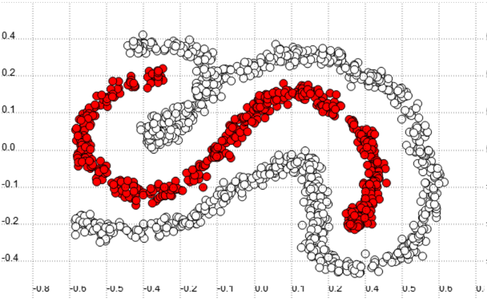
Handling unbalanced datasets and missing data
Incremental Learning

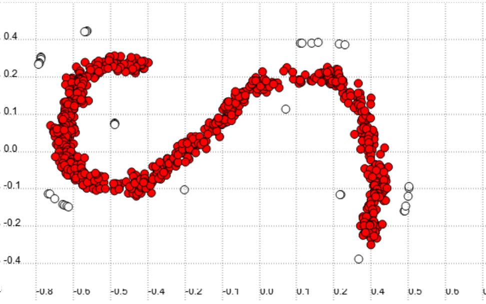
ML techniques assume balanced datasets

ML algorithms assumes that data are balanced.

- In classification or clustering, this means that one has comparative number of instances of each class.
- In regression, this means that samples are uniformly distributed.



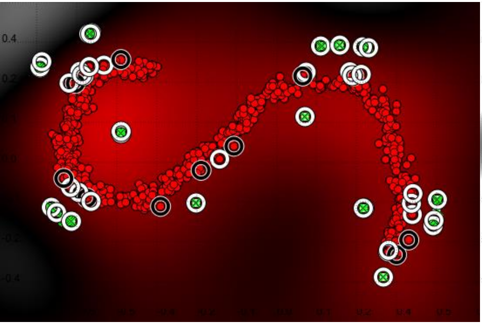
Balanced dataset



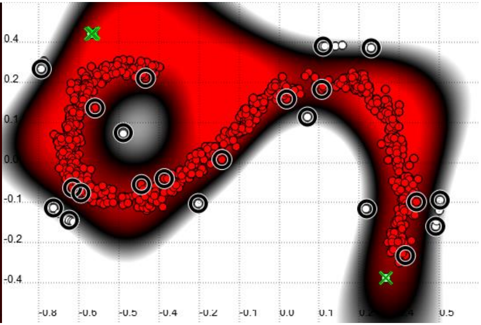
Imbalanced dataset
826 dataset points: 790 positive class, 36 negative class

Example of poor classification due to imbalance

If one does not apply a high penalty, SVM results in poor classification.

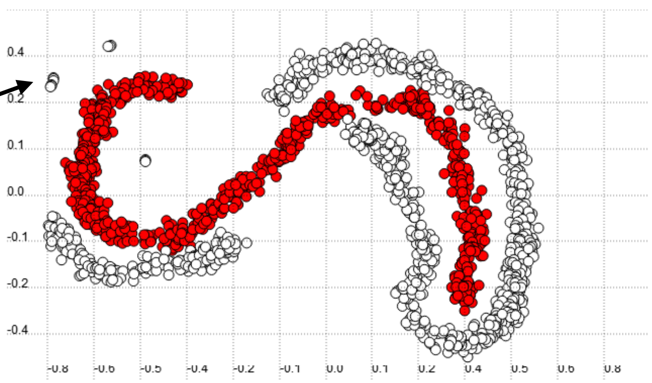


Low penalty for missclassified data
C=1



High penalty for missclassified data
C=1000

Imbalance versus missing data



Too few datapoints

Dataset is **balanced across classes** (comparable # of datapoints in both classes)

The white dataset is **not balanced within its class** (more data in some regions than others)

Imbalance data: examples

Imbalance arises as a result of:

- Rarity of data (# examples of tumor cells,)

E.g. Finance: modeling account creation likelihood based on demographics
Number of clients who closed account vs. nm clients who did not: 1%

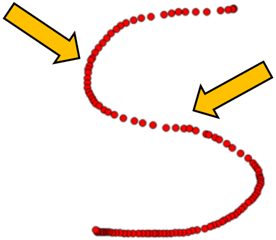
- Recording method (sampling frequency versus speed of observed phenemenon)

E.g. Robotics: Modeling self-obstacle avoidance
Number of joint configurations where the robot's limbs intersect with one another vs. number of configurations with no intersection: 4-5%

Imbalance data: examples

Record 2D position of a pencil during hand draw of a S shape.
Recording is gathered at a fixed frequency.

More datapoints when the pencil slows down



Fewer datapoints when the pencil accelerates

Imbalance: when, where and why

- ❑ Imbalances arises anywhere:
  - Between classes
  - Within classes

- ❑ Imbalance can sometimes result in complete lack of datapoints in an area, but not necessarily. It should not be confused with missing data that may result from neglecting areas when sampling.

- ❑ Imbalance affects often the group of points that matter the most (the rarest).
  - E.g in classification, one has usually much less datapoints from the adverse class (tumor vs no tumor).
  - This is unfortunate as we care most about avoiding misclassifying elements of this class.

Imbalance $\neq$ Rarity

Relative imbalance and absolute rarity

1’000’000 : 1’000 = 1’000 : 1 ?

The minority class may be outnumbered but may not be necessarily rare.

This case can be resolved by sampling more.

If the minority class is rare, this must be taken into account during learning, e.g. by outweighing the rare class.

Missing data

Values may be missing for various reasons:

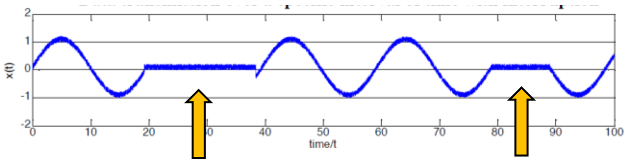
- sensor malfunction
- expensive data-gathering
- information non entered

Usually denoted by: NaN, or ?

Example the SECOM dataset:
https://archive.ics.uci.edu/ml/datasets/SECOM

Imbalanced Data vs. Missing Data

Continuous data transmission with sudden interruptions



Interruptions lead to transmission of value zero, a valid entry!

This leads the value zero to be over-represented and accounting for two types of information (true signal, missing signal)

Handling Missing Data

- Replace with the mean/most frequent value
- Approximate with regression/classification
- Interpolate when handling time-series

Missing data

Replacing with the mean/most frequent value

	Car width	Manufacturer
car 1	2.3.	?
car 2	?	Volkswagen
car 3	1.7	Volkswagen
car 4	1.8	Volkswagen
car 5	2	BMW
car 6	2.4	BMW
car 7	2.1	BMW
car 8	1.8	Volkswagen

Table: A toy dataset with missing values

Continuous data → replace with mean → width of car 2 is 2.01 Categorical data → replace with the most frequent label → Manufacturer of car 1 is Volskwagen

Missing data

Approximate missing values by performing regression/classification

$$y = f(x)$$

In regression, $f(x)$ predicts continuous output. In classification, $f(x)$ predicts the label.

- y is the dimension with missing data and
- x all the other dimensions.
- x contains only samples without missing data perform
- cross-validation as usual

Missing data

Approximate missing values by performing regression/classification

	Car width	Manufacturer
car 1	2.3.	?
car 2	?	Volkswagen
car 3	1.7	Volkswagen
car 4	1.8	Volkswagen
car 5	2	BMW
car 6	2.4	BMW
car 7	2.1	BMW
car 8	1.8	Volkswagen

Table: A toy dataset with missing values

Use regression for *car width* entries

Use classification for *manufacturer* entries

Missing data – Expectation maximization[1]

Model data as a mixture model using:

- Gaussian distributions when data is continuous,
- Idea: Handle missing values similarly to unknown model parameters Joint distribution:

$$p(X|\vartheta) = p(X^{o}, X^{m}|\vartheta) = p(X^{o}|\vartheta)\, p(X^{m}|X^{o}, \vartheta)$$

X^{o} observed data, X^{m} missing data

Recompute Log-likelihood:

$$\begin{aligned} L(\vartheta|X) &= L(\vartheta|X^{o}, X^{m}) \\ &= L(\vartheta|X^{o}) + \log P(X^{m}|X^{o}, \vartheta) \end{aligned}$$

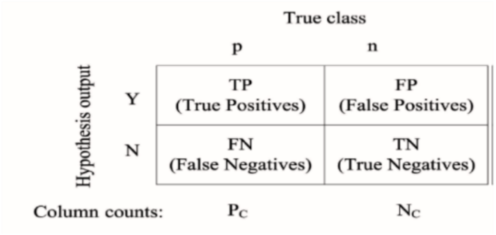
Unbalanced Datasets

Unbalanced datasets: One class contains significantly less samples than others (example: the Exoplanet detection dataset)

- Report significant metrics (classification error vs confusion matrix)
- Rebalance datasets:
  - Down-sample dominant classes
  - Over-sample (create artificial samples) non-dominant classes
- Modify cost function

Unbalanced Datasets – Reporting significant metrics

Traditional accuracy measure are sensitive to the data distribution



		True class	
		p	n
Hypothesis output	Y	TP (True Positives)	FP (False Positives)
	N	FN (False Negatives)	TN (True Negatives)
Column counts:		P_C	N_C

$$Accuracy = \frac{TP + TN}{P_C + N_C}$$

$$ErrorRate = 1 - accuracy$$

The F-measure is better adapted as it evaluates performance on one class

$$F\text{-}Measure = \frac{(1+\beta)^2 \cdot Recall \cdot Precision}{\beta^2 \cdot Recall + Precision}$$

$\beta = 1$, usually

$$Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN},$$

Unbalanced Datasets – Reporting significant metrics

Confusion matrices provide information on which classes are merged (confused) with which by a classifier.

Actual Class \ Predicted Class	C_1	C_2	$\cdots$	C_c
C_1	n_{11}	n_{12}	$n_{1\ldots}$	n_{1c}
C_2	n_{21}	n_{22}	$n_{2\ldots}$	n_{2c}
.	.	.	.	.
C_c	n_{c1}	n_{c2}	$\cdots$	n_{cc}

Table: Structure of a confusion matrix

$n_{ij} \rightarrow$ Number of samples that belong to class i and classified as class j

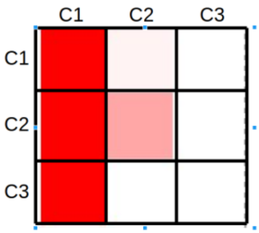
Unbalanced Datasets – Reporting significant metrics

Classification error vs. Unbalanced Dataset

Comparison of two classifiers (kNN - GMM) on an unbalanced dataset:

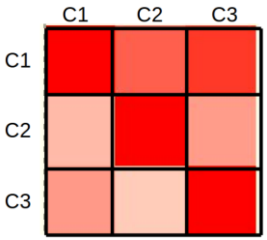
kNN:

- Classification Error: 17.4 %
- Confusion Matrix:



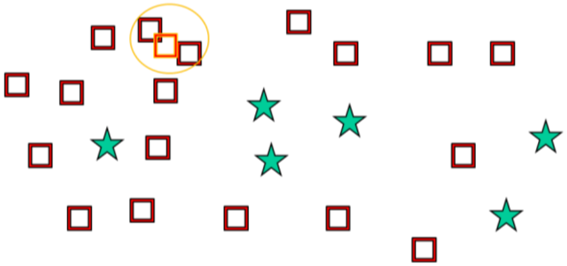
GMM:

- Classification Error: 55 %
- Confusion Matrix:



Unbalanced Datasets – Down-sampling

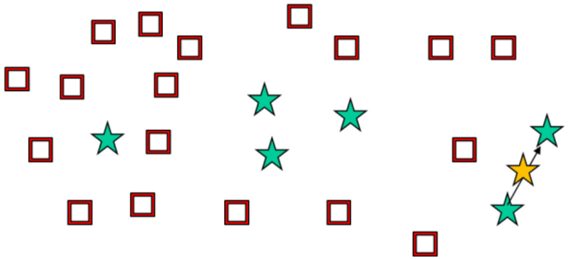
Remove redundant datapoints



Looses statistics!
Good only if enough datapoints on undersampled class and for low dimensional datasets.

Unbalanced Datasets – Over-sampling

Pick neighbour and create new datapoint



Risk overfitting, especially if one does this for points that are noise

Unbalanced Datasets – Down-sampling

Match the samples' number of the classes by down-sampling the most dominant class/classes

1. Partition data into training/testing datasets
2. In the training set, down-sample the most dominant classes:
   - Select randomly samples and remove them
   - Approximate distribution of a class with, e.g. GMM, and select samples in densest regions, i.e. regions with highest likelihood
3. Train the classifier with the down-sampled dataset.
4. Evaluate on the testing set
5. Repeat n times (n-fold validation) Drawbacks:

   Ending up with very few samples for training

- Downsampled dataset does not capture reliably the distribution of
- dominant classes.

Unbalanced Datasets – Oversampling

Create artificial samples of the non-dominant classes

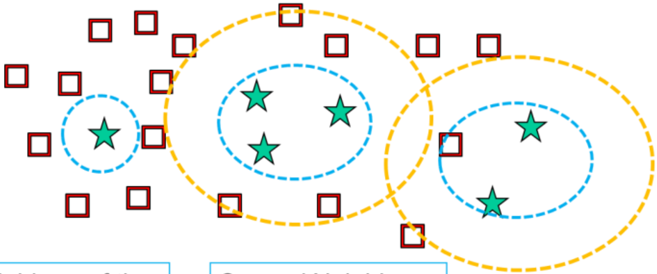
1. Partition data into training/testing dataset
2. In the training set, over-sample the non-dominant classes:
   - Approximate the distribution of a class, e.g. with GMM
   - Sample the required amount of additional data from the GMM
3. Train the classifier with the over-sampled dataset.
4. Evaluate on the testing set
5. Repeat n times (n-fold validation)

Drawbacks:

- Computational complexity

Unbalanced Datasets – Synthetic Minority Oversampling Technique

Determines criteria to sample new points → Generate new samples inbetween existing datapoints **based on their local density and their borders with the other class**. Use cleaning techniques (undersampling) to remove redundancy in the end.



No Neighbors of the same class → **noise**

Several Neighbors of the same class

Surrounded by the other class → **in danger**

Surrounded only on one side by the other class → **safe**

Unbalanced Datasets – Modify Cost Function

- SVM with asymmetric misclassification cost

  → Vary the penalty placed on each datapoint depending on its class

$$\text{minimize } \frac{1}{2}\|w\|^2 + \frac{C}{M}\sum_{i=1}^{M}\beta_i\left(\xi_i + \xi_i^*\right)$$

- SVM class boundary adjustement

  → Changing *b* changes the boundary between the two classes

$$y = \operatorname{sgn}\left(\sum_{i=1}^{M}\alpha_i k\left(x^i, x\right) + b\right)$$

Incremental Learning: Motivation

- In ML, you will start by gathering a dataset, which you will split into training, validation and testing sets.
- You will evaluate the performance of your trained algorithm through crossvalidation, and use values obtained on test set to give you a sense of its expected performance once deployed.
- Yet, this does not guarantee that your algorithm will perform as well as it did on test set, once deployed.
- Indeed, It is rare that you can gather a dataset that will be representative of all you would ever see when deploying the algorithm. The reasons are multi-fold:
- Gathering a dataset representative of all you could ever see would lead to a massive amount of data (possible only if you have the computing resources).
- Gathering a dataset representative of all you could ever see is possible only if what you try to represent will not change in the future.

Incremental Learning: Motivation

Examples of phenomenon that are not static in time:

❖ Face appearance will change with age, haircut, beard shape, etc.

❖ Voice tonality will change with age, accents may appear/disappear (living abroad, living with someone with a foreign accent)

➢ When the evolution is known, one can try to model it, e.g. aging faces / voices.

➢ Yet this model will be imprecise and will add uncertainty to the accuracy of the dataset and model's prediction.

Incremental Learning: Principle

An alternative is to update the model as new data arise.
This is known as *incremental learning.*

Main problems to resolve with incremental learning are:

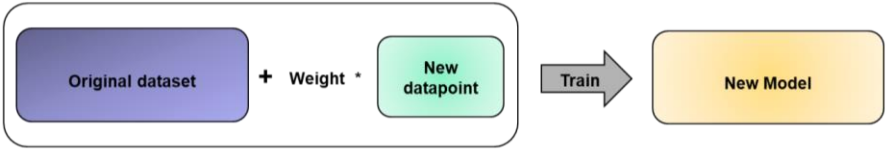
- When to decide to incorporate new knowledge?
- How to preserve old knowledge as we incorporate new knowledge.

Incremental Learning: Principle

Two main approaches to perform incremental learning:

1: Keep both old and new dataset and retrain the model with both

- ➢ New dataset is often smaller than old dataset.
- ➢ How much influence to give to new datapoints?
- ➢ This is costly: requires large storage capacity and large computational cost to retrain the model with the entire dataset.



E.g. choose a weight to give x times more influence to new datapoints:

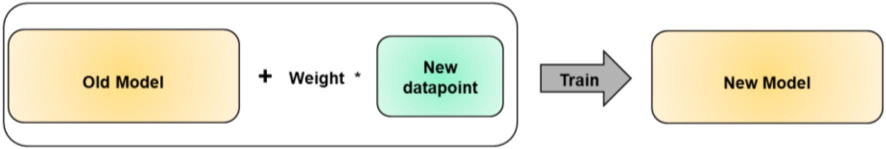
Weight = x * (# new datapoints / # datapoints in original dataset)

Incremental Learning: Principle

Two main approaches to perform incremental learning:

2: Discard old dataset after training to reduce storage costs

- ➢ Update the model based on new datapoints only
- ➢ How to preserve old knowledge stored in the model's dataset
- ➢ How much influence to give to new datapoints?



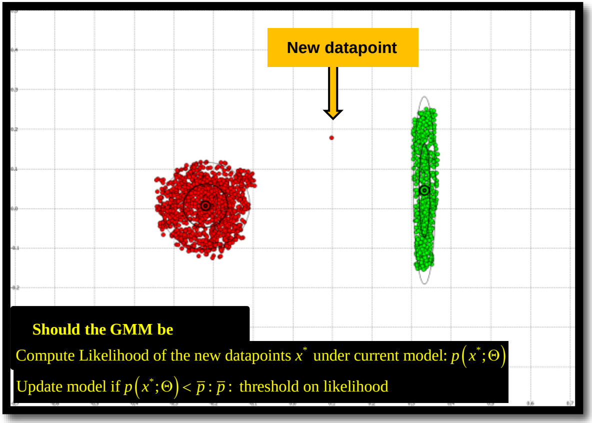
E.g. choose a weight to give x times more influence to new datapoints:

Weight = x * (# new datapoints / # datapoints in original dataset)

Incremental Learning: Example

Update GMM to incorporate new datapoints.

Start with a known model: $p(x;\Theta)=\sum_{k=1}^{K}\alpha_k p_k\left(x;\mu^k,\Sigma^k\right)$ with $\Theta=\{\mu^1,\Sigma^1,...\mu^K,\Sigma^K\}$, $\alpha_k\in[0,1]$.



Should the GMM be

Compute Likelihood of the new datapoints x^* under current model: $p\left(x^*;\Theta\right)$

Update model if $p\left(x^*;\Theta\right)<\bar{p}$: $\bar{p}$: threshold on likelihood

Incremental Learning: Example

Update GMM with incremental E-M

Calls the update step (M-Step) in E-M updates the means μ^k, priors α_k and covariances matrices Σ^k for each of the k Gauss functions:

$$\mu^{k(t+1)} = \frac{\sum_j p\left(k \mid x^j, \Theta^{(t)}\right) \cdot x^j}{\sum_j p\left(k \mid x^j, \Theta^{(t)}\right)}$$

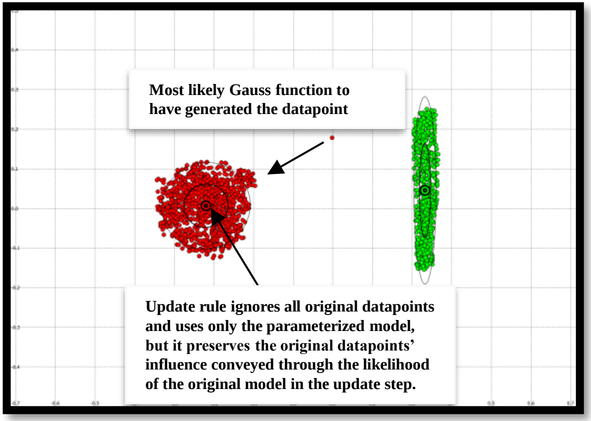
$$\alpha_k^{(t+1)} = \frac{1}{M} \sum_j p\left(k \mid x^j, \Theta^{(t)}\right)$$

$$\Sigma^{k(t+1)} = \frac{\sum_j p\left(k \mid x^j, \Theta^{(t)}\right)\left(x^j - \mu^{k(t+1)}\right)\left(x^j - \mu^{k(t+1)}\right)^T}{\sum_i p\left(k \mid x^i, \Theta^{(t)}\right)}$$

This can be sped up by updating solely the Gauss function that is most likely to have generated the new datapoints, i.e. Gauss function solely to:

$$k^* = \underset{k}{\operatorname{argmax}}\ p\left(k \mid x^*, \Theta^{(t)}\right)$$

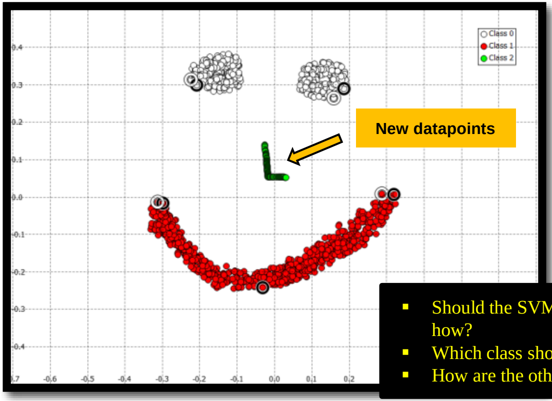
Incremental Learning: Example



Relative importance of datapoint measured through increment in likelihood after update of mean and covariance of the k* Gauss function:

$$\alpha_{k^*}^{(t+1)} = \left(1+\beta_{k^*}\right) p\left(k^* \mid x^*, \Theta^{(t)}\right) \quad \text{and} \quad \beta_{k^*} = \frac{p_{k^*}\left(x^*; \mu^{k(t+1)}, \Sigma^{k(t+1)}\right) - p_{k^*}\left(x^*; \mu^{k}, \Sigma^{k}\right)}{p_{k^*}\left(x^*; \mu^{k(t+1)}, \Sigma^{k(t+1)}\right)}$$

Incremental Learning: Example

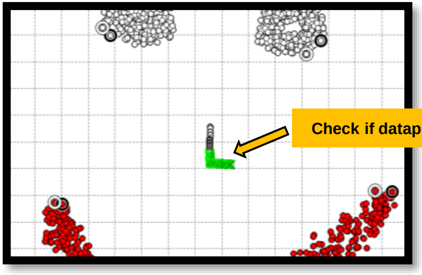


- Should the SVM be updated and how?
- Which class should be modified?
- How are the other classes affected?

The model learned by Support Vector Machine is parameterized by the Support Vectors,

Support Vectors are all points x^i such that $\alpha_i \neq 0$ in the decision function: $f(x) = sgn\left(\sum_{i=1}^{M} \alpha_i y_i k(x, x^i) + b\right)$

Incremental Learning: Example



Check if datapoints violate the constraints

If datapoints violate the constraints, opt for an incremental update SVM optimization (Shilton et al. 2005)

Shilton, Alistair, et al. "Incremental training of support vector machines." *IEEE transactions on neural networks* 16.1 (2005): 114-131.

Summary

- Missing values:
  - Replace with mean (continous) or most frequent value (categorical) Approximate
  - regression/classification

    Best estimate through Expectation-Maximization

- Unbalanced datasets:

  Importance of performance metric

  Rebalance Datasets: Undersampling or Oversampling
  - Adapt cost function

- **Incremental learning**
  - Assess if current model is sufficient
  - Update model with incremental update