Machine Learning and Prediction

FRM Part I

1 A story: the loan officer's notebook

- Past loans with known outcomes are the training examples
- The model learns a pattern linking facts to outcomes
- It then predicts the outcome for a brand new borrower
- Test: does the pattern hold on cases it has never seen?

IN THE LESSON

- Picture a loan officer who keeps a notebook.
- For every past loan she wrote down a few facts about the borrower, and later whether the loan went bad and how much was lost.
- After years, she starts to see patterns.
- A borrower with heavy debts and a short job history is more likely to default.
- A loan with a small deposit loses more when it fails.
- Machine learning is this same idea done with a computer.
- It learns patterns from past examples where the answer is known, then predicts the answer for a new case.
- The big question is always the same: will the pattern still work on cases it has not seen?

2 Linear regression forecasts an amount

- Target is a continuous amount: a loss, or loss given default
- Y = b0 + b1·X1 + b2·X2 + …, fitted by least squares
- Forecast = put the new predictor values into the equation
- Each coefficient: change in Y per unit of that predictor

IN THE LESSON

- Start with the workhorse.
- A linear regression forecasts a continuous quantity, a number that can take any value in a range, such as next quarter's credit loss or the loss given default on a loan.
- The model writes the target as an intercept plus a coefficient times each predictor.
- Least squares chooses the intercept and coefficients that make the squared gaps between actual and fitted values as small as possible.
- To forecast, you plug the new predictor values into the fitted equation and read off a single answer, called a point forecast.
- Each coefficient says how much the forecast moves when that predictor rises by one unit and the others stay fixed.

3 Logistic regression forecasts an event

ln(p ÷ (1 − p)) = z = b0 + b1·X1 + …
p = 1 ÷ (1 + e^(−z))

Log-odds, z	z: the linear score, the log-odds
Probability, p	p: probability, always in (0, 1)
Odds per unit	Odds multiply by e^b per unit

IN THE LESSON

- Many risk questions are yes or no: will the borrower default or not?
- A straight line is a poor fit here, because it can produce answers below zero or above one.
- Logistic regression fixes this.
- It builds a linear score, called z, from the predictors, and treats that score as the log odds of the event.
- Odds are the probability of the event divided by the probability of no event.
- Turning the score back into a probability gives one divided by one plus e to the minus z, which always lies between zero and one.
- A coefficient acts on the odds: for each extra unit of the predictor, the odds are multiplied by e raised to the coefficient.

4 Example 1: from score to probability

Illustration: z = −4 + 0.5 × leverage. Borrower has leverage 6

Score z	−4 + 0.5 × 6	**−1**
Odds	e^(−1)	**0.368**
Probability of default	1 ÷ (1 + e^1)	**26.9%**
Odds per extra unit	e^0.5	**1.65×**

IN THE LESSON

- Let's use made up numbers.
- Suppose a model says the score equals minus four plus a half times the borrower's leverage, and a borrower has leverage of six.
- The score is minus four plus three, which is minus one.
- The odds are e to the minus one, about zero point three six eight, meaning roughly one default for every two point seven no defaults.
- The probability is one divided by one plus e to the plus one, which is about twenty six point nine percent.
- Notice the score is not a probability, and neither are the odds.
- Finally, each extra unit of leverage multiplies the odds by e to the half, about one point six five.

5 Example 2: combining the two forecasts

Illustration: LGD = 0.10 + 0.50 × loan-to-value; EL = PD × LGD × exposure

LGD (linear regression)	0.10 + 0.50 × 0.60	**40%**
PD (logistic regression)	fitted probability	**3%**
Exposure	outstanding amount	**$2,000,000**
Expected loss	0.03 × 0.40 × 2,000,000	**$24,000**

IN THE LESSON

- Now see how the two regressions work together.
- Expected loss equals the probability of default times the loss given default times the exposure.
- A logistic regression supplies the probability of default.
- A linear regression supplies the loss given default, because that is an amount, not an event.
- Suppose a linear model says the loss given default equals zero point one plus a half times the loan to value ratio, and our loan has a ratio of zero point six.
- That gives zero point four, forty percent.
- The logistic model gives a default probability of three percent.
- The exposure is two million dollars.
- Multiply: zero point zero three times zero point four times two million is twenty four thousand dollars of expected loss.

6 Judging a logistic model on unseen cases

- Judge on a holdout or cross-validated sample
- Confusion matrix at a cutoff: 4 kinds of case
- ROC: true vs false positive rate as cutoff moves
- AUC: 0.5 = chance, 1.0 = perfect ranking

IN THE LESSON

- How do we know whether a logistic model is any good?
- The key rule is to judge it on cases it has not seen, using a holdout sample kept aside, or cross validation, which rotates the holdout.
- A model always looks good on the data it was fitted to.
- At a chosen cutoff, say flag everyone above ten percent, we count a confusion matrix: defaults caught, defaults missed, false alarms and correct clears.
- Move the cutoff from one down to zero and plot the true positive rate against the false positive rate.
- That curve is the receiver operating characteristic, or R O C.
- The area under it, the A U C, summarizes ranking power in one number.

7 Example 3: AUC by counting pairs

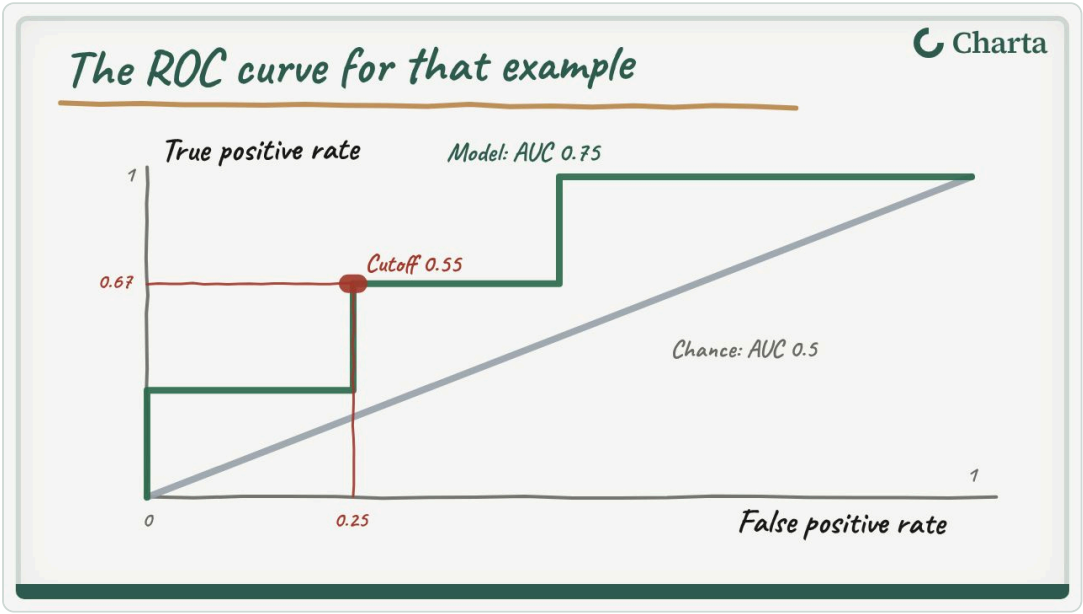
Illustration: defaulters 0.8, 0.6, 0.3. Non-defaulters 0.7, 0.5, 0.2, 0.1

Pairs to compare	3 × 4	**12**
Defaulter 0.8 ranks higher	beats 4 of 4	**4 wins**
Defaulters 0.6 and 0.3	3 + 2	**5 wins**
AUC	9 ÷ 12	**0.75**

IN THE LESSON

- There is a friendly way to read the A U C.
- It is the chance that a randomly chosen defaulter gets a higher score than a randomly chosen non defaulter.
- Take three defaulters scored zero point eight, zero point six and zero point three, and four non defaulters scored zero point seven, zero point five, zero point two and zero point one.
- There are three times four, twelve pairs to compare.
- The defaulter at zero point eight beats all four.
- The defaulters at zero point six and zero point three win three and two pairs, five between them.
- That is nine wins out of twelve, so the A U C is zero point seven five.
- A coin flip would score a half, so this model ranks well.

8 The ROC curve for that example



IN THE LESSON

- Here is the R O C curve for those seven scores.
- Along the bottom is the false positive rate, the share of non defaulters we wrongly flag.
- Up the side is the true positive rate, the share of defaulters we catch.
- Start at the origin, with a cutoff so high that nobody is flagged.
- As we lower the cutoff past each score, the curve steps up when we meet a defaulter and steps right when we meet a non defaulter.
- The grey diagonal is pure chance, an A U C of one half.
- Our curve bows above it.
- The marked point is a cutoff of zero point five five: it catches two of three defaulters at the price of flagging one of four good borrowers.

9 Why accuracy can mislead

Illustration: 1,000 loans, 20 defaults. Model always says 'no default'

Share of loans that default	20 ÷ 1,000	**2%**
Accuracy of the lazy model	980 ÷ 1,000	**98%**
Defaults caught (recall)	0 ÷ 20	**0%**
Is the model useful?	accuracy hides it	**No**

IN THE LESSON

- Now a warning about accuracy, the share of cases classified correctly.
- Imagine a portfolio of one thousand loans in which only twenty, or two percent, default.
- A lazy model that predicts no default for everyone is right on nine hundred and eighty loans, so its accuracy is ninety eight percent.
- That sounds superb.
- But it caught zero of the twenty defaults, a recall of zero percent, which is useless for a risk team.
- When one class is rare, accuracy mostly rewards guessing the common class.
- That is why risk work leans on recall, precision, the A U C and the cost of each type of error, and not on accuracy alone.

10 Overfitting, cutoffs and calibration

- Big in-sample vs out-of-sample gap = overfitting
- AUC averages all cutoffs; the business uses one region
- Ranking asks: are risky cases scored above safe ones?
- Calibration: do predicted rates match observed rates?

IN THE LESSON

- Three more checks.
- First, overfitting.
- If performance is much better in sample than out of sample, the model has memorized noise, and the gap is the warning sign.
- Second, the A U C averages over every cutoff, but the business uses only one region.
- Two models can have the same A U C and differ greatly where the lender actually draws the line, so look at the part of the curve that matters.
- Third, calibration.
- Ranking asks whether risky borrowers score higher than safe ones.
- Calibration asks whether a predicted probability of five percent really means about five in a hundred default.
- A model can rank well and still be badly calibrated, so check both.

11 Encoding categories as numbers

Match the encoding to the kind of category

Method	Best used for
One-hot (0/1 columns)	Nominal: sector
Ordinal (1, 2, 3)	Ordered: rating buckets
Target encoding	Many levels: postcode
Frequency / binary	Compact alternatives

IN THE LESSON

- Models need numbers, so a category such as sector or country must be encoded.
- One hot encoding makes one zero or one column for each category.
- It suits nominal categories with no natural order.
- Ordinal encoding assigns one, two, three, and suits genuinely ordered categories such as rating buckets.
- On a nominal variable it would invent a false order and spacing, as if country three were three times country one.
- When a category has very many levels, target encoding replaces each level with the average outcome seen for it.
- Frequency encoding and binary encoding are compact alternatives.
- The danger of target encoding is leakage, which we meet on the next slide.

12 Example 4: dummies and target encoding

Illustration: sector = Banks / Utilities / Tech. Reference = Utilities

Dummy columns kept	3 − 1	**2**
Tech vs Utilities, odds	e^0.9	**2.46×**
Tech raw default rate	2 defaults ÷ 4 loans	**50%**
Smoothed (overall 2%, m 16)	(4×50% + 16×2%) ÷ 20	**11.6%**

IN THE LESSON

- Take sector with three categories: banks, utilities and tech.
- In a regression with an intercept, we keep only k minus one columns, so two.
- The dropped one, utilities, is the reference.
- Keeping all three would make the columns add up to the intercept, the dummy variable trap, a perfect multicollinearity.
- Each dummy coefficient is measured against the reference.
- If the tech coefficient in a logistic model is zero point nine, tech odds are e to the zero point nine, about two point four six times utilities odds.
- Now target encoding: tech has four loans with two defaults, a raw rate of fifty percent.
- Smoothing toward the overall two percent, with a weight of sixteen, gives eleven point six percent.
- Smoothing pulls a thin level toward the overall rate, and computing it out of fold stops a loan's own outcome leaking into its own feature.

13 Regularization: a penalty on big coefficients

- Many or correlated predictors: unstable coefficients
- Penalty trades a little bias for less variance
- Lambda = strength: zero is ordinary least squares
- Standardize first; intercept not penalized
- Choose lambda by cross-validation

IN THE LESSON

- When a model has many predictors, or predictors that are highly correlated, ordinary least squares can chase noise.
- The coefficients become large and unstable, and the model overfits.
- Regularization adds a penalty on coefficient size to the quantity being minimized.
- The model accepts a little bias, because it is no longer the best possible fit to the training data, in return for a large drop in variance, so it predicts better on new data.
- The penalty has a strength, called lambda.
- When lambda is zero we are back to ordinary least squares.
- As lambda grows, coefficients shrink toward zero.
- Standardize the predictors first, since the penalty is not fair across different units.
- The intercept is normally left unpenalized.

14 Ridge and LASSO side by side

Ridge: RSS + λ × Σ β²
LASSO: RSS + λ × Σ |β|

Squared errors, RSS	Squared fitting errors
Penalty strength, lambda	Strength; 0 gives plain OLS
Size of the coefficients	Squares or absolutes of β
What the fit minimizes	Minimized to choose the fit

IN THE LESSON

- Both methods minimize the squared errors, written R S S, plus a penalty.
- Ridge adds lambda times the sum of the squared coefficients, called the L two penalty.
- LASSO adds lambda times the sum of the absolute values of the coefficients, the L one penalty.
- Because of the geometry of that absolute value, LASSO can push some coefficients exactly to zero, so it performs variable selection.
- Ridge shrinks every coefficient but rarely to zero.
- With highly correlated predictors, ridge tends to share the weight among them, whereas LASSO tends to keep one and drop the rest.
- The strength lambda is chosen by cross validation, trying many values and keeping the one that predicts best on held out data.

15 Example 5: shrinking two coefficients

Illustration: orthonormal predictors (uncorrelated, unit scale); λ = 1

Method	Strong | weak coefficient
Ordinary least squares	2.0 | 0.4
Ridge: b ÷ (1 + λ)	1.0 | 0.2
LASSO: b − λ÷2, floor 0	1.5 | 0.0

IN THE LESSON

- Here is the difference in numbers, using the simplest case of orthonormal predictors, meaning uncorrelated and on a unit scale, with the penalty written as lambda times the penalty term, and lambda equal to one.
- Suppose ordinary least squares gives a strong coefficient of two and a weak one of zero point four.
- Ridge divides each by one plus lambda, which is two.
- So the coefficients become one and zero point two.
- Both are shrunk, and neither reaches zero.
- LASSO subtracts half of lambda, a half, from the size of each coefficient, and stops at zero.
- The strong one falls from two to one and a half.
- The weak one, zero point four, is smaller than the half, so it is set exactly to zero, and that predictor is dropped.

16 Decision trees: questions, one at a time

- Yes/no questions, grown top-down and greedily
- Try every feature and threshold; keep the best split
- Impurity: Gini/entropy, or squared errors
- Children's impurity weighted by size
- Stop at a depth/leaf limit, or prune a full tree

IN THE LESSON

- A decision tree predicts by asking a chain of yes or no questions about the features, such as is leverage above four.
- It is grown from the top down and greedily, by recursive binary splitting.
- All observations start together in the root.
- At each node the algorithm tries every feature and every threshold and picks the split that most reduces impurity, which measures how mixed the classes are.
- For classification, impurity is Gini or entropy.
- For regression it is the sum of squared errors.
- The impurity of the two children is weighted by their sizes.
- Growing stops at a limit such as maximum depth or a minimum leaf size, or the tree is grown fully and then pruned back.

17 Example 6: one split, by Gini impurity

Illustration: Gini = 1 − (p_1^2 + p_2^2). Split 100 loans on leverage

Root: 40 of 100 default	1 − (0.4² + 0.6²)	**0.480**
Left: 30 of 40 default	1 − (0.75² + 0.25²)	**0.375**
Right: 10 of 60 default	1 − (0.167² + 0.833²)	**0.278**
Weighted children	0.4×0.375 + 0.6×0.278	**0.317**
Impurity reduction	0.480 − 0.317	**0.163**

IN THE LESSON

- Gini impurity is one minus the sum of the squared class shares.
- Start with a root of one hundred loans, forty of which default.
- The shares are point four and point six, so Gini is one minus point one six plus point three six, which is point four eight.
- Now split on leverage.
- The left node has forty loans with thirty defaults, shares of point seven five and point two five, giving a Gini of point three seven five.
- The right node has sixty loans with ten defaults, giving about point two seven eight.
- Weight by size: point four times point three seven five plus point six times point two seven eight is about point three one seven.
- The split cut impurity by point one six three, so it is a good one.

18 Reading a tree, and its limits

- Follow the path from the root: every leaf is a rule
- Classification leaf: majority class. Regression leaf: mean
- Strength: easy to read and explain
- Weakness: deep trees overfit and are unstable
- Regression tree: no predictions beyond training range

IN THE LESSON

- To read a tree, simply follow a path from the root.
- If leverage is above four, go right.
- If the loan to value ratio is above eighty percent, go right again.
- You arrive at a leaf, and each leaf is a rule.
- In classification, the leaf predicts the majority class, with the share of that class as a rough probability.
- In regression, it predicts the mean of the training values in the leaf.
- This readability is the tree's great strength.
- The weaknesses are just as important.
- Deep trees overfit, memorizing noise in tiny leaves.
- Small changes in the data can change the tree.
- And a regression tree can never predict outside the range of the values in its training leaves.

19 Ensembles: the variance of an average

Var(average) = ρ × σ² + (1 − ρ) × σ² ÷ B

Variance of the average	Variance of the averaged forecast
Correlation, rho	Correlation between learners
Variance of one learner	Variance of one learner
Number of learners, B	How many learners are averaged

IN THE LESSON

- An ensemble combines many learners so the group predicts better than any one.
- Bagging, short for bootstrap aggregation, trains the same type of learner on many bootstrap resamples, drawn from the data with replacement.
- It averages the outputs for regression, or takes a majority vote for classification.
- This mainly cuts variance, but how much depends on how correlated the learners are.
- The variance of an average of B learners, each with variance sigma squared and pairwise correlation rho, is rho times sigma squared, plus one minus rho times sigma squared divided by B.
- Adding learners shrinks only the second term.
- The first term, from correlation, never goes away.

20 Example 7: how many learners help?

Illustration: each learner has variance 1.0; correlation ρ = 0.3

One learner	B = 1	**1.00**
Ten learners	0.3 + 0.7 ÷ 10	**0.37**
One hundred learners	0.3 + 0.7 ÷ 100	**0.307**
Floor as B grows	ρ × σ²	**0.30**
Forest: ρ 0.1, 100 learners	0.1 + 0.9 ÷ 100	**0.109**

IN THE LESSON

- Let each learner have a variance of one and a pairwise correlation of zero point three.
- With one learner the variance is one.
- With ten learners it is zero point three plus zero point seven divided by ten, which is zero point three seven.
- With one hundred learners it is zero point three plus zero point seven divided by one hundred, zero point three zero seven.
- Even with endless learners the variance cannot fall below rho times sigma squared, which is zero point three.
- The cure is to lower the correlation.
- A random forest does this by letting each split look at only a random subset of the features.
- With a correlation of zero point one and one hundred learners, the variance is only zero point one zero nine.

21 Bagging, forests, boosting and stacking

- Bagging: average resampled learners; cuts variance
- Random forest: bagged trees with random feature subsets
- Boosting: sequential, fixes earlier errors; cuts bias
- Stacking: a second model combines predictions
- More learners do not cure correlated errors

IN THE LESSON

- Let's compare the families.
- Bagging averages learners trained independently on resamples, and it mainly reduces variance.
- A random forest is bagged trees with the extra twist of random feature subsets, which lowers the correlation between the trees.
- Boosting is different: it is sequential.
- Each new learner concentrates on what the earlier ones got wrong, either by giving the missed observations more weight or by fitting the leftover residuals.
- Each is added with a small learning rate, and boosting mainly reduces bias.
- Stacking trains a second model to combine the predictions of the first level learners.
- A common trap is to believe that more learners cure correlated errors.
- They do not.

22 Example 8: one round of boosting

Illustration: true value 10; first learner predicts 6; learning rate 0.5

Residual after round one	10 − 6	**4**
Learner two predicts	its guess of the residual	**3**
Updated prediction	6 + 0.5 × 3	**7.5**
New residual	10 − 7.5	**2.5**

IN THE LESSON

- Here is boosting on a single borrower, with made up numbers.
- The true loss is ten.
- The first learner predicts six, so the residual, the part still unexplained, is ten minus six, which is four.
- The second learner is trained to predict that residual, and being a weak learner it predicts three.
- We do not add the whole three.
- With a learning rate of a half, the updated prediction is six plus a half times three, which is seven point five.
- The new residual is ten minus seven point five, or two point five.
- The error has fallen from four to two point five.
- More rounds keep chipping away, and the small learning rate stops any single learner from dominating.

23 Example 9: K nearest neighbors

Illustration: new case at (3, 3). Others: E (1,1) no-default 2.83; D (6,3) no-default 3.00

A (2,3) default	$\sqrt{(1^2 + 0^2)}$	**1.00**
C (4,4) no default	$\sqrt{(1^2 + 1^2)}$	**1.41**
B (3,5) default	$\sqrt{(0^2 + 2^2)}$	**2.00**
K = 3 vote	2 default, 1 not	**Default**
K = 5 vote	2 default, 3 not	**No default**

IN THE LESSON

- K nearest neighbors is lazy: nothing is trained.
- To classify a new case, measure its distance to every training case, usually Euclidean, find the K closest, and take the majority class.
- A new borrower sits at three, three on two standardized features.
- Training point A at two, three, a defaulter, is a distance of one.
- Point C at four, four, a non defaulter, is the square root of two, about one point four one.
- Point B at three, five, a defaulter, is two.
- Points E and D, both non defaulters, are further away, at two point eight three and three.
- With K equal to three, the neighbors are A, C and B: two defaults to one, so we predict default.
- With K equal to five, three non defaulters outvote two defaulters, so the prediction flips.
- Standardize the features first, or large units dominate the distance.
- A small K is noisy, a large K gives a smoother boundary, and K is chosen by cross validation.

24 Example 10: the support vector machine

Illustration: hyperplane $3 \cdot x_1 + 4 \cdot x_2 - 10 = 0$. Support vectors score ±1

Length of the weights	$\sqrt{(3^2 + 4^2)}$	**5**
Score at point (4, 3)	$3 \times 4 + 4 \times 3 - 10$	**14**
Distance to hyperplane	$\lvert 14 \rvert \div 5$	**2.8**
Margin	$2 \div 5$	**0.4**

IN THE LESSON

- A support vector machine seeks the hyperplane that separates the two classes with the widest margin.
- Only the points on or inside the margin, the support vectors, determine it.
- Suppose the weights are three and four and the intercept is minus ten.
- The length of the weight vector is the square root of nine plus sixteen, which is five.
- A borrower at four, three scores three times four plus four times three minus ten, which is fourteen.
- The distance to the hyperplane is the absolute score divided by the length, fourteen over five, two point eight.
- With the usual scaling, where support vectors score one, each sits one fifth, zero point two, from the plane, so the margin across both sides is two divided by five, zero point four.
- That borrower is far outside the margin.
- A soft margin allows some violations at a cost C, and a kernel handles curved boundaries.

25 Neural networks: layers of neurons

- Layers: input, one or more hidden, output
- Neuron: weighted sum + bias, then an activation function
- Sigmoid, tanh or ReLU supplies the non-linearity
- Weights start small and random; loss from a forward pass
- Backpropagation: chain rule gives each weight's gradient

IN THE LESSON

- A feed forward neural network is arranged in layers: an input layer, one or more hidden layers, and an output layer.
- Each neuron takes a weighted sum of the previous layer's outputs and adds a bias.
- It then passes the result through an activation function, such as the sigmoid, the hyperbolic tangent or the rectified linear unit.
- The activation is essential.
- Without it, stacked layers collapse into a single linear model, so there would be no point in having hidden layers.
- The weights are not found by a formula.
- They start as small random numbers.
- A forward pass produces predictions and a loss, such as squared error or cross entropy.
- Back propagation then uses the chain rule to find how the loss changes with each weight.

26 Example 11: one neuron and the parameters

Illustration: inputs 2 and 1; weights 0.5 and −1; bias 0.5

Weighted sum z	0.5×2 + (−1)×1 + 0.5	**0.5**
Sigmoid output	1 ÷ (1 + e^(−0.5))	**0.622**
ReLU output	max(0, 0.5)	**0.5**
Parameters, 3-4-1 network	(3×4+4) + (4×1+1)	**21**

IN THE LESSON

- Let's run one neuron.
- Two inputs are two and one.
- The weights are a half and minus one, and the bias is a half.
- The weighted sum is a half times two, minus one times one, plus a half: one, minus one, plus a half, which is a half.
- Pass that through a sigmoid and the output is about zero point six two two.
- Through a rectified linear unit, which keeps positive values, the output stays at a half.
- Finally, count the parameters in a network with three inputs, one hidden layer of four neurons and one output.
- The hidden layer has three times four weights plus four biases, sixteen.
- The output layer has four weights plus one bias, five.
- In all, twenty one parameters.

27 Example 12: gradient descent on a weight

Illustration: new w = w − rate × gradient. Rate 0.10; start w = 0.80

Gradient 0.5: step size	0.10 × 0.5	**0.05**
New weight	0.80 − 0.05	**0.75**
Next gradient 0.3	0.75 − 0.10 × 0.3	**0.72**
Gradient reaches zero	the weight stops moving	**Local minimum**

IN THE LESSON

- Now the update that finds the weights.
- Gradient descent moves each weight a small step against its gradient, the slope of the loss.
- The new weight equals the old weight minus the learning rate times the gradient.
- Suppose a weight is zero point eight, its gradient is zero point five, and the learning rate is zero point one.
- The step is zero point one times zero point five, zero point zero five, so the new weight is zero point seven five.
- After another pass the gradient has shrunk to zero point three, so the weight falls to zero point seven two.
- When the gradient reaches zero the weight stops moving, at a flat point of the loss, ideally a minimum.
- Repeating over many epochs, with early stopping, dropout or regularization, controls overfitting.

28 Example 13: a confusion matrix, logistic model

Illustration: logistic. TP 12, FN 8, FP 60, TN 920 (1,000 loans)

Accuracy	(12 + 920) ÷ 1,000	**93.2%**
Recall	12 ÷ (12 + 8)	**60%**
Precision	12 ÷ (12 + 60)	**16.7%**
Specificity	920 ÷ (920 + 60)	**93.9%**
F1	2×12 ÷ (24 + 60 + 8)	**26.1%**

IN THE LESSON

- Now we compare models, starting with the logistic regression, using a common holdout of one thousand loans, twenty of which default.
- At the chosen cutoff it catches twelve defaults, the true positives, and misses eight, the false negatives.
- It raises sixty false alarms and clears nine hundred and twenty loans correctly.
- Accuracy is the true positives plus the true negatives, over the total: nine hundred and thirty two over a thousand, ninety three point two percent.
- Recall is twelve over twenty, sixty percent.
- Precision is twelve over seventy two, sixteen point seven percent.
- Specificity is nine twenty over nine eighty, ninety three point nine percent.
- And F one is two times twelve, over twenty four plus sixty plus eight: twenty six point one percent.

29 Logistic regression against the neural network

Same 1,000 loans, 20 defaults. Neural net: TP 14, FN 6, FP 90, TN 890

Measure	Logistic | Neural net
Accuracy	93.2% | 90.4%
Recall	60% | 70%
Precision	16.7% | 13.5%
Specificity	93.9% | 90.8%
F1	26.1% | 22.6%

IN THE LESSON

- Now the neural network, on the same thousand loans and the same cutoff rule.
- It catches fourteen defaults, misses six, raises ninety false alarms and clears eight hundred and ninety.
- Compute the same measures and put them side by side.
- The logistic model has the higher accuracy, ninety three point two against ninety point four.
- It also has higher precision, higher specificity and a higher F one.
- But the neural network catches seventy percent of the defaults against sixty.
- A reader who looks only at accuracy would pick the logistic model, yet the neural network finds two more defaults out of twenty.
- Each matrix describes one cutoff only, and raising recall normally lowers precision.
- So which is better depends on what the errors cost.

30 Example 14: choosing by cost

Illustration, $ thousands: missed default 100; false alarm 5

Logistic: 8 misses, 60 alarms	8×100 + 60×5	**1,100**
Neural net: 6 misses, 90 alarms	6×100 + 90×5	**1,050**
Cheaper model	lower total cost	**Neural net**
Higher accuracy	not the same thing	**Logistic**

IN THE LESSON

- Finally, weigh the errors by what they cost.
- Say a missed default costs one hundred, in thousands of dollars, and a false alarm costs five, the cost of an unnecessary review.
- The logistic model made eight misses and sixty false alarms: eight times one hundred plus sixty times five, which is eight hundred plus three hundred, eleven hundred.
- The neural network made six misses and ninety false alarms: six hundred plus four hundred and fifty, a total of ten fifty.
- So the neural network is cheaper by fifty, even though its accuracy is lower.
- The ranking could reverse if a false alarm cost much more.
- The usual mistake is to prefer the higher accuracy under imbalanced classes.

31 Four mistakes to avoid

- Odds or log-odds are not a probability: convert first
- Holdout only; accuracy misleads if classes are rare
- No leakage; drop one dummy; no ordinal codes on nominal
- More learners, bigger models: no cure; curb overfitting

IN THE LESSON

- Mistake one: reading the log odds or the odds as a probability.
- Always convert with one over one plus e to the minus the score.
- Mistake two: judging a model on its training data, or trusting accuracy when the classes are imbalanced.
- Use a holdout and look at recall, precision and the A U C.
- Mistake three: leaking information, as with target encoding computed on the same data, or falling into the dummy variable trap, or using ordinal codes on nominal categories.
- Mistake four: assuming more learners cure correlated errors, or that a bigger model is always better.
- Penalties, pruning, cross validation and early stopping exist to control overfitting.

32 Recap: the main ideas

- Linear: amounts. Logistic: probability. Both feed EL
- Unseen data: confusion matrix, ROC, AUC, calibration
- Encode with care; ridge shrinks, LASSO selects
- Trees, ensembles, KNN, SVM, neural networks
- Compare on recall, precision and cost of errors

IN THE LESSON

- Let's pull it together.
- Linear regression forecasts an amount, and logistic regression forecasts the probability of an event; together they give expected loss.
- A model must be judged on unseen data, with the confusion matrix, the R O C curve and the A U C, and calibration is a separate question.
- Categories are encoded by one hot, ordinal or target methods, with care.
- Ridge and LASSO penalize coefficient size, and LASSO can select variables.
- Trees split greedily on impurity, ensembles reduce variance or bias, nearest neighbors and support vector machines classify by distance and margin, and neural networks learn weights by gradient descent.
- Compare models with a confusion matrix and the cost of errors.

33 Try three quick questions

1. Logistic score z = 2. Probability of default?
   Answer: 1 ÷ (1 + e^−2) = 88.1%.
2. Flags 40: 30 defaults, 10 false alarms. Misses 15 defaults. Recall, precision?
   Answer: Recall = 30 ÷ 45 = 66.7%. Precision = 30 ÷ 40 = 75%.
3. 50 learners, variance 1, ρ = 0.5. Variance of the average?
   Answer: 0.5 + 0.5 ÷ 50 = 0.51.

IN THE LESSON

- Now try three quick questions yourself.
- Pause after each one.
- Question one.
- A logistic model gives a score of two.
- What is the probability of default?
- One over one plus e to the minus two, about eighty eight percent.
- Question two.
- A model flags forty loans, of which thirty are real defaults and ten are false alarms, and it misses fifteen other defaults.
- What are its recall and precision?
- Recall is thirty over forty five, sixty six point seven percent.
- Precision is thirty over forty, seventy five percent.
- Question three.
- Fifty learners each have variance one and correlation a half.
- What is the variance of their average?
- A half plus a half over fifty, point five one.