INFSCI 2915: Machine Learning

Shrinkage Methods – Regularization

Mai Abdelhakim

School of Computing and Information

610 IS Building

Spring 2018

Answers of Previous Class Exercise

- Jupyter notebooks are posted on courseweb

- Linear regression for the advertising dataset with TV, Radio Newspaper

- What is the confidence interval of TV coefficient generated by the code?
  - The 95% confidence interval is: [0.042 : 0.053]

  - The 95% confidence interval does not include zero, which indicates that TV has impact on the advertising

Answers of Previous Class Exercise

- Calculate the MSE with and without Newspaper advertisement, you should find the difference is very low!
  - MSE difference is 0.1

Code that includes all features:

```
from pandas import read_csv
from sklearn.linear_model import LinearRegression

AdvertisingData=read_csv('Advertising.csv')
X = AdvertisingData[['Radio', 'TV','Newspaper']].values
Y = AdvertisingData.Sales

X_train, X_test, Y_train, Y_test= train_test_split(X, Y, random_state= 0)

linreg= LinearRegression().fit(X_train, Y_train)
Target_predicted= linreg.predict(X_test)
MSE=mean_squared_error(Y_test,Target_predicted)
print('mean square error', MSE)
```

Answers of Exercise in Previous Unit

Questions:

A) Use **auto** dataset, and fit a linear model to predict the miles per gallon (**mpg**) from **horsepower**. Find the R^2 metric:

B) Find the R^2 metric when we include both the horsepower feature and **(horsepower)^2**

Optional: increase the degree of the polynomial to 3,then 4, then 5 and check the accuracy in each case

C). Repeat with KNN

Solution: with random_state=0 (code next slide)

With polynomial of degree 1 the R squared score of linear regression is: 0.62176588114
With polynomial of degree 2 the R squared score of linear regression is: 0.727103150464
With polynomial of degree 3 the R squared score of linear regression is: 0.728238860119
With polynomial of degree 4 the R squared score of linear regression is: 0.729574758258
With polynomial of degree 5 the R squared score of linear regression is: 0.732051116484
Comments:

- Performance improves by adding quadratic feature to the linear regression model
- KNN performs better that linear regression with a single feature (horsepower)
- Linear regression performs better than KNN when the non-linear terms are added..

```
AutoData=read_csv('Auto_modify.csv')

X_auto_hp=AutoData.horsepower.values.reshape(-1,1)
Y_auto_mpg=AutoData.mpg.values.reshape(-1,1)

modelAuto2=LinearRegression()
X=X_auto_hp
for power in [1,2,3,4,5]:
    if power>1:
        X=np.concatenate((X,X_auto_hp**power),axis=1)
    X_train, X_test, Y_train, Y_test= train_test_split(X, Y_auto_mpg, random_state= 1)

)

    Auto_fitted_model2=modelAuto2.fit(X_train_transformed,Y_train)
    R2_auto_hp_RegScale=Auto_fitted_model2.score(X_test_transformed,Y_test)
    print('With polynomial of degree', power, 'R squared score of linear regression with scaling is:', R2_auto_hp_RegScale)
```

Objectives of this Unit

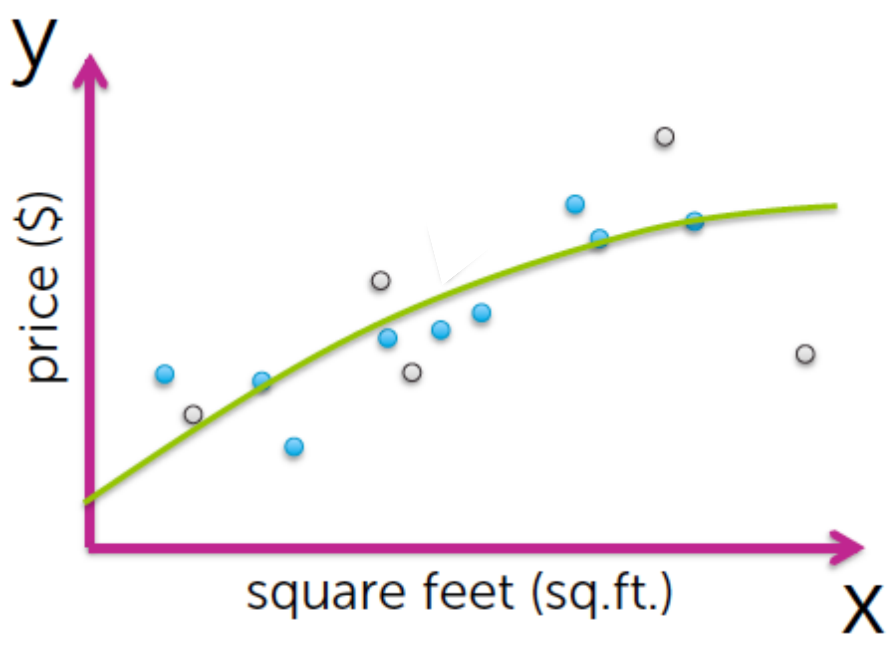
- Shrinkage methods:
  - Ridge regression
  - Lasso regression

Impact of Number of Features

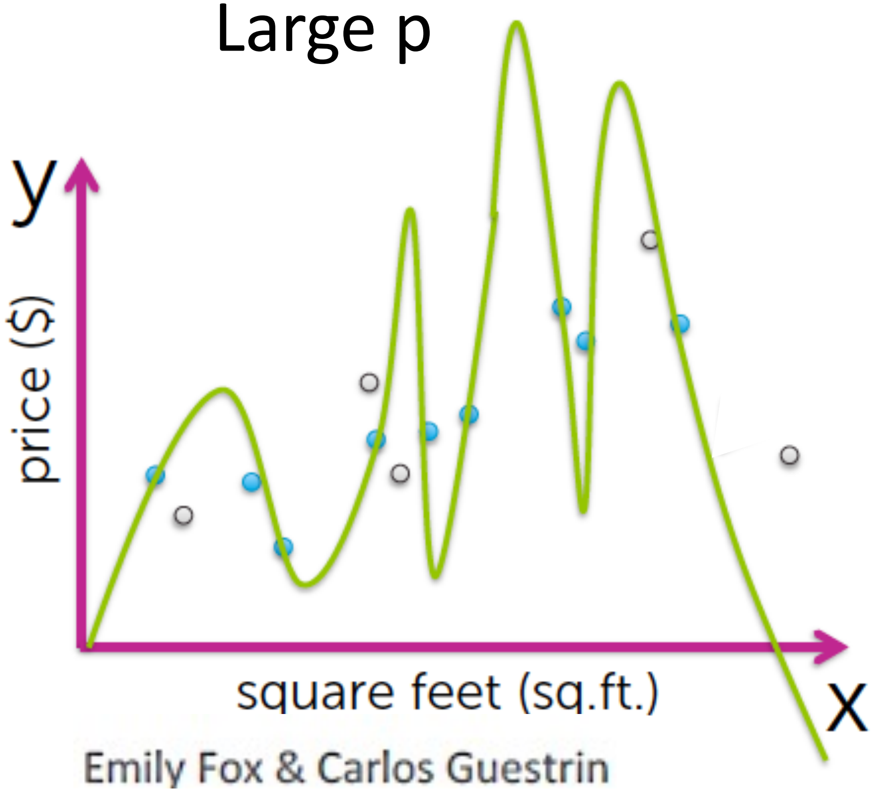
- We can define a polynomial regression function with p features as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2\, X_1^2 + \beta_3\, X_1^3 + \ldots + \beta_p\, X_1^p$$

Small p



Large p

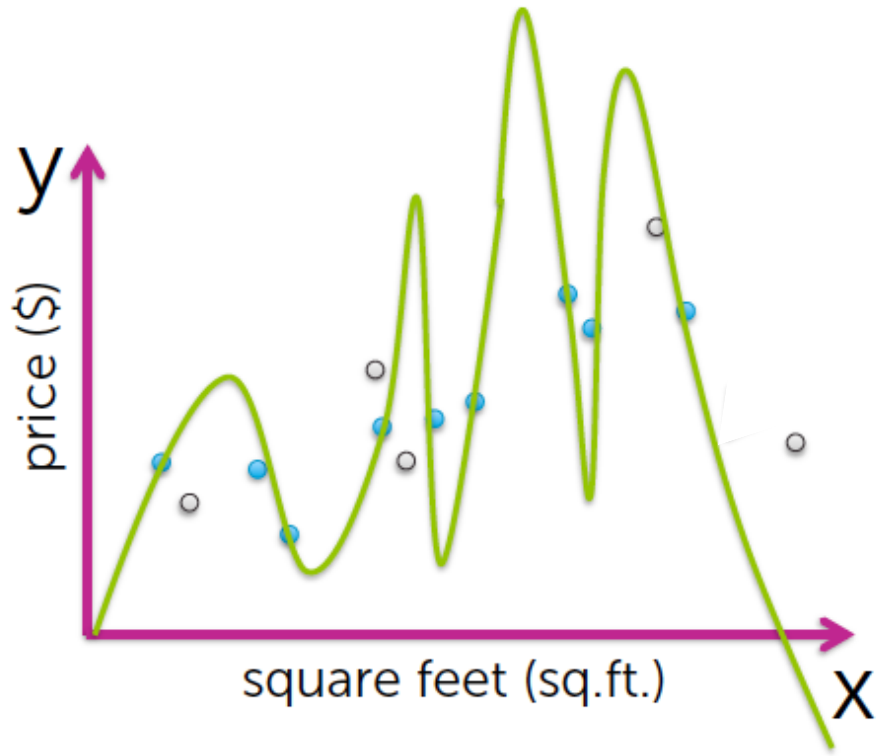


Impact of the Number of Observations

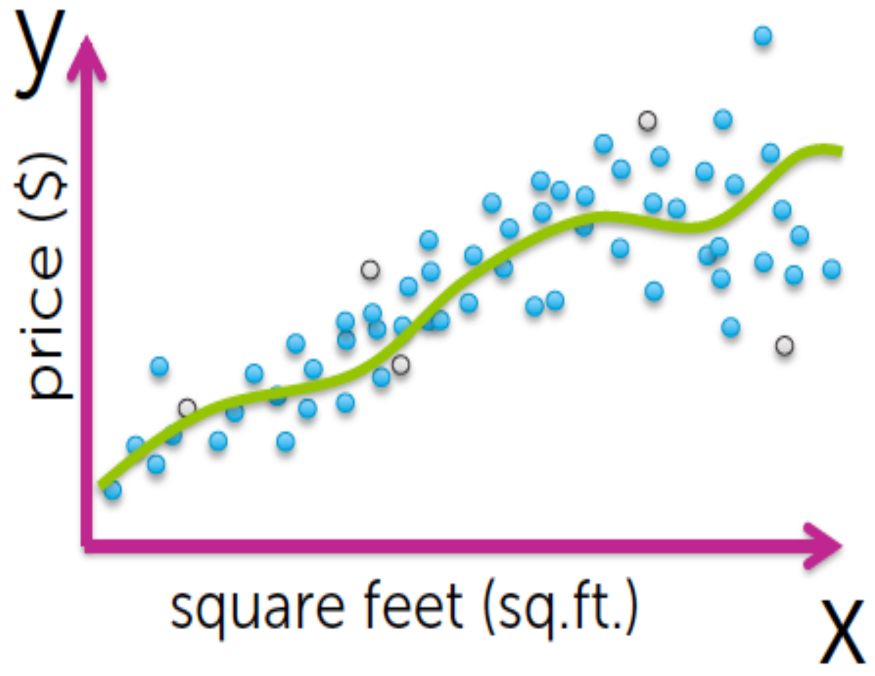
- Needs a lot of observations to avoid overfitting

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_1^3 + \ldots + \beta_p X_1^p$$

Large p, small n



Large p, large n



Emily Fox & Carlos Guestrin

- Same phenomena applies when there are many features in a linear regression model without using polynomials
  - We need number of features p << n

- Accuracy: if number of features (p) is greater than number of observations, accuracy will degrade (large variance).
  - We need data that reflects all possible combinations between the features and the response

- Interpretability: if we remove irrelevancy features, the model can be interpreted easily

- Can we do better with linear regression?
  - **Can we include large number of features, without overfitting?**

- Can we replace the ordinary least square fitting by another fitting that solve this problem?

Feature Selection

- Recall the concept of feature selection methods:
  - Best subset: search over all possible combinations of features
  - Forward selection
  - Backward selection
  - Mixed selection

- We can use the above methods and least squares fit to find a good subset of features

- Alternatively, fit and single model and include all features , but use a technique that shrinks some coefficient estimates towards zero. (why zero?)
  - This is the main idea behind **Ridge and Lasso regression**

Ridge Regression

- Ordinary Least Squares (OLS) estimates the coefficients by minimizing

$$\mathrm{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

- Ridge Regression, also called L_2 *regularization (as it uses the L2 norm),*
  - Modifies the objective function (that needs to be minimized) to

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

λ is a tuning parameter

$$= \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2$$

Shrinkage penalty

➔ ***L_2 norm of coefficients (excluding $\boldsymbol{\beta_0}$)***

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$$

- The first term: Ridge regression tries to find coefficient estimate that **minimizes** the **RSS** (same as least squares)
  - To better fit to the training data
- The second term is called shrinkage penalty, as it has the effect of **shrinking** coefficients towards **zero**
  - To avoid overfit by reducing the variance of the fitted model
- **λ is a tuning parameter** $(\lambda \geq 0)$ controls the **relative impact of these two terms**
  - Selection of this parameter can be made through **cross-validation** (discussed later)

- The objective function to minimize is: $\mathbf{J(\boldsymbol{\beta}) = RSS(\boldsymbol{\beta}) + \boldsymbol{\lambda} \sum_{j=1}^{p} \boldsymbol{\beta}_j^2}$

- **If λ=0** ➔ J(β)= RSS(β) , same least squares solution as before
  - May result in overfitting

- **If λ is very large (λ=∞)** ➔ minimizing J(β) will result in setting all coefficients to zero (low magnitude)
  - This results in underfitting

Finding Coefficients

The optimal solution can be obtained by:

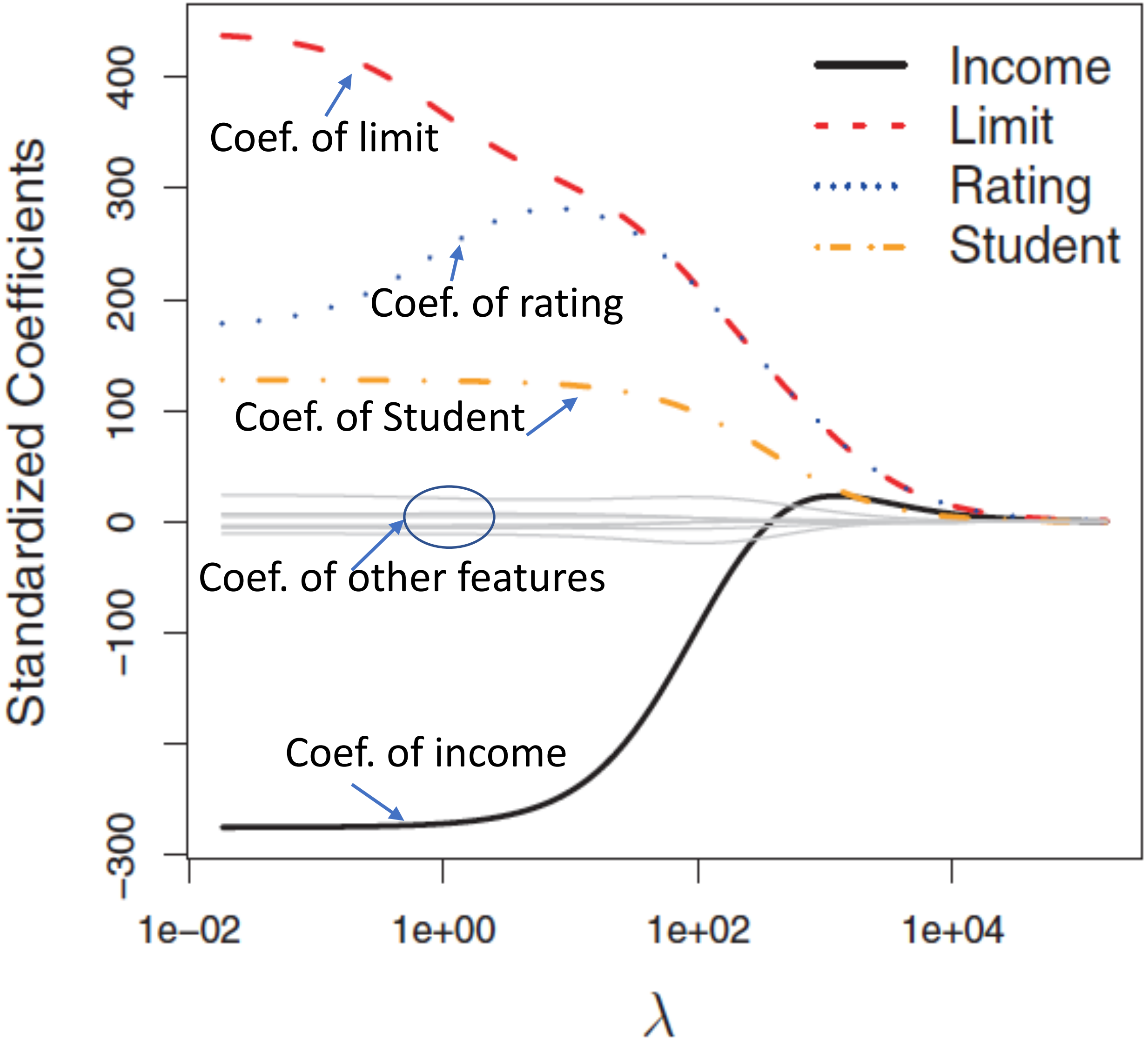
- Close-form solution: $\frac{\partial \mathrm{J}(\beta)}{\partial \beta}$ =0 ➔ $\hat{\beta}$ =$(X^T X + \lambda\, I_m)^{-1}\, X^T\, y$
  - I_m is the (p+1)x (p+1) identity matrix with first row all zeros, and rest of rows have ones on diagonal elements
    - For example, if p=2, then $I_m = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Gradient descent, same iterative procedure as described before

Example: Credit dataset

- Credit data set: Records balance (average credit card debt for a number of individuals), *age* ,number of *cards*, years of *education*, income, credit *limit*, *student* status, and credit *rating*, other features
- Using ridge regression with different values of λ
  - Figure shows the change of coefficient with λ
    - λ close to zero ➔ least square estimates
    - λ large ➔ coefficient shrinks to zero

Standardized coefficient are the coefficient estimate when features are scaled to have unit variance



Comments about Feature Scaling

- Feature scaling may not be critical for ordinary least square with closed-form solution
  - Scaling the feature (multiply by constant), scales the coefficient (multiply by 1/constant)
  - However this would impact the interpretability, and hence scaling is still recommended

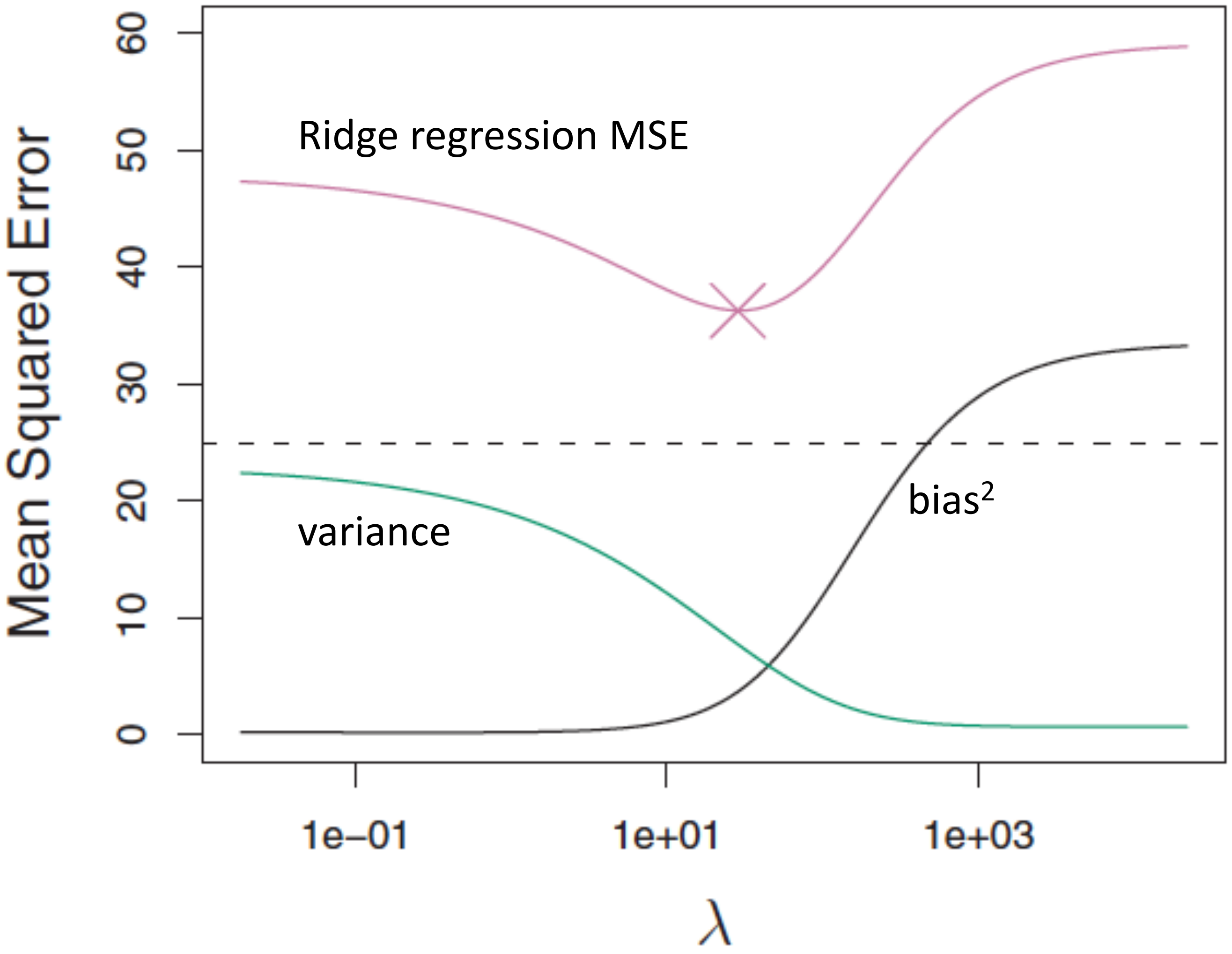
- Scaling is important if gradient descent is used

- With Ridge regression, features need to be on the same scale (feature scaling is recommended)

How Does that Solve Overfitting?

Figure shows simulated data with n=50 training examples and p=45 features

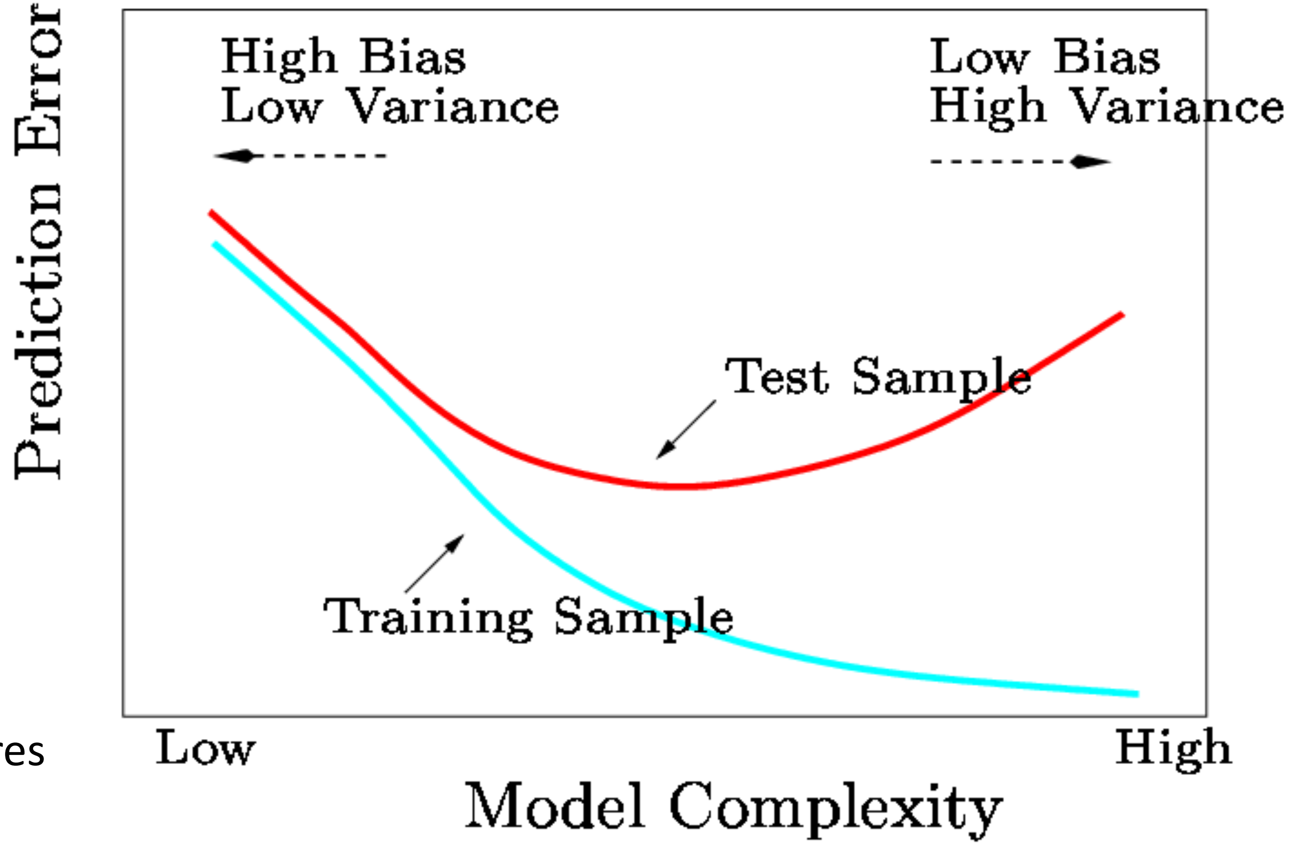
The shrinkage parameter is selected to achieve good bias-variance tradeoff



Ridge regression works in situations where OLS has high variance (p≈n or p > n)

Bias-Variance Tradeoff

- λ **increases** => flexibility of the model decreases (**less complex**)
  - At extreme case with very large λ : no features will be included (simple/trivial model)
- Ridge regression works in situations where OLS has high variance (p≈n or p > n)



Here, complexity is measured by number of features

Ridge Regression

- Advantages:
  - **Reduce variance**, avoid overfitting when p is large
  - Fit **single model**
- Disadvantages:
  - All coefficients shrink towards zero, but non of them will be set exactly to zero (if $\lambda \neq \infty$)
    - Will **not exclude any feature**
      - Credit card data: Ridge will always include all 10 features instead of selecting the most relevant ones
    - **Challenge in the model interpretation**

Lasso Regression

- Tries to overcome disadvantages of Ridge regression
- Modifies the objective function to use the L_1 norm instead of the L_2 norm

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda \boxed{\sum_{j=1}^{p}|\beta_j|}$$

$$= \mathrm{RSS} + \lambda \sum_{j=1}^{p}|\beta_j|$$

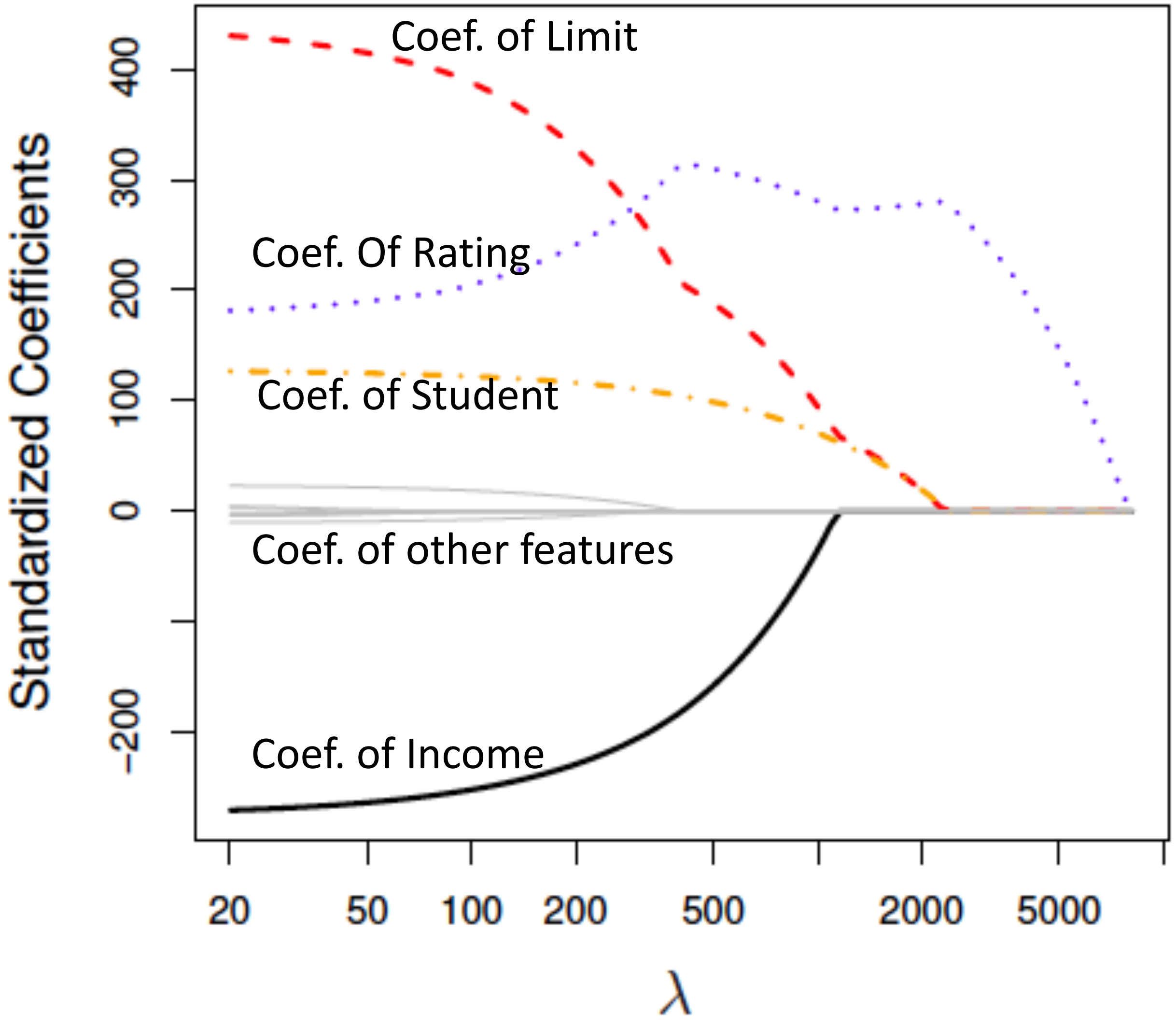
$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$

- When the **tuning parameter** (λ) is sufficiently **large**, **<u>some</u> coefficients will be forced to be zero**
  - Equivalent to feature selection
  - Easy to interpret

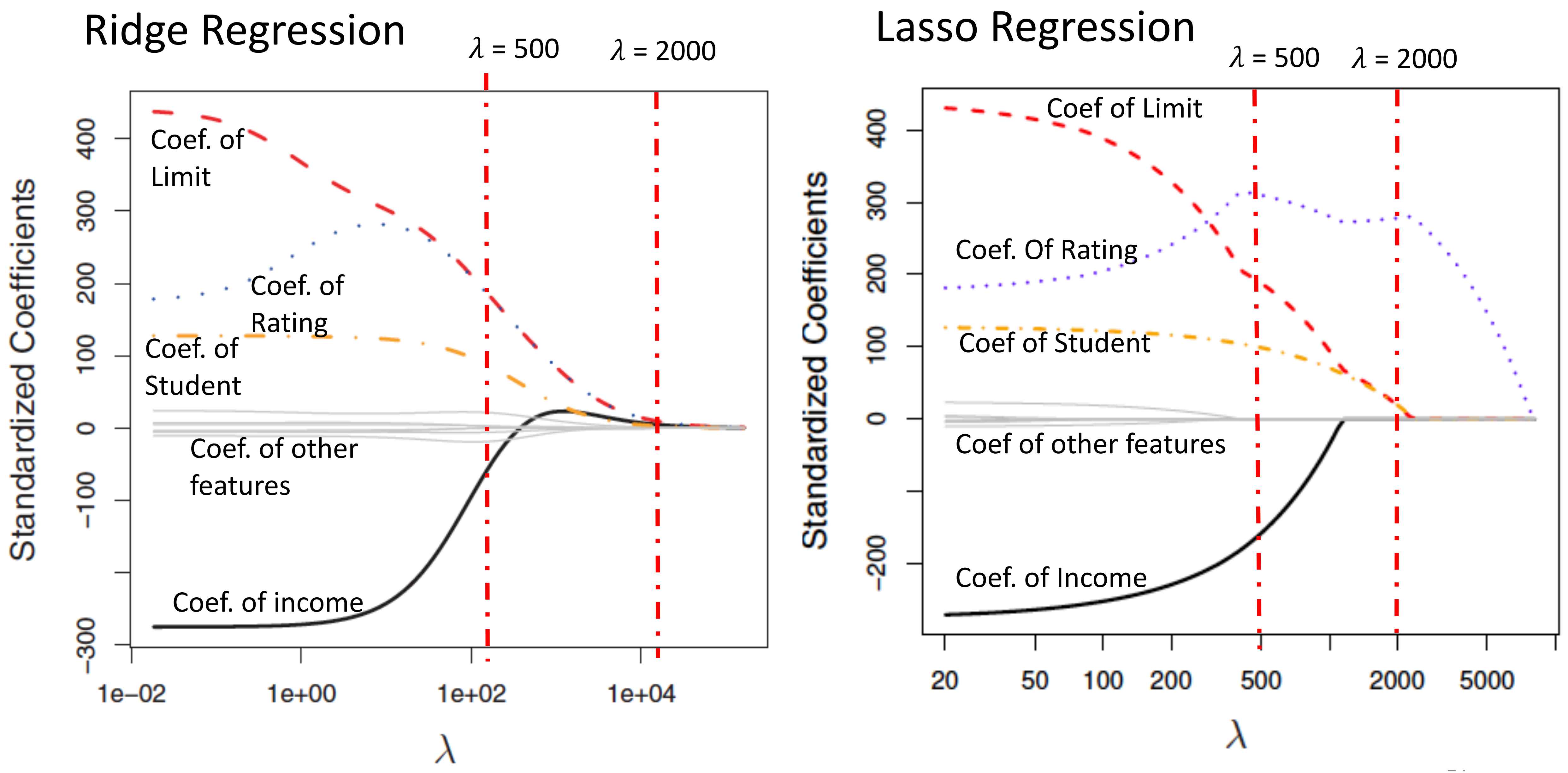
- Called sparse model, as it contains subset of features

Example: Credit Dataset with Lasso Regression

- Features: Limit, Income, Rating Student, other features
- Apply the Lasso to the credit data set
  - λ close to zero ➔ least square estimates
  - λ large ➔ coefficient shrinks to zero
- For a given λ, subset of features can be selected, and other coefficients are set to zero



Example: Compare Ridge and Lasso



Ridge vs Lasso

- **Lasso** performs better when **small number of features are in fact related to the response** (have substantial coefficients)

- **Ridge** performs better when the **response is a function of all features** (coefficients of roughly equal size)
  - All contribute to response with a small amount

- But the number of features that are related to response is typically unknown

- Cross-validation can be used to find which approach works better on a particular data set

Ridge Regression in Python

- Default value for tuning parameter (called alpha in python) is $\lambda = 1$

```
from sklearn.linear_model import Ridge

# train and fit the ridge regression model with training data
RidgeModel=Ridge( ).fit(X_train, Y_train) # this uses default alpha of 1

#find the R2 metric with the .score
RidgeModel.score(X_test,Y_test)
```

- To specify a value of λ (referred to as alpha in python): for example set $\lambda = 10$
  - RidgeModel10=**Ridge(alpha=10)**.fit(X_train, Y_train)

Lasso Regression in Python

- Default value for tuning parameter (called alpha in python) is λ = 1

  from **sklearn.linear_model** import **Lasso**

  lassoModel=**Lasso( ).**fit(X_train, Y_train)

- Update the tuning parameter to 0.01

  LassoModel001=Lasso(alpha=0.01). fit(X_train, Y_train)

- Use the .score method to get the performance

- You can find number of coefficients that are equal to zero using: numpy.sum(LassoModel001.coef_==0)

Exercise

- A) Use the Boston dataset, and use Ridge regression model with tuning parameter set to 100 (alpha =100). Find the R^2 score and number of non zero coefficients.
- B) Use Lasso regression instead of Ridge regression, also set the tuning parameter to 100. Find the R^2 score and number of non zero coefficients.
- C) Change the tuning parameter of the Lasso model to a very low value (alpha =0.001). What is the R^2 score.
- D) Comment on your result. In this problem, do all feature seem important in making predictions?