Data Science Looks at Discrimination

A toolkit for investigating bias in race, gender, age and so on

Norman Matloff, Project Lead
Taha Abdullah Arjun Ashok
Shubhada Martha Aditya Mittal
Billy Ouattara Jonathan Tran
Brandon Zarate

2023-09-16

Table of contents

Overview

Discrimination is a key social issue in the US and in a number of other countries. There is lots of available data with which one might investigate possible discrimination. But how might such investigations be conducted?

Our **dsld** package provides both graphical and analytical tools for this purpose. It is widely applicable; here are just a few use cases:

- Quantitative analysis in instruction and research in the social sciences.
- Corporate HR analysis and research.
- Litigation involving discrimination and related issues.
- Concerned citizenry.

This book provides a tutorial regarding applicable methodology, using **dsld** examples (though it is not a user manual for the package).

Prerequisite background

In addition to having rudimentary skill in R, the user should have a very basic knowledge of statistical inference–mean, variance, confidence intervals and tests, and histograms. A short, intuitive, "bare bones" refresher, focused on confidence intervals, is given in Appendix A.

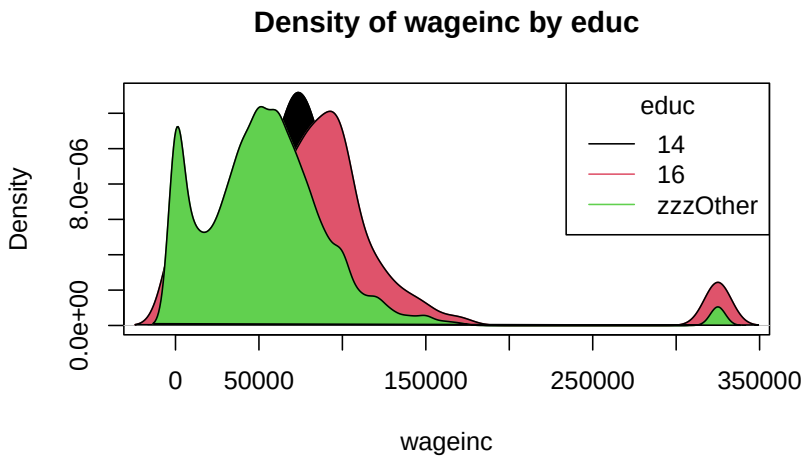
Mathematical background, e.g. linear algebra, calculus and probability theory, are **not** required, and concepts are explained via intuitive language.

The dsld package

The R package **dsld**, which this tutorial uses for examples, has two aims:

- To enable exploratory analysis of possible discrimination effects through various graphical and tabular functions.
- To enable formal statistical analysis of such effects via both enhanced access to general R functions such as **lm()** and **glm()** and various functions from the **qeML** machine learning package.

Python wrappers are included for many functions.


Density of wageinc by educ
Density
0.0e+00
8.0e−06
0
50000
150000
250000
350000
wageinc
educ
14
16
zzzOther

1 Introduction and Motivating Examples

To set the stage, consider the following:

1.1 UC Berkeley discrimination claims

UC Berkeley was accused of discriminating against female applicants for graduate school, and indeed the overall acceptance rate for women was lower than that for men. This seemed odd, given Berkeley's liberal reputation.

However, upon breaking the data down according to the program students were applying to, it was found that in every department, the female acceptance rate *within that department* was either higher than the male rate or of similar level. The problem: women were applying to more selective programs, causing their overall rate to below that of men.

This data is included in R, as the built-in dataset **UCBAdmissions**.

1.2 US Census data

The **svcensus** dataset is a subset of US census data from back in 2000, focusing on six engineering occupations. The question at hand is whether there is a gender pay gap. Again, the overall pay for men is higher, by about 25%. But what if we break things down by occupation? Though it does turn out that some occupations pay more than others, and that men and women

Included in the **dsld** package.

are not distributed evenly among the occupations, there still is a gender pay gap, of about 16%.

1.3 Commonality

In both examples, we have an outcome variable Y of interest—acceptance rate and wage income—and a sensitive variable S, which was gender in both examples. But in both cases, we were concerned that merely comparing mean Y for each gender was an oversimplication, due to a possible *confounder* C—department in the first example, occupation in the second— that is related to both variables. Failure to take confounders (there can be more than one, and usually are so) into account can lead to spurious "relations" between S and Y.

As noted earlier, this book avoids technical definitions, keeping to the intuitive, including our notion here of a confounder. Actually, a fully precise definition is rather problematic.

> **Confounder adjustment analysis**
>
> So, in general, we wish to *estimate the impact* of a sensitive variable S on an outcome variable Y, but *accounting for confounders* C. Let's call such analysis "confounder adjustment."

The above discussion summarizes the goal of Part I of this book. Now contrast the above examples with a different kind, which will concern Part II:

1.4 COMPAS recidivism data

COMPAS is a commercial machine learning software tool for aiding judges in sentencing defendants convicted of a crime. The tool's main function is to predict recidivism by a defendant. But a 2016 *Pro Publica* investigation found that the tool to be racially biased; African-American defendants tended to be given harsher ratings—i.e. higher estimated probabilities of recidivism—than similarly situated white defendants.

Northpointe, the firm that developed COMPAS, rejected the *Pro Publica* analysis, and we are not supporting either side

here.

But if the COMPAS tool were in fact biased, how could the analysis be fixed?

Note that both *Pro Publica*'s analysis and that of Northpointe used methodology like that presented in this book.

A key point is that any remedy must not only avoid using race directly, but must also minimize the impact of variables O that are separate from race but still correlated with it, known as *proxies*. If, say, educational attainment is correlated with race, the inclusion of that variable in our analysis will mean that race is still playing a role in our analysis after all. On the other hand, eliminating all proxies may severely compromise our predictive ability.

> **Fair ML analysis**
>
> Thus our goal is to *predict the outcome* variable Y, without using the sensitive variable S, while making only limited use of the proxy variables O.

1.5 Summary: the two kinds of discrimination analysis covered here

Note the difference between accounting for confounders on the one hand, and fair ML on the other. Here is a side-by-side comparison:

aspect	confounder adjustment	fair ML
goal	estimate an effect	predict an outcome
harm	comes from society	comes from an algorithm
side info	adjust for confounders	limit impact of proxies

Part I, on confounder adjustment, focuses on discrimination examples but is applicable to confounder adjustment applications in general. Part II, on fair ML, is more specific to discrimination settings.

1.6 Summary of symbols

The symbol X will at first denote all the variables other than Y and S. The general terminology is that Y is variously called the *outcome variable*, *target variable* or *dependent variable*; the X variables are known collectively as *covariates*, *features* or *independent variables*.

Among the variables in X, we will separate out some to play the role of C (Part I) or O (Part II), after which X will refer to all variables other than Y, S, C and O.

In terms of the above examples, here are the roles of the variables:

example	Y	C	S	O
UCB admits	acceptance	department	gender	-
Census	wage	e.g. occupation	gender	-
COMPAS	recidivate	-	race	e.g. education

1.7 A word before getting started

As noted, the book consists of two main topics:

- Part I, adjusting for confounders
- Part II, fair ML

So, let's get started. One key point first, though:

> **i** Notes on modeling, the role of the software, etc.
>
> This book makes use of the **dsld** software, but is definitely not a user manual for that package. Instead, it is a guide to the statistical principles, with the software playing a supporting role.
> Any statistical model is approximate. And virtually any relation of interest in practice is nonzero. Modern statistical thinking places reduced emphasis on significance tests and p-values, and asks instead whether A has an effect on

The pioneering statistician George Box famously said, "All models are wrong, but some are useful."

B that is substantial enough to be of interest.
A related point is that, unlike some readers may have experienced in some statistics courses, real-world statistical analysis is not conducted in a formulaic, "Step 1, Step 2,..." manner. Instead, decisions on say, which model to use, must be made by you, the analyst, based on your overall assessment of the available information. The software cannot make your decisions for you.

2 Part I: Adjustment for Confounders

```
library(dsld)
data(svcensus)
```

How do we adjust for confounders? The most common approach involves *linear models*, with which we express the mean Y for given values of the X, C and S variables in a linear form.

> **i** Important term: the *regression function*
>
> The relation of mean Y to the X, C (or O) and S variables, that is the conditional mean of Y given the other variables, is formally called the *regression function* of Y on ithose variables. Our first model below, which expreses mean income as a function of age and gender, will assume this relation as linear, but the term *regression function* is general.
> Indeed, one commonality between statistics and machine learning (ML) methods is that both types of analysis typically involve estimation of the regression function, even though they differ in the uses to which they put such estimates: In statistics, the goal can be either effect estimation (e.g. the impact of gender on wages) or prediction, while the latter is almost always the goal of ML.

2.1 Linear model example: a simple gender wage gap analysis

Consider the **svcensus** data example in Section 1.2 above, investigating a possible gender pay gap. So Y is wage and S is

gender. We might treat age as a confounder C, reasoning as follows. Older workers tend to have more experience and thus higher wages, and if there is an age differential in our data, say with female workers tending to be older, this may mask a gender pay gap: If men make more money than women of the same age, but women tend to be older, the gender and age effects may largely cancel out.

So, let's take the set of confounders C to consist of age, and for simplicity in this introductory example, not include any other confounders, such as occupation, and not include any other variables X.

2.1.1 Initial analysis

Our linear model would thus be

$$\text{mean W} = \beta_0 + \beta_1 \text{ A} + \beta_2 \text{ M}$$

where W is wage, A is age and M is an indicator variable, with M = 1 for men and M = 0 for women. The parameters β_i are estimated by fitting the model to the data:

```
svcensus1 <-
   svcensus[,c(1,4,6)]  # age, wage, gender
z <- dsldLinear(svcensus1,'wageinc','gender')
coef(z)  # print the estimated coefficients b_i
```

```
$gender
(Intercept)         age  gendermale
 31079.9174    489.5728  13098.2091
```

The column svcensus$gender is an R factor. Our function **dsldLinear** calls R's **lm**, which replaces that column by a dummy variable **gendermale**, our M above. If a factor has f levels, i.e. represents f categories, R will create f-1 dummies.

The familiar R **lm** function is called behind the scenes, but the call includes an addiitonal argument S, in this case gender. This is a theme common to many **dsld** functions: A call to these functions ultimately translates to calls to functions in base R or other packages – but with extra discrimination-specific information in the input and output.

Let's use b_i to denote our estimated β_i. So for instance b_1 = 489.5728 is our estimate of the unknown population parameter

β_1.

The b_i are computed using *least squares*, which find the b_i that minimize the sum of the square of the differences between the observed Y and fitted Y.

2.1.2 Interpretation of β_2

Lots in the output to discuss, which we will gradually cover below. For now, note that the estimate b_2 turns out to be about \$13,000, which is the (estimated) wage gap, if any. Here's why:

Always keep in mind that statistical quantities are only estimated, since we work only with sample data from some population, real or conceptual. Hence the need for standard errors, confidence intervals and so on.

Under the model, the mean wage for, say, 36-year-old men is

$$\beta_0 + 36\ \beta_1 + 1\ \beta_2$$

while for women of that age it is

$$\beta_0 + 36\ \beta_1$$

The difference is β_2. But if we look at, for instance, people of age 43, the mean wages for men and women are

$$\beta_0 + 43\ \beta_1 + 1\ \beta_2$$

and

$$\beta_0 + 43\ \beta_1$$

and the difference *is still* β_2.

> **"The" effect of gender**
>
> Thus we can speak of β_2 as *the* gender wage gap, at any age. According to the model, younger men earn an estimated \$13,000 more than younger women, with the *same-sized* gap between older men and older women.

The above approach to dealing with confounders is a very common one. But it raises questions, such as:

- What are the assumptions underlying that model? And how might we check whether they are (approximately) valid?

- We chose only one C variable here, age. We might also include occupation, as noted earlier. In some datasets, might have dozens of possible confounders. How do we choose which ones to use in our model? And for that matter, why not use them all?
- The above model, in which the gender wage gap was uniform across all wages, may not be adequate. How can we determine this, and what alternative models might we use?

2.1.3 Statistical inference

The full output of **dsldLinear()** goes to the heart of discrimination analysis, enabling statistical inferences on differences in levels of the sensitive variable S. Let's take a look, continuing from the above code:

```
summary(z)
```

```
$`Summary Coefficients`
    Covariate   Estimate StandardError PValue
1 (Intercept) 31079.9174    1378.08158      0
2         age   489.5728      30.26461      0
3  gendermale 13098.2091     790.44515      0

$`Sensitive Factor Level Comparisons`
         Factors Compared Estimates Standard Errors P-Value
Estimate    male - female  13098.21        790.4451       0
```

The first half of this output is from **lm()**, which is called by **dsldLinear()**. The second half is the "value added" material from **dsld**.

So, an approximate 95% confidence interval for the gender wage gap is

$$13098.2091 \pm 1.96 \text{ x } 790.4451$$

or (11548.94,14647.48).

Since the estimated gender gap here is simply b_2, the CI could of course have also been obtained directly from the **lm** half of the output. But with an S having more than two levels, e.g. race, the **dsld** enhancement is quite valuable.

2.1.4 With-interactions model

As discussed above, in our model

$$\text{mean W} = \beta_0 + \beta_1 \text{ A} + \beta_2 \text{ M}$$

we identified b_2 as *the* difference in mean wage between men and women, regardless of age, so that for instance:

> According to the model, younger men earn about $13,000 more than younger women, with the same-sized gap between older men and older women.

But that may not be true. On the contrary, gender discrimination and age discrimination may interact. It may be, for instance, that the gender gap is small at younger ages but much larger for older people.

Interaction between two types of discrimination is called *intersectionality* by some analysts.

Technically, the with-interactions model adds a product term:

$$\text{mean W} = \beta_0 + \beta_1 \text{ A} + \beta_2 \text{ M} + \beta_3 \text{ AM}$$

So for example, the gender pay gap for people of age 36 is

$(\beta_0 + \beta_1\ 36 + \beta_2\ 1 + \beta_3\ 36)$ - $(\beta_0 + \beta_1\ 36\ ) =$

$\beta_2\ 1 + \beta_3\ 36$

And at age 43, the gap is

$\beta_2\ 1 + \beta_3\ 43$

So, this model does indeed allow for interaction between age and gender.

However, this added-product-term is a bit abstract, and it is easier (and approximately equivalent) to simply fit two linear models, one for men and one for women.

> **Note**
>
> The **dsldLinear** function includes an argument **interactions**. The default value is FALSE, but if TRUE, it fits separate linear models for each level of S. An additional argument 'newData' is now needed, through which

the user specifies a data frame consisting of one or more (X,C) values at which to compare the effect of S.

```
newData <- data.frame(age=c(36,43))
z <- dsldLinear(svcensus1,'wageinc','gender',interactions=T,
   newData)
summary(z)
```

```
$female
    Covariate   Estimate StandardError PValue
1 (Intercept) 30551.4302    2123.44361      0
2         age   502.9624      52.07742      0

$male
    Covariate  Estimate StandardError PValue
1 (Intercept) 44313.159    1484.82216      0
2         age   486.161      36.02116      0

$`Sensitive Factor Level Comparisons`
  Factors Compared New Data Row Estimates Standard Errors
1    female - male            1 -13156.88        710.9696
2    female - male            2 -13039.27        710.7782
```

In setting that **newData** argument, we need one row for every variable other than Y and S. In this case, there is just one such variable, age. So, in the above call, we are fitting two linear models, one for men and another for women, then comparing regression function values at the two specified ages.

So the gender pay gap is estimated to be -13156.88 at age 36, and -13039.27 at age 43, differing by only about $100. The estimated gap between ages 36 and 53, not shown, is larger, close to $300, but it seems there is not much interaction here. The no-interactions model should be adequate after all.

The classical approach to choosing between the no-interaction and with-interaction models is of course to test the hypothesis $H_0 : \beta_3 = 0$. As noted earlier and detailed in Section 2.6, modern practice discourages such approaches, which can be misleading.

2.1.5 Linearity and other assumptions

As noted, linear models are ubiquitous in observational data analysis. Open any professional journal in medicine, sociology,

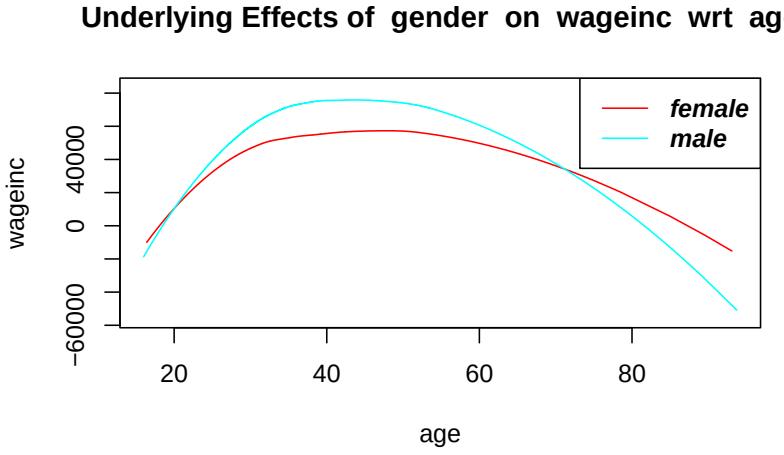
economics and so on, and you'll see many applications of this methodology. But how would one check that most basic assumption, the linearity of the mean Y for given X, C (or O) and S values?

> **i** Assumptions–not just a formality
>
> Assumptions *matter*. They are never perfectly satisfied, but failure to be even approximately valid can mean deciding that there is no discrimination when it actually is there, or vice versa. It can mean bad medication being declared by the government as good, or vice versa. In litigation, if a key expert witness is exposed by opposing counsel as not having checked the assumptions in his/her analysis, the side for which the witness was testifying will likely lose the case on the spot.

Typically, linearity is checked graphically. A common approach involves plotting the *residuals*, which are the differences between the fitted line and the Y values. Here, though, we use another graphical approach, via a **dsld** function that may be more informative.

Returning to our earlier setting with just S = gender for our example, we run

```
dsldConditDisparity(svcensus1,'wageinc','gender','age')
```



The function plots a smoothed graph of Y against a user-specified C variable, once for each level of S. "Smoothing" here groups data points of similar age, and plots their average Y value. So, the call here says, "Plot smoothed wage income against age, for each gender."

The model has mean Y being a linear of function of age, so we should expect to see approximate straight lines here. Yet the relation certainly looks nonlinear, possibly reflecting age discrimination against both very young and very old workers. We are already investigating one kind of discrimination here, gender, so again for simplicity let's just keep age as a confounder C rather than a sensitive variable S.

2.1.6 Updated model

But we must do something about the substantial nonlinearity we've discovered, and one possible remedy is to add an age^2 term be added to the equation:

mean W = $\beta_0 + \beta_1$ A + β_2 A^2 + β_3 M

Let's fit the updated model:

Adding a squared term does not make our model nonlinear, as it is still linear in the β_i; if we, say, double each of those, the entire expression is doubled, the definition of linearity. The model is nonlinear in age but linear in the β_i.

```
svcensus1$age2 <- svcensus1$age^2
z <- dsldLinear(svcensus1,'wageinc','gender')
coef(z)  # print the estimated coefficients b_i
```

```
$gender
  (Intercept)           age    gendermale          age2
-104196.65579    7251.30962   15270.56685     -79.16059
```

So we see that the original wage gap figure of about $13,000 was incorrect; it's actually estimated to be over $15,000, so the original model underestimated the gap by about 15%.

We see in this example that misspecifying a linear model can have a major impact on its accuracy. As usual, though, we will try to keep things simple, and thus use only the original linear model in our subsequent examples below.

2.1.7 Other assumptions

Other than linearity, the standard errors reported by **lm()** also assume that variance of wage income is approximately constant across ages and genders. Lack of this property has some effect on the accuracy of reported standard errors, and thus on the validity of confidence intervals and p-values but this can be adjusted via the so-called *sandwich* operation, an option in **dsldLinear()**.

It is also assumed that wage income has a normal/gaussian distribution at each level, but the Central Limit Theorem's implication for the sums created by **lm()** is that the b_i are in fact approximately normal. The normality assumption is not very important.

2.2 S may consist of more than one factor

In introducing this example, we noted the need to start simple. Let's move away from that a bit.

In the **svcensus** data, both age and gender are potential areas of discrimination. We can treat both as such by combining these two R factor variables into one "super R factor," as follows.

We'll need to discretize age, and since federal law on age discrimination uses age 40 as the definition of "older," let's use that as an example:

```
svc <- svcensus
age <- svc$age
age <- ifelse(age >= 40,'40+','under40')
age <- as.factor(age)
head(age)
```

```
[1] 40+     40+     under40 40+     40+     40+
Levels: 40+ under40
```

```
svc$age <- age
```

Now let's make our "super factor," using a function from **qeML**:

```
AgeGend <- cartesianFactor('svc',c('age','gender'))
svc$AgeGend <- AgeGend
head(svc)
```

```
      age     educ occ wageinc wkswrkd gender        AgeGend
1     40+ zzzOther 102   75000      52 female     40+.female
2     40+ zzzOther 101   12300      20   male       40+.male
3 under40 zzzOther 102   15400      52 female under40.female
4     40+ zzzOther 100       0      52   male       40+.male
5     40+ zzzOther 100     160       1 female     40+.female
6     40+ zzzOther 100       0       0   male       40+.male
```

We have only three education codes here, with 14 and 16 representing a Master's degree and 16 a PhD, and 'zzzOther' denoting all others. Since this dataset consists of Silicon Valley programmers and engineers, the vast majority of "others" have a Bachelor's degree.

We no longer need the original age and gender columns, so we'll delete them and then try some analysis:

```
svc$age <- NULL
svc$gender <- NULL
w <- dsldLinear(svc,'wageinc','AgeGend')
summary(w)
```

```
$`Summary Coefficients`
               Covariate   Estimate StandardError       PValue
1            (Intercept)   2482.330    1529.05560 1.044954e-01
2                 educ16   8194.121    1717.11275 1.823745e-06
3           educzzzOther -14554.025     747.54506 0.000000e+00
4                 occ101   1703.896     899.52840 5.819707e-02
5                 occ102  13627.089     829.14115 0.000000e+00
6                 occ106   1351.235    2013.86471 5.022422e-01
7                 occ140  11409.603    1643.94237 3.910205e-12
8                 occ141  11407.453    1039.62019 0.000000e+00
9               wkswrkd   1313.971      20.77242 0.000000e+00
10       AgeGend40+.male   9834.035    1064.86666 0.000000e+00
11 AgeGendunder40.female  -8176.053    1229.58483 2.942069e-11
```

```
12   AgeGendunder40.male   -408.267    1030.00692 6.918298e-01

$`Sensitive Factor Level Comparisons`
               Factors Compared Estimates Standard Errors      P-Value
1         40+.female - 40+.male -9834.035       1064.8667 0.000000e+00
2   40+.female - under40.female  8176.053       1229.5848 2.942069e-11
3     40+.female - under40.male   408.267       1030.0069 6.918298e-01
4     40+.male - under40.female 18010.088        990.2130 0.000000e+00
5       40+.male - under40.male 10242.302        705.2504 0.000000e+00
6 under40.female - under40.male -7767.786        951.1388 2.220446e-16
```

Ah, this is a more nuanced probe than the ones above in which we simply used age as a confounder. The male-female differences at both the older and younger age levels, about $9800 and $7800, are both substantial, but smaller than the $13,100 overall figure we obtained earlier.

This is an example of *Simpson's Paradox*, in which an overall average effect might be larger than the terms that make up the average. In fact, the algebraic signs may change, as we saw with the UC Berkeley admissionss data in Section 1.1.

Note too the impact of age within genders. Older women made about $8200 more than younger women, but for men the figure was rather larger, about $10,200.

On the other hand, this analysis is probably too coarse with respect to age, as it does not reveal the negative impact of age well beyond 40. It may be worth trying a finer discretization of age, say, 35-, 35-55 and 55+.

2.3 The Logistic Model

The *logistic* model is an example of a *generalized linear model*, whose name stems from it having a linear component in the formula, as will be seen below.

2.3.1 General form of the model

Just as linear models are the most commonly used for numeric Y, in the binary-Y case the go-to standard is the logistic model. To introduce it, let's first consider a very simple prediction problem, in which Y is gender, say 1 for male, 0 for female, and X is simply income, using the **svcensus** data (no C here).

> **Probability is a special case of a mean**
>
> Note that mean Y is now the probability that Y = 1. That's because the average of a bunch of 1s and 0s is the proportion of 1s. In the data 1,0,1,1 for instance, the mean is (1+0+1+1) / 4 = 3/4, and indeed 3/4 of those numbers are 1s; if we were to choose one of those four numbers at random, the probability of obtaining a 1 would be 3/4.

Now, what is the logistic model (often called "logit"), and where does it come from? To motivate this, suppose that within each gender Y, the income level X has a normal (Gaussian) distribution, the familiar "bell-shaped" curve, with the same standard deviation of X within each gender. Then it turns out that one can show mathematically that

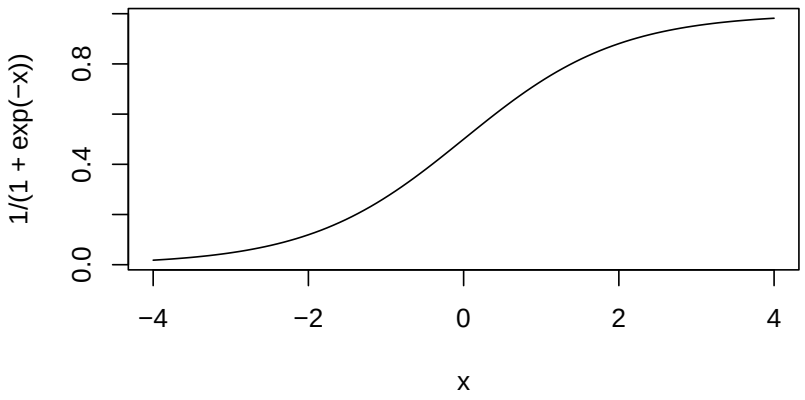
probability male =

$$\frac{1}{1+e^{-(\beta_0+\beta_1 income)}}$$

So the probability has a linear component to it, $\beta_0+\beta_1 income$, hence the term "generalized" linear model..

That formula follows the form of the *logistic function*, $f(t) = 1/[1+e^{-t}]$, which has the shape of an S-curve:

```
curve(1/(1+exp(-x)),-4,4)
```



So at least the model retains a linear component, with much the same interpretability. For instance, if $\beta_2 > 0$, then the usual monotonicity relation holds, i.e. the higher the income, the greater the probability that the person is male.

The estimates b_i of the population values β_i are obtained via a method generalizing the least-squares method used in the linear case.

In the case of multiple predictor variables, the logistic form can again be motivated by considering within-group distributions:

There is a similar situation for the linear case. If Y and the predictor variables have a multivariate normal distribution, one can show that mean Y as a function of the features is linear in the features, with homogeneous conditional variance and normal conditional distributions.

Say we predict gender from age and wage income. If the latter two variables have a *bivariate normal* distribution (two-dimensional histogram has a 3-D bell shape) with the same variance matrices within each gender, it turns out that we again get a logistic form:

probability male =

$$\frac{1}{1 + e^{-(\beta_0 + \beta_1 age + \beta_2 income)}}$$

Now, what about that assumption of the normal distributions and so on? Just as many regression functions for numeric Y in practice are roughly linear, in predicting binary Y the S-curve model is often roughly valid. Moreover, the logistic model has two desirable properties for predicting binary Y:

- Its value is between 0 and 1, appropriate for modeling a probability.
- As noted earlier, the expression $1/[1 + e^{-t}]$ is increasing in t, which we wish to model when our predictors have monotonic relations with Y.

The point is that the logistic (popularly referred to as "logit") is often a good model for binary-Y settings in general.

2.3.2 We no longer have a no-interactions case

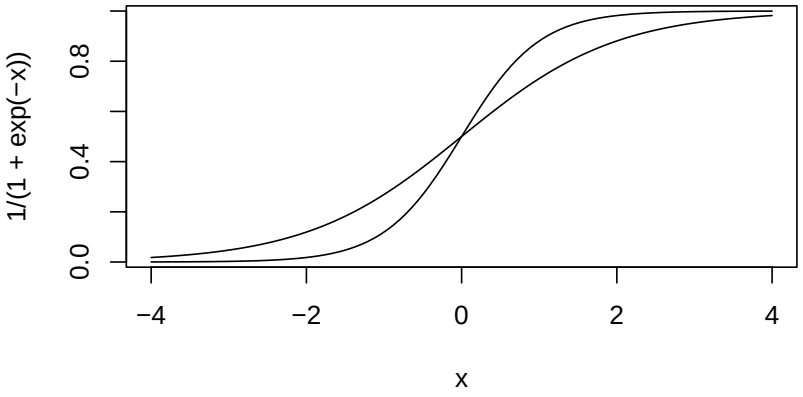
In our earlier linear model, predicting wage income from age and gender,

$$\text{mean W} = \beta_0 + \beta_1 \text{ A} + \beta_2 \text{ M}$$

recall that here β_2 has the nice interpretation of there being a uniform gender gap, independent of age. Similar statements hold for β_1; a 1-year increase in age, for instance, on average is associated with a β_1 increase in mean income, identically for either gender.

Geometrically, if we were to plot separate male and female regression lines against age, the male and female lines would be parallel. That's not possible in the logit case, as logit curves cannot be parallel:

```
curve(1/(1+exp(-x)),-4,4)
curve(1/(1+exp(-2*x)),-4,4,add=TRUE)
```



The curves are no longer parallel; they even cross. They typically don't cross in applied settings, but they are always non-parallel.

The implication of this is the same as in the linear case. We cannot speak of "the" impact of S on Y, as it will not be the same at different levels of the X and C variables. So there is no direct analog of the no-interactions case for linear models, in which we could speak of β_2 as being "the" gender pay gap.

Some books motivate the logistic approach as the *log-odds* ratio, meaning in this example that the logarithm of the ratio (probability male) / (probability female) is linear in age and income.

$$\log [\text{probability male / probability female}] = \beta_0 + \beta_1 \text{ A} + \beta_2 \text{ W}$$

So here we do have a formulation in which the impact of wage is the same for any age level. But this log scale is hard to interpret.

Note that a log-odds measure can take on any value betwen $-\infty$ and ∞.

To be sure, we still choose whether or not to have product terms, of S with other predictors, by setting **interactions** to TRUE or FALSE, respectively. In the TRUE case, the interactions become even more complex.

2.3.3 Example: mortgage data

This dataset and its documentation are included in **dsld** from the **SortedEffects** package. package. The issue here is whether racism played a role in mortgage denials in the Boston area. As this is a binary outcome, we might consider a logit model.

We have done small modifications, to create R factors for some columns.

```
data(mortgageSE)
z <- dsldLogit(mortgageSE,'deny','black',yesYVal='1')
summary(z)
```

```
$`Summary Coefficients`
     Covariate    Estimate Standard.Error       PValue
1  (Intercept) -4.68021873     0.73384720 1.798289e-10
2       p_irat  5.15585388     1.06258659 1.221161e-06
3       black1  0.64088712     0.18356747 4.806957e-04
4      hse_inc -0.83228276     1.28336021 5.166497e-01
5     loan_val  0.20205974     0.70171872 7.733852e-01
6       ccred2  0.72353389     0.21417086 7.293486e-04
7       ccred3  0.89094547     0.31542102 4.733628e-03
8       ccred4  1.56403348     0.33660382 3.375955e-06
9       ccred5  1.18009433     0.24808769 1.967220e-06
```

```
10       ccred6  1.55681669     0.23373704 2.728094e-11
11       mcred2  0.32167763     0.19927066 1.064678e-01
12       mcred3  0.45713207     0.47817345 3.390741e-01
13       mcred4  0.21616464     0.66923366 7.466928e-01
14      pubrec1  1.26756099     0.21044970 1.711007e-09
15      denpmi1  4.61566552     0.56425832 2.837083e-16
16     selfemp1  0.63445399     0.21665184 3.406571e-03
17      single1  0.41614600     0.15962315 9.132523e-03
18      hischl1 -1.11171191     0.42286015 8.562887e-03
19     probunmp  0.05565252     0.03459848 1.077202e-01
20       condo1 -0.12673301     0.17493093 4.687744e-01
21     ltv_med1  0.42773600     0.21618162 4.786155e-02
22    ltv_high1  1.50099888     0.42183272 3.732914e-04

$`Sensitive Factor Level Comparisons`
          Factors Compared Estimates Standard Errors      P-Value
Estimate            1 - 0 0.6408871       0.1835675 0.0004806957
```

So, being Black resulted in an average increase in log-odds of about 0.6409. A 95% confidence interval is 0.6409 $\pm$ 1.96 x 0.1836 = (0.2810,1.0008). By comparison, being self-employed, for instance, has a similar estimated coefficient of about 0.6355. Again, interpretation on a log scale is difficult – imagine trying to convince a nontechnical person that logarithms of probabilities are meaningful – but at least it is interesting that being Black is a similar handicap to being self-employed; the latter trait means lack of a steady income, thus a possible risk to not repay the loan.

2.4 Machine learning approaches

> **Important note**
>
> In spite of the fancy name *machine learning*, the methods discussed here, *k-Nearest Neighbor* estimation, *random forests* and *gradient boosting*, were originally developed in the statistics field, and have long been used by statisticians.

We referred to the quantities β_i above as "parameters." The linear and logistic models are thus called *parametric* models. In each case, the regression function is modeled as linear, "S curve-shaped" and so on, that can be described with just a few parameters.

But these models make assumptions, such as assuming the regression function is approximately linear. It would be nice to have available methods that don't rely on such assumptions.

Machine learning (ML) algorithms typically do not assume the regression function has any particular form. They are thus "safer," though on the other hand they are less interpretable than, say, that β_2 quantity in our no-interactions linear model above. Nor are standard errors available for regression function values at given points.

ML is mainly concerned with prediction, while we here are interested in estimation of effect sizes. However, ML algorithms either directly or indirectly do their prediction by estimating the regression function, as we do here, so they can be quite useful in our context.

2.4.1 The k-Nearest Neighbor algorithm

We'll focus here on the k-Nearest Neighbor (kNN) algorithm, as it is the simplest to explain. Say we are predicting people's weights, knowing only their heights. We have training data, on which we know both the height and weight of each person. Faced with a new case of a person 70.2 inches tall but with unknown weight, we find the people in our training set whose heights are close to 70.2 inches, and average their weights. That average is our predicted weight for the new person, as well as our estimate for the value of the regression function at height = 70.2.

How is "close to" defined? Should we look at the closest 5 people, the closest 50, or what? That number is k, the number of nearest neighbors. The number k here is called a *hyperparameter* of the algorithm. It's chosen by the analyst, just like the analyst chooses the number of bins in plotting a histogram.

If in forming a histogram, we use too many bins, each bin may have a very small sample, thus producing a ragged graph. If we have too few bins, there may be rather disparate values in the same bin, producing a graph that is *too* smooth.

The situation with kNN is very similar. If k is too small, we average over a small sample; if it is too large, then we might be including data points that are far from, in the above situation, 72.5, thus not representative.

Typically k is chosen by trying several different values of k, then using the one that best predicts in the holdout set. Once we've settled on k, we might use the full dataset, no holdout.

Most **dsld** functions have an argument **holdout**. Say we set it to 1000 and our dataset has 5000 rows. Then the function will randomly partition our data into a *training set* of 4000 rows and the holdout set of 1000 rows. The give algorithm, say kNN, will fit the training data, and then use the result to predict the holdout data. The latter is considered "fresh" data, so we obtain an unbiased estimate of our predictive power.

We can then run the algorithm with various values of k, and settle on the one that predicts the holdout data best.

The function **fineTuning** in the **regtools** package can be used to automate the process of trying many values of the hyperparameters, averaging over many holdout sets.

```
set.seed(9999)
z <- dsldML(svcensus,'wageinc','gender',
   qeMLftnName='qeKNN',opts=list(k=50))  # default holdout size
print(z)
```

```
$testAccs
  female     male
21336.37 27196.96

$comparisons
           age     educ occ wkswrkd female  male
10502 37.43326 zzz0ther 101      52  50614 65596
19689 47.75017 zzz0ther 100      52  57420 61382
13609 42.46283 zzz0ther 102      52  66122 83352
12419 46.67517       14 100      52  71954 85050
14402 52.66371 zzz0ther 106      52  60590 73848
```

The first three arguments are as usual. The fourth states that the ML algorithm we wish to use is kNN; the **qeML** package also includes random forests, boosting and neural networks. The **opts** argument states which non-default values we wish to use for arguments to the ML function; the default k actually is 25.

The comparison cases are by default a random sample of 5 rows of the training set. In order to obtain consistent results each time this book is processed by Quarto, we've called R's **set.seed** function.

It's important to keep in mind that the output here is not simply for 5 persons; each row represents a subpopulation. Thus for instance in that first row of output, we estimate among *all* Silicon Valley techies of age 37.4, with a Bachelor's degree or less, in occupation 101, and having worked 52 weeks, women's average wage is 48,914 and for men it's 61,156.

As before, we see a substantial gender wage gap, but now we can feel more confident about it, since this analysis is unencumbered by assumptions on the form of the regression function. On the other hand, as noted, there is no easy way to obtain standard errors.

2.4.2 The random forests algorithm

In the above example, in which we are predicting wage income, a random forests (RF) analysis would generate a large number of *decision trees*. As explained below, each tree would give us a predicted income, and our final prediction would simply the average of all those predicted values.

So, what is a decision tree? It's just a flow chart. Consider this example involving vertebral disease. Y is categorical, with values NO (normal), SL and DH. To predict a new case, we look at the X variables (which have the nondescript names V1 through V6) one at a time, and branch through the tree accordingly.

For example, if the new case to be predicted has V6 of at least 16, we branch right, and immediately decide to predict that this patient's disease status is SL. If on the other hand, V6 is

less than 16, we look at V4. If it's under 28, we immediately predict status DH, and so on.

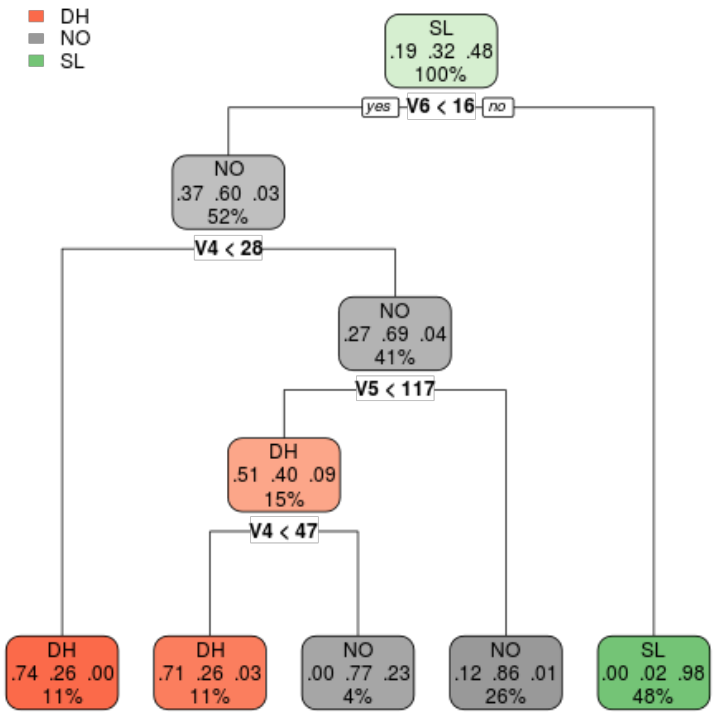


Figure 2.1: A decision tree

The order in which variables are considered in building a tree is random for each tree. So, while V6 is considered first in the tree displayed here, in some other tree it may be, say, V2. The split points, e.g. 16 in the root node in the picture, are obtained through formulas that depend on the data in complex (though not terribly deep) ways.

There is no "universally best" predictive algorithm, with performance of a given algorithm being dependent on the given dataset. So, instead of doing a kNN analysis, we could try RF:

```
set.seed(9999)
w <- dsldML(svcensus,'wageinc','gender',
   qeMLftnName='qeRF')
```

```
Loading required namespace: randomForest
```

```
print(w)
```

```
$testAccs
  female     male
20748.78 27955.95

$comparisons
           age     educ occ wkswrkd   female     male
10502 37.43326 zzzOther 101      52 51937.17 72126.99
19689 47.75017 zzzOther 100      52 47494.44 67178.71
13609 42.46283 zzzOther 102      52 71107.03 83606.01
12419 46.67517       14 100      52 56353.56 69237.44
14402 52.66371 zzzOther 106      52 58431.52 63392.62
```

Though still indicating a gender wage gap, the results here are rather different from what we obtained with kNN. Which one is better in this particular application? Again, we may decide this on the basis of prediction accuracy (**testAcc** values) on a *holdout set*, as follows:

2.4.3 Other ML methods

The most famous class of ML these days is of course *neural networks*, especially the refined version *deep learning*. These have been found highly useful in image recognition and language processing.

In contrast to these fields of application, the ML people refer to the types of data we see in this book, where each row represents, say, one person, as *tabular* data. Their go-to method for such data is *gradient boosting* (GB), more specifically XGBoost, accessible in **qeML** via **qeXGBoost**. GB is again a

tree-based method, but it works by starting with a primitive tree and repeatedly refining it.

2.5 Example: The Law School Admissions dataset

This is the dataset **law.school.admissions**, included in **dsld** via **fairml**. It's quite well-known in the ML world, but its full provenance is unclear. For instance, the age variable skews far to middle-aged and older, seemingly not consistent with the data's description on Kaggle.

Possible the 'age' variable is birth year. The data are from 1991, so an 'age' value of 69 would mean born in 1969, now age 22.

Thus it should be kept in mind that this is just an illustration of methodology, and firm conclusions about the legal education process should not be drawn. The data concern students who were admitted to law school, so in spite of the title, it's not about the admissions process itself.

The main variables of interest here are:

- **decile1**, **decile2**: Student's standing after Year 1 and Year 3 of law school.
- **fam_inc**: Apparently the income level of the family in which the law students was raised in.
- **lsat**: Score on the LSAT, a major law school admissions test.
- **ugpa**: Undergraduate GPA.
- **gender**: Gender.
- **race1**: Primary racial group.
- **cluster**: Level of prestige of the law school.
- **bar**: After law school, did this person pass the bar examination?

Here's what the data looks like:

```
data(law.school.admissions)
head(law.school.admissions)
```

```
   age decile1 decile3 fam_inc lsat ugpa gender race1 cluster fulltime   bar
2   62      10      10       5 44.0  3.5 female white       1        1  TRUE
3   62       5       4       4 29.0  3.5 female white       2        1  TRUE
6   61       8       7       3 37.0  3.4   male white       1        1  TRUE
7   60       8       7       4 43.0  3.3 female white       1        1  TRUE
9   57       3       2       4 41.0  3.3 female white       4        1  TRUE
11  59       1       1       4 24.5  2.2   male white       3        1 FALSE
```

```
names(law.school.admissions)
```

```
 [1] "age"      "decile1"  "decile3"  "fam_inc"  "lsat"     "ugpa"
 [7] "gender"   "race1"    "cluster"  "fulltime" "bar"
```

```
lsa <- law.school.admissions
```

2.5.1 Is the LSAT fair?

There has been concern that the LSAT and other similar tests are biased against Black and Latino students, and might otherwise have racial issues. Let's investigate, using **dsld**.

```
z <- dsldLinear(lsa,'lsat','race1')
summary(z)
```

```
$`Summary Coefficients`
     Covariate    Estimate StandardError       PValue
1  (Intercept) 31.98578856   0.448435264 0.000000e+00
2          age  0.02082458   0.005841758 3.641634e-04
3      decile1  0.12754812   0.020946536 1.134602e-09
4      decile3  0.21495015   0.020918737 0.000000e+00
5      fam_inc  0.30085804   0.035953051 0.000000e+00
6         ugpa -0.27817274   0.080430542 5.430993e-04
7   gendermale  0.51377385   0.060037102 0.000000e+00
8   race1black -4.74826307   0.198088318 0.000000e+00
9    race1hisp -2.00145969   0.203504412 0.000000e+00
10  race1other -0.86803104   0.262528590 9.449471e-04
11  race1white  1.24708760   0.154627086 6.661338e-16
```

```
12    cluster2 -5.10668358   0.119798362 0.000000e+00
13    cluster3 -2.43613709   0.074744210 0.000000e+00
14    cluster4  1.21094567   0.088478368 0.000000e+00
15    cluster5  3.79427535   0.124476695 0.000000e+00
16    cluster6 -5.53216090   0.210750853 0.000000e+00
17   fulltime2 -1.38882076   0.116212777 0.000000e+00
18     barTRUE  1.74973262   0.102818692 0.000000e+00

$`Sensitive Factor Level Comparisons`
   Factors Compared Estimates Standard Errors      P-Value
1     asian - black  4.748263       0.1980883 0.000000e+00
2      asian - hisp  2.001460       0.2035044 0.000000e+00
3     asian - other  0.868031       0.2625286 9.449471e-04
4     asian - white -1.247088       0.1546271 6.661338e-16
5      black - hisp -2.746803       0.1863750 0.000000e+00
6     black - other -3.880232       0.2515488 0.000000e+00
7     black - white -5.995351       0.1409991 0.000000e+00
8      hisp - other -1.133429       0.2562971 9.764506e-06
9      hisp - white -3.248547       0.1457509 0.000000e+00
10    other - white -2.115119       0.2194472 0.000000e+00
```

Note the retrospective view–using later events to "predict" the past. This is valid, but may seem odd at first.

There are very concerning racial differences here. Two very similar people—who attended the same quality law school, with the same undergraduate grades, the same law school grades, even having the same bar passage status—will have LSAT scores differing on average by almost 6 points if one person is Black and the other is white.

Again, one must be very cautious in drawing conclusions as to causes, not only because of the questionable quality of the dataset but also because *hidden confounders* may be at work here. For instance, though we have data on undergraduate GPA, we don't know the quality of the undergraduate institution. Equally important, we are looking only at those who were admitted to law school; there may be a different pattern in the general population. But the results here raise serious concerns.

2.5.2 Is the bar exam fair?

And what about passage of the bar exam? Does race play a role?

```
comparisonPts <- lsa[c(2,22,222,2222),-c(8,11)]
w <- dsldLogit(lsa,'bar','race1',comparisonPts,yesYVal= 'TRUE')
summary(w)
```

```
$`Summary Coefficients`
     Covariate    Estimate Standard.Error        PValue
1  (Intercept) -6.24270431    0.401137499  1.308749e-54
2          age  0.03422736    0.004412075  8.651327e-15
3      decile1  0.04757930    0.018427921  9.825421e-03
4      decile3  0.48948790    0.020478831 2.909741e-126
5      fam_inc -0.02199998    0.030373942  4.688788e-01
6         lsat  0.08539196    0.005869773  6.036082e-48
7         ugpa  0.35866792    0.068480789  1.627690e-07
8   gendermale  0.15784971    0.052981691  2.888835e-03
9   race1black  0.20964666    0.130530665  1.082497e-01
10   race1hisp  0.07078955    0.136700672  6.045675e-01
11  race1other  0.06268978    0.185086652  7.348320e-01
12  race1white  0.37434379    0.111291746  7.692575e-04
13    cluster2 -0.85200531    0.096842597  1.394793e-18
14    cluster3 -0.15391198    0.069754152  2.734957e-02
15    cluster4 -0.23351498    0.082896615  4.848324e-03
16    cluster5  0.25793171    0.142650370  7.058485e-02
17    cluster6 -1.33844995    0.145701743  4.068449e-20
18   fulltime2 -0.54114731    0.088416220  9.330973e-10

$`Sensitive Factor Level Comparisons`
   Factors Compared    Estimates Standard Errors      P-Value
1     asian - black -0.209646661      0.13053067 0.1082496509
2      asian - hisp -0.070789554      0.13670067 0.6045674668
3     asian - other -0.062689778      0.18508665 0.7348319612
4     asian - white -0.374343793      0.11129175 0.0007692575
5      black - hisp  0.138857107      0.11256856 0.2173906884
6     black - other  0.146956884      0.16926160 0.3852820509
7     black - white -0.164697131      0.09362748 0.0785806070
8      hisp - other  0.008099776      0.17545132 0.9631788810
```

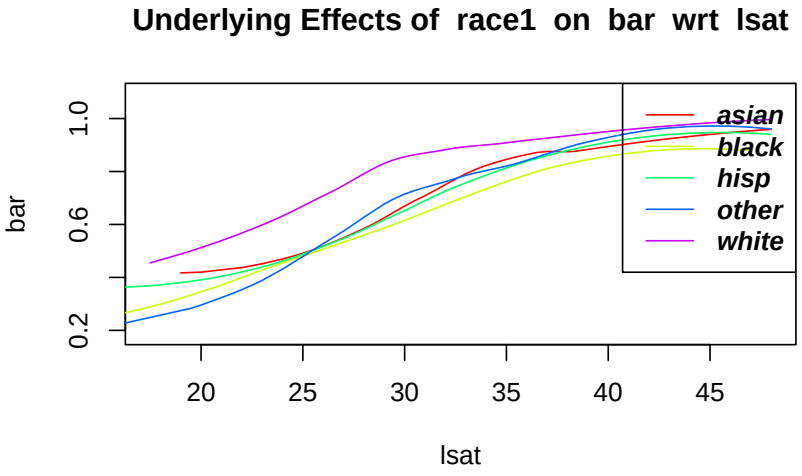
```
9      hisp - white -0.303554239       0.09885125 0.0021374826
10    other - white -0.311654015       0.15832839 0.0490349927
```

That's a lot of output. Let's take a closer look.

At least judging from the rows labled 'black - white', Black and white students having the same traits appear to have passed the bar exam at about the same rates.

We might also do a little check of the appropriateness of the logistic model for this data. One rough approach might be to use **dsldConditDisparity()**, as we did in the linear case:

```
# Y needs to be numeric, in this case 0,1
lsab <- lsa
lsab$bar <- ifelse(lsab$bar=='TRUE',1,0)
dsldConditDisparity(lsab,'bar','race1','lsat','lsab$age > 0')
```



Looks pretty good. But we can go further, using kNN analysis, as it makes no assumptions about the form of the regression function:

```
w <- dsldML(lsa,'bar','race1',qeMLftnName='qeKNN',
   opts=list(k=50,yesYVal='TRUE'))
print(w)
```

```
$testAccs
    asian     black      hisp     other     white
0.2151899 0.3583333 0.2473118 0.1081081 0.0930000

$comparisons
      age decile1 decile3 fam_inc lsat ugpa gender cluster fulltime asian black
16719  56       9       9       3   32  3.5   male       4        1  1.00  0.90
25464  61       6       5       2   29  3.1 female       2        1  0.68  0.46
16441  62       8       7       4   36  2.8   male       4        1  0.92  0.84
24103  57       9       9       4   33  3.3   male       1        1  0.98  0.90
22083  57       7       6       5   39  3.4   male       3        1  0.92  0.82
      hisp other white
16719 0.98  1.00  1.00
25464 0.78  0.94  0.78
16441 0.94  0.84  1.00
24103 0.96  0.96  1.00
22083 0.86  0.90  1.00
```

The pattern here seems to be a bit uneven. Black and white test takers seem to be on par with each other in three cases, but with major differences in the other two rows. We should look at more cases, and revisit the logit analysis, possible checking for quadratic trends..

2.6 Case study: problems with significance testing

In 2016, the American Statistical Association released its first-ever position paper, to warn of the problems of significance testing and "p-values." Though the issues had been well known for years—some journals had even banned the use of p-values in their published research papers—it was "significant" that the ASA finally took a stand. Let's use the law school example in Section 2.5 to illustrate.

There is concern that the LSAT and other similar tests may be heavily influenced by family income, thus unfair, especially to underrepresented minorities. To investigate this, let's consider

Review: To test the hypothesis $H_0 : \beta_i = 0$, one computes twice the area to the right of $|b_i|$ in the standard normal distribution. That number is the *significance* level, with values under 0.05 being termed *significant*; 0.01 is the criterion for *highly significant*, and so on.

the b_i, the estimated coefficients in our linear model for the LSAT above.

In particular, look at the coefficient for family income, 0.3009. The p-value is essentially 0, which in an academic research journal would classically be heralded with much fanfare, termed "very highly significant," with a 3-star insignia. Indeed, the latter is seen in the output above. (This comes from R, not **dlsd**.) But actually, the impact of family income is not very large. Here's why:

Family income in this dataset is measured by quintiles. So this estimated coefficient says that, for example, if we compare people who grew up in the bottom 20% of income with those who were raised in the next 20%, the mean LSAT score rises by only about 1/3 of 1 point—a minuscule difference on a test where scores are typically in the 20s, 30s and 40s. The 95% confidence interval (CI), (0.2304,0.3714), again indicates that the effect size here is very small.

Mathematically, testing for a 0 effect is equivalent to checking whether the CI contains 0. But this is missing the point of the CI, which is to (a) give us an idea of the effect *size*, and (b) to indicate how accurate our estimate is of that size. Aspect (a) is given by the location of the center of the interval, while (b) is seen from the CI's width

So family income is not an important factor after all, and the significance test was highly misleading.

Some who read this may object, "Sure, there sometimes may be a difference between statistical significance and practical significance. But I just want to check whether my model fits the data." Actually, it's the same problem.

For instance, suppose we are considering adding an interaction term between race and undergraduate GPA to our above model. For simplicity, let's use R's linear model function, **lm()**, directly:

```
w <- lm(lsat ~ .,lsa)  # predict lsat from all other variables
w1 <- lm(lsat ~ .+race1:ugpa,lsa)  # add interaction
summary(w1)
```

```
Call:
lm(formula = lsat ~ . + race1:ugpa, data = lsa)

Residuals:
```

```
     Min       1Q   Median       3Q      Max
-19.1783  -2.8065   0.1219   2.8879  16.0633

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     26.574993   1.219611  21.790  < 2e-16 ***
age              0.020612   0.005837   3.531 0.000415 ***
decile1          0.127585   0.020926   6.097 1.10e-09 ***
decile3          0.213918   0.020902  10.234  < 2e-16 ***
fam_inc          0.295042   0.035939   8.210 2.35e-16 ***
ugpa             1.417659   0.363389   3.901 9.60e-05 ***
gendermale       0.513686   0.059986   8.563  < 2e-16 ***
race1black       4.121631   1.439354   2.864 0.004194 **
race1hisp        1.378504   1.570833   0.878 0.380191
race1other       2.212299   1.976702   1.119 0.263073
race1white       6.838251   1.201559   5.691 1.28e-08 ***
cluster2        -5.105703   0.119879 -42.590  < 2e-16 ***
cluster3        -2.427800   0.074862 -32.430  < 2e-16 ***
cluster4         1.208794   0.088453  13.666  < 2e-16 ***
cluster5         3.777611   0.124422  30.361  < 2e-16 ***
cluster6        -5.565130   0.210945 -26.382  < 2e-16 ***
fulltime2       -1.406151   0.116132 -12.108  < 2e-16 ***
barTRUE          1.743800   0.102855  16.954  < 2e-16 ***
ugpa:race1black -2.876555   0.460281  -6.250 4.20e-10 ***
ugpa:race1hisp  -1.022786   0.494210  -2.070 0.038508 *
ugpa:race1other -0.941852   0.617940  -1.524 0.127479
ugpa:race1white -1.737553   0.370283  -4.693 2.72e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.193 on 20778 degrees of freedom
Multiple R-squared:  0.3948,    Adjusted R-squared:  0.3942
F-statistic: 645.4 on 21 and 20778 DF,  p-value: < 2.2e-16
```

```r
typx <- lsa[1,-5]  # set up an example case
predict(w,typx)  # estimated regression function value
```

```
      2
40.2294
```

```
predict(w1,typx)  # estimated regression function value
```

```
       2
40.2056
```

Indeed, the Black and white interaction terms are "very highly significant." But that does mean we should use the more complex model?

Recall that many ML algorithms, including standard linear regression, use the estimated regression function value as its predicted value. So we see here that adding in term interaction term changed the estimated value of the regession function by only about 0.02 out of a 40.23 baseline. So, we may well prefer the simpler, no-interaction model.

Again, we must not take small p-values literally.

The basic problem with significance testing

The central issue in the above examples, and essentially in any other testing situation, is that *the test is not answering the question of interest to us.*
We wish to know whether family income plays a substantial role in the LSAT, not whether they is any relation at all, no matter how meaningless. Similarly, we wish to know whether the interaction between race and GPA is substantial enough to include it in our model, not whether there is any interaction at all, no matter how tiny.
The question at hand in research studies is rarely, if ever, whether a quantity is *exactly* 0, i.e. 0.000... to infinitely many decimal places. Indeed, in most cases our measuring instrument is not this accurate in the first place, and there will always be systemic bias or missingness, unobserved variables and so on, so exact zeroness is not even a meaningful concept.
Thus in almost all cases, significance tests don't address the issue of interest, which is whether some population quantity is substantial enough to be considered important. Analysts should not be misled by words like "significant."

> As noted, modern statistical practice places reduced value, or in the view of many, no value at all, on significance testing.

To be sure, not all statistical professionals are "modern." Some on the ASA committee that produced the position paper were more reserved on the matter, resulting in the paper not going further than some would have preferred. Nor is modernness necessarily a virtue.

But readers of this book are urged to always keep in mind the two examples above—family income and an interaction term each having minuscule impact even though the tests declared them "significant"—in conducting future analyses, or assessing the analysis of others.

Instead, form a confidence interval for the quantity of interest. Do *not* just note whether the CI contains 0. Where is the CI's center? How wide is it, relative to the accuracy you wish to have? Then make your decision regarding importance of the effect on the basis of the overall situation, *not* by mechanically performing a test.

2.7 Example: COMPAS dataset

Recall that COMPAS is a tool developed to help judges decide on sentences in criminal trials, which it does by predicting whether the defendant is likely to recidivate. As noted earlier, one study found the tool to be biased against African-Americans; the creator of the tool, Northpointe, disagrees.

Let's investigate this, by predicting Y = the COMPAS score, with S = race and C = the remaining variables. Does S have much effect on Y?

```
data(compas1)
w <- dsldLinear(compas1,'decile_score','race')
summary(w)
```

```
$`Summary Coefficients`
             Covariate     Estimate StandardError       PValue
1          (Intercept)  12.46413291   1.452859168 0.000000e+00
2                  age  -0.09404721   0.002497211 0.000000e+00
3        juv_fel_count   0.39479964   0.073934975 9.303767e-08
4       juv_misd_count   0.16212898   0.061011256 7.875488e-03
5      juv_other_count   0.29435177   0.060546864 1.164686e-06
6         priors_count   0.22714719   0.006509637 0.000000e+00
7              sexMale  -0.13540331   0.069213068 5.042679e-02
8    two_year_recidYes   0.82273099   0.062559111 0.000000e+00
9            raceAsian  -0.68595234   0.393500071 8.129735e-02
10       raceCaucasian  -0.46795096   0.062339397 6.061818e-14
11        raceHispanic  -0.93831638   0.101286306 0.000000e+00
12 raceNative American   0.38640654   0.554347886 4.857734e-01
13           raceOther  -1.30753368   0.123716368 0.000000e+00
14           c_jail_in -69.51719068   9.027235193 1.354472e-14
15          c_jail_out  70.72627748   7.301645508 0.000000e+00
16      c_offense_date  -4.21691395   1.326098890 1.473059e-03
17      screening_date  -7.75559525   5.422330875 1.526291e-01
18          in_custody  -6.31642337   2.419684027 9.042765e-03
19         out_custody   9.97020948   1.951867970 3.255267e-07

$`Sensitive Factor Level Comparisons`
                     Factors Compared  Estimates Standard Errors      P-Value
1            African-American - Asian  0.6859523       0.3935001 8.129735e-02
2         African-American - Caucasian  0.4679510       0.0623394 6.061818e-14
3          African-American - Hispanic  0.9383164       0.1012863 0.000000e+00
4  African-American - Native American -0.3864065       0.5543479 4.857734e-01
5            African-American - Other  1.3075337       0.1237164 0.000000e+00
6                   Asian - Caucasian -0.2180014       0.3935940 5.796653e-01
7                    Asian - Hispanic  0.2523640       0.4015448 5.296877e-01
8             Asian - Native American -1.0723589       0.6779716 1.137143e-01
9                       Asian - Other  0.6215813       0.4077123 1.273692e-01
10               Caucasian - Hispanic  0.4703654       0.1033446 5.328378e-06
11        Caucasian - Native American -0.8543575       0.5555712 1.240975e-01
12                  Caucasian - Other  0.8395827       0.1253271 2.096612e-11
13         Hispanic - Native American -1.3247229       0.5612839 1.826678e-02
14                   Hispanic - Other  0.3692173       0.1485386 1.293095e-02
15            Native American - Other  1.6939402       0.5659026 2.759400e-03
```

Y, the decile score, is the risk of recidivism; the larger the value

of Y, the greater the risk.

A 95% confidence interval for the mean difference between Black and Caucasian risk scores, holding age, juvenile felony count and so on constant, is 0.4680 $\pm$ 1.96 x 0.0623 = (0.3459,0.5902). Since the risk of recidivism is given in deciles, the value is 1,2,...,10, we see that the tool does indeed appear to be biased against Black defendants. Though the effect is rather small, around 0.6 of a decile point or less,
and as usual, one must keep in mind the possibility of unseen confounders, it is still a matter of serious concern. Note that the estimated effect of being Blac, 0.4680, is actually larger than the estimated effect of having one additional juvenile felony count, 0.3948.

2.8 Deciding on a set of confounders C

One may have specific confounders in mind for a particular analysis, but it is often unclear as to which to use, or for that matter, why not use them all? This section addresses those issues.

We will address that second question first: Why not use all our variables, other than Y and S, as confounders? We will see that if we have a large number of variables, there is good reason to select only a subset of potential confounders.

How might we do so? We will first recommend some graphical and tabular methods aimed at preliminary exploration toward this end, and then present a **dsld** function that users may find useful for more systematic selection of confounders.

2.8.1 The problem

The German credit dataset, included in the **fairml** package and thus with **dsld**, consists of a total of 21 variables:

```
data(german.credit)
names(german.credit)
```

```
 [1] "Account_status"           "Duration"
 [3] "Credit_history"           "Purpose"
 [5] "Credit_amount"            "Savings_bonds"
 [7] "Present_employment_since" "Installment_rate"
 [9] "Other_debtors_guarantors" "Resident_since"
[11] "Property"                 "Age"
[13] "Other_installment_plans"  "Housing"
[15] "Existing_credits"         "Job"
[17] "People_maintenance_for"   "Telephone"
[19] "Foreign_worker"           "Credit_risk"
[21] "Gender"
```

Excluding Y = credit risk and S = gender, that gives us 19 variables to choose among for our confounders. Which ones should we use?

Technically, almost any variable is a confounder. The impact may quite minuscule, but through a long chain of relations among many variables, there will usually be at least some connection, though again possibly very faint.

Well, then, why not use them all? That is what we've done in our earlier examples. But there are several issues to consider not using the full set of variable,s i.e. every variable other than Y and S:

- It may result in overfitting, resulting in large standard errors.
- It is unwieldy, difficult to interpret. Many treatments of these issues speak of a desire for a "parsimonious" model.
- There is a concern regarding duplication. It may be that, say, some pair of confounders suffices, and adding further confounders does nothing to further clarify discrimination effects.

Concerning this last point: We do not merely have 19 choices for confounders; we must choose from the set of all possible groups of confounders: singletons, pairs, triplets and so on. There are 2^{19} possibilities here, about half a million.

Our specific purpose here is to find a reasonable set of confounders. In essence, that means find variables C such that C

is substantially correlated with both Y and S. One way to approach that is to do separate predictions of Y and S, and see which features turn out to predict both well.

2.8.2 Exploratory example: engineering wages

One useful tool here is the function **dsldFrequencyByS()**, which aims to analyze categorical (not numeric) columns by printing out a list of frequencies. Since education level in the **svcensus** data is categorical, we can call the function as follows:

```
library(dsld)
data(svcensus)
dsldFrequencyByS(svcensus, "educ", "gender")
```

```
       Frequency of zzzOther Frequency of 14 Frequency of 16
female             0.2068052      0.02098615       0.7722086
male               0.2177579      0.04110130       0.7411408
```

The similar frequency values between men and women suggests not using education as a confounder.

On the other hand, we *do* see more interesting results if we look at *occupation* instead of *education*:

```
library(dsld)
data(svcensus)
dsldFrequencyByS(svcensus, "occ", "gender")
```

```
       Frequency of 102 Frequency of 101 Frequency of 100 Frequency of 141
female        0.3117359        0.2349226        0.3274246       0.04258354
male          0.2016862        0.2203267        0.3433671       0.01923330
       Frequency of 140 Frequency of 106
female       0.02587612       0.05745721
male         0.04446055       0.17092610
```

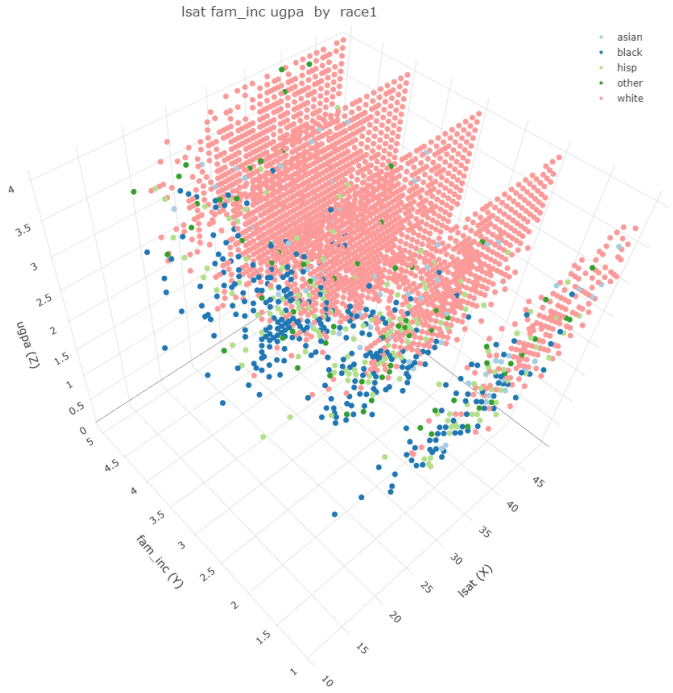
Notice for instance the difference between the proportion of males in occupation 106 versus females in that same occupation–a difference of approximately 11%. The difference in frequencies here is much greater than with the 'educ' example, which suggests using occupation as a confounder. Since different occupations correlate with different wages, it is possible that this gender difference in occupations proportions could be affecting the perceived relationship between gender and wage.

One might consider conducting formal statistical inference in this comparison, calculating a p-value from a *chi-square test*. But as we saw in Section 2.6. testing for model fit suffers from the same problems as testing for nonzero effects. At any rate, here we are merely doing an exploratory analysis anyway.

2.8.3 Exploratory example: Law school admissions data

Suppose we are investigating the relationships among the variables LSAT score, GPA and race. One way to visualize these relationships would be through **dsldScatterPlot3D**:

```
data(law.school.admissions)
dsldScatterPlot3D(law.school.admissions,
   c('lsat','fam_inc','ugpa'), 'race1', pointSize = 4)
```



This is a **plotly** interactive graph. To fully understand this function, the reader should execute the function outside of Quarto, i.e. run the above code directly from R. Try try features such as move, rotate, annotation display and so on.

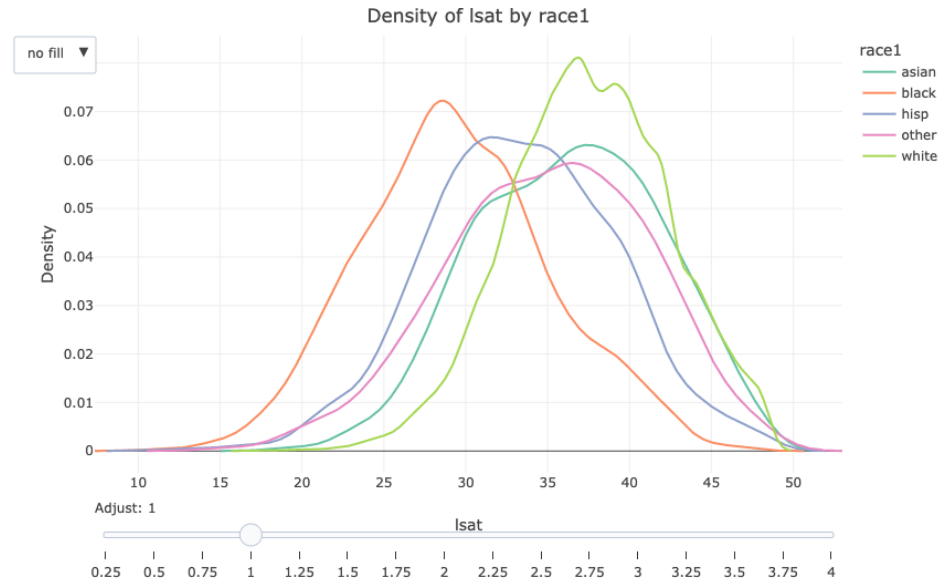
Of course, visualizing a 3-dimensional scatter plog is more challenging than viewing the ordinary 2-dimensional kind. But it does enable richer search for relationships. Here are a few trends suggested by the plot:

- On the **fam_inc** axis, the lowest quintile of family income is mostly populated by Black and Latino students, while the upper two levels are almost entirely made up of white students.
- On the **lsat** axis, most of those with a lower score happen to be non-white, across all income levels.
- The **ugpa** axis has a similar trend to that of the **lsat** axis, albeit to a much weaker extent.

Once again, this analysis is merely exploratory, but the graph lends some credence to claims that family income may confound the relationship between race and LSAT score. Note however, that these graphs do not do much to answer the question of whether, in this case, that relationship is substantial. The formal analysis we did earlier indicates that it is not.

Returning to the question of whether race have a substantial impact on exam results, we can look at the density plot of LSAT scores against different races.

```
# requires 'webshot' package
data(law.school.admissions)
dsldDensityByS(law.school.admissions, cName = "lsat", sName = "race1")
```



Each curve represents the distribution of LSAT scores for one race. The graph suggests the possibility of racial bias in the LSAT. But again, this bias may be confounded.

This suggests that family income could be confounding the potential relationship between race and LSAT scores. But, as noted, our earlier analysis indicated that the degree of this relationship is very small.

If we want to investigate other potential confounders, we can call **dsldConfounders()** with 'race1' as our sensitive variable.

i Almost anything is a confounder

Once one accounts for links of two variables, then links of links and so on, virtually everything in practice is a confounder to *some* degree, so it is up to the analyst to decide when a feature is confounding enough to be considered for inclusion in a model.

2.8.4 Variable selection methodology

The above graphical and tabular approaches may suffice to determine one's confounders in some applications. We now present more formal, systematic methods.

Many, many approaches to this *variable selection* or *feature selection* problem have been proposed over the years. The **qeML** package includes five of them (see the documentation), and there are myriad others.

A nice overview of such methodology, and implementations in R and Python, are given in Parr *et al*.

One widely-used method is *permutation*, which we will use here. To gauge the importance of a variable V in the context of using some ML algorithm, we do two runs through the data, first with the original dataset, and then with the V column changed through a random shuffling of the values in that column. In that second run, our predictive accuracy should be compromised, and the importance measure is taken to be the proportional increase in error rate. The larger the increase, the more important we deem the variable.

2.8.5 The dsldCHunting() function

This function serves as an aid to selecting confounders, using the permutation-based computation in the **randomForests** package.

Its operation is best described by example, which we will present next. Before reading it, though, recall that we are interested in finding variables that are correlated with, i.e. predict well, both Y and S.

2.8.6 Example: Boston mortgage data

Here is an example, using the mortgage data.

```
data(mortgageSE)
dsldCHunting(mortgageSE,'deny','black')
```

```
$impForY
       denpmi        p_irat      loan_val       hse_inc       ltv_med
 0.0147809927  0.0145627277  0.0115778621  0.0113391532  0.0078584746
       single         mcred         condo      ltv_high         ccred
 0.0018733865  0.0018533472  0.0012931002  0.0010681350  0.0010592614
     probunmp        pubrec       selfemp        hischl
 0.0005752676  0.0004823619  0.0004626509 -0.0001048125

$impForS
     loan_val        p_irat       ltv_med       hse_inc         condo
 0.0129525280  0.0094211116  0.0090298573  0.0061711267  0.0044294133
        ccred        single        pubrec      ltv_high        denpmi
 0.0026117584  0.0023983252  0.0016617747  0.0008335726  0.0005516326
       hischl       selfemp      probunmp         mcred
 0.0001515775  0.0001217180 -0.0004341749 -0.0008345865

$inCommon
$inCommon[[1]]
character(0)

$inCommon[[2]]
[1] "p_irat"

$inCommon[[3]]
[1] "p_irat"   "loan_val"

$inCommon[[4]]
[1] "p_irat"   "loan_val" "hse_inc"

$inCommon[[5]]
[1] "p_irat"   "loan_val" "hse_inc"  "ltv_med"

$inCommon[[6]]
[1] "p_irat"   "loan_val" "hse_inc"  "ltv_med"

$inCommon[[7]]
[1] "p_irat"   "loan_val" "hse_inc"  "ltv_med"  "single"

$inCommon[[8]]
[1] "p_irat"   "loan_val" "hse_inc"  "ltv_med"  "single"   "condo"
```

```
$inCommon[[9]]
[1] "p_irat"   "loan_val" "hse_inc"  "ltv_med"  "single"   "condo"    "ltv_high"

$inCommon[[10]]
[1] "denpmi"   "p_irat"   "loan_val" "hse_inc"  "ltv_med"  "single"   "condo"
[8] "ltv_high" "ccred"
```

The importance measures are printed out (the ranking is what matters), followed by the "top-i" sets in common.

For instance, the variables (excluding S) that predict Y well are first **p_irat**, then **denpmi** and so on. In predicting S, excluding Y, the best are **p_irat**, then **loan_val** et cetera.

Since we are focusing on variables that are correlated with both Y and S, we take the intersection. For example, the intersection of the top three Y predictors and top three S predicts is **p_irat** and **loan_val**.

As usual, there are no magic formulas to use here. The analyst is given a choice. One might use just the top two confounders, or the top three and so on.

2.8.7 Example: Employer bias test

In the paper "Are Emily and Greg More Employable Than Lakisha and Jamal? A Field Experiment on Labor Market Discrimination" (*Am. Econ. Rev.*, Sept. 2004), researcher "test" employers by sending CVs with "white-sounding" and "Black-sounding" names, checking for bias. Did employers call "whites" more often than "Blacks" for an interview? The resulting data is **lak** in **dsld**.

```
data(lak)
dim(lak)
```

```
[1] 447  63
```

```
names(lak)
```

```
 [1] "education"          "ofjobs"             "yearsexp"
 [4] "honors"             "volunteer"          "military"
 [7] "empholes"           "occupspecific"      "occupbroad"
[10] "workinschool"       "email"              "computerskills"
[13] "specialskills"      "firstname"          "sex"
[16] "race"               "h"                  "l"
[19] "call"               "city"               "kind"
[22] "adid"               "fracblack"          "fracwhite"
[25] "lmedhhinc"          "fracdropout"        "fraccolp"
[28] "linc"               "col"                "expminreq"
[31] "schoolreq"          "eoe"                "parent_sales"
[34] "parent_emp"         "branch_sales"       "branch_emp"
[37] "fed"                "fracblack_empzip"   "fracwhite_empzip"
[40] "lmedhhinc_empzip"   "fracdropout_empzip" "fraccolp_empzip"
[43] "linc_empzip"        "manager"            "supervisor"
[46] "secretary"          "offsupport"         "salesrep"
[49] "retailsales"        "req"                "expreq"
[52] "comreq"             "educreq"            "compreq"
[55] "orgreq"             "manuf"              "transcom"
[58] "bankreal"           "trade"              "busservice"
[61] "othservice"         "missind"            "ownership"
```

We'll take Y to be **call**, which records whether the "applicant" received a call back in response to submitting the CV. S will be race, which now leaves us with 63 - 2 = 61 potential confounders!

Actually, it's even worse. Many rows of the dataset contain missing values (coded NA in R). The version here uses only intact rows, of which there are 447. (The original data had 4870 rows.) A rough rule of thumb commonly cited in statistics is that in regression estimation, the number of features should be less than the square root of the number of data points; by this or almost any other measure, 61 predictors is well beyond the capacity of 447 data points. And, as noted, that number of variables is unwieldy. Let's see what we can do to reduce that.

```
dsldCHunting(lak,'call','race',25)
```

```
$impForY
```

```
   fraccolp_empzip fracdropout_empzip       linc_empzip   fracblack_empzip
      4.772650e-03       4.627861e-03      4.349551e-03      3.744587e-03
              linc         branch_emp   lmedhhinc_empzip        parent_emp
      2.847057e-03       2.422533e-03      2.301625e-03      2.096364e-03
      branch_sales          lmedhhinc      parent_sales           manager
      1.682490e-03       1.662307e-03      1.365164e-03      1.147235e-03
          fraccolp   fracwhite_empzip              city           compreq
      1.062280e-03       8.188254e-04      7.840211e-04      7.511824e-04
       fracdropout           yearsexp              adid        expminreq
      7.088380e-04       6.996802e-04      6.727436e-04      6.175214e-04
               sex          education         secretary           educreq
      4.642575e-04       4.314903e-04      3.976907e-04      3.701735e-04
        occupbroad                req             trade        othservice
      3.412396e-04       3.239205e-04      2.631443e-04      2.504701e-04
                 h                col     occupspecific        schoolreq
      2.215522e-04       1.723987e-04      1.457608e-04      1.444915e-04
               fed             orgreq      workinschool               eoe
      1.143394e-04       7.610630e-05      7.230243e-05      6.991121e-05
    computerskills      specialskills             comreq         ownership
      4.764673e-05       3.129636e-05      3.076923e-05      2.289900e-05
          salesrep           transcom           missind              kind
      1.328484e-06       0.000000e+00      0.000000e+00     -2.825163e-06
               email             ofjobs          bankreal         volunteer
     -6.121203e-06      -6.360268e-06     -1.725572e-05     -2.572591e-05
          empholes            honors            expreq        supervisor
     -3.131322e-05      -3.380254e-05     -7.057885e-05     -7.493107e-05
                 l       retailsales        offsupport        busservice
     -7.664920e-05      -8.825736e-05     -1.131883e-04     -1.433669e-04
              manuf          fracblack          military          fracwhite
     -1.932477e-04      -2.030080e-04     -2.143711e-04     -2.353473e-04
          firstname
     -5.472534e-04

$impForS
          firstname          lmedhhinc               linc          fracwhite
      6.686840e-02       3.792683e-03      2.184668e-03       1.937198e-03
           military               city           manager               sex
      1.170462e-03       9.194788e-04      6.147443e-04       5.498834e-04
      workinschool          ownership           educreq           fraccolp
      4.687138e-04       9.049113e-05      8.116598e-05       7.425017e-05
           missind              trade             email           transcom
```

```
  0.000000e+00   -1.202213e-05   -1.494294e-05   -3.162388e-05
      empholes     fracdropout               l        compreq
 -1.064715e-04   -1.220808e-04   -1.365759e-04   -1.391709e-04
      bankreal      supervisor       volunteer         comreq
 -1.918775e-04   -1.949936e-04   -1.995482e-04   -2.254765e-04
     offsupport     occupbroad             col          manuf
 -2.358398e-04   -2.809561e-04   -3.519604e-04   -3.533836e-04
             eoe        salesrep       secretary      othservice
 -3.593646e-04   -3.622374e-04   -4.093889e-04   -4.351911e-04
             req        schoolreq         expreq             h
 -4.837630e-04   -4.902926e-04   -5.306430e-04   -5.625066e-04
computerskills        education         orgreq          honors
 -5.889709e-04   -6.004271e-04   -6.325028e-04   -6.408441e-04
             fed    specialskills     retailsales       fracblack
 -6.540626e-04   -6.617703e-04   -7.498588e-04   -8.162425e-04
      parent_sales    occupspecific            kind  fraccolp_empzip
 -8.293911e-04   -9.536404e-04   -1.061916e-03   -1.245931e-03
            ofjobs fracdropout_empzip      busservice lmedhhinc_empzip
 -1.247235e-03   -1.280764e-03   -1.294118e-03   -1.483864e-03
     parent_emp fracwhite_empzip        yearsexp      linc_empzip
 -1.682487e-03   -2.194421e-03   -2.316927e-03   -2.384270e-03
      expminreq            adid fracblack_empzip      branch_emp
 -2.548126e-03   -3.453402e-03   -3.700286e-03   -3.934042e-03
      branch_sales
 -4.905349e-03

$inCommon
$inCommon[[1]]
character(0)

$inCommon[[2]]
character(0)

$inCommon[[3]]
character(0)

$inCommon[[4]]
character(0)

$inCommon[[5]]
[1] "linc"
```

```
$inCommon[[6]]
[1] "linc"

$inCommon[[7]]
[1] "linc"

$inCommon[[8]]
[1] "linc"

$inCommon[[9]]
[1] "linc"

$inCommon[[10]]
[1] "linc"      "lmedhhinc"

$inCommon[[11]]
[1] "linc"      "lmedhhinc"

$inCommon[[12]]
[1] "linc"      "lmedhhinc" "manager"

$inCommon[[13]]
[1] "linc"      "lmedhhinc" "manager"   "fraccolp"

$inCommon[[14]]
[1] "linc"      "lmedhhinc" "manager"   "fraccolp"

$inCommon[[15]]
[1] "linc"      "lmedhhinc" "manager"   "fraccolp"  "city"

$inCommon[[16]]
[1] "linc"      "lmedhhinc" "manager"   "fraccolp"  "city"

$inCommon[[17]]
[1] "linc"      "lmedhhinc" "manager"   "fraccolp"  "city"

$inCommon[[18]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "fracdropout"
```

```
$inCommon[[19]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "fracdropout"

$inCommon[[20]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout"

$inCommon[[21]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout" "sex"

$inCommon[[22]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout" "sex"

$inCommon[[23]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout" "sex"

$inCommon[[24]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout" "sex"         "educreq"

$inCommon[[25]]
[1] "linc"        "lmedhhinc"   "manager"     "fraccolp"    "city"
[6] "compreq"     "fracdropout" "sex"         "educreq"
```

We are seeking variables that are correlated both with Y and S, yet there appear to be many that correlate with just one of these. That's to be expected for datasets having a large number pf variables. But still, in the end we see a few that meet our goals.

2.9 Observationational Studies

An *observational study* is, in essence, one that is not planned. For instance, say we are comparing an old and a new drug for

hypertension. Suppose that, unknown to us, the new medication does well on younger patients but not on older ones. Say the nature of the data collection process results in disproportionately sampling older patients, but our data itself does not include patient ages. This could unfairly make the new drug appear ineffective. So age would be a confounder, but alas, an unobsered one.

In a *randomized clinical trial* (RCT), we would randomly assign treatments to patients, so there would likely be no strong imbalance in age distribution in the two drug groups. So, even if our data still did not record patient age, there would be no age bias in our analysis. But in most cases in practice, RCTs are infeasible or impossible; we cannot "assign" race, for instance.

The analyst must always keep this issue in mind.

2.10 On Causal Analysis (advanced material)

Causal analysis (CA) takes on many forms, and indeed, there is no good definition of what it entails. Moreover, it tends to be controversial. our discussion here will focus on two topics:

- covariate matching methods.
- *directed acyclic graphs* (DAGs)

2.10.1 Covariate matching

The basic idea here is quite simple. Consider our above example comparing two medications, A and B, with Y being the response variable. Instead of comparing the two sample means of Y, we form pairs of data points, with an A group and B group member in each pair. These pairs form our new data. (Not every one of the original data points will now be represented, though some may appear more than once.) We then compare mean Y values among the two groups in this new data.

Here is an example, involving a jobs training program::

```
data(lalonde,package='Matching')
ll <- lalonde
ll$treat <- as.factor(ll$treat)
ll$re74 <- NULL
ll$re75 <- NULL
summary(dsldMatchedATE(ll,'re78','treat','1'))
```

The data structure is

```
  age educ black hisp married nodegr     re78 u74 u75 treat
1  37   11     1    0       1      1  9930.05   1   1     1
2  22    9     0    1       0      1  3595.89   1   1     1
3  30   12     1    0       0      0 24909.50   1   1     1
4  27   11     1    0       0      1  7506.15   1   1     1
5  33    8     1    0       0      1   289.79   1   1     1
6  22    9     1    0       0      1  4056.49   1   1     1
...
```

Participants in the program had **treat** coded 1, the others 0. Income several years later (1978) is shown in **re78**. In the matched pairs, the average difference in income was estimated to be 1374.7.

A key point is the manner in which the matching is done. In most cases, it can be done only approximately. In the medication example, for instance„we probably could not find two patients of the same age *and* the same gender *and* the same race and so on. But the more variables we have, the harder it is to find even approximate matches. One solution is to use *propensity scores*, not covered here.

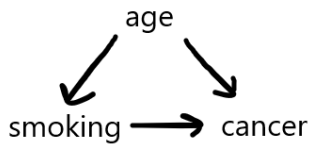
An even larger problem is interpretation. Let X denote the vector of our predictors, e.g. age, gender and so on. The regression methods covered earlier in this book estimate the mean Y for a given X, whereas matching methods estimate the unconditional mean of Y. The interpretation of the latter may be problematic, since our matching process itself may have changed the distribution of X.

We then compute mean Y

2.10.2 DAGS

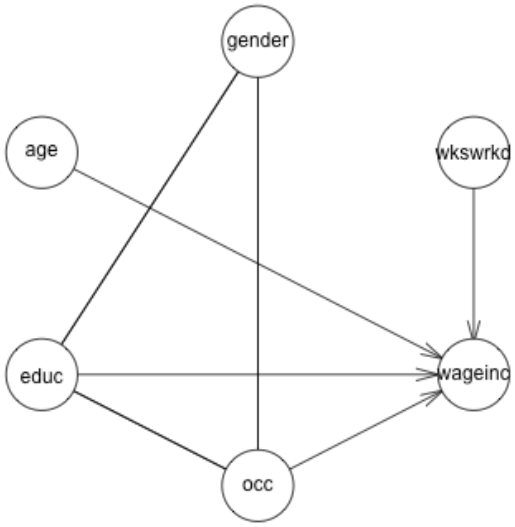
In a DAG, an arrow from A to B signifies that A "causes" B.

For instance, a graphical description of a situation in which older people (say who came of age before anti-smoking campaigns) might be more likely to smoke, and for other reasons may be more likely to develop cancer even if they are nonsmokers. A graphical description would be



Here is an example the **bnlearn** package on our **svcensus** dataset:

```
data(svcensus)
# iamb does not accept integer data
svcensus$wkswrkd <- as.numeric(svcensus$wkswrkd)
svcensus$wageinc <- as.numeric(svcensus$wageinc)
iambOut <- dsldIamb(svcensus)
plot(iambOut)
```



There are a couple of undirected arcs, e.g. between gender and occupation–found to be relations, but with indeterminate direction, using available data.

More interesting, though, is the lack of a causal arc from gender to wage income. This is in stark contrast to what **dsldLinear** suggests:

Though again we omit a detailed treatment of CA DAGs here, we note that that field defines a confounder of two variables as a variable that is an "ancestor" to both, education being a confounder for income and occupation.

```
library(dsld)
data(svcensus)
w <- dsldLinear(svcensus,'wageinc','gender',interactions=FALSE)
coef(w)
```

```
$gender
 (Intercept)          age       educ16 educzzzOther       occ101       occ102
 -21264.9056     469.8373    7597.2521  -14167.3888    1838.5510   13688.3012
      occ106       occ140       occ141      wkswrkd   gendermale
   1380.0254   11732.5837   10791.0158    1323.1999    8595.5831
```

Here, even correcting for age, education and so on, we still find a substantial gender pay gap. Gender does seem to matter, on its own.

It should be noted that if the **bnlearn** function **hc** is used instead of **iamb** (not shown), there is indeed a link from 'gender' to 'wageinc', but apparently at the cost of other anomalies. For example, there is no arc, or path of arcs, from 'wkswrkd' to 'wageinc'.

CA DAGs are of course quite visually appealing, and many people find causal concepts helpful in their thoughts processes regarding the data. But as pointed out by the developers of such methodology, the graphs can be very misleading. Using DAGs effectively requires deep understanding of the concepts. A treatment of these concepts is well beyond the scope of this course, but we urge users and consumers of CA to keep the following points in mind:

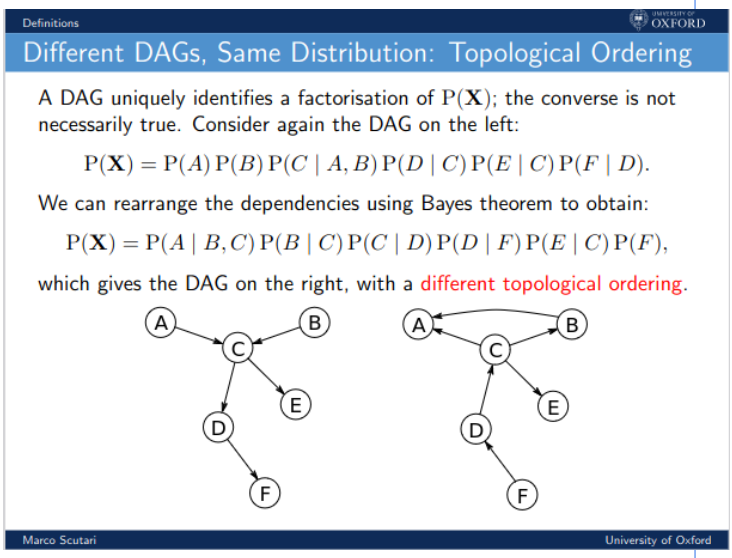
- The definition of *cause* is of course central to proper use of CA DAGs. It does *not* mean what one ordinarily thinks of as causal ("The sidewalk was icy, so the man slipped and fell"), and it can become quite complex; "wrapping one's head around" the definition can be a challenge even for experienced probabilistic modelers.
- Unseen variables, such as age in the above example, can be just as harmful in CI contexts as in observational data.
- Causal analysis is just as much at the mercy of sample size as is classical statistical analysis, indeed even more so, since the causal version in effect estimates many more parameters. Absence of a link may be due to insufficient data, and nonlinks at the population level may appear as "links" in our graph by random chance in small datasets.
- Most important, DAGs have a highly concerning uniqueness problem:

> **DAGS are not unique**
>
> - Most important, for any given dataset, there is typically no unique DAG: DAG. Typically many DAGs

CMU professor C. Shalizi has bluntly noted that in cases of nonuniqueness, presenting just one DAG without mentioning the equivalent ones, "...[which] is often what people seem to do," might be viewed as "lying by omission." Recognizing the nonuniqueness is "simply a matter of scientific honesty."

can fit the dataset equally well, and they can be in conflict with each other. One DAG might say that A causes B, while another, equally credible in terms of data fit, might have B causing A, while a third says A and B are independent.

M. Scutari, author of **bnlearn**, shows an example in his Cambridge lectures:



The nonuniqueness is of course very concerning. In recent years, there has been much handwringing in science about the Replication Crisis, in which scientists try to replicate the findings of an earlier published paper, and discover serious problems with the earlier results. Imagine the replication problems that can arise if authors of a paper present just one of many equally-credible DAGs, as is usually the case.

DAGs can indeed lead to valuable insight, but can be misleading as well. A healthy skepticism and a solid understanding of the concepts are mandatory for avoiding misleading or invalid results.

3 Part II: Discovering/Mitigating Bias in Machine Learning

In modern applications of machine learning as a predictive modeling tool, it would be irresponsible to produce a model that biases one sensitive group over another. As such, it becomes imperative to find methods of uncovering and reducing any such biases.

3.1 Goals

There are two main aspects of fair machine learning:

- **Measuring unfairness**: A number of measures have been proposed.
- **Reducing unfairness**: For a given ML algorithm, how can we ameliorate its unfairness, yet still maintain an acceptable utility (predictive power) level?

In our earlier COMPAS example: Is the risk assessment tool biased against African-Americans? And if so, how can we reduce that bias while still maintaining good predictive ability?

3.2 Comparison to Part I

First, recall our notation:

- Y: outcome variable, to be predicted
- S: sensitive variable
- O: proxy variables
- X: other variables to be used to predict Y

As before with C, think of X as, at first, consisting of all variables other than Y and S, but then selecting some of X as O, with X then being all variables but Y, S and O.

We wish to predict Y from X and O, omitting S, but with concern that we may be indirectly using S via O. Contrast this from our material in Part I:

- In Part I, we fit models for predicting Y, but with the goal of using such models to assess the effect of S on Y; we were not interested in actually predicting Y. We included S in our models, but wished to find variables C that were correlated with both Y and S, so as to avoid distorting our look at the impact of S on Y. Any X variable unrelated to Y was not of interest.
- Here in Part II, prediction of Y is our central goal, rather than effect assessment. We will omit S, relying our prediction fully on X and partly on O. The variables O are related to S; the stronger the relation of an O variable to S, the less weight we will put on that variable in predicting Y.

We will describe some common measures of unfairness shortly. But first, how do we choose the O variables?

3.2.1 Deciding proxies

As with choosing confounders in Part I, the analyst may simply choose all possible candidate O variables.
the proxies based on his/her domain expertise. But a more formal approach may involve correlation. The function **dsldOHunting** calculates the correlations between S and

It should be kept in mind that, as with any statistic, all utility and fairness measures are subject to sampling variation.

Here's an example, using the COMPAS data:

```
library(dsld)
```

```
Loading required package: fairml
```

```
Loading required package: regtools
```

```
Loading required package: FNN
```

```
**********************


Latest version of regtools at GitHub.com/matloff


Type ?regtools to see function list by category

Loading required package: qeML

Loading required package: rmarkdown

Loading required package: tufte




**********************


  Navigating qeML:

       Type vignette("Quick_Start") for a quick overview!

       Type vignette("Function_List") for a categorized function list

       Type vignette("ML_Overview") for an introduction to machine learning


Attaching package: 'qeML'

The following object is masked _by_ '.GlobalEnv':

    evalr
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

```
data(compas1)
cmp <- compas1[,-3]  # omit decile; we are developing our own risk tool
dsldOHunting(cmp,'two_year_recid','race')
```

```
                            age juv_fel_count juv_misd_count juv_other_count
race.African-American -0.156957517    0.10966300    0.114479498      0.07478931
race.Asian             0.016385005   -0.01239908   -0.005952701     -0.01014768
race.Caucasian         0.147628857   -0.08028929   -0.090027168     -0.04101560
race.Hispanic          0.022247056   -0.03052358   -0.030760605     -0.02561435
race.Native American   0.002329938    0.03159615   -0.011713229      0.01229258
race.Other             0.002437568   -0.03832635   -0.020729116     -0.04670631
                      priors_count   sex.Female     sex.Male    c_jail_in
race.African-American   0.17509146 -0.041459873  0.041459873  0.002270144
race.Asian             -0.03073693 -0.021694502  0.021694502 -0.004670234
race.Caucasian         -0.10017594  0.068682539 -0.068682539  0.002112483
race.Hispanic          -0.06814588 -0.026131928  0.026131928 -0.006687662
race.Native American    0.02388670 -0.006505234  0.006505234 -0.004382357
race.Other             -0.08725931 -0.012952574  0.012952574  0.001166723
                        c_jail_out c_offense_date screening_date   in_custody
race.African-American  0.012784849  -0.0084427004   -0.004593993  0.060971397
race.Asian            -0.004725654  -0.0006422783   -0.004131066 -0.021151109
race.Caucasian        -0.005997440   0.0117472861    0.009717379 -0.027889553
race.Hispanic         -0.009454646  -0.0080839293   -0.004909945 -0.033209571
race.Native American  -0.004711864  -0.0083578408   -0.003890418 -0.005750257
race.Other            -0.001456317   0.0058788162   -0.002179469 -0.027193953
                       out_custody
race.African-American  0.071601855
race.Asian            -0.023006228
race.Caucasian        -0.036894524
race.Hispanic         -0.036660474
race.Native American  -0.003430048
race.Other            -0.027359200
```

The output here suggests possibly using, say, the **age** and **priors_count** variables as proxies.

The function does not use the classic *Pearson product moment* correlation here, opting instead for *Kendall's tau* correlation. Both are widely-used, and both take on values in [-1,], but while Pearson is geared toward continuous numeric variables, Kendall is also usable for binary or ordinal integer-valued variables.

Tau is defined in terms of *concordances* and *discordances*. Say we look at height and weight, and compare two people. If one of the people is both taller and heavier than the other, that is a concordance. If on the other hand, one is shorter but heavier than the other, that is a discordance. We consider all possible pairs of people in our dataset, then sets tau to the difference in concordance and discordance counts, divided by the number of pairs.

3.3 Measuring utility

Utility in ML classification algorithms in general, and both utility and fairness in the fair ML classification realm, often (but far from always) make use of quantitaties like False Positive Rate (FPR). So, let's start by defining these rates.

The famous rates FPR etc.

These are conditional probabilities, but it's important not to confuse the event with the condition.
Consider binary Y classification problems, where we label Y is either *positive* (e.g. patient has the disease) or *negative* (e.g. patient does not have the disease).
After we fit our ML tool to predict Y, we use it for prediction. Consider a long period of time in which we do such predictions. During that time, define the following counts:

- FP: Count of the number of times we predict Y to be positive and actually it's negative.
- FN: Count of the number of times we predict Y to be negative and actually it's positive.
- TP: Count of the number of times we predict Y to be positive and actually it's positive.
- TN: Count of the number of times we predict Y to be negative and actually it's negative.

Then some key rates are:

- FPR: FP / (FP + TN) = P(guess Y positive | Y actually is negative)

It is standard to guess Y = 1 if the probability of that event is at least 0.5. But other thresholds can be used, with TPR and FPR varying as we vary the threshold. The *ROC curve* is the resulting graph of TPR vs. FPR; see **qeROC** in the **qeML** package.

- TPR: TP / (TP + FN) = P(guess Y positive | Y actually is positive)
- FNR: FN / (TP + FN) = P(guess Y negative | Y actually is positive)
- TNR: TN / (FP + TN) = P(guess Y negative | Y actuually is negative)

So for instance, FPR is the proportion of time we guess positive, *among those times* in which Y is actually negative. Two other common terms:

- *recall*: same as TPR
- *sensitivity*: same as TPR
- *precision*: P(Y is actually positive | we guess Y is positive)= (TP + FN) / (TP + FP)

A simple but quite common measure of utility in binary classification problems is the overall misclassification probability of misclassification; **qeML** predictive functions report this in the **testAcc** component of the functions' return object. There are many, many other measures.

The *F1-score* has recently been especially popular in the ML research world. It is defined as the harmonic mean of precision and recall, and is thought to be especially useful in applications in which one class or the other is very rare.

For whatever reason, ML research has tended to focus on that binary Y case, and there are no analogous acronyms for numeric Y. Accuracy of the latter is handled simply as Mean Squared Prediction Error (MSPE), the average squared difference between predicted and actual Y value, or Mean Absolute Prediction Error (MAPE).

3.4 Measuring unfairness

Many unfairness criteria have been proposed. We preseent a few of them in this section. It should be kept in mind that, just as there is no single ML algorithm that predicts the best in all applications, one's choice of fairness measure also will depend on the given application.

See the Xiang and Raji for an excellent analysis of the legal implications of various fairness measures.

3.4.1 S-Correlation

A direct way to measure where Y and S are still related in spite of physically omitting the latter is to compute the correlation between predicted Y, to be denoted $\hat{Y}$ and S. As noted earlier, we use Kendall's Tau correlation here.

For instance, let's consider our mortgage example, say with k-Nearest Neighbors as our ML prediction tool:

Here and below, to keep things simple, we will not use a holdout set.

```
library(dsld)
library(qeML)
z <- qeKNN(cmp,'two_year_recid',holdout=NULL,yesYVal='Yes')
# look at the fitted model's probability of recidivism,
# i.e. regression estimates
probs <- z$regests
# the variable 'race' is an R factor, need a numeric
black <- ifelse(cmp$race=='African-American',1,0)
cor(black,probs,method='kendall')
```

```
[1] 0.2343988
```

That's a pretty substantial correlation, definitely a cause for concern that our ML analysis here is unfair. Of course, it's not the algorithm itself's fault, but we must find a way to mitigate the problem.

An advantage of the S-Correlation measure is that it can also be used in non-classification problems, say predicting wage income, and take age as our sensitive variable S:

Prediction of income might be of interest in, say, a marketing context. Actually, the field of marketing has been the subject of much concern in fair ML; e.g. see FWA.

```
data(svcensus)
z <- qeKNN(svcensus,'wageinc',holdout=NULL)
cor(z$regests,svcensus$age,method='kendall')
```

```
[1] 0.225566
```

So, again, our ML tool seems to be biased, this time in terms of age.

3.4.2 Demographic Parity

The criterion for demographic parity is that the same proportion of each sub-group within the sensitive feature is classified at equal rates for each of the possible outcomes

For example, let's consider the COMPAS dataset again. Demographic Parity would require

P(predict recidivate | Black) = P(predict recidivate | white),

i.e. that the quanitty

(TP+FP) / (TP+FP+TN+FN)

has the same value for each race.

Below is an example using the COMPAS data:

```
z <- qeKNN(cmp,'two_year_recid',holdout=NULL,yesYVal='Yes')
# determine which rows were for Black applicants, which not
BlackRows <- which(cmp$race == 'African-American')
NonblackRows <- setdiff(1:nrow(cmp),BlackRows)  # all the others
# regests, the output of qeKNN, is the vector of fitted probabilities
# (recall that for the Y = 0,1 case, the mean reduces to the probability
# of a 1)
BlackProbs <- z$regests[BlackRows]
NonblackProbs <- z$regests[NonblackRows]
# if a probability is > 0.5, we will guess Y = 1, otherwise guess Y = 0;
# conveniently, that's same as rounding to the nearest integer
BlackYhats <- round(BlackProbs)
NonblackYhats <- round(NonblackProbs)
# again, recall that the mean of a bunch of 0s and 1s is the proportion
# of 1s, i.e. the probability of a 1
mean(BlackYhats)
```

```
[1] 0.5073104
```

```
mean(NonblackYhats)
```

```
[1] 0.2724777
```

That's quite a difference! Overall, our ML model predicts about 51% of Black defendants to recidivate,j versus than 27% for non-Blacks.

However, such a criterion is generally considered too coarse, since it doesn't account for possible differences in qualifications between the two groups. In other words, one must take confounders into account, as we did in Part I.

3.4.3 Equalized Odds

This criterion takes a retrospective view, asking in the case of COMPAS:

> Among those who recidivate, what proportion of them had been predicted to do so? And, does that proportion vary by race?

If the answer to that second question is No, we say our prediction tool satisfies the Equalized Odds criterion.

So, Equalized Odds requires the quantity

TP / (TP+FN)

to be the same for each sensitive group.

3.4.4 The fairness package

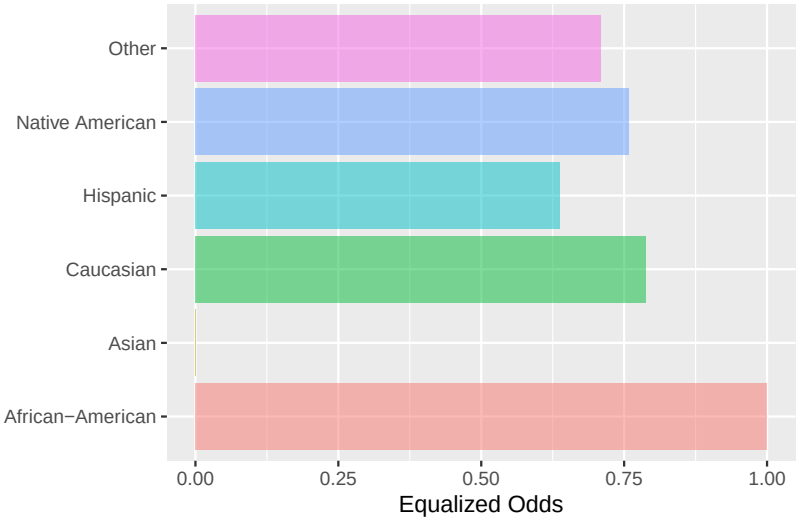
This package calculates and graphically displays a wide variety of fairness criteria. For instance, let's use it to evaluate the Equalized Odds criterion in the above COMPAS example.

```
library(fairness)
```

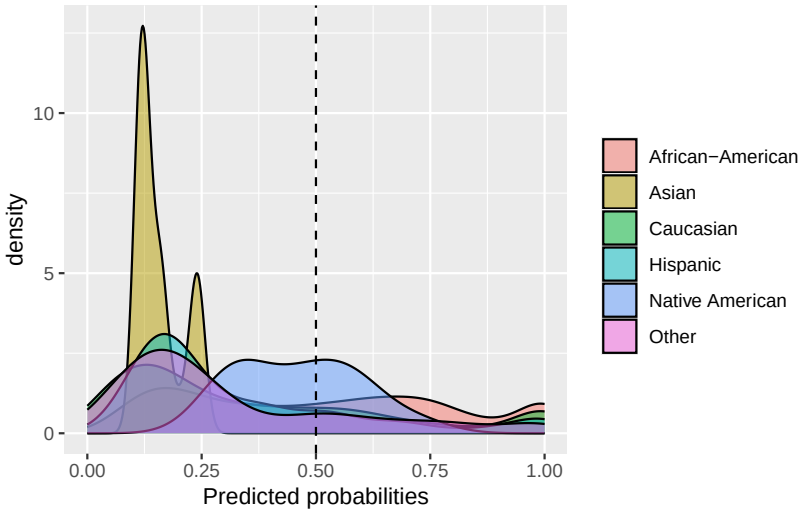
```
Attaching package: 'fairness'

The following object is masked from 'package:fairml':

    compas
```

```
equal_odds(cmp,'two_year_recid','race',probs=z$regests)
```

```
$Metric
             African-American Asian    Caucasian    Hispanic Native American
Sensitivity          0.7536694     0    0.5943627   0.4810811       0.5714286
Equalized odds       1.0000000     0    0.7886253   0.6383184       0.7581952
Group size        2941.0000000    28 2055.0000000 501.0000000      14.0000000
                    Other
Sensitivity      0.5344828
Equalized odds   0.7091740
Group size     316.0000000

$Metric_plot
```




```
$Probability_plot
```



Taking African-Americans as the base, we see that the Equalized Odds criterion was not met, even approximately. Nor was Demographic Parity.

3.5 Remedies

Having established that ML prediction models can be biased against certain sensitive groups, what remedies are available? The **dsld** package includes a few of these, presented in this section. Of course, all of them recognize a basic principle:

> **The Fairness-Utility Tradeoff**
>
> Of course, nothing comes for free: the inherent tradeoff of increasing fairness is reduced utility (reduced predictive power over the dataset). Thus, a means of balancing this tradeoff between fairness and utility becomes essential in any future implementations of machine learning.

Any algorithm for ameliorating unfairness will thus include one or more parameters that one can use to achieve a desired level of compromise between fairness and utility. The parameters essentially allow us to "dial" the weight that our proxies will

play in predicting Y; lighter weight means more fairness but poorer utility, and vice versa.

Our context will be:

- Due to legal requirements or simply a desire for fairness, we will omit S from all analyses, other than for fairness assessment of our derived prediction tool.
- But we are concerned about the impact of proxies, and have chosen a set of variables O to play this role.
- We have chosen fairness and utility measures with which we will choose a desired point in the Fairness-Utility Tradefoff.

Many, many fair ML algorithms have been proposed, most of which are technically complex. The ones we present here have been chosen (a) for their technical simplicity and (b) availability as R packages. We will begin with the simplest algorithms (which have been developed by one of the authors of this book).

Let's take as a running example the COMPAS data, predicting recidivism, with age and prior convictions count as proxies. To illustrate prediction, we will predict the first case in the dataset:

```
newx <- cmp[1,-(7:8)]  # just X and O, not Y and S
```

3.6 dsldQeFairRF

This function fits an RF model, but with deweighting of the proxies.

```
z <- dsldQeFairRF(cmp,'two_year_recid','race',
   list(age=0.2,priors_count=0.1),yesYVal='Yes')
```

```
Warning: Split select weights used. Variable importance measures are only comparable for varial
```

```
z$corrs
```

```
African-American            Asian        Caucasian         Hispanic
      0.16640601       0.13165968       0.12214907       0.16423427
 Native American            Other
      0.02270242       0.14697506
```

```
predict(z,newx)
```

```
$predClasses
[1] "No"

$probs
            No       Yes
[1,] 0.6840967 0.3159033
```

Recall that in RFs, each tree will use a different random ordering of the X and O variables. At any given node in a tree, a variable is chosen at random to set up a possible split point. But here we are specifying that the age and priors count variables be used only 20% and 10% as often as other variables, in order to reduce their impact.

Yet the S-Correlation valuea are still substantial. We need to give the O variables even smaller weights or possibly include additional variables in our O set.

3.7 dsIdQeFairKNN

Remember, the common theme here is reducing the role played in prediction by the proxies O. How might this be done with k-NN?

An easy answer is to weight the distance metric. Ordinarily, if we move 3.2 meters to the right and then 1.1 meters forward, the distance between our new point and our original one is

$\sqrt{3.2^2 + 1.1^2} = 3.38$

That puts equal weight in the left-right direction and the forward-back direction, which makes sense for geometric distance. But in data prediction, we can use different weights. In prediction wage income in the Census data, say, we can place large weight on age and occupation, and less weight on education. The above call would change to:

```
z <- dsldQeFairKNN(cmp,'two_year_recid','race',
   list(age=0.2,priors_count=0.1),yesYVal='Yes')
z$corrs
predict(z,newx)
```

3.8 Remedies based on "shrunken" linear models

One of the most striking advances in modern statistics was the discovery that classical estimators tend to be "too big," and that one may improve accuracy by "shrinking" them. Here is the intuition:

Consider the example in Section 2.7. The vector of estimated regression coefficients was

(12.46,-0.94,0.39,0.16,...)

We might (crudely) shrink this vector by multiplying by a factor of, say, 0.5, yielding

(6.23,-0.47,0.20,0.08,...)

The actual shrinkage mechanisms used in the regression context are much more complex than simply multiplying every component of the vector by the same constant, but this is the basic principle.

Why might this odd action be helpful? Shrinking introduces a bias, but reduces variance (roughly speaking, smaller variables vary less). If outliers (extreme values, not errors) are common in the data, these result in large estimator variance, possibly so much that shrinkage's reduction in variance overwhelms our increase in bias.

Some readers may have heard of *ridge regression* and the *LASSO*, both of which impose shrinkage. For our fair ML context, though we wish to perform our shrinkage focusing on *only some* of the estimated regression coefficients, specifically those corresponding to our proxies. To be sure, it should be noted that non-proxy coefficients are affected too, but the main effects will be on the proxies.

Note that these estimators also have the potential ancillary benefit obtained from shrinkage in general, i.e. better utility.

3.8.1 The dsldFgrrm function

This is a wrapper to a corresponding function in the **fairml** package, based on a paper by the authors of the package.

Let's see what it does with the COMPAS data:

```
dsldFgrrm(cmp,'two_year_recid','race',unfairness=0.1,family='binomial')
dsldFgrrm(cmp,'two_year_recid','race',unfairness=0.01,family='binomial')
```

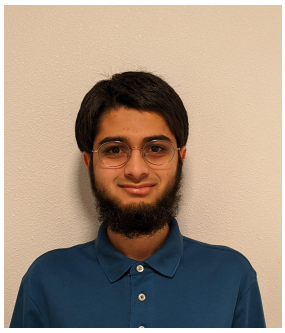
The **unfairness** argument is a number in (0,1]. The smaller the value, the fairer the fit. The first value, 0.1, gave an estimate coefficient for Caucasian of -0.73, while using 0.01 reduced this to -0.15. (Note that the base for the dummy variables is apparently African-American.] But both values are small relative some of the other coefficients.

4 Author Bios

Norman Matloff

is Professor Emeritus of Computer Science, and was formerly a Professor of Statistics at that university. His participation in this project is informed in part by his experience serving as an expert witness in in a number of litigation cases involving discrimination. His work has been recognized in various forms, including the university-wide Distinguished Teaching Award, Outstanding Adviser Award, and Distinguished Public Service Award. His book, *Statistical Regression and Classification: from Linear Models to Machine Learning* was the 2017 recipient of the Ziegal Award, given by the statistics journal *Technometrics*.

Taha Abdullah



is studying for a B.S. in Computer Science at University of California, Davis. He has a keen interest in pursuing a career in Software Engineering.

Arjun Ashok

is currently a sophomore undergraduate at UC Davis double-majoring in Computer Science and Statistics with a minor in Economics. As indicated by his past research and publications, he's primarily interested in the R&D side of machine learning, although he's intent on taking his knowledge into real-world applications for developmental purposes.

Shubhada Martha

is a third year Computer Science student pursuing her undergraduate studies at UC Davis. She's passionate about ethics in technology in the areas of full stack development and AI and Machine Learning. During her spare time, she likes to read spiritual and self-improvement books and paint.

Aditya Mittal

is pursuing a B.S. in Statistics with a minor in Computer Science at University of California, Davis. During the summer, he is employed as a Business Analyst Intern at Cisco and will be returning in summer 2024 for another round. He is interested in pursuing a career in machine learning/software engineering. Fun fact: He moved to the U.S. in 2014 from Mumbai, India.

Billy Ouattara

is a senior Computer Science and Engineering student, His academic journey has been defined by a strong commitment to ethical considerations in the development and deployment of ML/AI models. This dedication is one of the key reasons he eagerly participated in this project.

Jonathan Tran

is studying computer science at UC Davis, with an interest in machine learning and large language models.

Brandon Zarate

is a senior student at UC Davis studying Computer Science.

5 Appendices

5.1 Appendix A: Standard Errors—Statistical Inference in a Nutshell

Say I wish to find the mean weight μ of all students at the University of California, Davis. It would be infeasible to measure them all, so I take a random sample of 100 students. The mean weight of those 100 students, denoted $\bar{X}$, is an estimate of μ.

I know that $\bar{X}$ will have some unknowm amount of error as an estimate of μ. One measure of this is its *standard error* (SE). What is this?

Remember, we took a random sample of students. Different samples will have different values of $\bar{X}$, i.e. $\bar{X}$ will have *sampling variation*. Then the SE of $\bar{X}$ is defined to be the standard deviation of all possible $\bar{X}$ values, from all possible samples.

The smaller the SE is, the more confident we are that the $\bar{X}$ from our particular sample is pretty accurate. Of course, we can't be completely sure, but we can quantify it probabilistically:

> An approximate 95% *confidence interval* for μ, based on $\bar{X}$, is $\bar{X} \pm 1.96 \times SE$.

Meaning of that 95% probability figure: Think of all possible samples. Each one has an $\bar{X}$ value, and an SE value. So, each possible sample, there is a different confidence interval (CI). 95% of those CIs will contain the true μ.

The reader may have heard of the Central Limit Theorem, which says that the sum of many random variables, itself a random variable, has an approximately normal distribution. This is the case for most classical statistical estimators, such as estimated coefficients in linear and generalized linear models. Hence we can find confidence intervals as above, taking the estimator plus and minus 1.96 times the SE of the estimator.

CIs are generally taught together with *hypothesis testing* or *significance testing*, which in many cases uses standard errors. Section 2.6 reviews the notion, and discusses problems with this kind of analysis.

5.2 Appendix B: Standard Errors via the Bootstrap

(to be added)

5.3 Appendix C: Python Interface

As previously mentioned, Python wrappers are included for most functions.

5.3.1 Requirements For All Python Functions

To use these Python functions, users need to have Python version **3.10** installed on their system.

If the user has a later version of Python installed (such as 3.11), and wants to keep that version of Python in their system, they will have to manage multiple Python versions on their system. One easy way to manage multiple environments (in our case, multiple versions of Python for users who need it) is to use Conda, which is a package and environment manager included in both Anaconda and Miniconda. For our purposes, Miniconda is more useful as a minimalistic way to install and use Conda. See this guide for information on how to install Miniconda – keep in mind that, at this time, this package requires Python 3.10 (NOT the latest version of Python), so when the guide prompts you to choose a Miniconda installer, you can choose a link for Python 3.10 from this list which is also linked in the aforementioned guide.

After installing miniconda, refer to this section and later sections of the Conda user guide to set up an environment with Python 3.10.

Python 3.10 conveniently comes with **pip**, the Python package installer. It can be used to install various Python packages/modules that will be required to run the dsld Python functions. **All** of our Python functions require **rpy2**, which allows for Python/R environment interaction.

To install rpy2, run **pip install rpy2** within the Python 3.10 environment, or see this link

5.3.2 Running dsld Python Functions

As an example, let's use the Python interface for ##dsldTakeALookAround## (the Python-equivalent function is called **dsldPyTakeALookAround**)
to view tabular information on how data features relate to Y and S. The Python function allows users to enter arguments using Python data types and converts the resultant R dataframe into a Pandas dataframe.

Different Python functions will require different libraries depending on what the function does – **Pandas** is required for dsldPyTakeALookAround.

These libraries can also (usually) be installed with **pip**

The user should open the Python Shell Prompt by typing **python** from the package Python directory (/dsld/inst/Python) in the terminal/cmd prompt. Then, to run the Python function, we can run the following commands in the Python shell:

```
python} from dsldTakeALook_Py_R import dsldPyTakeALookAround
import rpy2.robjects as robjects    robjects.r['data']('svcensus')
data = robjects.r('svcensus')     result = dsldPyTakeALookAround(data,
'wageinc', 'gender')    print(result)
```

The result should be the same information from the R dataframe, but converted to a Pandas dataframe.