

Random Jungle

1.0

Generated by Doxygen 1.7.6.1

Fri Feb 10 2012 04:14:05

Contents

1 Namespace Index	1
1.1 Namespace List	1
2 Class Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	11
5.1 rjungle Namespace Reference	11
5.1.1 Function Documentation	11
5.1.1.1 magicAt	11
5.1.2 Variable Documentation	11
5.1.2.1 strLom	11
5.2 rjungle::Generator Namespace Reference	12
5.3 rjungle::Generator::CART Namespace Reference	12
5.3.1 Function Documentation	13
5.3.1.1 assignFunctions	13
5.4 rjungle::Generator::CARTcor Namespace Reference	13
5.4.1 Detailed Description	13
5.4.2 Function Documentation	13
5.4.2.1 assignFunctions	13
5.5 rjungle::Generator::CARTreg Namespace Reference	13

5.5.1	Function Documentation	14
5.5.1.1	assignFunctions	14
5.6	rjungle::Generator::CARTregTwoing Namespace Reference	14
5.6.1	Function Documentation	14
5.6.1.1	assignFunctions	14
5.7	rjungle::Generator::CARTregWithMiss Namespace Reference	14
5.7.1	Detailed Description	15
5.7.2	Function Documentation	15
5.7.2.1	assignFunctions	15
5.8	rjungle::Generator::CARTsrt Namespace Reference	15
5.8.1	Detailed Description	15
5.8.2	Function Documentation	15
5.8.2.1	assignFunctions	15
5.9	rjungle::Generator::CARTsse Namespace Reference	15
5.9.1	Function Documentation	16
5.9.1.1	assignFunctions	16
5.10	rjungle::Generator::CARTtwoing Namespace Reference	16
5.10.1	Function Documentation	16
5.10.1.1	assignFunctions	16
5.11	rjungle::Generator::CARTtwoWayIntAdd Namespace Reference	16
5.11.1	Detailed Description	17
5.11.2	Function Documentation	17
5.11.2.1	assignFunctions	17
5.12	rjungle::Generator::CICase1 Namespace Reference	17
5.12.1	Detailed Description	17
5.12.2	Function Documentation	17
5.12.2.1	assignFunctions	18
5.13	rjungle::Generator::CICase3 Namespace Reference	18
5.13.1	Detailed Description	18
5.13.2	Function Documentation	18
5.13.2.1	assignFunctions	18
5.14	rjungle::Generator::CISNP Namespace Reference	18
5.14.1	Detailed Description	18
5.14.2	Function Documentation	19

5.14.2.1	assignFunctions	19
5.15	rjungle::Generator::imputeTree Namespace Reference	19
5.15.1	Detailed Description	19
5.15.2	Function Documentation	19
5.15.2.1	assignFunctions	19
5.16	rjungle::Generator::LOTUS Namespace Reference	19
5.16.1	Detailed Description	20
5.16.2	Function Documentation	20
5.16.2.1	assignFunctions	20
5.17	rjungle::Generator::SCART Namespace Reference	20
5.17.1	Detailed Description	20
5.17.2	Function Documentation	20
5.17.2.1	assignFunctions	20
5.18	rjungle::Generator::trendCART Namespace Reference	20
5.18.1	Detailed Description	21
5.18.2	Function Documentation	21
5.18.2.1	assignFunctions	21
5.19	rjungle::Generator::UCART Namespace Reference	21
5.19.1	Detailed Description	21
5.19.2	Function Documentation	21
5.19.2.1	assignFunctions	21
5.20	TermResult Namespace Reference	22
5.20.1	Detailed Description	22
5.20.2	Function Documentation	22
5.20.2.1	getMean	22
5.20.2.2	getMostFreq	23
5.20.2.3	getTermLotus	23
5.20.2.4	getTermMean	23
5.20.2.5	getTermMostFreq	24
6	Class Documentation	25
6.1	BitMatrix Class Reference	25
6.1.1	Detailed Description	28
6.1.2	Constructor & Destructor Documentation	28

6.1.2.1	BitMatrix	28
6.1.2.2	~BitMatrix	28
6.1.2.3	BitMatrix	28
6.1.3	Member Function Documentation	28
6.1.3.1	at	28
6.1.3.2	at	29
6.1.3.3	count	29
6.1.3.4	getNcol	29
6.1.3.5	getNrow	29
6.1.3.6	getSizeOfCol	29
6.1.3.7	getSizeOfRow	30
6.1.3.8	init	30
6.1.3.9	operator =	30
6.1.3.10	print	31
6.1.3.11	receiveMpi	31
6.1.3.12	receiveStrMpi	31
6.1.3.13	sendMpi	31
6.1.3.14	setAll	31
6.1.3.15	setNone	32
6.1.4	Member Data Documentation	32
6.1.4.1	dep	32
6.1.4.2	ncol	32
6.1.4.3	nrow	32
6.2	BuildinGenFct< T > Class Template Reference	32
6.2.1	Detailed Description	35
6.2.2	Constructor & Destructor Documentation	35
6.2.2.1	BuildinGenFct	35
6.2.2.2	~BuildinGenFct	35
6.2.3	Member Data Documentation	35
6.2.3.1	addVarProx	35
6.2.3.2	checkUsability	35
6.2.3.3	classifyCmpldTree	35
6.2.3.4	classifyIDcmpldTree	36
6.2.3.5	cmplTree	36

6.2.3.6	getAccuracyOfCmpldTrees	36
6.2.3.7	getBestVariable	36
6.2.3.8	getCenter	36
6.2.3.9	getCutoffs	36
6.2.3.10	getNodePerformance	37
6.2.3.11	getTermResult	37
6.2.3.12	newImportanceObject	37
6.2.3.13	setMtry	37
6.3	ClassAtom< T, C > Class Template Reference	37
6.3.1	Detailed Description	40
6.3.2	Constructor & Destructor Documentation	41
6.3.2.1	ClassAtom	41
6.3.2.2	ClassAtom	41
6.3.2.3	~ClassAtom	41
6.3.3	Member Function Documentation	41
6.3.3.1	classify	41
6.3.3.2	classify	41
6.3.3.3	getErrorMissing	42
6.3.3.4	getMissingCode	42
6.3.3.5	getResult	42
6.3.3.6	getType	43
6.3.3.7	isConsistent	43
6.3.3.8	isThereVarID	43
6.3.3.9	print	43
6.3.3.10	printXml	44
6.3.3.11	resultingNodeSize	44
6.3.3.12	setErrorMissing	44
6.3.3.13	setMissingCode	45
6.3.3.14	size	45
6.3.4	Member Data Documentation	45
6.3.4.1	errorMissing	45
6.3.4.2	missingCode	45
6.4	CmpldTree< T > Class Template Reference	46
6.4.1	Detailed Description	47

6.4.2	Constructor & Destructor Documentation	47
6.4.2.1	CmpldTree	47
6.4.2.2	~CmpldTree	48
6.4.3	Member Function Documentation	48
6.4.3.1	hasVarID	48
6.4.3.2	print	48
6.4.3.3	printXml	48
6.4.3.4	pushBackNode	49
6.4.3.5	pushBackToLastValues	49
6.4.3.6	pushBackToLastValues	49
6.4.3.7	size	49
6.4.3.8	toBranches	50
6.4.3.9	toLastBranches	50
6.4.3.10	toLastValues	50
6.4.3.11	toLastVarID	50
6.4.3.12	toLastVarID	51
6.4.3.13	toValues	51
6.4.3.14	toVarID	51
6.4.3.15	variableToXML	51
6.4.3.16	variableToXMLWithMax	52
6.4.4	Member Data Documentation	52
6.4.4.1	branches	52
6.4.4.2	branchSize	52
6.4.4.3	classes	52
6.4.4.4	extra	52
6.4.4.5	indexes	52
6.4.4.6	termID	53
6.4.4.7	values	53
6.4.4.8	valueSize	53
6.4.4.9	valueWidth	53
6.4.4.10	varID	53
6.4.4.11	varSize	53
6.5	CmplFct Class Reference	53
6.5.1	Detailed Description	55

6.5.2	Constructor & Destructor Documentation	55
6.5.2.1	CmplFct	55
6.5.2.2	~CmplFct	55
6.5.3	Member Function Documentation	56
6.5.3.1	addVarProxT2	56
6.5.3.2	classifyIAM2Way	56
6.5.3.3	classifyIAM2WaynodeID	56
6.5.3.4	classifyS	57
6.5.3.5	classifySnodeID	57
6.5.3.6	classifyT2	57
6.5.3.7	classifyT2nodeID	58
6.5.3.8	getCutoffsT2	58
6.5.3.9	IAM2Way	58
6.5.3.10	Lotus	58
6.5.3.11	makeIAM2Way	59
6.5.3.12	makeLotus	59
6.5.3.13	makeS	59
6.5.3.14	makeT2	60
6.5.3.15	predictLotus	60
6.5.3.16	S	60
6.5.3.17	T2	60
6.6	CondInf< T > Class Template Reference	61
6.6.1	Detailed Description	65
6.6.2	Constructor & Destructor Documentation	65
6.6.2.1	CondInf	65
6.6.2.2	~CondInf	65
6.6.3	Member Function Documentation	66
6.6.3.1	C_kronecker	66
6.6.3.2	C_maxabsConditionalPvalue	66
6.6.3.3	calcCMax	66
6.6.3.4	calcCMaxCase2	66
6.6.3.5	calcdCov_T	66
6.6.3.6	calcdCov_y	66
6.6.3.7	calcdExp_T	66

6.6.3.8	calcdExp_y	66
6.6.3.9	calcPAsym	66
6.6.3.10	calcPAsymCase1	67
6.6.3.11	calcPAsymCase2	67
6.6.3.12	calcSplitCase1	67
6.6.3.13	calcSplitCase2	67
6.6.3.14	calcSplitCase3	67
6.6.3.15	calcSumw	67
6.6.3.16	calcSumwxCT1	67
6.6.3.17	calcT	67
6.6.3.18	checkForCase1	67
6.6.3.19	checkForCase3	68
6.6.3.20	checkForCI	68
6.6.3.21	checkForTwoSampleData	68
6.6.3.22	CICase1	68
6.6.3.23	CICase3	68
6.6.3.24	CIClassification	68
6.6.3.25	CISNP	69
6.6.3.26	getPAsymCase1	69
6.6.3.27	getPAsymCase2	69
6.6.3.28	getPAsymCase3	69
6.6.3.29	prepareXCase1	69
6.6.3.30	prepareXCase2	69
6.6.3.31	prepareXCase3	70
6.6.3.32	prepareYCase1	70
6.6.3.33	prepareYCase2	70
6.6.3.34	prepareYCase3	70
6.6.3.35	setP	70
6.6.3.36	setQ	70
6.6.3.37	trafoRank	70
6.6.4	Member Data Documentation	70
6.6.4.1	abseps	70
6.6.4.2	cMax	71
6.6.4.3	CT1	71

6.6.4.4	dCov_T	71
6.6.4.5	dCov_y	71
6.6.4.6	dExp_T	71
6.6.4.7	dExp_y	71
6.6.4.8	error	71
6.6.4.9	getPAsym	71
6.6.4.10	getSplit	72
6.6.4.11	inform	72
6.6.4.12	maxpts	72
6.6.4.13	p	72
6.6.4.14	pAsym	72
6.6.4.15	prepareX	72
6.6.4.16	prepareY	72
6.6.4.17	q	72
6.6.4.18	redeps	73
6.6.4.19	rng	73
6.6.4.20	split	73
6.6.4.21	sumw	73
6.6.4.22	swx	73
6.6.4.23	t	73
6.6.4.24	tol	73
6.6.4.25	w	73
6.6.4.26	x	74
6.6.4.27	xlsNominal	74
6.6.4.28	y	74
6.6.4.29	yIsNominal	74
6.7	DataFrame< T > Class Template Reference	74
6.7.1	Detailed Description	78
6.7.2	Constructor & Destructor Documentation	78
6.7.2.1	DataFrame	78
6.7.2.2	DataFrame	78
6.7.2.3	DataFrame	78
6.7.2.4	~DataFrame	78
6.7.3	Member Function Documentation	78

6.7.3.1	add	78
6.7.3.2	add	79
6.7.3.3	anyMissingsInCol	79
6.7.3.4	at	79
6.7.3.5	checkVarNames	79
6.7.3.6	createUnsupervisedData	79
6.7.3.7	div	79
6.7.3.8	div	79
6.7.3.9	divDiag	79
6.7.3.10	divNoDiag	79
6.7.3.11	find	80
6.7.3.12	getArray	80
6.7.3.13	getCol	80
6.7.3.14	getCol	80
6.7.3.15	getColMaskedRow	80
6.7.3.16	getColMiss	80
6.7.3.17	getColNoMissings	80
6.7.3.18	getColNoMissingsExt	80
6.7.3.19	getColOrig	80
6.7.3.20	getDataFromCSV	81
6.7.3.21	getDepVar	81
6.7.3.22	getIndexOfVarName	81
6.7.3.23	getIndexOfVarNames	81
6.7.3.24	getMissingCode	81
6.7.3.25	getMissings	81
6.7.3.26	getncol	81
6.7.3.27	getnrow	81
6.7.3.28	getRow	81
6.7.3.29	getRow	82
6.7.3.30	getRowMaskedClassesInCol	82
6.7.3.31	getRowMaskedCol	82
6.7.3.32	getRowMaskedColNoMissing	82
6.7.3.33	getRowMaskedColNoMissing	82
6.7.3.34	getRowMaskedColNoMissing2	82

6.7.3.35	getRowMaskedColNoMissing2AndPair	82
6.7.3.36	getRowMaskedColNoMissing2AndPair	83
6.7.3.37	getRowMaskedColNoMissing3	83
6.7.3.38	getRowPieceWise	83
6.7.3.39	getVarNames	83
6.7.3.40	initMatrix	83
6.7.3.41	makeDepVecs	83
6.7.3.42	minus	83
6.7.3.43	minusXPow2	83
6.7.3.44	missingsNotAllowed	84
6.7.3.45	mult	84
6.7.3.46	nanToMinusFive	84
6.7.3.47	operator[]	84
6.7.3.48	pow2	84
6.7.3.49	print	84
6.7.3.50	printCSV	84
6.7.3.51	printSummary	84
6.7.3.52	pushBackLom	84
6.7.3.53	rm	85
6.7.3.54	set	85
6.7.3.55	setAll	85
6.7.3.56	setArray	85
6.7.3.57	setCol	85
6.7.3.58	setColMaskedRow	85
6.7.3.59	setColSynth	85
6.7.3.60	setDepVar	85
6.7.3.61	setDepVarName	85
6.7.3.62	setDim	86
6.7.3.63	setMissingCode	86
6.7.3.64	setVarNames	86
6.7.3.65	sqrareroot	86
6.7.3.66	storeCategories	86
6.7.3.67	storeHalfCategories	86
6.7.4	Member Data Documentation	86

6.7.4.1	catLimit	86
6.7.4.2	depIdxVec	86
6.7.4.3	depValVec	86
6.7.4.4	isUnsupervised	87
6.7.4.5	lom	87
6.7.4.6	m_data	87
6.7.4.7	missings	87
6.7.4.8	par	87
6.7.4.9	ped_data	87
6.7.4.10	varCategories	87
6.7.4.11	varClassIndexer	87
6.7.4.12	varHalfCategories	87
6.7.4.13	varNames	88
6.8	DataTreeSet Class Reference	88
6.8.1	Detailed Description	91
6.8.2	Constructor & Destructor Documentation	91
6.8.2.1	DataTreeSet	91
6.8.2.2	DataTreeSet	91
6.8.2.3	DataTreeSet	91
6.8.3	Member Function Documentation	91
6.8.3.1	combineMpi	91
6.8.3.2	getNsmpl	91
6.8.3.3	getNtree	91
6.8.3.4	getSmplSize	92
6.9	Exception Class Reference	92
6.9.1	Detailed Description	92
6.9.2	Constructor & Destructor Documentation	92
6.9.2.1	Exception	92
6.9.2.2	Exception	92
6.9.2.3	~Exception	92
6.9.3	Member Function Documentation	93
6.9.3.1	what	93
6.9.4	Member Data Documentation	93
6.9.4.1	status	93

6.9.4.2	str	93
6.10	FittingFct Class Reference	93
6.10.1	Detailed Description	94
6.10.2	Constructor & Destructor Documentation	94
6.10.2.1	FittingFct	94
6.10.2.2	~FittingFct	94
6.10.3	Member Function Documentation	94
6.10.3.1	CARTgini	95
6.10.3.2	CARTginiCor	95
6.10.3.3	CARTginiNoDenominator	95
6.10.3.4	CARTginiSrt	95
6.10.3.5	CARTginiSrtOld	95
6.10.3.6	CARTginiSrtWithMissing	95
6.10.3.7	CARTginisse	95
6.10.3.8	CARTreg	96
6.10.3.9	CARTregTwoing	96
6.10.3.10	CARTtwoing	96
6.10.3.11	CARTtwoWayIntAdd	96
6.10.3.12	checkUsabilityLOTUS	96
6.10.3.13	checkUsabilitySse	96
6.10.3.14	imputeTree	96
6.10.3.15	LOTUS	96
6.10.3.16	SCARTgini	97
6.10.3.17	trendCARTgini	97
6.10.3.18	unbiasedCARTgini	97
6.10.3.19	unbiasedCARTgini_	97
6.11	FittingFctPar< T > Class Template Reference	97
6.11.1	Detailed Description	99
6.11.2	Constructor & Destructor Documentation	99
6.11.2.1	FittingFctPar	99
6.11.2.2	FittingFctPar	100
6.11.2.3	~FittingFctPar	100
6.11.3	Member Data Documentation	100
6.11.3.1	bestVar	100

6.11.3.2	colMaskVec	100
6.11.3.3	data	100
6.11.3.4	depth	100
6.11.3.5	depVar	100
6.11.3.6	importance	100
6.11.3.7	io	101
6.11.3.8	iteration	101
6.11.3.9	nodePerformance	101
6.11.3.10	parent	101
6.11.3.11	rng	101
6.11.3.12	rowMaskVec	101
6.11.3.13	rowWeight	101
6.11.3.14	stopGrowing	101
6.11.3.15	treeImprovement	101
6.11.3.16	twoWayInteraction	101
6.12	Generator< T > Class Template Reference	102
6.12.1	Detailed Description	102
6.12.2	Constructor & Destructor Documentation	102
6.12.2.1	Generator	102
6.12.2.2	~Generator	102
6.12.3	Member Function Documentation	102
6.12.3.1	CLASSIFYCMPLDTREEFCT	102
6.12.3.2	CMPLTREEFCT	103
6.12.3.3	GETBESTVARFCT	103
6.12.3.4	GETNODEPERFFCT	103
6.12.3.5	GETTERMRESFCT	103
6.13	Helper Class Reference	103
6.13.1	Detailed Description	108
6.13.2	Constructor & Destructor Documentation	108
6.13.2.1	Helper	108
6.13.2.2	~Helper	108
6.13.3	Member Function Documentation	108
6.13.3.1	add	108
6.13.3.2	addMasked	109

6.13.3.3 addToCIGrid	109
6.13.3.4 clearPtrVec	109
6.13.3.5 convertOOBtoMask	109
6.13.3.6 copy	109
6.13.3.7 getAbs	109
6.13.3.8 getClasses	110
6.13.3.9 getClassesAndRate	110
6.13.3.10 getClassifiedVectors	110
6.13.3.11 getColSelection	110
6.13.3.12 getCountOfInterval	111
6.13.3.13 getCSVdim	111
6.13.3.14 getCSVdim	111
6.13.3.15 getDiff	111
6.13.3.16 getIndexVector	111
6.13.3.17 getIntervalGroup	111
6.13.3.18 getMad	112
6.13.3.19 getMask	112
6.13.3.20 getMaskedVec	112
6.13.3.21 getMax	112
6.13.3.22 getMean	112
6.13.3.23 getMean2	112
6.13.3.24 getMean3	113
6.13.3.25 getMean4	113
6.13.3.26 getMeanProx	113
6.13.3.27 getMeanProxX	113
6.13.3.28 getMedian	113
6.13.3.29 getMedianWithSort	113
6.13.3.30 getMedianX	113
6.13.3.31 getMedianX2	113
6.13.3.32 getMin	114
6.13.3.33 getMode	114
6.13.3.34 getMostFreq	114
6.13.3.35 getMostFreqIndex	114
6.13.3.36 getMostFreqLD	114

6.13.3.37 getMostFreqProp	114
6.13.3.38 getMostFreqProp	114
6.13.3.39 getMostFreqProx	114
6.13.3.40 getMostFreqProx2	115
6.13.3.41 getMostFreqX	115
6.13.3.42 getOne	115
6.13.3.43 getPearsonCor	115
6.13.3.44 getPowerSet	115
6.13.3.45 getPredScore	115
6.13.3.46 getProp	116
6.13.3.47 getQuantiles	116
6.13.3.48 getQuantilesLotusSplit	116
6.13.3.49 getRanks	116
6.13.3.50 getRanks	116
6.13.3.51 getRanks	116
6.13.3.52 getSize	116
6.13.3.53 getSlicedVec	117
6.13.3.54 getSNPsInHighLD	117
6.13.3.55 getSNPsInLD	117
6.13.3.56 getSum	117
6.13.3.57 getTreeSizeProb	117
6.13.3.58 getTreeSizeX	117
6.13.3.59 inc	117
6.13.3.60 inverseMap	118
6.13.3.61 isClassTree	118
6.13.3.62 isElement	118
6.13.3.63 isEven	118
6.13.3.64 isGuessingLomNumeric	118
6.13.3.65 isIn	118
6.13.3.66 isOdd	118
6.13.3.67 makeMap	118
6.13.3.68 makeMap2SortedInvPair	118
6.13.3.69 makePairVec	119
6.13.3.70 makeToFreq	119

6.13.3.71	one	119
6.13.3.72	performLrAndGetDeviance	119
6.13.3.73	permuteInGrid	119
6.13.3.74	printTime	119
6.13.3.75	printTreeMap	120
6.13.3.76	printTreePair	120
6.13.3.77	printTreePair2Way	120
6.13.3.78	printVec	120
6.13.3.79	printVec	120
6.13.3.80	printVec	120
6.13.3.81	purityGini	120
6.13.3.82	purityLotus	121
6.13.3.83	puritySos	121
6.13.3.84	purityTwoing	122
6.13.3.85	putPairFirstToVec	122
6.13.3.86	putPairSecondToVec	122
6.13.3.87	seq	122
6.13.3.88	strSplit	122
6.13.3.89	strSplit	123
6.13.3.90	vecToPairs	123
6.13.3.91	vectorToXML	123
6.14	IAM2WayImportance< T > Class Template Reference	123
6.14.1	Detailed Description	126
6.14.2	Constructor & Destructor Documentation	126
6.14.2.1	IAM2WayImportance	126
6.14.2.2	~IAM2WayImportance	126
6.14.3	Member Function Documentation	127
6.14.3.1	add	127
6.14.3.2	add	127
6.14.3.3	newImportanceObject	127
6.14.3.4	print	127
6.14.3.5	reset	127
6.14.4	Member Data Documentation	127
6.14.4.1	amounts	127

6.14.4.2	freqPairs	127
6.14.4.3	idx	128
6.14.4.4	idxlt	128
6.14.4.5	indexer	128
6.14.4.6	vecSizeLimit	128
6.15	IAMClassAtom< T > Class Template Reference	128
6.15.1	Detailed Description	131
6.15.2	Constructor & Destructor Documentation	131
6.15.2.1	IAMClassAtom	131
6.15.2.2	~IAMClassAtom	132
6.15.3	Member Function Documentation	132
6.15.3.1	classify	132
6.15.3.2	classify	132
6.15.3.3	getClassSet	132
6.15.3.4	getType	132
6.15.3.5	getVarID	133
6.15.3.6	isConsistent	133
6.15.3.7	isThereVarID	133
6.15.3.8	operator==	133
6.15.3.9	print	133
6.15.3.10	printXml	134
6.15.3.11	resultingNodeSize	134
6.15.3.12	setClassSet	134
6.15.3.13	setVarID	134
6.15.4	Member Data Documentation	135
6.15.4.1	classSet	135
6.15.4.2	varID	135
6.16	Importance< T > Class Template Reference	135
6.16.1	Detailed Description	138
6.16.2	Constructor & Destructor Documentation	138
6.16.2.1	Importance	138
6.16.2.2	~Importance	138
6.16.3	Member Function Documentation	139
6.16.3.1	add	139

6.16.3.2	combineMpi	139
6.16.3.3	getBestVariables	139
6.16.3.4	getIteration	139
6.16.3.5	init	139
6.16.3.6	print	139
6.16.3.7	printHeader	139
6.16.3.8	reset	140
6.16.3.9	save	140
6.16.3.10	setBaseData	140
6.16.3.11	setBaseDataFrom	140
6.16.3.12	setIteration	140
6.16.4	Member Data Documentation	140
6.16.4.1	data	140
6.16.4.2	io	140
6.16.4.3	iteration	140
6.16.4.4	par	141
6.17	INode< T > Class Template Reference	141
6.17.1	Detailed Description	144
6.17.2	Constructor & Destructor Documentation	144
6.17.2.1	INode	144
6.17.2.2	INode	144
6.17.2.3	INode	144
6.17.2.4	~INode	145
6.17.3	Member Function Documentation	145
6.17.3.1	classify	145
6.17.3.2	getClassifier	145
6.17.3.3	getLeafSum	145
6.17.3.4	getMaxDepth	145
6.17.3.5	getNumOfLeafs	145
6.17.3.6	getNumOfNodes	145
6.17.3.7	isThereVarID	146
6.17.3.8	print	146
6.17.3.9	printXml	146
6.17.3.10	setClassifier	146

6.17.4 Member Data Documentation	146
6.17.4.1 classifier	146
6.17.4.2 outcomeEmergency	146
6.18 JTreeCtrl< T > Class Template Reference	146
6.18.1 Detailed Description	149
6.18.2 Constructor & Destructor Documentation	149
6.18.2.1 JTreeCtrl	149
6.18.2.2 JTreeCtrl	149
6.18.2.3 ~JTreeCtrl	149
6.18.3 Member Function Documentation	149
6.18.3.1 addProximity2order	149
6.18.3.2 getColBuffer	149
6.18.3.3 getColChooseBuffer	150
6.18.3.4 getData	150
6.18.3.5 getFreqFormNode	150
6.18.3.6 getFreqImportance	150
6.18.3.7 getFreqMap	150
6.18.3.8 getRowBuffer	150
6.18.3.9 getRowChooseBuffer	150
6.18.3.10 getVarImp	150
6.18.3.11 makeBuffer	151
6.18.3.12 makeChoose	151
6.18.3.13 makeNode	151
6.18.3.14 makeTree	151
6.18.3.15 setData	151
6.18.3.16 setFreqMap	151
6.18.4 Member Data Documentation	151
6.18.4.1 colBuffer	151
6.18.4.2 colBufferSize	151
6.18.4.3 colChoose	152
6.18.4.4 data	152
6.18.4.5 fitFctPar	152
6.18.4.6 freqMap	152
6.18.4.7 gen	152

6.18.4.8 importance	152
6.18.4.9 io	152
6.18.4.10 par	152
6.18.4.11 rowBuffer	152
6.18.4.12 rowChoose	152
6.18.4.13 sizeCutoff	153
6.18.4.14 trainSetSize	153
6.18.4.15 trainSetWeight	153
6.18.4.16 twoWayInteraction	153
6.18.4.17 varImp	153
6.19 JTreeTweaks Struct Reference	153
6.19.1 Detailed Description	153
6.19.2 Member Data Documentation	154
6.19.2.1 maxTreeDepth	154
6.19.2.2 targetPartitionSize	154
6.20 Logistic Class Reference	154
6.20.1 Detailed Description	154
6.20.2 Member Function Documentation	154
6.20.2.1 comp_w_z	155
6.20.2.2 gsl_vector_sum	155
6.20.2.3 left_side_matrix	155
6.20.2.4 logistic	155
6.20.2.5 logistic_reg	155
6.20.2.6 logistic_reg_iter	155
6.20.2.7 logisticreg_iter_2	155
6.20.2.8 right_side_vector	155
6.20.2.9 solve_sym	155
6.21 LotusTermClassAtom< T, C > Class Template Reference	156
6.21.1 Detailed Description	159
6.21.2 Constructor & Destructor Documentation	159
6.21.2.1 LotusTermClassAtom	159
6.21.2.2 LotusTermClassAtom	159
6.21.2.3 LotusTermClassAtom	159
6.21.2.4 ~LotusTermClassAtom	159

6.21.3 Member Data Documentation	160
6.21.3.1 betas	160
6.21.3.2 val	160
6.21.3.3 varID	160
6.22 Node< T, C > Class Template Reference	160
6.22.1 Detailed Description	161
6.22.2 Constructor & Destructor Documentation	161
6.22.2.1 Node	161
6.22.2.2 ~Node	161
6.22.2.3 ~Node	161
6.22.3 Member Function Documentation	161
6.22.3.1 at	161
6.22.3.2 at	162
6.22.3.3 classify	162
6.22.3.4 getLeafSum	162
6.22.3.5 getMaxDepth	162
6.22.3.6 getNumOfLeafs	162
6.22.3.7 getNumOfNodes	162
6.22.3.8 getParent	162
6.22.3.9 getPath	163
6.22.3.10 isThereVarID	163
6.22.3.11 operator[]	163
6.22.3.12 operator[]	163
6.22.3.13 print	163
6.22.3.14 printXml	163
6.22.3.15 push_back	163
6.22.3.16 resize	163
6.22.3.17 setParent	164
6.22.3.18 size	164
6.22.4 Member Data Documentation	164
6.22.4.1 kids	164
6.22.4.2 parent	164
6.23 PermlImportance< T > Class Template Reference	164
6.23.1 Detailed Description	167

6.23.2	Constructor & Destructor Documentation	168
6.23.2.1	PermlImportance	168
6.23.2.2	~PermlImportance	168
6.23.3	Member Function Documentation	168
6.23.3.1	add	168
6.23.3.2	combineMpi	168
6.23.3.3	finalize	168
6.23.3.4	getBestVariables	168
6.23.3.5	getScaling	168
6.23.3.6	init	168
6.23.3.7	load	169
6.23.3.8	newImportanceObject	169
6.23.3.9	reset	169
6.23.3.10	save	169
6.23.3.11	setBaseData	169
6.23.4	Member Data Documentation	169
6.23.4.1	amounts_breiman	169
6.23.4.2	amounts_liaw	169
6.23.4.3	amounts_meng	170
6.23.4.4	amounts_raw	170
6.23.4.5	expX2_breiman	170
6.23.4.6	expX2_liaw	170
6.23.4.7	expX_values	170
6.23.4.8	gen	170
6.23.4.9	ntrees	170
6.23.4.10	sd_breiman	170
6.23.4.11	sd_liaw	170
6.24	Prediction< T > Class Template Reference	171
6.24.1	Detailed Description	173
6.24.2	Constructor & Destructor Documentation	173
6.24.2.1	Prediction	173
6.24.2.2	~Prediction	173
6.24.3	Member Function Documentation	174
6.24.3.1	add	174

6.24.3.2	addOne	174
6.24.3.3	at	174
6.24.3.4	combineMpi	174
6.24.3.5	init	174
6.24.3.6	initMatrix	174
6.24.3.7	initOne	174
6.24.3.8	removePred	174
6.24.3.9	setBaseData	175
6.24.4	Member Data Documentation	175
6.24.4.1	acc	175
6.24.4.2	data	175
6.24.4.3	gen	175
6.24.4.4	io	175
6.24.4.5	ncol	175
6.24.4.6	nrow	175
6.24.4.7	par	175
6.24.4.8	pred	175
6.25	Profiler Class Reference	176
6.25.1	Detailed Description	176
6.25.2	Constructor & Destructor Documentation	176
6.25.2.1	Profiler	176
6.25.2.2	~Profiler	176
6.25.3	Member Function Documentation	177
6.25.3.1	start	177
6.25.3.2	stop	177
6.25.4	Member Data Documentation	177
6.25.4.1	isTicking	177
6.25.4.2	timeStamp	177
6.25.4.3	totalTime	177
6.26	ProtCount Class Reference	177
6.26.1	Detailed Description	178
6.26.2	Constructor & Destructor Documentation	178
6.26.2.1	ProtCount	178
6.26.2.2	ProtCount	178

6.26.3 Member Function Documentation	178
6.26.3.1 loadUp	178
6.26.4 Member Data Documentation	178
6.26.4.1 numOfSamples	178
6.26.4.2 row	178
6.27 Proximities< T > Class Template Reference	179
6.27.1 Detailed Description	179
6.27.2 Constructor & Destructor Documentation	180
6.27.2.1 Proximities	180
6.27.2.2 Proximities	180
6.27.2.3 Proximities	180
6.27.2.4 Proximities	180
6.27.2.5 ~Proximities	180
6.27.3 Member Function Documentation	180
6.27.3.1 add	180
6.27.3.2 add	180
6.27.3.3 addVecs	180
6.27.3.4 at	181
6.27.3.5 combineMpi	181
6.27.3.6 div	181
6.27.3.7 div	181
6.27.3.8 getCol	181
6.27.3.9 getncol	181
6.27.3.10 getnrow	181
6.27.3.11 getRow	181
6.27.3.12 initMatrix	181
6.27.3.13 initWithData	182
6.27.3.14 print	182
6.27.3.15 printCSV	182
6.27.3.16 reset	182
6.27.3.17 set	182
6.27.3.18 setDiag	182
6.27.3.19 setDim	182
6.27.4 Member Data Documentation	182

6.27.4.1	data	182
6.27.4.2	ncol	182
6.27.4.3	nrow	183
6.27.4.4	proxMatrix	183
6.28	RJungleAcc< T > Class Template Reference	183
6.28.1	Detailed Description	183
6.28.2	Constructor & Destructor Documentation	183
6.28.2.1	RJungleAcc	183
6.28.2.2	~RJungleAcc	183
6.28.3	Member Function Documentation	184
6.28.3.1	getAccuracyCmpld	184
6.28.3.2	getAccuracyCmpldSos	184
6.28.3.3	getDevianceCmpld	184
6.29	RJungleBinder< T > Class Template Reference	184
6.29.1	Detailed Description	186
6.29.2	Constructor & Destructor Documentation	186
6.29.2.1	RJungleBinder	186
6.29.2.2	~RJungleBinder	186
6.29.3	Member Data Documentation	186
6.29.3.1	cmlpdTrees	186
6.29.3.2	freqPairs	187
6.29.3.3	gen	187
6.29.3.4	intrinsicImportance	187
6.29.3.5	io	187
6.29.3.6	oobSet	187
6.29.3.7	par	187
6.29.3.8	permImportance	187
6.29.3.9	pred	187
6.29.3.10	proxi	187
6.29.3.11	trees	187
6.29.3.12	twoWayInteraction	188
6.29.3.13	varProxi	188
6.29.3.14	yaimp	188
6.30	RJungleCompiler< T > Class Template Reference	188

6.30.1	Detailed Description	188
6.30.2	Constructor & Destructor Documentation	188
6.30.2.1	RJungleCompiler	188
6.30.2.2	~RJungleCompiler	189
6.30.3	Member Function Documentation	189
6.30.3.1	compileTrees	189
6.31	RJungleConfusion< T > Class Template Reference	189
6.31.1	Detailed Description	189
6.31.2	Constructor & Destructor Documentation	189
6.31.2.1	RJungleConfusion	189
6.31.2.2	~RJungleConfusion	189
6.31.3	Member Function Documentation	190
6.31.3.1	printConfMatNew	190
6.32	RJungleCtrl< T > Class Template Reference	190
6.32.1	Detailed Description	190
6.32.2	Constructor & Destructor Documentation	191
6.32.2.1	RJungleCtrl	191
6.32.2.2	~RJungleCtrl	191
6.32.3	Member Function Documentation	191
6.32.3.1	autoBuild	191
6.32.3.2	autoBuildInternal	191
6.32.3.3	imputeSNPs	191
6.32.3.4	setParAndIO	191
6.32.3.5	tuneMtry	191
6.32.4	Member Data Documentation	191
6.32.4.1	binder	192
6.33	RJungleFromXML< T > Class Template Reference	192
6.33.1	Detailed Description	194
6.33.2	Constructor & Destructor Documentation	195
6.33.2.1	RJungleFromXML	195
6.33.2.2	~RJungleFromXML	195
6.33.2.3	getNextTree	195
6.33.3	Member Function Documentation	195
6.33.3.1	getNextTree	195

6.33.3.2	getPar	195
6.33.3.3	getParFromXmlFile	195
6.33.3.4	processNode	196
6.33.3.5	startXmlReader	196
6.33.3.6	stopXmlReader	196
6.33.4	Member Data Documentation	196
6.33.4.1	branches	196
6.33.4.2	branchSize	196
6.33.4.3	classes	196
6.33.4.4	cmlldTree	197
6.33.4.5	curOptionName	197
6.33.4.6	curVarName	197
6.33.4.7	data	197
6.33.4.8	extra	197
6.33.4.9	indexes	197
6.33.4.10	par	197
6.33.4.11	reader	197
6.33.4.12	ret	197
6.33.4.13	stopParsing	197
6.33.4.14	termID	198
6.33.4.15	values	198
6.33.4.16	valueSize	198
6.33.4.17	valueWidth	198
6.33.4.18	varID	198
6.33.4.19	varSize	198
6.34	RJungleGen< T > Class Template Reference	198
6.34.1	Detailed Description	199
6.34.2	Constructor & Destructor Documentation	199
6.34.2.1	RJungleGen	199
6.34.2.2	~RJungleGen	199
6.34.3	Member Function Documentation	199
6.34.3.1	init	199
6.34.4	Member Data Documentation	199
6.34.4.1	fct	199

6.34.4.2	isPlugin	199
6.34.4.3	sdl_library	199
6.34.4.4	wasInitialized	200
6.35	RJungleGrow< T > Class Template Reference	200
6.35.1	Detailed Description	200
6.35.2	Constructor & Destructor Documentation	200
6.35.2.1	RJungleGrow	200
6.35.2.2	~RJungleGrow	201
6.35.3	Member Function Documentation	201
6.35.3.1	doFetchJungleFromFile	201
6.35.3.2	doPermlmp	201
6.35.3.3	doSampleProxi	201
6.35.3.4	doSaveJungle	201
6.35.3.5	doVarProxi	201
6.35.3.6	growLocal	201
6.35.3.7	twoWayInteractionCmp	202
6.36	RJungleHelper< T > Class Template Reference	202
6.36.1	Detailed Description	202
6.36.2	Constructor & Destructor Documentation	203
6.36.2.1	RJungleHelper	203
6.36.2.2	~RJungleHelper	203
6.36.3	Member Function Documentation	203
6.36.3.1	printFooter	203
6.36.3.2	printHeader	203
6.36.3.3	printRJunglePar	203
6.36.3.4	printXml	203
6.36.3.5	printXmlFooter	203
6.36.3.6	printXmlHeader	203
6.36.3.7	printXmlRaw	203
6.36.3.8	setMtryClassification	204
6.36.3.9	setMtryRegression	204
6.36.3.10	summary	204
6.36.3.11	tuneMtry	204
6.36.3.12	tuneNtree	204

6.37	RJungleImportance< T > Class Template Reference	204
6.37.1	Detailed Description	205
6.37.2	Constructor & Destructor Documentation	205
6.37.2.1	RJungleImportance	205
6.37.2.2	~RJungleImportance	205
6.37.3	Member Function Documentation	205
6.37.3.1	makePermVarImp2	205
6.37.3.2	makePermVarImp3	205
6.37.3.3	printVarImp	205
6.37.3.4	printPermVarImp	206
6.38	RJungleImpute< T > Class Template Reference	206
6.38.1	Detailed Description	206
6.38.2	Constructor & Destructor Documentation	206
6.38.2.1	RJungleImpute	206
6.38.2.2	~RJungleImpute	206
6.38.3	Member Function Documentation	206
6.38.3.1	imputeData	207
6.38.3.2	imputeSNPs	207
6.38.3.3	imputeTrivially	207
6.38.3.4	imputeTriviallyGWA	207
6.39	RJungleIO Class Reference	207
6.39.1	Detailed Description	208
6.39.2	Constructor & Destructor Documentation	208
6.39.2.1	RJungleIO	208
6.39.2.2	RJungleIO	208
6.39.2.3	RJungleIO	208
6.39.2.4	~RJungleIO	208
6.39.3	Member Function Documentation	209
6.39.3.1	close	209
6.39.3.2	isClassTree	209
6.39.3.3	open	209
6.39.4	Member Data Documentation	209
6.39.4.1	fileVerbose	209
6.39.4.2	inXMLjungle	209

6.39.4.3	outConfusion	209
6.39.4.4	outConfusion2	209
6.39.4.5	outExtractData	209
6.39.4.6	outImportance	209
6.39.4.7	outImportance2	209
6.39.4.8	outImputedData	210
6.39.4.9	outLog	210
6.39.4.10	outOOB	210
6.39.4.11	outOutlier	210
6.39.4.12	outPrediction	210
6.39.4.13	outPrototypes	210
6.39.4.14	outSamProximity	210
6.39.4.15	outSummary	210
6.39.4.16	outTuneMtry	210
6.39.4.17	outVarProximity	210
6.39.4.18	outVerbose	211
6.39.4.19	outVotes	211
6.39.4.20	outXmlJungle	211
6.39.4.21	outYaimp	211
6.40	RJungleOutlier Class Reference	211
6.40.1	Detailed Description	211
6.40.2	Constructor & Destructor Documentation	212
6.40.2.1	RJungleOutlier	212
6.40.2.2	~RJungleOutlier	212
6.40.3	Member Function Documentation	212
6.40.3.1	getOutlier	212
6.41	RJunglePar Struct Reference	212
6.41.1	Detailed Description	213
6.41.2	Member Data Documentation	214
6.41.2.1	allcont_flag	214
6.41.2.2	backSel	214
6.41.2.3	classweights	214
6.41.2.4	colSelection	214
6.41.2.5	condimp	214

6.41.2.6	cutoffHighLD	214
6.41.2.7	delimiter	214
6.41.2.8	delimScale	214
6.41.2.9	depVar	214
6.41.2.10	depVarCol	214
6.41.2.11	depVarName	215
6.41.2.12	downsampling_flag	215
6.41.2.13	extractdata_flag	215
6.41.2.14	filename	215
6.41.2.15	gwa_flag	215
6.41.2.16	impMeasure	215
6.41.2.17	imputelt	215
6.41.2.18	maxTreeDepth	215
6.41.2.19	memMode	215
6.41.2.20	missingcode	215
6.41.2.21	mpi	216
6.41.2.22	mpild	216
6.41.2.23	mpiSize	216
6.41.2.24	mtry	216
6.41.2.25	ncol	216
6.41.2.26	nrow	216
6.41.2.27	nthreads	216
6.41.2.28	ntree	216
6.41.2.29	ntreeMpi	216
6.41.2.30	numOfImpVar	216
6.41.2.31	oob_flag	217
6.41.2.32	outlier	217
6.41.2.33	outprefix	217
6.41.2.34	pedfile_flag	217
6.41.2.35	permresponse_flag	217
6.41.2.36	plugin	217
6.41.2.37	pluginPar	217
6.41.2.38	predict	217
6.41.2.39	prototypes	217

6.41.2.40	rng	217
6.41.2.41	sampleproximities_flag	218
6.41.2.42	saveJungleType	218
6.41.2.43	seed	218
6.41.2.44	skipCol	218
6.41.2.45	skipRow	218
6.41.2.46	summary_flag	218
6.41.2.47	targetPartitionSize	218
6.41.2.48	testlib_flag	218
6.41.2.49	transpose_flag	218
6.41.2.50	treeType	218
6.41.2.51	tunemtry	219
6.41.2.52	varNamesRow	219
6.41.2.53	varproximities	219
6.41.2.54	verbose_flag	219
6.41.2.55	version	219
6.41.2.56	votes_flag	219
6.41.2.57	weightsim_flag	219
6.41.2.58	yaimp_flag	219
6.42	RJunglePrediction< T > Class Template Reference	219
6.42.1	Detailed Description	220
6.42.2	Constructor & Destructor Documentation	220
6.42.2.1	RJunglePrediction	220
6.42.2.2	~RJunglePrediction	220
6.42.3	Member Function Documentation	220
6.42.3.1	showPredictionCmpld	220
6.43	RJungleProto< T > Class Template Reference	220
6.43.1	Detailed Description	221
6.43.2	Constructor & Destructor Documentation	221
6.43.2.1	RJungleProto	221
6.43.2.2	~RJungleProto	221
6.43.3	Member Function Documentation	221
6.43.3.1	countNumOfSamples	221
6.43.3.2	getPrototypes	221

6.44	RJungleProxi< T > Class Template Reference	221
6.44.1	Detailed Description	222
6.44.2	Constructor & Destructor Documentation	222
6.44.2.1	RJungleProxi	222
6.44.2.2	~RJungleProxi	222
6.44.3	Member Function Documentation	222
6.44.3.1	finalize	222
6.44.3.2	finalizeVarProx	223
6.44.3.3	initVarProx	223
6.44.3.4	normalizeProximitiesCmpld	223
6.44.3.5	updateProximitiesCmpld	223
6.44.3.6	updateVarProximitiesCmpld	223
6.44.3.7	updateVarProximitiesCmpld1	223
6.44.3.8	updateVarProximitiesCmpld2	223
6.45	RJungleTuneMtry< T > Class Template Reference	224
6.45.1	Detailed Description	224
6.45.2	Constructor & Destructor Documentation	224
6.45.2.1	RJungleTuneMtry	224
6.45.2.2	~RJungleTuneMtry	224
6.45.3	Member Function Documentation	224
6.45.3.1	tuneMtry	224
6.46	SaveCollector Class Reference	225
6.46.1	Detailed Description	227
6.46.2	Constructor & Destructor Documentation	227
6.46.2.1	SaveCollector	227
6.46.2.2	~SaveCollector	228
6.46.3	Member Function Documentation	228
6.46.3.1	clear	228
6.46.3.2	getIdxMax	228
6.46.3.3	print	228
6.46.3.4	printElement	228
6.46.3.5	push_back	228
6.46.3.6	push_back	228
6.46.3.7	push_back	228

6.46.3.8	push_back	228
6.46.3.9	push_back	229
6.46.4	Member Data Documentation	229
6.46.4.1	colNames	229
6.46.4.2	doubleVec	229
6.46.4.3	intVec	229
6.46.4.4	isAvailable	229
6.46.4.5	orderCol	229
6.46.4.6	orderRow	229
6.46.4.7	par	229
6.46.4.8	repeatLast	229
6.46.4.9	showDepVar	229
6.46.4.10	showHeader	230
6.46.4.11	size_tVec	230
6.46.4.12	stringVec	230
6.46.4.13	uli_tVec	230
6.47	SClassAtom< T > Class Template Reference	230
6.47.1	Detailed Description	233
6.47.2	Constructor & Destructor Documentation	233
6.47.2.1	SClassAtom	233
6.47.2.2	SClassAtom	234
6.47.2.3	~SClassAtom	234
6.47.3	Member Function Documentation	234
6.47.3.1	classify	234
6.47.3.2	classify	234
6.47.3.3	getClassSet	234
6.47.3.4	getType	234
6.47.3.5	getVarID	235
6.47.3.6	isConsistent	235
6.47.3.7	isThereVarID	235
6.47.3.8	operator==	235
6.47.3.9	print	236
6.47.3.10	printXml	236
6.47.3.11	resultingNodeSize	236

6.47.3.12	setClassSet	236
6.47.3.13	setVarID	237
6.47.4	Member Data Documentation	237
6.47.4.1	classSet	237
6.47.4.2	varID	237
6.48	StringIterator Class Reference	237
6.48.1	Detailed Description	238
6.48.2	Constructor & Destructor Documentation	238
6.48.2.1	StringIterator	238
6.48.2.2	StringIterator	238
6.48.2.3	~StringIterator	238
6.48.3	Member Function Documentation	238
6.48.3.1	empty	238
6.48.3.2	init	239
6.48.3.3	init	239
6.48.3.4	mystrtok	239
6.48.3.5	next	239
6.48.3.6	next	239
6.48.3.7	setDelimiters	239
6.48.4	Member Data Documentation	240
6.48.4.1	delimiters	240
6.48.4.2	lastDelim	240
6.48.4.3	myLastToken	240
6.48.4.4	strTmp	240
6.48.4.5	token	240
6.49	T2ClassAtom< T > Class Template Reference	240
6.49.1	Detailed Description	243
6.49.2	Constructor & Destructor Documentation	243
6.49.2.1	T2ClassAtom	243
6.49.2.2	T2ClassAtom	244
6.49.2.3	T2ClassAtom	244
6.49.2.4	~T2ClassAtom	244
6.49.3	Member Function Documentation	244
6.49.3.1	classify	244

6.49.3.2	classify	244
6.49.3.3	getThreshold	244
6.49.3.4	getType	245
6.49.3.5	getVarID	245
6.49.3.6	isConsistent	245
6.49.3.7	isThereVarID	245
6.49.3.8	operator==	246
6.49.3.9	print	246
6.49.3.10	printXml	246
6.49.3.11	resultingNodeSize	246
6.49.3.12	setThreshold	247
6.49.3.13	setVarID	247
6.49.4	Member Data Documentation	247
6.49.4.1	threshold	247
6.49.4.2	varID	247
6.50	TermClassAtom< T, C > Class Template Reference	247
6.50.1	Detailed Description	250
6.50.2	Constructor & Destructor Documentation	250
6.50.2.1	TermClassAtom	250
6.50.2.2	TermClassAtom	250
6.50.2.3	TermClassAtom	250
6.50.2.4	~TermClassAtom	251
6.50.3	Member Function Documentation	251
6.50.3.1	getResult	251
6.50.3.2	getType	251
6.50.3.3	isConsistent	251
6.50.3.4	print	251
6.50.3.5	printXml	252
6.50.4	Member Data Documentation	252
6.50.4.1	val	252
6.51	TestClass Class Reference	252
6.51.1	Detailed Description	253
6.51.2	Constructor & Destructor Documentation	253
6.51.2.1	TestClass	253

6.51.2.2	~TestClass	253
6.51.3	Member Function Documentation	253
6.51.3.1	helpers	253
6.51.3.2	lr	254
6.51.3.3	mk_train_lr_predict	254
6.51.3.4	testC_maxabsConditionalPvalue	254
6.51.3.5	testCI	254
6.51.3.6	testDataFrame	254
6.51.3.7	testDataFrameAndNode	254
6.51.3.8	testFind	254
6.51.3.9	testGini1	254
6.51.3.10	testGini2	254
6.51.3.11	testJTreeCtrl4	254
6.51.3.12	testJTreeCtrl5	254
6.51.3.13	testMvt	254
6.51.3.14	testNode	254
6.51.3.15	testPurity	254
6.51.3.16	testRandomJungleCtrl1	254
6.51.3.17	testRandomJungleCtrl2	254
6.51.3.18	testSClassAtom	255
6.51.3.19	testT2ClassAtom	255
6.51.3.20	testTMClassAtom	255
6.51.3.21	testTree	255
6.51.3.22	train_save	255
6.52	TimeProf Class Reference	255
6.52.1	Detailed Description	256
6.52.2	Constructor & Destructor Documentation	256
6.52.2.1	TimeProf	256
6.52.2.2	~TimeProf	256
6.52.3	Member Function Documentation	256
6.52.3.1	start	256
6.52.3.2	stop	256
6.52.3.3	writeToFile	256
6.52.4	Member Data Documentation	256

6.52.4.1 profilers	256
6.53 TImportance< T > Class Template Reference	257
6.53.1 Detailed Description	260
6.53.2 Constructor & Destructor Documentation	260
6.53.2.1 TImportance	260
6.53.2.2 ~TImportance	260
6.53.3 Member Function Documentation	261
6.53.3.1 add	261
6.53.3.2 add	261
6.53.3.3 disableHeader	261
6.53.3.4 enableHeader	261
6.53.3.5 getBestVariables	261
6.53.3.6 init	261
6.53.3.7 newImportanceObject	261
6.53.3.8 reset	262
6.53.3.9 save	262
6.53.3.10 setIteration	262
6.53.4 Member Data Documentation	262
6.53.4.1 amounts	262
6.53.4.2 freqPairs	262
6.53.4.3 ids	262
6.53.4.4 iterationVec	262
6.53.4.5 saveCol	262
6.54 TMClassAtom< T > Class Template Reference	263
6.54.1 Detailed Description	266
6.54.2 Constructor & Destructor Documentation	267
6.54.2.1 TMClassAtom	267
6.54.2.2 ~TMClassAtom	267
6.54.2.3 ~TMClassAtom	267
6.54.3 Member Function Documentation	267
6.54.3.1 classify	267
6.54.3.2 classify	267
6.54.3.3 getThresholds	267
6.54.3.4 getType	268

6.54.3.5	getVarID	268
6.54.3.6	isConsistent	268
6.54.3.7	isThereVarID	268
6.54.3.8	operator==	269
6.54.3.9	print	269
6.54.3.10	printXml	269
6.54.3.11	resultingNodeSize	269
6.54.3.12	setThresholds	270
6.54.3.13	setThresholdsNoSort	270
6.54.3.14	setVarID	270
6.54.3.15	size	270
6.54.4	Member Data Documentation	270
6.54.4.1	thresholds	270
6.54.4.2	varID	270
6.55	Tree< T, C > Class Template Reference	270
6.55.1	Detailed Description	271
6.55.2	Constructor & Destructor Documentation	271
6.55.2.1	Tree	271
6.55.2.2	Tree	271
6.55.2.3	~Tree	271
6.55.3	Member Function Documentation	271
6.55.3.1	classify	271
6.55.3.2	getRoot	272
6.55.3.3	isThereVarID	272
6.55.3.4	print	272
6.55.3.5	printXml	272
6.55.3.6	setRoot	272
6.55.3.7	summary	272
6.55.4	Member Data Documentation	272
6.55.4.1	root	272
7	File Documentation	273
7.1	src/library/BitMatrix.cpp File Reference	273
7.2	src/library/BitMatrix.h File Reference	273

7.2.1	Define Documentation	274
7.2.1.1	NULL	274
7.2.2	Function Documentation	274
7.2.2.1	operator<<	274
7.3	src/library/BuildinGenerators.cpp File Reference	275
7.4	src/library/BuildinGenerators.h File Reference	275
7.5	src/library/ClassAtom.cpp File Reference	277
7.6	src/library/ClassAtom.h File Reference	278
7.6.1	Function Documentation	278
7.6.1.1	operator<<	279
7.7	src/library/CmpldTree.cpp File Reference	279
7.8	src/library/CmpldTree.h File Reference	279
7.8.1	Function Documentation	280
7.8.1.1	operator<<	280
7.9	src/library/CmplFct.cpp File Reference	280
7.10	src/library/CmplFct.h File Reference	280
7.11	src/library/CondInf.cpp File Reference	281
7.12	src/library/CondInf.h File Reference	281
7.13	src/library/DataFrame.cpp File Reference	282
7.14	src/library/DataFrame.h File Reference	282
7.14.1	Function Documentation	283
7.14.1.1	operator<<	283
7.15	src/library/DataTreeSet.cpp File Reference	283
7.16	src/library/DataTreeSet.h File Reference	283
7.16.1	Function Documentation	284
7.16.1.1	operator<<	284
7.17	src/library/ErrorCodes.h File Reference	285
7.17.1	Define Documentation	287
7.17.1.1	ERRORCODE_1	287
7.17.1.2	ERRORCODE_10	287
7.17.1.3	ERRORCODE_11	287
7.17.1.4	ERRORCODE_12	287
7.17.1.5	ERRORCODE_13	287
7.17.1.6	ERRORCODE_14	287

7.17.1.7 ERRORCODE_15	287
7.17.1.8 ERRORCODE_16	288
7.17.1.9 ERRORCODE_17	288
7.17.1.10 ERRORCODE_18	288
7.17.1.11 ERRORCODE_19	288
7.17.1.12 ERRORCODE_2	288
7.17.1.13 ERRORCODE_20	288
7.17.1.14 ERRORCODE_21	288
7.17.1.15 ERRORCODE_22	288
7.17.1.16 ERRORCODE_23	288
7.17.1.17 ERRORCODE_24	288
7.17.1.18 ERRORCODE_25	289
7.17.1.19 ERRORCODE_26	289
7.17.1.20 ERRORCODE_27	289
7.17.1.21 ERRORCODE_28	289
7.17.1.22 ERRORCODE_29	289
7.17.1.23 ERRORCODE_3	289
7.17.1.24 ERRORCODE_30	289
7.17.1.25 ERRORCODE_31	289
7.17.1.26 ERRORCODE_32	289
7.17.1.27 ERRORCODE_33	289
7.17.1.28 ERRORCODE_34	290
7.17.1.29 ERRORCODE_35	290
7.17.1.30 ERRORCODE_36	290
7.17.1.31 ERRORCODE_37	290
7.17.1.32 ERRORCODE_38	290
7.17.1.33 ERRORCODE_39	290
7.17.1.34 ERRORCODE_4	290
7.17.1.35 ERRORCODE_40	290
7.17.1.36 ERRORCODE_41	290
7.17.1.37 ERRORCODE_42	290
7.17.1.38 ERRORCODE_43	291
7.17.1.39 ERRORCODE_44	291
7.17.1.40 ERRORCODE_45	291

7.17.1.41 ERRORCODE_46	291
7.17.1.42 ERRORCODE_47	291
7.17.1.43 ERRORCODE_48	291
7.17.1.44 ERRORCODE_49	291
7.17.1.45 ERRORCODE_5	291
7.17.1.46 ERRORCODE_50	291
7.17.1.47 ERRORCODE_51	292
7.17.1.48 ERRORCODE_52	292
7.17.1.49 ERRORCODE_53	292
7.17.1.50 ERRORCODE_54	292
7.17.1.51 ERRORCODE_55	292
7.17.1.52 ERRORCODE_56	292
7.17.1.53 ERRORCODE_57	292
7.17.1.54 ERRORCODE_58	292
7.17.1.55 ERRORCODE_59	292
7.17.1.56 ERRORCODE_6	293
7.17.1.57 ERRORCODE_60	293
7.17.1.58 ERRORCODE_61	293
7.17.1.59 ERRORCODE_62	293
7.17.1.60 ERRORCODE_63	293
7.17.1.61 ERRORCODE_64	293
7.17.1.62 ERRORCODE_7	293
7.17.1.63 ERRORCODE_8	293
7.17.1.64 ERRORCODE_9	293
7.18 src/library/Exception.cpp File Reference	294
7.19 src/library/Exception.h File Reference	294
7.20 src/library/FittingFct.cpp File Reference	295
7.21 src/library/FittingFct.h File Reference	295
7.22 src/library/Generator.cpp File Reference	296
7.23 src/library/Generator.h File Reference	296
7.24 src/library/gzstream.cpp File Reference	297
7.25 src/library/gzstream.h File Reference	297
7.25.1 Define Documentation	298
7.25.1.1 igzstream	298

7.25.1.2	ogzstream	298
7.26	src/library/Helper.cpp File Reference	298
7.27	src/library/Helper.h File Reference	298
7.28	src/library/IAM2WayImportance.cpp File Reference	299
7.29	src/library/IAM2WayImportance.h File Reference	299
7.30	src/library/IAMClassAtom.cpp File Reference	300
7.31	src/library/IAMClassAtom.h File Reference	300
7.31.1	Define Documentation	301
7.31.1.1	CCvector	301
7.31.1.2	Tvector	301
7.31.2	Function Documentation	301
7.31.2.1	operator<<	301
7.32	src/library/Importance.cpp File Reference	302
7.33	src/library/Importance.h File Reference	302
7.34	src/library/init.h File Reference	302
7.34.1	Function Documentation	304
7.34.1.1	operator<<	304
7.34.1.2	operator<<	304
7.34.1.3	operator>>	304
7.34.1.4	operator>>	304
7.35	src/library/INode.cpp File Reference	304
7.36	src/library/INode.h File Reference	304
7.37	src/library/ioLines.cpp File Reference	305
7.38	src/library/ioLines.h File Reference	305
7.39	src/library/JTreeCtrl.h File Reference	306
7.40	src/library/JTreeTweaks.cpp File Reference	307
7.41	src/library/JTreeTweaks.h File Reference	308
7.41.1	Typedef Documentation	308
7.41.1.1	JTreeTweaks	308
7.41.1.2	uli_t	308
7.42	src/library/librjungle.cpp File Reference	308
7.42.1	Function Documentation	309
7.42.1.1	initRJunglePar	309
7.42.1.2	randomJungle	309

7.42.1.3	testLibrandomjungle	309
7.43	src/library/librjungle.h File Reference	309
7.43.1	Function Documentation	310
7.43.1.1	initRJunglePar	310
7.43.1.2	randomJungle	310
7.43.1.3	testLibrandomjungle	310
7.44	src/library/Logistic.cpp File Reference	310
7.44.1	Define Documentation	311
7.44.1.1	MPFR_MYPREC	311
7.45	src/library/Logistic.h File Reference	311
7.46	src/library/LotusTermClassAtom.cpp File Reference	312
7.47	src/library/LotusTermClassAtom.h File Reference	312
7.48	src/library/mvt.cpp File Reference	314
7.48.1	Function Documentation	314
7.48.1.1	gsl_rng_uniform_fortran_	314
7.48.1.2	gsl_rng_uniform_fortran_	314
7.49	src/library/mvt.h File Reference	315
7.49.1	Function Documentation	315
7.49.1.1	gsl_rng_uniform_fortran_	316
7.49.1.2	gsl_rng_uniform_fortran_	316
7.49.1.3	mvtdst_	316
7.50	src/library/mvtfortran.f File Reference	316
7.50.1	Function Documentation	317
7.50.1.1	MVBVN	317
7.50.1.2	MVBVT	317
7.50.1.3	MVBVTC	317
7.50.1.4	mvbvtl	317
7.50.1.5	MVBVU	317
7.50.1.6	MVCHNC	317
7.50.1.7	MVCHNV	317
7.50.1.8	MVKBRV	318
7.50.1.9	MVKRSV	318
7.50.1.10	MVLIMS	318
7.50.1.11	MVPHI	318

7.50.1.12 MVPHNV	318
7.50.1.13 MVSORT	318
7.50.1.14 MVSPCL	318
7.50.1.15 MVSSWP	319
7.50.1.16 MVSTDT	319
7.50.1.17 MVSUBR	319
7.50.1.18 MVSWAP	319
7.50.1.19 MVTDNS	319
7.50.1.20 MVTDST	319
7.50.1.21 MVUNI	319
7.50.1.22 MVVLSB	319
7.51 src/library/Node.cpp File Reference	320
7.52 src/library/Node.h File Reference	320
7.52.1 Function Documentation	321
7.52.1.1 operator<<	321
7.53 src/library/PermlImportance.cpp File Reference	321
7.54 src/library/PermlImportance.h File Reference	321
7.55 src/library/Prediction.cpp File Reference	322
7.56 src/library/Prediction.h File Reference	322
7.57 src/library/Profiler.cpp File Reference	323
7.58 src/library/Profiler.h File Reference	324
7.59 src/library/Proximities.cpp File Reference	324
7.60 src/library/Proximities.h File Reference	324
7.60.1 Function Documentation	325
7.60.1.1 operator<<	325
7.61 src/library/RJungleAcc.cpp File Reference	325
7.62 src/library/RJungleAcc.h File Reference	325
7.63 src/library/RJungleBinder.cpp File Reference	326
7.64 src/library/RJungleBinder.h File Reference	326
7.65 src/library/RJungleCompiler.cpp File Reference	327
7.66 src/library/RJungleCompiler.h File Reference	327
7.67 src/library/RJungleConfusion.cpp File Reference	329
7.68 src/library/RJungleConfusion.h File Reference	329
7.69 src/library/RJungleCtrl.cpp File Reference	330

7.70	src/library/RJungleCtrl.h File Reference	330
7.71	src/library/RJungleFromXML.cpp File Reference	331
7.72	src/library/RJungleFromXML.h File Reference	331
7.73	src/library/RJungleGen.cpp File Reference	333
7.74	src/library/RJungleGen.h File Reference	333
7.75	src/library/RJungleGrow.cpp File Reference	333
7.76	src/library/RJungleGrow.h File Reference	333
7.77	src/library/RJungleHelper.cpp File Reference	334
7.78	src/library/RJungleHelper.h File Reference	334
7.79	src/library/RJungleImportance.cpp File Reference	335
7.80	src/library/RJungleImportance.h File Reference	335
7.81	src/library/RJungleImpute.cpp File Reference	336
7.82	src/library/RJungleImpute.h File Reference	336
7.83	src/library/RJungleIO.cpp File Reference	337
7.84	src/library/RJungleIO.h File Reference	337
7.85	src/library/RJungleOutlier.cpp File Reference	338
7.86	src/library/RJungleOutlier.h File Reference	339
7.87	src/library/RJunglePar.cpp File Reference	339
7.88	src/library/RJunglePar.h File Reference	339
7.88.1	Define Documentation	340
7.88.1.1	bool	340
7.88.2	Typedef Documentation	341
7.88.2.1	RJunglePar	341
7.89	src/library/RJunglePrediction.cpp File Reference	341
7.90	src/library/RJunglePrediction.h File Reference	341
7.91	src/library/RJungleProto.cpp File Reference	342
7.92	src/library/RJungleProto.h File Reference	343
7.93	src/library/RJungleProxi.cpp File Reference	343
7.94	src/library/RJungleProxi.h File Reference	343
7.95	src/library/RJungleTuneMtry.cpp File Reference	344
7.96	src/library/RJungleTuneMtry.h File Reference	345
7.97	src/library/SaveCollector.cpp File Reference	345
7.98	src/library/SaveCollector.h File Reference	346
7.98.1	Function Documentation	347

7.98.1.1 operator<<	347
7.99 src/library/SClassAtom.cpp File Reference	347
7.100src/library/SClassAtom.h File Reference	347
7.100.1 Define Documentation	348
7.100.1.1 CCvector	348
7.100.1.2 Tvector	348
7.100.2 Function Documentation	348
7.100.2.1 operator<<	348
7.101src/library/T2ClassAtom.cpp File Reference	349
7.102src/library/T2ClassAtom.h File Reference	349
7.102.1 Function Documentation	350
7.102.1.1 operator<<	350
7.103src/library/TermClassAtom.cpp File Reference	350
7.104src/library/TermClassAtom.h File Reference	350
7.105src/library/TermResult.cpp File Reference	351
7.106src/library/TermResult.h File Reference	351
7.107src/library/TestClass.cpp File Reference	352
7.108src/library/TestClass.h File Reference	353
7.109src/library/TimeProf.cpp File Reference	354
7.110src/library/TimeProf.h File Reference	354
7.110.1 Define Documentation	355
7.110.1.1 TIMEPROF_START	355
7.110.1.2 TIMEPROF_STOP	356
7.111src/library/TImportance.cpp File Reference	356
7.112src/library/TImportance.h File Reference	356
7.113src/library/TMClassAtom.cpp File Reference	356
7.114src/library/TMClassAtom.h File Reference	356
7.114.1 Define Documentation	358
7.114.1.1 CCvector	358
7.114.1.2 Tvector	358
7.114.2 Function Documentation	358
7.114.2.1 operator<<	358
7.115src/library/Tree.cpp File Reference	358
7.116src/library/Tree.h File Reference	358

7.116.1 Function Documentation	359
7.116.1.1 operator<<	359
7.117src/library/treedefs.h File Reference	359
7.117.1 Define Documentation	360
7.117.1.1 __cdecl	360
7.117.1.2 __lrj	360
7.117.1.3 BACKWARDELIMINATION	361
7.117.1.4 CLASSATOMTYPE	361
7.117.1.5 GSL_RNG_SEED	361
7.117.1.6 IMPORTANCEMEASURE	361
7.117.1.7 LVLOFMEASUREMENT	361
7.117.1.8 MISSINGCODE	361
7.117.1.9 MPI_TAG_BITMATRIX_DEP	361
7.117.1.10MPI_TAG_BITMATRIX_NCOL	361
7.117.1.11MPI_TAG_BITMATRIX_NROW	361
7.117.1.12MPI_TAG_DATATREESET_SLAVEID	361
7.117.1.13MPI_TAG_EX2B	362
7.117.1.14MPI_TAG_EX2L	362
7.117.1.15MPI_TAG_EXV	362
7.117.1.16MPI_TAG_NTREE	362
7.117.1.17MPI_TAG_PREDICTION_ACC	362
7.117.1.18MPI_TAG_PREDICTION_LEN	362
7.117.1.19MPI_TAG_PREDICTION_PRED	362
7.117.1.20MYDATATYPE	362
7.117.1.21SAVECOLLECTORTYPE	362
7.117.1.22TERMINALNODE	362
7.117.1.23TREETYPE	363
7.117.1.24UZI	363
7.117.1.25VARPROXI	363
7.117.1.26VARTYPE	363
7.117.2 Typedef Documentation	363
7.117.2.1 uli_t	363
7.117.3 Enumeration Type Documentation	363
7.117.3.1 BackwardElimination	363

CONTENTS

7.117.3.2 ClassAtomType	363
7.117.3.3 ImportanceMeasure	364
7.117.3.4 LvlOfMeasurement	364
7.117.3.5 SaveCollectorType	364
7.117.3.6 TreeType	365
7.117.3.7 VarProxi	365
7.117.3.8 VarType	365

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

rjungle	11
rjungle::Generator	12
rjungle::Generator::CART	12
rjungle::Generator::CARTcor Experimental fitting function	13
rjungle::Generator::CARTreg	13
rjungle::Generator::CARTregTwoing	14
rjungle::Generator::CARTregWithMiss Experimental fitting function	14
rjungle::Generator::CARTsrt Assign all CART (y=nominal/x=numeric) specific functions	15
rjungle::Generator::CARTsse	15
rjungle::Generator::CARTtwoing	16
rjungle::Generator::CARTtwoWayIntAdd Experimental fitting function	16
rjungle::Generator::CICase1 Conditional Inference trees	17
rjungle::Generator::CICase3 Conditional Inference trees	18
rjungle::Generator::CISNP Conditional Inference trees for SNP data	18
rjungle::Generator::imputeTree Experimental fitting function	19
rjungle::Generator::LOTUS Experimental fitting function	19
rjungle::Generator::SCART Experimental fitting function	20
rjungle::Generator::trendCART Experimental fitting function	20

rjungle::Generator::UCART		
Experimental fitting function	21
TermResult		
A set of functions which yield TermClassAtoms for terminal nodes	..	22

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

BitMatrix	25
DataTreeSet	88
BuildinGenFct< T >	32
ClassAtom< T, C >	37
TermClassAtom< T, C >	247
LotusTermClassAtom< T, C >	156
ClassAtom< T, uli_t >	37
IAMClassAtom< T >	128
SClassAtom< T >	230
T2ClassAtom< T >	240
TMClassAtom< T >	263
CmpldTree< T >	46
CmplFct	53
CondInf< T >	61
DataFrame< T >	74
Exception	92
FittingFct	93
FittingFctPar< T >	97
Generator< T >	102
Helper	103
Importance< T >	135
IAM2WayImportance< T >	123
TImportance< T >	257
PermlImportance< T >	164
JTreeCtrl< T >	146
JTreeTweaks	153
Logistic	154
Node< T, C >	160

Node< T, uli_t >	160
INode< T >	141
Prediction< T >	171
Profiler	176
ProtCount	177
Proximities< T >	179
RJungleAcc< T >	183
RJungleBinder< T >	184
RJungleCompiler< T >	188
RJungleConfusion< T >	189
RJungleCtrl< T >	190
RJungleFromXML< T >	192
RJungleGen< T >	198
RJungleGrow< T >	200
RJungleHelper< T >	202
RJungleImportance< T >	204
RJungleImpute< T >	206
RJungleIO	207
RJungleOutlier	211
RJunglePar	212
RJunglePrediction< T >	219
RJungleProto< T >	220
RJungleProxi< T >	221
RJungleTuneMtry< T >	224
SaveCollector	225
StringIterator	237
TestClass	252
TimeProf	255
Tree< T, C >	270

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

BitMatrix	Representation of a bit matrix that can store bits exclusively	25
BuildinGenFct< T >	A binder for all tree specific functions	32
ClassAtom< T, C >	Smallest classifier unit	37
CmpldTree< T >	46
CmplFct	Representation of a set of compiler functions	53
CondInf< T >	CI Functions for various cases as follows:	61
DataFrame< T >	74
DataTreeSet	88
Exception	92
FittingFct	93
FittingFctPar< T >	97
Generator< T >	102
Helper	103
IAM2WayImportance< T >	123
IAMClassAtom< T >	128
Importance< T >	135
INode< T >	141
JTreeCtrl< T >	146
JTreeTweaks	153
Logistic	154
LotusTermClassAtom< T, C >	156
Node< T, C >	160
PermlImportance< T >	164
Prediction< T >	171

Profiler	176
ProtCount	177
Proximities< T >	179
RJungleAcc< T >	183
RJungleBinder< T >	184
RJungleCompiler< T >	188
RJungleConfusion< T >	189
RJungleCtrl< T >	190
RJungleFromXML< T >	192
RJungleGen< T >	198
RJungleGrow< T >	200
RJungleHelper< T >	202
RJungleImportance< T >	204
RJungleImpute< T >	206
RJungleIO	207
RJungleOutlier	211
RJunglePar	212
RJunglePrediction< T >	219
RJungleProto< T >	220
RJungleProxi< T >	221
RJungleTuneMtry< T >	224
SaveCollector	225
SClassAtom< T >	230
StringIterator	
A class for iterating through tokens from a c-string	237
T2ClassAtom< T >	240
TermClassAtom< T, C >	247
TestClass	252
TimeProf	255
TImportance< T >	257
TMClassAtom< T >	263
Tree< T, C >	270

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

src/library/ BitMatrix.cpp	273
src/library/ BitMatrix.h	273
src/library/ BuildinGenerators.cpp	275
src/library/ BuildinGenerators.h	275
src/library/ ClassAtom.cpp	277
src/library/ ClassAtom.h	278
src/library/ CmpldTree.cpp	279
src/library/ CmpldTree.h	279
src/library/ CmplFct.cpp	280
src/library/ CmplFct.h	280
src/library/ CondInf.cpp	281
src/library/ CondInf.h	281
src/library/ DataFrame.cpp	282
src/library/ DataFrame.h	282
src/library/ DataTreeSet.cpp	283
src/library/ DataTreeSet.h	283
src/library/ ErrorCodes.h	285
src/library/ Exception.cpp	294
src/library/ Exception.h	294
src/library/ FittingFct.cpp	295
src/library/ FittingFct.h	295
src/library/ Generator.cpp	296
src/library/ Generator.h	296
src/library/ gzstream.cpp	297
src/library/ gzstream.h	297
src/library/ Helper.cpp	298
src/library/ Helper.h	298
src/library/ IAM2WayImportance.cpp	299
src/library/ IAM2WayImportance.h	299

src/library/IAMClassAtom.cpp	300
src/library/IAMClassAtom.h	300
src/library/Importance.cpp	302
src/library/Importance.h	302
src/library/init.h	302
src/library/INode.cpp	304
src/library/INode.h	304
src/library/ioLines.cpp	305
src/library/ioLines.h	305
src/library/JTreeCtrl.h	306
src/library/JTreeTweaks.cpp	307
src/library/JTreeTweaks.h	308
src/library/librjungle.cpp	308
src/library/librjungle.h	309
src/library/Logistic.cpp	310
src/library/Logistic.h	311
src/library/LotusTermClassAtom.cpp	312
src/library/LotusTermClassAtom.h	312
src/library/mvt.cpp	314
src/library/mvt.h	315
src/library/mvtfortran.f	316
src/library/Node.cpp	320
src/library/Node.h	320
src/library/PermlImportance.cpp	321
src/library/PermlImportance.h	321
src/library/Prediction.cpp	322
src/library/Prediction.h	322
src/library/Profiler.cpp	323
src/library/Profiler.h	324
src/library/Proximities.cpp	324
src/library/Proximities.h	324
src/library/RJungleAcc.cpp	325
src/library/RJungleAcc.h	325
src/library/RJungleBinder.cpp	326
src/library/RJungleBinder.h	326
src/library/RJungleCompiler.cpp	327
src/library/RJungleCompiler.h	327
src/library/RJungleConfusion.cpp	329
src/library/RJungleConfusion.h	329
src/library/RJungleCtrl.cpp	330
src/library/RJungleCtrl.h	330
src/library/RJungleFromXML.cpp	331
src/library/RJungleFromXML.h	331
src/library/RJungleGen.cpp	333
src/library/RJungleGen.h	333
src/library/RJungleGrow.cpp	333
src/library/RJungleGrow.h	333
src/library/RJungleHelper.cpp	334
src/library/RJungleHelper.h	334
src/library/RJungleImportance.cpp	335

src/library/RJungleImportance.h	335
src/library/RJungleImpute.cpp	336
src/library/RJungleImpute.h	336
src/library/RJungleIO.cpp	337
src/library/RJungleIO.h	337
src/library/RJungleOutlier.cpp	338
src/library/RJungleOutlier.h	339
src/library/RJunglePar.cpp	339
src/library/RJunglePar.h	339
src/library/RJunglePrediction.cpp	341
src/library/RJunglePrediction.h	341
src/library/RJungleProto.cpp	342
src/library/RJungleProto.h	343
src/library/RJungleProxi.cpp	343
src/library/RJungleProxi.h	343
src/library/RJungleTuneMtry.cpp	344
src/library/RJungleTuneMtry.h	345
src/library/SaveCollector.cpp	345
src/library/SaveCollector.h	346
src/library/SClassAtom.cpp	347
src/library/SClassAtom.h	347
src/library/T2ClassAtom.cpp	349
src/library/T2ClassAtom.h	349
src/library/TermClassAtom.cpp	350
src/library/TermClassAtom.h	350
src/library/TermResult.cpp	351
src/library/TermResult.h	351
src/library/TestClass.cpp	352
src/library/TestClass.h	353
src/library/TimeProf.cpp	354
src/library/TimeProf.h	354
src/library/TImportance.cpp	356
src/library/TImportance.h	356
src/library/TMClassAtom.cpp	356
src/library/TMClassAtom.h	356
src/library/Tree.cpp	358
src/library/Tree.h	358
src/library/treedefs.h	359

Chapter 5

Namespace Documentation

5.1 rjungle Namespace Reference

Namespaces

- namespace [Generator](#)

Functions

- template<class T >
T [magicAt](#) (T *vec, [uli_t](#) j)

Variables

- static const char [strLom](#) [4][8]

5.1.1 Function Documentation

5.1.1.1 template<class T > T [rjungle::magicAt](#) (T * vec, [uli_t](#) j) [inline]

Definition at line 219 of file init.h.

5.1.2 Variable Documentation

5.1.2.1 const char [rjungle::strLom](#)[4][8] [static]

Initial value:

```
{  
    "undef", "nominal", "ordinal", "numeric" }
```

Definition at line 224 of file init.h.

5.2 rjungle::Generator Namespace Reference

Namespaces

- namespace **CART**
- namespace **CARTcor**

Experimental fitting function.
- namespace **CARTreg**
- namespace **CARTregTwoing**
- namespace **CARTregWithMiss**

Experimental fitting function.
- namespace **CARTsrt**

*Assign all **CART** ($y=nominal/x=numerical$) specific functions.*
- namespace **CARTsse**
- namespace **CARTtwoing**
- namespace **CARTTwoWayIntAdd**

Experimental fitting function.
- namespace **CICase1**

Conditional Inference trees.
- namespace **CICase3**

Conditional Inference trees.
- namespace **CISNP**

Conditional Inference trees for SNP data.
- namespace **imputeTree**

Experimental fitting function.
- namespace **LOTUS**

Experimental fitting function.
- namespace **SCART**

Experimental fitting function.
- namespace **trendCART**

Experimental fitting function.
- namespace **UCART**

Experimental fitting function.

5.3 rjungle::Generator::CART Namespace Reference

Functions

- template<class T >
- void **assignFunctions** (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nominal/x=numerical$) specific functions.*

5.3.1 Function Documentation

5.3.1.1 `template<class T > void rjungle::Generator::CART::assignFunctions (BuildinGenFct< T > * genFct)`

Assign all `CART` ($y=nominal/x=numerical$) specific functions.

Parameters

<code>genFct</code>	<code>Generator</code> object to be set.
---------------------	--

Definition at line 138 of file BuildinGenerators.h.

5.4 rjungle::Generator::CARTcor Namespace Reference

Experimental fitting function.

Functions

- `template<class T >`
`void assignFunctions (BuildinGenFct< T > *genFct)`

5.4.1 Detailed Description

Experimental fitting function.

Parameters

<code>genFct</code>	<code>Generator</code> object to be set.
---------------------	--

5.4.2 Function Documentation

5.4.2.1 `template<class T > void rjungle::Generator::CARTcor::assignFunctions (BuildinGenFct< T > * genFct)`

Definition at line 296 of file BuildinGenerators.h.

5.5 rjungle::Generator::CARTreg Namespace Reference

Functions

- `template<class T >`
`void assignFunctions (BuildinGenFct< T > *genFct)`

Assign all `CART` ($y=numerical/x=numerical$) specific functions.

5.5.1 Function Documentation

5.5.1.1 `template<class T > void rjungle::Generator::CARTreg::assignFunctions (BuildinGenFct< T > * genFct)`

Assign all **CART** (y=nemonic/x=nemonic) specific functions.

Parameters

<code>genFct</code>	<code>Generator</code> object to be set.
---------------------	--

Definition at line 216 of file BuildinGenerators.h.

5.6 rjungle::Generator::CARTregTwoing Namespace Reference

Functions

- `template<class T >`
`void assignFunctions (BuildinGenFct< T > *genFct)`

*Assign all **CART** (y=nemonic/x=nominal) specific functions.*

5.6.1 Function Documentation

5.6.1.1 `template<class T > void rjungle::Generator::CARTreg-
Twoing::assignFunctions (BuildinGenFct< T > * genFct
)`

Assign all **CART** (y=nemonic/x=nominal) specific functions.

Parameters

<code>genFct</code>	<code>Generator</code> object to be set.
---------------------	--

Definition at line 242 of file BuildinGenerators.h.

5.7 rjungle::Generator::CARTregWithMiss Namespace Reference

Experimental fitting function.

Functions

- `template<class T >`
`void assignFunctions (BuildinGenFct< T > *genFct)`

5.7.1 Detailed Description

Experimental fitting function.

Parameters

<i>genFct</i>	Generator object to be set.
---------------	-----------------------------

5.7.2 Function Documentation

5.7.2.1 template<class T > void rjungle::Generator::CARTregWithMiss::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 537 of file BuildinGenerators.h.

5.8 rjungle::Generator::CARTsrt Namespace Reference

Assign all **CART** (y=nominal/x=numeric) specific functions.

Functions

- template<class T >
void **assignFunctions** (BuildinGenFct< T > **genFct*)

5.8.1 Detailed Description

Assign all **CART** (y=nominal/x=numeric) specific functions.

Parameters

<i>genFct</i>	Generator object to be set.
---------------	-----------------------------

5.8.2 Function Documentation

5.8.2.1 template<class T > void rjungle::Generator::CARTsrt::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 271 of file BuildinGenerators.h.

5.9 rjungle::Generator::CARTsse Namespace Reference

Functions

- template<class T >
void **assignFunctions** (BuildinGenFct< T > *genFct)
*Assign all **CART** (y=nominal/x=numeric) specific functions.*

5.9.1 Function Documentation

5.9.1.1 template<class T > void rjungle::Generator::CARTsse::assignFunctions (BuildinGenFct< T > * genFct)

Assign all **CART** (y=nominal/x=numeric) specific functions.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	------------------------------------

Definition at line 164 of file BuildinGenerators.h.

5.10 rjungle::Generator::CARTtwoing Namespace Reference

Functions

- template<class T >
void **assignFunctions** (BuildinGenFct< T > *genFct)
*Assign all **CART** (y=nominal/x=nominal) specific functions.*

5.10.1 Function Documentation

5.10.1.1 template<class T > void rjungle::Generator::CARTtwoing::assignFunctions (BuildinGenFct< T > * genFct)

Assign all **CART** (y=nominal/x=nominal) specific functions.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	------------------------------------

Definition at line 190 of file BuildinGenerators.h.

5.11 rjungle::Generator::CARTtwoWayIntAdd Namespace Reference

Experimental fitting function.

Functions

- template<class T >
void [assignFunctions](#) ([BuildinGenFct](#)< T > *genFct)

5.11.1 Detailed Description

Experimental fitting function.

Parameters

<i>genFct</i>	Generator object to be set.
---------------	---

5.11.2 Function Documentation

- 5.11.2.1 template<class T > void rjungle::Generator::CARTtwoWay-
[IntAdd](#)::assignFunctions ([BuildinGenFct](#)< T > * *genFct*
)

Definition at line 562 of file BuildinGenerators.h.

5.12 rjungle::Generator::CICase1 Namespace Reference

Conditional Inference trees.

Functions

- template<class T >
void [assignFunctions](#) ([BuildinGenFct](#)< T > *genFct)

5.12.1 Detailed Description

Conditional Inference trees. Restrictions as follows:

Variables: y (nominal) -> h(y,...) = e_k(y) x_j (numeric/ordered) -> g(x) = x or rank(x)

Parameters

<i>genFct</i>	Generator object to be set.
---------------	---

5.12.2 Function Documentation

5.12.2.1 template<class T > void rjungle::Generator::CICase1::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 354 of file BuildinGenerators.h.

5.13 rjungle::Generator::CICase3 Namespace Reference

Conditional Inference trees.

Functions

- template<class T >
void [assignFunctions](#) (BuildinGenFct< T > **genFct*)

5.13.1 Detailed Description

Conditional Inference trees. Restrictions as follows:

Variables: y (numeric) -> h(y,...) = y x_j (numeric) -> g(x) = x

Parameters

<i>genFct</i>	Generator object to be set.
---------------	---

5.13.2 Function Documentation

5.13.2.1 template<class T > void rjungle::Generator::CICase3::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 383 of file BuildinGenerators.h.

5.14 rjungle::Generator::CISNP Namespace Reference

Conditional Inference trees for SNP data.

Functions

- template<class T >
void [assignFunctions](#) (BuildinGenFct< T > **genFct*)

5.14.1 Detailed Description

Conditional Inference trees for SNP data.

Parameters

<i>genFct</i>	Generator object to be set.
---------------	---

5.14.2 Function Documentation

5.14.2.1 template<class T > void rjungle::Generator::CISNP::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 325 of file BuildinGenerators.h.

5.15 rjungle::Generator::imputeTree Namespace Reference

Experimental fitting function.

Functions

- template<class T >
void [assignFunctions](#) (BuildinGenFct< T > **genFct*)

5.15.1 Detailed Description

Experimental fitting function.

Parameters

<i>genFct</i>	Generator object to be set.
---------------	---

5.15.2 Function Documentation

5.15.2.1 template<class T > void rjungle::Generator::imputeTree::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 512 of file BuildinGenerators.h.

5.16 rjungle::Generator::LOTUS Namespace Reference

Experimental fitting function.

Functions

- template<class T >
void [assignFunctions](#) (BuildinGenFct< T > **genFct*)

5.16.1 Detailed Description

Experimental fitting function. Implements [LOTUS](#) trees.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	---

5.16.2 Function Documentation

5.16.2.1 `template<class T > void rjungle::Generator::LOTUS::assignFunctions (BuildinGenFct< T > * genFct)`

Definition at line 412 of file BuildinGenerators.h.

5.17 rjungle::Generator::SCART Namespace Reference

Experimental fitting function.

Functions

- `template<class T >`
`void assignFunctions (BuildinGenFct< T > *genFct)`

5.17.1 Detailed Description

Experimental fitting function.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	---

5.17.2 Function Documentation

5.17.2.1 `template<class T > void rjungle::Generator::SCART::assignFunctions (BuildinGenFct< T > * genFct)`

Definition at line 462 of file BuildinGenerators.h.

5.18 rjungle::Generator::trendCART Namespace Reference

Experimental fitting function.

Functions

- template<class T >
void [assignFunctions](#) ([BuildinGenFct](#)< T > *genFct)

5.18.1 Detailed Description

Experimental fitting function.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	---

5.18.2 Function Documentation

5.18.2.1 template<class T > void rjungle::Generator::trendCART::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 487 of file BuildinGenerators.h.

5.19 rjungle::Generator::UCART Namespace Reference

Experimental fitting function.

Functions

- template<class T >
void [assignFunctions](#) ([BuildinGenFct](#)< T > *genFct)

5.19.1 Detailed Description

Experimental fitting function.

Parameters

<code>genFct</code>	Generator object to be set.
---------------------	---

5.19.2 Function Documentation

5.19.2.1 template<class T > void rjungle::Generator::UCART::assignFunctions (BuildinGenFct< T > * *genFct*)

Definition at line 437 of file BuildinGenerators.h.

5.20 TermResult Namespace Reference

A set of functions which yield TermClassAtoms for terminal nodes.

Functions

- template<class T >
`TermClassAtom< T, uli_t > * getTermMostFreq (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
Get most frequent value in vector.
- template<class T >
`TermClassAtom< T, uli_t > * getTermMean (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
Get mean of vector.
- template<class T >
`TermClassAtom< T, uli_t > * getTermLotus (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
- template<class T >
`T getMostFreq (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)`
Get most frequent value in vector.
- template<class T >
`static T getMean (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)`
Get mean of vector.

5.20.1 Detailed Description

A set of functions which yield TermClassAtoms for terminal nodes.

5.20.2 Function Documentation

- 5.20.2.1 template<class T > static T TermResult::getMean (DataFrame< T > & data, RJunglePar & par, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< double > & rowWeight) [static]

Get mean of vector.

Parameters

<code>dataVec</code>	input vector
<code>missing-Code</code>	missing code

Returns

mean value

Definition at line 245 of file TermResult.h.

5.20.2.2 template<class T > T TermResult::getMostFreq (DataFrame< T > & *data*, RJunglePar & *par*, uli_t *col*, std::vector< uli_t > & *rowMaskVec*, std::vector< double > & *rowWeight*)

Get most frequent value in vector.

Choose value at random if multiple values are most frequent.

Parameters

<i>data</i>	global data
<i>par</i>	random jungle paramter
<i>col</i>	input vector
<i>rowMaskVec</i>	selection of samples
<i>rowWeight</i>	sample weights

Returns

Definition at line 194 of file TermResult.h.

5.20.2.3 template<class T > TermClassAtom<T, uli_t>* TermResult::getTermLotus (DataFrame< T > & *data*, RJunglePar & *par*, uli_t *col*, std::vector< uli_t > & *rowMaskVec*, std::vector< uli_t > & *colMaskVec*, std::vector< double > & *rowWeight*)

Definition at line 88 of file TermResult.h.

5.20.2.4 template<class T > TermClassAtom<T, uli_t>* TermResult::getTermMean (DataFrame< T > & *data*, RJunglePar & *par*, uli_t *col*, std::vector< uli_t > & *rowMaskVec*, std::vector< uli_t > & *colMaskVec*, std::vector< double > & *rowWeight*)

Get mean of vector.

Parameters

<i>dataVec</i>	input vector
<i>missing-Code</i>	missing code

Returns

mean value

Definition at line 76 of file TermResult.h.

5.20.2.5 template<class T> TermClassAtom<T, uli_t>* TermResult::getTermMost-Freq (DataFrame< T > & data, RJunglePar & par, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< uli_t > & colMaskVec, std::vector< double > & rowWeight)

Get most frequent value in vector.

Choose value at random if multiple values are most frequent.

Parameters

<i>data</i>	global data
<i>par</i>	random jungle paramter
<i>col</i>	input vector
<i>rowMaskVec</i>	selection of samples
<i>rowWeight</i>	sample weights

Returns

Definition at line 57 of file TermResult.h.

Chapter 6

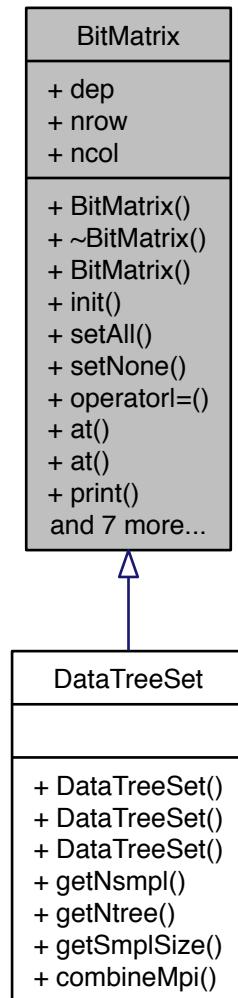
Class Documentation

6.1 BitMatrix Class Reference

Representation of a bit matrix that can store bits exclusively.

```
#include <BitMatrix.h>
```

Inheritance diagram for BitMatrix:



Public Member Functions

- [`BitMatrix \(\)`](#)
Constructor that sets the data dimensions to 0.
- [`virtual ~BitMatrix \(\)`](#)
Destructor.

- **BitMatrix (uli_t nrow, uli_t ncol)**
Constructor that sets the data dimensions.
- **void init (uli_t nrow, uli_t ncol)**
Initializes the bit matrix with "false".
- **void setAll ()**
Sets all cells in bit matrix to "true".
- **void setNone ()**
Sets all cells in bit matrix to "false".
- **void operator|= (const BitMatrix &dataSet)**
Adds the given data to bit matrix (OR operator)
- **bool at (uli_t row, uli_t col) const**
Returns logical value at a specific position in bit matrix.
- **void at (uli_t row, uli_t col, bool val)**
Sets logical value at a specific position.
- **std::ostream & print (std::ostream &os) const**
Prints the content of bit matrix to output stream.
- **uli_t getSizeOfRow (unsigned int i)**
Returns the sum of row i.
- **uli_t getSizeOfCol (unsigned int j)**
Returns the sum of column j.
- **uli_t getNrow ()**
Returns the row number of bit matrix.
- **uli_t getNcol ()**
Returns the column number of bit matrix.
- **uli_t count ()**
Returns the total number of "true"s in bit matrix.
- **virtual void sendMpi ()**
Sends bit matrix to master process.
- **virtual void receiveMpi (int mpild)**
Receives bit matrix from a slave.
- **virtual std::string receiveStrMpi (int mpild)**
Receives bit matrix from a slave.

Public Attributes

- **boost::dynamic_bitset dep**
bit matrix containing data-col dependencies
- **uli_t nrow**
Row number of bit matrix.
- **uli_t ncol**
Column number of bit matrix.

6.1.1 Detailed Description

Representation of a bit matrix that can store bits exclusively.

Definition at line 43 of file BitMatrix.h.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 **BitMatrix::BitMatrix() [inline]**

Constructor that sets the data dimensions to 0.

Definition at line 48 of file BitMatrix.h.

6.1.2.2 **virtual BitMatrix::~BitMatrix() [inline, virtual]**

Destructor.

Definition at line 53 of file BitMatrix.h.

6.1.2.3 **BitMatrix::BitMatrix(uli_t nrow, uli_t ncol) [inline]**

Constructor that sets the data dimensions.

Parameters

<i>nrow</i>	Number of rows.
<i>ncol</i>	Number of columns.

Definition at line 61 of file BitMatrix.h.

6.1.3 Member Function Documentation

6.1.3.1 **bool BitMatrix::at(uli_t row, uli_t col) const [inline]**

Returns logical value at a specific position in bit matrix.

Parameters

<i>row</i>	Row position in bit matrix.
<i>col</i>	Column position in bit matrix.

Returns

Logical value at that specific position

Definition at line 112 of file BitMatrix.h.

6.1.3.2 void BitMatrix::at(uli_t row, uli_t col, bool val) [inline]

Sets logical value at a specific position.

Parameters

<i>row</i>	Row position in bit matrix.
<i>col</i>	Column position in bit matrix.
<i>val</i>	Value to be set.

Definition at line 123 of file BitMatrix.h.

6.1.3.3 uli_t BitMatrix::count() [inline]

Returns the total number of "true"s in bit matrix.

Returns

Total number of "true"

Definition at line 200 of file BitMatrix.h.

6.1.3.4 uli_t BitMatrix::getNcol() [inline]

Returns the column number of bit matrix.

Returns

Number of columns.

Definition at line 193 of file BitMatrix.h.

6.1.3.5 uli_t BitMatrix::getNrow() [inline]

Returns the row number of bit matrix.

Returns

Number of rows.

Definition at line 186 of file BitMatrix.h.

6.1.3.6 uli_t BitMatrix::getSizeOfCol(unsigned int j) [inline]

Returns the sum of column j.

Parameters

<i>j</i>	Index of column.
----------	------------------

Returns

Sum of column j.

Definition at line 171 of file BitMatrix.h.

6.1.3.7 uli_t BitMatrix::getSizeOfRow (unsigned int *i*) [inline]

Returns the sum of row i.

Parameters

<i>i</i>	Index of row.
----------	---------------

Returns

Sum of row i

Definition at line 154 of file BitMatrix.h.

6.1.3.8 void BitMatrix::init (uli_t *nrow*, uli_t *ncol*) [inline]

Initializes the bit matrix with "false".

Dimensions have to be given and old matrix data will be deleted.

Parameters

<i>nrow.</i>	
<i>ncol.</i>	

Definition at line 73 of file BitMatrix.h.

6.1.3.9 void BitMatrix::operator|= (const BitMatrix & *dataSet*) [inline]

Adds the given data to bit matrix (OR operator)

Parameters

<i>dataSet</i>	A bit matrix that should be added.
----------------	------------------------------------

Definition at line 99 of file BitMatrix.h.

6.1.3.10 `std::ostream& BitMatrix::print(std::ostream & os) const [inline]`

Prints the content of bit matrix to output stream.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream

Definition at line 135 of file BitMatrix.h.

6.1.3.11 `virtual void BitMatrix::receiveMpi(int mpild) [inline, virtual]`

Receives bit matrix from a slave.

Parameters

<code>mpild</code>	Slave id.
--------------------	-----------

Definition at line 228 of file BitMatrix.h.

6.1.3.12 `virtual std::string BitMatrix::receiveStrMpi(int mpild) [inline, virtual]`

Receives bit matrix from a slave.

Matrix is coded as a string.

Parameters

<code>mpild</code>	Slave id.
--------------------	-----------

Definition at line 256 of file BitMatrix.h.

6.1.3.13 `virtual void BitMatrix::sendMpi() [inline, virtual]`

Sends bit matrix to master process.

Definition at line 205 of file BitMatrix.h.

6.1.3.14 `void BitMatrix::setAll() [inline]`

Sets all cells in bit matrix to "true".

Definition at line 83 of file BitMatrix.h.

6.1.3.15 void BitMatrix::setNone() [inline]

Sets all cells in bit matrix to "false".

Definition at line 90 of file BitMatrix.h.

6.1.4 Member Data Documentation

6.1.4.1 boost::dynamic_bitset BitMatrix::dep

bit matrix containing data-col dependencies

Definition at line 282 of file BitMatrix.h.

6.1.4.2 uli_t BitMatrix::ncol

Column number of bit matrix.

Definition at line 288 of file BitMatrix.h.

6.1.4.3 uli_t BitMatrix::nrow

Row number of bit matrix.

Definition at line 285 of file BitMatrix.h.

The documentation for this class was generated from the following file:

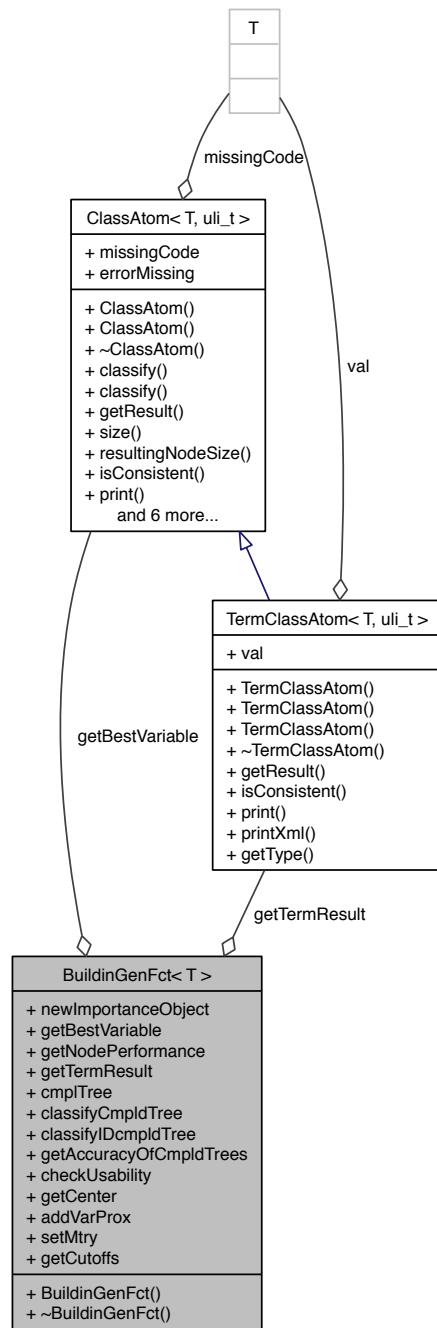
- src/library/[BitMatrix.h](#)

6.2 BuildinGenFct< T > Class Template Reference

A binder for all tree specific functions.

```
#include <BuildinGenerators.h>
```

Collaboration diagram for BuildinGenFct< T >:



Public Member Functions

- `BuildinGenFct ()`
Constructor the resets all pointer to NULL.
- `virtual ~BuildinGenFct ()`
Destructor that does nothing.

Public Attributes

- `Importance< T > *(* newImportanceObject)(void)`
Importance score structure and functions.
- `ClassAtom< T, uli_t > *(* getBestVariable)(FittingFctPar< T > &fitFctPar)`
Fitting function that determines the "best" variable in data set.
- `double(* getNodePerformance)(DataFrame< T > &data, uli_t depVar, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)`
Function that calculates the performance of the current node.
- `TermClassAtom< T, uli_t > *(* getTermResult)(DataFrame< T > &data, R-JunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
Fitting function that determines the "best" variable in data set.
- `CmpldTree< T > *(* cmplTree)(Tree< T, uli_t > &tree)`
Function that compiles a pointer based tree to a flat data structure.
- `T(* classifyCmpldTree)(T *sample, CmpldTree< T > *cmpldTree)`
Classifier function that classifies a sample using a compiled tree.
- `unsigned int(* classifyIDcmpldTree)(T *sample, CmpldTree< T > *cmpldTree)`
Classifier function that classifies a sample using a compiled tree and returns the id of the result.
- `double(* getAccuracyOfCmpldTrees)(DataFrame< T > *data, T *pred, uli_t depVar, DataTreeSet &dataSet, long int treId, bool one, bool showInfos, R-JungleIO &io)`
Function that returns the accuracy of current tree(s).
- `void(* checkUsability)(DataFrame< T > &data)`
Pre function that checks before tree growing if data is suitable for specific tree type.
- `double(* getCenter)(std::vector< T > &vec)`
Function that returns the center of a vector such as mean or median.
- `void(* addVarProx)(DataFrame< T > &data, Proximities< T > &proxi, CmpldTree< T > *cmpldTree, std::vector< uli_t > *colMask)`
Function that updates the proximity matrix.
- `uli_t(* setMtry)(uli_t ncol)`
Function that calculates the default mtry value.
- `void(* getCutoffs)(CmpldTree< T > &cmpldTree, uli_t var, std::vector< T > &cutoffs)`
Function that returns all cutoffs within a tree. (E.g. for conditional importance).

6.2.1 Detailed Description

```
template<class T>class BuildinGenFct< T >
```

A binder for all tree specific functions.

This is kind of configuration that will be used to handle several functionalities.

Definition at line 39 of file BuildinGenerators.h.

6.2.2 Constructor & Destructor Documentation

```
6.2.2.1 template<class T> BuildinGenFct< T >::BuildinGenFct( ) [inline]
```

Constructor the resets all pointer to NULL.

Definition at line 45 of file BuildinGenerators.h.

```
6.2.2.2 template<class T> virtual BuildinGenFct< T >::~BuildinGenFct( )  
[inline, virtual]
```

Destructor that does nothing.

Definition at line 56 of file BuildinGenerators.h.

6.2.3 Member Data Documentation

```
6.2.3.1 template<class T> void(* BuildinGenFct< T >::addVarProx)(DataFrame< T >  
&data, Proximities< T > &proxi, CmpIdTree< T > *cmpIdTree, std::vector< uli_t  
> *colMask)
```

Function that updates the proximity matrix.

Definition at line 99 of file BuildinGenerators.h.

```
6.2.3.2 template<class T> void(* BuildinGenFct< T >::checkUsability)(DataFrame<  
T > &data)
```

Pre function that checks before tree growing if data is suitable for specific tree type.

Definition at line 93 of file BuildinGenerators.h.

```
6.2.3.3 template<class T> T(* BuildinGenFct< T >::classifyCmpIdTree)(T *sample,  
CmpIdTree< T > *cmpIdTree)
```

Classifier function that classifies a sample using a compiled tree.

Definition at line 82 of file BuildinGenerators.h.

6.2.3.4 template<class T> unsigned int(* BuildinGenFct< T >::classifyIDcmpldTree)(T *sample, CmpldTree< T > *cmpldTree)

Classifier function that classifies a sample using a compiled tree and returns the id of the result.

Definition at line 85 of file BuildinGenerators.h.

6.2.3.5 template<class T> CmpldTree<T>*(* BuildinGenFct< T >::cmplTree)(Tree< T, uli_t > &tree)

Function that compiles a pointer based tree to a flat data structure.

Definition at line 79 of file BuildinGenerators.h.

6.2.3.6 template<class T> double(* BuildinGenFct< T >::getAccuracyOfCmpldTrees)(DataFrame< T > *data, T *pred, uli_t depVar, DataTreeSet &dataSet, long int treeld, bool one, bool showInfos, RJungleIO &io)

Function that returns the accuracy of current tree(s).

Definition at line 88 of file BuildinGenerators.h.

6.2.3.7 template<class T> ClassAtom<T, uli_t>*(* BuildinGenFct< T >::getBestVariable)(FittingFctPar< T > &fitFctPar)

Fitting function that determines the "best" variable in data set.

Definition at line 64 of file BuildinGenerators.h.

6.2.3.8 template<class T> double(* BuildinGenFct< T >::getCenter)(std::vector< T > &vec)

Function that returns the center of a vector such as mean or median.

Definition at line 96 of file BuildinGenerators.h.

6.2.3.9 template<class T> void(* BuildinGenFct< T >::getCutoffs)(CmpldTree< T > &cmpldTree, uli_t var, std::vector< T > &cutoffs)

Function that returns all cutoffs within a tree. (E.g. for conditional importance).

Definition at line 106 of file BuildinGenerators.h.

6.2.3.10 template<class T> double(* BuildinGenFct< T >::getNodePerformance)(-
 DataFrame< T > &data, uli_t depVar, std::vector< uli_t > &rowMaskVec,
 std::vector< double > &rowWeight)

Function that calculates the performance of the current node.

Definition at line 67 of file BuildinGenerators.h.

6.2.3.11 template<class T> TermClassAtom<T, uli_t>*(* BuildinGenFct< T
 >::getTermResult)(DataFrame< T > &data, RJunglePar &par, uli_t col,
 std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector<
 double > &rowWeight)

Fitting function that determines the "best" variable in data set.

Function that determines the classifier of current terminal node.

Definition at line 74 of file BuildinGenerators.h.

6.2.3.12 template<class T> Importance<T>*(* BuildinGenFct< T
 >::newImportanceObject)(void)

[Importance](#) score structure and functions.

Definition at line 61 of file BuildinGenerators.h.

6.2.3.13 template<class T> uli_t(* BuildinGenFct< T >::setMtry)(uli_t ncol)

Function that calculates the default mtry value.

Definition at line 103 of file BuildinGenerators.h.

The documentation for this class was generated from the following file:

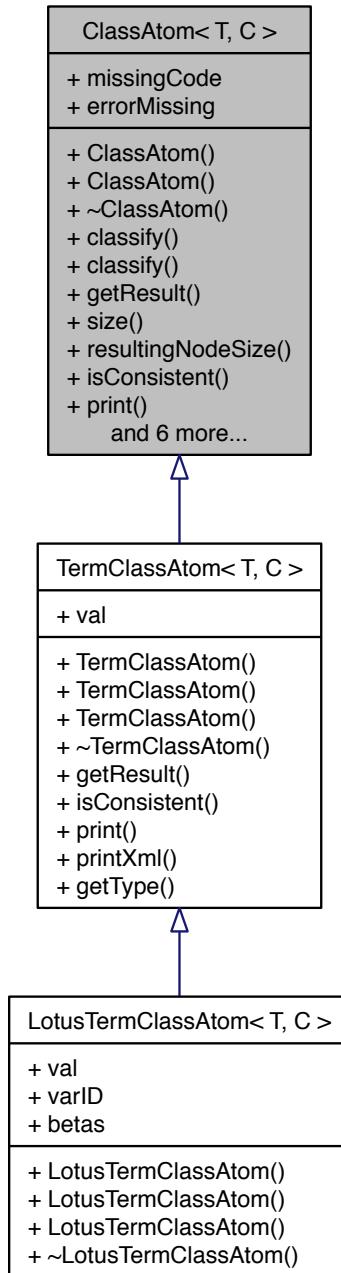
- [src/library/BuildinGenerators.h](#)

6.3 ClassAtom< T, C > Class Template Reference

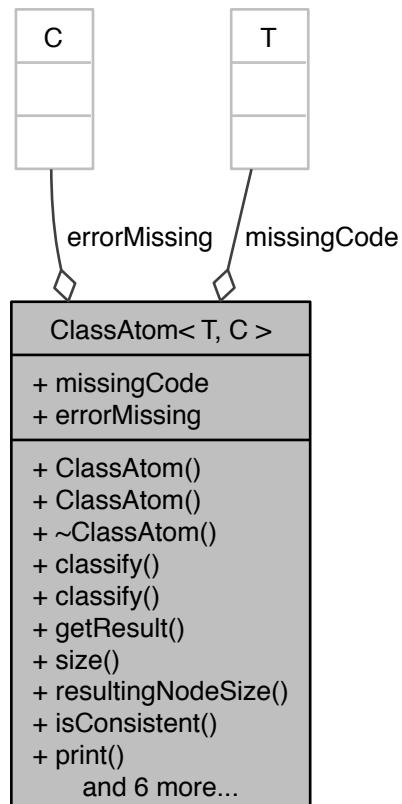
Smallest classifier unit.

```
#include <ClassAtom.h>
```

Inheritance diagram for ClassAtom< T, C >:



Collaboration diagram for ClassAtom< T, C >:



Public Member Functions

- [ClassAtom \(\)](#)
Constructor that sets the internal missing code.
- [ClassAtom \(const ClassAtom< T, C > &classAtom\)](#)
Constructor that sets the internal missing code.
- [virtual ~ClassAtom \(\)](#)
- [virtual C classify \(const std::vector< T > &vecIn\) const](#)
Classifies an input value.
- [virtual C classify \(T *vecIn\) const](#)
Classifies an input value.

- virtual T **getResult** () const
Returns the value of this classifier.
- virtual uli_t **size** () const
Returns the size of classifier.
- virtual uli_t **resultingNodeSize** () const
Returns the size of resulting node.
- virtual bool **isConsistent** () const
Returns if the classifier is consistent.
- virtual std::ostream & **print** (std::ostream &os) const
Prints the classifier to output stream.
- virtual void **printXml** (std::ostream &os) const
Prints the classifier to output stream in XML format.
- virtual uli_t **getType** () const
Returns the type of current classifier.
- virtual void **setMissingCode** (T val)
Sets the value that means "missing value".
- virtual T **getMissingCode** () const
Returns the missing value coding.
- virtual void **setErrorMissing** (C val)
Sets the value that means "error there is a missing value".
- virtual C **getErrorMissing** () const
Returns the error missing value coding.
- virtual bool **isThereVarID** (uli_t varID) const
Return if classifier has got a specific variable ID.

Public Attributes

- T **missingCode**
Missing code.
- C **errorMissing**
Error missing code.

6.3.1 Detailed Description

template<class T, class C>class ClassAtom< T, C >

Smallest classifier unit.

It classify a "value" with a low level function. I.e. classification via threshold or logic regression.

Definition at line 37 of file ClassAtom.h.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 template<class T, class C> **ClassAtom< T, C >::ClassAtom()** [inline]

Constructor that sets the internal missing code.

Definition at line 43 of file ClassAtom.h.

6.3.2.2 template<class T, class C> **ClassAtom< T, C >::ClassAtom(const ClassAtom< T, C > & classAtom)** [inline]

Constructor that sets the internal missing code.

Parameters

<i>Object</i>	of classAtom (tree node classifier).
---------------	--------------------------------------

Definition at line 50 of file ClassAtom.h.

6.3.2.3 template<class T, class C> virtual **ClassAtom< T, C >::~ClassAtom()** [inline, virtual]

Definition at line 51 of file ClassAtom.h.

6.3.3 Member Function Documentation

6.3.3.1 template<class T, class C> virtual C **ClassAtom< T, C >::classify(const std::vector< T > & vecIn) const** [inline, virtual]

Classifies an input value.

Parameters

<i>vecIn</i>	Sample to be classified
--------------	-------------------------

Returns

Classification value

Definition at line 60 of file ClassAtom.h.

6.3.3.2 template<class T, class C> virtual C **ClassAtom< T, C >::classify(T * vecIn) const** [inline, virtual]

Classifies an input value.

Parameters

<code>vecIn</code>	Sample to be classified
--------------------	-------------------------

Returns

Classification value

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), and [T2ClassAtom< T >](#).

Definition at line 71 of file ClassAtom.h.

6.3.3.3 template<class T, class C> virtual C ClassAtom< T, C >::getErrorMissing() const [inline, virtual]

Returns the error missing value coding.

Returns

Error missing value coding.

Definition at line 163 of file ClassAtom.h.

6.3.3.4 template<class T, class C> virtual T ClassAtom< T, C >::getMissingCode() const [inline, virtual]

Returns the missing value coding.

Returns

Missing value coding.

Definition at line 149 of file ClassAtom.h.

6.3.3.5 template<class T, class C> virtual T ClassAtom< T, C >::getResult() const [inline, virtual]

Returns the value of this classifier.

Returns

Classifier value

Reimplemented in [TermClassAtom< T, C >](#), and [TermClassAtom< T, uli_t >](#).

Definition at line 80 of file ClassAtom.h.

6.3.3.6 **template<class T, class C> virtual uli_t ClassAtom< T, C >::getType() const [inline, virtual]**

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), [T2ClassAtom< T >](#), [TermClassAtom< T, C >](#), and [TermClassAtom< T, uli_t >](#).

Definition at line 135 of file ClassAtom.h.

6.3.3.7 **template<class T, class C> virtual bool ClassAtom< T, C >::isConsistent() const [inline, virtual]**

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), [T2ClassAtom< T >](#), [TermClassAtom< T, C >](#), and [TermClassAtom< T, uli_t >](#).

Definition at line 105 of file ClassAtom.h.

6.3.3.8 **template<class T, class C> virtual bool ClassAtom< T, C >::isThereVarID(uli_t varID) const [inline, virtual]**

Return if classifier has got a specific variable ID.

Parameters

<i>varID</i>	ID of variable.
--------------	-----------------

Returns

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), and [T2ClassAtom< T >](#).

Definition at line 172 of file ClassAtom.h.

6.3.3.9 **template<class T, class C> virtual std::ostream& ClassAtom< T, C >::print(std::ostream & os) const [inline, virtual]**

Prints the classifier to output stream.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream.

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), [T2ClassAtom< T >](#), [TermClassAtom< T, C >](#), and [TermClassAtom< T, uli_t >](#).

Definition at line 116 of file ClassAtom.h.

6.3.3.10 template<class T, class C> virtual void ClassAtom< T, C >::printXml (std::ostream & os) const [inline, virtual]

Prints the classifier to output stream in XML format.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream.

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), [T2ClassAtom< T >](#), [TermClassAtom< T, C >](#), and [TermClassAtom< T, uli_t >](#).

Definition at line 127 of file ClassAtom.h.

6.3.3.11 template<class T, class C> virtual uli_t ClassAtom< T, C >::resultingNodeSize () const [inline, virtual]

Returns the size of resulting node.

Returns

Size of resulting node.

Reimplemented in [TMClassAtom< T >](#), [IAMClassAtom< T >](#), [SClassAtom< T >](#), and [T2ClassAtom< T >](#).

Definition at line 98 of file ClassAtom.h.

6.3.3.12 template<class T, class C> virtual void ClassAtom< T, C >::setErrorMissing (C val) [inline, virtual]

Sets the value that means "error there is a missing value".

Parameters

<i>val</i>	Error missing value coding.
------------	-----------------------------

Definition at line 156 of file ClassAtom.h.

6.3.3.13 template<class T, class C> virtual void ClassAtom< T, C >::setMissingCode (T *val*) [inline, virtual]

Sets the value that means "missing value".

Parameters

<i>val</i>	Missing value coding.
------------	-----------------------

Definition at line 142 of file ClassAtom.h.

6.3.3.14 template<class T, class C> virtual uli_t ClassAtom< T, C >::size () const [inline, virtual]

Returns the size of classifier.

Returns

Size of classifier.

Reimplemented in [TMClassAtom< T >](#).

Definition at line 89 of file ClassAtom.h.

6.3.4 Member Data Documentation

6.3.4.1 template<class T, class C> C ClassAtom< T, C >::errorMissing

Error missing code.

Definition at line 178 of file ClassAtom.h.

6.3.4.2 template<class T, class C> T ClassAtom< T, C >::missingCode

Missing code.

Definition at line 175 of file ClassAtom.h.

The documentation for this class was generated from the following file:

- [src/library/ClassAtom.h](#)

6.4 CmpldTree< T > Class Template Reference

```
#include <CmpldTree.h>
```

Public Member Functions

- **CmpldTree (uli_t varSize, uli_t valueWidth, uli_t valueSize, uli_t branchSize)**
Construct that sets several tree parameters.
- virtual **~CmpldTree ()**
Destructor that does nothing.
- void **printXml (std::ostream &output)**
Prints the tree to output stream.
- std::ostream & **print (std::ostream &os, bool isXML=false) const**
Prints the tree to output stream.
- void **pushBackNode ()**
Appends a clean node to flattened structure.
- void **toLastVarID (uli_t id)**
Sets variable ID of last node.
- void **toLastVarID (uli_t i, uli_t id)**
Sets multidimensional variable of last node.
- void **toLastValues (uli_t i, T val)**
Sets value of last node.
- void **pushBackToLastValues (T val)**
Appends value to last value vector.
- void **pushBackToLastValues (uli_t i, T val)**
Appends value to last value vector i.
- void **toLastBranches (uli_t i, uli_t val)**
Sets a branch value to a specific value.
- void **toVarID (uli_t at, uli_t id)**
Sets ID of a variable.
- void **toValues (uli_t at, uli_t i, T val)**
Sets value of a node.
- void **toBranches (uli_t at, uli_t i, uli_t val)**
Sets branch of a node.
- **uli_t size ()**
Gets size of variable ID vector.
- **bool hasVarID (unsigned int id)**
Returns if a specific variable ID is within the tree.
- template<class TT >
void variableToXML (TT &val, std::string name, std::ostream &out)
Prints a variable to output stream.
- template<class TT >
void variableToXMLWithMax (TT &val, std::string name, std::ostream &out)
Prints a variable to output stream.

Public Attributes

- std::vector< std::vector< uli_t > > **varID**
Vector of multi dimensional variables IDs (prefix tree).
- std::vector< std::vector < std::vector< T > > > **values**
Vector of multi dimensional values (prefix tree).
- std::vector< std::vector< uli_t > > **branches**
Vector of multi dimensional branches (prefix tree).
- std::vector< T > **classes**
Vector of all classes.
- std::vector< uli_t > **indexes**
Vector of all indexes.
- std::vector< std::vector < double > > **extra**
Vector of multi doubles (e.g. for term node infos).
- **uli_t varSize**
Size of multi dimensional variables.
- **uli_t valueSize**
Size of multi dimensional values.
- **uli_t valueWidth**
Width of a value.
- **uli_t branchSize**
Number of branches per node.
- **uli_t termID**
ID of a terminal node.

6.4.1 Detailed Description

template<class T>class CmpldTree< T >

Definition at line 35 of file CmpldTree.h.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 template<class T> CmpldTree< T >::CmpldTree (**uli_t varSize, uli_t valueWidth, uli_t valueSize, uli_t branchSize**) [inline]

Construct that sets several tree parameters.

Parameters

varSize	Size of variable structure.
valueWidth	Width of value structure.
valueSize	Size of value structure.
branchSize	Size of branch structure.

Definition at line 46 of file CmpldTree.h.

6.4.2.2 template<class T> virtual CmpldTree< T >::~CmpldTree() [inline, virtual]

Destructor that does nothing.

Definition at line 58 of file CmpldTree.h.

6.4.3 Member Function Documentation

6.4.3.1 template<class T> bool CmpldTree< T >::hasVarID (unsigned int *id*) [inline]

Returns if a specific variable ID is within the tree.

Parameters

<i>id</i>	The ID of variable.
-----------	---------------------

Returns

If variable is within the tree.

Definition at line 253 of file CmpldTree.h.

6.4.3.2 template<class T> std::ostream& CmpldTree< T >::print (std::ostream & *os*, bool *isXML* = false) const [inline]

Prints the tree to output stream.

Parameters

<i>os</i>	Output stream.
<i>isXML</i>	Is it in XML format.

Returns

Output stream.

Definition at line 88 of file CmpldTree.h.

6.4.3.3 template<class T> void CmpldTree< T >::printXml (std::ostream & *output*) [inline]

Prints the tree to output stream.

Parameters

<i>output</i>	Output stream.
---------------	----------------

Definition at line 65 of file CmpldTree.h.

6.4.3.4 template<class T> void CmpldTree< T >::pushBackNode() [inline]

Appends a clean node to flattend structure.

Definition at line 141 of file CmpldTree.h.

6.4.3.5 template<class T> void CmpldTree< T >::pushBackToLastValues(T val) [inline]

Appends value to last value vector.

Parameters

<i>val</i>

Definition at line 181 of file CmpldTree.h.

6.4.3.6 template<class T> void CmpldTree< T >::pushBackToLastValues(uli_t i, T val) [inline]

Appends value to last value vector i.

Parameters

<i>i</i>	Index of value in vector.
<i>val</i>	

Definition at line 191 of file CmpldTree.h.

6.4.3.7 template<class T> uli_t CmpldTree< T >::size() [inline]

Gets size of variable ID vector.

Returns

Size of variable ID vector.

Definition at line 242 of file CmpldTree.h.

6.4.3.8 template<class T> void CmpldTree< T >::toBranches (uli_t at, uli_t i, uli_t val) [inline]

Sets branch of a node.

Parameters

<i>at</i>	Position in list of nodes.
<i>i</i>	Index in branch vector.
<i>val</i>	Value.

Definition at line 233 of file CmpldTree.h.

6.4.3.9 template<class T> void CmpldTree< T >::toLastBranches (uli_t i, uli_t val) [inline]

Sets a branch value to a specific value.

Parameters

<i>i</i>	Index in last branch.
<i>val</i>	Value.

Definition at line 201 of file CmpldTree.h.

6.4.3.10 template<class T> void CmpldTree< T >::toLastValues (uli_t i, T val) [inline]

Sets value of last node.

Parameters

<i>i</i>	Index of value.
<i>val</i>	Value.

Definition at line 172 of file CmpldTree.h.

6.4.3.11 template<class T> void CmpldTree< T >::toLastVarID (uli_t id) [inline]

Sets variable ID of last node.

Parameters

<i>id</i>	The ID.
-----------	---------

Definition at line 152 of file CmpldTree.h.

6.4.3.12 template<class T> void CmpldTree< T >::toLastVarID (uli_t *i*, uli_t *id*)
 [inline]

Sets multidimensional variable of last node.

Parameters

<i>i</i>	Dimension position.
<i>id</i>	The ID.

Definition at line 162 of file CmpldTree.h.

6.4.3.13 template<class T> void CmpldTree< T >::toValues (uli_t *at*, uli_t *i*, T *val*)
 [inline]

Sets value of a node.

Parameters

<i>at</i>	Position in list of nodes.
<i>i</i>	Index in value vector.
<i>val</i>	Value.

Definition at line 222 of file CmpldTree.h.

6.4.3.14 template<class T> void CmpldTree< T >::toVarID (uli_t *at*, uli_t *id*)
 [inline]

Sets ID of a variable.

Parameters

<i>at</i>	Position.
<i>id</i>	ID of variable.

Definition at line 211 of file CmpldTree.h.

6.4.3.15 template<class T> template<class TT> void CmpldTree< T >::variableToXML
 (TT & *val*, std::string *name*, std::ostream & *out*) [inline]

Prints a variable to output stream.

Parameters

<i>val</i>	Value.
<i>name</i>	Name of variable.
<i>out</i>	Output stream.

Definition at line 265 of file CmpldTree.h.

```
6.4.3.16 template<class T> template<class TT > void CmpldTree< T
>::variableToXMLWithMax ( TT & val, std::string name, std::ostream & out )
[inline]
```

Prints a variable to output stream.

The maximum will be printed as "MAX" string.

Parameters

<i>val</i>	Value.
<i>name</i>	Name of variable.
<i>out</i>	Output stream.

Definition at line 278 of file CmpldTree.h.

6.4.4 Member Data Documentation

```
6.4.4.1 template<class T> std::vector<std::vector<uli_t > > CmpldTree< T
>::branches
```

Vector of multi dimensional branches (prefix tree).

Definition at line 292 of file CmpldTree.h.

```
6.4.4.2 template<class T> uli_t CmpldTree< T >::branchSize
```

Number of branches per node.

Definition at line 313 of file CmpldTree.h.

```
6.4.4.3 template<class T> std::vector<T > CmpldTree< T >::classes
```

Vector of all classes.

Definition at line 295 of file CmpldTree.h.

```
6.4.4.4 template<class T> std::vector<std::vector<double> > CmpldTree< T >::extra
```

Vector of multi doubles (e.g. for term node infos).

Definition at line 301 of file CmpldTree.h.

```
6.4.4.5 template<class T> std::vector<uli_t > CmpldTree< T >::indexes
```

Vector of all indexes.

Definition at line 298 of file CmpldTree.h.

6.4.4.6 template<class T> uli_t CmpldTree< T >::termID

ID of a terminal node.

Definition at line 316 of file CmpldTree.h.

6.4.4.7 template<class T> std::vector<std::vector<std::vector<T > > > CmpldTree< T >::values

Vector of multi dimensional values (prefix tree).

Definition at line 289 of file CmpldTree.h.

6.4.4.8 template<class T> uli_t CmpldTree< T >::valueSize

Size of multi dimensional values.

Definition at line 307 of file CmpldTree.h.

6.4.4.9 template<class T> uli_t CmpldTree< T >::valueWidth

Width of a value.

Definition at line 310 of file CmpldTree.h.

6.4.4.10 template<class T> std::vector<std::vector<uli_t > > CmpldTree< T >::varID

Vector of multi dimensional variables IDs (prefix tree).

Definition at line 286 of file CmpldTree.h.

6.4.4.11 template<class T> uli_t CmpldTree< T >::varSize

Size of multi dimensional variables.

Definition at line 304 of file CmpldTree.h.

The documentation for this class was generated from the following file:

- [src/library/CmpldTree.h](#)

6.5 CmplFct Class Reference

Representation of a set of compiler functions.

```
#include <CmplFct.h>
```

Public Member Functions

- **CmplFct ()**
Constructor of CmplFct.
- virtual ~CmplFct ()
Destructor of CmplFct.

Static Public Member Functions

- template<class T >
static CmpldTree< T > * T2 (Tree< T, uli_t > &tree)
Compile a T2 tree (see T2ClassAtom class)
- template<class T >
static void makeT2 (INode< T > *node, CmpldTree< T > *cmpldTree)
Recursive wrapper function for compiling flat trees.
- template<class T >
static T classifyT2 (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T >
static unsigned int classifyT2nodeID (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T >
static void getCutoffsT2 (CmpldTree< T > &cmpldTree, uli_t var, std::vector< T > &cutoffs)
Get cutoffs of a specific variable in tree.
- template<class T >
static void addVarProxT2 (DataFrame< T > &data, Proximities< T > &prox, CmpldTree< T > *cmpldTree, std::vector< uli_t > *colMask)
Add sample proximities of current tree to proximity matrix.
- template<class T >
static CmpldTree< T > * S (Tree< T, uli_t > &tree)
Compile a S tree (see SClassAtom class)
- template<class T >
static void makeS (INode< T > *node, CmpldTree< T > *cmpldTree)
Recursive wrapper function for compiling flat trees.
- template<class T >
static T classifyS (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T >
static unsigned int classifySnodeID (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T >
static CmpldTree< T > * IAM2Way (Tree< T, uli_t > &tree)
Compile a interaction tree (see IAMClassAtom class)

- template<class T>
 static void **makeIAM2Way** (INode< T > *node, CmpldTree< T > *cmpldTree)
Recursive wrapper function for compiling flat trees.
- template<class T>
 static T **classifyIAM2Way** (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T>
 static unsigned int **classifyIAM2WaynodeID** (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.
- template<class T>
 static CmpldTree< T > * **Lotus** (Tree< T, uli_t > &tree)
Compile a Lotus tree (see [LotusTermClassAtom](#) class)
- template<class T>
 static void **makeLotus** (INode< T > *node, CmpldTree< T > *cmpldTree)
Recursive wrapper function for compiling flat trees.
- template<class T>
 static T **predictLotus** (T *sample, CmpldTree< T > *cmpldTree)
Classify a sample utilizing a compiled tree.

6.5.1 Detailed Description

Representation of a set of compiler functions.

This binder serves serveral functions to different tree types as follows:

- compiling tree that consits of pointers
- classify samples
- get cutoffs of the tree
- add proximities

Definition at line 60 of file CmplFct.h.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 CmplFct::CmplFct()

Constructor of [CmplFct](#).

6.5.2.2 virtual CmplFct::~CmplFct() [virtual]

Destructor of [CmplFct](#).

6.5.3 Member Function Documentation

6.5.3.1 `template<class T> static void CmplFct::addVarProxT2 (DataFrame< T > & data, Proximities< T > & prox, CmpldTree< T > * cmpldTree, std::vector< uli_t > * colMask) [inline, static]`

Add sample proximities of current tree to proximity matrix.

Parameters

<code>data</code>	Data set which contains sample data.
<code>prox</code>	Sample proximity matrix.
<code>cmpldTree</code>	Compiled tree.
<code>colMask</code>	Mask of pre selected variables.

Definition at line 241 of file CmplFct.h.

6.5.3.2 `template<class T> static T CmplFct::classifyIAM2Way (T * sample, CmpldTree< T > * cmpldTree) [inline, static]`

Classify a sample utilizing a compiled tree.

Parameters

<code>sample</code>	Sample that should be classified.
<code>cmpldTree</code>	Classifier that should be used.

Returns

Terminal node value.

Definition at line 533 of file CmplFct.h.

6.5.3.3 `template<class T> static unsigned int CmplFct::classifyIAM2WaynodeID (T * sample, CmpldTree< T > * cmpldTree) [inline, static]`

Classify a sample utilizing a compiled tree.

Parameters

<code>sample</code>	Sample that should be classified.
<code>cmpldTree</code>	Classifier that should be used.

Returns

[Node](#) ID of terminal node.

Definition at line 573 of file CmplFct.h.

6.5.3.4 template<class T > static T CmplFct::classifyS (T * *sample*, CmpldTree< T > * *cmpldTree*) [inline, static]

Classify a sample utilizing a compiled tree.

Parameters

<i>sample</i>	Sample that should be classified.
<i>cmpldTree</i>	Classifier that should be used.

Returns

Terminal node value.

Definition at line 386 of file CmplFct.h.

6.5.3.5 template<class T > static unsigned int CmplFct::classifySnodeID (T * *sample*, CmpldTree< T > * *cmpldTree*) [inline, static]

Classify a sample utilizing a compiled tree.

Parameters

<i>sample</i>	Sample that should be classified.
<i>cmpldTree</i>	Classifier that should be used.

Returns

[Node](#) ID of terminal node.

Definition at line 423 of file CmplFct.h.

6.5.3.6 template<class T > static T CmplFct::classifyT2 (T * *sample*, CmpldTree< T > * *cmpldTree*) [inline, static]

Classify a sample utilizing a compiled tree.

Parameters

<i>sample</i>	Sample that should be classified.
<i>cmpldTree</i>	Classifier that should be used.

Returns

Terminal node value.

Definition at line 138 of file CmplFct.h.

**6.5.3.7 template<class T > static unsigned int CmplFct::classifyT2nodeID (T * *sample*,
CmpldTree< T > * *cpldTree*) [inline, static]**

Classify a sample utilizing a compiled tree.

Parameters

<i>sample</i>	Sample that should be classified.
<i>cpldTree</i>	Classifier that should be used.

Returns

[Node](#) ID of terminal node.

Definition at line 169 of file CmplFct.h.

**6.5.3.8 template<class T > static void CmplFct::getCutoffsT2 (CmpldTree< T > &
cpldTree, uli_t *var*, std::vector< T > & *cutoffs*) [inline, static]**

Get cutoffs of a specific variable in tree.

Parameters

<i>cpldTree</i>	A compiled tree.
<i>var</i>	Index of variable.
<i>cutoffs</i>	Cutoffs of a variable in tree. (Return value).

Definition at line 200 of file CmplFct.h.

**6.5.3.9 template<class T > static CmpldTree< T > * CmplFct::IAM2Way (Tree< T,
uli_t > & *tree*) [inline, static]**

Compile a interaction tree (see [IAMClassAtom](#) class)

Parameters

<i>tree</i>	Pointer based tree.
-------------	---------------------

Returns

flat data structure which represents the tree

Definition at line 465 of file CmplFct.h.

**6.5.3.10 template<class T > static CmpldTree< T > * CmplFct::Lotus (Tree< T, uli_t >
& *tree*) [inline, static]**

Compile a Lotus tree (see [LotusTermClassAtom](#) class)

Parameters

<i>tree</i>	Pointer based tree.
-------------	---------------------

Returns

flat data structure which represents the tree

Definition at line 618 of file CmplFct.h.

6.5.3.11 template<class T > static void CmplFct::makeIAM2Way (INode< T > * *node*, CmpldTree< T > * *cmpldTree*) [inline, static]

Recursive wrapper function for compiling flat trees.

Parameters

<i>node</i>	Node which should be compiled.
<i>cmpldTree</i>	Container which contains resulting tree.

Definition at line 484 of file CmplFct.h.

6.5.3.12 template<class T > static void CmplFct::makeLotus (INode< T > * *node*, CmpldTree< T > * *cmpldTree*) [inline, static]

Recursive wrapper function for compiling flat trees.

Parameters

<i>node</i>	Node which should be compiled.
<i>cmpldTree</i>	Container which contains resulting tree.

Definition at line 636 of file CmplFct.h.

6.5.3.13 template<class T > static void CmplFct::makeS (INode< T > * *node*, CmpldTree< T > * *cmpldTree*) [inline, static]

Recursive wrapper function for compiling flat trees.

Parameters

<i>node</i>	Node which should be compiled.
<i>cmpldTree</i>	Container which contains resulting tree.

Definition at line 340 of file CmplFct.h.

**6.5.3.14 template<class T> static void CmplFct::makeT2 (INode< T > * *node*,
CmpldTree< T > * *cmpldTree*) [inline, static]**

Recursive wrapper function for compiling flat trees.

Parameters

<i>node</i>	Node which shoulb be compiled.
<i>cmpldTree</i>	Container which contains resulting tree.

Definition at line 98 of file CmplFct.h.

**6.5.3.15 template<class T> static T CmplFct::predictLotus (T * *sample*, CmpldTree<
T > * *cmpldTree*) [inline, static]**

Classify a sample utilizing a compiled tree.

Parameters

<i>sample</i>	Sample that should be classified.
<i>cmpldTree</i>	Classifier that should be used.

Returns

Terminal node value.

Definition at line 690 of file CmplFct.h.

**6.5.3.16 template<class T> static CmpldTree< T > * CmplFct::S (Tree< T, uli_t > &
tree) [inline, static]**

Compile a S tree (see [SClassAtom](#) class)

Parameters

<i>tree</i>	Pointer based tree.
-------------	---------------------

Returns

flat data structure which represents the tree

Definition at line 321 of file CmplFct.h.

**6.5.3.17 template<class T> static CmpldTree< T > * CmplFct::T2 (Tree< T, uli_t > &
tree) [inline, static]**

Compile a T2 tree (see [T2ClassAtom](#) class)

Parameters

<i>tree</i>	Pointer based tree.
-------------	---------------------

Returns

flat data structure which represents the tree

Definition at line 80 of file CmplFct.h.

The documentation for this class was generated from the following file:

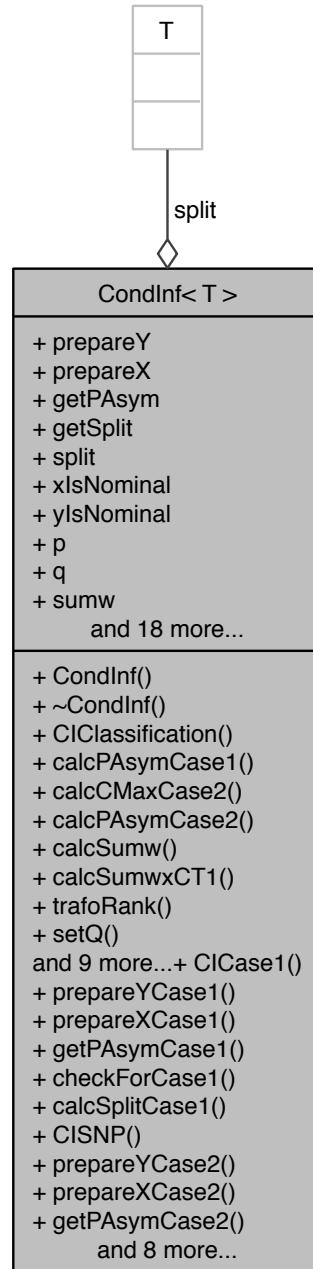
- [src/library/CmplFct.h](#)

6.6 CondInf< T > Class Template Reference

CI Functions for various cases as follows:

```
#include <CondInf.h>
```

Collaboration diagram for CondInf< T >:



Public Member Functions

- `CondInf()`
- `virtual ~CondInf()`
- `ClassAtom< T, uli_t > * CIClassification (FittingFctPar< T > &fitFctPar)`
- `void calcPAsymCase1 ()`
- `void calcCMaxCase2 ()`
- `void calcPAsymCase2 ()`
- `void calcSumw ()`
- `void calcSumwxCT1 ()`
- `void trafoRank (std::vector< T > &vec)`
- `void setQ ()`
- `void setP ()`
- `void calcdExp_y ()`
- `void calcdCov_y ()`
- `void calcdCov_T ()`
- `void calcdExp_T ()`
- `void calcT ()`
- `void calcCMax ()`
- `void calcPAsym ()`
- `double C_maxabsConditionalPvalue ()`
- `void C_kronecker (std::vector< double > &A, uli_t m, uli_t n, std::vector< double > &B, uli_t r, uli_t s, std::vector< double > &ans)`

Static Public Member Functions

- `static ClassAtom< T, uli_t > * CICase1 (FittingFctPar< T > &fitFctPar)`
- `static void prepareYCase1 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void prepareXCase1 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void getPAsymCase1 (CondInf< T > &ci)`
- `static void checkForCase1 (DataFrame< T > &data)`
- `static void calcSplitCase1 (CondInf< T > &ci)`
- `static ClassAtom< T, uli_t > * CISNP (FittingFctPar< T > &fitFctPar)`

Conditional Inference Trees for SNPs.

- `static void prepareYCase2 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void prepareXCase2 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void getPAsymCase2 (CondInf< T > &ci)`
- `static void checkForTwoSampleData (DataFrame< T > &data)`
- `static void calcSplitCase2 (CondInf< T > &ci)`
- `static ClassAtom< T, uli_t > * CICase3 (FittingFctPar< T > &fitFctPar)`

Conditional Inference Trees for.

- `static void prepareYCase3 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void prepareXCase3 (CondInf< T > &ci, DataFrame< T > &data, uli_t j)`
- `static void getPAsymCase3 (CondInf< T > &ci)`
- `static void checkForCase3 (DataFrame< T > &data)`
- `static void calcSplitCase3 (CondInf< T > &ci)`
- `static void checkForCI (DataFrame< T > &data)`

Public Attributes

- `void(* prepareY)(CondInf< T > &ci, DataFrame< T > &, uli_t row)`
Modify y vector (generic function)
- `void(* prepareX)(CondInf< T > &ci, DataFrame< T > &, uli_t row)`
Modify y vector (generic function)
- `void(* getPAsym)(CondInf< T > &ci)`
Calc. asymptotic p-value (generic function)
- `void(* getSplit)(CondInf< T > &ci)`
Get cutoff/split (generic function)
- `T split`
Data set is partitioned at this value.
- `bool xIsNominal`
Indicators / level of measurement of variables.
- `bool yIsNominal`
- `uli_t p`
Number of classes in x.
- `uli_t q`
Number of classes in y.
- `double sumw`
Sum of weights.
- `double cMax`
- `double pAsym`
P-value of test.
- `std::vector< T > x`
Vectors of x values.
- `std::vector< T > y`
Vector of y values.
- `std::vector< double > w`
Weights.
- `std::vector< double > t`
Measurement of association between y and x via multivariate linear statistic t.
- `std::vector< double > dExp_T`
Conditional expectation of T statistic.
- `std::vector< double > dCov_T`
Covariance matrix of T statistic.
- `std::vector< double > dExp_y`
Conditional expectation of the influence function.
- `std::vector< double > dCov_y`
Conditional covariance of the influence function.
- `std::vector< double > swx`
*Sum of the case weights * x.*
- `std::vector< double > CT1`

- **int maxpts**
Parameter of Fortran function mvtdst.
- **int inform**
Parameter of Fortran function mvtdst.
- **double tol**
Parameter of Fortran function mvtdst.
- **double abseps**
Parameter of Fortran function mvtdst.
- **double redeps**
Parameter of Fortran function mvtdst.
- **double error**
Parameter of Fortran function mvtdst.
- **gsl_rng * rng**
Random number generator.

6.6.1 Detailed Description

template<class T>class CondInf< T >

CI Functions for various cases as follows:

- y (numerical), x(numerical)
- y (nominal), x(numerical)
- SNP data: y (2-classes), x(numerical)

Definition at line 41 of file CondInf.h.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 template<class T> CondInf< T >::CondInf() [inline]

Definition at line 43 of file CondInf.h.

6.6.2.2 template<class T> virtual CondInf< T >::~CondInf() [inline,
virtual]

Definition at line 47 of file CondInf.h.

6.6.3 Member Function Documentation

6.6.3.1 `template<class T> void CondInf< T >::C_kronecker(std::vector< double > & A,
 uli_t m, uli_t n, std::vector< double > & B, uli_t r, uli_t s, std::vector< double >
 & ans) [inline]`

Definition at line 897 of file CondInf.h.

6.6.3.2 `template<class T> double CondInf< T >::C_maxabsConditionalPvalue() [inline]`

Definition at line 788 of file CondInf.h.

6.6.3.3 `template<class T> void CondInf< T >::calcCMax() [inline]`

Definition at line 765 of file CondInf.h.

6.6.3.4 `template<class T> void CondInf< T >::calcCMaxCase2() [inline]`

Definition at line 370 of file CondInf.h.

6.6.3.5 `template<class T> void CondInf< T >::calcdCov_T() [inline]`

Definition at line 684 of file CondInf.h.

6.6.3.6 `template<class T> void CondInf< T >::calcdCov_y() [inline]`

Definition at line 640 of file CondInf.h.

6.6.3.7 `template<class T> void CondInf< T >::calcdExp_T() [inline]`

Definition at line 733 of file CondInf.h.

6.6.3.8 `template<class T> void CondInf< T >::calcdExp_y() [inline]`

Definition at line 610 of file CondInf.h.

6.6.3.9 `template<class T> void CondInf< T >::calcPAsym() [inline]`

Definition at line 781 of file CondInf.h.

6.6.3.10 template<class T> void CondInf< T >::calcPAsymCase1() [inline]

Definition at line 267 of file CondInf.h.

6.6.3.11 template<class T> void CondInf< T >::calcPAsymCase2() [inline]

Definition at line 397 of file CondInf.h.

6.6.3.12 template<class T> static void CondInf< T >::calcSplitCase1 (CondInf< T > & ci) [inline, static]

Definition at line 223 of file CondInf.h.

6.6.3.13 template<class T> static void CondInf< T >::calcSplitCase2 (CondInf< T > & ci) [inline, static]

Definition at line 335 of file CondInf.h.

6.6.3.14 template<class T> static void CondInf< T >::calcSplitCase3 (CondInf< T > & ci) [inline, static]

Definition at line 456 of file CondInf.h.

6.6.3.15 template<class T> void CondInf< T >::calcSumw() [inline]

Definition at line 504 of file CondInf.h.

6.6.3.16 template<class T> void CondInf< T >::calcSumwxCT1() [inline]

Definition at line 512 of file CondInf.h.

6.6.3.17 template<class T> void CondInf< T >::calcT() [inline]

Definition at line 744 of file CondInf.h.

6.6.3.18 template<class T> static void CondInf< T >::checkForCase1 (DataFrame< T > & data) [inline, static]

Definition at line 219 of file CondInf.h.

6.6.3.19 template<class T> static void CondInf< T >::checkForCase3 (DataFrame< T > & *data*) [inline, static]

Definition at line 452 of file CondInf.h.

6.6.3.20 template<class T> static void CondInf< T >::checkForCI (DataFrame< T > & *data*) [inline, static]

Definition at line 919 of file CondInf.h.

6.6.3.21 template<class T> static void CondInf< T >::checkForTwoSampleData (DataFrame< T > & *data*) [inline, static]

Definition at line 328 of file CondInf.h.

6.6.3.22 template<class T> static ClassAtom<T, uli_t>* CondInf< T >::CICase1 (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 185 of file CondInf.h.

6.6.3.23 template<class T> static ClassAtom<T, uli_t>* CondInf< T >::CICase3 (FittingFctPar< T > & *fitFctPar*) [inline, static]

Conditional Inference Trees for.

Variables: y (numeric) -> h(y,...) = y x_j (numeric) -> g(x) = x

Global hypo.:

- Using only "standardized linear statistic": $c_{\max} = \max|(\bar{t})_k / (\{\bar{k}\})|$
- Bonferroni correction for mult. testing (unused)

Splitting:

- Find best split point in ordered subset A. So, find split in $O(n)$.

Definition at line 419 of file CondInf.h.

6.6.3.24 template<class T> ClassAtom<T, uli_t>* CondInf< T >::CIClassification (FittingFctPar< T > & *fitFctPar*) [inline]

Definition at line 58 of file CondInf.h.

6.6.3.25 template<class T> static ClassAtom<T, uli_t>* CondInf< T >::CISNP (FittingFctPar< T > & fitFctPar) [inline, static]

Conditional Inference Trees for SNPs.

Variables: $y \{0,1\}$ (nominal) $\rightarrow h(y, \dots) = e_2(y)$ $x_j \{0,1,2\}$ (numeric) $\rightarrow g(x) = x$

Global hypo.:

- Using only "standardized linear statistic": $c_{\max} = \max|t_k - k / (\sum k)|$
- Bonferroni correction for mult. testing (unused)

Splitting:

- Find best split point in ordered subset A. So, find split in $O(n)$.

Definition at line 300 of file CondInf.h.

6.6.3.26 template<class T> static void CondInf< T >::getPAsymCase1 (CondInf< T > & ci) [inline, static]

Definition at line 214 of file CondInf.h.

6.6.3.27 template<class T> static void CondInf< T >::getPAsymCase2 (CondInf< T > & ci) [inline, static]

Definition at line 323 of file CondInf.h.

6.6.3.28 template<class T> static void CondInf< T >::getPAsymCase3 (CondInf< T > & ci) [inline, static]

Definition at line 447 of file CondInf.h.

6.6.3.29 template<class T> static void CondInf< T >::prepareXCase1 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]

Definition at line 203 of file CondInf.h.

6.6.3.30 template<class T> static void CondInf< T >::prepareXCase2 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]

Definition at line 316 of file CondInf.h.

6.6.3.31 `template<class T> static void CondInf< T >::prepareXCase3 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]`

Definition at line 436 of file CondInf.h.

6.6.3.32 `template<class T> static void CondInf< T >::prepareYCase1 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]`

Definition at line 194 of file CondInf.h.

6.6.3.33 `template<class T> static void CondInf< T >::prepareYCase2 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]`

Definition at line 309 of file CondInf.h.

6.6.3.34 `template<class T> static void CondInf< T >::prepareYCase3 (CondInf< T > & ci, DataFrame< T > & data, uli_t j) [inline, static]`

Definition at line 428 of file CondInf.h.

6.6.3.35 `template<class T> void CondInf< T >::setP() [inline]`

Definition at line 594 of file CondInf.h.

6.6.3.36 `template<class T> void CondInf< T >::setQ() [inline]`

Definition at line 577 of file CondInf.h.

6.6.3.37 `template<class T> void CondInf< T >::trafoRank (std::vector< T > & vec) [inline]`

Definition at line 551 of file CondInf.h.

6.6.4 Member Data Documentation

6.6.4.1 `template<class T> double CondInf< T >::abseps`

Parameter of Fortran function mvtdst.

Definition at line 1001 of file CondInf.h.

6.6.4.2 template<class T> double CondInf< T >::cMax

Value of test statistics c. Here, it is the maximum of the absolute values of the standardized linear statistic.j

Definition at line 956 of file CondInf.h.

6.6.4.3 template<class T> std::vector<double> CondInf< T >::CT1

Helping structure.

Definition at line 989 of file CondInf.h.

6.6.4.4 template<class T> std::vector<double> CondInf< T >::dCov_T

Covariance matrix of T statistic.

Definition at line 977 of file CondInf.h.

6.6.4.5 template<class T> std::vector<double> CondInf< T >::dCov_y

Conditional covariance of the influence function.

Definition at line 983 of file CondInf.h.

6.6.4.6 template<class T> std::vector<double> CondInf< T >::dExp_T

Conditional expectation of T statistic.

Definition at line 974 of file CondInf.h.

6.6.4.7 template<class T> std::vector<double> CondInf< T >::dExp_y

Conditional expectation of the influence function.

Definition at line 980 of file CondInf.h.

6.6.4.8 template<class T> double CondInf< T >::error

Parameter of Fortran function mvtdst.

Definition at line 1007 of file CondInf.h.

6.6.4.9 template<class T> void(* CondInf< T >::getPAsym)(CondInf< T > &ci)

Calc. asymptotic p-value (generic function)

Definition at line 931 of file CondInf.h.

6.6.4.10 template<class T> void(* CondInf< T >::getSplit)(CondInf< T > &ci)

Get cutoff/split (generic function)

Definition at line 934 of file CondInf.h.

6.6.4.11 template<class T> int CondInf< T >::inform

Parameter of Fortran function mvtdst.

Definition at line 995 of file CondInf.h.

6.6.4.12 template<class T> int CondInf< T >::maxpts

Parameter of Fortran function mvtdst.

Definition at line 992 of file CondInf.h.

6.6.4.13 template<class T> uli_t CondInf< T >::p

Number of classes in x.

Definition at line 944 of file CondInf.h.

6.6.4.14 template<class T> double CondInf< T >::pAsym

P-value of test.

Definition at line 959 of file CondInf.h.

**6.6.4.15 template<class T> void(* CondInf< T >::prepareX)(CondInf< T > &ci,
 DataFrame< T > &, uli_t row)**

Modify y vector (generic function)

Definition at line 928 of file CondInf.h.

**6.6.4.16 template<class T> void(* CondInf< T >::prepareY)(CondInf< T > &ci,
 DataFrame< T > &, uli_t row)**

Modify y vector (generic function)

Definition at line 925 of file CondInf.h.

6.6.4.17 template<class T> uli_t CondInf< T >::q

Number of classes in y.

Definition at line 947 of file CondInf.h.

6.6.4.18 template<class T> double CondInf< T >::releps

Parameter of Fortran function mvtdst.

Definition at line 1004 of file CondInf.h.

6.6.4.19 template<class T> gsl_rng* CondInf< T >::rng

Random number generator.

Definition at line 1010 of file CondInf.h.

6.6.4.20 template<class T> T CondInf< T >::split

Data set is partitioned at this value.

Definition at line 938 of file CondInf.h.

6.6.4.21 template<class T> double CondInf< T >::sumw

Sum of weights.

Definition at line 950 of file CondInf.h.

6.6.4.22 template<class T> std::vector<double> CondInf< T >::swx

Sum of the case weights * x.

Definition at line 986 of file CondInf.h.

6.6.4.23 template<class T> std::vector<double> CondInf< T >::t

Measurement of association between y and x via multivariate linear statistic t.

Definition at line 971 of file CondInf.h.

6.6.4.24 template<class T> double CondInf< T >::tol

Parameter of Fortran function mvtdst.

Definition at line 998 of file CondInf.h.

6.6.4.25 template<class T> std::vector<double> CondInf< T >::w

Weights.

Definition at line 968 of file CondInf.h.

6.6.4.26 template<class T> std::vector<T> CondInf< T >::x

Vectors of x values.

Definition at line 962 of file CondInf.h.

6.6.4.27 template<class T> bool CondInf< T >::xIsNominal

Indicators / level of measurement of variables.

Definition at line 941 of file CondInf.h.

6.6.4.28 template<class T> std::vector<T> CondInf< T >::y

Vector of y values.

Definition at line 965 of file CondInf.h.

6.6.4.29 template<class T> bool CondInf< T >::yIsNominal

Definition at line 941 of file CondInf.h.

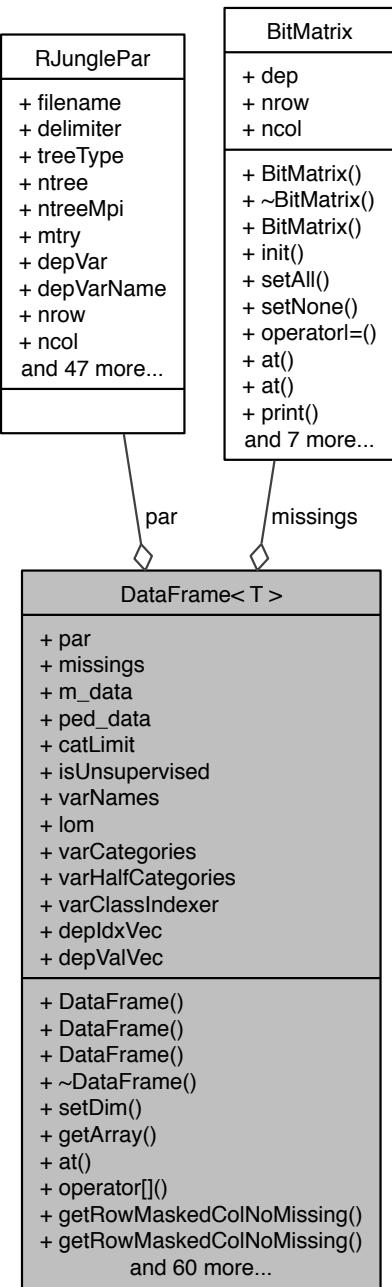
The documentation for this class was generated from the following file:

- src/library/CondInf.h

6.7 DataFrame< T > Class Template Reference

```
#include <DataFrame.h>
```

Collaboration diagram for DataFrame< T >:



Public Member Functions

- `DataFrame (RJunglePar &par)`
- `DataFrame (uli_t size)`
- `DataFrame (const DataFrame &dataFrame)`
- `virtual ~DataFrame ()`
- `void setDim (uli_t nrow, uli_t ncol)`
- `T getArray (uli_t i, uli_t j)`
- `T at (uli_t i, uli_t j) const`
- `const T *const operator[] (uli_t i)`
- `void getRowMaskedColNoMissing (uli_t j, std::vector< uli_t > &maskVec, std::vector< T > &outVec, std::vector< uli_t > &idxVec) const`
- `void getRowMaskedColNoMissing (uli_t j, std::vector< uli_t > &maskVec, std::vector< T > &outVec) const`
- `void getRowMaskedColNoMissing2 (uli_t j, std::vector< uli_t > &maskVec, std::vector< T > &outVec, std::vector< uli_t > &idxVec, std::vector< uli_t > &idxMissVec) const`
- `void getRowMaskedColNoMissing2AndPair (uli_t j, std::vector< uli_t > &maskVec, std::vector< std::pair< T, uli_t > > &result, std::vector< uli_t > &idxMissVec) const`
- `void getRowMaskedColNoMissing2AndPair (uli_t j, std::vector< uli_t > &maskVec, std::vector< std::pair< T, uli_t > > &result) const`
- `void getRowMaskedColNoMissing3 (size_t j, std::vector< size_t > &maskVec, std::vector< double > &outVec) const`
- `void getRowMaskedClassesInCol (uli_t j, std::vector< uli_t > &maskVec, std::vector< T > &outVec) const`
- `void getRowMaskedCol (uli_t j, std::vector< uli_t > &maskVec, std::vector< T > &outVec) const`
- `void getRow (uli_t j, std::vector< T > &outVec) const`
- `void getRow (uli_t j, T *&outVec) const`
- `void getRowPieceWise (uli_t j, T *&outVec) const`
- `void getCol (uli_t j, std::vector< T > &outVec) const`
- `void getColMiss (uli_t j, std::vector< T > &outVec) const`
- `bool anyMissingsInCol (uli_t j) const`
- `void getColOrig (uli_t j, T *inVec) const`
- `void setColSynth (uli_t j, T *outVec)`
- `void getColNoMissings (uli_t j, std::vector< T > &outVec) const`
- `void getColNoMissingsExt (uli_t j, std::vector< T > &outVec) const`
- `RJunglePar getDataFromCSV ()`
- `void getColMaskedRow (uli_t i, std::vector< uli_t > &maskVec, T *outVec) const`
- `void setArray (uli_t i, uli_t j, T val)`
- `void set (uli_t i, uli_t j, T val)`
- `void rm (uli_t i, uli_t j)`
- `void setAll (T val)`
- `uli_t getnrow ()`
- `uli_t getncol ()`
- `std::ostream & print (std::ostream &os) const`

- void `printCSV` (std::ostream &os, std::vector< `uli_t` > *colMaskVec=NULL, bool printDepVar=true) const
- const std::vector< std::string > & `getVarNames` () const
- void `setVarNames` (std::vector< std::string > &names)
- void `makeDepVecs` ()
- `uli_t` `find` (T val, std::vector< T > &vec)
- void `storeCategories` ()
- void `storeHalfCategories` ()
- void `printSummary` ()
- void `setMissingCode` (int val)
- T `getMissingCode` ()
- void `setDepVar` (`uli_t` idx)
- `uli_t` `getDepVar` ()
- void `setDepVarName` (std::string depVarName)
- void `setCol` (`uli_t` col, T *vec)
- void `setColMaskedRow` (`uli_t` i, std::vector< `uli_t` > &maskVec, T *inVec)
- void `getCol` (`uli_t` col, T *vec)
- std::vector< `uli_t` > * `getIndexOfVarNames` (std::vector< std::string > selNames)
- `uli_t` `getIndexOfVarName` (std::string selName)
- void `getMissings` ()
- void `div` (T val)
- void `divDiag` (T val)
- void `divNoDiag` (T val)
- void `mult` (T val)
- void `div` (DataFrame< T > &dataFrame)
- void `pow2` ()
- void `squareroot` ()
- void `nanToMinusFive` ()
- void `minusXPow2` (DataFrame< T > &dataFrame)
- void `minus` (DataFrame< T > &dataFrame)
- void `add` (DataFrame< T > &dataFrame)
- void `add` (unsigned int i, unsigned int j, T val)
- void `createUnsupervisedData` ()
- void `checkVarNames` (char *token, unsigned int i)
- virtual void `initMatrix` ()
- void `pushBackLom` (char *token)
- bool `missingsNotAllowed` () const

Public Attributes

- `RJunglePar` par
- `BitMatrix` missings
- T ** `m_data`
- std::string ** `ped_data`
- `uli_t` catLimit

- `bool isUnsupervised`
- `std::vector< std::string > varNames`
- `std::vector< char > lom`
- `std::vector< std::vector< uli_t > > varCategories`
- `std::vector< std::vector< uli_t > > varHalfCategories`
- `std::vector< std::vector< T > > varClassIndexer`
- `std::vector< uli_t > depIdxVec`
- `std::vector< T > depValVec`

6.7.1 Detailed Description

`template<class T>class DataFrame< T >`

Definition at line 37 of file DataFrame.h.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 `template<class T> DataFrame< T >::DataFrame (RJunglePar & par) [inline]`

Definition at line 39 of file DataFrame.h.

6.7.2.2 `template<class T> DataFrame< T >::DataFrame (uli_t size) [inline]`

Definition at line 46 of file DataFrame.h.

6.7.2.3 `template<class T> DataFrame< T >::DataFrame (const DataFrame< T > & dataFrame) [inline]`

Definition at line 54 of file DataFrame.h.

6.7.2.4 `template<class T> virtual DataFrame< T >::~DataFrame () [inline, virtual]`

Definition at line 77 of file DataFrame.h.

6.7.3 Member Function Documentation

6.7.3.1 `template<class T> void DataFrame< T >::add (DataFrame< T > & dataFrame) [inline]`

Definition at line 1093 of file DataFrame.h.

6.7.3.2 `template<class T> void DataFrame< T >::add (unsigned int i, unsigned int j, T val) [inline]`

Definition at line 1103 of file DataFrame.h.

6.7.3.3 `template<class T> bool DataFrame< T >::anyMissingInCol (uli_t j) const [inline]`

Definition at line 335 of file DataFrame.h.

6.7.3.4 `template<class T> T DataFrame< T >::at (uli_t i, uli_t j) const [inline]`

Definition at line 112 of file DataFrame.h.

6.7.3.5 `template<class T> void DataFrame< T >::checkVarNames (char * token, unsigned int i) [inline]`

Definition at line 1146 of file DataFrame.h.

6.7.3.6 `template<class T> void DataFrame< T >::createUnsupervisedData () [inline]`

Definition at line 1107 of file DataFrame.h.

6.7.3.7 `template<class T> void DataFrame< T >::div (T val) [inline]`

Definition at line 992 of file DataFrame.h.

6.7.3.8 `template<class T> void DataFrame< T >::div (DataFrame< T > & dataFrame) [inline]`

Definition at line 1031 of file DataFrame.h.

6.7.3.9 `template<class T> void DataFrame< T >::divDiag (T val) [inline]`

Definition at line 1002 of file DataFrame.h.

6.7.3.10 `template<class T> void DataFrame< T >::divNoDiag (T val) [inline]`

Definition at line 1010 of file DataFrame.h.

6.7.3.11 `template<class T> uli_t DataFrame<T>::find (T val, std::vector<T> & vec) [inline]`

Definition at line 824 of file DataFrame.h.

6.7.3.12 `template<class T> T DataFrame<T>::getArray (uli_t i, uli_t j) [inline]`

Definition at line 105 of file DataFrame.h.

6.7.3.13 `template<class T> void DataFrame<T>::getCol (uli_t j, std::vector<T> & outVec) const [inline]`

Definition at line 308 of file DataFrame.h.

6.7.3.14 `template<class T> void DataFrame<T>::getCol (uli_t col, T * vec) [inline]`

Definition at line 935 of file DataFrame.h.

6.7.3.15 `template<class T> void DataFrame<T>::getColMaskedRow (uli_t i, std::vector<uli_t> & maskVec, T * outVec) const [inline]`

Definition at line 566 of file DataFrame.h.

6.7.3.16 `template<class T> void DataFrame<T>::getColMiss (uli_t j, std::vector<T> & outVec) const [inline]`

Definition at line 318 of file DataFrame.h.

6.7.3.17 `template<class T> void DataFrame<T>::getColNoMissings (uli_t j, std::vector<T> & outVec) const [inline]`

Definition at line 356 of file DataFrame.h.

6.7.3.18 `template<class T> void DataFrame<T>::getColNoMissingsExt (uli_t j, std::vector<T> & outVec) const [inline]`

Definition at line 365 of file DataFrame.h.

6.7.3.19 `template<class T> void DataFrame<T>::getColOrig (uli_t j, T * inVec) const [inline]`

Definition at line 346 of file DataFrame.h.

6.7.3.20 `template<class T> RJunglePar DataFrame< T >::getDataFromCSV() [inline]`

Definition at line 379 of file DataFrame.h.

6.7.3.21 `template<class T> uli_t DataFrame< T >::getDepVar() [inline]`

Definition at line 910 of file DataFrame.h.

6.7.3.22 `template<class T> uli_t DataFrame< T >::getIndexOfVarName(std::string selName) [inline]`

Definition at line 959 of file DataFrame.h.

6.7.3.23 `template<class T> std::vector<uli_t>* DataFrame< T >::getIndexOfVarNames(std::vector< std::string > selNames) [inline]`

Definition at line 942 of file DataFrame.h.

6.7.3.24 `template<class T> T DataFrame< T >::getMissingCode() [inline]`

Definition at line 907 of file DataFrame.h.

6.7.3.25 `template<class T> void DataFrame< T >::getMissings() [inline]`

Definition at line 968 of file DataFrame.h.

6.7.3.26 `template<class T> uli_t DataFrame< T >::getncol() [inline]`

Definition at line 610 of file DataFrame.h.

6.7.3.27 `template<class T> uli_t DataFrame< T >::getnrow() [inline]`

Definition at line 609 of file DataFrame.h.

6.7.3.28 `template<class T> void DataFrame< T >::getRow(uli_t j, std::vector< T > & outVec) const [inline]`

Definition at line 287 of file DataFrame.h.

```
6.7.3.29 template<class T> void DataFrame< T >::getRow ( uli_t j, T *& outVec ) const  
[inline]
```

Definition at line 297 of file DataFrame.h.

```
6.7.3.30 template<class T> void DataFrame< T >::getRowMaskedClassesInCol  
( uli_t j, std::vector< uli_t > & maskVec, std::vector< T > & outVec ) const  
[inline]
```

Definition at line 198 of file DataFrame.h.

```
6.7.3.31 template<class T> void DataFrame< T >::getRowMaskedCol ( uli_t j,  
std::vector< uli_t > & maskVec, std::vector< T > & outVec ) const [inline]
```

Definition at line 276 of file DataFrame.h.

```
6.7.3.32 template<class T> void DataFrame< T >::getRowMaskedColNoMissing (   
uli_t j, std::vector< uli_t > & maskVec, std::vector< T > & outVec, std::vector<  
uli_t > & idxVec ) const [inline]
```

Definition at line 123 of file DataFrame.h.

```
6.7.3.33 template<class T> void DataFrame< T >::getRowMaskedColNoMissing  
( uli_t j, std::vector< uli_t > & maskVec, std::vector< T > & outVec ) const  
[inline]
```

Definition at line 135 of file DataFrame.h.

```
6.7.3.34 template<class T> void DataFrame< T >::getRowMaskedColNoMissing2 (   
uli_t j, std::vector< uli_t > & maskVec, std::vector< T > & outVec, std::vector<  
uli_t > & idxVec, std::vector< uli_t > & idxMissVec ) const [inline]
```

Definition at line 145 of file DataFrame.h.

```
6.7.3.35 template<class T> void DataFrame< T >::getRowMaskedColNoMissing2-  
AndPair ( uli_t j, std::vector< uli_t > & maskVec, std::vector< std::pair< T, uli_t  
> > & result, std::vector< uli_t > & idxMissVec ) const [inline]
```

Definition at line 161 of file DataFrame.h.

```
6.7.3.36 template<class T> void DataFrame< T >::getRowMaskedColNoMissing2-
AndPair ( uli_t j, std::vector< uli_t > & maskVec, std::vector< std::pair< T, uli_t
> > & result ) const [inline]
```

Definition at line 175 of file DataFrame.h.

```
6.7.3.37 template<class T> void DataFrame< T >::getRowMaskedColNoMissing3 (
size_t j, std::vector< size_t > & maskVec, std::vector< double > & outVec ) const
[inline]
```

Definition at line 185 of file DataFrame.h.

```
6.7.3.38 template<class T> void DataFrame< T >::getRowPieceWise ( uli_t j, T *&
outVec ) const [inline]
```

Definition at line 302 of file DataFrame.h.

```
6.7.3.39 template<class T> const std::vector<std::string>& DataFrame< T
>::getVarNames ( ) const [inline]
```

Definition at line 805 of file DataFrame.h.

```
6.7.3.40 template<class T> virtual void DataFrame< T >::initMatrix ( ) [inline,
virtual]
```

Definition at line 1173 of file DataFrame.h.

```
6.7.3.41 template<class T> void DataFrame< T >::makeDepVecs ( ) [inline]
```

Definition at line 813 of file DataFrame.h.

```
6.7.3.42 template<class T> void DataFrame< T >::minus ( DataFrame< T > &
dataFrame ) [inline]
```

Definition at line 1083 of file DataFrame.h.

```
6.7.3.43 template<class T> void DataFrame< T >::minusXPow2 ( DataFrame< T > &
dataFrame ) [inline]
```

Definition at line 1073 of file DataFrame.h.

6.7.3.44 **template<class T> bool DataFrame< T >::missingsNotAllowed() const [inline]**

Definition at line 1230 of file DataFrame.h.

6.7.3.45 **template<class T> void DataFrame< T >::mult(T val) [inline]**

Definition at line 1021 of file DataFrame.h.

6.7.3.46 **template<class T> void DataFrame< T >::nanToMinusFive() [inline]**

Definition at line 1061 of file DataFrame.h.

6.7.3.47 **template<class T> const T* const DataFrame< T >::operator[](uli_t i) [inline]**

Definition at line 118 of file DataFrame.h.

6.7.3.48 **template<class T> void DataFrame< T >::pow2() [inline]**

Definition at line 1041 of file DataFrame.h.

6.7.3.49 **template<class T> std::ostream& DataFrame< T >::print(std::ostream & os) const [inline]**

Definition at line 612 of file DataFrame.h.

6.7.3.50 **template<class T> void DataFrame< T >::printCSV(std::ostream & os, std::vector< uli_t > * colMaskVec = NULL, bool printDepVar = true) const [inline]**

Definition at line 676 of file DataFrame.h.

6.7.3.51 **template<class T> void DataFrame< T >::printSummary() [inline]**

Definition at line 887 of file DataFrame.h.

6.7.3.52 **template<class T> void DataFrame< T >::pushBackLom(char * token) [inline]**

Definition at line 1220 of file DataFrame.h.

6.7.3.53 template<class T> void DataFrame< T >::rm (uli_t *i*, uli_t *j*) [inline]

Definition at line 589 of file DataFrame.h.

6.7.3.54 template<class T> void DataFrame< T >::set (uli_t *i*, uli_t *j*, T *val*)
[inline]

Definition at line 584 of file DataFrame.h.

6.7.3.55 template<class T> void DataFrame< T >::setAll (T *val*) [inline]

Definition at line 594 of file DataFrame.h.

6.7.3.56 template<class T> void DataFrame< T >::setArray (uli_t *i*, uli_t *j*, T *val*)
[inline]

Definition at line 573 of file DataFrame.h.

6.7.3.57 template<class T> void DataFrame< T >::setCol (uli_t *col*, T * *vec*)
[inline]

Definition at line 920 of file DataFrame.h.

6.7.3.58 template<class T> void DataFrame< T >::setColMaskedRow (uli_t *i*,
std::vector<uli_t> & *maskVec*, T * *inVec*) [inline]

Definition at line 927 of file DataFrame.h.

6.7.3.59 template<class T> void DataFrame< T >::setColSynth (uli_t *j*, T * *outVec*)
[inline]

Definition at line 351 of file DataFrame.h.

6.7.3.60 template<class T> void DataFrame< T >::setDepVar (uli_t *idx*)
[inline]

Definition at line 909 of file DataFrame.h.

6.7.3.61 template<class T> void DataFrame< T >::setDepVarName (std::string
depVarName) [inline]

Definition at line 912 of file DataFrame.h.

6.7.3.62 `template<class T> void DataFrame< T >::setDim (uli_t nrow, uli_t ncol) [inline]`

Definition at line 100 of file DataFrame.h.

6.7.3.63 `template<class T> void DataFrame< T >::setMissingCode (int val) [inline]`

Definition at line 906 of file DataFrame.h.

6.7.3.64 `template<class T> void DataFrame< T >::setVarNames (std::vector< std::string > & names) [inline]`

Definition at line 809 of file DataFrame.h.

6.7.3.65 `template<class T> void DataFrame< T >::squareroot () [inline]`

Definition at line 1051 of file DataFrame.h.

6.7.3.66 `template<class T> void DataFrame< T >::storeCategories () [inline]`

Definition at line 828 of file DataFrame.h.

6.7.3.67 `template<class T> void DataFrame< T >::storeHalfCategories () [inline]`

Definition at line 859 of file DataFrame.h.

6.7.4 Member Data Documentation

6.7.4.1 `template<class T> uli_t DataFrame< T >::catLimit`

Definition at line 1245 of file DataFrame.h.

6.7.4.2 `template<class T> std::vector<uli_t> DataFrame< T >::depIdxVec`

Definition at line 1252 of file DataFrame.h.

6.7.4.3 `template<class T> std::vector<T> DataFrame< T >::depValVec`

Definition at line 1253 of file DataFrame.h.

6.7.4.4 `template<class T> bool DataFrame< T >::isUnsupervised`

Definition at line 1246 of file DataFrame.h.

6.7.4.5 `template<class T> std::vector<char> DataFrame< T >::lom`

Definition at line 1249 of file DataFrame.h.

6.7.4.6 `template<class T> T** DataFrame< T >::m_data`

Definition at line 1242 of file DataFrame.h.

6.7.4.7 `template<class T> BitMatrix DataFrame< T >::missings`

Definition at line 1241 of file DataFrame.h.

6.7.4.8 `template<class T> RJunglePar DataFrame< T >::par`

Definition at line 1239 of file DataFrame.h.

6.7.4.9 `template<class T> std::string** DataFrame< T >::ped_data`

Definition at line 1243 of file DataFrame.h.

6.7.4.10 `template<class T> std::vector<std::vector<uli_t> > DataFrame< T >::varCategories`

Definition at line 1250 of file DataFrame.h.

6.7.4.11 `template<class T> std::vector<std::vector<T> > DataFrame< T >::varClassIndexer`

Definition at line 1251 of file DataFrame.h.

6.7.4.12 `template<class T> std::vector<std::vector<uli_t> > DataFrame< T >::varHalfCategories`

Definition at line 1250 of file DataFrame.h.

6.7.4.13 template<class T> std::vector<std::string> DataFrame< T >::varNames

Definition at line 1248 of file DataFrame.h.

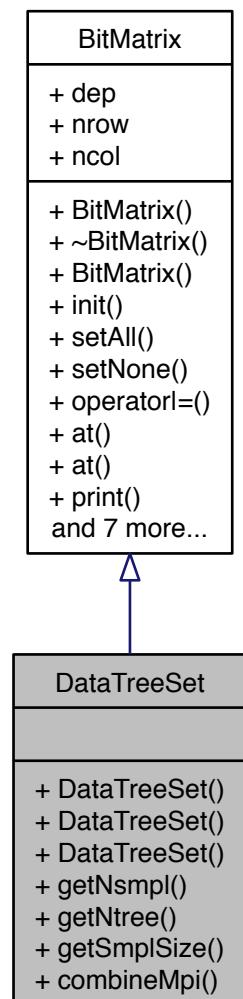
The documentation for this class was generated from the following file:

- src/library/[DataFrame.h](#)

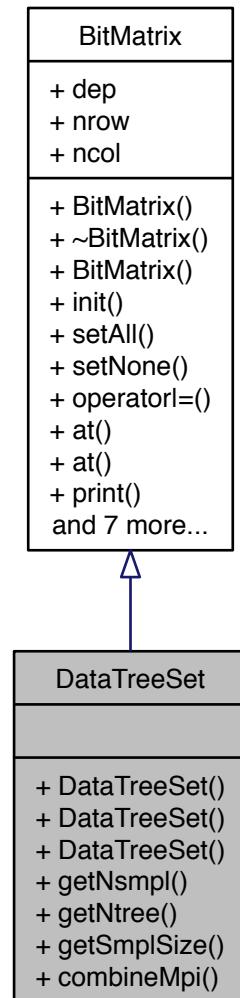
6.8 DataTreeSet Class Reference

```
#include <DataTreeSet.h>
```

Inheritance diagram for DataTreeSet:



Collaboration diagram for DataTreeSet:



Public Member Functions

- [DataTreeSet \(\)](#)
- [DataTreeSet \(uli_t nrow, uli_t ncol\)](#)
- [DataTreeSet \(const DataTreeSet &set\)](#)

Copy constructor.

- `uli_t getNsmp()`
- `uli_t getNtree()`
- `uli_t getSmplSize(unsigned int j)`
- `virtual void combineMPI(RJunglePar &par)`

6.8.1 Detailed Description

Definition at line 29 of file DataTreeSet.h.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 DataTreeSet::DataTreeSet() [inline]

Definition at line 31 of file DataTreeSet.h.

6.8.2.2 DataTreeSet::DataTreeSet(`uli_t nrow, uli_t ncol`) [inline]

Definition at line 33 of file DataTreeSet.h.

6.8.2.3 DataTreeSet::DataTreeSet(`const DataTreeSet & set`) [inline]

Copy constructor.

Parameters

<code>dataFrame</code>

Definition at line 39 of file DataTreeSet.h.

6.8.3 Member Function Documentation

6.8.3.1 virtual void DataTreeSet::combineMPI(`RJunglePar & par`) [inline, virtual]

Definition at line 49 of file DataTreeSet.h.

6.8.3.2 `uli_t DataTreeSet::getNsmp()` [inline]

Definition at line 45 of file DataTreeSet.h.

6.8.3.3 `uli_t DataTreeSet::getNtree()` [inline]

Definition at line 46 of file DataTreeSet.h.

6.8.3.4 `uli_t DataTreeSet::getSmpISize (unsigned int j) [inline]`

Definition at line 47 of file DataTreeSet.h.

The documentation for this class was generated from the following file:

- [src/library/DataTreeSet.h](#)

6.9 Exception Class Reference

```
#include <Exception.h>
```

Public Member Functions

- [Exception \(std::string str\)](#)
- [Exception \(int status, std::string str\)](#)
- [virtual ~Exception \(\)](#)
- [virtual const char * what \(\) const throw \(\)](#)

Public Attributes

- [std::string str](#)
- [int status](#)

6.9.1 Detailed Description

Definition at line 16 of file Exception.h.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 **Exception::Exception (std::string str)**

Definition at line 10 of file Exception.cpp.

6.9.2.2 `Exception::Exception (int status, std::string str) [inline]`

Definition at line 21 of file Exception.h.

6.9.2.3 `virtual Exception::~Exception () [inline, virtual]`

Definition at line 23 of file Exception.h.

6.9.3 Member Function Documentation

6.9.3.1 `const char * Exception::what() const throw() [virtual]`

Definition at line 14 of file Exception.cpp.

6.9.4 Member Data Documentation

6.9.4.1 `int Exception::status`

Definition at line 28 of file Exception.h.

6.9.4.2 `std::string Exception::str`

Definition at line 27 of file Exception.h.

The documentation for this class was generated from the following files:

- [src/library/Exception.h](#)
- [src/library/Exception.cpp](#)

6.10 FittingFct Class Reference

```
#include <FittingFct.h>
```

Public Member Functions

- [FittingFct \(\)](#)
- [virtual ~FittingFct \(\)](#)

Static Public Member Functions

- `template<class T>`
 `static ClassAtom< T, uli_t > * CARTreg (FittingFctPar< T > &fFP)`
- `template<class T>`
 `static ClassAtom< T, uli_t > * CARTregTwoing (FittingFctPar< T > &fitFctPar)`
- `template<class T>`
 `static ClassAtom< T, uli_t > * CARTtwoing (FittingFctPar< T > &fitFctPar)`
- `template<class T>`
 `static ClassAtom< T, uli_t > * CARTtwoWayIntAdd (FittingFctPar< T > &fitFctPar)`
- `template<class T>`
 `static void checkUsabilitySse (DataFrame< T > &data)`

- template<class T >
static ClassAtom< T, uli_t > * CARTgini (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * CARTginiNoDenominator (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * CARTginiSrtOld (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * CARTginiSrt (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * CARTginisse (FittingFctPar< T > &fitFctPar)
- Cart srt using sse.*
- template<class T >
static ClassAtom< T, uli_t > * CARTginiSrtWithMissings (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * trendCARTgini (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * LOTUS (FittingFctPar< T > &fitFctPar)
 - template<class T >
static void checkUsabilityLOTUS (DataFrame< T > &data)
 - template<class T >
static ClassAtom< T, uli_t > * SCARTgini (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * unbiasedCARTgini__ (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * unbiasedCARTgini (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * imputeTree (FittingFctPar< T > &fitFctPar)
 - template<class T >
static ClassAtom< T, uli_t > * CARTginiCor (FittingFctPar< T > &fitFctPar)

6.10.1 Detailed Description

Definition at line 103 of file FittingFct.h.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 FittingFct::FittingFct()

6.10.2.2 virtual FittingFct::~FittingFct() [virtual]

6.10.3 Member Function Documentation

6.10.3.1 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTgini (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 1000 of file FittingFct.h.

6.10.3.2 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTginiCor (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 3938 of file FittingFct.h.

6.10.3.3 template<class T > static ClassAtom<T, uli_t>* FittingFct::CAR-TginiNoDenominator (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 1210 of file FittingFct.h.

6.10.3.4 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTginiSrt (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 1868 of file FittingFct.h.

6.10.3.5 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTginiSrtOld (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 1648 of file FittingFct.h.

6.10.3.6 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTgini-SrtWithMissings (FittingFctPar< T > & *fitFctPar*) [inline, static]

Definition at line 2265 of file FittingFct.h.

6.10.3.7 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTginisse (FittingFctPar< T > & *fitFctPar*) [inline, static]

Cart srt using sse.

Parameters

<i>fitFctPar</i>

Returns

Definition at line 2079 of file FittingFct.h.

6.10.3.8 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTreg (FittingFctPar< T > & fFP) [inline, static]

Definition at line 116 of file FittingFct.h.

6.10.3.9 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTregTwoing (FittingFctPar< T > & fitFctPar) [inline, static]

Definition at line 285 of file FittingFct.h.

6.10.3.10 template<class T > static ClassAtom<T, uli_t>* FittingFct::CARTtwoing (FittingFctPar< T > & fitFctPar) [inline, static]

Definition at line 495 of file FittingFct.h.

6.10.3.11 template<class T > static ClassAtom<T, uli_t>* FittingFct::CAR-TwoWayIntAdd (FittingFctPar< T > & fitFctPar) [inline, static]

Definition at line 726 of file FittingFct.h.

6.10.3.12 template<class T > static void FittingFct::checkUsabilityLOTUS (DataFrame< T > & data) [inline, static]

Definition at line 3102 of file FittingFct.h.

6.10.3.13 template<class T > static void FittingFct::checkUsabilitySse (DataFrame< T > & data) [inline, static]

Definition at line 966 of file FittingFct.h.

6.10.3.14 template<class T > static ClassAtom<T, uli_t>* FittingFct::imputeTree (FittingFctPar< T > & fitFctPar) [inline, static]

Definition at line 3740 of file FittingFct.h.

6.10.3.15 template<class T > static ClassAtom<T, uli_t>* FittingFct::LOTUS (FittingFctPar< T > & fitFctPar) [inline, static]

Definition at line 2830 of file FittingFct.h.

```
6.10.3.16 template<class T> static ClassAtom<T, uli_t>* FittingFct::SCARTgini (
    FittingFctPar< T > & fitFctPar ) [inline, static]
```

Definition at line 3123 of file FittingFct.h.

```
6.10.3.17 template<class T> static ClassAtom<T, uli_t>* FittingFct::trendCARTgini (
    FittingFctPar< T > & fitFctPar ) [inline, static]
```

Definition at line 2487 of file FittingFct.h.

```
6.10.3.18 template<class T> static ClassAtom<T, uli_t>* FittingFct-
    ::unbiasedCARTgini ( FittingFctPar< T > & fitFctPar ) [inline,
    static]
```

Definition at line 3513 of file FittingFct.h.

```
6.10.3.19 template<class T> static ClassAtom<T, uli_t>* FittingFct::unbiased-
    CARTgini__ ( FittingFctPar< T > & fitFctPar ) [inline,
    static]
```

Definition at line 3220 of file FittingFct.h.

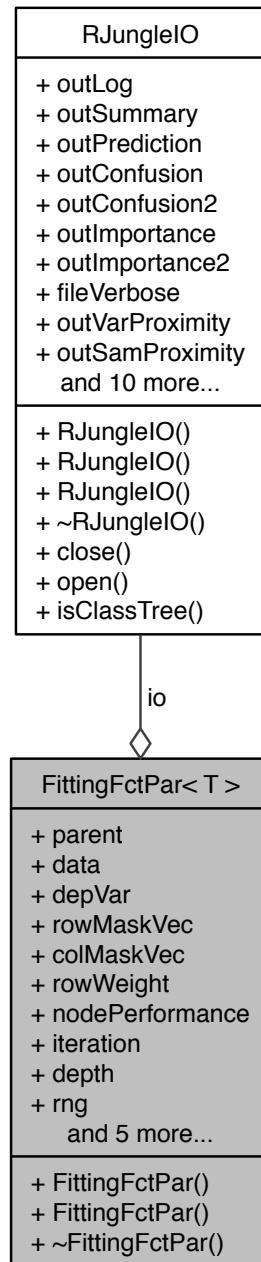
The documentation for this class was generated from the following file:

- [src/library/FittingFct.h](#)

6.11 FittingFctPar< T > Class Template Reference

```
#include <FittingFct.h>
```

Collaboration diagram for FittingFctPar< T >:



Public Member Functions

- `FittingFctPar ()`
- `FittingFctPar (INode< T > *parent, DataFrame< T > *data, uli_t depVar, std::vector< uli_t > *rowMaskVec, std::vector< uli_t > *colMaskVec, std::vector< double > *rowWeight, double nodePerformance, uli_t iteration, uli_t depth, gsl_rng *rng, Importance< T > *importance, double *treelImprovement, uli_t *bestVar, bool stopGrowing, std::vector< std::pair< double, std::pair< uli_t, uli_t > > > *twoWayInteraction, RJungleIO *io)`
- `virtual ~FittingFctPar ()`

Public Attributes

- `INode< T > * parent`
- `DataFrame< T > * data`
- `uli_t depVar`
- `std::vector< uli_t > * rowMaskVec`
- `std::vector< uli_t > * colMaskVec`
- `std::vector< double > * rowWeight`
- `double nodePerformance`
- `uli_t iteration`
- `uli_t depth`
- `gsl_rng * rng`
- `Importance< T > * importance`
- `double * treelImprovement`
- `uli_t * bestVar`
- `bool stopGrowing`
- `std::vector< std::pair< double, std::pair< uli_t, uli_t > > > * twoWayInteraction`
- `RJungleIO * io`

6.11.1 Detailed Description

`template<class T>class FittingFctPar< T >`

Definition at line 46 of file FittingFct.h.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 `template<class T> FittingFctPar< T >::FittingFctPar() [inline]`

Definition at line 48 of file FittingFct.h.

```
6.11.2.2 template<class T> FittingFctPar< T >::FittingFctPar ( INode< T > * parent,
    DataFrame< T > * data, uli_t depVar, std::vector< uli_t > * rowMaskVec,
    std::vector< uli_t > * colMaskVec, std::vector< double > * rowWeight, double
    nodePerformance, uli_t iteration, uli_t depth, gsl_rng * rng, Importance< T
    > * importance, double * treeImprovement, uli_t * bestVar, bool stopGrowing,
    std::vector< std::pair< double, std::pair< uli_t, uli_t > > > * twoWayInteraction,
    RJungleIO * io ) [inline]
```

Definition at line 55 of file FittingFct.h.

```
6.11.2.3 template<class T> virtual FittingFctPar< T >::~FittingFctPar ( )
    [inline, virtual]
```

Definition at line 80 of file FittingFct.h.

6.11.3 Member Data Documentation

```
6.11.3.1 template<class T> uli_t* FittingFctPar< T >::bestVar
```

Definition at line 96 of file FittingFct.h.

```
6.11.3.2 template<class T> std::vector<uli_t>* FittingFctPar< T >::colMaskVec
```

Definition at line 87 of file FittingFct.h.

```
6.11.3.3 template<class T> DataFrame<T>* FittingFctPar< T >::data
```

Definition at line 84 of file FittingFct.h.

```
6.11.3.4 template<class T> uli_t FittingFctPar< T >::depth
```

Definition at line 91 of file FittingFct.h.

```
6.11.3.5 template<class T> uli_t FittingFctPar< T >::depVar
```

Definition at line 85 of file FittingFct.h.

```
6.11.3.6 template<class T> Importance<T>* FittingFctPar< T >::importance
```

Definition at line 94 of file FittingFct.h.

6.11.3.7 template<class T> R JungleIO* FittingFctPar< T >::io

Definition at line 99 of file FittingFct.h.

6.11.3.8 template<class T> uli_t FittingFctPar< T >::iteration

Definition at line 90 of file FittingFct.h.

6.11.3.9 template<class T> double FittingFctPar< T >::nodePerformance

Definition at line 89 of file FittingFct.h.

6.11.3.10 template<class T> INode<T>* FittingFctPar< T >::parent

Definition at line 83 of file FittingFct.h.

6.11.3.11 template<class T> gsl_rng* FittingFctPar< T >::rng

Definition at line 92 of file FittingFct.h.

6.11.3.12 template<class T> std::vector<uli_t>* FittingFctPar< T >::rowMaskVec

Definition at line 86 of file FittingFct.h.

6.11.3.13 template<class T> std::vector<double>* FittingFctPar< T >::rowWeight

Definition at line 88 of file FittingFct.h.

6.11.3.14 template<class T> bool FittingFctPar< T >::stopGrowing

Definition at line 97 of file FittingFct.h.

6.11.3.15 template<class T> double* FittingFctPar< T >::treeImprovement

Definition at line 95 of file FittingFct.h.

6.11.3.16 template<class T> std::vector<std::pair<double, std::pair<uli_t, uli_t>>>* FittingFctPar< T >::twoWayInteraction

Definition at line 98 of file FittingFct.h.

The documentation for this class was generated from the following file:

- [src/library/FittingFct.h](#)

6.12 Generator< T > Class Template Reference

```
#include <Generator.h>
```

Public Member Functions

- [Generator \(\)](#)
- [virtual ~Generator \(\)](#)

Static Public Member Functions

- [virtual static GETBESTVARFCT \(T\)](#)
- [virtual static GETNODEPERFFCT \(T\)](#)
- [virtual static GETTERMRESFCT \(T\)](#)
- [virtual static CMPLTREEFCT \(T\)](#)
- [virtual static CLASSIFYCMPLDTREEFCT \(T\)](#)

6.12.1 Detailed Description

```
template<class T>class Generator< T >
```

Definition at line 29 of file Generator.h.

6.12.2 Constructor & Destructor Documentation

```
6.12.2.1 template<class T> Generator< T >::Generator( ) [inline]
```

Definition at line 32 of file Generator.h.

```
6.12.2.2 template<class T> virtual Generator< T >::~Generator( ) [inline,  
virtual]
```

Definition at line 33 of file Generator.h.

6.12.3 Member Function Documentation

```
6.12.3.1 template<class T> virtual static Generator< T >::CLASSIFYCMPLDTREEFCT  
( T ) [inline, static, virtual]
```

Definition at line 52 of file Generator.h.

6.12.3.2 `template<class T> virtual static Generator< T >::CMPLTREEFCT(T)`
[inline, static, virtual]

Definition at line 48 of file Generator.h.

6.12.3.3 `template<class T> virtual static Generator< T >::GETBESTVARFCT(T)`
[inline, static, virtual]

Definition at line 36 of file Generator.h.

6.12.3.4 `template<class T> virtual static Generator< T >::GETNODEPERFFCT(T)`
[inline, static, virtual]

Definition at line 40 of file Generator.h.

6.12.3.5 `template<class T> virtual static Generator< T >::GETTERMRESFCT(T)`
[inline, static, virtual]

Definition at line 44 of file Generator.h.

The documentation for this class was generated from the following file:

- [src/library/Generator.h](#)

6.13 Helper Class Reference

```
#include <Helper.h>
```

Public Member Functions

- [Helper \(\)](#)
- [virtual ~Helper \(\)](#)

Static Public Member Functions

- [static void printTime \(double timeEst, std::ostream &outVerbose\)](#)
Converts seconds to human readable format and prints it to output stream.
- [static std::vector< std::string > getColSelection \(std::string colSelection, char delimiter= ':'\)](#)
Reads column names from a file.
- [static std::pair< uli_t, uli_t > getCSVdim \(char *fileName, char delimiter= ','\)](#)
Returns the dimensions of data in input file.

- template<class T >
static uli_t getSize (std::vector< T > &vec, T missingCode)
Gets number of non missing values in vector.
- template<class T >
static double purityLotus (DataFrame< T > &data, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
Purity of lotus node.
- template<class T >
static double performLrAndGetDeviance (std::vector< T > nodeVec, std::vector< T > outcomeVec)
- template<class T >
static double purityGini (DataFrame< T > &data, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
Returns the Gini index of a specific data column.
- template<class T >
static double purityTwoing (DataFrame< T > &data, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
Returns the purity of a specific data column.
- template<class T >
static double puritySos (DataFrame< T > &data, uli_t depVar, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
Returns the purity of a specific data column.
- template<class T >
static void getClassifiedVectors (DataFrame< T > &data, std::vector< uli_t > &rowMaskVec, ClassAtom< T, uli_t > *classifier, std::vector< uli_t > &classVec, std::vector< std::vector< uli_t > * > &classMaskMultiVec)
Returns classification of samples sorted by prediction class.
- template<class T >
static double getSum (std::vector< T > &dataVec)
- template<class T >
static std::vector< T > getAbs (std::vector< T > &dataVec)
- template<class T >
static std::vector< T > getDiff (std::vector< T > &dataVec, T val)
- template<class T >
static T getMad (std::vector< T > &dataVec, double constant=1.4826)
- template<class T >
static double getMean (std::vector< T > &dataVec)
- template<class T >
static T getMean2 (std::vector< T > &dataVec, T missingCode)
- template<class T >
static T getMean4 (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
- template<class T >
static T getMean3 (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
- template<class T >
static T getMeanProx (unsigned int row1, std::vector< T > &dataVec, T missingCode, Proximities< T > &prox)

- template<class T >
static T [getMeanProxX](#) (unsigned int row1, std::vector< T > &dataVec, T missingCode, [Proximities](#)< T > &proxi)
- template<class T >
static std::vector< size_t > [getRanks](#) (std::vector< T > &in, [bool](#) ascending=true)
- template<class T >
static void [getRanks](#) (std::vector< T > &in, std::vector< size_t > &out, [bool](#) ascending=true)
- template<class T >
static void [getRanks](#) (DataFrame< T > &data, std::vector< uli_t > &idx, uli_t j, std::vector< size_t > &out, [bool](#) ascending=true)
- template<class T >
static double [getMedian](#) (std::vector< T > &dataVec)
- template<class T >
static T [getMedianX2](#) (std::vector< T > &dataVec)
- template<class T >
static T [getMedianX](#) (std::vector< T > &dataVec)
- template<class T >
static T [getOne](#) (std::vector< T > &dataVec, T missingCode)
- static uli_t [getMostFreqIndex](#) (std::vector< uli_t > &idxVec, uli_t termCode)
- template<class T >
static T [getMostFreq](#) (std::vector< T > &dataVec, T missingCode)
- template<class T >
static T [getMostFreqProp](#) (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
- template<class T >
static T [getMostFreqProp](#) (RJunglePar &par, std::vector< T > &dataVec, T missingCode, DataFrame< T > *data=NULL, [bool](#) showInfos=false, std::ostream *outStream=NULL, uli_t sample=0)
- template<class T >
static T [getMostFreqX](#) (std::vector< T > &dataVec, T missingCode)
- template<class T >
static double [getMode](#) (std::vector< T > &dataVec)
- template<class T >
static T [getMostFreqProx](#) (unsigned int row1, std::vector< T > &dataVec, T missingCode, [Proximities](#)< T > &proxi)
- template<class T >
static std::pair< T, double > [getMostFreqProx2](#) (unsigned int row1, std::vector< T > &dataVec, T missingCode, [Proximities](#)< T > &proxi)
- template<class T >
static std::pair< T, double > [getMostFreqLD](#) (unsigned int row1, std::vector< T > &dataVec, uli_t missingCode, std::vector< double > &ldVec)
- template<class T >
static T [getMedianWithSort](#) (std::vector< T > &dataVec)
- template<class T >
static void [getCountOfInterval](#) (std::vector< T > &dataVec, std::vector< double > &quantVec, std::vector< uli_t > &outVec)

- template<class T >
static T **getMax** (std::vector< T > &dataVec)
- template<class T >
static T **getMin** (std::vector< T > &dataVec)
- template<class T , class C >
static void **inverseMap** (const std::map< T, C > &o, std::map< C, T > &result)
- template<class T , class C >
static void **makePairVec** (std::vector< T > &colVec, std::vector< C > &idxVec,
std::vector< std::pair< T, C > > &result)
- template<class T , class C >
static void **makeMap** (std::vector< T > &colVec, std::vector< C > &idxVec, std-
::map< T, C > &result)
- template<class T >
static void **printVec** (std::vector< T > &vec)
- template<class T >
static void **printVec** (T *vec, size_t len)
- template<class T >
static void **printVec** (const std::vector< T > &vec)
- template<class T >
static bool **isElement** (T elem, std::vector< T > &vec)
- template<class T >
static void **getMask** (T classVal, std::vector< T > &inVec, std::vector< uli_t >
&maskVec)
- template<class T >
static void **getMaskedVec** (std::vector< T > &inVec, std::vector< uli_t > &mask-
Vec, std::vector< T > &outVec)
- template<class T >
static void **getIndexedVector** (std::vector< T > &inVec, std::vector< uli_t >
&indexesVec, std::vector< T > &outVec)
- template<class T >
static void **getClasses** (std::vector< T > &inVec, std::vector< T > &classVec)
- template<class T >
static void **getClassesAndRate** (std::vector< T > &inVec, typename std::map<
T, unsigned int > &classMap)
- template<class C , class T , class D >
static void **printTreeMap** (std::map< C, T > &mapInverse, DataFrame< D >
&data, uli_t ntree)
- template<class D >
static void **printTreePair** (std::vector< std::pair< double, uli_t > > &freq-
Pairs, DataFrame< D > &data, uli_t ntree, uli_t iteration=0, std::ostream
&outstream=std::cout)
- template<class D >
static void **printTreePair2Way** (std::vector< std::pair< double, uli_t > > &freq-
Pairs, std::vector< std::pair< uli_t, uli_t > > &indexer, DataFrame< D > &data,
uli_t ntree, uli_t iteration=0, std::ostream &outstream=std::cout)
- static void **makeMap2SortedInvPair** (std::map< uli_t, double > &freqMap, std-
::vector< std::pair< double, uli_t > > &freqPairs)
- static bool **isOdd** (uli_t val)

- static `bool isEven (uli_t val)`
- template<class T >
 static double `getPredScore` (std::vector< T > &outcomeVec, std::vector< T > &classMeVec)
- static double `getProp` (uli_t ntree, uli_t mtry, uli_t numOfVars)
- static uli_t `getTreeSizeX` (uli_t ntree, uli_t mtry1, uli_t numOfVars1, uli_t mtry2, uli_t numOfVars2)
- static uli_t `getTreeSizeProb` (double p, uli_t mtry, uli_t numOfVars)
- static void `getCSVdim` (RJunglePar &par)
- static void `add` (std::vector< double > &to, std::vector< double > &from)
- static void `addMasked` (std::vector< double > &to, std::vector< uli_t > &maskVec, std::vector< double > &from)
- static void `vecToPairs` (std::vector< std::pair< double, uli_t > > &pairs, std::vector< double > &vec)
- static void `makeToFreq` (std::vector< double > &vec)
- template<class T >
 static void `getPowerSet` (boost::dynamic_bitset<> &powerSet, std::vector< T > &from, std::vector< T > &to)
- static void `inc` (boost::dynamic_bitset<> &bitset)
- template<class T >
 static void `getSNPsInHighLD` (DataFrame< T > &data, std::vector< double > &ldVecFull, std::vector< double > &ldVec, std::vector< uli_t > *&colMaskFullVec, std::vector< uli_t > *&colMaskVec)
- template<class T >
 static void `getSNPsInLD` (DataFrame< T > &data, unsigned int col, std::vector< uli_t > *colMaskVec, std::vector< double > &ldVec)
- template<class T >
 static void `vectorToXML` (std::vector< T > &vec, std::string &name, std::string &xmlString)
- static void `strSplit` (std::string &str, std::string delim, std::vector< std::string > &results)
- template<class T >
 static void `copy` (std::vector< T > vec1, T *vec2)
- template<class T >
 static double `one` (DataFrame< T > &data, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)
- static void `strSplit` (char *in, const char *delim, std::vector< std::string > &out)
- template<class T >
 static `bool isn` (T val, std::vector< T > &vec)
- static void `getSlicedVec` (uli_t ncol, std::vector< uli_t > &mtrys, double theta)
- template<class T >
 static void `clearPtrVec` (std::vector< T > &vec)
- template<class T >
 static void `seq` (std::vector< T > &vec, T from, T to, T step)
- static `bool isClassTree` (RJunglePar &par)
- static `bool isGuessingLomNumeric` (RJunglePar &par)

- template<class T >
 static double **getPearsonCor** (DataFrame< T > &data, std::vector< uli_t > &idx,
 uli_t j1, uli_t j2)
Calculates the sample correlation coefficient, i.e., Pearson's correlation coefficient between two variables.
- template<class T >
 static void **getIntervalGroup** (DataFrame< T > &data, std::vector< uli_t > &idx,
 uli_t j1, std::vector< T > &cutoffs, std::vector< uli_t > &groups)
- static void **convertOOBtoMask** (DataTreeSet &oobSet, long int treeld, std::vector< uli_t > &rowMaskVec)
- static void **addToCIGrid** (std::vector< uli_t > &groups, std::vector< uli_t > &grid)
- template<class T >
 static void **permuteInGrid** (gsl_rng *rng, T *vec, std::vector< uli_t > &grid)
- template<class T >
 static void **getQuantiles** (std::vector< double > &qvec, std::vector< std::pair< T, uli_t > > &rowPairVec, std::vector< T > &quantVec)
- template<class T >
 static void **getQuantilesLotusSplit** (std::vector< std::pair< T, uli_t > > &rowPairVec, std::vector< T > &quantVec)
- template<class T >
 static void **putPairFirstToVec** (std::vector< std::pair< T, uli_t > > &rowPairVec, std::vector< T > &vec)
- template<class T >
 static void **putPairSecondToVec** (std::vector< std::pair< T, uli_t > > &rowPairVec, std::vector< uli_t > &vec)

6.13.1 Detailed Description

Definition at line 45 of file Helper.h.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 Helper::Helper() [inline]

Definition at line 47 of file Helper.h.

6.13.2.2 virtual Helper::~Helper() [inline, virtual]

Definition at line 48 of file Helper.h.

6.13.3 Member Function Documentation

6.13.3.1 static void Helper::add(std::vector< double > & to, std::vector< double > & from) [inline, static]

Definition at line 1573 of file Helper.h.

6.13.3.2 **static void Helper::addMasked (std::vector< double > & to, std::vector< uli_t > & maskVec, std::vector< double > & from) [inline, static]**

Definition at line 1578 of file Helper.h.

6.13.3.3 **static void Helper::addToCIGrid (std::vector< uli_t > & groups, std::vector< uli_t > & grid) [inline, static]**

/brief Expand conditional importance grid by a new variable

Parameters

<i>groups</i>	
<i>grid</i>	

Definition at line 2042 of file Helper.h.

6.13.3.4 **template<class T> static void Helper::clearPtrVec (std::vector< T > & vec) [inline, static]**

Definition at line 1897 of file Helper.h.

6.13.3.5 **static void Helper::convertOOBtoMask (DataTreeSet & oobSet, long int treeld, std::vector< uli_t > & rowMaskVec) [inline, static]**

/brief Convert oobSet sample information to a traditional rowMaskVec

Parameters

<i>oobSet</i>	OOB set
<i>treeld</i>	ID of current tree
<i>rowMaskVec</i>	Output vector

Definition at line 2024 of file Helper.h.

6.13.3.6 **template<class T> static void Helper::copy (std::vector< T > vec1, T * vec2) [inline, static]**

Definition at line 1831 of file Helper.h.

6.13.3.7 **template<class T> static std::vector< T > Helper::getAbs (std::vector< T > & dataVec) [inline, static]**

Definition at line 465 of file Helper.h.

6.13.3.8 template<class T > static void Helper::getClasses (std::vector< T > & inVec,
std::vector< T > & classVec) [inline, static]

Definition at line 1323 of file Helper.h.

6.13.3.9 template<class T > static void Helper::getClassesAndRate (std::vector< T
> & inVec, typename std::map< T, unsigned int > & classMap) [inline,
static]

Definition at line 1337 of file Helper.h.

6.13.3.10 template<class T > static void Helper::getClassifiedVectors (DataFrame<
T > & data, std::vector< uli_t > & rowMaskVec, ClassAtom< T, uli_t > *
classifier, std::vector< uli_t > & classVec, std::vector< std::vector< uli_t > * >
& classMaskMultiVec) [inline, static]

Returns classification of samples sorted by prediction class.

Parameters

<i>data</i>	Data frame.
<i>rowMaskVec</i>	Selection of samples.
<i>classifier</i>	Classifier which classifies the samples.
<i>classVec</i>	Classes in vectors.
<i>classMask- MultiVec</i>	Sample index of different prediction classes.

Definition at line 422 of file Helper.h.

6.13.3.11 static std::vector<std::string> Helper::getColSelection (std::string
colSelection, char *delimiter* = ' ; ') [inline, static]

Reads column names from a file.

Parameters

<i>colSelection</i>	File that contains the column names.
<i>delimiter</i>	Delimeter (";" is default or use new line).

Returns

Column names.

Definition at line 79 of file Helper.h.

```
6.13.3.12 template<class T > static void Helper::getCountOfInterval ( std::vector< T >
& dataVec, std::vector< double > & quantVec, std::vector< uli_t > & outVec )
[inline, static]
```

Definition at line 1180 of file Helper.h.

```
6.13.3.13 static std::pair<uli_t, uli_t> Helper::getCSVdim ( char * fileName, char
delimiter = ';' ) [inline, static]
```

Returns the dimensions of data in input file.

Parameters

<i>fileName</i>	File name of input data file.
<i>delimiter</i>	Delimeter that seperates data cells. (";" is default).

Returns

Dimension.

Definition at line 144 of file Helper.h.

```
6.13.3.14 static void Helper::getCSVdim ( R JunglePar & par ) [inline,
static]
```

Definition at line 1515 of file Helper.h.

```
6.13.3.15 template<class T > static std::vector<T> Helper::getDiff ( std::vector< T > &
dataVec, T val ) [inline, static]
```

Definition at line 478 of file Helper.h.

```
6.13.3.16 template<class T > static void Helper::getIndexedVector ( std::vector< T >
& inVec, std::vector< uli_t > & indexesVec, std::vector< T > & outVec )
[inline, static]
```

Definition at line 1309 of file Helper.h.

```
6.13.3.17 template<class T > static void Helper::getIntervalGroup ( DataFrame<
T > & data, std::vector< uli_t > & idx, uli_t j1, std::vector< T > & cutoffs,
std::vector< uli_t > & groups ) [inline, static]
```

/brief Group input vector elements using interval vector. (Assign element to a interval)

Parameters

<i>data</i>	Input data set (DataFrame)
<i>idx</i>	Indexes of input observations
<i>j1</i>	Index of variable
<i>cutoffs</i>	Vector of cutoff values (intervals)
<i>groups</i>	Resulting interval assignment

Definition at line 1982 of file Helper.h.

6.13.3.18 template<class T > static T Helper::getMad (std::vector< T > & *dataVec*, double *constant* = 1.4826) [inline, static]

Definition at line 504 of file Helper.h.

6.13.3.19 template<class T > static void Helper::getMask (T *classVal*, std::vector< T > & *inVec*, std::vector< uli_t > & *maskVec*) [inline, static]

Definition at line 1285 of file Helper.h.

6.13.3.20 template<class T > static void Helper::getMaskedVec (std::vector< T > & *inVec*, std::vector< uli_t > & *maskVec*, std::vector< T > & *outVec*) [inline, static]

Definition at line 1297 of file Helper.h.

6.13.3.21 template<class T > static T Helper::getMax (std::vector< T > & *dataVec*) [inline, static]

Definition at line 1189 of file Helper.h.

6.13.3.22 template<class T > static double Helper::getMean (std::vector< T > & *dataVec*) [inline, static]

Definition at line 521 of file Helper.h.

6.13.3.23 template<class T > static T Helper::getMean2 (std::vector< T > & *dataVec*, T *missingCode*) [inline, static]

Definition at line 545 of file Helper.h.

```
6.13.3.24 template<class T> static T Helper::getMean3 ( DataFrame< T > & data,
RJunglePar & par, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector<
double > & rowWeight ) [inline, static]
```

Definition at line 581 of file Helper.h.

```
6.13.3.25 template<class T> static T Helper::getMean4 ( DataFrame< T > & data,
RJunglePar & par, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector<
double > & rowWeight ) [inline, static]
```

Definition at line 561 of file Helper.h.

```
6.13.3.26 template<class T> static T Helper::getMeanProx ( unsigned int row1,
std::vector< T > & dataVec, T missingCode, Proximities< T > & prox )
[inline, static]
```

Definition at line 601 of file Helper.h.

```
6.13.3.27 template<class T> static T Helper::getMeanProxX ( unsigned int row1,
std::vector< T > & dataVec, T missingCode, Proximities< T > & prox )
[inline, static]
```

Definition at line 617 of file Helper.h.

```
6.13.3.28 template<class T> static double Helper::getMedian ( std::vector< T > &
dataVec ) [inline, static]
```

Definition at line 721 of file Helper.h.

```
6.13.3.29 template<class T> static T Helper::getMedianWithSort ( std::vector< T > &
dataVec ) [inline, static]
```

Definition at line 1164 of file Helper.h.

```
6.13.3.30 template<class T> static T Helper::getMedianX ( std::vector< T > & dataVec )
[inline, static]
```

Definition at line 743 of file Helper.h.

```
6.13.3.31 template<class T> static T Helper::getMedianX2 ( std::vector< T > & dataVec
) [inline, static]
```

Definition at line 732 of file Helper.h.

6.13.3.32 `template<class T > static T Helper::getMin (std::vector< T > & dataVec) [inline, static]`

Definition at line 1201 of file Helper.h.

6.13.3.33 `template<class T > static double Helper::getMode (std::vector< T > & dataVec) [inline, static]`

Definition at line 985 of file Helper.h.

6.13.3.34 `template<class T > static T Helper::getMostFreq (std::vector< T > & dataVec, T missingCode) [inline, static]`

Definition at line 768 of file Helper.h.

6.13.3.35 `static uli_t Helper::getMostFreqIndex (std::vector< uli_t > & idxVec, uli_t termCode) [inline, static]`

Definition at line 753 of file Helper.h.

6.13.3.36 `template<class T > static std::pair< T, double > Helper::getMostFreqLD (unsigned int row1, std::vector< T > & dataVec, uli_t missingCode, std::vector< double > & IdVec) [inline, static]`

Definition at line 1113 of file Helper.h.

6.13.3.37 `template<class T > static T Helper::getMostFreqProp (DataFrame< T > & data, RJunglePar & par, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< double > & rowWeight) [inline, static]`

Definition at line 808 of file Helper.h.

6.13.3.38 `template<class T > static T Helper::getMostFreqProp (RJunglePar & par, std::vector< T > & dataVec, T missingCode, DataFrame< T > * data = NULL, bool showInfos = false, std::ostream * outStream = NULL, uli_t sample = 0) [inline, static]`

Definition at line 853 of file Helper.h.

6.13.3.39 `template<class T > static T Helper::getMostFreqProx (unsigned int row1, std::vector< T > & dataVec, T missingCode, Proximities< T > & prox) [inline, static]`

Definition at line 1021 of file Helper.h.

6.13.3.40 `template<class T> static std::pair<T, double> Helper::getMostFreqProx2 (unsigned int row1, std::vector< T > & dataVec, T missingCode, Proximities< T > & prox1) [inline, static]`

Definition at line 1065 of file Helper.h.

6.13.3.41 `template<class T> static T Helper::getMostFreqX (std::vector< T > & dataVec, T missingCode) [inline, static]`

Definition at line 946 of file Helper.h.

6.13.3.42 `template<class T> static T Helper::getOne (std::vector< T > & dataVec, T missingCode) [inline, static]`

Definition at line 749 of file Helper.h.

6.13.3.43 `template<class T> static double Helper::getPearsonCor (DataFrame< T > & data, std::vector< uli_t > & idx, uli_t j1, uli_t j2) [inline, static]`

Calculates the sample correlation coefficient, i.e., Pearson's correlation coefficient between two variables.

Parameters

<i>idx</i>	Index vector of samples
<i>j1</i>	Index of 1st variable
<i>j2</i>	Index of 2nd variable

Returns

Sample correlation coefficient

Definition at line 1940 of file Helper.h.

6.13.3.44 `template<class T> static void Helper::getPowerSet (boost::dynamic_bitset<> & powerSet, std::vector< T > & from, std::vector< T > & to) [inline, static]`

Definition at line 1602 of file Helper.h.

6.13.3.45 `template<class T> static double Helper::getPredScore (std::vector< T > & outcomeVec, std::vector< T > & classMeVec) [inline, static]`

Definition at line 1449 of file Helper.h.

6.13.3.46 static double Helper::getProp (uli_t ntree, uli_t mtry, uli_t numOfVars)
[inline, static]

Definition at line 1474 of file Helper.h.

6.13.3.47 template<class T > static void Helper::getQuantiles (std::vector< double > & qvec, std::vector< std::pair< T, uli_t > > & rowPairVec, std::vector< T > & quantVec) [inline, static]

Definition at line 2121 of file Helper.h.

6.13.3.48 template<class T > static void Helper::getQuantilesLotusSplit (std::vector< std::pair< T, uli_t > > & rowPairVec, std::vector< T > & quantVec)
[inline, static]

Definition at line 2147 of file Helper.h.

6.13.3.49 template<class T > static std::vector< size_t > Helper::getRanks (std::vector< T > & in, bool ascending = true) [inline, static]

Definition at line 636 of file Helper.h.

6.13.3.50 template<class T > static void Helper::getRanks (std::vector< T > & in, std::vector< size_t > & out, bool ascending = true) [inline, static]

Definition at line 648 of file Helper.h.

6.13.3.51 template<class T > static void Helper::getRanks (DataFrame< T > & data, std::vector< uli_t > & idx, uli_t j, std::vector< size_t > & out, bool ascending = true) [inline, static]

Definition at line 679 of file Helper.h.

6.13.3.52 template<class T > static uli_t Helper::getSize (std::vector< T > & vec, T missingCode) [inline, static]

Gets number of non missing values in vector.

Parameters

<i>vec</i>	Vector.
<i>missing-Code</i>	Missing code.

Returns

Number of non missing values in vector.

Definition at line 201 of file Helper.h.

```
6.13.3.53 static void Helper::getSlicedVec ( uli_t ncol, std::vector<uli_t> & mtrys,  
double theta ) [inline, static]
```

Definition at line 1869 of file Helper.h.

```
6.13.3.54 template<class T> static void Helper::getSNPsInHighLD ( DataFrame< T  
> & data, std::vector< double > & IdVecFull, std::vector< double > & IdVec,  
std::vector< uli_t > * & colMaskFullVec, std::vector< uli_t > * & colMaskVec )  
[inline, static]
```

Definition at line 1632 of file Helper.h.

```
6.13.3.55 template<class T> static void Helper::getSNPsInLD ( DataFrame< T > &  
data, unsigned int col, std::vector< uli_t > * colMaskVec, std::vector< double >  
& IdVec ) [inline, static]
```

Definition at line 1677 of file Helper.h.

```
6.13.3.56 template<class T> static double Helper::getSum ( std::vector< T > & dataVec )  
[inline, static]
```

Definition at line 453 of file Helper.h.

```
6.13.3.57 static uli_t Helper::getTreeSizeProb ( double p, uli_t mtry, uli_t numVars )  
[inline, static]
```

Definition at line 1503 of file Helper.h.

```
6.13.3.58 static uli_t Helper::getTreeSizeX ( uli_t ntree, uli_t mtry1, uli_t numVars1,  
uli_t mtry2, uli_t numVars2 ) [inline, static]
```

Definition at line 1485 of file Helper.h.

```
6.13.3.59 static void Helper::inc ( boost::dynamic_bitset<> & bitset ) [inline,  
static]
```

Definition at line 1622 of file Helper.h.

6.13.3.60 **template<class T , class C > static void Helper::inverseMap (const std::map< T, C > & o, std::map< C, T > & result)** [inline, static]

Definition at line 1213 of file Helper.h.

6.13.3.61 **static bool Helper::isClassTree (RJunglePar & par)** [inline, static]

Definition at line 1917 of file Helper.h.

6.13.3.62 **template<class T > static bool Helper::isElement (T elem, std::vector< T > & vec)** [inline, static]

Definition at line 1270 of file Helper.h.

6.13.3.63 **static bool Helper::isEven (uli_t val)** [inline, static]

Definition at line 1444 of file Helper.h.

6.13.3.64 **static bool Helper::isGuessingLomNumeric (RJunglePar & par)** [inline, static]

Definition at line 1924 of file Helper.h.

6.13.3.65 **template<class T > static bool Helper::isIn (T val, std::vector< T > & vec)** [inline, static]

Definition at line 1856 of file Helper.h.

6.13.3.66 **static bool Helper::isOdd (uli_t val)** [inline, static]

Definition at line 1440 of file Helper.h.

6.13.3.67 **template<class T , class C > static void Helper::makeMap (std::vector< T > & colVec, std::vector< C > & idxVec, std::map< T, C > & result)** [inline, static]

Definition at line 1235 of file Helper.h.

6.13.3.68 **static void Helper::makeMap2SortedInvPair (std::map< uli_t, double > & freqMap, std::vector< std::pair< double, uli_t > > & freqPairs)** [inline, static]

Definition at line 1426 of file Helper.h.

```
6.13.3.69 template<class T , class C > static void Helper::makePairVec ( std::vector< T >
& colVec, std::vector< C > & idxVec, std::vector< std::pair< T, C > > & result )
[inline, static]
```

Definition at line 1221 of file Helper.h.

```
6.13.3.70 static void Helper::makeToFreq ( std::vector< double > & vec ) [inline,
static]
```

Definition at line 1591 of file Helper.h.

```
6.13.3.71 template<class T > static double Helper::one ( DataFrame< T > & data, uli_t
col, std::vector< uli_t > & rowMaskVec, std::vector< double > & rowWeight )
[inline, static]
```

Definition at line 1837 of file Helper.h.

```
6.13.3.72 template<class T > static double Helper::performLrAndGetDeviance
( std::vector< T > nodeVec, std::vector< T > outcomeVec ) [inline,
static]
```

Definition at line 243 of file Helper.h.

```
6.13.3.73 template<class T > static void Helper::permuteInGrid ( gsl_rng * rng, T * vec,
std::vector< uli_t > & grid ) [inline, static]
```

/brief Permutes values in vec using conditional grid

Parameters

<i>rng</i>	Random number generator (GSL)
<i>vec</i>	Vector of Observations
<i>grid</i>	Conditional grid

Definition at line 2081 of file Helper.h.

```
6.13.3.74 static void Helper::printTime ( double timeEst, std::ostream & outVerbose )
[inline, static]
```

Converts seconds to human readable format and prints it to output stream.

Parameters

<i>timeEst</i>	Seconds
<i>outVerbose</i>	Output stream.

Definition at line 57 of file Helper.h.

```
6.13.3.75 template<class C , class T , class D > static void Helper::printTreeMap ( std::map< C, T > & mapInverse, DataFrame< D > & data, uli_t ntree ) [inline, static]
```

Definition at line 1352 of file Helper.h.

```
6.13.3.76 template<class D > static void Helper::printTreePair ( std::vector< std::pair< double, uli_t > > & freqPairs, DataFrame< D > & data, uli_t ntree, uli_t iteration = 0, std::ostream & ostream = std::cout ) [inline, static]
```

Definition at line 1369 of file Helper.h.

```
6.13.3.77 template<class D > static void Helper::printTreePair2Way ( std::vector< std::pair< std::pair< double, uli_t > > & freqPairs, std::vector< std::pair< uli_t, uli_t > > & indexer, DataFrame< D > & data, uli_t ntree, uli_t iteration = 0, std::ostream & ostream = std::cout ) [inline, static]
```

Definition at line 1394 of file Helper.h.

```
6.13.3.78 template<class T > static void Helper::printVec ( std::vector< T > & vec ) [inline, static]
```

Definition at line 1249 of file Helper.h.

```
6.13.3.79 template<class T > static void Helper::printVec ( T * vec, size_t len ) [inline, static]
```

Definition at line 1256 of file Helper.h.

```
6.13.3.80 template<class T > static void Helper::printVec ( const std::vector< T > & vec ) [inline, static]
```

Definition at line 1263 of file Helper.h.

```
6.13.3.81 template<class T > static double Helper::purityGini ( DataFrame< T > & data, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< double > & rowWeight ) [inline, static]
```

Returns the Gini index of a specific data column.

Parameters

<i>data</i>	Data frame.
<i>col</i>	Index of column in data frame.
<i>rowMaskVec</i>	Selection of samples.
<i>rowWeight</i>	Weights of samples.

Returns

Gini index.

Definition at line 287 of file Helper.h.

```
6.13.3.82 template<class T> static double Helper::purityLotus ( DataFrame< T > &
    data, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< double > &
    rowWeight ) [inline, static]
```

Purity of lotus node.

Parameters

<i>data</i>	global data
<i>col</i>	variable index
<i>rowMaskVec</i>	selection of samples
<i>rowWeight</i>	weights of samples

Returns

deviance

Definition at line 224 of file Helper.h.

```
6.13.3.83 template<class T> static double Helper::puritySos ( DataFrame< T > &
    data, uli_t depVar, std::vector< uli_t > & rowMaskVec, std::vector< double > &
    rowWeight ) [inline, static]
```

Returns the purity of a specific data column.

Purity equals sum of squares divided by sample size.

Parameters

<i>data</i>	Data frame.
<i>col</i>	Index of column in data frame.
<i>rowMaskVec</i>	Selection of samples.
<i>rowWeight</i>	Weights of samples.

Returns

Purity.

Definition at line 385 of file Helper.h.

```
6.13.3.84 template<class T> static double Helper::purityTwoing ( DataFrame< T > &
    data, uli_t col, std::vector< uli_t > & rowMaskVec, std::vector< double > &
    rowWeight ) [inline, static]
```

Returns the purity of a specific data column.

If perfect clean then return 0 otherwise 0.5.

Parameters

<i>data</i>	Data frame.
<i>col</i>	Index of column in data frame.
<i>rowMaskVec</i>	Selection of samples.
<i>rowWeight</i>	Weights of samples.

Returns

Purity.

Definition at line 346 of file Helper.h.

```
6.13.3.85 template<class T> static void Helper::putPairFirstToVec ( std::vector<
    std::pair< T, uli_t > > & rowPairVec, std::vector< T > & vec ) [inline,
    static]
```

Definition at line 2158 of file Helper.h.

```
6.13.3.86 template<class T> static void Helper::putPairSecondToVec ( std::vector<
    std::pair< T, uli_t > > & rowPairVec, std::vector< uli_t > & vec ) [inline,
    static]
```

Definition at line 2174 of file Helper.h.

```
6.13.3.87 template<class T> static void Helper::seq ( std::vector< T > & vec, T from, T to,
    T step ) [inline, static]
```

Definition at line 1909 of file Helper.h.

```
6.13.3.88 static void Helper::strSplit ( std::string & str, std::string delim, std::vector<
    std::string > & results ) [inline, static]
```

Definition at line 1816 of file Helper.h.

```
6.13.3.89 static void Helper::strSplit ( char * in, const char * delim, std::vector< std::string > & out ) [inline, static]
```

Definition at line 1842 of file Helper.h.

```
6.13.3.90 static void Helper::vecToPairs ( std::vector< std::pair< double, uli_t > > & pairs, std::vector< double > & vec ) [inline, static]
```

Definition at line 1584 of file Helper.h.

```
6.13.3.91 template<class T> static void Helper::vectorToXML ( std::vector< T > & vec, std::string & name, std::string & xmlString ) [inline, static]
```

Definition at line 1795 of file Helper.h.

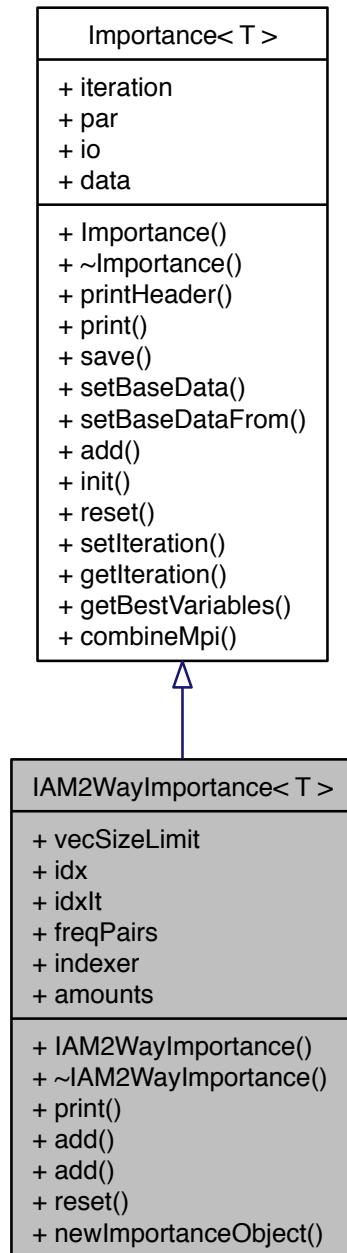
The documentation for this class was generated from the following file:

- [src/library/Helper.h](#)

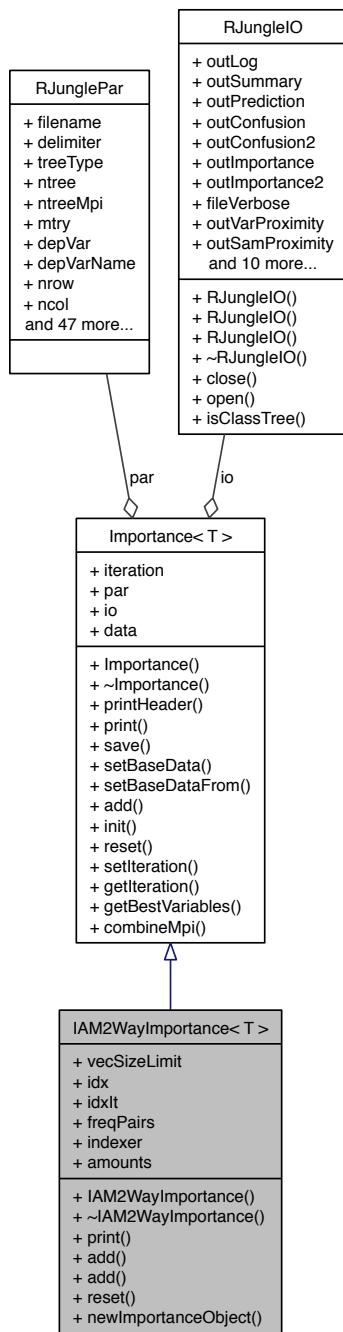
6.14 IAM2WayImportance< T > Class Template Reference

```
#include <IAM2WayImportance.h>
```

Inheritance diagram for IAM2WayImportance< T >:



Collaboration diagram for IAM2WayImportance< T >:



Public Member Functions

- `IAM2WayImportance (uli_t vecSizeLimit=1000000)`
- `virtual ~IAM2WayImportance ()`
- `virtual void print ()`
- `void add (uli_t varID1, uli_t varID2, double val)`
- `virtual void add (Importance< T > *impx)`
- `virtual void reset ()`

Static Public Member Functions

- `static Importance< T > * newImportanceObject ()`

Public Attributes

- `uli_t vecSizeLimit`
- `uli_t idx`
- `std::vector< std::pair< uli_t, uli_t > >::iterator idxIt`
- `std::vector< std::pair< double, uli_t > > freqPairs`
- `std::vector< std::pair< uli_t, uli_t > > indexer`
- `std::vector< double > amounts`

6.14.1 Detailed Description

`template<class T>class IAM2WayImportance< T >`

Definition at line 16 of file IAM2WayImportance.h.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 `template<class T> IAM2WayImportance< T >::IAM2WayImportance (uli_t vecSizeLimit = 1000000) [inline]`

Definition at line 18 of file IAM2WayImportance.h.

6.14.2.2 `template<class T> virtual IAM2WayImportance< T >::~IAM2WayImportance () [inline, virtual]`

Definition at line 21 of file IAM2WayImportance.h.

6.14.3 Member Function Documentation

6.14.3.1 `template<class T> void IAM2WayImportance< T >::add (uli_t varID1, uli_t varID2, double val) [inline]`

Definition at line 42 of file IAM2WayImportance.h.

6.14.3.2 `template<class T> virtual void IAM2WayImportance< T >::add (Importance< T > * impx) [inline, virtual]`

Reimplemented from [Importance< T >](#).

Definition at line 74 of file IAM2WayImportance.h.

6.14.3.3 `template<class T> static Importance< T >* IAM2WayImportance< T >::newImportanceObject () [inline, static]`

Definition at line 86 of file IAM2WayImportance.h.

6.14.3.4 `template<class T> virtual void IAM2WayImportance< T >::print () [inline, virtual]`

Reimplemented from [Importance< T >](#).

Definition at line 24 of file IAM2WayImportance.h.

6.14.3.5 `template<class T> virtual void IAM2WayImportance< T >::reset () [inline, virtual]`

Reimplemented from [Importance< T >](#).

Definition at line 82 of file IAM2WayImportance.h.

6.14.4 Member Data Documentation

6.14.4.1 `template<class T> std::vector<double> IAM2WayImportance< T >::amounts`

Definition at line 96 of file IAM2WayImportance.h.

6.14.4.2 `template<class T> std::vector<std::pair<double, uli_t> > IAM2WayImportance< T >::freqPairs`

Definition at line 94 of file IAM2WayImportance.h.

6.14.4.3 template<class T> uli_t IAM2WayImportance< T >::idx

Definition at line 92 of file IAM2WayImportance.h.

**6.14.4.4 template<class T> std::vector<std::pair<uli_t, uli_t> >::iterator
IAM2WayImportance< T >::idxIt**

Definition at line 93 of file IAM2WayImportance.h.

**6.14.4.5 template<class T> std::vector<std::pair<uli_t, uli_t> >
IAM2WayImportance< T >::indexer**

Definition at line 95 of file IAM2WayImportance.h.

6.14.4.6 template<class T> uli_t IAM2WayImportance< T >::vecSizeLimit

Definition at line 91 of file IAM2WayImportance.h.

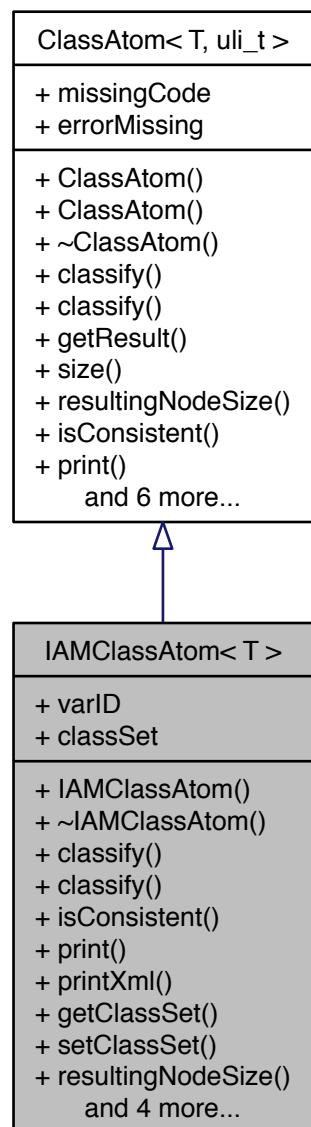
The documentation for this class was generated from the following file:

- src/library/IAM2WayImportance.h

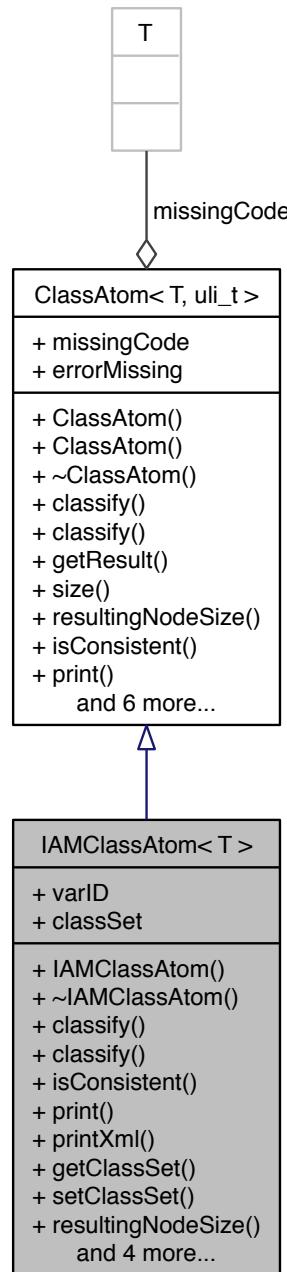
6.15 IAMClassAtom< T > Class Template Reference

```
#include <IAMClassAtom.h>
```

Inheritance diagram for IAMClassAtom< T >:



Collaboration diagram for IAMClassAtom< T >:



Public Member Functions

- **IAMClassAtom ()**
- virtual **~IAMClassAtom (void)**
- virtual **uli_t classify (Tvector &sample) const**
Classifies an input value.
- virtual **uli_t classify (T *sample) const**
Classifies an input value.
- **bool isConsistent () const**
Returns if the classifier is consistent.
- virtual std::ostream & **print (std::ostream &os) const**
Prints the classifier to output stream.
- virtual void **printXml (std::ostream &os) const**
Prints the classifier to output stream in XML format.
- const std::vector< std::vector < T > > & **getClassSet () const**
- void **setClassSet (std::vector< std::vector < T > > &set)**
- virtual **uli_t resultingNodeSize () const**
Returns the size of resulting node.
- **bool operator== (const IAMClassAtom< T > &iAMClassAtom)**
- virtual **uli_t getType () const**
Returns the type of current classifier.
- void **setVarID (std::vector< uli_t > &varID)**
- std::vector< uli_t > **getVarID () const**
- virtual **bool isThereVarID (uli_t varID) const**
Return if classifier has got a specific variable ID.

Public Attributes

- std::vector< **uli_t** > **varID**
- std::vector< std::vector < T > > **classSet**

6.15.1 Detailed Description

template<class T>class IAMClassAtom< T >

Definition at line 23 of file IAMClassAtom.h.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 template<class T> IAMClassAtom< T >::IAMClassAtom () [inline]

Definition at line 25 of file IAMClassAtom.h.

6.15.2.2 `template<class T> virtual IAMClassAtom< T >::~IAMClassAtom(void) [inline, virtual]`

Definition at line 27 of file IAMClassAtom.h.

6.15.3 Member Function Documentation

6.15.3.1 `template<class T> virtual uli_t IAMClassAtom< T >::classify(Tvector & sample) const [inline, virtual]`

Definition at line 33 of file IAMClassAtom.h.

6.15.3.2 `template<class T> virtual uli_t IAMClassAtom< T >::classify(T * vecIn) const [inline, virtual]`

Classifies an input value.

Parameters

<code>vecIn</code>	Sample to be classified
--------------------	-------------------------

Returns

Classification value

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 56 of file IAMClassAtom.h.

6.15.3.3 `template<class T> const std::vector<std::vector<T> >& IAMClassAtom< T >::getClassSet() const [inline]`

Definition at line 104 of file IAMClassAtom.h.

6.15.3.4 `template<class T> virtual uli_t IAMClassAtom< T >::getType() const [inline, virtual]`

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 113 of file IAMClassAtom.h.

```
6.15.3.5 template<class T> std::vector<uli_t> IAMClassAtom< T >::getVarID( )  
const [inline]
```

Definition at line 116 of file IAMClassAtom.h.

```
6.15.3.6 template<class T> bool IAMClassAtom< T >::isConsistent( ) const  
[inline, virtual]
```

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 76 of file IAMClassAtom.h.

```
6.15.3.7 template<class T> virtual bool IAMClassAtom< T >::isThereVarID( uli_t  
varID ) const [inline, virtual]
```

Return if classifier has got a specific variable ID.

Parameters

<i>varID</i>	ID of variable.
--------------	-----------------

Returns

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 118 of file IAMClassAtom.h.

```
6.15.3.8 template<class T> bool IAMClassAtom< T >::operator==( const  
IAMClassAtom< T > & iAMClassAtom ) [inline]
```

Definition at line 109 of file IAMClassAtom.h.

```
6.15.3.9 template<class T> virtual std::ostream& IAMClassAtom< T >::print(   
std::ostream & os ) const [inline, virtual]
```

Prints the classifier to output stream.

Parameters

<i>os</i>	Output stream.
-----------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 80 of file IAMClassAtom.h.

6.15.3.10 template<class T> virtual void IAMClassAtom< T >::printXml (std::ostream & os) const [inline, virtual]

Prints the classifier to output stream in XML format.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 94 of file IAMClassAtom.h.

6.15.3.11 template<class T> virtual uli_t IAMClassAtom< T >::resultingNodeSize () const [inline, virtual]

Returns the size of resulting node.

Returns

Size of resulting node.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 107 of file IAMClassAtom.h.

6.15.3.12 template<class T> void IAMClassAtom< T >::setClassSet (std::vector< std::vector< T > > & set) [inline]

Definition at line 105 of file IAMClassAtom.h.

6.15.3.13 template<class T> void IAMClassAtom< T >::setVarID (std::vector< uli_t > & varID) [inline]

Definition at line 115 of file IAMClassAtom.h.

6.15.4 Member Data Documentation

6.15.4.1 template<class T> std::vector<std::vector<T>> IAMClassAtom< T >::classSet

Definition at line 126 of file IAMClassAtom.h.

6.15.4.2 template<class T> std::vector<uli_t> IAMClassAtom< T >::varID

Definition at line 125 of file IAMClassAtom.h.

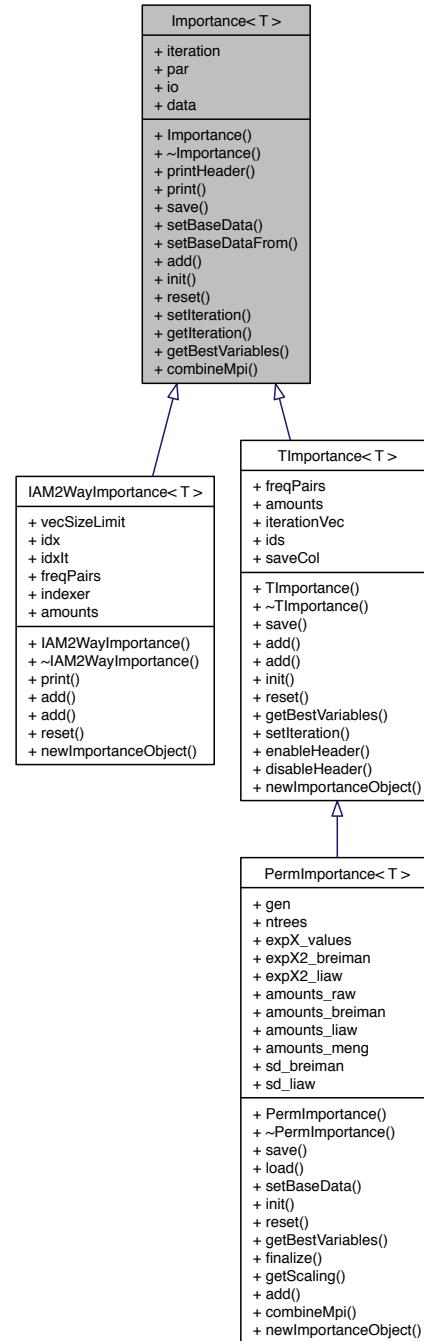
The documentation for this class was generated from the following file:

- [src/library/IAMClassAtom.h](#)

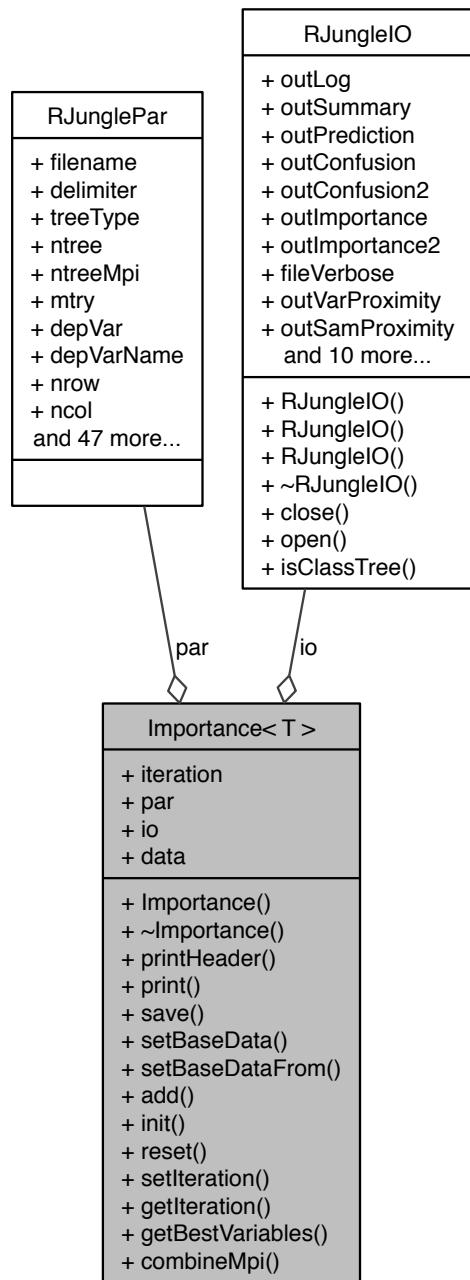
6.16 Importance< T > Class Template Reference

```
#include <Importance.h>
```

Inheritance diagram for Importance< T >:



Collaboration diagram for Importance< T >:



Public Member Functions

- `Importance ()`
- virtual `~Importance ()`
- virtual void `printHeader ()`
- virtual void `print ()`
- virtual void `save ()`
- void `setBaseData (RJunglePar *par, RJungleIO *io, DataFrame< T > *data)`
- void `setBaseDataFrom (Importance< T > &imp)`
- virtual void `add (Importance< T > *imp)`
- virtual void `init ()`
- virtual void `reset ()`
- virtual void `setIteration (uli_t iteration)`
- virtual `uli_t getIteration ()`
- virtual void `getBestVariables (size_t size, std::vector< uli_t > &outVec)`
- virtual void `combineMpi ()`

Public Attributes

- `uli_t iteration`
- `RJunglePar * par`
- `RJungleIO * io`
- `DataFrame< T > * data`

6.16.1 Detailed Description

`template<class T>class Importance< T >`

Definition at line 19 of file `Importance.h`.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 `template<class T> Importance< T >::Importance () [inline]`

Definition at line 21 of file `Importance.h`.

6.16.2.2 `template<class T> virtual Importance< T >::~Importance () [inline, virtual]`

Definition at line 23 of file `Importance.h`.

6.16.3 Member Function Documentation

6.16.3.1 `template<class T> virtual void Importance< T >::add (Importance< T > * imp) [inline, virtual]`

Reimplemented in [IAM2WayImportance< T >](#), and [TImportance< T >](#).

Definition at line 41 of file Importance.h.

6.16.3.2 `template<class T> virtual void Importance< T >::combineMpi () [inline, virtual]`

Definition at line 50 of file Importance.h.

6.16.3.3 `template<class T> virtual void Importance< T >::getBestVariables (size_t size, std::vector< uli_t > & outVec) [inline, virtual]`

Reimplemented in [PermlImportance< T >](#), and [TImportance< T >](#).

Definition at line 49 of file Importance.h.

6.16.3.4 `template<class T> virtual uli_t Importance< T >::getIteration () [inline, virtual]`

Definition at line 47 of file Importance.h.

6.16.3.5 `template<class T> virtual void Importance< T >::init () [inline, virtual]`

Reimplemented in [PermlImportance< T >](#), and [TImportance< T >](#).

Definition at line 43 of file Importance.h.

6.16.3.6 `template<class T> virtual void Importance< T >::print () [inline, virtual]`

Reimplemented in [IAM2WayImportance< T >](#).

Definition at line 26 of file Importance.h.

6.16.3.7 `template<class T> virtual void Importance< T >::printHeader () [inline, virtual]`

Definition at line 25 of file Importance.h.

6.16.3.8 template<class T> virtual void Importance< T >::reset() [inline, virtual]

Reimplemented in [PermlImportance< T >](#), [IAM2WayImportance< T >](#), and [T-Importance< T >](#).

Definition at line 44 of file [Importance.h](#).

6.16.3.9 template<class T> virtual void Importance< T >::save() [inline, virtual]

Reimplemented in [PermlImportance< T >](#), and [TImportance< T >](#).

Definition at line 27 of file [Importance.h](#).

6.16.3.10 template<class T> void Importance< T >::setBaseData(RJunglePar * par, RJungleIO * io, DataFrame< T > * data) [inline]

Definition at line 29 of file [Importance.h](#).

6.16.3.11 template<class T> void Importance< T >::setBaseDataFrom(Importance< T > & imp) [inline]

Definition at line 35 of file [Importance.h](#).

6.16.3.12 template<class T> virtual void Importance< T >::setIteration(uli_t iteration) [inline, virtual]

Reimplemented in [TImportance< T >](#).

Definition at line 46 of file [Importance.h](#).

6.16.4 Member Data Documentation

6.16.4.1 template<class T> DataFrame<T>*>* Importance< T >::data

Definition at line 55 of file [Importance.h](#).

6.16.4.2 template<class T> RJungleIO* Importance< T >::io

Definition at line 54 of file [Importance.h](#).

6.16.4.3 template<class T> uli_t Importance< T >::iteration

Definition at line 50 of file [Importance.h](#).

6.16.4.4 template<class T> RJunglePar* Importance< T >::par

Definition at line 53 of file Importance.h.

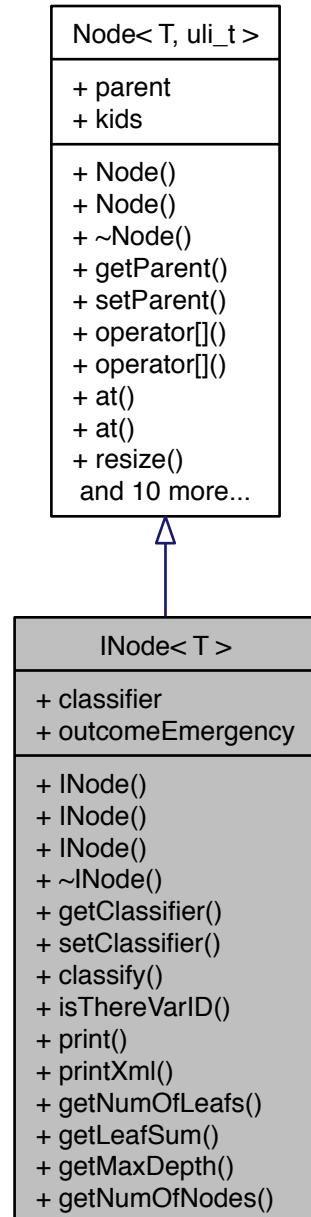
The documentation for this class was generated from the following file:

- [src/library/Importance.h](#)

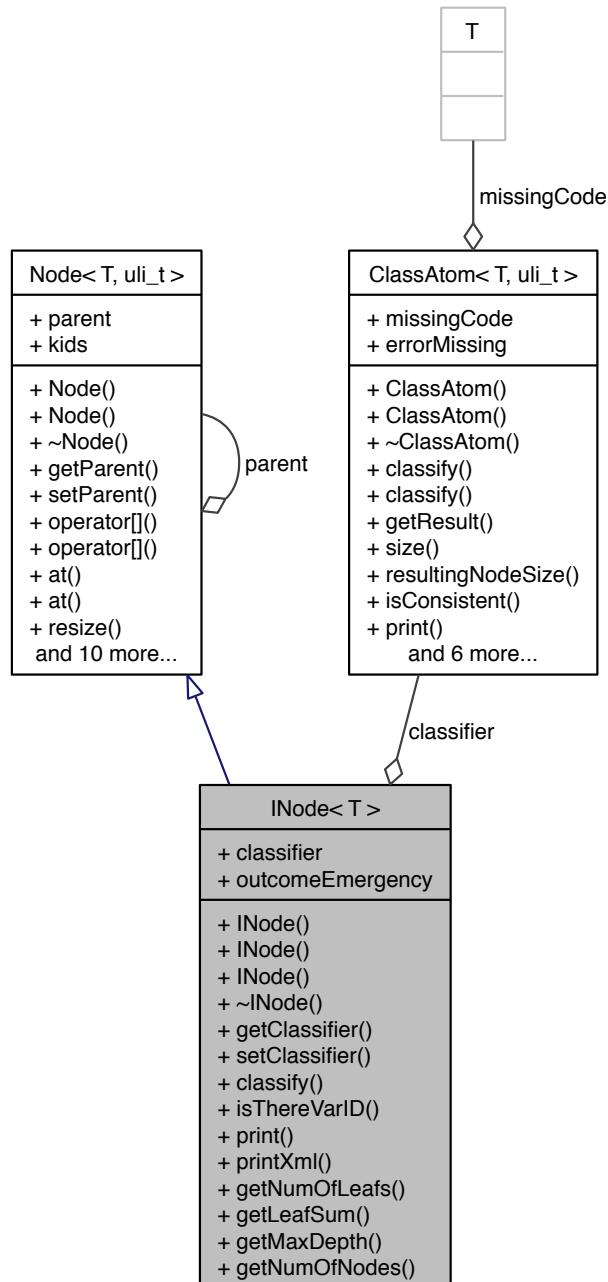
6.17 INode< T > Class Template Reference

```
#include <INode.h>
```

Inheritance diagram for INode< T >:



Collaboration diagram for INode< T >:



Public Member Functions

- `INode ()`
- `INode (ClassAtom< T, uli_t > *classifier, uli_t outcomeEmergency=std::numeric_limits< uli_t >::max())`
- `INode (const INode< T > &iNode)`
- virtual `~INode ()`
- `ClassAtom< T, uli_t > * getClassifier () const`
- `void setClassifier (ClassAtom< T, uli_t > *classifier)`
- virtual `T classify (const std::vector< T > &sample) const`
- virtual `bool isThereVarID (uli_t varID) const`
- virtual `std::ostream & print (std::ostream &os) const`
- virtual `void printXml (std::ostream &os) const`
- virtual `uli_t getNumOfLeafs ()`
- virtual `double getLeafSum ()`
- virtual `uli_t getMaxDepth ()`
- virtual `uli_t getNumOfNodes ()`

Public Attributes

- `ClassAtom< T, uli_t > * classifier`
- `uli_t outcomeEmergency`

6.17.1 Detailed Description

`template<class T>class INode< T >`

Definition at line 27 of file INode.h.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 `template<class T> INode< T >::INode () [inline]`

Definition at line 29 of file INode.h.

6.17.2.2 `template<class T> INode< T >::INode (ClassAtom< T, uli_t > * classifier, uli_t outcomeEmergency = std::numeric_limits<uli_t>::max()) [inline]`

Definition at line 32 of file INode.h.

6.17.2.3 `template<class T> INode< T >::INode (const INode< T > & iNode) [inline]`

Definition at line 40 of file INode.h.

6.17.2.4 `template<class T> virtual INode< T >::~INode() [inline, virtual]`

Definition at line 43 of file INode.h.

6.17.3 Member Function Documentation

6.17.3.1 `template<class T> virtual T INode< T >::classify(const std::vector< T > & sample) const [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 55 of file INode.h.

6.17.3.2 `template<class T> ClassAtom< T, uli_t >* INode< T >::getClassifier() const [inline]`

Definition at line 47 of file INode.h.

6.17.3.3 `template<class T> virtual double INode< T >::getLeafSum() [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 173 of file INode.h.

6.17.3.4 `template<class T> virtual uli_t INode< T >::getMaxDepth() [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 189 of file INode.h.

6.17.3.5 `template<class T> virtual uli_t INode< T >::getNumOfLeafs() [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 155 of file INode.h.

6.17.3.6 `template<class T> virtual uli_t INode< T >::getNumOfNodes() [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 193 of file INode.h.

6.17.3.7 `template<class T> virtual bool INode< T >::isThereVarID (uli_t varID) const [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 85 of file INode.h.

6.17.3.8 `template<class T> virtual std::ostream& INode< T >::print (std::ostream & os) const [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 100 of file INode.h.

6.17.3.9 `template<class T> virtual void INode< T >::printXml (std::ostream & os) const [inline, virtual]`

Reimplemented from [Node< T, uli_t >](#).

Definition at line 129 of file INode.h.

6.17.3.10 `template<class T> void INode< T >::setClassifier (ClassAtom< T, uli_t > * classifier) [inline]`

Definition at line 51 of file INode.h.

6.17.4 Member Data Documentation

6.17.4.1 `template<class T> ClassAtom< T, uli_t >* INode< T >::classifier`

Definition at line 210 of file INode.h.

6.17.4.2 `template<class T> uli_t INode< T >::outcomeEmergency`

Definition at line 211 of file INode.h.

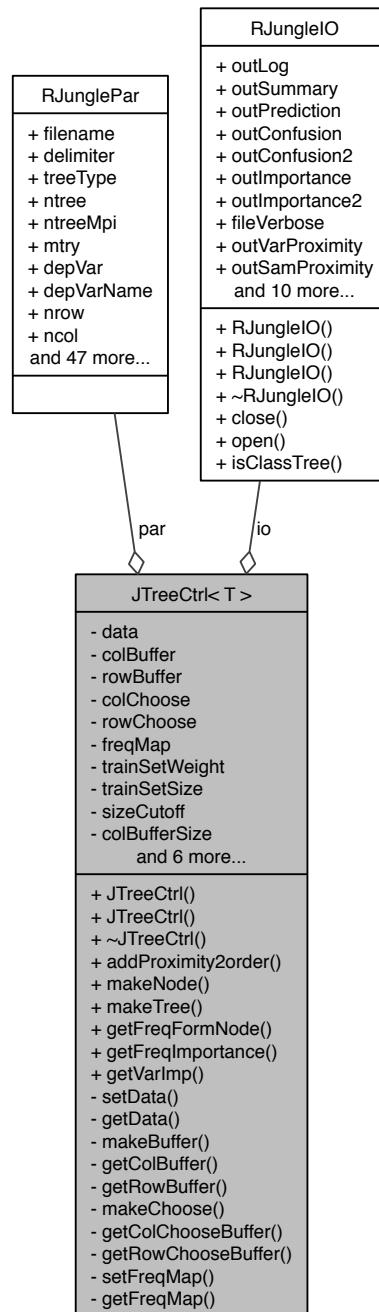
The documentation for this class was generated from the following file:

- [src/library/INode.h](#)

6.18 JTreeCtrl< T > Class Template Reference

```
#include <JTreeCtrl.h>
```

Collaboration diagram for JTreeCtrl< T >:



Public Member Functions

- `JTreeCtrl ()`
- `JTreeCtrl (RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen)`
- `virtual ~JTreeCtrl ()`
- `void addProximity2order (Node< T, uli_t > *node, DataFrame< double > *varProxiSum, std::vector< uli_t > &idxVec)`
- `virtual INode< T > * makeNode (DataFrame< T > &data, RJunglePar &par, std::vector< uli_t > &rowMaskVec, std::vector< double > &trainSetWeight, uli_t depth=0, INode< T > *parent=NULL, double lastImprovement=0)`
- `virtual Tree< T, uli_t > * makeTree (DataFrame< T > *data, std::vector< uli_t > *colMaskVec, uli_t treelid, bool downsampling_flag, std::vector< uli_t > &trainDataRows, DataTreeSet &oobSet, std::vector< uli_t > &oobDataRows, - DataFrame< double > *yaimp_=NULL)`
- `void getFreqFormNode (INode< T > *inode, std::map< uli_t, double > *freqMap)`
- `void getFreqImportance (Tree< T, uli_t > *tree, DataFrame< T > *data, std::map< uli_t, double > *freqMap)`
- `Importance< T > * getVarImp ()`

Private Member Functions

- `void setData (DataFrame< T > *data, std::vector< uli_t > *idxVec=NULL)`
- `DataFrame< T > * getData () const`
- `void makeBuffer (std::vector< uli_t > *idxVec)`
- `uli_t * getColBuffer () const`
- `uli_t * getRowBuffer () const`
- `void makeChoose ()`
- `uli_t * getColChooseBuffer () const`
- `uli_t * getRowChooseBuffer () const`
- `void setFreqMap (std::map< uli_t, T > *freqMap)`
- `std::map< uli_t, T > * getFreqMap () const`

Private Attributes

- `DataFrame< T > * data`
- `uli_t * colBuffer`
- `uli_t * rowBuffer`
- `uli_t * colChoose`
- `uli_t * rowChoose`
- `std::map< uli_t, T > * freqMap`
- `std::vector< double > trainSetWeight`
- `uli_t trainSetSize`
- `uli_t sizeCutoff`
- `uli_t colBufferSize`
- `Importance< T > * importance`

- std::vector< double > varImp
- std::vector< std::pair< double, std::pair< uli_t, uli_t > > > twoWayInteraction
- FittingFctPar< T > * fitFctPar
- RJunglePar par
- RJungleIO io
- RJungleGen< T > gen

6.18.1 Detailed Description

```
template<class T>class JTreeCtrl< T >
```

Definition at line 41 of file JTreeCtrl.h.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 template<class T> JTreeCtrl< T >::JTreeCtrl() [inline]

Definition at line 43 of file JTreeCtrl.h.

6.18.2.2 template<class T> JTreeCtrl< T >::JTreeCtrl(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen) [inline]

Definition at line 55 of file JTreeCtrl.h.

6.18.2.3 template<class T> virtual JTreeCtrl< T >::~JTreeCtrl() [inline, virtual]

Definition at line 67 of file JTreeCtrl.h.

6.18.3 Member Function Documentation

6.18.3.1 template<class T> void JTreeCtrl< T >::addProximity2order(Node< T, uli_t > * node, DataFrame< double > * varProxiSum, std::vector< uli_t > & idxVec) [inline]

Definition at line 77 of file JTreeCtrl.h.

6.18.3.2 template<class T> uli_t* JTreeCtrl< T >::getColBuffer() const [inline, private]

Definition at line 485 of file JTreeCtrl.h.

```
6.18.3.3 template<class T> uli_t* JTreeCtrl< T >::getColChooseBuffer( ) const  
[inline, private]
```

Definition at line 505 of file JTreeCtrl.h.

```
6.18.3.4 template<class T> DataFrame<T>* JTreeCtrl< T >::getData( ) const  
[inline, private]
```

Definition at line 439 of file JTreeCtrl.h.

```
6.18.3.5 template<class T> void JTreeCtrl< T >::getFreqFormNode( INode< T > *  
inode, std::map< uli_t, double > * freqMap ) [inline]
```

Definition at line 398 of file JTreeCtrl.h.

```
6.18.3.6 template<class T> void JTreeCtrl< T >::getFreqImportance( Tree< T, uli_t  
> * tree, DataFrame< T > * data, std::map< uli_t, double > * freqMap )  
[inline]
```

Definition at line 412 of file JTreeCtrl.h.

```
6.18.3.7 template<class T> std::map<uli_t, T>* JTreeCtrl< T >::getFreqMap( ) const  
[inline, private]
```

Definition at line 519 of file JTreeCtrl.h.

```
6.18.3.8 template<class T> uli_t* JTreeCtrl< T >::getRowBuffer( ) const  
[inline, private]
```

Definition at line 489 of file JTreeCtrl.h.

```
6.18.3.9 template<class T> uli_t* JTreeCtrl< T >::getRowChooseBuffer( ) const  
[inline, private]
```

Definition at line 509 of file JTreeCtrl.h.

```
6.18.3.10 template<class T> Importance<T>* JTreeCtrl< T >::getVarImp( )  
[inline]
```

Definition at line 426 of file JTreeCtrl.h.

```
6.18.3.11 template<class T> void JTreeCtrl< T >::makeBuffer ( std::vector< uli_t > *  
idxVec ) [inline, private]
```

Definition at line 445 of file JTreeCtrl.h.

```
6.18.3.12 template<class T> void JTreeCtrl< T >::makeChoose ( ) [inline,  
private]
```

Definition at line 496 of file JTreeCtrl.h.

```
6.18.3.13 template<class T> virtual INode< T >* JTreeCtrl< T >::makeNode  
( DataFrame< T > & data, RJunglePar & par, std::vector< uli_t > &  
rowMaskVec, std::vector< double > & trainSetWeight, uli_t depth = 0, INode< T  
> * parent = NULL, double lastImprovement = 0 ) [inline, virtual]
```

Definition at line 97 of file JTreeCtrl.h.

```
6.18.3.14 template<class T> virtual Tree< T, uli_t >* JTreeCtrl< T >::makeTree (   
DataFrame< T > * data, std::vector< uli_t > * colMaskVec, uli_t treeld, bool  
downsampling_flag, std::vector< uli_t > & trainDataRows, DataTreeSet & oobSet,  
std::vector< uli_t > & oobDataRows, DataFrame< double > * yaimp_ = NULL )  
[inline, virtual]
```

Definition at line 252 of file JTreeCtrl.h.

```
6.18.3.15 template<class T> void JTreeCtrl< T >::setData ( DataFrame< T > * data,  
std::vector< uli_t > * idxVec = NULL ) [inline, private]
```

Definition at line 432 of file JTreeCtrl.h.

```
6.18.3.16 template<class T> void JTreeCtrl< T >::setFreqMap ( std::map< uli_t, T > *  
freqMap ) [inline, private]
```

Definition at line 516 of file JTreeCtrl.h.

6.18.4 Member Data Documentation

```
6.18.4.1 template<class T> uli_t* JTreeCtrl< T >::colBuffer [private]
```

Definition at line 492 of file JTreeCtrl.h.

```
6.18.4.2 template<class T> uli_t JTreeCtrl< T >::colBufferSize [private]
```

Definition at line 543 of file JTreeCtrl.h.

6.18.4.3 `template<class T> uli_t* JTreeCtrl< T >::colChoose [private]`

Definition at line 512 of file JTreeCtrl.h.

6.18.4.4 `template<class T> DataFrame<T>* JTreeCtrl< T >::data [private]`

Definition at line 442 of file JTreeCtrl.h.

6.18.4.5 `template<class T> FittingFctPar<T>* JTreeCtrl< T >::fitFctPar [private]`

Definition at line 550 of file JTreeCtrl.h.

6.18.4.6 `template<class T> std::map<uli_t, T>* JTreeCtrl< T >::freqMap [private]`

Definition at line 522 of file JTreeCtrl.h.

6.18.4.7 `template<class T> RJungleGen<T> JTreeCtrl< T >::gen [private]`

Definition at line 554 of file JTreeCtrl.h.

6.18.4.8 `template<class T> Importance<T>* JTreeCtrl< T >::importance [private]`

Definition at line 545 of file JTreeCtrl.h.

6.18.4.9 `template<class T> RJungleIO JTreeCtrl< T >::io [private]`

Definition at line 553 of file JTreeCtrl.h.

6.18.4.10 `template<class T> RJunglePar JTreeCtrl< T >::par [private]`

Definition at line 552 of file JTreeCtrl.h.

6.18.4.11 `template<class T> uli_t* JTreeCtrl< T >::rowBuffer [private]`

Definition at line 494 of file JTreeCtrl.h.

6.18.4.12 `template<class T> uli_t* JTreeCtrl< T >::rowChoose [private]`

Definition at line 514 of file JTreeCtrl.h.

6.18.4.13 template<class T> uli_t JTreeCtrl< T >::sizeCutoff [private]

Definition at line 542 of file JTreeCtrl.h.

6.18.4.14 template<class T> uli_t JTreeCtrl< T >::trainSetSize [private]

Definition at line 541 of file JTreeCtrl.h.

6.18.4.15 template<class T> std::vector<double> JTreeCtrl< T >::trainSetWeight
[private]

Definition at line 540 of file JTreeCtrl.h.

6.18.4.16 template<class T> std::vector<std::pair<double, std::pair<uli_t, uli_t>>>
JTreeCtrl< T >::twoWayInteraction [private]

Definition at line 547 of file JTreeCtrl.h.

6.18.4.17 template<class T> std::vector<double> JTreeCtrl< T >::varImp
[private]

Definition at line 546 of file JTreeCtrl.h.

The documentation for this class was generated from the following file:

- [src/library/JTreeCtrl.h](#)

6.19 JTreeTweaks Struct Reference

```
#include <JTreeTweaks.h>
```

Public Attributes

- [uli_t maxTreeDepth](#)
- [uli_t targetPartitionSize](#)

6.19.1 Detailed Description

Definition at line 18 of file JTreeTweaks.h.

6.19.2 Member Data Documentation

6.19.2.1 `uli_t JTreeTweaks::maxTreeDepth`

Definition at line 19 of file JTreeTweaks.h.

6.19.2.2 `uli_t JTreeTweaks::targetPartitionSize`

Definition at line 20 of file JTreeTweaks.h.

The documentation for this struct was generated from the following file:

- `src/library/JTreeTweaks.h`

6.20 Logistic Class Reference

```
#include <Logistic.h>
```

Static Public Member Functions

- static int `logistic` (`gsl_vector *z, gsl_vector *b, double &val`)
- static void `logisticreg_iter_2` (`gsl_matrix *X, double *resp, gsl_vector *b, gsl_matrix *inf, double *diff`)
- static int `comp_w_z` (`gsl_vector *X, double *y, gsl_vector *b, gsl_vector *w, gsl_vector *z`)
- static int `gsl_vector_sum` (`gsl_vector *w, double &sum`)
- static int `left_side_matrix` (`gsl_matrix *inf, gsl_vector *X, gsl_vector *w`)
- static int `right_side_vector` (`gsl_vector *rs, gsl_vector *X, gsl_vector *w, gsl_vector *z`)
- static int `solve_sym` (`gsl_matrix *A, gsl_vector *b, gsl_vector *x, gsl_matrix *A_inv`)
- static int `logistic_reg_iter` (`gsl_vector *X, double *resp, gsl_vector *b, gsl_matrix *cov, double *diff`)
- static void `logistic_reg` (`gsl_vector *X, double *resp, gsl_vector *b, gsl_matrix *cov, double epsilon, bool &isNotConverging`)

6.20.1 Detailed Description

Definition at line 9 of file Logistic.h.

6.20.2 Member Function Documentation

6.20.2.1 `int Logistic::comp_w_z(gsl_vector * X, double * y, gsl_vector * b, gsl_vector * w, gsl_vector * z) [static]`

Definition at line 68 of file Logistic.cpp.

6.20.2.2 `int Logistic::gsl_vector_sum(gsl_vector * w, double & sum) [static]`

Definition at line 125 of file Logistic.cpp.

6.20.2.3 `int Logistic::left_side_matrix(gsl_matrix * inf, gsl_vector * X, gsl_vector * w) [static]`

Definition at line 149 of file Logistic.cpp.

6.20.2.4 `int Logistic::logistic(gsl_vector * z, gsl_vector * b, double & val) [static]`

Definition at line 33 of file Logistic.cpp.

6.20.2.5 `void Logistic::logistic_reg(gsl_vector * X, double * resp, gsl_vector * b, gsl_matrix * cov, double epsilon, bool & isNotConverging) [static]`

Definition at line 407 of file Logistic.cpp.

6.20.2.6 `int Logistic::logistic_reg_iter(gsl_vector * X, double * resp, gsl_vector * b, gsl_matrix * cov, double * diff) [static]`

Definition at line 271 of file Logistic.cpp.

6.20.2.7 `static void Logistic::logisticreg_iter_2(gsl_matrix * X, double * resp, gsl_vector * b, gsl_matrix * inf, double * diff) [static]`

6.20.2.8 `int Logistic::right_side_vector(gsl_vector * rs, gsl_vector * X, gsl_vector * w, gsl_vector * z) [static]`

Definition at line 199 of file Logistic.cpp.

6.20.2.9 `int Logistic::solve_sym(gsl_matrix * A, gsl_vector * b, gsl_vector * x, gsl_matrix * A_inv) [static]`

Definition at line 240 of file Logistic.cpp.

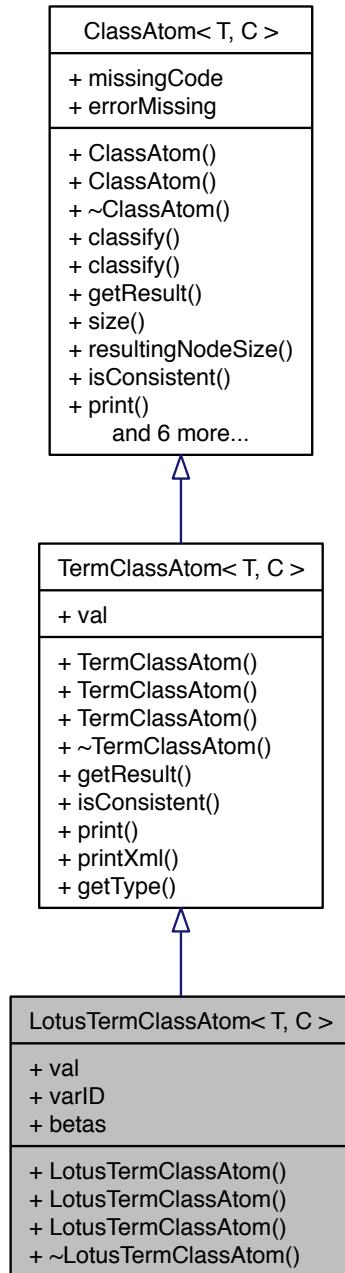
The documentation for this class was generated from the following files:

- [src/library/Logistic.h](#)
- [src/library/Logistic.cpp](#)

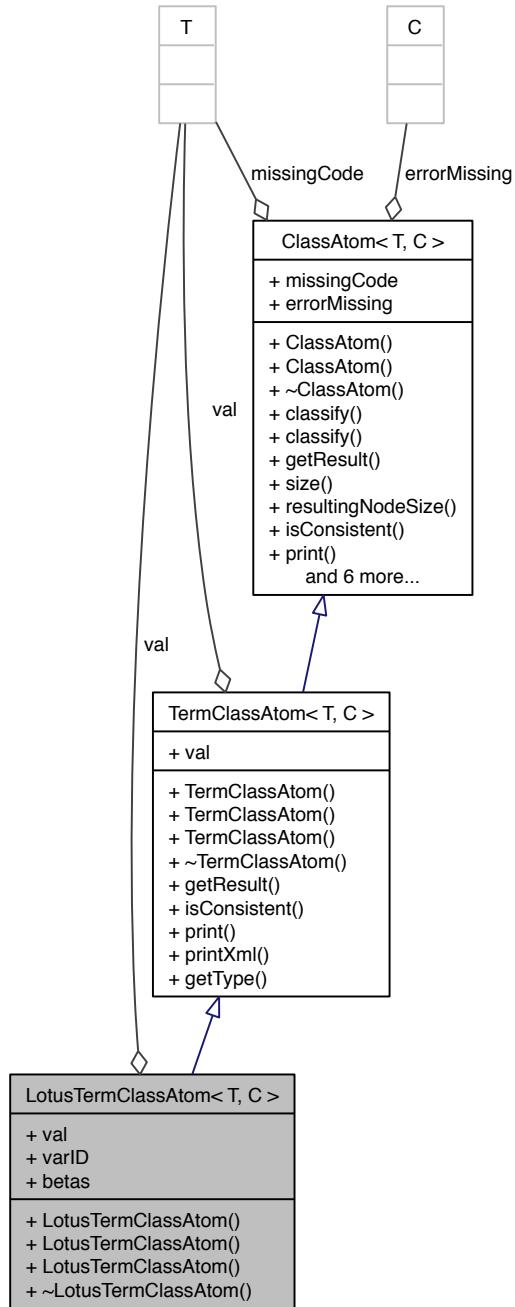
6.21 LotusTermClassAtom< T, C > Class Template Reference

```
#include <LotusTermClassAtom.h>
```

Inheritance diagram for LotusTermClassAtom< T, C >:



Collaboration diagram for `LotusTermClassAtom< T, C >`:



Public Member Functions

- `LotusTermClassAtom ()`
- `LotusTermClassAtom (const LotusTermClassAtom< T, C > &termCA)`
- `LotusTermClassAtom (T val)`
- `virtual ~LotusTermClassAtom (void)`

Public Attributes

- `T val`
`waste`
- `uli_t varID`
`x predictor variable`
- `std::vector< double > betas`
`Betas of logistic regression.`

6.21.1 Detailed Description

`template<class T, class C>class LotusTermClassAtom< T, C >`

Definition at line 11 of file LotusTermClassAtom.h.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 `template<class T, class C> LotusTermClassAtom< T, C >::LotusTermClassAtom() [inline]`

Definition at line 13 of file LotusTermClassAtom.h.

6.21.2.2 `template<class T, class C> LotusTermClassAtom< T, C >::LotusTermClassAtom (const LotusTermClassAtom< T, C > & termCA) [inline]`

Definition at line 14 of file LotusTermClassAtom.h.

6.21.2.3 `template<class T, class C> LotusTermClassAtom< T, C >::LotusTermClassAtom (T val) [inline]`

Definition at line 15 of file LotusTermClassAtom.h.

6.21.2.4 `template<class T, class C> virtual LotusTermClassAtom< T, C >::~LotusTermClassAtom (void) [inline, virtual]`

Definition at line 17 of file LotusTermClassAtom.h.

6.21.3 Member Data Documentation

6.21.3.1 `template<class T, class C> std::vector<double> LotusTermClassAtom< T, C >::betas`

Betas of logistic regression.

Definition at line 27 of file `LotusTermClassAtom.h`.

6.21.3.2 `template<class T, class C> T LotusTermClassAtom< T, C >::val`

waste

Reimplemented from `TermClassAtom< T, C >`.

Definition at line 18 of file `LotusTermClassAtom.h`.

6.21.3.3 `template<class T, class C> uli_t LotusTermClassAtom< T, C >::varID`

x predictor variable

Definition at line 24 of file `LotusTermClassAtom.h`.

The documentation for this class was generated from the following file:

- [src/library/LotusTermClassAtom.h](#)

6.22 Node< T, C > Class Template Reference

```
#include <Node.h>
```

Public Member Functions

- [Node \(\)](#)
- [Node \(const Node< T, C > &node\)](#)
- [virtual ~Node \(\)](#)
- [const Node< T, C > * getParent \(\) const](#)
- [void setParent \(Node< T, C > *parent\)](#)
- [Node< T, C > * operator\[\] \(uli_t i\) const](#)
- [Node< T, C > * operator\[\] \(uli_t i\)](#)
- [Node< T, C > * at \(uli_t i\) const](#)
- [Node< T, C > * at \(uli_t i\)](#)
- [void resize \(uli_t size\)](#)
- [void push_back \(Node< T, C > *node\)](#)
- [uli_t size \(\)](#)
- [virtual uli_t getNumOfLeafs \(\)](#)
- [virtual double getLeafSum \(\)](#)

- virtual `uli_t getMaxDepth ()`
- virtual `uli_t getNumOfNodes ()`
- virtual `C getPath (const T &val) const`
- virtual `T classify (const std::vector< T > &sample) const`
- virtual `std::ostream & print (std::ostream &os) const`
- virtual `bool isThereVarID (uli_t varID) const`
- virtual `void printXml (std::ostream &os) const`

Public Attributes

- `Node< T, C > * parent`
- `TCNodeVector kids`

6.22.1 Detailed Description

`template<class T, class C>class Node< T, C >`

Definition at line 18 of file Node.h.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 `template<class T, class C> Node< T, C >::Node () [inline]`

Definition at line 20 of file Node.h.

6.22.2.2 `template<class T, class C> Node< T, C >::Node (const Node< T, C > & node) [inline]`

Definition at line 23 of file Node.h.

6.22.2.3 `template<class T, class C> virtual Node< T, C >::~Node () [inline, virtual]`

Definition at line 25 of file Node.h.

6.22.3 Member Function Documentation

6.22.3.1 `template<class T, class C> Node< T, C >*> Node< T, C >::at (uli_t i) const [inline]`

Definition at line 41 of file Node.h.

6.22.3.2 **template<class T, class C> Node<T, C>*> Node<T, C>::at (uli_t i)**
[inline]

Definition at line 42 of file Node.h.

6.22.3.3 **template<class T, class C> virtual T Node<T, C>::classify (const std::vector<T> & sample) const** [inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 63 of file Node.h.

6.22.3.4 **template<class T, class C> virtual double Node<T, C>::getLeafSum ()**
[inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 52 of file Node.h.

6.22.3.5 **template<class T, class C> virtual uli_t Node<T, C>::getMaxDepth ()**
[inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 53 of file Node.h.

6.22.3.6 **template<class T, class C> virtual uli_t Node<T, C>::getNumOfLeafs ()**
[inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 51 of file Node.h.

6.22.3.7 **template<class T, class C> virtual uli_t Node<T, C>::getNumOfNodes ()**
[inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 54 of file Node.h.

6.22.3.8 **template<class T, class C> const Node<T, C>*> Node<T, C>::getParent ()**
const [inline]

Definition at line 35 of file Node.h.

6.22.3.9 template<class T, class C> virtual C Node< T, C >::getPath (const T & val) const
[inline, virtual]

Definition at line 59 of file Node.h.

6.22.3.10 template<class T, class C> virtual bool Node< T, C >::isThereVarID (uli_t
varID) const [inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 71 of file Node.h.

6.22.3.11 template<class T, class C> Node< T, C >* Node< T, C >::operator[] (uli_t i)
const [inline]

Definition at line 38 of file Node.h.

6.22.3.12 template<class T, class C> Node< T, C >* Node< T, C >::operator[] (uli_t i)
[inline]

Definition at line 39 of file Node.h.

6.22.3.13 template<class T, class C> virtual std::ostream& Node< T, C >::print (std::ostream &
os) const [inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 67 of file Node.h.

6.22.3.14 template<class T, class C> virtual void Node< T, C >::printXml (std::ostream &
os) const [inline, virtual]

Reimplemented in [INode< T >](#).

Definition at line 73 of file Node.h.

6.22.3.15 template<class T, class C> void Node< T, C >::push_back (Node< T, C > *
node) [inline]

Definition at line 47 of file Node.h.

6.22.3.16 template<class T, class C> void Node< T, C >::resize (uli_t size)
[inline]

Definition at line 45 of file Node.h.

6.22.3.17 template<class T, class C> void Node< T, C >::setParent (Node< T, C > * *parent*) [inline]

Definition at line 36 of file Node.h.

6.22.3.18 template<class T, class C> uli_t Node< T, C >::size () [inline]

Definition at line 49 of file Node.h.

6.22.4 Member Data Documentation

6.22.4.1 template<class T, class C> TCNodeVector Node< T, C >::kids

Definition at line 77 of file Node.h.

6.22.4.2 template<class T, class C> Node< T, C >* Node< T, C >::parent

Definition at line 76 of file Node.h.

The documentation for this class was generated from the following file:

- src/library/[Node.h](#)

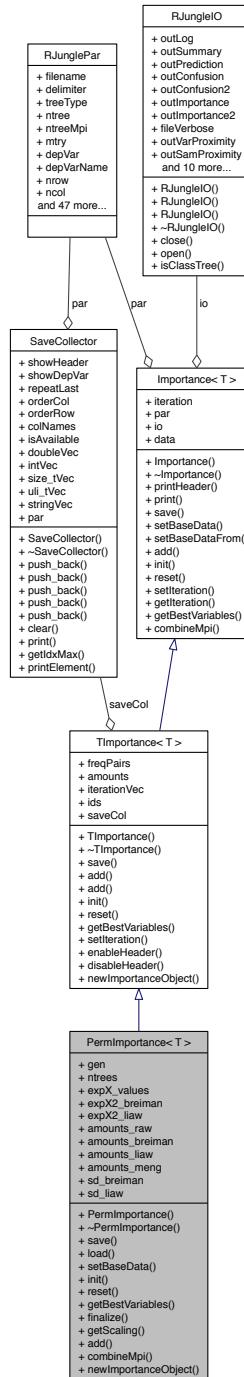
6.23 PermImportance< T > Class Template Reference

```
#include <PermImportance.h>
```

Inheritance diagram for PermlImportance< T >:



Collaboration diagram for PermlImportance< T >:



Public Member Functions

- `PermlImportance ()`
- `virtual ~PermlImportance ()`
- `virtual void save ()`
- `virtual void load ()`
- `void setBaseData (RJunglePar *par, RJungleIO *io, RJungleGen< T > *gen, DataFrame< T > *data)`
- `virtual void init ()`
- `virtual void reset ()`
- `virtual void getBestVariables (size_t size, std::vector< uli_t > &outVec)`
- `void finalize ()`
- `double getScaling (double accur, double variation)`
- `void add (CmpldTree< T > *cmpldTree, DataTreeSet &oobSet, long int treeld, Prediction< T > &predOrg, std::vector< uli_t > *&colMaskVec)`
- `virtual void combineMpi (std::vector< uli_t > *&colMaskVec)`

Static Public Member Functions

- `static Importance< T > * newImportanceObject ()`

Public Attributes

- `RJungleGen< T > * gen`
- `std::vector< double > ntrees`
- `std::vector< double > expX_values`
- `std::vector< double > expX2_breiman`
- `std::vector< double > expX2_liaw`
- `std::vector< double > amounts_raw`
- `std::vector< double > amounts_breiman`
- `std::vector< double > amounts_liaw`
- `std::vector< double > amounts_meng`
- `std::vector< double > sd_breiman`
- `std::vector< double > sd_liaw`

6.23.1 Detailed Description

`template<class T>class PermlImportance< T >`

Definition at line 27 of file PermlImportance.h.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 `template<class T> PermlImportance< T >::PermlImportance() [inline]`

Definition at line 29 of file PermlImportance.h.

6.23.2.2 `template<class T> virtual PermlImportance< T >::~PermlImportance() [inline, virtual]`

Definition at line 34 of file PermlImportance.h.

6.23.3 Member Function Documentation

6.23.3.1 `template<class T> void PermlImportance< T >::add(CmpldTree< T > * cmpldTree, DataTreeSet & oobSet, long int treeld, Prediction< T > & predOrg, std::vector< uli_t > *& colMaskVec) [inline]`

Definition at line 211 of file PermlImportance.h.

6.23.3.2 `template<class T> virtual void PermlImportance< T >::combineMpi(std::vector< uli_t > *& colMaskVec) [inline, virtual]`

Definition at line 395 of file PermlImportance.h.

6.23.3.3 `template<class T> void PermlImportance< T >::finalize() [inline]`

Definition at line 137 of file PermlImportance.h.

6.23.3.4 `template<class T> virtual void PermlImportance< T >::getBestVariables(size_t size, std::vector< uli_t > & outVec) [inline, virtual]`

Reimplemented from [TImportance< T >](#).

Definition at line 111 of file PermlImportance.h.

6.23.3.5 `template<class T> double PermlImportance< T >::getScaling(double accur, double variation) [inline]`

Definition at line 189 of file PermlImportance.h.

6.23.3.6 `template<class T> virtual void PermlImportance< T >::init() [inline, virtual]`

Reimplemented from [TImportance< T >](#).

Definition at line 59 of file PermlImportance.h.

6.23.3.7 `template<class T> virtual void PermlImportance< T >::load() [inline, virtual]`

Definition at line 47 of file PermlImportance.h.

6.23.3.8 `template<class T> static Importance<T>* PermlImportance< T >::newImportanceObject() [inline, static]`

Reimplemented from [TImportance< T >](#).

Definition at line 107 of file PermlImportance.h.

6.23.3.9 `template<class T> virtual void PermlImportance< T >::reset() [inline, virtual]`

Reimplemented from [TImportance< T >](#).

Definition at line 87 of file PermlImportance.h.

6.23.3.10 `template<class T> virtual void PermlImportance< T >::save() [inline, virtual]`

Reimplemented from [TImportance< T >](#).

Definition at line 38 of file PermlImportance.h.

6.23.3.11 `template<class T> void PermlImportance< T >::setBaseData(RJunglePar * par, RJungleIO * io, RJungleGen< T > * gen, DataFrame< T > * data) [inline]`

Definition at line 51 of file PermlImportance.h.

6.23.4 Member Data Documentation

6.23.4.1 `template<class T> std::vector<double> PermlImportance< T >::amounts_breiman`

Definition at line 512 of file PermlImportance.h.

6.23.4.2 `template<class T> std::vector<double> PermlImportance< T >::amounts_liaw`

Definition at line 513 of file PermlImportance.h.

6.23.4.3 `template<class T> std::vector<double> PermlImportance< T >::amounts_meng`

Definition at line 514 of file PermlImportance.h.

6.23.4.4 `template<class T> std::vector<double> PermlImportance< T >::amounts_raw`

Definition at line 511 of file PermlImportance.h.

6.23.4.5 `template<class T> std::vector<double> PermlImportance< T >::expX2_breiman`

Definition at line 508 of file PermlImportance.h.

6.23.4.6 `template<class T> std::vector<double> PermlImportance< T >::expX2_liaw`

Definition at line 509 of file PermlImportance.h.

6.23.4.7 `template<class T> std::vector<double> PermlImportance< T >::expX_values`

Definition at line 507 of file PermlImportance.h.

6.23.4.8 `template<class T> RJungleGen<T>* PermlImportance< T >::gen`

Definition at line 503 of file PermlImportance.h.

6.23.4.9 `template<class T> std::vector<double> PermlImportance< T >::ntrees`

Definition at line 505 of file PermlImportance.h.

6.23.4.10 `template<class T> std::vector<double> PermlImportance< T >::sd_breiman`

Definition at line 516 of file PermlImportance.h.

6.23.4.11 `template<class T> std::vector<double> PermlImportance< T >::sd_liaw`

Definition at line 517 of file PermlImportance.h.

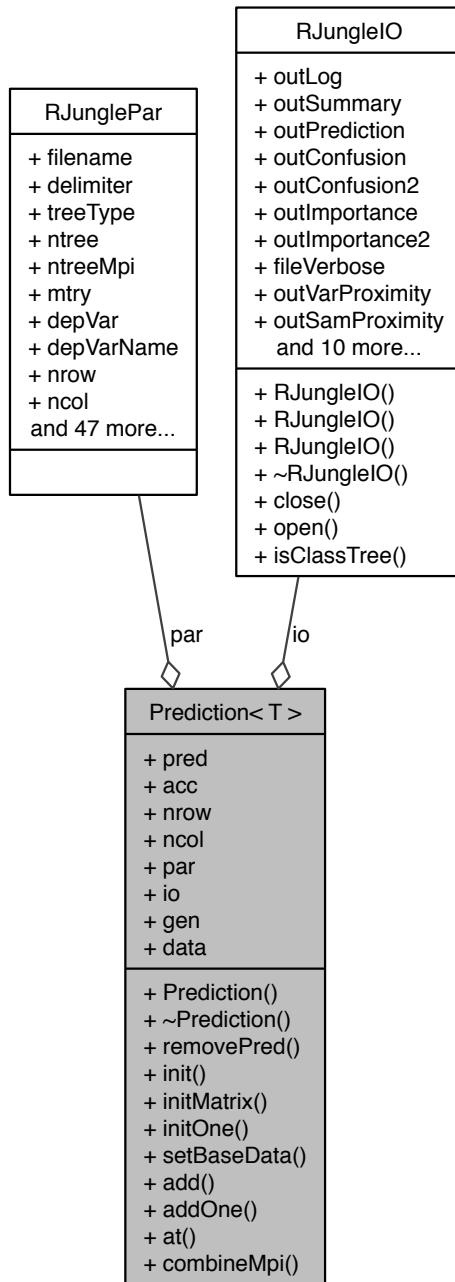
The documentation for this class was generated from the following file:

- [src/library/PermlImportance.h](#)

6.24 Prediction< T > Class Template Reference

```
#include <Prediction.h>
```

Collaboration diagram for Prediction< T >:



Public Member Functions

- `Prediction ()`
- `virtual ~Prediction ()`
- `void removePred ()`
- `void init (RJunglePar *par, RJungleIO *io, RJungleGen< T > *gen, DataFrame< T > *data)`
- `virtual void initMatrix (size_t nrow, size_t ntree)`
- `void initOne (RJunglePar *par, RJungleIO *io, RJungleGen< T > *gen, DataFrame< T > *data)`
- `void setBaseData (RJunglePar *par, RJungleIO *io, DataFrame< T > *data, RJungleGen< T > *gen)`
- `void add (CmpldTree< T > *cmpldTree, DataTreeSet *oobSet, long int treeId)`
- `void addOne (CmpldTree< T > *cmpldTree, DataTreeSet *oobSet, long int treeId)`
- `T at (size_t sample, size_t tree)`
- `void combineMpi ()`

Public Attributes

- `T * pred`
- `double * acc`
- `size_t nrow`
- `size_t ncol`
- `RJunglePar * par`
- `RJungleIO * io`
- `RJungleGen< T > * gen`
- `DataFrame< T > * data`

6.24.1 Detailed Description

`template<class T>class Prediction< T >`

Definition at line 20 of file Prediction.h.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 template<class T> Prediction< T >::Prediction () [inline]

Definition at line 22 of file Prediction.h.

6.24.2.2 template<class T> virtual Prediction< T >::~Prediction () [inline, virtual]

Definition at line 27 of file Prediction.h.

6.24.3 Member Function Documentation

6.24.3.1 `template<class T> void Prediction< T >::add (CmpIdTree< T > * cmpIdTree, DataTreeSet * oobSet, long int treelid) [inline]`

Definition at line 90 of file Prediction.h.

6.24.3.2 `template<class T> void Prediction< T >::addOne (CmpIdTree< T > * cmpIdTree, DataTreeSet * oobSet, long int treelid) [inline]`

Definition at line 112 of file Prediction.h.

6.24.3.3 `template<class T> T Prediction< T >::at (size_t sample, size_t tree) [inline]`

Definition at line 131 of file Prediction.h.

6.24.3.4 `template<class T> void Prediction< T >::combineMpi () [inline]`

Definition at line 135 of file Prediction.h.

6.24.3.5 `template<class T> void Prediction< T >::init (RJunglePar * par, RJungleIO * io, RJungleGen< T > * gen, DataFrame< T > * data) [inline]`

Definition at line 44 of file Prediction.h.

6.24.3.6 `template<class T> virtual void Prediction< T >::initMatrix (size_t nrow, size_t ntree) [inline, virtual]`

Definition at line 62 of file Prediction.h.

6.24.3.7 `template<class T> void Prediction< T >::initOne (RJunglePar * par, RJungleIO * io, RJungleGen< T > * gen, DataFrame< T > * data) [inline]`

Definition at line 74 of file Prediction.h.

6.24.3.8 `template<class T> void Prediction< T >::removePred () [inline]`

Definition at line 32 of file Prediction.h.

6.24.3.9 template<class T> void Prediction< T >::setBaseData (RJunglePar * *par*,
RJungleIO * *io*, DataFrame< T > * *data*, RJungleGen< T > * *gen*)
[inline]

Definition at line 81 of file Prediction.h.

6.24.4 Member Data Documentation

6.24.4.1 template<class T> double* Prediction< T >::acc

Definition at line 208 of file Prediction.h.

6.24.4.2 template<class T> DataFrame<T>* Prediction< T >::data

Definition at line 214 of file Prediction.h.

6.24.4.3 template<class T> RJungleGen<T>* Prediction< T >::gen

Definition at line 213 of file Prediction.h.

6.24.4.4 template<class T> RJungleIO* Prediction< T >::io

Definition at line 212 of file Prediction.h.

6.24.4.5 template<class T> size_t Prediction< T >::ncol

Definition at line 209 of file Prediction.h.

6.24.4.6 template<class T> size_t Prediction< T >::nrow

Definition at line 209 of file Prediction.h.

6.24.4.7 template<class T> RJunglePar* Prediction< T >::par

Definition at line 211 of file Prediction.h.

6.24.4.8 template<class T> T* Prediction< T >::pred

Definition at line 207 of file Prediction.h.

The documentation for this class was generated from the following file:

- [src/library/Prediction.h](#)

6.25 Profiler Class Reference

```
#include <Profiler.h>
```

Public Member Functions

- **Profiler ()**
Constructor.
- **~Profiler ()**
Destructor.
- **void start ()**
Start the stop watch.
- **void stop ()**
Stop the watch and add time difference to total time.

Public Attributes

- **clock_t timeStamp**
When did the profiler was starting the stop watch.
- **clock_t totalTime**
What is the total time that was consumed.
- **bool isTicking**
Tick indicator.

6.25.1 Detailed Description

Definition at line 25 of file Profiler.h.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 Profiler::Profiler ()

Constructor.

Definition at line 3 of file Profiler.cpp.

6.25.2.2 Profiler::~Profiler ()

Destructor.

Definition at line 9 of file Profiler.cpp.

6.25.3 Member Function Documentation

6.25.3.1 void Profiler::start()

Start the stop watch.

Definition at line 12 of file Profiler.cpp.

6.25.3.2 void Profiler::stop()

Stop the watch and add time difference to total time.

Definition at line 21 of file Profiler.cpp.

6.25.4 Member Data Documentation

6.25.4.1 bool Profiler::isTick

Tick indicator.

Definition at line 54 of file Profiler.h.

6.25.4.2 clock_t Profiler::timeStamp

When did the profiler was starting the stop watch.

Definition at line 48 of file Profiler.h.

6.25.4.3 clock_t Profiler::totalTime

What is the total time that was consumed.

Definition at line 51 of file Profiler.h.

The documentation for this class was generated from the following files:

- [src/library/Profiler.h](#)
- [src/library/Profiler.cpp](#)

6.26 ProtCount Class Reference

```
#include <RJungleProto.h>
```

Public Member Functions

- [ProtCount \(size_t numSamples, uli_t row\)](#)

- [ProtCount \(\)](#)
- [void loadUp \(ProtCount &prot\)](#)

Public Attributes

- [size_t numOfSamples](#)
- [uli_t row](#)

6.26.1 Detailed Description

Definition at line 11 of file R JungleProto.h.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 ProtCount::ProtCount (*size_t numSamples, uli_t row*) [inline]

Definition at line 13 of file R JungleProto.h.

6.26.2.2 ProtCount::ProtCount () [inline]

Definition at line 18 of file R JungleProto.h.

6.26.3 Member Function Documentation

6.26.3.1 void ProtCount::loadUp (ProtCount & prot) [inline]

Definition at line 23 of file R JungleProto.h.

6.26.4 Member Data Documentation

6.26.4.1 size_t ProtCount::numOfSamples

Definition at line 30 of file R JungleProto.h.

6.26.4.2 uli_t ProtCount::row

Definition at line 31 of file R JungleProto.h.

The documentation for this class was generated from the following file:

- [src/library/R JungleProto.h](#)

6.27 Proximities< T > Class Template Reference

```
#include <Proximities.h>
```

Public Member Functions

- `Proximities ()`
- `Proximities (uli_t nrow, uli_t ncol)`
- `Proximities (uli_t nrow, uli_t ncol, DataFrame< T > *data)`
- `Proximities (DataFrame< T > *data)`
- `void initWithData (DataFrame< T > *data)`
- `virtual ~Proximities ()`
- `void initMatrix ()`
- `uli_t getnrow ()`
- `uli_t getncol ()`
- `std::vector< double > getRow (uli_t j)`
- `std::vector< double > getCol (uli_t j)`
- `std::ostream & print (std::ostream &os) const`
- `void printCSV (std::ostream &os) const`
- `void reset ()`
- `double at (uli_t i, uli_t j) const`
- `void add (size_t i, size_t j, double val)`
- `void add (Proximities< T > &prox)`
- `void div (double val)`
- `void div (size_t i, size_t j, double val)`
- `void set (size_t i, size_t j, double val)`
- `void setDiag (double val)`
- `void setDim (size_t i, size_t j)`
- `virtual void combineMPI ()`
- `void addVecs (double *&to, std::vector< double > &from)`

Public Attributes

- `uli_t nrow`
- `uli_t ncol`
- `double ** proxMatrix`
- `DataFrame< T > * data`

6.27.1 Detailed Description

```
template<class T>class Proximities< T >
```

Definition at line 21 of file Proximities.h.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 `template<class T> Proximities< T >::Proximities() [inline]`

Definition at line 24 of file Proximities.h.

6.27.2.2 `template<class T> Proximities< T >::Proximities(uli_t nrow, uli_t ncol) [inline]`

Definition at line 29 of file Proximities.h.

6.27.2.3 `template<class T> Proximities< T >::Proximities(uli_t nrow, uli_t ncol, DataFrame< T > * data) [inline]`

Definition at line 36 of file Proximities.h.

6.27.2.4 `template<class T> Proximities< T >::Proximities(DataFrame< T > * data) [inline]`

Definition at line 43 of file Proximities.h.

6.27.2.5 `template<class T> virtual Proximities< T >::~Proximities() [inline, virtual]`

Definition at line 69 of file Proximities.h.

6.27.3 Member Function Documentation

6.27.3.1 `template<class T> void Proximities< T >::add(size_t i, size_t j, double val) [inline]`

Definition at line 176 of file Proximities.h.

6.27.3.2 `template<class T> void Proximities< T >::add(Proximities< T > & proxy) [inline]`

Definition at line 180 of file Proximities.h.

6.27.3.3 `template<class T> void Proximities< T >::addVecs(double *& to, std::vector< double > & from) [inline]`

Definition at line 264 of file Proximities.h.

6.27.3.4 template<class T> double Proximities< T >::at(uli_t i, uli_t j) const
[inline]

Definition at line 173 of file Proximities.h.

6.27.3.5 template<class T> virtual void Proximities< T >::combineMpi()
[inline, virtual]

Definition at line 220 of file Proximities.h.

6.27.3.6 template<class T> void Proximities< T >::div(double val) [inline]

Definition at line 188 of file Proximities.h.

6.27.3.7 template<class T> void Proximities< T >::div(size_t i, size_t j, double val)
[inline]

Definition at line 196 of file Proximities.h.

6.27.3.8 template<class T> std::vector<double> Proximities< T >::getCol(uli_t j)
[inline]

Definition at line 117 of file Proximities.h.

6.27.3.9 template<class T> uli_t Proximities< T >::getncol() [inline]

Definition at line 104 of file Proximities.h.

6.27.3.10 template<class T> uli_t Proximities< T >::getnrow() [inline]

Definition at line 103 of file Proximities.h.

6.27.3.11 template<class T> std::vector<double> Proximities< T >::getRow(uli_t j)
[inline]

Definition at line 106 of file Proximities.h.

6.27.3.12 template<class T> void Proximities< T >::initMatrix() [inline]

Definition at line 82 of file Proximities.h.

6.27.3.13 `template<class T> void Proximities< T >::initWithData(DataFrame< T > * data) [inline]`

Definition at line 57 of file Proximities.h.

6.27.3.14 `template<class T> std::ostream& Proximities< T >::print(std::ostream & os) const [inline]`

Definition at line 128 of file Proximities.h.

6.27.3.15 `template<class T> void Proximities< T >::printCSV(std::ostream & os) const [inline]`

Definition at line 147 of file Proximities.h.

6.27.3.16 `template<class T> void Proximities< T >::reset() [inline]`

Definition at line 164 of file Proximities.h.

6.27.3.17 `template<class T> void Proximities< T >::set(size_t i, size_t j, double val) [inline]`

Definition at line 200 of file Proximities.h.

6.27.3.18 `template<class T> void Proximities< T >::setDiag(double val) [inline]`

Definition at line 204 of file Proximities.h.

6.27.3.19 `template<class T> void Proximities< T >::setDim(size_t i, size_t j) [inline]`

Definition at line 212 of file Proximities.h.

6.27.4 Member Data Documentation

6.27.4.1 `template<class T> DataFrame<T>* Proximities< T >::data`

Definition at line 273 of file Proximities.h.

6.27.4.2 `template<class T> uli_t Proximities< T >::ncol`

Definition at line 271 of file Proximities.h.

6.27.4.3 template<class T> uli_t Proximities< T >::nrow

Definition at line 270 of file Proximities.h.

6.27.4.4 template<class T> double Proximities< T >::proxMatrix**

Definition at line 272 of file Proximities.h.

The documentation for this class was generated from the following file:

- [src/library/Proximities.h](#)

6.28 R JungleAcc< T > Class Template Reference

```
#include <R JungleAcc.h>
```

Public Member Functions

- [R JungleAcc \(\)](#)
- [virtual ~R JungleAcc \(\)](#)

Static Public Member Functions

- static double [getAccuracyCmpld \(DataFrame< T > *data, T *pred, uli_t depVar, DataTreeSet &dataSet, long int treeld, bool one, bool showInfos, R JungleIO &io\)](#)
- static double [getAccuracyCmpldSos \(DataFrame< T > *data, T *pred, uli_t depVar, DataTreeSet &dataSet, long int treeld, bool one, bool showInfos, R JungleIO &io\)](#)
- static double [getDevianceCmpld \(DataFrame< T > *data, T *pred, uli_t depVar, DataTreeSet &dataSet, long int treeld, bool one, bool showInfos, R JungleIO &io\)](#)

6.28.1 Detailed Description

template<class T>class R JungleAcc< T >

Definition at line 45 of file R JungleAcc.h.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 template<class T > R JungleAcc< T >::R JungleAcc()**6.28.2.2 template<class T > virtual R JungleAcc< T >::~R JungleAcc() [virtual]**

6.28.3 Member Function Documentation

6.28.3.1 template<class T > static double RJungleAcc< T >::getAccuracyCmpld (DataFrame< T > * *data*, T * *pred*, uli_t *depVar*, DataTreeSet & *dataSet*, long int *treeId*, bool *one*, bool *showInfos*, RJungleIO & *io*) [inline, static]

Definition at line 50 of file RJungleAcc.h.

6.28.3.2 template<class T > static double RJungleAcc< T >::getAccuracyCmpldSos (DataFrame< T > * *data*, T * *pred*, uli_t *depVar*, DataTreeSet & *dataSet*, long int *treeId*, bool *one*, bool *showInfos*, RJungleIO & *io*) [inline, static]

Definition at line 266 of file RJungleAcc.h.

6.28.3.3 template<class T > static double RJungleAcc< T >::getDevianceCmpld (DataFrame< T > * *data*, T * *pred*, uli_t *depVar*, DataTreeSet & *dataSet*, long int *treeId*, bool *one*, bool *showInfos*, RJungleIO & *io*) [inline, static]

Definition at line 378 of file RJungleAcc.h.

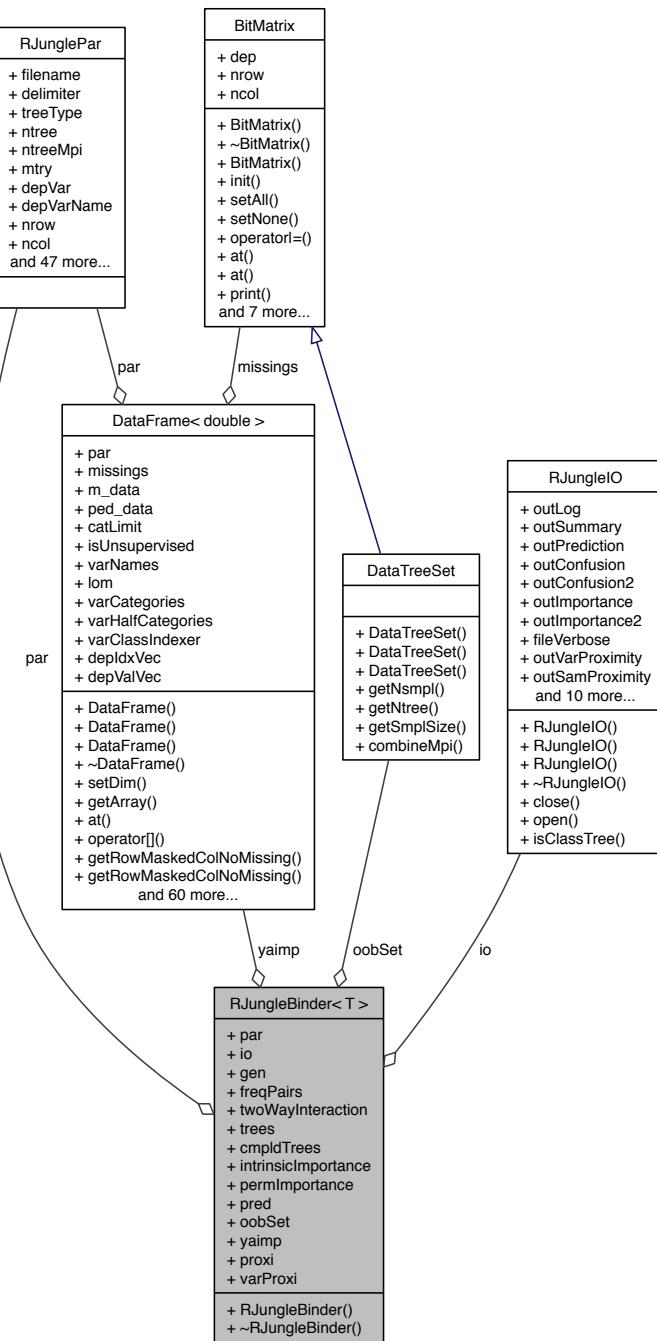
The documentation for this class was generated from the following file:

- src/library/RJungleAcc.h

6.29 RJungleBinder< T > Class Template Reference

```
#include <RJungleBinder.h>
```

Collaboration diagram for RJungleBinder< T >:



Public Member Functions

- [RJungleBinder \(\)](#)
- [virtual ~RJungleBinder \(\)](#)

Public Attributes

- [RJunglePar par](#)
- [RJungleIO io](#)
- [RJungleGen< T > gen](#)
- [std::vector< std::pair< double, uli_t > > freqPairs](#)
- [std::vector< std::pair< double, std::pair< uli_t, uli_t > > > twoWayInteraction](#)
- [std::vector< Tree< T, uli_t > * > trees](#)
- [std::vector< CmpldTree< T > * > cmpldTrees](#)
- [Importance< T > * intrinsicImportance](#)
- [PermlImportance< T > * permImportance](#)
- [Prediction< T > pred](#)
- [DataTreeSet oobSet](#)
- [DataFrame< double > * yaimp](#)
- [Proximities< T > prox](#)
- [Proximities< T > varProxi](#)

6.29.1 Detailed Description

`template<class T>class RJungleBinder< T >`

Definition at line 20 of file `RJungleBinder.h`.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 `template<class T> RJungleBinder< T >::RJungleBinder() [inline]`

Definition at line 22 of file `RJungleBinder.h`.

6.29.2.2 `template<class T> virtual RJungleBinder< T >::~RJungleBinder() [inline, virtual]`

Definition at line 27 of file `RJungleBinder.h`.

6.29.3 Member Data Documentation

6.29.3.1 `template<class T> std::vector<CmpldTree<T> *> RJungleBinder< T >::cmpldTrees`

Definition at line 69 of file `RJungleBinder.h`.

6.29.3.2 template<class T> std::vector<std::pair<double, uli_t>> **RJungleBinder< T >::freqPairs**

Definition at line 64 of file RJungleBinder.h.

6.29.3.3 template<class T> **RJungleGen<T>** **RJungleBinder< T >::gen**

Definition at line 62 of file RJungleBinder.h.

6.29.3.4 template<class T> **Importance<T>*** **RJungleBinder< T >::intrinsicImportance**

Definition at line 70 of file RJungleBinder.h.

6.29.3.5 template<class T> **RJungleIO** **RJungleBinder< T >::io**

Definition at line 61 of file RJungleBinder.h.

6.29.3.6 template<class T> **DataTreeSet** **RJungleBinder< T >::oobSet**

Definition at line 75 of file RJungleBinder.h.

6.29.3.7 template<class T> **RJunglePar** **RJungleBinder< T >::par**

Definition at line 60 of file RJungleBinder.h.

6.29.3.8 template<class T> **PermlImportance<T>*** **RJungleBinder< T >::permImportance**

Definition at line 71 of file RJungleBinder.h.

6.29.3.9 template<class T> **Prediction<T>** **RJungleBinder< T >::pred**

Definition at line 72 of file RJungleBinder.h.

6.29.3.10 template<class T> **Proximities<T>** **RJungleBinder< T >::prox**

Definition at line 81 of file RJungleBinder.h.

6.29.3.11 template<class T> std::vector<Tree<T, uli_t>*> **RJungleBinder< T >::trees**

Definition at line 68 of file RJungleBinder.h.

6.29.3.12 `template<class T> std::vector<std::pair<double, std::pair<uli_t, uli_t>>>`
`RJungleBinder< T >::twoWayInteraction`

Definition at line 65 of file RJungleBinder.h.

6.29.3.13 `template<class T> Proximities<T> RJungleBinder< T >::varProxi`

Definition at line 84 of file RJungleBinder.h.

6.29.3.14 `template<class T> DataFrame<double>* RJungleBinder< T >::yaimp`

Definition at line 78 of file RJungleBinder.h.

The documentation for this class was generated from the following file:

- `src/library/RJungleBinder.h`

6.30 RJungleCompiler< T > Class Template Reference

```
#include <RJungleCompiler.h>
```

Public Member Functions

- `RJungleCompiler ()`
- `virtual ~RJungleCompiler ()`

Static Public Member Functions

- `static void compileTrees (RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, std::vector< Tree< T, uli_t > * > &trees, std::vector< CmpldTree< T > * > &cmpldTrees)`

6.30.1 Detailed Description

`template<class T>class RJungleCompiler< T >`

Definition at line 29 of file RJungleCompiler.h.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 `template<class T> RJungleCompiler< T >::RJungleCompiler()`

6.30.2.2 template<class T> virtual RJungleCompiler< T >::~RJungleCompiler()
[virtual]

6.30.3 Member Function Documentation

6.30.3.1 template<class T> static void RJungleCompiler< T >::compileTrees(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen, std::vector< Tree< T, uli_t > * > & trees, std::vector< CmpIdTree< T > * > & cmpIdTrees)
[inline, static]

Definition at line 34 of file RJungleCompiler.h.

The documentation for this class was generated from the following file:

- [src/library/RJungleCompiler.h](#)

6.31 RJungleConfusion< T > Class Template Reference

```
#include <RJungleConfusion.h>
```

Public Member Functions

- [RJungleConfusion\(\)](#)
- [virtual ~RJungleConfusion\(\)](#)

Static Public Member Functions

- static void [printConfMatNew\(RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, T *pred, DataTreeSet &oobSet, uli_t iteration=0, uli_t numOfVars=0\)](#)

6.31.1 Detailed Description

```
template<class T> class RJungleConfusion< T >
```

Definition at line 28 of file RJungleConfusion.h.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 template<class T> [RJungleConfusion< T >::RJungleConfusion\(\)](#)

6.31.2.2 template<class T> virtual [RJungleConfusion< T >::~RJungleConfusion\(\)](#)
[virtual]

6.31.3 Member Function Documentation

6.31.3.1 `template<class T > static void RJungleConfusion< T >::printConfMatNew (RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, T * pred, DataTreeSet & oobSet, uli_t iteration = 0, uli_t numOfVars = 0) [inline, static]`

Definition at line 33 of file RJungleConfusion.h.

The documentation for this class was generated from the following file:

- [src/library/RJungleConfusion.h](#)

6.32 RJungleCtrl< T > Class Template Reference

```
#include <RJungleCtrl.h>
```

Public Member Functions

- [RJungleCtrl \(\)](#)
- [virtual ~RJungleCtrl \(\)](#)
- [void setParAndIO \(RJunglePar &par_, RJungleIO &io_\)](#)
- [void autoBuildInternal \(RJunglePar &par_, RJungleIO &io_, RJungleGen< T > &gen_, DataFrame< T > &data, std::vector< uli_t > *colMaskVec\)](#)
- [void imputeSNPs \(RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, std::vector< uli_t > *colMaskVec\)](#)
- [void tuneMtry \(RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, std::vector< uli_t > *colMaskVec\)](#)

Static Public Member Functions

- [static void autoBuild \(RJunglePar par\)](#)

Private Attributes

- [RJungleBinder< T > binder](#)

6.32.1 Detailed Description

```
template<class T>class RJungleCtrl< T >
```

Definition at line 51 of file RJungleCtrl.h.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 `template<class T> RJungleCtrl< T >::RJungleCtrl() [inline]`

Definition at line 53 of file RJungleCtrl.h.

6.32.2.2 `template<class T> virtual RJungleCtrl< T >::~RJungleCtrl() [inline, virtual]`

Definition at line 56 of file RJungleCtrl.h.

6.32.3 Member Function Documentation

6.32.3.1 `template<class T> static void RJungleCtrl< T >::autoBuild(RJunglePar par) [inline, static]`

Definition at line 520 of file RJungleCtrl.h.

6.32.3.2 `template<class T> void RJungleCtrl< T >::autoBuildInternal(RJunglePar & par_, RJungleIO & io_, RJungleGen< T > & gen_, DataFrame< T > & data, std::vector< uli_t > * colMaskVec) [inline]`

Definition at line 68 of file RJungleCtrl.h.

6.32.3.3 `template<class T> void RJungleCtrl< T >::imputeSNPs(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, std::vector< uli_t > * colMaskVec) [inline]`

Definition at line 505 of file RJungleCtrl.h.

6.32.3.4 `template<class T> void RJungleCtrl< T >::setParAndIO(RJunglePar & par_, RJungleIO & io_) [inline]`

Definition at line 59 of file RJungleCtrl.h.

6.32.3.5 `template<class T> void RJungleCtrl< T >::tuneMtry(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, std::vector< uli_t > * colMaskVec) [inline]`

Definition at line 511 of file RJungleCtrl.h.

6.32.4 Member Data Documentation

6.32.4.1 template<class T> RJungleBinder<T> RJungleCtrl< T >::binder
[private]

Definition at line 664 of file RJungleCtrl.h.

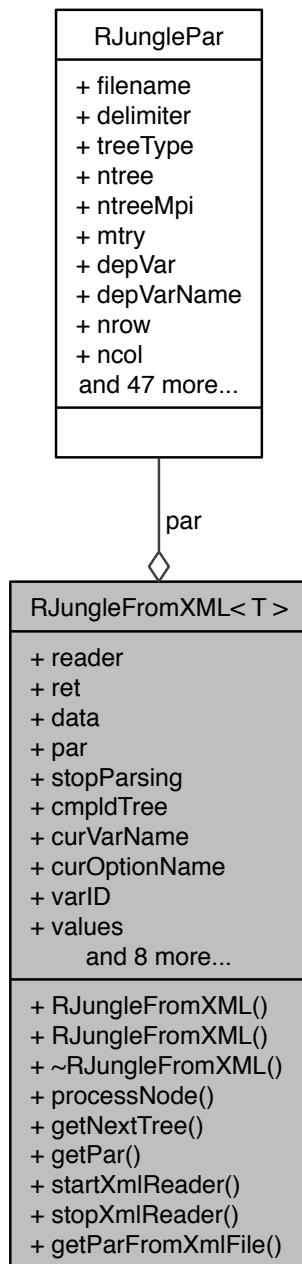
The documentation for this class was generated from the following file:

- src/library/[RJungleCtrl.h](#)

6.33 RJungleFromXML< T > Class Template Reference

```
#include <RJungleFromXML.h>
```

Collaboration diagram for RJungleFromXML< T >:



Public Member Functions

- `RJungleFromXML ()`
- `RJungleFromXML (RJunglePar &par)`
- virtual `~RJungleFromXML ()`
- void `processNode ()`
`Processes each node in XML file and creates a whole tree successively.`
- `CmpldTree< T > * getNextTree ()`
`Parse the RJungle XML file for next tree and store it.`
- `RJunglePar getPar ()`
`Parse the RJungle XML file for parameters and store it.`
- void `startXmlReader (DataFrame< T > *data)`
`Initialize XML reader and open file.`
- void `stopXmlReader ()`
`Stop XML reading and clean up the memory.`
- `RJunglePar getParFromFile ()`

Public Attributes

- `xmlTextReaderPtr reader`
- int `ret`
- `DataFrame< T > * data`
- `RJunglePar par`
- bool `stopParsing`
- `CmpldTree< T > * cmpldTree`
- std::string `curVarName`
- std::string `curOptionName`
- std::vector< std::vector< uli_t > > `varID`
- std::vector< std::vector< std::vector< T > > > `values`
- std::vector< std::vector< uli_t > > `branches`
- std::vector< T > `classes`
- std::vector< uli_t > `indexes`
- std::vector< std::vector< double > > `extra`
- `uli_t varSize`
- `uli_t valueSize`
- `uli_t valueWidth`
- `uli_t branchSize`
- `uli_t termID`

6.33.1 Detailed Description

`template<class T>class RJungleFromXML< T >`

Definition at line 25 of file `RJungleFromXML.h`.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 `template<class T> RJungleFromXML< T >::RJungleFromXML() [inline]`

Definition at line 27 of file RJungleFromXML.h.

6.33.2.2 `template<class T> RJungleFromXML< T >::RJungleFromXML(RJunglePar & par) [inline]`

Definition at line 30 of file RJungleFromXML.h.

6.33.2.3 `template<class T> virtual RJungleFromXML< T >::~RJungleFromXML() [inline, virtual]`

Definition at line 33 of file RJungleFromXML.h.

6.33.3 Member Function Documentation

6.33.3.1 `template<class T> CmpIdTree<T>* RJungleFromXML< T >::getNextTree() [inline]`

Parse the RJungle XML file for next tree and store it.

Returns

A [CmpIdTree](#) object

Definition at line 209 of file RJungleFromXML.h.

6.33.3.2 `template<class T> RJunglePar RJungleFromXML< T >::getPar() [inline]`

Parse the RJungle XML file for parameters and store it.

Returns

RJungle Parameter

Definition at line 240 of file RJungleFromXML.h.

6.33.3.3 `template<class T> RJunglePar RJungleFromXML< T >::getParFromXmlFile() [inline]`

Definition at line 334 of file RJungleFromXML.h.

**6.33.3.4 template<class T> void RJungleFromXML< T >::processNode ()
[inline]**

Processes each node in XML file and creates a whole tree successively.

Parameters

<i>cmpldTree</i>	Pointer to location where the grown tree is stored
------------------	--

Definition at line 42 of file RJungleFromXML.h.

6.33.3.5 template<class T> void RJungleFromXML< T >::startXmlReader (DataFrame< T > * *data*) [inline]

Initialize XML reader and open file.

Parameters

<i>Data</i>	a pointer to the input data
-------------	-----------------------------

Definition at line 272 of file RJungleFromXML.h.

**6.33.3.6 template<class T> void RJungleFromXML< T >::stopXmlReader ()
[inline]**

Stop XML reading and clean up the memory.

Definition at line 290 of file RJungleFromXML.h.

6.33.4 Member Data Documentation

6.33.4.1 template<class T> std::vector<std::vector<uli_t>> RJungleFromXML< T >::branches

Definition at line 365 of file RJungleFromXML.h.

6.33.4.2 template<class T> uli_t RJungleFromXML< T >::branchSize

Definition at line 373 of file RJungleFromXML.h.

6.33.4.3 template<class T> std::vector<T> RJungleFromXML< T >::classes

Definition at line 366 of file RJungleFromXML.h.

6.33.4.4 template<class T> CmpIdTree<T>* **RJungleFromXML< T >::cmpIdTree**

Definition at line 357 of file RJungleFromXML.h.

6.33.4.5 template<class T> std::string **RJungleFromXML< T >::curOptionName**

Definition at line 361 of file RJungleFromXML.h.

6.33.4.6 template<class T> std::string **RJungleFromXML< T >::curVarName**

Definition at line 360 of file RJungleFromXML.h.

6.33.4.7 template<class T> DataFrame<T>* **RJungleFromXML< T >::data**

Definition at line 348 of file RJungleFromXML.h.

6.33.4.8 template<class T> std::vector<std::vector<double>> **RJungleFromXML< T >::extra**

Definition at line 368 of file RJungleFromXML.h.

6.33.4.9 template<class T> std::vector<ul_i_t> **RJungleFromXML< T >::indexes**

Definition at line 367 of file RJungleFromXML.h.

6.33.4.10 template<class T> **RJunglePar RJungleFromXML< T >::par**

Definition at line 351 of file RJungleFromXML.h.

6.33.4.11 template<class T> xmlTextReaderPtr **RJungleFromXML< T >::reader**

Definition at line 344 of file RJungleFromXML.h.

6.33.4.12 template<class T> int **RJungleFromXML< T >::ret**

Definition at line 345 of file RJungleFromXML.h.

6.33.4.13 template<class T> bool **RJungleFromXML< T >::stopParsing**

Definition at line 354 of file RJungleFromXML.h.

6.33.4.14 template<class T> uli_t **RJungleFromXML**< T >::termID

Definition at line 374 of file RJungleFromXML.h.

6.33.4.15 template<class T> std::vector<std::vector<std::vector<T>>> **RJungleFromXML**< T >::values

Definition at line 364 of file RJungleFromXML.h.

6.33.4.16 template<class T> uli_t **RJungleFromXML**< T >::valueSize

Definition at line 371 of file RJungleFromXML.h.

6.33.4.17 template<class T> uli_t **RJungleFromXML**< T >::valueWidth

Definition at line 372 of file RJungleFromXML.h.

6.33.4.18 template<class T> std::vector<std::vector<uli_t>> **RJungleFromXML**< T >::varID

Definition at line 363 of file RJungleFromXML.h.

6.33.4.19 template<class T> uli_t **RJungleFromXML**< T >::varSize

Definition at line 370 of file RJungleFromXML.h.

The documentation for this class was generated from the following file:

- src/library/[RJungleFromXML.h](#)

6.34 **RJungleGen**< T > Class Template Reference

```
#include <RJungleGen.h>
```

Public Member Functions

- [RJungleGen \(\)](#)
- [~RJungleGen \(\)](#)
- void [init \(RJunglePar &par, DataFrame< T > &data\)](#)

Public Attributes

- `BuildinGenFct< T > fct`
- `void * sdl_library`
- `bool isPlugin`
- `bool wasInitialized`

6.34.1 Detailed Description

`template<class T>class RJungleGen< T >`

Definition at line 33 of file RJungleGen.h.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 `template<class T> RJungleGen< T >::RJungleGen() [inline]`

Definition at line 35 of file RJungleGen.h.

6.34.2.2 `template<class T> RJungleGen< T >::~RJungleGen() [inline]`

Definition at line 41 of file RJungleGen.h.

6.34.3 Member Function Documentation

6.34.3.1 `template<class T> void RJungleGen< T >::init(RJunglePar & par, DataFrame< T > & data) [inline]`

Definition at line 55 of file RJungleGen.h.

6.34.4 Member Data Documentation

6.34.4.1 `template<class T> BuildinGenFct<T> RJungleGen< T >::fct`

Definition at line 148 of file RJungleGen.h.

6.34.4.2 `template<class T> bool RJungleGen< T >::isPlugin`

Definition at line 159 of file RJungleGen.h.

6.34.4.3 `template<class T> void* RJungleGen< T >::sdl_library`

Definition at line 153 of file RJungleGen.h.

6.34.4.4 template<class T> bool RJungleGen< T >::wasInitialized

Definition at line 160 of file RJungleGen.h.

The documentation for this class was generated from the following file:

- [src/library/RJungleGen.h](#)

6.35 RJungleGrow< T > Class Template Reference

```
#include <RJungleGrow.h>
```

Public Member Functions

- [RJungleGrow \(\)](#)
- [virtual ~RJungleGrow \(\)](#)

Static Public Member Functions

- [static void growLocal \(DataFrame< T > &data, RJungleBinder< T > &binder, std::vector< uli_t > *colMaskVec, int iteration\)](#)

Grow many trees and process various additional methods for analysing data.
- [static bool twoWayInteractionCmp \(std::pair< double, std::pair< uli_t, uli_t > > a, std::pair< double, std::pair< uli_t, uli_t > > b\)](#)
- [static bool doSampleProxi \(RJunglePar &par\)](#)
- [static bool doPermImp \(RJunglePar &par, std::vector< uli_t > *colMaskVec\)](#)
- [static bool doVarProxi \(RJunglePar &par\)](#)
- [static bool doSaveJungle \(RJunglePar &par\)](#)
- [static bool doFetchJungleFromFile \(RJunglePar &par\)](#)

6.35.1 Detailed Description

```
template<class T>class RJungleGrow< T >
```

Definition at line 41 of file RJungleGrow.h.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 template<class T > RJungleGrow< T >::RJungleGrow () [inline]

Definition at line 43 of file RJungleGrow.h.

6.35.2.2 template<class T > virtual **RJungleGrow< T >::~RJungleGrow()**
[inline, virtual]

Definition at line 46 of file RJungleGrow.h.

6.35.3 Member Function Documentation

6.35.3.1 template<class T > static bool **RJungleGrow< T >::doFetchJungleFromFile(RJunglePar & par)** [inline, static]

Definition at line 570 of file RJungleGrow.h.

6.35.3.2 template<class T > static bool **RJungleGrow< T >::doPermlmp(RJunglePar & par, std::vector< uli_t > * colMaskVec)** [inline, static]

Definition at line 541 of file RJungleGrow.h.

6.35.3.3 template<class T > static bool **RJungleGrow< T >::doSampleProxi(RJunglePar & par)** [inline, static]

Definition at line 531 of file RJungleGrow.h.

6.35.3.4 template<class T > static bool **RJungleGrow< T >::doSaveJungle(RJunglePar & par)** [inline, static]

Definition at line 566 of file RJungleGrow.h.

6.35.3.5 template<class T > static bool **RJungleGrow< T >::doVarProxi(RJunglePar & par)** [inline, static]

Definition at line 556 of file RJungleGrow.h.

6.35.3.6 template<class T > static void **RJungleGrow< T >::growLocal(DataFrame< T > & data, RJungleBinder< T > & binder, std::vector< uli_t > * & colMaskVec, int iteration)** [inline, static]

Grow many trees and process various additional methods for analysing data.

Parameters

<i>data</i>	RJungle input data.
<i>binder</i>	A binder which contains all relevant informations for forest growing.
<i>colMaskVec</i>	A selection of all available variables.
<i>iteration</i>	Number of passed iteration in backelimination / imputing method.

Definition at line 59 of file R JungleGrow.h.

```
6.35.3.7 template<class T> static bool R JungleGrow< T >::twoWayInteractionCmp (
    std::pair< double, std::pair< uli_t, uli_t > > a, std::pair< double, std::pair< uli_t,
    uli_t > > b ) [inline, static]
```

Definition at line 525 of file R JungleGrow.h.

The documentation for this class was generated from the following file:

- [src/library/R JungleGrow.h](#)

6.36 R JungleHelper< T > Class Template Reference

```
#include <R JungleHelper.h>
```

Public Member Functions

- [R JungleHelper \(\)](#)
- [virtual ~R JungleHelper \(\)](#)

Static Public Member Functions

- [static void printXml \(R JungleIO &io, std::vector< Tree< T, uli_t > * > &trees\)](#)
- [static void printHeader \(R JunglePar &par, R JungleIO &io, time_t &start\)](#)
- [static void printFooter \(R JunglePar &par, R JungleIO &io, time_t &start, time_t &end, clock_t &startgrow, clock_t &endgrow\)](#)
- [static void printR JunglePar \(R JunglePar &par, std::ostream &os\)](#)
- [static void printXmlHeader \(R JunglePar &par, std::ostream &os\)](#)
- [static void printXmlFooter \(R JungleIO &io\)](#)
- [static void printXmlRaw \(R JungleIO &io, CmpldTree< T > *&cmpldTree, uli_t treeld\)](#)
- [static void summary \(R JungleIO &io, CmpldTree< T > &cmpldTree\)](#)
- [static void tuneMtry \(R JunglePar &par, std::vector< uli_t > *colMaskVec\)](#)
- [static void tuneNtree \(R JunglePar &par, std::vector< uli_t > *colMaskVec\)](#)
- [static uli_t setMtryClassification \(uli_t ncol\)](#)
- [static uli_t setMtryRegression \(uli_t ncol\)](#)

6.36.1 Detailed Description

```
template<class T> class R JungleHelper< T >
```

Definition at line 27 of file R JungleHelper.h.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 `template<class T > RJungleHelper< T >::RJungleHelper()`

6.36.2.2 `template<class T > virtual RJungleHelper< T >::~RJungleHelper() [virtual]`

6.36.3 Member Function Documentation

6.36.3.1 `template<class T > static void RJungleHelper< T >::printFooter (RJunglePar & par, RJungleIO & io, time_t & start, time_t & end, clock_t & startgrow, clock_t & endgrow) [inline, static]`

Definition at line 83 of file RJungleHelper.h.

6.36.3.2 `template<class T > static void RJungleHelper< T >::printHeader (RJunglePar & par, RJungleIO & io, time_t & start) [inline, static]`

Definition at line 59 of file RJungleHelper.h.

6.36.3.3 `template<class T > static void RJungleHelper< T >::printRJunglePar (RJunglePar & par, std::ostream & os) [inline, static]`

Definition at line 91 of file RJungleHelper.h.

6.36.3.4 `template<class T > static void RJungleHelper< T >::printXml (RJungleIO & io, std::vector< Tree< T, uli_t >*> & trees) [inline, static]`

Definition at line 32 of file RJungleHelper.h.

6.36.3.5 `template<class T > static void RJungleHelper< T >::printXmlFooter (RJungleIO & io) [inline, static]`

Definition at line 189 of file RJungleHelper.h.

6.36.3.6 `template<class T > static void RJungleHelper< T >::printXmlHeader (RJunglePar & par, std::ostream & os) [inline, static]`

Definition at line 127 of file RJungleHelper.h.

6.36.3.7 `template<class T > static void RJungleHelper< T >::printXmlRaw (RJungleIO & io, CmpIdTree< T >*& cmpIdTree, uli_t treeld) [inline, static]`

Definition at line 194 of file RJungleHelper.h.

6.36.3.8 template<class T > static uli_t RJungleHelper< T >::setMtryClassification (uli_t ncol) [inline, static]

Definition at line 257 of file RJungleHelper.h.

6.36.3.9 template<class T > static uli_t RJungleHelper< T >::setMtryRegression (uli_t ncol) [inline, static]

Definition at line 261 of file RJungleHelper.h.

6.36.3.10 template<class T > static void RJungleHelper< T >::summary (RJungleIO & io, CmpldTree< T > & cmpldTree) [inline, static]

Definition at line 201 of file RJungleHelper.h.

6.36.3.11 template<class T > static void RJungleHelper< T >::tuneMtry (RJunglePar & par, std::vector< uli_t > * colMaskVec) [inline, static]

Definition at line 239 of file RJungleHelper.h.

6.36.3.12 template<class T > static void RJungleHelper< T >::tuneNtree (RJunglePar & par, std::vector< uli_t > * colMaskVec) [inline, static]

Definition at line 246 of file RJungleHelper.h.

The documentation for this class was generated from the following file:

- [src/library/RJungleHelper.h](#)

6.37 RJungleImportance< T > Class Template Reference

```
#include <RJungleImportance.h>
```

Public Member Functions

- [RJungleImportance \(\)](#)
- virtual [~RJungleImportance \(\)](#)

Static Public Member Functions

- static void [makePermVarImp2 \(RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, DataTreeSet &oobSet, std::vector< uli_t > * &colMaskVec, std::vector< CmpldTree< T > * > &cmpldTrees, std::vector< std::pair< double, uli_t > > &permVarImpPairVec\)](#)

- static void `makePermVarImp3` (RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, DataTreeSet &oobSet, std::vector< uli_t > *&colMaskVec, std::vector< CmpldTree< T > * > &cmpldTrees, std::vector< std::pair< double, uli_t > > &permVarImpPairVec)
- static void `printVarImp` (RJunglePar &par, RJungleIO &io, DataFrame< T > &data, std::vector< std::pair< double, uli_t > > &freqPairs, uli_t iteration=0)
- static void `printPermVarImp` (RJunglePar &par, RJungleIO &io, DataFrame< T > &data, std::vector< std::pair< double, uli_t > > &freqPairs, uli_t iteration=0)

6.37.1 Detailed Description

`template<class T>class RJungleImportance< T >`

Definition at line 29 of file RJungleImportance.h.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 `template<class T > RJungleImportance< T >::RJungleImportance()`

6.37.2.2 `template<class T > virtual RJungleImportance< T >::~RJungleImportance() [virtual]`

6.37.3 Member Function Documentation

6.37.3.1 `template<class T > static void RJungleImportance< T >::makePermVarImp2(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, DataTreeSet & oobSet, std::vector< uli_t > *& colMaskVec, std::vector< CmpldTree< T > * > & cmpldTrees, std::vector< std::pair< double, uli_t > > & permVarImpPairVec) [inline, static]`

Definition at line 38 of file RJungleImportance.h.

6.37.3.2 `template<class T > static void RJungleImportance< T >::makePermVarImp3(RJunglePar & par, RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, DataTreeSet & oobSet, std::vector< uli_t > *& colMaskVec, std::vector< CmpldTree< T > * > & cmpldTrees, std::vector< std::pair< double, uli_t > > & permVarImpPairVec) [inline, static]`

Definition at line 389 of file RJungleImportance.h.

6.37.3.3 `template<class T > static void RJungleImportance< T >::printVarImp(RJunglePar & par, RJungleIO & io, DataFrame< T > & data, std::vector< std::pair< double, uli_t > > & freqPairs, uli_t iteration = 0) [inline, static]`

Definition at line 735 of file RJungleImportance.h.

6.37.3.4 template<class T > static void **RJungleImportance**< T >::printPermVarImp (RJunglePar & par, RJungleIO & io, DataFrame< T > & data, std::vector< std::pair< double, uli_t > > & freqPairs, uli_t iteration = 0) [inline, static]

Definition at line 743 of file **RJungleImportance.h**.

The documentation for this class was generated from the following file:

- [src/library/RJungleImportance.h](#)

6.38 **RJungleImpute**< T > Class Template Reference

```
#include <RJungleImpute.h>
```

Public Member Functions

- [RJungleImpute \(\)](#)
- [virtual ~RJungleImpute \(\)](#)

Static Public Member Functions

- [static void imputeData \(RJunglePar &par, RJungleIO &io, DataFrame< T > &data, std::vector< uli_t > *colMaskVec, Proximities< T > &proxi\)](#)
- [static void imputeTrivially \(DataFrame< T > &data\)](#)
- [static void imputeTriviallyGWA \(RJunglePar &par, DataFrame< T > &data\)](#)
- [static void imputeSNPs \(RJunglePar &par, RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, std::vector< uli_t > *colMaskFullVec\)](#)

6.38.1 Detailed Description

template<class T>class RJungleImpute< T >

Definition at line 32 of file **RJungleImpute.h**.

6.38.2 Constructor & Destructor Documentation

6.38.2.1 **template<class T > RJungleImpute< T >::RJungleImpute ()**

6.38.2.2 **template<class T > virtual RJungleImpute< T >::~RJungleImpute () [virtual]**

6.38.3 Member Function Documentation

6.38.3.1 template<class T > static void **RJungleImpute**< T >::imputeData (**RJunglePar** & *par*, **RJungleIO** & *io*, **DataFrame**< T > & *data*, std::vector< uli_t > * & *colMaskVec*, **Proximities**< T > & *prox*) [inline, static]

Definition at line 37 of file **RJungleImpute.h**.

6.38.3.2 template<class T > static void **RJungleImpute**< T >::imputeSNPs (**RJunglePar** & *par*, **RJungleIO** & *io*, **RJungleGen**< T > & *gen*, **DataFrame**< T > & *data*, std::vector< uli_t > * *colMaskFullVec*) [inline, static]

Definition at line 210 of file **RJungleImpute.h**.

6.38.3.3 template<class T > static void **RJungleImpute**< T >::imputeTrivially (**DataFrame**< T > & *data*) [inline, static]

Definition at line 162 of file **RJungleImpute.h**.

6.38.3.4 template<class T > static void **RJungleImpute**< T >::imputeTriviallyGWA (**RJunglePar** & *par*, **DataFrame**< T > & *data*) [inline, static]

Definition at line 182 of file **RJungleImpute.h**.

The documentation for this class was generated from the following file:

- [src/library/RJungleImpute.h](#)

6.39 R JungleIO Class Reference

```
#include <RJungleIO.h>
```

Public Member Functions

- [RJungleIO \(\)](#)
- [RJungleIO \(const RJungleIO &io\)](#)
- [RJungleIO \(RJunglePar &par\)](#)
- [virtual ~RJungleIO \(\)](#)
- [void close \(\)](#)
- [void open \(RJunglePar &par\)](#)
- [bool isClassTree \(RJunglePar &par\)](#)

Public Attributes

- [std::ofstream * outLog](#)
- [std::ofstream * outSummary](#)

- std::ofstream * [outPrediction](#)
- std::ofstream * [outConfusion](#)
- std::ofstream * [outConfusion2](#)
- std::ofstream * [outImportance](#)
- std::ofstream * [outImportance2](#)
- std::ofstream * [fileVerbose](#)
- std::ofstream * [outVarProximity](#)
- std::ofstream * [outSamProximity](#)
- std::ofstream * [outImputedData](#)
- std::ofstream * [outXmlJungle](#)
- std::ofstream * [outTuneMtry](#)
- std::ofstream * [outOutlier](#)
- std::ofstream * [outVotes](#)
- std::ofstream * [outOOB](#)
- std::ofstream * [outPrototypes](#)
- igzstream * [inXMLjungle](#)
- ogzstream * [outExtractData](#)
- std::ofstream * [outYaimp](#)
- std::ostream * [outVerbose](#)

6.39.1 Detailed Description

Definition at line 24 of file R JungleIO.h.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 `R JungleIO::R JungleIO()` [inline]

Definition at line 26 of file R JungleIO.h.

6.39.2.2 `R JungleIO::R JungleIO(const R JungleIO & io)` [inline]

Definition at line 52 of file R JungleIO.h.

6.39.2.3 `R JungleIO::R JungleIO(R JunglePar & par)` [inline]

Definition at line 77 of file R JungleIO.h.

6.39.2.4 `virtual R JungleIO::~R JungleIO()` [inline, virtual]

Definition at line 105 of file R JungleIO.h.

6.39.3 Member Function Documentation

6.39.3.1 `void RJungleIO::close() [inline]`

Definition at line 108 of file RJungleIO.h.

6.39.3.2 `bool RJungleIO::isClassTree(RJunglePar & par) [inline]`

Definition at line 422 of file RJungleIO.h.

6.39.3.3 `void RJungleIO::open(RJunglePar & par) [inline]`

Definition at line 241 of file RJungleIO.h.

6.39.4 Member Data Documentation

6.39.4.1 `std::ofstream* RJungleIO::fileVerbose`

Definition at line 436 of file RJungleIO.h.

6.39.4.2 `igzstream* RJungleIO::inXMLjungle`

Definition at line 448 of file RJungleIO.h.

6.39.4.3 `std::ofstream* RJungleIO::outConfusion`

Definition at line 432 of file RJungleIO.h.

6.39.4.4 `std::ofstream* RJungleIO::outConfusion2`

Definition at line 433 of file RJungleIO.h.

6.39.4.5 `ogzstream* RJungleIO::outExtractData`

Definition at line 449 of file RJungleIO.h.

6.39.4.6 `std::ofstream* RJungleIO::outImportance`

Definition at line 434 of file RJungleIO.h.

6.39.4.7 `std::ofstream* RJungleIO::outImportance2`

Definition at line 435 of file RJungleIO.h.

6.39.4.8 std::ostream* R JungleIO::outImputedData

Definition at line 441 of file R JungleIO.h.

6.39.4.9 std::ostream* R JungleIO::outLog

Definition at line 429 of file R JungleIO.h.

6.39.4.10 std::ostream* R JungleIO::outOOB

Definition at line 446 of file R JungleIO.h.

6.39.4.11 std::ostream* R JungleIO::outOutlier

Definition at line 444 of file R JungleIO.h.

6.39.4.12 std::ostream* R JungleIO::outPrediction

Definition at line 431 of file R JungleIO.h.

6.39.4.13 std::ostream* R JungleIO::outPrototypes

Definition at line 447 of file R JungleIO.h.

6.39.4.14 std::ostream* R JungleIO::outSamProximity

Definition at line 439 of file R JungleIO.h.

6.39.4.15 std::ostream* R JungleIO::outSummary

Definition at line 430 of file R JungleIO.h.

6.39.4.16 std::ostream* R JungleIO::outTuneMtry

Definition at line 443 of file R JungleIO.h.

6.39.4.17 std::ostream* R JungleIO::outVarProximity

Definition at line 437 of file R JungleIO.h.

6.39.4.18 std::ostream* R JungleIO::outVerbose

Definition at line 451 of file R JungleIO.h.

6.39.4.19 std::ofstream* R JungleIO::outVotes

Definition at line 445 of file R JungleIO.h.

6.39.4.20 std::ofstream* R JungleIO::outXmlJungle

Definition at line 442 of file R JungleIO.h.

6.39.4.21 std::ofstream* R JungleIO::outYaimp

Definition at line 450 of file R JungleIO.h.

The documentation for this class was generated from the following file:

- [src/library/R JungleIO.h](#)

6.40 R JungleOutlier Class Reference

```
#include <R JungleOutlier.h>
```

Public Member Functions

- [R JungleOutlier \(\)](#)
- [virtual ~R JungleOutlier \(\)](#)

Static Public Member Functions

- [template<class T >
static void getOutlier \(R JungleIO &io, DataFrame< T > &data, Proximities< T >
&proxi\)](#)

6.40.1 Detailed Description

Definition at line 19 of file R JungleOutlier.h.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 `RJungleOutlier::RJungleOutlier()`

Definition at line 10 of file `RJungleOutlier.cpp`.

6.40.2.2 `RJungleOutlier::~RJungleOutlier() [virtual]`

Definition at line 15 of file `RJungleOutlier.cpp`.

6.40.3 Member Function Documentation

6.40.3.1 `template<class T> static void RJungleOutlier::getOutlier(RJungleIO & io, DataFrame< T > & data, Proximities< T > & proxi) [inline, static]`

Definition at line 26 of file `RJungleOutlier.h`.

The documentation for this class was generated from the following files:

- [src/library/RJungleOutlier.h](#)
- [src/library/RJungleOutlier.cpp](#)

6.41 `RJunglePar` Struct Reference

```
#include <RJunglePar.h>
```

Public Attributes

- `char * filename`
- `char delimiter`
- `unsigned int treeType`
- `uli_t ntree`
- `uli_t ntreeMpi`
- `uli_t mtry`
- `unsigned int depVar`
- `char * depVarName`
- `uli_t nrow`
- `uli_t ncol`
- `unsigned int varNamesRow`
- `unsigned int depVarCol`
- `char * outprefix`
- `uli_t skipRow`
- `uli_t skipCol`

- int `missingcode`
- unsigned int `impMeasure`
- unsigned int `backSel`
- `uli_t` `numOfImpVar`
- bool `downsampling_flag`
- bool `verbose_flag`
- unsigned int `memMode`
- unsigned int `saveJungleType`
- char * `predict`
- int `varproximities`
- bool `summary_flag`
- bool `testlib_flag`
- char * `plugin`
- char * `colSelection`
- unsigned int `imputelt`
- bool `gwa_flag`
- bool `allcont_flag`
- bool `transpose_flag`
- bool `sampleproximities_flag`
- bool `weightsim_flag`
- bool `extractdata_flag`
- bool `yaimp_flag`
- char * `classweights`
- unsigned int `seed`
- int `nthreads`
- bool `pedfile_flag`
- char * `pluginPar`
- double `tunemtry`
- int `outlier`
- bool `votes_flag`
- bool `oob_flag`
- int `prototypes`
- int `mpi`
- int `mpild`
- int `mpiSize`
- double `condimp`
- bool `permresponse_flag`
- char `delimScale`
- double `cutoffHighLD`
- `gsl_rng` * `rng`
- char * `version`
- `uli_t` `maxTreeDepth`
- `uli_t` `targetPartitionSize`

6.41.1 Detailed Description

Definition at line 20 of file `RJunglePar.h`.

6.41.2 Member Data Documentation

6.41.2.1 bool **RJunglePar::allcont_flag**

Definition at line 52 of file RJunglePar.h.

6.41.2.2 unsigned int **RJunglePar::backSel**

Definition at line 38 of file RJunglePar.h.

6.41.2.3 char* **RJunglePar::classweights**

Definition at line 58 of file RJunglePar.h.

6.41.2.4 char* **RJunglePar::colSelection**

Definition at line 49 of file RJunglePar.h.

6.41.2.5 double **RJunglePar::condimp**

Definition at line 71 of file RJunglePar.h.

6.41.2.6 double **RJunglePar::cutoffHighLD**

Definition at line 75 of file RJunglePar.h.

6.41.2.7 char **RJunglePar::delimiter**

Definition at line 22 of file RJunglePar.h.

6.41.2.8 char **RJunglePar::delimScale**

Definition at line 74 of file RJunglePar.h.

6.41.2.9 unsigned int **RJunglePar::depVar**

Definition at line 27 of file RJunglePar.h.

6.41.2.10 unsigned int **RJunglePar::depVarCol**

Definition at line 32 of file RJunglePar.h.

6.41.2.11 char* RJunglePar::depVarName

Definition at line 28 of file RJunglePar.h.

6.41.2.12 bool RJunglePar::downsampling_flag

Definition at line 40 of file RJunglePar.h.

6.41.2.13 bool RJunglePar::extractdata_flag

Definition at line 56 of file RJunglePar.h.

6.41.2.14 char* RJunglePar::filename

Definition at line 21 of file RJunglePar.h.

6.41.2.15 bool RJunglePar::gwa_flag

Definition at line 51 of file RJunglePar.h.

6.41.2.16 unsigned int RJunglePar::impMeasure

Definition at line 37 of file RJunglePar.h.

6.41.2.17 unsigned int RJunglePar::imputelt

Definition at line 50 of file RJunglePar.h.

6.41.2.18 uli_t RJunglePar::maxTreeDepth

Definition at line 81 of file RJunglePar.h.

6.41.2.19 unsigned int RJunglePar::memMode

Definition at line 42 of file RJunglePar.h.

6.41.2.20 int RJunglePar::missingcode

Definition at line 36 of file RJunglePar.h.

6.41.2.21 int RJunglePar::mpi

Definition at line 68 of file RJunglePar.h.

6.41.2.22 int RJunglePar::mpild

Definition at line 69 of file RJunglePar.h.

6.41.2.23 int RJunglePar::mpiSize

Definition at line 70 of file RJunglePar.h.

6.41.2.24 uli_t RJunglePar::mtry

Definition at line 26 of file RJunglePar.h.

6.41.2.25 uli_t RJunglePar::ncol

Definition at line 30 of file RJunglePar.h.

6.41.2.26 uli_t RJunglePar::nrow

Definition at line 29 of file RJunglePar.h.

6.41.2.27 int RJunglePar::nthreads

Definition at line 60 of file RJunglePar.h.

6.41.2.28 uli_t RJunglePar::ntree

Definition at line 24 of file RJunglePar.h.

6.41.2.29 uli_t RJunglePar::ntreeMpi

Definition at line 25 of file RJunglePar.h.

6.41.2.30 uli_t RJunglePar::numOfImpVar

Definition at line 39 of file RJunglePar.h.

6.41.2.31 bool RJunglePar::oob_flag

Definition at line 66 of file RJunglePar.h.

6.41.2.32 int RJunglePar::outlier

Definition at line 64 of file RJunglePar.h.

6.41.2.33 char* RJunglePar::outprefix

Definition at line 33 of file RJunglePar.h.

6.41.2.34 bool RJunglePar::pedfile_flag

Definition at line 61 of file RJunglePar.h.

6.41.2.35 bool RJunglePar::permresponse_flag

Definition at line 72 of file RJunglePar.h.

6.41.2.36 char* RJunglePar::plugin

Definition at line 48 of file RJunglePar.h.

6.41.2.37 char* RJunglePar::pluginPar

Definition at line 62 of file RJunglePar.h.

6.41.2.38 char* RJunglePar::predict

Definition at line 44 of file RJunglePar.h.

6.41.2.39 int RJunglePar::prototypes

Definition at line 67 of file RJunglePar.h.

6.41.2.40 gsl_rng* RJunglePar::rng

Definition at line 77 of file RJunglePar.h.

6.41.2.41 bool RJunglePar::sampleproximities_flag

Definition at line 54 of file RJunglePar.h.

6.41.2.42 unsigned int RJunglePar::saveJungleType

Definition at line 43 of file RJunglePar.h.

6.41.2.43 unsigned int RJunglePar::seed

Definition at line 59 of file RJunglePar.h.

6.41.2.44 uli_t RJunglePar::skipCol

Definition at line 35 of file RJunglePar.h.

6.41.2.45 uli_t RJunglePar::skipRow

Definition at line 34 of file RJunglePar.h.

6.41.2.46 bool RJunglePar::summary_flag

Definition at line 46 of file RJunglePar.h.

6.41.2.47 uli_t RJunglePar::targetPartitionSize

Definition at line 82 of file RJunglePar.h.

6.41.2.48 bool RJunglePar::testlib_flag

Definition at line 47 of file RJunglePar.h.

6.41.2.49 bool RJunglePar::transpose_flag

Definition at line 53 of file RJunglePar.h.

6.41.2.50 unsigned int RJunglePar::treeType

Definition at line 23 of file RJunglePar.h.

6.41.2.51 double RJunglePar::tunemtry

Definition at line 63 of file RJunglePar.h.

6.41.2.52 unsigned int RJunglePar::varNamesRow

Definition at line 31 of file RJunglePar.h.

6.41.2.53 int RJunglePar::varproximities

Definition at line 45 of file RJunglePar.h.

6.41.2.54 bool RJunglePar::verbose_flag

Definition at line 41 of file RJunglePar.h.

6.41.2.55 char* RJunglePar::version

Definition at line 78 of file RJunglePar.h.

6.41.2.56 bool RJunglePar::votes_flag

Definition at line 65 of file RJunglePar.h.

6.41.2.57 bool RJunglePar::weightsim_flag

Definition at line 55 of file RJunglePar.h.

6.41.2.58 bool RJunglePar::yaimp_flag

Definition at line 57 of file RJunglePar.h.

The documentation for this struct was generated from the following file:

- [src/library/RJunglePar.h](#)

6.42 RJunglePrediction< T > Class Template Reference

```
#include <RJunglePrediction.h>
```

Public Member Functions

- [RJunglePrediction \(\)](#)
- virtual [~RJunglePrediction \(\)](#)

Static Public Member Functions

- static void [showPredictionCmpld \(RJungleIO &io, DataFrame< T > &data, - Prediction< T > &pred, DataTreeSet &dataSet, uli_t iteration=0\)](#)

6.42.1 Detailed Description

`template<class T>class RJunglePrediction< T >`

Definition at line 30 of file `RJunglePrediction.h`.

6.42.2 Constructor & Destructor Documentation

6.42.2.1 `template<class T > RJunglePrediction< T >::RJunglePrediction()`

6.42.2.2 `template<class T > virtual RJunglePrediction< T >::~RJunglePrediction() [virtual]`

6.42.3 Member Function Documentation

6.42.3.1 `template<class T > static void RJunglePrediction< T >::showPredictionCmpld (RJungleIO & io, DataFrame< T > & data, Prediction< T > & pred, DataTreeSet & dataSet, uli_t iteration = 0) [inline, static]`

Definition at line 35 of file `RJunglePrediction.h`.

The documentation for this class was generated from the following file:

- `src/library/RJunglePrediction.h`

6.43 RJungleProto< T > Class Template Reference

```
#include <RJungleProto.h>
```

Public Member Functions

- [RJungleProto \(\)](#)
- virtual [~RJungleProto \(\)](#)

Static Public Member Functions

- static void `getPrototypes (RJungleIO &io, RJungleGen< T > &gen, DataFrame< T > &data, std::vector< uli_t > *colMaskVec, Proximities< T > &proxi)`
- static void `countNumOfSamples (std::vector< size_t > &ranks, uli_t row, DataFrame< T > &data, std::vector< ProtCount > &protCounts)`

6.43.1 Detailed Description

`template<class T>class RJungleProto< T >`

Definition at line 35 of file RJungleProto.h.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 `template<class T > RJungleProto< T >::RJungleProto ()`

6.43.2.2 `template<class T > virtual RJungleProto< T >::~RJungleProto () [virtual]`

6.43.3 Member Function Documentation

6.43.3.1 `template<class T > static void RJungleProto< T >::countNumOfSamples (std::vector< size_t > & ranks, uli_t row, DataFrame< T > & data, std::vector< ProtCount > & protCounts) [inline, static]`

Definition at line 110 of file RJungleProto.h.

6.43.3.2 `template<class T > static void RJungleProto< T >::getPrototypes (RJungleIO & io, RJungleGen< T > & gen, DataFrame< T > & data, std::vector< uli_t > * colMaskVec, Proximities< T > & proxi) [inline, static]`

Definition at line 40 of file RJungleProto.h.

The documentation for this class was generated from the following file:

- `src/library/RJungleProto.h`

6.44 RJungleProxi< T > Class Template Reference

`#include <RJungleProxi.h>`

Public Member Functions

- [RJungleProxi \(\)](#)
- [virtual ~RJungleProxi \(\)](#)

Static Public Member Functions

- [static void finalize \(Proximities< T > &proxi\)](#)
- [static void updateProximitiesCmpld \(DataFrame< T > *data, CmpldTree< T > *cmpldTree, BuildinGenFct< T > &genFct, DataTreeSet &oobSet, std::vector< double > &oobpair, Proximities< T > &proxi, uli_t treeld\)](#)
- [static void normalizeProximitiesCmpld \(DataFrame< T > *data, std::vector< double > &oobpair, Proximities< T > &proxi\)](#)
- [static void initVarProx \(DataFrame< T > &data, Proximities< T > &varProxi\)](#)
- [static void updateVarProx \(RJungleGen< T > &gen, DataFrame< T > &data, CmpldTree< T > *cmpldTree, std::vector< uli_t > *colMask, - Proximities< T > &varProxi\)](#)
- [static void finalizeVarProx \(RJungleIO &io, DataFrame< T > &data, std::vector< uli_t > *colMask, Proximities< T > &varProxi\)](#)
- [static void updateVarProx \(RJungleIO &io, DataFrame< T > *data, CmpldTree< T > *cmpldTree, BuildinGenFct< T > &genFct, std::vector< uli_t > *colMask, Proximities< T > &proxi\)](#)
- [static void updateVarProx \(RJungleIO &io, DataFrame< T > *data, CmpldTree< T > *cmpldTree, BuildinGenFct< T > &genFct, std::vector< uli_t > *colMask, Proximities< T > &proxi\)](#)

6.44.1 Detailed Description

`template<class T>class RJungleProxi< T >`

Definition at line 29 of file RJungleProxi.h.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 `template<class T > RJungleProxi< T >::RJungleProxi ()`

6.44.2.2 `template<class T > virtual RJungleProxi< T >::~RJungleProxi () [virtual]`

6.44.3 Member Function Documentation

6.44.3.1 `template<class T > static void RJungleProxi< T >::finalize (Proximities< T > & proxi) [inline, static]`

Definition at line 35 of file RJungleProxi.h.

```
6.44.3.2 template<class T> static void RJungleProxi< T >::finalizeVarProx (
    RJungleIO & io, DataFrame< T > & data, std::vector< uli_t > * colMask,
    Proximities< T > & varProxi ) [inline, static]
```

Definition at line 143 of file RJungleProxi.h.

```
6.44.3.3 template<class T> static void RJungleProxi< T >::initVarProx ( DataFrame<
    T > & data, Proximities< T > & varProxi ) [inline, static]
```

Definition at line 117 of file RJungleProxi.h.

```
6.44.3.4 template<class T> static void RJungleProxi< T >::normalizeProximities-
    Cmpld ( DataFrame< T > * data, std::vector< double > & oobpair,
    Proximities< T > & prox ) [inline, static]
```

Definition at line 94 of file RJungleProxi.h.

```
6.44.3.5 template<class T> static void RJungleProxi< T >::updateProximitiesCmpld (
    DataFrame< T > * data, CmpldTree< T > * cmpldTree, BuildinGenFct< T >
    & genFct, DataTreeSet & oobSet, std::vector< double > & oobpair, Proximities<
    T > & prox, uli_t treeld ) [inline, static]
```

Definition at line 49 of file RJungleProxi.h.

```
6.44.3.6 template<class T> static void RJungleProxi< T >::updateVarProximities-
    Cmpld ( RJungleGen< T > & gen, DataFrame< T > & data, CmpldTree< T
    > *& cmpldTree, std::vector< uli_t > *& colMask, Proximities< T > & varProxi )
    [inline, static]
```

Definition at line 126 of file RJungleProxi.h.

```
6.44.3.7 template<class T> static void RJungleProxi< T >::updateVarProximities-
    Cmpld1 ( DataFrame< T > * data, CmpldTree< T > * cmpldTree,
    BuildinGenFct< T > & genFct, std::vector< uli_t > * colMask, Proximities< T
    > & prox ) [inline, static]
```

Definition at line 188 of file RJungleProxi.h.

```
6.44.3.8 template<class T> static void RJungleProxi< T >::updateVarProximities-
    Cmpld2 ( DataFrame< T > * data, CmpldTree< T > * cmpldTree,
    BuildinGenFct< T > & genFct, std::vector< uli_t > * colMask, Proximities< T
    > & prox ) [inline, static]
```

Definition at line 222 of file RJungleProxi.h.

The documentation for this class was generated from the following file:

- src/library/RJungleProxi.h

6.45 RJungleTuneMtry< T > Class Template Reference

```
#include <RJungleTuneMtry.h>
```

Public Member Functions

- [RJungleTuneMtry \(\)](#)
- virtual [~RJungleTuneMtry \(\)](#)

Static Public Member Functions

- static void [tuneMtry \(RJunglePar &_par, RJungleIO &_io, RJungleGen< T > &_gen, DataFrame< T > &data, std::vector< uli_t > *colMaskVec\)](#)

6.45.1 Detailed Description

```
template<class T>class RJungleTuneMtry< T >
```

Definition at line 14 of file RJungleTuneMtry.h.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 template<class T > [RJungleTuneMtry< T >::RJungleTuneMtry \(\)](#)

6.45.2.2 template<class T > virtual [RJungleTuneMtry< T >::~RJungleTuneMtry \(\)](#)
[virtual]

6.45.3 Member Function Documentation

6.45.3.1 template<class T > static void [RJungleTuneMtry< T >::tuneMtry \(RJunglePar & _par, RJungleIO & _io, RJungleGen< T > & _gen, DataFrame< T > & data, std::vector< uli_t > * colMaskVec \)](#) [inline, static]

Definition at line 22 of file RJungleTuneMtry.h.

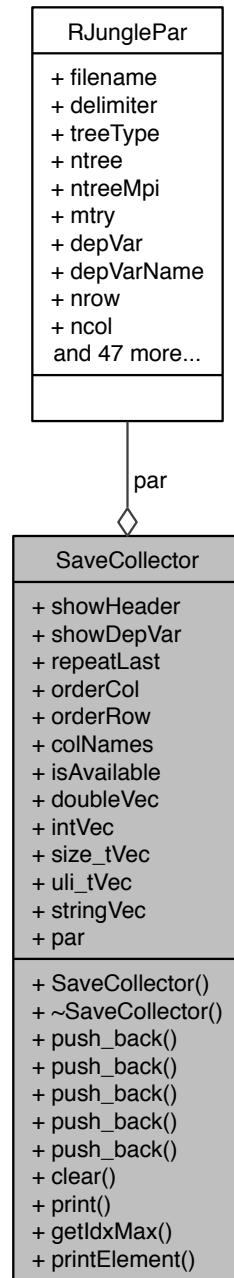
The documentation for this class was generated from the following file:

- src/library/[RJungleTuneMtry.h](#)

6.46 SaveCollector Class Reference

```
#include <SaveCollector.h>
```

Collaboration diagram for SaveCollector:



Public Member Functions

- `SaveCollector ()`
- `virtual ~SaveCollector ()`
- `void push_back (std::vector< double > *, std::string)`
- `void push_back (std::vector< int > *, std::string)`
- `void push_back (std::vector< unsigned int > *, std::string)`
- `void push_back (std::vector< unsigned long int > *, std::string)`
- `void push_back (std::vector< std::string > *, std::string)`
- `void clear ()`
- `std::ostream & print (std::ostream &) const`

Static Public Member Functions

- `template<class T >`
`static size_t getIdxMax (const std::vector< std::vector< T > * > &dataVec)`
- `template<class T >`
`static void printElement (std::ostream &os, const std::vector< T > &vec, size_t pos, bool repeatLast)`

Public Attributes

- `bool showHeader`
- `bool showDepVar`
- `bool repeatLast`
- `std::vector< size_t > orderCol`
- `std::vector< size_t > orderRow`
- `std::vector< std::string > colNames`
- `std::vector< char > isAvailable`
- `std::vector< std::vector < double > * > doubleVec`
- `std::vector< std::vector< int > * > intVec`
- `std::vector< std::vector < unsigned int > * > size_tVec`
- `std::vector< std::vector < unsigned long int > * > uli_tVec`
- `std::vector< std::vector < std::string > * > stringVec`
- `RJunglePar * par`

6.46.1 Detailed Description

Definition at line 19 of file SaveCollector.h.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 SaveCollector::SaveCollector()

Definition at line 11 of file SaveCollector.cpp.

6.46.2.2 `SaveCollector::~SaveCollector()` [virtual]

Definition at line 18 of file SaveCollector.cpp.

6.46.3 Member Function Documentation

6.46.3.1 `void SaveCollector::clear()`

Definition at line 52 of file SaveCollector.cpp.

6.46.3.2 `template<class T> static size_t SaveCollector::getIdxMax(const std::vector<std::vector<T>*> & dataVec)` [inline, static]

Definition at line 38 of file SaveCollector.h.

6.46.3.3 `std::ostream & SaveCollector::print(std::ostream & os) const`

Definition at line 61 of file SaveCollector.cpp.

6.46.3.4 `template<class T> static void SaveCollector::printElement(std::ostream & os, const std::vector<T> & vec, size_t pos, bool repeatLast)` [inline, static]

Definition at line 53 of file SaveCollector.h.

6.46.3.5 `void SaveCollector::push_back(std::vector<double> * vec, std::string name)`

Definition at line 22 of file SaveCollector.cpp.

6.46.3.6 `void SaveCollector::push_back(std::vector<int> * vec, std::string name)`

Definition at line 28 of file SaveCollector.cpp.

6.46.3.7 `void SaveCollector::push_back(std::vector<unsigned int> * vec, std::string name)`

Definition at line 34 of file SaveCollector.cpp.

6.46.3.8 `void SaveCollector::push_back(std::vector<unsigned long int> * vec, std::string name)`

Definition at line 40 of file SaveCollector.cpp.

6.46.3.9 `void SaveCollector::push_back (std::vector< std::string > * vec, std::string name)`

Definition at line 46 of file SaveCollector.cpp.

6.46.4 Member Data Documentation

6.46.4.1 `std::vector<std::string> SaveCollector::colNames`

Definition at line 73 of file SaveCollector.h.

6.46.4.2 `std::vector<std::vector<double> *> SaveCollector::doubleVec`

Definition at line 76 of file SaveCollector.h.

6.46.4.3 `std::vector<std::vector<int> *> SaveCollector::intVec`

Definition at line 77 of file SaveCollector.h.

6.46.4.4 `std::vector<char> SaveCollector::isAvailable`

Definition at line 74 of file SaveCollector.h.

6.46.4.5 `std::vector<size_t> SaveCollector::orderCol`

Definition at line 71 of file SaveCollector.h.

6.46.4.6 `std::vector<size_t> SaveCollector::orderRow`

Definition at line 72 of file SaveCollector.h.

6.46.4.7 `RJunglePar* SaveCollector::par`

Definition at line 82 of file SaveCollector.h.

6.46.4.8 `bool SaveCollector::repeatLast`

Definition at line 68 of file SaveCollector.h.

6.46.4.9 `bool SaveCollector::showDepVar`

Definition at line 67 of file SaveCollector.h.

6.46.4.10 bool SaveCollector::showHeader

Definition at line 66 of file SaveCollector.h.

6.46.4.11 std::vector<std::vector<unsigned int> *> SaveCollector::size_tVec

Definition at line 78 of file SaveCollector.h.

6.46.4.12 std::vector<std::vector<std::string> *> SaveCollector::stringVec

Definition at line 80 of file SaveCollector.h.

6.46.4.13 std::vector<std::vector<unsigned long int> *> SaveCollector::uli_tVec

Definition at line 79 of file SaveCollector.h.

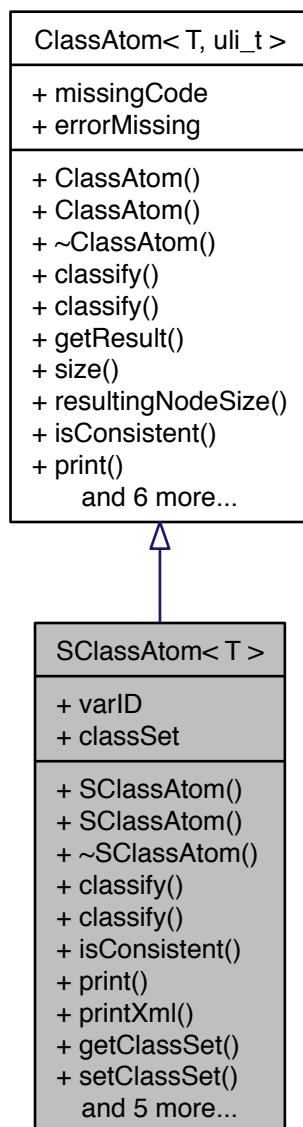
The documentation for this class was generated from the following files:

- src/library/[SaveCollector.h](#)
- src/library/[SaveCollector.cpp](#)

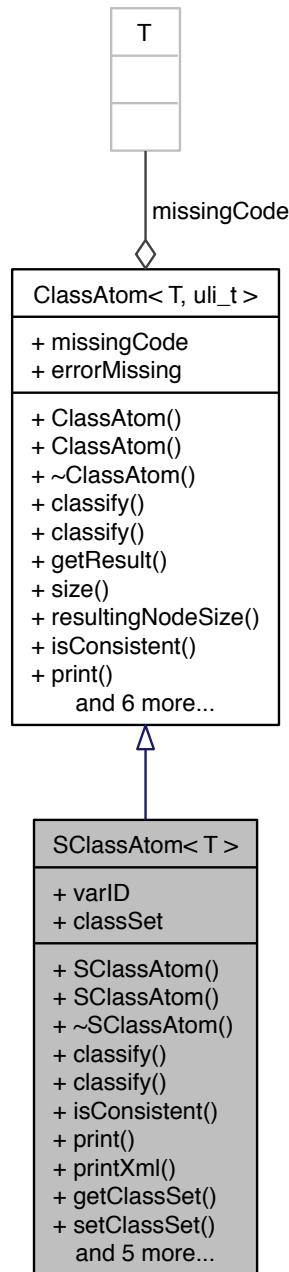
6.47 SClassAtom< T > Class Template Reference

```
#include <SClassAtom.h>
```

Inheritance diagram for SClassAtom< T >:



Collaboration diagram for SClassAtom< T >:



Public Member Functions

- `SClassAtom ()`
- `SClassAtom (std::vector< T > &v)`
- virtual `~SClassAtom (void)`
- virtual `uli_t classify (Tvector &sample) const`
- virtual `uli_t classify (T *sample) const`

Classifies an input value.
- `bool isConsistent () const`

Returns if the classifier is consistent.
- virtual `std::ostream & print (std::ostream &os) const`

Prints the classifier to output stream.
- virtual `void printXml (std::ostream &os) const`

Prints the classifier to output stream in XML format.
- `const std::vector< T > & getClassSet () const`
- `void setClassSet (std::vector< T > &set)`
- virtual `uli_t resultingNodeSize () const`

Returns the size of resulting node.
- `bool operator== (const SClassAtom< T > &sClassAtom)`
- virtual `uli_t getType () const`

Returns the type of current classifier.
- `void setVarID (uli_t varID)`
- `uli_t getVarID () const`
- virtual `bool isThereVarID (uli_t varID) const`

Return if classifier has got a specific variable ID.

Public Attributes

- `uli_t varID`
- `std::vector< T > classSet`

6.47.1 Detailed Description

`template<class T>class SClassAtom< T >`

Definition at line 29 of file SClassAtom.h.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 `template<class T> SClassAtom< T >::SClassAtom () [inline]`

Definition at line 31 of file SClassAtom.h.

**6.47.2.2 template<class T> SClassAtom< T >::SClassAtom (std::vector< T > & v)
[inline]**

Definition at line 32 of file SClassAtom.h.

**6.47.2.3 template<class T> virtual SClassAtom< T >::~SClassAtom (void)
[inline, virtual]**

Definition at line 34 of file SClassAtom.h.

6.47.3 Member Function Documentation

6.47.3.1 template<class T> virtual uli_t SClassAtom< T >::classify (Tvector & sample) const [inline, virtual]

Definition at line 39 of file SClassAtom.h.

6.47.3.2 template<class T> virtual uli_t SClassAtom< T >::classify (T * vecIn) const [inline, virtual]

Classifies an input value.

Parameters

<i>vecIn</i>	Sample to be classified
--------------	-------------------------

Returns

Classification value

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 53 of file SClassAtom.h.

6.47.3.3 template<class T> const std::vector< T >& SClassAtom< T >::getClassSet () const [inline]

Definition at line 99 of file SClassAtom.h.

6.47.3.4 template<class T> virtual uli_t SClassAtom< T >::getType () const [inline, virtual]

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 110 of file SClassAtom.h.

6.47.3.5 template<class T> uli_t SClassAtom< T >::getVarID() const [inline]

Definition at line 113 of file SClassAtom.h.

6.47.3.6 template<class T> bool SClassAtom< T >::isConsistent() const [inline, virtual]

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 71 of file SClassAtom.h.

6.47.3.7 template<class T> virtual bool SClassAtom< T >::isThereVarID(uli_t varID) const [inline, virtual]

Return if classifier has got a specific variable ID.

Parameters

<i>varID</i>	ID of variable.
--------------	-----------------

Returns

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 115 of file SClassAtom.h.

6.47.3.8 template<class T> bool SClassAtom< T >::operator==(const SClassAtom< T > & sClassAtom) [inline]

Definition at line 106 of file SClassAtom.h.

6.47.3.9 `template<class T> virtual std::ostream& SClassAtom< T >::print(std::ostream & os) const [inline, virtual]`

Prints the classifier to output stream.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 75 of file SClassAtom.h.

6.47.3.10 `template<class T> virtual void SClassAtom< T >::printXml(std::ostream & os) const [inline, virtual]`

Prints the classifier to output stream in XML format.

Parameters

<code>os</code>	Output stream.
-----------------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 83 of file SClassAtom.h.

6.47.3.11 `template<class T> virtual uli_t SClassAtom< T >::resultingNodeSize() const [inline, virtual]`

Returns the size of resulting node.

Returns

Size of resulting node.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 102 of file SClassAtom.h.

6.47.3.12 `template<class T> void SClassAtom< T >::setClassSet(std::vector< T > & set) [inline]`

Definition at line 100 of file SClassAtom.h.

6.47.3.13 `template<class T> void SClassAtom< T >::setVarID (uli_t varID)`
[inline]

Definition at line 112 of file SClassAtom.h.

6.47.4 Member Data Documentation

6.47.4.1 `template<class T> std::vector<T> SClassAtom< T >::classSet`

Definition at line 118 of file SClassAtom.h.

6.47.4.2 `template<class T> uli_t SClassAtom< T >::varID`

Definition at line 117 of file SClassAtom.h.

The documentation for this class was generated from the following file:

- [src/library/SClassAtom.h](#)

6.48 StringIterator Class Reference

A class for iterating through tokens from a c-string.

```
#include <iolines.h>
```

Public Member Functions

- `StringIterator (char delimiters[]=" \n\r")`
- `StringIterator ()`
- `virtual ~StringIterator ()`
- `void setDelimiters (char delimiters[]=" \n\r")`
- `char * init (char *theString)`
Initializes the token.
- `char * init (std::string theCppString)`
Initializes the token.
- `bool empty () const`
- `char * next ()`
Jump to next token.
- `char * next (char *delims)`
Jump to next token.
- `char * mystrtok (char *s1, const char *delimit)`
Get next string.

Public Attributes

- `char * token`
The token.
- `char * strTmp`
temp string
- `char lastDelim`
last delimiter
- `char * myLastToken`
last token

Private Attributes

- `char delimiters [10]`

6.48.1 Detailed Description

A class for iterating through tokens from a c-string.

Author

Roman Pahl (original) and Daniel F Schwarz (modified heavily)

Definition at line 18 of file ioLines.h.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 `StringIterator::StringIterator (char delimiters[] = " \n\r") [inline]`

Definition at line 20 of file ioLines.h.

6.48.2.2 `StringIterator::StringIterator () [inline]`

Definition at line 25 of file ioLines.h.

6.48.2.3 `virtual StringIterator::~StringIterator () [inline, virtual]`

Definition at line 29 of file ioLines.h.

6.48.3 Member Function Documentation

6.48.3.1 `bool StringIterator::empty () const [inline]`

Returns

true if token is empty, else false

Definition at line 69 of file ioLines.h.

6.48.3.2 char* StringIterator::init (char * *theString*) [inline]

Initializes the token.

Parameters

<i>theString</i>	the string to extract from
------------------	----------------------------

Definition at line 43 of file ioLines.h.

6.48.3.3 char* StringIterator::init (std::string *theCppString*) [inline]

Initializes the token.

Parameters

<i>theString</i>	the string to extract from
------------------	----------------------------

Definition at line 53 of file ioLines.h.

6.48.3.4 char* StringIterator::mystrtok (char * *s1*, const char * *delimit*) [inline]

Get next string.

Definition at line 86 of file ioLines.h.

6.48.3.5 char* StringIterator::next () [inline]

Jump to next token.

Definition at line 74 of file ioLines.h.

6.48.3.6 char* StringIterator::next (char * *delims*) [inline]

Jump to next token.

Definition at line 80 of file ioLines.h.

6.48.3.7 void StringIterator::setDelimiters (char *delimiters*[] = " \n\r") [inline]

Definition at line 35 of file ioLines.h.

6.48.4 Member Data Documentation

6.48.4.1 `char StringIterator::delimiters[10]` [private]

Definition at line 127 of file ioLines.h.

6.48.4.2 `char StringIterator::lastDelim`

last delimiter

Definition at line 120 of file ioLines.h.

6.48.4.3 `char* StringIterator::myLastToken`

last token

Definition at line 123 of file ioLines.h.

6.48.4.4 `char* StringIterator::strTmp`

temp string

Definition at line 117 of file ioLines.h.

6.48.4.5 `char* StringIterator::token`

The token.

Definition at line 114 of file ioLines.h.

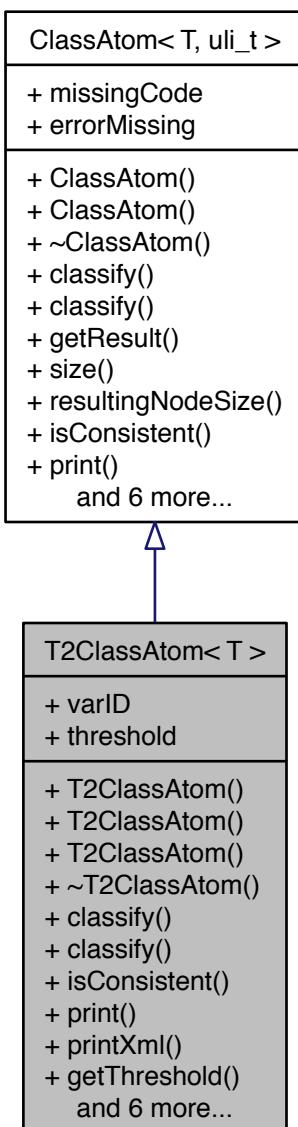
The documentation for this class was generated from the following file:

- [src/library/ioLines.h](#)

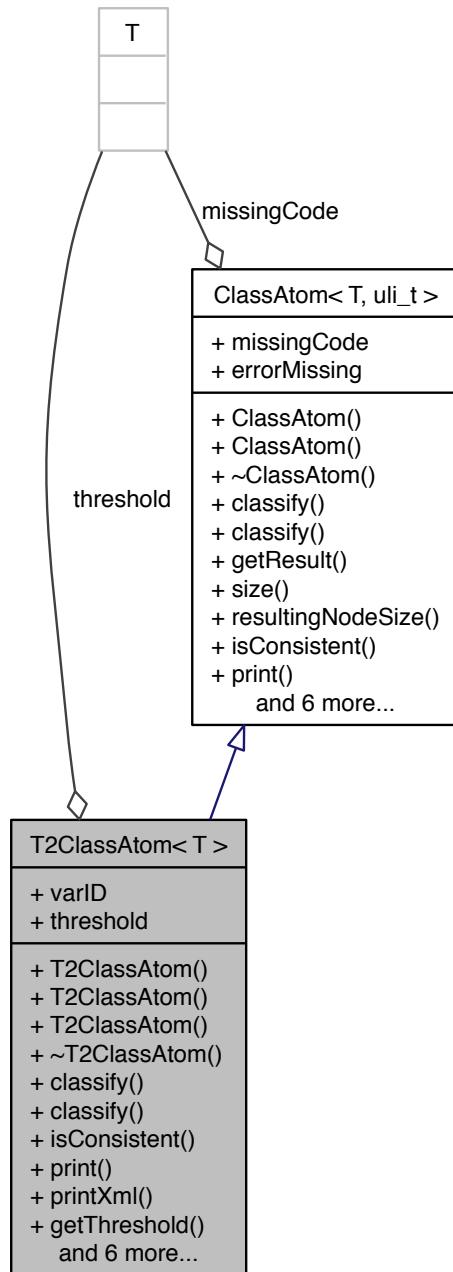
6.49 T2ClassAtom< T > Class Template Reference

```
#include <T2ClassAtom.h>
```

Inheritance diagram for T2ClassAtom< T >:



Collaboration diagram for T2ClassAtom< T >:



Public Member Functions

- `T2ClassAtom ()`
- `T2ClassAtom (const T &v)`
- `T2ClassAtom (T val)`
- `virtual ~T2ClassAtom (void)`
- `virtual uli_t classify (Tvector &sample) const`
- `virtual uli_t classify (T *sample) const`

Classifies an input value.
- `bool isConsistent () const`

Returns if the classifier is consistent.
- `virtual std::ostream & print (std::ostream &os) const`

Prints the classifier to output stream.
- `virtual void printXml (std::ostream &os) const`

Prints the classifier to output stream in XML format.
- `const T getThreshold () const`
- `void setThreshold (T val)`
- `virtual uli_t resultingNodeSize () const`

Returns the size of resulting node.
- `bool operator== (const T2ClassAtom< T > &t2ClassAtom)`
- `virtual uli_t getType () const`

Returns the type of current classifier.
- `void setVarID (uli_t varID)`
- `uli_t getVarID () const`
- `virtual bool isThereVarID (uli_t varID) const`

Return if classifier has got a specific variable ID.

Public Attributes

- `uli_t varID`
- `T threshold`

6.49.1 Detailed Description

template<class T>class T2ClassAtom< T >

Definition at line 29 of file T2ClassAtom.h.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 template<class T> T2ClassAtom< T >::T2ClassAtom() [inline]

Definition at line 31 of file T2ClassAtom.h.

6.49.2.2 template<class T> T2ClassAtom< T >::T2ClassAtom (const T & v)
[inline]

Definition at line 32 of file T2ClassAtom.h.

6.49.2.3 template<class T> T2ClassAtom< T >::T2ClassAtom (T val) [inline]

Definition at line 33 of file T2ClassAtom.h.

6.49.2.4 template<class T> virtual T2ClassAtom< T >::~T2ClassAtom (void)
[inline, virtual]

Definition at line 35 of file T2ClassAtom.h.

6.49.3 Member Function Documentation

6.49.3.1 template<class T> virtual uli_t T2ClassAtom< T >::classify (Tvector & sample) const [inline, virtual]

Definition at line 40 of file T2ClassAtom.h.

6.49.3.2 template<class T> virtual uli_t T2ClassAtom< T >::classify (T * vecIn) const
[inline, virtual]

Classifies an input value.

Parameters

vecIn	Sample to be classified
-------	-------------------------

Returns

Classification value

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 46 of file T2ClassAtom.h.

6.49.3.3 template<class T> const T T2ClassAtom< T >::getThreshold () const
[inline]

Definition at line 77 of file T2ClassAtom.h.

6.49.3.4 template<class T> virtual uli_t T2ClassAtom< T >::getType() const
[inline, virtual]

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 88 of file T2ClassAtom.h.

6.49.3.5 template<class T> uli_t T2ClassAtom< T >::getVarID() const [inline]

Definition at line 91 of file T2ClassAtom.h.

6.49.3.6 template<class T> bool T2ClassAtom< T >::isConsistent() const
[inline, virtual]

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 56 of file T2ClassAtom.h.

6.49.3.7 template<class T> virtual bool T2ClassAtom< T >::isThereVarID(uli_t varID
) const [inline, virtual]

Return if classifier has got a specific variable ID.

Parameters

varID	ID of variable.
-------	-----------------

Returns

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 93 of file T2ClassAtom.h.

6.49.3.8 template<class T> bool T2ClassAtom< T >::operator==(const T2ClassAtom< T > & t2ClassAtom) [inline]

Definition at line 84 of file T2ClassAtom.h.

6.49.3.9 template<class T> virtual std::ostream& T2ClassAtom< T >::print(std::ostream & os) const [inline, virtual]

Prints the classifier to output stream.

Parameters

os	Output stream.
----	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 60 of file T2ClassAtom.h.

6.49.3.10 template<class T> virtual void T2ClassAtom< T >::printXml(std::ostream & os) const [inline, virtual]

Prints the classifier to output stream in XML format.

Parameters

os	Output stream.
----	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 66 of file T2ClassAtom.h.

6.49.3.11 template<class T> virtual uli_t T2ClassAtom< T >::resultingNodeSize() const [inline, virtual]

Returns the size of resulting node.

Returns

Size of resulting node.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 80 of file T2ClassAtom.h.

6.49.3.12 template<class T> void T2ClassAtom< T >::setThreshold (T val)
[inline]

Definition at line 78 of file T2ClassAtom.h.

6.49.3.13 template<class T> void T2ClassAtom< T >::setVarID (uli_t varID)
[inline]

Definition at line 90 of file T2ClassAtom.h.

6.49.4 Member Data Documentation

6.49.4.1 template<class T> T T2ClassAtom< T >::threshold

Definition at line 96 of file T2ClassAtom.h.

6.49.4.2 template<class T> uli_t T2ClassAtom< T >::varID

Definition at line 95 of file T2ClassAtom.h.

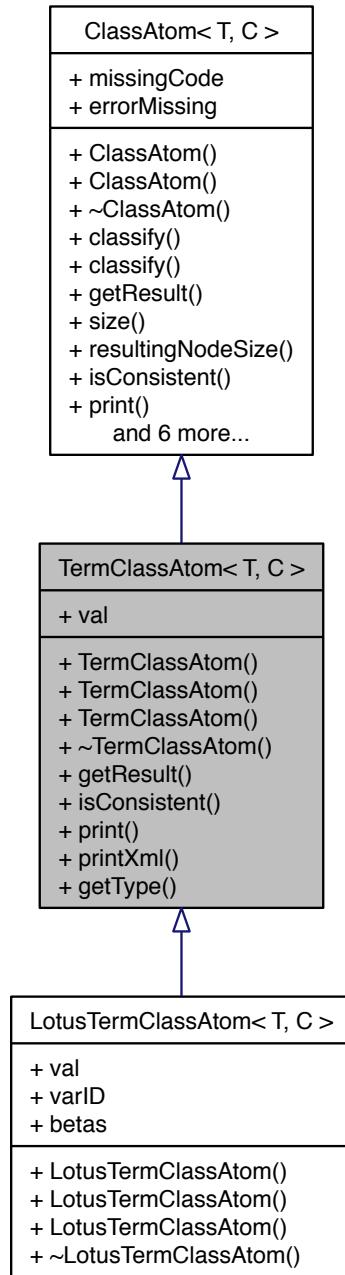
The documentation for this class was generated from the following file:

- [src/library/T2ClassAtom.h](#)

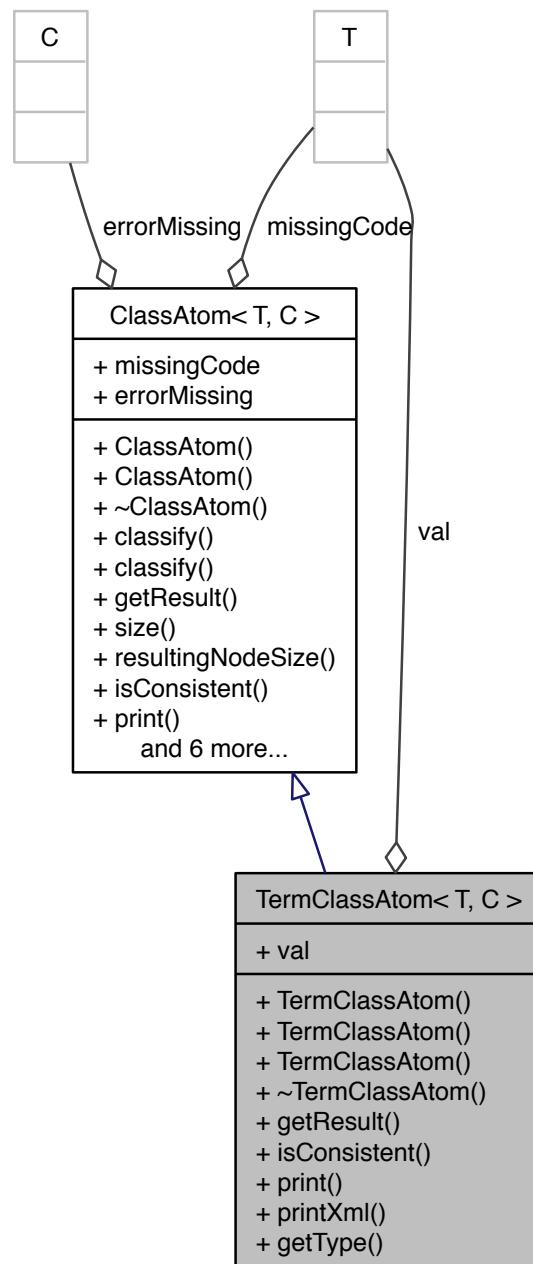
6.50 TermClassAtom< T, C > Class Template Reference

```
#include <TermClassAtom.h>
```

Inheritance diagram for TermClassAtom< T, C >:



Collaboration diagram for TermClassAtom< T, C >:



Public Member Functions

- [TermClassAtom \(\)](#)
- [TermClassAtom \(const TermClassAtom< T, C > &termCA\)](#)
- [TermClassAtom \(T val\)](#)
- [virtual ~TermClassAtom \(void\)](#)
- [virtual T getResult \(\) const](#)
Returns the value of this classifier.
- [virtual bool isConsistent \(\) const](#)
Returns if the classifier is consistent.
- [virtual std::ostream & print \(std::ostream &os\) const](#)
Prints the classifier to output stream.
- [virtual void printXml \(std::ostream &os\) const](#)
Prints the classifier to output stream in XML format.
- [virtual uli_t getType \(\) const](#)
Returns the type of current classifier.

Public Attributes

- [T val](#)

6.50.1 Detailed Description

`template<class T, class C>class TermClassAtom< T, C >`

Definition at line 11 of file TermClassAtom.h.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 `template<class T, class C> TermClassAtom< T, C >::TermClassAtom ()`
`[inline]`

Definition at line 13 of file TermClassAtom.h.

6.50.2.2 `template<class T, class C> TermClassAtom< T, C >::TermClassAtom (const`
`TermClassAtom< T, C > & termCA)` `[inline]`

Definition at line 14 of file TermClassAtom.h.

6.50.2.3 `template<class T, class C> TermClassAtom< T, C >::TermClassAtom (T val)`
`[inline]`

Definition at line 15 of file TermClassAtom.h.

```
6.50.2.4 template<class T, class C> virtual TermClassAtom< T, C >::~TermClassAtom  
( void ) [inline, virtual]
```

Definition at line 17 of file TermClassAtom.h.

6.50.3 Member Function Documentation

```
6.50.3.1 template<class T, class C> virtual T TermClassAtom< T, C >::getResult( )  
const [inline, virtual]
```

Returns the value of this classifier.

Returns

Classifier value

Reimplemented from [ClassAtom< T, C >](#).

Definition at line 20 of file TermClassAtom.h.

```
6.50.3.2 template<class T, class C> virtual uli_t TermClassAtom< T, C >::getType( )  
const [inline, virtual]
```

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented from [ClassAtom< T, C >](#).

Definition at line 44 of file TermClassAtom.h.

```
6.50.3.3 template<class T, class C> virtual bool TermClassAtom< T, C >::isConsistent  
( ) const [inline, virtual]
```

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented from [ClassAtom< T, C >](#).

Definition at line 24 of file TermClassAtom.h.

```
6.50.3.4 template<class T, class C> virtual std::ostream& TermClassAtom< T, C >::print(   
std::ostream & os ) const [inline, virtual]
```

Prints the classifier to output stream.

Parameters

<i>os</i>	Output stream.
-----------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, C >](#).

Definition at line 32 of file TermClassAtom.h.

6.50.3.5 template<class T, class C> virtual void TermClassAtom< T, C >::printXml (std::ostream & *os*) const [inline, virtual]

Prints the classifier to output stream in XML format.

Parameters

<i>os</i>	Output stream.
-----------	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, C >](#).

Definition at line 37 of file TermClassAtom.h.

6.50.4 Member Data Documentation

6.50.4.1 template<class T, class C> T TermClassAtom< T, C >::val

Reimplemented in [LotusTermClassAtom< T, C >](#).

Definition at line 46 of file TermClassAtom.h.

The documentation for this class was generated from the following file:

- [src/library/TermClassAtom.h](#)

6.51 TestClass Class Reference

```
#include <TestClass.h>
```

Public Member Functions

- [TestClass \(\)](#)

- virtual ~TestClass ()
 - void `testT2ClassAtom` (void) const
 - void `testTMClassAtom` (void) const
 - void `testSClassAtom` (void) const
 - void `testNode` (void) const
 - void `testDataFrame` (void) const
 - void `testDataFrameAndNode` (void) const
 - void `testFind` (void) const
 - void `testPurity` () const
 - void `testGini1` () const
 - void `testGini2` () const
 - void `testTree` () const
 - void `testJTreeCtrl4` () const
 - void `testJTreeCtrl5` () const
 - void `testRandomJungleCtrl1` () const
 - void `testRandomJungleCtrl2` () const
 - void `testCI` () const
 - void `testMvt` () const
 - void `testC_maxabsConditionalPvalue` () const
 - void `helpers` (void) const
 - void `lr` (void) const
 - void `train_save` (char *savename, lr_predict *lrp) const
- Test logistic regression.*
- lr_predict * `mk_train_lr_predict` (spardat *sp, dym *factors, dyv *outputs, lr_options *opts) const

6.51.1 Detailed Description

Definition at line 26 of file TestClass.h.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 TestClass::TestClass ()

Definition at line 22 of file TestClass.cpp.

6.51.2.2 TestClass::~TestClass () [virtual]

Definition at line 24 of file TestClass.cpp.

6.51.3 Member Function Documentation

6.51.3.1 void TestClass::helpers (void) const

Definition at line 92 of file TestClass.cpp.

6.51.3.2 void **TestClass::lr**(void) const

Definition at line 48 of file TestClass.cpp.

6.51.3.3 lr_predict * **TestClass::mk_train_lr_predict**(spardat * *sp*, dym * *factors*, dyv * *outputs*, lr_options * *opts*) const

Definition at line 37 of file TestClass.cpp.

6.51.3.4 void **TestClass::testC_maxabsConditionalPvalue**(void) const

Definition at line 378 of file TestClass.cpp.

6.51.3.5 void **TestClass::testCI**(void) const

Definition at line 523 of file TestClass.cpp.

6.51.3.6 void **TestClass::testDataFrame**(void) const

6.51.3.7 void **TestClass::testDataFrameAndNode**(void) const

6.51.3.8 void **TestClass::testFind**(void) const

6.51.3.9 void **TestClass::testGini1**() const

6.51.3.10 void **TestClass::testGini2**() const

6.51.3.11 void **TestClass::testJTreeCtrl4**() const

6.51.3.12 void **TestClass::testJTreeCtrl5**() const

Definition at line 988 of file TestClass.cpp.

6.51.3.13 void **TestClass::testMvt**(void) const

Definition at line 444 of file TestClass.cpp.

6.51.3.14 void **TestClass::testNode**(void) const

6.51.3.15 void **TestClass::testPurity**() const

6.51.3.16 void **TestClass::testRandomJungleCtrl1**() const

6.51.3.17 void **TestClass::testRandomJungleCtrl2**() const

6.51.3.18 void TestClass::testSClassAtom (void) const

Definition at line 816 of file TestClass.cpp.

6.51.3.19 void TestClass::testT2ClassAtom (void) const

6.51.3.20 void TestClass::testTMClassAtom (void) const

6.51.3.21 void TestClass::testTree () const

6.51.3.22 void TestClass::train_save (char * savename, lr_predict * lrp) const

Test logistic regression.

Definition at line 30 of file TestClass.cpp.

The documentation for this class was generated from the following files:

- [src/library/TestClass.h](#)
- [src/library/TestClass.cpp](#)

6.52 TimeProf Class Reference

```
#include <TimeProf.h>
```

Public Member Functions

- [**TimeProf \(\)**](#)
Constructor.
- [**virtual ~TimeProf \(\)**](#)
Destructor.
- [**void start \(std::string name\)**](#)
Start the stop watch.
- [**void stop \(std::string name\)**](#)
Stop the watch and add time difference to total time.
- [**void writeToFile \(std::ostream &io\)**](#)
Streams for output profiled informations.

Public Attributes

- [**std::map< std::string, Profiler > profilers**](#)
Map of all time trigger and profiler.

6.52.1 Detailed Description

Definition at line 31 of file TimeProf.h.

6.52.2 Constructor & Destructor Documentation

6.52.2.1 TimeProf::TimeProf()

Constructor.

Definition at line 4 of file TimeProf.cpp.

6.52.2.2 TimeProf::~TimeProf() [virtual]

Destructor.

Definition at line 7 of file TimeProf.cpp.

6.52.3 Member Function Documentation

6.52.3.1 void TimeProf::start(std::string name)

Start the stop watch.

Definition at line 11 of file TimeProf.cpp.

6.52.3.2 void TimeProf::stop(std::string name)

Stop the watch and add time difference to total time.

Definition at line 26 of file TimeProf.cpp.

6.52.3.3 void TimeProf::writeToFile(std::ostream & io)

Streams for output profiled informations.

Definition at line 39 of file TimeProf.cpp.

6.52.4 Member Data Documentation

6.52.4.1 std::map<std::string, Profiler> TimeProf::profilers

Map of all time trigger and profiler.

Definition at line 58 of file TimeProf.h.

The documentation for this class was generated from the following files:

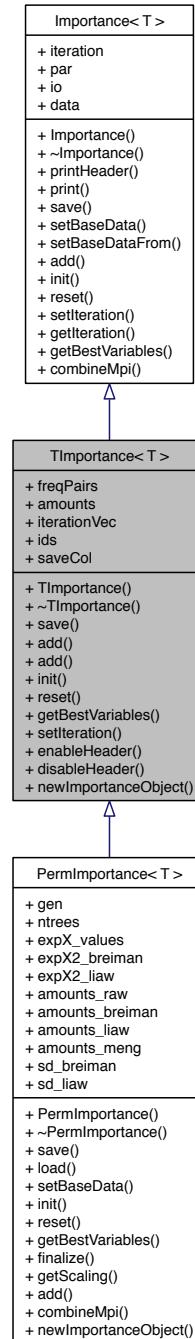
- [src/library/TimeProf.h](#)

- [src/library/TimeProf.cpp](#)

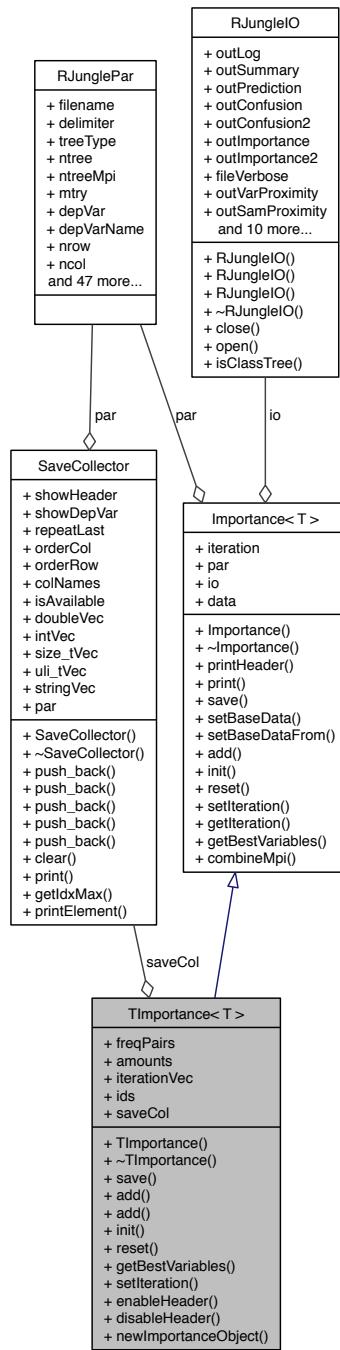
6.53 TImportance< T > Class Template Reference

```
#include <TImportance.h>
```

Inheritance diagram for TImportance< T >:



Collaboration diagram for TImportance< T >:



Public Member Functions

- [TImportance \(\)](#)
- [virtual ~TImportance \(\)](#)
- [virtual void save \(\)](#)
- [virtual void add \(uli_t varID, double val\)](#)
- [virtual void add \(Importance< T > *impx\)](#)
- [virtual void init \(\)](#)
- [virtual void reset \(\)](#)
- [virtual void getBestVariables \(size_t size, std::vector< uli_t > &outVec\)](#)
- [void setIteration \(uli_t it\)](#)
- [void enableHeader \(\)](#)
- [void disableHeader \(\)](#)

Static Public Member Functions

- [static Importance< T > * newImportanceObject \(\)](#)

Public Attributes

- [std::vector< std::pair< double, uli_t > > freqPairs](#)
- [std::vector< double > amounts](#)
- [std::vector< uli_t > iterationVec](#)
- [std::vector< size_t > ids](#)
- [SaveCollector saveCol](#)

6.53.1 Detailed Description

`template<class T>class TImportance< T >`

Definition at line 18 of file TImportance.h.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 template<class T> TImportance< T >::TImportance () [inline]

Definition at line 20 of file TImportance.h.

6.53.2.2 template<class T> virtual TImportance< T >::~TImportance () [inline, virtual]

Definition at line 24 of file TImportance.h.

6.53.3 Member Function Documentation

6.53.3.1 `template<class T> virtual void TImportance< T >::add(uli_t varID, double val) [inline, virtual]`

Definition at line 37 of file TImportance.h.

6.53.3.2 `template<class T> virtual void TImportance< T >::add(Importance< T > * impX) [inline, virtual]`

Reimplemented from [Importance< T >](#).

Definition at line 41 of file TImportance.h.

6.53.3.3 `template<class T> void TImportance< T >::disableHeader() [inline]`

Definition at line 111 of file TImportance.h.

6.53.3.4 `template<class T> void TImportance< T >::enableHeader() [inline]`

Definition at line 107 of file TImportance.h.

6.53.3.5 `template<class T> virtual void TImportance< T >::getBestVariables(size_t size, std::vector< uli_t > & outVec) [inline, virtual]`

Reimplemented from [Importance< T >](#).

Reimplemented in [PermlImportance< T >](#).

Definition at line 74 of file TImportance.h.

6.53.3.6 `template<class T> virtual void TImportance< T >::init() [inline, virtual]`

Reimplemented from [Importance< T >](#).

Reimplemented in [PermlImportance< T >](#).

Definition at line 47 of file TImportance.h.

6.53.3.7 `template<class T> static Importance< T >* TImportance< T >::newImportanceObject() [inline, static]`

Reimplemented in [PermlImportance< T >](#).

Definition at line 70 of file TImportance.h.

6.53.3.8 `template<class T> virtual void TImportance< T >::reset() [inline, virtual]`

Reimplemented from [Importance< T >](#).

Reimplemented in [PermlImportance< T >](#).

Definition at line 65 of file TImportance.h.

6.53.3.9 `template<class T> virtual void TImportance< T >::save() [inline, virtual]`

Reimplemented from [Importance< T >](#).

Reimplemented in [PermlImportance< T >](#).

Definition at line 28 of file TImportance.h.

6.53.3.10 `template<class T> void TImportance< T >::setIteration(uli_t it) [inline, virtual]`

Reimplemented from [Importance< T >](#).

Definition at line 98 of file TImportance.h.

6.53.4 Member Data Documentation

6.53.4.1 `template<class T> std::vector<double> TImportance< T >::amounts`

Definition at line 116 of file TImportance.h.

6.53.4.2 `template<class T> std::vector<std::pair<double, uli_t> > TImportance< T >::freqPairs`

Definition at line 115 of file TImportance.h.

6.53.4.3 `template<class T> std::vector<size_t> TImportance< T >::ids`

Definition at line 119 of file TImportance.h.

6.53.4.4 `template<class T> std::vector<uli_t> TImportance< T >::iterationVec`

Definition at line 118 of file TImportance.h.

6.53.4.5 `template<class T> SaveCollector TImportance< T >::saveCol`

Definition at line 121 of file TImportance.h.

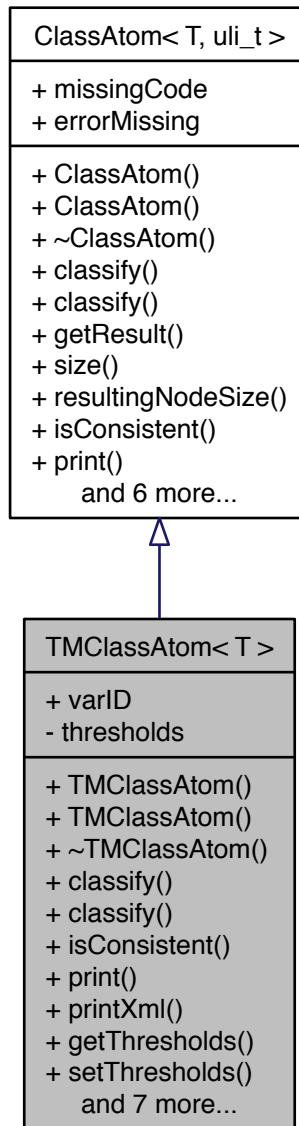
The documentation for this class was generated from the following file:

- [src/library/TImportance.h](#)

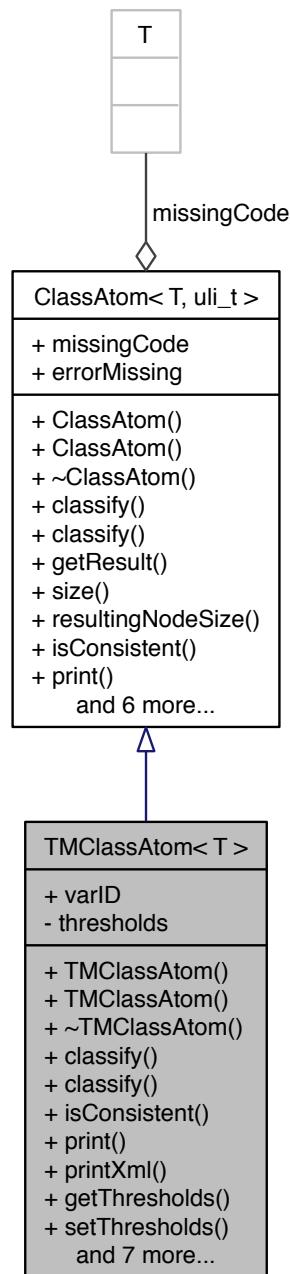
6.54 TMClassAtom< T > Class Template Reference

```
#include <TMClassAtom.h>
```

Inheritance diagram for TMClassAtom< T >:



Collaboration diagram for TMClassAtom< T >:



Public Member Functions

- [TMClassAtom \(\)](#)
- [TMClassAtom \(std::vector< T > &vec\)](#)
- virtual [~TMClassAtom \(void\)](#)
- virtual [uli_t classify \(Tvector &sample\) const](#)
- virtual [uli_t classify \(T *sample\) const](#)

Classifies an input value.
- [bool isConsistent \(\) const](#)

Returns if the classifier is consistent.
- virtual [std::ostream & print \(std::ostream &os\) const](#)

Prints the classifier to output stream.
- virtual [void printXml \(std::ostream &os\) const](#)

Prints the classifier to output stream in XML format.
- [Tvector & getThresholds \(\)](#)
- void [setThresholds \(Tvector vec\)](#)
- void [setThresholdsNoSort \(Tvector vec\)](#)
- virtual [uli_t resultingNodeSize \(\) const](#)

Returns the size of resulting node.
- virtual [uli_t size \(\) const](#)

Returns the size of classifier.
- [bool operator== \(const TMClassAtom< T > &tMClassAtom\)](#)

Returns the type of current classifier.
- virtual [uli_t getType \(\) const](#)

Return if classifier has got a specific variable ID.

Public Attributes

- [uli_t varID](#)

Private Attributes

- [Tvector thresholds](#)

6.54.1 Detailed Description

template<class T>class TMClassAtom< T >

Definition at line 34 of file TMClassAtom.h.

6.54.2 Constructor & Destructor Documentation

6.54.2.1 `template<class T> TMClassAtom< T >::TMClassAtom() [inline]`

Definition at line 36 of file TMClassAtom.h.

6.54.2.2 `template<class T> TMClassAtom< T >::TMClassAtom(std::vector< T > & vec) [inline]`

Definition at line 37 of file TMClassAtom.h.

6.54.2.3 `template<class T> virtual TMClassAtom< T >::~TMClassAtom(void) [inline, virtual]`

Definition at line 39 of file TMClassAtom.h.

6.54.3 Member Function Documentation

6.54.3.1 `template<class T> virtual uli_t TMClassAtom< T >::classify(Tvector & sample) const [inline, virtual]`

Definition at line 44 of file TMClassAtom.h.

6.54.3.2 `template<class T> virtual uli_t TMClassAtom< T >::classify(T * vecIn) const [inline, virtual]`

Classifies an input value.

Parameters

<code>vecIn</code>	Sample to be classified
--------------------	-------------------------

Returns

Classification value

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 70 of file TMClassAtom.h.

6.54.3.3 `template<class T> Tvector& TMClassAtom< T >::getThresholds() [inline]`

Definition at line 142 of file TMClassAtom.h.

6.54.3.4 **template<class T> virtual uli_t TMClassAtom< T >::getType () const [inline, virtual]**

Returns the type of current classifier.

Returns

Classifier type.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 164 of file TMClassAtom.h.

6.54.3.5 **template<class T> uli_t TMClassAtom< T >::getVarID () const [inline]**

Definition at line 167 of file TMClassAtom.h.

6.54.3.6 **template<class T> bool TMClassAtom< T >::isConsistent () const [inline, virtual]**

Returns if the classifier is consistent.

Returns

If the classifier is consistent.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 104 of file TMClassAtom.h.

6.54.3.7 **template<class T> virtual bool TMClassAtom< T >::isThereVarID (uli_t varID) const [inline, virtual]**

Return if classifier has got a specific variable ID.

Parameters

<code>varID</code>	ID of variable.
--------------------	-----------------

Returns

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 169 of file TMClassAtom.h.

6.54.3.8 template<class T> bool TMClassAtom< T >::operator==(const TMClassAtom< T > & tMClassAtom) [inline]

Definition at line 160 of file TMClassAtom.h.

6.54.3.9 template<class T> virtual std::ostream& TMClassAtom< T >::print (std::ostream & os) const [inline, virtual]

Prints the classifier to output stream.

Parameters

os	Output stream.
----	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 108 of file TMClassAtom.h.

6.54.3.10 template<class T> virtual void TMClassAtom< T >::printXml (std::ostream & os) const [inline, virtual]

Prints the classifier to output stream in XML format.

Parameters

os	Output stream.
----	----------------

Returns

Output stream.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 123 of file TMClassAtom.h.

6.54.3.11 template<class T> virtual uli_t TMClassAtom< T >::resultingNodeSize () const [inline, virtual]

Returns the size of resulting node.

Returns

Size of resulting node.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 152 of file TMClassAtom.h.

6.54.3.12 template<class T> void TMClassAtom< T >::setThresholds (Tvector vec) [inline]

Definition at line 143 of file TMClassAtom.h.

6.54.3.13 template<class T> void TMClassAtom< T >::setThresholdsNoSort (Tvector vec) [inline]

Definition at line 148 of file TMClassAtom.h.

6.54.3.14 template<class T> void TMClassAtom< T >::setVarID (uli_t varID) [inline]

Definition at line 166 of file TMClassAtom.h.

6.54.3.15 template<class T> virtual uli_t TMClassAtom< T >::size () const [inline, virtual]

Returns the size of classifier.

Returns

Size of classifier.

Reimplemented from [ClassAtom< T, uli_t >](#).

Definition at line 156 of file TMClassAtom.h.

6.54.4 Member Data Documentation

6.54.4.1 template<class T> Tvector TMClassAtom< T >::thresholds [private]

Definition at line 173 of file TMClassAtom.h.

6.54.4.2 template<class T> uli_t TMClassAtom< T >::varID

Definition at line 171 of file TMClassAtom.h.

The documentation for this class was generated from the following file:

- src/library/[TMClassAtom.h](#)

6.55 Tree< T, C > Class Template Reference

```
#include <Tree.h>
```

Public Member Functions

- `Tree ()`
- `Tree (Node< T, C > *root)`
- `virtual ~Tree ()`
- `void setRoot (Node< T, C > *root)`
- `Node< T, C > * getRoot () const`
- `virtual T classify (const std::vector< T > &sample) const`
- `virtual std::ostream & print (std::ostream &os) const`
- `virtual bool isThereVarID (uli_t varID) const`
- `virtual void printXml (std::ostream &os) const`
- `virtual void summary () const`

Public Attributes

- `Node< T, C > * root`

6.55.1 Detailed Description

`template<class T, class C>class Tree< T, C >`

Definition at line 24 of file Tree.h.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 `template<class T, class C> Tree< T, C >::Tree () [inline]`

Definition at line 26 of file Tree.h.

6.55.2.2 `template<class T, class C> Tree< T, C >::Tree (Node< T, C > * root) [inline]`

Definition at line 27 of file Tree.h.

6.55.2.3 `template<class T, class C> virtual Tree< T, C >::~Tree () [inline, virtual]`

Definition at line 28 of file Tree.h.

6.55.3 Member Function Documentation

6.55.3.1 `template<class T, class C> virtual T Tree< T, C >::classify (const std::vector< T > & sample) const [inline, virtual]`

Definition at line 36 of file Tree.h.

6.55.3.2 **template<class T, class C> Node<T, C>* Tree< T, C >::getRoot() const [inline]**

Definition at line 33 of file Tree.h.

6.55.3.3 **template<class T, class C> virtual bool Tree< T, C >::isThereVarID(uli_t varID) const [inline, virtual]**

Definition at line 45 of file Tree.h.

6.55.3.4 **template<class T, class C> virtual std::ostream& Tree< T, C >::print(std::ostream & os) const [inline, virtual]**

Definition at line 40 of file Tree.h.

6.55.3.5 **template<class T, class C> virtual void Tree< T, C >::printXml(std::ostream & os) const [inline, virtual]**

Definition at line 49 of file Tree.h.

6.55.3.6 **template<class T, class C> void Tree< T, C >::setRoot(Node< T, C > * root) [inline]**

Definition at line 32 of file Tree.h.

6.55.3.7 **template<class T, class C> virtual void Tree< T, C >::summary() const [inline, virtual]**

Definition at line 57 of file Tree.h.

6.55.4 Member Data Documentation

6.55.4.1 **template<class T, class C> Node<T, C>* Tree< T, C >::root**

Definition at line 72 of file Tree.h.

The documentation for this class was generated from the following file:

- [src/library/Tree.h](#)

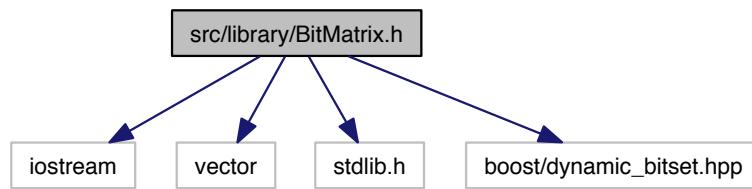
Chapter 7

File Documentation

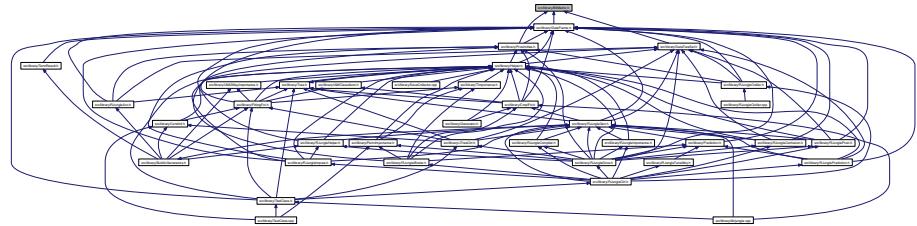
7.1 src/library/BitMatrix.cpp File Reference

7.2 src/library/BitMatrix.h File Reference

```
#include <iostream> #include <vector> #include <stdlib.h> #include <boost/dynamic_bitset.hpp> Include dependency graph  
for BitMatrix.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [BitMatrix](#)

Representation of a bit matrix that can store bits exclusively.

Defines

- `#define NULL 0`

Functions

- `std::ostream & operator<< (std::ostream &os, const BitMatrix &bitMatrix)`
Prints a bit matrix to stream.

7.2.1 Define Documentation

7.2.1.1 #define NULL 0

Definition at line 36 of file BitMatrix.h.

7.2.2 Function Documentation

7.2.2.1 `std::ostream& operator<< (std::ostream & os, const BitMatrix & bitMatrix)` [inline]

Prints a bit matrix to stream.

Parameters

<code>os</code>	Output stream.
<code>bitMatrix</code>	Bit matrix to be printed.

Returns

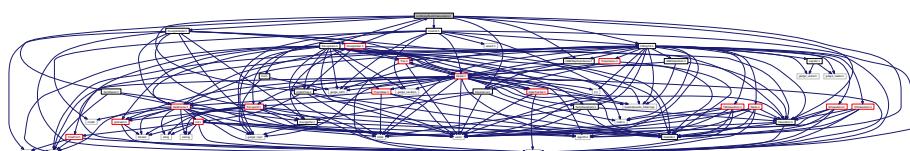
Output stream.

Definition at line 299 of file BitMatrix.h.

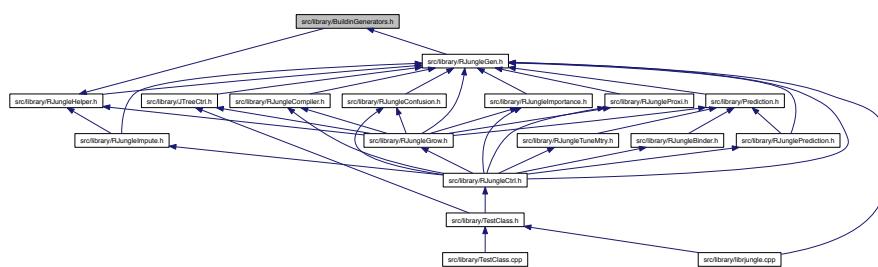
7.3 src/library/BuildinGenerators.cpp File Reference

7.4 src/library/BuildinGenerators.h File Reference

```
#include "treedefs.h" #include "RJungleHelper.h" #include "Importance.h" #include "ClassAtom.h" #include "Fitting-Fct.h" #include "CmpldTree.h" #include "CondInf.h" #include "RJungleAcc.h" #include "IAM2WayImportance.h" #include "-TermResult.h" Include dependency graph for BuildinGenerators.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [BuildinGenFct< T >](#)
A binder for all tree specific functions.

Namespaces

- namespace [rjungle](#)

- namespace `rjungle::Generator`
- namespace `rjungle::Generator::CART`
- namespace `rjungle::Generator::CARTsse`
- namespace `rjungle::Generator::CARTtwoing`
- namespace `rjungle::Generator::CARTreg`
- namespace `rjungle::Generator::CARTregTwoing`
- namespace `rjungle::Generator::CARTsrt`

*Assign all **CART** ($y=nominal/x=numeric$) specific functions.*
- namespace `rjungle::Generator::CARTcor`

Experimental fitting function.
- namespace `rjungle::Generator::CISNP`

Conditional Inference trees for SNP data.
- namespace `rjungle::Generator::CICase1`

Conditional Inference trees.
- namespace `rjungle::Generator::CICase3`

Conditional Inference trees.
- namespace `rjungle::Generator::LOTUS`

Experimental fitting function.
- namespace `rjungle::Generator::UCART`

Experimental fitting function.
- namespace `rjungle::Generator::SCART`

Experimental fitting function.
- namespace `rjungle::Generator::trendCART`

Experimental fitting function.
- namespace `rjungle::Generator::imputeTree`

Experimental fitting function.
- namespace `rjungle::Generator::CARTregWithMiss`

Experimental fitting function.
- namespace `rjungle::Generator::CARTtwoWayIntAdd`

Experimental fitting function.

Functions

- template<class T >
void `rjungle::Generator::CART::assignFunctions` (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nominal/x=numeric$) specific functions.*
- template<class T >
void `rjungle::Generator::CARTsse::assignFunctions` (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nominal/x=numeric$) specific functions.*
- template<class T >
void `rjungle::Generator::CARTtwoing::assignFunctions` (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nominal/x=nominal$) specific functions.*

- template<class T >
void **rjungle::Generator::CARTreg::assignFunctions** (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nemonic/x=nemonic$) specific functions.*

- template<class T >
void **rjungle::Generator::CARTregTwoing::assignFunctions** (BuildinGenFct< T > *genFct)

*Assign all **CART** ($y=nemonic/x=nominal$) specific functions.*

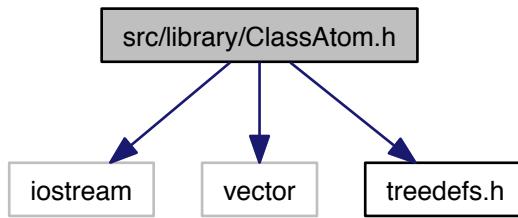
- template<class T >
void **rjungle::Generator::CARTsrt::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CARTcor::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CISNP::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CICase1::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CICase3::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::LOTUS::assignFunctions** (BuildinGenFct< T > *genFct)

- template<class T >
void **rjungle::Generator::UCART::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::SCART::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::trendCART::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::imputeTree::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CARTregWithMiss::assignFunctions** (BuildinGenFct< T > *genFct)
- template<class T >
void **rjungle::Generator::CARTtwoWayIntAdd::assignFunctions** (BuildinGenFct< T > *genFct)

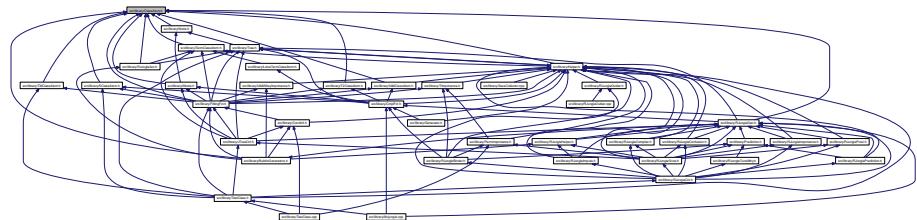
7.5 src/library/ClassAtom.cpp File Reference

7.6 src/library/ClassAtom.h File Reference

```
#include <iostream> #include <vector> #include "treedefs.h"
"Include dependency graph for ClassAtom.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [ClassAtom< T, C >](#)

Smallest classifier unit.

Functions

- template<class T , class C >

std::ostream & [operator<<](#) (std::ostream &os, const [ClassAtom< T, C >](#) &class-
Atom)

Prints the classifier to output stream.

7.6.1 Function Documentation

```
7.6.1.1 template<class T , class C > std::ostream& operator<< ( std::ostream & os, const
ClassAtom< T, C > & classAtom )
```

Prints the classifier to output stream.

Parameters

<i>os</i>	Output stream.
<i>classAtom</i>	Classifier object.

Returns

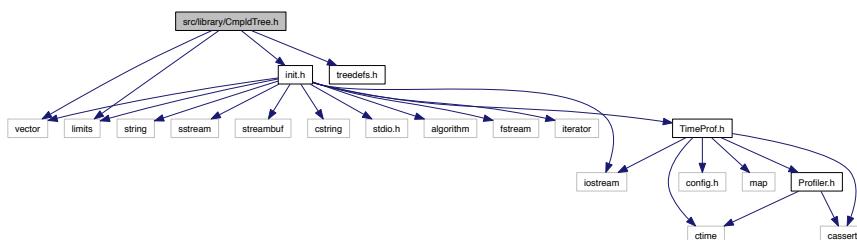
Output stream.

Definition at line 194 of file ClassAtom.h.

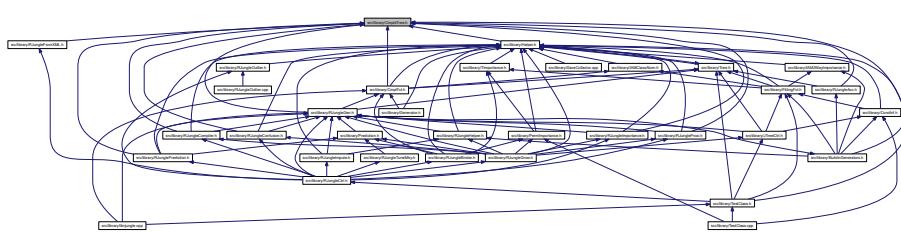
7.7 src/library/CmpldTree.cpp File Reference

7.8 src/library/CmpldTree.h File Reference

```
#include <vector> #include <limits> #include "init.h" ×
#include "treedefs.h" Include dependency graph for CmpldTree.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [CmpldTree< T >](#)

Functions

- template<class T >
`std::ostream & operator<< (std::ostream &os, CmpldTree< T > &cmpldTree)`
Prints compiled tree to output stream.

7.8.1 Function Documentation

7.8.1.1 template<class T > std::ostream& operator<< (std::ostream & os, CmpldTree< T > & cmpldTree) [inline]

Prints compiled tree to output stream.

Parameters

<code>os</code>	Output stream.
<code>cmpldTree</code>	A compiled tree.

Returns

Output stream.

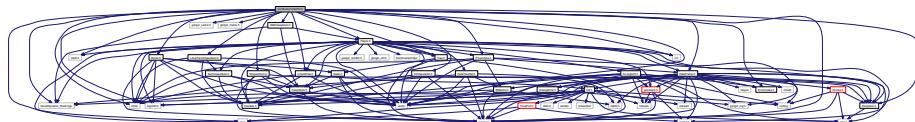
Definition at line 328 of file CmpldTree.h.

7.9 src/library/CmplFct.cpp File Reference

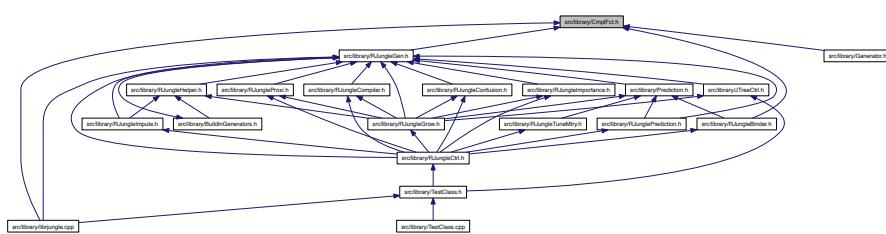
7.10 src/library/CmplFct.h File Reference

```
#include <iostream> #include <vector> #include <map> ×
#include <limits> #include <algorithm> #include <math.h>
#include <gsl/gsl_vector.h> #include <gsl/gsl_matrix.h>
#include <boost/dynamic_bitset.hpp> #include "ErrorCodes.h"
#include "CmpldTree.h" #include "INode.h" #include "T2ClassAtom.h"
#include "SClassAtom.h" #include "IAM-ClassAtom.h" #include "LotusTermClassAtom.h"
#include "-Tree.h" #include "Helper.h" #include "Exception.h" #include "DataFrame.h"
#include "Proximities.h" #include "lr.h" ×
```

Include dependency graph for CmplFct.h:



This graph shows which files directly or indirectly include this file:



Classes

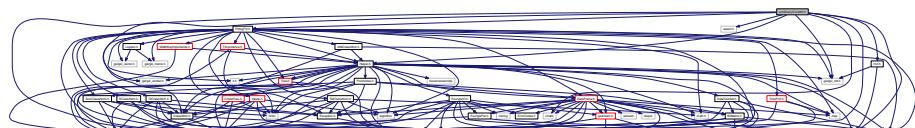
- class [CmplFct](#)

Representation of a set of compiler functions.

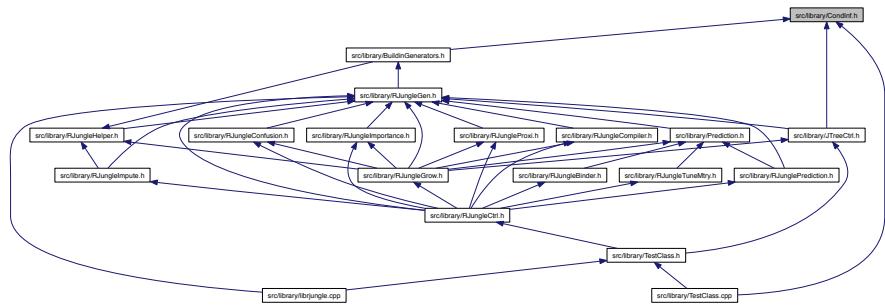
7.11 src/library/CondInf.cpp File Reference

7.12 src/library/CondInf.h File Reference

```
#include <vector> #include <cmath> #include <iostream>
#include <assert.h> #include <gsl/gsl_rng.h> #include
<gsl/gsl_cdf.h> #include <gsl/gsl_randist.h> #include "-_
Helper.h" #include "FittingFct.h" #include "mvt.h" Include de-
pendency graph for CondInf.h:
```



This graph shows which files directly or indirectly include this file:



Classes

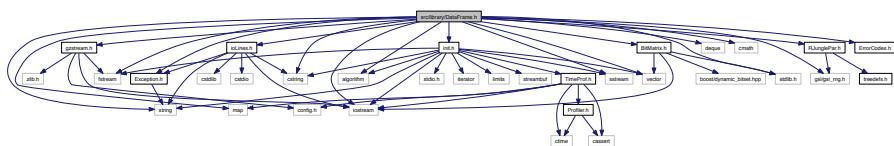
- class CondInf< T >

CI Functions for various cases as follows:

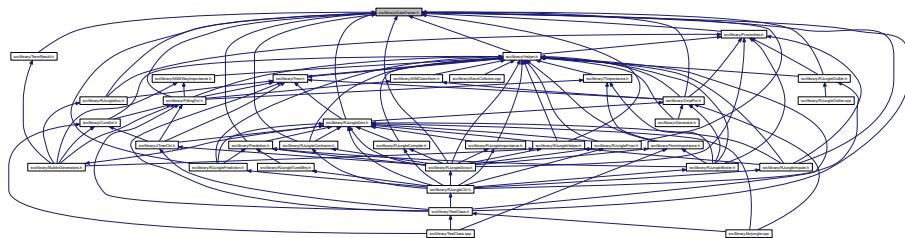
7.13 src/library/DataFrame.cpp File Reference

7.14 src/library/DataFrame.h File Reference

```
#include <algorithm> #include <iostream> #include <fstream> x
#include <sstream> #include <string> #include <cstring>
#include <vector> #include <deque> #include <cmath> x
#include <map> #include <stdlib.h> #include <gsl/gsl_
rng.h> #include "ErrorCodes.h" #include "init.h" #include
"ioLines.h" #include "Exception.h" #include "gzstream.h"
#include "BitMatrix.h" #include "RJunglePar.h" Include depen-
dency graph for DataFrame.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [DataFrame< T >](#)

Functions

- template<class T >
std::ostream & [operator<<](#) (std::ostream &os, [DataFrame< T >](#) &dataFrame)

7.14.1 Function Documentation

7.14.1.1 template<class T > std::ostream& operator<< (std::ostream & os, [DataFrame< T >](#) & [dataFrame](#)) [inline]

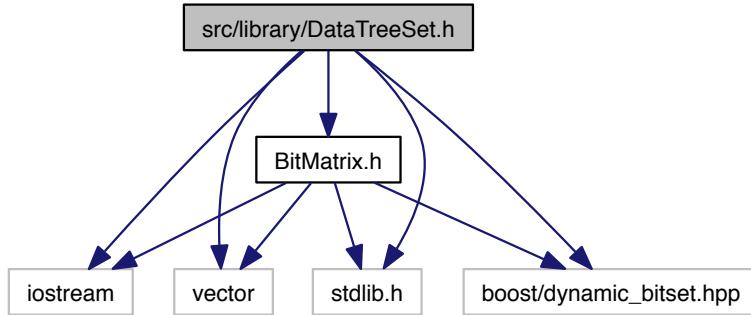
Definition at line 1262 of file DataFrame.h.

7.15 src/library/DataTreeSet.cpp File Reference

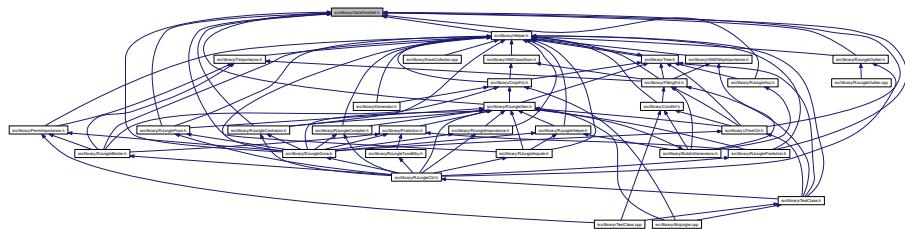
7.16 src/library/DataTreeSet.h File Reference

```
#include <iostream> #include <vector> #include <stdlib.h> #include <boost/dynamic_bitset.hpp> #include "Bit-
```

Matrix.h" Include dependency graph for DataTreeSet.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [DataTreeSet](#)

Functions

- std::ostream & [operator<<](#) (std::ostream &os, const [DataTreeSet](#) &dataTreeSet)

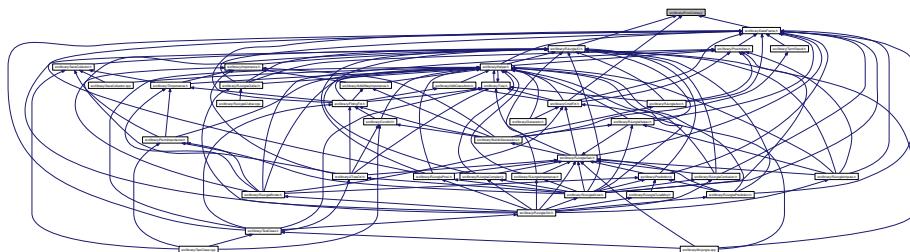
7.16.1 Function Documentation

7.16.1.1 std::ostream& operator<< (std::ostream & os, const DataTreeSet & dataTreeSet) [inline]

Definition at line 94 of file DataTreeSet.h.

7.17 src/library/ErrorCodes.h File Reference

This graph shows which files directly or indirectly include this file:



Defines

- #define [ERRORCODE_1](#) "Invalid index number (perhaps to big)"
- #define [ERRORCODE_2](#) "Writing a duplication of a pointer to a vector (same adresses)"
- #define [ERRORCODE_3](#) "Writing a wrong sized variable to a dataframe"
- #define [ERRORCODE_4](#) "Writing a variable to a dataframe, which name is already in use (same name)"
- #define [ERRORCODE_5](#) "No data served ([NULL](#)-Pointer)"
- #define [ERRORCODE_6](#) "Outcome vector has not got one or two classes."
- #define [ERRORCODE_7](#) "File not found."
- #define [ERRORCODE_8](#) "Error while parsing csv file."
- #define [ERRORCODE_9](#) "No fitting function giving in constructor of tree"
- #define [ERRORCODE_10](#) "No tree served ([NULL](#)-Pointer)"
- #define [ERRORCODE_11](#) "Empty tree served (root node == [NULL](#)-Pointer)"
- #define [ERRORCODE_12](#) "Vector has not got two classes."
- #define [ERRORCODE_13](#) "Dependent variable has to be binary"
- #define [ERRORCODE_14](#) "Classifier points to a children node that does not exist."
- #define [ERRORCODE_15](#) "Classifier points to a children node that is a [NULL](#) pointer."
- #define [ERRORCODE_16](#) "Infile has less columns than wanted."
- #define [ERRORCODE_17](#) "Unknown level of measurement."
- #define [ERRORCODE_18](#) "Dependency variable index is bigger than the number of columns."
- #define [ERRORCODE_19](#) "Can not find specified dependency variable in the data. Index too big."
- #define [ERRORCODE_20](#) "Can not find classes 0,1 in the dependency variable."
- #define [ERRORCODE_21](#) "Can not open file for writing."
- #define [ERRORCODE_22](#) "Has got allocated memory already. Array pointer not [NULL](#)."
- #define [ERRORCODE_23](#) "Array is empty. Array pointer is [NULL](#)."

- #define **ERRORCODE_24** "Error while calculating with mpfr."
- #define **ERRORCODE_25** "Can not find file. Or number of rows and/or columns is equals to 0. Or more column names than columns in data."
- #define **ERRORCODE_26** "Child is **NULL**"
- #define **ERRORCODE_27** "No Children there"
- #define **ERRORCODE_28** "In makeNode recursion depth >= 15000"
- #define **ERRORCODE_29** "Can not load plugin"
- #define **ERRORCODE_30** "Extraction of plugin content failed. (No function librjungle_plugin_init?)"
- #define **ERRORCODE_31** "Do not know this person"
- #define **ERRORCODE_32** "Tree size is zero"
- #define **ERRORCODE_33** "The mtry < 2"
- #define **ERRORCODE_34** "Unknown dependent variable name"
- #define **ERRORCODE_35** "DataFrame is **NULL**"
- #define **ERRORCODE_36** "Too less columns for output. (<2)"
- #define **ERRORCODE_37** "Missings in Data"
- #define **ERRORCODE_38** "The mtry is bigger than nmber of columns (ncol < mtry)"
- #define **ERRORCODE_39** "Unknown memmode"
- #define **ERRORCODE_40** "Your program was compiled without plugin support"
- #define **ERRORCODE_41** "Can not find column specified in column selection file (-C parameter)"
- #define **ERRORCODE_42** "Can not find category"
- #define **ERRORCODE_43** "Variables in data are wrongly named or orderes (FID, IID, PAT, MAT, SEX, PHENOTYPE)"
- #define **ERRORCODE_44** "Is not a valid ped format or can not use options together: --pedfile and --transpose"
- #define **ERRORCODE_45** "Option -D has to been PHENOTYPE or leave it empty"
- #define **ERRORCODE_46** "The delimeter has to be a SPACE character in --pedfile mode"
- #define **ERRORCODE_47** "Can not find variable with that name"
- #define **ERRORCODE_48** "Data has not got 2 groups to classify on"
- #define **ERRORCODE_49** "Minimal partition size is too small (choose option -l 20 or higher)"
- #define **ERRORCODE_50** "This type of tree has got no function for updating proximities"
- #define **ERRORCODE_51** "Do not use backward elimination and tune mtry together"
- #define **ERRORCODE_52** "Do not use backward elimination and sample proximities together"
- #define **ERRORCODE_53** "Do not use backward elimination and variable proximities together"
- #define **ERRORCODE_54** "A data cell in the input file is empty"
- #define **ERRORCODE_55** "Invalid PED format. Needed: FID IID PAT MAT SEX PHENOTYPE"
- #define **ERRORCODE_56** "Can not fetch next tree from file. Rjungle XML file corrupt? Or can not read from XML file. (Not there?)"

- #define **ERRORCODE_57** "Some rows differs in number of columns (or a space at the end of a column?)"
- #define **ERRORCODE_58** "Perform permutation importance when using MPI"
- #define **ERRORCODE_59** "Number of processes is greater than ntree. That is not allowed."
- #define **ERRORCODE_60** "Unknown format (--write)"
- #define **ERRORCODE_61** "Can not read from XML file"
- #define **ERRORCODE_62** "Conditional importrance is available in the following tree types (more will be added in future): 1, 3, 5"
- #define **ERRORCODE_63** "CPU does not support SSE"
- #define **ERRORCODE_64** "Lotus can be used in <<double or float>>-memory mode exclusively."

7.17.1 Define Documentation

7.17.1.1 #define **ERRORCODE_1** "Invalid index number (perhaps to big)"

Definition at line 11 of file ErrorCodes.h.

7.17.1.2 #define **ERRORCODE_10** "No tree served (NULL-Pointer)"

Definition at line 20 of file ErrorCodes.h.

7.17.1.3 #define **ERRORCODE_11** "Empty tree served (root node == NULL-Pointer)"

Definition at line 21 of file ErrorCodes.h.

7.17.1.4 #define **ERRORCODE_12** "Vector has not got two classes."

Definition at line 22 of file ErrorCodes.h.

7.17.1.5 #define **ERRORCODE_13** "Dependent variable has to be binary"

Definition at line 23 of file ErrorCodes.h.

7.17.1.6 #define **ERRORCODE_14** "Classifier points to a children node that does not exist."

Definition at line 24 of file ErrorCodes.h.

7.17.1.7 #define **ERRORCODE_15** "Classifier points to a children node that is a NULL pointer."

Definition at line 25 of file ErrorCodes.h.

7.17.1.8 #define **ERRORCODE_16** "Infile has less columns than wanted."

Definition at line 26 of file ErrorCodes.h.

7.17.1.9 #define **ERRORCODE_17** "Unknown level of measurement."

Definition at line 27 of file ErrorCodes.h.

7.17.1.10 #define **ERRORCODE_18** "Dependency variable index is bigger than the number of columns."

Definition at line 28 of file ErrorCodes.h.

7.17.1.11 #define **ERRORCODE_19** "Can not find specified dependency variable in the data. Index too big."

Definition at line 29 of file ErrorCodes.h.

7.17.1.12 #define **ERRORCODE_2** "Writing a duplication of a pointer to a vector (same adresses)"

Definition at line 12 of file ErrorCodes.h.

7.17.1.13 #define **ERRORCODE_20** "Can not find classes 0,1 in the dependency variable."

Definition at line 30 of file ErrorCodes.h.

7.17.1.14 #define **ERRORCODE_21** "Can not open file for writing."

Definition at line 31 of file ErrorCodes.h.

7.17.1.15 #define **ERRORCODE_22** "Has got allocated memory already. Array pointer not NULL."

Definition at line 32 of file ErrorCodes.h.

7.17.1.16 #define **ERRORCODE_23** "Array is empty. Array pointer is NULL."

Definition at line 33 of file ErrorCodes.h.

7.17.1.17 #define **ERRORCODE_24** "Error while calculating with mpfr."

Definition at line 34 of file ErrorCodes.h.

7.17.1.18 `#define ERRORCODE_25 "Can not find file. Or number of rows and/or columns is equals to 0. Or more column names than columns in data."`

Definition at line 35 of file ErrorCodes.h.

7.17.1.19 `#define ERRORCODE_26 "Child is NULL"`

Definition at line 36 of file ErrorCodes.h.

7.17.1.20 `#define ERRORCODE_27 "No Children there"`

Definition at line 37 of file ErrorCodes.h.

7.17.1.21 `#define ERRORCODE_28 "In makeNode recursion depth >= 15000"`

Definition at line 38 of file ErrorCodes.h.

7.17.1.22 `#define ERRORCODE_29 "Can not load plugin"`

Definition at line 39 of file ErrorCodes.h.

7.17.1.23 `#define ERRORCODE_3 "Writing a wrong sized variable to a dataframe"`

Definition at line 13 of file ErrorCodes.h.

7.17.1.24 `#define ERRORCODE_30 "Extraction of plugin content failed. (No function librjungle_plugin_init?)"`

Definition at line 40 of file ErrorCodes.h.

7.17.1.25 `#define ERRORCODE_31 "Do not know this person"`

Definition at line 41 of file ErrorCodes.h.

7.17.1.26 `#define ERRORCODE_32 "Tree size is zero"`

Definition at line 42 of file ErrorCodes.h.

7.17.1.27 `#define ERRORCODE_33 "The mtry < 2"`

Definition at line 43 of file ErrorCodes.h.

7.17.1.28 #define **ERRORCODE_34** "Unknown dependent variable name"

Definition at line 44 of file ErrorCodes.h.

7.17.1.29 #define **ERRORCODE_35** "DataFrame is NULL"

Definition at line 45 of file ErrorCodes.h.

7.17.1.30 #define **ERRORCODE_36** "Too less columns for output. (<2)"

Definition at line 46 of file ErrorCodes.h.

7.17.1.31 #define **ERRORCODE_37** "Missings in Data"

Definition at line 47 of file ErrorCodes.h.

7.17.1.32 #define **ERRORCODE_38** "The mtry is bigger than nmber of columns (ncol < mtry)"

Definition at line 48 of file ErrorCodes.h.

7.17.1.33 #define **ERRORCODE_39** "Unknown memmode"

Definition at line 49 of file ErrorCodes.h.

7.17.1.34 #define **ERRORCODE_4** "Writing a variable to a dataframe, which name is already in use (same name)"

Definition at line 14 of file ErrorCodes.h.

7.17.1.35 #define **ERRORCODE_40** "Your program was compiled without plugin support"

Definition at line 50 of file ErrorCodes.h.

7.17.1.36 #define **ERRORCODE_41** "Can not find column specified in column selection file (-C parameter)"

Definition at line 51 of file ErrorCodes.h.

7.17.1.37 #define **ERRORCODE_42** "Can not find category"

Definition at line 52 of file ErrorCodes.h.

7.17.1.38 `#define ERRORCODE_43 "Variables in data are wrongly named or orderes (FID, IID, PAT, MAT, SEX, PHENOTYPE)"`

Definition at line 53 of file ErrorCodes.h.

7.17.1.39 `#define ERRORCODE_44 "Is not a valid ped format or can not use options together: --pedfile and --transpose"`

Definition at line 54 of file ErrorCodes.h.

7.17.1.40 `#define ERRORCODE_45 "Option -D has to been PHENOTYPE or leave it empty"`

Definition at line 55 of file ErrorCodes.h.

7.17.1.41 `#define ERRORCODE_46 "The delimeter has to be a SPACE character in --pedfile mode"`

Definition at line 56 of file ErrorCodes.h.

7.17.1.42 `#define ERRORCODE_47 "Can not find variable with that name"`

Definition at line 57 of file ErrorCodes.h.

7.17.1.43 `#define ERRORCODE_48 "Data has not got 2 groups to classify on"`

Definition at line 58 of file ErrorCodes.h.

7.17.1.44 `#define ERRORCODE_49 "Minimal partition size is too small (choose option -I 20 or higher)"`

Definition at line 59 of file ErrorCodes.h.

7.17.1.45 `#define ERRORCODE_5 "No data served (NULL-Pointer)"`

Definition at line 15 of file ErrorCodes.h.

7.17.1.46 `#define ERRORCODE_50 "This type of tree has got no function for updating proximities"`

Definition at line 60 of file ErrorCodes.h.

7.17.1.47 `#define ERRORCODE_51 "Do not use backward elimination and tune mtry together"`

Definition at line 61 of file ErrorCodes.h.

7.17.1.48 `#define ERRORCODE_52 "Do not use backward elimination and sample proximities together"`

Definition at line 62 of file ErrorCodes.h.

7.17.1.49 `#define ERRORCODE_53 "Do not use backward elimination and variable proximities together"`

Definition at line 63 of file ErrorCodes.h.

7.17.1.50 `#define ERRORCODE_54 "A data cell in the input file is empty"`

Definition at line 64 of file ErrorCodes.h.

7.17.1.51 `#define ERRORCODE_55 "Invalid PED format. Needed: FID IID PAT MAT SEX PHENOTYPE"`

Definition at line 65 of file ErrorCodes.h.

7.17.1.52 `#define ERRORCODE_56 "Can not fetch next tree from file. RJungle XML file corrupt? Or can not read from XML file. (Not there?)"`

Definition at line 66 of file ErrorCodes.h.

7.17.1.53 `#define ERRORCODE_57 "Some rows differs in number of columns (or a space at the end of a column?)"`

Definition at line 67 of file ErrorCodes.h.

7.17.1.54 `#define ERRORCODE_58 "Perform permutation importance when using MPI"`

Definition at line 68 of file ErrorCodes.h.

7.17.1.55 `#define ERRORCODE_59 "Number of processes is greater than ntree. That is not allowed."`

Definition at line 69 of file ErrorCodes.h.

7.17.1.56 `#define ERRORCODE_6 "Outcome vector has not got one or two classes."`

Definition at line 16 of file ErrorCodes.h.

7.17.1.57 `#define ERRORCODE_60 "Unknown format (--write)"`

Definition at line 70 of file ErrorCodes.h.

7.17.1.58 `#define ERRORCODE_61 "Can not read from XML file"`

Definition at line 71 of file ErrorCodes.h.

7.17.1.59 `#define ERRORCODE_62 "Conditional importance is available in the following tree types (more will be added in future): 1, 3, 5"`

Definition at line 72 of file ErrorCodes.h.

7.17.1.60 `#define ERRORCODE_63 "CPU does not support SSE"`

Definition at line 73 of file ErrorCodes.h.

7.17.1.61 `#define ERRORCODE_64 "Lotus can be used in <<double or float>>-memory mode exclusively."`

Definition at line 74 of file ErrorCodes.h.

7.17.1.62 `#define ERRORCODE_7 "File not found."`

Definition at line 17 of file ErrorCodes.h.

7.17.1.63 `#define ERRORCODE_8 "Error while parsing csv file."`

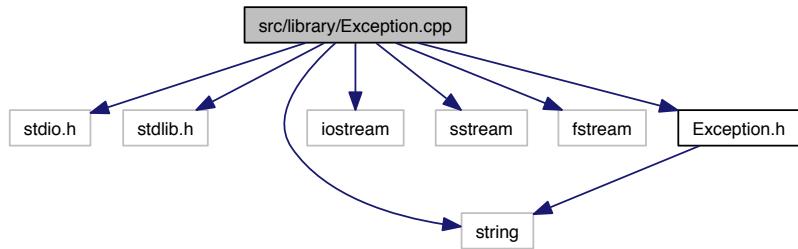
Definition at line 18 of file ErrorCodes.h.

7.17.1.64 `#define ERRORCODE_9 "No fitting function giving in constructor of tree"`

Definition at line 19 of file ErrorCodes.h.

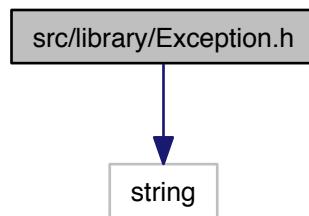
7.18 src/library/Exception.cpp File Reference

```
#include <stdio.h> #include <stdlib.h> #include <string> x  
#include <iostream> #include <sstream> #include <fstream> x  
#include "Exception.h" Include dependency graph for Exception.cpp:
```

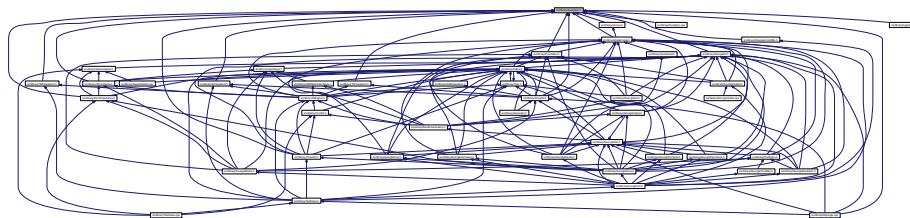


7.19 src/library/Exception.h File Reference

```
#include <string> Include dependency graph for Exception.h:
```



This graph shows which files directly or indirectly include this file:



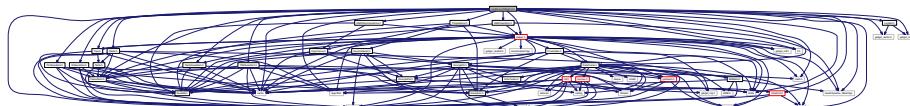
Classes

- class [Exception](#)

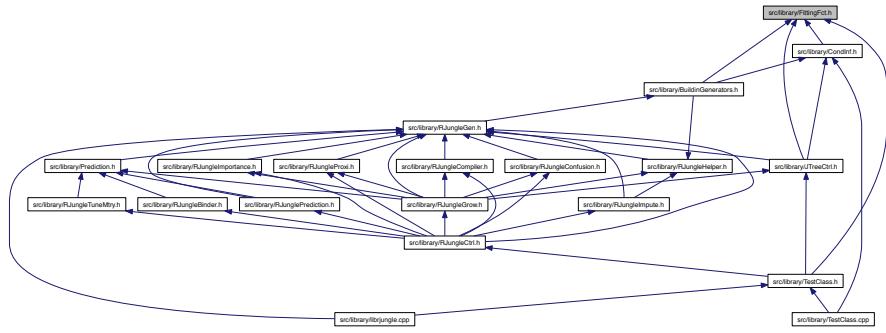
7.20 src/library/FittingFct.cpp File Reference

7.21 src/library/FittingFct.h File Reference

```
#include <iostream> #include <vector> #include <map> x
#include <limits> #include <algorithm> #include <math.h>
#include <gsl/gsl_vector.h> #include <gsl/gsl_matrix.h>
#include <gsl/gsl_cdf.h> #include <boost/dynamic_bitset.hpp>
#include "DataFrame.h" #include "Logistic.h"
#include "ClassAtom.h" #include "SClassAtom.h" #include
"ClassAtom.h" #include "TClassAtom.h" #include "IA-MClassAtom.h"
#include "TermClassAtom.h" #include "T-Importance.h"
#include "IAM2WayImportance.h" #include "-Tree.h"
#include "Helper.h" #include "INode.h" #include "-TimeProf.h"
#include "lr.h" Include dependency graph for FittingFct.h:
```



This graph shows which files directly or indirectly include this file:



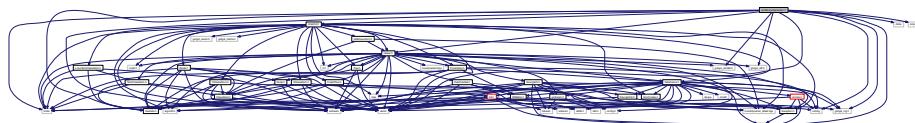
Classes

- class `FittingFctPar< T >`
 - class `FittingFct`

7.22 src/library/Generator.cpp File Reference

7.23 src/library/Generator.h File Reference

```
#include <iostream> #include <cstring> #include <vector>
#include <string>   #include <limits>    #include <ctime>
#include <omp.h> #include <gsl/gsl_rng.h> #include <gsl/gsl-
_randist.h> #include <gsl/gsl_cdf.h> #include "Helper.h"
#include "CmplFct.h" #include "treedefs.h" Include dependency
graph for Generator.h:
```

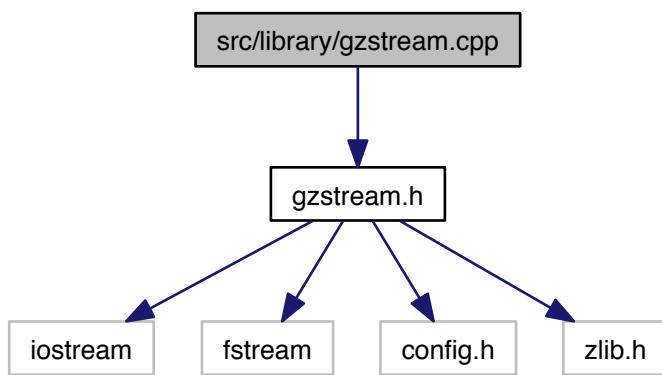


Classes

- class Generator< T >

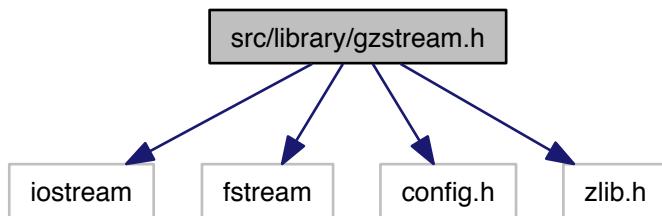
7.24 src/library/gzstream.cpp File Reference

```
#include "gzstream.h" Include dependency graph for gzstream.cpp:
```

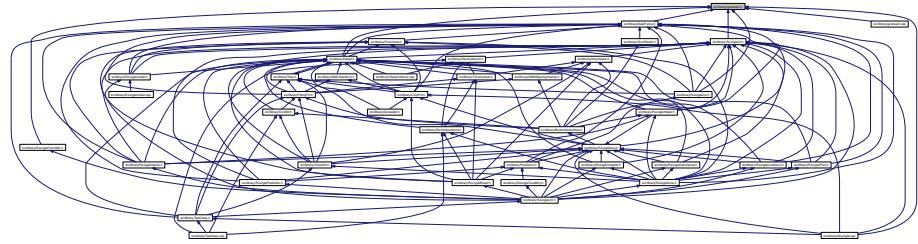


7.25 src/library/gzstream.h File Reference

```
#include <iostream> #include <fstream> #include "config.-  
h" #include <zlib.h> Include dependency graph for gzstream.h:
```



This graph shows which files directly or indirectly include this file:



Defines

- #define `igzstream` std::ifstream
- #define `ogzstream` std::ofstream

7.25.1 Define Documentation

7.25.1.1 #define `igzstream` std::ifstream

Definition at line 122 of file gzstream.h.

7.25.1.2 #define `ogzstream` std::ofstream

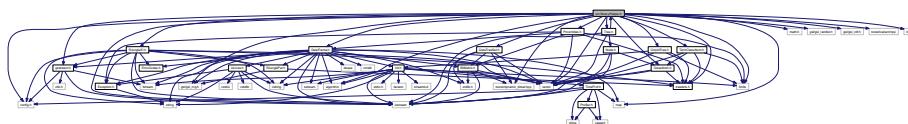
Definition at line 123 of file gzstream.h.

7.26 src/library/Helper.cpp File Reference

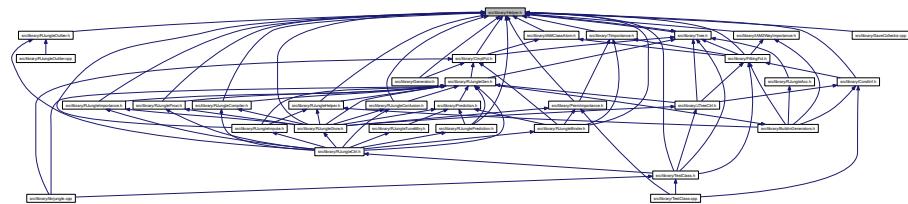
7.27 src/library/Helper.h File Reference

```
#include <iostream>  #include <vector>  #include <map>  ×  
#include <limits>  #include <algorithm>  #include <math.h>  
#include <gsl/gsl_rng.h>  #include <gsl/gsl_randist.h>  
#include <gsl/gsl_cdf.h>  #include <boost/dynamic_bitset.hpp>  
#include <boost/variant.hpp>  #include "config.h"  
#include "RJungleIO.h"  
#include "Proximities.h"  
#include "DataFrame.h"  
#include "DataTreeSet.h"  
#include "ClassAtom.h"  
#include "TermClassAtom.h"  
#include "Tree.h"  ×  
#include "CmpldTree.h"  #include "gzstream.h"  #include
```

```
"lr.h" #include "treedefs.h" Include dependency graph for Helper.h:
```



This graph shows which files directly or indirectly include this file:



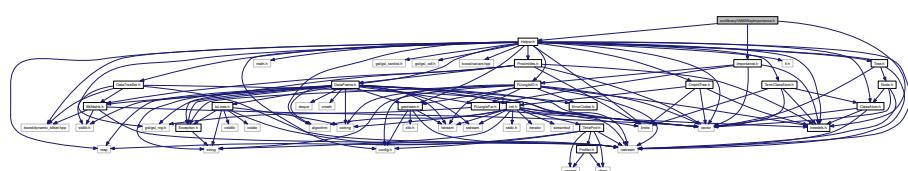
Classes

- class [Helper](#)

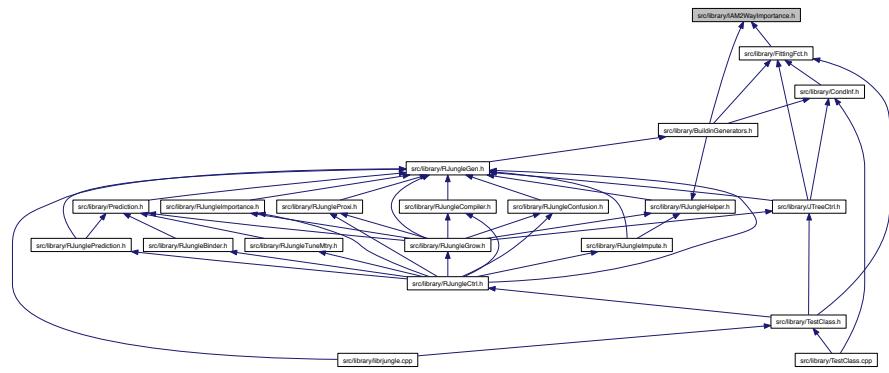
7.28 src/library/IAM2WayImportance.cpp File Reference

7.29 src/library/IAM2WayImportance.h File Reference

```
#include "treedefs.h"    #include "Helper.h"    #include "-  
Importance.h" Include dependency graph for IAM2WayImportance.h:
```



This graph shows which files directly or indirectly include this file:



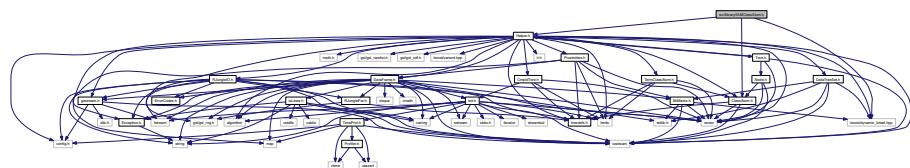
Classes

- class [IAM2WayImportance< T >](#)

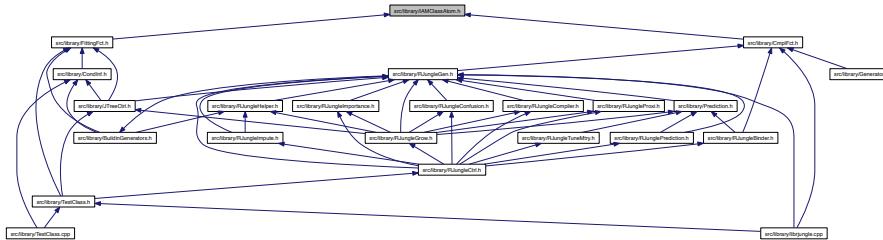
7.30 src/library/IAMClassAtom.cpp File Reference

7.31 src/library/IAMClassAtom.h File Reference

```
#include <boost/dynamic_bitset.hpp> #include "ClassAtom.h" #include "Helper.h" Include dependency graph for IAMClassAtom.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [IAMClassAtom< T >](#)

Defines

- `#define Tvector std::vector<T>`
- `#define CCvector std::vector<uli_t>`

Functions

- template<class T>
`std::ostream & operator<< (std::ostream &os, const IAMClassAtom< T > &iAMClassAtom)`

7.31.1 Define Documentation

7.31.1.1 #define CCvector std::vector<uli_t>

Definition at line 19 of file IAMClassAtom.h.

7.31.1.2 #define Tvector std::vector<T>

Definition at line 16 of file IAMClassAtom.h.

7.31.2 Function Documentation

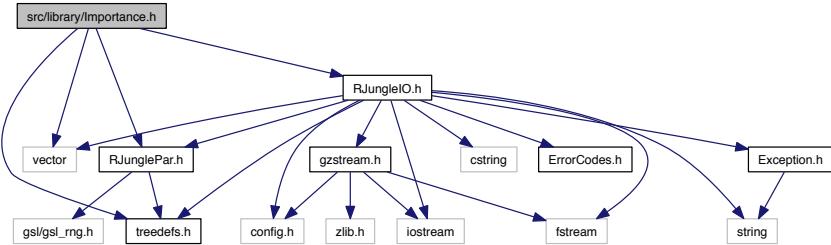
7.31.2.1 template<class T> std::ostream& operator<< (std::ostream & os, const IAMClassAtom< T > & iAMClassAtom)

Definition at line 135 of file IAMClassAtom.h.

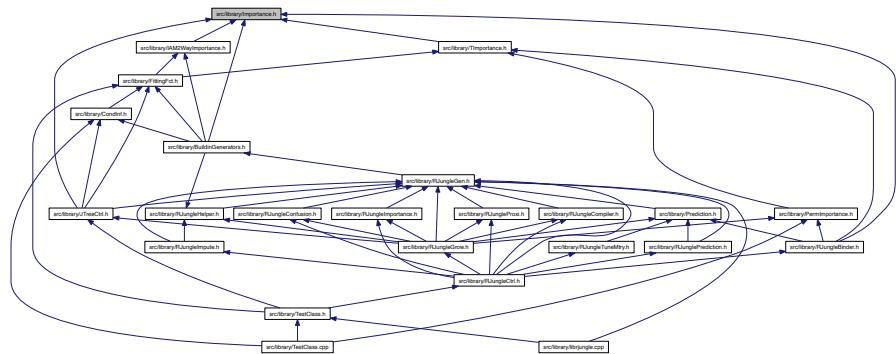
7.32 src/library/Importance.cpp File Reference

7.33 src/library/Importance.h File Reference

```
#include <vector> #include "treedefs.h" #include "RJunglePar.h" #include "RJungleIO.h" Include dependency graph for Importance.h:
```



This graph shows which files directly or indirectly include this file:



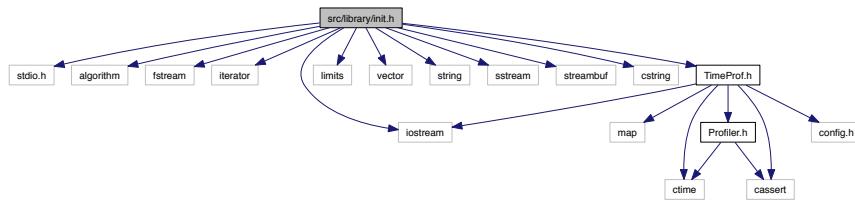
Classes

- class `Importance< T >`

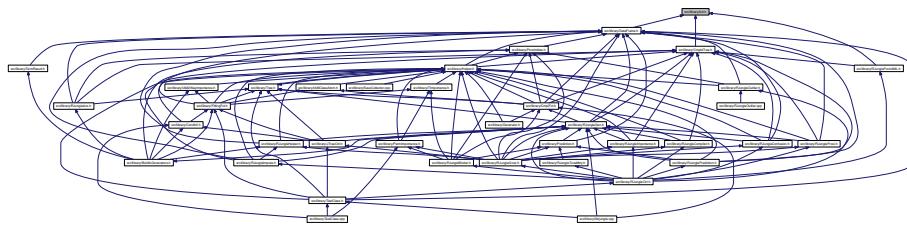
7.34 src/library/init.h File Reference

```
#include <stdio.h> #include <algorithm> #include <fstream> #include <iterator> #include <iostream> #include <limits> #include <vector> #include <string> #include <sstream>
```

```
#include <streambuf> #include <cstring> #include "TimeProf.h" Include dependency graph for init.h:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [rjungle](#)

Functions

- template<class T >
`std::istream & operator>> (std::istream &is, std::vector< std::vector< T > > &tVec)`
- template<class T >
`std::istream & operator>> (std::istream &is, std::vector< T > &tVec)`
- template<class T >
`std::ostream & operator<< (std::ostream &os, std::vector< std::vector< T > > &vec)`
- template<class T >
`std::ostream & operator<< (std::ostream &os, std::vector< T > &vec)`
- template<class T >
`T rjungle::magicAt (T *vec, uli_t j)`

Variables

- static const char [rjungle::strLom](#) [4][8]

7.34.1 Function Documentation

7.34.1.1 `template<class T> std::ostream& operator<< (std::ostream & os, std::vector< std::vector< T > > & vec)`

Definition at line 167 of file init.h.

7.34.1.2 `template<class T> std::ostream& operator<< (std::ostream & os, std::vector< T > & vec)`

Definition at line 188 of file init.h.

7.34.1.3 `template<class T> std::istream& operator>> (std::istream & is, std::vector< std::vector< T > > & tVec)`

Definition at line 28 of file init.h.

7.34.1.4 `template<class T> std::istream& operator>> (std::istream & is, std::vector< T > & tVec)`

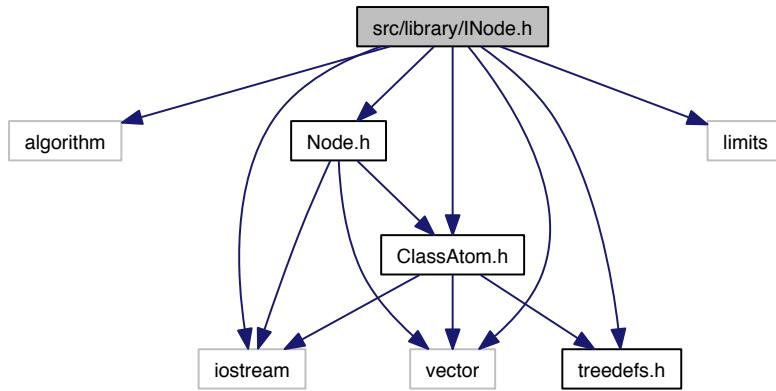
Definition at line 94 of file init.h.

7.35 src/library/INode.cpp File Reference

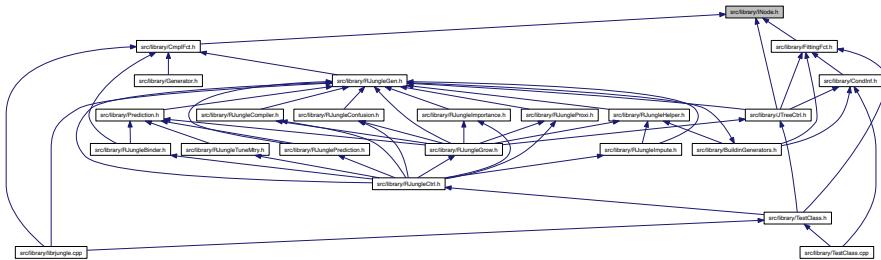
7.36 src/library/INode.h File Reference

```
#include <algorithm> #include <iostream> #include <vector> x
#include <limits> #include "ClassAtom.h" #include "Node.-
```

h" #include "treedefs.h" Include dependency graph for INode.h:



This graph shows which files directly or indirectly include this file:



Classes

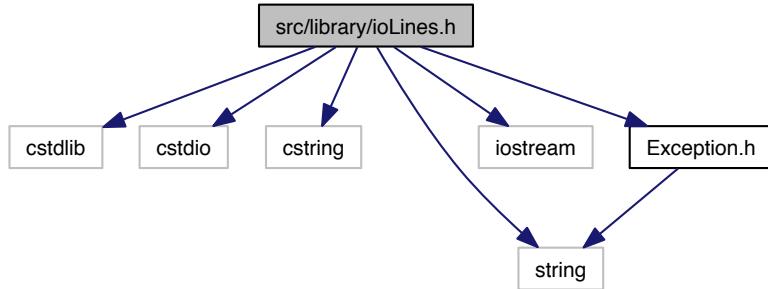
- class `INode< T >`

7.37 src/library/ioLines.cpp File Reference

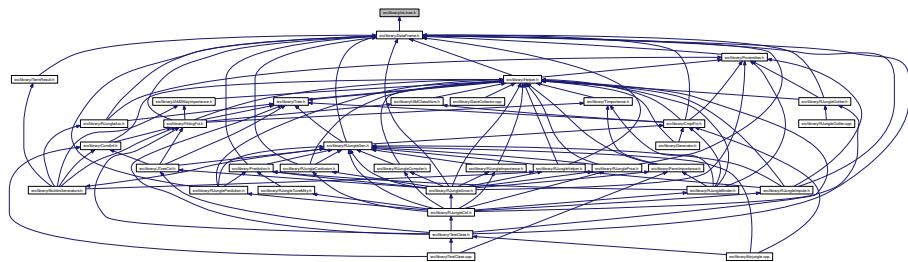
7.38 src/library/ioLines.h File Reference

```
#include <cstdlib> #include <cstdio> #include <cstring>
#include <string> #include <iostream> #include "Exception.h"
```

h" Include dependency graph for ioLines.h:



This graph shows which files directly or indirectly include this file:



Classes

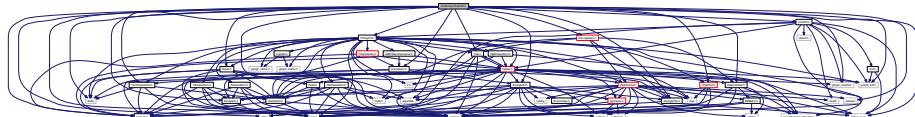
- class [StringIterator](#)

A class for iterating through tokens from a c-string.

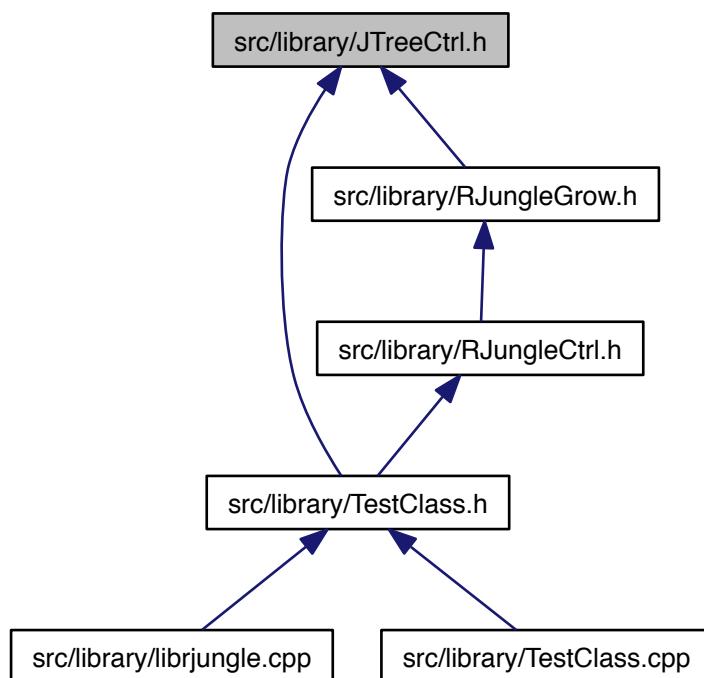
7.39 src/library/JTreeCtrl.h File Reference

```
#include <iostream> #include <sstream> #include <limits> ×
#include <string> #include <vector> #include <gsl/gsl-
rng.h> #include <gsl/gsl_randist.h> #include <gsl/gsl-
cdf.h> #include "treedefs.h" #include "Tree.h" #include "I-
Node.h" #include "Exception.h" #include "TermClassAtom.h"
#include "Importance.h" #include "FittingFct.h" #include
"CondInf.h" #include "DataTreeSet.h" #include "RJungle-
```

```
Par.h" #include "RJungleIO.h" #include "RJungleGen.h" ×  
#include "TimeProf.h" Include dependency graph for JTreeCtrl.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [JTreeCtrl< T >](#)

7.40 src/library/JTreeTweaks.cpp File Reference

7.41 src/library/JTreeTweaks.h File Reference

Classes

- struct [JTreeTweaks](#)

Typedefs

- typedef unsigned long int [uli_t](#)
- typedef struct [JTreeTweaks](#) [JTreeTweaks](#)

7.41.1 Typedef Documentation

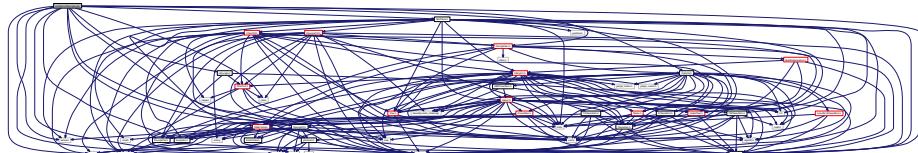
7.41.1.1 [typedef struct JTreeTweaks JTreeTweaks](#)

7.41.1.2 [typedef unsigned long int uli_t](#)

Definition at line 11 of file [JTreeTweaks.h](#).

7.42 src/library/librjungle.cpp File Reference

```
#include <ctime> #include <string> #include <iostream>
#include <fstream> #include <sstream> #include "config.-  
h" #include "treedefs.h" #include "librjungle.h" #include  
"TestClass.h" #include "Exception.h" #include "gzstream.-  
h" #include "CmplFct.h" #include "RJungleIO.h" #include "R-  
JungleGen.h" Include dependency graph for librjungle.cpp:
```



Functions

- void [randomJungle](#) ([RJunglePar](#) par)
- void [__cdecl testLibrandomjungle](#) ()
- [RJunglePar](#) [__cdecl initRJunglePar](#) ()

7.42.1 Function Documentation

7.42.1.1 RJunglePar __cdecl initRJunglePar ()

Definition at line 151 of file librjungle.cpp.

7.42.1.2 void randomJungle (RJunglePar par)

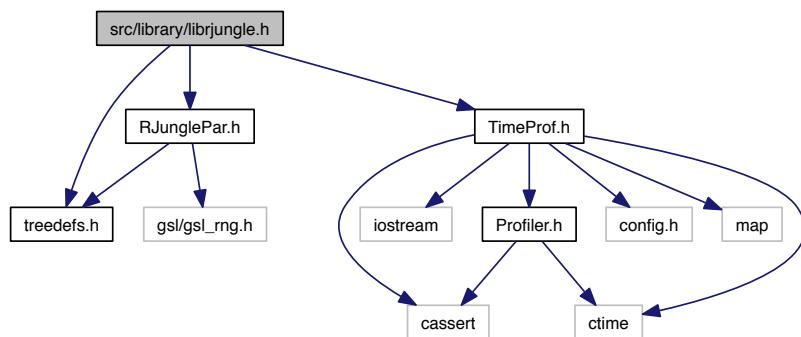
Definition at line 34 of file librjungle.cpp.

7.42.1.3 void __cdecl testLibrandomjungle ()

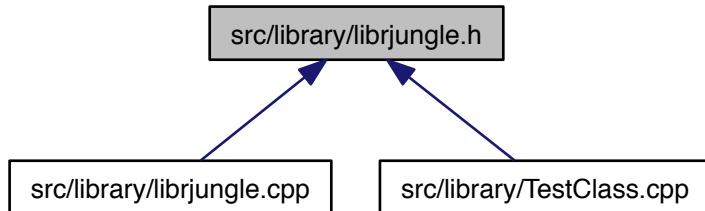
Definition at line 126 of file librjungle.cpp.

7.43 src/library/librjungle.h File Reference

```
#include "treedefs.h" #include "RJunglePar.h" #include "-TimeProf.h" Include dependency graph for librjungle.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- void [__cdecl randomJungle \(RJunglePar\)](#)
- [RJunglePar __cdecl initRJunglePar \(\)](#)
- void [__cdecl testLibrandomjungle \(\)](#)

7.43.1 Function Documentation

7.43.1.1 RJunglePar __cdecl initRJunglePar ()

Definition at line 151 of file librjungle.cpp.

7.43.1.2 void __cdecl randomJungle (RJunglePar)

Definition at line 34 of file librjungle.cpp.

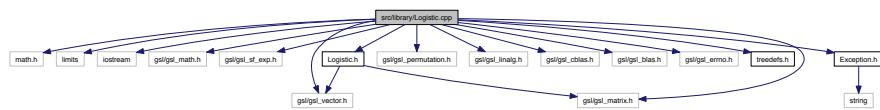
7.43.1.3 void __cdecl testLibrandomjungle ()

Definition at line 126 of file librjungle.cpp.

7.44 src/library/Logistic.cpp File Reference

```
#include <math.h> #include <limits> #include <iostream>
#include <gsl/gsl_math.h> #include <gsl/gsl_sf_exp.h> ×
#include <gsl/gsl_vector.h> #include <gsl/gsl_matrix.-h>
#include <gsl/gsl_permutation.h> #include <gsl/gsl_linalg.h>
#include <gsl/gsl_cblas.h> #include <gsl/gsl_blas.h>
#include <gsl/gsl_errno.h> #include "treedefs.h"
```

```
#include "Logistic.h" #include "Exception.h" Include dependency
graph for Logistic.cpp:
```



Defines

- `#define MPFR_MYPREC 512`

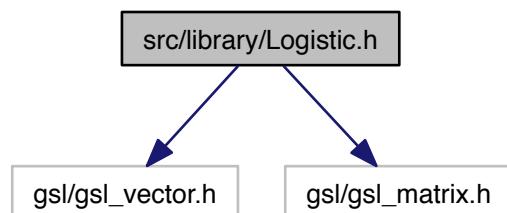
7.44.1 Define Documentation

7.44.1.1 `#define MPFR_MYPREC 512`

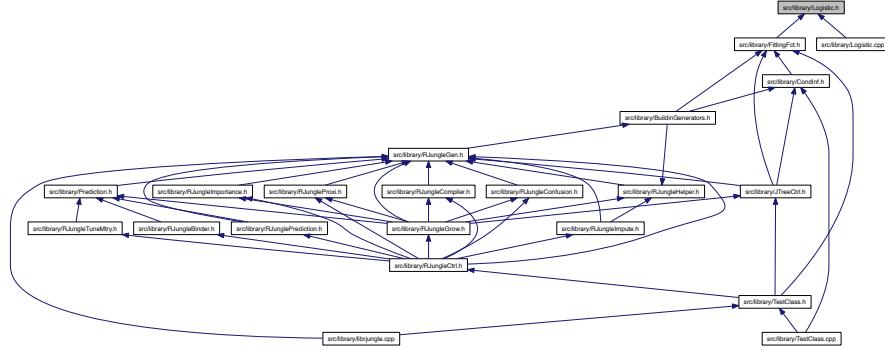
Definition at line 29 of file `Logistic.cpp`.

7.45 src/library/Logistic.h File Reference

```
#include <gsl/gsl_vector.h> #include <gsl/gsl_matrix.h> ×
Include dependency graph for Logistic.h:
```



This graph shows which files directly or indirectly include this file:



Classes

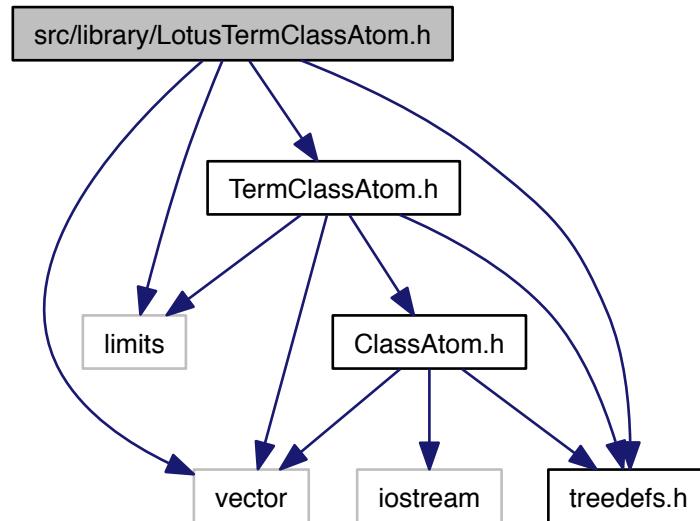
- class [Logistic](#)

7.46 src/library/LotusTermClassAtom.cpp File Reference

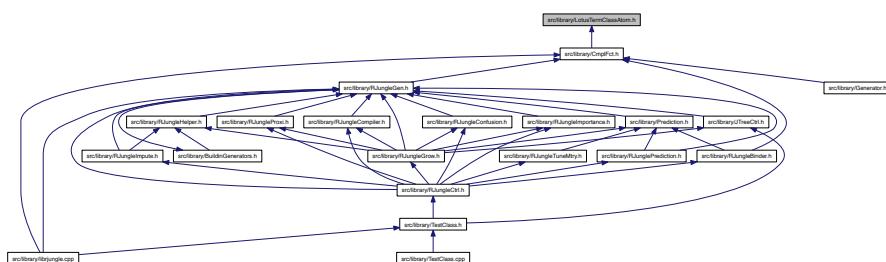
7.47 src/library/LotusTermClassAtom.h File Reference

```
#include <vector> #include <limits> #include "TermClassAtom.h" #include "treedefs.h" Include dependency graph for LotusTerm-
```

ClassAtom.h:



This graph shows which files directly or indirectly include this file:

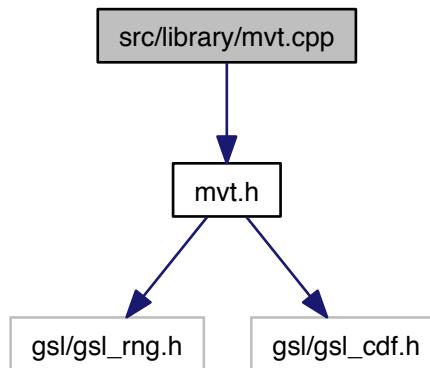


Classes

- class `LotusTermClassAtom< T, C >`

7.48 src/library/mvt.cpp File Reference

#include "mvt.h" Include dependency graph for mvt.cpp:



Functions

- int [gsl_rng_uniform_fortran_](#)(double *x, gsl_rng **r, int pointerwaste)
- int [gsl_rng_uniform_fortran__](#)(double *x, gsl_rng **r, int pointerwaste)

7.48.1 Function Documentation

7.48.1.1 int [gsl_rng_uniform_fortran_](#)(double * x, gsl_rng ** r, int *pointerwaste*)

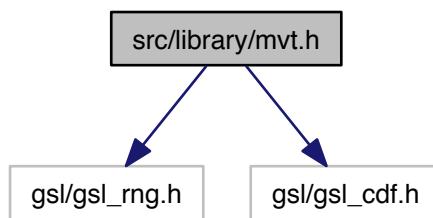
Definition at line 13 of file mvt.cpp.

7.48.1.2 int [gsl_rng_uniform_fortran__](#)(double * x, gsl_rng ** r, int *pointerwaste*)

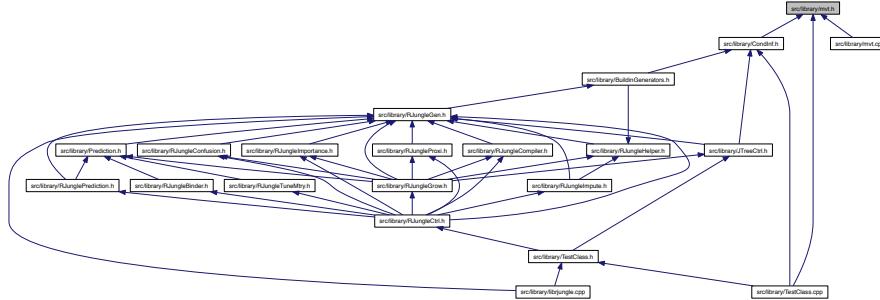
Definition at line 19 of file mvt.cpp.

7.49 src/library/mvt.h File Reference

```
#include <gsl/gsl_rng.h> #include <gsl/gsl_cdf.h> Include dependency graph for mvt.h:
```



This graph shows which files directly or indirectly include this file:



Functions

- int [gsl_rng_uniform_fortran_](#) (double **x, gsl_rng **r, int pointerwaste)
- int [gsl_rng_uniform_fortran_](#) (double **x, gsl_rng **r, int pointerwaste)
- void [mvtdst_](#) (int *n, int *nu, double *lower, double *upper, int *infin, double *corr, double *delta, int *maxpts, double *abseps, double *releps, double *error, double *value, int *inform, gsl_rng **r)

7.49.1 Function Documentation

7.49.1.1 `int gsl_rng_uniform_fortran_(double * x, gsl_rng ** r, int pointerwaste)`

Definition at line 13 of file mvt.cpp.

7.49.1.2 `int gsl_rng_uniform_fortran__(double * x, gsl_rng ** r, int pointerwaste)`

Definition at line 19 of file mvt.cpp.

7.49.1.3 `void mvtdst_(int * n, int * nu, double * lower, double * upper, int * infin, double * corr, double * delta, int * maxpts, double * abseps, double * releps, double * error, double * value, int * inform, gsl_rng ** r)`

7.50 src/library/mvtfortran.f File Reference

Functions

- subroutine `MVTDST` (N, NU, LOWER, UPPER, INFIN, CORREL, DELTA, MAXPTS, ABSEPS, RELEPS, ERROR, VALUE, INFORM, GSLPOINTER)
- subroutine `MVSUBR` (N, W, NF, F)
- subroutine `MVSPCL` (ND, NU, A, B, DL, COV, INFI, SNU, VL, ER, INFORM)
- subroutine `MVVLSB` (N, W, R, DL, INFI, A, B, COV, Y, DI, EI, ND, VALUE)
- subroutine `MVSORT` (N, LOWER, UPPER, DELTA, CORREL, INFIN, Y, PIVOT, ND, A, B, DL, COV, INFI, INFORM)
- DOUBLE PRECISION function `MVTDNS` (NU, X)
- subroutine `MVLIMS` (A, B, INFIN, LOWER, UPPER)
- subroutine `MVSSWP` (X, Y)
- subroutine `MVSWAP` (P, Q, A, B, D, INFIN, N, C)
- DOUBLE PRECISION function `MVPHI` (Z)
- DOUBLE PRECISION function `MVPHNV` (P)
- DOUBLE PRECISION function `MVBVN` (LOWER, UPPER, INFIN, CORREL)
- DOUBLE PRECISION function `MVBVU` (SH, SK, R)
- DOUBLE PRECISION function `MVSTD` (NU, T)
- DOUBLE PRECISION function `MVBVT` (NU, LOWER, UPPER, INFIN, CORREL)
- DOUBLE PRECISION function `MVBVTC` (NU, L, U, INFIN, RHO)
- double precision function `mvbvtl` (nu, dh, dk, r)
- DOUBLE PRECISION function `MVCHNV` (N, P)
- DOUBLE PRECISION function `MVCHNC` (LKN, N, P, R)
- subroutine `MVKBRV` (NDIM, MINVLS, MAXVLS, NF, FUNSUB, ABSEPS, RELEPS, ABSERR, FINEST, INFORM, GSLPOINTER)
- subroutine `MVKRSV` (NDIM, KL, VALUES, PRIME, VK, NF, FUNSUB, X, R, PR, FS, GSLPOINTER)
- DOUBLE PRECISION function `MVUNI` (GSLPOINTER)

7.50.1 Function Documentation

7.50.1.1 DOUBLE PRECISION function **MVBVN** (DOUBLE PRECISION, dimension(*) *LOWER*,
DOUBLE PRECISION, dimension(*) *UPPER*, INTEGER, dimension(*) *INFIN*, DOUBLE
PRECISION *CORREL*)

Definition at line 678 of file mvtfortran.f.

7.50.1.2 DOUBLE PRECISION function **MVBVT** (INTEGER *NU*, DOUBLE PRECISION,
dimension(*) *LOWER*, DOUBLE PRECISION, dimension(*) *UPPER*, INTEGER,
dimension(*) *INFIN*, DOUBLE PRECISION *CORREL*)

Definition at line 877 of file mvtfortran.f.

7.50.1.3 DOUBLE PRECISION function **MVBVTC** (INTEGER *NU*, DOUBLE PRECISION,
dimension(*) *L*, DOUBLE PRECISION, dimension(*) *U*, INTEGER, dimension(*) *INFIN*,
DOUBLE PRECISION *RHO*)

Definition at line 928 of file mvtfortran.f.

7.50.1.4 double precision function **mvbvtl** (integer *nu*, double precision *dh*, double precision
dk, double precision *r*)

Definition at line 981 of file mvtfortran.f.

7.50.1.5 DOUBLE PRECISION function **MVBVU** (DOUBLE PRECISION *SH*, DOUBLE
PRECISION *SK*, DOUBLE PRECISION *R*)

Definition at line 723 of file mvtfortran.f.

7.50.1.6 DOUBLE PRECISION function **MVCHNC** (DOUBLE PRECISION *LKN*, INTEGER *N*,
DOUBLE PRECISION *P*, DOUBLE PRECISION *R*)

Definition at line 1123 of file mvtfortran.f.

7.50.1.7 DOUBLE PRECISION function **MVCHNV** (INTEGER *N*, DOUBLE PRECISION *P*)

Definition at line 1073 of file mvtfortran.f.

7.50.1.8 subroutine **MVKBRV** (INTEGER *NDIM*, INTEGER *MINVLS*, INTEGER *MAXVLS*,
INTEGER *NF*, , external *FUNSUB*, DOUBLE PRECISION *ABSEPS*, DOUBLE PRECISION
RELEPS, DOUBLE PRECISION *ABSERR*, DOUBLE PRECISION, dimension(*) *FINEST*,
INTEGER *INFORM*, *GSLPOINTER*)

Definition at line 1189 of file mvtfortran.f.

7.50.1.9 subroutine **MVKRSV** (INTEGER *NDIM*, INTEGER *KL*, DOUBLE PRECISION,
dimension(*) *VALUES*, INTEGER *PRIME*, DOUBLE PRECISION, dimension(*) *VK*,
INTEGER *NF*, *FUNSUB*, DOUBLE PRECISION, dimension(*) *X*, DOUBLE PRECISION,
dimension(*) *R*, INTEGER, dimension(*) *PR*, DOUBLE PRECISION, dimension(*) *FS*,
GSLPOINTER)

Definition at line 1466 of file mvtfortran.f.

7.50.1.10 subroutine **MVLIMS** (DOUBLE PRECISION *A*, DOUBLE PRECISION *B*, INTEGER
INFIN, DOUBLE PRECISION *LOWER*, DOUBLE PRECISION *UPPER*)

Definition at line 470 of file mvtfortran.f.

7.50.1.11 DOUBLE PRECISION function **MVPHI** (DOUBLE PRECISION *Z*)

Definition at line 519 of file mvtfortran.f.

7.50.1.12 DOUBLE PRECISION function **MVPHNV** (DOUBLE PRECISION *P*)

Definition at line 569 of file mvtfortran.f.

7.50.1.13 subroutine **MVSORT** (INTEGER *N*, DOUBLE PRECISION, dimension(*) *LOWER*,
DOUBLE PRECISION, dimension(*) *UPPER*, DOUBLE PRECISION, dimension(*)
DELTA, DOUBLE PRECISION, dimension(*) *CORREL*, INTEGER, dimension(*) *INFIN*,
DOUBLE PRECISION, dimension(*) *Y*, LOGICAL *PIVOT*, INTEGER *ND*, DOUBLE
PRECISION, dimension(*) *A*, DOUBLE PRECISION, dimension(*) *B*, DOUBLE
PRECISION, dimension(*) *DL*, DOUBLE PRECISION, dimension(*) *COV*, INTEGER,
dimension(*) *INFI*, INTEGER *INFORM*)

Definition at line 265 of file mvtfortran.f.

7.50.1.14 subroutine **MVSPCL** (INTEGER *ND*, INTEGER *NU*, DOUBLE PRECISION,
dimension(*) *A*, DOUBLE PRECISION, dimension(*) *B*, DOUBLE PRECISION,
dimension(*) *DL*, DOUBLE PRECISION, dimension(*) *COV*, INTEGER, dimension(*)
INFI, DOUBLE PRECISION *SNU*, DOUBLE PRECISION *VL*, DOUBLE PRECISION *ER*,
INTEGER *INFORM*)

Definition at line 136 of file mvtfortran.f.

7.50.1.15 subroutine MVSSWP (DOUBLE PRECISION X, DOUBLE PRECISION Y)

Definition at line 482 of file mvtfortran.f.

7.50.1.16 DOUBLE PRECISION function MVSTDT (INTEGER NU, DOUBLE PRECISION T)

Definition at line 841 of file mvtfortran.f.

7.50.1.17 subroutine MVSUBR (INTEGER N, DOUBLE PRECISION, dimension(*) W, INTEGER NF, DOUBLE PRECISION, dimension(*) F)

Definition at line 100 of file mvtfortran.f.

7.50.1.18 subroutine MVSWAP (INTEGER P, INTEGER Q, DOUBLE PRECISION, dimension(*) A, DOUBLE PRECISION, dimension(*) B, DOUBLE PRECISION, dimension(*) D, INTEGER, dimension(*) INFIN, INTEGER N, DOUBLE PRECISION, dimension(*) C)

Definition at line 489 of file mvtfortran.f.

7.50.1.19 DOUBLE PRECISION function MVTDNS (INTEGER NU, DOUBLE PRECISION X)

Definition at line 448 of file mvtfortran.f.

7.50.1.20 subroutine MVTDST (INTEGER N, INTEGER NU, DOUBLE PRECISION, dimension(*) LOWER, DOUBLE PRECISION, dimension(*) UPPER, INTEGER, dimension(*) INFIN, DOUBLE PRECISION, dimension(*) CORREL, DOUBLE PRECISION, dimension(*) DELTA, INTEGER MAXPTS, DOUBLE PRECISION ABSEPS, DOUBLE PRECISION RELEPS, DOUBLE PRECISION ERROR, DOUBLE PRECISION VALUE, INTEGER INFORM, GSLPOINTER)

Definition at line 8 of file mvtfortran.f.

7.50.1.21 DOUBLE PRECISION function MVUNI (GSLPOINTER)

Definition at line 1514 of file mvtfortran.f.

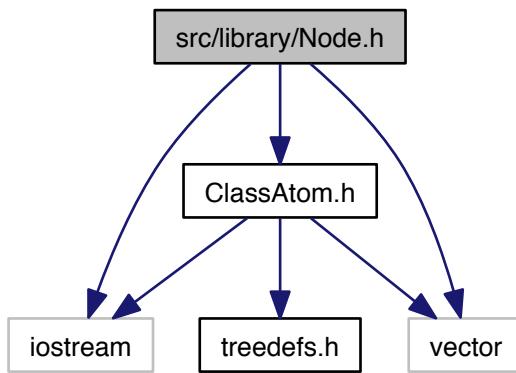
7.50.1.22 subroutine MVVLSB (INTEGER N, DOUBLE PRECISION, dimension(*) W, DOUBLE PRECISION R, DOUBLE PRECISION, dimension(*) DL, INTEGER, dimension(*) INFI, DOUBLE PRECISION, dimension(*) A, DOUBLE PRECISION, dimension(*) B, DOUBLE PRECISION, dimension(*) COV, DOUBLE PRECISION, dimension(*) Y, DOUBLE PRECISION DI, DOUBLE PRECISION EI, INTEGER ND, DOUBLE PRECISION VALUE)

Definition at line 213 of file mvtfortran.f.

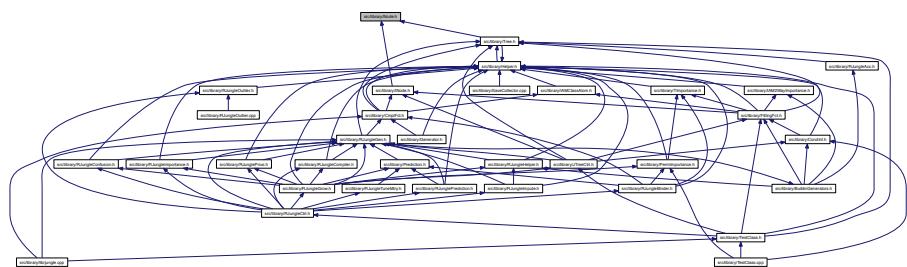
7.51 src/library/Node.cpp File Reference

7.52 src/library/Node.h File Reference

```
#include <iostream>  #include <vector>  #include "ClassAtom.h" Include dependency graph for Node.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Node< T, C >](#)

Functions

- template<class T , class C >
std::ostream & **operator<<** (std::ostream &os, Node< T, C > &node)

7.52.1 Function Documentation

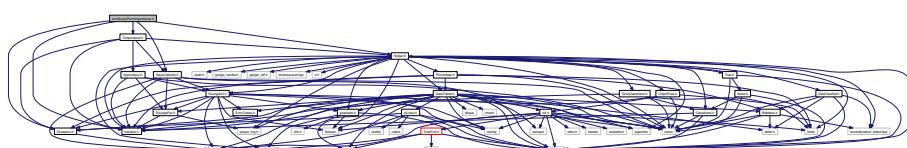
7.52.1.1 template<class T , class C > std::ostream& operator<< (std::ostream & os,
Node< T, C > & node)

Definition at line 81 of file Node.h.

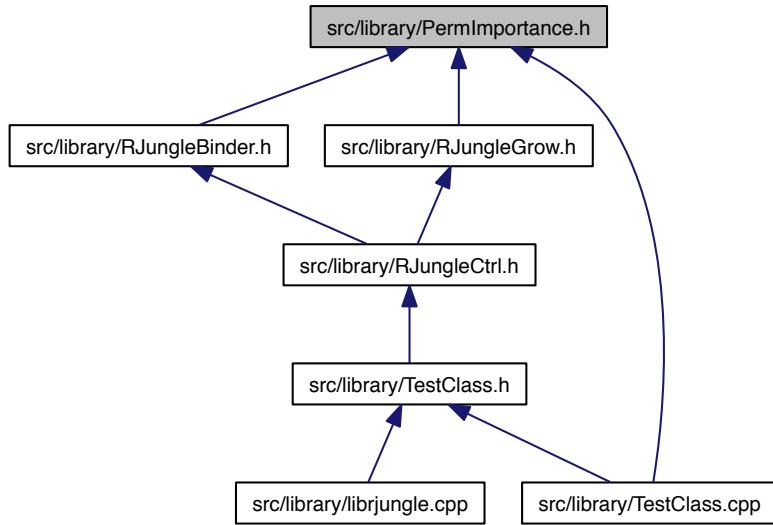
7.53 src/library/PermlImportance.cpp File Reference

7.54 src/library/PermlImportance.h File Reference

```
#include "treedefs.h"    #include "Helper.h"    #include "-  
TImportance.h"    #include "Exception.h"    #include "Save-  
Collector.h" Include dependency graph for PermlImportance.h:
```



This graph shows which files directly or indirectly include this file:



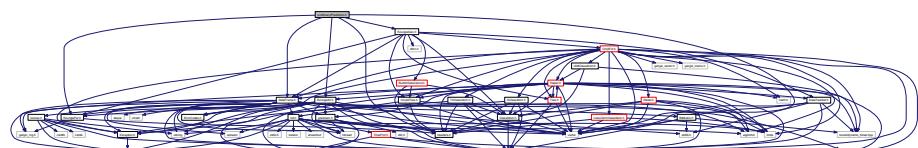
Classes

- class [PermlImportance< T >](#)

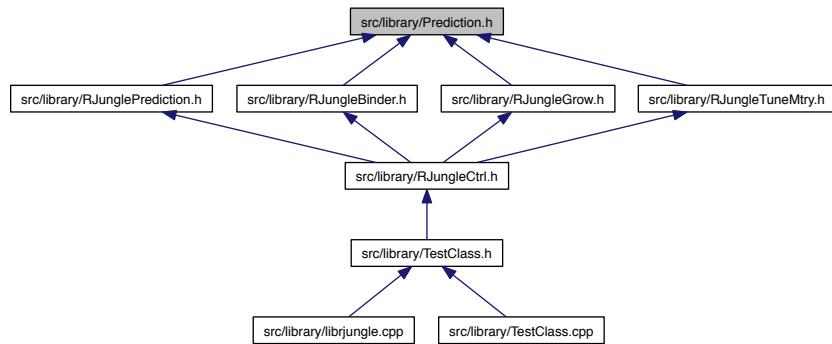
7.55 src/library/Prediction.cpp File Reference

7.56 src/library/Prediction.h File Reference

```
#include <vector> #include "treedefs.h" #include "Data-Frame.h" #include "RJunglePar.h" #include "RJungleIO.h" #include "RJungleGen.h" Include dependency graph for Prediction.h:
```



This graph shows which files directly or indirectly include this file:

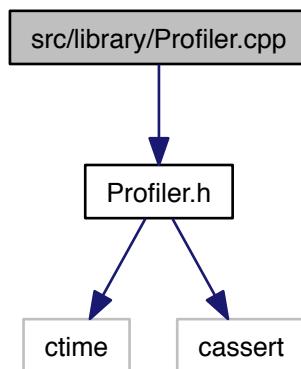


Classes

- class [Prediction< T >](#)

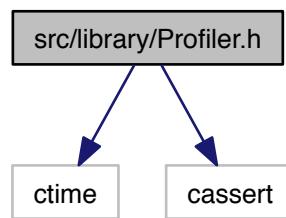
7.57 src/library/Profiler.cpp File Reference

#include "Profiler.h" Include dependency graph for Profiler.cpp:

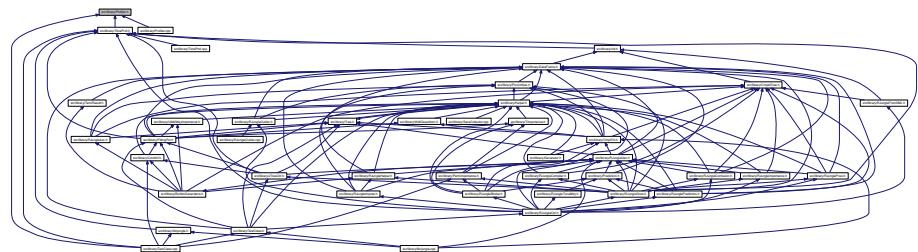


7.58 src/library/Profiler.h File Reference

```
#include <ctime> #include <cassert> Include dependency graph for -  
Profiler.h:
```



This graph shows which files directly or indirectly include this file:



Classes

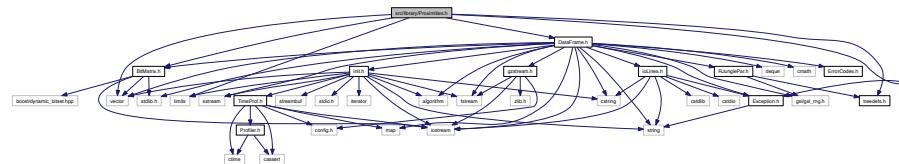
- class [Profiler](#)

7.59 src/library/Proximities.cpp File Reference

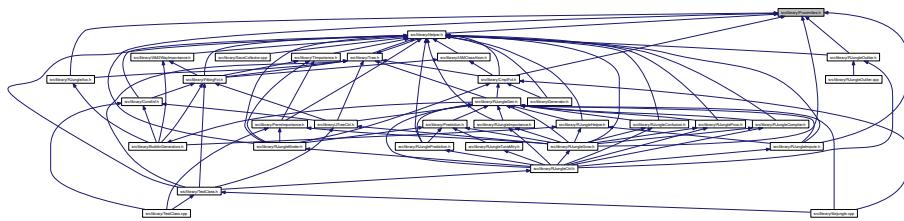
7.60 src/library/Proximities.h File Reference

```
#include <vector> #include <limits> #include "DataFrame.-  
h" #include "Exception.h" #include "BitMatrix.h" #include
```

"treedefs.h" Include dependency graph for Proximities.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Proximities< T >

Functions

- template<class T >
std::ostream & operator<< (std::ostream &os, Proximities< T > &proximities)

7.60.1 Function Documentation

7.60.1.1 template<class T> std::ostream& operator<<(std::ostream & os, Proximities<T> & proximities) [inline]

Definition at line 277 of file Proximities.h.

7.61 src/library/RJungleAcc.cpp File Reference

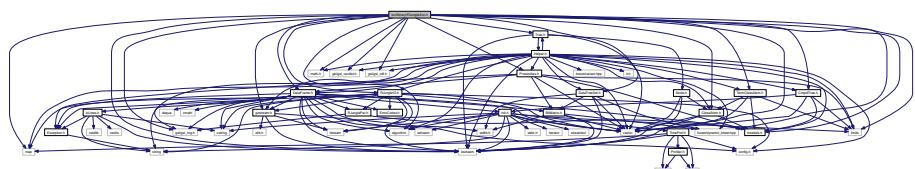
7.62 src/library/RJungleAcc.h File Reference

```
#include <iostream> #include <vector> #include <map> x  
#include <limits> #include <algorithm> #include <math.h>
```

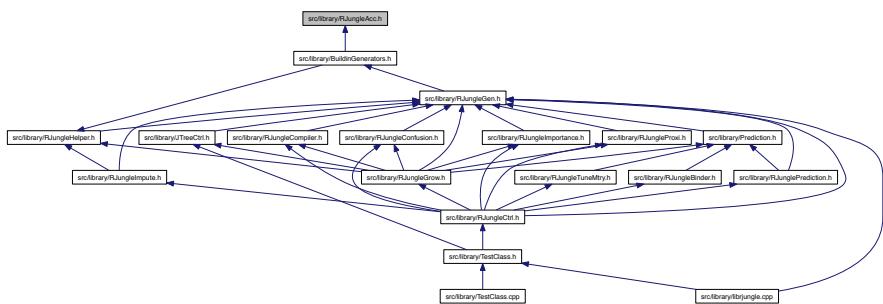
```

h> #include <gsl/gsl_rng.h> #include <gsl/gsl_randist.h>
h> #include <gsl/gsl_cdf.h> #include <boost/dynamic_bitset.hpp> #include "RJungleIO.h" #include "Proximities.h"
h> #include "DataFrame.h" #include "DataTreeSet.h" #include "ClassAtom.h" #include "TermClassAtom.h" #include "Tree.h"
h> #include "CmpldTree.h" #include "treedefs.h" #include "gzstream.h" Include dependency graph for RJungleAcc.h:

```



This graph shows which files directly or indirectly include this file:



Classes

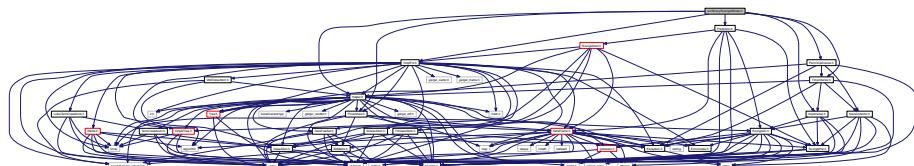
- class [RJungleAcc< T >](#)

7.63 src/library/RJungleBinder.cpp File Reference

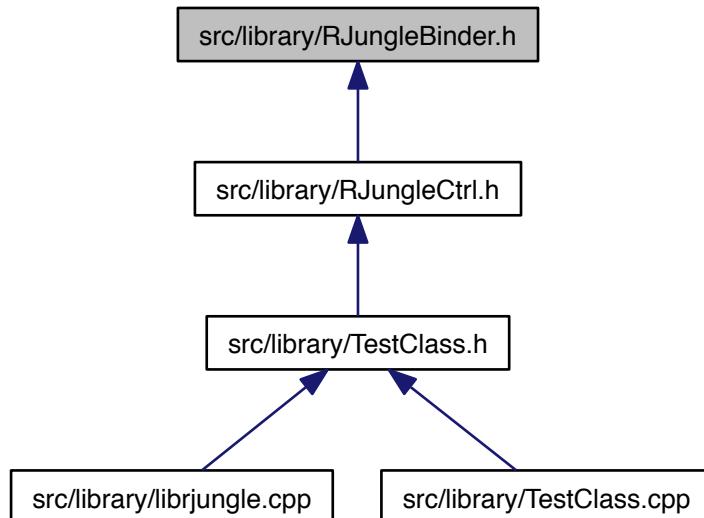
7.64 src/library/RJungleBinder.h File Reference

```
#include "Prediction.h" #include "Importance.h" #include "TImportance.h" #include "PermImportance.h" #include "-Helper.h" #include "CmplFct.h" #include "Proximities.h" x
```

Include dependency graph for RJungleBinder.h:



This graph shows which files directly or indirectly include this file:



Classes

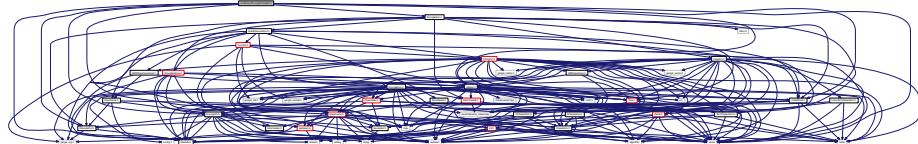
- class [RJungleBinder< T >](#)

7.65 src/library/RJungleCompiler.cpp File Reference

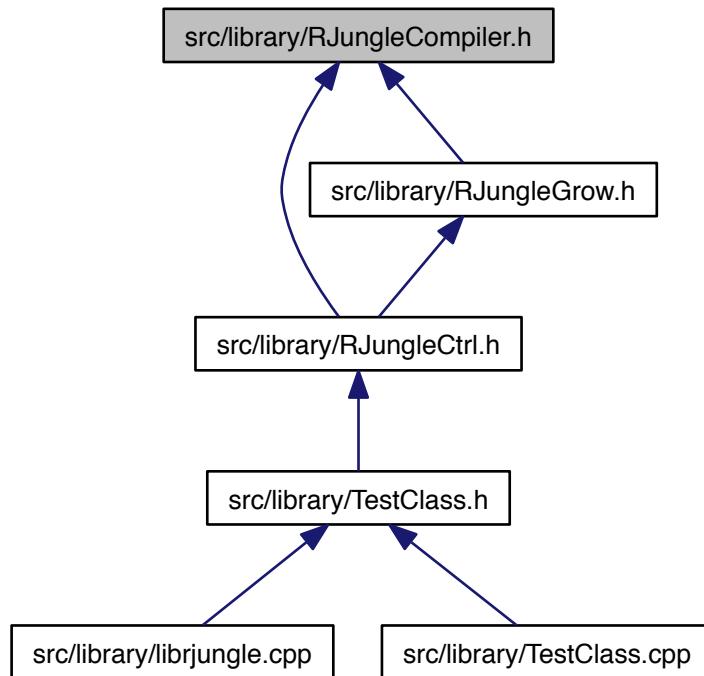
7.66 src/library/RJungleCompiler.h File Reference

```
#include <iostream> #include <vector> #include "RJungle-
```

```
Gen.h"    #include "RJungleIO.h"    #include "RJunglePar.h"  
#include "CmpldTree.h" #include "Helper.h" #include "treedefs.-  
h" Include dependency graph for RJungleCompiler.h:
```



This graph shows which files directly or indirectly include this file:



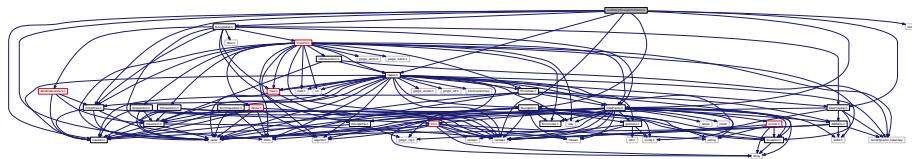
Classes

- class [RJungleCompiler< T >](#)

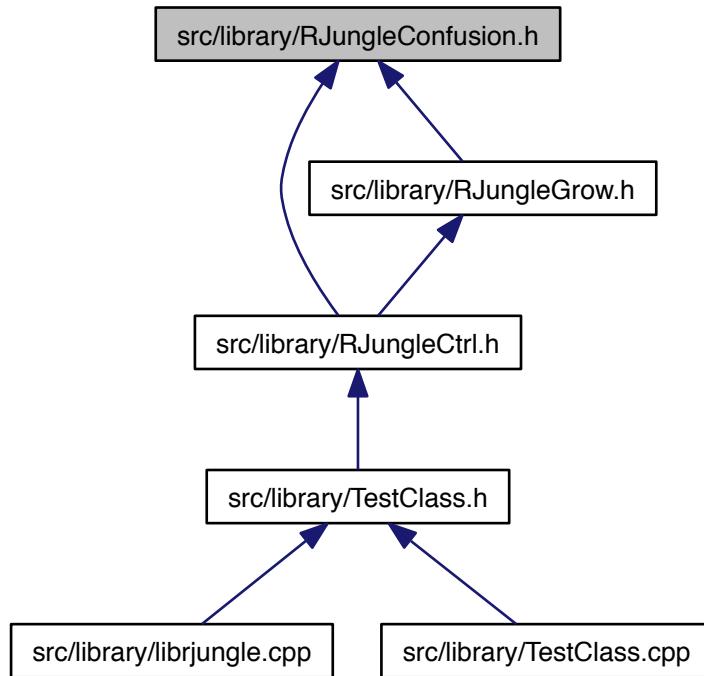
7.67 src/library/RJungleConfusion.cpp File Reference

7.68 src/library/RJungleConfusion.h File Reference

```
#include <iostream> #include <vector> #include <ctime>
#include "RJungleIO.h" #include "RJungleGen.h" #include
"DataFrame.h" #include "DataTreeSet.h" #include "Cmpld-
Tree.h" #include "Helper.h" #include "treedefs.h" Include de-
pendency graph for RJungleConfusion.h:
```



This graph shows which files directly or indirectly include this file:



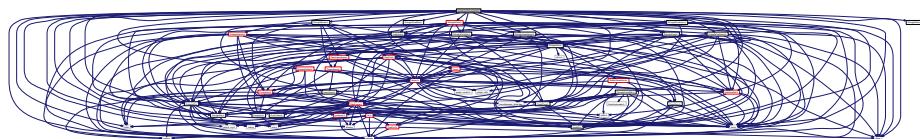
Classes

- class [RJungleConfusion< T >](#)

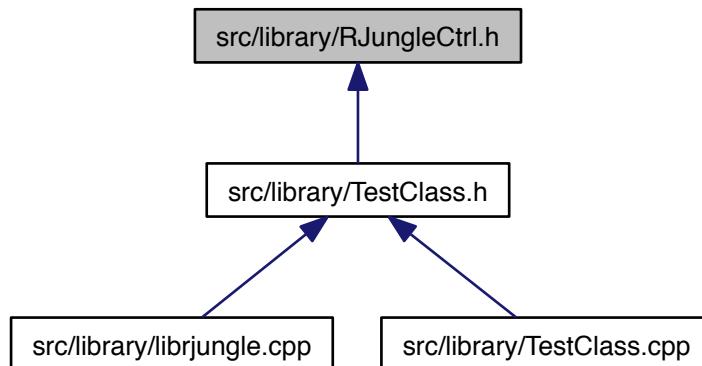
7.69 src/library/RJungleCtrl.cpp File Reference

7.70 src/library/RJungleCtrl.h File Reference

```
#include <iostream> #include <cstring> #include <vector> ×
#include <string>  #include <limits>  #include <ctime> ×
#include <boost/dynamic_bitset.hpp>  #include <gsl/gsl-
rng.h>  #include <gsl/gsl_randist.h>  #include <gsl/gsl-
_cdf.h>  #include "treedefs.h"  #include "RJungleIO.h" ×
#include "RJunglePar.h"  #include "RJungleGen.h"  #include
"RJungleFromXML.h"  #include "RJungleCompiler.h"  #include
"RJungleConfusion.h" #include "RJunglePrediction.h" #include
"RJungleImportance.h"  #include "RJungleBinder.h"  #include
"RJungleGrow.h"  #include "RJungleProxi.h"  #include "R-
JungleImpute.h"  #include "RJungleTuneMtry.h"  #include "R-
JungleProto.h" #include "RJungleOutlier.h" #include "Time-
Prof.h" Include dependency graph for RJungleCtrl.h:
```



This graph shows which files directly or indirectly include this file:



Classes

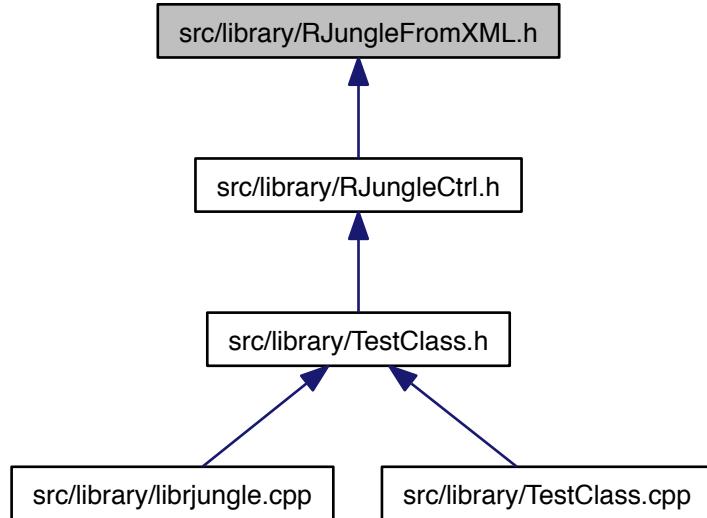
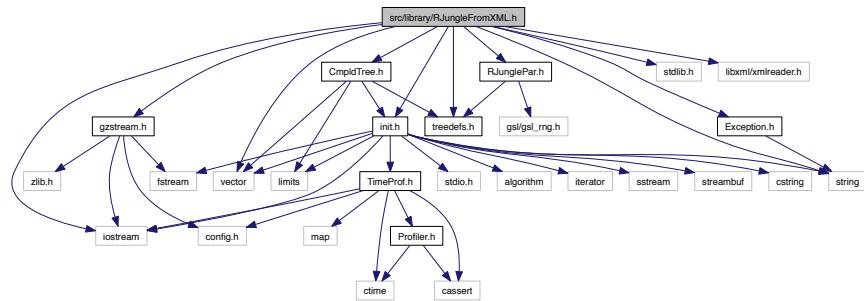
- class [RJungleCtrl< T >](#)

7.71 src/library/RJungleFromXML.cpp File Reference

7.72 src/library/RJungleFromXML.h File Reference

```
#include <iostream> #include <stdlib.h> #include <vector> x
#include <string> #include <libxml/xmlreader.h> #include
"init.h" #include "Exception.h" #include "gzstream.h" x
#include "RJunglePar.h" #include "CmpldTree.h" #include
```

"treedefs.h" Include dependency graph for RJungleFromXML.h:



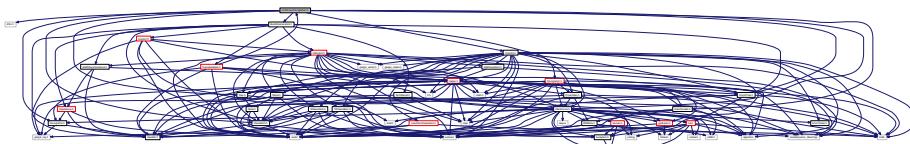
Classes

- class [RJungleFromXML< T >](#)

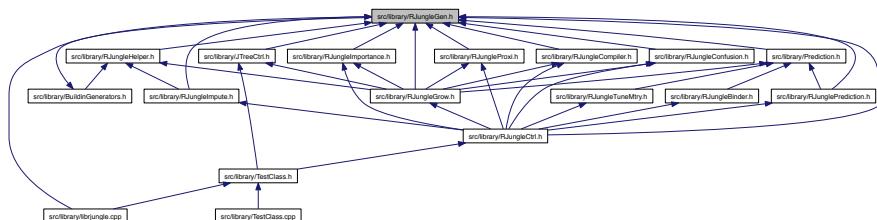
7.73 src/library/RJungleGen.cpp File Reference

7.74 src/library/RJungleGen.h File Reference

```
#include <dlfcn.h> #include "CmplFct.h" #include "Buildin-
Generators.h" #include "DataFrame.h" #include "RJungle-
Par.h" #include "ClassAtom.h" #include "CmpldTree.h" ×
#include "DataTreeSet.h" #include "Tree.h" Include dependency
graph for RJungleGen.h:
```



This graph shows which files directly or indirectly include this file:



Classes

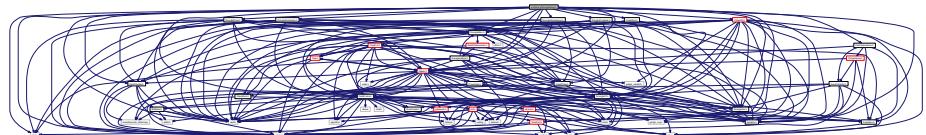
- class [RJungleGen< T >](#)

7.75 src/library/RJungleGrow.cpp File Reference

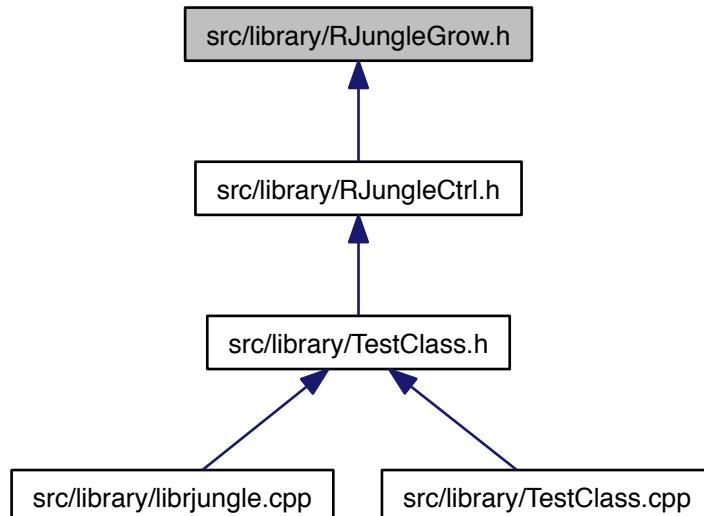
7.76 src/library/RJungleGrow.h File Reference

```
#include <iostream> #include <vector> #include <ctime>
#include "RJunglePar.h" #include "RJungleIO.h" #include "-
RJungleGen.h" #include "RJungleProxi.h" #include "RJungle-
Helper.h" #include "RJungleCompiler.h" #include "RJungle-
Confusion.h" #include "RJungleImportance.h" #include "-
DataFrame.h" #include "DataTreeSet.h" #include "Prediction.-
h" #include "PermImportance.h" #include "Proximities.h" ×
```

```
#include "Helper.h" #include "treedefs.h" #include "JTreeCtrl.h" #include "TimeProf.h" Include dependency graph for R JungleGrow.h:
```



This graph shows which files directly or indirectly include this file:



Classes

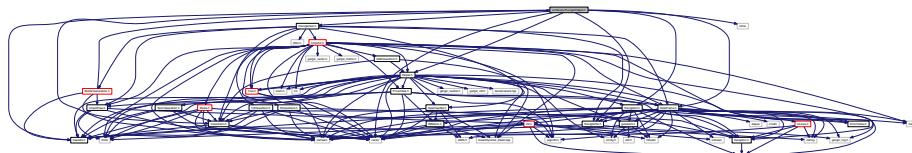
- class [RJungleGrow< T >](#)

7.77 `src/library/RJungleHelper.cpp` File Reference

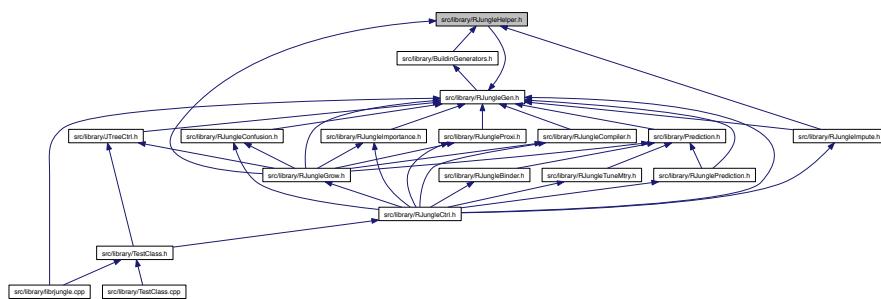
7.78 `src/library/RJungleHelper.h` File Reference

```
#include <iostream> #include <vector> #include <ctime>
```

```
#include "RJunglePar.h" #include "RJungleIO.h" #include "-
RJungleGen.h" #include "DataFrame.h" #include "Helper.h"
#include "treedefs.h" Include dependency graph for RJungleHelper.h:
```



This graph shows which files directly or indirectly include this file:

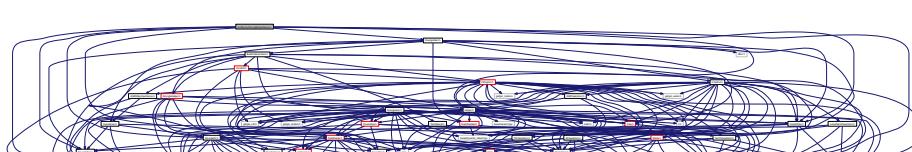


Classes

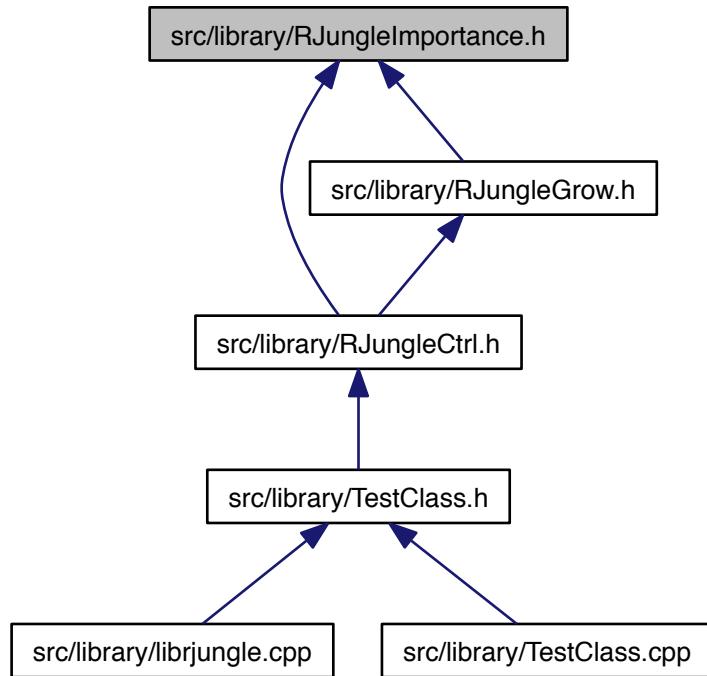
- class [RJungleHelper< T >](#)

7.79 src/library/RJungleImportance.cpp File Reference

```
#include <iostream> #include <vector> #include "RJungle-
Gen.h" #include "RJungleIO.h" #include "RJunglePar.h"
#include "CmpldTree.h" #include "Helper.h" #include "treedefs.-
h" Include dependency graph for RJungleImportance.h:
```



This graph shows which files directly or indirectly include this file:



Classes

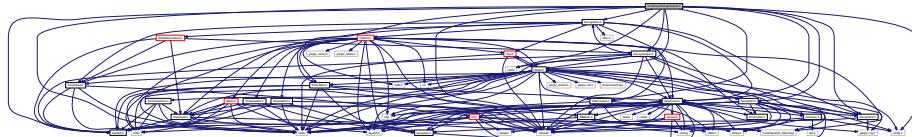
- class [RJungleImportance< T >](#)

7.81 src/library/RJungleImport.cpp File Reference

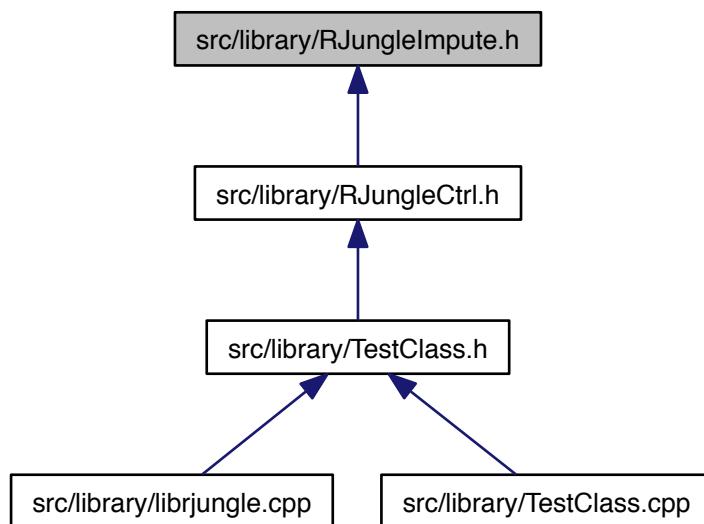
7.82 src/library/RJungleImport.h File Reference

```
#include <iostream> #include <vector> #include <ctime>
#include "config.h" #include "RJunglePar.h" #include "-RJungleIO.h"
#include "RJungleGen.h" #include "RJungleHelper.h" #include "Proximities.h"
#include "DataFrame.h" #include "DataTreeSet.h" #include "Helper.h" #include
```

"treedefs.h" Include dependency graph for RJungleImpute.h:



This graph shows which files directly or indirectly include this file:



Classes

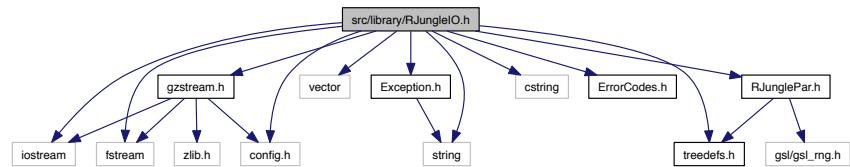
- class [RJungleImpute< T >](#)

7.83 src/library/RJungleIO.cpp File Reference

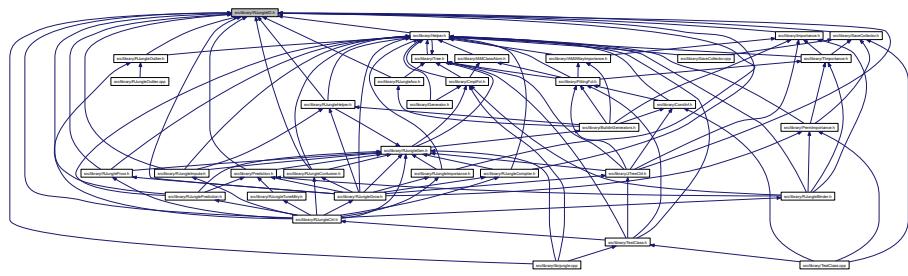
7.84 src/library/RJungleIO.h File Reference

```
#include <iostream> #include <fstream> #include <vector> x
#include <string>    #include <cstring>   #include "Error-
```

Codes.h" #include "config.h" #include "treedefs.h" #include "RJunglePar.h" #include "gzstream.h" #include "Exception.h" Include dependency graph for RJungleIO.h:



This graph shows which files directly or indirectly include this file:

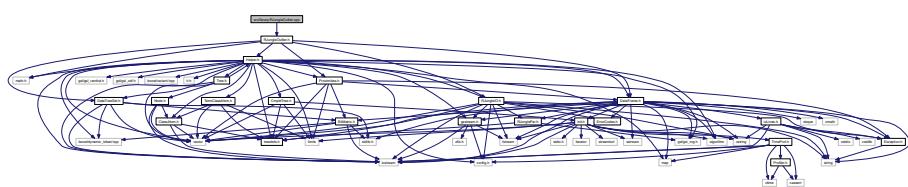


Classes

- class [RJungleIO](#)

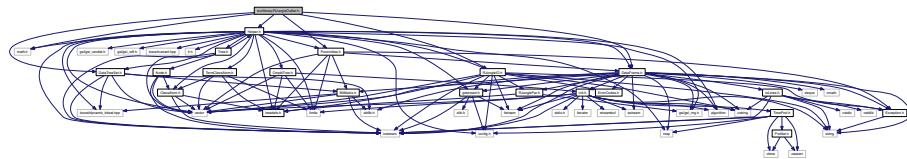
7.85 src/library/RJungleOutlier.cpp File Reference

#include "RJungleOutlier.h" Include dependency graph for RJungleOutlier.cpp:

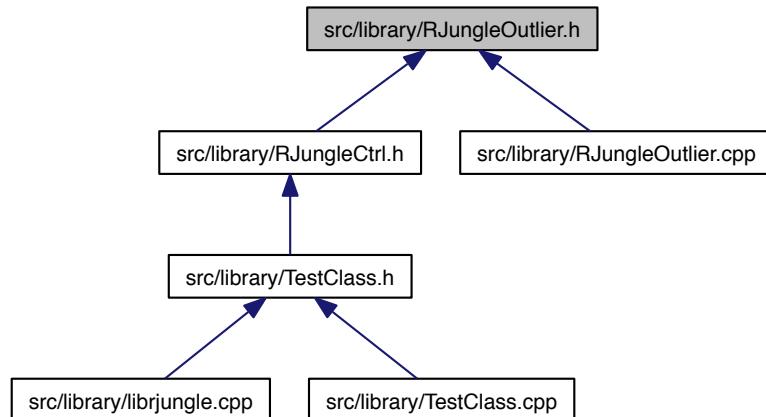


7.86 src/library/RJungleOutlier.h File Reference

```
#include <math.h> #include "Helper.h" #include "RJungle-
IO.h" #include "DataFrame.h" #include "DataTreeSet.h" ×
#include "Proximities.h" Include dependency graph for RJungleOutlier.h:
```



This graph shows which files directly or indirectly include this file:



Classes

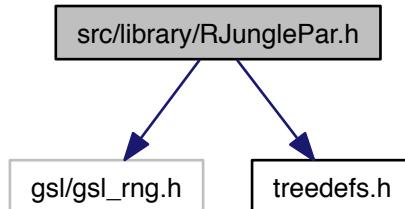
- class [RJungleOutlier](#)

7.87 src/library/RJunglePar.cpp File Reference

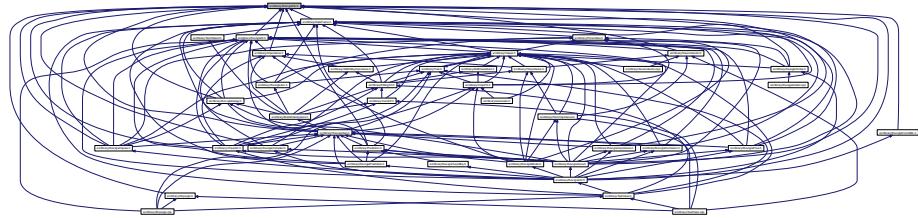
7.88 src/library/RJunglePar.h File Reference

```
#include <gsl/gsl_rng.h> #include "treedefs.h" Include depen-
```

dency graph for RJunglePar.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [RJunglePar](#)

Defines

- #define [bool](#) int

Typedefs

- typedef struct [RJunglePar](#) [RJunglePar](#)

7.88.1 Define Documentation

7.88.1.1 #define bool int

Definition at line 11 of file [RJunglePar.h](#).

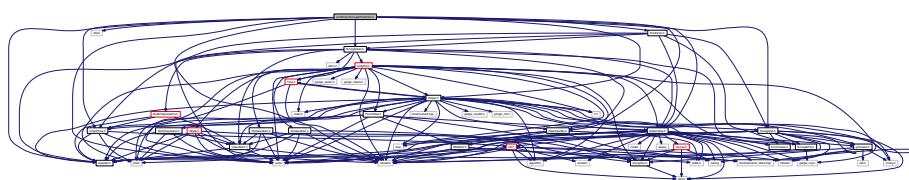
7.88.2 Typedef Documentation

7.88.2.1 typedef struct RJunglePar RJunglePar

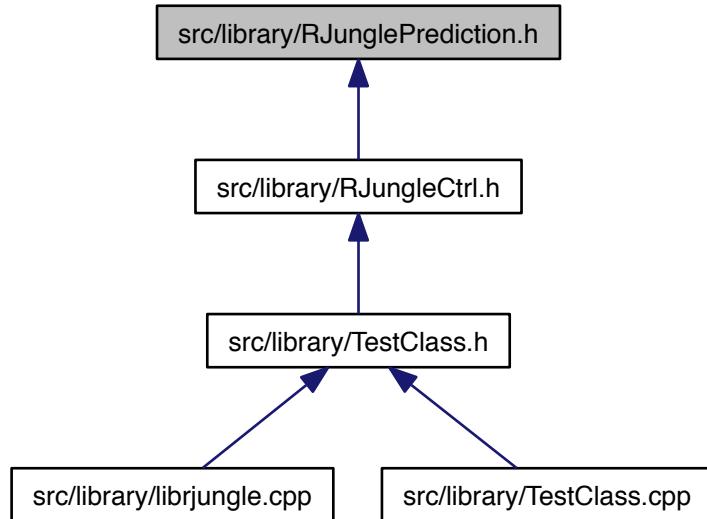
7.89 src/library/RJunglePrediction.cpp File Reference

7.90 src/library/RJunglePrediction.h File Reference

```
#include <iostream> #include <vector> #include <ctime>
#include "RJungleIO.h" #include "RJungleGen.h" #include
>DataFrame.h" #include "DataTreeSet.h" #include "Cmpld-
Tree.h" #include "Helper.h" #include "treedefs.h" #include
"Prediction.h" Include dependency graph for RJunglePrediction.h:
```



This graph shows which files directly or indirectly include this file:



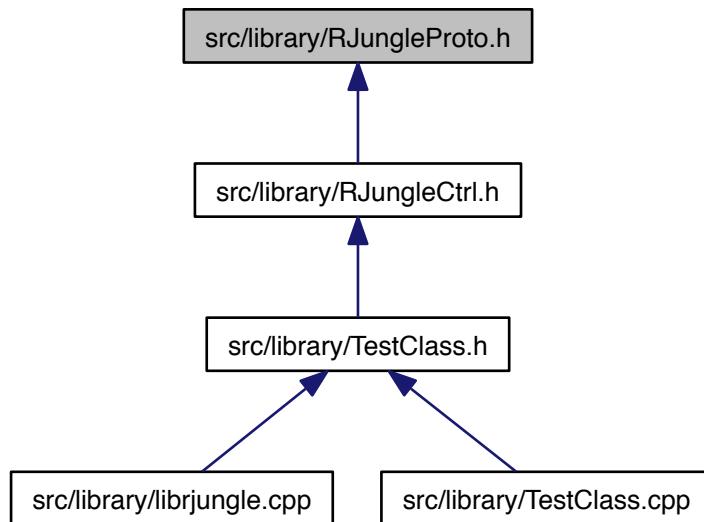
Classes

- class [RJunglePrediction< T >](#)

7.91 src/library/RJungleProto.cpp File Reference

7.92 src/library/RJungleProto.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

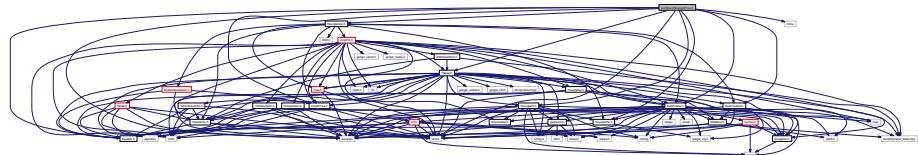
- class [ProtCount](#)
- class [RJungleProto< T >](#)

7.93 src/library/RJungleProxi.cpp File Reference

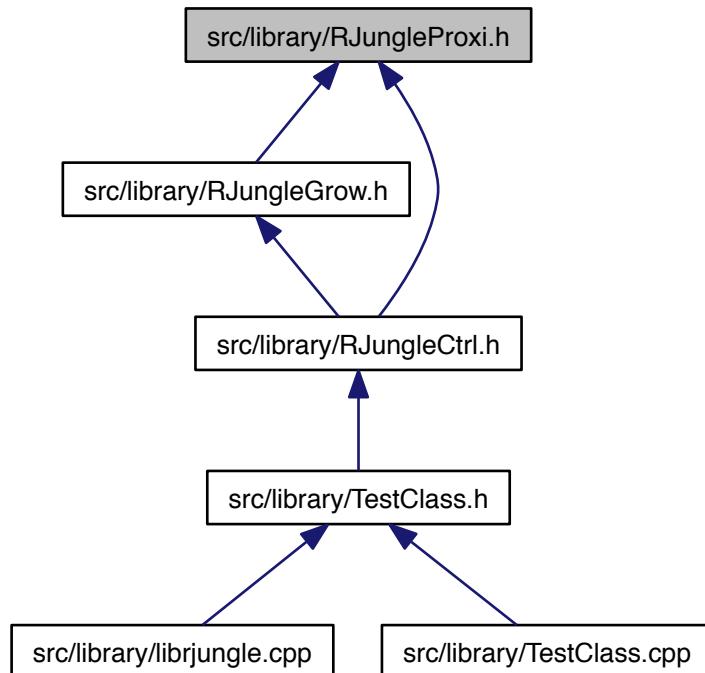
7.94 src/library/RJungleProxi.h File Reference

```
#include <iostream> #include <vector> #include <ctime>
#include "RJunglePar.h" #include "RJungleIO.h" #include "-RJungleGen.h"
#include "DataFrame.h" #include "DataTree-
Set.h" #include "CmpldTree.h" #include "Helper.h" #include
```

"treedefs.h" Include dependency graph for R JungleProxi.h:



This graph shows which files directly or indirectly include this file:



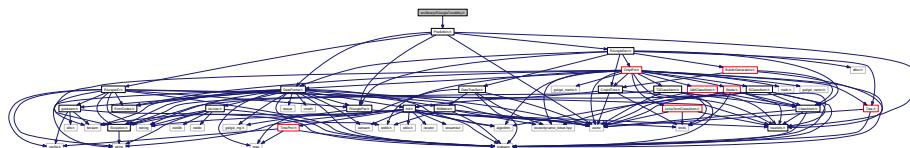
Classes

- class [R JungleProxi< T >](#)

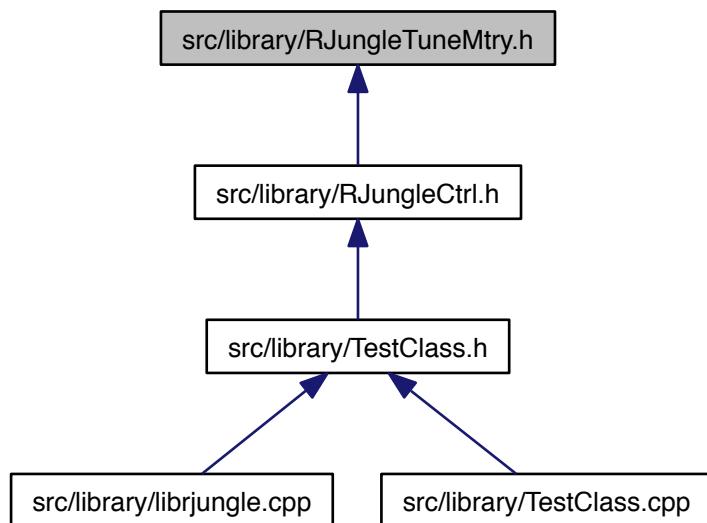
7.95 src/library/R JungleTuneMtry.cpp File Reference

7.96 src/library/RJungleTuneMtry.h File Reference

```
#include "Prediction.h" Include dependency graph for RJungleTuneMtry.h:
```



This graph shows which files directly or indirectly include this file:



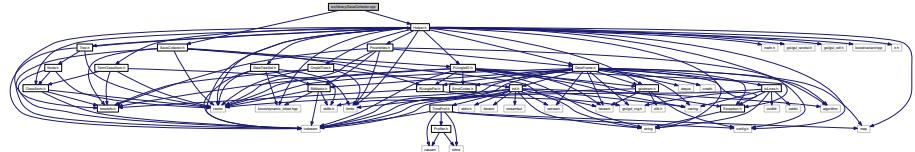
Classes

- class [RJungleTuneMtry< T >](#)

7.97 src/library/SaveCollector.cpp File Reference

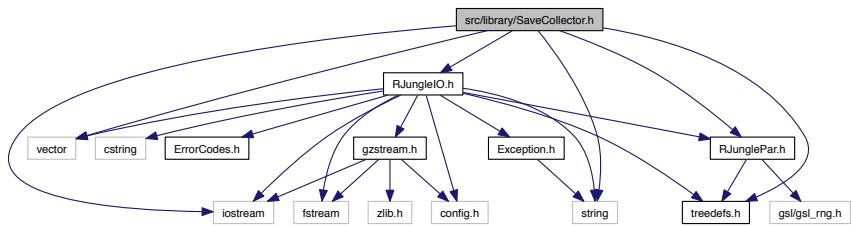
```
#include "SaveCollector.h" #include "Helper.h" Include depen-
```

dency graph for SaveCollector.cpp:

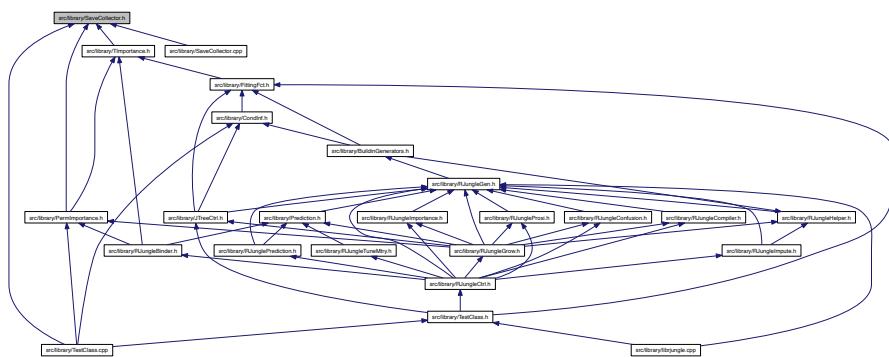


7.98 src/library/SaveCollector.h File Reference

```
#include <iostream> #include <vector> #include <string>
#include "treedefs.h" #include "RJungleIO.h" #include "R-
JunglePar.h" Include dependency graph for SaveCollector.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [SaveCollector](#)

Functions

- std::ostream & `operator<<` (std::ostream &os, const `SaveCollector` &saveCollector)

7.98.1 Function Documentation

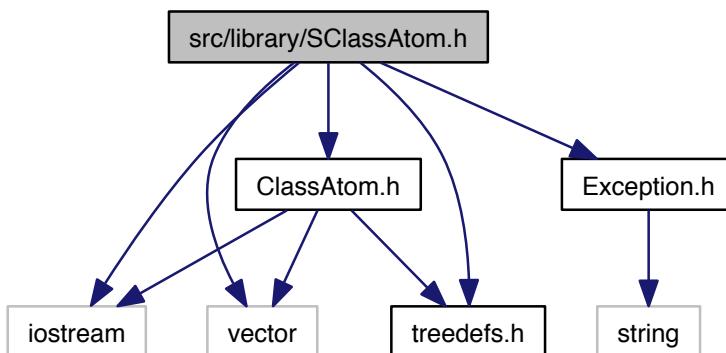
7.98.1.1 `std::ostream& operator<< (std::ostream & os, const SaveCollector & saveCollector) [inline]`

Definition at line 86 of file `SaveCollector.h`.

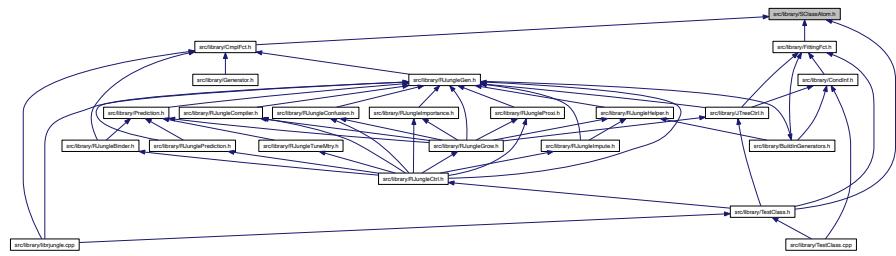
7.99 src/library/SClassAtom.cpp File Reference

7.100 src/library/SClassAtom.h File Reference

```
#include <iostream> #include <vector> #include "treedefs.h"
#include "ClassAtom.h" #include "Exception.h" Include dependency graph for SClassAtom.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class SClassAtom< T >

Defines

- `#define Tvector std::vector<T >`
 - `#define CCvector std::vector<uli_t >`

Functions

- template<class T>
std::ostream & operator<< (std::ostream &os, const SClassAtom< T > &s-
ClassAtom)

7.100.1 Define Documentation

7.100.1.1 #define CCvector std::vector<uli t >

Definition at line 25 of file SClassAtom.h.

7.100.1.2 #define Tvector std::vector<T>

Definition at line 22 of file SClassAtom.h.

7.100.2 Function Documentation

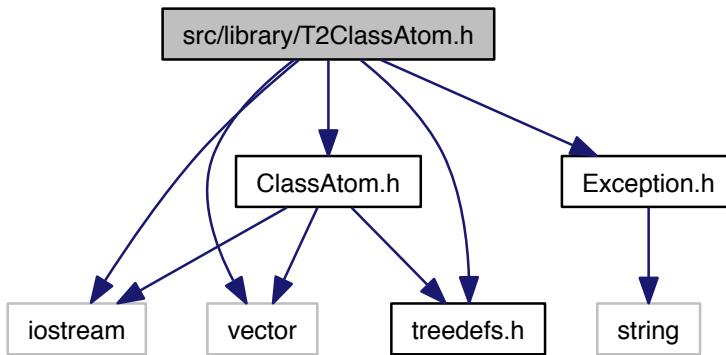
7.100.2.1 template<class T > std::ostream& operator<< (std::ostream & os, const SClassAtom< T > & sClassAtom)

Definition at line 127 of file SClassAtom.h.

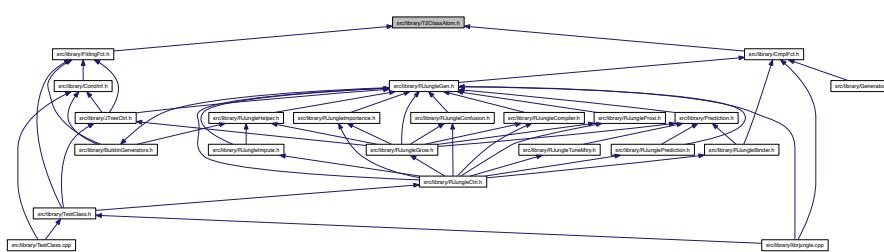
7.101 src/library/T2ClassAtom.cpp File Reference

7.102 src/library/T2ClassAtom.h File Reference

```
#include <iostream> #include <vector> #include "treedefs.h"
#include "ClassAtom.h" #include "Exception.h" Include dependency graph for T2ClassAtom.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `T2ClassAtom< T >`

Functions

- template<class T >
std::ostream & **operator<<** (std::ostream &os, const **T2ClassAtom**< T > &t2-
ClassAtom)

7.102.1 Function Documentation

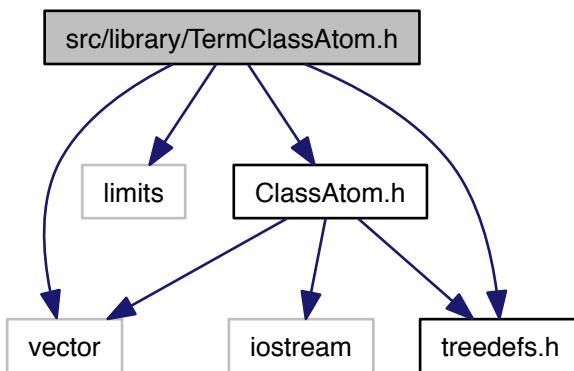
7.102.1.1 template<class T > std::ostream& operator<< (std::ostream & os, const T2ClassAtom< T > & t2ClassAtom)

Definition at line 105 of file T2ClassAtom.h.

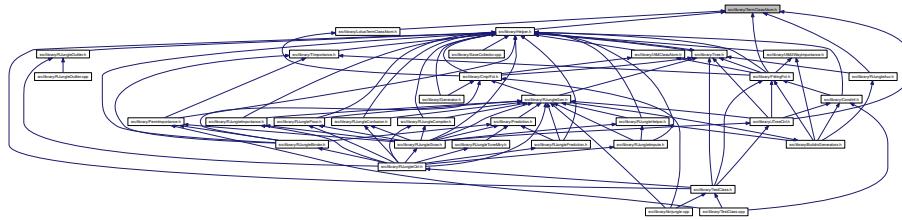
7.103 src/library/TermClassAtom.cpp File Reference

7.104 src/library/TermClassAtom.h File Reference

```
#include <vector> #include <limits> #include "ClassAtom.-  
h" #include "treedefs.h" Include dependency graph for TermClassAtom.h:
```



This graph shows which files directly or indirectly include this file:



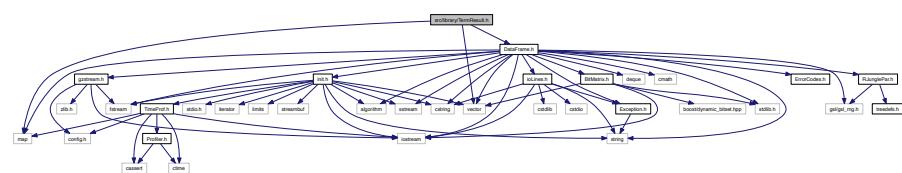
Classes

- class [TermClassAtom< T, C >](#)

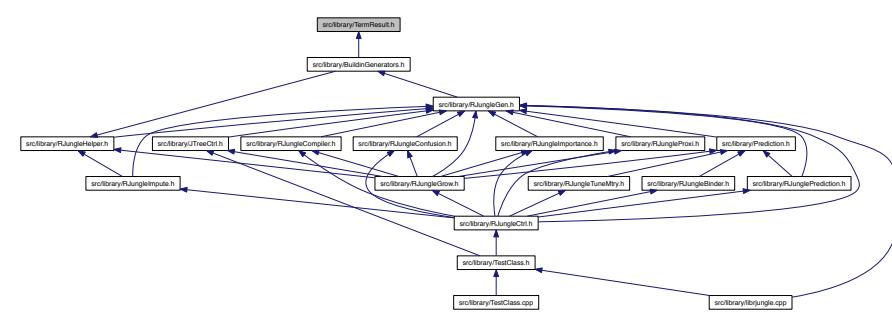
7.105 src/library/TermResult.cpp File Reference

7.106 src/library/TermResult.h File Reference

```
#include <vector> #include <map> #include "DataFrame.h" ×
Include dependency graph for TermResult.h:
```



This graph shows which files directly or indirectly include this file:



Namespaces

- namespace [TermResult](#)

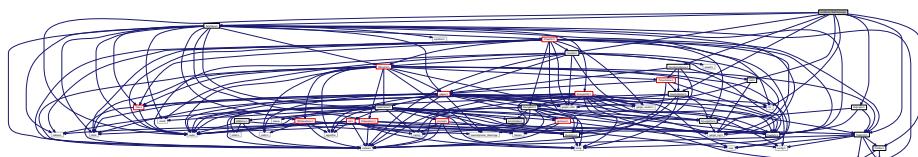
A set of functions which yield TermClassAtoms for terminal nodes.

Functions

- template<class T >
`TermClassAtom< T, uli_t > * TermResult::getTermMostFreq (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
Get most frequent value in vector.
- template<class T >
`TermClassAtom< T, uli_t > * TermResult::getTermMean (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
Get mean of vector.
- template<class T >
`TermClassAtom< T, uli_t > * TermResult::getTermLotus (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< uli_t > &colMaskVec, std::vector< double > &rowWeight)`
- template<class T >
`T TermResult::getMostFreq (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)`
Get most frequent value in vector.
- template<class T >
`static T TermResult::getMean (DataFrame< T > &data, RJunglePar &par, uli_t col, std::vector< uli_t > &rowMaskVec, std::vector< double > &rowWeight)`
Get mean of vector.

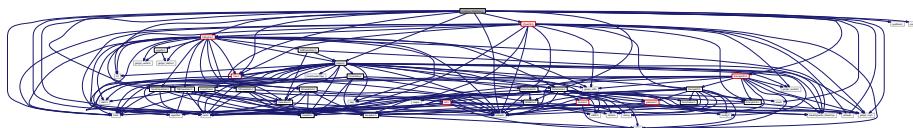
7.107 src/library/TestClass.cpp File Reference

```
#include <vector> #include "TestClass.h" #include "CondInf.h" #include "mvt.h" #include "SaveCollector.h" #include "PermImportance.h" #include "Profiler.h" #include "TimeProf.h" #include "librjungle.h" #include "lr.h" Include dependency graph for TestClass.cpp:
```

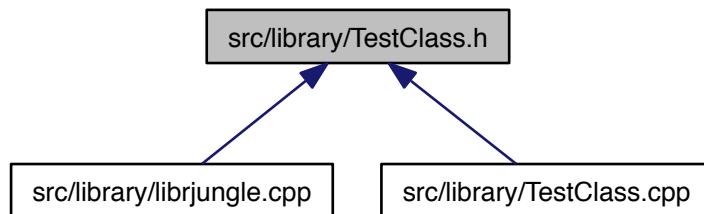


7.108 src/library/TestClass.h File Reference

```
#include <iostream> #include <sys/time.h> #include <limits> ×  
#include <cmath> #include <vector> #include <gsl/gsl-  
_rng.h> #include <cassert> #include <sstream> #include  
"treedefs.h" #include "DataFrame.h" #include "Helper.h" ×  
#include "FittingFct.h" #include "TMClassAtom.h" #include  
"SClassAtom.h" #include "Tree.h" #include "JTreeCtrl.h" ×  
#include "RJungleCtrl.h" #include "lr.h" Include dependency graph  
for TestClass.h:
```



This graph shows which files directly or indirectly include this file:

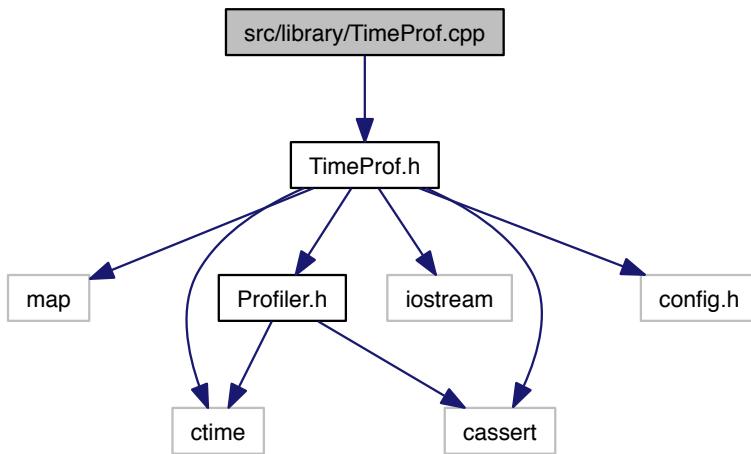


Classes

- class [TestClass](#)

7.109 src/library/TimeProf.cpp File Reference

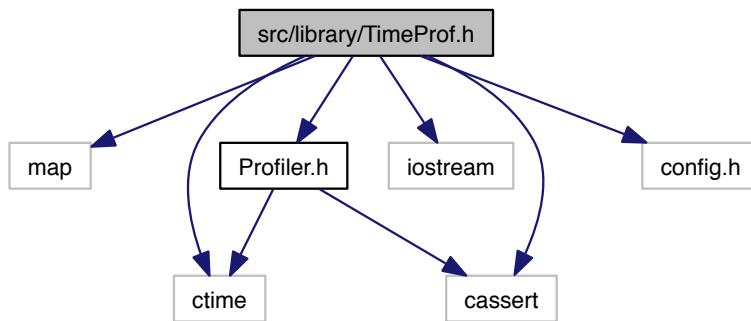
```
#include "TimeProf.h" Include dependency graph for TimeProf.cpp:
```



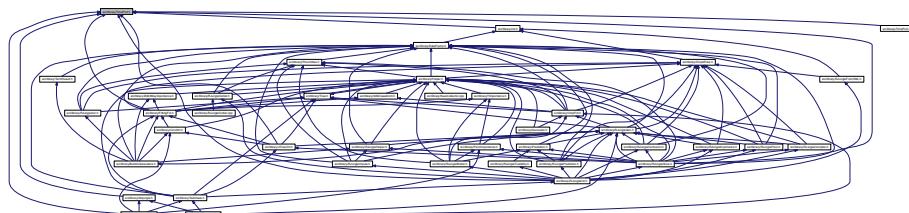
7.110 src/library/TimeProf.h File Reference

```
#include <map>    #include <ctime>    #include <cassert> x  
#include <iostream> #include "Profiler.h" #include "config.h"
```

h" Include dependency graph for TimeProf.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [TimeProf](#)

Defines

- `#define TIMEPROF_START(x) ;`
- `#define TIMEPROF_STOP(x) ;`

7.110.1 Define Documentation

7.110.1.1 `#define TIMEPROF_START(x) ;`

Definition at line 71 of file TimeProf.h.

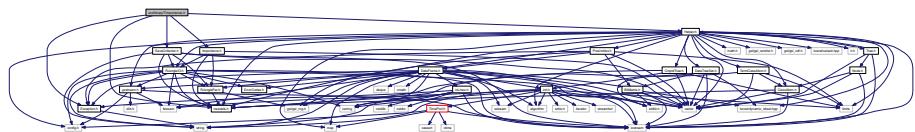
```
7.110.1.2 #define TIMEPROF_STOP( x );
```

Definition at line 72 of file TimeProf.h.

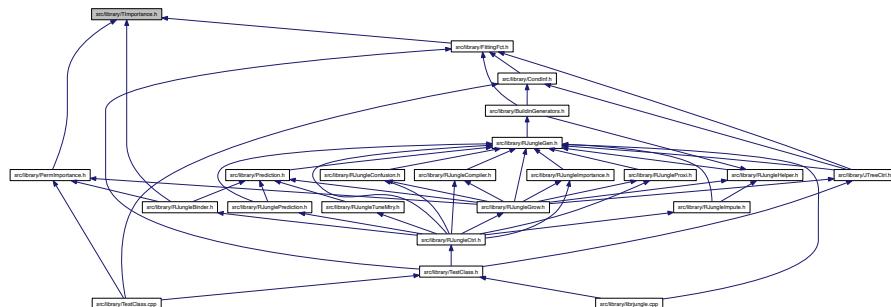
7.111 src/library/TImportance.cpp File Reference

7.112 src/library/TImportance.h File Reference

```
#include "treedefs.h" #include "Helper.h" #include "-Importance.h" #include "SaveCollector.h" #include "Exception.h" Include dependency graph for TImportance.h:
```



This graph shows which files directly or indirectly include this file:



Classes

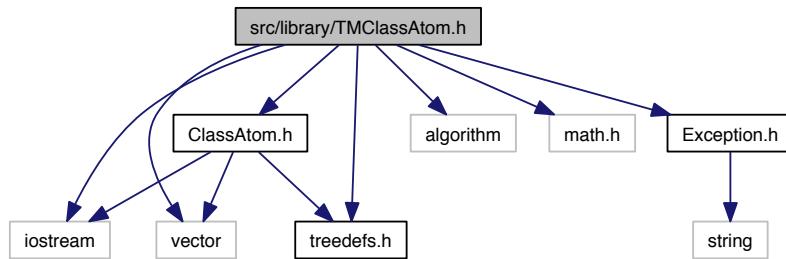
- class [TImportance< T >](#)

7.113 src/library/TMClassAtom.cpp File Reference

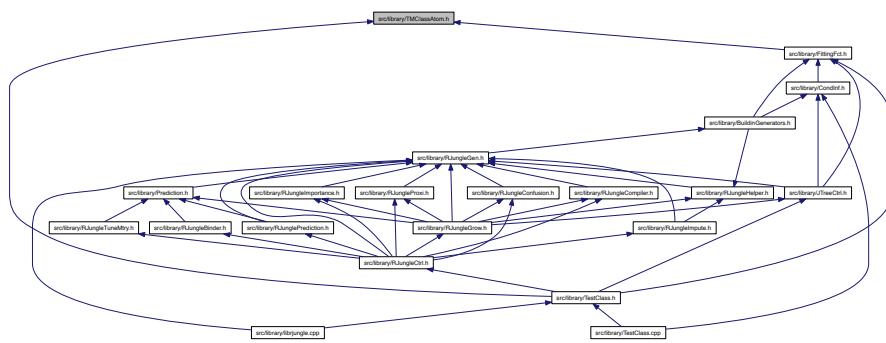
7.114 src/library/TMClassAtom.h File Reference

```
#include <iostream> #include <vector> #include <algorithm> x
#include <math.h> #include "treedefs.h" #include "Class-
```

Atom.h #include "Exception.h" Include dependency graph for TMClass-Atom.h:



This graph shows which files directly or indirectly include this file:



Classes

- class `TMClassAtom< T >`

Defines

- `#define Tvector std::vector<T >`
 - `#define CCvector std::vector<uli_t >`

Functions

- template<class T >
std::ostream & operator<< (std::ostream &os, const TMClassAtom< T > &tM-
ClassAtom)

7.114.1 Define Documentation

7.114.1.1 `#define CCvector std::vector<uli_t >`

Definition at line 27 of file TMClassAtom.h.

7.114.1.2 `#define Tvector std::vector<T >`

Definition at line 24 of file TMClassAtom.h.

7.114.2 Function Documentation

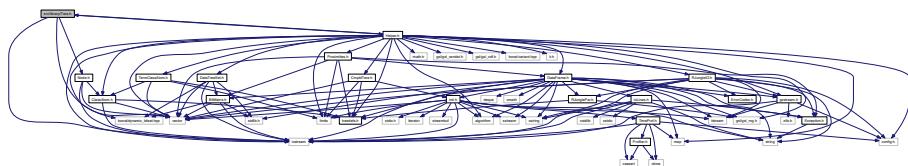
7.114.2.1 `template<class T > std::ostream& operator<< (std::ostream & os, const TMClassAtom< T > & tMCClassAtom)`

Definition at line 182 of file TMClassAtom.h.

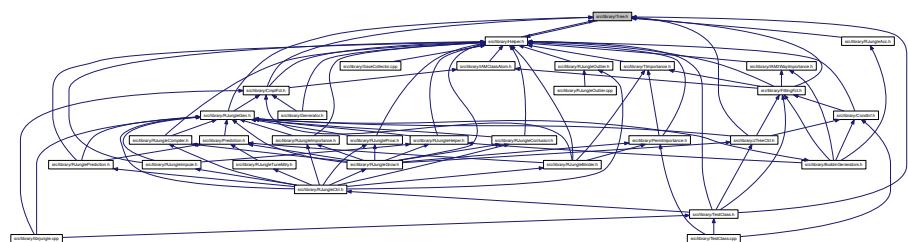
7.115 src/library/Tree.cpp File Reference

7.116 src/library/Tree.h File Reference

```
#include <iostream> #include <vector> #include "Helper.-  
h" #include "Node.h" Include dependency graph for Tree.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class `Tree< T, C >`

Functions

- template<class T , class C >
std::ostream & `operator<<` (std::ostream &os, `Tree< T, C > &tree)`

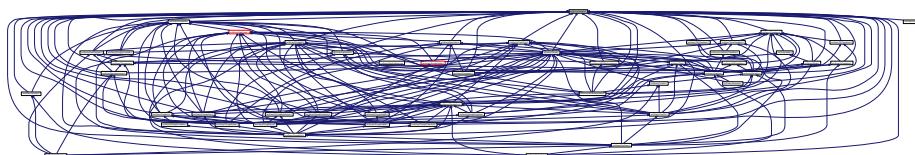
7.116.1 Function Documentation

7.116.1.1 template<class T , class C > std::ostream& operator<< (std::ostream & os,
`Tree< T, C > & tree)`

Definition at line 76 of file Tree.h.

7.117 src/library/treedefs.h File Reference

This graph shows which files directly or indirectly include this file:



Defines

- `#define MPI_TAG_NTREE 1001`
- `#define MPI_TAG_EXV 1003`
- `#define MPI_TAG_EX2B 1004`
- `#define MPI_TAG_EX2L 1005`
- `#define MPI_TAG_BITMATRIX_NROW 1006`
- `#define MPI_TAG_BITMATRIX_NCOL 1007`
- `#define MPI_TAG_BITMATRIX_DEP 1008`
- `#define MPI_TAG_DATATREESET_SLAVEID 1009`
- `#define MPI_TAG_PREDICTION_LEN 1010`
- `#define MPI_TAG_PREDICTION_PRED 1011`
- `#define MPI_TAG_PREDICTION_ACC 1012`
- `#define MYDATATYPE float`
- `#define MISSINGCODE -99`
- `#define SAVECOLLECTORTYPE`
- `#define LVLOFMEASUREMENT`

- #define CLASSATOMTYPE
- #define TREETYPE
- #define VARTYPE
- #define IMPORTANCEMEASURE
- #define BACKWARDELIMINATION
- #define VARPROXI
- #define GSL_RNG_SEED 15051980
- #define UZI unsigned short int
- #define TERMINALNODE 3
- #define __lrj
- #define __cdecl

TypeDefs

- typedef unsigned int uli_t

Enumerations

- enum SaveCollectorType { sc_double = 0, sc_int = 1, sc_size_t = 2, sc_uli_t = 3, sc_string = 4, sc_size = 5 }
- enum LvlOfMeasurement { lom_undef = 0, lom_nominal = 1, lom_ordinal = 2, lom_numeric = 3 }
- enum ClassAtomType { ca_BASE = 0, ca_TERM = 3, ca_T2 = 4, ca_TM = 5, ca_S = 6, ca_IAM = 7 }
- enum TreeType { tt_CART = 1, tt_CARTtwoing = 2, tt_CARTreg = 3, tt_CARTregTwoing = 4, tt_CARTsrt = 5, tt_CARTcor = 6, tt_CARTsse = 7, tt_CICase1 = 101, tt_CICase3 = 103, tt_CISNP = 111, tt_twoWayIntTree = 201, tt_LOTUS = 205, tt_UCART = 206, tt_SCART = 207, tt_trendCART = 208, tt_CARTregWithMiss = 209, tt_imputeTree = 210 }
- enum VarType { vt_CONT = 1, vt_CATE = 2 }
- enum ImportanceMeasure { im_intrinsic = 1, im_perm_breiman = 2, im_perm_liaw = 3, im_perm_raw = 4, im_perm_meng = 5 }
- enum BackwardElimination { bs_50P = 1, bs_33P = 2, bs_NEG = 3, bs_DIAZURIATE = 4 }
- enum VarProxi { vp_SIMPLE = 1, vp_EXTEND1 = 2 }

7.117.1 Define Documentation

7.117.1.1 #define __cdecl

Definition at line 158 of file treedefs.h.

7.117.1.2 #define __lrj

Definition at line 157 of file treedefs.h.

7.117.1.3 #define BACKWARDELIMINATION

Definition at line 98 of file treedefs.h.

7.117.1.4 #define CLASSATOMTYPE

Definition at line 44 of file treedefs.h.

7.117.1.5 #define GSL_RNG_SEED 15051980

Definition at line 118 of file treedefs.h.

7.117.1.6 #define IMPORTANCEMEASURE

Definition at line 89 of file treedefs.h.

7.117.1.7 #define LVLOFMEASUREMENT

Definition at line 36 of file treedefs.h.

7.117.1.8 #define MISSINGCODE -99

Definition at line 23 of file treedefs.h.

7.117.1.9 #define MPI_TAG_BITMATRIX_DEP 1008

Definition at line 10 of file treedefs.h.

7.117.1.10 #define MPI_TAG_BITMATRIX_NCOL 1007

Definition at line 9 of file treedefs.h.

7.117.1.11 #define MPI_TAG_BITMATRIX_NROW 1006

Definition at line 8 of file treedefs.h.

7.117.1.12 #define MPI_TAG_DATATREESET_SLAVEID 1009

Definition at line 11 of file treedefs.h.

7.117.1.13 #define MPI_TAG_EX2B 1004

Definition at line 6 of file treedefs.h.

7.117.1.14 #define MPI_TAG_EX2L 1005

Definition at line 7 of file treedefs.h.

7.117.1.15 #define MPI_TAG_EXV 1003

Definition at line 5 of file treedefs.h.

7.117.1.16 #define MPI_TAG_NTREE 1001

Definition at line 4 of file treedefs.h.

7.117.1.17 #define MPI_TAG_PREDICTION_ACC 1012

Definition at line 14 of file treedefs.h.

7.117.1.18 #define MPI_TAG_PREDICTION_LEN 1010

Definition at line 12 of file treedefs.h.

7.117.1.19 #define MPI_TAG_PREDICTION_PRED 1011

Definition at line 13 of file treedefs.h.

7.117.1.20 #define MYDATATYPE float

Definition at line 18 of file treedefs.h.

7.117.1.21 #define SAVECOLLECTORTYPE

Definition at line 26 of file treedefs.h.

7.117.1.22 #define TERMINALNODE 3

Definition at line 128 of file treedefs.h.

7.117.1.23 #define TREETYPE

Definition at line 54 of file treedefs.h.

7.117.1.24 #define UZI unsigned short int

Definition at line 119 of file treedefs.h.

7.117.1.25 #define VARPROXI

Definition at line 111 of file treedefs.h.

7.117.1.26 #define VARTYPE

Definition at line 82 of file treedefs.h.

7.117.2 Typedef Documentation**7.117.2.1 typedef unsigned int uli_t**

Definition at line 122 of file treedefs.h.

7.117.3 Enumeration Type Documentation**7.117.3.1 enum BackwardElimination**

Enumerator:

bs_50P

bs_33P

bs_NEG

bs_DIAZURIATE

Definition at line 99 of file treedefs.h.

7.117.3.2 enum ClassAtomType

Enumerator:

ca_BASE

ca_TERM

ca_T2

ca_TM

ca_S

ca_IAM

Definition at line 45 of file treedefs.h.

7.117.3.3 enum ImportanceMeasure

Enumerator:

im_intrinsic
im_perm_breiman
im_perm_liaw
im_perm_raw
im_perm_meng

Definition at line 90 of file treedefs.h.

7.117.3.4 enum LvlOfMeasurement

Enumerator:

lom_undef
lom_nominal
lom_ordinal
lom_numeric

Definition at line 37 of file treedefs.h.

7.117.3.5 enum SaveCollectorType

Enumerator:

sc_double
sc_int
sc_size_t
sc_uli_t
sc_string
sc_size

Definition at line 27 of file treedefs.h.

7.117.3.6 enum TreeType

Enumerator:

tt_CART
tt_CARTtwoing
tt_CARTreg
tt_CARTregTwoing
tt_CARTsrt
tt_CARTcor
tt_CARTsse
tt_CICase1
tt_CICase3
tt_CISNP
tt_twoWayIntTree
tt_LOTUS
tt_UCART
tt_SCART
tt_trendCART
tt_CARTregWithMiss
tt_imputeTree

Definition at line 55 of file treedefs.h.

7.117.3.7 enum VarProxi

Enumerator:

vp_SIMPLE
vp_EXTEND1

Definition at line 112 of file treedefs.h.

7.117.3.8 enum VarType

Enumerator:

vt_CONT
vt_CATE

Definition at line 83 of file treedefs.h.