Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Технологии машинного обучения»

Отчет по лабораторной работе №3:
«Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных»

Выполнил:
студент группы ИУ5-63
Курганова Александра

Подпись и дата:

Проверил:

Подпись и дата:

Москва, 2019 г.

Отчет по лабораторной работе №3: "Обработка пропусков в данных, кодирование категориальных признаков, масштабирование данных".

In [1]:
```
import numpy as np
import pandas as pd
pd.set_option('display.max.rows', 1000)
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style='ticks')
```

In [2]:
```
data = pd.read_csv('лр3.csv')
data.head()
```

Out[2]:

	Id	source	latitude	longitude	region	country	admin1	localityName	localityQuality	observationDate	...	speciesDesc
0	230399	OIE	-27.900000	30.800000	Africa	South Africa	KwaZulu-Natal	HPAI_H5N8_2017_019	Exact	17/08/2017	...	dor unspecifie
1	230381	OIE	54.837037	73.354155	Europe	Russian Federation	Omskaya Oblast	Novaya Stanica	Exact	16/08/2017	...	domestic,
2	230333	OIE	-21.077740	30.211620	Africa	Zimbabwe	Masvingo	Mwambe	Exact	16/08/2017	...	domestic,
3	230396	OIE	-26.000000	28.300000	Africa	South Africa	Gauteng	HPAI_H5N8_2017_020	Exact	15/08/2017	...	wild, unsp
4	230371	OIE	49.237900	17.700200	Europe	Czech Republic	Jihomoravsky	Hvozdná	Exact	15/08/2017	...	wild, wil

5 rows × 24 columns

In [3]:
```
data.shape
```

Out[3]:
```
(17008, 24)
```

In [4]:
```
data.dtypes
```

Out[4]:
```
Id                    int64
source               object
latitude            float64
longitude           float64
region               object
country              object
admin1               object
localityName         object
localityQuality      object
observationDate      object
reportingDate        object
status               object
disease              object
serotypes            object
speciesDescription   object
sumAtRisk           float64
sumCases            float64
sumDeaths           float64
sumDestroyed        float64
sumSlaughtered      float64
humansGenderDesc     object
humansAge           float64
humansAffected      float64
humansDeaths        float64
dtype: object
```

In [5]:
```
data.isnull().sum()
```

Out[5]:
```
Id                          0
source                      0
latitude                    0
longitude                   0
region                      0
country                     0
admin1                      0
localityName                0
localityQuality             0
observationDate           502
reportingDate               0
status                      0
disease                     0
serotypes                6941
speciesDescription       1648
sumAtRisk                7251
sumCases                 2473
sumDeaths                2840
sumDestroyed             4003
sumSlaughtered           4773
humansGenderDesc        16648
humansAge               15940
humansAffected          15591
humansDeaths            16557
dtype: int64
```

анализ данных датасета

In [6]:
```
# удаление колонок с пропусками в больше 70 процентов
data = data.drop(columns=['humansGenderDesc', 'humansAge', 'humansAffected', 'humansDeaths'])
data.isnull().sum()
```

Out[6]:
```
Id                     0
source                 0
latitude               0
longitude              0
region                 0
country                0
admin1                 0
localityName           0
localityQuality        0
observationDate      502
reportingDate          0
status                 0
disease                0
serotypes           6941
speciesDescription  1648
sumAtRisk           7251
sumCases            2473
sumDeaths           2840
sumDestroyed        4003
sumSlaughtered      4773
dtype: int64
```

In [7]:
```
# нахождение колонок с нулевыми значениями
colnull = []
for col in data.columns:
    null_count = data[col].isnull().sum()
    if data[col].dtype in ('float64', 'int64') and null_count != 0:
        colnull.append(col)

print('колонки с нулевыми значениями:', colnull)
```

```
колонки с нулевыми значениями: ['sumAtRisk', 'sumCases', 'sumDeaths', 'sumDestroyed', 'sumSlaughtered']
```

In [8]:
```
from sklearn.impute import SimpleImputer
from sklearn.impute import MissingIndicator
```

In [9]:
```
# заполнение пустых числовых колонок с помощью медианы
for col in colnull:
    i = MissingIndicator()
    miss = i.fit_transform(data[[col]])
    med = SimpleImputer(strategy='median')
    data[[col]] = med.fit_transform(data[[col]])
```

In [10]:
```
data.isnull().sum()
```

Out[10]:
```
Id                     0
source                 0
latitude               0
longitude              0
region                 0
country                0
admin1                 0
localityName           0
localityQuality        0
observationDate      502
reportingDate          0
status                 0
disease                0
serotypes           6941
speciesDescription  1648
sumAtRisk              0
sumCases               0
sumDeaths              0
sumDestroyed           0
sumSlaughtered         0
dtype: int64
```

In [11]:
```
# удаление ненужных для анализа категориальных колонок
data = data.drop(columns=['observationDate', 'reportingDate', 'speciesDescription'])
# заполнение пустых значений в категориальных данных
cat_colnull = []
for col in data.columns:
    null_count = data[col].isnull().sum()
    if data[col].dtype not in ('float64', 'int64') and null_count != 0:
        cat_colnull.append(col)

print('категориальные колонки с нулевыми значениями:', cat_colnull)
```

```
категориальные колонки с нулевыми значениями: ['serotypes']
```

In [12]:
```
# заполнение пустых значений в категориальных колонках с помощью моды
for col in cat_colnull:
    i = MissingIndicator()
    miss = i.fit_transform(data[[col]])
    mod = SimpleImputer(strategy='most_frequent')
    data[[col]] = mod.fit_transform(data[[col]])
```

In [13]:
```
data.isnull().sum()
```

Out[13]:
```
Id                0
source            0
latitude          0
longitude         0
region            0
country           0
admin1            0
localityName      0
localityQuality   0
status            0
disease           0
serotypes         0
sumAtRisk         0
sumCases          0
sumDeaths         0
sumDestroyed      0
sumSlaughtered    0
dtype: int64
```

преобразование категориальных признаков в числовые

In [14]:
```
cat_coll = []
for col in data.columns:
    if data[col].dtype == 'object':
        cat_coll.append(col)
for col in cat_coll:
    print(col, len(data[col].unique()))
```

```
source 12
region 4
country 122
admin1 941
localityName 9279
localityQuality 5
status 2
disease 26
serotypes 60
```

In [15]:
```
from sklearn.preprocessing import LabelEncoder
```

In [16]:
```
# использование готового класса для кодирования
en_cat = {}
for col in cat_coll:
    le = LabelEncoder()
    # заполнение и преобразование
    data[[col]] = le.fit_transform(data[col])
    print(col, data[col].unique())
    en_cat[col] = le
```

```
source [ 7  6  1 10  2  4  0  5  8  9 11  3]
region [0 3 2 1]
country [ 99  93 121  27 113 102  88  52 103  61  53  23  43  80  92  94 111  87
 105  79  41  24 107  68 100  64  84  34  89 101 116   8 118  36  56  48
  29   0  32  62 108  91 114 119  70  65  35  63  90  82 104  75 117  95
 110 109   7  73  46  16  97 120  45   1  30  31  15   2  83  54  74  98
  12  55  25  13   5  19  81  96  51  39 112  66  47  49  22  18  76  58
 115  26  85   9  40  42  11  44  78  86  60  38  72  28  33  17 106  50
  20  14   6  69   3  59  77  57  67   4  71  37  21  10]
admin1 [427 577 485 248 342 192 360 448 456 628 512 813 865 392 680 924 901 703
 334 451 882 708 564 545 511 691 526 217 869 879 414 607 127 710 660 472
 218 686 884 671 802 583 915 712 492 167 742 632 145 646 554 805 578 709
 611 146 390 918 129 338 376  96  10 753 322 211 694 159 475 553  23   2
 672 416 134 282 727 278 285 453 760 930 330 191  67 585 287 791  59 551
  76 406 909  78  34 296 284 825 319 599 851 261 496 558 262 538 881 227
 883 238 542 725 298 246 277 154 305 733 816 613 486 466 935 297 363  28
  38  84  87 432 729  79 477 152 430 782 498   6 144 516 184 411 429 401
  99 121 461 559 419 185 237 784 678 331 385 677 631 101 188  20 368 723
 468 351 876 902  18 571   8 255  47 852  22 604 294  50  77 779 198 849
 303 499 136 482 478  45 276 830 908 124 701 808 221 776 912 462 232 172
 174 173 654 339 443 205 667 910 821  89 438 125 831 616 658 539 352  93
 749 186 397 685 422 326  55 295 126 529 408 293 343 110 932   1 576  11
 717 759 447 619 156 239 814 706  26 822  80  13 514 141  73 590 874  36
 575 169 122 153 507 850 681 166 572 321 673 199 772 252 826 503 104 213
  49 436 201 487 629 228 283 931 164 384 877 396 317 549 635 593 497 589
 382 207 470 195 123 868 316 412  98 815 567 840 758 524 288 659 176 391
 721 823 323 161 904 332 905 595 756 544 379 457 896 697 809  53 209  62
 906 676 375 301 741 843 900 118 314 493 833 418 281 845 716 258 570 695
 893 341 194 634  56 523 105  97 754 371 111 615  30 726 106 836  54 245
 130 532 460 359 346 789  83 407 771 494 244 587 260 519 358 925 872 740
  32 196 748 229 744 155 937 829 871 664  39 720 674 444 537 765  66 892
 581 214 610 624 345 696 605 324 165 307 374 236 828 405 917 637 796 329
 939 773 115 895 355 484 230 698 394 561 267 834 769 270 162 193 897 627
 560 292 767 630 781 372 216  37  33  14 586 222 459 518 728 318 197 423
 788 860   7 562 638 766 226 187 388 491 450 920 929 792 569 309 504 306
 863 393   3 271 827 302 732  52 133 764 657 259 774 513 268 117 452 250
  64 689 311 817 922 471 666 665 743 862 525 540 580 286  82 752 365 356
 880 442 417 640  25 913 204 378 143 534 750 251 266 737 367 147 940 889
 224 480 279 731 308  35 921 373 903 888 873 794 420 844 337 713 517 431
 350 751 175 530 320 911  68 602 719  71 596 648 291 353 304 500 763 275
 150 149  19 626 552 474 824 272 936 934 402 645 157 479 208  12 799 745
```

```
 449 257  48 536 231 403 510 445 812 333 866 386 315 235  16  95 527 938
 383 690 440 837 398 116 233   0 535 469 225 702 684  88 508 548 854 107
 591 832 132 163 582 927 804 887  43 643 520 662 182 797 464 515 563 290
 592 274 247 650 465 803 256 841 170 787 785 148 734  57 203 476 670 158
 249 633 454  60 119 410 693 280 864 885 669 738 135 234  41 820 370  91
 746 621 642 916 618 801 441 179 299  75 140 439 177 369 886 300 112 400
 625 354 623 711 505 919 189 541 598 846 722 473  24 506 120 606  40 357
 424 335 509 853 109 914 735  86 243 636 361 768 609 835 639 692 617 566
 395 547 730  94 325 806 614   5 269 923 206 898  58 190 603 597 328 528
 579 783 313 718 608 641 867 242 818 210 644 793 425 377 434 747 455 415
  31 488 890 409 770  70  44 848 421 878  90 663 775 891   9 790 181 362
 102 778 433 253 933  81 855 557 366 655 502 647 601 699 568 870 679 857
  27 458  63 859 178 467 875 241 856 668 894 347 289 463 807 139 113 446
 651 573 348 652 675 724 349 310 413 588 928 926  15 344 811 556 757 273
 584 533  85 839 336 327 380 215 340 212 254 495 240 661 653 387 312  65
 183 219 847 404 842 762 437 714  17  42 220 171 522 795 656 108  51 202
 435 755  72 736 142 546 200 838  61  29 899 574 265  92 777  69  46 103
 819 426 800 521 907  74 483 861 428   4 399 700 761 381 649  21 481 543
 114 810 131 364 531 168 715 389 264 151 180 555 612 622 739 798 160 128
 682 620 137 683 858 780 100 565 550 786 490 138 704 688 263 501 707 223
 489 600 687 705 594]
localityName [2997 5577 5301 ... 5823 5694 2015]
localityQuality [3 2 0 4 1]
status [0 1]
disease [14  1 12 19 15 17 25 18 11 22 23  0  4 24  2 21 20  3 10  7  8  5  9 16
  6 13]
serotypes [36 11 14 17 15 21 57 52 55  8 38 51 26  7  9 42 27 10 58 53 54 45 24 28
 20 40 47 39 12 41 46 44 56  6 43 49 16 48  0  5  4 50 59 35 13  1 18 32
 25  3  2 22 31 37 34 33 29 30 19 23]
```

In [17]:
```
data.dtypes
```

Out[17]:
```
Id                  int64
source              int64
latitude          float64
longitude         float64
region              int64
country             int64
admin1              int64
localityName        int64
localityQuality     int64
status              int64
disease             int64
serotypes           int64
sumAtRisk         float64
sumCases          float64
sumDeaths         float64
sumDestroyed      float64
sumSlaughtered    float64
dtype: object
```

In [18]:
```
# выполнение декодирования
en_cat
```

Out[18]:
```
{'source': LabelEncoder(),
 'region': LabelEncoder(),
 'country': LabelEncoder(),
 'admin1': LabelEncoder(),
 'localityName': LabelEncoder(),
 'localityQuality': LabelEncoder(),
 'status': LabelEncoder(),
 'disease': LabelEncoder(),
 'serotypes': LabelEncoder()}
```

In [19]:
```
en_cat['region'].inverse_transform([0, 3, 2, 1])
```

Out[19]:
```
array(['Africa', 'Europe', 'Asia', 'Americas'], dtype=object)
```

In [20]:
```
en_cat['source'].inverse_transform([7, 6, 1, 10, 2, 4, 0, 5, 8, 9, 11, 3])
```

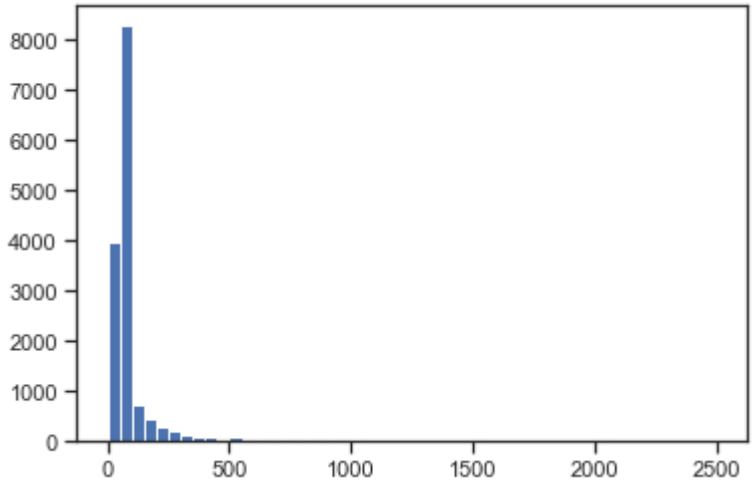
Out[20]:
```
array(['OIE', 'National authorities', 'FAO Field Officer', 'WHO',
       'FAO officer', 'International reference laboratory', 'EC',
       'National Institute for Communicable Diseases, NICD', 'Other',
       'Publications', 'sequence only', 'FAO-Report'], dtype=object)
```

масштабирование

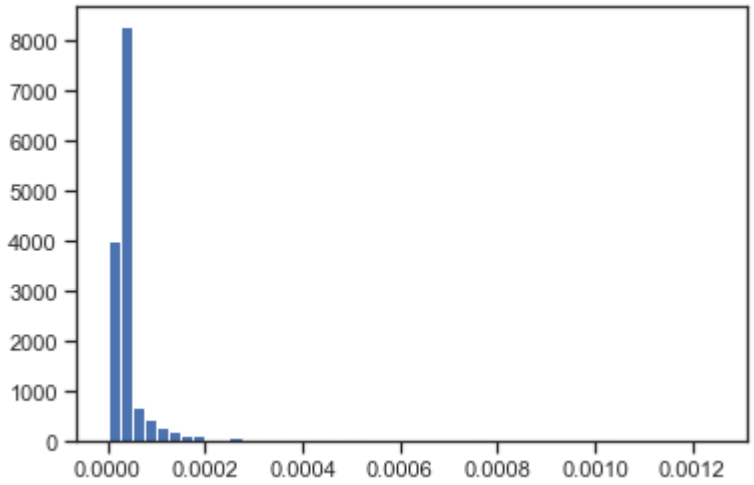
In [21]:
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler, Normalizer
```

In [22]:
```
sc1 = MinMaxScaler()
sc1_data = sc1.fit_transform(data[['sumAtRisk']])
```

In [40]:
```
plt.hist(data['sumAtRisk'], 50, range=[0, 2500])
plt.show()
```

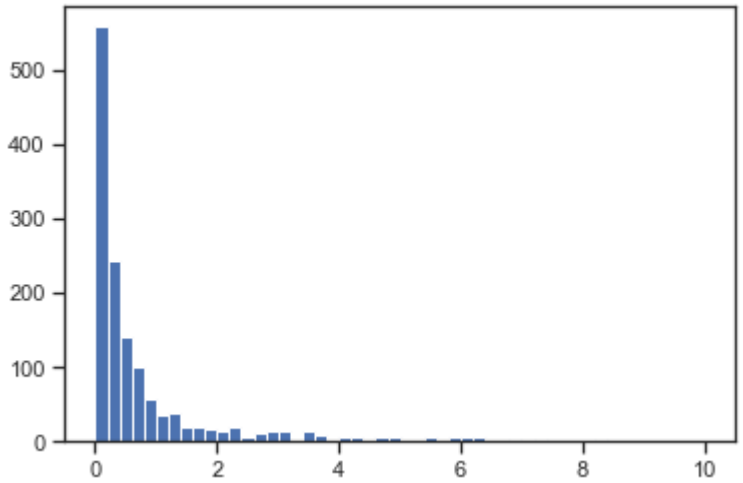


In [43]:
```
plt.hist(sc1_data, 50, range=[0, 0.00125])
plt.show()
```



In [59]:
```
# на основе z-оценки
sc2 = StandardScaler()
sc2_data = sc2.fit_transform(data[['sumAtRisk']])
```

In [60]:
```
plt.hist(sc2_data, 50, range=[0, 10])
plt.show()
```



In [61]:
```
# нормализация
sc3 = Normalizer()
sc3_data = sc3.fit_transform(data[['sumAtRisk']])
plt.hist(sc3_data, 50)
plt.show()
```