NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science
Master's Program 'Data Science'

Report: Exploring Personal Medical Cost

Course: Modern Methods for Data Analysis

By Anna Beketova,
Anatolii Burtsev,
Eric George Parakal

2018

Table of content

Homework 1 - Introduction

It is crucial for insurance companies to be able to reliably predict with reasonable accuracy, the amount of medical expenses for their insured customers. Based on this information a medical insurance company may be able to decide how much its insurance certificate should cost and predict the amount of money it would make by selling each certificate

Since, medical expenses can vary a lot depending on person's age, sex and overall health condition (whether a person is overweight or underweight, is a smoker or not, in what region he/she lives, how many children he/she has and various other factors). However, there are some conditions that are more prevalent for certain segments of the population. For example, heart diseases are more likely to appear when a person is overweight as compared to someone who is not, smokers have higher probability of suffering from lung diseases (e.g lung cancer, oral cancer etc.) as opposed to non-smokers.

The aim of this research is to use patient data to discover some features, trends and possible correlations thereby allowing us to estimate population specifics leading to us being able to cluster people using their correlated characteristics and be able to forecast their respective medical care expenses.

Content

The file includes 1338 examples of beneficiaries currently enrolled in the insurance plan, with features indicating characteristics of the patient as well as the total medical expenses charged to the plan for the calendar year.

Data may be found via link https://www.kaggle.com/mirichoi0218/insurance .

Columns:

Table 1 "Dataset description"

Column name	Description
Age	An integer indicating the age of the primary beneficiary (excluding those above 64 years, since they are generally covered by the government)
Sex	The policy holder's gender, either 'male' or 'female'
BMI	The body mass index (BMI), which provides a sense of how over- or under-weight a person is relative to their height. BMI is equal to weight (in kilograms) divided by height (in meters) squared. An ideal BMI is within the range of 18.5 to 24.9

Column name	Description
Children	An integer indicating the number of children/dependents covered by the insurance plan
Smoker	A ‘yes’ or ‘no’ categorical variable that indicates whether the insured regularly smokes tobacco
Region	The beneficiary's place of residence in the US, divided into four geographic regions: northeast, southeast, southwest, or northwest
Charges	Individual medical costs billed by health insurance

Main descriptive statistics are presented below. The table shows that there are no missing values, variables have various scales and variations. So latter in this report scaling using range and std is going to be implemented.

Table 2 “Dataset statistics”

Statistic	Age	BMI	Children	Charges
count	1338	1338	1338	1338
mean	39,21	30,66	1,09	13 270,42
std	14,05	6,10	1,21	12 110,01
min	18,00	15,96	0,00	1 121,87
25%	27,00	26,30	0,00	4 740,29
50%	39,00	30,40	1,00	9 382,03
75%	51,00	34,69	2,00	16 639,91
max	64,00	53,13	5,00	63 770,43

Topics covered in the report

There are several tasks that we are going to accomplish in this work:

- clustering people with several characteristics into groups according to their “risk rate” of being highly charged (collective factor – to be calculated later), age, sex, BMI, the number of children, whether he/she smokes or not, in what region he/she lives;
- evaluating the quality of clustering by computing confidence intervals for mean just as is and using bootstrap (both pivotal and non-pivotal versions);
- exploring categorical features by contingency tables. Conditional frequency table, Quetelet relative index table and chi-square- summary Quetelet index for both are discussed;
- finding hidden factors using Principal Component Analysis and Singular Value Decomposition and visualizing dataset not in high dimensional space but in lower one;
- predicting the amount of money that would be charged from that particular person according to person’s age via linear regression. Evaluating the model quality.

Materials

The research is conducted according to homeworks of the Master's Course "Modern Methods for Data Analysis" taught by Boris Mirkin, PhD (Mathematics), DSc (Engineering), Professor of the Department of Data Analysis and Artificial Intelligence FCS NRU HSE Moscow, Emeritus Professor of the University of London.

The main literature:

1) Course's lectures;
2) Mirkin, B. (2011). Core concepts in data analysis: summarization, correlation and visualization. Springer Science & Business Media;
3) Some reliable web sources (https://wikipedia.org/, http://www.machinelearning.ru/).

Homework 2 – K-means and confidence intervals

K-means algorithm

K-Means clustering is a clustering method that groups together similar data into a set $S_k(k = 1,2\ldots K)$ together using 3 simple steps that begin after we decide the number of clusters $K$and the centers of each cluster $c_k(k = 1,2\ldots K)$

(i) Update the sets $S_k(k = 1,2\ldots K)$using the minimum distance rule
(ii) Update the centers $c_k(k = 1,2\ldots K)$as means of S_k
(iii) If new centers coincide with the previous ones, stop. Else go to step (i)

The minimum distance, also called the Euclidian distance can be defined as the sum of squares of differences for each feature between the center c_k and the object i as shown below

$d(i, c_k) = \sum_{v \in V}(i_v - c_{kv})^2$ where $V$is the set of all features in the dataset

It is also advisable to always try to minimise the K-Means criterion in order to get the ideal number of clusters K

Whose equation is given by $D(S, c) = \sum_{k=1}^{K} \sum_{i \in S_k} d\,(i, c_k)$

Choosing random state

The K-means cluster partition depends highly on the first initialisations of cluster centers. Authors choose the best random initialization of K-means by comparing sum of squared distances (SSD) of samples to their closest cluster center out of 10 random states.

Table 3. Finding the best random state

Random initialization	1	2	3	4	5
Random state	374	75	132	90	746
SSD for 5 clusters	118,9873	**118,9866**	118,9889	118,9885	118,9867
SSD for 9 clusters	76,4779	76,2556	76,2519	**75,9893**	76,8290
Random initialization	6	7	8	9	10
Random state	590	763	433	106	851
SSD for 5 clusters	118,9873	118,9880	118,9882	118,9873	118,9876
SSD for 9 clusters	76,4944	76,5636	75,9911	76,5544	75,9925

The better partition is considered to have less SSD. By choosing so distances from each observation to its corresponding cluster center are minimal.

Interpretation of partitions

The result of K-means clustering presented above. The data is sorted by sex, age, bmi, children, smoker to show more intuitive interpretation.

Table 4. Means of every variable in 5 cluster case

cluster	age	sex	bmi	children	smoker
4	26,42	0	29,61	0,90	0,18
2	50,89	0	31,04	1,23	0,17
3	25,39	1	30,14	0,50	0,25
1	39,89	1	31,34	2,72	0,26
0	53,11	1	31,51	0,45	0,20

People are divided into:

1) Young women, on average with lower weight, few children, less smoking;
2) Older women with average weight having at least on child, less smoking;
3) Young men with average weight the least number of children who smoke more than an average person of this dataset;
4) Middle-aged men, on average with higher weight, at least 2 children who smoke more frequent than other people of this dataset;
5) Older men with the highest weight minimum number of children and who are the average smokers.

Table 5. Means of every variable in 9 cluster case

cluster	age	sex	bmi	children	smoker
1	25,74	0	29,69	0,32	0,18
2	30,76	0	29,18	2,61	0,19
3	51,45	0	32,11	2,53	0,15
4	51,95	0	30,82	0,39	0,17
5	22,18	1	30,29	0,23	0,23
6	29,23	1	30,50	2,67	0,26
7	38,88	1	30,25	0,66	0,26
8	48,24	1	31,54	2,56	0,28
9	56,14	1	32,01	0,29	0,17

This way 9 clusters also could be interpretable. Still, having more clusters narrowing the description, but variable clusters means coincide a lot.

The more distant cluster means are the better partition it is. So **the division into 5 clusters are more appropriate** as it shows more general but still specific division.

Bootstrap approach explanation

In case when we have too small sample we could use bootstrap to get more information about original distribution. We have sample X_origin and we pick N times random element from this sample. The new sample of these N elements could be use to estimate some statistics of origin distribution.

There are 2 methods to estimate mean and confidence interval in bootstrapped sample:
1) Pivotal method: We don't take into account the fact that sample is bootstrapped. Mean is ordinary mean. 95% confidence interval = Mean +- 1.96 Std.
2) Non-pivotal method: we sort this sample X by value. Median element in this sorted array - is mean, and confidence interval's borders are 0.025 * N-th and 0.975*N-th elements. (if N = 1000, 95% confidence interval is (25-th, 975-th) elements).

We take 2 of our 5 clusters and calculate mean and confidence intervals for them.

Feature selection

Clustering into 5 groups shows better division by age variable. It is presented below:

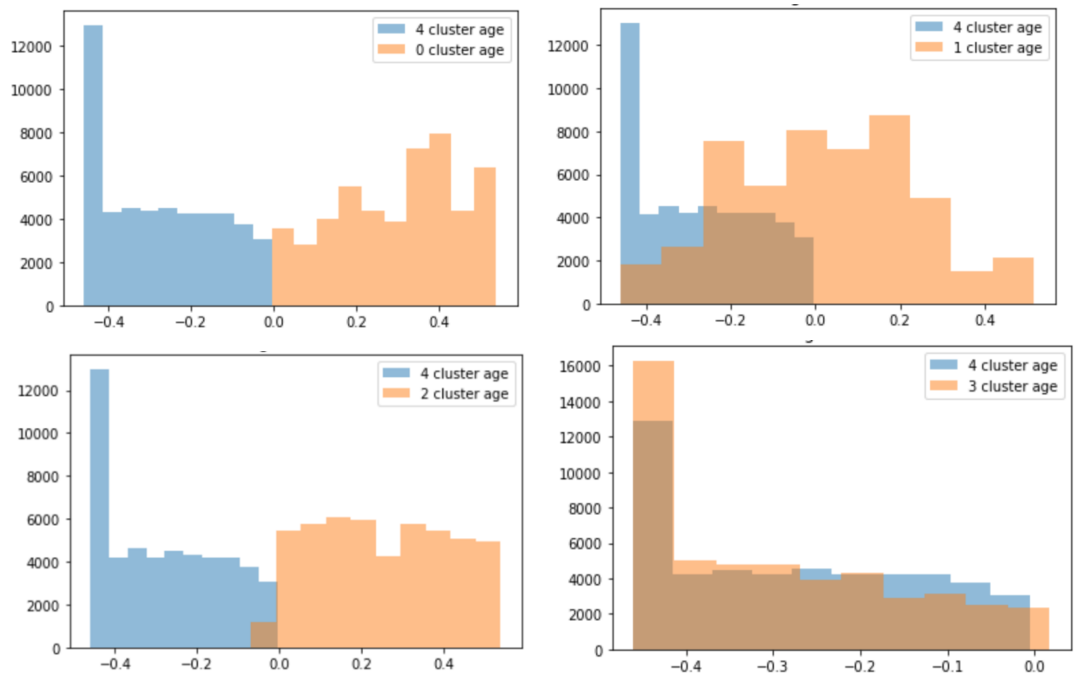


Figure 1. Histogram of age for different clusters in clustering by 5 groups.

Figure shows that clusters 0 and 4 are **lineary separable**. And the separation treshold is -0.0045 or 39 years old in the initial insurance dataset.

We will use below clusters 0 and 4 to show results of mean comparison (including pivotal and non-pivotal bootstrap calculation).

Feature mean comparison

Here are 2 tables with statistics calculated:

Table 6. Pivotal method.

Cluster	Mean	95% conf interval left	95% conf interval right
1	0,302	0,007	0,598
2	-0,277	-0,545	-0,01

Table 7. Non-pivotal method.

Cluster	Mean	95% conf interval left	95% conf interval right
1	0,322	0,017	0,539
2	-0,287	-0,461	-0,026

We can see 2 things:

1. confidence intervals **don't intersect**;
2. mean and confidence interval's borders almost the same for pivotal and non-pivotal methods.

Then we calculate mean values for 1-st cluster and for all the rest data within 1-st cluster. Results are written in the following table:

Table 8. Comparison

	Pivotal grand mean	Non-pivotal grand mean
Within cluster 1	0,302	0,322
All other data	0	-0,005

Mean values are almost the same for both methods. It is clear that first cluster is more distant, it lies not in the middle of the sample given, **highly distinguishable** from other data.

Homework 3 – Contingency Table

In order to receive some insights from categorical data contingency tables may be used. A contingency table has its rows corresponding to categories of one feature and columns to categories of the other feature filled with the count of occurrence in exact categories.

Categorical feature construction

Authors took 1 exact feature from given dataset. Two other features are calculated according to their histograms. Groups were taken when clearly separable.

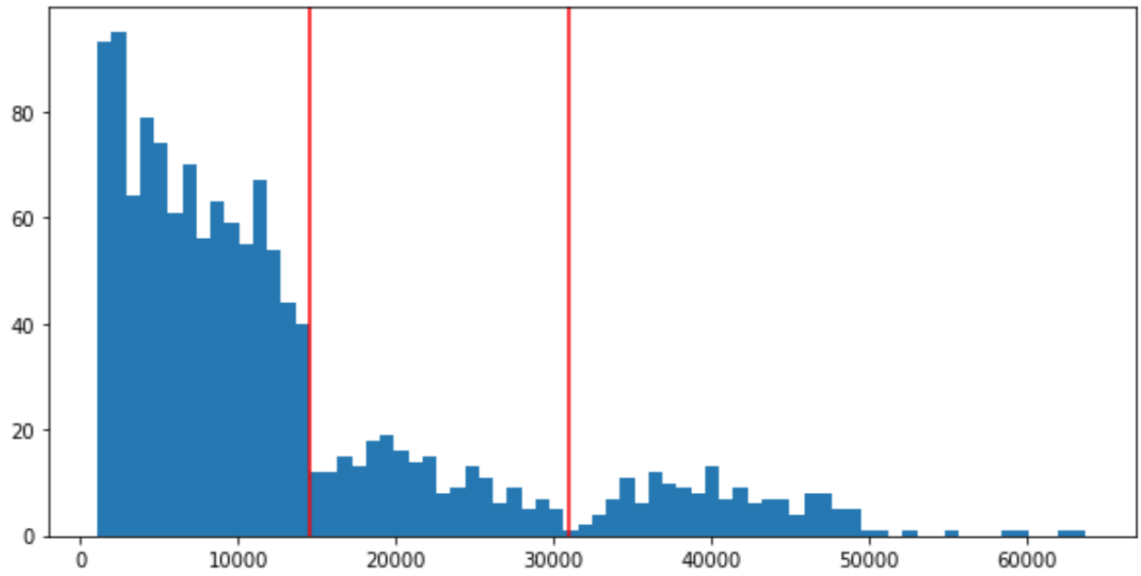


Figure 2. Histogram of charge with the division into 3 clusters.

The histogram of **charges** depicts that there could be 3 main groups:

1) below $14500 paid;
2) between $14500 and $31000 paid;
3) abode $31000 paid.

Second feature is from initial dataset – **children**. It is crucial to investigate some relations on the amount of money spent on the insurance bills depending on how many children a person has. Consideration of this variable reflects whether a person with more children tends to pay more.

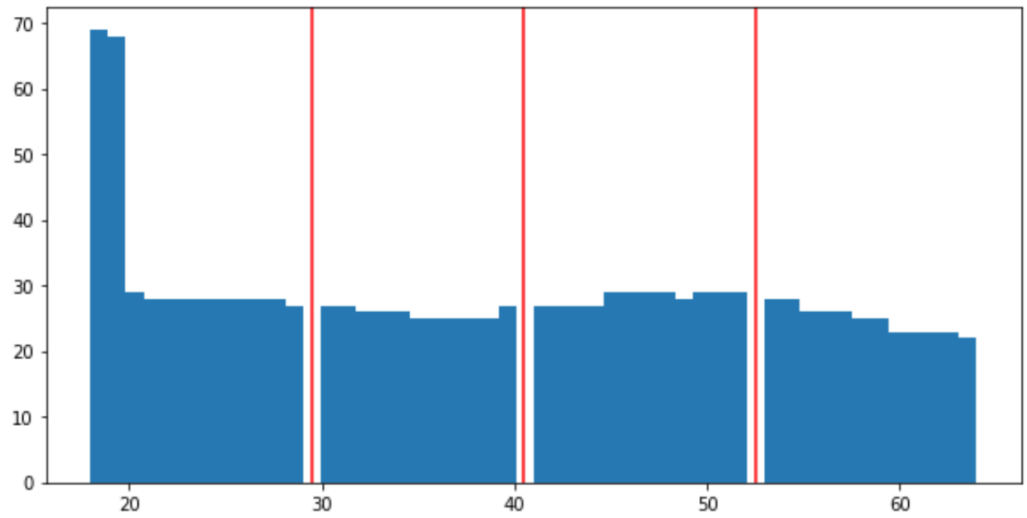


Figure 3. Histogram of age with the division into 4 clusters.

The **age** histogram provides with the division into the following groups:
1) younger than 29;
2) between 29 and 40;
3) between 40 and 52;
4) older than 52.

These are the categorical variables that we are comparing latter in this chapter.

Contingency (co-occurrence) table

This kind of tables represent how many people are in 2 categories. For instance, there were 310 people who paid less than $14500 and who are younger than 29 simultaneously.

Table 9. Contingency table of age and charges categories.

Charges category, th $	Age category				Total
	(.., 29]	(29, 40]	(40, 52]	(52, ..)	
(.. , 14.5]	310	210	245	209	974
(14.5, 31]	61	42	52	53	208
(31, ..)	46	32	42	36	156
Total	417	284	339	298	1338

Table 10. Contingency table of children and charges categories.

Charges category	Children						Total
	0	1	2	3	4	5	
(.. , 14.5]	432	242	163	103	17	17	974
(14.5, 31]	63	33	38	20	2	0	156
(31, ..)	79	49	39	34	6	1	208
Total	574	324	240	157	25	18	1338

From these tables other specific tables may be derivable. And then from new tables some dependencies may be found.

Relative contingency (co-occurrence) table

In order to build relative contingency table one should normalize contingency table by the number of all observations. It shows shares of the sample that is in the intersection of these categories.

Table 11. Relative contingency table of age and charges categories.

Charges category	Age category				Total
	(.., 29]	(29, 40]	(40, 52]	(52, ..)	
(.. , 14.5]	0,232	0,157	0,183	0,156	0,728
(14.5, 31]	0,046	0,031	0,039	0,04	0,155
(31, ..)	0,034	0,024	0,031	0,027	0,117
Total	0,312	0,212	0,253	0,223	1

Relative contingency (co-occurrence) table of age and charges categories shows that according to this dataset the probability 0.728 of people paid less than $14500 is much higher than to other charges categories (0.155 for charges between 14.5 and 31 and 0.117 for charges over 31). So no matter what the age is more people pay less than $14500.

Table 12. Relative contingency table of children and charges categories.

Charges category	Children						Total
	0	1	2	3	4	5	
(.. , 14.5]	**0,323**	0,181	0,122	0,077	0,013	0,013	0,728
(14.5, 31]	0,047	0,025	0,028	0,015	0,001	0,000	**0,117**
(31, ..)	0,059	0,037	0,029	0,025	0,004	0,001	**0,155**
Total	0,429	0,242	0,179	0,117	0,019	0,013	1,000

In the dataset there are more people (32,3% of sample) with no children who spend less than $14500 on insurance. For any number of children there is a bigger share of people with insurance bills greater than $31000 (11,7%) in comparison with insurance bills between $14500 and $31000 (15,5%).

Quetelet relative index table

Quetelet Index is a metric that is used to measure correlation patterns in a dataset among 2 row and column categories i.e how dependent one is on the other. It helps us to measure the strength of the association as compared to the average case.

Quetelet Tables are usually used in conjunction with contingency tables or conditional frequency tables. The formula for computing the Quetelet Index of a cell is given as

$$Q_{kv} = (P(k|v) - P_v)/P_v$$

where:

- $P(k|v)$is the probability of belonging to cluster or group k, given that the entity has variable or feature v;
- $P_v$is the probabiltiy of feature $v$in the dataset.

This formula can further be simplified to give $Q_{kv} = P_{kv}/(P_k P_v - 1)$where $P_{kv}$is the probabiltiy of belonging to cluster or group k and that the entity has variable or feature v.

When Quetelet Indices are positive, there is a strong increase in average probabilty of an entity belonging to a cluster or group subject to having a particular feature, the opposite is true when the Quetelet Index is negative.

Table 13. Quetelet relative index table for age and charges categories.

Charges category	Age category			
	(.., 29]	(29, 40]	(40, 52]	(52, ..)
(.. , 14.5]	49,18	24,76	-13,18	-57,10
(14.5, 31]	-18,52	-8,03	19,66	9,72
(31, ..)	-26,90	-15,28	-5,16	57,07

Some charges patterns could be derived from the table above. For instance, people younger than 29 spend money in amount over $31000 26,9% less often than on average.

Table 14. Quetelet relative index table for children and charges categories.

Charges category	Number of children					
	0	1	2	3	4	5
(.. , 14.5]	3,39	2,60	-6,70	-9,88	-6,59	29,74
(14.5, 31]	-5,86	-12,64	35,80	9,26	-31,38	-100,00
(31, ..)	-11,47	-2,72	4,53	39,31	**54,38**	-64,26

So knowing the number of children we could find out several charges patterns. For example, people with 4 children spend much insurance money 54% more often than average.

Summary Quetelet index

Summary Quetelet index - this is inner product of relative co-occurrence matrix and Quetelet index.

Table 15. Summary Quetelet index of age, charges case.

Charges category	Age category				Total
	(.., 29]	(29, 40]	(40, 52]	(52, ..)	
(.. , 14.5]	2,12	1,58	-0,72	-3,66	-0,67
(14.5, 31]	-5,39	-3,36	6,26	3,61	1,13
(31, ..)	-5,90	-4,87	-1,33	14,41	2,31
Total	-9,16	-6,65	4,22	14,37	2,77

On average knowledge of the age group “adds” 16,22% to frequency of insurance bills.

Table 16. Summary Quetelet index of children, charges case.

Charges category	Children						Total
	0	1	2	3	4	5	
(.. , 14.5]	1,094	0,471	-0,816	-0,760	-0,084	0,378	0,282
(14.5, 31]	-0,276	-0,312	1,017	0,138	-0,047	0,000	0,520
(31, ..)	-0,677	-0,099	0,132	0,999	0,244	-0,048	0,550
Total	0,141	0,060	0,332	0,377	0,113	0,330	1,353

It means that on average knowledge of the number of children “adds” 135.3% to frequency of insurance bills.

Sufficiency for confidence levels

Pearson’s chi-squared score is the following:

$$\boldsymbol{X^2 = \sum_{k=1}^{K} \sum_{l=1}^{L} \frac{\big(p(Hk \cap Gl) - p(Hk)p(Gl)\big)^2}{p(Hk)p(Gl)}}$$

Data sufficiency may be determined according to χ^2 criteria. If the observed in sample $N * \chi^2$ is lower than $\chi^2$of specific confidence level α, then the independence could not be rejected on that confidence level. Variables are dependent.

Table 17. Chi squared and needed sample size.

	Chi squared	N needed
Our table chi sq	2,2512	1338
Chi sq for 99 %	16,812	9992
Chi sq for 95 %	12,592	7484

So in order to say that these categories (age and charges) are not interconnected on 95% confidence level sample size should be more than 7484 observations and more than 9992 observations for 99% level. It is not so and on both these confidence levels there **is a relationship** between category variables.

For the case of sex and charges we receive following:
χ^2_{our}= 18,1
$\chi^2_{95\%}$= 18,3
$\chi^2_{99\%}$= 23,2

$\chi^2_{our} > \chi^2_{99\%}$ then with 99% and 95% confidence levels these variables **are dependent.**

To sum up, age and the number of children are need to describe charges as they are related to each other on 95% confidence level.

Hence, in the statistics textbooks, the chi-squared is not recommended as a measure of association between features.

Homework 4 – PCA/SVD

For the given dataset, a person's general health may be characterised in terms of his age (older people are more likely to be ill), bmi (a high or low bmi determines if a person is overweight or underweight), the number of children he has had (a higher number of children implies the person is more healthy), his medical charges (charges give us an idea about the person's general health as higher charges suggests that a person needs more medical care) .For this reason, we include these features in the subset of features that we have chosen.

For PCA, we use two approaches:
(i) Conventional Approach: Using Covariance Matrix
(ii) Modern Approach: Using SVD

Data scatter

Table 18. Data Scatter by component in PCA

Component number	Singular value squared	Singular value squared, %
1	132.97	46.1
1+2	210.437	72.9
1+2+3	256.559	88.9
1+2+3+4	288.39	100

So the first component describes 46,1% of variation of target variable charges. So the only one component contains not much information about data. Initial dataset is highly dimensional. First three components cumulatively describe 88,9% of y variance.

Conventional PCA approach

The classic approach to PCA is to perform the eigendecomposition on the covariance matrix Σ, which is a $d \times d$ matrix where each element represents the covariance between two features. The covariance between two features is calculated as follows:

$$\sigma_{jk} = \frac{1}{n-1} \sum_{i=1}^{N} \left(x_{ij} - x_j \right) \left(x_{ik} - x_k \right)$$

We can summarize the calculation of the covariance matrix via the following matrix equation:

$$\Sigma = \frac{1}{n-1} \left((X - \bar{x})^T (X - \bar{x}) \right)$$

where $\bar{x}$ is the mean vector $\bar{x} = \sum_{k=1}^{n} x_i$

The mean vector is a d-dimensional vector where each value in this vector represents the sample mean of a feature column in the dataset.

Next, we perform an eigen decomposition on the covariance matrix in order to get the eigen vectors and eigen values.

After this, we are reducing the 5-dimensional feature space to a 2-dimensional feature subspace, by choosing the "top 2" eigenvectors with the highest eigenvalues to construct our $d \times k$-dimensional eigenvector matrix **W**.

In this last step we will use the 5×2-dimensional projection matrix **W** to transform our samples onto the new subspace via the equation
P=**X**×**W**, where **P** is a 1338×2 matrix of our transformed samples.

With this we can visualise our Principal Components in two dimensions.

Modern SVD approach

While the eigen decomposition of the covariance matrix may be more intuitiuve, most PCA implementations perform a Singular Vector Decomposition (SVD) to improve the computational efficiency. Singular value decomposition takes a rectangular matrix of gene expression data (defined as A, where A is a n x p matrix) in which the n rows represents the genes, and the p columns represents the experimental conditions. The SVD theorem states:

$$A_{nxp} = U_{nxn} S_{nxp} V_{pxp}^{T}$$

where $U^T U = I_{nxn}$, $V^T V = I_{pxp}$

Where the columns of U are the left singular vectors (gene coefficient vectors); S (the same dimensions as A) has singular values and is diagonal (mode amplitudes); and V^T has rows that are the right singular vectors (expression level vectors). The SVD represents an expansion of the original data in a coordinate system where the covariance matrix is diagonal.

Calculating the SVD consists of finding the eigenvalues and eigenvectors of AA^T and $A^T A$. The eigenvectors of $A^T A$ make up the columns of V , the eigenvectors of AA^T make up the columns of U. Also, the singular values in S are square roots of eigenvalues from AA^T or $A^T A$.

Visualization

For the Conventional PCA and modern SVD approach we received the same results in visualisation, as both approaches are developed to solve the same task.

Authors implement 2 types of scaling: by range and by std. Result pictures differ significantly on the factor of scaling.

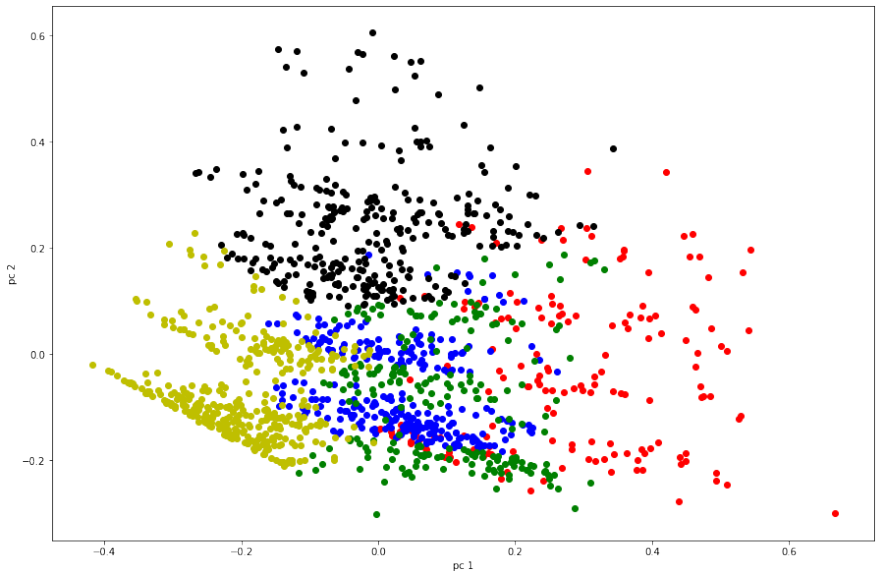


Figure 4. First and second components with scaling by std.

In this case black and yellow clusters separate pretty well.

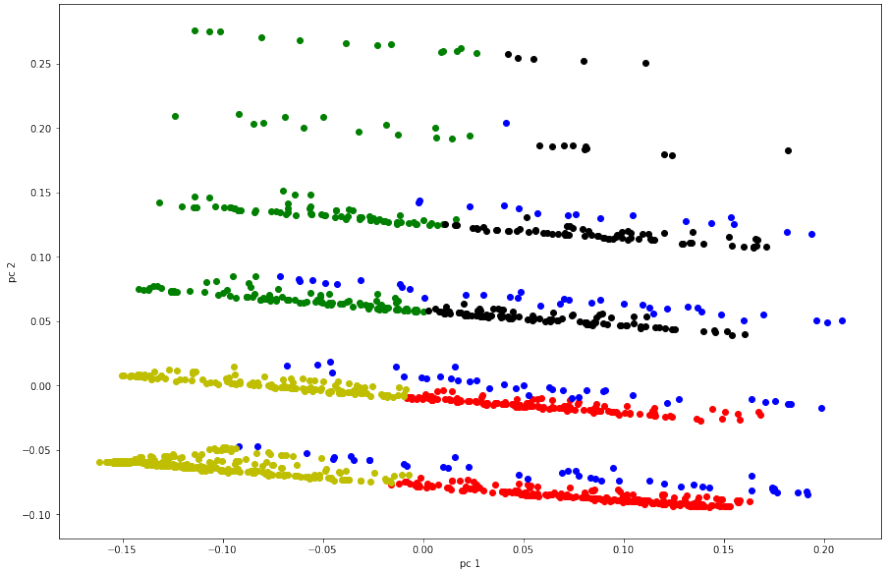


Figure 5. First and second components with scaling by range.

For such case yellow and green clusters separate good.

Clusters were taken only by charges categories presented in contingency tables home assignment. It is clear that this division is not perfect one.

Hidden factors

In order to compute and interpret a hidden factor we will go through this method:

1) To take features. So variables age, bmi, children are taken;
2) To normalize them into [0-100] scale. We implement it this way:

```
max_ = data.max()
min_ = data.min()
mean_ = data.mean()
scaled_data = data.copy()
scaled_data  = (scaled_data - min_) * 100 / (max_ - min_)
X = scaled_data.values
```

3) To calculate first singular triplet. Implementation method:

```
Z, Mu, C = np.linalg.svd(X)
z = Z[:, 0]
c = C[0, :]
z = -z
c = -c
```

4) To determine factor α. It is done by:

```
alpha = 1. / c.sum()
alpha = 0.7048567189773052
```

5) To compute the PCA hidden factor score vector. Following code:
   hidden factor = np.dot(X, c) * alpha

   We receive the following vector: c = [0.805485, 0.592243, 0.020999].
   So that **hidden factor** = 0.80*age + 0.59*bmi + 0.02*children

6) To determine its contribution to the data scatter.
   100 * Mu[0]**2 / (X*X).sum()

   And the result is 82.31 . So by means of SVD approach the explaining factor is 82.31. It is near 100 and it shows that hidden factors describe the data quite well.

Homework 5 – Linear Regression

According to scatter plots (see in the supplement materials in the end of report) variables of charges and age seem to have higher correlation then other variables. It also could be proven by the idea of having more serious and chronic diseases when getting older. Human's organism depletes by aging, ecological, social factors.

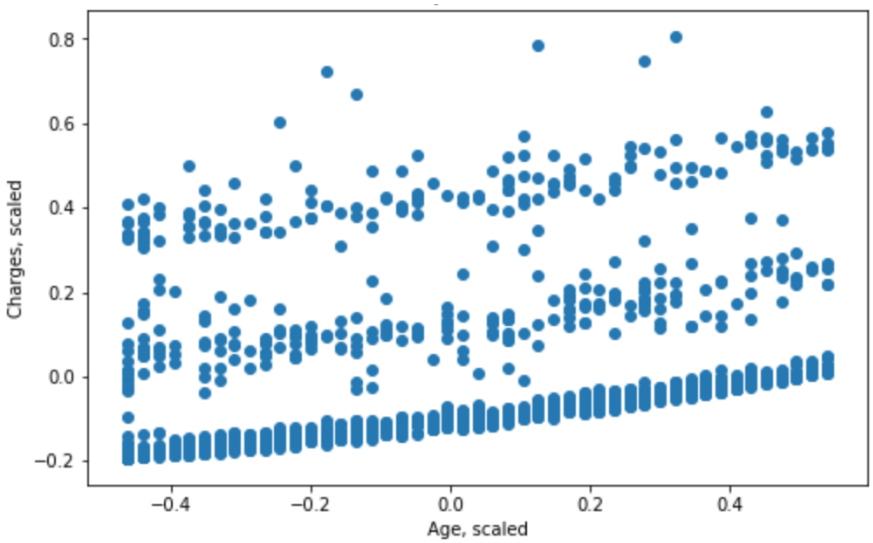


Figure 6. Scatter plot of scaled variables of age and charges.

Math behind linear regression

We describe the following relationship of the target variable y (charges) and dependent variable x (age):

$$y_i = \alpha + \beta x_i + \varepsilon_i$$

Using least-squares approach one should solve the following optimization task:

$$\text{Find } \min_{a,\,b} Q(a,b), \quad \text{for } Q(a,b) = \sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} = \sum_{i=1}^{n} (y_i - a - b x_i)^2$$

It is derivable from the optimisation task:

$$\hat{\alpha} = \bar{y} - \hat{\beta}\,\bar{x}$$

$$\hat{\beta} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{\operatorname{Cov}(x,y)}{\operatorname{Var}(x)}$$

where

- $\bar{y}$ and $\bar{x}$ are sample averages for y_i and x_i respectively;
- r_{xy} – sample correlation coefficient between x and y;
- Cov(x,y) – covariance between x and y;
- Var(x) – variance of x.

$$r_{xy} = \frac{\overline{xy} - \bar{x}\bar{y}}{\sqrt{\left(\overline{x^2} - \bar{x}^2\right)\left(\overline{y^2} - \bar{y}^2\right)}}$$

Coefficients

Cov(x,y) = 23.6, Var(x) = 124.7

The intercept $\boldsymbol{\alpha}$ **is almost 0** (true value obtained is 1.8800347057592752e-08). It is due to the scaling of variables before modelling. If back transformed to the initial scale, then it is **13270.42**. So the average prediction without taking age into account is $13270.42 charge paid to insurance company.

The slope **beta is 0.189**. It means that there is a weak positive correlation between the age and charges. The more the age the more the charges.

There is a weak positive **correlation of 29.90%** between the age and charges variables. As well **determinacy coefficient equals 8.94%** and it implies that as much as 91.06% of y-scatter left unexplained).

So the only age is not good enough to describe y and predict it correctly.

Calculating prediction

Using the parameters of α and β prediction could be calculated by solving:

$$y_i = \alpha + \beta * x_i$$

Algorithm of predictions calculations:

1) real age of 20, 30, 40, 50, 60 was taken;
2) values were scaled with subtracting mean and dividing by range;
3) using the parameters of α and β predictions were calculated by solving:

$$y_i = \alpha + \beta * x_i$$

4) values of charges predictions were scaled back to obtain real values.

Result of this algorithm implemented is shown below.

Table 19. Some prediction by linear regression

Real age	Scaled age	Real charges prediction	Scaled charges prediction
20	-0,41754	8320,34	-0,07901
30	-0,20015	10897,56	-0,03788
40	0,01724	13474,79	0,00326
50	0,23463	16052,02	0,04440
60	0,45202	18629,24	0,08554

It is intuitive that growing older people spend more money on insurance charges. Some problems tend to appear only after 30, 40 or 50 years. Also, some external factors cumulate and affect people's health. For example, ecological factors such as polluted air may lead to respiratory disease. Renters are less active and the spine and bones problems.

Model evaluation

$$MRAE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - y_{pred}}{y_i}\right|$$

Mean relative absolute error = 265.34%. It means that our predictions on average is 265.34% smaller or larger than the target.

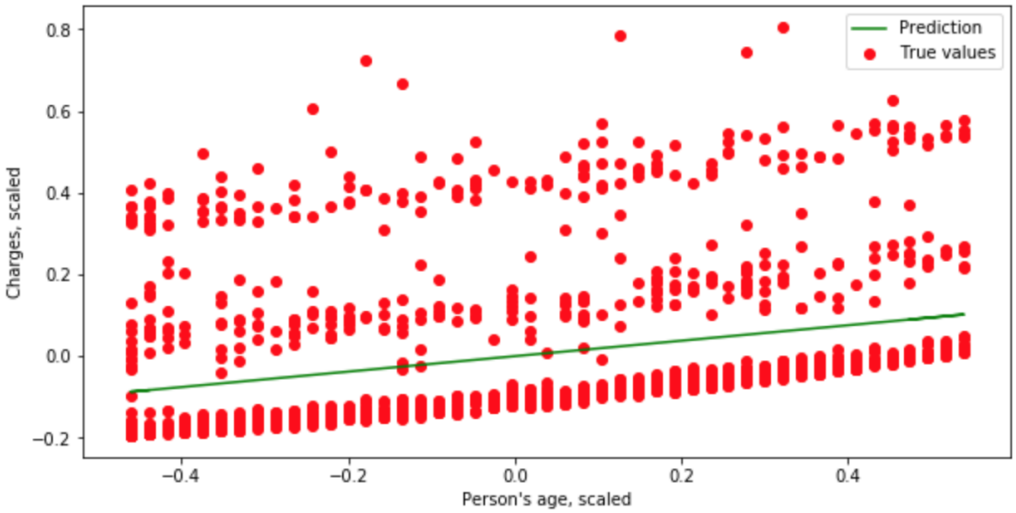


Figure 7. Scatter plot of scaled age and charges with the prediction for every charge by linear regression.

Looking at the figure above it is obvious that the only age can not describe target variable perfectly. Still, it has the highest correlation coefficient with the target.

According to determinacy coefficient R^2 = 8.94% and Mean relative absolute error = 265.34%, may be concluded that age alone in not very good predictor for charges.

Supplement materials

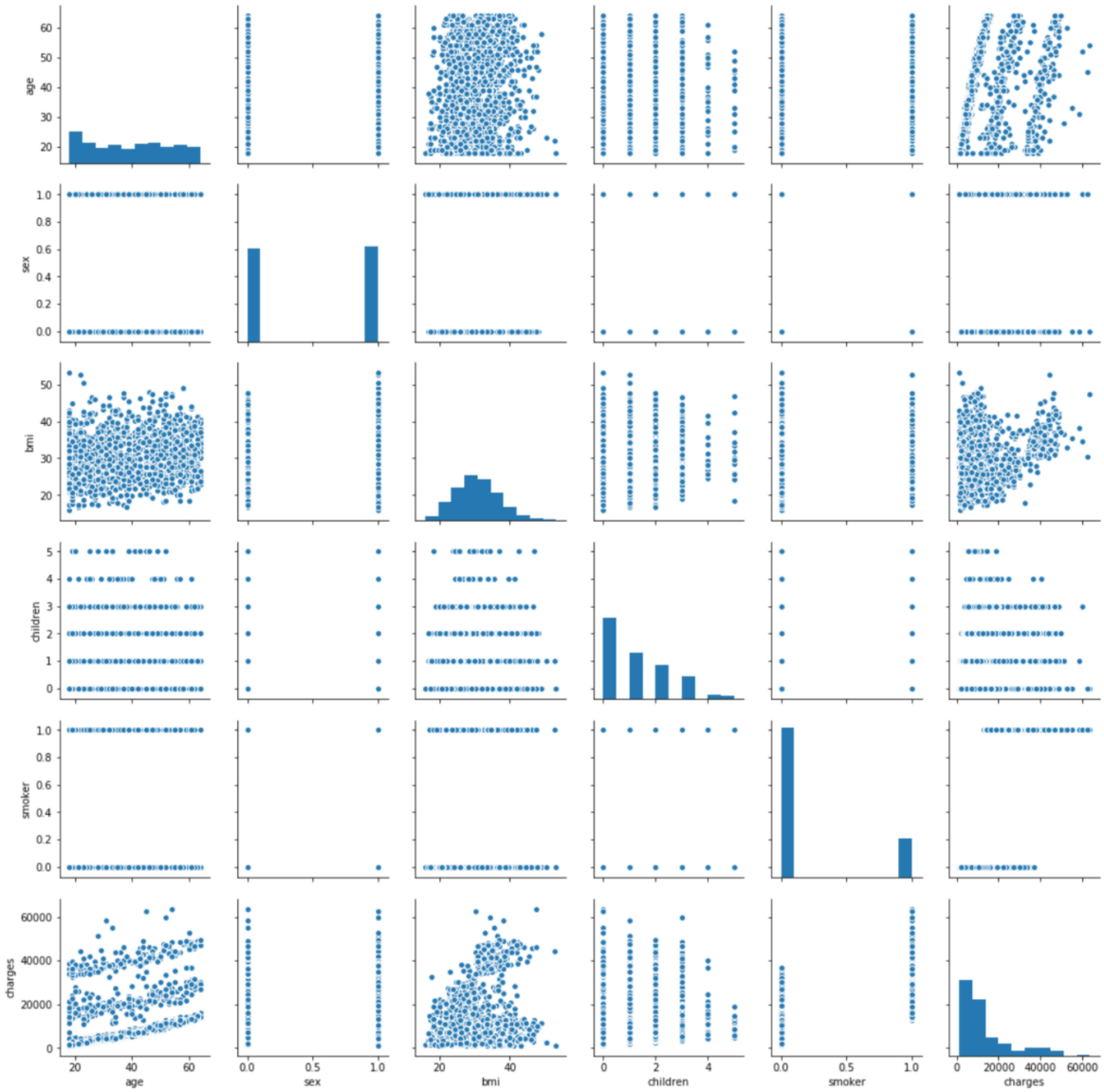


Figure 8. Scatter plots and histograms of Insurance dataset features.

Homework code

Homework 1 code

```
# -*- coding: utf-8 -*-
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from random import randint

np.random.RandomState(seed = 42)

data = pd.read_csv( "insurance.csv")
Y = data["charges"]
data.drop("charges", axis=1, inplace=True)

le1 = LabelEncoder()
le2 = LabelEncoder()
ohe = OneHotEncoder()
scaler = StandardScaler()

data["sex"] = le1.fit_transform(data["sex"])
data["smoker"] = le2.fit_transform(data["smoker"])

b = pd.get_dummies(data["region"])
decrease_coeff = np.sqrt(b.shape[1])
b /= decrease_coeff
data_ohe = data.drop("region", axis=1).join(b)

#scale
min_ = data_ohe.min()
max_ = data_ohe.max()
range_ = max_ - min_
mean_ = data_ohe.mean()
scaled_data = (data_ohe - mean_ ) / range_

# scaled_data = scaler.fit_transform(data_ohe)
data_ohe_scaled = pd.DataFrame(scaled_data, columns=data_ohe.columns)

def draw_pca(data, clusters):
```

```
    n_clusters = len(np.unique(clusters))
    colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'olive', 'w']
    pca = PCA(n_components=2)
    data_pca = pca.fit_transform(data)
    plt.figure()

    for i in range(n_clusters):
        data_cl = data_pca[clusters == i]
        plt.scatter(data_cl[:, 0], data_cl[:, 1], c=colors[i])


# HW 2,I. Task1:
# Choose 3-6 features, Explain the choice, Apply K-means:
# ◦ At K=5
# ◦ At K=9
# ◦ In both cases: 10 or more random initializations, chose the best over the K-means criterion
# выбрал численные фичи (не бинарно-категориальные)
features_k_means = ["age", "bmi", "sex", "children"]
data_k = data_ohe_scaled[features_k_means]

k = 5
random_starts_number = 5
#random_states = [randint(0, 1000) for _ in range(random_starts_number)]
random_states = [374, 75, 132, 90, 746,
                 590, 763, 433, 106, 851]


# squared errors
sse = {}
for i in range(len(random_states)):
    kmeans = KMeans(n_clusters=k, init="random", n_jobs=-1,
                    random_state=random_states[i]).fit(data_k)
    sse[i] = kmeans.inertia_

plt.figure(figsize=(10,5))
plt.plot(list(sse.keys()), list(sse.values()))
plt.title(f"Looking for best random state for {k} clusters")
plt.xlabel("number of random random state")
plt.ylabel("SSD")
plt.show()

# find minimum sse
best_random_state_5 = random_states[np.argmin(list(sse.values()))]
kmeans = KMeans(n_clusters=k, init="random", n_jobs=-1,
```

```
                random_state=best_random_state_5).fit(data_k)
 # draw it
clusters = kmeans.predict(data_k)
draw_pca(data_k, clusters)

data_k_cls_5 = []
for i in range(len(np.unique(clusters))):
    data_k_cls_5.append(data_k[clusters==i])

# clustering explanation
df_mean = pd.DataFrame(data=data_k_cls_5[0].mean())
for i in range(1,k):
    df_mean[f"{i}"] = data_k_cls_5[i].mean()
df_mean = df_mean.T

def plot_it(df):
    plt.figure(figsize=(10,5))
    plt.title("clustering")
    for i in range(df.shape[0]):
        plt.plot(list(df.columns), df.iloc[i, :].values, label = f"{i} cluster")
    plt.legend(bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)
    plt.show()

# just to improve visualization
df_mean["bmi"] *= 10
plot_it(df_mean)

# unscaled mean values by clusters
data_cluster = data.copy()
data_cluster["cluster"] = clusters
data_cluster_mean_5 = data_cluster.groupby("cluster").mean()

k = 9
# squared errors
sse = {}
for i in range(len(random_states)):
    kmeans = KMeans(n_clusters=k, init="random", n_jobs=-1,
                random_state=random_states[i]).fit(data_k)
    sse[i] = kmeans.inertia_

plt.figure(figsize=(10,5))
plt.plot(list(sse.keys()), list(sse.values()))
plt.title(f"Looking for best random state for {k} clusters")
plt.xlabel("number of random random state")
plt.ylabel("SSD")
```

```
plt.show()

# find minimum sse
best_random_state_9 = random_states[np.argmin(list(sse.values()))]
kmeans = KMeans(n_clusters=k, init="random", n_jobs=-1,
                random_state=best_random_state_9).fit(data_k)
 # draw it
clusters = kmeans.predict(data_k)
draw_pca(data_k, clusters)

data_k_cls_9 = []
for i in range(len(np.unique(clusters))):
    data_k_cls_9.append(data_k[clusters==i])

# clustering explanation
df_mean = pd.DataFrame(data=data_k_cls_9[0].mean())
for i in range(1,k):
    df_mean[f"{i}"] = data_k_cls_9[i].mean()
df_mean = df_mean.T

# just to improve visualization
df_mean["bmi"] *= 12
plot_it(df_mean)

# unscaled mean values by clusters
data_cluster = data.copy()
data_cluster["cluster"] = clusters
data_cluster_mean_9 = data_cluster.groupby("cluster").mean()
data_cluster_mean_9

# k = 5 better

# HomeWork 2, II
# 3. Take one of the partitions
#◦ 3.1. Compare one of the features between two clusters
#with using bootstrap
#◦ 3.2.Take a feature, find the 95% confidence interval for its grand mean by using
bootstrap
#◦ 3.3.Take a cluster, and compare the grand mean with the within-cluster mean
for the feature by using bootstrap
#◦ Note: each application of bootstrap should be done in both, pivotal and non-
pivotal, versions

print("Task #3")
feature = "age"
```

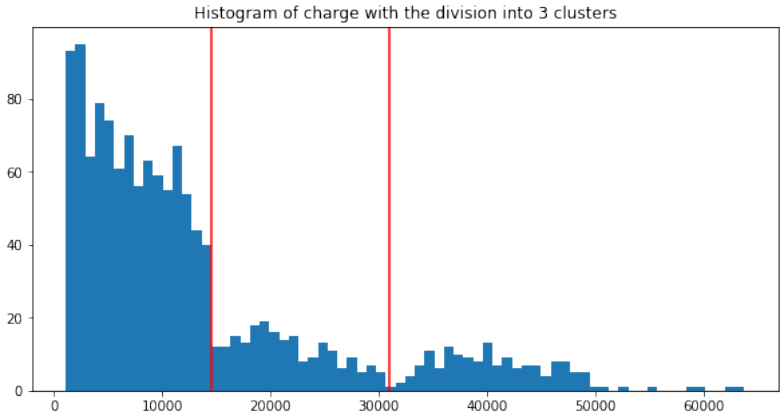
```
first_cluster_no = -1
second_cluster_no = 0

data_part1 = data_k_cls_5[first_cluster_no][feature]
data_part2 = data_k_cls_5[second_cluster_no][feature]

# number of bootstrap iterations
M = 50000
r_indeces_1 = []
r_indeces_2 = []
for i in range(M):
    r_indeces_1.append(randint(0, data_part1.shape[0] - 1))
    r_indeces_2.append(randint(0, data_part2.shape[0] - 1))

dp1_bootstraped = data_part1.iloc[r_indeces_1]
dp2_bootstraped = data_part2.iloc[r_indeces_2]


plt.hist(dp1_bootstraped, label = f'4 cluster {feature}', alpha=0.5)
plt.title(f"feature's historogram for {ind} and 4 clusters")
plt.hist(dp2_bootstraped, label = f'{second_cluster_no} cluster {feature}',
alpha=0.5)
plt.legend()
plt.show();

# pivotal method
mmr1 = dp1_bootstraped.mean()
smr1 = dp1_bootstraped.std()
mmr2 = dp2_bootstraped.mean()
smr2 = dp2_bootstraped.std()
# 3.1
print(f"compare feature pivotal method. \nmean cluster1 {mmr1}, \nmean
cluster2 {mmr2}")

mean_interval1 = (mmr1 - 1.96 * smr1, mmr1 + 1.96 * smr1)
mean_interval2 = (mmr2 - 1.96 * smr2, mmr2 + 1.96 * smr2)

print("\nTask 3.2")
# 3.2
print("mean intervals with 95% confidence, pivotal bootstrap method for 2
clusters:")
print(f"mean interval for cluster 1: {mean_interval1}")
print(f"mean interval for cluster 2: {mean_interval2}")
# non-pivotal method
dp1_bootstraped = dp1_bootstraped.sort_values()
```

```
dp2_bootstraped = dp2_bootstraped.sort_values()
low_index = int(np.ceil(M * 0.025))
median_index = int(M * 0.5)
high_index = int(M * 0.975)

mmrn1 = dp1_bootstraped.iloc[median_index]
mmrn2 = dp2_bootstraped.iloc[median_index]
print(f"\ncompare feature non-pivotal method. \nmean cluster1 {mmrn1}, \nmean
cluster2 {mmrn2}")

mean_interval1_method2 = (dp1_bootstraped.iloc[low_index],
                          dp1_bootstraped.iloc[high_index])
mean_interval2_method2 = (dp2_bootstraped.iloc[low_index],
                          dp2_bootstraped.iloc[high_index])
# 3.2
print("mean intervals with 95% confidence, non-pivotal bootstrap method for 2
clusters")
print(f"cluster 1: {mean_interval1_method2}")
print(f"cluster 2: {mean_interval2_method2}")

# 3.3
print("\nTask 3.3")
data_out_of_cluster = data_k[clusters != first_cluster_no][feature]
r_indeces = []
for i in range(M):
    r_indeces.append(randint(0, data_out_of_cluster.shape[0] - 1))
d_bootstraped = data_out_of_cluster.iloc[r_indeces]
mmr = d_bootstraped.mean()
print("compare grand mean (pivotal method) for cluster 1 and all others data")
print(f"mean cluster1: {mmr1} \nmean others data: {mmr}")

d_bootstraped = d_bootstraped.sort_values()
mmrn = d_bootstraped.iloc[median_index]
print("compare grand mean (non-pivotal method) for cluster 1 and all others
data")
print(f"mean cluster1: {mmrn1} \nmean others data: {mmrn}")
```

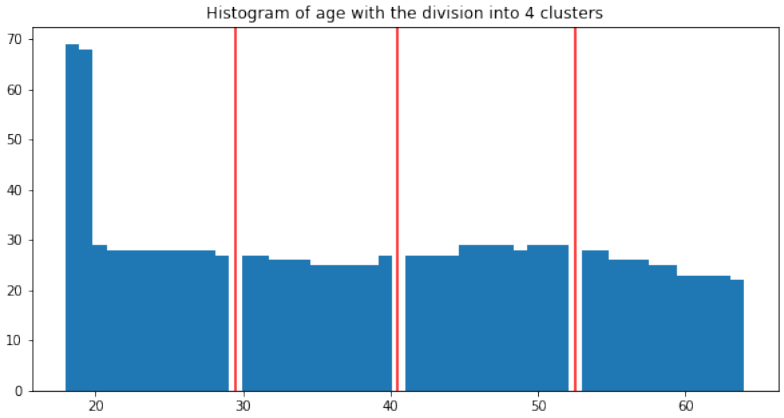
Homework 3 code

Homework 3 - Contingency Table

```
plt.figure(figsize=(10,5))
plt.title("Histogram of charge with the division into 3 clusters")
plt.hist(data_f.charges, bins = 70)
plt.axvline(x = 14500, color = "red")
plt.axvline(x = 31000, color = "red")
plt.show()
```




```
plt.figure(figsize=(10,5))
plt.title("Histogram of age with the division into 4 clusters")
plt.hist(data_f.age, bins = 50)
plt.axvline(x = 29.5, color = "red")
plt.axvline(x = 40.5, color = "red")
plt.axvline(x = 52.5, color = "red")
plt.show()
```




```
data_f["charges_category"] = data_f["charges"].apply(
    lambda x: "(.. , 14.5]" if x <=14500
        else ("(31, ..)" if x>31000
            else "(14.5, 31]")
                                        )
data_f["age_category"] = data_f["age"].apply(
    lambda x: "(.., 29]" if x <=29.5
        else ("(29, 40]" if x <= 40.5
```

```
            else ("(40, 52]" if x <= 52.5 else "(52, ..)")
             )
                                    )
data_f.head()
```

	age	sex	bmi	children	smoker	region	charges	charges_category	age_category
0	19	0	27.900	0	1	southwest	16884.92400	(14.5, 31]	(.., 29]
1	18	1	33.770	1	0	southeast	1725.55230	(.. , 14.5]	(.., 29]
2	28	1	33.000	3	0	southeast	4449.46200	(.. , 14.5]	(.., 29]
3	33	1	22.705	0	0	northwest	21984.47061	(14.5, 31]	(29, 40]
4	32	1	28.880	0	0	northwest	3866.85520	(.. , 14.5]	(29, 40]

```
ct_real_no_total = pd.crosstab(data_f["charges_category"],
data_f["age_category"])
ct_real_no_total.loc["Total"] = ct_real_no_total.loc["(.. , 14.5]"] \
                    + ct_real_no_total.loc["(14.5, 31]"] \
                    + ct_real_no_total.loc["(31, ..)"]

ct_real_no_total["Total"] = ct_real_no_total["(.., 29]"] \
                        + ct_real_no_total["(29, 40]"] \
                        + ct_real_no_total["(40, 52]"] \
                        + ct_real_no_total["(52, ..)"]

# Contingency (co-occurence) table
ct_real_no_total
```

age_category	(.., 29]	(29, 40]	(40, 52]	(52, ..)	Total
charges_category					
(.. , 14.5]	310	210	245	209	974
(14.5, 31]	61	42	52	53	208
(31, ..)	46	32	42	36	156
Total	417	284	339	298	1338

```
def table_rounding(table, dec = 3):
    for i in range(table.shape[0]):
        for j in range(table.shape[1]):
            table.iloc[i,j] = round(table.iloc[i,j],dec)
    return table

rel_ct_age_charge = pd.crosstab(data_f["charges_category"],
                        data_f["age_category"],
                        normalize="all")
```

```
rel_ct_age_charge.loc["Total"] = rel_ct_age_charge.loc["(.. , 14.5]"] \
                    + rel_ct_age_charge.loc["(14.5, 31]"] \
                    + rel_ct_age_charge.loc["(31, ..)"]

rel_ct_age_charge["Total"] = rel_ct_age_charge["(.., 29]"] \
                      + rel_ct_age_charge["(29, 40]"] \
                      + rel_ct_age_charge["(40, 52]"] \
                      + rel_ct_age_charge["(52, ..)"]
```

Relative contingency (co-occurence) table shows that according to this dataset the probability 0.728 of people paid less than $14500 is much higher than other charges categories (0.155 for charges between 14.5 and 31 and 0.117 for charges over 31). So no matter what the age is more people pay less than$ 14500is much higher than other charges categories(0.155for charges between 14.5and31and0.117for charges over31).So no matter what the age is more people pay less than14500.

table_rounding(rel_ct_age_charge)

age_category	(.., 29]	(29, 40]	(40, 52]	(52, ..)	Total
charges_category					
(.. , 14.5]	0.232	0.157	0.183	0.156	0.728
(14.5, 31]	0.046	0.031	0.039	0.040	0.155
(31, ..)	0.034	0.024	0.031	0.027	0.117
Total	0.312	0.212	0.253	0.223	1.000

```
a = rel_ct_age_charge.loc["Total"].values.reshape(5,1)[:4]
b = rel_ct_age_charge["Total"].values[:3].reshape(1,3)

diff_age_charge = pd.DataFrame(np.dot(a,b)).T
for i in range(diff_age_charge.shape[0]):
    for j in range(diff_age_charge.shape[1]):
        diff_age_charge.iloc[i,j] = diff_age_charge.iloc[i,j] ** 2

chi = pd.DataFrame(np.dot(a,b))

chi2_age_charge = 0
for row in range(chi.shape[0]):
    for col in range(chi.shape[1]):
        chi2_age_charge+= chi.iloc[row,col]

chi2_age_charge
1.0
```

```
cond_prob_age_charge = pd.crosstab(data_f["charges_category"],
                                   data_f["age_category"],
                                   normalize="columns")

cond_prob_age_charge.loc["Total"] = cond_prob_age_charge.loc["(.. , 14.5]"] \
                                  + cond_prob_age_charge.loc["(14.5, 31]"] \
                                  + cond_prob_age_charge.loc["(31, ..)"]

cond_prob_age_charge["Total"] = cond_prob_age_charge["(.., 29]"] \
                                + cond_prob_age_charge["(29, 40]"] \
                                + cond_prob_age_charge["(40, 52]"] \
                                + cond_prob_age_charge["(52, ..)"]
```

Conditional probability table of age category given charges category. There appears no indicative relations. Probabilities for age groups by every charges category are almost the same.

cond_prob_age_charge

age_category	(.., 29]	(29, 40]	(40, 52]	(52, ..)	Total
charges_category					
(.. , 14.5]	0.743	0.739	0.723	0.701	2.907
(14.5, 31]	0.146	0.148	0.153	0.178	0.625
(31, ..)	0.110	0.113	0.124	0.121	0.468
Total	1.000	1.000	1.000	1.000	4.000

```
quetelet_indices = cond_prob_age_charge/rel_ct_age_charge - 1
quetelet_indices
```

age_category	(.., 29]	(29, 40]	(40, 52]	(52, ..)	Total
charges_category					
(.. , 14.5]	2.203	3.707	2.951	3.494	2.993
(14.5, 31]	2.174	3.774	2.923	3.450	3.032
(31, ..)	2.235	3.708	3.000	3.481	3.000
Total	2.205	3.717	2.953	3.484	3.000

```
ct_real_charges_sex = pd.crosstab(data_f["charges_category"], data_f["sex"])
ct_real_charges_sex.loc["Total"] = ct_real_charges_sex.loc["(.. , 14.5]"] \
                                 + ct_real_charges_sex.loc["(14.5, 31]"] \
                                 + ct_real_charges_sex.loc["(31, ..)"]

ct_real_charges_sex["Total"] = ct_real_charges_sex[0] \
```

```
                    + ct_real_charges_sex[1]

cond_prob_sex_charge = pd.crosstab(data_f["charges_category"], data_f["sex"],
normalize="columns")
cond_prob_sex_charge.loc["Total"] = cond_prob_sex_charge.loc["(.. , 14.5]"] \
                    + cond_prob_sex_charge.loc["(14.5, 31]"] \
                    + cond_prob_sex_charge.loc["(31, ..)"]

cond_prob_sex_charge["Total"] = cond_prob_sex_charge[0] \
                    + cond_prob_sex_charge[1]
rel_ct_charges_sex  =   pd.crosstab(data_f["charges_category"],  data_f["sex"],
normalize="all")

rel_ct_charges_sex.loc["Total"] = rel_ct_charges_sex.loc["(.. , 14.5]"] \
                    + rel_ct_charges_sex.loc["(14.5, 31]"] \
                    + rel_ct_charges_sex.loc["(31, ..)"]

rel_ct_charges_sex["Total"] = rel_ct_charges_sex[0] \
                    + rel_ct_charges_sex[1]

quetelet_indices_sex = cond_prob_sex_charge/rel_ct_charges_sex - 1
quetelet_indices_sex
```

sex	0	1	Total
charges_category			
(.. , 14.5]	1.022	0.978	1.000
(14.5, 31]	1.025	0.987	1.006
(31, ..)	1.047	0.986	0.991
Total	1.020	0.980	1.000

```
# Quetelet index code

# -*- coding: utf-8 -*-
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from random import randint
```

```
# load data
datapath = “path"
data = pd.read_csv(datapath + "insurance.csv")
data = data[[ "age", "charges"]]
```

	age_cat	charges_category
0	0	2
1	0	0
2	0	0
3	1	2
4	1	0

```
# preprocessing
data["age_cat"] = data["age"].apply(
    lambda x: 0 if x <=29
        else (1 if x<=40
            else (2 if x <= 52
                else 3)))

data["charges_category"] = data["charges"].apply(
    lambda x: 0 if x <=14500
        else (1 if x>31000
            else 2))
data.drop(["age", "charges"], inplace=True, axis=1)

# create co-occurrence table
d = np.zeros([3, 4])
for age in range(4):
    for charg in range(3):
        d[charg, age] = data[data["age_cat"]==age][data["charges_category"]==charg].count()[0]

# count
df = pd.DataFrame(data=d, columns=["age_cat_1", "age_cat_2", "age_cat_3", "age_cat_4"])
```

	age_cat_1	age_cat_2	age_cat_3	age_cat_4
0	310.0	210.0	245.0	209.0
1	46.0	32.0	42.0	36.0
2	61.0	42.0	52.0	53.0

```
# calculate probabilites
```

df= df / data.shape[0]

	age_cat_1	age_cat_2	age_cat_3	age_cat_4
0	0.231689	0.156951	0.183109	0.156203
1	0.034380	0.023916	0.031390	0.026906
2	0.045590	0.031390	0.038864	0.039611

# calculate Quetelet index
qi = (df.divide(df.sum(axis=0), axis=1).divide(df.sum(axis=1), axis=0) - 1) * 100

	age_cat_1	age_cat_2	age_cat_3	age_cat_4
0	2.122819	1.577638	-0.719594	-3.655444
1	-5.386460	-3.358613	6.262764	3.613836
2	-5.900664	-4.868635	-1.327434	14.406944

# calculate chi-square
phk = df.sum(axis=0)
pgl = df.sum(axis=1)

```
s = 0
for i in range(4):
    for j in range(3):
        s += (df.iloc[j,i] - phk[i] * pgl[j])**2 / phk[i] / pgl[j]

data.shape[0] * s
```

```
2.1708919942823957
```

summary_qi = df * qi

	age_cat_1	age_cat_2	age_cat_3	age_cat_4
0	0.491834	0.247611	-0.131764	-0.570992
1	-0.185185	-0.080326	0.196589	0.097233
2	-0.269014	-0.152827	-0.051589	0.570679

Homework 4 code

```
# load libraries
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from random import randint

datapath = “path”
data = pd.read_csv(datapath + "insurance.csv")

# взял численные фичи
feature_1 = "age"
feature_2 = "bmi"
feature_3 = "children"
target = "charges"

data = data[[feature_1, feature_2, feature_3, target]]
data["charges_category"] = data["charges"].apply(
    lambda x: 1 if x <=14500
        else (2 if x>31000
            else 3))

cat = data[['charges_category']]
data.drop(["charges_category"], axis=1, inplace=True)

def scale_data(data):
    min_ = data.min()
    max_ = data.max()
    range_ = max_ - min_
    mean_ = data.mean()
    std_ = data.std()
    scaled_data = (data - mean_ ) / range_
    return scaled_data

X = scale_data(data).values

Z,Mu,C = np.linalg.svd(X)
z1 = Z[:, 0] * np.sqrt(Mu[0])
z2 = Z[:, 1] * np.sqrt(Mu[1])
explainded_variance = Mu[0]**2 + Mu[1]**2
```

```
all_variance = sum([x**2 for x in Mu])
print("natural contribution: {nc}\ncontribution in percent: {pc}"
      .format(nc=explainded_variance,     pc=100     *     explainded_variance     /
all_variance))

cat["x"] = z1
cat["y"] = z2

cats = []
for i in range(3):
    cats.append(cat[cat["charges_category"] == i + 1])
plt.figure(figsize=(15,10))
plt.xlabel("pc 1")
plt.ylabel("pc 2")
plt.scatter(cats[0]["x"], cats[0]["y"], c="r")
plt.scatter(cats[1]["x"], cats[1]["y"], c="b")
plt.scatter(cats[2]["x"], cats[2]["y"], c="g");
```

Homework 5 code

```
sns.pairplot(dataset);

plt.figure(figsize=(8,5))
plt.title("Scatter plot of variables")
plt.scatter(dataset.age, dataset.charges)
plt.ylabel("Charges, scaled")
plt.xlabel("Age, scaled")
plt.show()

X = dataset.age
y = dataset.charges

X_mean = np.mean(X)
y_mean = np.mean(y)
cov_xy = 0
var_x = 0

for i in range(len(X)):
    cov_xy += (X[i] - X_mean)*(y[i] - y_mean)
    var_x += (X[i] - X_mean)**2

back_scale_transform(alpha, "charges")

xy_sum = 0
X_sq_sum = 0
y_sq_sum = 0

for i in range(len(X)):
    xy_sum += X[i]*y[i]
    X_sq_sum += X[i] ** 2
    y_sq_sum += y[i] ** 2
    cov_xy += (X[i] - X_mean)*(y[i] - y_mean)
    var_x += (X[i] - X_mean)**2

xy_mean = xy_sum/len(X)
X_sq_mean = X_sq_sum/len(X)
y_sq_mean = y_sq_sum/len(y)

corr_coef = (xy_mean - X_mean*y_mean)/(((X_sq_mean - X_mean ** 2)*(y_sq_mean - y_mean ** 2))**0.5)

determinacy_coef = corr_coef**2
```

```
age_list = [20,30,40,50, 60]
age_list_scaled = [scale_transform(age, "age") for age in age_list]

charges_list = []
charges_list_scaled = []
i = 0

for age in age_list_scaled:
    charge_for_age_scaled = alpha + beta * age
    charge_for_age = back_transform(charge_for_age_scaled, "charges")
    charges_list.append(charge_for_age)
    charges_list_scaled.append(charge_for_age_scaled)
    print(f"For age = {age_list[i]} prediction is charges = {charge_for_age}")
    i+=1

prediction = []
for age in X:
    prediction.append(alpha + beta*age)

prediction = np.array(prediction)

plt.figure(figsize=(10,5))
plt.title("True and predicted values")
plt.plot(X, prediction, color="green", label = "Prediction")
plt.scatter(X, y, color="red", label = "True values")
plt.ylabel("Charges, scaled")
plt.xlabel("Person's age, scaled")
plt.legend()
plt.show()

mean_relative_absolut_error = np.mean(abs(y - prediction)/abs(y))
```