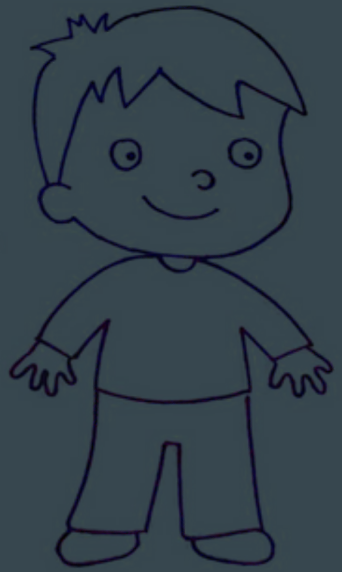
Machine Learning

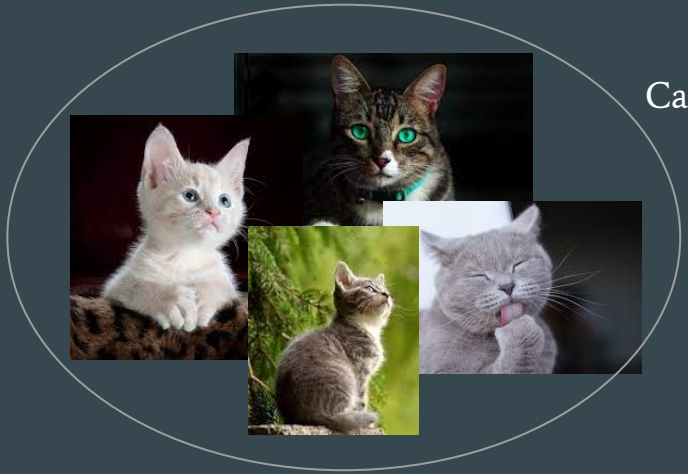
•••

Supervised learning

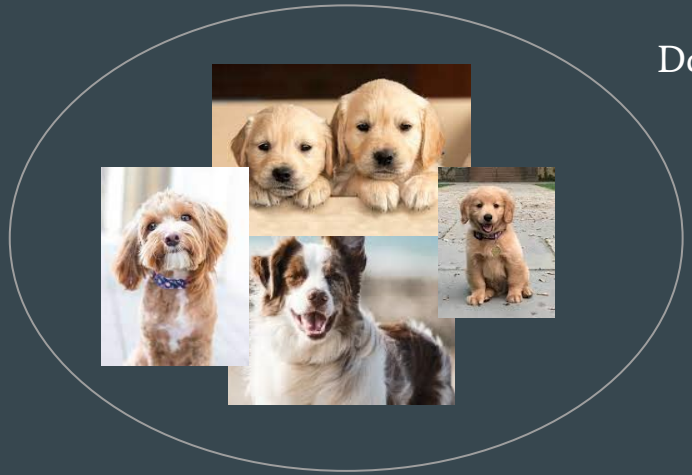
Machine Learning problems

Machine learning





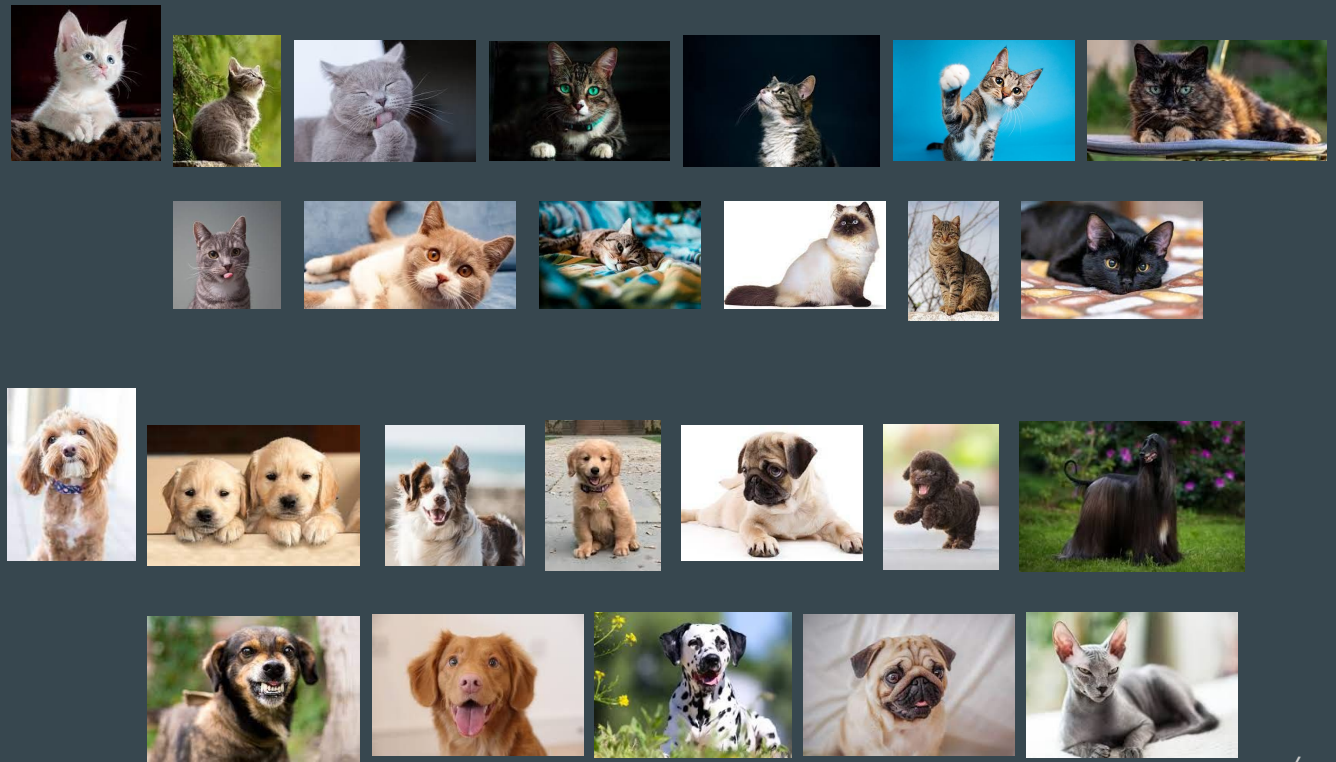
Cats



Dogs

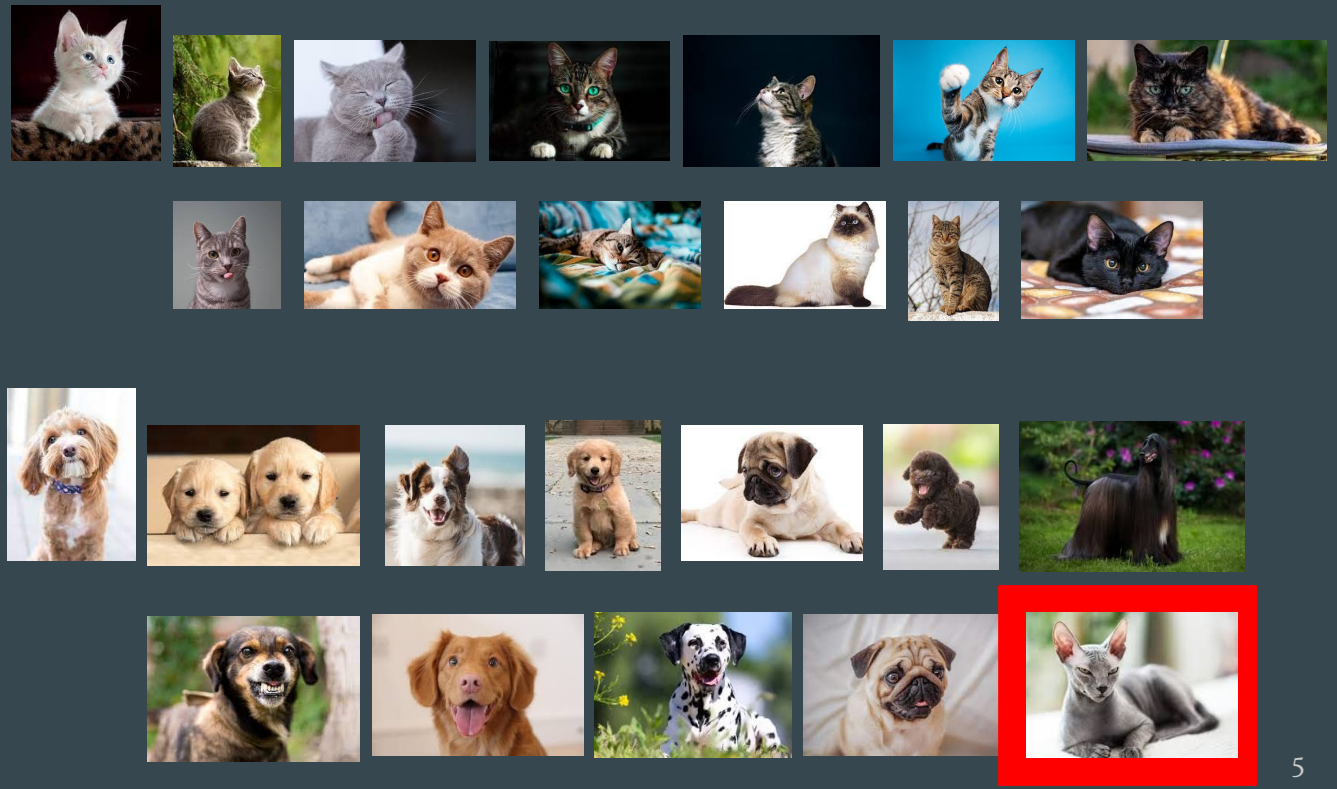
Machine learning

ML



Machine learning

ML



Supervised learning

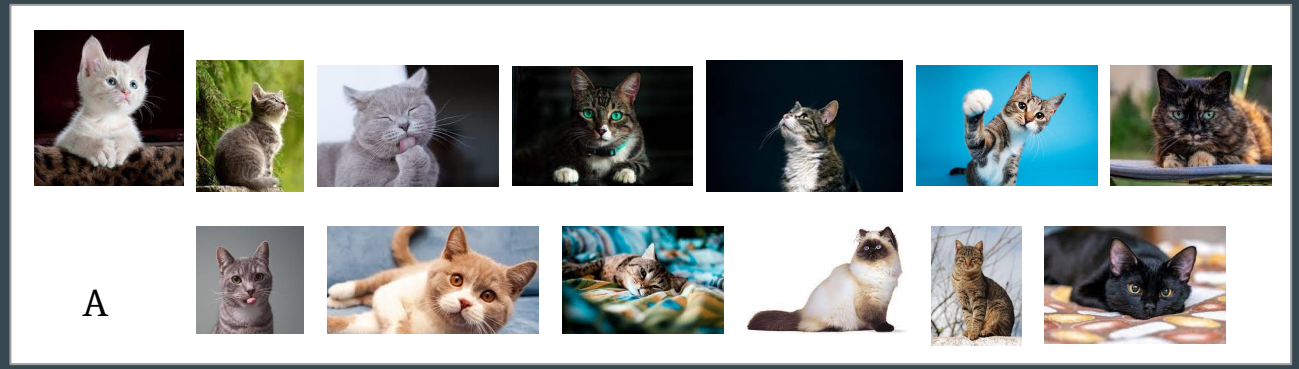
ML

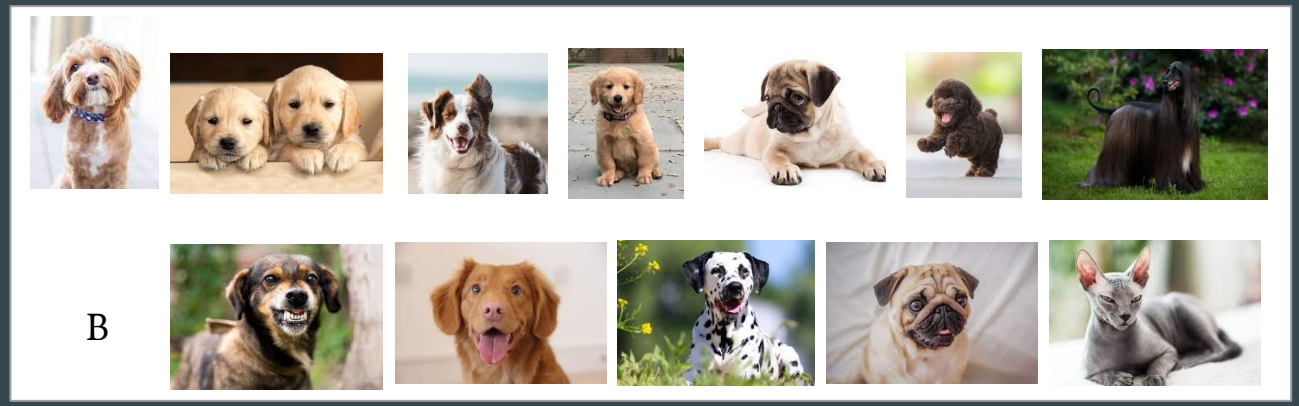
Cats

Dogs

Unsupervised learning

ML





Supervised learning

Supervised learning is a type of machine learning where the algorithm is trained on labeled data. Supervised learning is suitable for solving problems where the goal is to predict the output value based on the input data.

- There is a clear relationship between the input data and the output labels.
- Sufficient labeled data is available to train the algorithm.
- The problem can be defined as a classification or regression task.
- The goal is to make accurate predictions on new, unseen data.

Some examples of applications that use supervised learning include spam filtering, image recognition, speech recognition, and predicting housing prices

Unsupervised learning

Unsupervised clustering is used to group similar data points into clusters without any predefined labels or categories. Unsupervised clustering is suitable for solving problems where the goal is to discover patterns and relationships in the data.

- There is no clear relationship between the input data and the output labels.
- There is a large amount of unlabeled data available.
- The goal is to discover patterns and relationships in the data.
- The data can be grouped into natural clusters based on similarity.

Some examples of applications that use unsupervised clustering include image segmentation, clustering, dimensionality reduction.

Regression problems

Regression

Regression is used to determine the correlation between one dependent variable Y and a set of other variables X_1, X_2, ... which are considered independent variables. The goal of regression analysis is to understand how changes in the independent variables are associated with changes in the dependent variable, and to use this information to make predictions about the value of the dependent variable.

Regression investigates the correlation between variables observed in a data set, and quantifies whether those correlations are statistically significant or not.

Examples of problems that can be addressed with regression include prediction of product price, forecasting demand for a product and prediction of disease risk.

Regression

Regression can be linear, or non-linear and it can be distinguished in simple and multi based on the number of the independent variables.

Simple linear regression: $Y = a + b * X + u$

Multiple linear regression: $Y = a + b_1 * X_1 + b_2 * X_2 + b_3 * X_3 + \ldots + u$

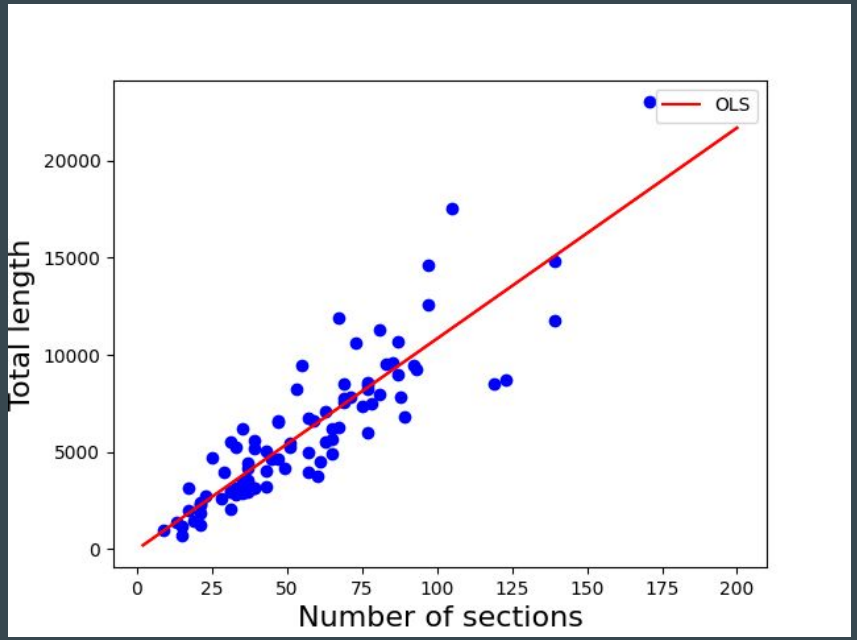
Y: dependent

X: independent

u: error

Linear Regression

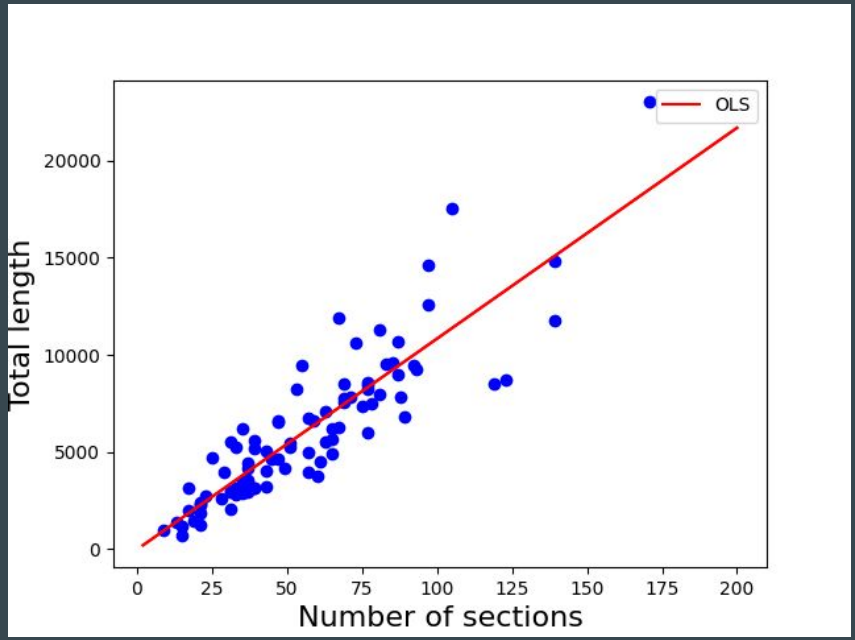
First we need to test if variables are correlated (remember the correlation statistical tests). Once we identify a valid correlation, we can use linear regression to identify what's the type of dependency that can describe our dataset.



Least squares (linear regression)

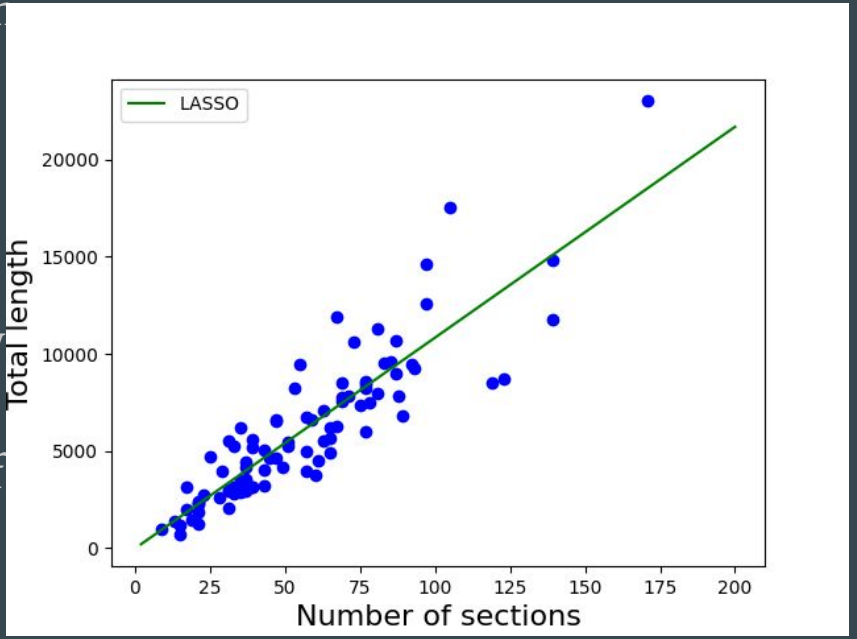
Least square method aims to minimize the sum of squared error between the predicted values and the actual values.

$$\min_{w} ||Xw - y||_2^2$$



Lasso

Lasso (least absolute shrinkage and selection operator) is a linear regression method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the model. LASSO regression adds a penalty term to the objective function of linear regression, which is a multiple of the sum of the **absolute values** of the coefficients.

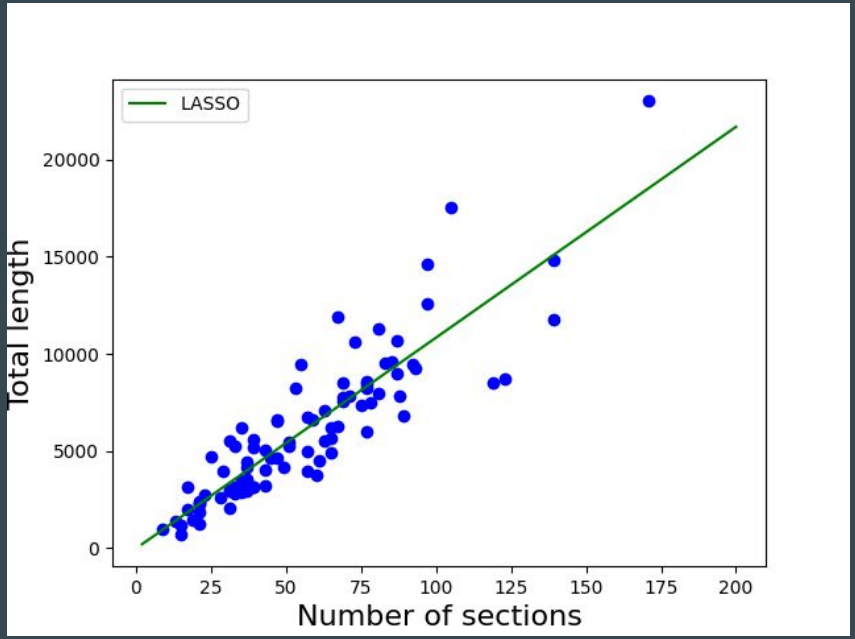


Lasso

The objective function of LASSO regression can be expressed as:

$$\min_{w} \frac{1}{2n_{\text{samples}}} ||Xw - y||_2^2 + \alpha ||w||_1$$

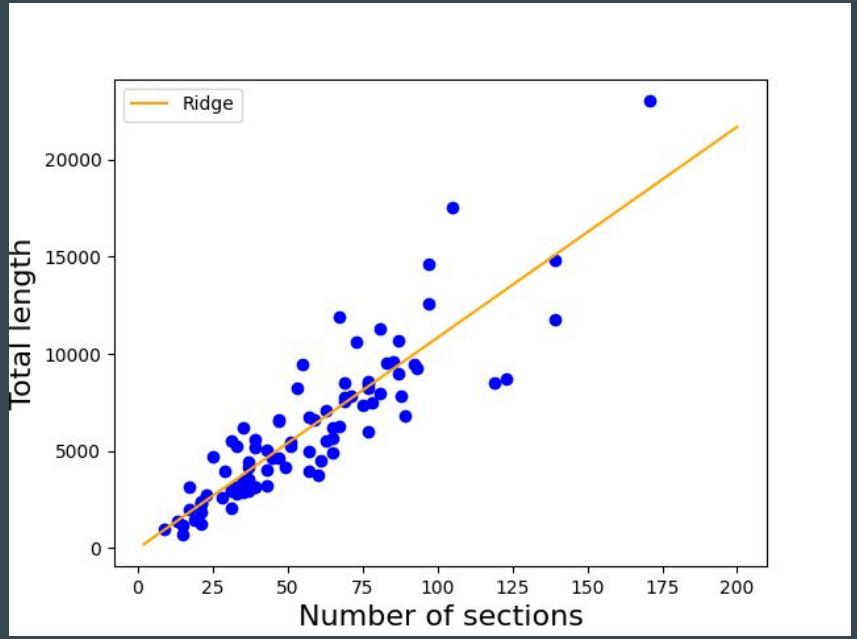
Coefficient Error term



Ridge

Ridge regression adds a penalty term to the objective function of linear regression, which is a multiple of the sum of the **squares** of the coefficients.

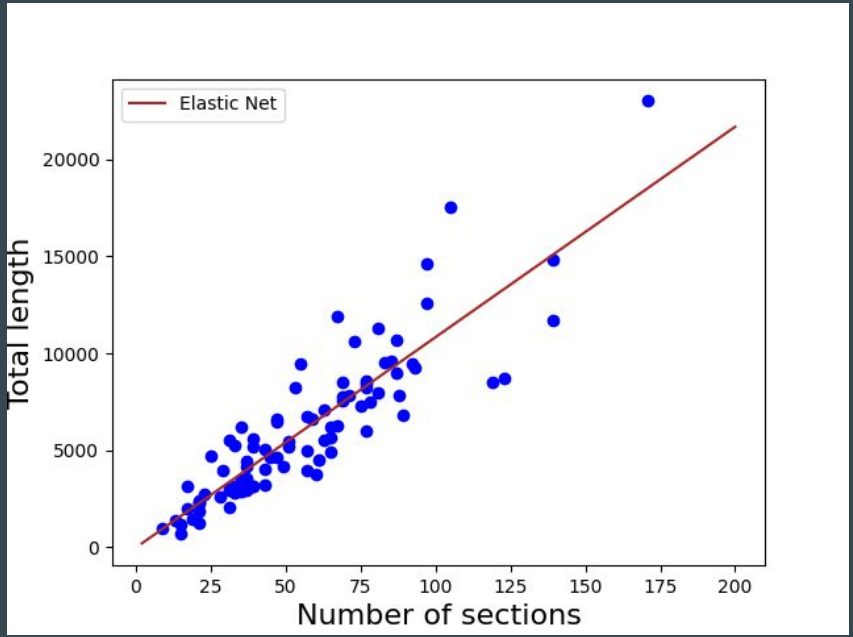
$$\min_w ||Xw - y||_2^2 + \alpha||w||_2^2$$



Elastic-net

Elastic net adds a penalty term to the objective function of linear regression, which is a weighted sum of the L1 (absolute value) and L2 (squared value) norms of the coefficients.

$$\min_w \frac{1}{2n_{\text{samples}}} ||Xw - y||_2^2 + \alpha\rho||w||_1 + \frac{\alpha(1-\rho)}{2}||w||_2^2$$



Linear Regression

Any observations for linear regression methods?

Linear Regression

Any observations for linear regression methods?

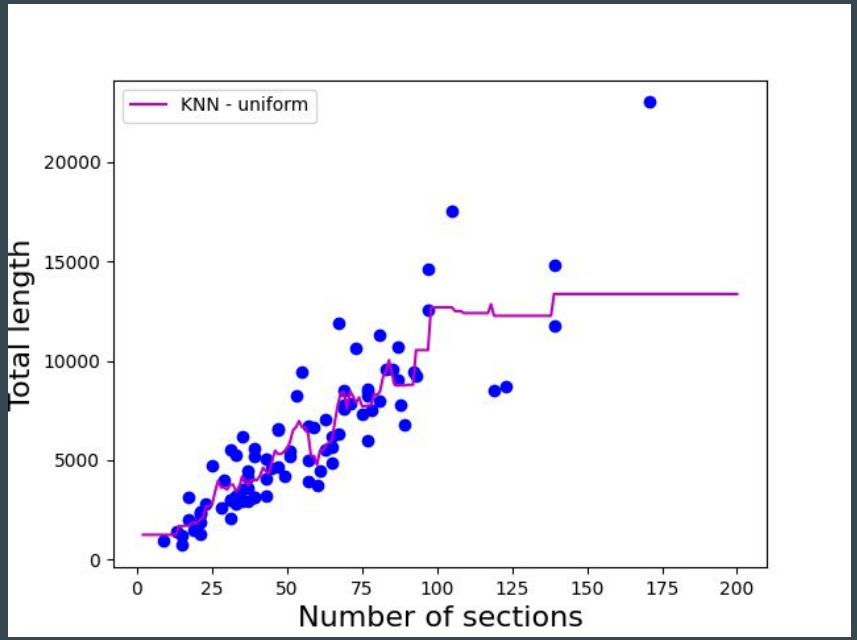
The specific methodology and the additional error term does not seem to have a significant effect on the results (for this particular problem).

However the results depend on the characteristics of the problem, so it's not always trivial to judge which method will perform better. It is important to keep in mind that the total error alone is not an indication for the method's power. For example, a line that connects all the points will introduce a minimal error, but will have very limited predictive power for new data. This effect is also known as overfitting.

K-Nearest neighbors (non-linear)

Nearest Neighbors Regression is a machine learning algorithm which predicts the value of a continuous variable based on local interpolation from the values of the nearest neighboring data points in the training dataset.

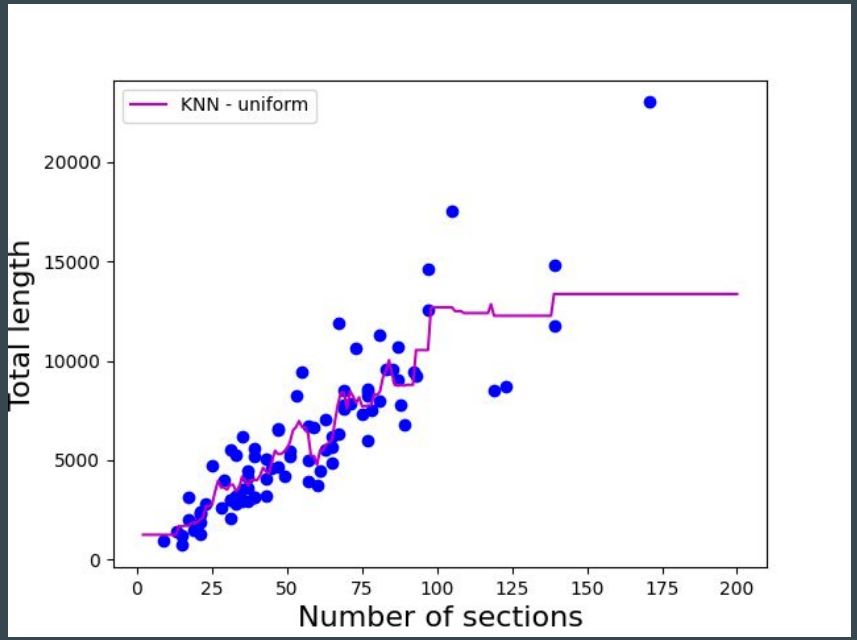
The algorithm calculates the distances between the new observation and all other observations in the training dataset. It then selects the k-nearest observations based on the minimum distance and computes the average value of their target variables to predict the target variable of the new observation.



K-Nearest neighbors (non-linear)

Nearest Neighbors results depend on the number of neighbors (k) as well as the weights that are used to calculate the target value from the k-nearest neighbors.

Uniform weights account all k-neighbors equally.

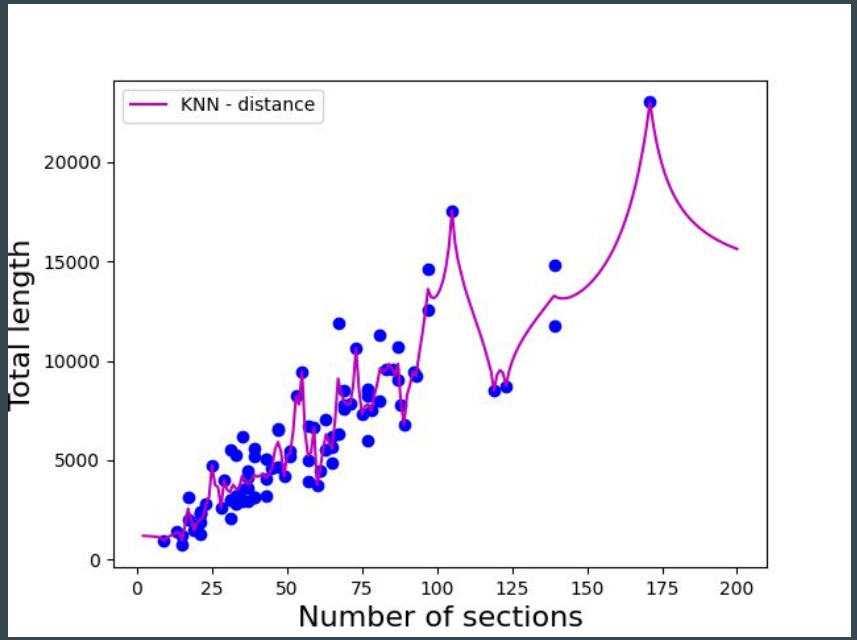


Nearest neighbors (non-linear)

Nearest Neighbors results depend on the number of neighbors (k) as well as the weights that are used to calculate the target value from the k-nearest neighbors.

Uniform weights account for all k-neighbors equally.

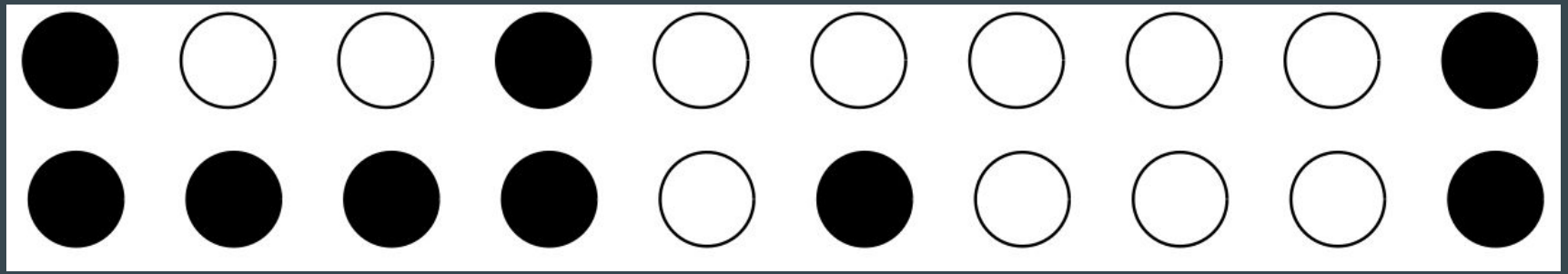
Distance dependent method introduces a weight to each point that is inverse to the distance from target point. Points close to the target count more than those that are further away.



Supervised classification results

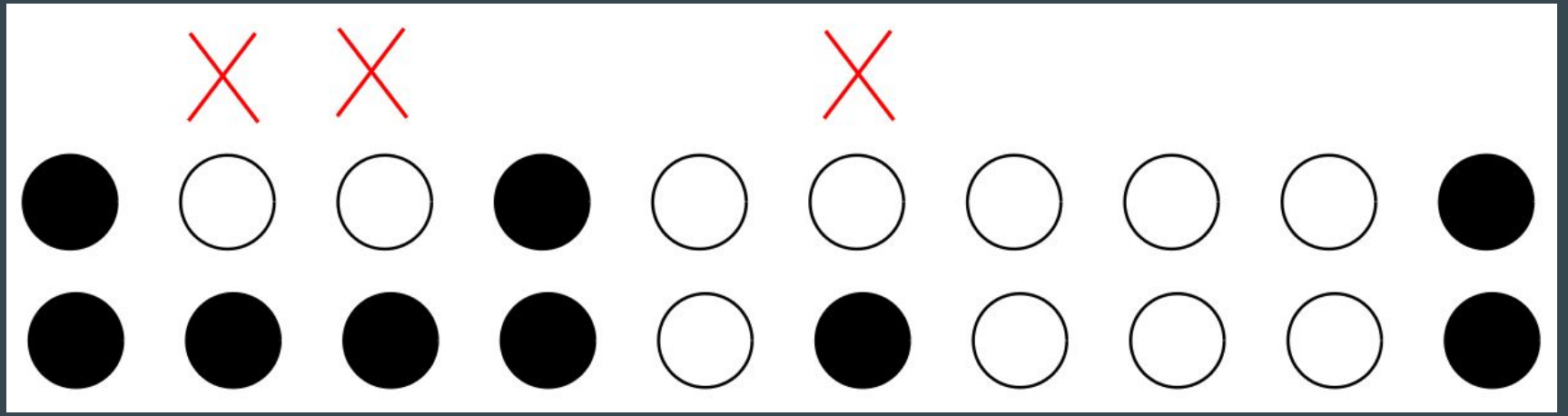
Metrics: accuracy

Accuracy is a metric used to evaluate the performance of a machine learning model for a classification task. It measures the proportion of correct predictions made by the model over the total number of predictions. For example, in a binary classification task, the top row is real data and the bottom is the model's predictions:



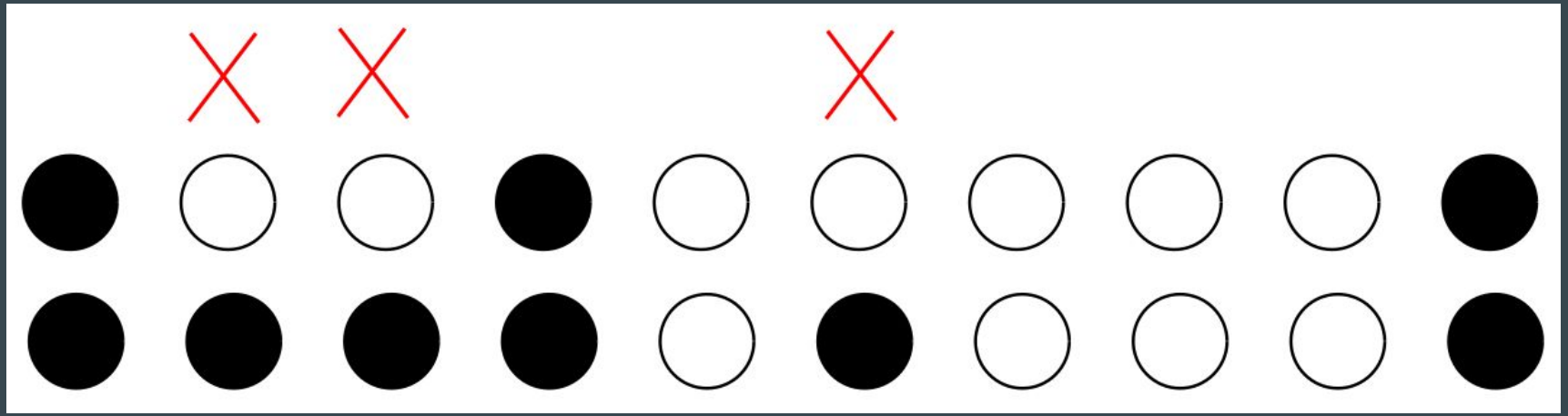
Metrics: accuracy

In this example, the accuracy is the number of correct predictions over the total number of predictions: 7 / 10 or 70%



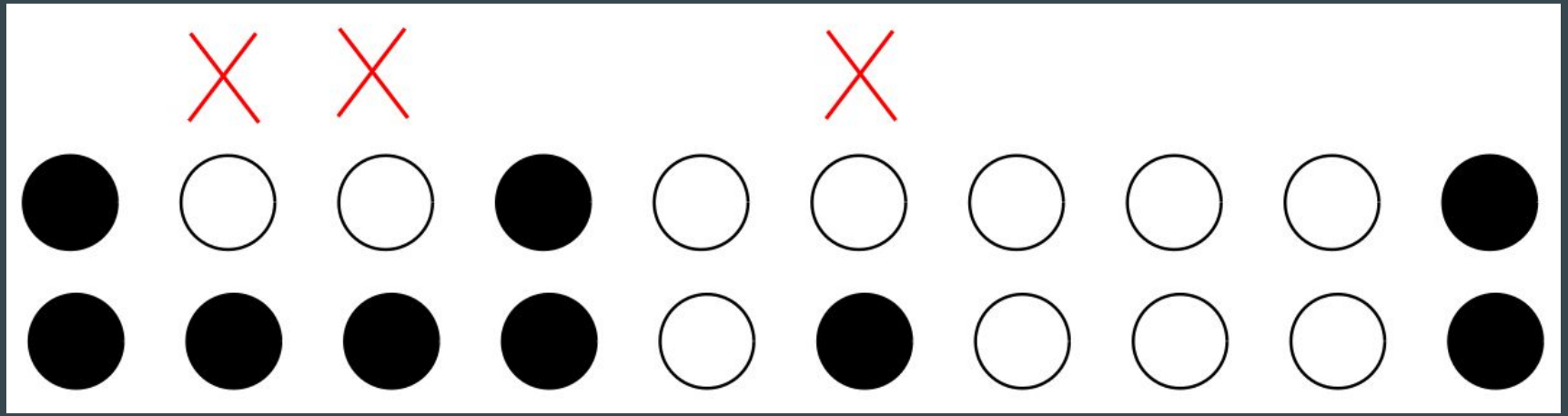
Metrics: balanced accuracy

Balanced accuracy is useful when the original labels are not balanced. For example, in a binary classification task when one class has 7 members and the other 3 members, like the previous example:



Metrics: balanced accuracy

Balanced accuracy is measured as the accuracy in class I plus the accuracy in class II divided by 2: (3/3 + 4/7) / 2 as opposed to (3 + 4) / 10 in simple accuracy. In this case the balanced accuracy will be ~78% as opposed to the accuracy of 70%.



Accuracy

Accuracy of a test is particularly important in cases of highly unbalanced datasets. For example, in a rare disease test:

0.3% of the population has a disease that needs to be treated.

Test A has 30% false positive and 1% false negative

Test B has 1% false positive and 30% false negative

Which test should be preferred?

Accuracy

For example, in a rare disease (0.3%) test:

Test A has 30% false positive and 10% false negative

Test B has 10% false positive and 30% false negative

	+ (0.3)	- (99.7)
Test A	30% error	10% error
Test B	10% error	30% error

Accuracy

For example in 10.000 samples, we would have 30 individuals with the disease and 9970 healthy individuals. Test A will detect 21 of 30 disease individuals while test B will detect all 30 disease individuals.

However, the simple accuracy of test A would be 1% while test B would be 30%.

	+ (0.3%)	- (99.7%)
Test A	21	9870
Test B	30	6979

Accuracy

For example in 10.000 samples, we would have 30 individuals with the disease and 9970 healthy individuals.

However, the simple accuracy of test A would be 99% while test B would be 70%

The balanced accuracy for test A would be 84% and for test B would be 85%

	+ (0.3%)	- (99.7%)
Test A	21	9870
Test B	30	6979

Accuracy

When computing accuracy, we need to account for the following factors:

1. Compute on the validation set (not on the training)
2. Take into account the number of samples for each class
3. Compare balanced and simple accuracy

It is not always evident which accuracy measurement is the best to use, and it often depends on the specific task.

Log-loss

Log-loss accuracy, also known as cross-entropy loss measures the difference between the predicted probabilities and the true class labels.

The log-loss function computes the logarithm of the predicted probability for the correct label, which is a value between 0 and 1. If the predicted probability is close to 1 for the correct label, the log-loss value will be close to 0, indicating high accuracy. Conversely, if the predicted probability is close to 0, the log-loss value will be high, indicating low accuracy.

Log-loss

In binary classification problems, the log-loss function for each sample is defined as the negative log-likelihood of the classifier given the true label:

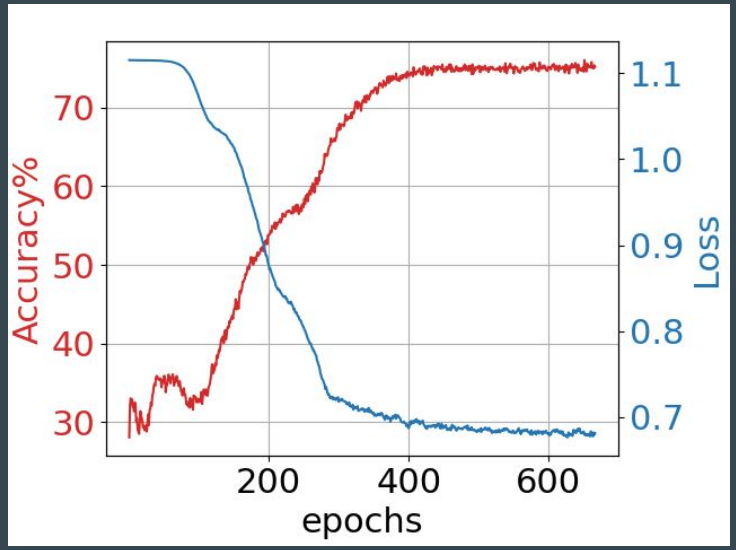
$$L_{\log}(y, p) = -\log \Pr(y|p) = -(y \log(p) + (1 - y) \log(1 - p))$$

where y is the true label (either 0 or 1), and p is the predicted probability of the positive class (usually denoted as class 1).

Log-loss accuracy is calculated as the average log-loss over all the samples in the test set. A lower log-loss indicates better performance of the model.

Log-loss

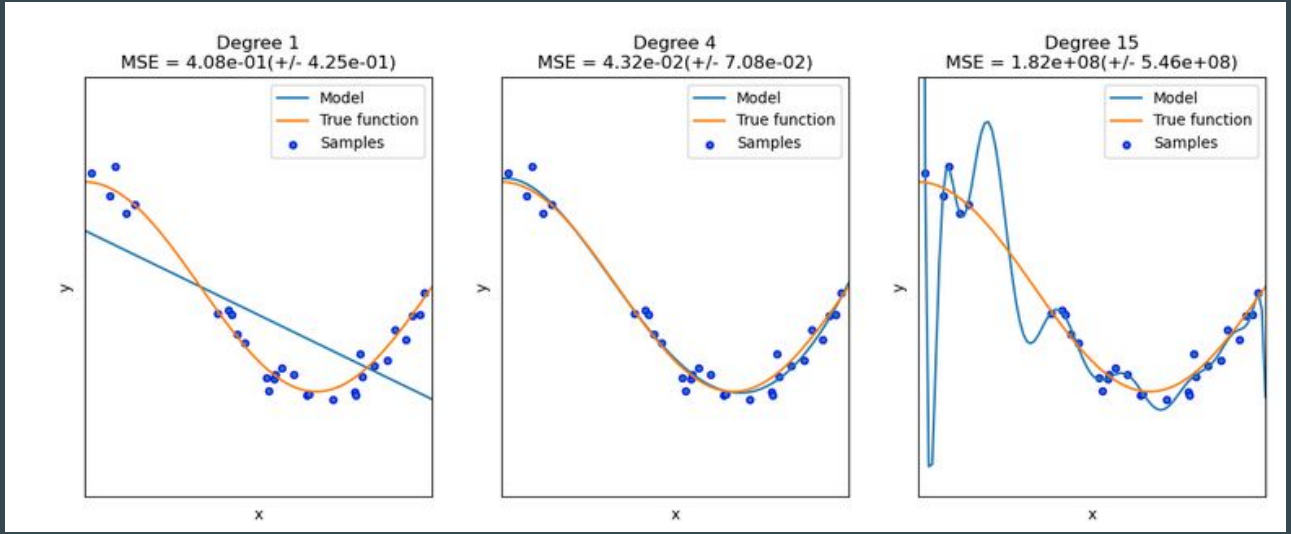
In summary, log-loss accuracy takes into account the confidence of the model's predictions, while accuracy only measures the percentage of correctly classified samples. Log-loss accuracy is a more sensitive metric for imbalanced datasets or when the cost of false positives and false negatives is different.



Bandyopadhyay and Peter 2020

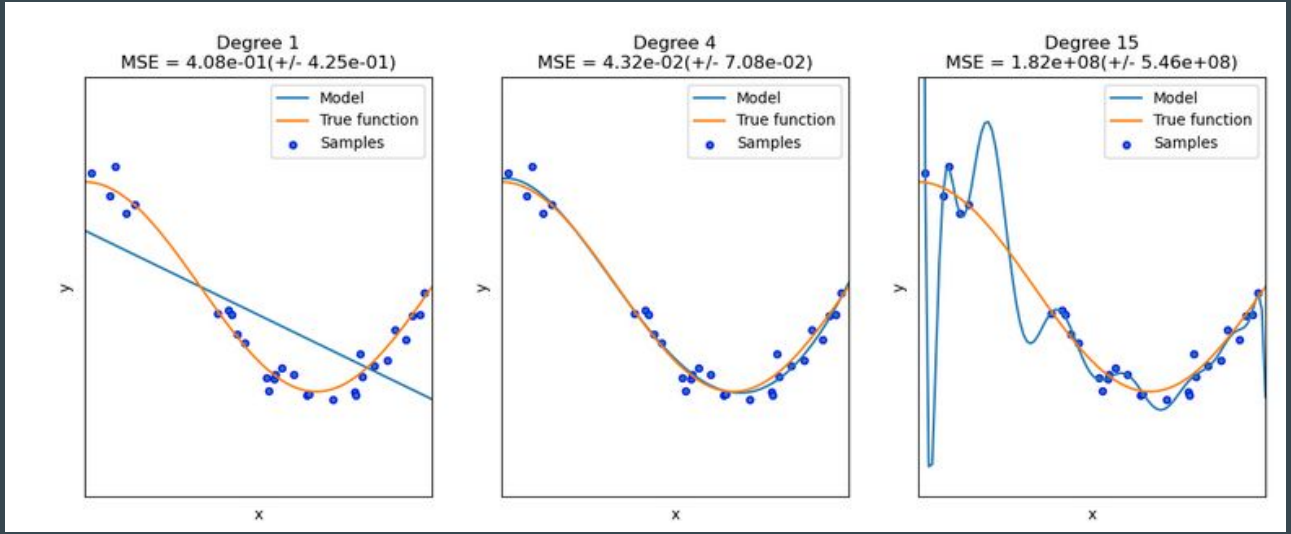
Underfitting

Underfitting occurs when a model is too simple to capture the underlying patterns in the data. It often results in low performance of the model. For example, when a linear regression model is used to capture a higher order relation between two variables, the complexity of the model is not sufficient (first panel)



Overfitting

Overfitting occurs when a model fits the training data too well and captures noise and random fluctuations in the data rather than the underlying patterns. For example, when a fitting curve represents the noise in the data, rather than the relationship between two variables (third panel)



Sklearn

Overfitting

Overfitting can be detected by the following methods:

- Plotting both training and validation curves:

  The model can be trained on a subset of the data (training set), and tested on the rest of the data (validation set). Plotting the model's training and validation accuracy as a function of the number of training epochs can help identify overfitting. If the training accuracy improves while the validation accuracy starts to plateau or decline, it may be an indication of overfitting.

Avoid Overfitting

To avoid overfitting one can use:

Cross-validation: splitting the data into multiple subsets (training and validation sets) and training the model on training sets, but testing on the validation sets. This can help detect overfitting by providing an estimate of how well the model will generalize to new data.

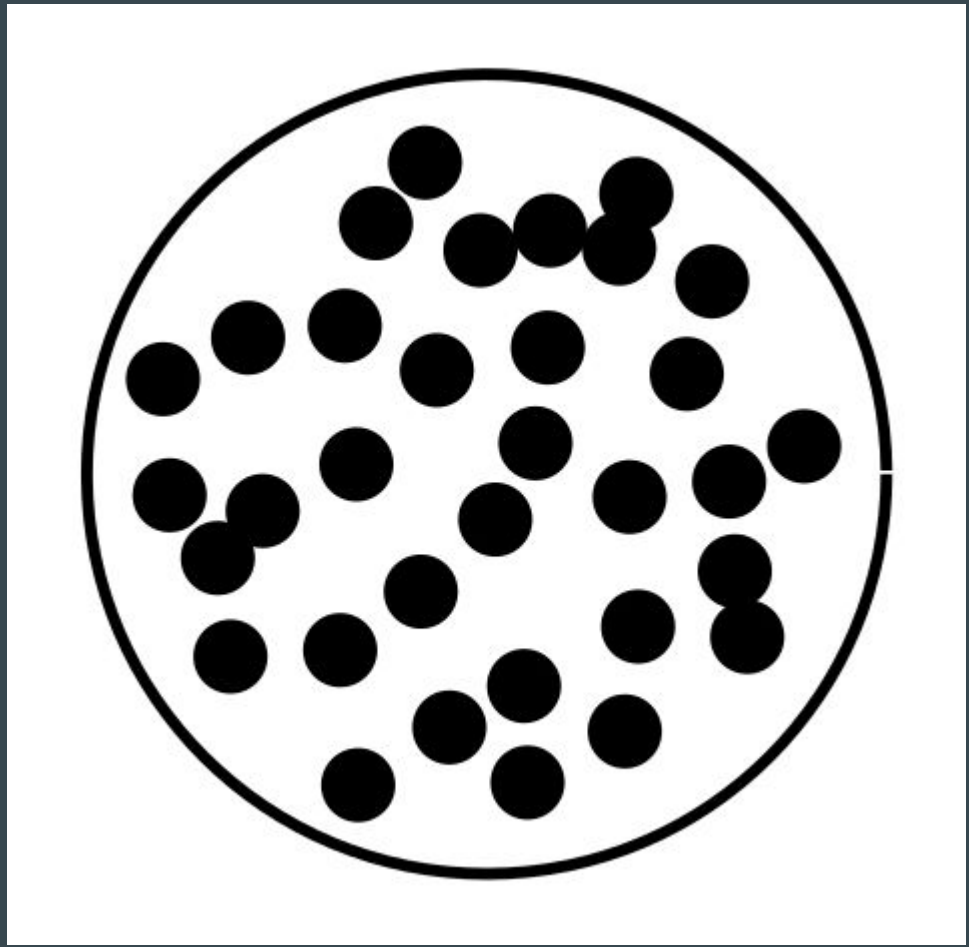
Reduce complexity: Overfitting can occur when the model is too complex for the sample size. Simplifying the model by reducing the number of features, using a smaller network architecture, or reducing the depth of the decision tree can help prevent overfitting.

Training / Test / Validation sets

Training - Test sets

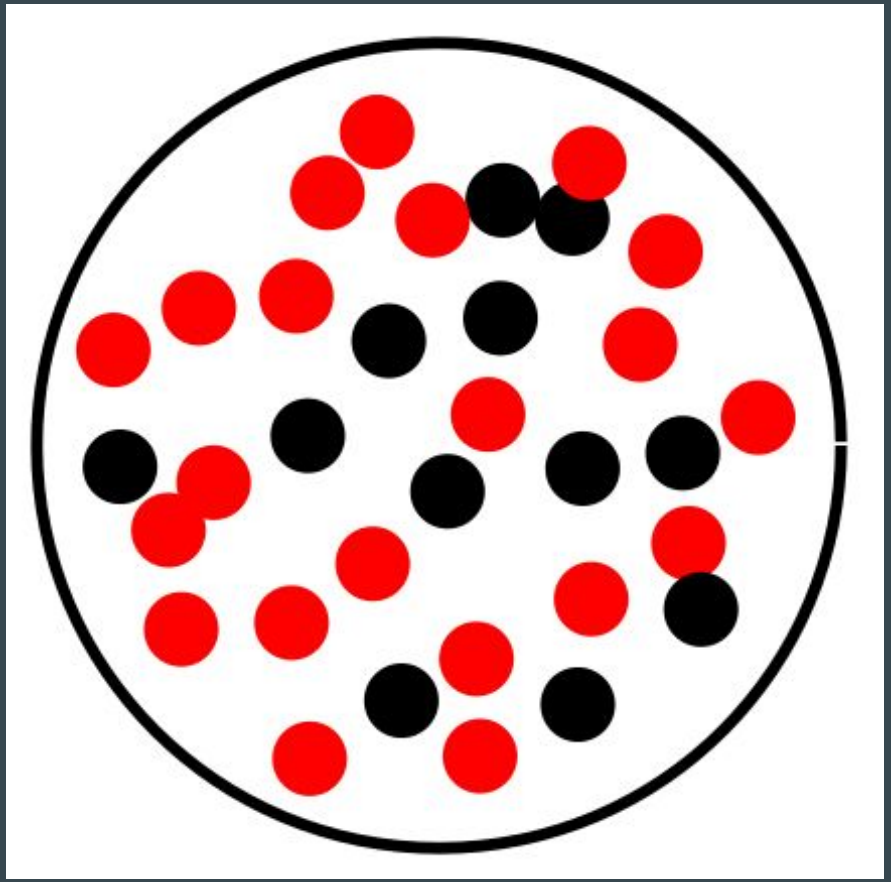
In classification, a test set and a validation set are used to evaluate the performance of a classification model.

The training set is the data used to train the model. Once the model is trained, it is important to evaluate how well it generalizes to new, unseen data.



Training - Test sets

The test set is an independent set of data that is used to evaluate the performance of the final selected model. It provides an unbiased estimate of how well the model will perform on new, unseen data.

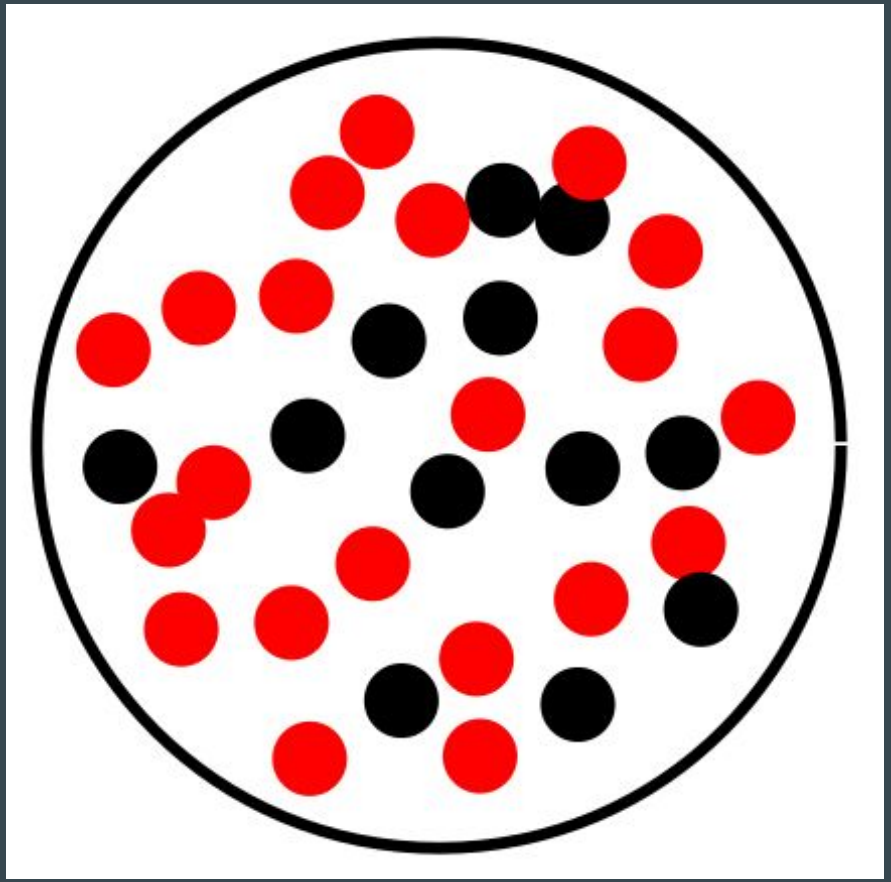


Training - Test sets

For classification methods that do not include hyperparameters, the training and validation sets are sufficient for a classification task:

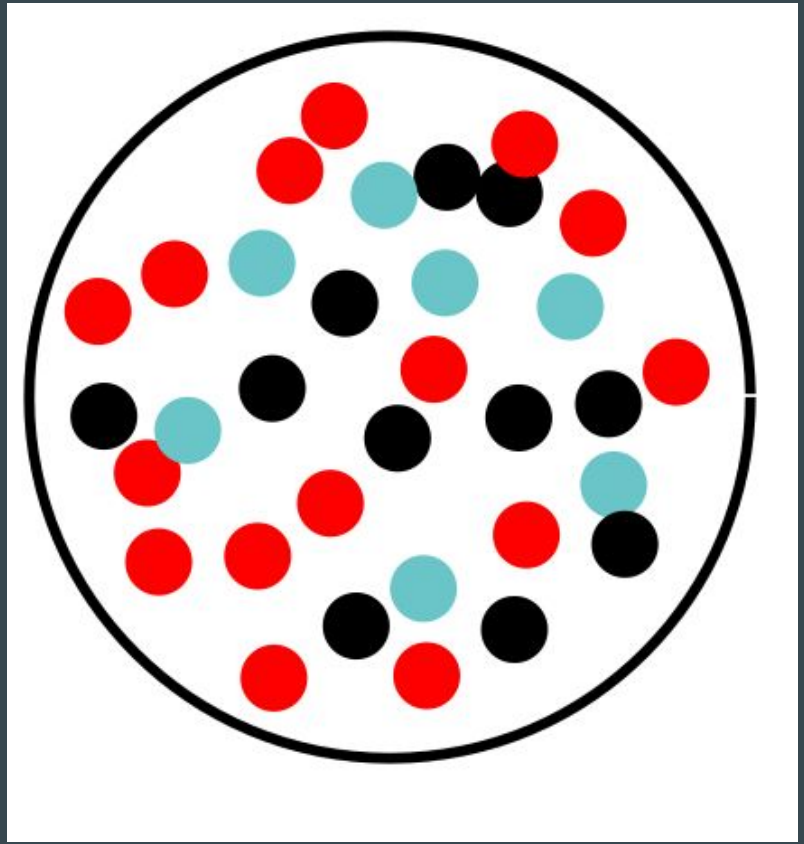
The training set is used to train the machine learning model

The test set is used to test the performance of the model in unseen data. Accuracy is evaluated on the test set.

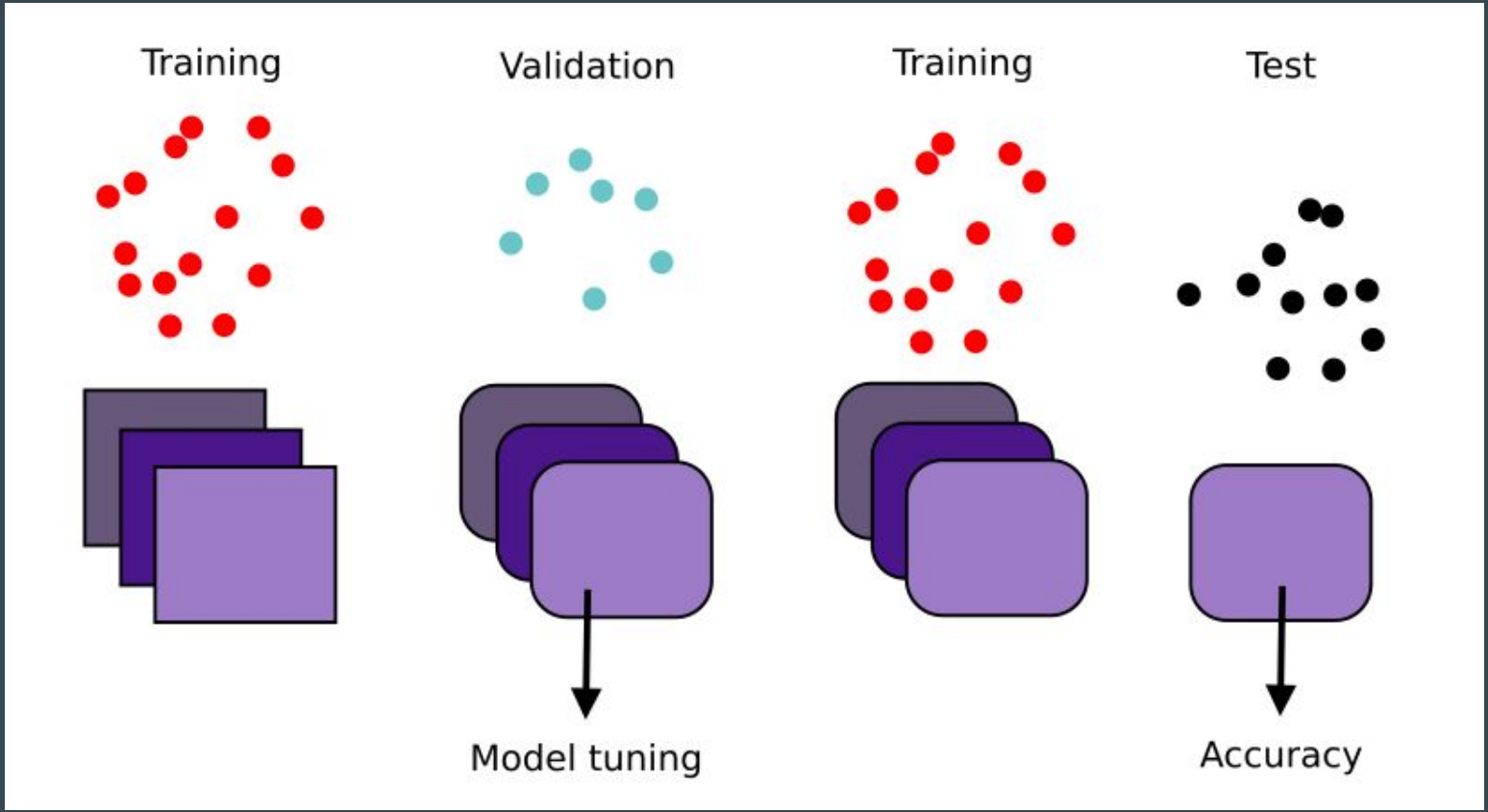


Training - Test sets

However, when the machine learning model has hyperparameters that can be manually selected, i.e. the learning rate of a learning algorithm, the number of layers and their architecture in a neural network etc. a validation set is also required. The validation set will be used to tune the hyperparameters.



Training - Test sets



Supervised classification problems

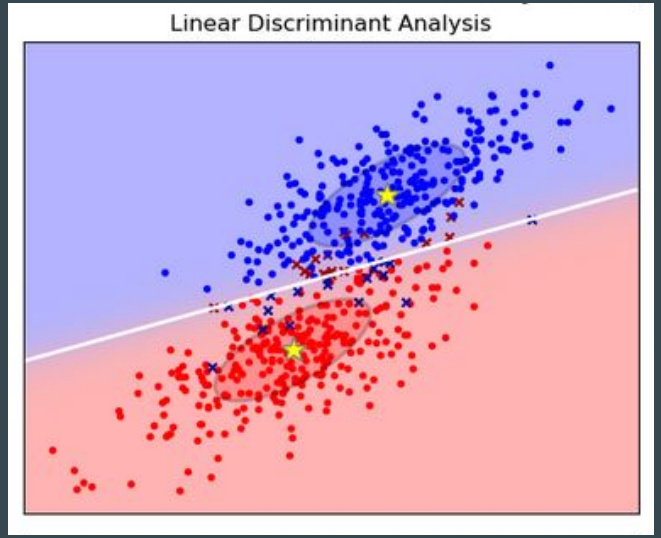
Supervised learning

- Linear discriminant analysis
- Quadratic Discriminant Analysis
- Decision trees
- Random forest
- Support vector machine
- Perceptron
- Multi-layer perceptron
- Neural networks

Supervised learning: Linear discriminant analysis

Linear Discriminant Analysis (LDA) is a supervised linear dimensionality reduction method that maximizes the separation between different classes in the data. It is commonly used in pattern recognition, face recognition, and text classification.

In LDA, the goal is to find a linear discriminant function that maximizes the separation between the classes while minimizing the variance within each class. The method involves projecting the original data onto a lower-dimensional space while preserving the class separability.



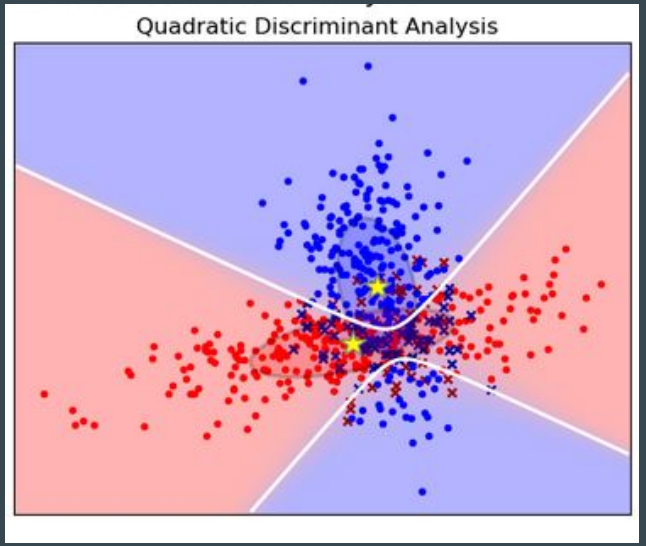
Sklearn

Supervised learning: Quadratic discriminant analysis

Quadratic Discriminant Analysis (QDA) is a classification method used to classify data into classes based on a set of features. QDA is an extension of Linear Discriminant Analysis (LDA) that allows for non-linear relationships between the features.

In QDA, each class is modeled by a separate multivariate normal distribution with its own mean vector and covariance matrix. The probability of an observation belonging to a particular class is then calculated using Bayes' theorem.

Sklearn

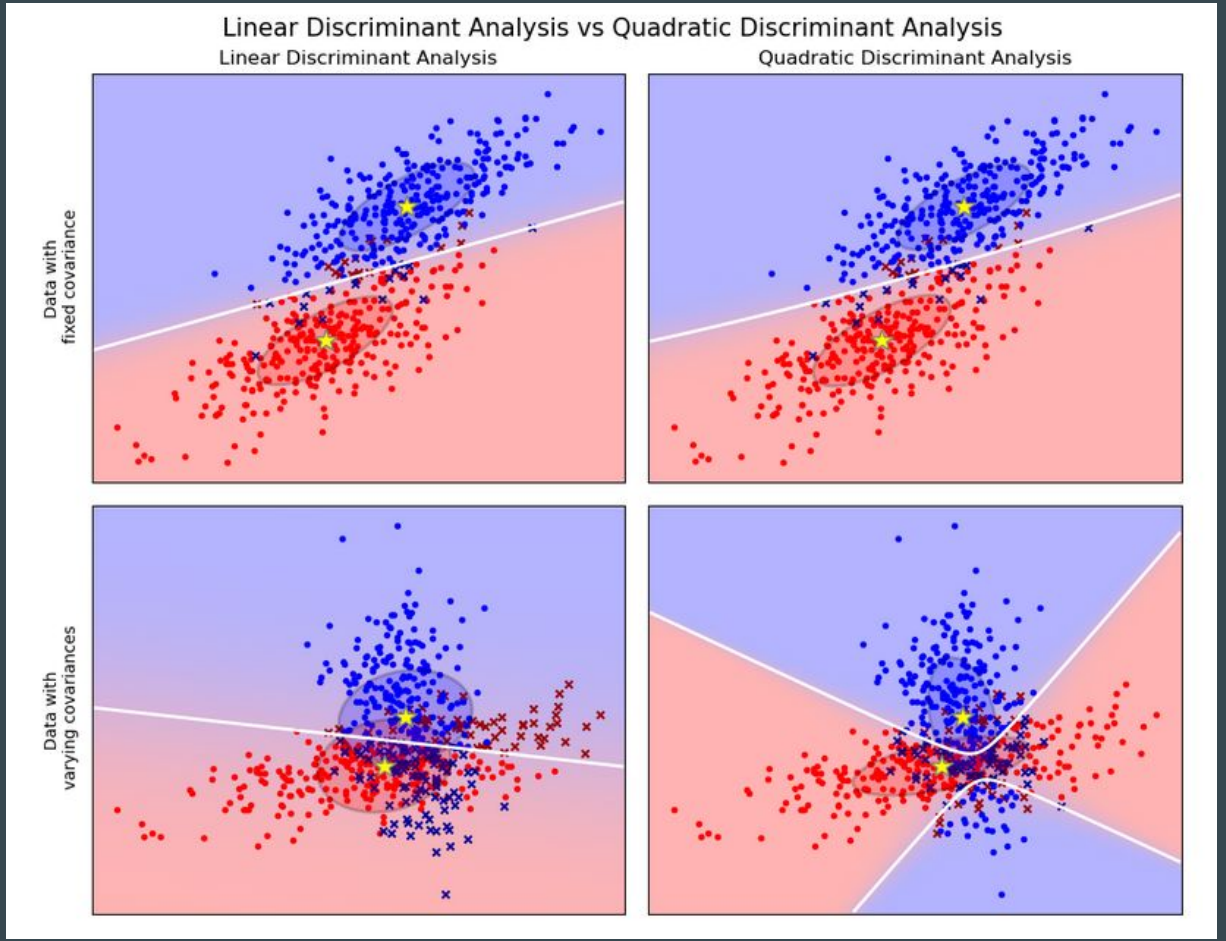


LDA vs QDA

When the covariance is fixed in the data LDA and QDA have similar results.

For varying covariance, QDA can capture non linearities in the classification space

Sklearn



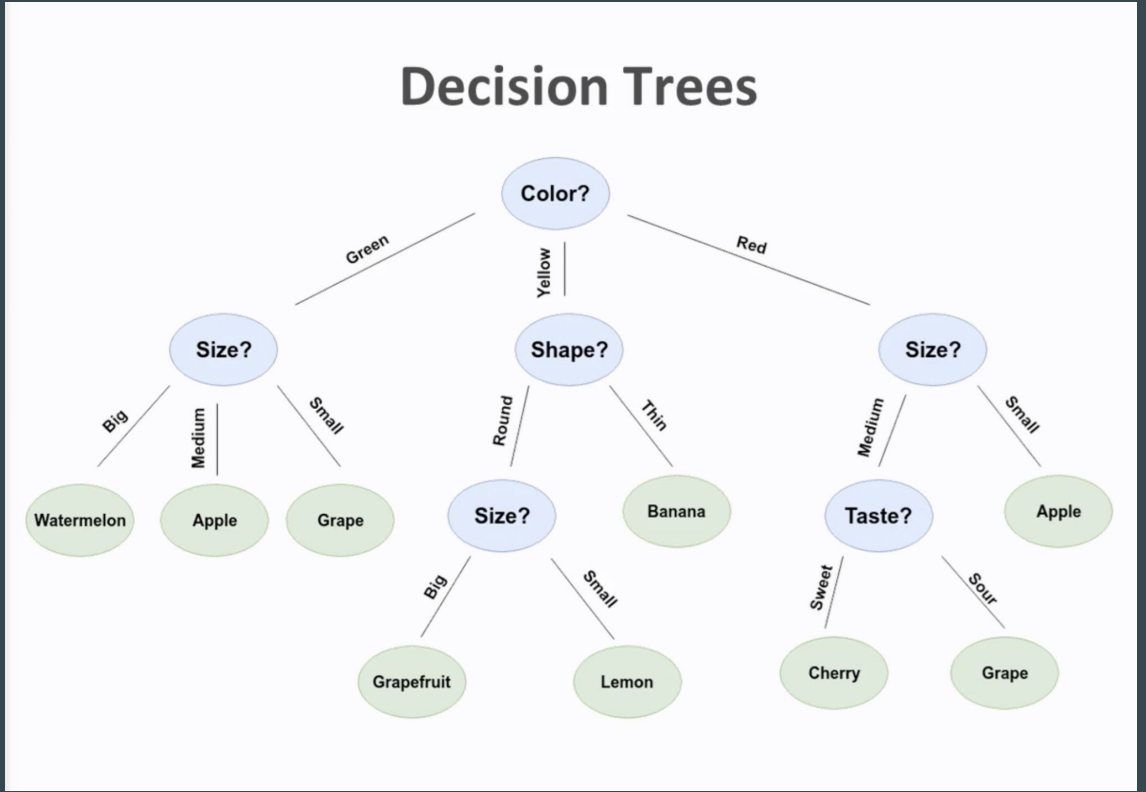
Supervised learning: Decision trees

Decision Trees are a non-parametric supervised learning method used for classification. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation. At each level a decision is made and a binary distribution of data into classes is assigned.

Supervised learning: Decision trees

The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. For example, a decision tree on fruits could look like:

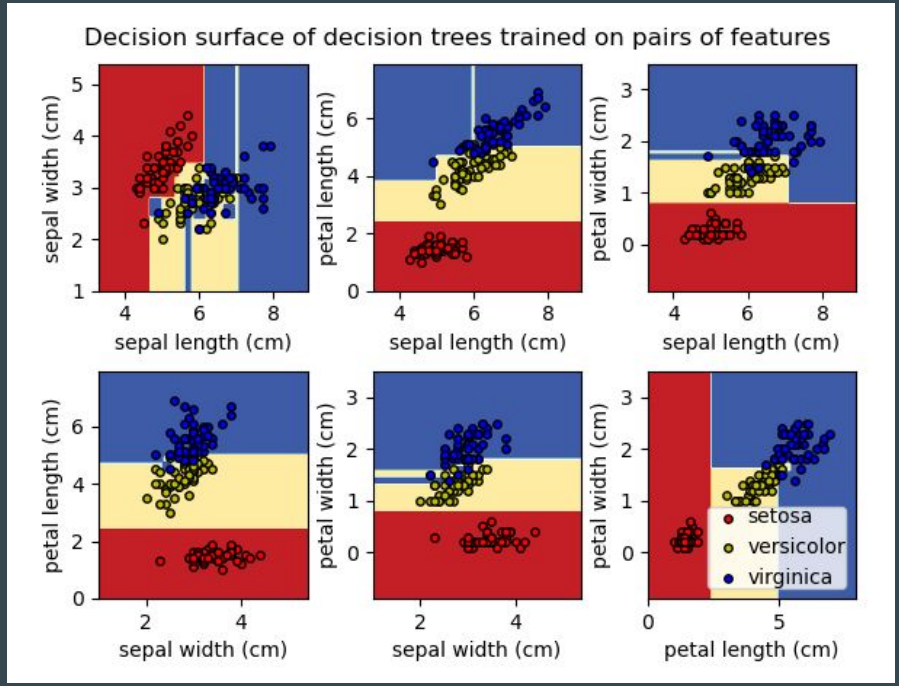
Vamshi Krishna Gunji, 2019



Supervised learning: Decision trees

Each node in DT corresponds to a decision that divides the feature space further, improving the classification results.

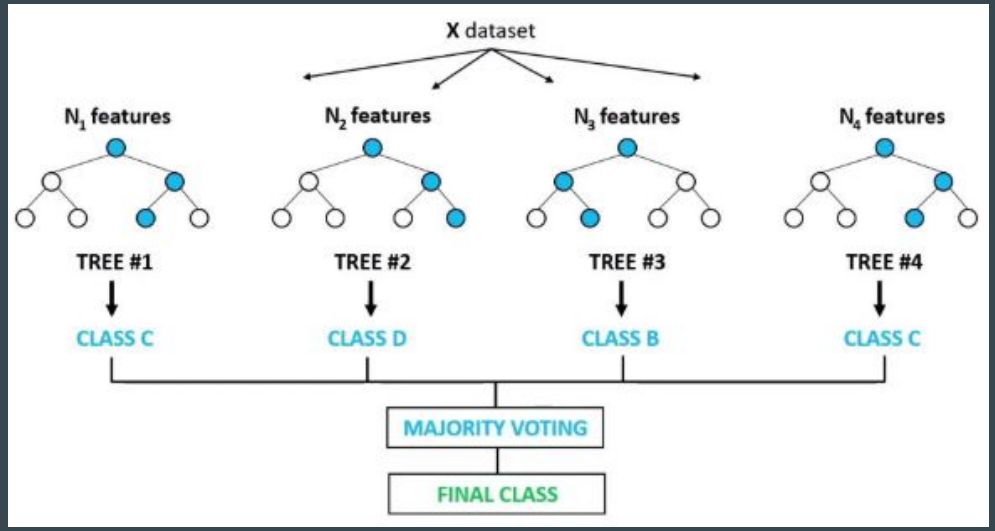
Decision trees are easy to interpret and can handle both numerical and categorical data. However, decision trees can be prone to overfitting and may not generalize well to new data if the tree is too complex. To address this issue, ensemble methods such as Random Forest and Gradient Boosted Trees can be used.



Sklearn

Supervised learning: Random forest

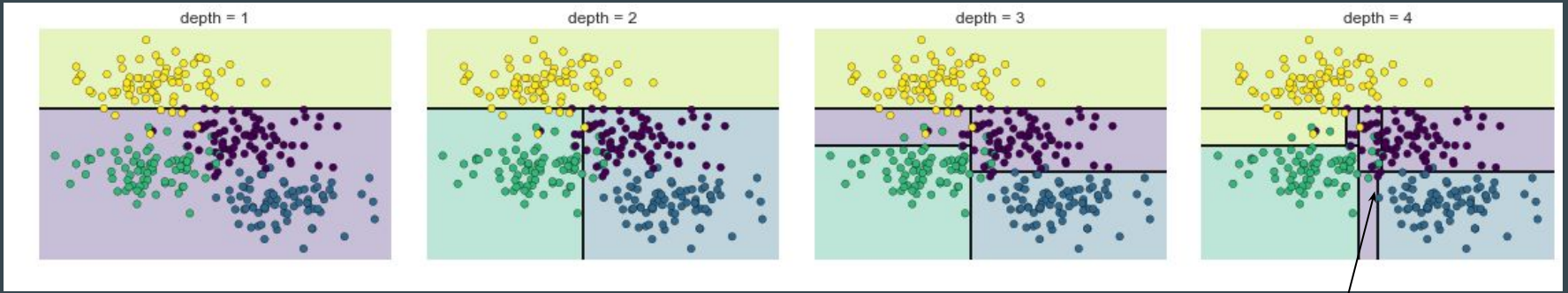
The basic idea of the random forest algorithm is to create multiple decision trees, each of which learns from a random subset of the training data and a random subset of the input features.



Davis David (@Davis_McDavid)

Supervised learning: Random forest

The stochastic element that is introduced helps to reduce overfitting and make the model more robust to noisy data. At test dataset, the random forest combines the individual predictions of all the individual trees to make a final prediction.



Sklearn

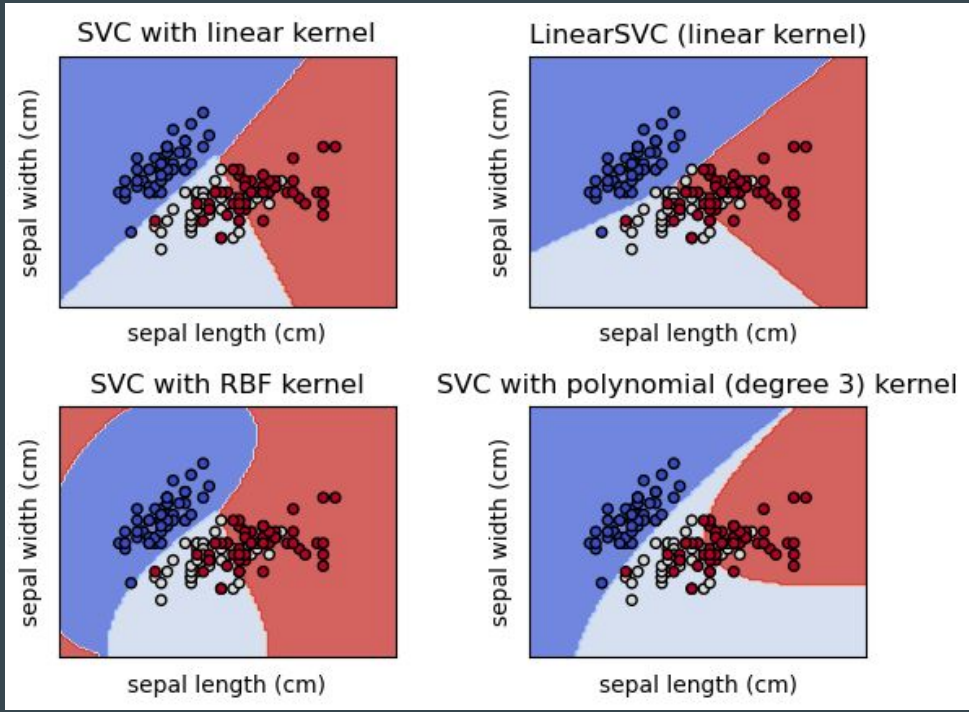
Overfitting in decision trees can be avoided

Supervised learning: Support vector machine (SVM)

Support Vector Machine, can be used for both binary and multiclass classification problems. The key idea behind SVM is to find the optimal hyperplane that separates the data points of different classes in the feature space with maximum margin. The data points that are proximal to the hyperplane are called support vectors, and they are important in determining the position and orientation of the hyperplane. SVM has different kernel implementations that allow it to handle nonlinear data and improve its performance in complex classification problems. It can handle high-dimensional data and it is robust to overfitting. One problem with SVM is it might be tricky to fine tune the parameters to find the optimal results.

Supervised learning: Support vector machine (SVM)

SVM is a powerful algorithm that can work well with both linearly separable and non-linearly separable data. For non-linearly separable data, SVM uses a technique called the kernel trick, which maps the input data to a higher-dimensional space where the data can be linearly separable. This allows SVM to find a non-linear decision boundary that separates the data into different classes.

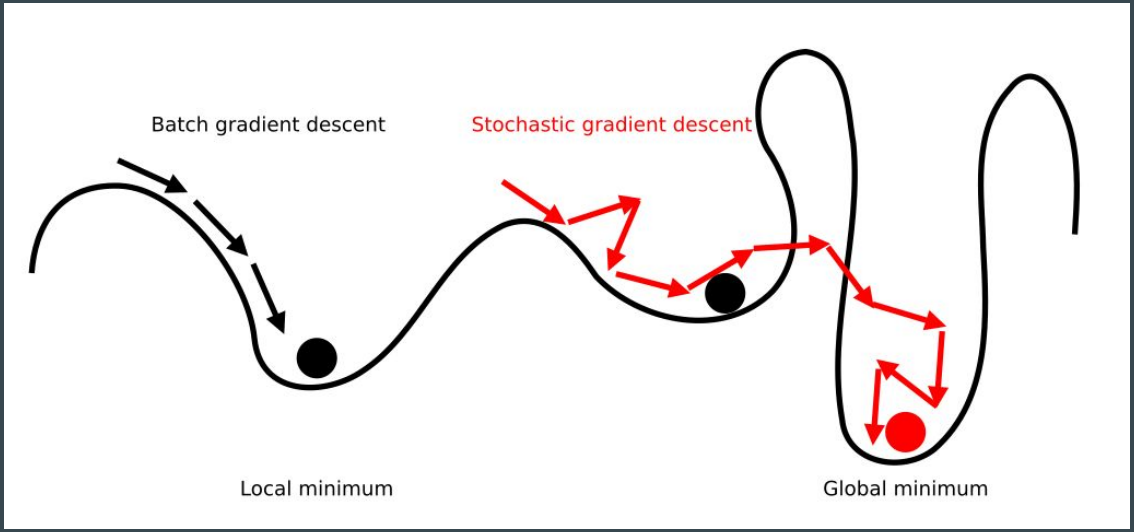


Sklearn

Supervised learning: Stochastic Gradient Descent

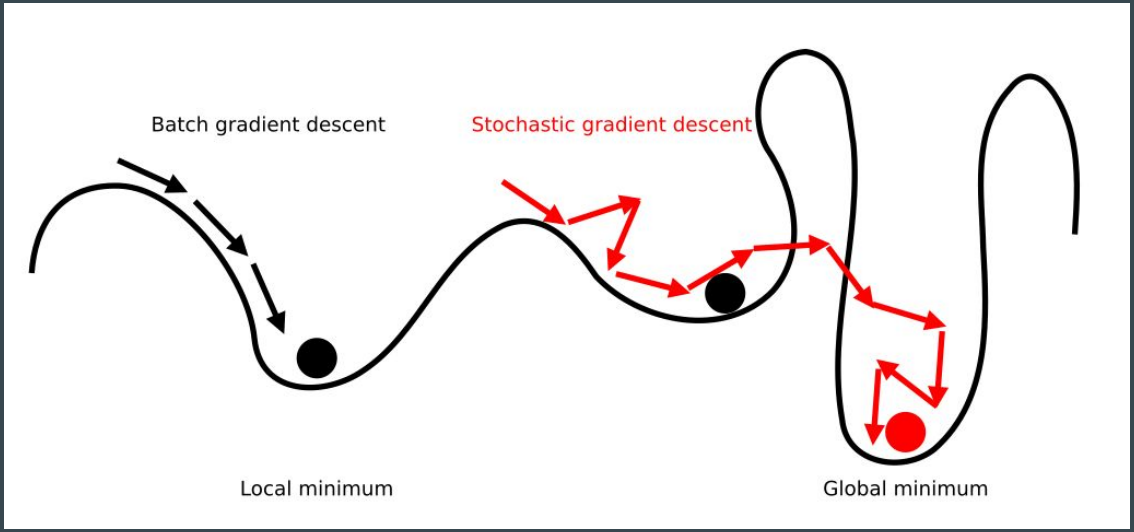
Stochastic gradient descent (SGD) is commonly used in classification tasks to minimize the loss function of a model, which measures how well the model predicts the correct class labels. The model parameters are updated after each example in the training set is presented to the model in contrast to batch gradient descent, where the parameters are updated after processing the entire training set. The advantage of SGD is that it is computationally efficient, as the updates can be performed on each example in parallel. It is also less likely to get stuck in local minima, as it has more chances to explore the parameter space.

Supervised learning: Stochastic Gradient Descent



SGD minimizes the loss function, i.e. the error between actual and predicted labels.

Supervised learning: Stochastic Gradient Descent



It is less likely to get stuck at a local minimum due to the stochastic element of the algorithm that allows it to explore a parameter space.

Supervised learning: Stochastic Gradient Descent

Stochastic Gradient Descent is sensitive to feature scaling, so it is useful to scale your data. For example standardize all the data to have mean 0 and variance 1.

$$\bar{f} = \frac{f - < mean >}{< std >}$$

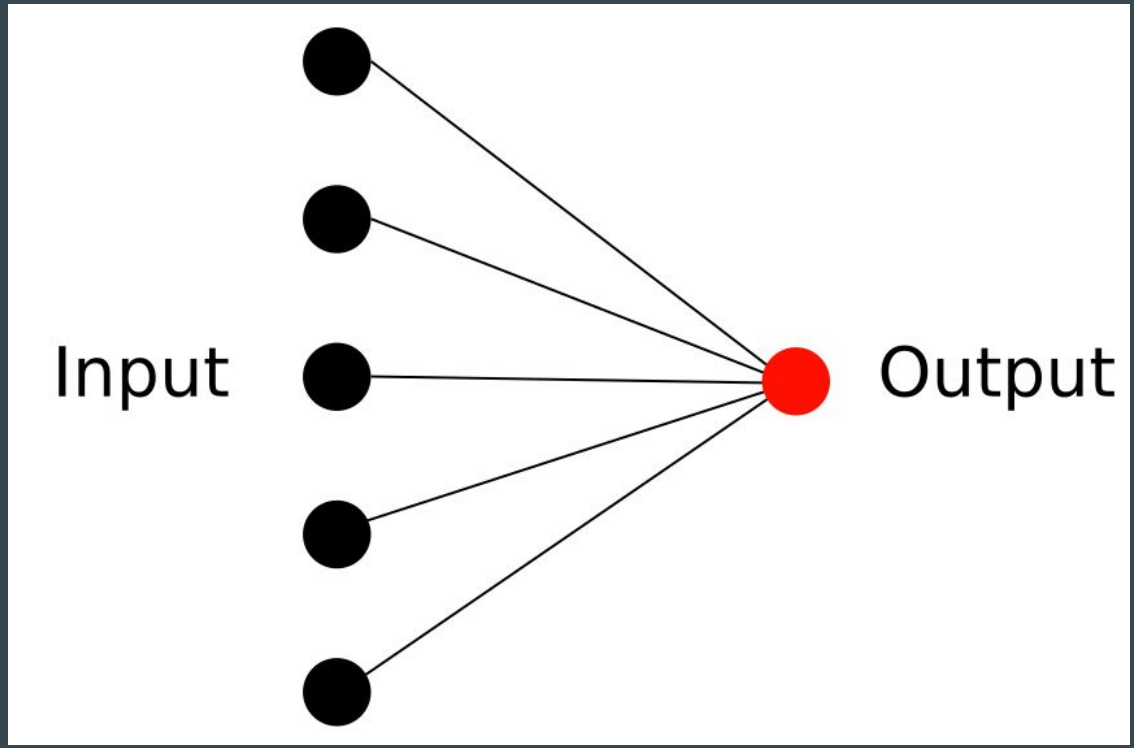
Note that the same scaling must be applied to the test vector to obtain meaningful results. In addition, the global values of the features need to be normalized, a common mistake is normalization per class which leads to overfitting.

Supervised learning: Perceptron

A perceptron is a simple artificial neural network that is used for binary classification tasks. The perceptron consists of multiple input nodes, which receive the input data, and an output node, which gives the binary classification result. Each input node is connected to the output node through a weight. The weights are adjusted during training, based on the input data and the desired output, to improve the accuracy of the classification.

Supervised learning: Perceptron

Multiple input nodes are connected to the same output node. In the modern era of neural networks this looks like a naive design, but for the 40s that was a novel and very powerful design.



Supervised learning: Perceptron

At each iteration, during the training process, the perceptron is presented with input data and the corresponding output label. If the perceptron produces the correct output, the weights are left unchanged. If the output is incorrect, the weights are adjusted to reduce the error. This process is repeated until the perceptron reaches a satisfactory level of accuracy.

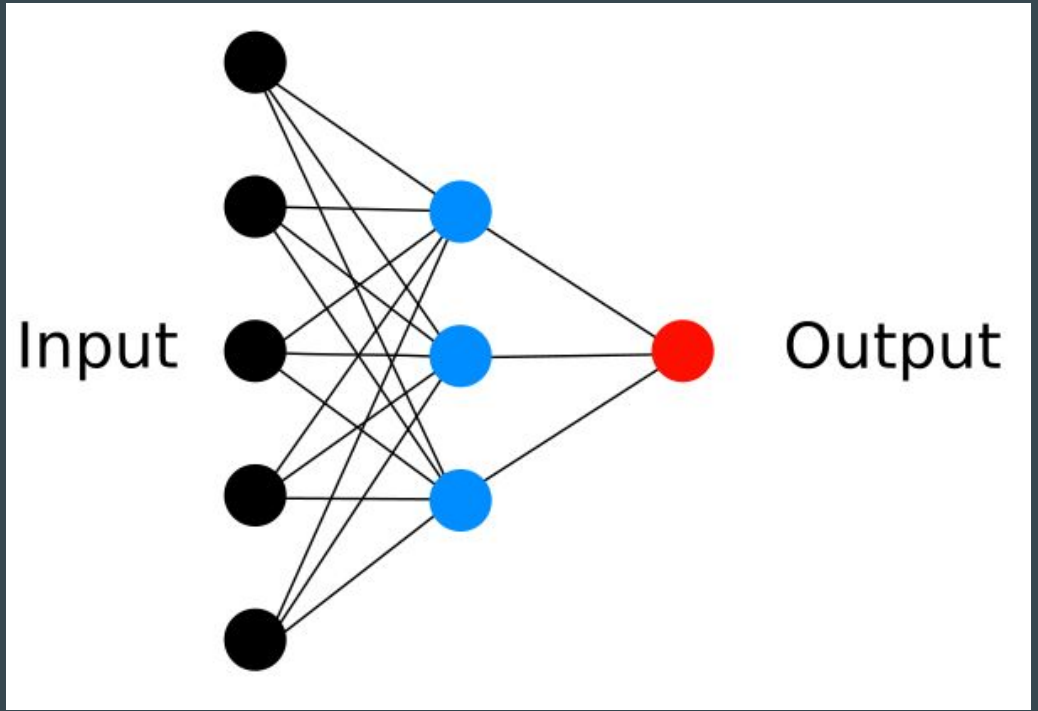
The perceptron algorithm is simple and efficient, but it can only classify data that can be separated into two groups by a straight line or a hyperplane. For more complex problems, to improve performance a multilayer perceptron is often used.

Supervised learning: Multi-layer perceptron

A multilayer perceptron (MLP) is an artificial neural network that consists of multiple layers of interconnected neurons. It is a feedforward neural network, i.e. the information flows in one direction from the input layer to the output layer.

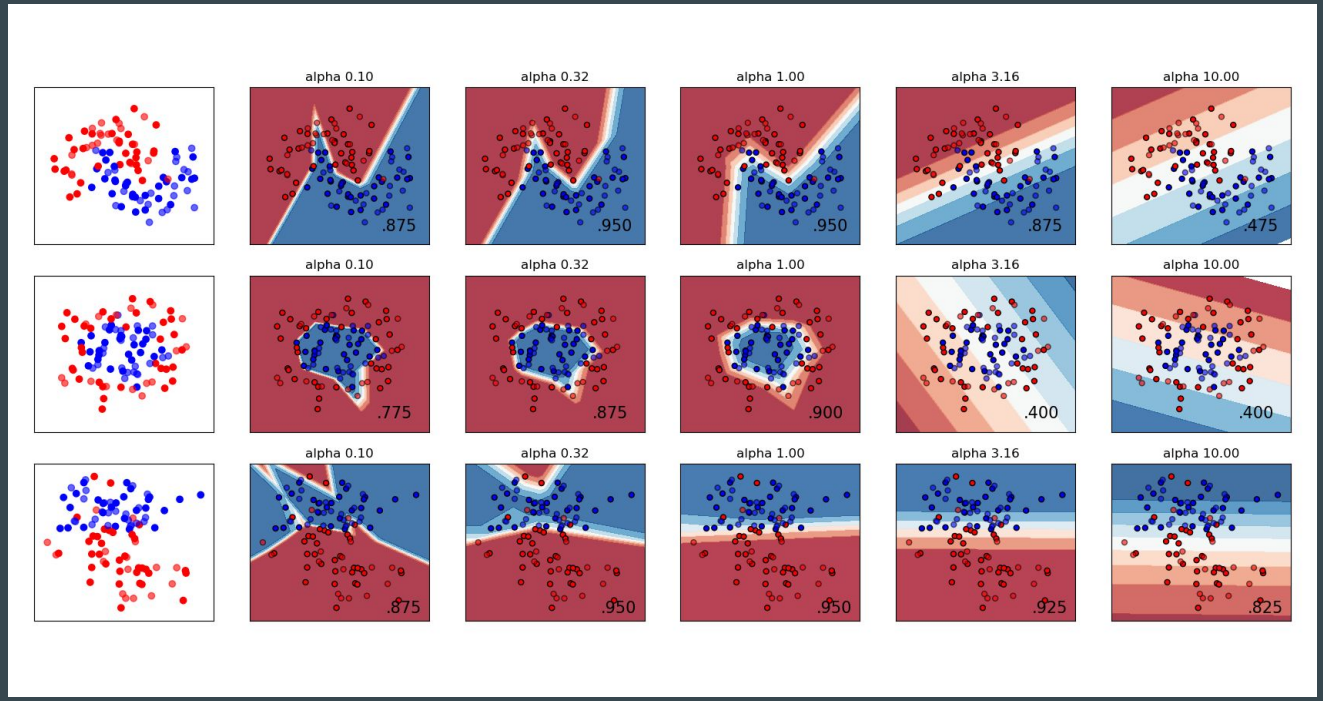
Supervised learning: Multi-layer perceptron

Typically, the MLP consists of an input layer, one or more fully connected layers, and an output layer. The weights between neurons of adjacent layers are modified during the training process to improve the accuracy of the network's predictions.



Supervised learning: Multi-layer perceptron

Example of classification results for multiple example inputs and for varied parameters of the MLP. Optimal parameters depend on the input dataset



Sklearn

Supervised learning: Neural networks

Neural networks, refer to a broad range of models inspired by the structure and function of biological neural networks. They can take many different forms and architectures, including MLPs, convolutional neural networks (CNNs), recurrent neural networks (RNNs).

One key difference between MLPs and other types of neural networks is their architecture. MLPs are fully connected feedforward networks, i.e. each neuron in a given layer is connected to all neurons in the previous and next layers. CNNs, for example, have convolutional layers that are specialized for processing spatial data, such as images. RNNs have recurrent connections that allow them to process sequential data, such as time series data or text.

New insights into the classification and nomenclature of cortical GABAergic interneurons

DeFelipe et al. 2013

Phenotypic variation of transcriptomic cell types in mouse motor cortex

- A feature-based classification and agreed-upon nomenclature of GABAergic interneurons of the cerebral cortex is much needed but currently lacking
- We designed a web-based interactive system that allowed 42 neuroscience experts to classify a representative sample of 320 cortical neurons and a selected set of simple morphology features based on reconstructions of their axonal arbors
- The consensus on and usefulness of these features and neuron names were investigated using agreement analysis, clustering algorithms, Bayesian networks and supervised classification on the resulting data.

Phenotypic variation of transcriptomic cell types in mouse motor cortex

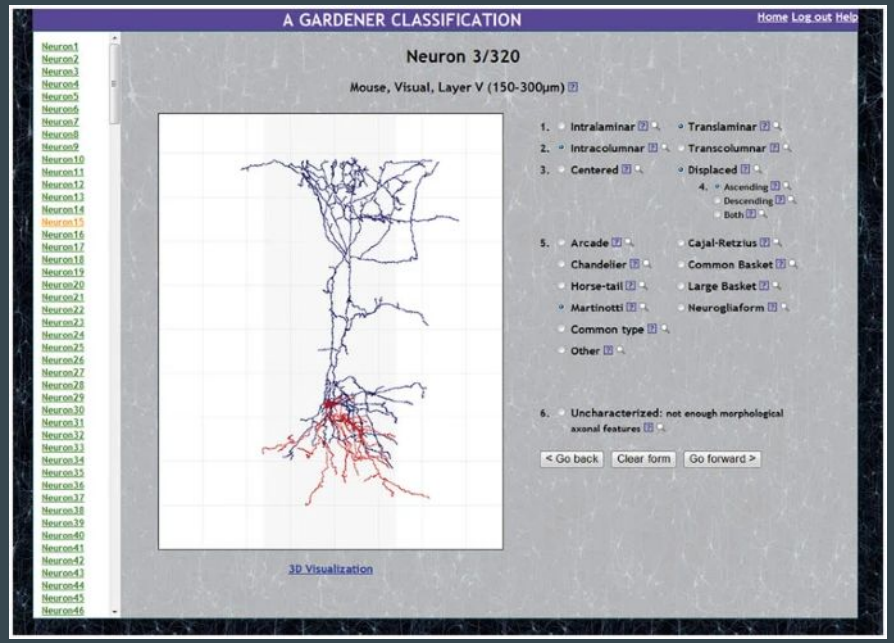
- The results quantitatively confirm the impression that different investigators use their own, mutually inconsistent classification schemes based on morphological criteria.
- However, the analyses also demonstrate that the community may be reaching consensus for a practical approach to the naming of certain anatomical terms that are useful for neuronal characterization and classification.
- State-of-the-art machine learning approaches were shown to achieve discrimination capability equivalent to or better than human performance, opening the possibility of creating an objective computer tool for automatic classification of neurons, a Neuroclassifier.

Phenotypic variation of transcriptomic cell types in mouse motor cortex

Problem: classify neurons automatically

Experts have proposed classifications by looking at the neuronal shapes in the microscope

Morphological features can be extracted from the digital reconstruction of neurons

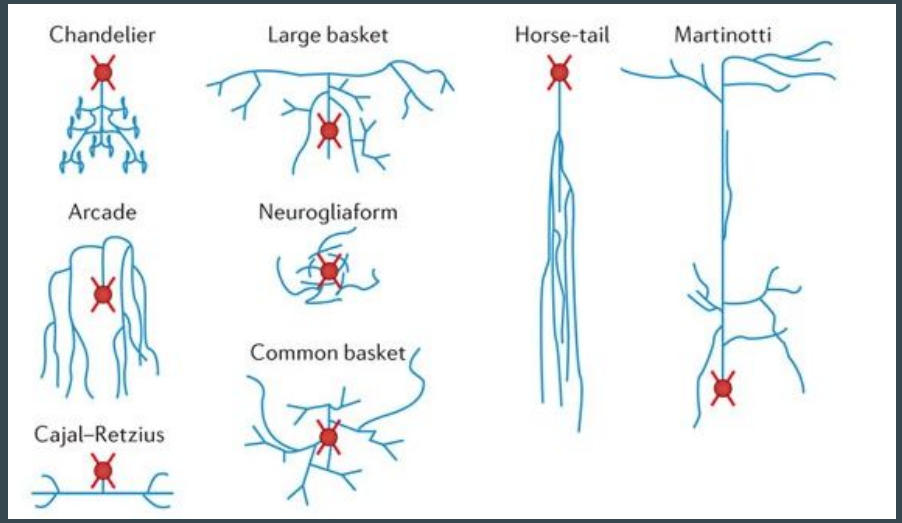


Phenotypic variation of transcriptomic cell types in mouse motor cortex

Example classification scheme:

Based on the shapes of the axons, interneurons have been assigned into ~8 categories.

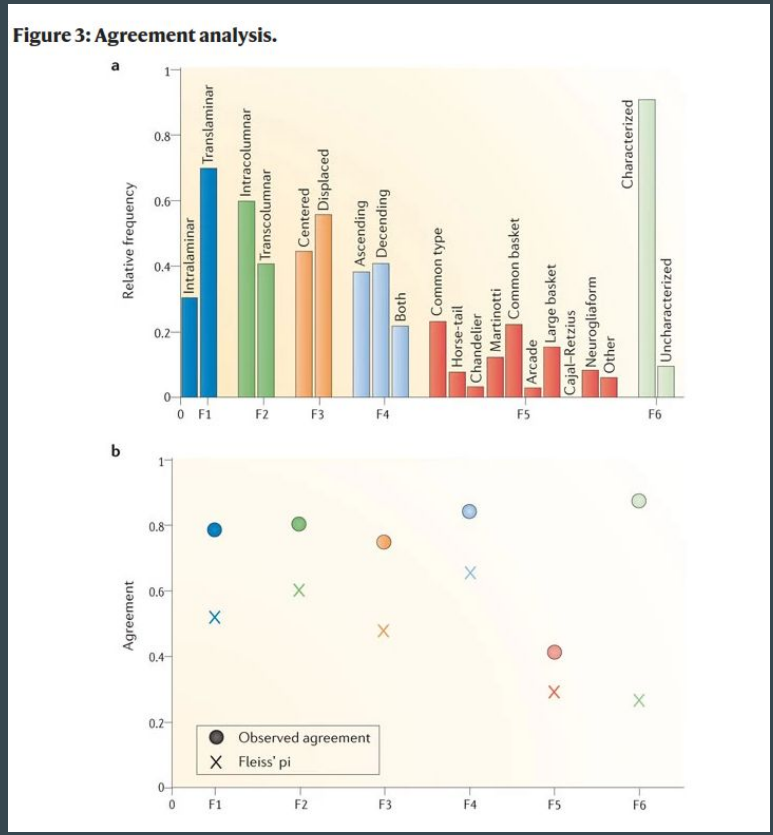
The schematic illustration shows the typical shapes of these categories



Phenotypic variation of transcriptomic cell types in mouse motor cortex

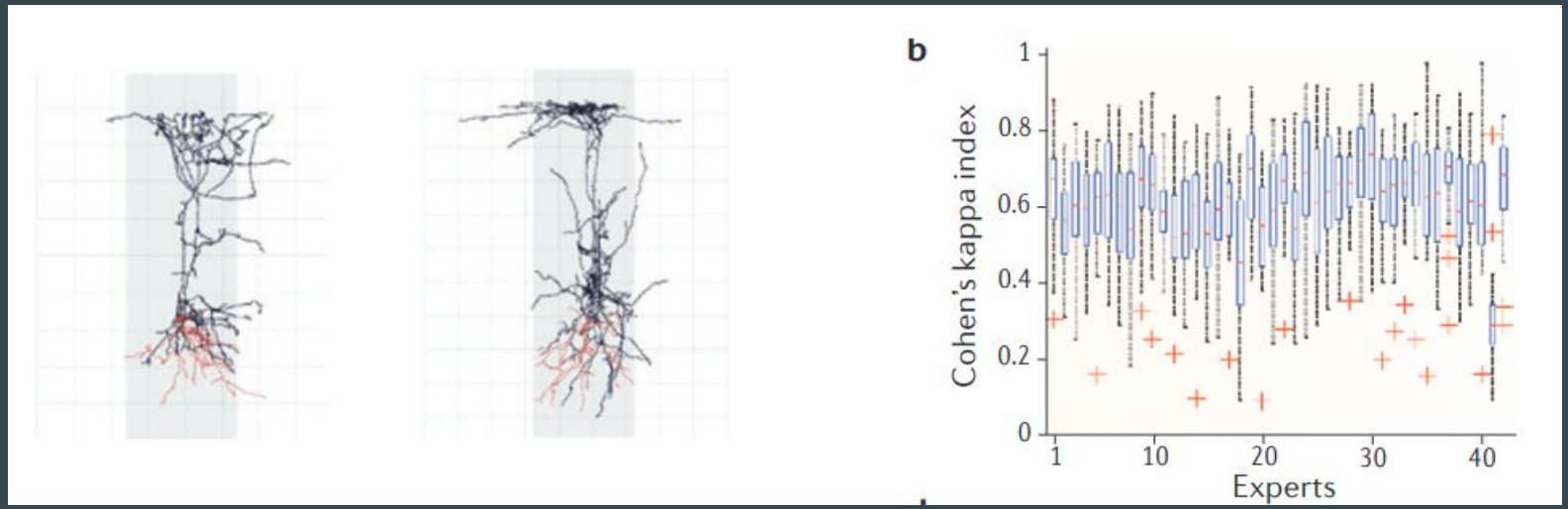
Agreement between experts

Relative frequency of each category for each feature (F1 to F6): that is, the number of times a category was selected divided by the total number of ratings for the relevant feature. (b) Overall observed agreement (circles) and chance-corrected (crosses) for each feature, indicating the degree of concordance between the experts.



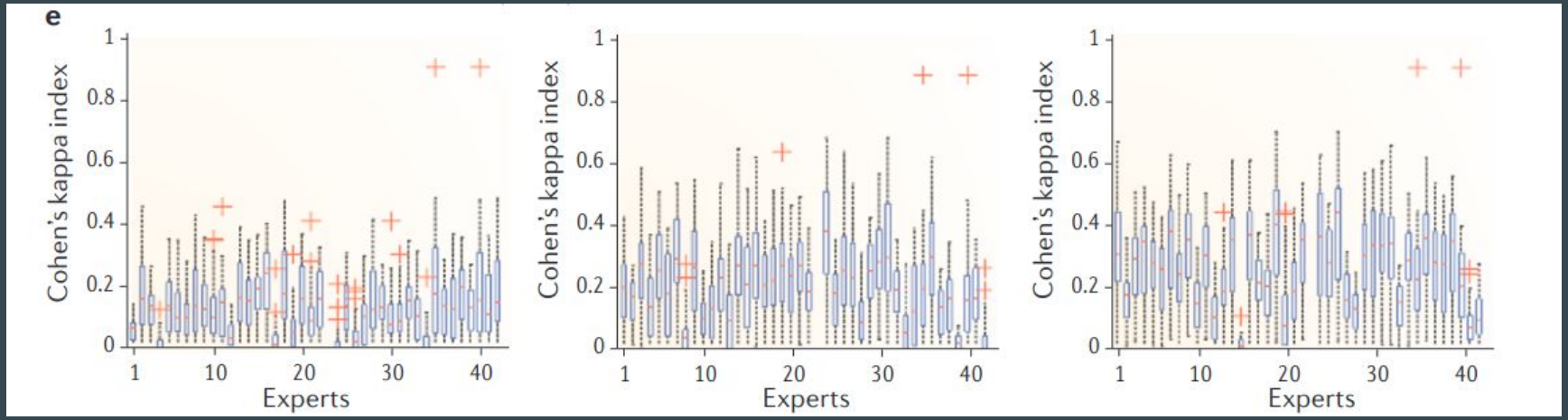
Phenotypic variation of transcriptomic cell types in mouse motor cortex

For some cell types, experts agreement is very high (for example 41/45)



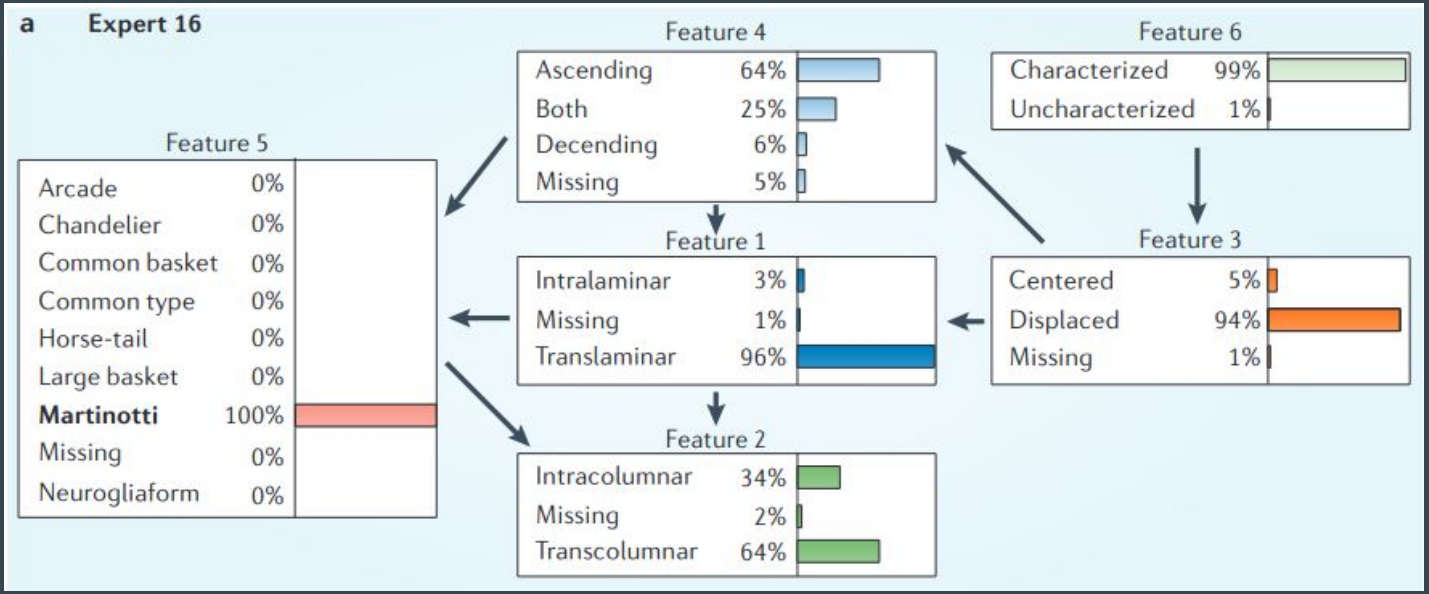
Phenotypic variation of transcriptomic cell types in mouse motor cortex

However, for other cell types, expert agreement is very low (~30%)



Phenotypic variation of transcriptomic cell types in mouse motor cortex

Bayesian networks describe experts' decision on a neuron type. Each expert used a different decision-tree like process to assess neuronal types.



Phenotypic variation of transcriptomic cell types in mouse motor cortex

- This study empirically and quantitatively demonstrates that the gardener's approach to neuron classification is untenable at this time and confirms the impression that different investigators use their own, mutually inconsistent schemes for classifying neurons based on morphological criteria. Many ambiguities are independent of the relative reconstruction quality and completeness of the tested neurons.
- A striking indication of the problem is that in several cases, experts assigned a different name to a neuron than the term they had chosen in their own original publication from which that same neuron was taken.

Phenotypic variation of transcriptomic cell types in mouse motor cortex

- In such a use-case it is important to keep in mind that any supervised classification scheme is highly biased on the expert labels that were presented to it. Therefore, an algorithm that yields very high accuracy (~100%) is probably an indication of overfitting, picking up noise, rather than real features.
- A way to miss overfitting in this instance is the use of high number of input features. For example if we separate 200 cells into 5 classes, but we use 500 features, it will always be possible for a classifier to find combinations of relevant features that increase the accuracy.

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination

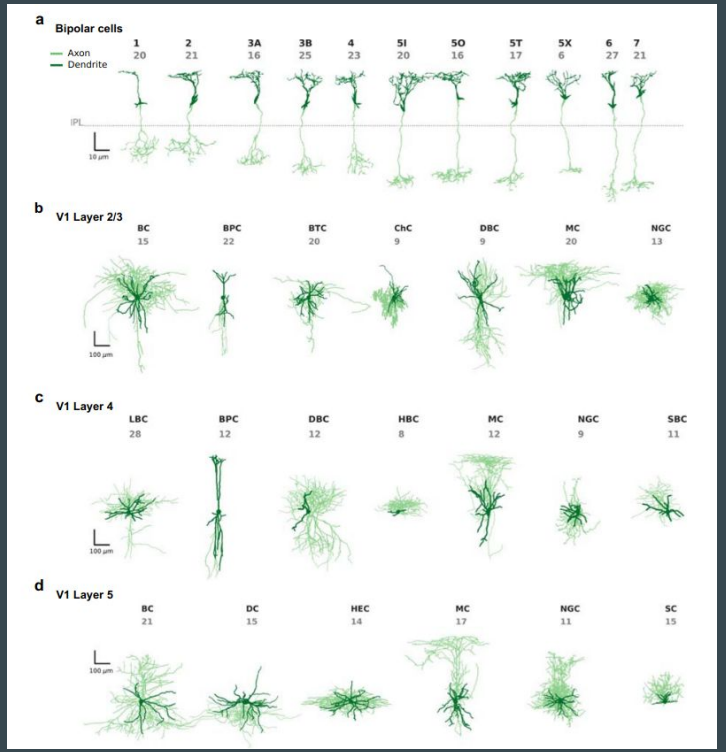
Sophie Laturnus et al. 2020

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination

Quantitative analysis of neuronal morphologies usually begins with choosing a particular feature representation in order to make individual morphologies amenable to standard statistics tools and machine learning algorithms. Many different feature representations have been suggested in the literature, ranging from density maps to intersection profiles, but they have never been compared side by side. Here we performed a systematic comparison of various representations, measuring how well they were able to capture the difference between known morphological cell types. We found that the best performing feature representations were two-dimensional density maps, two-dimensional persistence images and morphometric statistics, which continued to perform well even when neurons were only partially traced. Combining these feature representations together led to further performance increases suggesting that they captured non-redundant information.

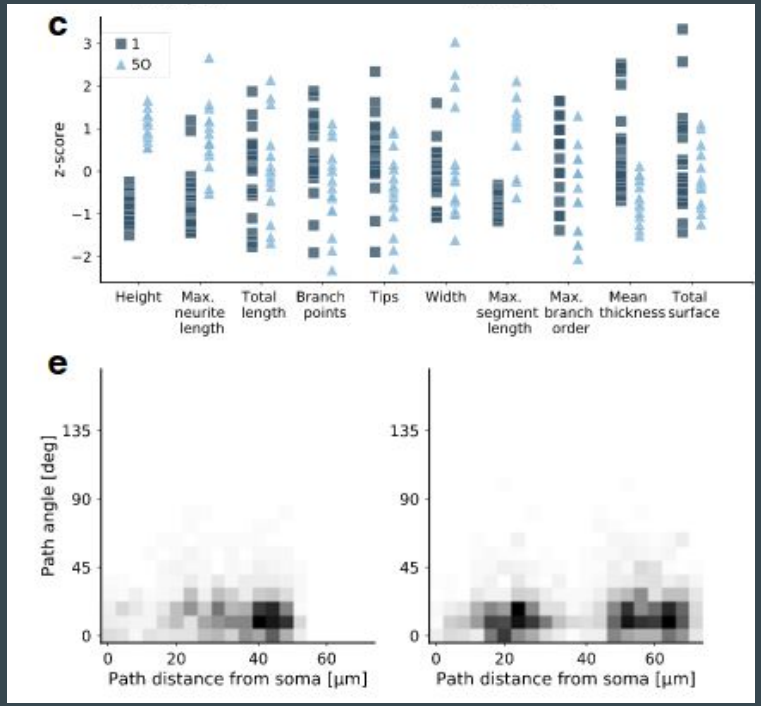
A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination

Exemplary cells of each cell type for all four data sets. Axons are shown in light green, dendrites in dark green.
(a) Mouse retinal bipolar cells
(b) L2/3 inhibitory interneurons in primary visual cortex of adult mice (Jiang et al. 2015)
(c) L4 inhibitory interneurons in primary visual cortex of adult mice (Scala et al. 2019)
(d) L5 inhibitory interneurons in primary visual cortex of adult mice (Jiang et al. 2015)

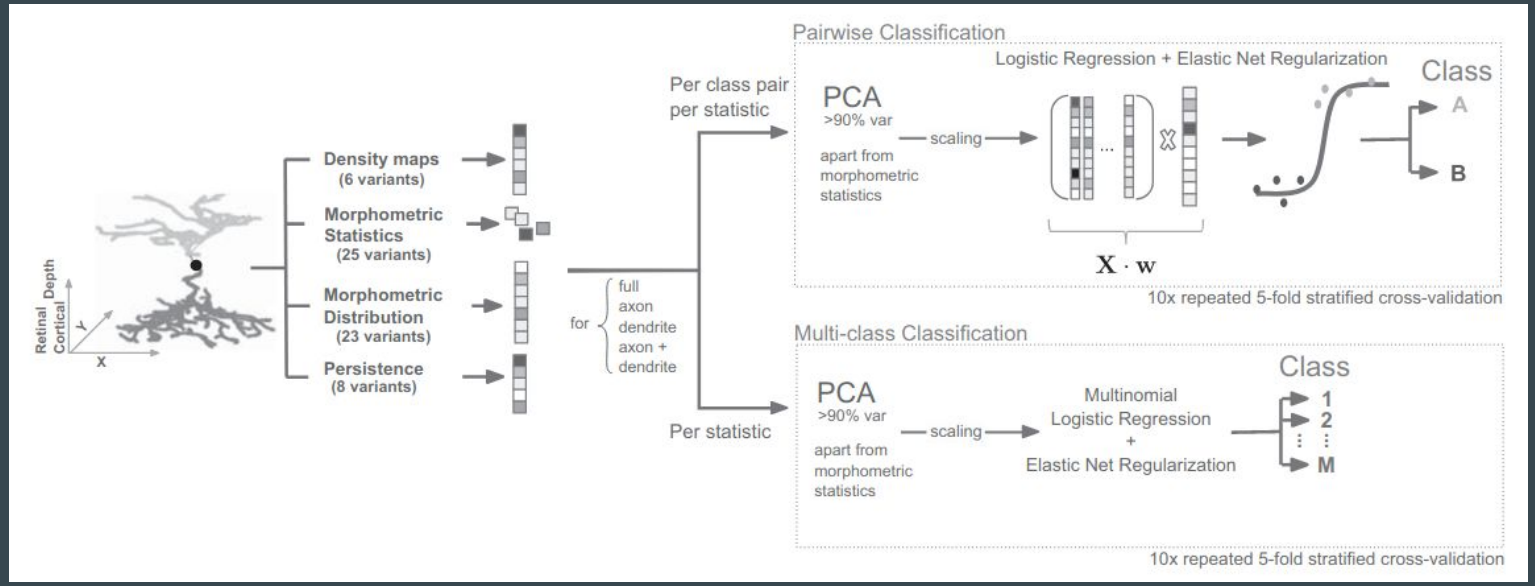


A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination

Examples of features used in the classification process. Features include:
Density maps,
Morphometrics,
Distributions of morphometrics,
Topological descriptors

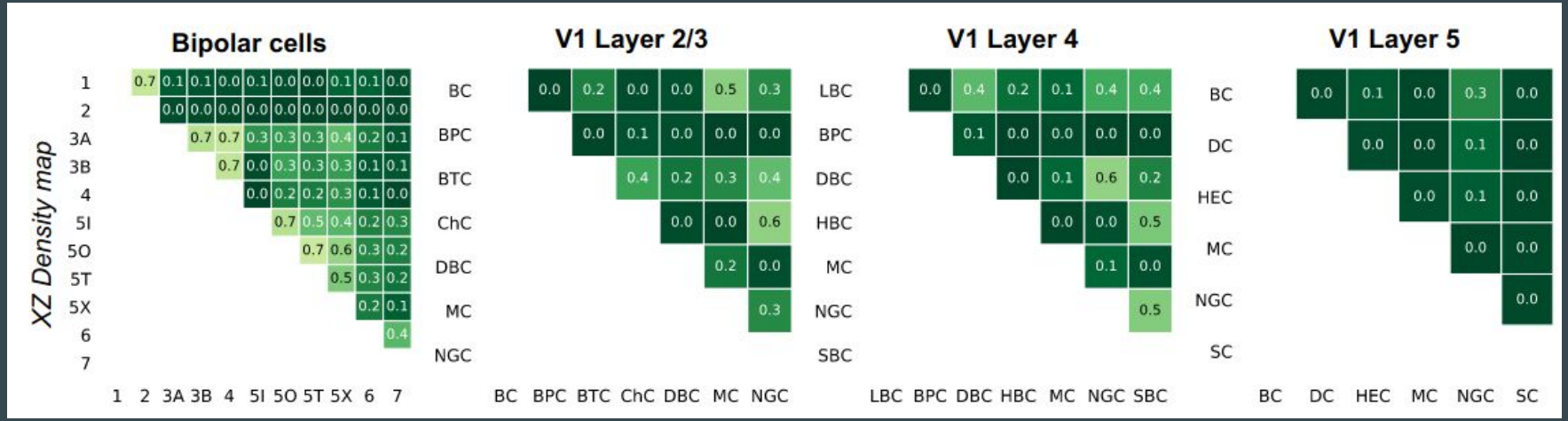


A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



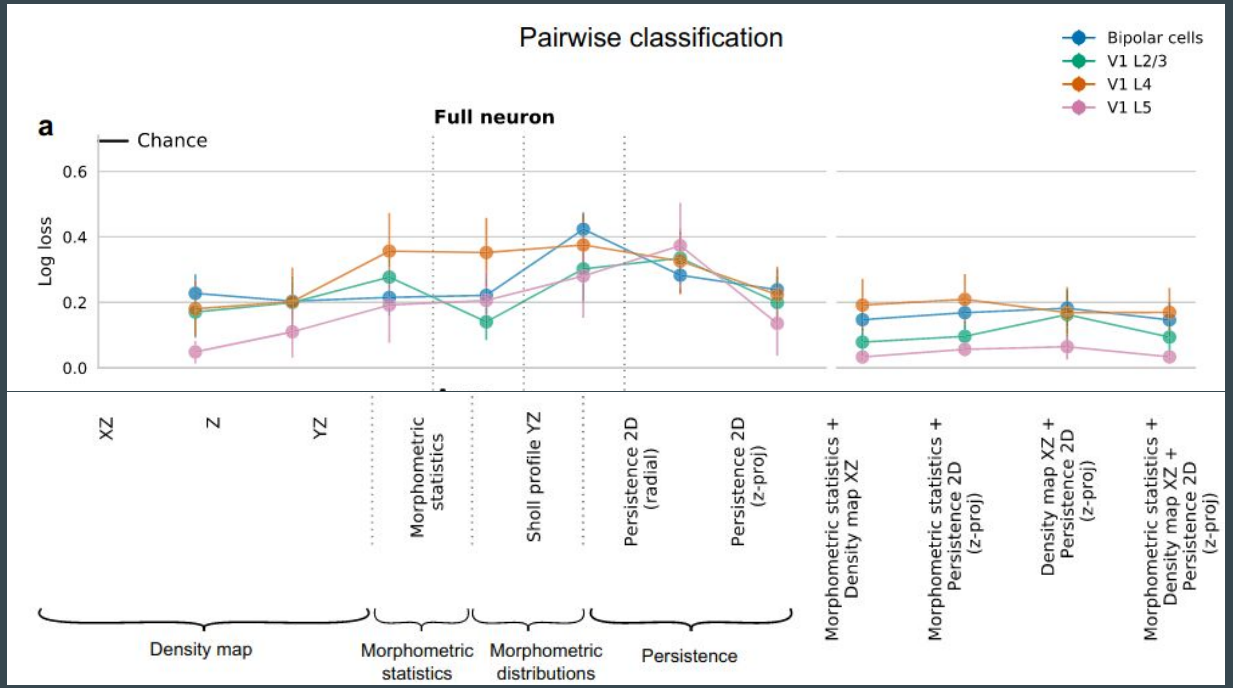
Methods: (1) feature extraction (2) Normalizations (3) Principal component analysis (dimensionality reduction) (4) Logistic regression / Elastic Net

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



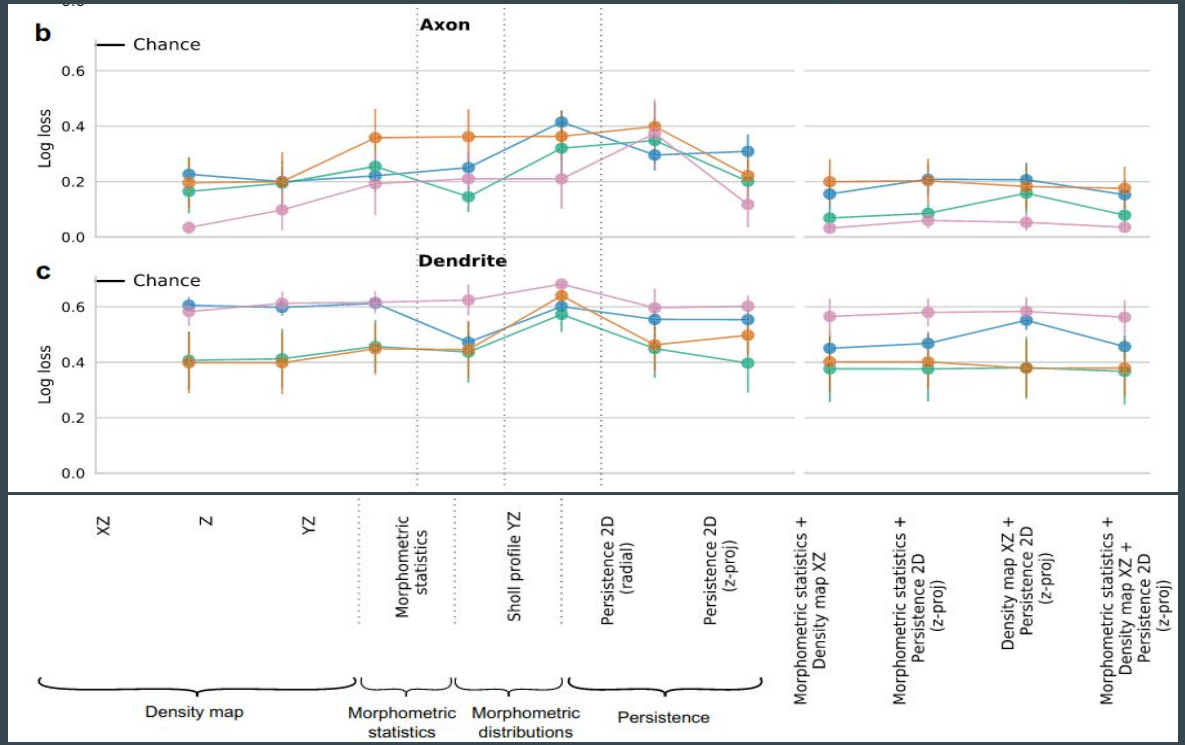
Cross-validated log-loss for each pair of morphological types in each data set using XZ density maps on full neurons as predictors in logistic regression. Zero log-loss corresponds to perfect prediction, $\ln(2) \approx 0.69$ corresponds to random guessing.

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



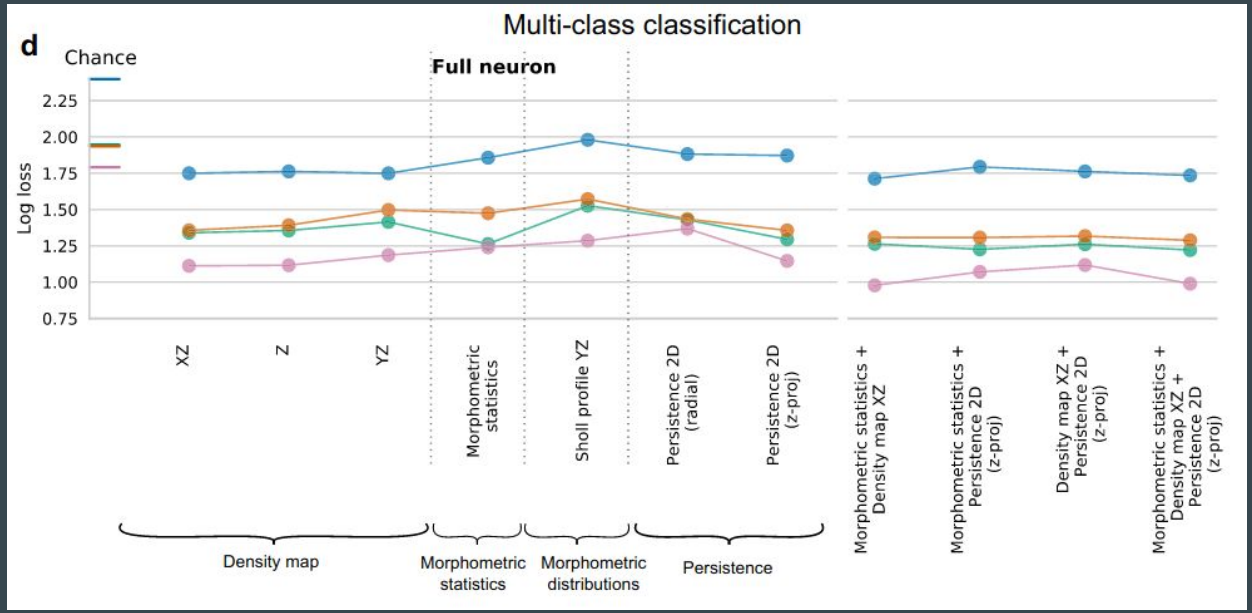
Pairwise classification performance of the top performing feature representations based on the full-neuron (a) features for each data set. Error bars correspond to 95% confidence intervals. Chance-level log-loss equals $\ln(2) \approx 0.69$.

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



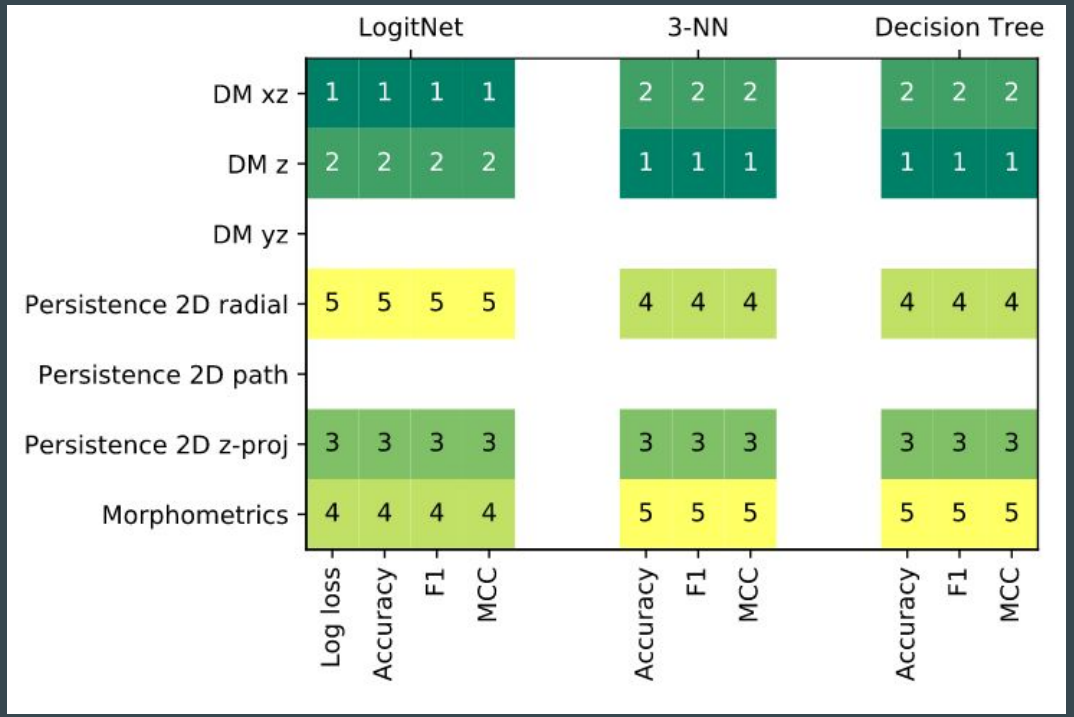
Pairwise classification performance of the top performing feature representations based on the axonal (b), and dendritic (c) features for each data set. Feature representations are grouped into density maps, morphometric statistics, morphometric distributions, persistence images, and combinations of the top three feature representations.

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



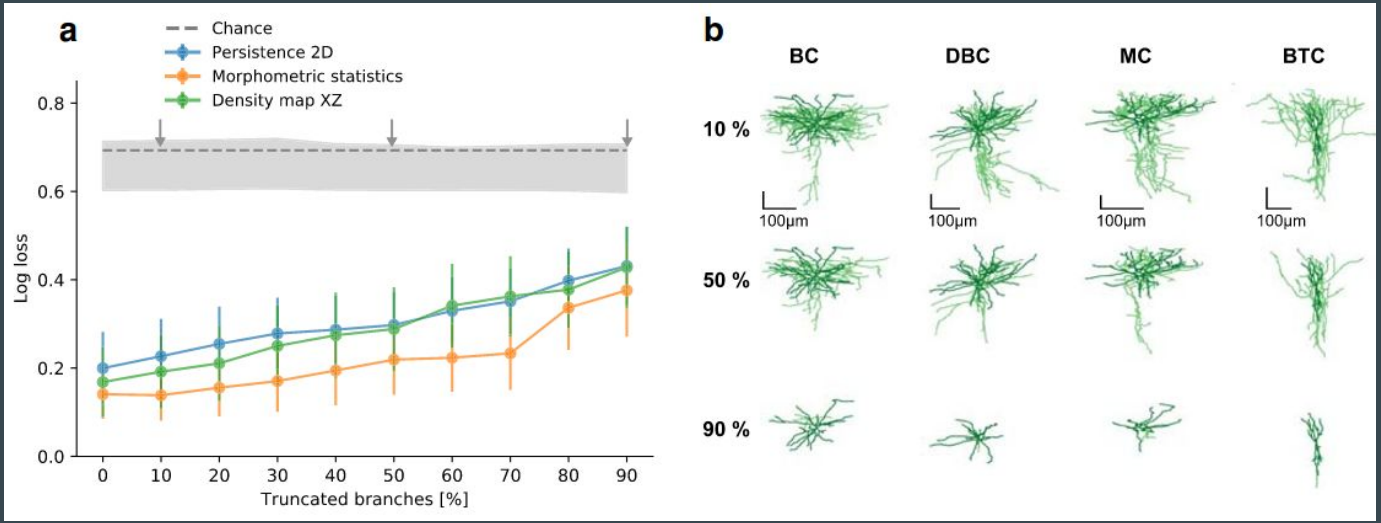
Multi-class classification performance of the top performing feature representations based on the full neuron features for each data set.

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



Ranked top five feature representations for each classification scheme using different performance measures on full-neuron data. All measures and all classification schemes selected the same top-5 features

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination



Cross-validated log-loss of XZ density maps, morphometric statistics and z-projection-based 2D persistence as a function of truncation level. Branches were truncated to mimic what happens when neurons are only partially traced

A Systematic Evaluation of Interneuron Morphology Representations for Cell Type Discrimination

- We found that density maps, z-projection-based 2D persistence images and morphometric statistics yield the best predictions of cell type labels, and showed that they do so even if substantial parts of the traced morphologies are removed.
- This study applied the same standardized classification procedure to each morphological representation, using well-curated data sets with well-defined cell types. This comparison revealed that density maps contain enough information to accurately discriminate most inhibitory cell types. This implies that the spatial extent and overall shape of the axonal arbour, as a consequence of a neuron's connectivity, are more relevant than precise branching characteristics

Questions?