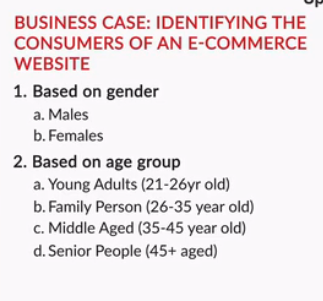
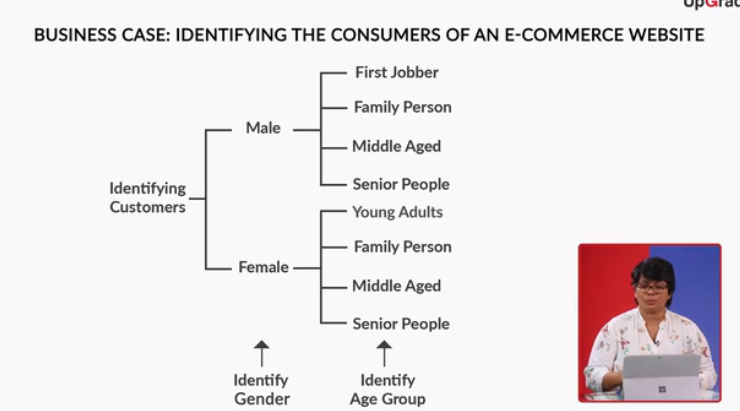
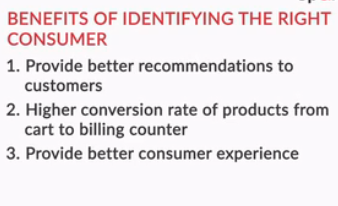
Model selection

Welcome to the session on '**Model Selection**'. Till now, you have learnt various machine learning models such as **logistic regression, support vector machines, decision trees, and random forests**. The question is, **how do you choose**which**model(s)** to apply to a given business problem?

One option is to apply all the models and compare their results; but is it always feasible to apply all the models? Sometimes, you won’t have enough time to try all the available options. More importantly, it is helpful to be able to identify some guiding principles behind the choice of models, rather than using a hit-and-trial approach. We will address this problem in this module.









**Comprehension - Logistic Regression**

You have already learnt about logistic regression in detail. In this segment, you will understand about the **decision boundary** of a logistic regression model.

Recall that the **equation** for a **logistic regression** model is given by

P=11+e−(β0+β1x1+β2x2)P=11+e−(β0+β1x1+β2x2)

In the context of the business problem that you you are going to solve, P denotes the probability of a consumer being **male**,

x1x1 is the attribute - time of the day,

x2x2 is the attribute - ratio of items bought / items added to cart, and

β0β0 is the intercept term, β1β1 and β2β2 are the coefficients of the attributes.

Recall that the above equation can be rewritten in the **log odds** form:

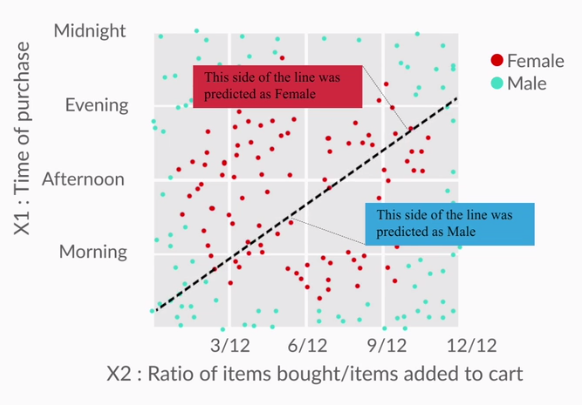
ln(P1−P)=β0+β1x1+β2x2ln(P1−P)=β0+β1x1+β2x2

In the above equation, the term P1−PP1−P is known as the **odds**. Here, the odds indicate the chances of a consumer being male (P) as a proportion of chances of the consumer being female (1-P).

The right hand side of the log odds equation is used to interpret the **decision boundary** of a logistic regression  
model. The gender of the person can be determined by using a threshold value t. Recall that while modelling a  
logistic regression model, you choose a cutoff value c, say 0.5. In a binary classification task(y = 1|0), if P > c, then the predicted value is 1 else 0. You can calculate t using c by substituting c in the following equation:

t=ln(c1−c)t=ln(c1−c)

Since β0+β1x1+β2x2=tβ0+β1x1+β2x2=t represents a straight line, the conditions β0+β1x1+β2x2>tβ0+β1x1+β2x2>t and β0+β1x1+β2x2<tβ0+β1x1+β2x2<t represent two areas divided by the straight line (corresponding to the two classes). Thus, the straight line β0+β1x1+β2x2=tβ0+β1x1+β2x2=t is the **decision boundary**of a logistic regression model which looks something like this**:**



**Decision Boundary of Logistic Regression**

**Comprehension - Logistic Regression**

Suppose you want to identify the gender of the consumers in an ecommerce space and you have two attributes -

1. The time of shopping (X1X1 ), and
2. Ratio of items bought / items added to cart (X2X2 ).

The log odds equation for this problem is given by

ln(P1−P)=β0+β1x1+β2x2ln(P1−P)=β0+β1x1+β2x2

where P is the probability of the consumer being a **male**.

You choose to identify the gender of the consumer using the threshold value (t) of 0.7, and the value of β0+β1x1+β2x2β0+β1x1+β2x2 is 0.4.

What is the gender of the consumer?

Top of Form



**Female**

**Feedback :***The log odds equation can be used to identify the gender by comparing it with the threshold. Compare*β0+β1x1+β2x2β0+β1x1+β2x2*with the threshold. Since 0.4 is less than the threshold of 0.7, the gender is female.*

Bottom of Form

The log odds equation for this problem is given by

ln(P1−P)=β0+β1x1+β2x2ln(P1−P)=β0+β1x1+β2x2

where P is the probability of the consumer being a male.

You choose to identify the gender of the consumer using the threshold value (t) of 0.5. What is the gender of the consumer if the time of the day (x1x1) is 11 and the ratio of items bought/items added to the cart (x2x2) is 0.3.

Use β0=1.2β0=1.2, β1=−0.3β1=−0.3 and β2=9

**Male**

**Feedback :***Substituting*X1X1*and*X2X2*along with*β0β0*,*β1β1*and*β2β2*in the equation*β0+β1x1+β2x2β0+β1x1+β2x2*, we get 0.6. Since 0.6 is greater than 0.5 (threshold value), the consumer is labelled as male.*

# Comparing Different Machine Learning Models - I

You understood the business problem. Now, let’s see how different machine learning models classify the gender of the consumers of an e-commerce website.

The models that you will use are

1. Logistic Regression
2. Decision Trees
3. Support Vector Machine
4. 
5. **A logistic regression model calculates the class probabilities of all the classes of the outcome variable, while predicting a test case.**

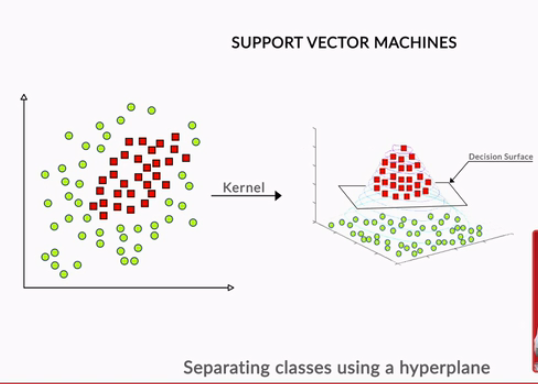
Logistic regression calculates the class probabilities of all the classes present in the outcome variable, using the logistic function. The final class is predicted by providing a cutoff value.

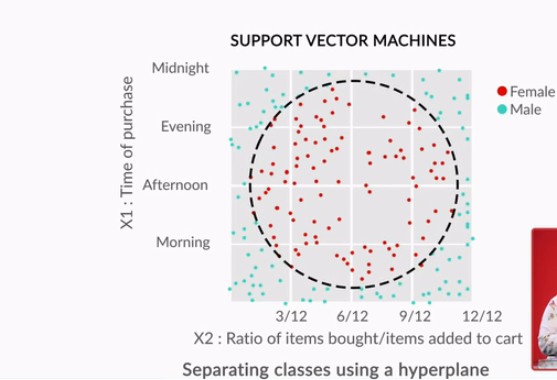
**The decision boundary of an LR model is a straight line.**

The logistic regression model separates two different classes using a line linearly. The sigmoid curve is only used to calculate class probabilities. The final classes are predicted based on the cutoff chosen after building the model.

Decision trees certainly did a better job of differentiating between the two classes. You saw two decision tree models. The first one, with only three decision rules, was a relatively simple tree, while the second one, with 12 decision rules, was a more complex tree. In comparison to the first tree, the second one could potentially **overfit** the training set.

# Comparing Different Machine Learning Models - II





SVM differentiates the two classes almost perfectly, using a hyperplane that looks like a circle when picturing it in a two-dimensional space. In this business problem, SVM does a much better job than a decision tree or logistic regression.

# Pros and Cons of Different Machine Learning Models

## Pros

1. **Logistic regression**
   1. It is convenient for generating probability scores.
   2. Efficient implementation is available across different tools.
   3. The issue of multicollinearity can be countered with regularisation.
   4. It has widespread industry use.
2. **Decision trees**
   1. Intuitive decision rules make it easy to interpret.
   2. Trees handle nonlinear features well.
   3. The variable interaction is taken into account.
3. **Support vector machines**
   1. SVMs can handle large feature space.
   2. These can handle nonlinear feature interaction.
   3. They do not rely on the entire dimensionality of the data for the transformation.

## Cons

1. **Logistic regression**
   1. It does not perform well when the features space is too large.
   2. It does not perform well when there are a lot of categorical variables in the data.
   3. The nonlinear features have to be transformed to linear features in order to efficiently use them for a logistic model.
   4. It relies on entire data i.e. if there is even a small change in the data, the logistic model can change significantly.
2. **Decision trees**
   1. Trees are highly biased towards the training set and overfit it more often than not.
   2. There is no meaningful probability score as the output.
3. **Support vector machines**
   1. SVMs are not efficient in terms of computational cost when the number of observations is large.
   2. It is tricky and time-consuming to find the appropriate kernel for a given data.

Among logistic regression, decision trees and support vector machines, which one is best suited for a dataset having lots of categorical variables?

Decision trees are best suited for a dataset with a lot of categorical data because of the way in which node splitting is performed. Decision trees do not need the categorical features to be converted into numeric features.

Say you work for a large e-commerce company such as Amazon and need to build a classification model to classify a user as likely to buy / unlikely to buy. You have a large number of features and observations, and have to deploy the model in real time.

Compare the pros and cons of logistic regression, decision trees and support vector machines in such a case. Write your arguments and the final choice/approach in the box below.

Suggested Answer

*Cons: 1. Logistic regression might not perform as well as other algorithms in terms of accuracy and other such performance metrics because of the potential nonlinearity in the dataset. 2. Decision trees are prone to overfit the data by creating complex rules which mug up the whole data. 3. Support vector machines might not be appropriate for this task since it requires the model to be deployed in real time, and as discussed earlier, SVMs are resource hungry and slow as compared to other machine learning models. Pros: 1. Since the project is to be deployed in real time, logistic regression and decision trees will be the right choice since they are faster to build than support vector machines. 2. In general, support vector machines give a really good performance as compared to logistic regression or decision trees when the number of features is large. In the end, you have to test and compare all the models in terms of the following - 1. Predictive power (accuracy, sensitivity and specificity, AUC etc.), and 2. Computational cost After analysing the above, you have to choose the model that gives a right balance of both the goals.*

Among logistic regression, decision trees and support vector machines, which one is the least suited for a nonlinear decision boundary?

Logistic regression is a linear model and it can not create a nonlinear boundary.

You could get overwhelmed by the choice of algorithms available for classification. To summarise—

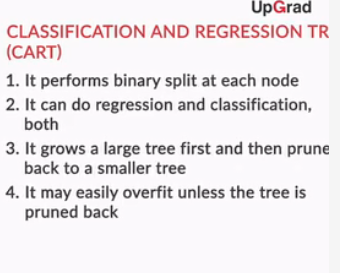
1. **Start with logistic regression**. Using a logistic regression model serves two purposes: 1) It acts as a **baseline** (benchmark) model. 2) It gives you an idea about the important variables.
2. Then, go for **decision trees** and compare their performance with the logistic regression model. If there is no significant improvement in their performance, then just use the important variables drawn from the logistic regression model.

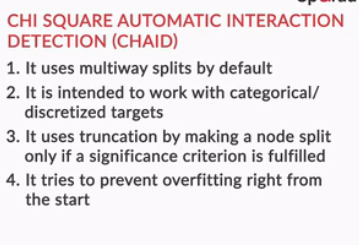
Finally, if you still do not meet the performance requirements, use **support vector machines**. But, keep in mind the **time and resource constraints**, because it takes time to find an appropriate kernel for SVM. Also, they are computationally expensive.

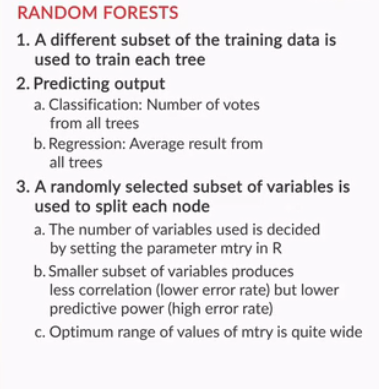
So far, you studied a specific type of tree: **CART (Classification and Regression Trees)**. There is one more tree that is used widely. It is called **CHAID (Chi-square Automatic Interaction Detection)**. Both of these trees have different applications. A **chi-square test** is a statistical hypothesis test where the test statistic is chi-squared distribution. This test is used to compare the interaction of independent variables with the dependent variable.

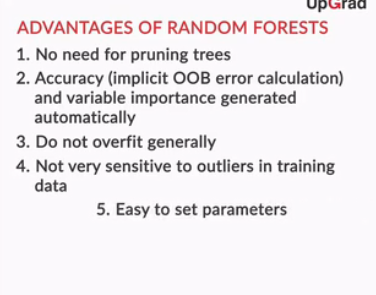
You are already familiar with **CART**, which creates a **binary tree-**a tree with a maximum of two child nodes for any node in the tree. Sometimes CART is not appropriate to visualise the important features in a dataset because binary trees tend to be much **deeper** and more **complex** than a **non-binary tree-** a tree which can have more than two child nodes for any node in the tree. This is where **CHAID** comes in. CHAID can create non-binary trees which tend to be shallower than the binary trees. This makes CHAID trees easier to look at and understand the important drivers (features) in a business problem. The process of finding out important features is also referred to as **driver analysis**.

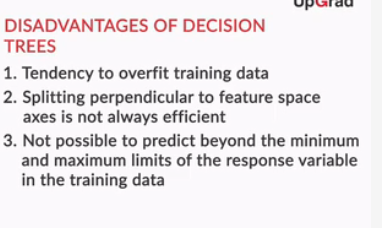
You looked at the different applications of CART and CHAID trees. To put them in the form of an analogy, suppose you are working with the Indian cricket team, and you want to **predict** whether the team will win a particular tournament or not. In this case, **CART** would be more preferable because it is more suitable for prediction tasks. Whereas, if you want to look at the **factors** that are going to influence the win/loss of the team, then a **CHAID** tree would be more preferable.











**Disadvantages of decision trees:**

1. Trees have a tendency to **overfit** the training data.
2. Splitting with **multiple linear decision boundaries that are perpendicular to the feature space**is not always efficient.
3. It is not possible to **predict beyond the range** of the response variable in the training data in a regression problem. Suppose you want to predict house prices using a decision tree and the range of the the house price (response variable) is $5000 to $35000. While predicting, the output of the decision tree will always be within that range.

**Advantages of random forests:**

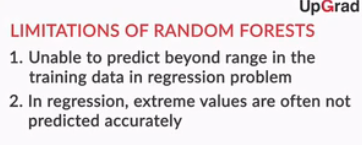
1. No need to **prune** the trees of a forest.
2. The **OOB error** can be calculated from the training data itself which gives a good estimate of the model performance on unseen data.
3. It is hard for a random forest to **overfit**the training data.
4. A random forest is not affected by **outliers** as much because of the aggregation strategy.

Random forests use bagging along with sampling the features randomly at each node split. This prevents them from overfitting the data, unlike decision trees.

There is no need to prune trees in a random forest because even if some trees overfit the training set, it will not matter when the results of all the trees are aggregated.

The limitations of a random forest are:

1. Owing to their origin to decision trees, random forests have the same problem of **not predicting beyond the range of the response variable** in the training set.
2. The **extreme values are often not predicted** because of the aggregation strategy. To illustrate this, let’s take the house prices example, where the response variable is the price of a house. Suppose the range of the price variable is between $5000 and $35000. You train the random forest and then make predictions. While making predictions for an expensive house, there will be some trees in the forest which predict the price of the house as $35000, but there will be other trees in the same forest with values close to $35000 but not exactly $35000. In the end, when the final price is decided by aggregating using the mean of all the predictions of the trees of the forest, the predicted value will be close to the extreme value of $35000 but not exactly $35000. Unless all the trees of the forest predict the house price to be $35000, this extreme value will not be predicted.



So far, you learnt multiple machine learning models. They include

1. Logistic regression
2. Decision trees
3. Support vector machines
4. Types of decision trees
5. Random forests

To summarise, you should start with a **logistic regression** model. Then, build a **decision tree** model. While building a decision tree, you should choose the appropriate method: **CART** for predicting and **CHAID** for driver analysis. If you are not satisfied with the model performance mentioned so far, and you have sufficient time and resources in hand, then go ahead and build more complex models like **random forests** and **support vector machines**.

Starting from a basic model helps in two ways: 1) If the model performs as per requirement, there is no need to go to complex models. This saves time and resources. 2) If it does not perform well, it can be used to benchmark the performance of other models.

# Summary

In this session, you learnt how to **choose an appropriate machine learning model** for a given problem. So now if you have a business problem, you can run different models and **select the optimal model**, under given **time and resource constraints**.

You also learnt the pros and cons of **logistic regression, decision trees, and support vector machines**.

You learnt about two types of trees: **CART (Classification and Regression Tree)**and the**CHAID (Chi-square Automatic Interaction Detection) tree**. You also learnt about the applications of the two, and that **CART** is suitable for **prediction**, while a **CHAID** tree is suitable for **variable analysis**.

You learnt the pros and cons of decision trees and random forests, and when to choose random forests over decision trees.

Finally, you learnt how to go about **model building** — right from starting by building a simple logistic regression model to building complex models such as support vector machines and random forests.

Which of these is most appropriate to use as a baseline (benchmark) model?

Logistic regression is simple to run, with no hyperparameters to tune. It can be used as a benchmark to compare the performance of other models.

Which of the following algorithms is not advisable to use when you have limited CPU resources and time, and when the data set is relatively large?

Support vector machines can take quite a bit of time to run because of their resource-intensive nature.

On a binary classification task, a logistic regression model gives 67% accuracy on the training set, a decision tree model gives 82% accuracy and an SVM model with a linear kernel gives 69% accuracy. What is NOT a likely reason for the superior results of the tree model?

A decision tree generally does not perform well on a dataset with a lot of continuous variables. Since the tree is performing well on the dataset, it is highly unlikely that the data has only continuous attributes.

Given a decision tree model on a binary classification task, which among the following is the best way to check whether the tree is overfitting?

**:**If the difference between training and validation accuracy is significant, then you can conclude that the tree has overfitted the data.

iven a binary classification dataset, say having Y = 1 or 0 as the target variable and X1X1  and X2X2  as two numeric attributes, what does it mean to say that the dataset is linearly separable?

**A straight**aX1+bX2+c=0aX1+bX2+c=0**line can separate the points Y=0 from Y=1**

A dataset is linearly separable when the different classes can be separated using a line. Here, classes 0 and 1 are being separated by the given equation of line.

The equation of log odds is ​ln(P1−P)=β0+β1X1+β2X2ln(P1−P)=β0+β1X1+β2X2​ The right handside of the above equation is a linear line where the log odds of the target variable Y and the attributes X1X1 and X2X2, all lie on the same line.

On a binary classification task, a logistic regression model gives 67% accuracy on the training set, a decision tree model gives 82% accuracy and an SVM model with a linear kernel gives 69% accuracy. What is NOT a likely reason for the superior results of the tree model?