Week 11 – Modeling Data

Joshua Rosenberg and Alex Lishinski

April 1, 2021

Welcome!

Welcome to *week 11*!

Record the meeting

Breakout rooms!

Starting with whoever has consumed the least caffeine (in chocolate, tea, or coffee) today . . .

- What was the most familiar to you about the code / techniques used in the walkthrough on illuminating inequities in education through analyzing large-scale data sets?
- What was the least familiar to you?

Prepare one–three responses to each of the above questions to share with the whole class!

https://datascienceineducation.com/c09.html

Topics for today

Record the meeting

Modeling continued

A. Recap from the buffet and further expansion

B. Model outputs and summaries

C. Model helpers – tests, diagnostics, and model components

A. Buffet of models recap

What a model is:

model: a simplified *representation* of your data that can be informative to you (and others) about your data – and, maybe, what your data represents.

From this broad definition, models can take many different forms:

- A sample statistic (e.g., a *mean* of a variable)
- A relationship describing how two variables co–vary (e.g., a bivariate *correlation*)
- A linear regression model

A. Buffet of models recap

How do we represent model equations in R? Many R packages share a common modeling syntax, or interface: the formula syntax.

This code represents the regression of `hp` upon `mpg`:

```
mpg ~ hp
```

Then there are other formula operators:

```
# additional independent variables
mpg ~ hp + disp
# interactions with main effects
mpg ~ hp + disp + hp*disp
# interactions without main effects
mpg ~ hp + disp + hp:disp
# All remaining variables
mpg ~ .
```

A. Buffet of models recap

There are a number of helper functions that work with `lm` and other models

And there's some more advanced formula tricks too

```
# Polynomial Regression
y ~ x + I(x^2) + I(x^3)
```

```
## y ~ x + I(x^2) + I(x^3)
```

```
# Factorial ANOVA
y ~ (a*b*c)^2
```

```
## y ~ (a * b * c)^2
```

```
# Variable transformations
Sepal.Width ~ Petal.Width + log(Petal.Length) + Species
```

```
## Sepal.Width ~ Petal.Width + log(Petal.Length) + Species
```

A. Buffet of models recap

The lm() function is the base case of using these model formulas

Estimating a model; seeing the result:

```
lm(FinalGradeCEMS ~ TimeSpent_hours, data = d)
```

```
##
## Call:
## lm(formula = FinalGradeCEMS ~ TimeSpent_hours, data = d)
##
## Coefficients:
##     (Intercept)  TimeSpent_hours
##         65.8085           0.3648
```

A. Buffet of models recap

The lm() function is the base case of using these model formulas

Saving the output to an *object* and printing a summary of the results

```
m1 <- lm(FinalGradeCEMS ~ TimeSpent_hours, data = d)
summary(m1)
```

```
##
## Call:
## lm(formula = FinalGradeCEMS ~ TimeSpent_hours, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -67.136  -7.805   4.723  14.471  30.317
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     65.80851    1.49120   44.13   <2e-16 ***
## TimeSpent_hours  0.36484    0.03889    9.38   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.71 on 571 degrees of freedom
##   (30 observations deleted due to missingness)
## Multiple R-squared:  0.1335,    Adjusted R-squared:  0.132
## F-statistic: 87.99 on 1 and 571 DF,  p-value: < 2.2e-16
```

A. Buffet of models recap

The lm() function is the base case of using these model formulas

Making the model more complex – a multiple regression

```
m2 <- lm(FinalGradeCEMS ~ TimeSpent_hours + int + Gender, data = d)
summary(m2)
```

```
##
## Call:
## lm(formula = FinalGradeCEMS ~ TimeSpent_hours + int + Gender,
##     data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.593  -7.382   4.761  14.534  30.618
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     69.61325    7.06075   9.859   <2e-16 ***
## TimeSpent_hours  0.36962    0.04198   8.804   <2e-16 ***
## int             -0.99359    1.58756  -0.626    0.532
## GenderM         -0.54962    2.06489  -0.266    0.790
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.03 on 499 degrees of freedom
##   (100 observations deleted due to missingness)
## Multiple R-squared:  0.1375,    Adjusted R-squared:  0.1323
## F-statistic: 26.51 on 3 and 499 DF,  p-value: 6.362e-16
```

A. Buffet of models recap

But other functions and packages use this as well

t–test

```
m_t_test <- t.test(FinalGradeCEMS ~ Gender, data = d)
m_t_test
```

```
##
##     Welch Two Sample t-test
##
## data:  FinalGradeCEMS by Gender
## t = -0.30379, df = 327.71, p-value = 0.7615
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.579370  3.354211
## sample estimates:
## mean in group F mean in group M
##        77.01877        77.63135
```

ANOVA

```
m_anova <- aov(FinalGradeCEMS ~ subject, data = d)
m_anova
```

```
## Call:
##    aov(formula = FinalGradeCEMS ~ subject, data = d)
##
## Terms:
##                   subject Residuals
## Sum of Squares   13484.46 269057.23
## Deg. of Freedom         4       568
##
## Residual standard error: 21.76447
## Estimated effects may be unbalanced
```

A. Buffet of models recap

Some packages add to this syntax too -- Multi-level model

```
library(lme4)
m5 <- lmer(FinalGradeCEMS ~ TimeSpent_hours + int*Gender + (1|course_id), data = d)
summary(m5)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: FinalGradeCEMS ~ TimeSpent_hours + int * Gender + (1 | course_id)
##    Data: d
##
## REML criterion at convergence: 4433.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4970 -0.4169  0.2413  0.6507  2.3171
##
## Random effects:
##  Groups    Name        Variance Std.Dev.
##  course_id (Intercept)  46.47    6.817
##  Residual              384.21   19.601
## Number of obs: 503, groups:  course_id, 26
##
## Fixed effects:
##                  Estimate Std. Error t value
## (Intercept)      74.22969    8.45385   8.781
## TimeSpent_hours   0.43078    0.04128  10.435
## int              -2.84129    1.89455  -1.500
## GenderM         -26.55507   13.10001  -2.027
## int:GenderM       6.39449    3.09236   2.068
##
## Correlation of Fixed Effects:
##             (Intr) TmSpn_ int    GendrM
## TimSpnt_hrs -0.239
## int         -0.963  0.091
## GenderM     -0.595  0.021  0.611
## int:GenderM  0.583 -0.027 -0.609 -0.989
```

A. Buffet of models recap

Weights using `lm()`

```
# Adding weights
wts1 <- rnorm(n = nrow(d), mean = 0, sd = 1)
wts2 <- rnorm(n = nrow(d), mean = 0, sd = 1)

wts <- abs(wts1/wts2)

m1_wt <- lm(FinalGradeCEMS ~ TimeSpent_hours, data = d, weights = wts)
summary(m1_wt)
```

```
##
## Call:
## lm(formula = FinalGradeCEMS ~ TimeSpent_hours, data = d, weights = wts)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -497.90  -11.04    2.67   16.23  298.08
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     57.09622    1.70425   33.50   <2e-16 ***
## TimeSpent_hours  0.65608    0.04876   13.46   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 49.4 on 571 degrees of freedom
##   (30 observations deleted due to missingness)
## Multiple R-squared:  0.2408,    Adjusted R-squared:  0.2394
## F-statistic: 181.1 on 1 and 571 DF,  p-value: < 2.2e-16
```

A. Buffet of models recap

The `glm()` function – the way to fit other types of regression models

`glm()` lets you specify distribution families for different kinds of outcomes:

- "gaussian" = standard regression for continuous dependent variables (default)
- "binomial" = binary (0/1) outcome variables
- "poisson" = count outcome variables
- several others are available

A. Buffet of models recap

The base case with default options will be same as `lm()`

```
glm1 <- glm(FinalGradeCEMS ~ TimeSpent_hours, data = d)
summary(glm1)
```

```
## 
## Call:
## glm(formula = FinalGradeCEMS ~ TimeSpent_hours, data = d)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -67.136   -7.805    4.723   14.471   30.317  
## 
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)     65.80851    1.49120   44.13   <2e-16 ***
## TimeSpent_hours  0.36484    0.03889    9.38   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for gaussian family taken to be 428.7479)
## 
##     Null deviance: 282542  on 572  degrees of freedom
## Residual deviance: 244815  on 571  degrees of freedom
##   (30 observations deleted due to missingness)
## AIC: 5103
## 
## Number of Fisher Scoring iterations: 2
```

A. Buffet of models recap

The `family` argument to `glm()` helps you change the kind of model being estimated.

```
d <- d %>%
  mutate(Points_Earned_Bin = ifelse(d$Points_Earned > mean(d$Points_Earned, na.rm = T), 1, 0))

glm1 <- glm(Points_Earned_Bin ~ TimeSpent_hours, data = d, family = "binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = Points_Earned_Bin ~ TimeSpent_hours, family = "binomial",
##     data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6393  -0.6166  -0.5992  -0.5624   2.0446
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -1.484105   0.199288  -7.447 9.55e-14 ***
## TimeSpent_hours -0.004482   0.005535  -0.810    0.418
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 455.26  on 506  degrees of freedom
## Residual deviance: 454.58  on 505  degrees of freedom
##   (96 observations deleted due to missingness)
## AIC: 458.58
##
## Number of Fisher Scoring iterations: 4
```

B. Model outputs and summaries

We've previously discussed making better looking output tables in .Rmd documents

```
library(sjPlot)

tab_model(m1)
```

	Final Grade CEMS		
Predictors	*Estimates*	*CI*	*p*
(Intercept)	65.81	62.88 – 68.74	**<0.001**
TimeSpent_hours	0.36	0.29 – 0.44	**<0.001**
Observations	573		
R^2 / R^2 adjusted	0.134 / 0.132		

B. Model outputs and summaries

Another package makes it easier to pull out model estimates in an easier format to use for further operations: `broom`

`tidy()` Makes `lm()` coefficient output into a tidy data frame format, which you can then use all of your tidyverse tools on

`glance()` works similarly but with the whole model diagnostic statistics such as R^2

```
library(broom)

# Model coefficents
tidy(m1)
```

```
## # A tibble: 2 x 5
##   term            estimate std.error statistic   p.value
##   <chr>              <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)       65.8      1.49       44.1  3.71e-186
## 2 TimeSpent_hours    0.365    0.0389      9.38 1.53e- 19
```

```
# whole model stats
glance(m1)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.134         0.132  20.7      88.0 1.53e-19     1 -2548. 5103. 5116.
## # … with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

B. Model outputs and summaries

`broom` can also help you grab other diagnostics from the model with `augment()`

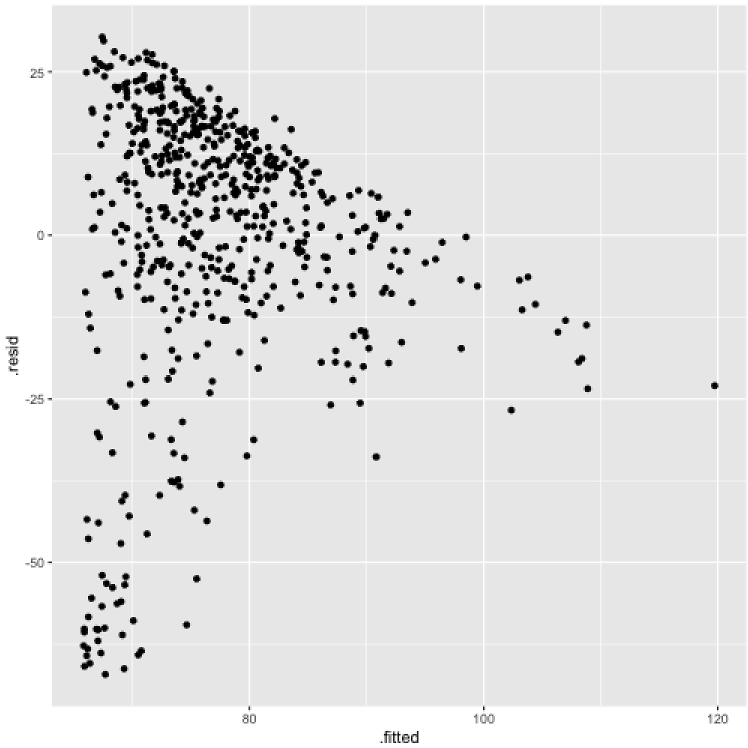
```r
# Data level additional model generated values
head(augment(m1))
```

```
## # A tibble: 6 x 9
##   .rownames FinalGradeCEMS TimeSpent_hours .fitted .resid    .hat .sigma .cooksd
##   <chr>              <dbl>           <dbl>   <dbl>  <dbl>   <dbl>  <dbl>   <dbl>
## 1 1                   93.5            25.9    75.3  18.2  0.00184   20.7 7.14e-4
## 2 2                   81.7            23.0    74.2   7.49 0.00198   20.7 1.30e-4
## 3 3                   88.5            14.3    71.0  17.4  0.00275   20.7 9.82e-4
## 4 4                   81.9            26.6    75.5   6.32 0.00182   20.7 8.52e-5
## 5 5                   84              24.7    74.8   9.18 0.00190   20.7 1.87e-4
## 6 7                   83.6            22.0    73.8   9.74 0.00204   20.7 2.27e-4
## # … with 1 more variable: .std.resid <dbl>
```

B. Model outputs and summaries

`broom` can also help you grab other diagnostics from the model

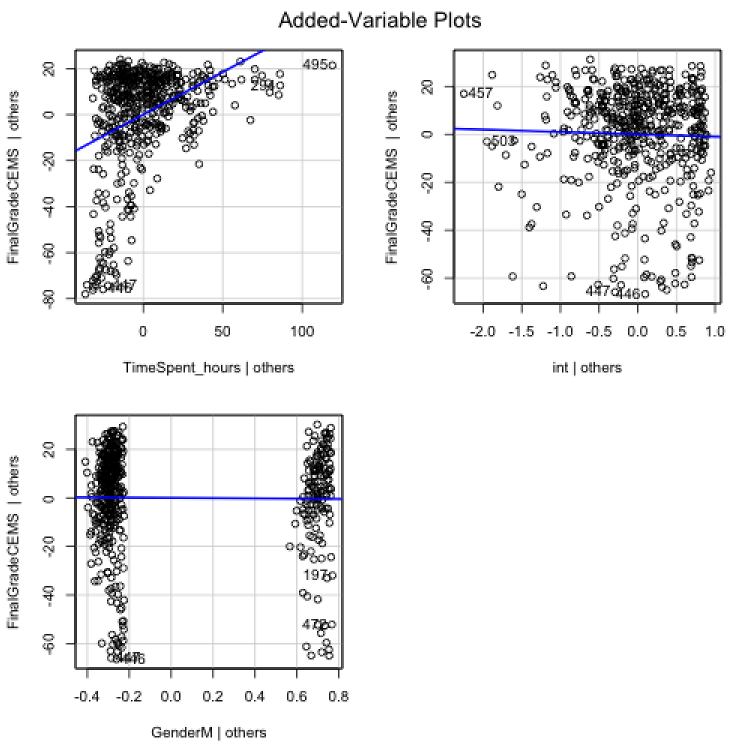
```
f <- augment(m1)

ggplot(f) +
  geom_point(aes(x=.fitted, y=.resid))
```



B. Model outputs and summaries

The `car` package has plotting and testing functions that can help you evaluate models

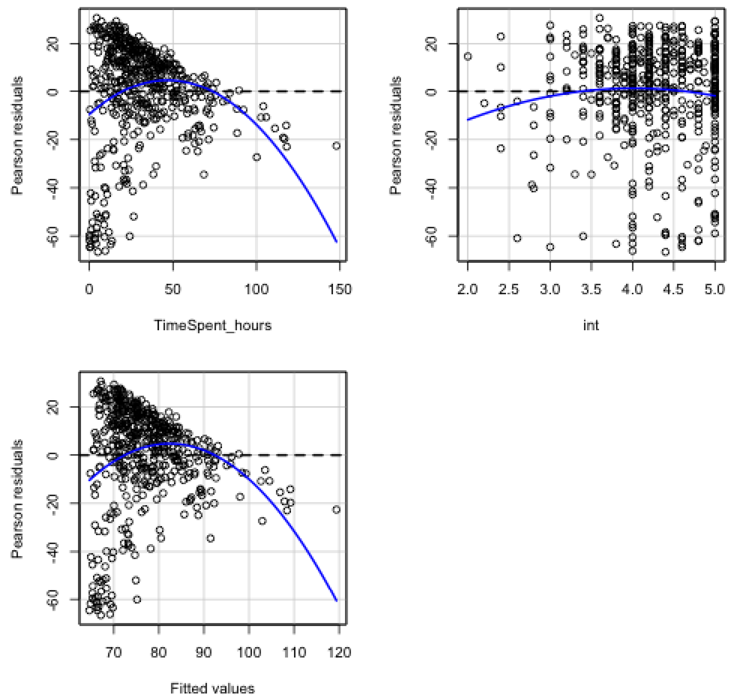
```
library(car)

avPlots(m2)
```



B. Model outputs and summaries

The `car` package has plotting and testing functions that can help you evaluate models

```
residualPlots(m2)
```




```
##                 Test stat Pr(>|Test stat|)
## TimeSpent_hours   -6.3056        6.337e-10 ***
## int               -1.7279          0.08462 .
## Tukey test        -6.3520        2.125e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

B. Model outputs and summaries

The `report` package can generate written summaries of the results of your models

```
#devtools::install_github("https://github.com/easystats/report")
library(report)

report(m1)
```

```
## We fitted a linear model (estimated using OLS) to predict FinalGradeCEMS with TimeSpent_hours (formula: F
##
##   - The effect of TimeSpent_hours is statistically significant and positive (beta = 0.36, 95% CI [0.29, 0
##
## Standardized parameters were obtained by fitting the model on a standardized version of the dataset.
```

B. Model outputs and summaries

Components of an **`lm()`** object, some of which you can get with helper functions

```
m1$coefficients
```

```
##     (Intercept) TimeSpent_hours
##      65.8085094       0.3648365
```

```
head(m1$residuals)
```

```
##         1         2         3         4         5         7
## 18.188854  7.485674 17.447115  6.323524  9.181244  9.742307
```

C. Model helpers – tests, diagnostics, etc.

There are a number of helper functions that work with `lm` and other models

```
# model coefficients
coef(m1)
```

```
##     (Intercept) TimeSpent_hours
##      65.8085094       0.3648365
```

```
# model residuals
head(residuals(m1))
```

```
##         1         2         3         4         5         7
## 18.188854  7.485674 17.447115  6.323524  9.181244  9.742307
```

```
# regression fitted values
head(fitted(m1))
```

```
##        1        2        3        4        5        7
## 75.26487 74.21617 71.04047 75.52907 74.81876 73.84596
```

```
# Get the data used to fit the model
head(model.frame(m1))
```

```
##   FinalGradeCEMS TimeSpent_hours
## 1       93.45372        25.91944
## 2       81.70184        23.04500
## 3       88.48758        14.34056
## 4       81.85260        26.64361
## 5       84.00000        24.69667
## 7       83.58827        22.03027
```

C. Model helpers – tests, diagnostics, etc.

There are a number of helper functions that work with `lm` and other models – How do we know which ones?

```
library(sloop)
s3_methods_class("lm")
```

```
## # A tibble: 85 x 4
##    generic       class visible source
##    <chr>         <chr> <lgl>   <chr>
##  1 add1          lm    FALSE   registered S3method
##  2 alias         lm    FALSE   registered S3method
##  3 anova         lm    FALSE   registered S3method
##  4 Anova         lm    FALSE   registered S3method
##  5 as.data.frame lm    FALSE   registered S3method
##  6 augment       lm    FALSE   registered S3method
##  7 avPlot        lm    FALSE   registered S3method
##  8 Boot          lm    FALSE   registered S3method
##  9 bootCase      lm    FALSE   registered S3method
## 10 boxCox        lm    FALSE   registered S3method
## # … with 75 more rows
```

```
confint(m1)
```

```
##                      2.5 %    97.5 %
## (Intercept)     62.8796038 68.737415
## TimeSpent_hours  0.2884451  0.441228
```

```
vcov(m1)
```

```
##                 (Intercept) TimeSpent_hours
## (Intercept)      2.22367608    -0.047242592
## TimeSpent_hours -0.04724259     0.001512691
```

C. Model helpers – tests, diagnostics, etc.

The `lmSupport` package provides number of additional tools for regression models

```
library(lmSupport)

m1 <- lm(FinalGradeCEMS ~ TimeSpent_hours, data = d)
m2 <- lm(FinalGradeCEMS ~ TimeSpent_hours + int + Gender, data = d)

# A function to check regression model assumptions
modelAssumptions(m1)
```

```
## Descriptive Statistics for Studentized Residuals
##
## Call:
## lm(formula = FinalGradeCEMS ~ TimeSpent_hours, data = d)
##
## Coefficients:
##     (Intercept)  TimeSpent_hours
##         65.8085           0.3648
##
##
## ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
## USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
## Level of Significance =  0.05
##
## Call:
##  gvlma(x = Model)
##
##                     Value   p-value                   Decision
## Global Stat        313.26 0.000e+00 Assumptions NOT satisfied!
## Skewness           182.13 0.000e+00 Assumptions NOT satisfied!
## Kurtosis            69.89 1.110e-16 Assumptions NOT satisfied!
## Link Function       40.33 2.143e-10 Assumptions NOT satisfied!
## Heteroscedasticity  20.91 4.820e-06 Assumptions NOT satisfied!
```

C. Model helpers – tests, diagnostics, etc.

The `lmSupport` package provides number of additional tools for regression models

```
m2 <- lm(FinalGradeCEMS ~ TimeSpent_hours + int + Gender, data = d)
m1 <- lm(FinalGradeCEMS ~ TimeSpent_hours, data = model.frame(m2$model))

# F Test difference between 2 models
modelCompare(m1, m2)
```

```
## SSE (Compact) =  220820.1
## SSE (Augmented) =  220624.9
## Delta R-Squared =  0.0007632935
## Partial Eta-Squared (PRE) =  0.0008841496
## F(2,499) = 0.2207905, p = 0.8019629
```

Addendum: Other modeling packages to be aware of

- `lme4` and `nlme`: hierarchical linear models (aka multilevel models)
- `lavaan`: structural equation models
- `MASS`: Robust regression
- `caret`: Lots of different classification and regression models for machine learning applications
- `parsnip`: (tidyverse) Unified interface to many different models, largely machine learning type
- Time series: https://cran.r-project.org/web/views/TimeSeries.html

Other types of models that may be more applicable to your field can also be found in the CRAN task views: https://cran.r-project.org/web/views/

Tutorials for many different varieties of Regression can be found here: https://stats.idre.ucla.edu/other/dae/

Other packages that can help you with your models https://easystats.github.io/easystats/ e.g. the `performance` package does model assumption checking and model comparison

Live coding

Let's head over to the following file for a demonstration: `week-11-demo.R`

Homework 10

Fit a machine learning model to a set of educational log–trace data

Using the `caret` package:

- Partition the data into training and testing sets
- Pre–process the data: identify linear dependencies, scale the variables and impute missing values
- Train the model using a Random Forest, Support Vector Machine, or Neural Network model (defend your choice)
- Choose bootstrap or k–fold cross validation for your model fitting (defend your choice)
- Evaluate model performance with calibration curves


BEWARE OF
APRIL FOOLS
JOKES
#TWODOTS

Logistics

This week

- Homework 10: Available tomorrow by noon tomorrow; **Due by Thursday, 4/8**
- Reading: * https://www.tmwr.org/base-r.html#formula

Assignment updates

- Final project
  - We will provide feedback to you within the next week (before our next class on Thursday, April 8)
  - We recommending getting started!
  - We recommend having starting in earnest within the next two–three weeks (depending on your need for feedback)
- Curating a data science resource

Wrapping up

In your base group's Slack channel:

- What is one thing you learned today?
- What is something you want to learn more about?
- Share your feelings in GIF form!