ECL DOCUMENTATION

Go Up

Name	SupportVectorMachines
Version	1.1
Description	Support Vector Machines 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

OVERVIEW

SupportVectorMachines

Support Vector Machine implementations.

Prerequisites: this bundle is a wrapper for the LibSVM library, and so requires LibSVM be installed on all nodes of the cluster. Note: some distributions may have a development version, and that the development version is required if there is more than one version.

Table of Contents

SVC

Go Up

IMPORTS

Types | libsvm.Types | PBblas | ML_Core | ML_Core.Types | ML_Core.Interfaces |

DESCRIPTIONS

SVC

SVC

```
( Types.SVM_Type svmType = LibSVM_Types.LibSVM_Type.C_SVC, Types.Kernel_Type
kernelType = LibSVM_Types.LibSVM_Kernel.RBF, REAL8 gamma = 0.05, REAL8 C = 1,
INTEGER4 degree = 3, REAL8 coef0 = 0.0, REAL8 eps = 0.001, REAL8 nu = 0.5, REAL8 p
= 0.1, BOOLEAN shrinking = true, BOOLEAN prob_est = true, BOOLEAN scale = true,
INTEGER4 nr_weight = 0, DATASET(Types.I4Entry) lbl = DATASET([], Types.I4Entry),
DATASET(Types.R8Entry) weight = DATASET([], Types.R8Entry))
```

Support vector machine classification.

Utilizes the open-source libSVM under the hood.

This module is appropriate for small to medium sized Machine Learning problems or multitudes of small-to-medium problems using the Myriad interface.

This is due to both scaling limitations endemic to SVM, as well as the fact that libSVM runs independently on each node, and cannot, therefore scale to very large single problems.

Other techniques should be employed for Machine Learning with more than 10,000 data points.

This module also provides a mechanism for doing a grid search for regularization parameters using the full

resources of the HPCC cluster rather than searching sequentially (see GridSearch.ecl).

PARAMETER **svmType** ||| UNSIGNED2 — The SVC type, which may be one of 0 (C_SVC, default), 1 (NU_SVC), or 2 (ONE_CLASS).

PARAMETER **kernelType** ||| UNSIGNED2 — The kernel used in training and predicting, which may be one of 0 (LINEAR), 1 (POLY), 2 (RBF, default), 3 (SIGMOID), or 4 (PRECOMPUTED).

PARAMETER **gamma** ||| REAL8 — Parameter needed for all kernels except LINEAR (default: 0.05).

PARAMETER **C** ||| REAL8 — Cost of constraint violation (default: 1).

PARAMETER **degree** ||| INTEGER4 — Parameter needed for kernel of type POLY (default: 3).

PARAMETER **coef0** ||| REAL8 — Parameter needed for kernels of type POLY and SIGMOID (default: 0).

PARAMETER **eps** ||| REAL8 — Tolerance of termination criterion (default: 0.001).

PARAMETER **nu** ||| REAL8 — Parameter needed for NU_SVC and ONE_CLASS (default: 0.5).

PARAMETER **p** ||| REAL8 — Epsilon in the insensitive-loss function (default: 0.1).

PARAMETER **shrinking** ||| BOOLEAN — Flag indicating the use of shrinking-heuristics (default: true).

PARAMETER **prob_est** ||| BOOLEAN — Whether to train for probability estimates (default true).

PARAMETER **scale** ||| BOOLEAN — Whether to standardize the data (subtract mean, divide by sd) before fitting.

PARAMETER **nr_weight** ||| INTEGER4 — The number of elements in the 'lbl' parameter (default: 0).

PARAMETER **lbl** ||| TABLE (I4Entry) — Labels to indicate classes, used with the 'weight' parameter (default: []).

PARAMETER **weight** ||| TABLE (R8Entry) — Class weights, assigned to classes using the 'lbl' parameter (default: []).

PARENT **ML_Core.Interfaces.IClassify** </home/lily/source/ML_Core/Interfaces/IClassify.ecl>

Children

1. GetModel : Calculate a model to fit the observation data to the observed classes
2. Classify : Classify the values for new observations using models trained by the GetModel function
3. Report : Report the confusion matrix for the classifier and training data
4. Tune : Perform a regularization tuning in order to align the granularity of the algorithm with the complexity of the data
5. GetTunedModel : Choose the best set of regularization parameters and use it to train the models

6. CrossValidate : Perform n-fold cross-validation of a given model for each work-item
7. ModelSummary : Generate human-readable model summary of trained SVM model(s)

GETMODEL

SVC /

DATASET(Layout_Model)	**GetModel**
(DATASET(NumericField) observations, DATASET(DiscreteField) classifications)	

Calculate a model to fit the observation data to the observed classes. For a single given set of model parameters, models can be fit to a number of datasets by concatenating multiple datasets into single 'observations' and 'classifications' datasets, with separate datasets being identified by a work-item number ('wi'), in the NumericField and DiscreteField datasets.

PARAMETER **observations** ||| TABLE (NumericField) — The observed explanatory values in NumericField format.

PARAMETER **classifications** ||| TABLE (DiscreteField) — The observed classification used to build the model in DiscreteField format.

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED8 id , UNSIGNED4 number , REAL8 value })** — The encoded models in Layout_Model format.

SEE ML_Core.Types.NumericField

SEE ML_Core.Types.DiscreteField

SEE ML_Core.Types.Layout_Model

OVERRIDE

CLASSIFY

SVC /

DATASET(ML_Types.Classify_Result)	**Classify**
(DATASET(Layout_Model) model, DATASET(NumericField) new_observations)	

Classify the values for new observations using models trained by the GetModel function.

PARAMETER **model** ||| TABLE (Layout_Model) — The models, which should be produced by a corresponding GetModel function.

PARAMETER **new_observations** ||| TABLE (NumericField) — Observations to be classified in NumericField format.

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED8 id , UNSIGNED4 number , INTEGER4 value , REAL8 conf })** — Classifications with a probability value in Classify_Results format.

SEE ML_Core.Types.NumericField

SEE ML_Core.Types.Classify_Results

OVERRIDE

REPORT

SVC /

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

Report the confusion matrix for the classifier and training data.

PARAMETER **model** ||| TABLE (Layout_Model) — The models, which should be produced by a corresponding GetModel function.

PARAMETER **observations** ||| TABLE (NumericField) — The explanatory values in NumericField format.

PARAMETER **classifications** ||| TABLE (DiscreteField) — The classifications associated with the observations in DiscreteField format.

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED4 classifier , INTEGER4 actual_class , INTEGER4 predict_class , UNSIGNED4 occurs , BOOLEAN correct , REAL8 pctActual , REAL8 pctPred })** — The confusion matrix showing correct and incorrect results in Confusion_Detail format.

SEE ML_Core.Types.NumericField

SEE ML_Core.Tyeps.DiscreteField

SEE ML_Core.Types.Confusion_Detail

OVERRIDE

TUNE

SVC /

DATASET(Types.GridSearch_Result)	**Tune**

```
( INTEGER4 folds = 10, REAL8 start_log2C = -5,
REAL8 stop_log2C = 15, REAL8 maxIncr_log2C = 2,
REAL8 start_log2gamma = -15, REAL8 stop_log2gamma =
3, REAL8 maxIncr_log2gamma = 2,
DATASET(NumericField) observations,
DATASET(DiscreteField) classifications)
```

Perform a regularization tuning in order to align the granularity of the algorithm with the complexity of the data. This is to avoid under or over fitting of the data.

Finds a reasonable setting for the regularization parameters gamma and C by performing a grid search over them and testing each using cross-validation. The parameters that provide the lowest out-of-sample error (i.e. when tested on data not in the training set) are the ones chosen.

Returns a set of training parameter combinations and their results that can then be passed to GetTunedModel below to acquire a model that has been properly regularized.

The grid resolution is increased automatically to utilize any otherwise idle nodes.

For a single given set of model parameters, models can be tuned to a number of datasets by concatenating multiple datasets into single 'observations' and 'classifications' datasets, with separate datasets being identified by a work ID column, 'wi'.

PARAMETER **folds** ||| INTEGER4 — The number of cross-validation folds for evaluating each candidate model.

PARAMETER **start_log2C** ||| REAL8 — The lower bound for log2(C): C >= 2^(start_log2C).

PARAMETER **stop_log2C** ||| REAL8 — The upper bound for log2(C): C <= 2^(start_log2C).

PARAMETER **maxIncr_log2C** ||| REAL8 — Taximum allowable exponential increment for C.

PARAMETER **start_log2gamma** ||| REAL8 — The lower bound for log2(gamma): gamma >= 2^(start_log2gamma).

PARAMETER **stop_log2gamma** ||| REAL8 — The upper bound for log2(gamma): gamma <= 2^(start_log2gamma).

PARAMETER **maxIncr_log2gamma** ||| REAL8 — Taximum allowable exponential increment for gamma.

PARAMETER **observations** ||| TABLE (NumericField) — The observed explanatory values in NumericField format.

PARAMETER **classifications** ||| TABLE (DiscreteField) — The observed classification used to build the model in DiscreteField format.

RETURN **TABLE ({ UNSIGNED2 wi , INTEGER4 id , REAL8 correct , REAL8 mse , REAL8 r_sq , UNSIGNED2 svmType , UNSIGNED2 kernelType , INTEGER4 degree , REAL8 coef0 , REAL8 eps , REAL8 nu , REAL8 p , INTEGER4 nr_weight , BOOLEAN shrinking , BOOLEAN prob_est , BOOLEAN scale , TABLE (I4Entry) lbl , TABLE (R8Entry) weight , REAL8 gamma , REAL8 C })** — Dataset with sets of model parameters and corresponding cross-validated scores in GridSearch_Result format.

SEE GetTunedModel

SEE Types.GridSearch_Result

GETTUNEDMODEL

SVC /

DATASET(Layout_Model)	**GetTunedModel**

```
( DATASET(Types.GridSearch_Result) tuneResult,
DATASET(NumericField) observations, DATASET(DiscreteField)
classifications)
```

Choose the best set of regularization parameters and use it to train the models. Using the output of Tune(), find the best set of modeling parameters for each work id, and train the corresponding models. The the most regularized (i.e. coarsest) set of parameters that achieved near-maximum performance is used to create the models.

PARAMETER **tuneResult** ||| TABLE (GridSearch_Result) — The results of a grid search over C and gamma, produced by Tune().

PARAMETER **observations** ||| TABLE (NumericField) — The observed explanatory values in NumericField format.

PARAMETER **classifications** ||| TABLE (DiscreteField) — The observed classification used to build the model in DiscreteField format.

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED8 id , UNSIGNED4 number , REAL8 value })** — The encoded models in Layout_Model format.

SEE Tune

SEE ML_Core.Types.NumericField

SEE ML_Core.Types.DiscreteField

SEE ML_Core.Types.Layout_Model

CROSSVALIDATE

SVC /

DATASET(Types.CrossValidate_Result)	**CrossValidate**
(INTEGER4 folds = 10, DATASET(NumericField) observations, DATASET(DiscreteField) classifications)	

Perform n-fold cross-validation of a given model for each work-item.

For a single given set of model parameters, models can be cross-validated against a number of datasets by concatenating multiple datasets into single 'observations' and 'classifications' datasets, with separate datasets being identified by a work ID column, 'wi'.

PARAMETER **folds** ||| INTEGER4 — The number of cross-validation folds.

PARAMETER **observations** ||| TABLE (NumericField) — The observed explanatory values in NumericField format.

PARAMETER **classifications** ||| TABLE (DiscreteField) — The observed classification used to build in DiscreteField format.

RETURN **TABLE ({ UNSIGNED2 wi , INTEGER4 id , REAL8 correct , REAL8 mse , REAL8 r_sq })** — Dataset of cross-validated scores in CrossValidate_Result format.

SEE ML_Core.NumericField

SEE ML_Core.DiscreteField

SEE Types.CrossValidate_Result

MODELSUMMARY

SVC /

DATASET({UNSIGNED4 r, STRING60 Txt})	**ModelSummary**
(DATASET(Layout_Model) model)	

Generate human-readable model summary of trained SVM model(s).

Multiple models can be simultaneously summarized by concatenating a number of models into a single 'model' object, with separate models being identified by a work ID column, 'wi'.

PARAMETER **model** ||| TABLE (Layout_Model) — The models, which should be produced by a corresponding GetModel function.

RETURN **TABLE ({ UNSIGNED4 r , STRING60 txt })** — Single-column dataset with textual description of models.

SVR

Go Up

IMPORTS

Types | libsvm.Types | PBblas | ML_Core | ML_Core.Types | ML_Core.Interfaces |

DESCRIPTIONS

SVR

SVR

```
( DATASET(NumericField) X = DATASET([], NumericField), DATASET(NumericField) Y =
DATASET([], NumericField), Types.SVM_Type svmType = LibSVM_Types.LibSVM_Type.C_SVC,
Types.Kernel_Type kernelType = LibSVM_Types.LibSVM_Kernel.RBF, REAL8 gamma = 0.05,
REAL8 C = 1, INTEGER4 degree = 3, REAL8 coef0 = 0.0, REAL8 eps = 0.001, REAL8 nu =
0.5, REAL8 p = 0.1, BOOLEAN shrinking = true, BOOLEAN prob_est = true, BOOLEAN
scale = true, INTEGER4 nr_weight = 0, DATASET(Types.I4Entry) lbl = DATASET([],
Types.I4Entry), DATASET(Types.R8Entry) weight = DATASET([], Types.R8Entry))
```

Support Vector Machine Regression.

Utilizes the open-source libSVM under the hood.

This module is appropriate for small to medium sized Machine Learning problems or multitudes of small-to-medium problems using the Myriad interface.

This is due to both scaling limitations endemic to SVM, as well as the fact that libSVM runs independently on each node, and cannot, therefore scale to very large single problems.

Other techniques should be employed for Machine Learning with more than 10,000 data points.

This module also provides a mechanism for doing a grid search for regularization parameters using the full resources of the HPCC cluster rather than searching sequentially (see GridSearch.ecl).

PARAMETER **X** ||| TABLE (NumericField) — The observed explanatory values in NumericField format.

PARAMETER **Y** ||| TABLE (NumericField) — The observed values the model aims to fit in NumericField format.

PARAMETER **svmType** ||| UNSIGNED2 — The SVR type, which may be one of 3 (EPSILON_SVR, default), or 4 (NU_SVR).

PARAMETER **kernelType** ||| UNSIGNED2 — The kernel used in training and predicting, which may be one of 0 (LINEAR), 1 (POLY), 2 (RBF, default), 3 (SIGMOID), or 4 (PRECOMPUTED).

PARAMETER **gamma** ||| REAL8 — regularization parameter needed for all kernels except LINEAR (default: 0.05).

PARAMETER **C** ||| REAL8 — Cost of constraint violation regularization parameter(default: 1).

PARAMETER **degree** ||| INTEGER4 — Parameter needed for kernel of type POLY (default: 3).

PARAMETER **coef0** ||| REAL8 — Parameter needed for kernels of type POLY and SIGMOID (default: 0).

PARAMETER **eps** ||| REAL8 — Tolerance of termination criterion (default: 0.001).

PARAMETER **nu** ||| REAL8 — Parameter needed for NU_SVC and ONE_CLASS (default: 0.5).

PARAMETER **p** ||| REAL8 — Epsilon in the insensitive-loss function (default: 0.1).

PARAMETER **shrinking** ||| BOOLEAN — Flag indicating the use of shrinking-heuristics (default: true).

PARAMETER **prob_est** ||| BOOLEAN — Whether to train for probability estimates (default true).

PARAMETER **scale** ||| BOOLEAN — Whether to standardize the data (subtract mean, divide by sd) before fitting.

PARAMETER **nr_weight** ||| INTEGER4 — No Doc

PARAMETER **lbl** ||| TABLE (I4Entry) — No Doc

PARAMETER **weight** ||| TABLE (R8Entry) — No Doc

SEE ML_Core.Types.NumericField

PARENT **ML_Core.Interfaces.IRegression** </home/lily/source/ML_Core/Interfaces/IRegression.ecl>

Children

1. GetModel : Train and return a model that fits the observation data to the observed values

2. Predict : Predict values for the new observations using models trained by the GetModel function
3. Tune : Perform a regularization tuning in order to align the granularity of the algorithm with the complexity of the data
4. GetTunedModel : Choose the best set of regularization parameters and use it to train the models
5. CrossValidate : Perform n-fold cross-validation of a given model for each work ID
6. ModelSummary : Generate human-readable model summary of trained SVM model(s)

GETMODEL

SVR /

DATASET(Layout_Model)	**GetModel**

Train and return a model that fits the observation data to the observed values. For a single given set of model parameters, models can be fit to a number of datasets by concatenating multiple datasets into single 'X' and 'Y' datasets, with separate datasets being identified by a work-item column, 'wi'.

RETURN — The encoded models in Layout_Model format.

SEE ML_Core.Types.Layout_Model

OVERRIDE

PREDICT

SVR /

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

Predict values for the new observations using models trained by the GetModel function.

PARAMETER **model** ||| TABLE (Layout_Model) — The models, which should be produced by a corresponding GetModel function.

PARAMETER **newX** ||| TABLE (NumericField) — Observations to be classified in NumericField format.

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED8 id , UNSIGNED4 number , REAL8 value })** — Predictions in NumericField format.

SEE ML_Core.Types.NumericField

OVERRIDE

TUNE

SVR /

DATASET(Types.GridSearch_Result)	**Tune**

```
( INTEGER4 folds = 10, REAL8 start_log2C = -5,
REAL8 stop_log2C = 15, REAL8 maxIncr_log2C = 2,
REAL8 start_log2gamma = -15, REAL8 stop_log2gamma =
3, REAL8 maxIncr_log2gamma = 2)
```

Perform a regularization tuning in order to align the granularity of the algorithm with the complexity of the data. This is to avoid under or over fitting of the data.

Finds a reasonable setting for the regularization parameters gamma and C by performing a grid search over them and testing each using cross-validation. The parameters that provide the lowest out-of-sample error (i.e. when tested on data not in the training set) are the ones chosen.

Returns a set of training parameter combinations and their results that can then be passed to GetTunedModel below to acquire a model that has been properly regularized.

The grid resolution is increased automatically to utilize any otherwise idle nodes.

For a single given set of model parameters, models can be tuned to a number of datasets by concatenating multiple datasets into single 'observations' and 'classifications' datasets, with separate datasets being identified by a work ID column, 'wi'.

PARAMETER **folds** ||| INTEGER4 — The number of cross-validation folds for evaluating each candidate model.

PARAMETER **start_log2C** ||| REAL8 — The lower bound for log2(C): C >= 2^(start_log2C).

PARAMETER **stop_log2C** ||| REAL8 — The upper bound for log2(C): C <= 2^(start_log2C).

PARAMETER **maxIncr_log2C** ||| REAL8 — Taximum allowable exponential increment for C.

PARAMETER **start_log2gamma** ||| REAL8 — The lower bound for log2(gamma): gamma >= 2^(start_log2gamma).

PARAMETER **stop_log2gamma** ||| REAL8 — The upper bound for log2(gamma): gamma <= 2^(start_log2gamma).

PARAMETER **maxIncr_log2gamma** ||| REAL8 — Taximum allowable exponential increment for gamma.

RETURN **TABLE ({ UNSIGNED2 wi , INTEGER4 id , REAL8 correct , REAL8 mse , REAL8 r_sq , UNSIGNED2 svmType , UNSIGNED2 kernelType , INTEGER4 degree , REAL8 coef0 , REAL8 eps , REAL8 nu , REAL8 p , INTEGER4 nr_weight , BOOLEAN shrinking , BOOLEAN prob_est , BOOLEAN scale , TABLE (I4Entry) lbl , TABLE (R8Entry) weight , REAL8 gamma , REAL8 C })** — Dataset with sets of model parameters and corresponding cross-validated scores in GridSearch_Result format.

SEE GetTunedModel

SEE Types.GridSearch_Result

GETTUNEDMODEL

SVR /

DATASET(Layout_Model)	**GetTunedModel**
(DATASET(Types.GridSearch_Result) tuneResult)	

Choose the best set of regularization parameters and use it to train the models. Using the output of Tune(), find the best set of modeling parameters for each work id, and train the corresponding models. The the most regularized (i.e. coarsest) set of parameters that achieved near-maximum performance is used to create the models.

PARAMETER **tuneResult** ||| TABLE (GridSearch_Result) — The results of a grid search over C and gamma, produced by Tune().

RETURN **TABLE ({ UNSIGNED2 wi , UNSIGNED8 id , UNSIGNED4 number , REAL8 value })** — The encoded models.

CROSSVALIDATE

SVR /

DATASET(Types.CrossValidate_Result)	**CrossValidate**
(INTEGER4 folds = 10)	

Perform n-fold cross-validation of a given model for each work ID. For a single given set of model parameters, models can be cross-validated against a number of datasets by concatenating multiple datasets into single 'X' and 'Y' datasets, with separate datasets being identified by a work ID column, 'wi'.

PARAMETER **folds** ||| INTEGER4 — The number of cross-validation folds.

RETURN **TABLE ({ UNSIGNED2 wi , INTEGER4 id , REAL8 correct , REAL8 mse , REAL8 r_sq })** — Dataset of cross-validated scores i CrossValidate_Result format.

SEE Types.CrossValidate_Result

MODELSUMMARY

SVR /

DATASET({UNSIGNED4 r, STRING60 Txt})	**ModelSummary**
(DATASET(Layout_Model) model)	

Generate human-readable model summary of trained SVM model(s).

Multiple models can be simultaneously summarized by concatenating a number of models into a single 'model' object, with separate models being identified by a work ID column, 'wi'.

PARAMETER **model** ||| TABLE (Layout_Model) — The models, which should be produced by a corresponding GetModel function.

RETURN **TABLE ({ UNSIGNED4 r , STRING60 txt })** — Single-column dataset with textual description of models.

Types

Go Up

IMPORTS

libsvm.Types |

DESCRIPTIONS

TYPES

Types

SupportVectorMachines type definitions.

Children

1. CrossValidate_Result : Record to hold the results of call to CrossValidate
2. GridSearch_Result : Record for the results of call to GridSearch Contains both CrossValidate_Result and Training_Parameters

---

CROSSVALIDATE_RESULT

Types /

CrossValidate_Result

Record to hold the results of call to CrossValidate

FIELD **wi** ||| UNSIGNED2 — The work-item number.

FIELD **id** ||| INTEGER4 — The id of the cross-validation set (i.e. fold).

FIELD **correct** ||| REAL8 — The number of correct values.

FIELD **mse** ||| REAL8 — The mean squared error of the regression

FIELD **r_sq** ||| REAL8 — The R-squared value indicating the strength of the regression.

GRIDSEARCH_RESULT

Types /

GridSearch_Result

Record for the results of call to GridSearch Contains both CrossValidate_Result and Training_Parameters.

FIELD **wi** ||| UNSIGNED2 — The work-item number.

FIELD **id** ||| INTEGER4 — The id of the cross-validation set (i.e. fold).

FIELD **correct** ||| REAL8 — The number of correct values.

FIELD **mse** ||| REAL8 — The mean squared error of the regression

FIELD **r_sq** ||| REAL8 — The R-squared value indicating the strength of the regression.

FIELD **gamma** ||| REAL8 — The gamma regularization parameter value.

FIELD **C** ||| REAL8 — The C regularization parameter value.

FIELD **svmtype** ||| UNSIGNED2 — No Doc

FIELD **kerneltype** ||| UNSIGNED2 — No Doc

FIELD **degree** ||| INTEGER4 — No Doc

FIELD **coef0** ||| REAL8 — No Doc

FIELD **eps** ||| REAL8 — No Doc

FIELD **nu** ||| REAL8 — No Doc

FIELD **p** ||| REAL8 — No Doc

FIELD **nr_weight** ||| INTEGER4 — No Doc

FIELD **shrinking** ||| BOOLEAN — No Doc

FIELD **prob_est** ||| BOOLEAN — No Doc

FIELD **scale** ||| BOOLEAN — No Doc

FIELD **lbl** ||| TABLE (I4Entry) — No Doc

FIELD **weight** ||| TABLE (R8Entry) — No Doc
