

# Package ‘VectorForgeML’

February 23, 2026

**Type** Package

**Title** High-Performance Machine Learning Framework with C++  
Acceleration

**Version** 0.1.0

**Description** Machine learning utilities for fast vectorized model training.  
Methods are based on standard statistical learning references such as  
Hastie et al. (2009) <[doi:10.1007/978-0-387-84858-7](https://doi.org/10.1007/978-0-387-84858-7)>.

**License** Apache License (>= 2)

**Encoding** UTF-8

**Depends** R (>= 3.5.0)

**Imports** methods, Rcpp

**LinkingTo** Rcpp

**SystemRequirements** OpenMP (optional)

**URL** <https://vectorforgeml.work.gd>

**BugReports** <https://github.com/mohd-musheer/VectorForgeML/issues>

**NeedsCompilation** yes

**RoxygenNote** 7.3.3

**Author** Musheer Mohd [aut, cre]

**Maintainer** Musheer Mohd <[musheerayan@gmail.com](mailto:musheerayan@gmail.com)>

**Archs** x64

## Contents

VectorForgeML-package . . . . .	2
accuracy_score . . . . .	3
ColumnTransformer-class . . . . .	3
confusion_matrix . . . . .	4
confusion_stats . . . . .	5
DecisionTree-class . . . . .	5
drop_constant_columns . . . . .	6
f1_score . . . . .	7
find_best_k . . . . .	7
fit_linear_model . . . . .	8
KMeans-class . . . . .	9

KNN-class . . . . .	9
LabelEncoder-class . . . . .	10
LinearRegression-class . . . . .	10
LogisticRegression-class . . . . .	11
macro_f1 . . . . .	12
macro_precision . . . . .	12
macro_recall . . . . .	13
MinMaxScaler-class . . . . .	13
mse . . . . .	14
OneHotEncoder-class . . . . .	14
PCA-class . . . . .	15
Pipeline-class . . . . .	15
plot_confusion_matrix . . . . .	16
precision_score . . . . .	17
predict_linear_model . . . . .	17
r2_score . . . . .	18
RandomForest-class . . . . .	19
recall_score . . . . .	19
RidgeRegression-class . . . . .	20
rmse . . . . .	21
SoftmaxRegression-class . . . . .	21
StandardScaler-class . . . . .	22
train_test_split . . . . .	22
<b>Index</b>	<b>24</b>

---

VectorForgeML-package    *VectorForgeML: High-Performance ML Framework*

---

## Description

Fast machine learning models implemented in C++.

## Author(s)

**Maintainer:** Musheer Mohd <musheerayan@gmail.com>

## See Also

Useful links:

- <https://vectorforgeml.work.gd>
- Report bugs at <https://github.com/mohd-musheer/VectorForgeML/issues>

---

accuracy_score	<i>Accuracy Score</i>
----------------	-----------------------

---

**Description**

Computes classification accuracy.

**Usage**

```
accuracy_score(y_true, y_pred)
```

**Arguments**

y_true	true labels
y_pred	predicted labels

**Details**

Provides functionality for accuracy\_score operations.

**Value**

numeric accuracy

**See Also**

[VectorForgeML-package](#)

**Examples**

```
y_true <- c(1,0,1,1)
y_pred <- c(1,0,0,1)
accuracy_score(y_true, y_pred)
```

---

ColumnTransformer-class
<i>Column Transformer</i>

---

**Description**

Applies transformations to specific columns.

**Details**

Provides functionality for ColumnTransformer operations.

**Value**

ColumnTransformer object

**See Also**[VectorForgeML-package](#)**Examples**

```
model <- ColumnTransformer$new(num_cols="A", cat_cols="B")
```

---

confusion\_matrix*Confusion Matrix*

---

**Description**

Computes confusion matrix.

**Usage**

```
confusion_matrix(y_true, y_pred)
```

**Arguments**

y_true	true labels
y_pred	predicted labels

**Details**

Provides functionality for confusion\_matrix operations.

**Value**

matrix

**See Also**[VectorForgeML-package](#)**Examples**

```
y_true <- c(1,0,1,1)
y_pred <- c(1,0,0,1)
confusion_matrix(y_true, y_pred)
```

---

confusion_stats	<i>Confusion Matrix Statistics</i>
-----------------	------------------------------------

---

**Description**

Calculates accuracy, precision, recall, F1 from confusion matrix.

**Usage**

```
confusion_stats(cm)
```

**Arguments**

cm	confusion matrix
----	------------------

**Details**

Provides functionality for confusion\_stats operations.

**Value**

list

**See Also**

[VectorForgeML-package](#)

**Examples**

```
cm <- matrix(c(10, 2, 1, 15), nrow=2)
try({ confusion_stats(cm) })
```

---

DecisionTree-class	<i>Decision Tree Model</i>
--------------------	----------------------------

---

**Description**

Tree-based classification/regression algorithm.

**Details**

Provides functionality for DecisionTree operations.

**Value**

DecisionTree object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- DecisionTree$new()
X <- matrix(rnorm(20), nrow=10)
y <- sample(0:1, 10, replace=TRUE)
model$fit(X,y)
model$predict(X)
```

---

drop\_constant\_columns *Drop Constant Columns*

---

**Description**

Removes columns with zero variance.

**Usage**

```
drop_constant_columns(X, eps = 1e-12)
```

**Arguments**

X	input matrix/dataframe
eps	for param eps

**Details**

Provides functionality for drop\_constant\_columns operations.

**Value**

cleaned matrix

**See Also**

[VectorForgeML-package](#)

**Examples**

```
x <- data.frame(a=c(1,1,1), b=c(1,2,3))
drop_constant_columns(x)
```

---

f1\_score*F1 Score*

---

**Description**

Harmonic mean of precision and recall.

**Usage**

```
f1_score(y_true, y_pred, positive = NULL)
```

**Arguments**

y_true	true labels
y_pred	predicted labels
positive	positive class label

**Details**

Provides functionality for f1\_score operations.

**Value**

numeric f1 score

**See Also**

[VectorForgeML-package](#)

**Examples**

```
y_true <- c(1,0,1,1)
y_pred <- c(1,0,0,1)
f1_score(y_true, y_pred)
```

---

find\_best\_k*Find Best K*

---

**Description**

Finds optimal K value for KNN.

**Usage**

```
find_best_k(X, y, k_values = seq(1, 15, 2))
```

**Arguments**

X	features
y	labels
k_values	for k value

**Details**

Provides functionality for find\_best\_k operations.

**Value**

numeric best k

**See Also**

[VectorForgeML-package](#)

**Examples**

```
x <- matrix(rnorm(200), nrow=100)
y <- sample(0:1, 100, replace=TRUE)
find_best_k(x, y, k_values=c(1,3,5))
```

---

fit_linear_model	<i>Fit Linear Model (Fast C++ backend)</i>
------------------	--

---

**Description**

Internal helper for linear regression training.

**Usage**

```
fit_linear_model(X, y)
```

**Arguments**

X	numeric matrix
y	numeric vector

**Details**

Provides functionality for fit\_linear\_model operations.

**Value**

model object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
X <- matrix(rnorm(20), nrow=10)
y <- rnorm(10)
try({ fit_linear_model(X, y) })
```

---

**KMeans-class***KMeans Clustering*

---

**Description**

Unsupervised clustering algorithm.

**Details**

Provides functionality for KMeans operations.

**Value**

KMeans object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
x <- matrix(rnorm(20), nrow=10)
model <- KMeans$new()
model$fit(x)
```

---

**KNN-class***K-Nearest Neighbors Model*

---

**Description**

Instance-based learning algorithm.

**Details**

Provides functionality for KNN operations.

**Value**

KNN object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- KNN$new(k=3, mode="classification")
X <- matrix(rnorm(20), nrow=10)
y <- sample(0:1, 10, replace=TRUE)
model$fit(X,y)
model$predict(X)
```

---

LabelEncoder-class	<i>Label Encoder</i>
--------------------	----------------------

---

**Description**

Converts categorical labels into numeric values.

**Details**

Provides functionality for LabelEncoder operations.

**Value**

LabelEncoder object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
enc <- LabelEncoder$new()
x <- c("a", "b", "a")
enc$fit(x)
enc$transform(x)
```

---

LinearRegression-class	<i>Linear Regression Model</i>
------------------------	--------------------------------

---

**Description**

Fast linear regression implemented in C++ backend.

**Details**

Provides functionality for LinearRegression operations.

**Value**

LinearRegression object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- LinearRegression$new()  
X <- matrix(rnorm(100), 50, 2)  
y <- rnorm(50)  
model$fit(X, y)  
model$predict(X)
```

---

LogisticRegression-class

*Logistic Regression Model*

---

**Description**

Binary classification logistic regression.

**Details**

Provides functionality for LogisticRegression operations.

**Value**

LogisticRegression object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- LogisticRegression$new()  
X <- matrix(rnorm(20), nrow=10)  
y <- sample(0:1, 10, replace=TRUE)  
model$fit(X, y)  
model$predict(X)
```

---

`macro_f1`*Macro Precision*

---

**Description**

Computes macro-averaged precision.

**Usage**

```
macro_f1(y_true, y_pred)
```

**Arguments**

<code>y_true</code>	true labels
<code>y_pred</code>	predicted labels

**Details**

Provides functionality for `macro_f1` operations.

**Value**

numeric score

**See Also**

[VectorForgeML-package](#)

---

`macro_precision`*Macro Precision*

---

**Description**

Computes macro-averaged precision.

**Usage**

```
macro_precision(y_true, y_pred)
```

**Arguments**

<code>y_true</code>	true labels
<code>y_pred</code>	predicted labels

**Details**

Provides functionality for `macro_precision` operations.

**Value**

numeric score

**See Also**[VectorForgeML-package](#)

---

macro_recall	<i>Macro Precision</i>
--------------	------------------------

---

**Description**

Computes macro-averaged precision.

**Usage**

```
macro_recall(y_true, y_pred)
```

**Arguments**

y_true	true labels
y_pred	predicted labels

**Details**

Provides functionality for macro\_recall operations.

**Value**

numeric score

**See Also**[VectorForgeML-package](#)

---

MinMaxScaler-class	<i>Standard Scaler</i>
--------------------	------------------------

---

**Description**

Standardizes features by removing mean and scaling to unit variance.

**Details**

Provides functionality for MinMaxScaler operations.

**Value**

StandardScaler object

**See Also**[VectorForgeML-package](#)

Examples

```
s <- MinMaxScaler$new()
x <- matrix(rnorm(20), nrow=10)
s$fit(x)
s$transform(x)
```

---

mse	<i>Mean Squared Error</i>
-----	---------------------------

---

Description

Calculates regression error.

Usage

```
mse(y_true, y_pred)
```

Arguments

y_true	true values
y_pred	predicted values

Details

Provides functionality for mse operations.

Value

numeric mse

See Also

[VectorForgeML-package](#)

---

OneHotEncoder-class	<i>One Hot Encoder</i>
---------------------	------------------------

---

Description

Converts categorical variables into binary vectors.

Details

Provides functionality for OneHotEncoder operations.

Value

OneHotEncoder object

**See Also**[VectorForgeML-package](#)**Examples**

```
enc <- OneHotEncoder$new()  
df <- data.frame(a=c("x", "y", "x"))  
enc$fit(df)  
enc$transform(df)
```

---

PCA-class*Principal Component Analysis*

---

**Description**

Dimensionality reduction technique.

**Details**

Provides functionality for PCA operations.

**Value**

PCA object

**See Also**[VectorForgeML-package](#)**Examples**

```
model <- PCA$new(n_components=2)  
X <- matrix(rnorm(30), nrow=10)  
model$fit(X)  
model$transform(X)
```

---

Pipeline-class*Pipeline*

---

**Description**

Chains preprocessing and model steps.

**Details**

Provides functionality for Pipeline operations.

**Value**

Pipeline object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- Pipeline$new(list(StandardScaler$new()))
```

---

plot\_confusion\_matrix *Plot Confusion Matrix*

---

**Description**

Visualizes confusion matrix.

**Usage**

```
plot_confusion_matrix(cm, normalize = TRUE)
```

**Arguments**

cm	confusion matrix
normalize	Normlize

**Details**

Provides functionality for plot\_confusion\_matrix operations.

**Value**

plot

**See Also**

[VectorForgeML-package](#)

**Examples**

```
cm <- matrix(c(10, 2, 1, 15), nrow=2)
try({ plot_confusion_matrix(cm) })
```

---

precision_score	<i>Precision Score</i>
-----------------	------------------------

---

**Description**

Computes precision metric.

**Usage**

```
precision_score(y_true, y_pred, positive = NULL)
```

**Arguments**

y_true	true labels
y_pred	predicted labels
positive	positive class label

**Details**

Provides functionality for precision\_score operations.

**Value**

numeric precision

**See Also**

[VectorForgeML-package](#)

**Examples**

```
y_true <- c(1,0,1,1)
y_pred <- c(1,0,0,1)
precision_score(y_true, y_pred)
```

---

predict_linear_model	<i>Predict Linear Model</i>
----------------------	-----------------------------

---

**Description**

Predict values using trained linear model.

**Usage**

```
predict_linear_model(model, X)
```

**Arguments**

model	trained model
X	matrix

**Details**

Provides functionality for predict\_linear\_model operations.

**Value**

numeric vector

**See Also**

[VectorForgeML-package](#)

**Examples**

```
X <- matrix(rnorm(20), nrow=10)
y <- rnorm(10)
model <- fit_linear_model(X, y)
predict_linear_model(model, X)
```

---

r2_score	<i>R2 Score</i>
----------	-----------------

---

**Description**

Coefficient of determination.

**Usage**

```
r2_score(y_true, y_pred)
```

**Arguments**

y_true	true values
y_pred	predicted values

**Details**

Provides functionality for r2\_score operations.

**Value**

numeric r2 score

**See Also**

[VectorForgeML-package](#)

---

RandomForest-class	<i>Random Forest Model</i>
--------------------	----------------------------

---

**Description**

Ensemble of decision trees.

**Details**

Provides functionality for RandomForest operations.

**Value**

RandomForest object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- RandomForest$new(ntrees=5)
X <- matrix(rnorm(20), nrow=10)
y <- sample(0:1, 10, replace=TRUE)
model$fit(X,y)
model$predict(X)
```

---

recall_score	<i>Recall Score</i>
--------------	---------------------

---

**Description**

Computes recall metric.

**Usage**

```
recall_score(y_true, y_pred, positive = NULL)
```

**Arguments**

y_true	true labels
y_pred	predicted labels
positive	positive class label

**Details**

Provides functionality for recall\_score operations.

**Value**

numeric recall

**See Also**

[VectorForgeML-package](#)

**Examples**

```
y_true <- c(1,0,1,1)
y_pred <- c(1,0,0,1)
recall_score(y_true, y_pred)
```

---

RidgeRegression-class    *Ridge Regression Model*

---

**Description**

Linear regression with L2 regularization.

**Details**

Provides functionality for RidgeRegression operations.

**Value**

RidgeRegression object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- RidgeRegression$new()
X <- matrix(rnorm(20), nrow=10)
y <- rnorm(10)
model$fit(X,y,lambda=1.0)
model$predict(X)
```

---

rmse	<i>Root Mean Squared Error</i>
------	--------------------------------

---

**Description**

Square root of MSE.

**Usage**

```
rmse(y_true, y_pred)
```

**Arguments**

y_true	true values
y_pred	predicted values

**Details**

Provides functionality for rmse operations.

**Value**

numeric rmse

**See Also**

[VectorForgeML-package](#)

---

SoftmaxRegression-class	<i>Softmax Regression Model</i>
-------------------------	---------------------------------

---

**Description**

Multiclass logistic regression.

**Details**

Provides functionality for SoftmaxRegression operations.

**Value**

SoftmaxRegression object

**See Also**

[VectorForgeML-package](#)

**Examples**

```
model <- SoftmaxRegression$new()
X <- matrix(rnorm(20), nrow=10)
y <- sample(0:2, 10, replace=TRUE)
model$fit(X,y)
model$predict(X)
```

---

StandardScaler-class    *Drop Constant Columns*

---

**Description**

Removes columns with zero variance.

**Arguments**

X                      input matrix/dataframe

**Details**

Provides functionality for StandardScaler operations.

**Value**

cleaned matrix

**See Also**

[VectorForgeML-package](#)

**Examples**

```
s <- StandardScaler$new()
x <- matrix(rnorm(20), nrow=10)
s$fit(x)
s$transform(x)
```

---

train\_test\_split        *Train Test Split*

---

**Description**

Splits dataset into training and testing sets.

**Usage**

```
train_test_split(X, y, test_size = 0.2, seed = NULL)
```

**Arguments**

<code>X</code>	features
<code>y</code>	labels
<code>test_size</code>	proportion for test set
<code>seed</code>	for random seed

**Details**

Provides functionality for `train_test_split` operations.

**Value**

list

**See Also**

[VectorForgeML-package](#)

**Examples**

```
X <- matrix(rnorm(20), nrow=10)
y <- sample(0:1, 10, replace=TRUE)
train_test_split(X, y, test_size=0.2)
```

# Index

## \* **internal**

VectorForgeML-package, [2](#)

accuracy\_score, [3](#)

ColumnTransformer

(ColumnTransformer-class), [3](#)

ColumnTransformer-class, [3](#)

confusion\_matrix, [4](#)

confusion\_stats, [5](#)

DecisionTree (DecisionTree-class), [5](#)

DecisionTree-class, [5](#)

drop\_constant\_columns, [6](#)

f1\_score, [7](#)

find\_best\_k, [7](#)

fit\_linear\_model, [8](#)

KMeans (KMeans-class), [9](#)

KMeans-class, [9](#)

KNN (KNN-class), [9](#)

KNN-class, [9](#)

LabelEncoder (LabelEncoder-class), [10](#)

LabelEncoder-class, [10](#)

LinearRegression

(LinearRegression-class), [10](#)

LinearRegression-class, [10](#)

LogisticRegression

(LogisticRegression-class), [11](#)

LogisticRegression-class, [11](#)

macro\_f1, [12](#)

macro\_precision, [12](#)

macro\_recall, [13](#)

MinMaxScaler (MinMaxScaler-class), [13](#)

MinMaxScaler-class, [13](#)

mse, [14](#)

OneHotEncoder (OneHotEncoder-class), [14](#)

OneHotEncoder-class, [14](#)

PCA (PCA-class), [15](#)

PCA-class, [15](#)

Pipeline (Pipeline-class), [15](#)

Pipeline-class, [15](#)

plot\_confusion\_matrix, [16](#)

precision\_score, [17](#)

predict\_linear\_model, [17](#)

r2\_score, [18](#)

RandomForest (RandomForest-class), [19](#)

RandomForest-class, [19](#)

recall\_score, [19](#)

RidgeRegression

(RidgeRegression-class), [20](#)

RidgeRegression-class, [20](#)

rmse, [21](#)

SoftmaxRegression

(SoftmaxRegression-class), [21](#)

SoftmaxRegression-class, [21](#)

StandardScaler (StandardScaler-class),  
[22](#)

StandardScaler-class, [22](#)

train\_test\_split, [22](#)

VectorForgeML (VectorForgeML-package), [2](#)

VectorForgeML-package, [2](#)