Spam versus Ham:
An Introduction to Machine Learning through Introductory Statistics

A Sentry and Homeland Security Module

Mariah Birgen, Wartburg College
Melanie Brown, Champlain College
Joyati Debnath, Winona State University

MODULE SUMMARY:

Our email inboxes are bombarded daily. Most email clients now have filters to help determine whether an email received is one that was wanted to receive ("ham") or one that was sent automatically ("spam"). However, these filters are not perfect, so sometimes a spam email lands in the main inbox, while an important email might be sent to a spam folder. In this module, students will explore different types of regression to predict whether an email is "spam" or "ham," and then use these same skills to build a model to classify another data set. We present a second data set example that relates to climate change. This regression model is a type of artificial intelligence, and these classifying filters help protect personal security by identifying emails that may contain malicious links or phishing attempts.

TARGET AUDIENCE

This module is written for any Introduction to Statistics course. The material can be covered in 1-2 class periods. We suggest that this module is used after an introduction to Linear Regression. We also note that this module could be expanded and used in some upper-level math/statistics classes related to Multilinear and Logistic Regressions. These regression models are introduced without any exploration of the underlying theory, and the models are used to practice training and testing data. While this module introduces information about Spam and Ham emails and related cybersecurity issues, it does not guarantee protection from the breach of security that can occur while taking all the precautions provided in the module with technology.

PREREQUISITES

Students are expected to:

- Have a fundamental knowledge of basic linear regression from an introductory Statistics class.
- Be comfortable following instructions to run pre-written code.
- Have access to a computer and high-speed internet.
- Create a free account at https://posit.co .

COMPONENTS OF THE MODULE

The module introduces the basics of machine learning by extending basic linear regression included in an Introductory Statistics class with other forms of regression, motivated by classic classification problems. Students do not need any experience with coding to use the self-contained R program that is hosted through Posit Cloud; however, students and instructors will need to create free accounts with Posit to access the program. In the first part of this module, students follow the instructions in the module to execute the code and interpret the output to create a classification model for emails (spam vs. ham). In the second part, students are given a new data set and then adapt the existing code to create a new classification model to determine the efficacy.

Students are expected first to do the following:

a. Follow the instructions given to run Posit code.
b. Interpret code output to answer questions about the data.

Students then repeat these steps on a new data set to:

a. Determine which features in a data set help to create a regression.
b. Partition data into Training Data and Testing Data.
c. Train data into a multilinear regression model, test the model, and then determine accuracy.
d. Train data into a Naïve Bayes model, test the model, and then determine accuracy.
e. Explore how changing the Training/Testing split impacts the model accuracy.

ANTICIPATED NUMBER OF MEETINGS

Depending on student knowledge and course competencies, the module will require between one to two meetings.

LEARNING OUTCOMES

After completing this module, the students will be able to:

- Understand how spam filters work for email.
- Run code to create regression models for classification problems in machine learning.
- Determine how accurate a model is.
- Recognize how different samples can bias the outcomes of data analysis.
- Develop computational, logical, and critical thinking skills.

ACKNOWLEDGEMENT AND DISCLAIMER

This module was developed as a part of the ***Reconnect 2024 Workshop on Artificial Intelligence (AI)*** held at Mendenhall Inn, Chadds Ford, PA, from June 16 – 19, 2024 organized by the Department of Homeland Security (**DHS**), the Center of Discrete Mathematics and Theoretical Computer Science (**DIMACS**), and **SENTRY** (**S**oft-target **E**ngineering to **N**eutralize the **T**hreat **R**ealit**Y**) and funded by National Science Foundation (**NSF**).

The authors would like to thank Dr. Midge Cozzens for her valuable suggestions and continued support. In addition, the authors want to thank the speakers Dr. Vivek Singh, Professor in the School of Communications and Information at Rutgers University; Dr. Sinan Ozdemir, an AI expert, inventor on several AI patents and current AI consultant; and Dr. Donna Beers, Professor at Simmons College for valuable information on AI and Security measurements.

I. Introduction to Artificial Intelligence and Linear Regression

Artificial Intelligence (AI) can be described in many ways. Some say it is a technology that generates, classifies, and performs tasks like humans only faster and more efficiently. Others say AI enables computers and machines to simulate human learning, comprehension, problem solving, decision making, creativity and autonomy. Yet, others describe it as the simulation of human intelligence processes by machines, especially by computer systems.

In AI, machines learn from repeated experiences and then adapt those experiences to new inputs to execute tasks, which is similar to human capabilities. Without actively thinking about it, we use past

experiences and information to predict what should happen next. There are a variety of AI models that do the same thing. Some use probabilities to predict what happens next, whereas others use *regression* to predict what happens next.

In linear regression analysis, the goal is to use one variable to predict another variable using a relationship described by a linear function. The variable we use to predict is the *independent variable x*, and the variable that is being predicted is the *dependent variable y* since its value depends on the independent variable.

Sometimes the relationship between the two variables doesn't fit a linear function, so we can look to other functions to be a better *fit*. Other times, we have data with more than one independent variable. When we use multiple independent variables to predict a dependent variable, we have *multiple regression*.

When we talk about artificial intelligence, we are talking about this same process, that is using multiple variables to make a prediction. This module will focus on *machine learning*. Machine Learning is an area of artificial intelligence that uses algorithms to teach machines to learn from data and make decisions without needing specific programming. A given data set is split into two parts: a Training Set and a Testing Set. The Training Set is given to the machine to "learn." This mean that the model is identifying patterns in the data and trying to determine how different independent variables correspond to the dependent variable. Once the model is "trained," we then give the machine the Testing Set to see what predictions the model makes based on what it "learned." Of course, we know what the right answers should be, so we can assess how accurate the predictions are.

In this module, we are going to work on *classification problems*. A classic example of this type of problem is "Spam vs. Ham." Your inbox is bombarded daily with emails. Some are emails you want to receive ("ham"), but most are emails you never asked for ("spam"). Spam emails also present security risks since they often include dangerous links from criminals trying to steal your personal data to access credits cards and bank accounts. Other spam emails have attachments that contain *malware*.

Most email clients now include a spam filter to automatically divert spam emails into a spam folder. The spam filter is a classification problem: given an email, how does the filter know whether it is spam or ham?

II. Linear Regression

Linear Regression is a type of machine learning that learns from the labeled datasets and creates a linear function that can be used for making predictions with new datasets. Labeled datasets have classified target values that are already known and then regression can predict continuous outputs based on the independent input variables. The goal is to find the *best fit linear equation* that can predict the values of the dependent variable based on the independent variables. The best fit linear equation should minimize the error between the predicted value and the actual values when it occurs.

There are two main types of linear regression:

- Simple Linear Regression involves one independent variable and one dependent variable, and the equation created looks like

$$y = \beta_0 + \beta_1 x,$$

where y is the dependent variable, x is the independent variable and β_0 is the y-intercept and β_1 is the slope.

- Multiple linear regression uses more than one independent variable and only one dependent variable and the equation created looks like
$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_n x_n,$$
where y is the dependent variable, x_i's are the different independent variables, β_0 is the y-intercept and β_i's are the slopes.

There are also few restrictions in applying linear regression, depending on the data given. Linear regression assumes a linear relationship between the variables, which makes it sensitive to outliers. Furthermore, linear regression is prone to overfitting or underfitting which can lead to poor prediction.

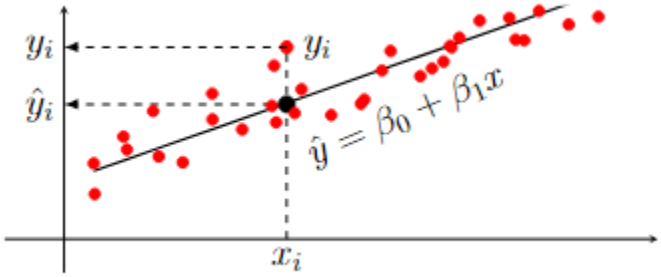

Figure 1: A depiction of a best-fit linear regression for the given data set.

In Figure 1: A depiction of a best-fit linear regression for the given data set. we see a scatterplot of paired data, and the regression line
$$y = \beta_0 + \beta_1 x.$$
For the indicated value x_i, the actual paired value is y_i. However, the regression line has a predicted value of
$$\hat{y_i} = \beta_0 + \beta_1(x_i).$$
In this case, the predicted value is slightly less than the actual value. For a Spam Filter to be really effective, the predictions for which emails are spam should be pretty good. In other words, we want a regression line that is a really good fit.

III. Using R and Posit

In this module, we will be using the programming language R to find the coefficients in our regression models. However, you do not need to be able to program in R (or program at all!) to take advantage of its capabilities. We have a free R notebook available at https://posit.cloud/content/8964873 that trains and tests data to make predictions. To get started with this notebook, you will need to make a free Posit account OR sign in with a Google account. Accessing this notebook creates a temporary copy on your workspace, so you can make changes without affecting the original notebook we provided.

Your free account allows you 20 hours to run code on Posit. This timer runs whenever you have a notebook in Posit open, so we encourage you to close Posit when you are not actively working on this module.

Getting Started: Once you log in (and it may take a minute or so, depending on your internet speed), you will see the screen shown in Figure 2.

In the top menu bar, go to File → Open File and select "File.Rmd." You can also access this file directly by double-clicking the file name in the lower right window. This now opens a new window in your workspace up at the top. You can hover your cursor over the borders to change the sizes. In the next set of screenshots, we have minimized the Console part of the workspace. We will focus on the window that says File.Rmd, as indicated by the arrow shown in Figure 2: .

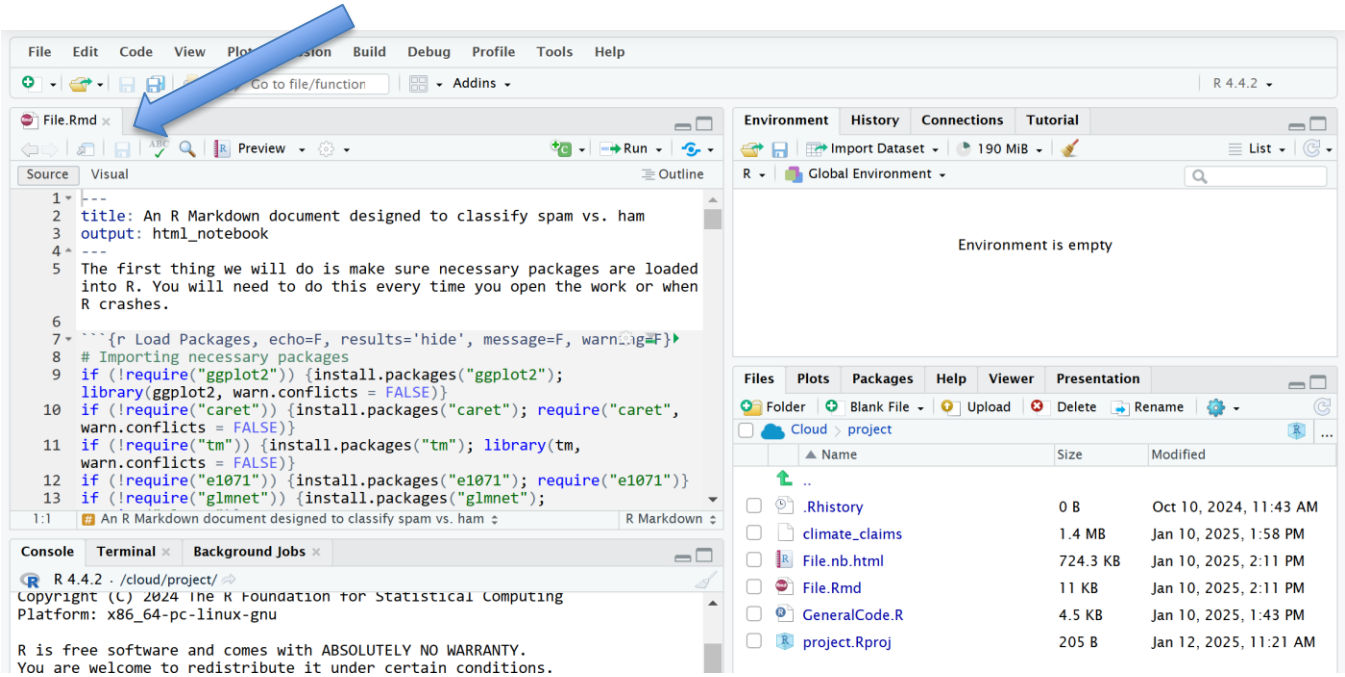

Figure 2: Location of file in Posit Window

This new File.Rmd window is where we will be training and testing data to create our spam vs. ham classification model. This window has some text with a gray background, which is the actual R code, and the other text that has a white background, which is the comments describing what the code does.

In Exercise 1, you will be working with data that is accessed from https://raw.githubusercontent.com/mbirgen/reconnect2024/refs/heads/main/emails.csv/emails.csv . This data set was originally accessed from Kaggle (https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv), but the Github hosting improves the efficiency of the Posit notebook. In Example 2, you will be making slight modifications to this code to test and train on a different data set.

Although you will not need to write any code, you will need to run this code in chunks by clicking on the green triangle in the top-right corner of each chunk. The code also has reference line numbers. In the image in Figure 3, the arrow is pointing to Line 7.

The code in the notebook will help you import and clean up the data. To create our model, we will need to separate our data into a Training Set and then a Testing Set. The Training Set is used to create the regression model, and then we test the model using the Testing Set. We can determine how good the model is at classifying emails as spam or ham based on how close the actual Testing Set values are to the predictions the model gives.

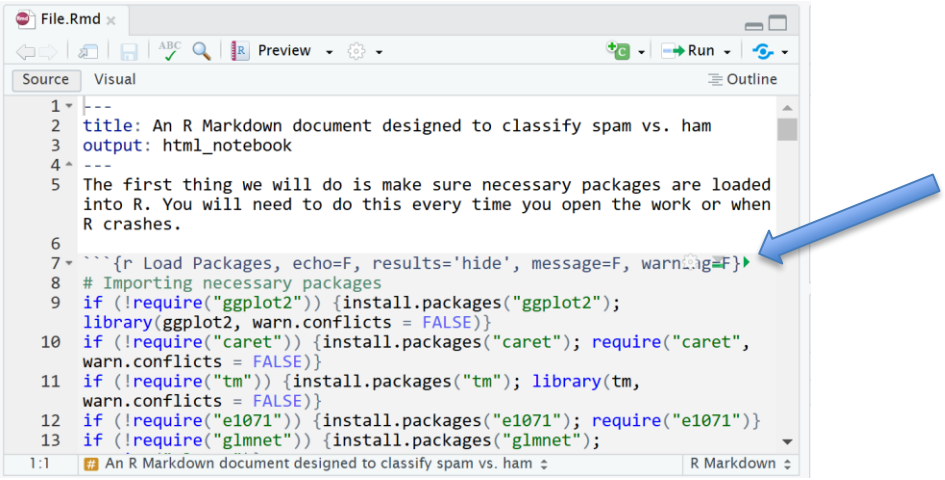

Figure 3: The green triangle for running code. This example triangle runs the code block that begins on Line 7.

Click on the green triangle shown in Figure 3: The green triangle for running code. This example triangle runs the code block that begins on Line 7. to load any missing packages (also known as libraries). You will need to do this each time you open the notebook. When any missing packages are loaded, you'll see text in red, like in Figure 4. This is NOT an error message, just a warning, so do not worry.

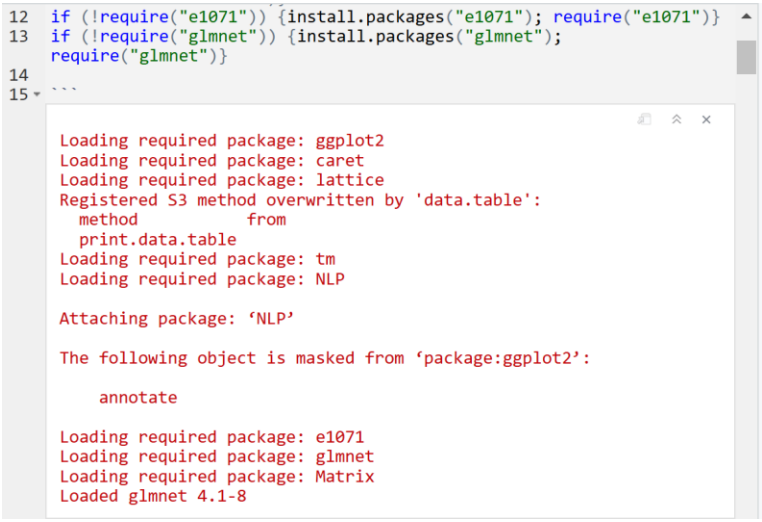

Figure 4: Missing package warning notifications

When we evaluate the chunk of code that starts on Line 23, we import the data set. We can see that this happened by looking in the upper left window where we now have Data as shown in Figure 5: The data window in Posit. We imported data with 5728 observations and 2 variables. Each observation is an email, and the variables are "text" and "spam." The variable X represents text, and the variable Y represents spam. The "spam" variable gives the classification of the email as 0 if the email is not spam and 1 if the email is spam. This variable answers the question "Is this email spam?" with a true (1) or false (0).

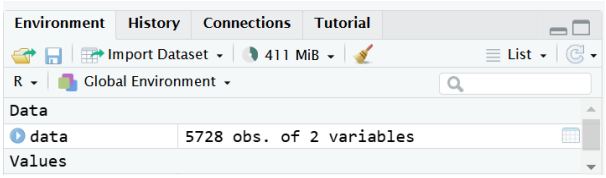

Figure 5: The data window in Posit

Now that you are getting comfortable working with this program, we are going to start working with this data set to create a regression model to predict whether a given email is ham or spam.

Exercise 1: Spam vs. Ham

1. Run the code in Line 7 to import any missing libraries and then run the code in Line 19 to download the data.
2. Run the code in Line 31 to get information on the structure of the Data.
    a. What is the first subject line you see?
    "Subject: naturally irresistible your corporate identity lt is really hard to recollect a company : the market"
    b. Based on the subject line, would you classify that email as spam or ham?
    Spam
3. Run the code in Line 35 to see the data in the variable "text" for each of our emails. Now that you see the full text of this email, would you classify it as 0 for false (or ham) or a 1 for true (spam). Explain your answer in a few sentences. What features in the email text influenced your decision?
    Answers may vary, but this is most likely spam, so we would classify it as 1. There are many typos in the email, and the phrasing does not read like the author was a fluent English-speaker.
4. Run the code in Line 41. Did your hypothesis in Question 3 match the data?
    Yes. The actual data has a classification value of 1, indicating the email is spam. Note that `[1]` in the code refers to Observation 1, whereas the `1` afterwards is the spam label value.
5. Run the code in Line 47 to create a bar graph of the spam and ham emails in the data set.
    a. Based on this bar graph, estimate the number of spam emails in the data set.
    Answers may vary, but there are around 4400 spam emails.
    b. What percentage of the data is spam?
    Since there are 5728 observations, we can take 4400/5728=0.768 to estimate that about 77% of the data is spam.
    c. Based on this bar graph, estimate the number of ham emails in the data set
    Answers may vary, but there are around 1400 ham emails.
    d. What percentage of the data is ham?
    Since there are 5728 observations, we can take 1400/5728=0.244 to estimate that about 24% of the data is spam.
6. Run the code in Line 53 to generate a frequency table. Now that you have exact figures, what percentage of the data appears to be spam?
    The actual values are 4360 spam emails and 1368 ham emails. We can correct our work in Question 5 and obtain 4360/5728=0.761 to see that approximately 76% of the data is spam.
7. Run the code in Line 61 to determine how many duplicates are in the data set. How many duplicates are there?

There are 33 duplicates in the data set.

8. Run the code in Line 65 to remove the duplicates. Look in the upper right where the data information is shown. How many observations are now in our data set?
   There are now 5695 observations in the data set.
9. Run the code in Line 70 to check that all duplicates are removed. What do you see in the output that lets you know the duplicates are removed?
   The same output that showed 33 duplicates when we ran Line 61 now shows a value of 0. This means that there are no duplicates remaining.
10. Run the code in Line 76 to split the data into X (the text) and Y (the classification that answers the question "Is this spam?"). You will see X and Y appear in the upper right corner where we were seeing the data summaries. The X data is the 5695 emails, and the Y data is the classification labels of 0 or 1.
11. Run the code in Line 83 to take a smaller sample of the original data to ensure that the notebook will run for you in the free version of Posit. You will see X_small and Y_small appear in the upper right corner.
    a. How many values are in X_small?
       There are 570 observations.
    b. What percentage of the original data does this represent?
       Approximately 10%. We can see this in the code in Line 86 as `p=0.1`.
12. Run the code in Line 95 to partition the data in our smaller sample into a Training Set (called data_train_small) and a Testing Set (called data_test_small). In this notebook, we will put 80% of the sample in the Training Set, and the remaining 20% into the Testing Set.
    a. How many observations are in the Training Set? Does this align with the percentage designated in the code?
       There are 456 observations, and we can compute 456/570=0.8 to see that this is indeed 80% of our sample size.
    b. How many observations are in the Testing Set? Does this align with the percentage designated in the code?
       There are 114 observations, and we can compute 114/570=0.2 to see that this is indeed 20% of our sample size.
13. Run the code in Line 118 to check to be sure that the size of our X Training Set matches the size of our Y Training Set. Do the numbers match?
    Yes!

We are now going to train and test two regression models. We will first use a Logistic Regression Model and then use a Naïve Bayes Model. Logistic regression is a type of multilinear regression, and Naïve Bayes is based on a more modern interpretation of statistics.

14. Run the code in Line 138 to further clean and process the data in preparation for the training and then train and test the Logistic Regression Model. The value displayed in Line 163 is the accuracy of the model that represents the percent of predictions that were correct. It is the result of the formula

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total sample size}}$$

    a. What is the accuracy of this model? Explain what this value means in a sentence or two.
       The accuracy is 0.6315789. This means that approximately 63% of the predictions were correct when classifying an email as spam or ham.

b. How well does the model predict spam? Is this good enough for an automatic spam filter?
Answers will vary. The authors of this module do not think this model performs well at all. The model is correct less than 2/3 of the time, which means over 1/3 of the emails will be incorrectly classified.

15. Run the code in Line 166 to train and test the data in a Naïve Bayes Model. The value displayed on Line 178 is the accuracy of this model and follows the same formula.
    a. What is the accuracy of this model? Explain what this value means in a sentence or two.
    The accuracy is 0.7568042. This means that approximately 76% of the predictions were correct when classifying an email as spam or ham.
    b. How well does the model predict spam? Is this good enough for an automatic spam filter?
    Answers will vary. The Naïve Bayes Model performs slightly better than the Logistic Regression, but it is still wrong almost ¼ of the time.
    c. In Question 11, we determined that our sample was only 10% of the original data set. Would changing the sample size improve the accuracy of the models? Explain your conclusion in a few sentences.
    Yes. Having a larger sample would most likely improve the accuracy.
16. To complete this exploration, run the code on Line 184 to create a function that tests particular text on the two models we created.
17. Run the code on Line 204 to test the text "Hey i am Elon Musk. Get a million dollars free in prize" on the logistic model we created.
    a. How did the logistic regression (mlr) classify this text?
    Spam. The output was "1" and that is the label we are using for spam.
    b. Is this prediction correct?
    Most likely yes. It is highly unlikely that Elon Musk is emailing a random individual to offer $1 million.
18. Run the code on Line 212 to test the text "Hey i am Elon Musk. Get a million dollars free in prize" on the Naïve Bayes model we created.
    a. How did the Naïve Bayes regression (nb) classify this text?
    Spam. The output was "0" and that is the label we are using for ham.
    b. Is this prediction correct?
    Most likely no. It is highly unlikely that Elon Musk is emailing a random individual to offer $1 million, so we would expect this to be labeled as spam.
19. Run the code on Line 221 to test the text "thanks for your response" on the logistic model we created.
    a. How did the logistic regression (mlr) classify this text?
    Spam. The output was "1" and that is the label we are using for spam.
    b. Is this prediction correct?
    Without seeing the rest of the email, it's hard to say.
20. Run the code on Line 230 to test the text "thanks for your response" on the logistic model we created.
    a. How did the Naïve Bayes regression regression (nb) classify this text?
    Spam. The output was "0" and that is the label we are using for ham.
    b. Is this prediction correct?
    Without seeing the rest of the email, it's hard to say.
21. [Optional] Experiment with different phrases in the previous code blocks to see how each model classifies them. Are these predictions correct?

Exercise 2: Climate Change

Now that you are comfortable working with Posit, running code, and interpreting output, you are going to follow a similar approach to create a classification model on climate change data. The following exercises could be applied to any dataset that classifies each observation with only 2 labels. We are using the CLIMATE-FEVER data [1] from https://www.sustainablefinance.uzh.ch/en/research/climate-fever.html, but the data that you will be using in Posit has already been cleaned and processed. The classification model you will be creating is based on data pulled from Wikipedia about claims that events were related to climate change. Each observation in the original data was classified by climate scientists into one of the four categories of "Supports," "Refutes," "Disputed," or "Not Enough Information." To simplify our analysis in this module, the data has been filtered to only include instances of "Supports" and "Refutes" to answer the question "Is this claim related to climate change?"

1. How many observations are in this data set? How many variables (columns) are included?
   907 observations with 53 variables.
2. A word occurring right after the $ symbol is the column label. Which columns in the data set will be helpful in running the regression?
   We need the `claim` and `claim_label` in order to create our model.
3. Separate the data into the Training Set and the Testing Set. What percentage of the data are we including in the Training Set?
   We can see in Line 251 that we are using p=0.8, so we are taking 80% of the data in our Training Set.
4. Train the Naïve Bayes Model. What is the accuracy of this model? Explain what this means in a sentence or two.
   The accuracy is 0.5138122, so approximately 51% of the claims are classified correctly.
5. Train the Multilinear Regression Model. What is the accuracy of this model? Explain what this means in a sentence or two.
   The accuracy is 0.519337, so approximately 52% of the claims are classified correctly.
6. How good are these models overall? Which one seems to be more accurate?
   Based on the particular training and testing split we did, the two models are pretty similar. Neither is much better than a coin-flip at correctly predicting.
7. Go back to Line 275 and change the seed value to something other than 0. Re-run your code to train a different Naïve Bayes Model and a different Multilinear Regression Model. Does your answer stay the same?
   As an example, changing the seed to 28 results in accuracy decreasing to around 44% in each model.

IV. Conclusions

"We are not wet computers. The computer is making decisions using statistics and not actually 'thinking.' Take linear algebra and more statistics!"

Machine learning and deep learning algorithms can analyze transactional patterns and can flag anomalies. Like unusual languages in the email, login locations, and fraudulent behaviors. Machine learning and deep learning models can analyze data from sensors, Internet of Things devices, and Operational Technology to forecast when something unusual event occurs. This module demonstrates that these techniques are not fool-proof. Students are able to create toy examples of these algorithms based on their knowledge from their coursework and can explore the accuracy of the models.

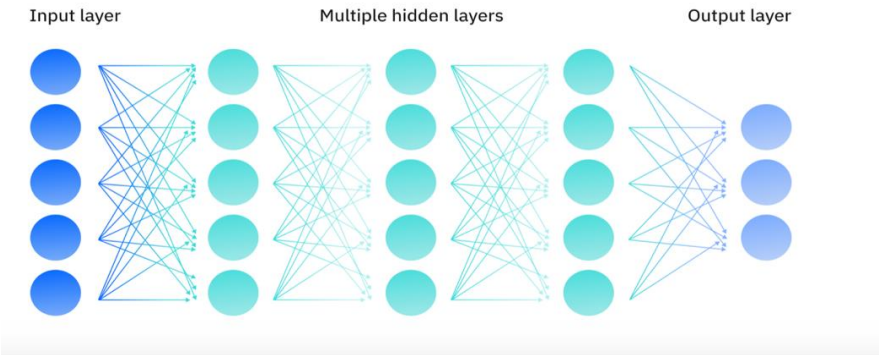

Figure 6: Diagram of more sophisticated Machine Learning

Figure 6: Diagram of more sophisticated Machine Learning is a conceptual diagram of machine learning [2,3]. Machine learning works by training on the input data, then tuning from multiple hidden layers, and then generating output content. Your work in the Posit notebook followed this same process, but there were no hidden layers. You worked solely with inputs and outputs.

In addition to what you discovered about models not being fool-proof, it is also important to recognize that there are other potential issues with AI models. Since each model relies on data sets, it is vulnerable to data poisoning, data tampering and data breaches. Data poisoning involves introducing false data into a system to degrade its accuracy or functionality. Data tampering is the unauthorized alteration of data to mislead or manipulate results. Data breaches occur when sensitive information is accessed or stolen by unauthorized individuals. Watch out for threat actors who target AI models to steal and do unauthorized manipulations. If left unmonitored, AI systems can risk committing privacy violations and producing biased outcomes.

V. References

[1] Diggelmann, Thomas; Boyd-Graber, Jordan; Bulian, Jannis; Ciaramita, Massimiliano; Leippold, Markus (2020). CLIMATE-FEVER: A Dataset for Verification of Real-World Climate Claims. In: Tackling Climate Change with Machine Learning workshop at NeurIPS 2020, Online, 11 December 2020 - 11 December 2020.

[2] Rajan, Ranjitha & Kumar, S N. (2022). IoT based optical coherence tomography retinal images classification using OCT Deep Net2. Measurement: Sensors. 25. 100652. 10.1016/j.measen.2022.100652.

[3] PAC World. (2024, September 22). *AI in electric power systems protection and Control*. https://www.pacw.org/ai-in-electric-power-systems-protection-and-control

[4] *What is (AI) Artificial Intelligence? | Online Master of Engineering | University of Illinois Chicago*. (n.d.). https://meng.uic.edu/news-stories/ai-artificial-intelligence-what-is-the-definition-of-ai-and-how-does-ai-work/

Appendix: Using Posit for Instructors

If this is your first time using RStudio, this appendix is for you.

I. How do I run Posit Code?

The document consists of text, as well as code chunks. Each code chunk is surrounded by ```{r*} Code Chunk* ``` The background is also lightly gray.

There are multiple ways of running code.

- The method suggested in the module is to click on the Green Arrow ( ).
- There is a Drop-Down Menu, seen in *Figure 7: Run Drop Down Menu* called Run (Run) that gives you multiple options that you can choose from.

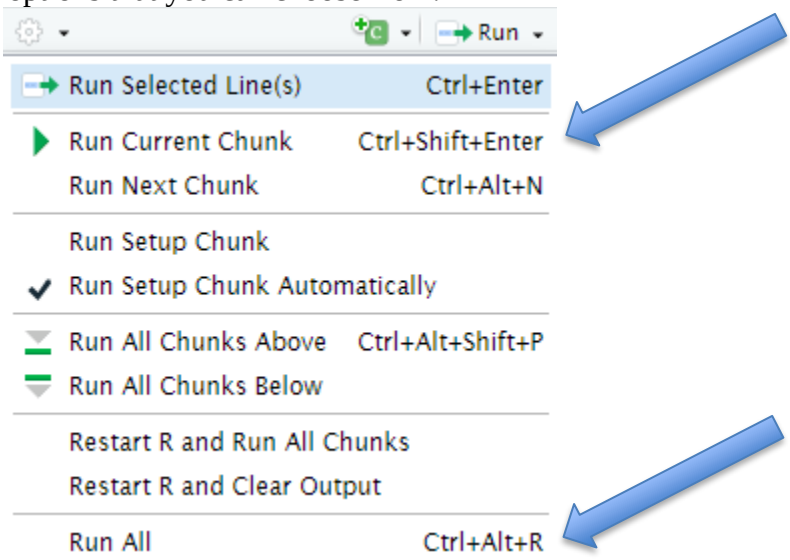

Figure 7: Run Drop Down Menu

- Here in Figure 7: Run Drop Down Menu, you can see that, in windows, *control-shift-enter* is the same as clicking .
- As an instructor, you may just choose to Run All so that everything will be loaded in your notebook, and you can get all the answers.

II. How do I reset Posit Cloud?

After you have been working with the web browser, it is no longer going to look like what a new user will see. Clicking on the Environment tab will show you things that have been added to the memory of the system. Even when you log out and log back in again, these items are retained.

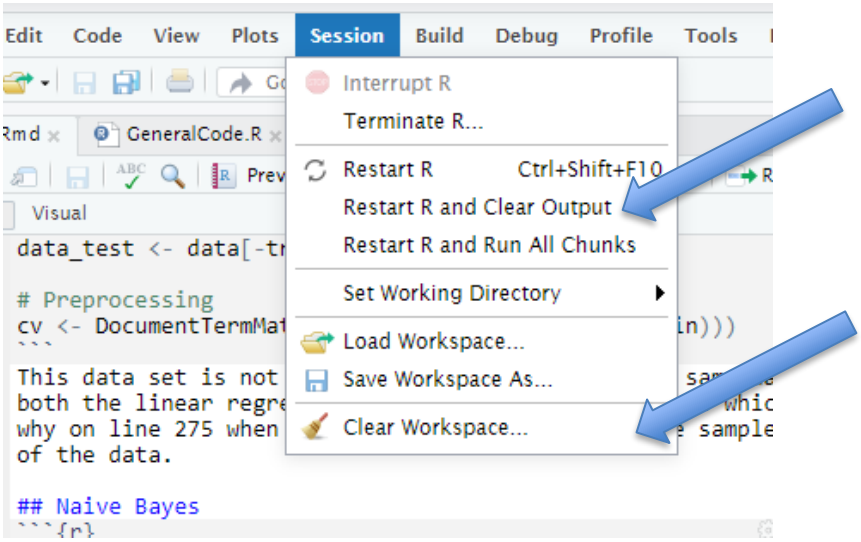

Figure 8: Clearing the Session

As shown in *Figure 8: Clearing the Session* under the session tab you need to do two things. First you need to **Clear Workspace** which is at the bottom of this pull down menu. Second, you need to **Restart R and Clear Output** which is about halfway down the pull-down menu. This will make this workbook look like what your students see when they first open it.

III. Other Notes

- If a student clicks "Save a Permanent Copy," that's okay. It means the student has saved a local copy, but any changes in their local copy will not affect the notebook we have shared with you.
- When running the Naïve Bayes code on Line 230, you or your students may see the warning message in Figure 9. When the authors were testing the code, some saw the warning and some did not. Do not worry. The important information for your students to see is the `[1] 0` that indicates that the model predicted that the email is ham.

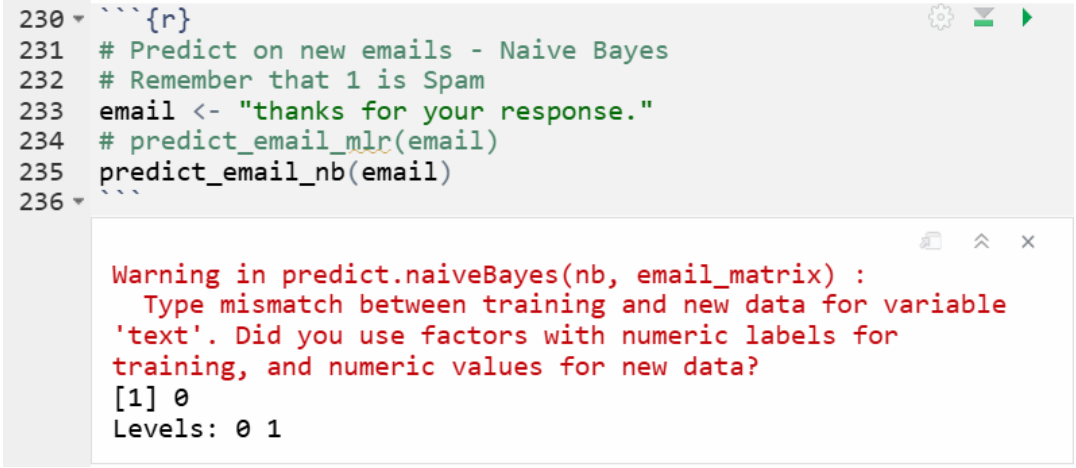

Figure 9: Warning Message for Naive Bayes