Липецкий государственный технический университет

Факультет автоматизации и информатики Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

Экспертные системы. Оценка планирования продаж.

Студент Сухоруков К.О.

Группа М-ИАП-22

Руководитель Кургасов В.В.

Задание кафедры

Задать значения количества продаж по 10 товарам в течение 12 месяцев (помесячно). Для каждого из товаров спрогнозировать количество продаж на следующий, 13 месяц и провести анализ достоверности планирования продаж.

Ход работы

Используемые библиотеки:

- numpy библиотека для языка python предназначена для поддержки многомерных массивов (включая матрицы), поддержки высокоуровневых математических функций;
- pandas библиотека предназначена для обработки и анализа массивов данных;
- seaborn библиотека предназначена для создания статистических графиков.

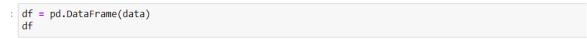
```
import numpy as np
import pandas as pd
import seaborn as sns
```

Рисунок 1 – Импорт библиотек

Информация о продажах за 12 месяцев

```
data = {
    'Молоко': np.random.normal(50, 5, 12),
    'Яйца': np.random.normal(67, 6, 12),
    'Хлеб': np.random.normal(100, 7, 12),
    'Мясо': np.random.normal(48, 4, 12),
    'Соль': np.random.normal(89, 40, 12),
    'Сахар': np.random.normal(100, 49, 12),
    'Пшено': np.random.normal(48, 7, 12),
    'Гречка': np.random.normal(75, 10, 12),
    'Мыло': np.random.normal(15, 4, 12),
    'Сигареты': np.random.normal(70, 15, 12),
}
```

Рисунок 2 – Генерация данных



	Молоко	Яйца	Хлеб	Мясо	Соль	Caxap	Пшено	Гречка	Мыло	Сигареты
0	53.649023	64.291265	102.742867	45.138335	84.427350	143.675071	45.520696	80.905552	21.732227	93.095026
1	55.126842	54.796162	102.659712	39.503222	108.280588	86.349450	49.930199	62.158183	13.699081	61.755103
2	49.809361	65.338214	105.217241	48.673802	149.708291	128.063355	55.561930	65.132405	11.677861	79.537628
3	54.222145	63.455317	111.625469	45.126682	137.701846	135.042070	56.070516	85.425823	11.582962	96.930538
4	49.833048	54.540157	114.026672	48.912228	106.899096	147.736962	55.502079	61.477481	8.927637	68.694460
5	49.991662	71.622997	100.863118	47.443815	61.708558	28.300059	51.165066	84.808487	11.451578	72.238726
6	47.866936	66.652143	96.131570	51.069329	91.222629	134.292076	42.974019	85.511556	18.956356	64.369269
7	46.814373	67.928753	96.163063	48.850835	108.691761	170.861295	49.174472	79.315860	17.586951	97.767819
8	46.964985	61.465777	101.144567	57.179178	26.474924	207.735677	55.744550	83.181311	16.114406	69.300047
9	55.935781	65.689406	99.332788	53.714008	140.345422	40.295558	44.733599	73.947378	19.012367	82.524694
10	55.708337	55.101896	95.661290	49.618247	48.543126	57.656897	55.605442	81.920374	17.987888	65.021965
11	48.672543	60.444875	98.826135	49.385307	82.793977	92.354922	50.159713	92.867417	19.781087	43.751813

Рисунок 3 – Вывод сгенерированных данных

Обработка данных

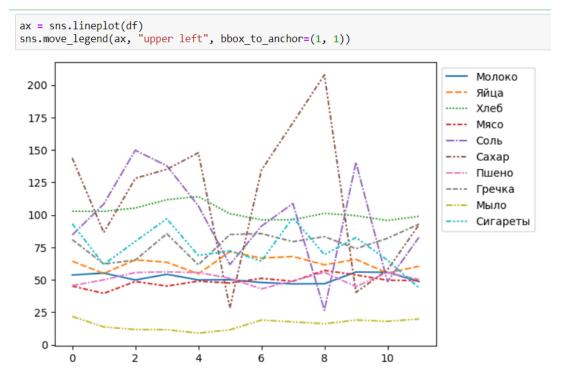


Рисунок 4 – Выборка данных на графике

```
p0 = df.sum() / df.shape[0]
p0
Молоко
             51.216253
Яйца
             62.610580
Хлеб
            102.032874
Мясо
             48.717916
Соль
             95.566464
Caxap
            114.363616
Пшено
             51.011857
Гречка
             78.054319
Мыло
             15.709200
Сигареты
             74.582257
dtype: float64
```

Рисунок 5 — Расчёт \hat{p}_0

```
std = ((df - p0) ** 2).sum() / (df.shape[0] - 1)
std = std ** (1 / 2)
```

Рисунок 6 – Расчет среднеквадратичного отклонения

```
x_extrapol = p0 + np.random.normal(0, std, len(p0))
pd.concat([df, pd.DataFrame([x_extrapol], columns=x_extrapol.index)]).reset_index(drop=True)
```

Молоко	Яйца	Хлеб	Мясо	Соль	Caxap	Пшено	Гречка	Мыло	Сигареты
53.649023	64.291265	102.742867	45.138335	84.427350	143.675071	45.520696	80.905552	21.732227	93.095026
55.126842	54.796162	102.659712	39.503222	108.280588	86.349450	49.930199	62.158183	13.699081	61.755103
49.809361	65.338214	105.217241	48.673802	149.708291	128.063355	55.561930	65.132405	11.677861	79.537628
54.222145	63.455317	111.625469	45.126682	137.701846	135.042070	56.070516	85.425823	11.582962	96.930538
49.833048	54.540157	114.026672	48.912228	106.899096	147.736962	55.502079	61.477481	8.927637	68.694460
49.991662	71.622997	100.863118	47.443815	61.708558	28.300059	51.165066	84.808487	11.451578	72.238726
47.866936	66.652143	96.131570	51.069329	91.222629	134.292076	42.974019	85.511556	18.956356	64.369269
46.814373	67.928753	96.163063	48.850835	108.691761	170.861295	49.174472	79.315860	17.586951	97.767819
46.964985	61.465777	101.144567	57.179178	26.474924	207.735677	55.744550	83.181311	16.114406	69.300047
55.935781	65.689406	99.332788	53.714008	140.345422	40.295558	44.733599	73.947378	19.012367	82.524694
55.708337	55.101896	95.661290	49.618247	48.543126	57.656897	55.605442	81.920374	17.987888	65.021965
48.672543	60.444875	98.826135	49.385307	82.793977	92.354922	50.159713	92.867417	19.781087	43.751813
48.578766	70.617188	102.612828	44.692790	73.164614	133.842040	41.034738	74.434870	14.620284	82.191351
	53.649023 55.126842 49.809361 54.222145 49.833048 49.991662 47.866936 46.814373 46.964985 55.935781 55.708337 48.672543	53.649023 64.291265 55.126842 54.796162 49.809361 65.338214 54.222145 63.455317 49.893048 54.540157 49.991662 71.622997 47.866936 66.652143 46.814373 67.928753 46.964985 61.465777 55.935781 65.689406 55.708337 55.101896 48.672543 60.444875	53.649023 64.291265 102.742867 55.126842 54.796162 102.659712 49.809361 65.338214 105.217241 54.222145 63.455317 111.625469 49.833048 54.540157 114.026672 49.991662 71.622997 100.863118 47.866936 66.652143 96.131570 46.814373 67.928753 96.163063 46.964985 61.465777 101.144567 55.935781 65.689406 99.332788 55.708337 55.101896 95.661290 48.672543 60.444875 98.826135	53.649023 64.291265 102.742867 45.138335 55.126842 54.796162 102.659712 39.503222 49.809361 65.338214 105.217241 48.673802 54.222145 63.455317 111.625469 45.126682 49.833048 54.540157 114.026672 48.912228 49.991662 71.622997 100.863118 47.443815 47.866936 66.652143 96.131570 51.069329 46.814373 67.928753 96.163063 48.850835 46.964985 61.465777 101.144567 57.179178 55.935781 65.689406 99.332788 53.714008 55.708337 55.101896 95.661290 49.618247 48.672543 60.444875 98.826135 49.385307	53.649023 64.291265 102.742867 45.138335 84.427350 55.126842 54.796162 102.659712 39.503222 108.280588 49.809361 65.338214 105.217241 48.673802 149.708291 54.222145 63.455317 111.625469 45.126682 137.701846 49.833048 54.540157 114.026672 48.912228 106.899096 49.991662 71.622997 100.863118 47.443815 61.708558 47.866936 66.652143 96.131570 51.069329 91.222629 46.814373 67.928753 96.163063 48.850835 108.691761 46.964985 61.465777 101.144567 57.179178 26.474924 55.708337 55.101896 99.332788 53.714008 140.345422 55.708337 55.101896 95.661290 49.618247 48.543126 48.672543 60.444875 98.826135 49.385307 82.793977	53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 46.964985 61.465777 101.144567 57.179178 26.474924 207.735677 55.708337 55.101896 99.332788 53.714008 140.345422 40.29558 55.708337 55.101896 95.661290 49.618247 48.543126 57.656897 48.672543 <th>53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472 46.964985 61.465777 101.144567 57.179178 26.474924 207.735677 55.744550 55.7935781 65.689406 99.332788 53.714008 140.345422</th> <th>53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 80.905552 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 62.158183 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 65.132405 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 85.425823 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 61.477481 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 84.808487 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 85.51156 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472 79.315860 46.964985 61.465777 101.144567 57.179178 26.474924</th> <th>53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 80.905552 21.732227 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 62.158183 13.699081 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 65.132405 11.677861 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 85.425823 11.582962 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 61.477481 8.927637 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 84.808487 11.451578 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 85.511556 18.956356 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472<!--</th--></th>	53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472 46.964985 61.465777 101.144567 57.179178 26.474924 207.735677 55.744550 55.7935781 65.689406 99.332788 53.714008 140.345422	53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 80.905552 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 62.158183 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 65.132405 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 85.425823 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 61.477481 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 84.808487 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 85.51156 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472 79.315860 46.964985 61.465777 101.144567 57.179178 26.474924	53.649023 64.291265 102.742867 45.138335 84.427350 143.675071 45.520696 80.905552 21.732227 55.126842 54.796162 102.659712 39.503222 108.280588 86.349450 49.930199 62.158183 13.699081 49.809361 65.338214 105.217241 48.673802 149.708291 128.063355 55.561930 65.132405 11.677861 54.222145 63.455317 111.625469 45.126682 137.701846 135.042070 56.070516 85.425823 11.582962 49.833048 54.540157 114.026672 48.912228 106.899096 147.736962 55.502079 61.477481 8.927637 49.991662 71.622997 100.863118 47.443815 61.708558 28.300059 51.165066 84.808487 11.451578 47.866936 66.652143 96.131570 51.069329 91.222629 134.292076 42.974019 85.511556 18.956356 46.814373 67.928753 96.163063 48.850835 108.691761 170.861295 49.174472 </th

Рисунок 7 – Расчёт планируемого показателя

Оценки

```
reliability = std / p0
  reliability
Молоко
              0.067951
  Яйца
              0.088125
  Хлеб
              0.057340
  Мясо
              0.090883
  Соль
              0.394204
              0.473630
  Caxap
  Пшено
              0.093348
  Гречка
              0.130268
  Мыло
              0.260796
  Сигареты
              0.215675
  dtype: float64
```

Рисунок 8 — Расчёт $\frac{\widehat{\sigma}_p}{\widehat{p}_0}$

Уловие 1
$$X_i - \hat{p_0} < 2\hat{\sigma}: \forall i$$

```
cond1 = ((df - p0) < 2 * std).all()</pre>
  cond1
Молоко
                True
                True
  Яйца
  Хлеб
               False
  Мясо
               True
  Соль
                True
  Caxap
               True
               True
  Пшено
  Гречка
                True
                True
  Мыло
  Сигареты
                True
  dtype: bool
```

Рисунок 9 – Проверка первого условия

Условие 2 $\stackrel{\wedge}{p_0} > 2\hat{\sigma}$

cond2 = p0 : cond2	2 * std
Молоко	True
Яйца	True
хлеб	True
Мясо	True
Соль	True
Caxap	True
Пшено	True
Гречка	True
Мыло	True
Сигареты	True
dtype: bool	

Рисунок 10 – Проверка второго условия

Условие 3 $X_i > 0$: $\forall i$

cond3 = (df cond3	> 0).all()
Молоко	True
Яйца	True
Хлеб	True
Мясо	True
Соль	True
Caxap	True
Пшено	True
Гречка	True
Мыло	True
Сигареты	True
dtype: bool	

Рисунок 11 – Проверка третьего условия

Цветовая классификация

```
product color = pd.Series(dtype='string')
for name in df.columns:
   if (not cond3[name]):
        product color[name] = 'Красный'
   elif (not cond1[name] and not cond2[name]):
       product_color[name] = 'Оранжевый'
    elif (not cond1[name] or not cond2[name]):
       product color[name] = 'Желтый'
   else:
        product_color[name] = 'Зеленый'
product_color
Молоко
           Зеленый
Яйца
           Зеленый
Хлеб
           Желтый
Мясо
           Зеленый
           Зеленый
Соль
           Зеленый
Caxap
Пшено
           Зеленый
           Зеленый
Гречка
Мыло
           Зеленый
Сигареты
          Зеленый
dtype: object
```

Рисунок 12 – Пример итогов работы программы

Таким образом, для товара хлеб нужна корректировка полученных данных.

```
Код программы
#!/usr/bin/env python
# coding: utf-8
# In[1]:
import numpy as np
import pandas as pd
import seaborn as sns
# ## Информация о продажах за 12 месяцев
# In[13]:
data = {
  'Молоко': np.random.normal(50, 5, 12),
  'Яйца': np.random.normal(67, 6, 12),
  'Хлеб': np.random.normal(100, 7, 12),
  'Msco': np.random.normal(48, 4, 12),
  'Соль': np.random.normal(89, 40, 12),
  'Caxap': np.random.normal(100, 49, 12),
  'Пшено': np.random.normal(48, 7, 12),
  'Гречка': np.random.normal(75, 10, 12),
  'Мыло': np.random.normal(15, 4, 12),
  'Сигареты': np.random.normal(70, 15, 12),
}
# In[14]:
df = pd.DataFrame(data)
df
# ## Обработка данных
# In[15]:
ax = sns.lineplot(df)
sns.move_legend(ax, "upper left", bbox_to_anchor=(1, 1))
# In[16]:
```

```
p0 = df.sum() / df.shape[0]
      p0
      # In[17]:
      std = ((df - p0) ** 2).sum() / (df.shape[0] - 1)
      std = std ** (1 / 2)
      # In[18]:
      x_{extrapol} = p0 + np.random.normal(0, std, len(p0))
      pd.concat([df,
                                                       pd.DataFrame([x_extrapol],
columns=x_extrapol.index)]).reset_index(drop=True)
      # ## Оценки
      # In[19]:
      reliability = std / p0
      reliability
      #### Уловие 1 X_i - hat\{p_0\} < 2 hat\{sigma\}: forall i
      # In[20]:
      cond1 = ((df - p0) < 2 * std).all()
      cond1
      #### Условие 2 \hat{p}_0 > 2\hat{\sigma} $
      # In[21]:
      cond2 = p0 > 2 * std
      cond2
      #### Условие 3 $ X_i > 0: \forall i $
      # In[22]:
      cond3 = (df > 0).all()
      cond3
      # ### Цветовая классификация
      # In[23]:
      product_color = pd.Series(dtype='string')
      for name in df.columns:
        if (not cond3[name]):
```

```
product_color[name] = 'Красный'
elif (not cond1[name] and not cond2[name]):
    product_color[name] = 'Оранжевый'
elif (not cond1[name] or not cond2[name]):
    product_color[name] = 'Желтый'
else:
    product_color[name] = 'Зеленый'
product_color
```

Вывод

В ходе выполнения данной лабораторной работы мы спрогнозировали количество продаж на тринадцатый месяц и провели анализ достоверности планирования продаж.