Многомерное масштабирование

Многомерное масштабирование (MDS) - это средство визуализации уровня сходства отдельных объектов в наборе данных. MDS используется для перевода информации о попарных« расстояниях между n объектами в наборе данных в конфигурацию n точек, отображаемых в абстрактное декартово пространство. С технической точки зрения, MDS относится к набору связанных методов ординации , используемых при визуализации информации , в частности, для отображения информации, содержащейся в матрице расстояний . Это форма нелинейного уменьшения размерности. Учитывая матрицу расстояний расстояниями между каждой парой объектов в наборе данных, и выбрав число измерений N , MDA алгоритм помещает каждый объект в N - мерном пространстве (низкоразмерное представление), так что между объектами расстояния сохранялись как можно лучше. Для N = 1, 2 и 3 полученные точки можно визуализировать на диаграмме рассеяния .

Формально можно задачу многомерного шкалирования сформулировать следующим образом. Имеется обучающая выборка объектов $\mathbf{X}^p = \{x_1, ..., x_p\} \subset \mathbf{X}$. Заданы расстояния $R_{ij} = \rho(x_i, x_j)$ для некоторых пар обучающих объектов $(i, j) \in D$ Требуется для каждого объекта $x_i \in \mathbf{X}^p$ построить его признаковое описание вектор $x_i = (x_i^1, ..., x_i^n)$ в евклидовом пространстве $\Re^n$ так, чтобы евклидовы расстояния d_{ij} между объектами x_i и x_j:

$$d_{ij}^2 = \sum_{d=1}^{n} (x_i^d - x_j^d)^2 \quad (1)$$

как можно точнее приближали исходные расстояния R_{ij} для всех $(i, j) \in D$.

Данное требование можно формализовать по-разному; один из наиболее распространённых способов через минимизацию функционала, называемого стрессом:

$$S(\mathbf{X}^p) = \sum_{(i,j)\in D} w_{ij}(d_{ij} - R_{ij})^2 \to \min \quad (2)$$

где минимум берётся по совокупности pn переменных $(x_i^d)_{i=1,p}^{d=1,n}$

Размерность n обычно задаётся небольшая. В частности, при $n = 2$ многомерное шкалирование позволяет отобразить выборку в виде множества точек на плоскости. Плоское представление, как правило, иска-

жено ($S > 0$), но в целом отражает основные структурные особенности многомерной выборки, в частности, её кластерную структуру. Поэтому двумерное шкалирование часто используют как инструмент предварительного анализа и понимания данных.

Веса w_{ij} задаются исходя из целей шкалирования. Обычно берут $w_{ij} = (R_{ij})^{\gamma}$. При $\gamma < 0$ приоритет отдаётся более точному приближению малых расстояний; при $\gamma > 0$ больших расстояний. Наиболее адекватным считается значение $\gamma = -2$, когда функционал стресса приобретает физический смысл потенциальной энергии системы p материальных точек, связанных упругими связями; требуется найти равновесное состояние системы, в котором потенциальная энергия минимальна. Функционал стресса $S(\mathbf{X}^p)$ сложным образом зависит от pn переменных, имеет огромное количество локальных минимумов, и его вычисление довольно трудоёмко.Поэтому многие алгоритмы многомерного шкалирования основаны на итерационном размещении объектов по одному. На каждой итерации оптимизируются евклидовы координаты только одного из объектов, а координаты остальных объектов, вычисленные на предыдущих итерациях, полагаются фиксированными. Кратко формально опишим варианты реализации MDS

Классическое многомерное масштабирование

MDS также известен как масштабирование Торгерсона – Гауэра. Он, как уже говорилось выше, принимает входную матрицу, определяющую различия между парами элементов, и выводит координатную матрицу, конфигурация которой минимизирует функцию потерь, называемую деформацией. На практике MDS стремится аппроксимировать исходное пространство данных его представлением пространством более низкой размерности, минимизируя функцию потерь. Общие формы функций потерь называются напряжением на расстоянии MDS и деформацией в классической MDS. Напряжение определяется:

$$S_{train_D}(x_1, x_2, ..., x_N) = (\frac{\sum_{i,j} (b_{ij} - \mathbf{x}_i^T \mathbf{x}_j)^2}{\sum_{i,j} b_{ij}^2})^{1/2} \quad (3)$$

где $\mathbf{x}_i$ вектор в N - мерном пространстве; $\mathbf{x}_i^T \mathbf{x}_j$ - скалярное произведение векторов $\mathbf{x}_i$ и $\mathbf{x}_j$; b_{ij} - элементы матрицы $\mathbf{B}$ определенные на шаге 2 следующего алгоритма, которые вычисляются из расстояний.

Шаги классического алгоритма MDS:

Классическая MDS использует тот факт, что координатная матри-

ца $\mathbf{X}$ сможет быть получена разложением на собственные значения из $\mathbf{B} = \mathbf{X}\mathbf{X}^T$. Матрица $\mathbf{B}$ получается из матрицы ,близости $\mathbf{D}$ с помощью двойного центрирования.

1. Получить квадратную матрицу близости $\mathbf{D}^{(2)} = [d_{ij}^2]$.

2. Применить двойное центрирование: $\mathbf{B} = \frac{1}{2}\mathbf{C}\mathbf{D}^{(2)}$, здесь $\mathbf{C} = \mathbf{I} - \frac{1}{p}\mathbf{J}_p$ где p - количество объектов; $\mathbf{I}$ -это единичная матрица размерности $p \times p$; $\mathbf{J}_p$ также единичная матрица размерности $p \times p$.

3. Из матрицы $\mathbf{B}$ получаем m наибольших собственных значений $\lambda_1, \lambda_2, ..., \lambda_m$ и соответствующие собственные вектора $\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_m$ (m - требуемая размерность редуцированного пространства объектов).

4. Получение $\mathbf{X} = \mathbf{E}_m \mathbf{\Lambda}_m^{1/2}$, где $\mathbf{E}_m$ матрица их m собственных векторов; $\mathbf{\Lambda}_m$ -диагональная матрица из m собственных значений матрицы $\mathbf{B}$

Классическая MDS предполагает евклидовы расстояния.

Метрическое многомерное масштабирование ($mMDS$)

Это надмножество классической MDS, которое обобщает процедуру оптимизации на множество функций потерь и входных матриц известных расстояний с весами и т. Д. Полезная функция потерь в этом контексте называется стрессом (см. выше) , который часто минимизируется с помощью процедуры, называемой мажоризацией стресса . Метрика MDS минимизирует функцию затрат, называемую «стресс», которая представляет собой остаточную сумму квадратов:

$$S_{tress_D}(x_1, x_2, ..., x_N) = (\sum_{i \neq j = 1, ...N} (d_{ij} - \|\mathbf{x}_i - \mathbf{x}_j\|)^2)^{1/2} \quad (4)$$

При масштабировании показателей используется преобразование степени с показателем степени, управляемым пользователем. γ :d_{ij}^{γ} и $d_{ij}^{2\gamma}$.В классическом масштабировании $\gamma = 1$.

Неметрическое многомерное масштабирование ($nMDS$)

В отличие от метрической MDS, неметрическая MDS находит как непараметрическую монотонную связь между различиями в матрице элемент-элемент и евклидовыми расстояниями между элементами, так и расположением каждого элемента в низкоразмерном пространстве. Взаимосвязь обычно определяется с помощью изотонической регрессии : пусть $\mathbf{x}$ обозначим вектор близости, $\mathbf{f}(\mathbf{x})$ монотонное преобразование $\mathbf{x}$, а также d точечные расстояния; затем необходимо найти координаты, которые минимизируют так называемое напряжение:

$$S_{tress} = \sqrt{\frac{\sum (\mathbf{f}(\mathbf{x}) - d)^2}{\sum d^2}} \quad (5)$$

Существует несколько вариантов этой функции затрат. Программы MDS автоматически минимизируют стресс, чтобы получить решение. Ядро неметрического алгоритма MDS - это двойной процесс оптимизации. Сначала нужно найти оптимальное монотонное преобразование близости. Во-вторых, точки конфигурации должны быть расположены оптимальным образом, чтобы их расстояния как можно ближе соответствовали масштабируемой близости. Основные шаги неметрического алгоритма MDS:

1. Найдти случайную конфигурацию точек, например, путем выборки из нормального распределения.

2. Вычислить расстояния d между точками.

3. Найдти оптимальное монотонное преобразование близости, чтобы получить оптимально масштабированные данные $\mathbf{f}(\mathbf{x})$.

4. Минимизировать напряжение между оптимально масштабированными данными и расстояниями, найдя новую конфигурацию точек.

5. Сравнить напряжение с некоторым критерием. Если напряжение достаточно мало, выйдти из алгоритма, иначе вернуться к 2.

Анализ наименьшего пространства (SSA) Луи Гутмана является примером неметрической процедуры MDS.

Обобщенное многомерное масштабирование (GMD)

Расширение метрического многомерного масштабирования, в котором целевым пространством является произвольное гладкое неевклидово пространство. В случаях, когда различия - это расстояния на поверхности, а целевое пространство - это другая поверхность, GMDS позволяет найти вложение одной поверхности в другую с минимальным искажением.

Анализируемые данные представляют собой совокупность M объектов, для которых определена функция расстояния, $d_{i,j}$: дистанция между i-тым и j-ым объекектами. Эти расстояния являются элементами матрицы несходства D.

Целью MDS является зная D, найти M векторов $\mathbf{x}_1, ...\mathbf{x}_M \in \Re^N$ таких,что $\|\mathbf{x}_i - \mathbf{x}_j\| \approx d_{ij}$ для всех $i, j \in 1, ...M$

где $\|.\|$ - векторная норма. В классическом MDS эта норма является евклидовым расстоянием, но, в более широком смысле, это может быть

метрическая или произвольная функция расстояния.

Другими словами, MDS пытается найти отображение M объектов в $\Re^N$ таким образом, что расстояния сохраняются.

Если размер N выбран 2 или 3, то можно построить вектора $\mathbf{x}_i$ чтобы получить визуализацию сходства между M объекты. При этом важно, что эти векторы не уникальны пр данном евклидовым расстоянием они могут произвольно перемещаться, вращаться и отражаться, поскольку эти преобразования не изменяют попарные расстояния $\|\mathbf{x}_i - \mathbf{x}_j\|$

Существуют различные подходы к определению векторов $\mathbf{x}_i$.Обычно МДС формулируется как задача оптимизации,где $\mathbf{x}_1, ...\mathbf{x}_M$ достовляют минимум некоторой функции стоимости:

$$\arg\min_{\mathbf{x}_1,....,\mathbf{x}_M} \sum_{i<j} (\|\mathbf{x}_i - \mathbf{x}_j\| - d_{ij})^2 \quad (6)$$

Затем решение может быть найдено с помощью методов численной оптимизации. Для некоторых специально выбранных функций стоимости минимизаторы могут быть сформулированы аналитически в терминах разложения собственных матриц.

originalsource

Пример реализации алгоритма MDS **на** *python* **и** R**.**

Для рачетов использовались данный о Российских регинах. В *python* реализации исходным было $3D$ пространство, регионы размечены на три класса. Для R реализаци исходное пространство $4D$, а регионы размечены на четыре класса.

В scikitlearn мы задаем два параметра ncomponents = 2 (размерность выходного редуцируемого пространства) и metric = True (метрический MDS) или False (неметрический MDS) остальные параметры принимаются по умолчанию

Код *python* :

```
‘‘‘python
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import scipy.stats as st
import matplotlib.pyplot as plt
from pandas import Series, DataFrame
import pandas as pd
from sklearn.manifold import MDS
```

```
result =pd.read_table('dan04.txt',sep='\s+')
print(result)

X =DataFrame(result, columns=['P10','P11','P14'])
X=np.array (X)

SUMX=X[:,0] +X[:,1] + X[:,2]
X[:,0] =X[:,0] /SUMX
X[:,1] =X[:,1] /SUMX
X[:,2] =X[:,2] /SUMX



YT=result['P18']
y=np.array(YT)

XX1=result.query("P18 in [0]")
XX2=result.query("P18 in [1]")
XX3=result.query("P18 in [2]")

XX1 =DataFrame(XX1, columns=['P10','P11','P14'])
XX1=np.array (XX1)
XX2 =DataFrame(XX2, columns=['P10','P11','P14'])
XX2=np.array (XX2)
XX3 =DataFrame(XX3, columns=['P10','P11','P14'])
XX3=np.array (XX3)

SUMX1=XX1[:,0] +XX1[:,1] + XX1[:,2]
XX1[:,0] =XX1[:,0] /SUMX1
XX1[:,1] =XX1[:,1] /SUMX1
XX1[:,2] =XX1[:,2] /SUMX1

SUMX2=XX2[:,0] +XX2[:,1] + XX2[:,2]
XX2[:,0] =XX2[:,0] /SUMX2
XX2[:,1] =XX2[:,1] /SUMX2
XX2[:,2] =XX2[:,2] /SUMX2
```

```
SUMX3=XX3[:,0] +XX3[:,1] + XX3[:,2]
XX3[:,0] =XX3[:,0] /SUMX3
XX3[:,1] =XX3[:,1] /SUMX3
XX3[:,2] =XX3[:,2] /SUMX3

# Преобразование данных, уменьшающее размерность до 2
mod_mds = MDS(n_components=2, metric = True)
X_mds = mod_mds .fit_transform(X)
X_mds.shape


XI =DataFrame(X_mds, columns=['1','2'])
XI = XI.join(YT)

XI1=XI.query("P18 in [0]")
XI2=XI.query("P18 in [1]")
XI3=XI.query("P18 in [2]")

XI1 =DataFrame(XI1, columns=['1','2'])
XI1=np.array (XI1)
XI2 =DataFrame(XI2, columns=['1','2'])
XI2=np.array (XI2)
XI3 =DataFrame(XI3, columns=['1','2'])
XI3=np.array (XI3)

#построение исходного графика 3 D
fig = plt.figure(1, figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

xs1 = XX1[:,0]
ys1 = XX1[:,1]
zs1 = XX1[:,2]

xs2 = XX2[:,0]
ys2 = XX2[:,1]
zs2 = XX2[:,2]

xs3 = XX3[:,0]
```

```
ys3 = XX3[:,1]
zs3 = XX3[:,2]

ax.scatter(xs1, ys1, zs1,c='g', cmap=plt.cm.nipy_spectral,
edgecolor='k',s=80,label='A')
ax.scatter(xs2, ys2, zs2,c='r', cmap=plt.cm.nipy_spectral,
edgecolor='k',s=80,label='P')
ax.scatter(xs3, ys3, zs3,c='b', cmap=plt.cm.nipy_spectral,
edgecolor='k',s=80,label='D')
ax.set_title("Исходное пространство предикторов",fontsize=16,
fontweight='bold')
ax.set_xlabel('P10',fontsize=14, fontweight='bold')
ax.set_ylabel('P11',fontsize=14, fontweight='bold')
ax.set_zlabel('P14',fontsize=14, fontweight='bold')
ax.legend(loc="best")

y = np.choose(y, [1, 2, 0]).astype(np.float)

fig = plt.figure(2,figsize=(10, 8))
plt.clf()
plt.cla()

plt.scatter(XI1[:, 0], XI1[:, 1], c='g',
cmap=plt.cm.nipy_spectral, edgecolor='k',s=100,label='A')

plt.scatter(XI2[:, 0], XI2[:, 1], c='r',
cmap=plt.cm.nipy_spectral, edgecolor='k',s=100,label='P')

plt.scatter(XI3[:, 0], XI3[:, 1], c='b',
cmap=plt.cm.nipy_spectral, edgecolor='k',s=100,label='D')

plt.title("Снижение размерности методом MDS"
,fontsize=16, fontweight='bold')

plt.xlabel("MDS1",fontsize=14, fontweight='bold')

plt.ylabel("MDS2",fontsize=14, fontweight='bold')
plt.legend(loc='best')
```

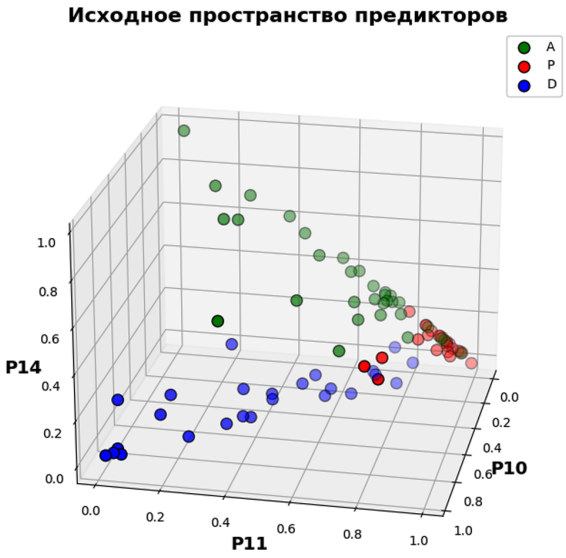

Рис. 1: Исходное 3D пространство описания регионов

```
plt.show()
```
‘‘‘

Графические результаты:

Как видно из графиков реализованный в sklearn.manifold модуль MDS при заданных параметрах в метрическом варианте гораздо лучше сохраняет топологию объектов, чем неметрический вариант многомерного масштабирования.

Код R :

‘‘‘r
```
library(gmodels)
library(lattice)
library(vegan)
library(grDevices)
library(reshape2)
```

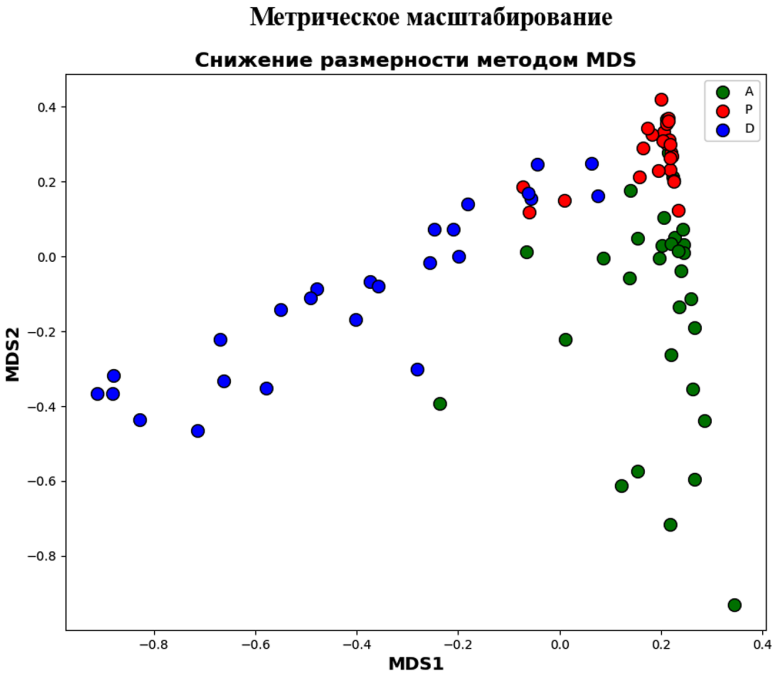

Рис. 2: Редуцированное 2D пространство mMDS

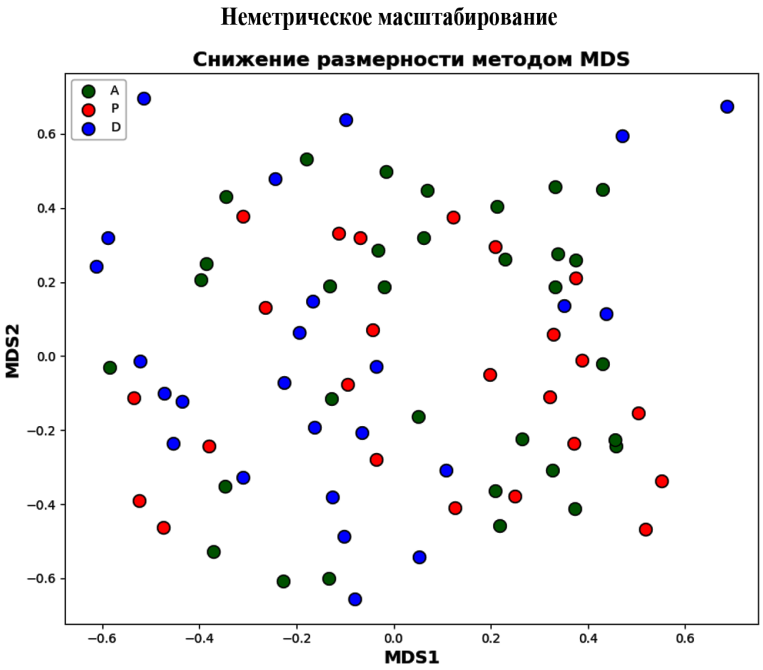

Рис. 3: Редуцированное 2D пространство nMDS

```
library(ggplot2)
library(ggthemes)
library(psych)
library(caret)

load("xRegion.RData")

#ФОРМИРОВАНИЕ data.frame ОСНОВНЫЕ
#СОЦИАЛЬНО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ РЕГИОНОВ РФ
xreg <-data.frame(P1=x$Pok_001,P2=x$Pok_002,P3=x$Pok_003,
P4=x$Pok_004,
P5=x$Pok_005,P6=x$Pok_006,P7=x$Pok_007,P8=x$Pok_008,
P9=x$Pok_009,P10=x$Pok_010,
P11=x$Pok_011,P12=x$Pok_012,P13=x$Pok_013,P14=x$Pok_014,
P15=x$Pok_015,P16=x$Pok_016,
P17=x$Pok_017,P18=x$Pok_018)
#xreg
#Формирование фактора тип экономики региона
xreg_f <- as.factor(x$Pok_018)

#выбор исследуемого подмножества показателей регионов
w=c(10,11,14,16)
reg.ekon <-xreg[,w]
valreg <-rowSums(reg.ekon)
xreg_nor <-reg.ekon/valreg
Y <- xreg_nor
X=Y
X <- as.matrix(X)

#Метод nMDS=======================================

#метрики "euc", "man", "bray", "jac", "kul"
x.mds = metaMDS(X, distance = "euc")
x.mds
#plot(x.mds)
data.scores = as.data.frame(scores(x.mds))
y.MDS <-data.frame(y1=data.scores[,1],y2=data.scores[,2],Econ=xreg_f)
```

```
#График регионов по классам в двухмерном пространстве NMDS
x11()
g4 <-ggplot(data= y.MDS, aes(x=y1,y=y2,colour = Econ),
colors = "Accent")
g4 <-g4 +geom_point(size=4)
g4 <-g4 +labs(title ="Регионы в пространсте двух компонент nMDS",
x="Компонента 1", y="Компонента  2",colour = "Классы")
g4 <-g4 + theme_fivethirtyeight() +
scale_colour_manual( values = c('darkgoldenrod4',
'blue','red','green'))
#g4 <-g4 + theme_stata() + scale_colour_stata()
g4

# mMDS=========================================

df.mds <- cmdscale(dist(X), eig = TRUE, k = 2)

df.mds$eig

#plot(df.mds$points[, 1], df.mds$points[, 2], col = xreg_f)

y2.MDS <-data.frame(y1=df.mds$points[, 1],y2=df.mds$points[, 2],Econ=xreg_f)

x11()
g4 <-ggplot(data= y2.MDS, aes(x=y1,y=y2,colour = Econ),
colors = "Accent")
g4 <-g4 +geom_point(size=4)
g4 <-g4 +labs(title ="Регионы в пространсте двух компонент mMDS",
x="Компонента 1", y="Компонента  2",colour = "Классы")
g4 <-g4 + theme_fivethirtyeight() +
scale_colour_manual( values = c('darkgoldenrod4',
'blue','red','green'))
#g4 <-g4 + theme_stata() + scale_colour_stata()
g4

```
```

Результаты работы скрипта:
```

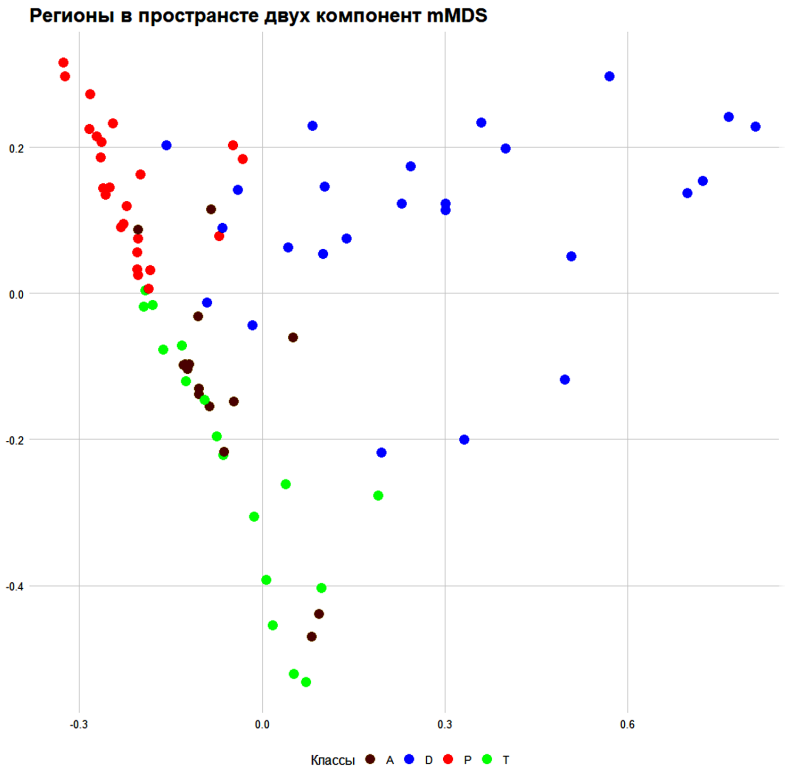

Рис. 4: Редуцированное 2D пространство mMDS

Собственные значения

```
5.594666e+00  3.431601e+00  4.873459e-01  7.099948e-16  3.665833e-16
3.070452e-16  3.003194e-16  2.319904e-16  1.603943e-16  1.397336e-16
1.298035e-16  1.220804e-16  1.173736e-16  1.128432e-16  9.612986e-17
8.106940e-17  7.306289e-17  6.340372e-17  6.186022e-17  5.229623e-17
5.140607e-17  4.800041e-17  4.519516e-17  4.333911e-17  4.005085e-17
3.899894e-17  3.493997e-17  2.595204e-17  2.565454e-17  2.552193e-17
2.544538e-17  2.204441e-17  2.026732e-17  1.588723e-17  1.560446e-17
1.514225e-17  1.458622e-17  1.231192e-17  1.587089e-18  1.538873e-18
7.924113e-19 -6.966414e-19 -2.326128e-18 -3.187938e-18 -3.493108e-18
-3.610405e-18 -5.591804e-18 -6.694794e-18 -7.343486e-18 -7.610262e-18
-7.692168e-18 -1.200938e-17 -1.346449e-17 -1.696788e-17 -1.821920e-17
-1.968668e-17 -2.774987e-17 -3.018549e-17 -3.385961e-17 -3.966822e-17
-5.018254e-17 -5.939872e-17 -6.230701e-17 -6.618591e-17 -6.664015e-17
-6.704644e-17 -7.042683e-17 -7.939269e-17 -8.706743e-17 -1.022151e-16
-1.599372e-16 -2.432393e-16 -2.744472e-16 -2.916891e-16 -3.940536e-16
-3.991335e-16 -4.352564e-16 -4.675905e-16 -5.431969e-16 -8.029666e-16
-1.331363e-15
```

Рис. 5: Собственные значения

В R метрический и неметрический методы реализованы в разных пакетах. Результаты работы метрического и неметрического методов практически совпадают. Здесь оба подхода к многомерному шкалированию хорошо сохраняют топологию регионов в редудицированном пространстве 2D.

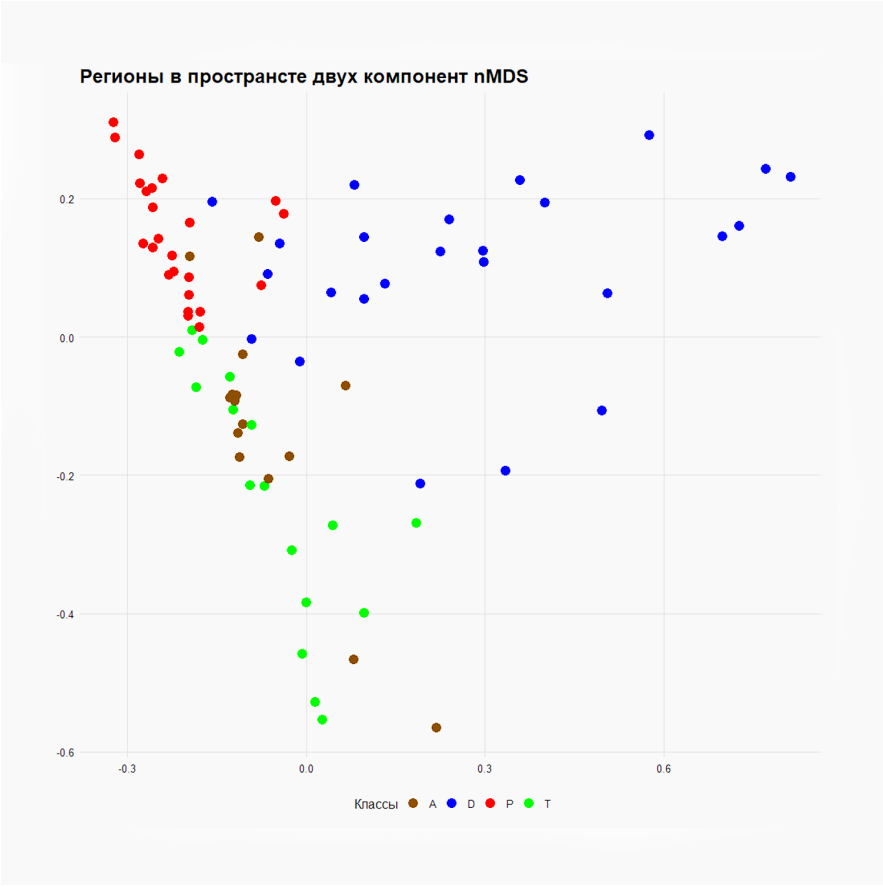

Рис. 6: Редуцированное 2D пространство nMDS