BCB 726: *Machine Learning for Computational Biology*


# The Three Cultures of Data

*October 22nd, 2025*

# “Cultures”?

*Abstract.* There are two cultures in the use of statistical modeling to reach conclusions from data. One assumes that the data are generated by a given stochastic data model. The other uses algorithmic models and treats the data mechanism as unknown. The statistical community has been committed to the almost exclusive use of data models. This commitment has led to irrelevant theory, questionable conclusions, and has kept statisticians from working on a large range of interesting current problems. Algorithmic modeling, both in theory and practice, has developed rapidly in fields outside statistics. It can be used both on large complex data sets and as a more accurate and informative alternative to data modeling on smaller data sets. If our goal as a field is to use data to solve problems, then we need to move away from exclusive dependence on data models and adopt a more diverse set of tools.

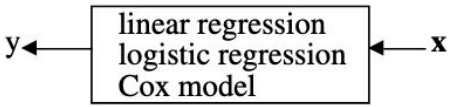
## Statistical Modeling: The Two Cultures

**Leo Breiman**

### The Data Modeling Culture

The analysis in this culture starts with assuming a stochastic data model for the inside of the black box. For example, a common data model is that data are generated by independent draws from

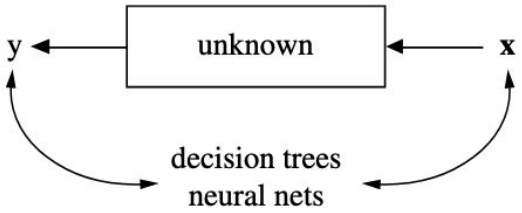
response variables = $f$(predictor variables, random noise, parameters)




*Model validation.* Yes–no using goodness-of-fit tests and residual examination.
*Estimated culture population.* 98% of all statisticians.

### The Algorithmic Modeling Culture

The analysis in this culture considers the inside of the box complex and unknown. Their approach is to find a function $f(\mathbf{x})$—an algorithm that operates on $\mathbf{x}$ to predict the responses $\mathbf{y}$. Their black box looks like this:




*Model validation.* Measured by predictive accuracy.
*Estimated culture population.* 2% of statisticians, many in other fields.

# The Three Cultures of Data

- **Statistics**
- **Machine Learning**
- **Deep Learning**

…and a lot of other ones we won't talk about:

*Bayesian Statistics, Statistical Learning, Econometrics, Cybernetics, Control Theory, Signal Processing, Inverse Problems, Operations Research, Causal Inference, Actuarial Science, &c*

**What do I mean by culture?**

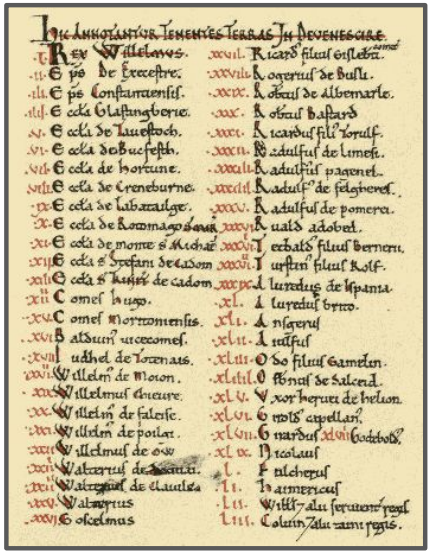
A community of researchers with socially determined assumptions about:

- What is the goal of distilling data into a mathematical model?
- What are the important properties of a data infused mathematical model?

…who then teach classes and write textbooks, creating a socially constructed aura of "the right way to work with data"

# *A historical interlude*

# Measurement & estimation aren't new...



William The Conqueror's Domesday Book of 1086

*"What did I just conquer?"*

וַיְדַבֵּר יְהוָה אֶל־מֹשֶׁה לֵאמֹר׃

יהוה spoke to Moses, saying:

כִּי תִשָּׂא אֶת־רֹאשׁ בְּנֵי־יִשְׂרָאֵל לִפְקֻדֵיהֶם וְנָתְנוּ אִישׁ כֹּפֶר נַפְשׁוֹ לַיהוָה
בִּפְקֹד אֹתָם וְלֹא־יִהְיֶה בָהֶם נֶגֶף בִּפְקֹד אֹתָם׃

When you take a census of the Israelite men according to their army enrollment, each shall pay יהוה a ransom for himself on being enrolled, that no plague may come upon them through their being enrolled.

זֶה | יִתְּנוּ כָּל־הָעֹבֵר עַל־הַפְּקֻדִים מַחֲצִית הַשֶּׁקֶל בְּשֶׁקֶל הַקֹּדֶשׁ עֶשְׂרִים
גֵּרָה הַשֶּׁקֶל מַחֲצִית הַשֶּׁקֶל תְּרוּמָה לַיהוָה׃

This is what everyone who is entered in the records shall pay: a half-shekel by the sanctuary weight—twenty *gerah*s to the shekel—a half-shekel as an offering to יהוה.

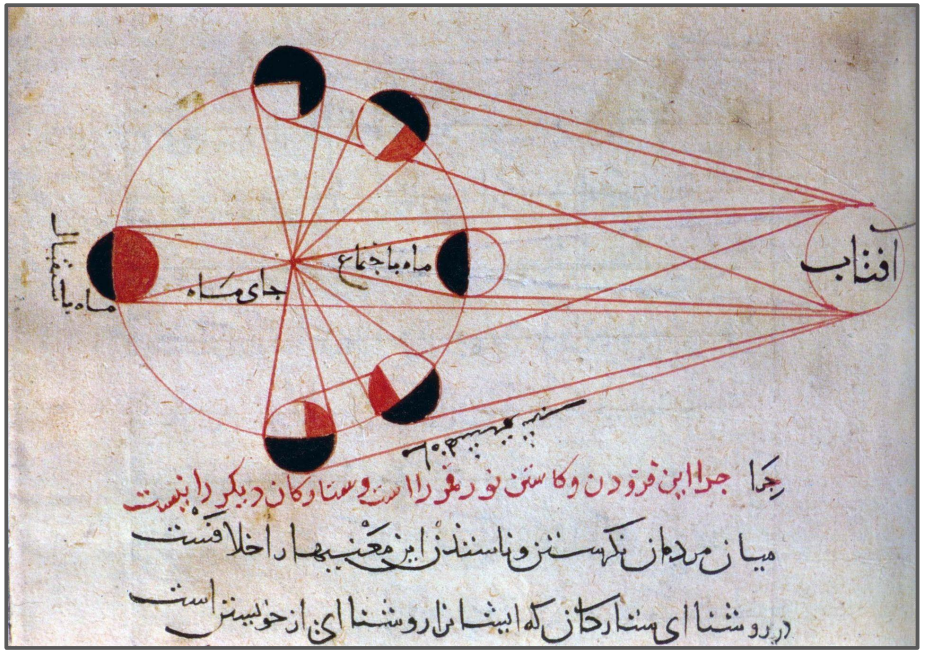
כֹּל הָעֹבֵר עַל־הַפְּקֻדִים מִבֶּן עֶשְׂרִים שָׁנָה וָמָעְלָה יִתֵּן תְּרוּמַת יְהוָה׃

Everyone who is entered in the records, from the age of twenty years up, shall give יהוה's offering:

Beginning of the Torah portion *Ki Tisa*, Moses asked to take a census and collect a tax

# Sparks of mathematical data modeling in early science

- Interplay between **observation** & strong **inductive** priors
- Abstract models <u>but</u> often more descriptive than mathematical
- Validated by agreement with observation <u>but</u> also:
  - Elegance / aesthetics
  - Great “masters”
  - Theology & philosophy



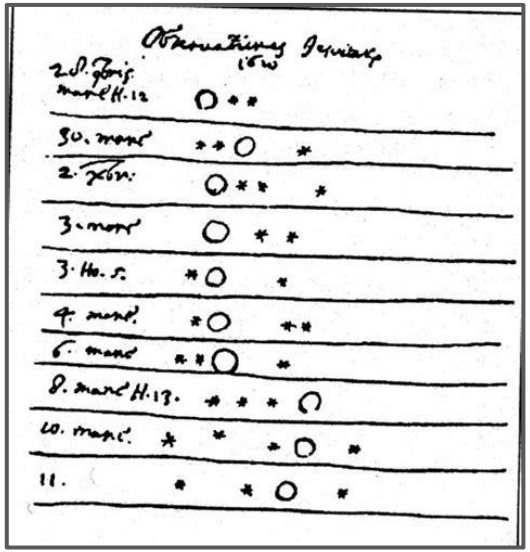

Abu-Rayhan al-Biruni’s *Al-Tafhim li Awa'il Sana'at al-Tanjim* (Book on the Elements of Astrology)

# Modern science = strict empiricism

- 1600s science narrowed "natural philosophy" to:
  - **Collect data**
  - **Build mathematical models** (repeat)

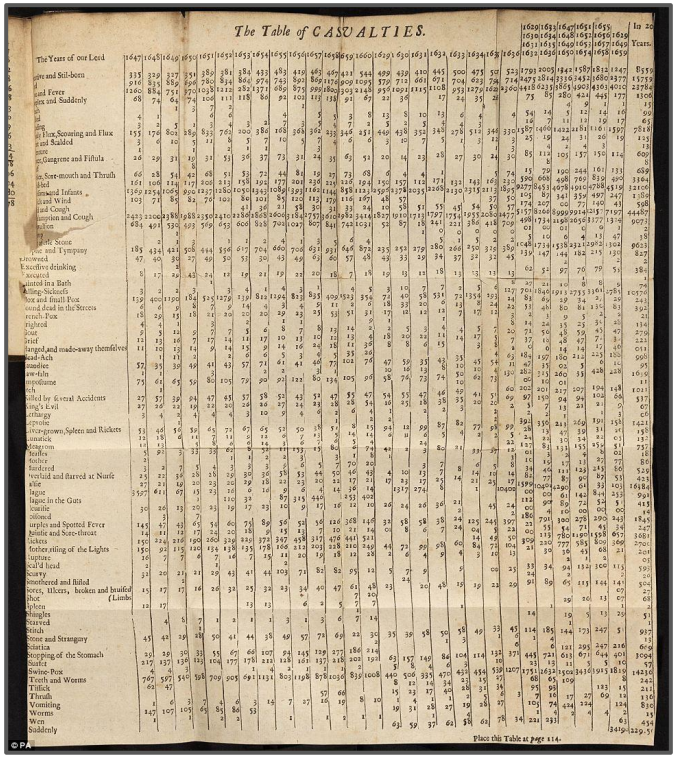
*"the Universe — which stands continually open to our gaze, but it cannot be understood unless one first learns to comprehend the language and interpret the characters in which it is written. It is written in the language of mathematics"* – Galileo in The Assayer


Galileo Gallilei's "The Assayer"



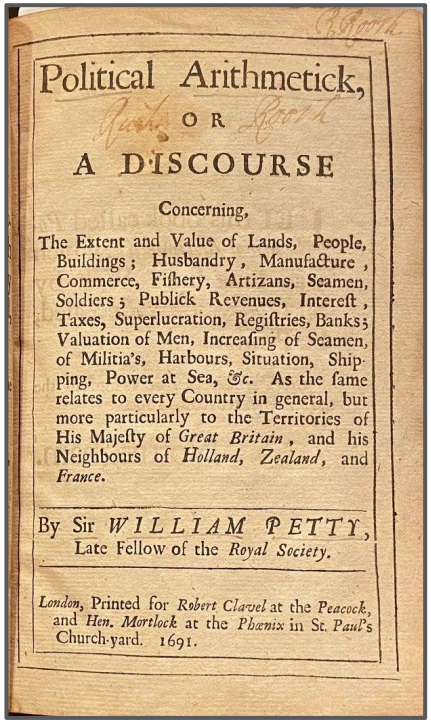
Galileo's notebook observing moons of Jupiter (interpreted as evidence for refuting geocentrism)

# Statistics = seeing like a state



The Table of CASUALTIES.

Table from John Graunt's *Natural and Political Observations Made Upon the Bills of Mortality* (1662)



Political Arithmetick,

OR

A DISCOURSE

Concerning,

The Extent and Value of Lands, People, Buildings; Husbandry, Manufacture, Commerce, Fishery, Artizans, Seamen, Soldiers; Publick Revenues, Interest, Taxes, Superlucration, Registries, Banks; Valuation of Men, Increasing of Seamen, of Militia's, Harbours, Situation, Shipping, Power at Sea, &c. As the same relates to every Country in general, but more particularly to the Territories of His Majesty of *Great Britain*, and his Neighbours of *Holland*, *Zealand*, and *France*.

By Sir *WILLIAM PETTY*, Late Fellow of the *Royal Society*.

*London*, Printed for *Robert Clavel* at the *Peacock*, and *Hen. Mortlock* at the *Phœnix* in St. *Paul's* Church-yard. 1691.

William Petty's *Political Arithmetic* (1691)

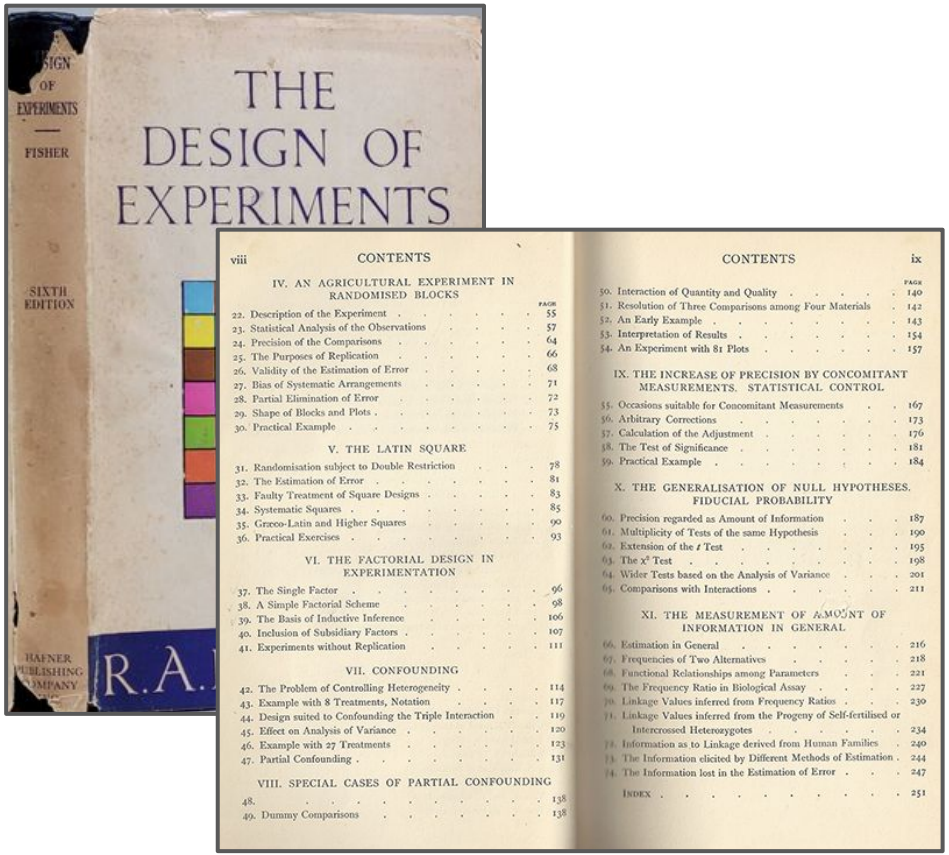
# *Statistics*

# So, what is classical statistics?

- There is some real world quantity
  - …how do we finite noisy measurements into a robust estimate of the “true” value?

- We have a mathematical model of reality
  - …how do rigorously we use finite noisy measurements to reject (or conditionally accept) the model?

# 1920s/30s: statistics infiltrates science




viii CONTENTS

IV. AN AGRICULTURAL EXPERIMENT IN RANDOMISED BLOCKS



V. THE LATIN SQUARE



VI. THE FACTORIAL DESIGN IN EXPERIMENTATION



VII. CONFOUNDING



VIII. SPECIAL CASES OF PARTIAL CONFOUNDING



CONTENTS ix



IX. THE INCREASE OF PRECISION BY CONCOMITANT MEASUREMENTS. STATISTICAL CONTROL

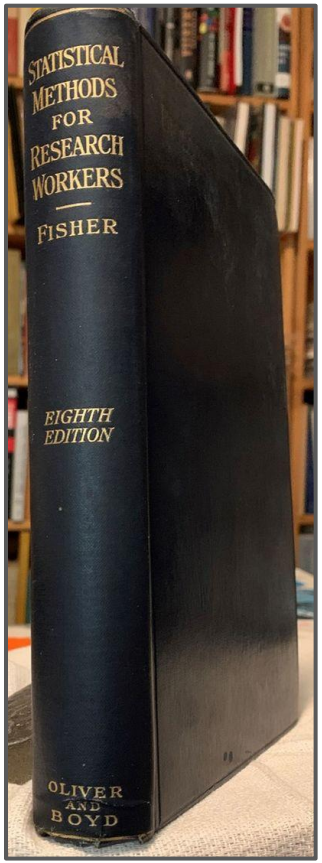


X. THE GENERALISATION OF NULL HYPOTHESES. FIDUCIAL PROBABILITY



XI. THE MEASUREMENT OF AMOUNT OF INFORMATION IN GENERAL





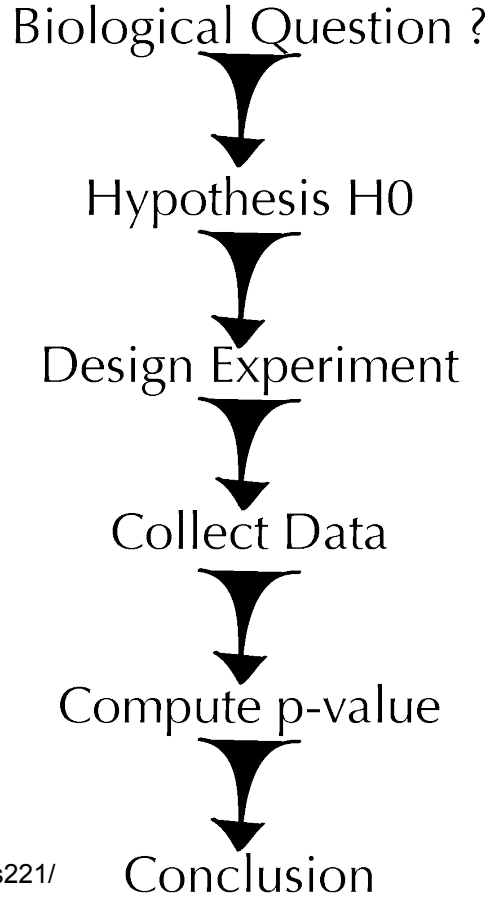

## CONTENTS



## TABLES

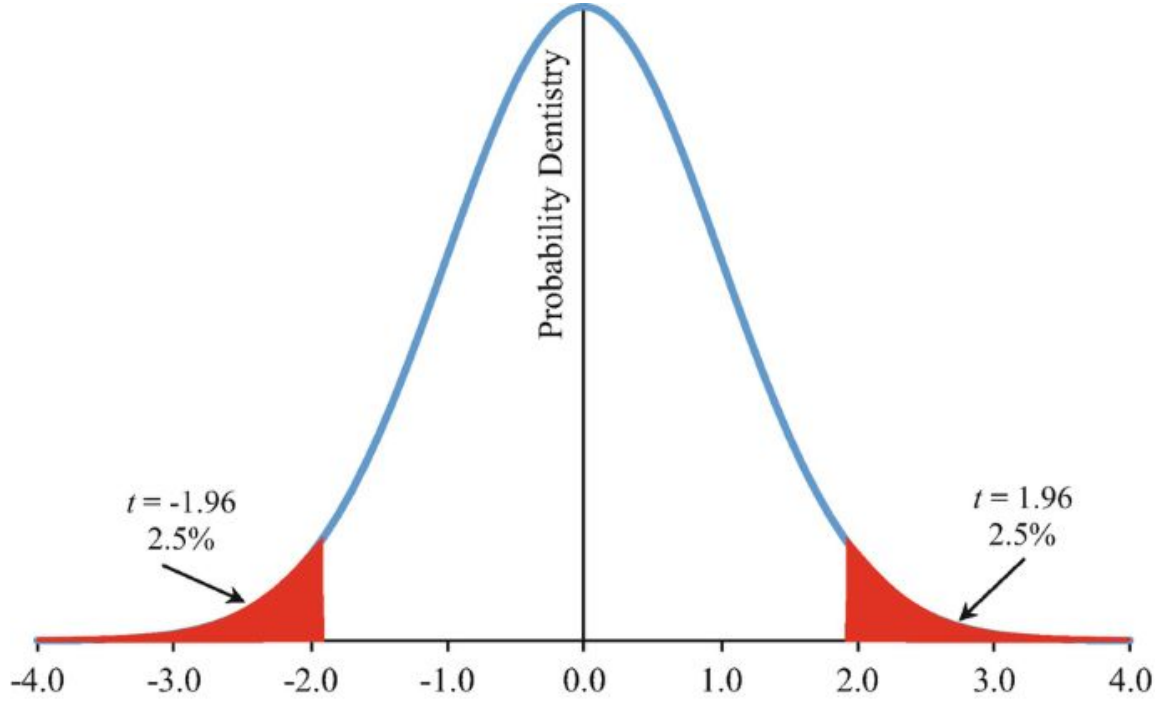
# Tools of classical statistics

- Experimental design
- Inference & estimators
  - consistent, unbiased, efficient
- Confidence intervals
- Statistical hypothesis testing (Neyman-Pearson)
- Null hypothesis models (Fisher)
- Their unholy marriage: NHST




https://web.stanford.edu/class/bios221/

# Null Hypothesis Significance Testing




*Teaching Null Hypothesis Significance Testing (NHST) in the Health Sciences: The Significance of Significance*

# Exploratory Data Analysis: You're allowed to look at your data



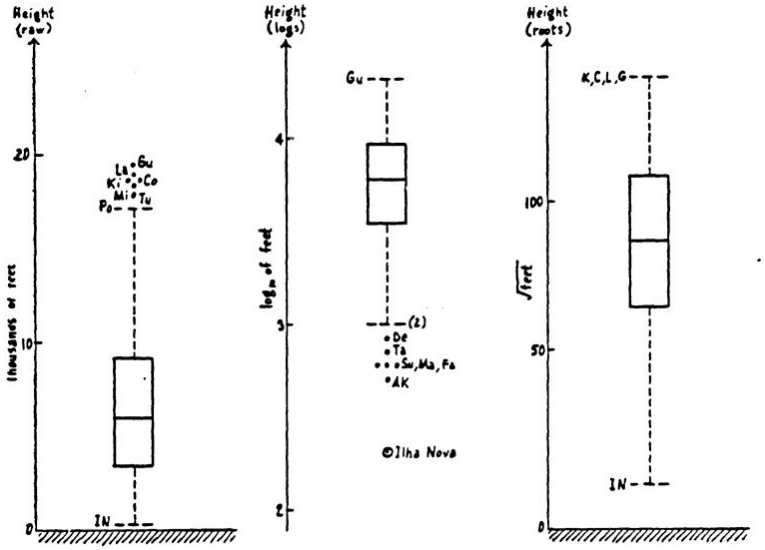
**Exploratory Data Analysis: Past, Present, and Future**

*John W. Tukey*[1]

*Technical Report No. 302*
*Princeton University, 408 Fine Hall, Washington Road, Princeton, NJ 08544-1000*

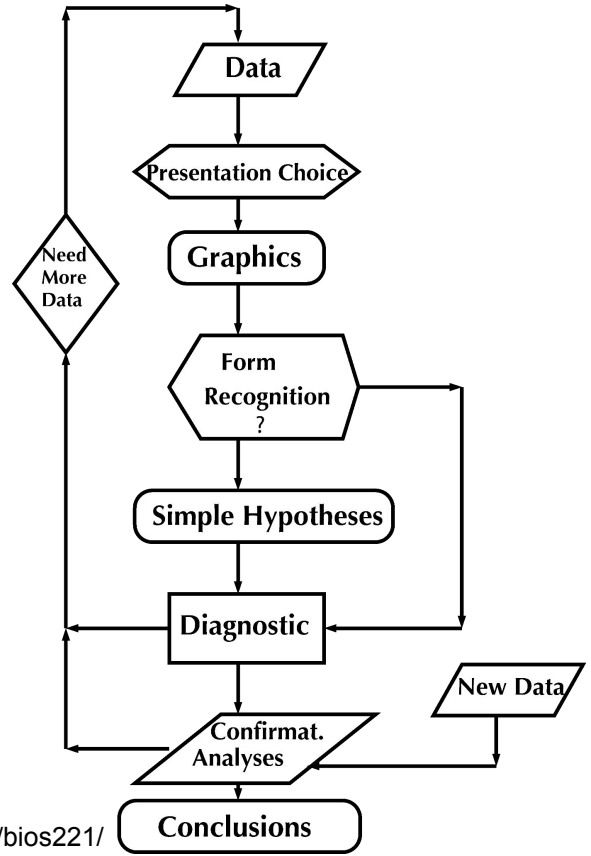
**Abstract**

The 1971-1977 early formulation of Exploratory Data Analysis, in terms of (a) results of some of its techniques and considerations which underlay, at various depths, the choices realized in the books. The 1991–1995 development of Exploratory Analysis of Variance, described in its simplest (two–way table) form and barely sketched in general. Discussion of the changes in apparent philosophy caused by the need to communicate more complicated things, notches, hints, the likely impact on a revised edition of Exploratory Data Analysis 1977. Dreams and targets for what might happen in 1996–2005, with emphasis on Exploratory Regression and the combined use of multiple description.



# Exploratory Data Analysis: workflow

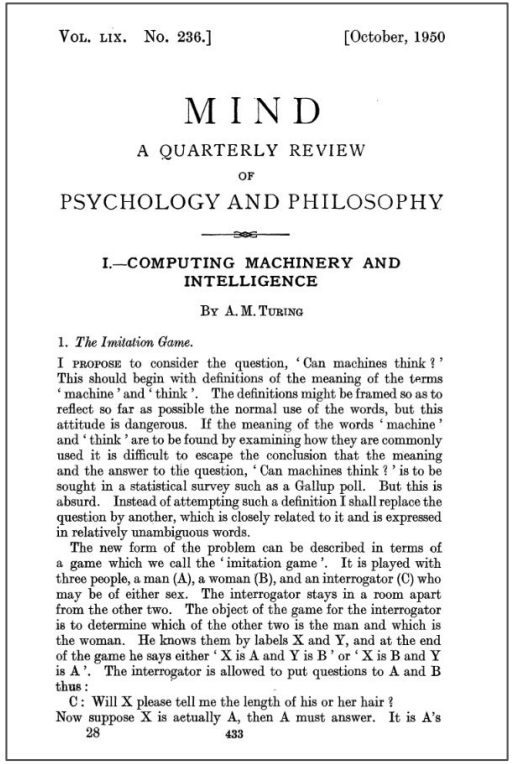
- Emphasis on visualization
- Look at residuals of your model
  - …you can even try different models!
- Does it make sense?
- …classical stats left for “confirmatory analyses”




https://web.stanford.edu/class/bios221/

# *Machine Learning*

# ML deep origins (40s-60s): let's make artificial intelligence with logic!



VOL. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

BY A. M. TURING

1. *The Imitation Game.*

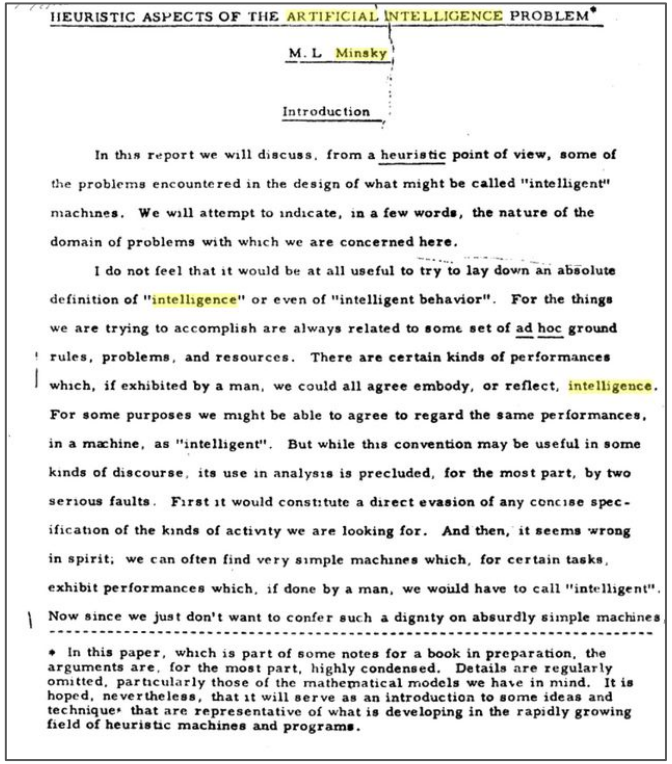
I PROPOSE to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, 'Can machines think?' is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words.

The new form of the problem can be described in terms of a game which we call the 'imitation game'. It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either 'X is A and Y is B' or 'X is B and Y is A'. The interrogator is allowed to put questions to A and B thus:

C: Will X please tell me the length of his or her hair?

Now suppose X is actually A, then A must answer. It is A's





HEURISTIC ASPECTS OF THE ARTIFICIAL INTELLIGENCE PROBLEM*

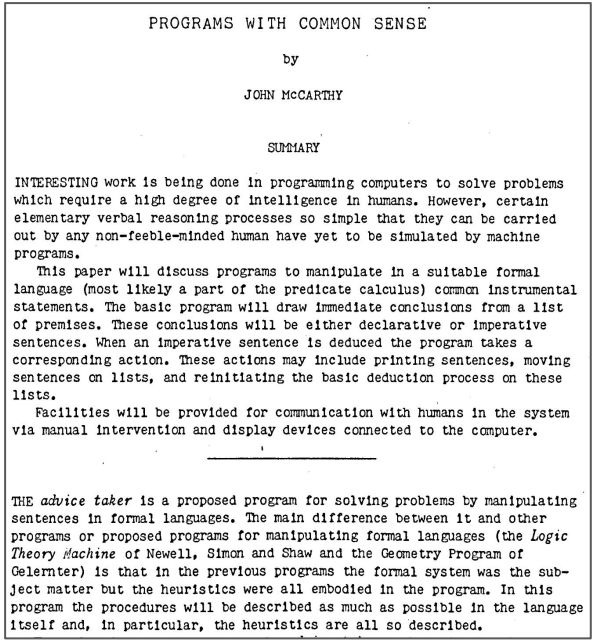
M. L Minsky

Introduction

In this report we will discuss, from a heuristic point of view, some of the problems encountered in the design of what might be called "intelligent" machines. We will attempt to indicate, in a few words, the nature of the domain of problems with which we are concerned here.

I do not feel that it would be at all useful to try to lay down an absolute definition of "intelligence" or even of "intelligent behavior". For the things we are trying to accomplish are always related to some set of ad hoc ground rules, problems, and resources. There are certain kinds of performances which, if exhibited by a man, we could all agree embody, or reflect, intelligence. For some purposes we might be able to agree to regard the same performances, in a machine, as "intelligent". But while this convention may be useful in some kinds of discourse, its use in analysis is precluded, for the most part, by two serious faults. First it would constitute a direct evasion of any concise specification of the kinds of activity we are looking for. And then, it seems wrong in spirit; we can often find very simple machines which, for certain tasks, exhibit performances which, if done by a man, we would have to call "intelligent". Now since we just don't want to confer such a dignity on absurdly simple machines

* In this paper, which is part of some notes for a book in preparation, the arguments are, for the most part, highly condensed. Details are regularly omitted, particularly those of the mathematical models we have in mind. It is hoped, nevertheless, that it will serve as an introduction to some ideas and techniques that are representative of what is developing in the rapidly growing field of heuristic machines and programs.



PROGRAMS WITH COMMON SENSE

by

JOHN McCARTHY

SUMMARY

INTERESTING work is being done in programming computers to solve problems which require a high degree of intelligence in humans. However, certain elementary verbal reasoning processes so simple that they can be carried out by any non-feeble-minded human have yet to be simulated by machine programs.

This paper will discuss programs to manipulate in a suitable formal language (most likely a part of the predicate calculus) common instrumental statements. The basic program will draw immediate conclusions from a list of premises. These conclusions will be either declarative or imperative sentences. When an imperative sentence is deduced the program takes a corresponding action. These actions may include printing sentences, moving sentences on lists, and reinitiating the basic deduction process on these lists.

Facilities will be provided for communication with humans in the system via manual intervention and display devices connected to the computer.

---

THE *advice taker* is a proposed program for solving problems by manipulating sentences in formal languages. The main difference between it and other programs or proposed programs for manipulating formal languages (the *Logic Theory Machine* of Newell, Simon and Shaw and the Geometry Program of Gelernter) is that in the previous programs the formal system was the subject matter but the heuristics were all embodied in the program. In this program the procedures will be described as much as possible in the language itself and, in particular, the heuristics are all so described.

# ML origins (60s-80s): symbolic AI isn't working, seems like AIs need to work with data

Creates many less ambitious sub-fields:

- Information Retrieval
- Natural Language Processing
- Computer Vision
- Speech Recognition
- …and a base of techniques that mix:
  - *Computer Science*
  - *Optimization*
  - *Signal Processing*

**Some Studies in Machine Learning Using the Game of Checkers**

**Abstract:** Two machine-learning procedures have been investigated in some detail using the game of checkers. Enough work has been done to verify the fact that a computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program. Furthermore, it can learn to do this in a remarkably short period of time (8 or 10 hours of machine-playing time) when given only the rules of the game, a sense of direction, and a redundant and incomplete list of parameters which are thought to have something to do with the game, but whose correct signs and relative weights are unknown and unspecified. The principles of machine learning verified by these experiments are, of course, applicable to many other situations.

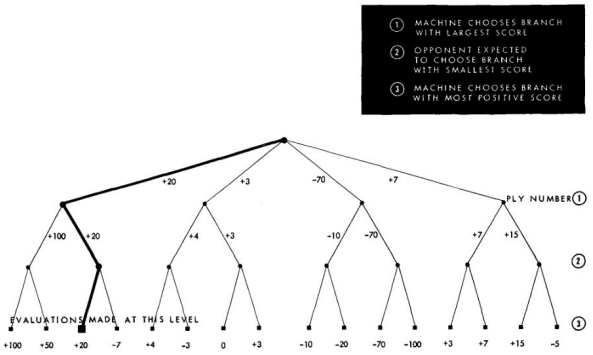



*Figure 2* Simplified diagram showing how the evaluations are backed-up through the "tree" of possible moves to arrive at the best next move. The evaluation process starts at ②.

# Deep Learning origins (40s-60s): let's imitate brains with computers

## A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH and WALTER H. PITTS

Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means for nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. It is shown that many particular choices among possible neurophysiological assumptions are equivalent, in the sense that for every net behaving under one assumption, there exists another net which behaves under the other and gives the same results, although perhaps not in the same time. Various applications of the calculus are discussed.

### INTRODUCTION

THEORETICAL neurophysiology rests on certain cardinal assumptions. The nervous system is a net of neurons, each having a soma and an axon. Their adjunctions, or synapses, are always between the axon of one neuron and the soma of another. At any instant a neuron has some threshold, which excitation must exceed to initiate an impulse. This, except for the fact and the time of its occurrence, is determined by the neuron, not by the excitation. From the point of excitation the impulse is propagated to all parts of the neuron. The velocity along the axon varies directly with its diameter, from less than one meter per second in thin axons, which are usually short, to more than 150 meters per second in thick axons, which are usually long. The time for axonal conduction is consequently of little importance in determining the time

## Analysis of a Four-Layer Series-Coupled Perceptron. II*

H. D. BLOCK, B. W. KNIGHT, JR., AND F. ROSENBLATT

*Cornell University, Ithaca, New York*

### 1. INTRODUCTION

THE preceding paper[1] presented motivation and background for the general subject of perceptrons and gave some analysis and results for a simple three-layer perceptron. While it has been shown there that it is possible to associate any arbitrary set of responses to an arbitrary set of stimuli in a simple three-layer perceptron, such a perceptron characteristically requires a large representative sample of each kind of pattern (e.g., letters "A" and "B"), covering all parts of the retina, before it will recognize an arbitrarily positioned stimulus which is similar to one which it has seen before. In other words, a three-layer perceptron has no concept of "similarity" based on any criterion other than the intersections of sets of retinal elements. In a previous paper,[2] Rosenblatt has shown that a "cross-coupled perceptron," in which $A$ units are connected to one another by modifiable connections, should tend to develop an improved similarity criterion for generalizing responses from one stimulus to another when exposed to a suitably organized environment. In this paper a simpler network, consisting of four layers of units but without cross coupling, is analyzed in a more rigorous fashion, and is shown to possess the same property.

The perceptron of the present paper is "self-organizing" in the sense that during the training period the experimenter does not tell the machine the category of each stimulus. As the analysis below will show, the only contact between the experimenter and the machine is the presentation of the stimuli.

### 2. THE MODEL

The model to be analyzed here is a four-layer perceptron of the schematic type $S$–$A^{I}$–$A^{II}$–$R$, as indicated in Fig. 1.

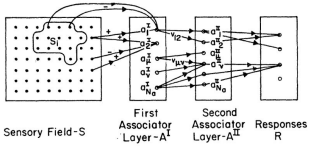



FIG. 1. Organization of four-layer series-coupled perceptron.

* Research sponsored by the Office of Naval Research.

[1] H. D. Block, Revs. Modern Phys. **34**, 123 (1962).

[2] See, F. Rosenblatt, in *Self-Organizing Systems*, edited by M. Yovits and S. Cameron (Pergamon Press, New York, 1960).

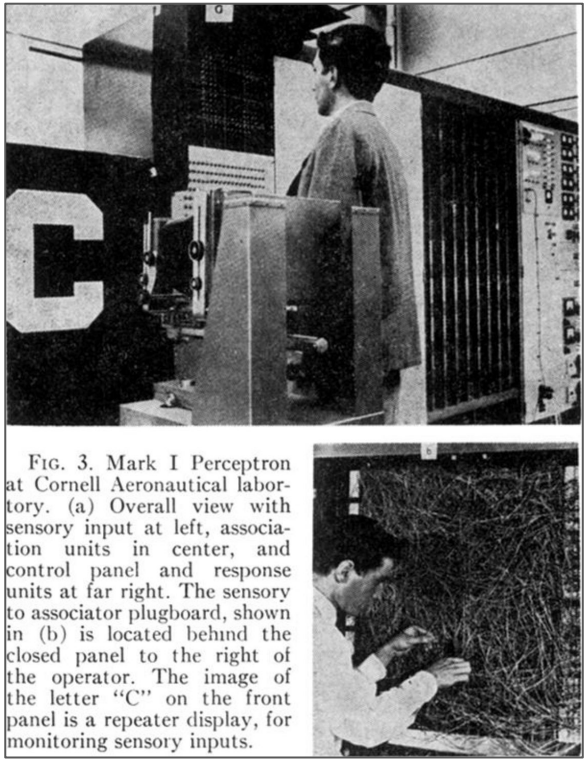


FIG. 3. Mark I Perceptron at Cornell Aeronautical laboratory. (a) Overall view with sensory input at left, association units in center, and control panel and response units at far right. The sensory to associator plugboard, shown in (b) is located behind the closed panel to the right of the operator. The image of the letter "C" on the front panel is a repeater display, for monitoring sensory inputs.

# Machine Learning

- Forget models of reality!
- …let's just do **function approximation** instead.
- Data is some set of **featurized vectors**
  - can have labels (or not)
- Learn functions parameterized by **weights**
  - minimize a **loss function** however you can
  - control **model complexity** to avoid overfitting
- Evaluate **predictive** accuracy on held-out data

# Deep Learning

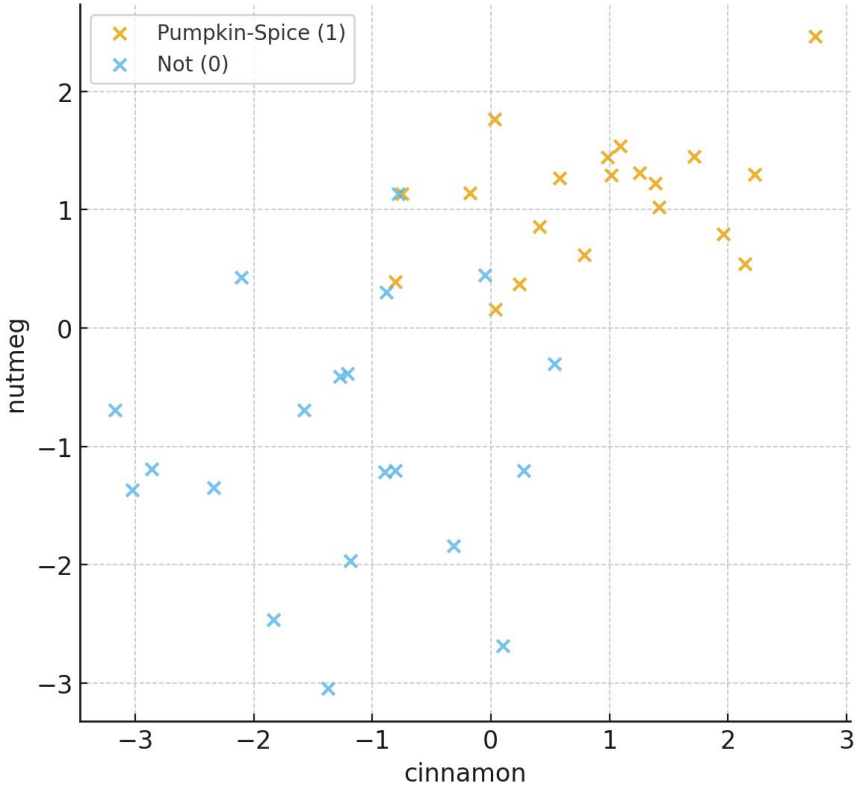
Like machine learning but…

- terminology inspired by brains
  - “**neurons**”
  - “**layers**”
- make the models as big as possible
  - surprisingly, **overfitting doesn’t matter**
- train arbitrary **differentiable compute graphs** with **stochastic gradient descent**

# *A Pumpkin Flavored Example*

# Is it a Pumpkin Spice Latte?

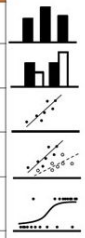
```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score, log_loss, accuracy_score, brier_score_loss

rng = np.random.default_rng(0)

# Tiny, overlapping 2D Gaussians (avoid separation)
n_per = 20
ps_mean = np.array([ 0.8,  0.8])  # Pumpkin-Spice: higher cinnamon/nutmeg
no_mean = np.array([-0.8, -0.8])  # Not
cov = np.array([[1.0, 0.2],
                [0.2, 1.0]])

X_ps = rng.multivariate_normal(ps_mean, cov, size=n_per)
X_no = rng.multivariate_normal(no_mean, cov, size=n_per)
X = np.vstack([X_ps, X_no])
y = np.hstack([np.ones(n_per), np.zeros(n_per)])

df = pd.DataFrame(X, columns=["cinnamon","nutmeg"]).assign(psl=y.astype(int))

# Shared train/test split; *all* cultures use the same training data
X_train, X_test, y_train, y_test = train_test_split(
    df[["cinnamon","nutmeg"]].values, df["psl"].values,
    test_size=0.4, random_state=0, stratify=df["psl"].values
)
```



# Statistics Flavored Logistic Regression

| Dependent variable | Explanatory variables | Model type |
|---|---|---|
| Quantitative | 1 qualitative with k levels | 1-way ANOVA |
| | Several qualitative | Multi-way ANOVA |
| | 1 or several quantitative | Linear regression |
| | Mixture of qualitative & quantitative | ANCOVA |
| Qualitative | 1 or several quantitative or qualitative | Logistic regression |
| Counts with many zeros | 1 or several quantitative or qualitative | Log-linear regression |

Link: which modeling method to choose?



*Consult an expert to make sure you're modeling the data correctly.*

```
import numpy as np, pandas as pd, statsmodels.api as sm
```

```
Xtr_sm = sm.add_constant(X_train)
logit = sm.Logit(y_train, Xtr_sm)
```

*Model specification*

```
res_ridge = logit.fit_regularized(alpha=1.0, L1_wt=0.0)
```

*Estimation*

# Then: inspect everything!

```python
# === Coef table ===
terms  = res_ridge.model.exog_names
beta   = np.asarray(res_ridge.params)
coef_tbl = pd.DataFrame({"term": terms, "coef": beta, "odds_ratio": np.exp(beta)}).round(3)
print("=== Coefficients (ridge) ===")
print(coef_tbl.to_string(index=False))

# === Helper: unpenalized log-likelihood at given beta ===
def loglik_logit(beta, X, y):
    z = X @ beta
    # numerically stable log-sigmoid pieces
    # log(sigmoid(z)) = -softplus(-z); log(1-sigmoid(z)) = -softplus(z)
    ll = (y * (-np.log1p(np.exp(-z))) + (1 - y) * (-np.log1p(np.exp(z)))).sum()
    return float(ll)

# Train diagnostics (unpenalized LL evaluated at penalized beta)
n, k = Xtr_sm.shape
ll_full = loglik_logit(beta, Xtr_sm, y_train)

# Null (intercept-only) model log-likelihood
p0 = y_train.mean()
ll_null = (y_train*np.log(p0) + (1-y_train)*np.log(1-p0)).sum()

deviance   = -2.0 * ll_full
mcfadden_r2 = 1.0 - (ll_full / ll_null)
aic = 2*k - 2*ll_full     # using unpenalized LL
bic = np.log(n)*k - 2*ll_full

print("\n=== Train diagnostics (evaluated at penalized estimates) ===")
print(f"Log-likelihood (full): {ll_full:.3f}")
print(f"Log-likelihood (null): {ll_null:.3f}")
print(f"Deviance:              {deviance:.3f}")
print(f"McFadden R^2:          {mcfadden_r2:.3f}")
print(f"AIC (k={k}):           {aic:.3f}")
print(f"BIC (k={k}):           {bic:.3f}")
print("Note: AIC/BIC use the unpenalized log-likelihood at the penalized β̂; exact penalty-aware ICs require effective df.")

# === Test-set metrics ===
Xte_sm = sm.add_constant(X_test, has_constant='add')
proba  = res_ridge.predict(Xte_sm)
pred   = (proba >= 0.5).astype(int)

auc   = roc_auc_score(y_test, proba)
lloss = log_loss(y_test, proba, labels=[0,1])
acc   = accuracy_score(y_test, pred)
brier = brier_score_loss(y_test, proba)
tn, fp, fn, tp = confusion_matrix(y_test, pred).ravel()
sens = tp / (tp + fn) if (tp+fn) else np.nan   # recall, TPR
spec = tn / (tn + fp) if (tn+fp) else np.nan   # TNR

print("\n=== Test metrics ===")
print(f"AUC:      {auc:.3f}")
print(f"LogLoss: {lloss:.3f}")
print(f"Acc:      {acc:.3f}")
print(f"Brier:    {brier:.3f}")
print(f"Confusion: TP={tp}, FP={fp}, TN={tn}, FN={fn}")
print(f"Sensitivity (TPR): {sens:.3f}")
print(f"Specificity (TNR): {spec:.3f}")
```

```
=== Coefficients (ridge) ===
 term  coef  odds_ratio
const 0.000       1.000
   x1 0.983       2.673
   x2 2.029       7.606

=== Train diagnostics (evaluated at penalized estimates) ===
Log-likelihood (full): -3.337
Log-likelihood (null): -16.636
Deviance:               6.674
McFadden R^2:           0.799
AIC (k=3):             12.674
BIC (k=3):             16.208
```

```
=== Test metrics ===
AUC:      1.000
LogLoss: 0.250
Acc:      0.875
Brier:    0.089
Confusion: TP=8, FP=2, TN=6, FN=0
Sensitivity (TPR): 1.000
Specificity (TNR): 0.750
```

# Machine Learning

```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

pipe = Pipeline([
    ("scaler", StandardScaler()),
    ("logreg", LogisticRegression())
])
pipe.fit(X_train, y_train)

pred_probs = pipe.predict_proba(X_test)[:,1]
pred_labels  = (proba_ml >= 0.5).astype(int)

print("AUC:", round(roc_auc_score(y_test, pred_probs), 3))
print("Accuracy:", round(accuracy_score(y_test, pred_labels), 3))
```

*Model specification*

```
pipe = Pipeline([
    ("scaler", StandardScaler()),
    ("logreg", LogisticRegression())
])
```

*Fit the weights*

```
pipe.fit(X_train, y_train)
```

*Metrics*

```
AUC: 1.0
Accuracy: 0.812
```

# Deep Learning

```python
import torch
from torch import nn
import numpy as np
from sklearn.metrics import roc_auc_score, log_loss, accuracy_score

# Assume X_train, y_train, X_test, y_test are NumPy arrays
torch.manual_seed(0)

# Tensors
Xtr = torch.tensor(X_train, dtype=torch.float32)
ytr = torch.tensor(y_train, dtype=torch.float32).view(-1, 1)
Xte = torch.tensor(X_test,  dtype=torch.float32)
yte = torch.tensor(y_test,  dtype=torch.float32).view(-1, 1)

# Model: linear -> sigmoid (returns probabilities in (0,1))
model = nn.Sequential(
    nn.Linear(Xtr.shape[1], 1),
    nn.Sigmoid()
)

# Loss takes probabilities because we already applied Sigmoid
loss_fn = nn.BCELoss()
opt = torch.optim.SGD(model.parameters(), lr=0.1)

# Train (full-batch for simplicity)
for _ in range(200):
    p = model(Xtr)                  # predicted probabilities
    loss = loss_fn(p, ytr)
    opt.zero_grad()
    loss.backward()
    opt.step()

# Evaluate
with torch.no_grad():
    pred_probs = model(Xte).numpy().ravel()   # probabilities directly
pred_labels = (pred_probs >= 0.5).astype(int)

print("AUC:",      round(roc_auc_score(y_test, pred_probs), 3))
print("Accuracy:", round(accuracy_score(y_test, pred_labels), 3))
```

## *Network architecture*

```python
# Model: linear -> sigmoid (returns probabilities in (0,1))
model = nn.Sequential(
    nn.Linear(Xtr.shape[1], 1),
    nn.Sigmoid()
)
```

## *Fit the weights (with gradient descent)*

```python
for _ in range(200):
    p = model(Xtr)                  # predicted probabilities
    loss = loss_fn(p, ytr)
    opt.zero_grad()
    loss.backward()
    opt.step()
```

## *Metrics*

```
AUC: 1.0
Accuracy: 0.812
```

# Summary: Three Cultures of Data

- **Statistics**
  - “So, I’m about to run an experiment, what’s the right design for that experiment and one right way to analyze the data?”
  - “...actually, I already ran the experiment, I’m sorry, please help.”
- **Machine Learning**
  - “So, I found this dataset and want to use it to make a predictor from hand-crafted feature values to labels that will work on new unseen data…”
- **Deep Learning**
  - Dream up a differentiable compute graph with predictions as outputs
  - Put the raw data in the box
  - Stochastic gradient descent goes brrrrrrr…


Fin