Root

Table of Contents

LinearRegression

Name	LinearRegression
Version	3.0.0
Description	Linear Regression Algorithm Bundle
License	http://www.apache.org/licenses/LICENSE-2.0
Copyright	Copyright (C) 2017 HPCC Systems
Authors	HPCCSystems
DependsOn	ML_Core, PBblas
Platform	6.2.0

Table of Contents

LinearRegression.OLS

IMPORTS

- ML_Core
- ML_Core.Types
- PBblas
- PBblas.Types
- PBblas.Converted
- PBblas.MatUtils
- ML_Core.Math

DESCRIPTIONS

MODULE : OLS

Up :

	OLS
(DATASET(NumericField) X=empty_data, DATASET(NumericField) Y=empty_data)	

Ordinary Least Squares (OLS) Linear Regression aka Ordinary Linear Regression Regression learns a function that maps a set of input data (independents) to one or more output variables (dependents). The resulting learned function is known as the model. That model can then be used repetitively to predict (i.e. estimate) the output value(s) based on new input data. Two major use cases are supported: 1) Learn and return a model 2) Use an existing (e.g. persisted) model to predict new values for Y Of course, both can be done in a single run. Alternatively, the model can be persisted and used indefinitely for prediction of Y values, as long as the record format has not changed, and the original training data remains representative of the population. OLS supports any number of independent variables (Multiple Regression) and multiple dependent variables (Multivariate Regression). In this way, multiple variables' values can be predicted from the same input (i.e. independent) data. Training data is presented as parameters to this module. When using a previously persisted model (use case 2 above), these parameters should be omitted. This module provides a rich set of analytics to assess the usefulness of the resulting linear regression model, and to determine the best subset of independent variables to include in the model. These include: For the whole model: - Analysis of Variance (ANOVA) - R-squared - Adjusted R-squared - F-Test - Akaike Information Criterion (AIC) For each coefficient: - Standard Error (SE) - T-statistic - P-value - Confidence Interval

Parameter X ||| The independent variable training data in DATASET(NumericField) format. Each observation (e.g. record) is identified by 'id', and each feature is identified by field number (i.e. 'number'). Omit this parameter when predicting from a persisted model.

Parameter Y ||| The dependent variable training data in DATASET(NumericField) format. Each observation (e.g. record) is identified by 'id', and each feature is identified by field number. Omit this parameter when predicting from a persisted model.

GetModel | Betas | Predict | makeRSQ | RSquared | AnovaRec | calcAnova | Anova | SE | TStat | AdjRSquared | AICRec | AIC | RangeVec | DistributionBase | TDistribution | FDistribution | NormalDistribution | pVal | ConfintRec | ConfInt | FTestRec | FTest |

ATTRIBUTE : GetModel

Up : OLS \

DATASET(Layout_Model)	GetModel

GetModel Returns the learned model that maps X's to Y's. In the case of OLS, the model represents a set of Betas which are the coefficients of the linear model: Beta0 * 1 + Beta1 * Field1 + Beta2 * Field2 . . . The ID of each model record specifies to which Y variable the coefficient applies. The Field Number ('number') indicates to which field of X the beta is to be applied. Field number 1 provides the intercept portion of the linear model and is always multiplied by 1. Note that if multiple work-items are provided within X and Y, there will be multiple models returned. The models can be separated by their work item id (i.e. 'wi'). A single model can be extracted from a myriad model by using e.g., model(wi=myWI_id). GetModel should not be called when predicting using a previously persisted model (i.e. when training data was not passed to the module.

Return Model in DATASET(Layout_Model) format

See ML_core/Types.Layout_Model

OVERRIDE True

FUNCTION : Betas

Up : OLS \

DATASET(NumericField)	Betas
(DATASET(Layout_Model) model=GetModel)	

Return raw Beta values as numeric fields Extracts Beta values from the model. Can be used during training and prediction phases. For use during training phase, the 'model' parameter can be omitted. GetModel will be called to retrieve the model based on the training data. For use during prediction phase, a previously persisted model should be provided. The 'number' field of the returned NumericField records specifies to which Y the coefficient applies. The 'id' field of the returned record indicates the position of the Beta value. ID = 1 provides the Beta for the constant term (i.e. the Y intercept) while subsequent values reflect the Beta for each correspondingly numbered X feature. Feature 1 corresponds to Beta with 'id' = 2 and so on. If 'model' contains multiple work-items, Separate sets of Betas will be returned for each of the 'myriad' models (distinguished by 'wi').

Parameter model ||| Optional parameter provides a model that was previously retrieved using GetModel. If omitted, GetModel will be used as the model.

Return DATASET(NumericField) containing the Beta values.

FUNCTION : Predict

Up : OLS \

DATASET(NumericField)	Predict
(DATASET(NumericField) newX, DATASET(Layout_Model) model=GetModel)	

Predict the dependent variable values (Y) for any set of independent variables (X). Returns a predicted Y values for each observation (i.e. record) of X. This supports the 'myriad' style interface in that multiple independent work items may be present in 'newX', and multiple independent models may be provided in 'model'. The resulting predicted values will also be separable by work item (i.e. wi).

Parameter newX ||| The set of observations of independent variables in DATASET(NumericField) format.

Parameter model ||| Optional. A model that was previously returned from GetModel (above). Note that a model from a previous run will only be valid if the field numbers in X are the same as when the model was learned. If this parameter is omitted, the current model will be used.

Return An estimation of the corresponding Y value for each observation of newX. Returned in DATASET(NumericField) format with field number (i.e. 'number') indicating the dependent variable that is predicted.

OVERRIDE True

TRANSFORM : makeRSQ

Up : OLS \

R2Rec	makeRSQ
(CoCoRec coco)	

ATTRIBUTE : RSquared

Up : OLS \

DATASET(R2Rec)	RSquared

RSquared Calculate the R-Squared Metric used to assess the fit of the regression line to the training data. Since the regression has chosen the best (i.e. least squared error) line matching the data, this can be thought of as a measurement of the linearity of the training data. R Squared generally varies between 0 and 1, with 1 indicating an exact linear fit, and 0 indicating that a linear fit will have no predictive power. Negative values are possible under certain conditions, and indicate that the mean(Y) will be more predictive than any linear fit. Moderate values of R squared (e.g. .5) may indicate that the relationship of X -> Y is non-linear, or that the measurement error is high relative to the linear correlation (e.g. many outliers). In the former case, increasing the dimensionality of X, such as by using polynomial variants of the features, may yield a better fit. R squared always increases when additional independent variables are added, so it should not be used to determine the optimal set of X variables to include. For that purpose, use Adjusted R Squared (below) which penalizes larger numbers of variables. Note that the result of this call is only meaningful during training phase (use case 1 above) as it is an analysis based on the training data which is not provided during a prediction-only phase.

Return DATASET(R2Rec) with one record per dependent variable, per work-item. The number field indicates the dependent variable and coresponds to the number field of the dependent (Y) variable to which it applies.

RECORD : AnovaRec

Up : OLS \

	AnovaRec

TRANSFORM : calcAnova

Up : OLS \

AnovaRec	calcAnova
(tmpRec le)	

ATTRIBUTE : Anova

Up : OLS \

	Anova

ANOVA (Analysis of Variance) report Analyzes the sources of variance. Basic ANOVA equality: Model + Error = Total Determines how much of the variance of Y is explained by the regression model, versus how much is due to the error term (i.e. unexplained variance). This attribute is only meaningful during the training phase. Provides one record per work-item. Each record provides the following statistics: - Total_SS – Total Sum of Squares (SS) variance of the dependent data - Model_SS – The SS variance represented within the model - Error_SS – The SS variance not reflected by the model (i.e. Total_SS - Error_SS) - Total_DF – The total degrees of freedom within the dependent data - Model_DF – Degrees of freedom of the model - Error_DF – Degrees of freedom of the error component - Total_MS – The Mean Square (MS) variance of the dependent data - Model_MS – The Mean Square (MS) variance represented within the model - Error_MS – The MS variance not reflected by the model - Model_F – The F-Test statistic: Model_MS / Error_MS

Return DATASET(AnovaRec), one per work-item per dependent (Y) variable The number field indicates the dependent variable to which the analysis applies.

ATTRIBUTE : SE

Up : OLS \

DATASET(NumericField)	SE

Standard Error of the Regression Coefficients Describes the variability of the regression error for each coefficient. Only meaningful during the training phase.

Return DATASET(NumericField), one record per Beta coefficient per dependent variable per work-item. The 'id' field is the coefficient number, with 1 being the Y intercept, 2 being the coefficient for the first feature, etc. The 'number' field indicates the dependent variable to which the coefficient applies.

ATTRIBUTE : TStat

Up : OLS \

DATASET(NumericField)	TStat

T-Statistic The T-statistic identifies the significance of the value of each regression coefficient. Its calculation is simply the value of the coefficient divided by the Standard Error of the coefficient. A larger absolute value of the T-statistic indicates that the coefficient is more significant. Only meaningful during the training phase.

Return DATSET(NumericField), one record per Beta coefficient per dependent variable per work-item. The 'id' field is the coefficient number, with 1 being the Y intercept, 2 being the coefficient for the first feature, etc. The number field indicates the dependent variable to which the coefficient applies.

ATTRIBUTE : AdjRSquared

Up : OLS \

DATASET(R2Rec)	AdjRSquared

Adjusted R2 Calculate Adjusted R Squared which is a scaled version of R Squared that does not arbitrarily increase with the number of features. Adjusted R2, rather than R2 should always be used when trying to determine the best set of features to include in a model. When adding features, R2 will always increase, whether or not it improves the predictive power of the model. Adjusted R2, however, will only increase with the predictive power of the model.

Return DATASET(R2Rec), one record per dependent variable per work-item. The number field indicates the dependent variable and corresponds to the number field of the dependent (Y) variable to which it applies.

RECORD : AICRec

Up : OLS \

	AICRec

ATTRIBUTE : AIC

Up : OLS \

DATASET(AICRec)	AIC

Akaike Information Criterion (AIC) Information theory based criterion for assessing Goodness of Fit (GOF). Lower values mean better fit.

Return DATASET(AICRec), one record per dependent variable per work-item. The number field indicates the dependent variable and corresponds to the number field of the dependent (Y) variable to which it applies.

RECORD : RangeVec

Up : OLS \

	RangeVec

MODULE : DistributionBase

Up : OLS \

	DistributionBase
(t_Count Nranges = 10000)	

Low | High | Density | RangeWidth | DensityV | CumulativeV | Cumulative | NTile | InvDensity | Discrete |

ATTRIBUTE : Low

Up : OLS \ DistributionBase \

	Low

ATTRIBUTE : High

Up : OLS \ DistributionBase \

	High

FUNCTION : Density

Up : OLS \ DistributionBase \

t_FieldReal	Density
(t_FieldReal t)	

ATTRIBUTE : RangeWidth

Up : OLS \ DistributionBase \

	RangeWidth

FUNCTION : DensityV

Up : OLS \ DistributionBase \

DATASET(RangeVec)	DensityV
()	

FUNCTION : CumulativeV

Up : OLS \ DistributionBase \

	CumulativeV
()	

FUNCTION : Cumulative

Up : OLS \ DistributionBase \

t_FieldReal	Cumulative
(t_FieldReal t)	

FUNCTION : NTile

Up : OLS \ DistributionBase \

t_FieldReal	NTile
(t_FieldReal Pc)	

FUNCTION : InvDensity

Up : OLS \ DistributionBase \

	InvDensity
(t_FieldReal delta)	

ATTRIBUTE : Discrete

Up : OLS \ DistributionBase \

	Discrete

MODULE : TDistribution

Up : OLS \

	TDistribution
(t_Discrete v_in,t_Count NRanges = 10000)	

DensityV | NTile | Discrete | InvDensity | High | Low | RangeWidth | Density | CumulativeV | Cumulative |

FUNCTION : DensityV

Up : OLS \ TDistribution \

DATASET(RangeVec)	DensityV
()	

OVERRIDE True

FUNCTION : NTile

Up : OLS \ TDistribution \

t_FieldReal	NTile
(t_FieldReal Pc)	

OVERRIDE True

ATTRIBUTE : Discrete

Up : OLS \ TDistribution \

	Discrete

INHERITED True

FUNCTION : InvDensity

Up : OLS \ TDistribution \

	InvDensity
(t_FieldReal delta)	

OVERRIDE True

ATTRIBUTE : High

Up : OLS \ TDistribution \

	High

OVERRIDE True

ATTRIBUTE : Low

Up : OLS \ TDistribution \

	Low

INHERITED True

ATTRIBUTE : RangeWidth

Up : OLS \ TDistribution \

	RangeWidth

OVERRIDE True

FUNCTION : Density

Up : OLS \ TDistribution \

t_FieldReal	Density
(t_FieldReal t)	

OVERRIDE True

FUNCTION : CumulativeV

Up : OLS \ TDistribution \

	CumulativeV
()	

OVERRIDE True

FUNCTION : Cumulative

Up : OLS \ TDistribution \

t_FieldReal	Cumulative
(t_FieldReal t)	

OVERRIDE True

MODULE : FDistribution

Up : OLS \

	FDistribution
(t_Discrete d1_in, t_Discrete d2_in, t_Count NRanges = 10000)	

DensityV | CumulativeV | Cumulative | NTile | InvDensity | Discrete | Low | High | RangeWidth | Density |

FUNCTION : DensityV

Up : OLS \ FDistribution \

DATASET(RangeVec)	DensityV
()	

OVERRIDE True

FUNCTION : CumulativeV

Up : OLS \ FDistribution \

	CumulativeV
()	

OVERRIDE True

FUNCTION : Cumulative

Up : OLS \ FDistribution \

t_FieldReal	Cumulative
(t_FieldReal t)	

OVERRIDE True

FUNCTION : NTile

Up : OLS \ FDistribution \

t_FieldReal	NTile
(t_FieldReal Pc)	

OVERRIDE True

FUNCTION : InvDensity

Up : OLS \ FDistribution \

	InvDensity
(t_FieldReal delta)	

INHERITED True

ATTRIBUTE : Discrete

Up : OLS \ FDistribution \

	Discrete

INHERITED True

ATTRIBUTE : Low

Up : OLS \ FDistribution \

	Low

INHERITED True

ATTRIBUTE : High

Up : OLS \ FDistribution \

	High

OVERRIDE True

ATTRIBUTE : RangeWidth

Up : OLS \ FDistribution \

	RangeWidth

OVERRIDE True

FUNCTION : Density

Up : OLS \ FDistribution \

t_FieldReal	Density
(t_FieldReal t)	

OVERRIDE True

MODULE : NormalDistribution

Up : OLS \

	NormalDistribution
(t_Count NRanges)	

Low | High | RangeWidth | DensityV | CumulativeV | Cumulative | NTile | InvDensity | Discrete | Density |

ATTRIBUTE : Low

Up : OLS \ NormalDistribution \

	Low

INHERITED True

ATTRIBUTE : High

Up : OLS \ NormalDistribution \

	High

INHERITED True

ATTRIBUTE : RangeWidth

Up : OLS \ NormalDistribution \

	RangeWidth

OVERRIDE True

FUNCTION : DensityV

Up : OLS \ NormalDistribution \

DATASET(RangeVec)	DensityV
()	

OVERRIDE True

FUNCTION : CumulativeV

Up : OLS \ NormalDistribution \

	CumulativeV
()	

OVERRIDE True

FUNCTION : Cumulative

Up : OLS \ NormalDistribution \

t_FieldReal	Cumulative
(t_FieldReal t)	

OVERRIDE True

FUNCTION : NTile

Up : OLS \ NormalDistribution \

t_FieldReal	NTile
(t_FieldReal Pc)	

OVERRIDE True

FUNCTION : InvDensity

Up : OLS \ NormalDistribution \

	InvDensity
(t_FieldReal delta)	

INHERITED True

ATTRIBUTE : Discrete

Up : OLS \ NormalDistribution \

	Discrete

INHERITED True

FUNCTION : Density

Up : OLS \ NormalDistribution \

t_FieldReal	Density
(t_FieldReal t)	

OVERRIDE True

ATTRIBUTE : pVal

Up : OLS \

	pVal

P-Value Calculate the P-value for each coefficient, which is the probability that the coefficient is insignificant (i.e. actually zero). A low P-value (e.g. .05) provides evidence that the coefficient is significant in the model. A high P-value indicates that the coefficient value should, in fact, be zero. P-value is related to the T-Statistic, and can be thought of as a normalized version of the T-Statistic. Only meaningful during the training phase.

Return DATSET(NumericField), one record per Beta coefficient per dependent variable per work-item. The 'id' field is the coefficient number, with 1 being the Y intercept, 2 being the coefficient for the first feature, etc. The number field indicates the dependent variable and corresponds to the number field of the dependent (Y) variable to which it applies.

RECORD : ConfintRec

Up : OLS \

	ConfintRec

FUNCTION : ConfInt

Up : OLS \

	ConfInt
(Types.t_fieldReal level)	

Confidence Interval The Confidence Interval determines the upper and lower bounds of each estimated coefficient given a confidence level (level) that is required. For example, one could say that there is a 95% probability (level) that the coefficient of the first independent variable is between 2.05 and 3.62. This allows error margins to be determined with the desired confidence level. If the confidence interval spans zero, it implies that the coefficient may not be significant at the specified confidence level.

Parameter level ||| The level of confidence required, expressed as a percentage from 0.0 to 100.0

Return DATASET(ConfintRec) with one record per coefficient per dependent variable per work-item. The 'id' field is the coefficient number, with 1 being the Y intercept, 2 being the coefficient for the first feature, etc. The number field indicates the dependent variable and corresponds to the number field of the dependent (Y) variable to which it applies.

RECORD : FTestRec

Up : OLS \

	FTestRec

ATTRIBUTE : FTest

Up : OLS \

DATASET(FTestRec)	FTest

F-Test Calculate the P-value for the full regression, which is the probability that all of the coefficients are insignificant (i.e. actually zero). A low P-value (e.g. .05) provides evidence that at least one coefficient is significant. A high P-value indicates that all the coefficient values should in fact be zero, implying that the regression has no statistically significant predictive power. P-value is related to the ANOVA F-Statistic, and can be thought of as a standardized version of the ANOVA F-Statistic. The F-Test and T-Test are similar, except that the T-test is used to test the significance of each coefficient, while the F-Test is used to test the significance of the entire regression. For simple linear regression (i.e. only one independent variable, the T-Test and F-Test are equivalent.

Return DATASET(FTestRec), one record per dependent variable per work-item. The number field indicates the dependent variable and corresponds to the number field of the dependent (Y) variable to which it applies.

LogisticRegression

Name	LogisticRegression
Version	1.0.0
Description	Logistic Regression implementation
License	http://www.apache.org/licenses/LICENSE-2.0
Copyright	Copyright (C) 2017 HPCC Systems
Authors	HPCCSystems
DependsOn	ML_Core, PBblas
Platform	6.2.0

Table of Contents

Extract the beta values form the model dataset
ExtractBeta_CI.ecl Extract the beta values form the model dataset
ExtractBeta_pval.ecl Extract the beta values form the model dataset
ExtractReport.ecl Extract Report records from model
LogitPredict.ecl Predict the category values with the logit function and the the supplied beta coefficients
LogitScore.ecl Calculate the score using the logit function and the the supplied beta coefficients
Model_Deviance.ecl Model Deviance
Null_Deviance.ecl Deviance for the null model, that is, a model with only an intercept
Types.ecl

LogisticRegression.BinomialConfusion

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : BinomialConfusion

Up :

DATASET(Types.Binomial_Confusion_Summary)	BinomialConfusion
(DATASET(Core_Types.Confusion_Detail) d)	

Binomial confusion matrix. Work items with multinomial responses are ignored by this function. The higher value lexically is considered to be the positive indication.

Parameter d ||| confusion detail for the work item and classifier

Return confusion matrix for a binomial classifier

LogisticRegression.BinomialLogisticRegression

IMPORTS

- LogisticRegression
- LogisticRegression.Constants
- ML_Core.Interfaces
- ML_Core.Types

DESCRIPTIONS

MODULE : BinomialLogisticRegression

Up :

	BinomialLogisticRegression
(UNSIGNED max_iter=200, REAL8 epsilon=Constants.default_epsilon, REAL8 ridge=Constants.default_ridge)	

Binomial logistic regression using iteratively re-weighted least squares.

Parameter max_iter ||| maximum number of iterations to try

Parameter epsilon ||| the minimum change in the Beta value estimate to continue

Parameter ridge ||| a value to populate a diagonal matrix that is added to a matrix help assure that the matrix is invertible.

GetModel | Classify | Report |

FUNCTION : GetModel

Up : BinomialLogisticRegression \

DATASET(Types.Layout_Model)	GetModel
(DATASET(Types.NumericField) observations, DATASET(Types.DiscreteField) classifications)	

Calculate the model to fit the observation data to the observed classes.

Parameter observations ||| the observed explanatory values

Parameter classifications ||| the observed classification used to build the model

Return the encoded model

OVERRIDE True

FUNCTION : Classify

Up : BinomialLogisticRegression \

DATASET(Types.Classify_Result)	Classify
(DATASET(Types.Layout_Model) model, DATASET(Types.NumericField) new_observations)	

Classify the observations using a model.

Parameter model ||| The model, which must be produced by a corresponding getModel function.

Parameter new_observations ||| observations to be classified

Return Classification with a confidence value

OVERRIDE True

FUNCTION : Report

Up : BinomialLogisticRegression \

DATASET(Types.Confusion_Detail)	Report
(DATASET(Types.Layout_Model) model, DATASET(Types.NumericField) observations, DATASET(Types.DiscreteField) classifications)	

Report the confusion matrix for the classifier and training data.

Parameter model ||| the encoded model

Parameter observations ||| the explanatory values.

Parameter classifications ||| the classifications associated with the observations

Return the confusion matrix showing correct and incorrect results

OVERRIDE True

LogisticRegression.Confusion

IMPORTS

- ML_Core
- ML_Core.Types
- LogisticRegression
- LogisticRegression.Types

DESCRIPTIONS

FUNCTION : Confusion

Up :

DATASET(Confusion_Detail)	Confusion
(DATASET(DiscreteField) dependents, DATASET(DiscreteField) predicts)	

Detail confusion records to compare actual versus predicted response variable values.

Parameter dependents ||| the original response values

Parameter predicts ||| the predicted responses

Return confusion counts by predicted and actual response values.

LogisticRegression.Constants

IMPORTS

DESCRIPTIONS

MODULE : Constants

Up :

	Constants

limit_card | default_epsilon | default_ridge | local_cap | id_base | id_iters | id_delta | id_correct | id_incorrect | id_stat_set | id_betas | id_betas_coef | id_betas_SE | base_builder | base_max_iter | base_epsilon | base_ind_vars | base_dep_vars | base_obs | builder_irls_local | builder_irls_global | builder_softmax |

ATTRIBUTE : limit_card

Up : Constants \

UNSIGNED2	limit_card

ATTRIBUTE : default_epsilon

Up : Constants \

REAL8	default_epsilon

ATTRIBUTE : default_ridge

Up : Constants \

REAL8	default_ridge

ATTRIBUTE : local_cap

Up : Constants \

UNSIGNED4	local_cap

ATTRIBUTE : id_base

Up : Constants \

	id_base

ATTRIBUTE : id_iters

Up : Constants \

	id_iters

ATTRIBUTE : id_delta

Up : Constants \

	id_delta

ATTRIBUTE : id_correct

Up : Constants \

	id_correct

ATTRIBUTE : id_incorrect

Up : Constants \

	id_incorrect

ATTRIBUTE : id_stat_set

Up : Constants \

	id_stat_set

ATTRIBUTE : id_betas

Up : Constants \

	id_betas

ATTRIBUTE : id_betas_coef

Up : Constants \

	id_betas_coef

ATTRIBUTE : id_betas_SE

Up : Constants \

	id_betas_SE

ATTRIBUTE : base_builder

Up : Constants \

	base_builder

ATTRIBUTE : base_max_iter

Up : Constants \

	base_max_iter

ATTRIBUTE : base_epsilon

Up : Constants \

	base_epsilon

ATTRIBUTE : base_ind_vars

Up : Constants \

	base_ind_vars

ATTRIBUTE : base_dep_vars

Up : Constants \

	base_dep_vars

ATTRIBUTE : base_obs

Up : Constants \

	base_obs

ATTRIBUTE : builder_irls_local

Up : Constants \

	builder_irls_local

ATTRIBUTE : builder_irls_global

Up : Constants \

	builder_irls_global

ATTRIBUTE : builder_softmax

Up : Constants \

	builder_softmax

LogisticRegression.DataStats

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- LogisticRegression.Constants
- ML_Core.Types

DESCRIPTIONS

FUNCTION : DataStats

Up :

DATASET(Types.Data_Info)	DataStats
(DATASET(Core_Types.NumericField) indep, DATASET(Core_Types.DiscreteField) dep, BOOLEAN field_details=FALSE)	

Information about the datasets. Without details the range for the x and y (independent and dependent) columns. Note that a column of all zero values cannot be distinguished from a missing column. When details are requested, the cardinality, minimum, and maximum values are returned. A zero cardinality is returned when the field cardinality exceeds the Constants.limit_card value.

Parameter indep ||| data set of independent variables

Parameter dep ||| data set of dependent variables

Parameter field_details ||| Boolean directive to provide field level info

LogisticRegression.Deviance_Analysis

IMPORTS

- LogisticRegression
- LogisticRegression.Types

DESCRIPTIONS

FUNCTION : Deviance_Analysis

Up :

DATASET(Types.AOD_Record)	Deviance_Analysis
(DATASET(Types.Deviance_Record) proposed, DATASET(Types.Deviance_Record) base)	

Compare deviance information for an analysis of deviance.

Parameter proposed ||| the proposed model

Parameter base ||| the base model for comparison

Return the comparison of the deviance between the models

LogisticRegression.Deviance_Detail

IMPORTS

- ML_Core
- ML_Core.Types
- LogisticRegression
- LogisticRegression.Types

DESCRIPTIONS

FUNCTION : Deviance_Detail

Up :

DATASET(Types.Observation_Deviance)	Deviance_Detail
(DATASET(Core_Types.DiscreteField) dependents, DATASET(Types.Raw_Prediction) predicts)	

Detail deviance for each observation.

Parameter dependents ||| original dependent records for the model

Parameter predicts ||| the predicted values of the response variable

Return the deviance information by observation and the log likelihood of the predicted result.

LogisticRegression.dimm

IMPORTS

- std.BLAS
- std.BLAS.Types

DESCRIPTIONS

EMBED : dimm

Up :

Types.matrix_t	dimm
(BOOLEAN transposeA, BOOLEAN transposeB, BOOLEAN diagonalA, BOOLEAN diagonalB, Types.dimension_t m, Types.dimension_t n, Types.dimension_t k, Types.value_t alpha, Types.matrix_t A, Types.matrix_t B, Types.value_t beta=0.0, Types.matrix_t C=[])	

Matrix multiply when either A or B is a diagonal and is passed as a vector. alpha*op(A) op(B) + beta*C where op() is transpose

Parameter transposeA ||| true when transpose of A is used

Parameter transposeB ||| true when transpose of B is used

Parameter diagonalA ||| true when A is the diagonal matrix

Parameter diagonalB ||| true when B is the diagonal matrix

Parameter m ||| number of rows in product

Parameter n ||| number of columns in product

Parameter k ||| number of columns/rows for the multiplier/multiplicand

Parameter alpha ||| scalar used on A

Parameter A ||| matrix A

Parameter B ||| matrix B

Parameter beta ||| scalar for matrix C

Parameter C ||| matrix C or empty

LogisticRegression.Distributions

IMPORTS

- ML_Core.Constants
- ML_Core.Math

DESCRIPTIONS

MODULE : Distributions

Up :

	Distributions

Normal_CDF | Normal_PPF | T_CDF | T_PPF | Chi2_CDF | Chi2_PPF |

FUNCTION : Normal_CDF

Up : Distributions \

REAL8	Normal_CDF
(REAL8 x)	

Cumulative Distribution of the standard normal distribution, the probability that a normal random variable will be smaller than x standard deviations above or below the mean. Taken from C/C++ Mathematical Algorithms for Scientists and Engineers, n. Shammas, McGraw-Hill, 1995

Parameter x ||| the number of standard deviations

FUNCTION : Normal_PPF

Up : Distributions \

REAL8	Normal_PPF
(REAL8 x)	

Normal Distribution Percentage Point Function. Translated from C/C++ Mathematical Algorithms for Scientists and Engineers, N. Shammas, McGraw-Hill, 1995

Parameter x ||| probability

FUNCTION : T_CDF

Up : Distributions \

REAL8	T_CDF
(REAL8 x, REAL8 df)	

Students t distribution integral evaluated between negative infinity and x. Translated from NIST SEL DATAPAC Fortran TCDF.f source

Parameter x ||| value of the evaluation

Parameter df ||| degrees of freedom

FUNCTION : T_PPF

Up : Distributions \

REAL8	T_PPF
(REAL8 x, REAL8 df)	

Percentage point function for the T distribution. Translated from NIST SEL DATAPAC Fortran TPPF.f source

FUNCTION : Chi2_CDF

Up : Distributions \

REAL8	Chi2_CDF
(REAL8 x, REAL8 df)	

The cumulative distribution function for the Chi Square distribution. the CDF for the specfied degrees of freedom. Translated from the NIST SEL DATAPAC Fortran subroutine CHSCDF.

FUNCTION : Chi2_PPF

Up : Distributions \

REAL8	Chi2_PPF
(REAL8 x, REAL8 df)	

The Chi Squared PPF function. Translated from the NIST SEL DATAPAC Fortran subroutine CHSPPF.

LogisticRegression.ExtractBeta

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : ExtractBeta

Up :

	ExtractBeta
(DATASET(Core_Types.Layout_Model) mod_ds)	

Extract the beta values form the model dataset.

Parameter mod_ds ||| the model dataset

Return a beta values as Model Coefficient records, zero as the constant term.

LogisticRegression.ExtractBeta_CI

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : ExtractBeta_CI

Up :

DATASET(Types.Confidence_Model_Coef)	ExtractBeta_CI
(DATASET(Core_Types.Layout_Model) mod_ds, REAL8 level)	

Extract the beta values form the model dataset.

Parameter mod_ds ||| the model dataset

Parameter level ||| the significance value for the intervals

Return the beta values with confidence intervals term.

LogisticRegression.ExtractBeta_pval

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : ExtractBeta_pval

Up :

DATASET(Types.pval_Model_Coef)	ExtractBeta_pval
(DATASET(Core_Types.Layout_Model) mod_ds)	

Extract the beta values form the model dataset.

Parameter mod_ds ||| the model dataset

Return the beta values with p-values as Model Coefficient records, zero as the constant term.

LogisticRegression.ExtractReport

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- LogisticRegression.Constants
- ML_Core.Types

DESCRIPTIONS

FUNCTION : ExtractReport

Up :

DATASET(Types.Model_Report)	ExtractReport
(DATASET(Core_Types.Layout_Model) mod_ds)	

Extract Report records from model

Parameter mod_ds ||| the model dataset

Return the model report dataset

LogisticRegression.LogitPredict

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : LogitPredict

Up :

DATASET(Classify_Result)	LogitPredict
(DATASET(Model_Coef) coef, DATASET(NumericField) independents)	

Predict the category values with the logit function and the the supplied beta coefficients.

Parameter coef ||| the model beta coefficients

Parameter independents ||| the observations

Return the predicted category values and a confidence score

LogisticRegression.LogitScore

IMPORTS

- LogisticRegression
- LogisticRegression.Types
- ML_Core.Types

DESCRIPTIONS

FUNCTION : LogitScore

Up :

DATASET(Raw_Prediction)	LogitScore
(DATASET(Model_Coef) coef, DATASET(NumericField) independents)	

Calculate the score using the logit function and the the supplied beta coefficients.

Parameter coef ||| the model beta coefficients

Parameter independents ||| the observations

Return the raw prediction value

LogisticRegression.Model_Deviance

IMPORTS

- LogisticRegression
- LogisticRegression.Types

DESCRIPTIONS

FUNCTION : Model_Deviance

Up :

DATASET(Types.Deviance_Record)	Model_Deviance
(DATASET(Types.Observation_Deviance) od, DATASET(Types.Model_Coef) mod)	

Model Deviance.

Parameter od ||| observation deviance record

Parameter mod ||| model co-efficients

Return model deviance

LogisticRegression.Null_Deviance

IMPORTS

- LogisticRegression
- LogisticRegression.Types

DESCRIPTIONS

FUNCTION : Null_Deviance

Up :

DATASET(Types.Deviance_Record)	Null_Deviance
(DATASET(Types.Observation_Deviance) od)	

Deviance for the null model, that is, a model with only an intercept.

Parameter od ||| Observation Deviance record set.

Return a data set of the null model deviances for each work item and classifier.

LogisticRegression.Types

IMPORTS

- ML_Core.Types

DESCRIPTIONS

MODULE : Types

Up :

	Types

t_Universe | Field_Desc | Data_Info | NumericField_U | DiscreteField_U | Layout_Column_Map | Classifier_Stats | Model_Report | Binomial_Confusion_Summary | Model_Coef | Confidence_Model_Coef | pval_Model_Coef | Raw_Prediction | Observation_Deviance | Deviance_Record | AOD_Record |

ATTRIBUTE : t_Universe

Up : Types \

	t_Universe

RECORD : Field_Desc

Up : Types \

	Field_Desc

RECORD : Data_Info

Up : Types \

	Data_Info

RECORD : NumericField_U

Up : Types \

	NumericField_U

RECORD : DiscreteField_U

Up : Types \

	DiscreteField_U

RECORD : Layout_Column_Map

Up : Types \

	Layout_Column_Map

RECORD : Classifier_Stats

Up : Types \

	Classifier_Stats

RECORD : Model_Report

Up : Types \

	Model_Report

RECORD : Binomial_Confusion_Summary

Up : Types \

	Binomial_Confusion_Summary

RECORD : Model_Coef

Up : Types \

	Model_Coef

RECORD : Confidence_Model_Coef

Up : Types \

	Confidence_Model_Coef

RECORD : pval_Model_Coef

Up : Types \

	pval_Model_Coef

RECORD : Raw_Prediction

Up : Types \

	Raw_Prediction

RECORD : Observation_Deviance

Up : Types \

	Observation_Deviance

RECORD : Deviance_Record

Up : Types \

	Deviance_Record

RECORD : AOD_Record

Up : Types \

	AOD_Record

ML_Core

Name	ML_Core
Version	3.1.0
Description	Common definitions for Machine Learning
License	See LICENSE.TXT
Copyright	Copyright (C) 2017 HPCC Systems
Authors	HPCCSystems
Platform	6.2.0

Table of Contents

ML_Core.AppendID

IMPORTS

DESCRIPTIONS

MACRO : AppendID

Up :

	AppendID
(dIn,idfield,dOut)	

ML_Core.AppendSeqID

IMPORTS

DESCRIPTIONS

MACRO : AppendSeqID

Up :

	AppendSeqID
(dIn,idfield,dOut)	

ML_Core.Config

IMPORTS

DESCRIPTIONS

MODULE : Config

Up :

	Config

MaxLookup | Discrete | RoundingError |

ATTRIBUTE : MaxLookup

Up : Config \

	MaxLookup

ATTRIBUTE : Discrete

Up : Config \

	Discrete

ATTRIBUTE : RoundingError

Up : Config \

	RoundingError

ML_Core.Constants

IMPORTS

DESCRIPTIONS

MODULE : Constants

Up :

	Constants

Useful constants

Pi | Root_2 |

ATTRIBUTE : Pi

Up : Constants \

	Pi

Constant PI

ATTRIBUTE : Root_2

Up : Constants \

	Root_2

Constant square root of 2

ML_Core.Discretize

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

MODULE : Discretize

Up :

	Discretize

c_Method | r_Method | i_ByRounding | ByRounding | i_ByBucketing | ByBucketing | i_ByTiling | ByTiling | Do |

ATTRIBUTE : c_Method

Up : Discretize \

	c_Method

RECORD : r_Method

Up : Discretize \

r_Method

FUNCTION : i_ByRounding

Up : Discretize \

i_ByRounding

(SET OF Types.t_FieldNumber f, REAL Scale=1.0,REAL Delta=0.0)

FUNCTION : ByRounding

Up : Discretize \

ByRounding

(DATASET(Types.NumericField) d,REAL Scale=1.0, REAL Delta=0.0)

FUNCTION : i_ByBucketing

Up : Discretize \

i_ByBucketing

(SET OF Types.t_FieldNumber f, Types.t_Discrete N=ML_Core.Config.Discrete)

FUNCTION : ByBucketing

Up : Discretize \

	ByBucketing
(DATASET(Types.NumericField) d, Types.t_Discrete N=ML_Core.Config.Discrete)	

FUNCTION : i_ByTiling

Up : Discretize \

	i_ByTiling
(SET OF Types.t_FieldNumber f, Types.t_Discrete N=ML_Core.Config.Discrete)	

FUNCTION : ByTiling

Up : Discretize \

	ByTiling
(DATASET(Types.NumericField) d, Types.t_Discrete N=ML_Core.Config.Discrete)	

FUNCTION : Do

Up : Discretize \

	Do
(DATASET(Types.NumericField) d, DATASET(r_Method) to_do)	

ML_Core.FieldAggregates

IMPORTS

- ML_Core
- ML_Core.Types
- ML_Core.Utils
- std.system.ThorLib

DESCRIPTIONS

MODULE : FieldAggregates

Up :

	FieldAggregates
(DATASET(Types.NumericField) d)	

Simple | SimpleRanked | Medians | MinMedNext | Buckets | BucketRanges | Modes | Cardinality | RankedInput | NTiles | NTileRanges |

ATTRIBUTE : Simple

Up : FieldAggregates \

	Simple

ATTRIBUTE : SimpleRanked

Up : FieldAggregates \

	SimpleRanked

ATTRIBUTE : Medians

Up : FieldAggregates \

	Medians

ATTRIBUTE : MinMedNext

Up : FieldAggregates \

	MinMedNext

FUNCTION : Buckets

Up : FieldAggregates \

<table>
<tr><td></td><td>Buckets</td></tr>
<tr><td colspan="2">(Types.t_Discrete n)</td></tr>
</table>

FUNCTION : BucketRanges

Up : FieldAggregates \

	BucketRanges
(Types.t_Discrete n)	

ATTRIBUTE : Modes

Up : FieldAggregates \

	Modes

ATTRIBUTE : Cardinality

Up : FieldAggregates \

	Cardinality

ATTRIBUTE : RankedInput

Up : FieldAggregates \

	RankedInput

FUNCTION : NTiles

Up : FieldAggregates \

	NTiles
(Types.t_Discrete n)	

FUNCTION : NTileRanges

Up : FieldAggregates \

	NTileRanges
(Types.t_Discrete n)	

ML_Core.FromField

IMPORTS

DESCRIPTIONS

MACRO : FromField

Up :

	FromField
(dIn,lOut,dOut,dMap=”)	

ML_Core.Generate

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

MODULE : Generate

Up :

	Generate

tp_Method | MethodName | ToPoly |

ATTRIBUTE : tp_Method

Up : Generate \

	tp_Method

FUNCTION : MethodName

Up : Generate \

	MethodName
(tp_Method x)	

FUNCTION : ToPoly

Up : Generate \

	ToPoly
(DATASET(Types.NumericField) seedCol, UNSIGNED maxN=6)	

ML_Core.ToField

IMPORTS

DESCRIPTIONS

MACRO : ToField

Up :

	ToField
(dIn,dOut,idfield=”, wifield=”, wivalue=”,datafields=”)	

ML_Core.Types

IMPORTS

DESCRIPTIONS

MODULE : Types

Up :

	Types

t_RecordID | t_FieldNumber | t_FieldReal | t_FieldSign | t_Discrete | t_Item | t_Count | t_Work_Item | AnyField | NumericField | DiscreteField | Layout_Model | Classify_Result | l_result | Confusion_Detail | ItemElement | t_node | t_level | NodeID |

ATTRIBUTE : t_RecordID

Up : Types \

	t_RecordID

ATTRIBUTE : t_FieldNumber

Up : Types \

	t_FieldNumber

ATTRIBUTE : t_FieldReal

Up : Types \

	t_FieldReal

ATTRIBUTE : t_FieldSign

Up : Types \

	t_FieldSign

ATTRIBUTE : t_Discrete

Up : Types \

	t_Discrete

ATTRIBUTE : t_Item

Up : Types \

	t_Item

ATTRIBUTE : t_Count

Up : Types \

	t_Count

ATTRIBUTE : t_Work_Item

Up : Types \

	t_Work_Item

RECORD : AnyField

Up : Types \

	AnyField

RECORD : NumericField

Up : Types \

	NumericField

RECORD : DiscreteField

Up : Types \

	DiscreteField

RECORD : Layout_Model

Up : Types \

	Layout_Model

RECORD : Classify_Result

Up : Types \

	Classify_Result

RECORD : l_result

Up : Types \

	l_result

RECORD : Confusion_Detail

Up : Types \

	Confusion_Detail

RECORD : ItemElement

Up : Types \

	ItemElement

ATTRIBUTE : t_node

Up : Types \

	t_node

ATTRIBUTE : t_level

Up : Types \

	t_level

RECORD : NodeID

Up : Types \

	NodeID

Interfaces

Table of Contents

ML_Core.Interfaces.IClassify

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

MODULE : IClassify

Up :

	IClassify

Interface definition for Classification. Actual implementation modules will probably take parameters.

GetModel | Classify | Report |

FUNCTION : GetModel

Up : IClassify \

DATASET(Types.Layout_Model)	GetModel
(DATASET(Types.NumericField) observations, DATASET(Types.DiscreteField) classifications)	

Calculate the model to fit the observation data to the observed classes.

Parameter observations ||| the observed explanatory values

Parameter classifications ||| the observed classification used to build the model

Return the encoded model

FUNCTION : Classify

Up : IClassify \

DATASET(Types.Classify_Result)	Classify
(DATASET(Types.Layout_Model) model, DATASET(Types.NumericField) new_observations)	

Classify the observations using a model.

Parameter model ||| The model, which must be produced by a corresponding getModel function.

Parameter new_observations ||| observations to be classified

Return Classification with a confidence value

FUNCTION : Report

Up : IClassify \

DATASET(Types.Confusion_Detail)	Report
(DATASET(Types.Layout_Model) model, DATASET(Types.NumericField) observations, DATASET(Types.DiscreteField) classifications)	

Report the confusion matrix for the classifier and training data.

Parameter model ||| the encoded model

Parameter observations ||| the explanatory values.

Parameter classifications ||| the classifications associated with the observations

Return the confusion matrix showing correct and incorrect results

---

ML_Core.Interfaces.IRegression

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

MODULE : IRegression

Up :

<table>
<tr><td></td><td>IRegression</td></tr>
<tr><td colspan="2">(DATASET(NumericField) X=empty_data, DATASET(NumericField) Y=empty_data)</td></tr>
</table>

Interface Definition for Regression Modules Regression learns a function that maps a set of input data to one or more output variables. The resulting learned function is known as the model. That model can then be used repetitively to predict (i.e. estimate) the output value(s) based on new input data.

Parameter X ||| The independent data in DATASET(NumericField) format. Each statistical unit (e.g. record) is identified by 'id', and each feature is identified by field number (i.e. 'number').

Parameter Y ||| The dependent variable(s) in DATASET(NumericField) format. Each statistical unit (e.g. record) is identified by 'id', and each feature is identified by field number (i.e. 'number').

GetModel | Predict |

ATTRIBUTE : GetModel

Up : IRegression \

DATASET(Layout_Model)	GetModel

Calculate and return the 'learned' model The model may be persisted and later used to make predictions using 'Predict' below.

Return DATASET(LayoutModel) describing the learned model parameters

---

FUNCTION : Predict

Up : IRegression \

DATASET(NumericField)	Predict
(DATASET(NumericField) newX, DATASET(Layout_Model) model)	

Predict the output variable(s) based on a previously learned model

Parameter newX ||| DATASET(NumericField) containing the X values to b predicted.

Return DATASET(NumericField) containing one entry per observation (i.e. id) in newX. This represents the predicted values for Y.

Math

Table of Contents

ML_Core.Math.Beta

IMPORTS

- ML_Core.Math

DESCRIPTIONS

FUNCTION : Beta

Up :

	Beta
(REAL8 x, REAL8 y)	

Return the beta value of two positive real numbers, x and y

Parameter x ||| the value of the first number

Parameter y ||| the value of the second number

Return the beta value

ML_Core.Math.Distributions

IMPORTS

- ML_Core.Constants
- ML_Core.Math

DESCRIPTIONS

MODULE : Distributions

Up :

	Distributions

Normal_CDF | Normal_PPF | T_CDF | T_PPF | Chi2_CDF | Chi2_PPF |

FUNCTION : Normal_CDF

Up : Distributions \

REAL8	Normal_CDF
(REAL8 x)	

Cumulative Distribution of the standard normal distribution, the probability that a normal random variable will be smaller than x standard deviations above or below the mean. Taken from C/C++ Mathematical Algorithms for Scientists and Engineers, n. Shammas, McGraw-Hill, 1995

Parameter x ||| the number of standard deviations

FUNCTION : Normal_PPF

Up : Distributions \

REAL8	Normal_PPF
(REAL8 x)	

Normal Distribution Percentage Point Function. Translated from C/C++ Mathematical Algorithms for Scientists and Engineers, N. Shammas, McGraw-Hill, 1995

Parameter x ||| probability

FUNCTION : T_CDF

Up : Distributions \

REAL8	T_CDF
(REAL8 x, REAL8 df)	

Students t distribution integral evaluated between negative infinity and x. Translated from NIST SEL DATAPAC Fortran TCDF.f source

Parameter x ||| value of the evaluation

Parameter df ||| degrees of freedom

FUNCTION : T_PPF

Up : Distributions \

REAL8	T_PPF
(REAL8 x, REAL8 df)	

Percentage point function for the T distribution. Translated from NIST SEL DATAPAC Fortran TPPF.f source

---

FUNCTION : Chi2_CDF

Up : Distributions \

REAL8	Chi2_CDF
(REAL8 x, REAL8 df)	

The cumulative distribution function for the Chi Square distribution. the CDF for the specfied degrees of freedom. Translated from the NIST SEL DATAPAC Fortran subroutine CHSCDF.

FUNCTION : Chi2_PPF

Up : Distributions \

REAL8	Chi2_PPF
(REAL8 x, REAL8 df)	

The Chi Squared PPF function. Translated from the NIST SEL DATAPAC Fortran subroutine CHSPPF.

ML_Core.Math.DoubleFac

IMPORTS

DESCRIPTIONS

EMBED : DoubleFac

Up :

REAL8	DoubleFac
(INTEGER2 i)	

The ’double’ factorial is defined for ODD n and is the product of all the odd numbers up to and including that number. We are extending the meaning to even numbers to mean the product of the even numbers up to and including that number. Thus DoubleFac(8) = 8*6*4*2 We also defend against i < 2 (returning 1.0)

Parameter i ||| the value used in the calculation

Return the factorial of the sequence, declining by 2

ML_Core.Math.Fac

IMPORTS

DESCRIPTIONS

EMBED : Fac

Up :

REAL8	Fac
(UNSIGNED2 i)	

Factorial function

Parameter i ||| the value used, (i)(i-1)(i-2). . . (2)

Return the factorial i!

ML_Core.Math.gamma

IMPORTS

DESCRIPTIONS

EMBED : gamma

Up :

REAL8	gamma
(REAL8 x)	

Return the value of gamma function of real number x A wrapper for the standard C tgamma function.

Parameter x ||| the input x

Return the value of GAMMA evaluated at x

ML_Core.Math.log_gamma

IMPORTS

DESCRIPTIONS

EMBED : log_gamma

Up :

REAL8	log_gamma
(REAL8 x)	

Return the value of the log gamma function of the absolute value of X. A wrapper for the standard C lgamma function. Avoids the race condition found on some platforms by taking the absolute value of the of the input argument.

Parameter x ||| the input x

Return the value of the log of the GAMMA evaluated at ABS(x)

ML_Core.Math.lowerGamma

IMPORTS

DESCRIPTIONS

EMBED : lowerGamma

Up :

REAL8	lowerGamma
(REAL8 x, REAL8 y)	

Return the lower incomplete gamma value of two real numbers, x and y

Parameter x ||| the value of the first number

Parameter y ||| the value of the second number

Return the lower incomplete gamma value

ML_Core.Math.NCK

IMPORTS

- ML_Core.Math

DESCRIPTIONS

FUNCTION : NCK

Up :

REAL8	NCK
(INTEGER2 N, INTEGER2 K)	

ML_Core.Math.Poly

IMPORTS

DESCRIPTIONS

EMBED : Poly

Up :

REAL8	Poly
(REAL8 x, SET OF REAL8 Coeffs)	

Evaluate a polynomial from a set of co-effs. Co-effs 1 is assumed to be the HIGH order of the equation. Thus for ax^2+bx+c - the set would need to be Coef := [a,b,c];

Parameter x ||| the value of x in the polynomial

Parameter Coeffs ||| a set of coefficients forthe polynomial. The ALL set is considered to be all zero values

Return value of the polynomial at x

ML_Core.Math.StirlingFormula

IMPORTS

- ML_Core.Math
- ML_Core.Constants

DESCRIPTIONS

FUNCTION : StirlingFormula

Up :

	StirlingFormula
(REAL x)	

Stirling's formula

Parameter x ||| the point of evaluation

Return evaluation result

ML_Core.Math.upperGamma

IMPORTS

DESCRIPTIONS

EMBED : upperGamma

Up :

REAL8	upperGamma
(REAL8 x, REAL8 y)	

Return the upper incomplete gamma value of two real numbers, x and y.

Parameter x ||| the value of the first number

Parameter y ||| the value of the second number

Return the upper incomplete gamma value

Tests

Table of Contents

ML_Core.Tests.Check_Dist

IMPORTS

- ML_Core.Math.Distributions
- ML_Core
- python

DESCRIPTIONS

ATTRIBUTE : Check_Dist

Up :

	Check_Dist

ML_Core.Tests.field_aggregates

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

ATTRIBUTE : field_aggregates

Up :

	field_aggregates

ML_Core.Tests.generate

IMPORTS

- ML_Core

DESCRIPTIONS

ATTRIBUTE : generate

Up :

	generate

ML_Core.Tests.test_appends

IMPORTS

- ML_Core
- std.system.thorlib

DESCRIPTIONS

ATTRIBUTE : test_appends

Up :

	test_appends

ML_Core.Tests.test_discrete

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

ATTRIBUTE : test_discrete

Up :

	test_discrete

ML_Core.Tests.to_from

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

ATTRIBUTE : to_from

Up :

	to_from

ML_Core.Tests.Validate_Betas

IMPORTS

- ML_Core
- ML_Core.Math
- python

DESCRIPTIONS

ATTRIBUTE : Validate_Betas

Up :

	Validate_Betas

ML_Core.Tests.Validate_Gammas

IMPORTS

- ML_Core
- ML_Core.Math
- python

DESCRIPTIONS

ATTRIBUTE : Validate_Gammas

Up :

	Validate_Gammas

Utils

Table of Contents

ML_Core.Utils.Fat

IMPORTS

- ML_Core.Types

DESCRIPTIONS

FUNCTION : Fat

Up :

DATASET(Types.NumericField)	Fat
(DATASET(Types.NumericField) d0, Types.t_FieldReal v=0)	

Will take a potentially sparse file d and fill in the missing with value v for Numeric Field datasets

Parameter d0 ||| They myriad format Numeric Field dataset to be filled

Parameter v ||| The value to assign missing records

Return A full Numeric Field dataset with every field populated

ML_Core.Utils.FatD

IMPORTS

- ML_Core.Types

DESCRIPTIONS

FUNCTION : FatD

Up :

DATASET(Types.DiscreteField)	FatD
(DATASET(Types.DiscreteField) d0, Types.t_Discrete v=0)	

Will take a potentially sparse file d and fill in the missing with value v for Discrete Field datasets

Parameter d0 ||| They myriad format Discrete Field dataset to be filled

Parameter v ||| The value to assign missing records

Return A full Discrete Field dataset with every field populated

---

ML_Core.Utils.Gini

IMPORTS

DESCRIPTIONS

MACRO : Gini

Up :

	Gini
(infile, pivot, target, wi_name=’wi’)	

Creates a file of pivot/target pairs with a Gini impurity value.

Parameter infile ||| the input file, any type with a work item field

Parameter pivot ||| the name of the pivot field

Parameter target ||| the name of the field used as the target

Parameter wi_name ||| the name of the work item field, default is ”wi” return A table by Work Item and Pivot value giving count and Gini impurity value

ML_Core.Utils.SequenceInField

IMPORTS

DESCRIPTIONS

MACRO : SequenceInField

Up :

	SequenceInField
(infile,infield,seq,wi_name=’wi’)	

Given a file which is sorted by the work item identifier and INFIELD (and possibly other values), add sequence numbers within the range of each infield. Slighly elaborate code is to avoid having to partition the data to one value of infield per node and to work with very large numbers of records where a global count project would be inappropriate. This is useful for assigning rank positions with the groupings.

Parameter infile ||| the input file, any type

Parameter infield ||| field name of grouping field

Parameter seq ||| name of the field to receive the sequence number

Parameter wi_name ||| work item field name, default is wi

Return a file of the same type with sequence numbers applied

PBblas

Name	PBblas
Version	3.0.1
Description	Parallel Block Basic Linear Algebra Subsystem
License	http://www.apache.org/licenses/LICENSE-2.0
Copyright	Copyright (C) 2016, 2017 HPCC Systems
Authors	HPCCSystems
DependsOn	ML_Core
Platform	6.2.0

Table of Contents

Element-wise multiplication of X * Y
IElementFunc.ecl Function prototype for a function to apply to each element of the
MatUtils.ecl Provides various utility attributes for manipulating cell-based matrixes
potrf.ecl Implements Cholesky factorization of A = U**T * U if Triangular.Upper requested or A = L * L**T if Triangualr.Lower is requested
scal.ecl Scale a matrix by a constant Result is alpha * X This supports a ”myriad” style interface in that X may be a set of independent matrices separated by different work-item ids
tran.ecl Transpose a matrix and sum into base matrix
trsm.ecl Partitioned block parallel triangular matrix solver
Types.ecl Types for the Parallel Block Basic Linear Algebra Sub-programs support WARNING: attributes marked with WARNING can not be changed without making corresponding changes to the C++ attributes
Vector2Diag.ecl Convert a vector into a diagonal matrix

PBblas.Apply2Elements

IMPORTS

- PBblas
- PBblas.Types
- std.BLAS

DESCRIPTIONS

FUNCTION : Apply2Elements

Up :

DATASET(Layout_Cell)	Apply2Elements
(DATASET(Layout_Cell) X, IElementFunc f)	

Apply a function to each element of the matrix Use PBblas.IElementFunc as the prototype function. Input and ouput may be a single matrix, or myriad matrixes with different work item ids.

Parameter X ||| A matrix (or multiple matrices) in Layout_Cell form

Parameter f ||| A function based on the IElementFunc prototype

Return A matrix (or multiple matrices) in Layout_Cell form

See PBblas/IElementFunc

See PBblas/Types.Layout_Cell

PBblas.asum

IMPORTS

- PBblas
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims
- PBblas.internal.Converted
- std.BLAS

DESCRIPTIONS

FUNCTION : asum

Up :

DATASET(Layout_Norm)	asum
(DATASET(Layout_Cell) X)	

Absolute sum – the ”Entrywise” 1-norm Compute SUM(ABS(X))

Parameter X ||| Matrix or set of matrices in Layout_Cell format

Return DATASET(Layout_Norm) with one record per work item

See PBblas/Types.Layout_Cell

PBblas.axpy

IMPORTS

- PBblas
- PBblas.Types

DESCRIPTIONS

FUNCTION : axpy

Up :

DATASET(Layout_Cell)	axpy
(value_t alpha, DATASET(Layout_Cell) X, DATASET(Layout_Cell) Y)	

Implements alpha*X + Y X and Y must have same shape

Parameter alpha ||| Scalar multiplier for the X matrix

Parameter X ||| X matrix in DATASET(Layout_Cell) form

Parameter Y ||| Y matrix in DATASET(Layout_Call) form

Return Matrix in DATASET(Layout_Cell) form

See PBblas/Types.layout_cell

PBblas.Constants

IMPORTS

DESCRIPTIONS

MODULE : Constants

Up :

	Constants

Block_Minimum | Block_NoSplit | Block_Maximum | Block_Vec_Rows | Dimension_Incompat | Dimension_IncompatZ | Distribution_Error | Distribution_ErrorZ | Not_Square | Not_SquareZ | Not_PositiveDef | Not_PositiveDefZ | Not_Single_Block | Not_Single_BlockZ | Not_Block_Vector | Not_Block_VectorZ |

ATTRIBUTE : Block_Minimum

Up : Constants \

	Block_Minimum

ATTRIBUTE : Block_NoSplit

Up : Constants \

	Block_NoSplit

ATTRIBUTE : Block_Maximum

Up : Constants \

	Block_Maximum

ATTRIBUTE : Block_Vec_Rows

Up : Constants \

	Block_Vec_Rows

ATTRIBUTE : Dimension_Incompat

Up : Constants \

	Dimension_Incompat

ATTRIBUTE : Dimension_IncompatZ

Up : Constants \

	Dimension_IncompatZ

ATTRIBUTE : Distribution_Error

Up : Constants \

	Distribution_Error

ATTRIBUTE : Distribution_ErrorZ

Up : Constants \

	Distribution_ErrorZ

ATTRIBUTE : Not_Square

Up : Constants \

	Not_Square

ATTRIBUTE : Not_SquareZ

Up : Constants \

	Not_SquareZ

ATTRIBUTE : Not_PositiveDef

Up : Constants \

	Not_PositiveDef

ATTRIBUTE : Not_PositiveDefZ

Up : Constants \

	Not_PositiveDefZ

ATTRIBUTE : Not_Single_Block

Up : Constants \

	Not_Single_Block

ATTRIBUTE : Not_Single_BlockZ

Up : Constants \

	Not_Single_BlockZ

ATTRIBUTE : Not_Block_Vector

Up : Constants \

	Not_Block_Vector

ATTRIBUTE : Not_Block_VectorZ

Up : Constants \

	Not_Block_VectorZ

PBblas.Converted

IMPORTS

- PBblas
- PBblas.Types
- ML_Core.Types

DESCRIPTIONS

MODULE : Converted

Up :

	Converted

Module to convert between ML_Core/Types Field layouts (i.e. NumericField and DiscreteField) and PBblas matrix layout (i.e. Layout_Cell)

NFToMatrix | DFToMatrix | MatrixToNF | MatrixToDF |

FUNCTION : NFToMatrix

Up : Converted \

DATASET(Layout_Cell)	NFToMatrix
(DATASET(NumericField) recs)	

Convert NumericField dataset to Matrix

Parameter recs ||| Record Dataset in DATASET(NumericField) format

Return Matrix in DATASET(Layout_Cell) format

See PBblas/Types.Layout_Cell

See ML_Core/Types.NumericField

FUNCTION : DFToMatrix

Up : Converted \

DATASET(Layout_Cell)	DFToMatrix
(DATASET(DiscreteField) recs)	

Convert DiscreteField dataset to Matrix

Parameter recs ||| Record Dataset in DATASET(DiscreteField) format

Return Matrix in DATASET(Layout_Cell) format

See PBblas/Types.Layout_Cell

See ML_Core/Types.DiscreteField

FUNCTION : MatrixToNF

Up : Converted \

DATASET(NumericField)	MatrixToNF
(DATASET(Layout_Cell) mat)	

Convert Matrix to NumericField dataset

Parameter mat ||| Matrix in DATASET(Layout_Cell) format

Return NumericField Dataset

See PBblas/Types.Layout_Cell

See ML_Core/Types.NumericField

FUNCTION : MatrixToDF

Up : Converted \

DATASET(DiscreteField)	MatrixToDF
(DATASET(Layout_Cell) mat)	

Convert Matrix to DiscreteField dataset

Parameter mat ||| Matrix in DATASET(Layout_Cell) format

Return DiscreteField Dataset

See PBblas/Types.Layout_Cell

See ML_Core/Types.DiscreteField

PBblas.ExtractTri

IMPORTS

- PBblas
- std.BLAS
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims
- PBblas.internal.Converted

DESCRIPTIONS

FUNCTION : ExtractTri

Up :

DATASET(Layout_Cell)	ExtractTri
(Triangle tri, Diagonal dt, DATASET(Layout_Cell) A)	

Extract the upper or lower triangle from the composite output from getrf (LU Factorization).

Parameter tri ||| Triangle type: Upper or Lower (see Types.Triangle)

Parameter dt ||| Diagonal type: Unit or non unit (see Types.Diagonal)

Parameter A ||| Matrix of cells. See Types.Layout_Cell

Return Matrix of cells in Layout_Cell format representing a triangular matrix (upper or lower)

See Std.PBblas.Types

PBblas.gemm

IMPORTS

- PBblas
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- std.BLAS
- PBblas.internal.MatDims
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : gemm

Up :

<table>
<tr><td>DATASET(Layout_Cell)</td><td>gemm</td></tr>
<tr><td colspan="2">(BOOLEAN transposeA, BOOLEAN transposeB, value_t alpha,
DATASET(Layout_Cell) A_in, DATASET(Layout_Cell) B_in,
DATASET(Layout_Cell) C_in=emptyC, value_t beta=0.0)</td></tr>
</table>

Extended Parallel Block Matrix Multiplication Module Implements: Result = alpha * op(A)op(B) + beta * C. op is No Transpose or Transpose. Multiplies two matrixes A and B, with an optional pre-multiply transpose for each Optionally scales the product by the scalar ”alpha”. Then adds an optional C matrix to the product after scaling C by the scalar ”beta”. A, B, and C are specified as DATASET(Layout_Cell), as is the Resulting matrix. Layout_Cell describes a sparse matrix stored as a list of x, y, and value. This interface also provides a ”Myriad” capability allowing multiple similar operations to be performed on independent sets of matrixes in parallel. This is done by use of the work-item id (wi_id) in each cell of the matrixes. Cells with the same wi_id are considered part of the same matrix. In the myriad form, each input matrix A, B, and (optionally) C can contain many independent matrixes. The wi_ids are matched up such that each operation involves the A, B, and C with the same wi_id. A and B must therefore contain the same set of wi_ids, while C is optional for any wi_id. The same parameters: alpha, beta, transposeA, and transposeB are used for all work-items. The result will contain cells from all provided work-items. Result has same shape as C if provided. Note that matrixes are not explicitly dimensioned. The shape is determined by the highest value of x and y for each work-item.

Parameter transposeA ||| Boolean indicating whether matrix A should be transposed before multiplying

Parameter transposeB ||| Same as above but for matrix B

Parameter alpha ||| Scalar multiplier for alpha * A * B

Parameter A_in ||| 'A' matrix (multiplier) in Layout_Cell format

Parameter B_in ||| Same as above for the 'B' matrix (multiplicand)

Parameter C_in ||| Same as above for the 'C' matrix (addend). May be omitted.

Parameter beta ||| A scalar multiplier for beta * C, scales the C matrix before addition. May be omitted.

Return Result matrix in Layout_Cell format.

See PBblas/Types.Layout_Cell

PBblas.getrf

IMPORTS

- PBblas
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- std.BLAS
- PBblas.internal.MatDims
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : getrf

Up :

DATASET(Layout_Cell)	getrf
(DATASET(Layout_Cell) A)	

LU Factorization Splits a matrix into Lower and Upper triangular factors Produces composite LU matrix for the diagonal blocks. Iterates through the matrix a row of blocks and column of blocks at a time. Partition A into M block rows and N block columns. The A11 cell is a single block. A12 is a single row of blocks with N-1 columns. A21 is a single column of blocks with M-1 rows. A22 is a sub-matrix of M-1 x N-1 blocks. | A11 A12 | | L11 0 | | U11 U12 | | A21 A22 | == | L21 L22 | * | 0 U22 | | L11*U11 L11*U12 | == | L21*U11 L21*U12 + L22*U22 | Based upon PB-BLAS: A set of parallel block basic linear algebra subprograms by Choi and Dongarra This module supports the ”Myriad” style interface, allowing many independent problems to be worked on at once. The A matrix can contain multiple matrixes to be factored, indicated by different values for work-item id (wi_id). Note: The returned matrix includes both the upper and lower factors. This matrix can be used directly by trsm which will only use the part indicated by trsm's 'triangle' parameter (i.e. upper or lower). To extract the upper or lower triangle explicitly for other purposes, use the ExtractTri function. When passing the Lower matrix to the triangle solver (trsm), set the "Diagonal" parameter to "UnitTri". This is necessary because both triangular matrixes returned from this function are packed into a square matrix with only one diagonal. By convention, The Lower triangle is assumed to be a Unit Triangle (diagonal all ones), so the diagonal contained in the returned matrix is for the Upper factor and must be ignored (i.e. assumed to be all ones) when referencing the Lower triangle.

Parameter A ||| The input matrix in Types.Layout_Cell format

Return Resulting factored matrix in Layout_Cell format

See Types.Layout_Cell

See ExtractTri

PBblas.HadamardProduct

IMPORTS

- PBblas
- PBblas.internal
- PBblas.internal.MatDims
- PBblas.Types
- PBblas.internal.Types
- PBblas.internal.Converted
- std.BLAS
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : HadamardProduct

Up :

DATASET(Layout_Cell)	HadamardProduct
(DATASET(Layout_Cell) X, DATASET(Layout_Cell) Y)	

Element-wise multiplication of X * Y. Supports the ”myriad” style interface – X and Y may contain multiple separate matrixes. Each X will be multiplied by the Y with the same work-item id. Note: This performs element-wise multiplication. For dot-product matrix multiplication, use PBblas.gemm.

Parameter X ||| A matrix (or multiple matrices) in Layout_Cell form

Parameter Y ||| A matrix (or multiple matrices) in Layout_Cell form

Return A matrix (or multiple matrices) in Layout_Cell form

See PBblas/Types.Layout_Cell

PBblas.IElementFunc

IMPORTS

- PBblas

DESCRIPTIONS

FUNCTION : IElementFunc

Up :

value_t	IElementFunc
(value_t v, dimension_t r, dimension_t c)	

Function prototype for a function to apply to each element of the distributed matrix Base your function on this prototype:

Parameter v ||| Input value

Parameter r ||| Row number (1 based)

Parameter c ||| Column number (1 based)

Return Output value

See PBblas/Apply2Elements

PBblas.MatUtils

IMPORTS

- PBblas
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims

DESCRIPTIONS

MODULE : MatUtils

Up :

	MatUtils

Provides various utility attributes for manipulating cell-based matrixes

See Std/PBblas/Types.Layout_Cell

GetWorkItems | InsertCols | Transpose |

FUNCTION : GetWorkItems

Up : MatUtils \

DATASET(Layout_WI_ID)	GetWorkItems
(DATASET(Layout_Cell) cells)	

Get a list of work-item ids from a matrix containing one or more work items

Parameter cells ||| A matrix in Layout_Cell format

Return DATASET(Layout_WI_ID), one record per work-item

See PBblas/Types.Layout_Cell

See PBblas/Types.Layout_WI_ID

FUNCTION : InsertCols

Up : MatUtils \

DATASET(Layout_Cell)	InsertCols
(DATASET(Layout_Cell) M, UNSIGNED cols_to_insert=1, value_t insert_val=1)	

Insert one or more columns of a fixed value into a matrix. Columns are inserted before the first original column. This attribute supports the myriad interface. Multiple independent matrixes can be represented by M.

Parameter M ||| the input matrix

Parameter cols_to_insert ||| the number of columns to insert, default 1

Parameter insert_val ||| the value for each cell of the new column(s), default 0

Return matrix in Layout_Cell format with additional column(s)

FUNCTION : Transpose

Up : MatUtils \

DATASET(Layout_Cell)	Transpose
(DATASET(Layout_Cell) M)	

Transpose a matrix This attribute supports the myriad interface. Multiple independent matrixes can be represented by M.

Parameter M ||| A matrix represented as DATASET(Layout_Cell)

Return Transposed matrix in Layout_Cell format

See PBblas/Types.Layout_Cell

PBblas.potrf

IMPORTS

- PBblas
- PBblas.Types
- std.BLAS
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims
- PBblas.internal.Converted
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : potrf

Up :

DATASET(Layout_Cell)	potrf
(Triangle tri, DATASET(Layout_Cell) A_in)	

Implements Cholesky factorization of A = U**T * U if Triangular.Upper requested or A = L * L**T if Triangualr.Lower is requested. The matrix A must be symmetric positive definite.

```
  | A11   A12 |      |  L11   0    |    | L11**T     L21**T |
  | A21   A22 |  ==  |  L21   L22  | *  |  0           L22  |

                     | L11*L11**T          L11*L21**T       |
                 ==  | L21*L11**T  L21*L21**T + L22*L22**T |
```

So, use Cholesky on the first block to get L11. L21 = A21*L11**T**-1 which can be found by dtrsm on each column block A22' is A22 - L21*L21**T

Based upon PB-BLAS: A set of parallel block basic linear algebra subprograms by Choi and Dongarra

This module supports the "Myriad" style interface, allowing many independent problems to be worked on at once. The A matrix can contain multiple matrixes to be factored, indicated by different values for work-item id (wi_id).

Parameter tri ||| Types.Triangle enumeration indicating whether we are looking for the Upper or the Lower factor

Parameter A_in ||| The matrix or matrixes to be factored in Types.Layout_Cell format

Return Triangular matrix in Layout_Cell format

See Std.PBblas.Types.Layout_Cell

See Std.PBblas.Types.Triangle

PBblas.scal

IMPORTS

- PBblas
- PBblas.Types

DESCRIPTIONS

FUNCTION : scal

Up :

DATASET(Layout_Cell)	scal
(value_t alpha, DATASET(Layout_Cell) X)	

Scale a matrix by a constant Result is alpha * X This supports a "myriad" style interface in that X may be a set of independent matrices separated by different work-item ids.

Parameter alpha ||| A scalar multiplier

Parameter X ||| The matrix(es) to be scaled in Layout_Cell format

Return Matrix in Layout_Cell form, of the same shape as X

See PBblas/Types.Layout_Cell

PBblas.tran

IMPORTS

- PBblas
- PBblas.Types
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims
- PBblas.internal.Converted
- std.BLAS
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : tran

Up :

DATASET(Layout_Cell)	tran
(value_t alpha, DATASET(Layout_Cell) A, value_t beta=0, DATASET(Layout_Cell) C=empty_c)	

Transpose a matrix and sum into base matrix result <== alpha * A**t + beta * C, A is n by m, C is m by n A**T (A Transpose) and C must have same shape

Parameter alpha ||| Scalar multiplier for the A**T matrix

Parameter A ||| A matrix in DATASET(Layout_Cell) form

Parameter beta ||| Scalar multiplier for the C matrix

Parameter C ||| C matrix in DATASET(Layout_Call) form

Return Matrix in DATASET(Layout_Cell) form alpha * A**T + beta * C

See PBblas/Types.layout_cell

PBblas.trsm

IMPORTS

- PBblas
- PBblas.Types
- std.BLAS
- PBblas.internal
- PBblas.internal.Types
- PBblas.internal.MatDims
- PBblas.internal.Converted
- std.system.Thorlib

DESCRIPTIONS

FUNCTION : trsm

Up :

DATASET(Layout_Cell)	trsm
(Side s, Triangle tri, BOOLEAN transposeA, Diagonal diag, value_t alpha, DATASET(Layout_Cell) A_in, DATASET(Layout_Cell) B_in)	

Partitioned block parallel triangular matrix solver. Solves for X using: AX = B or XA = B A is is a square triangular matrix, X and B have the same dimensions. A may be an upper triangular matrix (UX = B or XU = B), or a lower triangular matrix (LX = B or XL = B). Allows optional transposing and scaling of A. Partially based upon an approach discussed by MJ DAYDE, IS DUFF, AP CERFACS. A Parallel Block implementation of Level-3 BLAS for MIMD Vector Processors ACM Tran. Mathematical Software, Vol 20, No 2, June 1994 pp 178-193 and other papers about PB-BLAS by Choi and Dongarra This module supports the ”Myriad” style interface, allowing many independent problems to be worked on at once. Corresponding A and B matrixes are related by a common work-item identifier (wi_id) within each cell of the matrix. The returned X matrix will contain cells for the same set of work-items as specified for the A and B matrices.

Parameter s ||| Types.Side enumeration indicating whether we are solving AX = B or XA = B

Parameter tri ||| Types.Triangle enumeration indicating whether we are solving an Upper or Lower triangle.

Parameter transposeA ||| Boolean indicating whether or not to transpose the A matrix before solving

Parameter diag ||| Types.Diagonal enumeration indicating whether A is a unit matrix or not. This is primarily used after factoring matrixes using getrf (LU factorization). That module produces a factored matrix stored within the same space as the original matrix. Since the diagonal is used by both factors, by convention, the Lower triangle has a unit matrix (diagonal all 1's) while the Upper triangle uses the diagonal cells. Setting this to UnitTri, causes the contents of the diagonal to be ignored, and assumed to be 1. NotUnitTri should be used for most other cases.

Parameter alpha ||| Multiplier to scale A

Parameter A_in ||| The A matrix in Layout_Cell format

Parameter B_in ||| The B matrix in Layout_Cell format

Return X solution matrix in Layout_Cell format

See Types.Layout_Cell

See Types.Triangle

See Types.Side

PBblas.Types

IMPORTS

- ML_Core
- ML_Core.Types

DESCRIPTIONS

MODULE : Types

Up :

	Types

Types for the Parallel Block Basic Linear Algebra Sub-programs support WARNING: attributes marked with WARNING can not be changed without making corresponding changes to the C++ attributes.

dimension_t | partition_t | work_item_t | value_t | m_label_t | Triangle | Diagonal | Side | t_mu_no | Layout_Cell | Layout_Norm |

ATTRIBUTE : dimension_t

Up : Types \

	dimension_t

Type for matrix dimensions. Uses UNSIGNED four as matrixes are not designed to support more than 4 B rows or columns.

ATTRIBUTE : partition_t

Up : Types \

	partition_t

Type for partition id – only supports up to 64K partitions

ATTRIBUTE : work_item_t

Up : Types \

	work_item_t

Type for work-item id – only supports up to 64K work items

ATTRIBUTE : value_t

Up : Types \

	value_t

Type for matrix cell values WARNING: type used in C++ attribute

---

ATTRIBUTE : m_label_t

Up : Types \

	m_label_t

Type for matrix label. Used for Matrix dimensions (see Layout_Dims) and for partitions (see Layout_Part)

ATTRIBUTE : Triangle

Up : Types \

	Triangle

Enumeration for Triangle type WARNING: type used in C++ attribute

---

ATTRIBUTE : Diagonal

Up : Types \

	Diagonal

Enumeration for Diagonal type WARNING: type used in C++ attribute

---

ATTRIBUTE : Side

Up : Types \

	Side

Enumeration for Side type WARNING: type used in C++ attribute

---

ATTRIBUTE : t_mu_no

Up : Types \

```
t_mu_no
```

Type for matrix universe number Allow up to 64k matrices in one universe

---

RECORD : Layout_Cell

Up : Types \

```
Layout_Cell
```

Layout for Matrix Cell Main representation of Matrix cell at interface to all PBBlas functions. Matrixes are represented as DATASET(Layout_Cell), where each cell describes the row and column position of the cell as well as its value. Only the non-zero cells need to be contained in the dataset in order to describe the matrix since all unspecified cells are considered to have a value of zero. The cell also contains a work-item number that allows multiple separate matrixes to be carried in the same dataset. This supports the ”myriad” style interface that allows the same operations to be performed on many different sets of data at once. Note that these matrixes do not have an explicit size. They are sized implicitly, based on the maximum row and column presented in the data. A matrix can be converted to an explicit dense form (see matrix_t) by using the utility module MakeR8Set. This module should only be used for known small matrixes (< 1M cells) or for partitions of a larger matrix. The Converted module provides utility functions to convert to and from a set of partitions (See Layout_parts).

Field wi_id ||| Work Item Number – An identifier from 1 to 64K-1 that separates and identifies individual matrixes

Field x ||| 1-based row position within the matrix

Field y ||| 1-based column position within the matrix

Field v ||| Real value for the cell

See matrix_t

See Std/PBblas/MakeR8Set.ecl

See Std/PBblas/Converted.ecl WARNING: Used as C++ attribute. Do not change without corresponding changes to MakeR8Set.

RECORD : Layout_Norm

Up : Types \

	Layout_Norm

Layout for Norm results.

Field wi_id ||| Work Item Number – An identifier from 1 to 64K-1 that separates and identifies individual matrixes

Field v ||| Real value for the norm

PBblas.Vector2Diag

IMPORTS

- PBblas
- PBblas.internal
- PBblas.internal.MatDims
- PBblas.Types
- PBblas.internal.Types
- PBblas.Constants

DESCRIPTIONS

FUNCTION : Vector2Diag

Up :

DATASET(Layout_Cell)	Vector2Diag
(DATASET(Layout_Cell) X)	

Convert a vector into a diagonal matrix. The typical notation is D = diag(V). The input X must be a 1 x N column vector or an N x 1 row vector. The resulting matrix, in either case will be N x N, with zero everywhere except the diagonal.

Parameter X ||| A row or column vector (i.e. N x 1 or 1 x N) in Layout_Cell format

Return An N x N matrix in Layout_Cell format

See PBblas/Types.Layout_cell