Sampath Kumar Grandhi
sxg180072

1 Nonlinear Regression

a Setup

b Generate Synthetic Data

This plot shows data set $D = (x_i, y_i)_{i=1}^n$ and the true function $f_{true}(x)$

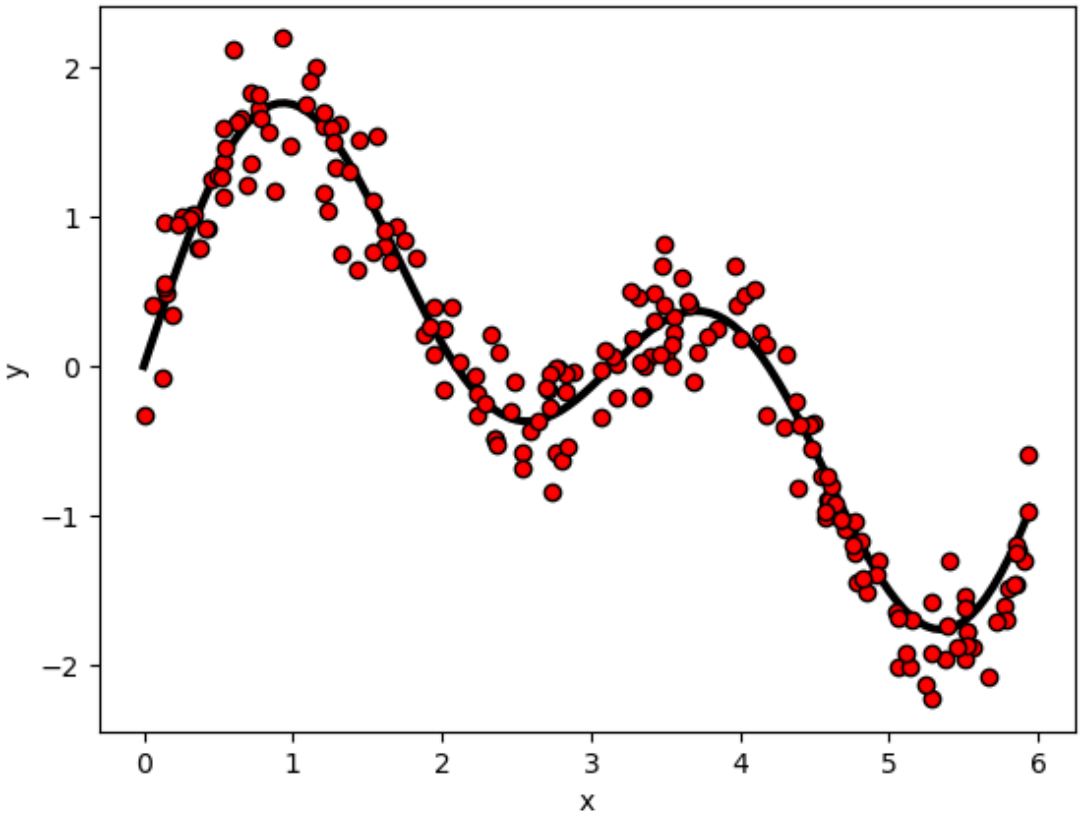


Figure 1: Synthetic Data

c Create Training and Test Sets

d Monomial Basis Functions

e Training and Testing

f Model Selection

As we can see from the observed table values from Figure 2. The value of ϵ_{MSE} changes with different degree of polynomial d. According to the test data we could say that when $d = 6$ is the best as the ϵ_{MSE} is the least compared to other degree d of polynomials. Also if we observe the plotted visualization. We could say that when $d = 6$ the colored line *green* is shown. If we consider Occam's razor "The simplest model that fits the data is also the most plausible" to choose the best model.
In this nonlinear regression example bias-variance could be discussed as we could observe for some higher values of d (higher degree of polynomial) ϵ_{MSE} is very high. The MSE can be decomposed into a bias and

d	e_mse
5	2.075192684876852
6	2.0621799248434756
7	2.206877275855123
8	2.153458789082902
9	2.1476534050428224
10	2.10843865296542
11	1.7379075906390304
12	156.8864720408041
13	172.24523768437092
14	239.46979210425252
15	1716.4957454158564

Figure 2: Table showing test error values observed for different degree of the polynomial

variance term $MSE = Bias^2 + Var$. If the model focuses on minimizing error(variance) then it overfits the data. If the model focuses on shrinking the coefficients (bias) it underfits the data. Therefore balancing bias and variance is crucial. Also we may run into the curse of dimensionality in the case of higher dimensions as computational difficulty increases exponentially. The best model is the one that generalizes well, that is, the best model trades-off effectively between bias and variance and can be expected to perform well on future data.
Discussed with Prof. Gautam Kunapuli and Sri Ram Chappidi.

g Visualization

Please refer the following page.
Discussed with Prof. Gautam Kunapuli and Sri Ram Chappidi.

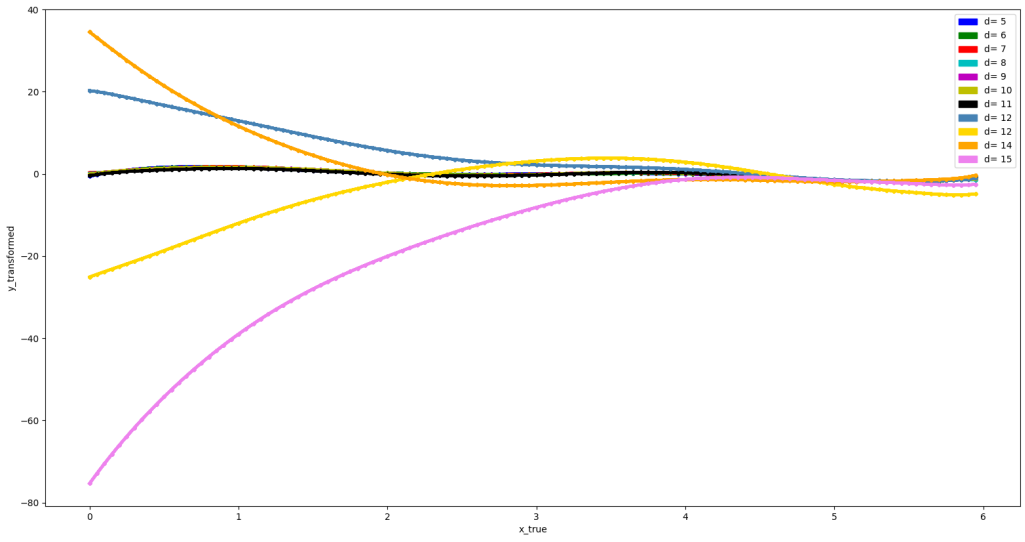


Figure 3: visualisation of nonlinear regression models for d = 5, . . . , 16

2 Kernel Perceptron

a 2d to 3d

Let $\mathbf{x}$ and $\mathbf{z}$ be two points in $\mathbb{R}^2$.
Given to consider the feature transformation from $\mathbb{R}^2$ to $\mathbb{R}^3 : \phi([x_1, x_2]) \rightarrow [x_1^2, \sqrt{2}x_1x_2, x_2^2]$

Given to show that $\phi(\mathbf{x})^T\phi(\mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$

Proof. Solving the left hand side of the proof first. Let $\mathbf{x} = [x_1, x_2]$ and $\mathbf{z} = [z_1, z_2]$
considering the transformation on $\mathbf{x}$ we can say $\phi(\mathbf{x}) = \begin{bmatrix} x_1^2 & \sqrt{2}x_1x_2 & x_2^2 \end{bmatrix}$, so $\phi(\mathbf{x})^T = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}$
applying the transformation on $\mathbf{z}$ so $\phi(\mathbf{z}) = \begin{bmatrix} z_1^2 & \sqrt{2}z_1z_2 & z_2^2 \end{bmatrix}$
$\phi(\mathbf{x})^T\phi(\mathbf{z}) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \begin{bmatrix} z_1^2 & \sqrt{2}z_1z_2 & z_2^2 \end{bmatrix}$
Performing matrix multiplication
$\phi(\mathbf{x})^T\phi(\mathbf{z}) = x_1^2z_1^2 + \sqrt{2}\sqrt{2}x_1x_2z_1z_2 + x_2^2z_2^2$
$\phi(\mathbf{x})^T\phi(\mathbf{z}) = x_1^2z_1^2 + 2x_1x_2z_1z_2 + x_2^2z_2^2$
Therefore $\phi(\mathbf{x})^T\phi(\mathbf{z}) = (x_1z_1 + x_2z_2)^2$

Solving the right hand side of the proof.
Based on our assumption of $\mathbf{x}$ and $\mathbf{z}$ we have $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} z_1 & z_2 \end{bmatrix}$

So we have $\mathbf{x}^T = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
Using them we have $(\mathbf{x}^T\mathbf{z})^2 = (\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} z_1 & z_2 \end{bmatrix})^2$

$(\mathbf{x}^T\mathbf{z})^2 = (x_1z_1 + x_2z_2)^2$
This is the same as the one we have solved for $\phi(\mathbf{x})^T\phi(\mathbf{z})$
So we can say $\phi(\mathbf{x})^T\phi(\mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$

□

Discussed with Prof. Gautam Kunapuli and Sri Ram Chappidi.

b 2d to 4d

Let $\mathbf{x}$ and $\mathbf{z}$ be two points in $\mathbb{R}^2$.
Given to consider the feature transformation from $\mathbb{R}^2$ to $\mathbb{R}^4 : \varphi([x_1, x_2]) \rightarrow [x_1^2, x_1x_2, x_2^2, x_2x_1]$

Given to show that $\varphi(\mathbf{x})^T\varphi(\mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$

Proof. Solving the left hand side of the proof first. Let $\mathbf{x} = [x_1, x_2]$ and $\mathbf{z} = [z_1, z_2]$
considering the transformation on $\mathbf{x}$ we can say $\varphi(\mathbf{x}) = \begin{bmatrix} x_1^2 & x_1x_2 & x_2^2 & x_2x_1 \end{bmatrix}$, so $\varphi(\mathbf{x})^T = \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \\ x_2x_1 \end{bmatrix}$
applying the transformation on $\mathbf{z}$ so $\varphi(\mathbf{z}) = \begin{bmatrix} z_1^2 & z_1z_2 & z_2^2 & z_2z_1 \end{bmatrix}$
$\varphi(\mathbf{x})^T\varphi(\mathbf{z}) = \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \\ x_2x_1 \end{bmatrix} \begin{bmatrix} z_1^2 & z_1z_2 & z_2^2 & z_2z_1 \end{bmatrix}$
Performing matrix multiplication
$\varphi(\mathbf{x})^T\varphi(\mathbf{z}) = x_1^2z_1^2 + x_1x_2z_1z_2 + x_2^2z_2^2 + x_2x_1z_2z_1$
$\varphi(\mathbf{x})^T\phi(\mathbf{z}) = x_1^2z_1^2 + 2x_1x_2z_1z_2 + x_2^2z_2^2$
Therefore $\varphi(\mathbf{x})^T\varphi(\mathbf{z}) = (x_1z_1 + x_2z_2)^2$

Solving the right hand side of the proof.
Based on our assumption of $\mathbf{x}$ and $\mathbf{z}$ we have $\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} z_1 & z_2 \end{bmatrix}$

So we have $\mathbf{x}^T = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
Using them we have $(\mathbf{x}^T\mathbf{z})^2 = (\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} z_1 & z_2 \end{bmatrix})^2$

$(\mathbf{x}^T\mathbf{z})^2 = (x_1z_1 + x_2z_2)^2$
This is the same as the one we have solved for $\varphi(\mathbf{x})^T\varphi(\mathbf{z})$
So we can say $\varphi(\mathbf{x})^T\varphi(\mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$

□

c Linear Combination

Stochastic gradient descent update for the perceptron at step i, where a data point $(\mathbf{x_i}, y_i)$ is used to compute $\mathbf{w_{i+1}} = \mathbf{w_i} + \alpha_i y_i \mathbf{x_i}$ with $\mathbf{w_1} = \mathbf{0}$ Unrolling this expression recursively.
$\mathbf{w_2} = \mathbf{w_1} + \alpha_1 y_1 \mathbf{x_1}$
As we have $\mathbf{w_1} = \mathbf{0}$
$\mathbf{w_2} = \alpha_1 y_1 \mathbf{x_1}$

$\mathbf{w_3} = \mathbf{w_2} + \alpha_2 y_2 \mathbf{x_2}$
As we have $\mathbf{w_2} = \alpha_1 y_1 \mathbf{x_1}$
$\mathbf{w_3} = \alpha_1 y_1 \mathbf{x_1} + \alpha_2 y_2 \mathbf{x_2}$

$\mathbf{w_4} = \mathbf{w_3} + \alpha_3 y_3 \mathbf{x_3}$
As we have $\mathbf{w_3} = \alpha_1 y_1 \mathbf{x_1} + \alpha_2 y_2 \mathbf{x_2}$
$\mathbf{w_4} = \alpha_1 y_1 \mathbf{x_1} + \alpha_2 y_2 \mathbf{x_2} + \alpha_3 y_3 \mathbf{x_3}$

$\mathbf{w_5} = \mathbf{w_4} + \alpha_4 y_4 \mathbf{x_4}$
As we have $\mathbf{w_4} = \alpha_1 y_1 \mathbf{x_1} + \alpha_2 y_2 \mathbf{x_2} + \alpha_3 y_3 \mathbf{x_3}$
$\mathbf{w_5} = \alpha_1 y_1 \mathbf{x_1} + \alpha_2 y_2 \mathbf{x_2} + \alpha_3 y_3 \mathbf{x_3} + \alpha_4 y_4 \mathbf{x_4}$

Similarly this follows for

$$\mathbf{w_{n+1}} = \mathbf{w_n} + \alpha_n y_n \mathbf{x_n}$$

Therefore we can write the above as

$$\mathbf{w_{i+1}} = \sum_{j=1}^{i} \alpha_j y_j \mathbf{x_j}$$

Given α_i is some update constant that depends on the gradient of the loss function and the learning rate. If $\alpha_j = 0$, we can say the training data points are at the *global minimum* and are *correctly classified* as the loss is 0.
Discussed with Sri Ram Chappidi.

d Kernel Perceptron

For the transformation $\phi(x)$ We have

$$\mathbf{w_{i+1}} = \sum_{j=1}^{i} \alpha_j y_j \phi(\mathbf{x_j})$$

For the new data point $\mathbf{x_{test}}$

$\mathbf{w}_{i+1} = \alpha_1 y_1 \phi(\mathbf{x_1}) + \alpha_2 y_2 \phi(\mathbf{x_2}) + \alpha_3 y_3 \phi(\mathbf{x_3}) + ... + \alpha_i y_i \phi(\mathbf{x_i})$
For a matrix $\mathbf{A}$ we know that $(\alpha\mathbf{A})^T = \alpha\mathbf{A}^T$
So for $\mathbf{w}_{i+1}^T = \alpha_1 y_1 \phi(\mathbf{x_1})^T + \alpha_2 y_2 \phi(\mathbf{x_2})^T + \alpha_3 y_3 \phi(\mathbf{x_3})^T + ... + \alpha_i y_i \phi(\mathbf{x_i})^T$
So $\mathbf{w}_{i+1}^T \phi(\mathbf{x_{test}}) = \alpha_1 y_1 \phi(\mathbf{x_1})^T \phi(\mathbf{x_{test}}) + \alpha_2 y_2 \phi(\mathbf{x_2})^T \phi(\mathbf{x_{test}}) + \alpha_3 y_3 \phi(\mathbf{x_3})^T \phi(\mathbf{x_{test}}) + ... + \alpha_i y_i \phi(\mathbf{x_i})^T \phi(\mathbf{x_{test}})$

From part *a* of this exercise we have $\phi(\mathbf{x})^T\phi(\mathbf{z}) = (\mathbf{x}^T\mathbf{z})^2$
Similarly transforming $\phi(\mathbf{x_i})^T$ as $\phi(\mathbf{x})^T$ and $\phi(\mathbf{x_{test}})$ as $\phi(\mathbf{z})$
$\mathbf{w}_{i+1}^T \phi(\mathbf{x_{test}}) = \alpha_1 y_1 (\mathbf{x_1}^T \mathbf{x_{test}})^2 + \alpha_2 y_2 (\mathbf{x_2}^T \mathbf{x_{test}})^2 + \alpha_3 y_3 (\mathbf{x_3}^T \mathbf{x_{test}})^2 + ... + \alpha_i y_i (\mathbf{x_i}^T \mathbf{x_{test}})^2$
This can be written as $\mathbf{w}_{i+1}^T \phi(\mathbf{x_{test}}) = \sum_{j=1}^{i} \alpha_j y_j (\mathbf{x_j}^T \mathbf{x_{test}})^2$

For the transformation $\varphi(x)$ We have

$$\mathbf{w_{i+1}} = \sum_{j=1}^{i} \alpha_j y_j \varphi(\mathbf{x_j})$$

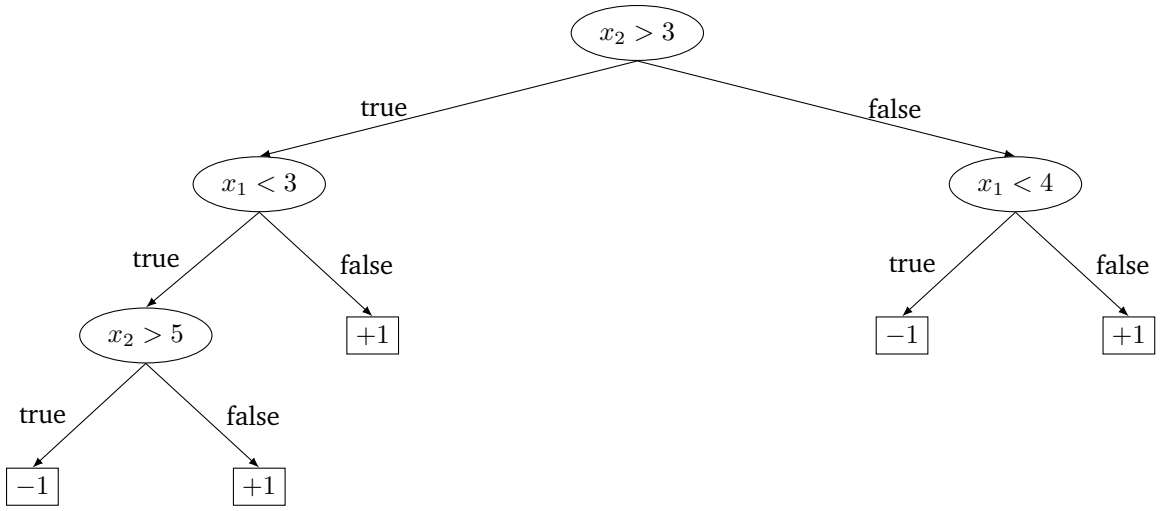
For the new data point $\mathbf{x_{test}}$

$\mathbf{w}_{i+1} = \alpha_1 y_1 \varphi(\mathbf{x_1}) + \alpha_2 y_2 \varphi(\mathbf{x_2}) + \alpha_3 y_3 \varphi(\mathbf{x_3}) + ... + \alpha_i y_i \varphi(\mathbf{x_i})$
For a matrix $\mathbf{A}$ we know that $(\alpha \mathbf{A})^T = \alpha \mathbf{A}^T$
So for $\mathbf{w}_{i+1}^T = \alpha_1 y_1 \varphi(\mathbf{x_1})^T + \alpha_2 y_2 \varphi(\mathbf{x_2})^T + \alpha_3 y_3 \varphi(\mathbf{x_3})^T + ... + \alpha_i y_i \varphi(\mathbf{x_i})^T$
So $\mathbf{w}_{i+1}^T \varphi(\mathbf{x_{test}}) = \alpha_1 y_1 \varphi(\mathbf{x_1})^T \varphi(\mathbf{x_{test}}) + \alpha_2 y_2 \varphi(\mathbf{x_2})^T \varphi(\mathbf{x_{test}}) + \alpha_3 y_3 \varphi(\mathbf{x_3})^T \varphi(\mathbf{x_{test}}) + ... + \alpha_i y_i \varphi(\mathbf{x_i})^T \varphi(\mathbf{x_{test}})$

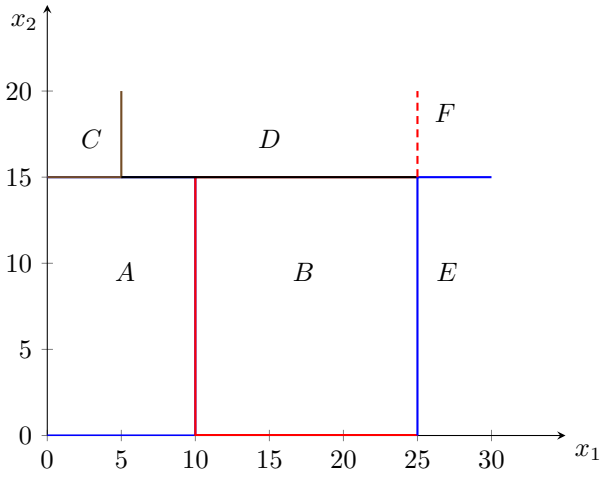
From part *b* of this exercise we have $\varphi(\mathbf{x})^T \varphi(\mathbf{z}) = (\mathbf{x}^T \mathbf{z})^2$
Similarly transforming $\varphi(\mathbf{x_i})^T$ as $\varphi(\mathbf{x})^T$ and $\varphi(\mathbf{x_{test}})$ as $\varphi(\mathbf{z})$
$\mathbf{w}_{i+1}^T \varphi(\mathbf{x_{test}}) = \alpha_1 y_1 (\mathbf{x_1}^T \mathbf{x_{test}})^2 + \alpha_2 y_2 (\mathbf{x_2}^T \mathbf{x_{test}})^2 + \alpha_3 y_3 (\mathbf{x_3}^T \mathbf{x_{test}})^2 + ... + \alpha_i y_i (\mathbf{x_i}^T \mathbf{x_{test}})^2$
This can be written as $\mathbf{w}_{i+1}^T \varphi(\mathbf{x_{test}}) = \sum_{j=1}^{i} \alpha_j y_j (\mathbf{x_j}^T \mathbf{x_{test}})^2$
Therefore we can say that $\mathbf{w}_{i+1}^T \phi(\mathbf{x_{test}})$ and $\mathbf{w}_{i+1}^T \varphi(\mathbf{x_{test}})$ are equal.
Discussed with Prof. Gautam Kunapuli and Sri Ram Chappidi.

3 Decision Trees

a Interpreting a decision tree

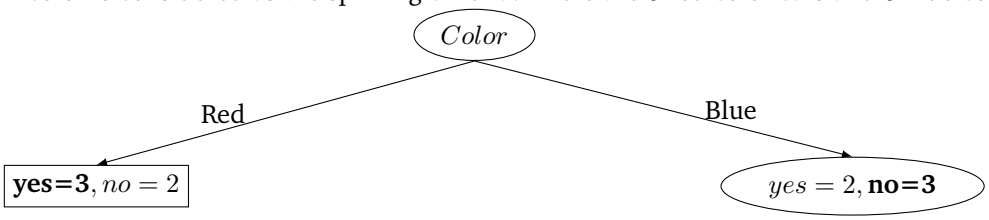


b Visualizing a decision tree



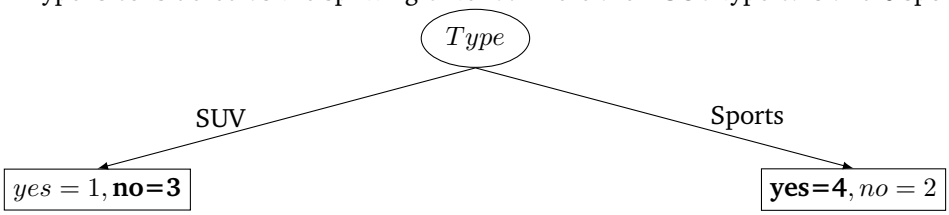
c Learning a decision tree

Using the given data set, we predict if a car is going to be bought based on three features: its color, type and origin. Given to draw the decision tree if we use *accuracy* as the splitting criteria.
If color is considered as the splitting criteria. There are 5 red color cars and 5 blue color cars.



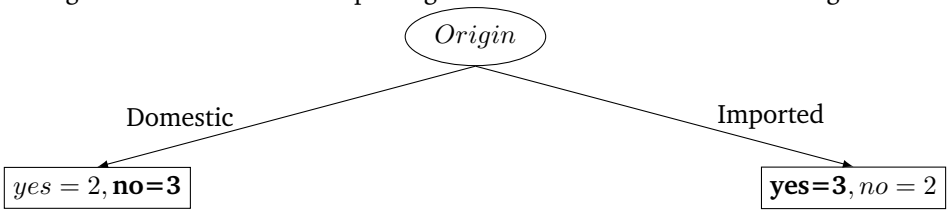
By majority voting the expected value if the color of the Car is Red is Yes and the expected value is No if the color of the car is Blue.
Errors generated when color of the Car is red $= 2$
Errors generated when color of the Car is blue $= 2$
Total error $= 2 + 2 = 4$
Overall error = $number\ of\ mistakes/total$
overall error $= 4/10 = 0.4$
Overall accuracy $= 1 - overall\ error = 0.6$
Accuracy percentage $= 0.6 * 100 = 60\%$
Therefore the accuracy of the decision tree if Color is selected as a splitting criterion is 60%

If Type is considered as the splitting criteria. There are 4 SUV type cars and 6 sports type cars.

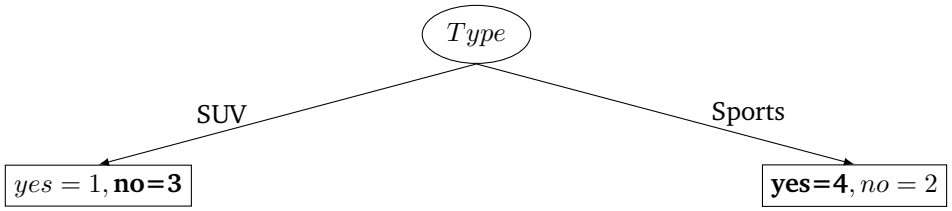


By majority voting the expected value if the type of the Car is SUV is No and the expected value is Yes if the type of the car is Sports.
Errors generated when type of the Car is SUV $= 1$
Errors generated when type of the Car is Sports $= 2$
Total error $= 1 + 2 = 3$
Overall error = $number\ of\ mistakes/total$
overall error $= 3/10 = 0.3$
Overall accuracy $= 1 - overall\ error = 0.7$
Accuracy percentage $= 0.7 * 100 = 70\%$
Therefore the accuracy of the decision tree if Type is selected as a splitting criterion is 70%

If Origin is considered as the splitting criteria. There are 5 domestic origin and 5 imported cars.



By majority voting the expected value if origin of the Car is Domestic is No and the expected value is Yes if the origin of the car is Imported.
Errors generated when origin of the Car is domestic $= 2$
Errors generated when origin of the Car is imported $= 2$
Total error $= 2 + 2 = 4$
Overall error = $number\ of\ mistakes/total$
overall error $= 4/10 = 0.4$
Overall accuracy $= 1 - overall\ error = 0.6$
Accuracy percentage $= 0.6 * 100 = 60\%$
Therefore the accuracy of the decision tree if origin of the car is selected as a splitting criterion is 60%
We can see that accuracy is highest (70%) when *type* of the car is selected as the splitting criterion.
The decision tree learned on this criterion is



Discussed with Prof. Gautam Kunapuli.

d Analyzing a decision tree

Overfitting is a practical difficulty in machine learning models as the learning algorithm continues to develop hypotheses that reduce training set error at the cost of increased set error. Overfitting is the phenomenon in which the learning system tightly fits the given training data so much that it would be inaccurate in predicting the outcomes of the untrained data. In decision trees, overfitting happens when the tree is designed so as to perfectly fit all samples in the training data set. Thus it ends up with branches with strict rules of sparse data. Therefore this effects the accuracy when predicting samples that are not part of the training set. *Decision trees will always overfit!* as It is always possible to obtain zero training error on the input data with a deep enough tree (if there is no noise in the labels). Random noise or outliers in the training examples also leads to overfitting. Few of the methods used to avoid overfitting in decision tree are pruning and early stopping.