# Evaporative Cooling

1.0

Generated by Doxygen 1.8.0

Tue Apr 24 2012 22:08:33

# 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** **9**

5.1 insilico Namespace Reference . . . 9

5.1.1 Function Documentation . . . 10

5.1.1.1 best_n . . . 10

5.1.1.2 get_bits . . . 10

5.1.1.3 join . . . 11

5.1.1.4 join . . . 11

5.1.1.5 join . . . 11

5.1.1.6 split . . . 11

5.1.1.7 split . . . 11

5.1.1.8 split . . . 11

5.1.1.9 split . . . 11

5.1.1.10 split . . . 11

5.1.1.11 split . . . 11

5.1.1.12 split . . . 11

5.1.1.13 split . . . 12

5.1.1.14 split . . . 12

5.1.1.15 split . . . 12

5.1.1.16 split_if . . . 12

5.1.1.17 split_if . . . 12

5.1.1.18 split_if . . . 12

5.1.1.19 to_lower . . . 12
5.1.1.20 to_lower . . . 12
5.1.1.21 to_lower . . . 12
5.1.1.22 to_upper . . . 12
5.1.1.23 to_upper . . . 13
5.1.1.24 to_upper . . . 13
5.1.1.25 trim . . . 13
5.1.1.26 trim . . . 13
5.1.1.27 trim . . . 13
5.1.1.28 trim_left . . . 13
5.1.1.29 trim_left . . . 13
5.1.1.30 trim_left . . . 13
5.1.1.31 trim_right . . . 13
5.1.1.32 trim_right . . . 13
5.1.1.33 trim_right . . . 13
5.1.1.34 zeroPadNumber . . . 14

**6 Class Documentation** **15**

6.1 ArffDataset Class Reference . . . 15
6.1.1 Detailed Description . . . 18
6.1.2 Constructor & Destructor Documentation . . . 18
6.1.2.1 ArffDataset . . . 18
6.1.2.2 ~ArffDataset . . . 18
6.1.3 Member Function Documentation . . . 19
6.1.3.1 GetTypeOf . . . 19
6.1.3.2 LoadSnps . . . 19
6.1.3.3 PrintNominalsMapping . . . 19
6.1.4 Member Data Documentation . . . 19
6.1.4.1 attributeTypes . . . 19
6.1.4.2 missingAttributeValuesToCheck . . . 19
6.1.4.3 missingClassValuesToCheck . . . 19
6.1.4.4 nominalValues . . . 19
6.1.4.5 relationName . . . 20
6.2 BirdseedData Class Reference . . . 20
6.2.1 Detailed Description . . . 22
6.2.2 Constructor & Destructor Documentation . . . 22
6.2.2.1 BirdseedData . . . 22
6.2.2.2 ~BirdseedData . . . 22
6.2.3 Member Function Documentation . . . 22
6.2.3.1 GetAlleleCounts . . . 22

6.2.3.2 GetGenotypeCounts . . . 22
6.2.3.3 GetMajorAlleleFrequency . . . 22
6.2.3.4 GetMajorMinorAlleles . . . 22
6.2.3.5 GetMissingValues . . . 23
6.2.3.6 GetNumSNPs . . . 23
6.2.3.7 GetNumSubjects . . . 23
6.2.3.8 GetSamplePhenotype . . . 23
6.2.3.9 GetSNPNames . . . 23
6.2.3.10 GetSubjectGenotypes . . . 23
6.2.3.11 GetSubjectLabels . . . 23
6.2.3.12 GetSubjectNames . . . 23
6.2.3.13 HasPhenotypes . . . 23
6.2.3.14 HasSubjectLabels . . . 24
6.2.3.15 LoadData . . . 24
6.2.3.16 PrintAlleleCounts . . . 24
6.2.3.17 PrintInfo . . . 24
6.2.4 Member Data Documentation . . . 24
6.2.4.1 excludeSnps . . . 24
6.2.4.2 excludeSnpsFilename . . . 25
6.2.4.3 genotypeCounts . . . 25
6.2.4.4 hasExcludedSnps . . . 25
6.2.4.5 hasIncludedSnps . . . 25
6.2.4.6 hasPhenotypes . . . 25
6.2.4.7 hasSubjectLabels . . . 25
6.2.4.8 includeSnps . . . 25
6.2.4.9 includeSnpsFilename . . . 25
6.2.4.10 missingValues . . . 25
6.2.4.11 phenosFilename . . . 25
6.2.4.12 phenotypes . . . 25
6.2.4.13 snpAlleleCounts . . . 26
6.2.4.14 snpGenotypes . . . 26
6.2.4.15 snpMajorAlleleFreq . . . 26
6.2.4.16 snpMajorMinorAlleles . . . 26
6.2.4.17 snpNames . . . 26
6.2.4.18 snpsFilename . . . 26
6.2.4.19 subjectLabels . . . 26
6.2.4.20 subjectLabelsFilename . . . 26
6.2.4.21 subjectNames . . . 26
6.3 ChiSquared Class Reference . . . 27
6.3.1 Detailed Description . . . 29

6.3.2 Constructor & Destructor Documentation . . . 29
6.3.2.1 ChiSquared . . . 29
6.3.2.2 ~ChiSquared . . . 29
6.3.3 Member Function Documentation . . . 29
6.3.3.1 ClearTables . . . 29
6.3.3.2 ComputeScore . . . 29
6.3.3.3 ComputeScores . . . 30
6.3.3.4 GetFrequencyCounts . . . 30
6.3.3.5 PrepareForAttribute . . . 30
6.3.3.6 PrintScores . . . 30
6.3.3.7 PrintTables . . . 30
6.3.3.8 WriteScores . . . 30
6.3.4 Member Data Documentation . . . 31
6.3.4.1 chiSquaredValues . . . 31
6.3.4.2 dataset . . . 31
6.3.4.3 expectedContingencyTable . . . 31
6.3.4.4 numClasses . . . 31
6.3.4.5 numLevels . . . 31
6.3.4.6 observedFreqTable . . . 31
6.3.4.7 scores . . . 31
6.4 Dataset Class Reference . . . 31
6.4.1 Detailed Description . . . 40
6.4.2 Constructor & Destructor Documentation . . . 40
6.4.2.1 Dataset . . . 40
6.4.2.2 ~Dataset . . . 41
6.4.3 Member Function Documentation . . . 41
6.4.3.1 AttributeInteractionInformation . . . 41
6.4.3.2 CalculateDistanceMatrix . . . 41
6.4.3.3 CalculateDistanceMatrix . . . 41
6.4.3.4 CalculateGainMatrix . . . 41
6.4.3.5 CalculateInteractionInformation . . . 42
6.4.3.6 CheckHardyWeinbergEquilibrium . . . 42
6.4.3.7 ComputeInstanceToInstanceDistance . . . 43
6.4.3.8 CreateDummyAlleles . . . 43
6.4.3.9 DetermineTreeType . . . 43
6.4.3.10 ExcludeMonomorphs . . . 43
6.4.3.11 ExtractAttributes . . . 43
6.4.3.12 GetAlternatePhenotypesFilename . . . 44
6.4.3.13 GetAttribute . . . 44
6.4.3.14 GetAttributeAlleles . . . 44

6.4.3.15 GetAttributeIndexFromName . . . 44
6.4.3.16 GetAttributeMAF . . . 45
6.4.3.17 GetAttributeMutationType . . . 45
6.4.3.18 GetAttributeNames . . . 45
6.4.3.19 GetAttributeRowCol . . . 45
6.4.3.20 GetAttributeTiTvCounts . . . 46
6.4.3.21 GetAttributeValues . . . 46
6.4.3.22 GetAttributeValues . . . 46
6.4.3.23 GetClassColumn . . . 46
6.4.3.24 GetClassIndexes . . . 46
6.4.3.25 GetClassProbability . . . 47
6.4.3.26 GetClassValues . . . 47
6.4.3.27 GetDistanceMetrics . . . 47
6.4.3.28 GetInstance . . . 47
6.4.3.29 GetInstanceIds . . . 48
6.4.3.30 GetInstanceIndexForID . . . 48
6.4.3.31 GetIntForGenotype . . . 48
6.4.3.32 GetJukesCantorDistance . . . 48
6.4.3.33 GetKimuraDistance . . . 48
6.4.3.34 GetMeanForNumeric . . . 48
6.4.3.35 GetMinMaxForContinuousPhenotype . . . 49
6.4.3.36 GetMinMaxForNumeric . . . 49
6.4.3.37 GetNumeric . . . 49
6.4.3.38 GetNumericIndexFromName . . . 49
6.4.3.39 GetNumericRowCol . . . 50
6.4.3.40 GetNumericsFilename . . . 50
6.4.3.41 GetNumericsNames . . . 50
6.4.3.42 GetNumericValues . . . 50
6.4.3.43 GetNumericValues . . . 51
6.4.3.44 GetProbabilityValueGivenClass . . . 51
6.4.3.45 GetRandomInstance . . . 51
6.4.3.46 GetSnpsFilename . . . 51
6.4.3.47 GetVariableNames . . . 52
6.4.3.48 HasAllelicInfo . . . 52
6.4.3.49 HasAlternatePhenotypes . . . 52
6.4.3.50 HasAlternatePhenotypes . . . 52
6.4.3.51 HasContinuousPhenotypes . . . 52
6.4.3.52 HasGenotypes . . . 52
6.4.3.53 HasNumerics . . . 52
6.4.3.54 HasNumerics . . . 52

6.4.3.55 HasPhenotypes . . . 52
6.4.3.56 IsLoadableInstanceID . . . 53
6.4.3.57 LoadAlternatePhenotypes . . . 53
6.4.3.58 LoadDataset . . . 53
6.4.3.59 LoadDataset . . . 53
6.4.3.60 LoadDataset . . . 54
6.4.3.61 LoadDataset . . . 54
6.4.3.62 LoadNumerics . . . 54
6.4.3.63 LoadSnps . . . 55
6.4.3.64 MaskGetAllVariableNames . . . 55
6.4.3.65 MaskGetAttributeIndices . . . 55
6.4.3.66 MaskGetAttributeMask . . . 55
6.4.3.67 MaskGetInstanceIds . . . 56
6.4.3.68 MaskGetInstanceIndices . . . 56
6.4.3.69 MaskGetInstanceMask . . . 56
6.4.3.70 MaskIncludeAllAttributes . . . 56
6.4.3.71 MaskIncludeAllInstances . . . 56
6.4.3.72 MaskPopAll . . . 57
6.4.3.73 MaskPushAll . . . 57
6.4.3.74 MaskRemoveInstance . . . 57
6.4.3.75 MaskRemoveVariable . . . 57
6.4.3.76 MaskRemoveVariableType . . . 57
6.4.3.77 MaskSearchInstance . . . 58
6.4.3.78 MaskSearchVariableType . . . 58
6.4.3.79 MaskWriteNewDataset . . . 58
6.4.3.80 NumAttributes . . . 58
6.4.3.81 NumClasses . . . 59
6.4.3.82 NumInstances . . . 59
6.4.3.83 NumLevels . . . 59
6.4.3.84 NumNumerics . . . 59
6.4.3.85 NumVariables . . . 59
6.4.3.86 Print . . . 59
6.4.3.87 PrintAttributeLevelsSeen . . . 59
6.4.3.88 PrintClassIndexInfo . . . 60
6.4.3.89 PrintLevelCounts . . . 60
6.4.3.90 PrintMaskStats . . . 60
6.4.3.91 PrintMissingValuesStats . . . 60
6.4.3.92 PrintNumericsStats . . . 60
6.4.3.93 PrintStats . . . 60
6.4.3.94 PrintStatsSimple . . . 60

6.4.3.95 ProcessExclusionFile . . . 60
6.4.3.96 RunSnpDiagnosticTests . . . 61
6.4.3.97 SetDistanceMetrics . . . 61
6.4.3.98 SNPHWE . . . 61
6.4.3.99 SwapAttributes . . . 61
6.4.3.100 UpdateAllLevelCounts . . . 62
6.4.3.101 UpdateLevelCounts . . . 62
6.4.3.102 WriteLevelCounts . . . 62
6.4.3.103 WriteNewDataset . . . 62
6.4.3.104 WriteNewDataset . . . 63
6.4.3.105 WriteNewPlinkPedDataset . . . 63
6.4.3.106 WriteSnpTiTvInfo . . . 63
6.4.4 Member Data Documentation . . . 63
6.4.4.1 alternatePhenotypesFilename . . . 63
6.4.4.2 attributeAlleleCounts . . . 63
6.4.4.3 attributeAlleles . . . 63
6.4.4.4 attributeLevelsSeen . . . 64
6.4.4.5 attributeMinorAllele . . . 64
6.4.4.6 attributeMutationMap . . . 64
6.4.4.7 attributeMutationTypes . . . 64
6.4.4.8 attributeNames . . . 64
6.4.4.9 attributesMask . . . 64
6.4.4.10 attributesMaskPushed . . . 64
6.4.4.11 classColumn . . . 64
6.4.4.12 classIndexes . . . 64
6.4.4.13 continuousPhenotypeMinMax . . . 65
6.4.4.14 genotypeCounts . . . 65
6.4.4.15 hasAllelicInfo . . . 65
6.4.4.16 hasAlternatePhenotypes . . . 65
6.4.4.17 hasContinuousPhenotypes . . . 65
6.4.4.18 hasGenotypes . . . 65
6.4.4.19 hasNumerics . . . 65
6.4.4.20 hasPhenotypes . . . 65
6.4.4.21 instanceIds . . . 65
6.4.4.22 instanceIdsToLoad . . . 66
6.4.4.23 instances . . . 66
6.4.4.24 instancesMask . . . 66
6.4.4.25 instancesMaskPushed . . . 66
6.4.4.26 levelCounts . . . 66
6.4.4.27 levelCountsByClass . . . 66

6.4.4.28 maskIsPushed . . . 66
6.4.4.29 missingNumericValues . . . 66
6.4.4.30 missingValues . . . 66
6.4.4.31 numDiff . . . 66
6.4.4.32 numericsFilename . . . 67
6.4.4.33 numericsIds . . . 67
6.4.4.34 numericsMask . . . 67
6.4.4.35 numericsMaskPushed . . . 67
6.4.4.36 numericsMinMax . . . 67
6.4.4.37 numericsNames . . . 67
6.4.4.38 numMetric . . . 67
6.4.4.39 phenotypesIds . . . 67
6.4.4.40 rng . . . 68
6.4.4.41 snpDiff . . . 68
6.4.4.42 snpMetric . . . 68
6.4.4.43 snpsFilename . . . 68
6.5 DatasetInstance Class Reference . . . 68
6.5.1 Detailed Description . . . 71
6.5.2 Constructor & Destructor Documentation . . . 71
6.5.2.1 DatasetInstance . . . 71
6.5.2.2 ∼DatasetInstance . . . 71
6.5.3 Member Function Documentation . . . 71
6.5.3.1 AddInfluenceFactorD . . . 72
6.5.3.2 AddNumeric . . . 72
6.5.3.3 ClearInfluenceFactors . . . 72
6.5.3.4 GetAttribute . . . 72
6.5.3.5 GetClass . . . 72
6.5.3.6 GetDatasetPtr . . . 72
6.5.3.7 GetInfluenceFactorD . . . 72
6.5.3.8 GetNNearestInstances . . . 73
6.5.3.9 GetNNearestInstances . . . 73
6.5.3.10 GetNNearestInstances . . . 73
6.5.3.11 GetNumeric . . . 73
6.5.3.12 GetPredictedValueTau . . . 74
6.5.3.13 LoadInstanceFromVector . . . 74
6.5.3.14 NumAttributes . . . 74
6.5.3.15 NumNumerics . . . 74
6.5.3.16 Print . . . 74
6.5.3.17 PrintDistancePairs . . . 74
6.5.3.18 SetClass . . . 75

6.5.3.19 SetDistanceSums . . . 75
6.5.3.20 SetDistanceSums . . . 75
6.5.3.21 SetPredictedValueTau . . . 75
6.5.3.22 SwapAttributes . . . 75
6.5.4 Member Data Documentation . . . 76
6.5.4.1 attributes . . . 76
6.5.4.2 bestNeighborIds . . . 76
6.5.4.3 bestNeighborIdsDiffClass . . . 76
6.5.4.4 bestNeighborIdsSameClass . . . 76
6.5.4.5 classLabel . . . 76
6.5.4.6 dataset . . . 76
6.5.4.7 neighborInfluenceFactorDs . . . 76
6.5.4.8 numerics . . . 76
6.5.4.9 predictedValueTau . . . 77
6.6 deref_less Class Reference . . . 77
6.6.1 Detailed Description . . . 77
6.6.2 Member Function Documentation . . . 77
6.6.2.1 operator() . . . 77
6.7 deref_less_bcw Class Reference . . . 77
6.7.1 Detailed Description . . . 77
6.7.2 Member Function Documentation . . . 77
6.7.2.1 operator() . . . 77
6.8 DgeData Class Reference . . . 78
6.8.1 Detailed Description . . . 79
6.8.2 Constructor & Destructor Documentation . . . 79
6.8.2.1 DgeData . . . 79
6.8.2.2 ∼DgeData . . . 79
6.8.3 Member Function Documentation . . . 79
6.8.3.1 GetGeneMinMax . . . 79
6.8.3.2 GetGeneNames . . . 79
6.8.3.3 GetNormalizationFactors . . . 79
6.8.3.4 GetNumGenes . . . 80
6.8.3.5 GetNumSamples . . . 80
6.8.3.6 GetSampleCounts . . . 80
6.8.3.7 GetSampleNames . . . 80
6.8.3.8 GetSamplePhenotype . . . 80
6.8.3.9 LoadData . . . 80
6.8.3.10 PrintSampleStats . . . 80
6.8.4 Member Data Documentation . . . 80
6.8.4.1 counts . . . 80

6.8.4.2 countsFilename . . . 81
6.8.4.3 geneNames . . . 81
6.8.4.4 hasNormFactors . . . 81
6.8.4.5 minMaxGeneCounts . . . 81
6.8.4.6 minMaxSampleCounts . . . 81
6.8.4.7 normFactors . . . 81
6.8.4.8 normsFilename . . . 81
6.8.4.9 phenosFilename . . . 81
6.8.4.10 phenotypes . . . 81
6.8.4.11 sampleNames . . . 82
6.8.4.12 sampleZeroes . . . 82
6.9 insilico::do_to_lower< charT > Class Template Reference . . . 82
6.9.1 Detailed Description . . . 82
6.9.2 Constructor & Destructor Documentation . . . 82
6.9.2.1 do_to_lower . . . 82
6.9.2.2 do_to_lower . . . 82
6.9.3 Member Function Documentation . . . 83
6.9.3.1 operator() . . . 83
6.9.4 Member Data Documentation . . . 83
6.9.4.1 m_ctype . . . 83
6.10 insilico::do_to_upper< charT > Class Template Reference . . . 83
6.10.1 Detailed Description . . . 83
6.10.2 Constructor & Destructor Documentation . . . 83
6.10.2.1 do_to_upper . . . 83
6.10.2.2 do_to_upper . . . 83
6.10.3 Member Function Documentation . . . 84
6.10.3.1 operator() . . . 84
6.10.4 Member Data Documentation . . . 84
6.10.4.1 m_ctype . . . 84
6.11 EvaporativeCooling Class Reference . . . 84
6.11.1 Detailed Description . . . 87
6.11.2 Constructor & Destructor Documentation . . . 88
6.11.2.1 EvaporativeCooling . . . 88
6.11.2.2 EvaporativeCooling . . . 88
6.11.2.3 ~EvaporativeCooling . . . 88
6.11.3 Member Function Documentation . . . 88
6.11.3.1 ComputeClassificationErrorRJ . . . 88
6.11.3.2 ComputeECScores . . . 88
6.11.3.3 ComputeFreeEnergy . . . 89
6.11.3.4 GetAlgorithmType . . . 89

6.11.3.5 GetECScores . . . 89
6.11.3.6 GetRandomJungleScores . . . 89
6.11.3.7 GetReliefFScores . . . 89
6.11.3.8 OptimizeTemperature . . . 89
6.11.3.9 PrintAllScoresTabular . . . 89
6.11.3.10 PrintAttributeScores . . . 89
6.11.3.11 PrintKendallTaus . . . 90
6.11.3.12 PrintRFAttributeScores . . . 90
6.11.3.13 PrintRJAttributeScores . . . 90
6.11.3.14 RemoveWorstAttributes . . . 90
6.11.3.15 RunReliefF . . . 90
6.11.3.16 WriteAttributeScores . . . 90
6.11.4 Member Data Documentation . . . 91
6.11.4.1 algorithmType . . . 91
6.11.4.2 analysisType . . . 91
6.11.4.3 bestClassificationError . . . 91
6.11.4.4 dataset . . . 91
6.11.4.5 ecScores . . . 91
6.11.4.6 evaporatedAttributes . . . 91
6.11.4.7 freeEnergyScores . . . 91
6.11.4.8 numRFThreads . . . 91
6.11.4.9 numTargetAttributes . . . 92
6.11.4.10 numToRemoveNextIteration . . . 92
6.11.4.11 numToRemovePerIteration . . . 92
6.11.4.12 optimalTemperature . . . 92
6.11.4.13 optimizeTemperature . . . 92
6.11.4.14 outFilesPrefix . . . 92
6.11.4.15 paramsMap . . . 92
6.11.4.16 randomJungle . . . 92
6.11.4.17 reliefF . . . 92
6.11.4.18 rfScores . . . 92
6.11.4.19 rjScores . . . 93
6.12 GSLRandomBase Class Reference . . . 93
6.12.1 Detailed Description . . . 94
6.12.2 Constructor & Destructor Documentation . . . 94
6.12.2.1 GSLRandomBase . . . 94
6.12.2.2 GSLRandomBase . . . 94
6.12.2.3 ∼GSLRandomBase . . . 94
6.12.3 Member Function Documentation . . . 94
6.12.3.1 nextRandVal . . . 94

6.12.3.2 state . . . 94
6.12.4 Member Data Documentation . . . 95
6.12.4.1 rStatePtr_ . . . 95
6.13 GSLRandomFlat Class Reference . . . 95
6.13.1 Detailed Description . . . 96
6.13.2 Constructor & Destructor Documentation . . . 97
6.13.2.1 GSLRandomFlat . . . 97
6.13.2.2 ∼GSLRandomFlat . . . 97
6.13.3 Member Function Documentation . . . 97
6.13.3.1 nextRandVal . . . 97
6.13.4 Member Data Documentation . . . 97
6.13.4.1 lower_ . . . 97
6.13.4.2 upper_ . . . 97
6.14 insilico::is_classified< Type, charT > Class Template Reference . . . 97
6.14.1 Detailed Description . . . 97
6.14.2 Constructor & Destructor Documentation . . . 98
6.14.2.1 is_classified . . . 98
6.14.2.2 is_classified . . . 98
6.14.3 Member Function Documentation . . . 98
6.14.3.1 operator() . . . 98
6.14.4 Member Data Documentation . . . 98
6.14.4.1 m_ctype . . . 98
6.15 PlinkBinaryDataset Class Reference . . . 98
6.15.1 Detailed Description . . . 101
6.15.2 Constructor & Destructor Documentation . . . 101
6.15.2.1 PlinkBinaryDataset . . . 101
6.15.2.2 ∼PlinkBinaryDataset . . . 102
6.15.3 Member Function Documentation . . . 102
6.15.3.1 GetAttributeMAF . . . 102
6.15.3.2 GetAttributeMutationType . . . 102
6.15.3.3 LoadSnps . . . 102
6.15.3.4 ReadBimFile . . . 103
6.15.3.5 ReadFamFile . . . 103
6.15.4 Member Data Documentation . . . 103
6.15.4.1 filenameBase . . . 103
6.15.4.2 instanceIndicesToKeep . . . 104
6.15.4.3 missingAttributeValuesToCheck . . . 104
6.15.4.4 missingClassValuesToCheck . . . 104
6.15.4.5 missingPhenoLines . . . 104
6.15.4.6 numAttributesRead . . . 104

6.15.4.7 numClassesRead . . . 104
6.15.4.8 numInstancesRead . . . 104
6.15.4.9 validAttributeValues . . . 104
6.16 PlinkDataset Class Reference . . . 104
6.16.1 Detailed Description . . . 107
6.16.2 Constructor & Destructor Documentation . . . 107
6.16.2.1 PlinkDataset . . . 107
6.16.2.2 ∼PlinkDataset . . . 107
6.16.3 Member Function Documentation . . . 107
6.16.3.1 GetAttributeMAF . . . 107
6.16.3.2 GetAttributeMutationType . . . 108
6.16.3.3 LoadSnps . . . 108
6.16.4 Member Data Documentation . . . 109
6.16.4.1 filenameBase . . . 109
6.16.4.2 missingClassValuesToCheck . . . 109
6.17 PlinkRawDataset Class Reference . . . 109
6.17.1 Detailed Description . . . 112
6.17.2 Constructor & Destructor Documentation . . . 112
6.17.2.1 PlinkRawDataset . . . 112
6.17.2.2 ∼PlinkRawDataset . . . 112
6.17.3 Member Function Documentation . . . 112
6.17.3.1 LoadSnps . . . 112
6.18 RandomJungle Class Reference . . . 113
6.18.1 Detailed Description . . . 115
6.18.2 Constructor & Destructor Documentation . . . 116
6.18.2.1 RandomJungle . . . 116
6.18.2.2 RandomJungle . . . 116
6.18.2.3 ∼RandomJungle . . . 116
6.18.3 Member Function Documentation . . . 116
6.18.3.1 ComputeAttributeScores . . . 116
6.18.3.2 ComputeAttributeScoresRjungle . . . 116
6.18.3.3 GetClassificationError . . . 117
6.18.3.4 GetScores . . . 117
6.18.3.5 ReadClassificationError . . . 117
6.18.3.6 ReadClassificationError . . . 117
6.18.3.7 ReadScores . . . 117
6.18.3.8 RunClassifier . . . 117
6.18.4 Member Data Documentation . . . 118
6.18.4.1 classificationError . . . 118
6.18.4.2 dataset . . . 118

6.18.4.3 rjParams . . . 118
6.18.4.4 runMode . . . 118
6.18.4.5 scores . . . 118
6.19 ReliefF Class Reference . . . 118
6.19.1 Detailed Description . . . 122
6.19.2 Constructor & Destructor Documentation . . . 122
6.19.2.1 ReliefF . . . 122
6.19.2.2 ReliefF . . . 122
6.19.2.3 ReliefF . . . 123
6.19.2.4 ∼ReliefF . . . 123
6.19.3 Member Function Documentation . . . 123
6.19.3.1 ComputeAttributeScores . . . 123
6.19.3.2 ComputeAttributeScoresIteratively . . . 123
6.19.3.3 ComputeWeightByDistanceFactors . . . 123
6.19.3.4 GetScores . . . 124
6.19.3.5 PreComputeDistances . . . 124
6.19.3.6 PreComputeDistancesByMap . . . 124
6.19.3.7 PrintAttributeScores . . . 124
6.19.3.8 ResetForNextIteration . . . 124
6.19.3.9 WriteAttributeScores . . . 124
6.19.4 Member Data Documentation . . . 124
6.19.4.1 analysisType . . . 124
6.19.4.2 dataset . . . 125
6.19.4.3 doRemovePercent . . . 125
6.19.4.4 finalScores . . . 125
6.19.4.5 k . . . 125
6.19.4.6 m . . . 125
6.19.4.7 numDiff . . . 125
6.19.4.8 numMetric . . . 125
6.19.4.9 one_over_m_times_k . . . 125
6.19.4.10 randomlySelect . . . 126
6.19.4.11 removePercentage . . . 126
6.19.4.12 removePerIteration . . . 126
6.19.4.13 scoreNames . . . 126
6.19.4.14 snpDiff . . . 126
6.19.4.15 snpMetric . . . 126
6.19.4.16 W . . . 126
6.19.4.17 weightByDistanceMethod . . . 127
6.19.4.18 weightByDistanceSigma . . . 127
6.20 RReliefF Class Reference . . . 127

6.20.1 Detailed Description . . . 130
6.20.2 Constructor & Destructor Documentation . . . 130
6.20.2.1 RReliefF . . . 130
6.20.2.2 RReliefF . . . 130
6.20.2.3 RReliefF . . . 130
6.20.2.4 ∼RReliefF . . . 131
6.20.3 Member Function Documentation . . . 131
6.20.3.1 ComputeAttributeScores . . . 131

**7 File Documentation** **133**

7.1 src/library/ArffDataset.cpp File Reference . . . 133
7.2 src/library/ArffDataset.h File Reference . . . 133
7.2.1 Enumeration Type Documentation . . . 134
7.2.1.1 ArffAttributeType . . . 134
7.3 src/library/BestN.h File Reference . . . 135
7.3.1 Detailed Description . . . 136
7.4 src/library/BirdseedData.cpp File Reference . . . 136
7.5 src/library/BirdseedData.h File Reference . . . 136
7.6 src/library/ChiSquared.cpp File Reference . . . 137
7.7 src/library/ChiSquared.h File Reference . . . 138
7.8 src/library/Dataset.cpp File Reference . . . 139
7.9 src/library/Dataset.h File Reference . . . 139
7.10 src/library/DatasetInstance.cpp File Reference . . . 140
7.10.1 Typedef Documentation . . . 141
7.10.1.1 T . . . 141
7.11 src/library/DatasetInstance.h File Reference . . . 141
7.12 src/library/DgeData.cpp File Reference . . . 141
7.13 src/library/DgeData.h File Reference . . . 142
7.14 src/library/DistanceMetrics.cpp File Reference . . . 142
7.14.1 Function Documentation . . . 143
7.14.1.1 CheckMissing . . . 143
7.14.1.2 CheckMissingNumeric . . . 144
7.14.1.3 diffAMM . . . 144
7.14.1.4 diffGMM . . . 144
7.14.1.5 diffKM . . . 145
7.14.1.6 diffManhattan . . . 145
7.14.1.7 diffNCA . . . 145
7.14.1.8 diffPredictedValueTau . . . 145
7.14.1.9 norm . . . 146
7.15 src/library/DistanceMetrics.h File Reference . . . 146

7.15.1 Detailed Description . . . 147
7.15.2 Function Documentation . . . 147
7.15.2.1 CheckMissing . . . 147
7.15.2.2 CheckMissingNumeric . . . 147
7.15.2.3 diffAMM . . . 148
7.15.2.4 diffGMM . . . 148
7.15.2.5 diffKM . . . 148
7.15.2.6 diffManhattan . . . 149
7.15.2.7 diffNCA . . . 149
7.15.2.8 diffPredictedValueTau . . . 149
7.15.2.9 norm . . . 149
7.16 src/library/EvaporativeCooling.cpp File Reference . . . 150
7.16.1 Function Documentation . . . 150
7.16.1.1 scoresSortAsc . . . 150
7.16.1.2 scoresSortAscByName . . . 150
7.16.1.3 scoresSortDesc . . . 151
7.17 src/library/EvaporativeCooling.h File Reference . . . 151
7.17.1 Typedef Documentation . . . 152
7.17.1.1 EcScores . . . 152
7.17.1.2 EcScoresCIt . . . 152
7.17.1.3 EcScoresIt . . . 152
7.17.2 Enumeration Type Documentation . . . 152
7.17.2.1 EcAlgorithmType . . . 152
7.17.3 Function Documentation . . . 153
7.17.3.1 libec_is_present . . . 153
7.18 src/library/GSLRandomBase.h File Reference . . . 153
7.19 src/library/GSLRandomFlat.h File Reference . . . 153
7.20 src/library/Insilico.cpp File Reference . . . 154
7.20.1 Function Documentation . . . 155
7.20.1.1 ChooseSnpsDatasetByExtension . . . 155
7.20.1.2 DetectClassType . . . 155
7.20.1.3 DetermineRandomJungleTreeType . . . 156
7.20.1.4 GetConfigValue . . . 156
7.20.1.5 GetFileBasename . . . 156
7.20.1.6 GetFileExtension . . . 156
7.20.1.7 GetMatchingIds . . . 156
7.20.1.8 LoadNumericIds . . . 156
7.20.1.9 LoadPhenoIds . . . 157
7.20.1.10 ProtectedLog . . . 157
7.20.1.11 Timestamp . . . 157

7.21 src/library/Insilico.h File Reference . . . 157
7.21.1 Detailed Description . . . 160
7.21.2 Typedef Documentation . . . 160
7.21.2.1 AttributeLevel . . . 160
7.21.2.2 ClassLevel . . . 160
7.21.2.3 ConfigMap . . . 160
7.21.2.4 DistancePair . . . 160
7.21.2.5 DistancePairs . . . 160
7.21.2.6 DistancePairsIt . . . 161
7.21.2.7 NumericLevel . . . 161
7.21.3 Enumeration Type Documentation . . . 161
7.21.3.1 AnalysisType . . . 161
7.21.3.2 AttributeMutationType . . . 161
7.21.3.3 AttributeType . . . 161
7.21.3.4 ClassType . . . 162
7.21.3.5 OutputDatasetType . . . 162
7.21.3.6 RandomJungleRunMode . . . 162
7.21.3.7 RandomJungleTreeType . . . 162
7.21.3.8 ValueType . . . 163
7.21.4 Function Documentation . . . 163
7.21.4.1 ChooseSnpsDatasetByExtension . . . 163
7.21.4.2 DetectClassType . . . 163
7.21.4.3 DetermineRandomJungleTreeType . . . 163
7.21.4.4 GetConfigValue . . . 164
7.21.4.5 GetFileBasename . . . 164
7.21.4.6 GetFileExtension . . . 164
7.21.4.7 GetMatchingIds . . . 164
7.21.4.8 LoadNumericIds . . . 165
7.21.4.9 LoadPhenoIds . . . 165
7.21.4.10 PrintVector . . . 165
7.21.4.11 ProtectedLog . . . 166
7.21.4.12 Timestamp . . . 166
7.21.5 Variable Documentation . . . 166
7.21.5.1 COMMAND_LINE_ERROR . . . 166
7.21.5.2 DATASET_LOAD_ERROR . . . 166
7.21.5.3 INVALID_ATTRIBUTE_VALUE . . . 166
7.21.5.4 INVALID_DISCRETE_CLASS_VALUE . . . 166
7.21.5.5 INVALID_DISTANCE . . . 166
7.21.5.6 INVALID_INDEX . . . 166
7.21.5.7 INVALID_INT_VALUE . . . 167

7.21.5.8 INVALID_NUMERIC_CLASS_VALUE . . . 167
7.21.5.9 INVALID_NUMERIC_VALUE . . . 167
7.21.5.10 MISSING_ATTRIBUTE_VALUE . . . 167
7.21.5.11 MISSING_DISCRETE_CLASS_VALUE . . . 167
7.21.5.12 MISSING_NUMERIC_CLASS_VALUE . . . 167
7.21.5.13 MISSING_NUMERIC_VALUE . . . 167
7.22 src/library/PlinkBinaryDataset.cpp File Reference . . . 167
7.23 src/library/PlinkBinaryDataset.h File Reference . . . 168
7.24 src/library/PlinkDataset.cpp File Reference . . . 169
7.25 src/library/PlinkDataset.h File Reference . . . 169
7.25.1 Enumeration Type Documentation . . . 170
7.25.1.1 MapFileType . . . 170
7.26 src/library/PlinkRawDataset.cpp File Reference . . . 170
7.27 src/library/PlinkRawDataset.h File Reference . . . 171
7.28 src/library/RandomJungle.cpp File Reference . . . 172
7.29 src/library/RandomJungle.h File Reference . . . 172
7.30 src/library/ReliefF.cpp File Reference . . . 173
7.30.1 Typedef Documentation . . . 174
7.30.1.1 AttributeIndex . . . 174
7.30.1.2 AttributeIndexIt . . . 174
7.30.1.3 ScoresMap . . . 174
7.30.1.4 ScoresMapIt . . . 174
7.30.1.5 T . . . 175
7.30.2 Function Documentation . . . 175
7.30.2.1 attributeSort . . . 175
7.30.2.2 librelieff_is_present . . . 175
7.30.2.3 scoreSort . . . 175
7.31 src/library/ReliefF.h File Reference . . . 175
7.32 src/library/RReliefF.cpp File Reference . . . 176
7.33 src/library/RReliefF.h File Reference . . . 176
7.34 src/library/Statistics.cpp File Reference . . . 177
7.34.1 Define Documentation . . . 178
7.34.1.1 DEBUG_E . . . 178
7.34.1.2 DEBUG_Z . . . 178
7.34.2 Function Documentation . . . 178
7.34.2.1 condentropy . . . 178
7.34.2.2 ConditionalEntropy . . . 179
7.34.2.3 ConstructAttributeCart . . . 179
7.34.2.4 Entropy . . . 179
7.34.2.5 KendallTau . . . 179

7.34.2.6 KendallTau . . . 179
7.34.2.7 KendallTau . . . 179
7.34.2.8 PrintHistogram . . . 179
7.34.2.9 SelfEntropy . . . 179
7.34.2.10 ZTransform . . . 179
7.35 src/library/Statistics.h File Reference . . . 180
7.35.1 Typedef Documentation . . . 181
7.35.1.1 Histogram . . . 181
7.35.1.2 HistogramIt . . . 181
7.35.1.3 VectorDouble . . . 181
7.35.1.4 VectorDoubleIt . . . 181
7.35.2 Function Documentation . . . 182
7.35.2.1 condentropy . . . 182
7.35.2.2 ConditionalEntropy . . . 182
7.35.2.3 ConstructAttributeCart . . . 182
7.35.2.4 Entropy . . . 182
7.35.2.5 KendallTau . . . 182
7.35.2.6 KendallTau . . . 183
7.35.2.7 KendallTau . . . 183
7.35.2.8 PrintHistogram . . . 183
7.35.2.9 SelfEntropy . . . 183
7.35.2.10 VarStd . . . 184
7.35.2.11 ZTransform . . . 184
7.36 src/library/StringUtils.h File Reference . . . 184
7.36.1 Detailed Description . . . 186

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

insilico . . . . . 9

# Chapter 2

# Class Index

## 2.1 Class Hierarchy

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


BirdseedData . . . 20
ChiSquared . . . 27
Dataset . . . 31
  ArffDataset . . . 15
  PlinkBinaryDataset . . . 98
  PlinkDataset . . . 104
  PlinkRawDataset . . . 109
DatasetInstance . . . 68
deref_less . . . 77
deref_less_bcw . . . 77
DgeData . . . 78
insilico::do_to_lower< charT > . . . 82
insilico::do_to_upper< charT > . . . 83
EvaporativeCooling . . . 84
GSLRandomBase . . . 93
  GSLRandomFlat . . . 95
insilico::is_classified< Type, charT > . . . 97
RandomJungle . . . 113
ReliefF . . . 118
  RReliefF . . . 127

# Chapter 3

# Class Index

## 3.1 Class List

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


ArffDataset
: ARFF file format reader . . . 15

BirdseedData
: Read Broad's Birdsuite Birdseed-called SNP data . . . 20

ChiSquared
: Chi-squared attribute ranking algorithm . . . 27

Dataset
: Base class for collections of instances containing attributea and class . . . 31

DatasetInstance
: Class to hold dataset instances (rows of attributes) . . . 68

deref_less . . . 77

deref_less_bcw . . . 77

DgeData
: Digital gene expression data . . . 78

insilico::do_to_lower< charT > . . . 82

insilico::do_to_upper< charT > . . . 83

EvaporativeCooling
: Evaporative Cooling attribute ranking algorithm . . . 84

GSLRandomBase
: A base class for GNU Scientific Library (GSL) random number functions . . . 93

GSLRandomFlat
: Random numbers in a flat, or uniform distribution . . . 95

insilico::is_classified< Type, charT > . . . 97

PlinkBinaryDataset
: Plink binary PED/BED file format reader . . . 98

PlinkDataset
: Plink MAP/PED file format reader . . . 104

PlinkRawDataset
: Plink recodeA/RAW file format reader . . . 109

RandomJungle
: RandomJungle attribute ranking algorithm . . . 113

ReliefF
: ReliefF attribute ranking algorithm . . . 118

RReliefF
: Regression ReliefF attribute ranking algorithm . . . 127

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all files with brief descriptions:


src/library/ArffDataset.cpp . . . 133
src/library/ArffDataset.h . . . 133
src/library/BestN.h
Find the best n keeping original order for ties - stable sort . . . 135
src/library/BirdseedData.cpp . . . 136
src/library/BirdseedData.h . . . 136
src/library/ChiSquared.cpp . . . 137
src/library/ChiSquared.h . . . 138
src/library/Dataset.cpp . . . 139
src/library/Dataset.h . . . 139
src/library/DatasetInstance.cpp . . . 140
src/library/DatasetInstance.h . . . 141
src/library/DgeData.cpp . . . 141
src/library/DgeData.h . . . 142
src/library/DistanceMetrics.cpp . . . 142
src/library/DistanceMetrics.h
Distance metrics for ReliefF . . . 146
src/library/EvaporativeCooling.cpp . . . 150
src/library/EvaporativeCooling.h . . . 151
src/library/GSLRandomBase.h . . . 153
src/library/GSLRandomFlat.h . . . 153
src/library/Insilico.cpp . . . 154
src/library/Insilico.h
Common functions for Insilico Lab projects . . . 157
src/library/PlinkBinaryDataset.cpp . . . 167
src/library/PlinkBinaryDataset.h . . . 168
src/library/PlinkDataset.cpp . . . 169
src/library/PlinkDataset.h . . . 169
src/library/PlinkRawDataset.cpp . . . 170
src/library/PlinkRawDataset.h . . . 171
src/library/RandomJungle.cpp . . . 172
src/library/RandomJungle.h . . . 172
src/library/ReliefF.cpp . . . 173
src/library/ReliefF.h . . . 175
src/library/RReliefF.cpp . . . 176
src/library/RReliefF.h . . . 176
src/library/Statistics.cpp . . . 177
src/library/Statistics.h . . . 180

src/library/StringUtils.h
Various string-related utilities . . . . . 184

# Chapter 5

# Namespace Documentation

## 5.1 insilico Namespace Reference

### Classes

- class is_classified
- class do_to_upper
- class do_to_lower

### Functions

- template<typename InputIt , typename OutputIt , typename Comp >
  void best_n (InputIt begin, InputIt end, OutputIt out, size_t n, Comp comp)

  *Get the best n values with ties keeping same original order.*
- template<typename stringT >
  stringT trim_left (const stringT &s, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT trim_right (const stringT &s, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT trim (const stringT &s, const std::locale &loc=std::locale())
- template<typename Container , typename stringT >
  void split (Container &cont, const stringT &s, const std::locale &loc=std::locale())
- template<typename Container , typename stringT >
  void split (Container &cont, const stringT &s, const stringT &delim)
- template<typename Container , typename stringT , typename Pred >
  void split_if (Container &cont, const stringT &s, const Pred &pred)
- template<typename It , typename stringT >
  stringT join (const It &begin, const It &end, const stringT &delim)
- template<typename stringT >
  stringT to_upper (const stringT &str, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT to_lower (const stringT &str, const std::locale &loc=std::locale())
- std::string trim_left (const char *s, const std::locale &loc=std::locale())
- std::wstring trim_left (const wchar_t *s, const std::locale &loc=std::locale())
- std::string trim_right (const char *s, const std::locale &loc=std::locale())
- std::wstring trim_right (const wchar_t *s, const std::locale &loc=std::locale())
- std::string trim (const char *s, const std::locale &loc=std::locale())
- std::wstring trim (const wchar_t *s, const std::locale &loc=std::locale())
- template<typename Container >
  void split (Container &cont, const char *s, const std::locale &loc=std::locale())

- template<typename Container >
  void split (Container &cont, const wchar_t *s, const std::locale &loc=std::locale())
- template<typename Container >
  void split (Container &cont, const std::string &s, const char *delim)
- template<typename Container >
  void split (Container &cont, const char *s, const std::string &delim)
- template<typename Container >
  void split (Container &cont, const char *s, const char *delim)
- template<typename Container >
  void split (Container &cont, const std::wstring &s, const wchar_t *delim)
- template<typename Container >
  void split (Container &cont, const wchar_t *s, const std::wstring &delim)
- template<typename Container >
  void split (Container &cont, const wchar_t *s, const wchar_t *delim)
- template<typename Container , typename Pred >
  void split_if (Container &cont, const char *s, const Pred &pred)
- template<typename Container , typename Pred >
  void split_if (Container &cont, const wchar_t *s, const Pred &pred)
- template<typename It >
  std::string join (const It &begin, const It &end, const char *delim)
- template<typename It >
  std::wstring join (const It &begin, const It &end, const wchar_t *delim)
- std::string to_upper (const char *s, const std::locale &loc=std::locale())
- std::wstring to_upper (const wchar_t *s, const std::locale &loc=std::locale())
- std::string to_lower (const char *s, const std::locale &loc=std::locale())
- std::wstring to_lower (const wchar_t *s, const std::locale &loc=std::locale())
- template<typename T >
  std::string get_bits (T value)
- template<typename T >
  std::string zeroPadNumber (T num, int padSize)

### 5.1.1 Function Documentation

#### 5.1.1.1 template<typename InputIt , typename OutputIt , typename Comp > void insilico::best_n ( InputIt *begin*, InputIt *end*, OutputIt *out*, size_t *n*, Comp *comp* )

Get the best n values with ties keeping same original order.

**Parameters**

| | | |
|---|---|---|
| in | *begin* | iterator of the beginning of a input container |
| in | *end* | iterator of the end of a input container |
| out | *out* | iterator of the beginning of a output container |
| in | *size* | best n value |
| in | *comp* | compare functor |

**Returns**

path/filename without extension

Definition at line 30 of file BestN.h.

#### 5.1.1.2 template<typename T > std::string insilico::get_bits ( T *value* )

Definition at line 324 of file StringUtils.h.

#### 5.1.1.3 template<typename It , typename stringT > stringT insilico::join ( const It & *begin,* const It & *end,* const stringT & *delim* )

Definition at line 198 of file StringUtils.h.

#### 5.1.1.4 template<typename It > std::string insilico::join ( const It & *begin,* const It & *end,* const char * *delim* ) [inline]

Definition at line 300 of file StringUtils.h.

#### 5.1.1.5 template<typename It > std::wstring insilico::join ( const It & *begin,* const It & *end,* const wchar_t * *delim* ) [inline]

Definition at line 304 of file StringUtils.h.

#### 5.1.1.6 template<typename Container , typename stringT > void insilico::split ( Container & *cont,* const stringT & *s,* const std::locale & *loc* = `std::locale()` ) [inline]

Definition at line 148 of file StringUtils.h.

#### 5.1.1.7 template<typename Container , typename stringT > void insilico::split ( Container & *cont,* const stringT & *s,* const stringT & *delim* )

Definition at line 156 of file StringUtils.h.

#### 5.1.1.8 template<typename Container > void insilico::split ( Container & *cont,* const char * *s,* const std::locale & *loc* = `std::locale()` ) [inline]

Definition at line 258 of file StringUtils.h.

#### 5.1.1.9 template<typename Container > void insilico::split ( Container & *cont,* const wchar_t * *s,* const std::locale & *loc* = `std::locale()` ) [inline]

Definition at line 263 of file StringUtils.h.

#### 5.1.1.10 template<typename Container > void insilico::split ( Container & *cont,* const std::string & *s,* const char * *delim* ) [inline]

Definition at line 268 of file StringUtils.h.

#### 5.1.1.11 template<typename Container > void insilico::split ( Container & *cont,* const char * *s,* const std::string & *delim* ) [inline]

Definition at line 272 of file StringUtils.h.

#### 5.1.1.12 template<typename Container > void insilico::split ( Container & *cont,* const char * *s,* const char * *delim* ) [inline]

Definition at line 276 of file StringUtils.h.

5.1.1.13 template<typename Container > void **insilico::split** ( Container & *cont,* const std::wstring & *s,* const wchar_t * *delim* ) [inline]

Definition at line 280 of file StringUtils.h.

5.1.1.14 template<typename Container > void **insilico::split** ( Container & *cont,* const wchar_t * *s,* const std::wstring & *delim* ) [inline]

Definition at line 284 of file StringUtils.h.

5.1.1.15 template<typename Container > void **insilico::split** ( Container & *cont,* const wchar_t * *s,* const wchar_t * *delim* ) [inline]

Definition at line 288 of file StringUtils.h.

5.1.1.16 template<typename Container , typename stringT , typename Pred > void **insilico::split_if** ( Container & *cont,* const stringT & *s,* const Pred & *pred* )

Definition at line 178 of file StringUtils.h.

5.1.1.17 template<typename Container , typename Pred > void **insilico::split_if** ( Container & *cont,* const char * *s,* const Pred & *pred* ) [inline]

Definition at line 292 of file StringUtils.h.

5.1.1.18 template<typename Container , typename Pred > void **insilico::split_if** ( Container & *cont,* const wchar_t * *s,* const Pred & *pred* ) [inline]

Definition at line 296 of file StringUtils.h.

5.1.1.19 template<typename stringT > stringT **insilico::to_lower** ( const stringT & *str,* const std::locale & *loc =* std::locale() )

Definition at line 224 of file StringUtils.h.

5.1.1.20 std::string **insilico::to_lower** ( const char * *s,* const std::locale & *loc =* std::locale() ) [inline]

Definition at line 315 of file StringUtils.h.

5.1.1.21 std::wstring **insilico::to_lower** ( const wchar_t * *s,* const std::locale & *loc =* std::locale() ) [inline]

Definition at line 319 of file StringUtils.h.

5.1.1.22 template<typename stringT > stringT **insilico::to_upper** ( const stringT & *str,* const std::locale & *loc =* std::locale() )

Definition at line 214 of file StringUtils.h.

5.1.1.23 std::string insilico::to_upper ( const char * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 307 of file StringUtils.h.

5.1.1.24 std::wstring insilico::to_upper ( const wchar_t * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 311 of file StringUtils.h.

5.1.1.25 template<typename stringT > stringT insilico::trim ( const stringT & *s*, const std::locale & *loc* = std::locale() )

Definition at line 123 of file StringUtils.h.

5.1.1.26 std::string insilico::trim ( const char * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 249 of file StringUtils.h.

5.1.1.27 std::wstring insilico::trim ( const wchar_t * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 253 of file StringUtils.h.

5.1.1.28 template<typename stringT > stringT insilico::trim_left ( const stringT & *s*, const std::locale & *loc* = std::locale() )

Definition at line 101 of file StringUtils.h.

5.1.1.29 std::string insilico::trim_left ( const char * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 233 of file StringUtils.h.

5.1.1.30 std::wstring insilico::trim_left ( const wchar_t * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 237 of file StringUtils.h.

5.1.1.31 template<typename stringT > stringT insilico::trim_right ( const stringT & *s*, const std::locale & *loc* = std::locale() )

Definition at line 112 of file StringUtils.h.

5.1.1.32 std::string insilico::trim_right ( const char * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 241 of file StringUtils.h.

5.1.1.33 std::wstring insilico::trim_right ( const wchar_t * *s*, const std::locale & *loc* = std::locale() ) [inline]

Definition at line 245 of file StringUtils.h.

#### 5.1.1.34 template<typename T > std::string insilico::zeroPadNumber ( T *num,* int *padSize* )

Definition at line 333 of file StringUtils.h.

# Chapter 6

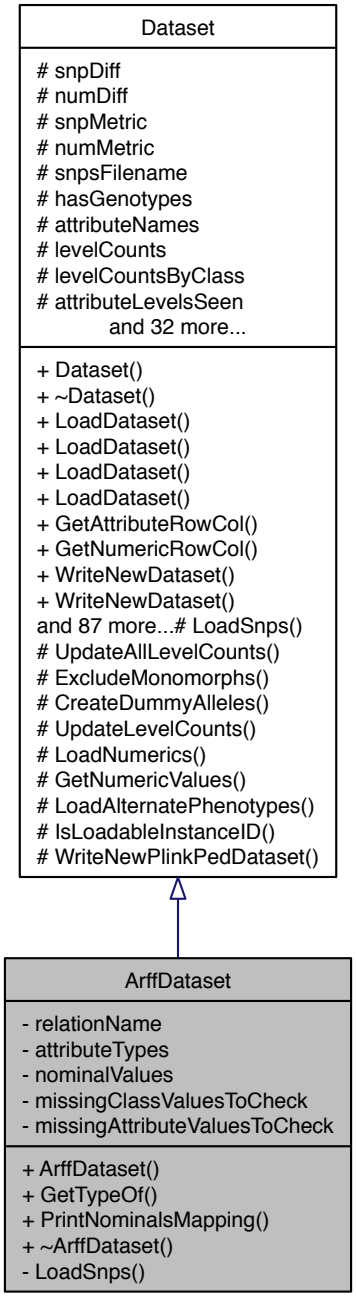
# Class Documentation

## 6.1 ArffDataset Class Reference

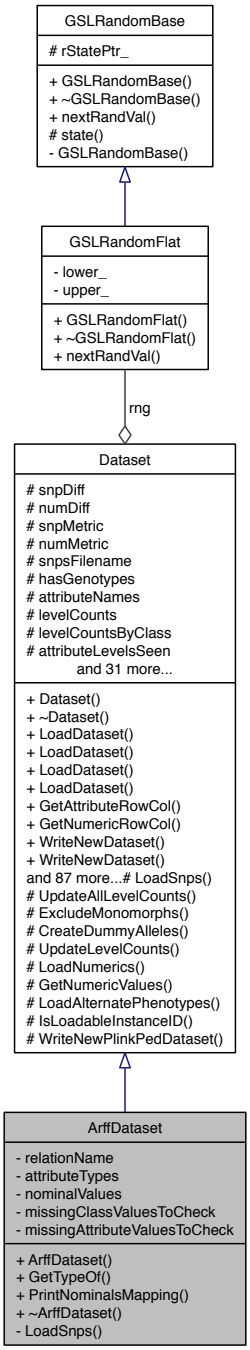
ARFF file format reader.

`#include <ArffDataset.h>`

Inheritance diagram for ArffDataset:



Collaboration diagram for ArffDataset:




## Public Member Functions

- ArffDataset ()
- ArffAttributeType GetTypeOf (unsigned int columnIndex)
- void PrintNominalsMapping ()
- ~ArffDataset ()

### Private Member Functions

- bool LoadSnps (std::string filename)

  *Load SNPs from file using the data set filename.*

### Private Attributes

- std::string relationName

  *ARFF relation name.*
- std::vector< ArffAttributeType > attributeTypes

  *vector of attribute types*
- std::map< std::string, std::vector< std::string > > nominalValues

  *map of attribute names to valid nominal values*
- std::vector< std::string > missingClassValuesToCheck

  *missing class values*
- std::vector< std::string > missingAttributeValuesToCheck

  *missing attribute values*

## 6.1.1 Detailed Description

ARFF file format reader.

http://www.cs.waikato.ac.nz/ml/weka/arff.html

**See also**

Dataset

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 2/24/11

Definition at line 38 of file ArffDataset.h.

## 6.1.2 Constructor & Destructor Documentation

### 6.1.2.1 ArffDataset::ArffDataset ( )

Definition at line 25 of file ArffDataset.cpp.

### 6.1.2.2 ArffDataset::~ArffDataset ( ) [inline]

Definition at line 54 of file ArffDataset.h.

### 6.1.3 Member Function Documentation

#### 6.1.3.1 ArffAttributeType ArffDataset::GetTypeOf ( unsigned int *columnIndex* )

Definition at line 375 of file ArffDataset.cpp.

#### 6.1.3.2 bool ArffDataset::LoadSnps ( std::string *filename* ) [private, virtual]

Load SNPs from file using the data set filename.

---------- Beginning of private methods ----------

**Parameters**

| | | |
|---|---|---|
| in | *filename* | SNPs filename |
| in | *deRecodeA* | perform a recodeA operation after reading raw data? |

**Returns**

success

Open the data file and read line-by-line

Detect the class type

Reimplemented from Dataset.

Definition at line 31 of file ArffDataset.cpp.

#### 6.1.3.3 void ArffDataset::PrintNominalsMapping ( )

Definition at line 382 of file ArffDataset.cpp.

### 6.1.4 Member Data Documentation

#### 6.1.4.1 std::vector<ArffAttributeType> ArffDataset::attributeTypes [private]

vector of attribute types

Definition at line 61 of file ArffDataset.h.

#### 6.1.4.2 std::vector<std::string> ArffDataset::missingAttributeValuesToCheck [private]

missing attribute values

Definition at line 68 of file ArffDataset.h.

#### 6.1.4.3 std::vector<std::string> ArffDataset::missingClassValuesToCheck [private]

missing class values

Definition at line 66 of file ArffDataset.h.

#### 6.1.4.4 std::map<std::string, std::vector<std::string> > ArffDataset::nominalValues [private]

map of attribute names to valid nominal values

Definition at line 63 of file ArffDataset.h.

#### 6.1.4.5 std::string ArffDataset::relationName [private]

ARFF relation name.

Definition at line 59 of file ArffDataset.h.

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

- src/library/ArffDataset.h
- src/library/ArffDataset.cpp

## 6.2 BirdseedData Class Reference

Read Broad's Birdsuite Birdseed-called SNP data.

`#include <BirdseedData.h>`

### Public Member Functions

- BirdseedData ()
- virtual ~BirdseedData ()
- bool LoadData (std::string snpsFile, std::string phenoFile="", std::string subjsFile="", std::string includeSnpsFile="", std::string excludeSnpsFile="")

  *Create a new set of Birdseed data with a SNPs file and optional phenotype file and optional subject names file.*
- std::vector< std::string > GetSubjectNames ()

  *Get the subject names/IDs.*
- std::vector< std::string > GetSubjectLabels ()

  *Get the subject labels.*
- bool HasSubjectLabels ()

  *Do the subjects have labels?*
- std::vector< std::string > GetSNPNames ()

  *Get the SNP names/IDs.*
- int GetNumSubjects ()

  *Get the number of subjects.*
- int GetNumSNPs ()

  *Get the number of SNPs.*
- std::vector< int > GetSubjectGenotypes (int subjectIndex)

  *Get SNPs for sample at index.*
- int GetSamplePhenotype (int subjectIndex)

  *Get the phenotype at sample index.*
- void PrintInfo ()

  *Print basic statistics to the console.*
- bool HasPhenotypes ()

  *Does this data have phenotypes?*
- std::pair< char, char > GetMajorMinorAlleles (int snpIndex)

  *Get the major and minor alleles for a SNP.*
- double GetMajorAlleleFrequency (int snpIndex)

  *Get the major allele frequency for a SNP.*
- std::map< char, unsigned int > GetAlleleCounts (int snpIndex)

  *Get the allele counts for a SNP.*
- std::map< std::string,
  unsigned int > GetGenotypeCounts (int snpIndex)

*Get the original string genotype counts for a SNP.*

- bool GetMissingValues (std::string subjectName, std::vector< unsigned int > &missingValueIndices)

  *get the missing value indices for the subject name*

- void PrintAlleleCounts ()

  *Print the allele counts for each SNP to the console.*

## Private Attributes

- std::string snpsFilename

  *Filename containing birdseed-called SNPs.*

- std::string subjectLabelsFilename

  *Filename containing subject names.*

- std::vector< std::string > subjectLabels
- bool hasSubjectLabels
- std::vector< std::string > subjectNames
- std::string excludeSnpsFilename
- std::vector< std::string > excludeSnps
- bool hasExcludedSnps
- std::string includeSnpsFilename
- std::vector< std::string > includeSnps
- bool hasIncludedSnps
- std::vector< std::string > snpNames

  *SNP names.*

- std::vector< std::vector< int > > snpGenotypes

  *SNP genotypes.*

- std::vector< std::map < std::string, unsigned int > > genotypeCounts

  *SNP genotype->count.*

- std::vector< std::pair< char, char > > snpMajorMinorAlleles

  *SNP genotypes alleles.*

- std::vector< double > snpMajorAlleleFreq

  *SNP genotypes major allele frequency.*

- std::vector< std::map< char, unsigned int > > snpAlleleCounts

  *SNP allele->count.*

- std::map< std::string, std::vector< unsigned int > > missingValues

  *subject name -> attribute indices*

- std::string phenosFilename

  *Sample phenotypes Filename containing subject phenotypes.*

- std::vector< int > phenotypes

  *vector of phenotypes (case-control)*

- bool hasPhenotypes

  *has phenotypes?*

### 6.2.1 Detailed Description

Read Broad's Birdsuite Birdseed-called SNP data.

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 2/12/12

Definition at line 20 of file BirdseedData.h.

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 BirdseedData::BirdseedData ( )

Definition at line 24 of file BirdseedData.cpp.

#### 6.2.2.2 BirdseedData::~BirdseedData ( ) [virtual]

Definition at line 36 of file BirdseedData.cpp.

### 6.2.3 Member Function Documentation

#### 6.2.3.1 map< char, unsigned int > BirdseedData::GetAlleleCounts ( int *snpIndex* )

Get the allele counts for a SNP.

Definition at line 554 of file BirdseedData.cpp.

#### 6.2.3.2 map< string, unsigned int > BirdseedData::GetGenotypeCounts ( int *snpIndex* )

Get the original string genotype counts for a SNP.

Definition at line 565 of file BirdseedData.cpp.

#### 6.2.3.3 double BirdseedData::GetMajorAlleleFrequency ( int *snpIndex* )

Get the major allele frequency for a SNP.

Definition at line 543 of file BirdseedData.cpp.

#### 6.2.3.4 pair< char, char > BirdseedData::GetMajorMinorAlleles ( int *snpIndex* )

Get the major and minor alleles for a SNP.

Definition at line 532 of file BirdseedData.cpp.

#### 6.2.3.5 bool BirdseedData::GetMissingValues ( std::string *subjectName,* std::vector< unsigned int > & *missingValueIndices* )

get the missing value indices for the subject name

Definition at line 576 of file BirdseedData.cpp.

#### 6.2.3.6 int BirdseedData::GetNumSNPs ( )

Get the number of SNPs.

Definition at line 481 of file BirdseedData.cpp.

#### 6.2.3.7 int BirdseedData::GetNumSubjects ( )

Get the number of subjects.

Definition at line 477 of file BirdseedData.cpp.

#### 6.2.3.8 int BirdseedData::GetSamplePhenotype ( int *subjectIndex* )

Get the phenotype at sample index.

Definition at line 500 of file BirdseedData.cpp.

#### 6.2.3.9 vector< string > BirdseedData::GetSNPNames ( )

Get the SNP names/IDs.

Definition at line 473 of file BirdseedData.cpp.

#### 6.2.3.10 vector< int > BirdseedData::GetSubjectGenotypes ( int *subjectIndex* )

Get SNPs for sample at index.

Definition at line 485 of file BirdseedData.cpp.

#### 6.2.3.11 vector< string > BirdseedData::GetSubjectLabels ( )

Get the subject labels.

Definition at line 465 of file BirdseedData.cpp.

#### 6.2.3.12 vector< string > BirdseedData::GetSubjectNames ( )

Get the subject names/IDs.

Definition at line 461 of file BirdseedData.cpp.

#### 6.2.3.13 bool BirdseedData::HasPhenotypes ( )

Does this data have phenotypes?

Definition at line 528 of file BirdseedData.cpp.

#### 6.2.3.14 bool BirdseedData::HasSubjectLabels ( )

Do the subjects have labels?

Definition at line 469 of file BirdseedData.cpp.

#### 6.2.3.15 bool BirdseedData::LoadData ( std::string *snpsFile,* std::string *phenoFile =* " ", std::string *subjsFile =* " ", std::string *includeSnpsFile =* " ", std::string *excludeSnpsFile =* " " )

Create a new set of Birdseed data with a SNPs file and optional phenotype file and optional subject names file.

read subjects file if specified

read SNP exclusion file

read SNP inclusion file

read SNPs data from the Birdsuite Birdseed SNP call file

skip any header comment lines

assumption: past any comment rows and at the the header row 7 fields per subject

read SNP genotypes across all SNPs and all subjects

split the line into genotypes

first field is the Affymetrix SNP ID

check for the inclusion/exclusion of this snpID

found - skip this SNP

not found - skip this SNP

skip missing genotypes for allele updates

missing data detected

save the genotypic/allelic distribution for this SNP

for each SNP, map two-allele genotypes to integers using allele frequencies

map genotype string vector to genotype int vector

read phenotypes

Definition at line 39 of file BirdseedData.cpp.

#### 6.2.3.16 void BirdseedData::PrintAlleleCounts ( )

Print the allele counts for each SNP to the console.

Definition at line 584 of file BirdseedData.cpp.

#### 6.2.3.17 void BirdseedData::PrintInfo ( )

Print basic statistics to the console.

Definition at line 510 of file BirdseedData.cpp.

### 6.2.4 Member Data Documentation

#### 6.2.4.1 std::vector<std::string> BirdseedData::excludeSnps [private]

Definition at line 73 of file BirdseedData.h.

#### 6.2.4.2 std::string BirdseedData::excludeSnpsFilename [private]

Definition at line 72 of file BirdseedData.h.

#### 6.2.4.3 std::vector<std::map<std::string, unsigned int> > BirdseedData::genotypeCounts [private]

SNP genotype->count.

Definition at line 83 of file BirdseedData.h.

#### 6.2.4.4 bool BirdseedData::hasExcludedSnps [private]

Definition at line 74 of file BirdseedData.h.

#### 6.2.4.5 bool BirdseedData::hasIncludedSnps [private]

Definition at line 77 of file BirdseedData.h.

#### 6.2.4.6 bool BirdseedData::hasPhenotypes [private]

has phenotypes?

Definition at line 101 of file BirdseedData.h.

#### 6.2.4.7 bool BirdseedData::hasSubjectLabels [private]

Definition at line 69 of file BirdseedData.h.

#### 6.2.4.8 std::vector<std::string> BirdseedData::includeSnps [private]

Definition at line 76 of file BirdseedData.h.

#### 6.2.4.9 std::string BirdseedData::includeSnpsFilename [private]

Definition at line 75 of file BirdseedData.h.

#### 6.2.4.10 std::map<std::string, std::vector<unsigned int> > BirdseedData::missingValues [private]

subject name -> attribute indices

Definition at line 93 of file BirdseedData.h.

#### 6.2.4.11 std::string BirdseedData::phenosFilename [private]

Sample phenotypes Filename containing subject phenotypes.

Definition at line 97 of file BirdseedData.h.

#### 6.2.4.12 std::vector<int> BirdseedData::phenotypes [private]

vector of phenotypes (case-control)

Definition at line 99 of file BirdseedData.h.

#### 6.2.4.13 std::vector<std::map<char, unsigned int> > BirdseedData::snpAlleleCounts [private]

SNP allele->count.

Definition at line 90 of file BirdseedData.h.

#### 6.2.4.14 std::vector<std::vector<int> > BirdseedData::snpGenotypes [private]

SNP genotypes.

Definition at line 81 of file BirdseedData.h.

#### 6.2.4.15 std::vector<double> BirdseedData::snpMajorAlleleFreq [private]

SNP genotypes major allele frequency.

Definition at line 88 of file BirdseedData.h.

#### 6.2.4.16 std::vector<std::pair<char, char> > BirdseedData::snpMajorMinorAlleles [private]

SNP genotypes alleles.

Definition at line 86 of file BirdseedData.h.

#### 6.2.4.17 std::vector<std::string> BirdseedData::snpNames [private]

SNP names.

Definition at line 79 of file BirdseedData.h.

#### 6.2.4.18 std::string BirdseedData::snpsFilename [private]

Filename containing birdseed-called SNPs.

Definition at line 64 of file BirdseedData.h.

#### 6.2.4.19 std::vector<std::string> BirdseedData::subjectLabels [private]

Definition at line 68 of file BirdseedData.h.

#### 6.2.4.20 std::string BirdseedData::subjectLabelsFilename [private]

Filename containing subject names.

Definition at line 67 of file BirdseedData.h.

#### 6.2.4.21 std::vector<std::string> BirdseedData::subjectNames [private]

Definition at line 70 of file BirdseedData.h.

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

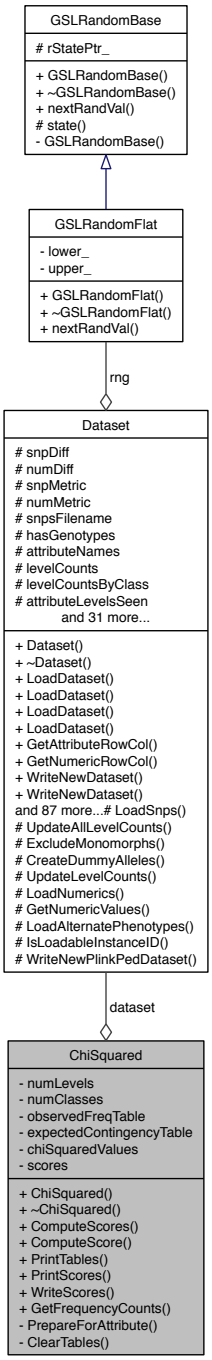
- src/library/BirdseedData.h
- src/library/BirdseedData.cpp

## 6.3 ChiSquared Class Reference

Chi-squared attribute ranking algorithm.

`#include <ChiSquared.h>`

Collaboration diagram for ChiSquared:



## Public Member Functions

- ChiSquared (Dataset *ds)

  *Construct an chi-squared algorithm object.*
- ~ChiSquared ()
- const std::vector< std::pair < double, double > > & ComputeScores ()

  *For each attribue, calculate chi-squared and associated p-value.*
- std::pair< double, double > ComputeScore (unsigned int index)

  *For the attribue at the specified index, calculate the chi-squared and associated p-value.*
- void PrintTables ()

  *Print calculation tables.*
- void PrintScores (std::ofstream &outStream, unsigned int topN=0)

  *Print the scores to a stream.*
- void WriteScores (std::string outFilename, unsigned int topN=0)

  *Print the scores to a stream.*
- std::vector< std::vector < double > > GetFrequencyCounts ()

  *Get the observed frequencies table as a vector of vector of doubles.*

## Private Member Functions

- void PrepareForAttribute (unsigned int attributeIndex)

  *Private method to setup the chi-squared contingency tables for a particular attribute.*
- void ClearTables ()

  *Clear calculation tables.*

## Private Attributes

- Dataset * dataset

  *pointer to a Dataset object*
- unsigned int numLevels

  *number of levels in the attributes*
- unsigned int numClasses

  *number of classes in the instances*
- std::vector< std::vector < double > > observedFreqTable

  *observed frequencies*
- std::vector< std::vector < double > > expectedContingencyTable
- std::vector< std::vector < double > > chiSquaredValues

  *chi squared computed values*
- std::vector< std::pair< double, double > > scores

  *chi-squared value, p-value for each attribute*

### 6.3.1 Detailed Description

Chi-squared attribute ranking algorithm.

ChiSquared algorithm interface. For performing chi-squared tests of association between an attribute and its class across all instances in a data set.

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 6/15/05

Definition at line 25 of file ChiSquared.h.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 ChiSquared::ChiSquared ( Dataset * *ds* )

Construct an chi-squared algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |

Definition at line 21 of file ChiSquared.cpp.

#### 6.3.2.2 ChiSquared::~ChiSquared ( )

Definition at line 31 of file ChiSquared.cpp.

### 6.3.3 Member Function Documentation

#### 6.3.3.1 void ChiSquared::ClearTables ( ) [private]

Clear calculation tables.

Definition at line 230 of file ChiSquared.cpp.

#### 6.3.3.2 pair< double, double > ChiSquared::ComputeScore ( unsigned int *index* )

For the attribue at the specified index, calculate the chi-squared and associated p-value.

Return as a pair.

**Parameters**

| | | |
|---|---|---|
| in | *index* | index into the attributes of the data set |

**Returns**

pairs of chi-squared score and associated p-value for the attribute

Definition at line 46 of file ChiSquared.cpp.

#### 6.3.3.3 const vector< pair< double, double > > & ChiSquared::ComputeScores ( )

For each attribue, calculate chi-squared and associated p-value.

Return in a vector of pairs indexed by attribute index.

**Returns**

vector of pairs of chi-squared scores and associated p-values

Definition at line 34 of file ChiSquared.cpp.

#### 6.3.3.4 std::vector<std::vector<double> > ChiSquared::GetFrequencyCounts ( ) [inline]

Get the observed frequencies table as a vector of vector of doubles.

Definition at line 62 of file ChiSquared.h.

#### 6.3.3.5 void ChiSquared::PrepareForAttribute ( unsigned int *attributeIndex* ) [private]

Private method to setup the chi-squared contingency tables for a particular attribute.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | attribute index |

Definition at line 209 of file ChiSquared.cpp.

#### 6.3.3.6 void ChiSquared::PrintScores ( std::ofstream & *outStream,* unsigned int *topN* = 0 )

Print the scores to a stream.

**Parameters**

| | | |
|---|---|---|
| in | *outStream* | reference to an output stream |
| in | *topN* | top number of attributes to print |

Definition at line 176 of file ChiSquared.cpp.

#### 6.3.3.7 void ChiSquared::PrintTables ( )

Print calculation tables.

Definition at line 145 of file ChiSquared.cpp.

#### 6.3.3.8 void ChiSquared::WriteScores ( std::string *outFilename,* unsigned int *topN* = 0 )

Print the scores to a stream.

**Parameters**

| | | |
|---|---|---|
| in | *outFilename* | filename to write scores to |
| in | *topN* | top number of attributes to print |

Definition at line 193 of file ChiSquared.cpp.

### 6.3.4 Member Data Documentation

#### 6.3.4.1 std::vector<std::vector<double> > ChiSquared::chiSquaredValues [private]

chi squared computed values

Definition at line 86 of file ChiSquared.h.

#### 6.3.4.2 Dataset* ChiSquared::dataset [private]

pointer to a Dataset object

Definition at line 76 of file ChiSquared.h.

#### 6.3.4.3 std::vector<std::vector<double> > ChiSquared::expectedContingencyTable [private]

Definition at line 84 of file ChiSquared.h.

#### 6.3.4.4 unsigned int ChiSquared::numClasses [private]

number of classes in the instances

Definition at line 80 of file ChiSquared.h.

#### 6.3.4.5 unsigned int ChiSquared::numLevels [private]

number of levels in the attributes

Definition at line 78 of file ChiSquared.h.

#### 6.3.4.6 std::vector<std::vector<double> > ChiSquared::observedFreqTable [private]

observed frequencies

Definition at line 82 of file ChiSquared.h.

#### 6.3.4.7 std::vector<std::pair<double, double> > ChiSquared::scores [private]

chi-squared value, p-value for each attribute

Definition at line 88 of file ChiSquared.h.

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

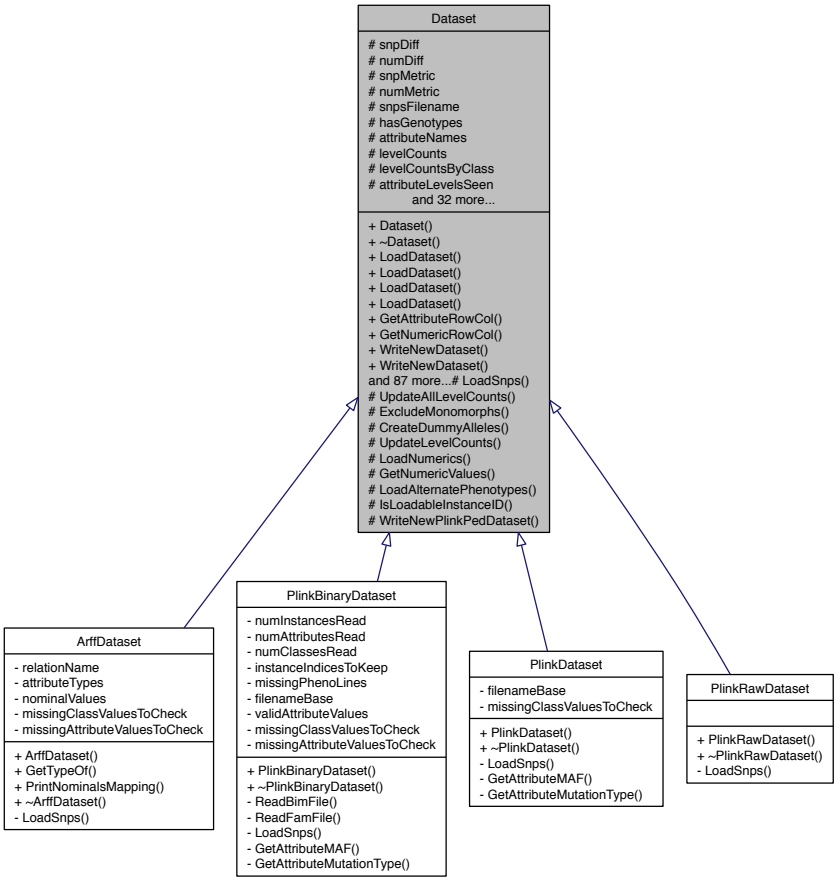
- src/library/ChiSquared.h
- src/library/ChiSquared.cpp

## 6.4 Dataset Class Reference

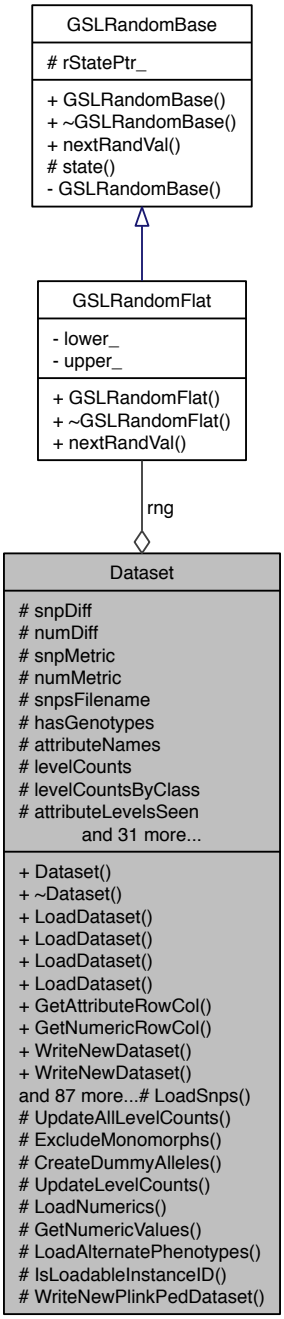
Base class for collections of instances containing attributea and class.

`#include <Dataset.h>`

Inheritance diagram for Dataset:



Collaboration diagram for Dataset:




## Public Member Functions

- Dataset ()

  *Construct a default data set.*

- virtual ~Dataset ()

  *Destruct all dynamically allocated memory.*

- bool LoadDataset (std::vector< std::vector< int > > &dataMatrix, std::vector< int > &classLabels, std::vector< std::string > &attrNames)

  *Load the data set from "raw data".*
- bool LoadDataset (std::string snpsFilename, std::string numericsFilename, std::string altPhenoFilename, std::vector< std::string > ids)

  *Load the data set from files passed as parameters.*
- bool LoadDataset (DgeData *dgeData)

  *Load the data set from DGE data.*
- bool LoadDataset (BirdseedData *birdseedData)

  *Load the data set from Birdseed data.*
- bool GetAttributeRowCol (unsigned int row, unsigned int col, AttributeLevel &attrVal)

  *Get the attribute value at row, column.*
- bool GetNumericRowCol (unsigned int row, unsigned int col, NumericLevel &numVal)

  *Get the numeric value at row, column.*
- bool WriteNewDataset (std::string newDatasetFilename, OutputDatasetType outputDatasetType)

  *Write the data set to a new filename, respecting masked attributes and numerics and class/phenotype data type.*
- bool WriteNewDataset (std::string newDatasetFilename, std::vector< std::string > attributes, OutputDatasetType outputDatasetType)

  *Write the data set to a new filename, writing only the names in the passed attributes list and also respecting masked attributes and numerics and class/phenotype data type.*
- bool ExtractAttributes (std::string scoresFilename, unsigned int topN, std::string newDatasetFilename)

  *Extracts top N attributes based on a file of attribute scores and writes a new dataset.*
- bool SwapAttributes (unsigned int a1, unsigned int a2)

  *Swap two attributes/columns in the dataset.*
- unsigned int NumVariables ()

  *Return the number of discrete plus continuous variables in the data set.*
- std::vector< std::string > GetVariableNames ()

  *Returns the names of discrete and continuous variables in the data set.*
- virtual unsigned int NumInstances ()

  *Returns the number of instances in the data set.*
- DatasetInstance * GetInstance (unsigned int index)

  *Returns a pointer to a dataset instance selected by index.*
- DatasetInstance * GetRandomInstance ()

  *Returns a pointer to a randomly chosen data set instance.*
- std::vector< std::string > GetInstanceIds ()

  *Get all instance IDs.*
- bool GetInstanceIndexForID (std::string ID, unsigned int &instanceIndex)

  *Get the instance index from the instance ID.*
- virtual unsigned int NumAttributes ()

  *Return the number of unmasked discrete attributes in the data set.*
- std::vector< std::string > GetAttributeNames ()

  *Return the discrete (SNP) attribute names.*
- bool GetAttributeValues (unsigned int attributeIndex, std::vector< AttributeLevel > &attributeValues)

  *Loads the referenced vector with an attribute's values (column).*
- bool GetAttributeValues (std::string attributeName, std::vector< AttributeLevel > &attributeValues)

  *Loads the referenced vector with an attribute's values (column) from the dataset.*
- std::string GetSnpsFilename ()

  *Get the filename SNPs were read from.*
- unsigned int GetAttributeIndexFromName (std::string attributeName)

  *Looks up original attribute index from attribute name.*
- bool HasGenotypes ()

*Does the data set have genotype variables?*

- bool HasAllelicInfo ()

  *Does the data set have allelic information for genotypes?*
- AttributeLevel GetAttribute (unsigned instanceIndex, std::string name)

  *Get attribute value for attribute name at instance index.*
- std::pair< char, char > GetAttributeAlleles (unsigned int attributeIndex)

  *Get attribute major and minor alleles.*
- virtual std::pair< char, double > GetAttributeMAF (unsigned int attributeIndex)

  *Get attribute minor allele and frequency.*
- bool ProcessExclusionFile (std::string exclusionFilename)

  *Remove file of attribute names from consideration in analyses.*
- virtual AttributeMutationType GetAttributeMutationType (unsigned int attributeIndex)

  *Get attribute mutation type.*
- double GetJukesCantorDistance (DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Apply Jukes-Cantor distance.*
- double GetKimuraDistance (DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Get Kimura Two-Parameter distance.*
- bool GetIntForGenotype (std::string genotype, AttributeLevel &newAttr)

  *Get integer value for string genotype.*
- unsigned int NumLevels (unsigned int index)

  *Returns the number of levels in a given attribute index.*
- unsigned int NumNumerics ()

  *Return the number of unmasked discrete attributes in the data set.*
- std::vector< std::string > GetNumericsNames ()

  *Return the numeric attribute names.*
- std::pair< double, double > GetMinMaxForNumeric (unsigned int numericIdx)

  *Get the minimum and maximum values for a numeric at index.*
- double GetMeanForNumeric (unsigned int numericIdx)

  *Get the mean/average of numeric at index.*
- bool HasNumerics ()

  *Does the data set have numeric variables? setter/getter.*
- void HasNumerics (bool setHasNumerics)
- NumericLevel GetNumeric (unsigned int instanceIndex, std::string name)

  *Get numeric value for numeric name at instance index.*
- bool GetNumericValues (std::string numericName, std::vector< NumericLevel > &numericValues)

  *Loads the referenced vector with a numeric's values (column) from the dataset.*
- std::string GetNumericsFilename ()

  *Get the filename numerics were read from.*
- unsigned int GetNumericIndexFromName (std::string numericName)

  *Looks up original numeric index from numeric name.*
- unsigned int NumClasses ()

  *Get the number of classes in the data set.*
- unsigned int GetClassColumn ()

  *Get the class column as read from the file.*
- bool GetClassValues (std::vector< ClassLevel > &classValues)

  *Loads the referenced vector with the dataset's class labels.*
- const std::map< ClassLevel,
  std::vector< unsigned int > > & GetClassIndexes ()

  *Get a map from class levels to a vector of instance indices.*
- bool HasAlternatePhenotypes ()

  *Does the data set have alternate phenotypes loaded?*

- void HasAlternatePhenotypes (bool setHasAlternatePhenotypes)
- std::string GetAlternatePhenotypesFilename ()

  *Get the alternate phenotype filename.*
- bool HasContinuousPhenotypes ()

  *Does the data set have continuous phenotypes?*
- bool HasPhenotypes ()

  *Does the data set have any valid phenotypes?*
- std::pair< double, double > GetMinMaxForContinuousPhenotype ()

  *Get the minumum and maximum values for the continuous phenotype.*
- void Print ()

  *Print the entire data set in compact format.*
- void PrintStats ()

  *Print basic statstics about the data set - discrete/SNPs only.*
- void PrintNumericsStats ()

  *Print statistics about the data set including numerics.*
- void PrintStatsSimple (std::ostream &outStream=std::cout)

  *Print very simple statistics abou the data set with no formatting.*
- void PrintClassIndexInfo (std::ostream &outStream=std::cout)

  *Print class index information.*
- void PrintMissingValuesStats ()

  *Print missing value statistics.*
- void PrintLevelCounts ()

  *Prit attribute level counts.*
- void WriteLevelCounts (std::string levelsFilename)

  *Write attribute level counts to a text file.*
- void PrintAttributeLevelsSeen ()

  *Print unique attribute levels seen.*
- bool MaskRemoveVariable (std::string variableName)

  *Removes the variable name from consideration in any data set operations.*
- bool MaskRemoveVariableType (std::string variableName, AttributeType varType)

  *Removes the attribute name from consideration in any data set operations.*
- bool MaskSearchVariableType (std::string variableName, AttributeType attrType)

  *Determines if the named variable is in the current masked data set.*
- bool MaskIncludeAllAttributes (AttributeType attrType)

  *Mark all attributes for inclusion in data set operations.*
- std::vector< unsigned int > MaskGetAttributeIndices (AttributeType attrType)

  *Return a vector of all the attribute indices under consideration.*
- const std::map< std::string,
  unsigned int > & MaskGetAttributeMask (AttributeType attrType)

  *Return a map of attribute name to attribute index of attributes to include.*
- std::vector< std::string > MaskGetAllVariableNames ()

  *Return a vector of all the variable names under consideration.*
- bool MaskRemoveInstance (std::string instanceId)

  *Removes the instance from consideration in any data set operations.*
- bool MaskSearchInstance (std::string instanceId)

  *Determines if the names Instance is in the current masked dataaset.*
- bool MaskIncludeAllInstances ()

  *Mark all instances for inclusion in algorithms.*
- std::vector< unsigned int > MaskGetInstanceIndices ()

  *Return a vector of all the instance indices under consideration.*
- std::vector< std::string > MaskGetInstanceIds ()

*Return a vector of all the instance ids under consideration.*

- const std::map< std::string, unsigned int > & MaskGetInstanceMask ()

  *Return a map of instance name to instance index of instances to include.*

- bool MaskPushAll ()

  *Save the current masks for later restore.*

- bool MaskPopAll ()

  *Restore the masks previously pushed.*

- bool MaskWriteNewDataset (std::string newDatasetFilename)

  *Saved the unmasked attributes as a tab-delimited text file.*

- void PrintMaskStats ()

  *Print mask statistics.*

- void RunSnpDiagnosticTests (std::string logFilename, double globalGenotypeThreshold=0.01, unsigned int cellThreshold=5)

  *Perform and report SNP diagnostic test information.*

- bool CheckHardyWeinbergEquilibrium (std::vector< unsigned int > &chkGenotypeCounts)

  *Calculate whether passed genotype counts are in HWE.*

- double SNPHWE (int obs_hets, int obs_hom1, int obs_hom2)

  *This code implements an exact SNP test of Hardy-Weinberg Equilibrium.*

- double GetClassProbability (ClassLevel thisClass)

  *Get the probability of a class value in the data set.*

- double GetProbabilityValueGivenClass (unsigned int attributeIndex, AttributeLevel A, ClassLevel classValue)

  *Get the probability of an attribute value at an attribute index.*

- void AttributeInteractionInformation ()

  *Calculate and display interaction information for all attribute combinations.*

- void CalculateInteractionInformation (std::map< std::pair< int, int >, std::map< std::string, double > > &results)

  *Calculate all the information needed to construct the interaction diagram.*

- bool CalculateGainMatrix (double **gainMatrix, std::string matrixFilename="")

  *Calculate the GAIN matrix to run snprank on this data set.*

- bool CalculateDistanceMatrix (double **distanceMatrix, std::string matrixFilename="")

  *Calculate the instance-to-instance distance matrix for this data set.*

- bool CalculateDistanceMatrix (std::vector< std::vector< double > > &distanceMatrix)

  *Calculate the instance-to-instance distance matrix for this data set.*

- double ComputeInstanceToInstanceDistance (DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Compute the distance between two DatasetInstances.*

- bool SetDistanceMetrics (std::string newSnpMetric, std::string newNumMetric="manhattan")

  *Set the the distance metrics used to compute instance-to-instance distances.*

- std::pair< std::string, std::string > GetDistanceMetrics ()

  *Get the the distance metrics used to compute instance-to-instance distances.*

- std::pair< unsigned int, unsigned int > GetAttributeTiTvCounts ()

  *Get the the mutation transition and transversion counts.*

- std::pair < RandomJungleTreeType, std::string > DetermineTreeType ()

  *Determine the Random Jungle tree type from data set characteristics.*

- bool WriteSnpTiTvInfo (std::string titvFilename)

  *Dump the SNP transition/transversion information to file.*

### Protected Member Functions

- virtual bool LoadSnps (std::string filename)

  *Load SNPs from file using the data set filename.*
- void UpdateAllLevelCounts ()

  *Update level counts for all instances by calling UpdateLevelCounts(inst)*
- void ExcludeMonomorphs ()

  *Exclude any monomorphic SNPs, since they add no information about class.*
- void CreateDummyAlleles ()

  *Create dummy alleles from genotypes for data sets that have no allele info.*
- void UpdateLevelCounts (DatasetInstance *dsi)

  *Update all attribute level counts from one data set instance.*
- bool LoadNumerics (std::string filename)

  *Load numerics (continuous attributes) from a file set in the constructor.*
- bool GetNumericValues (unsigned int numericIndex, std::vector< NumericLevel > &numericValues)

  *Loads the referenced vector with an numeric's values (column).*
- bool LoadAlternatePhenotypes (std::string filename)

  *Load alternate phenotype/class values from a plink covariate .cov file.*
- bool IsLoadableInstanceID (std::string ID)

  *Is the passed instance ID loadable (not filtered).*
- bool WriteNewPlinkPedDataset (std::string baseDatasetFilename)

  *Write the dataset to a new PLINK PED/MAP format, respecting masked attributes class/phenotype data type.*

### Protected Attributes

- double(* snpDiff )(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Compute the discrete difference in an attribute between two instances for determining nearest neighbors.*
- double(* numDiff )(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Compute the continuous difference in an attribute between two instances.*
- std::string snpMetric

  *the name of discrete diff(erence) function*
- std::string numMetric

  *the name of continuous diff(erence) function*
- std::string snpsFilename

  *file from which the discrete attributes (SNPSs) were read*
- bool hasGenotypes

  *does the data set contain any genotypes?*
- std::vector< std::string > attributeNames

  *discrete attribute names read from file*
- std::vector< std::map < AttributeLevel, unsigned int > > levelCounts

  *attribute values/levels counts*
- std::vector< std::map < std::pair< AttributeLevel, ClassLevel >, unsigned int > > levelCountsByClass

  *attribute values/levels counts by discrete class*
- std::vector< std::set < std::string > > attributeLevelsSeen

  *unique attribute values/levels read from file*
- std::vector< std::pair< char, char > > attributeAlleles

*allele1, allele2*

- std::vector< std::map< char, unsigned int > > attributeAlleleCounts

  *allele->count*

- std::vector< std::pair< char, double > > attributeMinorAllele

  *minor allele, minor allele frequency*

- bool hasAllelicInfo

  *Does this data set have alelelic information?*

- std::vector< std::map < std::string, unsigned int > > genotypeCounts

  *genotype->count*

- std::vector < AttributeMutationType > attributeMutationTypes

  *Keep mutation type for all attributes.*

- std::map< std::pair< char, char >, AttributeMutationType > attributeMutationMap

  *Lookup table for mutation type.*

- std::string numericsFilename

  *file from which the continuous attributes were read*

- bool hasNumerics

  *does the data set contain any continuous attributes?*

- std::vector< std::string > numericsIds

  *IDs associated with the numerics read from file.*

- std::vector< std::pair < NumericLevel, NumericLevel > > numericsMinMax

  *the minimum and maximum value for each continuous attribute*

- std::vector< std::string > numericsNames

  *continuous attribute names read from file*

- bool hasPhenotypes

  *Does the data set contain phenotypes?*

- std::string alternatePhenotypesFilename

  *file from which the alternate phenotypes (class labels) were read*

- bool hasAlternatePhenotypes

  *does the data set contain alternate phenotypes?*

- std::vector< std::string > phenotypesIds

  *IDs associated with the phenotypes/classes read from file.*

- bool hasContinuousPhenotypes

  *does the data set contain continuous phenotypes?*

- std::pair< NumericLevel, NumericLevel > continuousPhenotypeMinMax

  *the minimum and maximum value for each continuous phenotype*

- std::vector< DatasetInstance * > instances

  *vector of pointers to all instances in the data set*

- std::vector< std::string > instanceIds

  *IDs associated with the instances read from file.*

- std::vector< std::string > instanceIdsToLoad

  *IDs of instances to load from numeric and/or phenotype files.*

- std::map< std::string, std::vector< unsigned int > > missingValues

  *missing discrete values and their instance indices*

- std::map< std::string, std::vector< unsigned int > > missingNumericValues

  *missing continuous values and their instance indices*
- unsigned int classColumn

  *class column from the original data set*
- std::map< ClassLevel, std::vector< unsigned int > > classIndexes

  *class values mapped to instance indices*
- std::map< std::string, unsigned int > attributesMask
- std::map< std::string, unsigned int > numericsMask
- std::map< std::string, unsigned int > instancesMask
- std::map< std::string, unsigned int > attributesMaskPushed

  *masks can be temporarily pushed and popped*
- std::map< std::string, unsigned int > numericsMaskPushed
- std::map< std::string, unsigned int > instancesMaskPushed
- bool maskIsPushed
- GSLRandomFlat * rng

  *random number generator classes use GNU Scienitifc Library (GSL)*

### 6.4.1 Detailed Description

Base class for collections of instances containing attributea and class.

Added interaction infomation week of 4/18-26/06 Totally redone for McKinney Lab. February 2011.

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 6/14/05

Definition at line 36 of file Dataset.h.

### 6.4.2 Constructor & Destructor Documentation

#### 6.4.2.1 Dataset::Dataset ( )

Construct a default data set.

Set private data to defaults.

Load attribute mutation map for transitions/transversions.

Definition at line 47 of file Dataset.cpp.

#### 6.4.2.2 Dataset::~Dataset ( ) [virtual]

Destruct all dynamically allocated memory.

Definition at line 94 of file Dataset.cpp.

### 6.4.3 Member Function Documentation

#### 6.4.3.1 void Dataset::AttributeInteractionInformation ( )

Calculate and display interaction information for all attribute combinations.

call CalculateInteractionInformation

display results of interaction calculations

display results header line

get the column sum

display results detail; I(A;B|C) column as percentage

Definition at line 2291 of file Dataset.cpp.

#### 6.4.3.2 bool Dataset::CalculateDistanceMatrix ( double ** *distanceMatrix,* std::string *matrixFilename* = " " )

Calculate the instance-to-instance distance matrix for this data set.

Uses OpenMP to calculate matrix entries in parallel threads.

**Parameters**

| | | |
|---|---|---|
| out | *distanceMatrix* | pointer to an allocated m x m matrix, m = number of instances |
| in | *distanceMatrix-Filename* | filename to write matrix |

**Returns**

success

#### 6.4.3.3 bool Dataset::CalculateDistanceMatrix ( std::vector< std::vector< double > > & *distanceMatrix* )

Calculate the instance-to-instance distance matrix for this data set.

Uses OpenMP to calculate matrix entries in parallel threads.

**Parameters**

| | | |
|---|---|---|
| out | *distanceMatrix* | vector of vectors of double: m x m matrix, m = number of instances |

**Returns**

success

#### 6.4.3.4 bool Dataset::CalculateGainMatrix ( double ** *gainMatrix,* std::string *matrixFilename* = " " )

Calculate the GAIN matrix to run snprank on this data set.

Uses OpenMP to calculate matrix entries in parallel threads.

**Parameters**

| | | |
|---|---|---|
| out | *gainMatrix* | pointer to an allocated n x n matrix, n = number of attributes |

**Returns**

success

Calculate the interaction information from entropies

Populate the GAIN matrix

write header

write all m-by-m matrix entries

Definition at line 2481 of file Dataset.cpp.

#### 6.4.3.5 void Dataset::CalculateInteractionInformation ( std::map< std::pair< int, int >, std::map< std::string, double > > & *results* )

Calculate all the information needed to construct the interaction diagram.

**Parameters**

| | | |
|---|---|---|
| out | *results* | map of attribute combinations to results |

so many way to fail before getting started

vectors to hold sequences for attributes a and b with class c ab is an attribute constructed from the cartesian product of a and b

Get the class values once

for all possible (unique) interactions, ie nCk

use OpenMP to run in parallel the construction of the attribute interaction matrix

THREAD STARTS

load attribute values (columns) into vectors for entropy routines

construct a new attribute that is the cartesian product of a and b

compute all entropies, calculate information theoretic quantities and save the results

inner loop over j ends THREAD ENDS

outer loop over i ends

end - no return value

Definition at line 2350 of file Dataset.cpp.

#### 6.4.3.6 bool Dataset::CheckHardyWeinbergEquilibrium ( std::vector< unsigned int > & *chkGenotypeCounts* )

Calculate whether passed genotype counts are in HWE.

**Parameters**

| | |
|---|---|
| *genotypeCounts* | vector of genotype counts: AA, Aa, aa |

**Returns**

counts are in HWE?

observered counts

HWE probabilities

expected values

perform Pearson's chi-squared test

one degree of freedom (# genotypes - # alleles), 5% significance level

Definition at line 2112 of file Dataset.cpp.

#### 6.4.3.7 double Dataset::ComputeInstanceToInstanceDistance ( DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Compute the distance between two DatasetInstances.

**Parameters**

| | | |
|---|---|---|
| in | *dsi1* | pointer to DatasetInstance 1 |
| in | *dsi2* | pointer to DatasetInstance 2 |

**Returns**

distance

Definition at line 2550 of file Dataset.cpp.

#### 6.4.3.8 void Dataset::CreateDummyAlleles ( ) [protected]

Create dummy alleles from genotypes for data sets that have no allele info.

assign major and minor alleles

Definition at line 3089 of file Dataset.cpp.

#### 6.4.3.9 pair< RandomJungleTreeType, string > Dataset::DetermineTreeType ( )

Determine the Random Jungle tree type from data set characteristics.

Definition at line 2652 of file Dataset.cpp.

#### 6.4.3.10 void Dataset::ExcludeMonomorphs ( ) [protected]

Exclude any monomorphic SNPs, since they add no information about class.

Definition at line 3072 of file Dataset.cpp.

#### 6.4.3.11 bool Dataset::ExtractAttributes ( std::string *scoresFilename*, unsigned int *topN*, std::string *newDatasetFilename* )

Extracts top N attributes based on a file of attribute scores and writes a new dataset.

Revised 10/3/11 for numerics and continuous class/phenotypes.

**Parameters**

| | | |
|---|---|---|
| in | *scoresFilename* | filename of attribute scores and names |
| in | *topN* | top N attributes |
| in | *newDataset-Filename* | filename of new dataset |

**Returns**

success

Definition at line 845 of file Dataset.cpp.

#### 6.4.3.12 string Dataset::GetAlternatePhenotypesFilename ( )

Get the alternate phenotype filename.

Definition at line 1343 of file Dataset.cpp.

#### 6.4.3.13 AttributeLevel Dataset::GetAttribute ( unsigned *instanceIndex,* std::string *name* )

Get attribute value for attribute name at instance index.

**Parameters**

| | | |
|---|---|---|
| in | *instanceIndex* | instance index |
| in | *name* | attribute name |

**Returns**

attributevalue

Definition at line 1057 of file Dataset.cpp.

#### 6.4.3.14 pair< char, char > Dataset::GetAttributeAlleles ( unsigned int *attributeIndex* )

Get attribute major and minor alleles.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

pair (major allele, minor allele frequency)

Definition at line 1073 of file Dataset.cpp.

#### 6.4.3.15 unsigned int Dataset::GetAttributeIndexFromName ( std::string *attributeName* )

Looks up original attribute index from attribute name.

**Parameters**

| | | |
|---|---|---|
| in | *attributeName* | attribute name |

**Returns**

attribute index or INVALID_INDEX

Definition at line 1215 of file Dataset.cpp.

#### 6.4.3.16 pair< char, double > Dataset::GetAttributeMAF ( unsigned int *attributeIndex* ) [virtual]

Get attribute minor allele and frequency.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

pair (minor allele, minor allele frequency)

An Introduction to Genetic Analysis by Griffiths, Miller, Suzuki, Lewontin and Gelbart, 2000, page 715.

Reimplemented in PlinkDataset, and PlinkBinaryDataset.

Definition at line 1091 of file Dataset.cpp.

#### 6.4.3.17 AttributeMutationType Dataset::GetAttributeMutationType ( unsigned int *attributeIndex* ) [virtual]

Get attribute mutation type.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

mutation type (transition, transversion, unknown)

Reimplemented in PlinkDataset, and PlinkBinaryDataset.

Definition at line 1147 of file Dataset.cpp.

#### 6.4.3.18 vector< string > Dataset::GetAttributeNames ( )

Return the discrete (SNP) attribute names.

**Returns**

vector of attribute names

Definition at line 998 of file Dataset.cpp.

#### 6.4.3.19 bool Dataset::GetAttributeRowCol ( unsigned int *row,* unsigned int *col,* AttributeLevel & *attrVal* )

Get the attribute value at row, column.

Same as instance index, attribute index.

**Parameters**

| | | |
|---|---|---|
| in | *row* | instance row |
| in | *col* | attribute column |
| out | *attrVal* | attribute value |

**Returns**

success

Definition at line 426 of file Dataset.cpp.

#### 6.4.3.20 pair< unsigned int, unsigned int > Dataset::GetAttributeTiTvCounts ( )

Get the the mutation transition and transversion counts.

**Returns**

pair<number of transitions, number of transversions>

Definition at line 2637 of file Dataset.cpp.

#### 6.4.3.21 bool Dataset::GetAttributeValues ( unsigned int *attributeIndex,* std::vector< AttributeLevel > & *attributeValues* )

Loads the referenced vector with an attribute's values (column).

from the dataset

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | attribute index |
| out | *attributeValues* | reference to a a vector allocated by the caller |

**Returns**

success

#### 6.4.3.22 bool Dataset::GetAttributeValues ( std::string *attributeName,* std::vector< AttributeLevel > & *attributeValues* )

Loads the referenced vector with an attribute's values (column) from the dataset.

**Parameters**

| | | |
|---|---|---|
| in | *attributeName* | attribute name |
| out | *attributeValues* | reference to a a vector allocated by the caller |

**Returns**

success

#### 6.4.3.23 unsigned int Dataset::GetClassColumn ( )

Get the class column as read from the file.

Definition at line 1307 of file Dataset.cpp.

#### 6.4.3.24 const std::map< ClassLevel, std::vector< unsigned int > > & Dataset::GetClassIndexes ( )

Get a map from class levels to a vector of instance indices.

**Returns**

map of class => instance indices

Definition at line 1331 of file Dataset.cpp.

#### 6.4.3.25 double Dataset::GetClassProbability ( ClassLevel *thisClass* )

Get the probability of a class value in the data set.

**Parameters**

| | |
|---|---|
| *thisClass* | class value |

**Returns**

probability

Definition at line 2269 of file Dataset.cpp.

#### 6.4.3.26 bool Dataset::GetClassValues ( std::vector< ClassLevel > & *classValues* )

Loads the referenced vector with the dataset's class labels.

**Parameters**

| | | |
|---|---|---|
| out | *classValues* | reference to a a vector allocated by the caller |

**Returns**

success

Definition at line 1316 of file Dataset.cpp.

#### 6.4.3.27 pair< string, string > Dataset::GetDistanceMetrics ( )

Get the the distance metrics used to compute instance-to-instance distances.

**Returns**

pair<snp distance metric name, numeric distance metric name>

Definition at line 2633 of file Dataset.cpp.

#### 6.4.3.28 DatasetInstance * Dataset::GetInstance ( unsigned int *index* )

Returns a pointer to a dataset instance selected by index.

**Parameters**

| | | |
|---|---|---|
| in | *index* | index of instance |

**Returns**

pointer to an instance

Definition at line 960 of file Dataset.cpp.

#### 6.4.3.29 vector< string > Dataset::GetInstanceIds ( )

Get all instance IDs.

**Returns**

vector of instance IDs

Definition at line 973 of file Dataset.cpp.

#### 6.4.3.30 bool Dataset::GetInstanceIndexForID ( std::string *ID,* unsigned int & *instanceIndex* )

Get the instance index from the instance ID.

**Parameters**

| | | |
|---|---|---|
| in | *ID* | string ID |
| out | *instanceIndex* | instance index |

**Returns**

success

Definition at line 982 of file Dataset.cpp.

#### 6.4.3.31 bool Dataset::GetIntForGenotype ( std::string *genotype,* AttributeLevel & *newAttr* )

Get integer value for string genotype.

**Parameters**

| | | |
|---|---|---|
| in | *genotype* | genotype string |
| out | *newAttr* | new attribute value |

**Returns**

success

#### 6.4.3.32 double Dataset::GetJukesCantorDistance ( DatasetInstance * *dsi1,* DatasetInstance * *dsi2* )

Apply Jukes-Cantor distance.

Definition at line 1155 of file Dataset.cpp.

#### 6.4.3.33 double Dataset::GetKimuraDistance ( DatasetInstance * *dsi1,* DatasetInstance * *dsi2* )

Get Kimura Two-Parameter distance.

Definition at line 1174 of file Dataset.cpp.

#### 6.4.3.34 double Dataset::GetMeanForNumeric ( unsigned int *numericIdx* )

Get the mean/average of numeric at index.

**Parameters**

| | | |
|---|---|---|
| in | *numericIdx* | numeric index |

**Returns**

average value of numeric attribute at index

Definition at line 1242 of file Dataset.cpp.

#### 6.4.3.35 pair< double, double > Dataset::GetMinMaxForContinuousPhenotype ( )

Get the minumum and maximum values for the continuous phenotype.

**Returns**

minimum/maximum pair

Definition at line 1355 of file Dataset.cpp.

#### 6.4.3.36 pair< NumericLevel, NumericLevel > Dataset::GetMinMaxForNumeric ( unsigned int *numericIdx* )

Get the minimum and maximum values for a numeric at index.

**Parameters**

| | | |
|---|---|---|
| in | *numericIdx* | numeric index |

**Returns**

minimum/maximum pair

Definition at line 1237 of file Dataset.cpp.

#### 6.4.3.37 NumericLevel Dataset::GetNumeric ( unsigned int *instanceIndex,* std::string *name* )

Get numeric value for numeric name at instance index.

**Parameters**

| | | |
|---|---|---|
| in | *instanceIndex* | instance index |
| in | *name* | numeric name |

**Returns**

numeric value at index

Definition at line 1260 of file Dataset.cpp.

#### 6.4.3.38 unsigned int Dataset::GetNumericIndexFromName ( std::string *numericName* )

Looks up original numeric index from numeric name.

**Parameters**

| | | |
|---|---|---|
| in | *numericName* | numeric name |

**Returns**

attribute index or INVALID_INDEX

Definition at line 1291 of file Dataset.cpp.

#### 6.4.3.39 bool Dataset::GetNumericRowCol ( unsigned int *row,* unsigned int *col,* NumericLevel & *numVal* )

Get the numeric value at row, column.

Same as instance index, numeric index.

**Parameters**

| | | |
|---|---|---|
| in | *row* | instance row |
| in | *col* | numeric column |
| out | *numVal* | numeric value |

**Returns**

success

Definition at line 439 of file Dataset.cpp.

#### 6.4.3.40 std::string Dataset::GetNumericsFilename ( )

Get the filename numerics were read from.

Definition at line 1287 of file Dataset.cpp.

#### 6.4.3.41 vector< string > Dataset::GetNumericsNames ( )

Return the numeric attribute names.

**Returns**

vector of attribute names

Definition at line 1228 of file Dataset.cpp.

#### 6.4.3.42 bool Dataset::GetNumericValues ( std::string *numericName,* std::vector< NumericLevel > & *numericValues* )

Loads the referenced vector with a numeric's values (column) from the dataset.

**Parameters**

| | | |
|---|---|---|
| in | *numericName* | numeric name |
| out | *numericValues* | reference to a a vector allocated by the caller |

**Returns**

success

#### 6.4.3.43 bool Dataset::GetNumericValues ( unsigned int *numericIndex*, std::vector< NumericLevel > & *numericValues* ) [protected]

Loads the referenced vector with an numeric's values (column).

from the dataset

**Parameters**

| | | |
|---|---|---|
| in | *numericIndex* | numeric index |
| out | *numericValues* | reference to a a vector allocated by the caller |

**Returns**

success

#### 6.4.3.44 double Dataset::GetProbabilityValueGivenClass ( unsigned int *attributeIndex*, AttributeLevel *A*, ClassLevel *classValue* )

Get the probability of an attribute value at an attribute index.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | attribute index |
| in | *A* | attribute value |
| in | *classValue* | class value |

**Returns**

probability of the value in attribute given class

Definition at line 2276 of file Dataset.cpp.

#### 6.4.3.45 DatasetInstance * Dataset::GetRandomInstance ( )

Returns a pointer to a randomly chosen data set instance.

The random number generator is set to give values in range of instance indexes.

**Returns**

pointer to a data set instance

Definition at line 968 of file Dataset.cpp.

#### 6.4.3.46 std::string Dataset::GetSnpsFilename ( )

Get the filename SNPs were read from.

Definition at line 1045 of file Dataset.cpp.

#### 6.4.3.47 vector< string > Dataset::GetVariableNames ( )

Returns the names of discrete and continuous variables in the data set.

**Returns**

vector of names as strings

Definition at line 944 of file Dataset.cpp.

#### 6.4.3.48 bool Dataset::HasAllelicInfo ( )

Does the data set have allelic information for genotypes?

Definition at line 1053 of file Dataset.cpp.

#### 6.4.3.49 bool Dataset::HasAlternatePhenotypes ( )

Does the data set have alternate phenotypes loaded?

Definition at line 1335 of file Dataset.cpp.

#### 6.4.3.50 void Dataset::HasAlternatePhenotypes ( bool *setHasAlternatePhenotypes* )

Definition at line 1339 of file Dataset.cpp.

#### 6.4.3.51 bool Dataset::HasContinuousPhenotypes ( )

Does the data set have continuous phenotypes?

Definition at line 1347 of file Dataset.cpp.

#### 6.4.3.52 bool Dataset::HasGenotypes ( )

Does the data set have genotype variables?

Definition at line 1049 of file Dataset.cpp.

#### 6.4.3.53 bool Dataset::HasNumerics ( )

Does the data set have numeric variables? setter/getter.

Definition at line 1252 of file Dataset.cpp.

#### 6.4.3.54 void Dataset::HasNumerics ( bool *setHasNumerics* )

Definition at line 1256 of file Dataset.cpp.

#### 6.4.3.55 bool Dataset::HasPhenotypes ( )

Does the data set have any valid phenotypes?

Definition at line 1351 of file Dataset.cpp.

#### 6.4.3.56 bool Dataset::IsLoadableInstanceID ( std::string *ID* ) [protected]

Is the passed instance ID loadable (not filtered).

**Parameters**

| | | |
|---|---|---|
| in | *ID* | instance ID |

**Returns**

[out] success

Definition at line 3490 of file Dataset.cpp.

#### 6.4.3.57 bool Dataset::LoadAlternatePhenotypes ( std::string *filename* ) [protected]

Load alternate phenotype/class values from a plink covariate .cov file.

Format described here: http://pngu.mgh.harvard.edu/~purcell/plink/data.shtml#covar
MAJOR CHANGES: for continuous phenotypes/class - 9/29/11

**Parameters**

| | | |
|---|---|---|
| in | *filename* | alternate phenotype data filename in PLINK covar format |

**Returns**

success

Detect the class type

Definition at line 3306 of file Dataset.cpp.

#### 6.4.3.58 bool Dataset::LoadDataset ( std::vector< std::vector< int > > & *dataMatrix*, std::vector< int > & *classLabels*, std::vector< std::string > & *attrNames* )

Load the data set from "raw data".

**Parameters**

| | | |
|---|---|---|
| in | *dataMatrix* | reference to a matrix of SNP values 0-1-2-missing |
| in | *classLabels* | reference to a vector of case-control labels |
| in | *attrNames* | reference to a vector of attribute names |

**Returns**

success

#### 6.4.3.59 bool Dataset::LoadDataset ( std::string *snpsFilename*, std::string *numericsFilename*, std::string *altPhenoFilename*, std::vector< std::string > *ids* )

Load the data set from files passed as parameters.

**Parameters**

| | | |
|---|---|---|
| in | *snpFilename* | discrete values (SNPs) filename |
| in | *doRecodeA* | perform recodeA encoding after reading |

| | | |
|---|---|---|
| in | *numerics-Filename* | continuous values (numerics) filename or empty string |
| in | *altPheno-Filename* | alternate class (phenotype) filename or empty string |
| in | *ids* | vector of possibly empty IDs to match in auxiliary files |

**Returns**

success

#### 6.4.3.60 bool Dataset::LoadDataset ( DgeData * *dgeData* )

Load the data set from DGE data.

**Parameters**

| | | |
|---|---|---|
| in | *dgeData* | pointer to a digital gene expression (DGE) data object |

**Returns**

success

Definition at line 270 of file Dataset.cpp.

#### 6.4.3.61 bool Dataset::LoadDataset ( BirdseedData * *birdseedData* )

Load the data set from Birdseed data.

**Parameters**

| | | |
|---|---|---|
| in | *birdseedData* | pointer to a Birdseed-called SNP data object |

**Returns**

success

add allelic info

Definition at line 319 of file Dataset.cpp.

#### 6.4.3.62 bool Dataset::LoadNumerics ( std::string *filename* ) [protected]

Load numerics (continuous attributes) from a file set in the constructor.

**Parameters**

| | | |
|---|---|---|
| in | *filename* | numerics data filename in PLINK covar format |

**Returns**

success

Definition at line 3141 of file Dataset.cpp.

#### 6.4.3.63 bool Dataset::LoadSnps ( std::string *filename* ) [protected, virtual]

Load SNPs from file using the data set filename.

---------- Beginning of private methods ----------

**Parameters**

| | | |
|---|---|---|
| in | *filename* | SNPs filename |
| in | *deRecodeA* | perform a recodeA operation after reading raw data? |

**Returns**

success

Open the data file and read line-by-line

Detect the class type

Reimplemented in ArffDataset, PlinkDataset, PlinkBinaryDataset, and PlinkRawDataset.

Definition at line 2808 of file Dataset.cpp.

#### 6.4.3.64 vector< string > Dataset::MaskGetAllVariableNames ( )

Return a vector of all the variable names under consideration.

**Returns**

vector of discrete and numeric variable

Definition at line 1761 of file Dataset.cpp.

#### 6.4.3.65 vector< unsigned int > Dataset::MaskGetAttributeIndices ( AttributeType *attrType* )

Return a vector of all the attribute indices under consideration.

**Parameters**

| | |
|---|---|
| *attrType* | attribute type |

**Returns**

vector of indices into currently considered discrete attributes

Definition at line 1736 of file Dataset.cpp.

#### 6.4.3.66 const map< string, unsigned int > & Dataset::MaskGetAttributeMask ( AttributeType *attrType* )

Return a map of attribute name to attribute index of attributes to include.

**Parameters**

| | | |
|---|---|---|
| in | *attrType* | attribute type |

**Returns**

attributes mask: name->index

Definition at line 1753 of file Dataset.cpp.

#### 6.4.3.67 vector< string > Dataset::MaskGetInstanceIds ( )

Return a vector of all the instance ids under consideration.

**Returns**

vector of ids of currently included instances

Definition at line 1816 of file Dataset.cpp.

#### 6.4.3.68 vector< unsigned int > Dataset::MaskGetInstanceIndices ( )

Return a vector of all the instance indices under consideration.

vector of indices into current instances

Definition at line 1807 of file Dataset.cpp.

#### 6.4.3.69 const map< string, unsigned int > & Dataset::MaskGetInstanceMask ( )

Return a map of instance name to instance index of instances to include.

**Returns**

instances mask: instance ID=>vector of instance indices

Definition at line 1825 of file Dataset.cpp.

#### 6.4.3.70 bool Dataset::MaskIncludeAllAttributes ( AttributeType *attrType* )

Mark all attributes for inclusion in data set operations.

**Parameters**

| | | |
|---|---|---|
| in | *attrType* | attribute type |

**Returns**

success

Definition at line 1714 of file Dataset.cpp.

#### 6.4.3.71 bool Dataset::MaskIncludeAllInstances ( )

Mark all instances for inclusion in algorithms.

**Returns**

success

Definition at line 1795 of file Dataset.cpp.

#### 6.4.3.72 bool Dataset::MaskPopAll ( )

Restore the masks previously pushed.

**Returns**

success

Definition at line 1843 of file Dataset.cpp.

#### 6.4.3.73 bool Dataset::MaskPushAll ( )

Save the current masks for later restore.

**Returns**

success

Definition at line 1829 of file Dataset.cpp.

#### 6.4.3.74 bool Dataset::MaskRemoveInstance ( std::string *instanceId* )

Removes the instance from consideration in any data set operations.

**Parameters**

| | | |
|---|---|---|
| in | *instanceId* | instance ID |

**Returns**

success

Definition at line 1774 of file Dataset.cpp.

#### 6.4.3.75 bool Dataset::MaskRemoveVariable ( std::string *variableName* )

Removes the variable name from consideration in any data set operations.

**Parameters**

| | | |
|---|---|---|
| in | *variableName* | variable name |

**Returns**

success

Definition at line 1656 of file Dataset.cpp.

#### 6.4.3.76 bool Dataset::MaskRemoveVariableType ( std::string *variableName,* AttributeType *varType* )

Removes the attribute name from consideration in any data set operations.

**Parameters**

| | | |
|---|---|---|
| in | *attributeName* | attribute name |
| in | *attrType* | attribute type |

**Returns**

success

Definition at line 1668 of file Dataset.cpp.

#### 6.4.3.77 bool Dataset::MaskSearchInstance ( std::string *instanceId* )

Determines if the names Instance is in the current masked dataaset.

**Parameters**

| | | |
|---|---|---|
| in | *instanceID* | instance ID |

**Returns**

true if instance ID is in the dataset, considering instance mask

Definition at line 1786 of file Dataset.cpp.

#### 6.4.3.78 bool Dataset::MaskSearchVariableType ( std::string *variableName,* AttributeType *attrType* )

Determines if the named variable is in the current masked data set.

**Parameters**

| | | |
|---|---|---|
| in | *attributeName* | attribute name |
| in | *attributeType* | attribute type |

**Returns**

true if discrete attribute name is being considered in operations.

Definition at line 1694 of file Dataset.cpp.

#### 6.4.3.79 bool Dataset::MaskWriteNewDataset ( std::string *newDatasetFilename* )

Saved the unmasked attributes as a tab-delimited text file.

**Parameters**

| | | |
|---|---|---|
| in | *newDataset-Filename* | new data set filename |

**Returns**

success

Definition at line 1856 of file Dataset.cpp.

#### 6.4.3.80 unsigned int Dataset::NumAttributes ( ) [virtual]

Return the number of unmasked discrete attributes in the data set.

Definition at line 994 of file Dataset.cpp.

#### 6.4.3.81 unsigned int Dataset::NumClasses ( )

Get the number of classes in the data set.

Definition at line 1300 of file Dataset.cpp.

#### 6.4.3.82 unsigned int Dataset::NumInstances ( ) [virtual]

Returns the number of instances in the data set.

Definition at line 956 of file Dataset.cpp.

#### 6.4.3.83 unsigned int Dataset::NumLevels ( unsigned int *index* )

Returns the number of levels in a given attribute index.

**Parameters**

| | | |
|---|---|---|
| in | *index* | attribute index |

**Returns**

number of levels

Definition at line 1205 of file Dataset.cpp.

#### 6.4.3.84 unsigned int Dataset::NumNumerics ( )

Return the number of unmasked discrete attributes in the data set.

Definition at line 1224 of file Dataset.cpp.

#### 6.4.3.85 unsigned int Dataset::NumVariables ( )

Return the number of discrete plus continuous variables in the data set.

The number does not include masked variables removed.

**Returns**

number of discrete plus continuous variables

Definition at line 940 of file Dataset.cpp.

#### 6.4.3.86 void Dataset::Print ( )

Print the entire data set in compact format.

Definition at line 1359 of file Dataset.cpp.

#### 6.4.3.87 void Dataset::PrintAttributeLevelsSeen ( )

Print unique attribute levels seen.

Definition at line 1641 of file Dataset.cpp.

#### 6.4.3.88 void Dataset::PrintClassIndexInfo ( std::ostream & *outStream* = `std::cout` )

Print class index information.

Definition at line 1506 of file Dataset.cpp.

#### 6.4.3.89 void Dataset::PrintLevelCounts ( )

Prit attribute level counts.

Definition at line 1554 of file Dataset.cpp.

#### 6.4.3.90 void Dataset::PrintMaskStats ( )

Print mask statistics.

Definition at line 1904 of file Dataset.cpp.

#### 6.4.3.91 void Dataset::PrintMissingValuesStats ( )

Print missing value statistics.

Definition at line 1522 of file Dataset.cpp.

#### 6.4.3.92 void Dataset::PrintNumericsStats ( )

Print statistics about the data set including numerics.

Definition at line 1429 of file Dataset.cpp.

#### 6.4.3.93 void Dataset::PrintStats ( )

Print basic statstics about the data set - discrete/SNPs only.

Definition at line 1389 of file Dataset.cpp.

#### 6.4.3.94 void Dataset::PrintStatsSimple ( std::ostream & *outStream* = `std::cout` )

Print very simple statistics abou the data set with no formatting.

Definition at line 1471 of file Dataset.cpp.

#### 6.4.3.95 bool Dataset::ProcessExclusionFile ( std::string *exclusionFilename* )

Remove file of attribute names from consideration in analyses.

**Parameters**

| | | |
|---|---|---|
| `in` | *excusion-Filename* | filename of attributes to exclude |

**Returns**

success

Definition at line 1121 of file Dataset.cpp.

#### 6.4.3.96 void Dataset::RunSnpDiagnosticTests ( std::string *logFilename,* double *globalGenotypeThreshold* = 0.01, unsigned int *cellThreshold* = 5 )

Perform and report SNP diagnostic test information.

**Parameters**

| | | |
|---|---|---|
| in | *logFilename* | log filename |
| in | *globalGenotype-Threshold* | genotype count threshold |
| in | *cellThreshold* | x^2 cell count threshold |

open the diagnostic log file

write diagnostic log information collected in screwySnps to file

Definition at line 1913 of file Dataset.cpp.

#### 6.4.3.97 bool Dataset::SetDistanceMetrics ( std::string *newSnpMetric,* std::string *newNumMetric* = "manhattan" )

Set the the distance metrics used to compute instance-to-instance distances.

**Parameters**

| | | |
|---|---|---|
| in | *snpMetric* | name of SNP metric |
| in | *numMetric* | name of the numeric metric |

**Returns**

distance

set the SNP metric function pointer

Definition at line 2591 of file Dataset.cpp.

#### 6.4.3.98 double Dataset::SNPHWE ( int *obs_hets,* int *obs_hom1,* int *obs_hom2* )

This code implements an exact SNP test of Hardy-Weinberg Equilibrium.

As described in Wigginton, JE, Cutler, DJ, and Abecasis, GR (2005) A Note on Exact Tests of Hardy-Weinberg Equilibrium. American Journal of Human Genetics: 76. Written by Jan Wigginton.

**Parameters**

| | | |
|---|---|---|
| in | *obs_hets* | observed heterozygotes |
| in | *obs_hom1* | observed homozygotes type 1 |
| in | *obs_hom2* | homozygotes type 2 |

**Returns**

HWE value

Definition at line 2176 of file Dataset.cpp.

#### 6.4.3.99 bool Dataset::SwapAttributes ( unsigned int *a1,* unsigned int *a2* )

Swap two attributes/columns in the dataset.

**Parameters**

| | | |
|---|---|---|
| in | *a1* | attribue index 1 |
| in | *a2* | attribue index 2 |

**Returns**

success

Definition at line 931 of file Dataset.cpp.

#### 6.4.3.100 void Dataset::UpdateAllLevelCounts ( ) [protected]

Update level counts for all instances by calling UpdateLevelCounts(inst)

initialize level count maps to contain at least three levels

exclude monomorphic SNPs

Definition at line 3024 of file Dataset.cpp.

#### 6.4.3.101 void Dataset::UpdateLevelCounts ( DatasetInstance * *dsi* ) [protected]

Update all attribute level counts from one data set instance.

Updates levelCountsByClass.

**Parameters**

| | | |
|---|---|---|
| in | *dsi* | pointer to a data set instance |

Definition at line 3056 of file Dataset.cpp.

#### 6.4.3.102 void Dataset::WriteLevelCounts ( std::string *levelsFilename* )

Write attribute level counts to a text file.

**Parameters**

| | | |
|---|---|---|
| in | *levelsFilename* | filename to write levels to |

Definition at line 1569 of file Dataset.cpp.

#### 6.4.3.103 bool Dataset::WriteNewDataset ( std::string *newDatasetFilename,* OutputDatasetType *outputDatasetType* )

Write the data set to a new filename, respecting masked attributes and numerics and class/phenotype data type.

**Parameters**

| | | |
|---|---|---|
| in | *newDataset-Filename* | new data set filename |
| in | *outputDataset-Type* | type of file to write |

**Returns**

success

#### 6.4.3.104 bool Dataset::WriteNewDataset ( std::string *newDatasetFilename,* std::vector< std::string > *attributes,* OutputDatasetType *outputDatasetType* )

Write the data set to a new filename, writing only the names in the passed attributes list and also respecting masked attributes and numerics and class/phenotype data type.

**Parameters**

| | | |
|---|---|---|
| in | *newDataset-Filename* | new data set filename |
| in | *attributes* | list of attribute names to write |
| in | *outputDataset-Type* | type of file to write |

**Returns**

success

#### 6.4.3.105 bool Dataset::WriteNewPlinkPedDataset ( std::string *baseDatasetFilename* ) [protected]

Write the dataset to a new PLINK PED/MAP format, respecting masked attributes class/phenotype data type.

**Parameters**

| | | |
|---|---|---|
| in | *baseDataset-Filename* | base data set filename without extension |

**Returns**

success

Definition at line 3502 of file Dataset.cpp.

#### 6.4.3.106 bool Dataset::WriteSnpTiTvInfo ( std::string *titvFilename* )

Dump the SNP transition/transversion information to file.

Definition at line 2705 of file Dataset.cpp.

### 6.4.4 Member Data Documentation

#### 6.4.4.1 std::string Dataset::alternatePhenotypesFilename [protected]

file from which the alternate phenotypes (class labels) were read

Definition at line 690 of file Dataset.h.

#### 6.4.4.2 std::vector<std::map<char, unsigned int> > Dataset::attributeAlleleCounts [protected]

allele->count

Definition at line 664 of file Dataset.h.

#### 6.4.4.3 std::vector<std::pair<char, char> > Dataset::attributeAlleles [protected]

allele1, allele2

Definition at line 662 of file Dataset.h.

#### 6.4.4.4 std::vector<std::set<std::string> > Dataset::attributeLevelsSeen [protected]

unique attribute values/levels read from file

Definition at line 660 of file Dataset.h.

#### 6.4.4.5 std::vector<std::pair<char, double> > Dataset::attributeMinorAllele [protected]

minor allele, minor allele frequency

Definition at line 666 of file Dataset.h.

#### 6.4.4.6 std::map<std::pair<char, char>, AttributeMutationType> Dataset::attributeMutationMap [protected]

Lookup table for mutation type.

Definition at line 674 of file Dataset.h.

#### 6.4.4.7 std::vector<AttributeMutationType> Dataset::attributeMutationTypes [protected]

Keep mutation type for all attributes.

Definition at line 672 of file Dataset.h.

#### 6.4.4.8 std::vector<std::string> Dataset::attributeNames [protected]

discrete attribute names read from file

Definition at line 654 of file Dataset.h.

#### 6.4.4.9 std::map<std::string, unsigned int> Dataset::attributesMask [protected]

Definition at line 721 of file Dataset.h.

#### 6.4.4.10 std::map<std::string, unsigned int> Dataset::attributesMaskPushed [protected]

masks can be temporarily pushed and popped

Definition at line 725 of file Dataset.h.

#### 6.4.4.11 unsigned int Dataset::classColumn [protected]

class column from the original data set

Definition at line 712 of file Dataset.h.

#### 6.4.4.12 std::map<ClassLevel, std::vector<unsigned int> > Dataset::classIndexes [protected]

class values mapped to instance indices

Definition at line 714 of file Dataset.h.

#### 6.4.4.13 std::pair<NumericLevel, NumericLevel> Dataset::continuousPhenotypeMinMax [protected]

the minimum and maximum value for each continuous phenotype

Definition at line 698 of file Dataset.h.

#### 6.4.4.14 std::vector<std::map<std::string, unsigned int> > Dataset::genotypeCounts [protected]

genotype->count

Definition at line 670 of file Dataset.h.

#### 6.4.4.15 bool Dataset::hasAllelicInfo [protected]

Does this data set have alelelic information?

Definition at line 668 of file Dataset.h.

#### 6.4.4.16 bool Dataset::hasAlternatePhenotypes [protected]

does the data set contain alternate phenotypes?

Definition at line 692 of file Dataset.h.

#### 6.4.4.17 bool Dataset::hasContinuousPhenotypes [protected]

does the data set contain continuous phenotypes?

Definition at line 696 of file Dataset.h.

#### 6.4.4.18 bool Dataset::hasGenotypes [protected]

does the data set contain any genotypes?

Definition at line 652 of file Dataset.h.

#### 6.4.4.19 bool Dataset::hasNumerics [protected]

does the data set contain any continuous attributes?

Definition at line 679 of file Dataset.h.

#### 6.4.4.20 bool Dataset::hasPhenotypes [protected]

Does the data set contain phenotypes?

Definition at line 688 of file Dataset.h.

#### 6.4.4.21 std::vector<std::string> Dataset::instanceIds [protected]

IDs associated with the instances read from file.

Definition at line 703 of file Dataset.h.

#### 6.4.4.22 std::vector<std::string> Dataset::instanceIdsToLoad [protected]

IDs of instances to load from numeric and/or phenotype files.

Definition at line 705 of file Dataset.h.

#### 6.4.4.23 std::vector<DatasetInstance*> Dataset::instances [protected]

vector of pointers to all instances in the data set

Definition at line 701 of file Dataset.h.

#### 6.4.4.24 std::map<std::string, unsigned int> Dataset::instancesMask [protected]

Definition at line 723 of file Dataset.h.

#### 6.4.4.25 std::map<std::string, unsigned int> Dataset::instancesMaskPushed [protected]

Definition at line 727 of file Dataset.h.

#### 6.4.4.26 std::vector<std::map<AttributeLevel, unsigned int> > Dataset::levelCounts [protected]

attribute values/levels counts

Definition at line 656 of file Dataset.h.

#### 6.4.4.27 std::vector<std::map<std::pair<AttributeLevel, ClassLevel>, unsigned int> > Dataset::levelCountsByClass [protected]

attribute values/levels counts by discrete class

Definition at line 658 of file Dataset.h.

#### 6.4.4.28 bool Dataset::maskIsPushed [protected]

Definition at line 728 of file Dataset.h.

#### 6.4.4.29 std::map<std::string, std::vector<unsigned int> > Dataset::missingNumericValues [protected]

missing continuous values and their instance indices

Definition at line 709 of file Dataset.h.

#### 6.4.4.30 std::map<std::string, std::vector<unsigned int> > Dataset::missingValues [protected]

missing discrete values and their instance indices

Definition at line 707 of file Dataset.h.

#### 6.4.4.31 double(* Dataset::numDiff)(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2) [protected]

Compute the continuous difference in an attribute between two instances.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into vector of all attributes |
| in | *dsi1* | pointer to DatasetInstance 1 |
| in | *dsi2* | pointer to DatasetInstance 2 |

**Returns**

diff(erence)

Definition at line 640 of file Dataset.h.

#### 6.4.4.32 std::string Dataset::numericsFilename [protected]

file from which the continuous attributes were read

Definition at line 677 of file Dataset.h.

#### 6.4.4.33 std::vector<std::string> Dataset::numericsIds [protected]

IDs associated with the numerics read from file.

Definition at line 681 of file Dataset.h.

#### 6.4.4.34 std::map<std::string, unsigned int> Dataset::numericsMask [protected]

Definition at line 722 of file Dataset.h.

#### 6.4.4.35 std::map<std::string, unsigned int> Dataset::numericsMaskPushed [protected]

Definition at line 726 of file Dataset.h.

#### 6.4.4.36 std::vector< std::pair<NumericLevel, NumericLevel> > Dataset::numericsMinMax [protected]

the minimum and maximum value for each continuous attribute

Definition at line 683 of file Dataset.h.

#### 6.4.4.37 std::vector<std::string> Dataset::numericsNames [protected]

continuous attribute names read from file

Definition at line 685 of file Dataset.h.

#### 6.4.4.38 std::string Dataset::numMetric [protected]

the name of continuous diff(erence) function

Definition at line 647 of file Dataset.h.

#### 6.4.4.39 std::vector<std::string> Dataset::phenotypesIds [protected]

IDs associated with the phenotypes/classes read from file.

Definition at line 694 of file Dataset.h.

#### 6.4.4.40 GSLRandomFlat* Dataset::rng [protected]

random number generator classes use GNU Scienitifc Library (GSL)

Definition at line 731 of file Dataset.h.

#### 6.4.4.41 double(* Dataset::snpDiff)(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2) [protected]

Compute the discrete difference in an attribute between two instances for determining nearest neighbors.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into vector of all attributes |
| in | *dsi1* | pointer to DatasetInstance 1 |
| in | *dsi2* | pointer to DatasetInstance 2 |

**Returns**

diff(erence)

Definition at line 630 of file Dataset.h.

#### 6.4.4.42 std::string Dataset::snpMetric [protected]

the name of discrete diff(erence) function

Definition at line 645 of file Dataset.h.

#### 6.4.4.43 std::string Dataset::snpsFilename [protected]

file from which the discrete attributes (SNPSs) were read

Definition at line 650 of file Dataset.h.

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

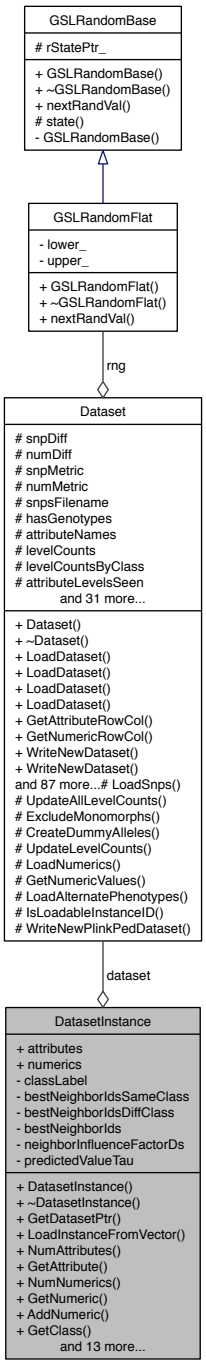
- src/library/Dataset.h
- src/library/Dataset.cpp

## 6.5 DatasetInstance Class Reference

Class to hold dataset instances (rows of attributes).

`#include <DatasetInstance.h>`

Collaboration diagram for DatasetInstance:




## Public Member Functions

- DatasetInstance (Dataset *ds)

  *Construct an data set instance object.*
- ~DatasetInstance ()
- Dataset * GetDatasetPtr ()

  *return the Dataset pointer associated with this instance*

- bool LoadInstanceFromVector (std::vector< AttributeLevel > newAttributes)

  *Load this instance with the attributes and class value from the newAttributes vector.*
- unsigned int NumAttributes ()

  *return the number of discrete attributes*
- AttributeLevel GetAttribute (unsigned int index)

  *Get and return an attribute value at index.*
- unsigned int NumNumerics ()

  *return the number of continuous attributes*
- NumericLevel GetNumeric (unsigned int index)

  *Get and return numeric value at index.*
- bool AddNumeric (NumericLevel newNum)

  *Add a numeric value to the instance's numerics vector.*
- ClassLevel GetClass ()

  *Get the discrete class value.*
- void SetClass (ClassLevel classValue)

  *Set the discrete class value.*
- double GetPredictedValueTau ()

  *Get the continuous class value.*
- void SetPredictedValueTau (double newValue)

  *Set the continuous class value.*
- double GetInfluenceFactorD (unsigned int neighborIndex)

  *Get the nearest neighbor value at neighborIndex.*
- void ClearInfluenceFactors ()

  *Clear all nearest neighbor values.*
- bool AddInfluenceFactorD (double factor)

  *Add the next nearest neighbor influence factor.*
- void Print ()

  *Print the attributes, numerics and class name of this instance to stdout.*
- bool SwapAttributes (unsigned int a1, unsigned int a2)

  *Swap attribute/column values in this instance.*
- void SetDistanceSums (unsigned int kNearestNeighbors, DistancePairs &sameClassSums, std::map< ClassLevel, DistancePairs > &diffClassSums)

  *Set the best kNearestNeighbors from the same and different classes SIDE_EFFECT: Sorts and loads class the vairables: sameSums snd diffSums from the neighbors.*
- void SetDistanceSums (unsigned int kNearestNeighbors, DistancePairs instancesSums)

  *Set the best kNearestNeighbors from all other instances/neighbors.*
- void PrintDistancePairs (const DistancePairs &distPairs)

  *Prints passed distance pairs.*
- bool GetNNearestInstances (unsigned int n, std::vector< unsigned int > &sameClassInstances, std::vector< unsigned int > &diffClassInstances)

  *Returns N closest instances using the sameSums and diffSums class variables.*
- bool GetNNearestInstances (unsigned int n, std::vector< unsigned int > &sameClassInstances, std::map< ClassLevel, std::vector< unsigned int > > &diffClassInstances)

  *Returns N closest instances using the sameSums and diffSums class variables.*
- bool GetNNearestInstances (unsigned int n, std::vector< unsigned int > &closestInstances)

  *Returns N closest instances to this instance.*

## Public Attributes

- std::vector< AttributeLevel > attributes

  *discrete attributes*
- std::vector< NumericLevel > numerics

  *continuous attributes*

### Private Attributes

- Dataset * dataset

  *pointer to a Dataset object*
- ClassLevel classLabel

  *the class value for this instance*
- std::vector< std::string > bestNeighborIdsSameClass

  *vector of instance IDs for the best neighbors in this instance's class*
- std::map< ClassLevel, std::vector< std::string > > bestNeighborIdsDiffClass

  *vector of instance IDs for the best neighbors of different class(es)*
- std::vector< std::string > bestNeighborIds

  *best neighbor IDs for continuous class*
- std::vector< double > neighborInfluenceFactorDs

  *nearest neighbor weighting factors*
- double predictedValueTau

  *countinuous value for this class*

## 6.5.1 Detailed Description

Class to hold dataset instances (rows of attributes).

Reworked entirely for McKinney Lab work - 2/28/11

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 6/14/05

Definition at line 28 of file DatasetInstance.h.

## 6.5.2 Constructor & Destructor Documentation

### 6.5.2.1 DatasetInstance::DatasetInstance ( Dataset * *ds* )

Construct an data set instance object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |

Definition at line 33 of file DatasetInstance.cpp.

### 6.5.2.2 DatasetInstance::~DatasetInstance ( )

Definition at line 39 of file DatasetInstance.cpp.

## 6.5.3 Member Function Documentation

#### 6.5.3.1 bool DatasetInstance::AddInfluenceFactorD ( double *factor* )

Add the next nearest neighbor influence factor.

Definition at line 129 of file DatasetInstance.cpp.

#### 6.5.3.2 bool DatasetInstance::AddNumeric ( NumericLevel *newNum* )

Add a numeric value to the instance's numerics vector.

**Parameters**

| | | |
|---|---|---|
| in | *newNum* | new numeric value |

**Returns**

success

Definition at line 99 of file DatasetInstance.cpp.

#### 6.5.3.3 void DatasetInstance::ClearInfluenceFactors ( )

Clear all nearest neighbor values.

Definition at line 125 of file DatasetInstance.cpp.

#### 6.5.3.4 AttributeLevel DatasetInstance::GetAttribute ( unsigned int *index* )

Get and return an attribute value at index.

**Parameters**

| | | |
|---|---|---|
| in | *index* | attribute index |

**Returns**

attribute value at index

Definition at line 64 of file DatasetInstance.cpp.

#### 6.5.3.5 ClassLevel DatasetInstance::GetClass ( )

Get the discrete class value.

Definition at line 105 of file DatasetInstance.cpp.

#### 6.5.3.6 Dataset * DatasetInstance::GetDatasetPtr ( )

return the Dataset pointer associated with this instance

Definition at line 42 of file DatasetInstance.cpp.

#### 6.5.3.7 double DatasetInstance::GetInfluenceFactorD ( unsigned int *neighborIndex* )

Get the nearest neighbor value at neighborIndex.

Definition at line 121 of file DatasetInstance.cpp.

#### 6.5.3.8 bool DatasetInstance::GetNNearestInstances ( unsigned int *n*, std::vector< unsigned int > & *sameClassInstances*, std::vector< unsigned int > & *diffClassInstances* )

Returns N closest instances using the sameSums and diffSums class variables.

**Parameters**

| | | |
|---|---|---|
| in | *n* | n nearest nerighbors |
| in | *sameCLass-Instances* | vector of same class instances indices |
| in | *diffClass-Instances* | vector of different class instance indices |

**Returns**

success

#### 6.5.3.9 bool DatasetInstance::GetNNearestInstances ( unsigned int *n*, std::vector< unsigned int > & *sameClassInstances*, std::map< ClassLevel, std::vector< unsigned int > > & *diffClassInstances* )

Returns N closest instances using the sameSums and diffSums class variables.

**Parameters**

| | | |
|---|---|---|
| in | *n* | n nearest nerighbors |
| in | *sameCLass-Instances* | vector of same class instances indices |
| in | *diffClass-Instances* | vector of different classes instance indices |

**Returns**

success

#### 6.5.3.10 bool DatasetInstance::GetNNearestInstances ( unsigned int *n*, std::vector< unsigned int > & *closestInstances* )

Returns N closest instances to this instance.

**Parameters**

| | | |
|---|---|---|
| in | *n* | n nearest neighbors |
| in | *closestInstances* | reference to a vector of instance indices |

**Returns**

success

#### 6.5.3.11 double DatasetInstance::GetNumeric ( unsigned int *index* )

Get and return numeric value at index.

**Parameters**

| | | |
|---|---|---|
| in | *index* | numeric index |

**Returns**

numeric value at index

Definition at line 84 of file DatasetInstance.cpp.

#### 6.5.3.12 double DatasetInstance::GetPredictedValueTau ( )

Get the continuous class value.

Definition at line 113 of file DatasetInstance.cpp.

#### 6.5.3.13 bool DatasetInstance::LoadInstanceFromVector ( std::vector< AttributeLevel > *newAttributes* )

Load this instance with the attributes and class value from the newAttributes vector.

**Parameters**

| | | |
|---|---|---|
| in | *newAttributes* | vector of new attribute values |

**Returns**

success

Definition at line 47 of file DatasetInstance.cpp.

#### 6.5.3.14 unsigned int DatasetInstance::NumAttributes ( )

return the number of discrete attributes

Definition at line 60 of file DatasetInstance.cpp.

#### 6.5.3.15 unsigned int DatasetInstance::NumNumerics ( )

return the number of continuous attributes

Definition at line 80 of file DatasetInstance.cpp.

#### 6.5.3.16 void DatasetInstance::Print ( )

Print the attributes, numerics and class name of this instance to stdout.

Definition at line 134 of file DatasetInstance.cpp.

#### 6.5.3.17 void DatasetInstance::PrintDistancePairs ( const DistancePairs & *distPairs* )

Prints passed distance pairs.

**Parameters**

| | | |
|---|---|---|
| in | *distPairs* | distance pairs |

Definition at line 240 of file DatasetInstance.cpp.

#### 6.5.3.18 void DatasetInstance::SetClass ( ClassLevel *classValue* )

Set the discrete class value.

Definition at line 109 of file DatasetInstance.cpp.

#### 6.5.3.19 void DatasetInstance::SetDistanceSums ( unsigned int *kNearestNeighbors,* DistancePairs & *sameClassSums,* std::map< ClassLevel, DistancePairs > & *diffClassSums* )

Set the best kNearestNeighbors from the same and different classes SIDE_EFFECT: Sorts and loads class the vairables: sameSums snd diffSums from the neighbors.

**Parameters**

| | | |
|---|---|---|
| in | *kNearest-Neighbors* | k nearest nerighbors, |
| in | *sameCLass-Sums* | vectors of pairs <instance, sum> of same class |
| in | *diffClassSums* | vectors of pairs <instance, sum> of other classes |

**Returns**

nothing

#### 6.5.3.20 void DatasetInstance::SetDistanceSums ( unsigned int *kNearestNeighbors,* DistancePairs *instancesSums* )

Set the best kNearestNeighbors from all other instances/neighbors.

SIDE_EFFECT: Sorts and loads neighborSums from the instanceSums

**Parameters**

| | | |
|---|---|---|
| in | *kNearest-Neighbors* | k nearest neighbors |
| in | *instanceSums* | vectors of k pairs <instance, sum> for neighbors |

**Returns**

nothing

Definition at line 215 of file DatasetInstance.cpp.

#### 6.5.3.21 void DatasetInstance::SetPredictedValueTau ( double *newValue* )

Set the continuous class value.

Definition at line 117 of file DatasetInstance.cpp.

#### 6.5.3.22 bool DatasetInstance::SwapAttributes ( unsigned int *a1,* unsigned int *a2* )

Swap attribute/column values in this instance.

**Parameters**

| | | |
|---|---|---|
| in | *a1* | attribue index 1 |
| in | *a2* | attribue index 2 |

**Returns**

bool success

Definition at line 154 of file DatasetInstance.cpp.

### 6.5.4 Member Data Documentation

#### 6.5.4.1 std::vector<AttributeLevel> DatasetInstance::attributes

discrete attributes

Definition at line 147 of file DatasetInstance.h.

#### 6.5.4.2 std::vector<std::string> DatasetInstance::bestNeighborIds [private]

best neighbor IDs for continuous class

Definition at line 160 of file DatasetInstance.h.

#### 6.5.4.3 std::map<ClassLevel, std::vector<std::string> > DatasetInstance::bestNeighborIdsDiffClass [private]

vector of instance IDs for the best neighbors of different class(es)

Definition at line 158 of file DatasetInstance.h.

#### 6.5.4.4 std::vector<std::string> DatasetInstance::bestNeighborIdsSameClass [private]

vector of instance IDs for the best neighbors in this instance's class

Definition at line 156 of file DatasetInstance.h.

#### 6.5.4.5 ClassLevel DatasetInstance::classLabel [private]

the class value for this instance

Definition at line 154 of file DatasetInstance.h.

#### 6.5.4.6 Dataset* DatasetInstance::dataset [private]

pointer to a Dataset object

Definition at line 152 of file DatasetInstance.h.

#### 6.5.4.7 std::vector<double> DatasetInstance::neighborInfluenceFactorDs [private]

nearest neighbor weighting factors

Definition at line 162 of file DatasetInstance.h.

#### 6.5.4.8 std::vector<NumericLevel> DatasetInstance::numerics

continuous attributes

Definition at line 149 of file DatasetInstance.h.

#### 6.5.4.9 double DatasetInstance::predictedValueTau [private]

countinuous value for this class

Definition at line 164 of file DatasetInstance.h.

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

- src/library/DatasetInstance.h
- src/library/DatasetInstance.cpp

## 6.6 deref_less Class Reference

### Public Member Functions

- bool operator() (const T a, const T b) const

### 6.6.1 Detailed Description

Definition at line 59 of file ReliefF.cpp.

### 6.6.2 Member Function Documentation

#### 6.6.2.1 bool deref_less::operator() ( const T *a*, const T *b* ) const [inline]

Definition at line 63 of file ReliefF.cpp.

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

- src/library/ReliefF.cpp

## 6.7 deref_less_bcw Class Reference

### Public Member Functions

- bool operator() (const T a, const T b) const

### 6.7.1 Detailed Description

Definition at line 24 of file DatasetInstance.cpp.

### 6.7.2 Member Function Documentation

#### 6.7.2.1 bool deref_less_bcw::operator() ( const T *a*, const T *b* ) const [inline]

Definition at line 28 of file DatasetInstance.cpp.

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

- src/library/DatasetInstance.cpp

## 6.8 DgeData Class Reference

Digital gene expression data.

`#include <DgeData.h>`

### Public Member Functions

- DgeData ()
- virtual ~DgeData ()
- bool LoadData (std::string countsFile, std::string normsFile="")

  *Create a new set of DGE data with a counts file and a phenotype file.*
- std::vector< std::string > GetSampleNames ()

  *Get the sample names/IDs.*
- std::vector< std::string > GetGeneNames ()

  *Get the gene names/IDs.*
- std::pair< double, double > GetGeneMinMax (int geneIndex)

  *Get the min and max values for gene at index.*
- int GetNumSamples ()

  *Get the number of samples.*
- int GetNumGenes ()

  *Get the number of genes.*
- std::vector< double > GetSampleCounts (int sampleIndex)

  *Get sample counts for sample at index.*
- int GetSamplePhenotype (int sampleIndex)

  *Get the phenotype at sample index.*
- std::vector< double > GetNormalizationFactors ()

  *Get the normalization factors.*
- void PrintSampleStats ()

  *Print the Sample statistics to the console.*

### Private Attributes

- std::string countsFilename

  *Filename containing DGE counts.*
- std::string phenosFilename

  *Filename containing DGE phenotypes.*
- std::string normsFilename

  *Filename containing DGE normalization factors.*
- bool hasNormFactors

  *Are we using normalization?*
- std::vector< double > normFactors

  *Vector of (optional) normalization factors for each sample.*
- std::vector< std::string > geneNames

  *Gene names.*
- std::vector< std::vector < double > > counts

  *Digital gene expression counts.*
- std::vector< std::string > sampleNames

  *Sample names.*
- std::vector< int > phenotypes

*Sample phenotypes.*

- std::vector< std::pair< double, double > > minMaxGeneCounts

  *Min and max count for genes.*

- std::vector< std::pair< double, double > > minMaxSampleCounts

  *Min and max values for samples.*

- std::vector< std::vector< int > > sampleZeroes

  *Zero count sample indices.*

### 6.8.1 Detailed Description

Digital gene expression data.

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 1/18/12

Definition at line 16 of file DgeData.h.

### 6.8.2 Constructor & Destructor Documentation

#### 6.8.2.1 DgeData::DgeData ( )

Definition at line 23 of file DgeData.cpp.

#### 6.8.2.2 DgeData::~DgeData ( ) [virtual]

Definition at line 27 of file DgeData.cpp.

### 6.8.3 Member Function Documentation

#### 6.8.3.1 pair< double, double > DgeData::GetGeneMinMax ( int *geneIndex* )

Get the min and max values for gene at index.

Definition at line 221 of file DgeData.cpp.

#### 6.8.3.2 vector< string > DgeData::GetGeneNames ( )

Get the gene names/IDs.

Definition at line 217 of file DgeData.cpp.

#### 6.8.3.3 vector< double > DgeData::GetNormalizationFactors ( )

Get the normalization factors.

Definition at line 265 of file DgeData.cpp.

#### 6.8.3.4 int DgeData::GetNumGenes ( )

Get the number of genes.

Definition at line 236 of file DgeData.cpp.

#### 6.8.3.5 int DgeData::GetNumSamples ( )

Get the number of samples.

Definition at line 232 of file DgeData.cpp.

#### 6.8.3.6 vector< double > DgeData::GetSampleCounts ( int *sampleIndex* )

Get sample counts for sample at index.

Definition at line 240 of file DgeData.cpp.

#### 6.8.3.7 vector< string > DgeData::GetSampleNames ( )

Get the sample names/IDs.

Definition at line 213 of file DgeData.cpp.

#### 6.8.3.8 int DgeData::GetSamplePhenotype ( int *sampleIndex* )

Get the phenotype at sample index.

Definition at line 255 of file DgeData.cpp.

#### 6.8.3.9 bool DgeData::LoadData ( std::string *countsFile,* std::string *normsFile* = " " )

Create a new set of DGE data with a counts file and a phenotype file.

read gene counts

load all counts for this gene as doubles

save this gene's counts to the counts class member variable

get min and max sample counts, and sample zeroes

Definition at line 30 of file DgeData.cpp.

#### 6.8.3.10 void DgeData::PrintSampleStats ( )

Print the Sample statistics to the console.

Definition at line 269 of file DgeData.cpp.

### 6.8.4 Member Data Documentation

#### 6.8.4.1 std::vector<std::vector<double> > DgeData::counts [private]

Digital gene expression counts.

Definition at line 54 of file DgeData.h.

#### 6.8.4.2 std::string DgeData::countsFilename [private]

Filename containing DGE counts.

Definition at line 42 of file DgeData.h.

#### 6.8.4.3 std::vector<std::string> DgeData::geneNames [private]

Gene names.

Definition at line 52 of file DgeData.h.

#### 6.8.4.4 bool DgeData::hasNormFactors [private]

Are we using normalization?

Definition at line 48 of file DgeData.h.

#### 6.8.4.5 std::vector<std::pair<double, double> > DgeData::minMaxGeneCounts [private]

Min and max count for genes.

Definition at line 60 of file DgeData.h.

#### 6.8.4.6 std::vector<std::pair<double, double> > DgeData::minMaxSampleCounts [private]

Min and max values for samples.

Definition at line 62 of file DgeData.h.

#### 6.8.4.7 std::vector<double> DgeData::normFactors [private]

Vector of (optional) normalization factors for each sample.

Definition at line 50 of file DgeData.h.

#### 6.8.4.8 std::string DgeData::normsFilename [private]

Filename containing DGE normalization factors.

Definition at line 46 of file DgeData.h.

#### 6.8.4.9 std::string DgeData::phenosFilename [private]

Filename containing DGE phenotypes.

Definition at line 44 of file DgeData.h.

#### 6.8.4.10 std::vector<int> DgeData::phenotypes [private]

Sample phenotypes.

Definition at line 58 of file DgeData.h.

#### 6.8.4.11 std::vector<std::string> DgeData::sampleNames [private]

Sample names.

Definition at line 56 of file DgeData.h.

#### 6.8.4.12 std::vector<std::vector<int> > DgeData::sampleZeroes [private]

Zero count sample indices.

Definition at line 64 of file DgeData.h.

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

- src/library/DgeData.h
- src/library/DgeData.cpp

## 6.9 insilico::do_to_lower< charT > Class Template Reference

`#include <StringUtils.h>`

### Public Member Functions

- do_to_lower (std::ctype< charT > &ct)
- do_to_lower (const std::locale &loc=std::locale())
- charT operator() (charT c) const

### Private Attributes

- std::ctype< charT > const & m_ctype

### 6.9.1 Detailed Description

template<class charT = char>class insilico::do_to_lower< charT >

Definition at line 79 of file StringUtils.h.

### 6.9.2 Constructor & Destructor Documentation

#### 6.9.2.1 template<class charT = char> insilico::do_to_lower< charT >::do_to_lower ( std::ctype< charT > & *ct* ) [inline]

Definition at line 83 of file StringUtils.h.

#### 6.9.2.2 template<class charT = char> insilico::do_to_lower< charT >::do_to_lower ( const std::locale & *loc* = std::locale() ) [inline]

Definition at line 86 of file StringUtils.h.

### 6.9.3 Member Function Documentation

#### 6.9.3.1 template<class charT = char> charT insilico::do_to_lower< charT >::operator() ( charT *c* ) const [inline]

Definition at line 89 of file StringUtils.h.

### 6.9.4 Member Data Documentation

#### 6.9.4.1 template<class charT = char> std::ctype<charT> const& insilico::do_to_lower< charT >::m_ctype [private]

Definition at line 93 of file StringUtils.h.

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

- src/library/StringUtils.h

## 6.10 insilico::do_to_upper< charT > Class Template Reference

`#include <StringUtils.h>`

### Public Member Functions

- do_to_upper (std::ctype< charT > &ct)
- do_to_upper (const std::locale &loc=std::locale())
- charT operator() (charT c) const

### Private Attributes

- std::ctype< charT > const & m_ctype

### 6.10.1 Detailed Description

template<class charT = char>class insilico::do_to_upper< charT >

Definition at line 59 of file StringUtils.h.

### 6.10.2 Constructor & Destructor Documentation

#### 6.10.2.1 template<class charT = char> insilico::do_to_upper< charT >::do_to_upper ( std::ctype< charT > & *ct* ) [inline]

Definition at line 63 of file StringUtils.h.

#### 6.10.2.2 template<class charT = char> insilico::do_to_upper< charT >::do_to_upper ( const std::locale & *loc* = std::locale() ) [inline]

Definition at line 66 of file StringUtils.h.

### 6.10.3 Member Function Documentation

#### 6.10.3.1 template<class charT = char> charT insilico::do_to_upper< charT >::operator() ( charT *c* ) const [inline]

Definition at line 69 of file StringUtils.h.

### 6.10.4 Member Data Documentation

#### 6.10.4.1 template<class charT = char> std::ctype<charT> const& insilico::do_to_upper< charT >::m_ctype [private]

Definition at line 73 of file StringUtils.h.

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

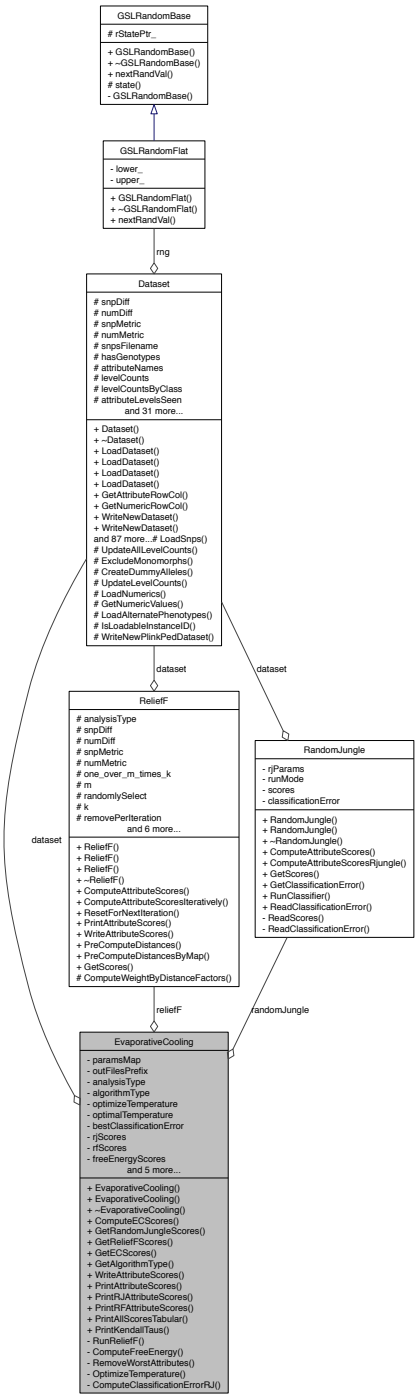
- src/library/StringUtils.h

## 6.11 EvaporativeCooling Class Reference

Evaporative Cooling attribute ranking algorithm.

`#include <EvaporativeCooling.h>`

Collaboration diagram for EvaporativeCooling:




## Public Member Functions

- EvaporativeCooling (Dataset *ds, po::variables_map &vm, AnalysisType anaType=SNP_ONLY_ANALYSIS)

  *Construct an EC algorithm object.*

- EvaporativeCooling (Dataset *ds, ConfigMap &configMap, AnalysisType anaType=SNP_ONLY_ANALYSIS)

*Construct an EC algorithm object.*

- virtual ~EvaporativeCooling ()
- bool ComputeECScores ()

  *Compute the EC scores based on the current set of attributes.*
- EcScores & GetRandomJungleScores ()

  *Get the last computed RandomJungle scores.*
- EcScores & GetReliefFScores ()

  *Get the last computed ReliefF scores.*
- EcScores & GetECScores ()

  *Get the last computed EC scores.*
- EcAlgorithmType GetAlgorithmType ()

  *Return the algorithm type: EC_ALL, EC_RJ or EC_RF.*
- void WriteAttributeScores (std::string baseFilename)

  *Write the scores and attribute names to file.*
- void PrintAttributeScores (std::ofstream &outStream)

  *Write the EC scores and attribute names to stream.*
- void PrintRJAttributeScores (std::ofstream &outStream)

  *Write the RJ scores and attribute names to stream.*
- void PrintRFAttributeScores (std::ofstream &outStream)

  *Write the RF scores and attribute names to stream.*
- bool PrintAllScoresTabular ()

  *Print the current attributes scores to stdout in tab-delimited format.*
- bool PrintKendallTaus ()

  *Print the kendall taus between the ReliefF and RandomJungle scores.*

## Private Member Functions

- bool RunReliefF ()

  *Run the ReliefF algorithm.*
- bool ComputeFreeEnergy (double temperature)

  *Compute the attributes' free energy using the couple temperature.*
- bool RemoveWorstAttributes (unsigned int numToRemove=1)

  *Remove the worst attribute based on free energy scores.*
- double OptimizeTemperature (std::vector< double > deltas)

  *optimize the temperature coupling constant*
- double ComputeClassificationErrorRJ ()

  *use Random Jungle to compute the classification error of the current set of attributes with numToRemovePerIteration attributes removed*

## Private Attributes

- Dataset * dataset

  *pointer to a Dataset object*
- po::variables_map paramsMap

  *command line parameters map*
- std::string outFilesPrefix

  *prefix for all output files*
- AnalysisType analysisType

  *type of analysis to perform*
- EcAlgorithmType algorithmType

*algorithm steps to perform*

- ReliefF * reliefF

  *pointer to a ReliefF or RReliefF algorithm object*

- RandomJungle * randomJungle

  *pointer to a RandomJungle algorithm object*

- bool optimizeTemperature
- double optimalTemperature
- double bestClassificationError
- EcScores rjScores

  *current random jungle scores*

- EcScores rfScores

  *current relieff scores*

- EcScores freeEnergyScores

  *current free energy scores*

- unsigned int numRFThreads
- unsigned int numToRemovePerIteration

  *number of attributes to remove per iteration*

- unsigned int numToRemoveNextIteration

  *number of attributes to remove next iteration*

- unsigned int numTargetAttributes

  *number of target attributes*

- EcScores evaporatedAttributes

  *attributes that have been evaporated so far*

- EcScores ecScores

  *current set of ec scores*

### 6.11.1 Detailed Description

Evaporative Cooling attribute ranking algorithm.

Implements the Evaporative Cooling algorithm in: McKinney, et. al. "Capturing the Spectrum of Interaction Effects in Genetic Association Studies by Simulated Evaporative Cooling Network Analysis." PLoS Genetics, Vol 5, Issue 3, 2009.

**See also**

ReliefF
RReliefF
RandomJungle

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 7/14/11

Definition at line 53 of file EvaporativeCooling.h.

### 6.11.2 Constructor & Destructor Documentation

#### 6.11.2.1 EvaporativeCooling::EvaporativeCooling ( Dataset * *ds*, po::variables_map & *vm*, AnalysisType *anaType* = SNP_ONLY_ANALYSIS )

Construct an EC algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *vm* | reference to a Boost map of command line options |
| in | *anaType* | analysis type |

Definition at line 54 of file EvaporativeCooling.cpp.

#### 6.11.2.2 EvaporativeCooling::EvaporativeCooling ( Dataset * *ds*, ConfigMap & *configMap*, AnalysisType *anaType* = SNP_ONLY_ANALYSIS )

Construct an EC algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *configMap* | reference to a ConfigMap (map<string, string>) |
| in | *anaType* | analysis type |

Definition at line 157 of file EvaporativeCooling.cpp.

#### 6.11.2.3 EvaporativeCooling::~EvaporativeCooling ( ) [virtual]

Definition at line 270 of file EvaporativeCooling.cpp.

### 6.11.3 Member Function Documentation

#### 6.11.3.1 double EvaporativeCooling::ComputeClassificationErrorRJ ( ) [private]

use Random Jungle to compute the classification error of the current set of attributes with numToRemovePerIteration attributes removed

get the best attribute names based on free energy score

write new data set with worst attributes removed

create a configuration map for RandomJungle constructor

run Random Jungle classifier and read classification error

remove the temporary file

return the classification error on this data

Definition at line 795 of file EvaporativeCooling.cpp.

#### 6.11.3.2 bool EvaporativeCooling::ComputeECScores ( )

Compute the EC scores based on the current set of attributes.

Definition at line 279 of file EvaporativeCooling.cpp.

#### 6.11.3.3 bool EvaporativeCooling::ComputeFreeEnergy ( double *temperature* ) [private]

Compute the attributes' free energy using the couple temperature.

**Parameters**

| | | |
|---|---|---|
| in | *tempreatire* | coupling temperature T |

**Returns**

distance

Definition at line 685 of file EvaporativeCooling.cpp.

#### 6.11.3.4 EcAlgorithmType EvaporativeCooling::GetAlgorithmType ( )

Return the algorithm type: EC_ALL, EC_RJ or EC_RF.

Definition at line 459 of file EvaporativeCooling.cpp.

#### 6.11.3.5 EcScores & EvaporativeCooling::GetECScores ( )

Get the last computed EC scores.

Definition at line 455 of file EvaporativeCooling.cpp.

#### 6.11.3.6 EcScores & EvaporativeCooling::GetRandomJungleScores ( )

Get the last computed RandomJungle scores.

Definition at line 447 of file EvaporativeCooling.cpp.

#### 6.11.3.7 EcScores & EvaporativeCooling::GetReliefFScores ( )

Get the last computed ReliefF scores.

Definition at line 451 of file EvaporativeCooling.cpp.

#### 6.11.3.8 double EvaporativeCooling::OptimizeTemperature ( std::vector< double > *deltas* ) [private]

optimize the temperature coupling constant

for each delta, run a classifier on the best attributes according to the free energy and update best temperature

if classification error is lower at this delta, update best temperature and best classification error

Definition at line 760 of file EvaporativeCooling.cpp.

#### 6.11.3.9 bool EvaporativeCooling::PrintAllScoresTabular ( )

Print the current attributes scores to stdout in tab-delimited format.

Definition at line 558 of file EvaporativeCooling.cpp.

#### 6.11.3.10 void EvaporativeCooling::PrintAttributeScores ( std::ofstream & *outStream* )

Write the EC scores and attribute names to stream.

**Parameters**

| | | |
|---|---|---|
| in | *outStream* | stream to write score-attribute name pairs |

Definition at line 463 of file EvaporativeCooling.cpp.

#### 6.11.3.11 bool EvaporativeCooling::PrintKendallTaus ( )

Print the kendall taus between the ReliefF and RandomJungle scores.

Definition at line 592 of file EvaporativeCooling.cpp.

#### 6.11.3.12 void EvaporativeCooling::PrintRFAttributeScores ( std::ofstream & *outStream* )

Write the RF scores and attribute names to stream.

**Parameters**

| | | |
|---|---|---|
| in | *outStream* | stream to write score-attribute name pairs |

Definition at line 480 of file EvaporativeCooling.cpp.

#### 6.11.3.13 void EvaporativeCooling::PrintRJAttributeScores ( std::ofstream & *outStream* )

Write the RJ scores and attribute names to stream.

**Parameters**

| | | |
|---|---|---|
| in | *outStream* | stream to write score-attribute name pairs |

Definition at line 471 of file EvaporativeCooling.cpp.

#### 6.11.3.14 bool EvaporativeCooling::RemoveWorstAttributes ( unsigned int *numToRemove* = `1` ) `[private]`

Remove the worst attribute based on free energy scores.

**Parameters**

| | | |
|---|---|---|
| in | *numToRemove* | number of attributes to remove/evaporate |

**Returns**

distance

Definition at line 728 of file EvaporativeCooling.cpp.

#### 6.11.3.15 bool EvaporativeCooling::RunReliefF ( ) `[private]`

Run the ReliefF algorithm.

Definition at line 634 of file EvaporativeCooling.cpp.

#### 6.11.3.16 void EvaporativeCooling::WriteAttributeScores ( std::string *baseFilename* )

Write the scores and attribute names to file.

**Parameters**

| | | |
|---|---|---|
| in | *baseFilename* | filename to write score-attribute name pairs |

Definition at line 489 of file EvaporativeCooling.cpp.

### 6.11.4 Member Data Documentation

#### 6.11.4.1 EcAlgorithmType EvaporativeCooling::algorithmType [private]

algorithm steps to perform

Definition at line 138 of file EvaporativeCooling.h.

#### 6.11.4.2 AnalysisType EvaporativeCooling::analysisType [private]

type of analysis to perform

**See also**

ReliefF

Definition at line 136 of file EvaporativeCooling.h.

#### 6.11.4.3 double EvaporativeCooling::bestClassificationError [private]

Definition at line 147 of file EvaporativeCooling.h.

#### 6.11.4.4 Dataset* EvaporativeCooling::dataset [private]

pointer to a Dataset object

Definition at line 128 of file EvaporativeCooling.h.

#### 6.11.4.5 EcScores EvaporativeCooling::ecScores [private]

current set of ec scores

Definition at line 168 of file EvaporativeCooling.h.

#### 6.11.4.6 EcScores EvaporativeCooling::evaporatedAttributes [private]

attributes that have been evaporated so far

Definition at line 166 of file EvaporativeCooling.h.

#### 6.11.4.7 EcScores EvaporativeCooling::freeEnergyScores [private]

current free energy scores

Definition at line 154 of file EvaporativeCooling.h.

#### 6.11.4.8 unsigned int EvaporativeCooling::numRFThreads [private]

Definition at line 157 of file EvaporativeCooling.h.

#### 6.11.4.9 unsigned int EvaporativeCooling::numTargetAttributes [private]

number of target attributes

Definition at line 164 of file EvaporativeCooling.h.

#### 6.11.4.10 unsigned int EvaporativeCooling::numToRemoveNextIteration [private]

number of attributes to remove next iteration

Definition at line 161 of file EvaporativeCooling.h.

#### 6.11.4.11 unsigned int EvaporativeCooling::numToRemovePerIteration [private]

number of attributes to remove per iteration

Definition at line 159 of file EvaporativeCooling.h.

#### 6.11.4.12 double EvaporativeCooling::optimalTemperature [private]

Definition at line 146 of file EvaporativeCooling.h.

#### 6.11.4.13 bool EvaporativeCooling::optimizeTemperature [private]

Definition at line 145 of file EvaporativeCooling.h.

#### 6.11.4.14 std::string EvaporativeCooling::outFilesPrefix [private]

prefix for all output files

Definition at line 132 of file EvaporativeCooling.h.

#### 6.11.4.15 po::variables_map EvaporativeCooling::paramsMap [private]

command line parameters map

Definition at line 130 of file EvaporativeCooling.h.

#### 6.11.4.16 RandomJungle* EvaporativeCooling::randomJungle [private]

pointer to a RandomJungle algorithm object

Definition at line 143 of file EvaporativeCooling.h.

#### 6.11.4.17 ReliefF* EvaporativeCooling::reliefF [private]

pointer to a ReliefF or RReliefF algorithm object

Definition at line 141 of file EvaporativeCooling.h.

#### 6.11.4.18 EcScores EvaporativeCooling::rfScores [private]

current relieff scores

Definition at line 152 of file EvaporativeCooling.h.

#### 6.11.4.19 EcScores EvaporativeCooling::rjScores [private]

current random jungle scores

Definition at line 150 of file EvaporativeCooling.h.

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

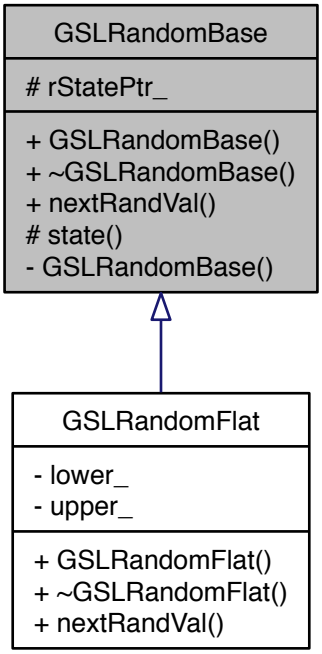
- src/library/EvaporativeCooling.h
- src/library/EvaporativeCooling.cpp

## 6.12 GSLRandomBase Class Reference

A base class for GNU Scientific Library (GSL) random number functions.

`#include <GSLRandomBase.h>`

Inheritance diagram for GSLRandomBase:




### Public Member Functions

- GSLRandomBase (int seedVal)
- virtual ~GSLRandomBase ()
- virtual double nextRandVal ()=0

### Protected Member Functions

- gsl_rng * state ()

### Protected Attributes

- gsl_rng * rStatePtr_

### Private Member Functions

- GSLRandomBase (const GSLRandomBase &rhs)

## 6.12.1 Detailed Description

A base class for GNU Scientific Library (GSL) random number functions.

The setup, initialization and clean-up is the same for all GSL random number functions. This class abstracts away these details, placing the stup and initialization in the class constructor and the clean-up in the class destructor. The class constructor is passed a seed value for the random number generator.

A class that provides access to one or more GSL random number functions should be derived from this class. This class must provide an implementation for the nextRandVal() pure virtual function. The nextRandVal will call the specific random number function (for example gsl_ran_ugaussian() for Gaussian distribution or gsl_ran_flat() for a flat random number distribution).

This class uses the default random number generator. At least on Windows XP using the Visual C++ 6.0 compiler the type definitions for the random functions (for example gsl_rng_mt19937 or gsl_rng_knuthran) would not link properly. Perhaps they are not properly exported from the pre-built library.

I decided to use the GSL because is is supported on all major platforms (UNIX, Linux and Windows) and provides high quality pseudo-random number generation support. The standard POSIX rand() function is notorious for its poor quality. While the random() function on UNIX provides better pseudo-random number quality, but is still not as good as functions like MT19937.

Definition at line 39 of file GSLRandomBase.h.

## 6.12.2 Constructor & Destructor Documentation

### 6.12.2.1 GSLRandomBase::GSLRandomBase ( const GSLRandomBase & *rhs* ) [private]

### 6.12.2.2 GSLRandomBase::GSLRandomBase ( int *seedVal* ) [inline]

Definition at line 52 of file GSLRandomBase.h.

### 6.12.2.3 virtual GSLRandomBase::~GSLRandomBase ( ) [inline, virtual]

Definition at line 67 of file GSLRandomBase.h.

## 6.12.3 Member Function Documentation

### 6.12.3.1 virtual double GSLRandomBase::nextRandVal ( ) [pure virtual]

Implemented in GSLRandomFlat.

### 6.12.3.2 gsl_rng* GSLRandomBase::state ( ) [inline, protected]

Definition at line 45 of file GSLRandomBase.h.

### 6.12.4 Member Data Documentation

#### 6.12.4.1 gsl_rng* GSLRandomBase::rStatePtr_ [protected]

Definition at line 48 of file GSLRandomBase.h.

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

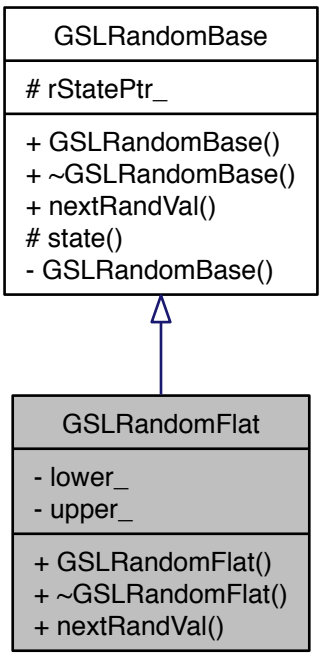
- src/library/GSLRandomBase.h

## 6.13 GSLRandomFlat Class Reference

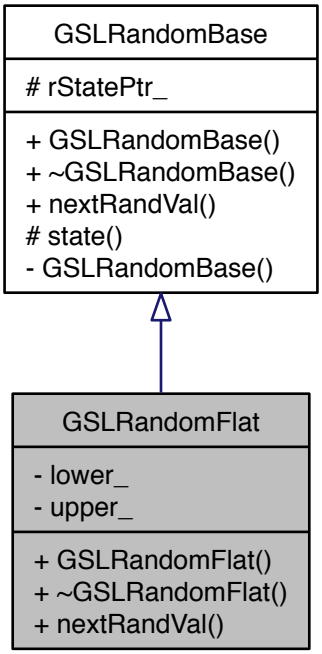
Random numbers in a flat, or uniform distribution.

`#include <GSLRandomFlat.h>`

Inheritance diagram for GSLRandomFlat:



Collaboration diagram for GSLRandomFlat:




### Public Member Functions

- GSLRandomFlat (int seedVal, double lower, double upper)
- ~GSLRandomFlat ()
- double nextRandVal ()

### Private Attributes

- double lower_
- double upper_

### 6.13.1 Detailed Description

Random numbers in a flat, or uniform distribution.

The class constructor is given a seed and a lower and upper bound value for the uniform distribution. The random numbers that result will be a uniform distribution in the range

```
lower <= randVal < upper
```

Definition at line 21 of file GSLRandomFlat.h.

### 6.13.2 Constructor & Destructor Documentation

#### 6.13.2.1 GSLRandomFlat::GSLRandomFlat ( int *seedVal,* double *lower,* double *upper* ) [inline]

Definition at line 27 of file GSLRandomFlat.h.

#### 6.13.2.2 GSLRandomFlat::~GSLRandomFlat ( ) [inline]

Definition at line 36 of file GSLRandomFlat.h.

### 6.13.3 Member Function Documentation

#### 6.13.3.1 double GSLRandomFlat::nextRandVal ( ) [inline, virtual]

Implements GSLRandomBase.

Definition at line 40 of file GSLRandomFlat.h.

### 6.13.4 Member Data Documentation

#### 6.13.4.1 double GSLRandomFlat::lower_ [private]

Definition at line 23 of file GSLRandomFlat.h.

#### 6.13.4.2 double GSLRandomFlat::upper_ [private]

Definition at line 23 of file GSLRandomFlat.h.

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

- src/library/GSLRandomFlat.h

## 6.14 insilico::is_classified< Type, charT > Class Template Reference

`#include <StringUtils.h>`

### Public Member Functions

- is_classified (std::ctype< charT > &ct)
- is_classified (const std::locale &loc=std::locale())
- bool operator() (charT c) const

### Private Attributes

- std::ctype< charT > const & m_ctype

### 6.14.1 Detailed Description

template<std::ctype_base::mask Type, class charT = char>class insilico::is_classified< Type, charT >

Definition at line 40 of file StringUtils.h.

### 6.14.2 Constructor & Destructor Documentation

#### 6.14.2.1 template<std::ctype_base::mask Type, class charT = char> insilico::is_classified< Type, charT >::is_classified ( std::ctype< charT > & *ct* ) [inline]

Definition at line 44 of file StringUtils.h.

#### 6.14.2.2 template<std::ctype_base::mask Type, class charT = char> insilico::is_classified< Type, charT >::is_classified ( const std::locale & *loc* = std::locale() ) [inline]

Definition at line 47 of file StringUtils.h.

### 6.14.3 Member Function Documentation

#### 6.14.3.1 template<std::ctype_base::mask Type, class charT = char> bool insilico::is_classified< Type, charT >::operator() ( charT *c* ) const [inline]

Definition at line 50 of file StringUtils.h.

### 6.14.4 Member Data Documentation

#### 6.14.4.1 template<std::ctype_base::mask Type, class charT = char> std::ctype<charT> const& insilico::is_classified< Type, charT >::m_ctype [private]

Definition at line 54 of file StringUtils.h.

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

- src/library/StringUtils.h

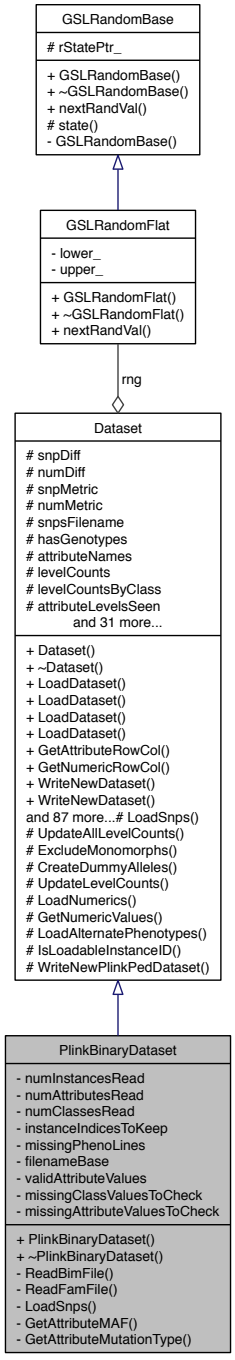
## 6.15 PlinkBinaryDataset Class Reference

Plink binary PED/BED file format reader.

`#include <PlinkBinaryDataset.h>`

Inheritance diagram for PlinkBinaryDataset:

Collaboration diagram for PlinkBinaryDataset:




## Public Member Functions

- PlinkBinaryDataset ()
- ~PlinkBinaryDataset ()

### Private Member Functions

- bool ReadBimFile (std::string bimFilename)

  *Load attribute information.*
- bool ReadFamFile (std::string famFilename)

  *Load individual information.*
- bool LoadSnps (std::string filename)

  *Load SNPs from file using the data set filename.*
- std::pair< char, double > GetAttributeMAF (unsigned int attributeIndex)

  *Get attribute minor allele and frequency.*
- AttributeMutationType GetAttributeMutationType (unsigned int attributeIndex)

  *Get attribute mutation type.*

### Private Attributes

- unsigned int numInstancesRead
- unsigned int numAttributesRead
- unsigned int numClassesRead
- std::vector< int > instanceIndicesToKeep
- std::vector< int > missingPhenoLines
- std::string filenameBase
- std::vector< std::string > validAttributeValues

  *for checking attribute values*
- std::vector< std::string > missingClassValuesToCheck

  *missing class values*
- std::vector< std::string > missingAttributeValuesToCheck

  *missing attribute values*

## 6.15.1 Detailed Description

Plink binary PED/BED file format reader.

**See also**

Dataset

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 3/10/11

Definition at line 22 of file PlinkBinaryDataset.h.

## 6.15.2 Constructor & Destructor Documentation

### 6.15.2.1 PlinkBinaryDataset::PlinkBinaryDataset ( )

Definition at line 35 of file PlinkBinaryDataset.cpp.

#### 6.15.2.2 PlinkBinaryDataset::~PlinkBinaryDataset ( ) [inline]

Definition at line 26 of file PlinkBinaryDataset.h.

### 6.15.3 Member Function Documentation

#### 6.15.3.1 pair< char, double > PlinkBinaryDataset::GetAttributeMAF ( unsigned int *attributeIndex* ) [private, virtual]

Get attribute minor allele and frequency.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

pair (minor allele, minor allele frequency)

An Introduction to Genetic Analysis by Griffiths, Miller, Suzuki, Lewontin and Gelbart, 2000, page 715.

Reimplemented from Dataset.

Definition at line 549 of file PlinkBinaryDataset.cpp.

#### 6.15.3.2 AttributeMutationType PlinkBinaryDataset::GetAttributeMutationType ( unsigned int *attributeIndex* ) [private, virtual]

Get attribute mutation type.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

mutation type (transition, transversion, unknown)

Reimplemented from Dataset.

Definition at line 557 of file PlinkBinaryDataset.cpp.

#### 6.15.3.3 bool PlinkBinaryDataset::LoadSnps ( std::string *filename* ) [private, virtual]

Load SNPs from file using the data set filename.

---------- Beginning of private methods ----------

**Parameters**

| | | |
|---|---|---|
| in | *filename* | SNPs filename |
| in | *deRecodeA* | perform a recodeA operation after reading raw data? |

**Returns**

success

Remove instances that are not in instanceIdsToLoad or marked as missing phenotype - 11/1/11 Only remove missing phenotypes if no alt pheno file - 1/23/12

Passed all tests, so add this instance to the data set

Release memory used by filtered out instances

Open the data file and read line-by-line

Detect the class type

Reimplemented from Dataset.

Definition at line 46 of file PlinkBinaryDataset.cpp.

#### 6.15.3.4 bool PlinkBinaryDataset::ReadBimFile ( std::string *bimFilename* ) [private]

Load attribute information.

**Parameters**

| | | |
|---|---|---|
| in | *PLINK* | bim filename |

**Returns**

success

set the mutation type

Definition at line 369 of file PlinkBinaryDataset.cpp.

#### 6.15.3.5 bool PlinkBinaryDataset::ReadFamFile ( std::string *famFilename* ) [private]

Load individual information.

**Parameters**

| | | |
|---|---|---|
| in | *PLIN* | fam filename |

**Returns**

success

Detect the class type

Read attribute information from the fam file

assign class level

Create a new instance for this individual

Definition at line 438 of file PlinkBinaryDataset.cpp.

### 6.15.4 Member Data Documentation

#### 6.15.4.1 std::string PlinkBinaryDataset::filenameBase [private]

Definition at line 51 of file PlinkBinaryDataset.h.

#### 6.15.4.2 std::vector<int> PlinkBinaryDataset::instanceIndicesToKeep [private]

Definition at line 48 of file PlinkBinaryDataset.h.

#### 6.15.4.3 std::vector<std::string> PlinkBinaryDataset::missingAttributeValuesToCheck [private]

missing attribute values

Definition at line 58 of file PlinkBinaryDataset.h.

#### 6.15.4.4 std::vector<std::string> PlinkBinaryDataset::missingClassValuesToCheck [private]

missing class values

Definition at line 56 of file PlinkBinaryDataset.h.

#### 6.15.4.5 std::vector<int> PlinkBinaryDataset::missingPhenoLines [private]

Definition at line 49 of file PlinkBinaryDataset.h.

#### 6.15.4.6 unsigned int PlinkBinaryDataset::numAttributesRead [private]

Definition at line 45 of file PlinkBinaryDataset.h.

#### 6.15.4.7 unsigned int PlinkBinaryDataset::numClassesRead [private]

Definition at line 46 of file PlinkBinaryDataset.h.

#### 6.15.4.8 unsigned int PlinkBinaryDataset::numInstancesRead [private]

Definition at line 44 of file PlinkBinaryDataset.h.

#### 6.15.4.9 std::vector<std::string> PlinkBinaryDataset::validAttributeValues [private]

for checking attribute values

Definition at line 54 of file PlinkBinaryDataset.h.

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

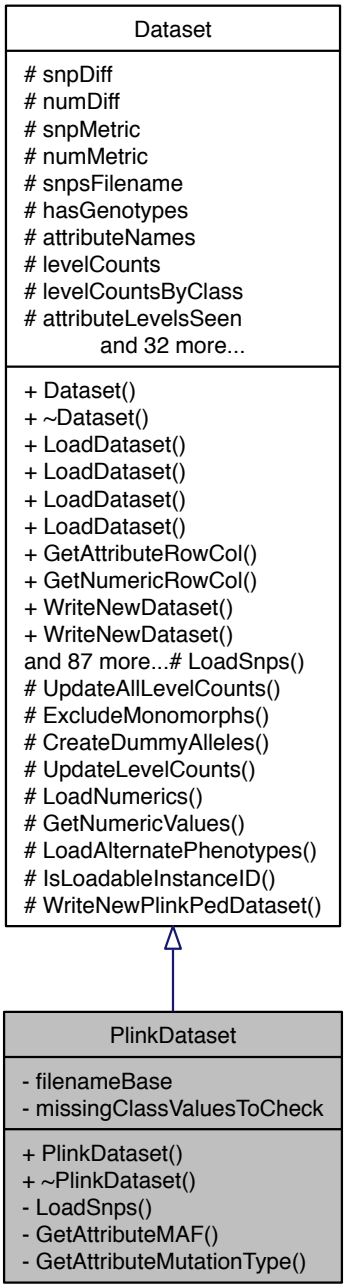
- src/library/PlinkBinaryDataset.h
- src/library/PlinkBinaryDataset.cpp

## 6.16 PlinkDataset Class Reference

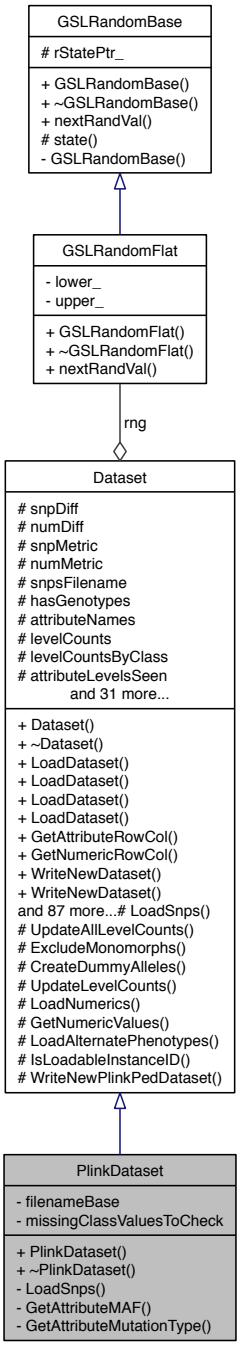
Plink MAP/PED file format reader.

`#include <PlinkDataset.h>`

Inheritance diagram for PlinkDataset:



Collaboration diagram for PlinkDataset:




## Public Member Functions

- PlinkDataset ()

  *Construct a PLINK data set reader. Calls Dataset base class constructor.*

- ~PlinkDataset ()

### Private Member Functions

- bool LoadSnps (std::string filename)

  *Load SNPs from file using the data set filename.*
- std::pair< char, double > GetAttributeMAF (unsigned int attributeIndex)

  *Get attribute minor allele and frequency.*
- AttributeMutationType GetAttributeMutationType (unsigned int attributeIndex)

  *Get attribute mutation type.*

### Private Attributes

- std::string filenameBase

  *base filename for auxiliary files*
- std::vector< std::string > missingClassValuesToCheck

  *missing class values*

## 6.16.1 Detailed Description

Plink MAP/PED file format reader.

**See also**

Dataset

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 2/1/11

Definition at line 36 of file PlinkDataset.h.

## 6.16.2 Constructor & Destructor Documentation

### 6.16.2.1 PlinkDataset::PlinkDataset ( )

Construct a PLINK data set reader. Calls Dataset base class constructor.

Definition at line 25 of file PlinkDataset.cpp.

### 6.16.2.2 PlinkDataset::~PlinkDataset ( ) [inline]

Definition at line 41 of file PlinkDataset.h.

## 6.16.3 Member Function Documentation

### 6.16.3.1 pair< char, double > PlinkDataset::GetAttributeMAF ( unsigned int *attributeIndex* ) [private, virtual]

Get attribute minor allele and frequency.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

pair (minor allele, minor allele frequency)

An Introduction to Genetic Analysis by Griffiths, Miller, Suzuki, Lewontin and Gelbart, 2000, page 715.

Reimplemented from Dataset.

Definition at line 399 of file PlinkDataset.cpp.

#### 6.16.3.2 AttributeMutationType PlinkDataset::GetAttributeMutationType ( unsigned int *attributeIndex* ) [private, virtual]

Get attribute mutation type.

**Parameters**

| | | |
|---|---|---|
| in | *attribute* | index |

**Returns**

mutation type (transition, transversion, unknown)

Reimplemented from Dataset.

Definition at line 408 of file PlinkDataset.cpp.

#### 6.16.3.3 bool PlinkDataset::LoadSnps ( std::string *filename* ) [private, virtual]

Load SNPs from file using the data set filename.

---------- Beginning of private methods ----------

**Parameters**

| | | |
|---|---|---|
| in | *filename* | SNPs filename |
| in | *deRecodeA* | perform a recodeA operation after reading raw data? |

**Returns**

success

read attribute information from the map file

Detect the class type

read attribute values from the ped file

determine the MAP file type

get ID for matching between PLINK data, numeric and pheno files

assign class level

set the mutation type

Open the data file and read line-by-line

Detect the class type

Reimplemented from Dataset.

Definition at line 30 of file PlinkDataset.cpp.

### 6.16.4 Member Data Documentation

#### 6.16.4.1 std::string PlinkDataset::filenameBase [private]

base filename for auxiliary files

Definition at line 48 of file PlinkDataset.h.

#### 6.16.4.2 std::vector<std::string> PlinkDataset::missingClassValuesToCheck [private]

missing class values

Definition at line 50 of file PlinkDataset.h.

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

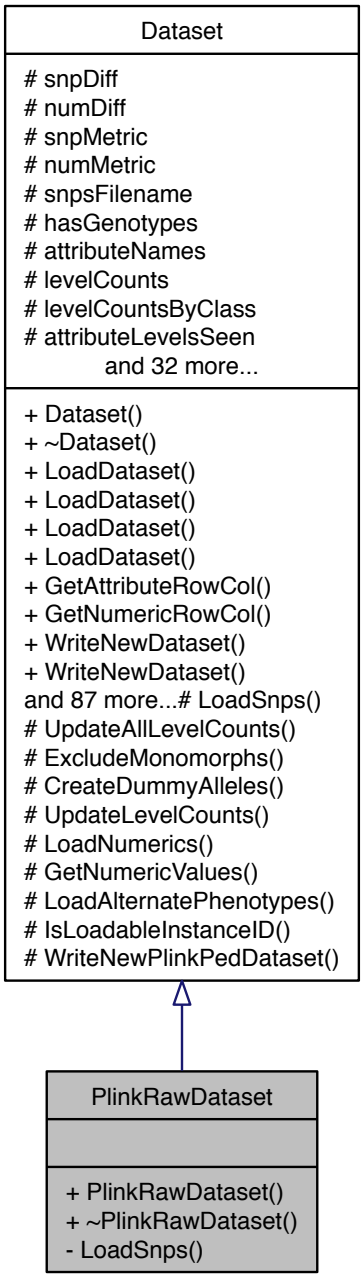
- src/library/PlinkDataset.h
- src/library/PlinkDataset.cpp

## 6.17 PlinkRawDataset Class Reference

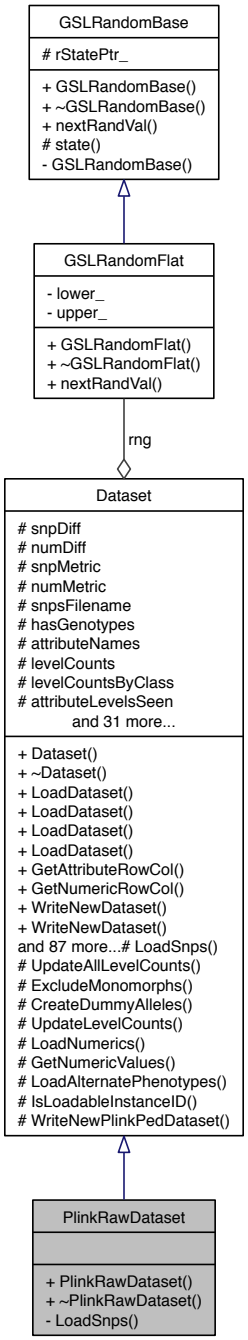
Plink recodeA/RAW file format reader.

`#include <PlinkRawDataset.h>`

Inheritance diagram for PlinkRawDataset:



Collaboration diagram for PlinkRawDataset:




## Public Member Functions

- PlinkRawDataset ()
- ~PlinkRawDataset ()

### Private Member Functions

- bool LoadSnps (std::string filename)

  *Load SNPs from file using the data set filename.*

## 6.17.1 Detailed Description

Plink recodeA/RAW file format reader.

**See also**

Dataset

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 2/24/11

Definition at line 23 of file PlinkRawDataset.h.

## 6.17.2 Constructor & Destructor Documentation

### 6.17.2.1 PlinkRawDataset::PlinkRawDataset ( )

Definition at line 22 of file PlinkRawDataset.cpp.

### 6.17.2.2 PlinkRawDataset::~PlinkRawDataset ( ) [inline]

Definition at line 27 of file PlinkRawDataset.h.

## 6.17.3 Member Function Documentation

### 6.17.3.1 bool PlinkRawDataset::LoadSnps ( std::string *filename* ) [private, virtual]

Load SNPs from file using the data set filename.

---------- Beginning of private methods ----------

**Parameters**

| | | |
|---|---|---|
| in | *filename* | SNPs filename |
| in | *deRecodeA* | perform a recodeA operation after reading raw data? |

**Returns**

success

Detect the class type

Open the data file and read line-by-line

Detect the class type

Reimplemented from Dataset.

Definition at line 25 of file PlinkRawDataset.cpp.

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

- src/library/PlinkRawDataset.h

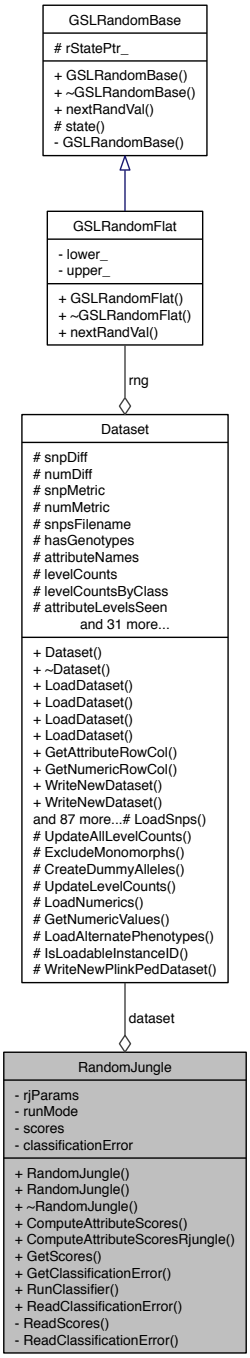
- src/library/PlinkRawDataset.cpp

## 6.18 RandomJungle Class Reference

RandomJungle attribute ranking algorithm.

`#include <RandomJungle.h>`

Collaboration diagram for RandomJungle:




## Public Member Functions

- RandomJungle (Dataset *ds, po::variables_map &vm)

  *Construct an RandomJungle algorithm object.*
- RandomJungle (Dataset *ds, ConfigMap &vm)

  *Construct an RandomJungle algorithm object.*
- virtual ~RandomJungle ()

- bool ComputeAttributeScores ()

  *Score attributes by getting Random Jungle importance scores.*
- bool ComputeAttributeScoresRjungle ()

  *Score attributes by getting Random Jungle importance scores.*
- std::vector< std::pair< double, std::string > > GetScores ()

  *Get the (importance) scores as a vector of pairs: score, attribute name.*
- double GetClassificationError ()

  *Get the classification error of the last classifier run.*

### Static Public Member Functions

- static bool RunClassifier (std::string csvFile, ConfigMap &vm, RandomJungleTreeType treeType, double &classError)

  *Run Random jungle as a classifier without instantiating a Random Jungle.*
- static bool ReadClassificationError (std::string confusionFilename, RandomJungleTreeType treeType, double &classifierError)

  *Read the classification error from file into variable.*

### Private Member Functions

- bool ReadScores (std::string importanceFilename)

  *Read the importance scores as attribute rankings from file into member vector scores: pair<double, string>*
- bool ReadClassificationError (std::string confusionFilename)

  *Read classification error from file into member variable classificationError.*

### Private Attributes

- RJunglePar rjParams

  *RandomJungle parameters object.*
- RandomJungleRunMode runMode

  *RandomJungle calling style.*
- Dataset * dataset

  *pointer to a Dataset object*
- std::vector< std::pair< double, std::string > > scores

  *vector of pairs: scores, attribute names*
- double classificationError

  *last classification error*

## 6.18.1 Detailed Description

RandomJungle attribute ranking algorithm.

Adapter class to map EC call for Random Jungle importance scores to Random Jungle library functions.

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 10/16/11

Definition at line 32 of file RandomJungle.h.

### 6.18.2 Constructor & Destructor Documentation

#### 6.18.2.1 RandomJungle::RandomJungle ( Dataset * *ds*, po::variables_map & *vm* )

Construct an RandomJungle algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *vm* | reference to a Boost map of command line options |

set rjParams

Definition at line 127 of file RandomJungle.cpp.

#### 6.18.2.2 RandomJungle::RandomJungle ( Dataset * *ds*, ConfigMap & *vm* )

Construct an RandomJungle algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *configMap* | reference ConfigMap (map<string, string>) |

Definition at line 182 of file RandomJungle.cpp.

#### 6.18.2.3 RandomJungle::~RandomJungle ( ) [virtual]

Definition at line 226 of file RandomJungle.cpp.

### 6.18.3 Member Function Documentation

#### 6.18.3.1 bool RandomJungle::ComputeAttributeScores ( )

Score attributes by getting Random Jungle importance scores.

loads rjScores map

loads rj classification error from confusion file - 4/11/12

Definition at line 232 of file RandomJungle.cpp.

#### 6.18.3.2 bool RandomJungle::ComputeAttributeScoresRjungle ( )

Score attributes by getting Random Jungle importance scores.

save the current data set to a temporary file for rjungle

run rjungle through a system call to the shell

loads rjScores map from importance file

loads rj classification error from confusion file

remove the temporary file

Definition at line 468 of file RandomJungle.cpp.

#### 6.18.3.3 double RandomJungle::GetClassificationError ( )

Get the classification error of the last classifier run.

Definition at line 540 of file RandomJungle.cpp.

#### 6.18.3.4 vector< pair< double, string > > RandomJungle::GetScores ( )

Get the (importance) scores as a vector of pairs: score, attribute name.

**Returns**

vector of pairs

Definition at line 536 of file RandomJungle.cpp.

#### 6.18.3.5 bool RandomJungle::ReadClassificationError ( std::string *confusionFilename,* RandomJungleTreeType *treeType,* double & *classifierError* ) [static]

Read the classification error from file into variable.

open the confusion file

strip the header line(s), read the error, cast to double

Definition at line 78 of file RandomJungle.cpp.

#### 6.18.3.6 bool RandomJungle::ReadClassificationError ( std::string *confusionFilename* ) [private]

Read classification error from file into member variable classificationError.

Definition at line 618 of file RandomJungle.cpp.

#### 6.18.3.7 bool RandomJungle::ReadScores ( std::string *importanceFilename* ) [private]

Read the importance scores as attribute rankings from file into member vector scores: pair<double, string>

Definition at line 544 of file RandomJungle.cpp.

#### 6.18.3.8 bool RandomJungle::RunClassifier ( std::string *csvFile,* ConfigMap & *vm,* RandomJungleTreeType *treeType,* double & *classError* ) [static]

Run Random jungle as a classifier without instantiating a Random Jungle.

run rjungle through a system call to the shell

loads rj classification error from confusion file

Definition at line 36 of file RandomJungle.cpp.

### 6.18.4 Member Data Documentation

#### 6.18.4.1 double RandomJungle::classificationError [private]

last classification error

Definition at line 82 of file RandomJungle.h.

#### 6.18.4.2 Dataset* RandomJungle::dataset [private]

pointer to a Dataset object

Definition at line 78 of file RandomJungle.h.

#### 6.18.4.3 RJunglePar RandomJungle::rjParams [private]

RandomJungle parameters object.

Definition at line 74 of file RandomJungle.h.

#### 6.18.4.4 RandomJungleRunMode RandomJungle::runMode [private]

RandomJungle calling style.

Definition at line 76 of file RandomJungle.h.

#### 6.18.4.5 std::vector<std::pair<double, std::string> > RandomJungle::scores [private]

vector of pairs: scores, attribute names

Definition at line 80 of file RandomJungle.h.

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

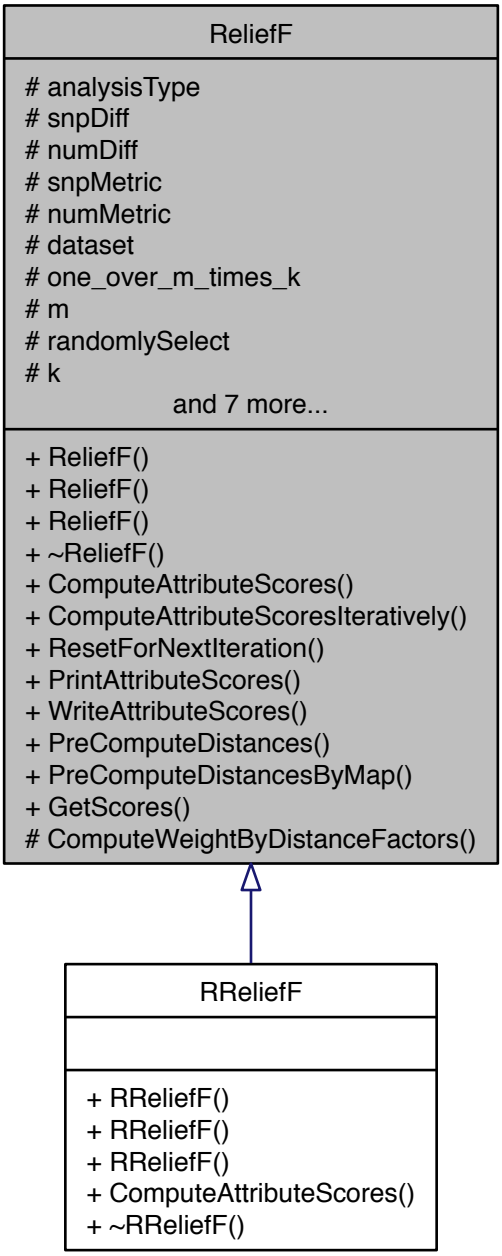
- src/library/RandomJungle.h
- src/library/RandomJungle.cpp

## 6.19 ReliefF Class Reference

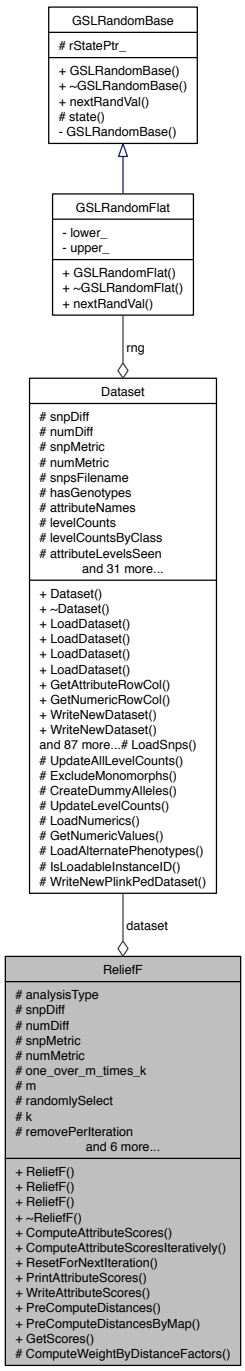
ReliefF attribute ranking algorithm.

`#include <ReliefF.h>`

Inheritance diagram for ReliefF:



Collaboration diagram for ReliefF:




## Public Member Functions

- ReliefF (Dataset ∗ds, AnalysisType anaType)

  *Construct an ReliefF algorithm object.*
- ReliefF (Dataset ∗ds, po::variables_map &vm, AnalysisType anaType)

  *Construct an ReliefF algorithm object.*
- ReliefF (Dataset ∗ds, ConfigMap &vm, AnalysisType anaType)

*Construct an ReliefF algorithm object.*

- virtual ~ReliefF ()
- virtual bool ComputeAttributeScores ()

  *Compute the ReliefF scores for the current set of attributes.*
- bool ComputeAttributeScoresIteratively ()

  *Compute the ReliefF scores by iteratively removing worst attributes.*
- bool ResetForNextIteration ()

  *Resets some data structures for the next iteration of ReliefF.*
- void PrintAttributeScores (std::ofstream &outStream)

  *Write the scores and attribute names to stream.*
- void WriteAttributeScores (std::string baseFilename)

  *Write the scores and attribute names to file.*
- bool PreComputeDistances ()

  *Precompute all pairwise instance-to-instance distances.*
- bool PreComputeDistancesByMap ()

  *Precompute all pairwise distances homoring excluded instances.*
- std::vector< std::pair< double, std::string > > GetScores ()

  *Get the last computed ReliefF scores.*

## Protected Member Functions

- bool ComputeWeightByDistanceFactors ()

  *Compute the weight by distance factors for nearest neighbors.*

## Protected Attributes

- AnalysisType analysisType

  *type of analysis to perform*
- double(* snpDiff )(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Compute the discrete difference in an attribute between two instances.*
- double(* numDiff )(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Compute the continuous difference in an attribute between two instances.*
- std::string snpMetric

  *the name of discrete diff(erence) function*
- std::string numMetric

  *the name of continuous diff(erence) function*
- Dataset * dataset

  *the dataset on which the algorithm is working*
- double one_over_m_times_k

  *nomalizing factor for ReliefF m * k loop*
- unsigned int m

  *number of instances to sample*
- bool randomlySelect

  *are instances being randomly selected?*
- unsigned int k

  *k nearest neighbors*
- unsigned int removePerIteration

  *number of attributes to remove each iteration if running iteratively*
- bool doRemovePercent

*are we removing a percentage per iteration?*

- double removePercentage

  *percentage of attributes to remove per iteration if running iteratively*

- std::string weightByDistanceMethod

  *name of the weight-by-distance method*

- double weightByDistanceSigma

  *sigma value used in exponential decay weight-by-distance*

- std::vector< double > W

  *attribute scores/weights*

- std::vector< std::string > scoreNames

  *attribute names associated with scores*

- std::map< std::string, double > finalScores

  *final scores after all iterations*

### 6.19.1 Detailed Description

ReliefF attribute ranking algorithm.

Totally redone for the McKinney insilico lab in 2011. Large refactoring to move all attribute elimination handling to the Dataset and its subclasses. 9/11/11

**See also**

RReliefF

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 7/16/05

Definition at line 32 of file ReliefF.h.

### 6.19.2 Constructor & Destructor Documentation

#### 6.19.2.1 ReliefF::ReliefF ( Dataset * *ds*, AnalysisType *anaType* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *anaType* | analysis type |

Definition at line 68 of file ReliefF.cpp.

#### 6.19.2.2 ReliefF::ReliefF ( Dataset * *ds*, po::variables_map & *vm*, AnalysisType *anaType* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *vm* | reference to a Boost map of command line options |
| in | *anaType* | analysis type |

set the SNP metric function pointer

Definition at line 120 of file ReliefF.cpp.

#### 6.19.2.3 ReliefF::ReliefF ( Dataset * *ds*, ConfigMap & *vm*, AnalysisType *anaType* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *configMap* | reference to a ConfigMap (map<string, string>) |
| in | *anaType* | analysis type |

set the SNP metric function pointer

Definition at line 263 of file ReliefF.cpp.

#### 6.19.2.4 ReliefF::~ReliefF ( ) [virtual]

Definition at line 415 of file ReliefF.cpp.

### 6.19.3 Member Function Documentation

#### 6.19.3.1 bool ReliefF::ComputeAttributeScores ( ) [virtual]

Compute the ReliefF scores for the current set of attributes.

Implements ReliefF algorithm: Marko Robnik-Sikonja, Igor Kononenko: Theoretical and Empirical Analysis of ReliefF and RReliefF. Machine Learning Journal, 53:23-69, 2003 http://lkm.fri.uni-lj.si/rmarko/papers/robnik03-mlj.pdf algorithm line 1

algorithm line 2

algorithm lines 4, 5 and 6

algorithm line 7

algorithm line 8

algorithm line 9

Reimplemented in RReliefF.

Definition at line 418 of file ReliefF.cpp.

#### 6.19.3.2 bool ReliefF::ComputeAttributeScoresIteratively ( )

Compute the ReliefF scores by iteratively removing worst attributes.

Definition at line 590 of file ReliefF.cpp.

#### 6.19.3.3 bool ReliefF::ComputeWeightByDistanceFactors ( ) [protected]

Compute the weight by distance factors for nearest neighbors.

Definition at line 962 of file ReliefF.cpp.

#### 6.19.3.4 vector< pair< double, string > > ReliefF::GetScores ( )

Get the last computed ReliefF scores.

Definition at line 948 of file ReliefF.cpp.

#### 6.19.3.5 bool ReliefF::PreComputeDistances ( )

Precompute all pairwise instance-to-instance distances.

be sure to call Dataset::ComputeInstanceToInstanceDistance

Definition at line 702 of file ReliefF.cpp.

#### 6.19.3.6 bool ReliefF::PreComputeDistancesByMap ( )

Precompute all pairwise distances homoring excluded instances.

Definition at line 835 of file ReliefF.cpp.

#### 6.19.3.7 void ReliefF::PrintAttributeScores ( std::ofstream & *outStream* )

Write the scores and attribute names to stream.

**Parameters**

| | | |
|---|---|---|
| in | *outStream* | stream to write score-attribute name pairs |

Definition at line 671 of file ReliefF.cpp.

#### 6.19.3.8 bool ReliefF::ResetForNextIteration ( )

Resets some data structures for the next iteration of ReliefF.

Definition at line 664 of file ReliefF.cpp.

#### 6.19.3.9 void ReliefF::WriteAttributeScores ( std::string *baseFilename* )

Write the scores and attribute names to file.

**Parameters**

| | | |
|---|---|---|
| in | *baseFIlename* | filename to write score-attribute name pairs |

Definition at line 682 of file ReliefF.cpp.

### 6.19.4 Member Data Documentation

#### 6.19.4.1 AnalysisType ReliefF::analysisType [protected]

type of analysis to perform

Definition at line 88 of file ReliefF.h.

#### 6.19.4.2 Dataset* ReliefF::dataset [protected]

the dataset on which the algorithm is working

Definition at line 114 of file ReliefF.h.

#### 6.19.4.3 bool ReliefF::doRemovePercent [protected]

are we removing a percentage per iteration?

Definition at line 126 of file ReliefF.h.

#### 6.19.4.4 std::map<std::string, double> ReliefF::finalScores [protected]

final scores after all iterations

Definition at line 139 of file ReliefF.h.

#### 6.19.4.5 unsigned int ReliefF::k [protected]

k nearest neighbors

Definition at line 122 of file ReliefF.h.

#### 6.19.4.6 unsigned int ReliefF::m [protected]

number of instances to sample

Definition at line 118 of file ReliefF.h.

#### 6.19.4.7 double(* ReliefF::numDiff)(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2) [protected]

Compute the continuous difference in an attribute between two instances.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into vector of all attributes |
| in | *dsi1* | pointer to DatasetInstance 1 |
| in | *dsi2* | pointer to DatasetInstance 2 |

**Returns**

diff(erence)

Definition at line 106 of file ReliefF.h.

#### 6.19.4.8 std::string ReliefF::numMetric [protected]

the name of continuous diff(erence) function

Definition at line 112 of file ReliefF.h.

#### 6.19.4.9 double ReliefF::one_over_m_times_k [protected]

nomalizing factor for ReliefF m * k loop

Definition at line 116 of file ReliefF.h.

#### 6.19.4.10 bool ReliefF::randomlySelect [protected]

are instances being randomly selected?

Definition at line 120 of file ReliefF.h.

#### 6.19.4.11 double ReliefF::removePercentage [protected]

percentage of attributes to remove per iteration if running iteratively

Definition at line 128 of file ReliefF.h.

#### 6.19.4.12 unsigned int ReliefF::removePerIteration [protected]

number of attributes to remove each iteration if running iteratively

Definition at line 124 of file ReliefF.h.

#### 6.19.4.13 std::vector<std::string> ReliefF::scoreNames [protected]

attribute names associated with scores

Definition at line 137 of file ReliefF.h.

#### 6.19.4.14 double(* ReliefF::snpDiff)(unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2) [protected]

Compute the discrete difference in an attribute between two instances.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into vector of all attributes |
| in | *dsi1* | pointer to DatasetInstance 1 |
| in | *dsi2* | pointer to DatasetInstance 2 |

**Returns**

diff(erence)

Definition at line 96 of file ReliefF.h.

#### 6.19.4.15 std::string ReliefF::snpMetric [protected]

the name of discrete diff(erence) function

Definition at line 110 of file ReliefF.h.

#### 6.19.4.16 std::vector<double> ReliefF::W [protected]

attribute scores/weights

Definition at line 135 of file ReliefF.h.

#### 6.19.4.17 std::string ReliefF::weightByDistanceMethod [protected]

name of the weight-by-distance method

Definition at line 130 of file ReliefF.h.

#### 6.19.4.18 double ReliefF::weightByDistanceSigma [protected]

sigma value used in exponential decay weight-by-distance

Definition at line 132 of file ReliefF.h.

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

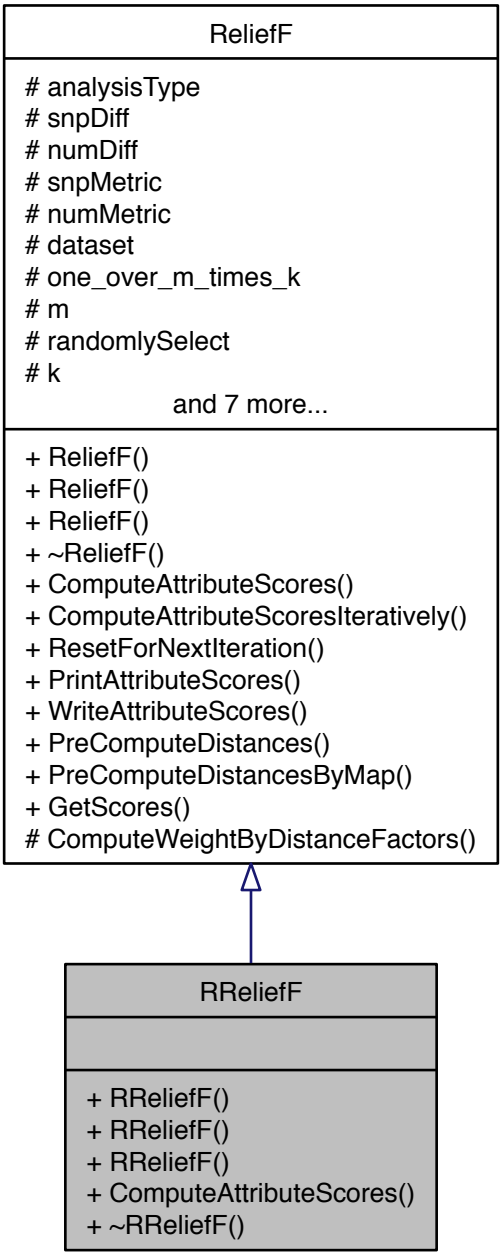
- src/library/ReliefF.h
- src/library/ReliefF.cpp

## 6.20 RReliefF Class Reference

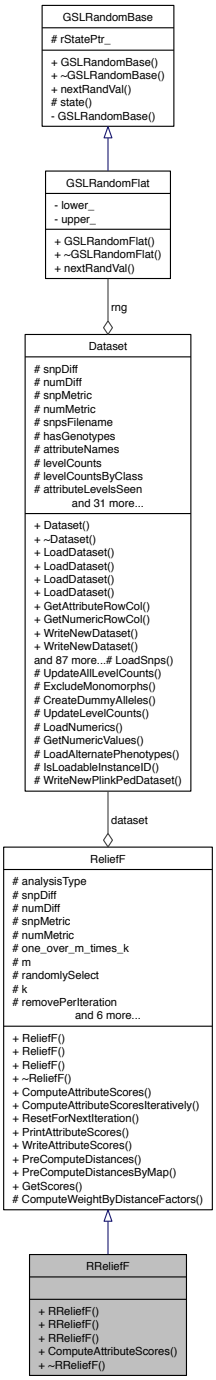
Regression ReliefF attribute ranking algorithm.

`#include <RReliefF.h>`

Inheritance diagram for RReliefF:



Collaboration diagram for RReliefF:




## Public Member Functions

- RReliefF (Dataset *ds)

  *Construct an ReliefF algorithm object.*
- RReliefF (Dataset *ds, po::variables_map &vm)

  *Construct an ReliefF algorithm object.*
- RReliefF (Dataset *ds, ConfigMap &configMap)

*Construct an ReliefF algorithm object.*

- bool ComputeAttributeScores ()

  *Compute the ReliefF scores for the current set of attributes.*

- virtual ~RReliefF ()

### 6.20.1 Detailed Description

Regression ReliefF attribute ranking algorithm.

Totally redone for the McKinney insilico lab in 2011. Large refactoring to move all attribute elimination handling to the Dataset and its subclasses. 9/11/11

**See also**

ReliefF

**Author**

Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 9/27/11

Definition at line 33 of file RReliefF.h.

### 6.20.2 Constructor & Destructor Documentation

#### 6.20.2.1 RReliefF::RReliefF ( Dataset * *ds* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |

Definition at line 19 of file RReliefF.cpp.

#### 6.20.2.2 RReliefF::RReliefF ( Dataset * *ds*, po::variables_map & *vm* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *vm* | reference to a Boost map of command line options |

Definition at line 34 of file RReliefF.cpp.

#### 6.20.2.3 RReliefF::RReliefF ( Dataset * *ds*, ConfigMap & *configMap* )

Construct an ReliefF algorithm object.

**Parameters**

| | | |
|---|---|---|
| in | *ds* | pointer to a Dataset object |
| in | *configMap* | reference to a ConfigMap (map<string, string>) |

Definition at line 49 of file RReliefF.cpp.

#### 6.20.2.4 RReliefF::~RReliefF ( ) [virtual]

Definition at line 64 of file RReliefF.cpp.

### 6.20.3 Member Function Documentation

#### 6.20.3.1 bool RReliefF::ComputeAttributeScores ( ) [virtual]

Compute the ReliefF scores for the current set of attributes.

Implements ReliefF algorithm: Marko Robnik-Sikonja, Igor Kononenko: Theoretical and Empirical Analysis of ReliefF and RReliefF. Machine Learning Journal, 53:23-69, 2003 http://lkm.fri.uni-lj.si/rmarko/papers/robnik03-mlj.pdf Used to hold the probability of a different class val given nearest instances (numeric class)

Used to hold the prob of different value of an attribute given nearest instances (numeric class case)

Used to hold the prob of a different class val and different att val given nearest instances (numeric class case)

algorithm line 1

algorithm line 2

algorithm lines 4, 5 and 6

algorithm line 7

algorithm line 8

algorithm line 9

Reimplemented from ReliefF.

Definition at line 67 of file RReliefF.cpp.

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

- src/library/RReliefF.h
- src/library/RReliefF.cpp

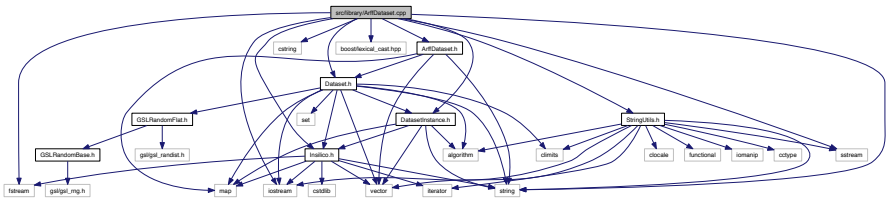
# Chapter 7

# File Documentation

## 7.1 src/library/ArffDataset.cpp File Reference

```
#include <string>
#include <iostream>
#include <fstream>
#include <cstring>
#include <sstream>
#include <boost/lexical_cast.hpp>
#include "Dataset.h"
#include "DatasetInstance.h"
#include "StringUtils.h"
#include "ArffDataset.h"
#include "Insilico.h"
```

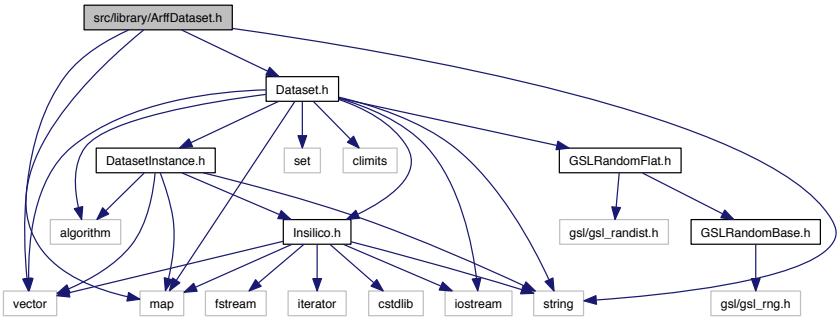
Include dependency graph for ArffDataset.cpp:



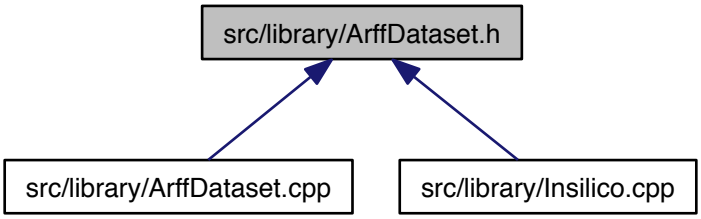

## 7.2 src/library/ArffDataset.h File Reference

```
#include <vector>
#include <map>
#include <string>
#include "Dataset.h"
```

Include dependency graph for ArffDataset.h:




This graph shows which files directly or indirectly include this file:




## Classes

- class ArffDataset

  *ARFF file format reader.*

## Enumerations

- enum ArffAttributeType {
  ARFF_NUMERIC_TYPE, ARFF_NOMINAL_TYPE, ARFF_STRING_TYPE, ARFF_DATE_TYPE,
  ARFF_ERROR_TYPE }

### 7.2.1 Enumeration Type Documentation

#### 7.2.1.1 enum ArffAttributeType

ARFF attribute types.

**Enumerator:**

***ARFF_NUMERIC_TYPE*** continuous levels

***ARFF_NOMINAL_TYPE*** discrete levels

***ARFF_STRING_TYPE*** string levels

***ARFF_DATE_TYPE*** date levels

***ARFF_ERROR_TYPE*** unknown type

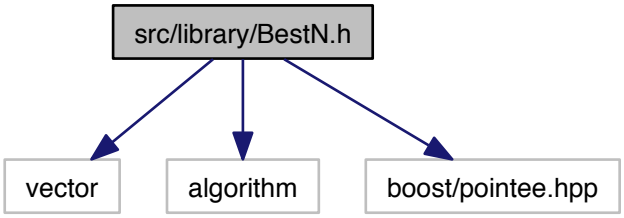
Definition at line 29 of file ArffDataset.h.

## 7.3 src/library/BestN.h File Reference

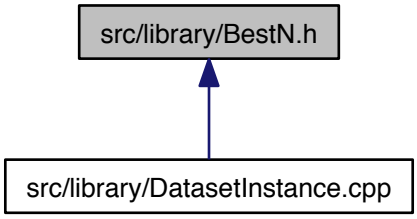
Find the best n keeping original order for ties - stable sort.

```
#include <vector>
#include <algorithm>
#include <boost/pointee.hpp>
```

Include dependency graph for BestN.h:




This graph shows which files directly or indirectly include this file:




### Namespaces

- namespace insilico

### Functions

- template<typename InputIt , typename OutputIt , typename Comp >
  void insilico::best_n (InputIt begin, InputIt end, OutputIt out, size_t n, Comp comp)

  *Get the best n values with ties keeping same original order.*

#### 7.3.1 Detailed Description

Find the best n keeping original order for ties - stable sort.

**Author**

Nate Barney

**Version**

1.0

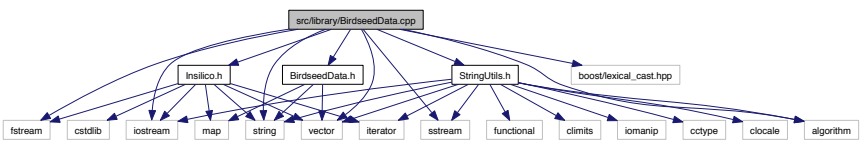
Contact: bill.c.white@gmail.com Created on: 4/7/04

Definition in file BestN.h.

## 7.4 src/library/BirdseedData.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <algorithm>
#include <boost/lexical_cast.hpp>
#include "BirdseedData.h"
#include "Insilico.h"
#include "StringUtils.h"
```

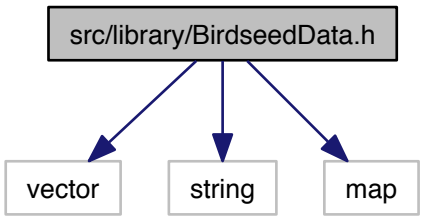
Include dependency graph for BirdseedData.cpp:



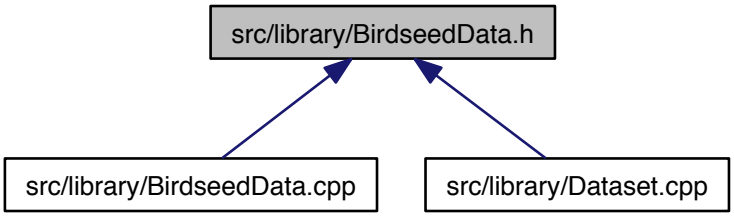

## 7.5 src/library/BirdseedData.h File Reference

```
#include <vector>
#include <string>
#include <map>
```

Include dependency graph for BirdseedData.h:




This graph shows which files directly or indirectly include this file:




### Classes

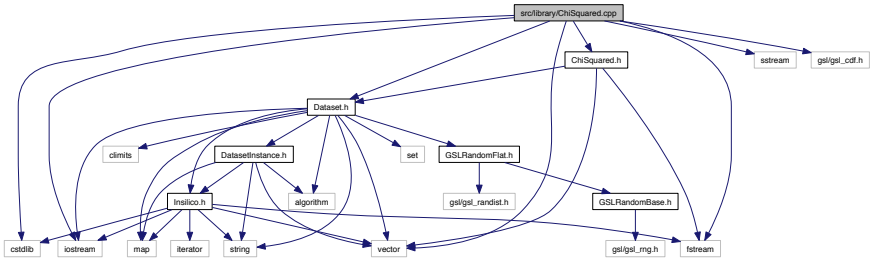
- class BirdseedData

  *Read Broad's Birdsuite Birdseed-called SNP data.*

## 7.6 src/library/ChiSquared.cpp File Reference

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include "gsl/gsl_cdf.h"
#include "ChiSquared.h"
#include "Dataset.h"
```

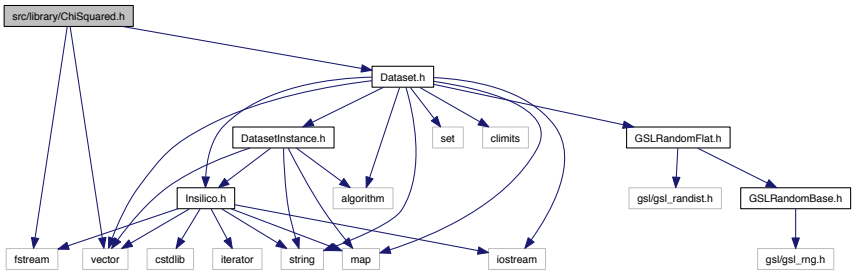
Include dependency graph for ChiSquared.cpp:



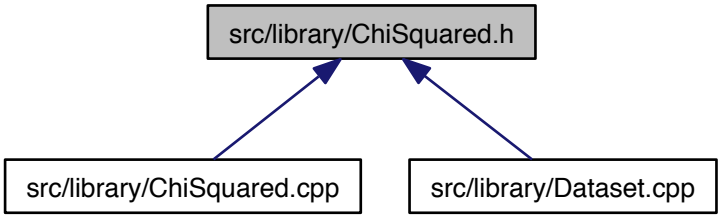

## 7.7 src/library/ChiSquared.h File Reference

```
#include <vector>
#include <fstream>
#include "Dataset.h"
```

Include dependency graph for ChiSquared.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class ChiSquared

  *Chi-squared attribute ranking algorithm.*

## 7.8 src/library/Dataset.cpp File Reference

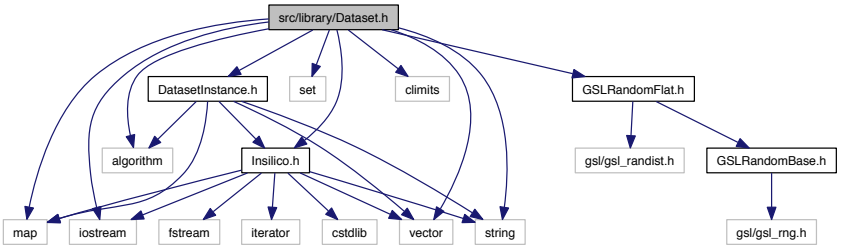
```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>
#include <vector>
#include <set>
#include <map>
#include <iterator>
#include <cmath>
#include <algorithm>
#include <numeric>
#include <sstream>
#include <limits.h>
#include <sys/types.h>
#include <unistd.h>
#include <assert.h>
#include <time.h>
#include <boost/lexical_cast.hpp>
#include "gsl/gsl_cdf.h"
#include "GSLRandomFlat.h"
#include "ChiSquared.h"
#include "Dataset.h"
#include "DatasetInstance.h"
#include "StringUtils.h"
#include "Statistics.h"
#include "Insilico.h"
#include "DgeData.h"
#include "BirdseedData.h"
#include "DistanceMetrics.h"
```

Include dependency graph for Dataset.cpp:

## 7.9 src/library/Dataset.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
#include <map>
#include <set>
#include <algorithm>
#include <climits>
#include "DatasetInstance.h"
#include "Insilico.h"
#include "GSLRandomFlat.h"
```

Include dependency graph for Dataset.h:




This graph shows which files directly or indirectly include this file:

## Classes

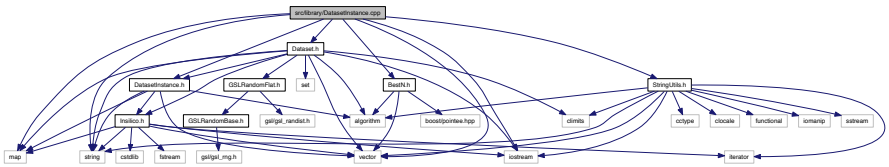
- class Dataset

  *Base class for collections of instances containing attributea and class.*

## 7.10 src/library/DatasetInstance.cpp File Reference

```
#include <iostream>
#include <string>
#include <vector>
#include <map>
#include "Dataset.h"
#include "DatasetInstance.h"
#include "StringUtils.h"
#include "BestN.h"
```

Include dependency graph for DatasetInstance.cpp:




## Classes

- class deref_less_bcw

## Typedefs

- typedef DistancePair T

*functor for T comparison*

### 7.10.1 Typedef Documentation

#### 7.10.1.1 typedef DistancePair T

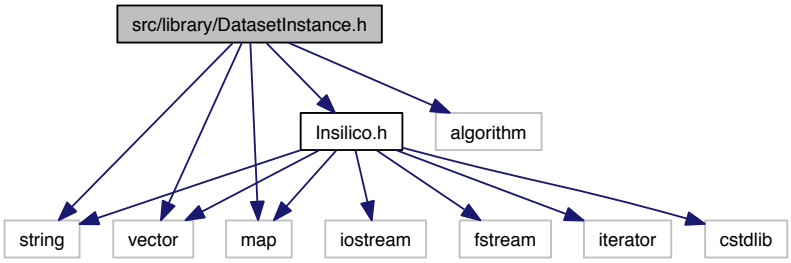
functor for T comparison

Definition at line 22 of file DatasetInstance.cpp.

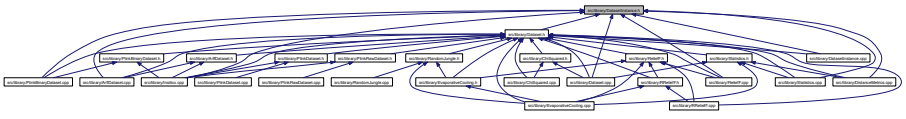
## 7.11 src/library/DatasetInstance.h File Reference

```
#include <string>
#include <vector>
#include <map>
#include <algorithm>
#include "Insilico.h"
```

Include dependency graph for DatasetInstance.h:




This graph shows which files directly or indirectly include this file:



### Classes

- class DatasetInstance

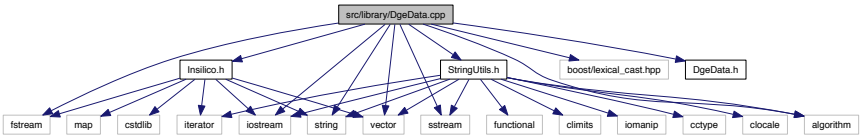
  *Class to hold dataset instances (rows of attributes).*

## 7.12 src/library/DgeData.cpp File Reference

```
#include <iostream>
```

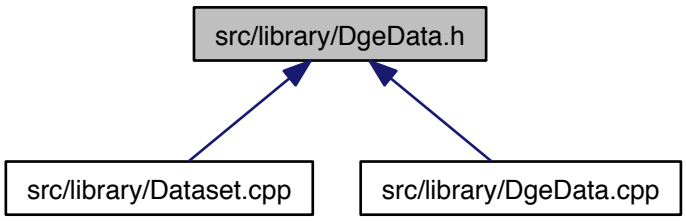
```
#include <fstream>
#include <string>
#include <sstream>
#include <vector>
#include <algorithm>
#include <boost/lexical_cast.hpp>
#include "DgeData.h"
#include "Insilico.h"
#include "StringUtils.h"
```

Include dependency graph for DgeData.cpp:




## 7.13 src/library/DgeData.h File Reference

This graph shows which files directly or indirectly include this file:




### Classes

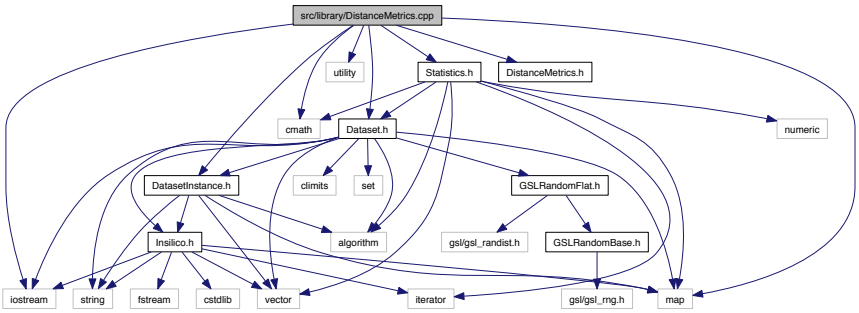
- class DgeData

  *Digital gene expression data.*

## 7.14 src/library/DistanceMetrics.cpp File Reference

```
#include <cmath>
#include <iostream>
#include <map>
#include <utility>
#include "Dataset.h"
#include "DistanceMetrics.h"
#include "DatasetInstance.h"
#include "Statistics.h"
```

Include dependency graph for DistanceMetrics.cpp:




## Functions

- pair< bool, double > CheckMissing (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Check for a missing discrete value and return value.*
- pair< bool, double > CheckMissingNumeric (unsigned int numericIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Check for a missing continuous value and return value.*
- double norm (double x, double minX, double maxX)

  *Normalizes a given value of a numeric attribute.*
- double diffAMM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Allele mismatch metric.*
- double diffGMM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Genotype mismatch metric.*
- double diffNCA (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Nucleotide count array (NCA) metric.*
- double diffKM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Kimura distance - considers transition/transversion mutation types.*
- double diffManhattan (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *"Manhattan" distance between continuous attributes.*
- double diffPredictedValueTau (DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Same as "Manhattan" distance but uses method calls versus public variables.*

### 7.14.1 Function Documentation

#### 7.14.1.1 pair<bool, double> CheckMissing ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Check for a missing discrete value and return value.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

pair: has missing value?, value for missing (true) or 0.0 (false)

Definition at line 21 of file DistanceMetrics.cpp.

#### 7.14.1.2 pair<bool, double> CheckMissingNumeric ( unsigned int *numericIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Check for a missing continuous value and return value.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

pair: has missing value?, value for missing (true) or 0.0 (false)

Definition at line 88 of file DistanceMetrics.cpp.

#### 7.14.1.3 double diffAMM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Allele mismatch metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) between attribute values: 0.0, 0.5, 1.0

Definition at line 135 of file DistanceMetrics.cpp.

#### 7.14.1.4 double diffGMM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Genotype mismatch metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) between attribute values: 0.0 (same) or 1.0 (not same)

Definition at line 150 of file DistanceMetrics.cpp.

#### 7.14.1.5 double diffKM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Kimura distance - considers transition/transversion mutation types.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) considering nucleotide mutation types

Definition at line 205 of file DistanceMetrics.cpp.

#### 7.14.1.6 double diffManhattan ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

"Manhattan" distance between continuous attributes.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

absolute value of difference divided by attribute's range

Definition at line 226 of file DistanceMetrics.cpp.

#### 7.14.1.7 double diffNCA ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Nucleotide count array (NCA) metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) considering nucleotide counts

Definition at line 164 of file DistanceMetrics.cpp.

#### 7.14.1.8 double diffPredictedValueTau ( DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Same as "Manhattan" distance but uses method calls versus public variables.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

absolute value of difference divided by attribute's range

Definition at line 251 of file DistanceMetrics.cpp.

#### 7.14.1.9 double norm ( double *x*, double *minX*, double *maxX* )

Normalizes a given value of a numeric attribute.

Borrowed from Weka 8/18/11

**Parameters**

| | | |
|---|---|---|
| in | *x* | value |
| in | *minX* | minimum value for x |
| in | *maxX* | maximum value for x |

**Returns**

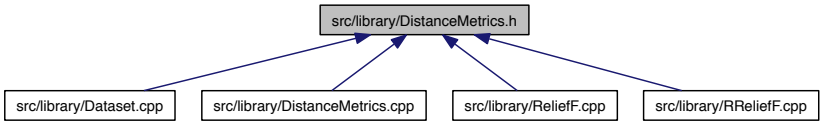
normalized value

Definition at line 127 of file DistanceMetrics.cpp.

## 7.15 src/library/DistanceMetrics.h File Reference

Distance metrics for ReliefF.

This graph shows which files directly or indirectly include this file:




### Functions

- std::pair< bool, double > CheckMissing (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Check for a missing discrete value and return value.*
- std::pair< bool, double > CheckMissingNumeric (unsigned int numericIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Check for a missing continuous value and return value.*
- double norm (double x, double minX, double maxX)

  *Normalizes a given value of a numeric attribute.*
- double diffAMM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Allele mismatch metric.*
- double diffGMM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Genotype mismatch metric.*
- double diffNCA (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Nucleotide count array (NCA) metric.*

- double diffKM (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Kimura distance - considers transition/transversion mutation types.*
- double diffManhattan (unsigned int attributeIndex, DatasetInstance *dsi1, DatasetInstance *dsi2)

  *"Manhattan" distance between continuous attributes.*
- double diffPredictedValueTau (DatasetInstance *dsi1, DatasetInstance *dsi2)

  *Same as "Manhattan" distance but uses method calls versus public variables.*

### 7.15.1 Detailed Description

Distance metrics for ReliefF.

**Author**

: Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on 3/29/11

Definition in file DistanceMetrics.h.

### 7.15.2 Function Documentation

#### 7.15.2.1 std::pair<bool, double> CheckMissing ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Check for a missing discrete value and return value.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

pair: has missing value?, value for missing (true) or 0.0 (false)

Definition at line 21 of file DistanceMetrics.cpp.

#### 7.15.2.2 std::pair<bool, double> CheckMissingNumeric ( unsigned int *numericIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Check for a missing continuous value and return value.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

pair: has missing value?, value for missing (true) or 0.0 (false)

Definition at line 88 of file DistanceMetrics.cpp.

#### 7.15.2.3 double diffAMM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Allele mismatch metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) between attribute values: 0.0, 0.5, 1.0

Definition at line 135 of file DistanceMetrics.cpp.

#### 7.15.2.4 double diffGMM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Genotype mismatch metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) between attribute values: 0.0 (same) or 1.0 (not same)

Definition at line 150 of file DistanceMetrics.cpp.

#### 7.15.2.5 double diffKM ( unsigned int *attributeIndex*, DatasetInstance * *dsi1*, DatasetInstance * *dsi2* )

Kimura distance - considers transition/transversion mutation types.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) considering nucleotide mutation types

Definition at line 205 of file DistanceMetrics.cpp.

#### 7.15.2.6 double diffManhattan ( unsigned int *attributeIndex,* DatasetInstance * *dsi1,* DatasetInstance * *dsi2* )

"Manhattan" distance between continuous attributes.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

absolute value of difference divided by attribute's range

Definition at line 226 of file DistanceMetrics.cpp.

#### 7.15.2.7 double diffNCA ( unsigned int *attributeIndex,* DatasetInstance * *dsi1,* DatasetInstance * *dsi2* )

Nucleotide count array (NCA) metric.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

diff(erence) considering nucleotide counts

Definition at line 164 of file DistanceMetrics.cpp.

#### 7.15.2.8 double diffPredictedValueTau ( DatasetInstance * *dsi1,* DatasetInstance * *dsi2* )

Same as "Manhattan" distance but uses method calls versus public variables.

**Parameters**

| | | |
|---|---|---|
| in | *attributeIndex* | index into the vector of attributes |
| in | *dsi1* | data set instance 1 |
| in | *dsi2* | data set instance 2 |

**Returns**

absolute value of difference divided by attribute's range

Definition at line 251 of file DistanceMetrics.cpp.

#### 7.15.2.9 double norm ( double *x,* double *minX,* double *maxX* )

Normalizes a given value of a numeric attribute.

Borrowed from Weka 8/18/11

**Parameters**

| | | |
|---|---|---|
| in | *x* | value |
| in | *minX* | minimum value for x |
| in | *maxX* | maximum value for x |

**Returns**

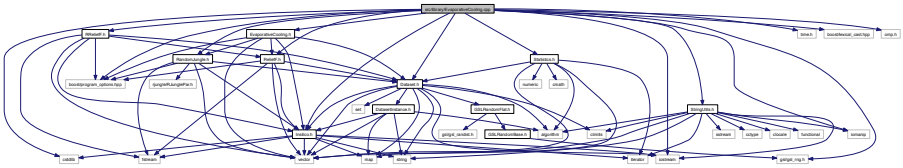
normalized value

Definition at line 127 of file DistanceMetrics.cpp.

## 7.16 src/library/EvaporativeCooling.cpp File Reference

```
#include <cstdlib>
#include <iostream>
#include <iomanip>
#include <time.h>
#include <boost/program_options.hpp>
#include <boost/lexical_cast.hpp>
#include <omp.h>
#include <gsl/gsl_rng.h>
#include "EvaporativeCooling.h"
#include "Dataset.h"
#include "Statistics.h"
#include "StringUtils.h"
#include "RandomJungle.h"
#include "ReliefF.h"
#include "RReliefF.h"
#include "Insilico.h"
```

Include dependency graph for EvaporativeCooling.cpp:



### Functions

- bool scoresSortAsc (const pair< double, string > &p1, const pair< double, string > &p2)
- bool scoresSortAscByName (const pair< double, string > &p1, const pair< double, string > &p2)
- bool scoresSortDesc (const pair< double, string > &p1, const pair< double, string > &p2)

### 7.16.1 Function Documentation

#### 7.16.1.1 bool scoresSortAsc ( const pair< double, string > & *p1*, const pair< double, string > & *p2* )

Definition at line 39 of file EvaporativeCooling.cpp.

#### 7.16.1.2 bool scoresSortAscByName ( const pair< double, string > & *p1*, const pair< double, string > & *p2* )

Definition at line 44 of file EvaporativeCooling.cpp.

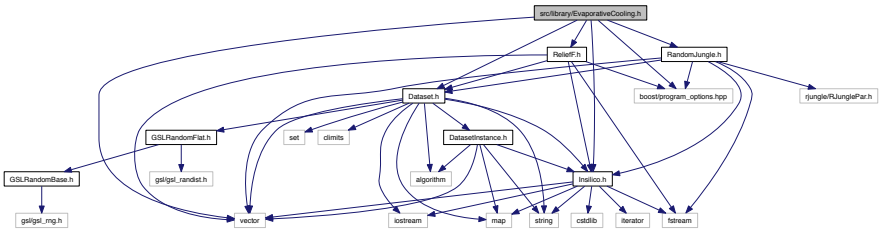
#### 7.16.1.3 bool scoresSortDesc ( const pair< double, string > & *p1*, const pair< double, string > & *p2* )

Definition at line 49 of file EvaporativeCooling.cpp.

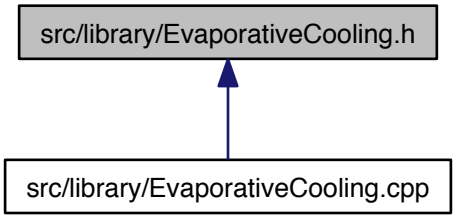
## 7.17 src/library/EvaporativeCooling.h File Reference

```
#include <vector>
#include <boost/program_options.hpp>
#include "Dataset.h"
#include "RandomJungle.h"
#include "ReliefF.h"
#include "Insilico.h"
```

Include dependency graph for EvaporativeCooling.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class EvaporativeCooling

  *Evaporative Cooling attribute ranking algorithm.*

### Typedefs

- typedef std::vector< std::pair
  < double, std::string > > EcScores

  *evaporative cooling scores - sorted by score key*

- typedef std::vector< std::pair < double, std::string > >::iterator EcScoresIt

  *evaporative cooling scores iterator - sorted by score key*

- typedef std::vector< std::pair < double, std::string > >::const_iterator EcScoresCIt

  *evaporative cooling scores constant iterator - sorted by score key*

## Enumerations

- enum EcAlgorithmType { EC_ALL, EC_RJ, EC_RF }

## Functions

- void libec_is_present (void)

  *HACK FOR AUTOTOOLS LIBRARY DETECTION.*

### 7.17.1 Typedef Documentation

#### 7.17.1.1 typedef std::vector<std::pair<double, std::string> > EcScores

evaporative cooling scores - sorted by score key

Definition at line 36 of file EvaporativeCooling.h.

#### 7.17.1.2 typedef std::vector<std::pair<double, std::string> >::const_iterator EcScoresCIt

evaporative cooling scores constant iterator - sorted by score key

Definition at line 40 of file EvaporativeCooling.h.

#### 7.17.1.3 typedef std::vector<std::pair<double, std::string> >::iterator EcScoresIt

evaporative cooling scores iterator - sorted by score key

Definition at line 38 of file EvaporativeCooling.h.

### 7.17.2 Enumeration Type Documentation

#### 7.17.2.1 enum EcAlgorithmType

Type of algorithm steps to perform.

**Enumerator:**

***EC_ALL*** Run RandomJungle and ReliefF.

***EC_RJ*** Run only RandomJungle.

***EC_RF*** Run only ReliefF.

Definition at line 46 of file EvaporativeCooling.h.

### 7.17.3 Function Documentation

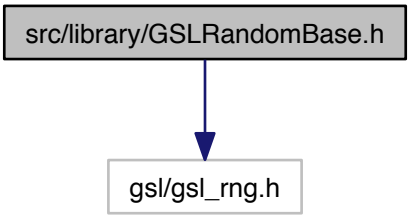
#### 7.17.3.1 void libec_is_present ( void )

HACK FOR AUTOTOOLS LIBRARY DETECTION.

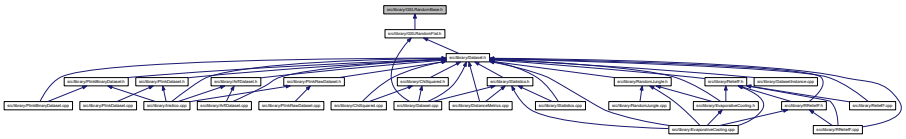
## 7.18 src/library/GSLRandomBase.h File Reference

```
#include "gsl/gsl_rng.h"
```

Include dependency graph for GSLRandomBase.h:




This graph shows which files directly or indirectly include this file:



### Classes

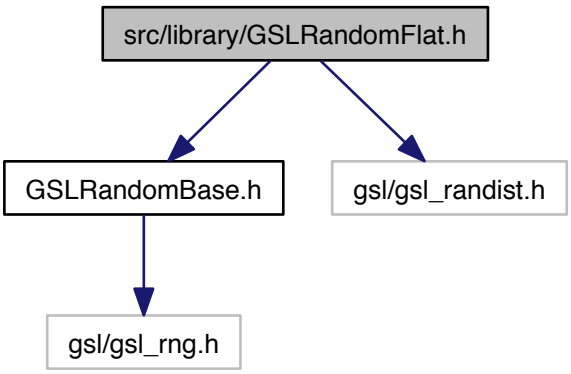
- class GSLRandomBase

  *A base class for GNU Scientific Library (GSL) random number functions.*

## 7.19 src/library/GSLRandomFlat.h File Reference

```
#include "GSLRandomBase.h"
#include "gsl/gsl_randist.h"
```

Include dependency graph for GSLRandomFlat.h:




This graph shows which files directly or indirectly include this file:

### Classes

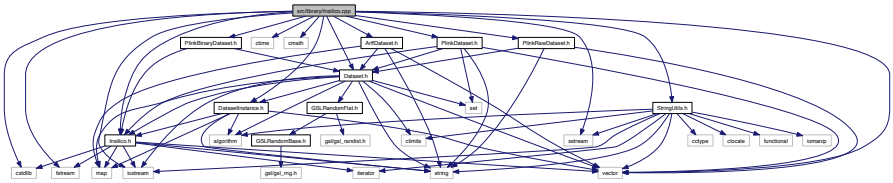
- class GSLRandomFlat

  *Random numbers in a flat, or uniform distribution.*

## 7.20 src/library/Insilico.cpp File Reference

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <vector>
#include <sstream>
#include <ctime>
#include <cmath>
#include "Dataset.h"
#include "DatasetInstance.h"
#include "ArffDataset.h"
#include "PlinkDataset.h"
#include "PlinkRawDataset.h"
#include "PlinkBinaryDataset.h"
#include "StringUtils.h"
#include "Insilico.h"
```

Include dependency graph for Insilico.cpp:



## Functions

- RandomJungleTreeType DetermineRandomJungleTreeType (AttributeType attributeType, ClassType classType)

  *Return random jungle tree type from the class and attribute types.*
- string Timestamp ()

  *Return a timestamp string for logging purposes.*
- Dataset * ChooseSnpsDatasetByExtension (string snpsFilename)
- bool LoadNumericIds (string filename, vector< string > &retIds)
- bool LoadPhenoIds (string filename, vector< string > &retIds)
- bool GetMatchingIds (string numericsFilename, string altPhenotypeFilename, vector< string > numericsIds, vector< string > phenoIds, vector< string > &matchingIds)
- ClassType DetectClassType (std::string filename, int classColumn, bool hasHeader)

  *Detect the class type by reading the specified column from a whitespace- delimited text file.*
- bool GetConfigValue (ConfigMap &configMap, std::string key, std::string &value)

  *Get the parameter value from the configuration map key.*
- string GetFileBasename (string fileName)
- string GetFileExtension (string fileName)
- double ProtectedLog (double x)

  *protected log function returns 0 for 0*

### 7.20.1 Function Documentation

#### 7.20.1.1 Dataset* ChooseSnpsDatasetByExtension ( string *snpsFilename* )

Definition at line 81 of file Insilico.cpp.

#### 7.20.1.2 ClassType DetectClassType ( std::string *filename,* int *classColumn,* bool *hasHeader* )

Detect the class type by reading the specified column from a whitespace- delimited text file.

**Parameters**

| | | |
|---|---|---|
| in | *filename* | whitespace-delimited text file name |
| in | *classColumn* | the column containing the class values |
| in | *heasHeader* | does the file have a header line? |

**Returns**

ClassType defined in Dataset.h

Open the file for reading

Skip the header if it has one

Determine the phenotype type

Definition at line 275 of file Insilico.cpp.

#### 7.20.1.3 RandomJungleTreeType DetermineRandomJungleTreeType ( AttributeType *attributeType,* ClassType *classType* )

Return random jungle tree type from the class and attribute types.

**Parameters**

| | | |
|---|---|---|
| in | *attributeType* | attribute data type |
| in | *classType* | class data type |

**Returns**

Random Jungle tree type

Definition at line 31 of file Insilico.cpp.

#### 7.20.1.4 bool GetConfigValue ( ConfigMap & *configMap,* std::string *key,* std::string & *value* )

Get the parameter value from the configuration map key.

**Parameters**

| | | |
|---|---|---|
| in | *configMap* | reference to a configuration map |
| in | *key* | parameter name |
| out | *parameter* | value |

**Returns**

true if key found, false if not found

Definition at line 342 of file Insilico.cpp.

#### 7.20.1.5 string GetFileBasename ( string *fileName* )

Definition at line 354 of file Insilico.cpp.

#### 7.20.1.6 string GetFileExtension ( string *fileName* )

Definition at line 359 of file Insilico.cpp.

#### 7.20.1.7 bool GetMatchingIds ( string *numericsFilename,* string *altPhenotypeFilename,* vector< string > *numericsIds,* vector< string > *phenoIds,* vector< string > & *matchingIds* )

Definition at line 221 of file Insilico.cpp.

#### 7.20.1.8 bool LoadNumericIds ( string *filename,* vector< string > & *retIds* )

Definition at line 120 of file Insilico.cpp.

#### 7.20.1.9 bool LoadPhenolds ( string *filename,* vector< string > & *retIds* )

Definition at line 172 of file Insilico.cpp.

#### 7.20.1.10 double ProtectedLog ( double *x* )

protected log function returns 0 for 0

Definition at line 364 of file Insilico.cpp.

#### 7.20.1.11 string Timestamp ( )

Return a timestamp string for logging purposes.

**Returns**

fixed-length, formatted timestamp as a string

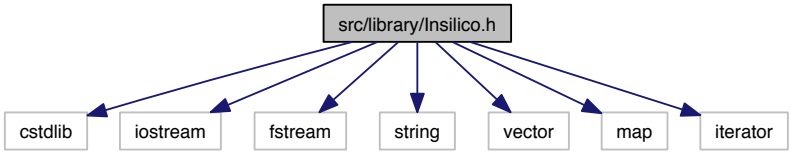
Definition at line 69 of file Insilico.cpp.

## 7.21 src/library/Insilico.h File Reference

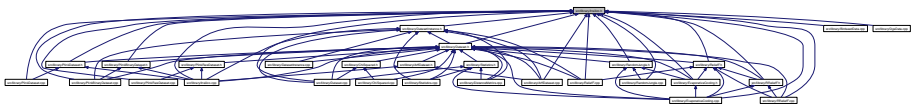
Common functions for Insilico Lab projects.

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <map>
#include <iterator>
```

Include dependency graph for Insilico.h:




This graph shows which files directly or indirectly include this file:



### Typedefs

- typedef int AttributeLevel

*T Y P E D E F S.*

- typedef double NumericLevel

  *type of continuous attributes*

- typedef int ClassLevel

  *type of instance class labels*

- typedef std::pair< double, std::string > DistancePair

  *distance pair type: distance, instance ID*

- typedef std::vector< DistancePair > DistancePairs

  *vector of distance pairs represents distances to nearest neighbors*

- typedef DistancePairs::const_iterator DistancePairsIt

  *distance pairs iterator*

- typedef std::map< std::string, std::string > ConfigMap

  *Configuration map as an alternative to Boost::program_options.*

## Enumerations

- enum OutputDatasetType { TAB_DELIMITED_DATASET, CSV_DELIMITED_DATASET, ARFF_DATASET, PLINK_PED_DATASET, PLINK_BED_DATASET, NO_OUTPUT_DATASET }

  *E N U M S.*

- enum AnalysisType { SNP_ONLY_ANALYSIS, NUMERIC_ONLY_ANALYSIS, INTEGRATED_ANALYSIS, DIAGNOSTIC_ANALYSIS, REGRESSION_ANALYSIS, DGE_ANALYSIS, BIRDSEED_ANALYSIS, DISTANCE_MATRIX_ANALYSIS, DATASET_CONVERSION, NO_ANALYSIS }

- enum ValueType { NUMERIC_VALUE, DISCRETE_VALUE, MISSING_VALUE, NO_VALUE }

- enum AttributeType { NUMERIC_TYPE, DISCRETE_TYPE, NO_TYPE }

- enum ClassType { CONTINUOUS_CLASS_TYPE, CASE_CONTROL_CLASS_TYPE, MULTI_CLASS_TYPE, NO_CLASS_TYPE }

- enum AttributeMutationType { TRANSITION_MUTATION, TRANSVERSION_MUTATION, UNKNOWN_MUTATION }

- enum RandomJungleTreeType { UNKNOWN_TREE_TYPE = 0, NOMINAL_NUMERIC_TREE, NOMINAL_NOMINAL_TREE, NUMERIC_NUMERIC_TREE, NUMERIC_NOMINAL_TREE, NOMINAL_NUMERIC_FLOATS }

- enum RandomJungleRunMode { UNKNOWN_RUN_MODE, LIBRARY_RUN_MODE, SYSTEM_CALL_RUN_MODE }

## Functions

- RandomJungleTreeType DetermineRandomJungleTreeType (AttributeType attributeType, ClassType classType)

  *Return random jungle tree type from the class and attribute types.*
- std::string Timestamp ()

  *Return a timestamp string for logging purposes.*
- Dataset * ChooseSnpsDatasetByExtension (std::string snpsFilename)

  *Determines the data set type to instantiate based on the data set filenames's extension.*
- bool LoadNumericIds (std::string filename, std::vector< std::string > &retIds)

  *Loads the individual (instance) IDs from the numerics file.*
- bool LoadPhenoIds (std::string filename, std::vector< std::string > &retIds)

  *Loads the individual (instance) IDs from the numerics file.*
- bool GetMatchingIds (std::string numericsFilename, std::string altPhenotypeFilename, std::vector< std::string > numericsIds, std::vector< std::string > phenoIds, std::vector< std::string > &matchingIds)

  *Return matching IDs from numeric and/or phenotype file IDs.*
- ClassType DetectClassType (std::string filename, int classColumn, bool hasHeader)

  *Detect the class type by reading the specified column from a whitespace- delimited text file.*
- bool GetConfigValue (ConfigMap &configMap, std::string key, std::string &value)

  *Get the parameter value from the configuration map key.*
- std::string GetFileBasename (std::string fullFilename)

  *Get the full filename without the extension.*
- std::string GetFileExtension (std::string fullFilename)

  *Get the filename extension.*
- template<class T >
  void PrintVector (std::vector< T > vec, std::string title="")

  *Print a vector of T values with optional title.*
- double ProtectedLog (double x)

  *protected log function returns 0 for 0*

## Variables

- static const int COMMAND_LINE_ERROR = EXIT_FAILURE

  *C O N S T A N T S.*
- static const int DATASET_LOAD_ERROR = EXIT_FAILURE
- static const int INVALID_DISTANCE = INT_MAX

  *return value for invalid distance*
- static const int INVALID_INDEX = INT_MAX

  *return value for invalid index into attributes*
- static const unsigned int INVALID_INT_VALUE = UINT_MAX

  *return value for invalid index into attributes*
- static const AttributeLevel INVALID_ATTRIBUTE_VALUE = INT_MIN

  *invalid attribute value*
- static const NumericLevel INVALID_NUMERIC_VALUE = INT_MIN

  *invalid attribute value*
- static const ClassLevel INVALID_DISCRETE_CLASS_VALUE = INT_MIN

  *stored value for missing discrete class*
- static const NumericLevel INVALID_NUMERIC_CLASS_VALUE = INT_MIN

  *stored value for missing numeric class*
- static const AttributeLevel MISSING_ATTRIBUTE_VALUE = -9

  *stored value for missing discrete attribute*

- static const NumericLevel MISSING_NUMERIC_VALUE = -9
  *stored value for missing numeric attribute*
- static const ClassLevel MISSING_DISCRETE_CLASS_VALUE = -9
  *stored value for missing discrete class*
- static const NumericLevel MISSING_NUMERIC_CLASS_VALUE = -9
  *stored value for missing numeric class*

### 7.21.1 Detailed Description

Common functions for Insilico Lab projects.

**Author**

: Bill White

**Version**

1.0

Contact: bill.c.white@gmail.com Created on 10/13/11

Definition in file Insilico.h.

### 7.21.2 Typedef Documentation

#### 7.21.2.1 typedef int AttributeLevel

T Y P E D E F S.

type of discrete attribute values

Definition at line 24 of file Insilico.h.

#### 7.21.2.2 typedef int ClassLevel

type of instance class labels

Definition at line 33 of file Insilico.h.

#### 7.21.2.3 typedef std::map<std::string, std::string> ConfigMap

Configuration map as an alternative to Boost::program_options.

Definition at line 43 of file Insilico.h.

#### 7.21.2.4 typedef std::pair<double, std::string> DistancePair

distance pair type: distance, instance ID

Definition at line 36 of file Insilico.h.

#### 7.21.2.5 typedef std::vector<DistancePair> DistancePairs

vector of distance pairs represents distances to nearest neighbors

Definition at line 38 of file Insilico.h.

#### 7.21.2.6 typedef DistancePairs::const_iterator DistancePairsIt

distance pairs iterator

Definition at line 40 of file Insilico.h.

#### 7.21.2.7 typedef double NumericLevel

type of continuous attributes

Definition at line 31 of file Insilico.h.

### 7.21.3 Enumeration Type Documentation

#### 7.21.3.1 enum AnalysisType

Type of analysis to perform.

**Enumerator:**

***SNP_ONLY_ANALYSIS*** discrete analysis
***NUMERIC_ONLY_ANALYSIS*** continuous attributes
***INTEGRATED_ANALYSIS*** discrete and continuous analysis
***DIAGNOSTIC_ANALYSIS*** diagnostic mode - no ReliefF analysis
***REGRESSION_ANALYSIS*** regression ReliefF analysis
***DGE_ANALYSIS*** digital gene expression (DGE) analysis
***BIRDSEED_ANALYSIS*** Birdseed called SNPs analysis.
***DISTANCE_MATRIX_ANALYSIS*** distance matrix calculation
***DATASET_CONVERSION*** convert data set format types
***NO_ANALYSIS*** no analysis specified

Definition at line 95 of file Insilico.h.

#### 7.21.3.2 enum AttributeMutationType

Type of attribute mutation.

**Enumerator:**

***TRANSITION_MUTATION*** transition within family
***TRANSVERSION_MUTATION*** transversion between families
***UNKNOWN_MUTATION*** unknown - no allele information

Definition at line 148 of file Insilico.h.

#### 7.21.3.3 enum AttributeType

Type of attributes that are stored in data set instances.

**Enumerator:**

***NUMERIC_TYPE*** continuous numeric type
***DISCRETE_TYPE*** discrete genotype type
***NO_TYPE*** default no type

Definition at line 125 of file Insilico.h.

#### 7.21.3.4 enum ClassType

Type of classes that are stored in data set instances.

**Enumerator:**

***CONTINUOUS_CLASS_TYPE*** continuous numeric type
***CASE_CONTROL_CLASS_TYPE*** discrete case-control type
***MULTI_CLASS_TYPE*** multiclass type
***NO_CLASS_TYPE*** default no type

Definition at line 136 of file Insilico.h.

#### 7.21.3.5 enum OutputDatasetType

E N U M S.

Type of data set to write filtered output.

**Enumerator:**

***TAB_DELIMITED_DATASET*** tab-delimited .txt file
***CSV_DELIMITED_DATASET*** comma separated values .csv file
***ARFF_DATASET*** WEKA ARFF format .arff file.
***PLINK_PED_DATASET*** PLINK ped/map format.
***PLINK_BED_DATASET*** PLINK bed/bim/fam format.
***NO_OUTPUT_DATASET*** no output data set specified

Definition at line 82 of file Insilico.h.

#### 7.21.3.6 enum RandomJungleRunMode

Run mode for random jungle.

**Enumerator:**

***UNKNOWN_RUN_MODE*** unknown run mode
***LIBRARY_RUN_MODE*** call Random Jungle through C++ library calls
***SYSTEM_CALL_RUN_MODE*** call Random Jungle through C system() call

Definition at line 173 of file Insilico.h.

#### 7.21.3.7 enum RandomJungleTreeType

Type random jungle trees.

**Enumerator:**

***UNKNOWN_TREE_TYPE*** place holder = 0
***NOMINAL_NUMERIC_TREE*** classification trees, numeric attributes (integers)
***NOMINAL_NOMINAL_TREE*** classification trees, discrete attributes (0/1/2)
***NUMERIC_NUMERIC_TREE*** regression trees, numeric attributes (doubles)
***NUMERIC_NOMINAL_TREE*** regression trees, discrete attributes (0/1/2)
***NOMINAL_NUMERIC_FLOATS*** classification trees, numeric attributes (doubles)

Definition at line 159 of file Insilico.h.

#### 7.21.3.8 enum ValueType

Return types for determing a value's type.

**Enumerator:**

***NUMERIC_VALUE*** continuous numeric value

***DISCRETE_VALUE*** discrete genotype value

***MISSING_VALUE*** missing value

***NO_VALUE*** default no value type

Definition at line 113 of file Insilico.h.

### 7.21.4 Function Documentation

#### 7.21.4.1 Dataset* ChooseSnpsDatasetByExtension ( std::string *snpsFilename* )

Determines the data set type to instantiate based on the data set filenames's extension.

**Parameters**

| | | |
|---|---|---|
| in | *snpsFilename* | SNP data set filename |

**Returns**

pointer to new dataset or NULL if could not match filename extension

#### 7.21.4.2 ClassType DetectClassType ( std::string *filename,* int *classColumn,* bool *hasHeader* )

Detect the class type by reading the specified column from a whitespace- delimited text file.

**Parameters**

| | | |
|---|---|---|
| in | *filename* | whitespace-delimited text file name |
| in | *classColumn* | the column containing the class values |
| in | *heasHeader* | does the file have a header line? |

**Returns**

ClassType defined in Dataset.h

Open the file for reading

Skip the header if it has one

Determine the phenotype type

Definition at line 275 of file Insilico.cpp.

#### 7.21.4.3 RandomJungleTreeType DetermineRandomJungleTreeType ( AttributeType *attributeType,* ClassType *classType* )

Return random jungle tree type from the class and attribute types.

**Parameters**

| | | |
|---|---|---|
| in | *attributeType* | attribute data type |
| in | *classType* | class data type |

**Returns**

Random Jungle tree type

Definition at line 31 of file Insilico.cpp.

#### 7.21.4.4 bool GetConfigValue ( ConfigMap & *configMap,* std::string *key,* std::string & *value* )

Get the parameter value from the configuration map key.

**Parameters**

| | | |
|---|---|---|
| in | *configMap* | reference to a configuration map |
| in | *key* | parameter name |
| out | *parameter* | value |

**Returns**

true if key found, false if not found

Definition at line 342 of file Insilico.cpp.

#### 7.21.4.5 std::string GetFileBasename ( std::string *fullFilename* )

Get the full filename without the extension.

**Parameters**

| | | |
|---|---|---|
| in | *fullFilename* | complete filename |

**Returns**

path/filename without extension

#### 7.21.4.6 std::string GetFileExtension ( std::string *fullFilename* )

Get the filename extension.

**Parameters**

| | | |
|---|---|---|
| in | *fullFilename* | complete filename |

**Returns**

filename extension

#### 7.21.4.7 bool GetMatchingIds ( std::string *numericsFilename,* std::string *altPhenotypeFilename,* std::vector< std::string > *numericsIds,* std::vector< std::string > *phenoIds,* std::vector< std::string > & *matchingIds* )

Return matching IDs from numeric and/or phenotype file IDs.

**Parameters**

| | | |
|---|---|---|
| in | *numerics-Filename* | name of the PLINK covar format file |
| in | *altPhenotype-Filename* | name of the alternate pheno file PLINK |
| in | *numericsIds* | covar format file ids |
| in | *phenoIds* | alternate phenotype file ids |
| out | *matchingIds* | ids that match between numerics and phenotypes |

**Returns**

success

#### 7.21.4.8 bool LoadNumericIds ( std::string *filename,* std::vector< std::string > & *retIds* )

Loads the individual (instance) IDs from the numerics file.

Returns the IDs through reference parameter retIds.

**Parameters**

| | | |
|---|---|---|
| in | *filename* | filename that contains numerics IDs |
| out | *vector* | of individual (instance) IDs (strings) |

**Returns**

success

#### 7.21.4.9 bool LoadPhenoIds ( std::string *filename,* std::vector< std::string > & *retIds* )

Loads the individual (instance) IDs from the numerics file.

Returns the IDs through reference parameter retIds.

**Parameters**

| | | |
|---|---|---|
| in | *filename* | filename that contains numerics IDs |
| out | *vector* | of individual (instance) IDs (strings) |

**Returns**

success

#### 7.21.4.10 template<class T > void PrintVector ( std::vector< T > *vec,* std::string *title* = " " )

Print a vector of T values with optional title.

**Parameters**

| | | |
|---|---|---|
| in | *vec* | vector of T type values |
| in | *title* | optional title to print before the vector |

Definition at line 267 of file Insilico.h.

#### 7.21.4.11 double ProtectedLog ( double *x* )

protected log function returns 0 for 0

Definition at line 364 of file Insilico.cpp.

#### 7.21.4.12 std::string Timestamp ( )

Return a timestamp string for logging purposes.

**Returns**

fixed-length, formatted timestamp as a string

Definition at line 69 of file Insilico.cpp.

### 7.21.5 Variable Documentation

#### 7.21.5.1 const int COMMAND_LINE_ERROR = EXIT_FAILURE [static]

C O N S T A N T S.

Error codes.

Definition at line 48 of file Insilico.h.

#### 7.21.5.2 const int DATASET_LOAD_ERROR = EXIT_FAILURE [static]

Definition at line 49 of file Insilico.h.

#### 7.21.5.3 const AttributeLevel INVALID_ATTRIBUTE_VALUE = INT_MIN [static]

invalid attribute value

Definition at line 59 of file Insilico.h.

#### 7.21.5.4 const ClassLevel INVALID_DISCRETE_CLASS_VALUE = INT_MIN [static]

stored value for missing discrete class

Definition at line 63 of file Insilico.h.

#### 7.21.5.5 const int INVALID_DISTANCE = INT_MAX [static]

return value for invalid distance

Definition at line 52 of file Insilico.h.

#### 7.21.5.6 const int INVALID_INDEX = INT_MAX [static]

return value for invalid index into attributes

Definition at line 54 of file Insilico.h.

#### 7.21.5.7 const unsigned int INVALID_INT_VALUE = UINT_MAX [static]

return value for invalid index into attributes

Definition at line 56 of file Insilico.h.

#### 7.21.5.8 const NumericLevel INVALID_NUMERIC_CLASS_VALUE = INT_MIN [static]

stored value for missing numeric class

Definition at line 65 of file Insilico.h.

#### 7.21.5.9 const NumericLevel INVALID_NUMERIC_VALUE = INT_MIN [static]

invalid attribute value

Definition at line 61 of file Insilico.h.

#### 7.21.5.10 const AttributeLevel MISSING_ATTRIBUTE_VALUE = -9 [static]

stored value for missing discrete attribute

Definition at line 68 of file Insilico.h.

#### 7.21.5.11 const ClassLevel MISSING_DISCRETE_CLASS_VALUE = -9 [static]

stored value for missing discrete class

Definition at line 72 of file Insilico.h.

#### 7.21.5.12 const NumericLevel MISSING_NUMERIC_CLASS_VALUE = -9 [static]

stored value for missing numeric class

Definition at line 74 of file Insilico.h.

#### 7.21.5.13 const NumericLevel MISSING_NUMERIC_VALUE = -9 [static]

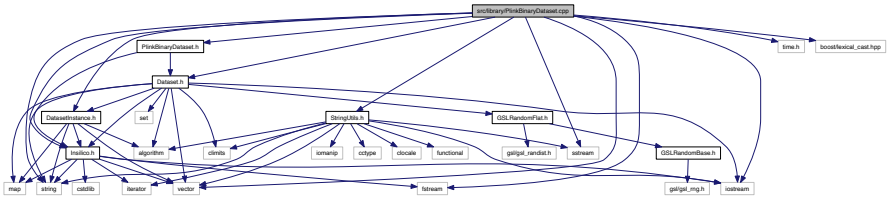
stored value for missing numeric attribute

Definition at line 70 of file Insilico.h.

## 7.22 src/library/PlinkBinaryDataset.cpp File Reference

```
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <time.h>
#include <sstream>
#include <boost/lexical_cast.hpp>
#include "Dataset.h"
#include "DatasetInstance.h"
#include "StringUtils.h"
#include "PlinkBinaryDataset.h"
#include "Insilico.h"
```

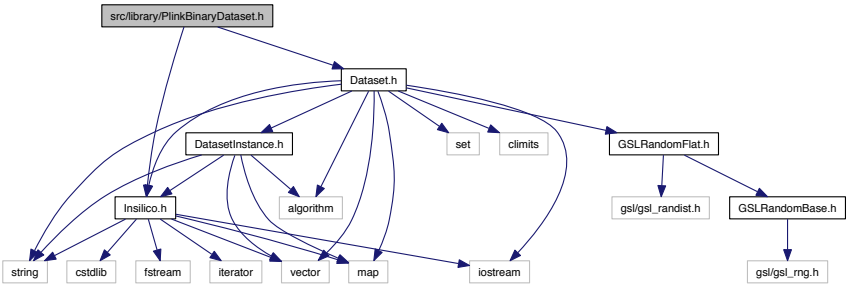
Include dependency graph for PlinkBinaryDataset.cpp:



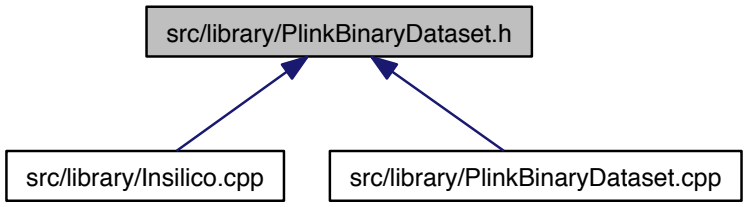

## 7.23 src/library/PlinkBinaryDataset.h File Reference

```
#include "Dataset.h"
#include "Insilico.h"
```

Include dependency graph for PlinkBinaryDataset.h:




This graph shows which files directly or indirectly include this file:




### Classes

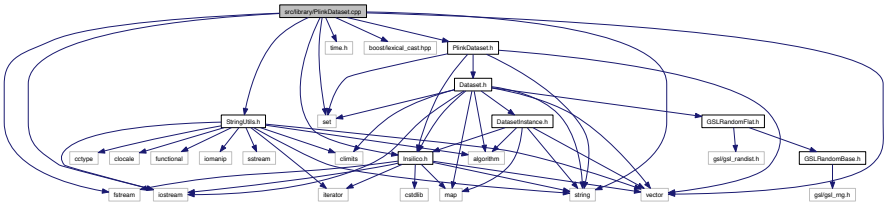
- class PlinkBinaryDataset

  *Plink binary PED/BED file format reader.*

## 7.24 src/library/PlinkDataset.cpp File Reference

```
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <set>
#include <time.h>
#include <boost/lexical_cast.hpp>
#include "StringUtils.h"
#include "PlinkDataset.h"
#include "Insilico.h"
```

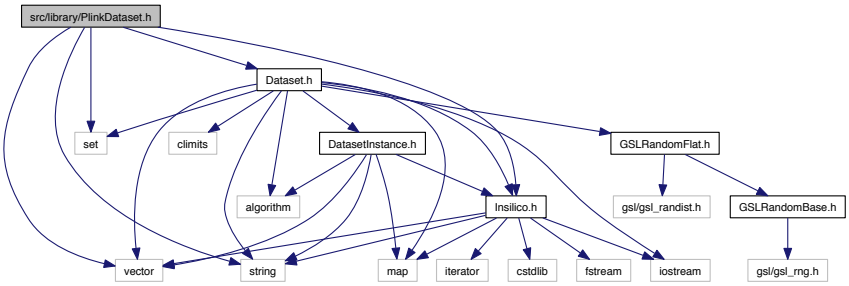
Include dependency graph for PlinkDataset.cpp:



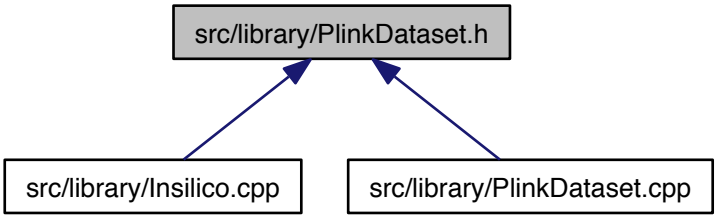

## 7.25 src/library/PlinkDataset.h File Reference

```
#include <set>
#include <vector>
#include <string>
#include "Dataset.h"
#include "Insilico.h"
```

Include dependency graph for PlinkDataset.h:



This graph shows which files directly or indirectly include this file:




### Classes

- class PlinkDataset

  *Plink MAP/PED file format reader.*

### Enumerations

- enum MapFileType { MAP3_FILE, MAP4_FILE, ERROR_FILE }

#### 7.25.1 Enumeration Type Documentation

##### 7.25.1.1 enum MapFileType

PLINK map file types.

**Enumerator:**

***MAP3_FILE*** map 3 simplified format

***MAP4_FILE*** map 4 standard format

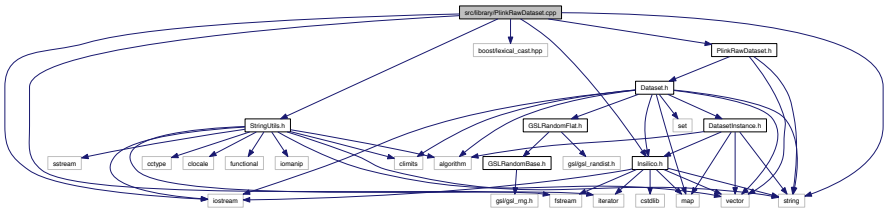
***ERROR_FILE*** default

Definition at line 29 of file PlinkDataset.h.

## 7.26 src/library/PlinkRawDataset.cpp File Reference

```
#include <string>
#include <iostream>
#include <fstream>
#include <boost/lexical_cast.hpp>
#include "StringUtils.h"
#include "PlinkRawDataset.h"
#include "Insilico.h"
```

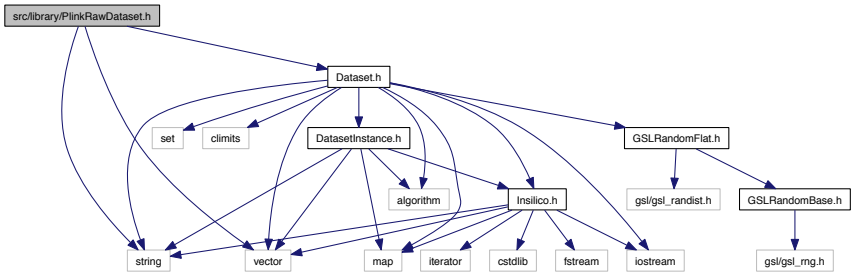
Include dependency graph for PlinkRawDataset.cpp:



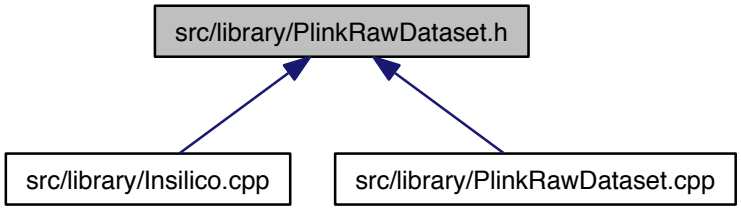

## 7.27 src/library/PlinkRawDataset.h File Reference

```
#include <string>
#include <vector>
#include "Dataset.h"
```

Include dependency graph for PlinkRawDataset.h:




This graph shows which files directly or indirectly include this file:




### Classes

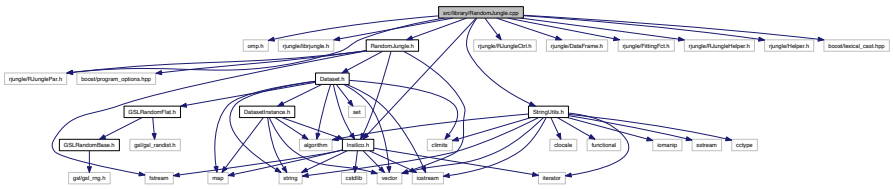
- class PlinkRawDataset

  *Plink recodeA/RAW file format reader.*

## 7.28 src/library/RandomJungle.cpp File Reference

```
#include <omp.h>
#include "rjungle/librjungle.h"
#include "rjungle/RJunglePar.h"
#include "rjungle/RJungleCtrl.h"
#include "rjungle/DataFrame.h"
#include "rjungle/FittingFct.h"
#include "rjungle/RJungleHelper.h"
#include "rjungle/Helper.h"
#include "boost/lexical_cast.hpp"
#include "RandomJungle.h"
#include "StringUtils.h"
#include "Insilico.h"
```

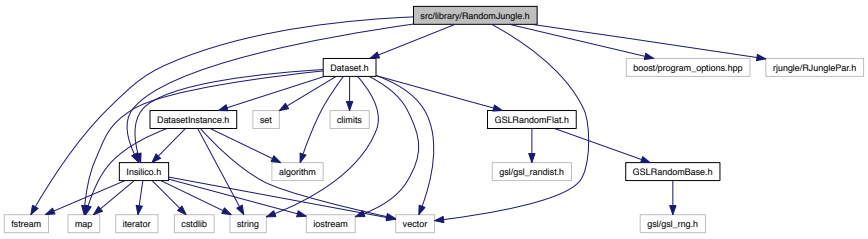
Include dependency graph for RandomJungle.cpp:



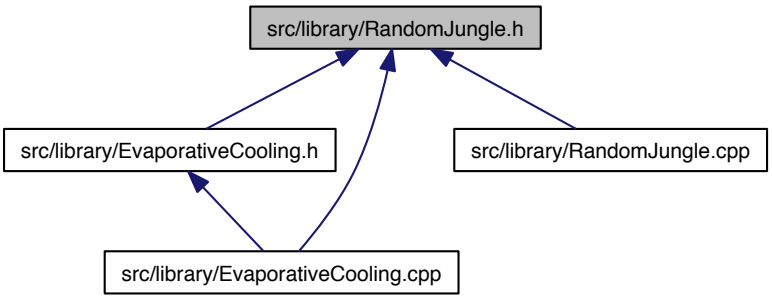

## 7.29 src/library/RandomJungle.h File Reference

```
#include <vector>
#include <fstream>
#include "Insilico.h"
#include "Dataset.h"
#include <boost/program_options.hpp>
#include "rjungle/RJunglePar.h"
```

Include dependency graph for RandomJungle.h:



This graph shows which files directly or indirectly include this file:




### Classes

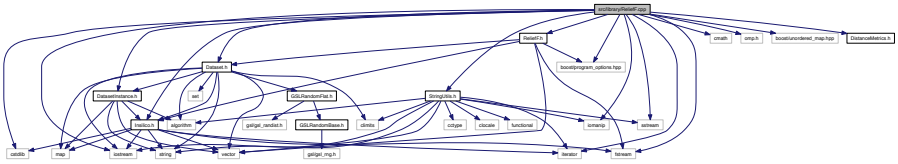
- class RandomJungle

  *RandomJungle attribute ranking algorithm.*

## 7.30 src/library/ReliefF.cpp File Reference

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <iomanip>
#include <iterator>
#include <cmath>
#include <sstream>
#include <omp.h>
#include <boost/program_options.hpp>
#include <boost/unordered_map.hpp>
#include "ReliefF.h"
#include "Dataset.h"
#include "DatasetInstance.h"
#include "StringUtils.h"
#include "DistanceMetrics.h"
#include "Insilico.h"
```

Include dependency graph for ReliefF.cpp:



### Classes

- class deref_less

### Typedefs

- typedef vector< pair< double, unsigned int > > ScoresMap

  *scores map: score->attribute index*
- typedef vector< pair< double, unsigned int > >::iterator ScoresMapIt

  *scores map iterator*
- typedef vector< pair< unsigned int, double > > AttributeIndex

  *attribute index map: attribute index->score*
- typedef vector< pair< unsigned int, double > >::const_iterator AttributeIndexIt

  *attribute index map iterator*
- typedef pair< unsigned int, DatasetInstance * > T

  *functor for T comparison*

### Functions

- bool scoreSort (const pair< double, string > &p1, const pair< double, string > &p2)

  *attribute score sorting functor*
- bool attributeSort (const pair< unsigned int, double > &p1, const pair< unsigned int, double > &p2)

  *attribute index sorting functor*
- void librelieff_is_present (void)

## 7.30.1 Typedef Documentation

### 7.30.1.1 typedef vector<pair<unsigned int, double> > AttributeIndex

attribute index map: attribute index->score

Definition at line 40 of file ReliefF.cpp.

### 7.30.1.2 typedef vector<pair<unsigned int, double> >::const_iterator AttributeIndexIt

attribute index map iterator

Definition at line 42 of file ReliefF.cpp.

### 7.30.1.3 typedef vector<pair<double, unsigned int> > ScoresMap

scores map: score->attribute index

Definition at line 36 of file ReliefF.cpp.

### 7.30.1.4 typedef vector<pair<double, unsigned int> >::iterator ScoresMapIt

scores map iterator

Definition at line 38 of file ReliefF.cpp.

#### 7.30.1.5 typedef pair<unsigned int, DatasetInstance*> T

functor for T comparison

Definition at line 57 of file ReliefF.cpp.

### 7.30.2 Function Documentation

#### 7.30.2.1 bool attributeSort ( const pair< unsigned int, double > & *p1*, const pair< unsigned int, double > & *p2* )

attribute index sorting functor

Definition at line 51 of file ReliefF.cpp.

#### 7.30.2.2 void librelieff_is_present ( void )

Definition at line 1006 of file ReliefF.cpp.

#### 7.30.2.3 bool scoreSort ( const pair< double, string > & *p1*, const pair< double, string > & *p2* )

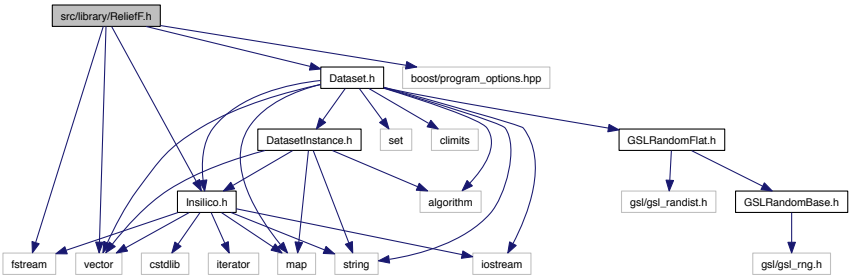
attribute score sorting functor

Definition at line 45 of file ReliefF.cpp.

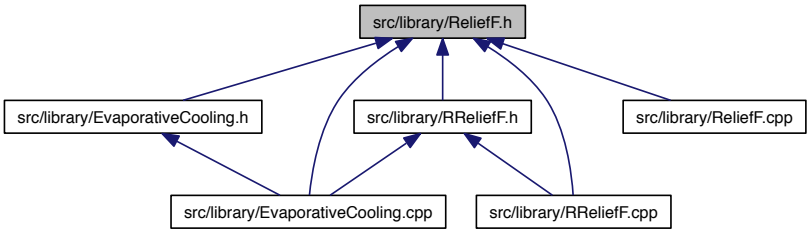
## 7.31 src/library/ReliefF.h File Reference

```
#include <vector>
#include <fstream>
#include <boost/program_options.hpp>
#include "Dataset.h"
#include "Insilico.h"
```

Include dependency graph for ReliefF.h:



This graph shows which files directly or indirectly include this file:




### Classes

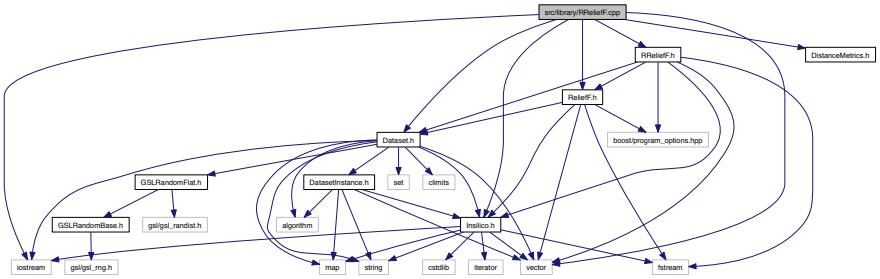
- class ReliefF

  *ReliefF attribute ranking algorithm.*

## 7.32 src/library/RReliefF.cpp File Reference

```
#include <iostream>
#include <vector>
#include "ReliefF.h"
#include "RReliefF.h"
#include "Dataset.h"
#include "DistanceMetrics.h"
#include "Insilico.h"
```

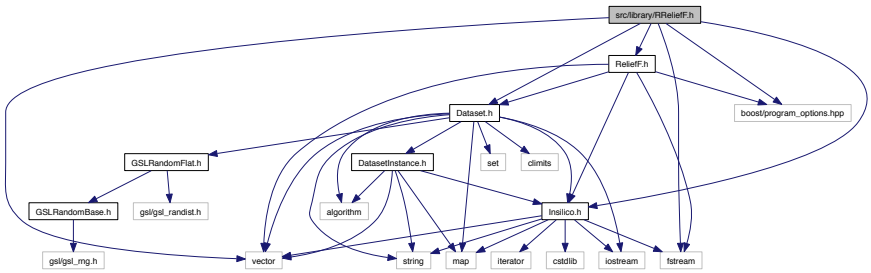
Include dependency graph for RReliefF.cpp:



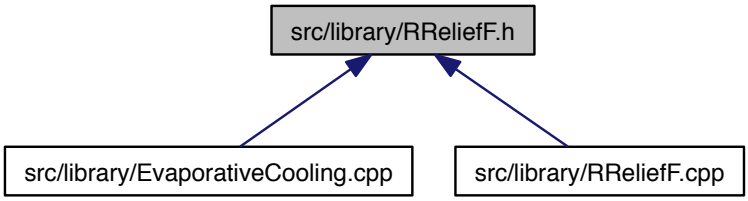

## 7.33 src/library/RReliefF.h File Reference

```
#include <vector>
#include <fstream>
#include "ReliefF.h"
#include "Dataset.h"
#include "Insilico.h"
#include <boost/program_options.hpp>
```

Include dependency graph for RReliefF.h:




This graph shows which files directly or indirectly include this file:




### Classes

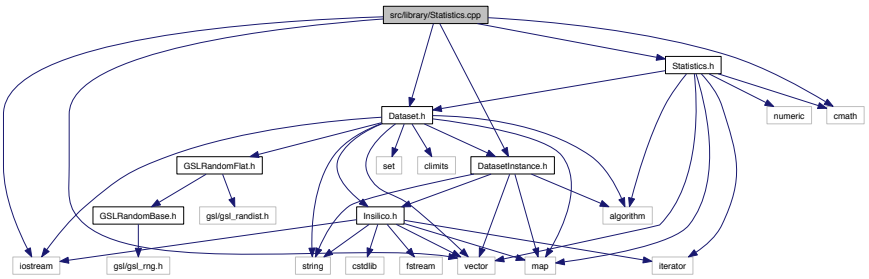
- class RReliefF

  *Regression ReliefF attribute ranking algorithm.*

## 7.34 src/library/Statistics.cpp File Reference

```
#include <iostream>
#include <vector>
#include <cmath>
#include "Dataset.h"
#include "DatasetInstance.h"
#include "Statistics.h"
```

Include dependency graph for Statistics.cpp:




## Defines

- #define DEBUG_Z 0
- #define DEBUG_E 1

## Functions

- void PrintHistogram (Histogram histogram)

  *Print a Histogram to cout.*
- bool ZTransform (const VectorDouble &inputValues, VectorDouble &outputValues)

  *ZTransform input values.*
- double SelfEntropy (const vector< AttributeLevel > &a, const vector< AttributeLevel > &c)
- double Entropy (const vector< AttributeLevel > &sequenceValues)
- double condentropy (const vector< AttributeLevel > &X, const vector< AttributeLevel > &Y)
- double ConditionalEntropy (const vector< AttributeLevel > &sequenceValues, const vector< AttributeLevel > &givenValues)
- bool ConstructAttributeCart (const vector< AttributeLevel > &a, const vector< AttributeLevel > &b, vector< AttributeLevel > &ab)
- double KendallTau (vector< string > X, vector< string > Y)
- double KendallTau (vector< double > X, vector< double > Y)
- double KendallTau (vector< int > X, vector< int > Y)

### 7.34.1 Define Documentation

#### 7.34.1.1 #define DEBUG_E 1

Definition at line 18 of file Statistics.cpp.

#### 7.34.1.2 #define DEBUG_Z 0

Definition at line 17 of file Statistics.cpp.

### 7.34.2 Function Documentation

#### 7.34.2.1 double condentropy ( const vector< AttributeLevel > & *X*, const vector< AttributeLevel > & *Y* )

Definition at line 118 of file Statistics.cpp.

#### 7.34.2.2 double ConditionalEntropy ( const vector< AttributeLevel > & *sequenceValues,* const vector< AttributeLevel > & *givenValues* )

convert from base e to base 2

Definition at line 127 of file Statistics.cpp.

#### 7.34.2.3 bool ConstructAttributeCart ( const vector< AttributeLevel > & *a,* const vector< AttributeLevel > & *b,* vector< AttributeLevel > & *ab* )

Get the number of levels in a for a multiplier

Definition at line 208 of file Statistics.cpp.

#### 7.34.2.4 double Entropy ( const vector< AttributeLevel > & *sequenceValues* )

Definition at line 93 of file Statistics.cpp.

#### 7.34.2.5 double KendallTau ( vector< string > *X,* vector< string > *Y* )

Definition at line 248 of file Statistics.cpp.

#### 7.34.2.6 double KendallTau ( vector< double > *X,* vector< double > *Y* )

Definition at line 284 of file Statistics.cpp.

#### 7.34.2.7 double KendallTau ( vector< int > *X,* vector< int > *Y* )

Definition at line 318 of file Statistics.cpp.

#### 7.34.2.8 void PrintHistogram ( Histogram *histogram* )

Print a Histogram to cout.

**Parameters**

| | | |
|---|---|---|
| in | *histogram* | Histogram to print |

Definition at line 20 of file Statistics.cpp.

#### 7.34.2.9 double SelfEntropy ( const vector< AttributeLevel > & *a,* const vector< AttributeLevel > & *c* )

Definition at line 87 of file Statistics.cpp.

#### 7.34.2.10 bool ZTransform ( const VectorDouble & *inputValues,* VectorDouble & *outputValues* )

ZTransform input values.

**Parameters**

| | | |
|---|---|---|
| in | *inputValues* | const vector of double input values |
| out | *outputValues* | transformed input values to z-scores with mean=0, stddev=1 |

**Returns**

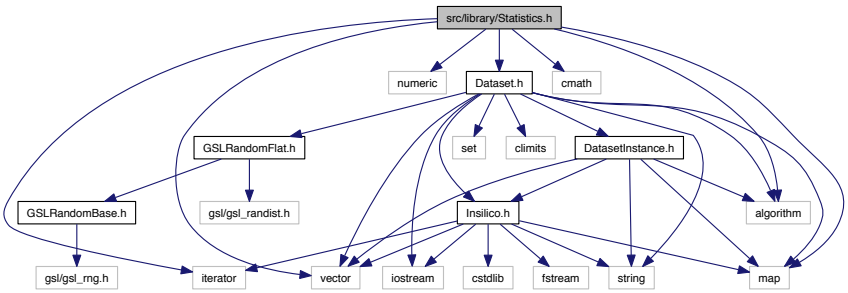
success

Definition at line 27 of file Statistics.cpp.

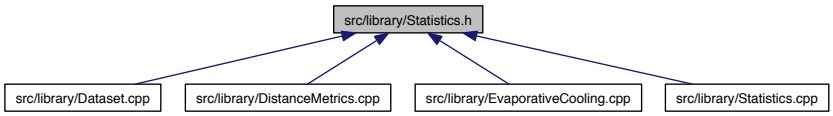
## 7.35 src/library/Statistics.h File Reference

```
#include <vector>
#include <map>
#include <numeric>
#include <iterator>
#include <cmath>
#include <algorithm>
#include "Dataset.h"
```

Include dependency graph for Statistics.h:




This graph shows which files directly or indirectly include this file:




### Typedefs

- typedef std::vector< double > VectorDouble

  *vector of doubles type*
- typedef std::vector< double > ::const_iterator VectorDoubleIt

  *vector of doubles iterator*
- typedef std::map < AttributeLevel, unsigned int > Histogram

  *histogram type is a map: value->count*
- typedef std::map < AttributeLevel, unsigned int > ::const_iterator HistogramIt

  *historgram iterator*

## Functions

- void PrintHistogram (Histogram histogram)

  *Print a Histogram to cout.*
- bool ZTransform (const VectorDouble &inputValues, VectorDouble &outputValues)

  *ZTransform input values.*
- double SelfEntropy (const std::vector< AttributeLevel > &a, const std::vector< AttributeLevel > &c)

  *Calculates the entropy of a sequence with itself and the class.*
- double Entropy (const std::vector< AttributeLevel > &attributeValues)

  *Calculates the entropy of a sequence of unsigned integers.*
- double ConditionalEntropy (const std::vector< AttributeLevel > &attributeValues, const std::vector< AttributeLevel > &givenValues)

  *Calculates the conditional entropy of a sequence of unsigned integers based (conditioned) on another sequence of unsigned integers (the givens).*
- double condentropy (const std::vector< AttributeLevel > &X, const std::vector< AttributeLevel > &Y)
- bool ConstructAttributeCart (const std::vector< AttributeLevel > &a, const std::vector< AttributeLevel > &b, std::vector< AttributeLevel > &ab)

  *Create a new attribute that is the cartesian product of a and b.*
- double KendallTau (std::vector< std::string > X, std::vector< std::string > Y)

  *Compute KendallTau for two ranked vectors of strings.*
- double KendallTau (std::vector< double > X, std::vector< double > Y)

  *Compute KendallTau for two ranked vectors of doubles.*
- double KendallTau (std::vector< int > X, std::vector< int > Y)

  *Compute KendallTau for two ranked vectors of integers.*
- template<class T >
  std::pair< double, double > VarStd (std::vector< T > &values)

  *Calculate variance and standard deviation of a vector of values.*

### 7.35.1 Typedef Documentation

#### 7.35.1.1 typedef std::map<AttributeLevel, unsigned int> Histogram

histogram type is a map: value->count

Definition at line 30 of file Statistics.h.

#### 7.35.1.2 typedef std::map<AttributeLevel, unsigned int>::const_iterator HistogramIt

historgram iterator

Definition at line 32 of file Statistics.h.

#### 7.35.1.3 typedef std::vector<double> VectorDouble

vector of doubles type

Definition at line 26 of file Statistics.h.

#### 7.35.1.4 typedef std::vector<double>::const_iterator VectorDoubleIt

vector of doubles iterator

Definition at line 28 of file Statistics.h.

### 7.35.2 Function Documentation

#### 7.35.2.1 double condentropy ( const std::vector< AttributeLevel > & *X*, const std::vector< AttributeLevel > & *Y* )

#### 7.35.2.2 double ConditionalEntropy ( const std::vector< AttributeLevel > & *attributeValues*, const std::vector< AttributeLevel > & *givenValues* )

Calculates the conditional entropy of a sequence of unsigned integers based (conditioned) on another sequence of unsigned integers (the givens).

P(sequenceValues | givenValues)

**Parameters**

| | | |
|---|---|---|
| in | *attributeValues* | vector of values |
| in | *givenValues* | vector of givens |

**Returns**

conditional entropy as a double-precision float

#### 7.35.2.3 bool ConstructAttributeCart ( const std::vector< AttributeLevel > & *a*, const std::vector< AttributeLevel > & *b*, std::vector< AttributeLevel > & *ab* )

Create a new attribute that is the cartesian product of a and b.

NOTE: works for genotypes; need to verify for missign data levels, etc.

**Parameters**

| | | |
|---|---|---|
| in | *a* | attributes vector a |
| in | *b* | attributes vector b |
| out | *vector* | ab, the cartesian product of a and b |

**Returns**

success

#### 7.35.2.4 double Entropy ( const std::vector< AttributeLevel > & *attributeValues* )

Calculates the entropy of a sequence of unsigned integers.

**Parameters**

| | | |
|---|---|---|
| in | *attributeValues* | vector of sequence values - unsigned ints - positive categorical |

**Returns**

entropy as a double-precision float

#### 7.35.2.5 double KendallTau ( std::vector< std::string > *X*, std::vector< std::string > *Y* )

Compute KendallTau for two ranked vectors of strings.

Why Kenall Tau - G. E. NOETHER http://www.rsscse-edu.org.uk/tsj/bts/noether/text.-html

**Parameters**

| | | |
|---|---|---|
| in | *X* | ranked attribute vector X |
| in | *Y* | ranked attribute vector Y |

**Returns**

Kendall Tau value (-1, 1)

#### 7.35.2.6 double KendallTau ( std::vector< double > *X*, std::vector< double > *Y* )

Compute KendallTau for two ranked vectors of doubles.

Why Kenall Tau - G. E. NOETHER http://www.rsscse-edu.org.uk/tsj/bts/noether/text.-html

**Parameters**

| | | |
|---|---|---|
| in | *X* | ranked attribute vector X |
| in | *Y* | ranked attribute vector Y |

**Returns**

Kendall Tau value (-1, 1)

#### 7.35.2.7 double KendallTau ( std::vector< int > *X*, std::vector< int > *Y* )

Compute KendallTau for two ranked vectors of integers.

Why Kenall Tau - G. E. NOETHER http://www.rsscse-edu.org.uk/tsj/bts/noether/text.-html

**Parameters**

| | | |
|---|---|---|
| in | *X* | ranked attribute vector X |
| in | *Y* | ranked attribute vector Y |

**Returns**

Kendall Tau value (-1, 1)

#### 7.35.2.8 void PrintHistogram ( Histogram *histogram* )

Print a Histogram to cout.

**Parameters**

| | | |
|---|---|---|
| in | *histogram* | Histogram to print |

Definition at line 20 of file Statistics.cpp.

#### 7.35.2.9 double SelfEntropy ( const std::vector< AttributeLevel > & *a*, const std::vector< AttributeLevel > & *c* )

Calculates the entropy of a sequence with itself and the class.

**Parameters**

| | | |
|---|---|---|
| in | *a* | vector of values |
| in | *c* | vector of class levels |

**Returns**

entropy as a double-precision float

#### 7.35.2.10 template<class T > std::pair<double, double> VarStd ( std::vector< T > & *values* )

Calculate variance and standard deviation of a vector of values.

**Parameters**

| | | |
|---|---|---|
| in | *ranked* | attribute lists X and Y |

**Returns**

Kendall Tau value (-1, 1)

Definition at line 116 of file Statistics.h.

#### 7.35.2.11 bool ZTransform ( const VectorDouble & *inputValues,* VectorDouble & *outputValues* )

ZTransform input values.

**Parameters**

| | | |
|---|---|---|
| in | *inputValues* | const vector of double input values |
| out | *outputValues* | transformed input values to z-scores with mean=0, stddev=1 |

**Returns**

success

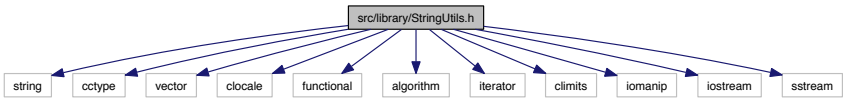
Definition at line 27 of file Statistics.cpp.

## 7.36 src/library/StringUtils.h File Reference

Various string-related utilities.

```
#include <string>
#include <cctype>
#include <vector>
#include <clocale>
#include <functional>
#include <algorithm>
#include <iterator>
#include <climits>
#include <iomanip>
#include <iostream>
#include <sstream>
```

Include dependency graph for StringUtils.h:




This graph shows which files directly or indirectly include this file:

## Classes

- class insilico::is_classified< Type, charT >
- class insilico::do_to_upper< charT >
- class insilico::do_to_lower< charT >

## Namespaces

- namespace insilico

## Functions

- template<typename stringT >
  stringT insilico::trim_left (const stringT &s, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT insilico::trim_right (const stringT &s, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT insilico::trim (const stringT &s, const std::locale &loc=std::locale())
- template<typename Container , typename stringT >
  void insilico::split (Container &cont, const stringT &s, const std::locale &loc=std::locale())
- template<typename Container , typename stringT >
  void insilico::split (Container &cont, const stringT &s, const stringT &delim)
- template<typename Container , typename stringT , typename Pred >
  void insilico::split_if (Container &cont, const stringT &s, const Pred &pred)
- template<typename It , typename stringT >
  stringT insilico::join (const It &begin, const It &end, const stringT &delim)
- template<typename stringT >
  stringT insilico::to_upper (const stringT &str, const std::locale &loc=std::locale())
- template<typename stringT >
  stringT insilico::to_lower (const stringT &str, const std::locale &loc=std::locale())
- std::string insilico::trim_left (const char *s, const std::locale &loc=std::locale())
- std::wstring insilico::trim_left (const wchar_t *s, const std::locale &loc=std::locale())
- std::string insilico::trim_right (const char *s, const std::locale &loc=std::locale())
- std::wstring insilico::trim_right (const wchar_t *s, const std::locale &loc=std::locale())
- std::string insilico::trim (const char *s, const std::locale &loc=std::locale())
- std::wstring insilico::trim (const wchar_t *s, const std::locale &loc=std::locale())
- template<typename Container >
  void insilico::split (Container &cont, const char *s, const std::locale &loc=std::locale())

- template<typename Container >
  void insilico::split (Container &cont, const wchar_t *s, const std::locale &loc=std::locale())
- template<typename Container >
  void insilico::split (Container &cont, const std::string &s, const char *delim)
- template<typename Container >
  void insilico::split (Container &cont, const char *s, const std::string &delim)
- template<typename Container >
  void insilico::split (Container &cont, const char *s, const char *delim)
- template<typename Container >
  void insilico::split (Container &cont, const std::wstring &s, const wchar_t *delim)
- template<typename Container >
  void insilico::split (Container &cont, const wchar_t *s, const std::wstring &delim)
- template<typename Container >
  void insilico::split (Container &cont, const wchar_t *s, const wchar_t *delim)
- template<typename Container , typename Pred >
  void insilico::split_if (Container &cont, const char *s, const Pred &pred)
- template<typename Container , typename Pred >
  void insilico::split_if (Container &cont, const wchar_t *s, const Pred &pred)
- template<typename It >
  std::string insilico::join (const It &begin, const It &end, const char *delim)
- template<typename It >
  std::wstring insilico::join (const It &begin, const It &end, const wchar_t *delim)
- std::string insilico::to_upper (const char *s, const std::locale &loc=std::locale())
- std::wstring insilico::to_upper (const wchar_t *s, const std::locale &loc=std::locale())
- std::string insilico::to_lower (const char *s, const std::locale &loc=std::locale())
- std::wstring insilico::to_lower (const wchar_t *s, const std::locale &loc=std::locale())
- template<typename T >
  std::string insilico::get_bits (T value)
- template<typename T >
  std::string insilico::zeroPadNumber (T num, int padSize)

### 7.36.1 Detailed Description

Various string-related utilities. This is originally from Nate Barney circa Moore Lab days 2003-2007. His function naming follows lowercase with underscores style, while my additions are camelCase.

**Author**

Bill White, Nate Barney

**Version**

1.0

Contact: bill.c.white@gmail.com Created on: 10/7/04

Definition in file StringUtils.h.

# Index


~ArffDataset
ArffDataset, 18
~BirdseedData
BirdseedData, 22
~ChiSquared
ChiSquared, 29
~Dataset
Dataset, 40
~DatasetInstance
DatasetInstance, 71
~DgeData
DgeData, 79
~EvaporativeCooling
EvaporativeCooling, 88
~GSLRandomBase
GSLRandomBase, 94
~GSLRandomFlat
GSLRandomFlat, 97
~PlinkBinaryDataset
PlinkBinaryDataset, 101
~PlinkDataset
PlinkDataset, 107
~PlinkRawDataset
PlinkRawDataset, 112
~RReliefF
RReliefF, 131
~RandomJungle
RandomJungle, 116
~ReliefF
ReliefF, 123

ARFF_DATASET
Insilico.h, 162
ARFF_DATE_TYPE
ArffDataset.h, 135
ARFF_ERROR_TYPE
ArffDataset.h, 135
ARFF_NOMINAL_TYPE
ArffDataset.h, 134
ARFF_NUMERIC_TYPE
ArffDataset.h, 134
ARFF_STRING_TYPE
ArffDataset.h, 135
AddInfluenceFactorD
DatasetInstance, 71
AddNumeric
DatasetInstance, 72
algorithmType
EvaporativeCooling, 91
alternatePhenotypesFilename
Dataset, 63
AnalysisType
Insilico.h, 161
analysisType
EvaporativeCooling, 91
ReliefF, 124
ArffDataset.h
ARFF_DATE_TYPE, 135
ARFF_ERROR_TYPE, 135
ARFF_NOMINAL_TYPE, 134
ARFF_NUMERIC_TYPE, 134
ARFF_STRING_TYPE, 135
ArffAttributeType
ArffDataset.h, 134
ArffDataset, 15
~ArffDataset, 18
ArffDataset, 18
ArffDataset, 18
attributeTypes, 19
GetTypeOf, 19
LoadSnps, 19
missingAttributeValuesToCheck, 19
missingClassValuesToCheck, 19
nominalValues, 19
PrintNominalsMapping, 19
relationName, 19
ArffDataset.h
ArffAttributeType, 134
attributeAlleleCounts
Dataset, 63
attributeAlleles
Dataset, 63
AttributeIndex
ReliefF.cpp, 174
AttributeIndexIt
ReliefF.cpp, 174
AttributeInteractionInformation
Dataset, 41
AttributeLevel
Insilico.h, 160
attributeLevelsSeen
Dataset, 64
attributeMinorAllele
Dataset, 64
attributeMutationMap
Dataset, 64
AttributeMutationType
Insilico.h, 161
attributeMutationTypes

Dataset, 64
attributeNames
Dataset, 64
attributeSort
ReliefF.cpp, 175
AttributeType
Insilico.h, 161
attributeTypes
ArffDataset, 19
attributes
DatasetInstance, 76
attributesMask
Dataset, 64
attributesMaskPushed
Dataset, 64

BIRDSEED_ANALYSIS
Insilico.h, 161
best_n
insilico, 10
bestClassificationError
EvaporativeCooling, 91
bestNeighborIds
DatasetInstance, 76
bestNeighborIdsDiffClass
DatasetInstance, 76
bestNeighborIdsSameClass
DatasetInstance, 76
BirdseedData, 20
∼BirdseedData, 22
BirdseedData, 22
BirdseedData, 22
excludeSnps, 24
excludeSnpsFilename, 24
genotypeCounts, 25
GetAlleleCounts, 22
GetGenotypeCounts, 22
GetMajorAlleleFrequency, 22
GetMajorMinorAlleles, 22
GetMissingValues, 22
GetNumSNPs, 23
GetNumSubjects, 23
GetSNPNames, 23
GetSamplePhenotype, 23
GetSubjectGenotypes, 23
GetSubjectLabels, 23
GetSubjectNames, 23
hasExcludedSnps, 25
hasIncludedSnps, 25
HasPhenotypes, 23
hasPhenotypes, 25
HasSubjectLabels, 23
hasSubjectLabels, 25
includeSnps, 25
includeSnpsFilename, 25
LoadData, 24
missingValues, 25
phenosFilename, 25
phenotypes, 25
PrintAlleleCounts, 24
PrintInfo, 24
snpAlleleCounts, 25
snpGenotypes, 26
snpMajorAlleleFreq, 26
snpMajorMinorAlleles, 26
snpNames, 26
snpsFilename, 26
subjectLabels, 26
subjectLabelsFilename, 26
subjectNames, 26

CASE_CONTROL_CLASS_TYPE
Insilico.h, 162
CONTINUOUS_CLASS_TYPE
Insilico.h, 162
CSV_DELIMITED_DATASET
Insilico.h, 162
COMMAND_LINE_ERROR
Insilico.h, 166
CalculateDistanceMatrix
Dataset, 41
CalculateGainMatrix
Dataset, 41
CalculateInteractionInformation
Dataset, 42
CheckHardyWeinbergEquilibrium
Dataset, 42
CheckMissing
DistanceMetrics.cpp, 143
DistanceMetrics.h, 147
CheckMissingNumeric
DistanceMetrics.cpp, 144
DistanceMetrics.h, 147
ChiSquared, 27
∼ChiSquared, 29
ChiSquared, 29
chiSquaredValues, 31
ChiSquared, 29
ClearTables, 29
ComputeScore, 29
ComputeScores, 29
dataset, 31
expectedContingencyTable, 31
GetFrequencyCounts, 30
numClasses, 31
numLevels, 31
observedFreqTable, 31
PrepareForAttribute, 30
PrintScores, 30
PrintTables, 30
scores, 31
WriteScores, 30
chiSquaredValues
ChiSquared, 31
ChooseSnpsDatasetByExtension
Insilico.cpp, 155
Insilico.h, 163
classColumn

Dataset, 64
classIndexes
Dataset, 64
classLabel
DatasetInstance, 76
ClassLevel
Insilico.h, 160
ClassType
Insilico.h, 161
classificationError
RandomJungle, 118
ClearInfluenceFactors
DatasetInstance, 72
ClearTables
ChiSquared, 29
ComputeAttributeScores
RandomJungle, 116
ReliefF, 123
RReliefF, 131
ComputeAttributeScoresIteratively
ReliefF, 123
ComputeAttributeScoresRjungle
RandomJungle, 116
ComputeClassificationErrorRJ
EvaporativeCooling, 88
ComputeECScores
EvaporativeCooling, 88
ComputeFreeEnergy
EvaporativeCooling, 88
ComputeInstanceToInstanceDistance
Dataset, 43
ComputeScore
ChiSquared, 29
ComputeScores
ChiSquared, 29
ComputeWeightByDistanceFactors
ReliefF, 123
condentropy
Statistics.cpp, 178
Statistics.h, 182
ConditionalEntropy
Statistics.cpp, 178
Statistics.h, 182
ConfigMap
Insilico.h, 160
ConstructAttributeCart
Statistics.cpp, 179
Statistics.h, 182
continuousPhenotypeMinMax
Dataset, 64
counts
DgeData, 80
countsFilename
DgeData, 80
CreateDummyAlleles
Dataset, 43

DATASET_CONVERSION
Insilico.h, 161
DGE_ANALYSIS
Insilico.h, 161
DIAGNOSTIC_ANALYSIS
Insilico.h, 161
DISCRETE_TYPE
Insilico.h, 161
DISCRETE_VALUE
Insilico.h, 163
DISTANCE_MATRIX_ANALYSIS
Insilico.h, 161
DATASET_LOAD_ERROR
Insilico.h, 166
DEBUG_E
Statistics.cpp, 178
DEBUG_Z
Statistics.cpp, 178
Dataset, 31
~Dataset, 40
alternatePhenotypesFilename, 63
attributeAlleleCounts, 63
attributeAlleles, 63
AttributeInteractionInformation, 41
attributeLevelsSeen, 64
attributeMinorAllele, 64
attributeMutationMap, 64
attributeMutationTypes, 64
attributeNames, 64
attributesMask, 64
attributesMaskPushed, 64
CalculateDistanceMatrix, 41
CalculateGainMatrix, 41
CalculateInteractionInformation, 42
CheckHardyWeinbergEquilibrium, 42
classColumn, 64
classIndexes, 64
ComputeInstanceToInstanceDistance, 43
continuousPhenotypeMinMax, 64
CreateDummyAlleles, 43
Dataset, 40
DetermineTreeType, 43
ExcludeMonomorphs, 43
ExtractAttributes, 43
genotypeCounts, 65
GetAlternatePhenotypesFilename, 44
GetAttribute, 44
GetAttributeAlleles, 44
GetAttributeIndexFromName, 44
GetAttributeMAF, 45
GetAttributeMutationType, 45
GetAttributeNames, 45
GetAttributeRowCol, 45
GetAttributeTiTvCounts, 46
GetAttributeValues, 46
GetClassColumn, 46
GetClassIndexes, 46
GetClassProbability, 47
GetClassValues, 47
GetDistanceMetrics, 47

GetInstance, 47
GetInstanceIds, 47
GetInstanceIndexForID, 48
GetIntForGenotype, 48
GetJukesCantorDistance, 48
GetKimuraDistance, 48
GetMeanForNumeric, 48
GetMinMaxForContinuousPhenotype, 49
GetMinMaxForNumeric, 49
GetNumeric, 49
GetNumericIndexFromName, 49
GetNumericRowCol, 50
GetNumericValues, 50, 51
GetNumericsFilename, 50
GetNumericsNames, 50
GetProbabilityValueGivenClass, 51
GetRandomInstance, 51
GetSnpsFilename, 51
GetVariableNames, 51
HasAllelicInfo, 52
hasAllelicInfo, 65
HasAlternatePhenotypes, 52
hasAlternatePhenotypes, 65
HasContinuousPhenotypes, 52
hasContinuousPhenotypes, 65
HasGenotypes, 52
hasGenotypes, 65
HasNumerics, 52
hasNumerics, 65
HasPhenotypes, 52
hasPhenotypes, 65
instanceIds, 65
instanceIdsToLoad, 65
instances, 66
instancesMask, 66
instancesMaskPushed, 66
IsLoadableInstanceID, 52
levelCounts, 66
levelCountsByClass, 66
LoadAlternatePhenotypes, 53
LoadDataset, 53, 54
LoadNumerics, 54
LoadSnps, 54
MaskGetAllVariableNames, 55
MaskGetAttributeIndices, 55
MaskGetAttributeMask, 55
MaskGetInstanceIds, 56
MaskGetInstanceIndices, 56
MaskGetInstanceMask, 56
MaskIncludeAllAttributes, 56
MaskIncludeAllInstances, 56
maskIsPushed, 66
MaskPopAll, 56
MaskPushAll, 57
MaskRemoveInstance, 57
MaskRemoveVariable, 57
MaskRemoveVariableType, 57
MaskSearchInstance, 58
MaskSearchVariableType, 58
MaskWriteNewDataset, 58
missingNumericValues, 66
missingValues, 66
NumAttributes, 58
NumClasses, 58
numDiff, 66
NumInstances, 59
NumLevels, 59
numMetric, 67
NumNumerics, 59
NumVariables, 59
numericsFilename, 67
numericsIds, 67
numericsMask, 67
numericsMaskPushed, 67
numericsMinMax, 67
numericsNames, 67
phenotypesIds, 67
Print, 59
PrintAttributeLevelsSeen, 59
PrintClassIndexInfo, 59
PrintLevelCounts, 60
PrintMaskStats, 60
PrintMissingValuesStats, 60
PrintNumericsStats, 60
PrintStats, 60
PrintStatsSimple, 60
ProcessExclusionFile, 60
rng, 67
RunSnpDiagnosticTests, 60
SNPHWE, 61
SetDistanceMetrics, 61
snpDiff, 68
snpMetric, 68
snpsFilename, 68
SwapAttributes, 61
UpdateAllLevelCounts, 62
UpdateLevelCounts, 62
WriteLevelCounts, 62
WriteNewDataset, 62
WriteNewPlinkPedDataset, 63
WriteSnpTiTvInfo, 63
dataset
ChiSquared, 31
DatasetInstance, 76
EvaporativeCooling, 91
RandomJungle, 118
ReliefF, 124
DatasetInstance, 68
∼DatasetInstance, 71
AddInfluenceFactorD, 71
AddNumeric, 72
attributes, 76
bestNeighborIds, 76
bestNeighborIdsDiffClass, 76
bestNeighborIdsSameClass, 76
classLabel, 76

ClearInfluenceFactors, 72
dataset, 76
DatasetInstance, 71
DatasetInstance, 71
GetAttribute, 72
GetClass, 72
GetDatasetPtr, 72
GetInfluenceFactorD, 72
GetNNearestInstances, 72, 73
GetNumeric, 73
GetPredictedValueTau, 74
LoadInstanceFromVector, 74
neighborInfluenceFactorDs, 76
NumAttributes, 74
NumNumerics, 74
numerics, 76
predictedValueTau, 76
Print, 74
PrintDistancePairs, 74
SetClass, 74
SetDistanceSums, 75
SetPredictedValueTau, 75
SwapAttributes, 75
DatasetInstance.cpp
T, 141
deref_less, 77
operator(), 77
deref_less_bcw, 77
operator(), 77
DetectClassType
Insilico.cpp, 155
Insilico.h, 163
DetermineRandomJungleTreeType
Insilico.cpp, 156
Insilico.h, 163
DetermineTreeType
Dataset, 43
DgeData, 78
∼DgeData, 79
counts, 80
countsFilename, 80
DgeData, 79
DgeData, 79
geneNames, 81
GetGeneMinMax, 79
GetGeneNames, 79
GetNormalizationFactors, 79
GetNumGenes, 79
GetNumSamples, 80
GetSampleCounts, 80
GetSampleNames, 80
GetSamplePhenotype, 80
hasNormFactors, 81
LoadData, 80
minMaxGeneCounts, 81
minMaxSampleCounts, 81
normFactors, 81
normsFilename, 81
phenosFilename, 81
phenotypes, 81
PrintSampleStats, 80
sampleNames, 81
sampleZeroes, 82
diffAMM
DistanceMetrics.cpp, 144
DistanceMetrics.h, 148
diffGMM
DistanceMetrics.cpp, 144
DistanceMetrics.h, 148
diffKM
DistanceMetrics.cpp, 144
DistanceMetrics.h, 148
diffManhattan
DistanceMetrics.cpp, 145
DistanceMetrics.h, 148
diffNCA
DistanceMetrics.cpp, 145
DistanceMetrics.h, 149
diffPredictedValueTau
DistanceMetrics.cpp, 145
DistanceMetrics.h, 149
DistanceMetrics.cpp
CheckMissing, 143
CheckMissingNumeric, 144
diffAMM, 144
diffGMM, 144
diffKM, 144
diffManhattan, 145
diffNCA, 145
diffPredictedValueTau, 145
norm, 146
DistanceMetrics.h
CheckMissing, 147
CheckMissingNumeric, 147
diffAMM, 148
diffGMM, 148
diffKM, 148
diffManhattan, 148
diffNCA, 149
diffPredictedValueTau, 149
norm, 149
DistancePair
Insilico.h, 160
DistancePairs
Insilico.h, 160
DistancePairsIt
Insilico.h, 160
do_to_lower
insilico::do_to_lower, 82
do_to_upper
insilico::do_to_upper, 83
doRemovePercent
ReliefF, 125

EC_ALL
EvaporativeCooling.h, 152
EC_RF

EvaporativeCooling.h, 152
EC_RJ
EvaporativeCooling.h, 152
ERROR_FILE
PlinkDataset.h, 170
EcAlgorithmType
EvaporativeCooling.h, 152
EcScores
EvaporativeCooling.h, 152
ecScores
EvaporativeCooling, 91
EcScoresCIt
EvaporativeCooling.h, 152
EcScoresIt
EvaporativeCooling.h, 152
Entropy
Statistics.cpp, 179
Statistics.h, 182
evaporatedAttributes
EvaporativeCooling, 91
EvaporativeCooling.h
EC_ALL, 152
EC_RF, 152
EC_RJ, 152
EvaporativeCooling, 84
~EvaporativeCooling, 88
algorithmType, 91
analysisType, 91
bestClassificationError, 91
ComputeClassificationErrorRJ, 88
ComputeECScores, 88
ComputeFreeEnergy, 88
dataset, 91
ecScores, 91
evaporatedAttributes, 91
EvaporativeCooling, 88
EvaporativeCooling, 88
freeEnergyScores, 91
GetAlgorithmType, 89
GetECScores, 89
GetRandomJungleScores, 89
GetReliefFScores, 89
numRFThreads, 91
numTargetAttributes, 91
numToRemoveNextIteration, 92
numToRemovePerIteration, 92
optimalTemperature, 92
OptimizeTemperature, 89
optimizeTemperature, 92
outFilesPrefix, 92
paramsMap, 92
PrintAllScoresTabular, 89
PrintAttributeScores, 89
PrintKendallTaus, 90
PrintRFAttributeScores, 90
PrintRJAttributeScores, 90
randomJungle, 92
reliefF, 92
RemoveWorstAttributes, 90
rfScores, 92
rjScores, 92
RunReliefF, 90
WriteAttributeScores, 90
EvaporativeCooling.cpp
scoresSortAsc, 150
scoresSortAscByName, 150
scoresSortDesc, 150
EvaporativeCooling.h
EcAlgorithmType, 152
EcScores, 152
EcScoresCIt, 152
EcScoresIt, 152
libec_is_present, 153
ExcludeMonomorphs
Dataset, 43
excludeSnps
BirdseedData, 24
excludeSnpsFilename
BirdseedData, 24
expectedContingencyTable
ChiSquared, 31
ExtractAttributes
Dataset, 43

filenameBase
PlinkBinaryDataset, 103
PlinkDataset, 109
finalScores
ReliefF, 125
freeEnergyScores
EvaporativeCooling, 91

GSLRandomBase, 93
~GSLRandomBase, 94
GSLRandomBase, 94
GSLRandomBase, 94
nextRandVal, 94
rStatePtr_, 95
state, 94
GSLRandomFlat, 95
~GSLRandomFlat, 97
GSLRandomFlat, 97
GSLRandomFlat, 97
lower_, 97
nextRandVal, 97
upper_, 97
geneNames
DgeData, 81
genotypeCounts
BirdseedData, 25
Dataset, 65
get_bits
insilico, 10
GetAlgorithmType
EvaporativeCooling, 89
GetAlleleCounts
BirdseedData, 22

GetAlternatePhenotypesFilename
Dataset, 44
GetAttribute
Dataset, 44
DatasetInstance, 72
GetAttributeAlleles
Dataset, 44
GetAttributeIndexFromName
Dataset, 44
GetAttributeMAF
Dataset, 45
PlinkBinaryDataset, 102
PlinkDataset, 107
GetAttributeMutationType
Dataset, 45
PlinkBinaryDataset, 102
PlinkDataset, 108
GetAttributeNames
Dataset, 45
GetAttributeRowCol
Dataset, 45
GetAttributeTiTvCounts
Dataset, 46
GetAttributeValues
Dataset, 46
GetClass
DatasetInstance, 72
GetClassColumn
Dataset, 46
GetClassIndexes
Dataset, 46
GetClassProbability
Dataset, 47
GetClassValues
Dataset, 47
GetClassificationError
RandomJungle, 117
GetConfigValue
Insilico.cpp, 156
Insilico.h, 164
GetDatasetPtr
DatasetInstance, 72
GetDistanceMetrics
Dataset, 47
GetECScores
EvaporativeCooling, 89
GetFileBasename
Insilico.cpp, 156
Insilico.h, 164
GetFileExtension
Insilico.cpp, 156
Insilico.h, 164
GetFrequencyCounts
ChiSquared, 30
GetGeneMinMax
DgeData, 79
GetGeneNames
DgeData, 79
GetGenotypeCounts
BirdseedData, 22
GetInfluenceFactorD
DatasetInstance, 72
GetInstance
Dataset, 47
GetInstanceIds
Dataset, 47
GetInstanceIndexForID
Dataset, 48
GetIntForGenotype
Dataset, 48
GetJukesCantorDistance
Dataset, 48
GetKimuraDistance
Dataset, 48
GetMajorAlleleFrequency
BirdseedData, 22
GetMajorMinorAlleles
BirdseedData, 22
GetMatchingIds
Insilico.cpp, 156
Insilico.h, 164
GetMeanForNumeric
Dataset, 48
GetMinMaxForContinuousPhenotype
Dataset, 49
GetMinMaxForNumeric
Dataset, 49
GetMissingValues
BirdseedData, 22
GetNNearestInstances
DatasetInstance, 72, 73
GetNormalizationFactors
DgeData, 79
GetNumGenes
DgeData, 79
GetNumSNPs
BirdseedData, 23
GetNumSamples
DgeData, 80
GetNumSubjects
BirdseedData, 23
GetNumeric
Dataset, 49
DatasetInstance, 73
GetNumericIndexFromName
Dataset, 49
GetNumericRowCol
Dataset, 50
GetNumericValues
Dataset, 50, 51
GetNumericsFilename
Dataset, 50
GetNumericsNames
Dataset, 50
GetPredictedValueTau
DatasetInstance, 74

GetProbabilityValueGivenClass
Dataset, 51
GetRandomInstance
Dataset, 51
GetRandomJungleScores
EvaporativeCooling, 89
GetReliefFScores
EvaporativeCooling, 89
GetSNPNames
BirdseedData, 23
GetSampleCounts
DgeData, 80
GetSampleNames
DgeData, 80
GetSamplePhenotype
BirdseedData, 23
DgeData, 80
GetScores
RandomJungle, 117
ReliefF, 124
GetSnpsFilename
Dataset, 51
GetSubjectGenotypes
BirdseedData, 23
GetSubjectLabels
BirdseedData, 23
GetSubjectNames
BirdseedData, 23
GetTypeOf
ArffDataset, 19
GetVariableNames
Dataset, 51

HasAllelicInfo
Dataset, 52
hasAllelicInfo
Dataset, 65
HasAlternatePhenotypes
Dataset, 52
hasAlternatePhenotypes
Dataset, 65
HasContinuousPhenotypes
Dataset, 52
hasContinuousPhenotypes
Dataset, 65
hasExcludedSnps
BirdseedData, 25
HasGenotypes
Dataset, 52
hasGenotypes
Dataset, 65
hasIncludedSnps
BirdseedData, 25
hasNormFactors
DgeData, 81
HasNumerics
Dataset, 52
hasNumerics
Dataset, 65
HasPhenotypes
BirdseedData, 23
Dataset, 52
hasPhenotypes
BirdseedData, 25
Dataset, 65
HasSubjectLabels
BirdseedData, 23
hasSubjectLabels
BirdseedData, 25
Histogram
Statistics.h, 181
HistogramIt
Statistics.h, 181

INTEGRATED_ANALYSIS
Insilico.h, 161
INVALID_DISTANCE
Insilico.h, 166
INVALID_INDEX
Insilico.h, 166
INVALID_INT_VALUE
Insilico.h, 166
includeSnps
BirdseedData, 25
includeSnpsFilename
BirdseedData, 25
insilico, 9
best_n, 10
get_bits, 10
join, 10, 11
split, 11, 12
split_if, 12
to_lower, 12
to_upper, 12, 13
trim, 13
trim_left, 13
trim_right, 13
zeroPadNumber, 13
Insilico.h
ARFF_DATASET, 162
BIRDSEED_ANALYSIS, 161
CASE_CONTROL_CLASS_TYPE, 162
CONTINUOUS_CLASS_TYPE, 162
CSV_DELIMITED_DATASET, 162
DATASET_CONVERSION, 161
DGE_ANALYSIS, 161
DIAGNOSTIC_ANALYSIS, 161
DISCRETE_TYPE, 161
DISCRETE_VALUE, 163
DISTANCE_MATRIX_ANALYSIS, 161
INTEGRATED_ANALYSIS, 161
LIBRARY_RUN_MODE, 162
MISSING_VALUE, 163
MULTI_CLASS_TYPE, 162
NO_ANALYSIS, 161
NO_CLASS_TYPE, 162
NO_OUTPUT_DATASET, 162
NO_TYPE, 161

NO_VALUE, 163
NOMINAL_NOMINAL_TREE, 162
NOMINAL_NUMERIC_FLOATS, 162
NOMINAL_NUMERIC_TREE, 162
NUMERIC_NOMINAL_TREE, 162
NUMERIC_NUMERIC_TREE, 162
NUMERIC_ONLY_ANALYSIS, 161
NUMERIC_TYPE, 161
NUMERIC_VALUE, 163
PLINK_BED_DATASET, 162
PLINK_PED_DATASET, 162
REGRESSION_ANALYSIS, 161
SNP_ONLY_ANALYSIS, 161
SYSTEM_CALL_RUN_MODE, 162
TAB_DELIMITED_DATASET, 162
TRANSITION_MUTATION, 161
TRANSVERSION_MUTATION, 161
UNKNOWN_MUTATION, 161
UNKNOWN_RUN_MODE, 162
UNKNOWN_TREE_TYPE, 162
Insilico.cpp
ChooseSnpsDatasetByExtension, 155
DetectClassType, 155
DetermineRandomJungleTreeType, 156
GetConfigValue, 156
GetFileBasename, 156
GetFileExtension, 156
GetMatchingIds, 156
LoadNumericIds, 156
LoadPhenoIds, 156
ProtectedLog, 157
Timestamp, 157
Insilico.h
AnalysisType, 161
AttributeLevel, 160
AttributeMutationType, 161
AttributeType, 161
COMMAND_LINE_ERROR, 166
ChooseSnpsDatasetByExtension, 163
ClassLevel, 160
ClassType, 161
ConfigMap, 160
DATASET_LOAD_ERROR, 166
DetectClassType, 163
DetermineRandomJungleTreeType, 163
DistancePair, 160
DistancePairs, 160
DistancePairsIt, 160
GetConfigValue, 164
GetFileBasename, 164
GetFileExtension, 164
GetMatchingIds, 164
INVALID_DISTANCE, 166
INVALID_INDEX, 166
INVALID_INT_VALUE, 166
LoadNumericIds, 165
LoadPhenoIds, 165
NumericLevel, 161
OutputDatasetType, 162
PrintVector, 165
ProtectedLog, 165
RandomJungleRunMode, 162
RandomJungleTreeType, 162
Timestamp, 166
ValueType, 162
insilico::do_to_lower
do_to_lower, 82
m_ctype, 83
operator(), 83
insilico::do_to_lower< charT >, 82
insilico::do_to_upper
do_to_upper, 83
m_ctype, 84
operator(), 84
insilico::do_to_upper< charT >, 83
insilico::is_classified
is_classified, 98
m_ctype, 98
operator(), 98
insilico::is_classified< Type, charT >, 97
instanceIds
Dataset, 65
instanceIdsToLoad
Dataset, 65
instanceIndicesToKeep
PlinkBinaryDataset, 103
instances
Dataset, 66
instancesMask
Dataset, 66
instancesMaskPushed
Dataset, 66
is_classified
insilico::is_classified, 98
IsLoadableInstanceID
Dataset, 52

join
insilico, 10, 11

k
ReliefF, 125
KendallTau
Statistics.cpp, 179
Statistics.h, 182, 183

LIBRARY_RUN_MODE
Insilico.h, 162
levelCounts
Dataset, 66
levelCountsByClass
Dataset, 66
libec_is_present
EvaporativeCooling.h, 153
librelieff_is_present
ReliefF.cpp, 175
LoadAlternatePhenotypes

Dataset, 53
LoadData
BirdseedData, 24
DgeData, 80
LoadDataset
Dataset, 53, 54
LoadInstanceFromVector
DatasetInstance, 74
LoadNumericIds
Insilico.cpp, 156
Insilico.h, 165
LoadNumerics
Dataset, 54
LoadPhenoIds
Insilico.cpp, 156
Insilico.h, 165
LoadSnps
ArffDataset, 19
Dataset, 54
PlinkBinaryDataset, 102
PlinkDataset, 108
PlinkRawDataset, 112
lower_
GSLRandomFlat, 97

m
ReliefF, 125
MAP3_FILE
PlinkDataset.h, 170
MAP4_FILE
PlinkDataset.h, 170
MISSING_VALUE
Insilico.h, 163
MULTI_CLASS_TYPE
Insilico.h, 162
m_ctype
insilico::do_to_lower, 83
insilico::do_to_upper, 84
insilico::is_classified, 98
MapFileType
PlinkDataset.h, 170
MaskGetAllVariableNames
Dataset, 55
MaskGetAttributeIndices
Dataset, 55
MaskGetAttributeMask
Dataset, 55
MaskGetInstanceIds
Dataset, 56
MaskGetInstanceIndices
Dataset, 56
MaskGetInstanceMask
Dataset, 56
MaskIncludeAllAttributes
Dataset, 56
MaskIncludeAllInstances
Dataset, 56
maskIsPushed
Dataset, 66
MaskPopAll
Dataset, 56
MaskPushAll
Dataset, 57
MaskRemoveInstance
Dataset, 57
MaskRemoveVariable
Dataset, 57
MaskRemoveVariableType
Dataset, 57
MaskSearchInstance
Dataset, 58
MaskSearchVariableType
Dataset, 58
MaskWriteNewDataset
Dataset, 58
minMaxGeneCounts
DgeData, 81
minMaxSampleCounts
DgeData, 81
missingAttributeValuesToCheck
ArffDataset, 19
PlinkBinaryDataset, 104
missingClassValuesToCheck
ArffDataset, 19
PlinkBinaryDataset, 104
PlinkDataset, 109
missingNumericValues
Dataset, 66
missingPhenoLines
PlinkBinaryDataset, 104
missingValues
BirdseedData, 25
Dataset, 66

NO_ANALYSIS
Insilico.h, 161
NO_CLASS_TYPE
Insilico.h, 162
NO_OUTPUT_DATASET
Insilico.h, 162
NO_TYPE
Insilico.h, 161
NO_VALUE
Insilico.h, 163
NOMINAL_NOMINAL_TREE
Insilico.h, 162
NOMINAL_NUMERIC_FLOATS
Insilico.h, 162
NOMINAL_NUMERIC_TREE
Insilico.h, 162
NUMERIC_NOMINAL_TREE
Insilico.h, 162
NUMERIC_NUMERIC_TREE
Insilico.h, 162
NUMERIC_ONLY_ANALYSIS
Insilico.h, 161
NUMERIC_TYPE
Insilico.h, 161

NUMERIC_VALUE
Insilico.h, 163
neighborInfluenceFactorDs
DatasetInstance, 76
nextRandVal
GSLRandomBase, 94
GSLRandomFlat, 97
nominalValues
ArffDataset, 19
norm
DistanceMetrics.cpp, 146
DistanceMetrics.h, 149
normFactors
DgeData, 81
normsFilename
DgeData, 81
NumAttributes
Dataset, 58
DatasetInstance, 74
numAttributesRead
PlinkBinaryDataset, 104
NumClasses
Dataset, 58
numClasses
ChiSquared, 31
numClassesRead
PlinkBinaryDataset, 104
numDiff
Dataset, 66
ReliefF, 125
NumInstances
Dataset, 59
numInstancesRead
PlinkBinaryDataset, 104
NumLevels
Dataset, 59
numLevels
ChiSquared, 31
numMetric
Dataset, 67
ReliefF, 125
NumNumerics
Dataset, 59
DatasetInstance, 74
numRFThreads
EvaporativeCooling, 91
numTargetAttributes
EvaporativeCooling, 91
numToRemoveNextIteration
EvaporativeCooling, 92
numToRemovePerIteration
EvaporativeCooling, 92
NumVariables
Dataset, 59
NumericLevel
Insilico.h, 161
numerics
DatasetInstance, 76
numericsFilename
Dataset, 67
numericsIds
Dataset, 67
numericsMask
Dataset, 67
numericsMaskPushed
Dataset, 67
numericsMinMax
Dataset, 67
numericsNames
Dataset, 67

observedFreqTable
ChiSquared, 31
one_over_m_times_k
ReliefF, 125
operator()
deref_less, 77
deref_less_bcw, 77
insilico::do_to_lower, 83
insilico::do_to_upper, 84
insilico::is_classified, 98
optimalTemperature
EvaporativeCooling, 92
OptimizeTemperature
EvaporativeCooling, 89
optimizeTemperature
EvaporativeCooling, 92
outFilesPrefix
EvaporativeCooling, 92
OutputDatasetType
Insilico.h, 162

PLINK_BED_DATASET
Insilico.h, 162
PLINK_PED_DATASET
Insilico.h, 162
paramsMap
EvaporativeCooling, 92
phenosFilename
BirdseedData, 25
DgeData, 81
phenotypes
BirdseedData, 25
DgeData, 81
phenotypesIds
Dataset, 67
PlinkDataset.h
ERROR_FILE, 170
MAP3_FILE, 170
MAP4_FILE, 170
PlinkBinaryDataset, 98
~PlinkBinaryDataset, 101
filenameBase, 103
GetAttributeMAF, 102
GetAttributeMutationType, 102
instanceIndicesToKeep, 103
LoadSnps, 102

missingAttributeValuesToCheck, 104
missingClassValuesToCheck, 104
missingPhenoLines, 104
numAttributesRead, 104
numClassesRead, 104
numInstancesRead, 104
PlinkBinaryDataset, 101
PlinkBinaryDataset, 101
ReadBimFile, 103
ReadFamFile, 103
validAttributeValues, 104
PlinkDataset, 104
~PlinkDataset, 107
filenameBase, 109
GetAttributeMAF, 107
GetAttributeMutationType, 108
LoadSnps, 108
missingClassValuesToCheck, 109
PlinkDataset, 107
PlinkDataset, 107
PlinkDataset.h
MapFileType, 170
PlinkRawDataset, 109
~PlinkRawDataset, 112
LoadSnps, 112
PlinkRawDataset, 112
PlinkRawDataset, 112
PreComputeDistances
ReliefF, 124
PreComputeDistancesByMap
ReliefF, 124
predictedValueTau
DatasetInstance, 76
PrepareForAttribute
ChiSquared, 30
Print
Dataset, 59
DatasetInstance, 74
PrintAllScoresTabular
EvaporativeCooling, 89
PrintAlleleCounts
BirdseedData, 24
PrintAttributeLevelsSeen
Dataset, 59
PrintAttributeScores
EvaporativeCooling, 89
ReliefF, 124
PrintClassIndexInfo
Dataset, 59
PrintDistancePairs
DatasetInstance, 74
PrintHistogram
Statistics.cpp, 179
Statistics.h, 183
PrintInfo
BirdseedData, 24
PrintKendallTaus
EvaporativeCooling, 90
PrintLevelCounts
Dataset, 60
PrintMaskStats
Dataset, 60
PrintMissingValuesStats
Dataset, 60
PrintNominalsMapping
ArffDataset, 19
PrintNumericsStats
Dataset, 60
PrintRFAttributeScores
EvaporativeCooling, 90
PrintRJAttributeScores
EvaporativeCooling, 90
PrintSampleStats
DgeData, 80
PrintScores
ChiSquared, 30
PrintStats
Dataset, 60
PrintStatsSimple
Dataset, 60
PrintTables
ChiSquared, 30
PrintVector
Insilico.h, 165
ProcessExclusionFile
Dataset, 60
ProtectedLog
Insilico.cpp, 157
Insilico.h, 165

REGRESSION_ANALYSIS
Insilico.h, 161
RReliefF, 127
~RReliefF, 131
ComputeAttributeScores, 131
RReliefF, 130
RReliefF, 130
rStatePtr_
GSLRandomBase, 95
RandomJungle, 113
~RandomJungle, 116
classificationError, 118
ComputeAttributeScores, 116
ComputeAttributeScoresRjungle, 116
dataset, 118
GetClassificationError, 117
GetScores, 117
RandomJungle, 116
RandomJungle, 116
ReadClassificationError, 117
ReadScores, 117
rjParams, 118
RunClassifier, 117
runMode, 118
scores, 118
randomJungle
EvaporativeCooling, 92

RandomJungleRunMode
Insilico.h, 162
RandomJungleTreeType
Insilico.h, 162
randomlySelect
ReliefF, 126
ReadBimFile
PlinkBinaryDataset, 103
ReadClassificationError
RandomJungle, 117
ReadFamFile
PlinkBinaryDataset, 103
ReadScores
RandomJungle, 117
relationName
ArffDataset, 19
ReliefF, 118
∼ReliefF, 123
analysisType, 124
ComputeAttributeScores, 123
ComputeAttributeScoresIteratively, 123
ComputeWeightByDistanceFactors, 123
dataset, 124
doRemovePercent, 125
finalScores, 125
GetScores, 124
k, 125
m, 125
numDiff, 125
numMetric, 125
one_over_m_times_k, 125
PreComputeDistances, 124
PreComputeDistancesByMap, 124
PrintAttributeScores, 124
randomlySelect, 126
ReliefF, 122, 123
ReliefF, 122, 123
removePerIteration, 126
removePercentage, 126
ResetForNextIteration, 124
scoreNames, 126
snpDiff, 126
snpMetric, 126
W, 126
weightByDistanceMethod, 126
weightByDistanceSigma, 127
WriteAttributeScores, 124
reliefF
EvaporativeCooling, 92
ReliefF.cpp
AttributeIndex, 174
AttributeIndexIt, 174
attributeSort, 175
librelieff_is_present, 175
scoreSort, 175
ScoresMap, 174
ScoresMapIt, 174
T, 174
removePerIteration
ReliefF, 126
removePercentage
ReliefF, 126
RemoveWorstAttributes
EvaporativeCooling, 90
ResetForNextIteration
ReliefF, 124
rfScores
EvaporativeCooling, 92
rjParams
RandomJungle, 118
rjScores
EvaporativeCooling, 92
rng
Dataset, 67
RunClassifier
RandomJungle, 117
runMode
RandomJungle, 118
RunReliefF
EvaporativeCooling, 90
RunSnpDiagnosticTests
Dataset, 60

SNP_ONLY_ANALYSIS
Insilico.h, 161
SYSTEM_CALL_RUN_MODE
Insilico.h, 162
SNPHWE
Dataset, 61
sampleNames
DgeData, 81
sampleZeroes
DgeData, 82
scoreNames
ReliefF, 126
scoreSort
ReliefF.cpp, 175
scores
ChiSquared, 31
RandomJungle, 118
ScoresMap
ReliefF.cpp, 174
ScoresMapIt
ReliefF.cpp, 174
scoresSortAsc
EvaporativeCooling.cpp, 150
scoresSortAscByName
EvaporativeCooling.cpp, 150
scoresSortDesc
EvaporativeCooling.cpp, 150
SelfEntropy
Statistics.cpp, 179
Statistics.h, 183
SetClass
DatasetInstance, 74
SetDistanceMetrics
Dataset, 61

SetDistanceSums
DatasetInstance, 75
SetPredictedValueTau
DatasetInstance, 75
snpAlleleCounts
BirdseedData, 25
snpDiff
Dataset, 68
ReliefF, 126
snpGenotypes
BirdseedData, 26
snpMajorAlleleFreq
BirdseedData, 26
snpMajorMinorAlleles
BirdseedData, 26
snpMetric
Dataset, 68
ReliefF, 126
snpNames
BirdseedData, 26
snpsFilename
BirdseedData, 26
Dataset, 68
split
insilico, 11, 12
split_if
insilico, 12
src/library/ArffDataset.cpp, 133
src/library/ArffDataset.h, 133
src/library/BestN.h, 135
src/library/BirdseedData.cpp, 136
src/library/BirdseedData.h, 136
src/library/ChiSquared.cpp, 137
src/library/ChiSquared.h, 138
src/library/Dataset.cpp, 139
src/library/Dataset.h, 139
src/library/DatasetInstance.cpp, 140
src/library/DatasetInstance.h, 141
src/library/DgeData.cpp, 141
src/library/DgeData.h, 142
src/library/DistanceMetrics.cpp, 142
src/library/DistanceMetrics.h, 146
src/library/EvaporativeCooling.cpp, 150
src/library/EvaporativeCooling.h, 151
src/library/GSLRandomBase.h, 153
src/library/GSLRandomFlat.h, 153
src/library/Insilico.cpp, 154
src/library/Insilico.h, 157
src/library/PlinkBinaryDataset.cpp, 167
src/library/PlinkBinaryDataset.h, 168
src/library/PlinkDataset.cpp, 169
src/library/PlinkDataset.h, 169
src/library/PlinkRawDataset.cpp, 170
src/library/PlinkRawDataset.h, 171
src/library/RReliefF.cpp, 176
src/library/RReliefF.h, 176
src/library/RandomJungle.cpp, 172
src/library/RandomJungle.h, 172
src/library/ReliefF.cpp, 173
src/library/ReliefF.h, 175
src/library/Statistics.cpp, 177
src/library/Statistics.h, 180
src/library/StringUtils.h, 184
state
GSLRandomBase, 94
Statistics.cpp
condentropy, 178
ConditionalEntropy, 178
ConstructAttributeCart, 179
DEBUG_E, 178
DEBUG_Z, 178
Entropy, 179
KendallTau, 179
PrintHistogram, 179
SelfEntropy, 179
ZTransform, 179
Statistics.h
condentropy, 182
ConditionalEntropy, 182
ConstructAttributeCart, 182
Entropy, 182
Histogram, 181
HistogramIt, 181
KendallTau, 182, 183
PrintHistogram, 183
SelfEntropy, 183
VarStd, 184
VectorDouble, 181
VectorDoubleIt, 181
ZTransform, 184
subjectLabels
BirdseedData, 26
subjectLabelsFilename
BirdseedData, 26
subjectNames
BirdseedData, 26
SwapAttributes
Dataset, 61
DatasetInstance, 75

T
DatasetInstance.cpp, 141
ReliefF.cpp, 174
TAB_DELIMITED_DATASET
Insilico.h, 162
TRANSITION_MUTATION
Insilico.h, 161
TRANSVERSION_MUTATION
Insilico.h, 161
Timestamp
Insilico.cpp, 157
Insilico.h, 166
to_lower
insilico, 12
to_upper
insilico, 12, 13
trim

insilico, 13
trim_left
insilico, 13
trim_right
insilico, 13

UNKNOWN_MUTATION
Insilico.h, 161
UNKNOWN_RUN_MODE
Insilico.h, 162
UNKNOWN_TREE_TYPE
Insilico.h, 162
UpdateAllLevelCounts
Dataset, 62
UpdateLevelCounts
Dataset, 62
upper_
GSLRandomFlat, 97

validAttributeValues
PlinkBinaryDataset, 104
ValueType
Insilico.h, 162
VarStd
Statistics.h, 184
VectorDouble
Statistics.h, 181
VectorDoubleIt
Statistics.h, 181

W
ReliefF, 126
weightByDistanceMethod
ReliefF, 126
weightByDistanceSigma
ReliefF, 127
WriteAttributeScores
EvaporativeCooling, 90
ReliefF, 124
WriteLevelCounts
Dataset, 62
WriteNewDataset
Dataset, 62
WriteNewPlinkPedDataset
Dataset, 63
WriteScores
ChiSquared, 30
WriteSnpTiTvInfo
Dataset, 63

ZTransform
Statistics.cpp, 179
Statistics.h, 184
zeroPadNumber
insilico, 13