MATH 342W / 642 / RM 742 Spring 2025 HW #4

Sergio E. Garcia Tapia

Monday 21st April, 2025

Problem 1

These are questions about Silver's book, chapters 7–11. You can skim chapter 10 as it is not so relevant for the class. For all parts in this question, answer using notation from class (i.e. $t, f, g, h^*, \delta, \epsilon, e, t, z_1, \ldots, z_t, \mathbb{D}, \mathcal{H}, \mathcal{A}, \mathcal{X}, \mathcal{Y}, X, y, n, p, x_{\cdot 1}, \ldots, x_{\cdot p}$, $x_{1\cdot}, \ldots, x_{n\cdot}$, etc.) as well as in-class concepts (e.g. simulation, validation, overfitting, etc) and also we now have $f_{pr}, h^*_{pr}, g_{pr}, p_{th}$, etc from probabilistic classification as well as different types of validation schemes).

Note: I will not ask questions in this assignment about Bayesian calculations and modeling (a large chunk of Chapters 8 and 10) as this is the subject of Math 341/343. We also won't cover chapters 12-13 and the conclusion on the homework.

(a) [easy] Why are flu fatalities hard to predict? Which type of error is most dominant in the models?

Flu fatalities are estimated by considering R_0, the basic reproduction number. However, to estimate reliably, the disease must sweep through a community. The fatality rate also cannot be measured accurately early on. The result is a need to extrapolate from few data points, which often leads to poor predictions. In this setting, the most dominant type of error is ignorance error δ, since n is small.

(b) [easy] In what context does Silver define extrapolation and what term did he use? Why does his terminology conflict with our terminology?

Silver defines extrapolation as "making the assumption that the current trend will continue indefinitely into the future". This sounds like extrapolation is when we believe a model is stationary, meaning that the relationship between the features and the target do not change over time. In class, we defined extrapolation as predicting outside the range of the design matrix X, which was the defined to be a rectangle delimited by the maximum and minimum feature values in each dimension. The time component or idea of stationarity did not play a role in this definition.

(c) [easy] Give a couple examples of extraordinary prediction failures (by very famous people who were considered heavy-hitting experts of their time) that were due to reckless extrapolations.

- In 1682, Sir William Petty predicted a population of 700 million in 2012, but it was instead 7 billion around 2011.
- In 1968, Paul Ehrlich and Anne Ehrlich incorrectly predicted that hundreds of millions of people would die from starvation.

(d) [easy] Using the notation from class, define "self-fulfilling prophecy" and "self-canceling prediction".

In the case of a self-fulfilling prophecy, it is as if a model initially makes a particular prediction $\hat{y}$. Then the $\hat{y}$ becomes a feature or strongly influences a feature, leading to other predictions being close to $\hat{y}$.

(e) [easy] Is the SIR model of infectious disease under or overfit? Why?

Underfit. One reason is that the SIR model assumes that the interactions between the population is random, which often does not hold. It also assumes that different subjects are equally likely to be susceptible. However, certain groups may be more susceptible than others, perhaps due to religion, or occupation, among other things. The model assumes that subjects are equally likely to be vaccinated, but that may not hold due to differing beliefs about the risk of contracting the disease, or due to other inherent beliefs held by a population. In short, the SIR model expects certain "homogeneous" conditions to hold, which in general do not. Because it fails to take into account many asymmetries and other complicated interactions, it underfits.

(f) [easy] What did the famous mathematician Norbert Weiner mean by "the best model of a cat is a cat"?

He meant that in order for a model to be completely accurate, it must be know everything about a phenomenon. Conversely, no model can be completely accurate because it lacks *something*, and that alone can lead to different predictions. Perhaps the model for a cat won't be a cat itself, even the creature it describes closely resembles a cat.

(g) [easy] Not in the book but about Norbert Weiner. From Wikipedia:

> Norbert Wiener is credited as being one of the first to theorize that all intelligent behavior was the result of feedback mechanisms, that could possibly be simulated by machines and was an important early step towards the development of modern artificial intelligence.

What do we mean by "feedback mechanisms" in the context of this class?

Feedback mechanisms refers to validation about the predictions that we make. In this class, we split our data set $\mathbb{D}$ into $\mathbb{D}_{\text{test}}$ and $\mathbb{D}_{\text{train}}$. By validating the prediction function $g = \mathcal{A}(\mathbb{D}_{\text{test}}, \mathcal{H})$ from our model $(\mathcal{A}, \mathcal{H})$, we learn whether our predictions are effective. Then we use this information to inform our decision about how to tune our model. This is the "learning from data" approach at play.

(h) [easy] I'm not going to both asking about the bet that gave Bob Voulgaris his start. But what gives Voulgaris an edge (p239)? Frame it in terms of the concepts in this class.

Voulgaris obtains many different data points (large n) and many features p. He analyzes the trends in the data carefully, and is careful not to overfit when building a model that he uses to make predictions and place bets.

(i) [easy] Why do you think a lot of science is not reproducible?

A lot of science is not reproducible because it is based on predictions made by fitting noise. Given we are in the era of Big Data, there is more noise and the same amount of objective truth. Thus scientists are likelier to overfit the data that they gather. Silver suggests that it's a consequence of applying the frequentist approach to probability, which encourages reducing error by collecting more data. However it does not encourage telling the noise from the data.

(j) [easy] Why do you think Fisher did not believe that smoking causes lung cancer?

Fisher was biased, and his statistical philosophy conflicted with the practice that was used to arrive at the hypothesis about smoking and cancer. He placed more emphasis on the methods than the interpretation of the results was too victim of getting caught in the noise.

(k) [easy] Is the world moving more in the direction of Fisher's Frequentism or Bayesianism?

Bayesianism, with Silver claiming that some researchers have begun arguing against it in undergraduate study.

(l) [easy] How did Kasparov defeat Deep Blue? Can you put this into the context of over and underfiting?

Kasparov made a move that does not occur often in a master competition. Therefore, the number of data points n and the number of features p were both low in regards to the move (how players have responded to it in the class, how effective it was, the win rate of the player who responded with that move, etc). If the computer were to build a predictive model to know how to counter that move, the model would suffer from underfitting because of the small n and p.

(m) [easy] Why was Fischer able to make such bold and daring moves?

Fischer was young enough to not be full "indoctrinated" or "biased" by the heuristics often associated with playing chess. In turn, Fischer predicted his opponent did adhere to said heuristics, and used this to (correctly) formulate a prediction about what his opponent would do in response, thereby enabling him to make such a move.

(n) [easy] What metric y is Google predicting when it returns search results to you? Why did they choose this metric?

When Google returns search results, they are measuring "usefulness" or "relevance" of the results. Google measures this in order to improve their search algorithms, thereby refining getting closer to providing you with the information that you are looking for.

(o) [easy] What do we call Google's "theories" in this class? And what do we call "testing" of those theories?

Google's theories are akin to creating a model, which consists of getting some data $\mathbb{D}$, an algorithm $\mathcal{A}$, and a set of candidate functions $\mathcal{H}$. Testing those theories corresponds to out-of-sample validation with $g = \mathcal{A}(\mathbb{D}, \mathcal{H})$ on $\mathbb{D}_{\text{test}}$.

(p) [easy] p315 give some very practical advice for an aspiring data scientist. There are a lot of push-button tools that exist that automatically fit models. What is your edge from taking this class that you have over people who are well-versed in those tools?

A student that succeeds in this class is well-equipped to assess their predictions and models. We are aware of the dangers of overfitting or even underfitting, the importance of using honest metrics and viewing results statistically without letting the model be the final say.

(q) [easy] Create your own 2×2 luck-skill matrix (Fig. 10-10) with your own examples (not the ones used in the book).

	Low luck	**High luck**
Low skill	Uno	Bingo
High skill	Billiards	Settlers of Catan

(r) [easy] [EC] Why do you think Billing's algorithms (and other algorithms like his) are not very good at no-limit hold em? I can think of a couple reasons why this would be.

(s) [easy] Do you agree with Silver's description of what makes people successful (pp326-327)? Explain.

Yes. If we were to think of "hard work", "natural talent", "opportunities", "environment" as predictors of "success", then yes, I think this is a good model for success. It stands to reason that if you work harder, your chance of being successful increases. Similarly, a person who has more life opportunities takes more "shots", and after enough of them, some of them ought to land in the basket.

(t) [easy] Silver brings up an interesting idea on p328. Should we remove humans from the predictive enterprise completely after a good model has been built? Explain

No. Part of what makes a "good" model is the process that led to its creation. Silver mentions that we can never be certain about the reason for an inaccurate prediction. However, being disciplined in our model construction and prediction practices, where we gather a lot of data, from different places, and test our predictions honestly, our models can become better. We mut remember that models do not have the final say, and they need to be imbued with meaning and refined each time.

(u) [easy] According to Fama, using the notation from this class, how would explain a mutual fund that performs spectacularly in a single year but fails to perform that well in subsequent years?

The mutual funds built a model g created a model whose $oosRMSE$ was lower for observations $\boldsymbol{x}$ that were not too far from Range$[X]$ but increased as $\boldsymbol{x}$ moved away from Range$[X]$. Put another way, the model suffers from extrapolation that is more severe as observations move further from Range$[X]$, where time t is part of the observation.

(v) [easy] Did the Manic Momentum model validate? Explain.

Yes. In the example given by Silver, the investor was able to use past data to create a model that earned him a profit (not accounting for transaction costs).

(w) [easy] Are stock market bubbles noticable while we're in them? Explain.

According to the efficient market hypothesis they are no, because if they were, then people could make a profit. However, under the efficient market hypothesis, one cannot beat the market consistently.

(x) [easy] What is the implication of Shiller's model for a long-term investor in stocks?

A long-term investor can make successful predictions using the P/E ratio if they wait *very* long, thereby being able to make a profit.

(y) [easy] In lecture one, we spoke about "heuristics" which are simple models with high error but extremely easy to learn and live by. What is the heuristic Silver quotes on p358 and why does it work so well?

The heuristic is "follow the crowd, especially when you don't know any better". It works well because if many people agree with a certain view, it is likelier that it has been considered more and people have had more chances to debunk it. Since people still consistently hold the view, chances are that it's a "safe" view to hold. Put another way, if we deviate much from it, it is more likely that we are seeing noise.

(z) [easy] Even if your model at predicting bubbles turned out to be good, what would prevent you from executing on it?

If the risk associated with the incorrect prediction is much more substantial than the reward for getting it right.

(aa) [easy] How can heuristics get us into trouble?

Heuristics are useful when empirical experience backs it up. However, they can get us into trouble if we attempt to apply them in every situation. A heuristic is not a proved or justified result, so we might say that its guidance is undeterministic. If we use a heuristic to make predictions, and the input that arrives does not fall within the assumptions of the heuristics, or differ from the experience on which the heuristic is based, we are likely to get our predictions very wrong.

Problem 2

These are some questions related to probability estimation modeling. Let X denote the design matrix with n rows and $p+1$ columns (with the first column being $\mathbf{1}_n$ and the other columns being linearly independent predictors) and $\boldsymbol{y}$ is the binary response vector of size n and use this notation throughout your responses.

(a) [easy] What is g_0 if you are modeling probability estimates?

$g_0 = \bar{y}$, the proportion of 1's in the n observations in our data set $\mathbb{D}$.

(b) [easy] What is $\mathcal{H}_{pr}$ for the probability estimation algorithm that employs the linear model in the covariates with logistic link function?

$$\mathcal{H}_{pr} = \left\{ \phi(\mathbf{w} \cdot \mathbf{x}) : \mathbf{w} \in \mathbb{R}^{p+1} \right\}$$

where ϕ is the logistic link function, given by

$$\phi(u) = \frac{1}{1 + e^{-u}}$$

(c) [easy] What is $\mathcal{H}_{pr}$ for the probability estimation algorithm that employs the linear model in the covariates with cloglog link function?

$$\mathcal{H}_{pr} = \left\{ \phi(\mathbf{w} \cdot \mathbf{x}) : \mathbf{w} \in \mathbb{R}^{p+1} \right\}$$

where ϕ is the cloglog link function, given by

$$\phi(u) = 1 - e^{e^{-u}}$$

(d) [easy] Let $\mathcal{A} : \boldsymbol{b} = \arg\max_{\boldsymbol{w} \in \mathbb{R}^{p+1}} \{\ldots\}$. Derive the expression that replaces the ... which will be a function of $X, \boldsymbol{y}, \boldsymbol{w}, n$. Note: this algorithm fits a "logistic regression".

First, suppose the phenomenon is described by

$$y = t(z_1, z_2, \ldots, z_t)$$

where $y \in \mathcal{Y}$ and $\mathcal{Y} = \{0, 1\}$. To use probabilistic estimation, we consider $Y \sim \text{Bernoulli}(t(z_1, \ldots, z_t))$, which is not random because t is deterministic. By using p features $x_1, \ldots, x_p$ are proxies to the true drivers $z_1, \ldots, z_t$, and modeling with a probabilistic function

$$f_{pr} : \mathbb{R}^{p+1} \to (0, 1)$$

we approximate Y as

$$Y \sim \text{Bernoulli}(f_{pr}(x_1, \ldots, x_p))$$

we get a random variable Y that depends on $\boldsymbol{x}$. We can calculate the probability

$$P(\mathbb{D}) = P(Y_1 = y_1, Y_2 = y_2, \ldots, Y_n = y_n | \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$$

where $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$ are the n data points in $\mathbb{D}$. If we assume that all n observations are independent, we can simplify our join probability expression to

$$P(\mathbb{D}) = \prod_{i=1}^{n} P(Y_i = y_i \mid \boldsymbol{x}_i)$$

Since $Y_i \mid \boldsymbol{x}_i$ is Bernoulli, we can use the Bernoulli PMF:

$$P(\mathbb{D}) = \prod_{i=1}^{n} f_{pr}(\boldsymbol{x}_i)^{y_i} (1 - f_{pr}(\boldsymbol{x}_i))^{1-y_i}$$

Now suppose ϕ is the logistic link function. We can approximate $P(\mathbb{D})$ using $\phi \in \mathcal{H}_{pr}(\phi)$:

$$\begin{aligned} P(\mathbb{D}) &\approx \prod_{i=1}^{n} \phi(\boldsymbol{w} \cdot \boldsymbol{x}_i)^{y_i} (1 - \phi(\boldsymbol{w} \cdot \boldsymbol{x}_i))^{1-y_i} \\ &= \prod_{i=1}^{n} \left(\frac{1}{1 + e^{-\boldsymbol{w} \cdot \boldsymbol{x}_i}} \right)^{y_i} \left(\frac{1}{1 + e^{\boldsymbol{w} \cdot \boldsymbol{x}_i}} \right)^{1-y_i} \end{aligned}$$

This final expression is what we attempt to maximize in logistic regression:

$$\mathcal{A} : \boldsymbol{b} = \arg\max_{\boldsymbol{w} \in \mathbb{R}^{p+1}} \{P(\mathbb{D})\}$$

(e) [easy] Why is logistic regression an example of a "generalized linear model" (glm)?

Because even though the candidate functions are not linear, they map lines to the probability space (via link functions). Since link functions are monotonic, they are also injective, and hence we can invert them to obtain an interpretation of the results in terms of the linear expression $\boldsymbol{w} \cdot \boldsymbol{x}$.

(f) [easy] Consider $\boldsymbol{x}_*$ to be a new unit. For its prediction, the probability estimate that $y_* = 1$ is 37%, what is the log odds of $y_* = 1$?

In probability estimation modeling, we estimate the probability that $y = 1$ for some input $\boldsymbol{x}$ as

$$\hat{p} = \phi(\boldsymbol{b} \cdot \boldsymbol{x})$$

for some $\boldsymbol{b} \in \mathbb{R}^{p+1}$. We are told that $\hat{p} = 0.37 = \phi(\boldsymbol{b} \cdot \boldsymbol{x}_*)$. Assuming ϕ is the logit link function used in logistic regression, the log odds is given by

$$\boldsymbol{b} \cdot \boldsymbol{x}_* = \ln\left(\frac{\hat{p}}{1-\hat{p}}\right) = \ln\left(\frac{0.37}{0.63}\right) \approx -0.53$$

(g) [easy] If, $\boldsymbol{x}_*\boldsymbol{b} = 3.1415$ where $\boldsymbol{b}$ is the result of the logistic regression fit, what is the probability estimate that $y_* = 1$?

If ϕ is the logit link function, then the probability estimate is

$$\hat{p} = \phi(\boldsymbol{x}_*\boldsymbol{b}) = \frac{1}{1 + e^{-\boldsymbol{x}_*\boldsymbol{b}}} \approx 0.96$$

(h) [harder] If, $\boldsymbol{x}_*\boldsymbol{b} = 3.1415$ where $\boldsymbol{b}$ is the result of the probit regression fit, what is the probability estimate that $y_* = 1$?

The probit link function is precisely the CDF of the standard normal random variable:

$$\phi(u) = \int_{-\infty}^{u} e^{-v^2/2} dv$$

This is often denoted as $\Phi(u)$ (capital phi), and its values are usually tabulated. The probability estimate is given by $\phi(\boldsymbol{x}_*\boldsymbol{b}) = \phi(\boldsymbol{x}_*\boldsymbol{b}) = \Phi(3.1415) \approx 0.9991595759563348$.

(i) [easy] In probability estimation modeling, what is the formula for the Brier Score performance metric? Prove the Brier score is always non-positive.

The formula for the Brier score is

$$\overline{s} := \frac{1}{n}\sum_{i=1}^{n} s_i$$

where

$$s_i := -(y_i - \hat{p}_i)^2$$

Here y_i is the response of the ith data point in $\mathbb{D}$, $\hat{p}_i$ is the probability estimate that the ith response will be 1, and n is the number of data points in $\mathbb{D}$. Since $y_i \in \{0, 1\}$ and $\hat{p}_i \in [0, 1]$, we have $(y_i - \hat{p}_i)^2 \in [0, 1]$ and hence $s_i \in [-1, 0]$. Since the Brier score $\overline{s}$ is the arithmetic mean of n values s_i, it too lies in $[0, 1]$, and hence it is non-negative.

(j) [easy] In probability estimation modeling, what is the formula for the Log Scoring Rule performance metric? Prove the Log Scoring Rule is always non-positive.

The formula for the Log Scoring Rule is

$$\overline{s} := \frac{1}{n}\sum_{i=1}^{n} s_i$$

where

$$s_i := y_i \ln(\hat{p}_i) + (1 - y_i) \ln(1 - \hat{p}_i)$$

where y_i is the ith response, $\hat{p}_i$ is the probability that the ith response is 1, and n is the number of data points in $\mathbb{D}$. It is sufficient to show that $s_i \leq 0$, since $\overline{s}$ is the average of these s_i values. To see this, note that since $\hat{p}_i$ is a (non-degenerate) probability, it follows that $\hat{p}_i \in (0, 1)$ and $(1 - \hat{p}_i) \in (0, 1)$. Thus, $\ln(\hat{p}_i) < 0$ and $\ln(1 - \hat{p}_i) < 0$. Since $y_i \in \{0, 1\}$, we either have $s_i = \ln(\hat{p}_i) < 0$ when $y_i = 1$, or $s_i = \ln(1 - \hat{p}_i) < 0$ when $y_i = 0$. In either case, $s_i \leq 0$.

(k) [difficult] Generalize linear probability estimation to the case where $\mathcal{Y} = \{C_1, C_2, C_3\}$, i.e. a nominal variable with $L = 3$ levels.

Assume the logistic link function. Write down the objective function that is argmax'd over the parameters (you define what these parameters are — that is part of the question). Once you get the answer you can see how this easily goes to $L > 3$, an

arbirtrary[1] number of response levels.

Suppose our phenomenon is described by

$$y = t(z_1, z_2, \ldots, z_t)$$

Given features $x_1, \ldots, x_p$ that are proxies to the true drivers, we introduce a probability function $f_{pr} : \mathbb{R}^{p+1} \to (0, 1)$ that maps these features to a probability. We introduce a random variable Y so that

$$Y \sim \text{Categorical}(f_{pr}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n))$$

If we assume the n observations are independent, we get

$$\begin{aligned} P(\mathbb{D}) &= P(Y_1 = y_1, \ldots, Y_n = y_n \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \\ &= \prod_{i=1}^{n} P(Y_i = y_i \mid \boldsymbol{x}_i) \end{aligned}$$

If $V \sim \text{Categorical}(p_0, p_1, p_2)$ with support $\{0, 1, 2\}$, where $p_0 + p_1 + p_2 = 1$, and the PMF is given by

$$f(V = i) = p_i$$

Alternatively, we can write

$$\begin{aligned} P(V = v) &= \prod_{k=0}^{2} p_k^{[v=k]} \\ &= p_0^{[v=0]} p_1^{[v=1]} p_2^{[v=2]} \end{aligned}$$

where $[v = k]$ is Iverson's bracket notation, which is 1 if $v = k$, and 0 otherwise. Now we can write

$$P(\mathbb{D}) = \prod_{i=1}^{n} P(Y_i = y_i \mid \boldsymbol{x}_i)$$

[1]Note: The algorithm for general L is known as all of the following: "multinomial logistic regression", "polytomous LR", "multiclass LR", "softmax regression", "multinomial logit" (mlogit), the "maximum entropy" (MaxEnt) classifier, and the "conditional maximum entropy model". You can inflate your resume with lots of redundant jazzy terms by doing this one question!

$$= \prod_{i=1}^{n} f_{pr}(\boldsymbol{x}_i)^{[y_i=0]} f_{pr}(\boldsymbol{x}_i)^{[y_i=1]} f_{pr}(\boldsymbol{x}_i)^{[y_i=2]}$$

Suppose that ϕ is the logistic link function

$$\phi(u) = \frac{1}{1+e^{-u}}$$

Consider the candidate set of functions of the form

$$\mathcal{H} = \{\, \phi(\boldsymbol{w} \cdot \boldsymbol{x}) \mid \boldsymbol{w} \in \mathbb{R}^{p+1} \,\}$$

Now we approximate the probability that $Y_i = 0$ by $\phi(\boldsymbol{v} \cdot \boldsymbol{x}_i)$, the probability that $Y_i = 1$ by $\phi(\boldsymbol{w} \cdot \boldsymbol{x}_i)$, and the probability that $Y_i = 2$ by $1 - (\phi(\boldsymbol{v} \cdot \boldsymbol{x}_i) + \phi(\boldsymbol{w} \cdot \boldsymbol{x}_i))$:

$$\begin{aligned} P(\mathbb{D}) &= \prod_{i=1}^{n} f_{pr}(\boldsymbol{x}_i)^{[y_i=0]} f_{pr}(\boldsymbol{x}_i)^{[y_i=1]} f_{pr}(\boldsymbol{x}_i)^{[y_i=2]} \\ &= \prod_{i=1}^{n} \phi(\boldsymbol{v} \cdot \boldsymbol{x})^{[y_i=0]} \phi(\boldsymbol{w} \cdot \boldsymbol{x})^{[y_i=1]} \left(1 - \phi(\boldsymbol{v} \cdot \boldsymbol{x}_i) - \phi(\boldsymbol{w} \cdot \boldsymbol{x}_i)\right)^{[y_i=2]} \end{aligned}$$

Thus our algorithm becomes

$$\mathcal{A} : \boldsymbol{b}_1, \boldsymbol{b_2} = \underset{\boldsymbol{v},\boldsymbol{w} \in \mathbb{R}^{p+1}}{\arg\max} \{P(\mathbb{D})\}$$

For the next two questions, let $n_1 := \sum \mathbb{1}_{y_i=1}$ and $n_0 := \sum \mathbb{1}_{y_i=0}$ so that $n = n_0 + n_1$. Then assume $n_1 \neq n_0$. This is equivalent to letting $n_1 = cn$ and $n_0 = (1-c)n$ and assuming $c \in [0,1]/\left\{\frac{1}{2}\right\}$.

(l) [harder] [MA] Prove the Brier score is always higher for g_0 vs the model where you set $\hat{p}_i = \frac{1}{2}$ for all i. Hint: $(1-c)c < \frac{1}{2}$ for $c \in [0,1]/\left\{\frac{1}{2}\right\}$.

(m) [difficult] [MA] Prove the Log Scoring Rule is always higher for g_0 vs the model where you set $\hat{p}_i = \frac{1}{2}$ for all i.

Problem 3

These are some questions related to polynomial-derived features and logarithm-derived features in use in OLS regression.

(a) [harder] What was the overarching problem we were trying to solve when we started to introduce polynomial terms into $\mathcal{H}$? What was the mathematical theory that justified this solution? Did this turn out to be a good solution? Why / why not?

Our intention was to reduce misspecification error because linear functions are not always appropriate for fitting a data set. The use of polynomials is justified by the Stone-Weierstrass Theorem, which asserts that a continuous real-valued function $f : \mathbb{R}^p \to \mathbb{R}$ can be accurately approximated by a polynomial. This turned out to be a good solution because when $n \gg p$, misspecification error can decrease much more than the estimation error might increase. Put another way, increasing p puts us at risk of overfitting, but if n is large, than it is unlikely that we are overfitting, and allowing for polynomials enables us to reduce misspecification error meaningfully.

Moreover, assuming we use polynomials of degree k, and our data is not categorical, as long as our data set has more than k distinct values of x, the resulting transformed matrix will be full rank. Hence, we can still use OLS.

One downside is that high degree polynomials are subject to Runge's Phenomenon, which may affect the accuracy at the endpoints of the interval that the inputs belong to.

(b) [harder] We fit the following model: $\hat{\boldsymbol{y}} = b_0 + b_1 x + b_2 x^2$. What is the interpretation of b_1? What is the interpretation of b_2? Although we didn't yet discuss the "true" interpretation of OLS coefficients, do your best with this.

When the input x changes by 1 unit, the response changes by $b_1 + b_2$ units. More generally, when the input changes by k units, the response changes by $b_1 + b_2 k$ units.

$$\begin{aligned}\Delta\bar{y} &= (b_0 + b_1 x_f + b_2 x_f^2) - (b_0 + b_1 x_0 + b_2 x_0^2) \\ &= b_1(\Delta x) + b_2(x_f^2 - x_0^2)\end{aligned}$$

(c) [difficult] Assuming the model from the previous question, if $x \in \mathcal{X} = [10.0, 10.1]$, do you expect to "trust" the estimates b_1 and b_2? Why or why not?

(d) [difficult] We fit the following model: $\hat{\boldsymbol{y}} = b_0 + b_1 x_1 + b_2 \ln(x_2)$. We spoke about in class that b_1 represents loosely the predicted change in response for a proportional movement in x_2. So e.g. if x_2 increases by 10%, the response is predicted to increase by $0.1 b_2$. Prove this approximation from first principles.

If x_2 changes by some amount $\Delta x_2 = x_{2,f} = x_{2,0}$ and x_1 does not change, we have

$$\begin{aligned}
\Delta\hat{y} &= (b_0 + b_1 x_1 + b_2 \ln(x_{2,f})) - (b_0 + b_1 x_1 + \ln(x_{2,0})) \\
&= b_2 \ln\left(\frac{x_{2,f}}{x_{2,0}}\right) \\
&\approx b_2 \left(\frac{x_{2,f}}{x_{2,0}} - 1\right) \\
&= b_2 \underbrace{\left(\frac{x_{2,f} - x_{2,0}}{x_{2,0}}\right)}_{\text{proportional change}}
\end{aligned}$$

Hence, if x_2 increases by 10%, this means the proportional change in the expression above is 0.10, and hence, $\Delta\hat{y} = 0.1 b_2$.

(e) [easy] When does the approximation from the previous question work? When do you expect the approximation from the previous question not to work?

The approximation works when $\ln(x_{2,f}/x_{2,0}) \approx \frac{x_{2,f}}{x_{2,0}} - 1$. This approximation is valid when $x_{2,f}/x_{2,0} \approx 0$. This comes from the Taylor series approximation to

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} + \cdots \approx x, \quad \text{if } x \approx 0$$

which implies

$$\ln(x) = \ln((1+x) - 1) \approx x - 1$$

Hence the approximation may not work when the proportional change in x_2 differs by a large percentage, say, as the proportional change approaches 1 and goes beyond that.

(f) [harder] We fit the following model: $\ln(\hat{\boldsymbol{y}}) = b_0 + b_1 x_1 + b_2 \ln(x_2)$. What is the interpretation of b_1? What is the *approximate* interpretation of b_2? Although we didn't yet discuss the "true" interpretation of OLS coefficients, do your best with this.

Suppose x_2 does not change, and x_1 changes by $\Delta x_1 = x_{1,f} - x_{1,0}$. Then the change in $\ln(\hat{y})$ is given by

$$\begin{aligned}
\Delta \ln(\hat{y}) &= \ln\left(\frac{\hat{y}_f}{\hat{y}_0}\right) \\
&= \ln(\hat{y}_f) - \ln(\hat{y}_0) \\
&= b_1(x_{1,f} - x_{1,0})
\end{aligned}$$

Using the approximation $\ln(u) = u - 1$:

$$\begin{aligned} b_1 \Delta x_1 &= \ln\left(\frac{\hat{y}_f}{\hat{y}_0}\right) \\ &\approx \left(\frac{\hat{y}_f}{\hat{y}_0} - 1\right) \\ &= \frac{\hat{y}_f - \hat{y}_0}{\hat{y}_0} \end{aligned}$$

Hence, if x_1 changes by Δx_1, the response undergoes a $b_1 \Delta x_1$ proportional change. Now suppose instead that x_1 does not change, but x_2 changes by Δx_2. Then a similar calculation yields

$$\begin{aligned} \frac{\hat{y}_f - \hat{y}_0}{\hat{y}_0} &\approx \ln\left(\frac{\hat{y}_f}{\hat{y}_0}\right) \\ &= b_2 \ln\left(\frac{x_{2,f}}{x_{2,0}}\right) \\ &\approx b_2 \left(\frac{x_{2,f} - x_{2,0}}{x_{2,0}}\right) \end{aligned}$$

Hence, a proportional change in x_2 by $\frac{x_{2,f} - x_{2,0}}{x_{2,0}}$ results in a $b_2 \cdot \left(\frac{x_{2,f} - x_{2,0}}{x_{2,0}}\right)$ for the predicted response $\hat{y}$.

(g) [easy] Show that the model from the previous question is equal to $\hat{\boldsymbol{y}} = m_0 m_1^{x_1} x_2^{b_2}$ and interpret m_1.

Assuming the previous model, we use exponentiation to get

$$\begin{aligned} e^{\ln \hat{y}} &= e^{b_0 + b_1 x_1 + b_2 \ln(x_2)} \\ \hat{y} &= e^{b_0} \cdot (e^{b_1})^{x_1} + (e^{\ln x_2})^{b_2} \\ \hat{y} &= m_0 m_1^{x_1} x_2^{b_2} \end{aligned}$$

where $m_0 = e^{b_0}$ and $m_1 = e^{b_1}$.

Problem 4

These are some questions related to extrapolation.

(a) [easy] Define extrapolation and describe why it is a net-negative during prediction.

Let X be a design matrix with p features. Suppose $X_{\cdot k,\min}$ denotes the minimum value for the kth feature, and $X_{\cdot k,\max}$ denotes the maximum value for the kth feature. If

$[X_{\cdot,k,\min}, X_{\cdot,k,\max}]$ denotes a closed interval, then the range of X is defined to be the rectangle obtained as the Cartesian product of all k such intervals:

$$\text{Range}[X] = [X_{\cdot,1,\min}, X_{\cdot,1,\max}] \times \cdot \times [X_{\cdot,p,\min}, X_{\cdot,p,\max}]$$

Extrapolation is predicting for observations $\boldsymbol{x}_*$ that do not belong to Range$[X]$. Extrapolation is a net negative because it assumes that the phenomenon will continue to behave the similarly for observations outside Range$[X]$. Put another way, it assumes that the trend implied by $\mathbb{D}$ continues to hold outside it. This can lead to large out-of-sample errors.

(b) [easy] Do models extrapolate differently? Explain.

Yes, different models extrapolate differently. For example, if we use a linear model, then a prediction function will assume a linear trend that continues to increase (or decrease, depending if the slope is negative) outside of Range$[X]$. Perhaps the phenomenon in actuality has a maximum value, but the linear model will nevertheless predict larger (or smaller) values for larger inputs. An exponential model will too do this, except the out-of-sample error will be much worse due to the rate at which it increases in comparison to linear functions.

(c) [easy] Why do polynomial regression models suffer terribly from extrapolation?

Polynomials suffer from Runge's phenomenon when they are used for interpolation in an interval, especially if they are of high degree. Due to the wild oscillations of such polynomials, their rate of change is large. This can lead to large out-of-sample errors, especially at the endpoints.

Problem 5

These are some questions related to the model selection procedure discussed in lecture.

(a) [easy] Define the fundamental problem of "model selection".

Given a data set $\mathbb{D}$, and M different modeling procedures $\{(\mathcal{A}_m, \mathcal{H}_m)\}_{m=1}^M$, select the best model to predict the response for the phenomenon captured by $\mathbb{D}$.

(b) [easy] Using two splits of the data, how would you select a model?

We partition the data set as $\mathbb{D} = \mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{test}}$, where $\mathbb{D}_{\text{train}} \cap \mathbb{D}_{\text{test}} = \varnothing$. Then, for each of the m models, where $m \in \{1, 2, \ldots, M\}$, we do the following:

- **Step 1**: Train the mth model $(\mathcal{A}_m, \mathcal{H}_m)$ on $\mathbb{D}_{\text{train}}$, yielding prediction function $g_m := \mathcal{A}_m(\mathbb{D}_{\text{train}}, \mathcal{H}_m)$.
- **Step 2**: Use g_m to predict on $\mathbb{D}_{\text{test}}$, yielding $\hat{\boldsymbol{y}}_m$.
- **Step 3**: Use $\hat{\boldsymbol{y}}_m$ to compute the out-of-sample error metric $oosRMSE_m$.

Then we choose $m_* := \underset{m \in \{1,2,\ldots,M\}}{\arg\min} \{oosRMSE_m\}$. We now compute

$$g_{\text{final}} := \mathcal{A}_{m_*}(\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{test}}, \mathcal{H}_{m_*})$$

and we describe its out-of-sample error using $oosRMSE_{m_*}$, which is a conservative estimate of its performance.

(c) [easy] Discuss the main limitation with using two splits to select a model.

In the two-split method, we "open" the test set $\mathbb{D}_{\text{test}}$ more than once, whereas $\mathbb{D}_{\text{test}}$ should be opaque, only opened once. The method is subject to overfitting if any of the M models happens too well on $\mathbb{D}_{\text{train}}$ and $\mathbb{D}_{\text{test}}$.

(d) [easy] Using three splits of the data, how would you perform model selection?

Let n be the number of data points in a data set $\mathbb{D}$, and let $\{(\mathcal{A}_m, \mathcal{H}_m)\}_{m=1}^M$ be M pre-specified models.

- **Step 0**: Select a value $K_{\text{test}} > 1$ and define

$$n_{\text{test}} := \frac{n}{K_{\text{test}}}$$

  Similarly, select a value $K_{\text{select}} > 1$, and define

$$n_{\text{select}} := \frac{n - n_{\text{test}}}{K_{\text{select}}}$$
$$n_{\text{train}} := n - n_{\text{test}} - n_{\text{select}}$$

- **Step 1**: Shuffle $\mathbb{D}$, and partition it into $\mathbb{D}_{\text{train}}$, $\mathbb{D}_{\text{select}}$, and $\mathbb{D}_{\text{test}}$, with n_{train}, n_{select}, and n_{test} data points, respectively.
- **Step 2**: For each model $(\mathcal{A}_m, \mathcal{H}_m)$, where $m \in \{1, 2, \ldots, M\}$:
  - Compute the prediction function $g_m := \mathcal{A}_m(\mathbb{D}_{\text{train}}, \mathcal{H}_m)$.
  - Use g_m to predict on $\mathbb{D}_{\text{select}}$, yielding predictions $\hat{\boldsymbol{y}}_m$.
  - Use $\boldsymbol{y}_{\text{train}}$ and $\boldsymbol{y}_m$ to compute $oosRMSE_m$.
- **Step 3**: Choose the model with the smallest out-of-sample error:

$$m_* := \underset{m \in \{1,2,\ldots,M\}}{\arg\min} \{oosRMSE_m\}$$

- **Step 4**: Use model m_* to compute the prediction function on $\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{select}}$:

$$g_{**} := \mathcal{A}_{m_*}(\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{select}}, \mathcal{H}_{m_*})$$

- **Step 5**: Use g_{**} to predict out-of-sample, that is, on $\mathbb{D}_{\text{test}}$. Then use the predictions to compute an out-of-sample error metric $oosRMSE_{**}$.

- **Step 6**: Use the entire data set to compute the final prediction function, still using model m_*:

$$g_{\text{final}} := \mathcal{A}_{m_*}(\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{select}} \cup \mathbb{D}_{\text{test}}, \mathcal{H}_{m_*})$$

The value $oosRMSE_{**}$ from step 5 is a conservative estimate of the out-of-sample performance of g_{final}.

(e) [easy] How does using both inner and outer folds in a double cross-validation nested resampling procedure improve the model selection procedure?

The nested resampling procedure is designed to allow $\mathbb{D}_{\text{test}}$ to vary rather than be fixed. After selecting a K_{test} proportion to be $\mathbb{D}_{\text{test}}^{(1)}$, the procedure performs a K_{select}-cross validation procedure, ultimately computing some $oosRMSE_1$. The procedure continues this way, eventually selecting a final K_{test} proportion to be $\mathbb{D}_{\text{test}}^{(K_{\text{test}})}$, and yielding a collection of out-of-sample error metrics $oosRMSE_1, \ldots, oosRMSE_{K_{\text{test}}}$. The model selection procedure aims to reduce the variation in the out-of-sample error metric when computing g_{final} on $\mathbb{D}$.

(f) [easy] Describe how g_{final} is constructed when using nested resampling on three splits of the data.

- **Step 0**: Select a value $K_{\text{test}} > 1$ and define

$$n_{\text{test}} := \frac{n}{K_{\text{test}}}$$

Similarly, select a value $K_{\text{select}} > 1$, and define

$$n_{\text{select}} := \frac{n - n_{\text{test}}}{K_{\text{select}}}$$
$$n_{\text{train}} := n - n_{\text{test}} - n_{\text{select}}$$

Also, shuffle $\mathbb{D}$.

- **Step 1**: For $k \in \{1, 2, \ldots, K_{\text{test}}\}$:
  - Select a portion of $\mathbb{D}$ to be $\mathbb{D}_{\text{test}}^{(k)}$, consisting of n_{test} points. Set aside the remaining $n - n_{\text{test}}$ points to use for $\mathbb{D}_{\text{train}}$ and $\mathbb{D}_{\text{select}}$.
  - Use a K_{select}-CV procedure to compute $\boldsymbol{e}_{\text{test}}^{(k)}$.
- **Step 2**: Aggregate the residual vectors across all folds into a single vector of length n:

$$\boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_{\text{test}}^{(1)} \\ \boldsymbol{e}_{\text{test}}^{(2)} \\ \vdots \\ \boldsymbol{e}_{\text{test}}^{(K_{\text{test}})} \end{bmatrix}$$

Use $\boldsymbol{e}$ to compute the $oosRMSE$ which serves as the conservative estimate of the performance of g_{final} (which we have yet to compute).

- **Step 3**: Apply select-CV, which is an extension of (d) where we fix $\mathbb{D}_{\text{test}}$ but let $\mathbb{D}_{\text{select}}$ vary, to compute g_{final}.

(g) [easy] Describe how you would use this model selection procedure to find hyperparameter values in algorithms that require hyperparameters.

We create a set Λ of M different hyperparameter values. For each hyperparameter value $\lambda_1, \lambda_2, \ldots, \lambda_M$ in Λ, we have an associated model $(\mathcal{A}_{\lambda_1}, \mathcal{H}), (\mathcal{A}_{\lambda_2}, \mathcal{H}), \ldots, (\mathcal{A}_{\lambda_M}, \mathcal{H})$. We apply either no-CV, or select CV, or nested resampling to compute the best model (and hence the best hyperparameter).

(h) [difficult] Given raw features $x_1, \ldots, x_{p_{raw}}$, produce the most expansive set of transformed p features you can think of so that $p \gg n$.

$$
\begin{aligned}
&x_1, x_2, \ldots, x_{p_{raw}} \\
&x_1^2, x_2^2, \ldots, x_{p_{raw}}^2 \\
&x_1^2, x_2^3, \ldots, x_{p_{raw}}^3 \\
&x_1x_2, \ldots, x_1x_{p_{raw}}, x_2x_3, \ldots, x_2x_{p_{raw}} \ldots x_{p_{raw}-1}x_{p_{raw}} \\
&\ln(x_1), \ln(x_2), \ldots, \ln(x_{p_{raw}}) \\
&e^{x_1}, e^{x_2}, \ldots e^{x_{p_{raw}}}
\end{aligned}
$$

(i) [easy] Describe the methodology from class that can create a linear model on a subset of the transformed features (from the previous problem) that will not overfit.

I would use the Greedy Forward Stepwise Modeling procedure.

- **Step 0**: Create a large set of derived features so that the set of candidate functions $\mathcal{H}$ is large. We did this in the previous part. Let p be the total number of features (after transformations). Begin with g_0 as our seed, so that

$$\mathcal{H}_0 = \{ w_0 \mid w_0 \in \mathbb{R} \}$$

- **Step 1**: For each of the p features, fit OLS to

$$\mathcal{H} = \{ w_0 + w_1 x_j \mid \boldsymbol{w} \in \mathbb{R}^2 \}$$

using $\mathbb{D}_{\text{train}}$. Then predict using $\mathbb{D}_{\text{select}}$ to obtain an $oosRMSE$ for each. Record the best feature, j_1, based on the best $oosRMSE$.

- **Step** N: At this point assume we have chosen N features. For each of the remaining $p - N$ features, fit OLS to

$$\mathcal{H} = \{ w_0 + w_1 x_{j_1} + \cdots + w_{j_N} + w_{j_{N+1}} x_{j_{\text{try}}} \mid \boldsymbol{w} \in \mathbb{R}^{N+2} \}$$

using $\mathbb{D}_{\text{train}}$. Predict using $\mathbb{D}_{\text{select}}$ to compute $oosRMSE$ for each, and record the best feature as j_{N+1}. Note that this removes any previously selected features from consideration.

Stop after N_0 steps, where N_0 is pre-specified maximum. Also, look at the results of each step, and when you are sure that the $oosRMSE$ is increasing, stop. It may start increasing due to overfitting, which is what we are trying to avoid. Having identified iteration t_* when the algorithm stops j_{t_*}, compute

$$g_* = \mathcal{A}(\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{select}}, \mathcal{H}_{t_*})$$

where $\mathcal{H}_{t_*}$ contains all the features that were selected through iteration t_*. Use g_{t_*} to predict on $\mathbb{D}_{\text{test}}$ to compute $oosRMSE_{t_*}$, which is a conservative out-of-sample performance estimate of the performance of

$$g_{\text{final}} = \mathcal{A}(\mathbb{D}_{\text{train}} \cup \mathbb{D}_{\text{select}}\mathbb{D}_{\text{test}}, \mathcal{H}_{t_*})$$

Problem 6

These are some questions related to the CART algorithms.

(a) [easy] Write down the step-by-step $\mathcal{A}$ for regression trees.

Start with $\mathbb{D}$. Let $x_{(i)}$ denote the ith smallest value in a vector $\boldsymbol{u}$.

- **Step 1**: Let $\boldsymbol{x}_k$ denote the kth vector, for $1 \leq k \leq p$, containing all n possible values for the kth feature. Let's consider every possible orthogonal-to-axis split, i.e:

$$\begin{aligned} x_1 \leq \boldsymbol{x}_{\cdot 1(1)}, x_1 \leq \boldsymbol{x}_{\cdot 1(2)}, \ldots, x_1 \leq \boldsymbol{x}_{\cdot 1(n-1)} \\ x_2 \leq \boldsymbol{x}_{\cdot 2(1)}, x_2 \leq \boldsymbol{x}_{\cdot 2(2)}, \ldots, x_2 \leq \boldsymbol{x}_{\cdot 2(n-1)} \\ x_p \leq \boldsymbol{x}_{\cdot p(1)}, x_p \leq \boldsymbol{x}_{\cdot p(2)}, \ldots, x_p \leq \boldsymbol{x}_{\cdot p(n-1)} \end{aligned}$$

  This means we have $n \cdot (p-1)$ possible splits. Note we do not consider, for example, $x_1 \leq \boldsymbol{x}_{\cdot 1(n)}$, or $x_2 \leq \boldsymbol{x}_{\cdot 2(n)}$, and so on because such a split would have *all* data on one side of the split, and hence there is no split at all.

- **Step 2**: Pick the best split, i.e., the one with the lowest error, where error is defined as:

$$SSE_{\text{weighted}} := \frac{n_L SSE_L + n_R SSE_R}{n_L + n_R}$$

  where n_L is the number of observations in the left child, n_R is the number of observations in the right child, SSE_L is the SSE in the left child, and SSR_R is the SSE in the right child. Note that since we are fitting a local optimization by picking the *best* split each time, this algorithm $\mathcal{A}$ is *greedy*.

- **Step 3**: Repeat steps (1) and (2) on the subset of $\mathbb{D}$ consisting of all the daughter nodes. Note that we can make at most n splits, otherwise we will have n dummies and we would surely overfit. Therefore, stop when the number of points in a child node is below some threshold N_0 (default $N_0 = 5$).

(b) [difficult] Describe $\mathcal{H}$ for regression trees. This is very difficult but doable. If you can't get it in mathematical form, describe it as best as you can in English.

Regression trees partition the input space into M regions $R_1, R_2, \ldots, R_M$. In region R_m, there are some data points from $\mathbb{D}$. If an incoming observation has its input fall R_m the predicted response will be the average c_m of the responses from $\mathbb{D}$ that fall in R_m. Therefore, the candidate functions will be:

$$\mathcal{H} = \left\{ \sum_{m=1}^{M} c_m \mathbb{I}_{x \in R_m} \mid c_m = \text{Mean}\{y \in R_m\} \right\}$$

(c) [harder] Think of another "leaf assignment" rule besides the average of the responses in the node that makes sense.

We can assign the median.

(d) [harder] Assume the y values are unique in $\mathbb{D}$. Imagine if $N_0 = 1$ so that each leaf gets one observation and its $\hat{\boldsymbol{y}} = y_i$ (where i denotes the number of the observation that lands in the leaf) and thus it's very overfit and needs to be "regularized". Write up an algorithm that finds the optimal tree by pruning one node at a time iteratively. "Prune" means to identify an inner node whose daughter nodes are both leaves and deleting both daughter nodes and converting the inner node into a leaf whose $\hat{\boldsymbol{y}}$ becomes the average of the responses in the observations that were in the deleted daughter nodes. This is an example of a "backwards stepwise procedure" i.e. the iterations transition from more complex to less complex models.

- **Step 1**: Start from n nodes, which with 1 observation.
- **Step 2**: For each pair of adjacent nodes (at the start, there are at most $\binom{n}{2}$ such pairs), combine the two into a single partition, and compute the resulting decrease in SSE. Prune the node that resulted in the smallest decrease in SSE.
- **Step 3**: Repeat step 2 until the number of observations in a node would result in more than N_0 observations, for some pre-specified N_0.

(e) [difficult] Provide an example of an $f(\boldsymbol{x})$ relationship with medium noise δ where vanilla OLS would beat regression trees in oos predictive accuracy. Hint: this is a trick question.

Let $f(x) = 2x + 1$. If we use OLS with the candidate set of linear functions

$$\mathcal{H} = \{\, w_0 + w_1 x \mid w_0, w_1 \in \mathbb{R} \,\}$$

then OLS will do better than regression trees, which would work by computing averages.

(f) [easy] Write down the step-by-step $\mathcal{A}$ for classification trees. This should be short because you can reference the steps you wrote for the regression trees in (a).

Start with $\mathbb{D}$. Let $x_{(i)}$ denote the ith smallest value in a vector $\boldsymbol{u}$.

- **Step 1**: Let $\boldsymbol{x}_k$ denote the kth vector, for $1 \leq k \leq p$, containing all n possible values for the kth feature. Let's consider every possible orthogonal-to-axis split, i.e:

$$\begin{aligned} x_1 \leq \boldsymbol{x}_{\cdot 1(1)}, x_1 \leq \boldsymbol{x}_{\cdot 1(2)}, \ldots, x_1 \leq \boldsymbol{x}_{\cdot 1(n-1)} \\ x_2 \leq \boldsymbol{x}_{\cdot 2(1)}, x_2 \leq \boldsymbol{x}_{\cdot 2(2)}, \ldots, x_2 \leq \boldsymbol{x}_{\cdot 2(n-1)} \\ x_p \leq \boldsymbol{x}_{\cdot p(1)}, x_p \leq \boldsymbol{x}_{\cdot p(2)}, \ldots, x_p \leq \boldsymbol{x}_{\cdot p(n-1)} \end{aligned}$$

  This means we have $n \cdot (p-1)$ possible splits. Note we do not consider, for example, $x_1 \leq \boldsymbol{x}_{\cdot 1(n)}$, or $x_2 \leq \boldsymbol{x}_{\cdot 2(n)}$, and so on because such a split would have *all* data on one side of the split, and hence there is no split at all.

- **Step 2**: Pick the best split, i.e., the one with the lowest error. We need a new error metric because SSE is not appropriate for a categorical response. Define the **Gini** error metric:

$$G_{\text{neighbor, avg}} = \frac{n_L G_L + n_R G_R}{n_L + n_R}$$

  where n_L is the number of observations in the left child, n_R is the number of observations in the right child. We get something similar to SSE by defining

$$G := \sum_{l=1}^{L} \hat{p}_\ell (1 - \hat{p}_\ell)$$

  where $\hat{p}_\ell$ is the proportion of observations of whose response is category ℓ in a given node:

$$\hat{p}_\ell := \frac{\text{number of observations with category } \ell \text{ in node}}{\text{total number of observations in node}}$$

  Note that since we fitting a local optimization by picking the *best* split each time, this algorithm $\mathcal{A}$ is *greedy*.

- **Step 3**: Repeat steps (1) and (2) on the subset of $\mathbb{D}$ consisting of all the daughter nodes. Note that we can make at most n splits, otherwise we will have n dummies and we would surely overfit. Therefore, stop when the number of points in a child node is below some threshold (default $N_0 = 1$, possibly because there are only L levels).

(g) [difficult] Think of another objective function that makes sense besides the Gini that can be used to compare the "quality" of splits within inner nodes of a classification tree.