Introductory Applied Machine Learning

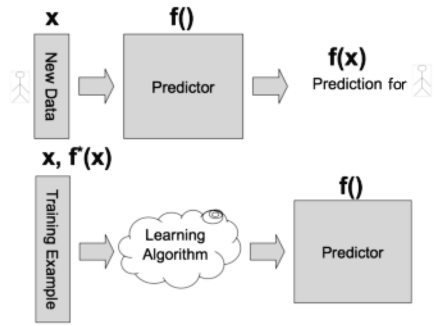
Thinking about Data

Victor Lavrenko and Nigel Goddard
School of Informatics
University of Edinburgh

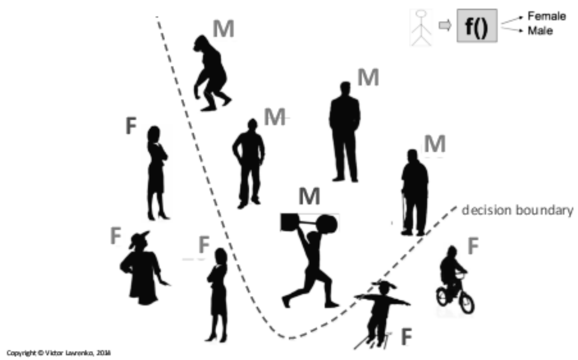
Overview

- What is machine learning?
  - examples: classification, regression, clustering
- Attribute-value pairs
  - bag-of-features representation
  - categorical attributes
  - ordinal attributes
  - numeric attributes, issues
- Examples of real data

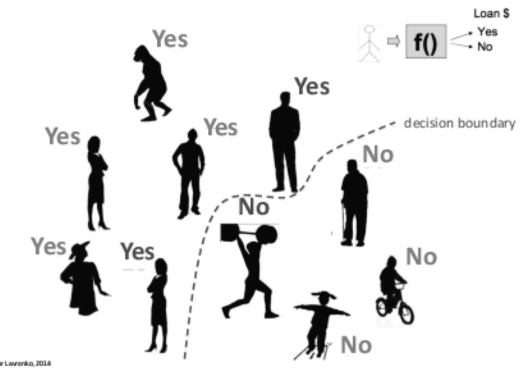
Learning from Examples



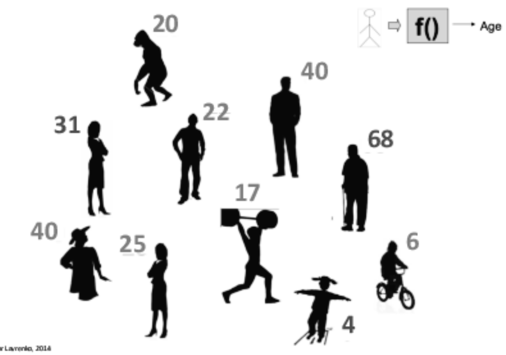
Classification (supervised learning)



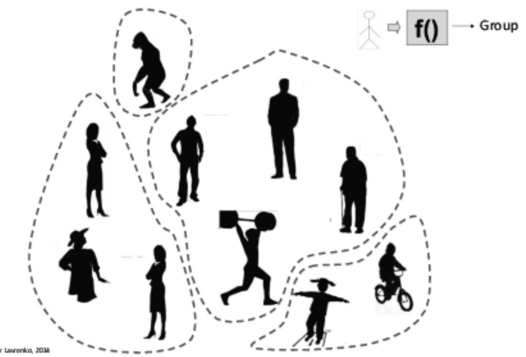
Classification (supervised learning)



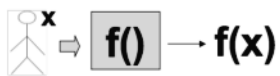
Regression (supervised learning)



Clustering (unsupervised learning)



Representing Data



- How do we represent mathematically?
- Depends on what we're trying to do:
  - deciding to loan money?
  - predicting gender?
- Represent as a set of attribute-value pairs
  - example: **x** = {height=*180cm*, eyes="*blue*", job="*student*"}

Attribute-value pairs

- **x** = {height=*180cm*, eyes="*blue*", job="*student*"}
- un-ordered "bag-of-features"
  - if structure is essential – embed it in the attributes
- Have to convert any dataset to this form
- Generally three types of attributes:
  - categorical: *red, blue, brown, yellow*
  - ordinal: *poor, satisfactory, good, excellent*
  - numeric: *-3.14, 6E23, 0, 1*

Categorical attributes

- Each instance falls into one of a set of categories
  - **genre**: {*classical, jazz, rock, techno*}
  - categories are mutually exclusive
- Categories usually encoded as numbers
  - no natural ordering to categories
  - only equality testing (=,≠) is meaningful
- Synonymy a major challenge for real datasets:
  - e.g. social tags: *country == folk? house == techno?*

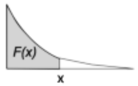
Ordinal attributes

- Instance falls into one of a set of categories
- There is a natural ordering to categories
  - **education level**: {*none, school, university, post-graduate*}
  - **Likert scale**: {*disagree, neutral, agree, strongly agree*}
- Encoded as numbers to preserve ordering
  - meaningful to compare values: (<,=,>)
  - should not add / multiply / measure "distance"
- Sometimes hard to differentiate from categorical:
  - does {*single, married, divorced*} have a natural ordering?

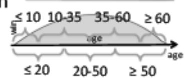
Numeric attributes

- Integers or real numbers
  - meaningful to add, multiply, compute mean / variance
  - integers not always the same as real numbers
- Usually want to normalize values (why?)
  - zero mean, unit variance: x' = (x - mean) / st.dev
  - sometimes want [0,1]: x' = (x - min) / (max - min)
- Sensitive to extreme (unusually large/small) values
  - e.g. height: {165,167,171,175,176,181,183,1820}[cm]
  - must handle this before normalization

Numeric attributes: issues



- Skewed distributions
  - systematic extreme values
  - affects regression, kNN, NB; but not DTs
  - simple fix: log(x) or atan(x), then normalize
  - cumulative distribution function: x' = F(x) = P(X≤x)
- Non-monotonic effect of attributes
  - affects regression, NB, DTs(gain); less important for kNN
  - monotonic: net worth and lending risk
    - higher net worth → lower lending risk
  - non-monotonic: age → win a marathon
    - sweet spot: not too young, not too old
  - simple fix: quantization
    - can be unsupervised, overlapping



Overview

- Attribute-value pairs
- Examples of real data
  - credit scoring
  - handwritten digits
  - object recognition
  - text classification
- Issues to consider

Example: credit scoring

- Numeric attributes:
  - loan amount (e.g. *$1000*)
  - installment / disposable income (e.g. *35%*)
- Ordinal:
  - savings: {*none, <100, 100..500, 500..1000, >1000*}
  - employed: {*unemployed, <1yr, 1..4yrs, 4..7yrs, >7yrs*}
- Categorical:
  - purpose: {*car, appliance, repairs, education, business*}
  - personal: {*single, married, divorced, separated*}
  - housing: {*for free, rents, owns*} ← perhaps ordinal?

Picking attributes

- Previous example: obvious attributes
  - not always the case (e.g. images)
- How do we pick a representation?
- Think about what we're trying to accomplish:
  - we're learning a predictor: f(x) → y
  - x should encode some information relevant to y
  - idea: "similar" representations iff x_1, x_2 in the same class:
    - similar values for attributes if x_1, x_2 in the same class
    - dissimilar values if not
    - "similar" not always a straightforward concept
  - a good intuition for thinking about representations

Example: digit recognition

- Recognize handwritten digits
  - application: automatic postal code processing
- Offline process
  - input: bitmap image
  - no pen stroke data
- Challenges:
  - varying style, slant pressure, pen type

Handwritten digits: attributes

- Represent each pixel as a separate attribute
  - 400 attributes (20x20 bitmap) $X = X_1 X_2 \ldots X_{20} X_{21} X \ldots X_{400}$
  - each attribute is a real number
    - degree of "blackness" of a pixel
  - could represent as binary (0,1)
    - 0 (white) if $x_i < t$, else 1 (black)
    - natural, space/CPU-efficient
  - thinking in terms of similarity
    - (0,1) will increase mismatches
    - may want to do the opposite: "blur" the image

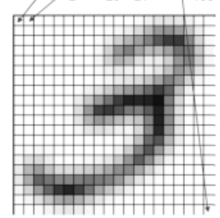
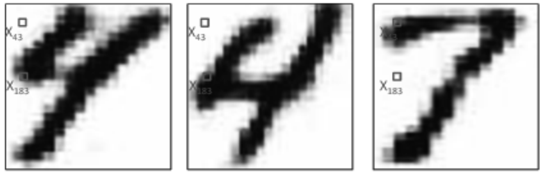
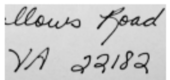


Image pixels as attributes

- works when same pixel = same meaning
  - X_{43} … stroke in the upper-left corner



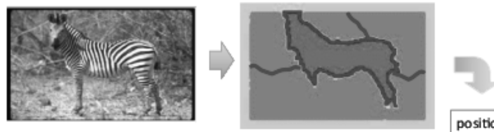
- true because we can isolate each digit, rescale, de-slant



Example: object recognition

- Recognize object in an image
  - animals, faces, military targets
- Input is a photograph (bitmap)
- Challenges:
  - position in a photo, orientation, scale
  - lighting differences, obstructions
- Using pixels as attributes will not work
  - want something that makes different zebras "similar"
  - many ways to achieve this, will outline one possibility

Object recognition: attributes



position (x,y)
relative area
circumference
convexity
orientation
color freq.
texture filters

- Segment the image into "regions"
  - algorithms: BlobWorld, Normalized Cuts
- Compute features describing the region
- Segmentation will make errors
  - hope these errors are systematic (same for all zebras)
  - sometimes can get away with simple rectangular grid

Example: text classification

- Assign class label to a text document
  - detect spam, identify topics/genres, predict events
  - input: string of characters
  - idea: words carry meaning
- Naïve way: words as values

X_1 = this
X_2 = proposition
X_3 = on
X_4 = behalf
X_5 = of
X_6 = mr
X_7 = lee
X_8 = kun

X_1 = this
X_2 = investment
X_3 = proposition
X_4 = on
X_5 = behalf
X_6 = of
X_7 = mr
X_8 = lee

Dear Sir

This ~~investment~~ proposition on behalf of Mr Lee Kun Hee (former chairman of Samsung Electronics). He requires an experienced business person or company that can profitably invest monies in excess of Fifty Two million US Dollars (US$52m) only, outside Asia. The sum of money will be paid from African Development Bank Group, South Africa…

Text classification: attributes

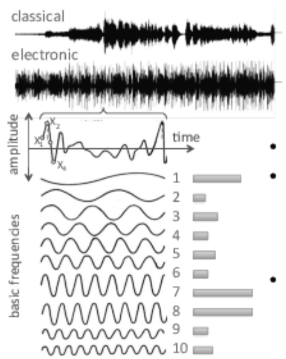
- Better way: words as numeric attributes
  - one attribute for **every possible word** in the language
  - value: 1 if word was observed in email, 0 otherwise
    - may use frequencies or tf-idf weights
  - note: 10^5-10^6 attributes, 99.99% zeros

Dear Sir

This investment proposition on behalf of Mr Lee Kun Hee (former chairman of Samsung Electronics). He requires an experienced business person or company that can profitably invest monies in excess of Fifty Two million US Dollars (US$52m) only, outside Asia. The sum of money will be paid from African Development Bank Group, South Africa..

0 aardvark
0 apple
1 africa
1 bank
0 bear
1 business
0 cat
1 funds
0 gorilla
1 investment
0 zebra
0 zoo

Example: music classification



- Music = time series
- Naive representation:
  - sample at regular intervals
  - X_t = amplitude at time t
- Periodic series = Σ sine waves
- Fourier transform:
  - decompose music into base frequencies f
  - find "weight" of each f
- Representation:
  - X_f = weight of frequency f
  - insensitive to shift, volume

Issues in Machine Learning

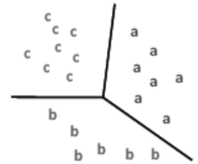
- Supervised vs. unsupervised
- What are we predicting?
- Outliers in the data
- Missing data
- Generative vs. discriminative
- Dimensionality

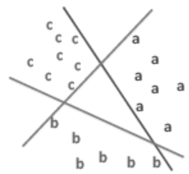
Supervised vs. Unsupervised

- Supervised learning:
  - trying to predict a specific quantity
  - have training examples with labels
  - can measure accuracy directly
- Unsupervised learning:
  - trying to "understand" the data
  - looking for structure or unusual patterns
  - not looking for something specific (supervised)
  - does not require labeled data
  - evaluation usually indirect or qualitative
- Semi-supervised:
  - using unsupervised methods to improve supervised algs.
  - usually few labeled examples + lots of unlabeled

Multi-class vs. Binary classification

- Multi-class:
  - classes mutually exclusive:
    - instance is either a or b or c
    - even if it's an outlier
  - NB, kNN, DT, logistic
- Binary:
  - one-vs-rest:
    - {a} vs {not a}, {b} vs {not b}
  - classes may overlap
    - instance can be both a and b
    - can be in none of the classes
  - SVM, logistic, perceptron



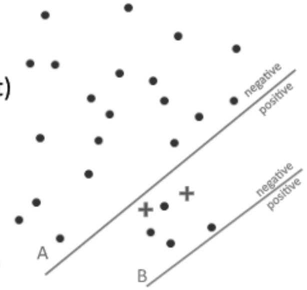


What are we predicting?

- Are there dominating classes?
  - does it affect anything?
- Example:
  - *Predict if scientific publication will lead to a Nobel prize*
  - claim: have a classifier that will be at least 99.99% accurate
- What is the appropriate error metric?
  - relative cost of false positives / false negatives
  - medical diagnosis vs. investment opportunities

Accuracy and un-balanced classes

- You're predicting Nobel prize (+) vs. not (•)
- Would you prefer classifier A or B?
- Is accuracy (% correct) higher for A or B?
- Accuracy / error rate poor metric here
- Want:
  - cost (Miss) > cost (FA)



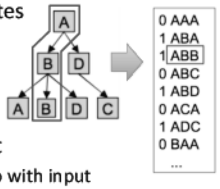
Generative vs. Discriminative

- Generative:
  - probabilistic "model" of each class
  - decision boundary:
    - where one model becomes more likely
  - natural use of unlabeled data

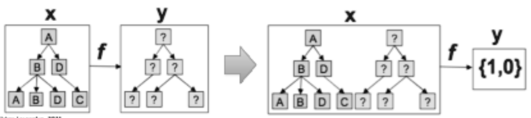
- Discriminative:
  - focus on the decision boundary
  - more powerful with lots of examples
  - not designed to use unlabeled data
  - only supervised tasks

Dealing with structure

- Structured input: embed in attributes
  - e.g. tree w. free branching, labels
    - meaning of "A" depends on level
    - one possible representation:
      - attributes = root-to-leaf paths

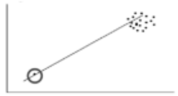


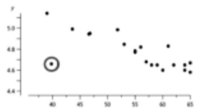
- Structured output: embed in input
  - predict 1/0: output does / doesn't go with input
  - search over possible outputs becomes main focus



Outliers in the data

- Isolated instances of a class that are unlike any other instance of that class
  - affect all learning methods to various degrees
- Extreme attribute values:
  - detect: confidence interval
  - remove or threshold
- Dissimilar to other instances
  - remove or try to fix (mis-labeled?)
- Always try to visualize the data
  - helps detect many irregularities





Missing data

- Real datasets often contain missing values
  - usually in relational (pre-structured) data
- Try to understand why
  - random? systematic?
- Categorical: introduce a special label ("N/A")
- Numeric:
  - "fill-in" the value (e.g. mean of all observed values)
  - remove instance altogether
  - remove offending attribute from all instances
- Some methods explicitly handle missing values
  - Naïve Bayes, rule-learners, decision trees
- Do we need to fill-in the values explicitly?

Other issues

- Assumptions about the data:
  - generated how? sources?
  - smooth? linear? noise?
  - bias towards particular type of model?
- Computational:
  - how fast does it have to be at prediction time?
  - do you care about training time?
    - what if you could use 100x more data if training was faster?
  - will you need to update / re-train frequently?
  - storage limits for the model / instances?