

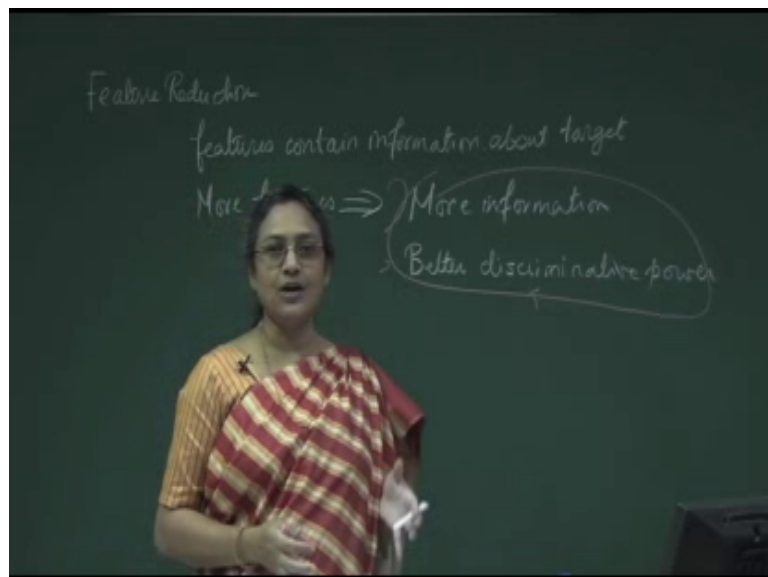
Introduction to Machine Learning
Prof. Sudeshna Sarkar
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Module - 3
Lecture - 11
Feature Selection

Good morning. Today, we are in the third module which is on Instance Based Learning and Feature Reduction. We have the Part B of this module where we will talk about feature selection.

In the last class we have looked at instance based learning, and we have seen that in instance based learning given the test data instance we have to find out the nearby instances for this we need a distance function. This distance function is computed in term of the features. If the number of features is large there is a problem, because the distance that you get may not be representative of the actual distance. So, this is a reason why feature reduction becomes important.

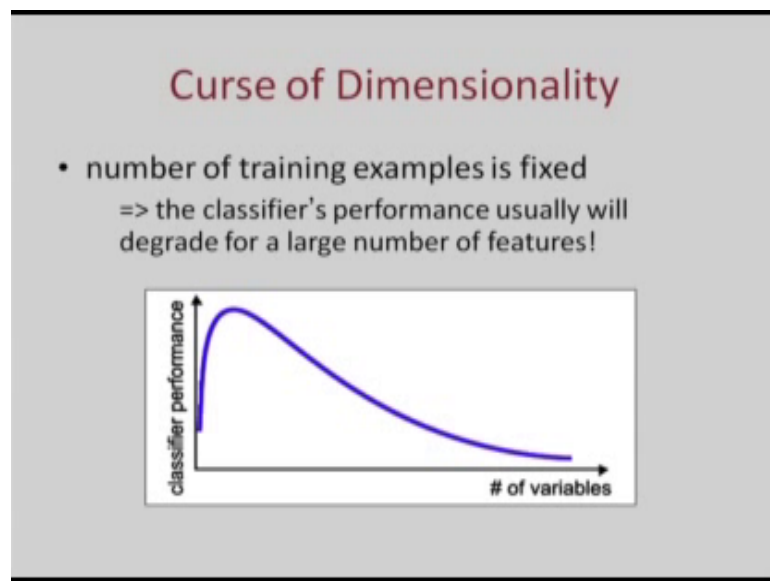
(Refer Slide Time: 01:10)



Now, we have seen that or you know that information about the target, so features contain information about the target. And the function the classification function is

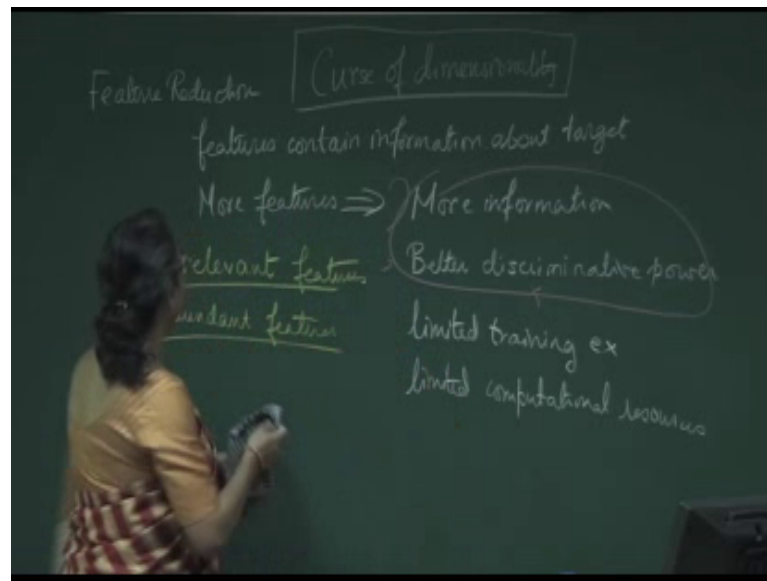
defined in terms of the features. So you may be tempted to think that more features means better information or more information, and better discriminative power or better classification power. But is this really the case always. We will see that this is not always the case; this may not hold always that just because you have more features does not mean you have more information or better classification performance.

(Refer Slide Time: 02:44)



If you look at the slide, this is one typical scenario. As you keep the number of training examples fixed and the training set is not extremely large then typically what may happen is that, when you increase the number of features initially the classifier performance may go up and then the classifier performance may be degradation. So, this is the reason why can it be.

(Refer Slide Time: 03:19)



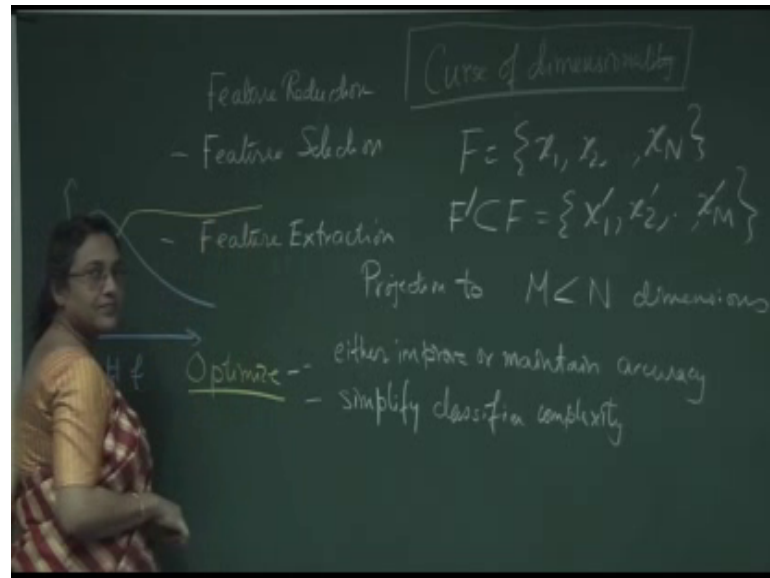
The reason is some features can be irrelevant. In algorithm such k nearest neighbor these irrelevant features introduce noise and they fool the learning algorithm. Because you are trying to find which instance is a close together, these irrelevant features or noisy features will make this make the result wrong.

Secondly you make have redundant features. If you have a fixed number of training examples and redundant features which do not contribute additional information they may lead to degradation in performance of the learning algorithm. These irrelevant features and redundant features can confuse learner, especially when you have limited training examples, limited computational resources. When we have a large number of features this source space the hypothesis space will be larger and searching may take more time depending on the algorithm that you can use. And with limited training examples you cannot work with large number features because we have seen that it leads to over fitting.

So, these things contribute to phenomena which we call the curse of dimensionality. When you have too many features, too many dimensions this will lead to the gradation of the learning algorithm more computational time and this phenomena is called the Curse of Dimensionality. To overcome this curse of dimensionality we want to do feature

reduction. There are 2 types of feature reduction; one is called Feature Selection, the other is called Feature Extraction.

(Refer Slide Time: 06:12)



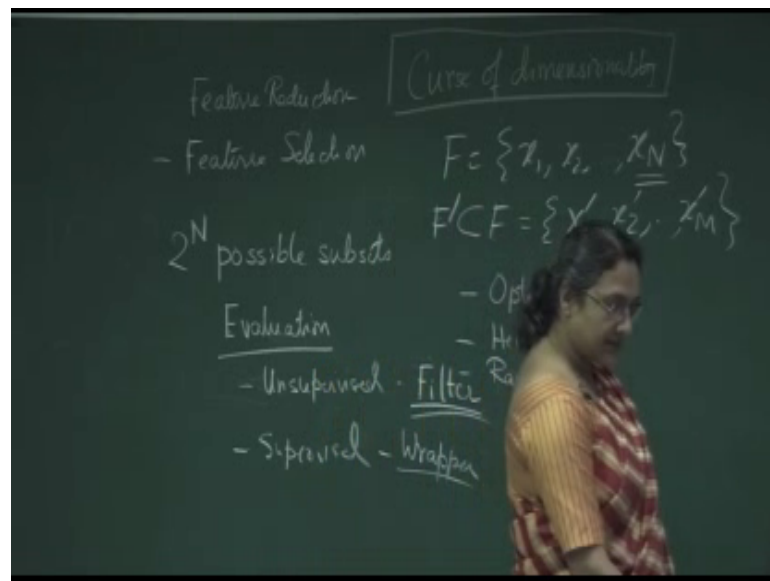
In feature selection what we do is that, given an initial set of features F equal to x_1, x_2, \dots, x_N we are given initially N number of features. We want to find the sub set F' which is a subset of F equal to let us say x'_1, x'_2, \dots, x'_M . We want to find a subset of those features so that it optimizes certain criteria. So, what do we want to optimize. So think about it. We will first talk about the next method of feature reduction which is called Feature Extraction. We will talk about feature extraction in the next class, but what feature extraction does is that it transforms or projects the original set of features into a new subspace which has smaller number of dimension. It will to a projection to M less than N dimensions.

Whereas, in feature selection you select a sub set of the features, will talk about feature extraction in the next class. In both these cases what you are seek into optimize is, you want to either improve or maintain classification accuracy, and you want to simply classifier complexity.

Earlier we saw curve which showed that as you increase the number features the

classification accuracy initially increases and then reduces. It could also be the case that the classification accuracy increases and then remains the same. So, you want to find a small number of features which either improves classification accuracy or may instance the same accuracy and simplifies the complexity of the classifier. These are the reasons why we do feature selection.

(Refer Slide Time: 09:42)

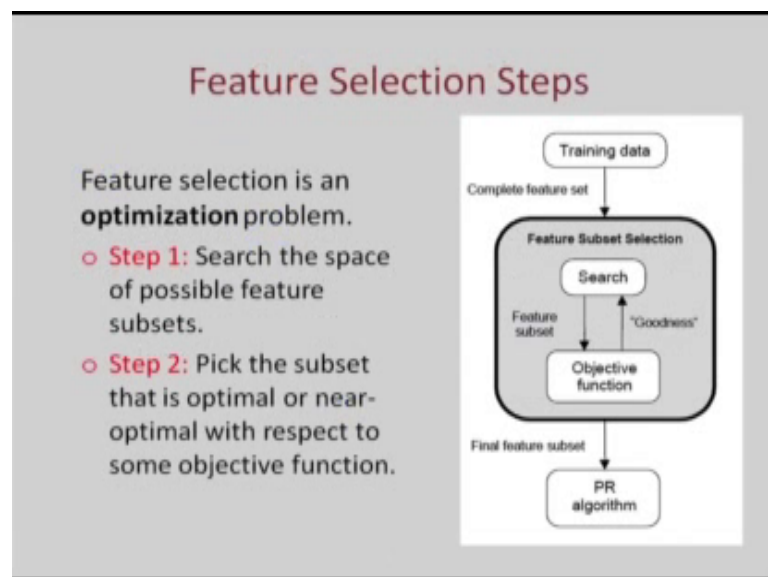


Now, we said that in feature selection we want to select a subset of the original feature set. Now let us see how we can select that subset. So, you can see that if we have N features the number of subsets possible is 2 to the power N , and it is impossible for us to enumerate each of these possible subsets and check how good it is. Because, the number of sub sets is exponentially. So we need a method which works in reasonable time.

The methods that we can use for feature subsets selection can be Optimum methods, can be Heuristic methods, and can be Randomized methods. However, we can use optimum methods if the hypothesis space or the feature subset space has a structure so that we can have a optimum algorithm which works in polynomial time. Otherwise, we can use a heuristic or greedy algorithm or some type of randomized algorithm.

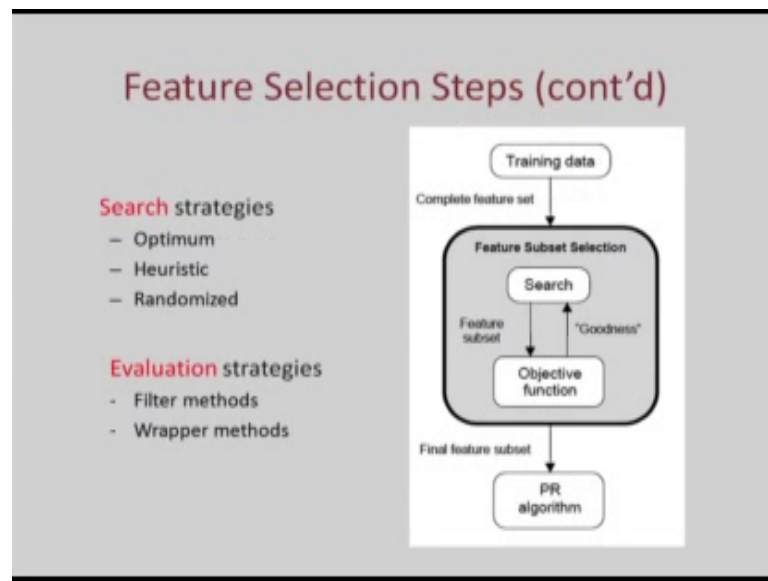
Now, the search algorithm which considers the different feature subsets will also have some mechanism to evaluate it subset. And for evaluation we use two types of methods which I will describe in detail; Unsupervised methods, and Supervised methods. In unsupervised methods, we do not evaluate the subsets over the training examples. We evaluate the information content in somewhere in an unsupervised way these methods are called Filter methods. In supervised methods also called the Wrapper methods, we evaluate the feature subset by using it on a learning algorithm. These are called supervised methods or wrapper methods.

(Refer Slide Time: 12:14)

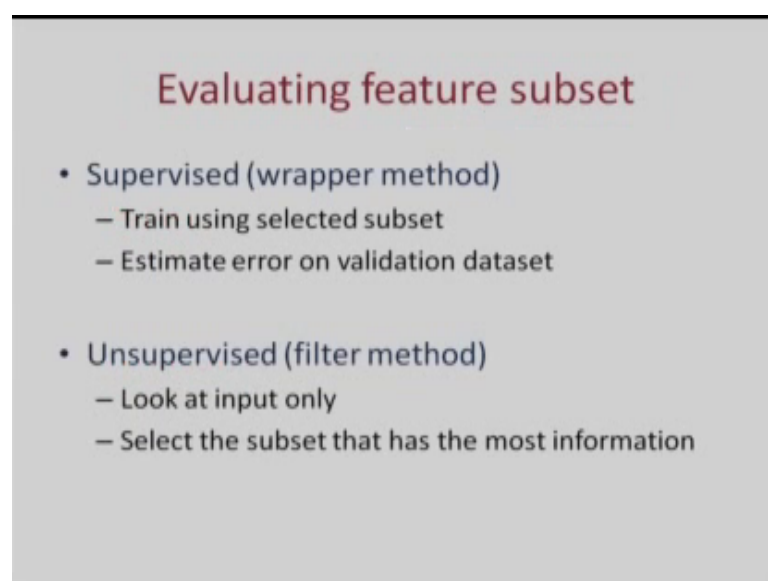


Now let us look at, as I said the features selection is an optimization problem. You have a search algorithm, you have an objective function, and you are trying to optimize the objective function. If you look at this diagram here the search algorithm will select a feature subset and score it on the objective function and find the goodness of the feature subset based on this it will decide which part of the search space to explore next. And after this module is completed you get a final feature sub set and this final feature subset is use by your machine learning or pattern recognition algorithm. So you want to pick the subset that is optimum or near optimal with respect to the objective function.

(Refer Slide Time: 13:05)

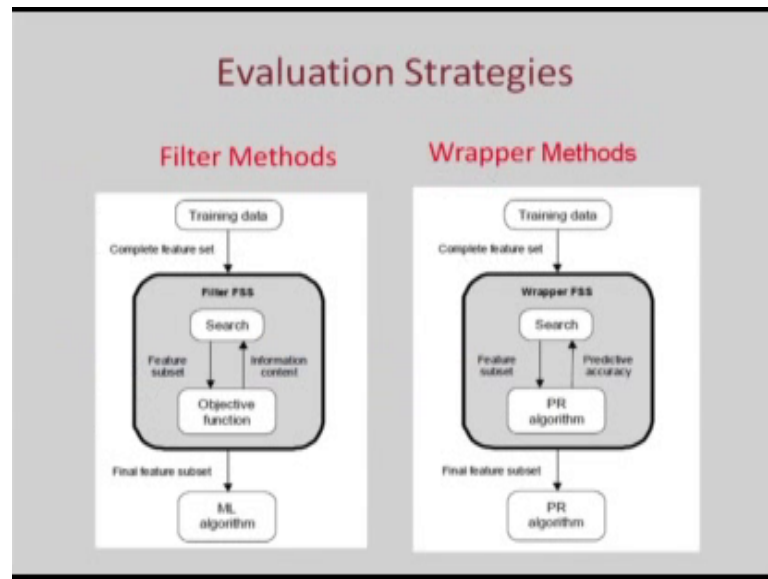


(Refer Slide Time: 13:08)



Now, as a features subset can be evaluated by two methods in supervise methods or wrapper methods we train using the selected subset and we estimates the error on the validation set. In unsupervised or filter methods we look at only the input and we select the subset which has the most information.

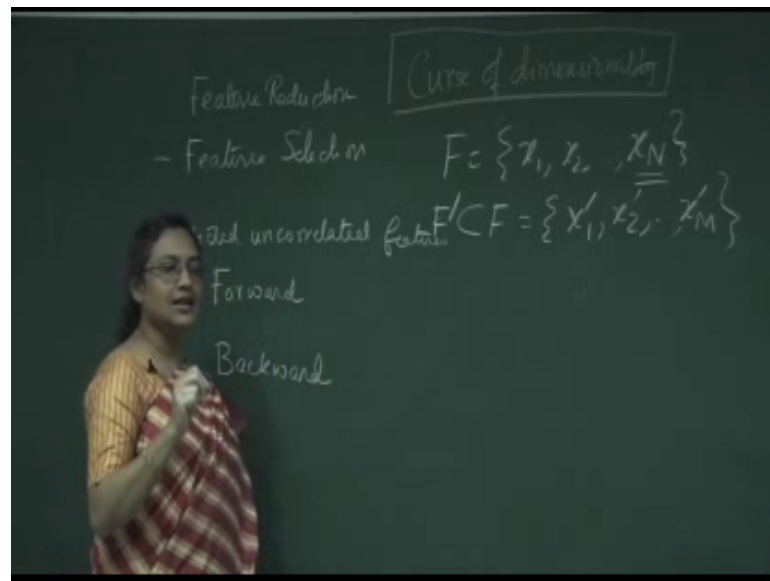
(Refer Slide Time: 13:36)



Now, these two types of methods are illustrated in this picture. In the filter method the search algorithm comes up with a feature subset that is evaluated for information context and based on that search algorithm proceeds and finally this module gives a final features subset. In the wrapper based method or supervised method the search algorithm output a sub features subset which is again used with the pattern recognition or machine learning algorithm and the prediction accuracy is obtained which is fed to the search algorithm. After this whole module is completed you have a feature subset which is used by your machine learning algorithm.

So, these are two different frameworks of feature selection. Now how do you do the feature selection algorithm? First of all what you can do is that, if features are redundant you may use all the features.

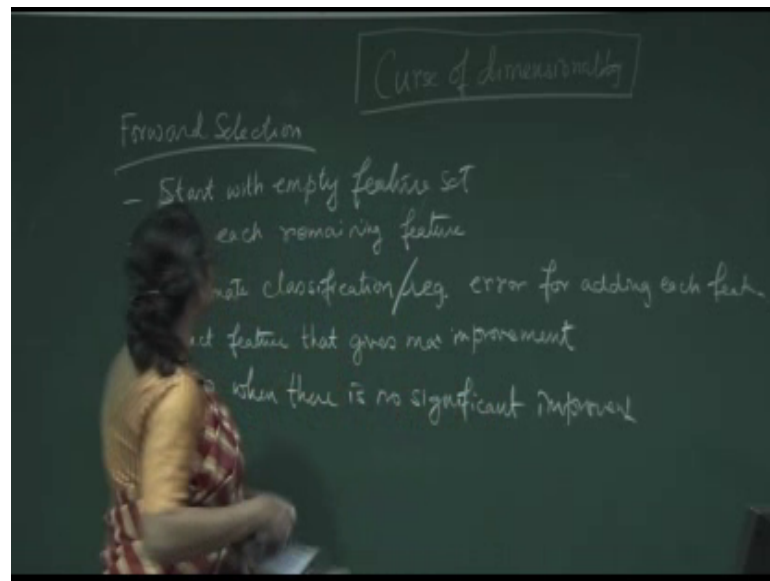
(Refer Slide Time: 14:49)



So, you can find uncorrelated features. You start with some features and then when then when you try to introduce the new feature you do not take that feature, if that feature is highly correlated with another feature which already contents the information about that feature. You select only uncorrelated features. So, you have to select uncorrelated features. And you also have to eliminate irrelevant features.

So, there is some heuristic algorithm that we can use. There are two frameworks for the simple heuristic algorithm for feature subset selection; there is forward selection algorithm, and backward selection algorithm. In forward selection algorithm you start with empty feature set and we add feature one at a time. So, you start from an empty set of features. Let us first write about the forward selection algorithm.

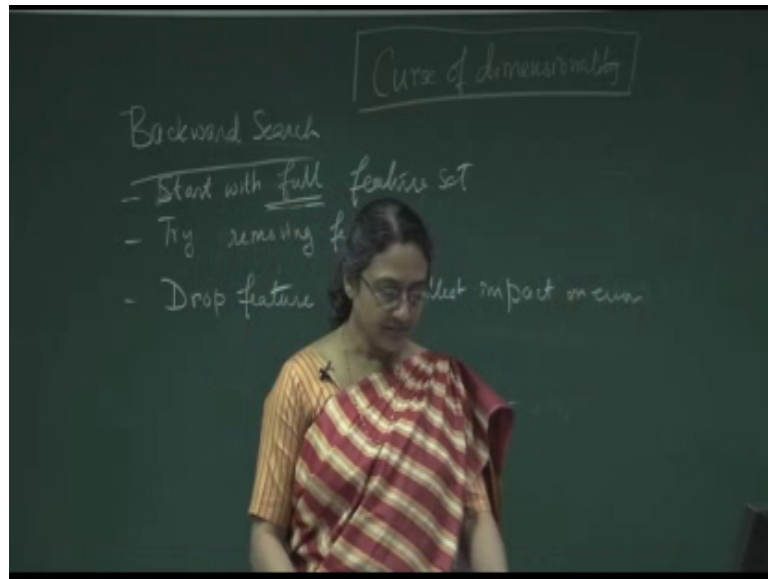
(Refer Slide Time: 16:10)



You start with empty feature set and then you add features one by one. So you try each of the remaining features, and for each of them you estimate the accuracy. You estimate the classification or regression error for adding each specific feature. And you select the feature that gives maximum improvement. To find which feature gives maximum improvement you can use the validation set and not the training set as we have discussed earlier. You can stop when there is no significant improvement.

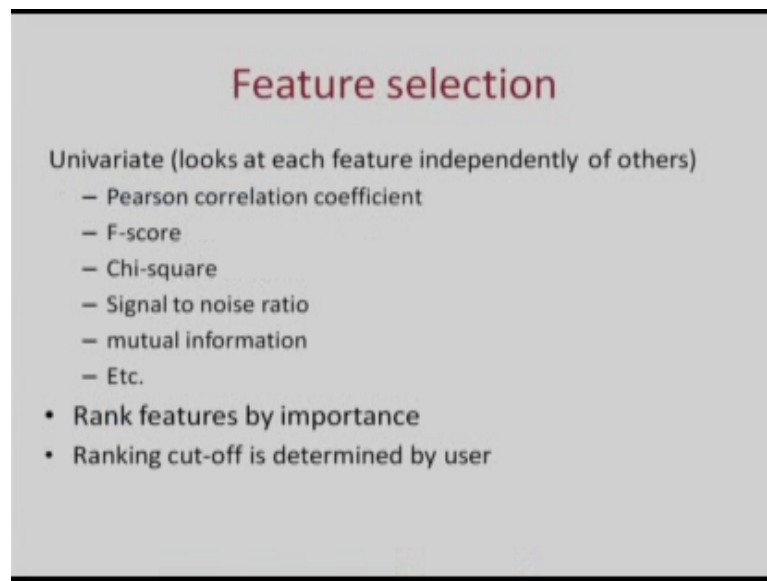
So, it is gradient algorithms we start with empty features set add features one by one. When you add a feature you try to evaluate the result of adding all the features and select that feature whose addition makes the maximum improvement and you continue this process until adding a feature does not give rise to significant improvement. Similarly, you can also have Backward Search.

(Refer Slide Time: 18:40)



In backward search what should you start with? In backward search you start with the full feature set. Then you try removing features from the features that you have. You find that feature whose removal gives rise to maximum improvement in performance. You try removing drop the feature. You drop the feature with the smallest improvement or with smallest impact on error. So, this is a similar algorithm for backward search to find a features subset.

(Refer Slide Time: 20:10)



Feature selection

Univariate (looks at each feature independently of others)

- Pearson correlation coefficient
- F-score
- Chi-square
- Signal to noise ratio
- mutual information
- Etc.

- Rank features by importance
- Ranking cut-off is determined by user

Now, feature selection methods can be Univariate methods which look at one feature at a time or can be Multivariate feature. In univariate method features methods. So why we do is that we look at each feature independently of the other features. For this we can look for some different methods for example; Pearson correlation coefficient, F-score, Chi-square, Signal to noise ratio, mutual information, etcetera. And based on the method that we select we will write the features by importance and the user will select a cut off and we will take the top M features above the cut off or top features, top few features or features which have ranked above the cut off.

(Refer Slide Time: 12:01)

Feature selection

Univariate (looks at each feature independently of others)

- Pearson correlation coefficient
- F-score
- Chi-square
- Signal to noise ratio
- mutual information
- Etc.

Univariate methods measure some type of correlation between two random variables

- the label (y_i) and a fixed feature (x_{ij} for fixed j)

- Rank features by importance
- Ranking cut-off is determined by user

I will just talk about one or two of these methods. But this univariate methods, basically what they do is that the measures some type of correlation between two random variables. In this case, we are measuring the correlation between a particular feature and the target variable, and they keep those features which have higher correlation.

(Refer Slide Time: 21:32)

Pearson correlation coefficient

- Measures the correlation between two variables
- Formula for Pearson correlation =

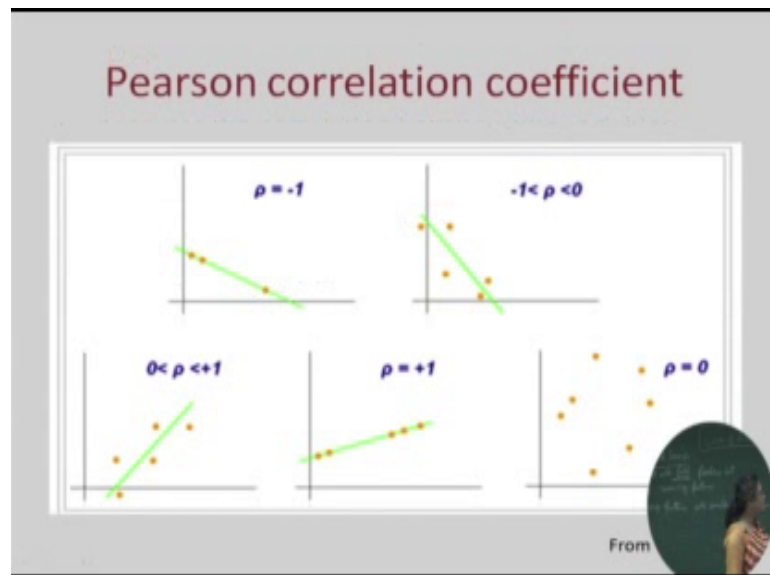
$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}}$$

- The correlation r is between $+1$ and -1 .
 - $+1$ means perfect positive correlation
 - -1 in the other direction

And a Pearson correlation coefficient measures the correlation. Correlation is given by the following formula; r equal to $\frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$. \bar{x} is the average value of the random variable x_i which is a feature, \bar{y} is the average of the random variable y divided by this.

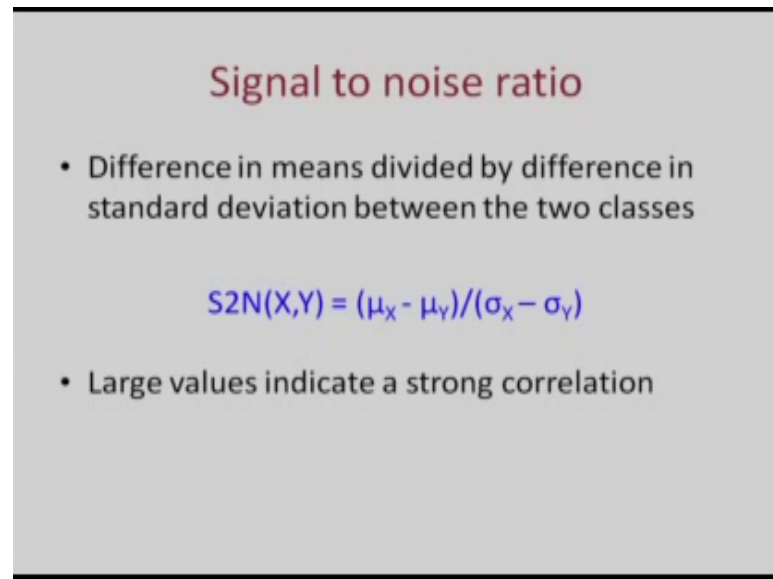
So, the correlation value is between plus 1 and minus 1. Plus 1 means perfect positive correlation minus 1 means perfect negative correlation, both plus and minus 1 or close to plus 1 and minus 1 means highly correlated, closer to 0 means no correlation. So, you want to select features which have high positive or high negative correlation.

(Refer Slide Time: 22:18)



This slide illustrates some points x and y , and for these points correlation is minus 1, for this points correlation is between 0 and minus 1, for these points correlation is between 0 and plus 1, these points correlation is plus 1 and for these points correlation is 0.

(Refer Slide Time: 22:44)



Signal to noise ratio

- Difference in means divided by difference in standard deviation between the two classes

$$S2N(X,Y) = (\mu_x - \mu_y)/(\sigma_x - \sigma_y)$$


- Large values indicate a strong correlation

So, that is Pearson correlation coefficient. I will just briefly mention another method called Signal to noise ratio which measures the difference in means divided by the difference in standard deviation between the two classes. And the formula is given by $\mu_x - \mu_y$ divided by $\sigma_x - \sigma_y$, and large values indicate a strong correlation. For every feature you can find the value of SNR on with y and then sort them according to this value and select the largest values or those values which are above a cut off.

(Refer Slide Time: 23:27)

Multivariate feature selection

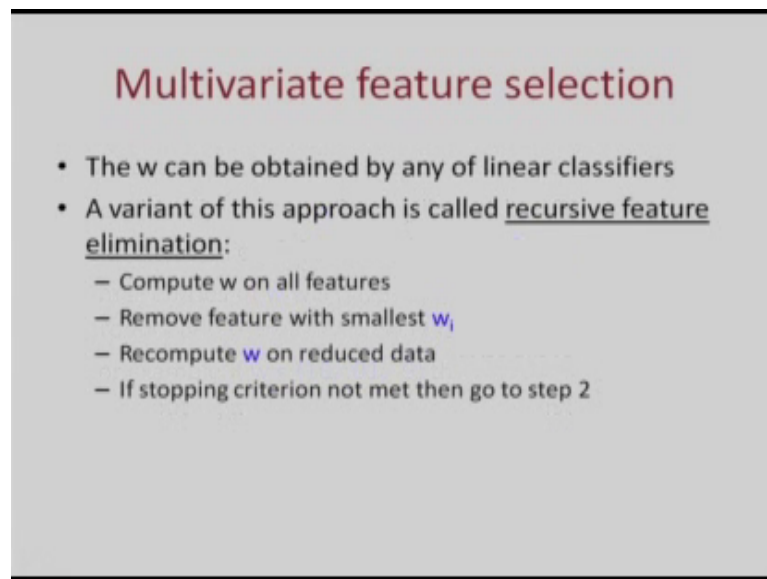
- Multivariate (considers all features simultaneously)
- Consider the vector w for any linear classifier.
- Classification of a point x is given by $w^T x + w_0$.
- Small entries of w will have little effect on the dot product and therefore those features are less relevant.
- For example if $w = (10, .01, -9)$ then features 0 and 2 are contributing more to the dot product than feature 1.
 - A ranking of features given by this w is 0, 2, 1.



In contrast to the univariate selection methods we also have multivariate selection methods. In multivariate selection methods they consider all the features. For example, we have talked about linear regression. Linear regression will find a set of weights. In our class we refer to those weights as beta 0, beta 1, beta 2, beta n which are the coefficients of the n attributes and the value of the bias; for classification of a point is given by this equation of the line and the coefficient metric is given by w transports.

Now, you look at the values of this metric. If w is small the corresponding feature, if we have $w_i x_i$ and w_i is small x_i has less contribution to y , but if w_i is large x_i has large contribution to y , if $x_i w_i$ is small it has small contribution. For example, if w values are 10, 0.01 minus 9 the once whose x the features whose coefficients are 10 and minus 9 they have high contribution to the output. But the one with coefficient 0.01 has no contribution to the output. So we can drop this feature.

(Refer Slide Time: 24:55)



Multivariate feature selection

- The w can be obtained by any of linear classifiers
- A variant of this approach is called recursive feature elimination:
 - Compute w on all features
 - Remove feature with smallest w_i
 - Recompute w on reduced data
 - If stopping criterion not met then go to step 2

This is multivariate feature selection. And the w can be obtained by any of linear classifiers. For example, we have seen a method in our class, a variant of this multivariate features selection approach is called Recursive feature elimination. In recursive feature elimination you compute the w on all the features and you remove the feature with the smallest value of w . Then you re-compute w on the reduce data. So initially, you have N features you remove the feature with the smallest w you are left with N minus 1 features. On this N minus 1 feature you re-compute w by invoking your algorithm again and recursively you keep doing this until the stopping criteria is met. This is about multivariate feature selection.

With this we come to the end of today's lecture. In the next volume, next part we will talk about feature extraction.

Thank you very much.