

ASTR 3890 - Selected Topics: Data Science for Large
Astronomical Surveys (Spring 2022)

Supervised Classification

Dr. Nina Hernitschek
April 25, 2022

info: Take Home Exam

The take home exam will be available on `github` at 4pm today.

It will consist of multiple problems.

You have 1 week to solve the problems and to submit on `github` - i.e.: The exam is due Monday, May 2 at 4pm.

To avoid any issues with `github`, I recommend to frequently upload the code to `github`, e.g. after solving each problem.

info: Take
Home Exam

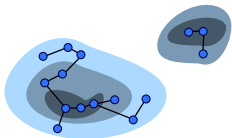
Supervised
Classification

Classification
Workflow

Classification
Algorithms

Supervised Machine Learning

we looked in detail at density estimation and clustering, which are **unsupervised** forms of classification



info: Take
Home Exam

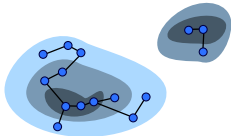
Supervised
Classification

Classification
Workflow

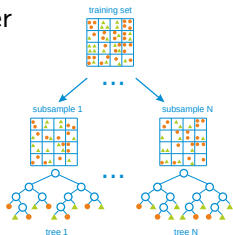
Classification
Algorithms

Supervised Machine Learning

we looked in detail at density estimation and clustering, which are **unsupervised** forms of classification



we now look in detail at **supervised** classification, where we actually know the 'truth' for a subset of our data and use that to *train* a classifier



Supervised vs. Unsupervised Machine Learning

Goals:

- In **supervised learning**, the goal is to predict outcomes for new data.
- In **unsupervised learning**, the goal is to get insights from large volumes of new data, where machine learning itself determines what is *interesting* from the dataset.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Supervised vs. Unsupervised Machine Learning

Goals:

- In **supervised learning**, the goal is to predict outcomes for new data.
- In **unsupervised learning**, the goal is to get insights from large volumes of new data, where machine learning itself determines what is *interesting* from the dataset.

Applications:

- **Supervised learning** models are ideal for e.g. astronomical source classification, e-mail spam detection, weather forecasting.
- In contrast, **unsupervised learning** is a great fit for anomaly detection, recommendation engines, and medical imaging.

Supervised vs. Unsupervised Machine Learning

Goals:

- In **supervised learning**, the goal is to predict outcomes for new data.
- In **unsupervised learning**, the goal is to get insights from large volumes of new data, where machine learning itself determines what is *interesting* from the dataset.

Applications:

- **Supervised learning** models are ideal for e.g. astronomical source classification, e-mail spam detection, weather forecasting.
- In contrast, **unsupervised learning** is a great fit for anomaly detection, recommendation engines, and medical imaging.

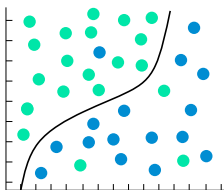
Complexity:

- **Supervised learning** models are generally computationally simpler, usually directly calculated through the use of packages in e.g. Python.
- In **unsupervised learning**, you need powerful tools for working with large amounts of unclassified data, thus often extended with faster programming languages such as C, usage of GPUs.

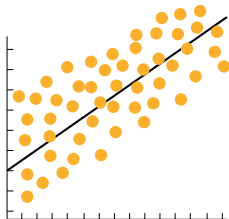
Supervised Machine Learning

Typical **tasks** for supervised machine learning:

- Classification: can we label the input?
- Regression: can we make a prediction?
- Recommendation: can we predict something based on user preference?



classification



regression

The Role of the Training Set

The objective of a supervised learning model is to predict the correct label for newly presented input data.

info: Take
Home Exam

**Supervised
Classification**

Classification
Workflow

Classification
Algorithms

The Role of the Training Set

The objective of a supervised learning model is to predict the correct label for newly presented input data.

When training a supervised learning algorithm, the **training set** will consist of inputs paired with the correct outputs. Inputs in the training set should represent the **target set** which we have to classify: composition of the data, data quality.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

The Role of the Training Set

The objective of a supervised learning model is to predict the correct label for newly presented input data.

When training a supervised learning algorithm, the **training set** will consist of inputs paired with the correct outputs. Inputs in the training set should represent the **target set** which we have to classify: composition of the data, data quality.

During **training**, the algorithm will search for patterns in the data that correlate with the desired outputs.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

The Role of the Training Set

The objective of a supervised learning model is to predict the correct label for newly presented input data.

When training a supervised learning algorithm, the **training set** will consist of inputs paired with the correct outputs. Inputs in the training set should represent the **target set** which we have to classify: composition of the data, data quality.

During **training**, the algorithm will search for patterns in the data that correlate with the desired outputs.

After training, a supervised learning algorithm will take in new unseen inputs and will determine which label the new inputs will be classified based on prior training data.

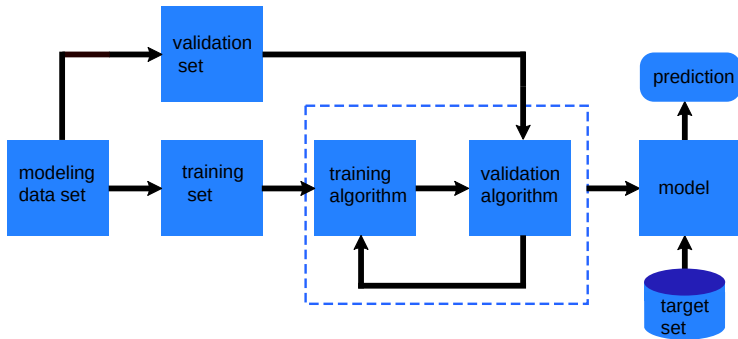
info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Classification Workflow



split modeling set into training set and validation set:

the validation set (also: test set) is used for the testing the model after the model has been trained on the training set - it is extremely important to test the model on data not being part of the training set

A **fundamental assumption** of supervised machine learning is that the distribution of training examples is identical to the distribution of validation examples and future unseen examples (the target set).

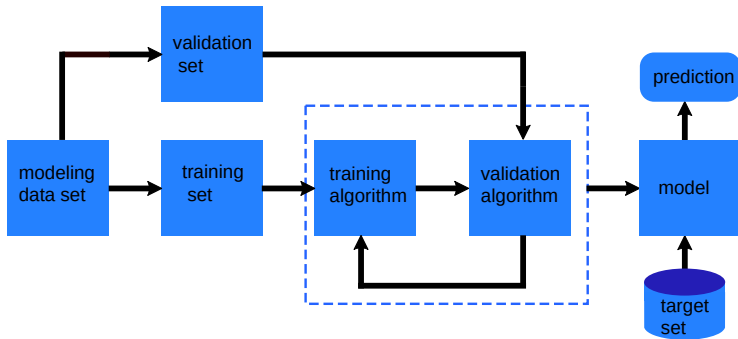
info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Classification Workflow



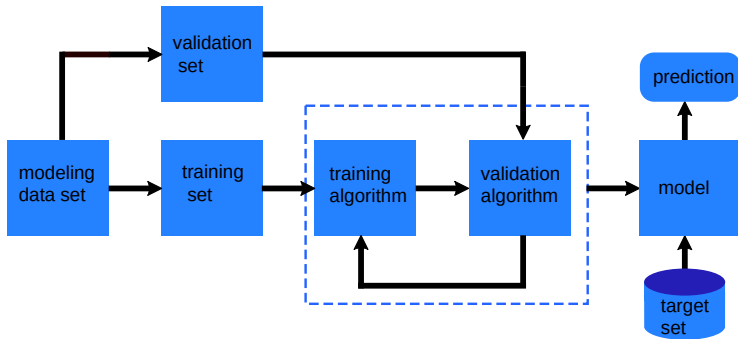
Training:

given a training set of labeled examples $(x_1, y_1, \dots, (x_n, y_n))$, estimate the prediction function f and parameters θ which minimizes the prediction error on the training set

Validation:

apply f to validation set x , output predicted value $y = f(x)$
from this we generate performance measures, also called accuracy measures

Classification Workflow



Application:

Run the model on the target set.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Pitfalls in Supervised Machine Learning

Overfitting: The model models the training data too well, thus does not **generalizes** well to unseen data (target set). Generalization refers to how well the concepts learned by a machine learning model apply to specific examples not seen by the model when it was learning.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Pitfalls in Supervised Machine Learning

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Overfitting: The model models the training data too well, thus does not **generalizes** well to unseen data (target set). Generalization refers to how well the concepts learned by a machine learning model apply to specific examples not seen by the model when it was learning.

Underfitting: Underfitting refers to a model that can **neither model** the training data **nor generalize** to new data. An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

Pitfalls in Supervised Machine Learning

Data leakage:

Data leakage happens when the training data contains information about the label, but similar data will not be available when the model is used for prediction. This leads to overly optimistic performance on the training and validation data, but the model will perform poorly in production on the target set data.

feature leakage (also: target leakage)

Feature or target leakage is caused by the inclusion of features (columns) into the training set which will not be available at the time you make predictions.

They are usually one of the following: a duplicate label, a proxy for the label, or the label itself. These features will not be available when the model is used for predictions.

For example, including `is_periodic` when predicting `phase_offset`, or a certain waveband from a targeted survey for e.g. only exoplanet host stars and is NaN otherwise.

Pitfalls in Supervised Machine Learning

Data leakage:

Data leakage happens when the training data contains information about the label, but similar data will not be available when the model is used for prediction. This leads to overly optimistic performance on the training and validation data, but the model will perform poorly in production on the target set data.

training set leakage

Row-wise leakage is caused by improper sharing of information between rows of data. Types of row-wise leakage include:

- premature featurization; leaking from premature featurization before CV/Train/Test split (must fit min, max, mean... on only the training set, split, then on the test set)
- duplicate rows between train/validation/test (e.g. oversampling a dataset to pad its size before splitting; e.g. bootstrap sampling before splitting; or duplicating rows to up sample the minority class)

Verification

never apply machine learning as a black box!

various **verification techniques** can be applied by splitting the modeling set into training and validation set

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Verification

never apply machine learning as a black box!

various **verification techniques** can be applied by splitting the modeling set into training and validation set

For simplicity, we consider here binary classification where each observation is assigned to either class 1 or 0 (= not 1).

In that case, there are the following outcomes (if you want identify class 1):

- True Positive = correctly identified (class 1 identified as class 1)
- True Negative = correctly rejected (class 0 rejected as class 0)
- False Positive = incorrectly identified (class 0 identified as class 1)
- False Negative = incorrectly rejected (class 1 rejected as class 0)

Verification

Based on these, we define either of the following pairs of terms:

$$\text{completeness} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

$$\text{contamination} = \frac{\text{false positives}}{\text{true positives} + \text{false positives}} = \text{false discovery rate}$$

Instead of contamination, often also efficiency (also called purity) is used:
 $\text{efficiency} = (1 - \text{contamination})$

or

$$\text{true positive rate} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

$$\text{false positive rate} = \frac{\text{false positives}}{\text{true negatives} + \text{false positives}}$$

Similarly

$$\text{efficiency} = 1 - \text{contamination} = \text{precision}.$$

Verification

To illustrate the differences between these measures, let's look at the following **example**:

We have a modeling set containing 100,000 stars and 1000 quasars. If you correctly identify 900 quasars and mistake 1000 stars for quasars, we have:

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Verification

To illustrate the differences between these measures, let's look at the following **example**:

We have a modeling set containing 100,000 stars and 1000 quasars. If you correctly identify 900 quasars and mistake 1000 stars for quasars, we have:

- $TP = 900$ (true positive)
- $FN = 100$ (false negative)
- $TN = 99,000$ (true negative)
- $FP = 1000$ (false positive)

Which gives

$$\text{true positive rate} = \frac{900}{900 + 100} = 0.9 = \text{completeness}$$

$$\text{false positive rate} = \frac{1000}{99000 + 1000} = 0.01$$

Verification

To illustrate the differences between these measures, let's look at the following **example**:

We have a modeling set containing 100,000 stars and 1000 quasars. If you correctly identify 900 quasars and mistake 1000 stars for quasars, we have:

- $TP = 900$ (true positive)
- $FN = 100$ (false negative)
- $TN = 99,000$ (true negative)
- $FP = 1000$ (false positive)

Which gives

$$\text{true positive rate} = \frac{900}{900 + 100} = 0.9 = \text{completeness}$$

$$\text{false positive rate} = \frac{1000}{99000 + 1000} = 0.01$$

question:

What do you think about these results?

Verification

To illustrate the differences between these measures, let's look at the following **example**:

We have a modeling set containing 100,000 stars and 1000 quasars. If you correctly identify 900 quasars and mistake 1000 stars for quasars, we have:

- $TP = 900$ (true positive)
- $FN = 100$ (false negative)
- $TN = 99,000$ (true negative)
- $FP = 1000$ (false positive)

Which gives

$$\text{true positive rate} = \frac{900}{900 + 100} = 0.9 = \text{completeness}$$

$$\text{false positive rate} = \frac{1000}{99000 + 1000} = 0.01$$

answer:

Despite the FPR doesn't look bad, there are a lot of stars, so the contamination rate isn't good: $\text{contamination} = \frac{1000}{900+1000} = 0.53$

Verification

To illustrate the differences between these measures, let's look at the following **example**:

We have a modeling set containing 100,000 stars and 1000 quasars. If you correctly identify 900 quasars and mistake 1000 stars for quasars, we have:

- TP = 900 (true positive)
- FN = 100 (false negative)
- TN = 99,000 (true negative)
- FP = 1000 (false positive)

Which gives

$$\text{true positive rate} = \frac{900}{900 + 100} = 0.9 = \text{completeness}$$

$$\text{false positive rate} = \frac{1000}{99000 + 1000} = 0.01$$

however:

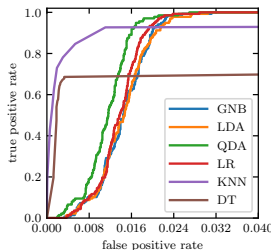
The classifier might be sufficient as one step in a classification pipeline.

Classifier Performance

tradeoff: contamination versus completeness



quantify this with a Receiver Operating Characteristic (ROC) curve which plots the true-positive vs. the false-positive rate



info: Take
Home Exam

Supervised
Classification

Classification
Workflow

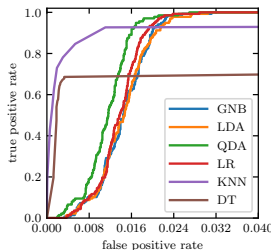
Classification
Algorithms

Classifier Performance

tradeoff: contamination versus completeness



quantify this with a Receiver Operating Characteristic (ROC) curve which plots the true-positive vs. the false-positive rate



One concern about ROC curves is that they are sensitive to the relative sample sizes: if there are many more background events than source events, small false positive results can dominate a signal.

For these cases we can plot completeness versus efficiency.

info: Take
Home Exam

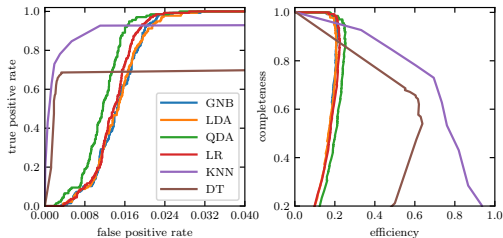
Supervised
Classification

Classification
Workflow

Classification
Algorithms

Classifier Performance

Here is a comparison of the two types of plots:



Here we see that to get higher completeness, you could actually suffer significantly in terms of efficiency, but your FPR might not go up that much if there are lots of true negatives.

Note that the desired completeness and efficiency is chosen by selecting a decision boundary. The curves show what these possible choices are. Generally, one wants to choose a decision boundary that maximizes the area under the ROC (or completeness versus efficiency) curve.

Classification Algorithms

With some assessment criteria defined, we can talk about classification algorithms itself.

info: Take
Home Exam


Supervised
Classification

Classification
Workflow

Classification
Algorithms

Classification Algorithms

Within supervised machine learning, we can further differentiate into **Generative** vs. **Discriminative Classification**:



```
graph TD; A[Within supervised machine learning, we can further differentiate into Generative vs. Discriminative Classification:] --> B[Which category is most likely to generate the observed result? using density estimation for classification => a full model of the density for each class is necessary]; A --> C[not caring about the full distribution, just defining boundaries => classification that finds the decision boundary that separates classes];
```

Which category is most likely to generate the observed result? using **density estimation** for classification \Rightarrow a full model of the density for each class is necessary

not caring about the full distribution, just defining boundaries \Rightarrow classification that finds the **decision boundary** that separates classes

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

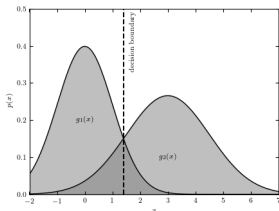
Classification Algorithms

Within supervised machine learning, we can further differentiate into **Generative** vs. **Discriminative Classification**:

Which category is most likely to generate the observed result?
using **density estimation** for classification \Rightarrow a full model of the density for each class is necessary

not caring about the full distribution, just defining boundaries \Rightarrow classification that finds the **decision boundary** that separates classes

example:



With these distributions, to classify a new object with $x = 1$, it would suffice to know that either

1. model 1 is a better fit than model 2 (generative classification), or
2. that the decision boundary is at $x = 1.4$ (discriminative classification).

Generative Classification

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

We can use Bayes' theorem to relate the labels to the features in an $N \times D$ data set X . The j th feature of the i th point is x_i^j and there are k classes giving discrete labels y_k . We have

$$p(y_k|x_i) = \frac{p(x_i|y_k)p(y_k)}{\sum_i p(x_i|y_k)p(y_k)},$$

where x_i is assumed to be a vector with j components.

$p(y = y_k)$ is the probability of any point having class k (equivalent to the prior probability of the class k).

In generative classifiers we model class-conditional densities $p(x|y = y_k)$.

Generative Classification

(a) The Discriminant Function

We can relate classification to density estimation and regression.

$\hat{y} = f(y|x)$ represents the best guess of y given x . So classification is just regression with discrete y values, e.g., $y = \{0, 1\}$.

In classification we refer to $f(y|x)$ as the discriminant function.

For a simple 2-class example, where $y = \{0, 1\}$:

$$\begin{aligned} g(x) = f(y|x) &= \int y p(y|x) dy \\ &= 1 \cdot p(y = 1|x) + 0 \cdot p(y = 0|x) = p(y = 1|x). \end{aligned}$$

and then using Bayes' rule:

$$g(x) = \frac{p(x|y = 1) p(y = 1)}{p(x|y = 1) p(y = 1) + p(x|y = 0) p(y = 0)}$$

The first equation is just the expectation value of y .

(b) Bayes Classifier

If the discriminant function gives a binary prediction, we call it a Bayes classifier, formulated as

$$\begin{aligned}\hat{y} &= \begin{cases} 1 & \text{if } g(x) > 1/2, \\ 0 & \text{otherwise,} \end{cases} \\ &= \begin{cases} 1 & \text{if } p(y = 1|x) > p(y = 0|x), \\ 0 & \text{otherwise.} \end{cases}\end{aligned}$$

This can be generalized to any number of classes, k , and not just two.

Generative Classification

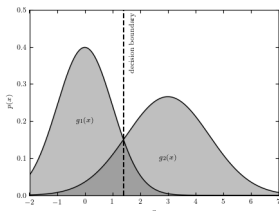
c) Decision Boundary

A decision boundary is just set of x values at which each class is equally likely:

$$p(x|y=1)p(y=1) = p(x|y=0)p(y=0);$$

$$g_1(x) = g_2(x) \text{ or } g(x) = 1/2$$

Below is an example of a decision boundary in 1-D, where each class is equally likely so we can just look at $p(x)$.



Discriminative Classification

Discriminative classification consists of methods that seek only to determine the **decision boundary in feature space**.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

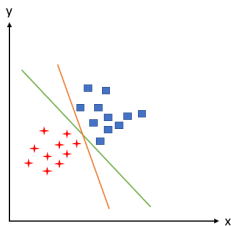
Discriminative Classification

Discriminative classification consists of methods that seek only to determine the **decision boundary in feature space**.

example:

We have the data as shown in the plot below. We could separate them by a line.

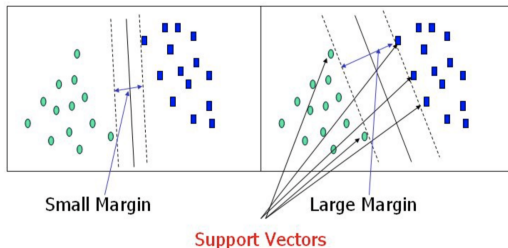
But: There are clearly lots of different lines that that would work. How do you do this optimally so it also works for the future target set? And what if the blobs are not perfectly well separated?



Discriminative Classification: Support Vector Machines

Support Vector Machines (SVM) define a **hyperplane** in $N - 1$ dimensions that maximizes the distance (the *margin*) of the closest point from each class. The points that touch the margin (or that are on the wrong side) are the **support vectors**.

There are lots of potential decision boundaries, but we want the one that maximize the distance of the support vectors from the decision hyperplane.



Discriminative Classification: Support Vector Machines

For **realistic data sets** where the decision boundary is not obvious, we relax the assumption that the classes are linearly separable. This changes the minimization condition and puts bounds on the number of misclassifications (which we would obviously like to minimize).

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Discriminative Classification: Support Vector Machines

Some comments on SVM:

- SVM is not scale invariant so it is often worth rescaling the data to $[0,1]$ or to whiten it to have a mean of 0 and variance 1 (remember to do this to the test data as well!).
- The data don't need to be separable (we can put a constraint in minimizing the number of 'failures').
- The median of a distribution is unaffected by large perturbations of outlying points, as long as those perturbations do not cross the boundary.

Discriminative Classification: Support Vector Machines

Some comments on SVM:

- In the same way, by maximizing the margin of support vectors rather than using all data points, SVM classification is similar to rank-based estimators. Once the support vectors are determined, changes to the positions or numbers of points beyond the margin will not change the decision boundary. For this reason, SVM can be a very powerful tool for discriminative classification.
- This is why there is a high completeness compared to the other methods: it does not matter that the background sources outnumber the RR Lyrae stars by a factor of ~ 200 to 1. It simply determines the best boundary between the small RR Lyrae clump and the large background clump.
- This completeness, however, comes at the cost of a relatively large contamination level.

Discriminative Classification: Decision Trees

A decision tree is a hierarchical application of decision boundaries:

- the top node contains the entire data set
- define some criteria to split the sample into 2 groups (not necessarily equal)
- splitting repeats, recursively, until a predefined stopping criteria is reached

info: Take
Home Exam

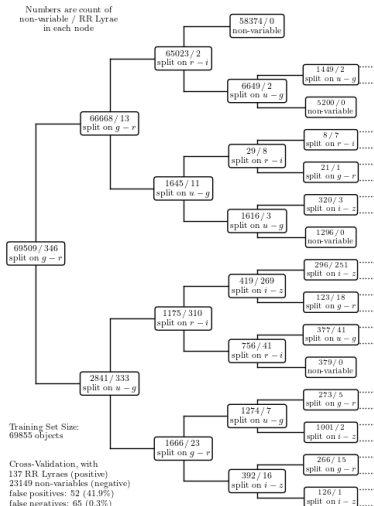
Supervised
Classification

Classification
Workflow

Classification
Algorithms

Discriminative Classification: Decision Trees

example:



The terminal nodes (leaf nodes) record the fraction of points that have one classification or the other in the training set.

The fraction of points from the training set classified as one belonging to one class or the other (in the leaf node) defines the class associated with that leaf node.

The binary splitting makes this extremely efficient. The trick is to ask the right questions.

Discriminative Classification: Decision Trees

One way to define **Splitting Criteria** is to use the information content (or *entropy*), $E(x)$, of the data

$$E(x) = - \sum_i p_i(x) \ln(p_i(x)),$$

where i is the class and $p_i(x)$ is the probability of that class given the training data. We can define the **information gain** as the reduction in entropy due to the partitioning of the data (i.e. by partitioning the data you have reduced the disorder). For a binary split with $i = 0$ representing those points below the split threshold and $i = 1$ as those points above the split threshold, the information gain $IG(x)$ is

$$IG(x|x_i) = E(x) - \sum_{i=0}^1 \frac{N_i}{N} E(x_i),$$

where N_i is the number of points, x_i , in the i -th class, and $E(x_i)$ is the entropy of that class. We are assessing the information gain as the difference between the entropy of the parent node and the sum of the entropies of the child nodes.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Discriminative Classification: Decision Trees

The typical process for finding the optimal decision boundary is to perform trial splits along each feature one at a time, within which the value of the feature to split at is also trialed. The feature that allows for the maximum information gain is the one that is split at this level.

Another commonly used "loss function" (especially for categorical classification) is the Gini coefficient:

$$G = \sum_i^k p_i(1 - p_i).$$

It essentially estimates the probability of incorrect classification by choosing both a point and (separately) a class randomly from the data.

Ensemble Learning

Ensemble learning is the process of using multiple models, trained over the same data, averaging the results of each model ultimately finding a more powerful prediction/classification result.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Ensemble Learning

Ensemble learning is the process of using multiple models, trained over the same data, averaging the results of each model ultimately finding a more powerful prediction/classification result.

The Random Forest algorithm combines ensemble learning methods with the decision tree framework to create multiple randomly drawn decision trees from the data, averaging the results to output a result that often times leads to strong predictions/classifications.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Ensemble Learning

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Ensemble learning is the process of using multiple models, trained over the same data, averaging the results of each model ultimately finding a more powerful prediction/classification result.

The Random Forest algorithm combines ensemble learning methods with the decision tree framework to create multiple randomly drawn decision trees from the data, averaging the results to output a result that often times leads to strong predictions/classifications.

The result is a more robust classifier.

Ensemble Learning: Random Forests

Random forests generate decision trees from bootstrap samples (drawing from the observed data set with replacement). This helps to overcome some of the limitations of decision trees.

In Random Forests, the splitting features on which to generate the tree are selected at random from the full set of features in the data.

The number of features selected per split level is typically the square root of the total number of features, \sqrt{D} .

The final classification from the random forest is based on the averaging of the classifications of each of the individual decision trees.

As before: cross-validation can be used to determine the optimal depth. Generally the number of trees, n , that are chosen is the number at which the cross-validation error plateaus.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Choosing the Right Classifier

no single model can be known in advance to be the best classifier

There are many factors, such as the size and structure of your dataset.

Advice: try many different algorithms for your problem, evaluate the performance for each and select the winner

Of course, the algorithms you try must be appropriate for your problem, which is where picking the right machine learning task comes in.

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms

Choosing the Right Classifier

In general, the level of accuracy increases for parametric models as:

- naive Bayes,
- linear discriminant analysis (LDA),
- logistic regression,
- linear support vector machines,
- quadratic discriminant analysis (QDA),
- linear ensembles of linear models.

For non-parametric models accuracy increases as:

- decision trees
- K -nearest-neighbor,
- neural networks
- kernel discriminant analysis,
- kernelized support vector machines
- random forests
- boosting

See also Ivezic, Table 9.1.

Break & Questions

afterwards we continue with `lecture_13.ipynb` from the `github` repository

info: Take
Home Exam

Supervised
Classification

Classification
Workflow

Classification
Algorithms