FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTE

OF HIGHER EDUCATION

ITMO UNIVERSITY

Report on learning practice # 1

Analysis of univariate random variables

Performed by:

Igor Vernyy, j4134c

Kirill Mukhin, j4134c

Alexander Petrov, j4134c

Bogdan Chertkov, j4132c

Saint-Petersburg

2022

**Substantiation of chosen subsample**

We have tried different combinations of dataset features to obtain well-interpreted results. A final set contains the following continuous features: “Hospital beds per thousand” (HBpT), “Excess mortality cumulative per million” (EMCpM) and “Excess mortality cumulative” (EMC).

## Non-parametric estimation of PDF

Figures 1 – 3 depict non-parametric estimations in form of histogram and using kernel density function for the chosen random variables.

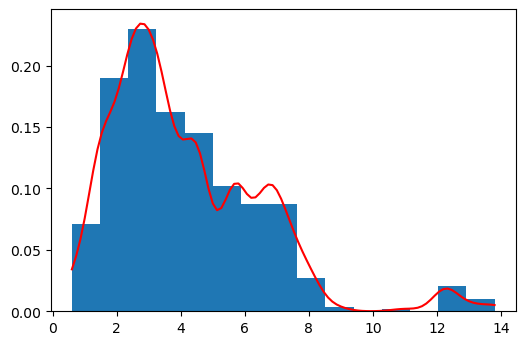


Figure 1 – Non-parametric estimation of PDF in form of histogram and using kernel density function for “Hospital beds per thousand”

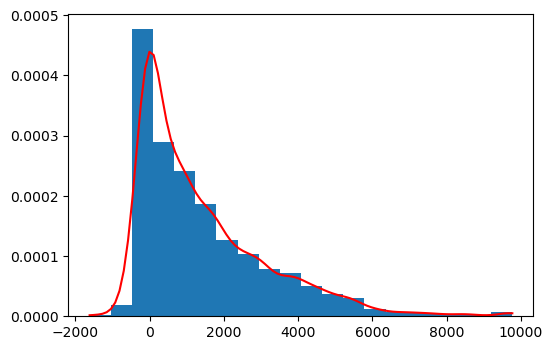


Figure 2 – Non-parametric estimation of PDF in form of histogram and using kernel density function for “Excess mortality cumulative per million”

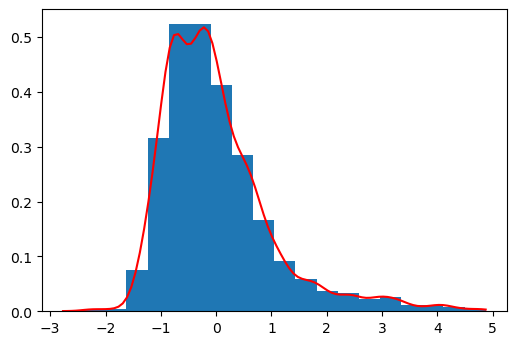


Figure 3 – Non-parametric estimation of PDF in form of histogram and using kernel density function for “Excess mortality cumulative”

## Order statistics estimation

Calculated order statistics for three random variables are presented in Table 1. Due to significant differences between variable’s mean and variance we have to normalize their values in order to draw them as box with whisker plot (Figure 4).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| HBpT |  |  |  |  |  |  |  |
| EMCpM |  |  |  |  |  |  |  |
| EMC |  |  |  |  |  |  |  |

Table 1 – Order statistics and some moments for the chosen features

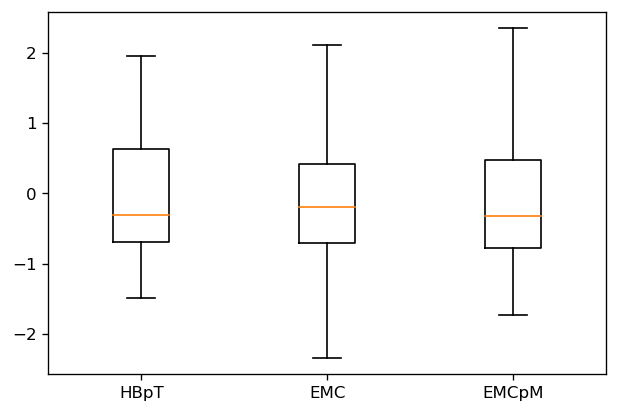


Figure 4 – Box with whisker plot for the normalized features

## Selection of theoretical distributions

Observing kernel density function plots for variables we can make propose Gamma distribution for the first variable (“Hospital beds per thousand”). You can see its probability density function at Figure 5.

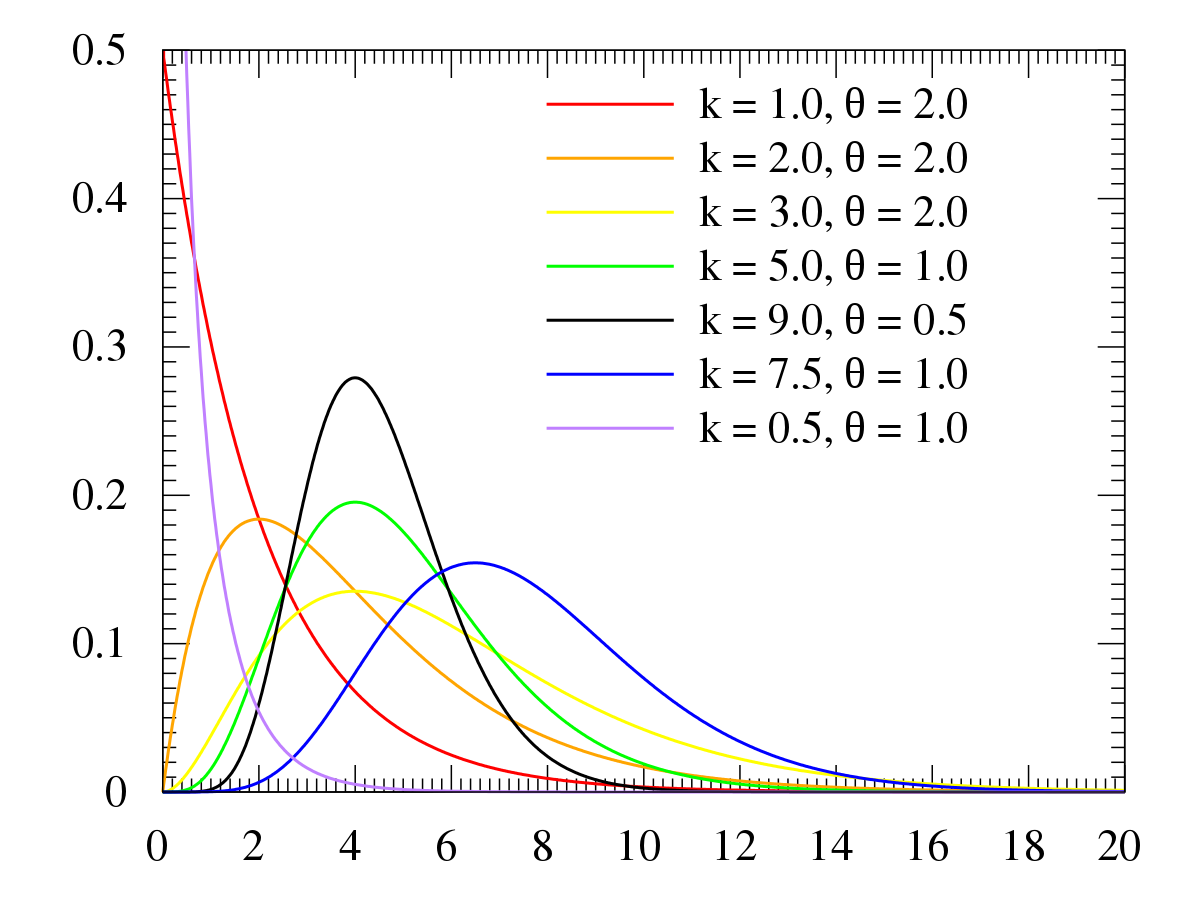


Figure 5 – Gamma distribution PDF

Figure 6 depicts chi-squared distribution which can be used to explain the second variable (“Excess mortality cumulative per million”).

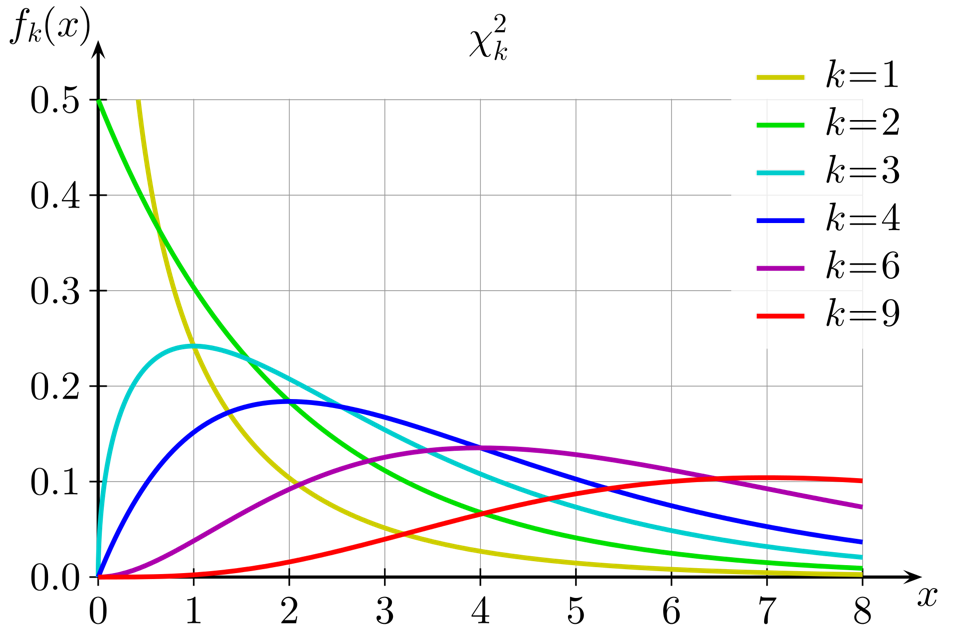


Figure 6 – Chi-squared distribution PDF

And the third variable “Excess mortality cumulative” seems like a slightly skewed normal distribution.

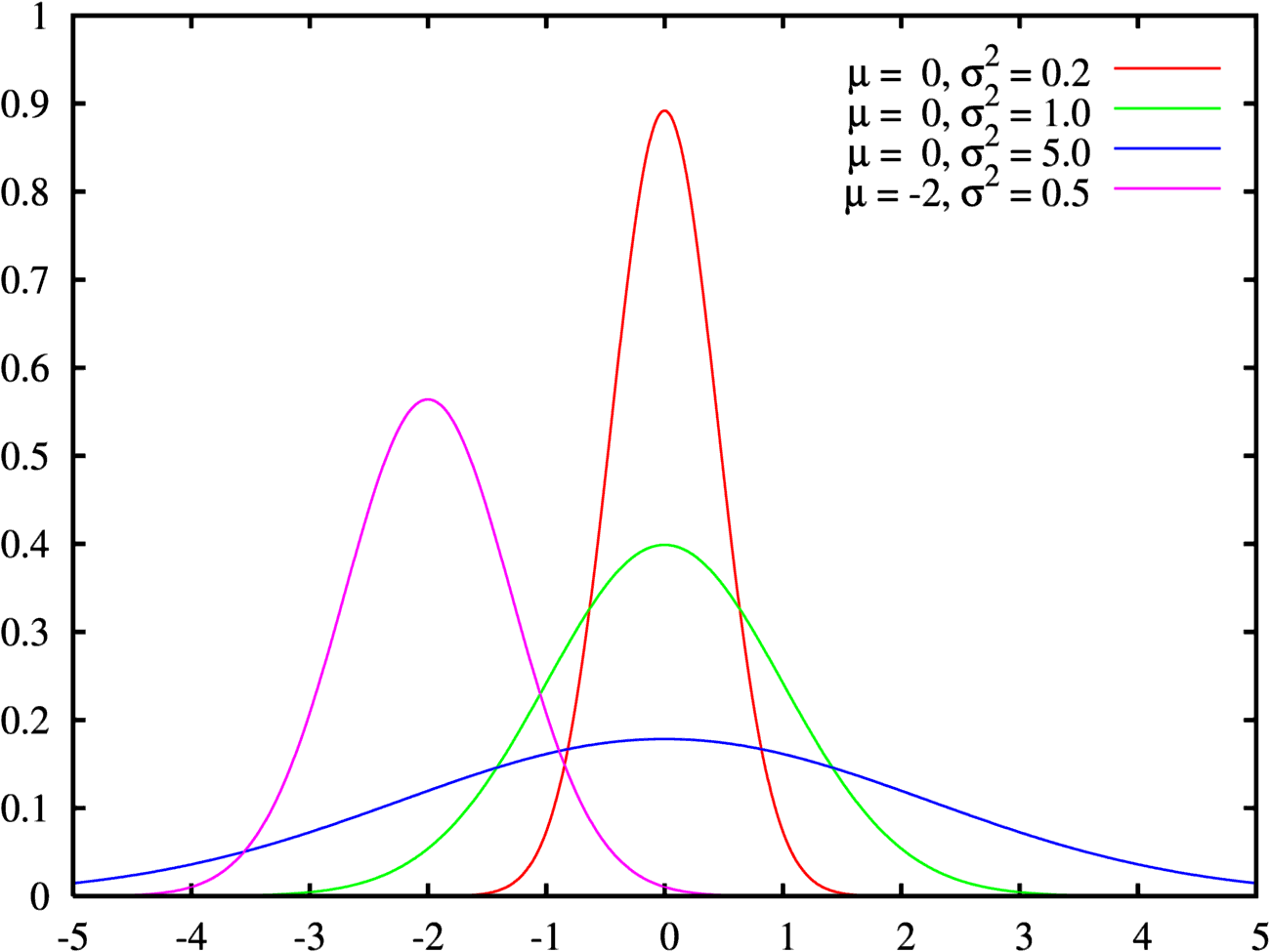


Figure 7 – Norma distribution PDF

## Estimation of random variable distribution parameters

The next step is to fit parameters for above-mentioned distributions using maximum likelihood estimation and least squares method. Figures 8 – 10 contain resulting plots.

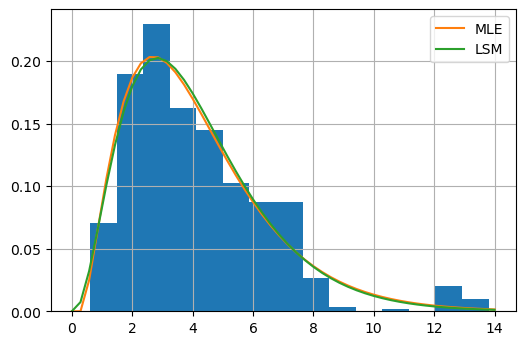


Figure 8 – Estimations of Gamma distribution using MLE and LSM

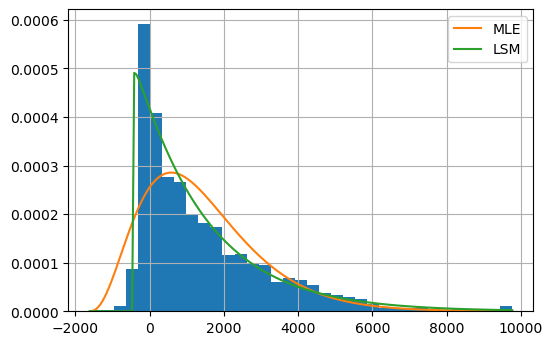


Figure 9 – Estimations of distribution using MLE and LSM

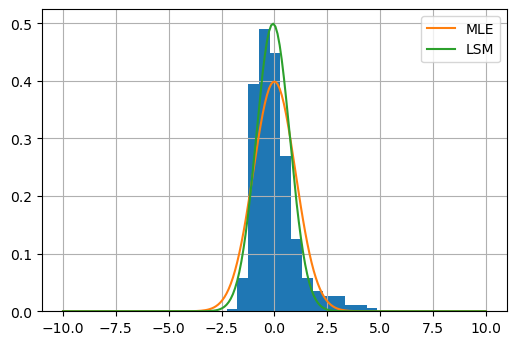


Figure 10 – Estimations of Normal distributions using MLE and LSM

## QQ-plots

In order to check the correctness of the choice of theoretical distributions for the data, we have built quantile-quantile plots. Figures 11 – 13 show them. Each figure contains plot for maximum likelihood estimation (left) and least squares method (right). You can see that points are spread along 45-degree line. Which indicates the correctness of our assumptions.

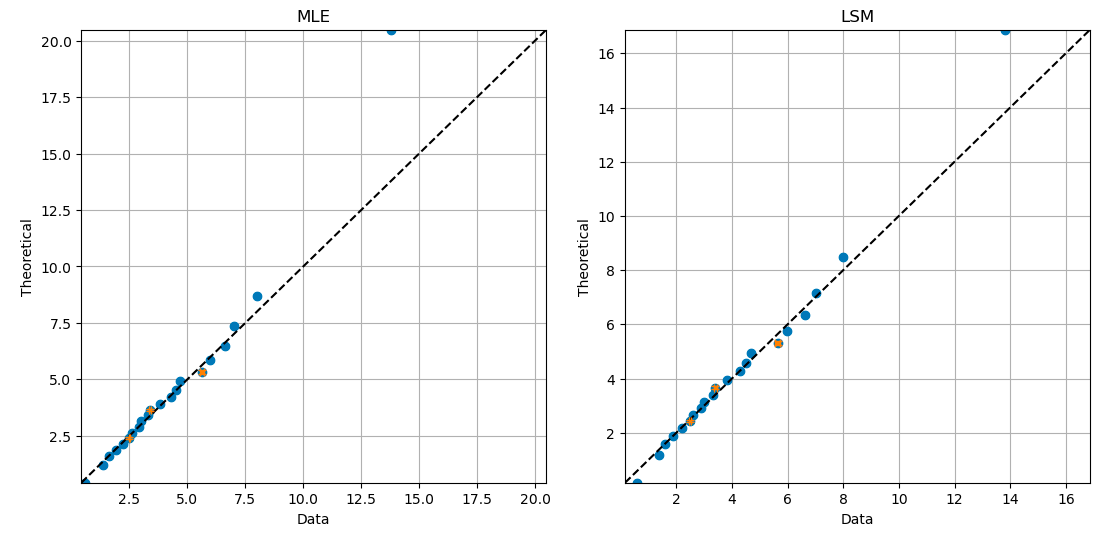


Figure 11 – QQ-plots for the first RV “Hospital beds per thousand”

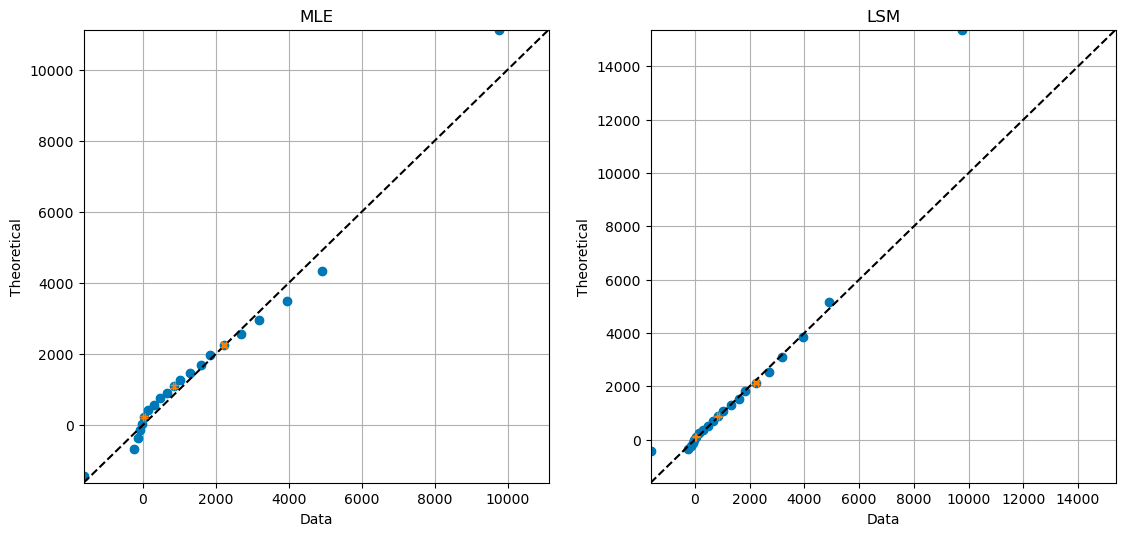


Figure 12 – QQ-plots for the second RV “Excess mortality cumulative per million”

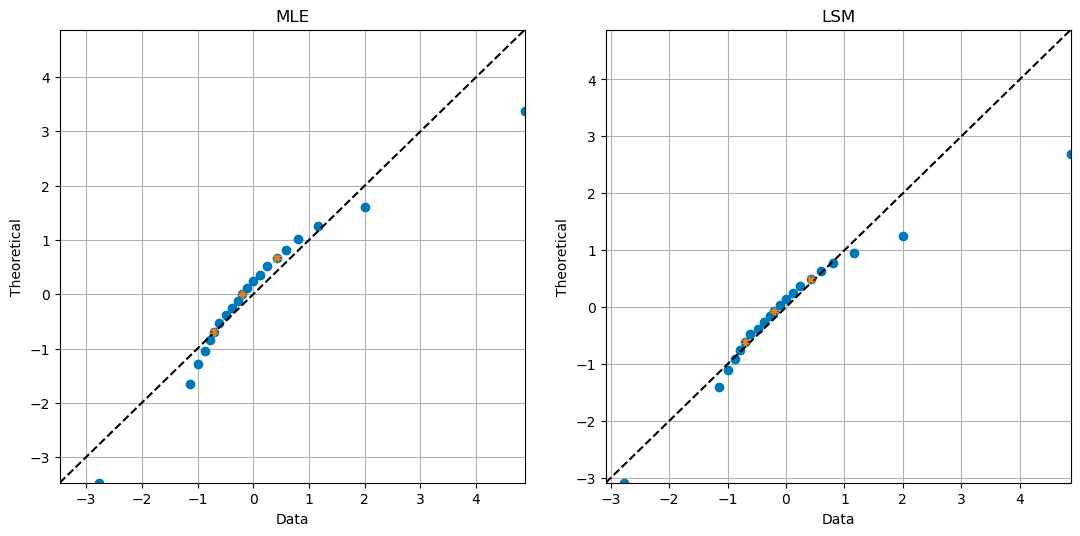


Figure 13 – QQ-plots for the third RV “Excess mortality cumulative”

## Statistical tests

Statistical tests are also an important criterion for testing hypotheses about the type of distribution. To check assumptions two tests are used – Kolmogorov (Table 2 – MLE, Table 3 – LSM) and Cramer-von-Mises (Table 4 – MLE, Table 5 – LSM). The result is defined assuming .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Distribution | Statistic | pValue | Result |
| HBpT | Gamma |  |  | Passed |
| EMCpM |  |  |  | Passed |
| EMC | Normal |  |  | Passed |

Table 2 – Kolmogorov test for MLE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Distribution | Statistic | pValue | Result |
| HBpT | Gamma |  |  | Passed |
| EMCpM |  |  |  | Passed |
| EMC | Normal |  |  | Passed |

Table 3 – Kolmogorov test for LSM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Distribution | Statistic | pValue | Result |
| HBpT | Gamma |  |  | Passed |
| EMCpM |  |  |  | Passed |
| EMC | Normal |  |  | Passed |

Table 2 – Cramer-von-Mises for MLE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Distribution | Statistic | pValue | Result |
| HBpT | Gamma |  |  | Passed |
| EMCpM |  |  |  | Passed |
| EMC | Normal |  |  | Passed |

Table 3 – Cramer-von-Mises for LSM

**Appendix**

*#!/usr/bin/env python*

*# coding: utf-8*

*# # Lab 1*

*# In[1]:*

**import** matplotlib **as** mpl

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

**import** pandas **as** pd

**import** scipy.stats

**from** scipy **import** optimize

**from** scipy **import** stats

**from** scipy.stats.distributions **import** gamma, expon, norm, chi2

**import** seaborn **as** sns

**from** IPython.display **import** display, Math

**from** functools **import** partial

*# ## Step 1. Choose subsample with main variables for your further analysis*

*# In[2]:*

path\_to\_file = '/opt/notebooks/owid-covid-data.csv'

source\_df = pd.read\_csv(path\_to\_file)

df = source\_df[['hospital\_beds\_per\_thousand', 'excess\_mortality\_cumulative', 'excess\_mortality\_cumulative\_per\_million']].copy().dropna()

*# In[3]:*

df.head(5)

*# ## Step 2. You need to make a non-parametric estimation of PDF in form of histogram and using kernel density function (or probability law in case of discrete RV).*

*# In[4]:*

**from** scipy.stats **import** kde

**def** kernel\_density\_estimation(x, bins):

density = kde.gaussian\_kde(x)

xgrid = np.linspace(x.min(), x.max(), 100)

mpl.rcParams['figure.dpi'] = 100

plt.hist(x, bins=bins, density = True)

plt.plot(xgrid, density(xgrid), 'r-')

*# ### Hospital beds per thousand*

*# In[5]:*

beds = df['hospital\_beds\_per\_thousand']

*# In[6]:*

kernel\_density\_estimation(beds, 15)

*# ### Excess mortality cumulative per million*

*# In[7]:*

emcpm = df['excess\_mortality\_cumulative\_per\_million'].dropna()

*# In[8]:*

kernel\_density\_estimation(emcpm, 20)

*# ### Exces mortality cumulative*

*# In[9]:*

emc = df['excess\_mortality\_cumulative'].dropna()

emc = (emc - emc.mean()) / emc.std()

*# In[10]:*

kernel\_density\_estimation(emc, 20)

*# ## Step 3. You need to make an estimation of order statistics and represent them as "box with whiskers" plot.*

*# In[11]:*

pd.DataFrame(

[

['Hospital beds per thousand (HBpT)', beds.min(), beds.max(), beds.mean(), beds.var(), beds.median(), np.percentile(beds, 25), np.percentile(beds, 75)],

['Excess mortality cumulative per million (EMCpM)', emcpm.min(), emcpm.max(), emcpm.mean(), emcpm.var(), emcpm.median(), np.percentile(emcpm, 25), np.percentile(emcpm, 75)],

['Excess mortality cumulative (EMC)', emc.min(), emc.max(), emc.mean(), emc.var(), emc.median(), np.percentile(emc, 25), np.percentile(emc, 75)]

],

columns=['Random Variable', 'min', 'max', 'mean', 'variance', 'median', 'Q1', 'Q3']

)

*# In[12]:*

beds\_norm = (beds - beds.mean()) / beds.std()

emc\_norm = (emc - emc.mean()) / emc.std()

emcpm\_norm = (emcpm - emcpm.mean()) / emcpm.std()

*# In[13]:*

fig1, (ax1) = plt.subplots(1, 1, dpi=120)

plt.subplots\_adjust(wspace=0.4)

ax1.boxplot(

(beds\_norm, emc\_norm, emcpm\_norm),

labels=('HBpT', 'EMC', 'EMCpM'),

showfliers=False

)

*# ax2.boxplot(*

*# ((), emca),*

*# labels=('', 'EMCA'),*

*# showfliers=False*

*# )*

*# ## Step 4. Find one or several theoretical distributions that could describe your sample on a basis of non-parametric analysis results*

*# ### Hospital beds per thousand*

*#*

*# It seems like a \*\*gamma distribution\*\*:*

*#*

*# $$x \sim \text{Gamma}(\alpha, \beta) = \frac{\beta^{\alpha}x^{\alpha-1}e^{-\beta x}}{\Gamma(\alpha)}$$*

*# ### Excess mortality cumulative per million*

*#*

*# It seems like a \*\*chi-squared\*\* distribution*

*# ### Excess mortality cumulative*

*#*

*# It seems like a \*\*Normal distribution\*\*.*

*# ## Step 5. Estimate parameters of chosen distributions using methods of maximum likelihood and least squares method*

*# In[14]:*

**def** lsm\_loss(func, data, params):

space = np.linspace(0.01, 0.95)

quantile = np.quantile(data, space)

quantile\_approximation = func.ppf(space, \*params)

**return** quantile - quantile\_approximation

*# ### Hospital beds per thousand*

*# #### Maximum likelihood method*

*# In[15]:*

beds\_mle = gamma.fit(beds, method='MLE')

beds\_mle

*# #### Least squares method*

*# In[16]:*

beds\_lsm = optimize.least\_squares(partial(lsm\_loss, gamma, beds), (1.0, 1.0, 1.0)).x

*# In[17]:*

beds\_lsm

*# In[39]:*

**from** scipy.stats.distributions **import** gamma

mpl.rcParams['figure.dpi'] = 100

beds.hist(bins=15, density=True)

plt.plot(np.linspace(0, 14), gamma.pdf(np.linspace(0, 14), \*beds\_mle), label='MLE')

plt.plot(np.linspace(0, 14), gamma.pdf(np.linspace(0, 14), \*beds\_lsm), label='LSM')

*# plt.title('Paramters estimation for "Hospital beds per thousand" feature')*

plt.legend()

*# ### Excess mortality cumulative per million*

*# #### Maximum likelihood method*

*# In[19]:*

emcpm\_mle = chi2.fit(emcpm, method='MLE')

emcpm\_mle

*# #### Least squares method*

*# In[20]:*

emcpm\_lsm = optimize.least\_squares(partial(lsm\_loss, chi2, emcpm), (1.0, 1.0, 1.0)).x

*# In[21]:*

emcpm\_lsm

*# In[40]:*

mpl.rcParams['figure.dpi'] = 100

emcpm.hist(bins=35, density=True)

space = np.linspace(min(emcpm), max(emcpm), 200)

plt.plot(space, chi2.pdf(space, \*emcpm\_mle), label='MLE')

plt.plot(space, chi2.pdf(space, \*emcpm\_lsm), label='LSM')

plt.legend()

*# ### Excess mortality cumulative*

*# #### Maximum likelihood method*

*# In[23]:*

emc\_mle = norm.fit(emc, method='MLE')

emc\_mle

*# #### Least squares method*

*# In[24]:*

emc\_lsm = scipy.optimize.least\_squares(partial(lsm\_loss, norm, emc), (1.0, 1.0)).x

*# In[25]:*

emc\_lsm

*# In[41]:*

mpl.rcParams['figure.dpi'] = 100

emc.hist(bins=15, density=True)

space = np.linspace(-10, 10, 200)

plt.plot(space, norm.pdf(space, \*emc\_mle), label='MLE')

plt.plot(space, norm.pdf(space, \*emc\_lsm), label='LSM')

plt.legend()

*# ## Step 6. Validate your estimated parameters using QQ biplots*

*# In[45]:*

**from** math **import** sqrt

*# Calculating confidence intervals for 25%, 50% and 75% quantiles*

**def** conf\_intervals(data, qn):

*# 95% quantile of Gaussian distribution*

norm\_q95 = scipy.stats.norm.ppf(0.95)

kernel = scipy.stats.gaussian\_kde(data)

p25 = len(data[data < qn[5]]) / len(data)

sigma25 = (sqrt((p25 \* (1 - p25)) / len(data))) / kernel(qn[5])

p50 = len(data[data < qn[10]]) / len(data)

sigma50 = (sqrt((p50 \* (1 - p50)) / len(data))) / kernel(qn[10])

p75 = len(data[data < qn[15]]) / len(data)

sigma75 = (sqrt((p75 \* (1 - p75)) / len(data))) / kernel(qn[15])

conf\_q25 = norm\_q95 \* sigma25

conf\_q50 = norm\_q95 \* sigma50

conf\_q75 = norm\_q95 \* sigma75

**return** [conf\_q25, conf\_q50, conf\_q75]

**def** qq(data, theoretical, feature\_name, method\_name):

*# Построение квантильного биплота для двух случайных величин*

plt.figure(figsize=(6, 6))

percs = np.linspace(0, 100, 21)

qn\_first = np.percentile(data, percs)

qn\_second = np.percentile(theoretical, percs)

min\_qn = np.min([qn\_first.min(), qn\_second.min()])

max\_qn = np.max([qn\_first.max(), qn\_second.max()])

x = np.linspace(min\_qn, max\_qn)

plt.plot(qn\_first, qn\_second, ls="", marker="o", markersize=6)

plt.plot(x, x, color="k", ls="--")

plt.xlabel('Data')

plt.ylabel('Theoretical')

plt.xlim([min\_qn, max\_qn])

plt.ylim([min\_qn, max\_qn])

plt.grid(True)

conf\_first = conf\_intervals(data, qn\_first)

conf\_second = conf\_intervals(theoretical, qn\_first)

conf\_first\_list = []

conf\_second\_list = []

**for** element1, element2 **in** zip(conf\_first, conf\_second):

conf\_first\_list.append(element1[0])

conf\_second\_list.append(element2[0])

*# Добавление доверительных интервалов на график*

plt.errorbar(

*# [25%, 50%, 75%]*

[np.percentile(data, 25), np.percentile(data, 50), np.percentile(data, 75)],

[np.percentile(theoretical, 25), np.percentile(theoretical, 50), np.percentile(theoretical, 75)],

xerr=conf\_first\_list,

yerr=conf\_second\_list,

ls='none',

capsize=3,

elinewidth=2

)

plt.title(method\_name)

plt.show()

*# ### Hospital beds per thousand*

*# #### Maximum likelihood method*

*# In[46]:*

beds\_theoretical\_mle = gamma.rvs(\*beds\_mle, size=len(beds))

qq(beds, beds\_theoretical\_mle, 'Hospital beds per thousand', 'MLE')

*# #### Least squares method*

*# In[47]:*

beds\_theoretical\_lsm = gamma.rvs(\*beds\_lsm, size=len(beds))

qq(beds, beds\_theoretical\_lsm, 'Hospital beds per thousand', 'LSM')

*# ### Excess mortality cumulative per million*

*# #### Maximum likelihood method*

*# In[48]:*

emcpm\_theoretical\_mle = chi2.rvs(\*emcpm\_mle, size=len(emcpm))

qq(emcpm, emcpm\_theoretical\_mle, 'Excess mortality cumulative per million', 'MLE')

*# #### Least squares method*

*# In[49]:*

emcpm\_theoretical\_lsm = chi2.rvs(\*emcpm\_lsm, size=len(emcpm))

qq(emcpm, emcpm\_theoretical\_lsm, 'Excess mortality cumulative per million', 'LSM')

*# ### Excess mortality cumulative*

*# #### Maximum likelihood method*

*# In[50]:*

emc\_theoretical\_mle = norm.rvs(\*emc\_mle, size=len(emc))

qq(emc, emc\_theoretical\_mle, 'Excess mortality cumulative', 'MLE')

*# #### Least squares method*

*# In[52]:*

emc\_theoretical\_lsm = norm.rvs(\*emc\_lsm, size=len(emc))

qq(emc, emc\_theoretical\_lsm, 'Excess mortality cumulative', 'LSM')

*# ## Step 7. Estimate correctness of fitted distributions using at least 2 statistical tests.*

*# In[34]:*

**def** ks\_test(data, distribution, params, alpha = 0.05, N = 50):

ks = stats.kstest(data.sample(N), distribution, params, N)

**if** ks[1] > alpha:

**print**(f'Kolmogorov test passed. Under the null hypothesis,**\n** the two {distribution} distributions are identical')

**else**:

**print**(f"Kolmogorov test failed. Hypothesis about {distribution} is false")

**print**(ks)

**def** cvm\_test(data, distribution, params, alpha = 0.05):

cvm = stats.cramervonmises(data.sample(50), distribution, params)

**if** cvm.pvalue > alpha:

**print**(f"Cramer-von-Mises test passed. Hypothesis about that distribution **\n** have cumulative {distribution} distribution is true")

**else**:

**print**(f"Cramer-von-Mises failed. We reject the null hypothesis **\n** that the observed sample is drawn from a {distribution} distribution")

**print**(cvm)

**print**('**\n**')

*# In[35]:*

**def** statistical\_tests(data, distribution, params, alpha = 0.05, N = 50):

ks\_test(data, distribution, params, alpha, N)

**print**('**\n**')

cvm\_test(data, distribution, params, alpha)

*# ### Hospital beds per thousand*

*# In[62]:*

statistical\_tests(beds, 'gamma', beds\_mle, alpha = 0.05, N = 50)

statistical\_tests(beds, 'gamma', beds\_lsm, alpha = 0.05, N = 50)

*# ### Excess mortality cumulative per million*

*# In[60]:*

statistical\_tests(emcpm, 'chi2', emcpm\_mle, alpha = 0.05, N = 50)

statistical\_tests(emcpm, 'chi2', emcpm\_lsm, alpha = 0.05, N = 50)

*# ### Excess mortality cumulative*

*# In[61]:*

statistical\_tests(emc, 'norm', emc\_mle, alpha = 0.05, N = 50)

statistical\_tests(emc, 'norm', emc\_lsm, alpha = 0.05, N = 50)