Computer Science Department - Rutgers University **Fall 2022**

Homework 1

16:198:461

1 Stats and Data

You want to try to determine who is going to win an upcoming election. To do so, you're going to poll some number of people to try to estimate the probability a given voter supports Candidate A over Candidate B (we will assume no one is indifferent between them).

1) If the probability a person supports Candidate A is p, how many people should you poll to try to be 95% confident you know the value of p to within an error of 0.01?

2) What if you can only afford to poll $k = 30$ people - how accurately can you say you know p (i.e., what error do you know p to) with 95% confidence?

3) Generate synthetic data by sampling a Bernoulli(0.6) distribution $k = 30$ times. What did you get for $\hat{p}$, and how does it compare with $p = 0.6$? Does this seem consistent with the previous answers?

However, note that the probability a person supports a candidate is not the probability that the candidate will win.

4) If N people vote in the election, each with a probability p of voting for Candidate A, then the number of votes Candidate A receives will be random. What is the distribution of the number of votes Candidate A receives?

5) Assume 1000 people vote. If $p = 0.6$, as above, what is the probability that candidate A wins the election? (i.e., receives a majority of the votes). Be clear on how you are calculating this.

6) If you estimate p with $\hat{p}$ based on your data, what is the probability that candidate A wins the election? Does this seem consistent with the true probability?

If we wanted to be a little more accurate (intellectually honest?) in our estimates, we wouldn't want to use $\hat{p}$ by itself to estimate p, since it is unlikely that $\hat{p}$ is *exactly* p. Using the *Central Limit Theorem* we can estimate our knowledge of p as

$$p \sim \mathcal{N}\left(\hat{p}, \frac{\hat{p}(1-\hat{p})}{k}\right), \tag{1}$$

i.e., that p is most likely to be near $\hat{p}$, but might be above or below it, with variance given by $\sigma^2 = \hat{p}(1-\hat{p})/k$.

7) Generate a 'guess' for the value of p by sampling a $\mathcal{N}\left(\hat{p}, \frac{\hat{p}(1-\hat{p})}{k}\right)$ distribution. Using this guess at p, compute the probability that Candidate A wins the election. Do this 1000 times, and average the probabilities of Candidate A winning, to a final estimate for the probability that Candidate A wins. What do you get, and how does it compare with just using $\hat{p}$ by itself?

> For this problem, you may have to look up how to generate normal random values in your favorite programming language. Be cautious as to whether the function asks for the variance or the standard deviation as a parameter. Additionally, since normals can be any real value, but we only want p between 0 and 1, you should reject any generated p that falls outside the interval $[0, 1]$, though there should not be many.

Bonus) In the 1948 US Presidential election, many papers and pollsters called the election for Thomas Dewey, but the results later showed Truman won by a significant margin. This is the source of the infamous, incorrect headline 'DEWEY DEFEATS TRUMAN'. Why were the predictions so wrong? Be thorough.

2 Regression Comparison

Generate synthetic data, each data point of the form $\underline{X} = (X_1, X_2, X_3, \ldots, X_d)$ (with d as given in a moment) in the following way:

- $X_1 \sim \mathcal{N}(3, 1)$
- $X_2 \sim \mathcal{N}(-2, 1)$
- $X_3 = X_1 + 2X_2$
- $X_4 = (X_2 + 2)^2$
- $X_5 \sim \text{Bernoulli}(0.8)$
- $X_6, X_7, \ldots, X_d \sim \mathcal{N}(0, 1)$

For each data point, take Y as

$$Y = 4 - 3X_1^2 + X_3 - 0.01X_4 + X_2 * X_5 + \mathcal{N}(0, 0.1), \tag{2}$$

where the last term indicates a random error added on to the Y value for each data point. Note that for data generated in this way, Y will be independent from X_6 through X_d - they provide no useful information about Y. Generate a dataset of size 10000 for your training data, and size 1000 for your testing data.

In this problem, we'll look at three different models to try to predict the Y value from the X-values.

1) If you had to model Y as a constant value, i.e., $f(\underline{x}) = c$, based on your data, what value c should you pick? Why? What is the error for the best c for your data? How does the error on the training vs testing set differ? Does the value of d matter? Why or why not.

A constant model like this can provide a good benchmark to make sure that more complicated models are actually doing something useful.

2) Write a program to take a data set and fit a decision tree to it. For $d = 10$, generate a plot of the decision tree error on the training and testing data as the depth of the tree you're fitting increases. What appears to be the optimal depth to grow the tree to to minimize the error? How does the performance of this tree compare to the performance of the constant model?
3) Repeat the experiment, but instead of truncating the tree by depth, truncate by sample size (i.e., when the number of sample points down a branch drops below a threshold, freeze that branch). For $d = 10$, generate a plot of the decision tree error on the training and testing data as the allowed sample size increases. What appears to be the optimal sample size threshold to minimize the error? How does the performance of this tree compare to the performance of the constant model?
4) Which is better for minimizing error? Truncating by depth or truncating by sample size?
5) Consider repeating this experiment but now with $d = 50$. Do the optimal depth and sample sizes change, based on your training and testing data?
6) Consider repeating the above experiments for different values of d. Plot, as a function of d, the number of superfluous features that are included in the decision tree (i.e., the number of variables $X_6, \ldots, X_d$ that are included in the decision tree). Which approach is better for excluding independent features?

For the final model, we'll consider a linear model, i.e., a function of the form

$$f(\underline{x}) = w_0 + w_1 x_1 + w_2 x_2 + \ldots w_d x_d \tag{3}$$

7) Write a program to take a data set and fit a linear model to it. For $d = 10$, give the coefficients for your fitted model. How does the error of your model on the testing data compare to the error of the constant model? Is overfitting an issue here?

8) Consider the following scheme to try to eliminate superfluous features: when you fit a model, look at the weight on each feature. If for feature i, $|w_i| \leq \epsilon$, eliminate that feature from consideration. Whatever features remain, re-fit a linear model on those features. For $d = 50$, plot the number of superfluous features that make it into the final model as a function of ϵ, and plot the error on the testing data for the final model as a function of ϵ. Is this a good strategy?

9) Which model is superior here? Why?

Bonus) Repeat these experiments for $d = 6$, but expanding the data set to include quadratic features, i.e.,

$$\underline{X} = (X_1, \ldots, X_6, X_1^2, X_1 * X_2, \ldots, X_1 * X_6, X_2^2, X_2 * X_3, \ldots, X_2 * X_6, \ldots, X_6^2) \tag{4}$$

How do the models compare now? Is overfitting an issue? Notice that for a linear model on this data, it should be able to fit Y perfectly except for the random noise. Which model is superior here?