HW1

1 Stats and Data

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?

By Heoffding Inequality,

$$P(|\bar{X}-p| \geq 0.01) \leq 0.05 \quad \Rightarrow \quad 2e^{-2(0.01)^2 n} \leq 0.05$$

$$n \geq \frac{\ln\left(\frac{0.05}{2}\right)}{-2(0.01)^2} = \frac{\ln(0.025)}{-0.0002} = 18444.44$$

Therefore, at least poll 18445 people.

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?

By Heoffding Inequality,

$$P(|\bar{X}-p| \geq \varepsilon) \leq 0.05 \quad \Rightarrow \quad 2e^{-2(\varepsilon)^2 \cdot 30} \leq 0.05$$

$$\varepsilon \geq \sqrt{\frac{\ln\left(\frac{0.05}{2}\right)}{-2 \cdot 30}} \approx 0.247954$$

Therefore, have 95% confidence that the value of p is within an error of 0.247954.

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?

Let x_i be each of the Bernoulli result. Generate by Python / Jupiter Notebook:

$$\hat{p} = \frac{1}{30} \cdot \sum_{i=1}^{30} x_i = \frac{21}{30} = 0.7, \qquad \varepsilon = |\bar{X} - p| = 0.15 \leq 0.247954$$

Therefore, although the $\hat{p}$ is larger than $p = 0.55$, since the error is within the range by (b), this $\hat{p}$ is reasonable and also consistent with the previous answers.

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?

Binomial distribution. Binomial(0.55)

5) Assume 1000 people vote. If $p = 0.55$, 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.

Let Y be the binomial variable that represents the number of votes A receive.

$$P\left(\frac{Y}{1000} > 0.5\right) = P(Y > 500) = \sum_{i>500} \binom{1000}{i} p^i (1-p)^{1000-i} = 1 - P(Y \leq 500)$$
$$= 0.9991534507833808$$

Therefore, that candidate A has probability $= 0.9991534507833808$ to win the election if 1000 people vote.

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?

Let Z be the binomial variable that represents the number of the votes A receive. By Python,

$$Z = 514 \Rightarrow \hat{p} = \frac{Z}{1000} = .514, \qquad \varepsilon = |\hat{p} - p| = .36$$

$$P(Z > 500) = \sum_{i>500} \binom{1000}{i} \hat{p}^i (1-\hat{p})^{1000-i} = 0.8035058249057571$$

Thus, using the $\hat{p}.514$ based on data, A has prob $= 0.8035$ to win the election.

$$P(|\hat{p} - p| \geq \varepsilon) \leq 0.05 \Rightarrow 2e^{-2\cdot\varepsilon^2\cdot 1000} = 0.05 \Rightarrow \varepsilon \geq 0.0429469 \dots$$

Therefore, we have 95% confidence that the value of p is within an error of 0.0429469, that is, within 95% confidence interval $= (\hat{p} - 0.043, \hat{p} + 0.043)$. That is, we should have 95% confidence that real p is within $(.514 - .043, .514 + .043) = (.471, .557)$ predicting through $\hat{p}$.

Since the true probability $p = 0.55$ is within the interval, $\hat{p}$ is consistent with p.

7) Generate a 'guess' for the value of p by sampling a $N\left(\hat{p}, \frac{\hat{p}(1-\hat{p})}{k}\right)$ 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?

By Python, generated the data through $N\left(.531, \frac{.531(1-.531)}{1000}\right)$, get the probability K_i. And for each i, calculate the probability that candidate A wins.

the average probability of winning is `0.9117421792201468`.

The probability that A win is much higher than use $\hat{p}$ itself.

8) (Bonus) Why the predictions so wrong?

There are several possible reasons.

First, if the actual support rate between the two are similar (with small error), it is acceptable that someone win with a narrow victory.

In addition, even though in a pretty large sample, the probability of candidate A to win is close to but never be 1 unless everyone $p = 1$ likely to choose A. That is, although it might small, $P(A\ win)$ or $P(B\ win)$ are greater than 0, both are possible to win.

Besides, the sample that selected by papers and pollsters might be biased. For example, if people who choose Thomas Dewey would be more likely to respond to the surveys, the results might be more likely to skewed to Dewey. Or, if Dewey tried to lead public opinion to guide the results, using herd mentality and controlling some papers and pollsters, it is obvious that those reports were not real.

2 Regression Comparison

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.

When using a constant value to model Y, using the expected value is good for prediction to try to minimize the loss. Thus, choose $c = E[Y]$ or the sample mean of training data, which should be similar with small difference, and using $E[Y]$ might be more general.

$$\begin{aligned} E[Y] &= E[4 - 3X_1^2 + X_3 - 0.01X_4 + X_2 * X_5 + N(0, 0.1)] \\ &= 4 - 3E[X_1^2] + E[X_1 + 2X_2] - 0.01E[X_2^2 + 2X_2 + 4] + E[X_2 \cdot X_5] + 0 \\ &= 4 - 3(3^2 + 1) + (3 + (-2) \cdot 2) - 0.01\left(((-2)^2 + 1) + \left(2 \cdot (-2)\right) + 4\right) + (-2) \cdot (0.8) \\ &= -28.65 \end{aligned}$$

sample mean of $Y = -28.64914554$

Error on training set is `317.78270401` and error in test set is `330.82362102`.

The difference is `-13.04091701`.

Since $f(\underline{x}) = c$ is not a function of x_i, it won't depend on any x to help prediction. Hence, the value of d is not matter.

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?

When the depth of the decision tree increase, the error decreases. However, since depth $= 15$, the loss starts to increase, which might attribute to overfitting. Thus, depth $= 15$ appears to be the optimal depth to grow the tree to minimize the error.

Compared to the $Loss = 300^{+}$ in the above constant model, this model has a much better performance, which could lower the predicting Losses to around 1.

The following is the decision tree error on the testing data as the depth of the tree increases.

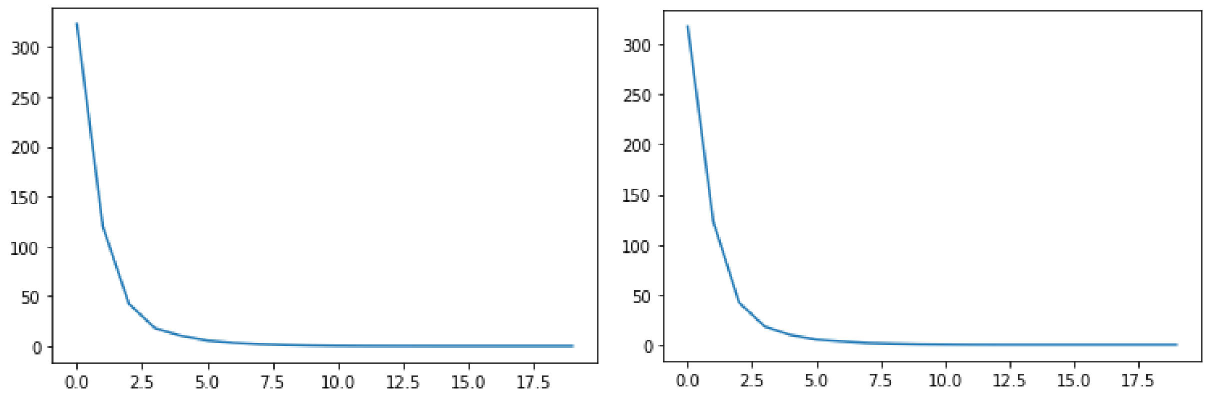

Figure 1 Loss(depth) in 2.2, train (Left), test (Right)

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?

As the size decrease, the error decreases. However, when the sample size becomes lower than 5 (`4.8828125`), the error start to increase, which might also contribute to overfitting. Thus, size $= 5$ should be an optimal size when $d = 10$.

Comparing to the constant model, this model has a much better performance, which could lower the predicting Losses to around 1.

The following is the decision tree error on the testing data as the sample size increases.

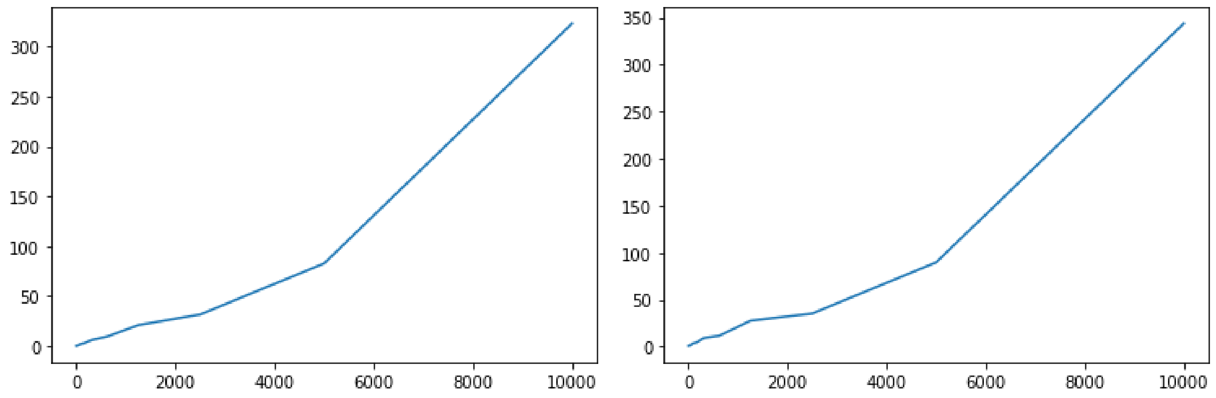

Figure 2 Loss(size) in 2.3, train (Left), test (Right)

4) Which is better for minimizing error? Truncating by depth or truncating by sample size?

Run with same datasets.

- For the model that truncating by depth, the optimal depth is around 13, with error = `0.4342409991503472`.
- For the model that truncating by sample size, the optimal size is around 5, with error = `0.4026291269598245`.

Compare the two errors which both generated through their optimal parameters, since $0.4342409991503472 < 0.4026291269598245$, truncating by depth is better for minimizing errors.

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?

For $d = 50$, the optimal depth and sample sizes changed. Optimal depth is 12, with error = 0.635680987574582; Optimal size is around 10 (`9.765625`), with error = `0.495259244569`.

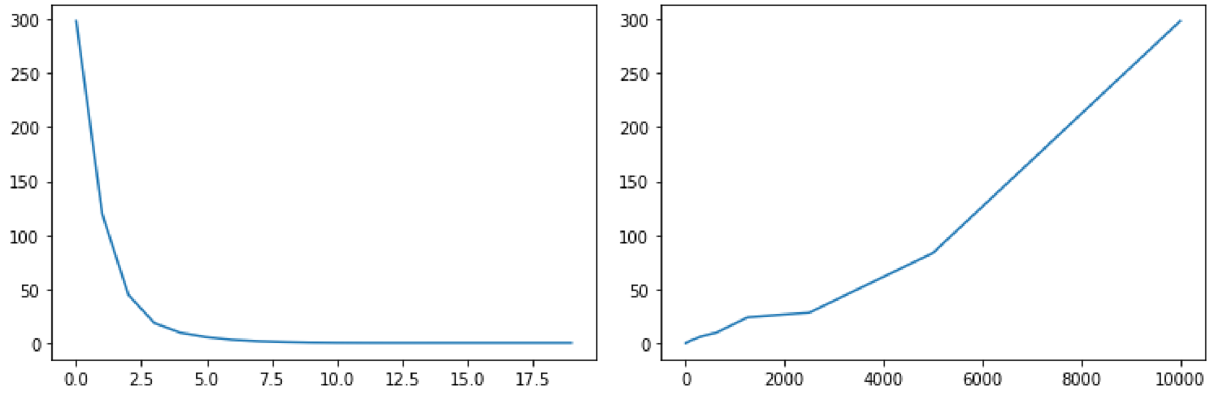

Figure 3 Loss(depth) [Left] & Loss(size) [Right] in 2.5, using test data

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?

When d increases, the number of superfluous features increases. This might attribute to the exponentially rising variance of $X_s = X_6 + \cdots + X_d \sim N(0, (d-5)^2 \cdot 1)$. From the generated results, using size is better for excluding independent features.

Use S to represent the number superfluous features that that are included in the decision tree depends on d, the following is the plot of $S(d)$.

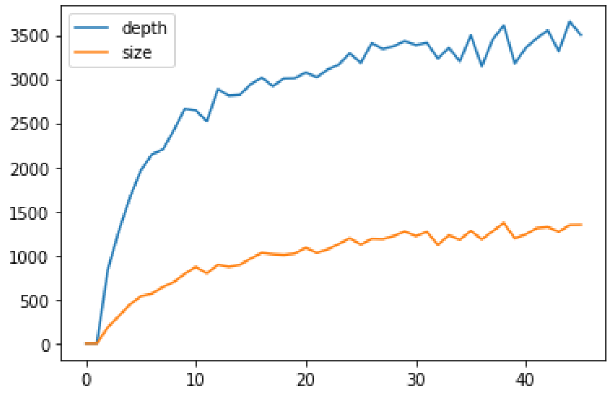

Figure 4 S(d) in 2.6

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?

i	w_i
0	29.29686922042101
1	-14.622756651610578
2	7.353327017849952
3	-2.2882881509562996
4	-0.006743268684635001
5	-1.9988494591631096
6	-0.019611401835880306
7	0.0017356098457102184
8	-0.04734800966068492
9	0.029186063278831012
10	0.020150842833630882

Error in training data is 18.68091096685721.
Error in testing data is 17.814054895724887.

The error of linear model is much smaller than the error of the constant model. Since the coefficient of $X_6, \dots, X_d$ is small compared to others, overfitting maybe exist but will remain small.

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 $[C(\epsilon)]$ that make it into the final model as a function of ϵ, and plot the error on the testing data $[Loss(\epsilon)]$ for the final model as a function of ϵ. Is this a good strategy?

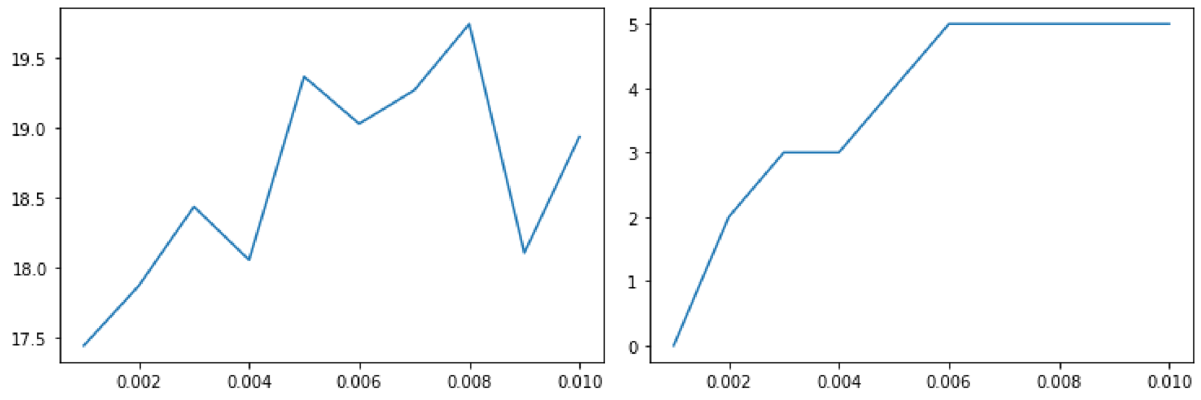

Figure 5 $Loss(\epsilon)$ [Left] & $C(\epsilon)$ [Right] in 2.8

Using small ϵ could help reduce the loss of the function. Thus, this is a good strategy.

9) Which model is superior here? Why?

Since both are good models with small testing error, choose the (2.8) one since it is faster.

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

$$\underline{X} = (X_1, \dots, X_6, X_1^2, X_1 \cdot X_2, \dots, X_1 \cdot X_6, X_2^2, X_2 \cdot X_3, \dots, X_2 \cdot X_6, \dots, \dots, X_6^2)$$

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?

This model is much better than as it could fit Y except the random noise theoretically. Overfitting is not a issue as the slopes of superfluous features are small. And this one is the superior.

Expected model:

X	w
X_0	$w_0 = 1$
X_1^2	$w_7 = -3$
X_3	$w_3 = 1$
X_4	$w_4 = -0.01$
$X_2 \cdot X_5$	$w_{17} = 1$
else	$w_i = 0$

Error in training data is `0.010167106511083629`.
Error in testing data is `0.009832119816946375`.

Comparing to models above, the error is much smaller. Therefore, this model is the superior.