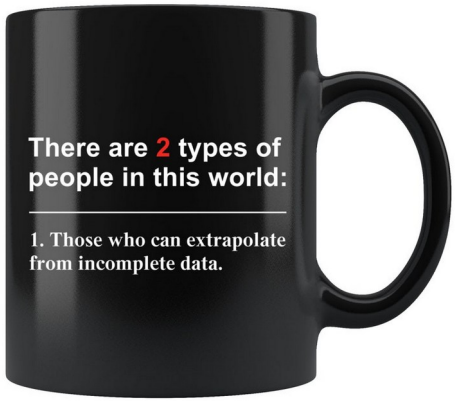


Data Analysis & Machine Learning
Missing data: mechanisms and solutions

Daniel Oberski

Department of Methodology & Statistics

Utrecht University

VERSION: 2026-09-29

Today

- **Why missing data matter**
- **Vocabulary: R / mask M, Y_obs, Y_mis, ψ; NDD / SDD / UDD**
- **Part A – inference: the classical toolbox**
  - deletion, mean / regression / stochastic imputation, multiple imputation; mice
- **Part B – prediction: the goal changes the rules**
  - missingness as a feature, native NA handling, sklearn pipelines, evaluation, shift
- **Part C – holes in the outcome**
- **A worked example (thyroid screening) and a checklist**

Reading

van Buuren, Flexible Imputation of Missing Data, §1.1–1.4 (stefvanbuuren.name/fimd)

Optional (prediction):

Josse, Chen, Prost, Varoquaux & Scornet (2024), Statistical Papers

Perez-Lebel et al. (2022), GigaScience

Le Morvan & Varoquaux (2025), ICLR

Lab this week: mechanisms by hand + mice (R)



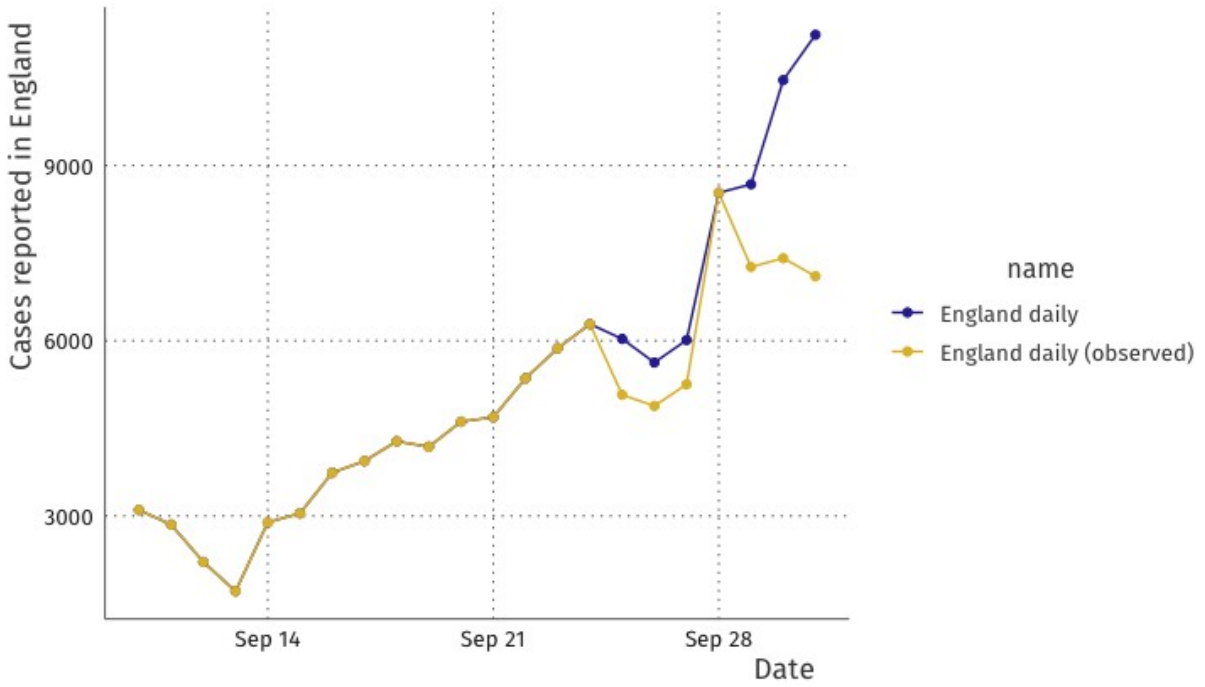
Image: [Sketchplanations]

Why is missing data important?

1. **Just annoying**
   - Most procedures don’t deal with missings by default
2. **Less information than planned**:
   - Uncertainty of estimates (e.g. “standard errors”, “power”, “C.I”, etc.)
   - Accuracy of predictive models
3. **Systematic biases**:
   - Estimates of interest wrong on average
   - Prediction error seems better than it will be in reality

Covid cases lost in Excel (UK, 2020)





Question: we want to estimate the trend in cases. What does the Excel error do to that estimate?

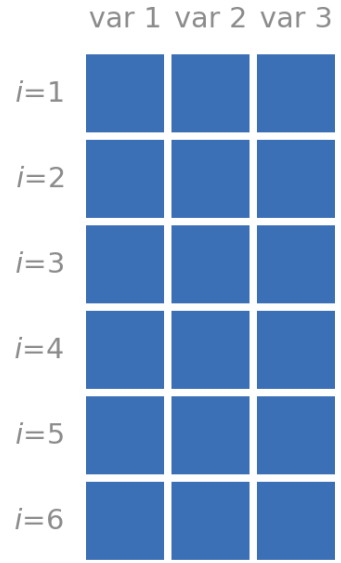
How are missing data “coded”?

- R data frames: `NA`
- Python/Pandas: `NaN`
- SQL: `NULL`
- JavaScript/ json: `null`
- .csv files: `"NA", "", "NULL"`
- SPSS: `-999, 99999, …` *sigh*.

So pay attention during data wrangling!

Notation: R, the mask M, Y_obs, Y_mis, ψ

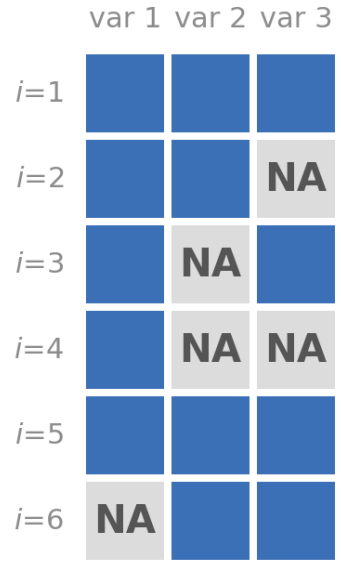
Y (complete data)



R (response indicator)

	var 1	var 2	var 3
$i=1$	1	1	1
$i=2$	1	1	0
$i=3$	1	0	1
$i=4$	1	0	0
$i=5$	1	1	1
$i=6$	0	1	1

Y_{obs} (what we see)

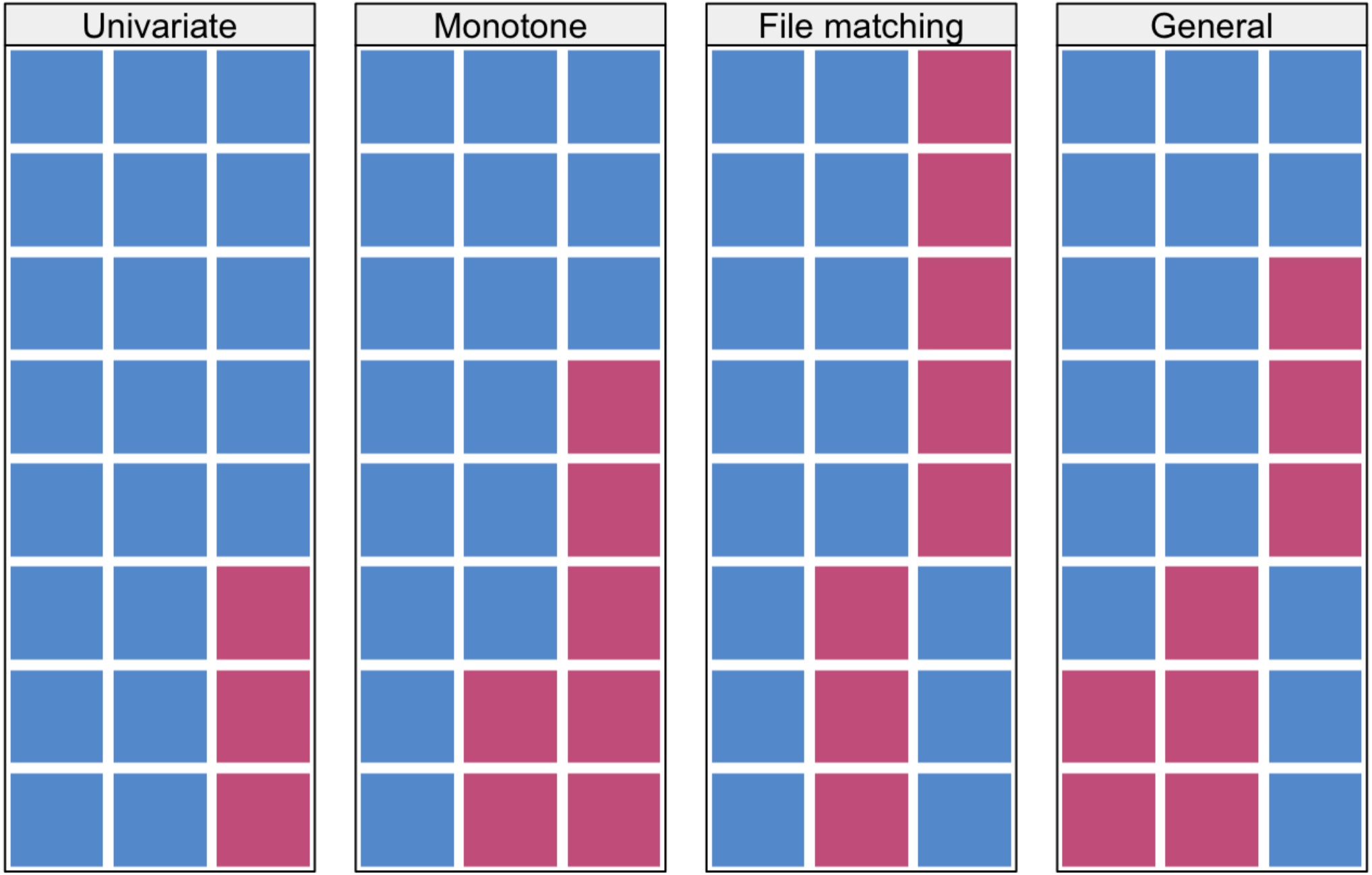


$Y = (Y_{obs}, Y_{mis})$

$R_{ij} = 1$ if observed, 0 if missing

$\Pr(R = 0 \mid Y_{obs}, Y_{mis}, \psi)$

the **mask** $M = 1 - R$ ($M_{ij} = 1$ if missing) is the same information, as machine learning writes it


Univariate
Monotone
File matching
General

Shining a light

Use ***what you do know*** *about* ***what you don't know***

NDD: It's missing for reasons unrelated to the data (MCAR)

Sickness preventing students from sitting exam

SDD: It's missing for reasons related to data you have got (MAR)

School discouraging less able students from sitting exam

UDD: missing because of the values you *would have* obtained (MNAR) Student realised revised wrong material, so didn't turn up to sit exam

Source: David Hand

NDD, SDD, UDD: what may R depend on?

NDD
"MCAR"

Not Data-Dependent

$\Pr(R = 0 \mid Y_{obs}, Y_{mis}, \psi) = \Pr(R = 0 \mid \psi)$

Server log deleted; sickness before the exam

SDD
"MAR"

Seen Data-Dependent

$\Pr(R = 0 \mid Y_{obs}, Y_{mis}, \psi) = \Pr(R = 0 \mid Y_{obs}, \psi)$

School keeps weak students (known previous test) from the exam

UDD
"MNAR"

Unseen Data-Dependent

$\Pr(R = 0 \mid Y_{obs}, Y_{mis}, \psi)$ – no simplification

Rich people don't report income; students who revised the wrong material skip the exam

NDD vs SDD can be checked against observed variables; SDD vs UDD cannot be tested from the data – it is an assumption.

MCAR, MAR, MNAR ????

MCAR: Missing Completely At Random (NDD)

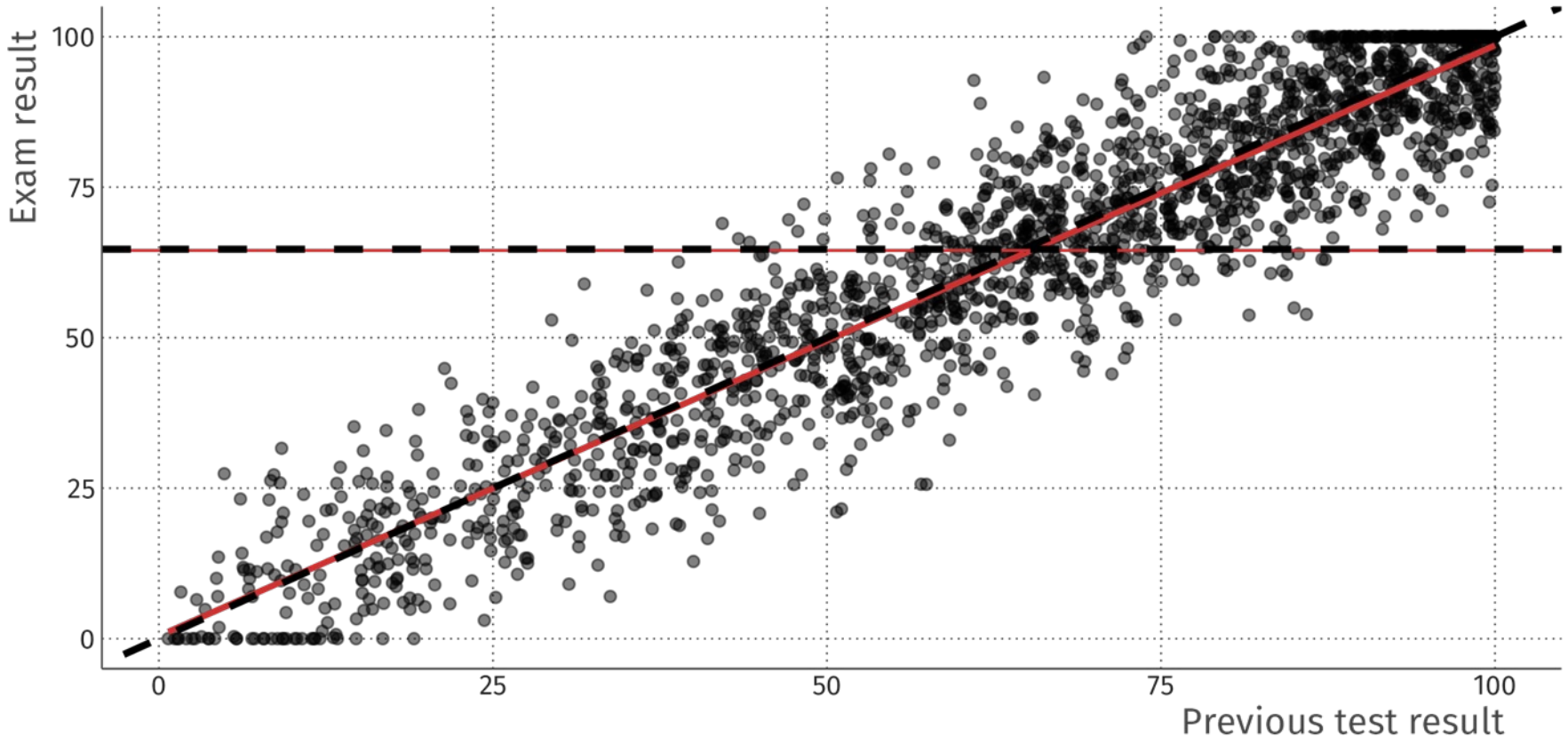
MAR: Missing At Random (SDD)

MNAR: Missing Not At Random (UDD)

WARNING

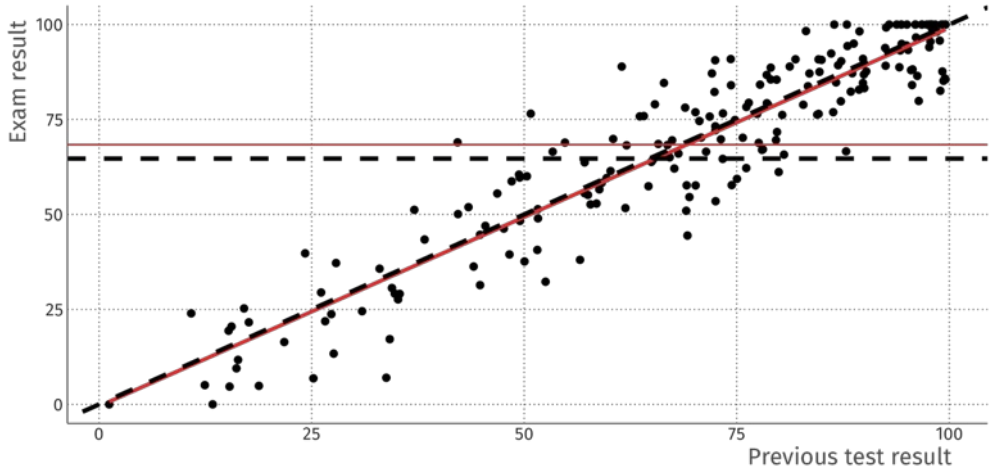
- This nomenclature is due to Rubin and has confused and confounded many generations of students and practictioners.
- Hand’s (2020) terms refer to the exact same concepts, but have the advantage of being intelligible.

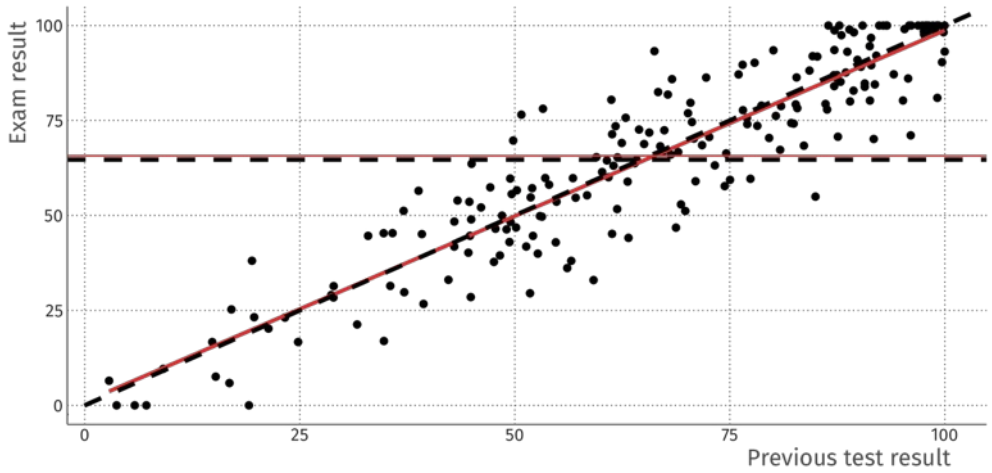
Complete data

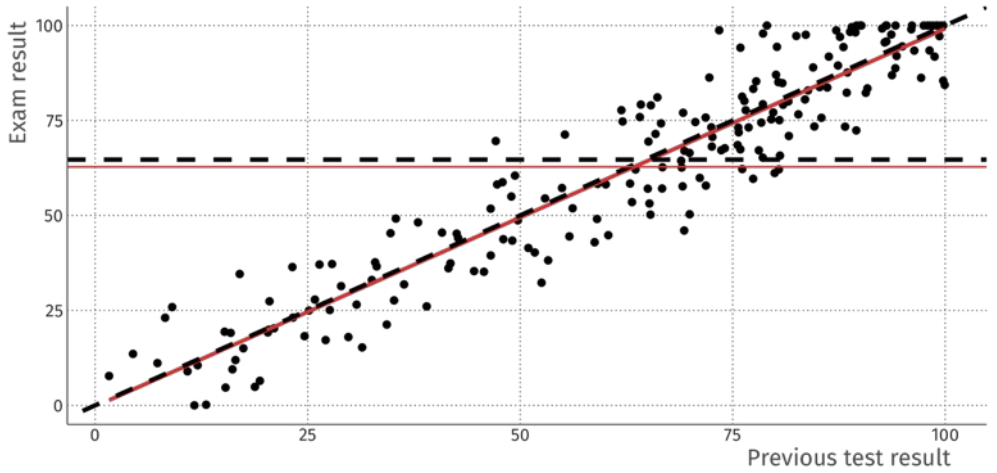


NDD Not Data-Dependent (“MCAR”)

Sickness preventing students from sitting exam

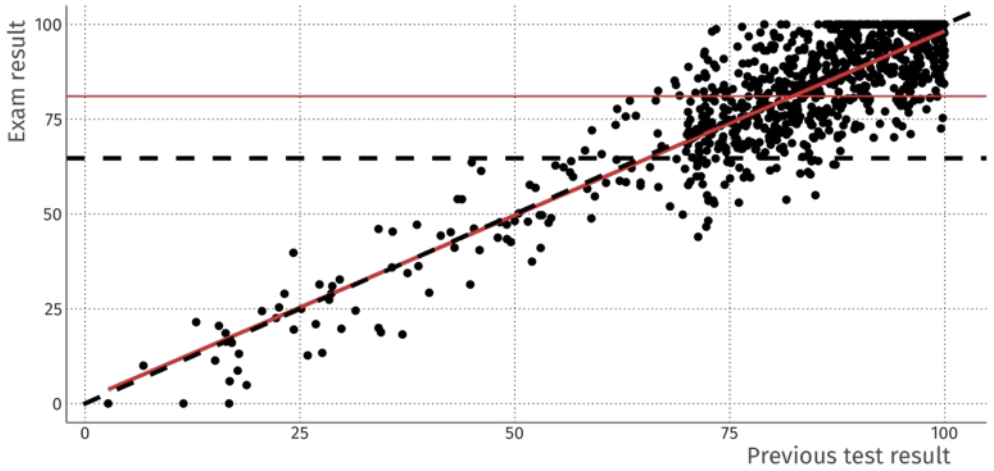


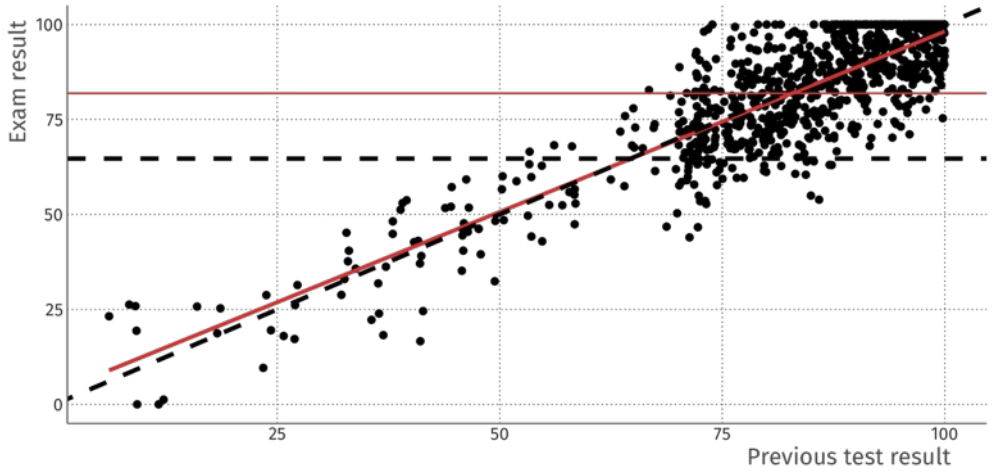


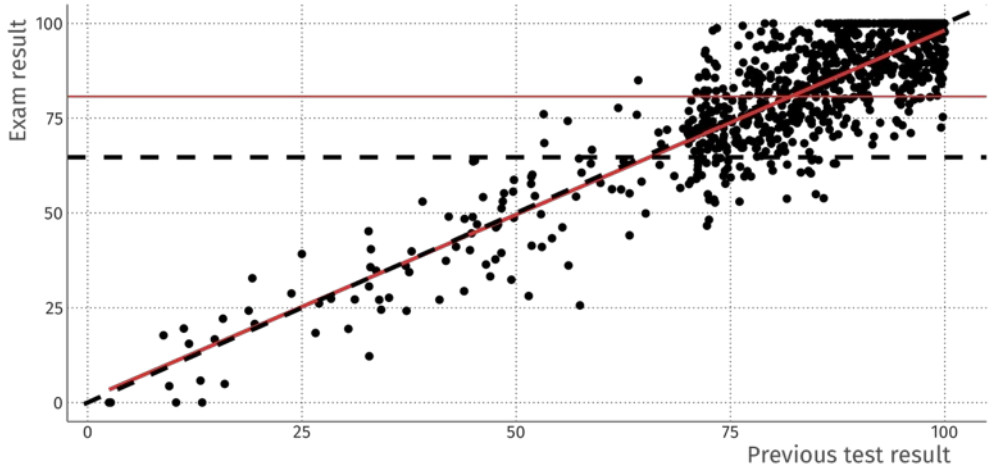


SDD *Seen* Data-Dependent (“MAR”)

School discouraging students from sitting exam based on previous test

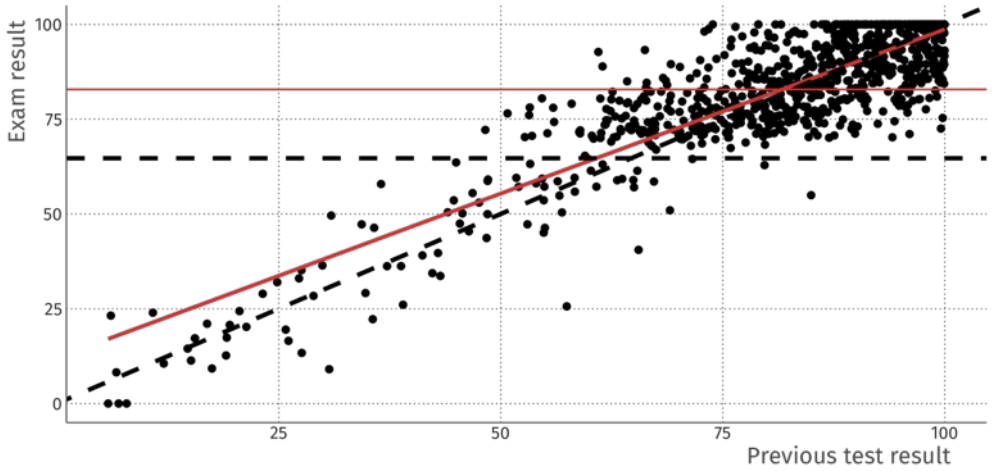


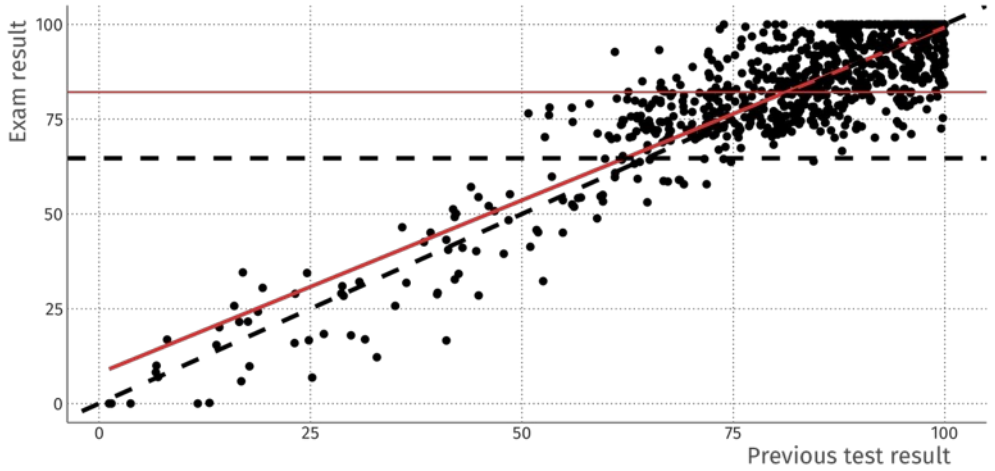


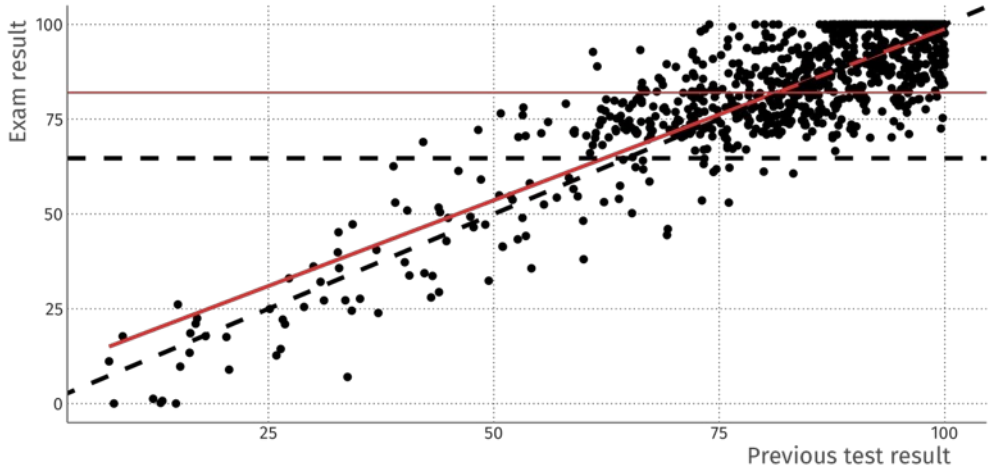


UDD *Unseen* Data-Dependent (“MNAR”)

Students realised they revised wrong material, so didn’t turn up to sit exam







Practice

Question: Is this (probably) NDD, SDD, or UDD:

1. We asked people's income, and everybody answered. But then a server log file was accidentally deleted;
2. We asked people's income, but the rich people did not want to say;
3. We asked people's income, but those with a university education did not want to say. We only observe income.
4. Same as 3, but now we also observe education.

Two goals, two kinds of holes

	holes in the features X	holes in the outcome y
Inference estimate θ with a valid SE	**Classical toolbox** multiple imputation / likelihood; needs SDD; imputations must be plausible	**Selection problem** complete cases OK if R ⊥ y | X; otherwise weighting or selection models
Prediction good ŷ for a new, incomplete row	**New rules** target E[y | X_obs, M]; missingness is a feature; constant + mask or native NA	**Training & evaluation** rows without y add little; missing labels bias your metrics

Part A – Inference with holes in X

	holes in the features X	holes in the outcome y
Inference estimate θ with a valid SE	**Classical toolbox** multiple imputation / likelihood; needs SDD; imputations must be plausible	**Selection problem** complete cases OK if R ⊥ y | X; otherwise weighting or selection models
Prediction good ŷ for a new, incomplete row	**New rules** target E[y | X_obs, M]; missingness is a feature; constant + mask or native NA	**Training & evaluation** rows without y add little; missing labels bias your metrics

The textbook toolbox (FIMD ch. 1): what each method assumes and what it gets wrong

INFERENCE

Listwise deletion (complete-case analysis)

For inference (estimate θ, SE)

✓ Simple; the default of lm(), glm(), …

✓ Unbiased under NDD; for a regression also when $R \perp y \mid X$

✗ Throws away rows: with many features almost nothing is left

✗ Biased under SDD and UDD, even for a mean

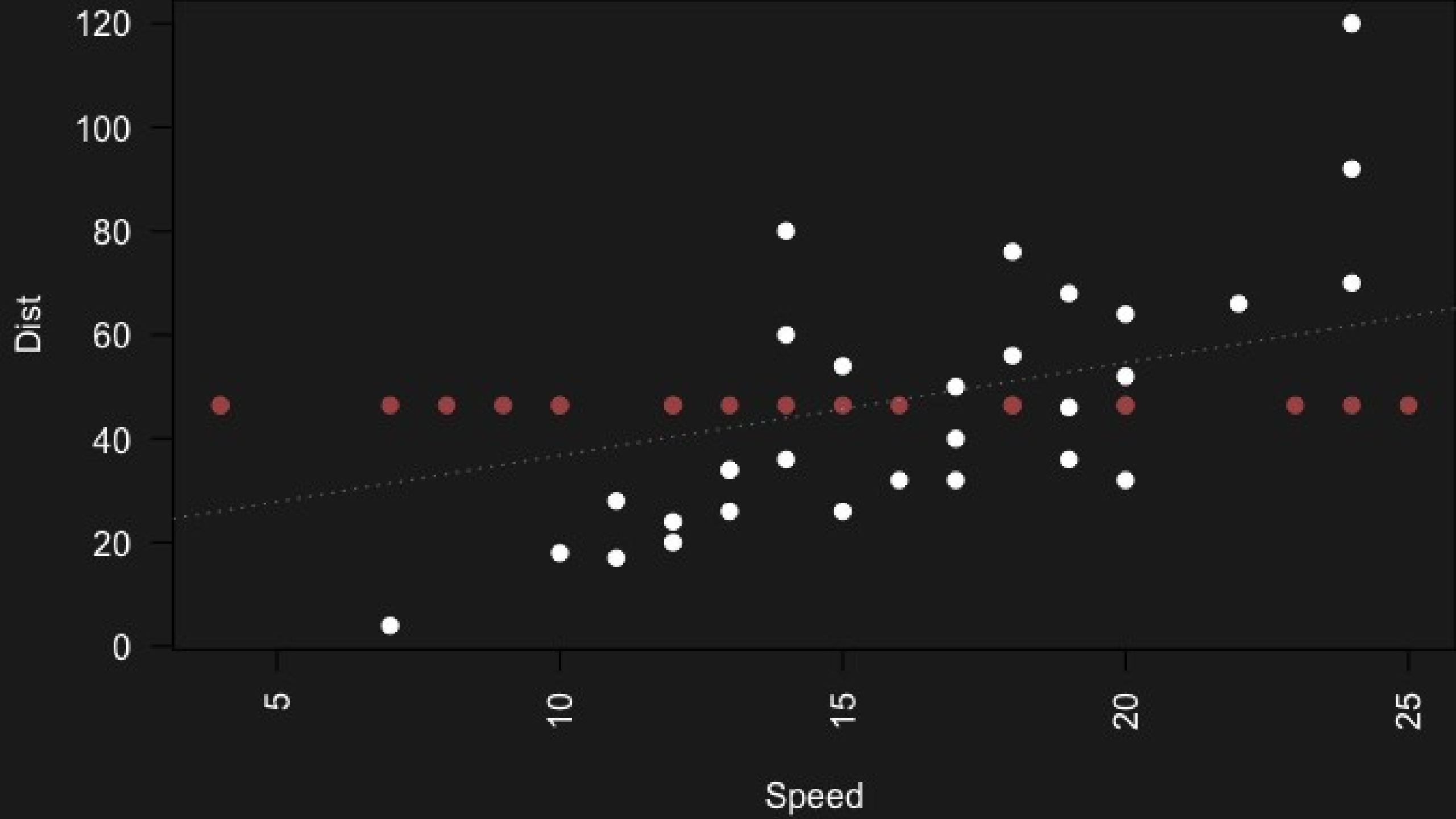
✗ SEs too large (less information than you had)

For prediction (ŷ for new rows)

✗ Cannot be applied at prediction time: an incomplete new row gets no prediction

✗ Loses training rows and the information in the pattern

- Fine only if deployment data are complete – then it is the complete pattern
- Rows with missing y: dropping them is the right default (Part C)


120
100
80
60
40
20
0
Dist
5
10
15
20
25
Speed

INFERENCE

Mean / median imputation

For inference (estimate θ, SE)

✓ Unbiased for the mean under NDD

✗ Disturbs the distribution; variance too small

✗ Correlations and regression coefficients biased towards 0

✗ Biased under SDD; SEs too small

✗ FIMD: ‘should be avoided in general’

For prediction (ŷ for new rows)

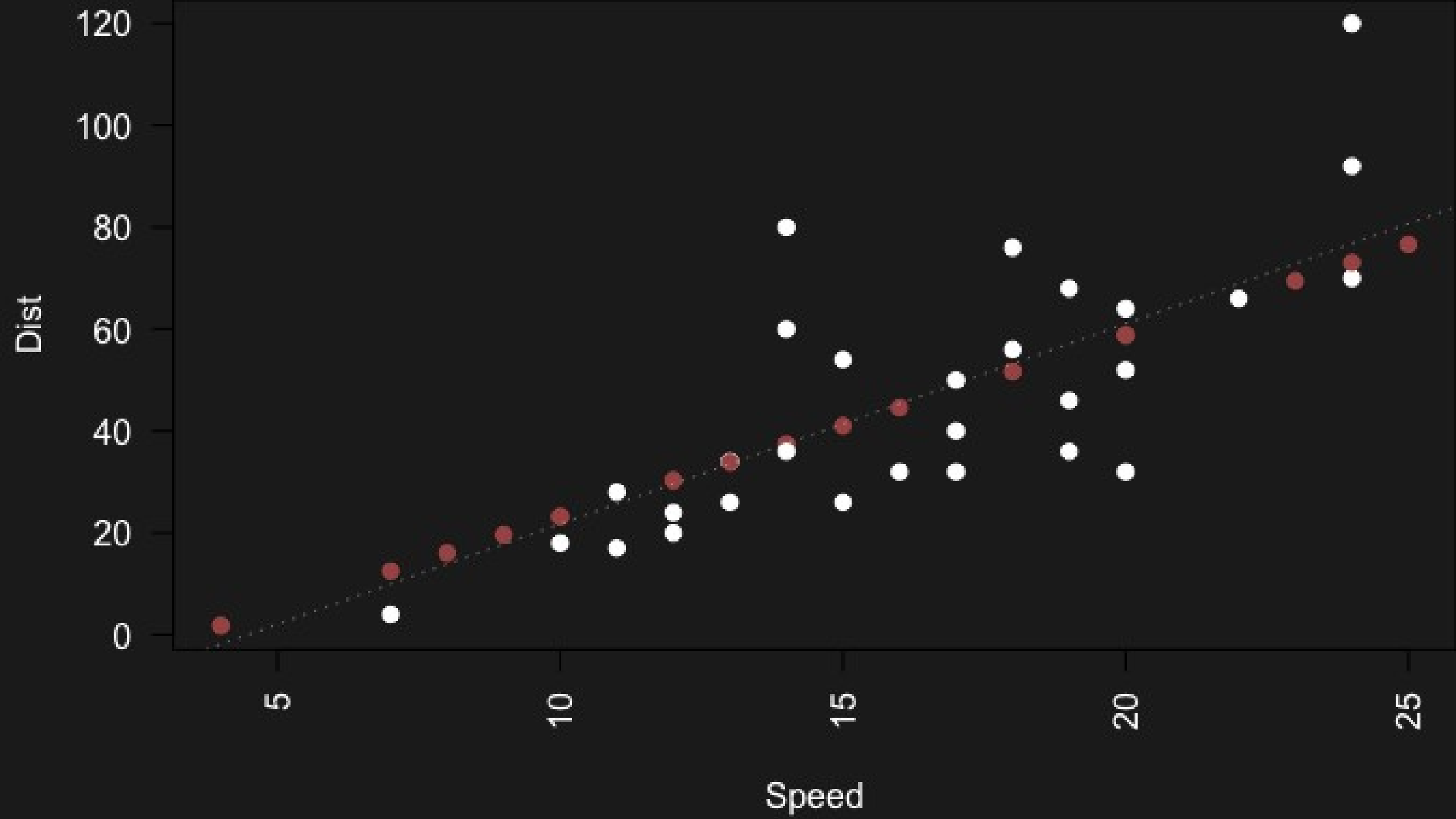
✓ Consistent: same constant at train and test + a learner that can spot it (constant + mask)

✓ Cheapest; hard to beat with boosted trees on real data

✓ Keeps the information that a value was missing

✗ A linear model needs the mask (and interactions) to use it

✗ Categorical: never the mode – add a ‘missing’ level


120
100
80
60
40
20
0
Dist
5
10
15
20
25
Speed

INFERENCE

Regression imputation (conditional mean)

For inference (estimate θ, SE)

✓ Regression coefficients unbiased under SDD

✗ Imputed points lie exactly on the line: correlations inflated, ‘too good to be true’

✗ Variance and SEs too small; CIs too narrow, p-values too optimistic

For prediction (ŷ for new rows)

✓ A fine deterministic feature transform: fit on the training fold, freeze, apply to new rows

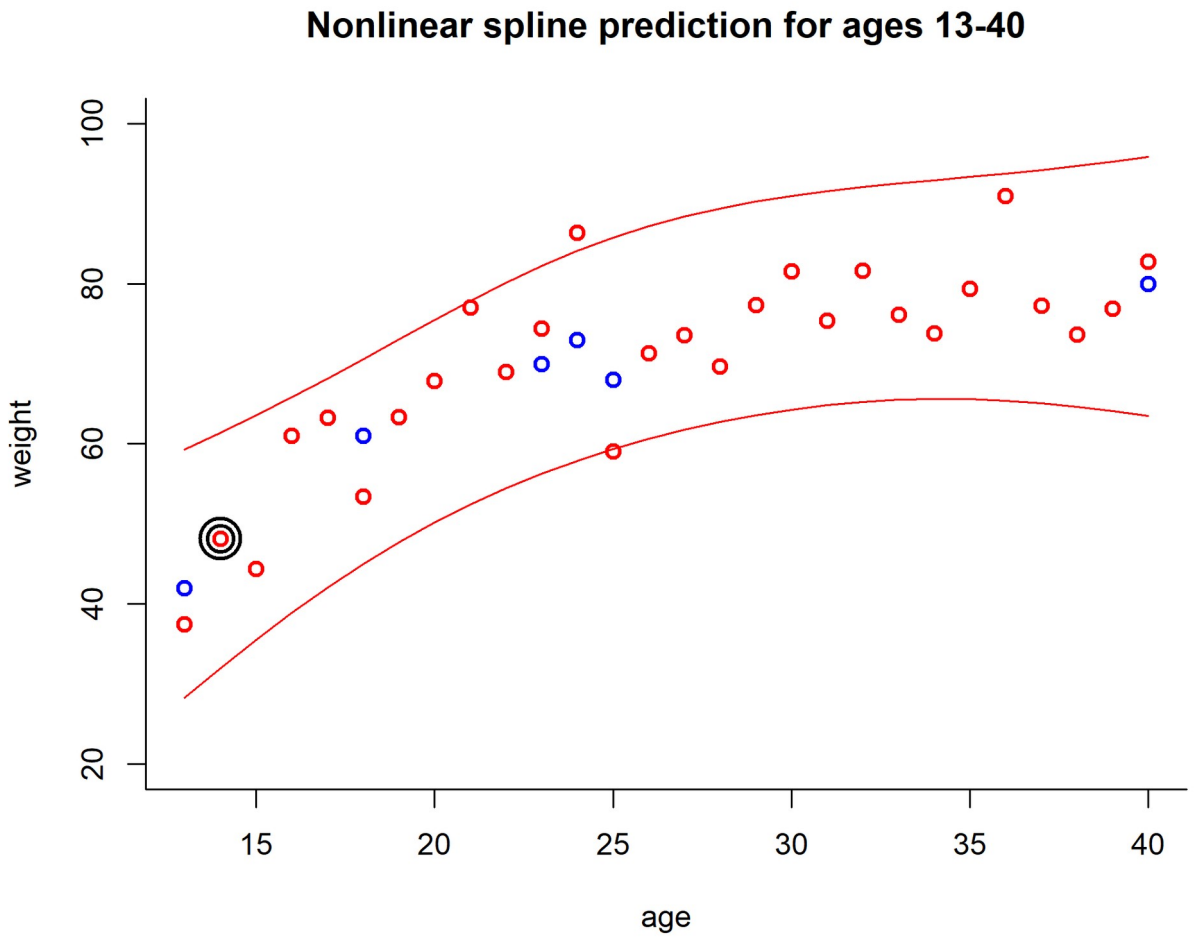
✓ Smoother function to learn: small gains for small n / linear models / correlated features

✗ Gains are small (diminishing returns) and cost 100–1000× more

✗ Hides missingness → add the mask; can hurt under UDD without it

✗ If the imputed variable is your outcome, CV error looks too good (Part C)

Stochastic regression imputation



INFERENCE

Stochastic regression imputation

For inference (estimate θ, SE)

✓ Preserves the distribution and the correlations

✓ Coefficients unbiased under SDD

✗ One draw still treats imputed values as known: SEs too small

- Fixed by doing it m times: multiple imputation (next)

For prediction (ŷ for new rows)

✗ A single random draw adds pure noise to a feature: worse than the conditional mean

✗ Pattern no longer recognisable from the value → the mask becomes essential

✓ Averaging predictions over many draws recovers the conditional mean (= the multiple-imputation ensemble)

- Useful when you want the ‘as if it had been measured’ prediction (shift-robust, needs SDD)

Same ideas, different names

Idea	R: mice	Python: scikit-learn
mean / median	`mice(method = "mean")`	`SimpleImputer(strategy="mean")`
regression (conditional mean)	`mice(method = "norm.predict", m = 1)`	`IterativeImputer()  # default: deterministic`
stochastic regression	`mice(method = "norm.nob") / "pmm" (default)`	`IterativeImputer(sample_posterior=True)`
chained equations = ‘iterative’	`mice(): each variable in turn, maxit cycles`	`IterativeImputer: same loop, max_iter`
missForest	`mice(method = "rf"), missForest()`	`IterativeImputer(estimator=ExtraTreesRegressor())`
multiple imputation (m sets)	`mice(m = 20) → with() → pool()`	`m × IterativeImputer(sample_posterior=True, random_state=k), average predictions`
mask / indicator	`is.na(); md.pattern(), ggmice`	`SimpleImputer(add_indicator=True), MissingIndicator`
native NA in trees	`xgboost, lightgbm, partykit (MIA); rpart: surrogates`	`HistGradientBoosting*, XGBoost, LightGBM`

Overview of methods (inference)

INFERENCE

	Assumption needed to get unbiased estimates			
	Mean	Regression coef.	Correlation	Standard Error
Listwise deletion	NDD	NDD	NDD	Too large
Mean imputation	NDD	-	-	Too small
Regression imputation	SDD	SDD	-	Too small
Stochastic imputation	SDD	SDD	SDD	Too small

INFERENCE

Multiple imputation

For inference (estimate θ, SE)

✓ m stochastic imputations → m analyses → pool (Rubin's rules): SEs correct

✓ The standard for valid inference under SDD

✗ Still assumes SDD; needs a good imputation model (include y!)

• Sensitivity check: does the conclusion change across the m datasets?

For prediction (ŷ for new rows)

✓ Averaging predictions over m imputed sets = an ensemble; often the best-ranked pipeline

✗ …but the gain is bagging, not better imputation; 100× the compute

✗ Needs the imputation model at prediction time, applied to each new row

• Targets 'y as if x had been measured': robust to missingness shift, needs SDD

✗ Rubin's rules pool parameters – there is nothing to pool for a ŷ

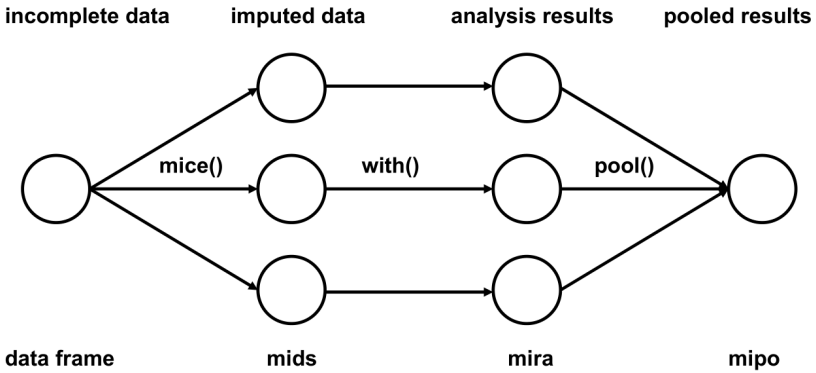
Inference in R: mice

```R
library(mice); library(ISLR)
d <- College[, c("Expend","Outstate","PhD","Terminal")]

# 1. impute: m = 20 stochastic imputations (pmm)
imp <- mice(d, m = 20, method = "pmm", maxit = 10, seed = 1)
plot(imp)          # convergence check

# 2. analyse each completed data set
fit <- with(imp, lm(Expend ~ Outstate + PhD))

# 3. pool with Rubin's rules: estimate, SE, fmi
summary(pool(fit))
```



Single imputation for comparison: mice(d, m = 1, method = "norm.predict") (regression) · method = "norm.nob" (stochastic)

Then look at: densityplot(imp) – do imputed values look like observed ones?

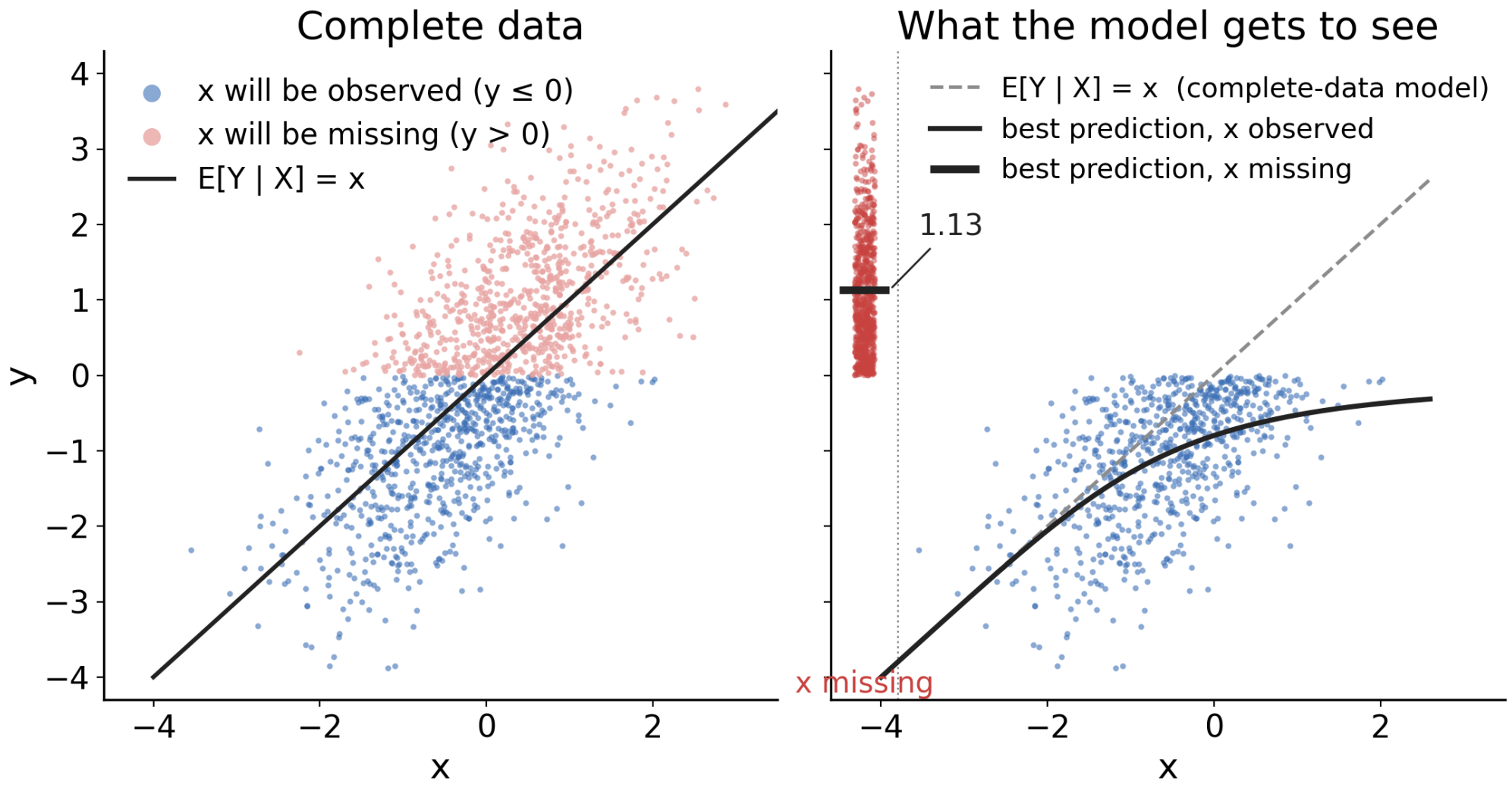
Part B – Prediction with holes in X

	holes in the features X	holes in the outcome y
Inference estimate θ with a valid SE	**Classical toolbox** multiple imputation / likelihood; needs SDD; imputations must be plausible	**Selection problem** complete cases OK if R ⊥ y | X; otherwise weighting or selection models
Prediction good ŷ for a new, incomplete row	**New rules** target E[y | X_obs, M]; missingness is a feature; constant + mask or native NA	**Training & evaluation** rows without y add little; missing labels bias your metrics

Same holes, different goal: a good ŷ for every new row, including incomplete ones

PREDICTION

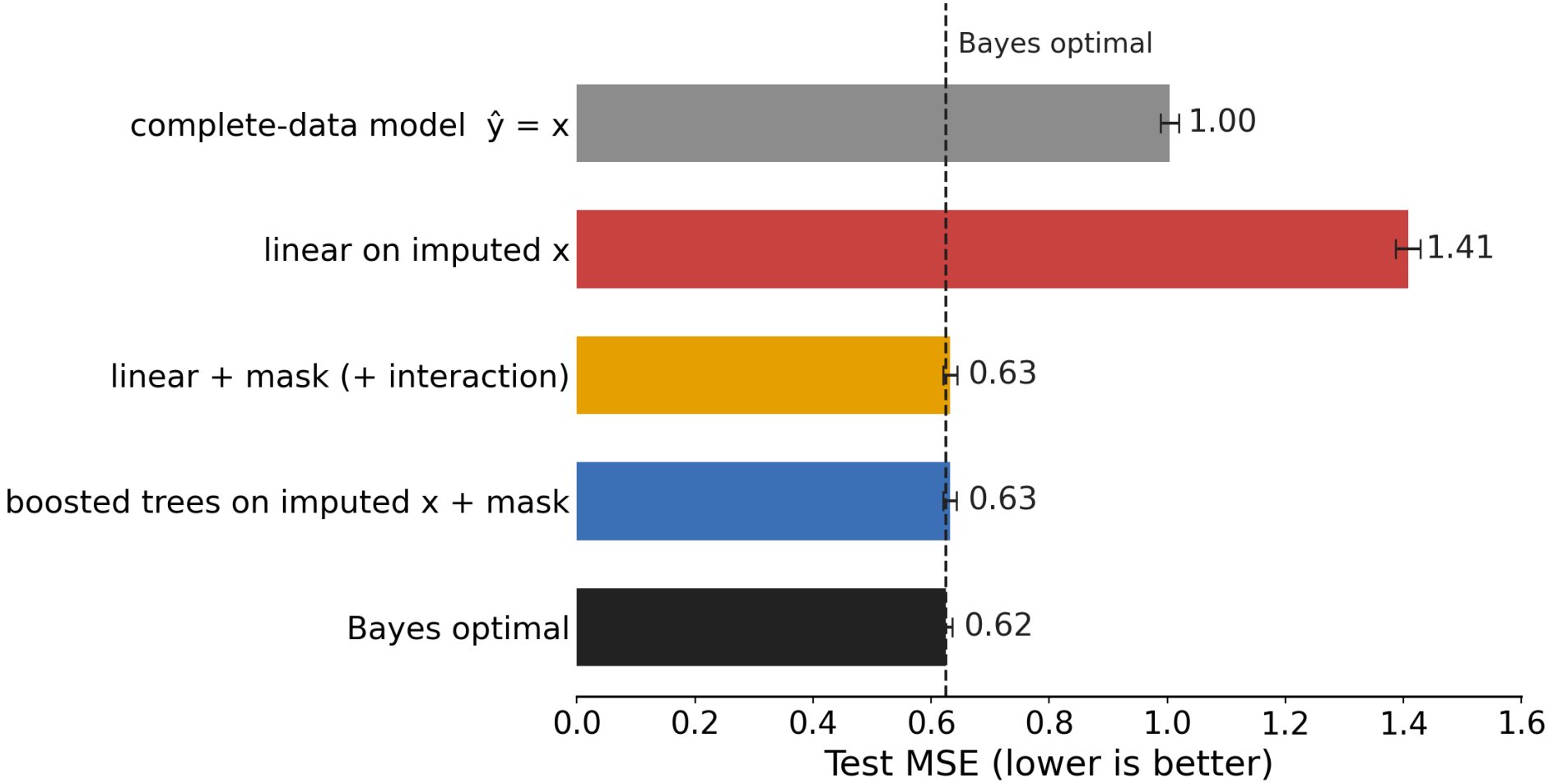
One best predictor per missingness pattern



Simulation: y = x + ε, x missing when y > 0. Best predictor E[y | x_obs, M]: x − φ(x)/Φ(−x) if observed, 2/√π if missing (Josse et al. 2024).

PREDICTION

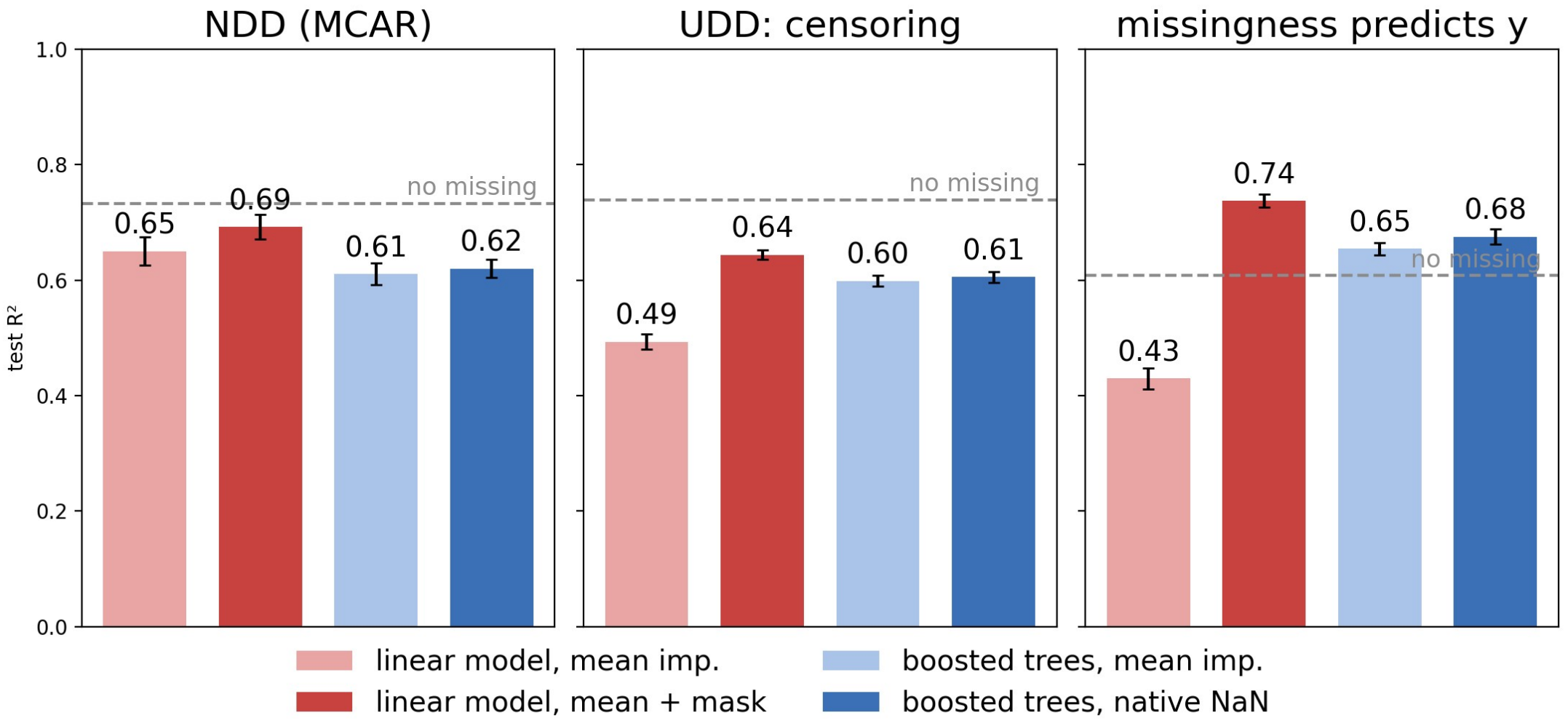
Result 1: constant imputation is consistent



Conditions: same rule at train & test • learner can recognise imputed values (constant, mask) • same missingness process at train & test

PREDICTION

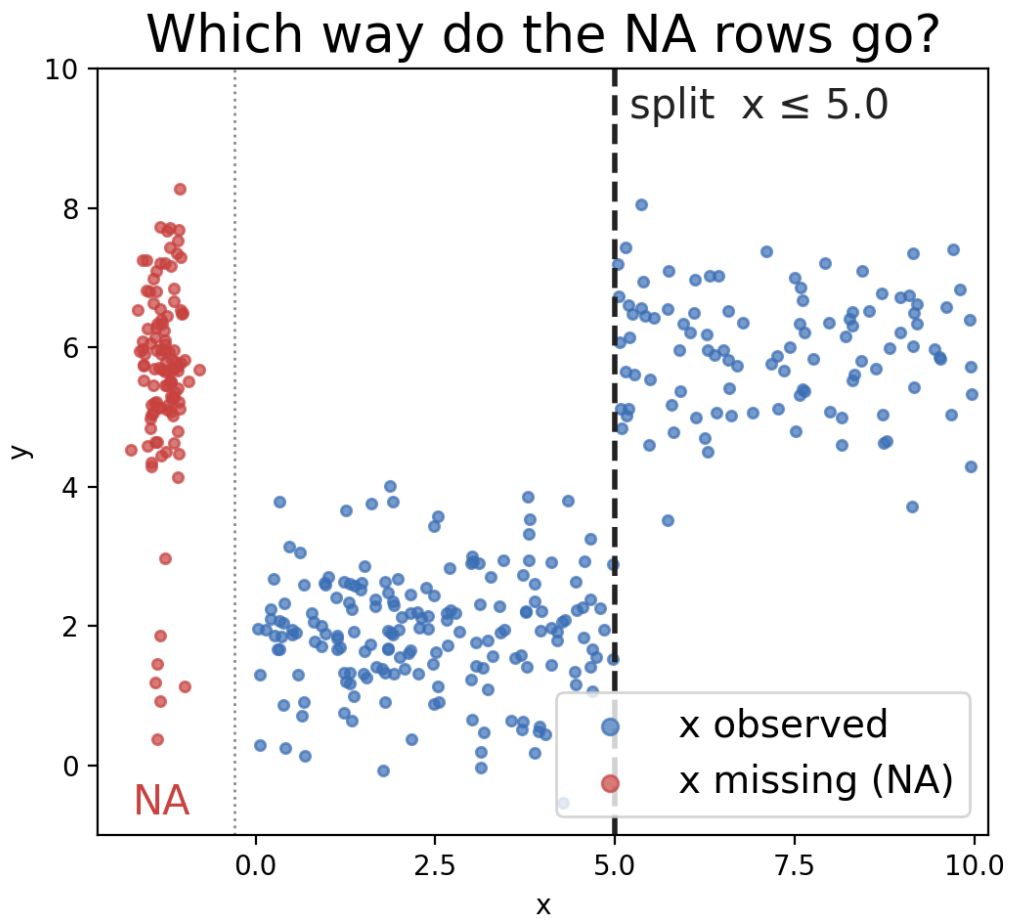
Result 2: add the mask M as features

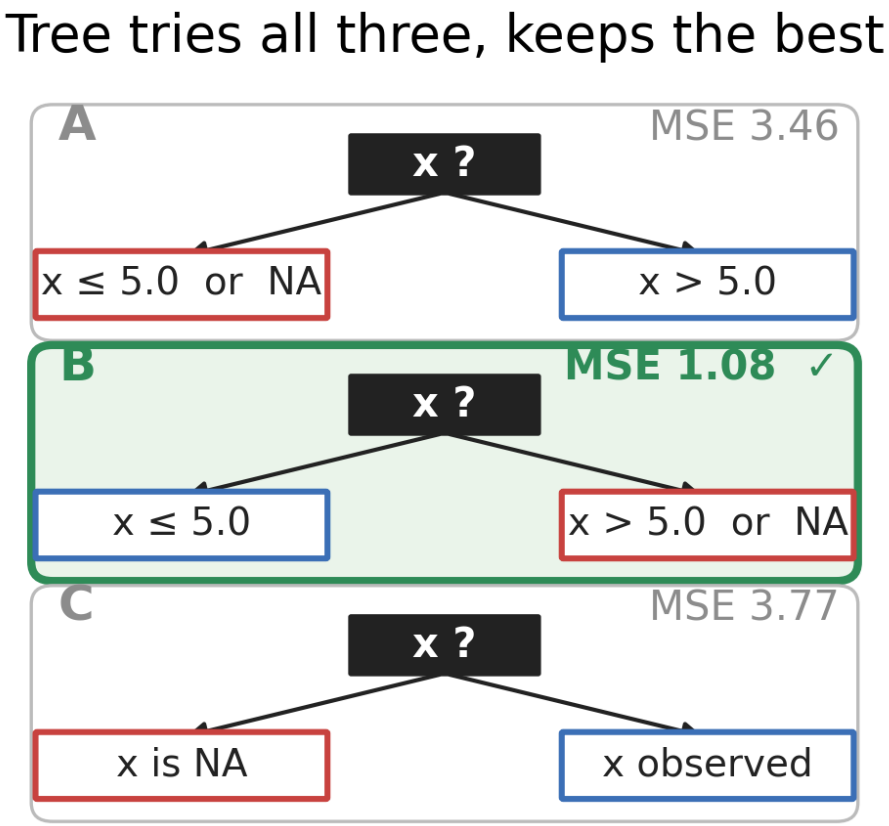


Same simulation: 6 correlated predictors, 20% missing in three of them, n = 2000. Mask = the M columns appended as features.

PREDICTION

Trees handle NA natively (MIA)





Prediction time: a new row with x = NA simply follows the learned branch – no imputation step. XGBoost, LightGBM, scikit-learn HistGradientBoosting do this by default (“missing incorporated in attribute”, Twala et al. 2008).

PREDICTION

Prediction in Python: sklearn pipelines

Python

```
from sklearn.ensemble import
HistGradientBoostingClassifier
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import
RepeatedStratifiedKFold

# A. boosted trees: NaN handled natively (MIA)
hgb = HistGradientBoostingClassifier(
        max_iter=200, learning_rate=0.05,
        max_leaf_nodes=15)

cv = RepeatedStratifiedKFold(n_splits=5,
        n_repeats=10, random_state=0)
cross_val_score(hgb, X, y, cv=cv,
                scoring="roc_auc")
```

Python

```
from sklearn.pipeline import make_pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

# B. other learners: constant + mask, in a Pipeline
lr_mask = make_pipeline(
    SimpleImputer(strategy="mean",
                  add_indicator=True),   # <- mask M
    StandardScaler(),
    LogisticRegression(max_iter=2000))

cross_val_score(lr_mask, X, y, cv=cv,
                scoring="roc_auc")
```

Why a Pipeline: means and scaler are fitted on the training fold only and re-used unchanged on the test fold – and on every new row. Imputing before the split leaks.

Categoricals: SimpleImputer(strategy="constant", fill_value="missing") → OneHotEncoder(handle_unknown="ignore")

Conditional means instead of the mean: IterativeImputer(add_indicator=True) – same place in the pipeline.

PREDICTION

Multiple imputation for prediction: code

Python

```
mu, sd = np.nanmean(X, 0), np.nanstd(X, 0)   # fixed scale: coefficients poolable
imps, Q, U = [], [], []
for k in range(m):                                # m = 20 stochastic imputations
    imp = IterativeImputer(sample_posterior=True, random_state=k)
    Zk  = (imp.fit_transform(X) - mu) / sd     # k-th completed training set (no
y)
    lr  = LogisticRegression(max_iter=3000).fit(Zk, y)
    Q.append(np.r_[lr.intercept_, lr.coef_.ravel()])  # estimates
    U.append(coef_var(Zk, lr))                        # their variances
    imps.append(imp)
Q, U = np.array(Q), np.array(U)

beta = Q.mean(0)                                  # Rubin: pooled coefficients
W, B = U.mean(0), Q.var(0, ddof=1)                # within / between variance
T    = W + (1 + 1/m) * B                          # total variance, SE = sqrt(T)
fmi  = (1 + 1/m) * B / T                          # fraction of missing information

def predict_proba(X_new):                         # new row: impute m times,
    p = []                                        # pooled model, average
    for imp in imps:
        Z = (imp.transform(X_new) - mu) / sd
        p.append(1 / (1 + np.exp(-(beta[0] + Z @ beta[1:]))))
    return np.mean(p, axis=0)
```

What it is

m stochastic imputations of the training features (no y: unavailable for new rows)

one model per completed set; coefficients pooled with Rubin’s rules

a new row is imputed m times → m probabilities → average

What it gives

the as-if-measured risk: robust to missingness shift, needs SDD

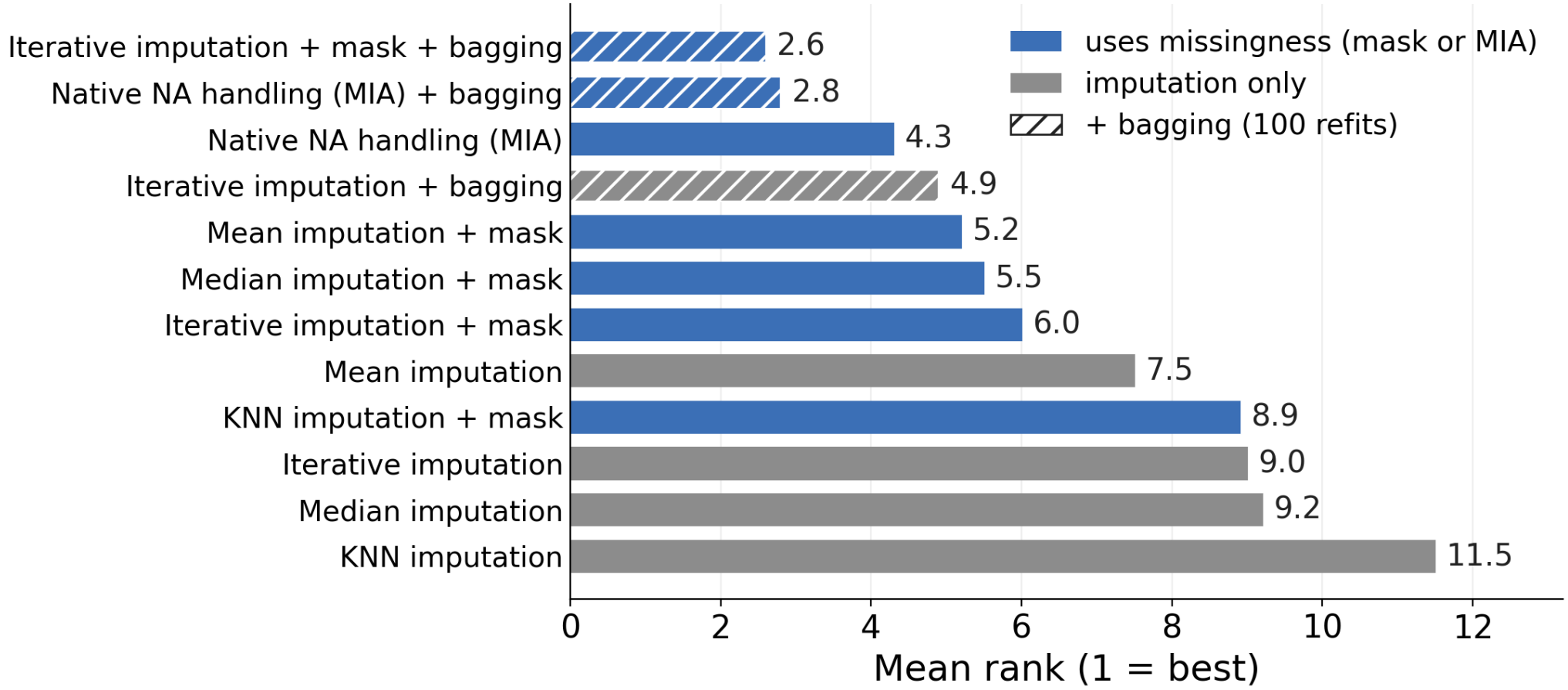
T and FMI tell you how much the missing values cost (T3: FMI 0.26)

What it costs

m× compute; the imputers must ship with the model

PREDICTION

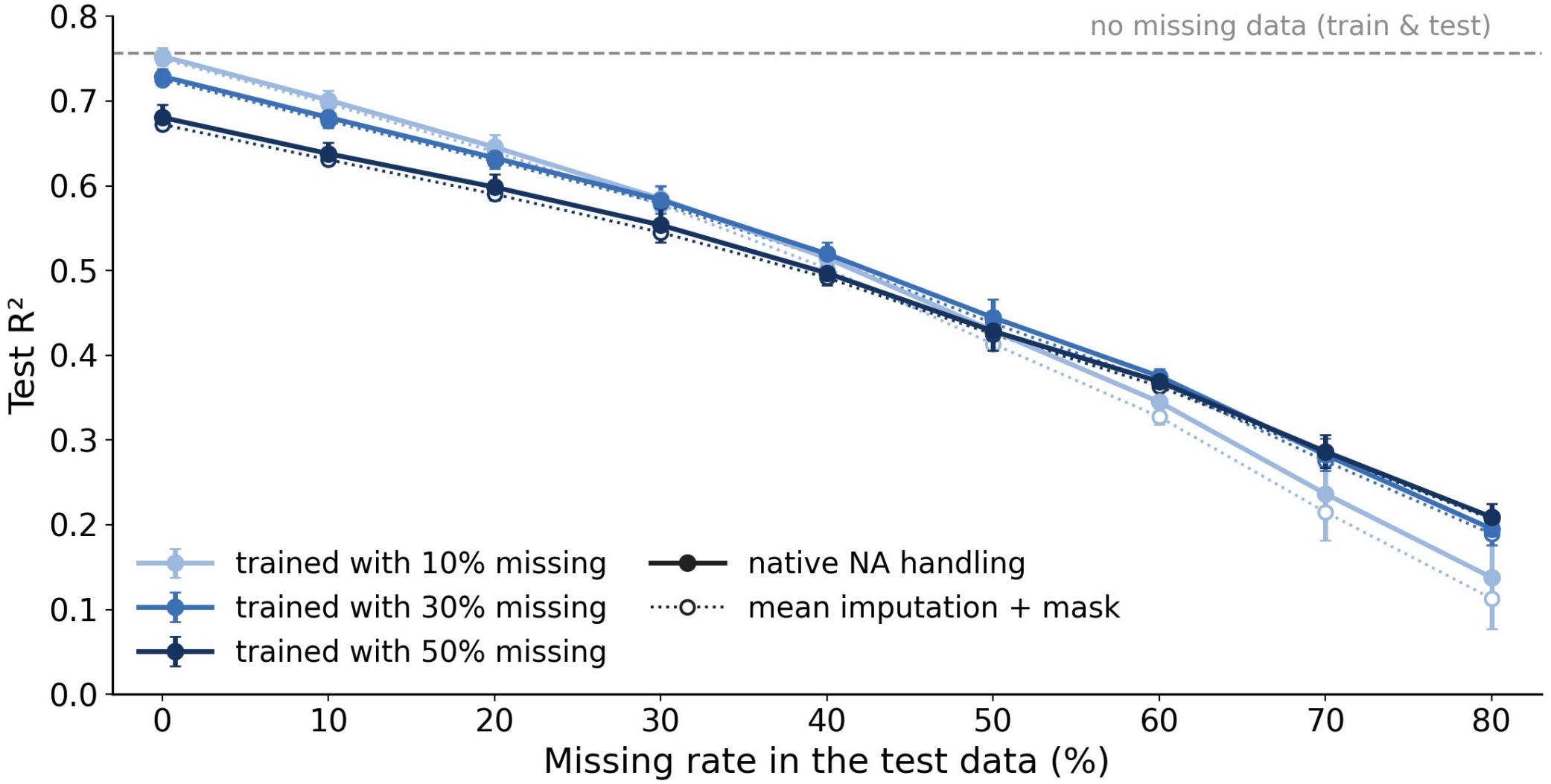
Real data: 13 clinical prediction tasks



Mean rank over 13 tasks (Traumabase, UK Biobank, MIMIC-III, NHIS); learner: gradient-boosted trees. bagging = 100 bootstrap refits (≈100–200× compute). Source: Perez-Lebel et al. 2022.

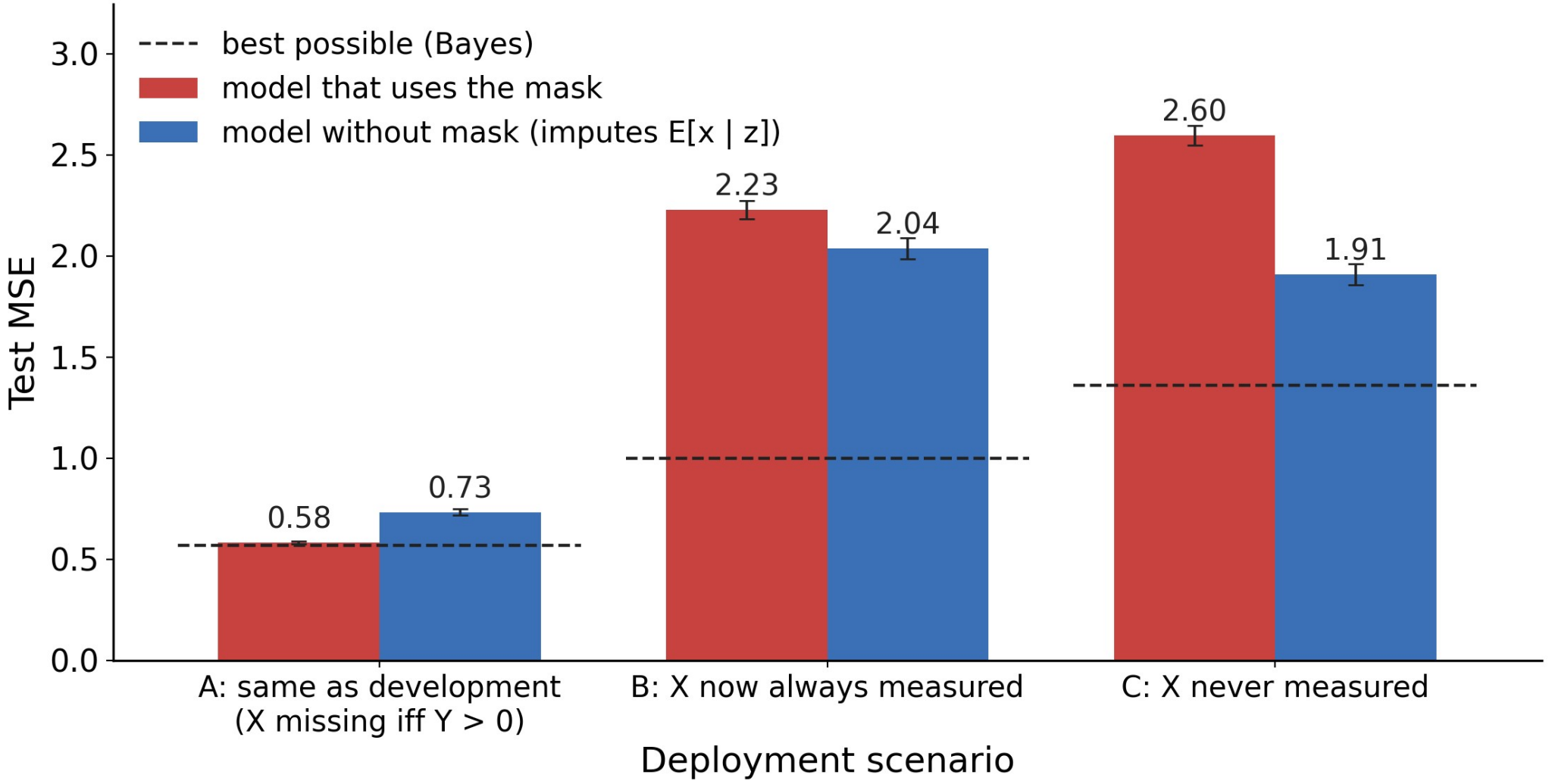
PREDICTION

What hurts: missingness at prediction time



PREDICTION

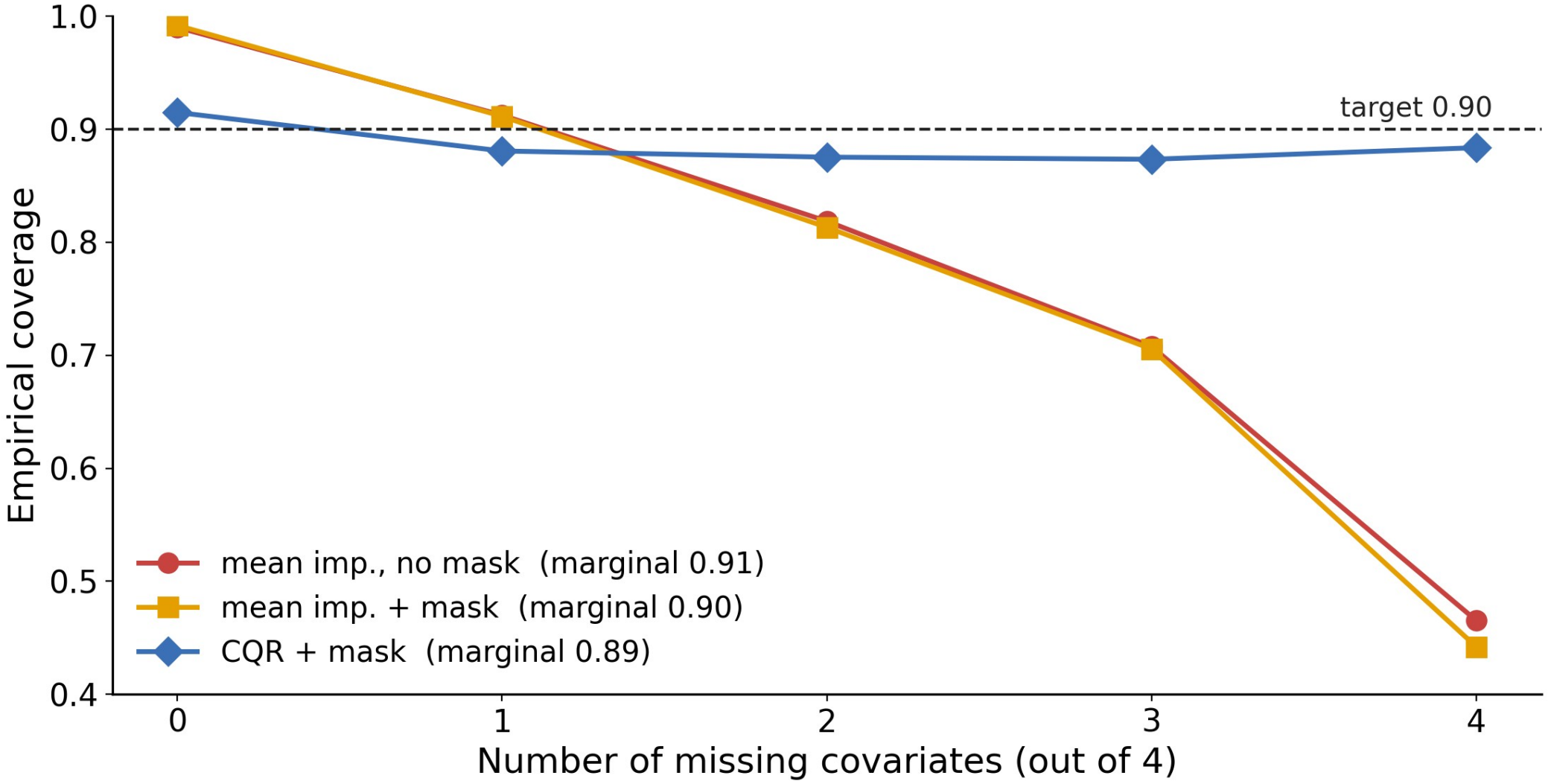
Missingness shift breaks the mask



Development data: x missing when y > 0. Deployment scenarios A–C. Boosted trees; 10 repetitions, n = 10 000. Dashed = best possible per scenario.

PREDICTION

Check calibration and coverage per pattern



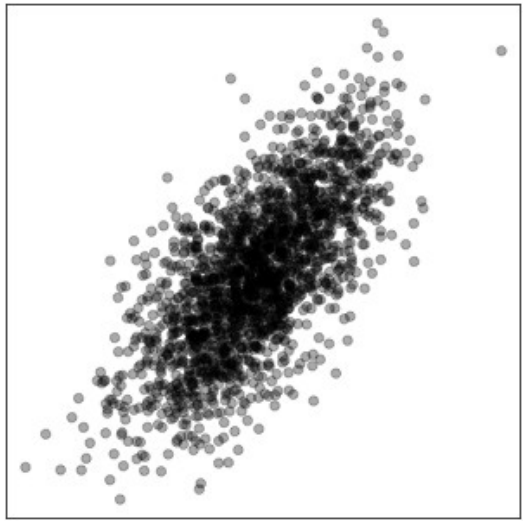
Part C – Holes in the outcome y

	holes in the features X	holes in the outcome y
Inference estimate θ with a valid SE	**Classical toolbox** multiple imputation / likelihood; needs SDD; imputations must be plausible	**Selection problem** complete cases OK if R ⊥ y | X; otherwise weighting or selection models
Prediction good ŷ for a new, incomplete row	**New rules** target E[y | X_obs, M]; missingness is a feature; constant + mask or native NA	**Training & evaluation** rows without y add little; missing labels bias your metrics

Missing labels in the training data: a selection problem for training and a trap for evaluation

Imputing the outcome y

Before missingness**After** mean imputation



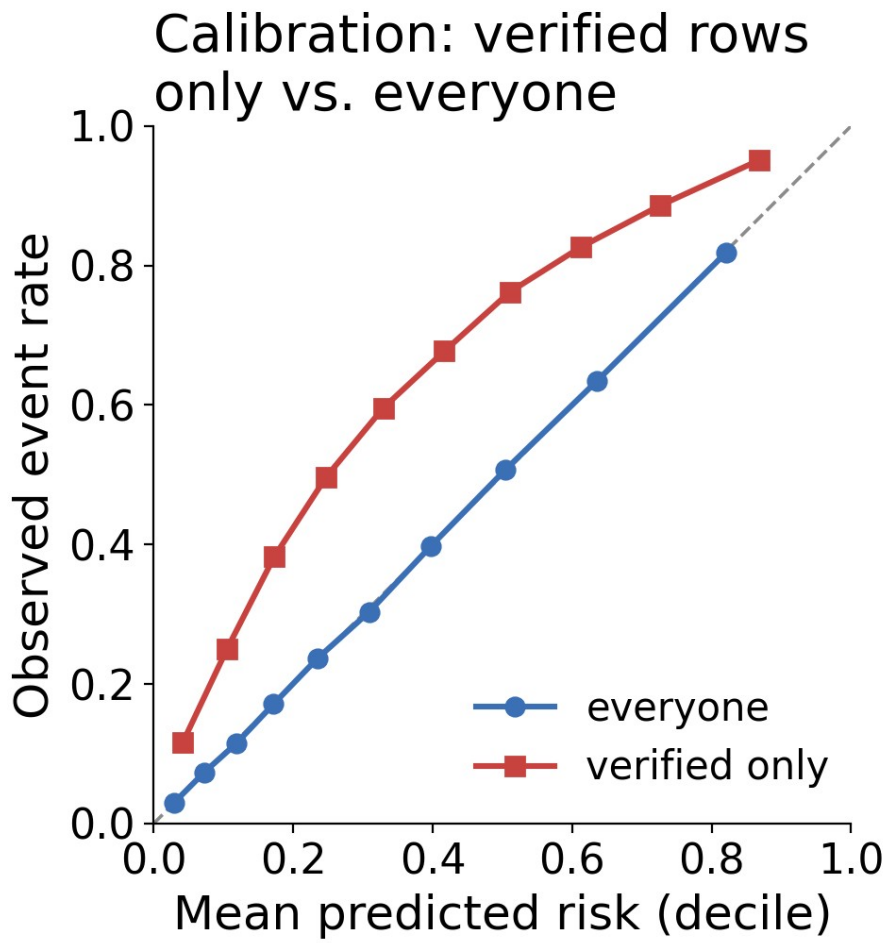


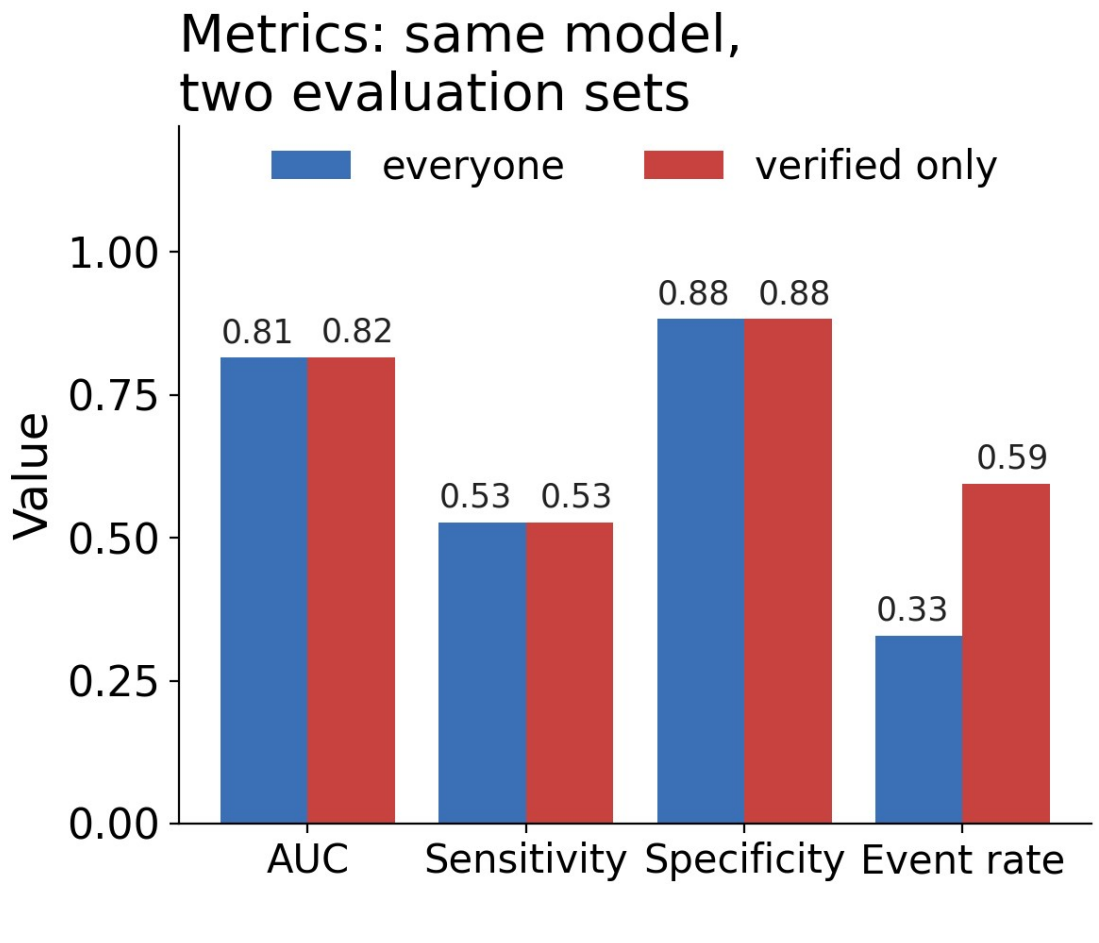
$$\text{var}(y) = 2$$
$$\text{MSE}(y|x) = 1$$

1. What is the **cross-validated MSE** of the best model after mean imputation?
2. What is the **true MSE** of this model for new data?

PREDICTION · missing y

Missing outcomes bias your evaluation

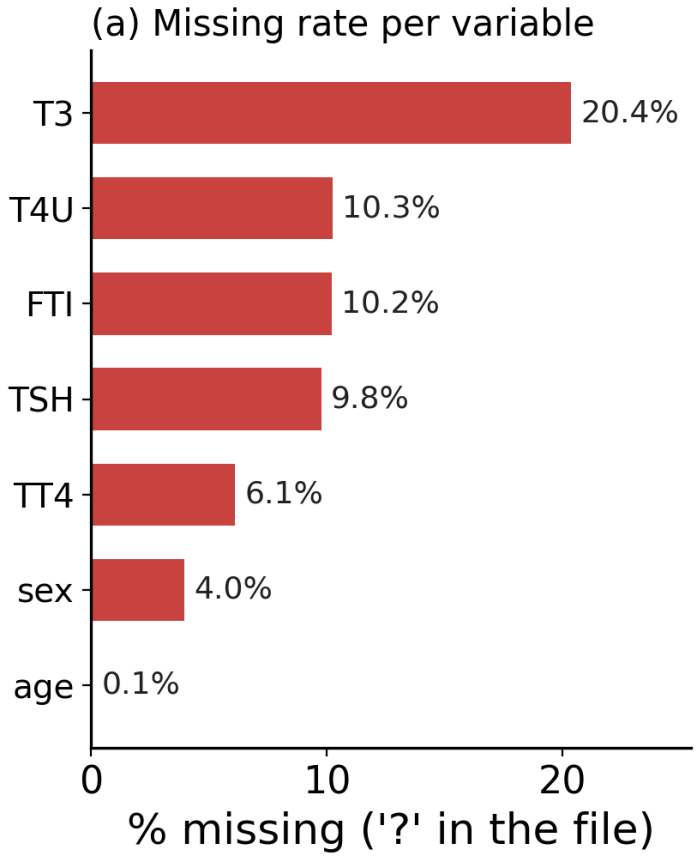


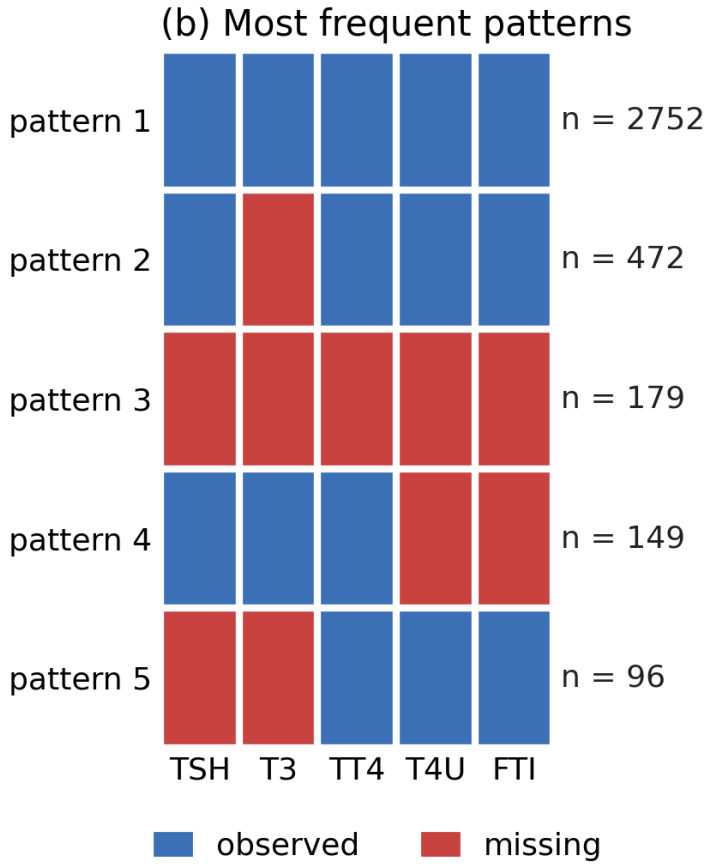


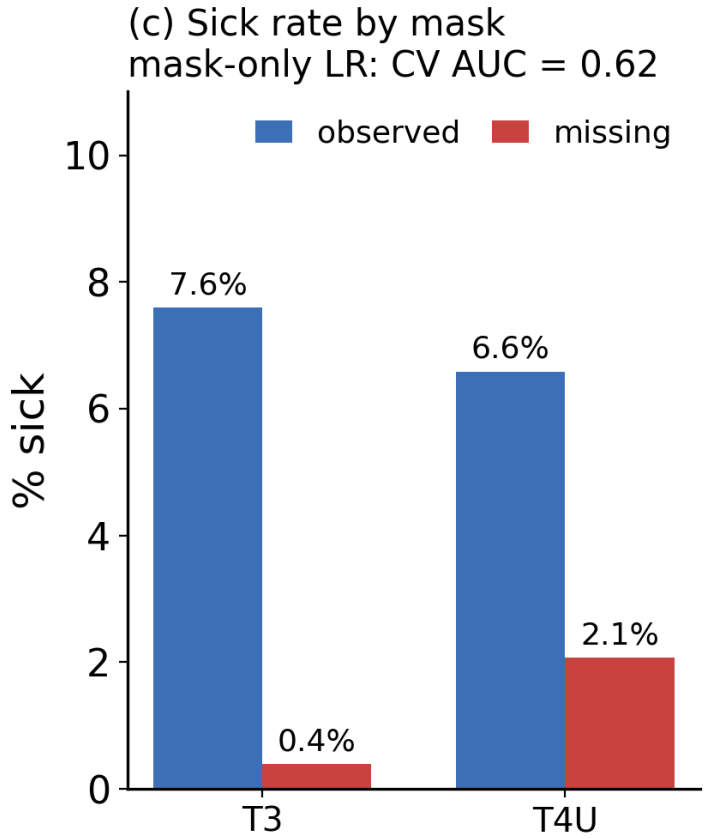
n = 200 000; outcome verified for 99,745 rows (prob. 0.9 if Y = 1; 0.3 if Y = 0). Sens. / spec. at threshold 0.5.

PREDICTION

Thyroid, steps 1–2: ideation & exploration



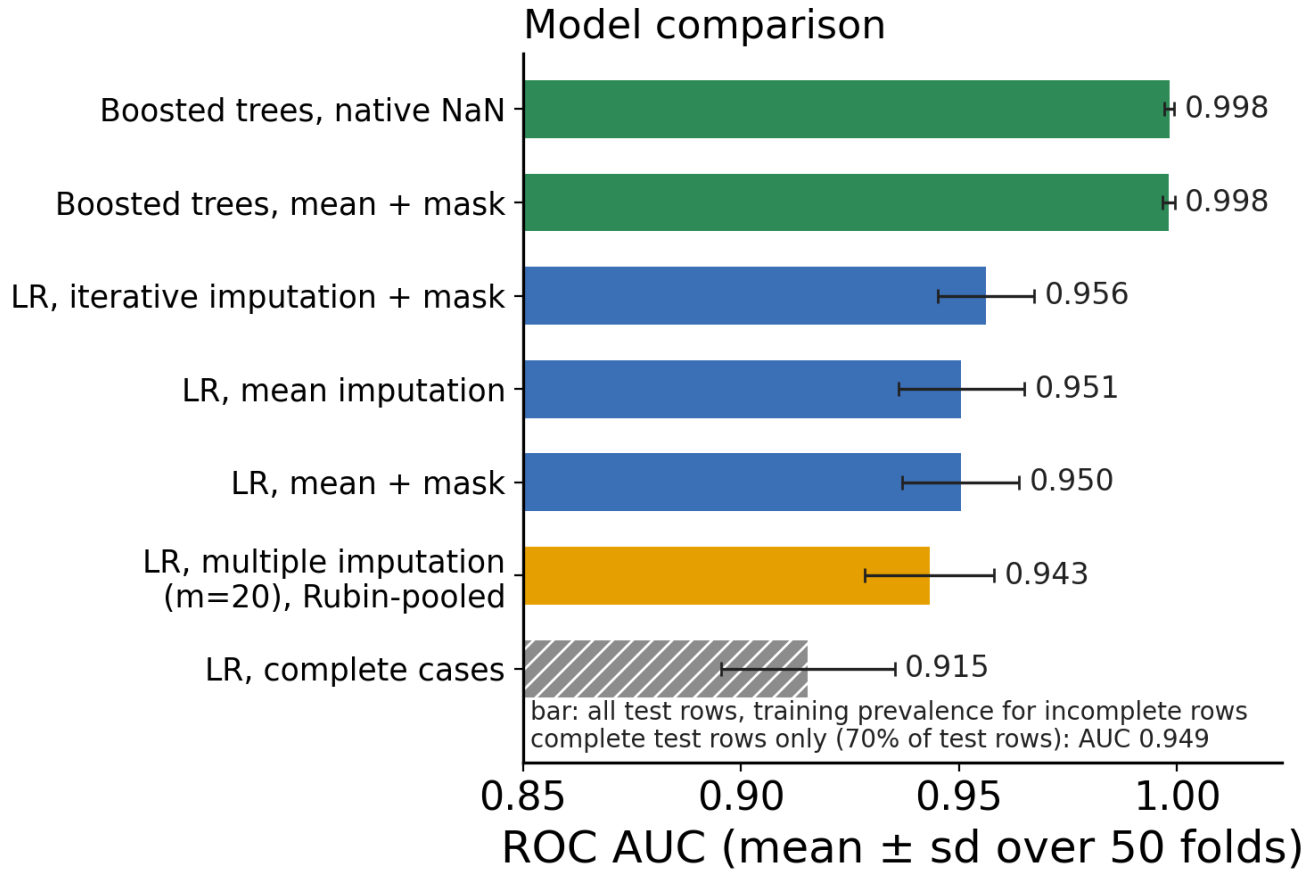


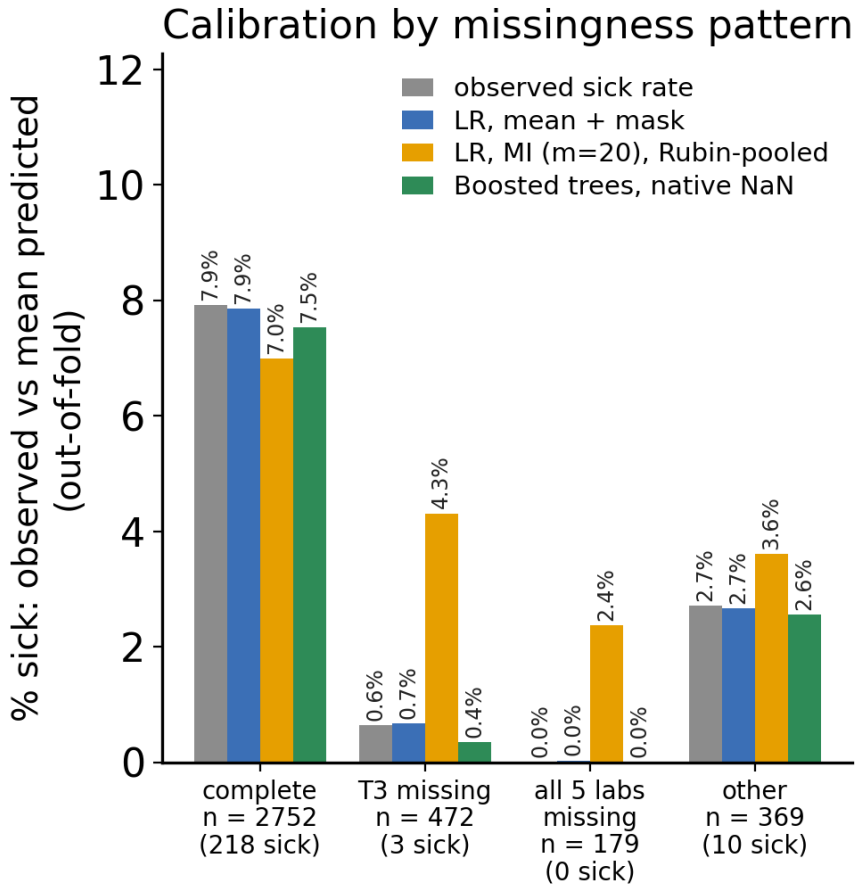


Thyroid 'sick', n = 3772 (OpenML 38). '?' read as NaN; *_measured columns (= the mask) dropped. Reproduce: thyroid_example.py

PREDICTION

Thyroid, steps 3–4: training & evaluation

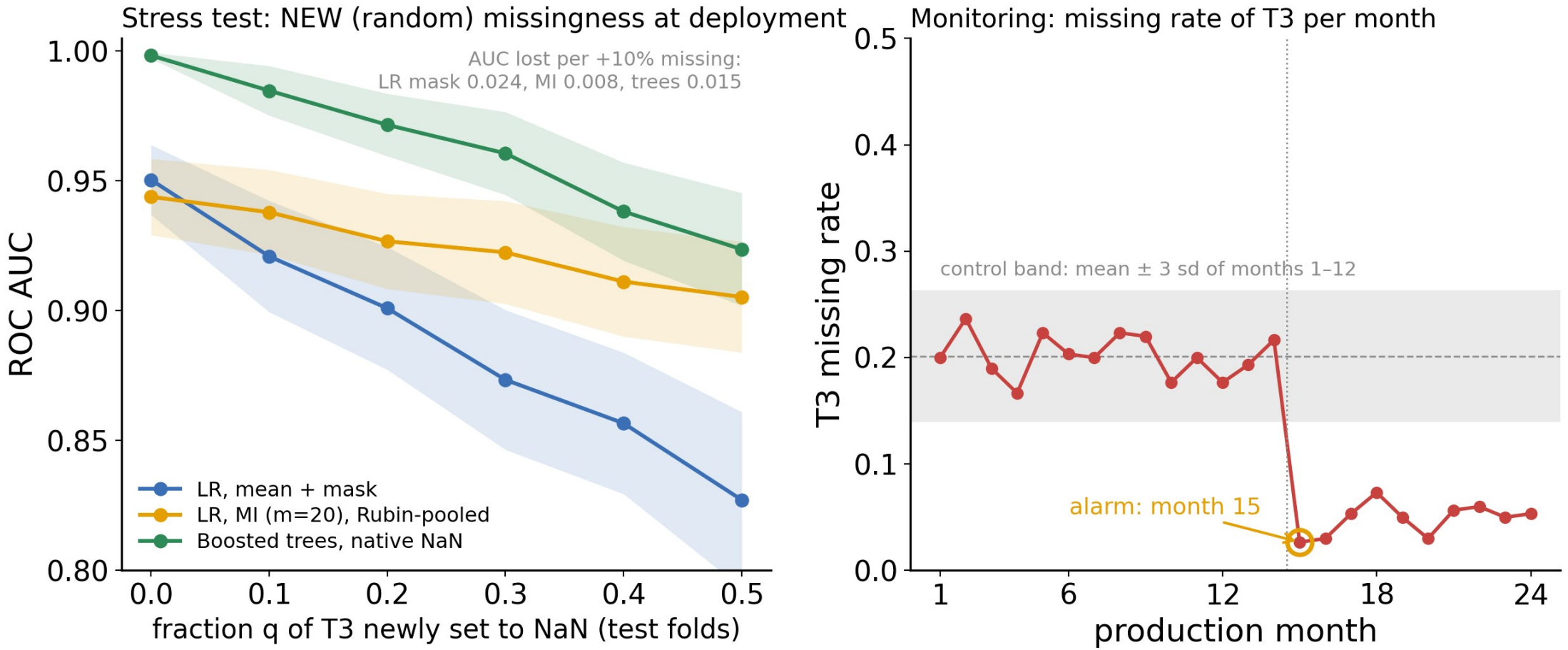




Thyroid 'sick' (OpenML 38), n = 3772, 6.1% sick. Stratified 5-fold CV × 10 repeats (random_state = 0); imputer, scaler and encoder are refit inside every training fold.
Right: mean out-of-fold predicted risk (averaged over the 10 repeats) vs observed sick rate; patterns over the 5 lab values.

PREDICTION

Thyroid, steps 4–6: stress, deploy, monitor



A pattern that was 20% of training data drops to 5%: the model's learned use of "T3 missing" no longer applies → retrain / re-validate.

Deployment: joblib.dump(pipeline.fit(X, y), "model.joblib") – imputer, mask and model ship as one object; log which inputs were missing per prediction.

PREDICTION

Defaults for a prediction project

1. Ideation

- Why is each variable ever missing?
- What is available at prediction time? Drop the rest
- Will the acquisition process stay the same?

2. Exploration

- Recode sentinel values (0, −999, ‘’) to NA
- Missing rate per variable; pattern table
- Does the mask alone predict y? (→ informative)

3. Training

- Boosted trees: pass NaN (native handling)
- Other learners: constant + mask; ‘missing’ level for categoricals
- Fancier imputation only if small n / linear model; inside the pipeline

4. Evaluation

- Imputer & scaler inside every CV fold
- Metrics and calibration per pattern
- Stress test: raise test-time missingness

5. Deployment

- Ship imputer + mask logic with the model (one Pipeline)
- Show which inputs were missing (and the wider uncertainty)

6. Monitoring

- Track missing rates & top patterns (control chart)
- New pattern at volume ⇒ re-validate, retrain

Conclusion

- Missing data are not just annoying – they bias means, trends, correlations and prediction models
- Use what you do know about what you don’t know: NDD / SDD / UDD is an assumption, not a test result
- Inference: single imputation is never enough – multiple imputation fixes the SE, but still assumes SDD
- Prediction: the target is E[y | X_obs, M] – missingness is a feature
  - constant + mask, or trees with native NA, is consistent and hard to beat; fancier imputation buys little
  - what hurts is missingness at prediction time – evaluate per pattern, stress-test, monitor
  - the mask is only as stable as the measurement process – watch for missingness shift
- Holes in the outcome: a selection problem for training and a trap for evaluation

References

Inference

van Buuren (2018). Flexible Imputation of Missing Data, 2nd ed. – stefvanbuuren.name/fimd

Rubin (1976). Inference and missing data. Biometrika. • Little & Rubin (2019). Statistical Analysis with Missing Data, 3rd ed. • Hand (2020). Dark Data.

Prediction

Josse, Chen, Prost, Varoquaux & Scornet (2024). On the consistency of supervised learning with missing values. Statistical Papers 65.

Le Morvan, Josse, Scornet & Varoquaux (2021). What's a good imputation to predict with missing values? NeurIPS.

Le Morvan & Varoquaux (2025). Imputation for prediction: beware of diminishing returns. ICLR.

Perez-Lebel, Varoquaux, Le Morvan, Josse & Poline (2022). Benchmarking missing-values approaches for predictive models on health databases. GigaScience.

Bertsimas, Delarue & Pauphilet (2024). Simple imputation rules for prediction with missing data. TMLR.

Shadbahr et al. (2023). The impact of imputation quality on ML classifiers for datasets with missing values. Communications Medicine.

Zaffran, Dieuleveut, Josse & Romano (2023). Conformal prediction with missing values. ICML.

Twala, Jones & Hand (2008). Good methods for coping with missing data in decision trees. Pattern Recognition Letters.

Sperrin, Martin, Sisk & Peek (2020). Missing data should be handled differently for prediction than for description or causal explanation. J Clin Epidemiol.

Sisk, Sperrin, Peek, van Smeden & Martin (2023). Imputation and missing indicators … development and deployment of clinical prediction models. SMMR.

Catoire, Genuer & Proust-Lima (2026). Prediction with missing data: target probabilities and missingness mechanisms. arXiv.

• Lakkaraju et al. (2017). The selective labels problem. KDD.