

# Package: variantspark (via r-universe)

September 14, 2024

**Type** Package

**Title** A 'Sparklyr' Extension for 'VariantSpark'

**Version** 0.1.1

**Maintainer** Samuel Macêdo <samuelmacedo@recife.ifpe.edu.br>

**Description** This is a 'sparklyr' extension integrating 'VariantSpark' and R. 'VariantSpark' is a framework based on 'scala' and 'spark' to analyze genome datasets, see <<https://bioinformatics.csiro.au/>>. It was tested on datasets with 3000 samples each one containing 80 million features in either unsupervised clustering approaches and supervised applications, like classification and regression. The genome datasets are usually writing in VCF, a specific text file format used in bioinformatics for storing gene sequence variations. So, 'VariantSpark' is a great tool for genome research, because it is able to read VCF files, run analyses and return the output in a 'spark' data frame.

**License** Apache License 2.0 | file LICENSE

**LazyData** true

**Imports** sparklyr (>= 1.0.1)

**RoxygenNote** 6.1.1

**Suggests** testthat

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Samuel Macêdo [aut, cre], Javier Luraschi [aut]

**Repository** CRAN

**Date/Publication** 2019-06-13 16:20:03 UTC

## Contents

importance_tbl . . . . .	2
sample_names . . . . .	3

vs_connect . . . . .	3
vs_importance_analysis . . . . .	4
vs_read_csv . . . . .	5
vs_read_labels . . . . .	6
vs_read_vcf . . . . .	7
<b>Index</b>	<b>8</b>

---

importance_tbl	<i>Extract the importance data frame</i>
----------------	--

---

## Description

This function extracts the importance data frame from the Importance Analysis job.

## Usage

```
importance_tbl(importance, name = "importance_tbl")
```

## Arguments

importance	A job from the class ImportanceAnalysis, usually the output of vs_importance_analysis().
name	The name to assign to the copied table in Spark.

## Examples

```
## Not run:
library(sparklyr)
sc <- spark_connect(master = "local")
vsc <- vs_connect(sc)

hipster_vcf <- vs_read_vcf(vsc,
                          system.file("extdata/hipster.vcf.bz2",
                                        package = "variantspark"))
labels <- vs_read_labels(vsc,
                        system.file("extdata/hipster_labels.txt",
                                    package = "variantspark"))

importance <- vs_importance_analysis(vsc, hipster_vcf, labels, 10)
importance_tbl(importance)

## End(Not run)
```

---

sample_names	<i>Display sample names</i>
--------------	-----------------------------

---

**Description**

This function display the first N variant names.

**Usage**

```
sample_names(vcf_source, n_samples = NULL)
```

**Arguments**

`vcf_source` An object with VCFFeatureSource class, usually the output of the `vs_read_vcf()`.  
`n_samples` The number os samples to display.

**Value**

spark\_jobj, shell\_jobj

**Examples**

```
## Not run:  
library(sparklyr)  
  
sc <- spark_connect(master = "local")  
vsc <- vs_connect(sc)  
  
hipster_vcf <- vs_read_vcf(vsc,  
                           system.file("extdata/hipster.vcf.bz2",  
                                       package = "variantspark"))  
  
sample_names(hipster_vcf, 3)  
  
## End(Not run)
```

---

vs_connect	<i>Creating a variantspark connection</i>
------------	---

---

**Description**

You need to create a variantspark connection to use this extension. To do this, you pass as argument a spark connection that you can create using `sparklyr::spark_connect()`.

**Usage**

```
vs_connect(sc)
```

**Arguments**

sc                    A spark connection.

**Value**

A variantspark connection

**Examples**

```
library(sparklyr)
sc <- spark_connect(master = "spark://HOST:PORT")
connection_is_open(sc)
vsc <- vs_connect(sc)
spark_disconnect(sc)
```

---

vs\_importance\_analysis

*Importance Analysis*

---

**Description**

This function performs an Importance Analysis using random forest algorithm. For more details, please look at [here](#).

**Usage**

```
vs_importance_analysis(vsc, vcf_source, labels, n_trees)
```

**Arguments**

vsc                    A variantspark connection.

vcf\_source            An object with VCFFeatureSource class, usually the output of the vs\_read\_vcf().

labels                An object with CsvLabelSource class, usually the output of the vs\_read\_labels().

n\_trees                The number of trees using in the random forest.

**Value**

spark\_jobj, shell\_jobj

## Examples

```
## Not run:
library(sparklyr)
sc <- spark_connect(master = "local")
vsc <- vs_connect(sc)

hipster_vcf <- vs_read_vcf(vsc,
                          system.file("extdata/hipster.vcf.bz2",
                                        package = "variantspark"))

labels <- vs_read_labels(vsc,
                        system.file("extdata/hipster_labels.txt",
                                    package = "variantspark"))

vs_importance_analysis(vsc, hipster_vcf, labels, 10)

## End(Not run)
```

---

vs\_read\_csv

*Reading a CSV file*

---

## Description

The `vs_read_csv()` reads a CSV file format and returns a `jobj` object from `CsvFeatureSource` scala class.

## Usage

```
vs_read_csv(vsc, path)
```

## Arguments

<code>vsc</code>	A <code>variantspark</code> connection.
<code>path</code>	The file's path.

## Value

`spark_jobj`, `shell_jobj`

## Examples

```
## Not run:
library(sparklyr)

sc <- spark_connect(master = "local")
vsc <- vs_context(sc)

hipster_labels <- vs_read_csv(vsc,
```

```
system.file("extdata/hipster_labels.txt",
            package = "variantspark"))

hipster_labels

## End(Not run)
```

---

vs\_read\_labels      *Reading labels*

---

### Description

This function reads only the label column of a CSV file and returns a jobj object from CsvLabelSource scala class.

### Usage

```
vs_read_labels(vsc, path, label = "label")
```

### Arguments

vsc	A variantspark connection.
path	The file's path.
label	A string with the label column name.

### Value

spark\_jobj, shell\_jobj

### Examples

```
## Not run:
library(sparklyr)

sc <- spark_connect(master = "local")
vsc <- vs_context(sc)

labels <- vs_read_labels(vsc,
                        system.file("extdata/hipster_labels.txt",
                                    package = "variantspark"))

labels

## End(Not run)
```

---

`vs_read_vcf`*Reading a VCF file*

---

**Description**

The Variant Call Format (VCF) specifies the format of a text file used in bioinformatics for storing gene sequence variations. The format has been developed with the advent of large-scale genotyping and DNA sequencing projects, such as the 1000 Genomes Project. The `vs_read_vcf()` reads this format and returns a `jobj` object from `VCFFeatureSource` scala class.

**Usage**

```
vs_read_vcf(vsc, path)
```

**Arguments**

<code>vsc</code>	A variantspark connection.
<code>path</code>	The file's path.

**Value**

```
spark_jobj, shell_jobj
```

**Examples**

```
## Not run:
library(sparklyr)

sc <- spark_connect(master = "local")
vsc <- vs_context(sc)

hipster_vcf <- vs_read_vcf(vsc,
                          system.file("extdata/hipster.vcf.bz2",
                                        package = "variantspark"))

hipster_vcf

## End(Not run)
```

# Index

importance\_tbl, [2](#)

sample\_names, [3](#)

vs\_connect, [3](#)

vs\_importance\_analysis, [4](#)

vs\_read\_csv, [5](#)

vs\_read\_labels, [6](#)

vs\_read\_vcf, [7](#)