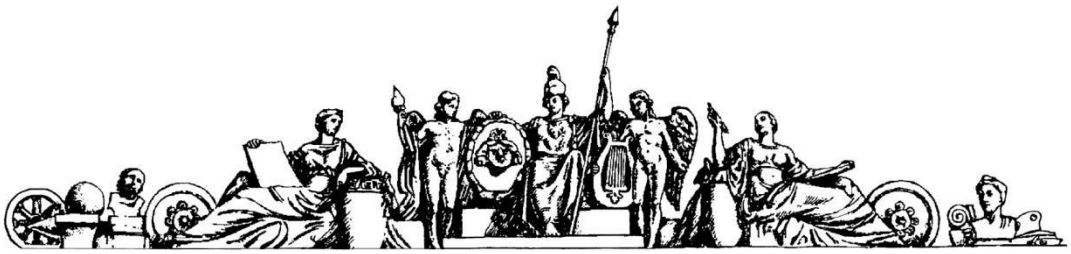

**Министерство образования и науки Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение**
**высшего профессионального образования**
**«Московский государственный технический университет имени Н. Э. Баумана»**
**(МГТУ им. Н. Э. Баумана)**

Факультет Информатики и систем Управления (ИУ)
Кафедра «Информационная безопасность» (ИУ8)

Отчет по лабораторной работе №5
по курсу
**«ИНТЕЛЛЕКТУАЛЬНЫЕ ТЕХНОЛОГИИ НА ОСНОВЕ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ»**
по теме:
Алгоритмы кластерного анализа данных

Выполнил:
студент группы ИУ8-61
Волков М. А.

Преподаватели:
Басараб М. А.
Коннова Н. С.

Москва 2017

**Цель работы**: Исследовать применение основных алгоритмов кластерного анализа, включая их модификации, на примере различных типов данных.

**Постановка задачи**.

Кластерный анализ – процедура, выполняющая сбор данных, содержащих информацию о выборке объектов, и затем упорядочивающая объекты в сравнительно однородные группы на основе какого-либо признака(-ов). Формально: Пусть – X множество объектов, Y– множество кластеров. Задана функция расстояния между объектами $\rho(x, x`)$. Имеется конечная обучающая выборка объектов $X^m = \{x_1, \dots, x_m\} \subset X$ . Требуется разбить выборку на непересекающиеся подмножества, называемые кластерами, так, чтобы каждый кластер состоял из объектов, близких по метрике $\rho$, а объекты разных кластеров существенно отличались. При этом каждому объекту $x_i \in X^m$ приписывается номер кластера . Алгоритм кластеризации – это функция $\varphi: X \to Y$, которая любому объекту $x \in X$ставит в соответствие номер кластера $y \in Y$. Множество Y в некоторых случаях известно заранее, однако чаще ставится задача определить оптимальное число кластеров, с точки зрения того или иного критерия качества кластеризации.

Нейронные сети Кохонена – класс нейронных сетей, основным элементом которых является слой Кохонена, состоящий из k адаптивных линейных сумматоров. Они имеют одинаковое число входов m и получают на свои входы вектор входных сигналов $(x_1, \dots, x_m)$. На выходе j-го линейного элемента получаем сигнал

$$y_j = w_{j0} + \sum_{i=1}^{m} w_{ji} x_i$$

Где j – номер нейрона, $w_{j0}$ − пороговый коэффициент, I – номер входа, $w_{ji}$ − весовой коэффициент i-го входа j-го нейрона.

Выходные сигналы слоя Кохонена обрабатываются по правилу «победитель получает всё»: наибольший сигнал превращается в единичный, остальные обращаются в ноль. Таким образом, применительно к задаче кластеризации каждому j-му нейрону ставятся в соответствие точки-центры кластеров, для входного вектора $x = (x_1, \dots, x_m)$ вычисляются расстояния $\rho_j(x)$, и тот нейрон, до которого это расстояние минимально, выдает единицу, остальные – ноль.

# Практическая часть.

| № пп | Алгоритм | Исходные кластеризуемые данны | $\rho$ |
|---|---|---|---|
| 2 | НС Кохонена | Выборка колледжей г. Москвы, http://data.mos.ru/datasets/546 координаты местоположения | Принадлежность округу Москвы (Евклидово расстояние до координат центра округа) |

Для центров кластеров я выбрал префектуры округов, но они далеко не точно отражают центры, но это и вызывает интерес.

| Название округа | Координата X | Координата Y |
|---|---|---|
| Центральный административный округ | 37,662131 | 55.737036 |
| Северный административный округ | 37.565703 | 55.814916 |
| Северо-Восточный административный округ | 37.633725 | 55.776371 |
| Восточный административный округ | 37.710262 | 55.796901 |
| Юго-Восточный административный округ | 37.715454 | 55.754081 |
| Южный административный округ | 37.665580 | 55.710246 |
| Юго-Западный административный округ | 37.578244 | 55.662382 |

| | | |
|---|---|---|
| Западный административный округ | 37.443237 | 55.727203 |
| Северо-Западный административный округ | 37.467837 | 55.784440 |
| Зеленоградский административный округ | 37.215344 | 55.990531 |

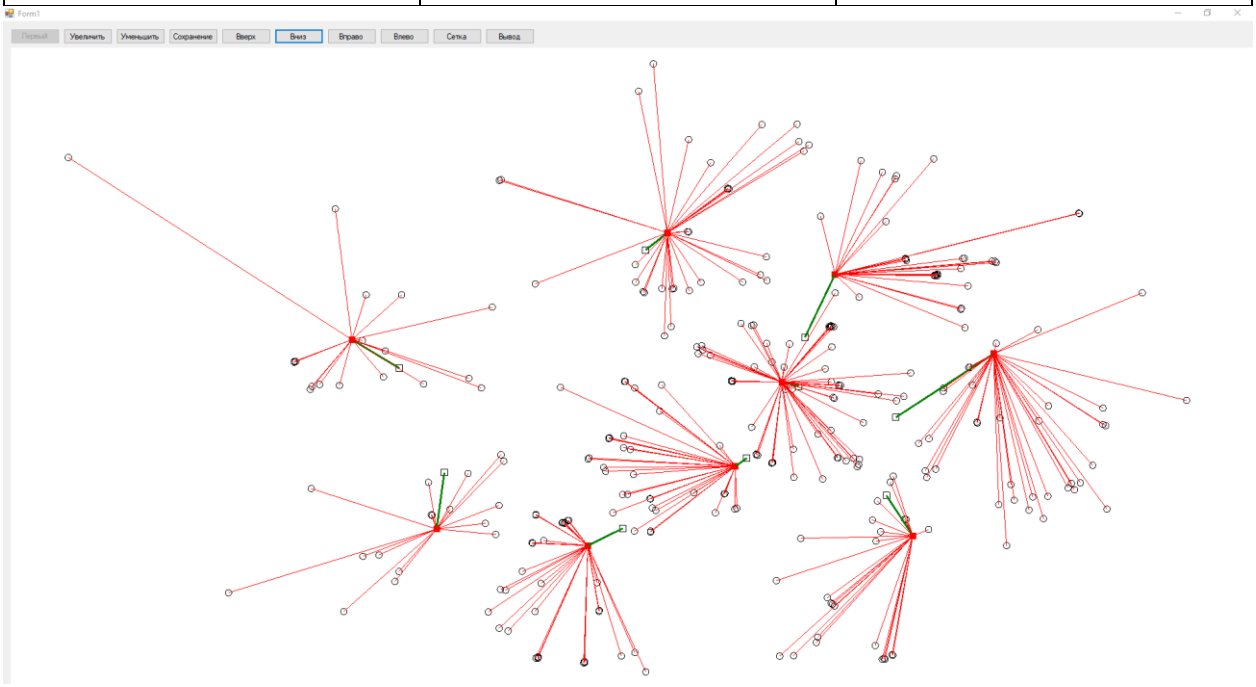


Рис 1 Работа программы

Кружки — это колледжи, они соединены и своими кластерами, кластеры в свою очередь тоже изменили свое положение – пустой квадратик — это то место, от которого он ушел, а закрашенный красный, его конечное положение. К какому кластеру ближе колледж, тому он и принадлежит, после все распределений перемещаем кластер по принципу центра масс, и так делаем до тех пор, пока кластеры не перестанут двигаться.

Ошибка в данном случае очень большая, только 47,7193% попадания, но это объясняется тем, что центры округов далеки от префектуры, к примеру, можно заметить, что:

| Название округа | Процент попадания действительной принадлежности и решенных с помощью кластеризации |
|---|---|
| Центральный административный округ | 49,15254 |

| | |
|---|---|
| Северный административный округ | 90,90909 |
| Северо-Восточный административный округ | 20,58824 |
| Восточный административный округ | 28 |
| Юго-Восточный административный округ | 22,22222 |
| Южный административный округ | 23,33333 |
| Юго-Западный административный округ | 46,875 |
| Западный административный округ | 100 |
| Северо-Западный административный округ | 82,35294 |
| Зеленоградский административный округ | 100 |

У половины округов процент крайне мал, действительно префектуры их округов стоят скорее ближе к границе раздела округов.

В результат получается такая принадлежность округам:

Центральный административный округ

|1) X = 37.6571236 Y = 55.7139015 |2) X = 37.6258774 Y = 55.7344208 |3) X = 37.6540871 Y = 55.7266159 |4) X = 37.6090584 Y = 55.7148896 |5) X = 37.6252098 Y = 55.7340088 |6) X = 37.6363793 Y = 55.7594834 |7) X = 37.6949806 Y = 55.7799416 |8) X = 37.6594887 Y = 55.7869187 |9) X = 37.6377945 Y = 55.7264366 |10) X = 37.6388207 Y = 55.7742729 |11) X = 37.6305466 Y = 55.7026634 |12) X = 37.6549797 Y = 55.7388649 |13) X = 37.7058029 Y = 55.7413445 |14) X = 37.6943131 Y = 55.7812004 |15) X = 37.7198181 Y = 55.7426224 |16) X = 37.6954689 Y = 55.7770157 |17) X = 37.6473885 Y = 55.7791786 |18) X = 37.6475907 Y = 55.7789574 |19) X = 37.6479683 Y = 55.7787934 |20) X = 37.6800919 Y = 55.7346154 |21) X = 37.7235604 Y = 55.7297592 |22) X = 37.6766587 Y = 55.7274590 |23) X = 37.6852036 Y = 55.7366638 |24) X = 37.6071129 Y = 55.7191773 |25) X = 37.6138191 Y = 55.7200928 |26) X = 37.6821556 Y = 55.6858864 |27) X = 37.7157707 Y = 55.7450562 |28) X = 37.6592484 Y = 55.7715760 |29) X =

37.6881409 Y = 55.7779885 |30) X = 37.6922264 Y = 55.7765656 |31) X = 37.6727066 Y = 55.7632332 |32) X = 37.6585922 Y = 55.7386055 |33) X = 37.6976204 Y = 55.7570305 |34) X = 37.6833573 Y = 55.7761460 |35) X = 37.6838188 Y = 55.7762909 |36) X = 37.6388893 Y = 55.7744751 |37) X = 37.6396027 Y = 55.7748642 |38) X = 37.6798668 Y = 55.7038765 |39) X = 37.6802712 Y = 55.7041893 |40) X = 37.6813507 Y = 55.7045174 |41) X = 37.6795082 Y = 55.7044754 |42) X = 37.6788063 Y = 55.7044297 |43) X = 37.6790772 Y = 55.7048798 |44) X = 37.6475144 Y = 55.7237969 |45) X = 37.6752701 Y = 55.7878647 |46) X = 37.6803017 Y = 55.7154846 |47) X = 37.6062584 Y = 55.7154465 |48) X = 37.6062584 Y = 55.7154465 |49) X = 37.6363144 Y = 55.7037888 |50) X = 37.6374130 Y = 55.7035713 |51) X = 37.6095162 Y = 55.7172585 |52) X = 37.6446648 Y = 55.7136574 |53) X = 37.6798897 Y = 55.7612686 |54) X = 37.6253014 Y = 55.7343064 |55) X = 37.6258774 Y = 55.7341042 |56) X = 37.6810684 Y = 55.7432328 |57) X = 37.6816483 Y = 55.7436219 |58) X = 37.6840592 Y = 55.7363892 |59) X = 37.6842003 Y = 55.7367554

Северный административный округ

|1) X = 37.5516625 Y = 55.8446007 |2) X = 37.5163193 Y = 55.8230667 |3) X = 37.5180207 Y = 55.8076439 |4) X = 37.5528260 Y = 55.8598518 |5) X = 37.5028763 Y = 55.8711166 |6) X = 37.5106240 Y = 55.8384590 |7) X = 37.5337372 Y = 55.8117981 |8) X = 37.5013962 Y = 55.8487778 |9) X = 37.5222817 Y = 55.8215523 |10) X = 37.5526619 Y = 55.8603249 |11) X = 37.5354233 Y = 55.8105202 |12) X = 37.5361176 Y = 55.8105812 |13) X = 37.5783577 Y = 55.8935318 |14) X = 37.5180245 Y = 55.8072891 |15) X = 37.4979134 Y = 55.8696365 |16) X = 37.5191498 Y = 55.8860550 |17) X = 37.5180168 Y = 55.8860245 |18) X = 37.5191498 Y = 55.8855743 |19) X = 37.5193024 Y = 55.8852730 |20) X = 37.5723229 Y = 55.8829346 |21) X = 37.5649414 Y = 55.8848305 |22) X = 37.5337372 Y = 55.8117981 |23) X =

37.5328293 Y = 55.8116074 |24) X = 37.5339852 Y = 55.8122330 |25) X =

37.5324517 Y = 55.8119545 |26) X = 37.5159722 Y = 55.8227196 |27) X =

37.4986573 Y = 55.8484344 |28) X = 37.5175820 Y = 55.8604736 |29) X =

37.5209389 Y = 55.8451080 |30) X = 37.4919777 Y = 55.8607407 |31) X =

37.5443001 Y = 55.8885994 |32) X = 37.5446549 Y = 55.8878174 |33) X =

37.5448113 Y = 55.8884087

Северо-Восточный административный округ

|1) X = 37.6282387 Y = 55.8038941 |2) X = 37.5662384 Y = 55.7642517 |3) X =

37.5315323 Y = 55.7376709 |4) X = 37.5859375 Y = 55.7506027 |5) X =

37.6164589 Y = 55.8064041 |6) X = 37.6188774 Y = 55.7696457 |7) X =

37.5818939 Y = 55.8036614 |8) X = 37.5809250 Y = 55.8164063 |9) X =

37.6046486 Y = 55.7953987 |10) X = 37.5838318 Y = 55.8051377 |11) X =

37.5716095 Y = 55.7854157 |12) X = 37.5584679 Y = 55.7653580 |13) X =

37.5720673 Y = 55.8164292 |14) X = 37.5809326 Y = 55.8169747 |15) X =

37.6271019 Y = 55.8043633 |16) X = 37.5698891 Y = 55.7716599 |17) X =

37.5602799 Y = 55.8063889 |18) X = 37.5656052 Y = 55.7964249 |19) X =

37.5666046 Y = 55.7343903 |20) X = 37.5671273 Y = 55.7344551 |21) X =

37.5721970 Y = 55.7380677 |22) X = 37.5869904 Y = 55.7363129 |23) X =

37.5805931 Y = 55.7986298 |24) X = 37.5808411 Y = 55.7988205 |25) X =

37.5580712 Y = 55.7654915 |26) X = 37.5580521 Y = 55.7657624 |27) X =

37.5683899 Y = 55.7963066 |28) X = 37.5466004 Y = 55.7768936 |29) X =

37.5473290 Y = 55.7763443 |30) X = 37.5562897 Y = 55.7836571 |31) X =

37.6217194 Y = 55.7958489 |32) X = 37.6212921 Y = 55.7959481 |33) X =

37.5552597 Y = 55.7815514 |34) X = 37.5661736 Y = 55.7707024

Восточный административный округ

|1) X = 37.6593933 Y = 55.8846703 |2) X = 37.7026024 Y = 55.8103524 |3) X =

37.6726189 Y = 55.8744659 |4) X = 37.7123451 Y = 55.7896042 |5) X =

37.7054138 Y = 55.7999001 |6) X = 37.7045365 Y = 55.8219643 |7) X =

37.7202072 Y = 55.8098870 |8) X = 37.7079659 Y = 55.8865852 |9) X =

37.7133065 Y = 55.8840942 |10) X = 37.6779442 Y = 55.8528976 |11) X =

37.7041397 Y = 55.8791504 |12) X = 37.6817703 Y = 55.8574257 |13) X =

37.6805611 Y = 55.8564529 |14) X = 37.6633225 Y = 55.8204041 |15) X =

37.6516876 Y = 55.8848915 |16) X = 37.6499367 Y = 55.8435135 |17) X =

37.7210694 Y = 55.8100853 |18) X = 37.6717835 Y = 55.8774261 |19) X =

37.7138214 Y = 55.8844261 |20) X = 37.7139969 Y = 55.7862625 |21) X =

37.7070503 Y = 55.8167038 |22) X = 37.7333985 Y = 55.8155480 |23) X =

37.7089463 Y = 55.8862152 |24) X = 37.6799469 Y = 55.8558083 |25) X =

37.7094002 Y = 55.8865891

Юго-Западный административный округ

|1) X = 37.8505898 Y = 55.6858521 |2) X = 37.7963372 Y = 55.8095665 |3) X =

37.7904663 Y = 55.7977333 |4) X = 37.7704811 Y = 55.7135735 |5) X =

37.8233528 Y = 55.7803841 |6) X = 37.7803002 Y = 55.7994538 |7) X =

37.7761612 Y = 55.8243752 |8) X = 37.8133431 Y = 55.7937584 |9) X =

37.7415924 Y = 55.7379952 |10) X = 37.8748474 Y = 55.7448235 |11) X =

37.7372895 Y = 55.7824593 |12) X = 37.7279053 Y = 55.7683449 |13) X =

37.8311615 Y = 55.8043747 |14) X = 37.7323952 Y = 55.7869988 |15) X =

37.8289452 Y = 55.7580338 |16) X = 37.7932854 Y = 55.7061043 |17) X =

37.8098717 Y = 55.7926979 |18) X = 37.8117066 Y = 55.7902489 |19) X =

37.8289528 Y = 55.7490464 |20) X = 37.7601166 Y = 55.7571755 |21) X =

37.7598915 Y = 55.7566643 |22) X = 37.7315521 Y = 55.7833786 |23) X =

37.7335739 Y = 55.7657280 |24) X = 37.8304329 Y = 55.7586441 |25) X =

37.7988129 Y = 55.7973709 |26) X = 37.7938385 Y = 55.7559777 |27) X =

37.7785263 Y = 55.7909241 |28) X = 37.7884445 Y = 55.8048706 |29) X =

37.7718964 Y = 55.8018875 |30) X = 37.7413712 Y = 55.7398491 |31) X =

37.7574997 Y = 55.7363205 |32) X = 37.7990990 Y = 55.7478142 |33) X =

37.7554207 Y = 55.7267380 |34) X = 37.7561302 Y = 55.7267304 |35) X = 37.8165932 Y = 55.7899666 |36) X = 37.7725258 Y = 55.7544213

Южный административный округ

|1) X = 37.6952248 Y = 55.6882248 |2) X = 37.7080193 Y = 55.6198502 |3) X = 37.7509766 Y = 55.6949082 |4) X = 37.6962967 Y = 55.6134377 |5) X = 37.7157555 Y = 55.6218262 |6) X = 37.7383538 Y = 55.6763954 |7) X = 37.7149658 Y = 55.6234284 |8) X = 37.7549248 Y = 55.6820870 |9) X = 37.7533074 Y = 55.7049828 |10) X = 37.7100106 Y = 55.6467934 |11) X = 37.7362023 Y = 55.6124573 |12) X = 37.7382202 Y = 55.6755486 |13) X = 37.7376709 Y = 55.6760292 |14) X = 37.7369728 Y = 55.6761742 |15) X = 37.7363892 Y = 55.6763382 |16) X = 37.7370491 Y = 55.6765557 |17) X = 37.7206154 Y = 55.6669693 |18) X = 37.7211685 Y = 55.6681595 |19) X = 37.7520752 Y = 55.6670723 |20) X = 37.7529831 Y = 55.6669274 |21) X = 37.7208672 Y = 55.6674004 |22) X = 37.7704544 Y = 55.6693650 |23) X = 37.7513199 Y = 55.6943245 |24) X = 37.7698937 Y = 55.6686592 |25) X = 37.7681275 Y = 55.6684990 |26) X = 37.8157311 Y = 55.6422310 |27) X = 37.8160057 Y = 55.6424256 |28) X = 37.6741486 Y = 55.6439438 |29) X = 37.7510529 Y = 55.6726341 |30) X = 37.7510529 Y = 55.6726341

Юго-Восточный административный округ

|1) X = 37.5726166 Y = 55.6704407 |2) X = 37.6612778 Y = 55.5934983 |3) X = 37.6419792 Y = 55.5936928 |4) X = 37.5885697 Y = 55.7094231 |5) X = 37.6649704 Y = 55.6082764 |6) X = 37.6231995 Y = 55.6291237 |7) X = 37.6677971 Y = 55.6048622 |8) X = 37.6622467 Y = 55.6030274 |9) X = 37.4980202 Y = 55.6242752 |10) X = 37.5774575 Y = 55.6857453 |11) X = 37.6314583 Y = 55.6799584 |12) X = 37.6444359 Y = 55.6661835 |13) X = 37.6018410 Y = 55.6019440 |14) X = 37.4990082 Y = 55.6239548 |15) X = 37.5727539 Y = 55.6799660 |16) X = 37.6011467 Y = 55.6523133 |17) X = 37.6018601 Y = 55.6524811 |18) X = 37.5743447 Y = 55.5754738 |19) X =

37.5825119 Y = 55.5603829 |20) X = 37.5768662 Y = 55.6852150 |21) X = 37.5177574 Y = 55.6809426 |22) X = 37.6021271 Y = 55.6847267 |23) X = 37.5871468 Y = 55.6835327 |24) X = 37.5928612 Y = 55.6837654 |25) X = 37.5937386 Y = 55.6833992 |26) X = 37.6412277 Y = 55.6760445 |27) X = 37.6085854 Y = 55.6801682 |28) X = 37.6139260 Y = 55.6145668 |29) X = 37.6239434 Y = 55.6288529 |30) X = 37.6231957 Y = 55.6286011 |31) X = 37.6448097 Y = 55.6791725 |32) X = 37.5922432 Y = 55.7043686

Западный административный округ

|1) X = 37.2617531 Y = 55.6117211 |2) X = 37.3855972 Y = 55.7234650 |3) X = 37.3953056 Y = 55.7371598 |4) X = 37.4564972 Y = 55.7358551 |5) X = 37.4080811 Y = 55.6399079 |6) X = 37.4444542 Y = 55.6869698 |7) X = 37.3994980 Y = 55.7360802 |8) X = 37.4249344 Y = 55.6870041 |9) X = 37.4941521 Y = 55.6937599 |10) X = 37.3943787 Y = 55.7387657 |11) X = 37.4813347 Y = 55.7327233 |12) X = 37.4345246 Y = 55.7319298 |13) X = 37.4103279 Y = 55.7365875 |14) X = 37.4881668 Y = 55.7379875 |15) X = 37.4228935 Y = 55.7120056 |16) X = 37.3863564 Y = 55.7234459 |17) X = 37.3855553 Y = 55.7238655 |18) X = 37.4354973 Y = 55.7176705

Северо-Западный административный округ

|1) X = 37.4320145 Y = 55.8294983 |2) X = 37.3948479 Y = 55.7931404 |3) X = 37.3495369 Y = 55.8501701 |4) X = 37.4408036 Y = 55.8439903 |5) X = 37.4430123 Y = 55.8385926 |6) X = 37.4126816 Y = 55.8605003 |7) X = 37.4808502 Y = 55.7848168 |8) X = 37.4978295 Y = 55.8023148 |9) X = 37.4612084 Y = 55.8078347 |10) X = 37.4607811 Y = 55.8075218 |11) X = 37.4591370 Y = 55.7897110 |12) X = 37.4709358 Y = 55.8045273 |13) X = 37.4903527 Y = 55.8121758 |14) X = 37.4960137 Y = 55.8182488 |15) X = 37.4232407 Y = 55.8302117 |16) X = 37.5004654 Y = 55.7780838 |17) X = 37.4989090 Y = 55.7745781

Зеленоградский административный округ

|1) X = 37.1967964 Y = 55.9742737

# Вывод

Принцип кластеризации удобен для задач классификации, но необходимо очень точно задавать кластеры, иначе будет большая ошибка.

# Листнинг программы.

```
public partial class Form1 : Form
    {
       public struct coord
        {
        public float X;
        public float Y;
            public coord(float X, float Y)
            {
                this.X = X;
                this.Y = Y;
            }
        }
        public Form1()
        {
            InitializeComponent();
        }
        static float Center_of_mass(List<int> temp, float[] mas)
        {
            float coord = 0;
            for (int i = 0; i < temp.Count; i++)
            {
                coord += mas[temp[i]];
            }
            return coord / temp.Count;
        }
        static float Adaptive_adder(float[,] w, float[]x, int j,int m)
        {
            float y = 0;
            for (int i = 0; i < m; i++)
            {
                y += w[j,i] * x[i];
            }
            return y;
        }

        static List<List<int>> countsGlobal = new List<List<int>>();
        static List<coord> main(List<coord> klaster, float[] X, float[] Y)
        {
            countsGlobal.Clear();
            List<coord> Cor = new List<coord>();
            float[,] W = new float[klaster.Count, X.Length];
            List<double> dist = new List<double>();
            for(int j = 0; j < klaster.Count; j++)
            {
                for (int i = 0; i < X.Length; i++)
                {
                    W[j, i] = distance(klaster[j].X, klaster[j].Y, X[i], Y[i]);
                }
            }
            for(int i = 0; i < klaster.Count; i++) { countsGlobal.Add(new List<int>()); }
            for (int i = 0; i < X.Length; i++)
            {
                float Attitudes = float.MaxValue;
                int count = 0;
                for (int j = 0; j < klaster.Count; j++)
                {
```

```
                    if (W[j, i] < Attitudes)
                    {
                        Attitudes = W[j, i];
                        count = j;
                    }

                }
                countsGlobal[count].Add(i);
            }
            List<coord> klasterNew = new List<coord>();
            for (int i = 0; i < klaster.Count; i++)
            {
                klasterNew.Add(new coord(Center_of_mass(countsGlobal[i], X),
Center_of_mass(countsGlobal[i], Y)));
            }
            if (klaster.SequenceEqual(klasterNew))
            {
                return klaster;
            }
            else { main(klasterNew,X,Y); }
            return klasterNew;
        }
static float distance(double x_central, double y_central,double x_point, double y_point)
        {
            return (float)Math.Sqrt(Math.Pow(Math.Abs(x_central-x_point), 2) +
Math.Pow(Math.Abs(y_central-y_point),2)); ;
        }
        int Enlargement = 1;
        int width_height = 10;
        int bias_horizon = 0;
        int bias_vertical = 0;
        bool triger = false;
        List<coord> klasterNew = new List<coord>();
        List<coord> klaster = new List<coord>();
        float[] XGlobal;
        float[] YGlobal;
        float Chance_of_hitting = 0;
        float[] error;
        private void button1_Click(object sender, EventArgs e)
        {
            List<string> okrug = new List<string>();
            string path = @"C:\Users\miked\Desktop\Колледжи.csv";
            string[] readline = File.ReadAllLines(path);
            triger = false;
            button4.Enabled = true;
            klasterNew.Clear();
            klaster.Clear();
            Graphics g = pictureBox1.CreateGraphics();
            g.FillRectangle(Brushes.White, new Rectangle(0, 0, pictureBox1.Width,
pictureBox1.Height));
            List<float> X = new List<float>();
            List<float> Y = new List<float>();
            for (int i = 1; i < readline.Count(); i++)
            {
                for (int j = 0; j < 3; j++)
                {
                    readline[i] = readline[i].Substring(readline[i].IndexOf(';')+1);
                }
                    readline[i] = readline[i].Substring(readline[i].IndexOf('"'));
                    readline[i] = readline[i].Substring(1);
                    okrug.Add(readline[i].Remove(readline[i].IndexOf('"')));
                readline[i] = readline[i].Substring(readline[i].IndexOf("37,"));
                X.Add((float.Parse(readline[i].Remove(readline[i].IndexOf('"'))) -
37)*100);
                readline[i] = readline[i].Substring(readline[i].IndexOf(';') + 2);
                Y.Add((float.Parse(readline[i].Remove(readline[i].IndexOf('"'))) - 55) *
100);
            }
            XGlobal = new float[readline.Count() - 1];
            YGlobal = new float[readline.Count() - 1];
            X.CopyTo(XGlobal);
```

```
            Y.CopyTo(YGlobal);
            for(int i = 0; i < XGlobal.Length; i++)
            {
                g.DrawEllipse(Pens.Black, Enlargement * XGlobal[i]+bias_horizon,
Enlargement * YGlobal[i]+bias_vertical, width_height, width_height);
            }
            klaster.Add(new coord((float)66.2131, (float)73.7036));//центральный
55.737036, 37.662131
            klaster.Add(new coord((float)56.5703, (float)81.4916));//Северный 55.814916,
37.565703
            klaster.Add(new coord((float)63.3725, (float)77.6371));//северо-восточный
55.776371, 37.633725
            klaster.Add(new coord((float)71.0262, (float)79.6901));//восточный 55.796901,
37.710262
            klaster.Add(new coord((float)71.5454, (float)75.4081));//юго-восточный
55.754081, 37.715454
            klaster.Add(new coord((float)66.5580, (float)71.0246));//южный 55.710246,
37.665580
            klaster.Add(new coord((float)57.8244, (float)66.2382));//юго-западный
55.662382, 37.578244
            klaster.Add(new coord((float)44.3237, (float)72.7203));//западный 55.727203,
37.443237
            klaster.Add(new coord((float)46.7837, (float)78.4440));//северо-западный
55.784440, 37.467837
            klaster.Add(new coord((float)21.5344, (float)99.0531));//зеленоградский
55.990531, 37.215344
            for (int i = 0; i < klaster.Count; i++)
            {
                    g.DrawRectangle(Pens.Black, Enlargement * klaster[i].X+bias_horizon,
Enlargement * klaster[i].Y+bias_vertical, width_height, width_height);
            }
            klasterNew = main(klaster,XGlobal,YGlobal);
            for (int i = 0; i < klaster.Count; i++)
            {
                if (klasterNew[i].Equals(new coord(float.NaN,float.NaN)))
                {
                    g.FillRectangle(new SolidBrush(Color.Red), Enlargement *
klaster[i].X+bias_horizon, Enlargement * klaster[i].Y+bias_vertical, width_height,
width_height);
                }
                else
                {
                    g.FillRectangle(new SolidBrush(Color.Red), Enlargement *
klasterNew[i].X+bias_horizon, Enlargement * klasterNew[i].Y+bias_vertical, width_height,
width_height);
                    g.DrawLine(new Pen(Brushes.Green, 3), new PointF(Enlargement *
klaster[i].X + 5 + bias_horizon, Enlargement * klaster[i].Y + 5 + bias_vertical), new
PointF(Enlargement * klasterNew[i].X + 5 + bias_horizon, Enlargement * klasterNew[i].Y +
5 + bias_vertical));

                }
            }
            for (int i = 0; i < countsGlobal.Count; i++)
            {
                for(int j = 0; j < countsGlobal[i].Count; j++)
                {
                    if (klasterNew[i].Equals(new coord(float.NaN, float.NaN)))
                    {
                        g.DrawLine(new Pen(Brushes.Red, 1), new PointF(Enlargement *
klaster[i].X+5+bias_horizon, Enlargement * klaster[i].Y+5+bias_vertical), new
PointF(Enlargement * XGlobal[countsGlobal[i][j]]+5+bias_horizon, Enlargement *
YGlobal[countsGlobal[i][j]]+5+bias_vertical));
                    }
                    else
                    {
                       g.DrawLine(new Pen(Brushes.Red, 1), new PointF(Enlargement *
klasterNew[i].X+5 + bias_horizon, Enlargement * klasterNew[i].Y+5 + bias_vertical), new
PointF(Enlargement * XGlobal[countsGlobal[i][j]]+5 + bias_horizon, Enlargement *
YGlobal[countsGlobal[i][j]]+5 + bias_vertical));
                    }
```

```
                }
            }
            string[] okrugDef = { "Центральный административный округ",  //центральный
55.737036, 37.662131
                "Северный административный округ",                      //Северный
55.814916, 37.565703
                "Северо-Восточный административный округ",              //северо-
восточный 55.776371, 37.633725
                "Восточный административный округ",                     //восточный
55.796901, 37.710262
                "Юго-Восточный административный округ",                  //юго-восточный
55.754081, 37.715454
                "Южный административный округ",                         //южный
55.710246, 37.665580
                "Юго-Западный административный округ",                  //юго-западный
55.662382, 37.578244
                "Западный административный округ",                      //западный
55.727203, 37.443237
                "Северо-Западный административный округ",               //северо-
западный 55.784440, 37.467837
                "Зеленоградский административный округ" };              //зеленоградский
55.990531, 37.215344
            Chance_of_hitting = 0;
            error = new float[okrugDef.Count()];
            float ch_error = 0 ;
            for (int j = 0; j < okrugDef.Count(); j++)
            {
                for (int i = 0; i < countsGlobal[j].Count(); i++)
                {
                    if (okrug[countsGlobal[j][i]] == okrugDef[j])
                    {
                        Chance_of_hitting++;
                        ch_error++;
                    }
                }
                error[j] = (ch_error / countsGlobal[j].Count)*100;
                ch_error = 0;
            }
            button1.Enabled = false;
            button10.Enabled = true;
        }
private void pictureBox1_Click(object sender, EventArgs e)
        {

        }

        private void button2_Click(object sender, EventArgs e)
        {
            //if (triger == true)
            {
                Enlargement++;
                button1_Click(sender, e);
            }
         //   else
            {
        //        button9_Click(sender, e);
            }

        }

        private void button3_Click(object sender, EventArgs e)
        {
        //    if (triger == true)
            {
                if (Enlargement > 1)
                {
                    Enlargement--;
                }
                button1_Click(sender, e);
            }
          //  else
```

```
            {
        //        button9_Click(sender, e);
            }

        }
        private void saveFileDialog1_FileOk(object sender, CancelEventArgs e)
        {

        }

        private void button4_Click(object sender, EventArgs e)
        {
            string[] okrug = {"Центральный", "Северный","Северо-
восточный","Восточный","Юго-западный","Южный","Юго-западный","Западный","Северо-западный"
};
            SaveFileDialog savefile = new SaveFileDialog();
            savefile.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";
            if (savefile.ShowDialog() == DialogResult.OK)
            {
                using (Stream s = File.Open(savefile.FileName, FileMode.CreateNew))
                using (StreamWriter sw = new StreamWriter(s))
                {
                    for (int i = 0; i < countsGlobal.Count; i++)
                    {
                        sw.WriteLine("{0} округ", okrug[i]);
                        for (int j = 0; j < countsGlobal[i].Count; j++)
                        {
                            if (klasterNew[i].Equals(new coord(float.NaN, float.NaN)))
                            {

                            }
                            else
                            {
                                sw.WriteLine("X = 37.{0} Y = 55.{1} ",
XGlobal[countsGlobal[i][j]], YGlobal[countsGlobal[i][j]]);
                            }

                        }
                    }

                }
            }
        }
private void button5_Click(object sender, EventArgs e)
        {
          //  if (triger == true)
            {
                bias_vertical -= 50;
                button1_Click(sender, e);
            }
         //   else
            {
         //       button9_Click(sender, e);
            }

        }

        private void button6_Click(object sender, EventArgs e)
        {
          //  if (triger == true)
            {
                bias_horizon -= 50;
                button1_Click(sender, e);
            }
        //    else
            {
            //button9_Click(sender, e);
            }
        }

        private void button7_Click(object sender, EventArgs e)
```

```
        {
        //    if (triger == true)
            {
                bias_vertical += 50;
                button1_Click(sender, e);
            }
        //    else
            {
        //        button9_Click(sender, e);
            }

        }

        private void button8_Click(object sender, EventArgs e)
        {
         //   if (triger == true)
            {
                bias_horizon += 50;
                button1_Click(sender, e);
            }
         //   else
            {
        //          button9_Click(sender, e);
            }

        }
private void button9_Click(object sender, EventArgs e)
        {
            if (triger == false)
            {
                Graphics g = pictureBox1.CreateGraphics();
                for (int i = 0; i < pictureBox1.Width; i++)
                {
                    g.DrawLine(new Pen(Brushes.Black, 1), new PointF(Enlargement * i + 5
, Enlargement * 0), new PointF(Enlargement * i + 5, pictureBox1.Height));
                }
                for (int i = 0; i < pictureBox1.Height; i++)
                {
                    g.DrawLine(new Pen(Brushes.Black, 1), new PointF(0, Enlargement * i +
5), new PointF(pictureBox1.Width, Enlargement * i + 5));

                }
                triger = true;
            }
            else
            {
                triger = false;
                button1_Click(sender, e);
            }

        }
private void button10_Click(object sender, EventArgs e)
        {
            string[] okrug = { "Центральный административный округ", "Северный
административный округ", "Северо-Восточный административный округ", "Восточный
административный округ", "Юго-Западный административный округ", "Южный административный
округ", "Юго-Восточный административный округ", "Западный административный округ",
"Северо-Западный административный округ", "Зеленоградский административный округ" };

            float result = (Chance_of_hitting / XGlobal.Count())*100;
            string output= result+"% общее попадание \n";
            for(int i = 0; i < error.Count(); i++)
            {
                output = output + error[i] + "% "+ okrug[i] +" \n";
            }
                {
                    for (int i = 0; i < okrug.Count(); i++)
                    {
                        output=output +okrug[i]+"\n";
                        for (int j = 0; j < countsGlobal[i].Count; j++)
                        {
```

```
                            if (klasterNew[i].Equals(new coord(float.NaN, float.NaN)))
                            {

                            }
                            else
                            {
                                output = output +"|"+ (j+1) + ") X = 37." +
100000*XGlobal[countsGlobal[i][j]] + " Y = 55." + 100000*YGlobal[countsGlobal[i][j]]+" ";
                            }
                        }
                    output = output + "\n";
                    }

                }
            MessageBox.Show(output);
        }
    }
```