Last time: $MSE_{train} = \frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2$ $\hat{y}_n = \hat{f}(x_n)$

training data: $\{(y_n, x_n)\}_{n=1}^{N}$

→ fit $\hat{f}$

What do I care about when evaluating a method.

Things I don't care about:

performance of $\hat{f}$ on the train data

↳ I already know the answer! y_n

Things I care about:

performance of $\hat{f}$ on new data I haven't seen before.

↳ I don't know the answer

Defn:

training data: data fed to approach to create $\hat{f}$ ← function of train data

$\{(x_n, y_n)\}_{n=1}^{N}$

test data: data from the same data generating process but that wasn't used to "train" $\hat{f}$.

$\{(x_n^*, y_n^*)\}_{n=1}^{N^*}$

Scheme for evaluation of $\hat{f}$

1. use training data to train $\hat{f}$
2. evaluate $\hat{f}$ on the test data

double up on metrics

RSS_{train}	RSS_{test}
MSE_{train}	MSE_{test}
$RMSE_{train}$	$RMSE_{test}$
calc. on training	replace training data w/ test

Why? Evaluating on training data is potentially misleading. (typically)

Intuition: Circular argument.

(*) $\hat{f} = \underset{f}{\operatorname{argmin}}\ \mathbb{E}[L(f)]$

$\hat{f}$ is chosen to minimize the expected (empirical) risk.

I train $\hat{f}$, by finding a $\hat{f}$ that predicts my

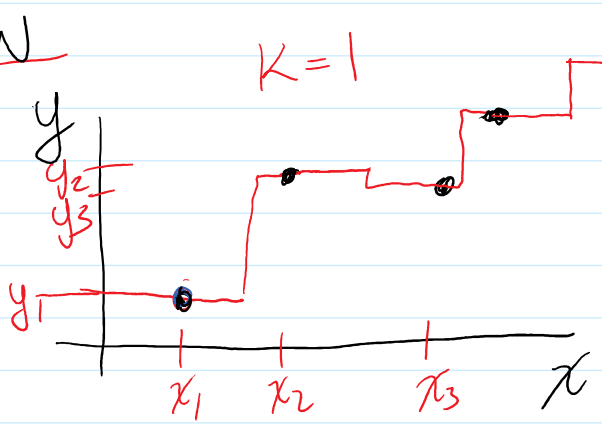
training data well.

So.... $\hat{f}$ predicts traing data well.

Result: traing metrics are too optmistic an performace of $\hat{f}$ on new/independent data

i.e. $RSS_{train} < RSS_{test}$

Ex. KNN



What value of K should I choose?

Bad way to decide: choose the value of K to minimize the trainig RSS

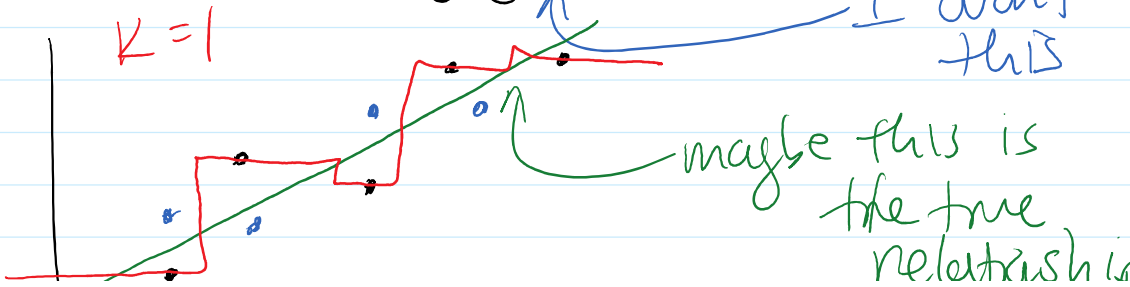
→ Q: what value of K minimizes RSS_{train}?

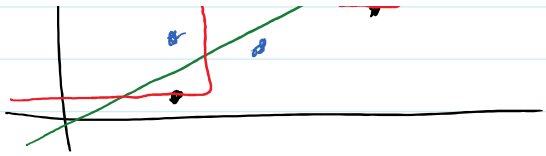
K=1

$$RSS_{train} = \sum_{n=1}^{N} (\hat{y}_n - y_n)^2 = 0$$

b/c $\hat{y}_n = y_n$

What is **true underlying model**?



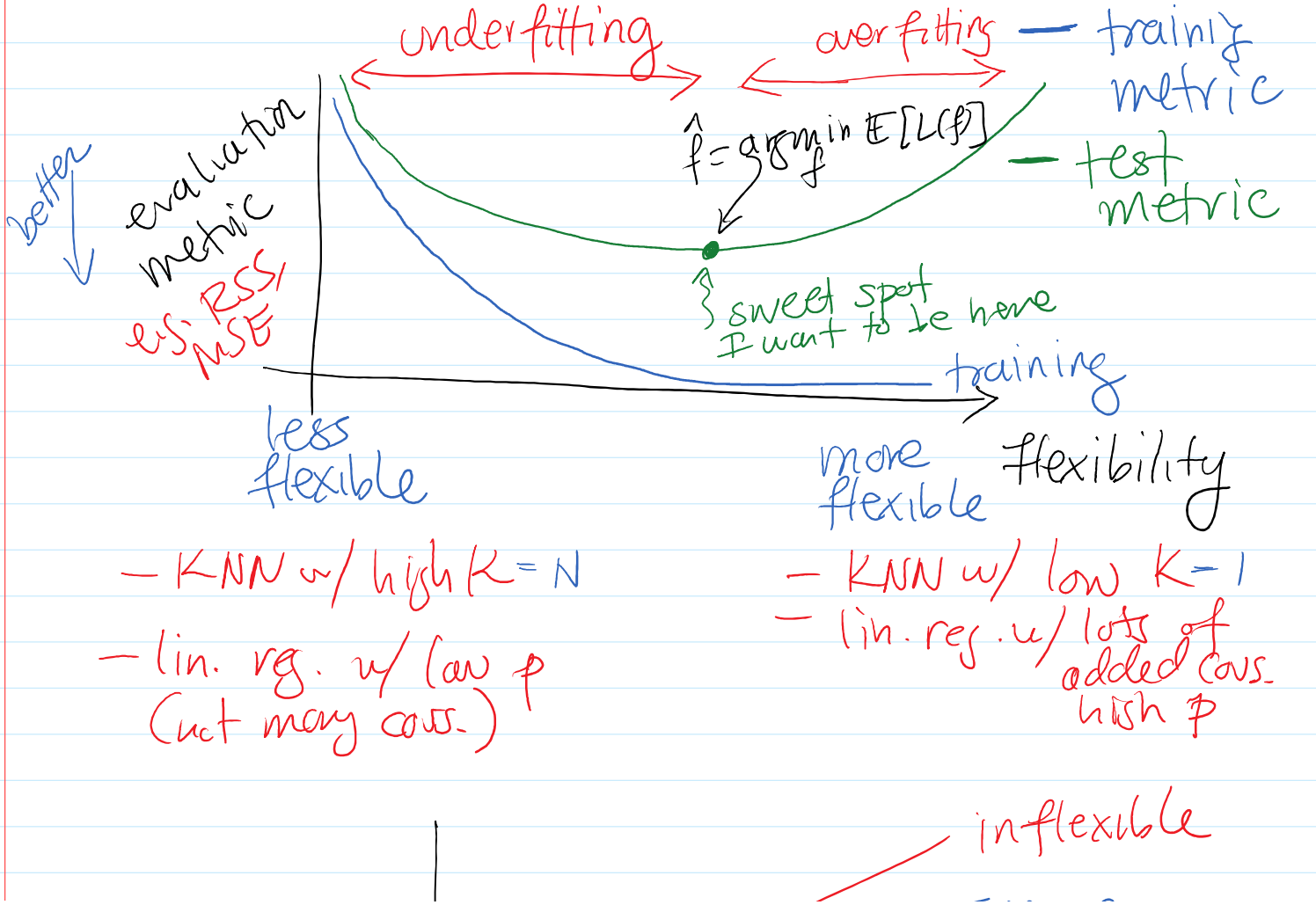


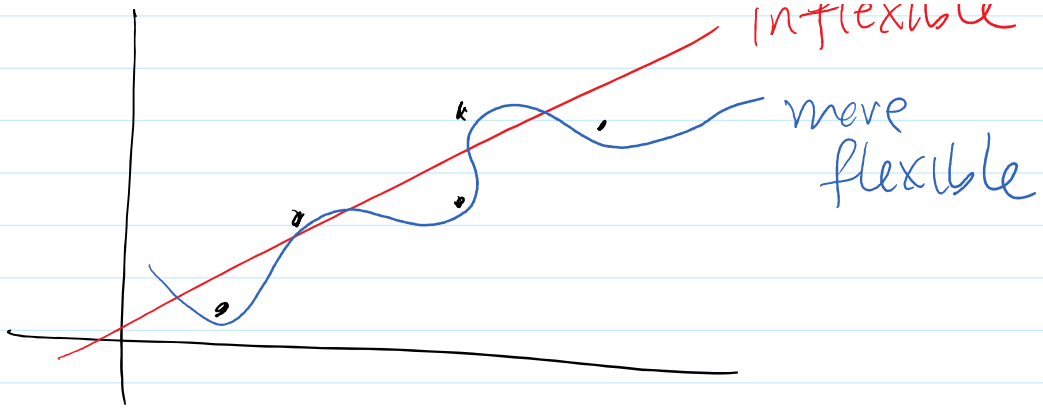
the true relationship

I'm not following the true underlying trend,
I'm chasing the noise.

When I generate new data:
it won't follow noise, but the underlying trend.

We call this overfitting.





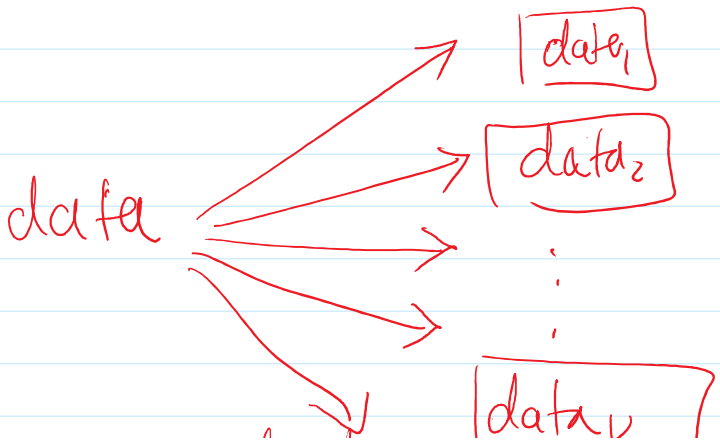
How do we solve this in real life?

I get some data set:
→ want to use to make $\hat{f}$ to predict well

Validation Set: test/train split

data → p% → traing data (fit)
typically $p \approx 50\%$
→ (1-p)% → test data (evaluate) 90%
split (randomly)

Cross-Validation: K-fold cross-validation



→ cycle through folds:

for k=1,..,K:

① train on all folds but

randomly split in "folds"

data$_K$

(1) train on all folds but k^{th}

(2) evaluate on k^{th} fold

→ after this I get K metrics

$m_1, m_2, \ldots, m_K$

Summarize into a final metric

e.g. $m = \text{median}(m_1, \ldots, m_K)$.