



From Linear Regression to Deep Networks

Evolution of models 01 | One dataset of 24 points fitted by mean, linear, polynomial, k-NN, tree, kernel smoother and a small MLP: what each step fixed, what it broke, and bias and variance measured at every step

#EvolutionOfModels #BiasVariance #Regression #Flexibility #SameData

Why one dataset, seven models

The flexibility ladder



We draw seven different curves through the same 24 dots. The first is a flat line; the last can wiggle anywhere. The best one is not the wiggliest: it is the one that guesses new dots best.

- train error always falls
- test error falls, then rises

Each step removes a limit of the previous model (cannot depend on x , cannot bend, bends everywhere at once, jagged, stepy) and adds a cost (variance, no extrapolation, hyperparameters).

- choose by held-out error
- a small test set is itself noisy



The ladder is a walk along the bias-variance trade-off and from global parametric to local non-parametric estimators; the MLP returns to a parametric form whose flexibility is set by width and training time.

- early stopping as regularisation
- inductive bias matters most at small n

The sheet's idea pitched to three readers

Must know

Mean squared error

average of squared residuals

Learn: machine-learning 07

Linear regression and least squares

Learn: machine-learning 02

k-NN, regression trees

Learn: machine-learning 04-05

Train / test split

Learn: machine-learning 08

Helpful

Bias-variance derivation

Learn: maths 14

MLP and backpropagation

Learn: deep-learning 01-02

Read a model ladder

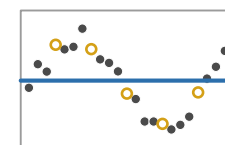
What this sheet recalls in one line and where each notion is taught

The toy problem

24 points, $x = 0, 0.5, \dots, 11.5$, generated as $y = 3 + 1.6 \sin(x / 1.6) + 0.05x + \text{Gaussian noise of standard deviation } 0.35$ (noise variance 0.12). Every 4th point (6 points) is held out as the test set; the other 18 train each model. The true curve rises, falls to a trough near $x = 7.5$ and rises again: one and a half waves.

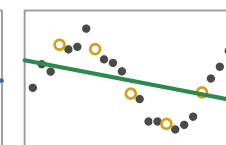
The ladder on the same data

1. 1 Mean



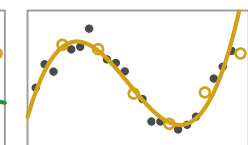
train 1.4 · test 1.32

2. 2 Linear



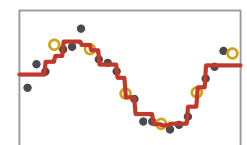
train 1.19 · test 1.24

3. 3 Cubic polynomial



train 0.102 · test 0.668

4. 4 k-NN, k = 3



train 0.096 · test 0.23

5. 5 Tree, depth 3



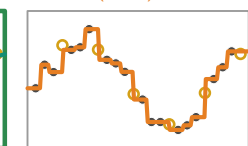
train 0.0439 · test 0.604

6. 6 Kernel, h = 0.6



train 0.0479 · test 0.0709

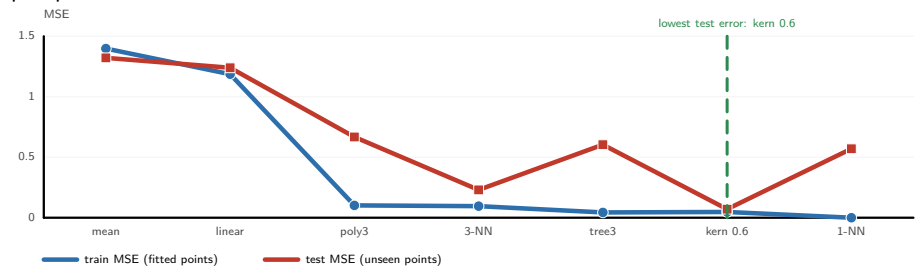
7. 7 1-NN (too far)



train 0 · test 0.57

● training point ○ test point (not used to fit)

Error per step: under-fit → over-fit



$MSE = (1/n) \sum (y_i - f(x_i))^2$ over the n points of a set; y_i : observed value, $f(x_i)$: the model's prediction at x_i . Train = the 18 fitted points, test = 6 unseen points (every 4th point held out).

Seven models, same 18 training points (dots) and 6 test points (rings). The renderer fits each model and computes train and test MSE; the summary marks the lowest test error

Step (era)	Previous limit	Idea	Train / test MSE	New limit
Mean (baseline)	none: the reference	predict the average y	1.40 / 1.32	ignores x entirely
Linear (least squares, 1805)	ignores x	$y = w x + b$	1.19 / 1.24	cannot bend
Polynomial, degree 3	cannot bend	add x^2, x^3 as features	0.10 / 0.67	global: one wave bends all; explodes outside the data
k-NN, k = 3 (1951)	global shape fixed in advance	average the 3 nearest y	0.10 / 0.23	jagged; flat outside data; slow at scale
Regression tree, depth 3 (CART 1984)	fixed neighbourhood size	split x where error drops most	0.04 / 0.60	piecewise constant steps; unstable
Kernel smoother, h = 0.6 (1964)	hard steps and edges	Gaussian-weighted average	0.05 / 0.07	bandwidth to tune; curse of dimensionality
MLP 1-16-1 tanh, 300 Adam steps (1986)	hand-picked basis or distance	learn the basis functions	approx. 0.04-0.13 / 0.09-0.15	needs tuning, data and early stopping
Same MLP, 20 000 steps	under-trained	train longer	approx. 0.00-0.02 / 0.24-0.53	memorises noise

Each step: the limit of the previous model, the idea that removed it, the new limit. MSE values as computed above (train / test); MLP from our own run, 3 random seeds

⚠ Read the MLP rows correctly

On 18 points the small MLP does not beat the kernel smoother: its advantage is that it learns its own features, which pays off with many inputs, many examples and structure a distance cannot capture (images, text). With little 1-D data, the model with the right inductive bias (local smoothness) wins. Sheet 08 refits the ladder on 4 times more data.

● Choosing a step, worked

Which model would you ship, and how sure are you?

PROBLEM

From the ladder above, pick a model and say whether its test MSE of 0.07 is believable.

1. Rank by test MSE

⇒kernel 0.07 < 3-NN 0.23 < tree 0.60 < cubic 0.67

why: held-out error estimates error on new data

2. Compare with the noise floor

⇒0.07 < 0.12

why: below the floor means the 6 test points happened to lie close to the curve

3. Estimate the uncertainty

⇒with n = 6, one squared residual of 0.25 moves the mean by 0.04

why: a tiny test set gives a noisy score

4. Check robustness

⇒kernel h = 0.3 → 0.10, h = 1.0 → 0.09; degree-5 polynomial → 0.09

why: neighbours on the ladder agree within noise

GIVEN

- test set n = 6 points
- noise variance $\sigma^2 = 0.35^2 = 0.12$: the best possible expected MSE
- kernel h = 0.6: test MSE 0.07; 3-NN: 0.23; cubic: 0.67

ANSWER

Ship a smooth local model (kernel h ≈ 0.6) or a degree-5 polynomial, but report the expected error as ≈ 0.12-0.2, not 0.07: use cross-validation on all 24 points for the final estimate

Check: Expected test MSE of 3-NN from the simulation below: 0.07 + 0.04 + 0.12 = 0.23, matching its measured 0.23

Under-fitting

train and test both high (mean, linear)

Tell: Is the training error already low?

vs

Over-fitting

train ≈ 0, test high (1-NN, long MLP training)

Global model

polynomial: every coefficient moves the whole curve

Tell: Does a far point change the prediction here?

vs

Local model

k-NN, tree, kernel: only nearby points matter

Interpolation

predict inside 0-11.5

Tell: Is x inside the training range? Local models go flat, polynomials diverge

vs

Extrapolation

predict at x = 15

Hand-chosen basis

x^2, x^3 or a distance, picked by you

Tell: Who decided the features: you or the optimiser?

vs

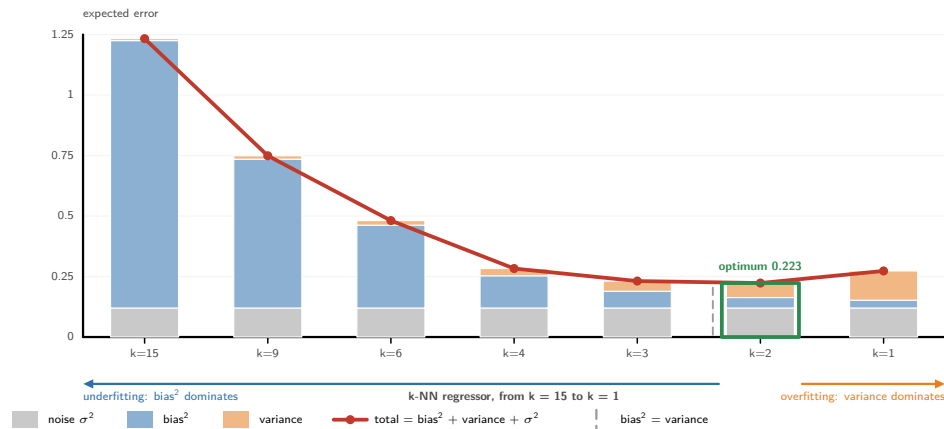
Learned basis

16 tanh units placed by gradient descent

Pairs this sheet keeps contrasting

Tadata | tadata.fr | contact@tadata.codes

⌵ Bias and variance at each step



k-NN regressor, from k = 15 to k = 1	k=15	k=9	k=6	k=4	k=3	k=2	k=1
bias ²	1.1	0.615	0.342	0.133	0.069	0.043	0.032
variance	0.008	0.014	0.019	0.03	0.042	0.06	0.121
noise σ^2	0.12	0.12	0.12	0.12	0.12	0.12	0.12
total	1.23	0.749	0.481	0.283	0.231	0.223	0.273

Optimum at k-NN regressor, from k = 15 to k = 1 k=2: total 0.223 = bias² 0.043 + variance 0.06 + noise 0.12. Bias² and variance cross between k=3 and k=2.

$$E[(y - \hat{f}(x))^2] = \text{Bias}[\hat{f}(x)]^2 + \text{Var}[\hat{f}(x)] + \sigma^2$$

$$\text{Bias}[\hat{f}(x)] = E[\hat{f}(x)] - f(x)$$

$$\text{Var}[\hat{f}(x)] = E[(\hat{f}(x) - E[\hat{f}(x)])^2]$$

where

y observed target = f(x) + ϵ

f true function

$\hat{f}$ model fitted on a random training set

E average over training sets (and noise)

σ^2 noise variance: irreducible error, no model removes it

ϵ noise with mean 0 and variance σ^2

Measured by simulation on this toy problem: 400 fresh training sets of the 18 x positions, k-NN refitted each time, bias² and variance averaged over x in 0-11.5. Noise 0.12 is fixed. Smaller k = more flexible

Expected test error at one step

FORMULA **E = B + V + N**

WHERE

- E = expected squared error on a new point, averaged over training sets (y units squared)
- B = bias²: squared gap between average prediction and true curve (y units squared)
- V = variance of the prediction across training sets (y units squared)
- N = noise variance σ^2 of y around the true curve (y units squared)

WORKED

B = 0.069
V = 0.042
N = 0.12
E = 0.069 + 0.042 + 0.12
E = 0.231 y units squared

RESULT E = 0.231 y units squared

READ IT For 3-NN on this problem: 0.23, close to the measured test MSE. The optimum (k = 3 or 4) is where B stops falling faster than V rises.

MISTAKES

- Reading test MSE on 6 points as E itself: it is one noisy sample of E
- Thinking N can be reduced by a better model: only better inputs (features) reduce it

Polynomial degree	Bias ²	Variance	Expected error (+ noise 0.12)
0 (mean)	1.04	0.01	1.17
1 (linear)	0.94	0.01	1.07
2	0.75	0.02	0.89
3	0.09	0.03	0.24
4	0.04	0.04	0.20
5	0.00	0.05	0.18

The same simulation for the polynomial family: bias collapses at degree 3, the shape of one and a half waves

💡 The pattern behind the whole ladder

Moving up a step lowers bias² and raises variance. A step is worth it while bias² falls more than variance rises. Regularisation (ridge, pruning, larger k, larger h, early stopping) moves a model back down its own ladder without changing family.

💡 Myths and recap

MYTH

✗ The most flexible model always wins

FACT

✓ It wins only while variance stays below the bias it removes
1-NN reaches train MSE 0 and test MSE 0.57 here, worse than 3-NN (0.23).

✗ Deep networks beat everything, even on small data

✓ With 18 points the kernel smoother beat the MLP
Learned features need data; a correct inductive bias wins at small n.

✗ Polynomials are a safe default

✓ They are global and explode outside the data
Degree 5 fits 0-11.5 well and diverges quickly beyond it.

✗ Zero training error means a perfect model

✓ It often means memorised noise
Noise variance is 0.12: a model with train MSE 0 has fitted that noise.

Remember

01

Each step removes a limit and adds a cost

cannot bend → bends globally → local but jagged → smooth → learned

02

Train error always falls up the ladder

test error falls, then rises

03

$E = \text{bias}^2 + \text{variance} + \text{noise}$

optimum where the sum is lowest

04

Local models cannot extrapolate

polynomials extrapolate wildly

05

Learned features need data

an MLP pays off at scale, not on 18 points

Q1 What limit of the linear model does the polynomial remove, and what new limit does it add?

A It can bend; but it is global (one region bends the whole curve) and diverges outside the data.

Q2 Why did k-NN improve on the cubic polynomial here?

A It is local: no fixed global shape, so it follows one and a half waves (test MSE 0.23 vs 0.67).

Q3 What does a regression tree add over k-NN, and at what cost?

A It chooses where to split from the data (adaptive neighbourhoods); predictions are piecewise constant steps and unstable.

Q4 What does the kernel smoother fix in k-NN and trees?

A Hard edges: it weights all points smoothly by distance, giving a smooth curve.

Q5 What does an MLP learn that the earlier models were given?

A Its basis functions (features); earlier models used a fixed basis (x^k) or a fixed distance.

Q6 1-NN: train MSE and why?

A 0: each training point is its own nearest neighbour, so the noise is memorised.

Q7 Expected test error of 3-NN from bias² 0.069, variance 0.042, noise 0.12?

A 0.23 (sum of the three).

Q8 Test MSE 0.07 below the noise variance 0.12: what does it tell you?

A The 6-point test set is too small; the estimate is noisy. Use cross-validation.

Q9 How do you move a model back down its own ladder?

A Regularise: larger k or h, pruning, ridge penalty, early stopping.