Package rocketc

```
import "github.com/aryanmaurya1/rocketc"
```

Overview ▾

Package rocketc is fast, simple and lightweight library for CSV data manipulation and mathematical computation involving 2D Matrices.

Index ▾

func VStack(m ...Matrix) Matrix
func Zeros(rows, cols int) Matrix
func (m *Matrix) Add(i float32, inplace bool) Matrix
func (m Matrix) Cols() int
func (m *Matrix) Div(i float32, inplace bool) Matrix
func (m *Matrix) Fill(i float32)
func (m Matrix) Filter(f func(v float32) bool) [][]bool
func (m *Matrix) Init()
func (m *Matrix) MakeMatrixUniform()
func (m *Matrix) Map(f func(v float32) float32, inplace bool) Matrix
func (m *Matrix) Mul(i float32, inplace bool) Matrix
func (m *Matrix) Ones()
func (m *Matrix) ReciproElementwise(inplace bool) Matrix
func (m Matrix) Rows() int
func (m Matrix) Shape() [2]int
func (m *Matrix) Sub(i float32, inplace bool) Matrix
func (m *Matrix) Transpose(inplace bool) Matrix
func (m *Matrix) Zeros()

Package files

rocketDataFrame.go rocketIO.go rocketMatrix.go rocketMatrixFunctions.go

func **DimensionEqual**

```
func DimensionEqual(m1, m2 Matrix) bool
```

DimensionEqual : Checks if m1 and m2 has same dimension, returns boolean.

func **PrintDataframe**

```
func PrintDataframe(d ...DataFrame)
```

PrintDataframe : for pretty printing of DataFrame.

func **PrintMatrix**

```
func PrintMatrix(m ...Matrix)
```

PrintMatrix : Prints Matrix in a pretty manner.

func **WriteCSVDataFrame**

```
func WriteCSVDataFrame(d DataFrame, fname string) error
```

WriteCSVDataFrame : Writes DataFrame into a file, so that DataFrame can be saved to disk for furthur. Takes a DataFrame and filename as arguments. Returns an error value in case of any error occured. Note : This function can only write DataFrame to file.

func **WriteCSVMatrix**

```
func WriteCSVMatrix(m Matrix, fname string) error
```

WriteCSVMatrix : Writes Matrix into a file, so that Matrix can be saved to disk for furthur. Takes a Matrix, slice of strings which are headers and filename as arguments. Returns an error value in case of any error occured. Note : This function can only write Matrix to file.

type **DataFrame**

DataFrame : Basic data container, stores data in form of 2D slices of string.

```
type DataFrame [][]string
```

func **Allocate**

```
func Allocate(row, col int) DataFrame
```

Allocate : Allocate a blank DataFrame of given size.

func **DropColumn**

```
func DropColumn(d DataFrame, i ...int) DataFrame
```

DropColumn : Drops columns from a DataFrame, takes a DataFrame and variable number of arguments which are indexes of columns to be droped.

func **GetColumnsDataFrame**

```
func GetColumnsDataFrame(d DataFrame, i ...int) DataFrame
```

GetColumnsDataFrame : Returns a DataFrame by only including specific columns whose column indexs are passed as argument. Take a DataFrame and variadic number integers which are column indexes.

func **ReadCSVDataFrame**

```
func ReadCSVDataFrame(fname string) (DataFrame, error)
```

ReadCSVDataFrame : Use this function to read CSV file completely. Takes a string filename. Returns DataFrame and not nil error value in case of any error occured. Currently ReadCSVDataFrame can only read matrices which do not have multiple values in single column. Note : Use this function if data contains both numeric and string values.

func **WipeDown**

```
func WipeDown(m DataFrame, l int) DataFrame
```

WipeDown : Returns unifom DataFrame by only including rows of length l in returned DataFrame. Takes a DataFrame and a integer as arguments.

func (DataFrame) Cols

```
func (d DataFrame) Cols() int
```

Cols : Returns number of columns in DataFrame, DataFrame must be uniform for accurate result.

func (DataFrame) Head

```
func (d DataFrame) Head(n int) DataFrame
```

Head : Returns first n rows of DataFrame including headers.

func (DataFrame) Headers

```
func (d DataFrame) Headers() []string
```

Headers : Returns header of the dataframe i.e row 0.

func (DataFrame) Rows

```
func (d DataFrame) Rows() int
```

Rows : Returns number of rows in DataFrame.

func (*DataFrame) SetHeaders

```
func (d *DataFrame) SetHeaders(header []string)
```

SetHeaders : Set custom column names to a DataFrame. Takes a slice of string containing name of columns.

func (DataFrame) Shape

```
func (d DataFrame) Shape() []int
```

Shape : Returns shape of DataFrame, slice of length 2.

type Matrix

Matrix : Basic numerical container (2D array), core datatype.

```
type Matrix [][]float32
```

func AddElementwise

```
func AddElementwise(m1, m2 Matrix) Matrix
```

AddElementwise : Adds two Matrix m1 and m2.

func ConvMatrix

```
func ConvMatrix(d DataFrame) (Matrix, error)
```

ConvMatrix : Converts numerical DataFrame into Matrix, returns err if dataframe contains values that cannot be converted into a float64.

func CopyMatrix

```
func CopyMatrix(src Matrix) Matrix
```

CopyMatrix : Returns the copy of src Matrix.

func DivElementwise

```
func DivElementwise(m1, m2 Matrix) Matrix
```

DivElementwise : Elementwise division of Matric m1 and m2.

func GetColumnsMatrix

```
func GetColumnsMatrix(m Matrix, i ...int) Matrix
```

GetColumnsMatrix : Returns a Matrix only containing columns whose index is passed. columns are 0 indexed. Takes variadic arguments i.e index.

func HStack

```
func HStack(m ...Matrix) Matrix
```

HStack : Returns a Matrix which is constructed by horizontally stacking every Matrix passed in argument. Takes variadic number of arguments i.e Matrix.

func Max

```
func Max(m Matrix, axis int) Matrix
```

Max : Returns a Matrix containing the maximum of elements of Matrix m according to given axis. If axis = 0, along the rows and if axis = 1, along the columns.

func Mean

```
func Mean(m Matrix, axis int) Matrix
```

Mean : Returns a Matrix containing the mean of elements of Matrix m according to given axis. If axis = 0, along the rows and if axis = 1, along the columns.

func Min

```
func Min(m Matrix, axis int) Matrix
```

Min : Returns a Matrix containing the minimum of elements of Matrix m according to given axis. If axis = 0, along the rows and if axis = 1, along the columns.

func MulElementwise

```
func MulElementwise(m1, m2 Matrix) Matrix
```

MulElementwise : Elementwise multiplication of Matrix m1 and m2.

func Multiply

```
func Multiply(m1, m2 Matrix) Matrix
```

Multiply : Returns the result of Matrix Multiplication of m1 and m2.

func Ones

```
func Ones(rows, cols int) Matrix
```

Ones : Returns a Matrix of ones of given rows and cols.

func Random

```
func Random(rows, cols int) Matrix
```

Random : Generates and returns a Matrix of given rows and cols containing random numbers.

func ReadCSVMatrix

```
func ReadCSVMatrix(fname string, dropFirst bool) (Matrix, error)
```

ReadCSVMatrix : Use this function to read CSV if you are sure that CSV file only contains parsable numerical values (float64).Takes a string filename and a boolean whether to drop first row or not.

Returns a Matrix. Note : Drop first row if it contains name of columns.

func SubElementwise

```
func SubElementwise(m1, m2 Matrix) Matrix
```

SubElementwise : Subtracts two Matrix m1 and m2.

func Sum

```
func Sum(m Matrix, axis int) Matrix
```

Sum : Returns a Matrix containing sum of elements of Matrix m according to given axis. If axis = 0, along the rows and if axis = 1, along the columns.

func VStack

```
func VStack(m ...Matrix) Matrix
```

VStack : Returns a Matrix which is constructed by vertically stacking every Matrix passed in argument. Takes variadic number of arguments i.e Matrix.

func Zeros

```
func Zeros(rows, cols int) Matrix
```

Zeros : Returns a Matrix of zeros of given rows and cols.

func (*Matrix) Add

```
func (m *Matrix) Add(i float32, inplace bool) Matrix
```

Add : Adds a given number to every element of Matrix, takes a boolean inplace if True performs addition inplace.

func (Matrix) Cols

```
func (m Matrix) Cols() int
```

Cols : Returns number of columns in Matrix.

func (*Matrix) Div

```
func (m *Matrix) Div(i float32, inplace bool) Matrix
```

Div : Divides every element of Matrix by i, takes a boolean inplace if True performs division inplace.

func (*Matrix) Fill

```
func (m *Matrix) Fill(i float32)
```

Fill : fills the Matrix with a given number i.

func (Matrix) Filter

```
func (m Matrix) Filter(f func(v float32) bool) [][]bool
```

Filter : Takes a function f as argument which in turn takes a float32 and returns bool. Filter function applies f to every element in m and returns the result as a boolean matrix.

func (*Matrix) Init

```
func (m *Matrix) Init()
```

Init : Initializes a Matrix to 1 X 1 Matrix containing zero.

func (*Matrix) MakeMatrixUniform

```
func (m *Matrix) MakeMatrixUniform()
```

MakeMatrixUniform : Makes Matrix rows of same size, by filling rest of the row with zero value of float32 if different size rows are present in Matrix.

func (*Matrix) Map

```
func (m *Matrix) Map(f func(v float32) float32, inplace bool) Matrix
```

Map : Applies given function to every element of Matrix, takes a boolean inplace if True performs mapping inplace.

func (*Matrix) Mul

```
func (m *Matrix) Mul(i float32, inplace bool) Matrix
```

Mul : Multiplies i to every element of the Matrix, takes a boolean inplace if True performs multiplication inplace.

func (*Matrix) Ones

```
func (m *Matrix) Ones()
```

Ones : Fills the Matrix with all zeros.

func (*Matrix) ReciproElementwise

```
func (m *Matrix) ReciproElementwise(inplace bool) Matrix
```

ReciproElementwise : Elementwise reciprocal of Matrix elements, takes a boolean inplace if True performs reciprocal inplace.

func (Matrix) Rows

```
func (m Matrix) Rows() int
```

Rows : Returns number of rows in Matrix.

func (Matrix) Shape

```
func (m Matrix) Shape() [2]int
```

Shape : Returns a int array []int containing the dimensions of Matrix.

func (*Matrix) Sub

```
func (m *Matrix) Sub(i float32, inplace bool) Matrix
```

Sub : Subtracts a given number from every element of Matrix, takes a boolean inplace if True performs subtraction inplace.

func (*Matrix) Transpose

```
func (m *Matrix) Transpose(inplace bool) Matrix
```

Transpose : Transpose the Matrix, takes a boolean inplace if True transposes Matrix inplace.

func (*Matrix) Zeros

```
func (m *Matrix) Zeros()
```

Zeros : Fills the Matrix with all ones.

Subdirectories

Name

..

examples