CHARTA LEARNING · LESSON NOTES

Machine learning: overfitting, supervised and unsupervised methods

Quantitative Methods

1 A story: flash cards and a pile of buttons

- Flash cards: question plus known answer
- Mixed buttons: no answers, make your own piles
- Answers given: supervised learning
- No answers given: unsupervised learning

IN THE LESSON

- Picture two ways of learning.
- In the first, you study with flash cards.
- Each card has a question on the front and the answer on the back.
- You guess, flip the card, and correct yourself.
- After a few hundred cards you can answer questions you have never seen.
- In the second, someone hands you a jar of mixed buttons and says, sort them.
- Nobody tells you the categories.
- You make piles by size and colour because that is what the buttons suggest.
- The first is supervised learning, because the answers supervise you.
- The second is unsupervised learning, because nothing does.
- At Marrow Hill Capital, Priya's lab faces both kinds of task every week.

2 Supervised and unsupervised, side by side

Supervised: the answers are given

Features (inputs) → Label (known answer) → Model learns the link → Predicts new answers

Unsupervised: no answers

Features only → Algorithm hunts for structure → Groups or fewer features

IN THE LESSON

- Here are the two families as pictures.
- Start with the top row, supervised learning.
- Every training observation comes with inputs, which we call features, and a known answer, which we call the label or the target.
- The model studies the link between the two.
- Then, given the features of a new observation, it predicts the answer.
- If the answer is a number, such as next month's return, the task is regression.
- If the answer is a category, such as default or no default, the task is classification.
- Now the bottom row, unsupervised learning.
- There is no label.
- The algorithm only sees the inputs, and hunts for structure, either grouping similar observations or cutting the number of features.

3 Deep learning and reinforcement learning

- Deep learning: neural network with many hidden layers
- Good for images, speech and text
- Reinforcement: agent learns by trial and reward
- The test is a labelled target, not the data type

IN THE LESSON

- Two more names appear in the exam.
- Deep learning uses a neural network, which is a stack of layers.
- Each layer takes numbers from the layer before, multiplies them by weights, and passes the result through a bendy rule called an activation function.
- When there are many hidden layers between input and output, the network is called deep.
- Deep networks are good with images, speech and text.
- Reinforcement learning is different again.
- An agent tries actions, receives a reward or a penalty, and over many trials learns to collect more reward.
- Here is the trap.
- Deep learning is a kind of model, not a third family.
- It can be supervised or unsupervised.
- What decides is whether a labelled target exists.

4 Example: sorting four projects at Marrow Hill

Is there a labelled target? What type of task is it?

Project	Type
Returns from 40 ratios	Regression
Default or not	Classification
800 funds, no labels	Clustering
Car parks to sales	Deep, supervised

IN THE LESSON

- Let us practise on four projects from Priya's list.
- Project one predicts next month's return from forty ratios.
- There is a numerical label, so this is supervised regression.
- Project two predicts whether a bond defaults, using past defaults.
- The label is a category, so this is supervised classification.
- Project three groups eight hundred funds by how their returns behave, with no categories supplied.
- No label, so this is unsupervised clustering.
- Project four feeds satellite photos of car parks to a deep network, with each photo paired to the sales figure that followed.
- Images call for deep learning, but the sales figure is a label, so it is still supervised.
- A trading program that learns only from a reward would be reinforcement learning.

5 A story: the student who memorised the answers

- Overfitting: learned the noise, not the pattern
- Great on the training sample
- Poor on data it has not seen
- So always hold back a validation sample

IN THE LESSON

- Now to the biggest danger in this field.
- A student gets ten years of past exam papers and learns every answer by heart, word for word, including the odd typos.
- On a mock test made from those papers, they score nearly perfect.
- On the real exam, with new questions, they do badly.
- They learned the papers, not the subject.
- That is overfitting.
- An overfitted model has learned the noise that is specific to its training data, so it does very well in sample and poorly out of sample.
- To catch it, the analyst holds back some data.
- The training sample teaches the model.
- The validation sample, which the model has never seen, tests it honestly.

6 Training error and validation error

Chart: Error rate vs Model complexity (tree depth)

- Training error
- Validation error
- Underfit
- Best: depth 5
- Overfit

IN THE LESSON

- Here is the picture.
- Along the bottom is model complexity, here the depth of a decision tree.
- Up the side is the error rate.
- Start with the training error, the lower line.
- It falls and falls as the model gets more complex, because a flexible model can bend to fit anything, even noise.
- Now the validation error.
- At first it falls too, because the model is too simple and misses real patterns.
- That left side is underfitting.
- Then it bottoms out at a depth of five, the best model, marked by the dashed line.
- After that, it climbs again, because the model is memorising noise.
- That right side is overfitting.
- The widening gap between the two lines is the warning sign.

7 Where the error comes from

Total error = Bias error + Variance error + Base error

Bias error	Too simple: underfitting
Variance error	Too flexible: overfitting
Base error	Noise nobody can remove
Total error	Lowest at the best model

IN THE LESSON

- The curriculum splits a model's total error into three parts.
- Bias error comes from underfitting.
- The model is too simple, so it misses the real pattern in the same way whichever sample it sees.
- Variance error comes from overfitting.
- The model is so flexible that its answers swing with whichever sample it was given.
- Base error is the noise in the data itself, which no model can remove.
- So total error is the sum of the three.
- Here is an illustration with made-up numbers.
- Bias four, variance three and base two give a total of nine.
- Adding complexity lowers bias but raises variance, so the best model sits where their sum is smallest.

8 Diagnosing the model from two numbers

Compare training error with validation error

What you see	Diagnosis
High on both	Underfit
Low train, high validation	Overfit
Both low and close	Good fit

IN THE LESSON

- Because we cannot see bias and variance directly, we diagnose from two error rates, one on the training sample and one on the validation sample.
- Read the table by rows.
- If the error is high on both, the model has not even learned its own training data.
- That is underfitting, high bias, and more complexity would help.
- If the training error is low but the validation error is much higher, the model learned the sample and not the pattern.
- That is overfitting, high variance.
- The cure runs the other way: less complexity.
- If both are low and close together, the model generalises well.
- Look at the gap between the two, not at either number on its own.

9 Example: three trees for Priya

Error rates: training and validation (illustration)

Depth 2	train 18%, validation 20%	**Underfit**
Depth 5	train 10%, validation 12%	**Best**
Depth 12	train 2%, validation 19%	**Overfit**
Gap at depth 12	19 − 2	**17 points**

Choose by validation error, never by training error

IN THE LESSON

- Priya builds a classification tree to predict which stocks will outperform, and tries three maximum depths.
- Depth two has a training error of eighteen per cent and a validation error of twenty.
- High on both, and the gap is only two points.
- Underfitting.
- Depth five has ten per cent on training and twelve on validation.
- That is the lowest validation error of the three, so it is the best.
- Depth twelve is superb on training, two per cent, but nineteen per cent on validation.
- The gap is seventeen points.
- Ranked by training error, depth twelve would have won easily.
- Ranked on data it has not seen, it is barely better than the tree that underfits.
- Priya picks depth five.

10 Ways to deal with overfitting

- Reduce complexity: prune trees, cap depth
- Penalised regression shrinks weak coefficients
- Train, then validate, then a final test sample
- Short of data? Use k-fold cross-validation

IN THE LESSON

- So how do we fight it?
- There are two ideas.
- The first is to reduce complexity.
- For a tree, that means pruning it, cutting back branches that add little, or capping its depth.
- For a regression, it means penalised regression, which we meet shortly, where a penalty pushes weak coefficients towards zero.
- Dropping useless features helps too.
- The second idea is to estimate out-of-sample error honestly, so you know how far to go.
- The usual layout is three samples.
- The training sample fits the model.
- The validation sample tunes it, such as choosing the depth.
- A final test sample, touched once at the very end, gives an unbiased verdict.
- More data also helps, as noise averages out.

11 k-fold cross-validation

Five-fold cross-validation, k = 5

Split data into five folds → Hold one fold out → Train on the other four → Rotate, then average errors

IN THE LESSON

- When data are scarce, setting aside a big validation sample is costly.
- Cross-validation is the clever fix.
- Follow the boxes.
- First, split the data into k equal parts, called folds.
- Here k is five.
- Second, hold one fold out and use it to validate.
- Third, train the model on the other four folds, and measure the error on the held-out fold.
- Fourth, rotate.
- Each of the five folds takes a turn as the validation fold, so every observation is used for validation exactly once and for training four times.
- At the end, average the five errors to get one estimate of out-of-sample error.
- It uses all of the data, and it is far less dependent on one lucky split.

12 Example: five-fold check on depth five

Validation errors from the five folds (illustration)

Sum of five fold errors	11+13+12+12+12	**60%**
Number of folds, k	five parts	**5**
Average validation error	60 ÷ 5	**12%**
Times a fold trains	k − 1	**4**

IN THE LESSON

- Back to Priya's depth five tree.
- She runs five-fold cross-validation.
- The five validation errors, one from each fold, come out at eleven, thirteen, twelve, twelve and twelve per cent.
- Add them up and the total is sixty.
- There are five folds, so the average validation error is sixty divided by five, which is twelve per cent.
- That matches the single validation sample she saw before, which is reassuring.
- Notice how each fold plays two roles.
- It is the validation fold once, and part of the training data in the other four rounds, which is k minus one, or four.
- With k set to five, no data is wasted, and no one split can fool her.

13 Four mistakes to avoid so far

- 1. Images or text do not make it unsupervised
- 2. Judging a model by its training error
- 3. Wrong cure: underfit and overfit are opposites
- 4. Peeking at the test sample while tuning

IN THE LESSON

- Let us pause on the common slips.
- One.
- Calling a problem unsupervised because the data are images or text.
- The test is the label, not the data type.
- Two.
- Judging a model by its training error.
- A low training error alone proves nothing, and the more complex model nearly always wins that contest.
- Three.
- Reaching for more complexity when the problem is overfitting, or less when the problem is underfitting.
- The two diagnoses need opposite cures.
- Four.
- Using the test data to tune the model.
- Once the test sample has influenced your choices, it is no longer an honest, unseen check.
- Keep it locked away until the very end.

14 Six supervised tools and one question

- Penalised regression (LASSO): many features, few matter
- Support vector machine: widest boundary
- k-nearest neighbour: ask the closest cases
- Tree, forest, ensemble: rules and votes

IN THE LESSON

- Now to the toolbox.
- Think of a workshop with six tools.
- You would not drive a nail with a screwdriver, and the exam keeps asking which tool suits which job.
- The trick is to learn how each one works, then its strengths follow.
- Penalised regression, of which LASSO is the best known, is a linear model with a built-in brake on complexity.
- A support vector machine draws the widest possible boundary between two classes.
- K nearest neighbour asks what the closest known cases were.
- A classification and regression tree splits the data into branches of simple rules.
- A random forest averages many trees.
- Ensemble learning is the general idea of combining many models.
- For each, ask three things: how many features, what shape, and must I explain it.

15 LASSO: a penalty for every coefficient

Minimise: SSR + λ × (sum of absolute coefficients)

Sum of squared residuals	How badly the model misses
Lambda: penalty weight	Set by you, not estimated
Sum of absolute coefficients	Coefficient sizes, signs ignored
Penalised objective	Smaller is better

IN THE LESSON

- LASSO stands for least absolute shrinkage and selection operator, but let us just call it lasso.
- Ordinary regression picks the coefficients that make the sum of squared residuals as small as possible.
- Lasso adds a charge.
- It adds lambda times the sum of the absolute values of the coefficients.
- Every coefficient now has to earn its place.
- A feature stays only if it cuts the squared residuals by more than it costs in penalty.
- Weak ones are driven all the way to zero, which means they drop out.
- That makes lasso ideal when you have many candidate features.
- Lambda is a hyperparameter.
- You set it before fitting, usually by cross-validation.
- Too small and the model overfits.
- Too large and it underfits.

16 Example: lasso picks the features

Model 1: SSR 120, b = 0.8, −0.5, 0.3. Model 2: SSR 124, b = 0.9, 0, 0

Penalty base, Model 1	0.8 + 0.5 + 0.3	**1.6**
Penalty base, Model 2	0.9 + 0 + 0	**0.9**
λ = 10: Model 1 | Model 2	136 vs 133	**Model 2**
λ = 2: Model 1 | Model 2	123.2 vs 125.8	**Model 1**

A bigger λ accepts a worse fit for a simpler model

IN THE LESSON

- Priya uses lasso to choose predictors of earnings surprises.
- Model one has a sum of squared residuals of one hundred and twenty, and three coefficients: zero point eight, minus zero point five and zero point three.
- Add their absolute values, ignoring signs, and the penalty base is one point six.
- Model two has a sum of squared residuals of one hundred and twenty four, with one coefficient of zero point nine and the other two at zero.
- Its penalty base is zero point nine.
- With lambda equal to ten, model one scores one hundred and thirty six, model two one hundred and thirty three.
- Lower wins, so model two is chosen.
- With lambda equal to two, model one scores one hundred and twenty three point two, model two one hundred and twenty five point eight, so model one wins.

17 Support vector machine: the widest road

Chart: Profitability vs Leverage

- o o o
- o o
- x x x
- x x
- Boundary
- Margin

IN THE LESSON

- Now the support vector machine.
- Imagine two classes of bonds drawn on a chart, with leverage along the bottom and profitability up the side.
- The circles are bonds that stayed healthy and the crosses are bonds that defaulted.
- Many straight lines could separate them.
- The machine picks one particular line, the solid one in the middle, the boundary.
- It is the line with the widest possible margin, the empty road on either side, shown by the two dashed lines.
- Only the few points touching the road edges, the support vectors, decide where it goes.
- Add a new bond and you read which side it falls on.
- It suits classification with many features.
- With a kernel, the boundary can be curved.

18 k nearest neighbour: who is closest?

New fund at (2, 3). Straight-line distance to five labelled funds

A at (1, 2): outperformed	√(1 + 1)	**1.41**
B at (2, 5): underperformed	√(0 + 4)	**2.00**
C at (4, 4): underperformed	√(4 + 1)	**2.24**
D at (3, 3): outperformed	√(1 + 0)	**1.00**
E at (5, 5): underperformed	√(9 + 4)	**3.61**

IN THE LESSON

- Next, k nearest neighbour.
- It has no equation to fit.
- To classify a new observation, it finds the k closest labelled observations and takes a majority vote.
- Priya has five past funds, each with two standardised scores, and a label saying if it later outperformed or underperformed.
- A new fund scores two on the first and three on the second.
- We measure straight-line, or Euclidean, distance.
- Fund A is at one and two, so its distance is the square root of one plus one, one point four one.
- Fund B at two and five is exactly two away.
- Fund C at four and four is two point two four.
- Fund D at three and three is one.
- Fund E at five and five is three point six one.
- Distance depends on the units, so put the features on the same scale first.

19 Example: the answer depends on k

Majority vote among the k closest funds

k = 1	D: outperformed	**Outperform**
k = 3	D, A outperformed; B not	**Outperform**
k = 5	2 outperformed, 3 did not	**Underperform**
Small k | large k	overfits | underfits	**Use cross-val**

IN THE LESSON

- Now vote.
- Order the funds by distance: D at one, A at one point four one, B at two, C at two point two four, then E.
- With k equal to one, only fund D votes, and it outperformed, so the new fund is called an outperformer.
- With k equal to three, the three closest are D, A and B.
- Two outperformed and one did not, so the majority says outperformer again.
- With k equal to five, everyone votes.
- Two outperformed and three underperformed, so the answer flips.
- The choice of k matters.
- A very small k chases individual points and overfits.
- A very large k blurs everything together and underfits.
- Choose k by cross-validation, and always say which distance measure you used.

20 Trees: readable rules, but they overfit

- Split on the feature and cut-off that separates best
- Repeat down each branch, stop at a rule
- Category target: classification; number: regression
- Easy to read, but a deep tree overfits

IN THE LESSON

- A classification and regression tree, called a cart for short, works like a game of twenty questions.
- At the top, the algorithm looks at every feature and every possible cut-off, and picks the single split that best separates the target.
- Perhaps, is the debt ratio above sixty per cent.
- Each answer sends the observations down a branch, and the process repeats on each branch until a stopping rule is met.
- The final groups are called terminal nodes.
- A category target gives a classification tree, and a numerical one a regression tree.
- The big plus is that anyone can read the rules.
- The big minus is that a deep tree memorises its training data, so it overfits.
- The cures are pruning and a limit on depth.

21 Random forest: a crowd of trees

Random forest

Many bootstrap samples → One tree per sample, random features → Each tree votes → Majority vote decides

IN THE LESSON

- Here is the main fix for a single tree's overfitting: ensemble learning, which means combining many models, so that individual mistakes cancel out.
- The best known case is the random forest.
- Follow the boxes.
- First, draw many bootstrap samples, which are random samples of the data, taken with replacement, so each sample is a little different.
- Second, grow one tree on each sample, and at every split let the tree choose only from a random subset of the features.
- That makes the trees different from each other.
- Third, each tree casts a vote.
- Fourth, take the majority.
- If sixty two of a hundred trees say default, the forest says default.
- The price is lost readability.
- A forest is hard to explain, but it overfits less than any single tree.

22 Which tool for which problem?

Match the problem to the tool

Many features, few matter	shrink weak ones to zero	**LASSO**
Two classes, many features	widest margin	**SVM**
Judge by similar past cases	vote of k closest	**k-NN**
Must show readable rules	if-then splits	**Single tree**
Accuracy over readability	average many trees	**Random forest**

IN THE LESSON

- Now put it together, the way the exam asks.
- If you have many candidate features and suspect most of them do not matter, use penalised regression such as lasso, as it sets weak coefficients to zero.
- If you have two classes and many features, and want the cleanest boundary, use a support vector machine.
- If there is no tidy model, and the best guide is similarity to past cases, use k nearest neighbour, though it slows down on big data sets.
- If you must show a client the exact rules, use a single tree, accepting the overfitting risk.
- If accuracy matters more than readability, use a random forest, or another ensemble.
- The trap is to pick the fanciest method.
- Match the tool to the job.

23 A story: the shelf and the shirts

- No label: look for structure in the inputs
- PCA squeezes columns: many features to few
- Clustering sorts rows: many stocks to few groups
- Features versus observations: do not mix them up

IN THE LESSON

- Now the other family.
- Unsupervised learning has no label, so we ask what structure the inputs have by themselves.
- Imagine a spreadsheet of three hundred stocks, with ten columns of data about each.
- Two different jobs come up.
- The first is that several columns say almost the same thing, such as three different valuation ratios that move together.
- You would like to squeeze ten columns into a few.
- That is dimension reduction, and the tool is principal components analysis.
- The second is that you want to sort the three hundred stocks, the rows, into a few shelves of similar stocks.
- That is clustering.
- Principal components analysis reduces features, and clustering groups observations.
- Keep those two jobs apart.

24 PCA: how much does each component explain?

Share of variance = eigenvalue ÷ sum of all eigenvalues
Keep components until the running total reaches the threshold

Eigenvalue	Variance one component explains
Sum of eigenvalues	Equals the number of features
Share of variance	Eigenvalue ÷ sum
Running total	Add shares, largest first
Threshold	Often 85% to 95%

IN THE LESSON

- Principal components analysis, or P C A, replaces many correlated features with a smaller set of new composite features, called components.
- The components are uncorrelated with each other.
- The first captures as much variation as possible, the second as much of what remains, and so on.
- Each component comes with an eigenvalue, which measures the variance it explains.
- To get its share, divide its eigenvalue by the sum of all the eigenvalues.
- For standardised data, that total equals the number of features.
- Then add shares up as a running total, and keep enough components to pass a chosen threshold, often eighty five to ninety five per cent.
- A scree plot, which shows each component's share, helps you spot the elbow where the gains flatten.

25 A scree plot for ten features

Share of variance

PC1	**42%**
PC2	**23%**
PC3	**12%**
PC4	**8%**
PC5	**5%**
PC6	**4%**

Eigenvalues sum to 10.0 (illustration)

IN THE LESSON

- Here is the picture, a scree plot drawn as bars.
- Priya has ten standardised features for three hundred stocks.
- The ten eigenvalues sum to ten, one for each feature.
- The bars show the share of variance explained by the first six components.
- The first explains forty two per cent on its own.
- The second explains twenty three, the third twelve, the fourth eight, the fifth five and the sixth four.
- See how the bars shrink fast and then flatten.
- That flattening is the elbow.
- The small bars beyond it are the scree, the rubble at the foot of a slope, which is where the plot gets its name.
- The last four components together explain very little.

26 Example: how many components to keep?

Ten standardised features. Goal: keep 85% of the variance

Total variance	sum of eigenvalues	**10.0**
PC1 share	4.2 ÷ 10.0	**42%**
First three components	(4.2+2.3+1.2) ÷ 10.0	**77%**
First four components	(7.7+0.8) ÷ 10.0	**85%**

Ten features become four uncorrelated components

IN THE LESSON

- Priya wants to keep eighty five per cent of the variance.
- The ten eigenvalues are four point two, two point three, one point two, zero point eight, zero point five, zero point four, zero point three, zero point one five, zero point one and zero point zero five.
- They add up to ten.
- The first component's share is four point two over ten, forty two per cent.
- The first three: four point two plus two point three plus one point two is seven point seven, so seventy seven per cent.
- Not enough.
- Add the fourth, zero point eight, to reach eight point five, which is exactly eighty five per cent.
- So she keeps four.
- The trade-off: the four components are hard to interpret, because each blends many original features.

27 k-means: pick, assign, move, repeat

k-means clustering

Choose k, place k centroids →
Assign each point to nearest →
Move centroids to group average →
Repeat until nothing moves

IN THE LESSON

- Now clustering, starting with k-means.
- Here k is the number of clusters, and you must choose it before you start.
- Follow the boxes.
- First, pick k and place k starting points, called centroids.
- Second, assign every observation to its nearest centroid, which forms the clusters.
- Third, move each centroid to the average position of the observations in its cluster.
- Fourth, go back to the second step and repeat.
- Stop when no observation changes cluster.
- The result depends a little on the starting points, so analysts often run it several times.
- K-means is fast and works well on large data sets, but you must know, or guess, how many groups you want.
- And it likes clusters of roughly round, similar size.

28 Example: k-means on six risk scores

Scores 1, 2, 3, 9, 10, 12. Two clusters, poor start at 1 and 2

Start	centroids at 1 and 2	**k = 2**
After round 1	centroids 1 and 7.2	**1 | 5 points**
After round 2	centroids 2 and 10.33	**3 | 3 points**
Round 3	nobody changes group	**Stop**

Final clusters: {1, 2, 3} and {9, 10, 12}

IN THE LESSON

- A tiny example.
- Six funds have risk scores of one, two, three, nine, ten and twelve.
- Priya sets k to two, with a poor start: centroids at one and two.
- Round one.
- One goes with the first centroid.
- Two, three, nine, ten and twelve are all closer to the second.
- New centroids: one, and the average of those five, thirty six over five, seven point two.
- Round two.
- Now two and three are closer to one, and nine, ten and twelve to seven point two.
- New centroids: two, and thirty one over three, ten point three three.
- Round three.
- Nobody changes group, so it stops.
- The clusters are one, two, three, and nine, ten, twelve, just as the eye suggests.

29 Hierarchical clustering and the dendrogram

- Agglomerative: bottom up, merge the closest
- Divisive: top down, keep splitting
- Dendrogram: tree showing merge distances
- Cut the tree later, so k is not fixed in advance

IN THE LESSON

- Hierarchical clustering builds a whole tree of clusters rather than a single answer.
- The bottom-up version is called agglomerative.
- Every observation starts alone, and at each step the two closest clusters merge, until only one remains.
- The top-down version, divisive, starts with everything in one cluster and keeps splitting.
- The result is drawn as a dendrogram, a tree diagram where the height of each join shows the distance at which the merge happened.
- The big advantage is that you do not choose the number of clusters in advance.
- You look at the tree, then cut it at a height you like.
- The cost is speed.
- It is slower than k-means, so it is better on smaller data sets.

30 Example: the same six scores, as a tree

Merge the closest clusters first (nearest-member distance)

Merge 1	1 and 2 join	**distance 1**
Merge 2	3 joins {1, 2}	**distance 1**
Merge 3	9 and 10 join	**distance 1**
Merge 4	12 joins {9, 10}	**distance 2**
Merge 5	the two groups join	**distance 6**

Cut between 2 and 6: two clusters

IN THE LESSON

- Now the same six scores, one, two, three, nine, ten and twelve, merged bottom up.
- Here the distance between two clusters is the gap between their nearest members.
- One and two are one apart, so they merge first at a distance of one.
- Next, three is one away from the group one and two, so it joins at one.
- Nine and ten merge at one.
- Then twelve is two away from ten, so it joins at two.
- Finally, the two groups join, and their nearest members, three and nine, are six apart.
- So the last merge is at six.
- Cut the dendrogram anywhere between two and six, and you get two clusters, matching k-means.
- Cut lower, at one and a half, and you get three.

31 Which unsupervised tool for which problem?

Unsupervised: features versus observations

Problem	Best fit
Too many correlated features	PCA
Group rows, k is known	k-means
Group rows, k unknown	Hierarchical
Very large data set	k-means

IN THE LESSON

- Last, match problem to tool.
- If the trouble is too many correlated features, and you want a few uncorrelated ones, use principal components analysis.
- Remember that it works on columns, and its components are hard to interpret.
- If you want to group observations and already know how many groups you need, use k-means, which is fast on big data.
- If you do not know how many groups there should be, or you want to see how observations combine step by step, use hierarchical clustering and decide where to cut afterwards.
- The common slip is to say that principal components analysis groups similar stocks.
- It does not.
- It reduces the number of features.
- Clustering groups observations.

32 Recap: the whole lesson in five lines

- Label or no label decides supervised or not
- Overfit: low training error, high validation error
- Fix it: simplify, penalise, cross-validate
- Supervised: LASSO, SVM, k-NN, tree, forest
- Unsupervised: PCA for columns; clusters for rows

IN THE LESSON

- Let us pull it together.
- Supervised learning has a labelled target, a number for regression and a category for classification.
- Unsupervised learning has none, and it reduces features or groups observations.
- Deep learning is a model type, not a family.
- Overfitting means low training error but high validation error.
- Total error is bias plus variance plus base error, and the cures are less complexity and honest estimates of out-of-sample error such as k-fold cross-validation.
- The supervised tools are lasso, support vector machine, k nearest neighbour, tree, random forest and other ensembles, each suited to a different job.
- The unsupervised tools are principal components analysis for features, and k-means and hierarchical clustering for observations.

33 Try three quick questions

1. Eigenvalues 2.8, 1.6, 0.9, 0.4, 0.2, 0.1. Components needed for 90%?
   Answer: Three give 88.3%; four give 95%. So four.
2. Training error 3%, validation error 21%. Diagnosis and a cure?
   Answer: Overfitting (gap 18 points). Simplify: prune or penalise.
3. LASSO: SSR 50, b = 0.5, −0.2, 0, λ = 4. Objective?
   Answer: 50 + 4 × 0.7 = 52.8

IN THE LESSON

- Now try three questions yourself.
- Pause after each one.
- Question one.
- Six standardised features have eigenvalues of two point eight, one point six, zero point nine, zero point four, zero point two and zero point one.
- To keep ninety per cent of the variance, how many components?
- The total is six.
- Three components give five point three over six, eighty eight point three per cent, short of ninety.
- Four give five point seven over six, ninety five per cent.
- So four.
- Question two.
- A model has a training error of three per cent and a validation error of twenty one.
- What is the diagnosis and one cure?
- Overfitting, a gap of eighteen points.
- Simplify it, for example by pruning or penalising.
- Question three.
- A lasso has a sum of squared residuals of fifty, coefficients of zero point five, minus zero point two and zero, and a lambda of four.
- What is the objective?
- The penalty base is zero point seven, times four is two point eight, so the answer is fifty two point eight.