# IMSL Library for Java

Function Catalog

Version 2020.0

**IMSL**

# Table of Contents

# IMSL® LIBRARY for Java VERSION 2020.0

**THE IMSL LIBRARY FOR JAVA IS A PURE JAVA NUMERICAL LIBRARY, PROVIDING A BROAD RANGE OF ADVANCED ALGORITHMS FOR THE JAVA ENVIRONMENT. IT ALLOWS DEVELOPERS TO SEAMLESSLY INTEGRATE ADVANCED MATHEMATICAL, STATISTICAL, AND FINANCIAL FUNCTIONS INTO THEIR JAVA APPLICATIONS.**

# General Features of the Library

## IMSL NUMERICAL LIBRARIES

The IMSL Numerical Libraries offer an extensive and comprehensive package of trusted IMSL mathematical and statistical numerical algorithms. IMSL is available in Java™, Fortran, C, and Python.

These libraries free developers from building their own internal code by providing pre-written mathematical and statistical algorithms that can be embedded into C, Java™, Fortran, and Python applications.

## CONVENIENCE AND OPEN STANDARDS

The IMSL Library for Java (or JMSL Library) is a pure Java library that operates in the Java SE and Java EE frameworks. The library allows developers to seamlessly integrate advanced mathematical, statistical, and financial functions into their Java applications.

Rogue Wave Software has taken individual algorithms and released them as object-oriented, Java classes. The JMSL Library is thread safe and, like all IMSL products, is fully tested and documented, with code examples included.

The JMSL Library also adds financial functions to the library. The JMSL Library is designed with extensibility in mind; new classes may be derived from existing ones to add functionality to satisfy particular requirements.

## ERROR HANDLING

Diagnostic error messages are clear and informative – designed not only to convey the error condition, but also to suggest corrective action if appropriate. These error-handling features:

- Allow faster and easier program debugging

- Provide more productive programming and confidence that the algorithms are functioning properly

## COST-EFFECTIVE

The IMSL Library for Java significantly shortens program development time and promotes standardization. Using the JMSL Library saves time in the design, development, documentation, testing and maintenance of applications.

## PREDICTIVE ANALYTICS AND DATA MINING

The JMSL Library offers a comprehensive set of functions for data mining, modeling, prediction, and optimization to support every stage of the analytic process. Predictive models and time series include ARIMA, GARCH, vector auto-regression, linear and nonlinear regression, stochastic gradient boosting, and many others. JMSL stat and data mining packages include support vector machines, neural networks, decision trees, ensemble methods like random forest, hierarchical and K++ clustering, Bayes classification, and many other methods for pattern recognition and machine learning. For optimization problems, JMSL has least squares solvers, nonlinear programming, quadratic programming, dense and sparse linear programming, and many supporting functions such as numerical derivatives.

## DISTRIBUTED COMPUTING AND PARALLELISM

Distributed computing is the general term for breaking a problem down into parts and communicating the parts across a network. Parallel computing, on the other hand, refers to the simultaneous processing of parts of the larger problem.

For distributed computing, JMSL classes are easily embedded in Hadoop and Spark applications. For examples using linear regression and Apriori market basket analysis, see the white papers Using JMSL in Hadoop MapReduce applications and Using JMSL in Apache Spark.

The JMSL Library uses Java Threads to implement parallel processing. The JMSL implementation provides methods for specifying the number of threads to be used in selected classes that may benefit most from parallel processing, such as those in the Optimization, Time Series, and Data Mining categories.

## COMPREHENSIVE DOCUMENTATION

Documentation for the IMSL Library for Java is comprehensive, clearly written and standardized. Detailed information about each class consists of the name, purpose, synopsis, exceptions, return values and usage examples.

# com.imsl Package

## com.imsl

The com.imsl package contains basic informational and error handling classes.

| ERROR HANDLING | |
| --- | --- |
| **CLASS** | **DESCRIPTION** |
| **Messages** | Retrieves and formats message strings. |
| **IMSLFormatter** | Simple formatter for classes that implement logging. |
| **Warning** | Handles warning messages. |
| **WarningObject** | Handles warning messages. This class maintains a single, private, WarningObject that actually displays the warning messages. |
| **Version** | Prints the version information. |
| **IMSLException** | An Exception class for IMSL classes. |
| **IMSLRuntimeException** | A RuntimeException class for IMSL classes. |
| **IMSLUnexpectedErrorException** | An Exception class to handle critical errors. |
| **LicenseManagerException** | A LicenseManagerException exception is thrown if a license to use the product cannot be obtained. |

# com.imsl.math Package

## com.imsl.math

The com.imsl.math package contains a wide range of mathematical classes, beginning with basic data types such as complex numbers and matrices and progressing to advanced numerical analysis.

### BASIC TYPES

| CLASS | DESCRIPTION |
| --- | --- |
| **Complex** | Implements complex numbers, providing the basic operations (addition, subtraction, multiplication, division) as well as a set of complex functions. |
| **Matrix** | Matrix manipulation functions, including methods for matrix-vector, vector-matrix, matrix-matrix and vector-matrix-vector multiplications with general or symmetric matrices. |
| **ComplexMatrix** | Complex matrix manipulation functions. |
| **SparseMatrix** | Data structures and manipulation functions for sparse matrices. |
| **ComplexSparseMatrix** | Data structures and manipulation functions for complex sparse matrices. |

### LINEAR ALGEBRA

| CLASS | DESCRIPTION |
| --- | --- |
| **LU** | Solves a real general system of linear equations $Ax = b$. Includes methods for inverse, determinant and condition number. |
| **SuperLU** | Computes the LU factorization of a general sparse matrix of type SparseMatrix by a column method and solves a real sparse linear system of equations $Ax = b$. |
| **ComplexLU** | Solves a complex general system of linear equations $Ax = b$. Includes methods for inverse, determinant and condition number. |
| **ComplexSuperLU** | Computes the LU factorization of a general sparse matrix of type ComplexSparseMatrix by a column method and solves a complex sparse linear system of linear equations $Ax = b$. |
| **Cholesky** | Solves a real symmetric positive definite system of linear equations $Ax = b$. |
| **SparseCholesky** | Computes the Cholesky factorization of a matrix of type SparseMatrix. |
| **ComplexSparseCholesky** | Computes the Cholesky factorization of a matrix of type ComplexSparseMatrix. |
| **QR** | Computes the $QR$ decomposition of a matrix using Householder transformations. |

## LINEAR ALGEBRA

| CLASS | DESCRIPTION |
|-------|-------------|
| **SVD** | Computes the singular value decomposition of a real rectangular matrix $A$. |
| **ComplexSVD** | Computes the Singular Value Decomposition (SVD) of a general complex matrix. |
| **GenMinRes** | Linear system solver using the restarted generalized minimum residual (GMRES) method. |
| **ConjugateGradient** | Solves a real symmetric definite linear system using the conjugate gradient method with optional preconditioning. |

## EIGENSYSTEMS

| CLASS | DESCRIPTION |
|-------|-------------|
| **Eigen** | Computes the eigen expansion of a real matrix $A$. |
| **SymEigen** | Computes the eigen expansion of real symmetric matrix $A$. |
| **ComplexEigen** | Computes the eigenvalues and eigenvectors of a general square complex matrix. |

## INTERPOLATION AND APPROXIMATION

| CLASS | DESCRIPTION |
|-------|-------------|
| **Spline** | A collection of spline functions. Includes methods to evaluate, integrate and compute derivatives of splines. |
| **CsAkima** | Extension of the Spline class to compute an Akima cubic spline interpolant. |
| **CsTCB** | Extension of the Spline class to handle a tension-continuity-bias (TCB) cubic spline, also known as a Kochanek-Bartels spline and is a generalization of the Catmull-Rom spline. |
| **CsInterpolate** | Extension of the Spline class to compute a cubic spline interpolant with specified derivative endpoint conditions. |
| **CsPerodic** | Extension of the Spline class to compute a cubic spline interpolant with periodic boundary conditions. |
| **CsShape** | Extension of the Spline class to compute a cubic spline interpolant consistent with the concavity of the data. |
| **CsSmooth** | Extension of the Spline class to construct a smooth cubic spline to noisy data. |
| **CsSmoothC2** | Extension of the Spline class to construct a smooth cubic spline to noisy data using cross-validation to estimate the smoothing parameter. |
| **BSpline** | A collection of B-Spline functions, including methods to evaluate, integrate, and compute derivatives of B-Splines, plus conversion of B-Splines to piecewise polynomials for charting. |
| **BsInterpolate** | Extension of the BSpline class to interpolate data points. |

## INTERPOLATION AND APPROXIMATION

| CLASS | DESCRIPTION |
| --- | --- |
| **BsLeastSquares** | Extension of the BSpline class to compute a least squares B-spline approximation to data points. |
| **Spline2D** | Represents and evaluates tensor-product splines. |
| **Spline2DLeastSquares** | Computes a two-dimensional, tensor-product spline approximant using least squares. |
| **Spline2DInterpolate** | Computes a two-dimensional, tensor-product spline interpolant from two-dimensional, tensor-product data. |
| **RadialBasis** | Computes a least-squares fit to scattered data over multiple dimensions. |

## QUADRATURE

| CLASS | DESCRIPTION |
| --- | --- |
| **Quadrature** | A general-purpose integrator that uses a globally adaptive scheme to reduce the absolute error. |
| **HyperRectangleQuadrature** | Integrates a function on a hyper-rectangle using a quasi-Monte-Carlo method. |

## DIFFERENTIAL EQUATIONS

| CLASS | DESCRIPTION |
| --- | --- |
| **ODE** | ODE represents and solves an initial-value problem for ordinary differential equations. |
| **OdeRungeKutta** | Solves an initial-value problem for ordinary differential equations using the Runge-Kutta-Verner fifth-order and sixth-order methods. |
| **OdeAdamsGear** | Extension of the ODE class to solve a stiff initial-value problem for ordinary differential equations using the Adams-Gear methods. |
| **FeynmanKac** | Solves the generalized Feynman-Kac PDE. |

## TRANSFORMS

| CLASS | DESCRIPTION |
| --- | --- |
| **FFT** | Discrete Fourier transform of a real sequence. |
| **ComplexFFT** | Discrete Fourier transform of a complex sequence. |

## NONLINEAR EQUATIONS

| CLASS | DESCRIPTION |
| --- | --- |
| **ZeroPolynomial** | Finds the zeros of a polynomial with complex coefficients using Aberth's method. |
| **ZerosFunction** | Finds the real zeros of a real, continuous, univariate function, $f(x)$. |
| **ZeroSystem** | Solves a system of n nonlinear equations $f(x) = 0$ using a modified Powell hybrid algorithm. |

## OPTIMIZATION

| CLASS | DESCRIPTION |
| --- | --- |
| **MinUncon** | Finds the minimum point of a smooth function $f(x)$ of a single variable. |
| **MinUnconMultiVar** | Minimizes a function $f(x)$ of n variables using a quasi-Newton method. |

## OPTIMIZATION

| CLASS | DESCRIPTION |
| --- | --- |
| **NonlinLeastSquares** | Solves a nonlinear least-squares problem using a modified Levenberg-Marquardt algorithm. |
| **SparseLP** | Solves a sparse linear programming problem by an infeasible primal-dual interior-point method. |
| **DenseLP** | Solves a linear programming problem using an active set strategy. |
| **QuadraticProgramming** | Solves a quadratic programming problem subject to linear equality or inequality constraints. |
| **MinConGenLin** | Minimizes a general objective function subject to linear equality/inequality constraints. |
| **BoundedLeastSquares** | Solves a nonlinear least-squares problem subject to bounds on the variables using a modified Levenberg-Marquardt algorithm. |
| **BoundedVariableLeastSquares** | Solves a linear least-squares problem with bounds on the variables. |
| **NonNegativeLeastSquares** | Solves a linear least squares problem with non-negative constraints. |
| **MinConNLP** | Solves a general nonlinear programming problem using a sequential equality constrained quadratic programming method. |
| **NumericalDerivatives** | Computes the Jacobian matrix for a function $f(y)$ with m components in $n$ independent variables. |
| **NelderMead** | Solves unconstrained and box-constrained optimization problems using a direct search polytope method. |

## SPECIAL FUNCTIONS

| CLASS | DESCRIPTION |
| --- | --- |
| Sfun | Collection of special functions including beta, gamma and others. |
| Bessel | Collection of Bessel functions. |
| JMath | Pure Java implementation of the standard java.lang.Math class. |
| IEEE | Pure Java implementation of the IEEE 754 functions as specified in IEEE Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard 754-1985 (IEEE, New York). |
| Hyperbolic | Pure Java implementation of the hyperbolic functions and their inverses. |

## MISCELLANEOUS

| CLASS | DESCRIPTION |
| --- | --- |
| Physical | Returns the value of various mathematical and physical constants. |
| EpsilonAlgorithm | Determines the limit of a sequence of approximations by means of the Epsilon algorithm. |

## PRINTING

| CLASS | DESCRIPTION |
| --- | --- |
| PrintMatrix | Matrix printing utilities. |
| PrintMatrixFormat | Customizes the actions of PrintMatrix. |

# com.imsl.stat Package

## com.imsl.stat

The package com.imsl.stat contains a wide range of statistical classes, including summary statistics, regression, ANOVA.

### BASIC STATISTICS

| CLASS | DESCRIPTION |
|---|---|
| Summary | Computes basic univariate statistics. |
| NormOneSample | Computes statistics for mean and variance inferences using a sample from a normal population. |
| NormTwoSample | Computes statistics for mean and variance inferences using samples from two normal populations. |
| Sort | Sorts observations by specified keys. |
| Ranks | Computes the ranks, normal scores, or exponential scores for a vector of observations. |
| EmpiricalQuantiles | Determines empirical quantiles. |
| TableOneWay | Tallies observations into a one-way frequency table. |
| TableTwoWay | Tallies observations into a two-way frequency table. |
| TableMultiWay | Tallies observations into a multi-way frequency table. |
| WelchsTTest | Performs Welch's approximate t-test for two independent normal populations with unequal variances. |

### VARIANCES, COVARIANCES, AND CORRELATIONS

| CLASS | DESCRIPTION |
|---|---|
| Covariances | Computes the sample variance-covariance or correlation matrix. |
| PartialCovariances | Computes the partial covariances or partial correlations from an input covariance or correlation matrix. |
| PooledCovariances | Computes the pooled variance-covariance matrix from one or more sets of observations. |

## REGRESSION

| CLASS | DESCRIPTION |
|---|---|
| RegressorsForGLM | Generates regressors for a general linear model from a data matrix. |
| LinearRegression | Computes a new linear regression object using least squares. |
| NonlinearRegression | Fits a multivariate nonlinear regression model using least squares. |
| UserBasisRegression | Generates summary statistics using user supplied functions in a nonlinear regression model. |
| RegressionBasis | Interface for user defined regression basis functions. |
| SelectionRegression | Selects the best multiple linear regression models. |
| StepwiseRegression | Builds multiple linear regression models using forward selection, backward selection, or stepwise selection. |

## ANALYSIS OF VARIANCE

| CLASS | DESCRIPTION |
|---|---|
| ANOVA | Provides an Analysis of Variance table and related statistics. |
| ANOVAFactorial | Analyzes a balanced factorial design with fixed effects. |
| ANCOVA | Analyzes a one-way factorial classification model with covariates. |
| MultipleComparisons | Performs Student-Newman-Keuls multiple comparisons test. |

## CATEGORICAL AND DISCRETE DATA ANALYSIS

| CLASS | DESCRIPTION |
|---|---|
| ContingencyTable | Provides an Analysis of Variance table and related statistics. |
| CategoricalGenLinModel | Analyzes categorical data using logistic, probit, Poisson, and other generalized linear models. |

## NONPARAMETRIC STATISTICS

| CLASS | DESCRIPTION |
|---|---|
| SignTest | Performs a sign test. |
| WilcoxonRankSum | Performs a Wilcoxon rank sum test. |

## GOODNESS OF FIT

| CLASS | DESCRIPTION |
|---|---|
| **ChiSquaredTest** | Chi-squared goodness-of-fit test. |
| **NormalityTest** | Performs a test for normality. |
| **KolmogorovOneSample** | Performs a Kolmogorov-Smirnov goodness-of-fit test in one sample. |
| **KolmogorovTwoSample** | Computes Kolmogorov-Smirnov two-sample test statistics for testing that two continuous cumulative distribution functions (CDF's) are identical based upon two random samples. |

## TIME SERIES AND FORECASTING

| CLASS | DESCRIPTION |
| --- | --- |
| **AutoCorrelation** | Computes the sample autocorrelation function of a stationary time series. |
| **ARAutoUnivariate** | Automatically determines the best autoregressive time series model using Akaike's Information Criterion. |
| **ARSeasonalFit** | Estimates the optimum seasonality parameters for a time series using an autoregressive model, $AR(p)$, to represent the time series. |
| **ARMA** | Computes least-square estimates of parameters for an ARMA model. Also computes forecasts and their associated probability limits for an ARMA model. |
| **ARMAEstimateMissing** | Estimates missing values in a time series collected with equal spacing. Missing values can be replaced by these estimates prior to fitting a time series using the ARMA class. |
| **ARMAMaxLikelihood** | Computes maximum likelihood estimates of parameters for an ARMA model with $p$ and $q$ autoregressive and moving average terms respectively. |
| **ARMAOutlierIdentification** | Detects and determines outliers and simultaneously estimates the model parameters in a time series whose underlying outlier free series follows a general seasonal or nonseasonal ARMA model. |
| **AutoARIMA** | Automatically identifies time series outliers, determines parameters of a multiplicative seasonal model and produces forecasts that incorporate the effects of outliers whose effects persist beyond the end of the series. |
| **CrossCorrelation** | Computes the sample cross-correlation function of two stationary time series. |
| **Difference** | Differences a seasonal or nonseasonal time series. |
| **EGARCH** | Computes estimates of the parameters of the Exponential GARCH model. Extends the abstract class, ExtendedGARCH. |
| **ExtendedGARCH** | An abstract class with some common methods for extended GARCH models. |
| **GARCH** | Computes estimates of the parameters of a $GARCH(p, q)$ model. |
| **KalmanFilter** | Performs Kalman filtering and evaluates the likelihood function for the state-space model. |
| **MultiCrossCorrelation** | Computes the multichannel cross-correlation function of two mutually stationary multichannel time series. |
| **LackOfFit** | Performs lack-of-fit test for a univariate time series or transfer function given the appropriate correlation function. |

## TIME SERIES AND FORECASTING

| CLASS | DESCRIPTION |
|---|---|
| HoltWintersExponentialSmoothing | Calculates parameters and forecasts using the Holt-Winters Multiplicative or Additive forecasting method for seasonal data. |
| TimeSeries | A specialized class for time series data and analysis. |
| TimeSeriesOperations | Implements special methods for TimeSeries objects. |
| VectorAutoregression | Provides methods for vector autoregression (VAR). |

## MULTIVARIATE ANALYSIS

| CLASS | DESCRIPTION |
|---|---|
| ClusterKMeans | Performs a *K*-means (centroid) cluster analysis. |
| ClusterKNN | Performs a k-Nearest Neighbor classification. |
| Dissimilarities | Computes a matrix of dissimilarities (or similarities) between the columns (or rows) of a matrix. |
| ClusterHierarchical | Performs a hierarchical cluster analysis given a distance matrix. |
| FactorAnalysis | Performs Principal Component Analysis or Factor Analysis on a covariance or correlation matrix. |
| DiscriminantAnalysis | Performs a linear or a quadratic discriminant function analysis among several known groups. |

## SURVIVAL AND RELIABILTY ANALYSIS

| CLASS | DESCRIPTION |
|---|---|
| KaplanMeierECDF | Computes the Kaplan-Meier reliability function estimates or the CDF based on failure data that may be multi-censored. |
| KaplanMeierEstimates | Computes Kaplan-Meier (or product-limit) estimates of survival probabilities for a sample of failure times that possibly contain right censoring. |
| ProportionalHazards | Analyzes survival and reliability data using Cox's proportional hazards model. |
| LifeTables | Computes population (current) or cohort life tables based upon the observed population sizes at the middle (for population table) or the beginning (for cohort table) of some user specified age intervals. |

## PROBABILITY DISTRIBUTION FUNCTIONS AND INVERSES

| CLASS | DESCRIPTION |
|---|---|
| CDF | Cumulative distribution functions. |
| PDF | Probability density functions. |

## PROBABILITY DISTRIBUTION FUNCTIONS AND INVERSES

| CLASS | DESCRIPTION |
| --- | --- |
| InvCdf | Inverse cumulative probability distribution functions. |
| CdfFunction | Interface for user defined cumulative distribution function. |
| InverseCdf | Evaluates the inverse of a continuous, strictly monotone function. |
| Distribution | Interface for user defined distribution function. |
| ProbabilityDistribution | Interface for user defined probability distribution. |
| NormalDistribution | Evaluates a normal (Gaussian) probability distribution. |
| GammaDistribution | Evaluates a gamma probability distribution. |
| LogNormalDistribution | Evaluates a lognormal probability distribution. |
| PoissonDistribution | Evaluates a Poisson probability distribution. |

## RANDOM NUMBER GENERATION

| CLASS | DESCRIPTION |
| --- | --- |
| Random | Random number generators with support for several distributions. |
| FaureSequence | Generates the low-discrepancy shuffled Faure sequence. This is also called a quasi-random generator. |
| MersenneTwister | Generates uniform pseudorandom 32-bit numbers with a period of $2^{19937}-9937rs$ with a period of 2dom 32-bited Faure seque. |
| MersenneTwister64 | Generates uniform pseudorandom 64-bit numbers with a period of $2^{19937}-1$ and a 623-dimensional equidistribution property. |
| RandomSamples | Generates a simple pseudorandom sample from a finite population, a sample of indices, or a permutation of an array of indices. |
| RandomSequence | Interface implemented by generators of random or quasi-random multidimensional sequences. |

# com.imsl.datamining Package

## com.imsl.datamining

The package com.imsl.datamining contains data mining classes including Naïve Bayes classifier, Apriori, and Kohonen Self Organizing Map

| Datamining | |
|---|---|
| CLASS | DESCRIPTION |
| **NaiveBayesClassifier** | Trains a Naïve Bayes Classifier. |
| **Itemsets** | Sets of items discovered by the Apriori algorithm. |
| **AssociationRule** | Association rules generated by the Apriori algorithm. |
| **Apriori** | Perform the Apriori algorithm for association rule discovery. |
| **KohonenSOM** | A Kohonen selforganizing map. |
| **KohonenSOMTrainer** | Trains a Kohonen network. |
| **PredictiveModel** | Defines the members and methods common to predictive models in univariate prediction or classification problems. |
| **CrossValidation** | Performs V-Fold cross-validation for predictive models. |
| **BootstrapAggregation** | Performs bootstrap aggregation to generate predictions using predictive models. |
| **GradientBoosting** | Performs stochastic gradient boosting for a single response variable and multiple predictor variables. |
| **ContinuousAttribute** | Creates a continuous attribute type to be used for training a classifier. |

# com.imsl.datamining.decision Tree Package

## com.imsl.datamining.decisionTree

The package com.imsl.datamining.decisionTree contains decision tree data mining classes for CART™, C4.5, CHAID, and QUEST methods.

| Decision Trees | |
|---|---|
| **CLASS** | **DESCRIPTION** |
| **TreeNode** | A DecisionTree node which may be used as a child node of Tree. |
| **Tree** | Root node of a decision tree containing information about the relationship of child nodes. |
| **DecisionTree** | Generates a decision tree for a single response variable and two or more predictor variables. |
| **DecisionTreeInfoGain** | Extends DecisionTree for methods which use information gain criteria. |
| **ALACART** | Generates a decision tree using the CART™ method. |
| **C45** | Generates a decision tree using the C4.5 algorithm. |
| **CHAID** | Generates a decision tree using CHAID for categorical or discrete ordered predictor variables. |
| **QUEST** | Generates a decision tree using the QUEST algorithm. |
| **RandomTrees** | Performs the random forest ensemble method for decision trees. |

# com.imsl.datamining.neural Package

## com.imsl.datamining.neural

The com.imsl.datamining.neural package contains feed forward multilayer neural network training and forecasting engines plus algorithms to facilitate data pre- and post-processing.

| NEURAL NETS | |
| --- | --- |
| **CLASS** | **DESCRIPTION** |
| **Network** | A neural network. |
| **FeedForwardNetwork** | A feed forward neural network. |
| **Layer** | The base class for the input, hidden, and output layers in a neural network. |
| **InputLayer** | The input layer in a neural network. |
| **HiddenLayer** | The hidden layer in a neural network. |
| **OutputLayer** | The output layer in a neural network. |
| **Node** | A node with an input node or perceptron in a neural network. |
| **InputNode** | A node in the input layer. |
| **Perceptron** | A perceptron node in a neural network. |
| **OutputPerceptron** | A perceptron in the output layer. |
| **Activation** | Interface implemented by perceptron activation functions. |
| **Link** | A link in the neural network between two network nodes. |
| **Trainer** | Interface implemented by classes used to train a network. |
| **QuasiNewtonTrainer** | Trains a feed forward network using quasi-Newton optimization. |
| **LeastSquaresTrainer** | Trains a feed forward network using the Levenberg-Marquardt nonlinear least squares algorithm. |
| **EpochTrainer** | A two-stage trainer. Stage I trains using epochs of randomly selected training patterns. Stage II uses all training patterns and is optional. |
| **BinaryClassification** | Classifies patterns into two categories. |
| **MultiClassification** | Classifies patterns into multiple categories. |
| **ScaleFilter** | Scales or unscales continuous data prior to its use in neural network training, testing, or forecasting. |

## NEURAL NETS

| CLASS | DESCRIPTION |
|---|---|
| **UnsupervisedNominalFilter** | Encodes nominal data into a series of binary columns for input into a neural network. It also decodes binary columns in a single column of nominal values. |
| **UnsupervisedOrdinalFilter** | Encodes ordinal data into cumulative percentages. It also decodes cumulative percentages into an ordinal variable. |
| **TimeSeriesFilter** | Converts time series data to a format used for neural network training by lagging an input series into several columns, one for each lag requested. |
| **TimeSeriesClassFilter** | Converts categorized time series data to a format used for neural network training. An input series is lagged into several columns, one for each lag requested. Lagging is calculated within each class of a nominal variable. |

# com.imsl.datamining.supportvectormachine Package

## com.imsl.datamining.supportvectormachine

The com.imsl.datamining.supportvectormachine package contains classes featuring machine learning algorithms for classification, regression, and the so-called one class problem. Each formulation uses a kernel based optimization to find support vectors that optimally separate clouds of high dimensional data for the purpose of making predictions.  The classes are based on the LIBSVM package for support vector machines.

### SUPPORT VECTOR MACHINES

| CLASS | DESCRIPTION |
| --- | --- |
| DataNode | Specifies a data node for a support vector machine. |
| Kernel | Abstract class to specify a kernel function for support vector machines. |
| LinearKernel | Specifies the linear kernel for support vector machines. |
| PolynomialKernel | Specifies the polynomial kernel for support vector machines. |
| RadialBasisKernel | Specifies the radial basis kernel for support vector machines. |
| SigmoidKernel | Specifies the sigmoid kernel for support vector machines. |
| SupportVectorMachine | Abstract class for generating a support vector machine. |
| SVClassification | Specifies a support vector machine for classification (SVC). |
| SVModel | Contains model estimates after training a support vector machine. |
| SVOneClass | Specifies a support vector machine for the one class problem. |
| SVRegression | Specifies a support vector machine for regression (SVR). |

# com.imsl.stat.distributions Package

## com.imsl.stat.distributions

The com.imsl.stat.distributions package supports modeling and estimating univariate probability distributions.

| Distributions | |
|---|---|
| **CLASS** | **DESCRIPTION** |
| **MaximumLikelihoodEstimation** | Maximum likelihood parameter estimation. |
| **ProbabilityDistribution** | Defines members and methods common to univariate probability distributions and useful in parameter estimation. |
| **PDFGradientInterface** | Interface for probability distributions which supply a method to calculate the gradient of the density function. |
| **PDFHessianInterface** | Interface for probability distributions which supply methods to calculate the gradient and hessian of the density function. |
| **ClosedFormMaximumLikelihoodInterface** | Interface for probability distributions that provide a method for a closed form solution of the maximum likelihood function. |
| **BetaPD** | The beta probability distribution. |
| **GammaPD** | The gamma probability distribution. |
| **NormalPD** | The normal probability distribution. |
| **ContinuousUniformPD** | The continuous uniform probability distribution. |
| **ExponentialPD** | The exponential probability distribution. |

# com.imsl.finance Package

## com.imsl.finance

The com.imsl.finance package contains a set of classes covering a variety of investment calculations including an extensive collection of bond functions.

| Finance | |
| --- | --- |
| CLASS | DESCRIPTION |
| **BasisPart** | Interface for day count basis. |
| **Bond** | Collection of bond functions including interest, and price and yield methods. |
| **DayCountBasis** | Rules for computing the number or days between two dates or number of days in a year. For many securities, computations are based on rules other than on the actual calendar. |
| **Finance** | Collection of finance functions including depreciations, present values, and internal rate of return methods. |

# com.imsl.io Package

## com.imsl.io

The com.imsl.io package is designed to allow input/output operations.

| Input/Output | |
| --- | --- |
| **CLASS** | **DESCRIPTION** |
| **AbstractFlatFile** | Reads a text or a binary file as a ResultSet. |
| **FlatFile** | Reads a text file as a ResultSet. |
| **Tokenizer** | Breaks a line into tokens. |
| **MPSReader** | Reads a linear programming problem from an MPS file. |

# IMSL

## by Perforce

**AT PERFORCE, YOUR SUCCESS IS OUR PRIDE. IT'S TRUE WHETHER YOU'RE AN ENTERPRISE ORGANIZATION TRYING TO DEVELOP BETTER PRODUCTS FASTER, OR YOU'RE AN EMPLOYEE TRYING TO DEVELOP A BETTER CAREER. WE BELIEVE IN THE POWER OF TEAMWORK. AND THAT OUR DEVELOPMENT TOOLS SHOULD EMPOWER THE DIVERSE TEAMS THAT USE THEM BY ENHANCING COLLABORATION, CONTROL, QUALITY, SPEED, SCALE, AND FREEDOM.**