МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Статистические методы обработки экспериментальных данных»

Тема: Обработка экспериментальных данных

Студент гр. 5381 _____ Лянгузов А. А.

Преподаватель _____ Середа В. И.

Санкт-Петербург
2019

ЗАДАНИЕ
НА КУРСОВУЮ РАБОТУ

Студент Лянгузов А. А.

Группа 5381

Тема работы: Обработка экспериментальных данных

Исходные данные:
Генеральная совокупность объемом в 400 значений, представляющая два признака.

Содержание пояснительной записки:
«Содержание», «Введение», «Выравнивание статистических рядов», «Корреляционный и регрессионный анализ», «Кластерный анализ», «Заключение».

Предполагаемый объем пояснительной записки:
Не менее 50 страниц.

Дата выдачи задания: 02.04.2019

Дата сдачи реферата: 18.04.2019

Дата защиты реферата: 18.04.2019

Студент		Лянгузов А. А.
Преподаватель		Середа В. И.

АННОТАЦИЯ

Данная курсовая работа посвящена изучению методов статистической обработки данных. В ходе работы были изучены, реализованы на языке R и исследованы базовые алгоритмы обработки статистических данных небольшого объема.

Ключевые слова: математическая статистика, кластерный анализ, обработка данных, статистическая обработка данных, статистическая обработка данных в R.

СОДЕРЖАНИЕ

	Введение	4
1.	Выравнивание статистических рядов	5
1.1.	Теоретические сведения	6
1.2.	Первичная обработка выборки для первого признака	13
1.3.	Первичная обработка выборки для второго признака	23
1.4.	Выводы	31
2.	Корреляционный и регрессионный анализ	32
2.1.	Теоретические сведения	32
2.2.	Обработка и анализ двумерной выборки	37
2.3.	Выводы	44
3.	Кластерный анализ	46
3.1.	Теоретические сведения	46
3.2.	Кластеризация методом k-средних	52
3.3.	Кластеризация методом поиска сгущений	57
3.4.	Выводы	73
	Заключение	75
	Приложение А. Листинги с исходный кодом на языке R	76

ВВЕДЕНИЕ

Целью данной работы является изучение и освоение базовых способов обработки статистических данных с использованием языка программирования R.

Основными задачами для достижения цели стали: изучение теоретических сведений, необходимых для выполнения работы, формирование выборки заданного объема из предоставленной генеральной совокупности, разработка программного кода для первичной обработки выборки, получения статистических оценок, проверки простой гипотезы о согласии с нормальным законом распределения случайной величины по критерию Пирсона, для корреляционного, регрессионного и кластерного анализа, а также для иллюстрации промежуточных и конечных результатов.

В рамках регрессионного анализа рассматривается метод наименьших квадратов для построения уравнений прямой и параболы среднеквадратической регрессии.

В рамках кластерного анализа исследуются два метода кластеризации: метод к-средних (k-means) и метод поиска сгущений.

1. ВЫРАВНИВАНИЕ СТАТИСТИЧЕСКИХ РЯДОВ

1.1. Теоретические сведения

Генеральная совокупность – множество всех мыслимо возможных значений, относящихся к объекту исследуемой области.

Выборка – множество элементов из генеральной совокупности конечной длины.

Ранжированный ряд – упорядоченная по (неубыванию) последовательность элементов выборки.

Вариационный ряд - пара: упорядоченная последовательность уникальных значений выборки и соответствующие им частоты.

Интервальный ряд – пара: интервалы, на которые делится множество значений полученной выборки, и соответствующие им количества попаданий значений элементов выборки в каждый из интервалов.

Количество интервалов, на которые делится исходная выборка, можно определить с помощью формулы Стерджеса:

$$k = 1 + floor\left(log_2 N\right),\ где\ N - размер\ выборки\ .$$

Эмпирическая функция распределения для относительных частот определяется следующим образом:

$$F(x) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_{\{X_i \le x\}}$$

Функция $\mathbb{I}_{\{X_i \le x\}}$ – индикатор:

$$\mathbb{I}_A(x) = \begin{cases} 1, & x \in A, \\ 0, & x \notin A\ . \end{cases}$$

Для любого распределения *случайной величины* выделяют *параметры*: математическое ожидание, дисперсия, среднеквадратичное отклонение, асимметрия, эксцесс и другие. В терминах математической статистики вводят их аналоги – выборочные числовые характеристики (они же точечные статистические оценки соответствующих параметров распределения случайной

величины). Сами параметры распределения являются константами, но их оценки – это случайные величины (статистики), зависящие от выборки.

Пусть имеется выборка $X_1, \ldots, X_n$, тогда $\hat{\theta} = \hat{\theta}(X_1, \ldots, X_n)$ – *точечная оценка.*

Требования, предъявляемые к точечным оценкам:

1. Несмещенность – свойство точечной оценки, математическое ожидание которой совпадает с реальным значением параметра.

$$M\left(\hat{\theta}\right) = \theta$$

2. Эффективность – свойство оценки, имеющей наименьшую дисперсию среди всех возможных оценок.

$$\min_{\hat{\theta}} D(\hat{\theta})$$

3. Состоятельность – свойство оценки, сходящейся по вероятности к оцениваемому параметру.

$$\lim_{n\to\infty} P\left(\left|\theta - \hat{\theta}\right| < \varepsilon\right) = 1 .$$

Рассматриваемые в данной работе точечные оценки.

Здесь из исходной выборки сформирован интервальный (возможно также работать с вариационным) ряд: x_i - *варианта,* n_i *– абсолютная частота,* n *– объем выборки,* k *– количество интервалов.*

- Выборочное среднее (аналог мат.ожидания) – mean.

$$\bar{x_в} = \frac{1}{n}\sum_{i=1}^{k} x_i n_i$$

- Выборочная дисперсия – variance.

  Смещенная оценка:

$$D_в = \frac{1}{n}\sum_{i=1}^{k} (x_i - \bar{x_в})^2 n_i$$

  Исправленная выборочная дисперсия:

$$S^2 = \frac{n}{n-1} D_{в}$$

- Выборочное среднее квадратическое отклонение – deviation.

$$S = \sqrt[2]{S^2}$$

- Выборочная асимметрия – skewness.

$$A_{в} = \frac{1}{n * S^3} \sum_{i=1}^{k} (x_i - \bar{x_{в}})^3 n_i$$

- Выборочный эксцесс – kurtosis.

$$K_{в} = \frac{1}{n * S^4} \sum_{i=1}^{k} (x_i - \bar{x_{в}})^4 n_i - 3.$$

Для вычислений в работе используются условные варианты и начальные моменты.

Условная варианта u_i:

$$u_i = \frac{x_i - C}{h}.$$

В интервальном ряду $x_1, \ldots, x_k$ выбирается элемент C, располагающийся примерно посередине ряда, называемый ложным нулем, h – длина интервала.

Условный начальный момент порядка t:

$$M_t^* = \frac{1}{n} \sum_{i=1}^{k} u_i^t n_i .$$

Интервальная статистическая оценка параметра случайной величины представляет собой интервал, с определенной вероятностью покрывающий истинное значение оцениваемого параметра.

Пусть Q – оцениваемый параметр, тогда Q^* – его точечная оценка.

Для заданной надежности γ рассматривается следующая вероятность:

$$P\left(\left| Q^* - Q \right| < \delta \right) = \gamma, \; где \; \delta - точность .$$

Доверительным интервалом называют интервал $\left(Q^* - \delta;\; Q^* + \delta \right)$.

Можно также рассматривать нестрогое неравенство, тогда доверительный интервал будет закрытым.

Для *построения доверительного интервала для математического ожидания при неизвестном СКВО* рассматривается следующая статистика:

$$T = \frac{(\bar{x}_B - a)\sqrt{N}}{S}.$$

Здесь a – оцениваемый параметр (мат. ожидание), N – объем выборки; необходимо иметь точечную оценку математического ожидания $\bar{x}_B$, а также точечную оценку среднеквадратического отклонения S.

Утверждается (теорема Фишера), что данная статистика T имеет распределение Стьюдента с (N-1) степенью свободы: $T \sim St_{N-1}$.

Рассмотрим вероятность:

$$\mathbb{P}\left(\left| \frac{(\bar{x}_B - a)\sqrt{N}}{S} \right| \leq t_\gamma \right) = \gamma;$$

$$-t_{\frac{1+\gamma}{2}} \leq \frac{(\bar{x}_B - a)\sqrt{N}}{S} \leq t_{\frac{1+\gamma}{2}};$$

формула для интервальной оценки:

$$\bar{x}_B - \frac{S}{\sqrt{N}} t_{\frac{1+\gamma}{2}} \leq a \leq \bar{x}_B + \frac{S}{\sqrt{N}} t_{\frac{1+\gamma}{2}}.$$

Выбрав доверительную вероятность γ, мы можем вычислить соответствующую $\left(\frac{1+\gamma}{2}\right)$-*квантиль* распределения Стьюдента с *(N-1)* степенью свободы - $t_{\frac{1+\gamma}{2}}$, в силу симметричности распределения Стьюдента и наличия модуля мы берем это значение вместо *γ-кватили*.

Итого доверительный интервал для математического ожидания:

$$a \in \left[\bar{x_B} - \frac{S}{\sqrt{N}} t_{\frac{1+\gamma}{2}}; \bar{x_B} + \frac{S}{\sqrt{N}} t_{\frac{1+\gamma}{2}} \right].$$

Для *нахождения доверительного интервала для СКВО при неизвестном математическом ожидании* случайной величины рассматривается следующая статистика:

$$H = \frac{(N-1)D_B}{\sigma^2} \sim \chi^2_{N-1}$$

Эта случайная величина распределена как хи-квадрат со степенью свободы *N-1*, где N – объем выборки, D_B – исправленная выборочная дисперсия, а σ – это настоящее значение СКВО (оцениваемый параметр).

Для заданной надежности γ находятся соответствующие $\frac{1-\gamma}{2}$- и $\frac{1+\gamma}{2}$-квантили $\chi^2_{\frac{1-\gamma}{2},\ N-1}$ и $\chi^2_{\frac{1+\gamma}{2},\ N-1}$: $\mathbb{P}\left(\chi^2_{\frac{1-\gamma}{2},\ N-1} \le \frac{(N-1)D_B}{\sigma^2} \le \chi^2_{\frac{1+\gamma}{2},\ N-1}\right) = \gamma$.

После преобразований получаем итоговый доверительный интервал для СКВО:

$$\sigma \in \left[\sqrt{\frac{(N-1)D_в}{\chi^2_{N-1}\left(\frac{1+\gamma}{2}\right)}}; \sqrt{\frac{(N-1)D_в}{\chi^2_{N-1}\left(\frac{1-\gamma}{2}\right)}}\right].$$

Статистическая гипотеза – это утверждение, которое можно проверить, используя статистическую модель.

По результатам проверки гипотеза либо принимается, либо отвергается, это решение может быть ошибочным. Различают *ошибки первого и второго рода*. Ошибкой первого рода (ложноположительная) называют непринятие гипотезы, которая на самом деле верна. Ошибкой второго рода (ложноотрицательная) называют принятие нулевой гипотезы, которая на самом деле неверна.

		Нулевая гипотеза на самом деле	
		верна	неверна
Нулевая гипотеза была	принята	Верное решение (Вероятность 1-α)	Ошибка II рода (Вероятность ошибки β)
	отвергнута	Ошибка I рода (Вероятность ошибки α – **уровень значимости**)	Верное решение (Вероятность 1-β – **мощность критерия**)

Критерий Пирсона (хи-квадрат) и проверка простой гипотезы о нормальном распределении.

В общем случае критерий (тест) Пирсона хи-квадрат – это метод, позволяющий вычислить вероятность того, что наблюдаемое различие между двумя наборами группированных данных возникло случайно. Он используется для проверки нулевой гипотезы о том, что распределение частоты определенных событий, наблюдаемых в выборке, согласуется с конкретным теоретическим распределением. Важным условием, накладываемым на события, является их независимость друг от друга.

Для проверки простой гипотезы о нормальном распределении вводится следующая статистика:

$$X^2 = \sum_{i=1}^{K} \frac{\left(n_i - n_i'\right)^2}{n_i'}.$$

Здесь K – количество групп, на которые разбита выборка; n_i и n_i' – абсолютная и теоретическая частоты *i-ой* группы, соответственно.

Предполагается, что данная статистика имеет распределение хи-квадрат с *(K-m-1)* степенью свободы, где m – количество параметров распределения (для нормального m = 2). Необходимо выбрать уровень значимости α и для него вычислить критическое значение как (1-α)-квантиль распределения хи-квадрат с (K-3) степенями свободы.

После вычисления X^2 сравнивается с критическим значением: если X^2 превосходит критическое значение, нулевая гипотеза отвергается (при заданном уровне значимости), иначе – принимается (при заданном уровне значимости).

1.2. Первичная обработка выборки для первого признака

1) Была сформирована выборка объема N = 107.

v	501.00	369.00	344.00	473.00	426.00	528.00	497.00	467.00	506.00	431.00	454.00
E	130.40	84.30	86.80	137.90	121.10	163.40	147.30	140.50	158.40	125.00	131.10
v	371.00	482.00	393.00	441.00	463.00	440.00	481.00	340.00	468.00	397.00	496.00
E	89.20	139.90	103.20	122.80	129.10	128.50	135.20	85.10	142.00	108.60	143.10
v	434.00	541.00	352.00	438.00	453.00	423.00	351.00	525.00	409.00	469.00	386.00
E	122.30	146.80	87.70	134.90	119.50	131.10	89.00	165.90	121.00	131.50	95.50
v	505.00	436.00	488.00	449.00	493.00	512.00	472.00	423.00	465.00	351.00	359.00
E	137.50	114.30	134.10	124.50	129.70	169.90	134.20	130.80	140.70	102.90	71.90
v	457.00	467.00	400.00	418.00	492.00	434.00	510.00	392.00	463.00	459.00	397.00
E	126.40	135.10	114.60	118.60	137.50	110.50	140.60	82.70	125.00	145.40	106.80
v	424.00	436.00	429.00	398.00	493.00	522.00	518.00	463.00	437.00	386.00	493.00
E	119.00	116.70	112.90	109.00	154.50	154.50	144.40	121.20	121.80	105.80	151.20
v	414.00	480.00	585.00	562.00	508.00	421.00	463.00	422.00	406.00	544.00	345.00
E	113.50	153.90	177.70	175.90	159.00	117.80	136.70	122.90	110.10	166.70	95.90
v	478.00	393.00	437.00	448.00	458.00	422.00	468.00	430.00	371.00	543.00	471.00
E	126.60	122.80	115.10	121.90	121.70	115.70	144.90	104.30	91.90	155.40	143.90
v	475.00	521.00	353.00	437.00	362.00	490.00	484.00	459.00	480.00	482.00	522.00
E	132.00	139.60	98.00	118.40	111.70	139.90	140.40	136.70	153.30	148.20	143.80
v	576.00	390.00	514.00	442.00	421.00	443.00	438.00	429.00			
E	166.40	91.40	153.60	115.40	107.90	121.90	126.70	120.90			

2) Ранжированный ряд (по строкам).

340	344	345	351	351	352	353	359	362	369
371	371	386	386	390	392	393	393	397	397
398	400	406	409	414	418	421	421	422	422

423	423	424	426	429	429	430	431	434	434
436	436	437	437	437	438	438	440	441	442
443	448	449	453	454	457	458	459	459	463
463	463	463	465	467	467	468	468	469	471
472	473	475	478	480	480	481	482	482	484
488	490	492	493	493	493	496	497	501	505
506	508	510	512	514	518	521	522	522	525
528	541	543	544	562	576	585			

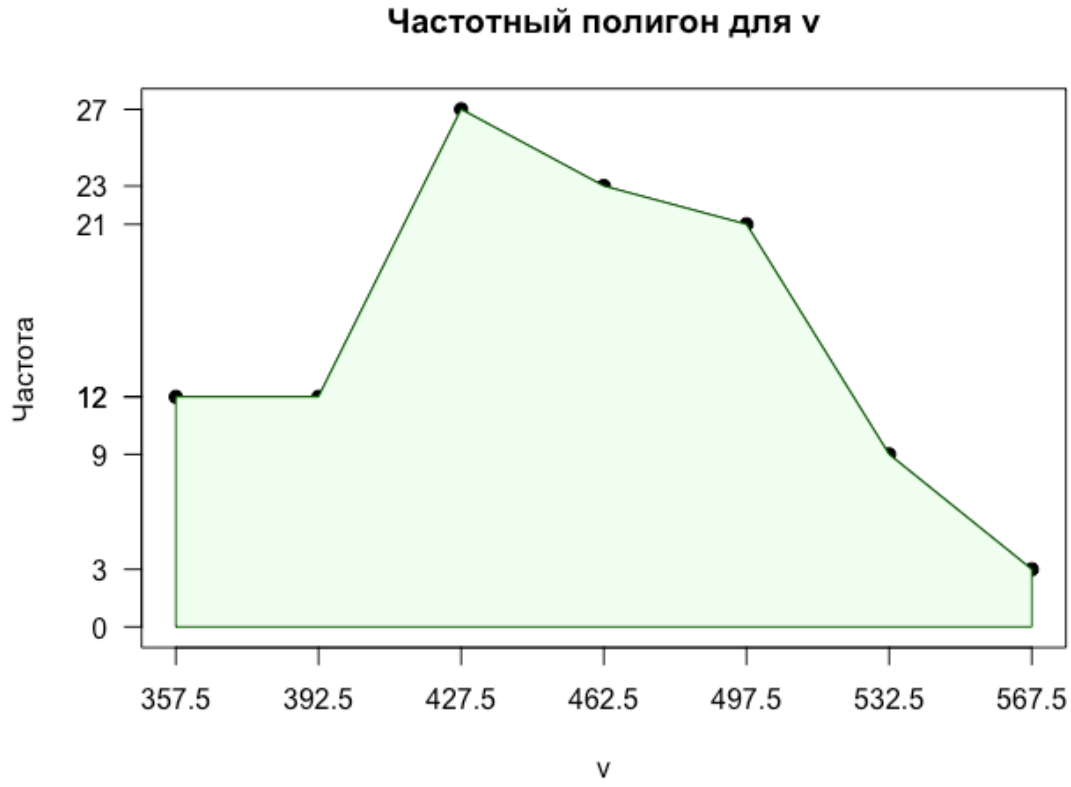
3) Вариационный ряд (значения упорядочены в столбец).

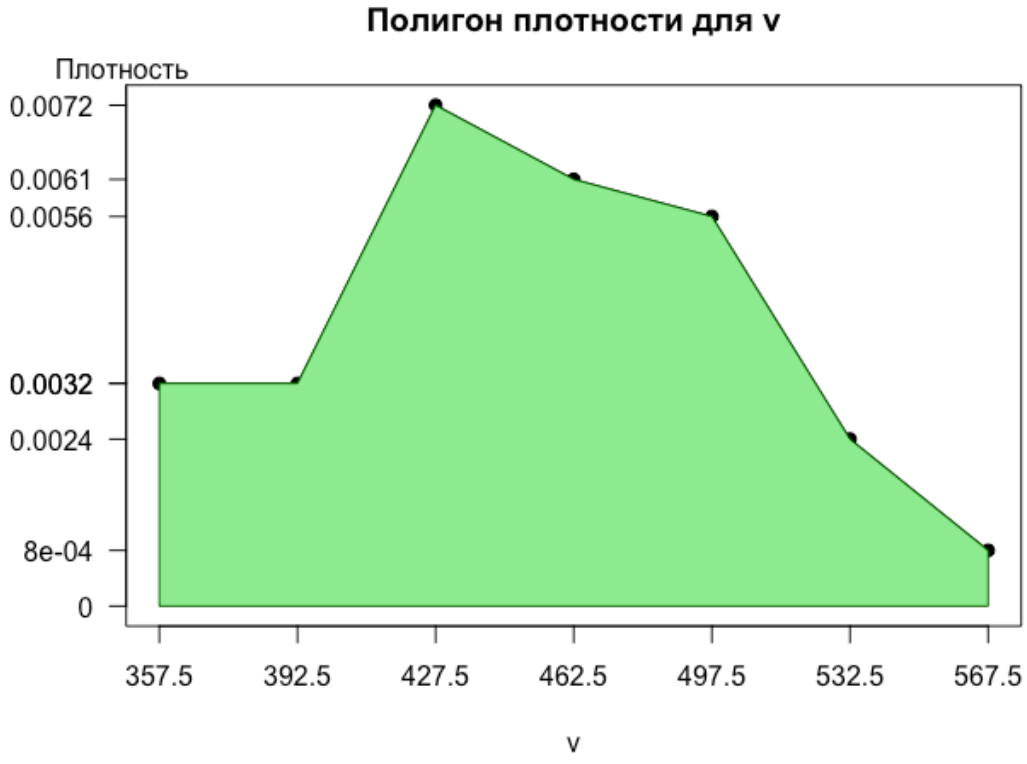
v	count	v	count	v	count	v	count	v	count
340	1	406	1	441	1	473	1	510	1
344	1	409	1	442	1	475	1	512	1
345	1	414	1	443	1	478	1	514	1
351	2	418	1	448	1	480	2	518	1
352	1	421	2	449	1	481	1	521	1
353	1	422	2	453	1	482	2	522	2
359	1	423	2	454	1	484	1	525	1
362	1	424	1	457	1	488	1	528	1
369	1	426	1	458	1	490	1	541	1
371	2	429	2	459	2	492	1	543	1
386	2	430	1	463	4	493	3	544	1
390	1	431	1	465	1	496	1	562	1
392	1	434	2	467	2	497	1	576	1
393	2	436	2	468	2	501	1	585	1
397	2	437	3	469	1	505	1		
398	1	438	2	471	1	506	1		
400	1	440	1	472	1	508	1		

4) Интервальный ряд. По формуле Стёрджеса было сформировано 7 интервалов. Длина интервала ≈ 35.

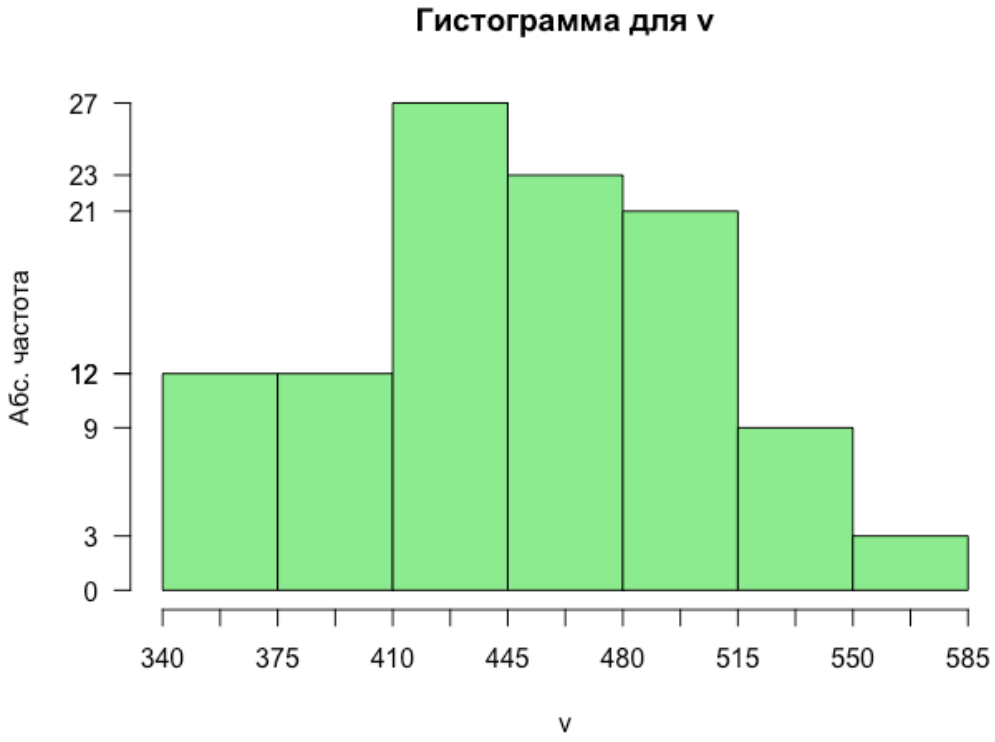
Интервалы	Частоты	Середины
[340; 375)	12	358
[375; 410)	12	392
[410; 445)	27	428
[445; 480)	23	462
[480; 515)	21	498
[515; 550)	9	532
[550; 585)	3	568

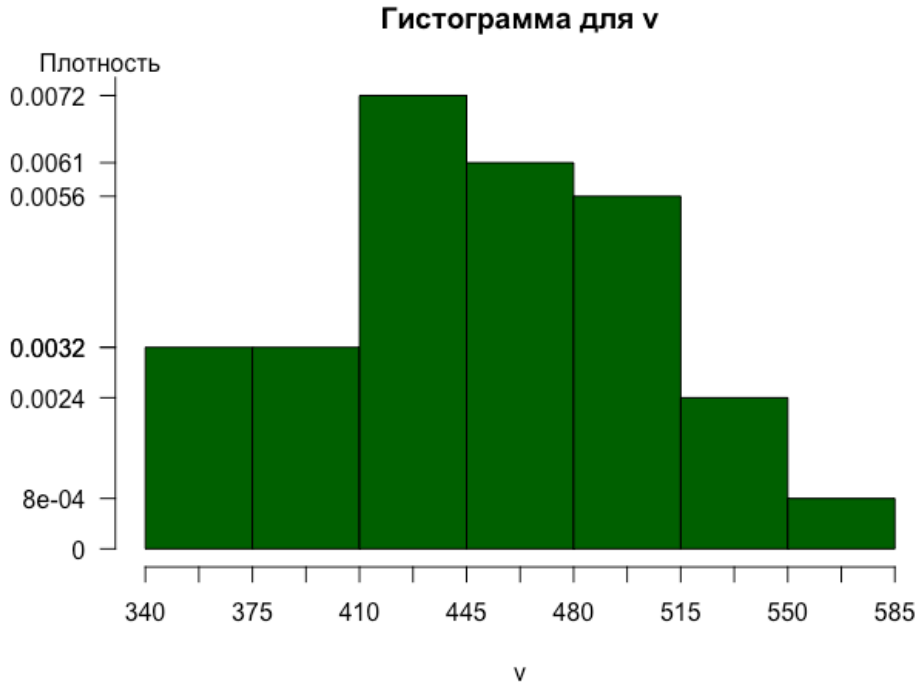
5) Полигон. В качестве вершин полигона взяты середины интервалов.





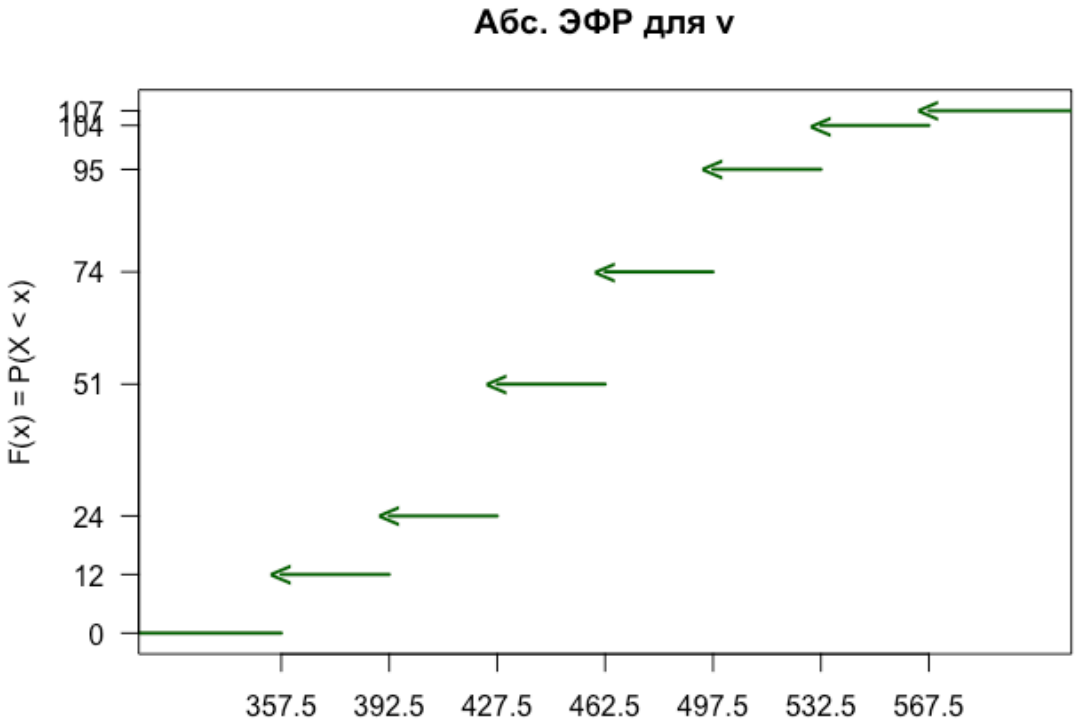
6) Гистограммы.

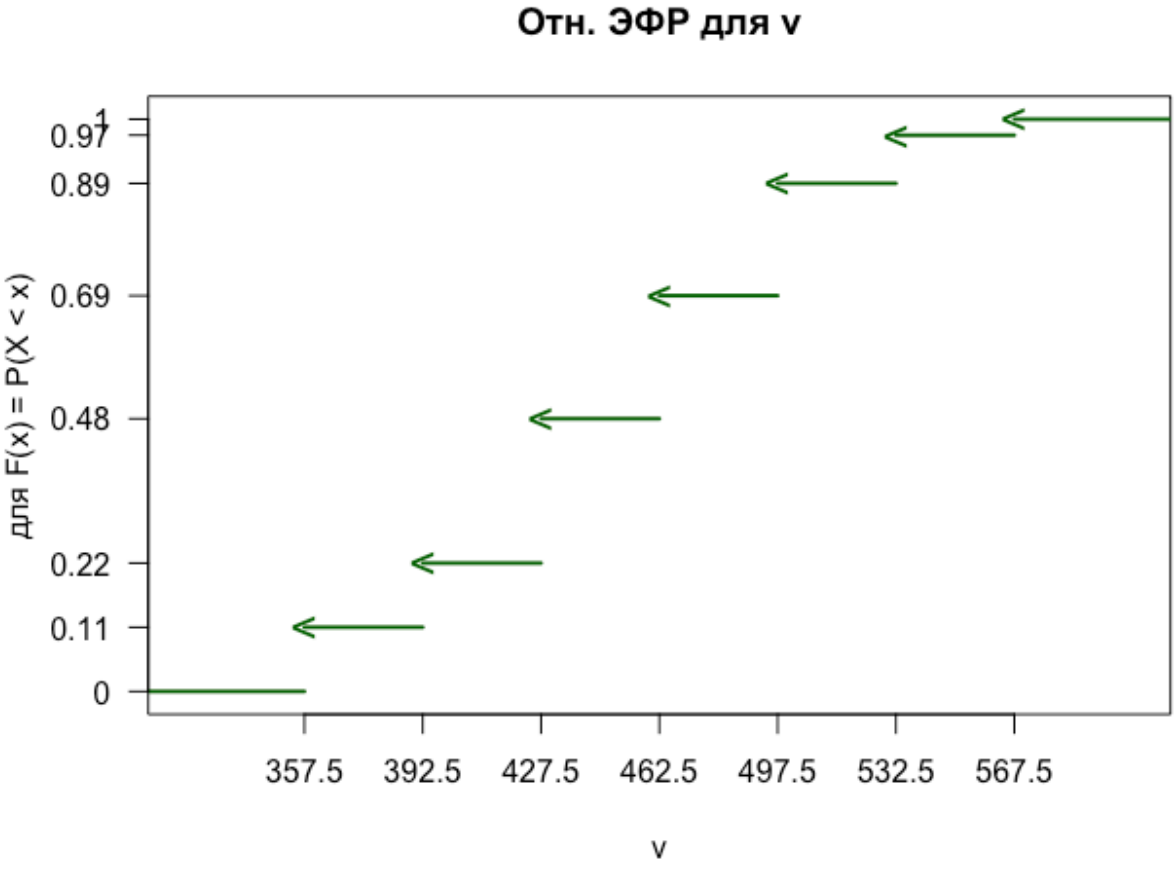




7) Эмпирические функции распределения.

Для $F(x) = \mathbb{P}(X < x)$





8) Условные варианты и условные начальные моменты.

В качестве ложного нуля был выбран четвертый интервал, который находится ровно посередине. Чтобы вычислить первые четыре условных начальных момента были посчитаны их слагаемые и занесены в следующие таблицы. Тогда условные моменты находятся как суммы по столбцам M1-M4.

$C = 462.5$

x_i	n_i	$\tilde{n}_i$	u_i	$u_i * \tilde{n}_i$	$u_i^2 * \tilde{n}_i$	$u_i^3 * \tilde{n}_i$	$u_i^4 * \tilde{n}_i$	$(u_i+1)^4 * \tilde{n}_i$
357.5	12	0.11214	-3	-0.336448	1.009345	-3.02803	9.08411	1.7943925
392.5	12	0.11214	-2	-0.224299	0.448598	-0.89719	1.79439	0.1121495
427.5	27	0.25233	-1	-0.252336	0.252336	-0.25233	0.25233	0.0000000
462.5	23	0.21495	0	0	0	0	0	0.2149533
497.5	21	0.19626	1	0.196261	0.196261	0.196261	0.19626	3.1401869
532.5	9	0.08411	2	0.168224	0.336449	0.672897	1.34579	6.8130841
567.5	3	0.02803	3	0.084112	0.252336	0.757009	2.27102	7.1775701

M_1	M_2	M_3	M_4	$\sum (u_i+1)^4 * \tilde{n}_i$
-0.36448598	2.49532710	-2.55140187	14.94392523	19.25233643

9) Выборочное среднее.

$$\bar{X}_в = M_1^* h + C, \quad где$$

$$h - длина\ интервала,$$

$$C - ложный\ ноль,$$

$$M_1^* - усл.\ нач.\ момент\ 1\ порядка.$$

$$\bar{x}_в = 449.74299065.$$

10) Исправленная выборочная дисперсия.

$$D_в = \frac{N}{N-1}\left(M_2^* - M_1^{*2}\right)h^2, \quad где\ N - объем\ выборки,$$

$$M_i^* - условный\ начальный\ момент\ i го\ порядка.$$

$$D_в = 2921.33662493.$$

11) Среднее квадратическое выборочное отклонение.

$$S = \sqrt[2]{D_в};$$

$$S = 54.04939061.$$

12) Выборочная асимметрия.

$$A = \left(M_3^* - 3M_2^*M_1^* + 2M_1^{*3}\right)\left(\frac{h}{S}\right)^3;$$

$$A = 0.02180169.$$

13) Выборочный эксцесс.

$$K = \left(M_4^* - 4M_3^*M_1^* + 6M_2^*M_1^{*2} - 3M_1^{*4}\right)\left(\frac{h}{S}\right)^4 - 3$$

$$K = -0.68595634.$$

14) Интервальная оценка для математического ожидания.

Зададим надежность оценки $\gamma = 0.95$.

Далее вычислим значение t_γ как квантиль распределения Стьюдента с N-1 степенью свободы, где N – объем выборки:

$$\mathbb{P}\left(\left|\frac{(\bar{x}_B - a)\sqrt{N}}{S}\right| \le t_\gamma\right) = \gamma$$

Поскольку распределение Стьюдента симметричное и оценка двусторонняя, вычисляем *(1+y)/2 (или –(1-y)/2)* квантиль. (Вычисление проведено с помощью языка программирование R).

Значение $t_{\frac{1+\gamma}{2}} = 1.983.$

Далее доверительные интервалы строятся по формулам:

$$a \in \left[\bar{x_B} - \frac{S}{\sqrt{N}}t_{\frac{1+\gamma}{2}}; \bar{x_B} + \frac{S}{\sqrt{N}}t_{\frac{1+\gamma}{2}}\right].$$

Тогда доверительный интервал:

$$\bar{x_v} \in [439;\ 460].$$

15) Интервальная оценка для среднеквадратического отклонения.

Зададим надежность оценки $\gamma = 0.95$.

Формулы для вычисления доверительного интервала для СКВО:

$$a \in \left[\sqrt{\frac{(N-1)D_в}{\chi^2_{N-1}\left(\frac{1+\gamma}{2}\right)}}; \sqrt{\frac{(N-1)D_в}{\chi^2_{N-1}\left(\frac{1-\gamma}{2}\right)}} \right].$$

Здесь $\chi^2_{N-1}(r)$ – это *r-квантиль* распределения хи-квадрат со степенью свободы N-1; $D_в$ – исправленная выборочная дисперсия; N – объем выборки.

С помощью языка R получим значения для квантилей:

$$\chi^2_{N-1}\left(\frac{1+\gamma}{2}\right) = 136.4;$$

$$\chi^2_{N-1}\left(\frac{1-\gamma}{2}\right) = 79.4\,.$$

Тогда доверительный интервал:

$$S_v \in [47.7;\ 62.4].$$

Примечание для полученных интервалов: для обоих параметров использовалось определение функции вероятности с нестрогим неравенством: $\mathbb{P}(X \le x) = \alpha$, поэтому доверительные интервалы представляют собой *замкнутые* интервалы.

16) Проверка простой гипотезы о нормальном распределении с помощью критерия Пирсона χ2.

Нулевая гипотеза H_0: $v_1, v_2, \ldots, v_N \sim \mathcal{N}(\bar{x}_в, D_в)$ – для величины v.

Идея состоит в том, чтобы для имеющегося интервального ряда вычислить значение хи-квадрат и сравнить его с критическим значением,

которое берется как квантиль распределения хи-квадрат с *(K-3)* степенями свободы, где K – количество интервалов. Если вычисленное по интервальному ряду значение превосходит критическое, то гипотеза отклоняется на данном уровне значимости.

Выберем уровень значимости α = 0.05.

Тогда критическое значение:

$$X^2_{кр} = \chi^2_{K-3}(1-\alpha) = \chi^2_4(0.95) = 9.487729.$$

Для вычисления интервального хи-квадрат составим таблицу.

F_{th} – функция распределения для нормального закона с мат. ожиданием $\bar{x_в} = 449.74299065$ и дисперсией $D_в = 2921.33662493$. Тогда пятая и шестая колонки в таблице содержат соответствующие значения функции распределения в точках левой (leftB) и правой (rightB) границ интервалов. Эти значения были получены с помощью языка R (встроенная функция pnorm).

Седьмой столбец таблицы содержит разницу между значениями шестой и пятой столбцов, а именно вероятность попадания в данный интервал.

Последний столбец вычисляется по формуле

$$n'_i = \left(F_{th}\left(rightB_i\right) - F_{th}\left(leftB_i\right)\right) * N,$$

где N – объем выборки, i – номер интервала, и показывает, сколько «попаданий» (абсолютная частота) элементов выборки объема N должно быть в данном интервале, если элементы выборки действительно распределены по нормальному закону с заданными параметрами – теоретические частоты.

Интервалы	Середины	Частоты абс.	Частоты отн.	$F_{th}LB$	$F_{th}RB$	$F_{th}RB - F_{th}LB$	Теор. частоты
[340; 375)	357,5	12	0.11214953	0.00000000	0.08335319	0.08335319	8.918791
[375; 410)	392,5	12	0.11214953	0.08335319	0.23107574	0.14772255	15.806313
[410; 445)	427,5	27	0.25233645	0.23107574	0.46503654	0.23396080	25.033806
[445; 480)	462,5	23	0.21495327	0.46503654	0.71219308	0.24715654	26.445750
[480; 515)	497,5	21	0.19626168	0.71219308	0.88635301	0.17415993	18.635112
[515; 550)	532,5	9	0.08411215	0.88635301	0.96819580	0.08184279	8.757178
[550; 585)	567,5	3	0.02803738	0.96819580	1.00000000	0.03180420	3.403050

Тогда значение хи-квадрат:

$$\chi^2 = \sum_{i}^{K} \frac{\left(freq_i - n_i'\right)^2}{n_i'};$$

$$\chi^2 = 2.939051.$$

Полученное значение хи-квадрат меньше критического ($X^2_{кр} = 9.487729$), *следовательно нулевая гипотеза принимается при уровне значимости* $\alpha = 0.05$.

1.3. Первичная обработка выборки для второго признака

1) Ранжированный ряд (по строкам).

71.9	82.7	84.3	85.1	86.8	87.7	89.0	89.2	91.4	91.9
95.5	95.9	98.0	102.9	103.2	104.3	105.8	106.8	107.9	108.6
109.0	110.1	110.5	111.7	112.9	113.5	114.3	114.6	115.1	115.4
115.7	116.7	117.8	118.4	118.6	119.0	119.5	120.9	121.0	121.1
121.2	121.7	121.8	121.9	121.9	122.3	122.8	122.8	122.9	124.5
125.0	125.0	126.4	126.6	126.7	128.5	129.1	129.7	130.4	130.8
131.1	131.1	131.5	132.0	134.1	134.2	134.9	135.1	135.2	136.7
136.7	137.5	137.5	137.9	139.6	139.9	139.9	140.4	140.5	140.6
140.7	142.0	143.1	143.8	143.9	144.4	144.9	145.4	146.8	147.3

148.2	151.2	153.3	153.6	153.9	154.5	154.5	155.4	158.4	159.0
163.4	165.9	166.4	166.7	169.9	175.9	177.7			

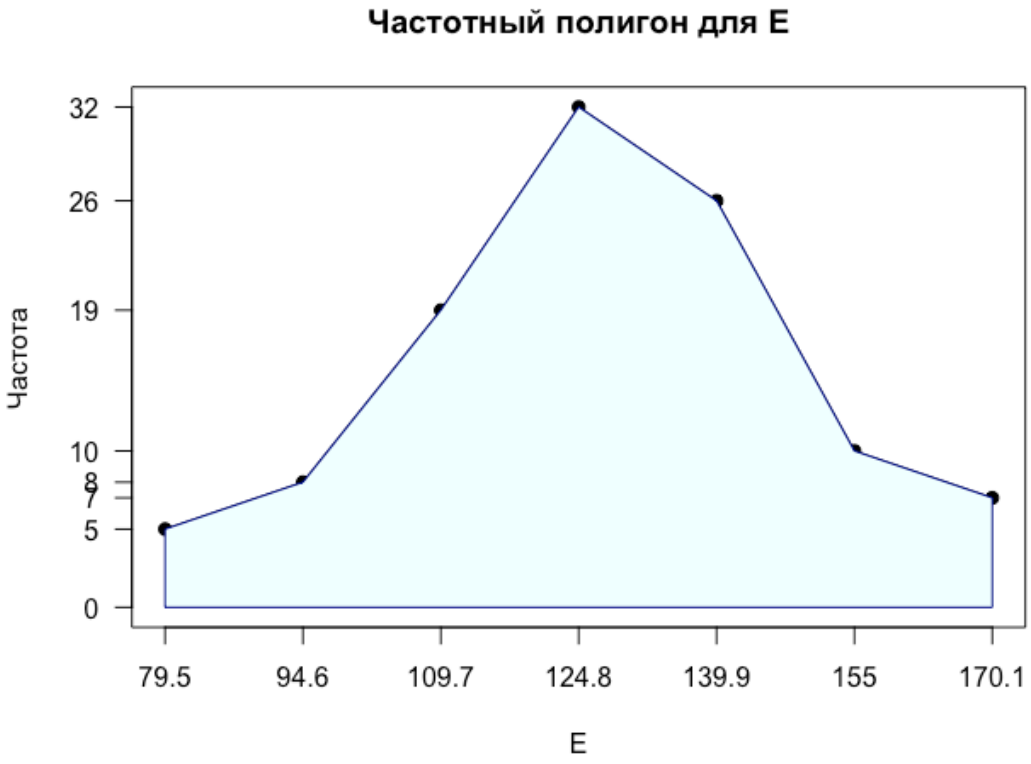
2) Вариационный ряд (значения упорядочены в столбец).

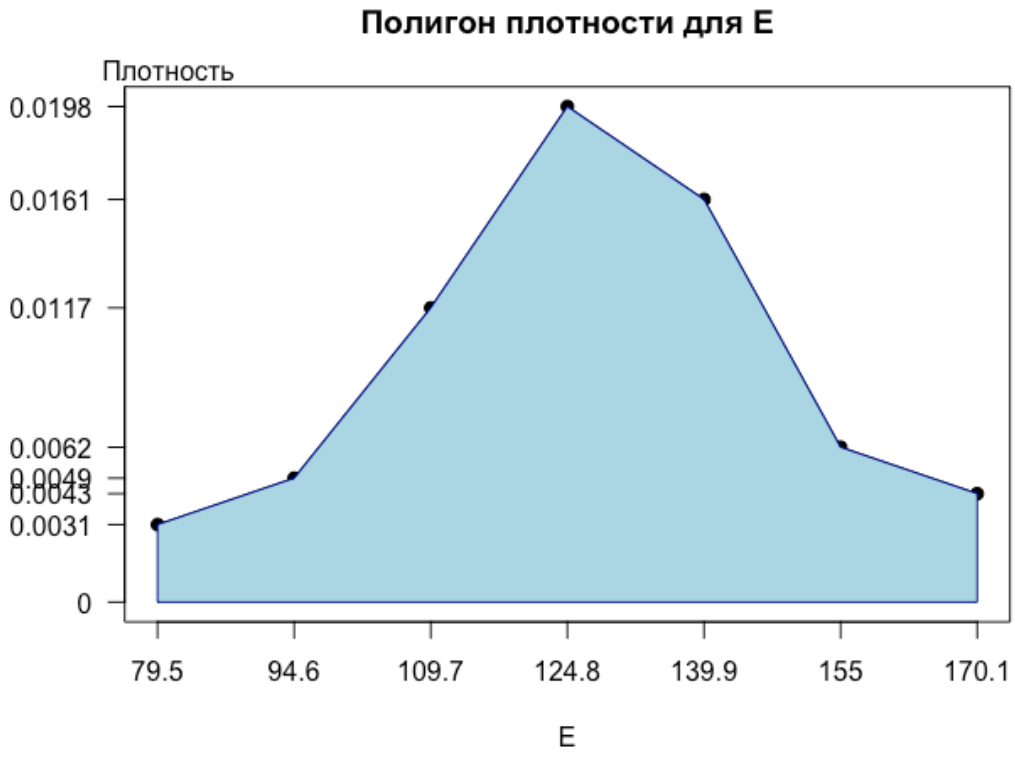
E	count	E	count	E	count	E	count	E	count
71.9	1	109.0	1	121.2	1	134.1	1	145.4	1
82.7	1	110.1	1	121.7	1	134.2	1	146.8	1
84.3	1	110.5	1	121.8	1	134.9	1	147.3	1
85.1	1	111.7	1	121.9	2	135.1	1	148.2	1
86.8	1	112.9	1	122.3	1	135.2	1	151.2	1
87.7	1	113.5	1	122.8	2	136.7	2	153.3	1
89.0	1	114.3	1	122.9	1	137.5	2	153.6	1
89.2	1	114.6	1	124.5	1	137.9	1	153.9	1
91.4	1	115.1	1	125.0	2	139.6	1	154.5	2
91.9	1	115.4	1	126.4	1	139.9	2	155.4	1
95.5	1	115.7	1	126.6	1	140.4	1	158.4	1
95.9	1	116.7	1	126.7	1	140.5	1	159.0	1
98.0	1	117.8	1	128.5	1	140.6	1	163.4	1
102.9	1	118.4	1	129.1	1	140.7	1	165.9	1
103.2	1	118.6	1	129.7	1	142.0	1	166.4	1
104.3	1	119.0	1	130.4	1	143.1	1	166.7	1
105.8	1	119.5	1	130.8	1	143.8	1	169.9	1
106.8	1	120.9	1	131.1	2	143.9	1	175.9	1
107.9	1	121.0	1	131.5	1	144.4	1	177.7	1
108.6	1	121.1	1	132.0	1	144.9	1		

3) Интервальный ряд. По формуле Стёрджеса было сформировано 7 интервалов. Длина интервала ≈ 15.1.

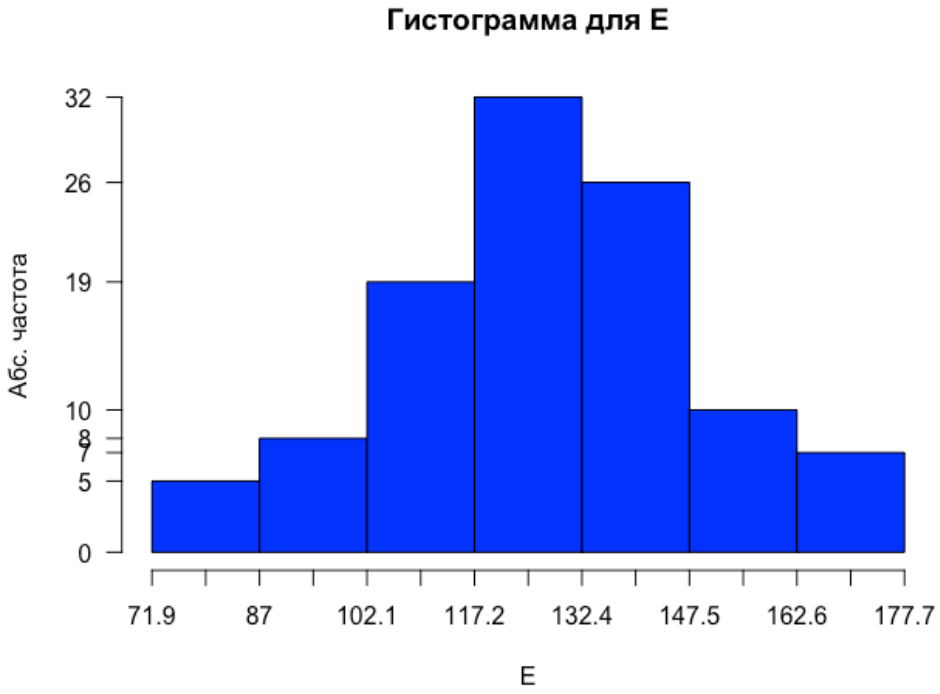
Интервалы	Частоты	Середины
[71.9,87)	5	79.5
[87,102)	8	94.6
[102,117)	19	110
[117,132)	32	125
[132,147)	26	140
[147,163)	10	155
[163,178]	7	170

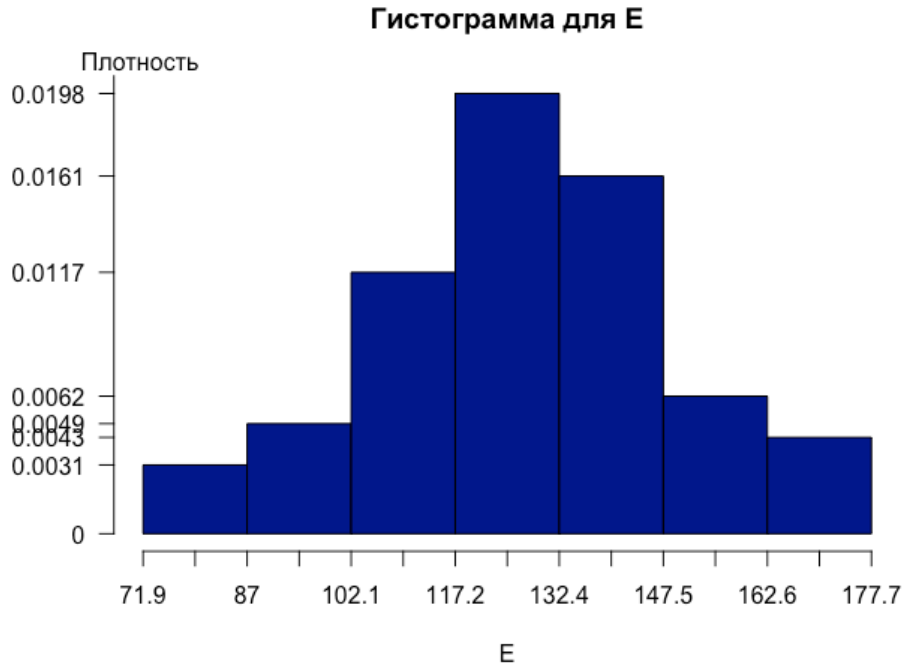
4) Полигон. В качестве вершин полигона взяты середины интервалов.





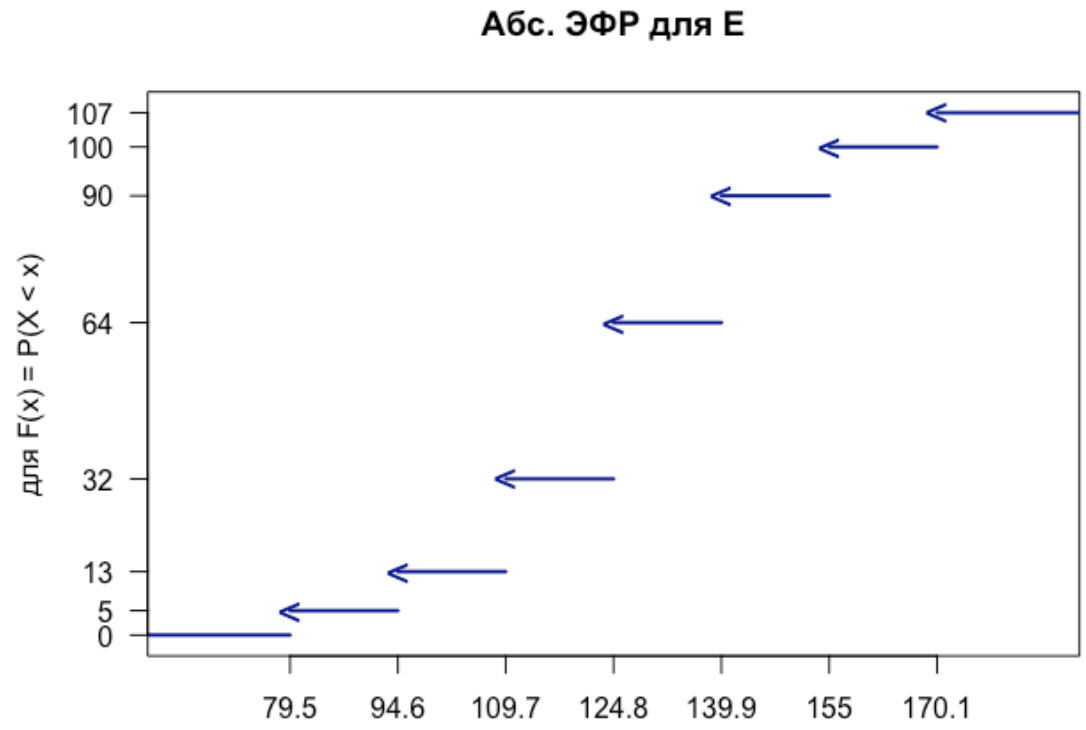
5) Гистограммы.

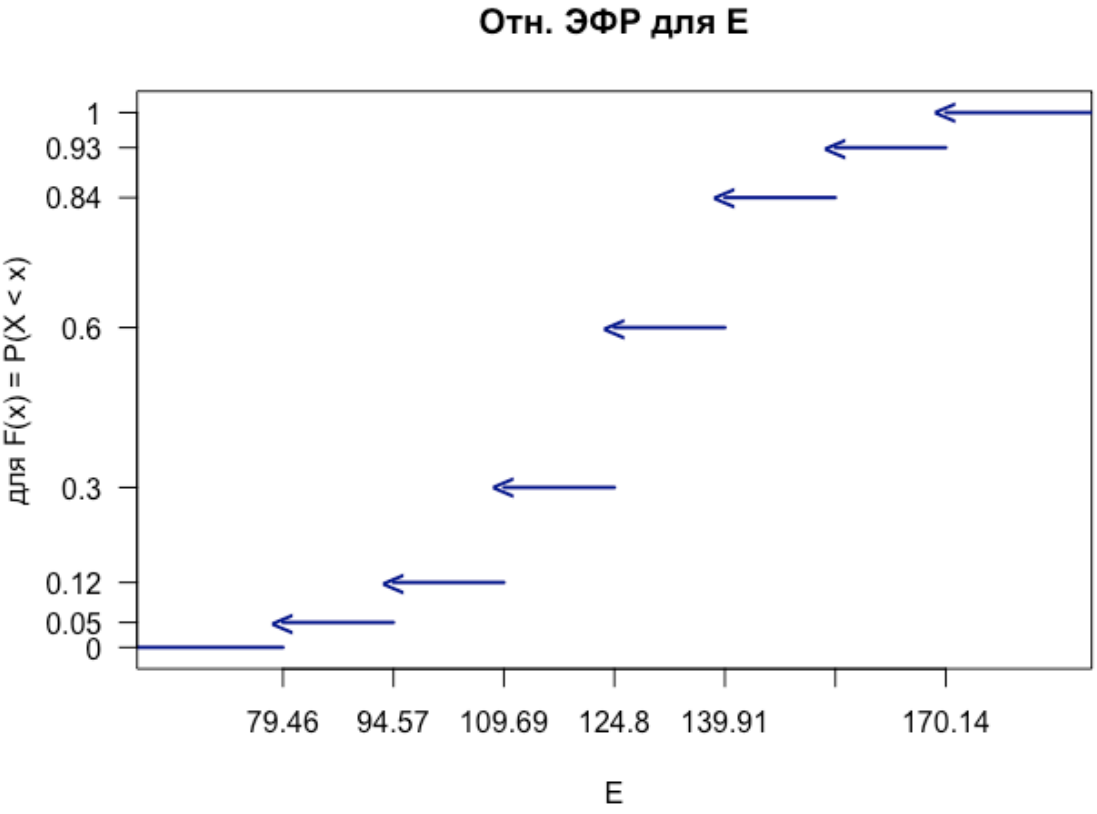




6) Эмпирические функции распределения.

Для $F(x) = \mathbb{P}(X < x)$





7) Условные варианты и условные начальные моменты.

$C = 124.80$

x_i	n_i	$\tilde{n}_i$	u_i	$u_i * \tilde{n}_i$	$u_i^2 * \tilde{n}_i$	$u_i^3 * \tilde{n}_i$	$u_i^4 * \tilde{n}_i$	$(u_i + 1)^4 * \tilde{n}_i$
79.45714	5	0.04672897	-3	-0.1401869	0.4205607	-1.2616822	3.7850467	7.476636e-01
94.57143	8	0.07476636	-2	-0.1495327	0.2990654	-0.5981308	1.1962617	7.476636e-02
109.68571	19	0.17757009	-1	-0.1775701	0.1775701	-0.1775701	0.1775701	6.477433e-62
124.80000	32	0.29906542	0	0.0000000	0.0000000	0.0000000	0.0000000	2.990654e-01
139.91429	26	0.24299065	1	0.2429907	0.2429907	0.2429907	0.2429907	3.887850e+00
155.02857	10	0.09345794	2	0.1869159	0.3738318	0.7476636	1.4953271	7.570093e+00
170.14286	7	0.06542056	3	0.1962617	0.5887850	1.7663551	5.2990654	1.674766e+01

M_1	M_2	M_3	M_4	$\sum (u_i + 1)^4 * \tilde{n}_i$
0.15887850	2.10280374	0.71962617	12.19626168	29.3271

8) Выборочное среднее.

$$\bar{X}_в = 127.20133511.$$

9) Исправленная выборочная дисперсия.

$$D_в = 479.07888153.$$

10) Среднее квадратическое выборочное отклонение.

$$S = 21.88787065.$$

11) Выборочная асимметрия.

$$A = -0.09042543.$$

12) Выборочный эксцесс.

$$K = -0.25892102.$$

13) Интервальная оценка для математического ожидания.

Надежность оценки $\gamma = 0.95$.

Доверительный интервал:

$$\bar{x_E} \in [123;\ 131].$$

14) Интервальная оценка для среднеквадратического отклонения.

Надежность оценки $\gamma = 0.95$.

Тогда доверительный интервал:

$$\bar{x_E} \in [19.3;\ 25.3].$$

Примечание для полученных интервалов: для обоих параметров использовалось определение функции вероятности с нестрогим неравенством: $\mathbb{P}(X \leq x) = \alpha$, поэтому доверительные интервалы представляют собой *замкнутые* интервалы.

15) Проверка простой гипотезы о нормальном распределении с помощью критерия Пирсона $\chi 2$.

Нулевая гипотеза H_0: $e_1, e_2, \ldots, e_N \sim \mathcal{N}(\bar{x}_в, D_в)$.

Проделаем все то же самое, что для первого параметра *(см. раздел 1.2 пункт 16)*.

Интервалы	Середины	Частоты абс.	Частоты отн.	$F_{th}LB$	$F_{th}RB$	$F_{th}RB - F_{th}LB$	Теор. частоты
[71.9,87)	79.45714	5	0.04672897	0.00000000	0.03317572	0.03317572	3.549802
[87,102)	94.57143	8	0.07476636	0.03317572	0.12599908	0.09282336	9.932099
[102,117)	109.68571	19	0.17757009	0.12599908	0.32456289	0.19856381	21.246327
[117,132)	124.80000	32	0.29906542	0.32456289	0.59311119	0.26854830	28.734668
[132,147)	139.91429	26	0.24299065	0.59311119	0.82279986	0.22968868	24.576689
[147,163)	155.02857	10	0.09345794	0.82279986	0.94701988	0.12422002	13.291542
[163,178]	170.14286	7	0.06542056	0.94701988	1.00000000	0.05298012	5.668873

Тогда значение хи-квадрат:

$$\chi^2 = \sum_i^K \frac{\left(freq_i - n_i'\right)^2}{n_i'};$$

$$\chi^2 = 2.786982.$$

Полученное значение хи-квадрат меньше критического ($X^2_{кр} = 9.487729$)*, следовательно нулевая гипотеза принимается при уровне значимости* $\alpha = 0.05$.

1.4. Выводы

Из генеральной совокупности была сформирована выборка по двум признакам и произведена ее первичная обработка: были сформированы ранжированный, вариационный и интервальный ряды. Для интервального ряда были построены гистограммы, полигоны и эмпирические функции распределения для абсолютных и относительных частот.

Графики для абсолютных и относительных частот выглядят идентично (кроме оси ординат), поскольку они отображают одно и то же соотношение.

Были получены точечные оценки параметров распределения случайных величин: выборочные среднее, исправленная дисперсия, среднеквадратичное отклонение, асимметрия и эксцесс.

Полученные выборочные средние по значению очень близки к соответствующим медианам. Для величины v асимметрия положительна, на ее гистограммах можно увидеть более крутой левый бок и более пологий правый. Для величины E, асимметрия оказалась отрицательной, это говорит о том, что левый бок на гистограммах более пологий, чем правый, что можно увидеть из соответствующих графиков. У обеих величин эксцесс оказался отрицательным, т.е. их пик более пологий, чем у нормального распределения.

Были получены интервальные оценки параметров распределения случайной величины: выборочные средние и среднеквадратичные отклонения для обеих величин. Также для обеих величины были проверены простые гипотезы о нормальном распределении с помощью критерия Пирсона хи-квадрат. Для обеих величин при выбранном уровне значимости гипотезы были приняты.

2. КОРРЕЛЯЦИОННЫЙ И РЕГРЕССИОННЫЙ АНАЛИЗ

2.1. Теоретические сведения

Функциональная зависимость (функция) между двумя множествами – соответствие (сопоставление) элементов одного множества элементам другого, такое что каждому элементу первого множества соответствует только один элемент из второго множества.

$$f : X \longrightarrow Y,\ т.ч.\ \ \forall x \in X\ \exists! y = f(x).$$

Статистическая зависимость между двумя (или более) случайными величинами – любое статистическое отношение между этими величинами. Если две случайные величины зависимы, они не отвечают свойству независимости.

Две случайные величины называются независимыми тогда, когда наблюдение за одной из них не влияет и не дает информации о другой величине.

Корреляционная зависимость – вид статистической зависимости между двумя (или более) случайными величинами. Корреляционная зависимость отображает, насколько зависимость между случайными величинами похожа на линейную.

Коэффициент корреляции Пирсона является наиболее часто используемым числовым показателем корреляционной зависимости.

Пусть имеются две случайные величины X и Yс математическими ожиданиями μ_x, μ_y и среднеквадратичными отклонениями σ_x, σ_y соответственно. Тогда линейный коэффициент корреляции Пирсона ρ:

$$\rho = \frac{cov(X,\ Y)}{\sigma_x \sigma_y} = \frac{E\Big((X-\mu_x)\big(Y-\mu_y\big)\Big)}{\sigma_x \sigma_y},\ где\ cov(X,Y) - ковариация;\ E(g) - мат.\ ожидание\ g.$$

Свойства коэффициента корреляции:

- Принимает значения от -1 до 1: $\rho \in [-1;\ 1]$.
- $\rho = 1$ означает, что случайные величины идеально коррелируют, т.е. имеют возрастающую линейную зависимость. При $\rho = -1$ наблюдается обратная идеальная корреляция, убывающая линейная зависимость.

- Если случайные величины независимы, то ρ = 0 (т.е. случайные величины не коррелируют). Обратное в общем случае неверно.

Также есть выборочный аналог для коэффициента корреляции:

$$r_в = \frac{\sum_{i=1}^{K_x}\sum_{j=1}^{K_y} n_{ij}x_i y_j - N\bar{x}_в\bar{y}_в}{NS_xS_y} \qquad (*)$$

и

$$r_в = \frac{\sum_{i=1}^{K_x}\sum_{j=1}^{K_y} n_{ij}u_i v_j - N\bar{u}_в\bar{v}_в}{NS_uS_v}, \quad (**)$$

где: $\boldsymbol{K_x}$, $\boldsymbol{K_y}$ – количество интервалов в интервальных рядах для выборок значений случайных величин $\boldsymbol{X}$ и $\boldsymbol{Y}$ соответственно;

$\boldsymbol{u}$, $\boldsymbol{v}$ – условные варианты, значения которых соответствуют выборочным (интервальным) значениям случайных величин $\boldsymbol{X}$ и $\boldsymbol{Y}$ соответственно;

$\boldsymbol{N}$ – общий объем выборки.

Для *построения доверительного интервала для коэффициента корреляции Пирсона* производится r-z преобразование Фишера:

$$z = \frac{1}{2}\ln\left(\frac{1+r}{1-r}\right) = arctgh(r).$$

Установлено, что если исходная двумерная выборка имеет двумерное нормальное распределение с коэффициентом корреляции ρ, а обе величины, ее составляющие, являются независимыми одинаково распределенными случайными величинами, то статистика z примерно имеет нормальное распределение с мат. ожиданием $z_\rho = \frac{1}{2}\ln\left(\frac{1+\rho}{1-\rho}\right)$ и дисперсией примерно $\frac{1}{N-3}$.

$$z \sim \mathcal{N}\left(\frac{1}{2}\ln\left(\frac{1+\rho}{1-\rho}\right), \frac{1}{N-3}\right).$$

Тогда согласно теореме Фишера:

$$\sqrt{m}\frac{\bar{z}-z_\rho}{\frac{1}{\sqrt{N-3}}} \sim \mathcal{N}(0,1), \text{ где}$$

$$(m=1)-\textit{объем выборки } z,$$

$$\bar{z}=z_r=\frac{1}{2}\ln\left(\frac{1+r_в}{1-r_в}\right)-\textit{среднее выборочное}.$$

После преобразований:

$$\left(z_r-z_\rho\right)\sqrt{N-3}\sim\mathcal{N}(0,1)$$

Чтобы построить оценку для ρ, сначала строится интервальная оценка для z_ρ:

$$\mathbb{P}\left(\left|z_r-z_\rho\right|\sqrt{N-3}\le t_\gamma\right)=\gamma;$$

$$\mathbb{P}\left(-t_{\frac{1+\gamma}{2}}\le\left(z_\rho-z_r\right)\sqrt{N-3}\le t_{\frac{1+\gamma}{2}}\right)=\gamma;$$

$$\mathbb{P}\left(z_r-\frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}}\le z_\rho\le z_r+\frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}}\right)=\gamma.$$

Здесь t_i – *i-квантиль* стандартного нормального распределения.

Таким образом интервальная оценка z_ρ:

$$\left(z_{\rho1},\quad z_{\rho2}\right)=\left(z_r-\frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}},\quad z_r+\frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}}\right).$$

Далее осуществляется обратное преобразование границ интервала для z_ρ.

$$r_1=\frac{\exp\left(2z_{\rho1}\right)-1}{\exp\left(2z_{\rho1}\right)+1};$$

$$r_2=\frac{\exp\left(2z_{\rho2}\right)-1}{\exp\left(2z_{\rho2}\right)+1}.$$

Доверительный интервал для коэффициента корреляции Пирсона:

$$\rho \in [r_1;\ r_2]\ .$$

Для *проверки статистической гипотезы о равенстве коэффициента корреляции нулю* вводится одна из статистик (r – выборочный коэффициент корреляции):

1. Для выборки объема N:

$$T = r\sqrt{\frac{N-2}{1-r^2}}$$

2. Для группированных данных (интервального ряда), n – количество интервалов:

$$T = \frac{r\sqrt{n-2}}{1-r^2}$$

В случае, когда исходная двумерная выборка имеет двумерное нормальное распределение, а ее случайные величины не коррелируют, данные статистики имеют распределение Стьюдента с (N-2) или (n-2) степенями свободы соответственно.

Затем вычисляется значение $T_{набл} = T(r_B)$ для оцененного коэффициента корреляции.

Необходимо выбрать уровень значимости α и вычислить для него критическое значение $t_{1-\frac{\alpha}{2}}$ как $\left(1-\frac{\alpha}{2}\right) - квантиль$ распределения Стьюдента с соответствующими степенями свободы. Критическое значение сравнивается с $T_{набл}$: если наблюдаемое T превосходит критическое значение, нулевая гипотеза о нулевой корреляции отвергается при заданном уровне значимости, иначе – принимается (для заданного уровня значимости).

Метод наименьших квадратов – стандартный способ аппроксимации решения переопределенных систем, т.е. тех, у которых количество уравнений превышает количество неизвестных. Решение, полученное с помощью МНК,

минимизирует сумму квадратов остатков. Остаток – это разница между наблюдаемым значением и моделируемым (аппроксимированным) значением.

Регрессионный анализ – это раздел статистики (или часть статистического мат. аппарата), занимающийся оценкой отношений между переменными. Регрессионный анализ помогает понять, как значение зависимой переменной меняется при варьировании одной из независимых переменных (факторов) при фиксированных остальных.

2.2. Обработка и анализ двумерной выборки

1) Двумерная выборка $N = 107$.

v	501.00	369.00	344.00	473.00	426.00	528.00	497.00	467.00	506.00	431.00	454.00
E	130.40	84.30	86.80	137.90	121.10	163.40	147.30	140.50	158.40	125.00	131.10
v	371.00	482.00	393.00	441.00	463.00	440.00	481.00	340.00	468.00	397.00	496.00
E	89.20	139.90	103.20	122.80	129.10	128.50	135.20	85.10	142.00	108.60	143.10
v	434.00	541.00	352.00	438.00	453.00	423.00	351.00	525.00	409.00	469.00	386.00
E	122.30	146.80	87.70	134.90	119.50	131.10	89.00	165.90	121.00	131.50	95.50
v	505.00	436.00	488.00	449.00	493.00	512.00	472.00	423.00	465.00	351.00	359.00
E	137.50	114.30	134.10	124.50	129.70	169.90	134.20	130.80	140.70	102.90	71.90
v	457.00	467.00	400.00	418.00	492.00	434.00	510.00	392.00	463.00	459.00	397.00
E	126.40	135.10	114.60	118.60	137.50	110.50	140.60	82.70	125.00	145.40	106.80
v	424.00	436.00	429.00	398.00	493.00	522.00	518.00	463.00	437.00	386.00	493.00
E	119.00	116.70	112.90	109.00	154.50	154.50	144.40	121.20	121.80	105.80	151.20
v	414.00	480.00	585.00	562.00	508.00	421.00	463.00	422.00	406.00	544.00	345.00
E	113.50	153.90	177.70	175.90	159.00	117.80	136.70	122.90	110.10	166.70	95.90
v	478.00	393.00	437.00	448.00	458.00	422.00	468.00	430.00	371.00	543.00	471.00
E	126.60	122.80	115.10	121.90	121.70	115.70	144.90	104.30	91.90	155.40	143.90
v	475.00	521.00	353.00	437.00	362.00	490.00	484.00	459.00	480.00	482.00	522.00
E	132.00	139.60	98.00	118.40	111.70	139.90	140.40	136.70	153.30	148.20	143.80
v	576.00	390.00	514.00	442.00	421.00	443.00	438.00	429.00			
E	166.40	91.40	153.60	115.40	107.90	121.90	126.70	120.90			

2) Корреляционная таблица.

$x_i y_i$	[71.9,87)	[87,102)	[102,117)	[117,132)	[132,147)	[147,163)	[163,178]	
[340,375)	4	6	2	0	0	0	0	12
[375,410)	1	2	7	2	0	0	0	12
[410,445)	0	0	10	16	1	0	0	27
[445,480)	0	0	0	12	11	0	0	23

$x_i y_i$	[71.9,87)	[87,102)	[102,117)	[117,132)	[132,147)	[147,163)	[163,178]	
[480,515)	0	0	0	2	10	8	1	21
[515,550)	0	0	0	0	4	2	3	9
[550,585]	0	0	0	0	0	0	3	3
	5	8	19	32	26	10	7	107

3) Выборочный коэффициент корреляции r_B был высчитан с помощью языка R по обеим формулам: (*) – для непосредственных значений, и (**) – для условных вариант.

$$r_в = \frac{\sum_{i=1}^{K_x}\sum_{j=1}^{K_y} n_{ij}x_i y_j - N\bar{x}_в\bar{y}_в}{NS_xS_y}, \qquad (*)$$

$$r_{B1} = 0.8825998.$$

$$r_в = \frac{\sum_{i=1}^{K_x}\sum_{j=1}^{K_y} n_{ij}u_i v_j - N\bar{u}_в\bar{v}_в}{NS_uS_v}, \quad (**)$$

$$r_{B2} = 0.8825998.$$

Значения коэффициентов совпадают.

4) Доверительный интервал для коэффициента корреляции.

Выберем надежность оценки $\gamma = 0.95$.

Затем, имея точечную оценку коэффициента корреляции $r_в$ находим z (преобразование Фишера):

$$z_r = \frac{1}{2}\ln\left(\frac{1+r_в}{1-r_в}\right) = 1.41\,.$$

Интервальная оценка для z_ρ:

$$\left(z_{\rho 1}, \quad z_{\rho 2}\right) = \left(z_r - \frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}}, \quad z_r + \frac{t_{\frac{1+\gamma}{2}}}{\sqrt{N-3}}\right)$$

Здесь $t_{\frac{1+\gamma}{2}} = t_{0.975} = 1.95$ – *0.975-квантиль* стандартного нормального распределения.

$$\left(z_{\rho 1}, \quad z_{\rho 2}\right) = \left(1.19522, \quad 1.57960\right).$$

Обратное преобразование:

$$r_i = \frac{\exp\left(2z_{\rho i}\right) - 1}{\exp\left(2z_{\rho i}\right) + 1}.$$

Доверительный интервал для коэффициента корреляции ρ:

$$\rho \in \left[0.8321907; 0.9185394\right].$$

5) Проверка статистической гипотезы о равенстве коэффициента корреляции нулю проводится для случая группированных данных, количество интервалов $K = 7$.

Выберем уровень значимости $\alpha = 0.05$.

Критическое значение $t_{кр} = 2.57$.

$$T_{набл} = \frac{r_{B1}\sqrt{K-2}}{1 - {r_{B1}}^2} = 8.929391.$$

Поскольку $T_{набл} > t_{кр}$ гипотезу о равенстве коэффициента корреляции нулю необходимо отвергнуть при уровне значимости $\alpha = 0.05$.

6) Построение прямых линейной регрессии.

Имеем модель линейной регрессии:

$$\begin{cases} y \sim kx + b \\ x \sim ty + d \end{cases}$$

Уравнения выборочных прямых среднеквадратической регрессии имеют следующий вид:

$$\begin{cases} y(x) = \bar{y} + r_B \frac{S_y}{S_x}(x - \bar{x}) \\ x(y) = \bar{x} + r_B \frac{S_x}{S_y}(y - \bar{y}) \end{cases}, \quad \text{где } \bar{x},\ \bar{y} - \textit{выборочные средние};$$

$r_B - \textit{выборочный коэффициент корреляции}$;

$S_x,\ S_y - \textit{выборочные СКВО}$.

После подстановки соответствующих значений уравнения прямых имеют вид:

$$\begin{cases} y(x) = 0.3574181x + 127.2013 \\ x(y) = 2.179471y + 449.743 \end{cases}$$

Оценки остаточной дисперсии:

- Остаточная дисперсия Y относительно X:

$$Rv_{\frac{y}{x}} = S_y^2(1 - r_B^2)$$

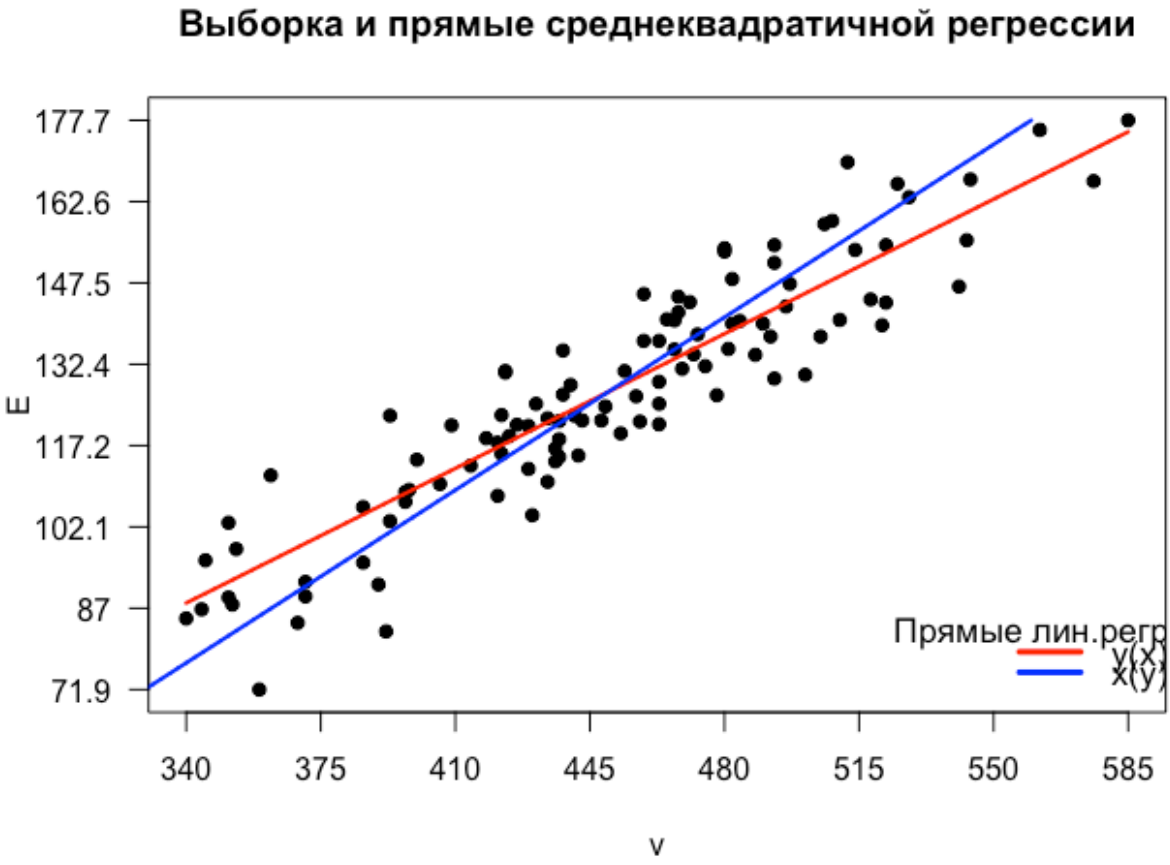
$$Rv_{\frac{y}{x}} = 497.74*(1 - 0.788) = 105.8849$$

- Остаточная дисперсия X относительно Y:

$$Rv_{\frac{x}{y}} = S_x^2(1 - r_B^2)$$

$$Rv_{\frac{x}{y}} = 645.6670$$

График полученных функций и точки, составляющие выборку.



7) Построение уравнений выборочных кривых для параболической среднеквадратической регрессии.

Уравнения имеют вид:

$$\begin{cases} y(x) = a_1 x^2 + b_1 x + c_1 \\ x(y) = a_2 y^2 + b_2 y + c_2 \end{cases}$$

Задача минимизации, N=107 – размер выборки:

$$\sum_{i=1}^{N} \left(y_i - a x_i^2 - b x_i - c \right)^2 \to min$$

Для x(y) задача выглядит аналогично.

$$\sum_{i=1}^{N} \left(x_i - a y_i^2 - b y_i - c \right)^2 \to min$$

Далее находятся выражения для частных производных по a, b и c и получаются следующие системы линейных уравнений:

Для y(x):

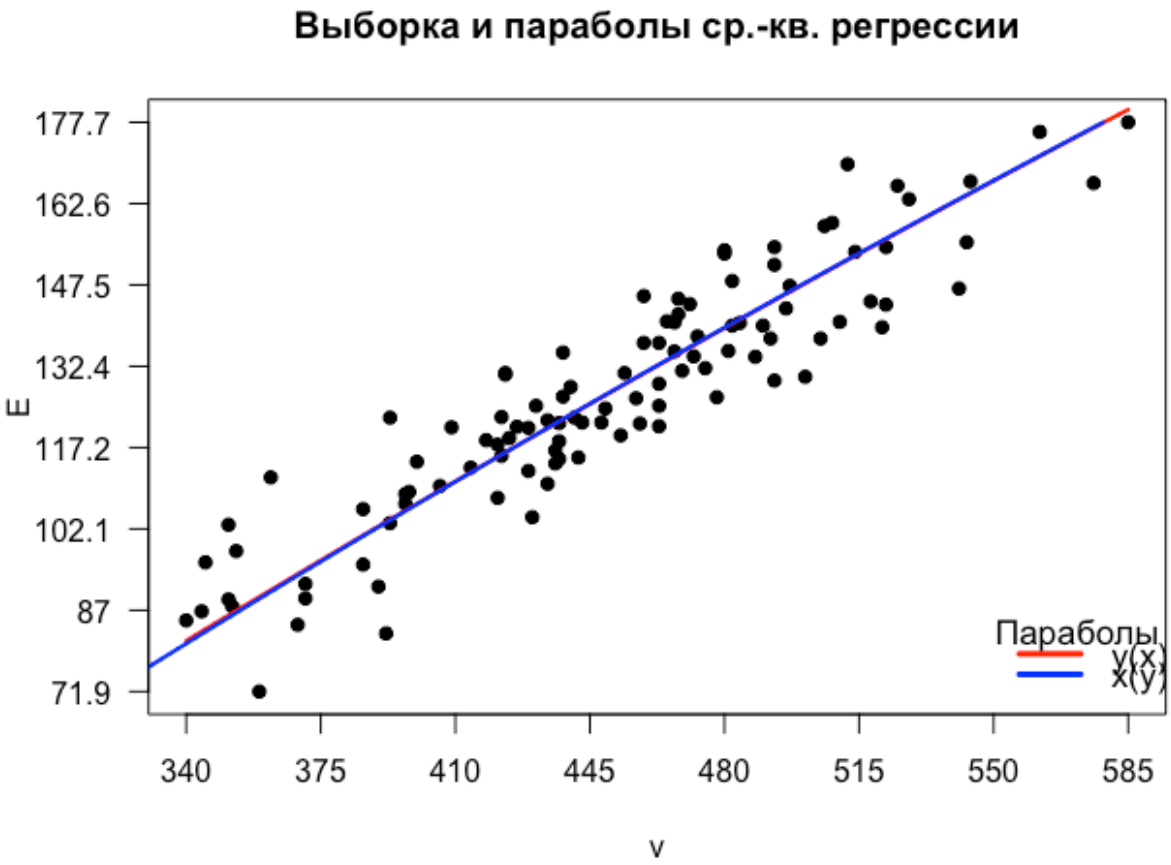
$$\begin{cases} a \sum_{i=1}^{N} x_i^4 + b \sum_{i=1}^{N} x_i^3 + c \sum_{i=1}^{N} x_i^2 = \sum_{i=1}^{N} y_i x_i^2 \\ a \sum_{i=1}^{N} x_i^3 + b \sum_{i=1}^{N} x_i^2 + c \sum_{i=1}^{N} x_i = \sum_{i=1}^{N} y_i x_i \\ a \sum_{i=1}^{N} x_i^2 + b \sum_{i=1}^{N} x_i + cN = \sum_{i=1}^{N} y_i \end{cases}$$

Для x(y):

$$\begin{cases} a \sum_{i=1}^{N} y_i^4 + b \sum_{i=1}^{N} y_i^3 + c \sum_{i=1}^{N} y_i^2 = \sum_{i=1}^{N} x_i y_i^2 \\ a \sum_{i=1}^{N} y_i^3 + b \sum_{i=1}^{N} y_i^2 + c \sum_{i=1}^{N} y_i = \sum_{i=1}^{N} x_i y_i \\ a \sum_{i=1}^{N} y_i^2 + b \sum_{i=1}^{N} y_i + cN = \sum_{i=1}^{N} x_i \end{cases}$$

После вычислений с помощью R:

$$\begin{cases} y(x) = -0.0001149475 \bullet x^2 + 0.5092418x - 78.51749 \\ x(y) = 0.002040411 \bullet y^2 + 1.934552y + 170.3467 \end{cases}$$



8) Оценка корреляционного отношения.

Преобразуем корреляционную таблицу (пункт 2) следующим образом: с каждым интервалом ассоциирована его середина – отобразим это в таблице, также запишем оценки условного математического ожидания ($\bar{y}_{x_i}$, $\bar{x}_{y_i}$) и дисперсии ($\bar{\delta}^2_{y/x_i}$,) для каждой группы.

$$\bar{y}_{x_i} = \frac{\sum_{j=1}^{K} y_j n_{ij}}{n_{x_i}};$$

$$\bar{x}_{y_i} = \frac{\sum_{j=1}^{K} x_j n_{ji}}{n_{y_i}};$$

$$\bar{\delta}^2_{x/y_i} = \frac{1}{n_{y_i}} \sum_{j=1}^{K} n_{ji}(x_j - \bar{x}_{y_i})^2;$$

$$\bar{D}^2_{y/x_i} = \frac{1}{n_{x_i}} \sum_{j=1}^{K} n_{ij}(y_j - \bar{y}_{x_i})^2 .$$

Здесь x_j, y_j – соответствующие середины j-го интервала; n_{ij} – значение в ячейке корреляционной матрицы на пересечении i-ой строки и j-го столбца; количество интервалов K = 7.

$x_i y_i$	[71.9,87)	[87,102)	[102,117)	[117,132)	[132,147)	[147,163)	[163,178]	N	y_{gr}	$D_{y\ gr\ i}$
[340,375)	4	6	2	0	0	0	0	12	92.05238	107.87522
[375,410)	1	2	7	2	0	0	0	12	107.16667	145.94882
[410,445)	0	0	10	16	1	0	0	27	119.76190	67.68641
[445,480)	0	0	0	12	11	0	0	23	132.02857	57.00245
[480,515)	0	0	0	2	10	8	1	21	145.67211	119.14189
[515,550)	0	0	0	0	4	2	3	9	153.34921	174.85656
[550,585]	0	0	0	0	0	0	3	3	170.14286	0.00000
N	5	8	19	32	26	10	7	107		
x_{gr}	364.5000	366.2500	407.2368	442.8125	485.3846	504.5000	542.5000			
$D_{x\ gr\ i}$	196.0000	229.6875	556.5097	607.7148	748.4098	196.0000	600.0000			

Рассчитаем внутригрупповые дисперсии:

$$\bar{Д}_{x_{внгр}} = \frac{1}{N} \sum_{i=1}^{K} (n_{y_i} * \bar{\delta}^2_{\frac{x}{y_i}});$$

$$\bar{Д}_{y_{внгр}} = \frac{1}{N} \sum_{i=1}^{K} (n_{x_i} * \bar{\delta}^2_{\frac{y}{x_i}}),$$

где $K = 7$ *– количество интервалов,* $N = 99$ *– объем выборки .*

$$\bar{Д}_{x_{внгр}} = 546.3244; \quad \bar{Д}_{y_{внгр}} = 95.88941.$$

Рассчитаем межгрупповые дисперсии:

$$\bar{Д}_{x_{межгр}} = \frac{1}{N} \sum_{i=1}^{K} (\bar{x}_{y_i} - \bar{x})^2 n_{y_i};$$

$$\bar{Д}_{y_{межгр}} = \frac{1}{N}\sum_{i=1}^{K}\left(\bar{y}_{x_i} - \bar{y}\right)^2 n_{x_i},$$

$$где\ \bar{x} = 458.1,\ \ \bar{y} = 130.7 - соответств.\ выб.\ средние$$

$$\bar{Д}_{x_{межгр}} = 2347.71;\ \ \bar{Д}_{y_{межгр}} = 378.7121.$$

Итого общая дисперсия:

$$\bar{Д}_{общ} = \bar{Д}_{внгр} + \bar{Д}_{межгр}$$

$$\bar{Д}_{x_{общ}} = 2894.034;\ \bar{Д}_{y_{общ}} = 474.6015.$$

Найдем корреляционное отношение:

$$\bar{\eta}^2 = \frac{\bar{Д}_{межгр}}{\bar{Д}_{общ}}$$

$$\bar{\eta^2}_{x/y} = 0.8112239; \quad \bar{\eta^2}_{y/x} = 0.7979581.$$

3. Выводы

Была сформирована двумерная выборка, была составлена корреляционная таблица для интервальных рядов, была получена точечная оценка коэффициента корреляции Пирсона по двум формулам, в обоих случая результаты совпали. Также был построен доверительный интервал для коэффициента корреляции и проверена и отвергнута гипотеза о равенстве коэффициента Пирсона нулю.

По полученному значению выборочного коэффициента корреляции (r_B = 0.8825998), близкому к единице, можно сделать вывод, что рассматриваемые случайные величины сильно коррелируют между собой. Именно этим объясняется тот факт, что нулевая гипотеза о равенстве коэффициента корреляции нулю будет отвергнута, что и было сделано по итогу проверки.

Для сформированной двумерной выборки были получены уравнения для прямых и парабол среднеквадратической регрессии, были построены их

графики, найдены оценки остаточных и общих дисперсий и корреляционного отношения.

3. КЛАСТЕРНЫЙ АНАЛИЗ

1. Теоретические сведения

Кластерный анализ (или кластеризация) – задача распределения однородных объектов из данного набора по группам (кластерам) таким образом, что объекты, принадлежащие одной группе, больше похожи друг на друга, чем на объекты из других групп. Процесс кластеризации подразумевает формирование этих групп. Это общая нечеткая формулировка задачи, на практике существует множество критериев «похожести» объектов – требований, предъявляемых кластерам, поэтому существует большое количество различных алгоритмов кластеризации, а также их модификаций.

Существующие методы можно разделить следующим образом:

I. По степени принадлежности объектов кластерам:
   - Строгая кластеризация – каждый объект может принадлежать только одному кластеру.
   - Нестрогая кластеризация – каждый объект может принадлежать нескольким кластерам в разной степени.

II. По способу формирования кластеров (по критериям кластеров):
   - Иерархические алгоритмы (по связанности объектов)
     (1) Аггломеративные (восходящие, объекты объединяются в кластеры)
     (2) Дивизивные (нисходящие, кластеры дробятся на более мелкие)
   - Алгоритмы, основанные на поиске/использовании центров кластеров (k-means)
   - Кластеризация по распределениям
   - Кластеризация по плотностям

В данной работе использовалось два *критерия качества* кластерных разбиений:

(3) Средний кластерный радиус $r_{кл}$:

$$r_{кл} = \frac{1}{K}\sum_{i=1}^{K}\left(\frac{1}{m_i}\sum_{j=1}^{m_i} dist\left(c_i,\ \ p_j^i\right)\right),$$

$$где\ K - количество\ кластеров,\ \ m_i - количество\ объектов\ в\ кластере\ i$$

$$c_i - центр\ кластера\ i,\ \ p_j^i - j ый\ объект,\ принадлежащий\ кластеру\ i$$

$$dist(x,\ y) - евклидово\ расстояние\ между\ x\ и\ y$$

Для данного показателя вычисляются для каждого кластера расстояния от центров до точек, им принадлежащих, находятся средние радиусы для каждого кластера, а затем вычисляется среднее значение из средних.

(4) Среднее внутрикластерное расстояние $d_{кл}$:

$$d_{кл} = \frac{1}{K}\sum_{i=1}^{K}\frac{2}{(m_i-1)m_i}\sum_{p,q\in\mathcal{K}_i} dist(p,\ q),$$

$$где\ K - количество\ кластеров,\ \ m_i - количество\ объектов\ в\ кластере\ i,$$

$$\mathcal{K}_i - кластер\ i,\ \ dist(p,\ q) - евклидово\ расстояние\ между\ p\ и\ q.$$

Положим, в кластере $\mathcal{K}_i$ содержится m_i элементов, тогда для каждого кластера вычисляются расстояния между его элементами и высчитывается среднее, а затем для всех кластеров находится среднее из средних.

Для измерения расстояния используется евклидово расстояние между точками в двумерном пространстве:

$$\mathrm{dist}(\mathrm{p},\ \mathrm{q}) = \sqrt[2]{\left(p_x - q_x\right)^2 + \left(p_y - q_y\right)^2}$$

За центр кластера принимается центр масс (барицентр, центроид) точек, ему принадлежащих:

$$\left(c_x,\ c_y\right)_i = \left(\frac{1}{m_i}\sum_{j=1}^{m_i} x_j^i,\quad \frac{1}{m_i}\sum_{j=1}^{m_i} y_j^i\right),$$

$$где\ m_i - количество\ элементов\ в\ кластере\ i,$$

$$\left(x_j^i,\ y_j^i\right) - координаты\ j^{ой}\ точки\ кластера\ i$$

Метод к-средних.

Смысл данного метода состоит в том, что каждый кластер определяется своим центром (необязательно элементом исходного множества), а каждый объект принадлежит к кластеру с ближайшим к этому элементу центром. В ходе работы алгоритма центры кластеров пересчитываются и точки перераспределяются между кластерами.

Входные данные:

- исходное множество, подлежащее кластеризации;
- К – количество кластеров;
- $\{p_1, p_2, \ldots, p_K\}$ – набор объектов исходного множества, которые принимаются за центры, начальное приближение.

В данной работе было реализовано два варианта алгоритма к-средних.

I. «Ленивый» (lazy) вариант:

1. Для начального приближения все точки исходного множества разбиваются на кластеры: каждая точка относится к кластеру с ближайшим к ней центром.
2. Центры кластеров пересчитываются.
3. Для пересчитанных центров снова все точки перераспределяются по кластерам.
4. Шаги 2 и 3 повторяются, пока кластеры не стабилизируются, т.е. пока разбиения не перестанут меняться.

II. «Долгий» (long) вариант:

1. Формирование начальных кластеров. Просматривается каждая точка исходного множества, кроме начальных приближений:
    i. добавляется в ближайший кластер,
    ii. производится перерасчет центров кластеров.
2. После формирования кластеров просматриваются уже все объекты множества по одному и перераспределяются в ближайший кластер, после перераспределения каждого элемента производится перерасчет центров.

Поиск продолжается, пока кластеры не стабилизируются. В качестве критерия стабилизации кластеров было принято неизменение среднего кластерного радиуса в течение пяти итераций.

Метод поиска сгущений (ForEl).

В данном методе каждый кластер определяется своим центром (как и в к-средних) и радиусом (одинаковым для всех кластеров), которые вместе образуют круг (шар). Эти области покрывают все множество исходных точек. Данному кластеру принадлежат точки, лежащие внутри его круга.

Таким образом, ситуация, когда области нескольких кластеров пересекаются, может возникнуть довольно часто. Необходимо обуславливаться, каким образом решать подобные конфликты. В данной работе исследовалась строгая кластеризация и предпочтение отдавалось кластеру с наибольшим количеством элементов, при равенстве элементов выбирался первый встретившийся кластер из соперничающих.

Входные данные:

- исходное множество, подлежащее кластеризации;
- R – радиус.

Выбор кластерного радиуса является нетривиальной задачей, поскольку зависит от вида кластеризуемого множества.

Также алгоритму входе поиска кластеров (итеративно) необходимо выбирать стартовые точки для инициализации центров. Конечный результат

может сильно варьироваться в зависимости от «удачности» начальных приближений.

В данной работе было реализовано две модификации метода поиска сгущений: «стандартный» и «с полным просмотром».

I. «Стандартный» вариант:

1. Из множества рассматриваемых точек по какому-то принципу выбирается первое начальное приближение – центр кластера.
2. Все рассматриваемые точки, лежащие «внутри» кластерного круга (т.е. расстояние до которых от центра кластера меньше радиуса), включаются в текущий кластер, все точки, лежащие вне круга, – кластеру не принадлежат.
3. Пересчитывается центр кластера.
4. Пункты 2 и 3 повторяются до тех пор, пока кластер не стабилизируется, т.е. нельзя ни добавить, ни исключить точки.
5. Все точки, добавленные в кластер, исключаются из рассмотрения. Начинается новая итерация поиска кластера с пункта 1.
6. Поиск продолжается, пока рассматриваемое множество не станет пусто.

Принципы выбора начальных приближений:

- Random – случайным образом.
- Minmedian – необходимо посчитать расстояния между всеми рассматриваемыми точками, затем для всех точки найти медианные расстояния и выбрать минимальное из них, точку с минимальным медианным расстоянием принять за начальное приближение.
- Maxmedian – то же самое, что minmedian, только выбирать максимальную медиану.

II. «С полным просмотром»:

1. Просматриваются все точки множества: каждая точка по очереди принимается за начальное приближение и находятся N (объем выборки) кандидатов в кластеры (уникальных кластеров может оказаться меньше), как описано выше.
2. В качестве найденного кластера выбирается содержащий наибольшее количество точек. Данные точки исключаются из дальнейшего поиска.
3. Повторяются первый и второй пункты для оставшихся точек, пока все точки исходного множества не будут разделены на кластеры.

2. Кластеризация методом k-средних

(1) Масштабированная двумерная выборка.

Для имеющейся двумерной выборки было произведено масштабирование значений (min-max normalization) так, чтобы они все попадали в интервал [0; 1].

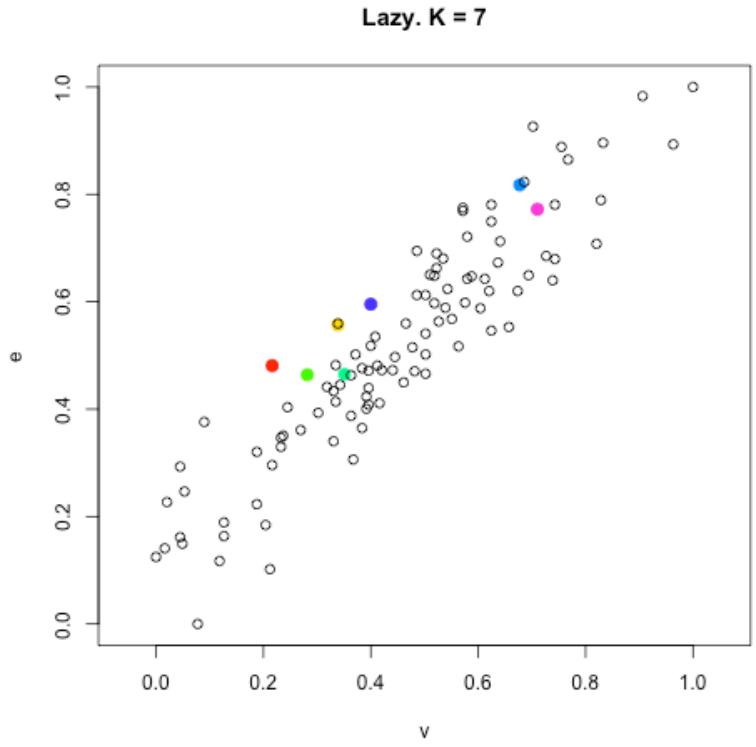
v	0.6571 4286	0.1183 6735	0.0163 2653	0.5428 5714	0.3510 2041	0.7673 4694	0.6408 1633	0.5183 6735	0.6775 5102	0.3714 2857	0.4653 0612
E	0.5529 301	0.1172 023	0.1408 318	0.6238 185	0.4650 284	0.8648 393	0.7126 654	0.6483 932	0.8175 803	0.5018 904	0.5595 463
v	0.1265 3061	0.5795 9184	0.2163 2653	0.4122 4490	0.5020 4082	0.4081 6327	0.5755 1020	0.0000 0000	0.5224 4898	0.2326 5306	0.6367 3469
E	0.1635 161	0.6427 221	0.2958 412	0.4810 964	0.5406 427	0.5349 716	0.5982 987	0.1247 637	0.6625 709	0.3468 809	0.6729 679
v	0.3836 7347	0.8204 0816	0.0489 7959	0.4000 0000	0.4612 2449	0.3387 7551	0.0448 9796	0.7551 0204	0.2816 3265	0.5265 3061	0.1877 5510
E	0.4763 705	0.7079 395	0.1493 384	0.5954 631	0.4499 055	0.5595 463	0.1616 257	0.8884 688	0.4640 832	0.5633 270	0.2230 624
v	0.6734 6939	0.3918 3673	0.6040 8163	0.4448 9796	0.6244 8980	0.7020 4082	0.5387 7551	0.3387 7551	0.5102 0408	0.0448 9796	0.0775 5102
E	0.6200 378	0.4007 561	0.5879 017	0.4971 645	0.5463 138	0.9262 760	0.5888 469	0.5567 108	0.6502 836	0.2930 057	0.0000 000
v	0.4775 5102	0.5183 6735	0.2448 9796	0.3183 6735	0.6204 0816	0.3836 7347	0.6938 7755	0.2122 4490	0.5020 4082	0.4857 1429	0.2326 5306
E	0.5151 229	0.5973 535	0.4035 917	0.4413 989	0.6200 378	0.3648 393	0.6493 384	0.1020 794	0.5018 904	0.6947 070	0.3298 677
v	0.3428 5714	0.3918 3673	0.3632 6531	0.2367 3469	0.6244 8980	0.7428 5714	0.7265 3061	0.5020 4082	0.3959 1837	0.1877 5510	0.6244 8980
E	0.4451 796	0.4234 405	0.3875 236	0.3506 616	0.7807 183	0.7807 183	0.6852 552	0.4659 735	0.4716 446	0.3204 159	0.7495 274
v	0.3020 4082	0.5714 2857	1.0000 0000	0.9061 2245	0.6857 1429	0.3306 1224	0.5020 4082	0.3346 9388	0.2693 8776	0.8326 5306	0.0204 0816
E	0.3931 947	0.7750 473	1.0000 000	0.9829 868	0.8232 514	0.4338 374	0.6124 764	0.4820 416	0.3610 586	0.8960 302	0.2268 431
v	0.5632 6531	0.2163 2653	0.3959 1837	0.4408 1633	0.4816 3265	0.3346 9388	0.5224 4898	0.3673 4694	0.1265 3061	0.8285 7143	0.5346 9388
E	0.5170 132	0.4810 964	0.4083 176	0.4725 898	0.4706 994	0.4139 887	0.6899 811	0.3062 382	0.1890 359	0.7892 250	0.6805 293
v	0.5510 2041	0.7387 7551	0.0530 6122	0.3959 1837	0.0897 9592	0.6122 4490	0.5877 5510	0.4857 1429	0.5714 2857	0.5795 9184	0.7428 5714
E	0.5680 529	0.6398 866	0.2466 919	0.4395 085	0.3761 815	0.6427 221	0.6474 480	0.6124 764	0.7693 762	0.7211 720	0.6795 841

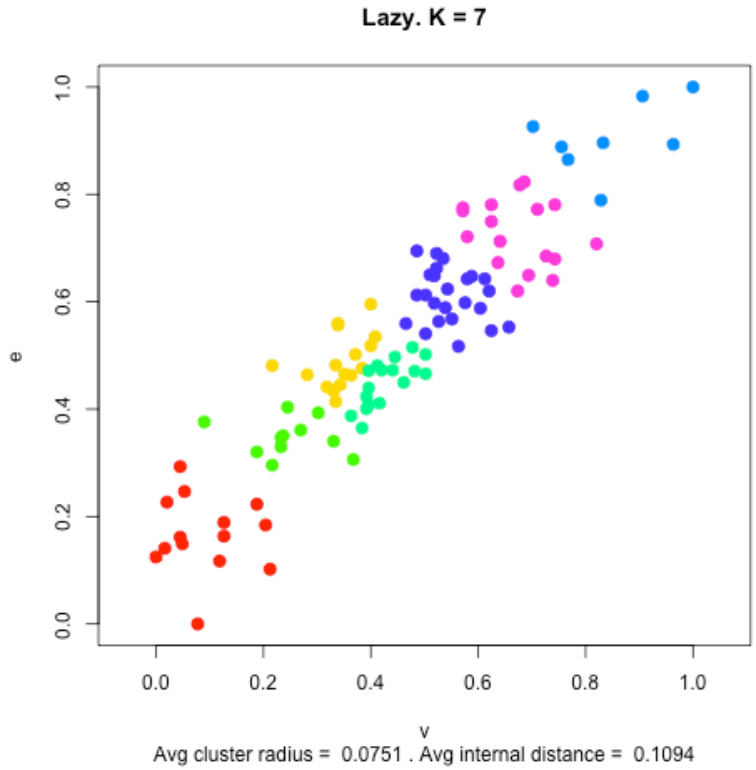
v	0.9632 6531	0.2040 8163	0.7102 0408	0.4163 2653	0.3306 1224	0.4204 0816	0.4000 0000	0.3632 6531			
E	0.8931 947	0.1843 100	0.7722 117	0.4111 531	0.3402 647	0.4725 898	0.5179 584	0.4631 380			

Теперь обе величины, составляющие двумерную выборку, вносят одинаковый вклад при расчете расстояния.

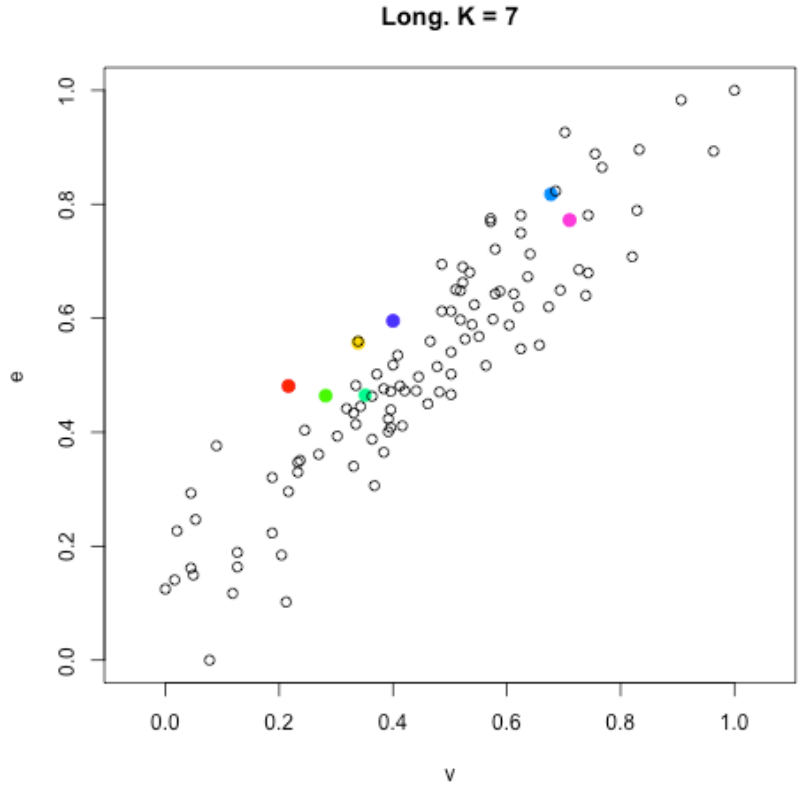
(2) Демонстрация результатов работы «ленивого» и «долгого» вариантов алгоритма K-means при случайном выборе начальных точек для фиксированного количества кластеров K = 7.

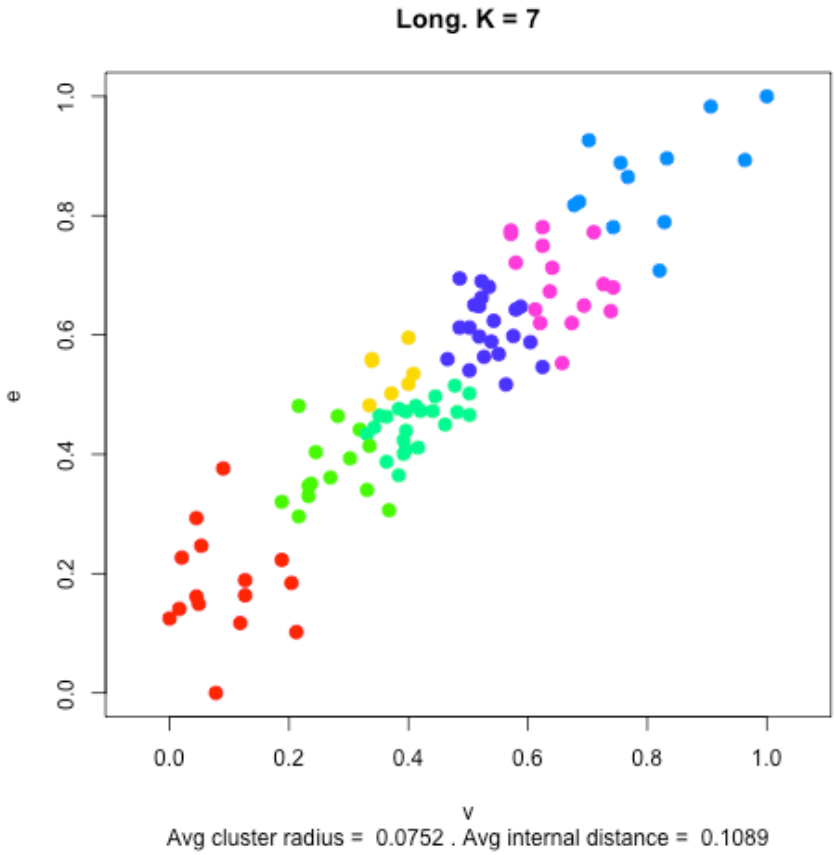
«Ленивый» k-means – пересчет центов кластеров после их формирования (распределения всех точек).





«Долгий» k-means – пересчет центров кластеров после добавления каждой точки.





Сравнительная таблица

№	Средний кластерный радиус		Среднее внутрикластерное расстояние	
	Lazy	Long	Lazy	Long
1	0.0751	0.0752	0.1094	0.1089

В отчете приведен один из серии экспериментов, подтверждающий, что обе реализации алгоритма дают приблизительно одинаковые по своим характеристикам кластеры.

3. Кластеризация методом поиска сгущений

(1) Кластеризация проводилась по масштабированным данным *(см. раздел 3.2 п. 1)*.

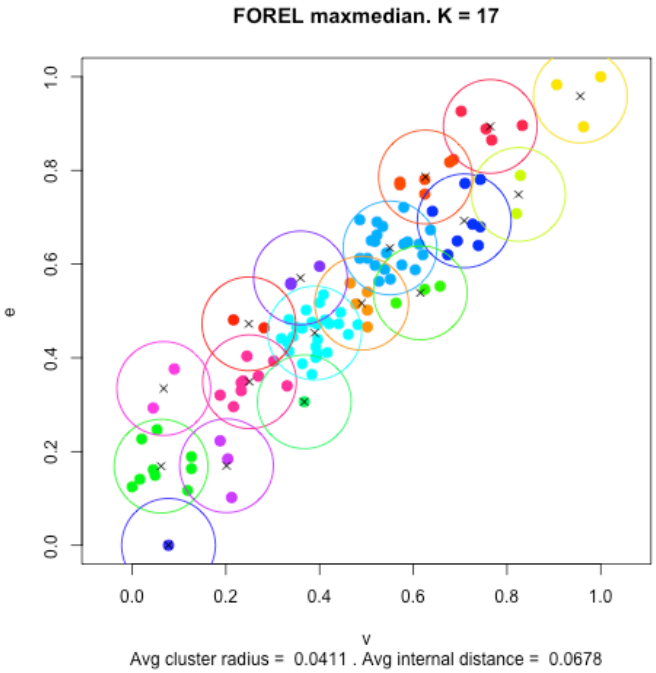
Было проведено два эксперимента: в первом варьировался радиус при поиске кластеров различными модификациями алгоритма, во втором радиус фиксировался, а начальные приближения выбирались случайно.

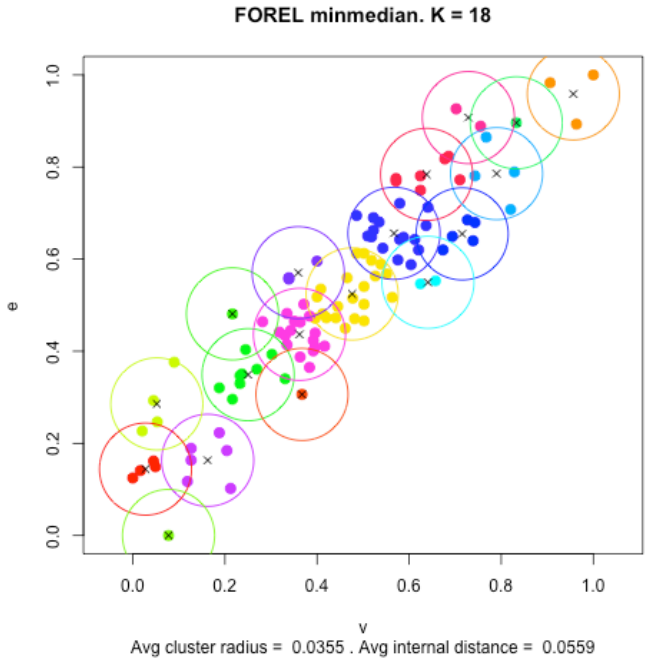
(2) Результаты первого эксперимента.

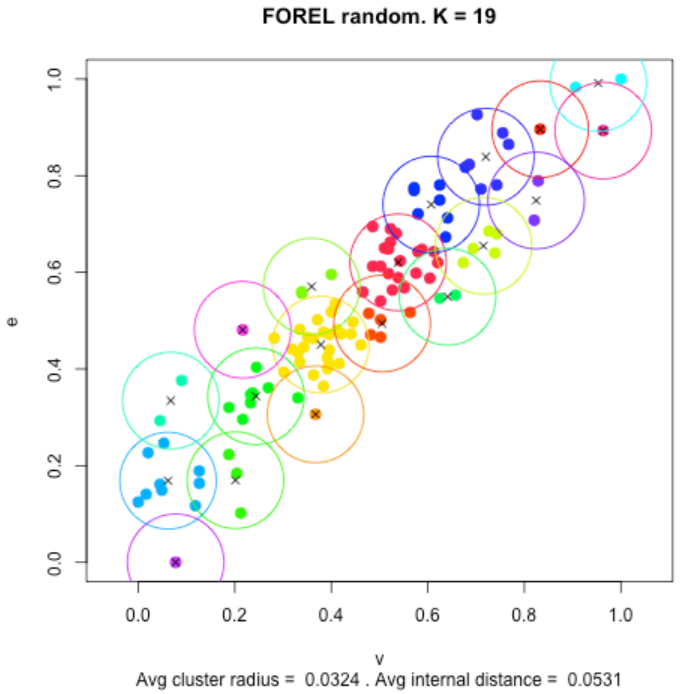
Радиус варьировался в промежутке [0.1; 0.325] с шагом в $\delta = 0.025$.

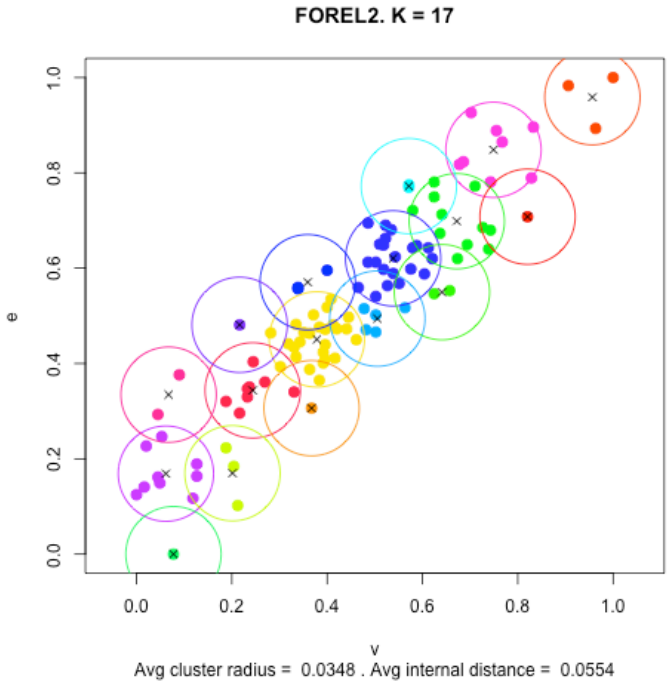
K – количество найденных кластеров. FOREL maxmedian, FOREL minmedian, FOREL random – «стандартный» вариант алгоритма с соответствующим способом выбора начальных приближений. FOREL2 – модификация «с полным просмотром».

R = 0.1

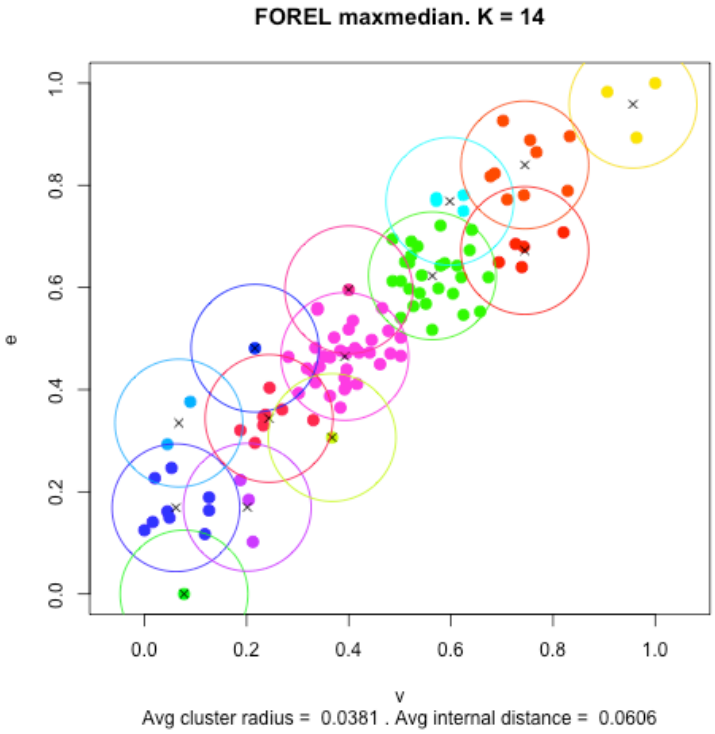

FOREL maxmedian. K = 17
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0411 . Avg internal distance = 0.0678

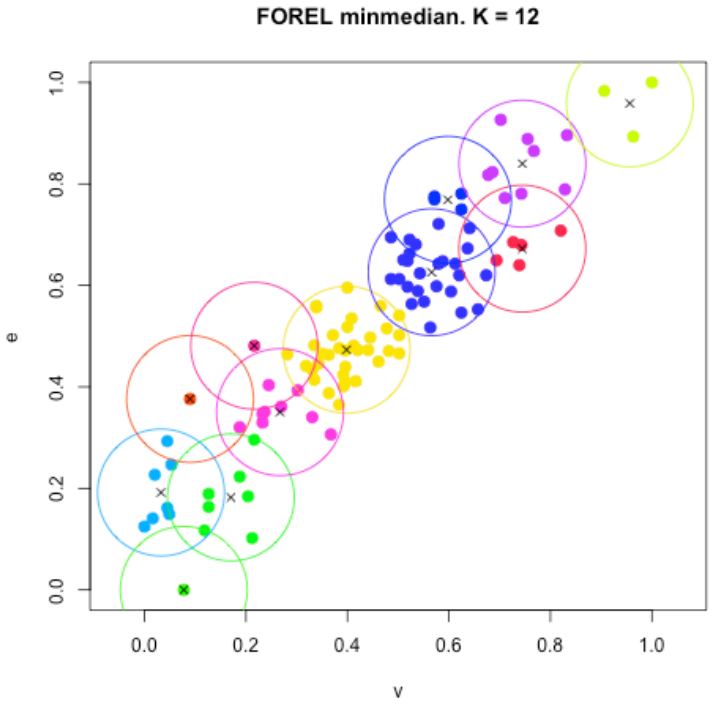

FOREL minmedian. K = 18
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0355 . Avg internal distance = 0.0559

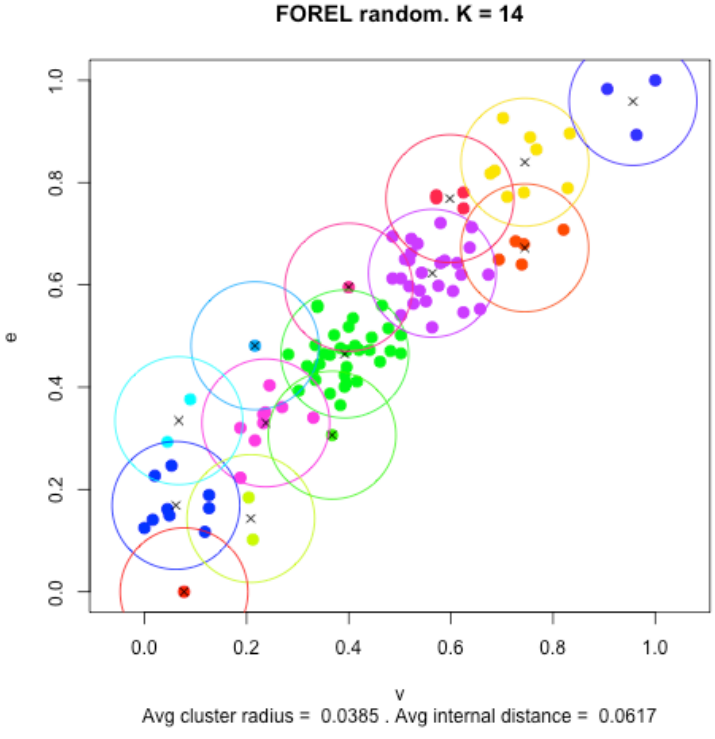

FOREL random. K = 19
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0324 . Avg internal distance = 0.0531

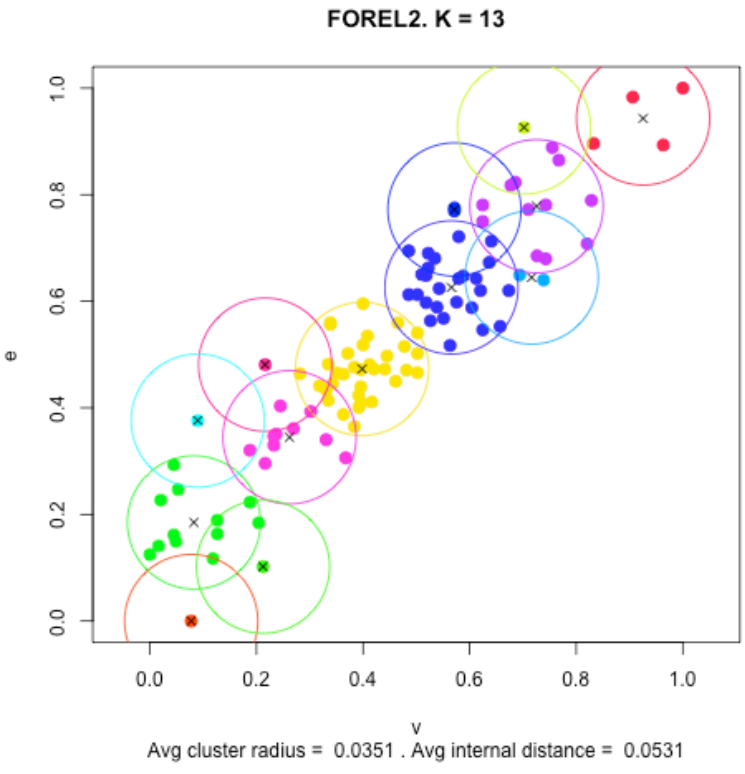

FOREL2. K = 17
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0348 . Avg internal distance = 0.0554

R = 0.125

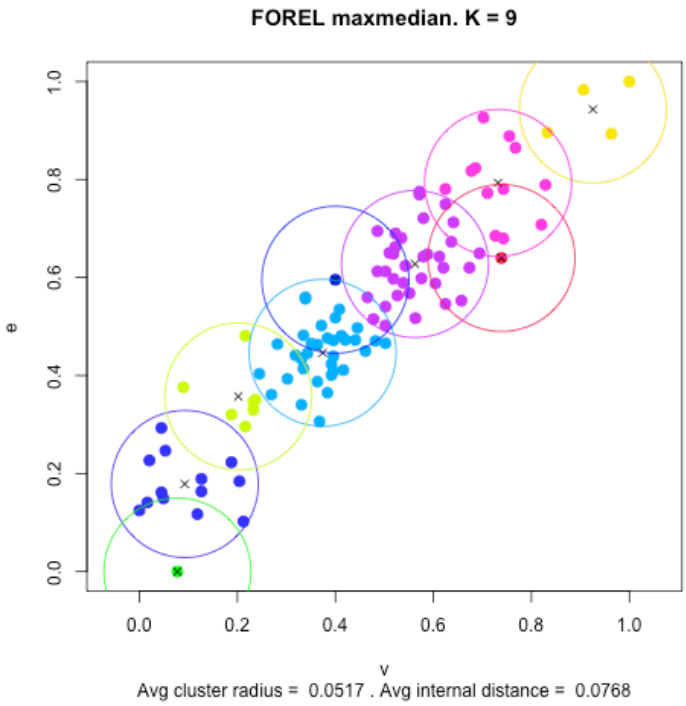

FOREL maxmedian. K = 14
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0381 . Avg internal distance = 0.0606

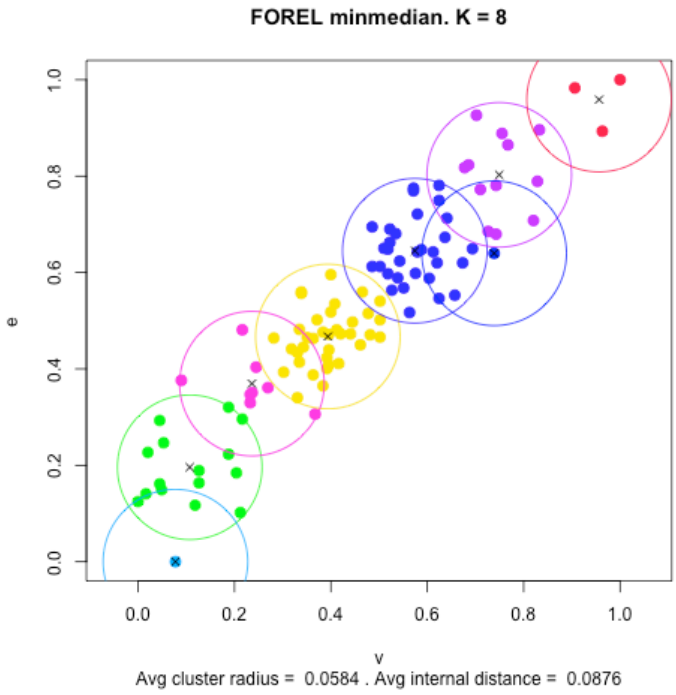

FOREL minmedian. K = 12
e
v
0.0
0.2
0.4
0.6
0.8
1.0

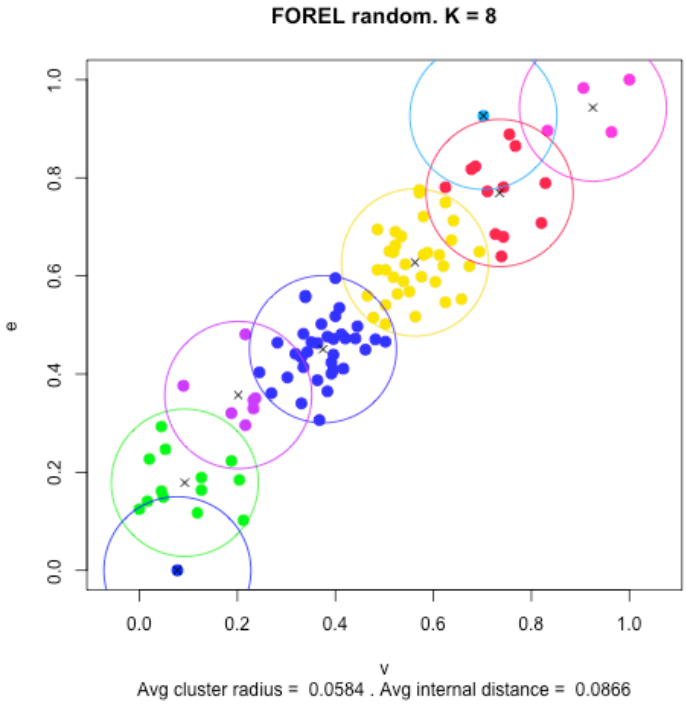

FOREL random. K = 14
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0385 . Avg internal distance = 0.0617

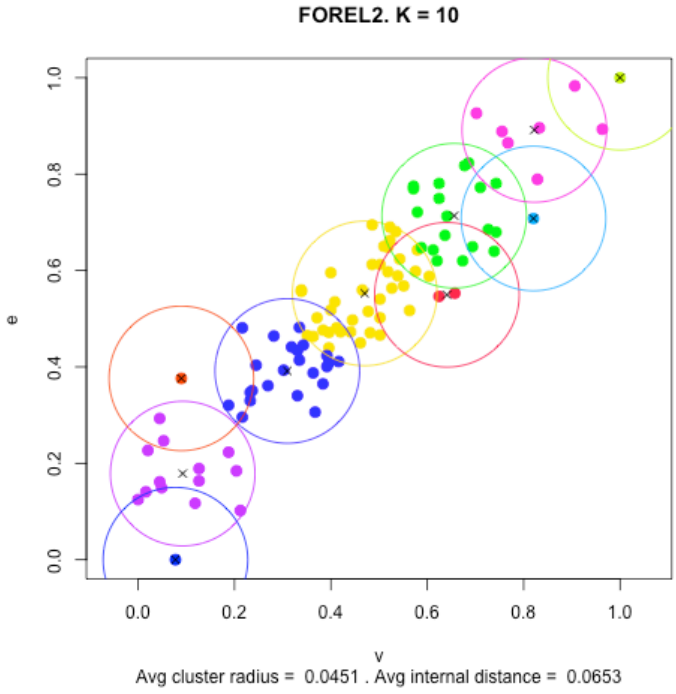

FOREL2. K = 13
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0351 . Avg internal distance = 0.0531

R = 0.15

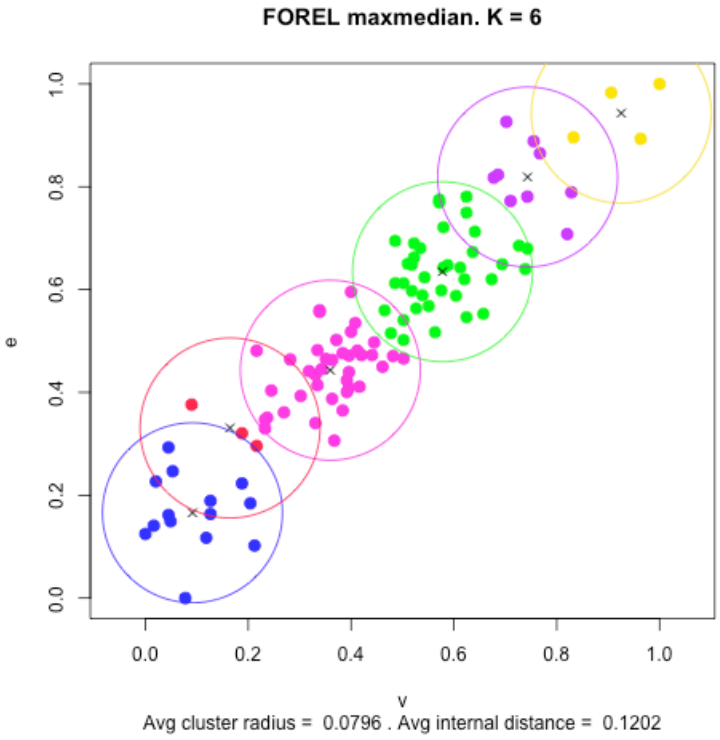

FOREL maxmedian. K = 9
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0517 . Avg internal distance = 0.0768


FOREL minmedian. K = 8
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0584 . Avg internal distance = 0.0876

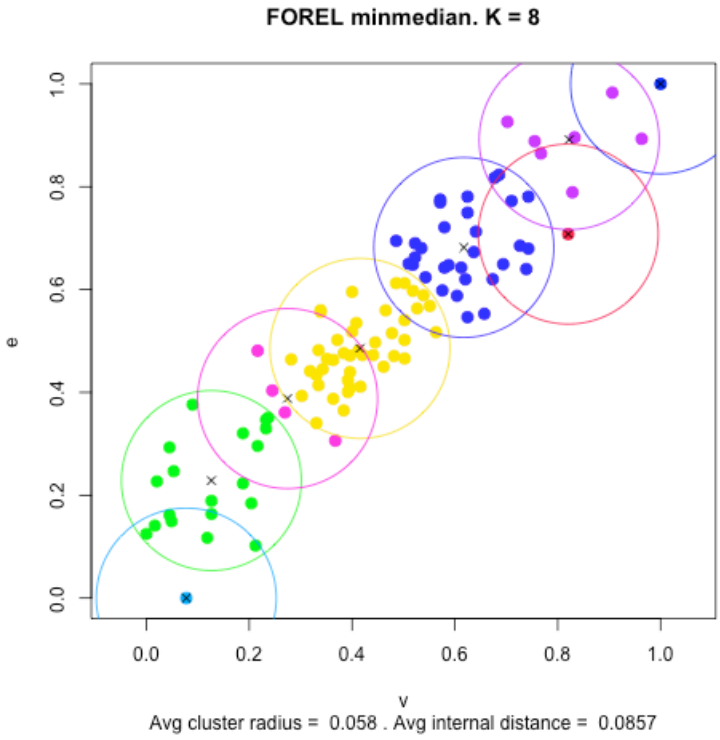

FOREL random. K = 8
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0584 . Avg internal distance = 0.0866


FOREL2. K = 10
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0451 . Avg internal distance = 0.0653

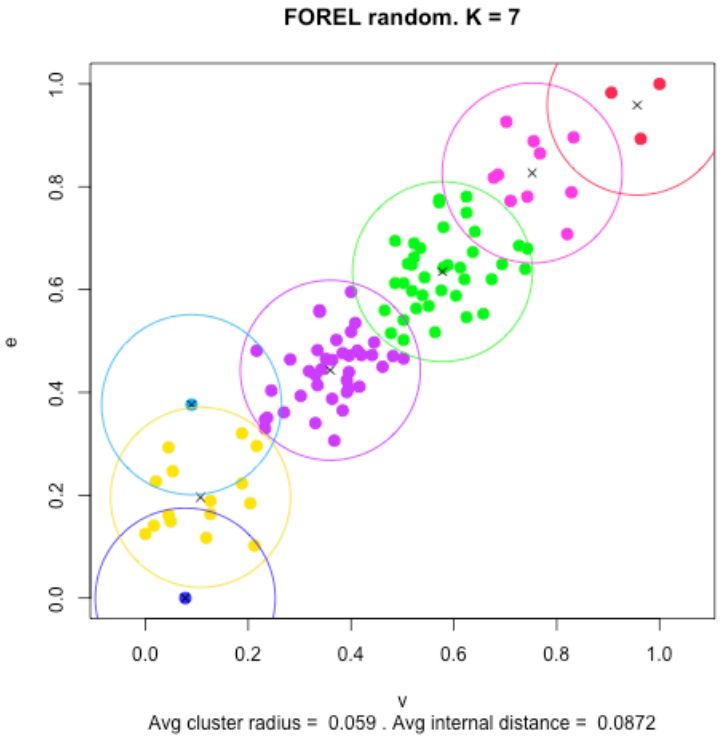
R = 0.175



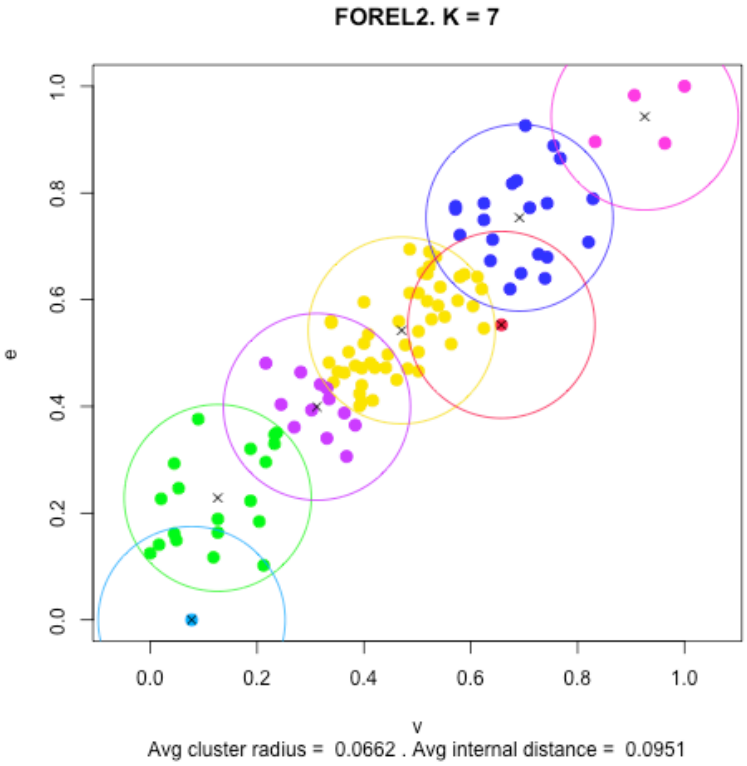
Avg cluster radius = 0.0796 . Avg internal distance = 0.1202



Avg cluster radius = 0.058 . Avg internal distance = 0.0857

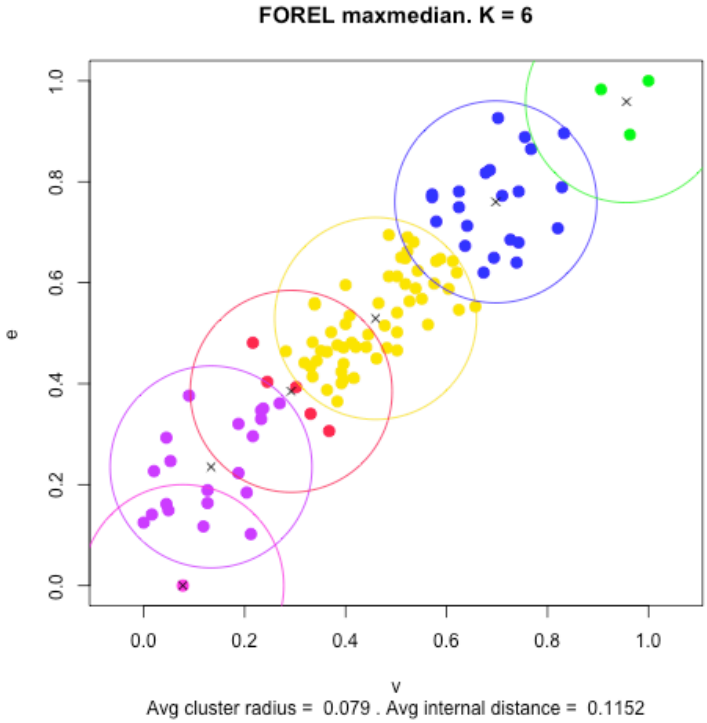


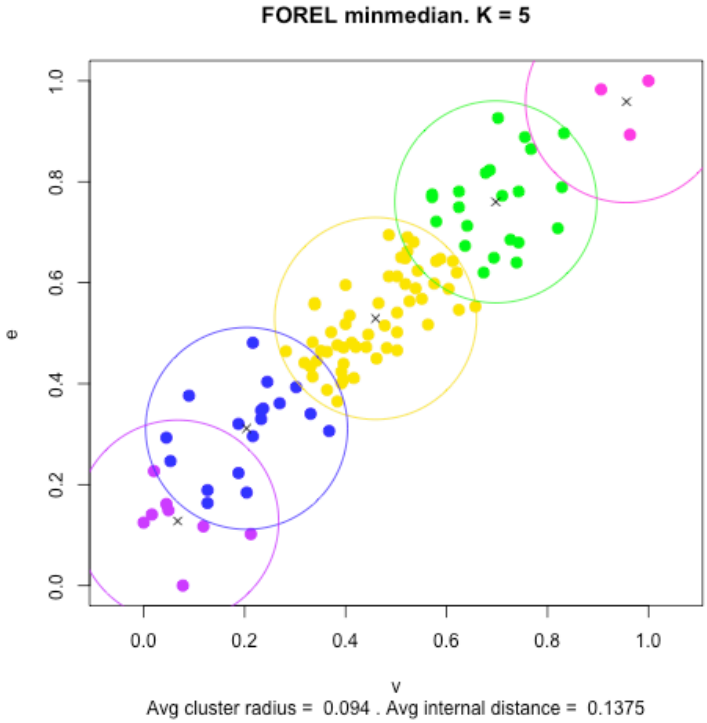
Avg cluster radius = 0.059 . Avg internal distance = 0.0872

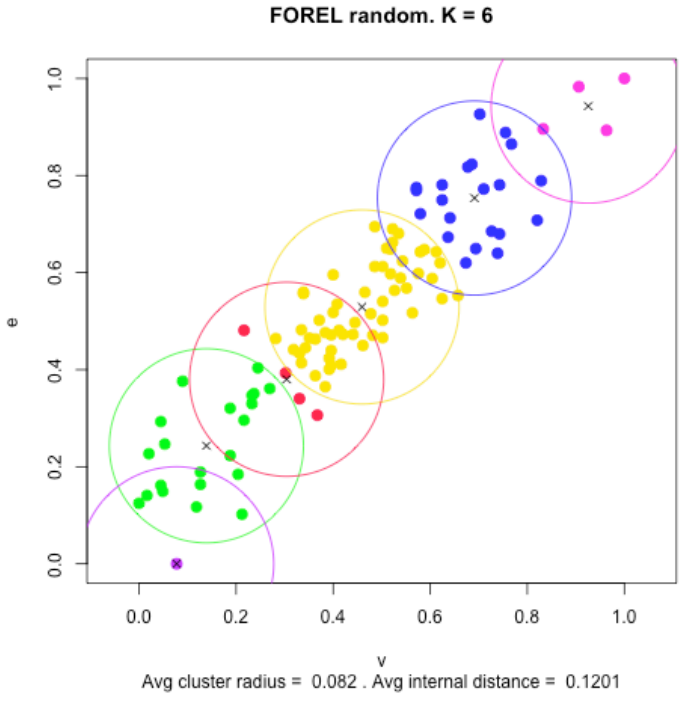


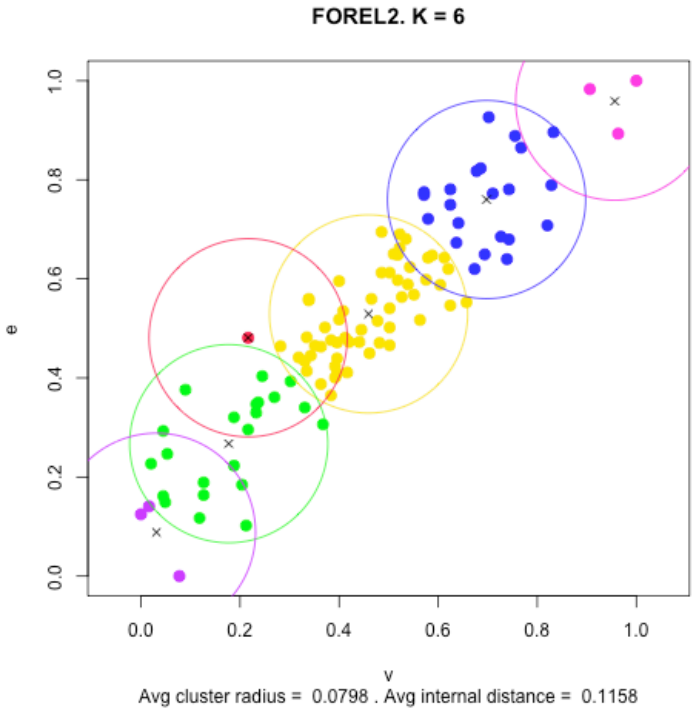
Avg cluster radius = 0.0662 . Avg internal distance = 0.0951

R = 0.2

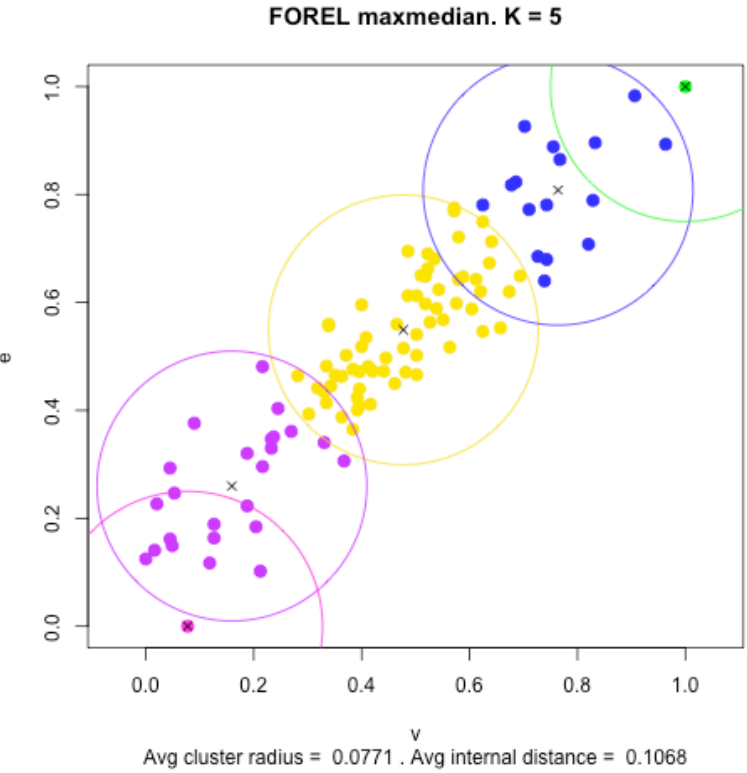

FOREL maxmedian. K = 6
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.079 . Avg internal distance = 0.1152

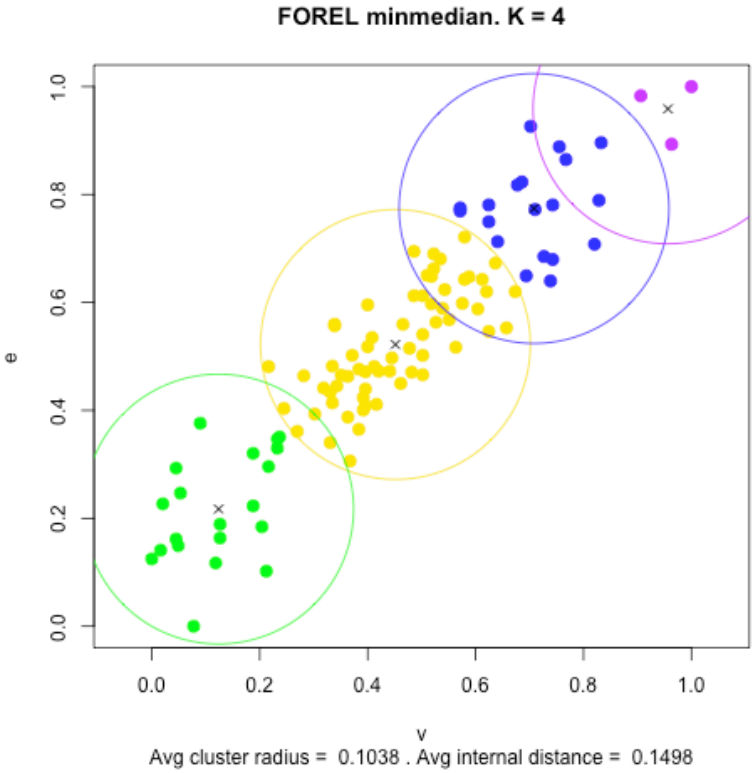

FOREL minmedian. K = 5
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.094 . Avg internal distance = 0.1375

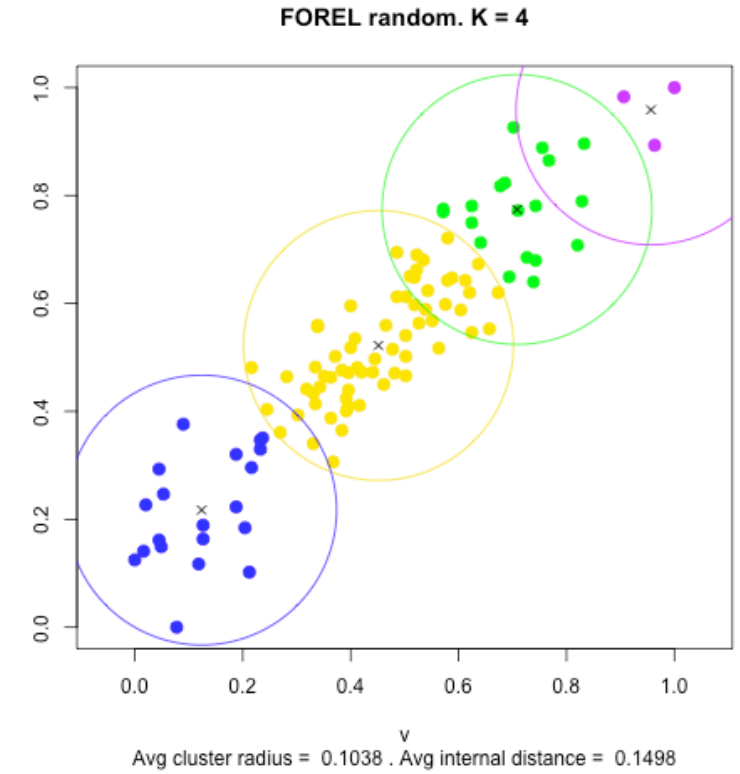

FOREL random. K = 6
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.082 . Avg internal distance = 0.1201

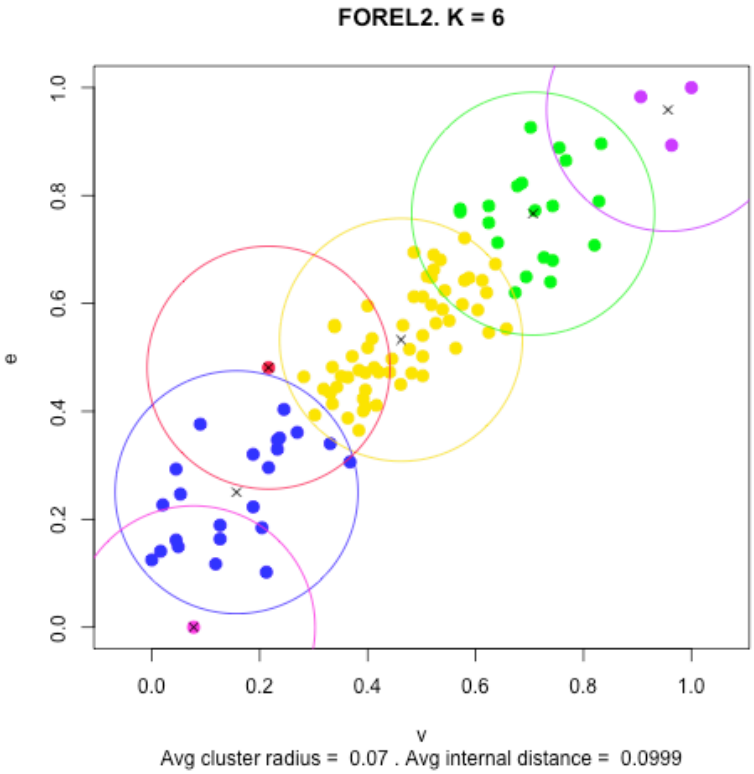

FOREL2. K = 6
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0798 . Avg internal distance = 0.1158

R = 0.225

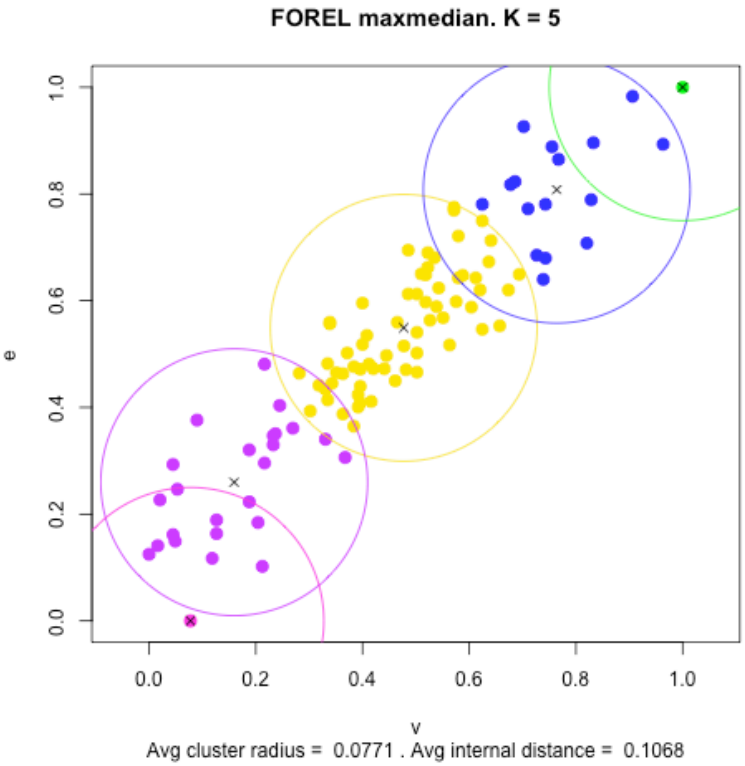

FOREL maxmedian. K = 5
e
v
Avg cluster radius = 0.0771 . Avg internal distance = 0.1068

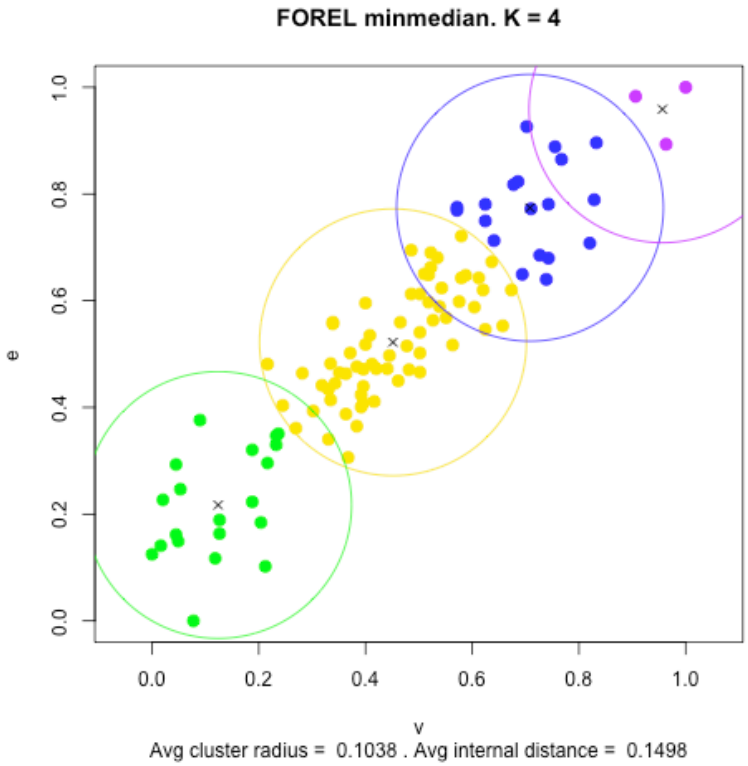

FOREL minmedian. K = 4
e
v
Avg cluster radius = 0.1038 . Avg internal distance = 0.1498

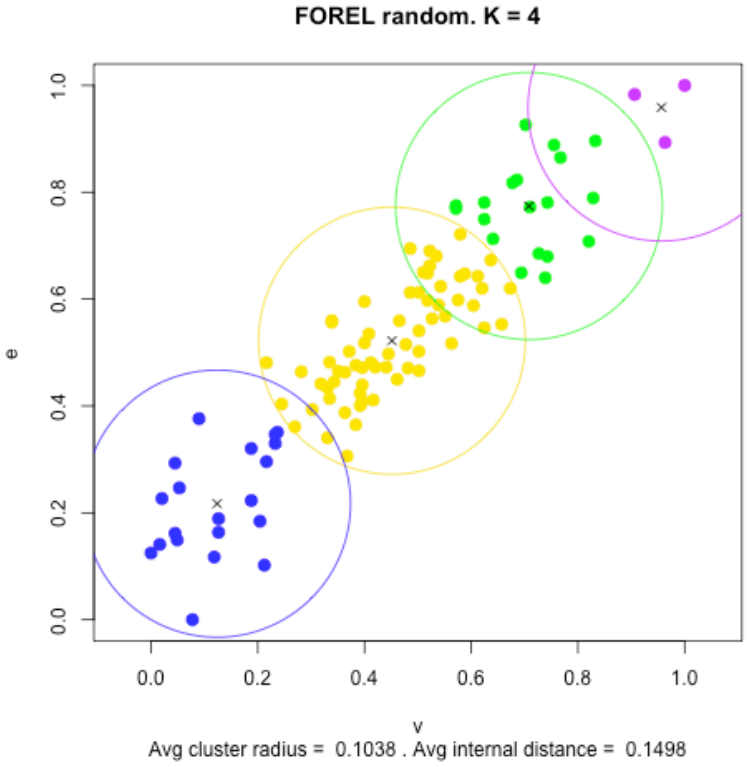

FOREL random. K = 4
v
Avg cluster radius = 0.1038 . Avg internal distance = 0.1498

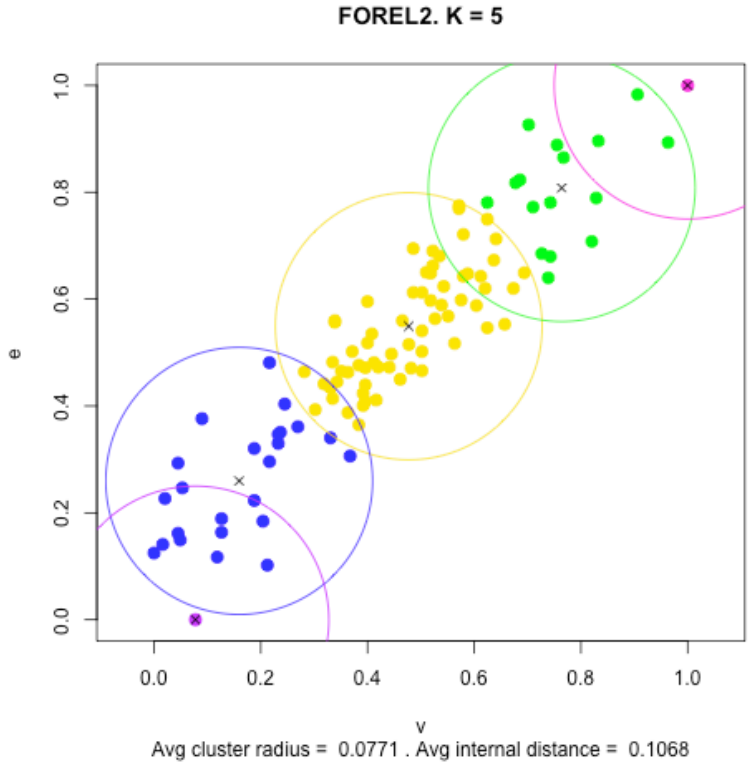

FOREL2. K = 6
e
v
Avg cluster radius = 0.07 . Avg internal distance = 0.0999

R = 0.25

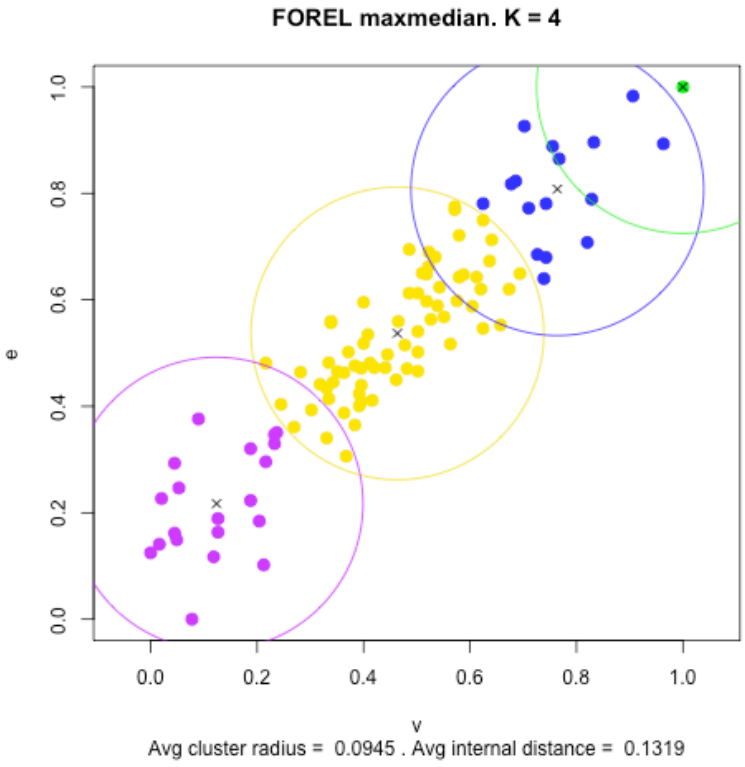

FOREL maxmedian. K = 5
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0771 . Avg internal distance = 0.1068

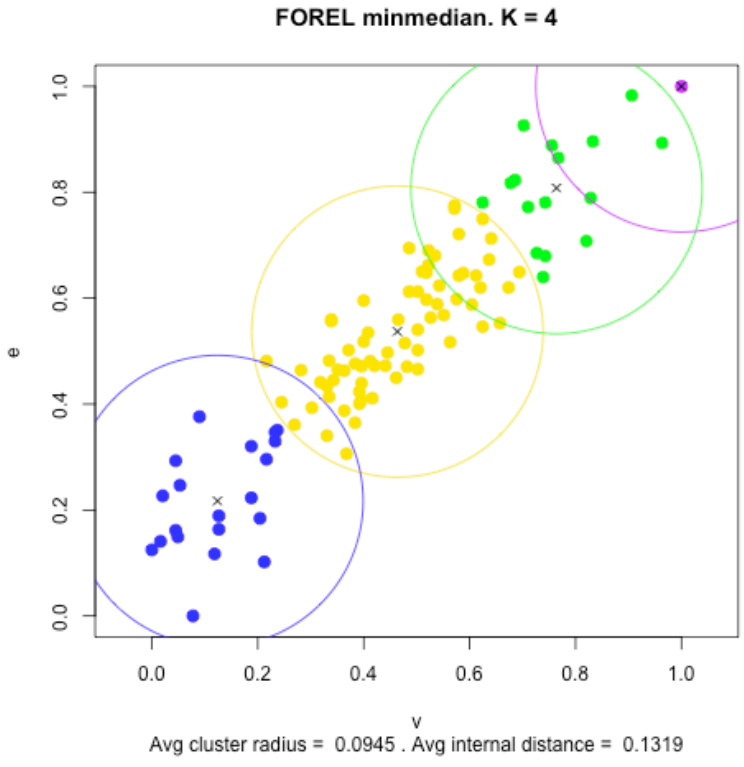

FOREL minmedian. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.1038 . Avg internal distance = 0.1498

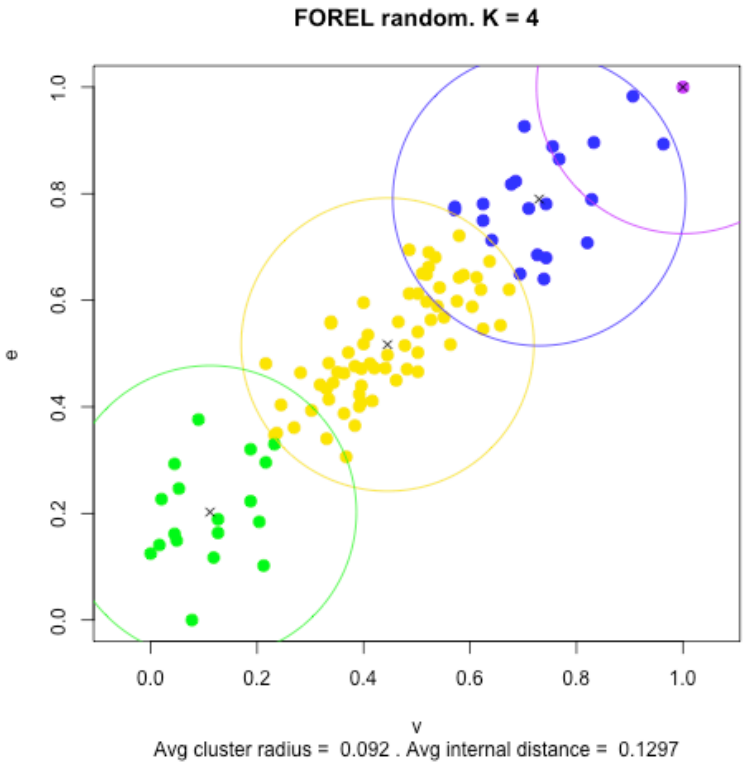

FOREL random. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.1038 . Avg internal distance = 0.1498

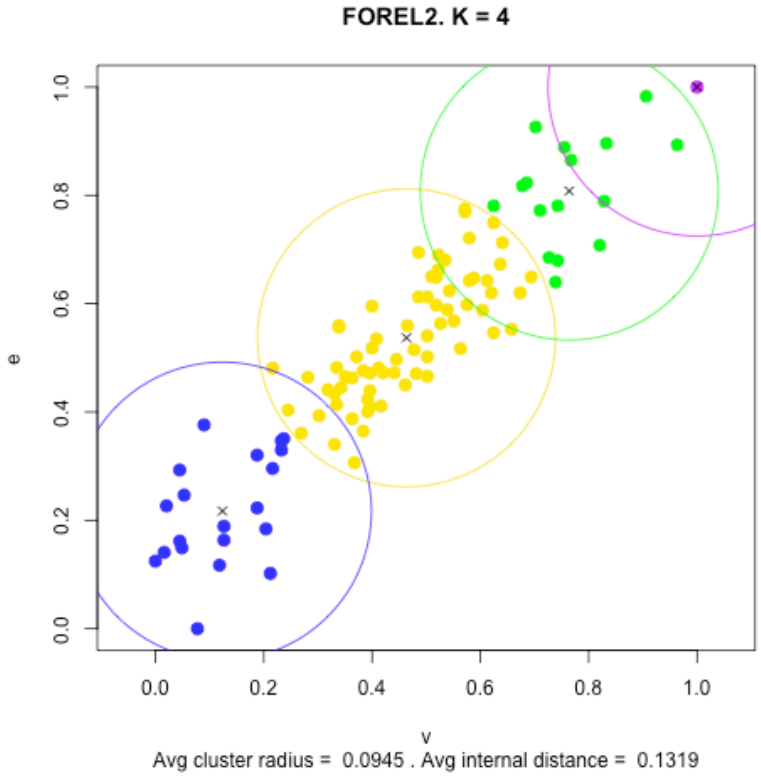

FOREL2. K = 5
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0771 . Avg internal distance = 0.1068

R = 0.275

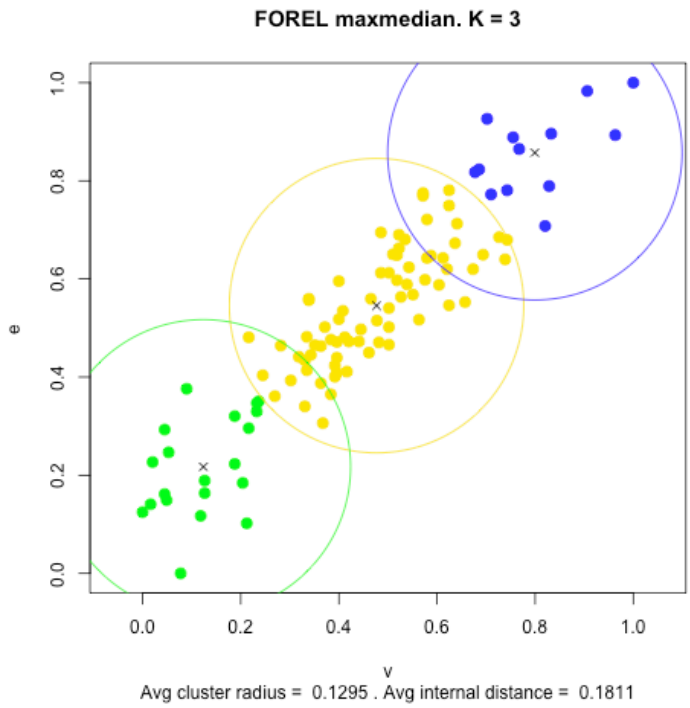

FOREL maxmedian. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0945 . Avg internal distance = 0.1319

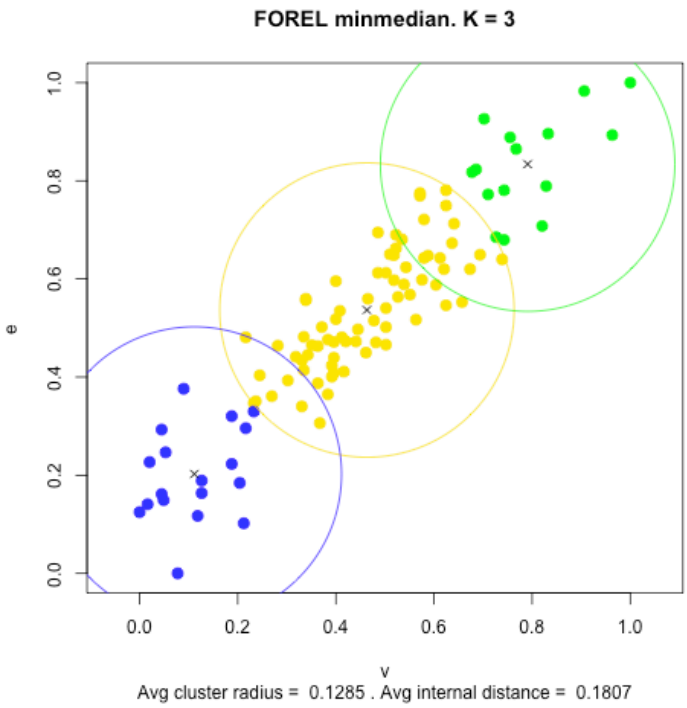

FOREL minmedian. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0945 . Avg internal distance = 0.1319

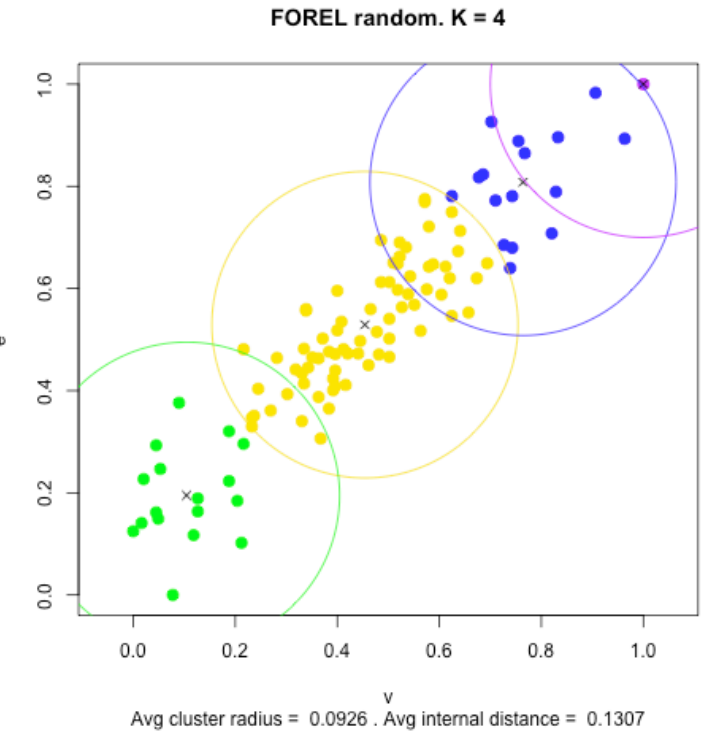

FOREL random. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.092 . Avg internal distance = 0.1297

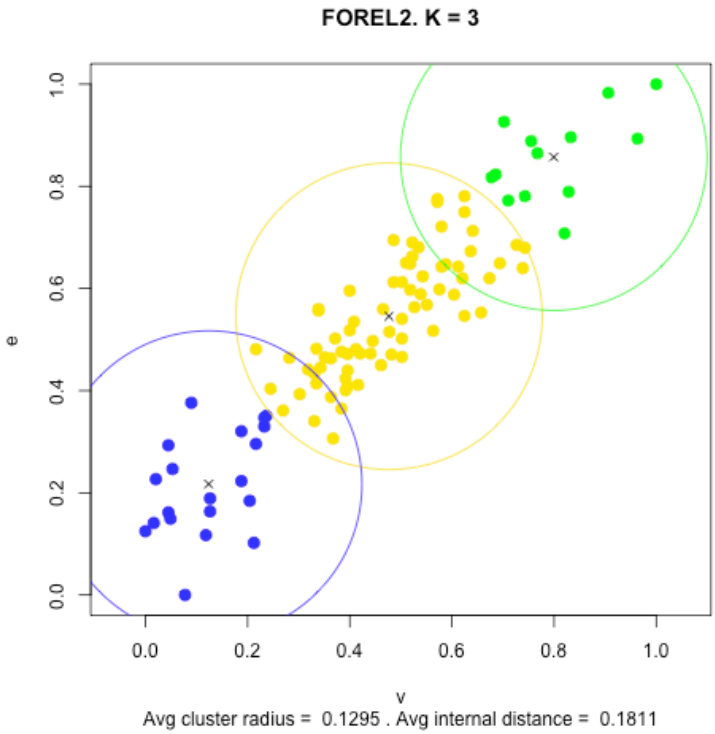

FOREL2. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0945 . Avg internal distance = 0.1319

R = 0.3

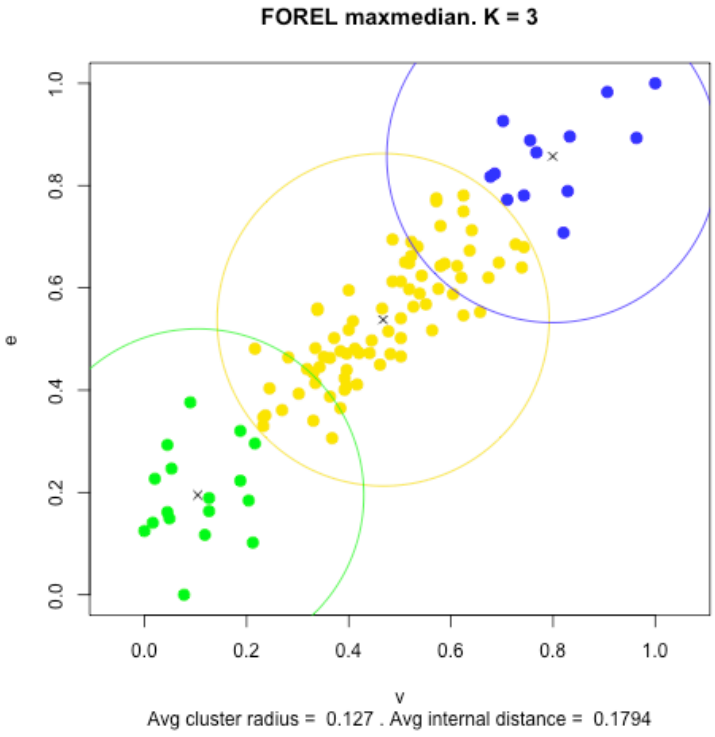

FOREL maxmedian. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.1295 . Avg internal distance = 0.1811

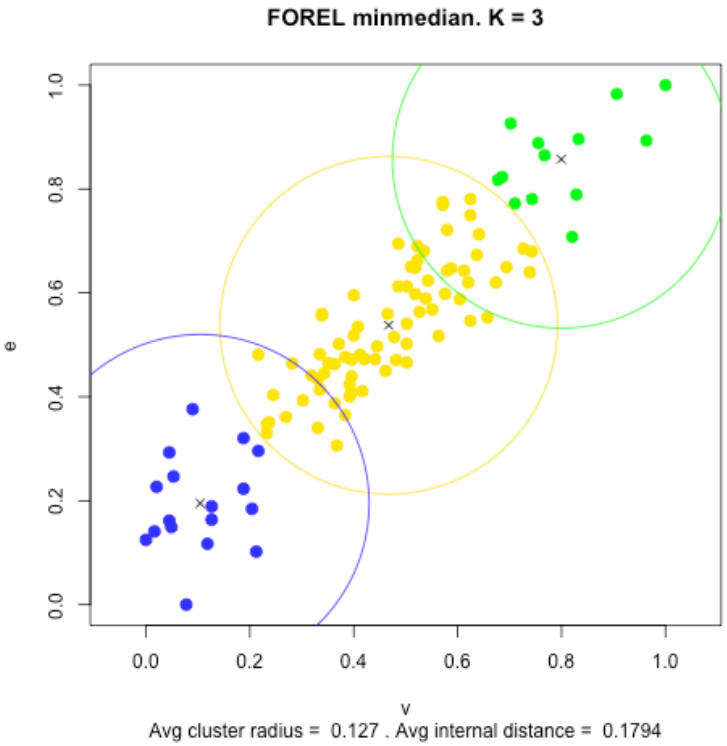

FOREL minmedian. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.1285 . Avg internal distance = 0.1807

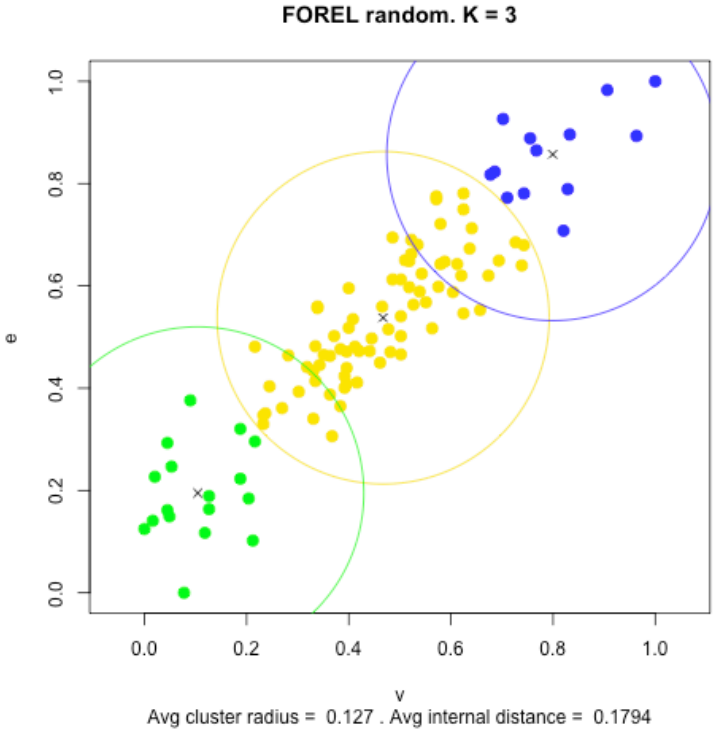

FOREL random. K = 4
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0926 . Avg internal distance = 0.1307

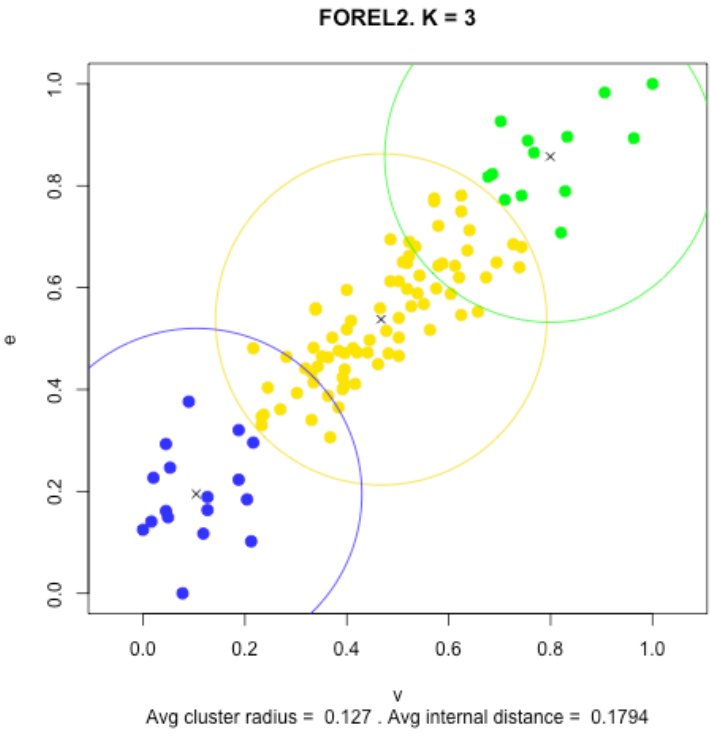

FOREL2. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.1295 . Avg internal distance = 0.1811

R = 0.325


FOREL maxmedian. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.127 . Avg internal distance = 0.1794


FOREL minmedian. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.127 . Avg internal distance = 0.1794


FOREL random. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.127 . Avg internal distance = 0.1794


FOREL2. K = 3
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.127 . Avg internal distance = 0.1794

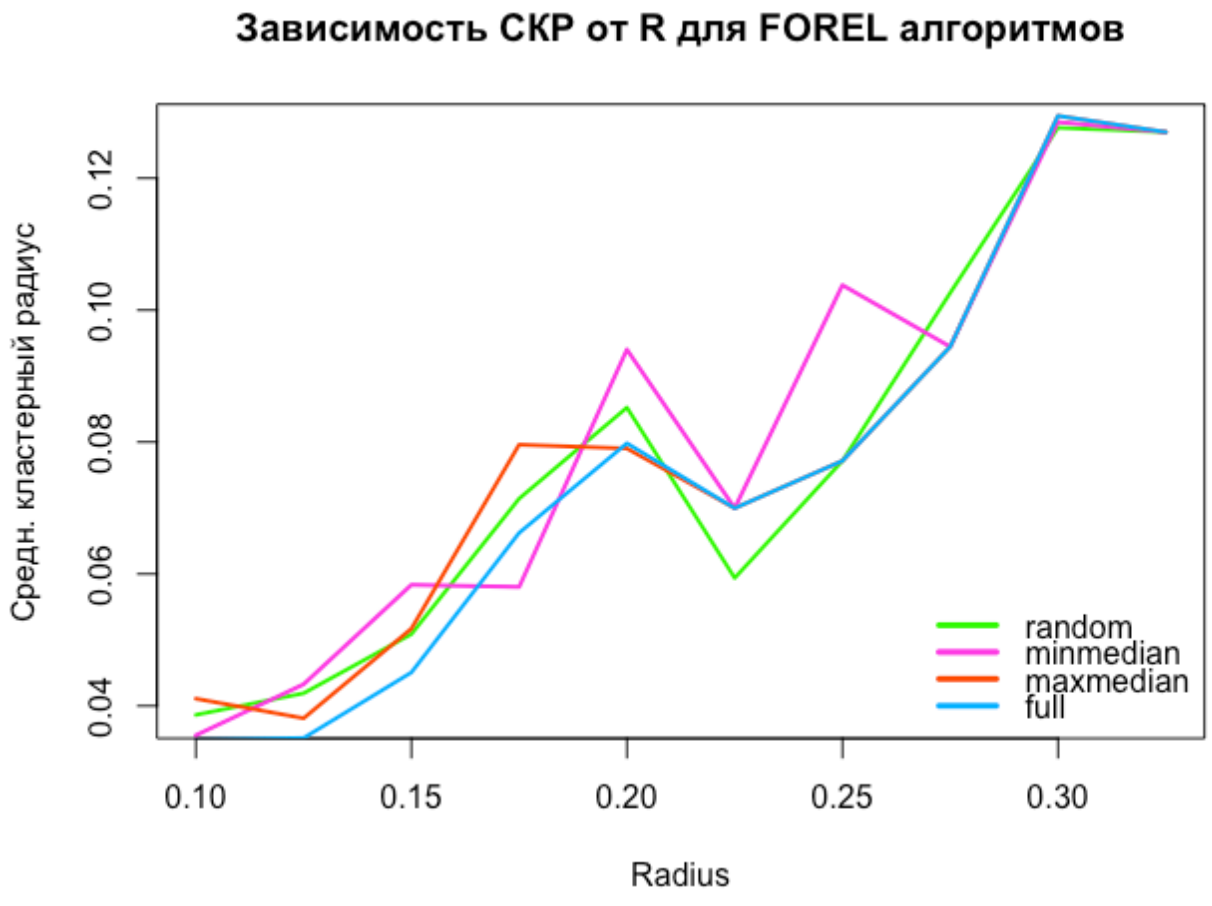
Сравнительная таблица

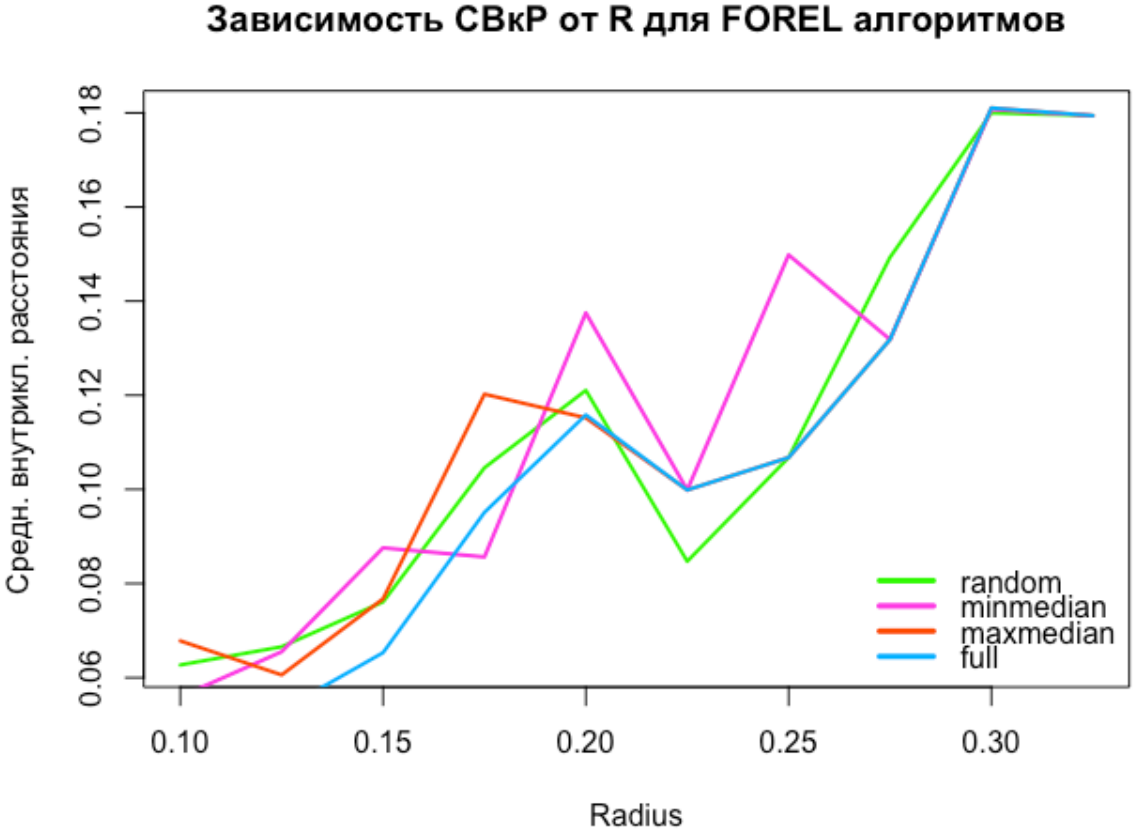
R	Средний кластерный радиус				Среднее внутрикластерное расстояние			
	Стандартный			С полным просмотром	Стандартный			С полным просмотром
	Maxmedian	Minmedian	Random		Maxmedian	Minmedian	Random	
0.1	0.04109562	0.03553614	0.03236344	0.03479994	0.06781942	0.05594195	0.05310429	0.05536840
0.125	0.03812300	0.04327640	0.03848928	0.03512750	0.06058621	0.06551348	0.06169720	0.05311393
0.15	0.05170254	0.05835503	0.05841075	0.04506971	0.07675329	0.08758054	0.08659528	0.06531378
0.175	0.07960202	0.05803598	0.05900465	0.06623173	0.12023622	0.08565217	0.08718532	0.09511398
0.2	0.07902859	0.09402532	0.08197344	0.07981205	0.11520496	0.13749269	0.12009480	0.11583243
0.225	0.06997255	0.06997255	0.05938874	0.06997255	0.09987377	0.09987377	0.08471683	0.09987377
0.25	0.07714720	0.10379381	0.10379381	0.07714720	0.10676493	0.14981927	0.14981927	0.10676493
0.275	0.09445128	0.09445128	0.09201983	0.09445128	0.13185564	0.13185564	0.12969719	0.13185564
0.3	0.12945532	0.12850745	0.09258588	0.12945532	0.18105053	0.18074903	0.13066532	0.18105053
0.325	0.12699336	0.12699336	0.12699336	0.12699336	0.17940961	0.17940961	0.17940961	0.17940961

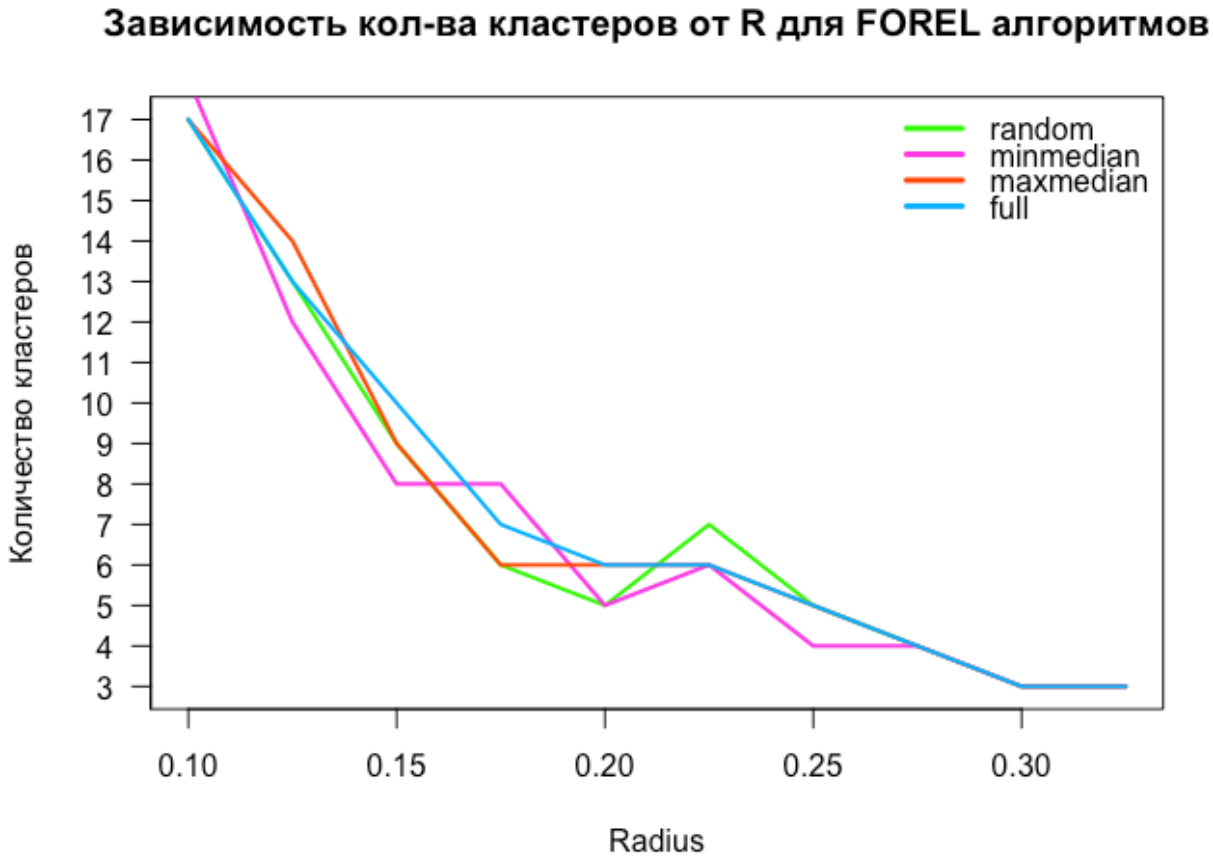
R	Количество кластеров			
	Стандартный			С полным просмотром
	Maxmedian	Minmedian	Random	
0.1	17	18	19	17
0.125	14	12	14	13

0.15	9	8	8	10	
0.175	6	8	7	7	
0.2	6	5	6	6	
0.225	6	6	7	6	
0.25	5	4	4	5	
0.275	4	4	4	4	
0.3	3	3	4	3	
0.325	3	3	3	3	

Для заданного радиуса в зависимости от способа выбора начальных приближений получается различное количество кластеров, в силу этого сложно оценивать качество полученных кластеров с помощью рассматриваемых характеристик.




Зависимость СВкР от R для FOREL алгоритмов
Средн. внутрикл. расстояния
0.06
0.08
0.10
0.12
0.14
0.16
0.18
0.10
0.15
0.20
0.25
0.30
Radius
random
minmedian
maxmedian
full


Зависимость кол-ва кластеров от R для FOREL алгоритмов
Количество кластеров
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
0.10
0.15
0.20
0.25
0.30
Radius
random
minmedian
maxmedian
full

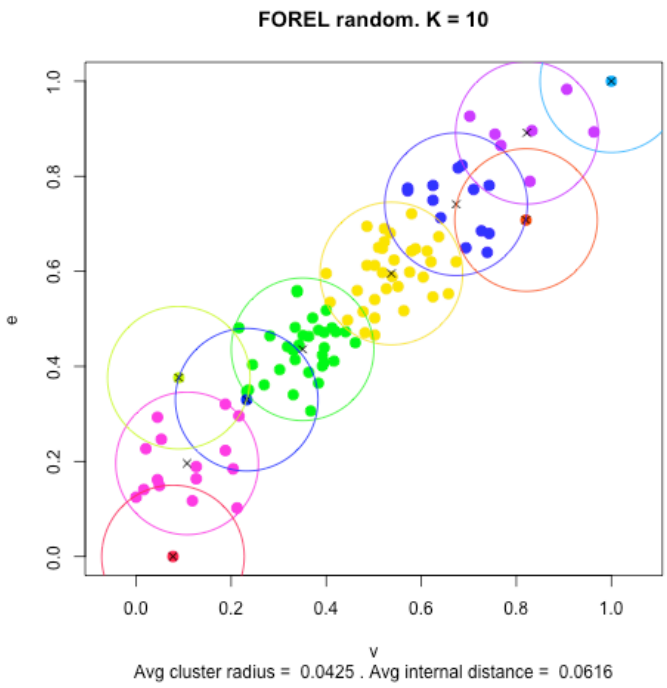
1) Результаты второго эксперимента. $R = 0.15$.

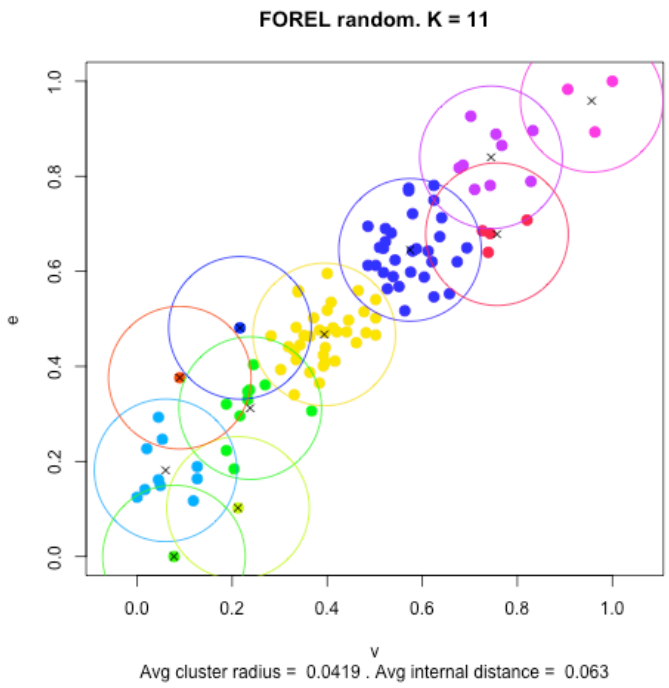
Значение радиуса было зафиксировано на уровне $R = 0.15$.

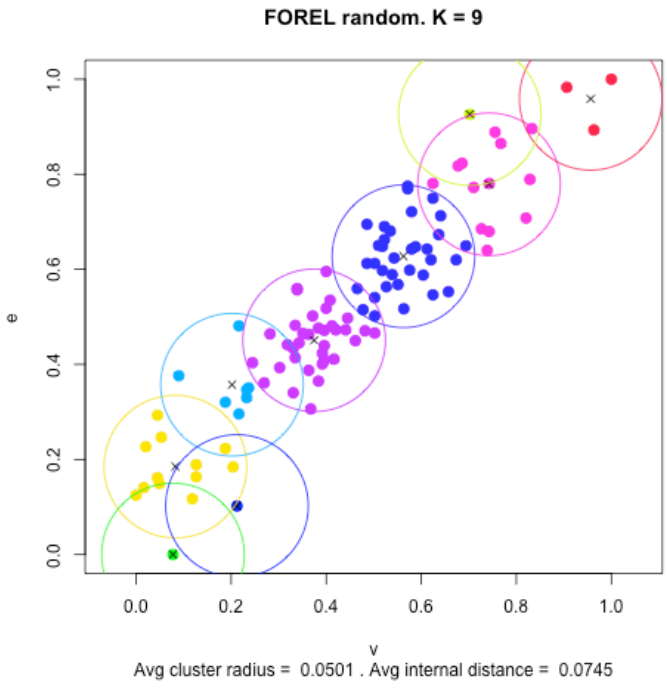
В этом эксперименте стандартный вариант алгоритма был запущен 10 раз. В результате получилось 10 различных разбиений на кластеры. Начальные приближения выбирались случайно. Далее представлена таблица с характеристиками разбиения.

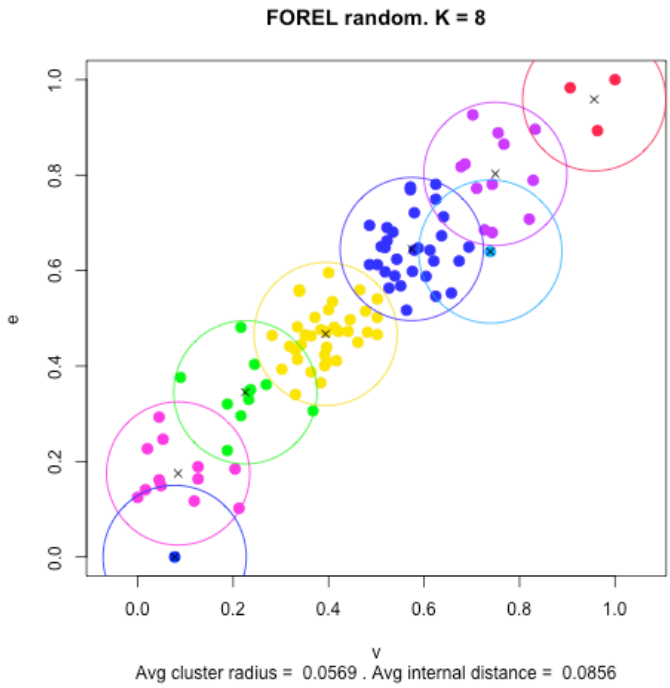
Получившиеся разбиения можно поделить на группы по количеству выделенных кластеров (K). На графиках представлены разбиения с лучшими характеристиками в своей группе.

№	Средний кластерный радиус	Среднее внутрикластерное расстояние	Количество кластеров
1	0.05252209	0.07770190	9
2	0.05382640	0.07967489	9
3	0.04245416	0.06156258	10
4	0.05841075	0.08659528	8
5	0.05054584	0.07553307	9
6	0.05072532	0.07607701	9
7	0.05087707	0.07609360	9
8	0.04185156	0.06298835	11
9	0.05013994	0.07454487	9
10	0.05688853	0.08563213	8


FOREL random. K = 10
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0425 . Avg internal distance = 0.0616


FOREL random. K = 11
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0419 . Avg internal distance = 0.063


FOREL random. K = 9
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0501 . Avg internal distance = 0.0745


FOREL random. K = 8
e
v
0.0
0.2
0.4
0.6
0.8
1.0
Avg cluster radius = 0.0569 . Avg internal distance = 0.0856

(4) Выводы

1. К-средних.

В ходе выполнения лабораторной работы ознакомились кластерным анализом, в частности, k-means - методом *k*-средних.

К недостаткам k-means можно отнести:

- необходимость заранее знать количество кластеров;
- чувствительность к выбору начальных центров кластеров.

В отчете использовался вариант k-means, когда пересчет центров кластеров осуществляется после распределения всех пар (v, E) по кластерам, пока центры не перестанут меняться (кодовое название: “Lazy”). Однако, разработанный скрипт также содержит другой вариант метода k-средних, где пересчет центров происходит после каждого добавления точки (кодовое название: “Long”).

В результате работы алгоритма k-средних, выборка была разделена на 7 кластеров.

2. Поиск сгущений.

В ходе лабораторной работы был реализован алгоритм поиска сгущений (ForEl) для кластеризации точек на двумерной плоскости в двух модификациях: «стандартной» и «с полным просмотром».

Для «стандартного» варианта было реализовано три способа выбора начальных приближений.

Было проведено два набора экспериментов: в первом варьировался радиус и способы поиска кластеров сравнивались между собой, во втором для фиксированного радиуса и случайных начальных приближений несколько раз запускался «стандартный» алгоритм для определения надежности характеристик полученного разбиения.

По результатам проведенного исследования можно сделать вывод, что при использовании метода поиска сгущений, значения среднего кластерного расстояния и среднего кластерного радиуса достигаются в случае успешного нахождения плотных зон (сгущений). Однако, при таком разбиении остаются точки, образующие кластеры нулевого радиуса и с нулевым кластерным расстоянием.

Также полученные значения среднего внутрикластерного расстояния и среднего кластерного радиуса довольно устойчивы, при фиксации радиуса кластера.

Основной задачей работы является визуализация результатов работы алгоритма. Результаты представлены в тексте работы в виде графиков и таблиц.

ЗАКЛЮЧЕНИЕ

В заключении, следует отметить, что статистическая модель включает в себя огромное множество методов, техник и инструментов для работы с экспериментальными данными, и в данной работе была изучена лишь малая часть этого аппарата.

В ходе работы были успешно достигнуты поставленные цели и в выполнены все задачи.

С помощью среды разработки RStudio был написан программный код, позволяющий производить первичную обработку выборки, получать изображения гистограммы, частотного полигона и ЭФР для интервального ряда, полученного из выборки, оценивать выборочные характеристики, проверять простые гипотезы о согласии с нормальным законом по критерию Пирсона, осуществлять обработку двумерных выборок, осуществлять корреляционный и регрессионный анализ.

Разработанные скрипты также позволяют проводить исследования и визуализировать методы кластеризации k-means и поиска сгущений.

ПРИЛОЖЕНИЕ А

ЛИСТИНГИ С ИСХОДНЫЙ КОДОМ НА ЯЗЫКЕ R

lab1.r

```
library("dplyr");
library("stats");
library("readr");

# входные данные

# задаем размер выборки
N <- 107
# считываем файл с выборкой
sample <- read_csv("input/sample_107.csv")

# ------------------

# рассчитываем количество интервалов
numClass <- 1 + floor(log2(N))

# ранжированные таблицы
a <- arrange(sample, v)
b <- arrange(sample, E)

# ранжированные массивы
v <- a[[1]]
e <- b[[2]]

# вариационные ряды
vStat <- count(a, v)
eStat <- count(b, E)

# размах выборок
vRange <- diff(range(v))
eRange <- diff(range(e))

# длины интервалов
vIntLength <- vRange/numClass
eIntLength <- eRange/numClass

# границы интервалов
vInterv <- seq(from = min(v), to = max(v), by = vRange/numClass)
eInterv <- seq(from = min(e), to = max(e), by = eRange/numClass)

# интервальные ряды
vIntSeq <- table(cut(v, vInterv, right = F, include.lowest = T))
vIntSeq <- as.data.frame(vIntSeq)
vIntSeq <- cbind(vIntSeq, vInterv[1:7] + (vRange/(2*numClass)))
colnames(vIntSeq) <- c('Intervals', 'Freq', "Mids")
vIntSeq <- as.tbl(vIntSeq)
```

```
eIntSeq <- table(cut(e, eInterv, right = F, include.lowest = T))
eIntSeq <- as.data.frame(eIntSeq)
eIntSeq <- cbind(eIntSeq, eInterv[1:7] + (eRange/(2*numClass)))
colnames(eIntSeq) <- c('Intervals', 'Freq', "Mids")
eIntSeq <- as.tbl(eIntSeq)

# Эмпирические функции распределения
# Абсолютная ЭФР
# Если определять F(x) = P(X < x) (лекция Середы)
plot(stepfun(vIntSeq$Mids, cumsum(c(0, vIntSeq$Freq)), right = F),
     verticals = F,
     lab = c(30, 15, 8),
     main = "Абс. ЭФР для v",
     ylab = "F(x) = P(X < x)",
     xlab = "v",
     col = "darkgreen",
     lwd = 2,
     cex.points = 1.5,
     pch = 60,
     xaxt = "n",
     yaxt = "n")
axis(1, at = vIntSeq$Mids, labels = round(vIntSeq$Mids, digits =
1))
axis(2, at = cumsum(c(0, vIntSeq$Freq)),
     labels = cumsum(c(0, vIntSeq$Freq)), las = 1)

## Это если определять F(x) = P(X <= x) (определяли раньше)
plot(stepfun(vIntSeq$Mids, cumsum(c(0, vIntSeq$Freq)), right = T),
     verticals = F,
     lab = c(30, 15, 8),
     main = "Абс. ЭФР для v",
     ylab = "для F(x) = P(X <= x)",
     xlab = "v",
     col = "darkgreen",
     lwd = 2,
     xaxt = "n",
     yaxt = "n")
axis(1, at = vIntSeq$Mids, labels = round(vIntSeq$Mids, digits =
1))
axis(2, at = cumsum(c(0, vIntSeq$Freq)),
     labels = cumsum(c(0, vIntSeq$Freq)), las = 1)

# Это F(x) = P(X < x)
plot(stepfun(eIntSeq$Mids, cumsum(c(0, eIntSeq$Freq)), right = F),
     verticals = F,
     lab = c(30, 15, 8),
     main = "Абс. ЭФР для E",
     ylab = "для F(x) = P(X < x)",
     xlab = "E",
     col = "darkblue",
```

```
     lwd = 2,
     cex.points = 1.5,
     pch = 60,
     xaxt = "n",
     yaxt = "n")
axis(1, at = eIntSeq$Mids, labels = round(eIntSeq$Mids, digits =
1))
axis(2, at = cumsum(c(0, eIntSeq$Freq)),
     labels = cumsum(c(0, eIntSeq$Freq)), las = 1)

# Это F(x) = P(X <= x)
plot(stepfun(eIntSeq$Mids, cumsum(c(0, eIntSeq$Freq)), right = T),
     verticals = F,
     lab = c(30, 15, 8),
     main = "Абс. ЭФР для E",
     ylab = "для F(x) = P(X <= x)",
     xlab = "E",
     col = "darkblue",
     lwd = 2,
     xaxt = "n",
     yaxt = "n")
axis(1, at = eIntSeq$Mids, labels = round(eIntSeq$Mids, digits =
1))
axis(2, at = cumsum(c(0, eIntSeq$Freq)),
     labels = cumsum(c(0, eIntSeq$Freq)), las = 1)

# Относительная ЭФР
# Это F(x) = P(X < x) для v
plot(stepfun(vIntSeq$Mids, cumsum(c(0, vIntSeq$Freq))/N, right =
F),
     verticals = F,
     lab = c(15, 5, 8),
     main = "Отн. ЭФР для v",
     ylab = "для F(x) = P(X < x)",
     xlab = "v",
     col = "darkgreen",
     lwd = 2,
     cex.points = 1.5,
     pch = 60,
     xaxt = "n",
     yaxt = "n")
axis(1, at = vIntSeq$Mids, labels = round(vIntSeq$Mids, digits =
2))
axis(2, at = cumsum(c(0, vIntSeq$Freq))/N,
     labels = round(cumsum(c(0, vIntSeq$Freq))/N, digits = 2), las
= 1)

# Это F(x) = P(X <= x) для v
plot(stepfun(vIntSeq$Mids, cumsum(c(0, vIntSeq$Freq))/N, right =
T),
     verticals = F,
```

```
     lab = c(15, 5, 8),
     main = "Отн. ЭФР для v",
     ylab = "для F(x) = P(X <= x)",
     xlab = "v",
     col = "darkgreen",
     lwd = 2,
     xaxt = "n",
     yaxt = "n")
axis(1, at = vIntSeq$Mids, labels = round(vIntSeq$Mids, digits =
2))
axis(2, at = cumsum(c(0, vIntSeq$Freq))/N,
     labels = round(cumsum(c(0, vIntSeq$Freq))/N, digits = 2), las
= 1)

# Это F(x) = P(X < x) для E
plot(stepfun(eIntSeq$Mids, cumsum(c(0, eIntSeq$Freq))/N, right =
F),
     verticals = F,
     lab = c(15, 5, 8),
     main = "Отн. ЭФР для E",
     ylab = "для F(x) = P(X < x)",
     xlab = "E",
     col = "darkblue",
     lwd = 2,
     cex.points = 1.5,
     pch = 60,
     xaxt = "n",
     yaxt = "n")
axis(1, at = eIntSeq$Mids, labels = round(eIntSeq$Mids, digits =
2))
axis(2, at = cumsum(c(0, eIntSeq$Freq))/N,
     labels = round(cumsum(c(0, eIntSeq$Freq))/N, digits = 2), las
= 1)

# Это F(x) = P(X <= x) для E
plot(stepfun(eIntSeq$Mids, cumsum(c(0, eIntSeq$Freq))/N, right =
T),
     verticals = F,
     lab = c(15, 5, 8),
     main = "Отн. ЭФР для E",
     ylab = "для F(x) = P(X <= x)",
     xlab = "E",
     col = "darkblue",
     lwd = 2,
     xaxt = "n",
     yaxt = "n")
axis(1, at = eIntSeq$Mids, labels = round(eIntSeq$Mids, digits =
2))
axis(2, at = cumsum(c(0, eIntSeq$Freq))/N,
     labels = round(cumsum(c(0, eIntSeq$Freq))/N, digits = 2), las
= 1)
```

```
###vecdf <- ecdf(rep(vInterv, vIntSeq$Freq), right = T)
###eecdf <- ecdf(rep(eIntSeq$Mids, eIntSeq$Freq))

# Гистограммы
vhist <- hist(v, breaks = vInterv, right = F, include.lowest = T)
#vhist$density <- vhist$density * vIntLength

{
  plot(vhist,
       #col = "honeydew",
       col = "lightgreen",
       main = "Гистограмма для v",
       ylab = "Абс. частота",
       xlab = "v",
       xaxt = "n",
       yaxt = "n")
  axis(1, at = c(vInterv, vIntSeq$Mids),
       labels = round( c(vInterv, vIntSeq$Mids), digits = 1 ) )
  axis(2, at = c(0, vhist$counts), las = 1,
       labels = c(0, vhist$counts))
}

{
  plot(vhist,
       freq = F,
       #col = "honeydew3",
       col = "darkgreen",
       main = "Гистограмма для v",
       ylab = "",
       xlab = "v",
       xaxt = "n",
       yaxt = "n")
  mtext(text = "Плотность", side = 3, line = 0,
        las = 1, adj = 1, padj=0, at = 345)
  axis(1, at = c(vInterv, vIntSeq$Mids),
       labels = round( c(vInterv, vIntSeq$Mids), digits = 1 ) )
  axis(2, at = c(0, vhist$density, 1), las = 1,
       labels = round( c(0, vhist$density, 1), digits = 4 ) )
}

ehist <- hist(e, breaks = eInterv, right = F, include.lowest = T)
#ehist$density <- ehist$density * eIntLength

{
  plot(ehist,
       col = "blue",
       main = "Гистограмма для E",
       ylab = "Абс. частота",
       xlab = "E",
```

```
      xaxt = "n",
      yaxt = "n")
  axis(1, at = c(eInterv, eIntSeq$Mids),
      labels = round( c(eInterv, eIntSeq$Mids), digits = 1 ) )
  axis(2, at = c(0, ehist$counts), las = 1,
      labels = c(0, ehist$counts))
}

{
  plot(ehist,
      freq = F,
      main = "Гистограмма для E",
      ylab = "",
      xlab = "E",
      col = "darkblue",
      xaxt = "n",
      yaxt = "n")
  mtext(text = "Плотность", side = 3, line = 0,
       las = 1, adj = 1, padj=0, at = 80)
  axis(1, at = c(eInterv, eIntSeq$Mids),
      labels = round( c(eInterv, eIntSeq$Mids), digits = 1 ) )
  axis(2, at = c(0, ehist$density, 1), las = 1,
      labels = round( c(0, ehist$density, 1), digits = 4 ) )
}

# Полигоны
{
  plot(vhist$mids, vhist$counts,
      #  xlim = c(min(vhist$breaks), max(vhist$breaks)),
      ylim = c(0, max(vhist$counts)),
      type = "o",
      xlab = "v",
      ylab = "Частота",
      main = "Частотный полигон для v",
      pch = 19,
      xaxt = "n",
      yaxt = "n")
  axis(1, at = vIntSeq$Mids,
      labels = round(vIntSeq$Mids, digits = 1 ) )
  axis(2, at = c(0, vhist$counts), las = 1,
      labels = c(0, vhist$counts))
  polygon(vhist$mids[c(1, 1:7, 7)],
          c(0, vhist$counts, 0),
          border = "darkgreen",
          col = "honeydew")
}

{
  plot(vhist$mids, vhist$density,
      #     xlim = range(vhist$breaks),
      ylim = c(0, max(vhist$density)),
```

```
      type = "o",
      xlab = "v",
      ylab = "",
      main = "Полигон плотности для v",
      pch = 19,
      xaxt = "n",
      yaxt = "n")
  mtext(text = "Плотность", side = 3, line = 0,
       las = 1, adj = 1, padj=0, at = 365)
  axis(1, at = vIntSeq$Mids,
      labels = round(vIntSeq$Mids, digits = 1 ) )
  axis(2, at = c(0, vhist$density, 1), las = 1,
      labels = round( c(0, vhist$density, 1), digits = 4 ) )
  polygon(vhist$mids[c(1, 1:7, 7)],
          c(0, vhist$density, 0),
          border = "darkgreen",
          col = "lightgreen")
}

{
  plot(ehist$mids, ehist$counts,
      #    xlim = c(min(ehist$breaks), max(ehist$breaks)),
      ylim = c(0, max(ehist$counts)),
      type = "o",
      xlab = "E",
      ylab = "Частота",
      main = "Частотный полигон для E",
      pch = 19,
      xaxt = "n",
      yaxt = "n")
  axis(1, at = eIntSeq$Mids,
      labels = round(eIntSeq$Mids, digits = 1 ) )
  axis(2, at = c(0, ehist$counts), las = 1,
      labels = c(0, ehist$counts))
  polygon(ehist$mids[c(1, 1:7, 7)],
          c(0, ehist$counts, 0),
          border = "darkblue",
          col = "azure")
}

{
  plot(ehist$mids, ehist$density,
      #    xlim = range(ehist$breaks),
      ylim = c(0, max(ehist$density)),
      type = "o",
      xlab = "E",
      ylab = "",
      main = "Полигон плотности для E",
      pch = 19,
      xaxt = "n",
      yaxt = "n")
```

```
  mtext(text = "Плотность", side = 3, line = 0,
        las = 1, adj = 1, padj=0, at = 88)
  axis(1, at = eIntSeq$Mids,
       labels = round(eIntSeq$Mids, digits = 1 ) )
  axis(2, at = c(0, ehist$density, 1), las = 1,
       labels = round( c(0, ehist$density, 1), digits = 4 ) )
  polygon(ehist$mids[c(1, 1:7, 7)],
          c(0, ehist$density, 0),
          border = "darkblue",
          col = "lightblue")
}
```

lab2.r

```
# vIntSeq
# eIntSeq

colnames(vIntSeq) <- c('Intervals', 'Counts', "Mids")
colnames(eIntSeq) <- c('Intervals', 'Counts', "Mids")

#добавим частоты
vIntSeq <- cbind(vIntSeq, Freq = vIntSeq$Counts/N)
eIntSeq <- cbind(eIntSeq, Freq = eIntSeq$Counts/N)

#условные варианты
vIntSeq <- cbind(vIntSeq, ui = (vIntSeq$Mids - vIntSeq$Mids[4])/
vIntLength)
eIntSeq <- cbind(eIntSeq, ui = (eIntSeq$Mids - eIntSeq$Mids[4])/
eIntLength)
vC <- vIntSeq$Mids[4]
eC <- eIntSeq$Mids[4]

# чисто для таблицы из отчета
#vIntSeq <- cbind(vIntSeq, step1 = (vIntSeq$ui * vIntSeq$Count))
#vIntSeq <- cbind(vIntSeq, step2 = (vIntSeq$ui * vIntSeq$step1))
#vIntSeq <- cbind(vIntSeq, step3 = (vIntSeq$ui * vIntSeq$step2))
#vIntSeq <- cbind(vIntSeq, step4 = (vIntSeq$ui * vIntSeq$step3))
# (x_i + 1)^4 * n_i
#vIntSeq <- cbind(vIntSeq, check_ = ((vIntSeq$ui + 1)^4 *
vIntSeq$Count))


#слагаемые первого условного момента
vIntSeq <- cbind(vIntSeq, M1 = (vIntSeq$ui * vIntSeq$Freq))
eIntSeq <- cbind(eIntSeq, M1 = (eIntSeq$ui * eIntSeq$Freq))

#слагаемые второго условного момента
vIntSeq <- cbind(vIntSeq, M2 = (vIntSeq$ui * vIntSeq$M1))
eIntSeq <- cbind(eIntSeq, M2 = (eIntSeq$ui * eIntSeq$M1))

#слагаемые третьего условного момента
vIntSeq <- cbind(vIntSeq, M3 = (vIntSeq$ui * vIntSeq$M2))
eIntSeq <- cbind(eIntSeq, M3 = (eIntSeq$ui * eIntSeq$M2))

#слагаемые четвертого условного момента
vIntSeq <- cbind(vIntSeq, M4 = (vIntSeq$ui * vIntSeq$M3))
eIntSeq <- cbind(eIntSeq, M4 = (eIntSeq$ui * eIntSeq$M3))

# проверочный столбец
vIntSeq <- cbind(vIntSeq, check_ = ((vIntSeq$ui + 1)^4 *
vIntSeq$Freq))
eIntSeq <- cbind(eIntSeq, check_ = ((eIntSeq$ui + 1)^4 *
eIntSeq$Freq))
```

```
# чисто для таблицы из отчета
my_eIntSeq <- cbind(eIntSeq, step1 = (eIntSeq$ui * eIntSeq$Count))
my_eIntSeq <- cbind(my_eIntSeq, step2 = (my_eIntSeq$ui *
my_eIntSeq$step1))
my_eIntSeq <- cbind(my_eIntSeq, step3 = (my_eIntSeq$ui *
my_eIntSeq$step2))
my_eIntSeq <- cbind(my_eIntSeq, step4 = (my_eIntSeq$ui *
my_eIntSeq$step3))
# (x_i + 1)^4 * n_i
my_eIntSeq <- cbind(my_eIntSeq, check_ = ((my_eIntSeq$ui + 1)^4 *
my_eIntSeq$Count))

moments <- data.frame(
  v = colSums(vIntSeq[c("M1", "M2", "M3", "M4")]),
  e = colSums(eIntSeq[c("M1", "M2", "M3", "M4")]))

###проверка
## vIntSeq$M4 + 4*vIntSeq$M3 + 6*vIntSeq$M2 + 4*vIntSeq$M1 +
vIntSeq$Freq
## ((vIntSeq$ui + 1)^4) * vIntSeq$Freq
#
## sum(eIntSeq$M4 + 4*eIntSeq$M3 + 6*eIntSeq$M2 + 4*eIntSeq$M1 +
eIntSeq$Freq)
# sum(((eIntSeq$ui + 1)^4) * eIntSeq$Freq)
###

#выборочное среднее
moments <- rbind(moments, mean = c(moments$v[1] * vIntLength + vC,
                                   moments$e[1] * eIntLength +
eC))

# дисперсия
moments <- rbind(moments, dispersion = c((moments$v[2] -
moments$v[1]^2)*(vIntLength^2),
                                         (moments$e[2] -
moments$e[1]^2)*(eIntLength^2)))
#исправленная дисперсия (несмещенная)
moments["var",] <- moments["dispersion",] * (N/(N-1))
moments <- moments[-c(6),]

#несмещенное средн.кв. отлонение
moments <- rbind(moments, deviation = sqrt(moments["var",]))

#оценка асимметрии
moments <- rbind(moments,
                 asymmetry = (moments[3,] -
3*moments[2,]*moments[1,] + 2*(moments[1,]^3))*((c(vIntLength,
eIntLength)/moments["deviation",])^3))

#оценка эксцесса
```

```
iod <- c(vIntLength, eIntLength)/moments["deviation",]
iod <- iod^4
m4 <- moments[4,] - 4*moments[3,]*moments[1,] +
6*moments[2,]*(moments[1,])^2 - 3*(moments[1,]^4)
moments <- rbind(moments, excess = m4*iod - 3)
```

lab3.r

```
#интервальные оценки для мат.ожидания
##надежность
y <- 0.95
### распределение Стьюдента - симметричное, поэтому делим на два
ty <- -qt((1-y)/2, df = N-1)

deviation <- moments$v[8];
conf <- (deviation * ty)/sqrt(N)
### доверительный интервал для мат.ожидания v
vMeanConfInt <- c(moments$v[5] - conf, moments$v[5] + conf)

deviation <- moments$e[8]
conf <- (deviation * ty)/sqrt(N)
### доверительный интервал для мат.ожидания E
eMeanConfInt <- c(moments$e[5] - conf, moments$e[5] + conf)

#интервальные оценки для СКВО
vDevConfInt <- sqrt((N-1)*moments["var","v"]/qchisq(c((1+y)/2, (1-
y)/2), df = N-1))
eDevConfInt <- sqrt((N-1)*moments["var","e"]/qchisq(c((1+y)/2, (1-
y)/2), df = N-1))

#проверка простой гипотезы о нормальном распределении
a <- pnorm(vInterv[2:7], mean = moments["mean", "v"], sd =
moments["deviation", "v"])
vHyp <- cbind(vIntSeq[, 1:4], pThLB = c(0, a), pThRB = c(a, 1))
vHyp <- cbind(vHyp, ThFreq = vHyp$pThRB - vHyp$pThLB)
vHyp <- cbind(vHyp, ThCounts = vHyp$ThFreq * N)
vChiSq <- sum( ((vHyp$Counts - vHyp$ThCounts)^2) / vHyp$ThCounts )
## a - уровень значимости, вероятность отклонить верную гипотезу
a <- 0.05
if (vChiSq > qchisq(1-a, df = numClass - 3)) 'Reject at a = 0.05'
else 'Accept at a = 0.05'

a <- pnorm(eInterv[2:7], mean = moments["mean", "e"], sd =
moments["deviation", "e"])
eHyp <- cbind(eIntSeq[, 1:4], pThLB = c(0, a), pThRB = c(a, 1))
eHyp <- cbind(eHyp, ThFreq = eHyp$pThRB - eHyp$pThLB)
eHyp <- cbind(eHyp, ThCounts = eHyp$ThFreq * N)
eChiSq <- sum( ((eHyp$Counts - eHyp$ThCounts)^2) / eHyp$ThCounts )
## a - уровень значимости, вероятность отклонить верную гипотезу
a <- 0.05
```

```
test <- qchisq(1-a, df = numClass - 3)
if (eChiSq > qchisq(1-a, df = numClass - 3)) 'Reject at a = 0.05'
else 'Accept at a = 0.05'
```

lab4.r

```
#корреляционная таблица
corTab <- table(cut(sample$v, vInterv, right = F, include.lowest =
T),
                cut(sample$E, eInterv, right = F, include.lowest =
T))

## обращение к элементам corTab[x, y], x - номер интервала для
выборки v (от 1 до 7),
##                                     y - номер интервала для
выборки E

#коэффициент корреляции
corMat <- as.matrix(corTab)

rb1 <- t(t(corMat * vIntSeq$Mids) * eIntSeq$Mids)
rb1 <- sum(rb1) - N*moments["mean", "v"]*moments["mean", "e"]
rb1 <- rb1/(N*moments["deviation", "v"]*moments["deviation", "e"])

rb2 <- t(t(corMat * vIntSeq$ui) * eIntSeq$ui)
rb2 <- sum(rb2) - N*moments["M1", "v"]*moments["M1", "e"]
rb2 <- rb2*vIntLength*eIntLength
rb2 <- rb2/(N*moments["deviation", "v"]*moments["deviation", "e"])

#доверительный интервал с надежностью a
a <- 0.95
z <- 0.5*log((1+rb2)/(1-rb2))
z <- z + c(-qnorm((1+a)/2)/sqrt(N-3), qnorm((1+a)/2)/sqrt(N-3))
rConfInt <- (exp(2*z) - 1)/(exp(2*z) + 1)

#проверка гипотезы H: r = 0
## a - уровень значимости, вероятность отклонить верную гипотезу
a <- 0.05
Tr <- rb2 * sqrt(numClass-2)/(1 - rb2^2)
ty <- qt(1-a/2, df = numClass-2) #критическое значение
if (Tr > ty) 'Reject at a = 0.05' else 'Accept at a = 0.05'
```

lab5.r

```
xv <- function(y) moments["mean", "v"] + rb1 *
(moments["deviation", "v"]/moments["deviation", "e"])*(y -
moments["mean", "e"])

{
  plot(sample)
  l <- legend("bottomright",
              legend = c("y(x)", "x(y)"),
              col = c("red", "blue"),
              lwd = 3,
              plot = T)
```

```
}

{
  plot(sample,
       pch = 19,
       cex = 0.9,
       xaxt = "n",
       yaxt = "n",
       ylab = "E",
       main = "Выборка и прямые среднеквадратичной регрессии")
  axis(1, at = vInterv,
       labels = round(vInterv, digits = 1 ) )
  axis(2, at = eInterv, las = 1,
       labels = round(eInterv, digits = 1 ) )
  curve(moments["mean", "e"] +
          rb1 * (moments["deviation", "e"]/moments["deviation",
"v"])*(x - moments["mean", "v"]),
        min(v), max(v),
        n = 10000,
        add = T,
        lwd = 2,
        col = "red")
  lines(xv( seq(min(e), max(e), by = 0.01) ),  seq(min(e), max(e),
by = 0.01),
        type = "l",
        lwd = 2,
        col = "blue")

  legend(x = c(l$rect$left, l$rect$left + l$rect$w),
         y = c(l$rect$top, l$rect$top - l$rect$h),
         legend = c("y(x)", "x(y)"),
         bty = "n",
         col = c("red", "blue"),
         lwd = 3,
         y.intersp = 0.5,
         title = "Прямые лин.регрессии",
         cex = 1.1,
         text.width = 20)
}

#остаточные дисперсии
resid_var <- c(moments["var", "e"]*(1 - rb2^2),
               moments["var", "v"]*(1 - rb2^2))

# групповые оценки для e (xi) и v (yi)
xi <- data.frame(Intervals = vIntSeq$Intervals, Mids =
vIntSeq$Mids)
yi <- data.frame(Intervals = eIntSeq$Intervals, Mids =
eIntSeq$Mids)
```

```
xi <- cbind(xi, yHits = rowSums(corMat)) #то же самое, что yHits =
vIntSeq$Counts
yi <- cbind(yi, xHits = colSums(corMat))

xi <- cbind(xi, Avg_y_for_Xint = rowSums(t(t(corMat) *
eIntSeq$Mids))/vIntSeq$Counts)
yi <- cbind(yi, Avg_x_for_Yint = colSums(corMat * vIntSeq$Mids)/
eIntSeq$Counts)

xi <- cbind(xi,
            deltaSq = rowSums(corMat * (rep.int(1, 7) %*%
t(yi$Mids) - xi$Avg_y_for_Xint)^2)/xi$yHits)
#дельта квадрат горизонтальная
yi <- cbind(yi,
            deltaSq = colSums(corMat * t((rep.int(1, 7) %*%
t(xi$Mids) - yi$Avg_x_for_Yint)^2))/yi$xHits)

# внутригрупповые дисперсии
Dx_in <- sum(yi$xHits * yi$deltaSq)/N
Dy_in <- sum(xi$yHits * xi$deltaSq)/N

# межгрупповые дисперсии
Dx_out <- sum(yi$xHits * (yi$Avg_x_for_Yint - moments["mean",
"v"])^2)/N
Dy_out <- sum(xi$yHits * (xi$Avg_y_for_Xint - moments["mean",
"e"])^2)/N

# корреляционное отношение
#Dy_out/(Dy_out + Dy_in)
etaSq <- Dx_out/(Dx_out + Dx_in)
etaSq_ <- Dy_out/(Dy_out + Dy_in)
sqrt(Dx_out/(Dx_out + Dx_in)) > rb1


# апроксимация параболой
## для y ~ x
t <- solve(matrix(c(sum(v^4), sum(v^3), sum(v^2),
                    sum(v^3), sum(v^2), sum(v),
                    sum(v^2), sum(v), N),
                  nrow = 3),
           c(sum(e*v^2),
             sum(e*v),
             sum(e)) )

## для x ~ y

t <- c(t,
       solve(matrix(c(sum(e^4), sum(e^3), sum(e^2),
                      sum(e^3), sum(e^2), sum(e),
                      sum(e^2), sum(e), N),
                    nrow = 3),
```

```
              c(sum(v*e^2),
                sum(v*e),
                sum(v)) )
)

{
  plot(sample,
       pch = 19,
       cex = 0.9,
       xaxt = "n",
       yaxt = "n",
       ylab = "E",
       main = "Выборка и параболы ср.-кв. регрессии")

  legend(x = c(l$rect$left, l$rect$left + l$rect$w),
         y = c(l$rect$top, l$rect$top - l$rect$h),
         inset = 0,
         legend = c("y(x)", "x(y)"),
         col = c("red", "blue"),
         bty = "n",
         title = "Параболы",
         lwd = 3,
         y.intersp = 0.5,
         cex = 1.1,
         text.width = 20)

  axis(1, at = vInterv,
       labels = round(vInterv, digits = 1 ) )
  axis(2, at = eInterv, las = 1,
       labels = round(eInterv, digits = 1 ) )
  curve(t[1]*x^2 + t[2]*x + t[3], min(v), max(v),
        n = 10000,
        add = T,
        lwd = 2,
        col = "red")
  lines(sapply(seq(min(e), max(e), by = 0.01), function(x)
t[4]*x^2 + t[5]*x + t[6]),
        seq(min(e), max(e), by = 0.01),
        type = "l",
        lwd = 2,
        col = "blue")
}
```

lab6.r

```
#reevaluateCenters <- function(clusters, centers){
#  for(i in 1:nrow(centers)){
#    centers[i, ] <- c(mean(clusters$v[which(clusters$clNum ==
i)]),
#                      mean(clusters$e[which(clusters$clNum ==
i)]))
```

```
#  }
#  return(centers)
#}

# reuire(purrr)

makeClusters <- function(clusteredPoints, centers){
  for(i in 1:nrow(clusteredPoints)){
    clusteredPoints[i, 3] <-
which.min(dist(rbind(clusteredPoints[i, 1:2], centers))
[1:nrow(centers)])
  }
  return(clusteredPoints)
}

clusterDensity <- function(clusters, centers, rad = T){
  clDen <- 0
  if (rad) {
    for (i in 1:nrow(centers)) {
      d <- rbind(centers[i, ], clusters[which(clusters$clNum ==
i), 1:2])
      clDen <- clDen + mean(dist(d)[1:(nrow(d)-1)])
    }
  } else {
    for (i in 1:nrow(centers)) {
      dd <- mean(dist(clusters[which(clusters$clNum == i), 1:2]))
      if (is.finite(dd)) clDen <- clDen + dd
    }
  }
  return(clDen/nrow(centers))
}

plotClusters <- function(clusters, K, alg_option, quality1 = NULL,
quality2 = NULL){
  plot(clusters[, 1:2],
       asp = 1, xlim = c(0, 1), ylim = c(0, 1),
       pch = 19,
       col = clusters$clNum,
       cex = 1.3,
       main = paste0(alg_option, ". K = ", K))
  if ( (!is.null(quality1)) && (!is.null(quality2)) ) title(sub =
paste("Avg cluster radius = ", round(quality1, digits = 4),

". Avg internal distance = ", round(quality2, digits = 4)))
  points(clusters[which(is.na(clusters$clNum)), 1:2])
}

reevaluateCenterOfCluster <- function(cluster){
  return(c(mean(cluster[[1]]), mean(cluster[[2]])))
}
```

```
reevaluateCenters <- function(clusters, K, centers){
  for(i in 1:K){
    centers[i, ] <-
reevaluateCenterOfCluster(clusters[which(clusters$clNum == i),])
  }
  return(centers)
}

initCenters <- function(sample, K, method){
  N <- nrow(sample)
  switch (method,
          random ={
            centers <- sample[sample(N, K),]
          },
          equidistant ={
            period <- floor(N/K)
            ind <- seq(from = floor(median(1:period)), to = N, by
= period)
            ind <- match(sort(sample$v)[ind], sample$v)
            centers <- sample[ind,]
          }
  )
  return(centers)
}

initClusters <- function(sample, centers){
  clusters <- left_join(sample, cbind(centers, clNum =
1:nrow(centers)))
  return(clusters)
}

determineClusterForPoint <- function(centers, pointXY){
  return(which.min(dist(rbind(pointXY, centers))
[1:nrow(centers)]))
}

kmeansOnePointAtATime <- function(clusteredPoints, centers){
  plotClusters(clusteredPoints, nrow(centers), alg_option =
"Long")
  for (i in 1:nrow(clusteredPoints)){
    if(is.na(clusteredPoints[i, 3])){
      j <- determineClusterForPoint(centers, clusteredPoints[i,
1:2])
      clusteredPoints[i, 3] <- j
      centers[j, ] <-
reevaluateCenterOfCluster(clusteredPoints[which(clusteredPoints$cl
Num == j), ])
      plotClusters(clusteredPoints, nrow(centers), alg_option =
"Long")
    }
  }
```

```
  t <- 0
  repeat {
    clustersQuality <- clusterDensity(clusteredPoints, centers)
    for(i in 1:nrow(clusteredPoints)){
      j <- determineClusterForPoint(centers, clusteredPoints[i,
1:2])
      if(clusteredPoints[i, 3] != j){
        clusteredPoints[i, 3] <- j
        centers[j, ] <-
reevaluateCenterOfCluster(clusteredPoints[which(clusteredPoints$cl
Num == j), ])
        plotClusters(clusteredPoints, nrow(centers),
                     alg_option = "Long",
                     quality1 = clustersQuality,
                     quality2 = clusterDensity(clusteredPoints,
centers, rad = F))
      }
    }
    if (clustersQuality == clusterDensity(clusteredPoints,
centers)) t <- t+1
    else t <- 0
    if (t == 5) break
  }
  centers <- reevaluateCenters(clusteredPoints, nrow(centers),
centers)
  print(clusterDensity(clusteredPoints, centers))
  print(clusterDensity(clusteredPoints, centers, F))
  for(i in 1:10) plotClusters(clusteredPoints, nrow(centers),
                              alg_option = "Long",
                              quality1 =
clusterDensity(clusteredPoints, centers),
                              quality2 =
clusterDensity(clusteredPoints, centers, rad = F))
  return(clusteredPoints)
}

kmeansLazy <- function(clusteredPoints, centers){
  plotClusters(clusteredPoints, nrow(centers), alg_option =
"Lazy")
  clusteredPoints <- makeClusters(clusteredPoints, centers)
  centers <- reevaluateCenters(clusteredPoints, nrow(centers),
centers)
  clustersQuality <- clusterDensity(clusteredPoints, centers)
  plotClusters(clusteredPoints, nrow(centers),
               alg_option = "Lazy",
               quality1 = clustersQuality,
               quality2 = clusterDensity(clusteredPoints, centers,
rad = F))
  t <- 0

  repeat {
```

```
    clusteredPoints <- makeClusters(clusteredPoints, centers)
    centers <- reevaluateCenters(clusteredPoints, nrow(centers),
centers)
    plotClusters(clusteredPoints, nrow(centers),
                 alg_option = "Lazy",
                 quality1 = clustersQuality,
                 quality2 = clusterDensity(clusteredPoints,
centers, rad = F))
    if (clustersQuality == clusterDensity(clusteredPoints,
centers)) t <- t+1
    else t <- 0
    if (t == 5) break
    clustersQuality <- clusterDensity(clusteredPoints, centers)
  }
  for(i in 1:10) plotClusters(clusteredPoints, nrow(centers),
                              alg_option = "Lazy",
                              quality1 =
clusterDensity(clusteredPoints, centers),
                              quality2 =
clusterDensity(clusteredPoints, centers, rad = F))
  return(clusteredPoints)
}

mykmeans <- function(sample, K, center_method = "random",
algoritm_option = "Lazy"){
  switch (center_method,
          random = {centers <- initCenters(sample, K, "random")},
          equidistant = {centers <- initCenters(sample, K,
"equidistant")}
  )
  png(file="example%03d.png")
  palette(sample(rainbow(K)))
  switch (algoritm_option,
          Lazy = {clusters <- kmeansLazy(initClusters(sample,
centers), centers)},
          Long = {clusters <-
kmeansOnePointAtATime(initClusters(sample, centers), centers)}
  )
  palette("default")
  dev.off()
  return(clusters)
}

mykmeans_both <- function(sample, K, center_method = "random"){
  switch (center_method,
          random = {centers <- initCenters(sample, K, "random")},
          equidistant = {centers <- initCenters(sample, K,
"equidistant")}
  )
  palette(sample(rainbow(K)))
```

```
  newdir <- paste0("randomLazy",K)
  dir.create(newdir)
  cwd <- getwd()
  setwd(newdir)

  png(file="rand_lazy%03d.png")

  clustered_points <- kmeansLazy(initClusters(sample, centers),
centers)
  new_centers <- reevaluateCenters(clustered_points, K, centers)
  result <- c(clusterDensity(clustered_points, new_centers),
              clusterDensity(clustered_points, new_centers, F))
  dev.off()
  setwd(cwd)

  newdir <- paste0("randomLong",K)
  dir.create(newdir)
  cwd <- getwd()
  setwd(newdir)
  png(file="rand_long%03d.png")
  clustered_points <- kmeansOnePointAtATime(initClusters(sample,
centers), centers)
  new_centers <- reevaluateCenters(clustered_points, K, centers)
  result <- c(result,
              clusterDensity(clustered_points, new_centers),
              clusterDensity(clustered_points, new_centers, F))
  dev.off()
  setwd(cwd)

  palette("default")

  return(result)
}

library("purrr", lib.loc="~/R/R-3.5.2/library")

ind <- function(i, j, n){
  # if(j == i) return(0)
  # if(j < i) return(N*(j -1) + i - j - (j*(j-1))/2)
  # N*(i -1) + j - i - (i*(i-1))/2

  result <-n*(i -1) + j - i - (i*(i-1))/2
  result[which(j == i)] <- 0
  t <- j[which(j < i)]
  result[which(j < i)] <- (n*(t -1) + i - t - (t*(t-1))/2)
  return(result)
}



chooseStart <- function(unclustered_points, method = "minmedian"){
  switch (method,
```

```
          maxmedian = {
            ll <- dist(unclustered_points)
            j <- which.max(sapply(1:nrow(unclustered_points),
                                  function(i) median(ll[ind(i,
1:nrow(unclustered_points), nrow(unclustered_points))])))
            return(unclustered_points[j,])
          },

          minmedian = {
            ll <- dist(unclustered_points)
            j <- which.min(sapply(1:nrow(unclustered_points),
                                  function(i) median(ll[ind(i,
1:nrow(unclustered_points), nrow(unclustered_points))])))
            return(unclustered_points[j,])
          },

          random = {

return(unclustered_points[sample.int(nrow(unclustered_points),
                                                 size = 1),])
          }
  )
}

formSphere <- function(unclustered_points,
                       radius,
                       cent_meth = "minmedian",
                       i,
                       visualise = T,
                       k = NULL){

  if (cent_meth == "seq") center <- unclustered_points[k,]
  else {
    center <- chooseStart(unclustered_points, cent_meth)
  }

  if (isTRUE(visualise)) plotClustSearch(unclustered_points,
center, radius, i)

  repeat{
    ll <- dist(rbind(center, unclustered_points))
    indices <- which(ll[1:nrow(unclustered_points)]
                     < radius)
    sphereCandidate <-  unclustered_points[indices,]
    center <- reevaluateCenterOfCluster(sphereCandidate)

    if (isTRUE(visualise)) plotClustSearch(unclustered_points,
center, radius, i)

    if (isTRUE(all.equal(which(dist(rbind(center,
unclustered_points))[1:nrow(unclustered_points)]
```

```
                                < radius),
                           indices))) break
  }
  if (isTRUE(visualise)) points(sphereCandidate,
                                col = i,
                                pch = 19)

  return(sphereCandidate)
}


forel <- function(unclustered_points, radius, cent_meth =
"minmedian"){
  number_of_unclustered_points <- nrow(unclustered_points)
  clusters <- data.frame(v = NULL, e = NULL, clNum = NULL)
  i <- 0


  repeat{
    i <- i + 1

    dd <- formSphere(unclustered_points, radius, cent_meth, i)
    clusters <- rbind(clusters,
                      cbind(dd, clNum = i))

    unclustered_points <- setdiff(unclustered_points,
                                  dd)
    number_of_unclustered_points <- nrow(unclustered_points)

    if (number_of_unclustered_points == 1) {
      clusters <- rbind(clusters,
                        cbind(unclustered_points, clNum = i+1))
      unclustered_points <- setdiff(unclustered_points,
unclustered_points)
      number_of_unclustered_points <- nrow(unclustered_points)
    }

    if (number_of_unclustered_points == 0) break
  }
  return(clusters)
}

###ØÒÎ ÇÄÅÑÜ ÏÐÎÈÑÕÎÄÈÒ??!! ÏÀÌÀÃÈÒÈ!!
forel2 <- function(unclustered_points, radius){

  number_of_unclustered_points <- nrow(unclustered_points)
  all_points <- unclustered_points

  for (j in 1:2) {
    plot(unclustered_points,
         asp = 1,
```

```
        xlim = c(0, 1),
        ylim = c(0, 1),
        main = "FOREL 2")
  }

  clusters <- data.frame(v = NULL, e = NULL, clNum = NULL)

  i <- 0

  repeat{
    i <- i + 1
    cc <- list(3)
    for (k in 1:nrow(unclustered_points)) {
      cluster <- formSphere(unclustered_points, radius,
                            cent_meth = "seq", i,
                            k, visualise = F)
      cc[[k]] <- cluster
    }

    {
      ww <- unique(cc)

      ws <- map(ww, reevaluateCenterOfCluster)
      ws <- as.data.frame(ws)
      ws <- data.frame(t(ws[1, ]),t(ws[2,]), row.names =
1:ncol(ws))
      colnames(ws) <- c("v", "e")

      plot(unclustered_points,
           asp = 1,
           xlim = c(0, 1),
           ylim = c(0, 1),
           main = "FOREL 2")

      for (j in 1:nrow(ws)) {
        polygon(circle(ws[j, ], radius),
                border = j)
      }

    }

    poo <- map_dbl(ww, function(x) nrow(x))
    dd <- ww[[which.max(poo)]]


    clusters <- rbind(clusters,
                      cbind(dd, clNum = i))
    unclustered_points <- setdiff(unclustered_points,
                                  dd)
    number_of_unclustered_points <- nrow(unclustered_points)
```

```
    {
      plot(unclustered_points,
           asp = 1,
           xlim = c(0, 1),
           ylim = c(0, 1),
           main = "FOREL 2")
      polygon(circle(reevaluateCenterOfCluster(dd), radius),
              border = i)
      points(clusters[, 1:2], col = clusters$clNum, pch = 19)
    }


    if (number_of_unclustered_points == 0) break
  }
  return(clusters)
}


plotClustSearch <- function(unclustered_points, center, radius, i)
{
  plot(unclustered_points, asp = 1, xlim = c(0, 1), ylim = c(0,
1))
  points(x = center[[1]],
         y = center[[2]],
         pch = 4)
  polygon(circle(center, radius),
          border = i)
}


circle <- function(center, radius){
  phi <- seq(0, 2*pi, by = 0.01)
  return(cbind(x = center[[1]] + radius*cos(phi),
               y = center[[2]] + radius*sin(phi)))
}


forel2Research <- function(sample_Scaled, radius){
  clustered_sample <- forel2(sample_Scaled, radius)
  clust_centers <- reevaluateCenters(clustered_sample,
                                     max(clustered_sample$clNum),

clustered_sample[1:max(clustered_sample$clNum),1:2])
  quality1 <- clusterDensity(clustered_sample,
                             clust_centers,
                             rad = T)

  quality2 <- clusterDensity(clustered_sample,
                             clust_centers,
                             rad = F)
  plotClusters(clustered_sample,
```

```
              K = nrow(clust_centers),
              alg_option = "FOREL2",
              quality1,
              quality2)

  points(clust_centers,
         pch = 4)

  for(j in 1:nrow(clust_centers)) {
    polygon(circle(clust_centers[j, ], radius),
            border = j)
  }
  for (j in 1:6) {
    plotClusters(clustered_sample,
                 K = nrow(clust_centers),
                 alg_option = "FOREL2",
                 quality1,
                 quality2)
  }

  return(c(quality1, quality2, nrow(clust_centers)))
}


forelResearch <- function(sample_Scaled, radius, cent_meth =
"minmedian"){
  plot(sample_Scaled, asp = 1, xlim = c(0, 1), ylim = c(0, 1),
       main = paste("FOREL", cent_meth))
  clustered_sample <- forel(sample_Scaled, radius, cent_meth)
  clust_centers <- reevaluateCenters(clustered_sample,
                                     max(clustered_sample$clNum),

clustered_sample[1:max(clustered_sample$clNum),1:2])

  quality1 <- clusterDensity(clustered_sample,
                             clust_centers,
                             rad = T)

  quality2 <- clusterDensity(clustered_sample,
                             clust_centers,
                             rad = F)
  plotClusters(clustered_sample,
               nrow(clust_centers),
               paste("FOREL", cent_meth),
               quality1,
               quality2)

  points(clust_centers,
         pch = 4)

  for(j in 1:nrow(clust_centers)) {
```

```
    polygon(circle(clust_centers[j, ], radius),
            border = j)
  }
  for (j in 1:6) {
    plotClusters(clustered_sample,
                 nrow(clust_centers),
                 paste("FOREL", cent_meth),
                 quality1,
                 quality2)
  }

  return(c(quality1, quality2, nrow(clust_centers)))
}

#масштабирование выборки
##стандартизация (z-score)

## require(dplyr)

######################t(sample) - as.vector(moments["mean",], mode
= "numeric")

zscore <- data.frame(v = (sample$v - moments["mean","v"])/
moments["deviation","v"],
                     e = (sample$E - moments["mean","e"])/
moments["deviation","e"])
zscore <- as.tbl(zscore)

##min-max normalization
sample_Scaled <- data.frame(v = (sample$v - min(sample$v))/vRange,
                            e = (sample$E - min(sample$E))/eRange)
sample_Scaled <- as.tbl(sample_Scaled)

# K - количество кластеров
# mykmeans(points, K, center_method, algoritm_option)
# center_method - способ выбора центров:
### "random" - случайные точки выборки
### "equidistant" - выбираются из отсортированной по v выборки с
одинаковым шагом
#  algoritm_option - вариант алгоритма:
### "Lazy" - сначала формируются кластеры, потом производится
пересчет центров
### "Long" - пересчет центров после каждой добавленной точки

for (K in 4:9){
  newdir <- paste0("Lazy_eq",K)
  dir.create(newdir)
  cwd <- getwd()
  setwd(newdir)
  mykmeans(sample_Scaled, K, "equidistant", "Lazy")
  setwd(cwd)
```

```
  newdir <- paste0("Long_eq",K)
  dir.create(newdir)
  cwd <- getwd()
  setwd(newdir)
  mykmeans(sample_Scaled, K, "equidistant", "Long")
  setwd(cwd)

}

K <- 7
comparisonTable <- data.frame(Lazy_centdist = 1:10,
                              Lazy_innerdist = 1,
                              Long_centdist = 1,
                              Long_innerdist = 1)
for (i in 1:10) {
  newdir <- paste0("Rand",i)
  dir.create(newdir)
  cwd <- getwd()
  setwd(newdir)
  comparisonTable[i,] <- mykmeans_both(sample_Scaled, K, "random")
  setwd(cwd)
}
```

lab7.r

```
#масштабирование выборки
##стандартизация (z-score)

## require(dplyr)
## require(colorspace)

zscore <- data.frame(v = (sample$v - moments["mean","v"])/
moments["deviation","v"],
                     e = (sample$E - moments["mean","e"])/
moments["deviation","e"])
zscore <- as.tbl(zscore)

##min-max normalization
sample_Scaled <- data.frame(v = (sample$v - min(sample$v))/vRange,
                            e = (sample$E - min(sample$E))/eRange)
sample_Scaled <- as.tbl(sample_Scaled)

unclustered_points <- sample_Scaled

number_of_unclustered_points <- N

#### cent_meth: "minmedian", "maxmedian", "random"

comparTForel <- data.frame(radius = 1:10,
                           Rand_cd = 1,
```

```
                              Rand_innd = 1,
                              Rand_K = 1,
                              Minm_cd = 1,
                              Minm_innd = 1,
                              Minm_K = 1,
                              Maxm_cd = 1,
                              Maxm_innd = 1,
                              Maxm_K = 1,
                              Sec_cd = 1,
                              Sec_innd = 1,
                              Sec_K = 1)

cn <- 10
palette(sample(rainbow(20)))

#plot(1:(2*cn), col = 1:(2*cn))

{
  newdir <- "ExperRadius"
  dir.create(newdir)
  swd <- getwd()
  setwd(newdir)

  radius <- 0.1
  for (j in 1:10) {
    comparTForel[j, 1] <- radius

    {
      newdir <- paste0("random", radius)
      dir.create(newdir)
      cwd <- getwd()
      setwd(newdir)

      png(file="rand%03d.png")
      comparTForel[j, 2:4] <- forelResearch(sample_Scaled, radius,
cent_meth ="random")

      dev.off()
      setwd(cwd)
    }

    {
      newdir <- paste0("minmedian", radius)
      dir.create(newdir)
      cwd <- getwd()
      setwd(newdir)

      png(file="minmedian%03d.png")
      comparTForel[j, 5:7] <- forelResearch(sample_Scaled, radius,
cent_meth ="minmedian")
```

```
      dev.off()
      setwd(cwd)
    }

    {
      newdir <- paste0("maxmedian", radius)
      dir.create(newdir)
      cwd <- getwd()
      setwd(newdir)

      png(file="maxmedian%03d.png")
      comparTForel[j, 8:10] <- forelResearch(sample_Scaled,
radius, cent_meth ="maxmedian")

      dev.off()
      setwd(cwd)
    }

    {
      newdir <- paste0("forelSec", radius)
      dir.create(newdir)
      cwd <- getwd()
      setwd(newdir)

      png(file="forelSec%03d.png")
      comparTForel[j, 11:13] <- forel2Research(sample_Scaled,
radius)

      dev.off()
      setwd(cwd)
    }

    radius <- radius + 0.025
  }
  setwd(swd)
}


comparTForelRand <- data.frame(Rand_cd = 1:10,
                               Rand_innd = 1,
                               K = 1)

{
  newdir <- "ExperRand"
  dir.create(newdir)
  swd <- getwd()
  setwd(newdir)

  radius <- 0.15
  for (j in 1:10) {
    newdir <- paste0("random", j)
```

```
    dir.create(newdir)
    cwd <- getwd()
    setwd(newdir)

    png(file="rand%03d.png")
    comparTForelRand[j, ] <- forelResearch(sample_Scaled, radius,
cent_meth ="random")

    dev.off()
    setwd(cwd)
  }

  setwd(swd)
}

{
  plot(comparTForel$radius,
       y = comparTForel$Rand_cd,
       type = "l",
       col = 17,
       lwd = 2,
       main = "Зависимость СКР от R для FOREL алгоритмов",
       xlab = "Radius",
       ylab = "Средн. кластерный радиус")
  lines(comparTForel$radius,
        y = comparTForel$Minm_cd,
        col = 3,
        lwd = 2)
  lines(comparTForel$radius,
        y = comparTForel$Maxm_cd,
        col = 12,
        lwd = 2)
  lines(comparTForel$radius,
        y = comparTForel$Sec_cd,
        col = 8,
        lwd = 2)

  legend(x = "bottomright",
         inset = 0,
         legend = c("random", "minmedian", "maxmedian", "full"),
         col = c(17, 3, 12, 8),
         bty = "n",
         lwd = 3,
         y.intersp = 0.7)
}

{
  plot(comparTForel$radius,
       y = comparTForel$Rand_innd,
       type = "l",
       col = 17,
```

```
      lwd = 2,
      main = "Зависимость СВкР от R для FOREL алгоритмов",
      xlab = "Radius",
      ylab = "Средн. внутрикл. расстояния")
  lines(comparTForel$radius,
       y = comparTForel$Minm_innd,
       col = 3,
       lwd = 2)
  lines(comparTForel$radius,
       y = comparTForel$Maxm_innd,
       col = 12,
       lwd = 2)
  lines(comparTForel$radius,
       y = comparTForel$Sec_innd,
       col = 8,
       lwd = 2)

  legend(x = "bottomright",
         inset = 0,
         legend = c("random", "minmedian", "maxmedian", "full"),
         col = c(17, 3, 12, 8),
         bty = "n",
         lwd = 3,
         y.intersp = 0.7)
}

{
  plot(comparTForel$radius,
      y = comparTForel$Rand_K,
      type = "l",
      col = 17,
      lwd = 2,
      main = "Зависимость кол-ва кластеров от R для FOREL
алгоритмов",
      xlab = "Radius",
      ylab = "Количество кластеров",
      yaxt = "n")
  axis(2, at = 1:19,
      labels = 1:19, las = 1)
  lines(comparTForel$radius,
       y = comparTForel$Minm_K,
       col = 3,
       lwd = 2)
  lines(comparTForel$radius,
       y = comparTForel$Maxm_K,
       col = 12,
       lwd = 2)
  lines(comparTForel$radius,
       y = comparTForel$Sec_K,
       col = 8,
       lwd = 2)
```

```
  legend(x = "topright",
         inset = 0,
         legend = c("random", "minmedian", "maxmedian", "full"),
         col = c(17, 3, 12, 8),
         bty = "n",
         lwd = 3,
         y.intersp = 0.7)
}
```