Machine Learning in R

Lecture 3 – Linear Algebra, Optimization and Validation

John Kornak – Epidemiology and Biostatistics

john.kornak@ucsf.edu

Projects

- How many people interested in performing their project on their own data?

In this lecture:

- Matrix algebra
- Validation and Cross-validation methods

Matrix algebra / Linear algebra

- ***Matrix*** (***Matrices pl.***) – rectangular array of numbers or variables
- ***Element*** *of a matrix* – a single value in a matrix
- ***Dimension*** *of a matrix* – number of rows × number of columns in a matrix (dimension: $m \times n$)

$$\begin{pmatrix} 6 & 3.1 & -4 & 12 \\ -9 & 4.7 & 3 & -4 \\ 6 & 3.2 & -4 & 12 \\ -5 & -4.1 & 2 & 1 \\ 1 & 9.0 & 9 & -5 \end{pmatrix}$$

Note that some prefer to use square rather than curved parentheses to represent a matrix – it's up to you.

Matrix algebra

- ***Matrix*** (***Matrices pl.***) – rectangular array of numbers or variables
- ***Element*** *of a matrix* – a single value in a matrix
- ***Dimension*** *of a matrix* – number of rows × number of columns in a matrix (dimension: $m \times n$)

$$
\begin{array}{c}
\text{4 columns} \\
\text{5 rows}\;
\begin{pmatrix}
6 & 3.1 & -4 & 12 \\
-9 & 4.7 & 3 & -4 \\
6 & 3.2 & -4 & 12 \\
-5 & -4.1 & 2 & 1 \\
1 & 9.0 & 9 & -5
\end{pmatrix}
\end{array}
$$

The above is a 5×4 matrix (5 rows and 4 columns, i.e. $m = 5$, $n = 4$).

Note that some prefer to use square rather than curved parentheses to represent a matrix – it's up to you.

Matrices in R

```
> elements <- c(6,-9,6,-5,1,3.1,4.7,3.2,-4.1,9.0,-4,3,-4
,2,9,12,-4,12,1,-5)
> X <- matrix(elements, nrow=5, ncol=4)
> is.matrix(X)
[1] TRUE
> X
     [,1] [,2] [,3] [,4]
[1,]    6  3.1   -4   12
[2,]   -9  4.7    3   -4
[3,]    6  3.2   -4   12
[4,]   -5 -4.1    2    1
[5,]    1  9.0    9   -5
> X[2,3]
[1] 3
> X[ ,1]
[1]  6 -9  6 -5  1
> X[1, ]
[1]  6.0  3.1 -4.0 12.0
>
> Z <- matrix(elements, nrow=5, ncol=4, byrow=TRUE)
> Z
     [,1] [,2] [,3] [,4]
[1,]  6.0 -9.0  6.0 -5.0
[2,]  1.0  3.1  4.7  3.2
[3,] -4.1  9.0 -4.0  3.0
[4,] -4.0  2.0  9.0 12.0
[5,] -4.0 12.0  1.0 -5.0
>
```

Element of X in 2^{nd} row and 3^{rd} column

1^{st} column of X

1^{st} row of X

$$\begin{array}{cc} & \text{4 columns} \\ \text{5 rows} & \begin{pmatrix} 6 & 3.1 & -4 & 12 \\ -9 & 4.7 & 3 & -4 \\ 6 & 3.2 & -4 & 12 \\ -5 & -4.1 & 2 & 1 \\ 1 & 9.0 & 9 & -5 \end{pmatrix} \end{array}$$

The above is a 5×4 matrix (5 rows and 4 columns, i.e. $m = 5$, $n = 4$).

Note that some prefer to use square rather than curved parentheses to represent a matrix – it's up to you.

Matrices

- Matrices are typically indexed by upper case letters, e.g. A, B, X, Y
- We can perform arithmetic on Matrices
- Matrices of important dimensions:

$$\begin{pmatrix} -4 \\ 2 \\ 7.3 \end{pmatrix}$$

Single column matrix or column vector, e.g. dimension 3×1

$$\begin{pmatrix} 6 & 2 & -2 & -1 \end{pmatrix}$$

Single row matrix or row vector, e.g. dimension 1×4

$$\begin{pmatrix} 4 & 2 \\ 4 & 3 \end{pmatrix}$$

Square matrix, e.g., dimension 2×2

Adding and subtracting matrices

To ***add or subtract matrices they need to be of the same dimension***.
Then you just add or subtract element-by-element:

$$\begin{pmatrix} 4 & 2 \\ 4 & 3 \end{pmatrix} + \begin{pmatrix} 7 & 3 \\ -2 & -6 \end{pmatrix} = \begin{pmatrix} 11 & 5 \\ 2 & -3 \end{pmatrix}$$

or,

$$\begin{pmatrix} 8.3 & 4.4 & 9.2 \\ 7.6 & 4.8 & 9.1 \end{pmatrix} - \begin{pmatrix} 6.2 & 7.1 & 9.4 \\ 6.8 & 6.7 & 9.1 \end{pmatrix} = \begin{pmatrix} 2.1 & -2.7 & -0.2 \\ 0.8 & -1.9 & 0.0 \end{pmatrix}$$

But not

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} + \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

Dimensions do not match
2×3 vs. 3×2

Adding and subtracting matrices

To add or subtract matrices they need to be of the same dimension. Then you just add or subtract element-by-element:

$$\begin{pmatrix} 4 & 2 \\ 4 & 3 \end{pmatrix} + \begin{pmatrix} 7 & 3 \\ -2 & -6 \end{pmatrix} = \begin{pmatrix} 11 & 5 \\ 2 & -3 \end{pmatrix}$$

or,

$$\begin{pmatrix} 8.3 & 4.4 & 9.2 \\ 7.6 & 4.8 & 9.1 \end{pmatrix} - \begin{pmatrix} 6.2 & 7.1 & 9.4 \\ 6.8 & 6.7 & 9.1 \end{pmatrix} = \begin{pmatrix} 2.1 & -2.7 & -0.2 \\ 0.8 & -1.9 & 0.0 \end{pmatrix}$$

But not

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} + \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$$

Dimensions do not match
2×3 vs. 3×2

```
> A <- matrix(c(4,4,2,3),nrow=2,ncol=2)
> A
     [,1] [,2]
[1,]    4    2
[2,]    4    3
> B <- matrix(c(7,-2,3,-6),2,2)
> B
     [,1] [,2]
[1,]    7    3
[2,]   -2   -6
> A+B
     [,1] [,2]
[1,]   11    5
[2,]    2   -3
> A-B
     [,1] [,2]
[1,]   -3   -1
[2,]    6    9
>
> C <- matrix(c(1:6),3,2)
> C
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> A+C
Error in A + C : non-conformable arrays
>
```

Matrix multiplication by a scalar

A *scalar* is a single value

When a matrix is multiplied by a scalar each element in the matrix is multiplied by the scalar e.g.

$$3\begin{pmatrix}3 & 4\\ 3 & 3\\ 4 & 3\end{pmatrix} = \begin{pmatrix}9 & 12\\ 9 & 9\\ 12 & 9\end{pmatrix}$$

or,

$$-2\begin{pmatrix}1 & -0.5\\ 1 & 0\end{pmatrix} = \begin{pmatrix}-2 & 1\\ -2 & 0\end{pmatrix}$$

```
> 3*matrix(c(3,3,4,4,3,3),3,2)
     [,1] [,2]
[1,]    9   12
[2,]    9    9
[3,]   12    9
> -2*matrix(c(1,1,-0.5,0),2,2)
     [,1] [,2]
[1,]   -2    1
[2,]   -2    0
> 
```

Matrix multiplication by a scalar

A *scalar* is a single value

When a matrix is multiplied by a scalar each element in the matrix is multiplied by the scalar e.g.

$$3\begin{pmatrix}3 & 4\\ 3 & 3\\ 4 & 3\end{pmatrix} = \begin{pmatrix}9 & 12\\ 9 & 9\\ 12 & 9\end{pmatrix}$$

or,

$$-2\begin{pmatrix}1 & -0.5\\ 1 & 0\end{pmatrix} = \begin{pmatrix}-2 & 1\\ -2 & 0\end{pmatrix}$$

```
> 3*matrix(c(3,3,4,4,3,3),3,2)
     [,1] [,2]
[1,]    9   12
[2,]    9    9
[3,]   12    9
> -2*matrix(c(1,1,-0.5,0),2,2)
     [,1] [,2]
[1,]   -2    1
[2,]   -2    0
> 
```

What about multiplication of 2 matrices?

Multiply a column vector by a matrix

Matrix multiplication diverges from what you might expect. We will start with a matrix multiplying a vector example:

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = ?$$

Multiply a column vector by a matrix

Matrix multiplication diverges from what you might expect. We will start with a matrix multiplying a vector example:

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} ? \\ ? \end{pmatrix}$$

Multiply a column vector by a matrix

Matrix multiplication diverges from what you might expect. We will start with a matrix multiplying a vector example:

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} ? \\ ? \end{pmatrix}$$

Multiply a column vector by a matrix

Matrix multiplication diverges from what you might expect. We will start with a matrix multiplying a vector example:

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} ? \\ ? \end{pmatrix}$$

Multiply a column vector by a matrix

Matrix multiplication diverges from what you might expect. We will start with a matrix multiplying a vector example:

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 7 \\ ? \end{pmatrix}$$

Multiply element by element and add together, i.e.

$$3\times 2 + 1\times 1 = 7$$

Multiply a column vector by a matrix

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \overset{\times \text{ and } +}{\curvearrowright} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 7 \\ ? \end{pmatrix}$$

Multiply element by element and add together, i.e.

$$3\times2 + 1\times1 = 7$$

Multiply a column vector by a matrix

× and +

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$$

Multiply element by element and add together, i.e.

$$2\times2 + 4\times1 = 8$$

Multiply a column vector by a matrix

× and +

$$\begin{pmatrix}3 & 1\\ 2 & 4\end{pmatrix}\begin{pmatrix}2\\ 1\end{pmatrix} = \begin{pmatrix}7\\ 8\end{pmatrix}$$

```
> V <- matrix(c(3,2,1,4),2,2)
> V
     [,1] [,2]
[1,]    3    1
[2,]    2    4
> W <- matrix(c(2,1),2,1)
> W
     [,1]
[1,]    2
[2,]    1
> V*W
Error in V * W : non-conformable arrays
```

Multiply a column vector by a matrix

$$\begin{pmatrix} 3 & 1 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$$

× and +

```
> V <- matrix(c(3,2,1,4),2,2)
> V
     [,1] [,2]
[1,]    3    1
[2,]    2    4
> W <- matrix(c(2,1),2,1)
> W
     [,1]
[1,]    2
[2,]    1
> V*W
Error in V * W : non-conformable arrays
> V %*% W
     [,1]
[1,]    7
[2,]    8
>
```

Multiply a column vector by a matrix – 2nd example

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} ? \\ ? \\ ? \end{pmatrix}$$

Multiply a column vector by a matrix – 2nd example

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ ? \\ ? \end{pmatrix}$$

$$-1 \times 2 + 2 \times (-1) + 0 \times 0.5 + 4 \times 3 = 12$$

Multiply a column vector by a matrix – 2nd example

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix}\begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ 18 \\ ? \end{pmatrix}$$

$$-1\times 2 + 2\times(-1) + 0\times 0.5 + 4\times 3 = 12$$

$$1\times 2 + (-4)\times(-1) + 3\times 0.5 + 3.5\times 3 = 18$$

Multiply a column vector by a matrix – 2nd example

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ 18 \\ 8 \end{pmatrix}$$

$$-1 \times 2 + 2 \times (-1) + 0 \times 0.5 + 4 \times 3 = 12$$

$$1 \times 2 + (-4) \times (-1) + 3 \times 0.5 + 3.5 \times 3 = 18$$

$$3 \times 2 + 2 \times (-1) + 2 \times 0.5 + 1 \times 3 = 8$$

Multiply a column vector by a matrix

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ 18 \\ 8 \end{pmatrix}$$

$$-1\times2 + 2\times(-1) + 0\times0.5 + 4\times3 = 12$$

$$1\times2 + (-4)\times(-1) + 3\times0.5 + 3.5\times3 = 18$$

$$3\times2 + 2\times(-1) + 2\times0.5 + 1\times3 = 8$$

```
> P <- matrix(c(-1,1,3,2,-4,2,0,3,2,4,3.5,1),3,4)
> Q <- matrix(c(2,-1,0.5,3),4,1)
> P
     [,1] [,2] [,3] [,4]
[1,]   -1    2    0  4.0
[2,]    1   -4    3  3.5
[3,]    3    2    2  1.0
> Q
     [,1]
[1,]  2.0
[2,] -1.0
[3,]  0.5
[4,]  3.0
> P %*% Q
     [,1]
[1,]    8
[2,]   18
[3,]    8
> 
```

Multiply a column vector by a matrix

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ 18 \\ 8 \end{pmatrix}$$

$$-1\times 2 + 2\times(-1) + 0\times 0.5 + 4\times 3 = 12$$

$$1\times 2 + (-4)\times(-1) + 3\times 0.5 + 3.5\times 3 = 18$$

$$3\times 2 + 2\times(-1) + 2\times 0.5 + 1\times 3 = 8$$

So why is all this useful?

```
> P <- matrix(c(-1,1,3,2,-4,2,0,3,2,4,3.5,1),3,4)
> Q <- matrix(c(2,-1,0.5,3),4,1)
> P
     [,1] [,2] [,3] [,4]
[1,]   -1    2    0  4.0
[2,]    1   -4    3  3.5
[3,]    3    2    2  1.0
> Q
     [,1]
[1,]  2.0
[2,] -1.0
[3,]  0.5
[4,]  3.0
> P %*% Q
     [,1]
[1,]    8
[2,]   18
[3,]    8
> 
```

Multiply a column vector by a matrix

$$\begin{pmatrix} -1 & 2 & 0 & 4 \\ 1 & -4 & 3 & 3.5 \\ 3 & 2 & 2 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ -1 \\ 0.5 \\ 3 \end{pmatrix} = \begin{pmatrix} 8 \\ 18 \\ 8 \end{pmatrix}$$

$$-1\times 2 + 2\times(-1) + 0\times 0.5 + 4\times 3 = 12$$

$$1\times 2 + (-4)\times(-1) + 3\times 0.5 + 3.5\times 3 = 18$$

$$3\times 2 + 2\times(-1) + 2\times 0.5 + 1\times 3 = 8$$

```
> P <- matrix(c(-1,1,3,2,-4,2,0,3,2,4,3.5,1),3,4)
> Q <- matrix(c(2,-1,0.5,3),4,1)
> P
     [,1] [,2] [,3] [,4]
[1,]   -1    2    0  4.0
[2,]    1   -4    3  3.5
[3,]    3    2    2  1.0
> Q
     [,1]
[1,]  2.0
[2,] -1.0
[3,]  0.5
[4,]  3.0
> P %*% Q
     [,1]
[1,]    8
[2,]   18
[3,]    8
>
```

So why is all this useful?

Because it allows simplification for computation and notation

Consider a data matrix

$$\begin{array}{cc} & \overbrace{\hphantom{x_{10} \quad x_{11} \quad x_{12} \quad \cdots \quad x_{1(p-1)}}}^{p\ \textbf{columns}} \\ n\ \textbf{rows} \left\{ \vphantom{\begin{matrix} 1\\1\\1\\1\\1\\1 \end{matrix}} \right. & \begin{pmatrix} x_{10} & x_{11} & x_{12} & \cdots & x_{1(p-1)} \\ x_{20} & x_{21} & x_{22} & \cdots & x_{2(p-1)} \\ x_{30} & x_{31} & x_{32} & \cdots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n0} & x_{n1} & x_{n2} & \cdots & x_{n(p-1)} \end{pmatrix} \end{array}$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)

a response column vector matrix

$$
n\ \textbf{rows}\left\{\overbrace{\begin{pmatrix}
x_{10} & x_{11} & x_{12} & \cdots & x_{1(p-1)} \\
x_{20} & x_{21} & x_{22} & \cdots & x_{2(p-1)} \\
x_{30} & x_{31} & x_{32} & \cdots & x_{3(p-1)} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
x_{n0} & x_{n1} & x_{n2} & \cdots & x_{n(p-1)}
\end{pmatrix}}^{p\ \textbf{columns}}\right.
\qquad
n\ \textbf{obs.}\left\{\begin{pmatrix}
y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n
\end{pmatrix}\right.
$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)
y_i is the observed outcome for the i^{th} observation (subject)

a vector of parameters

$$n \text{ rows} \left\{ \overbrace{\begin{pmatrix} x_{10} & x_{11} & x_{12} & \cdots & x_{1(p-1)} \\ x_{20} & x_{21} & x_{22} & \cdots & x_{2(p-1)} \\ x_{30} & x_{31} & x_{32} & \cdots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n0} & x_{n1} & x_{n2} & \cdots & x_{n(p-1)} \end{pmatrix}}^{p \text{ columns}} \right. \qquad n \text{ observations} \left\{ \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} \right. \qquad p \text{ parameters} \left\{ \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} \right.$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)
y_i is the observed outcome for the i^{th} observation (subject)
β_j is the j^{th} parameter (regression coefficient)

a vector of parameters

$$\begin{array}{cc} & p \text{ columns} \\ n \text{ rows} & \begin{pmatrix} x_{10} & x_{11} & x_{12} & \cdots & x_{1(p-1)} \\ x_{20} & x_{21} & x_{22} & \cdots & x_{2(p-1)} \\ x_{30} & x_{31} & x_{32} & \cdots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n0} & x_{n1} & x_{n2} & \cdots & x_{n(p-1)} \end{pmatrix} \end{array} \quad \begin{array}{cc} n \text{ observations} & \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} \end{array} \quad \begin{array}{cc} p \text{ parameters} & \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} \end{array} \quad \begin{array}{cc} n \text{ error variables} & \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix} \end{array}$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)
y_i is the observed outcome for the i^{th} observation (subject)
β_j is the j^{th} parameter (regression coefficient)
ε_i is the error term for the i^{th} observation (subject)

Now set $x_{i0} = 1$ for all i

$$\underbrace{\begin{pmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \cdots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \cdots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{n(p-1)} \end{pmatrix}}_{n \text{ rows},\ p \text{ columns}} \quad \underbrace{\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix}}_{n \text{ observations}} \quad \underbrace{\begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix}}_{p \text{ parameters}} \quad \underbrace{\begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}}_{n \text{ error variables}}$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)
y_i is the observed outcome for the i^{th} observation (subject)
β_j is the j^{th} parameter (regression coefficient)
ε_i is the error term for the i^{th} observation (subject)

And write

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \dots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \dots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{n(p-1)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

where x_{ij} is the value of the j^{th} variable for the i^{th} observation (subject)
y_i is the observed outcome for the i^{th} observation (subject)
β_j is the j^{th} parameter (regression coefficient)
ε_i is the error term for the i^{th} observation (subject)

And write

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \dots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \dots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{n(p-1)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Expanding the first row we get…

$y_1 = \beta_0 + x_{11}\beta_1 + x_{12}\beta_2 + \dots + x_{1n}\,\beta_{p-1} + \varepsilon_1$

And write

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \dots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \dots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{n(p-1)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Expanding the 1st row we get…

$y_1 = \beta_0 + x_{11}\beta_1 + x_{12}\beta_2 + \dots + x_{1(p-1)}\,\beta_{p-1} + \varepsilon_1$

And then for the 2nd row

$y_2 = \beta_0 + x_{21}\beta_1 + x_{22}\beta_2 + \dots + x_{2(p-1)}\,\beta_{p-1} + \varepsilon_2$

And write

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \dots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \dots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{n(p-1)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Expanding the 1st row we get…

$y_1 = \beta_0 + x_{11}\beta_1 + x_{12}\beta_2 + \dots + x_{1(p-1)}\,\beta_{p-1} + \varepsilon_1$

And then for the 2nd row

$y_2 = \beta_0 + x_{21}\beta_1 + x_{22}\beta_2 + \dots + x_{2(p-1)}\,\beta_{p-1} + \varepsilon_2$

down to

$y_n = \beta_0 + x_{n1}\beta_1 + x_{n2}\beta_2 + \dots + x_{n(p-1)}\,\beta_{p-1} + \varepsilon_n$

And write..

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1(p-1)} \\ 1 & x_{21} & x_{22} & \dots & x_{2(p-1)} \\ 1 & x_{31} & x_{32} & \dots & x_{3(p-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{n(p-1)} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{p-1} \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \vdots \\ \varepsilon_n \end{pmatrix}$$

Expanding the 1st row we get…

$y_1 = \beta_0 + x_{11}\beta_1 + x_{12}\beta_2 + \dots + x_{1(p-1)}\,\beta_{p-1} + \varepsilon_1$

And then for the 2nd row

$y_2 = \beta_0 + x_{21}\beta_1 + x_{22}\beta_2 + \dots + x_{2(p-1)}\,\beta_{p-1} + \varepsilon_2$

down to

$y_n = \beta_0 + x_{n1}\beta_1 + x_{n2}\beta_2 + \dots + x_{n(p-1)}\,\beta_{p-1} + \varepsilon_n$

Succinctly, we can write

$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

The linear model

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} ? & ? \\ ? & ? \end{pmatrix}$$

2×3 $\quad$ 3×2 $\quad$ 2×2

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} ? & ? \\ ? & ? \end{pmatrix}$$

2×3 $\quad$ 3×2 $\quad$ 2×2

Number of columns of first matrix must match number of rows of second matrix

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} ? & ? \\ ? & ? \end{pmatrix}$$

$2 \times 3 \qquad 3 \times 2 \qquad 2 \times 2$

Getting the dimensions
of the matrix answer

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 8 & ? \\ ? & ? \end{pmatrix}$$

Element $(1,1) = 1 \times 1 + 2 \times 2 + 3 \times 1 = 8$

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 8 & ? \\ 20 & ? \end{pmatrix}$$

Element $(1,1) = 1\times 1 + 2\times 2 + 3\times 1 = 8$

Element $(2,1) = 4\times 1 + 5\times 2 + 6\times 1 = 20$

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 8 & 10 \\ 20 & ? \end{pmatrix}$$

Element $(1,1) = 1\times1 + 2\times2 + 3\times1 = 8$
Element $(2,1) = 4\times1 + 5\times2 + 6\times1 = 20$
Element $(1,2) = 1\times2 + 2\times1 + 3\times2 = 10$

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 8 & 10 \\ 20 & 25 \end{pmatrix}$$

Element $(1,1) = 1\times1 + 2\times2 + 3\times1 = 8$
Element $(2,1) = 4\times1 + 5\times2 + 6\times1 = 20$
Element $(1,2) = 1\times2 + 2\times1 + 3\times2 = 10$
Element $(2,2) = 4\times2 + 5\times1 + 6\times2 = 25$

Matrix multiplication

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}\begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 8 & 10 \\ 20 & 25 \end{pmatrix}$$

Element $(1,1) = 1\times1 + 2\times2 + 3\times1 = 8$
Element $(2,1) = 4\times1 + 5\times2 + 6\times1 = 20$
Element $(1,2) = 1\times2 + 2\times1 + 3\times2 = 10$
Element $(2,2) = 4\times2 + 5\times1 + 6\times2 = 25$

```
> R <- matrix(1:6,2,3,byrow=TRUE)
> S <- matrix(rep(1:2,3),3,2)
> R%*%S
     [,1] [,2]
[1,]    8   10
[2,]   20   25
>
```

Element-by-element multiplication

$$\begin{pmatrix} 2 & 6 \\ 3 & 5 \\ 4 & 4 \end{pmatrix} * \begin{pmatrix} 2 & 6 \\ 4 & 7 \\ 2 & 3 \end{pmatrix} = \begin{pmatrix} 4 & 36 \\ 12 & 35 \\ 8 & 12 \end{pmatrix}$$

```
> U1 <- matrix(c(2,3,4,6,5,4),3,2)
> U2 <- matrix(c(2,4,2,6,7,3),3,2)
> U1*U2
     [,1] [,2]
[1,]    4   36
[2,]   12   35
[3,]    8   12
> 
```

Matrix transpose

To get a matrix transpose you reflect the matrix elements along the main diagonal:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

Matrix transpose

To get a matrix transpose you reflect the matrix elements along the main diagonal:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

Matrix transpose

To get a matrix transpose you reflect the matrix elements along the main diagonal:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

```
> X1 <- matrix(1:9,3,3)
> X1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
> t(X1)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Matrix transpose

To get a matrix transpose you reflect the matrix elements along the main diagonal:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

You can also transpose non-square matrices

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

Matrix transpose

To get a matrix transpose you reflect the matrix elements along the main diagonal:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix}$$

And note that if $A^{\mathrm{T}} = B$, then $B^{\mathrm{T}} = A$

$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

```
> X2 <- matrix(1:6,2,3)
> X2
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> X2T <- t(X2)
> X2T
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
> X2TT <- t(X2T)
> X2TT
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
>
```

Identity matrix

The identity matrix is a square matrix with ones on the diagonal and zeros everywhere else. Whenever, you multiply with the identity matrix you get the same answer (like multiplying scalars by one)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix} = \begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix}$$

Identity matrix

The identity matrix is a square matrix with ones on the diagonal and zeros everywhere else. Whenever, you multiply with the identity matrix you get the same answer (like multiplying scalars by one)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix} = \begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix}$$

Identity matrix denoted by I, e.g. I_2, and I_3

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

Works whether you pre- or post-multiply

Identity matrix

The identity matrix is a square matrix with ones on the diagonal and zeros everywhere else. Whenever, you multiply with the identity matrix you get the same answer (like multiplying scalars by one)

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix} = \begin{pmatrix} 6 & 7 \\ 12 & 2 \end{pmatrix}$$

Identity matrix denoted by I, e.g. I_2, and I_3

I_2

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$$

I_3

Works whether you pre- or post-multiply

Inverse matrix

For a matrix A it's inverse A^{-1} is such that

$$AA^{-1} = I = A^{-1}A$$

E.g. $\begin{pmatrix} 4 & 7 \\ 2 & 6 \end{pmatrix}^{-1} = \begin{pmatrix} 0.6 & -0.7 \\ -0.2 & 0.4 \end{pmatrix}$

Not all matrices have an inverse (most don't) – they have to be square and *nonsingular*

Calculating an inverse is quite involved but `R` can do it for you with the `solve` command

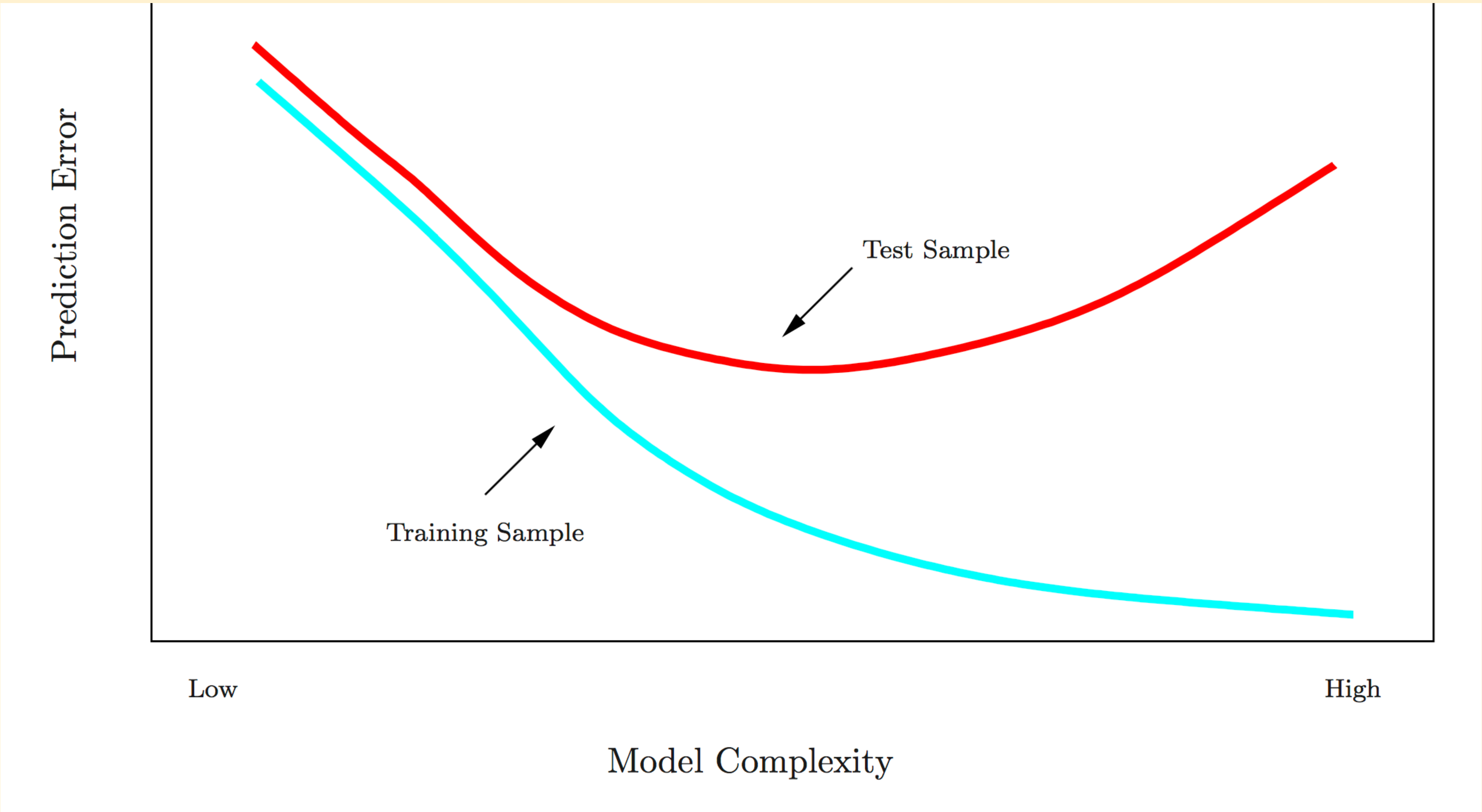
```
> Y1 <- matrix(c(4,2,7,6),2,2)
> Y1
     [,1] [,2]
[1,]    4    7
[2,]    2    6
> Y2 <- solve(Y1)
> Y2
     [,1] [,2]
[1,]  0.6 -0.7
[2,] -0.2  0.4
> Y1%*%Y2
     [,1] [,2]
[1,]    1    0
[2,]    0    1
>
```

Validation and Cross-validation

Training Error versus Test error

- Recall the distinction between the *test error* and the *training error*:
- The *test error* is the average error that results from using a machine learning method to predict the response on a new observation, i.e., one that was not used in training the method.
- In contrast, the *training error* is the average error of the machine learning method to predict the observations that were used in training.
- But the training error rate can be quite different from the test error rate, and in particular the training error can *dramatically underestimate* the the test error of interest.

Training- versus Test-Set Performance



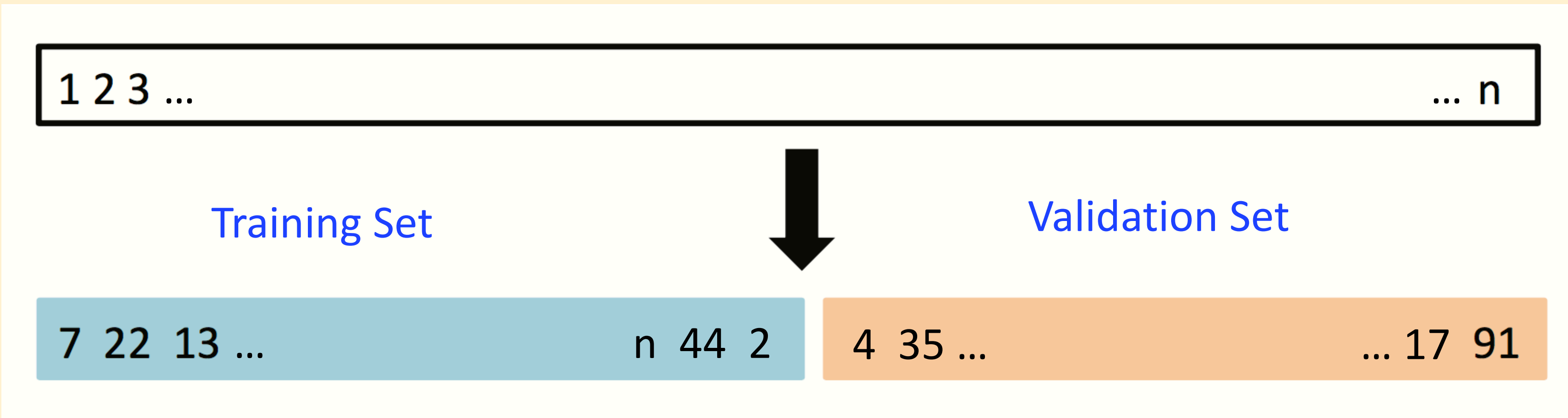
More on prediction-error estimates

- Best solution: a large designated independent test set. Often not available.
- Some methods make a mathematical adjustment to the training error rate in order to estimate the test error rate. These include *AIC* and *BIC*.
- Here we instead consider a class of methods that estimate the test error by removing a subset of the training observations from the fitting process, and then applying the fitted machine learning method to those held out observations (the *validation set* [sometimes called *hold-out set*], which is now effectively a test set)

Validation approach

- Here we randomly divide the available set of samples into two parts: a *training set* and a *validation set*.
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set.
- The resulting validation set error provides an estimate of the test error. This is typically assessed using MSE in the case of a quantitative response and misclassification rate in the case of a qualitative (discrete) response.

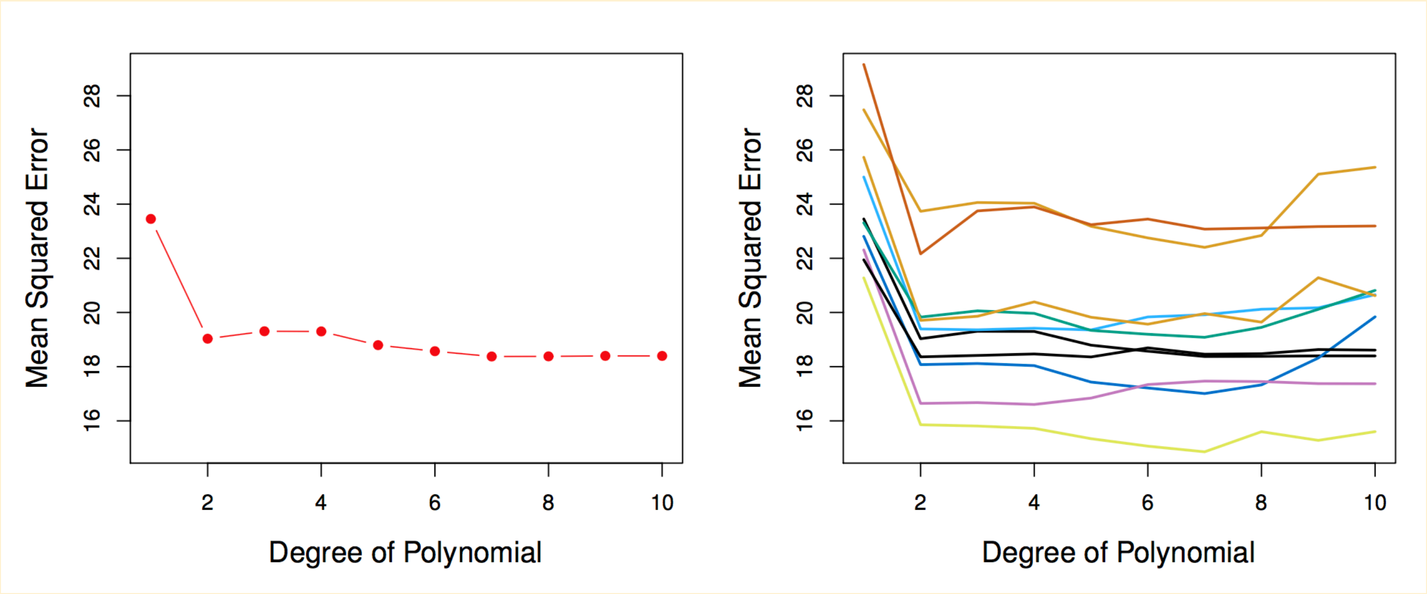
The Validation process



A random split is made into two halves: left part is training set, right part is validation set (it is also common to split 60:40)

Example

- Want to compare linear vs higher-order polynomial terms in a linear regression
- We randomly split the 392 observations into two sets, a training set containing 196 of the data points, and a validation set containing the remaining 196 observations.



Left panel shows single split; right panel shows multiple splits

Drawbacks of validation set approach

- The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the test set.
- In the validation approach, only a subset of the observations — those that are included in the training set rather than in the validation set — are used to fit the model.
- This suggests that the validation set error may tend to *overestimate* the test error for the model fit on the entire data set. (The model has been fit with a sample size of $n/2$ rather than n, which will clearly provide a less reliable fit.)

Alternative: K-fold cross-validation

- Widely used approach for estimating test error.
- Estimates can be used to select best model, and to give an idea of the test error of the final chosen model.
- Idea is to randomly divide the data into K equal-sized parts. We leave out part k, fit the model to the other $K-1$ parts (combined), and then obtain predictions for the left-out k^{th} part.
- This is done in turn for each part $k = 1,2, \dots K$, and then the results are combined.

K-fold Cross-validation in detail

Divide data into K roughly equal-sized parts ($K = 5$ here)

K-fold Cross-validation in detail

Divide data into K roughly equal-sized parts ($K = 5$ here)

Now fit the model on the four training parts combined
Calculate the prediction error on the validation set.

K-fold Cross-validation in detail

Divide data into K roughly equal-sized parts ($K = 5$ here)

Validation	Train	Train	Train	Train

Now fit the model on the four training parts combined
Calculate the prediction error on the validation set.
Now switch the validation set and refit to the 4 remaining train sets and calculate prediction error on new validation set.

Train	Validation	Train	Train	Train

K-fold Cross-validation in detail

Divide data into K roughly equal-sized parts ($K = 5$ here)

Validation	Train	Train	Train	Train

Now fit the model on the four training parts combined
Calculate the prediction error on the validation set.
Now switch the validation set and refit to the 4 remaining train sets and calculate prediction error on new validation set.

Train	Validation	Train	Train	Train

Repeat this process until every partition has been the validation set and average the prediction error over all 5 validation sets.

The details

- Let the K parts be $C_1, C_2, \dots C_K$, where C_k denotes the indices of the observations in part k. There are n_k observations in part k: if N is a multiple of K, then $n_k = n/K$.
- Compute
$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{MSE}_k$$
MSE for continuous outcome

  where $\mathrm{MSE}_k = \sum_{i \in C_k} (y_i - \hat{y}_i)^2 / n_k$ and $\hat{y}_i$ is the fit for observation i, obtained from the data with part k removed.
- Setting $K = n$ yields n-fold or leave-one-out cross-validation (LOOCV).

A nice special case!

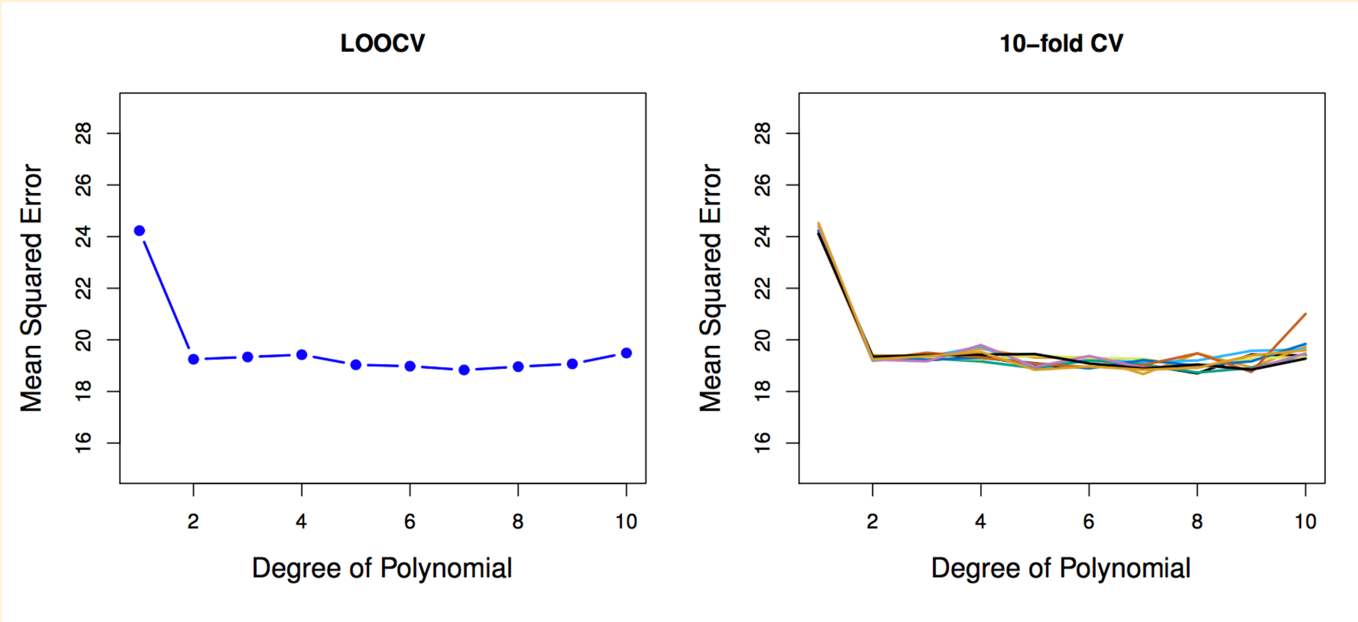
With least-squares linear or polynomial regression, an amazing shortcut makes the cost of LOOCV the same as that of a single model fit! The following formula holds:

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

where $\hat{y}_i$ is the i^{th} fitted value from the original least squares fit, and h_i is the leverage (diagonal of the "hat" matrix; see ISLR book for details.) This is like the ordinary MSE, except the i^{th} residual is divided by $1 - h_i$.

Limitations of LOOCV

- LOOCV sometimes useful, but typically doesn't *shake up* the data enough. The estimates from each fold are highly correlated and hence their average can have high variance.
- A better choice is $K = 5$ or 10.



Other issues with Cross-validation

- Since each training set is only $(K-1)/K$ as big as the original training set, the estimates of prediction error will typically be biased upward. *Why?*
- This bias is minimized when $K = n$ (LOOCV), but this estimate has high variance.
- $K = 5$ or 10 provides a good compromise for this bias-variance tradeoff.

Cross-Validation for Classification Problems

- We divide the data into K roughly equal-sized parts $C_1, C_2, \ldots C_K$. C_k denotes the indices of the observations in part k. There are n_k observations in part k: if n is a multiple of K, then $n_k = n/K$.
- Compute
$$\mathrm{CV}_K = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{Err}_k$$
where $\mathrm{Err}_k = \sum_{i \in C_k} I(y_i \neq \hat{y}_i)/n_k$.
- The estimated standard deviation of CV_K is
$$\widehat{\mathrm{SE}}(\mathrm{CV}_K) = \sqrt{\sum_{k=1}^{K} (\mathrm{Err}_k - \overline{\mathrm{Err}_k})^2/(K-1)}$$
- This is a useful estimate, but strictly speaking, not quite valid. *Why not?*

Cross-validation: right and wrong

Consider a simple classifier applied to some two-class data:

1. Starting with 5000 predictors and 50 samples, find the predictors having the largest correlation with the class labels
2. We then apply a classifier such as logistic regression, using only these 100 predictors

How do we estimate the test set performance of this classifier?

Can we apply cross-validation in step 2, forgetting about step 1?

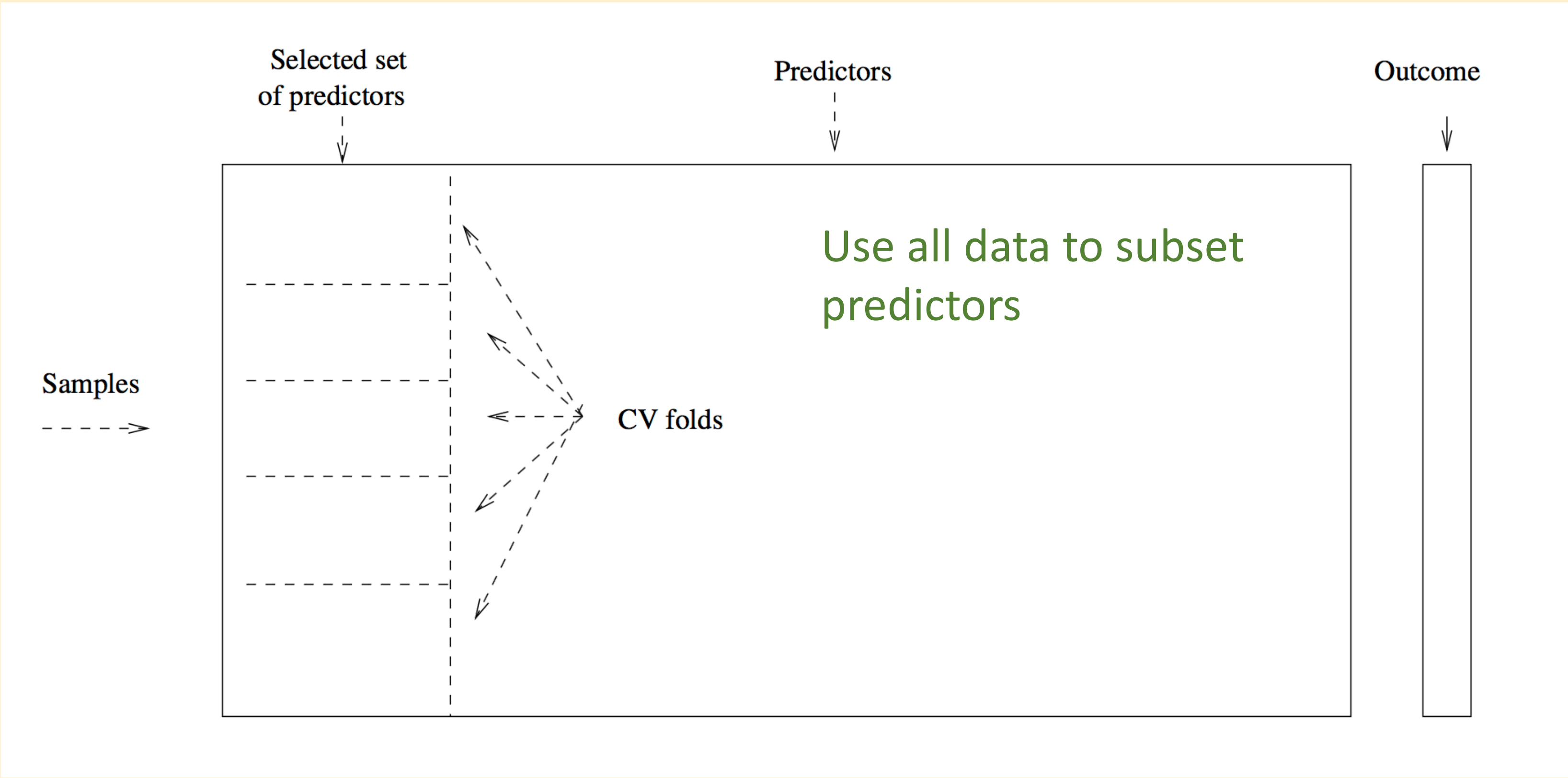
NO!

- This would ignore the fact that in Step 1, the procedure *has already seen the labels of the training data*, and made use of them. This is a form of training and must be included in the validation process.
- It is easy to simulate realistic data with the class labels independent of the outcome, so that true test error =50%, but the CV error estimate that ignores Step 1 is zero!
  Try to do this yourself
- This error has been made in many high profile genomics papers

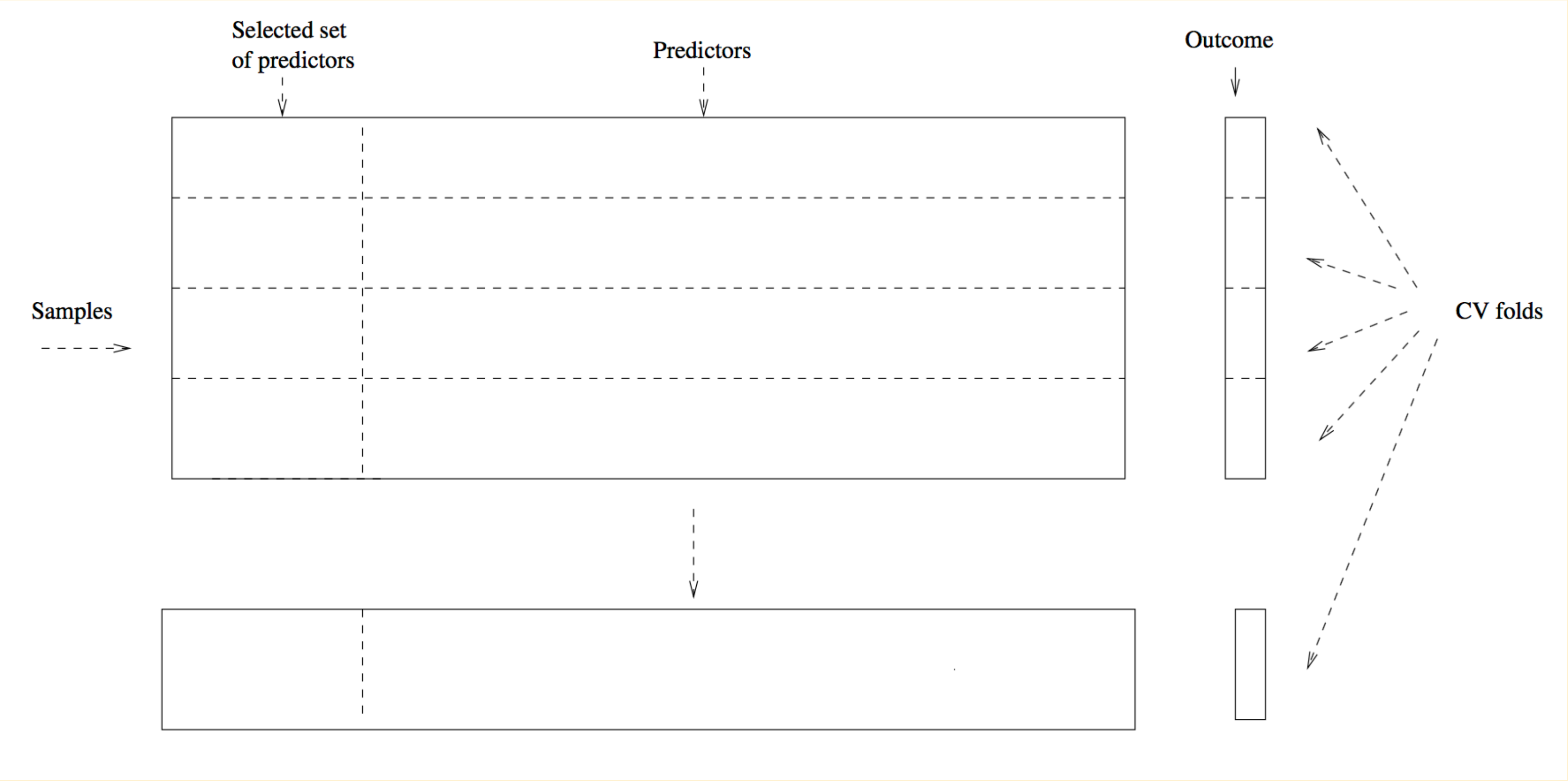
The Wrong and Right Way

- *Wrong:* Apply cross-validation in step 2
- *Right:* Apply cross-validation to steps 1 and 2.

Wrong way

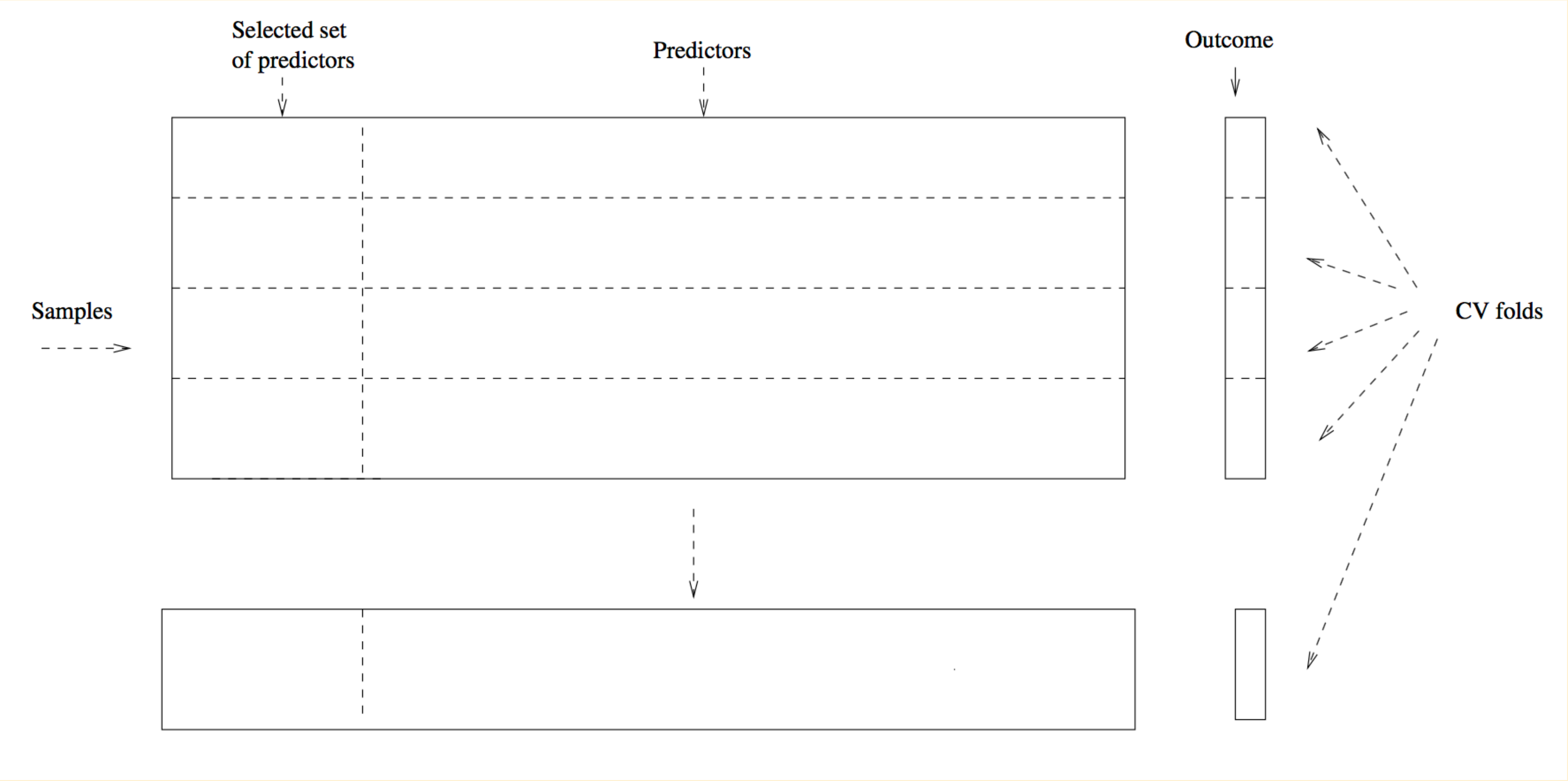

Selected set of predictors
Predictors
Outcome
Use all data to subset predictors
Samples
CV folds

Right way



Selected set
of predictors
Predictors
Outcome
Samples
CV folds

Right way



This means you will get different subsets of predictors for predicting each fold

Summary

- Matrix algebra
- Training-test validation
- Cross-validation

Next Lecture

- Support vector machines
- And maybe a few other bits and pieces