The course provides an introduction to the mathematics of data analysis and a detailed overview of statistical models for inference and prediction.

**Course Structure:**

- Weekly lectures [notes: www.ece.udel.edu/~arce/Courses]
- Homework & computer assignments [30%]
- Midterm & Final examinations [70%]

**Textbooks:**

- Papoulis and Pillai, Probability, random variables, and stochastic processes.
- Hastie, Tibshirani and Friedman, The elements of statistical learning.
- Haykin, Adaptive Filter Theory.
Logistic Regression

The logistic regression model arises from the desire to model the posterior probabilities of the $K$ classes via linear functions in $x$, while at the same time ensuring that they sum to one and remain in $[0, 1]$.

\[
\log \frac{Pr(G = 1 | X = x)}{Pr(G = K | X = x)} = \beta_{10} + \beta_1^T x \\
\log \frac{Pr(G = 2 | X = x)}{Pr(G = K | X = x)} = \beta_{20} + \beta_2^T x \\
\vdots \\
\log \frac{Pr(G = K - 1 | X = x)}{Pr(G = K | X = x)} = \beta_{(K-1)0} + \beta_1^T x
\]
Logistic Regression

Here $\boldsymbol{\beta}_k = [\beta_{k1}, \beta_{k2}, \ldots, \beta_{kp}]^T$, $\mathbf{x} = [x_1, x_2, \ldots, x_p]^T$.

The model is specified in terms of $K - 1$ log-odds or logit transformations (reflecting the constraint that the probabilities sum to one).

A simple calculation shows that

$$
Pr(G = k | X = \mathbf{x}) = \frac{\exp(\beta_{k0} + \mathbf{\beta}_k^T \mathbf{x})}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \mathbf{\beta}_l^T \mathbf{x})}, \quad k = 1, 2, \ldots, K - 1,
$$

(2)

$$
Pr(G = K | X = \mathbf{x}) = \frac{1}{1 + \sum_{l=1}^{K-1} \exp(\beta_{l0} + \mathbf{\beta}_l^T \mathbf{x})}
$$

To emphasize the dependence on the entire parameter set $\theta = \{\beta_{10}, \mathbf{\beta}_1^T, \ldots, \beta_{(K-1)0}, \mathbf{\beta}_{K-1}^T\}$, we denote the probabilities $Pr(G = k | X = \mathbf{x}) = p_k(\mathbf{x}; \theta)$.
Logistic regression models are usually fit by maximum likelihood, using the conditional likelihood of $G$ given $X$.

The log-likelihood for $N$ observations is

$$l(\theta) = \sum_{i=1}^{N} \log p_{g_i}(x_i; \theta)$$

where $p_k(x_i; \theta) = Pr(G = k | X = x_i; \theta)$.
Fitting Logistic Regression Models

Discuss in detail the two-class case.

- Code the two-class $g_i$ via a 0/1 response $y_i$, where $y_i = 1$ when $g_i = 1$, and $y_i = 0$ when $g_i = 2$.

- Let $p_1(x; \theta) = p(x; \theta)$, $p_2(x; \theta) = 1 - p(x; \theta)$

The log-likelihood can be written as

$$l(\theta) = \sum_{i=1}^{N} \{y_i \log p(x_i; \beta) + (1 - y_i) \log (1 - p(x_i; \beta))\}$$

$$= \sum_{i=1}^{N} \{y_i (\log p(x_i; \beta) - \log (1 - p(x_i; \beta))) + \log (1 - p(x_i; \beta))\}$$

$$= \sum_{i=1}^{N} \left\{y_i \log \frac{Pr(G = 1|X = x_i)}{Pr(G = 2|X = x_i)} + \log (Pr(G = 2|X = x_i))\right\}$$

$$= \sum_{i=1}^{N} \{y_i \beta^T x_i - \log (1 + e^{\beta^T x_i})\}$$
Fitting Logistic Regression Models

Here $\mathbf{\beta} = \{\beta_{10}, \beta_1\}$, and we assume that the vector of inputs $\mathbf{x}_i$ includes the constant term 1 to accommodate the intercept.

To maximize the log-likelihood, we set its derivatives to zero. These score equations are

$$\frac{\partial l(\mathbf{\beta})}{\partial \mathbf{\beta}} = \sum_{i=1}^{N} \mathbf{x}_i (y_i - p(\mathbf{x}_i; \mathbf{\beta})) = 0 \quad (5)$$

which are $p + 1$ equations nonlinear in $\mathbf{\beta}$.

Since the first component of $\mathbf{x}_i$ is 1, the first score equation specifies that $\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} p(\mathbf{x}_i; \mathbf{\beta})$, the expected number of class ones matches the observed number.
Newton–Raphson algorithm

To solve the score equations (5), we use the Newton–Raphson algorithm, which requires the second-derivative or Hessian matrix

\[
\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = - \sum_{i=1}^{N} x_i x_i^T p(x_i; \beta)(1 - p(x_i; \beta))
\]  

(6)

Starting with \( \beta^{old} \), a single Newton update is

\[
\beta^{new} = \beta^{old} - \left( \frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} \right)^{-1} \frac{\partial l(\beta)}{\partial \beta}
\]

(7)

where the derivatives are evaluated at \( \beta^{old} \).
Newton Raphson algorithm

- Write the score and Hessian in matrix notation.
- \( y \): \( N \) vector of \( y_i \) values.
- \( x_i \): \( p + 1 \) vector of input.
- \( X \): \( N \times (p + 1) \) matrix of \( x_i \) values.
- \( p \): \( N \) vector of fitted probabilities with \( i \)th element \( p(x_i; \beta_{\text{old}}) \).
- \( W \): \( N \times N \) diagonal matrix of weights with \( i \)th diagonal element \( p(x_i; \beta_{\text{old}})(1 - p(x_i; \beta_{\text{old}})) \).
Fitting Logistic Regression Models

Then we have

$$\frac{\partial I(\beta)}{\partial \beta} = X^T(y - p)$$

$$\frac{\partial^2 I(\beta)}{\partial \beta \partial \beta^T} = -X^TWX$$

(8)

The Newton step is thus

$$\beta^{new} = \beta^{old} + (X^TWX)^{-1}X^T(y - p)$$

$$= (X^TWX)^{-1}X^TW(X\beta^{old} + W^{-1}(y - p))$$

$$= (X^TWX)^{-1}X^TWz$$

(9)
In the second and third line we have re-expressed the Newton step as a weighted least squares step, with the response

\[ z = X\beta^{old} + W^{-1}(y - p) \]  

sometimes known as the *adjusted response*.

These equations get solved repeatedly, since at each iteration \( p \) changes, and hence so does \( W \) and \( z \).

This algorithm is referred to as *iteratively reweighted least squares* or IRLS, since each iteration solves the weighted least squares problem:

\[
\beta^{new} \leftarrow \arg\min_{\beta} (z - X\beta)^T W (z - X\beta)
\]
Exercise 4.10

Weekly data set is part of the ISLR package. It contains 1089 weekly percentage returns for 21 years, from the beginning of 1990 to the end of 2010.

<table>
<thead>
<tr>
<th>Lag1 to Lag5</th>
<th>The percentage returns for each of the five previous trading weeks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume</td>
<td>The number of shares traded on the previous week, in billions</td>
</tr>
<tr>
<td>Today</td>
<td>The percentage return on the week in question</td>
</tr>
<tr>
<td>Direction</td>
<td>Whether the market was Up or Down on this week</td>
</tr>
</tbody>
</table>
Exercise

(b) Use the full data set to perform a logistic regression.

library(ISLR)
attach(Weekly)
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, data=Weekly, family=binomial)
summary(glm.fit)

According to p-value, lag1, lag2 and lag4 appear to be statistically significant.
(c) Compute the confusion matrix and overall fraction of correct predictions.

```r
glm.probs = predict(glm.fit, type = "response")
glm.pred = rep("Down", 1089)
glm.pred[glm.probs > .5] = "Up"
table(glm.pred, Direction)
mean(glm.pred == Direction)
```

```
> table(glm.pred, Direction)

     Direction
glm.pred Down  Up
   Down  54  48
   Up    430 557

> mean(glm.pred == Direction)
[1] 0.5610652
```
(d) Fit the logistic regression model using a training data then predict.

```r
train=(Year<2009)
A=(Year==2009)
B=(Year==2010)
Weekly.2009=Weekly[A,]
Weekly.2010=Weekly[B,]
Direction.2009=Direction[A]
Direction.2010=Direction[B]
glm.fit=glm(Direction∼Lag2,data=Weekly,family=binomial,subset=train)
glm.probs=predict(glm.fit,Weekly.2009,type="response")
glm.pred =rep("Down",52)
glm.pred [glm.probs>.5]="Up"
table(glm.pred,Direction.2009)
mean(glm.pred==Direction.2009)
```
Exercise

**Figure:** prediction for 2009

```r
> table(glm.pred, Direction.2009)

<table>
<thead>
<tr>
<th>Direction.2009</th>
<th>glm.pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Down</td>
<td>4</td>
</tr>
<tr>
<td>Up</td>
<td>19</td>
</tr>
</tbody>
</table>

> mean(glm.pred==Direction.2009)

[1] 0.5576923

**Figure:** prediction for 2010

```r
> table(glm.pred, Direction.2010)

<table>
<thead>
<tr>
<th>Direction.2010</th>
<th>glm.pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>Down</td>
<td>5</td>
</tr>
<tr>
<td>Up</td>
<td>15</td>
</tr>
</tbody>
</table>

> mean(glm.pred==Direction.2010)

[1] 0.6923077
```
(d) Fit the LDA model using a training data then predict.

```r
library(MASS)
lda.fit=lda(Direction~Lag2,data=Weekly,subset=train)
lda.pred=predict(lda.fit,Weekly.2009)
lda.class=lda.pred$class
table(lda.class,Direction.2009)
mean(lda.class==Direction.2009)
```

<table>
<thead>
<tr>
<th></th>
<th>Down</th>
<th>Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>lda.class Down</td>
<td>4</td>
<td>19</td>
</tr>
<tr>
<td>lda.class Up</td>
<td>4</td>
<td>25</td>
</tr>
</tbody>
</table>

> mean(lda.class==Direction.2009)

[1] 0.5576923

**Figure:** prediction for 2009

<table>
<thead>
<tr>
<th></th>
<th>Down</th>
<th>Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>lda.class Down</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>lda.class Up</td>
<td>1</td>
<td>31</td>
</tr>
</tbody>
</table>

> mean(lda.class==Direction.2010)

[1] 0.6923077

**Figure:** prediction for 2010
(e) Fit the QDA model using a training data then predict.

```r
qda.fit = qda(Direction ~ Lag2, data = Weekly, subset = train)
qda.pred = predict(qda.fit, Weekly.2009)
qda.class = qda.pred$class
table(qda.class, Direction.2009)
mean(qda.class == Direction.2009)
```

Figure: prediction for 2009

Figure: prediction for 2010