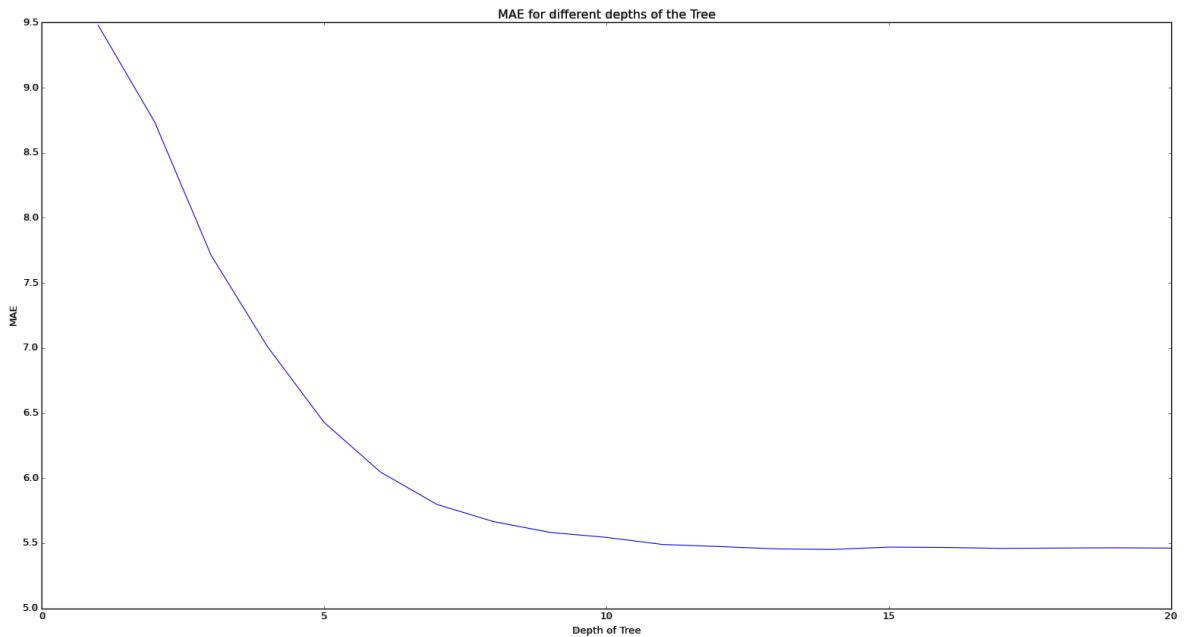


Σχήμα 7.1: Random Forest: Accuracy

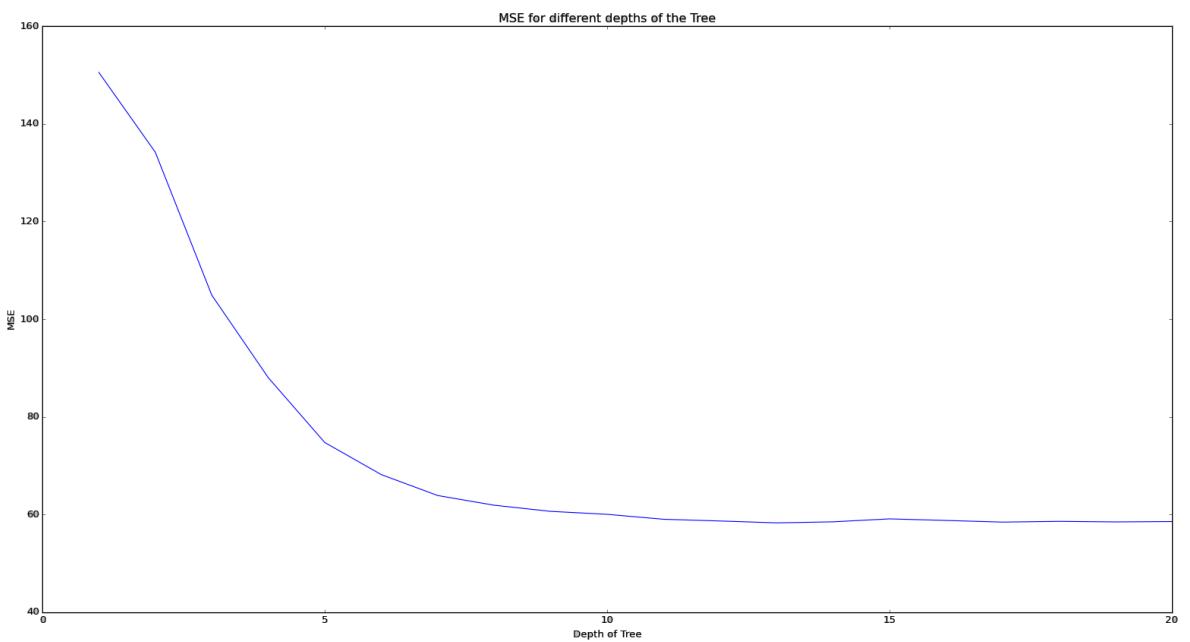
Για να δούμε την επίδοση του αλγορίθμου ανάλογα με το βάθος που επιλέγουμε να έχει το δέντρο μας, κάνουμε 20 iterations για βάθη 1 έως 20 (βάθος 1 σημαίνει πως το δέντρο έχει μόνο δύο φύλλα τα οποία δημιουργούνται από διαίρεση ενός χαρακτηριστικού σε κάποιο σημείο). Τα αποτελέσματα φαίνονται στα σχήματα 7.1 έως 7.3.

Για μικρά βάθη βλέπουμε μία μεγάλη αύξηση της ακρίβειας, από 30% για βάθος 1 σε 70% για βάθος 6. Από κει και πέρα η βελτίωση της επίδοσης αυξάνει πιο αργά, και μετά από βάθος 12-13 δεν υπάρχει καμία αύξηση, μπορεί μάλιστα να παρατηρηθεί και πολύ μικρή μείωση. Αυτό οφείλεται στο γεγονός πως μετά από ένα σημείο έχουμε overfitting των δεδομένων και δεν μπορούμε να δούμε βελτίωση.

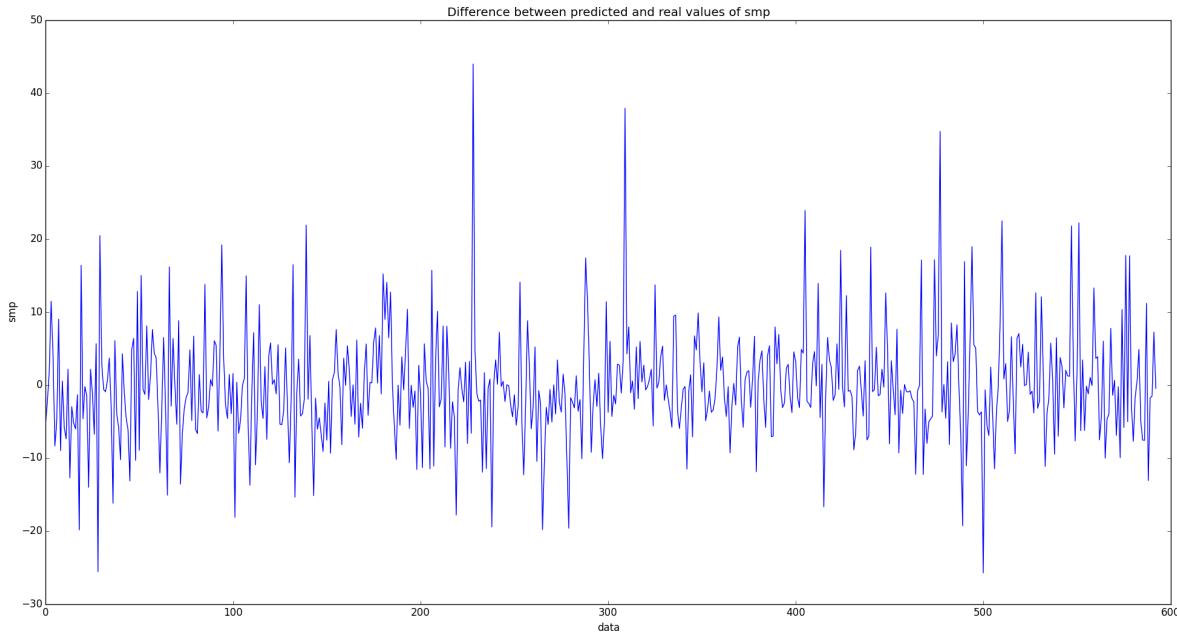
Με τον ίδιο τρόπο το MAE όπως και το MSE μειώνουν πολύ γρήγορα όσο αυξάνεται το βάθος, αλλά μετά από ένα σημείο έχουμε πλέον βελτιστοποιήσει την απόδοση και δεν μπορούμε να πετύχουμε κάτι καλύτερο.



Σχήμα 7.2: Random Forest: Mean absolute error



Σχήμα 7.3: Random Forest: Mean squared error



Σχήμα 7.4: Random Forest: Διαφορά real-expected τιμών του SMP

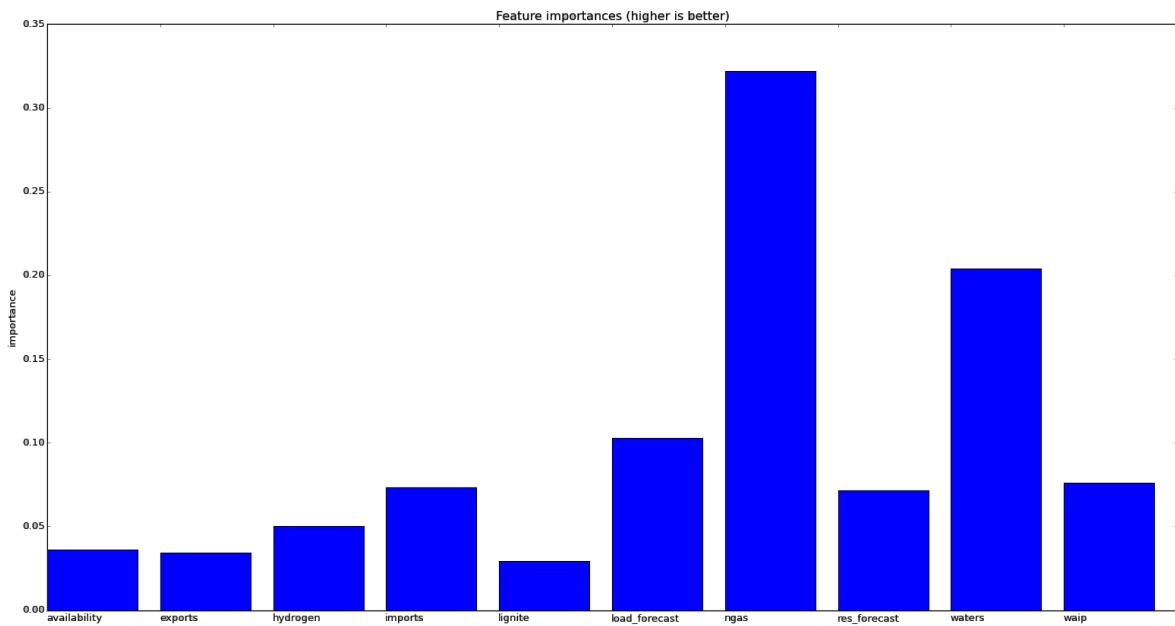
#### 7.4.3 Πραγματικές - εκτιμηθείσες τιμές SMP

Το σχήμα 7.4 παρουσιάζει την διαφορά των πραγματικών τιμών σε σχέση με τις τιμές που προβλέψθηκαν για το SMP. Ο άξονας x παρουσιάζει τα δεδομένα και όχι τον χρόνο, διότι έγινε τυχαία επιλογή των δεδομένων κατά την κατασκευή των δέντρων.

```

1 import matplotlib.pyplot as plt
2 plt.figure()
3 plt.plot(smp_test-y1)
4 plt.xlabel('data')
5 plt.ylabel('smp')
6 plt.title('Difference between predicted and real values of smp')
7 plt.show()
```

Οι προβλέψεις είναι πολύ καλές και ακολουθούν τις πραγματικές τιμές. Αυτό φαίνεται από το γεγονός πως γενικά το σφάλμα είναι μικρό, κατά μέσο όρο 5 μονάδες από το πραγματικό. Παρατηρούμε σε κάποια σημεία μεγάλες διαφορές, οι οποίες προέρχονται από το γεγονός ότι σε αυτά τα σημεία το SMP παίρνει ακραίες τιμές και το μοντέλο δεν ακολουθεί παίρνοντας πιο συγκρατημένες προβλέψεις πράγμα θετικό.



Σχήμα 7.5: Random Forest: Feature Importances

#### 7.4.4 Feature Importances

Θα δούμε ποιές μεταβλητές παιζουν τον πιο σημαντικό ρόλο στον καθορισμό του αποτελέσματος του αλγορίθμου.

```

1 fi = enumerate(rf.feature_importances_)
2 df2 = df.drop('smp', 1)
3 cols = df2.columns
4 print [(value,cols[i]) for (i,value) in fi]
5 features = mlab.csv2rec(path + 'rf_features.csv', delimiter=',')
6 plt.figure()
7 plt.bar(np.arange(len(features)),features['value'],label='values')
8 plt.xticks(range(len(features)),features['feature'], ha = 'left')
9 plt.ylabel('importance')
10 plt.title('Feature importances (higher is better)')
11 plt.show()
```

Feature	Importance
availability	0.0361405
exports	0.0344914
hydrogen	0.0502056
imports	0.0733868
lignite	0.0292468
load_forecast	0.1026408
ngas	0.3220485
res_forecast	0.0717241
waters	0.2040629
waip	0.0760525

Πίνακας 7.1: Random Forest: Σημαντικότητες χαρακτηριστικών

Το πιο σημαντικό χαρακτηριστικό στον καθορισμό της εξόδου είναι το ngas και έπειτα έρχεται το waters. Αυτά τα δύο χαρακτηριστικά καθορίζουν την τελική έξοδο σε ποσοστό μεγαλύτερο του 50%. Αρκετά σημαντικό ρόλο φαίνεται να έχει και το load forecast το οποίο καθορίζει σε ποσοστό 10% την τιμή του SMP. Οι υπόλοιπες μεταβλητές βοηθούν στον τελικό καθορισμό της εξόδου, αλλά σε μικρότερο ποσοστό.

## 7.5 CART (Regression Trees)

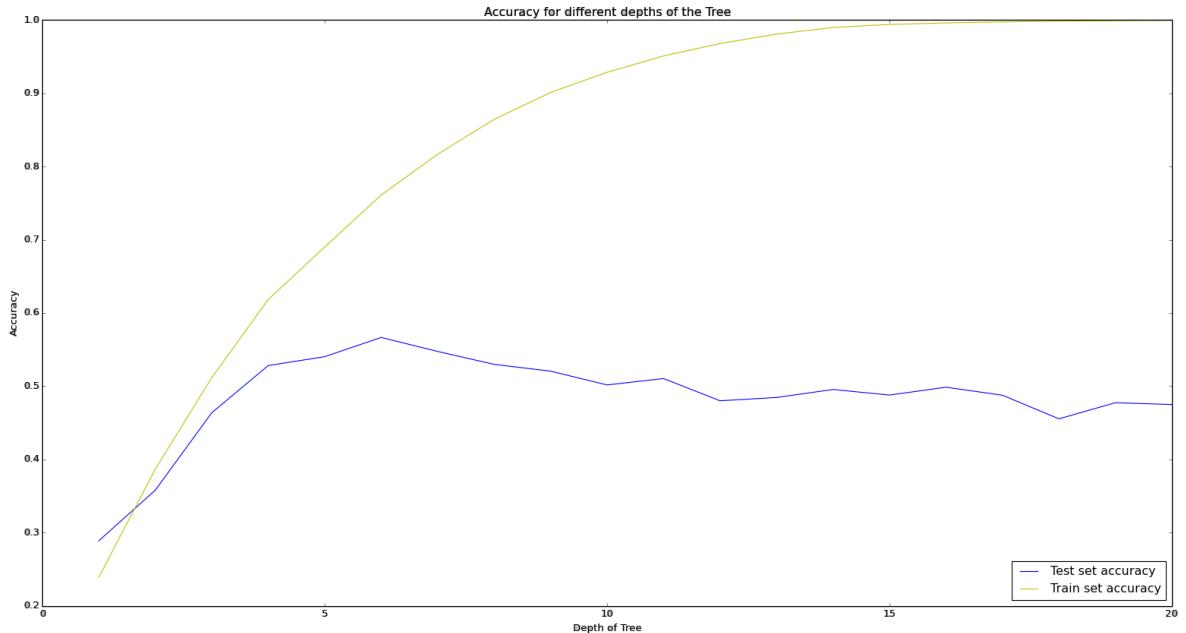
### 7.5.1 Εκτίμηση σφάλματος

Τρέχουμε τον αλγόριθμο για regression trees για βάθη από 1 έως 20 ώστε να δούμε ποιό είναι το βέλτιστο βάθος για το οποίο να πάρουμε αποτελέσματα.

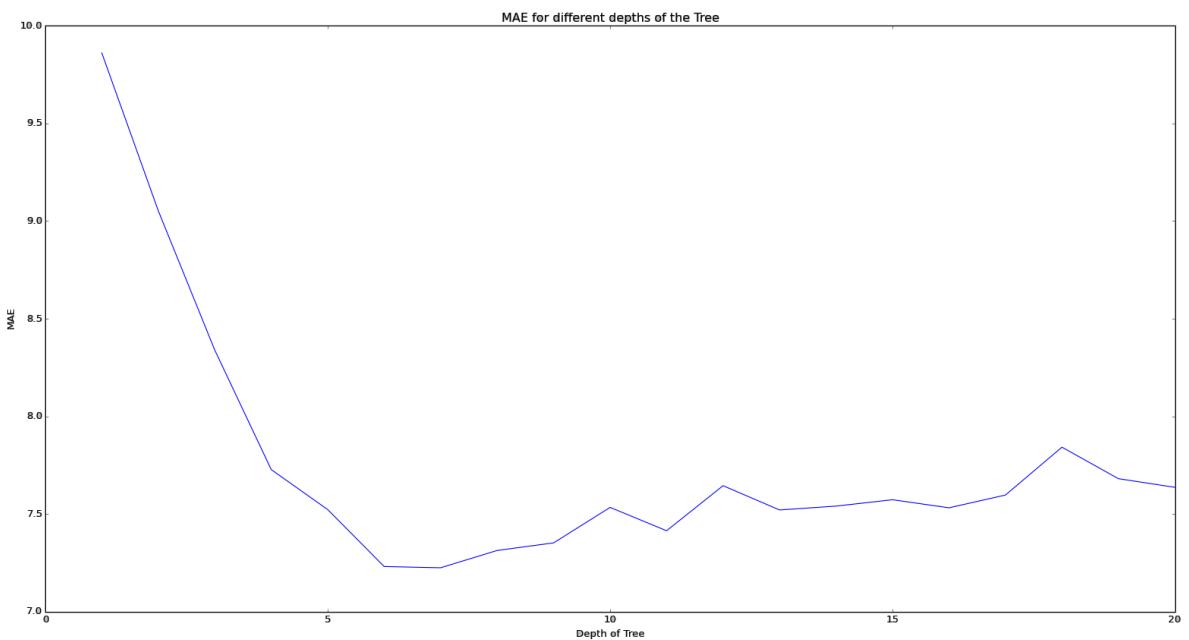
```

1 from sklearn import tree
2 from sklearn import metrics as skm
3 train_acc = test_acc = exp_var_score = mae = mse = np.zeros(20)
4 for i in range (1,21):
5     tr = tree.DecisionTreeRegressor(max_depth=i)
6     tr.fit(dataset_train, smp_train)
7     tr.get_params()
8     y1 = tr.predict(dataset_test)
9     train_acc[i-1] = tr.score(dataset_train, smp_train)
10    test_acc[i-1] = tr.score(dataset_test, smp_test)
11    exp_var_score[i-1] = skm.explained_variance_score(smp_test, y1)
12    mae[i-1] = skm.mean_absolute_error(smp_test, y1)
13    mse[i-1] = skm.mean_squared_error(smp_test, y1)

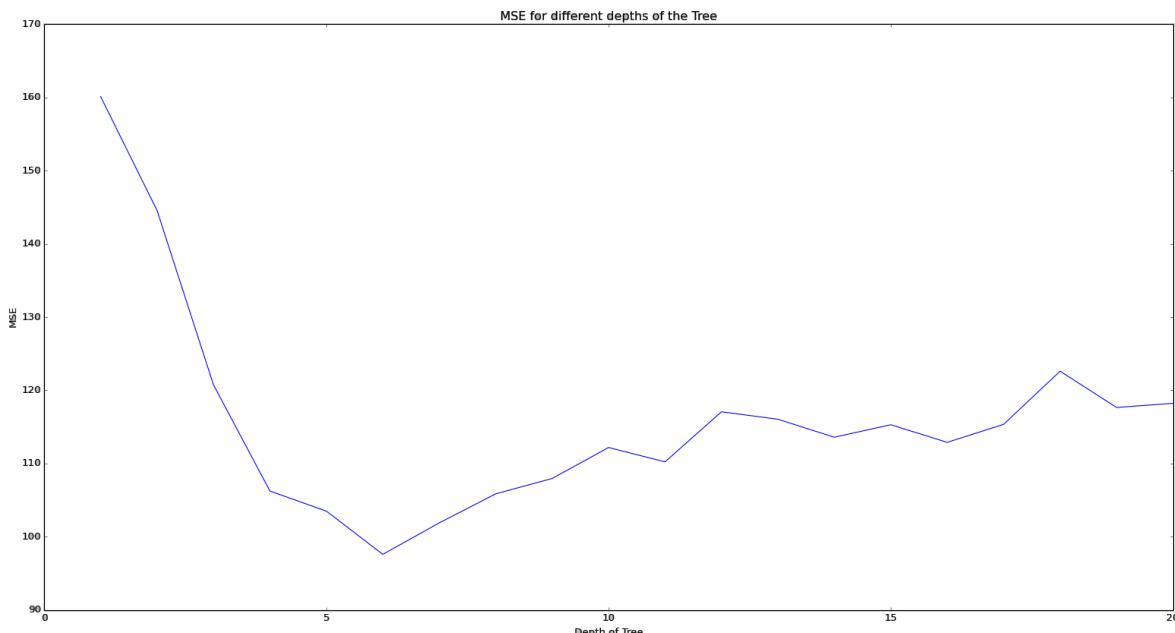
```



Σχήμα 7.6: Tree Regressor: Accuracy



Σχήμα 7.7: Tree Regressor: Mean absolute error



Σχήμα 7.8: Tree Regressor: Mean squared error

Σε αντίθεση με πριν, ο αλγόριθμος δεν έχει τόσο καλές επιδόσεις, και φτάνει μέγιστη ακρίβεια στο 54,75% για δέντρο βάθους 6. Όσο αυξάνεται το βάθος η ακρίβεια δεν αυξάνει, μάλιστα μειώνει το οποίο σημαίνει πως έχουμε overfitting στο μοντέλο. Για βάθος δέντρου ίσο με `max_depth` το αποτέλεσμα της πρόβλεψης είναι λίγο κάτω από 50%. Το ίδιο δείχνουν και τα αποτελέσματα για MAE και MSE, επομένως η ανάλυση θα γίνει για δέντρο βάθους 6.

### 7.5.2 Cross Validation

Όπως και στα random forest, χρησιμοποιούμε cross validation ώστε να επικυρώσουμε τα αποτελέσματά μας. Τα αποτελέσματα είναι όπως τα προβλεπόμενα.

5-fold Cross Validation Scores:

[0.57645212 0.55334927 0.64557411 0.6136979 0.54650035]

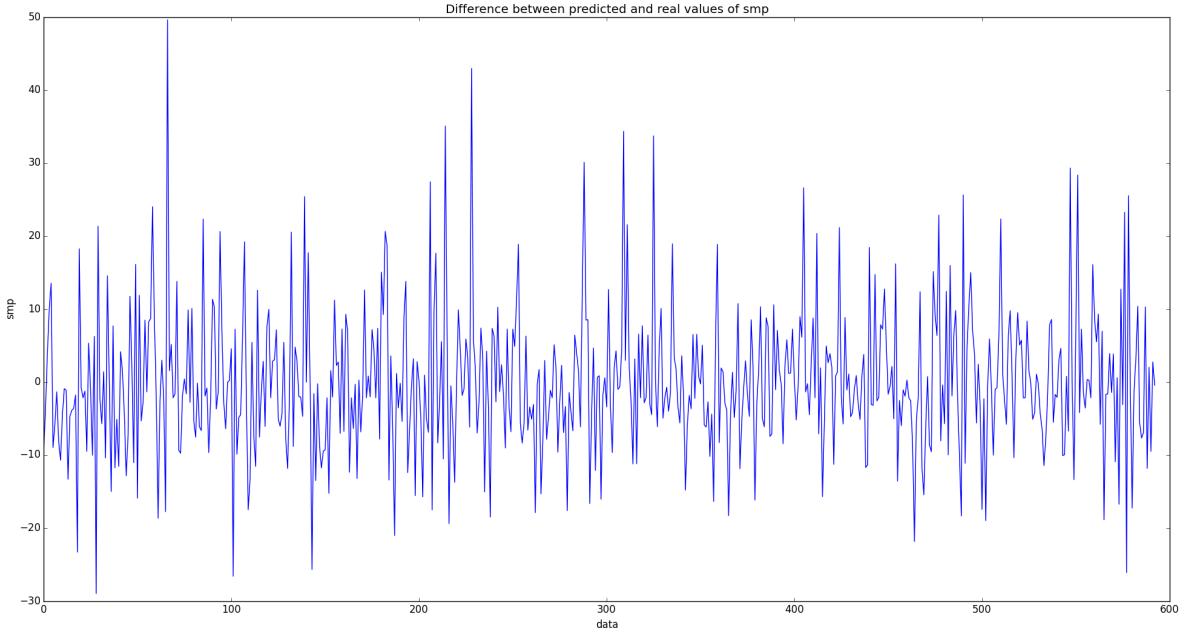
Cross Validation Accuracy: 0.59 (+/- 0.02)

10-fold Cross Validation Scores:

[0.64172542 0.42875275 0.53303435 0.59282483 0.67429731 0.59046097

0.64643966 0.52166911; 0.60511882 0.3935244]

Cross Validation Accuracy: 0.56 (+/- 0.04)



Σχήμα 7.9: Tree Regressor: Διαφορά real-expected τιμών του SMP

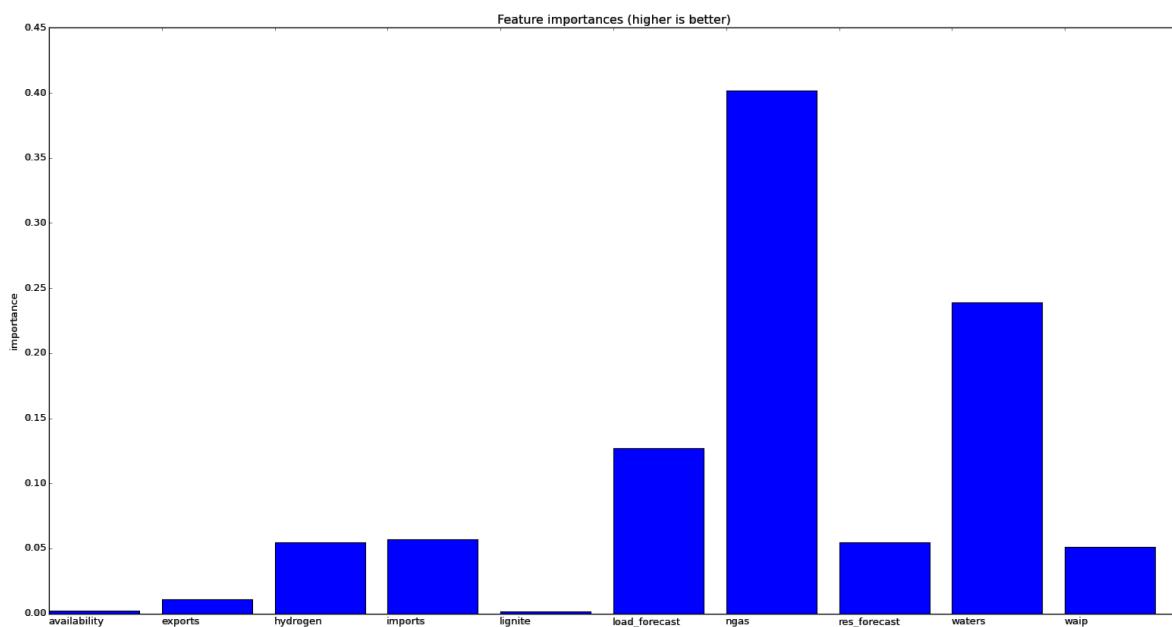
### 7.5.3 Πραγματικές - εκτιμηθείσες τιμές SMP

Στο σχήμα 7.9 βλέπουμε την διαφορά real και expected τιμών για το SMP σύμφωνα με το τον αλγόριθμο για regression trees.

Είναι εμφανής η διαφορά στην ακρίβεια σε σχέση με τον αλγόριθμο random forest. Έχουμε μεγαλύτερη απόκλιση από τις πραγματικές τιμές σε ολόκληρο το δείγμα. Επίσης φαίνεται πως το μοντέλο προσπαθεί να προβλέψει ακραίες τιμές, αρκετές φορές μάλιστα λάθος.

### 7.5.4 Feature Importances

Ο αλγόριθμος CART θεωρεί σημαντικό το χαρακτηριστικό ngas σε ποσοστό 40%, περισσότερο από ότι στον αλγόριθμο random forest. Το ίδιο συμβαίνει και με τα χαρακτηριστικά waters και load\_forecast τα οποία έχουν μεγαλύτερη σημαντικότητα σε σχέση με αυτή που έχουν στα random forest. Τα υπόλοιπα χαρακτηριστικά φαίνεται πως παίζουν μικρότερο ρόλο, μάλιστα τα lignite και availability έχουν μηδαμινό ποσοστό στο τελικό αποτέλεσμα.



Σχήμα 7.10: Tree Regressor: Feature Importances

Feature	Importance
availability	0.0023202
exports	0.0111215
hydrogen	0.0548785
imports	0.0570344
lignite	0.0013275
load_forecast	0.1269267
ngas	0.4016380
res_forecast	0.0545563
waters	0.2391213
waip	0.0510756

Πίνακας 7.2: Tree Regressor: Σημαντικότητες χαρακτηριστικών

Τα παραπάνω οφείλονται στο γεγονός ότι από την μία πλευρά το δέντρο μας έχει μικρό βάθος, επομένως είναι εύλογο τα ποσοστά των χαρακτηριστικών να είναι υψηλότερα, από την άλλη δεν μπορεί να υπολογίσει με τόσο καλή ακρίβεια όπως γίνεται στα random forest ποιά χαρακτηριστικά χρειάζονται και με τί ποσοστό για την βελτιστοποίηση των προβλέψεων.

Καταλήγουμε στο συμπέρασμα πως για πολύ καλό καθορισμό της εξόδου χρειάζονται όλα τα χαρακτηριστικά σε κάποιο βάθυ, αλλά κύριο ρόλο φαίνεται να παίζει το ngas. Επίσης είναι εμφανές ότι ο αλγόριθμος random forest βελτιστοποιεί την ιδέα που υπάρχει στον αλγόριθμο CART και παράγει έξοδο με πολύ μεγαλύτερη ακρίβεια, πράγμα που συμφωνεί με όσα ειπώθηκαν στην θεωρητική ανάλυση.

### 7.5.5 Τελικό Δέντρο

Δημιουργούμε το δέντρο και το κάνουμε εξαγωγή ως αρχείο .dot και ύστερα το ανοίγουμε μέσω του Graphviz.

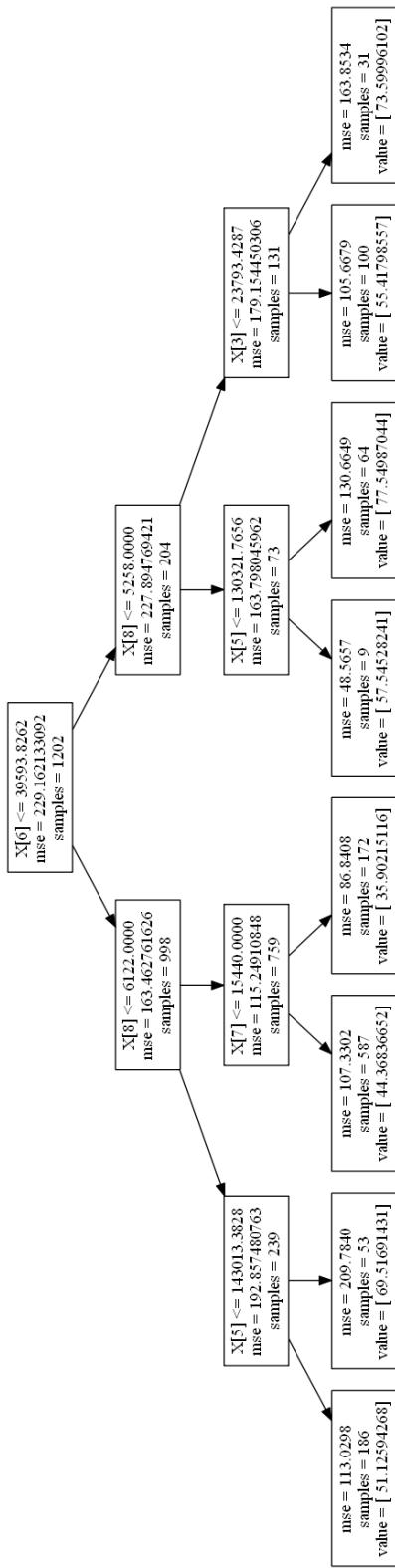
```
1 sklearn.externals.six import StringIO
2 with open(path+"tree_3.dot", 'w') as f:
3     f = tree.export_graphviz(tr, out_file=f)
```

Επειδή το δέντρο ήταν πολύ μεγάλο για να παρουσιαστεί εδώ, κλαδέψαμε τους κόμβους για βάθος 5 και 6 και οπότε τελικά παρουσιάζουμε το αποτέλεσμα για δέντρο βάθους 4 το οποίο δίνει mean accuracy score = 0.463656. Οι μεταβλητές που υπάρχουν στο δέντρο είναι οι εξής:

- $X[3]$  : imports
- $X[5]$  : load\_forecast
- $X[6]$  : ngas
- $X[7]$  : res\_forecast
- $X[8]$  : waters

Σε κάθε κλάδο του δέντρου έχουμε 3 χαρακτηριστικά: το όνομα της μεταβλητής, το MSE και τον αριθμό των δεδομένων που υπάρχουν στον συγκεκριμένο κλάδο. Εφόσον το training set περιέχει το 66.6% του συνόλου μας, έχουμε αρχικά 1202 δείγματα. Ανάλογα με το αν ισχύει η συνθήκη που περιγράφεται στον κάθε κόμβο ή όχι πάμε στο αντίστοιχο φύλλο, στο αριστερό αν η συνθήκη είναι αληθής και στο δεξιό αν δεν είναι (οι συνθήκες είναι της μορφής  $X \leq a$  το οποίο ισοδυναμεί με  $X > a$  με ναι στο δεξιό φύλλο και όχι στον αριστερό). Το δέντρο είναι δυαδικό.

Η μεταβλητή ngas παίζοντας τον κύριο ρόλο κάνει το πρώτο partition, ακολουθούμενο από την μεταβλητή waters. Το load\_forecast ως τρίτο σημαντικότερο κάνει 2 partitions στο επίπεδο 3. Παρατηρούμε πως σε κάθε βήμα το δέντρο δεν χωρίζεται σε δύο ισομεγέθη κομμάτια, αλλά περισσότερο σε μία αναλογία κοντά στο 3:1. Οι μεταβλητές εμφανίζονται σε σειρά σημαντικότητας σε κάθε επίπεδο, και μάλιστα (παρότι δεν φαίνεται στο σχήμα) σε μεγαλύτερα βάθη οι πιο σημαντικές μεταβλητές εμφανίζονται πολλαπλά (για παράδειγμα το ngas εμφανίζεται ξανά για βάθος 4 ενώ το lignite εμφανίζεται αρκετά πιο μετά). Σημαντικό επίσης το γεγονός ότι χρησιμοποιούμε μόνο 5 από τις 10 μεταβλητές για ακρίβεια 46% ενώ σε βάθος 6 με ακρίβεια 56% χρησιμοποιούνται όλες οι μεταβλητές το οποίο επιβεβαιώνει πως όλα τα χαρακτηριστικά του μοντέλου χρησιμεύουν στον καθορισμό της εξόδου.



**Σχήμα 7.11:** Δέντρο αλγορίθμου Tree Regressor - βάθος 4

Forward Pass									
iter	parent	var	knot	mse	terms	gcv	rsq	grsq	
0	-	-	-	229.162133	1	229.544	0.000	0.000	
1	0	6	647	155.881987	3	158.243	0.320	0.311	
2	0	8	815	121.801029	5	125.321	0.468	0.454	
3	0	3	215	104.633204	7	109.125	0.543	0.525	
4	0	2	251	94.637817	9	100.056	0.587	0.564	
5	0	9	515	87.405562	11	93.688	0.619	0.592	
6	0	7	-1	80.830007	12	87.240	0.647	0.620	
7	0	5	-1	75.674123	13	82.243	0.670	0.642	
8	0	0	1175	67.833522	15	74.755	0.704	0.674	
9	0	2	1075	66.539130	17	74.364	0.710	0.676	
10	0	4	1163	65.474324	19	74.215	0.714	0.677	
11	0	8	1111	64.428805	21	74.076	0.719	0.677	
12	0	1	23	63.556436	23	74.127	0.723	0.677	
13	0	1	1075	62.681360	25	74.169	0.726	0.677	
14	0	7	731	61.936551	27	74.361	0.730	0.676	
15	0	9	491	61.329937	29	74.718	0.732	0.674	

Stopping Condition 0: Reached maximum number of terms

Σχήμα 7.12: MARS: Forward Pass

## 7.6 MARS

Για το μοντέλο MARS χρησιμοποιήθηκε η βιβλιοθήκη pyearth που δημιούργησε ο Jason Rudy η οποία εφαρμόζει τον αλγόριθμο που αναπτύχθηκε από τον Jerome Friedman και παρουσιάστηκε αναλυτικά στο κεφάλαιο 3 (10).

```

1 from pyearth import Earth
2 model = Earth()
3 model.fit(dataset_train,smp_train)
4 print model.trace()
5 print model.summary()
```

### 7.6.1 Forward Pass

Στο σχήμα 7.12 παρουσιάζεται η πρώτη φάση του αλγορίθμου, όπου δημιουργείται το μοντέλο προσθέτοντας συναρτήσεις βάσης. Η αρχική μας συνάρτηση είναι η  $h_0(X) = 1$ . Οι μεταβλητές του συστήματός μας ονομάζονται var, terms είναι ο αριθμός των συναρτήσεων βάσης που έχουμε εισάγει μέχρι το συγκεκριμένο iteration του αλγορίθμου, gcv είναι το κριτήριο Generalized Cross Validation, rsq είναι το  $R^2$  του μοντέλου και grsq είναι μία εκτίμηση της προγνωστικής ικανότητας του μοντέλου.

Pruning Pass						
iter	bf	terms	mse	gcv	rsq	grsq
0	-	29	61.33	74.718	0.732	0.674
1	28	28	61.33	74.172	0.732	0.677
2	26	27	61.33	73.632	0.732	0.679
3	20	26	61.33	73.098	0.732	0.682
4	15	25	61.33	72.570	0.732	0.684
5	23	24	61.33	72.047	0.732	0.686
6	11	23	61.33	71.536	0.732	0.688
7	17	22	61.38	71.075	0.732	0.690
8	14	21	61.47	70.676	0.732	0.692
9	2	20	61.85	70.602	0.738	0.692
10	22	19	62.11	70.400	0.729	0.693
11	24	18	62.71	70.575	0.726	0.693
12	21	17	63.35	70.797	0.724	0.692
13	27	16	64.20	71.244	0.720	0.690
14	9	15	64.84	71.459	0.717	0.689
15	18	14	65.68	71.880	0.713	0.687
16	5	13	66.18	71.927	0.711	0.687
17	16	12	67.89	72.415	0.707	0.685
18	19	11	68.46	73.385	0.701	0.688
19	7	10	69.96	74.477	0.695	0.676
20	6	9	72.31	76.452	0.684	0.667
21	8	8	76.32	80.140	0.667	0.651
22	3	7	79.85	83.282	0.652	0.637
23	1	6	87.96	91.113	0.616	0.603
24	13	5	100.73	103.642	0.560	0.548
25	4	4	122.39	125.083	0.466	0.455
26	12	3	155.83	157.376	0.323	0.314
27	10	2	210.29	212.047	0.082	0.076
28	25	1	229.16	229.544	0.000	0.000

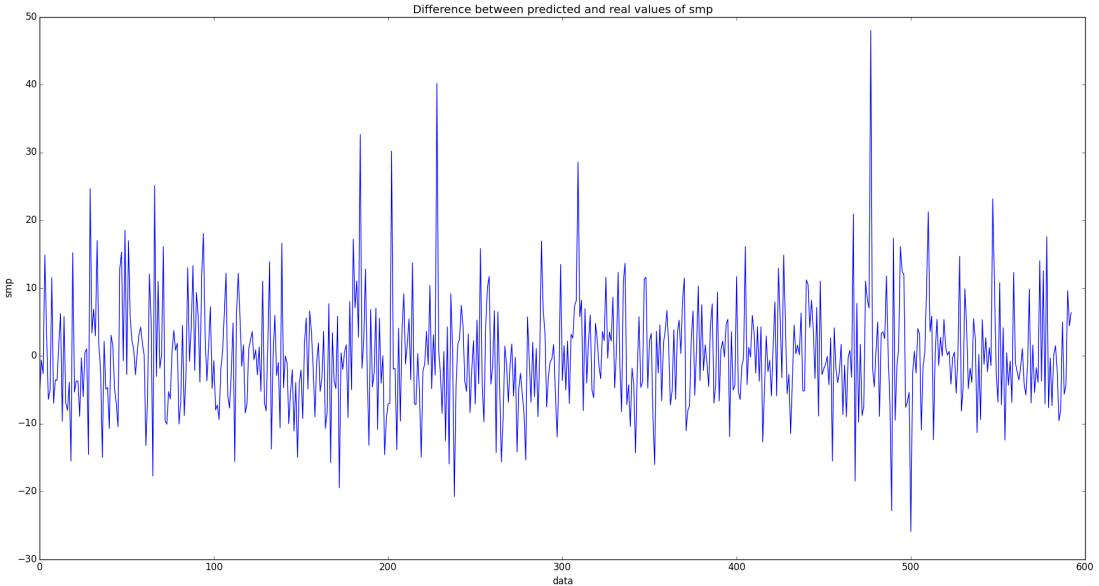
Selected iteration: 10

Σχήμα 7.13: MARS: Pruning Pass

Όπως και στους προηγούμενους αλγορίθμους που εξετάστηκαν, η πιο σημαντική μεταβλητή είναι το `ngas` ακολουθόμενο από τα `waters`. Οι δύο μεταβλητές μαζί μας δίνουν προγνωστική ικανότητα στο 45,4%. Ενδιαφέρον παρουσιάζει το γεγονός ότι οι επόμενες τρεις μεταβλητές για τις οποίες δημιουργούνται συναρτήσεις βάσεις είναι τα `imports`, `hydrogen` και `waip`. Παρατηρούμε πως όσο αυξάνουμε την πολυπλοκότητα του μοντέλου τότο μικρότερο είναι το MSE ενώ οι μεταβλητές `gcv`, `rsq` και `grsq` αυξάνουν με μειούμενο ρυθμό. Κατά την πρώτη φάση του αλγορίθμου παίρνουμε τιμές `gcv` = 74.71,  $R^2$  = 0.732 και `grsq` = 0.674, τιμές πολύ κοντά στις αντίστοιχες που είχαμε βρει με τον αλγόριθμο Random Forest.

## 7.6.2 Pruning Pass

Η δεύτερη φάση του αλγορίθμου έρχεται με την αντίστροφη διαδικασία κλαδέματος. Σκοπός είναι να κλαδευτούν συναρτήσεις που συμβάλλουν λιγότερο στο πόσο καλό fit θα κάνει το μοντέλο. Ο αλγόριθμος θα διαλέξει το iteration που ελαχιστοποιεί το `gcv`, το οποίο συμβαίνει στην δέκατη επανάληψη. Στην ίδια επανάληψη έχουμε και το μέγιστο `grsq`, το οποίο και υέλουμε να μεγιστοποιήσουμε. Επομένως το τελικό μας μοντέλο θα έχει 10 λιγότερους όρους σε σχέση με το αρχικό, και θα είναι το βέλτιστο μοντέλο που μπορεί να παράγει ο αλγόριθμος.



Σχήμα 7.14: MARS: Πραγματικές και εκτιμηθείσες τιμές του SMP

### 7.6.3 Τελικό μοντέλο

Στο σχήμα 7.13 βλέπουμε το τελικό μοντέλο του αλγορίθμου. Η πρώτη στήλη περιέχει όλες τις συναρτήσεις βάσεις που δημιουργήθηκαν κατά το forward pass, η δεύτερη στήλη μας λέει ποιές από αυτές τις συναρτήσεις κλαδεύτηκαν ενώ η τρίτη στήλη μας παρέχει πληροφορίες σχετικά με τον συντελεστή της κάθε συνάρτησης. Παρατηρούμε πως μία εκ των δύο συναρτήσεων βάσης της μεταβλητής `ngas` κλαδεύτηκε. Οι υπόλοιπες αρχικές συναρτήσεις έμειναν ως είχαν κατά το δεύτερο πέρασμα. Τα τελικά αποτελέσματα φαίνονται στην τελευταία γραμμή. Ο αλγόριθμός μας μπορεί να υπολογίσει με ποσοστό χοντά στο 70% την έξοδο, μία πολύ καλή επίδοση.

### 7.6.4 Πραγματικές - εκτιμηθείσες τιμές SMP

Στο σχήμα 7.14 έχουμε το διάγραμμα με τη διαφορά πραγματικών και expected τιμών του SMP. Το μοντέλο παράγει εμφανώς καλύτερα αποτελέσματα σε σχέση με τα αντίστοιχα του μοντέλου CART και είναι αντίστοιχο σε επίδοση με το μοντέλο Random Forest. Μάλιστα υπάρχουν σημεία που οι προβλέψεις του μοντέλου είναι καλύτερες από το Random Forest με μικρότερο σφάλμα και κάποια σημεία που το μοντέλο παρουσιάζει πολύ ακραίες τιμές. Σε γενικές γραμμές όμως μπορούμε να πούμε πως τα μοντέλα MARS και Random Forest δίνουν πολύ καλές προβλέψεις, ενώ το μοντέλο CART όχι τόσο ικανοποιητικές.

Earth Model			
Basis Function	Pruned	Coefficient	
(Intercept)	No	-10.1827	
h(x6-28691.9)	No	0.000294766	
h(28691.9-x6)	Yes	None	
h(x8-7108)	No	-0.00837911	
h(7108-x8)	No	0.00922639	
h(x3-22183.4)	No	-0.000766391	
h(22183.4-x3)	No	-0.000728432	
h(x2-17944.8)	No	0.000515783	
h(17944.8-x2)	No	-0.000747982	
h(x9-62.35)	No	28.3639	
h(62.35-x9)	No	-0.487655	
x7	Yes	None	
x5	No	0.000377777	
h(x0-8101)	No	-0.00662202	
h(8101-x0)	Yes	None	
h(x2-6321.04)	Yes	None	
h(6321.04-x2)	No	-0.00315083	
h(x4-49768.8)	Yes	None	
h(49768.8-x4)	No	0.000679306	
h(x8-3105)	No	0.00746316	
h(3105-x8)	Yes	None	
h(x1-16838.2)	No	0.00495245	
h(16838.2-x1)	Yes	None	
h(x1-3464.8)	Yes	None	
h(3464.8-x1)	No	-0.0013023	
h(x7-4820)	No	-0.000846537	
h(4820-x7)	Yes	None	
h(x9-62.5)	No	-28.3288	
h(62.5-x9)	Yes	None	

MSE: 62.1088, GCV: 70.4003, RSQ: 0.7290, GRSQ: 0.6933

Σχήμα 7.15: MARS: Earth Model



## Κεφάλαιο 8

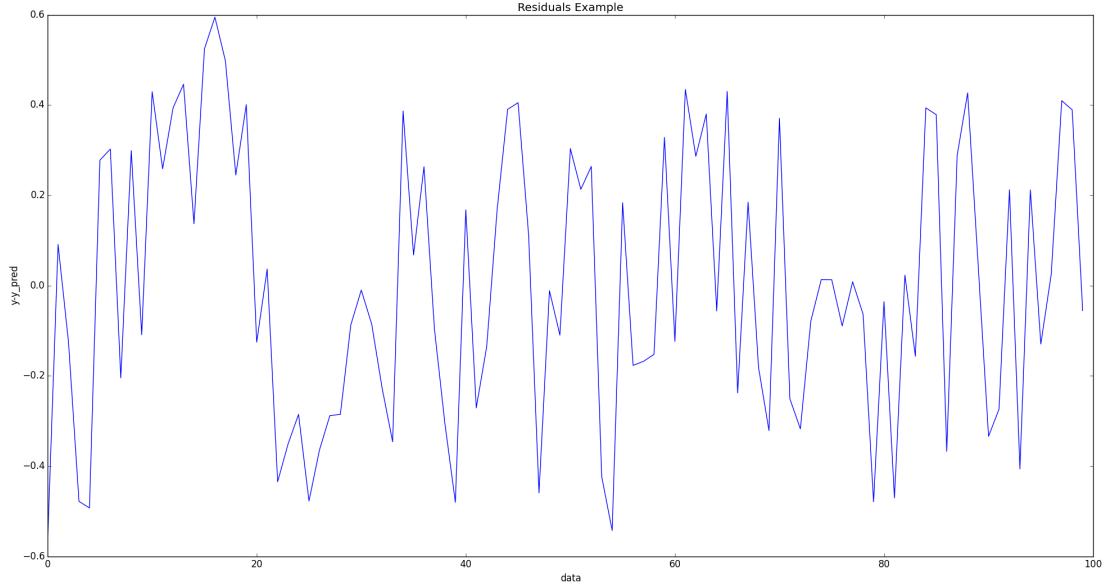
### Ανάλυση Χρονοσειράς

#### 8.1 Εισαγωγή

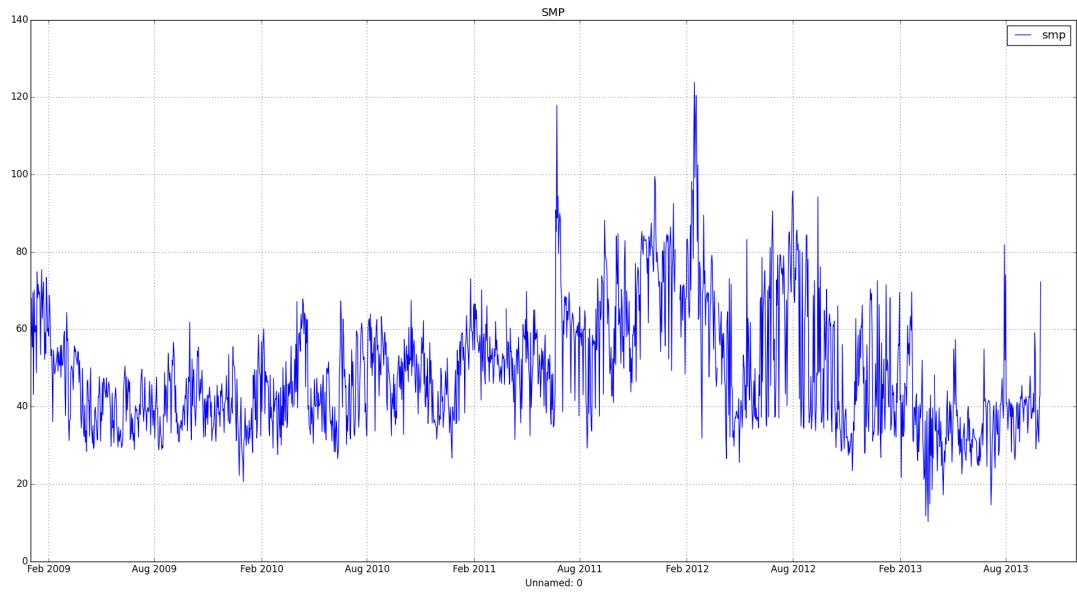
Στην προηγούμενη ανάλυση είδαμε πώς συμπεριφέρονται οι αλγόριθμοι πάνω σε ολόκληρη την χρονοσειρά επιλέγοντας τυχαία δείγματα δεδομένων για training και validation sets και αξιολογήσαμε τους αλγορίθμους και την προγνωστική τους ικανότητα. Σε αυτό το κεφάλαιο θα ασχοληθούμε με predictions πάνω στον χρόνο χωρίς να γνωρίζουμε ολόκληρο το σύνολο των δεδομένων. Η διαδικασία αυτή είναι αρκετά δυσκολότερη διότι μπορεί στο test set να περιλαμβάνονται ακραίες τιμές λόγω οικονομικοπολιτικής κατάστασης της περιόδου που εξετάζεται η οποία να διαφέρει αρκετά σε σχέση με το training set, οπότε και ο καθορισμός της εξόδου να μην είναι τόσο εύκολος. Επίσης, εφόσον δεν γνωρίζουμε τί trend μπορεί να έχουν τα τελικά δεδομένα, είναι κρίσιμης σημασίας και ο καθροισμός του training set, ώστε μέσω αυτού να παράγουμε τα βέλτιστα αποτελέσματα. Τέλος, επειδή έχουμε χρονοσειρά ίσως χρειαστεί να λάβουμε υπ' όψη τις προηγούμενες τιμές των μεταβλητών ώστε να ελαχιστοποιήσουμε τα σφάλματα εξόδου και να βελτιώσουμε την προγνωστική ικανότητα του μοντέλου μας.

#### 8.2 Residuals και lagged values

Στο παράδειγμα που περιγράφεται έχουμε δύο τυχαίες χρονοσειρές που έχουν εξάρτηση από την προηγούμενη τιμή τους και για τις οποίες κάνουμε ένα απλό linear regression και δημιουργούμε μία συνάρτηση της μορφής  $y = \alpha x + \beta$ . Υστερα μέσω της συνάρτησης παράγουμε τις αντίστοιχες προβλεπόμενες τιμές. Τα residuals είναι οι διαφορές που προκύπτουν από τις πραγματικές και τις προβλεπόμενες τιμές. Η διαφορά των residuals και των σφαλμάτων είναι στις πραγματικές τιμές που λαμβάνουμε: στην πρώτη περίπτωση αφαιρούμε την εκτιμηθείσα τιμή από την γνωστή πραγματική ενώ στην δεύτερη περίπτωση από την άγνωστη πραγματική.



Σχήμα 8.1: Residuals Example



Σχήμα 8.2: Χρονοσειρά SMP

Το σημαντικό γεγονός με τα residuals είναι πως θέλουμε στο σύστημα που δημιουργούμε τα residuals να τείνουν στο μηδέν. Μάλιστα στην ιδανική περίπτωση ο μέσος όρος αυτών είναι στο μηδέν. Στο παρόντα παραχάτω συναρτήσεις:

$$X(t) = X(t - 1) + \varepsilon(t), X(0) = 0, \varepsilon \sim N(0, 1) \quad (8.1)$$

$$Y(t) = 0.2 * Y(t - 1) + z(t), Y(0) = 0, z \sim N(0, 1) \quad (8.2)$$

Στο σχήμα 8.1 φαίνονται τα residuals του παραδείγματος. Παρατηρούμε πως οι τιμές των σφαλμάτων κυμαίνονται κοντά στο μηδέν. Μάλιστα ο μέσος όρος των residuals στο συγκεκριμένο παράδειγμα είναι 0.0028.

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 n=100
4 e = np.random.rand(n)
5 x = np.zeros(n)
6 for i in range (1,n):
7     x[i] = x[i-1] + e[i-1]
8
9 z = np.random.rand(n)
10 y = np.zeros(n)
11 for i in range (1,n):
12     y[i] = 0.2*y[i-1] + z[i-1]
13
14 reg = np.polyfit(x, y, 1)
15 y_pred = reg[0]*x + reg[1]
16 res = y-y_pred
17
18 m = 0
19 for i in range(1,n):
20     m = m + res[i-1]
21 ave_res = m/n
22
23 plt.figure()
24 plt.plot(res)
25 plt.xlabel('data')
26 plt.ylabel('y-y_pred')
27 plt.title('Residuals Example')
28 plt.show()
```

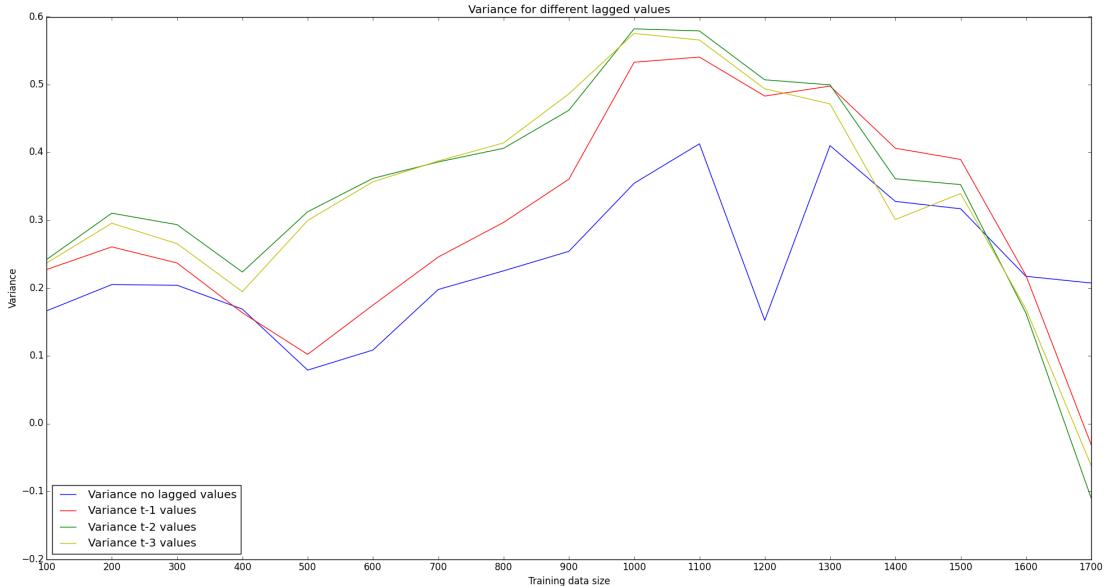
Με την προσθήκη lagged μεταβλητών, δηλαδή μεταβλητών που περιέχουν τις προηγούμενες χρονικές τιμές  $t - 1, t - 2, \dots$ , μπορούμε να κάνουμε το σύστημά μας πιο σταθερό και με residuals πιο κοντά στο μηδέν. Με αυτό τον τρόπο θα έχουμε και καλύτερη προβλεπτική ικανότητα. Αυτό δεν μπορεί να γίνει εφικτό όταν λαμβάνουμε τυχαίες μεταβλητές σε όλο το δείγμα για training και validation sets, αφού τότε δημιουργούμε διαφορετικές χρονοσειρές. Ανάλογα με το αν το δείγμα μας είναι ομοιογενές ή όχι, παράγουμε και τα αντίστοιχα αποτελέσματα. Όσο πιο ομοιογενές είναι το δείγμα, τόσο περισσότερο βοηθά η χρήση lagged μεταβλητών.

Στο σχήμα 8.2 παρουσιάζεται η χρονοσειρά του SMP από 1/1/09 έως 30/11/13. Το γράφημα δημιουργήθηκε ώστε να κάτανούμε σε ποιά σημεία τα δεδομένα μας είναι ομοιογενή ώστε να δούμε πού πρέπει να φτιάξουμε training και validation sets. Παρατηρούμε πως από τα μέσα του 2009 έως τις αρχές του 2011 η χρονοσειρά είναι ομαλή οπότε μπορούμε να υποθέσουμε πως ένας καλός αλγόριθμος μηχανικής μάθησης θα μπορούσε να παράγει ικανοποιητικές προβλέψεις. Από τα μέσα του 2011 και έπειτα όμως η χρονοσειρά δεν είναι ομαλή παρουσιάζοντας αυξομειώσεις λόγω της πολιτικής και οικονομικής κατάστασης που επικράτησε στην χώρα μας. Ειδικά για την περίοδο 2013 έχουμε πολύ μεγάλες διακυμάνσεις στο SMP και η πρόβλεψη θα είναι σαφώς δυσκολότερη.

### 8.3 Επιλογή lagged μεταβλητών

Αρχικά εφαρμόσαμε τον αλγόριθμο Random Forests στα δεδομένα μας για διαφορετικά lagged values. Συγκεκριμένα ο αλγόριθμος εφαρμόστηκε για δεδομένα χωρίς lagged μεταβλητές έως δεδομένα με μεταβλητές χρόνου  $t - 3$ . Για την δημιουργία των lagged μεταβλητών χρησιμοποιήσαμε την εντολή shift από την βιβλιοθήκη pandas ώστε να μετακινήσουμε τα δεδομένα ενός dataframe τις ίδιες που θέλουμε.

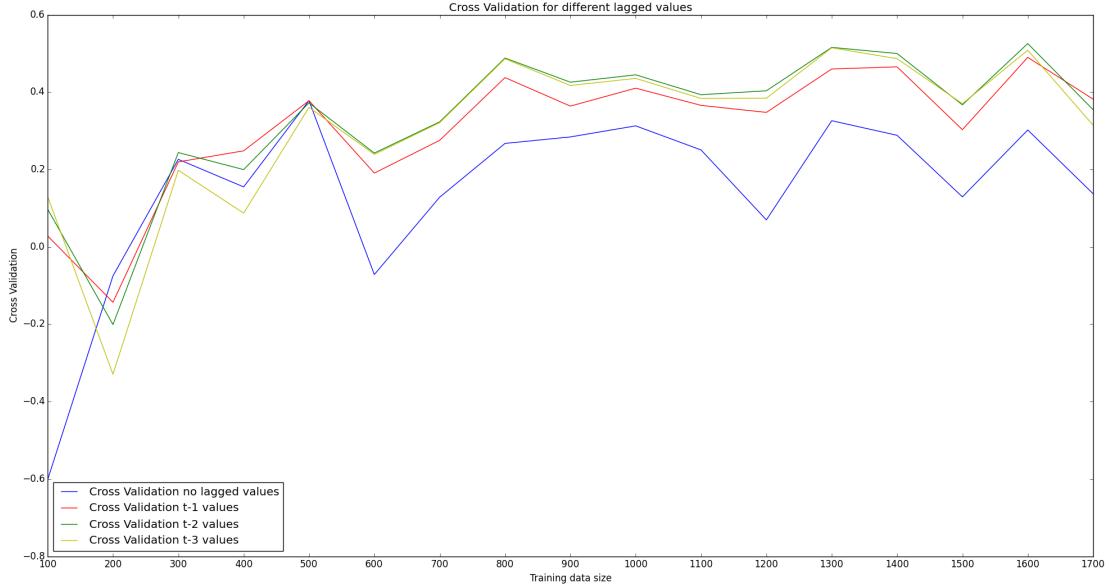
Ο αλγόριθμος εφαρμόστηκε για διαφορετικά train και test sets μέσα στην χρονοσειρά, ζεκινώντας από training sets των 100 μεταβλητών έως training sets των 1600 μεταβλητών. Τα αποτελέσματα φαίνονται στα γραφήματα 8.3 έως 8.6. Παρατηρούμε πως με την προσθήκη lagged μεταβλητών το σύστημα έκανε καλύτερες προβλέψεις σε σχέση με το αντίστοιχο χωρίς τις lagged μεταβλητές. Το γεγονός αυτό έχει να κάνει με την ανομοιογένεια της χρονοσειράς, η οποία επηρεάζει αρνητικά την ακρίβεια του μοντέλου και κάνει τις προβλέψεις δύσκολες. Η εισαγωγή μεταβλητών που περιέχουν πληροφορία της προηγούμενης χρονικής στιγμής βοηθά σε σημαντικό βαθμό το μοντέλο και την παραγωγή προβλέψεων. Η επιπλέον προσθήκη παλαιότερων μεταβλητών επηρεάζει λιγότερο θετικά, μάλιστα μετά από ένα σημείο επηρεάζει αρνητικά στην ακρίβεια του μοντέλου. Σύμφωνα με τα γραφήματα η βέλτιστη ακρίβεια λαμβάνεται με προσθήκη lagged μεταβλητών έως  $t - 2$ .



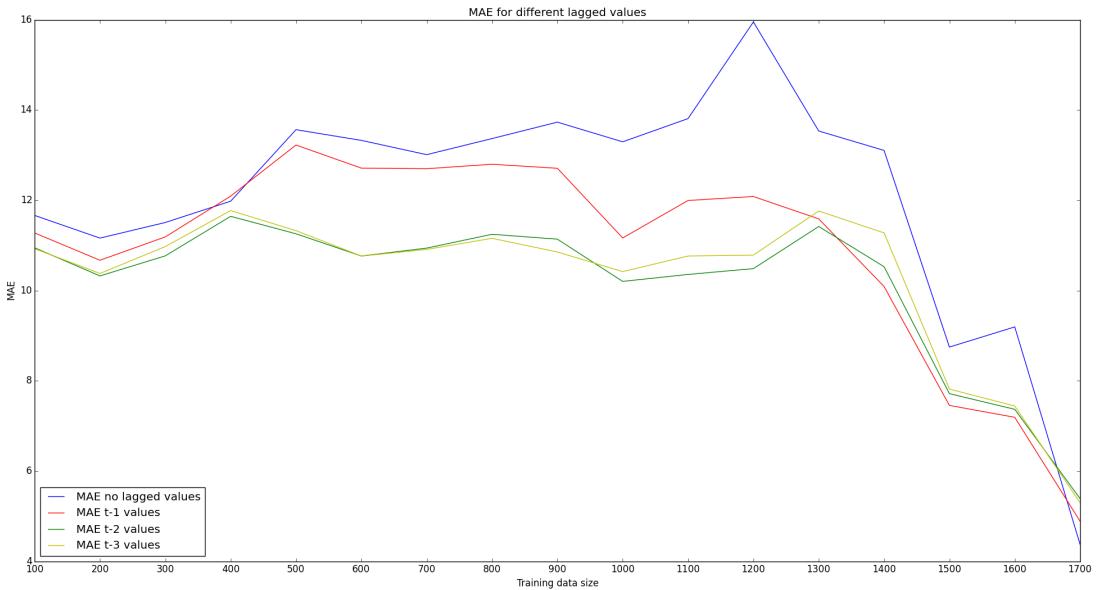
Σχήμα 8.3: Variance for different lagged values

Το πρώτο σημαντικό σημείο που παρατηρούμε είναι πως η ακρίβεια είναι αρκετά χαμηλότερη από αυτή που πετύχαμε στο προηγούμενο κεφάλαιο όπου χωρίσαμε την χρονοσειρά σε τυχαία training και validation sets, ακόμα και με την προσθήκη lagged μεταβλητών. Αυτό είναι αποτέλεσμα της ανομοιογένειας της χρονοσειράς που μελετάμε όπως και της ομοιογένειας των χρονοσειρών που δημιουργήσαμε στο προηγούμενο κεφάλαιο. Κατά την δημιουργία των προηγούμενων σετ μάθησης και επικύρωσης σημαντικό ρόλο έπαιζε η ελαχιστοποίηση των σφαλμάτων και η βελτιστοποίηση της επίδοσης των αλγορίθμων, οπότε και τα γραφήματα που δημιουργήθηκαν είχαν την αντίστοιχη δομή. Με αυτό τον τρόπο είδαμε ποιά χαρακτηριστικά είναι τα πιο σημαντικά μέσα στην χρονοσειρά όπως και την δύναμη του κάθε αλγορίθμου και την προβλεπτική του ικανότητα. Τώρα θέλουμε να βρούμε τον βέλτιστο τρόπο δημιουργίας του μοντέλου όπως και σε ποιά σημεία οι αλγόριθμοι παρουσιάζουν την καλύτερη συμπεριφορά σε σχέση με πραγματικές προβλέψεις.

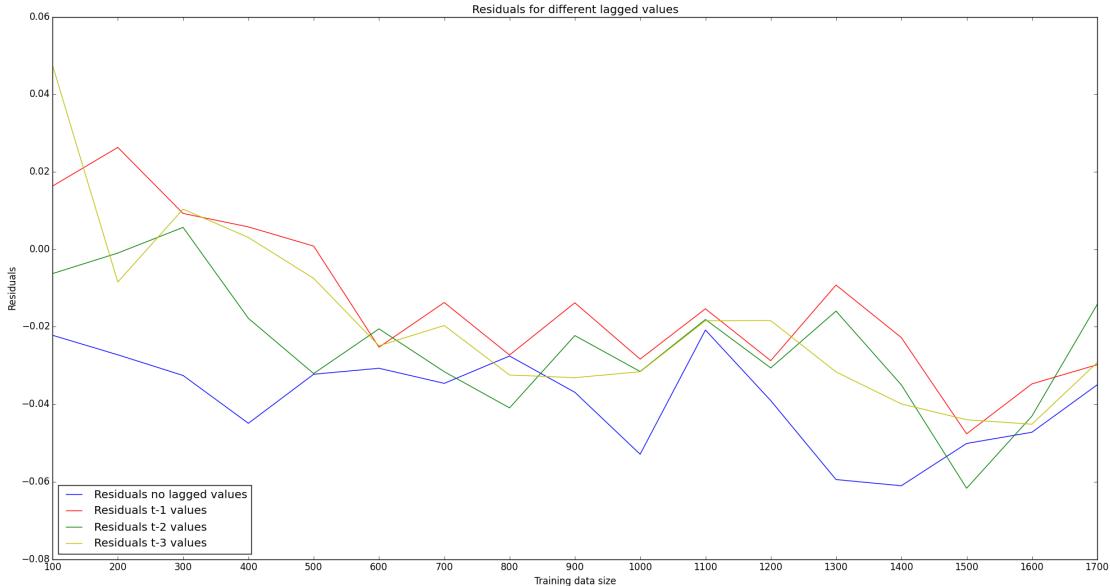
Στο σχήμα 8.4 έχουμε το explained Variance (ακρίβεια μοντέλου) για τα διαφορετικά lagged values που χρησιμοποιήθηκαν. Η καλύτερη ακρίβεια επιτυγχάνεται με training set 1000-1100 τιμών με lagged values, δηλαδή 60% του συνόλου και 40% validation set. Η ακρίβεια είναι κοντά στο 60%, μία πολύ καλή συμπεριφορά για τα ανομοιογενή δεδομένα μας. Επίσης παρατηρούμε πως για μετάλλιο training set, μικρό test set έχουμε χαμηλή ακρίβεια, και μάλιστα καλύτερη ακρίβεια με το μοντέλο χωρίς τα lagged values. Αυτό οφείλεται στο γεγονός ότι τα δεδομένα του SMP των τελευταίων 3 μηνών παρουσιάζουν πολύ μεγάλες αποκλίσεις μεταξύ τους.



Σχήμα 8.4: Cross Validation for different lagged values



Σχήμα 8.5: MAE for different lagged values



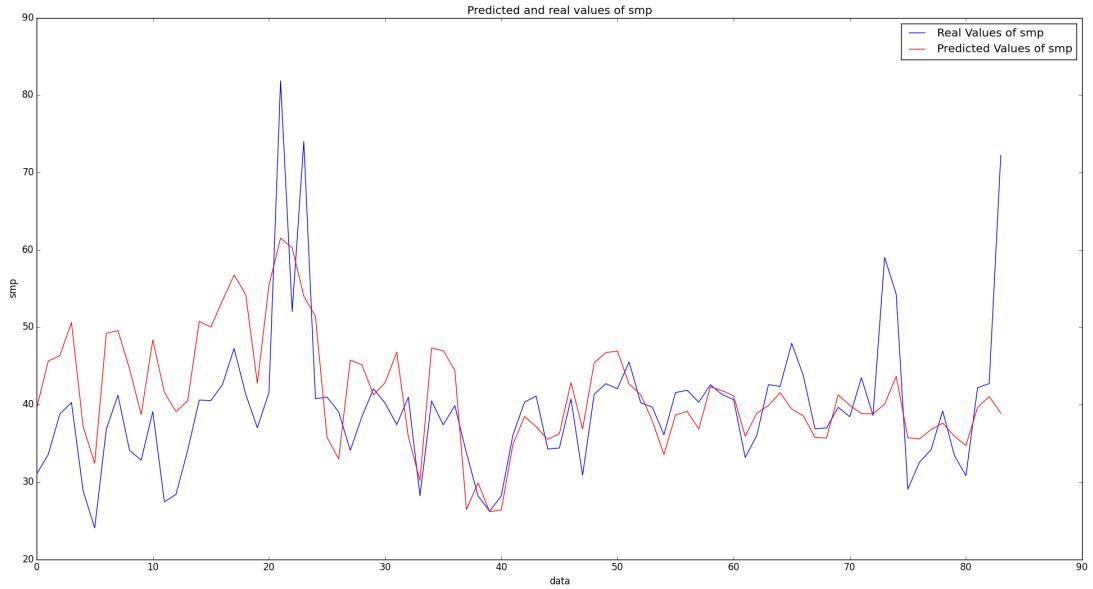
Σχήμα 8.6: Residuals for different lagged values

Στα σχήματα 8.4 και 8.5 βλέπουμε το cross validation και MAE. Το MAE όπως και το MSE είναι ελάχιστα για lagged values  $t - 2$  σύμφωνα και με αυτά που είπαμε πριν. Το cross validation είναι καλύτερο με lagged μεταβλητές, λόγω της βελτίωσης του μοντέλου και των residuals. Όσον αφορά τα residuals, έχουμε καλύτερες τιμές (χοντά στο μηδέν) για  $t-1$  και  $t-2$ .

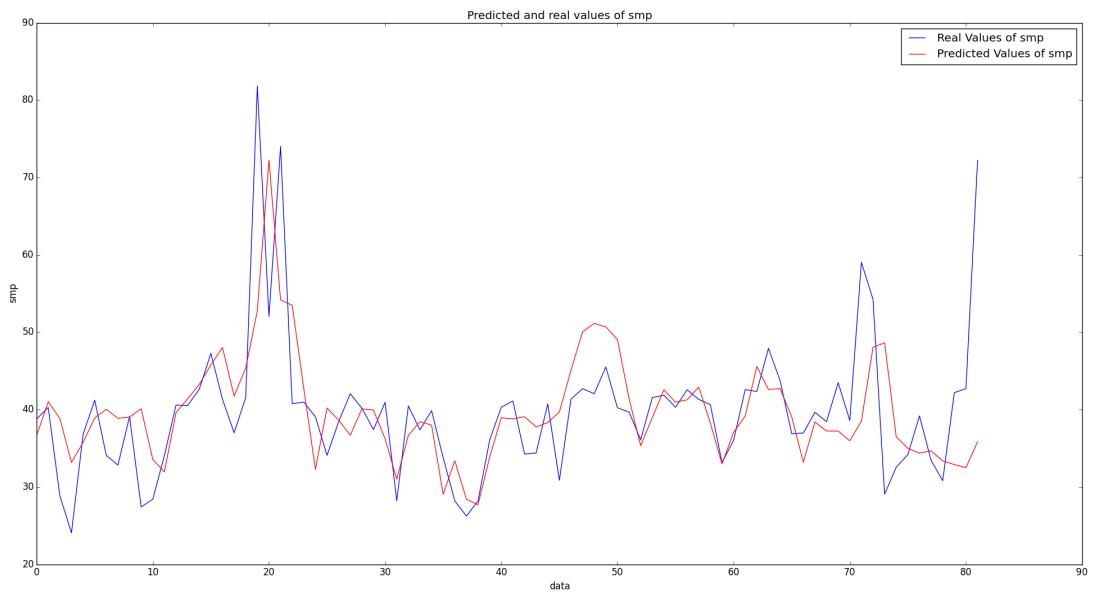
Όσον αφορά τα importances των μεταβλητών, ανάλογα με το μέγεθος του δείγματος είχαμε διαφορετικές σημαντικότητες, με το ngs να έχει μικρότερη σημαντικότητα σε μικρά δείγματα και να αυξάνει σημαντικότητα όσο είχαμε περισσότερα δεδομένα στο training set. Για τις προβλέψεις που είχαν καλό accuracy το ngs έχει σημαντικότητα στο 0.34 χωρίς lagged values και στο 0.11 με lagged μεταβλητές ενώ τότε η πιο σημαντική μεταβλητή ήταν η τιμή  $SMP(t - 1)$  με importance στο 0.45.

Τοτερα επιχειρήσαμε να προβλέψουμε ένα συγκεκριμένο test set το οποίο περιέχει τις τελευταίες 100 τιμές της χρονοσειράς, και τα αποτελέσματα ήταν αντίστοιχα με πριν. Οι μεταβλητές με lagged values για  $t - 2$  ήταν ελάχιστα πιο ακριβείς από  $t-1$  και μας έδωσαν τα καλύτερα αποτελέσματα. Σημαντικό το γεγονός ότι ο αλγόριθμος MARS πετυχαίνει και αυτός πολύ καλά αποτελέσματα.

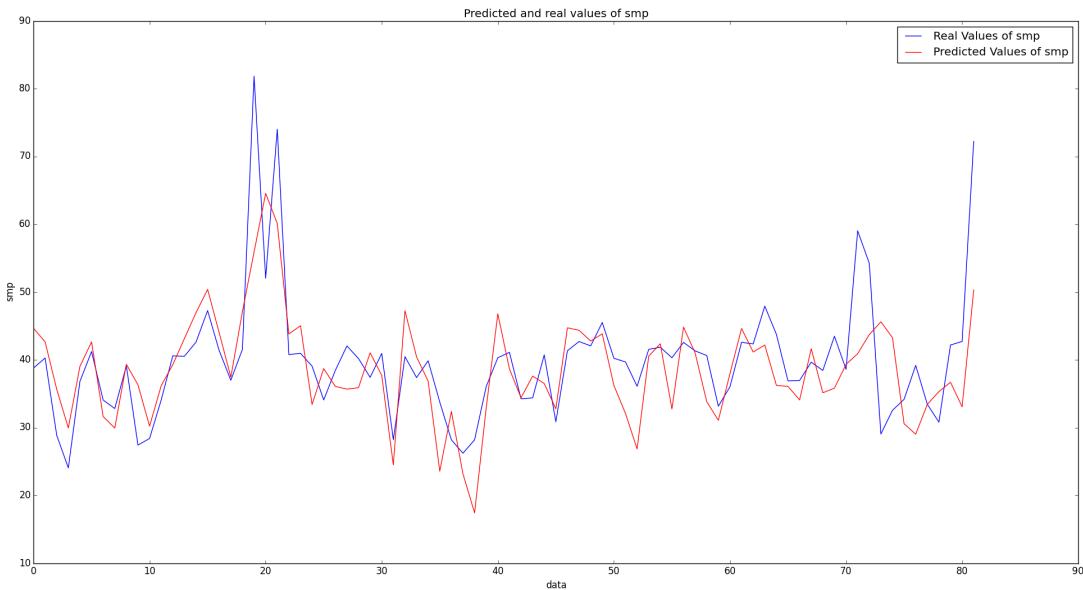
Στα σχήματα 8.7 έως 8.9 φαίνονται οι προβλέψεις για Random Forests χωρίς lagged μεταβλητές όπως και προβλέψεις με lagged  $t-2$  values με χρήση των αλγορίθμων Random Forests και MARS. Η διαφορά μεταξύ των σχημάτων για  $t-1$  και  $t-2$  lagged values είναι πολύ μικρή με πιο ομαλοποιημένα αποτελέσματα για  $t-2$ . Η υπόλοιπη ανάλυση μα γίνει με lag  $t-2$  στους αλγορίθμους Random Forest, MARS και CART.



$\Sigma\chi\nu\alpha$  8.7: Random Forest - no lag



$\Sigma\chi\nu\alpha$  8.8: Random Forest - lag  $\dot{\epsilon}\omega\varsigma$  t-2



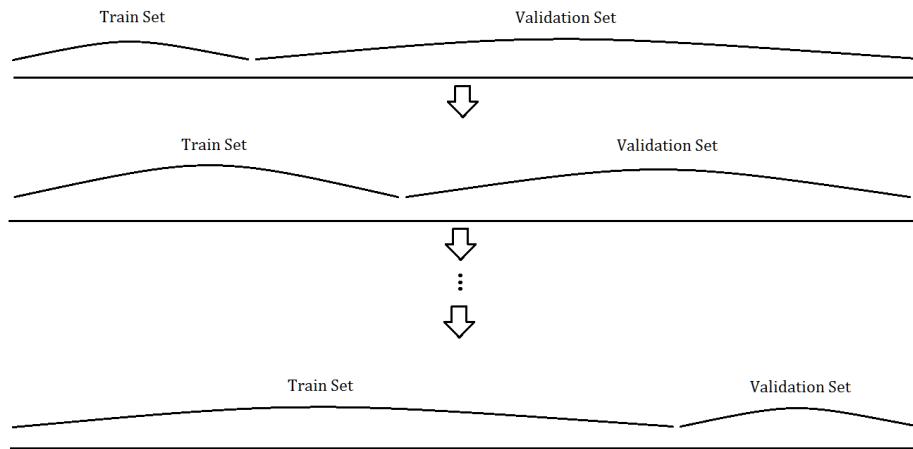
Σχήμα 8.9: MARS - lag έως t-2

## 8.4 Ανάλυση Χρονοσειράς

Στην επεξεργασία της χρονοσειράς χρησιμοποιήθηκαν και οι τρεις αλγόριθμοι με lagged values έως  $t - 2$ . Χρησιμοποιήθηκαν 3 διαφορετικοί τρόποι επεξεργασίας δεδομένων, τους οποίους αναλύουμε παρακάτω.

### 8.4.1 Διαφορετικά train και validation sets σε όλη την χρονοσειρά

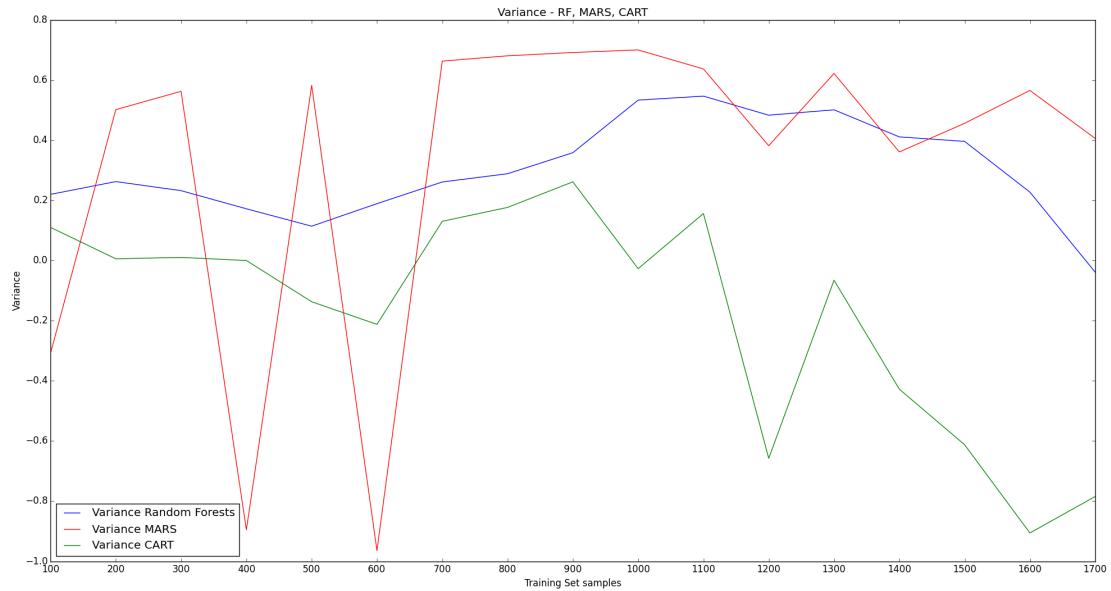
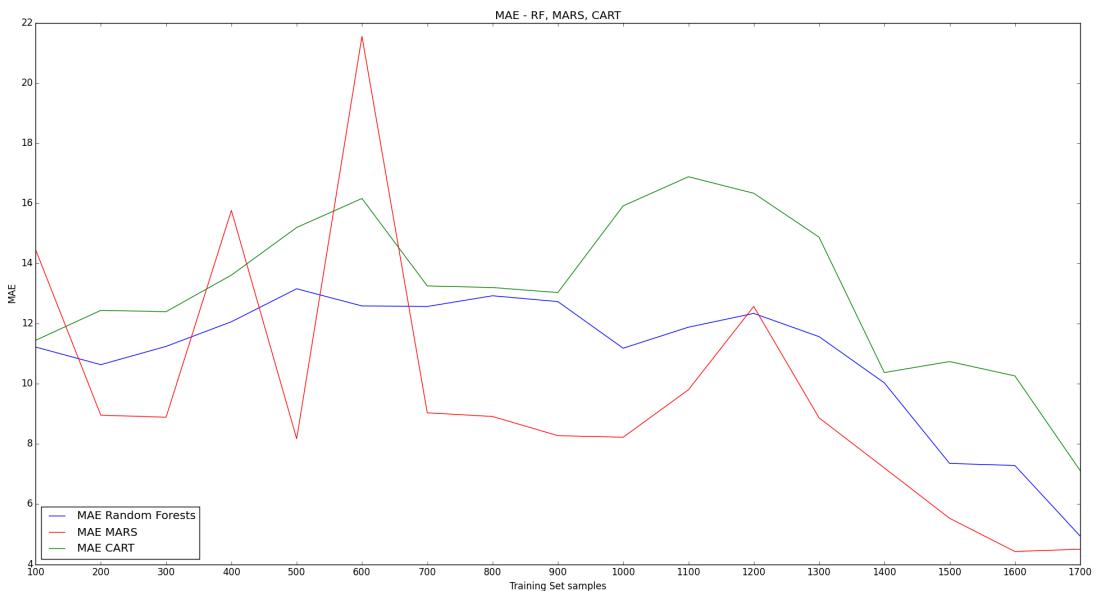
Αρχικά εφαρμόσαμε τους τρεις αλγορίθμους σε train και test sets που κάλυπταν όλη την χρονοσειρά. Το πιο μικρό train set ξεκινούσε από την αρχή έως τα 100 πρώτα δεδομένα και το υπόλοιπο ήταν test set, ενώ αυξάναμε κάθε φορά κατά 100 επιπλέον δεδομένα το train set και μειώναμε αντίστοιχα το test set ώσπου στο τέλος είχαμε 100 δεδομένα για test. Στο σχήμα 8.10 φαίνεται η διαδικασία που ακολουθήθηκε. Σε κάθε βήμα αποθηκεύαμε σε πίνακες τα δεδομένα για Mean Accuracy, Variance, MAE, MSE όπως και τις σημαντικότητες των μεταβλητών και ταυτόχρονα δημιουργούσαμε γραφικές παραστάσεις των προβλέψεων σε σχέση με τα πραγματικά μεγέθη.

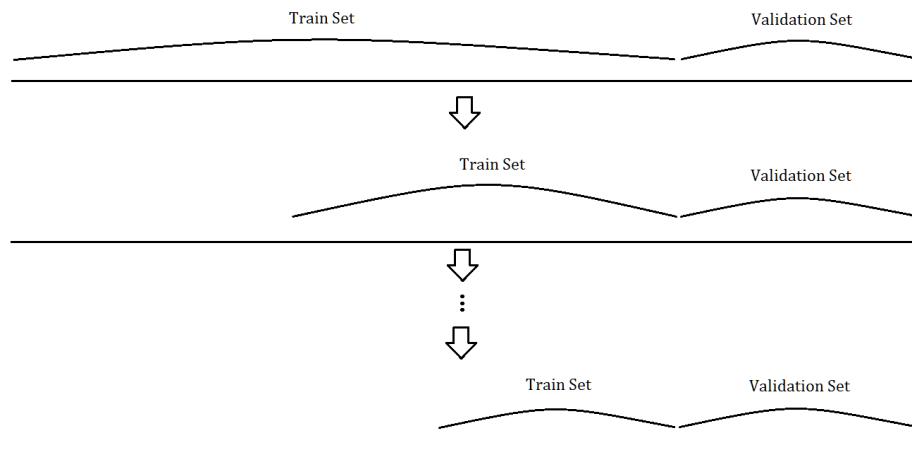


Σχήμα 8.10: Ανάλυση δεδομένων 1

Ο αλγόριθμος MARS παρουσίασε πολύ καλά αποτελέσματα, μάλιστα σε κάποια σημεία έφτασε ακρίβεια 70%. Ταυτόχρονα όμως για διαφορετικά δεδομένα είχε πολύ κακή επίδοση. Αντίθετα ο αλγόριθμος Random Forests ήταν πιο αξιόπιστος σε όλη την διάρκεια αλλά με μικρότερη επίδοση σε σχέση με τον MARS σε κάποια σημεία. Οι αλγόριθμοι παρουσίασαν τα βέλτιστα αποτελέσματα για train set μεγέθους 1000, το οποίο περιέχει μέσα όλες τις αρχικές τιμές της χρονοσειράς μαζί με κάποιες υψηλότερες τιμές. Στο σχήμα 8.2 που περιέχει το SMP, το βέλτιστο train set για την πρόβλεψη των υπόλοιπων τιμών έγινε από τις αρχές τους 2009 έως τέλη του 2011. Στην βέλτιστη περίπτωση ο αλγόριθμος MARS έχει variance στο 57% και παράγει καλύτερα αποτελέσματα. Σε άλλα training set όμως παρουσίασε βύθιση παράγοντας λανθασμένη έξοδο.

Όσον αφορά την σημαντικότητα των μεταβλητών, τόσο στον αλγόριθμο Random Forest όσο και στον αλγόριθμο MARS η πιο σημαντική μεταβλητή ήταν το SMP(t-1) ακολουθούμενο από το ngas. Στα Random Forest επόμενη πιο σημαντική μεταβλητή ήταν το smp(t-2) και έπειτα ερχόταν το load\_forecast, ngas(t-1), waip και waters. Στον αλγόριθμο MARS τρίτη σημαντικότερη μεταβλητή ήταν το ngas(t-1) ακολουθούμενο από τα smp(t-2), load\_forecast και waip. Συνολικά η σημαντικότητα των μεταβλητών παρουσιάζει πολλές ομοιότητες στους δύο αλγορίθμους.

 $\Sigma\chi\nu\alpha$  8.11: Variance - Ανάλυση 1η $\Sigma\chi\nu\alpha$  8.12: MAE - Ανάλυση 1η



Σχήμα 8.13: Ανάλυση δεδομένων 2

#### 8.4.2 Διαφορετικά train sets - ίδιο test set

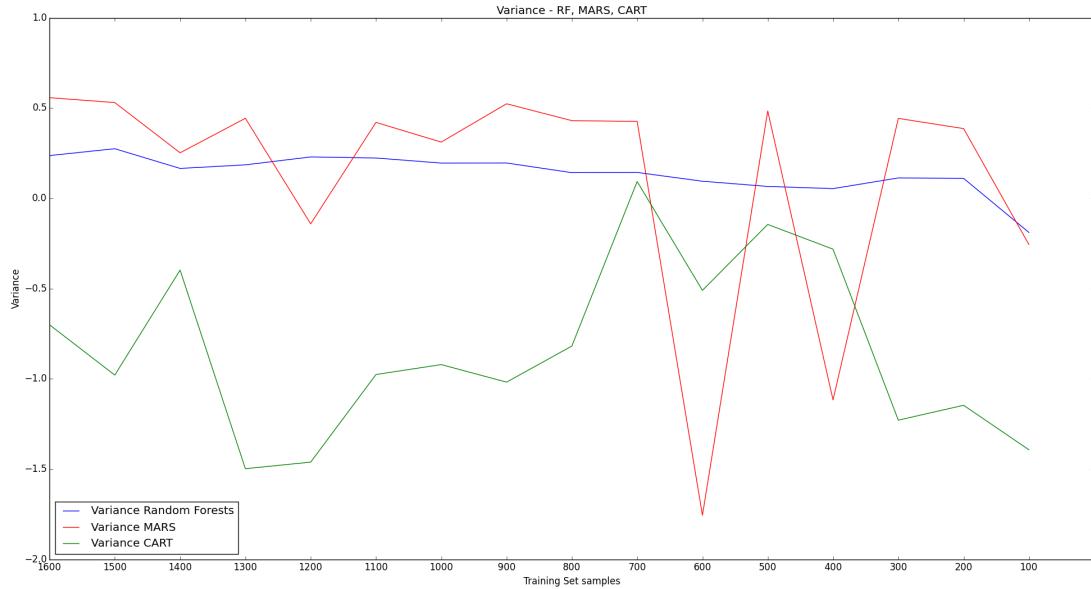
Τώρα έχουμε ένα validation set στο διάστημα [1600:] και διαφορετικά train sets τα οποία ανήκουν στο διάστημα [:1600-x],  $x_0 = 0$  και  $x$  αυξάνει ανά 100. Το παραπάνω φαίνεται στο σχήμα 8.13.

Τα αποτελέσματα φαίνονται στο σχήμα 8.14 και είναι παρόμοια με πριν. Έχουμε σε κάποια σημεία καλύτερες τιμές στο MARS, αλλά σε κάποια άλλα τιμές λανθασμένες. Το Random Forest παράγει μέτρια αποτελέσματα ενώ το MARS παράγει τα βέλτιστα αποτελέσματα για training sets 1500 και 1600 τιμών, δηλαδή ολόκληρου του δείγματος. Η σημαντικότητα των μεταβλητών σε αυτά τα δείγματα παραμένει ίδια με πριν και στους δύο αλγορίθμους. Επίσης παρατηρούμε πως εδώ ο αλγόριθμος CART δεν παράγει ικανά αποτελέσματα για προβλέψεις.

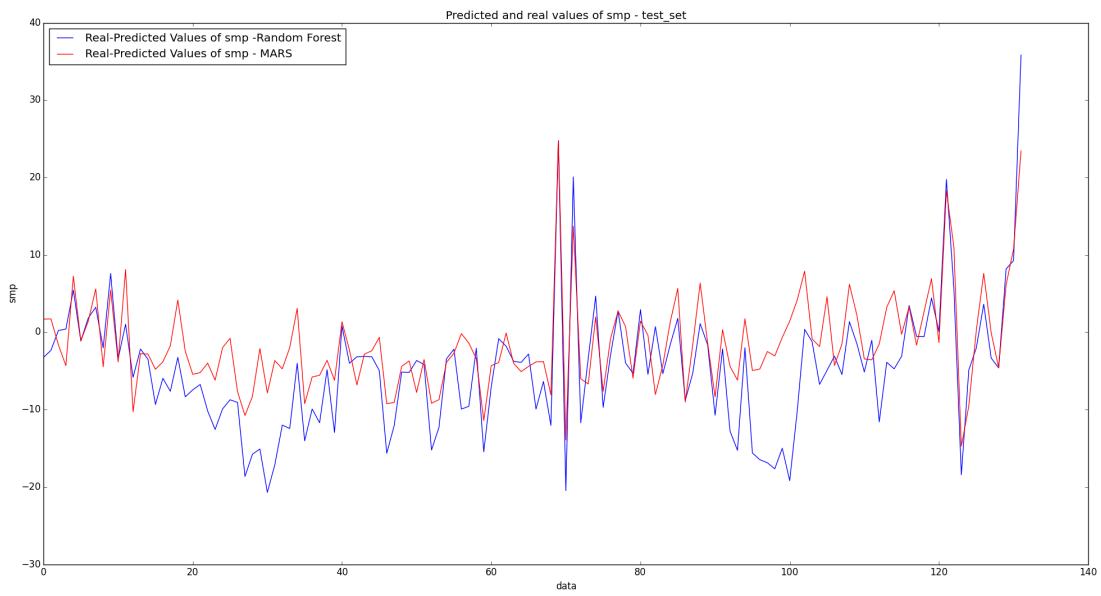
Στο σχήμα 8.15 έχουμε τις διαφορές των πραγματικών από τις εκτιμηθείσες τιμές για τους αλγορίθμους Random Forest και MARS και για train set 1600 τιμών. Παρατηρούμε πως ο αλγόριθμος MARS συνολικά δίνει καλύτερα αποτελέσματα. Οι αλγόριθμοι παίρνουν τιμές με μεγάλη απόχλιση από την πραγματική στα ίδια σημεία και συνολικά παρουσιάζουν παρόμοια συμπεριφορα.

#### 8.4.3 Μικρά train-test sets σε όλη την χρονοσειρά

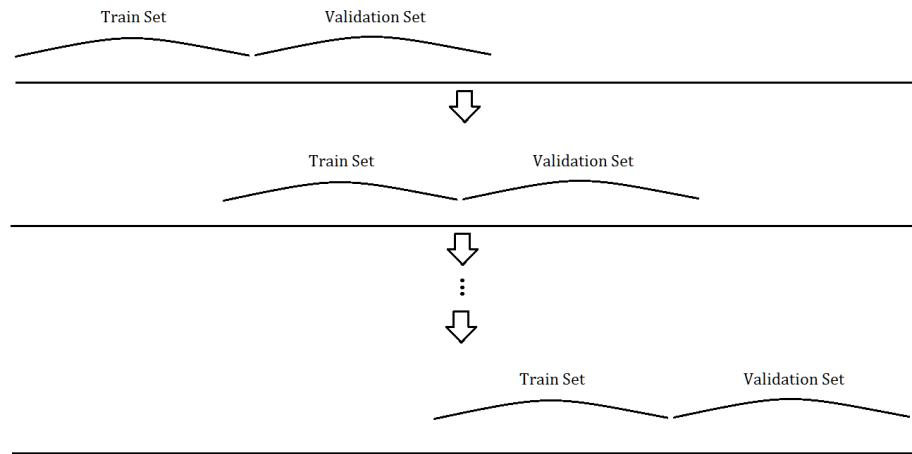
Για την επίδοση τιμημάτων δεδομένων στην χρονοσειρά, παίρνουμε train sets της μορφής  $[x:x+200]$  και test sets της μορφής  $[x+200:x+400]$ , όπου  $x_0 = 0$  και  $x$  αυξάνει κατά 100 σε κάθε iteration. Στο σχήμα 8.16 βλέπουμε την διαδικασία εποπτικά.



Σχήμα 8.14: Variance - Ανάλυση 2η



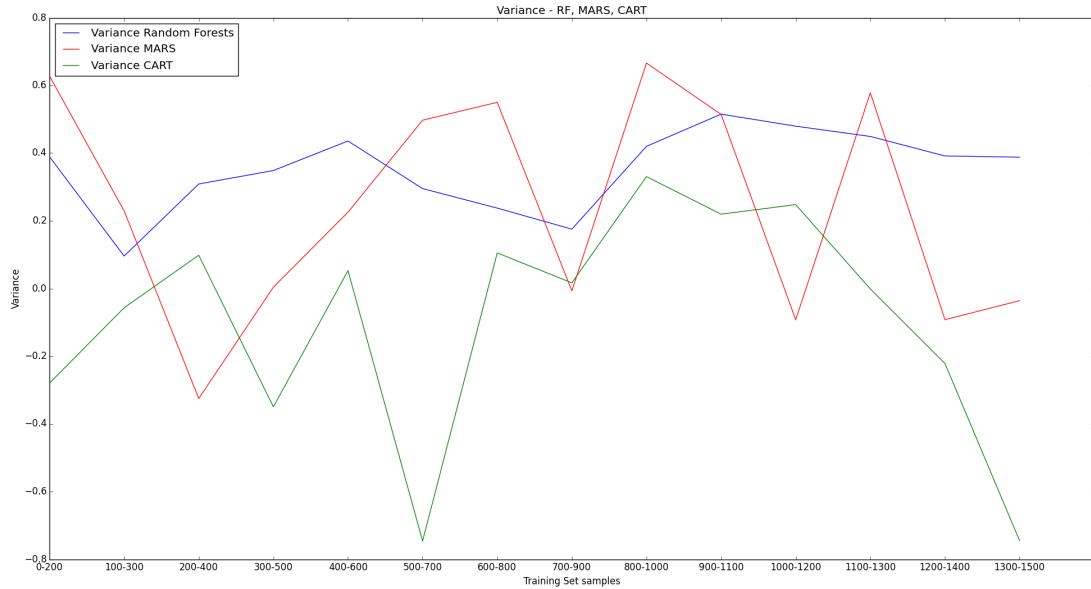
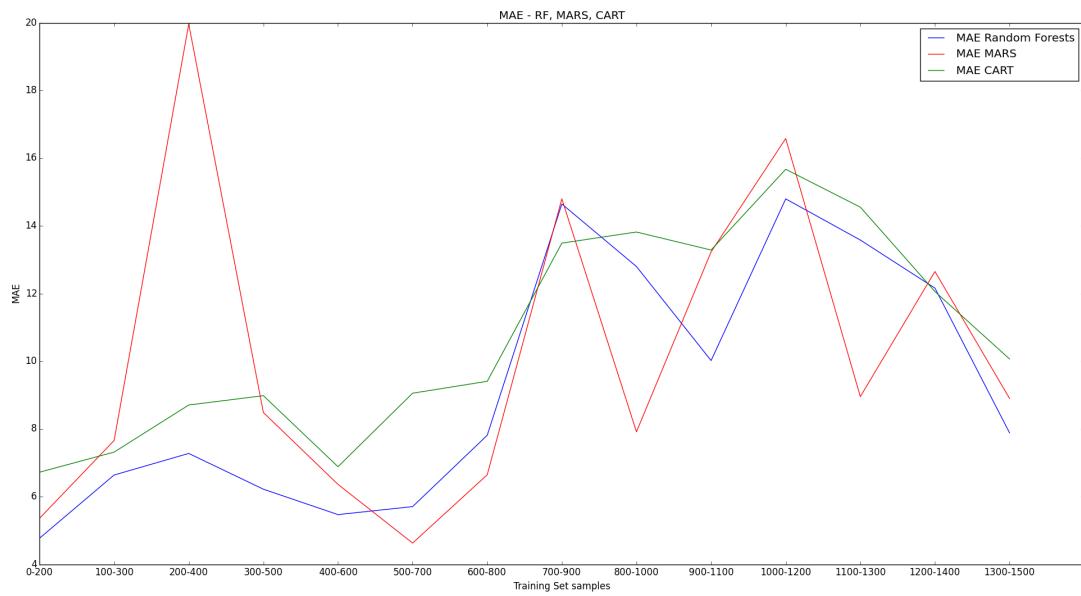
Σχήμα 8.15: Random Forest και MARS - Real - Predicted



Σχήμα 8.16: Ανάλυση δεδομένων 3

Τα αποτελέσματα είναι παρόμοια με πριν. Ο αλγόριθμος Random Forest έχει μεγάλη αξιοπιστία αλλά ο MARS μπορεί να δώσει καλύτερες προβλέψεις σε κάποιες περιπτώσεις ενώ σε κάποιες άλλες παράγει λανθασμένη έξοδο. Πιο συγκεκριμένα, ο αλγόριθμος MARS πετυχαίνει την βέλτιστη απόδοση στο train set [800:1000] ενώ ο Random Forest στο [900:1100]. Πολύ καλές επιδόσεις και από τους δύο αλγορίθμους υπάρχουν και σε άλλα σημεία της χρονοσειράς. Το βασικό χαρακτηριστικό όλων των σημείων είναι η ομοιογένειά τους, πράγμα που βοηθά τους αλγορίθμους στις προβλέψεις. Σημαντικό το γεγονός πως το μέσο απόλυτο σφάλμα είναι ελάχιστο και στους τρεις αλγορίθμους στο αρχικό κομμάτι της χρονοσειράς.

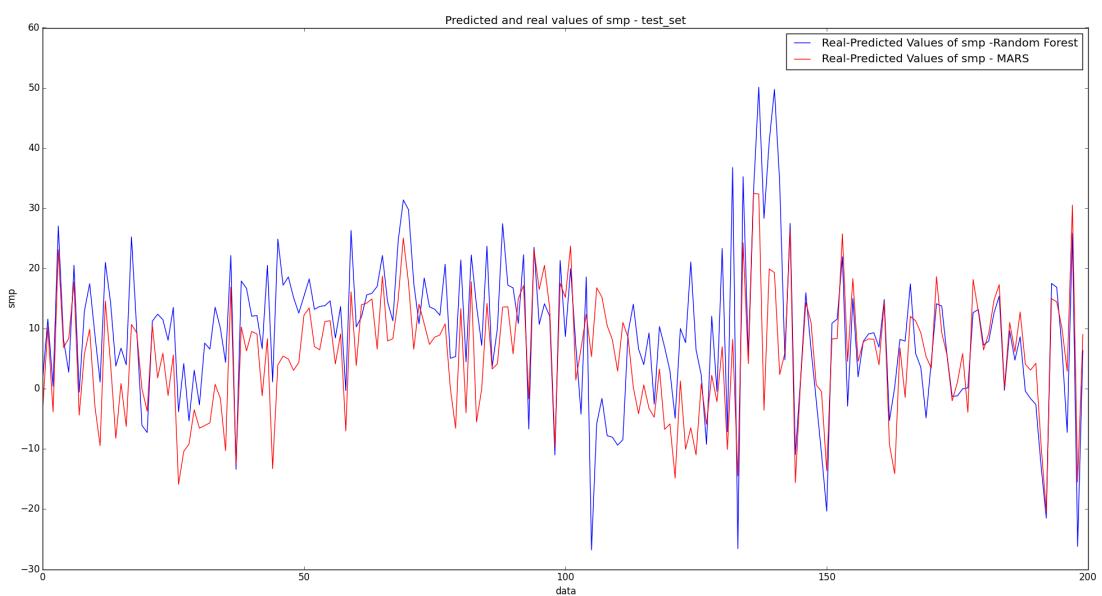
Σχετικά με τις σημαντικότητες, ο αλγόριθμος Random Forest βγάζει παρόμοια χαρακτηριστικά με πριν, με πολύ σημαντικό ρόλο να παίζει το SMP(t-1) και το ngas αλλά λόγω του μικρού δείγματος δεδομένων έχουμε αστάθεια στην σημαντικότητα της κάθε μεταβλητής. Στον αλγόριθμο MARS όμως τα πράγματα είναι ακόμη πιο διαφορετικά, με το ngas να κλαδεύεται κατά την δεύτερη φάση του αλγορίθμου και να μην παίζει ρόλο στην έξοδο. Στο σημείο [1100:1300] μάλιστα το ngas και το SMP(t-1) δεν χρησιμοποιούνται πολύ για τον καθορισμό της εξόδου με μεγαλύτερο ρόλο να παίζουν οι μεταβλητές load\_forecast και waters. Στο συγκεκριμένο σημείο είχαμε παρόμοια αποτελέσματα και για τον αλγόριθμο Random Forest ο οποίος θεώρησε την πιο σημαντική μεταβλητή το load\_forecast με 36.5%, 9.7% το ngas και 8.9% το SMP(t-1). Δύο στιγμιότυπα του αλγορίθμου MARS φαίνονται στα σχήματα 8.19 και 8.20.

 $\Sigma\chi\nu\alpha$  8.17: Variance - Ανάλυση 3η $\Sigma\chi\nu\alpha$  8.18: MAE - Ανάλυση 3η

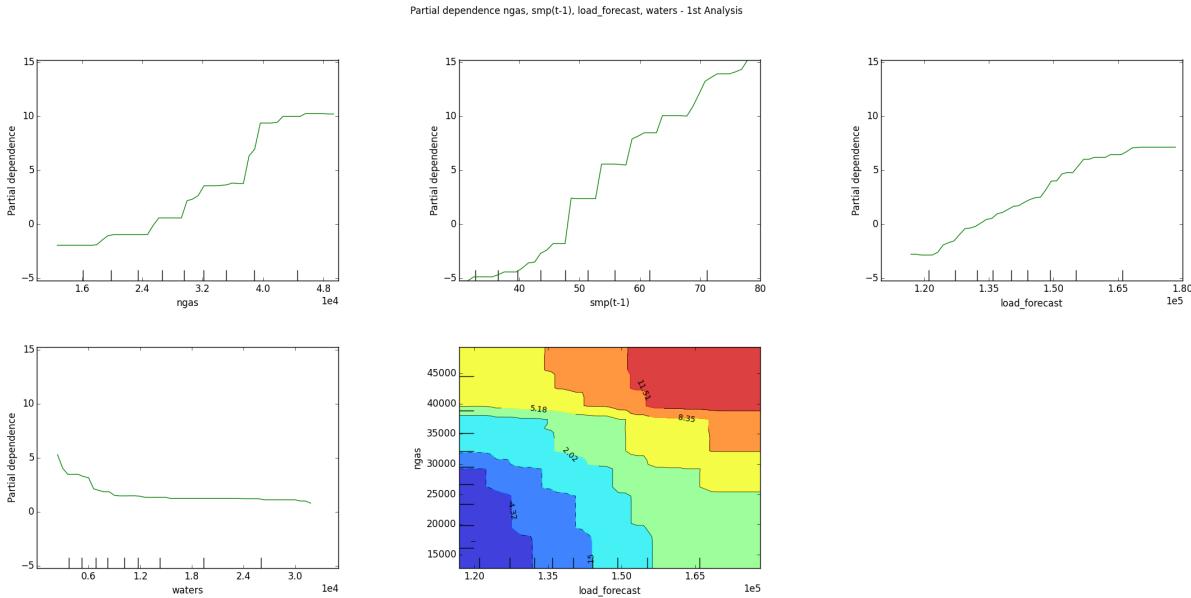
Basis Function	Pruned	Coefficient
(Intercept)	No	82.2856
h(smpbefore-54.33)	No	0.340432
h(54.33-smpbefore)	Yes	None
h(ngas-47872.2)	Yes	None
h(47872.2-ngas)	Yes	None
availabilitybefore	No	-0.00537945
load_forecast	No	0.000417858
load_forecastbefore	Yes	None
waters	No	-0.00228583
h(hydrogen-16294.8)	Yes	None
h(16294.8-hydrogen)	No	-0.00190056
res_forecast	Yes	None
availability	Yes	None
smpbefore2	Yes	None
hydrogenbefore2	Yes	None
h(ngasbefore2-30091.6)	Yes	None
h(30091.6-ngasbefore2)	Yes	None
res_forecastbefore	Yes	None
importsbefore2	Yes	None
lignite	Yes	None
exportsbefore	Yes	None
imports	Yes	None
waipbefore	Yes	None
waip	Yes	None
importsbefore	No	-0.00044731
ngasbefore	Yes	None
exportsbefore2	Yes	None

MSE: 46.5460, GCV: 60.7948, RSQ: 0.7272, GRSQ: 0.6472

Σχήμα 8.19: MARS Στιγμιότυπο - Ανάλυση 3η



Σχήμα 8.20: Random Forest και MARS - Real - Predicted Ανάλυση 3



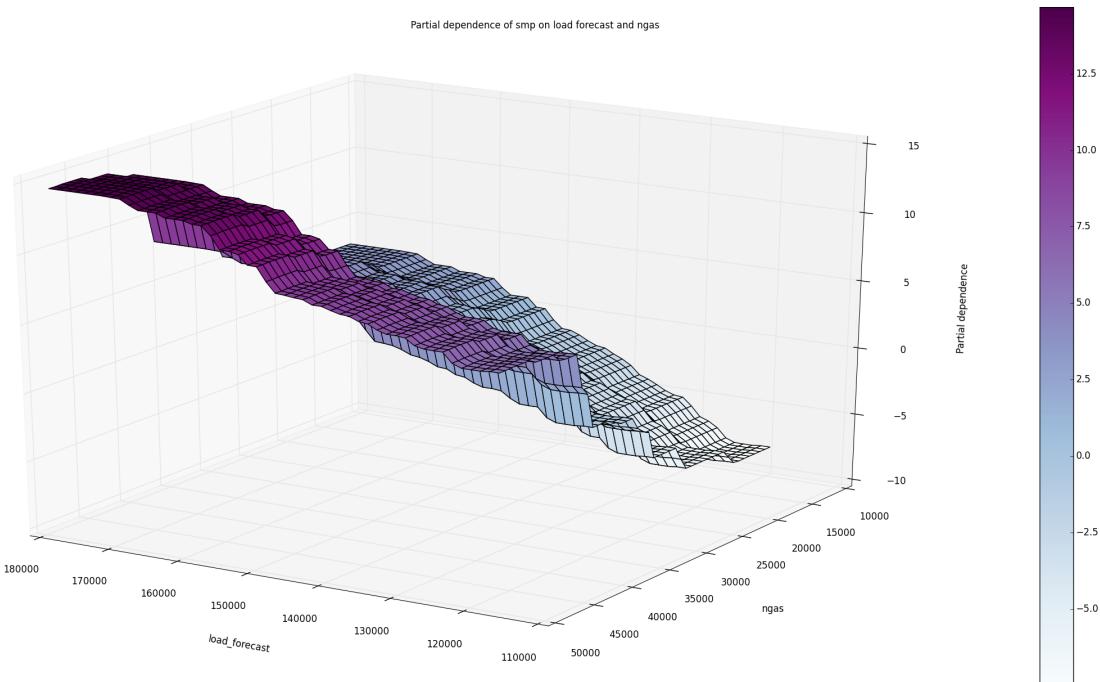
Σχήμα 8.21: Partial Dependence - Target Features

## 8.5 Partial Dependence Plots

Τα partial dependence plots δείχνουν την εξάρτηση μεταξύ μίας συνάρτησης στόχου και των σημαντικότερων χαρακτηριστικών που την καθορίζουν (target features) (3).

Για την δημιουργία των διαγραμμάτων επιλέξαμε τις μεταβλητές  $ngas$ ,  $SMP(t-1)$ ,  $load\_forecast$  και  $waters$  πάνω στο δείγμα της χρονοσειράς με train set [:1600] και validation set [1600:]. Τα partial dependence plots των target features φαίνονται στο σχήμα 8.21. Ο κάλετος άξονας είναι το  $SMP$  ομαλοποιημένο ως προς τις μεταβλητές που εξετάζουμε ενώ στον οριζόντιο άξονα έχουμε τα target features. Εξαίρεση αποτελεί το τελευταίο σχήμα στο οποίο δημιουργούμε επιφάνειες ώστε να απεικονίσουμε την σχέση δύο μεταβλητών με την έξοδο.

Αρχικά για το  $ngas$  και το  $SMP$  έχουμε μία σχέση που δείχνει σχεδόν σταθερή για μικρές τιμές  $SMP$  και αυξάνει απότομα όταν η τιμή του  $SMP$  μεγαλώνει. Αυτό έχει να κάνει με το γεγονός πως για μικρές τιμές ζήτησης και τιμής η παραγωγή φυσικού αερίου δεν επηρεάζει τόσο τις τιμές, λόγω του Variable Cost Recovery Mechanism που περιγράφτηκε στο κεφάλαιο 5. Σε χαμηλή ζήτηση οι παραγωγοί φυσικού αερίου λειτουργούν με χαμηλή δυναμικότητα και επιδοτούνται με τη διαφορά του μεταβλητού τους κόστους, οπότε το τελικό κόστος του φυσικού αερίου δεν εισέρχεται στις τιμές του  $SMP$  οπότε σε αυτή την περιπτώση η παραγωγή φυσικού αερίου δεν επηρεάζει τόσο την τιμή. Αντίθετα, όταν η ζήτηση αυξάνει, το φυσικό αέριο έχει έναν πολύ



Σχήμα 8.22: Partial Dependence of SMP on load\_forecast and ngas

σημαντικό ρόλο στον καθορισμό της προσφοράς λόγω του υψηλού οριακού κόστους παραγωγής του (κεφάλαιο 5.3) οπότε και είναι σημαντικός παράγοντας καθορισμού της τιμής του SMP.

Στο επόμενο μικρό σχήμα έχουμε την lagged value του SMP ( $SMP(t-1)$ ) η οποία βοηθά πάρα πολύ τα μοντέλα να πετύχουν καλύτερες προβλέψεις. Αυτό συμβαίνει διότι μέσω lagged μεταβλητών έχουμε την δυνατότητα να προσδιορίσουμε την τάση της χρονοσειράς, οπότε και να πετύχουμε καλύτερα αποτελέσματα. Παρατηρούμε πως η σχέση είναι σχεδόν γραμμική εκτός από ένα μικρό αρχικό κομμάτι, το οποίο σημαίνει πως η μεταβλητή αυτή είναι σημαντική σε ολόκληρο το μοντέλο.

Στο τρίτο σχήμα έχουμε την ζήτηση σε σχέση με το SMP. Αρχικά υπάρχει μία σχεδόν γραμμική συμπεριφορά των δύο μεταβλητών, αλλά μετά από κάποιο σημείο η επιπλέον αύξηση της ζήτησης δεν μπορεί να επηρεάσει το SMP. Αυτό συμβαίνει διότι το SMP είναι φραγμένη μεταβλητή και παίρνει τιμές στο διάστημα  $(0, 150]$  οπότε μετά από κάποιο σημείο η επιπλέον αύξηση της ζήτησης δεν μπορεί να προκαλέσει αλλαγή στην τιμή. Επίσης το μοντέλο μας σταθεροποιείται μετά από κάποια τιμή ζήτησης και οι μεταβλητές του επηρεάζουν με συγκεκριμένο τρόπο την έξοδο. Όσον αφορά την μεταβλητή waters, επηρεάζει με συγκεκριμένο και συνεχή τρόπο την έξοδο.

Στο τελευταίο μικρό σχήμα όπως και στο σχήμα 8.22 έχουμε εποπτικά μία επιφάνεια που δείχνει πόσο επηρεάζουν οι μεταβλητές ngas και load\_forecast το SMP. Ουσιαστικά δημιουργούνται περιοχές επηρεασμού της εξόδου, οι οποίες έχουν άμεση σχέση με την ζήτηση και την σημαντικότητα του φυσικού αερίου όπως περιγράψαμε.

## Κεφάλαιο 9

### Συμπεράσματα - Προοπτικές

Στην παρούσα διπλωματική εξετάστηκαν 3 πολύ σημαντικοί αλγόριθμοι του κλάδου της μηχανικής μάθησης οι οποίοι έχουν την δυνατότητα να προβλέψουν μη γραμμικές εξόδους με πολύ ικανοποιητικό τρόπο και υπερτερούν έναντι κλασσικών μοντέλων όπως για παράδειγμα το Linear regression. Η δυνατότητά τους να δημιουργούν δέντρα ή συναρτήσεις με βάση τα χαρακτηριστικά και την έξοδο που τους παρέχουμε χωρίς πιο πριν να έχουν γνώση του μοντέλου τα καθιστά δυνατά εργαλεία σε πολλές εφαρμογές.

Τα εργαλεία αυτά χρησιμοποιήθηκαν για να μοντελοποιήσουν την χονδρεμπορική αγορά της ηλεκτρικής ενέργειας βρίσκοντας σχέσεις και σημαντικότητες μεταξύ των predictors όπως και για να προβλέψουν την τιμή του SMP με βάση τα χαρακτηριστικά αυτά. Στο ερώτημα αν τα κατάφεραν η απάντηση είναι θετική. Οι προβλέψεις που παρήγαγαν τα μοντέλα κυμαίνονταν από 55% έως και περισσότερο από 75% με μικρές αποκλίσεις από τις πραγματικές τιμές. Πολύ σημαντικό είναι το γεγονός πως μπορούσαν να προβλέψουν με πολύ μεγάλη επιτυχία την τάση που θα είχε η τιμή, δηλαδή αν θα αυξανόταν ή μειωνόταν στη δεδομένη περίοδο. Η μεγάλη επιτυχία των μοντέλων αυτών, δεν προέρχεται μόνο από τα ίδια. Για να παράξουν τις επιθυμητές εξόδους χρειάζονταν μεταβλητές που να είναι σχετικές με το μοντέλο, χωρίς μεγάλη συνάφεια μεταξύ τους και κρίσιμες για την επιτυχία του. Επίσης ήταν σημαντικό που δεν παραλείψουν μεταβλητές κατά την κατασκευή του μοντέλου. Τα δύο αυτά στοιχεία -δύναμη των αλγορίθμων, σωστή επιλογή των μεταβλητών που επιλέχθηκαν- δημιούργησαν ένα πολύ καλό αποτέλεσμα.

Σχετικά με τις σχέσεις των μεταβλητών στο σύστημα, είδαμε συστηματικά πως το φυσικό αέριο διαδραματίζει σημαντικό ρόλο στον καυθορισμό της τιμής της χονδρεμπορικής αγοράς, και το είδαμε και από τους 3 αλγορίθμους σε πολλές διαφορετικές περιπτώσεις. Το γεγονός αυτό δεν είναι κάτι που μας εκπλήσσει δεδομένου πως έχουν γίνει έρευνες που περιγράφουν την σημαντικότητα του φυσικού αερίου στον καυθορισμό του SMP και υπάρχει και θεωρητική θεμελιώση πάνω σε αυτό η οποία περιγράφτηκε στα προηγούμενα κεφάλαια. Όμως αυτό το γεγονός σήγουρα αποδεικνύει πως τα μοντέλα κατανοούν σωστά τις αιτιώδεις σχέσεις μεταξύ των μεταβλητών στο σύστημα και παράγουν σημαντικότητες οι οποίες συμφωνούν με τα θεωρητικά συμπεράσματα.

Όσον αφορά την σταθερότητα των μοντέλων, αν εξαιρέσουμε την περίπτωση που παίρναμε πολύ μικρά train και validation sets, τα μοντέλα ήταν σταθερά ως προς την δομή τους και ως προς την σημαντικότητα των μεταβλητών. Με μικρότερα sets, μπορούσαμε να δούμε καλύτερα την δομή της αγοράς την περίοδο που εξετάζαμε οπότε και οι μεταβλητές είχαν διαφορετικούς ρόλους.

Σχετικά με τις προοπτικές που υπάρχουν στην αγορά της ηλεκτρικής ενέργειας, αρχικά μπορούμε να χρησιμοποιήσουμε επιπλέον αλγορίθμους μηχανικής μάθησης ώστε να κάνουμε έλεγχο της δυνατότητας των αλγορίθμων αυτών σε προβλέψεις. Μία επίδοση της τάξης του 65% είναι σημαντική, αλλά σίγουρα δεν είναι βέλτιστη. Θα μπορούσαμε να δοκιμάσουμε επιπλέον μεθόδους supervised learning ώστε να δούμε αν μπορούμε να πετύχουμε ακόμη καλύτερα αποτελέσματα.

Επιπλέον, μπορούμε να χρησιμοποιήσουμε τους ίδιους αλγορίθμους που εξετάστηκαν, και πιο συγκεκριμένα τους αλγορίθμους Random Forest και MARS που παράγουν καλύτερα αποτελέσματα, ώστε να ανακαλύψουμε τις αιτιώδεις σχέσεις μεταξύ των ανανεώσιμων και των μη ανανεώσιμων πηγών ενέργειας. Πιο συγκεκριμένα, μπορούμε να χρησιμοποιήσουμε τους αλγορίθμους αυτούς ώστε να εκτιμήσουμε την μείωση που δημιουργεί η χρήση ανανεώσιμων πηγών ενέργειας στην παραγωγή από συμβατικές πηγές. Με αυτό τον τρόπο θα μπορέσουμε να ποσοτικοποιήσουμε και το μειωμένο κόστος από την εισαγωγή φυσικού αερίου και λιγνίτη στο σύστημα όπως και τις εκπομπές αερίων που αποφεύγονται. Επιπρόσθετα, μπορούμε να προβλέψουμε την πιστωτική ικανότητα (capacity credit) των ανανεώσιμων πηγών ενέργειας, δηλαδή το μέγεθος των συμβατικών πηγών που θα μπορούσαν να αντικατασταθούν από ΑΠΕ, διατηρώντας παράλληλα το επίπεδο αξιοπιστίας του συστήματος.

Συμπερασματικά, οι αλγόριθμοι μηχανικής μάθησης μπορούν να αποβούν καθοριστικής σημασίας σε πολλούς και διαφορετικούς τομείς, και αποτελούν ένα πολύ δυνατό εργαλείο ανάλυσης το οποίο χρησιμοποιείται και θα χρησιμοποιηθεί ακόμη περισσότερο στο μέλλον, ειδικά αν αναλογιστούμε πως η πληροφορία μεγαλώνει εκθετικά, παρουσιάζοντας αντίστοιχη συμπεριφορά με τον νόμο του Moore. Πιο συγκεκριμένα, υπολογίζεται ότι μέχρι το 2020 τα ψηφιακά δεδομένα θα ξεπερνάνε σε μέγεθος τα 40 zettabytes το οποίο αντιστοιχεί σε 5200 GB για κάθε άνθρωπο στην γη [24]. Διασθητικά το μέγεθος αυτό είναι 57 φορές μεγαλύτερο από όλους τους κόχκους άμμου σε όλες τις παραλίες της γης. Από αυτά τα δεδομένα, το 33% θα μπορεί να αναλυθεί και να δώσει σημαντικά αποτελέσματα, το οποίο υποδεικνύει πως ο κλάδος αυτός θα παίζει καθοριστικό ρόλο στην κατανόηση των δεδομένων αυτών όπως και στην εύρεση λύσεων. Γνωρίζουμε εξάλλου πως παρότι δεν μπορούμε να προβλέψουμε ακριβώς το μέλλον, όσο πιο κοντά είμαστε στο να κατανοήσουμε τις αιτίες που δημιουργούν κάποιες καταστάσεις τόσο πιο κοντά είμαστε και στην εύρεση λύσεων οι οποίες θα μπορούν να σχηματίσουν το μέλλον πιο κοντά στις δικές μας απαιτήσεις.

# Βιβλιογραφία

- [1] Assessment of Policy Impacts on Sustainability in Europe.
- [2] Efron, Bradley and Tibshirani, Robert J. *An introduction to the bootstrap*. CRC press, 1994.
- [3] Marianthi Markatou George Hripcak Hong Tian, Shameek Biswas. *Analysis of Variance of Cross-Validation Estimators of the Generalization Error*. Journal of Machine Learning Research 6, 2005.
- [4] Robert Tibshirani Jerome Friedman, Trevor Hastie. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer, 2009.
- [5] Yaser S. Abu-Mostafa Malik Magdon-Ismail. *Learning From Data*. AMLBook, 2012.
- [6] Ron Kohavi. *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, 1995.
- [7] Leo Breiman Adele Cutler. *Random Forests*. Machine Learning Journal.
- [8] Ana-Maria Staicu. Classification Trees and MARS.
- [9] Andrew W and Schaal Stefan Atkeson, Christopher G and Moore. Locally weighted learning. *Artificial Intelligence Review*, 1999.
- [10] Berk, Richard A. *Statistical learning from a regression perspective*. Springer, 2008.
- [11] Bosco, Bruno and Parisio, Lucia and Pelagatti, Matteo and Baldi, Fabio. Long-run relations in European electricity prices. *Journal of applied econometrics*, 25(5):805–832, 2010.
- [12] Breiman, Leo and Friedman, Jerome and Stone, Charles J and Olshen, Richard A. *Classification and regression trees*. CRC press, 1984.

- [13] Carolin Strobl, James Malley, Gerhard Tutz. *An Introduction to Recursive Partitioning: Rationale, Application and Characteristics of Classification and Regression Trees, Bagging and Random Forests*. Psychol Methods, 2009.
- [14] Clauset, Aaron. A brief primer on probability distributions. *Santa Fe Institute*, 2011.
- [15] Criminisi, A. and Shotton, J. and Konukoglu, E. *Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning*. 2011.
- [16] De Jong, Cyriel and Schneider, Stefan. Cointegration between gas and power spot prices. *The Journal of Energy Markets*, 2(3):27–46, 2009.
- [17] Deng, Houtao and Runger, George and Tuv, Eugene. Bias of importance measures for multi-valued attributes and solutions. Springer, 2011.
- [18] Ferkingstad, Egil and Løland, Anders and Wilhelmsen, Mathilde. Causal modeling and inference for electricity markets. *Energy Economics*, 33(3):404–412, 2011.
- [19] Fielding, Alan. *Machine learning methods for ecological applications*. Springer, 1999.
- [20] Frayman, Yakov and Rolfe, Bernard F and Webb, Geoffrey I. Solving regression problems using competitive ensemble models. Springer, 2002.
- [21] Friedman, Jerome H. Multivariate adaptive regression splines. *The annals of statistics*, 1991.
- [22] Friedman, Jerome H and Roosen, Charles B. An introduction to multivariate adaptive regression splines. *Statistical Methods in Medical Research*, 4(3):197–217, 1995.
- [23] Furió, Dolores and Chuliá, Helena. Price and volatility dynamics between electricity and fuel costs: Some evidence for Spain. *Energy Economics*, 34(6):2058–2065, 2012.
- [24] Gantz, John and Reinsel, David. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the Future*, 2012.
- [25] Lee, Sang Jun and Siau, Keng. A review of data mining techniques. *Industrial Management & Data Systems*, 101(1):41–46, 2001.
- [26] Muñoz, Jesús and Felicísimo, Ángel M. Comparison of statistical methods commonly used in predictive modelling. *Journal of Vegetation Science*, 15(2):285–292, 2004.

- [27] Quinlan, J. Ross. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.
- [28] Saar-Tsechansky, Maytal and Provost, Foster. Handling missing values when applying classification models. 2007.
- [29] SPSS Inc. *SPSS Clementine 12.0 Algorithms Guide*. 2007.
- [30] Alexander I Tetko, Igor V and Livingstone, David J and Luik. Neural network studies. 1. Comparison of overfitting and overtraining. *Journal of Chemical Information and Computer Sciences*, 35(5):826–833, 1995.



# Iστότοποι

- (1) Cross Validation error. <http://www.cs.cmu.edu/~schneide/tut5/node42.html>.
- (2) Judging Model Quality by Residuals. <http://www.cs.cmu.edu/~schneide/tut5/node41.html>.
- (3) Partial Dependence Plots. [http://scikit-learn.org/stable/auto\\_examples/ensemble/plot\\_partial\\_dependence.html](http://scikit-learn.org/stable/auto_examples/ensemble/plot_partial_dependence.html).
- (4) Shannon Entropy. <http://www.ueltschi.org/teaching/chapShannon.pdf>.
- (5) Supervised Learning Workflow and Algorithms. <http://www.mathworks.com/help/stats/supervised-learning-machine-learning-workflow-and-algorithms.html/>.
- (6) Titanic: Machine Learning from Disaster. <http://www.kaggle.com/c/titanic-gettingStarted>.
- (7) A short example of multivariate adaptive regression splines (MARS). <http://qizeresearch.wordpress.com/2013/12/04/a-short-example-of-multivariate-adaptive-regression-splines-mars/>.
- (8) Binary recursive partitioning. <http://www.unc.edu/courses/2010spring/ecol/562/001/docs/lectures/lecture21.htm>.
- (9) Multivariate Adaptive Regression Splines. <https://www.statsoft.com/Textbook/Multivariate-Adaptive-Regression-Splines>.
- (10) py-earth. <https://github.com/jcrudy/py-earth>.
- (11) Scikit-learn. <http://scikit-learn.org/>.
- (12) Training in MARS. <http://www.salford-systems.com/videos/training/mars>.