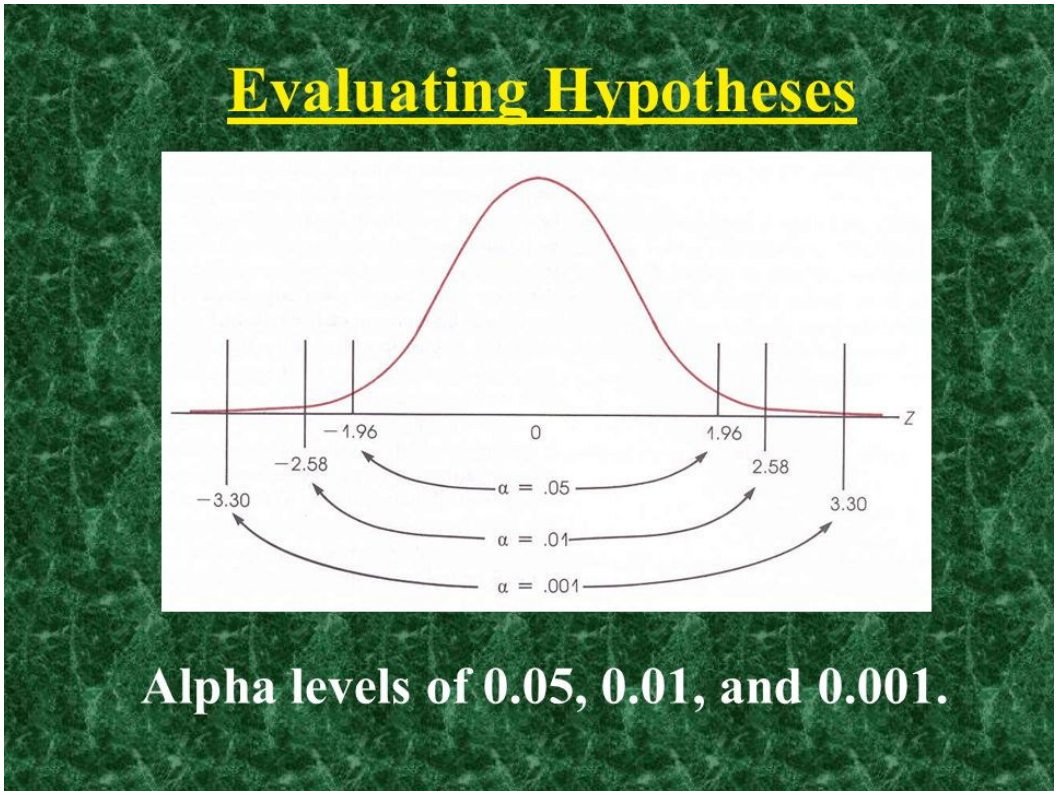
Κεφάλαιο 6

Μηχανική Μάθηση



Αξιολόγηση Μοντέλων
Evaluating Hypotheses

Εισαγωγή

- Για τη βελτίωση των μοντέλων απαιτούνται μηχανισμοί αξιολόγησής τους
- Το πρόβλημα της αξιολόγησης έχει ενδιαφέρουσες φιλοσοφικές προεκτάσεις.
  - Για χιλιετίες, οι φιλόσοφοι μελετούν το θέμα της αξιολόγησης επιστημονικών θεωριών.
  - Τελικά, τα εξαγόμενα μοντέλα είναι μια "θεωρία" των δεδομένων.
- Όπως έχουμε δει ως τώρα και θα δούμε και παρακάτω, υπάρχουν διαφόρων ειδών μοντέλα που μπορούμε να “μάθουμε” από δεδομένα (π.χ. γραμμική παρεμβολή, δένδρα απόφασης, KNN, Bayes, SVM, κ.λ.π.)
  - Έχοντας ένα πρόβλημα μάθησης, ποιο τελικά μοντέλο πρέπει να επιλέξουμε;
- Ερωτήματα:
  - Έχοντας ένα σύνολο εκπαίδευσης, μπορούμε να εξετάσουμε την απόδοση διαφόρων μοντέλων στο σύνολο αυτό ή σε ένα ξεχωριστό σύνολο δεδομένων;
  - Τι κάνουμε όταν δεν έχουμε πολλά διαθέσιμα δεδομένα για την κατασκευή του μοντέλου και στη συνέχεια για την αξιολόγηση του;
- Έχουν προταθεί διάφορες μέθοδοι και μετρικές αξιολόγησης των μοντέλων αλλά και στατιστικά τεστ που δείχνουν αν οι τυχόν διαφορές στην απόδοση είναι τυχαίες ή όχι.

Outline

Α) Μέθοδοι Αξιολόγησης/Επαλήθευση Μοντέλων

Β) Πολυπλοκότητα μοντέλου (bias-variance)

Γ) Πρόβλεψη Απόδοσης

Δ) Στατιστικά τεστ σύγκρισης αλγορίθμων

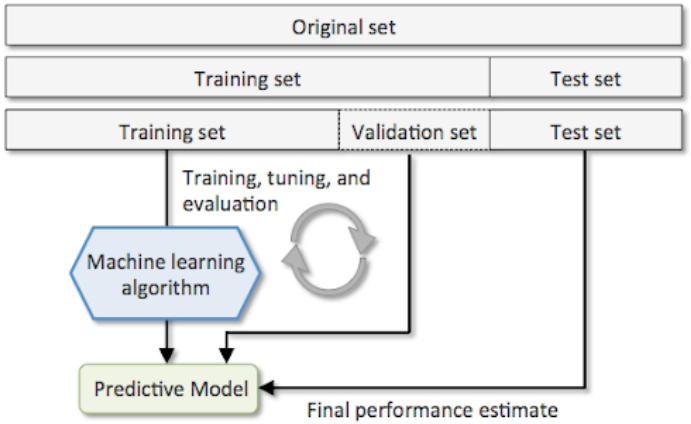
Ε) Μετρικές Αξιολόγησης Ταξινόμησης

Ζ) Μετρικές Αξιολόγησης Παρεμβολής

Η) Ασκήσεις

Α) Μέθοδοι Αξιολόγησης/Επαλήθευση Μοντέλων

- Αφού εκπαιδεύσουμε ένα μοντέλο πρόβλεψης από ένα σύνολο δεδομένων στη συνέχεια πρέπει να το αξιολογήσουμε υπολογίζοντας την απόδοση του σε ένα μικρότερο σύνολο που ονομάζεται *σύνολο δοκιμής* (*test set*).
  - Για λόγους αξιοπιστίας χρειαζόμαστε ένα νέο σύνολο δεδομένων, που **δεν συμμετείχε σε κανένα στάδιο της εκπαίδευσης του μοντέλου**.
- Η απόδοση του μοντέλου στα δεδομένα εκπαίδευσης μας ενδιαφέρει σε ειδικές περιπτώσεις,
  - όταν π.χ. το ζητούμενο είναι ο καθαρισμός των δεδομένων (*data cleaning*), η διόρθωση δηλαδή των τιμών ορισμένων πεδίων σε κάποια παραδείγματα εκπαίδευσης και όχι η πρόβλεψη.
  - Το σφάλμα στα δεδομένα εκπαίδευσης ονομάζεται επαναντικατάστασης (*resubstitution error*)
    - ✓ Αν και δεν είναι αξιόπιστη ένδειξη είναι χρήσιμο να το γνωρίζουμε



- Σε πολλές περιπτώσεις, χρησιμοποιείται και ένα τρίτο σύνολο δεδομένων, το *σύνολο επικύρωσης* (*validation set*)
  - χρησιμοποιείται για τη βελτιστοποίηση, π.χ. κλάδεμα, ρύθμιση παραμέτρων (hyperparameters tuning) του μοντέλου.
- Θεωρούμε πάντοτε ότι τα σύνολα εκπαίδευσης και δοκιμής αποτελούν αντιπροσωπευτικά δείγματα των δεδομένων του προβλήματος.
- Δύο είναι οι κύριες τεχνικές αξιολόγησης των μοντέλων.
  - Η *παρακράτηση* (*holdout*)
  - *Διασταυρωμένη επικύρωση* (*cross validation*)

Παρακράτηση (Holdout)

- Από το σύνολο των διαθέσιμων δεδομένων, ένα μέρος τους παρακρατείται για την εκπαίδευση του μοντέλου (σύνολο εκπαίδευσης) και το υπόλοιπο χρησιμοποιείται για τη δοκιμή του μοντέλου (σύνολο δοκιμής).
  - Η διαδικασία αυτή ονομάζεται διαδικασία παρακράτησης (*holdout procedure*).
  - Όπου απαιτείται, παρακρατείται και ένα μέρος για επικύρωση (validation).
  - Υπάρχει όμως ένα δίλημμα: πως θα χωρίσουμε τα υπάρχοντα δεδομένα όταν χρειάζεται ένα μεγάλο Σύνολο Εκπαίδευσης για τη δημιουργία ενός καλού μοντέλου, αλλά και ένα μεγάλο Σύνολο Δοκιμής για τον ακριβέστερο υπολογισμό της πιθανότητας σφάλματος;
  - **Στην πράξη, συχνά χρησιμοποιούνται τα 2/3 για την εκπαίδευση και το 1/3 για τη δοκιμή**.
    - ✓ sklearn.model_selection.train_test_split
    - ✓ Hold-out Method for Training Machine Learning Models
- Βέβαια, μπορεί να είμαστε άτυχοι στην επιλογή και τα σύνολα δοκιμής και εκπαίδευσης να μην είναι αντιπροσωπευτικά.
  - Μπορεί δηλαδή, οι κατηγορίες να μην εμφανίζονται με το ίδιο ποσοστό στα 2 σύνολα.
  - Μπορεί ακόμα και να μην υπάρχουν καθόλου παραδείγματα της μιας κατηγορίας σε ένα σύνολο!
  - Για το σκοπό αυτό συχνά χρησιμοποιείται η μέθοδος της διαστρωματωμένης παρακράτησης (stratified holdout), σύμφωνα με την οποία τα δεδομένα μοιράζονται με τέτοιο τρόπο ώστε κάθε κατηγορία να αντιπροσωπεύεται και στα δύο σύνολα από το ίδιο ποσοστό δεδομένων.
    - ✓ Στην Python: *train_test_split(X, y, stratify=y, test_size=0.3)*

Επαναληπτική Παρακράτηση (Repeated Holdout)

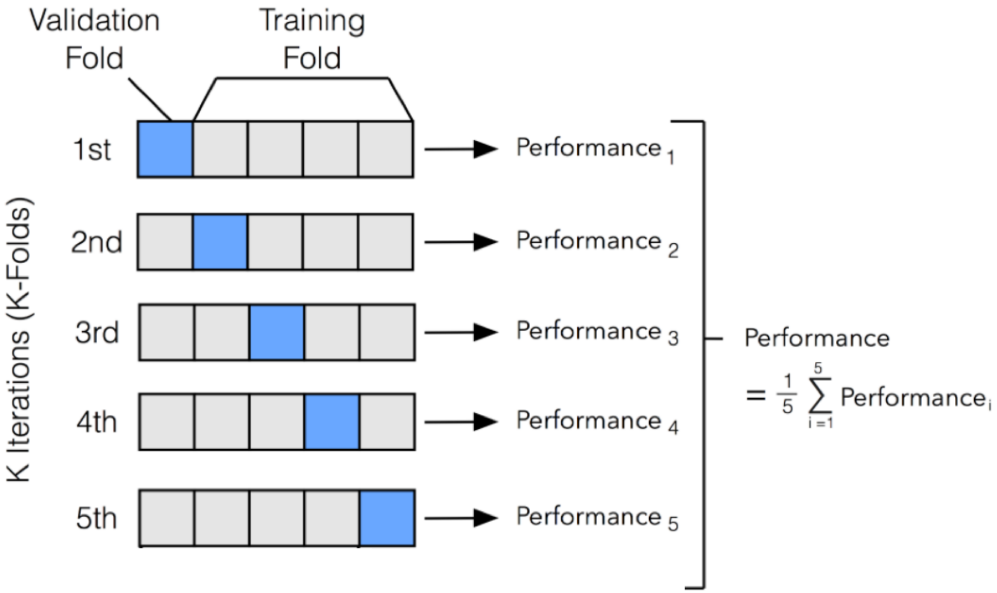
- Στην *επαναληπτική παρακράτηση* (*repeated holdout*), η διαδικασία επιλογής συνόλων εκπαίδευσης και δοκιμής καθώς και η εκπαίδευση του μοντέλου η δοκιμή του γίνεται πολλές φορές και τελικά υπολογίζεται ο μέσος όρος της απόδοσης.
  - Αυτός ο μέσος όρος είναι ουσιαστικά η εκτίμηση της απόδοσης του τελικού μοντέλου που θα προκύψει από εκπαίδευση σε ολόκληρο το σύνολο δεδομένων.
  - Το πρόβλημα στη διαδικασία της επαναληπτικής παρακράτησης είναι ότι τα σύνολα δοκιμής αλληλεπικαλύπτονται.

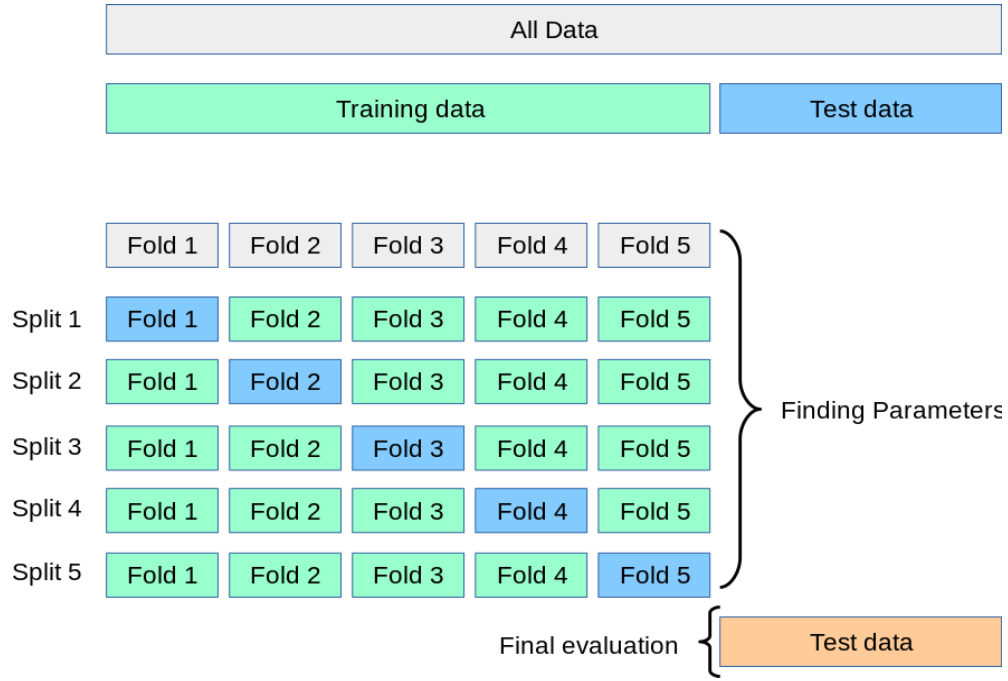
- Περιορισμένο Σύνολο Δεδομένων
  - Όταν υπάρχει πληθώρα δεδομένων, μπορούμε εύκολα να φτιάξουμε ένα μοντέλο με ένα μεγάλο σύνολο εκπαίδευσης και να το αξιολογήσουμε με ένα επίσης μεγάλο σύνολο δοκιμής.
  - Αυτό όμως δεν είναι πάντοτε εφικτό, ειδικά όταν τα δεδομένα εκπαίδευσης δεν είναι τόσα πολλά, όπως π.χ. όταν αυτά δεν προϋπάρχουν αλλά παράγονται από έναν ειδικό.
- Το πρόβλημα της πρόβλεψης της απόδοσης ενός μοντέλου όταν δεν υπάρχουν αρκετά δεδομένα για εκπαίδευση και δοκιμή είναι ιδιαίτερα σημαντικό και ακόμα αμφιλεγόμενο.
  - Υπάρχουν διάφορες τεχνικές, όμως η πιο διαδεδομένη και γενικά αποδεκτή είναι η *διασταυρωμένη επικύρωση* (*cross-validation*).
  - Στην Python: cross_val_score(estimator, X)

Διασταυρωμένη επικύρωση (Cross-Validation)

- Το πρόβλημα της αλληλοεπικάλυψης των συνόλων δοκιμής καθώς και το πρόβλημα των περιορισμένων δεδομένων αντιμετωπίζεται με τη μέθοδο της διασταυρωμένης επικύρωσης (cross-validation).
  - Βήμα 1: τα δεδομένα χωρίζονται σε k υποσύνολα ίσου μεγέθους
  - Βήμα 2: ένα υποσύνολο χρησιμοποιείται για τη δοκιμή του μοντέλου (test set) που εκπαιδεύεται από τα υπόλοιπα k-1 υποσύνολα (training set).
  - Επαναλαμβάνουμε το βήμα 2, k φορές, διαλέγοντας διαφορετικό υποσύνολο δοκιμής κάθε φορά.
  - Το τελικό ποσοστό σφάλματος είναι ο μέσος όρος των ποσοστών των k επαναλήψεων.
  - Όπως και στην επαναληπτική παρακράτηση, ο παραπάνω μέσος όρος αποτελεί μία εκτίμηση της απόδοσης του μοντέλου που θα προκύψει από εκπαίδευση σε ολόκληρο το σύνολο δεδομένων.
- Η μέθοδος αυτή ονομάζεται *k-fold cross-validation*
  - ✓ Στην Python `cross_val_score(estimator, X,cv=5)`
  - Συχνά τα υποσύνολα επιλέγονται με τη μέθοδο της διαστρωμάτωσης (stratification).
- Η πιο διαδεδομένη μέθοδος αξιολόγησης είναι η *10-fold cross-validation*.
  - Το k=10 αποδείχθηκε πειραματικά ότι είναι το καλύτερο για μια ακριβέστερη εκτίμηση.
  - Η βέλτιστη λύση είναι η επαναληπτική διαστρωματωμένη διασταύρωση (stratified k-fold cross-validation).
    - ✓ Το `cross_val_score` είναι by default stratified

Διασταύρωση (Cross-Validation) (συνεχ.)





Διασταύρωση (Cross-Validation) (συνεχ.)

- Μέθοδος "άφησε ένα έξω" (*leave one out*) cross-validation
  - Ουσιαστικά αποτελεί παραλλαγή της μεθόδου επικύρωσης k τμημάτων.
  - Στην παραλλαγή αυτή το k=n, όπου n είναι το πλήθος του συνόλου δεδομένων.
  - Για κάθε ένα παράδειγμα, το μοντέλο εκπαιδεύεται χρησιμοποιώντας τα υπόλοιπα n-1 (training set) και επικυρώνεται με αυτό που αφήσαμε έξω.
  - Η διαδικασία επαναλαμβάνεται n φορές. Στο τέλος υπολογίζεται το ποσοστό ορθών προβλέψεων.
  - LeaveOneOut()
- *Leave-p-out* cross-validation
  - Χρησιμοποιούνται p παραδείγματα σαν σύνολο επικύρωσης (validation set) και τις υπόλοιπες (n-p) σαν δεδομένα εκπαίδευσης (training set).
  - Επαναλαμβάνεται με όλους τους συνδυασμούς
  - Difference with K-fold: leave-p-out is exhaustive, k-fold is not. For example:
    - ✓ leave-5-out for 50 samples means that will have 2.118.760 iterations (all possible 5 elements are, in turn, used as validation set).
    - ✓ 5-fold instead is only 5 iterations (the data is split into five equally-sized blocks and each block is, in turn, used as validation set).
    - ✓ That's why the latter is usually preferred - but if it's worth it on theoretical grounds, I'm happy to take the performance hit (leave-2-out most likely, though)
  - LeavePOut(p)

Διασταύρωση (Cross-Validation) (συνεχ.)

- Repeated random sub-sampling validation
  - Η μέθοδος, γνωστή και ως Monte Carlo cross-validation, χωρίζει τυχαία το dataset σε σύνολα εκπαίδευσης (training) και επικύρωσης (validation data).
  - For each split, the model is fit to the training data, and predictive accuracy is assessed using the validation data.
  - The results are then averaged over the splits.
  - The advantage of this method (over k-fold cross validation) is that the proportion of the training/validation split is not dependent on the number of iterations (folds).
  - The disadvantage of this method is that some observations may never be selected in the validation subsample, whereas others may be selected more than once.
    - ✓ In other words, validation subsets may overlap.
    - ✓ As the number of random splits approaches infinity, the result of repeated random sub-sampling validation tends towards that of leave-p-out cross-validation.
  - ShuffleSplit(n_splits, train_size, test_size)
- Μέθοδος bootstrap
  - Δημιουργούνται πάλι πολλαπλά σύνολα επικύρωσης με δειγματοληψία αλλά με επανατοποθέτηση
    - ✓ Δηλ. κάθε παρατήρηση που επιλέγεται δεν αφαιρείται από το αρχικό δείγμα
    - ✓ Οπότε μια παρατήρηση μπορεί να επιλεγεί περισσότερες από μία φορές για να συμμετάσχει στο ίδιο σύνολο επικύρωσης ή καμία.

Nested Cross Validation

- Nested cross-validation is an approach to model hyperparameter optimization and model selection that attempts to overcome the problem of overfitting the training dataset.
  - As such, the k-fold cross-validation procedure for model hyperparameter optimization is nested inside the k-fold cross-validation procedure for model selection.
- Typically, the k-fold cross-validation procedure involves fitting a model on all folds but one (train dataset) and evaluating the fit model on the holdout fold (test dataset).
  - Each training dataset is then provided to a hyperparameter optimized procedure, such as grid search or random search, that finds an optimal set of hyperparameters for the model.
- The evaluation of each set of hyperparameters is performed using k-fold cross-validation that splits up the provided train dataset into k folds, not the original dataset.
  - Under this procedure, hyperparameter search does not have an opportunity to overfit the dataset as it is only exposed to a subset of the dataset provided by the outer cross-validation procedure.
- Cost of Nested Cross-Validation
  - If n * k models are fit and evaluated as part of a traditional cross-validation hyperparameter search for a given model, then this is increased to k * n * k as the procedure is then performed k more times for each fold in the outer loop of nested cross-validation
    - ✓ For example if you use k=5 for the hyperparameter search and test 100 combinations of model hyperparameters. A traditional hyperparameter search would, therefore, fit and evaluate 5 * 100 or 500 models. Nested cross-validation with k=10 folds in the outer loop would fit and evaluate 5,000 models.
    - ✓ Source: https://machinelearningmastery.com/nested-cross-validation-for-machine-learning-with-python/

Nested Cross Validation (cont…)

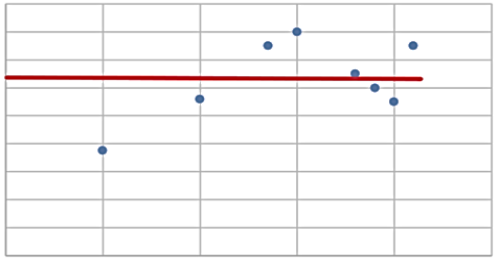
- 10-fold cross-validation: Train_i, Test_i
- Αν ο αλγόριθμος έχει υπερ-παράμετρο t τότε με χρήση εσωτερικού 10-fold cross-validation: Train_ij, Test_ij
  - Εκπαίδευση αλγορίθμου στο Train_ij με κάθε τιμή της tκαι δοκιμή στο Test_ij που δίνει σφάλμα e_{ijt}
- Επιλογή t'με ελάχιστο $e_{i1t}+e_{i2t}+\cdots e_{i10t}$
- Εκπαίδευση αλγορίθμου στο Train_i με t'και δοκιμή στο Test_i που δίνει σφάλμα e_i
- Εκτίμηση του συνολικού σφάλματος ως $(e_1+\cdots+e_{10})/10$

Β) Πολυπλοκότητα μοντέλου (bias-variance)

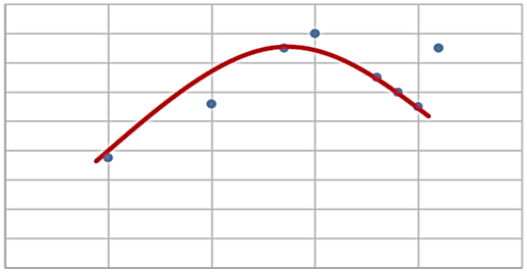
- Οι αλγόριθμοι μάθησης έχουν μία παράμετρο που καθορίζει την *πολυπλοκότητα* (*complexity*) των μοντέλων που παράγουν
  - Π.χ. το βάθος στα δένδρα απόφασης, ο αριθμός των επιπέδων στα νευρωνικά δίκτυα, ο βαθμός του πολυωνύμου στην πολυωνυμική παρεμβολή, κλπ.
- Η πολυπλοκότητα ενός μοντέλου έχει άμεση επίδραση στην απόδοσή του καθώς επηρεάζει τις δύο βασικές πηγές σφάλματος για ένα μοντέλο, την:
  - *Μεροληψία* (*bias*): Η συστηματική απόκλιση των προβλέψεων του μοντέλου από τις σωστές τιμές.
    - ✓ Υψηλή μεροληψία προκαλείται όταν **το μοντέλο είναι πολύ απλό** και αδυνατεί να προσεγγίσει επαρκώς τη συσχέτιση μεταξύ των ανεξάρτητων και της εξαρτημένης μεταβλητής και συνεπώς υποπροσαρμόζεται στα δεδομένα και τελικά παρουσιάζει κακή επίδοση (λανθασμένες προβλέψεις) ακόμη και στα δεδομένα εκπαίδευσης.
  - *Διακύμανση* (*variance*): Η ευαισθησία του μοντέλου σε μικρές αλλαγές στο σύνολο εκπαίδευσης
    - ✓ Υψηλή διακύμανση έχουμε όταν το μοντέλο παρουσιάζει πολύ καλή απόδοση στα δεδομένα εκπαίδευσης αλλά πολύ κακή σε άγνωστα δεδομένα (που ουσιαστικά αυτό μας ενδιαφέρει).
    - ✓ Προκαλείται όταν **το μοντέλο παρουσιάζει μεγάλη πολυπλοκότητα** και υπερπροσαρμόζεται στα δεδομένα εκπαίδευσης, λαμβάνοντας υπόψη του (δηλ. μοντελοποιώντας) ακόμη και αυτά που είναι σπάνια ή περιέχουν θόρυβο (δηλ. λάθος τιμές).

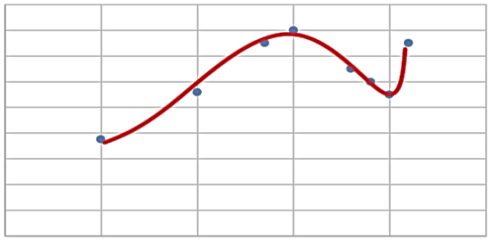
Πολυπλοκότητα μοντέλου (bias-variance) (συνεχ.)

- Στο Σχήμα παρουσιάζεται γραφικά η αναπαράσταση τριών μοντέλων από τα οποία
  - το πρώτο υποπροσαρμόζεται στα δεδομένα,
  - το δεύτερο (στο κέντρο) προσαρμόζεται ικανοποιητικά ενώ
  - το τρίτο υπερπροσαρμόζεται στα δεδομένα.



- Στη μάθηση με επίβλεψη:
  - η υποπροσαρμογή συμβαίνει όταν ένα μοντέλο δεν είναι ικανό να μοντελοποιήσει την κρυμμένη σχέση στα δεδομένα.
    - ✓ Τέτοια μοντέλα έχουν συνήθως **υψηλή μεροληψία και χαμηλή διακύμανση**.
    - ✓ Αυτό συμβαίνει όταν δεν υπάρχουν επαρκή δεδομένα για να φτιαχτεί ένα ακριβές μοντέλο ή όταν επιχειρούμε να φτιάξουμε ένα απλό μοντέλο με δεδομένα που συσχετίζονται με σύνθετο τρόπο
    - ✓ π.χ. ένα γραμμικό μοντέλο για μη γραμμικά δεδομένα (βλ. και το Σχήμα στην κορυφή).
    - ✓ Τέτοια μοντέλα (όπως π.χ. αυτά της γραμμικής και λογιστικής παρεμβολής) είναι πολύ απλά για να μοντελοποιήσουν σύνθετες σχέσεις σε δεδομένα.

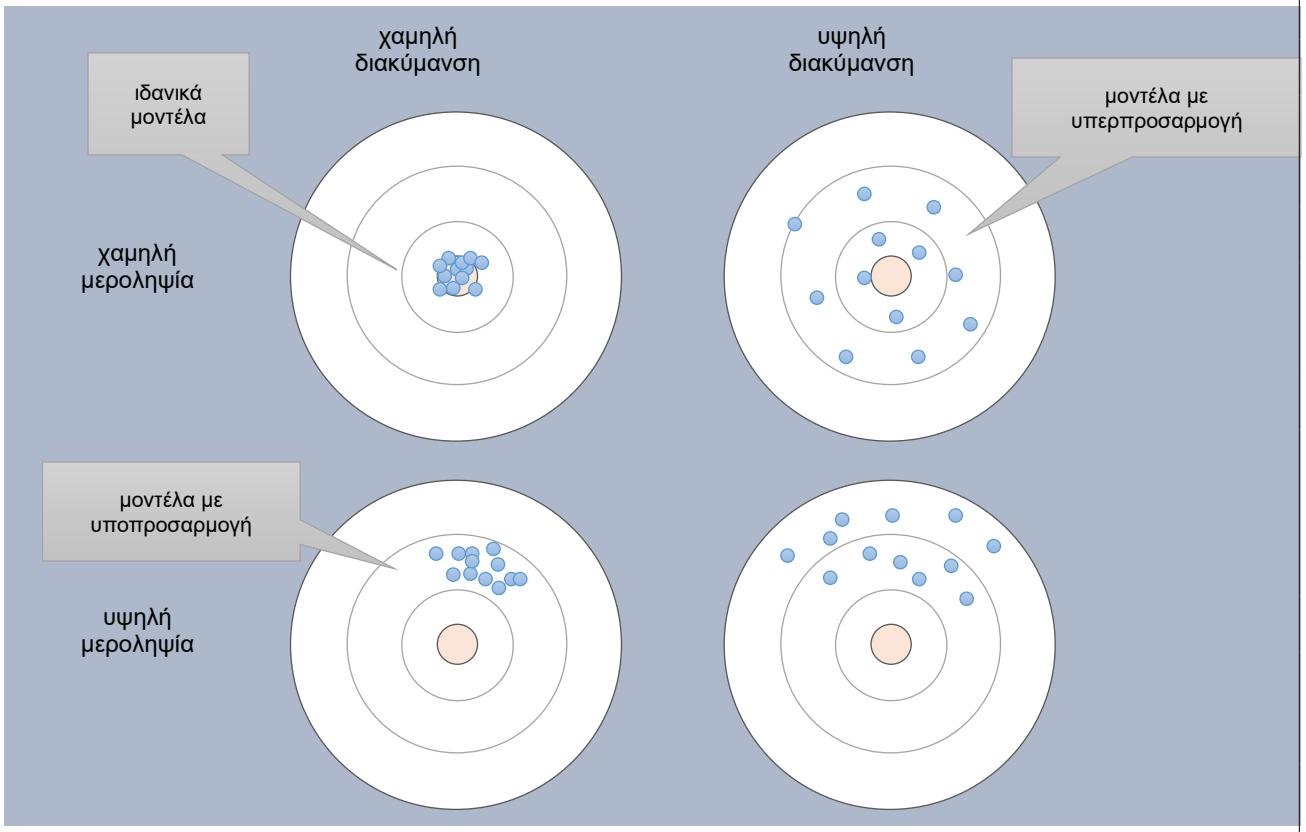




  - Η υπερπροσαρμογή συμβαίνει όταν μαζί με τη σχέση στα δεδομένα έχει επιπλέον μοντελοποιηθεί και θόρυβος.
    - ✓ Αυτό συμβαίνει όταν εκπαιδεύουμε το μοντέλο μας υπερβολικά πάνω σε δεδομένα εκπαίδευσης που έχουν θόρυβο.
    - ✓ Τέτοια μοντέλα έχουν **χαμηλή μεροληψία και υψηλή διακύμανση** και είναι κατά βάση πολύπλοκα (όπως π.χ. τα δένδρα ταξινόμησης που είναι επιρρεπή σε υπερπροσαρμογή).

Πολυπλοκότητα μοντέλου (bias-variance) (συνεχ.)

- Γραφική αναπαράσταση της ποιότητας ενός μοντέλου σε σχέση με τη μεροληψία και τη διακύμανση του.
  - Κάθε σημείο στους κύκλους αναπαριστά ένα διαφορετικό μοντέλο που προκύπτει μεταβάλλοντας τις παραμέτρους εκπαίδευσης.
  - Στο κέντρο του στόχου αναπαρίσταται ένα ιδανικό μοντέλο που προβλέπει τέλεια τις σωστές τιμές. Όσο απομακρυνόμαστε από το κέντρο οι προβλέψεις χειροτερεύουν.
  - Η αυξημένη μεροληψία παράγει συστηματικά μοντέλα που κάνουν λάθος
    - ✓ Λόγω υποπροσαρμογής
  - Η αυξημένη διακύμανση παράγει μοντέλα με σχετικά απρόβλεπτη και συνήθως χαμηλής ποιότητας απόδοση
    - ✓ Λόγω υπερπροσαρμογής

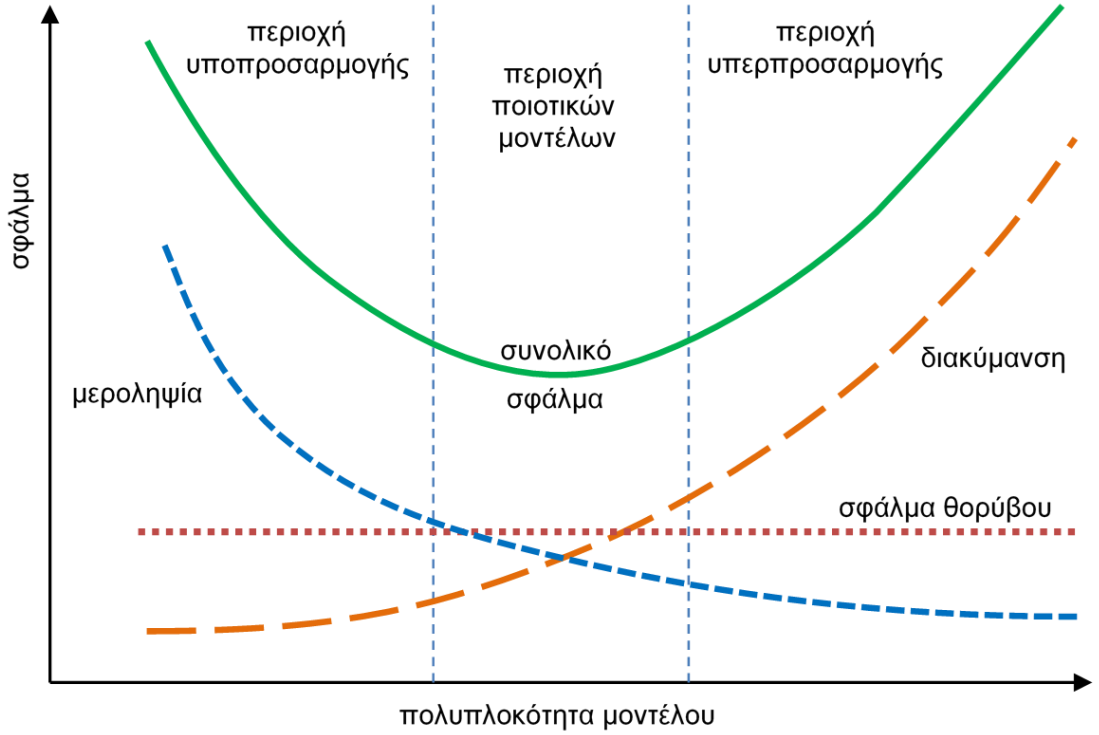


Πολυπλοκότητα μοντέλου (bias-variance) (συνεχ.)

- *Ένα μοντέλο με υψηλή πολυπλοκότητα έχει συνήθως μικρή μεροληψία αλλά μεγάλη διακύμανση* (overfits) και το αντίστροφο.
- Επειδή η βελτίωση της μιας μετρικής συχνά προκαλεί υποβάθμιση της άλλης, πρακτικά ενδιαφερόμαστε για μοντέλα που επιτυγχάνουν το βέλτιστο συμβιβασμό μεταξύ μεροληψίας και διακύμανσης (*bias-variance trade-off*)
  - Παράδειγμα για k-NN http://scott.fortmann-roe.com/docs/BiasVariance.html
    - ✓ Οι μικρές τιμές του k δημιουργούν ένα πολύπλοκο σύνορο διαχωρισμού των κλάσεων (μοντέλο), που αν και ταξινομεί σωστά τα δεδομένα εκπαίδευσης αδυνατεί να κάνει το ίδιο με τα νέα δεδομένα.
    - ✓ Αυξάνοντας το k, ελαττώνεται η πολυπλοκότητα οπότε αυξάνεται η μεροληψία αλλά μειώνεται η διακύμανση και τελικά βελτιώνεται η απόδοσή του.
    - ✓ Άρα χρειάζεται ένας συμβιβασμός στην επιλογή του k.
    - ✓ Why does decreasing K in K-nearest-neighbours increase complexity?
    - ✓ Machine Learning Fundamentals:Bias and Variance - Video
  - Άλλο παράδειγμα είναι οι (παραμετρικοί) αλγόριθμοι, γραμμικής και λογιστικής παρεμβολής.
    - ✓ Προσπαθώντας να μοντελοποιήσουν τα δεδομένα εκπαίδευσης με έναν συγκεκριμένο, συνήθως χαμηλό πλήθος παραμέτρων, αδυνατούν να περιγράψουν σύνθετα δεδομένα με αποτέλεσμα να παρουσιάζουν υψηλή μεροληψία.

Σχέση πολυπλοκότητας και σφάλματος

- Όσο αυξάνεται η πολυπλοκότητα του μοντέλου, μειώνεται η μεροληψία του (προβλέπει καλύτερα στα δεδομένα εκπαίδευσης) ενώ αυξάνεται η διακύμανση του και τελικά αυξάνεται το συνολικό του σφάλμα.



- Συνοψίζοντας, θα μπορούσαμε να πούμε ότι το συνολικό σφάλμα γενίκευσης ενός μοντέλου στα δεδομένα δοκιμής έχει <u>τρεις συνιστώσες</u>: προερχόμενες από τη μεροληψία του, τη διακύμανση και τους *μη αντιμετωπίσιμους παράγοντες* (*irreducible error*). Δηλαδή:

$$Συνολικό\ Σφάλμα\ =\ Σφ_{\text{μεροληψίας}}\ +Σφ_{\text{διακύμανσης}}+Σφ_{\text{irreducible}}$$

Γ) Πρόβλεψη Απόδοσης

- Η δοκιμή ενός μοντέλου μπορεί να θεωρηθεί ότι είναι μια μεροληπτική διαδικασία Bernoulli (*biased Bernoulli process*).
  - Μία διαδικασία Bernoulli είναι μία πεπερασμένη ή άπειρη ακολουθία ανεξάρτητων τυχαίων μεταβλητών X_1, X_2, X_3, … ,(ή γεγονότων) τέτοια ώστε
    - ✓ Για κάθε i η X_i παίρνει την τιμή 0 ή 1
    - ✓ Για κάθε i η πιθανότητα το X_i να είναι 1 είναι πάντα ίση με p
  - Βασικές Ιδιότητες Διαδικασιών Bernoulli: Ανεξαρτησία, Έλλειψη μνήμης
    - ✓ π.χ. στο παιχνίδι "κορώνα-γράμματα", στο οποίο το αποτέλεσμα κάθε φορά είναι ανεξάρτητο από το αποτέλεσμα της προηγούμενης ή της επόμενης ρίψης
    - ✓ θεωρείστε ότι κορώνα = επιτυχία και γράμματα = σφάλμα
  - Το λάθος ενός μοντέλου βέβαια δεν είναι τυχαίο αλλά μεροληπτικό (biased).
    - ✓ Αν είχαμε ένα "πειραγμένο" κέρμα, που π.χ. στις 100 ρίψεις οι 75 θα ήταν επιτυχείς θα μπορούσαμε να πούμε ότι η πιθανότητα και η επόμενη φορά να είναι επιτυχής θα ήταν 75%.
  - Γενικά, αν στις *N* δοκιμές ενός μοντέλου, οι *S* είναι επιτυχείς, τότε το ποσοστό επιτυχίας *f* είναι *f=S/N*
    - ✓ Το *f* είναι απλά μία εκτίμηση του ποσοστού επιτυχίας
    - ✓ Μπορούμε να υποθέσουμε ότι και σε ένα ξεχωριστό σύνολο δεδομένων, το ποσοστό επιτυχίας θα είναι περίπου το ίδιο, δηλ. *f*.
    - ✓ Πόση όμως θα ήταν η απόκλιση; 5%; 10%;
    - ✓ Σίγουρα ένα ποσοστό επιτυχίας που θα προέκυπτε από 10.000 παραδείγματα εκπαίδευσης θα ήταν πιο αξιόπιστο απ' ότι αν προέκυπτε από 100. Πόσο όμως πιο αξιόπιστο;
  - Η απάντηση δίνεται από το *διάστημα εμπιστοσύνης*.

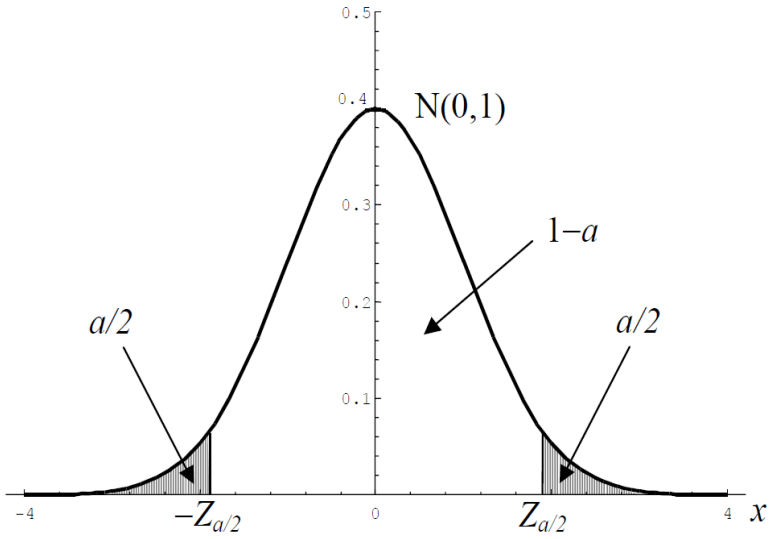
Διάστημα Εμπιστοσύνης (confidence interval)

- Είναι ένα διάστημα αριθμών που πιστεύεται (εκτιμάται) ότι εμπεριέχει μια άγνωστη παράμετρο p (π.χ. μέσο, τυπική απόκλιση) ενός πληθυσμού.
- Ταυτόχρονα, είναι ένα μέτρο της εμπιστοσύνης για την άγνωστη παράμετρο.
  - Δηλαδή το p ανήκει σε ένα *διάστημα εμπιστοσύνης* (*confidence interval* - *CI*) με μια ορισμένη *εμπιστοσύνη* c (*confidence*).
  - Π.χ. θα μπορούσαμε να πούμε ότι υπάρχει 90% βεβαιότητα ότι το ποσοστό των νέων 18-24 ετών που χρησιμοποιούν καθημερινά το internet είναι από 85 έως 95%. Δηλαδή το διάστημα εμπιστοσύνης είναι 85-95%.
- Για δεδομένο διάστημα εμπιστοσύνης μπορούμε να βρούμε την εμπιστοσύνη (βεβαιότητα-πιθανότητα) που αντιστοιχεί σε αυτό, αλλά συχνά ακολουθούμε την αντίστροφη διαδικασία.
  - Δηλαδή ορίζουμε έναν επιθυμητό βαθμό εμπιστοσύνης ως πιθανότητα, π.χ. 0.95 ή 95%, και βρίσκουμε σε ποιο διάστημα εμπιστοσύνης αντιστοιχεί.
    - ✓ Τα πιο συνήθη στην πράξη ποσοστά εμπιστοσύνης είναι 95, 98, 99 ή και 99.5% ακόμη.
    - ✓ Όσο μεγαλύτερη εμπιστοσύνη ορίζουμε, τόσο μεγαλύτερο το διάστημα εμπιστοσύνης που προκύπτει
- Παράδειγμα. Αν κάνουμε *N*=1.000 ρίψεις ενός κέρματος και οι *S*=750 από αυτές είναι επιτυχείς (δηλ. κορώνα), τότε το *f* = 75%
  - Δηλαδή προκύπτει ότι το *p* πρέπει να είναι κοντά στο 75%. Πόσο κοντά όμως;
  - Με εμπιστοσύνη ***c*** ορισμένη στο 80%, το *p* είναι ανάμεσα στο 73,3% και 76,8% (διάστημα εμπιστ.)
  - Αν το πείραμα όμως ήταν μικρότερο (*N*=100, *S*=75), με την ίδια εμπιστοσύνη, τότε προκύπτει ότι το διάστημα αυτό θα είναι μεγαλύτερο (70% - 81%)

Διάστημα Εμπιστοσύνης (confidence interval) (συνεχ.)

- Ο υπολογισμός του *p* προκύπτει ως εξής:
  - Η **μέση τιμή (mean)** και η **διακύμανση** ή **διασπορά (variance)** μιας επιτυχούς δοκιμής Bernoulli με ποσοστό επιτυχίας *p,* είναι *p* και *p·(1-p)* αντίστοιχα.
  - Για *N* δοκιμές, το εκτιμώμενο ποσοστό επιτυχίας *f=S/N* είναι μια τυχαία μεταβλητή με *μέση τιμή N·p* και *διακύμανση*[1] *N·p·(1-p)*
    - ✓ Για μεγάλες τιμές του *N*, η τυχαία αυτή μεταβλητή θεωρείται ότι ακολουθεί την κανονική κατανομή (από το κεντρικό οριακό θεώρημα)
    - ✓ Εξηγήσεις για τη Μέση τιμή
    - ✓ Εξηγήσεις για τη Διακύμανση
  - Γενικά, η πιθανότητα μιας τυχαίας μεταβλητής X με μέση τιμή 0 να ανήκει σε ένα διάστημα εμπιστοσύνης μεγέθους 2·z είναι:

    $P(-z < X < z) = 1 - a = c,$ όπου:
    - ✓ *c* ο συντελεστής εμπιστοσύνης, και
    - ✓ *a*: η πιθανότητα το X να βρεθεί έξω από το διάστημα, οπότε *a*/2 είναι η πιθανότητα P(X≥z) και P(X≤-z)
  - Οι αντίστοιχες τιμές του *z* για ένα επίπεδο εμπιστοσύνης *c*, για μια κανονική κατανομή, δίνονται από σχετικούς πίνακες (βλ. επόμενη διαφάνεια)



[1] Για ένα πεπερασμένο σύνολο αριθμών, η διακύμανση υπολογίζεται ως ο μέσος όρος των τετραγώνων των αποκλίσεων των τιμών από τη μέση τιμή τους. Η τετραγωνική ρίζα της διακύμανσης είναι η τυπική απόκλιση.

Διάστημα Εμπιστοσύνης (confidence interval) (συνεχ.)

- Διαδικασία:
  - Αν η τιμή της ζητούμενης εμπιστοσύνης είναι *c*, υπολογίζουμε το a/2 (δηλ. το (1-*c*)/2), και από τους πίνακες βρίσκουμε το σχετικό *z* με το οποίο υπολογίζουμε την παρακάτω εξίσωση που μας δίνει τα άκρα του ζητούμενου διαστήματος εμπιστοσύνης.

$$p = (f + \frac{2z^2}{2N} \pm z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}})/(1 + \frac{z^2}{N})$$

- Παράδειγμα
  - Έστω ότι σε N=1.000 δοκιμές ενός μοντέλου, οι S=750 ήταν επιτυχείς, οπότε το f είναι 75%.
  - Με ζητούμενη εμπιστοσύνη c=80% προκύπτει ότι το (1-c)/2=(1-0.8)/2=0.1.
  - Δηλαδή το a/2 ή η P(X≥z) είναι 0.1 ή 10% και ο Πίνακας δίνει το αντίστοιχο z ίσο με 1.28.
  - Συνεπώς από την προηγούμενη εξίσωση προκύπτει ότι το διάστημα εμπιστοσύνης είναι το [0.733, 0.768].
  - για *N*=100 το αντίστοιχο διάστημα εμπιστοσύνης είναι μεγαλύτερο, δηλ. το [0.7, 0.81].
  - Αύξηση του διαστήματος θα είχαμε επίσης και αν αυξάναμε την απαιτούμενη εμπιστοσύνη *c*.

P(X≥z)	z
0.1 %	3.09
0.5 %	2.58
1 %	2.33
5 %	1.65
10 %	1.28
20 %	0.84
40 %	0.25

- Θεωρούμε ότι τα αποτελέσματα των δοκιμών ακολουθούν κανονική κατανομή
  - Αλλιώς πρέπει να εκτελέσουμε τις δοκιμές πολλές φορές για να υπολογίσουμε το μέσο και την τυπική τους απόκλιση (std)
  - How to Calculate Confidence Intervals in Python / Python Scipy Confidence Interval

Python Example

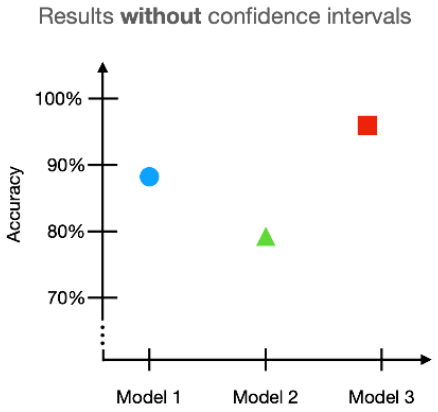
- Για τον υπολογισμό του confidence interval, πρέπει πρώτα να βρεθεί το z-score και στη συνέχεια υπολογίζονται τα όρια.
- π.χ. στο παράδειγμα που έχουμε στην προηγούμενη διαφάνεια, έχουμε μοντέλο που κάνει 1000 προβλέψεις με ακρίβεια 75%.
- Θέλουμε διάστημα εμπιστοσύνης με confidence 95% (Άρα a=5%).
- Για κανονική κατανομή της ακρίβειας (acc) η λύση είναι η ακόλουθη:

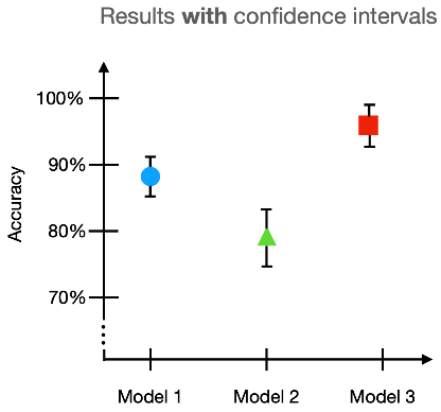
```
import math
from scipy.stats import norm

acc = 0.75
N = 1000
a = 0.05 (confidence=0.95)

std = math.sqrt((acc * (1 - acc)) / N)
z_score = norm.ppf((1 + a) / 2)

margin_of_error = z_score * std
interval = [acc - margin_of_error, acc + margin_of_error]
```





	Dataset A	Dataset B	Dataset C
Model 1	89.1%	...	...
Model 2	79.5%	...	...
Model 3	95.2%	...	...

	Dataset A	Dataset B	Dataset C
Model 1	89.1% ± 1.7%	...	...
Model 2	79.5% ± 2.2%	...	...
Model 3	95.2% ± 1.6%	...	...

or

	Dataset A	Dataset B	Dataset C
Model 1	89.1% (87.4%, 90.8%)	...	...
Model 2	79.5% (77.3%, 81.7%)	...	...
Model 3	95.2% (93.6%, 96.8%)	...	...

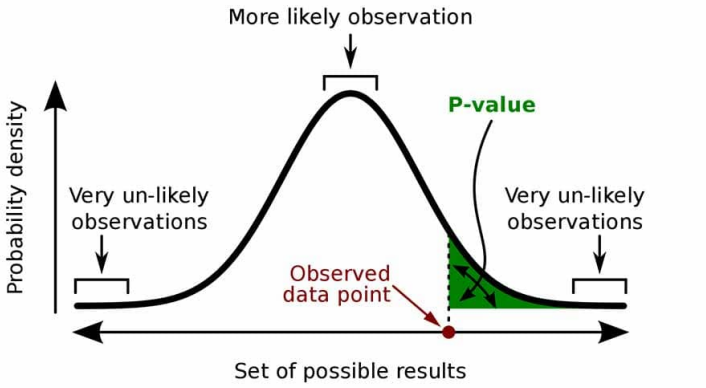
- Αν η μετρική acc δεν ακολουθεί κανονική κατανομή τότε θα χρειαστούν πολλά trials του ίδιου αλγορίθμου (seeds, cross-validation, κλπ) και πρέπει να αντικαταστήσουμε τον τύπο για τον υπολογισμό της τυπικής απόκλισης/standard deviation (std):

```
acc_list = [0.8, 0.6, 0.7, 0.8]
std = np.std(acc_list)
```

- Confidence Intervals for Machine Learning
- Creating Confidence Intervals for Machine Learning Classifiers

Δ) Στατιστικά τεστ σύγκρισης αλγορίθμων

- Χρησιμοποιούνται για να συγκρίνουμε έναν αλγόριθμό με έναν (ή παραπάνω) αλγορίθμους σε ένα ή περισσότερα data sets.
- Σκοπός: εύρεση ύπαρξης *στατιστικά σημαντικών* (σ.σ.) διαφορών μεταξύ τους
  - Ο όρος σημαντικότητα (*significance*) αναφέρεται στην πρακτική αξιοποίηση του αποτελέσματος
    - ✓ The term significance does not imply importance here
  - Ένας αλγόριθμος υπερισχύει άλλου αν υπάρχουν σ.σ. διαφορές
  - Μηδενική υπόθεση (Null Hypothesis): Υπόθεση ότι δεν υπάρχουν σ.σ. διαφορές μεταξύ των αλγορίθμων
  - *p*-value (*probability value*): Η πιθανότητα να πάρουμε αυτά τα δεδομένα όταν ισχύει η Μηδενική Υπόθεση
    - ✓ The lower the p-value, the greater the statistical significance of the observed difference.



A **p-value** (shaded green area) is the probability of an observed (or more extreme) result assuming that the null hypothesis is true.

  - *α*-value (*στάθμη σημαντικότητας*): Το όριο που θέτουμε ώστε τα δεδομένα να προέρχονται τυχαία (δηλ. να μην υπάρχουν σσ διαφορές)
    - ✓ Η στάθμη επιλέγεται πριν την συλλογή δεδομένων και τυπικά τίθεται στο 5% (=0.05) ή και λιγότερο
  - Αν $p \leq a$ τότε υπάρχει σ.σ. διαφορά. Αλλιώς ισχύει η Μηδενική Υπόθεση
    - ✓ SciPy Statistical Significance Tests

What is Statistical Significance Test

- In statistics, statistical significance means that the result produced, has a reason behind it
  - i.e. it was not produced randomly or by chance.
    - ✓ scipy.stats is a module in SciPy, which has functions for performing statistical significance tests.
- Null Hypothesis
  - It assumes that the observation is not statistically significant.
- Alpha value
  - Is the level of significance.
    - ✓ Example: How close to extremes the data must be for null hypothesis to be rejected.
    - ✓ It is usually taken as 0.01, 0.05, or 0.1.
- P value
  - P value tells how close to extreme the data actually is.
  - P value and alpha values are compared to establish the statistical significance.
  - If p value <= alpha we reject the null hypothesis and say that the data is statistically significant.
    - ✓ otherwise we accept the null hypothesis.
- T-Test
  - T-tests are used to determine if there is significant deference between means of two variables and let us know if they belong to the same distribution.
  - The function ttest_ind() takes two samples of same size and produces a tuple of t-statistic and p-value.
    - ✓ If you want to return only the p-value, use the pvalue: ttest_ind().pvalue

Στατιστικά τεστ σύγκρισης αλγορίθμων (συνεχ.)

- Η σύγκρισή των *k* αλγορίθμων μπορεί να γίνει σε *n* σύνολα δεδομένων
  - Ανάλογα με το πλήθος των αλγορίθμων και των dataset που θέλουμε να συγκρίνουμε, επιλέγουμε και αντίστοιχα τη μέθοδο που τα υποστηρίζει
    - ✓ Δεν υποστηρίζουν όλες οι μέθοδοι σύγκρισης k > 2 ή n > 1
  - Για πολλούς αλγορίθμους και πολλά σύνολα δεδομένων, τα αποτελέσματα παρουσιάζονται σε πίνακες με τους αλγορίθμους να συγκρίνονται ανά δύο σε ζεύγη
- Wilcoxon signed-rank test
  - Στατιστικό μέτρο ελέγχου της διαφοράς ανά ζεύγη (paired difference test)
  - Ελέγχει εάν οι αποδόσεις 2 αλγορίθμων σε 2 ή περισσότερα σύνολα δεδομένων έχουν σ.σ. διαφορές
  - Δηλαδή χρησιμοποιείται όταν είναι 2 αλγόριθμοι και ≥2 dataset
    - ✓ Python command (scipy): scipy.stats.wilcoxon(sample1, sample2)
- Friedman test
  - Ελέγχει εάν οι αποδόσεις 2 (ή παραπάνω) αλγορίθμων έχουν σ.σ. διαφορές σε περισσότερα από ένα σύνολα δεδομένων
  - ≥2 αλγόριθμοι, ≥ 2 dataset
    - ✓ Python (scipy): scipy.stats.friedmanchisquare(sample1, sample2, sample3, …)
  - Εάν υπάρχουν διαφορές, μπορούμε να δούμε ποιοι αλγόριθμοι είναι διαφορετικοί μεταξύ τους μέσω post-hoc analysis, π.χ. με:
    - ✓ Nemenyi test[123], Conover test[3], Siegel-Tukey test[3], Bonferroni-Dunn[12]
    - ✓ Python libraries: [1]=STAC, [2]=Orange, [3]=scikit-posthocs

ANOVA (Analysis of Variance)

- Ελέγχει εάν οι εκτιμήσεις 2 ή περισσότερων αλγορίθμων έχουν την ίδια μέση τιμή
- ≥2 αλγόριθμοι, 1 dataset
- Python command (scipy): scipy.stats.f_oneway(sample1, sample2, …)

Παράδειγμα Ranked average

	C4.5	1-NN	NaiveBayes	Kernel	CN2
Abalone*	0.219 (3)	0.202 (4)	0.249 (2)	0.165 (5)	0.261 (1)
Adult*	0.803 (2)	0.750 (4)	0.813 (1)	0.692 (5)	0.798 (3)
Australian	0.859 (1)	0.814 (4)	0.845 (2)	0.542 (5)	0.816 (3)
Autos	0.809 (1)	0.774 (3)	0.673 (4)	0.275 (5)	0.785 (2)
Balance	0.768 (3)	0.790 (2)	0.727 (4)	0.872 (1)	0.706 (5)
Breast	0.759 (1)	0.654 (5)	0.734 (2)	0.703 (4)	0.714 (3)
Bupa	0.693 (1)	0.611 (3)	0.572 (4.5)	0.689 (2)	0.572 (4.5)
Car	0.915 (1)	0.857 (3)	0.860 (2)	0.700 (5)	0.777 (4)
Cleveland	0.544 (2)	0.531 (4)	0.558 (1)	0.439 (5)	0.541 (3)
Crx	0.855 (2)	0.796 (4)	0.857 (1)	0.607 (5)	0.809 (3)
Dermatology	0.945 (3)	0.954 (2)	0.978 (1)	0.541 (5)	0.858 (4)
German	0.725 (2)	0.705 (4)	0.739 (1)	0.625 (5)	0.717 (3)
Glass	0.674 (4)	0.736 (1)	0.721 (2)	0.356 (5)	0.704 (3)
Hayes-Roth	0.801 (1)	0.357 (4)	0.520 (2.5)	0.309 (5)	0.520 (2.5)
Heart	0.785 (2)	0.770 (3)	0.841 (1)	0.659 (5)	0.759 (4)
Ion	0.906 (2)	0.359 (5)	0.895 (3)	0.641 (4)	0.918 (1)
Led7Digit	0.710 (2)	0.402 (4)	0.728 (1)	0.120 (5)	0.674 (3)
Letter*	0.691 (2)	0.827 (1)	0.667 (3)	0.527 (5)	0.638 (4)
Lymphography	0.743 (3)	0.739 (4)	0.830 (1)	0.549 (5)	0.746 (2)
Mushrooms*	0.990 (1.5)	0.482 (5)	0.941 (3)	0.857 (4)	0.990 (1.5)
OptDigits*	0.867 (3)	0.098 (1)	0.915 (2)	0.986 (1)	0.784 (4)
Satimage*	0.821 (3)	0.872 (2)	0.815 (4)	0.885 (1)	0.778 (5)
SpamBase*	0.893 (2)	0.824 (4)	0.902 (1)	0.739 (5)	0.885 (3)
Splice*	0.799 (2)	0.655 (4)	0.925 (1)	0.517 (5)	0.755 (3)
Tic-tac-toe	0.845 (1)	0.731 (2)	0.693 (4)	0.653 (5)	0.704 (3)
Vehicle	0.741 (1)	0.701 (2)	0.591 (5)	0.663 (3)	0.619 (4)
Vowel	0.799 (2)	0.994 (1)	0.603 (4)	0.269 (5)	0.621 (3)
Wine	0.949 (4)	0.955 (2)	0.989 (1)	0.770 (5)	0.954 (3)
Yeast	0.555 (3)	0.505 (4)	0.569 (1)	0.312 (5)	0.556 (2)
Zoo	0.928 (2.5)	0.928 (2.5)	0.945 (1)	0.419 (5)	0.897 (4)
average rank	2.100	3.250	2.200	4.333	3.117

Source: https://www.rdocumentation.org/packages/scmamp/versions/0.2.55/topics/data.gh.2008

Παράδειγμα (Friedman & Nemenyi post-hoc test)

- Η μέθοδος Friedman υπολογίζει το *p*-value
  - Στο συγκεκριμένο παράδειγμα, το *p*-value είναι πολύ μικρό (μικρότερο από το α=0.05) επομένως οι αλγόριθμοι έχουν σ.σ. διαφορές μεταξύ τους

```
Iman Davenport's correction of Friedman's rank sum test

data:  data.gh.2008
Corrected Friedman's chi-squared = 14.3087, df1 = 4, df2 = 116,
p-value = 1.593e-09
```

- Αφού επαληθεύσαμε ότι όντως υπάρχουν σ.σ. διαφορές, θέλουμε να ελέγξουμε μεταξύ ποιων αλγορίθμων εμφανίζονται οι διαφορές αυτές, οπότε χρησιμοποιούμε το τεστ *Nemenyi*:
  - Το τεστ υπολογίζει ένα μέγεθος που λέγεται *κρίσιμη διαφορά* (*critical difference*), που είναι ένα μέτρο εμπιστοσύνης που έχουμε στη μηδενική υπόθεση (δηλ. να μην υπάρχουν σ.σ. διαφορές)
  - Σε οποιαδήποτε περίπτωση η διαφορά των μέσων βαθμών (ranked averages) ενός αλγορίθμου με έναν άλλο είναι μεγαλύτερη από την κρίσιμη διαφορά, τότε αυτοί οι δυο αλγόριθμοι θεωρούνται σημαντικά διαφορετικοί
    - ✓ In statistics, the Nemenyi test is a post-hoc [2]test intended to find the groups of data that differ after a global statistical test (such as the Friedman test) has rejected the null hypothesis.
    - ✓ The test makes pair-wise tests of performance.

```
Nemenyi test

data:  data.gh.2008
Critical difference = 1.1277, k = 5, df = 145

> nm$diff.matrix

          C4.5  k-NN(k=1) NaiveBayes    Kernel        CN2
[1,]  0.000000 -1.1500000 -0.1000000 -2.233333 -1.0166667
[2,] -1.150000  0.0000000  1.0500000 -1.083333  0.1333333
[3,] -0.100000  1.0500000  0.0000000 -2.133333 -0.9166667
[4,] -2.233333 -1.0833333 -2.1333333  0.000000  1.2166667
[5,] -1.016667  0.1333333 -0.9166667  1.216667  0.0000000
```

[2] The term "post hoc" comes from the Latin for "after the event". A post hoc test is used only after we find a statistically significant result and need to determine where our differences truly came from.

Ε) Μετρικές Αξιολόγησης Ταξινόμησης

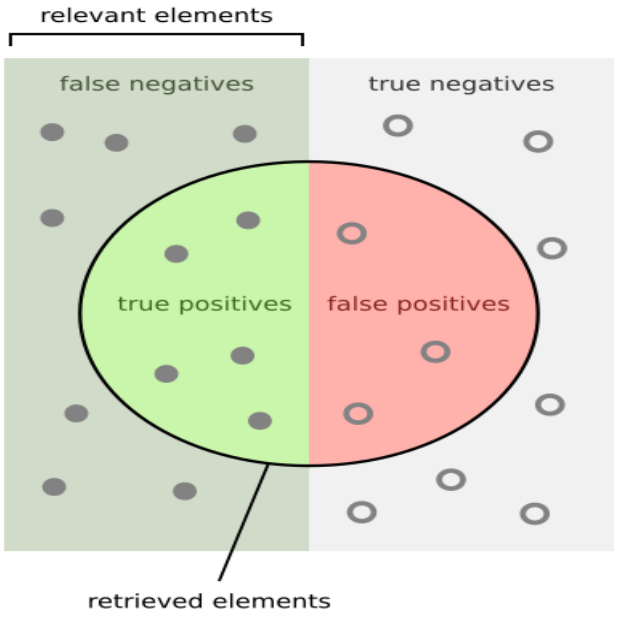
- Έστω ένας δυαδικός ταξινομητής με αποτελέσματα που χαρακτηρίζονται ως θετικά (positive - p) ή αρνητικά (negative – n).
- Έχουμε 4 περιπτώσεις αποτελεσμάτων που μπορούν να περιγραφούν σε έναν 2x2 πίνακα ενδεχομένων (*contingency matrix*) ή πίνακα σύγχυσης (confusion matrix)
  - Είναι ένας τρόπος παρουσίασης <u>των επιδόσεων ανά κλάση</u> ενός ταξινομητή

		Προβλεπόμενη Κλάση (Predicted class)	
Πραγματική Κλάση (Actual Class)		*Class= Positive*	*Class= Negative*
	Class= Positive	TP (True Positive)	FN (False Negative)
	Class=Negative	FP (False Positive)	TN (True Negative)

- Για παράδειγμα σε ένα διαγνωστικό τεστ για την εξακρίβωση μιας πάθησης
  - FP: το τέστ είναι θετικό αλλά στην πραγματικότητα ο ασθενής δεν έχει την πάθηση
  - FN: Το τέστ είναι αρνητικό, αλλά ο ασθενής έχει την πάθηση
- Από έναν πίνακα ενδεχομένων μπορούν να παραχθούν αρκετά μέτρα εκτίμησης

Μέτρα Εκτίμησης

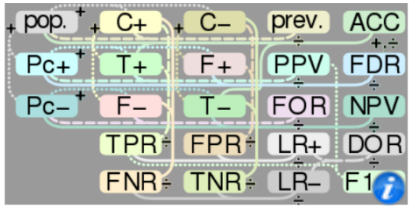
true positive (TP)
eqv. with hit
true negative (TN)
eqv. with correct rejection
false positive (FP)
eqv. with false alarm, Type I error
false negative (FN)
eqv. with miss, Type II error
true positive rate (TPR)
eqv. with hit rate, recall, sensitivity
$TPR = TP / P = TP / (TP + FN)$
false positive rate (FPR)
eqv. with false alarm rate, fall-out
$FPR = FP / N = FP / (FP + TN)$
accuracy (ACC)
$ACC = (TP + TN) / (P + N)$
specificity (SPC)
$SPC = TN / (FP + TN) = 1 - FPR$
positive predictive value (PPV)
eqv. with precision
$PPV = TP / (TP + FP)$
negative predictive value (NPV)
$NPV = TN / (TN + FN)$
false discovery rate (FDR)
$FDR = FP / (FP + TP)$



Confusion matrix

		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$	F_1 score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

Click thumbnail for interactive chart:



Μέτρα Εκτίμησης

- *Accuracy* (*Ακρίβεια*): το ποσοστό των παραδειγμάτων που ταξινομούνται σωστά:

$$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$$

  - Ευκολονόητη μετρική που όμως σε πολλές περιπτώσεις δίνει παραπλανητική πληροφόρηση και για αυτό είναι χρήσιμη μόνο σε συνδυασμό με τις άλλες μετρικές
  - Error Rate (Λόγος Λάθους) $\frac{\mathrm{FP}+\mathrm{FN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$
- *Precision* (P) (*Ευστοχία*) ή positive predictive value (PPV):
  - Ποσοστό παραδειγμάτων που ταξινομούνται ως θετικά και είναι σωστά (θετικά): $\mathrm{P}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$
  - Σημαντική μετρική όταν θέλουμε να είμαστε πολύ σίγουροι για την πρόβλεψη μας
    - ✓ Π.χ. ο εντοπισμός παράνομης χρήσης μιας πιστωτικής κάρτας ώστε να μην γίνεται αναίτια η ακύρωση της
- True Positive Rate (TPR) ή Sensitivity (ευαισθησία): $\mathrm{TPR}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$
  - Το ποσοστό των θετικών παραδειγμάτων που βρήκε ο ταξινομητής
  - Ισοδύναμο με την ανάκληση (Recall - r)
  - χρήσιμη μετρική όταν θέλουμε ο ταξινομητής μας να "πιάνει" όσο το δυνατόν περισσότερα θετικά παραδείγματα ακόμη και αν δεν είναι πολύ σίγουρος.
    - ✓ Για παράδειγμα σε ένα σύστημα πρόβλεψης μια ασθένειας
- False Negative Rate (FNR): Το ποσοστό των θετικών παραδειγμάτων που ταξινομούνται λάθος

$$\mathrm{FNR}=\frac{\mathrm{FN}}{\mathrm{TP}+\mathrm{FN}}=1-\mathrm{TPR}$$

Μέτρα Εκτίμησης (συνέχ.)

- Επειδή τα κριτήρια *Precision* και *Recall* δεν αρκούν από μόνα τους για να περιγράψουν την συνολική επίδοση του ταξινομητή συνήθως συνδυάζονται στο κριτήριο *F-measure*
  - ✓ (ή αλλιώς *F1-score*) sklearn.metrics.f1_score
- Είναι ο αρμονικός μέσος (harmonic average) της ακρίβειας (*precision*) και της ανάκλησης (*recall*)
  - ✓ Δηλ. το πηλίκο του γεωμετρικού μέσου προς το αλγεβρικό μέσο όρο των δύο κριτηρίων

$$\mathrm{F}_1 = 2 \cdot \frac{1}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

- Καλύτερη τιμή1 (ιδανικό precision και recall) και χειρότερη το 0
  - ✓ Ονομάζεται και *balanced F-score* ή F_1 measure, γιατί τα recall και precision συμμετέχουν ισότιμα
- Δύο άλλες μετρικές είναι η
- F_2 μετρική που δίνει μεγαλύτερη βαρύτητα στο recall (δίνοντας βάρος στα false negatives) και η
- $F_{0.5}$ που δίνει μικρότερη βαρύτητα στο recall (μειώνοντας την επίδραση των false negatives)
  - ✓ The F-score is often used in the field of information retrieval for measuring search, document classification, and query classification performance.
  - ✓ It has been widely used in the natural language processing literature, such as the evaluation of named entity recognition and word segmentation.
- Ο γενικός τύπος του *F-score* $\mathrm{F}_\beta = (1+\beta^2) \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{(\beta^2 \cdot \mathrm{precision}) + \mathrm{recall}}$ Three common values for the beta parameter:
  - ✓ $F_{0.5}$-Measure (beta=0.5): More weight on precision, less weight on recall.
  - ✓ F_1-Measure (beta=1.0): Balance the weight on precision and recall.
  - ✓ F_2-Measure (beta=2.0): Less weight on precision, more weight on recall

Μέτρα Εκτίμησης (συνέχ.)

- Τα κριτήρια *Precision* και *Recall* εστιάζουν αποκλειστικά στη μια κλάση (τα Θετικά)
  - Τα Αρνητικά δεν εμπλέκονται πουθενά
  - Μπορεί να αρκεί όταν είναι πολύ σημαντική η κλάση με τα θετικά (π.χ. στη διάγνωση μιας ασθένειας) αλλά πολλές φορές είναι σημαντική και η σωστή ταξινόμηση στην άλλη κλάση (με τα αρνητικά).
- Δύο άλλα μέτρα εκτίμησης απόδοσης:
  - True Negative Rate (TNR) ή Specificity (εξειδίκευση): $\text{TNR} = \frac{\text{TN}}{\text{TN+FP}}$
    - ✓ Το αντίστοιχο του Recall το οποίο είναι για τη θετική κλάση
  - Το ποσοστό των αρνητικών παραδειγμάτων που ταξινομούνται σωστά (δηλ. ως αρνητικά).
  - False Positive Rate (FPR): Το ποσοστό των αρνητικών παραδειγμάτων που ταξινομούνται λάθος (δηλαδή ως θετικά): $\text{FPR} = \frac{\text{FP}}{\text{TN+FP}} = 1 - TNR$
- Τα μέτρα ουσιαστικά είναι ίδια όσον αφορά τον μαθηματικό τους ορισμό και διαφέρουν στην κλάση στην οποία εστιάζουν
  - Π.χ. το *Sensitivity* εστιάζει στη μια κλάση (Θετικά) ενώ το *Specificity* στην άλλη (Αρνητικά)
- Όπως και με τις μετρικές *Precision/Recall* έτσι και οι μετρικές *Specificity/Sensitivity* από μόνες τους δεν περιγράφουν πλήρως την επίδοση του ταξινομητή.
  - Το πιο συνηθισμένο συνδυαστικό κριτήριο βασίζεται στο γράφημα FPR (που είναι το 1- Specificity) - Sensitivity (TPR) ή Recall, που ονομάζεται Receiver Operating Characteristic (ROC)

Ανάλυση ROC

(Receiver Operating Characteristic or Relative Characteristic Curve)

- Είναι ένα εργαλείο για την επιλογή βέλτιστων δυαδικών ταξινομητών και στηρίζεται στο χώρο ROC
  - Αναπτύχθηκε κατά τον Β' παγκόσμιο πόλεμο για την αναγνώριση εχθρικών αντικειμένων
- Ο χώρος ROC ορίζεται από τους άξονες FPR (άξονας x) και TPR (άξονας y)
  - Απεικονίζει την σχέση μεταξύ κόστους (False Positive δηλ. false alarms) και ωφέλειας (True Positive)
- Η απόδοση ενός ταξινομητή αναπαρίσταται ως ένα σημείο στο χώρο ROC
  - Η καλύτερη μέθοδος πρόβλεψης παράγει ένα σημείο στο (0,1) δηλαδή όταν:
    - ✓ FPR=0 (άρα TNR=1 δηλ. 100% specificity) και TPR=1 (δηλ. 100% sensitivity). Τέλεια ταξινόμηση
  - Μια εντελώς τυχαία πρόβλεψη παράγει ένα σημείο στη διαγώνιο
  - Τα σημεία επάνω από τη διαγώνιο δείχνουν καλά αποτελέσματα ταξινόμησης
  - Τα σημεία κάτω από τη διαγώνιο δείχνουν προβληματική μέθοδο

Παράδειγμα

- Έστω A, B, C τα αποτελέσματα 3 ταξινομητών για 100 θετικές και 100 αρνητικές περιπτώσεις

A

TP=63	FN=37	100
FP=28	TN=72	100
91	109	200

TPR = 0.63
FPR = 0.28
ACC = 0.68

B

TP=77	FN=23	100
FP=77	TN=23	100
154	46	200

TPR = 0.77
FPR = 0.77
ACC = 0.50

C

TP=24	FN=76	100
FP=88	TN=12	100
112	88	200

TPR = 0.24
FPR = 0.88
ACC = 0.18

C'

TP=88	FN=12	100
FP=24	TN=76	100
112	88	200

TPR = 0.88
FPR = 0.24
ACC = 0.82

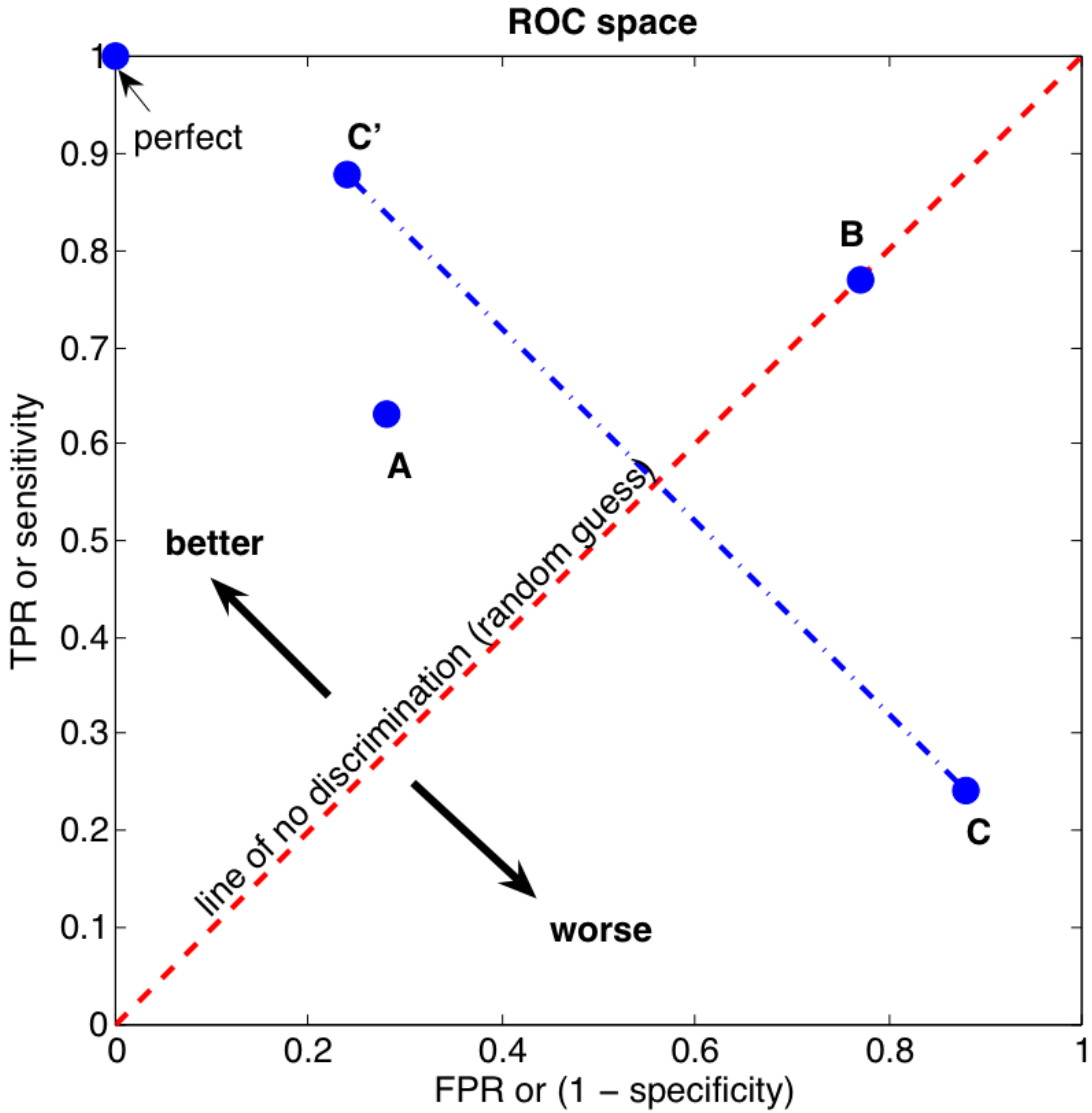
$ACC=\frac{TP+TN}{TP+TN+FP+FN}=135/200=0,68$

$TPR = \frac{TP}{TP+FN} = 63/100=0,63$ (Recall)

$FPR = \frac{FP}{TN+FP} = 28/100=0,28$ (

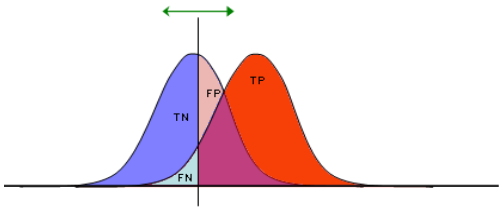
- Τα 3 αποτελέσματα απεικονίζονται ως σημεία στον χώρο ROC
  - Το A είναι σαφώς καλύτερο του B και C
  - Το B βρίσκεται πάνω στην διαγώνιο (τυχαία πρόβλεψη). Η ακρίβεια του είναι 50%
  - Αν το C κατοπτριστεί πάνω στη διαγώνιο, το C’ είναι καλύτερο (πλησιέστερο στην πάνω αριστερή γωνία) και από το A
    - ✓ Το C’ προκύπτει με την αντιστροφή των απαντήσεων του ταξινομητή C

Χώρος ROC

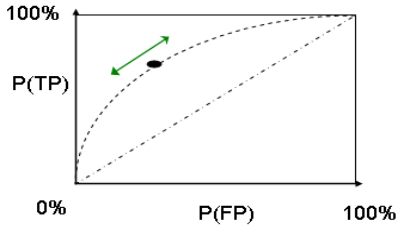

ROC space
perfect
C'
B
A
C
better
worse
line of no discrimination (random guess)
TPR or sensitivity
FPR or (1 – specificity)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.2
0.4
0.6
0.8
1

Καμπύλη ROC

- Οι ταξινομητές που προβλέπουν την πιο πιθανή κατηγορία μιας άγνωστης περίπτωσης, π.χ. Κανόνες, SVM, κλπ, απεικονίζονται στον χώρο ROC ως ένα σημείο
- Άλλοι ταξινομητές, όπως τα Δένδρα, ο Naïve Bayes, το Logistic regression και τα NN, παράγουν την πιθανότητα με την οποία κάποια περίπτωση ανήκει σε μια κλάση.
  - ✓ model.predict_proba(x)
  - ❑ Επίσης άλλοι ταξινομητές όπως ο SVM, μπορούν να γίνουν πιθανοκρατικοί με διάφορες επεκτάσεις



TP	FP
FN	TN
1	1

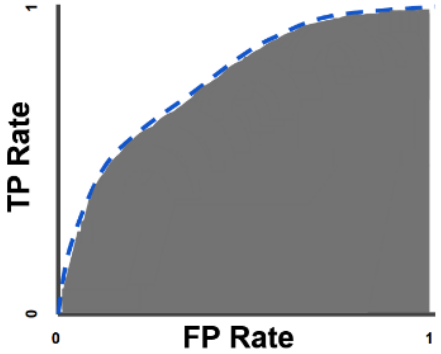


- Για τους ταξινομητές που παράγουν την πιθανότητα με την οποία κάποια περίπτωση ανήκει σε μια κλάση, η ταξινόμηση μιας περίπτωσης γίνεται με βάση κάποιο κατώφλι που έχουμε θέσει και το οποίο καθορίζει την απόδοση του και συνεπώς την απεικόνιση του στο χώρο ROC.
  - ❑ Μεταβάλλοντας τις τιμές του ορίου (threshold), παίρνουμε μια καμπύλη (Καμπύλη ROC)
    - ✓ Όταν το όριο ελαττώνεται, τότε περισσότερα δεδομένα ταξινομούνται ως θετικά, αυξάνοντας τα TPs αλλά και τα FPs και το σημείο στην καμπύλη μετακινείται προς τα δεξιά. Στο 0 θα έχουμε FP και TP 100%.
    - ✓ Όταν το όριο αυξάνεται, τότε το σημείο στην καμπύλη μετακινείται προς τα αριστερά. Στην μέγιστη τιμή, θα έχουμε FP σχεδόν 0, αλλά και το TP το ίδιο!
  - ❑ Ιδανικό σημείο, αυτό που βρίσκεται στην κορυφή της καμπύλης ROC (φαίνεται στο σχήμα)
    - ✓ Στην Python: *metrics.RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=roc_auc)*
    - ✓ Παράδειγμα: http://www.anaesthetist.com/mnm/stats/roc/Findex.htm

Μετρική AUC (Area Under Curve) ή AUROC (the area under the ROC curve)

$$AUC = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right)$$

- Υπολογίζει το ποσοστό του χώρου που βρίσκεται κάτω από την καμπύλη
  - ✓ Provides an aggregate measure of performance across all possible classification thresholds
  - ✓ Απεικονίζει την καμπύλη με μια αριθμητική τιμή και παίρνει τιμές από 0 έως 1
  - ✓ Η διαγώνια γραμμή έχει AUC = 0,5. Κάθε ταξινομητής καλύτερος της τυχαίας πρόβλεψης έχει AUC > 0,5.
  - ✓ Όσο μεγαλύτερη περιοχή AUC έχει ένας ταξινομητής τόσο καλύτερος είναι
  - ✓ Στην Python: roc_auc_score (y_true, y_score)



Καμπύλη Precision-Recall (PR) (PR Curve)

- Η γραφική παράσταση της *ευστοχίας* (*precision*) έναντι της *ανάκλησης* (*recall*) όταν μεταβάλλεται το κατώφλι ταξινόμησης. Όσο υψηλότερα είναι τόσο καλύτερο είναι το μοντέλο.
  - ✓ In other words, the PR curve contains TP/(TP+FN) on the y-axis and TP/(TP+FP) on the x-axis.
  - ✓ precision_recall_curve (y_true, y_score)
- Ο στόχος είναι να πάρουμε ένα μοντέλο στην επάνω αριστερή γωνία όπου έχουμε μόνο true positives χωρίς καθόλου false positives και false negatives. Τέλειος ταξινομητής.
- Η καμπύλη PR χρησιμοποιείται όταν τα αρνητικά παραδείγματα είναι πολύ περισσότερα από τα θετικά (ανισορροπία κλάσεων), καθώς δεν λαμβάνει υπόψη τα true negatives (TN) αφού δεν συμμετέχουν ούτε στο *precision* ούτε στο *recall*.
  - ✓ Αλλιώς χρησιμοποιούμε την καμπύλη ROC.

Περίπτωση πολλαπλών κλάσεων

- Όταν έχουμε πολλές κλάσεις (N) τότε ο Πίνακας Σύγχυσης έχει διάσταση N × N και ορίζεται όπως και στην περίπτωση των 2 κλάσεων

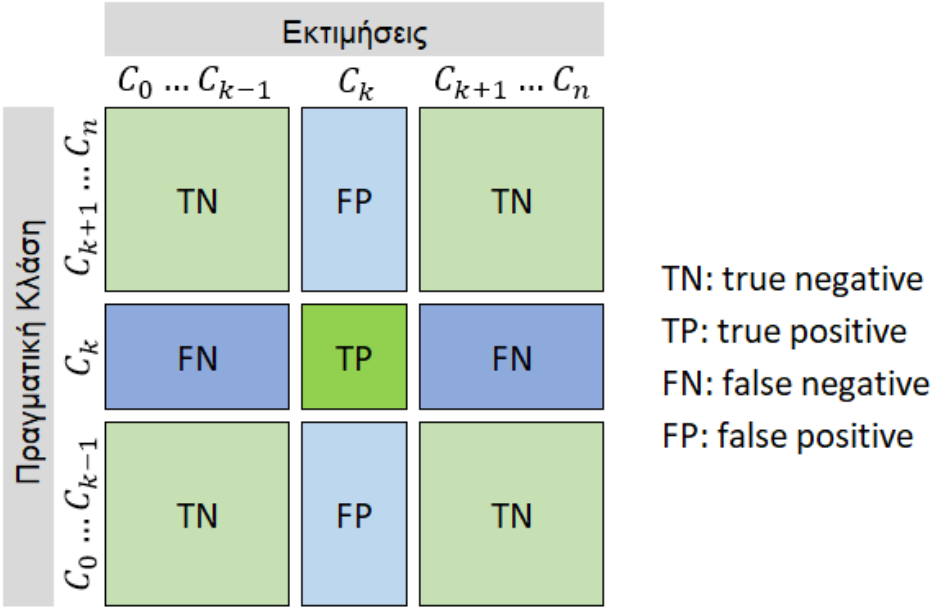
	Εκτιμώμενη κλάση			
Πραγματική κλάση	C=1	C=2	…	C=N
C=1	**n_{11}**	n_{12}	…	n_{1N}
C=2	n_{21}	**n_{22}**	…	n_{21}
…	…	…	**…**	…
C=N	n_{N1}	n_{N2}	…	**n_{NN}**

- Κάθε στοιχείο του πίνακα περιέχει τον αριθμό των δεδομένων που ταξινομήθηκαν στην αντίστοιχη κλάση
- Η διαγώνιος περιέχει τις σωστές προβλέψεις ενώ τα στοιχεία $i \neq j$ περιέχουν τα δεδομένα της κλάσης *i* που ταξινομήθηκαν λανθασμένα ως κλάση *j*.
- Υπολογίζουμε τα TP, FN, FP και TN από τον πίνακα ενδεχομένων για κάθε κλάση χωριστά:
  - TP: Τα στοιχεία της κύριας διαγώνιου
  - FN: Τα στοιχεία που βρίσκονται στην ίδια γραμμή με την κλάση που ελέγχουμε (εκτός του στοιχείου που βρίσκεται πάνω στην κύρια διαγώνιο)

Περίπτωση πολλαπλών κλάσεων(συνέχ.)

- FP: Τα στοιχεία που βρίσκονται στην ίδια στήλη με την κλάση που ελέγχουμε (εκτός του στοιχείου που βρίσκεται πάνω στην κύρια διαγώνιο)
- TN: Όλα τα υπόλοιπα στοιχεία του πίνακα, δηλαδή εκτός των γραμμών, στηλών και διαγώνιου, που αφορούν στην κλάση που ελέγχουμε

❖ Υπολογισμός μετρικών σε πίνακα ενδεχομένων Ν κλάσεων για την κλάση C_k:



Μικρο- και Μακρο- μέση τιμή

Micro- and Macro-average of Precision, Recall and F-Score

- Όταν έχουμε
  - Πολλά διαφορετικά σύνολα δεδομένων στα οποία εφαρμόζεται ο αλγόριθμος μας ή
  - Προβλήματα με πολλές κλάσεις
- Τότε υπάρχουν δύο μέθοδοι για τον υπολογισμό της συνολικής τους απόδοσης:
  - Ο μικρο-υπολογισμός (micro averaging) και
  - Ο μακρο-υπολογισμός (macro averaging)
- Οι οποίες υπολογίζουν τη *μέση ευστοχία* (*average precision*) και τη *μέση ανάκληση* (*average recall*) και στη συνέχεια τον αρμονικό τους μέσο (F-Score).
- Αρχικά υπολογίζουμε τις μετρικές TP, FN, FP και TN για κάθε σύνολο δεδομένων ή για κάθε κλάση χωριστά (με τη διαδικασία που περιγράφηκε προηγουμένως) και στη συνέχεια:
  - Στον μικρο-υπολογισμό, βρίσκουμε τα επιμέρους αθροίσματα όλων των μετρικών (TP, FP, TN και FN) και από αυτά υπολογίσουμε το precision και το recall
    - ✓ Είναι ιδανικός όταν δίνουμε μεγαλύτερη βαρύτητα σε κλάσεις με περισσότερα παραδείγματα
  - Στον μακρο-υπολογισμό, υπολογίζουμε τις μετρικές precision και recall για κάθε περίπτωση (δηλαδή για κάθε σύνολο δεδομένων ή για κάθε κλάση) και στη συνέχεια υπολογίζουμε τη μέση τιμή τους.
    - ✓ Είναι ιδανικός όταν όλες οι κλάσεις είναι το ίδιο σημαντικές, ανεξάρτητα του πλήθους τους.
  - Computing micro and macro f1 score using sklearn

❖ Παράδειγμα

- ❑ Έστω ότι σε ένα πρόβλημα με δύο κλάσεις ή από την εφαρμογή του αλγορίθμου σε δύο σύνολα δεδομένων, προκύπτουν τα ακόλουθα αποτελέσματα:
  - ✓ True Positive (TP1)=12, False Positive (FP1)=9, FalseNegative (FN1)=3
  - ✓ True Positive (TP2)=50, False Positive (FP2)=23, False Negative (FN2)=9
- ❑ Τότε το precision (P) και το recall (R) θα είναι P1=57.14 και R1=80 αντίστοιχα στην πρώτη περίπτωση και P2=68.49 και R2=84.75 στη δεύτερη.
- ❑ Η μέση ευστοχία (average precision) και η μέση ανάκληση (average recall) με τον μικρο-υπολογισμό θα είναι: $\text{Micro} - \text{average of precision} = \frac{\text{TP1+TP2}}{\text{TP1+TP2+FP1+FP2}} = \frac{12+50}{12+50+9+23} = 65.96$

$$\text{Micro} - \text{average of recall} = \frac{\text{TP1} + \text{TP2}}{\text{TP1} + \text{TP2} + \text{FN1} + \text{FN2}} = \frac{12 + 50}{12 + 50 + 3 + 9} = 83.78$$

  - ✓ Ο Micro-average F-Score θα είναι απλά ο αρμονικός μέσος των δυο ανωτέρω μετρικών, δηλαδή:

$$\text{F}_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = 2 \cdot \frac{65.96 \cdot 83.78}{65.96 + 83.78} = 73.80$$

- ❑ Η μέση ευστοχία (average precision) και μέση ανάκληση (average recall) με τον μακρο-υπολογισμό θα είναι απλά η μέση τιμή τους: $\text{Macro} - \text{average of precision} = \frac{\text{P1+P2}}{2} = \frac{57.14+68.49}{2} = 62.82$

$$\text{Macro} - \text{average of recall} = \frac{\text{R1} + \text{R2}}{2} = \frac{80 + 84.75}{2} = 82.25$$

  - ✓ Ο Macro-average F-Score θα είναι απλά ο αρμονικός μέσος των δυο ανωτέρω μετρικών, δηλαδή:

$$\text{F}_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = 2 \cdot \frac{62.82 \cdot 82.25}{62.82 + 82.25} = 71.23$$

Ζ) Μετρικές Αξιολόγησης Παρεμβολής

- Στην παλινδρόμηση τόσο οι πραγματικές Y όσο και οι προβλεπόμενες $\hat{Y}$ είναι πραγματικοί αριθμοί, οπότε δεν έχει νόημα να μιλάμε για μετρικές όπως η ακρίβεια, ευστοχία, κλπ, καθώς πρακτικά δεν θα επιτύχουμε ποτέ $Y = \hat{Y}$.
  - Απαιτούνται άλλα κριτήρια για τον υπολογισμό του σφάλματος, δηλαδή της διαφοράς μεταξύ της πραγματικής τιμής και της εξόδου του μοντέλου.
- Υπάρχει μια πληθώρα μετρικών, όπως:
  - Μέσο απόλυτο σφάλμα (Mean Absolute Error) ή μέση απόλυτη απόκλιση (mean absolute deviation):

$$\text{MAD ή MAE} = \frac{1}{n}\sum_{t=1}^{n}\left|Y_t - \hat{Y}_t\right| = \frac{1}{n}\sum_{t=1}^{n}|e_t|$$

    - ✓ Μέση τιμή των απόλυτων αποκλίσεων των προβλεπόμενων από τις πραγματικές τιμές.
    - ✓ Δηλαδή λαμβάνει υπόψη μόνο τις απόλυτες τιμές των σφαλμάτων και όχι τις πραγματικές, αγνοώντας αν οι προβλεπόμενες τιμές είναι υποεκτίμηση ή υπερεκτίμηση των πραγματικών.
    - ✓ Python: sklearn.metrics.mean_absolute_error
  - Μέσο απόλυτο ποσοστιαίο σφάλμα (Mean Absolute Percentage Error):

$$MAPE = \frac{1}{n}\cdot\sum\frac{\left|Y_t - \hat{Y}_t\right|}{Y_t} = \frac{1}{n}\cdot\sum_{t=1}^{n}\frac{|e_t|}{Y_t}$$

    - ✓ Εξετάζει τη συμπεριφορά της απόλυτης τιμής του σφάλματος πρόβλεψης σε σχέση με την πραγματική τιμή.
  - Μέσο ποσοστιαίο σφάλμα(Mean Percentage Error):

$$MPE = \frac{1}{n}\cdot\sum_{t=1}^{n}\frac{Y_t - \hat{Y}_t}{Y_t} = \frac{1}{n}\cdot\sum_{t=1}^{n}\frac{e_t}{Y_t}$$

- ✓ Εξετάζει τη συμπεριφορά του σφάλματος της πρόβλεψης σε σχέση με την πραγματική τιμή
- ✓ Χρησιμοποιείται όταν ενδιαφερόμαστε να προσδιορίσουμε αν η μέθοδος πρόβλεψης είναι μεροληπτική, δηλαδή αν οι προβλέψεις είναι συστηματικά μεγαλύτερες ή μικρότερες από τις αντίστοιχες πραγματικές.
- ✓ Όσο πιο κοντά στο μηδέν είναι η τιμή του, τόσο πιο αμερόληπτη και καλή είναι η μέθοδος πρόβλεψης.

❑ Normalized Mean Absolute Error (nMAE or MAE%) (or Coefficient of Variation of MAE)

- ✓ Είναι το MAE κανονικοποιημένο ως προς τη μέση τιμή των πραγματικών τιμών
- ✓ $$n\text{MAE} = \frac{MAE}{\frac{1}{\text{n}}\sum_{t=1}^{\text{n}}|\text{Y}_\text{t}|} = \frac{\frac{1}{\text{n}}\sum_{t=1}^{\text{n}}|\text{Y}_\text{t}-\widehat{\text{Y}_\text{t}}|}{\frac{1}{\text{n}}\sum_{t=1}^{\text{n}}|\text{Y}_\text{t}|} = \frac{\sum_{t=1}^{\text{n}}|\text{e}_\text{t}|}{\sum_{t=1}^{\text{n}}|\text{Y}_\text{t}|}$$
- ✓ Difference with MAPE: nMAE is different from MAPE in that the average of mean error is normalized over the average of all the actual values
- ✓ MAPE v/s MAE% v/s RMSE
- ✓ Understanding the Benefits of nMAE over MAPE for Estimating Load Forecast Accuracy

❑ Άθροισμα τετραγώνων των αποκλίσεων (residual sum of squares - RSS) ή (sum of squares error - SSE):

$$\text{RSS ή SSE} = \sum_{\text{t=1}}^{\text{n}}\left(\text{Y}_t - \widehat{\text{Y}}_\text{t}\right)^2 = \sum_{\text{t=1}}^{\text{n}}\text{e}_\text{t}^2$$

❑ Μέσο τετραγωνικό σφάλμα (Mean Squared Error):

$$\text{MSE} = \frac{1}{\text{n}}\text{RSS} = \frac{1}{n}\cdot\sum_{t=1}^{n}e_t^2$$

- ✓ Λόγω του τετραγώνου δίνεται μεγαλύτερη έμφαση στα σφάλματα της πρόβλεψης και θεωρείται πιο αξιόπιστο κριτήριο (μετρική) αξιολόγησης από την MAE.
- ✓ Η μονάδα μέτρησης της MSE είναι εκφρασμένη στη μονάδα μέτρησης των τιμών των παρατηρήσεων υψωμένη όμως στο τετράγωνο.

- ✓ Για αυτόν το λόγο, μερικές φορές χρησιμοποιούμε την τετραγωνική της ρίζα που ονομάζεται, *τετραγωνική ρίζα μέσου σφάλματος τετραγώνου* (root mean square Error - RMSE):

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \cdot \sum_{t=1}^{n} e_t^2}$$

- ✓ RMSE: np.sqrt(sklearn.metrics.mean_squared_error)
  - In the preceding code, we use the mean_squared_error() function and then we take the square root of this answer by using the np.sqrt() function from the numpy package.

- ❑ Huber Loss: Η μετρική Huber Loss αφορά μια υβριδική μετρική η οποία συνδυάζει τα πλεονεκτήματα του MAE και του MSE. Αρχικά ορίζεται μια τιμή Δέλτα. Για σφάλματα μικρότερα από την τιμή αυτή, η μετρική χρησιμοποιεί το μέσο τετραγωνικό σφάλμα, ενώ για σφάλματα μεγαλύτερα της συγκεκριμένης τιμής, μεταβαίνει στο μέσο απόλυτο σφάλμα.
  - ✓ Η Huber Loss είναι ιδιαίτερα χρήσιμη για την ελαχιστοποίηση των επιδράσεων από δεδομένα που περιέχουν Εκτός Ορίων Σημεία (Outliers), δηλαδή παρατηρήσεις που αποκλίνουν σημαντικά από τις υπόλοιπες τιμές και μπορεί να επηρεάσουν δυσανάλογα την απόδοση του μοντέλου.
  - ✓ Χρησιμοποιείται κυρίως ως loss function γιατί τα αποτελέσματα της δεν είναι επεξηγήσιμα.
  - ✓ Ο τύπος της συγκεκριμένης μετρικής δίνεται παρακάτω:

$$L_\delta(y, f(x)) = \{\frac{1}{2}\, e_i^2, \qquad |e_i| \le \delta\, \frac{1}{2}\, \delta^2\}$$

- ❑ Μετρικές στην Python:
  - ✓ Metrics and scoring: quantifying the quality of predictions
- ❑ A Comprehensive Introduction to Evaluating Regression Models

Πρόσθετες μετρικές

- Σχετικό απόλυτο σφάλμα (Relative Absolute Error):

$$\mathrm{RAE} = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\sum_{i=1}^{n}|y_i - \bar{y}|}$$

  - ✓ Υπολογίζει το συνολικό απόλυτο σφάλμα του μοντέλου μας ως προς τη διακύμανση των τιμών των προβλέψεων.
  - ✓ Η μετρική παίρνει τιμές από 0 (ιδανική πρόβλεψη) έως το άπειρο.

- Συνολικό άθροισμα τετραγώνων (sum of squares total – SST):

$$\mathrm{SST} = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

  - ✓ Υπολογίζει το άθροισμα των τετραγώνων των αποκλίσεων των πραγματικών τιμών από τη μέση τιμή τους.
  - ✓ Μπορεί να θεωρηθεί ως το τετράγωνο του σφάλματος ενός απλοϊκού μοντέλου πρόβλεψης το οποίο δίνει ως πρόβλεψη τη μέση τιμή $\bar{y}$ των πραγματικών τιμών y_i και η οποία ορίζεται ως:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i.$$

- Σχετικό τετραγωνικό σφάλμα (relative squared error – RSE):

$$\mathrm{RSE} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = \frac{\mathrm{RSS}}{\mathrm{SST}}$$

  - ✓ Υπολογίζει το συνολικό άθροισμα των τετραγώνων των αποκλίσεων του μοντέλου μας ως προς το τετράγωνο του σφάλματος ενός απλοϊκού μοντέλου πρόβλεψης.
  - ✓ Όπως και η RAE, η RSE παίρνει συνήθως τιμές από 0 (ιδανική πρόβλεψη) έως 1. Τιμές μεγαλύτερες του 1 δεν είναι εύλογες.
  - ✓ Χρησιμοποιείται στην R-square

R-Squared (R^2)

- R-squared ή R^2 ("coefficient of determination" or "R squared"):

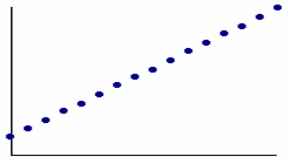
$$R^2 = 1 - \frac{RSS}{SST} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - RSE$$

- Η R-squared (R^2) περιγράφει τη διακύμανση (variance) των προβλέψεων σε σχέση με τη διακύμανση των πραγματικών τιμών (παρατηρήσεων) δηλ. το $(y_i - \bar{y})$.
- Ενώ η συσχέτιση (correlation) περιγράφει το βαθμό συσχέτισης των προβλέψεων με τις πραγματικές τιμές, η R^2 απεικονίζει το βαθμό που το μοντέλο μας περιγράφει τη διακύμανση των πραγματικών τιμών.
  - ✓ Δηλαδή υπολογίζει το πόσο καλά οι πραγματικές τιμές περιγράφονται από τις προβλέψεις του μοντέλου μας.
  - ✓ More specifically, R^2 gives you the percentage variation in y explained by x-variables.
  - ✓ It is the default score in scikit learn regressors
- Οι τιμές της κυμαίνονται συνήθως από 0 (κακή παρεμβολή) έως 1.0 (τέλεια παρεμβολή)
  - ✓ μπορεί να πάρει και αρνητικές τιμές όταν ή πρόβλεψη είναι χειρότερη από μια απλή πρόβλεψη που στηρίζεται στη μέση τιμή των πραγματικών τιμών.
- The coefficient of determination can be thought of as a percent.
  - ✓ It gives you an idea of how many data points fall within the results of the line formed by the regression equation.
  - ✓ The higher the coefficient, the higher percentage of points the line passes through when the data points and line are plotted.
  - ✓ If the coefficient is 0.80, then 80% of the points should fall within the regression line
- Python command (sklearn): sklearn.metrics.r2_score(y_true, y_predicted)

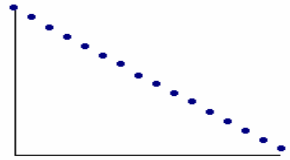
Συντελεστής Συσχέτισης (Correlation Coefficient)

- **Correlation** displays linear relationship between two random variables
  - Displays how strong of a linear relationship there is between two variables.
  - Important: Correlation does not mean causality!
- Ranges from -1 to 1
  - ✓ Guess what the values -1, 0 and 1 mean!
- Computing correlation:

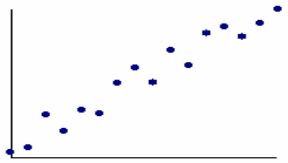
$$r = \frac{\sum_{j=1}^{N} w_j (X_j - \overline{X})(Y_j - \overline{Y})}{\sqrt{\left[\sum_{j=1}^{N} w_j (X_j - \overline{X})^2\right]\left[\sum_{j=1}^{N} w_j (Y_j - \overline{Y})^2\right]}}$$



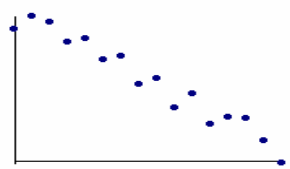
(r = +1,0)



(r = -1,0)



(r = +0,9)

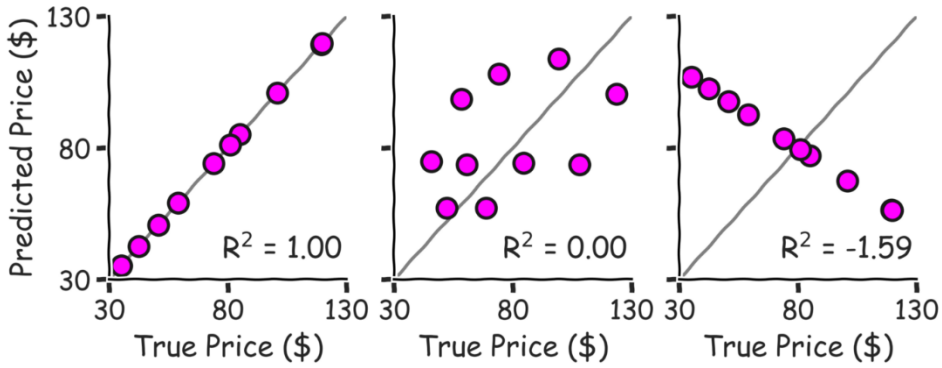


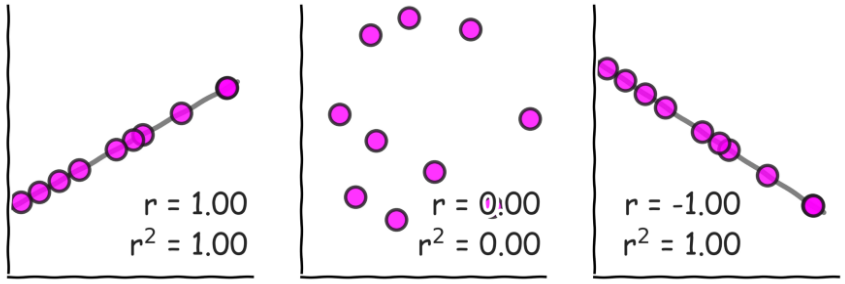
r= - 0,9)

  - ✓ Spearman and Pearson are well known statistical tests which measure the correlation between two variables
  - ✓ Unlike the Pearson correlation, the Spearman correlation does not assume that both datasets are normally distributed.
  - ✓ Python commands (scipy): **scipy.stats.spearmanr** (Spearman), **scipy.stats.pearsonr** (Pearson)
  - ✓ Python command (Pandas): **corr()** (Pearson)
- Used for:
  - Evaluating the quality of a prediction model, i.e. how much true and predicted values are related
  - Predicting missing feature values from other correlated values
  - Removing highly correlated features to each other so as not to produce misleading results
  - Removing low correlated features to target

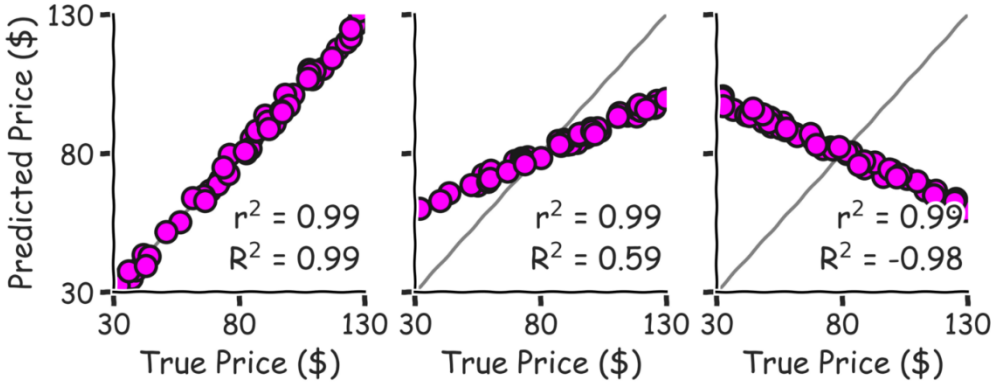
r vs. R-Squared: What's the Difference?

- r: Is the correlation between a predictor variable, x, or the observed (actual) values of the response variable y, and the predicted values of the response variable y made by the model.
- R^2: Is the proportion of the variance in the response variable that can be explained by the predictor variable in the regression model.
- Unlike the Pearson correlation coefficient (r), the coefficient of determination (R^2) measures how well the predicted values match (and not just follow) the observed values.
- Since R^2 indicates the distance of points from the line, it does depend on the magnitude of the numbers (unlike r and r^2).





- In some cases, r^2 is identical to R^2, but not always
  - https://towardsdatascience.com/r%C2%B2-or-r%C2%B2-when-to-use-what-4968eee68ed3
  - https://www.statology.org/r-vs-r-squared/



Τυπική Απόκλιση και Διακύμανση (ή Διασπορά)

- Στη στατιστική, η τυπική απόκλιση (σ) είναι ένα μέτρο που χρησιμοποιείται για να υπολογιστεί το ποσό της μεταβολής ή της διασποράς ενός συνόλου τιμών δεδομένων.
  - Η τυπική απόκλιση μιας τυχαίας μεταβλητής, ενός στατιστικού πληθυσμού, ενός συνόλου δεδομένων ή μιας κατανομής πιθανότητας, είναι η τετραγωνική ρίζα της διακύμανσης της (ή αλλιώς διασποράς).
  - Για ένα πεπερασμένο σύνολο αριθμών, η απόκλιση βρίσκεται λαμβάνοντας την τετραγωνική ρίζα του μέσου όρου των τετραγώνων των αποκλίσεων των τιμών από τη μέση τιμή τους.
    - ✓ Δηλαδή βρίσκουμε τη μέση τιμή (μέσο όρο) του συνόλου των αριθμών και στη συνέχεια υπολογίζονται τα τετράγωνα των αποκλίσεων του κάθε στοιχείου από τη μέση τιμή:
  - Η διακύμανση είναι ο μέσος των τιμών αυτών και η τυπική απόκλιση είναι η τετραγωνική ρίζα της διακύμανσης.
  - Η διακύμανση είναι η αναμενόμενη τιμή της τετραγωνικής απόκλισης μιας τυχαίας μεταβλητής από τη μέση τιμή, και
  - άτυπα μετρά πόσο μακριά ένα σύνολο (τυχαίων) αριθμών απλώνεται από τη μέση τιμή του.
- Εκτός από την έκφραση της μεταβλητότητας του πληθυσμού, η τυπική απόκλιση συνήθως χρησιμοποιείται για τη μέτρηση της εμπιστοσύνης στα στατιστικά συμπεράσματα.
- No need to compute both in a single problem - one is enough!
  - Variation has pretty mathematical properties that make it easy to work with in theoretical contexts
  - SD is easier for interpretation

Variance & Standard Deviation

- **Variance** measures how far the numbers of a set spread out away from its mean
  - Formal definition: $Var(X) = E[(X - \mu)^2]$, where $\mu = E[X]$
    - ✓ Variance of random variable *X* is the expected value of squared deviation from the mean of *X*
    - ✓ Python commands (numpy): **var()** method
- **Standard Deviation (SD)** is <u>almost</u> the same thing – it measures the amount of variation in a set of values
  - Formal definition: $\sigma = \sqrt{Var(X)}$
    - ✓ SD is the square root of standard deviation (so sometimes Variance is represented as σ^2)
    - ✓ Python commands (numpy): **std()** method
- No need to compute both in a single problem - one is enough!
  - Variation has pretty mathematical properties that make it easy to work with in theoretical contexts
  - SD is easier for interpretation

Ασκήσεις

- Υπολογίστε τα μέτρα εκτίμησης από τον ακόλουθο πίνακα ενδεχομένων (contingency) ενός ταξινομητή (Δεν χρειάζεται να γίνουν οι πράξεις)

TP=61	FN=30	91	TPR =
FP=39	TN=70	109	FPR =
100	100	200	ACC =

- Σε ένα πρόβλημα δυαδικής ταξινόμησης τα αποτελέσματα της αξιολόγησης ενός αλγορίθμου φαίνονται στον παρακάτω πίνακα ενδεχομένων ή σύγχυσης (confusion matrix).
  - Δώστε την ακρίβεια (Accuracy) και την ευστοχία (Precision) του ταξινομητή. (Δεν χρειάζεται να γίνουν οι πράξεις)

	Πρόβλεψη		
Πραγματική Κλάση		Ναι	Όχι
	Ναι	90	10
	Όχι	10	90

- Έστω δύο δυαδικοί ταξινομητές Α και Β με αποτελέσματα αξιολόγησης που φαίνονται στους παρακάτω πίνακες ενδεχομένων ή σύγχυσης (confusion matrices).

A	**Προβλεπόμενη Κλάση**		
Πραγματική Κλάση		**Ναι**	**Όχι**
	Ναι	80	20
	Όχι	30	70

B	**Προβλεπόμενη Κλάση**		
Πραγματική Κλάση		**Ναι**	**Όχι**
	Ναι	90	10
	Όχι	40	60

- Απεικονίστε τους ταξινομητές στο χώρο ROC
- Ποιόν ταξινομητή θα επιλέγατε με βάση την ανάλυση ROC;
- Ποιον θα επιλέγατε με βάση την ακρίβεια (ACC)
- Ποιον θα επιλέγατε με βάση το F-measure

…the end

Questions?