




“Difficult” datasets

Feature selection, evaluation metrics,
and class imbalance





Quick detour: MONAI



MONAI (<https://monai.io>)

- “A set of open-source, freely available collaborative frameworks built for accelerating research and clinical collaboration in Medical Imaging”
- Support for handling medical imaging formats (DICOM, etc.)
- Support for data anonymization
- Integration with tools, used by clinicians, for manual segmentation and labelling of images
- Support for distributed processing
- "Model Zoo": an incredibly broad and up-to-date of pre-trained (on public datasets) models (often, winners of specific challenges)

Feature selection

So many features

- In business, companies now collect large amounts of information on their customers and products
- In pharmaceutical research, thousands of features for each compound using quantitative structure-activity relationship (QSAR) methodology
- In biology, a vast array of biological predictors can be measured - *e.g.*, the “Omics” (genomics, proteomics, metabolomics, metagenomics, phenomics and transcriptomics)

What is feature selection?

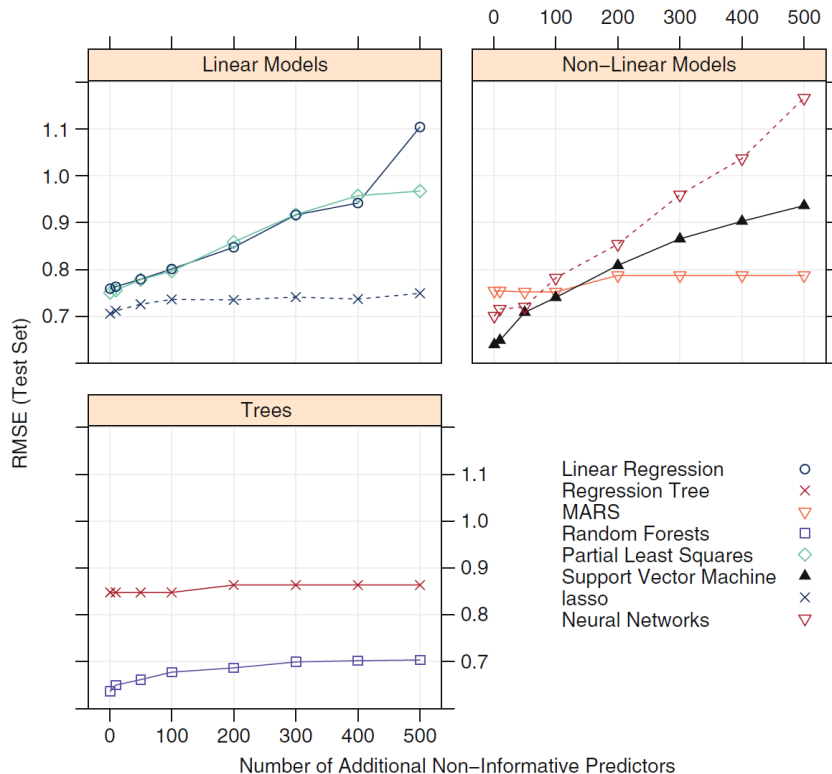
Choose the “best” subset of features (among the 2^P - P is the total number of features) for fitting our model

Types of “useless” features

- Non-informative features
 - Basically, such features do not “correlate” with what you want to predict (e.g., the number of your credit card is probably useless to determine your cancer risk)
 - In your model, they would act as “pure noise”
- Redundant features
 - Some features are highly correlated (e.g., weight and height)
 - Redundancy leads to more complex models (more parameters) yet adds little information
 - Highly unstable models, numerical errors, and degraded predictive performance
- We want to sort features according to their usefulness
 - Almost always, we can’t say: “this is non-informative/redundant. Period.”
 - We can try to say: this is more informative than this
 - Or, better: this is the set of features that work best together
- Many features + little data means discarding even very good features
 - In exchange, hopefully, for even better ones

Worse than useless

Adding non-informative features (predictors) degrades performance in many models



Classes of feature selection methods

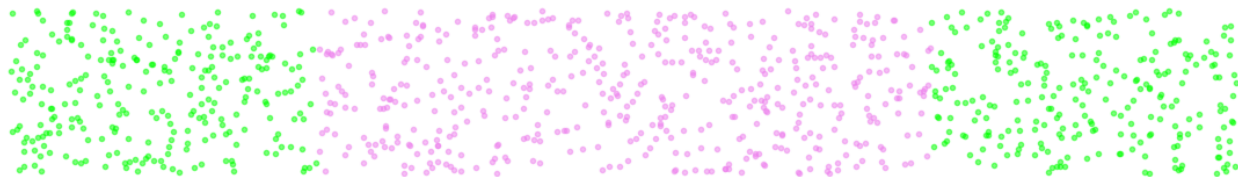
- Model agnostic methods (filters)
 - Computationally efficient
 - Many drawbacks
 - Basically, they are **no longer used** (except for exploratory analysis)
- Regularization methods (embedded)
 - Based on simple models
 - Computationally efficient
 - Yet quite powerful
- Model-based methods (wrappers)
 - Powerful, but computationally demanding
 - Higher risk of overfitting

“Unsupervised” filters

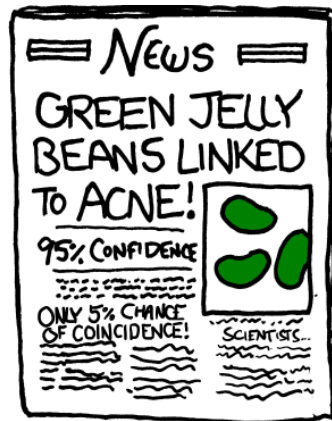
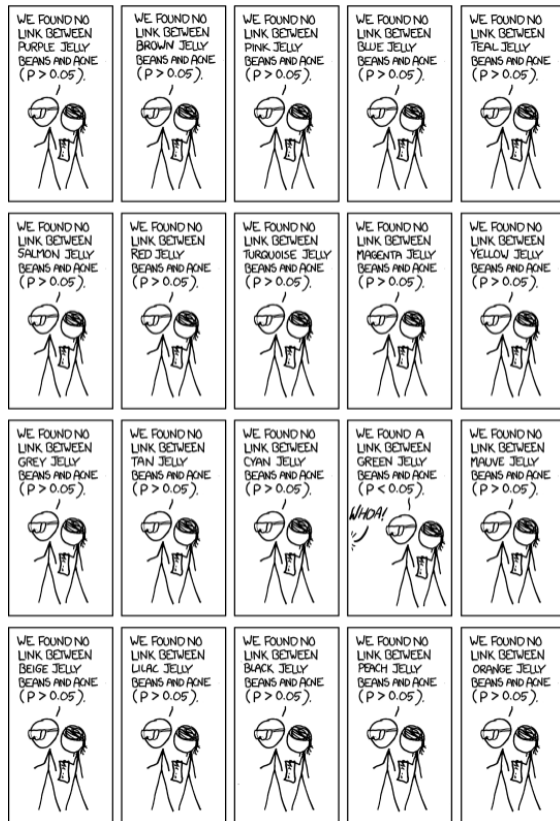
- “Near-zero variance” features
 - Attribute “male” is not very useful if we want to predict heart attacks in males (“zero variance”)
 - Near-zeros can be problematic (numerical problems, “pure” batches)
 - Yet: rare genetic profiles could be very useful (for a minority of cases)
- Correlated features
 - A and B: the two most correlated features (absolute value)
 - $c_A(c_B) < \text{correlation of feature A (B) with all the remaining features}$
 - If $c_A > c_B$: discard A; else: discard B
 - Repeat
 - **Or: Good ol’ PCA**
 - Yet: “fatness” is weight - α height...

“Supervised” filters

- Feature x , prediction y
- Measure how strong the relationship between x and y is
 - Pearson correlation, ANOVA, chi-squared test, t-test, mutual information, etc.
- Several problems
 - When to discard a feature? P-test? (see next slide!)
 - We can keep the “best” k — how to choose k ?
 - Significant is NOT (necessarily) “important” (“car lighter” and “price”)
 - What about “non-linear” features?



The p-value fallacy



**Bonferroni correction helps...
but probably too rigid**

Embedded methods

- LASSO (basically, L1 regularization)
 - Fit a simple model (like linear regression, logistic regression)
 - Discard all the features whose weight is very small (normalized features!)
 - The larger λ , the fewer features will survive
- Random Forest
 - Discard all the features not selected by the random forest
 - Or use average decrease of impurity
 - Or Shapley values
 - Different teams = different values. How much does each player contribute on average across all possible teams?
 - They can be efficiently computed for a tree

L1 sparsifies (and L2 rescales)

L2 $f(\theta) \simeq \frac{(\theta - \bar{\theta})^2}{2} + \lambda \theta^2$

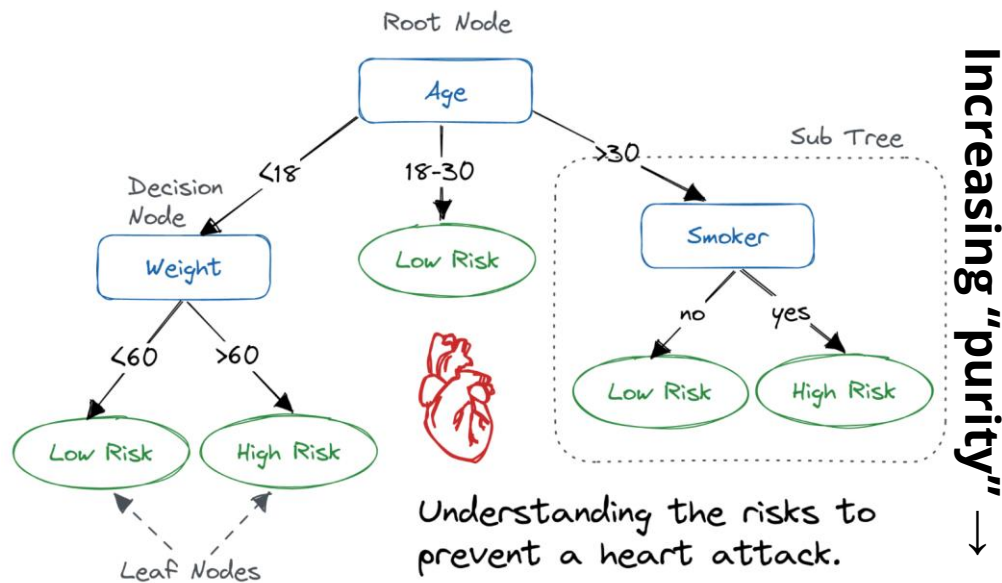
$$\theta = \frac{\bar{\theta}}{1 + 2\lambda}$$

L1 $f(\theta) \simeq \frac{(\theta - \bar{\theta})^2}{2} + \lambda |\theta|$

$$\theta = \bar{\theta} - \lambda \operatorname{sign}(\theta)$$

$$\text{If } |\bar{\theta}| < \lambda \implies \theta = 0$$

Decision trees and random forests



<https://www.datacamp.com/tutorial/decision-tree-classification-python>

Random forest

- Many shallow trees
- At each node, only a subset of features is considered

$$\hat{y}(x) = \frac{1}{T} \sum_i^T \text{tree}_i(x)$$

Wrappers

- Basic algorithm
 - Start with a subset of features
 - Add or subtract one of these features and fit the model ("**proposed change**")
 - Check if the new model is "better-enough" or "not-too-worse"
 - Accept or discard the **proposed change**
 - Repeat until we have to stop
- Add="forward selection", subtract="backward selection", both="stepwise"
 - We can add and subtract in various combinations
 - Simulated annealing, genetic algorithms
- How to measure "better-enough"
 - Performance (RMSE, cross entropy, etc.) + Complexity penalization (Akaike Information Criterion — AIC, Bayesian Information Criterion — BIC)
- Permutation importance
 - Take your model; shuffle the values of a feature; see how your performance is impacted

Feature selection and overfitting

Things favouring overfitting

- The data set is small
- The number of feature is large
- Large-variance models (e.g., deep networks)

Never ever use test data for feature selection

Too many levels (high cardinality)

- Categorical features with many possible values
 - Think: diagnosis, “previous employer”, postal codes, brand of a car, *etc.*
 - *Many* must be understood in comparison to the number of data points
- First: avoid one-hot encoding
- Binary encoding (and other forms of encoding)
- “Other” solution
- Grouping for effect (on part of the training set!)
- “Contrastive”-L1
- Expert grouping
- Boosted trees + “Fisher trick” (maximum homogeneity)
- Embedding
 - A priori embedding (for example, using LLM)
 - Learned embedding

Class imbalance

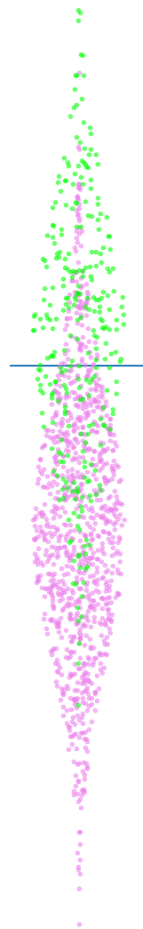
Class imbalance

- When in our dataset, one or more of the classes are very rare
- Examples
 - Online advertising: will the reader click on the ad? Clicks are like 2% of all visits
 - Drugs: of thousands of compounds, only a few are expected to be active
 - Frauds: we expect only a small fraction of “transactions” to be malicious
- Accuracy not a good metric
 - There is nothing wrong with accuracy. It is our intuition that can get derailed
 - If I say my model gets 99% accuracy, you’d probably say: great! (thank you)
 - Yet: if frauds rate is 1%, the **“blind” model** that always gives a “not-a-fraud” response reaches an accuracy of...? Yes, 99%
- It is difficult to make our model not too “blind”
 - Because in many respects there is too little to be gained (and too little information)
 - The more imbalance, the harder the problem gets

Classes of metrics

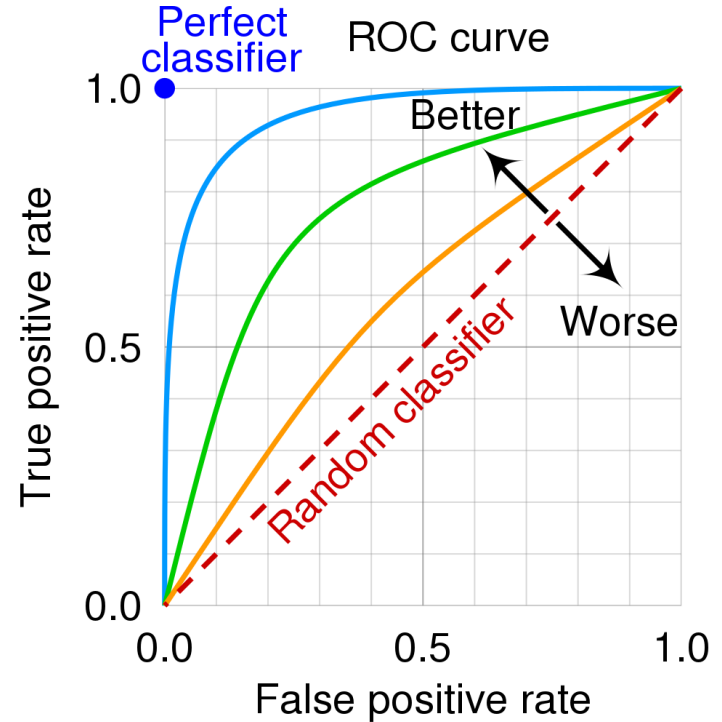
Classifier outputs a real value; higher = “belongs more to positive class”

- Threshold metrics
 - Accuracy, precision, recall, *etc.*
 - All the possible compositions of TP, FP, TN, and FN
- Ranking metrics
 - ROC Area Under the Curve (AUC), Precision-Recall AUC
- Probability metrics
 - Log-loss/cross entropy



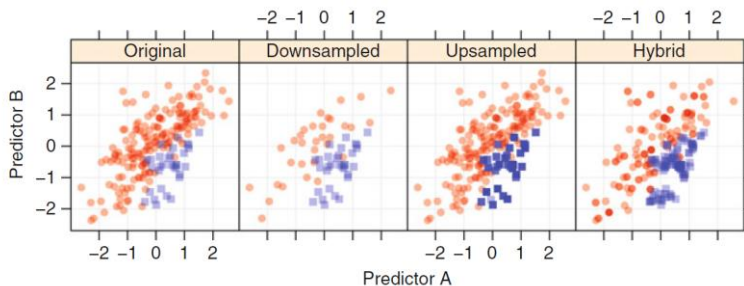
Receiver Operating Characteristic (ROC) curve

- ROC curve gives a graphical depiction of the diagnostic ability of a binary classifier as its discrimination threshold is varied
- High threshold: no “positives” ($TP = 0$, $FP = 0$)
- Low threshold: all “positives” ($TP / P = 1$, $FP / N = 1$)
- $AUC = \text{Probability that } f(\text{positive}) > f(\text{negative})$
- Changing decision threshold = different tradeoffs along the ROC



Dealing with imbalance

- Tuning hyperparameters
- Adjusting prior probabilities (Bayes)
- **Giving more weight to different classes** in your error function
 - Very effective (akin to up-sampling — see below)
- **Resampling techniques**
 - Up-sampling, down-sampling
 - SMOTE (**synthetic** minority over-sampling technique): down-sampling and up-sampling + K-nearest neighbours
 - All effective, yet no clear winner (different models requires different sampling techniques)





Ci vediamo lunedì prossimo (in Aula Marotta
– via del Castro Laurenziano, 10)!

