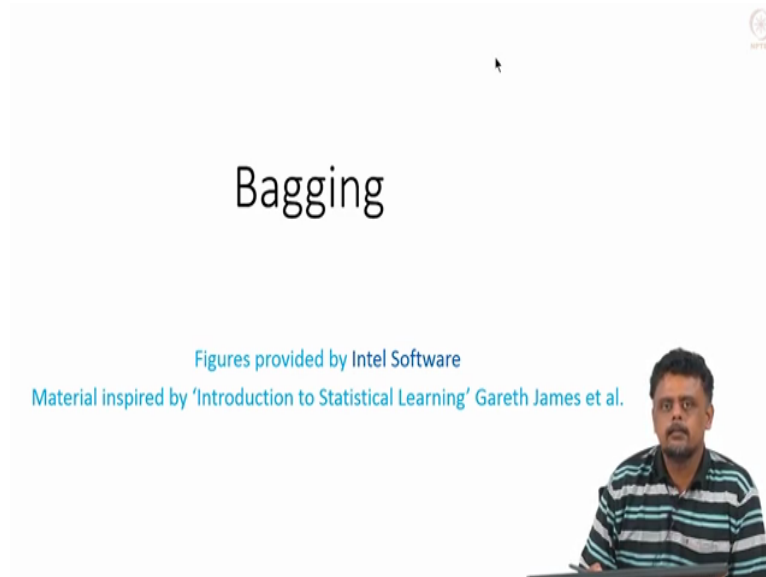


Machine Learning for Engineering and Science Applications
Professor Dr. Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology, Madras
Bagging

(Refer Slide Time: 00:13)



Hello and welcome back, in this video we will look at bagging continuing from our lecture on decision trees. So all the figures so in this presentation are provided by Intel software and the material is inspired from introduction to statistical learning by Gareth James.

(Refer Slide Time: 00:32)

The slide is titled "Introduction" in a black, sans-serif font. On the left side, there is a decision tree diagram with a root node (grey) that branches into two nodes (grey). The left branch leads to two leaf nodes (red and blue), and the right branch leads to two more nodes (grey), which then branch into four leaf nodes (red, blue, red, blue). To the right of the diagram is a list of three bullet points: "• Problem: decision trees tend to overfit", "• Pruning helps reduce variance to a point", and "• Often not significant for model to generalize well". The phrase "high variance" is written in red cursive above the second bullet point. In the bottom right corner, there is a small inset video of the same man from the previous slide, now holding a pen. A small circular logo is visible in the top right corner of the slide.

So based on what we saw last time what decision trees they have a tendency to over fit the data that is it performs very well on training data but new test data sets arrives and the errors are huge, so basically it is a high variance model, it is another way of looking at it, so it is high variance. So one solution to prevent over fitting in decision trees was to prune the trees but it helps reduce variance to a certain point but beyond that it does not help really and the effects are not very significant, ok.

So bagging is a procedure that is developed to improve the or to reduce the high variance of decision trees.

(Refer Slide Time: 01:25)

Bagging

z_1, \dots, z_n
Variance $= \sigma^2$
 $\frac{z_1 + z_n}{n}$
Variance $= \frac{\sigma^2}{n}$

Train a multitude of Decision Trees and combine their predictions to reduce variance

You will look at how it does, so the idea behind bagging is to train a lot of decision trees on a given data set ok, so we train a multitude of decision trees and combine their predictions to reduce the variance right, so if you have this is based on this very simple concept that let us say you have data points z_1 to z_n and your variance on individual data points is sigma square, so if there are n data points and if you look at the mean the variance of that mean is sigma square over 1 over n , right.

So as you increase the number of data points you are variance on the mean comes down that is the concept it exploited with bagging, so you train a lot of trees on a given data set and you take the average of these predictions if you are training regression trees or you combine the predictions for classification some way or the other.

(Refer Slide Time: 02:37)

How to train Multiple Trees?

Grow decision tree from each bootstrapped sample

Title	Budget	Turnover	Director	Rating
The Godfather Part II	13000000	173000000	Francis Ford Coppola	9.2
The Godfather	12000000	134000000	Francis Ford Coppola	9.2
The Godfather Part III	17000000	133000000	Francis Ford Coppola	8.9
The Untouchables	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Begins	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Continues	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Ends	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Lives	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Returns	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Rides Again	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Strikes Back	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Takes a Ride	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Ties the Knot	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Unites	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Wins	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Yields	10000000	100000000	John Huston	8.8
The Untouchables: The Legend Zips	10000000	100000000	John Huston	8.8

B -> boot stop samples are generated

So where do we get the data set for training these multiple trees, right because we have only one data set so ideally what we would like to do is to have multiple measurements and with each measurement in the centre is a measurement you collect data 100 times and with every data you collect you train a decision tree however that is typically not possible because you are generally given a data set or only a set of data is available and you have to grow your tree based on that.

So the way to train multiple trees is to grow decision trees from bootstrapped samples, ok so what do you mean by bootstrapping samples? That is what we are going to look at, so there is this data set this is about this about movies and their budget and the turnover, total gross steak who is a director of the movie, the rating and so on and so forth. This is a movie data set so we are trying to make some decision about this based on this data so we will not go into that detail I just to understand the data set.

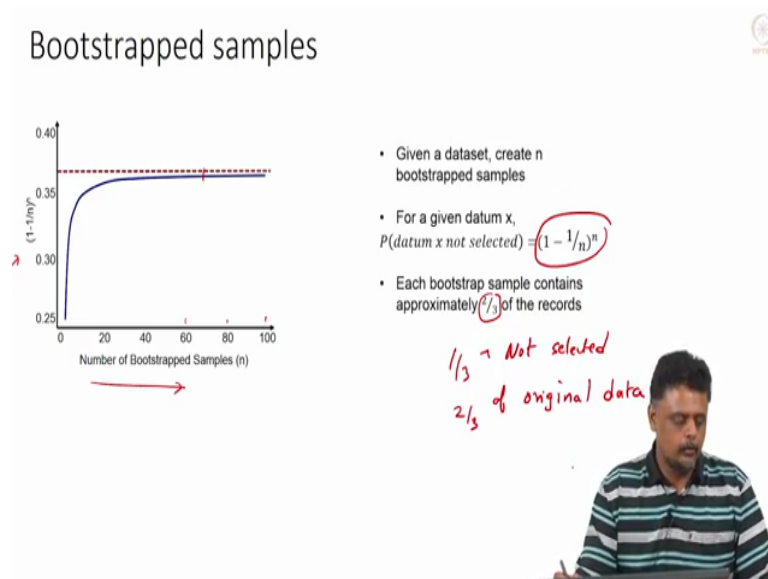
So the way we go about doing bootstrapping is to select a subset of this data with replacement, so if you look at this particular so the blue area is the sample data, so the idea behind bootstrapping is to sample your given data with replacement to create a new data set. So if you look at this particular movie database detail so there are 17 data points and we are selecting a subset of this data marked in blue right with replacement, so we do it let us say capital B number of times.

So B bootstrap samples are generated and we use each one of these B bootstrap samples to train a decision tree and use the output of the decision trees let us say as an average or a

voting scheme to get at the desired result. So the way it works is so if you look at this particular realization we have a different subset marked in blue here, we have another realization of the data that is sample of the data with this which is sampling a different portion of the data here as well as in this case another sample of the data.

So you can sample different subsets of the data with replacement and with each subset you can train a decision tree and use the output and average the output for classification regression and some voting scheme for the decision making.

(Refer Slide Time: 05:26)



So if you do that what happens what is the nature of the data sets that we get when we do this kind of sampling? So on an average let us say if you have this is in this plot you see this axis basically the number of bootstrap samples so each time you select 10, 20, 40 or 80 or 100 samples from your data and this is the probability of a particular data set data point not being present in the data ok, so that is given by this expression here 1 minus 1 over n raised to n.

So if you look beyond a point as you as a number of bootstrap samples increases, so we see that typically every bootstrap sample contains two thirds of your original data and one third of your original data is not present in the samples on an average so that is the makeup of your sample data. Now this again this is possible because we are sampling with replacement so just to reiterate as you increase the number of bootstrap samples you see that the number the percentage of data left out is like what one third, so one third is not selected typically your bootstrap samples are made up of two thirds of original data, so this refers to the individual data points, right

So we will use we will exploit this fact to do error estimation later on but we will just see how we can go about doing this using these bootstrap samples for training decision trees and obtaining an output.

(Refer Slide Time: 07:17)

Aggregate Results

Trees vote on or average result for each data point



Bagging = Bootstrap Aggregating

Form a
Single
Classifier



So we have let us say in this illustration we have about multiple trees ok and each of those trees are trained on one of the bootstrap samples, bootstrap data sets and there is an output corresponding to each of those trees. Let us say this is a classification task so each one of these trees outputs let us say it is a binary classification, so we are looking at either red or blue right, so each one of these trees outputs for a particular data point it outputs the decision.

So two of the trees said red and one tree said blue ok, so then you can so for a classification task you can just do maximum or the voting scheme so since two red gets the maximum number of votes you would classify that particular data point as right, ok. So similarly we can do this for all the data points, so for every data point so each one of these columns is a data point so every data point each of these trees will output a certain category red or blue and we assign a category which gets the maximum number of votes.

So as you go along the rows as you go along the columns each column you will select the category which got the maximum number of votes, ok so this is for a classification task, so in the end you will get a single classifier ok. There is also for instance regression wherein you will just get the take the mean, mean or the average ok, so there are many ways of going about this if you think about it let us say think of a classification task then based on the output of the individual decision trees you can formulate these output probabilities which is nothing

but so let us say there are three classes then you can have the probability of class 1, class 2 and class 3 and each as the fraction of the trees which what which gave this as output class 1 as output, right.

A fraction of trees that gave class 1 as output, so people there is it I mean you can do it this way but the better way would be to actually get the raw probabilities from the output of the tree itself right, so for instance each tree will output a certain class with the probability that you can calculate ok based on the leaf which is it is aware to which it is assigned and you can actually take the average probability over the leaves of every one of those trees and use that as to classify into a particular class ok but this is even though this is time you know appealing to do and the better approach would be to directly estimate the probability from the output of each of the individual trees and then average those probabilities instead.

In this case we are considering the fraction of trees that give a particular classes output ok. So in the end so what are we doing we are bootstrapping our data set and with each bootstrap sample we are building decision trees and aggregating the output of those Bayesian trees, so it is bootstrap so bagging is nothing but bootstrap aggregating, ok. So that is the idea behind bagging and the what it improves is the high variance so it brings down the variance in your model because decision trees tend to over fit.

So having multiple a multitude of decision trees strain on the similar data will give you a better average output.

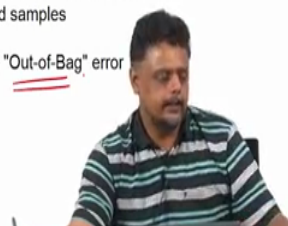
(Refer Slide Time: 11:25)

Bagging Error Calculations

Tree	Class	Percentage of Data	Percentage of Error
1	1	0.5	0.0
1	2	0.5	0.0
2	1	0.5	0.0
2	2	0.5	0.0
3	1	0.5	0.0
3	2	0.5	0.0
4	1	0.5	0.0
4	2	0.5	0.0
5	1	0.5	0.0
5	2	0.5	0.0
6	1	0.5	0.0
6	2	0.5	0.0
7	1	0.5	0.0
7	2	0.5	0.0
8	1	0.5	0.0
8	2	0.5	0.0
9	1	0.5	0.0
9	2	0.5	0.0
10	1	0.5	0.0
10	2	0.5	0.0
11	1	0.5	0.0
11	2	0.5	0.0
12	1	0.5	0.0
12	2	0.5	0.0
13	1	0.5	0.0
13	2	0.5	0.0
14	1	0.5	0.0
14	2	0.5	0.0
15	1	0.5	0.0
15	2	0.5	0.0
16	1	0.5	0.0
16	2	0.5	0.0
17	1	0.5	0.0
17	2	0.5	0.0
18	1	0.5	0.0
18	2	0.5	0.0
19	1	0.5	0.0
19	2	0.5	0.0
20	1	0.5	0.0
20	2	0.5	0.0
21	1	0.5	0.0
21	2	0.5	0.0
22	1	0.5	0.0
22	2	0.5	0.0
23	1	0.5	0.0
23	2	0.5	0.0
24	1	0.5	0.0
24	2	0.5	0.0
25	1	0.5	0.0
25	2	0.5	0.0
26	1	0.5	0.0
26	2	0.5	0.0
27	1	0.5	0.0
27	2	0.5	0.0
28	1	0.5	0.0
28	2	0.5	0.0
29	1	0.5	0.0
29	2	0.5	0.0
30	1	0.5	0.0
30	2	0.5	0.0
31	1	0.5	0.0
31	2	0.5	0.0
32	1	0.5	0.0
32	2	0.5	0.0
33	1	0.5	0.0
33	2	0.5	0.0
34	1	0.5	0.0
34	2	0.5	0.0
35	1	0.5	0.0
35	2	0.5	0.0
36	1	0.5	0.0
36	2	0.5	0.0
37	1	0.5	0.0
37	2	0.5	0.0
38	1	0.5	0.0
38	2	0.5	0.0
39	1	0.5	0.0
39	2	0.5	0.0
40	1	0.5	0.0
40	2	0.5	0.0
41	1	0.5	0.0
41	2	0.5	0.0
42	1	0.5	0.0
42	2	0.5	0.0
43	1	0.5	0.0
43	2	0.5	0.0
44	1	0.5	0.0
44	2	0.5	0.0
45	1	0.5	0.0
45	2	0.5	0.0
46	1	0.5	0.0
46	2	0.5	0.0
47	1	0.5	0.0
47	2	0.5	0.0
48	1	0.5	0.0
48	2	0.5	0.0
49	1	0.5	0.0
49	2	0.5	0.0
50	1	0.5	0.0
50	2	0.5	0.0

100 - trees
30 - trees do not use data point X40

- Bootstrapped samples provide built-in error estimate for each tree
- Create tree based on subset of data
- Measure error for that tree on unused samples
- Called "Out-of-Bag" error



So one of the ways of calculating the error or validating your machine learning algorithm as we have seen earlier is the cross validation that the k cross k fold cross qualifier cross validation rainfall cross validation, it is kind of difficult to do as you can imagine with the bagging approach. So what would be the typical approach taken in for when you are using bagging, right.

So we saw that about one third of the data samples are left out on an average when in every bootstrapped version of your data set, so once you have created a tree based on a subset of the data you can measure the error on the unused samples, so ok. So let us say you have about 100 trees right ok so let us say 30 trees do not use data point some data point I will call it X_{40} or something so let us say you have 120 data points X_{40} is one data point X_{40} is one data point and 30 of those trees do not use that so then you will evaluate the error on X_{40} on those 30 trees individually and average them, ok.

So that will give you so you can for every subset of trees that do not use a particular data point you can use those trees to evaluate the result for that data point and use that average as a measure of the error in your algorithm, right. So this can be root mean square error if you are doing a regression task or the classification error in case of a k fold or a twofold classification, ok.

This procedure is called the out of bag error estimation correct, so that is one of the this is one of the advantages of using bagging so you do not have to specifically do k fold cross validation you just have to identify data points that are not used in subsets of the decision trees that were used for the bagging procedure and you evaluate the error using only those trees on that particular data point.

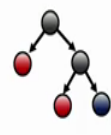
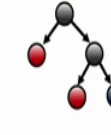
So you can like that you can identify on almost one third of your data set on an average it is not being used in one tree or the other, so then you can use you can accumulate their error overall those data points and calculate an average ok, that gives you either for classification or for regression.

(Refer Slide Time: 14:06)

Calculation of Feature Importance

Step	Title	Author	Stream/Platform/Source	View/Number
1	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
2	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
3	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
4	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
5	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
6	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
7	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
8	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
9	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
10	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
11	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
12	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
13	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
14	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
15	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
16	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
17	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
18	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
19	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
20	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
21	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
22	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
23	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
24	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
25	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
26	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
27	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
28	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
29	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
30	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
31	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
32	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
33	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
34	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
35	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
36	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
37	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
38	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
39	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
40	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
41	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
42	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
43	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
44	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
45	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
46	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
47	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
48	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
49	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
50	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18

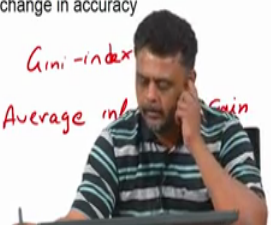
Step	Title	Author	Stream/Platform/Source	View/Number
1	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
2	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
3	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
4	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
5	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
6	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
7	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
8	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
9	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
10	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
11	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
12	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
13	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
14	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
15	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
16	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
17	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
18	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
19	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
20	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
21	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
22	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
23	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
24	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
25	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
26	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
27	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
28	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
29	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
30	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
31	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
32	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
33	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
34	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
35	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
36	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
37	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
38	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
39	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
40	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
41	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
42	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
43	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
44	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
45	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
46	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
47	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
48	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
49	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18
50	2018-11-15 The Average Salary Getting by...	1000000000000000000	Pratik Lakshani	296.12.18

• Fitting a bagged model doesn't produce coefficients like logistic regression

• Instead, feature importances are estimated using oob error

• Randomly permute data for particular feature and measure change in accuracy



Similarly you can do a same procedure for feature importance right, so feature importance can be either measured using the classification error or typically the Gini index will give you the measure of feature importance what do I mean by that is we saw that we are looking when we are trying to do the split at every node in a tree we are looking at we are choosing features based on the largest change in the Gini index or the entropy you are the classification error.

So if we can identify so it is the same procedure as before but we on the unused data set but we will use instead of accommodating the error we will accumulate Gini index change and average that overall the trees ok, so that way will give you a good idea of which feature is the most important. So recall that it is much easier to do this if you have a single decision tree because at every node whenever there is the whenever there is a split you know the change in the Gini index or the entropy that you or whatever criteria that we have used to make the split and you can use that as a measure of feature importance ok but when you are doing a bagging there is when you are averaging over a multitude of trees maybe several hundred of them and then it is very difficult to do that in a straightforward way.

So because the data set will keep changing between trees typically so then we just have to do a similar procedure that we did for calculating the output of the tree so instead of averaging over the root mean square error or the classification error you look at the Gini index and you average over that for every feature using data points that were not there as the training data or in the data that was not used to make that tree, ok.

So if you can use that for recall that when we were talking about when you are talking about this decision trees we also looked at something called feature importance, so we figured out that the feature that was the most pertinent to the task at hand was the one that gave the biggest information gain as measured with the Gini index for instance ok. So this was our way of figuring out the most significant features in our data, right.

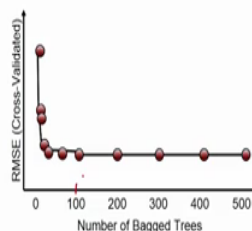
So when it comes to bagging it is not one decision tree that we are looking at we are looking at what 100 of trees and then it is very difficult to make this decision how do we figure out which is the most feature important feature right, so here again we do exactly something similar to what we did earlier for calculating the out of bag error so we will look at the average information gain, right so we look at the average information gained overall the trees for that particular feature and that gives you an idea of if that feature is very important or not, ok.

So it is during the process of training like we saw we take bootstrap samples and train every tree with it for every tree and whenever we make a split on the data based on the information gained we keep track of the information gained for that particular feature across all the trees whenever that feature is used and we take an average of that and that gives you the feature importance when you are using bagging as a (clas) for classifying right or even for regression right.

(Refer Slide Time: 18:04)

How many Trees?

- Bagging performance improvements increase with more trees




So are typically bagging performance increases with the number of trees ok, so if you have about 100 or more stated in many text books and on resources that per 100 trees should do it

ok they should be able to get a good average and since you can reduce the variance significantly if you use about 100 trees to fit your data ok and of course you can always measure the error using out of bag error and you can also look at a feature importance just by even averaging across the trees.

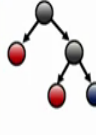
(Refer Slide Time: 18:37)

Bagging Advantages

Year	Type	Age	Qualification	Salary
2010	Full Time	25	Graduate	25000
2011	Part Time	30	Post Graduate	30000
2012	Full Time	35	Graduate	35000
2013	Part Time	40	Post Graduate	40000
2014	Full Time	45	Graduate	45000
2015	Part Time	50	Post Graduate	50000
2016	Full Time	55	Graduate	55000
2017	Part Time	60	Post Graduate	60000
2018	Full Time	65	Graduate	65000
2019	Part Time	70	Post Graduate	70000
2020	Full Time	75	Graduate	75000
2021	Part Time	80	Post Graduate	80000
2022	Full Time	85	Graduate	85000
2023	Part Time	90	Post Graduate	90000
2024	Full Time	95	Graduate	95000
2025	Part Time	100	Post Graduate	100000

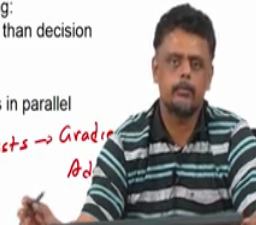


- Same as decision trees:
 - Easy to interpret and implement
 - Heterogeneous input data allowed, no preprocessing required



- Specific to bagging:
 - Less variability than decision trees
 - Can grow trees in parallel

Random forests → Gradient Boost



So the advantages of bagging they are same as decision trees they are easy to interpret and implement ok, you do not have to do any pre-processing especially for bagging right whatever pre-processing you do for to the data centering it at zero unit variance those things that you do for the data remains the same, so all that there is no extra data pre-processing or anything of that sort and of course doing bagging improves the variance, so there is less variance in your output and training multiple trees can be done in parallel because they are all done independently, there is no correlation in their training.

So in that way you know one the training of one tree dependent does not depend on the training of the other so you can train multiple trees simultaneously of course you have to implement them in in a certain programming construct so that there is no this can be done very quickly right. So bagging is a very useful tool to have especially if you are using decision trees and many of the python packages come with it so you can try is scikit learn or any other module python module that offers bagging inbuilt.

So we have looked at decision trees and we also now looked at bagging which is basically just exploits the fact that if you average over a large number of trees then you get a better variance but the problem still exists because if you see we are sampling the data with

replacement and the process for growing the trees remains the same, so if you consider a situation where in a certain feature let us say in this case a director ok of the movie in this case in this data set has a significant impact on whatever classification that you are trying to do with it and that is this is the most important feature.

So even though you are selecting bootstrap samples of data the first split is going to be on the director right, so that way all the trees that you trained or correlated and if you average over a bunch of correlated variables then you are then you do not get such a huge reduction in variance ok, so that is not great so if a data set is like that that where in a certain feature has a huge impact on the classification output then it does not really help.

So to alleviate that problem we will look at what are called random forests ok and random forests or just another variation of this not it is not exactly a variation but random forests what you do is the same thing you train a lot of trees but then you will not only select the data points at random will also select the features at random and that helps to weaken the correlation between the trees that you train and improves your output, ok.

So we have looked at look at this in slightly more detail and following random for us we will also look at a gradient boosting or and adaboost and these techniques are also generally very powerful for improving the accuracy of whatever classifier that you might be using, we look at these in the next few videos, thank you.