IAML Condensed Summary Notes For Quick In-Exam Strategic Fact Deployment

Maksymilian Mozolewski

December 9, 2020

Contents

IAML

Introduction

Machine Learning

A machine learning model **takes in** data, **outputs** predictions. It's a function of data really together with a set of training data.
Learning = Representation + Evaluation + Optimisation

Thinking about data

Classification

Sort data points into discrete buckets based on training data

Regression

Output a continuous/real value for each data point based on training data.

Clustering

Detect which data points are related to which other data points, find outliers.

Data representation

What format do we feed the data in ? Most likely as a **bag of features**. I.e. collection of attribute-value pairs, every data point must have an attribute-value pair for each property (in most cases)
Data representation has more impact on the performance of your ML algorithm than anything.

Types of attributes

- **Categorical**
  - e.g. red/blue/brown
  - a set of possible **mutually exclusive** values
  - meaningful operators: equality comparison
  - usually represented as numbers
  - problems: synnonymy is a major challenge e.g. some values might mean the same thing to a human but not to the machine (country == folk?)
- **Ordinal**
  - e.g. poor < satisfactory < good < excellent
  - a set of possible **mutually exclusive** values, but with a **natural ordering**
  - meaningful operators: equality comparison, sorting
  - problems: sometimes hard to differentiate from categorical (single < divorced)?
- **Numeric**
  - e.g. 3.1/5
  - meaningful operators: arithmetic, distance metrics, equality, sorting
  - problems:
    - sensitive to extreme outliers (handle these **before normalization**)
    - skewed distributions (assymetric) - outliers might actually be real data (e.g. personal wealth data)
    - Non-monotonic effect of attributes - e.g. predicting someone is going to win a marathon, here the relationship is not monotonic i.e. norect correlatio din, might be a curve with a "sweet spot"
  - solutions:
    - *Deal with outliers, maybe trim them for training phase only?*
    - *use a log/atan scale to make data more linear*
    - *discretize data into buckets*

Normalisation

Normalization is the process of converting all the data such that each different attribute is roughly in the same range, and comparable.
Normalization is mostly necessary for linear methods.

Picking attributes

We want:

- all our attributes to have similar values if the data points that posses them are similar themselves!
- small change in input $\rightarrow$ small change in values

Images

For images, can we use pixel data as attributes directly ? it depends, if the pixel 20,20 always corresponds to the middle of a letter then yes, this is a meaningful attribute which might help us discern whether digits given have strokes going through the middle of the image! What if the pixel is always something random ? This could happen whenever we are looking for an object which can be anywhere in the image, in which case the attribute would be gibberish! In the case of image classification, we want attributes which are:

- invariant to size
- invariant to rotation
- invariant to illumination

How can we classify whether an image contains a desired object ? In general we do the following:

- **Segment** the image, into regions of pixels which we believe are related (by colour, texture, position etc..)
- Pick a number of attributes which you will use to describe each of the regions
- Then use those features in your machine learning algorithm!

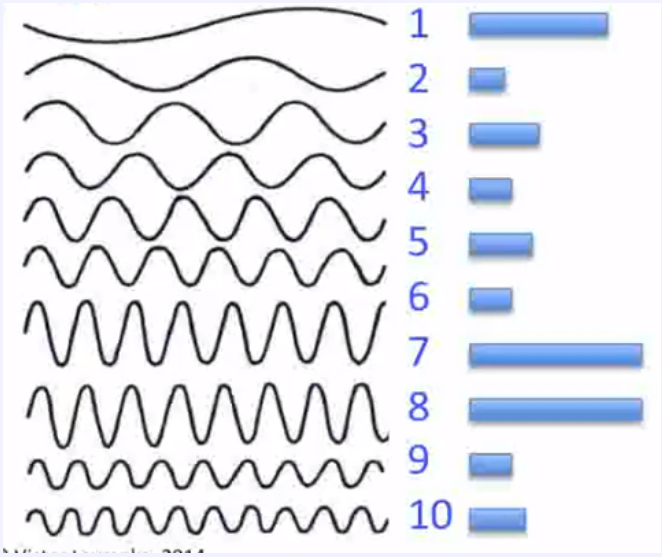
Keep in mind though, the segmentation **will** make errors, we can hope that these will be consistent across all images (all images will have their legs chopped off). Sometimes we can segment using a grid

Text

For textual data, often we can use the **bag-of-words** approach. I.e. we can form an attribute vector which counts the amount of occurrences of each word, regardless of position. This is invariant to word shuffling for example.

Sound

For sound, the data is sound waves. How can we select attributes here? We can count the number of different frequencies occurring in the piece, (using Fourier's transform) and treat this as a feature vector!



Supervised Learning

Supervised learning algorithms have some sort of "performance" metric they can use, i.e. test labels they can validate their guesses on. When the algorithm can measure accuracy directly it's a supervised algorithm.

Unsupervised Learning

Learning without a specific accuracy measure available. Algorithms in this area usually look for structure/patterns/information in the data which can be helpful in other ways. There is nothing specific the algorithm is looking for. Can be **direct** when the algorithm helps to make sense of the data directly, or **indirect** when it is "plugged" into another machine learning aglorithm as an attribute itself.

Semi-supervised Learning

Using unsupervised methods to improve supervised algorithms. Usually have a few labelled examples but lots more unlabelled.

Multi-class classification

Classification with multiple mutually exclusive labels/classes.
Might be hard to tell when something belongs to none of the available classes.

Binary classification

Classification with 2 mutually exclusive labels/classes in each "run". This way of classification can be applied to multiple-classes classification but with a "One-vs-Rest" meta-strategy (a vs not a, b vs not b). In this way a sample may belong to multiple classes but never to two sides of the one-vs-rest structure simultaneously in each run.
In this classification method we can actually tell when something doesn't belong to any class!

Analysing data

We have to check for a number of things in our data sets:

- Are there any dominating classes ? what would the best "dummy" model do ? always predicting no ?
- What should we use as the appropriate error metric ? How important are the false positives vs the false negatives ?

Generative model

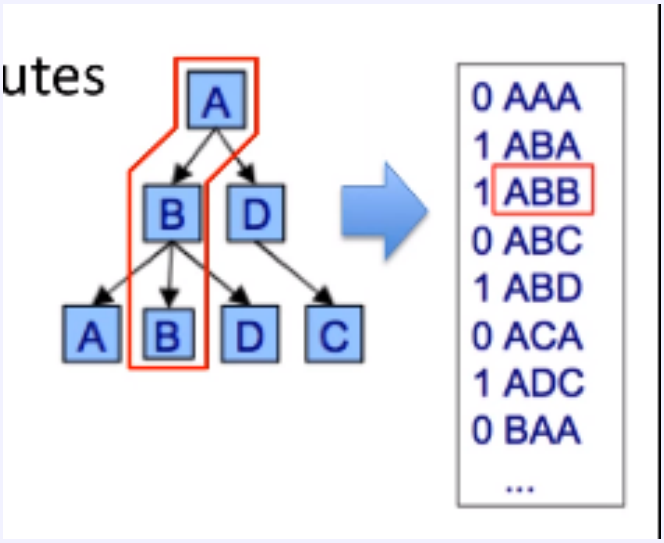
A generative model, develops a probabilistic model of each class, i.e. tries to "model" the underlying probability distribution directly. The decision boundary becomes implicitly defined by the probabilities of each input being in each class.

Discriminative model

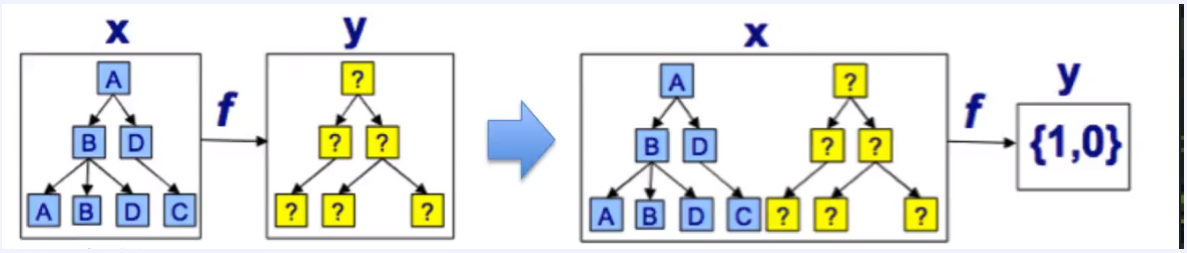
A discriminative model ignores the underlying model and tries to "separate" the data, i.e. it tries to model the boundaries that divide the classes. **Not designed to use unlabeled data** so cannot be used for unsupervised tasks.

Dealing with data structure

What do we do if the data input has some sort of hierarchical structure ? Where the position of occurrence of a node affects its meaning? We can encode as attributes the existence of root-to-leaf paths in the entire tree, and use this bag-of-words approach to perform machine learning



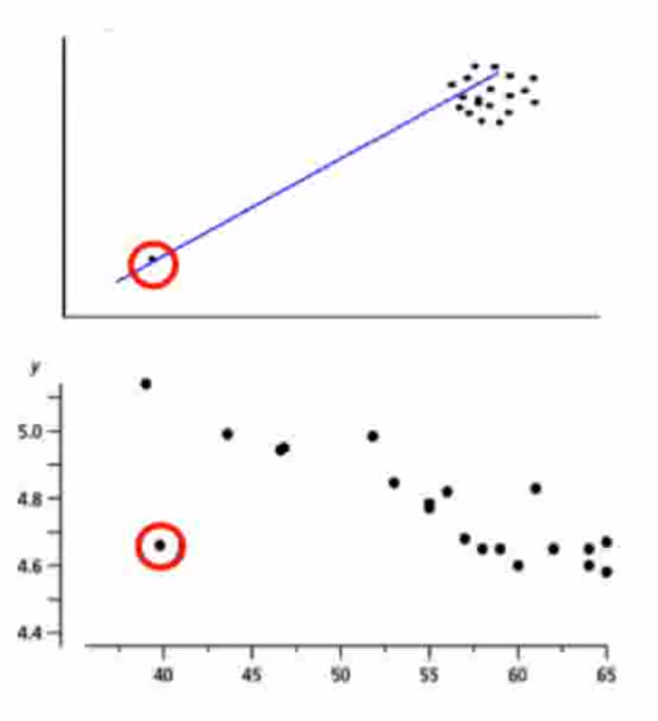
What if we need to predict the output structure from the input structure ? This is very difficult, but we can "trick" our classifier and turn this more into a search problem by embedding the possible outputs with each input and classifying on that instead:



This of course means we have to search for all possible output structures!

Dealing with outliers

Outliers are isolated instances of a class that are unlike any other instance of that class. These affect all learning models to some degree.
There are some ways we can deal with outliers. One method is to remove the outliers just before we perform any sort of normalisation on the data, (ONLY FOR THE PURPOSES OF TRAINING!!) We can also put a confidence interval around our data, and removing values outside of those intervals (with x,y values outside of a normal range). Some data points might still be outliers even though they are within expected x,y ranges! (second figure)



Best way to deal with outliers ? **VISUALISE YOUR DATA**

Naive Bayes

Decision Trees

Generalisation & Evaluation

Linear regression

Logistic regression

Optimisation & Regularisation

Support Vector Machines

Ethics

Nearest Neighbours

K-Means

Gaussian mixture models

Principal components analysis

Hierarchical Clustering

Perceptrons

Neural networks