COMP309 in Week 07, 2024

missing values

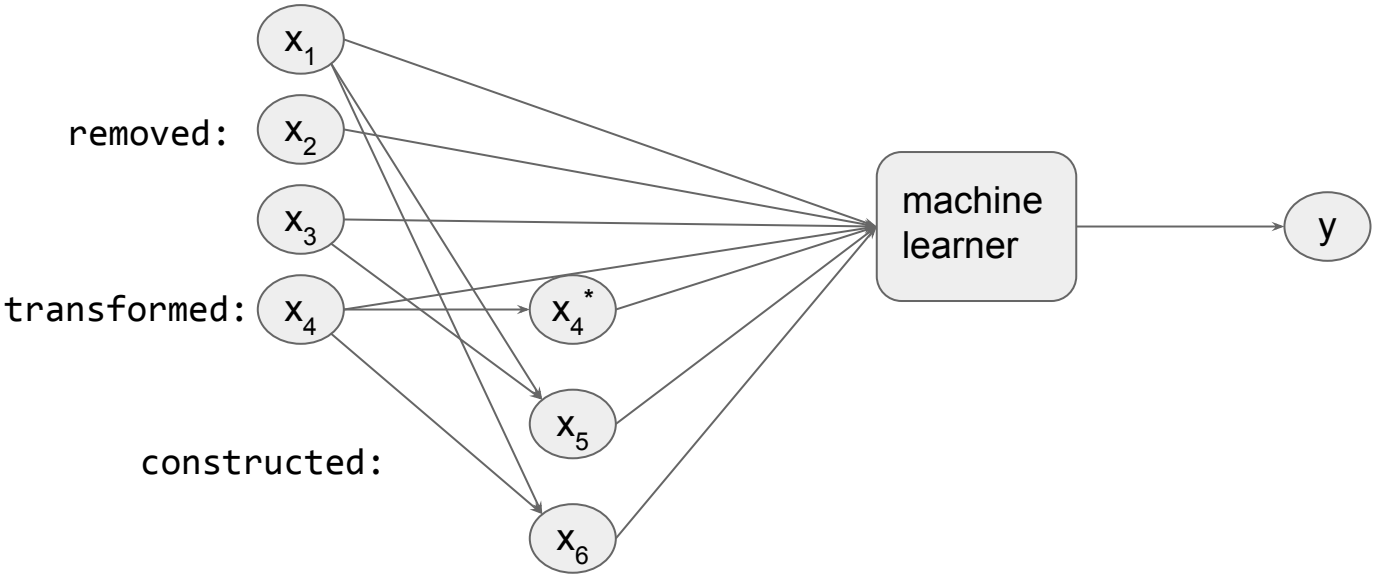
Marcus Frean marcus@ecs.vuw.ac.nz

but first: finishing off previous lecture

L1: feature engineering

L2: dealing with missing data

changes to features



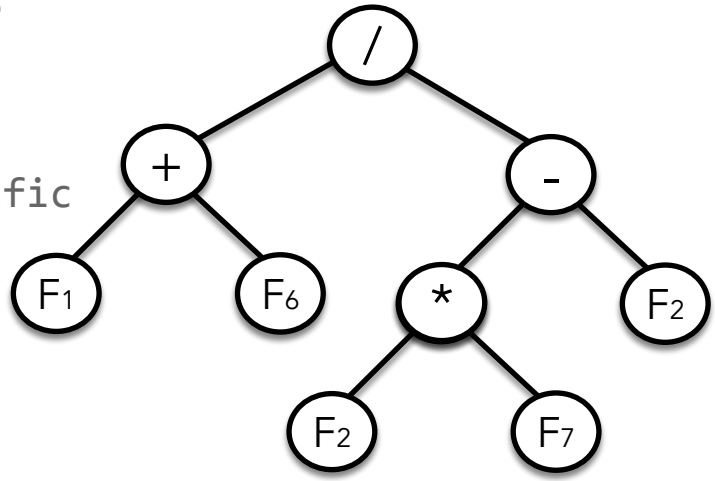
construct new features automatically with Genetic Programming?

Genetic Programming is a flexible way to make mathematical and logical functions

but warning: there isn't much structural (topological) information in the search space of possible functions

→ yes it's possible,
but be prepared to spend a lot of compute time trying out dud trees…

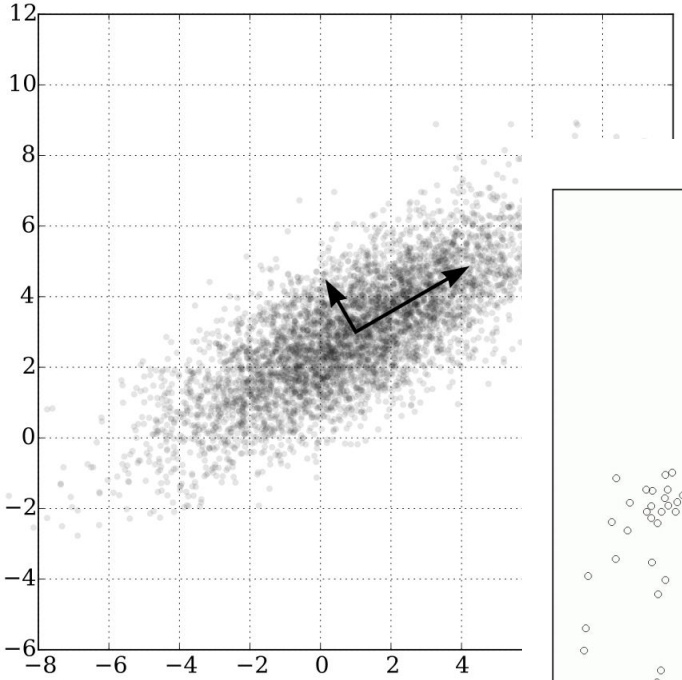
→ Also: just evolving good answers to specific problems is not a recipe for insight. Insight comes from having theories about the world (a.k.a. "a model") and testing it.

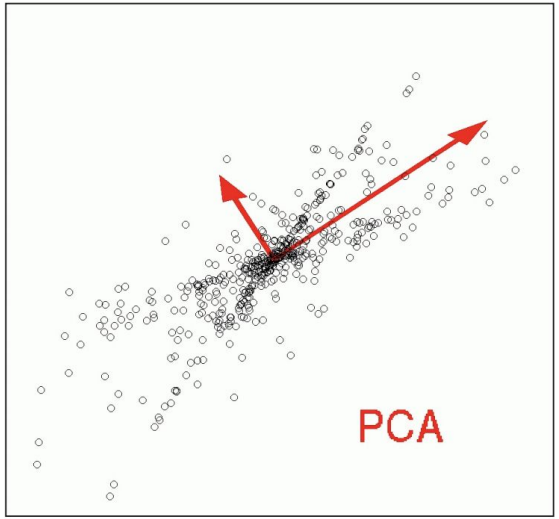


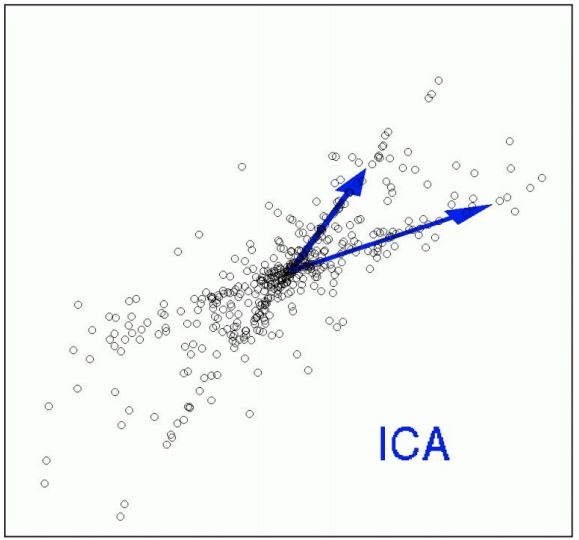
Some specific “many to many” feature methods:

- PCA
- ICA
- Kernel PCA
- autoencocers

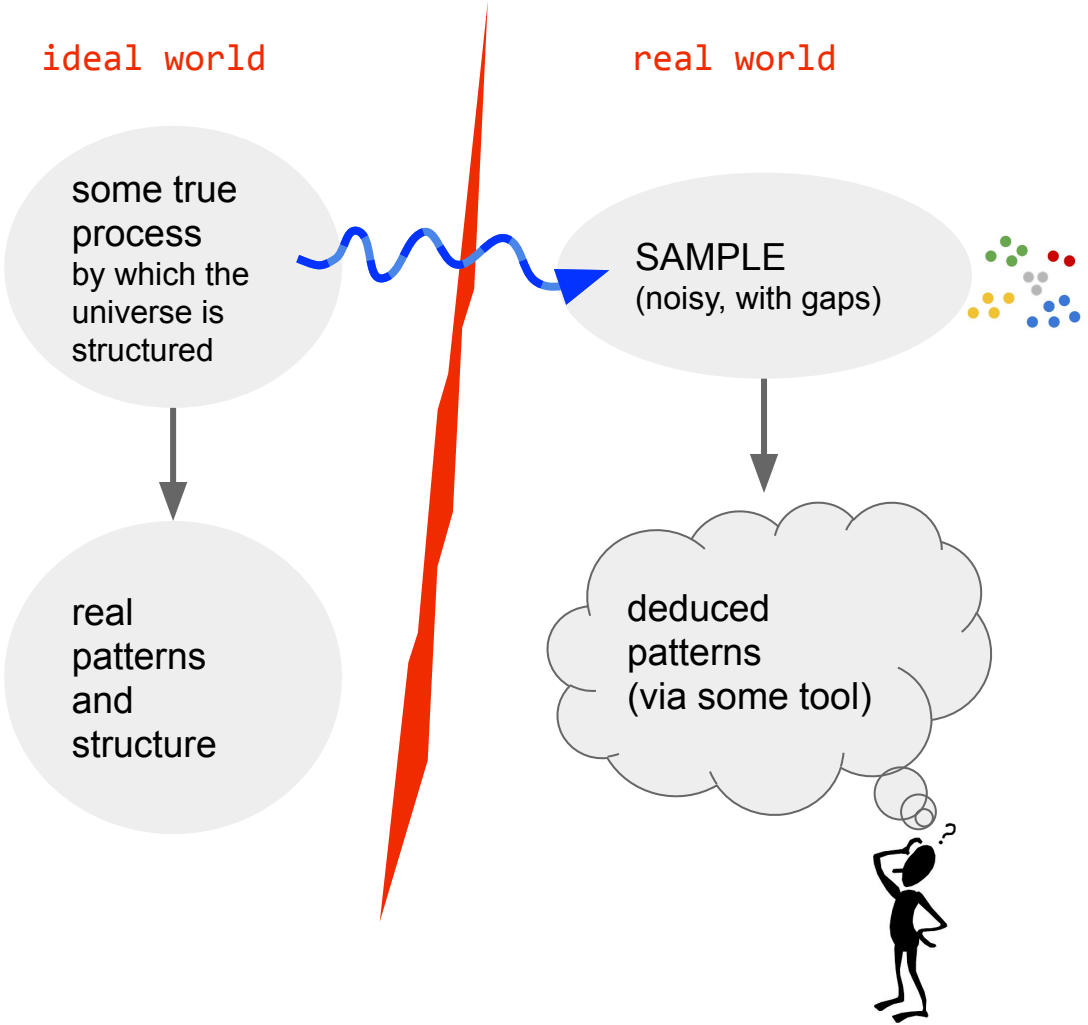
PCA (Principal Components Analysis)







the problem of finding true patterns from sample data



Missing values

many possible reasons, e.g:

➔ high cost involved in measuring variables,
➔ failure of sensors,
➔ reluctance of respondents to answer some q,
➔ an ill-designed survey

1. Real-world datasets often contain many missing values!
2. Most machine learning algorithms *require* all the values, for a given training example.

You can see the problem!

4 solutions:

- Remove the missing data instances from the dataset, or
- Remove the worst offending attributes from the dataset, or
- Use imputation methods to substitute in “fake” values, or
- Use a learner that is able to cope with missing values

Truly

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

ideal world

real world

Recorded

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	?	No
2	?	?	100K	No
3	No	Single	70K	No
4	Yes	Married	?	No
5	No	?	95K	Yes
6	?	Married	60K	No
7	Yes	Divorced	220K	No
8	No	?	?	Yes
9	?	Married	75K	No
10	No	Single	90K	Yes

- Missing data can also be in the form of features, or labels (just one class, or…?) or both

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	?	?	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	?	95K	Yes
6	?	Married	60K	No
7	Yes	Divorced	220K	No
8	No	?	85K	Yes
9	?	Married	75K	No
10	No	Single	90K	Yes

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	?
2	No	Married	100K	?
3	No	Single	70K	No
4	Yes	Married	120K	?
5	No	Divorced	95K	Yes
6	No	Married	60K	?
7	Yes	Divorced	220K	No
8	No	Single	85K	?
9	No	Married	75K	No
10	No	Single	90K	Yes

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	?	No
2	?	?	100K	?
3	No	Single	?	No
4	Yes	Married	120K	?
5	No	?	95K	Yes
6	?	Married	60K	?
7	Yes	Divorced	?	No
8	No	Single	85K	?
9	No	?	75K	No
10	?	Single	?	Yes

Option 1: just eliminate entire observations

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	?	Married	?	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	?	Divorced	?	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	?	?	75K	No
10	No	Single	90K	Yes

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
10	No	Single	90K	Yes

This seems extreme! It wastes lots of precious, expensive, data.

But let's just say we do this. **Is it even "okay" ?**

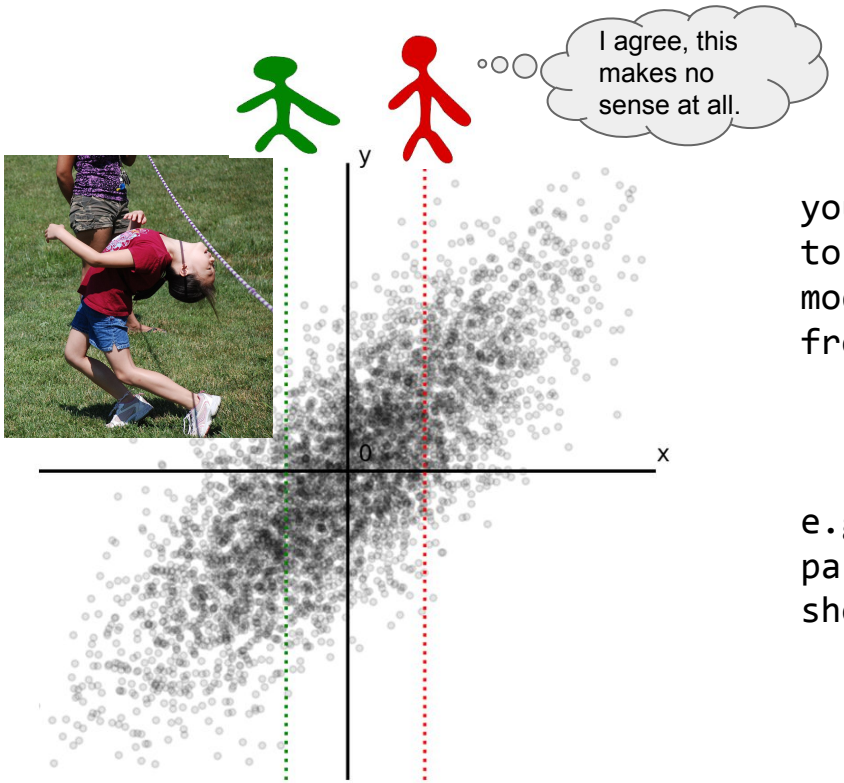
(e.g. suppose we knew *high values of* income were more likely to be missing…)

there are different kinds of missing!

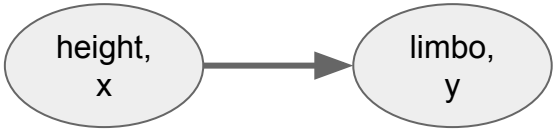
- MCAR missing, completely at random
- MAR missing at random
- MNAR missing, not at random

a toy example

x is height, y is ability to limbo (relative to their average values)

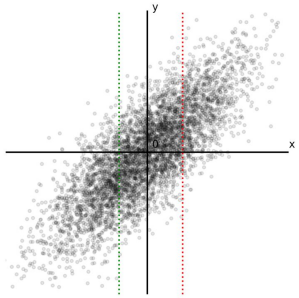


you're using this data to train a regression model to predict y, from x



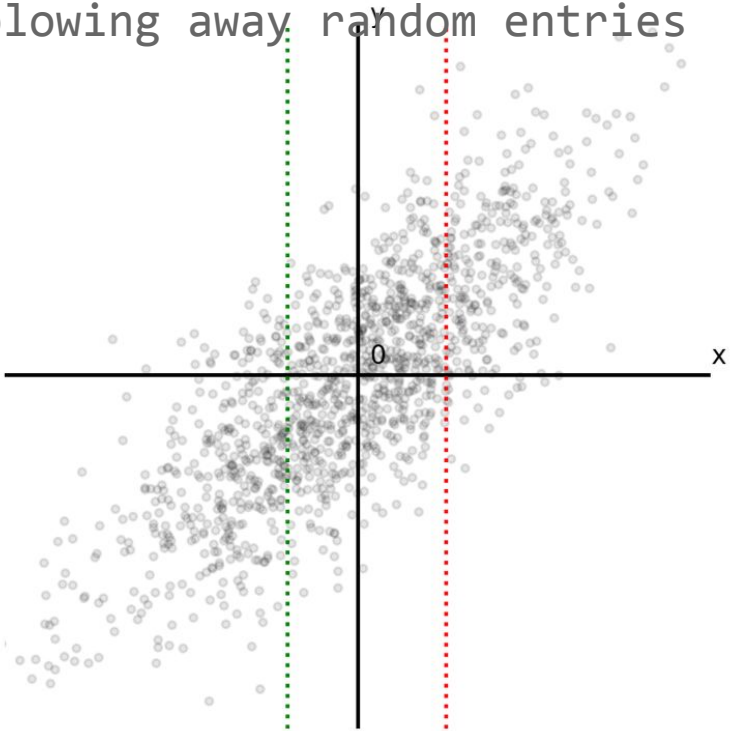
e.g. consider two particular cases, shown as ..... and .....

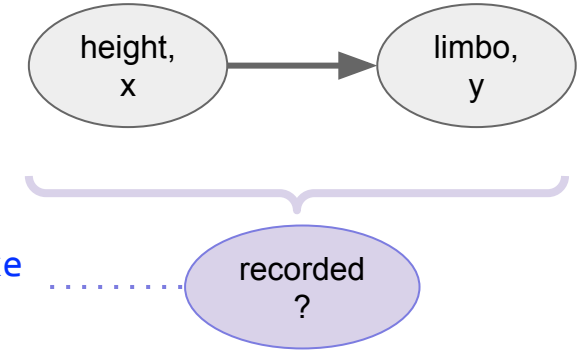
Missing Completely At Random (MCAR)



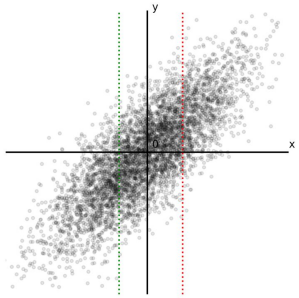
E.g.

I blasted my spreadsheet with a shotgun, blowing away random entries

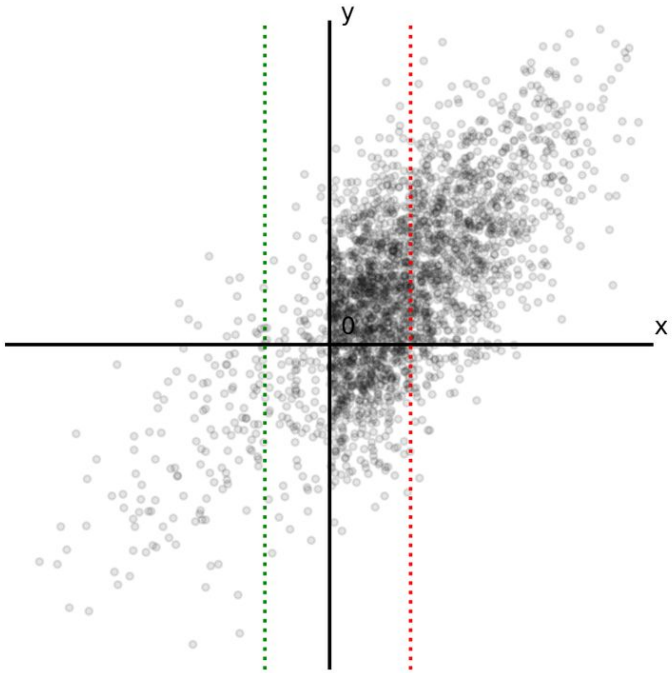


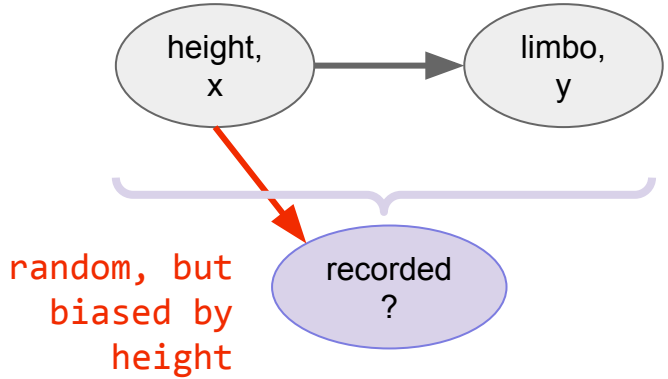


Missing At Random (MAR)

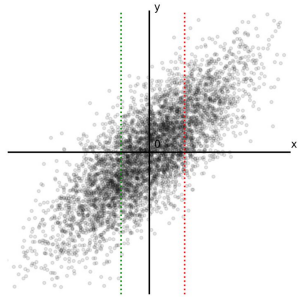


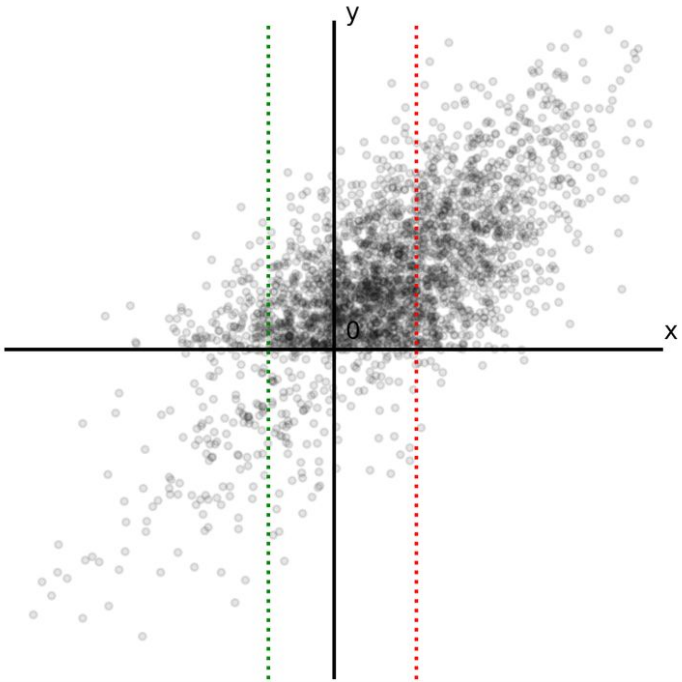
note this name is really confusing. A better name for MAR would have been be “Missing, conditionally at random”, but that would screw with the acronyms 🙄

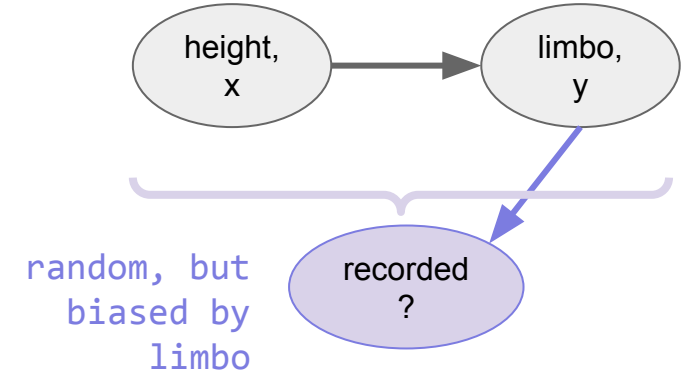




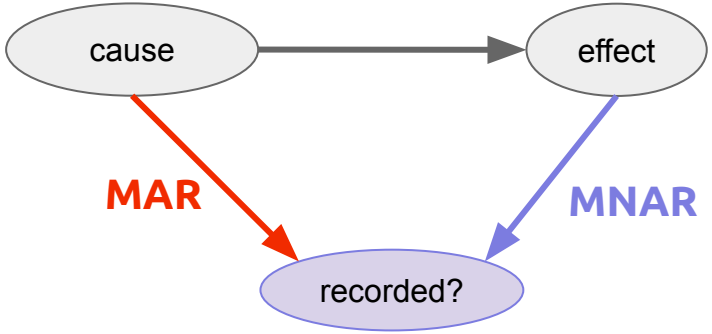
Missing Not At Random (MNAR)





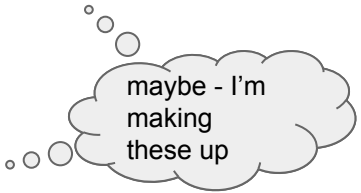


Q. would you say these (below) are cases of MCAR, MAR, or MNAR?



c.f. **MCAR** has *no* linkage

1. Half the survey forms weren't delivered or got lost, due to random beetle infestations
2. Males are less likely to fill in a depression survey, but this has nothing to do with their level of depression, after accounting for maleness
3. Depressed people are less likely to fill in a survey about depression
   (similarly, happy people are less likely to fill in a grievance survey)



conclusions?

So: simply avoiding data that has missing values isn't ideal

- reason 1?
  - wastes valuable data - what if *every* row had something missing?!
- reason 2?
  - could be MNAR, in which case results will be "biased" (ie. wrong)

but what else can we do?

We could "impute" data – replace missing with actual values.
But what values?!

Option 2: eliminate entire features

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	?	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	?	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	?	Married	75K	No
10	No	Single	90K	Yes

Tid		Marital Status	Taxable Income	Cheat
1		Single	125K	No
2		Married	100K	No
3		Single	70K	No
4		Married	120K	No
5		Divorced	95K	Yes
6		Married	60K	No
7		Divorced	220K	No
8		Single	85K	Yes
9		Married	75K	No
10		Single	90K	Yes

Seems a bit drastic

Option 3: Imputation

- Mean imputation
- Mode imputation
- Hot deck imputation
- other ideas…
- Multiple imputation
- MICE (used in sklearn)

impute with Mean, or Mode

- Mean: for continuous attributes
  - Fill in with average complete values

- Mode: for categorical attributes
  - Fill in with the most frequent value

Pros and cons:

(+) Doesn't changes the mean/mode of attributes

(-) Under-represents the variability in the data

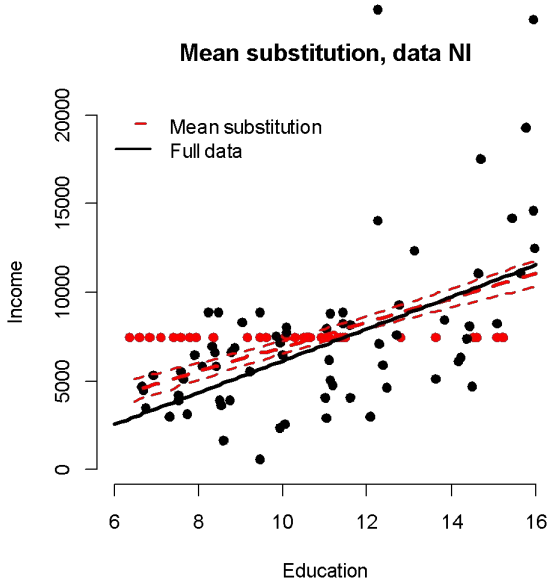
pretty bad imputation: Mean or Mode

Tid	Refund	Marital Status	Taxable Income	Cheat
1	?	Single	125K	No
2	No	?	?	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	?	Yes
6	No	?	60K	No
7	Yes	Divorced	220K	No
8	?	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes
	No	**Single**	**105K**	

most common/
mean value

Tid	Refund	Marital Status	Taxable Income	Cheat
1	**No**	Single	125K	No
2	No	**Single**	**105K**	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	**105K**	Yes
6	No	**Single**	60K	No
7	Yes	Divorced	220K	No
8	**No**	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

sklearn's SimpleImputer has options



“hot deck” imputation

For each record (row) containing missing values:
Find the most similar record, then fill missing values with the corresponding values from that!

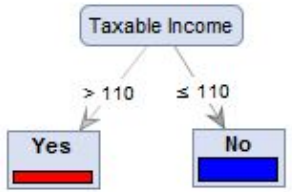
Pros and cons

(+) Imputes “realistic” values

(-) what do you think?

This is just “Nearest Neighbour” interpolation, right?
So we could use ***k***-NN instead.

Imputation using a decision rule?



Tid	Refund	Marital Status	Taxable Income	Cheat
1	?	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	?	Divorced	95K	Yes
6	?	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Imputation using Regression?

For each attribute containing missing values:

- Divide data into two parts:

  the records that are complete

  the records having missing values

- Use the complete data to estimate a regression model between the attribute and others, and use that to predict/suggest missing values in the attribute

  Pros and cons

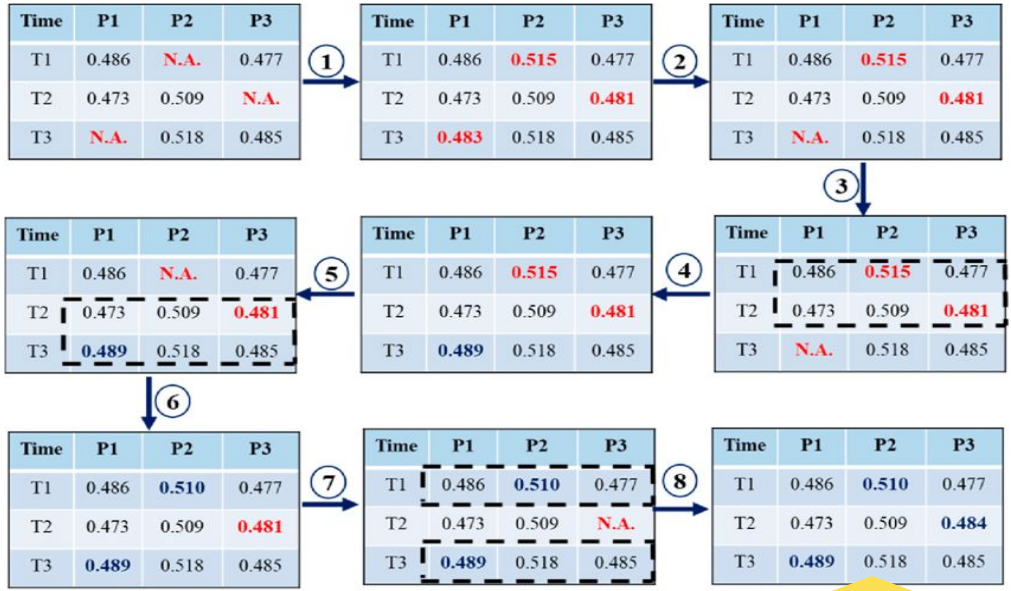
  (+) Maintains relationships between attributes

  (-) Makes an assumption - the regression model

MICE ("Multivariate Imputation by Chained Equations" 😕)

a version of multiple imputation that apparently works quite well and is implemented in sklearn:

1. Fill all missing values with random values (?!?)
2. Regress each attribute that contains missing values on other attributes, in a chain => 1 imputed dataset



sklearn's IterativeImputer does this, which is steps 1 and 2

3. Repeat whole procedure N times to generate N imputed datasets
4. Average N imputed datasets => final imputed dataset

https://www.researchgate.net/figure/Workflow-of-multiple-imputation-by-chained-equations-MICE_fig5_350620190

Option 4: use a learning algorithm that can cope with missing values

- Missing values can sometimes be taken into account during the learning process of acquiring knowledge

- For example:
  - Clustering algorithms: similarity between the objects can calculated using only the attributes that do not have missing values.
  - Classification: Gradient Boosting, kNN, and Random Forests

scikit-learn resources

`https://scikit-learn.org/stable/modules/impute.html`

6.4. Imputation of missing values

For various reasons, many real world datasets contain missing values, often encoded as blanks, NaNs or other placeholders. Such datasets however are incompatible with scikit-learn estimators which assume that all values in an array are numerical, and that all have and hold meaning. A basic strategy to use incomplete datasets is to discard entire rows and/or columns containing missing values. However, this comes at the price of losing data which may be valuable (even though incomplete). A better strategy is to impute the missing values, i.e., to infer them from the known part of the data. See the glossary entry on imputation.

6.4.1. Univariate vs. Multivariate Imputation

One type of imputation algorithm is univariate, which imputes values in the i-th feature dimension using only non-missing values in that feature dimension (e.g. `impute.SimpleImputer`). By contrast, multivariate imputation algorithms use the entire set of available feature dimensions to estimate the missing values (e.g. `impute.IterativeImputer`).