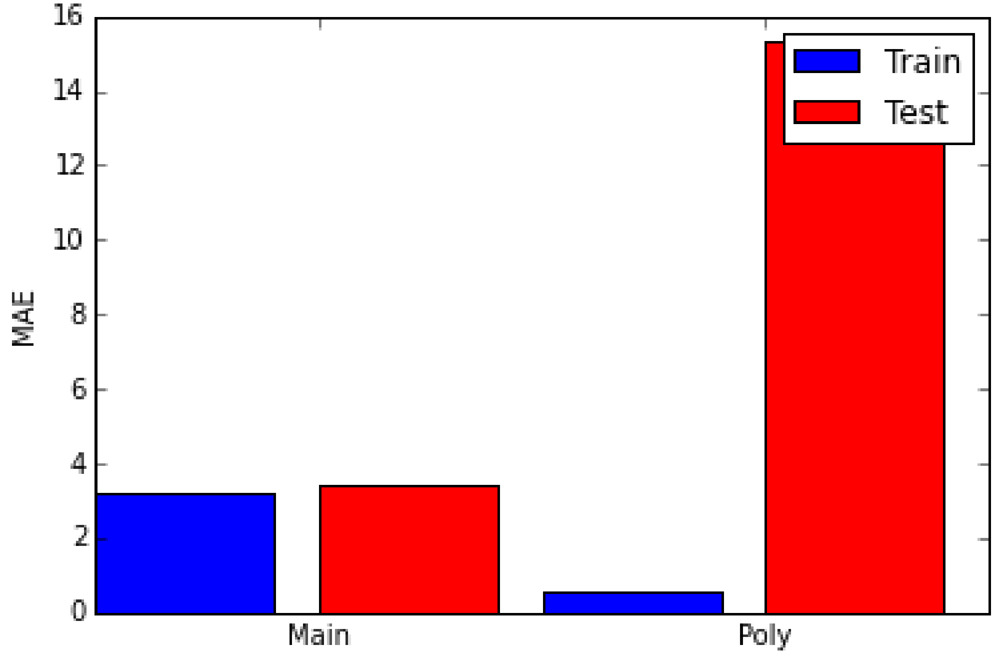
Over/under fitting – Hyperparameters

Grzegorz Chrupała
@gchrupala

Boston house prices

```
- CRIM     per capita crime rate by town
- ZN       proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS    proportion of non-retail business acres per town
- CHAS     Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX      nitric oxides concentration (parts per 10 million)
- RM       average number of rooms per dwelling
- AGE      proportion of owner-occupied units built prior to 1940
- DIS      weighted distances to five Boston employment centres
- RAD      index of accessibility to radial highways
- TAX      full-value property-tax rate per $10,000
- PTRATIO  pupil-teacher ratio by town
- B        1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT    % lower status of the population
- MEDV     Median value of owner-occupied homes in $1000's
```

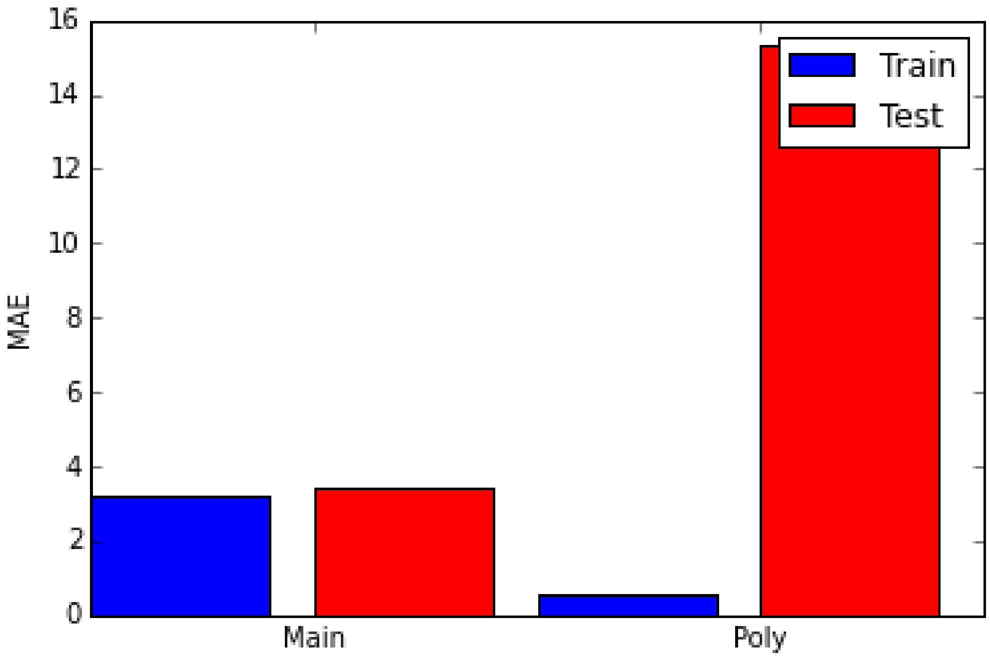
Two linear regression models



Overfitting

- Fit traing data very closely
- Able to predict very well on training data
- Doesn’t generalize well
  - High error on validation/test data

Boston house prices



- Main effects
  - [a b c]
- Interactions
  - [a b c ab ac abc]

- Always easier to fit (and overfit) with more features

Overfitting

- Fit traing data very closely
- Able to predict very well on training data
- Doesn’t generalize well
  - High error on validation/test data

How can we control fit?

- Features
  - More features – more fit
  - Fewer features – less fit
- Model hyperparameters

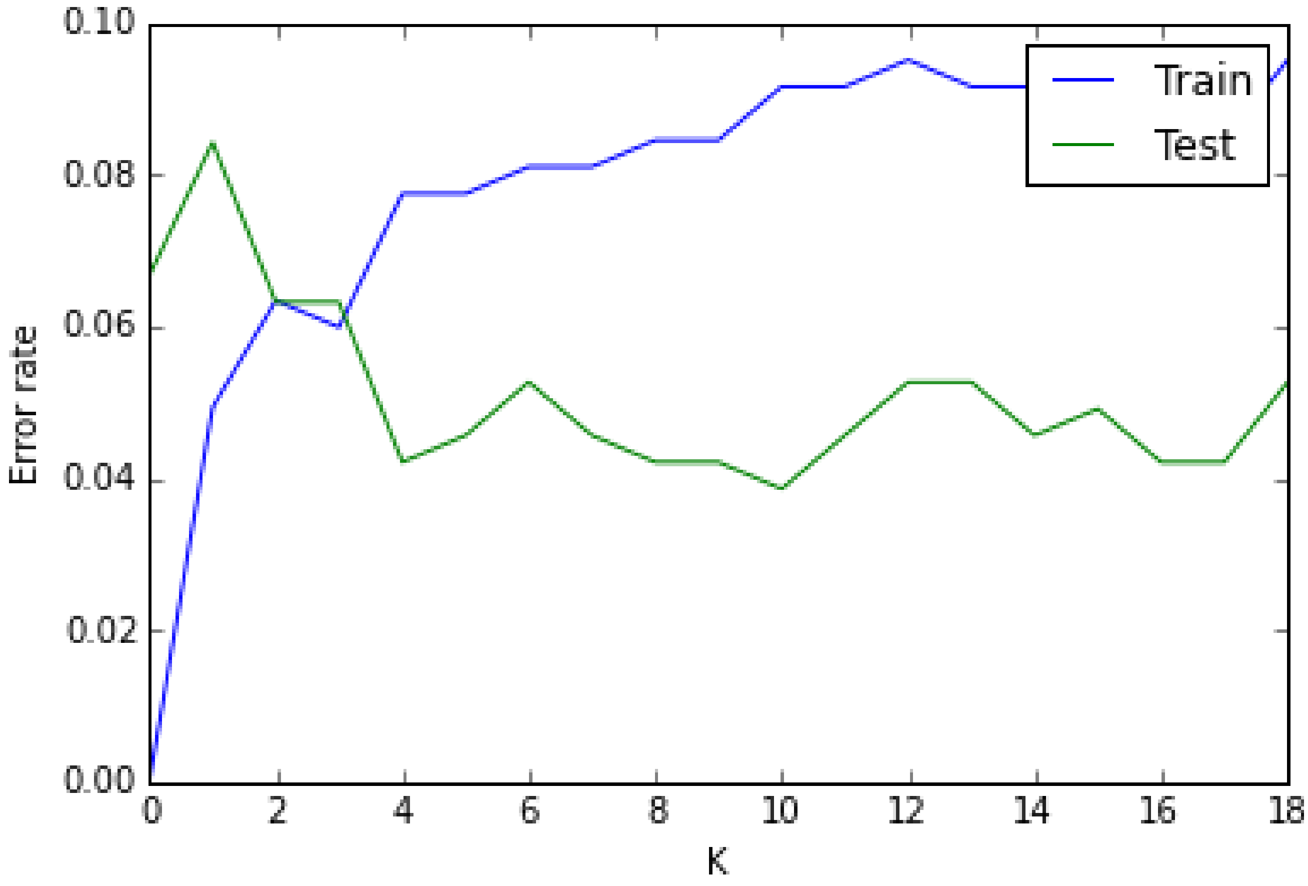
K-NN

- How can we control fit in K-NN?
- K
  - Smaller K – more fit
  - Larger K – less fit

Breast cancer diagnostic dataset

```
- radius (mean of distances from center to points on the perimeter)
- texture (standard deviation of gray-scale values)
- perimeter
- area
- smoothness (local variation in radius lengths)
- compactness (perimeter^2 / area - 1.0)
- concavity (severity of concave portions of the contour)
- concave points (number of concave portions of the contour)
- symmetry
- fractal dimension ("coastline approximation" - 1)
```

Train and test error as a function of K

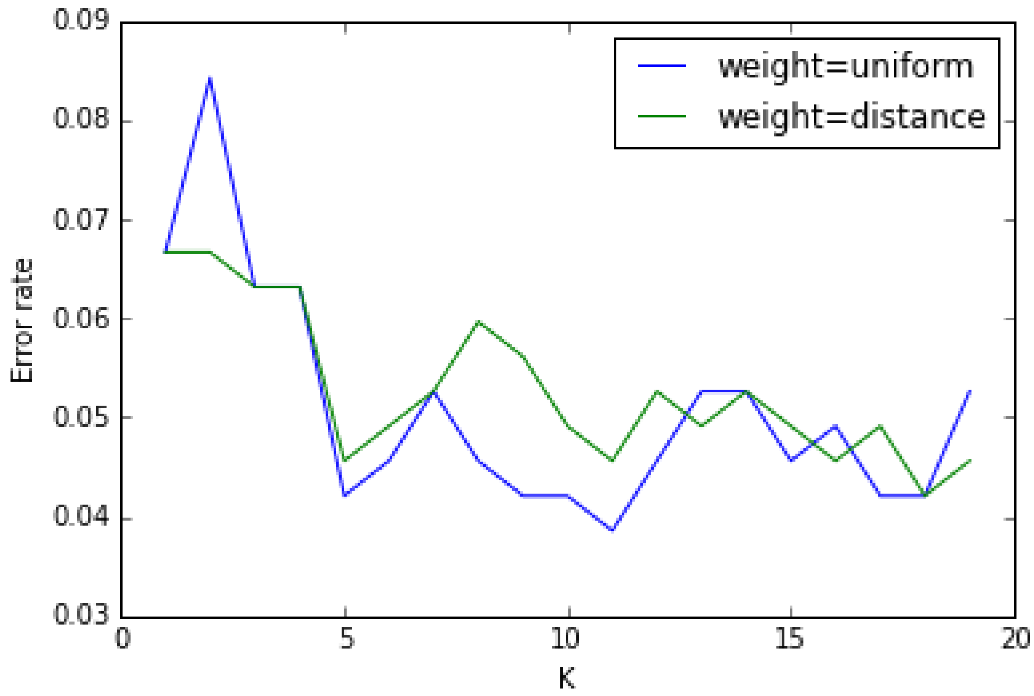


Hyperparameter

- **Hyperparameter**
  - variable part of the model which isn’t set during learning on training data
- Needs to be tuned on a validation set
- K is a hyperparameter of KNN

Tuning multiple hyperparameters

$$\text{weight} \in \{\text{uniform}, \text{distance}\}$$
$$K \in \{1, \ldots, 19\}$$



Grid search

- Systematic search for best hyperparameter settings
  - Choose values to for each hyperparam
  - Check validation error for all combinations
- Lots of computation!

K-NN hyperparams

- K
- Neighbor weights
- Distance metric

Linear regression

- How can we control fit for Linear Regression?
- We can add/remove/combine features
- Hyperparameters?

Ridge Regression

- Ridge regression
  - Fit the data, while keeping coefficients small
  - Hyperparameter controls how small

Standard Linear Regression

- Minimize Sum Squared Error
  - or Mean Squared Error, which gives the same result

$$SSE(f) = \sum_{i=1}^{N} (y^{(i)} - f(x^{(i)}))^2$$

Ridge Regression

$$\sum_{i=1}^{N}(y^{(i)} - f(x^{(i)}))^2 + \alpha \sum_{j=1}^{M} w_j^2$$

where **w** are the coefficients of the regression

Ridge Regression

$$\sum_{i=1}^{N}(y^{(i)} - f(x^{(i)}))^2 + \alpha \sum_{j=1}^{M} w_j^2$$

make model fit

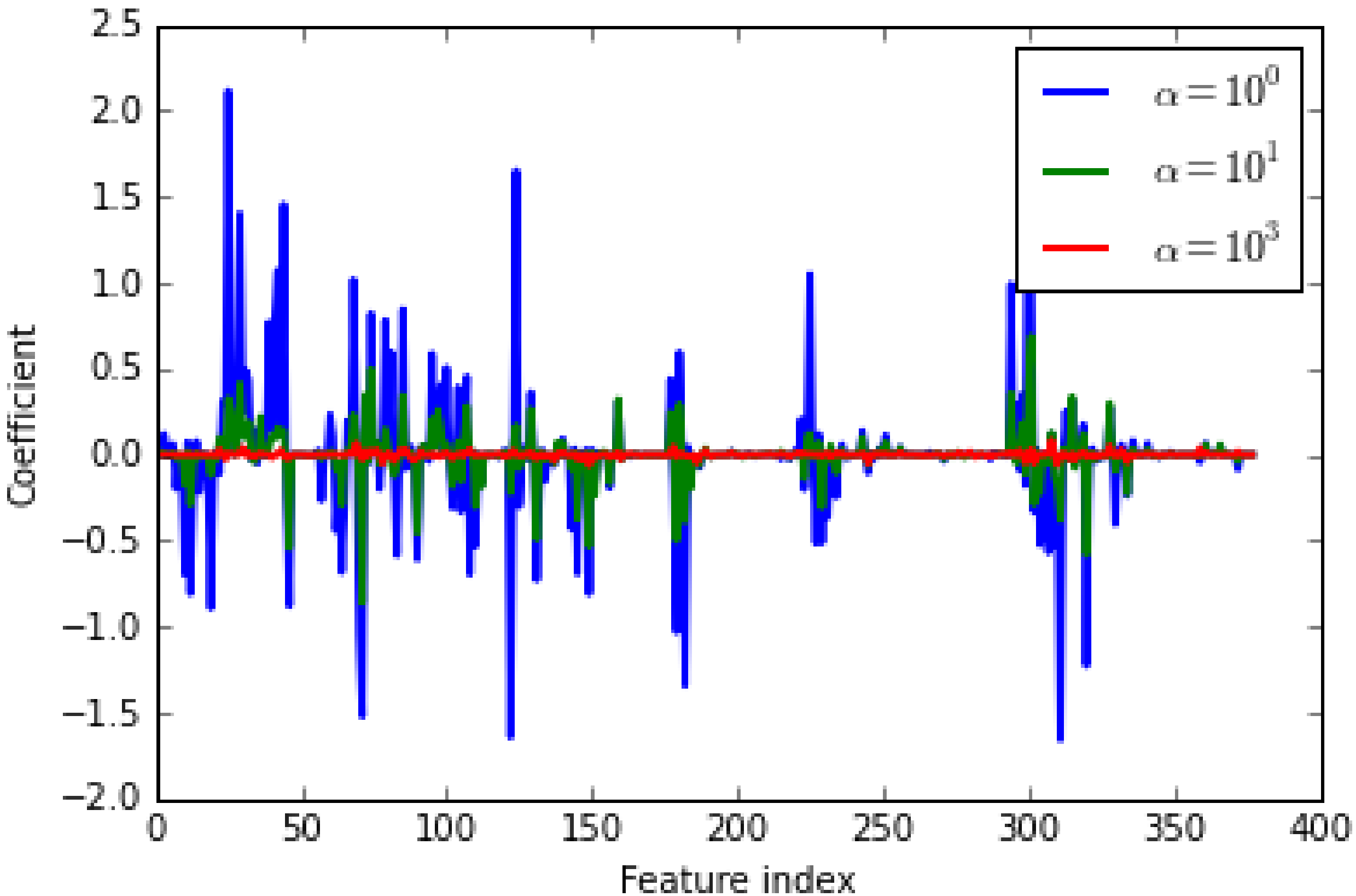
trade-off

make model simple

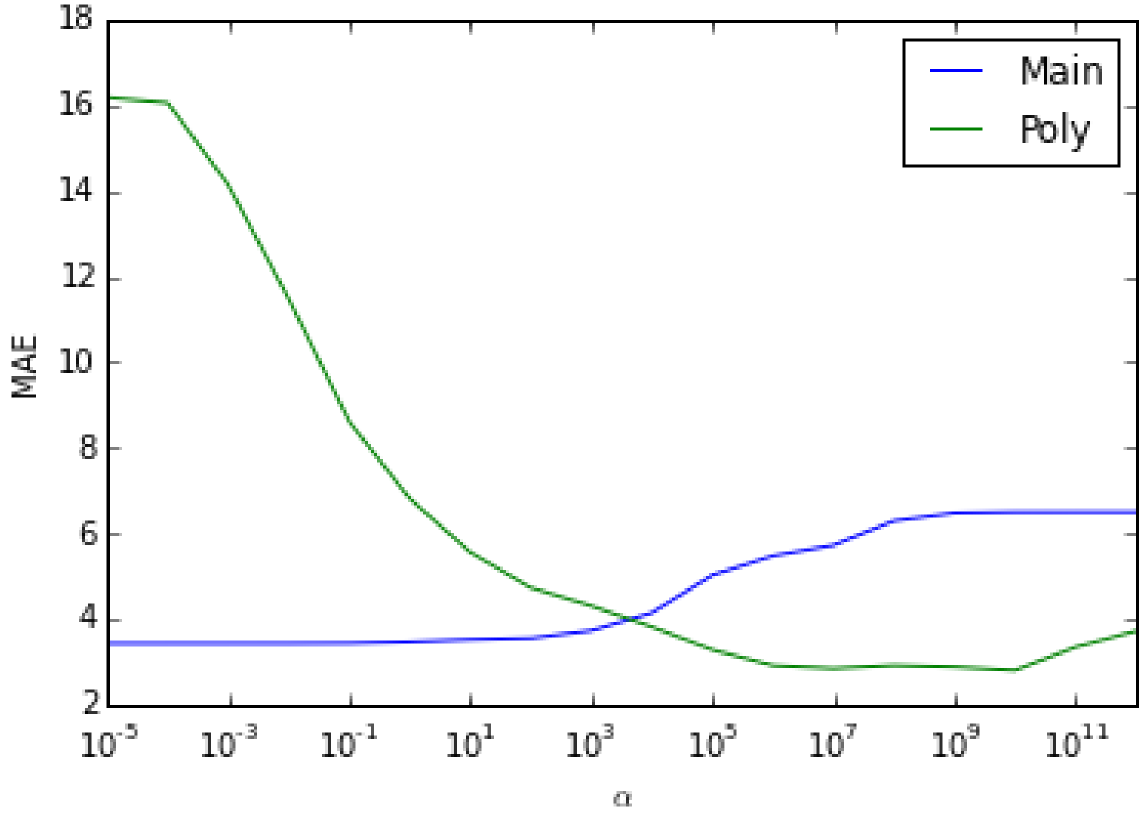
Role of alpha

- Alpha controls the strength of the penalty
- With stronger penalty, model tries to fit data less

Weight variance vs alpha



Effect of alpha



House prices dataset: two feature sets

Take aways?

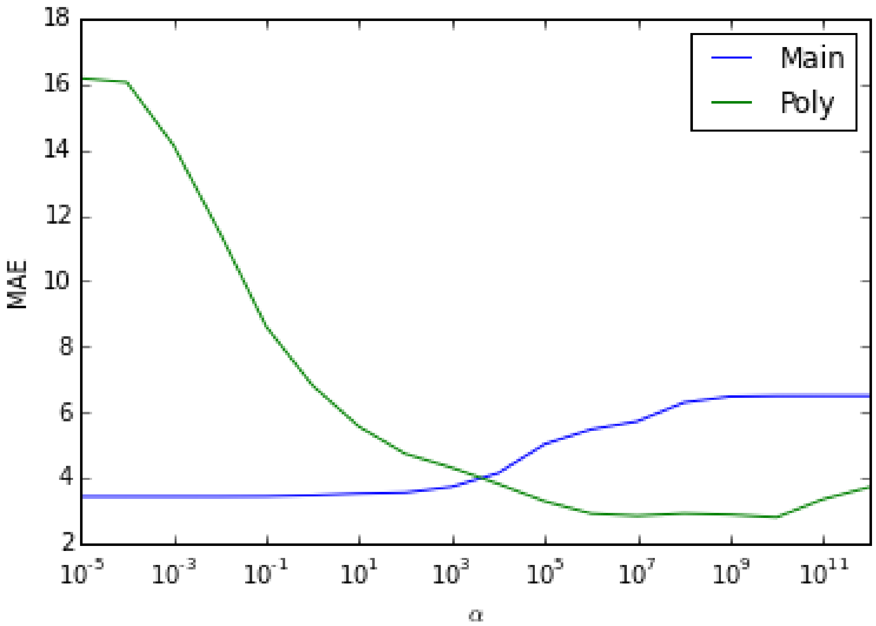
Take aways?

- Vary alpha **exponentially**

$$\alpha \in \{10^{-10}, \ldots, 10^{10}\}$$

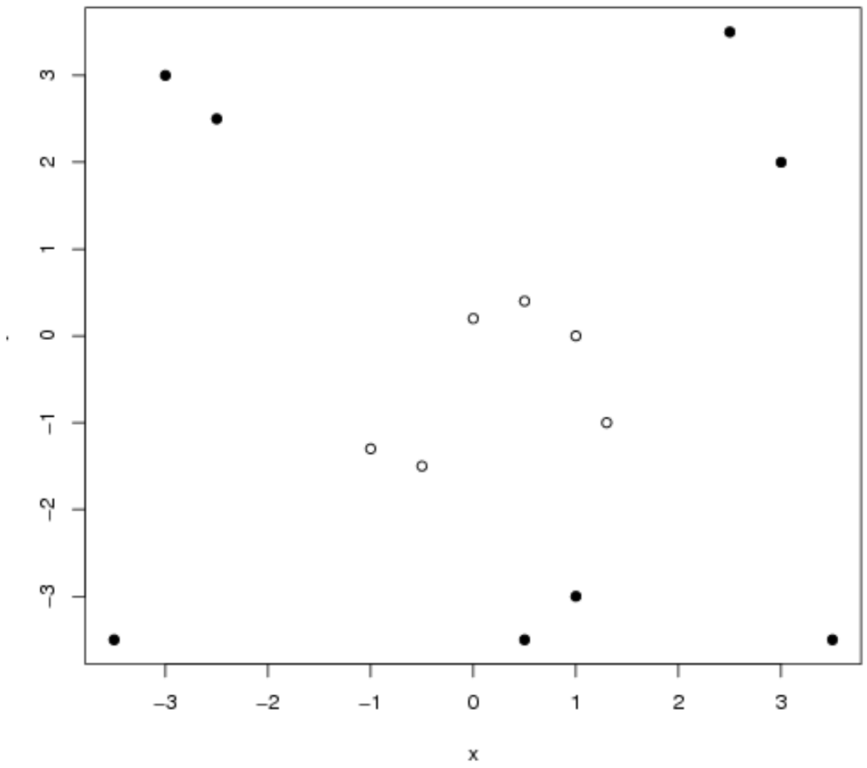
- Optimal alpha depends on feature set
- Larger number of features
  - → need to use larger alpha
- Optimize alpha **separately** for each variation of feature set

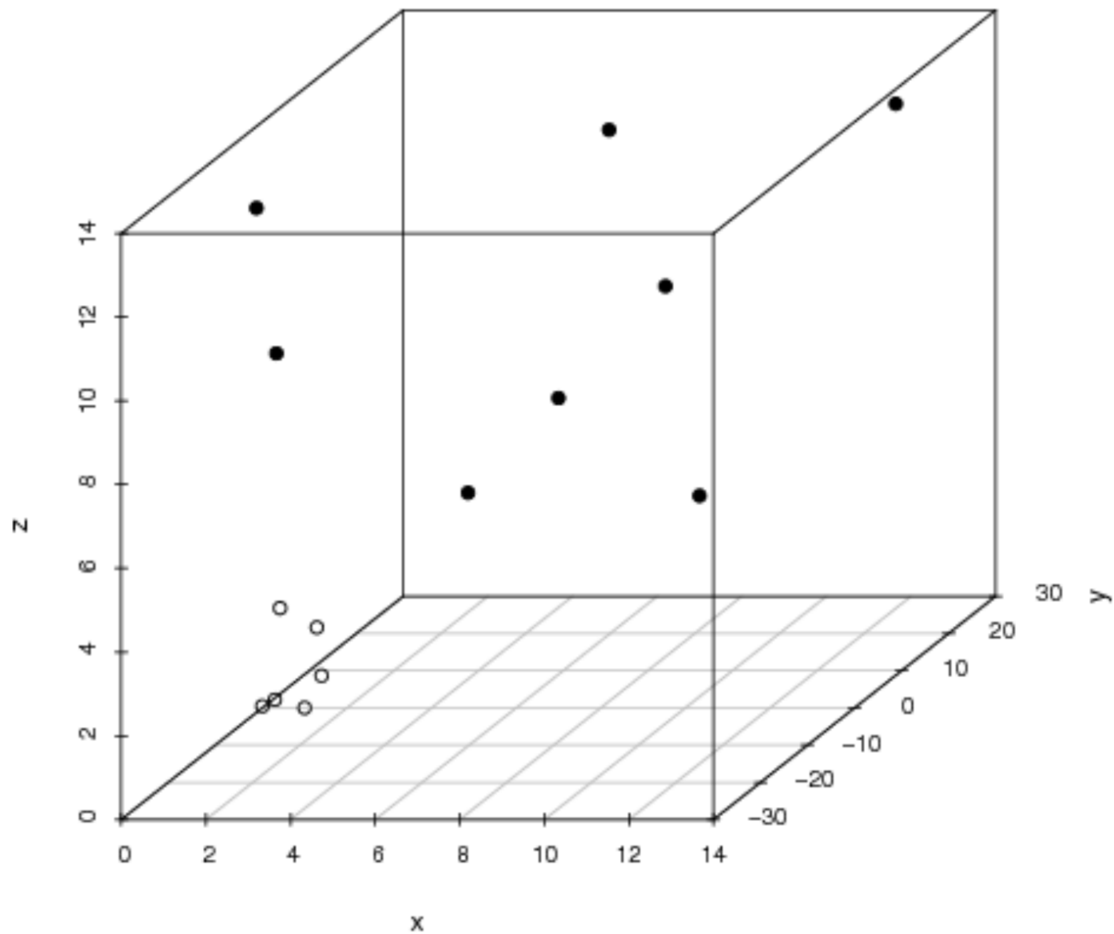
Specifically



- Best combination:
  - Feature set with interactions
  - heavily penalized
- Larger dataset very sensitive to alpha

Interactions

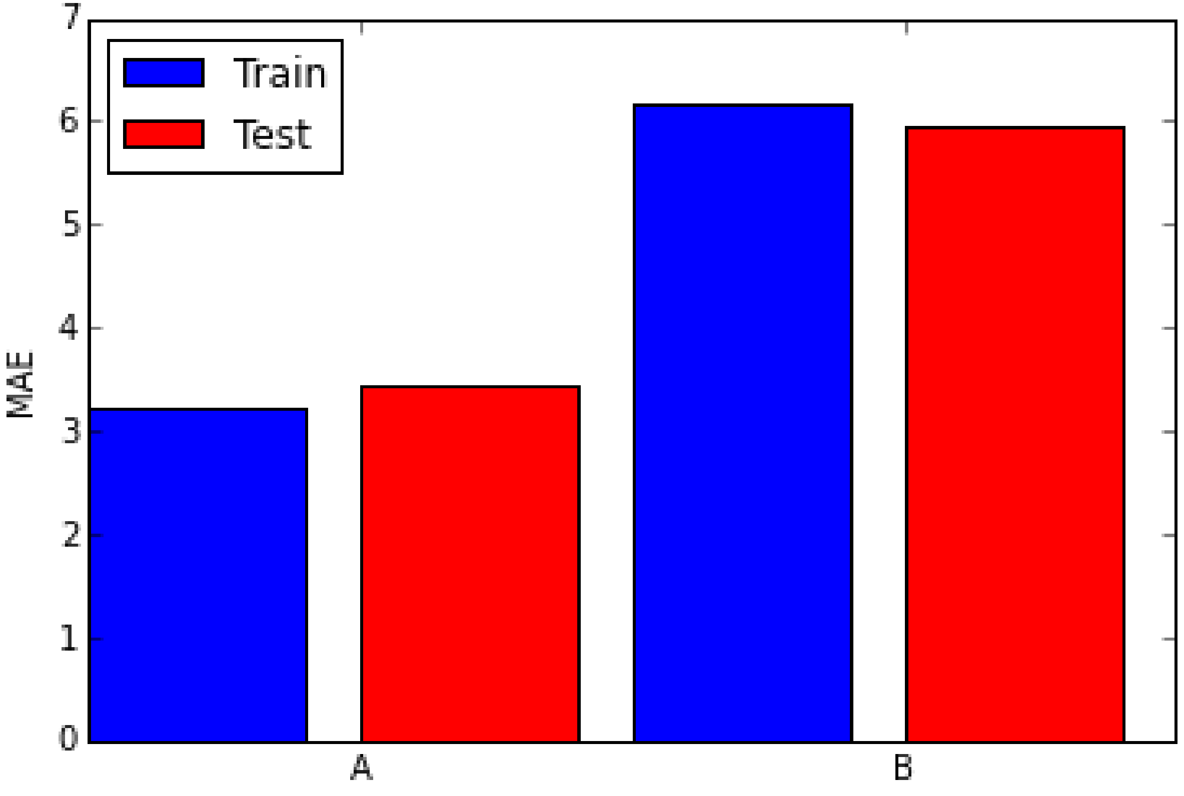




Advice

OK to use very large feature sets,
but make sure to penalize (regularize) your model!

The mystery of high error



- Is either model **A** or model **B** overfitting?
- Why is the test error of model **B** so high?
- Training error of model **B** is high
  - Model cannot even fit training data
  - **Underfitting**

Your model isn't working

- Is it underfitting or overfitting?

	TRAIN ERROR	VAL ERROR
OVER-FIT	LOW	HIGH
UNDER-FIT	HIGH	HIGH
FIT	LOW	LOW

Model selection

- Define **feature sets**
- Choose **hyperparameters**
  - Choose range of **values** to check
- Check performance on train and validation data
  - For as many **combinations** of the above as you can afford
- Pick combination with lowest error