Ασκήσεις #1 — Εκφωνήσεις και οδηγίες παράδοσης

Τι ζητείται

Παραδίδετε ένα notebook (`.ipynb`) που φτιάχνετε εσείς από το μηδέν και λύνει δύο ασκήσεις πάνω σε ένα dataset από scaffolds οστικού ιστού. Κάθε απάντηση προκύπτει από κώδικα: κανένας αριθμός δεν γράφεται με το χέρι.

Ένα εργαστήριο κατασκευάζει με 3D εκτύπωση πορώδη scaffolds (ικριώματα) για την αναγέννηση οστού. Για κάθε δείγμα έχουν καταγραφεί οι παράμετροι κατασκευής και δύο αποτελέσματα που μετρήθηκαν μετά: το θλιπτικό μέτρο ελαστικότητας και το αν η ενσωμάτωση ήταν καλή. Οι δοκιμές είναι ακριβές και αργές. Μπορούμε να τα προβλέψουμε από τις παραμέτρους κατασκευής;

	Ερώτημα	Τύπος προβλήματος	Μοντέλα
Άσκηση 1	Πόσο μεγάλο θα είναι το μέτρο ελαστικότητας;	παλινδρόμηση (`compressive_modulus_MPa`)	γραμμικό, δέντρο απόφασης
Άσκηση 2	Θα ενσωματωθεί καλά το scaffold;	ταξινόμηση (`integration_good`, 0 ή 1)	δέντρο απόφασης

Οδηγίες: πώς φτιάχνετε και παραδίδετε το notebook

1. **Δημιουργία.** Ανοίξτε ένα νέο notebook στο Google Colab ή στο JupyterLab και ονομάστε το `Επώνυμο_Ασκήσεις1.ipynb`.
2. **Δομή.** Στην αρχή βάλτε κελί κειμένου με ονοματεπώνυμο και αριθμό μητρώου. Για κάθε μέρος (1.1, 1.2, …) βάλτε ένα κελί κειμένου με τον τίτλο του μέρους και, από κάτω, τα κελιά κώδικα.
3. **Βιβλιοθήκες.** Το πρώτο κελί κώδικα φορτώνει όλες τις βιβλιοθήκες που χρειάζεστε: NumPy, Pandas, Matplotlib, και από το scikit-learn το `train_test_split`, το `LinearRegression`, τα `DecisionTreeRegressor` / `DecisionTreeClassifier` / `plot_tree` και τις μετρικές `mean_absolute_error`, `accuracy_score`, `confusion_matrix`. Αν κάποια εντολή δεν βρίσκεται, κατάλαβετε ποια βιβλιοθήκη την είχε το προβλήμα.
4. **Δεδομένα.** Κατεβάστε το `scaffold_dataset.csv` από εδώ: [ΣΥΝΔΕΣΜΟΣ ΛΗΨΗΣ ΔΕΔΟΜΕΝΩΝ]. Βάλτε το στον ίδιο φάκελο με το notebook (στο Colab, μέσω του πλαϊνού μενού «Αρχεία») και φορτώστε το με Pandas.
5. **Σύνοψη.** Στο τέλος του notebook, τυπώστε με ένα κελί τις τιμές όλων των μεταβλητών που ζητούνται (με βρόχο `for` πάνω σε μια λίστα ονομάτων). Έτσι ελέγχετε ότι δεν ξεχάσατε κάτι.
6. **Έλεγχος και παράδοση.** Πριν παραδώσετε: Kernel → Restart & Run All, χωρίς σφάλματα. Υποβάλετε το `.ipynb` με τον τρόπο και μέχρι την προθεσμία που ορίζονται στη σελίδα του μαθήματος.

Κανόνες

- Κάθε απάντηση προκύπτει από κώδικα. Κανένας αριθμός δεν γράφεται με το χέρι.
- Οι μεταβλητές που ζητούνται έχουν **συγκεκριμένα ονόματα** (π.χ. `mae_lin_test`). Μην τα αλλάξετε: η βαθμολόγηση τα διαβάζει.

- Γραφήματα: όνομα και μονάδα σε κάθε άξονα, `legend` όταν υπάρχουν πολλές ομάδες.
- Όπου ζητείται «γιατί», απαντάτε με σχόλιο # μέσα στο κελί (1–2 γραμμές).
- Παντού `test_size=0.2` και `random_state=42`.
- Επιτρέπεται ό,τι έχουμε δει στα μαθήματα και η επίσημη τεκμηρίωση. Όσες εντολές δεν έχουμε δει στο μάθημα αναφέρονται ονομαστικά στην εκφώνηση· βρείτε στην τεκμηρίωση πώς λειτουργούν.

Τα δεδομένα

Το `scaffold_dataset.csv` έχει 303 γραμμές (δείγματα) και 10 στήλες.

Στήλη	Περιγραφή	Μονάδα
`sample_id`	κωδικός δείγματος	—
`lab_notebook_page`	σελίδα του εργαστηριακού τετραδίου όπου καταγράφηκε το δείγμα	—
`HA_wt_pct`	περιεκτικότητα σε υδροξυαπατίτη (HA)	wt%
`porosity_pct`	πορώδες	%
`pore_size_um`	μέσο μέγεθος πόρων	μm
`layer_height_mm`	ύψος στρώσης εκτύπωσης	mm
`print_temp_C`	θερμοκρασία εκτύπωσης	°C
`post_treatment`	επιφανειακή κατεργασία μετά την εκτύπωση: `none`, `plasma`, `NaOH`	—
`compressive_modulus_MPa`	θλιπτικό μέτρο ελαστικότητας: στόχος της Άσκησης 1	MPa
`integration_good`	1 = καλή ενσωμάτωση, 0 = όχι: στόχος της Άσκησης 2	—

Οι τιμές δεν είναι πειραματικές μετρήσεις: παράχθηκαν τεχνητά για διδακτικούς σκοπούς και δεν πρέπει να ερμηνεύονται ως φυσικοί νόμοι. Το αρχείο έχει επίτηδες τα προβλήματα που έχουν τα πραγματικά δεδομένα εργαστηρίου: κενά, αντίγραφα, ασυνεπή γραφή, λάθη μονάδων και αδύνατες τιμές. Η πρώτη δουλειά σας είναι να τα βρείτε και να τα διορθώσετε.

Άσκηση 1 — Πρόβλεψη του μέτρου ελαστικότητας (παλινδρόμηση)

Στόχος (target): `compressive_modulus_MPa`. Είσοδοι (features): οι παράμετροι κατασκευής.

1.1 Φόρτωση και έλεγχος ποιότητας (8 μονάδες)

Φορτώστε το αρχείο στη μεταβλητή `raw`, δείτε τη δομή του και υπολογίστε το συνολικό πλήθος κενών τιμών (`n_missing`). Συνεχίστε με τον έλεγχο των δεδομένων. Υπάρχουν **πέντε** είδη προβλημάτων πέρα από τα κενά. Βρείτε τα με τα εργαλεία εξέτασης του μαθήματος (περιγραφικά στατιστικά, διαφορετικές τιμές μιας στήλης, λογικές συνθήκες) και **μετρήστε** το καθένα:

Μεταβλητή	Τι μετράει
`n_duplicates`	γραμμές που είναι ακριβές αντίγραφο προηγούμενης γραμμής
`n_labels_raw`	πόσες διαφορετικές τιμές έχει η `post_treatment`, όπως είναι γραμμένες (περιμένουμε 3)
`n_bad_porosity`	δείγματα με `porosity_pct` που δεν είναι σε μονάδες % (τιμή πολύ μικρή)
`n_bad_temp`	δείγματα με αδύνατη θερμοκρασία εκτύπωσης (πολύ μεγάλη τιμή, πιθανώς κωδικός σφάλματος)
`n_bad_modulus`	δείγματα με φυσικά αδύνατο μέτρο ελαστικότητας

Αναμενόμενα: `raw.shape` = (303, 10) · `n_missing` = 18 · `n_duplicates` = 3 · `n_labels_raw` = 8 · `n_bad_porosity` = 3 · `n_bad_temp` = 3 · `n_bad_modulus` = 1.

1.2 Καθαρισμός (12 μονάδες)

Φτιάξτε το `df_clean` από το `raw`, από το οποίο ξεκινάτε πάντα, ώστε το κελί να τρέχει όσες φορές θέλετε. Για κάθε πρόβλημα που βρήκατε στο 1.1 εφαρμόστε μία διόρθωση με αιτιολόγηση:

- αφαιρέστε τα αντίγραφα (γιατί προκαλούν διαρροή δεδομένων μεταξύ train και test;)
- βάλτε την `post_treatment` σε ενιαία γραφή
- διορθώστε το πορώδες που δόθηκε σε άλλη μονάδα
- αντικαταστήστε την κωδικοποιημένη τιμή σφάλματος στη θερμοκρασία με κενή τιμή (`NaN`)
- αντικαταστήστε το αδύνατο μέτρο ελαστικότητας με `NaN`

Δεν αφαιρείτε άλλες γραμμές και δεν συμπληρώνετε ακόμη κενά. Για τα βήματα αυτά χρειάζεστε εντολές που μέχρι τώρα δεν έχουμε δει: `drop_duplicates`, οι μέθοδοι κειμένου `.str.strip()` και `.str.lower()`, και `replace`. Διαβάστε την τεκμηρίωση του Pandas.

Ζητούνται επίσης: `n_labels_clean` (πόσες διαφορετικές τιμές έχει πλέον η `post_treatment`) και `n_missing_clean` (συνολικά κενά του `df_clean`).

Αναμενόμενα: `df_clean.shape` = (300, 10) · `n_labels_clean` = 3 · `n_missing_clean` = 22. Το πορώδες κυμαίνεται πλέον από ~40 έως 85 και η θερμοκρασία εκτύπωσης από ~80 έως 120 °C.

1.3 Κενά και διερευνητική ανάλυση (8 μονάδες)

1. Φτιάξτε το `df_reg` από το `df_clean`: αφαιρέστε τις γραμμές που δεν έχουν τιμή στόχου (`dropna` με τον ορισμό `subset`) και συμπληρώστε τα υπόλοιπα κενά με τη διάμεσο κάθε στήλης. Σχόλιο #: γιατί δεν συμπληρώνουμε την τιμή-στόχο με διάμεσο; Αναμενόμενο: `df_reg.shape` = (296, 10), 0 κενά.
2. Γραφήματα με όνομα και μονάδα στους άξονες: ιστόγραμμα του `compressive_modulus_MPa` (12 bins) και scatter πορώδες (x) έναντι μέτρου (y). Πρόβλεψη πριν το τρέξετε: η σχέση είναι ευθεία ή καμπύλη;
3. Συσχετίσεις: πίνακας συσχετίσεων των αριθμητικών στηλών χωρίς την `integration_good`. `corr_target`: η συσχέτιση κάθε στήλης με το μέτρο ελαστικότητας (Series, χωρίς τη συσχέτιση

του στόχου με τον εαυτό του). `top_feature_reg`: το όνομα της στήλης με τη μεγαλύτερη απόλυτη συσχέτιση (χρησιμεύει το `abs` και το `idxmax`). Σχόλιο #: η `lab_notebook_page` έχει σχεδόν μηδενική συσχέτιση. Θα την έβαζες feature; Γιατί;

Αναμενόμενο: `top_feature_reg` = `porosity_pct` (r ≈ −0.90).

1.4 Γραμμικό μοντέλο (9 μονάδες)

1. `features`: λίστα με τα 5 features `HA_wt_pct`, `porosity_pct`, `pore_size_um`, `layer_height_mm`, `print_temp_C`. X από το `df_reg`, y το `compressive_modulus_MPa`.
2. Χωρισμός σε `X_train, X_test, y_train, y_test` (80/20, `random_state=42`).
3. Εκπαίδευση `LinearRegression` και υπολογισμός των `mae_lin_train` και `mae_lin_test`.
4. `mae_baseline`: το MAE στο test set ενός «μοντέλου» που προβλέπει για όλα τα δείγματα τη μέση τιμή του `y_train`. Δείχνει πόσο καλύτερο είναι το μοντέλο από το «τίποτα».
5. Διάγραμμα προβλεπόμενων έναντι μετρημένων τιμών (test set) με τη διαγώνιο y = x.

Σχόλιο #: γιατί δεν βάζουμε την `integration_good` στα features;

Αναμενόμενα: `X_train.shape` = (236, 5) · `mae_lin_train` ≈ 12.8 · `mae_lin_test` ≈ 13.9 · `mae_baseline` ≈ 48.7 MPa.

1.5 Δέντρο απόφασης και υπερπροσαρμογή (9 μονάδες)

Με το ίδιο split:

1. Εκπαιδεύστε `DecisionTreeRegressor(random_state=42)` χωρίς όριο βάθους: `mae_tree_full_train`, `mae_tree_full_test`, `n_leaves_full` (πλήθος φύλλων, μέθοδος `get_n_leaves`).
2. Με βρόχο `for` δοκιμάστε `max_depth` από 1 έως 10 και φυλάξτε τα MAE σε λίστες `depths`, `mae_train_list`, `mae_test_list`. Γράφημα με δύο καμπύλες (train, test) έναντι του `max_depth`.
3. `depth_ok`: το μικρότερο `max_depth` με test MAE ≤ 15 MPa. `mae_tree_ok_test`: το test MAE του δέντρου με αυτό το βάθος.

Σημείωση: επιλέγοντας το βάθος με βάση το test set, το «καταναλώνουμε»: ο έλεγχος δεν είναι πλέον ανεξάρτητος. Για αυτό θα δούμε αργότερα το cross-validation.

Αναμενόμενα: α) 0.0 · 14.8 · 236 φύλλα. β) το train MAE πέφτει συνεχώς, το test MAE σταματά να βελτιώνεται γύρω στα 14–15 MPa. γ) `depth_ok` = 4 · `mae_tree_ok_test` ≈ 14.7.

1.6 Παράγωγο feature και τελική σύγκριση (9 μονάδες)

Στα πορώδη υλικά το μέτρο ελαστικότητας δεν πέφτει γραμμικά με το πορώδες. Στη θεωρία των κυψελωτών υλικών (Gibson & Ashby) το μέτρο είναι ανάλογο του (στερεού κλάσματος)^n, με n ≈ 2 για ανοιχτά κελιά, όπου στερεό κλάσμα = 1 − πορώδες.

1. Προσθέστε στο `df_reg` τις στήλες `solid_fraction` (= 1 − `porosity_pct`/100) και `solid_fraction_sq` (το τετράγωνό του).
2. `features_v2`: ίδια 5 features, αλλά με το `solid_fraction_sq` στη θέση του `porosity_pct`. Ίδιο split.

3. Γραμμική παλινδρόμηση: `mae_lin_v2_train`, `mae_lin_v2_test`.
4. Σε ένα dictionary βάλτε το test MAE των τριών μοντέλων: `"linear"` (1.4), `"tree"` (δέντρο με `depth_ok`) και `"linear_v2"`. `best_model`: το όνομα του μοντέλου με το μικρότερο test MAE. `mae_best_pct`: το MAE του καλύτερου ως % της μέσης τιμής του `y`. (Χρησιμεύει η συνάρτηση `min` με τον ορισμό `key`.)

Σχόλιο `#`: τι σημαίνει στην πράξη το `mae_best_pct` για έναν μηχανικό που θέλει να επιλέξει scaffold;

Αναμενόμενα: `mae_lin_v2_train` ≈ 11.4 · `mae_lin_v2_test` ≈ 12.3 · `best_model` = `linear_v2` · `mae_best_pct` ≈ 14%.

Άσκηση 2 — Θα ενσωματωθεί καλά το scaffold; (ταξινόμηση)

Ο στόχος είναι τώρα μια κλάση: `integration_good` = 1 ή 0. Χρησιμοποιείτε τον `DecisionTreeClassifier`, με το ίδιο API με τον `DecisionTreeRegressor`. Αντί για MAE, μετράτε **accuracy** (το ποσοστό των σωστών προβλέψεων). Για τη μήτρα σύγχυσης με θετική κλάση το 1: TN σωστά «όχι», FP είπαμε «ναι» ενώ ήταν «όχι», FN είπαμε «όχι» ενώ ήταν «ναι», TP σωστά «ναι». Για τα υπόλοιπα (`accuracy_score`, `confusion_matrix`, `plot_tree`, `feature_importances_`) δείτε την τεκμηρίωση του scikit-learn.

2.1 Δεδομένα και baseline (6 μονάδες)

1. `df_cls`: ξεκινήστε από το `df_clean` και συμπληρώστε τα κενά με τη διάμεσο. Κρατάτε και τις 300 γραμμές. Σχόλιο `#`: γιατί δεν χρειάζεται να πετάξουμε εδώ τις 4 γραμμές χωρίς μέτρο ελαστικότητας;
2. Τα ίδια 5 features (`features` του 1.4) και στόχος το `integration_good`. Split 80/20 με `random_state=42` στα `X_train_c, X_test_c, y_train_c, y_test_c`.
3. `frac_good`: το ποσοστό των δειγμάτων με `integration_good` = 1 στο `df_cls`.
4. `acc_baseline`: η accuracy στο test set ενός «μοντέλου» που προβλέπει πάντα την πιο συχνή κλάση του `y_train_c`. Κάθε χρήσιμο μοντέλο πρέπει να το ξεπερνά.

Αναμενόμενα: `X_train_c.shape` = (240, 5) · `frac_good` = 0.53 · `acc_baseline` ≈ 0.467.

2.2 Διερευνητική ανάλυση (6 μονάδες)

Scatter με x = `pore_size_um` και y = `porosity_pct`, ένα χρώμα ανά κλάση, με `legend` και άξονες με μονάδες (όπως στο γράφημα ανά ομάδα που είδαμε στο μάθημα). `separates`: το όνομα του άξονα (`"pore_size_um"` ή `"porosity_pct"`) κατά μήκος του οποίου οι δύο κλάσεις ξεχωρίζουν καλύτερα. Πρόβλεψη πριν το τρέξετε: πού θα δείτε διαχωρισμό;

2.3 Βάθος δέντρου και υπερπροσαρμογή (10 μονάδες)

1. Δέντρο χωρίς όριο βάθους: `acc_full_train`, `acc_full_test`, `n_leaves_c_full`.
2. Με βρόχο `for`, `max_depth` από 1 έως 10: λίστες `depths_c`, `acc_train_list`, `acc_test_list`. Γράφημα με τις δύο καμπύλες και μια οριζόντια γραμμή για το `acc_baseline`.
3. `best_depth_cls`: το μικρότερο βάθος που δίνει την υψηλότερη test accuracy.

Σχόλιο `#`: τι συμβαίνει στις δύο καμπύλες για μεγάλα βάθη; Πώς λέμε αυτό το φαινόμενο;

Αναμενόμενα: α) 1.0 · 0.783 · 46 φύλλα. β) το βάθος 1 δεν ξεπερνά το baseline· από βάθος 2 η test accuracy φτάνει ≈ 0.80 και δεν βελτιώνεται πλέον, ενώ η train accuracy ανεβαίνει προς το 1. γ) `best_depth_cls` = 2.

2.4 Μήτρα σύγχυσης (8 μονάδες)

Για να διαβάζεται το δέντρο στο 2.5, δουλέψτε με `max_depth=3`.

1. `clf3`: δέντρο ταξινόμησης με `max_depth=3`, `random_state=42`, εκπαιδευμένο στο train set. `y_pred_c`: οι προβλέψεις στο test set. `acc_3`: η accuracy.
2. `cm`: η μήτρα σύγχυσης· ξεπακετάρετε τα 4 νούμερά της σε `tn, fp, fn, tp`.
3. `more_frequent_error`: `"fp"` ή `"fn"`, ποιο λάθος κάνει το μοντέλο συχνότερα;

Σχόλιο `#` (1–2 γραμμές): σε αυτό το εργαστήριο, ποιο λάθος είναι πιο δαπανηρό και γιατί; Υπάρχουν δύο υπερασπίσιμες απαντήσεις· χρειάζεται αιτιολόγηση.

Αναμενόμενα: `acc_3` = 0.80 · `tn, fp, fn, tp` = 28, 4, 8, 20 · `more_frequent_error` = `"fn"`.

2.5 Ερμηνεία του δέντρου (10 μονάδες)

1. Σχεδιάστε το `clf3` με `plot_tree` (μεγάλο μέγεθος γραφήματος, ονόματα features, χρωματισμός των κόμβων).
2. Bar chart της σημασίας των features (`feature_importances_`) με ονόματα στον άξονα. `top_feature_cls`: το feature με τη μεγαλύτερη σημασία.
3. Από το γράφημα του δέντρου διάβαστε δύο όρια του μεγέθους πόρων: `pore_low` (η απόφαση στη ρίζα) και `pore_high` (η απόφαση του δεξιού κλάδου). Ευνοείται η ενσωμάτωση για `pore_low < pore_size_um ≤ pore_high`.
4. Επαλήθευση με Pandas (όχι με το δέντρο): μάσκα `in_window` για τα δείγματα μέσα στο παράθυρο και υπολογισμός των `rate_inside` και `rate_outside`, δηλαδή του ποσοστού `integration_good` = 1 μέσα και έξω από το παράθυρο. (Για την άρνηση μάσκας χρησιμεύει ο τελεστής ~.)

Σχόλιο `#`: τι «κανόνα σχεδιασμού» προτείνει το δέντρο στον μηχανικό; Μην ξεχνάτε ότι τα δεδομένα είναι συνθετικά.

Αναμενόμενα: `top_feature_cls` = `pore_size_um` (σημασία ≈ 0.79) · `pore_low` ≈ 250 · `pore_high` ≈ 507 · `rate_inside` ≈ 0.84 · `rate_outside` ≈ 0.23.

Bonus (προαιρετικό, έως +5 μονάδες)

Η `post_treatment` είναι κείμενο και δεν μπήκε στο μοντέλο. Κρατήστε μόνο τα δείγματα μέσα στο παράθυρο πόρων. Για κάθε τιμή της `post_treatment` υπολογίστε το ποσοστό `integration_good` = 1 και φυλάξτε το σε ένα dictionary `rate_in_by_treatment` με κλειδιά `"none"`, `"naoh"`, `"plasma"`. Σχόλιο `#`: ποιο συμπέρασμα βγάζετε; Πόσα δείγματα έχει κάθε ομάδα και πόσο σίγουροι μπορείτε να είστε για τις διαφορές;

Αναμενόμενα: περίπου `none` 0.78 · `naoh` 0.83 · `plasma` 0.95.

Βαθμολόγηση και έλεγχος πριν την παράδοση

Σύνολο 100 μονάδες, και έως +5 με το bonus.

Μέρος	Μονάδες
1.1 Φόρτωση και έλεγχος ποιότητας	8
1.2 Καθαρισμός	12
1.3 Κενά και διερευνητική ανάλυση	8
1.4 Γραμμικό μοντέλο	9
1.5 Δέντρο και υπερπροσαρμογή	9
1.6 Παράγωγο feature και σύγκριση	9
2.1 Δεδομένα και baseline	6
2.2 Διερευνητική ανάλυση	6
2.3 Βάθος δέντρου και υπερπροσαρμογή	10
2.4 Μήτρα σύγχυσης	8
2.5 Ερμηνεία του δέντρου	10
Ποιότητα: τρέχει από την αρχή έως το τέλος, άξονες με μονάδες, σχόλια	5
Bonus (προαιρετικό)	+5

Λίστα ελέγχου

- ☐ Το πρώτο κελί κώδικα φορτώνει τις βιβλιοθήκες και το notebook έχει ονοματεπώνυμο και αριθμό μητρώου.
- ☐ Κάθε μεταβλητή της εκφώνησης έχει το ακριβές όνομά της.
- ☐ Οι τιμές συγκρίνονται με τα «Αναμενόμενα»· όπου διαφέρουν, ελέγξτε τα βήματα καθαρισμού και το `random_state`.
- ☐ Κάθε γράφημα έχει όνομα και μονάδα σε κάθε άξονα.
- ☐ Τα σχόλια «γιατί» είναι γραμμένα μέσα στους κώδικες.
- ☐ Το τελευταίο κελί τυπώνει όλες τις μεταβλητές των ασκήσεων.
- ☐ Kernel → Restart & Run All χωρίς σφάλματα.