Fourth Industrial Summer School

Module 4: ML

Supervised Learning: Regression

Friday, July 5, 2024

Session Objectives

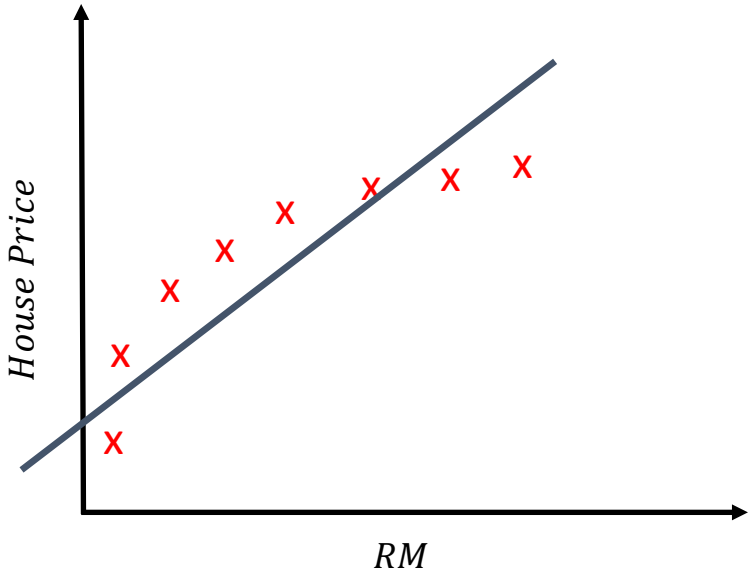
- ✓ Regularization
  - ✓ Ridge Regression
  - ✓ Lasso Regression
  - ✓ Elastic Net

Modeling Main Issues

We may experience two types of modeling issues:

- When we built a simple model, and it shows poor performance on both the training and testing stages - Underfitting (a model with high bias)
- On the other hand, we may build a very complex model which did fantastic job on training, but very poor on testing stage, then we may have an issue to generalize – or called Overfitting

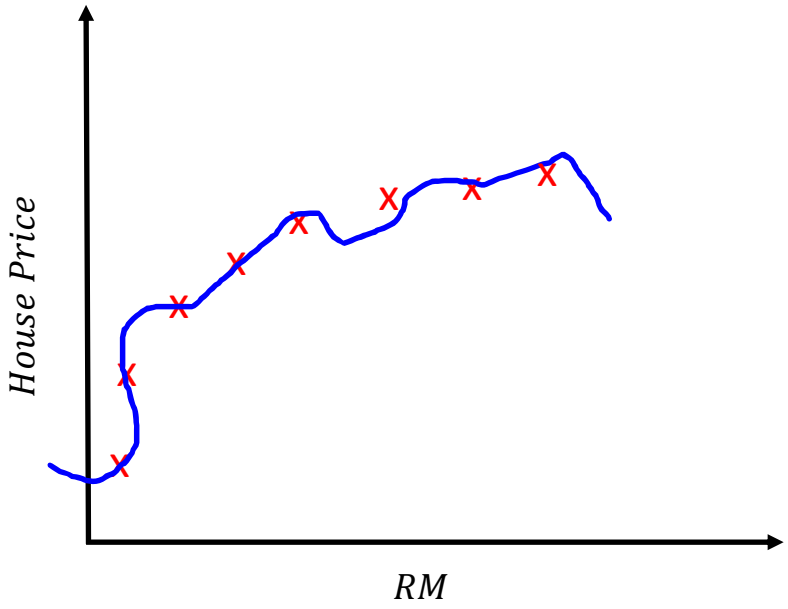
Underfitting



$$\beta_1 + \beta_0 X$$

- If we fit the above linear model, the resultant model will suffer underfitting (high bias) and probably will not generalize with new data.

Overfitting

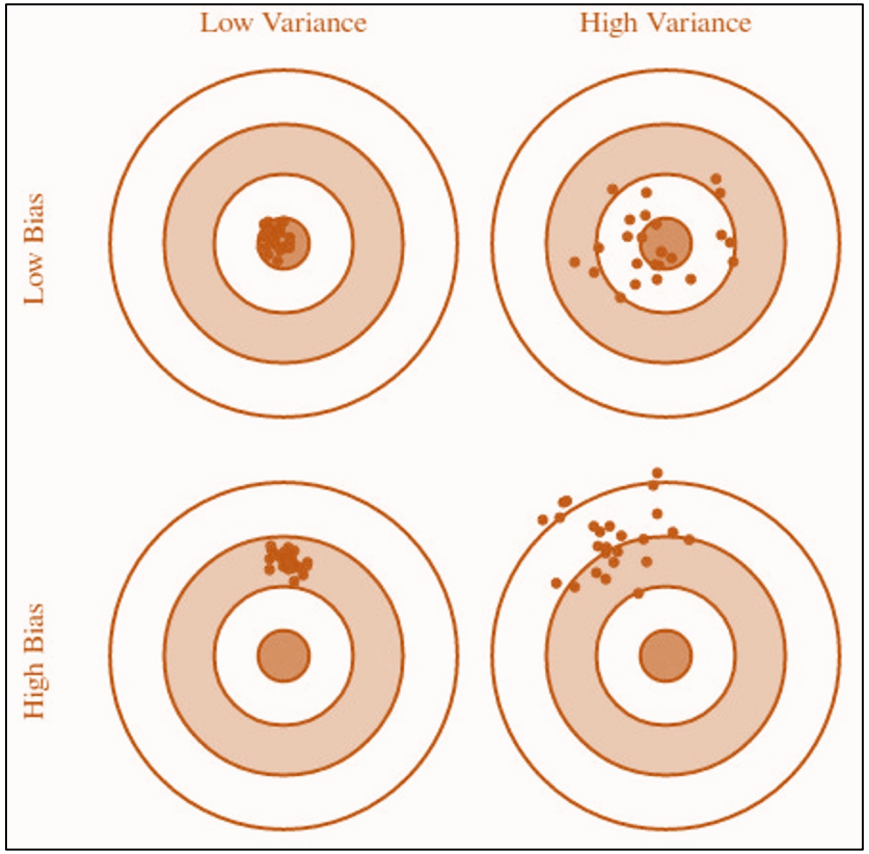


$$\beta_1 + \beta_0 X + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4$$

- If we fit the model too much, then we get a oscillatory model that may pass through all data points, and show perfect results(on training set).
- This model will not generalize well as it suffers overfitting (**high variance model)**

Bias and variance tradeoff

- Finding the right balance between bias and variance (sensitivity to training data) is a **key challenge in machine learning**, often referred to as the bias-variance tradeoff.



Middle dark circle is the true population; the small dots are samples generated from different estimators

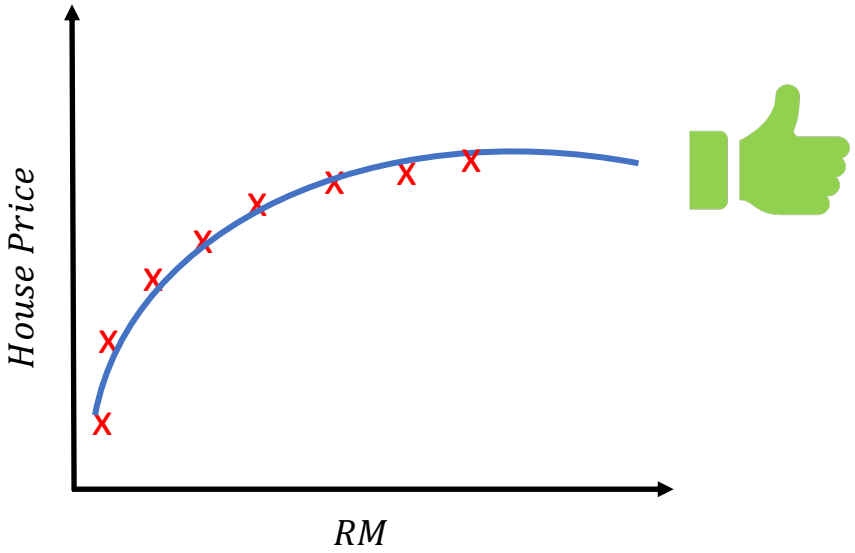
LV/LB: In theory, a perfect machine learning model would achieve both low bias and low variance. However, achieving both these qualities simultaneously is very difficult, if not impossible! Due to 1) the natural tradeoff between Bias/ variance, 2) nature of the read data, 3) the chosen ML algorithm

HV/LB: A model is flexible in learning the data patterns but might be too sensitive to the specific training data it was exposed to. This is the real situation where every machine learning engineer is trying to tune and improve!

LV/HB: This is an ideal situation for **underfitting**. You are making a strong assumption about the modeling, and never allow any flexibility to learn from the specifics of the data.

HV/HB: The model is both overly sensitive to the specifics of the training data and also limited in its ability to learn the underlying patterns. This combination leads to poor performance on both the training data and unseen data, lead to **overfitting** in general.

Generalize – Suitable model



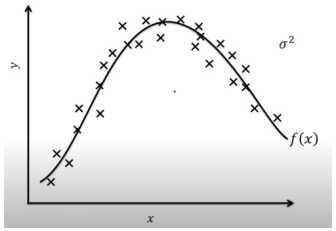
$$\beta_1 + \beta_0 X + \beta_2 X^2$$

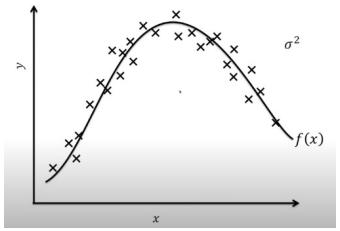
- This model is better than the previous two models simple and complex, hence, can be more generalized to new data

Bias issue

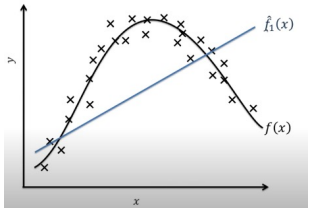
- With high assumptions, as in the below example, the models $\hat{f}(x)s$ **are different much from one another and comparing the true mean and models mean reveals high dissimilarity.**

Training dataset

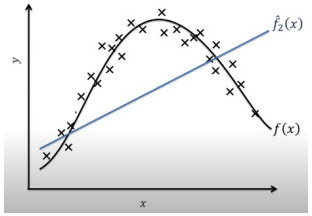




Simple models $f_1(x)$ … $f_n(x)$



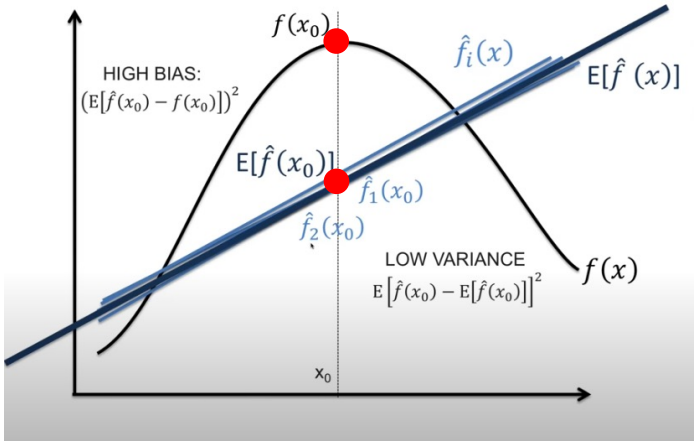
Many different Training sets



True mean is far away from predicted mean

$$f(x_0) >> E[\hat{f}(x_0)]$$

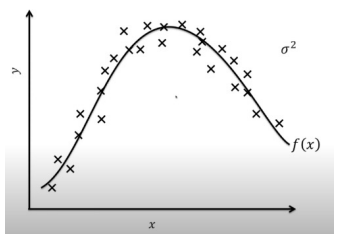
High Bias, but low variance

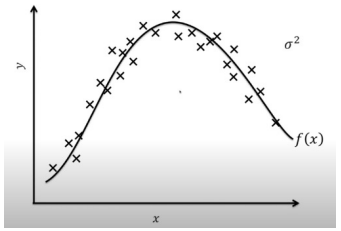


Variance issue

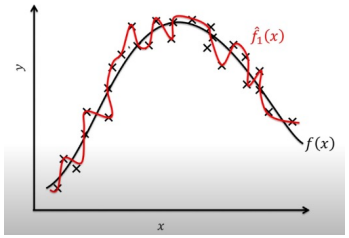
- With high flexible models, as in the below example, the models $\hat{f}(x)s$ **are vary a lot around on different samples.**

Training dataset

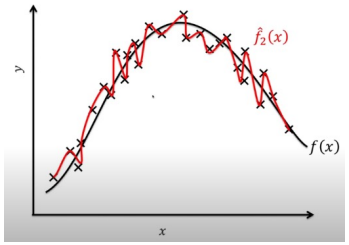




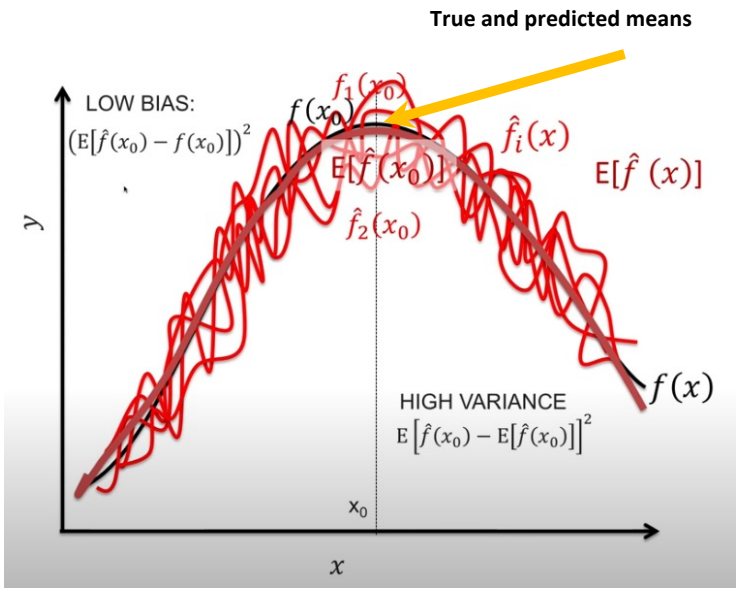
complex models $f_1(x)$ … $f_n(x)$



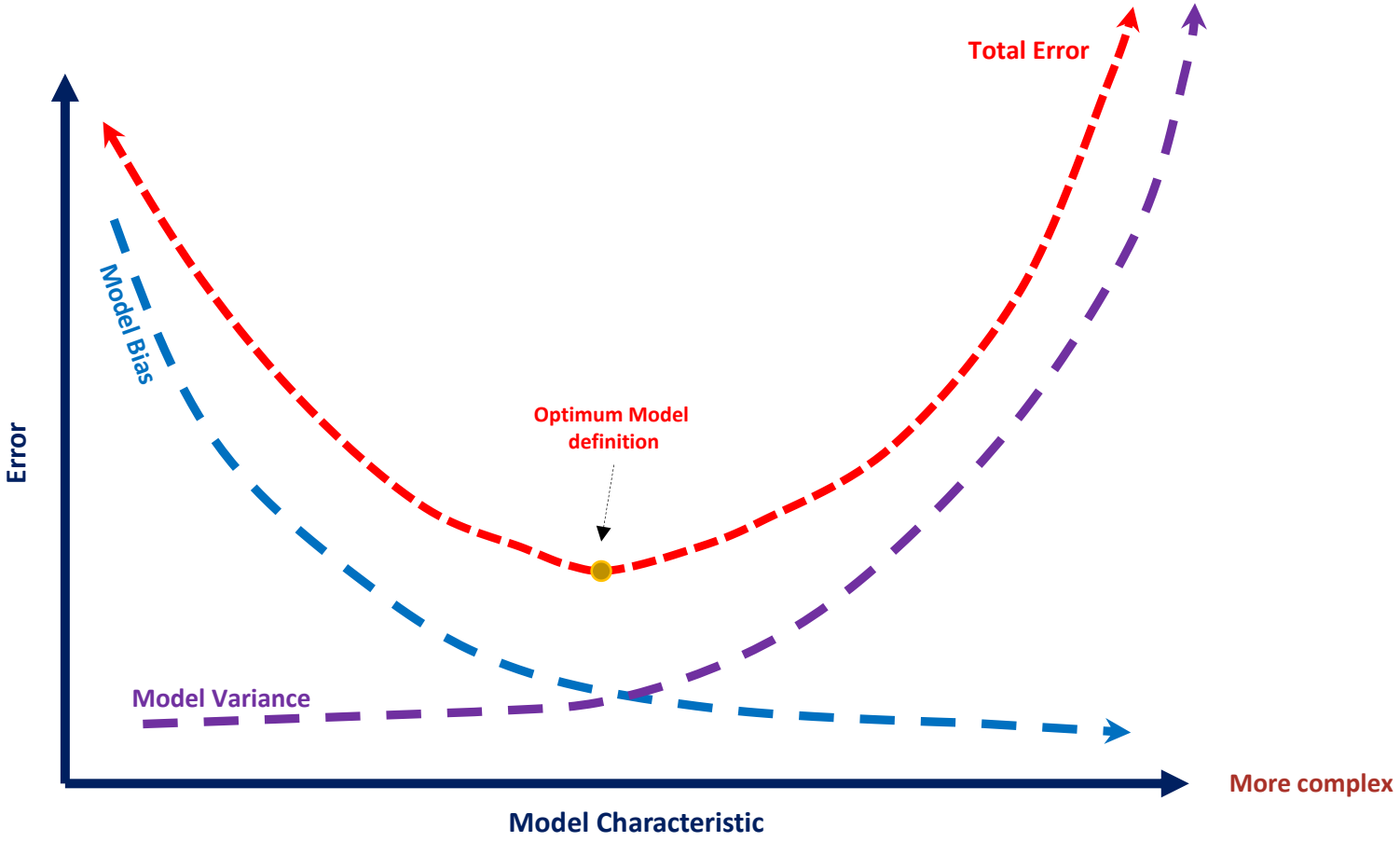
Many different Training sets



Overfitting - $\hat{f}(x)$ changes a lost on every sample



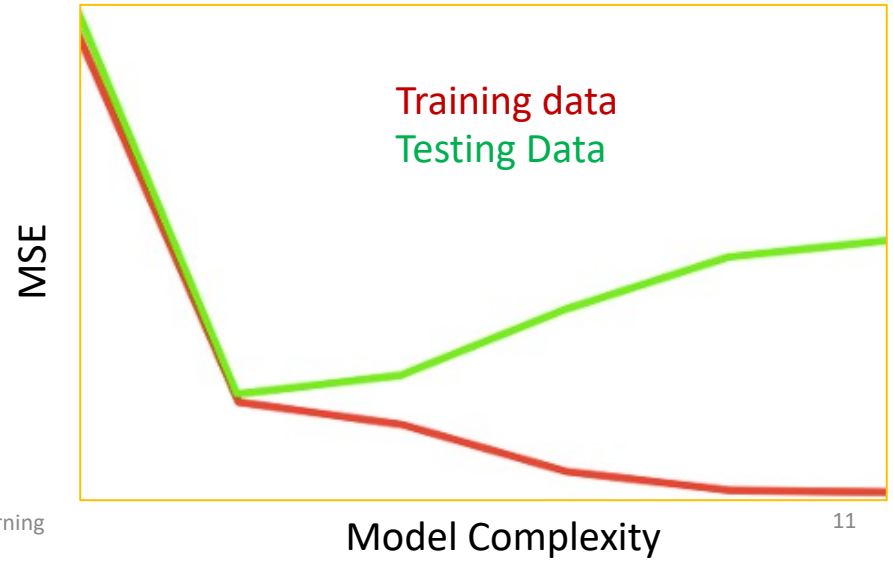
Model bias/variance vs. error



Bias vs. variance: during training/testing

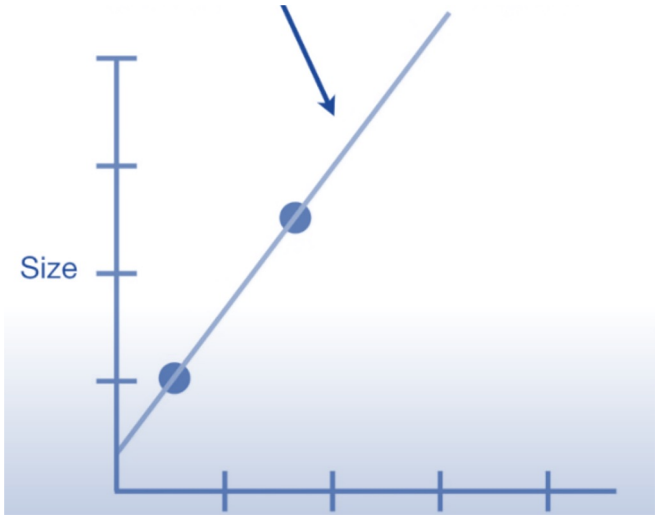
- Plot MSE as a function of model complexity
  - Polynomial order
- Training Error Decreases as model complexity increases
- What about test data?
  - Higher order model
    - Test Error increases
    - Overfitting
  - 0^{th} to 1^{th} order model
    - Tr/Ts Error increases
    - Underfitting

$$MSE = \frac{\sum(y_i - \hat{y}_i)^2}{m}$$

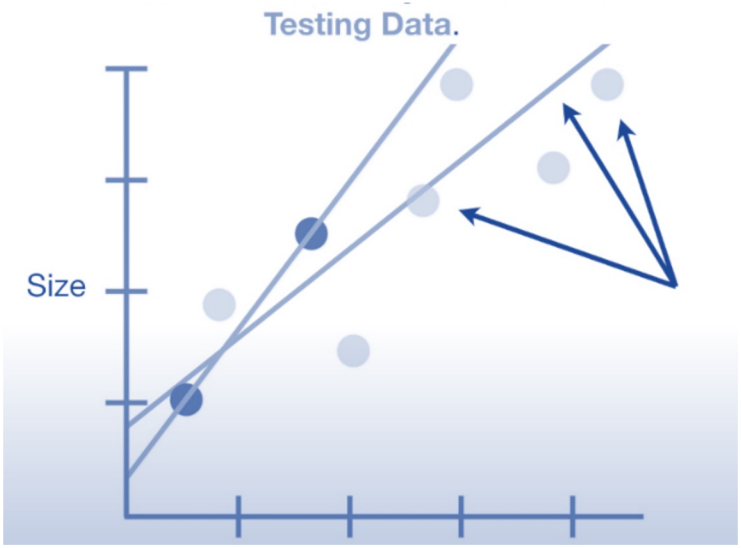


Example: Not enough data may lead to overfitting

Size = 0.4 + 1.3 * weight



SSE = 0

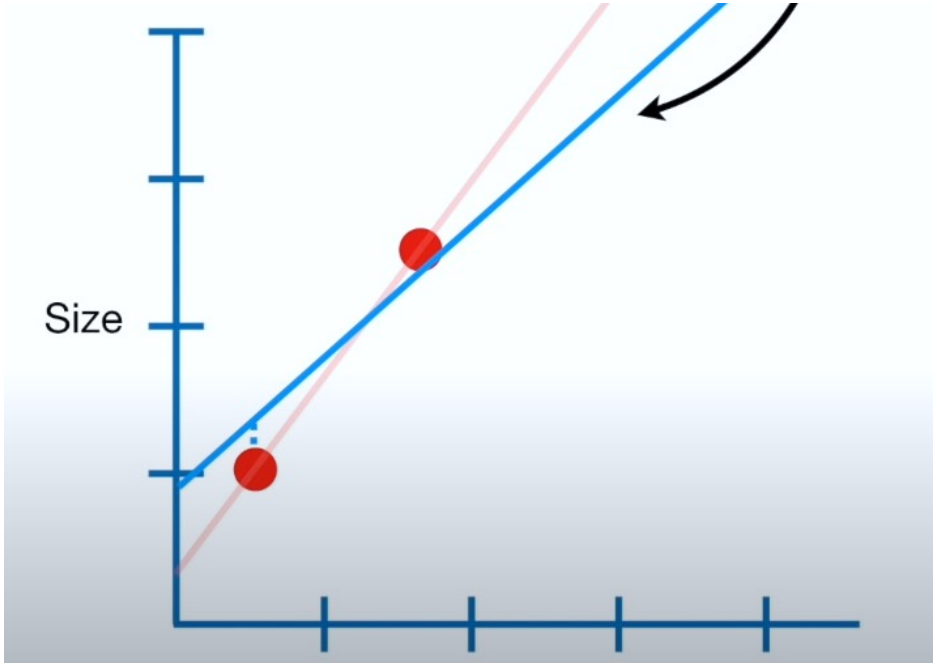


Testing residual is high
High variance model.
This is overfitting

- Another reason why to use regularization

Main idea of Regularization

- We build a model that is not fitted perfectly to the training data in avoidance of overfitting.



Remedy

- In case of **underfitting**, we usually increase the complexity of the model, so it can fit better to the data
- In case of **overfitting**, we simplify the model (aka. regularize)
- **Lasso and Ridge** Regression are two preventive methods reduce the issue of overfitting

Regularization: is a method to control the model params. by penalizing them so they are not overestimated!

Ridge Regression (L2 regularization)

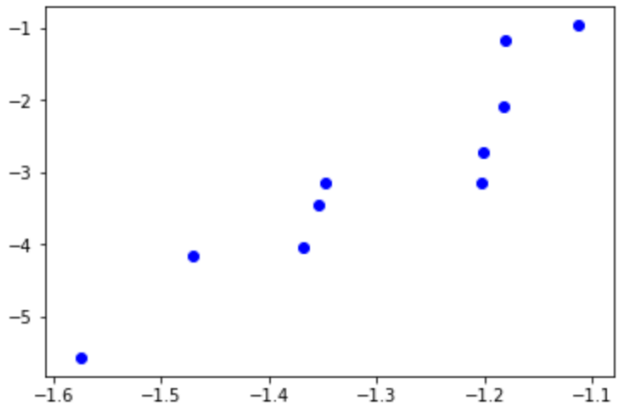
$$\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_i)\right)^2 + \lambda \sum_{i=1}^{n} \beta_i^2$$

- In Ridge regression the cost function is modified by adding a penalty equivalent to square of the magnitude of the coefficients.
- It is similar to simple LR, but with adding the following term:

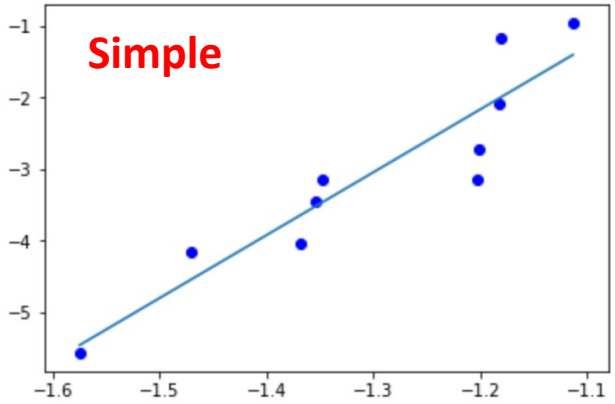
$$\sum_{i=1}^{n} \beta^2 \leq \mathrm{c}$$
$$for\ some\ \mathrm{c} > 0$$

- The penalty term regularizes the coefficients such that if the coefficients take large values the model is penalized.
- Ridge regression shrinks the model's coefficients, and it helps to reduce the model complexity and **multi-collinearity**.

Regularization: Simplify the parameters

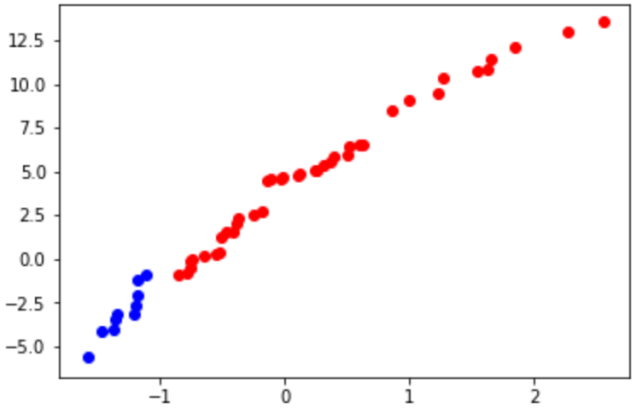


modeling

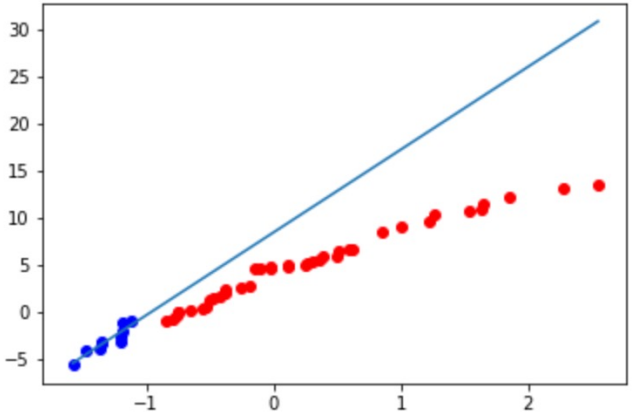


We sample more data

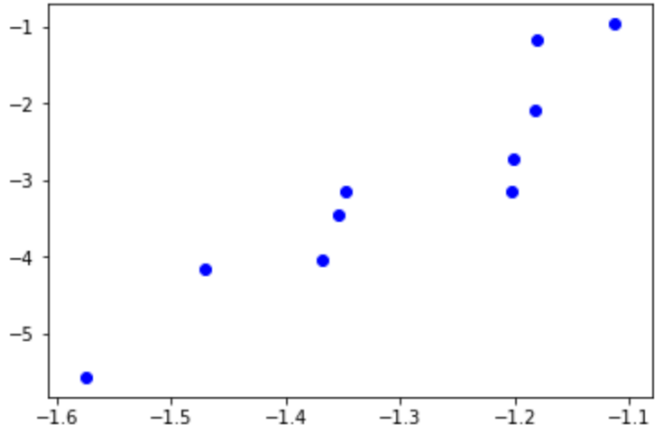
Using the model

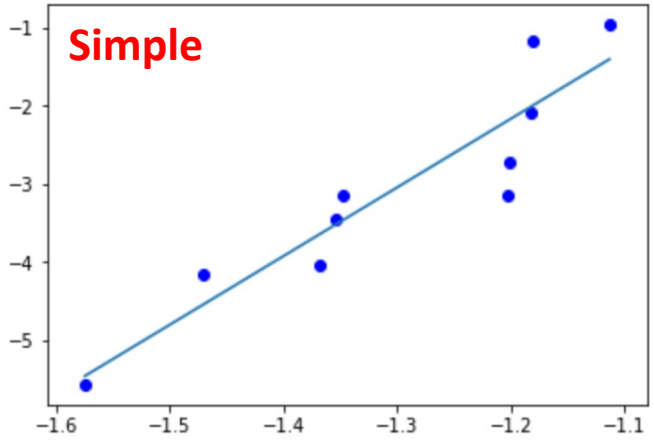


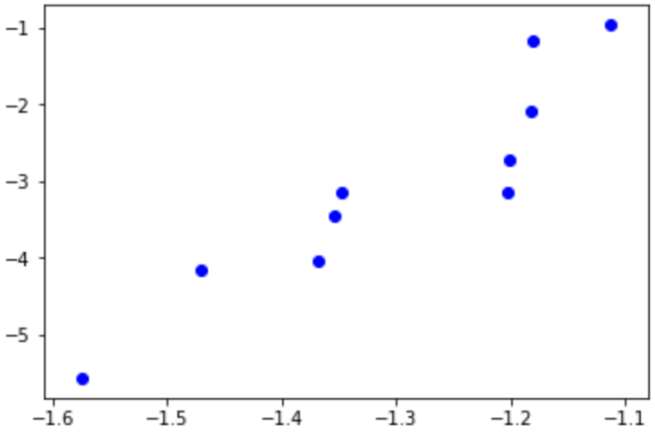
Test



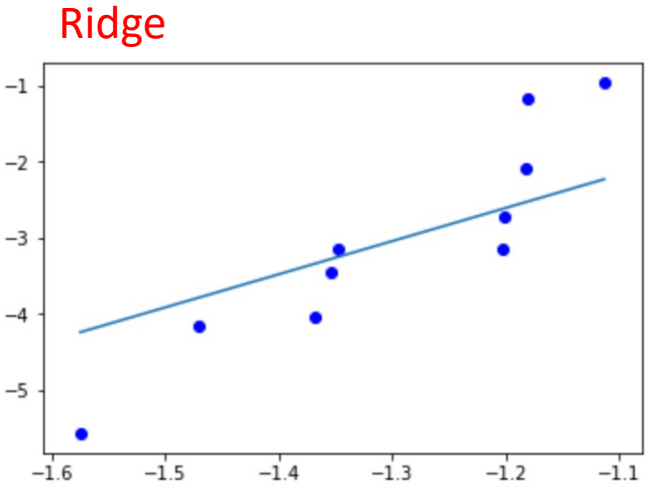
Regularization: Simplify the parameters



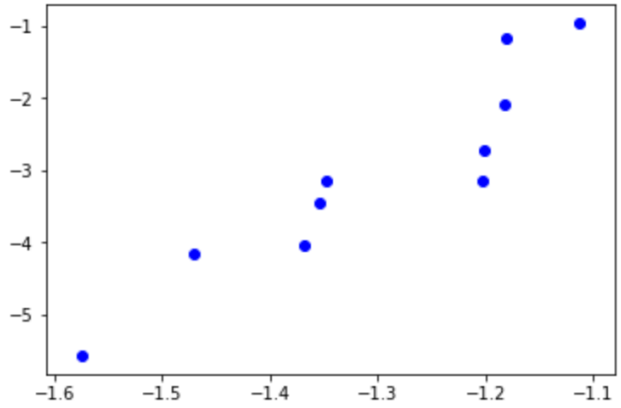




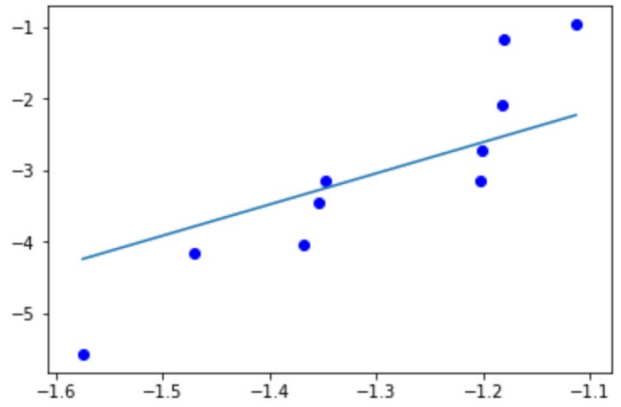
Regularized modeling



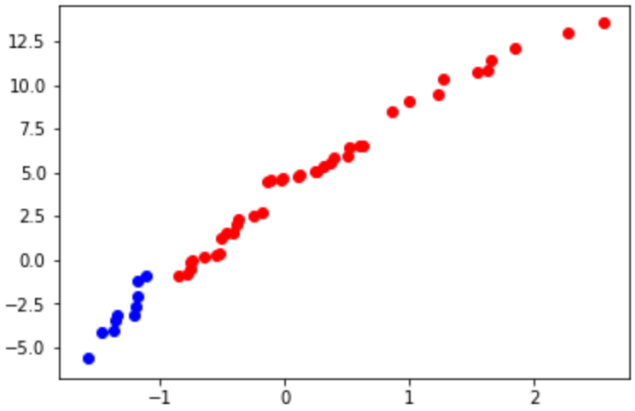
Regularization: Simplify the parameters

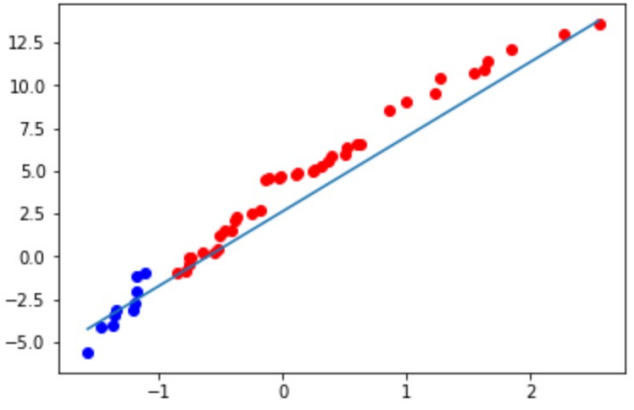


Ridge



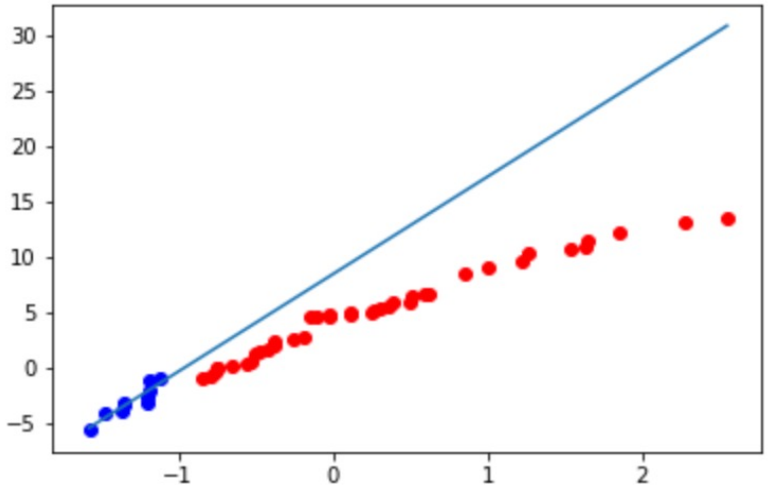
Extension with complete picture



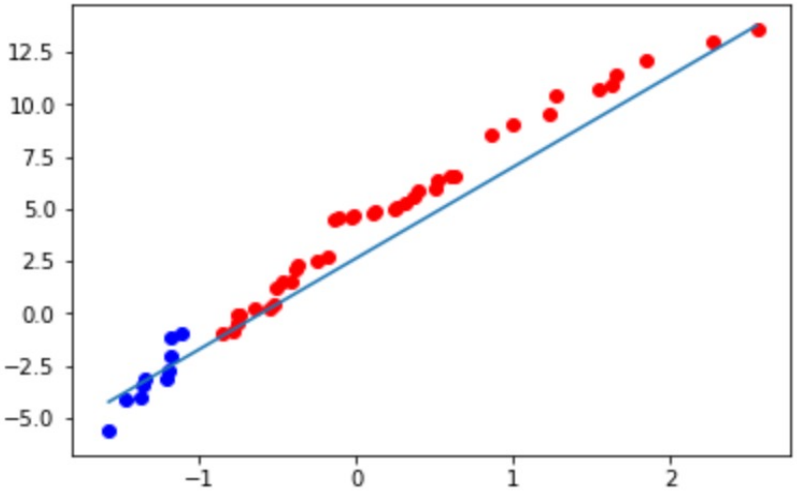


Regularization: Simplify the parameters

Simple



Ridge



Lasso Regression (L1 Regularization)

- The cost function for Lasso (least absolute shrinkage and selection operator) regression can be written as

$$\sum_{i=1}^{n}\left(y_i - (\beta_0 + \beta_1 x_i)\right)^2 + \lambda \sum_{i=1}^{n} |\beta|$$

$$for\ some\ c > 0, \sum_{i=1}^{n} \beta^2 < c$$

- Lasso regression coefficients subject to similar constrain as Ridge.
- Lasso has a different effect on the model parameters, where it **can remove them completely.**

Lasso vs. Ridge Regularization

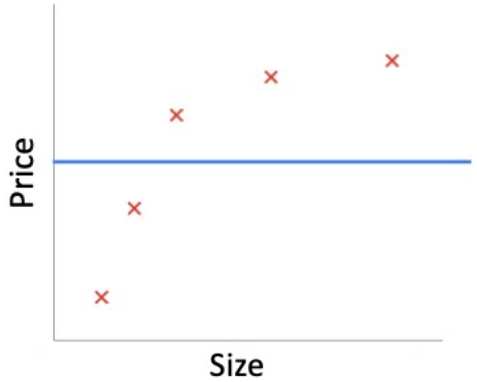
Lasso

$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

(0, 0, 0, 0)

Lasso is slightly better than Ridge in reducing model variance when it contains a lot of useless variables.

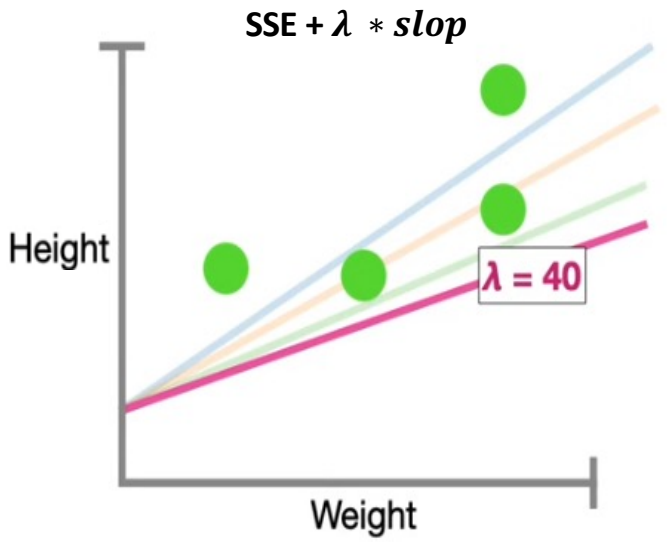
Ridge

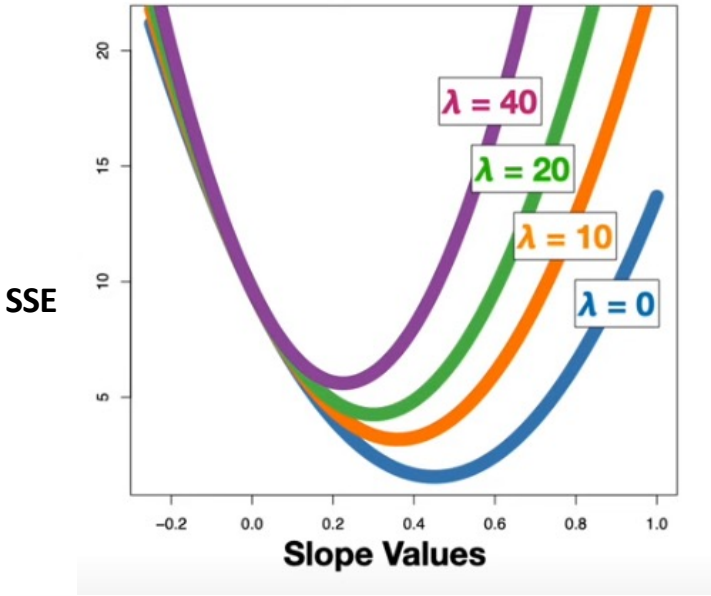


$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

Ridge is slightly better when most variables are useful but correlated.

Ridge - Regularization Effect on a feature

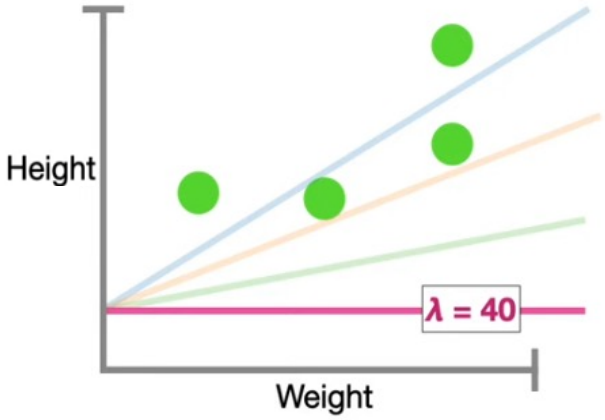


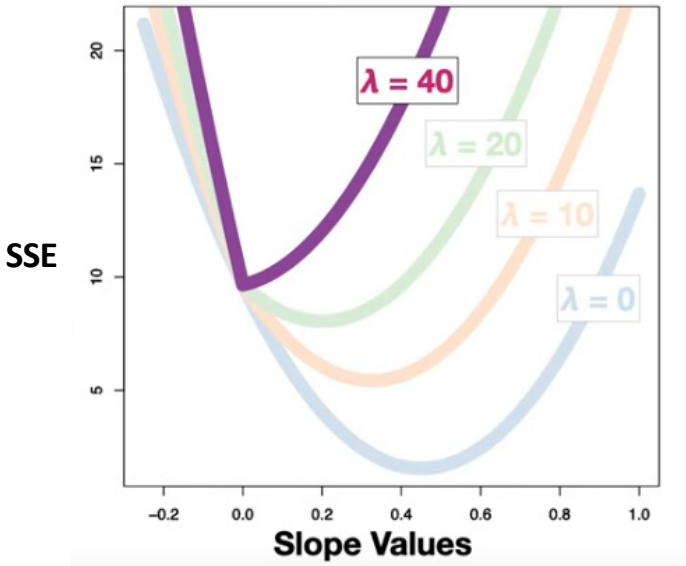


- The error curves are always smooth, even if we hit the model with large $\boldsymbol{\lambda = [\ 0\ to\ 40\ ]}$, the slop never become zero

Lasso - Regularization Effect on a feature

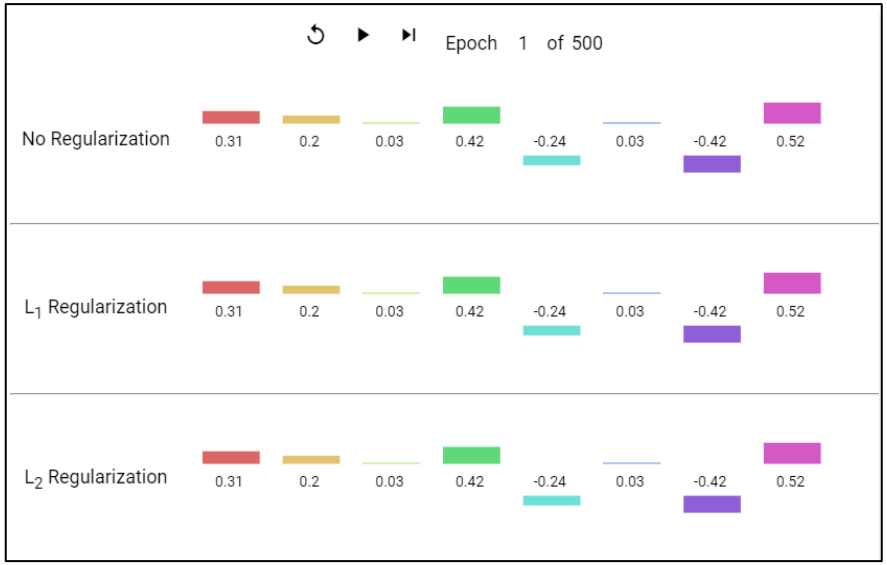
$\mathbf{SSE} + \boldsymbol{\lambda} * \boldsymbol{slop}$

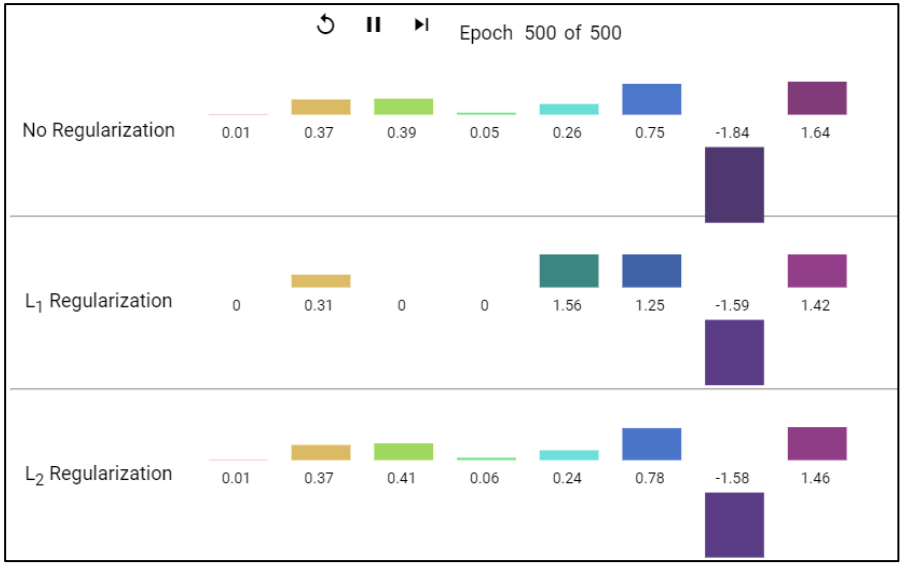




- The error curves are always is not smooth as we hit the model with different $\boldsymbol{\lambda} = [\ \mathbf{0}\ \boldsymbol{to}\ \mathbf{40}\ ]$, the slop may become **zero**

L1 vs. L2 impact on parameters





Reg.html

Elastic Net Regularization

- In case we have a large number of features, where it is hard to determine which regularization we need to use. Elastic Net regularization can be used with the hope to find a good model.
- Elastic Net combines both Lasso and Ridge regularization

$$J(\theta) = \sum_{i=1}^{n} \left((\theta_i x_i + \theta_0) - y_i \right)^2 + \alpha_1 \sum_{i=1}^{n} |\theta_i| + \alpha_2 \sum_{i=1}^{n} \theta_i^2$$

Scikit-learn:

- Ridge and lasso regressions are in scikit-learn **linear_model** package.

```
1 from sklearn.linear_model import Ridge
2 from sklearn.linear_model import Lasso
3 from sklearn.linear_model import ElasticNet
```

```
1 ridge = Ridge(alpha=0.02)
2 ridge.fit(X[:,None], y)
3
4 lasso = Lasso(alpha=0.1)
5 lasso.fit(X[:,None], y)
```

```
1 yr_pred = ridge.predict(X[:,None])
2 yr_pred = lasso.predict(X[:,None])
```

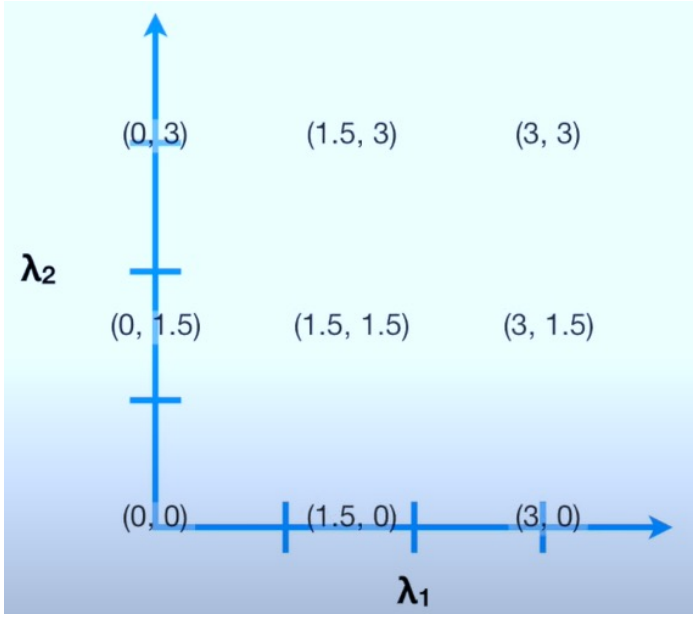
Regularization is sensitive to the scale of data; therefore, it is recommended to **normalize the data**.

How to identify good values of λ_1, λ_1

- We can use cross validation and use different values for λ_1 and λ_2, and pick the pair that produced the best results.

Checkout docs on sklearn

`sklearn.linear_model.ElasticNetCV`



Exercises (10)

- **Regularization using Ridge**
- **Regularization using lasso (optional)**