Package ‘SubCellBarCode’

April 29, 2024

Type Package

Title SubCellBarCode: Integrated workflow for robust mapping and visualizing whole human spatial proteome

Version 1.19.0

Author Taner Arslan

Maintainer Taner Arslan <taner.arslan@ki.se>

Description Mass-Spectrometry based spatial proteomics have enabled the proteome-wide mapping of protein subcellular localization (Orre et al. 2019, Molecular Cell). SubCellBarCode R package robustly classifies proteins into corresponding subcellular localization.

License GPL-2

Encoding UTF-8

LazyData true

Depends R (>= 3.6)

Suggests knitr, rmarkdown, BiocStyle

Imports Rtsne, scatterplot3d, caret, e1071, ggplot2, gridExtra, networkD3, ggrepel, graphics, stats, org.Hs.eg.db, AnnotationDbi

biocViews Proteomics, MassSpectrometry, Classification

RoxygenNote 7.1.1

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/SubCellBarCode

git_branch devel

git_last_commit 9fbd37c

git_last_commit_date 2023-10-24

Repository Bioconductor 3.19

Date/Publication 2024-04-28

Contents

`applyThresholdCompartment`

Apply thresholds to compartments

Description

Apply thresholds for all predictions to increase the true positive rate and remove poor classification.

Usage

```
applyThresholdCompartment(all.repA, all.repB, threshold.df)
```

Arguments

`all.repA`	data.frame; all predictions and probablity vectors for each protein in replicate A
`all.repB`	data.frame; all predictions and probablity vectors for each protein in replicate B
`threshold.df`	data.frame; collection od precision and recall values for each compaartment

Value

c.cls.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 550)
cls <- svmClassification(c.prots, df, markerProteins)

test.A <- cls[[1]]$svm.test.prob.out
test.B <- cls[[2]]$svm.test.prob.out

t.c.df <- computeThresholdCompartment(test.A, test.B)

all.A <- cls[[1]]$all.prot.pred
all.B <- cls[[2]]$all.prot.pred

c.cls.df <- applyThresholdCompartment(all.A, all.B, t.c.df)
}
```

---

applyThresholdNeighborhood

Apply thresholds to neighborhood classification

Description

Apply thresholds for all predictions at the neighborhood level to increase the true positive rate and remove poor classification.

Usage

```
applyThresholdNeighborhood(all.repA, all.repB, threshold.df)
```

Arguments

`all.repA`	data.frame; all predictions and probablity vectors for each protein in replicate A
`all.repB`	data.frame; all predictions and probablity vectors for each protein in replicate B
`threshold.df`	data.frame; collection od precision and recall values for each neighborhood

Value

n.cls.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 600)
cls <- svmClassification(c.prots, df, markerProteins)

test.A <- cls[[1]]$svm.test.prob.out
test.B <- cls[[2]]$svm.test.prob.out

t.n.df <- computeThresholdNeighborhood(test.A, test.B)

all.A <- cls[[1]]$all.prot.pred
all.B <- cls[[2]]$all.prot.pred

n.cls.df <- applyThresholdNeighborhood(all.A, all.B, t.n.df)
}
```

`calculateCoveredProtein` *Evaluate marker protein coverage*

Description

Given the proteomics data, number of overlapped marker proteins is calculated. Bar plot for each compartment is plotted.

Usage

```
calculateCoveredProtein(proteinIDs, markerproteins)
```

Arguments

`proteinIDs`	character; gene symbol id
`markerproteins`	character; 3365 proteins gene symbol ids

Value

covered.proteins

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])
}
```

---

calRowMean *Compute the means of replicates*

Description

Duplicated franctions A and B are summarized by taking their mean for each protein. After taking the mean, the data log2 transformed. Further, the 5 main fractions are used to check correlation between input datas. It is a helper function.

Usage

```
calRowMean(d.df)
```

Arguments

d.df	data.frame; A data frame of 10 fraction profiles consisting of replicate A and B.

Value

r.df

Examples

```
{

r.df <- calRowMean(SubCellBarCode::hcc827Ctrl)

}
```

candidateRelocatedProteins

Identify candidate relocated proteins

Description

Identify candidate condition-dependent relocated proteins by comparing neighborhood classifications with respect to protein-protein pearson correlation and minumum PSM, peptide spectrum matching, count.

Usage

```
candidateRelocatedProteins(
  sampleCls1,
  s1PSM,
  s1Quant,
  sampleCls2,
  s2PSM,
  s2Quant,
  annotation = FALSE,
  min.psm = 2,
  pearson.cor = 0.8
)
```

Arguments

sampleCls1	data.frame; merged classification, combination of compartment and neighborhood classification.
s1PSM	data.frame; minimum PSM count table across ten TMT channel
s1Quant	data.frame; fractionation quantification data
sampleCls2	data.frame; merged classification, combination of compartment and neighborhood classification.
s2PSM	data.frame; minimum PSM count table across ten TMT channel
s2Quant	data.frame; fractionation quantification data
annotation	boolean; labeling the selected proteins
min.psm	numeric; minimum psm, peptide spectra matching value
pearson.cor	numeric; pearson correlation threshold

Value

candidate.df

Examples

```
{

candidate.df <- candidateRelocatedProteins(hcc827GEFClass, hcc827GefPSMCount,
hcc827GEF, hcc827GEFClass, hcc827GefPSMCount, hcc827GEF,
annotation = FALSE)


}
```

---

compareCls *Compare exon and gene centric classifications*

Description

Comparison of the gene centric and exon centric classification. Additionally, correlation analysis is performed using quantification data.

Usage

```
compareCls(geneCls, exonCls)
```

Arguments

geneCls	data frame gene centric classification output
exonCls	data frame exon centric classification output

Value

c.df

Examples

```
{

exon.cls <- data.frame(Protein = c("ENSE00000331854",
                                   "ENSE00000331855",
                                   "ENSE00000331859"),
                       NeighborhoodCls = c("Cytosol",
                                           "Cytosol",
                                           "Cytosol"),
                       CompartmentCls = c("C1","C1","C1"),
                       Secretory = c(0.1, 0.1, 0.1),
                       Nuclear = c(0.2, 0.2, 0.2),
                       Cytosol = c(0.2, 0.2, 0.2),
                       Mitochondria = c(0.2, 0.2, 0.2),
                       S1 = c(0.2, 0.2, 0.2),
                       S2 = c(0.2, 0.2, 0.2),
                       S3 = c(0.2, 0.2, 0.2),
```

```
                          S4 = c(0.2, 0.2, 0.2),
                          N1 = c(0.2, 0.2, 0.2),
                          N2 = c(0.2, 0.2, 0.2),
                          N3 = c(0.2, 0.2, 0.2),
                          N4 = c(0.2, 0.2, 0.2),
                          C1 = c(0.2, 0.2, 0.2),
                          C2 = c(0.2, 0.2, 0.2),
                          C3 = c(0.2, 0.2, 0.2),
                          C4 = c(0.2, 0.2, 0.2),
                          C5 = c(0.2, 0.2, 0.2),
                          M1 = c(0.2, 0.2, 0.2),
                          M2 = c(0.2, 0.2, 0.2),
                          GeneSymbol = c("COPB1", "COPB1", "COPB1"),
                          PeptideCount = c(2, 4, 7))

gene.cls <- data.frame(Protein = c("COPB1"),
NeighborhoodCls = c("Cytosol"),
CompartmentCls = c("C1"),
Secretory = c(0.1),
Nuclear = c(0.2),
Cytosol = c(0.2),
Mitochondria = c(0.2),
S1 = c(0.2),
S2 = c(0.2),
S3 = c(0.2),
S4 = c(0.2),
N1 = c(0.2),
N2 = c(0.2),
N3 = c(0.2),
N4 = c(0.2),
C1 = c(0.2),
C2 = c(0.2),
C3 = c(0.2),
C4 = c(0.2),
C5 = c(0.2),
M1 = c(0.2),
M2 = c(0.2))

comp.df <- compareCls(gene.cls, exon.cls)

}
```

---

computeThresholdCompartment

Probability threshold for compartment classification

Description

Thresholds for each compartment are decided to get confident predictions.

Usage

```
computeThresholdCompartment(test.repA, test.repB)
```

Arguments

test.repA	data.frame; test predictions, observation and probablity vectors for each protein in replicate A
test.repB	data.frame; test predictions, observation and probablity vectors for each protein in replicate B

Value

threshold.compartment.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 550)
cls <- svmClassification(c.prots, df, markerProteins)

test.A <- cls[[1]]$svm.test.prob.out
test.B <- cls[[2]]$svm.test.prob.out

t.c.df <- computeThresholdCompartment(test.A, test.B)
}
```

---

computeThresholdNeighborhood

Probability threshold for neighborhood classification

Description

Thresholds for each neighborhood are decided to get confident predictions.

Usage

```
computeThresholdNeighborhood(test.repA, test.repB)
```

Arguments

test.repA	data.frame; test predictions, observation and probablity vectors for each protein in replicate A
test.repB	data.frame; test predictions, observation and probablity vectors for each protein in replicate B

Value

threshold.neighborhood.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 600)
cls <- svmClassification(c.prots, df, markerProteins)

test.A <- cls[[1]]$svm.test.prob.out
test.B <- cls[[2]]$svm.test.prob.out

t.n.df <- computeThresholdNeighborhood(test.A, test.B)
}
```

---

convert2symbol *Convert identifier to gene symbol*

Description

Identifier for each feature should be converted into gene symbols unless they are not gene symbols

Usage

```
convert2symbol(df, id = "UNIPROT")
```

Arguments

df	data.frame; fractionated proteomics data where data contains 10 columns of duplicated 5 fractionations and rownames must be identifier e.g. UNIPROT, Entrez ID
id	caharacter; identifier id for each protein

Value

df

Examples

```
{

df <- data.frame(Uniprot = c("A4D0S4","A8TX70","O00305","O00337"),
Organism = rep("Homo Sap.", 4))

rownames(df) <- df$Uniprot
}
```

`hcc827Ctrl` *HCC827 Control Cell Line*

Description

Subcellular fractionated cell line.

Usage

```
hcc827Ctrl
```

Format

A data frame where 10480 protein gene-centric ids and 5 replicated subcellular fractions.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827Ctrl)
}
```

`hcc827CtrlPSMCount` *Minimum PSM Count in HCC827Ctrl Cell Line.*

Description

Minimum PSM, Peptide Sequence Match, Count table for HCC827Ctrl Cell Line.

Usage

```
hcc827CtrlPSMCount
```

Format

A data frame where 10480 protein gene-centric ids minimum PSM count.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827CtrlPSMCount)
}
```

---

hcc827exon *HCC827 Control Exon Cell Line*

Description

Exon-centric sub data of hcc827 fractionated data.

Usage

```
hcc827exon
```

Format

A data frame where 500 exon-centric ensemble identifiers, corresponding gene symbols, 5 replicated subcellular fractions and number of unique peptides matched to associated exon.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827exon)
}
```

hcc827GEF — *Gefitinib treated HCC827 Cell Line*

Description

HCC827 cell line was treated with Gefitinib which is EGFR inhibition.

Usage

```
hcc827GEF
```

Format

A data frame where 10398 protein gene-centric ids and 5 replicated subcellular fractions with duplicates.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827GEF)
}
```

hcc827GEFClass — *Gefitinib treated HCC827 Cell Line Classification*

Description

Gefitinib treated HCC827 cell line classification contains both neighborhood and compartment level. The data will be used for the relocalization analysis.

Usage

```
hcc827GEFClass
```

Format

A data frame where 10398 protein gene-centric ids and corresponding compartment and neighborhood classification alon with classification probabilities.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827GEFClass)
}
```

---

hcc827GefPSMCount *Minimum PSM Count in HCC827 Gefitinib Cell Line.*

Description

Minimum PSM, Peptide Sequence Match, Count table for HCC827 Gefitinib Cell Line.

Usage

```
hcc827GefPSMCount
```

Format

A data frame where 10398 protein gene-centric ids minimum PSM count.

References

Orre et al. 2019 Cell 73, 1-17

Examples

```
{
head(hcc827GefPSMCount)
}
```

---

loadData *Load the fractionated proteomics data*

Description

Sampled median normalized TMT ratios are checked if there is any "NA" valeus. If any, the corresponding row is filtered out. Later, the data is normalized by taking log2.

Usage

```
loadData(protein.data)
```

Arguments

protein.data	data.frame; fractionated proteomics data where data contains 10 columns of duplicated 5 fractionations and rownames must be gene-centric protein names

Value

protein.data.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl[1:20,])
}
```

---

markerProteins *Marker Proteins Source*

Description

Data for the proteins whose localizations were well characterized. It also contains color codes for each compartment and median fractionation profiles for 5 fractions which are Cyto., Nsol., NucI., Horg., Lorg., with replicates A and B. These fractionation profiles will be used for the marker protein quality control.

Usage

```
markerProteins
```

Format

A data frame of 3365 proteins as rows and 13 columns headers.

References

Orre et al. 2019 Cell 73, 1-17

---

markerQualityControl *Evaluate the quality of the marker proteins*

Description

Given the proteomics data, quality of the overlapped marker proteins are evaluated by correlating replicates of fractions.

Usage

```
markerQualityControl(coveredProteins, protein.data)
```

Arguments

`coveredProteins`
: character; list of marker proteins, gene symbols, that are covered in 3365 marker proteins.

`protein.data`
: data.frame; fractionated proteomics data, rownames are gene symbols associated protein.

Value

robustMarkers

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

r.markers <- markerQualityControl(c.prots[1:5], df)
}
```

---

mergeCls — *Merge compartment and neighborhood classification*

Description

Compartment and neighborhood classifications are merged for the single output.

Usage

```
mergeCls(compartmentCls, neighborhoodCls)
```

Arguments

`compartmentCls`
: data.frame; all predictions, including unclassified as well, and probablity vectors for each protein in compartment classification

`neighborhoodCls`
: data.frame; all predictions, including unclassified as well, and probablity vectors for each protein in compartment classification

Value

cls.df

Examples

```
{

#create mock data
com.df <- data.frame(Proteins = "TP53",
svm.pred = "N1",
S1 = as.numeric(0.02),
S2 = as.numeric(0.02),
S3 = as.numeric(0.02),
S4 = as.numeric(0.02),
N1 = as.numeric(0.72),
N2 = as.numeric(0.02),
N3 = as.numeric(0.02),
N4 = as.numeric(0.02),
C1 = as.numeric(0.02),
C2 = as.numeric(0.02),
C3 = as.numeric(0.02),
C4 = as.numeric(0.02),
C5 = as.numeric(0.02),
M1 = as.numeric(0.02),
M2 = as.numeric(0.02))

rownames(com.df) <- "TP53"

neig.df <- data.frame(Proteins = "TP53",
svm.pred.all = "Nuclear",
Secretory = as.numeric(0.01),
Nuclear = as.numeric(0.95),
Cytosol = as.numeric(0.02),
Mitochondria = as.numeric(0.02))

rownames(neig.df) <- "TP53"

cls.df <- mergeCls(com.df, neig.df)

}
```

`mergeProbability` *Merge compartment probabilities to neighborhood probabilities*

Description

Compartment levels classifications are summed up to associated neighborhood levels. It is a helper function.

Usage

```
mergeProbability(df)
```

Arguments

df	data.frame; all predictions at the neighborhood level and probablity vectors for each protein

Value

merged.df

Examples

```
{

#create mock data
df <- data.frame(Protein = "TP53",
S1 = as.numeric(0.02),
S2 = as.numeric(0.02),
S3 = as.numeric(0.02),
S4 = as.numeric(0.02),
N1 = as.numeric(0.72),
N2 = as.numeric(0.02),
N3 = as.numeric(0.02),
N4 = as.numeric(0.02),
C1 = as.numeric(0.02),
C2 = as.numeric(0.02),
C3 = as.numeric(0.02),
C4 = as.numeric(0.02),
C5 = as.numeric(0.02),
M1 = as.numeric(0.02),
M2 = as.numeric(0.02))

rownames(df) <- "TP53"

merged.df <- mergeProbability(df)

}
```

---

plotBarcode *Visualize the SubCellBarCode*

Description

Stacked bar plot are plotted for compartment and neighborhood level with respect to classification probabilities.

Usage

```
plotBarcode(sampleClassification, protein, s1PSM)
```

Arguments

`sampleClassification`
: data.frame; merged classification, combination of compartment and neighborhood classification.

`protein`
: character; protein gene symbol name

`s1PSM`
: data.frame; minimum PSM count table. Row names should be gene centric protein id.

Value

proteinPlot

Examples

```
{

#create mock data
plot.df <- data.frame(Protein = "TP53",
NeighborhoodCls = "Nuclear",
CompartmentCls = "N1",
Secretory = as.numeric(0.01),
Nuclear = as.numeric(0.95),
Cytosol = as.numeric(0.02),
Mitochondria = as.numeric(0.02),
S1 = as.numeric(0.02),
S2 = as.numeric(0.02),
S3 = as.numeric(0.02),
S4 = as.numeric(0.02),
N1 = as.numeric(0.72),
N2 = as.numeric(0.02),
N3 = as.numeric(0.02),
N4 = as.numeric(0.02),
C1 = as.numeric(0.02),
C2 = as.numeric(0.02),
C3 = as.numeric(0.02),
C4 = as.numeric(0.02),
C5 = as.numeric(0.02),
M1 = as.numeric(0.02),
M2 = as.numeric(0.02))

rownames(plot.df) <- "TP53"

psm.df <- data.frame(Protein = "TP53",
PSMs.for.quant = as.numeric(31))

rownames(psm.df) <- "TP53"

proteinPlot <- plotBarcode(plot.df, "TP53", psm.df)
}
```

plotMultipleProtein *Visualization of multiple protein localizations*

Description

Distributions of subcellular localizations of multiple proteins both ar the compartment and neighborhood level are plotted.

Usage

```
plotMultipleProtein(sampleClassification, proteinList)
```

Arguments

sampleClassification	data.frame; merged classification, combination of compartment and neighborhood classifications per protein.
proteinList	vector; protein gene symbol names.

Value

multipleProt.df

Examples

```
{

proteasome26s <- c("PSMA7", "PSMC3", "PSMB1", "PSMA1", "PSMA3","PSMA4",
"PSMA5", "PSMB4", "PSMB6", "PSMB5","PSMC2", "PSMC4", "PSMB3", "PSMB2",
"PSMD4", "PSMA6", "PSMC1", "PSMC5", "PSMC6", "PSMB7", "PSMD13")

exp.cls.df <- SubCellBarCode::hcc827GEFClass

multipleProt.df <- plotMultipleProtein(exp.cls.df, proteasome26s )

}
```

replacePrediction *Replace compartment predictions to neighborhood predictions*

Description

Compartment level classifications are replaced with neighborhood level assignment. It is a helper function.

Usage

```
replacePrediction(df, column = c("svm.pred.all", "Observation", "svm.pred"))
```

Arguments

df	data.frame; all predictions at the compartment level and probablity vectors for each protein
column	character; selected column in the data frame, df

Value

replaced.df

Examples

```
{

#define mock data frame
df <- data.frame(svm.pred.all = c("S1","S2","S3","S4",
"N1","N2","N3","N4",
"C1","C2","C3","C4","C5",
"M1","M2"))

df$svm.pred.all <- as.character(df$svm.pred.all)
df$Prob <- "1"

df <- replacePrediction(df, column = "svm.pred.all")
}
```

---

sankeyPlot — *Sankey plot for condition-dependent protein relocalization*

Description

Identify candidate condition-dependent relocated proteins by comparing neighborhood classifications.

Usage

```
sankeyPlot(sampleCls1, sampleCls2)
```

Arguments

sampleCls1	data.frame; merged classification, combination of compartment and neighborhood classification.
sampleCls2	data.frame; merged classification, combination of compartment and neighborhood classification.

Value

label.link.df

Examples

```
{

exp.cls.df <- SubCellBarCode::hcc827GEFClass

sankeyData <- sankeyPlot(exp.cls.df, exp.cls.df)

}
```

---

sumProbability — *Sum compartment test data probabilities to neighborhood probabilities*

Description

Compartment levels classifications on the test data are summed up to associated neighborhood levels. It is a helper function.

Usage

```
sumProbability(df)
```

Arguments

df	data.frame; test data classifications at the neighborhood level and probablity vectors for each protein.

Value

summed.df

Examples

```
{

#create mock data
df <- data.frame(Protein = "TP53",
svm.pred = "N1",
S1 = as.numeric(0.02),
S2 = as.numeric(0.02),
S3 = as.numeric(0.02),
S4 = as.numeric(0.02),
N1 = as.numeric(0.72),
N2 = as.numeric(0.02),
N3 = as.numeric(0.02),
```

```
N4 = as.numeric(0.02),
C1 = as.numeric(0.02),
C2 = as.numeric(0.02),
C3 = as.numeric(0.02),
C4 = as.numeric(0.02),
C5 = as.numeric(0.02),
M1 = as.numeric(0.02),
M2 = as.numeric(0.02))

rownames(df) <- "TP53"

sum.df <- sumProbability(df)

}
```

`svmClassification` *Protein subcellular localization classification*

Description

Support Vector Machine classifier is trained and used for prediction of protein subcellular localization

Usage

```
svmClassification(markerProteins, protein.data, markerprot.df)
```

Arguments

`markerProteins`	character; robust marker proteins along with subcellular localization that are present in the given data.
`protein.data`	data.frame; fractionated proteomics data
`markerprot.df`	data.frame; collection of marker proteins along with corresponding subcellular localization

Value

all.classifications

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 500)
```

```
cls <- svmClassification(c.prots, df, markerProteins)

}
```

`svmExternalData` *Peptide/exon/transcript centric or PTM enriched classification*

Description

Peptide/exon/transcript centric or PTM enriched classification is applied to predict localization of them.

Usage

```
svmExternalData(df, modelA, modelB)
```

Arguments

df	data frame fractionated additional data
modelA	model for the replicate A classification
modelB	model for the replicate B classification

Value

c.cls.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(7)
c.prots <- sample(c.prots, 550)
cls <- svmClassification(c.prots, df, markerProteins)
modelA <- cls[[1]]$model
modelB <- cls[[2]]$model

exon.cls <- svmExternalData(SubCellBarCode::hcc827exon,
modelA = modelA, modelB = modelB)
}
```

tsneVisualization *Visualization of marker proteins by t-SNE map*

Description

The marker proteins are visualized in 3D t-SNE map to see the distributions of the marker proteins.

Usage

```
tsneVisualization(protein.data, markerProteins, dims, theta, perplexity)
```

Arguments

protein.data	data.frame; fractionated proteomics data
markerProteins	character; robust marker proteins, gene symbols, that are present in the given data and overlapped with package's marker protein list.
dims	integer; dimensionality
theta	numeric; Speed/accuracy trade-off ,increase for less accuracy
perplexity	integer; Perplexity parameter

Value

tsneMap.df

Examples

```
{

df <- loadData(SubCellBarCode::hcc827Ctrl)

c.prots <- calculateCoveredProtein(rownames(df), markerProteins[,1])

set.seed(21)
tsneMap.df <- tsneVisualization(protein.data = df,
markerProteins = c.prots[1:20],
dims = 2, theta = c(0.4), perplexity = c(5))
}
```

Index