

# 6. Making your work public: Considerations of more general use and prominent failures

# REFORMS checklist

To ensure reproducibility, include a checklist such as the REFORMS checklist into the publication of your results.

There are a couple of domain-specific checklists, especially in health and life sciences (STARD, CLAIM, see the EQUATOR<sup>\*</sup> network), that may be more appropriate - REFORMS aims to be most general.

REFORMS checklist: DOI 10.48550/arXiv.2308.07832, <https://reforms.cs.princeton.edu/>

<sup>\*</sup>EQUATOR Network, <https://www.equator-network.org/reporting-guidelines/>

# REFORMS: Example

## 1. Study goals

### 1a) Population or distribution about which the scientific claim is made

*The group to which the claim will be generalized*

# REFORMS: Example

## 1. Study goals

### 1b) Motivation for choosing this population or distribution

*Choice of a particular population of interest - pure scientific interest, need for applied knowledge, ...*

# REFORMS: Example

## 1. Study goals

### 1c) Motivation for the use of ML methods in the study

*Why focus is on building a model that reliably maps input data to output data, instead of using traditional statistical methods?*

# REFORMS: Example

## 2. Computational reproducibility

### 2a) Dataset

*Cite datasets with permanent links to clarify which version; if contains sensitive data: release synthetic datasets (synthpop library)*

# REFORMS: Example

## 2. Computational reproducibility

### 2b) Code

*Cite exact version of code (DOI, version number, commit tag)*

# REFORMS: Example

## 2. Computational reproducibility

### 2c) Computing environment

*Details about the hardware (CPU, RAM, disk space), software (operating system, programming language, version number for each package used), and computing resources (time taken to generate the results)*



# REFORMS: Example

## 2. Computational reproducibility

### 2d) Documentation

*Document installation, requirements, running the code, usage examples, expected results*

# REFORMS: Example

## 2. Computational reproducibility

### 2e) Reproduction script

*Reproduction scripts that prepare the environment, install the code, download datasets, and run code to reproduce the results in the paper*

# REFORMS: Example

## 3. Data quality

### 3a) Data source(s)

*When, where, how data were collected, how ground-truth annotations were performed*

# REFORMS: Example

## 3. Data quality

### 3b) Sampling frame

*List of people or units from which a sample is drawn*

# REFORMS: Example

## 3. Data quality

### 3c) Justification for why the dataset is useful for the modeling task

*Why the dataset is a good representative to model the question/answer*

# REFORMS: Example

## 3. Data quality

### 3d) Outcome variable

*How outcome variable is defined - this is usually not a perfect match for what it should represent*

# REFORMS: Example

## 3. Data quality

### 3e) Number of samples in the dataset

*Total sample size, number of samples in each class, number of individual data in the dataset*

# REFORMS: Example

## 3. Data quality

### 3f) Missingness

*Missing data*



# REFORMS: Example

## 3. Data quality

### 3g) Dataset for evaluation is representative

*Justify why data is representative for target selected in 1a)*

# REFORMS: Example

## 4. Data preprocessing

### 4a) Excluded data and rationale

*Justify why particular subset of data was chosen*

# REFORMS: Example

## 4. Data preprocessing

### 4b) How impossible or corrupt samples are dealt with

*How erroneous or impossible data points are identified and amended*

# REFORMS: Example

## 4. Data preprocessing

### 4c) Data transformations

*Normalizing, augmenting, imputing missing data, oversampling - the latter two must be done separately on each fold of the dataset!*

# REFORMS: Example

## 5. Modeling

### 5a) Model description

*Input, output, type of model, loss function, algorithm*

# REFORMS: Example

## 5. Modeling

### 5b) Justification for the choice of model types implemented

*Model needs to be interpretable, types of models considered*

# REFORMS: Example

## 5. Modeling

### 5c) Model evaluation method

*Model needs to be evaluated on test data different from training data*

# REFORMS: Example

## 5. Modeling

### 5d) Model selection method

*How final model(s) and hyperparameters were selected*



# REFORMS: Example

## 5. Modeling

### 5e) Hyperparameter selection

*How hyperparameters were optimized*

# REFORMS: Example

## 5. Modeling

### 5f) Appropriate baselines

*How baseline models were trained and optimized*

# REFORMS: Example

## 6. Data leakage

### 6a) Train-test separation is maintained

*Use a hold-out test set that is also not used in synthetic data generation*

# REFORMS: Example

## 6. Data leakage

### 6b) Dependencies or duplicates between datasets

*Could be more than one sample from same origin (patient); or time series is split randomly*

# REFORMS: Example

## 6. Data leakage

### 6c) Feature legitimacy

*Any feature in the training that somewhat identifies predicted variable*

# REFORMS: Example

## 7. Metrics and uncertainty quantification

### 7a) Performance metrics used

*Proper choice of metric depending on application*

See Leist et al., Sci. Adv. 8, eabk1942 (2022) Table 4 for appropriate metrics depending on the data and algorithm used

# REFORMS: Example

## 7. Metrics and uncertainty quantification

### 7b) Uncertainty estimates

*Randomness in training or evaluation data, or training process*

# REFORMS: Example

## 7. Metrics and uncertainty quantification

### 7c) Appropriate statistical tests

*Statistical testing of ML models*

For an overview, see Sebastian Raschka, Model evaluation, model selection, and algorithm selection in machine learning, arXiv:1811.12808



# REFORMS: Example

## 8. Generalizability and limitations

### 8a) Evidence of external validity

*Report how claim is generalizable to target population.*

# REFORMS: Example

## 8. Generalizability and limitations

### 8b) Contexts in which the study's findings will not hold

*Clear expectations and unjustified hype.*



SCIENTIFIC  
SOFTWARE  
CENTER



UNIVERSITÄT  
HEIDELBERG  
ZUKUNFT  
SEIT 1386

# Software security

# Security

- Threat modelling: who, what, how
- Data-oriented attack: Access training data, poison data, inject trojan data
- Model-oriented attack: Modify training process (pre-trained malicious models), manipulate the deployed model (model patches, privacy information leakage, model inversion attacks)
- System-oriented attack: specialised hardware accelerators for ML software (SOC, trojan in GPU/TPU, for model corruption, backdoor insertion, model extraction, spoofing, information extraction, sybil attack)
- Possibility to carry out *pentests*

# Security: best practices

Phases	Vulnerabilities Causes
Analysis phase	No risk analysis/ No security policy
	Biased risk analysis
	Unanticipated risks
Design phase	Relying on non-secure abstractions
	Security/Convenience tradeoff
	No logging
	Design does not capture all risks
Implementation phase	Insufficiently defensive input checking
	Non-atomic check and use
	Access validation errors
	Incorrect crypto primitive implementation
	Insecure handling of exceptional conditions
	Bugs in security logic
Deployment phase	Reuse in more hostile environments
	Complex or unnecessary configuration
	Insecure defaults
Maintenance phase	Feature interaction
	Insecure fallback

Chen, Barbar, Security for Machine Learning-based Software Systems, DOI 10.1145/3638531



SCIENTIFIC  
SOFTWARE  
CENTER



UNIVERSITÄT  
HEIDELBERG  
ZUKUNFT  
SEIT 1386

# Legal aspects

# Legal aspects

- If you reuse data / models / code: Make sure the license terms allow this and that your license(s) is (are) compatible
- Make sure you do not violate the DSGVO / GDPR / European Data Act / Copyright / European AI Act
- Once a model is made available, it is impossible to restrict its use!
- Examples:
  - Models can put out near-exact copies of images/text in training data, ie Dall-E/Stable Diffusion generating images with Shutterstock/Getty Images watermarks, or reproducing artist's work
  - Code-generating tools such as GitHub copilot allow recreation of code, that is already contained in other software, regardless of the license terms of that software



SCIENTIFIC  
SOFTWARE  
CENTER



UNIVERSITÄT  
HEIDELBERG  
ZUKUNFT  
SEIT 1386

# Ethical aspects



# Ethical aspects

- Be aware of people's tendency of overreliance!
  - ELIZA effect: the tendency of users to project human traits onto interactive software
  - Trusting into predictions above one's own assessment
- Misuse of AI by bad actors
  - Face recognition used to detect Uyghur population (China), Clearview used to track people's movement and employment status by police officers (even though use was prohibited)
  - Facial analysis used to track people's attention on billboards, change advertisement based on their demographic
- Source of truth during training
  - Data that foundation models are trained on does also contain false statements, ie law professor incorrectly accused of sexual harassment
  - Data in foundation models contains toxicity of the internet; human labor is used to label / annotate toxic text and images ie subcontracting workers in Kenya (OpenAI)



# Infamous AI mistakes

# AI in general

- **Automatization at amazon:** Experimental hiring tool, developed by a team of five, used artificial intelligence to give job candidates scores ranging from one to five stars
- Tool was trained on all resumes of the last 10 years
- Tool preferably suggested male candidates, penalizing resumes that contained the word “women’s” or graduates from all-female colleges
- ***Why?***

# Automated resume selection at amazon

- The dataset consisted of the resumes of the last 10 years: Predominantly male applicants
- Women are underrepresented in tech:

<https://fingfx.thomsonreuters.com/gfx/rngs/AMAZON.COM-JOBS-AUTOMATION/010080Q91F6/index.html>

- Thus, the model learnt it was more often correct if it suggested male candidates: Unbalanced representation of the dataset

# AI in general

- U.S. healthcare system uses commercial algorithms to guide health decisions
- Algorithm (Optum's Impact Pro) to target patients for “high-risk care management” programs
- Identify patients who benefit the most: <https://www.science.org/doi/10.1126/science.aax2342>
- Model is trained on healthcare spendings to determine the healthcare need
- Algorithm was much more likely to recommend white patients for these programs than black patients, even though the black patients were evidently sicker
- ***Why?***

# Bias in healthcare need estimation




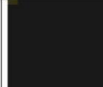
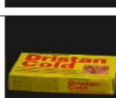


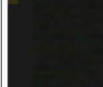

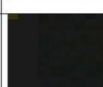
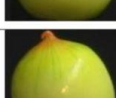
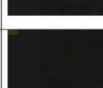
- Training based on healthcare spendings: But people of color are more likely to have lower incomes - making them less likely to access medical care even if they are insured
- Also, they may experience higher barriers to accessing health care (geography, transportation, work/childcare constraints), in addition to direct doctor-patient bias
- Data shows that race is correlated with substantial differences in health-care spendings: This results in a bias of the trained model


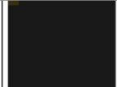

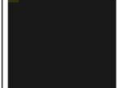

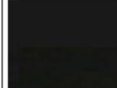
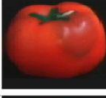
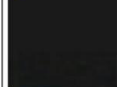

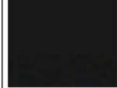

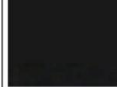
# AI mistakes in research software

- Applying machine learning methods to COVID-19 radiological imaging for improving the accuracy of diagnosis
- Distinguish patients with COVID-19 from patients without COVID-19 but also bacterial pneumonia
- Tools were trained on public datasets with CT and CXR images
- Predictive tools failed practical tests: <https://www.nature.com/articles/s42256-021-00307-0>
- ***Why?***

# Learning from the image background

- Image dataset was collected under controlled conditions:  
Does not represent the target distribution of interest
- CNN learnt to distinguish the image background rather than the image content
- Actually quite prevalent problem in computer vision

Subject No	Subject Image (128*128 pixels)	Rendered Image (22*29 pixels)
1		
		
		
2		
		
		

3		
		
		
4		
		
		



# AI mistakes in research software

- Predict whether a country is likely to slide into civil war based on GDP, poverty rates, type of government structure, etc.
- Complex models using Random Forests and Adaboost outperform more standard statistical approaches like logistic regression by far
- Missing values in the dataset were constructed using imputation on the complete dataset
- Models proved to be over-optimistic and erroneous <https://doi.org/10.1016/j.patter.2023.100804>
- ***Why?***

# Civil war predictions: Data leakage

- Data leakage: The data was imputed for missing values using the whole dataset
- Thus, the training dataset contained information about the test dataset
- This leads to an inflated estimate of the model performance

<https://doi.org/10.1016/j.patter.2023.100804> supplemental material



# Classification of failures/errors

# Data leakage

Spurious relationship between independent variables and target variable

Artifact of collection, sampling, pre-processing

Leads to inflated estimates of model performance

## **Lack of clean separation training/test**

- no test set
- pre-processing on training and test set (over/under sampling, imputation)
- feature selection on entire dataset
- duplicates in dataset

# Data leakage

## **Model uses features that are not legitimate**

- for example, use of a certain drug when predicting illness (hypertensive drug, antibiotics)

## **Test set is not drawn from distribution of scientific interest**

- temporal leakage (test set must not contain data from before the training set)
- non-independence between training and test samples (same people/units in both sets - use block crossvalidation)
- sampling bias in test distribution (spatial bias, age, image settings)

# Resources

- Good practices in machine learning (Mathieu Bauchy)  
(<https://www.youtube.com/watch?v=WScUQnU-ozQ&t=3213s>)
- Roadmaps <https://roadmap.sh/roadmaps>
- Kaggle <https://www.kaggle.com/learn>
- Hugging Face: <https://huggingface.co/learn>
- REFORMS checklist <https://reforms.cs.princeton.edu/>
- More resources <https://www.cs.princeton.edu/~arvindn/>,  
<https://www.aisnakeoil.com/p/introducing-the-ai-snake-oil-book>
- Scikit-learn resources [https://scikit-learn.org/stable/common\\_pitfalls.html](https://scikit-learn.org/stable/common_pitfalls.html)
- Testing of non-deterministic software  
[https://bssw.io/blog\\_posts/testing-non-deterministic-research-software](https://bssw.io/blog_posts/testing-non-deterministic-research-software)