Mining Environment Assusmptions for Cyber-Physical System Models

Authors:- Sara Mohammadinejad, Jyotirmoy V. Deshmukh, Aniruddh G. Puranic

Project Presentation, CS637A, 2021-22, Sem-I

Instructor:- Dr. Indranil Saha (CSE, IITK)

Group 8:-

- 21111006 Akash G. Panzade
- 21111041 Mayuresh D. Shandilya
- 21111047 Prajwal P. Thakare

Research Paper:-https://ieeexplore.ieee.org/document/9096037

What are Environment Assumptions?

- Autonomous cyber-physical systems can often be modeled as a system consisting of heterogeneous components. Each of these components could itself be complex.
- A large subset of input signals for which the corresponding output signals satisfy the output requirements.
- This subset can be compactly described using an STL (Signal Temporal Logic)formula which is known as Environment Assumption.
- Basically, Environment Assumptions are the conditions in Temporal logic on input traces which needs to be satisfied in order to get desirable output.

Background Idea !

- Given an output requirement, what are the assumptions on the model environment, i.e., input traces to the model, that guarantee that the corresponding output traces satisfy desirable conditions.

- Now, If we know what are the restrictions on the input signals we can easily model a system which can monitor these restrictions.

- Also, we can enumerate these conditions which are giving low accuracy to modify our model.

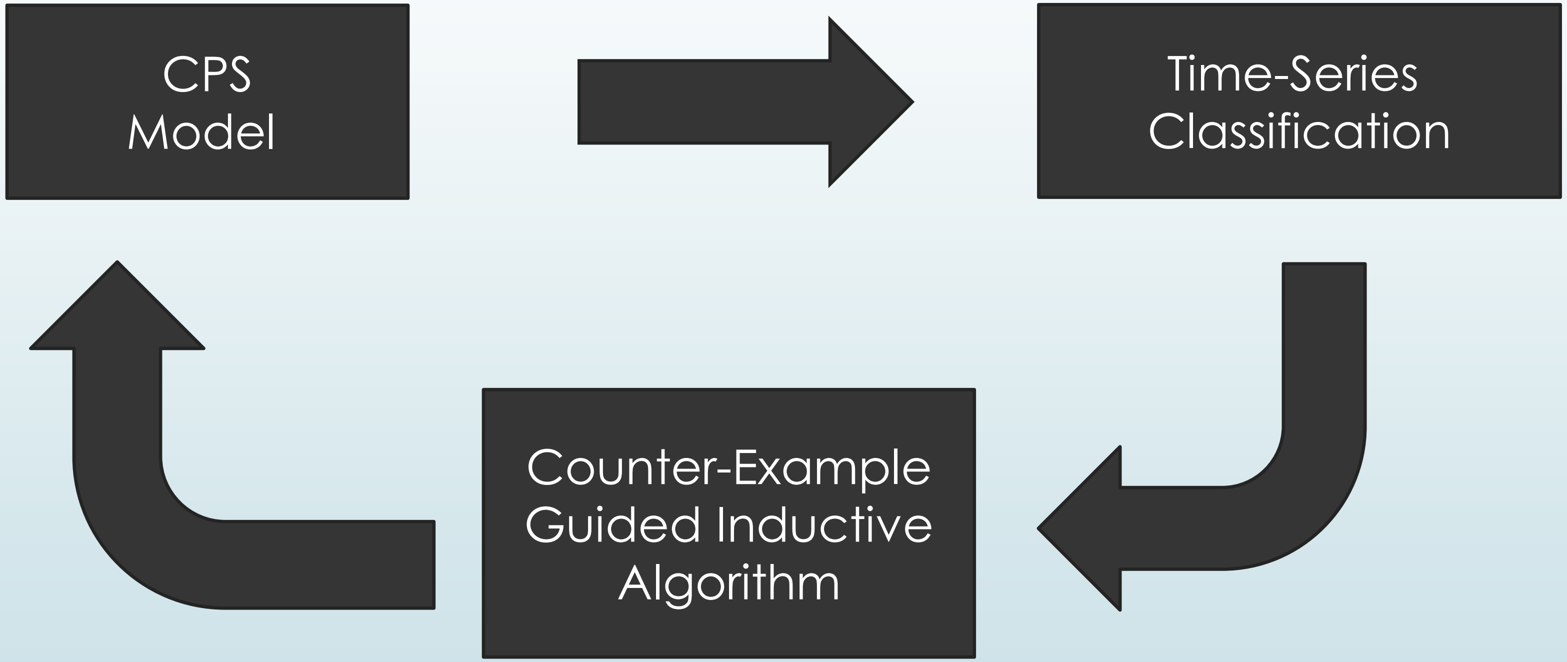
Model Representation:-

Input Assumptions → Cyber Physical System Component → STL Formula for the Assumptions

Approach of the Paper:-

- Firstly, we will mine the conditions based on input assumptions by using **Environment Assumption Mining Algorithm** which uses decision tree based approach of supervised learning.
- Then, we will traverse the tree to the leaves containing derirable class of output which in-turn gives an STL formula.
- In next step, Counterexample-guided inductive synthesis algorithm systematically enumerates parametric STL (PSTL) formulas, and attempts to find parameter valuations such that the resulting formula classifies the input traces with high accuracy.
- Falsification procedure checks if there exists an input trace to the model that satisfies input conditions but the corresponding output does not match with the actual output.
- These traces filtered by Flasification will be fed to CEGIS.

High-level View:-



Environment Assumption Mining Algorithm:-

- Initially, we randomly sample input traces (Line 1 and label them as good or bad (resp. Lines 3,4) depending on whether their corresponding outputs satisfy the given φout.

- At the beginning of the while-loop, we assume that there is a PSTL formula ψ which is considered as environment assumption.

- The first time the loop body is executed, this enumeration occurs in Line 5, otherwise a new PSTL formula is obtained in the loop in Line 16.

- Once we have a candidate PSTL formula ψ proposed, we use a supervised learning approach to obtain a decision tree Y[ψ proposed] from ψ proposed using supervised learning based decision tree.

Input: Input signal domain U, Output requirement φ_{out}, Input signal time domain $T(\mathbf{u})$, Model $M = (\mathbf{u}, \mathbf{y})$, Simulation Budget N for Falsification, Formula length limit $\ell_{\max}$, Classification Accuracy $1 - \epsilon$

Output: Environment Assumption φ_{in}

1 $\mathcal{T}$ = Sample input traces from U using time instants from $T(\mathbf{u})$
2 **foreach** $\mathbf{u}(t) \in \mathcal{T}$ **do**
3   **if** $M(\mathbf{u}(t)) \models \varphi_{\text{out}}$ **then** $\mathcal{T}_{\text{good}} = \mathcal{T}_{\text{good}} \cup \{\mathbf{u}(t)\}$
4   **else** $\mathcal{T}_{\text{bad}} = \mathcal{T}_{\text{bad}} \cup \{\mathbf{u}(t)\}$
5 ψ^{proposed} = EnumerateNextPSTL()
6 **while** $|\psi^{\text{proposed}}| < \ell_{\max}$ **do**
7   (accuracy, $\Upsilon[\psi^{\text{proposed}}]$) = DecisionTreeBasedSTLClassifier($\psi^{\text{proposed}}, \mathcal{T}_{\text{good}}, \mathcal{T}_{\text{bad}}$)
8   $\varphi_{\text{in}}^{\text{proposed}}$ = GetSTL($\Upsilon[\psi^{\text{proposed}}]$)
9   **if** *accuracy* $> 1 - \epsilon$ **then**
10     $\text{cex}(t)$ = Falsify($\mathbf{y} \models \varphi_{\text{out}}, N$)
11   subject to $\mathbf{u}(t) \models \varphi_{\text{in}}^{\text{proposed}}$
12         $\mathbf{y}(t) = M(\mathbf{u}(t))$
13   **if** $\text{cex}(t) \neq \emptyset$ **then** $\mathcal{T}_{\text{bad}} = \mathcal{T}_{\text{bad}} \cup \{\text{cex}(t)\}$
14   **else return** $\varphi_{\text{in}}^{\text{proposed}}$
15   **else**
16   ψ^{proposed} = EnumerateNextPSTL()

Environment Assumption Mining Algorithm (Contd.):-

- Now, we will filter the counter examples by using Falsify method.

- Falsify method will take out those traces whose input signals are satisfying the STL formula but their output labels are not matching with their actual output class.

- Now, these traces will be fed to the model to have a better accuracy over the traces and get a stronger STL formula.

Input: Input signal domain U, Output requirement φ_{out}, Input signal time domain $T(\mathbf{u})$, Model $M = (\mathbf{u}, \mathbf{y})$, Simulation Budget N for Falsification, Formula length limit $\ell_{\max}$, Classification Accuracy $1 - \epsilon$

Output: Environment Assumption φ_{in}

$\mathcal{T}$ = Sample input traces from U using time instants from $T(\mathbf{u})$

foreach $\mathbf{u}(t) \in \mathcal{T}$ **do**

  if $M(\mathbf{u}(t)) \models \varphi_{\text{out}}$ **then** $\mathcal{T}_{\text{good}} = \mathcal{T}_{\text{good}} \cup \{\mathbf{u}(t)\}$

  else $\mathcal{T}_{\text{bad}} = \mathcal{T}_{\text{bad}} \cup \{\mathbf{u}(t)\}$

ψ^{proposed} = EnumerateNextPSTL()

while $|\psi^{\text{proposed}}| < \ell_{\max}$ **do**

  (accuracy, $\Upsilon[\psi^{\text{proposed}}]$) = DecisionTreeBasedSTLClassifier($\psi^{\text{proposed}}, \mathcal{T}_{\text{good}}, \mathcal{T}_{\text{bad}}$)

  $\varphi_{\text{in}}^{\text{proposed}}$ = GetSTL($\Upsilon[\psi^{\text{proposed}}]$)

  if *accuracy* $> 1 - \epsilon$ **then**

    cex(t) = Falsify($\mathbf{y} \models \varphi_{\text{out}}, N$)

      subject to $\mathbf{u}(t) \models \varphi_{\text{in}}^{\text{proposed}}$

        $\mathbf{y}(t) = M(\mathbf{u}(t))$

    if cex$(t) \neq \emptyset$ **then** $\mathcal{T}_{\text{bad}} = \mathcal{T}_{\text{bad}} \cup \{\text{cex}(t)\}$

    else **return** $\varphi_{\text{in}}^{\text{proposed}}$

  else

    ψ^{proposed} = EnumerateNextPSTL()

Classification Using Decision Tree:-

- In this algorithm, firstly we will split the data into test and train data (line2).
- Then we will be selecting features of the training data by using robustness value of parameters.
- Now, we will train the decision tree using the selected features and compute the accuracy of the trained model using testing data.
- The decision tree and the computed accuracy will be returned to the main algorithm.

Input: $\psi, \mathcal{T}_{\text{good}}, \mathcal{T}_{\text{bad}}$
Output: accuracy, $\Upsilon[\psi]$

1 **Function** DecisionTreeBasedSTLClassifier($\psi, \mathcal{T}_{\text{good}}, \mathcal{T}_{\text{bad}}$):
`// Split data for train and test`
2 $\mathcal{T}_{\text{train}}, \mathcal{T}_{\text{test}} :=$ split($\mathcal{T}_{\text{good}} \cup \mathcal{T}_{\text{bad}}, 0.7$)
`// Compute robustness values as features for training`
3 $\mu_{\text{train}} :=$ computeFeatures($\mathcal{T}_{\text{train}}, \psi, \mathcal{D}_{\mathcal{P}}(\psi)$)
4 **foreach** $\mathbf{u}(t) \in (\mathcal{T}_{\text{test}} \cup \mathcal{T}_{\text{train}})$ **do**
5 $\ell(\mathbf{u}(t)) := (\mathbf{u}(t) \in \mathcal{T}_{\text{good}})$
`// Train decision tree using computed features`
6 $\Upsilon[\psi^{\text{proposed}}] :=$ TrainDecisionTree($\mu, \ell(\mu)$)
`// Compute accuracy`
7 **foreach** $\mathbf{u}(t) \in \mathcal{T}_{\text{test}}$ **do**
8 $\mu_{\text{test}}(\mathbf{u}(t)) :=$ computeFeatures($\mathcal{T}_{\text{test}}, \psi, \mathcal{D}_{\mathcal{P}}(\psi)$)
9 $\ell'(\mathbf{u}(t)) := \Upsilon[\psi^{\text{proposed}}](\mu_{\text{test}}(\mathbf{u}(t)))$
10 accuracy := computeAccuracy(ℓ, ℓ')
11 **return** accuracy, $\Upsilon[\psi]$

12 **Function** computeFeatures($\mathcal{T}, \psi, \mathcal{D}_{\mathcal{P}}(\psi)$):
`// Sample m parameter values`
13 $\mathcal{D}_{\mathcal{P}m} \leftarrow$ gridSample($\mathcal{D}_{\mathcal{P}}, m$)
14 **foreach** $\mathbf{u}(t) \in \mathcal{T}$ **do**
15 **for** $i \in [1, m]$ **do**
16 $\psi_i := \psi(\nu(\mathcal{D}_{\mathcal{P}m}(i)))$
17 $\mu(\mathbf{u}(t))[i] := \rho(\psi_i, \mathbf{u}, 0)$;
18 **return** μ

Decision Tree to STL Formula:-

- For the STL to be calculated, we will traverse every path of the decision tree till its leaves which satisfy the desired outputs.

- Now, we will do conjunction of all these Logical conditions (based on their input signal traces) lying in the path from the root of tree to the leaf having the desired output.

- We will continue this till we cover all the leaves having desired output.

- Now, we will do logical disjunction of the all above formulas with same output status.

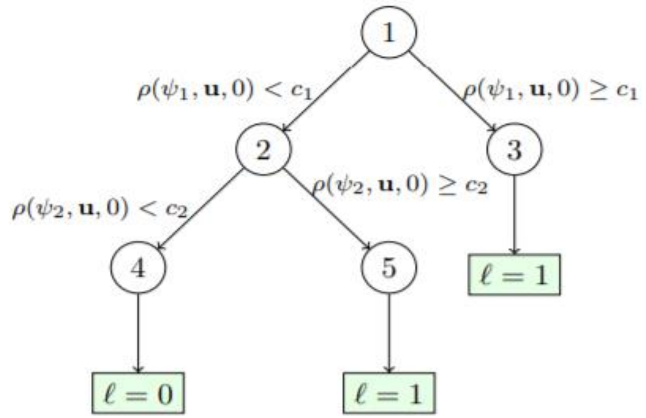


Fig. 3: Example Tree returned by DecisionTreeBasedSTLClassifier

Example:-

- STL formula can be derived from above tree as follows:

I. $\Phi1=\rho(\psi1, u, 0) < c1$

   $\Phi2=\rho(\psi2, u, 0) < c2$

   $\Phi(0)= \Phi1 \wedge \Phi2$ this will be the formula for the label l=0.

I. $\Phi1 =\rho(\psi1, u, 0) < c1$

   $\Phi2 =\rho(\psi2, u, 0) \geq c2$

   $\Phi3 =\rho(\psi1, u, 0) \geq c1$

   $\Phi(1)= (\Phi1 \wedge \Phi2) \vee \Phi3$

   This will be formula for the label l=1.

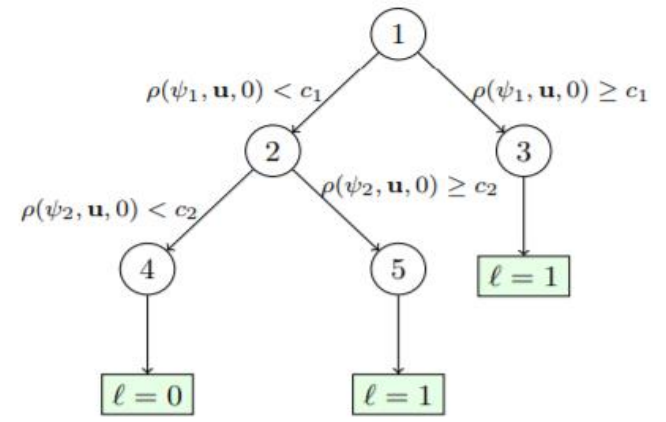


Fig. 3: Example Tree returned by DecisionTreeBasedSTLClassifier

Mining Environment Assumptions on ECG Five days Dataset

- The ECG Five days data set consists of echo-cardiogram signals recorded from a 67 year old male.

- The two classes correspond to two dates that the ECG was recorded, which are five days apart.

- We will be enumarating STL formulas for both the classes. One of which will be an Abnormal Class and other will be Normal Class.

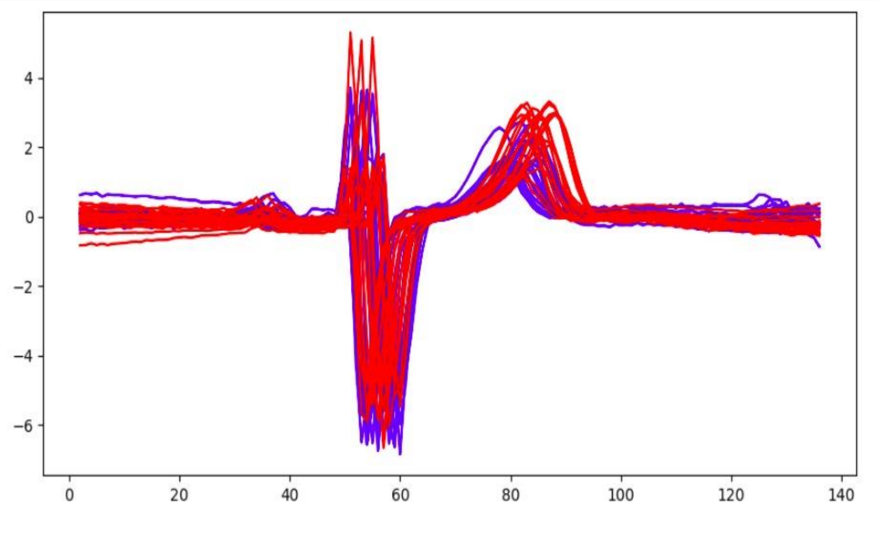
- We used 100 traces from this data set for training and 50 traces for testing purpose.

ECGFive Days Dataset (input to the Algorithm):-

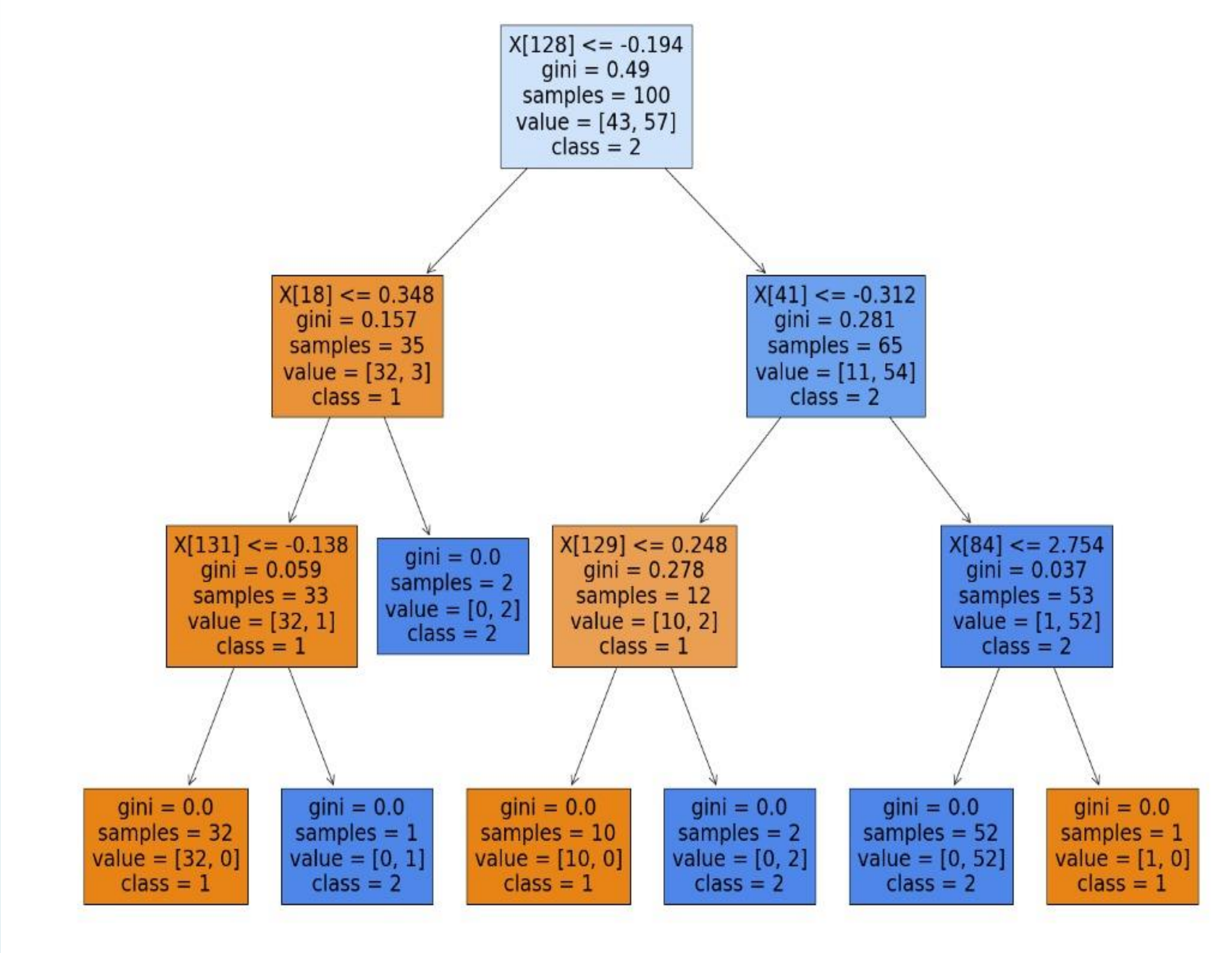
- The Figure shows traces of input data set which is recorded for ECG signals of a person for two different days with 5 days gap.

- It has red traces which represent the first day time-series of ECG signals. --> Class 1

- The blue traces denote the fifth day time-series of ECG signals. --> Class 2



Implementation Result(Decision Tree):-



Implementation Result(Decision Tree) (Contd.):-

- The above decision tree denotes the time-series classification of the dataset into two classes.

- All the leaves of the tree with orange color denotes the class label=1, which is the data from the traces of ECG for the first day.

- All the leaves of the tree with blue color denotes the class label=2, which is the data from the traces of ECG for the fifth day.

- Now , we will be traversing to the leaves having same class label and then find STL formula for the class and similarly for the other class.

Implementation Result(STL Formula):-

- For Class Label 1:-

  $\Phi1 = x[128] <= -0.194 \wedge x[18] <= 0.348 \wedge x[131] <= -0.138$

  $\Phi2 = x[128] > -0.194 \wedge x[41] <= -0.312 \wedge x[129] <= 0.248$

  $\Phi3 = x[128] > -0.194 \wedge x[41] > -0.312 \wedge x[84] > 2.754$

  And Now the STL Formula will be :- **$\Phi(1) = \Phi1 \;\; V \;\; \Phi2 \;\; V \;\; \Phi3$**

- For Class Label 2:-

  $\Phi1 = x[128] <= -0.194 \wedge x[18] <= 0.348 \wedge x[131] > -0.138$

  $\Phi2 = x[128] <= -0.194 \wedge x[18] > 0.348$

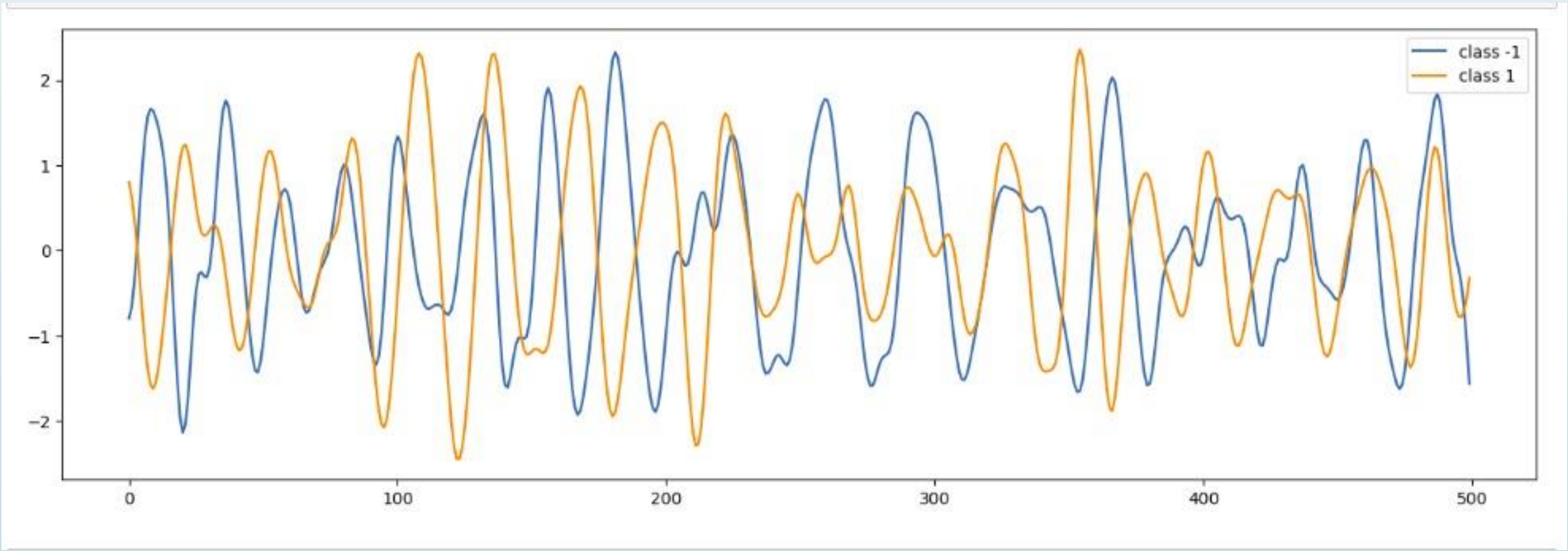
  $\Phi3 = x[128] > -0.194 \wedge x[41] > -0.312 \wedge x[84] <= 2.754$

  $\Phi4 = x[128] > -0.194 \wedge x[41] <= -0.312 \wedge x[129] > 0.248$

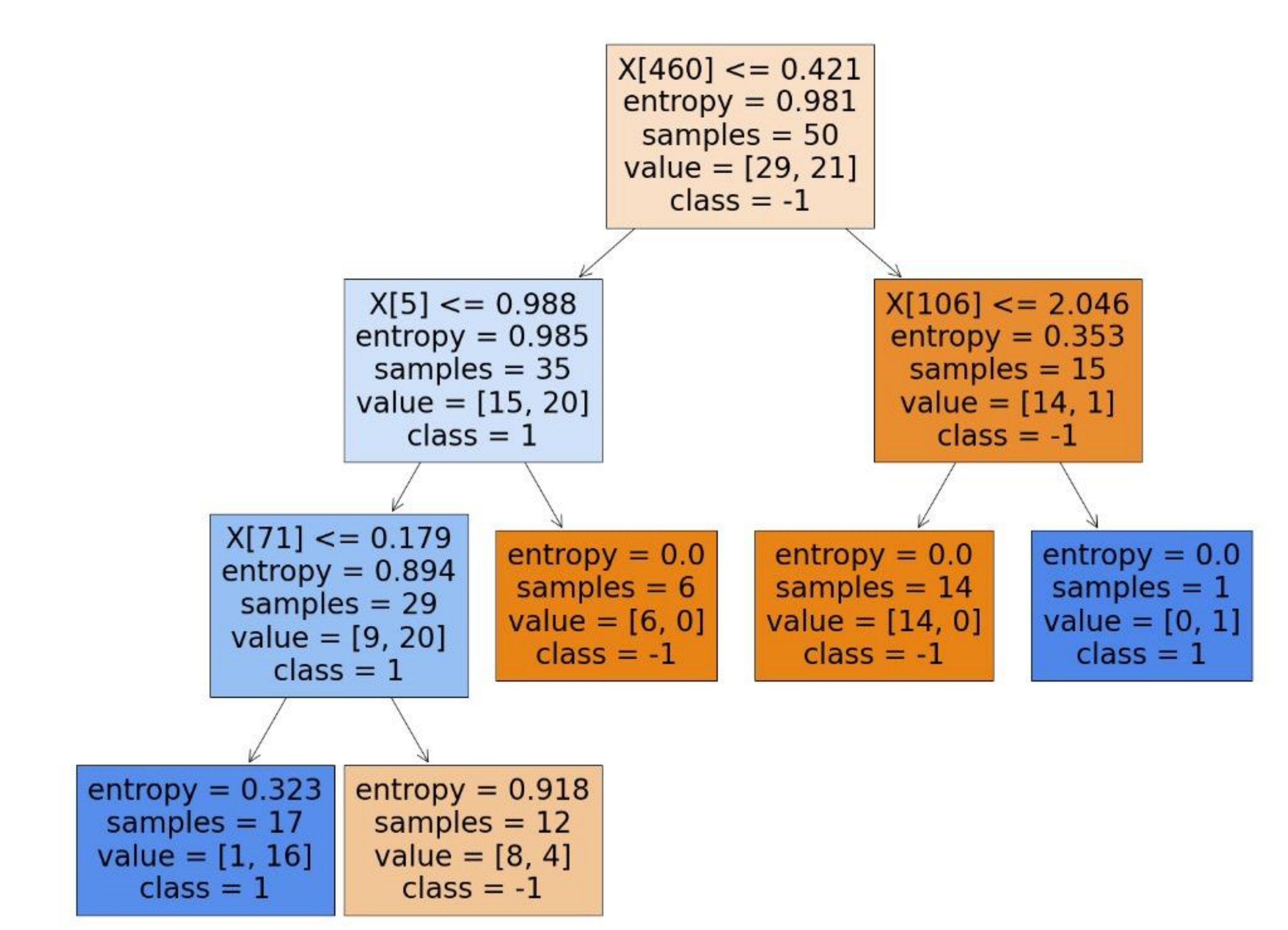
  So, the STL Formula will be :- **$\Phi(2) = \Phi1 \;\; V \;\; \Phi2 \;\; V \;\; \Phi3 \;\; V \;\; \Phi4$**

Other Example(Ford Motor Data):-

- We have taken a time-series data set of Ford A from UCR time-series repository.
- Here ,class +1 denotes some symptom exist in the vehicle system. Whereas Class -1 indicates that symptom not present in the vehicle system.
- The Input trace to the algorithm denoting two classes is as follows.



Decision Tree for above Example:-



STL Formula for Example:-

- STL formula for Class +1 is as follows:-

  $\Phi1 = x[460] <= 0.421 \wedge x[5] <= 0.988 \wedge x[71] <= 0.179$

  $\Phi2 = x[460] > 0.421 \wedge x[106] > 2.046$

  $\boldsymbol{\Phi(+1) = \Phi1 \ \vee \ \Phi2}$

- STL formula for Class -1 is as follows:-

  $\Phi1 = x[460] <= 0.421 \wedge x[5] <= 0.988 \wedge x[71] > 0.179$

  $\Phi2 = x[460] <= 0.421 \wedge x[5] > 0.988$

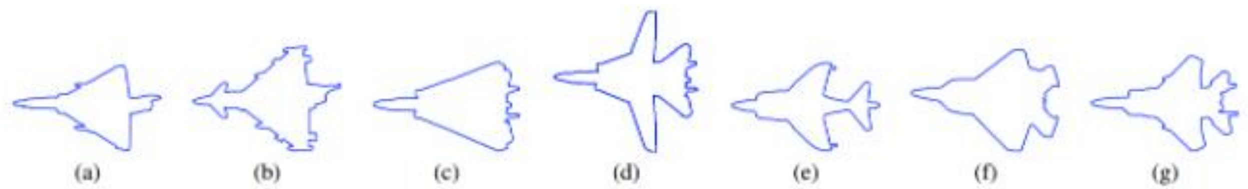
  $\Phi3 = x[460] > 0.421 \wedge x[106] <= 2.046$

  $\boldsymbol{\Phi(-1) = \Phi1 \ \vee \ \Phi2 \ \vee \ \Phi3}$

Post presentation work

Aeroplane outlines classification :

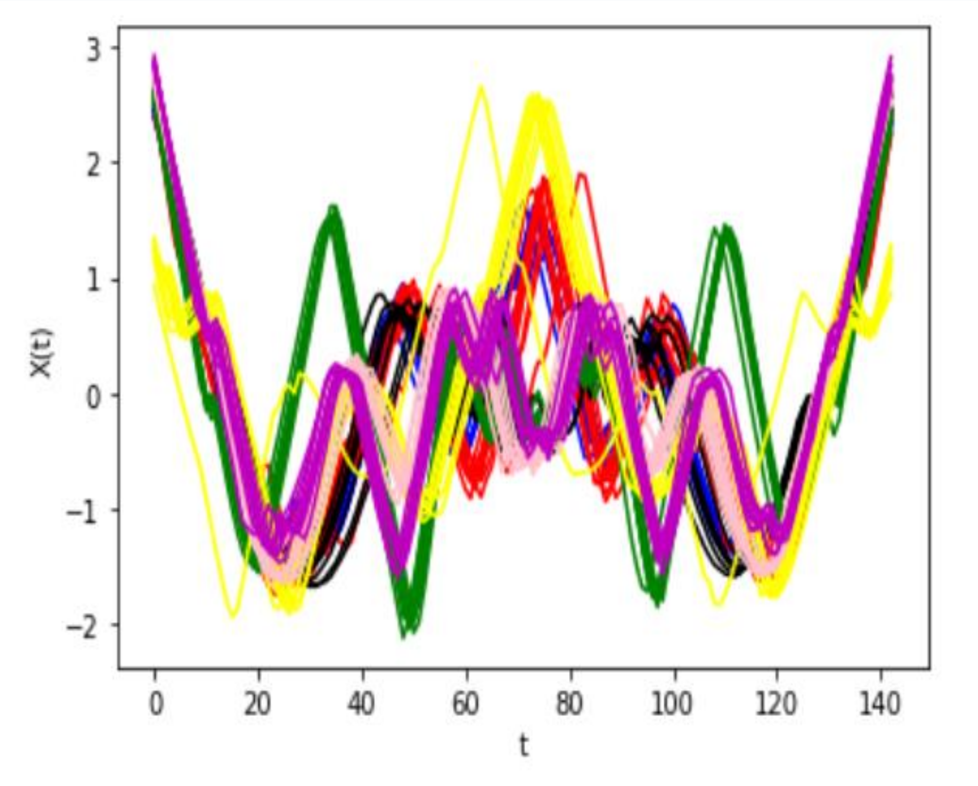
- In this we have taken time series data of plane outlines along with their classes.
- Now , we have trained the same decision tree algorithm according to taken dataset.
- After traversing the decision tree we have enumerated STL formula which indicates the outlines of plane according to their categories.
- It has seven different categories as shown below



Aeroplane shape classes: (a) Mirage, (b) Eurofighter, (c) F-14 wings closed, (d) F-14 wings opened, (e) Harrier, (f) F-22, (g) F-15.

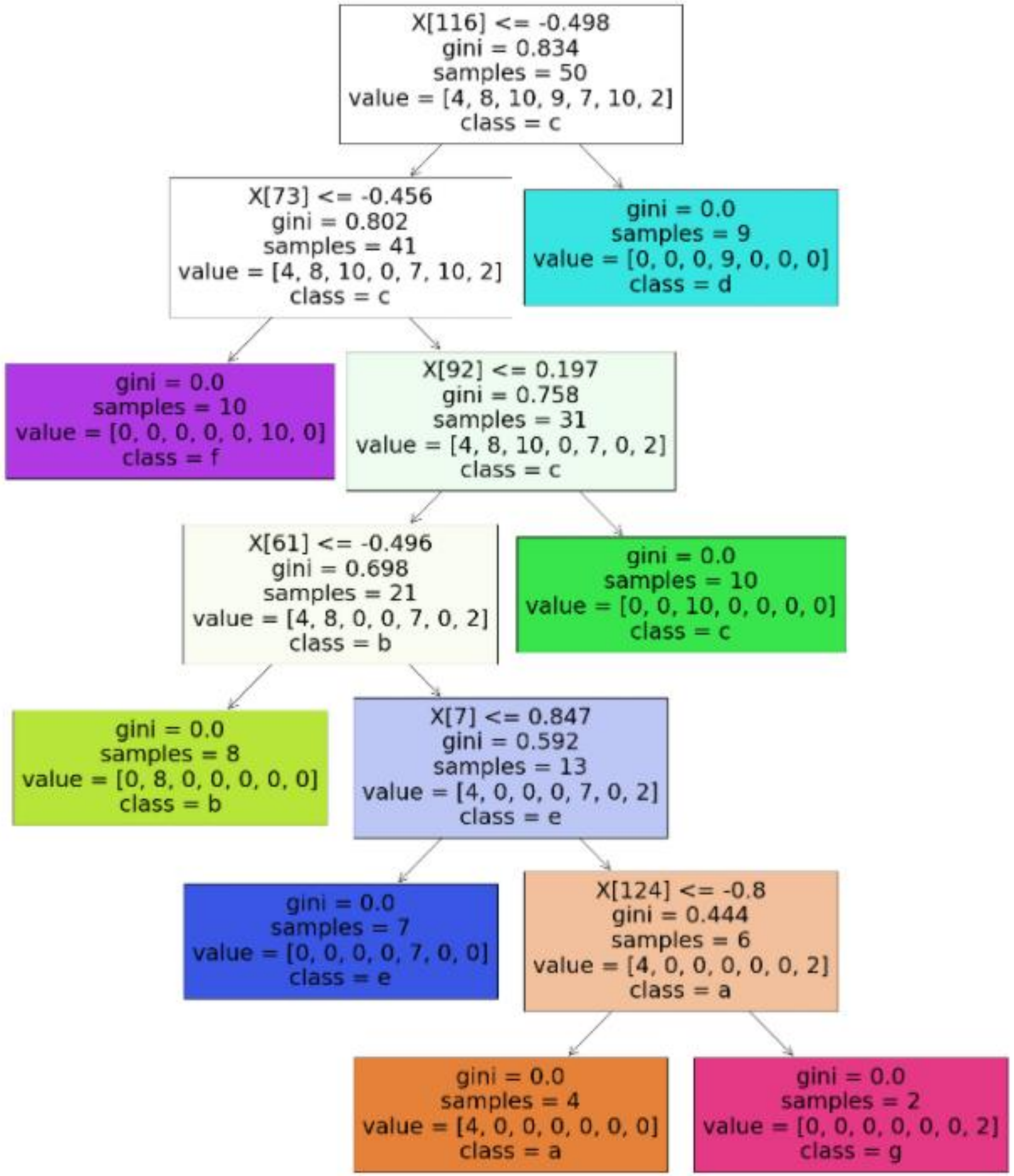
Aeroplane outlines classification(input to the Algorithm):-

- Seven different colored signals in the graph shows the traces of plane outlines for seven different classes mentioned in previous slide.
- These traces will be fed to algorithm for decision tree classification.



Decision tree :

- Every leaf in the tree represents classes a,b,c,d,e,f,g according to the input traces it contains.
- Also we can see that each leaf has gini index=0 which means that this algorithm perfectly and distinguishly classify the input traces of given dataset into these different classes.



STL formula :

- For class ‘f’ : X[116]<= -0.498 ^ X[73]<= -0.456
- For class ‘d’ : X[116]> -0.498
- For class ‘c’ : X[116]<= -0.498 ^ X[73]> -0.456 ^ X[92] > 0.197
- For class ‘b’ : X[116]<= -0.498 ^ X[73]> -0.456 ^ X[92] <= 0.197 ^ X[61] <= -0.496
- For class ‘e’ : X[116]<= -0.498 ^ X[73]> -0.456 ^ X[92] <= 0.197 ^ X[61] > -0.496 ^ X[7]<= 0.847
- For class ‘a’ : X[116]<= -0.498 ^ X[73]> -0.456 ^ X[92] <= 0.197 ^ X[61] > -0.496 ^ X[7] > 0.847 ^ X[124]<= -0.8
- For class ‘g’ : X[116]<= -0.498 ^ X[73]> -0.456 ^ X[92] <= 0.197 ^ X[61] > -0.496 ^ X[7] > 0.847 ^ X[124]> -0.8

Project Timeline :

- 20 sept → project paper selection
- 21 sept – 10 oct → paper reading and understanding
- 11 oct – 5 Nov → Mining dataset and implementation
- 6 Nov – 10 Nov → Preparation for presentation
- 11 Nov → Presentation

Percentage contribution of members:

- Akash Panzade (21111006) : 33%
- Mayuresh Shandilya (21111041) : 34%
- Prajwal Thakare (21111047) : 33%

Thank You!!