



HMMY · ΕΛΜΕΠΑ · 9ο εξάμηνο

Μηχανική Μάθηση και Εξόρυξη Γνώσης

Διάλεξη 2: Δεδομένα και προεπεξεργασία
Ένα έργο MM από την αρχή ως το τέλος

Λευτέρης Κουμάκης

Από την προηγούμενη φορά



Είδη μάθησης

Επιβλεπόμενη (ταξινόμηση, παλινδρόμηση), μη επιβλεπόμενη, ενισχυτική



CRISP-DM

Κύκλος: πρόβλημα → δεδομένα → προετοιμασία → μοντέλο → αξιολόγηση → ανάπτυξη



Bias–variance

Υποπροσαρμογή και υπερπροσαρμογή, το δίλημμα της πολυπλοκότητας



Αξιολόγηση

Train / validation / test και cross-validation

Στόχοι της σημερινής διάλεξης



Πλαισίωση

Να ορίζετε σωστά το πρόβλημα και το μέτρο απόδοσης



Εξερεύνηση

Να εντοπίζετε ευρήματα και παγίδες στα δεδομένα



Προετοιμασία

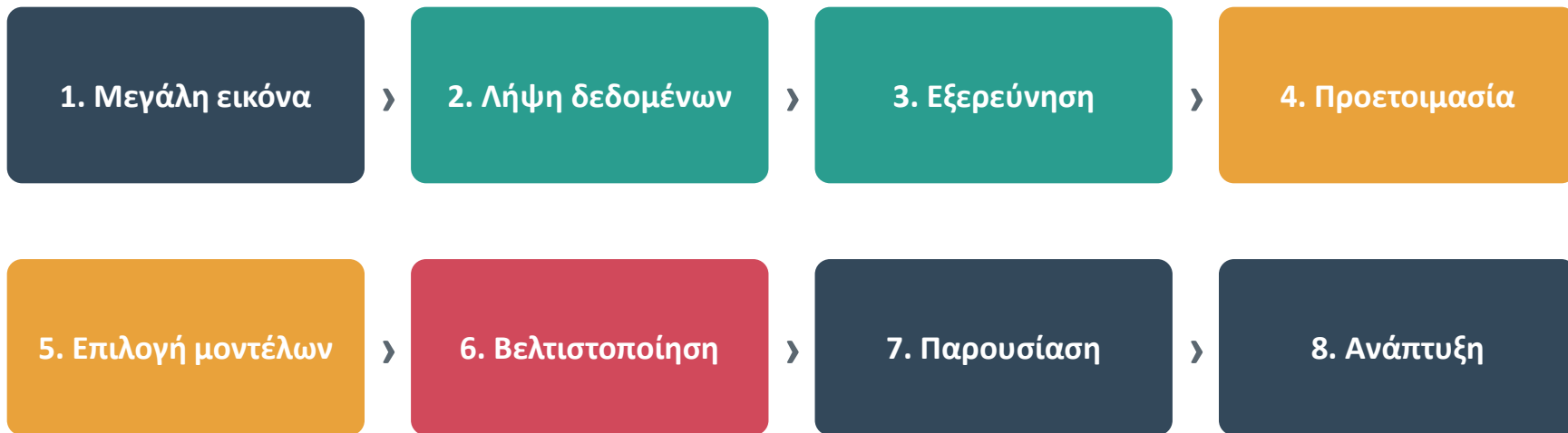
Ελλιπείς τιμές, κατηγορικά, κλιμάκωση, νέα χαρακτηριστικά



Pipelines

Αυτοματοποίηση χωρίς διαρροή δεδομένων

Οι 8 φάσεις ενός έργου MM



Σήμερα περνάμε από όλες τις φάσεις με ένα παράδειγμα. Οι φάσεις 5–6 αναλύονται σε βάθος στις επόμενες διαλέξεις.

Το σενάριο: τιμές κατοικιών στην Καλιφόρνια

20.640

περιοχές (γραμμές)

10

χαρακτηριστικά

1

κατηγορικό (εγγύτητα σε
ωκεανό)

1990

απογραφή ΗΠΑ

- Στόχος: η διάμεση τιμή κατοικίας κάθε περιοχής (median_house_value)
- Χαρακτηριστικά: γεωγρ. μήκος/πλάτος, ηλικία κατοικιών, δωμάτια, υπνοδωμάτια, πληθυσμός, νοικοκυριά, διάμεσο εισόδημα, εγγύτητα σε ωκεανό
- Διαθέσιμο με μία εντολή στο notebook του κεφαλαίου

ΦΑΣΗ 1

Η μεγάλη εικόνα

Πλαισίωση του προβλήματος και επιλογή μέτρου απόδοσης

Πλαισίωση: οι ερωτήσεις πριν από τον κώδικα



Ποιος είναι ο επιχειρησιακός στόχος;

Τι απόφαση θα πάρει κάποιος με βάση την πρόβλεψη;



Ποια είναι η τρέχουσα λύση;

Πώς γίνεται σήμερα; Αυτό είναι το σημείο αναφοράς (baseline).



Πού καταλήγει η έξοδος;

Σε άνθρωπο; Σε άλλο σύστημα; Σε ποια μορφή;



Ποιοι είναι οι περιορισμοί;

Χρόνος απόκρισης, όγκος δεδομένων, ερμηνευσιμότητα.

Τι είδους πρόβλημα είναι;



Επιβλεπόμενη

Έχουμε την τιμή-στόχο για κάθε περιοχή



Παλινδρόμηση

Προβλέπουμε αριθμό (τιμή σε \$)



Πολλαπλή, μονομεταβλητή

Πολλά χαρακτηριστικά εισόδου, μία τιμή εξόδου



Batch μάθηση

Δεν υπάρχει συνεχής ροή· τα δεδομένα χωρούν στη μνήμη

Μέτρα απόδοσης για παλινδρόμηση

$$\text{RMSE} = \sqrt{[(1/m) \cdot \Sigma (\hat{y}_i - y_i)^2]}$$

Root Mean Square Error · νόρμα ℓ_2
Τιμωρεί πολύ τα μεγάλα σφάλματα

$$\text{MAE} = (1/m) \cdot \Sigma |\hat{y}_i - y_i|$$

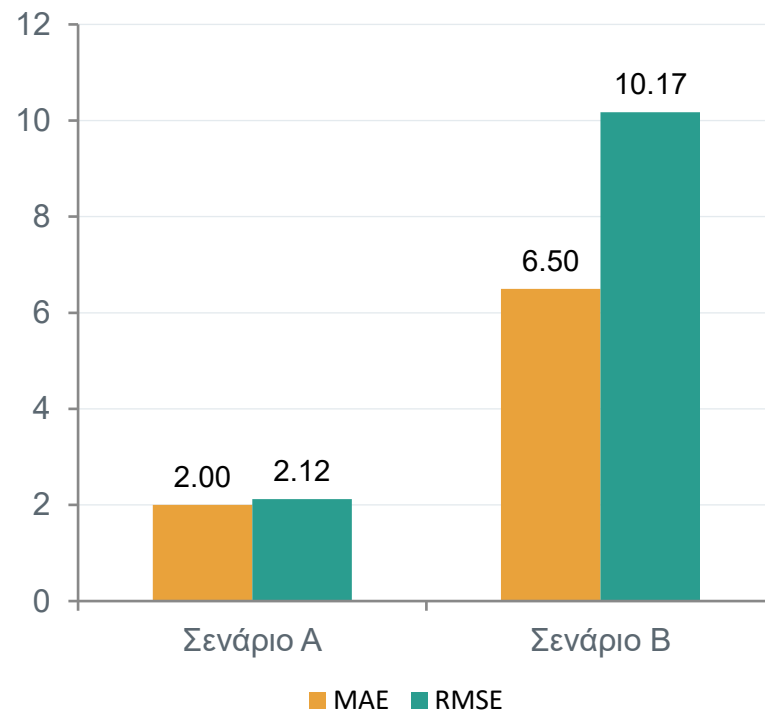
Mean Absolute Error · νόρμα ℓ_1
Πιο ανθεκτικό σε ακραίες τιμές

Κανόνας: όσο μεγαλύτερος ο εκθέτης της νόρμας, τόσο περισσότερο «μετράνε» οι μεγάλες τιμές. Γι' αυτό το RMSE προτιμάται όταν οι ακραίες τιμές είναι σπάνιες (π.χ. κανονική κατανομή σφαλμάτων).

Παράδειγμα: RMSE έναντι MAE με μία ακραία τιμή

	Σφάλματα	MAE	RMSE
Σενάριο A	2, -1, 3, -2	2,00	2,12
Σενάριο B	2, -1, 3, -20	6,50	10,17

- Ένα μόνο μεγάλο σφάλμα πολλαπλασιάζει το RMSE επί 4,8
- Το MAE αυξάνεται «μόνο» επί 3,25
- Επιλέγουμε μέτρο ανάλογα με το πόσο «κοστίζει» ένα μεγάλο λάθος



Ελέγξτε τις υποθέσεις σας

Υποθέτουμε: χρειάζονται τιμές
σε \$



Ρωτάμε την ομάδα του επόμενου
συστήματος



Απάντηση: «φθινό / μέτριο /
ακριβό»



Συμπέρασμα

Το πρόβλημα είναι ταξινόμηση, όχι παλινδρόμηση. Λίστα ελέγχου: ποια είναι η μορφή εξόδου, πόση ακρίβεια χρειάζεται και ποιο σφάλμα είναι αποδεκτό;



Ερώτηση Slido 1 από 3



Προβλέπετε χρόνους παράδοσης. Λίγες παραγγελίες έχουν τεράστιες καθυστερήσεις λόγω απεργιών, και δεν θέλετε να κυριαρχούν στην αξιολόγηση. Ποιο μέτρο είναι καταλληλότερο;

A

RMSE

B

MAE

Γ

R^2

Δ

Ακρίβεια (accuracy)

ΦΑΣΗ 2

Λήψη δεδομένων και test set

Πρώτη ματιά, και γιατί το test set δημιουργείται αμέσως

Πρώτη ματιά: τι μας λέει το info()

Χαρακτηριστικό	Μη κενές	Τύπος
longitude, latitude	20.640	float
housing_median_age	20.640	float
total_rooms	20.640	float
total_bedrooms	20.433	float
population, households	20.640	float
median_income	20.640	float
median_house_value	20.640	float
ocean_proximity	20.640	object



Ελλιπείς τιμές

207 περιοχές χωρίς total_bedrooms

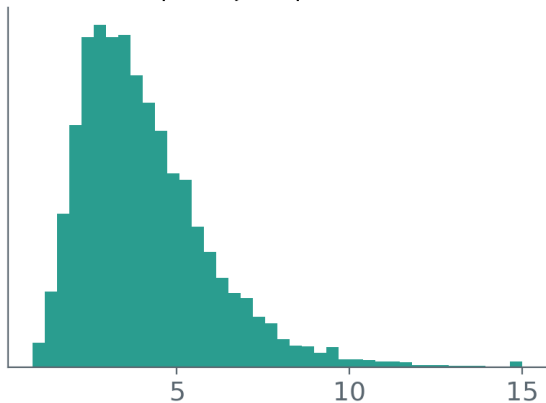


Κατηγορικό

ocean_proximity: 5 τιμές, π.χ. INLAND, NEAR BAY

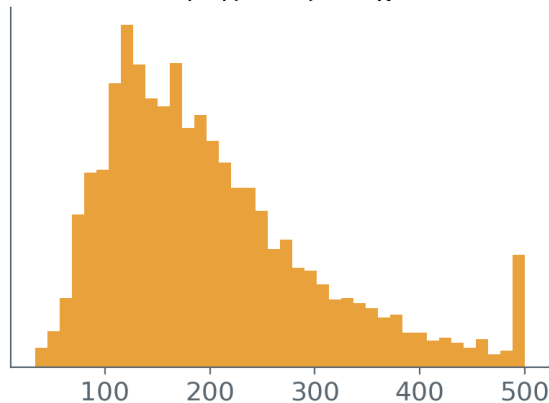
Τα ιστογράμματα αποκαλύπτουν παγίδες

Διάμεσο εισόδημα
(ουρά δεξιά, όριο στο 15)



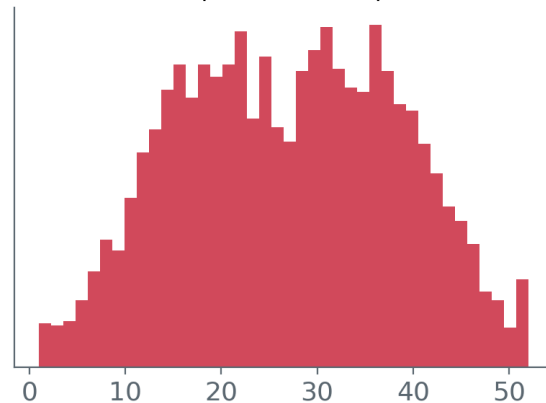
Κλιμακωμένο, όχι σε \$: ρωτάμε την ομάδα που το συνέλεξε

Τιμή κατοικίας (χιλ. \$)
(«κόφτης» στις 500 χιλ.)



Αν χρειαζόμαστε τιμές > 500 χιλ.,
συλλέγουμε σωστές ετικέτες ή
αφαιρούμε αυτές τις περιοχές

Ηλικία κατοικιών
(όριο στα 52 έτη)



Ουρές και διαφορετικές κλίμακες: θα
χρειαστούν μετασχηματισμοί

Μην κρυφοκοιτάξετε το test set (data snooping)

Κάνουμε το test set ΤΩΡΑ, πριν κοιτάξουμε αναλυτικά τα δεδομένα



Τι κινδυνεύουμε

Να «δούμε» ένα πρότυπο στο test set και να διαλέξουμε μοντέλο με βάση αυτό



Συνέπεια

Η εκτίμηση γενίκευσης γίνεται αισιόδοξη και το σύστημα απογοητεύει στην πράξη

Τυχαίος διαχωρισμός train/test

```
from sklearn.model_selection import train_test_split

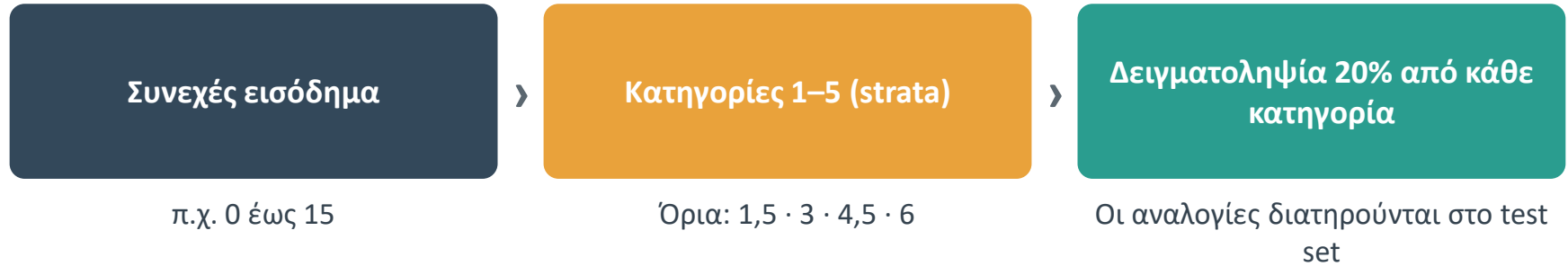
train_set, test_set = train_test_split(
    housing, test_size=0.2, random_state=42)

len(train_set), len(test_set)    # (16512, 4128)
```

- 20% για έλεγχο: 4.128 περιοχές
- random_state: ίδιος διαχωρισμός κάθε φορά
- Με νέα δεδομένα: διαχωρισμός με βάση hash ενός σταθερού ID

Πρόβλημα: αν το dataset δεν είναι μεγάλο, ένας τυχαίος διαχωρισμός μπορεί να δώσει μη αντιπροσωπευτικό test set.

Στρωματοποιημένη δειγματοληψία (stratified sampling)



Κάθε κατηγορία πρέπει να έχει αρκετά δείγματα, αλλιώς η εκτίμηση της σημασίας της θα είναι μεροληπτική. Γι' αυτό δεν φτιάχνουμε πάρα πολλές κατηγορίες.

Παράδειγμα: στρωματοποιημένο έναντι τυχαίου test set

Κατηγορία εισοδήματος	Σύνολο (%)	Στρωματοποιημένο (%)	Τυχαίο (%)
1	3,99	4,00	3,85
2	32,12	32,12	31,06
3	34,90	34,91	35,27
4	18,03	18,02	18,36
5	10,96	10,97	11,46

Προσομοίωση: 20.640 δείγματα, test set 20%

Στον κώδικα: μία παράμετρος αρκεί

```
import numpy as np, pandas as pd

housing["income_cat"] = pd.cut(
    housing["median_income"],
    bins=[0., 1.5, 3.0, 4.5, 6., np.inf],
    labels=[1, 2, 3, 4, 5])

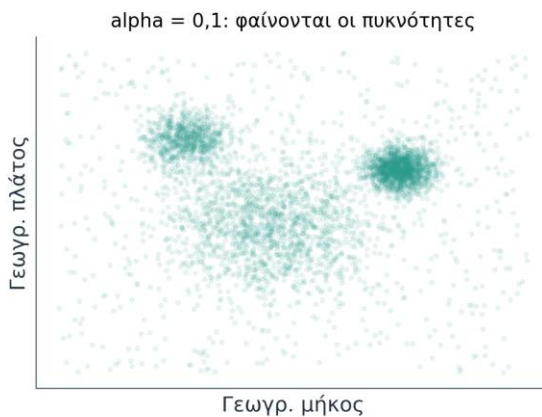
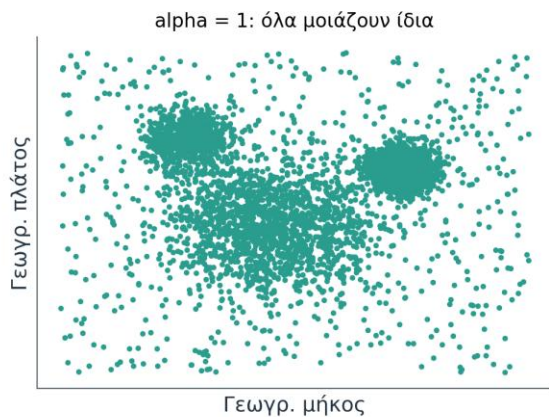
strat_train, strat_test = train_test_split(
    housing, test_size=0.2,
    stratify=housing["income_cat"], random_state=42)
```

ΦΑΣΗ 3

Εξερεύνηση και οπτικοποίηση

Γραφήματα, συσχετίσεις και συνδυασμοί χαρακτηριστικών

Γεωγραφική οπτικοποίηση: η διαφάνεια βοηθά

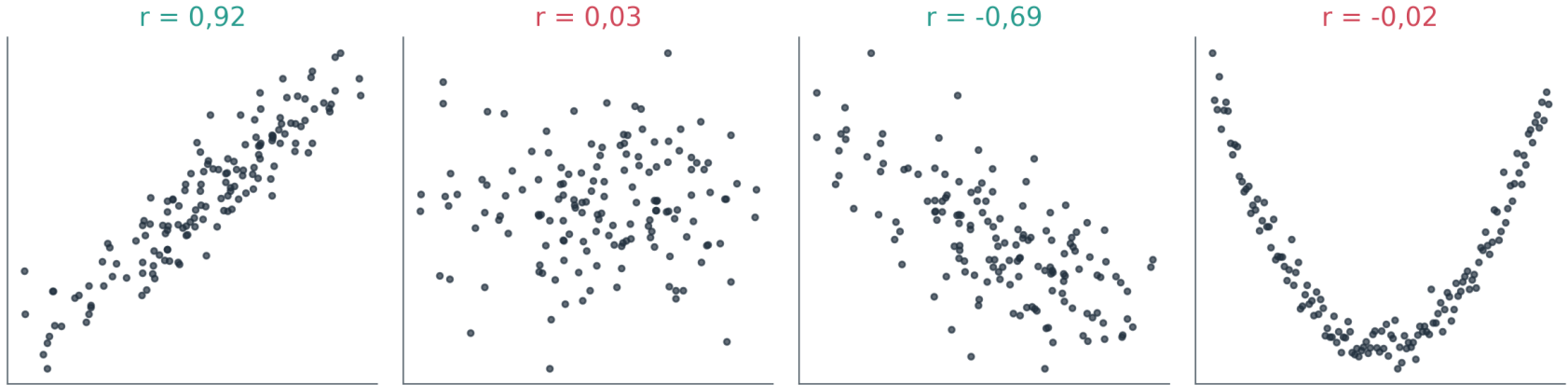


- Η τιμή σχετίζεται με την τοποθεσία και την πυκνότητα
- Χρώμα = τιμή, μέγεθος = πληθυσμός
- Ιδέα: απόσταση από το πλησιέστερο αστικό κέντρο ως νέο χαρακτηριστικό

Ένα καλό γράφημα οδηγεί σε ιδέες για νέα χαρακτηριστικά.

Ο συντελεστής συσχέτισης Pearson

$$r = \Sigma (x_i - \bar{x})(y_i - \bar{y}) / \sqrt{[\Sigma (x_i - \bar{x})^2 \cdot \Sigma (y_i - \bar{y})^2]}$$



Ισχυρή θετική · καμία · αρνητική · μη γραμμική σχέση με $r \approx 0$

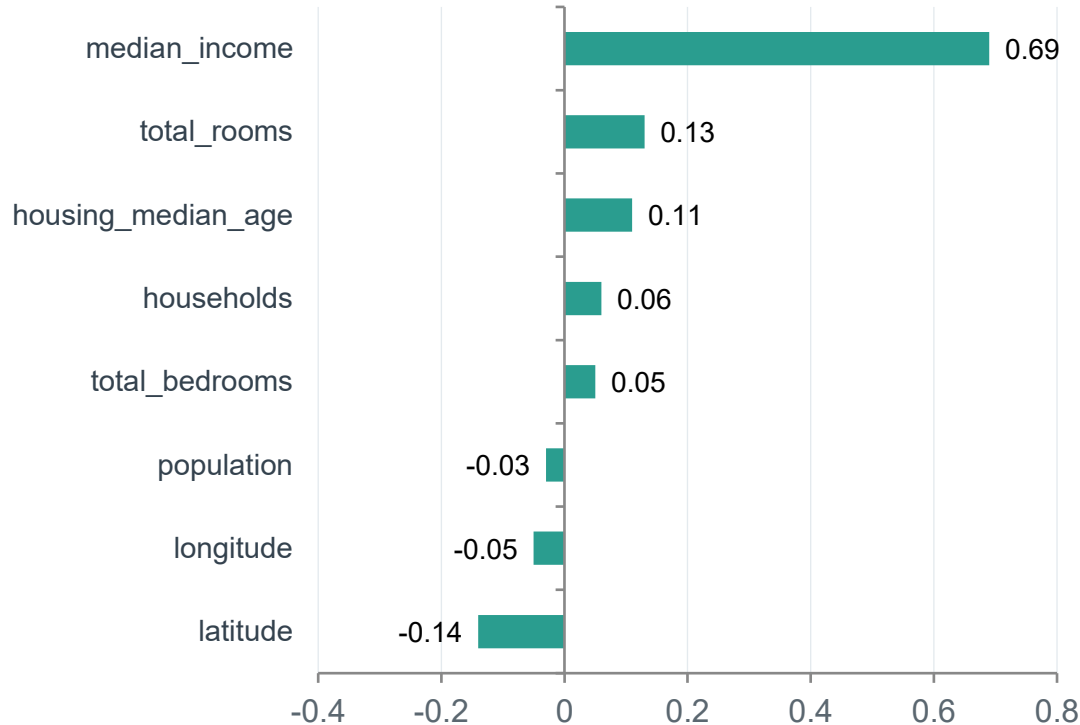
Παράδειγμα: υπολογισμός του r βήμα βήμα

x	y	$x - \bar{x}$	$y - \bar{y}$	γινόμενο	$(x - \bar{x})^2$	$(y - \bar{y})^2$
1	2	-2	-2	4	4	4
2	4	-1	0	0	1	0
3	5	0	1	0	0	1
4	4	1	0	0	1	0
5	5	2	1	2	4	1
			Σ	6	10	6

$$r = 6 / \sqrt{(10 \cdot 6)} \approx 0,77$$

Ισχυρή θετική γραμμική σχέση

Συσχέτιση με την τιμή κατοικίας



- Το εισόδημα είναι ο ισχυρότερος προγνωστικός παράγοντας
- Μικρό $r \neq$ άχρηστο: μπορεί να υπάρχει μη γραμμική σχέση
- Οριζόντιες «γραμμές» στο διάγραμμα διασποράς στα 500 χιλ. \$ (και αλλού): ύποπτα δεδομένα

Συνδυασμοί χαρακτηριστικών

Νέο χαρακτηριστικό	Ορισμός	Παράδειγμα
Δωμάτια ανά κατοικία	<code>total_rooms / households</code>	$880 / 126 \approx 6,98$
Λόγος υπνοδωματίων	<code>total_bedrooms / total_rooms</code>	$129 / 880 \approx 0,15$
Άτομα ανά κατοικία	<code>population / households</code>	$322 / 126 \approx 2,56$

Συμπέρασμα: σπίτια με μικρότερο ποσοστό υπνοδωματίων (δηλαδή περισσότερους άλλους χώρους) τείνουν να είναι ακριβότερα. Αυτή η πληροφορία «κρυβόταν» στα αρχικά χαρακτηριστικά.

ΦΑΣΗ 4

Προετοιμασία των δεδομένων

Ελλιπείς τιμές, ακραίες τιμές, κατηγορικά, κλιμάκωση, pipelines

Γιατί γράφουμε μετασχηματισμούς σε κώδικα



Αναπαραγωγιμότητα

Ξανατρέχουμε όλη τη ροή σε νέα έκδοση των δεδομένων



Παραγωγή

Ίδιος μετασχηματισμός στα νέα δεδομένα του live συστήματος



Πειραματισμός

Δοκιμάζουμε εύκολα εναλλακτικές ως υπερπαραμέτρους



Επαναχρησιμοποίηση

Βιβλιοθήκη μετασχηματισμών για επόμενα έργα

Ελλιπείς τιμές: τρεις επιλογές



Διαγραφή γραμμών

`dropna(subset=[...])`

Απλό, αλλά χάνουμε δεδομένα και ίσως εισάγουμε μεροληψία.



Διαγραφή στήλης

`drop(columns=[...])`

Μόνο αν λείπουν πολλές τιμές ή το χαρακτηριστικό είναι άχρηστο.



Συμπλήρωση (imputation)

Διάμεσος, μέση τιμή, σταθερά ή τιμή από γείτονες.

Συνήθως η προτιμώμενη λύση.

Παράδειγμα: μέση τιμή ή διάμεσος για συμπλήρωση;

Περιοχή	Υπνοδωμάτια	Με μέση τιμή	Με διάμεσο
A	3	3	3
B	5	5	5
Γ	—	28	4,5
Δ	4	4	4
E	100	100	100
Z	—	28	4,5



Μέση τιμή = 28

$$(3+5+4+100)/4$$



Διάμεσος = 4,5

Ανθεκτικός στην ακραία τιμή 100

Συμπλήρωση στο scikit-learn

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(strategy="median")
imputer.fit(housing_num)      # μαθαίνει διαμέσους
imputer.statistics_           # αποθηκευμένες τιμές
X = imputer.transform(housing_num)
```

- strategy: "mean", "median", "most_frequent", "constant"
- KNNImputer: μέσος όρος των k πλησιέστερων γειτόνων
- IterativeImputer: προβλέπει κάθε στήλη από τις υπόλοιπες

Όλοι οι μετασχηματιστές του scikit-learn έχουν την ίδια διεπαφή: fit(), transform(), fit_transform().

Ακραίες τιμές: ο κανόνας IQR

Δεδομένα: 12, 14, 15, 15, 16, 18, 19, 21, 55

14,5

Q1

20

Q3

5,5

$IQR = Q3 - Q1$

6,25 – 28,25

όρια $Q1 - 1,5 \cdot IQR$ και $Q3 + 1,5 \cdot IQR$

Το 55 είναι ακραία τιμή. Επιλογές: διαγραφή (αν είναι λάθος καταχώρισης), αποκοπή στο όριο (capping), κράτηση με ανθεκτικό μοντέλο, ή ανίχνευση με IsolationForest.

Κατηγορικά χαρακτηριστικά: ordinal ή one-hot;



Ordinal encoding

Μία στήλη με ακεραίους 0, 1, 2, ...

✓ Για διατακτικά: $S < M < L$, «κακό < μέτριο < καλό»

✗ Για ονομαστικά: εισάγει ψεύτικη σειρά

`OrdinalEncoder()`



One-hot encoding

Μία δυαδική στήλη ανά κατηγορία.

✓ Για ονομαστικά: περιοχή, χρώμα

✗ Πολλές στήλες όταν οι κατηγορίες είναι χιλιάδες

`OneHotEncoder(handle_unknown="ignore")`

Παράδειγμα: one-hot για το ocean_proximity

ocean_proximity	<1H OCEAN	INLAND	ISLAND	NEAR BAY	NEAR OCEAN
INLAND	0	1	0	0	0
NEAR BAY	0	0	0	1	0
<1H OCEAN	1	0	0	0	0
NEAR OCEAN	0	0	0	0	1

Το αποτέλεσμα είναι αραιός πίνακας (sparse): αποθηκεύονται μόνο οι θέσεις των «1», κάτι που εξοικονομεί μνήμη με χιλιάδες κατηγορίες.

Πάρα πολλές κατηγορίες (υψηλή πληθικότητα)



Αριθμητική αντικατάσταση

Ταχ. κώδικας → πληθυσμός,
εισόδημα περιοχής· εγγύτητα
σε ωκεανό → απόσταση σε km



Target encoding

Κάθε κατηγορία → μέσος όρος
του στόχου της (μόνο με
train/CV)



Embeddings

Πυκνά διανύσματα που
μαθαίνονται (νευρωνικά
δίκτυα, Διαλέξεις 9+)



Ερώτηση Slido 2 από 3



Χαρακτηριστικό «μέγεθος μπλουζάκι» με τιμές S, M, L, XL. Ποια κωδικοποίηση ταιριάζει καλύτερα;

A

One-hot encoding

B

Ordinal encoding (S=0, M=1, L=2, XL=3)

Γ

Διαγραφή του χαρακτηριστικού

Δ

Τυχαίοι ακέραιοι ανά κατηγορία

Γιατί χρειάζεται κλιμάκωση;

Άτομο	Ηλικία	Εισόδημα (€)	Απόσταση από Α
A	25	30.000	—
B	60	31.000	≈ 1.001
Γ	26	45.000	≈ 15.000



Εξαίρεση

Τα δένδρα απόφασης δεν επηρεάζονται από την κλίμακα.

Ο Β θεωρείται 15 φορές πιο «κοντινός» στον Α από τον Γ, αν και διαφέρει 35 χρόνια σε ηλικία. Αλγόριθμοι με αποστάσεις (kNN, SVM) και gradient descent επηρεάζονται έντονα.

Min-max κανονικοποίηση και τυποποίηση

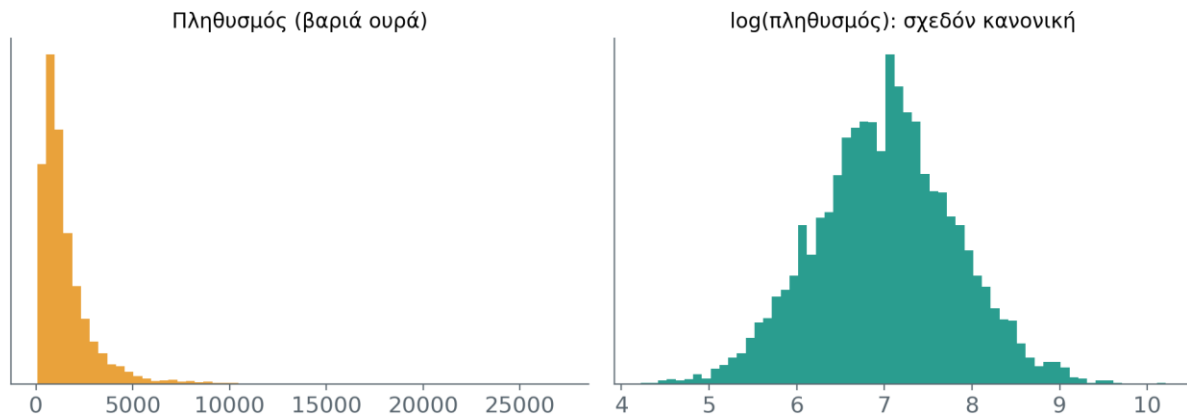
$$x' = (x - \min) / (\max - \min)$$

$$z = (x - \mu) / \sigma$$

x	10	20	30	40	100
Min-max	0	0,11	0,22	0,33	1
Τυποποίηση (z)	-0,95	-0,63	-0,32	0	1,90

- MinMaxScaler: τιμές στο [0, 1], ευαίσθητο στις ακραίες τιμές
- StandardScaler: μέση τιμή 0, τυπική απόκλιση 1, πιο ανθεκτικό

Κατανομές με βαριά ουρά: λογαριθμικός μετασχηματισμός



- Πρώτα λογάριθμος, μετά κλιμάκωση
- Για ισχυρές ουρές (power law) ο λογάριθμος είναι ιδανικός
- Εναλλακτικά: τετραγωνική ρίζα, bucketizing σε ποσοστημόρια

Παράδειγμα: $100 \rightarrow 4,6 \cdot 1.000 \rightarrow 6,9 \cdot 10.000 \rightarrow 9,2$

Νέα χαρακτηριστικά: bucketizing και ομοιότητα RBF



Bucketizing

Χωρίζουμε μια αριθμητική τιμή σε κατηγορίες και τις κωδικοποιούμε.

Παράδειγμα: ηλικία κατοικίας $\rightarrow$ «νέα / μέση / παλιά»

Χρήσιμο όταν η σχέση με τον στόχο δεν είναι μονότονη.



Ομοιότητα RBF

Πόσο κοντά είναι η τιμή σε ένα σημείο αναφοράς:
 $\exp(-\gamma \cdot (x - 35)^2)$, $\gamma = 0,1$

$$x = 35 \rightarrow 1$$

$$x = 38 \rightarrow 0,41$$

$$x = 45 \rightarrow 0,00005$$

ΦΑΣΕΙΣ 5–6

Επιλογή και βελτιστοποίηση μοντέλου

Από το πρώτο μοντέλο στο τελικό

Πρώτο μοντέλο: γραμμική παλινδρόμηση

```
from sklearn.linear_model import LinearRegression

lin_reg = make_pipeline(preprocessing,
                        LinearRegression())
lin_reg.fit(housing, housing_labels)
preds = lin_reg.predict(housing)
root_mean_squared_error(housing_labels, preds)
# ≈ 68.600
```

≈ 68.600 \$

RMSE στο training set

Υποπροσαρμογή → πιο ισχυρό μοντέλο, καλύτερα χαρακτηριστικά ή λιγότερη ομαλοποίηση.

Δεύτερο μοντέλο: δένδρο απόφασης

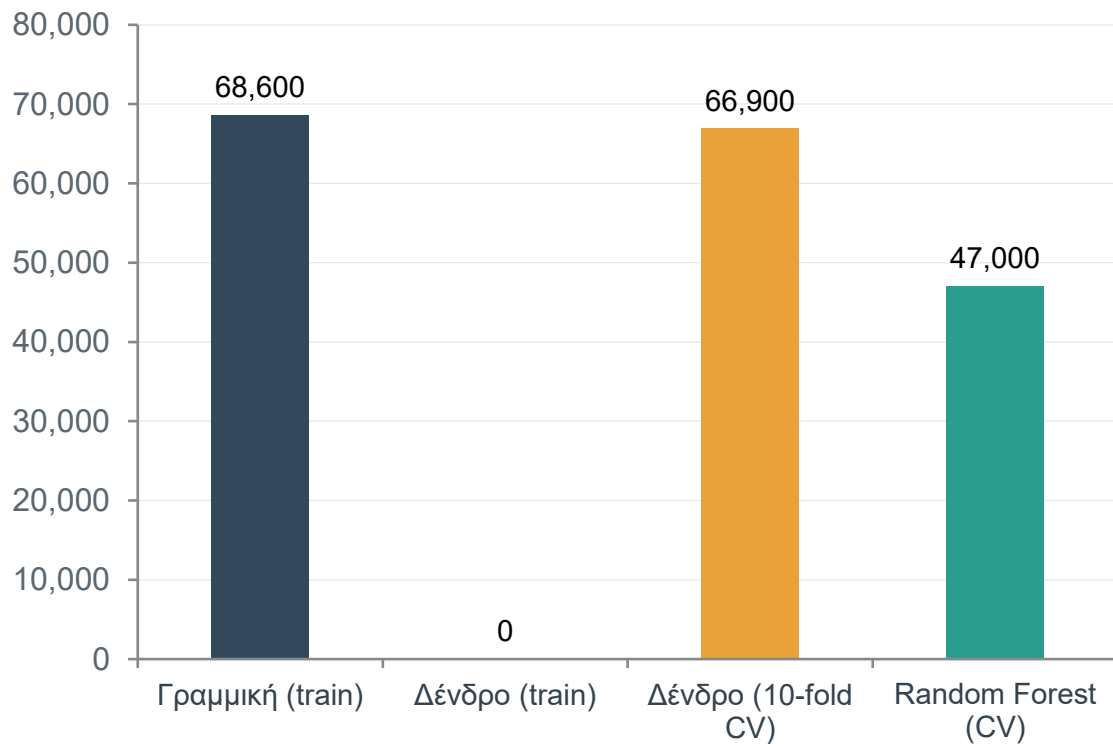
0 \$

RMSE δένδρου στο training set

Τέλειο μοντέλο ή υπερπροσαρμογή; Το ελέγχουμε με 10-fold CV, χωρίς να αγγίζουμε το test set.

```
from sklearn.model_selection import cross_val_score  
  
rmse = -cross_val_score(tree_reg, housing, housing_labels,  
                        scoring="neg_root_mean_squared_error", cv=10)
```

Σύγκριση μοντέλων με cross-validation



- Δένδρο: 0 στο train, ~66.900 στο CV → υπερπροσαρμογή
- Random Forest: το καλύτερο, αλλά και αυτό υπερπροσαρμόζεται λίγο
- Επιλέγουμε 2–5 υποσχόμενα μοντέλα για βελτιστοποίηση

Βελτιστοποίηση υπερπαραμέτρων: grid search

```
from sklearn.model_selection import GridSearchCV

param_grid = [
    {"prep__geo__n_clusters": [5, 8, 10],
     "rf__max_features": [4, 6, 8]},          # 3 × 3 = 9
    {"prep__geo__n_clusters": [10, 15],
     "rf__max_features": [6, 8, 10]}]        # 2 × 3 = 6
grid = GridSearchCV(pipeline, param_grid, cv=3,
                    scoring="neg_root_mean_squared_error")
```

15

συνδυασμοί

× 3 folds = 45 εκπαιδεύσεις

Randomized search και άλλες στρατηγικές



Grid search

Όλοι οι συνδυασμοί. Καλό για λίγες υπερπαραμέτρους.



Randomized search

Τυχαίοι συνδυασμοί από κατανομές, με σταθερό «προϋπολογισμό».



Successive halving

Πολλοί υποψήφιοι με λίγους πόρους, μετά κρατά τους καλύτερους.



Ensemble

Συνδυασμός των καλύτερων μοντέλων (Διάλεξη 7).

Ανάλυση του καλύτερου μοντέλου



Σημασία χαρακτηριστικών

Ποια χαρακτηριστικά
χρησιμοποιεί περισσότερο;
Αφαιρούμε τα άχρηστα.



Ανάλυση σφαλμάτων

Πού αποτυγχάνει; Σε ακριβές
περιοχές; Σε αγροτικές;



Δικαιοσύνη

Αποδίδει εξίσου καλά σε όλες
τις υποομάδες;

Τελική αξιολόγηση στο test set

```
final_model = grid.best_estimator_  
final_preds = final_model.predict(X_test)  
final_rmse = root_mean_squared_error(y_test, final_preds)
```



Διάστημα εμπιστοσύνης

Πόσο σίγουροι είμαστε για το RMSE; Π.χ.
διάστημα 95% με την κατανομή t ή με bootstrap.



Μην «πειράξετε» το μοντέλο

Αν το test σφάλμα απογοητεύει, δεν αλλάζουμε
υπερπαραμέτρους με βάση αυτό.

ΦΑΣΕΙΣ 7–8

Παρουσίαση, ανάπτυξη, παρακολούθηση

Το έργο δεν τελειώνει με την εκπαίδευση

Παρουσίαση και ανάπτυξη



Παρουσίαση (φάση 7)

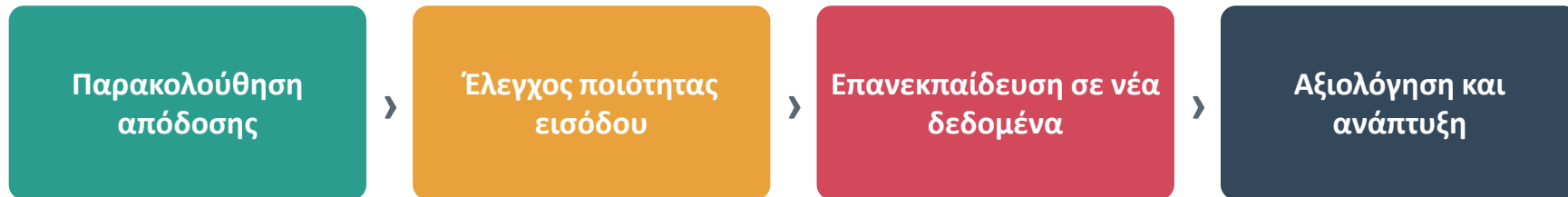
- Τι δούλεψε και τι όχι
- Υποθέσεις και περιορισμοί
- Σαφή γραφήματα, όχι πίνακες με 20 δεκαδικά
- Σύγκριση με τη σημερινή λύση (baseline)



Ανάπτυξη (φάση 8)

- Αποθήκευση μοντέλου (joblib)
- Web service / API
- Έλεγχοι ποιότητας εισόδου
- Εκδόσεις μοντέλων και δεδομένων, δυνατότητα επιστροφής σε προηγούμενο μοντέλο

Παρακολούθηση: τα μοντέλα «σαπίζουν»



Data drift

Αλλάζει η κατανομή των εισόδων (π.χ. νέες περιοχές, πληθωρισμός)



Concept drift

Αλλάζει η σχέση εισόδου–στόχου (π.χ. νέα επιτόκια)



Επιστροφή

Κρατάμε το προηγούμενο μοντέλο για γρήγορη επαναφορά

Σύνοψη

1

Πρώτα πλαισίωση: τύπος προβλήματος και μέτρο (RMSE, MAE)

2

Test set αμέσως, στρωματοποιημένο όπου χρειάζεται

3

Εξερεύνηση: ιστογράμματα, συσχετίσεις, νέοι λόγοι

4

Προετοιμασία: ελλειπίς, κατηγορικά, κλιμάκωση, λογάριθμος

5

fit μόνο στο train, όλα μέσα σε Pipeline

6

Σύγκριση με CV, βελτιστοποίηση, μία μέτρηση στο test

Για την επόμενη διάλεξη



Διάβασμα

- Géron, Κεφ. 2 (ολόκληρο, με το notebook)
- Géron, Παράρτημα A (λίστα ελέγχου)
- Géron, Κεφ. 4 (για τη Διάλεξη 3)
- ISL, Κεφ. 3



Εξάσκηση

- Τρέξτε το notebook του Κεφ. 2 στο Colab
- Δοκιμάστε MAE αντί για RMSE

Διάλεξη 3: Γραμμικά μοντέλα και βελτιστοποίηση



Ερωτήσεις;

koumakis@hmu.gr · Ώρες γραφείου: Δευτέρα 11:00–13:00