Teaching Stats for Data Science

Daniel Kaplan, Macalester College, `kaplan@macalester.edu`

The familiar mathematical topics of introductory statistics — means, proportions, t-tests, normal and t distributions, chi-squared, etc. — are a product of the first half of the 20th century. Naturally, they reflect the statistical conditions of that era: scarce, e.g. $n < 10$, data originating in benchtop or agricultural experiments; algorithms communicated via algebraic formulas. Today, applied statistics relates to a different environment: software is the means of algorithmic communication, observational and "unplanned" data are interpreted for causal relationships, and data are large both in n and the number of variables. This change in situation calls for a thorough rethinking of the topics in and approach to statistics education. (Cobb 2015)

In this paper, I present a set of ten organizing blocks for intro stats that, I claim, are better suited to today's environment.

1. Data tables
2. Data graphics
3. Model functions
4. Model training
5. Effect size and covariates
6. Displays of distributions
7. Bootstrap replication
8. Prediction error
9. Comparing models
10. Generalization and causality

The extent of each theme, and the division of the whole into ten blocks, has been made with an eye toward covering each in roughly one week of a standard course. Keeping in mind that one of the drivers of change in statistics has been the transition from algebra to software as the mode of describing algorithms, computing should be integrated thoroughly across blocks. I'll illustrate the sorts of algorithms that support the themes using two new packages for R: `ggformula` and `mosaicModel`. These packages integrate data-science software such as the `tidyverse` (Wickham and Grolemund 2016) with the pedagogical approach of the `mosaic` package (Pruim, Kaplan, and Horton 2017).

The discussion and computer commands in this paper are oriented toward instructors. An implementation of the blocks for students should take another form, one with much more basic and expository narrative and interactive scaffolding for computer commands.

Block 1: Data tables

Start with data tables: the standard row-and-column, case-and-variable organization. Important concepts to cover are:

- tidy data (Wickham and Grolemund 2016, Broman and Woo (2017))
- the physical meaning of a unit of observation, e.g. a person, a person at a medical checkup, etc.
- the distinction between quantitative and categorical variables
- the difference between a data table and the "presentation" of the information.

Do not assume that the concept of a data table is so obvious that it doesn't need to be taught. For instance, consider Figure 1, a problem from an open-source statistics textbook.(Diez, Barr, and Cetinkaya-Rundel 2014)

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

1.1 Migraine and acupuncture. A migraine is a particularly painful type of headache, which patients sometimes wish to treat with acupuncture. To determine whether acupuncture relieves migraine pain, researchers conducted a randomized controlled study where 89 females diagnosed with migraine headaches were randomly assigned to one of two groups: treatment or control. 43 patients in the treatment group received acupuncture that is specifically designed to treat migraines. 46 patients in the control group received placebo acupuncture (needle insertion at nonacupoint locations). 24 hours after patients received acupuncture, they were asked if they were pain free. Results are summarized in the contingency table below.[47]

		Pain free		
		Yes	No	Total
Group	Treatment	10	33	43
	Control	2	44	46
	Total	12	77	89

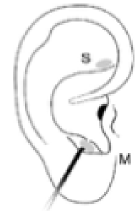


Figure from the original paper displaying the appropriate area (M) versus the inappropriate area (S) used in the treatment of migraine attacks.

Figure 1: *A conventional textbook problem that uses a cross-tabulation of data rather than a tidy data table.*

A stickler might argue that the table in Figure 1 can be construed as a data table, but for our purpose consider a pragmatic approach to defining what constitutes the sorts of data used in data science. In particular, focus on data that contain many rows and are available in machine-readable form. Or, stated another way, we'll work with data *before* they have been aggregated into the sort of presentation seen in Figure 1.

How to see that the table in Figure 1 is not tidy data? Try to answer these questions: What is the unit of observation? What are the variables? Are the variables quantitative or categorical?

It's easy to imagine what the disaggregated data table that underlies this presentation might look like: perhaps this one where the unit of observation is a person:

patient	accupuncture	pain	date	technician
A2322	control	yes	2014-03-15	Audrey
A2397	treatment	yes	2014-03-17	Audrey
A3213	treatment	no	2014-03-17	Bill
B8732	treatment	no	2014-03-18	Audrey
C6920	control	yes	2014-03-18	Bill
⋮	⋮	⋮	⋮	⋮

Figure 2: *The underlying data table from Figure 1 might have looked like this.*

By learning the tools to work with data tables, you can easily create presentations like the cross-tabulation in Figure 1. But you also have the ability to explore other possible explanations for the variation in pain, such as the effectiveness of the technician or the day of the week.

Block 2: Data Graphics

The crucial goal for data science is to transform data into information suited for human consumption. Visualization provides one of the most readable and compelling forms of information as well as a highly motivating early experience for beginning students.

This block covers the display of relationships among variables. Pruim et al. (2017) argue that it's best to put off displays of a *single* variable until after introducing notation for two or more variables. Accordingly, the topic of distributions of a single variable will be deferred until a later block.

The proper notation and terminology is important to describing and creating graphics. I've found the "Grammar of Graphics" an effective way to start. The well-regarded `ggplot2` package (Wickham 2009) is standard implementation of the grammar of graphics. Unfortunately, the notation can be difficult for

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

beginners and is not closely connected with other notation for statistical calculation. Here, we will use `ggformula` (D. Kaplan and Pruim 2017) which provides a formula-based interface to `ggplot2`.

Many of the graphical displays covered in introductory statistics date from a time when graphics were made by putting pen to paper. Modern computer graphics can include aspects such as color, opacity, facets and jittering. Figure 3 uses faceting and opacity to display height versus age and sex for the people included in the National Center for Health Statistics data. (These are available as `NHANES` in the `NHANES` package.)

```
NHANES %>%
  gf_point(Height ~ Age | Gender, alpha = 0.1)
```

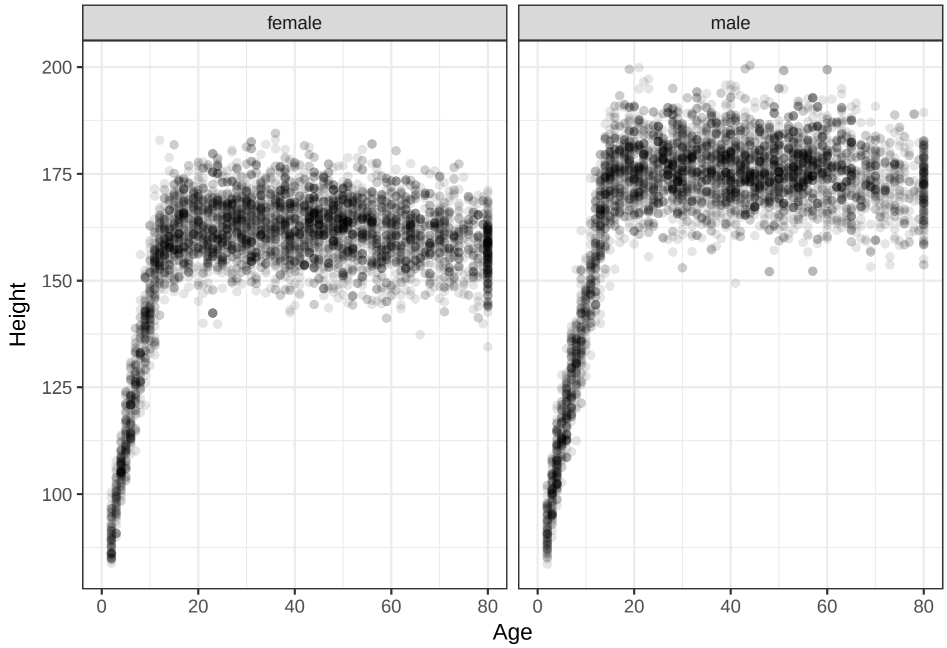


Figure 3: *Relationships among height, age, and sex in the* `NHANES` *data.*

The commands making Figure 3 are simple, yet there are important display choices being made: the mode of graphics, the graphical roles of the different variables, use of color and faceting. The `ggformula`/`ggplot2` framework used for graphics can scale nicely to handle more than the three variables in Figure 3. As such, graphics like these reflect the new emphases on statistical thinking advocated by the GAISE report.(2016)

a. *Teach statistics as an investigative process of problem-solving and decision-making.*
b. *Give students experience with multivariable thinking.*

If students are to engage with a process of decision-making, there need to be decisions that they can make. Starting with the graphics commands here, there are several questions that students are in a position to answer through experiment. Is color/fill contributing to the graphic? Are the right variables being used for the axes and facets? How do such decisions affect the "story" told by the graphics? How could the height vs age graph be improved to highlight the differences between the sexes in children? (Hint: Try interposing the wrangling command `filter(Age < 18)` before the `gf_point()` command.)

Block 3: Model functions

Statistics helps us describe relationships between different variables. This block introduces the notion of representing relationships by mathematical functions that have one or more variables as input and one variable as output. These are often called the *explanatory* variables and the *response* variable respectively.

High-school mathematics classes study, almost exclusively, functions of only one input variable. Models of real-world phenomena are more complicated, involving two or more input variables. I submit that it is a

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

good practice to start with functions of two variables. That way it's easy to generalize to more inputs or to simplify to a single (or no) inputs.

The next block deals with training models. But for the student, all that's needed at this stage is the function that relates the explanatory inputs — `Gender` and `Age` here — to the response variable. Let's call that function, `height()`. The following commands show how `height()` is being extracted from a statistical model using `mod_fun()`, but imagine that it is the `height()` function, rather than the training command, that is being provided for the students.

```
hmod1 <- lm(Height ~ Gender * ns(Age, 5), data = NHANES) #training
height <- mod_fun(hmod1) # pull out the function from the model
```

There are two inputs to the `height()` function: `Gender` and `Age`. One of the main things one does with functions is to evaluate them: supplying inputs and receiving an output, e.g.

```
height(Age = 25, Gender = "female")
```

```
##   Gender Age model_output
## 1 female  25        164.8
```

In this format, the model output is shown along with the corresponding inputs. This is useful if you want to examine the output for several different values of the input, e.g.

```
height(Age = 11:15, Gender = "female")
```

```
##   Gender Age model_output
## 1 female  11        147.3
## 2 female  12        151.5
## 3 female  13        155.3
## 4 female  14        158.4
## 5 female  15        161.0
```

Another important way to present models is graphically. For this, supply the `height()` function with a range of inputs, creating a data table. This data table can be graphed with the usual tools.

```
height(Age = 3:80, Gender = c("female", "male")) %>%
  gf_line(model_output ~ Age | Gender)
```


PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

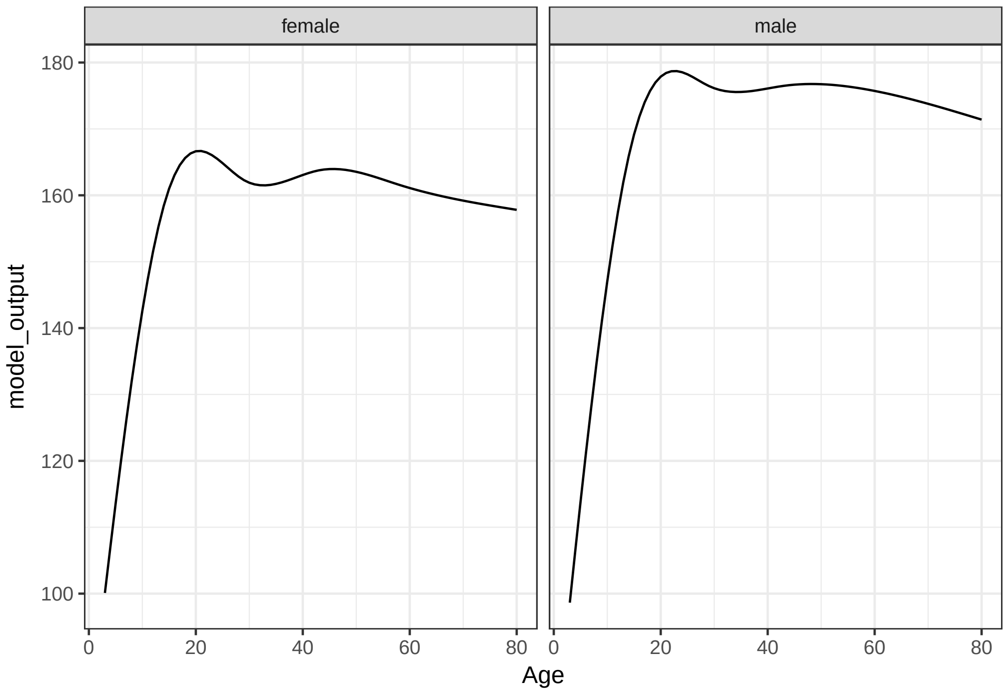


Figure 4: *A model of height as a function of* `Age` *and* `Gender`*.*

Reading such graphs is not trivial for newcomers to statistics. It's advisable to take time to help students learn what to look for. Note the connections to previous blocks: the function creates a data table; choices about the variables need to be made, e.g. should one use faceting or color to represent `Gender`? There are different ways one might present the model that are suited to different purposes. For example, Figure 5 is another graphical depiction of the same model that highlights how the relationship between women's and men's heights varies across ages:

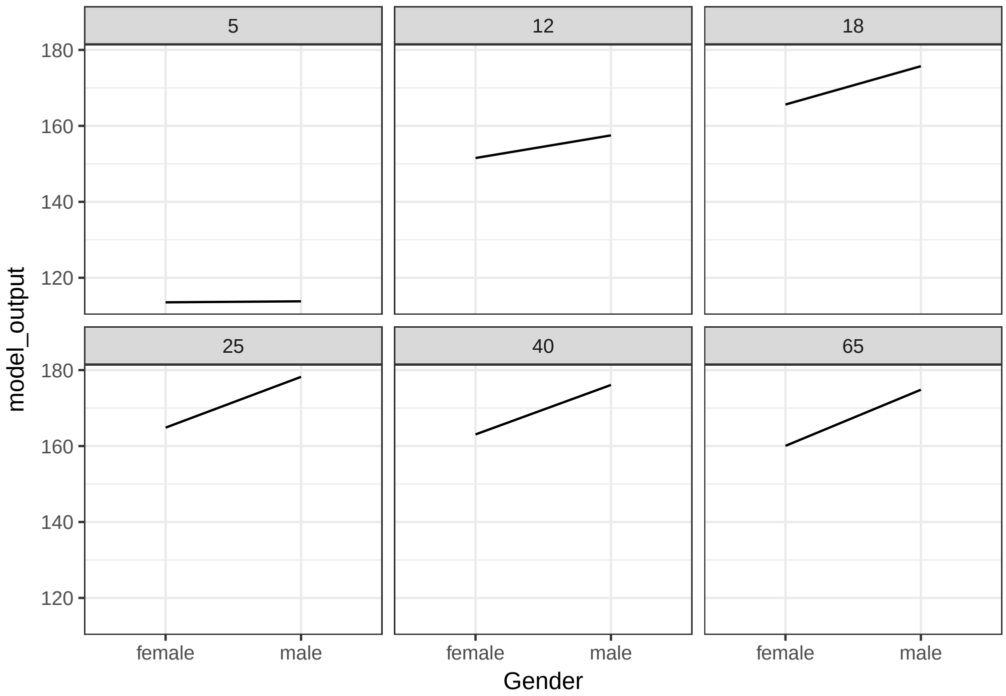


Figure 5: *Another depiction of the* `height()` *function.*

See Block 5 for an example of a classifier function, that is, a function whose output is categorical.

Block 4: Model training

Functions such as those shown in the previous section need to be constructed before they can be used. The verb "fit" is used to identify this process, as in choosing the right size clothing. Extend the metaphor to include a variety of different types of clothing — clothing for formal wear versus clothing for sports. The different types of clothing are made of different materials to serve different purposes.

In modeling, there are different types or *architectures* of model functions for different purposes. It's helpful to cover at least two architectures a regression model and a classifier. The modern computational environment gives considerable freedom here.

For the purposes of discussion, consider these three model architectures.

- Linear regression such as provided in R by `lm()`.
- Recursive partitioning, for instance `rpart()` in the `rpart` package.
- Logistic regression, as with `glm(..., family = "binomial")`

This set of three architectures satisfies a couple of important desiderata for use in teaching:

1. The calling syntax for the architectures are highly similar. `model_architecture`(`response_var`~`explanatory_vars`, data = `data_table`)‘
2. The explanatory variables can be a mix of quantitative or categorical.

The student's choice of architecture is largely set by the kind of response variable. `lm()` is suited to quantitative response variables, `rpart()` to either quantitative or categorical, and `glm()` to response variables with only two levels.

The functions of height vs age (and other variables) shown in Figure 5 were trained with `lm()`. Nonetheless, they are not "straight-line" functions but follow the data in a natural way. This points to a third desideratum

3. Model architectures should provide a means to generate functions that can flexibly follow the data.

For `lm()`, such flexibility can be introduced by wrapping terms with `ns()` and connecting terms with `*` rather than `+`. I do not want to suggest that the mathematics behind natural splines or interactions ought to be included in a first course, just that by using these techniques a natural-looking function becomes a choice available to the modeler. So think of `ns()` as standing for "not straight," with the parameter indicating the extent of flexibility.

In the days when statistical algorithms were communicated by algebraic formulas, it made sense to start with "simple regression." Today, simple regression is not any simpler than other kinds of model building. (Gould 2017) In hand-drawing a function through the data in Figure 3, it is unnatural to prefer a straight-line function to a function that can bend with the data. So avoid the suggestion that statistical models are always stiff; the question of how much bending is appropriate can itself become an opportunity for teaching statistical thinking.

Block 5: Effect size and covariates

One important use for statistical models is to describe how one variable depends on another. It's unnecessary to talk about "coefficients" in an introductory course. Instead, perform simple experiments on models: changing an input and observing how the output changes. For instance, to see the effect of one year's age on the growth of young children, evaluate the model at ages 5 and 6 and compare the outputs.

Consider `hmod1`, the model of height versus age and sex displayed in Figures 5 and 6. Suppose we want to find out how children's height changes with age. Simply evaluate the model at two different ages, say:

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

```
mod_eval(hmod1, Age = c(5, 6))
```

```
##   Gender Age model_output
## 1 female   5        113.5
## 2   male   5        113.8
## 3 female   6        119.9
## 4   male   6        121.1
```

Evidently, both boys and girls grow by about 7 cm/year.

Once students are introduced to interrogating a model in this way, the operation can be streamlined:

```
mod_effect(hmod1, ~ Age, Age = 5, step = 1, Gender = c("female", "male"))
```

```
##   slope Age to_Age Gender
## 1 6.414   5      6 female
## 2 7.265   5      6   male
```

The argument `~ Age` specifies which variable to use when looking at the effect size. Being able to calculate effect sizes easily supports a nuanced discussion of covariates.

Effect sizes are an appropriate way of describing classifier models, that is, models whose output is a category rather than a number. For the effect size, classifiers are configured to return a *probability* of each possible output.

Block 3 mentioned classifiers, a model function with a categorical output. To illustrate, consider data on smoking and mortality, specifically the `Whickham` data provided through the `mosaic` package. (Kaplan 2011, Pruim, Kaplan, and Horton (2017), Committee (2016)). The Whickham study involved interviews with registered voters 1972-1974 with follow-up twenty years later. Some of the interviewees had died in the interim, providing data to assess the association between smoking and mortality.

In traditional introductory stats, probabilities are usually a matter of counting and cross-tabulation, not part of a modeling framework. For instance, here is a tabulation of conditional probabilities of mortality `outcome` given smoking status. Smoking is associated with a reduction in mortality from 31.4% to 23.9%, that is, a decrease of 7.5 percentage points.

```
Whickham %>%
  df_props(outcome ~ smoker, wide = TRUE)
```

```
##   outcome smoker_No smoker_Yes
## 1   Alive    0.6858     0.7612
## 2    Dead    0.3142     0.2388
```

Such cross-tabulations have their place, but they do not generalize well to including covariates. In this example, smoking is misleadingly associated with *lower* mortality. This is the result of failure to adjust for a covariate: age.

Using a modeling framework empowers students to consider covariates. The following command trains a model that recapitulates the information in the cross-tabulation. Linear regression would do, but here I'll use logistic regression with an eye toward eventually adding a covariate.

```
wmod1 <- glm(outcome == "Dead" ~ smoker,
             data = Whickham, family = "binomial")
mod_eval(wmod1)
```

```
##   smoker model_output
## 1     No       0.3142
## 2    Yes       0.2388
```


PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

```
mod_effect(wmod1, ~ smoker, age = c(40, 50, 60))
```

```
##     change smoker to_smoker age
## 1 -0.07538     No       Yes  40
## 2 -0.07538     No       Yes  50
## 3 -0.07538     No       Yes  60
```

As before, smoking is associated with *lower* mortality by about 7 percentage points … unless age is included as a covariate. This is a matter of using `age` as an explanatory variable:

```
wmod2 <- glm(outcome == "Dead" ~ smoker + age,
             data = Whickham, family = "binomial")
mod_effect(wmod2, ~ smoker, age = c(40, 50, 60))
```

```
##    change smoker to_smoker age
## 1 0.01377     No       Yes  40
## 2 0.03420     No       Yes  50
## 3 0.05106     No       Yes  60
```

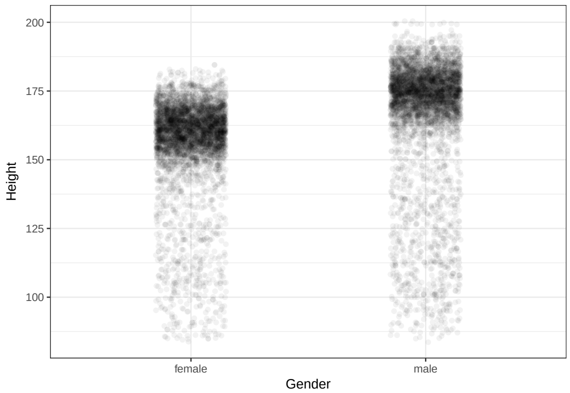
Even if the primary interest is in the effect of smoking, age is a relevant variable to include. Appendix B of the GAISE report (2016) uses the Whickham example to illustrate how to "account for the possible impact … confounding variables." Their starting point is a course cross-tabulation splitting age into two categories: 18-65 and 65+. Statistics for data science should build results from tidy data, not cross-tabulations. And who really thinks an 18-year old and a 65-year old are to be regarded as the same when it comes to 20-year mortality?

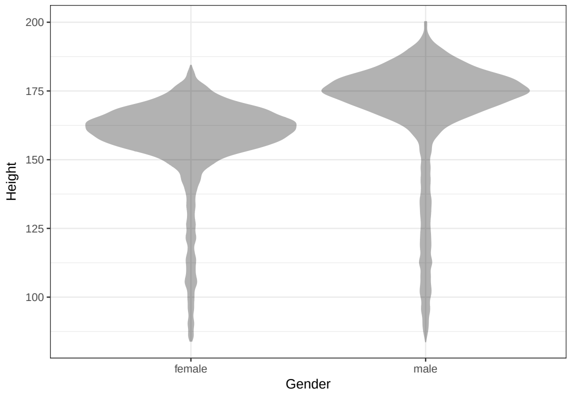
Block 6: Displays of distributions

Early in traditional statistics courses, students are introduced to ways of displaying and quantifying variation. I've deferred the topic to Block 6 because there are more compelling motivating topics to begin with. But we are now at the point where we need to consider random variation.

Traditional courses cover this is considerable depth, going in to shapes of distributions, presentations such as histograms, box-and-whisker plots, calculatations such as standard deviations, inter-quartile intervals, the five-number summary, etc. It's unclear to me how much of this is really useful for understanding data and how much is a set-up for later traditional techniques that build on standard deviations and the central limit theorem.

Some of these traditional topics are clearly oriented to doing calculations by hand, as opposed to generating insight into data. For instance histograms and box-and-whisker plots may not be as useful as density plots. To illustrate, Figure 6 presents the distribution of heights across genders in the `NHANES` data.





PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

Figure 6: *Displaying density using transparency and jittering (left). The violin plot (right) gives a more compact display.*

These displays are meant to help see patterns or tell a story. For instance, the story here might be that men tend to be somewhat taller than women, but that many women are taller than many men.

Later in the course we will want to quantify precision and investigate whether adding features to a model improves it. For these purposes, the coverage interval can provide a compact numerical description. (See, e.g. (Cumming and Finch 2005))

First, create a function to compute the coverage interval at the desired level. Then use

```
cover95 <- coverage(0.95)
NHANES %>%
  df_stats(Height ~ Gender, cover95)
```

```
## Warning: Excluding 353 rows due to missing data [df_stats()].
```

```
##   Gender lower upper
## 1 female 102.4 176.1
## 2   male 100.9 190.3
```

Block 7: Bootstrap replications

A bright spot in statistics education has been the introduction to the mainstream of simulation and re-sampling approaches to statistical inference. (See e.g. (R. H. Lock et al. 2012).) The concise commands provided by the `mosaic` R package (Pruim, Kaplan, and Horton 2017) provides commands such as `do()` and `resample()` that make bootstrapping concise.

The `mosaicModel` package creates even stronger connections of bootstrapping to model analysis. The primary interface is to create bootstrap ensembles of a model. These ensembles can then be used with model-analysis functions such as `mod_effect()`. To illustrate, in the following, the `mod_ensemble()` function is being used to create an ensemble of 100 for the model of height introduced earlier. The `mod_effect()` function, when applied to the ensemble, computes the effect size for each member of the ensemble.

```
hmod_ensemble <- mod_ensemble(hmod1, nreps = 100)
Trials <-
  mod_effect(hmod_ensemble, ~ Age,
             Age = 5, step = 1,
             Gender = c("male", "female"))
head(Trials, 4)
```

```
##   slope Age to_Age Gender .trial
## 1 7.273   5      6   male      1
## 2 6.504   5      6 female      1
## 3 7.307   5      6   male      2
## 4 6.460   5      6 female      2
```

The 95% confidence interval on the slope is the 95% coverage interval of the trials:

```
Trials %>%
  df_stats(slope ~ Gender, cover95)
```

```
##   Gender lower upper
## 1 female 6.277 6.561
## 2   male 7.169 7.388
```

Here you can see that the confidence intervals do not overlap. That guarantees significance at $p < 0.05$ and is a good indication that $p < 0.01$. In some educational contexts, instructors might want to introduce formal

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

tests (such as the t-test) for computing a p-value. Given the recent statement from the American Statistical Association encouraging use of confidence intervals and de-emphasizing p-values (Wasserstein and Lazar 2016), reasoning using confidence intervals may be preferred. (See (Cumming and Finch 2005).)

Block 8: Prediction error

Prediction error refers to the discrepancy between the model output for a given set of inputs and the actual value of the response variable for those inputs. To calculate the prediction error, evaluate the model and consider the squares of the differences between response and output, like this:

```
mod_eval(hmod1, data = NHANES) %>%
  mutate(sq_error = (model_output - Height)^2) %>%
  df_stats( ~ sq_error, mean)
```

```
## Warning: Excluding 353 rows due to missing data [df_stats()].
```

```
##   mean_sq_error
## 1         52.15
```

Once the concept has been introduced, it's helpful to condense the operation with a purpose-specific function:

```
mod_error(hmod1, testdata = NHANES)
```

```
## [1] 52.15
```

Error measured in this way on the same data used for training the model can be misleadingly optimistic. Fixing that over-optimism is the purpose of the next section.

Block 9: Comparing models

Cross-validation refers to a process of withholding a fraction of the cases when training, then evaluating the prediction error using the withheld cases. The withheld cases are called "test data," whereas the cases used for training are the "training data." The prediction error using the same training data in two roles, for training and for testing, is called the "in-sample" prediction error. The "out-of-sample" prediction error refers to using the training data to train the model and the test data to find prediction errors.

A good place to start in intro stats is to divide a dataset up into training and testing components, train the model with the training data, then compare the in-sample and out-of-sample prediction error. The following commands do this in a rather verbose way, suitable for a demonstration that in-sample prediction error is biased to be low.

```
Shuffled <- NHANES %>% sample_n(size = 1000)
Training_data <- head(Shuffled, 500)
Testing_data  <- tail(Shuffled, 500)
trained_model <-
  lm(Height ~ Gender * ns(Age,5), data = Training_data)
```

```
mod_error(trained_model, testdata = Training_data) # in sample
```

```
## [1] 54.37
```

```
mod_error(trained_model, testdata = Testing_data)  # out of sample
```

```
## [1] 57.18
```

This is one random trial of splitting into training and testing data. The specific prediction error values will vary from trial to trial. Thus it is advisible to run several trials.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

The `mod_cv()` function carries out several random trials and reports the results. By using k-fold cross-validation, the entire data set can be used in training the model, as with `hmod1` from Block 3.

```
mod_cv(hmod1)
```

```
##     mse model
## 1 52.26 hmod1
## 2 52.26 hmod1
## 3 52.23 hmod1
## 4 52.24 hmod1
## 5 52.26 hmod1
```

An important use of cross-validation is to compare two or more models. Consider testing a model `hmod2` with more nonlinearity than `hmod1`:

```
hmod2 <- lm(Height ~ Gender * ns(Age, 25), data = NHANES)
```

`mod_cv()` can take multiple models as arguments. The following command performs 50 cross-validation trials for each model and displays coverage intervals:

```
mod_cv(hmod1, hmod2, ntrials = 50) %>%
  df_stats(mse ~ model, cover95)
```

```
##   model lower upper
## 1 hmod1 52.22 52.30
## 2 hmod2 49.69 49.86
```

No overlap: `hmod2` has a smaller prediction error than `hmod1`.

The prediction error encountered in a traditional statistics course is the in-sample error. Much technical apparatus — degrees of freedom, t distributions, etc. — is applied to correct for the optimistic bias of in-sample error. But in an era when data are plentiful and computation is cheap, there's no reason to muddy the conceptual waters with this apparatus. It's simple enough to find the out-of-sample error and simple and reliable to compare the distributions of a few trials when comparing models.

Block 10: Generalization and causality

What's a representative sample? What are common sources of bias in surveys and experiments? How can one make reasonable inferences about causation?

These are important questions to cover in introductory statistics, and there are many existing resources that approach them at an introductory level. Rather than reiterate those sources, I will suggest two changes in approach.

1. Putting this block at the end of the course allows the instructor to use the techniques introduced earlier to illustrate problems and possible solutions. For instance, bias in sampling can be demonstrated with sampling, as can the use of covariates to improve estimates.
2. Many of the areas of application of data science are based in a concern with elucidating causal influences from observational data. We ought to be teaching students responsible methods to addressing this. See, for instance, the discussions in (Wainer 2015, Pearl, Glymour, and Jewell (2016), and Kaplan (2011)).

Discussion

The term "data science" reflects concepts and techniques useful for drawing conclusions from large and complex data sets. This differs from the traditional conception of intro stats held by many instructors; that the purpose is to introduce differences in means and proportions, the t-test, the χ^2 test, etc. Such tests were developed for posing simple questions about small datasets.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

There is a widespread recognition that the era of data science calls for a radical re-thinking about many aspects of intro stats. (Cobb 2015) For instance, returning to the GAISE (2016) emphasis on decision-making as part of statistical thinking:

> *Teach statistics as an investigative process of problem-solving and decision-making. Students should not leave their introductory statistics course with the mistaken impression that statistics consists of an unrelated collection of formulas and methods. Rather, students should understand that statistics is a problem-solving and decision-making process that is fundamental to scientific inquiry and essential for making sound decisions.*

In the traditional intro stat course, the primary decision to be made are which one of the many named tests to apply. With the ten conceptual blocks described here, students are given considerably more scope for decision making such as the choice of covariates to include in a model and the architecture of the model. The operations to be applied each reflect a different part of the process of statistical thinking: modeling is about constructing a description from data, effect size is about summarizing that description, bootstrapping addresses the issue of precision, and cross-validation informs choices about what explanatory variables or model architectures should be used. Importantly, the techniques, concepts, and vocabulary the students learn are consistent with contemporary statistical practice, relate to key elements of data science such as wrangling and visualization, and provide an introduction to machine learning techniques.

References

Broman, Karl, and Kara Woo. 2017. "Data Organization in Spreadsheets." *The American Statistician* this volume.

Cobb, George. 2015. "Mere Renovation Is Too Little Too Late: We Need to Rethink Our Undergraduate Curriculum from the Ground up." *The American Statistician* 69 (4): 266–82. doi:10.1080/00031305.2015.1093029.

Committee, GAISE ASA Revision. 2016. "Guidelines for Assessment and Instruction in Statistics Education College Report." http://www.amstat.org/education/gaise.

Cumming, Geoff, and Sue Finch. 2005. "Inference by Eye: Confidence Intervals and How to Read Pictures of Data." *American Psychologist*, no. February–March.

Diez, David, Christopher Barr, and Mine Cetinkaya-Rundel. 2014. *Introductory Statistics with Randomization and Simulation.* openintro.org. http://openintro.org.

Gould, Robert. 2017. "Data Literacy Is Statistical Literacy." *Statistics Education Research Journal* 16 (1): 22–25. https://iase-web.org/documents/SERJ/SERJ16(1)_Gould.pdf.

Kaplan, Daniel T. 2011. *Statistical Modeling: A Fresh Approach.* http://project-mosaic-books.com.

Kaplan, Daniel, and Randall Pruim. 2017. "Formula Interface for Ggplot2." Macalester College; Calvin College. https://CRAN.R-project.org/package=ggformula.

Lock, Robin H, Patti Frazer Lock, Kari Lock Morgan, Eric F Lock, and Dennis F Lock. 2012. *Statistics: Unlocking the Power of Data.* Wiley.

Pearl, Judea, Madelyn Glymour, and Nicholas P Jewell. 2016. *Causal Inference in Statistics: A Primer.* Wiley.

Pruim, Randall, Daniel T Kaplan, and Nicholas J Horton. 2017. "The mosaic Package: Helping Students to 'Think with Data' Using R." *The R Journal* 9 (1): 77–102. https://journal.r-project.org/archive/2017/RJ-2017-024/index.html.

Wainer, Howard. 2015. *Truth or Truthiness: Distinguishing Fact from Fiction by Learning to Think Like a Data Scientist.* Cambridge Univ. Press.

Wasserstein, Ronald L., and Nicole A. Lazar. 2016. "The Asa's Statement on P-Values: Context, Process,

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017

and Purpose." *The American Statistician* 70 (2): 129–33. doi:10.1080/00031305.2016.1154108.

Wickham, Hadley. 2009. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

Wickham, Hadley, and Garrett Grolemund. 2016. *R for Data Science*. O'Reilly Media.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3205v1 | CC BY 4.0 Open Access | rec: 29 Aug 2017, publ: 29 Aug 2017