INTRODUCING HIGH SCHOOL STATISTICS TEACHERS TO PREDICTIVE MODELLING AND APIs USING CODE-DRIVEN TOOLS

ANNA FERGUSSON
University of Auckland
a.fergusson@auckland.ac.nz

MAXINE PFANNKUCH
University of Auckland
m.pfannkuch@auckland.ac.nz

ABSTRACT

Tasks for teaching predictive modelling and APIs often require learners to use code-driven tools. Minimal research, however, exists about the design of tasks that support the introduction of high school students and teachers to these new statistical and computational methods. Using a design-based research approach, a web-based task was developed. The task was constructed using our design framework and implemented within a face-to-face professional development workshop involving six high school statistics teachers. The teachers were guided through the process of developing a prediction model using: an informal approach; visual prediction intervals; data about movie ratings from an API; and R code that ran in the browser. Our findings from this exploratory study indicate that the web-based task supported the development of new statistical and computational ideas related to predictive modelling and APIs.

Keywords: Data science education; Predictive modelling; Integrating statistical and computational thinking; Task design; High school teachers; APIs

1. INTRODUCTION

Teaching recommendations for implementing data science at the high school and introductory tertiary level include: placing greater emphasis on predictive modelling (Biehler & Schulte, 2017; Gould, 2017; Ridgway, 2016); immersing students in data-rich contexts by sourcing dynamic ("live") data from the internet (Engel, 2017; Hardin, 2018); and providing opportunities for students to integrate both computational and statistical thinking (e.g., De Veaux et al., 2017; Gould, 2021). An obstacle to implementing these recommendations at the high school level is teacher content knowledge and computing skills, particularly in the areas of machine learning and the use of computer programming (coding) to access, manipulate and visualize data from sources such as APIs (Application Programming Interfaces). While materials for teaching data science at the high school level provide examples of curriculum designs and how computational tools can be used (e.g., mobilizingcs.org/introduction-to-data-science, key2stats.com, prodabi.de, idssp.org), there is a need for the explication of the design principles used to develop the learning tasks.

Clear guidance is also needed for how to design learning tasks that will successfully engage a wide range of high school students with data science, particularly for those who lack confidence with mathematics and computing (Burr et al., 2021). As part of a larger research study, we created a design framework to inform the development of new tasks to introduce code-driven tools for teaching statistical modelling (Fergusson & Pfannkuch, 2021). In this paper, we explore the design and implementation of a task for introducing teachers to predictive modelling using dynamic data sourced from an API.

2. TEACHING PREDICTIVE MODELLING AND APIS

High school statistics courses have traditionally used data collected within formal studies to teach students about study design and statistical inference. Education researchers are re-thinking and

Statistics Education Research Journal, 21(2), Article 8. https://doi.org/10.52041/serj.v21i2.49

expanding their ideas about data and approaches to statistical modelling and have suggested the inclusion of machine learning approaches and associated algorithmic models in high school curricula (e.g., Biehler & Schulte, 2017). From a learning perspective, algorithmic models could offer a more accessible and conceptually simpler mechanism to introduce students to data science than inferential methods (Gould, 2017; Ridgway, 2016), and research by Zieffler et al. (2021) suggested there might be similar benefits for high school statistics teachers. Predictive modelling, with its focus on developing models by learning from features of data to make predictions and forecasts for likely future outcomes, could also provide opportunities for students to integrate both computational and statistical thinking (e.g., De Veaux et al., 2017). Although machine learning approaches to predictive modelling could be used, previous research cautions against using "black box" approaches to teaching modelling (e.g., Biehler & Schulte, 2017; Magana et al., 2011).

Regression models can be used for predictive modelling and simple linear regression is commonly taught at the high school level (e.g., Bargagliotti et al., 2020). However, a different perspective is needed to teach machine learning and algorithmic modelling than the traditional use of linear regression (Biehler & Schulte, 2017). Teachers' existing understanding about linear regression also needs to be considered. The purpose of the linear model can be unclear to teachers, for example, whether the line fitted represents a model for a general relationship or a summary of the data-specific relationship (Casey & Wasserman, 2015). Additionally, little is known about how teachers will reconcile algorithmic modelling approaches alongside traditional modelling approaches within the same teaching programme. Biehler and Schulte (2017) suggested that teaching predictive modelling from a data science perspective could build from familiar understandings of linear regression but include more emphasis on validation through residual analysis and predictive accuracy.

Simple linear regression models are too "simple" to produce high rates of predictive accuracy using point predictions, but more advanced approaches to regression would be beyond the scope of the high school statistics classroom. An informal approach to introducing predictive modelling could draw on the success of informal inference research (e.g., Makar & Rubin, 2018). A key characteristic of using an informal approach is employing visual representations to build concepts and inform decisions, utilizing specially designed software such as *TinkerPlots*™ (Konold & Miller, 2015) and *VIT* (Visual Inference Tools, Wild et al., 2017). An informal approach could be used for introducing predictive modelling, for example by generating prediction intervals based on a visual estimate for the prediction error and evaluating the model in terms of what percentage of the outcomes appear to be "covered" by the prediction intervals.

Students' experiences with training and testing models, such as cross validation, do not need to be formal. Using data from sources such as APIs provides an opportunity for students to train and test models with different data sets, therefore supporting learning about concepts such as overfitting, underfitting, and generalizability. The use of dynamic data more closely aligns with predictive modelling in modern applications, such as monitoring social media usage or customer interactions on web pages and using past customer transactions to predict future purchasing behaviours. Modern data contexts may also be more engaging for teaching high school students (Gould, 2010; Ridgway, 2016).

There are many computational barriers for high school teachers to use APIs for teaching. Providing "data portals" that allow students to access data from APIs without coding (see Erickson, 2020) are one possibility for opening up the world of dynamic data for teaching. However, accessing data from APIs is not the only computational barrier. Using data that is obtained can also raise difficulties. Data formats and hierarchical structures such as JSON (JavaScript Object Notation) and XML (Extensible Markup Language) are not common for teachers, nor are manipulations with the raw data obtained from the API, for example, working with timestamps or text. Recommendations to use interactive documents such as *RMarkdown* (Allaire et al., 2021) to access dynamic data (e.g., Hardin, 2018) are not likely to be widely adopted at the high school level as they involve installing and using specialist software. Teachers may prefer tools that are web-based, free, require minimal time to learn how to use, and offer collaboration and easy sharing of tasks (Biehler, 2018). Therefore, care is needed to design predictive modelling tasks that balance new statistical and computational ideas, with the learning demands of using new data technologies.

A common thread to discussions about data science education is that students need to integrate both statistical and computational thinking (e.g., De Veaux et al., 2017; National Academies of Sciences, Engineering, and Medicine, 2018), and that students need to develop at least some computer

programming skills (e.g., Cetinkaya-Rundel & Rundel, 2018; Nolan & Temple Lang, 2010). Arguments exist for and against teaching statistics at the high school level using *code-driven* tools, computational tools which users interact with predominantly by entering and executing text commands (code). Statistics education research at the high school level, however, has largely involved *GUI*-driven tools, computational tools which users interact with predominantly by pointing, clicking, or gesturing.

With respect to introducing predictive modelling and data sourced from APIs, using a code-driven tool could support the teaching of statistical and computational thinking. Central to statistical thinking is the use of statistical models (Wild & Pfannkuch, 1999). Using a code-driven tool could lower the cognitive demands of the statistical modelling task (Son et al., 2021), as code can be used to articulate modelling steps (Kaplan, 2007). We contend that computational transparency, that is, how obvious the computations performed by the code are to the learner, is an important consideration when using code-driven tools for teaching statistical modelling. By using a code-driven tool, students can modify requests when accessing and using data from APIs, allowing the data context to be central to their learning and decisions (Weiland, 2017; Wild & Pfannkuch, 1999). Furthermore, the use of a coding approach has the potential to allow for exploration of "what if?" scenarios, explorations that are often restricted by the options provided by GUI-driven tools.

The design of the task should take cognizance of the presentation and interface for the computational tool and allow students to tinker with a model articulated with code and to visualize changes instantaneously (cf. *TinkerPlots*™). The *R* (R Core Team, 2020) package *learnr* (Schloerke et al., 2018), for example, provides a way to produce an interactive web-based task where students can execute small "chunks" of *R* code within a web browser. Since the release of *learnr* in 2017, the package has had over 250,000 downloads but there are very few research articles that describe the use of the tool to design learning tasks (e.g., Fergusson & Pfannkuch, 2021; Wiedemann et al., 2020). In general, it is difficult to find substantial literature that explicitly communicates strategies for designing tasks that introduce code-driven tools for statistical modelling at the high school level.

2.1. THE NEW ZEALAND TEACHING CONTEXT

New Zealand has one national curriculum which is taught at nearly all high schools. Grade 12 statistics students are assessed against the curriculum using school-based assessment tasks and national examinations. Curriculum and assessment materials provided by government educational agencies (e.g., New Zealand Qualifications Authority, 2019) were used to review current approaches to the teaching and assessment of predictive modelling. We identified that Grade 12 statistics students are expected to use linear regression models to make predictions but are not required to engage with sample-to-population inference ideas when using linear regression models, for example, interpreting confidence intervals for the model parameters. Students are also not expected to: use a prediction model developed with one set of data to generate predictions for cases within a different set of data; generate prediction intervals from a model; evaluate a model in terms of predictive accuracy; discuss precision versus accuracy; access APIs as a source of data; or use computer programming as part of the predictive modelling process. Consequently, high school statistics teachers are unlikely to have experience with designing and implementing tasks that employ these approaches.

2.2. RESEARCH QUESTION

The purpose of this research is two-fold. The first purpose of the research was to test and refine a design framework that was previously developed. We wanted to ascertain if the design framework could be applied for developing a task that uses a different source of data (dynamic via APIs), a different statistical modelling situation (predictive modelling), and predominantly one type of tool (code-driven). The second purpose of the research was to trial the task with teachers to learn what new statistical and computational ideas and thinking might emerge. The research question is:

> *Given that teachers are familiar with simple linear regression, what new statistical and computational understandings emerge when they are exposed to a new code-driven learning environment that includes dynamic data via APIs and predictive modelling?*

3. RESEARCH APPROACH

New Zealand, like many countries, is considering how to implement data science education at the high school level. As data science approaches are not currently used within the teaching and assessment of statistics at the high school level, a design-based research approach (e.g., Bakker & van Eerde, 2015) was used for the larger research study this paper sits within. A key feature of design-based research is the design and testing of a significant teaching intervention (Anderson & Shattuck, 2012), which for the larger study was the development of new tasks to introduce code-driven tools for teaching statistical modelling from a data science perspective.

Design-based research seeks to simultaneously develop solutions to practical problems grounded in real learning environments alongside new and reusable design principles (Reeves, 2007). Design principles developed from design-based research are theories intended to support other designers to create or observe similar outcomes (Van den Akker, 1999). In alignment with the descriptions provided by Edelson (2002), McKenney and Reeves (2018), and Reeves (2007), the design-based research process used for the larger study involved iterations of four aspects: (1) identifying research problems through the analysis of a practical teaching issues; (2) constructing new learning tasks informed by existing and hypothetical design principles and technological innovations; (3) implementing and refining tasks through teaching experiments; and (4) reflecting and evaluating to produce new design principles and enhanced tasks.

3.1. TASK DESIGN FRAMEWORK

A design framework is a set of design guidelines to use to create a "product" that will support the desired goals for a specific learning context (Edelson, 2002). Through the development of another task within the larger study, we explicated a design framework to construct tasks that introduce learners to statistical modelling using code approaches (Fergusson & Pfannkuch, 2021). Because of the iterative nature of design-based research, we now present our evaluation of the task and design framework developed earlier and the resultant changes to the framework.

The learning task from the first iteration was constructed to move learners from a familiar GUI-driven tool to an unfamiliar code-driven tool for carrying out the randomization test, a familiar statistical modelling approach. Consequently, some of the design principles specifically mentioned GUI-driven tools and assumed existing knowledge of the statistical modelling approach. For the second iteration of the design framework, the following changes were made: the references to GUI-driven tools were replaced with references to statistical modelling ideas or processes; a new design consideration concerning the tools used was added; and the descriptions of the design principles and considerations were modified. The changes were made in order to produce reusable design principles, rather than ones that were too specific to the tools used for the previous task. An overview of the second iteration of the design framework used to construct the task for this paper is shown in Figure 1.

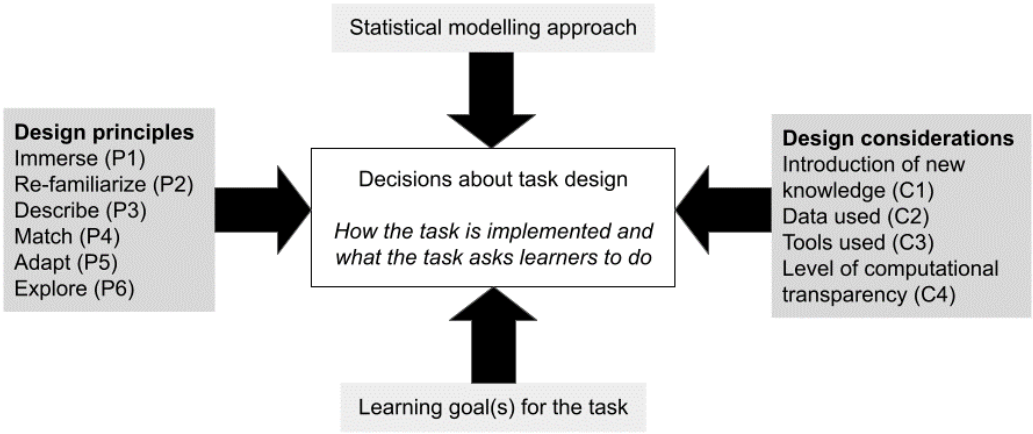


Figure 1. Design framework for constructing statistical modelling tasks that introduce code-driven tools

To use the design framework to construct a task that introduces a code-driven tool for statistical modelling, the task designer needs to decide what statistical modelling approach will be used. There will be an initial idea about the learning goal(s), which are shaped during the design of the task and finalized at the end of the construction process. The design principles are used to inform decisions about features of the learning task in terms of specific actions or experiences for learners and the chronological order of these actions or experiences. The design principles *immerse*, *re-familiarize*, *describe*, *match*, *adapt* and *explore* are presented in Table 1.

Table 1. The six design principles of the design framework

Design principle	Learning action or experience	Anticipated learning
Immerse in data context (P1)	Participate in activities that promote engagement with the data context	Understanding the nature of the data that is used across the task
Re-familiarize with statistical modelling ideas (P2)	Carry out familiar statistical modelling activities without using code	Application of statistical thinking
Describe computational steps of statistical modelling process (P3)	Use words to describe key computational steps of statistical modelling process	Decomposition of modelling steps and recognizing required computation
Match statistical modelling steps to code chunks (P4)	Read and match lines of code with statistical modelling steps	Recognizing aspects of code syntax and structure
Adapt code chunks with slight modifications (P5)	Identify features of code to change to complete a statistical modelling action	Integration of statistical and computational knowledge
Explore "what if?" changes to code (P6)	Modify at least one aspect of provided code to produce new or unexpected outputs	New knowledge gained by integrating statistical and computational thinking

Alongside the design principles, the construction of the task is simultaneously guided by four design considerations that inform broader decisions across the learning task: the *introduction of new knowledge*, the *data used*, the *tools used* and the *level of computational transparency*.

- The *introduction of new knowledge* (C1) refers to the use, content and sequence of phases and steps within a task to introduce learners to new ideas.
- The *data used* (C2) refers to selecting and using different variables or different sets of data within the same data context for the task.
- The *tools used* (C3) refers to combining different tools for statistical modelling (unplugged, code-driven, GUI-driven) and connecting actions or representations between tools within the task.
- The *level of computational transparency* (C4) refers to how obvious the computations performed by the tool are to the learner.

4. TASK CONSTRUCTION

We now discuss how our design framework was used to construct the task used for the research reported in this paper (see Appendix). The statistical modelling approach was an informal method that relied on teachers' reasoning with features of visualizations to construct a model that generated prediction intervals. The form of the prediction model was $predicted\ y = a + bx \pm error$, where a and b are the y-intercept and slope of a linear model respectively, and $error$ is a numeric value visually estimated by the teacher to model prediction error. The learning goal for the task was for the teachers to create a model that generated prediction intervals, using data sourced from an API.

4.1. DATA-RELATED DESIGN DECISIONS

The decision to use dynamic data from an API for the task was informed by the design consideration of *the data used* (C2), as well as the larger research study goal to provide a data science perspective for statistical modelling. The use of dynamic data provided an opportunity to learn about a new modern data source and related computational ideas, at the same time as supporting development of predictive modelling ideas by providing different data sets. Conventionally, the process of training and testing a prediction model would involve the same set of data, for example randomly allocating 80% for training and 20% for testing. However, as the data for this investigation was sourced from an API, it was decided to ask teachers to develop a model based on data from one search query and apply it to data from a different search query. The approach also allowed for similar reasoning one might face when using an inference made from a sample from one population and applying this generalization to another population or inferring to a wider population.

The OMDb API (omdbapi.com) was chosen because it provided data about movies, including information about movie ratings from three different sources. Additionally, OMDb provided an "API explorer", a graphical-user interface (GUI) to send requests to the API, without using a programming language, a feature that aligned with the design consideration of the *tools used* (C3). Specifically, it was decided to design questions that supported teachers to connect actions and representations between the GUI-driven "API explorer" and the code-driven tool. To support an informal approach to predictive modelling, facilitate access to dynamic data from an API and a code-driven tool, it was decided to design a web-based task.

4.2. WEB-BASED TASK DESIGN DECISIONS

The task was implemented using an interactive web page created using *RMarkdown* (Allaire et al., 2021), the *R* package *learnr* (Schloerke, Allaire, & Borges, 2018) and the tidyverse (Wickham, 2017) ecosystem of *R* packages. The *learnr* package provided many features that aligned with our design framework, including being able to: create individual steps for the task containing instructions, multimedia, and links to other webpages; run small "chunks" of *R* code within the task; embed interactive quiz questions within the task; and reveal each step of the task individually as the task progressed.

When developing the code provided to teachers, the statistical modelling approach for the task and the *level of computational transparency* (C4) were considered. The computations related to accessing data from the API were hidden to the teachers by using a function developed by the researcher (the first author). To support teachers' understanding of the prediction model, the computations related to linear regression and representing the prediction model graphically were revealed to the teachers. Code was developed using the *R* package *ggplot2* (Wickham, 2016) and provided to the teachers using code chunks. The code chunks facilitated fitting simple linear regression models, quantifying the average size of the prediction errors, tinkering with the model parameters, visualizing the prediction intervals, and training and testing a prediction model on different sets of data.

The quiz question functionality of *learnr* was used to construct what we call "tinker questions." A *tinker question* presents a set of related TRUE/FALSE statements that are deliberately written to require action by the learners *within a computational learning environment* in order to evaluate each statement. As opposed to quiz questions that assess existing or recently acquired knowledge, these *tinker questions* were used to stimulate the development of new computational knowledge by encouraging learners to make connections between actions and representations of the *tools used* (C3). Figure 2 presents a *tinker question*, which was used for step 1 of the task, along with a screenshot of the expected output from using the OMDb API explorer.

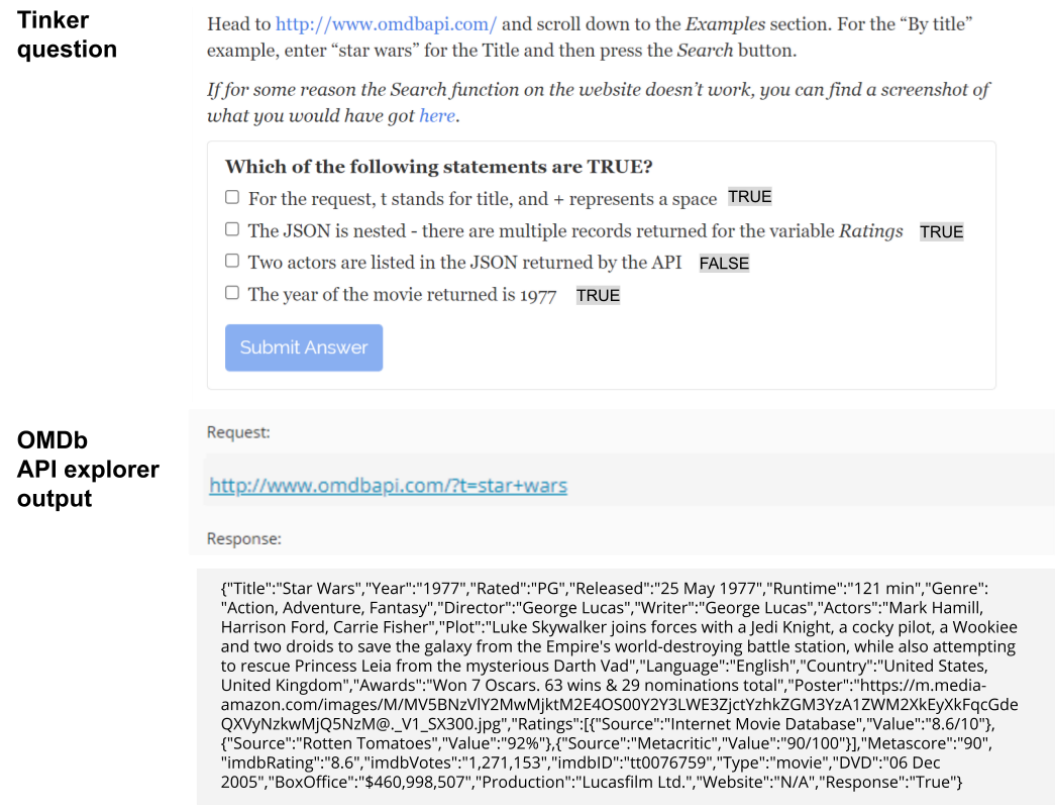


Figure 2. Example of tinker question used for step 1

The stem of the *tinker question* required the teachers to use the OMDb API explorer to send a request to the OMDb API. By searching for a movie with the title "star wars", both the URL request needed to make the request programmatically and the response to the request as JSON were generated. Each of the TRUE/FALSE statements presented for the *tinker question* was designed to help teachers to notice specific features of the output generated. Specific types of statements included those that were FALSE to create conflicting representations and those that encouraged a focus on recognizing patterns or structure within code or other computational representations. The teachers then submitted their answer, based on ticking which statements they believed were TRUE. If any of their answers were incorrect, the teachers then needed to repeat the process to identify which statements were TRUE or FALSE. The process of evaluating each TRUE/FALSE statement requires noticing and reflecting on the product(s) of each action, which we conjecture facilitates micro interrogative cycles (Wild & Pfannkuch, 1999).

The design consideration of the *introduction of new knowledge* (C1) led to the decision to use the *progressive reveal* setting for the *learnr*-constructed task. The steps of the task were revealed to the teachers progressively; when they completed each step, the next step appeared below the previous step(s) on the same web page. The *progressive* reveal setting also prevented the teachers from moving to the next step until the *tinker question* was successfully answered. Similarly, this setting prevented the teachers from moving to the next step until the code provided was executed at least once. These settings were used to carefully sequence the order and amount of new computational and statistical ideas introduced across the steps of the task.

4.3. SUMMARY OF TASK PHASES AND STEPS

A summary of the two phases of the task and how they relate to the six design principles and the steps of the task are provided in Table 2. The Appendix provides a static version of the full web-based task used, and links to the *R* code used to create the task and a demo version of the task. Each phase

and step of the task is now described and includes further explanations about how the design framework informed design decisions.

Table 2. A summary of the phases, design principle and steps used for the task

Phase	Summary of phase	Principle	Steps
1	Introduce dynamic data from an API	Immerse (P1)	1 to 5
2	Introduce predictive modelling ideas	Re-familiarize (P2), Describe (P3), Match (P4), Adapt (P5), Explore (P6)	6 to 14

The first phase focused on *immersing* (P1) teachers in dynamic data from an API, specifically movie ratings from the OMDb API. Tinker questions were used for four of the five steps of this phase (Q1, Q3, Q4, Q5). After being introduced to the structure of API requests and JSON in Step 1, Step 2 built further knowledge of the API by asking teachers to familiarize themselves with selected aspects of the API documentation. The knowledge introduced in Steps 1 and 2 was then used in combination with the code provided in Step 3 to modify requests to the API. Steps 4 and 5 encouraged teachers to adapt the code provided to change the API requests. These steps also allowed the teachers to further familiarize themselves with the nature of the data available about movies from the OMDb API, including the structure of the data and how the data could be manipulated to create new variables.

The second phase focused on introducing teachers to predictive modelling ideas, by drawing on familiar ideas of simple linear regression, notably the intercept, slope, and ideas of sampling variability. Discussion prompts were used across the questions in this phase to facilitate discussion between teachers and to stimulate thinking. Step 6 used a prompt asking teachers to discuss whether they expected there would be relationship between the Metascore and IMDb ratings for movies. Step 7 provided the complete code to create a scatter plot, but the x and y variables were incorrect, requiring a small change to produce the expected visual representation.

Steps 8 and 9 introduced teachers to fitting simple linear regression models using *R* code and interpreting relevant features of scatter plots. As it could not be assumed that teachers had constructed prediction intervals before, either informally or formally, these steps were used to *re-familiarize* (P2) teachers with the key computational steps required for an informal approach. In particular, Step 9 was used to shift the focus to prediction and to informal approaches for creating prediction intervals using visual features of the data and fitted line. The step specifically asked teachers to discuss whether "You can use an interval to give the predicted metascore rating based on the variation (vertical scatter) observed in the data/plot" and what two numbers they could use for a prediction interval of the `metascoreRating` of a movie that had an `imdbRating` of 7.

Step 10 presented the model to generate prediction intervals, and teachers were asked to estimate the error term for the model. The teachers needed to read the comments within the code that *described* (P3) the lines of the code and then *adapt* (P5) the code, by adjusting the values assigned to the y-intercept, slope, and error. The guidance provided to teachers about how to decide on the numeric value for the error was to base it on their visual estimate of how far away the points sat vertically from the line. Teachers were expected to use the visual representation of the prediction intervals generated – the line fitted and the yellow shaded band around this line – and to *match* (P4) these visual representations to the code used for their model. The discussion prompts in Step 10 were used to encourage teachers to think about model accuracy, by contrasting the percentage of correct predictions using point estimates (the fitted line only) with using prediction intervals (the model developed).

Step 11 asked teachers to discuss how well they thought their prediction model would work with movies that have the word "war" in their title. This step signaled a shift in the task to consider using the prediction model with a new set of data and introduced ideas of training and testing. Step 12 asked teachers to *adapt* (P5) code to *match* (P4) the model they developed in Step 10 and then to discuss how well their prediction model worked for movies with the word "war" in their title. To further explore the idea of generalizability and to provide another visual example of applying their prediction model to a new set of data, Step 13 provided teachers with code to generate and use their model with movies with the word "love" in their title.

Similar to Step 7, the code provided produced a visualization but there were a few more aspects of the code that needed *adapting* (P5) before the model could be evaluated using the new set of data. Step 14 provided teachers an opportunity to go back to any previous step and look more closely at the code used. This step also provided encouragement for teachers to *explore* (P6) changes to the code by following their own curiosity.

5. TASK IMPLEMENTATION

5.1. PARTICIPANTS

The participants were four female and two male Grade 12 statistics teachers. The teachers had, on average, 10.5 years high school teaching experience (mean = 10.5, min = 7, max = 14), and all had experience teaching Grade 12 statistics. Only one of the teachers was confident using the statistical programming language *R*, although this teacher had not taught *R* at the high school level. The teachers were participants in the larger study which involved four full-day professional development workshops. Permission to conduct the study was granted by the University of Auckland Human Participants Ethics Committee.

5.2. TEACHING EXPERIMENT

The teaching experiment, which is the focus of this paper, took place during the second day of four full-day professional development workshops that the teachers attended. This was the first task where the teachers were working entirely through a statistical modelling task using a code-driven tool and took place in the afternoon. The theme for the workshop was Star Wars and in the morning session teachers explored a Star Wars API (swapi.dev) by modifying URLs and finding information from the JSON returned (see Fergusson & Wild, 2021).

5.3. DATA COLLECTION

Teachers worked in pairs to complete the task with access to one laptop computer. The teacher pairings for this task were: Amelia and Ingrid, Alice and Naomi, and Harry and Nathan (pseudonyms have been used). All the teacher actions, responses, interactions with the software tools and conversations were captured using screen-based video and audio recordings with the browser-based tool Screencastify. Teachers were asked to think aloud as they completed the task, to support the analysis of thought processes, not just the product of thinking (Van Someren et al., 1994). To further capture teachers' thinking and actions while completing the task, reflective practice was encouraged after the task through a semi-structured group discussion. The group discussions were also used to elicit ideas and feedback from the teachers on the design of the task and its anticipated effectiveness for teaching students. Transcripts were made of any conversations and verbal responses to the task.

5.4. DATA ANALYSIS

The teacher responses and actions for each step of the task were compared and contrasted to explore how new computational ideas related to APIs were developing and what understandings about predictive modelling emerged. Using a task oriented qualitative analysis approach (Bakker & van Eerde, 2015), special attention was paid to episodes that revealed examples of statistical and computational understandings related to predictive modelling or APIs. The transcripts and screen recordings were read and viewed chronologically for each pair of teachers, and annotations made to the transcripts with conjectures about the nature of the teachers' thinking and reasoning. These annotations led to the identification of salient examples that would inform the research question with examples selected through a process of constant comparison (e.g., Bakker & van Eerde, 2015; Creswell, 2012).

6. RESULTS

All the teachers discussed the specific data context of movie ratings at various times throughout the task and developed new computational ideas related to APIs, including identifying features of JSON, modifying API queries, and using *R* code to access and visualize data from an API. Drawing on familiar ideas related to simple linear regression, the teachers were able to develop a model to generate prediction intervals by adapting code. They also demonstrated some understanding of training and testing a prediction model on different sets of data. We now present in more detail the results of the teachers' interactions with the task and the consequent emergent understandings that arose.

The focus for the task initially was immersing teachers in the data context of movie ratings, by integrating new computational knowledge related to APIs with familiar statistical knowledge including rectangular data sets and features of scatterplots (see Appendix, Steps 1 to 5). We observed that the teachers were able to read information about movies presented as JSON, even when it appeared they were unfamiliar with the associated vocabulary. For instance, when reading the statements for the Step 1 tinker question, Harry repeated the word "JSON" several times and Alice asked, "What's a JSON, is that the green thing?", indicating both teachers were unfamiliar with the word. The teachers then looked for "JSON" within the OMDb API documentation and successfully negotiated a new understanding that a "JSON" was a type of data structure or new computational representation. The teachers were able to identify the information required within the JSON to answer questions. To illustrate, when considering the statement that referred to the JSON being nested, Amelia responded, "Yes, it's got the little square brackets", demonstrating a connection between the word "nested" to this structural aspect of the computational representation.

The teachers demonstrated new ideas related to the structure of API queries, including the use of parameters and URL encoding. The sequencing of the questions, in particular introducing and progressively revealing new ideas at each step and re-using these new ideas in later questions, appeared to help to build these computational ideas. For example, when Harry and Nathan reached Step 3, the first statement they evaluated as TRUE or FALSE was, "There are around 728 (TV) series with wars in their title." Both teachers remembered from the previous step that there was a way to change the request and pointed to the `type=movie` part of the query request, before reviewing the OMDb API documentation to confirm the request needed to be changed to `type=series`. Although Step 3 reminded the teachers to use a "+" to represent a space for a search request, all the teachers referred back to knowledge gained from Step 1, when they used the API explorer to generate JSON for a search for "star wars". Thus, they connected actions and representations between the GUI-driven API explorer and the code-driven tool.

In Steps 4 and 5, where the teachers were provided with code that produced data about the movies in the form of an interactive table, they successfully modified the code and identified information about the number and nature of the variables available. For example, the teachers discovered the function `getResults()` will only return a maximum of 100 results and that data represented using JSON can also be represented as a data table. The task did not ask teachers to manipulate any of the variables provided in the datasets returned from the OMDb API. However, when answering the tinker question used for Step 5, the teachers developed new knowledge about the required computational actions for using other variables from the API data source. For instance, when considering whether the variable `Runtime` was numeric, Amelia said, "at the moment it's not, you would have to remove 'min'." Similarly, when discussing the variable `Genre`, Alice commented that each movie has "got multiple genres" and that the variable could not be used to focus on movies from just one genre.

The focus of the task then moved to predictive modelling, in particular, the development of an informal model to generate prediction intervals by extending familiar ideas of simple linear regression models (see Appendix, Steps 6 to 14). The teachers demonstrated they could quantify the size of the prediction errors using an informal visual approach that applied statistical thinking. To illustrate, in Step 9 the teachers were instructed to run the code provided, which produced a scatter plot with the least squares regression line fitted and were asked if the following statement was true: *You can use an interval to give the predicted metascore rating based on the variation (vertical scatter) observed in the data/plot*. The statement was specifically included to support teachers to think about predictive precision. Step 9 also included a discussion prompt asking teachers to create their own prediction

interval for the `metascoreRating` of another movie that had an `imdbRating` of 7. To determine their prediction interval, Nathan and Harry placed their pens on top of the computer screen and moved these up and down in parallel positions to the fitted line. Figure 3 shows Nathan's red pen in its final position and Harry's blue pen, which was still being positioned.

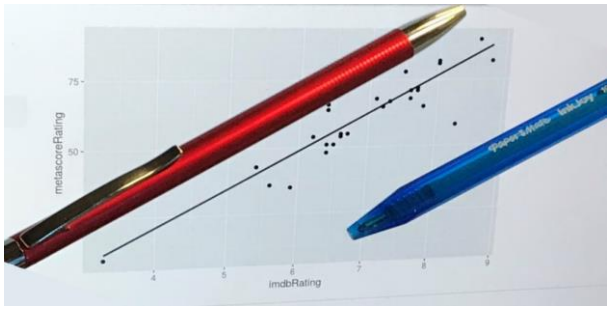


Figure 3. Nathan's red pen and Harry's blue pen being used to help determine their prediction interval

To help Harry position his pen, Nathan told him to, "Take out that bottom one! We're not using all the dots, are we?" Harry and Nathan then discussed how much "margin of error" they needed by estimating the vertical distance "above" and "below" the fitted line to their respective pens, settling on half a "square" or "12.5, around 12". Alice and Naomi used similar reasoning, but without pens, which was influenced by the scale breaks used for the `metascoreRating`, leading to the same prediction error value of 12.5.

Step 10 of the task required teachers to develop a prediction model using search data for movies with the word "star" in the title. Although the form of the prediction model was unfamiliar to the teachers, they were all able to adapt the code provided by changing the values for a, b, and error, thus demonstrating some understanding of how the code syntax and structure related to the prediction model. Each pair of teachers took a different approach to developing their prediction model, with their final attempts shown in Figure 4.

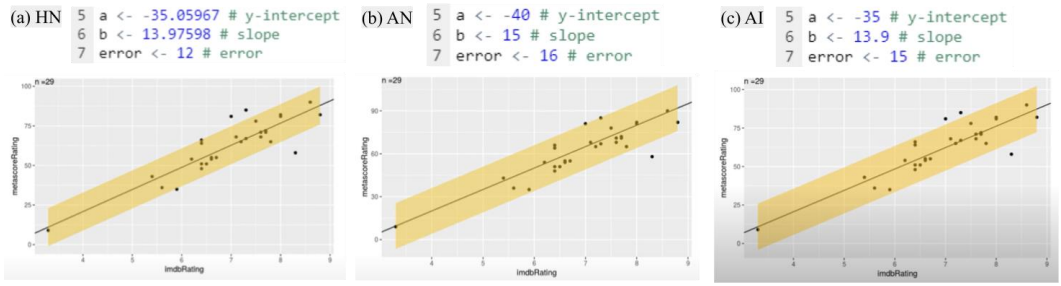


Figure 4. The prediction models developed by each pair of teachers for step 10 of the task

Harry and Nathan (HN, Figure 4) copied the coefficients for the fitted line, noticing that these were generated from the code provided for Step 10. When reading the line of code `error <- 3`, Harry said, "Error is 3, don't know what that means", and Nathan reminded him of Step 9 and the "margin of error", leading them to use the Step 9 error value of 12 for their model. Alice and Naomi (AN, Figure 4) decided on a and b values for their model entirely "by eye" rather than using the coefficients for the fitted line, which they didn't notice in the output. Alice was unsure about the value for the error, asking Naomi, "Can we play around with that as well?"

The teachers appeared to be guided by model accuracy when developing their informal prediction model. For example, Alice and Naomi initially started with the error used for Step 9 but increased the error value incrementally, with Naomi commenting "If we want 95%, we probably want it a little wider." While Alice and Naomi specifically discussed how many movies/points their prediction model "caught", the other two pairs of teachers only considered the accuracy of their prediction model after

reading the discussion prompt for Step 10. For example, Amelia stated in response to the prompt, "If you just use the line, you're basically all wrong, if you use the model we're about 90%".

The learning benefit of attempting to quantify the prediction error for the model first, before using code to visualize the error, can be illustrated in Amelia's and Ingrid's initial response to Step 10. Because they did not read the discussion prompt in Step 9, they did not create a prediction interval before reaching Step 10. Consequently, Amelia and Ingrid needed help from the researcher to begin the development of their prediction model in Step 10, as they were not able at this point in the task to integrate statistical and computational knowledge. There appeared to be too much new information for them to process in one step, as they had to grapple with new code, a new visualization, and a new type of model at the same time.

The focus on prediction intervals also appeared to support the teachers to consider the purpose of a prediction model. For instance, when the teachers were asked at the end of the task "In general, what do you want out of a prediction interval?", Naomi and Harry had the following exchange:

Naomi:	Well, you want it [the prediction interval] to be narrow but you also want it to be realistically narrow. It's no good saying you want it to be narrow if it doesn't actually predict very well.
Harry:	The thing is, I came back to the question, who is actually going to be using this. If you get a prediction interval of 24 overall it is almost a little bit meaningless, it's one quarter of 100.
Naomi:	Depends if you want to have a precise prediction, then yeah, you need a narrow interval for predicting a number but if you want to capture the variation then a wide interval is useful for communicating the variation.

In this exchange, Harry wanted a prediction interval that is *precise* or narrow and that can be used by someone to make a meaningful prediction, whereas Naomi considered the difference in modelling motivation between *explanation* and *prediction*.

The teachers were asked in Step 11 to discuss how well their model would work for movies with the word "war" in their title. They were given the opportunity to make changes to the code for their model, but none changed the y-intercept or slope for their model. Harry and Nathan kept the error value at 12, whereas Alice and Naomi decided to increase their error to 20, reasoning there would be more variation for the `imdbRating` scores for war movies. Amelia and Ingrid kept their error at 15 but only after a very long discussion on how males dominated the IMDb rating data. Later in the workshop, Amelia and Ingrid stated that they were unsure about whether they should have changed their model based on their contextual discussions or just used the features of the training data. Amelia reflected, "When building your model, and you know your training data is going to be quite different, do you leave it, or do you just make the error really big and be like 'it works'?" Hence, for four of the teachers, contextual considerations were an important part of the modelling process.

The teachers demonstrated awareness of an iterative approach to modelling, including the benefits of training and testing when developing a prediction model. Having easy access to new data sets via the API supported teachers to refine their informal prediction model and to appreciate the need to consider how well the model might work for new unseen data, as well as the need to consider uncertainty in the data and model. For example, Amelia made the following reflection:

> *I think it [the approach] makes the model more meaningful and the understanding about the uncertainty in it [the model], that makes that really realistic. Because the idea is, if you have all the data that you need right there, putting the model to it, you know how well it works for the data that's there. But if you are actually going to use that model to then make a prediction for something that you don't know as much about then 'oh my gosh!' you have to worry about uncertainty.*

When given the opportunity to test their model using two different sets of data in Steps 12 and 13 (movies with the word "war" in their title, then "love"), all pairs of teachers successfully adapted the code provided so that the data was appropriate. Additionally, all pairs of teachers modified the error value in response to the accuracy of the predictions. The modifications to the error value were based on considering how many movies/points were captured by the yellow band that represented the prediction intervals produced by the model. It was not intended that the teachers changed their models when using

a different set of data, as the task aimed to introduce teachers to the predictive modelling approach of training a model on one set of data and testing the *same model* on a new set of data. However, it appeared that by accessing different data sets and making changes to their prediction models, the task seemed to support ideas about generalizability, as can be seen in Naomi's reflection:

> *Actually one of the things that impressed me was that we didn't change the gradient or the intercept and yet that line fitted everything we tried pretty well. It pretty much went through the points no matter which thing we tried it on which was really good.*

Step 14 of the task was designed to encourage teachers to explore changes to the code with respect to the data and models, but all teachers did this as part of Step 13, when they were asked to test their model for movies with the word "love" in their title. For example, Naomi tried out searches for movie titles using other words without prompting from the researcher.

Overall, the teachers appeared to develop new statistical and computational ideas related to APIs and predictive modelling, which seemed to be influenced by the code-driven learning environment provided. The design of the web-based task meant that they could simultaneously: become familiar with the specific data-context of movie ratings; use *R* code to access and visualize data, as well as generate and visualize prediction intervals; and train and test a prediction model on different sets of data.

7. DISCUSSION

Our research question was interested in finding any new statistical and computational understandings related to APIs and predictive modelling that might emerge as teachers interacted with a task that was constructed using our design framework. We observed that all the teachers were able to use a code-driven tool to interact with APIs and to develop a model that generated prediction intervals. We believe the task provided a stimulating "first exposure" to predictive modelling for the teachers, which could be due to the specific design decisions made when constructing the task using the design framework. We now discuss specific design decisions, namely, the informal approach to development of a prediction model and the use of: dynamic data from an API; progressive reveal; code chunks; and tinker questions. We make tentative links between these decisions and the results presented about teachers' emergent understandings and reflect on our design framework.

It appears that the informal approach to developing a prediction model, primarily the visualization of error, provided an opportunity for teachers to develop their own reasoning for what made a good prediction model. Consistent with statistics education research concerning informal inference, the teachers developed their own rules for "making calls" when they decided the value of the error for their prediction model (e.g., Makar & Rubin, 2018; Fergusson & Pfannkuch, 2020). Additionally, the visualization of prediction intervals appeared to support teachers to assess their models in terms of both predictive precision and accuracy. Although specific concepts of underfitting, overfitting and generalizability were not explicitly discussed by the teachers during the task, Naomi's reflection about keeping the slope and intercept of the prediction model fixed indicated that the task provided a foundation for further development of these predictive modelling ideas.

The use of movie ratings data obtained directly from an API appeared to support the teachers' development of predictive modelling ideas across the task (cf. Weiland, 2017). Using their prediction model with a new set of data to generate prediction intervals appeared to help teachers become familiar with the ideas of data for training and data for testing, a necessary focus according to Biehler and Schulte (2017). The use of an API to access and use more than one data set as part of the modelling process, rather than just one static data set, also appeared to help teachers appreciate the predictive goal of the modelling task which, as Amelia said, was "to use that model to then make a prediction for something that you don't know" (cf. Casey & Wasserman, 2015).

The number and scope of new ideas related to predictive modelling and APIs introduced to teachers within the task was not trivial. The decision to use a web-based task, created using the *R* package *learnr*, provided the important feature of progressively revealing each step visually. To minimize cognitive load, the web-based tasks comprised a sequence of steps that carefully ordered the introduction of new statistical and computational ideas (cf. Wouters et al., 2008). The use of small chunks of code that only required small modifications allowed teachers to engage with new computational ideas such as creating scatterplots, similar to the findings of Wiedemann et al. (2020) from research involving high school

students using *learnr* to explore mathematical modelling. In alignment with Son et al. (2021), we argue that the use of *R* code in the task was germane load (Sweller et al., 1998), potentially because the computations represented by the code were familiar to the teachers. By considering the relationship between the tool, task and thinking (e.g., Biehler, 2018), we used code in the task to enable teachers to explore changes to their model and to visualize these changes instantaneously, thus potentially enhancing their statistical thinking (e.g., Ben-Zvi, 2000).

The design and use of tinker questions appeared to support the introduction of new computational ideas. As teachers only needed to focus on one TRUE/FALSE statement at a time, we observed learning about new computational representations or actions was reduced into "bite sized" interactions. By connecting actions or representations between GUI-driven and code-driven tools, the teachers appeared to develop new understanding of using APIs, for example features of data structures such as JSON. The interactive approach employed by the tinker questions to support the learning of new computational knowledge could also support students to become active participants in learning from modern data (e.g., Gould, 2010). Furthermore, the tinker questions could help develop *data habits of mind* (see Finzer, 2013) as they provided guidance for noticing and considering selected features of computational representations or actions.

The specific design decisions to use dynamic data from an API, progressive reveal code chunks and tinker questions for the web-based task were informed by the design considerations of our framework: the *introduction of new knowledge* (C1), the *data used* (C2), the *tools used* (C3) and the *level of computational transparency* (C4). The design principles of our framework (see Table 1) also guided the construction of task phases and steps. Using the *immerse* (P1) principle, the first phase of the task engaged teachers with the movie ratings data context. All teachers discussed specific contextual considerations of movie ratings, for example, the differences between the rating systems used by IMDb and Rotten Tomatoes and attempted to use contextual information to guide model decisions later in the task (cf. Pfannkuch, 2011). In the second phase of the task, using the *re-familiarize* (P2) principle, the teachers were first encouraged to extend familiar ideas of linear regression models to create prediction intervals without using code. The teachers demonstrated statistical thinking when they quantified the size of the prediction errors using an informal visual approach, using methods such as placing their pens on top of the computer screen.

Step 10 of the task used the *describe* (P3), *match* (P4) and *adapt* (P5) principles and the teacher interactions with this step varied, perhaps because there was too much new information presented in one step. The code provided descriptions of the computational steps and output in terms of the prediction model, however, teachers struggled with matching or adapting at least one aspect of the code. In the first iteration of the design framework (Fergusson & Pfannkuch, 2021), the task constructed used the design principles separately for consecutive steps, which meant the teachers knew what computation was needed for each modelling step before reading the code and could focus on how the code syntax and structure tells the computer how to carry out each step (cf. Pruim et al., 2017).

The *adapt* (P5) principle appeared to be more successfully applied in Steps 12 and 13. All pairs of teachers successfully integrated statistical and computational knowledge when they adapted the code to source appropriate data from the API and modified the error value of their prediction model in response to the accuracy of the predictions. Using the *explore* (P6) principle, Step 14 encouraged teachers to explore changes to the code with respect to the data and models, with the expectation that new or unexpected outputs from these adaptions would stimulate an integration of statistical and computational thinking and new knowledge. Although we have presented results that indicate new statistical and computational understandings did emerge for the teachers in earlier steps of the task, on reflection Step 14 could have asked the teachers to explore prediction models using a different combination of the variables and data available from the OMDb API.

There are of course limitations to this research. This was a small exploratory study involving six high school statistics teachers, all of whom were familiar with simple linear regression but unfamiliar with the informal predictive modelling approach introduced in the task. Furthermore, the teacher participants had minimal to no experience with APIs or with programming and had not used coding-based approaches for teaching statistical modelling. Although the emergent teacher understandings from our study cannot be generalized to all statistics teachers nor attributed directly to our design framework, we believe they provide important insights into how statistical modelling learning tasks could be designed, particularly for statistics teachers and students new to computer programming.

The learning task from the first iteration was constructed to move learners from a GUI-driven tool to a code-driven tool for carrying out the randomization test. The second iteration of our design framework was applied to develop a task that used a code-driven tool to learn about predictive modelling. Further iterations of our research will be used to evaluate and refine our design framework, by constructing and implementing tasks that introduce new sources of data and/or statistical modelling approaches alongside code-driven tools. For example, a follow up task could be constructed using our design framework that explores other variables and data available from the OMDb API, from both a predictive modelling approach as well as a data wrangling approach (cf. Hardin, 2018).

We have provided some practical design solutions that balance the learning of new statistical and computational ideas. In particular, the use of *tinker questions* within a web-based task has the potential to develop learners' confidence with code experimentation. The web-based task, created using the *R* package *learnr*, provided easy access to new data to test prediction models and appeared to support the development of new statistical and computational ideas related to predictive modelling and APIs. Furthermore, for high school implementation, our web-based task has the advantage of only needing a browser to engage with the code-driven tool, rather than additional knowledge of computer systems and software installation. The task has since been modified slightly and used successfully with hundreds of undergraduate statistics students learning remotely, and therefore there is the scope for broader implementation within an online environment. More research, however, is recommended on task design that supports teacher and student learning in data science at the high school level, which could include how teachers construct new tasks in line with the proposed design principles and considerations (cf. Sentance et al., 2019).

REFERENCES

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., & Iannone, R. (2021). rmarkdown: Dynamic documents for R. *RStudio*. https://rmarkdown.rstudio.com

Anderson, T., & Shattuck, J. (2012). Design-based research: A decade of progress in education research? *Educational Researcher*, *41*(1), 16–25. https://doi.org/10.3102%2F0013189X11428813

Bakker, A. (2018). *Design research in education: A practical guide for early career researchers.* Routledge. https://doi.org/10.4324/9780203701010

Bakker, A., & van Eerde, D. (2015). An introduction to design-based research with an example from statistics education. In A. Bikner-Ahsbahs, C. Knipping, & N. Presmeg (Eds.), *Approaches to qualitative research in mathematics education* (pp. 429–466). Springer. https://doi.org/10.1007/978-94-017-9181-6_16

Bargagliotti, A., Franklin, C., Arnold, P., Gould, R., Johnson, S., Perez, L., & Spangler, D. (2020). *Pre-K–12 Guidelines for Assessment and Instruction in Statistics Education (GAISE) report II*. American Statistical Association.

Ben-Zvi, D. (2000). Toward understanding the role of technological tools in statistical learning. *Mathematical Thinking and Learning*, *2*(1-2), 127–155. https://doi.org/10.1207/S15327833MTL0202_6

Biehler, R. (2018). Design principles, realizations and uses of software supporting the learning and the doing of statistics: A reflection on developments since the late 1990s. In M. A. Sorto, A. White, & L. Guyot (Eds.), *Looking back, looking forward. Proceedings of the Tenth International Conference on Teaching Statistics* (ICOTS10), Kyoto, Japan, July 8–13. International Statistical Institute. https://iase-web.org/icots/10/proceedings/pdfs/ICOTS10_1B1.pdf

Biehler, R., & Schulte, C. (2017). Perspectives for an interdisciplinary data science curriculum at German secondary schools. In R. Biehler, L. Budde, D. Frischemeier, B. Heinemann, S. Podworny, C. Schulte, & T. Wassong (Eds.), *Paderborn Symposium on Data Science Education at School Level 2017: The Collected Extended Abstracts* (pp. 2–14). Universitätsbibliothek Paderborn.

Burr, W., Chevalier, F., Collins, C., Gibbs, A. L., Ng, R., & Wild, C. J. (2021). Computational skills by stealth in introductory data science teaching. *Teaching Statistics*, *43*, S34–S51. https://doi.org/10.1111/test.12277

Casey, S. A., & Wasserman, N. H. (2015). Teachers' knowledge about informal line of best fit. *Statistics Education Research Journal*, *14*(1), 8–35. https://doi.org/10.52041/serj.v14i1.267

Cetinkaya-Rundel, M., & Rundel, C. (2018). Infrastructure and tools for teaching computing throughout the statistical curriculum. *The American Statistician*, *72*(1), 58–65. https://doi.org/10.1080/00031305.2017.1397549

De Veaux, R. D., Agarwal, M., Averett, M., Baumer, B. S., Bray, A., Bressoud, T. C., Bryant, L., Cheng, L. Z., Francis, A., Gould, R., Kim, A. Y., Kretchmar, M., Lu, Q., Moskol, A., Nolan, D., Pelayo, R., Raleigh, S., Sethi, R. J., Sondjaja, M., … Ye, P. (2017). Curriculum guidelines for undergraduate programs in data science. *Annual Review of Statistics and Its Application*, *4*, 15–30. https://doi.org/10.1146/annurev-statistics-060116-053930

Edelson, D. C. (2002). Design research: What we learn when we engage in design. *The Journal of the Learning sciences*, *11*(1), 105–121. https://doi.org/10.1207/S15327809JLS1101_4

Engel, J. (2017). Statistical literacy for active citizenship: A call for data science education. *Statistics Education Research Journal*, *16* (1), 44–49. https://doi.org/10.52041/serj.v16i1.213

Erickson, T. (2020). The BART data portal. *An introduction to data science with CODAP*. http://codap.xyz/awash/bart-chapter.html

Fergusson, A., & Pfannkuch, M. (2020). Development of an informal test for the fit of a probability distribution model for teaching. *Journal of Statistics Education*, *28*(3), 344–357. https://doi.org/10.1080/10691898.2020.1837039

Fergusson, A., & Pfannkuch, M. (2021). Introducing teachers who use GUI-driven tools for the randomization test to code-driven tools. *Mathematical Thinking and Learning.* https://doi.org/10.1080/10986065.2021.1922856

Fergusson, A., & Wild, C. J. (2021). On traversing the data landscape: Introducing APIs to data-science students. *Teaching Statistics*, *43*, S71–S83. https://doi.org/10.1111/test.12266

Finzer, W. (2013). The data science education dilemma. *Technology Innovations in Statistics Education*, *7*(2). https://doi.org/10.5070/T572013891

Gould, R. (2010). Statistics and the modern student. *International Statistical Review*, 78(2), 297–315. https://doi.org/10.1111/j.1751-5823.2010.00117.x

Gould, R. (2017). Data literacy is statistical literacy. *Statistics Education Research Journal*, *16*(1), 22–25. https://doi.org/10.52041/serj.v16i1.209

Gould, R. (2021). Toward data-scientific thinking. *Teaching Statistics*, *43*, S11–S22. https://doi.org/10.1111/test.12267

Hardin, J. (2018). Dynamic data in the statistics classroom. *Technology Innovations in Statistics Education*, *11*(1). https://doi.org/10.5070/T5111031079

Kaplan, D. (2007). Computing and introductory statistics. *Technology Innovations in Statistics Education*, *1*(1). https://doi.org/10.5070/T511000030

Konold, C., & Miller, C. (2015). *TinkerPlots™ Version 2.3 [Computer Software]*. Learn Troop. http://www.tinkerplots.com/

Magana, A. J., Vasileska, D., & Ahmed, S. (2011). Work in progress—a transparency and scaffolding framework for computational simulation tools. *2011 Frontiers in Education Conference (FIE)*, (pp. S4G–1). IEEE. https://doi.org/10.1109/FIE.2011.6142803

Makar, K., & Rubin, A. (2018). Learning about statistical inference. In D. Ben-Zvi, K. Makar, & J. Garfield (Eds.), *International handbook of research in statistics education* (pp. 261–294). Springer. https://doi.org/10.1007/978-3-319-66195-7_8

McKenney, S., & Reeves, T. C. (2018). *Conducting educational design research*. Routledge. https://doi.org/10.4324/9781315105642

National Academies of Sciences, Engineering, and Medicine. (2018). *Data science for undergraduates: Opportunities and options.* The National Academies of Sciences Engineering Medicine. https://doi.org/10.17226/25104

Nolan, D., & Temple Lang, D. (2010). Computing in the statistics curricula. *The American Statistician*, *64*(2), 97–107. https://doi.org/10.1198/tast.2010.09132

New Zealand Qualifications Authority. (2019). *Annotated exemplar Level 3 AS91581*. Author. https://www.nzqa.govt.nz/ncea/subjects/mathematics/exemplars/level-3-as91581/

Pfannkuch, M. (2011). The role of context in developing informal statistical inferential reasoning: A classroom study. *Mathematical Thinking and Learning*, *13*(1–2), 27–46. https://doi.org/10.1080/10986065.2011.538302

Pruim, R., Kaplan, D. T., & Horton, N. J. (2017). The mosaic package: Helping students to 'think with data' using R. *The R Journal*, *9*(1), 77–102.

R Core Team. (2020). *R: A language and environment for statistical computing*. https://www.R-project.org/

Reeves, T. C. (2007). Design-based research from a technology perspective. In J. Van den Akker, K. Gravemeijer, S. McKenney & N. Nieveen (Eds.), *Educational design research* (pp. 52–56). Routledge.

Ridgway, J. (2016). Implications of the data revolution for statistics education. *International Statistical Review, 84*(3), 528–549. https://doi.org/10.1111/insr.12110

Sentance, S., Waite, J., & Kallia, M. (2019). Teaching computer programming with PRIMM: A sociocultural perspective. *Computer Science Education, 29*(2-3), 136–176. https://doi.org/10.1080/08993408.2019.1608781

Shaughnessy, J. M. (1997). Missed opportunities in research on the teaching and learning of data and chance. In F. Biddulph & K. Carr (Eds.), *People in mathematics education. Proceedings of the Twentieth Annual Conference of the Mathematics Research Group of Australasia (MERGA-20, July, 1990), Rotorua, New Zealand* (Vol. 1, pp. 6–22). MERGA.

Schloerke, B., Allaire, J., & Borges, B. (2018). Learnr: Interactive tutorials for R. *CRAN*. https://CRAN.R-project.org/package=learnr

Son, J. Y., Blake, A. B., Fries, L., & Stigler, J. W. (2021). Modeling first: Applying learning science to the teaching of introductory statistics. *Journal of Statistics and Data Science Education*, *29*(1), 4–21. https://doi.org/10.1080/10691898.2020.1844106

Sweller, J., van Merriënboer, J. J. G., & Paas, F. G. W. (1998). Cognitive architecture and instructional design. *Educational Psychology Review*, *10*(3), 251–296. https://doi.org/10.1023/A:1022193728205

Van den Akker, J. (1999). Principles and methods of development research. In J. Van den Akker, R. M. Branch, K. Gustafson, N. Nieveen & T. Plomp (Eds.), *Design approaches and tools in education and training*, (pp. 1–14). Springer. https://doi.org/10.1007/978-94-011-4255-7_1

Van Someren, M. W., Barnard, Y. F., & Sandberg, J. A. C. (1994). The think aloud method: A practical approach to modelling cognitive processes. Academic Press.

Weiland, T. (2017). The importance of context in task selection. *Teaching Statistics, 39(*1), 20–25. https://doi.org/10.1111/test.12116

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag.

Wickham, H. (2017). Tidyverse: Easily install and load the "tidyverse". *CRAN*. https://CRAN.R-project.org/package=tidyverse

Wiedemann, K., Chao, J., Galluzzo, B., & Simoneau, E. (2020). Mathematical modeling with R: Embedding computational thinking into high school math classes. *ACM Inroads*, *11*(1), 33–42. https://doi.org/10.1145/3380956

Wild, C. J., & Pfannkuch, M. (1999). Statistical thinking in empirical enquiry. *International Statistical Review*, *67*(3), 223–248. https://doi.org/10.1111/j.1751-5823.1999.tb00442.x

Wild, C. J., Pfannkuch, M., Regan, M., & Parsonage, R. (2017). Accessible conceptions of statistical inference: Pulling ourselves up by the bootstraps. *International Statistical Review*, *85*(1), 84–107. https://doi.org/10.1111/insr.12117

Wouters, P., Paas, F., & van Merriënboer, J. J. (2008). How to optimize learning from animated models: A review of guidelines based on cognitive load. *Review of Educational Research*, *78*(3), 645–675. https://doi.org/10.3102%2F0034654308320320

Zieffler, A., Justice, N., delMas, R., & Huberty, M. D. (2021). The use of algorithmic models to develop secondary teachers' understanding of the statistical modeling process. *Journal of Statistics and Data Science Education, 29*(1), 131–147. https://doi.org/10.1080/26939169.2021.1900759

ANNA FERGUSSON
Department of Statistics
The University of Auckland
Private Bag 92019
Auckland, New Zealand

APPENDIX

A static version of the web-based task is provided below. The code used to create this task is available at https://gist.github.com/annafergusson/20f2149cde388e733b683d0436f227d3

A live version of this task is available at https://undercoverdatascience.shinyapps.io/SERJ_movies/, although note that the quiz questions may have incorrect answers as the data has changed since the task was implemented.

Building and testing an informal prediction model

Ratings wars

1.

Head to http://www.omdbapi.com/ and scroll down to the *Examples* section. For the "By title" example, enter "star wars" for the Title and then press the *Search* button.

If for some reason the Search function on the website doesn't work, you can find a screenshot of what you would have got here.

Which of the following statements are TRUE?

- ☐ The JSON is nested - there are multiple records returned for the variable *Ratings*
- ☐ The year of the movie returned is 1977
- ☐ For the request, t stands for title, and + represents a space
- ☐ Two actors are listed in the JSON returned by the API

Submit Answer

2.

Scroll up the page until you find the table that has *By Search* above it. This table shows you parameters you can use in your query to search for movies using the OMDb API.

Which of the following statements are TRUE?

- [ ] You can limit your search to the year of release
- [ ] You can select which page of the results to return
- [ ] You can only have the data returned as JSON
- [ ] There are three types movies you can search for

Submit Answer

3.

The code below makes a request to the OMDb API using some of the parameters from the *By search* table we just looked at. Try changing either the word(s) used in the search_string or parts of the url and running the code chunk, to answer the question that follows the code chunk.

Remember to use a + for a space

Code | Start Over | Run Code

```
search_string <- "wars"

# this line of code puts together the request that will be sent to the API
url <- paste0("http://www.omdbapi.com/?s=", search_string,"&type=movie&apikey=",omdbKey2)

# these lines of code make the request to the API and then returns how many results have beer
output <- fromJSON(url, flatten=TRUE)
as.numeric(output$totalResults)
```

Which of the following statements are TRUE?

- [ ] There are around 17 movies with both dog and cat in their title
- [ ] There are around 728 (TV) series with wars in their title
- [ ] There are around 2357 movies with star in their title
- [ ] There are more movies with cat in their title than dog

Submit Answer

4.

Let's get some data! The code below will request the first *n* results for whatever you search for, and display the results in a nice table.

Be patient - the code may take some time to run!

Code | Start Over | Run Code

```
search_string <- "star"
number_results <- 40
getResults(search_string, number_results)
```

Now have a go at this question, which may require some code tinkering!

Which of the following statements are TRUE?

- [ ] The last variable/column is called **DVD**
- [ ] You can't request any more than 100 results with this function
- [ ] There are 10 rows (or movies) on each 'page' of the data table
- [ ] The 68th movie when you search for star is called Lego Star Wars: The Padawan Menace

Submit Answer

5.

In today's task, we won't focus too much on data manipulation or cleaning. But take another look at some of those variables to answer the question below!

Code | Start Over | Run Code

```
search_string <- "star"
number_results <- 20
getResults(search_string, number_results)
```

Which of the following statements are TRUE?

- [ ] You could extract the month the movie was released from the variable **Released**
- [ ] **Runtime** could be a numeric variable but probably won't be read as one by graphing software
- [ ] You could attempt to estimate the gender of the Directors using their first name
- [ ] The variable **Genre** is totally fine how it is!

Submit Answer

6.

Our goal for this task is to build an informal prediction model that uses the **imdbRating** for a movie to predict the **metascoreRating** for a movie.

Read some more about how IMDb calculates their ratings for movies and how it is different from how metacritic calculates their ratings (called a metascore).

Thinking about our morning session on data structures, take yet another look at the data below and discuss with each other how each variable has been created or measured.

Code | Start Over | Run Code

```
search_string <- "star"
number_results <- 20
getResults(search_string, number_results)
```

Discuss with each other whether you expect there will be relationship between these two types of ratings for movies.

7.

Let's see what we can learn from the data! The code below will display a scatterplot, but not of the two variables we want! Run the code and discuss what you see.

Change the code so that it shows **metascoreRating** on the y-axis (the response variable) and **imdbRating** on the x-axis (the explanatory variable).

Code | Start Over | Run Code

```
# our search results are now stored in the variable star_movies
star_movies %>%
  ggplot(aes(x=Year, y=imdbRating)) + geom_point()
```

Discuss with each other what the plot reveals about the relationship between the **metascoreRating** and the **imdbRating** for these movies.

8.

Run the code below which will fit a line to the data and output the correlation coefficient for the two variables.

Code | Start Over | Run Code

```
star_movies %>%
  filter(!is.na(metascoreRating)) %>%
  ggplot(aes(x=imdbRating, y=metascoreRating)) + geom_point() + geom_smooth(method = "lm", se

# calculate correlation coefficent (r)
 cor(star_movies$imdbRating, star_movies$metascoreRating, use = "complete.obs")
```

Discuss with each other how well you think the line models the relationship between the **metascoreRating** and the **imdbRating** for these movies.

9.

Suppose we wanted to predict the metascoreRating of another movie that had a imdbRating of 7.

Using the data and line displayed in the plot in the previous section, answer the question below.

Which of the following statements are TRUE?

- [ ] Movies with the same imdbRatings can have different metascoreRatings
- [ ] You can use an interval to give the predicted metascore rating based on the variation (vertical scatter) observed in the data/plot
- [ ] You would be just as confident making predictions using imdbRatings of 3 or lower as you would making predictions using imdbRatings of between 5 to 9
- [ ] Using the line, you could predict the metascoreRating to be around 62

Submit Answer

Discuss with each other what two numbers you could use for a prediction interval of the **metascoreRating** of another movie that had a **imdbRating** of 7.

10.

We are now going to build an informal prediction model that has the form:

> predicted **metascoreRating** = a + b * **imdbRating** ± error

a is the y-intercept and *b* is the slope of the "line of best fit" (a simple linear regression model).

error will be based on your visual estimate of how far away the points sit vertically from the line.

This model will give you a prediction interval for the **metascoreRating** using a movie's **imdbRating**.

Run the code below and then read the comments within the code to learn how to build your informal prediction model. Adjust your model until you are happy with it.

Code | Start Over | Run Code

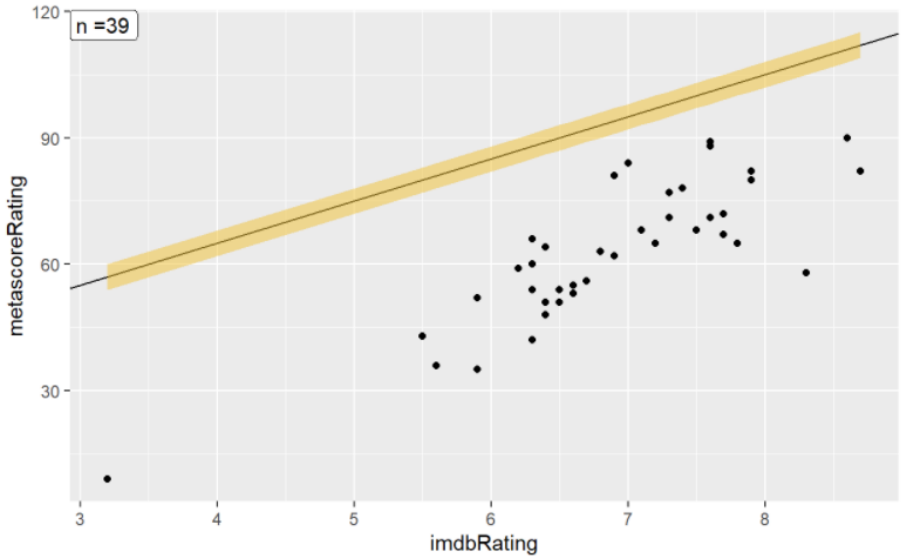
```
# These lines of code will give you the equation of the line fitted in the previous secti
fitted_line <- lm(metascoreRating ~ imdbRating, data = star_movies)

coef(fitted_line)# You will need to change the values assigned to each component of your
a <- 25 # y-intercept
b <- 10 # slope
error <- 3 # error

# we need to know how many points of data we are building our model with
points_plotted <- star_movies %>%
  filter(!is.na(metascoreRating)) %>%
  nrow()

#plot magic!
star_movies %>%
```

```
(Intercept)  imdbRating
  -36.22568    14.38823
```



Discuss the following with each other:

- If you only use the fitted line to make predictions, what percentage of movies have their **metascoreRating** correctly predicted from their **imdbRating**?
- If you use your informal prediction model to make predictions instead, what percentage of movies have their **metascoreRating** correctly predicted from their **imdbRating**?

11.

Time to test that model of yours out!

Before we do, discuss with each other how well you think your model will work to predict the **metascoreRating** from a movie's **imdbRating** for movies with war in their title.

PS - you've still got time to change your model!

12.

In the code below, add the details of your model BEFORE running the code (no sneaky peeks!)

Code | Start Over | Run Code

```
# You will need to change the values below to your model built in the previous sections
a <- 50 # y-intercept
b <- 0 # slope
error <- 5 # error

# we need to know how many points of data we are testing our model with
points_plotted <- war_movies %>%
  filter(!is.na(metascoreRating)) %>%
  nrow()

#plot magic!
war_movies %>%
  filter(!is.na(metascoreRating)) %>%
  ggplot(aes(x=imdbRating, y=metascoreRating)) + geom_point() + geom_abline(intercept = a, s
```

Discuss with each other how well your model performed on the test data.

13.

The final battle of the ratings!

Love is better than war, so let's try out your model on **50** movies with the love in their title.

This time you are going to have to change a few things in the code before you can test your model. Remember you can scroll back up to previous sections to see what was done there!

Code | Start Over | Run Code

```
search_string <- "hate"
number_results <- 20
love_movies <- getResults(search_string, number_results)

# You will need to change the values below to your model built in the previous sections
a <- 50 # y-intercept
b <- 0 # slope
error <- 5 # error

# we need to know how many points of data we are testing our model with
points_plotted <- love_movies %>%
  filter(!is.na(metascoreRating)) %>%
  nrow()

#plot magic!
```

Discuss with each other how well your model performed on this test data. You can also discuss any potential issues you noticed during the modelling process and what you might do differently next time to help deal with these issues.

14.

And we're done!

If there's still time, why don't you scroll back up the page and take a closer look at the code used for each section? Talk with your partner and see if you can work out what is going on.

Feel free to tinker with the code - you can't break anything, and ctrl+z will undo any changes you make :-)