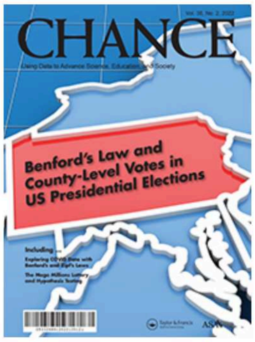

CHANCE

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ucha20

Spam Four Ways: Making Sense of Text Data

Nicholas J. Horton, Jie Chao, William Finzer & Phebe Palmer

To cite this article: Nicholas J. Horton, Jie Chao, William Finzer & Phebe Palmer (2022) Spam Four Ways: Making Sense of Text Data, CHANCE, 35:2, 32-40, DOI: 10.1080/09332480.2022.2066414

To link to this article: https://doi.org/10.1080/09332480.2022.2066414

Published online: 04 May 2022.

[Taking a Chance in the Classroom]

Mine Çetinkaya-Rundel and Maria Tackett
Column Editors

Spam Four Ways: Making Sense of Text Data

Nicholas J. Horton, Jie Chao, William Finzer, and Phebe Palmer

The world is full of data, and much of it is unstructured. An example is text data, which forms a critical part of our lives through books, magazines, and the internet. Surprisingly, despite the key role of language arts in all aspects of education, text analysis has not traditionally played a major part in statistical education.

While there are many interesting literary analyses one might consider, we explore a more mundane but familiar example by looking at a text string taken from an email subject line is spam (an unwanted or inappropriate email message).

What Is Spam?

Email users are unfortunately all too familiar with spam. The website *www.spamlaws.com*, based on materials created by David E. Sorkin from John Marshall Law School, provides additional background on these indiscriminate and unwanted communications. This site defines spam as "submitting the same message to a large group of individuals in an effort to force the message onto people who would otherwise choose not to receive this message." The problem of spam is notable: They estimate there are more than 14 billion spam messages created daily—a substantial fraction of all emails.

Spam Four Ways: Approaches to Teaching Students to Classify Spam

To keep things tractable, we will focus on ways one might use information from subject lines to determine whether emails are likely to be "spam" (unwanted messages) or "non-spam." This is a form of supervised learning or predictive analytics, where the email subject line is the data, the labels are the indicators of spam (or not), and our goal is predicting the outcome.

We realize whether a message is considered spam or not is subjective: The question of what label is given to subject lines can engender a productive conversation.

This form of text analytics is also a rudimentary type of natural language processing, albeit with the simple goal of classification, rather than the more challenging one of comprehension.

Five examples of the type of email subject lines we would call spam include the following:

1. Dear trusted one
2. From Mrs Kadirat Usman. thanks and remain bless. Urgent please
3. FBI & IRS seized goods at 99% off! Police Auctions!
4. Re: PROTECT YOUR COMPUTERAGAINST HARMFUL VIRUSES
5. Market Internet Access - No Investment Needed

Five examples of legitimate (non-spam) email subjects include the following:

1. Receipt for your Payment to Edible Twin Cities
2. Your Zappos.com order #: 65801179
3. STEM Education: faculty opening
4. Learning Outcomes working retreat
5. Re: Classifier software design

Before we proceed, it's important to keep in mind (particularly for instructors) that unsavory, obscene, or offensive email subject lines can arise in any real-world investigation involving data from the internet.

Model Eliciting Activities and Spam Classification

The CATALST (Change Agents for Teaching and Learning Statistics) Group defines Model Eliciting Activities (MEA) as those designed to let students explore open-ended problems representing real-world problems. We consider an MEA developed by researchers at the University of Minnesota in which students are encouraged to invent approaches to classify spam messages. This MEA has often been incorporated into an undergraduate introductory statistics course by the first author.

The activity is designed to fit within a 70-minute class period (though it could be split into two class sessions). Students begin by reading a brief overview of the problem of spam, and then individually complete some readiness questions (e.g., How can you determine the accuracy of a spam filter?). The overall goal of the task is described as developing a spam filter for their work supervisor.

The students are divided into groups of between two and five students. Each group is given two pieces of paper: One includes a list of 50 email subject lines labeled as "spam" and the other a list of 50 non-spam subject lines. (The sample subject lines given earlier were taken from these lists.)

After reviewing the two sets of 50 subject lines, the students are asked to develop a rule or set of rules classifying emails using only the subject line and to determine how they would judge the accuracy of their rule. These rules might involve the development of word lists (e.g., "Dear," "Blessed," "Urgent") indicative of spam or others (e.g., "Re" or "STEM") more commonly found in non-spam subject lines.

Once the group comes to a consensus regarding their rules and applies it to the data they were provided, they are asked to write the rules down and share them with the class. They were also asked to report on sensitivity (the proportion of spam defined as spam) and specificity (proportion of non-spam declared to be non-spam) of the rule on their training data in a table with rows for each group.

Rules vary, with some simple (e.g., if any of these words appear, call it spam) and others containing more complicated Boolean logic with multiple steps (e.g., if these words appear, declare it non-spam, else if more than two punctuation marks or majority capitalized, then declare it spam). It's been valuable to have each member of the team use the rule independently to calculate sensitivity and specificity to confirm there is a shared understanding of how it is implemented.

The next step is for the teams to apply their rule to a set of 100 new subject lines provided to them, 50 of these being spam and 50 non-spam. They then report their results. It is common for rules to fare slightly worse on this second testing data than on the original training data.

The last step in the MEA is for the group to write a one- to two-page report for their work supervisor on their results, including a description of the method, their measures of accuracy, an assessment of how it worked (or not), and a way they might adapt the spam filter based on their results. Students draft the report outside of class. This reflective assignment is intended to help make the concepts and process more concrete and aid students in improving their ability to practice the language of statistics.

On its own, this MEA has felt like a valuable way to introduce aspects of predictive analytics using logistic regression or decision trees. The activity helps students engage in feature engineering (also known as feature encoding), a key step in machine learning. It reinforces the idea statistics can be used to make decisions, albeit with some uncertainty, and various errors have different implications (e.g., seeing unflagged spam messages vs. having a non-spam message marked as spam). Choosing an example students have familiarity with may help them feel more confident in their assessments and therefore feel more comfortable and motivated to engage with the activity.

Classifying Spam Using CODAP

One major advantage of the spam MEA is no technology is needed, only paper (though teams typically create a shared Google Drive file to draft and edit their report). There are many ways to incorporate technology, allowing students to further explore classification of subject lines.

We will begin by demonstrating two approaches available within Common Online Data Analysis Platform (CODAP), developed and maintained by the Concord Consortium. CODAP is a free web-based environment for data analysis. This environment is attractive for this application because it is free, has a simple and clean user interface, and runs in a browser. CODAP documents can be distributed to students (or *CHANCE* readers) by clicking "share," which generates a URL allowing another session to be started from a specific working state. In recent years, a powerful plugin for feature creation and text classification has been developed through the "Narrative Modeling with StoryQ" project, an NSF-funded project (see Further Reading).

It is straightforward to load the spam MEA subject lines into a CODAP document as a .csv file (or access our example documents from the column website). Once there, it's possible to create new features that can be part of a rule. Figure 1 displays how a new dichotomous attribute ("dear_or_bless") can be added to the case table, which is true if the subject line includes any of a set of words (in this case, "dear," "bless," "almighty," or "urgent").

A cross-classification (2x2) table consisting of dots can be created using this feature (on the y-axis) and the true status (spam or not) on the x-axis. Figure 2 displays the results when exploring the "Dear or Bless" list.

More sophisticated machine learning approaches are straightforward, using extensions to CODAP created for the StoryQ project. As one example, a "bag of words" (or unigram) approach can be adopted in which every word appearing commonly is added as a dichotomous feature. Figure 3 displays the results once these features are created. For this example, the words "Re," "Dear," "Notification," "Urgent," and "Account" occur four or more times and are included as new dichotomous features.

A logistic regression can then be fit to predict spam or not using those features. Figure 4 displays the results of the model, where each of the features is used as predictors of whether the subject line is spam. Any observation with a predicted value greater than 0.5 is classified as spam.

CODAP also features a "tree builder," which allows the construction of a decision tree to classify spam. Figure 5 displays the simplest decision tree (null model).

A more sophisticated model can be created splitting on whether the words "dear" or "bless" are included in the subject line, as well as whether the string pattern "Re" is observed. Figure 6 displays the tree and the results from three models (null model, only "dear or bless," or the model incorporating "dear or bless" and "contains_re"). Other summaries are available, including the traditional confusion matrix.

Classifying Spam with a Shiny App

A third approach to facilitating exploration of classification of subject lines involves the use of a specially designed Shiny web application. The app has been set up with test and training data and a set of features used to predict whether the subject line is spam or not. Features available in the app include whether the message is all caps, whether it includes a dollar sign, whether it has multiple punctuation marks, whether it includes the string "Dear" or "Mister," or whether it includes religious subject matter.

The user can select which features are to be included and which model (logistic regression, decision tree, or random forest) is to be fit. Tabs allow choice of output, including logistic regression coefficients (see Figure 7), a display of the decision tree (see Figure 8), predicted values from the model, or a summary of the accuracy of the model on the test and training data.

Users of the app can compare how the accuracy changes when more predictors are added. They can explore how accuracy differs on the test and training data. Different directly interpretable models (e.g., logistic regression

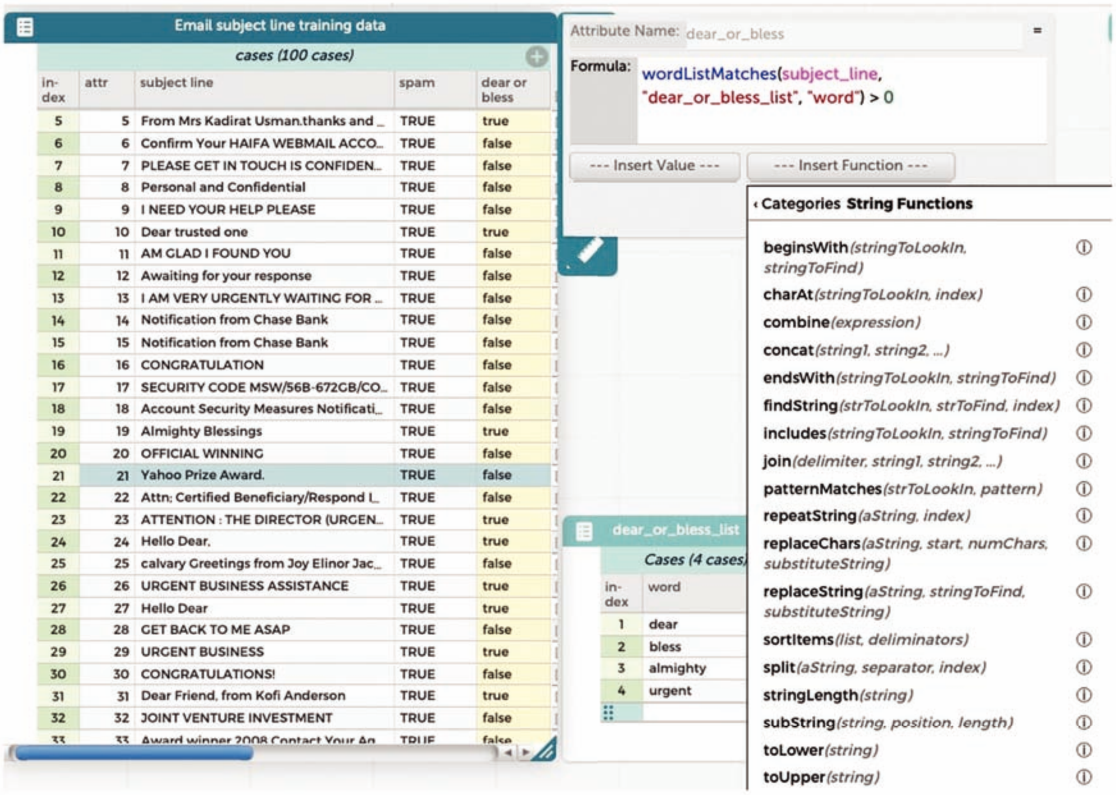

Figure 1. Example of creating a new feature (attribute) using a word list. Similar features can be added to the case table using regular expressions or other string functions.

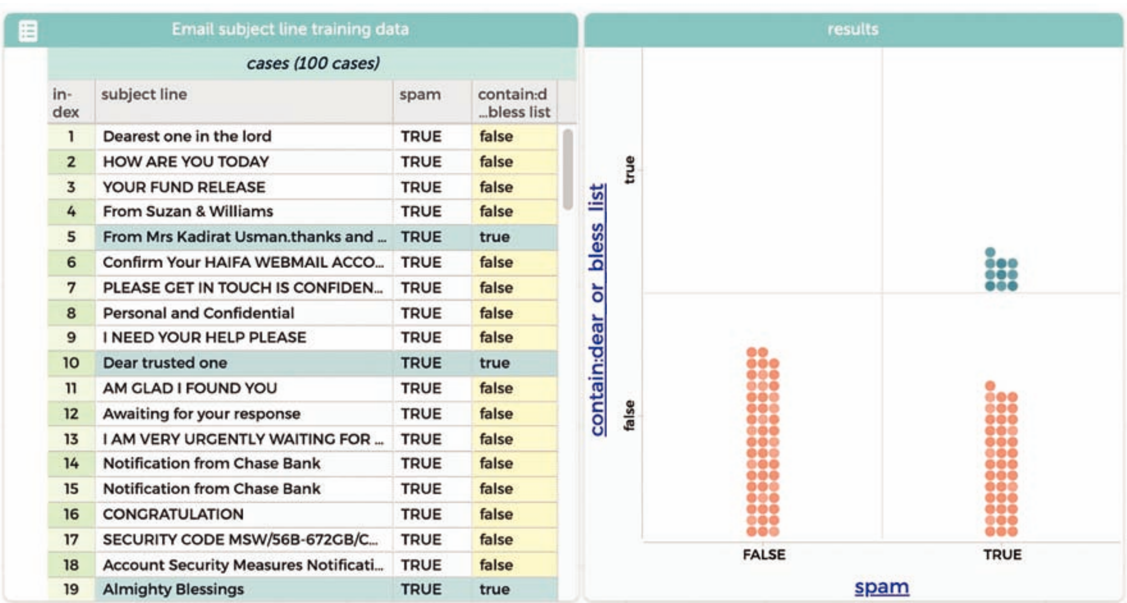

Figure 2. All subject lines matching words in the “Dear or Bless” list were spam. All non-spams are correctly predicted due to the absence of the words “Dear” or “Bless.” In this display, the user has highlighted the spam cell with “Dear or Bless” true, which also highlights those 10 subject lines in the case table.

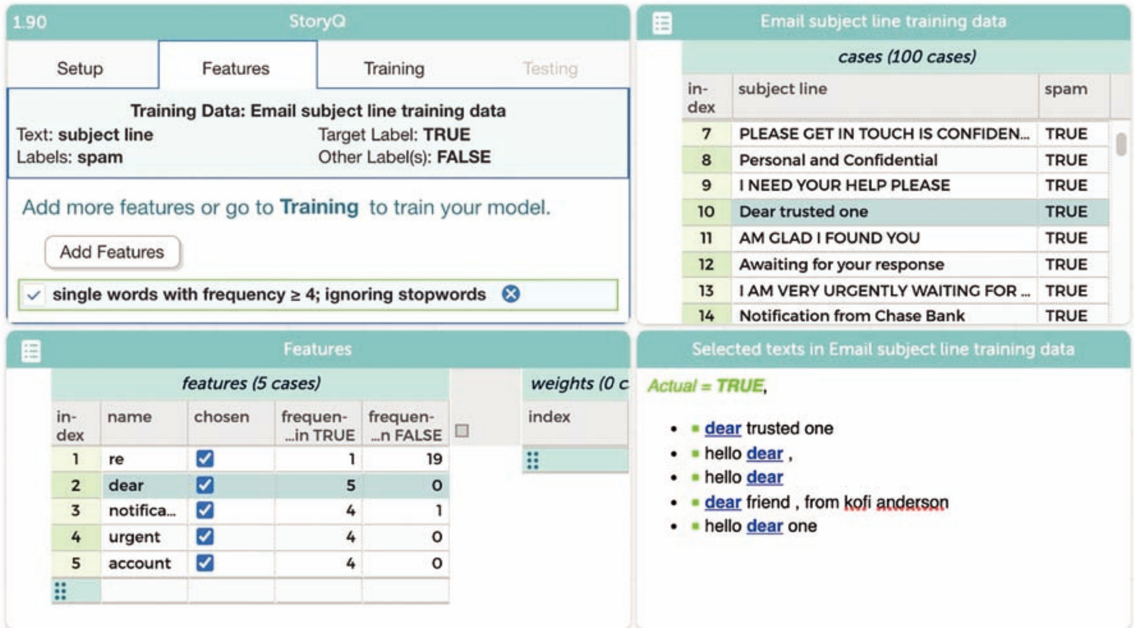

Figure 3. StoryQ within CODAP is used to create features using a "bag of words" for features. Certain words seem to be helpful in classifying the subject lines. For example, all five subject lines that include "dear" were spam.

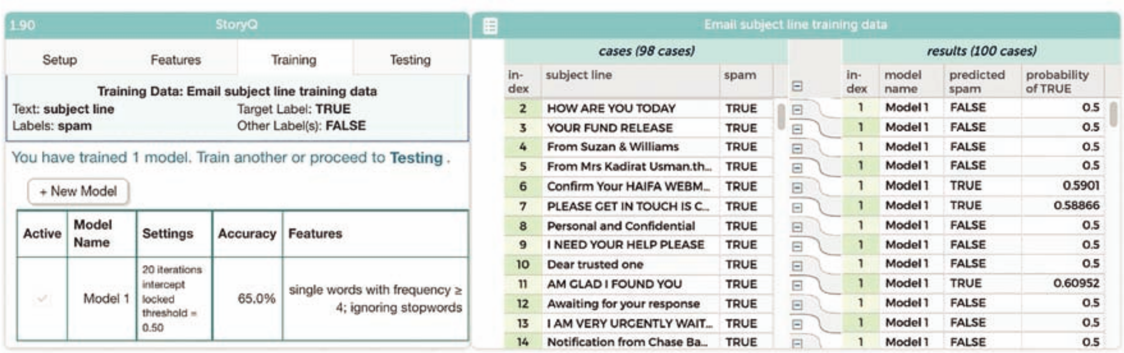

Figure 4. StoryQ within CODAP can facilitate training a model to classify spam. In this example, a logistic regression is used with a "bag of words" feature extraction. Predicted probabilities greater than 0.5 were called spam. The model had modest accuracy (0.65), compared to the accuracy of a null model (no predictors) of 0.5 (since exactly half the data was spam).

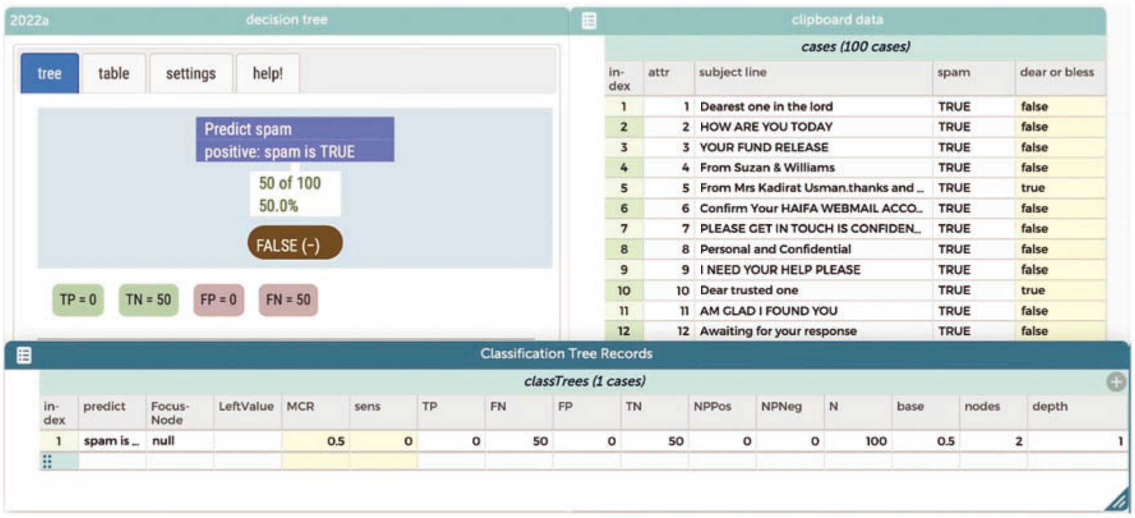

Figure 5. The simplest decision tree has no predictors, with all observations declared to be non-spam. The user can specify the outcome of each terminal node of the tree. Several summaries are included by default, including misclassification rate (MCR), sensitivity, true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN).

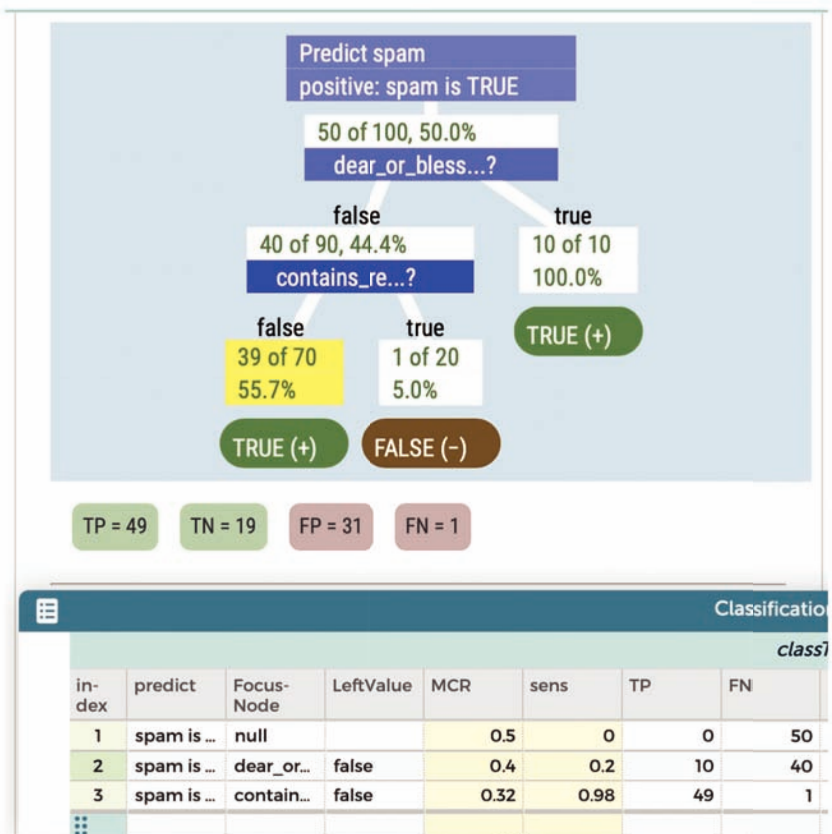

Figure 6. A decision that labels 10 subject lines with "dear" or "bless" as spam, along with 70 that don't include the string "Re." All 20 remaining messages are labeled as non-spam. The model has 68 percent accuracy. Model summaries (including misclassification rate [MCR], 1 - accuracy) are included for three models.

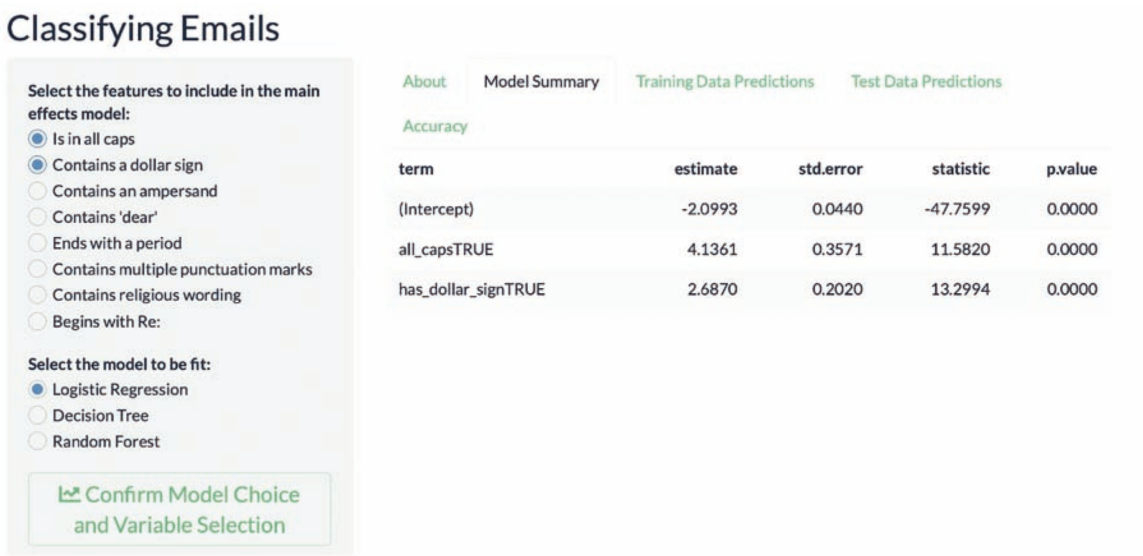

Figure 7. A display from the Shiny app in which several features are used to classify spam subject lines and the logistic regression coefficients from that specified model. Here, two features are included: whether the subject line is all caps and whether the subject line includes a dollar sign.

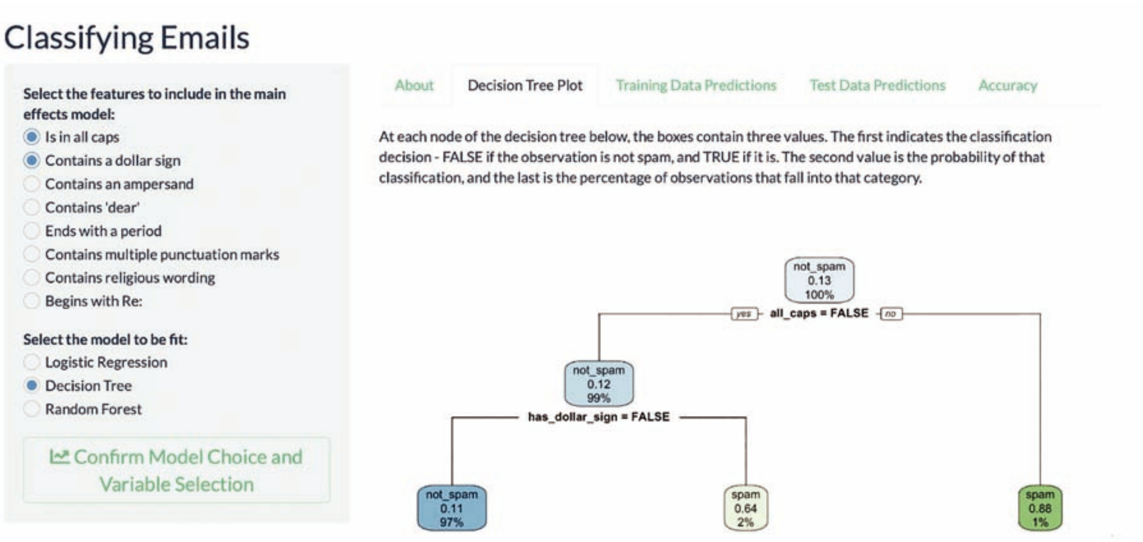

Figure 8. A display of the decision tree: The first split depends on whether the entire subject line is all caps. If TRUE, the message is marked spam. If FALSE, the next split assesses whether the subject line has a dollar sign. If TRUE, the message is declared to be spam.

and decision tree) can be compared. A random forest model can be specified. This ensemble approach sometimes yields better accuracy, albeit with less interpretability. These questions provide a useful motivation for a discussion of advantages and disadvantages of classification models.

While constrained to this example and feature set, the Shiny app allows the user to fit and interpret three powerful and flexible models without having to install R or write code. Several activities might be explored, including calculation of predicted probabilities for a simple logistic regression model or comparison of the predicted probabilities for a logistic regression vs. tree model. No coding is needed for this exploration: As is true for CODAP, all that is required is access to a browser.

An example of the click-bait Shiny app can be found at *nicholasjhorton.shinyapps.io/spam_classifier*.

Classifying Spam Using R

Want more? There are many ways to extend and further develop approaches to classify spam. One of the case studies in Nolan and Temple Lang's *Data Science in R* features an extensive exploration of spam classification. As done in our earlier approaches, they begin by using the subject line and develop and improve a naïve Bayes classifier. Later extensions describe how to ingest other information from the email, such as the header and attachments. They demonstrate ways to use this more extensive feature set, as is often done in commercial spam detection systems.

Conclusion

We provided an overview of four approaches to engage students in thinking about classifying email messages as spam or non-spam, a complex problem with nontraditional data. These approaches varied in their use of technology and student background, but all shared the common goal of using data to make better decisions (and how to judge how accurate those decisions were). We believe text data deserves a larger space in curriculum and offer these approaches as tractable ways to get started.

You can find the data, the code for reproducing the figures we presented, and links to the additional resources we mentioned in the GitHub repository for this column at *bit.ly/taking-a-chance*. **C**

Further Reading

CATALST (Change Agents for Teaching and Learning Statistics) Group. 2009. Creating a spam filter, *https://serc.carleton.edu/sp/cause/mea/examples/example3.html*.

CODAP (Common Online Data Access Platform). 2022, *https://codap.concord.org*.

Concord Consortium. 2021. Narrative modeling using StoryQ, *https://concord.org/our-work/research-projects/storyq*.

Nolan, D, and Temple Lang, D. 2015. Using statistics to identify spam. In Data science in R: A case studies approach to computational reasoning and problem solving. CRC Press, *https://rdatasciencecases.org*.

Spamlaws.com. 2022. Spamlaws: How to stop scams and fraud, *www.spamlaws.com*.

About the Authors

Nicholas J. Horton is Beitzel Professor of Technology and Society (statistics and data science) at Amherst College. He earned his doctorate in biostatistics from the Harvard School of Public Health in 1999 and has co-authored a series of books about data science and statistical computing. He is a member of the ASA Board of Directors and co-chair of the National Academies Committee on Applied and Theoretical Statistics. This work is part of a larger project with this team that stemmed from Horton's work as a Tinker Fellow with the Concord Consortium.

Jie Chao is a learning scientist at the Concord Consortium. She earned her PhD in instructional technology and STEM education from the University of Virginia in 2012. Chao is the principal investigator of multiple NSF-funded projects on innovative approaches to STEM teaching and learning. Her research focuses on designing learning environments, helping students develop computational thinking skills, mathematical modeling competencies, and understanding artificial intelligence.

William Finzer is a senior scientist at the Concord Consortium, where he leads the development of CODAP. He serves as co-principal investigator on the NSF-funded StoryQ, M2Studio, and Boosting Data Fluency projects. Finzer's work centers on bringing data science into the K–12 curriculum and integrated across subject areas through the creation of data exploration software designed to be accessible and usable in the classroom.

Phebe Palmer is a recent graduate from Amherst College, having earned a BA in statistics in 2021. Her research centers largely on STEM education, having assisted with projects focused on approaches to statistics pedagogy, as well as equitable access to STEM curriculum. She works as a research assistant at SageFox Consulting Group, based in Amherst, Massachusetts.