



ISSUES AND CHALLENGES IN ML ANALYSIS

Tengku Muhammad Hanis Bin Tengku Mokhtar

February 12, 2026

About me



Senior Lecturer
at **UniSZA**

Background

- PhD (Public Health Epidemiology) from USM, 2024
- MSc (Medical Statistics) from USM, 2019
- MBChB from Al-Azhar University, 2015

Interest:

- Medical statistics, meta-analysis, bibliometrics
- Machine learning and deep learning application in medical sciences
- Application of R on health/medical data

Contact me:

- Email: tengkuhanismokhtar@gmail.com
tengkuhanismokhtar@unisza.edu.my
- Website: <https://tengkuhanis.netlify.app/>
<https://jomresearch.netlify.app/>

Material

https://github.com/tengku-hanis/applied_ml_seberangjaya.git

Content

- [Data size](#)
- [Data leakage](#)
- [Curse of dimensionality](#)
- [Missing data](#)
- [Sparse category](#)
- [Scaling](#)
- [Interpretable ML](#)
- [Deployment](#)

Data size

- Generally the bigger the data, the better the model perform
- There are a few sample size calculation available such as using [AUC](#)
- However, these methods mostly not appropriate for model training and development - do not take into account the complexity of the algorithms, number of features, etc (personal opinion)
- The idea of sample size in statistical analysis is to have enough power to detect the significant difference - for example, in AUC to detect the difference between AUC of 0.5 (random guessing) and the hypothesized AUC
- These methods are appropriate:
 - As a rough guideline
 - Validating a pre-trained model

Data leakage

- Data leakage occurs when information outside of the training dataset is used to train the ML model
- Example:
 - Train-test contamination: mean imputation use the whole dataset instead of the training dataset
 - Target leakage:
 - Include predictors that are not available at the time of prediction
 - Include the diagnostic information when the model supposed to be a screening model

Curse of dimensionality

- As the number of features (dimensions) increases, the volume of the space grows exponentially, making data points sparse
- This sparsity
 - Makes it difficult for algorithms to find meaningful patterns
 - Increase computational costs
 - Reducing overall performance - models are more likely to overfit because they capture noise instead of the underlying pattern in the data
- Solution:
 - Feature selection
 - Use dimension reduction techniques - PCA, UMAP, t-SNE, etc

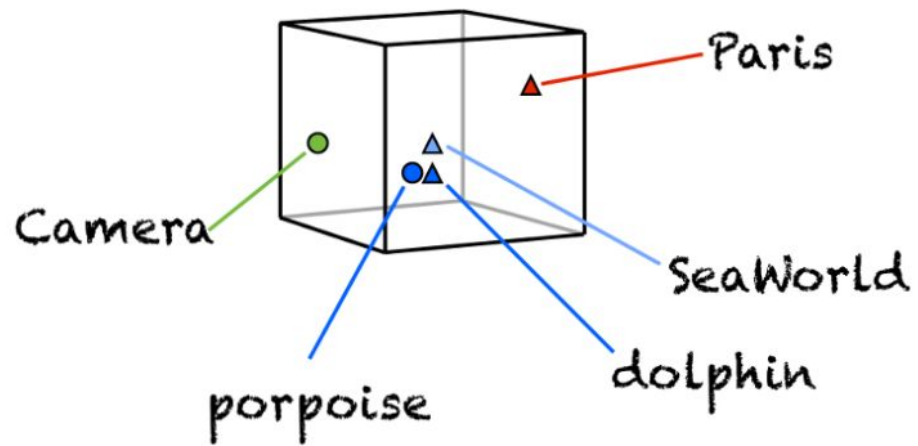
Missing data

- Data should be representative of the prediction task
- Missing data may compromise the data representativeness
- There are 3 types of missing data mechanisms:
 - Missing completely at random (MCAR):
 - Data missing due to chance
 - Eg: some of the medical records lost due to flooding
 - Missing at random (MAR):
 - Data missing related to the observed variable
 - The missingness can be explained by other variable
 - Eg: blood pressure info is missing due to a very big biceps (the clinic only has small to medium cuff)

- Missing not at random (MNAR);
 - Data missing due to unobserved variable
 - Eg: salary info is missing for people with high income as they feel insecure to disclose their salary
- Basically, only MCAR does not affect the data representativeness
- The most effective solution is to take extra measures to avoid missing data (especially MNAR type)
- The ML field is not well developed in this issue to resolve it compared to the conventional statistical analysis field:
 - ML field is more algorithmic or applied approach rather than theoretical one and more focus on prediction rather than inference

Sparse category

- Also known as high cardinality or rare category problem
- The effect of having a sparse category:
 - May lead to overfitting - the ML model might memorise the rare category as there is not enough data to learn from
 - Dimensionality explosion and slow model training process - if we apply one-hot encoding and we have many sparse categories
 - The “unseen” problem - if a sparse category appear only in the test dataset
- Solution:
 - Grouping/binning - combine all sparse category into one group, “others”
 - Embeddings - map categorical variables into a new dimensional space



Scaling

- Two most common methods:
 - Min-max normalisation

$$x' = \frac{x - \mu}{\max(x) - \min(x)}$$

- Z-score normalisation (standardisation)

$$z = \frac{x - \mu}{\sigma}$$

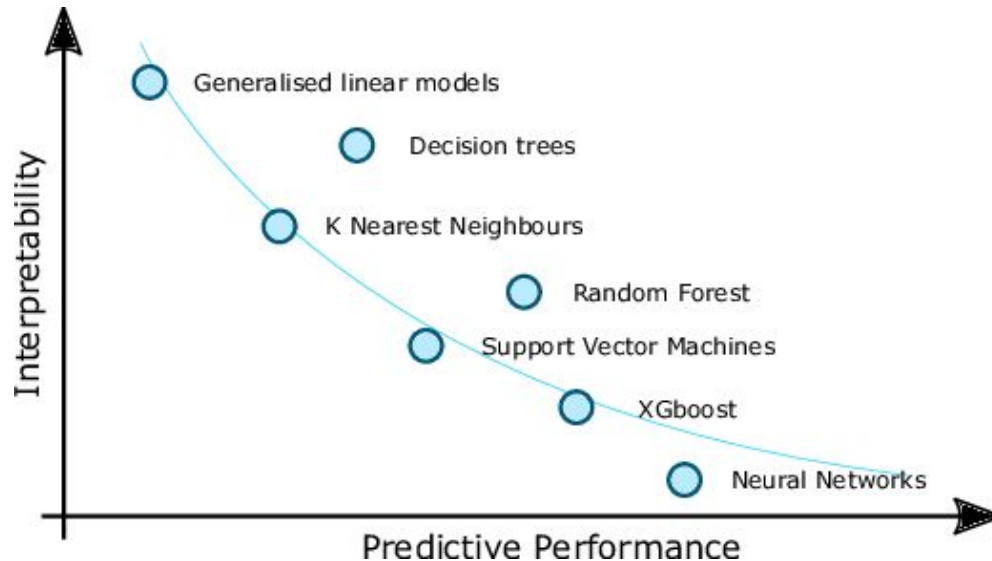
μ = Mean

σ = Standard Deviation

- Always scale after splitting the data
- Scaling may help with faster convergence and model performance
- However, not all model benefit from the scaling process:
 - Does not benefit - tree-based models (decision tree, random forest, XGBoost, LightGBM)
 - Benefit - distance-based models (k-NN, k-means, SVM), gradient descent-based model (linear/logistic regression, neural network)
 - More info - [refer to tidymodels book](#)

Interpretable ML

- Also known as explainable machine learning
- There is always an inverse relationship between model complexity and interpretability



- There are 2 types of ML model:
 - Glass/white box model:
 - ML models that we can understand the structure and their internal working
 - Decision tree, linear regression, etc
 - Black box model:
 - ML models that are hard/difficult to understand or follow how it reach the prediction
 - Random forest, XGBoost, deep learning, etc

- Some differentiate the terms:
 - Interpretable:
 - A model is considered interpretable if we can understand the internal structure of the model (ie: glass box model)
 - Explainable:
 - A model is explainable when the model itself is “black box”
- There are 2 types:
 - Model agnostic approach:
 - An approach that is universal and can be applied to ML models
 - Model specific approach:
 - An approach that is specific to certain models

- Can also classified as:
 - Global explainer:
 - Explain the behaviour of the model as a whole
 - Individual explainer
 - Explain the behaviour of the model at the specific instances
- Some of the packages that work well with tidymodels ecosystem:
 - [lime](#)
 - [vip](#)
 - [Dalex and DALEXtra](#) (personal recommendation)
 - [shapper](#)

- Example of interpretable ML approaches:

Model agnostic + global explainer	Model specific + global explainer
<ul style="list-style-type: none"> - Permutation-based variable importance 	<ul style="list-style-type: none"> - Tree-based variable importance
Model agnostic + individual explainer	Model specific + individual explainer
<ul style="list-style-type: none"> - LIME - Local Interpretable Model-agnostic Explanations - SHAP - Shapley Additive Explanation 	<ul style="list-style-type: none"> - LRP - Layer-wise Relevance Propagation (for DNN)

Deployment

- There are 2 main approaches (that I know of):
 - Plumber (API) and vetiver (version control)
 - Turn R-based ML model into a URL that other program can send data and get prediction
 - Shiny app - [example](#)

Suggested readings/references

- Kuhn, M., & Silge, J. (2022). [Tidy modeling with R: A framework for modeling in the tidyverse](#). " O'Reilly Media, Inc."
- Christoph, M. (2020). [Interpretable machine learning: A guide for making black box models explainable](#).
- Biecek, P., & Burzykowski, T. (2021). [Explanatory model analysis: explore, explain, and examine predictive models](#). Chapman and Hall/CRC.



Any question?



<https://tengkuhanis.netlify.app/>
<https://jomresearch.netlify.app/>