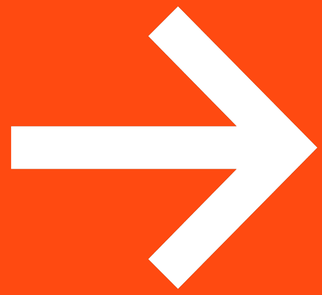
neue fische

School and Pool for Digital Talent

neue fische
School and Pool for Digital Talent

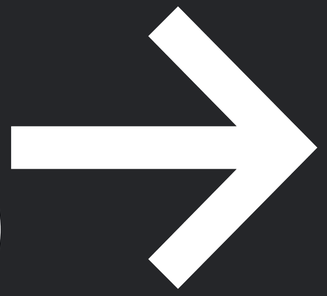
Regularization



neue fische
School and Pool for Digital Talent

Regularization

Part 1
Recap on Bias-Variance-TO



Bias - Variance Trade-Off

$$MSE(\hat{y}) = Bias^2 + Variance + Noise$$

→ to minimize the cost we need to find a good balance between
the Bias and Variance term of a model

→ we can influence bias and variance by changing the complexity of our model

Noise is the irreducible error of a model.
We cannot influence it.

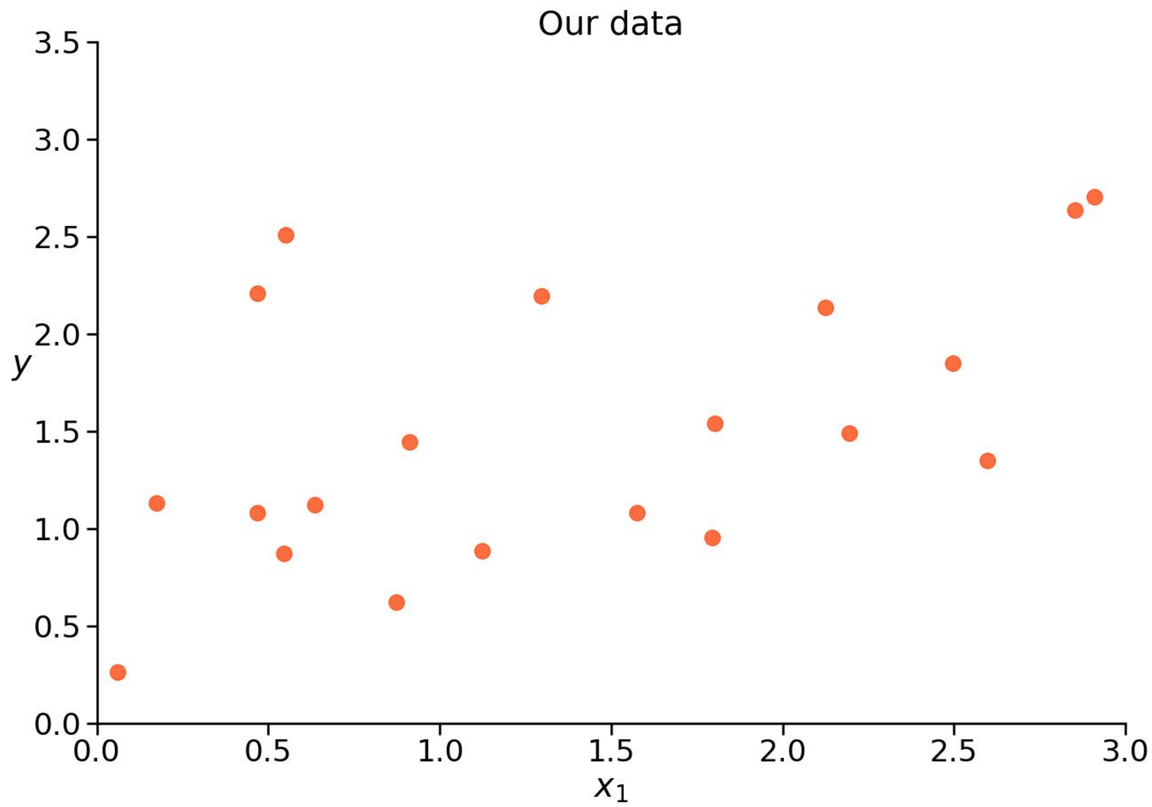
Example: Underfitting vs. Overfitting

We got data.

But we don’t know the underlying Data Generating Process.

So we want to model it.

Do you see a pattern/trend?



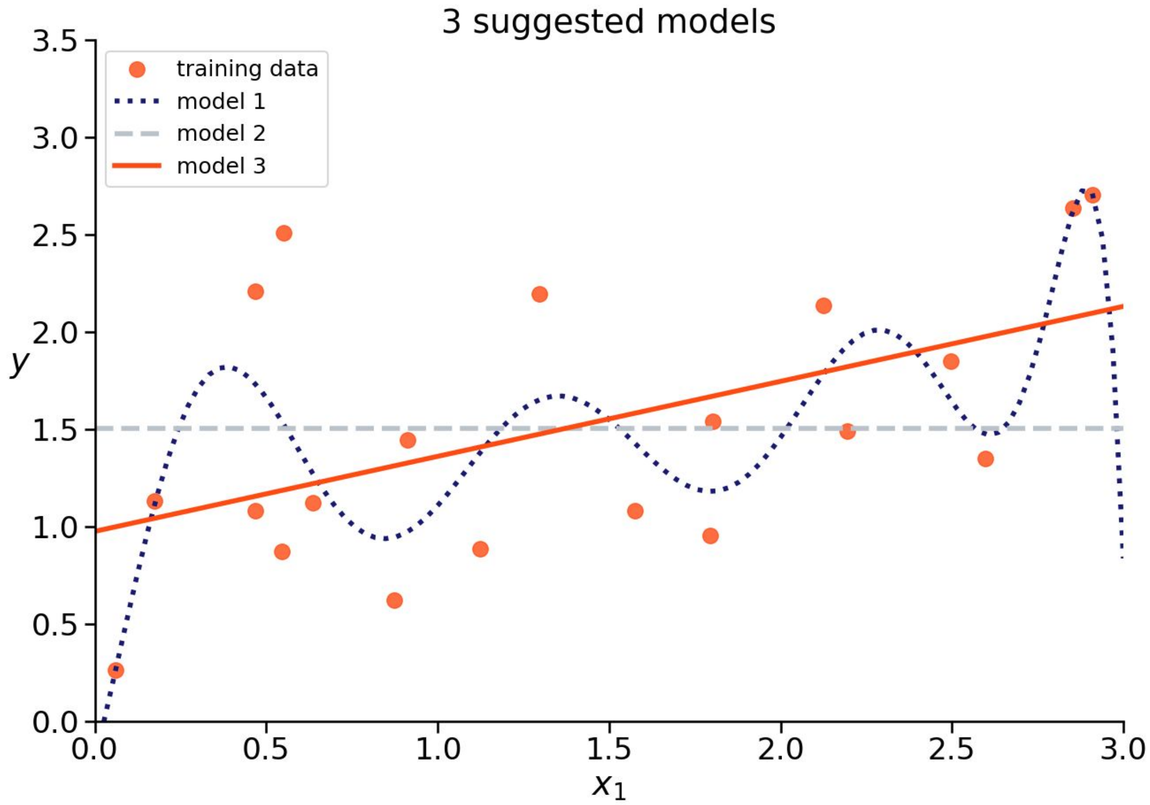
Example: Underfitting vs. Overfitting

Which model seems best?

Which model seems to underfit the data?

Which model might overfit the data?

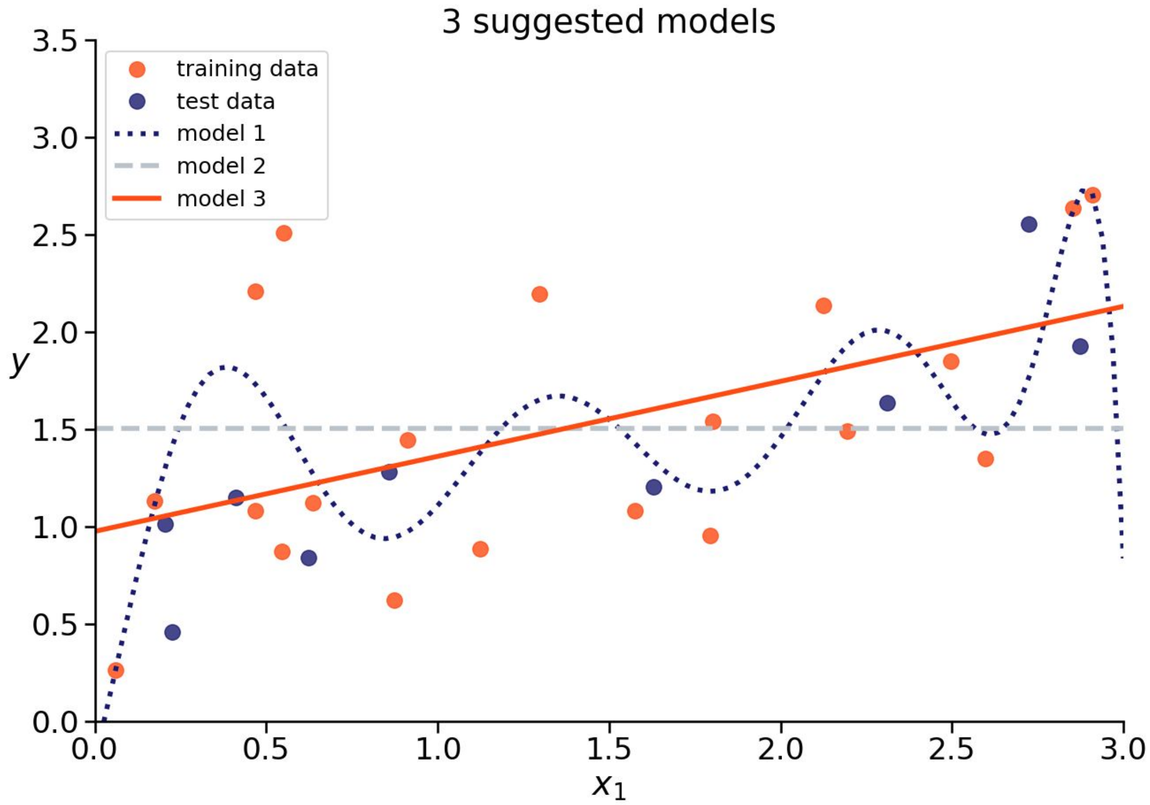
How to evaluate if a model is underfitting/overfitting?



Example: Underfitting vs. Overfitting

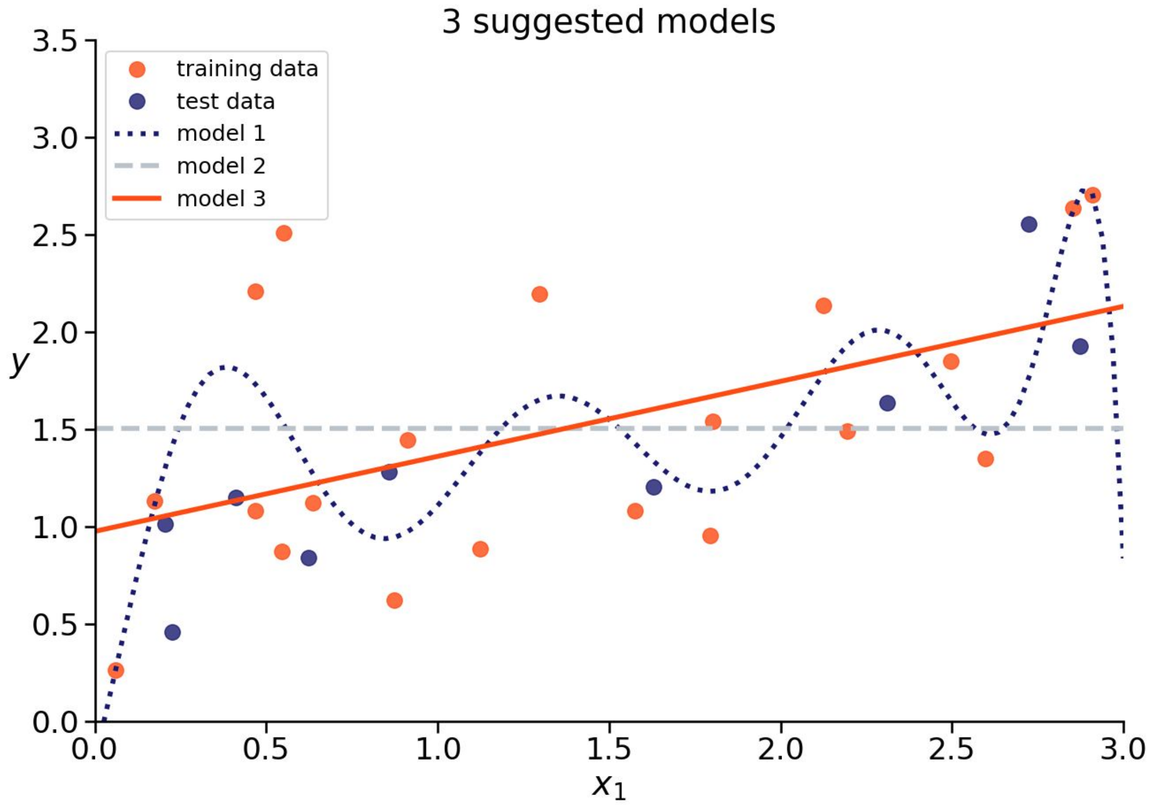
How to evaluate if a model is underfitting/overfitting?

- we need a cost function
- we need test data
- we should do error analysis



Example: Underfitting vs. Overfitting

How to figure out if your model is overfitting?

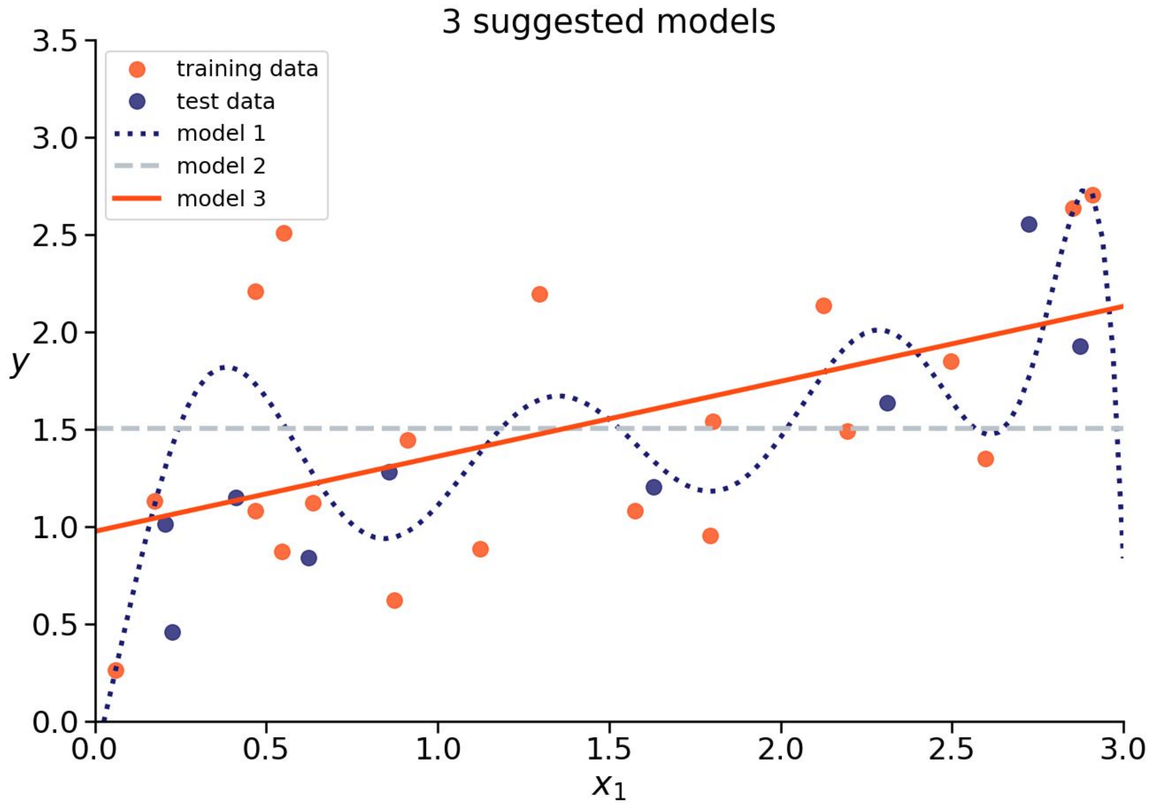


Example: Underfitting vs. Overfitting

How to figure out if your model is overfitting?

- error on training data is low
- error on test data is high

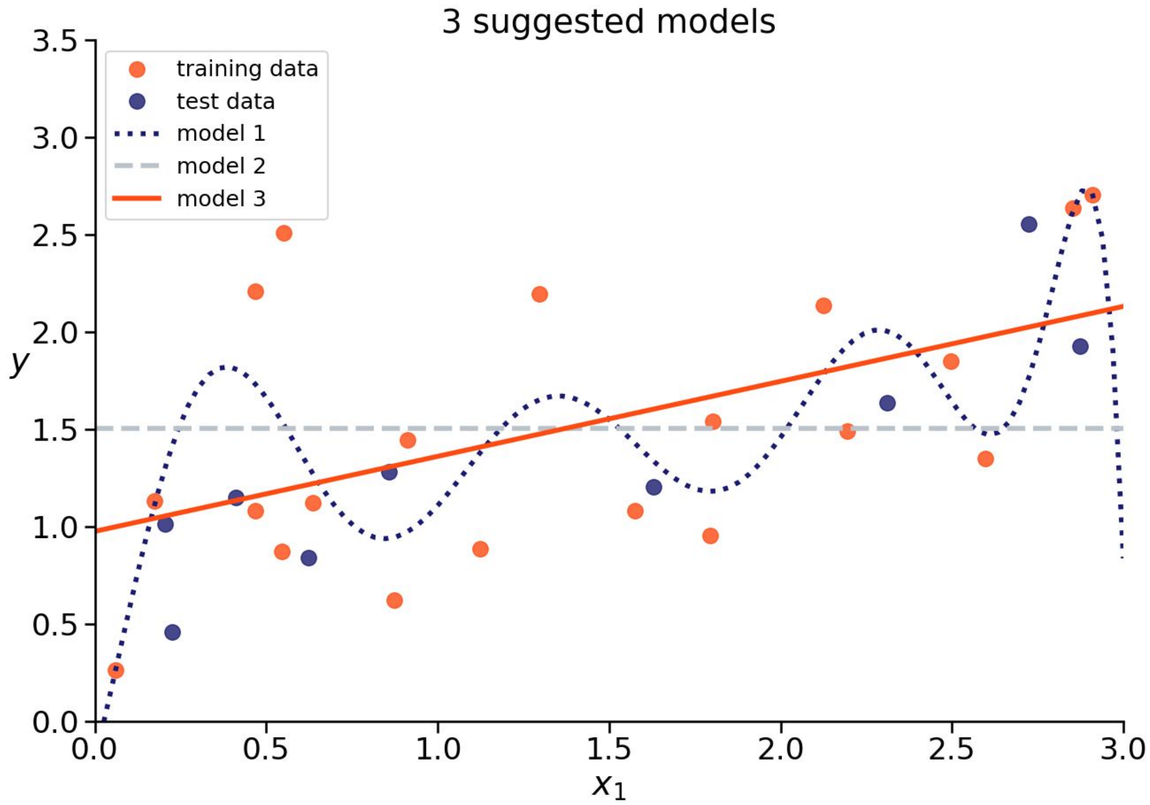
→ model memorizes the noise in the data



Example: Underfitting vs. Overfitting

How to figure out if your model is overfitting?

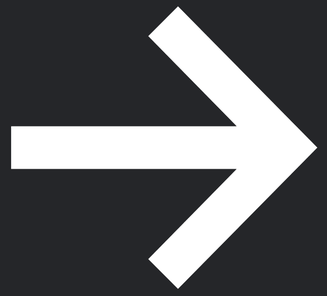
- error on training data is low
- error on test data is high



neue fische
School and Pool for Digital Talent

Regularization

Part 2
A Visual Approach



Prevent Overfitting

if we see overfitting of our model, we

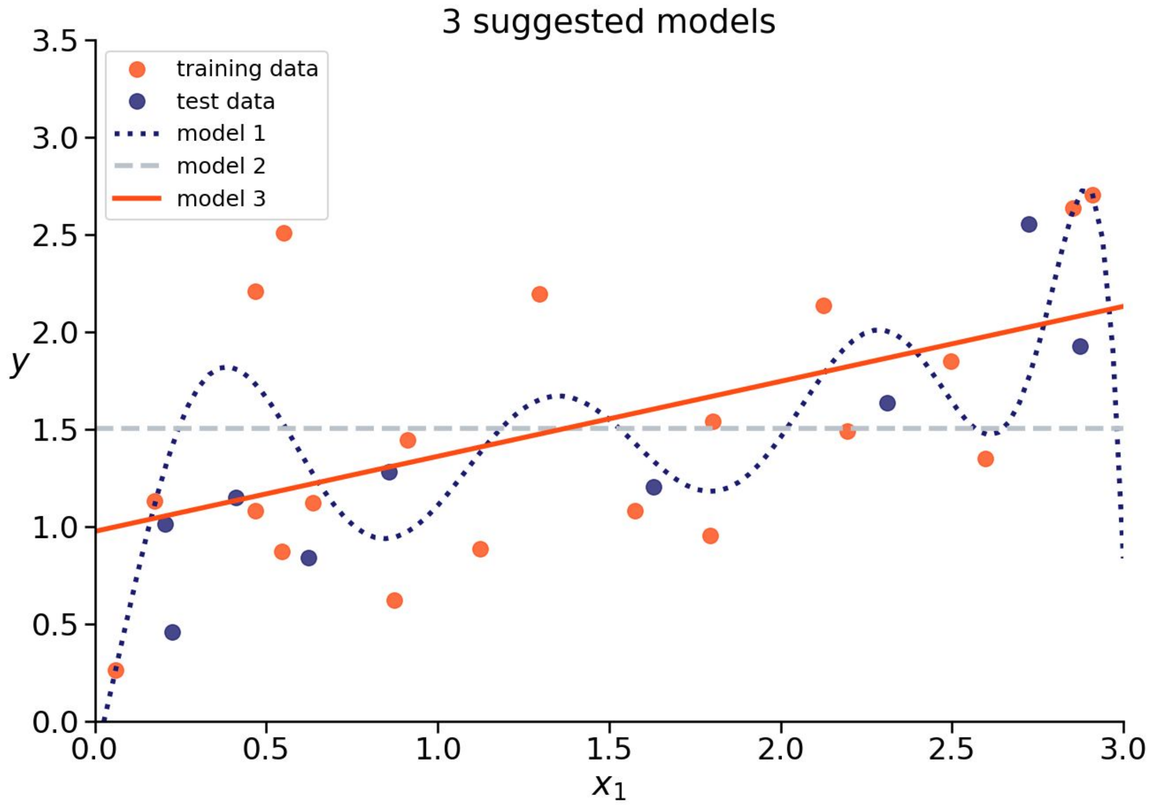
could gather more data

Prevent Overfitting

Every model type has a way to reduce model complexity.
We just learnt Linear Reg. that's why we will concentrate on the Regularization of those models for now.

if we see overfitting of our model, we could reduce its complexity.

HOW?

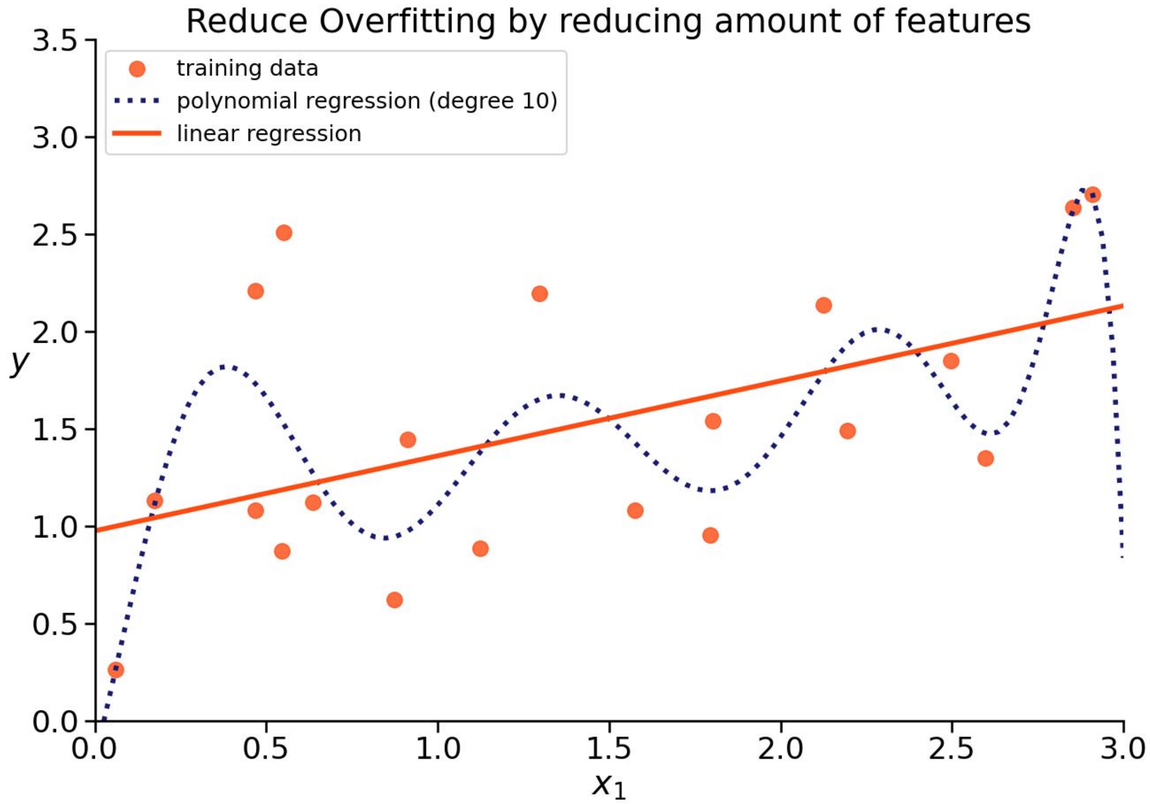


Prevent Overfitting

if we see overfitting of our model, we could reduce its complexity.

HOW?

- reduce amount of features



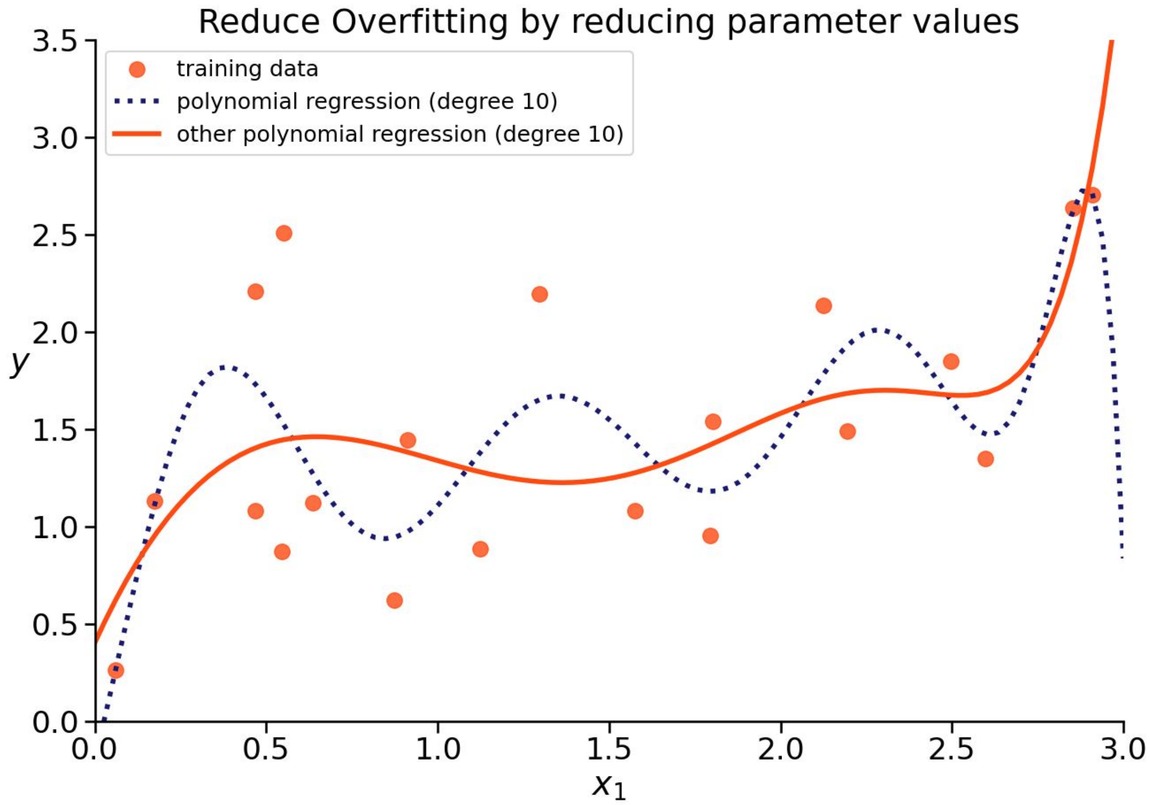
Prevent Overfitting

if we see overfitting of our model, we could reduce its complexity.

HOW?

- reduce amount of features

- make the model less susceptible to data by reducing the influence of features
  → smaller coefficients



Prevent Overfitting with Regularization

BOTH can be achieved with regularizing a model:

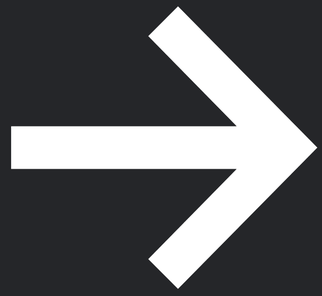
- reduce amount of features

- make the model less susceptive of data by reducing the influence of features (smaller coefficients)

neue fische
School and Pool for Digital Talent

Regularization

Part 3
Regularization



Regularization

Regularization conceptually uses a hard constraint to prevent coefficients from getting too large, at a small cost in overall accuracy. With the aim to get models that generalize better on new data.

Even with linear models, it can be useful to regularise them. Because they have a tendency to trace outliers in the training data.

Hard constraint

We add a hard constraint to our cost function:

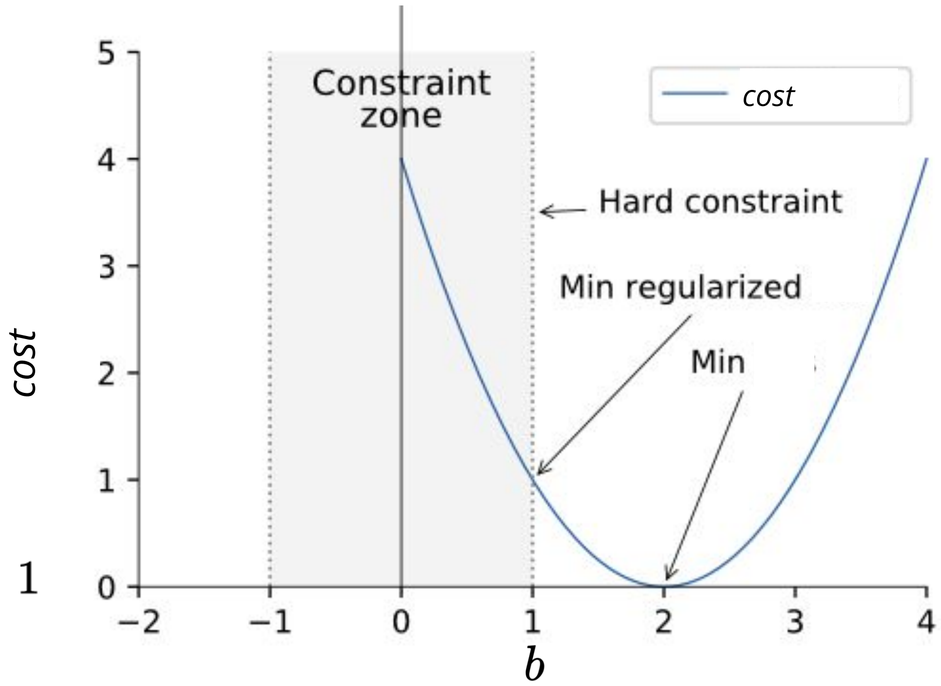
$$\min\ J(b_0, b_1) = \frac{1}{n}\sum (y_i - b_0 - b_1 x_i)^2\ subject\ to\ -1 \leq b_1 \leq 1$$

general form of the constraint:

$$-t \leq b_1 \leq t$$

What do we have to change to get to a form like this:

$$b_1 \leq t$$



We are not constraining the y-intercept

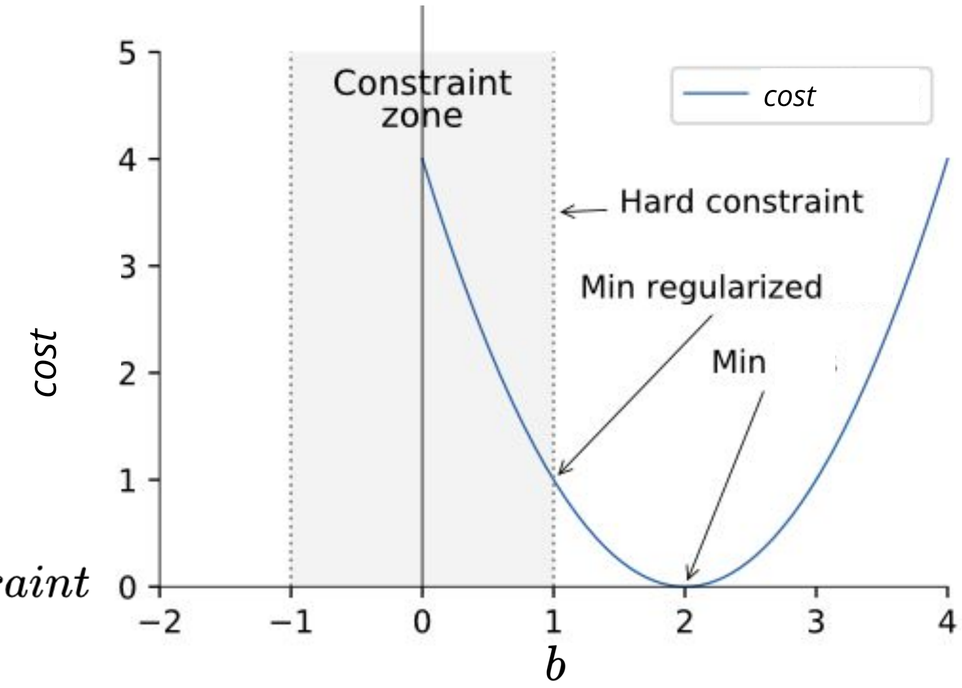
Hard constraint

We add a hard constraint to our cost function:

$$\min J(b_0, b_1) = \frac{1}{n} \sum (y_i - b_0 - b_1 x_i)^2 \; subject\, to\, L1/L2\, constraint$$

The most common regularization constraints:

$$L_1 : |b_1| \leq t$$
$$L_2 : b_1^2 \leq t$$



Hard constraint with more features

We add a hard constraint to our cost function:

$$\min J(b_0, b_1, \ldots, b_m) = \frac{1}{n}\sum (y_i - b_0 - b_1 x_i - \ldots - b_m x_m)^2 \; subject\, to\, L1/L2\, constraint$$

The most common regularization constraints:

$$L_1 : |b_1| + |b_2| + \ldots \leq t$$
$$L_2 : b_1^2 + b_2^2 + \ldots \leq t$$

We can add this constraint directly to our Loss function through the magic of Lagrange multipliers

If we have more than one feature, we need to bring them all to the same scale. Otherwise they contribute different to the penalty term.

Soft constraint

We can add this constraint directly to our Loss function through the magic of Lagrange multipliers (t becomes alpha (or lambda))

$$J(b) = \frac{1}{n}\Sigma(y - (b_0 + b_1x_1 + b_2x_2))^2 + \alpha\left(b_1^2 + b_2^2\right)$$

$$J(b) = \frac{1}{n}\Sigma(y - (b_0 + b_1x_1 + b_2x_2))^2 + \alpha(|b_1| + |b_2|)$$

Alpha is a hyperparameter. Before training the model we need to set it.

Soft constraint

We can add this constraint directly to our Loss function through the magic of Lagrange multipliers (t becomes alpha (or lambda))

$$J(b) = \frac{1}{n}\Sigma(y - (b_0 + b_1x_1 + b_2x_2))^2 + \alpha(b_1^2 + b_2^2)$$

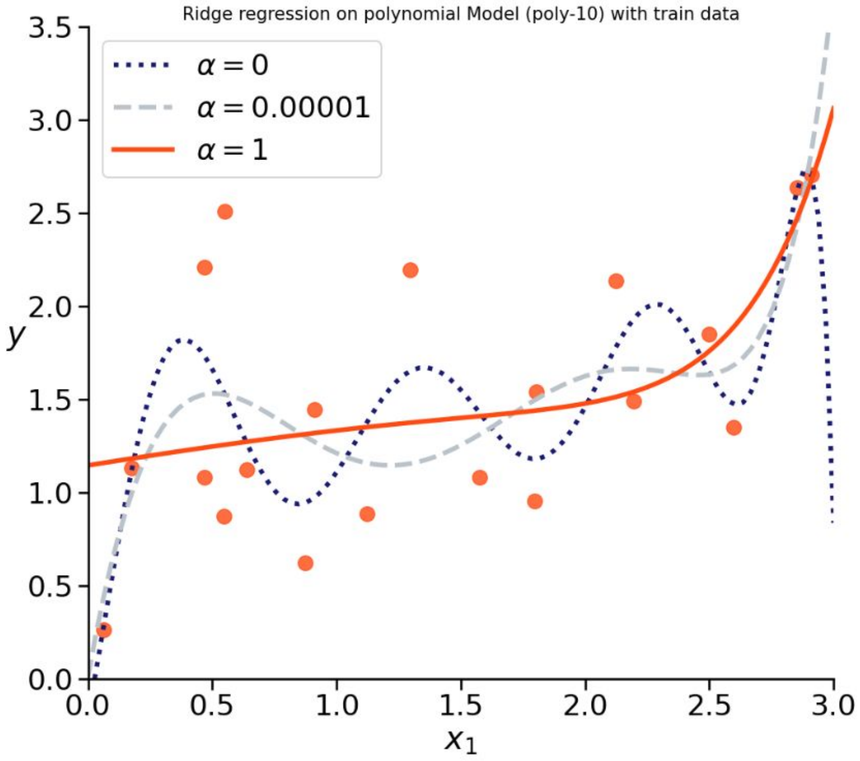
$$J(b) = \frac{1}{n}\Sigma(y - (b_0 + b_1x_1 + b_2x_2))^2 + \alpha(|b_1| + |b_2|)$$

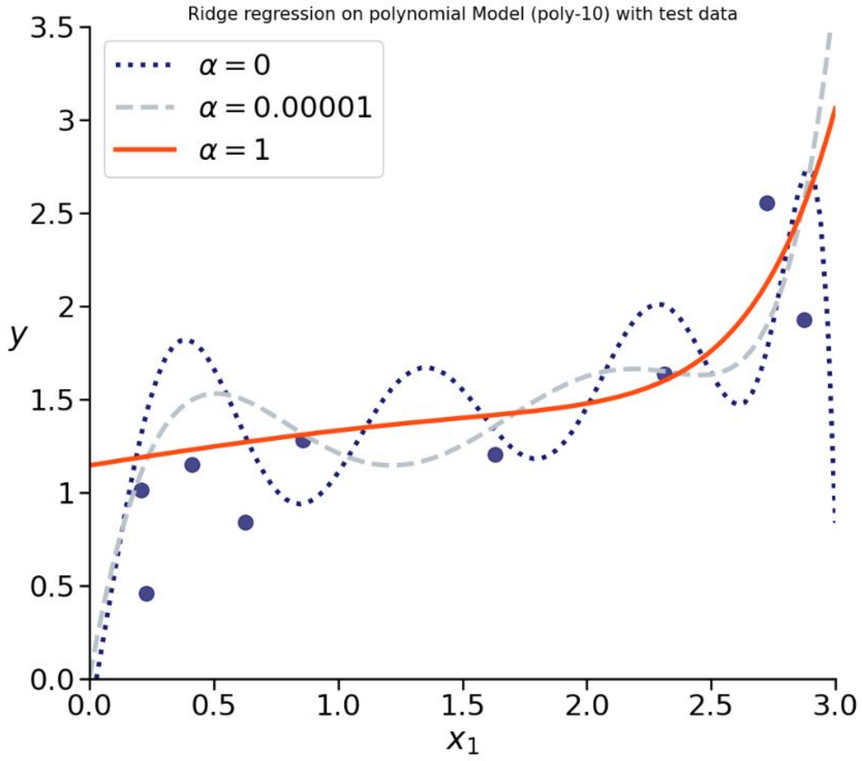
What happens if we set alpha to 0?

What happens if we set alpha to a very high value?

Some different alpha values

We have to test some values for alpha and check which give us best results on unseen data

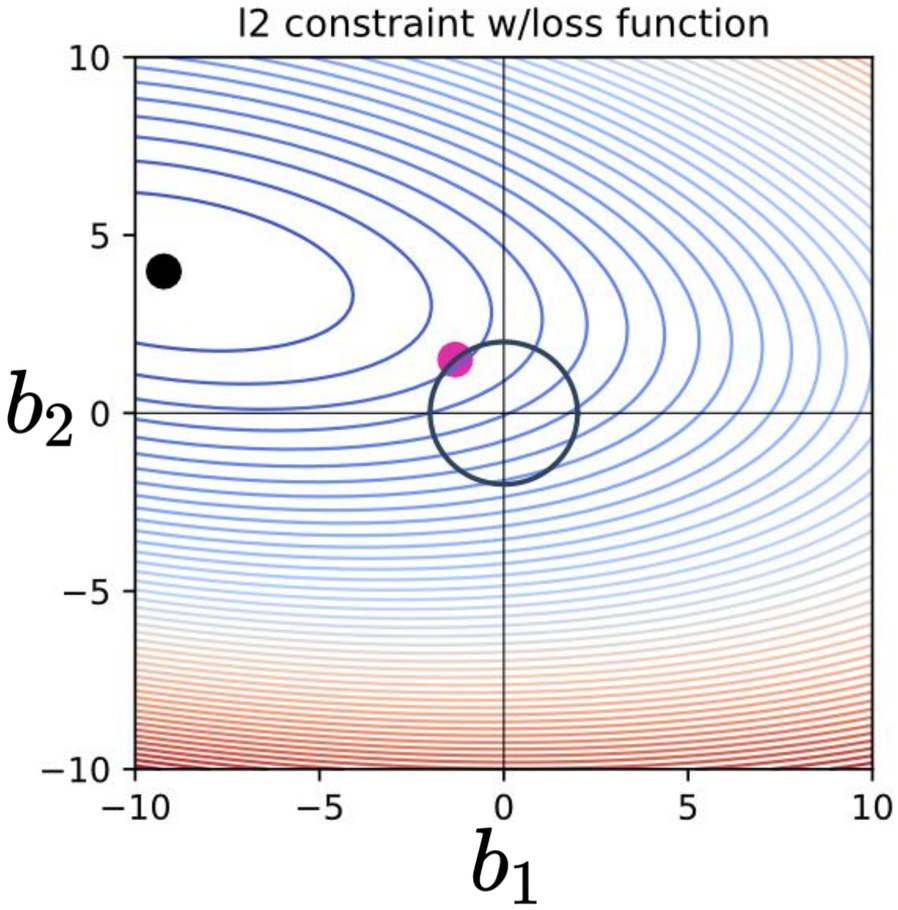




Ridge Regression

- Also called L2 Regularization / l_2 norm
- the regularization term forces the parameter estimates to be as small as possible
- **weight decay**

$$J(b) = \frac{1}{n}\Sigma(y - \hat{y}(b))^2 + \alpha \sum b_i^2$$

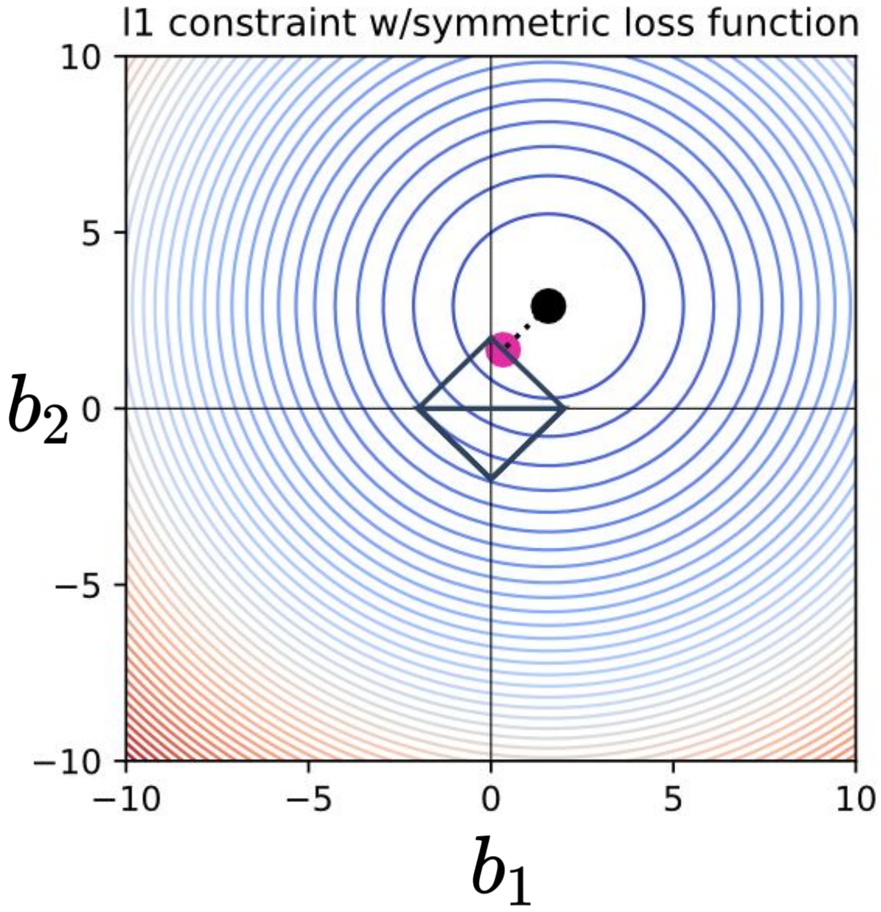


Lasso Regression

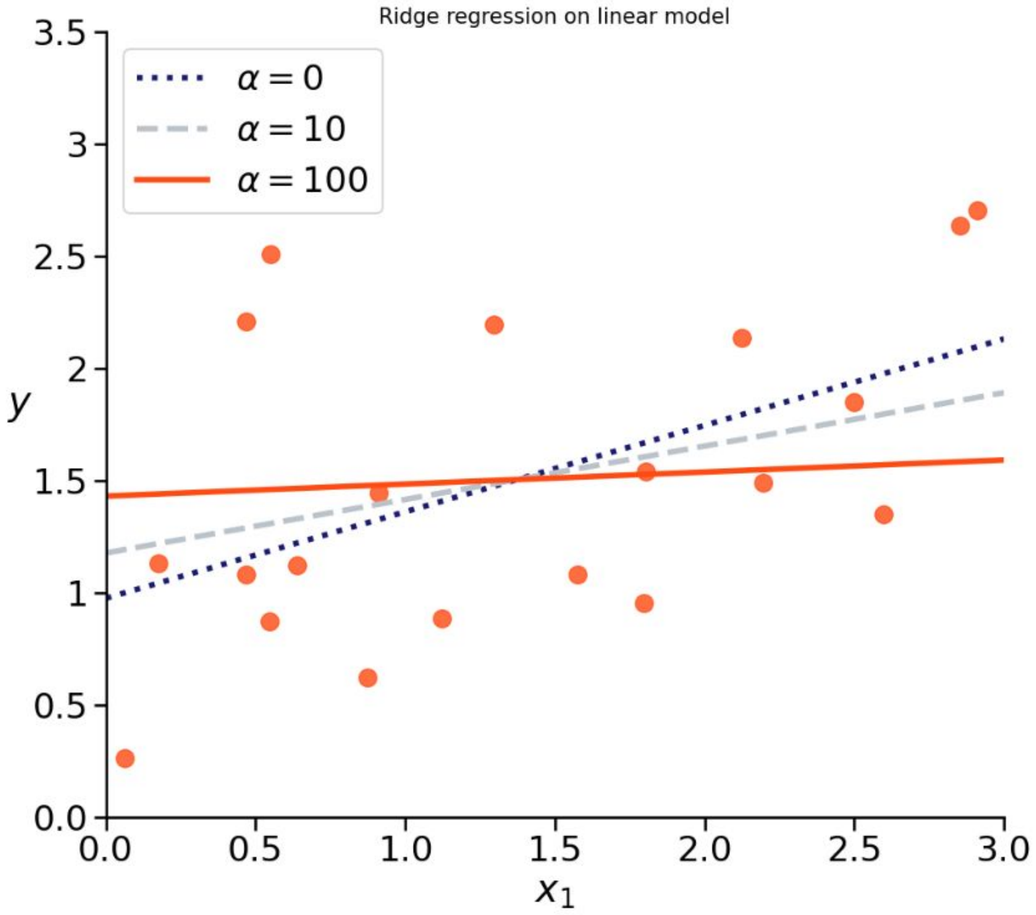
Least **A**bsolute **S**hrinkage and **S**election **O**perator

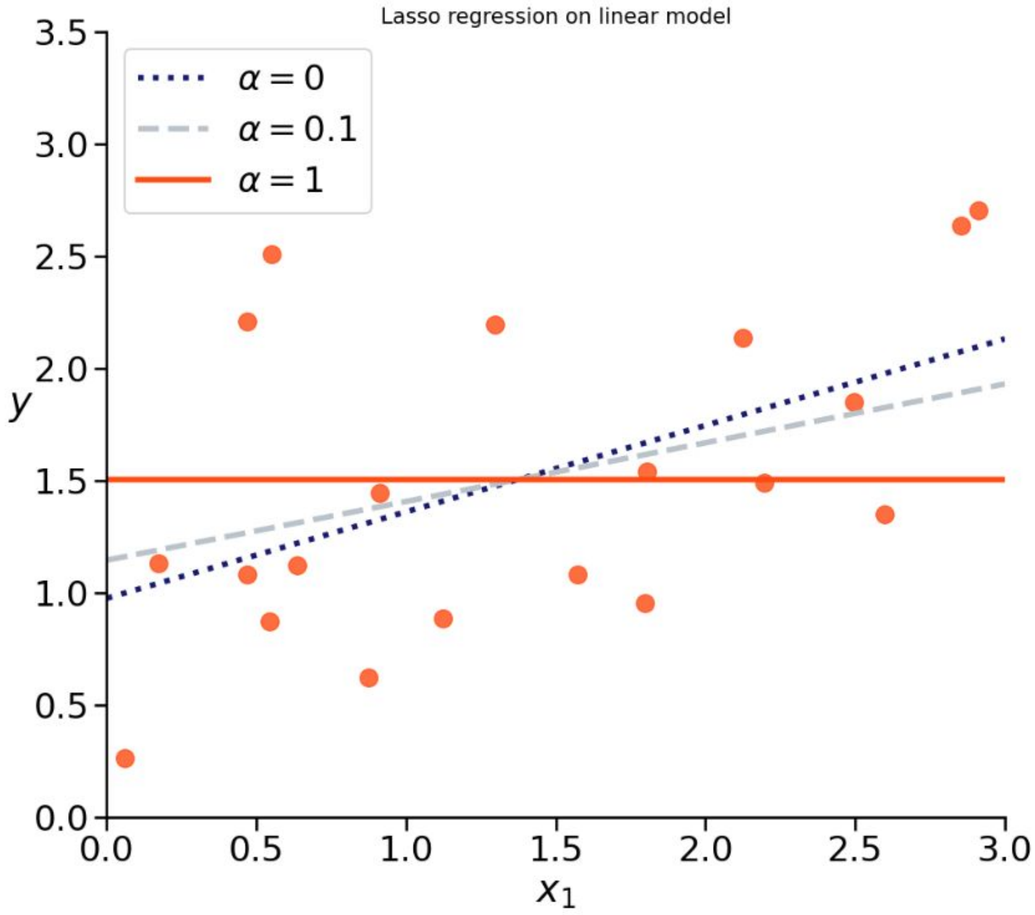
- Also called L1 Regression / l_1 norm
- Tends to **eliminate weights** = it automatically performs **feature selection**

$$J(b) = \frac{1}{n}\Sigma(y - \hat{y}(b))^2 + \alpha\sum|b_i|$$



Ridge vs Lasso





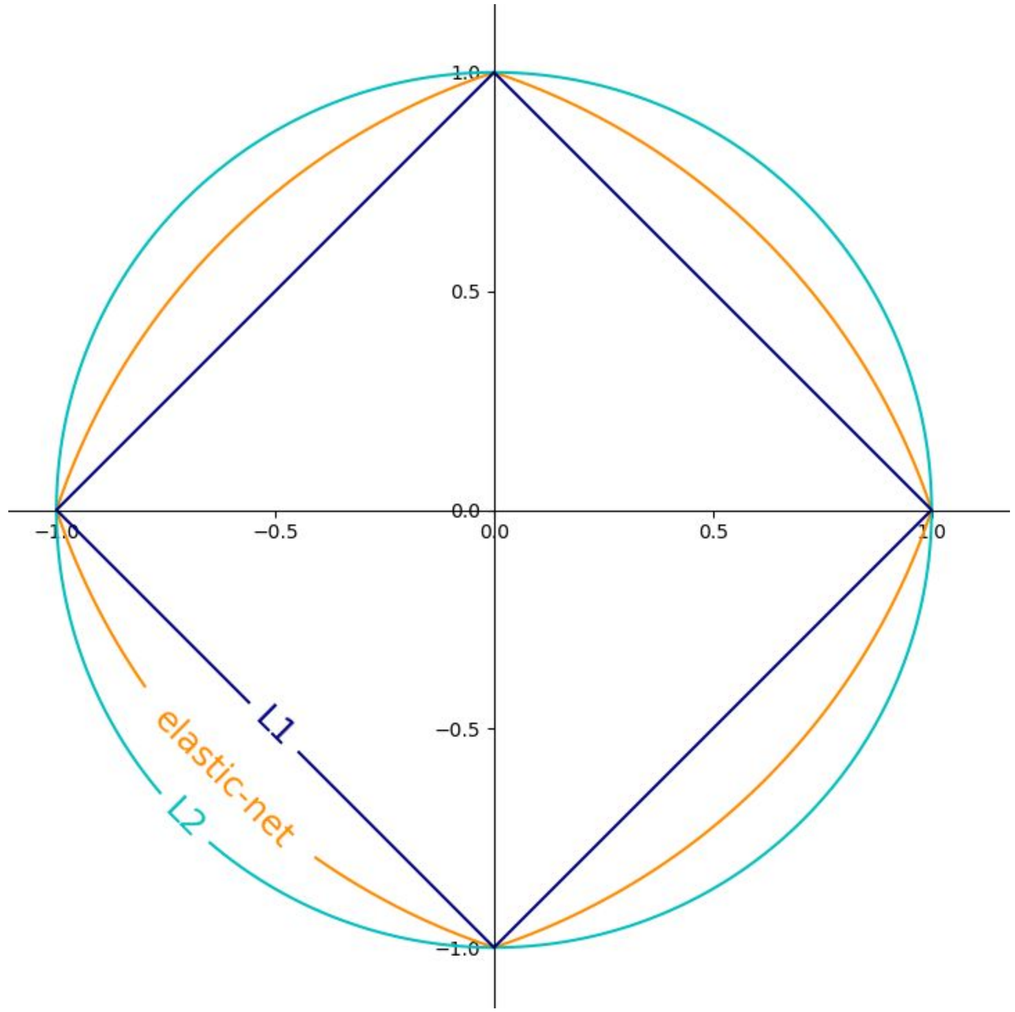
Elastic Net - Mixing Lasso and Ridge

- Regularization term is weighted average of Ridge and Lasso Regularization term
- When r = 0 it is equivalent to Ridge, if r = 1 it is equivalent to Lasso Regression
- Preferable to Lasso when features are highly correlated or to Ridge for high-dimensional data (more features than observations)

$$J(b) = \frac{1}{n}\Sigma(y - \hat{y}(b))^2 + \alpha r \sum |b_i| + \alpha \cdot (1 - r) \sum b_i^2$$

Comparison of regularization methods

- elastic net is between L1 and L2 (whatever you use as r... it will change its form more to L2 or L1



References

- Hands-on ML with scikit-learn and TensorFlow, Geron
- http://scott.fortmann-roe.com/docs/BiasVariance.html
- https://medium.com/analytics-vidhya/bias-variance-tradeoff-for-dummies-9f13147ab7d0
- Machine Learning - A probabilistic Perspective - Kevin P. Murphy
- https://explained.ai/regularization/constraints.html
- https://explained.ai/statspeak/index.html
- https://people.eecs.berkeley.edu/~jrs/189s21/