— 5 —

Test set reuse

Statistics prescribes the iron vault for test data. But the empirical reality of machine learning benchmarks couldn't be further from the prescription. Repeated adaptive testing brings theoretical risks and practical power.

Source: The Emerging Science of Machine Learning Benchmarks. M. Hardt, 2025. URL: https://mlbenchmarks.org. Compiled on 2025-11-13.

The previous chapter covered powerful theoretical guarantees for the holdout method. Unfortunately, these guarantees hold only under one-time use. If a researcher builds a model based on prior interactions with the holdout set, the holdout set loses those formal guarantees. This chapter develops a theoretical model that accommodates how people actually use the holdout method in practice. We then work out generalization bounds in this setting, contrasting them with the results of the previous chapter.

5.1 Test set reuse in machine learning benchmarks

Most machine learning benchmarks have a fixed test set freely available to researchers. Benchmarks are typically publicly available and researchers may use the test set for evaluation purposes without any limits or restrictions. Software frameworks and data sharing platforms make it easy to download and evaluate on popular test sets. Model evaluations on test sets have become so common that it's easy to overlook just how many evaluations there are. Given the enormous amount of research and engineering work in machine learning, it's safe to say that popular test sets often support millions of evaluations.

But what's concerning is not so much the sheer number of test set evaluations. Keep in mind, the previous chapter developed guarantees for the holdout method that deteriorate very slowly with the number of evaluations. The price for making k evaluations is a factor $\sqrt{\log k}$ in the error bound. By this calculation, the price of a million evaluations is a factor 3.71 in the error of the holdout method. This alone would be easy to address by making test sets just a bit larger.

What's more concerning is that these guarantees simply don't apply to how the holdout method is used in practice. The strong theoretical guarantees require the test set to be *fresh*: The models we evaluate may not in any way depend on previous evaluations on the test set. The only safe way guarantee this is to keep the test set "in a vault"—as the textbooks prescribe—and use it only once in the end. But that's far from reality. Machine learning researchers and engineers instead work repeatedly and incrementally with various test sets. Model builders typically use test sets as part of their model building pipeline for continual model improvements. This might include trying out tweaks to the architecture, its hyperparameters, the optimization method, and various other moving pieces. Future design choices necessarily depend on past evaluations. Likewise, scientific papers report new achievements on

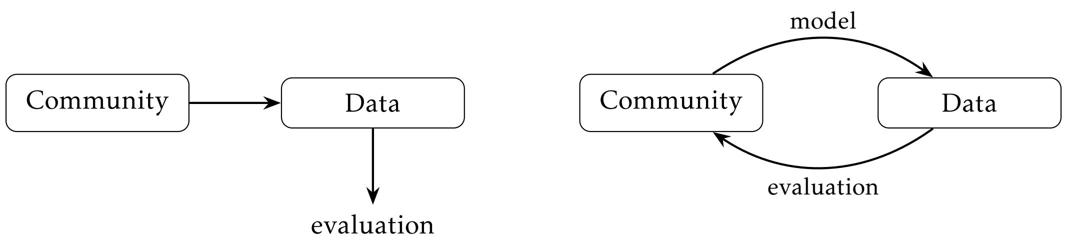


Figure 5.1: Left: Textbook view of non-adaptive data analysis. Right: Adaptive data analysis.

benchmark test sets. These reported numbers influence what scientists do in the future and what directions they choose to pursue.

By working incrementally with tests sets, researchers essentially do what Duda and Hart called *training on the test set*. They incrementally refine their model by repeated evaluations against the test set.

The practice of machine learning is, by nature, an *adaptive* process in which new steps depend on previous interactions with the data. Formally, this means there is a feedback loop between the model and the data. Evaluations on the data inform changes to the model, which in turn lead to new evaluations on the same data. We call this *adaptive data analysis*. In contrast, the traditional view of data analysis that we worked with in the last chapter is *non-adaptive*.

Adaptive data analysis

To formally reason about adaptive data analysis, it is helpful to frame the problem as an interaction between two parties. One party holds the dataset S. Think of this party as implementing the holdout method. The other party—called *analyst*—can *query* the dataset by requesting an estimate of the risk $R(f)$ for a given predictor f on the dataset S. In reality, the two parties might be one and the same researcher. Nevertheless, conceptually it's helpful to think of the problem as an interaction between these two parties.

The standard holdout method returns the empirical risk $R_S(f)$ given a query function $f\colon \mathcal{X} \to \mathcal{Y}$. But we'll allow for other methods that don't necessarily output the exact empirical risk. In fact, there are interesting alternatives to the standard holdout mechanism that enjoy stronger guarantees in the adaptive setting. Intuitively, these alternatives limit the amount of information about the holdout set revealed by each evaluation.

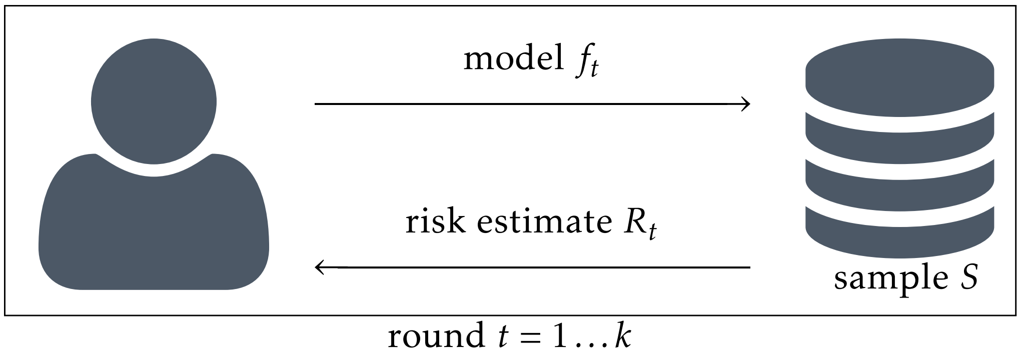


Figure 5.2: An adaptive analyst interacts with a dataset repeatedly.

Throughout this chapter, we restrict our attention to the case of the zero-one loss and binary prediction, although the theory extends to other settings.

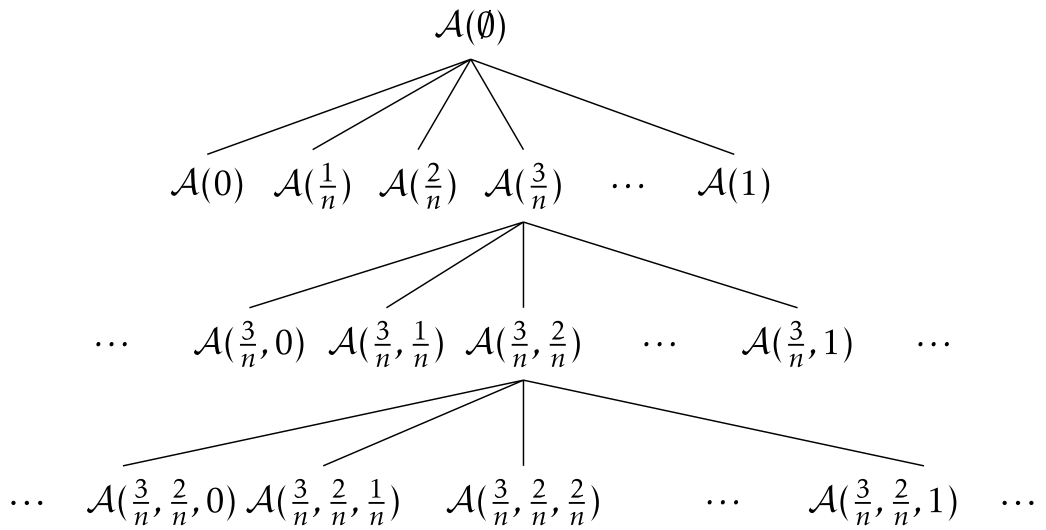
The two parties interact for some number k of rounds, thus creating a sequence of adaptively chosen predictors $f_1, \dots, f_k$. Keep in mind that this sequence depends on the dataset! In particular, when S is drawn at random, $f_2, \dots, f_k$ become random variables, too, that are in general not independent of each other.

Adaptive analysts

Formally, an adaptive analyst is an algorithm $\mathcal{A}$ that, given a sequence $f_1, R_1, \dots, f_t, R_t$ of queries and responses, returns a new query $f_{t+1} \colon \mathcal{X} \to \mathcal{Y}$, where we let $f_1 = \mathcal{A}(\emptyset)$ be the first query the analyst chooses. We assume that the analyst $\mathcal{A}$ is a deterministic algorithm.

A useful idea is to represent an adaptive analyst $\mathcal{A}$ as a tree. The root node is labeled by $f_1 = \mathcal{A}(\emptyset)$, i.e., the first function that the analyst queries without any context. The holdout mechanism then returns a response R_1. This response is generally a sample quantity like the empirical risk. Note that the empirical risk $R_S(f_1)$ can only take on $n+1$ possible values. This is because the zero-one loss can only take values in the set $\mathcal{R} = \{0, 1/n, 2/n, \dots, 1\}$. So, it makes sense to assume that $R_1 \in \mathcal{R}$ takes values in the same set. We can always achieve this by rounding R_1 to the nearest multiple of $1/n$ without changing the response significantly.

Each possible response value $r \in \mathcal{R}$ creates a new child node in the tree corresponding to the function $f = \mathcal{A}(r)$ that the analyst queries when receiving the response r to the first query f_1. This gives us $n+1$ children to the root node, one for each possible response $r \in \mathcal{R}$. Each child corresponds to one


$\mathcal{A}(\emptyset)$
$\mathcal{A}(0)$ $\mathcal{A}(\frac{1}{n})$ $\mathcal{A}(\frac{2}{n})$ $\mathcal{A}(\frac{3}{n})$ $\cdots$ $\mathcal{A}(1)$
$\cdots$ $\mathcal{A}(\frac{3}{n},0)$ $\mathcal{A}(\frac{3}{n},\frac{1}{n})$ $\mathcal{A}(\frac{3}{n},\frac{2}{n})$ $\cdots$ $\mathcal{A}(\frac{3}{n},1)$ $\cdots$
$\cdots$ $\mathcal{A}(\frac{3}{n},\frac{2}{n},0)$ $\mathcal{A}(\frac{3}{n},\frac{2}{n},\frac{1}{n})$ $\mathcal{A}(\frac{3}{n},\frac{2}{n},\frac{2}{n})$ $\cdots$ $\mathcal{A}(\frac{3}{n},\frac{2}{n},1)$ $\cdots$
Figure 5.3: Constructing a tree of depth k and degree $n+1$ representing an adaptive analyst. Each node corresponds to the predictor the analyst chooses based on the responses seen so far.

function. In this manner, recursively continue the process until you have a tree of depth k and degree $n+1$.

Note that this tree depends only on the analyst $\mathcal{A}$ and how it responds to all the possible transcripts that can occur in the interaction with a holdout set. The tree does not depend on the random sample, however. The tree is therefore data-independent. It's an explicit representation of the algorithm $\mathcal{A}$. This property is a useful tool for proving generalization bounds in the adaptive setting.

5.2 *Guarantees of the holdout method under adaptivity*

We can now derive guarantees for the holdout method in the adaptive analyst model. The idea is to fix an analyst and apply the analysis from Chapter 4 to all the functions appearing in the tree corresponding to the analyst.

Proposition 1. *For any sequence of k adaptively chosen predictors $f_1,\dots,f_k$, the holdout method satisfies with probability $1-\delta$,*

$$\mathbb{P}\left\{\max_{1\le t\le k}|\Delta_S(f_t)| \le \sqrt{\frac{k\log(4(n+1)/\delta)}{2n}}\right\} \ge 1-\delta\,.$$

Proof. Fix an adaptive analyst $\mathcal{A}$ and the corresponding tree. The tree is of size

$$1+(n+1)+(n+1)^2+\cdots+(n+1)^k = \frac{(n+1)^{k+1}-1}{n} \le 2(n+1)^k\,.$$

Let F be the set of functions appearing at any of the nodes in the tree. Since each node has one function, we have

$$|F| \le 2(n+1)^k\,.$$

Moreover, the set of functions is fully determined by the algorithm $\mathcal{A}$ and therefore does not depend on any sample. Now, pick a random sample S of size n. The model selection guarantee for the holdout method from the previous chapter gives us for every $\delta > 0$,

$$\mathbb{P}\left\{\max_{f\in F}|\Delta_S(f_t)| \le \sqrt{\frac{\log(2|F|/\delta)}{2n}}\right\} \ge 1-\delta\,.$$

Plugging in the upper bound on $|F|$,

$$\log(2|F|/\delta) \le k\log(4(n+1)/\delta)\,.$$

This completes the proof.

□

In the typical regime where $k \ge \log(n)$ is at least logarithmic in n and $\delta \ge 1/n$ is not too small, the bound on the maximum error simplifies to $O(\sqrt{k/n})$. This contrasts with the bound $O(\sqrt{\log(k)/n})$ that holds in the non-adaptive setting.

Analyst	Error bound
non-adaptive	$O(\sqrt{\log(k)/n})$
adaptive	$O(\sqrt{k/n})$

The dependence on k in the adaptive setting is exponentially worse than in the non-adaptive setting. This raises the question if there's a way to do better.

Climbing the leaderboard without looking at the data

So far, the guarantee of the standard holdout mechanism in the adaptive case is exponentially worse in k compared with the non-adaptive case. As it turns out, this is unavoidable in the worst case. What we'll work out is a worst-case *lower bound* on the gap between risk and empirical risk in the adaptive setting. To prove such a lower bound it is enough to exhibit an adaptive analyst that forces a large gap.

Indeed, there is a fairly natural sequence of k adaptively chosen predictors, resembling the practice of ensembling, on which the empirical risk diverges from the risk by at least $\Omega(\sqrt{k/n})$. This matches the upper bound from the previous section up to a constant factor. In particular, with $k \approx n$ queries, you can force a constant generalization gap. The lower bound is *worst-case*: It only holds for this one analyst and doesn't say anything about the typical behavior of the holdout method.

Throughout, focus on zero-one loss in a binary prediction problem. The core idea extends to many other settings.

Overfitting by ensembling:

1. Choose k random binary predictors $f_1, \dots, f_k \colon \mathcal{X} \to \{0,1\}$.
2. Compute the set $I = \{i \in [k] \colon R_S[f_i] < 1/2\}$.
3. Output the predictor $f = \text{majority}\{f_i \colon i \in I\}$ that takes a majority vote over all the predictors computed in the second step.

The key idea of the algorithm is to select all the predictors that have accuracy strictly better than random guessing. This selection step creates a bias that gives each selected predictor an advantage over random guessing. The majority vote in the third step amplifies this initial advantage into a larger advantage that grows with k. If we want to be a bit more clever, we don't have to throw away any functions at all. If $R_S[f_i] > 1/2$, then flipping all the bits gives $R_S(1-f_i) < 1/2$. So, include $1-f_i$ in the ensemble. This saves a factor two in the number of queries.

In practice, we can do a bit better still by weighting each function with its advantage over random guessing. The larger the advantage, the larger the weight of the function in the ensemble. This makes intuitive sense and leads to some improvements. What this algorithm essentially does is *boosting*. Boosting is a well-known technique for turning weak predictors into strong predictors.

The next proposition confirms that indeed this strategy finds a predictor

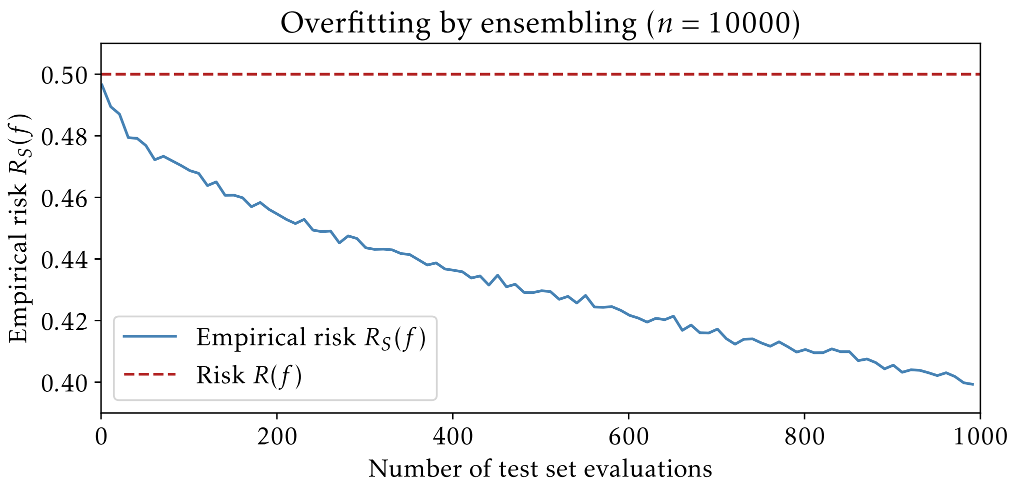


Figure 5.4: Overfitting by ensembling on a test set of size $n = 10000$. Thousand queries force a generalization gap of 10%.

whose empirical risk is bounded away from 1/2 (random guessing) by a margin of $\Omega(\sqrt{k/n})$. Since the predictor does nothing but taking a majority vote over random functions, its risk is of course no better than 1/2.

Proposition 2. *For sufficiently large $k \le n$, overfitting by ensembling returns a predictor f whose classification error satisfies with probability* 1/3*,*

$$R_S(f) \le \frac{1}{2} - \Omega(\sqrt{k/n}).$$

In particular, $\Delta_S(f) \ge \Omega(\sqrt{k/n})$.

Proof. (Sketch)

The formal proof is a bit tedious, but it's good to develop the intuition.

On a randomly chosen function the number of errors on the test set S is distributed according to the binomial distribution $B(n, 1/2)$. We expect to see $n/2$ errors and one standard deviation is $\sqrt{n}/2$. This is because the variance of $B(n,p)$ is $np(1-p)$.

When we observe that $R_S(f_t) < 1/2$ for a randomly chosen f_t, we know that it makes fewer than $n/2$ errors on the test set. But apart from this condition, the errors are still randomly distributed. So, we get the distribution of $B(n, 1/2)$ conditional on falling below its mean. The mean of this conditional distribution is smaller than $n/2 - \sqrt{n}/2$. If it's below the mean, it'll be so by at least one standard deviation more than half the time.

This is the first part of the argument. Our selection step biases the functions to make about $\sqrt{n}$ fewer errors on the test set than a random function.

How many functions do we select? In other words, how large is the index set I? Since the binomial distribution is symmetric around its mean, the probability of any given function being selected is about 1/2. So we expect the index set to be of size $m = k/2$. In fact, m concentrates around $k/2$. Therefore, we can be pretty sure that we select at least, say, $k/4$ functions. Up to constant factors, we can think of m and k as being the same.

To summarize, we know that we select many functions and each selected function is a bit better than random guessing.

The last step is to show that the majority vote amplifies this small advantage over random guessing. This is the same kind of argument we do when we reason about ensembling methods. Taking the majority vote over many experts—each a bit better than random guessing—results in a more accurate prediction.

So, consider the majority predictor $f = \text{majority}\{f_i \colon i \in I\}$. Fix a data point $(x,y) \in S$. What is the probability that f makes a mistake on x? For convenience, assume $y = 1$. The case of $y = 0$ is the same. By definition, the majority classifier makes a mistake if more than half the functions in I vote 0. Formally,

$$\sum_{i\in I} f_i(x) < \frac{m}{2}\,.$$

What is the chance that this happens? The errors that each f_i makes on the test set are randomly located, since our selection step is invariant under permutation of the test set. Therefore, each function we select is correct on x with probability $1/2 + \epsilon$ for some $\epsilon > 1/\sqrt{n}$.

This means that the random variable $X = \sum_{i\in I} f_i(x)$ follows the binomial distribution $B(m, 1/2+\epsilon)$. Its mean and variance are

$$\mathbb{E}\,X = \frac{m}{2} + \epsilon m\,, \qquad \mathbb{V}\,X = m(1/2+\epsilon)(1/2-\epsilon) \le m/4\,.$$

For the majority vote f to be wrong, the random variable X has to be below its mean by an additive ϵm. Since a standard deviation is less than $\sqrt{n}$, this deviation is equivalent to $\epsilon\sqrt{m}$ in units of standard deviation. Note that

$$\epsilon\sqrt{m} = \sqrt{\frac{m}{n}} \approx \sqrt{\frac{k}{n}}\,.$$

Since we assume $k \leq n$, we're talking about less than one standard deviation. For large enough n and constant p, the binomial distribution $B(n,p)$ behaves very much like a normal distribution with mean np and variance $np(1-p)$. Within one standard deviation of its mean, the normal distribution acts a lot like a uniform distribution. In particular, we can calculate that a deviation of $\sqrt{k/n}$ below its mean has probability less than $1/2 - c\sqrt{k/n}$ up to some positive constant $c > 0$ in front of the $\sqrt{k/n}$ term.

So, we can conclude that the probability of a mistake by our majority vote is at most

$$\mathbb{P}\left\{\sum_{i\in I} f_i(x) < \frac{m}{2}\right\} \leq \frac{1}{2} - c\sqrt{\frac{k}{n}},$$

for some positive constant $c > 0$. Written differently, this means

$$R_S(f) \leq \frac{1}{2} - c\sqrt{\frac{k}{n}}.$$

That's what we wanted to show. This concludes the proof sketch.

□

Zooming out again, ensembling by majority voting shows that the problem of "training on the testing data" is real. In the worst case, holdout data can quickly lose its guarantees.

5.3 Alternatives to the holdout method

In the worse case, the error of the holdout method grows with $\sqrt{k}$ not $\sqrt{\log k}$ as in the non-adaptive case. This is exponentially worse. If this pessimistic bound manifested in practice, popular benchmark datasets would quickly become useless. Is there anything we can do about the pessimistic lower bound we just encountered?

It turns out, there is. So far, we've analyzed the standard holdout method that returns the exact empirical risk $R_S(f)$ given a query function f. There's hope that alternative mechanisms might have stronger guarantees. For example, we could release an approximate answer rather than an exact answer. We have to be a bit careful though. The argument from the previous section extends to the case where the holdout method only gives approximate answers so long as these are within $o(1/\sqrt{n})$ from the exact answer. The reason is that $1/\sqrt{n}$ is roughly one standard deviation of the empirical risk.

If all answers deviate less than one standard deviation, the argument is roughly the same. So, the solution isn't as easy as rounding the answer to four digits of precision. This is a common heuristic in machine learning competitions, but it has no formal guarantees.

Nevertheless, a related idea works. If we carefully add certain noise variables to each answer, we can improve dependence on k from $\sqrt{k}$ to $k^{1/4}$. This alternative holdout method permits a quadratic number $k = n^2$ of queries before it becomes useless in the worst-case. The proof of this result is surprisingly tricky and requires several tools that are beyond the scope of this chapter. The key idea is based on the privacy guarantee *differential privacy*: If you can make the holdout mechanism differentially private, then no adaptive analyst can learn about the test set enough to overfit to it. This is a theorem that forms the basis for many alternative mechanisms that uses noise addition to achieve the quadratic number of queries. Unfortunately, researchers also showed that in the worst-case no holdout mechanism can give an answer to more than a quadratic number of queries with small constant error on every query.

The bounds in the previous chapter are optimistic. They are strong, but they require an assumption that is clearly violated in practice. The bounds in this chapter go about it from the other end. They allow for powerful adaptive analyses, like those you'll see in practice. The downside is that we end up with fairly pessimistic worst-case guarantees. If we typically encountered these bounds in practice, holdout sets would have a serious problem. Both perspectives are useful though. One tells us the best that we can hope for. The other shows the worst that could happen.

In the next two chapters we'll look at the empirical phenomena around test set reuse in machine learning research. The practice of benchmarking seems to successfully avoid the worst-case bounds from this chapter. Chapter 8 collects possible explanations for why this is. In that chapter, we'll also see another powerful alternative to the standard holdout method. We end this chapter on a closer look at the problem of adaptivity in the context of statistics.

5.4 *Freedman's paradox*

The problem with the holdout method we just saw has a close cousin in the field of statistics called Freedman's paradox. The statistician David Freedman pointed out this problem in 1983, although he thought that his observation was neither new, nor a paradox.

Freedman's problem routinely arises in the context of variable selection for statistical modeling. If we first select variables in a data-dependent way and then fit a model on the selected variables, the model will appear to fit better than it actually does. At a high-level this is directly analogous to the ensembling procedure we discussed above. The details are different though and require a bit of statistical nomenclature.

Freedman considers a linear regression problem

$$y = X\beta + \eta\,,$$

where the error term $\eta \sim N(0, \sigma^2)$ is sampled from a centered Gaussian distribution with variance σ^2. The matrix X has shape $n \times d$ where n is the number of observations and d is the number of features. The vector $\beta \in \mathbb{R}^d$ corresponds to the *unknown* solution to the system of equations. We only observe the matrix X and the vector y.

Least squares regression attempts to find an approximate solution to the linear equations by solving the optimization problem:

$$\min_{\hat\beta \in \mathbb{R}^d} \quad \|X\hat\beta - y\|^2\,.$$

Let $\hat y = X\hat\beta$ denote the optimal solution to least squares. Due to the noise in the system of equations, we can't hope to recover β exactly.

The situation statisticians want to avoid is that $\beta = 0$ but somehow we come away thinking that $\beta \neq 0$. The case $\beta = 0$ corresponds to the situation where there is no signal in the data. The vector y is just a random normal vector with no dependence on X. Statisticians call this a *null model*. A null model is a data-generating distribution corresponding to the case where there is nothing to discover. In contrast, the case $\beta \neq 0$ means that there is some relationship between X and y that we would like to discover and report.

To test if we're dealing with the null model, statisticians use hypothesis tests: Compute a *test statistic* of the data. If it exceeds a threshold, *reject* the null model. A common hypothesis test is the *F-test* that uses the *observed F-value* as a test statistic:

$$F^{\text{obs}} = \frac{\|\hat y - \bar y\|^2/d}{\|y - \hat y\|^2/(n-d-1)}\,.$$

Here, $\bar{y}$ is the mean $\bar{y} = \sum_{i=1}^{n} y_i$ of the observations. We call F^{obs} the *observed* F-value, since it's the one we compute from our sample.

The observed F-value is the test statistic. The way the hypothesis test works is that we're going to reject the null model if the observed F-value is above a certain threshold, call τ. Under the null model, the observed F-value follows an F-distribution. It doesn't matter what exactly this distribution is. It's just some continuous distribution we can compute.

The probability that the F-value we observe exceeds some threshold τ is given by $1 - \mathrm{CDF}(\tau)$, where CDF is the cumulative density function of the F-distribution. The probability that the F-value exceeds the value we observed is therefore

$$p = 1 - \mathrm{CDF}(F^{\text{obs}}).$$

Here, the tail probability p is called the *p-value* of the test. The p-value is the probability under the null model that the F-distribution exceeds the value F^{obs} we observe.

That means p-values are tail probabilities under the null model.

A fact called *integral probability theorem* implies that $\mathrm{CDF}(F^{\text{obs}})$ follows the uniform distribution $\mathrm{Uniform}([0,1])$ over the interval $[0,1]$. Therefore, the distribution of a p-value under the null model is the uniform distribution. This is the only fact about p-values we'll need.

So far, everything is working as intended. The p-values we see under the null model are uniformly distributed as they should. The chance of seeing a p-value as small as 0.05 is, by construction, at most 5%. This is the now infamous *significance level* at which statisticians will *reject the null hypothesis*. The idea is that seeing $p \leq 0.05$ renders the null model implausible.

Variable selection

Our goal is not only to avoid rejecting the null model when $\beta = 0$. This alone would be easy: Never reject anything. But our goal is also to actually reject the null model when $\beta \neq 0$ and so we should. The higher the dimension d of our regression problem, the harder it generally is to find signal in the data. An F-test on *all* variables—when there are many—is safe, but it's unlikely to find any signal in the data.

Therefore, it is common practice to first select some promising variables from the set of all variables, before fitting a model on the selected variables. To select variables, we can again perform hypothesis tests. Specifically, we can test individual coefficients in the regression model to see if they are significantly large using the T-statistic:

$$T_j^{\text{obs}} = \hat{\beta}_j / s_j, \quad \text{with} \quad s_j^2 = \hat{\sigma}^2 (X^T X)_j^{-1} j$$

Call the p-value associated with the j-th test p_j. That is, p_j is the probability of seeing a value as extreme as T_j under the null model, i.e., $\beta = 0$ and Gaussian errors. Again, under the null model all these p-values are uniformly distributed.

Now consider the following two step procedure:

Regression after variable selection:

1. Select all variables with $p_j < 0.25$. Call that index set I.
2. Solve a regression problem on X_I, the $n \times |I|$ matrix where we retain only the columns in X corresponding to selected variables.

Since the p-values in the first step are all uniformly distributed, the selection step simply picks a random subset of roughly $d/4$ variables. Any variable is equally likely to be chosen. Starting from $\beta = 0$, of course, there's still no signal in the linear system given by X_I and y. So, we still shouldn't reject the null hypothesis.

What Freedman observed, however, is that the p-value corresponding to an F-test on the final model is sharply biased towards smaller p-values. We're therefore much more likely to reject the null hypothesis ($\beta = 0$) than we should be.

The problem Freedman discovered is now often called *inference after selection*. We first select variables and then we'd like to do *inference*, i.e., compute p-values or confidence intervals, on the selected variables. There's nothing specific about F-tests or T-tests in this example. It's a fundamental problem with two-stage data-dependent analyses.

A safe way to do this is to perform the two steps on independent data sets. So, in a sense, sample splitting solves this two step problem. Statisticians have also come up with a number of clever methods to do the two steps correctly on the same sample.

More broadly, Freedman's paradox relates to a practice that later became known as *p-hacking*. The term describes the practice deliberately choosing an analysis in a data-dependent way so as to find a significant p-value. We'll return to this problem in the next chapter. Freedman's observation foreshadowed problems to come.

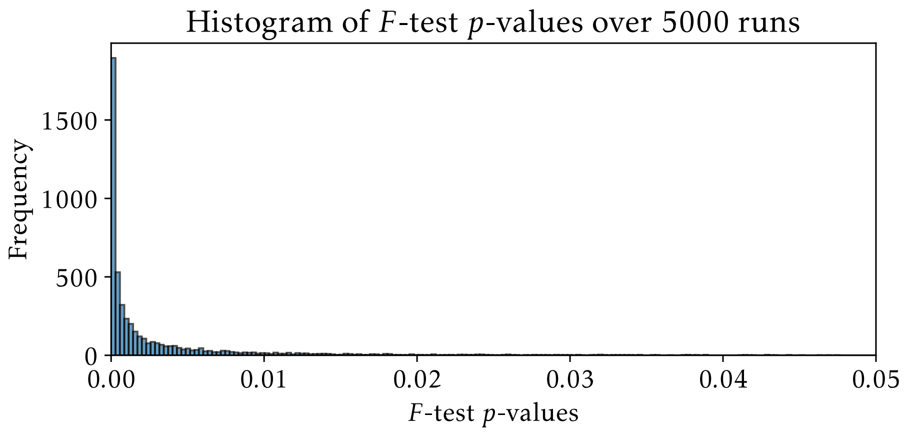


Figure 5.5: Biased p-values in Freedman's two-stage regression with $n = 1000$ samples and $d = 50$ variables

Notes

Dwork et al. initiated the study of *adaptive data analysis*.[1–3] Tools from differential privacy give variants of the holdout method that have stronger guarantees under adaptive use. The idea is to add a sufficient amount of noise to each empirical risk computation on the holdout data. Whereas the plain holdout method supports linear number of queries in the worst case, a holdout method based on noise addition can support a quadratic number of queries. Bassily et al.[4] improved the error bounds compared with the suboptimal bounds by Dwork et al. Under cryptographic hardness assumption, however, no holdout method can support more than a quadratic number of queries.[5]

Blum and Hardt[6] study holdout reuse in the context of machine learning benchmarks. The key observation they make is that ranking is different from evaluation: If the goal is to identify the best model from a sequence of model evaluations, a variant of the holdout method supports an exponential number of queries. We will return to this result in Chapter 8. In the same paper, they also point out how the basic holdout method fails in the adaptive setting due to overfitting via ensembling. Hardt provided a proof of this proposition in a subsequent paper.[7] Dwork et al. gave a similar argument for how adaptively fitting a linear model can lead to the same deviation between risk and empirical risk.[2] Feldman, Frostig, and Hardt further develop the connection between boosting and overfitting.[8]

In a 2015 blog post, I illustrated how overfitting by ensembling may be applied in a real data science competition.[9] On the Heritage Health Prize competition—that featured 3 million dolars in prize money—a variant of the algorithm can climb the public leaderboard form rank 146 to 6 with 700 queries. This requires solving an additional technical challenge. The algorithm in this chapter starts from accuracy 0.5. This isn't helpful, if the top of the leaderboard is already much more accurate than that. Instead of picking random functions in the ensemble, the fix is to instead pick functions that randomize over two well-performing models: Pick a random clasisifer by choosing for each input one of the two models at random. The ensemble will then improve relative to the average accuracy of the two models. You can think of the method in this chapter as randomizing over the all ones and the all zeros classifier.

Freedman described the problem with p-values after selection in a short 1983 paper.[10] Freedman neither called it a paradox, nor did he believe that the observation was new. Nevertheless, his note has been quite influential. There's now much work on this problem in statistics under the name *post-selection inference*[11] or *inference after selection*.[12] Hastie, Tibshirani, and Friedman discuss a variant of Freedman's paradox in the context of cross validation in Chapter 7.10.2 of their book.[13]

While adaptivity brings theoretical challenges, many recognize the need for permitting adaptivity in statistical analysis. In particular, John Tukey and and George Box both advocated for allowing incremental progress in data analysis.[14,15] See also the argument by Gelman and Loken[16] that we'll return to in the next chapter.

Bibliography

1. Dwork, C. *et al.* The reusable holdout: Preserving validity in adaptive data analysis. *Science* **349,** 636–638 (2015) (↑ 15).
2. Dwork, C. *et al. Preserving statistical validity in adaptive data analysis* in *ACM Symposium on Theory of Computing (STOC)* (2015), 117–126 (↑ 15).
3. Dwork, C. *et al. Generalization in adaptive data analysis and holdout reuse* in *Neural Information Processing Systems (NeurIPS)* (2015) (↑ 15).
4. Bassily, R. *et al. Algorithmic stability for adaptive data analysis* in *ACM Symposium on Theory of Computing (STOC)* (2016), 1046–1059 (↑ 15).
5. Hardt, M. & Ullman, J. *Preventing false discovery in interactive data analysis is hard* in *Annual Symposium on Foundations of Computer Science (FOCS)* (2014), 454–463 (↑ 15).
6. Blum, A. & Hardt, M. *The ladder: A reliable leaderboard for machine learning competitions* in *International Conference on Machine Learning (ICML)* (2015), 1006–1014 (↑ 15).
7. Hardt, M. Climbing a shaky ladder: Better adaptive risk estimation. *arXiv:1706.02733* (2017) (↑ 15).
8. Feldman, V., Frostig, R. & Hardt, M. *The advantages of multiple classes for reducing overfitting from test set reuse* in *International Conference on Machine Learning (ICML)* (2019), 1892–1900 (↑ 15).
9. Hardt, M. *Competing in a Data Science Contest Without Reading the Data* `https://blog.mrtz.org/2015/03/09/competition.html`. Accessed: 2025-11-13. Mar. 2015 (↑ 16).
10. Freedman, D. A. & Freedman, D. A. A note on screening regression equations. *the american statistician* **37,** 152–155 (1983) (↑ 16).
11. Lee, J. D., Sun, D. L., Sun, Y. & Taylor, J. E. Exact post-selection inference, with application to the lasso. *The Annals of Statistics* **44,** 907 (2016) (↑ 16).
12. Fithian, W., Sun, D. & Taylor, J. Optimal inference after model selection. *arXiv:1410.2597* (2014) (↑ 16).
13. Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Corrected 12th printing)* (Springer, 2017) (↑ 16).
14. Tukey, J. W. We need both exploratory and confirmatory. *The american statistician* **34,** 23–25 (1980) (↑ 16).
15. Box, G. Scientific method: The generation of knowledge and quality. *Quality Progress* **30,** 47 (1997) (↑ 16).
16. Gelman, A. & Loken, E. The garden of forking paths: Why multiple comparisons can be a problem, even when there is no “fishing expedition” or “p-hacking” and the research hypothesis was posited ahead of time. *Department of Statistics, Columbia University* **348,** 3 (2013) (↑ 16).