DV UML Description

Table of Contents

Example 3
Variable 3
Function 3
Analytics 3
Variables 3
Functions 4
GenerateAnalytics 4
Function 4
AddOldConfusionMatrices 4
Function 4
GetAllDataConfusionMatrix 4
Function 4
GetDataWithoutOverlapConfusionMatrix 5
Function: 5
GetOverlapConfusionMatrix 5
Function 5
GetWorstCaseConfusionMatrix 5
Function 5
GetUserValidationConfusionMatrix 5
Function 5
GetKFoldCrossValidation 5
Function 5
AnalyticsMenu 6
Function 6
AngleSliders 6

Function 6
ColorOptionsMenu 6
Function 6
DataObject 6
Variables 6
Functions 7
DataSetup 7
Variables 7
Functions 7
DataVisualization 9
Variables 9
Functions 9
AddGraph 10
Variables 10
Function 10
DV 10
Variables 10
Functions 13
Main 14
Function 14
RangeSlider 14
Functions 14
RangeSliderUI 15
Variables 15
Functions 15
ChangeHandler 16
Functions 16
RangeTrackListener 17
Functions 17
Resolutions 17
Variables 17

Function 18
ThresholdSliderUI 18
Variable 18
Functions 18
VisualizationOptionsMenu 19
Function 19

Example

Formatting for UML description document.

Variable

Name:	Type:
Description:	

Function

Name:	Params:
Return Type:	
Description:	

Analytics

Generates confusion matrices and k-fold cross validation results.

Variables

Name: percentageOverlapPointsUsed	Type: String
Description: Percentage of overlap points out of all points.	
Name: upper	Type: ArrayList<double[]>
Description: ArrayList holding overlap points on the upper graph.	
Name: lower	Type: ArrayList<double[]>
Description: ArrayList holding overlap points on the lower graph.	
Name: LDAFunction	Type: ArrayList<Double>
Description: ArrayList holding angles and threshold gotten from LDA applied to data without overlapping points.	
Name: curClasses	Type: ArrayList<String>
Description: ArrayList holding all currently visualized classes	
Name: CONFUSION_MATRICES	Type: Map<Integer, JTextArea>
Description: Map holding all created confusion matrices.	

Functions

Name: getCurClasses	Params:
Return Type: void	
Description: Gets current classes being visualized.	
Name: createCSVFileForConfusionMatrix	Params: ArrayList<ArrayList<double[]>> data, String fileName
Return Type: void	
Description: Creates CSV file with data to be used with LDA program.	
Name: LDAForConfusionMatrices	Params: boolean storeFunction, String fileName
Return Type: ArrayList<String>	
Description: Runs Linear Discriminant Analysis program on data in the file filename. Stores LDA function if storeFunction is true.	

GenerateAnalytics

Generates confusion matrices and k-fold cross validation results.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Generates all analytics in separate thread.	

AddOldConfusionMatrices

Gets old confusion matrices in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Adds old confusion matrices in separate thread.	

GetAllDataConfusionMatrix

Generates all data confusion matrix in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates allData confusion matrix in separate thread.	

GetDataWithoutOverlapConfusionMatrix

Generates data without overlap confusion matrix in a separate thread.

Function:

Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates dataWithoutOverlap confusion matrix in separate thread.	

GetOverlapConfusionMatrix

Generates overlap confusion matrix in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates overlap confusion matrix in separate thread.	

GetWorstCaseConfusionMatrix

Generates worst case confusion matrix in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates worst case confusion matrix in separate thread.	

GetUserValidationConfusionMatrix

Generates user validation confusion matrix in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates user validation confusion matrix in separate thread.	

GetKFoldCrossValidation

Generates k-fold cross validation in a separate thread.

Function

Name: doInBackground	Params:
Return Type: Boolean	
Description: Runs k-fold cross validation in separate thread.	

AnalyticsMenu

Menu for toggling on/off or adjusting all analytic options.

Function

Name: AnalyticsMenu	Params: Point mouseLocation
Return Type:	
Description: Constructor for AnalyticsMenu. Creates AnlyticsMenu on mouseLocation.	

AngleSliders

Creates panel with slider for each angle. For each feature/dimension one slider panel will be created.

Function

Name: createSliderPanel	Params: String fieldname, int angle, int index
Return Type: void	
Description: Creates angle slider for given a given feature.	

ColorOptionsMenu

Menu for changing the colors of the graphs.

Function

Name: ColorOptionsMenu	Params: Point mouseLocation
Return Type:	
Description: Constructor for ColorOptionsMenu. Creates ColorOptionsMenu on mouseLocation.	

DataObject

For one class, a DataObject holds the data and GLC-L coordinates for the current angles of the DV program.

Variables

Name: className	Type: String
Description: Class name of data.	
Name: data	Type: double[][]
Description: n-D data	
Name: coordinates	Type: double[][]
Description: X and Y coordinates for each value of each feature of each datapoint.	

Functions

Name: DataObject	Params: String name, double[][] dataValues
Return Type:	
Description: Constructor for DataObject. Instantiates className and data.	
Name: updateCoordinates	Params: double[] angles
Return Type: double	
Description: Updates coordinates of DataObject with given angles. Returns scale of updated coordinates.	
Name: generateCoordinates	Params: double[] datapoint, double[] angles
Return Type: double[][]	
Description: Generates updated coordinates for a single datapoint.	
Name: getXYPoint	Params: double value, double angle
Return Type: double[]	
Description: Generates coordinate for a single value.	

DataSetup

Sets up selected data to be used in the DV program.

Variables

Name: allClasses	Type: ArrayList<String>
Description: Hold classes for recently input data.	
Name: validationClasses	Type: ArrayList<String>
Description: Holds classes for recently input validation data.	

Functions

Name: setupWithData	Params: File dataFile
Return Type: boolean	
Description: Sets up data in dataFile for use in the DV program.	
Name: setupValidationData	Params: File valFile
Return Type: Boolean	
Description: Sets up validation data in valFile for use in the DV program.	
Name: setupImportData	Params: File importFile
Return Type: Boolean	
Description: Sets up data in importFile for use in the DV program.	
Name: setupProjectData	Params: File proectjFile
Return Type:	
Description: Sets up data in projectFile for use in the DV program.	
Name: checkFormat	Params: String[][] stringData
Return Type: Boolean	
Description: Checks if stringData's format is consistent with previously entered data.	

Name: getClassses	Params: String[][] stringData
Return Type: ArrayList<String>	
Description: Gets all classes from last column in stringData.	
Name: checkAllClasses	Params: String[][] stringData
Return Type: Boolean	
Description: Checks if classes in validation data are consistent with previously entered data.	
Name: getStringFromCSV	Params: String[][] stringData
Return Type: String[][]	
Description: Gets String[][] representation of data in csv file.	
Name: purgeID	Params: String[][] stringData
Return Type: String[][]	
Description: Removes ID column from stringData.	
Name: purgeClasses	Params: String[][] stringData
Return Type: String[][]	
Description: Removes class column from stringData.	
Name: getFieldNames	Params: String[][] stringData
Return Type: ArrayList<String>	
Description: Gets field names from header row of stringData.	
Name: stringToNumerical	Params: String[][] stringData
Return Type: double[][]	
Description: Transforms strings to double values.	
Name: normalizeData	Params: double[][] data
Return Type: double[][]	
Description: Uses z-Score Min-Max or Min-Max normalization to normalize data.	
Name: separateByClass	Params: double[][] data, ArrayList<String> classes
Return Type: ArrayList<double[][]>	
Description: Separates each class in data into a separates double[][].	
Name: createDataObjects	Params: Array List<double[][]> data
Return Type: ArrayList<DataObject>	
Description: Creates a DataObject for each double[][] in data.	
Name: addImportedData	Params: ArrayList<double[][]> data, Boolean original
Return Type: ArrayList<DataObject>	
Description: Updates data in DV with new imported data. If original is true, then update original data or else, update normalized data.	
Name: manualMinMaxEntry	Params: String message
Return Type: double[]	
Description: Forum for manual min max entry.	

DataVisualization

Draws and adjusts the graphs for the DV program.

Variables

Name: GRAPHS	Type: Map<Integer, JPanel>
Description: Holds upper and lower graphs.	

Functions

Name: optimizeSetup	Params:
Return Type: void	
Description: Optimizes visualization using LDA and optimizeThreshold().	
Name: optimizeThreshold	Params: double bestAccuracy
Return Type: void	
Description: Finds the best threshold for a visualization.	
Name: optimizeVisualization	Params:
Return Type: void	
Description: Finds the best angles and threshold for a visualization.	
Name: undoOptimization	Params:
Return Type: void	
Description: Reverts to the angle and threshold setup before using optimizeVisualization().	
Name: createCSVFile	Params:
Return Type: void	
Description: Creates csv file with data to be used with LDA program.	
Name: LDA	Params:
Return Type: void	
Description: Runs Linear Discriminant Analysis program on data to get the optimal angles and threshold.	
Name: getAccuracy	Params:
Return Type: void	
Description: gets the accuracy of the current visualization.	
Name: getOverlap	Params:
Return Type: void	
Description: Gets the overlap data of the current visualization.	
Name: drawGraphs	Params: int active
Return Type: void	
Description: Draws graphs.	
Name: getCoordinates	Params: ArrayList<DataObject> dataObjects
Return Type: double	
Description: Updates coordinates for each DataObject in dataObjects. Returns the largest scaling of the dataObjects.	

AddGraph

Draws a single graph in a separate thread.

Variables

Name: DATA_OBJECTS	Type: ArrayList<DataObject>
Description: List of DataObjects to be graphed	
Name: UPPER_OR_LOWER	Type: int
Description: If 0 draw up, else draw down.	
Name: ACTIVE	Type: int
Description: Actively moving part or the graph.	
Name: GRAPH_SCALER	Type: double
Description: Scaler for the graph.	

Function

Name: AddGraph	Params:
Return Type:	
Description: Constructor for AddGraph. Instantiates variables.	
Name: doInBackground	Params:
Return Type: Boolean	
Description: Creates graph in separate thread.	

DV

Main window for the DV program.

Variables

Name: domainSlider	Type: RangeSlider
Description: Slider for the domain.	
Name: overlapSlider	Type: RangeSlider
Description: Slider for the overlap.	
Name: thresholdSlider	Type: JSlider
Description: Slider for the threshold.	
Name: angleSliderPanel	Type: JPanel
Description: Panel that holds angle sliders.	
Name: confusionMatrixPanel	Type: JPanel
Description: Panel that holds confusion matrices.	
Name: crossValidationPanel	Type: JPanel
Description: Panel that holds k-fold cross validation results.	
Name: analyticsPanel	Type: JPanel
Description: Panel that holds confusionMatrixPanel and crossValidationPanel.	
Name: graphPanel	Type: JPanel
Description: Panel that holds graphs.	

Name: sliderPanel	Type: JPanel
Description: Panel that holds sliders.	
Name: graphPane	Type: JScrollPane
Description: Scroll pane for graphs.	
Name: anglesPane	Type: JScrollPane
Description: Scroll pane for angles.	
Name: analyticsPane	Type:
Description: Scroll pane for analytics.	
Name: mainFrame	Type: JFrame
Description: Frame of the DV programs main window.	
Name: domainLines	Type: Color
Description: Color of domain lines.	
Name: overlapLines	Type: Color
Description: Color of Overlap lines.	
Name: thresholdLine	Type: Color
Description: Color of threshold line.	
Name: background	Type: Color
Description: Background color of graphs.	
Name: graphColors	Type: Color[]
Description: Colors of upper and lower graphs.	
Name: showBars	Type: boolean
Description: Whether to show a frequency bar graph or individual marking points for the graphs or not.	
Name: drawOverlap	Type: boolean
Description: Whether to draw all data or just overlap data or not.	
Name: domainActive	Type: boolean
Description: Whether the domain lines are active or not.	
Name: domainArea	Type: double[]
Description: Upper and lower range of the domain.	
Name: overlapArea	Type: double[]
Description: Upper and lower range of the overlap.	
Name: threshold	Type: double
Description: Location of the overlap line.	
Name: prevThreshold	Type: double
Description: Location of the previous threshold before using optimizeVisualization().	
Name: upperClass	Type: int
Description: Index number of the class visualized on the upper graph.	
Name: lowerClasses	Type: ArrayList<Boolean>
Description: ArrayList of Booleans. A Boolean is true if that class is being visualized on the lower graph.	
Name: showPopup	Type: boolean
Description: Whether to show the graph scaling warning popup or not.	

Name: upperIsLower	Type: boolean
Description: Whether the upper class is on the lower or left side of the graph or not.	
Name: accuracy	Type: double
Description: Accuracy of the current visualization.	
Name: allDataCM	Type: String
Description: Confusion matrix for all data of the current visualization.	
Name: prevAllDataCM	Type: ArrayList<String>
Description: ArrayList of each all data confusion matrix before specifying the visualization. Only applies with 3+ class visualizations.	
Name: allDataClassifications	Type: int[]
Description: Correct and incorrect classifications of the current visualization.	
Name: prevAllDataClassifications	Type: ArrayList<int[]>
Description: ArrayList of each all data classification before specifying the visualization. Only applies with 3+ class visualizations.	
Name: prevAllDataChecked	Type: boolean
Description: Whether to display the prevAllData confusion matrix or not.	
Name: allDataChecked	Type: boolean
Description: Whether to display the allData confusion matrix or not.	
Name: withoutOverlapChecked	Type: boolean
Description: Whether to display the withoutOverlap confusion matrix or not.	
Name: overlapChecked	Type: boolean
Description: Whether to display the overlap confusion matrix or not.	
Name: worstCaseChecked	Type: boolean
Description: Whether to display the worst-case confusion matrix or not.	
Name: userValidationChecked	Type: boolean
Description: Whether to display the user validation confusion matrix or not.	
Name: userValidationImported	Type: boolean
Description: Whether the user validation data has been imported or not.	
Name: crossValidationChecked	Type: boolean
Description: Whether to display the k-fold cross validation results or not.	
Name: crossValidationNotGenerated	Type: boolean
Description: Whether the k-fold cross validation results have been generated or not.	
Name: kFolds	Type: int
Description: Number of folds to use in k-fold cross validation.	
Name: hasID	Type: boolean
Description: Whether the data's first column is for ID or not.	
Name: hasClasses	Type: boolean
Description: Whether the data's last column is for classes or not.	
Name: zScoreMinMax	Type: boolean
Description: Whether to use zScoreMinMax normalization or not.	
Name: angles	Type: double[]
Description: Current angles of the visualization.	

Name: prevAngles	Type: double[]
Description: Previous angles of the visualization before using optimzeVisualization().	
Name: data	Type: ArrayList<DataObject>
Description: ArrayList of DataObjects. Each class in the visualization has its own DataObject of normalized data.	
Name: originalData	Type: ArrayList<DataObject>
Description: ArrayList of DataObjects. Each class in the visualization has its own DataObject.	
Name: validationData	Type: ArrayList<DataObject>
Description: ArrayList of DataObjects. Each class in the validation set has its own DataObject of normalized data.	
Name: uniqueClasses	Type: ArrayList<String>
Description: Unique classes in the data.	
Name: classNumber	Type: int
Description: Number of classes in the data.	
Name: fieldNames	Type: ArrayList<String>
Description: Name of each feature/dimension in the data.	
Name: fieldLength	Type: int
Description: Number of dimensions in the data.	
Name: projectSaveName	Type: String
Description: Name of the projects save file.	

Functions

Name: DV	Params:
Return Type:	
Description: Constructor for the DV program. Creates the menu and tool bars.	
Name: createMenuBar	Params:
Return Type: void	
Description: Creates menu bar for the DV program.	
Name: createToolBar	Params:
Return Type: void	
Description: Creates tool bar for the DV program.	
Name: uiPanel	Params:
Return Type: JPanel	
Description: Creates main panel for the DV program.	
Name: blankGraph	Params:
Return Type: ChartPanel	
Description: Creates blank graph.	
Name: createNewProject	Params:
Return Type: void	
Description: Creates new project.	
Name: createUserValidationSet	Params:
Return Type: void	

Description: Creates user validation set.	
Name: importData	Params:
Return Type: void	
Description: Imports new data into project.	
Name: openSavedProject	Params:
Return Type: void	
Description: Opens previously saved project. Projects are saved as csv files.	
Name: saveProject	Params:
Return Type: void	
Description: Saves project with established project save. Projects are saved as csv files.	
Name: saveProjectAs	Params:
Return Type: void	
Description: Saves project with specified filename. Projects are saved as csv file.	
Name: normalizationInfoPopup	Params:
Return Type: void	
Description: Popup giving information on normalization methods.	
Name: resetProgram	Params:
Return Type: void	
Description: Resets program in preparation for a new project.	

Main

Runs DV program.

Function

Name: main	Params: String[] args
Return Type: void	
Description: Runs DV program.	

RangeSlider

Slider for the domain and overlap of the DV program.

Functions

Name: RangeSlider	Params:
Return Type:	
Description: Constructor for RangeSlider. Sets orientation to horizontal.	
Name: getValue	Params:
Return Type: int	
Description: Gets value of lower thumb.	
Name: setValue	Params:
Return Type: void	

Description: Sets value of lower thumb.	
Name: getUpperValue	Params:
Return Type: int	
Description: Gets value of upper thumb.	
Name: setUpperValue	Params:
Return Type: void	
Description: Sets value of upper thumb.	

RangeSliderUI

Look and feel of the RangeSlider.

Variables

Name: TRACK_COLOR	Type: Color
Description: Color of slider track.	
Name: LEFT_THUMB_COLOR	Type: Color
Description: Color of left thumb.	
Name: RIGHT_THUMB_COLOR	Type: Color
Description: Color of right thumb.	
Name: TRACK_SHAPE	Type: RoundRectangel2D.Float
Description: Shape of slider track.	
Name: upperThumbRect	Type: Rectangle
Description: Shape of upper thumb.	
Name: lowerDraggin	Type: boolean
Description: Whether the lower thumb is being dragged or not.	
Name: upperDragging	Type: boolean
Description: Whether the upper thumb is being dragged or not.	
Name: upperThumbSelected	Type: boolean
Description: Whether the upper thumb was the last selected thumb or not.	

Functions

Name: RangeSliderUI	Params: RangeSlider rs, Color track, Color left, Color right
Return Type:	
Description: Constructor for RangeSliderUI. Initializes track and thumb colors.	
Name: installUI	Params: JComponent c
Return Type: void	
Description: Creates upper thumb component.	
Name: createTrackListener	Params: JSlider slider
Return Type: TrackListener	
Description: Creates TrackListener for RangeSlider.	
Name: createChangeListener	Params: JSlider slider

Return Type: ChangeListener	
Description: Crates ChangeListener for RangeSlider.	
Name: calculateTrackRect	Params:
Return Type: void	
Description: Calculates the track rectangle.	
Name: calculateThumbSize	Params:
Return Type: void	
Description: Calculates thumb size.	
Name: calculateThumbLocation	Params:
Return Type: void	
Description: Calculates the location of the lower and upper thumbs.	
Name: getThumbSize	Params:
Return Type: Dimension	
Description: Gets the thumb size.	
Name: setUpperThumbLocation	Params: int x, int y
Return Type: void	
Description: Sets the location of the upper thumb.	
Name: paint	Params: Graphics g, JComponent c
Return Type: void	
Description: Paints slider and thumbs.	
Name: paintTrack	Params: Graphics g
Return Type: void	
Description: Paints slider track.	
Name: paintThumb	Params: Graphics g
Return Type: void	
Description: Overrides paintThumb to do nothing.	
Name: paintLowerThumb	Params: Graphics g
Return Type: void	
Description: Paints lower thumb.	
Name: paintUpperThumb	Params: Graphics g
Return Type: void	
Description: Paints upper thumb.	

ChangeHandler

Handles changes to a RangeSlider that happened without the slider.

Functions

Name: stateChanged	Params: ChangeEvent ag0
Return Type: void	
Description: Updates the RangeSlider if changed without slider.	

RangeTrackListener

Handles change to a RangeSlider that happen with the slider.

Functions

Name: mousePressed	Params: MouseEvent e
Return Type: void	
Description: Gets pressed thumb if mouse is pressed.	
Name: mouseReleased	Params: MouseEvent e
Return Type: void	
Description: Released selected thumb.	
Name: mouseDragged	Params: MouseEvent e
Return Type: void	
Description: Updates thumbs with updated locations.	
Name: moveLowerThumb	Params:
Return Type: void	
Description: Sets lower thumb in new location.	
Name: moveUpperThumb	Params:
Return Type: void	
Description: Sets upper thumb in new location.	

Resolutions

Resolutions for various portions of the DV program for various screen sizes.

Variables

Name: dvWindow	Type: int[]
Description: Resolution for DV program	
Name: angleSliderPanel	Type: int[]
Description: Resolution for angle slider panel.	
Name: chartPanel	Type: int[]
Description: Resolution for chart panel.	
Name: sliderPanel	Type: int[]
Description: Resolution for slider panel.	
Name: anglesPane	Type: int[]
Description: Resolution for angles pane.	
Name: domainSlider	Type: int[]
Description: Resolution for domain slider.	
Name: confusionMatrixPane	Type: int[]
Description: Resolution for confusion matrix pane	
Name: singleChartPanel	Type: int[]
Description: Resolution for single chart panel.	

Function

Name: setResolution	Params: int resolution
Return Type: void	
Description: Sets resolutions for different panes and panels of the DV program.	

ThresholdSliderUI

Look and feel of the Threshold Slider.

Variable

Name: TRACK_SHAPE	Type: RoundRectangle2D.Float
Description: Shape of ThresholdSlider track.	

Functions

Name: ThresholdSliderUI	Params: JSlider b
Return Type:	
Description: Constructor for ThresholdSliderUI	
Name: calculateTrackRect	Params:
Return Type: void	
Description: Calculates the track rectangle.	
Name: calcualteThumbLocation	Params:
Return Type: void	
Description: Calculates the location of the thumb.	
Name: getThumbSize	Params:
Return Type: Dimension	
Description: Gets the thumb size.	
Name: paint	Params: Graphics g, JComponent c
Return Type: void	
Description: Paints the slider.	
Name: paintTrack	Params: Graphics g
Return Type: void	
Description: Paints the track.	
Name: paintThumb	Params: Graphics g
Return Type: void	
Description: Paints the thumb.	

VisualizationOptionsMenu

Menu for various visualization options.

Function

Name: VisualizationOptionsMenu	Params: Point mouseLocation
Return Type:	
Description: Constructor for the VisualizationOptionsMenu. Creates VisualizationOptionsMenu on mouseLocation.	