10-315 Intro to Machine Learning (SCS Majors)

Lecture 7: Linear Regression

Leila Wehbe

Carnegie Mellon University

Machine Learning Department

Reading: **Elements of Statistical Learning Chapters 3.1, 3.2, 3.4**.

Lecture outcomes

- Linear regression definition
- OLS solution and interpretation
- Ridge solution and tradeoff
- Lasso (just the penalty and what it optimizes for)

Links (use the version you need)

- **[Notebook]**
- **[PDF slides]**

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm, multivariate_normal

# Sample data:
X = np.linspace(0,36,36)
coefs = [2,0.5]
Y = coefs[0] + coefs[1]*X + 2*norm.rvs(size = np.shape(X))
Y_seasonal = coefs[0] + coefs[1]*X + 4*np.sin(X*np.pi/6) + 0.5*norm.rvs(size = np.shape(X))
```

Regression vs. Classification

- So far, we've been interested in learning $P(Y|X)$ where Y has discrete values ('classification')
- What if Y is continuous? ('regression')
    - predict weight from gender, height, age, …
    - predict Google stock price today from Google, Yahoo, MSFT prices yesterday
    - predict each pixel intensity in robot's next camera image, from current image and current action

Linear regression

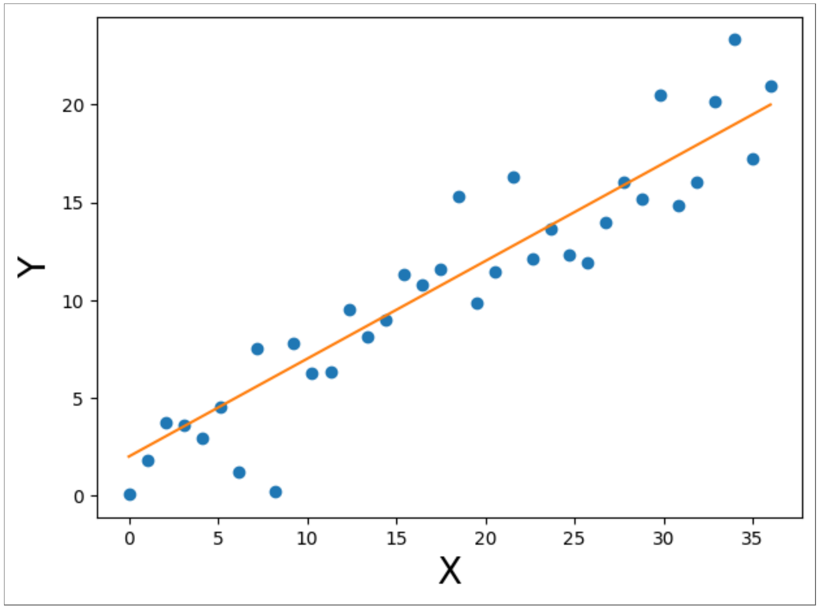
- We wish to learn a linear function $f : \mathbf{x} \rightarrow y$ where $y \in \mathbb{R}$ given $\{(\mathbf{x}^{(1)}, y^{(1)}), \ldots (\mathbf{x}^{(n)}, y^{(n)})\}$ with $\mathbf{x}^{(i)} \in \mathbb{R}^p$.
- Let's start with 1-dimensional x example (p=1):
    - We want to find the line that best "fits" the data
        - How do we define this best fit?

In [2]:

```
plt.figure(figsize=(7,5))
plt.plot(X,Y,'o');plt.xlabel('X', fontsize=20);plt.ylabel('Y',fontsize=20);
plt.plot(X,coefs[0]+coefs[1]*X)
```

Out[2]:

```
[<matplotlib.lines.Line2D at 0x12dab03d0>]
```



Linear Regression

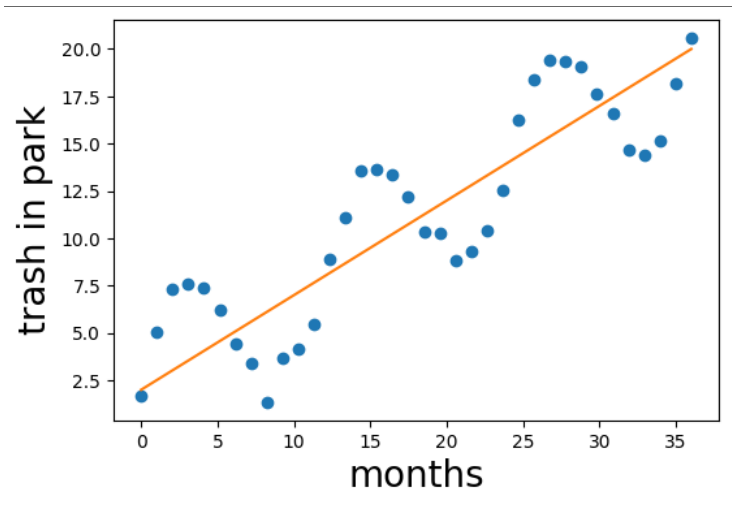
- Consider the example below: there exist a non-linear relationship that is a very good predictor of the data
  - there is a seasonal effect that varies with the months of the year, as well as a linear increase with time
  - linear regression is only able to capture linear relationships

In [3]:

```
plt.figure(figsize=(6,4))
plt.plot(X,Y_seasonal,'o');plt.xlabel('months', fontsize=20);plt.ylabel('trash in park',fontsize=20);
plt.plot(X,coefs[0]+coefs[1]*X)
```

Out[3]:

```
[<matplotlib.lines.Line2D at 0x12dc48ed0>]
```



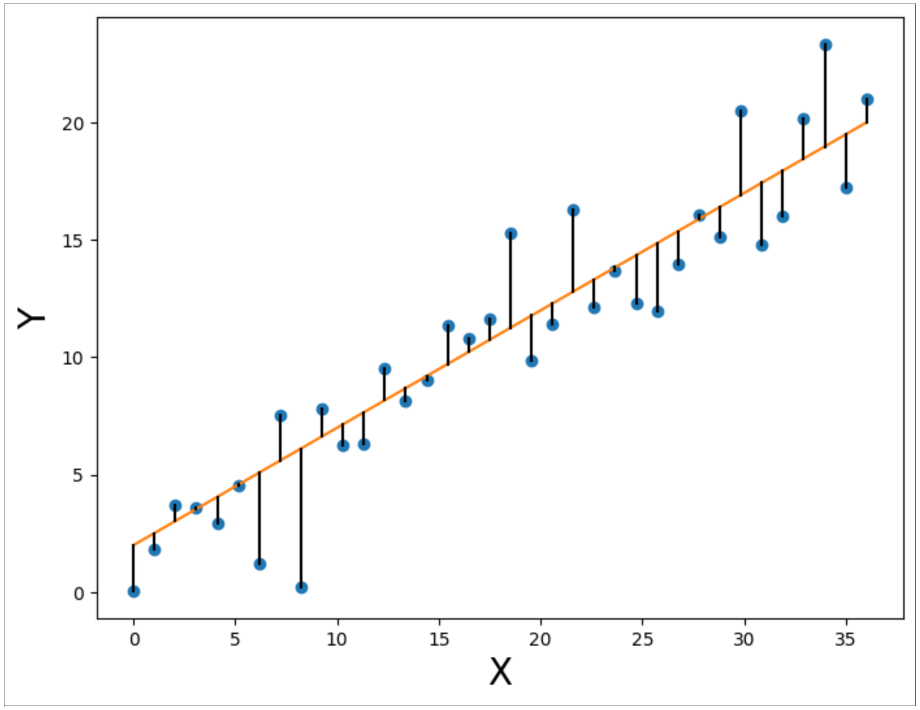
- even if the underlying relationship between X and Y is not linear, one can still use linear regression
    - the assumption is not satisfied, but we have seen previously that in some cases, a model can still perform "well" even if its assumptions are not specified

How to define goodness of fit?

- We define the goodness of fit based on the prediction error:
    - $\epsilon^{(i)} = y^{(i)} - \hat{y}^{(i)} = y^{(i)} - (w_0 + w_1 x^{(i)})$
    - vertical error in the plot below

In [4]:

```
plt.figure(figsize=(8,6))
plt.plot(X,Y,'o');plt.xlabel('X', fontsize=20);plt.ylabel('Y',fontsize=20);
plt.plot(X,coefs[0]+coefs[1]*X)
for i,Xi in enumerate(X):
    plt.plot( [Xi,Xi], [coefs[0]+coefs[1]*Xi, Y[i]],'k')
```



X In 2-D:

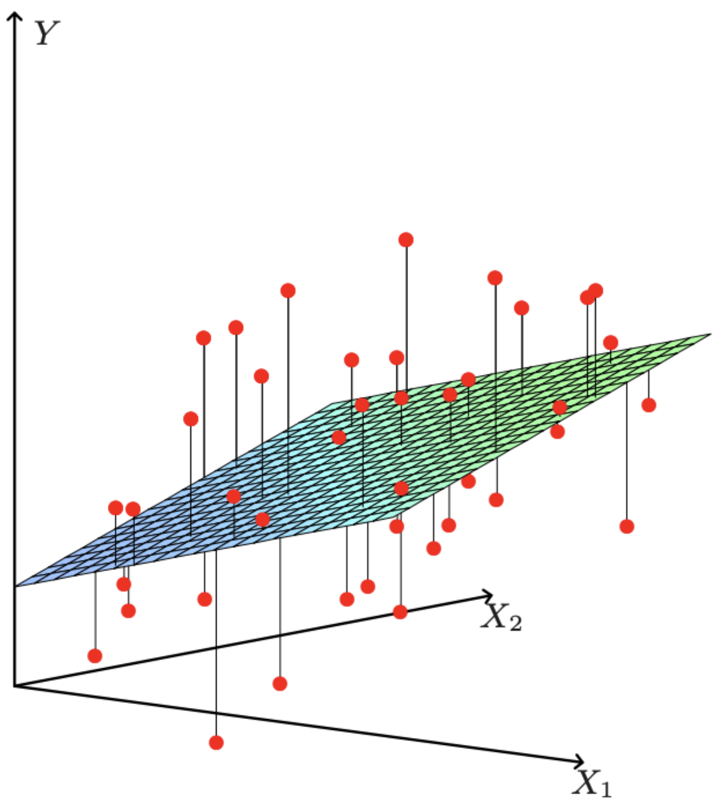


FIGURE 3.1. *Linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of X that minimizes the sum of squared residuals from Y.*

Source: Figure 3.1 from **ESL**

Approach 1: Minimizing the Residual Sum of Squares

- The Residual Sum of Squares (RSS) is:

$$\mathrm{RSS}(\mathbf{w}) = \sum_{i=1}^{n} \left(y^{(i)} - (w_0 + \sum_{j} w_j x_j^{(i)}) \right)^2 \tag{1}$$

- This corresponds to the sum of the square of the errors in predicting each $y^{(i)}$.
- If we change our notation so that now $\mathbf{x}^{(i)}$ has an additional entry $x_0^{(i)}$ always corresponding to 1:

$$\mathrm{RSS}(\mathbf{w}) = \sum_{i=1}^{n} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)} \right)^2 \tag{2}$$

- Note that the Mean Squared Error (MSE) is also often used in regression problems and corresponds to RSS/n. It should be clear that here minimizing MSE or RSS wields the same solution

How to minimize RSS?

- The Ordinary Least Squares (OLS) solution minimizes RSS:

$$\hat{\mathbf{w}}_{\text{OLS}} = \underset{\mathbf{w}}{\operatorname{argmin}} \operatorname{RSS}(\mathbf{w}) = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)} \right)^2 \tag{3}$$

- Let's write RSS in matrix notation:

$$\operatorname{RSS}(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) \tag{4}$$

```
- Where:
```

$$\mathbf{X} = \begin{bmatrix} {\mathbf{x}^{(1)}}^\top \\ {\mathbf{x}^{(2)}}^\top \\ \dots \\ {\mathbf{x}^{(n)}}^\top \end{bmatrix} = \begin{bmatrix} x_0^{(1)} & x_1^{(1)} & \dots & x_p^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \dots & x_p^{(2)} \\ \dots & \dots & \dots & \dots \\ x_0^{(n)} & x_1^{(n)} & \dots & x_p^{(n)} \end{bmatrix} \quad \text{and} \quad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \dots \\ y^{(n)} \end{bmatrix} \tag{5}$$

RSS(**w**) is convex in **w**

$$\frac{d\mathrm{RSS}(\mathbf{w})}{d\mathbf{w}} = \frac{d(\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w})}{d\mathbf{w}} \tag{6}$$

Poll: what is the size of $\frac{d\mathrm{RSS}(\mathbf{w})}{d\mathbf{w}}$?

- RSS is a scalar
- $\mathbf{w}$ is $(p \times 1)$

$$\frac{d\mathrm{RSS}(\mathbf{w})}{d\mathbf{w}} = -2\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) = -2\left(\mathbf{X}^\top \mathbf{y} - \mathbf{X}^\top \mathbf{X}\mathbf{w}\right) \tag{7}$$

$$\frac{d\mathrm{RSS}(\mathbf{w})}{d\mathbf{w}} = -2\mathbf{X}^\top\left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) = -2\left(\mathbf{X}^\top\mathbf{y} - \mathbf{X}^\top\mathbf{X}\mathbf{w}\right) \tag{8}$$

$$\frac{d\mathrm{RSS}(\mathbf{w})}{d\mathbf{w}}\Big|_{\hat{\mathbf{w}}_{\mathrm{OLS}}} = 0 \tag{9}$$

- if $\mathbf{X}^\top\mathbf{X}$ is invertible*:

$$\hat{\mathbf{w}}_{\mathrm{OLS}} = \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y} \tag{10}$$

- Predict for new point $\mathbf{x}^{new}$: $\hat{y}^{new} = \mathbf{x}^{new\top}\hat{\mathbf{w}}_{\mathrm{OLS}}$

*For a review: **Zico Kolter's Linear algebra notes**

Alternative geometric interpretation

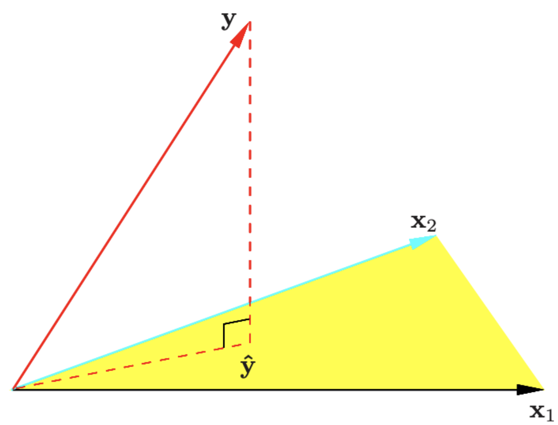


FIGURE 3.2. *The N-dimensional geometry of least squares regression with two predictors. The outcome vector $\mathbf{y}$ is orthogonally projected onto the hyperplane spanned by the input vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. The projection $\hat{\mathbf{y}}$ represents the vector of the least squares predictions*

Source: Figure 3.2 from **ESL**

- In this representation, $\mathbf{y}$ represents the real values for all points, and $\mathbf{x}_1$ and $\mathbf{x}_2$ are **columns** of $\mathbf{X}$.
- $\hat{\mathbf{y}}$ is the vectors of all predictions that lie in the space spaned by $\mathbf{x}_1$ and $\mathbf{x}_2$.
- $\hat{\mathbf{y}}$ is the orthogonal projection of $\mathbf{y}$ onto that space and $\mathbf{y} - \hat{\mathbf{y}}$ is the error.

Alternative geometric interpretation - more formally:

- Assume we have n tuples $(\mathbf{x}^{(i)}, y^{(i)})$ where $\mathbf{x}^{(i)} \in \mathbb{R}^p$ and $y^{(i)} \in \mathbb{R}$, $\mathbf{y} \in \mathbb{R}^n$.
- $\mathbf{X}$ is $n \times p$.
- The p columns of $\mathbf{X}$ span a subset of $\mathbb{R}^n$
    - Recall from linear algebra this subset is called the column space of $\mathbf{X}$

- The vector of predictions for all points $\hat{\mathbf{y}}$ is the orthogonal projection of $\mathbf{y}$ onto the linear subspace spanned by the columns of $\mathbf{X}$.
    - Recall this is due to our optimization procedure, in which we set:

$$\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) = 0$$

    - (the error is orthogonal to the space spanned by $\mathbf{X}$)

What happens if $\mathbf{x}^\top \mathbf{x}$ not invertible?

- Suppose $\mathbf{X}$ is not full rank, i.e. it's columns are not linearly independent
  - e.g. two of the input dimensions are perfectly correlated
  - e.g. one of the input dimensions is a linear combination of the others
  - or e.g. $p > n$
- Then, $\mathbf{X}^\top \mathbf{X}$ is singular and we cannot invert it.
  - there is not a unique solution $\hat{\mathbf{w}}_{\mathrm{OLS}}$
- Solutions to the problem: remove redundancy from $\mathbf{X}$, regularize (will discuss in a moment), add diagonal component (akin to specific type of regularization, to see later)...

What if $\mathbf{x}^\top \mathbf{x}$ is invertible but too large?

- Inverting $\mathbf{X}^\top \mathbf{X}$ might still be very slow!

- Can use a matrix decomposition to speed up inversion (e.g., SVD or Cholesky).

- Can do gradient descent:
  - initialize $\mathbf{w}^0$
  - update:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - 2\eta \mathbf{X}^\top \left(\mathbf{X}\mathbf{w}^t - \mathbf{y}\right)$$

    - The error $\mathbf{X}\mathbf{w}^t - \mathbf{y}$ reduces as $\mathbf{w}^{t+1}$ gets close to $\hat{\mathbf{w}}_{\mathrm{OLS}}$
    - convergence depends on learning rate (too small ==> slow, too big ==> possible oscillation and larger error if can't get close enough to $\hat{\mathbf{w}}_{\mathrm{OLS}}$. Can use adaptative learning rate.)

Probabilistic interpretation: MLE

- We state the problem as:

$$y^{(i)} = \mathbf{x}^{(i)\top}\mathbf{w} + \epsilon^{(i)} \tag{11}$$

$$\epsilon^{(i)} \sim \mathcal{N}(0, \sigma) \tag{12}$$

$$Y^{(i)} \sim \mathcal{N}(\mathbf{x}^{(i)\top}\mathbf{w}, \sigma) \tag{13}$$

- Maximizing the log-likelihood of the data simplifies to:

$$\hat{w}_{\mathrm{MLE}} = \underset{\mathbf{w}}{\operatorname{argmax}} \ln\left[\prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp\frac{-(y^{(i)} - \mathbf{x}^{(i)\top}\mathbf{w})^2}{2\sigma^2}\right] \tag{14}$$

$$= \underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - \mathbf{x}^{(i)\top}\mathbf{w}\right)^2 \tag{15}$$

This is the OLS problem! ==> same solution

Approach 2 - Ridge regression, adding L2 regularization

- Ridge regression minimizes the RSS with an additional penalty on the ℓ_2 norm of $\mathbf{w}$:

$$\hat{\mathbf{w}}_{\text{Ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y^{(i)} - {\mathbf{x}^{(i)}}^{\top} \mathbf{w} \right)^2 + \lambda \sum_{j} w_j^2 \tag{16}$$

- where $\lambda \geq 0$ is a penalty parameter

- Note: in practice, we often don't penalize the intercept term. Instead we first estimate the intercept as $\bar{\mathbf{y}}$ and remove it from $\mathbf{y}$, and run ridge regression with no intercept. Then we set the intercept as $\bar{\mathbf{y}}$.

- In matrix notation: $\hat{\mathbf{w}}_{\text{Ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left(\mathbf{y} - \mathbf{X}\mathbf{w} \right)^{\top} \left(\mathbf{y} - \mathbf{X}\mathbf{w} \right) + \lambda \mathbf{w}^{\top}\mathbf{w}$

- Solving

$$\frac{d\text{RSS}(\mathbf{w}) + \lambda \mathbf{w}^{\top}\mathbf{w}}{d\mathbf{w}} = -2\mathbf{X}^{\top} \left(\mathbf{y} - \mathbf{X}\mathbf{w} \right) + 2\lambda \mathbf{w} = 0 \tag{17}$$

$$\left(\mathbf{X}^{\top}\mathbf{X} + \lambda \mathbf{I}_p \right) \mathbf{w} = \mathbf{X}^{\top}\mathbf{y} \tag{18}$$

$$\hat{\mathbf{w}}_{\text{Ridge}} = \left(\mathbf{X}^{\top}\mathbf{X} + \lambda \mathbf{I}_p \right)^{-1} \mathbf{X}^{\top}\mathbf{y} \tag{19}$$

Probabilistic interpretation

- We state the problem as:

$$
\begin{aligned}
y^{(i)} &= \mathbf{x}^{(i)\top} \mathbf{w} + \epsilon^{(i)} && (20) \\
\epsilon^{(i)} &\sim \mathcal{N}(0, \sigma) && (21) \\
Y^{(i)} &\sim \mathcal{N}(\mathbf{x}^{(i)\top} \mathbf{w}, \sigma) && (22) \\
W_j &\sim \mathcal{N}(0, \gamma) && (23)
\end{aligned}
$$

- Maximizing the log-posterior probability of W:

$$
\begin{aligned}
\hat{\mathbf{w}}_{\mathrm{MAP}} &= \underset{\mathbf{w}}{\operatorname{argmax}} \log P(W) P(Y \mid W) && (24) \\
&= \underset{\mathbf{w}}{\operatorname{argmax}} \log \left(\left[\prod_j \frac{1}{\sqrt{2\pi\gamma^2}} \exp \frac{-W_j^2}{2\gamma^2} \right] \left[\prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp \frac{-(y^{(i)} - \mathbf{x}^{(i)\top} \mathbf{W})^2}{2\sigma^2} \right] \right) && (25)
\end{aligned}
$$

- Exercise: show that this results in the same problem as ridge regression
  - Ridge regression is equivalent to enforcing a zero mean gaussian prior on the individual weights.

What is the effect of λ? Which λ to choose?

$$\hat{\mathbf{w}}_{\text{Ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right)^\top \left(\mathbf{y} - \mathbf{X}\mathbf{w}\right) + \lambda \mathbf{w}^\top \mathbf{w} \tag{26}$$

$$= \left(\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_p\right)^{-1} \mathbf{X}^\top \mathbf{y} \tag{27}$$

- think of λ as a shrinkage parameter varying how much the weights are allowed to be close to the OLS solution.
  - when $\lambda \to 0$, $\hat{\mathbf{w}}_{\text{Ridge}} \to \hat{\mathbf{w}}_{\text{OLS}}$
  - when $\lambda \to \infty$, $\hat{\mathbf{w}}_{\text{Ridge}} \to 0_p$ (vector of 0s)

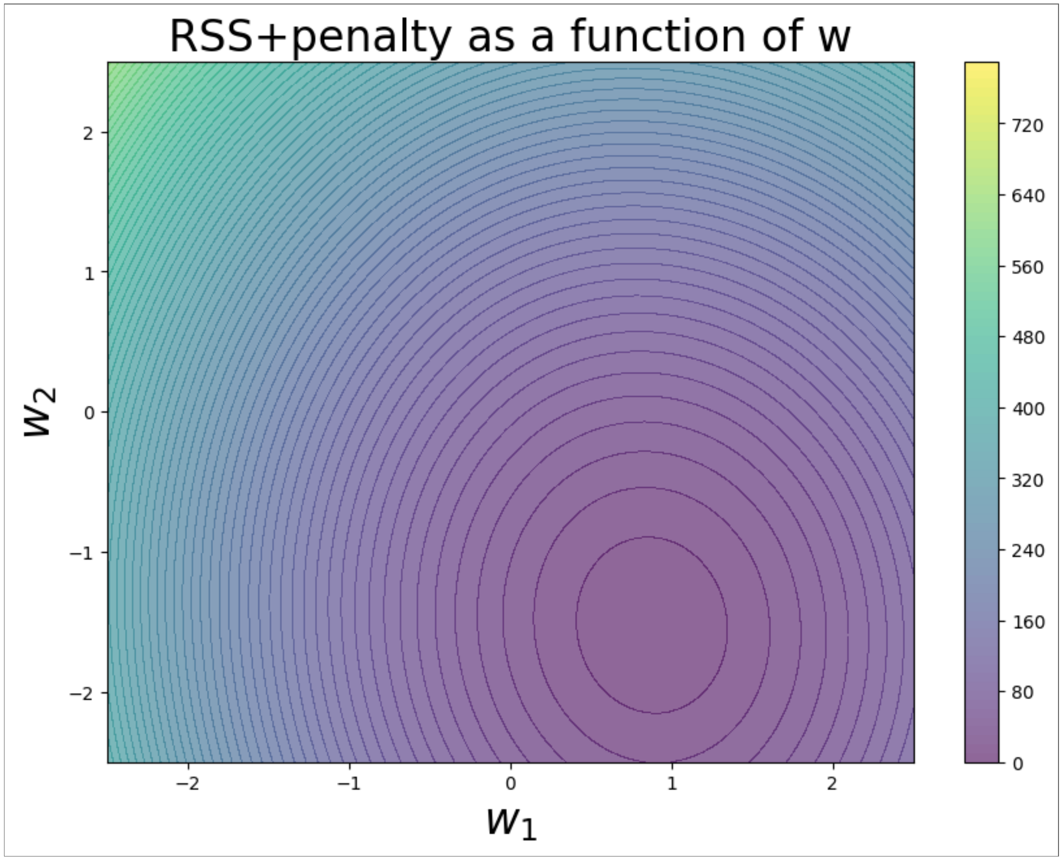
Let's look at a specific problem with two input features x_1 and x_2.

In [5]:

```
w1x = np.linspace(-2.5,2.5,100)
w2x = np.linspace(-2.5,2.5,100)
W1,W2 = np.meshgrid(w1x, w2x)

X = multivariate_normal.rvs(mean=np.array([0,0]),cov=1,size=20)
real_w = np.array([[0.8],[-1.5]])
Y = X.dot(real_w) + 0.4*norm.rvs(size=(20,1))

def rss(w1,w2):
    w = np.array([[w1],[w2]])
    loss = np.sum( (Y - X.dot(w)) **2)
    return loss
```

In [12]:

```
plt.figure(figsize=(10,7))

lmbda = 0.1
L_w = np.vectorize(rss)(*np.meshgrid(w1x, w2x)) +lmbda*(W1**2+W2**2)

cs = plt.contourf(W1, W2, L_w,levels=np.arange(0,800,10),alpha=0.6);
plt.colorbar()
plt.xlabel(r'$w_1$',fontsize=24)
plt.ylabel(r'$w_2$',fontsize=24)
plt.title('RSS+penalty as a function of w',fontsize=24);
```



Alternative formulation of optimization problem

$$\hat{\mathbf{w}}_{\text{Ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - {\mathbf{x}^{(i)}}^\top \mathbf{w} \right)^2 + \lambda \sum_j w_j^2 \tag{28}$$

- Can also be written as

$$\hat{\mathbf{w}}_{\text{Ridge}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_i \left(y^{(i)} - {\mathbf{x}^{(i)}}^\top \mathbf{w} \right)^2 \tag{29}$$

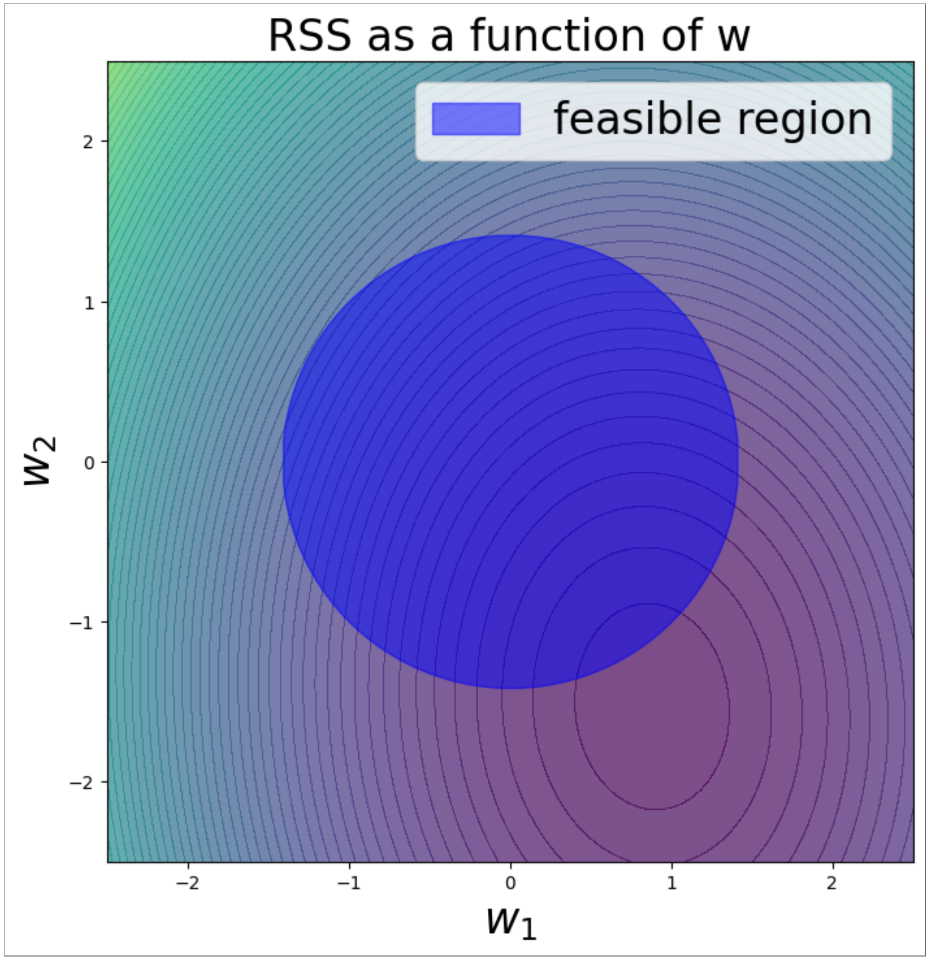
$$\text{subject to } \sum_j w_j^2 \leq t \tag{30}$$

- where, for each problem, there is a one-to-one correspondance between specific values of λ and t. We can use this formulation to better understand the effect of the constraint on the value of the parameters that is chosen

In [13]:

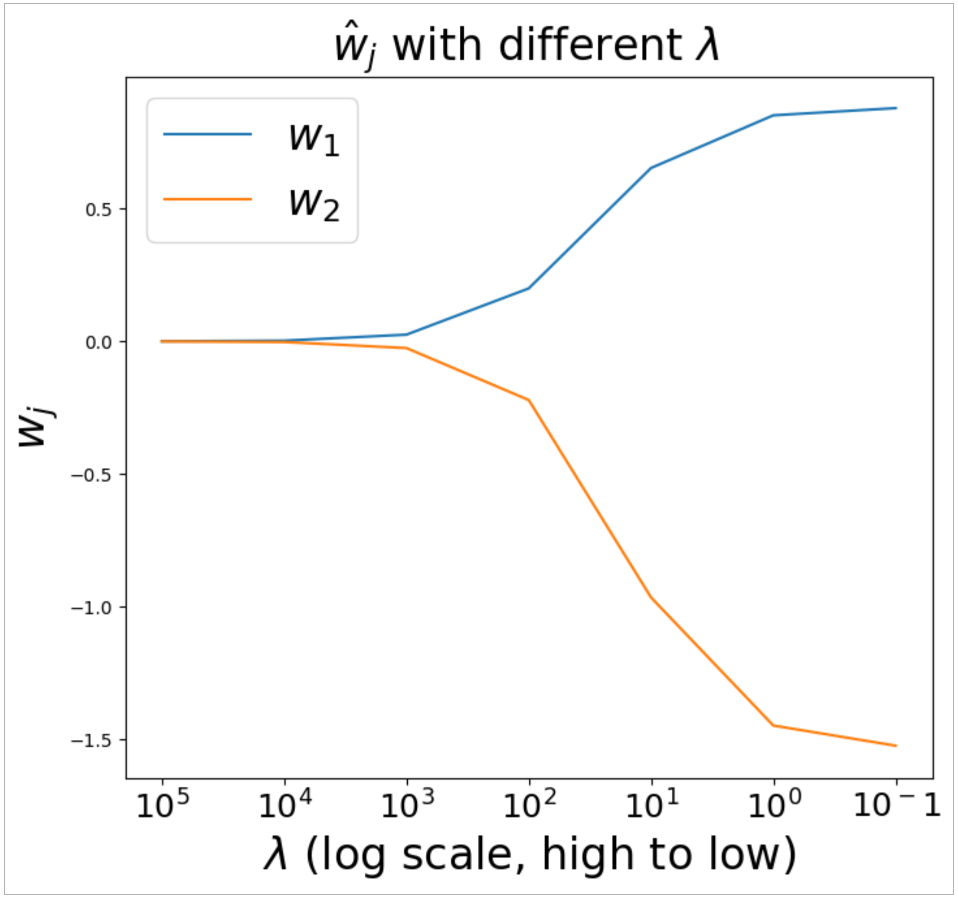
```
plt.figure(figsize=(8,8))

L_w = np.vectorize(rss)(*np.meshgrid(w1x, w2x))

cs = plt.contourf(W1, W2, L_w,levels=np.arange(0,800,10),alpha=0.7,aspect='equal');
plt.xlabel(r'$w_1$',fontsize=24)
plt.ylabel(r'$w_2$',fontsize=24)
plt.title('RSS as a function of w',fontsize=24);

t = 2
w1x_plot = np.linspace(-3,3,1000)
w1x_plot = w1x_plot[w1x_plot**2<=t]
w2_plot = np.nan_to_num(np.sqrt(t - w1x_plot**2))
plt.fill_between(w1x_plot, w2_plot, -w2_plot ,color='b',alpha=0.5,label='feasible region');
plt.legend(fontsize=24);
```

```
/var/folders/hx/t6_xnh5978s_6dsf2vb0kwv80000gn/T/ipykernel_67513/2374333085.py:5: UserWarning: The following
kwargs were not used by contour: 'aspect'
  cs = plt.contourf(W1, W2, L_w,levels=np.arange(0,800,10),alpha=0.7,aspect='equal');
```


RSS as a function of w
feasible region
w_1
w_2
−2
−1
0
1
2

In [14]:

```
plt.figure(figsize=(8,7))
from numpy.linalg import inv

def ridge(X,Y,lmbda):
    p = X.shape[1]
    return inv(X.T.dot(X) + lmbda*np.eye(p)).dot(X.T.dot(Y))

lmbdas = np.array([0.1,1,10,100,1000,10000,100000])
w_lambda = np.hstack([ridge(X,Y,L) for L in lmbdas])
plt.plot(np.arange(len(lmbdas)),w_lambda[0][::-1],label=r'$w_1$')
plt.plot(np.arange(len(lmbdas)),w_lambda[1][::-1],label=r'$w_2$')

xlabels = [r'$10^{}$'.format(int(np.log10(L))) for L in lmbdas]
plt.xticks(np.arange(len(lmbdas)), xlabels[::-1],fontsize=18 )

plt.xlabel(r'$\lambda$ (log scale, high to low)',fontsize=24)
plt.ylabel(r'$w_j$',fontsize=24)
plt.title('$\hat w_j$ with different $\lambda$ ',fontsize=24)
plt.legend(fontsize=24);
```



$\hat{w}_j$ with different λ
w_1
w_2
w_j
0.5
0.0
−0.5
−1.0
−1.5
10^5
10^4
10^3
10^2
10^1
10^0
10^{-1}
λ (log scale, high to low)

Bias-variance trade-off

- Given $P(X,Y)$, let $\mathbf{w}^* \in \mathbb{R}^p$ the parameters of the best linear approximation of Y given X.
    - We attempt to estimate $\mathbf{w}^*$ using a finite sample from $P(X,Y)$.
- How good is our estimate $\hat{\mathbf{w}}$?
    - **bias**: if we could repeat the experiment multiple times (and thus calculate $\hat{\mathbf{w}}$ multiple times):
        - would the average $\hat{\mathbf{w}}$ be close to $\mathbf{w}^*$?
    - **variance**: if we could repeat the experiment multiple times:
        - how much would the $\hat{\mathbf{w}}$s agree? would they be very different?

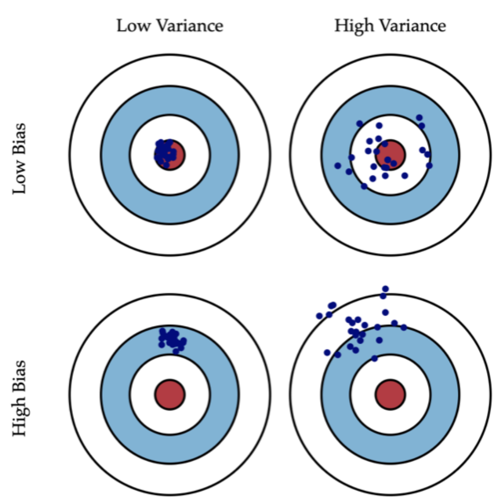


Fig. 1 Graphical illustration of bias and variance.

Source: **Understanding the Bias Variance Tradeoff by Scott Fortmann-Roe**

Effect of λ - tradeoff between bias and variance

- when $\lambda \to 0$: high variance, bias $\to 0$ (OLS solution is unbiased)
- when $\lambda \to \infty$: high bias, variance $\to 0$ (since converging to the zero vector)

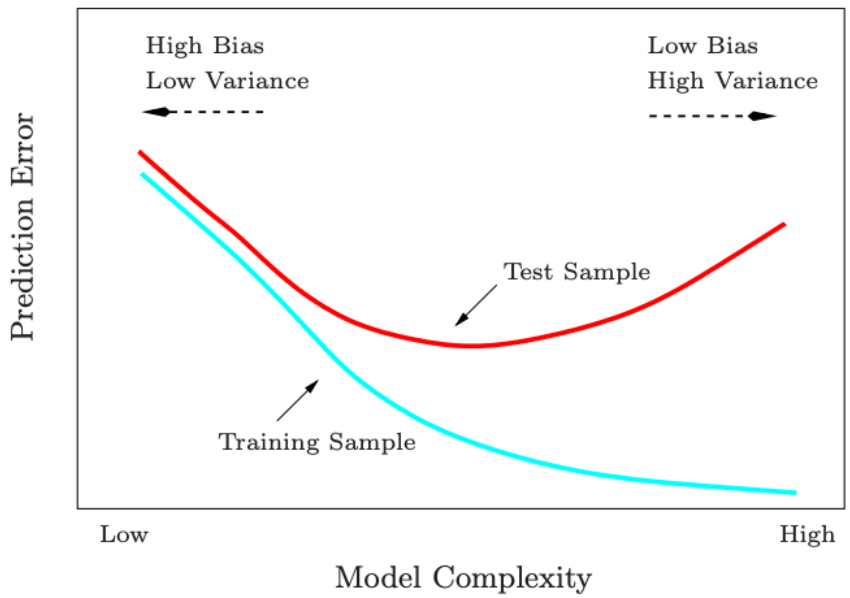


FIGURE 2.11. *Test and training error as a function of model complexity.*

Source: Figure 2.11 from **ESL**

Approach 3 - Lasso, adding L1 regularization

- The minimizes the Residual Sum of Squares (RSS) with an additional penalty on the ℓ_1 norm of $\mathbf{w}$:

$$\hat{\mathbf{w}}_{\text{Lasso}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y^{(i)} - \mathbf{x}^{(i)\top} \mathbf{w} \right)^2 + \lambda \sum_j |w_j| \tag{31}$$

- where $\lambda \geq 0$ is a penalty parameter
- Alternative formulation of optimization problem

$$\hat{\mathbf{w}}_{\text{Lasso}} = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i} \left(y^{(i)} - \mathbf{x}^{(i)\top} \mathbf{w} \right)^2 \tag{32}$$

$$\text{subject to } \sum_j |w_j| \leq t \tag{33}$$

- where, for each problem, there is a one-to-one correspondance between specific values of λ and t.
- The Lasso is also equivalent to imposing a Laplace prior on the parameters $w_j \sim \exp \frac{-|w_j|}{b}$.

Lasso optimization problem

- The Lasso optimization problem does not have a closed form solution, quadratic optimization problem.
    - More in 10-725
- The Lasso problem encourages sparsity! With high penalty (high λ or low t), few parameters will be non-zero
    - Think of it as taking a bet that only a few parameters are important

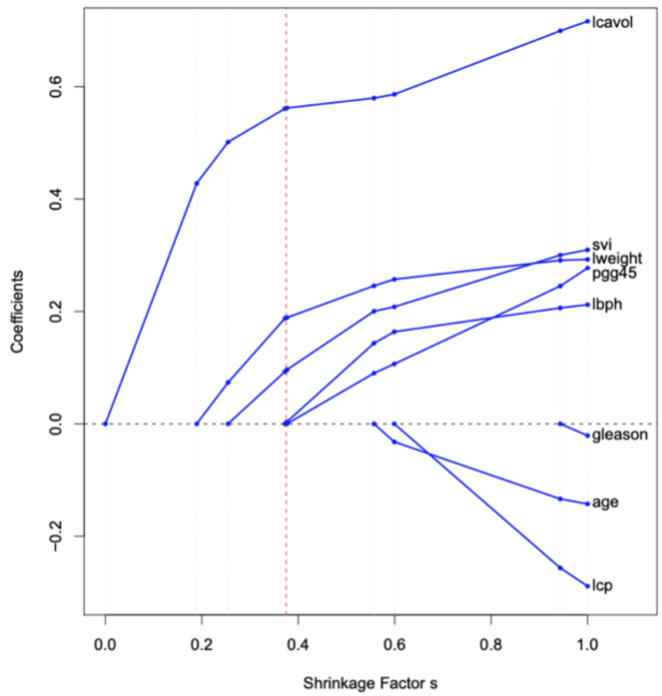


FIGURE 3.10. *Profiles of lasso coefficients, as the tuning parameter t is varied. Coefficients are plotted versus $s = t/\sum_1^p |\hat{\beta}_j|$. A vertical line is drawn at $s = 0.36$, the value chosen by cross-validation. Compare Figure 3.8 on page 65; the lasso profiles hit zero, while those for ridge do not. The profiles are piece-wise linear, and so are computed only at the points displayed; see Section 3.4.4 for details.*

Source: Figure 3.10 from **ESL**

Compare to Ridge solution

- High penalty causes weights to become smaller, but without being exactly 0.

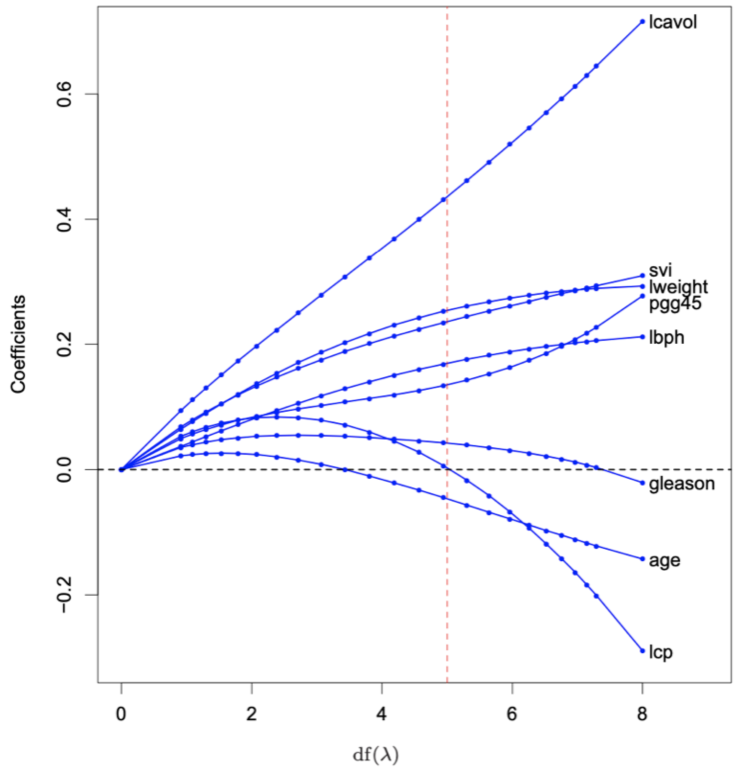


FIGURE 3.8. *Profiles of ridge coefficients for the prostate cancer example, as the tuning parameter λ is varied. Coefficients are plotted versus $\mathrm{df}(\lambda)$, the effective degrees of freedom. A vertical line is drawn at $\mathrm{df} = 5.0$, the value chosen by cross-validation.*

Source: Figure 3.8 from **ESL**

Lasso vs. Ridge solution

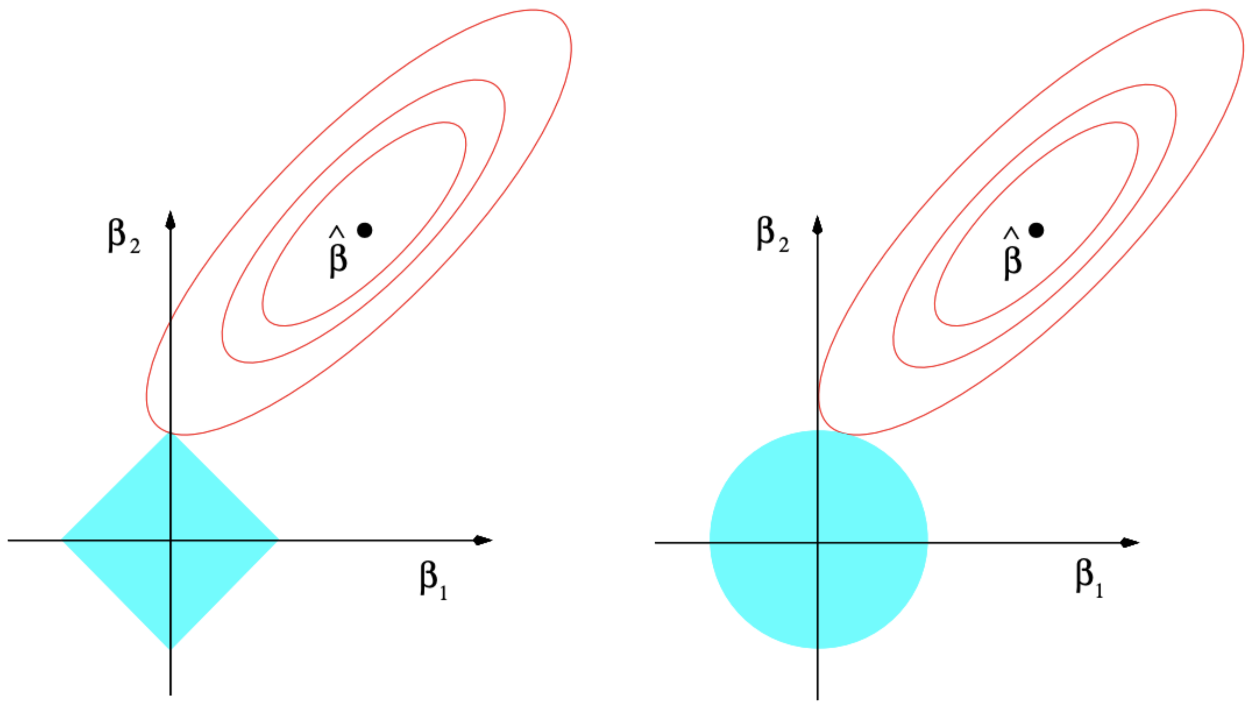


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \le t$ *and* $\beta_1^2 + \beta_2^2 \le t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

Source: Figure 3.11 from **ESL**

- In high dimensions, in Lasso, more likely to encounter edges or peaks.

How to pick λ?

- Divide training set into train and validation:
    - train with different λ settings
    - pick the λ with smallest **validation** error (not test error!)

- K-fold cross-validation:
    - Divide your training set into K folds, for each fold i:
        - train with different λ settings on the other K-1 folds
        - compute error on fold i for each λ
    - average error across fold and pick λ with smallest cross-validation error

- Other types of cross-validation (leave-one-out cross-validation etc...)

What you should know

- Linear regression definition
- OLS solution and interpretation
- Ridge solution and tradeoff
- Lasso (just the penalty and what it optimizes for)

There is a lot more to learn about regression!

- Class in statistics department (e.g. 36-707)
- questions to think about:
    - what happens when Y is multidimensional? How to adapt the solution?
    - see section 3.4.1 for an interpretation of the effect of Ridge on different dimensions in the X (there is more shrinkage applied to the directions of variance corresponding to the small eigenvalues).
    - how can we use the ridge regression solution to formulate kernel regression?