Multiple linear regression
Outline for today

Review of linear regression

Multiple linear regression

Manipulating data with dplyr
Big Data Baseball

Thoughts?
Questions about worksheet 4?
Regression

Regression is method of using one variable to predict the value of a second variable

In linear regression we fit a line to the data, called the regression line.

\[ \hat{y} = a + b \cdot x \]

*Response = a + b \cdot Explanatory*
Runs and BA regression

\[ \hat{y} = a + b \cdot x \]

\[ a = -529.8 \]

\[ b = 4755.7 \]

R: `fit <- lm(y ~ x)`

\[ \hat{r} = -529.8 + 4755.7 \cdot BA \]
The residual at a data value is the difference between the observed \( (y) \) and predicted value \( (\hat{y}) \) of the response variable

\[
\text{Residual} = \text{Observed} - \text{Predicted} = y - \hat{y}
\]
Measuring goodness of fit

If the residuals are small, then the line does a good job describing the data. We can measure how well the line fits the data using the equation:

\[ MSE = \frac{1}{n} \sum_{i}^{n} (\hat{y}_i - y_i)^2 \]
Least squares line

The **least squares line**, also called “**the line of best fit**”, is the line which **minimizes the sum of squared residuals** i.e., the least squares line is the line that minimizes the Mean Squared Error (MSE)

\[
MSE = \frac{1}{n} \sum_{i}^{n} (\hat{y}_i - y_i)^2
\]

[https://emeyers.shinyapps.io/baseball_regression_app/](https://emeyers.shinyapps.io/baseball_regression_app/)
Linear regression in R

```r
fit <- lm(team.batting.162$R ~ team.batting.162$BA)

What is the output ln.fit???

> class(2)
[1] "numeric"

> class(fit)
[1] "lm"  # actually a list

> names(fit)
[1] "coefficients" "residuals" "effects" "rank" "fitted.values" "assign"
[7] "qr" "df.residual" "xlevels" "call" "terms" "model"

> ln.fit$coefficients
          (Intercept) team.batting.162$BA
-786.211       5797.657
```
Compare batting measures based on RMSE

We can find the ‘best’ statistic based on the RMSE

• i.e., which statistic leads to a model with the minimal squared residuals
Compare batting measures based on RMSE

<table>
<thead>
<tr>
<th></th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>60.76</td>
</tr>
<tr>
<td>BA</td>
<td>51.35</td>
</tr>
<tr>
<td>OBP</td>
<td>39.74</td>
</tr>
<tr>
<td>Slug</td>
<td>36.95</td>
</tr>
<tr>
<td>OPS</td>
<td>27.46</td>
</tr>
</tbody>
</table>
Compare batting measures based on RMSE

<table>
<thead>
<tr>
<th></th>
<th>RMSE</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>60.76</td>
<td>0.74</td>
</tr>
<tr>
<td>BA</td>
<td>51.35</td>
<td>0.82</td>
</tr>
<tr>
<td>OBP</td>
<td>39.74</td>
<td>0.90</td>
</tr>
<tr>
<td>Slug</td>
<td>36.95</td>
<td>0.91</td>
</tr>
<tr>
<td>OPS</td>
<td>27.46</td>
<td>0.95</td>
</tr>
</tbody>
</table>

$$r^2 = 1 - \frac{\text{MSE}}{\text{var}(y)} \cdot \frac{[n-1]}{n}$$
Regression caution # 1

Avoid trying to apply the regression line to predict values far from those that were used to create the line. i.e., do not extrapolate too far.
Regression caution # 2

Plot the data! Regression lines are only appropriate when there is a linear trend in the data.
Regression caution #3

Be aware of outliers – they can have an huge effect on the regression line.
Multiple regression
Creating better statistics

On-base plus slugging (OPS and BRA) were the best statistics for predicting runs scored we have found so far.

But who says we can’t do better!
Creating better statistics

Batting average:

\[ BA = \frac{1 \cdot 1B + 1 \cdot 2B + 1 \cdot 3B + 1 \cdot HR}{AB} \]

Slugging percentage:

\[ \text{Slug} = \frac{1 \cdot 1B + 2 \cdot 2B + 3 \cdot 3B + 4 \cdot HR}{AB} \]

On-base percentage:

\[ \text{OBP} = \frac{1 \cdot BB + 1 \cdot HBP + 1 \cdot 1B + 1 \cdot 2B + 1 \cdot 3B + 1 \cdot HR}{PA} \]

Optimal statistic:

\[ \text{OPT} = w_1 \cdot BB + w_2 \cdot HBP + w_3 \cdot 1B + w_4 \cdot 2B + w_5 \cdot 3B + w_6 \cdot HR + w_0 \]
What are the optimal weights?

Any ideas for the best $w_i$’s?

$$OPT = w_1 \cdot BB + w_2 \cdot HBP + w_3 \cdot 1B + w_4 \cdot 2B + w_5 \cdot 3B + w_6 \cdot HR + w_0$$

Let’s find the $w_i$’s that minimize: sum of $(OPT - R)^2$

Fitting multiple weights is called **multiple regression**

Fortunately for least squares fits this is easy to do in R:

```r
> fit <- lm(R ~ BB + HBP + H + X2B + X3B + HR, data = team.batting.162)
```
What are the optimal weights?

<table>
<thead>
<tr>
<th></th>
<th>w_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-497.44</td>
</tr>
<tr>
<td>HBP</td>
<td>0.42</td>
</tr>
<tr>
<td>BB</td>
<td>0.34</td>
</tr>
<tr>
<td>X1B</td>
<td>0.56</td>
</tr>
<tr>
<td>X2B</td>
<td>0.75</td>
</tr>
<tr>
<td>X3B</td>
<td>1.40</td>
</tr>
<tr>
<td>HR</td>
<td>1.44</td>
</tr>
</tbody>
</table>

> `coef(fit)`

Do these weights make sense?

Trying writing this in the form of an equation
What are the optimal weights?

\[
\hat{r} = 0.34 \cdot BB + 0.42 \cdot HBP + 0.56 \cdot 1B + 0.75 \cdot 2B + 1.40 \cdot 3B + 1.44 \cdot HR - 497.44
\]

<table>
<thead>
<tr>
<th>(\text{(Intercept)})</th>
<th>(w_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-497.44</td>
</tr>
<tr>
<td>HBP</td>
<td>0.42</td>
</tr>
<tr>
<td>BB</td>
<td>0.34</td>
</tr>
<tr>
<td>X1B</td>
<td>0.56</td>
</tr>
<tr>
<td>X2B</td>
<td>0.75</td>
</tr>
<tr>
<td>X3B</td>
<td>1.40</td>
</tr>
<tr>
<td>HR</td>
<td>1.44</td>
</tr>
</tbody>
</table>
How good is our optimal statistic?

<table>
<thead>
<tr>
<th>Metric</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>60.42</td>
</tr>
<tr>
<td>BA</td>
<td>42.17</td>
</tr>
<tr>
<td>OBP</td>
<td>31.83</td>
</tr>
<tr>
<td>SlugPct</td>
<td>31.55</td>
</tr>
<tr>
<td>OPS</td>
<td>23.46</td>
</tr>
<tr>
<td>OPT</td>
<td>23.28</td>
</tr>
</tbody>
</table>

Can we do even better???

What if we include PA?
How good is our optimal statistic?

<table>
<thead>
<tr>
<th></th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>60.42</td>
</tr>
<tr>
<td>BA</td>
<td>42.17</td>
</tr>
<tr>
<td>OBP</td>
<td>31.83</td>
</tr>
<tr>
<td>SlugPct</td>
<td>31.55</td>
</tr>
<tr>
<td>OPS</td>
<td>23.46</td>
</tr>
<tr>
<td>OPT</td>
<td>23.28</td>
</tr>
<tr>
<td>OPT*</td>
<td>22.90</td>
</tr>
</tbody>
</table>

Can we do even better than this???

What if we included BRA?

or ... BRA\(^2\), or (SB*BRA)\(^7\) or ... ???
How low can we go???

Let’s see who can come up with the lowest RMSE by creating more complex regression models!!!

```r
> fit <- lm(R ~ BB + HBP + SB, data = team.batting.162)
```

Remember we can add new variables to a data frame using:

```r
> team.batting.162$Hsqrd <- (team.batting.162$H)^2
```

A quicker way to calculate the RMSE is:

```r
> sqrt(mean(fit$residuals^2))
```
Who got the lowest RMSE?

Do we believe that this is the best model for predicting runs?
Overfitting

One problem with our optimal statistic is that we used the same data to fit our model (find the $w_i$’s) as we did to evaluate whether it was a good fit.

Thus it is possible that our weights we found we too tailored to the data at hand and our estimate of

Fitting a model to precisely to the data at hand in such a way that it does not generalize to new data is called overfitting.
Overfitting

One should always use different data when fitting and evaluating a model!
Cross-validation is a method for assessing the goodness of a model in a way that can avoid overfitting.

What we do is build the model on one set of data, called the training set.
- i.e., find the coefficients on one set of data

Then we evaluate whether the model fits well on a second set of data, called the test set.

If the model is truly good, we should get good predictions on the test set.
- i.e., a small RMSE on the test set
Creating training and test sets

load('/home/shared/baseball_stats_2017/team_batting_stats.Rda')

num.cases <- dim(team.batting.162)[1]

half.point <- round(num.cases/2)

train.data <- team.batting.162[1:half.point, ]

test.data <- team.batting.162[(1 + half.point):num.cases, ]

Even better would be to choose random cases in the training and test data, rather than the 1\textsuperscript{st} half of cases in the training data and the second half of the cases in the test data.
Cross-validation in R

Build a model from the training data:
```r
> fit.train <- lm(R ~ BB + TB + X2B + X3B + HR + PA + BB*SB + SB + PA, data = train.data)
```

Calculate RMSE from the same training data:
```r
> sqrt(mean((train.data$R - predict(fit.train))^2))
[1] 22.86
```

Make predictions on data from the test:
```r
> predictions.test <- predict(fit.train, newdata = test.data)
```

Calculate the RMSE on these predictions from the test:
```r
> sqrt(mean((test.data$R - predictions.test)^2))
[1] 25.03
```
Fitting on the 2012 season, measuring the fit on the 2013 season

<table>
<thead>
<tr>
<th></th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>HR</td>
<td>62.85</td>
</tr>
<tr>
<td>BA</td>
<td>49.29</td>
</tr>
<tr>
<td>OBP</td>
<td>38.40</td>
</tr>
<tr>
<td>Slug</td>
<td>34.50</td>
</tr>
<tr>
<td>OPS</td>
<td>26.61</td>
</tr>
<tr>
<td>OPT*</td>
<td>30.39</td>
</tr>
</tbody>
</table>

“Optimal” fit no longer that optimal