Метод стохастического вложения соседей (t-SNE)

Рассматриваемые ниже методы SNE (англ. Stochastic Neighbor Embedding) и $t-SNE$ (англ. t-Distributed Stochastic Neighbor Embedding), базируются на тех же исходных данных (матрица предикторов X) и том же формальном представлении этих данных, что и метод PCA. При этом измерительной основой также являются метрические шкалы.

Формальные основания алгоритма

Классический метод SNE начинается с преобразования многомерной евклидовой дистанции между точками в условные вероятности, отражающие сходство точек. Математически это выглядит следующим образом (1):

$$p_{j|i} = \frac{\exp(-\|X_i - X_j\| / 2 \cdot \sigma_i^2)}{\sum_{k \neq i} \exp(-\|X_i - X_k\| / 2 \cdot \sigma_i^2)} \quad (1)$$

Эта формула показывает, насколько точка X_j (j-ая строка матрицы предикторов) близка к точке X_i (i-ая строка матрицы предикторов) при гауссовом распределении вокруг X_i с заданным отклонением σ. Сигма будет различной для каждой точки. Она выбирается так, чтобы точки в областях с большей плотностью имели меньшую дисперсию. Для этого используется оценка перплексии:

$$Petp(P_i) = 2^{H(P_i)}$$

где $H(P_i)$ -энтропия Шеннона в битах (2):

$$H(P_i) = -\sum_j p_{j|i} \cdot \log_2 p_{j|i} \circ \quad (2)$$

В данном случае перплексия может быть интерпретирована как сглаженная оценка эффективного количества «соседей» для точки X_i. Она задается в качестве параметра метода. Авторы рекомендуют использовать значение в интервале от 5 до 50. Сигма определяется для каждой пары X_i и X_j при помощи алгоритма бинарного поиска.

Для двумерных или трехмерных «коллег» пары X_i и X_j, назовем их для ясности X_i^{dr} и X_j^{dr}, не представляет труда оценить условную вероятность, используя ту же формулу 1. Стандартное отклонение предлагается установить в $\frac{1}{\sqrt{2}}$:

$$q_{j|i} = \frac{\exp(-\left\|X_i^{dr} - X_j^{dr}\right\|^2)}{\sum_{k \neq i} \exp(-\left\|X_i^{dr} - X_k^{dr}\right\|^2)} \quad (3)$$

Если точки отображения X_i^{dr} и X_j^{dr} корректно моделируют сходство между исходными точками высокой размерности X_i и X_j, то соответствующие условные вероятности $p_{j|i}$ и $q_{j|i}$ будут эквивалентны. В качестве очевидной оценки качества, с которым $q_{j|i}$ отражает $p_{j|i}$, используется дивергенция или расстояние Кульбака-Лейблера. SNE минимизирует сумму таких расстояний для всех точек отображения при помощи градиентного спуска. Функция потерь для данного метода будет определяться формулой 4:

$$Cost = \sum_i KL\left(P_i \,||\, Q_i\right) = \sum_i \sum_j p_{j|i} \cdot \log \frac{p_{j|i}}{q_{j|i}} \quad (4)$$

При этом градиент выглядит на удивление просто (5):

$$\frac{\partial Cost}{\partial X_i^{dr}} = 2 \cdot \sum_j \left(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j}\right)\left(X_i^{dr} - X_j^{dr}\right) \quad (5)$$

Авторы предлагают следующую физическую аналогию для процесса оптимизации: Все точки отображения соединены пружинами. Жесткость пружины, соединяющей точки i и j зависит от разности между сходством двух точек в многомерном пространстве и двух точек в пространстве отображения. В этой аналогии, градиент — это результирующая сила, действующая на точку в пространстве отображения. Если систему «отпустить», через какое-то время она придет в равновесие, это и будет искомое распределение. Алгоритмически, поиск равновесия предлагается делать с учетом моментов:

$$X_t^{dr} = X_{t-1}^{dr} + \eta \cdot \frac{\partial Cost}{\partial X^{dr}} + \alpha(t) \cdot \left(X_{t-1}^{dr} - X_{t-2}^{dr}\right) \quad (6)$$

где – η параметр, определяющий скорость обучения (длину шага), а α – коэффициент инерции. Использование классического SNE позволяет получить неплохие результаты, но может быть связано с трудностями в оптимизации функции потерь и проблемой скученности (в оригинале – crowding problem). $t-SNE$ если и не решает эти проблемы совсем, то существенно облегчает. Функция потерь t-SNE имеет два принципиальных отличая:

- Во-первых, у $t-SNE$ симметричная форма сходства в многомерном пространстве и более простой вариант градиента.

- Во-вторых, вместо гауссова распределения для точек из пространства отображения используется t-распределение (Стьюдента), «тяжелые» хвосты которого облегчают оптимизацию и решают проблему скученности.

В качестве альтернативы минимизации суммы дивергенций Кульбака-Лейблера между условными вероятностями $p_{i|j}$ и $q_{i|j}$ предлагается минимизировать одиночную дивергенцию между совместной вероятностью P в многомерном пространстве и совместной вероятностью Q в пространстве отображения:

$$Cost = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (7),$$

где p_{ii} и $q_{ii} = 0$, $p_{ij} = p_{ji}$, $q_{ij} = q_{ji}$ для любых i и j, а p_{ij} определяется по формуле:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \quad (8),$$

где n — количество точек в наборе данных. Градиент для симметричного SNE получается существенно проще, чем для классического:

$$\frac{\partial Cost}{\partial X_i^{dr}} = 4 \cdot \sum_j (p_{ij} - q_{ij}) (X_i^{dr} - X_j^{dr}) \quad (9)$$

Проблема скученности заключается в том, что расстояние между двумя точками в пространстве отображения, соответствующими двум среднеудаленным точкам в многомерном пространстве, должно быть существенно больше, нежели расстояние, которое позволяет получить гауссово распределение. Проблему решают хвосты Стьюдента. В t-SNE используется t-распределение с одной степенью свободы. Совместная вероятность для пространства отображения в этом случае будет определяться формулой 10:

$$q_{ij} = \frac{(1 + || X_i^{dr} - X_j^{dr} ||^2)^{-1}}{\sum_{k \neq l} (1 + || X_k^{dr} - X_l^{dr} ||^2)^{-1}} \quad (10)$$

А соответствующий градиент – выражением 11:

$$\frac{\partial Cost}{\partial X_i^{dr}} = 4 \cdot \sum_j (p_{ij} - q_{ij}) (X_i^{dr} - X_j^{dr}) (1 + || X_i^{dr} - X_j^{dr} ||^2)^{-1} \quad (11)$$

Возвращаясь к физической аналогии, результирующая сила, определяемая формулой 11, будет существенным образом стягивать точки пространства отображения для близлежащих точек многомерного пространства, и отталкивать — для удаленных.

Алгоритм

Алгоритм $t - SNE$ в упрощенном виде можно представить следующим псевдокодом:

Data: набор данных $X = (x_1, x_2, \ldots, x_n)$, параметр функции потерь: перплексия *Perp*, Параметры оптимизации: количество итераций T, скорость обучения η, момент $\alpha(t)$. *Result*: представление данных $X^{dr}(T) = (X_1^{dr}, X_2^{dr}, \ldots, X_n^{dr})$ (в $2D$ или $3D$).

begin

вычислить попарное сходство $p_{j|i}$ с перплексией *Perp* (используя формулу 1)

установить $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$

инициализировать $X^{dr}(0) = (X_1^{dr}, X_2^{dr} ...., X_n^{dr})$ точками нормального распределения ($mean = 0$, $sd = 1e - 4$

for $t = 1$ to T do

вычислить сходство точек в пространстве отображения q_{ij} (по формуле 10)

вычислить градиент (по формуле 11) установить $X^{dr}(t)$ (по формуле 6)

end

end

Пример реализации алгоритма $t - SNE$ на *Python* и *R*.

В качестве исходных данных используем информацию об основных социально-экономических показателях российских регионов за 2018 год. Исходная матрица предикторов имеет размерность 81x16 в качестве выходной переменной используется номинальная переменная тип экономики региона. В реализации на Python переменной "тип экономики"присваиваются значения: A-аграрный; D-добывающий, P- промышленный; а вот в реализации на R добовляется значение: T-торговый.

Код на Python

Здесь из исходной таблицы выбираются три столбца P10,P11,P14 , а выходная переменная P18 принимает три значения A,D,P

```
    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 TSNE


result =pd.read_table('dan04.txt',sep='\s+')
print(result)
X =DataFrame(result, columns=['P10','P11','P14'])
X=np.array (X)

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

#описательные статистики
print('Описательные статистики предикторов')
for num in range(3) :
  print('Номер переменной: %.0f' %num)
  print('число элентов массива =',len(X[:,num]))
  print('среднее значение =',np.mean(X[:,num]))
  print('стандартное отклонение =',np.std(X[:,num]))
  print('мин мах =',np.min(X[:,num]), np.max(X[:,num]))

 # Преобразование данных, уменьшающее размерность до 2
Mod_t = TSNE(n_components=2, perplexity = 18.0, early_exaggeration = 15.0)
X_t =Mod_t.fit_transform(X)
X_t.shape
XT =DataFrame(X_t, columns=['1','2'])
XT = XT.join(YT)

XT1=XT.query("P18 in [0]")
XT2=XT.query("P18 in [1]")
XT3=XT.query("P18 in [2]")

XT1 =DataFrame(XT1, columns=['1','2'])
XT1=np.array (XT1)
XT2 =DataFrame(XT2, columns=['1','2'])
XT2=np.array (XT2)
XT3 =DataFrame(XT3, columns=['1','2'])
XT3=np.array (XT3)

#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")
ax.set_xlabel('P10')
ax.set_ylabel('P11')
ax.set_zlabel('P14')
ax.legend(loc='upper left')
y = np.choose(y, [1, 2, 0]).astype(np.float)
fig = plt.figure(2,figsize=(10, 8))
plt.clf()
plt.cla()

plt.scatter(XT1[:, 0], XT1[:, 1], c='g', cmap=plt.cm.nipy_spectral,
 edgecolor='k',s=100,label='A')
plt.scatter(XT2[:, 0], XT2[:, 1], c='r', cmap=plt.cm.nipy_spectral,
 edgecolor='k',s=100,label='P')
plt.scatter(XT3[:, 0], XT3[:, 1], c='b', cmap=plt.cm.nipy_spectral,
 edgecolor='k',s=100,label='D')
```

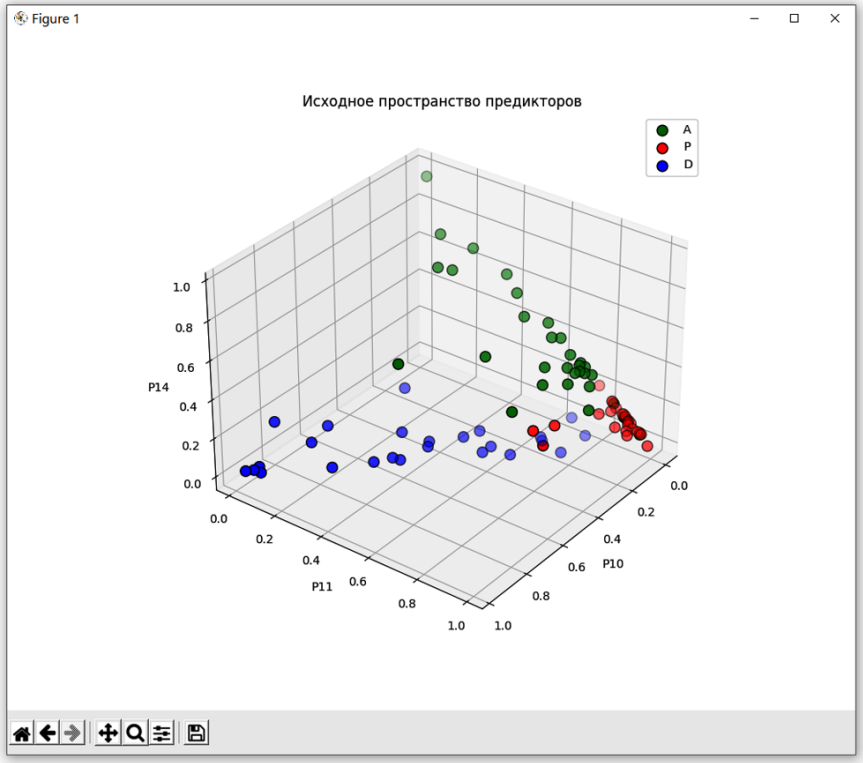

Рис. 1: Данные в исходном трехмерном пространстве

```
plt.title("Снижение размерности методом t-SNE")
plt.xlabel("SNE1")
plt.ylabel("SNE2")
plt.legend(loc='upper left')
plt.show()
```

Графические результаты представлены ниже

Код R

Здесь из исходной таблицы выбираются шестнадцать столбцов P1-P16 , а выходная переменная P18 принимает четыре значения A,D,P,T

```r
library(ggplot2)
library(ggthemes)
rm(list=ls(all=TRUE))
```

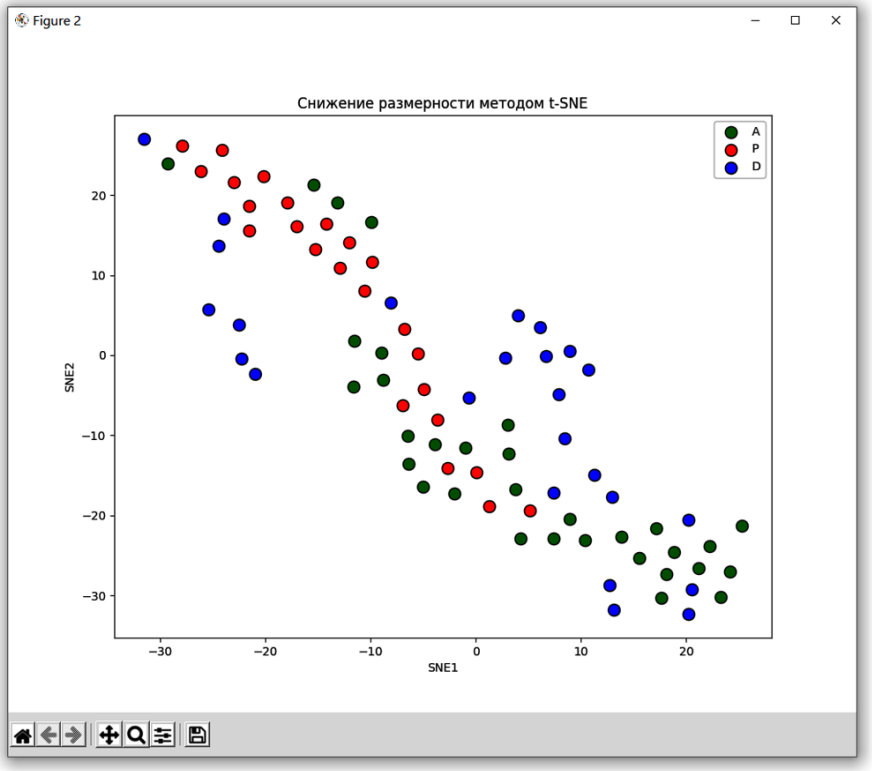

Рис. 2: Данные в редуцированном двухмерном пространстве

```
load("xRegion.RData")
x[,1:2]
#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, ind =x$IndexReg)
#xreg
#Формирование фактора тип экономики региона
xreg_f <- as.factor(x$Pok_018)
ind_f <-as.factor(x$IndexReg)

#ИСХОДНОЕ МНОЖЕСТВО ОСНОВНЫХ
 СОЦИАЛЬНО-ЭКОНОМИЧЕСКИХ ПОКАЗАТЕЛЕЙ
  РЕГИОНОВ РФ 2018 г.
#P1  - Площадь территории,тыс. км2
#P2  - Численность  населения на 1 января 2019 г., тыс. человек
#P3  - Среднегодовая численность  занятых, тыс. человек
#P4  - Среднедушевые денежные доходы (в месяц), руб
#P5  - Потребительские расходы в среднем на душу
населения (в месяц), руб
#P6  - Среднемесячная номинальная начисленная з
аработная плата работников организаций, руб.
#P7  - Валовой  региональный  продукт в 2018 г. , млн руб
#P8  - Инвестиции в основной  капитал, млн руб
#P9  - Основные фонды в экономике (по полной учетной
 стоимости; на конец года),млн руб.
#P10 - Добыча  полезных  ископаемых ,млн. руб
#P11 - Обрабатывающие производства, млн руб
#P12 - Обеспечение электрической энергией, газом
 и паром, млн руб
#P13 - Водоснабжение; водоотведение, организация
сбора и утилизации отходов, млн руб
```

```
#P14 - Продукция сельского хозяйства , млн руб
#P15 - Ввод  в действие  жилых домов, тыс. м2
#P16 - Оборот  розничной торговли, млн руб
#P17 - Сальдированный финансовый результат
 деятельности организаций, млн руб.
#P18 - Тип региональной экономики (A,D,P,T)
#ind - номер региона

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

#СНИЖЕНИЕ РАЗМЕРНОСТИ
#Метод t-SNE================================
library(tsne)
X=Y
X <- as.matrix(X)
xstr_nor.tsne <-tsne(X, initial_config = NULL, k =2, initial_dims = 16,
perplexity = 10, max_iter = 1000, min_cost = 0,
epoch_callback = NULL, whiten = FALSE, epoch=100)
summary(xstr_nor.tsne)

xreg_tsne <-data.frame(x1=xstr_nor.tsne[,1],x2=xstr_nor.tsne[,2])
xreg_tsne <-cbind(xreg_tsne,xreg_f)

#График регионов по классам в двухмерном пространстве t-SNE
x11()
g4 <-ggplot(data= xreg_tsne, aes(x=x1,y=x2,colour = xreg_f),
colors = "Accent")
g4 <-g4 +geom_point(size=4)
g4 <-g4 +labs(title ="Регионы в пространсте двух компонент t-SNE 1",
x="Компонента t-SNE 1", y="Компонента t-SNE 2",colour = "Классы")
#g4 <-g4 + theme_fivethirtyeight() +  scale_color_fivethirtyeight()
#g4 <-g4 + theme_stata() + scale_colour_stata()
```

```
g4

#Метод  Rtsne ----------------------------------------------------------------
library(Rtsne)
tsne_out <- Rtsne(X,dims=2,pca=TRUE,perplexity=8,theta=0.0)
tsne_out$Y

y.Rtsne <-data.frame(y1=tsne_out$Y[,1],y2=tsne_out$Y[,2])
y.Rtsne <-cbind(y.Rtsne,xreg_f)

#График регионов по классам в двухмерном пространстве Rtsne
x11()
g4 <-ggplot(data= y.Rtsne, aes(x=y1,y=y2,colour = xreg_f),
colors = "Accent")
g4 <-g4 +geom_point(size=4)
g4 <-g4 +labs(title ="Регионы в пространсте двух компонент t-SNE 2",
x="Компонента t-SNE 1", y="Компонента t-SNE 2",colour = "Классы")
#g4 <-g4 + theme_fivethirtyeight() +  scale_color_fivethirtyeight()
#g4 <-g4 + theme_stata() + scale_colour_stata()
g4
```

Графические результаты представлены ниже

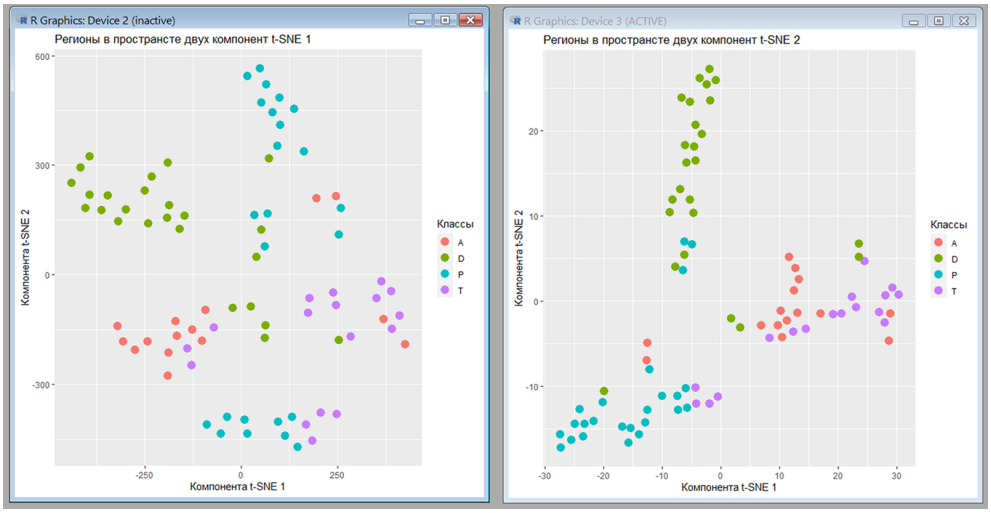

Рис. 3: Данные в редуцированном двухмерном пространстве