Predicting a New Response

Recall the regression model

\[ y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \epsilon = \mathbf{x}' \beta + \epsilon, \]

and the estimated mean response at \( \mathbf{x}_0 \):

\[ \hat{y}(\mathbf{x}_0) = \mathbf{x}_0' \hat{\beta}. \]

To predict a single new response at \( \mathbf{x}_0 \), we still use \( \hat{y}(\mathbf{x}_0) \) as the best predictor, but the mean squared prediction error is

\[ \sigma^2 \left[ 1 + \mathbf{x}_0' (\mathbf{X}'\mathbf{X})^{-1} \mathbf{x}_0 \right]. \]
So the $100(1 - \alpha)\%$ prediction interval is

$$\hat{y}(x_0) \pm t_{\alpha/2, n-p} \sqrt{\hat{\sigma}^2 \left[1 + x'_0 (X'X)^{-1} x_0\right]}$$

Note that this is wider than the $100(1 - \alpha)\%$ confidence interval for the mean response at $x_0$,

$$\hat{y}(x_0) \pm t_{\alpha/2, n-p} \sqrt{\hat{\sigma}^2 x'_0 (X'X)^{-1} x_0}$$

because the prediction interval must allow for the $\epsilon$ in the new response:

new response = mean response + $\epsilon$. 
R command

Use predict(..., interval = "prediction"): 

```r
predict(viscosityLm, 
    newdata = data.frame(Temperature = 90, CatalystFeedRate = 10), 
    se.fit = TRUE, interval = "prediction")
```

Output

```r
$fit
   fit   lwr   upr
1 2337.842 2301.360 2374.325

$se.fit
[1] 4.192114

$df
[1] 13

$residual.scale
[1] 16.35860
```
The prediction interval is centered at the same value of fit as the confidence interval.

The prediction interval is wider than the confidence interval, because of the variability in a single observation.

Both of these intervals used the default confidence/prediction level of 95%; use `predict(..., level = .99)`, for instance, to change the level.
Regression Diagnostics

Standard residual plots are:
- Qq-plot (probability plot) of residuals;
- Plot residuals against fitted values;
- Plot residuals against regressors;
- Plot (square roots of) absolute residuals against fitted values.

More diagnostics are usually examined after a regression analysis.
Scaled Residuals

Residuals are usually scaled in various ways.

E.g. the *standardized* residual

\[ d_i = \frac{e_i}{\sqrt{\hat{\sigma}^2}}, \]

is dimensionless; they satisfy

\[ \sum_{i=1}^{n} d_i = 0 \]

and

\[ \sum_{i=1}^{n} d_i^2 = n - p. \]
The $d_i$ are therefore “standardized” in an average sense.

But the standard deviation of the $i^{th}$ residual $e_i$ usually depends on $i$.

So the $d_i$ are not individually standardized.
The hat matrix:

\[ \hat{y} = X\hat{\beta} = X(X'X)^{-1}X'y = Hy \]

where \( H = X(X'X)^{-1}X' \) is the hat matrix (so called because it “puts the hat on \( y \”).

So the residuals \( e \) satisfy

\[ e = y - \hat{y} = (I - H)y \]

and

\[ \text{Cov}(e) = \sigma^2(I - H). \]
So the variance of the $i^{th}$ residual is

$$V(e_i) = \sigma^2 \left(1 - h_{i,i}\right)$$

where $h_{i,i}$ is the $i^{th}$ diagonal entry of $\mathbf{H}$.

The *studentized* residual is

$$r_i = \frac{e_i}{\sqrt{\hat{\sigma}^2 \left(1 - h_{i,i}\right)}} = \frac{d_i}{\sqrt{(1 - h_{i,i})}}$$

with population mean 0 and variance 1 for each $i$:

$$E(r_i) = 0, \quad V(r_i) = 1.$$
Cross Validation

Suppose we predict $y_i$ from a data set excluding $y_i$.

New parameter estimates $\hat{\beta}_{(i)}$.

Predicted value is $\hat{y}(i) = x_i'\hat{\beta}_{(i)}$ and the corresponding residual satisfies

$$e(i) = y_i - \hat{y}(i) = \frac{e_i}{1 - h_{i,i}}.$$ 

The PRediction Error Sum of Squares statistic (PRESS) is

$$\text{PRESS} = \sum_{i=1}^{n} e_{(i)}^2.$$
Approximate $R^2$ for prediction:

$$R^2_{\text{prediction}} = 1 - \frac{\text{PRESS}}{SS_T}.$$ 

E.g. for viscosity example, PRESS = 5207.7, so $R^2_{\text{prediction}} = .891$.

Recall $R^2 = .927$ and $R^2_{\text{adj}} = .916$: $R^2_{\text{prediction}}$ penalizes over-fitting more than does $R^2_{\text{adj}}$. 
R function

RsqPred <- function(l) {
  infl <- influence(l)
  PRESS <- sum((infl$wt.res / (1 - infl$hat))^2)
  rsq <- summary(l)$r.squared
  sst <- sum(infl$wt.res^2) / (1 - rsq)
  1 - PRESS / sst
}

RsqPred(lm(Viscosity ~ CatalystFeedRate + Temperature, viscosity))

[1] 0.8906768
One more scaled residual: $R$-student is like the studentized residual, but $\sigma^2$ is estimated \textit{from the data set with $y_i$ excluded}:

$$t_i = \frac{e_i}{\sqrt{S^2_{(i)} (1 - h_{i,i})}}.$$  

Under the usual normal distribution assumptions for $\epsilon_i$, $R$-student has Student’s $t$-distribution with $n - p - 1$ degrees of freedom.

We can use $t$-tables to test for outliers.
Leverage and Influence

When the $h_{i,i}$ are not all equal, each observation has its own weight in determining the fit, usually measured by $h_{i,i}$.

Average value of $h_{i,i}$ is always $p/n$.

Conventionally, if $h_{i,i} > 2p/n$, $x_i$ is a high leverage point.
High leverage points do not mean that a fit is bad, just sensitive to outliers.

Cook’s $D_i$ measures how much the parameter estimates are affected by excluding $y_i$:

$$D_i = \frac{\left(\hat{\beta}_{(i)} - \hat{\beta}\right)\prime X'X \left(\hat{\beta}_{(i)} - \hat{\beta}\right)}{p \times MS_E}$$

$$= \frac{r_i^2}{p} \times \frac{h_{i,i}}{1 - h_{i,i}} = \frac{e_i^2}{p\hat{\sigma}^2} \times \frac{h_{i,i}}{(1 - h_{i,i})^2}.$$

$D_i > 1 \Rightarrow i^{th}$ observation has high influence.
R commands

```r
cooks.distance(viscosityLm)
max(cooks.distance(viscosityLm))
which.max(cooks.distance(viscosityLm))
```

Output

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1.370211e-01</td>
<td>2.328096e-02</td>
<td>4.631904e-02</td>
<td>6.051051e-04</td>
<td>3.033025e-02</td>
<td>2.768744e-01</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>3.718495e-03</td>
<td>9.699079e-02</td>
<td>1.109258e-01</td>
<td>1.108032e-02</td>
<td>3.538676e-01</td>
<td>6.183881e-02</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.615253e-02</td>
<td>3.097853e-06</td>
<td>2.331613e-02</td>
<td>4.676090e-03</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

No \( D_i > 1 \), so no individual data point has too much influence on \( \hat{\beta} \).
The function `influence()` produces:

- hat values $h_{i,i}$ in $\hat{h}$;
- leave-one-out parameter estimate changes $\hat{\beta}_i - \hat{\beta}$ in $\hat{\beta}$;
- leave-one-out standard deviation estimates $S(i)$ in $\hat{\sigma}$;
- ordinary residuals in $\text{wt.res}$. 
R command

influence(lm(Viscosity ~ CatalystFeedRate + Temperature, viscosity))

Output

$\hat{\epsilon}$

\[
\begin{array}{cccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 \\
0.34950693 & 0.10247249 & 0.17667095 & 0.25108380 & 0.07689010 & 0.26532800 & 0.31935115 \\
8 & 9 & 10 & 11 & 12 & 13 & 14 \\
0.09797056 & 0.14189415 & 0.07989138 & 0.27835739 & 0.09618408 & 0.28948121 & 0.18519842 \\
15 & 16 \\
0.13415273 & 0.15556667 \\
\end{array}
\]
$coefficients$

<table>
<thead>
<tr>
<th></th>
<th>(Intercept)</th>
<th>CatalystFeedRate</th>
<th>Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>35.63580480</td>
<td>-0.877700039</td>
<td>-0.280170980</td>
</tr>
<tr>
<td>2</td>
<td>-1.33084057</td>
<td>0.376379857</td>
<td>-0.036994601</td>
</tr>
<tr>
<td>3</td>
<td>-14.98857700</td>
<td>-0.086739050</td>
<td>0.184190892</td>
</tr>
<tr>
<td>4</td>
<td>-1.18980912</td>
<td>-0.054701202</td>
<td>0.018255822</td>
</tr>
<tr>
<td>5</td>
<td>-0.72738417</td>
<td>-0.297566252</td>
<td>0.029247965</td>
</tr>
<tr>
<td>6</td>
<td>13.21234005</td>
<td>1.458155876</td>
<td>-0.328860687</td>
</tr>
<tr>
<td>7</td>
<td>-5.74474826</td>
<td>0.151407566</td>
<td>0.043914749</td>
</tr>
<tr>
<td>8</td>
<td>-14.49483414</td>
<td>-0.163180204</td>
<td>0.196756377</td>
</tr>
<tr>
<td>9</td>
<td>17.57548135</td>
<td>-0.970879305</td>
<td>-0.100847328</td>
</tr>
<tr>
<td>10</td>
<td>-3.67591761</td>
<td>0.174935976</td>
<td>0.027899272</td>
</tr>
<tr>
<td>11</td>
<td>-41.98605988</td>
<td>1.943496093</td>
<td>0.264053698</td>
</tr>
<tr>
<td>12</td>
<td>13.80976682</td>
<td>-0.374480138</td>
<td>-0.12507098</td>
</tr>
<tr>
<td>13</td>
<td>-5.86991930</td>
<td>-0.504921342</td>
<td>0.128078772</td>
</tr>
<tr>
<td>14</td>
<td>0.05401898</td>
<td>0.004540686</td>
<td>-0.001024070</td>
</tr>
<tr>
<td>15</td>
<td>11.89223268</td>
<td>-0.325259570</td>
<td>-0.085816755</td>
</tr>
<tr>
<td>16</td>
<td>-1.01518787</td>
<td>-0.192015133</td>
<td>0.029369774</td>
</tr>
</tbody>
</table>
Output, continued

$\sigma$

1 2 3 4 5 6 7 8
9 10 11 12 13 14 15 16

$wt$.$res$

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
11.5402553 5 -12.1213615 4 11.9447620 -1.0417083 5 -16.4271831 4 -21.2642561
7 8 9 10 11 12
13 14 15 16
7.1144538 3 0.0944214 2 10.2276690 2 -4.1481587 3
R command

Leverage and Cook’s $D_i$ are shown in the fourth residual plot:

```r
plot(lm(Viscosity ~ CatalystFeedRate + Temperature, viscosity))
```
Testing for Lack of Fit

In regression analysis, $t$-statistics and $F$-ratios are computed using

$$s^2 = \text{Mean Square for Residuals}$$

as the estimate of $\sigma^2$.

But $s^2$ is an unbiased estimator of $\sigma^2$ only if the model is correctly specified.

If the design has replicated observations, the residual sum of squares can be decomposed into pure error and lack of fit:

$$SS_E = SS_{PE} + SS_{LOF}$$
Example

Problem 10.12

Problem10p12 <- read.table("data/Problem-10-12.txt", header = TRUE)
summary(lm(y ~ x1 + x2, Problem10p12))

Output

Coefficients:

|          | Estimate | Std. Error | t value   | Pr(>|t|) |
|----------|----------|------------|-----------|----------|
| (Intercept) | 49.635   | 7.988      | -6.214    | 0.000156 *** |
| x1       | 18.355   | 7.615      | 2.410     | 0.039218 *   |
| x2       | 46.116   | 2.887      | 15.975    | 6.52e-08 *** |

---
Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 9.483 on 9 degrees of freedom
Multiple R-squared: 0.9771, Adjusted R-squared: 0.972
F-statistic: 191.8 on 2 and 9 DF, p-value: 4.178e-08
Break down the residual sum of squares by adding `factor(x1):factor(x2)` to the analysis of variance of the model:

```
summary(aov(y ~ x1 + x2 + factor(x1) : factor(x2), Problem10p12))
```

**Output**

<table>
<thead>
<tr>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>1</td>
<td>11552</td>
<td>11552</td>
<td>270.750</td>
</tr>
<tr>
<td>x2</td>
<td>1</td>
<td>22950</td>
<td>22950</td>
<td>537.898</td>
</tr>
<tr>
<td>factor(x1):factor(x2)</td>
<td>6</td>
<td>681</td>
<td>114</td>
<td>2.662</td>
</tr>
<tr>
<td>Residuals</td>
<td>3</td>
<td>128</td>
<td>43</td>
<td></td>
</tr>
</tbody>
</table>

---

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

**Note**

Do *not* add this interaction to the formula in `lm()`. It changes the estimated regression coefficients!