МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный технический университет

имени П.О.Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ЛАБОРАТОРНАЯ РАБОТА №5**

по дисциплине: «Введение в кластерный анализ и классификацию»

на тему: **«**Статистические методы классификации**»**

Выполнил: студент гр. ИТИ-22

Василенко Р.В.

Принял: ассистент

Карась О. В.

Гомель 2024

**Цель работы:** разработать программное обеспечение для решения задачи классификации (обучение с учителем) на основе статистических методов классификации

Ход работы:

1. Загрузить исходные данные согласно варианта (таблица 5) из файла

2. На основании считанных данных сформировать DataFrame

3. Провести анализ на полноту данных

4. При необходимости выполнить нормировку, квантование и фильтрацию

данных

5. Вычислить статистические характеристики данных (мат. Ожидание, дисперсию)

6. Построить графики распределения случайной величины

7. Графическим способом подобрать законы распределения случайных величин

8. С помощью заданного критерия, проверить гипотезу о законе распределения

9. При необходимости провести селекцию признаков и понижение размерности задачи

10.Согласно варианту построить классификатор

11.Оценить качество проведённой классификации с использованием указанного метода валидации.

12.Построить матрицу ошибок

13.Вычислить аккуратность, точность и полноту

14.Оценить площадь под кривой ошибок

15.Построить логистическую функцию потерь

16.Выполнить робастную оценку среднего

17.Провести варьирование параметрами классификатора с целью улучшения качества классификации

18.Провести анализ результатов работы классификатора на основе построенных графиков

19.Сделать выводы по результатам проделанной работы

Вариант 2 – Рисунок 1

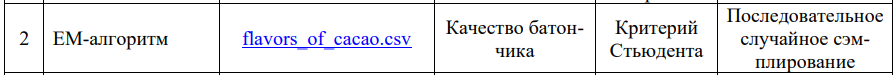


Рисунок 1 – Задание варианта

**Ход работы и результаты выполнения**

Чтение данных из .csv файла происходит с помощью библиотеки pandas, команды .read\_csv(). На рисунке 2 изображен результат чтения csv файла.

**

Рисунок 2 – Результат выполнения задания.

Проверка на полноту данных происходит с помощью метода .info(), в котором мы можем проверить, есть ли в столбцах нулевые ячейки. С помощью .dropna() мы избавляемся от всех записей, в которых есть нулевые значения. На рисунке 3 демонстрируется цветовая карта задачи.

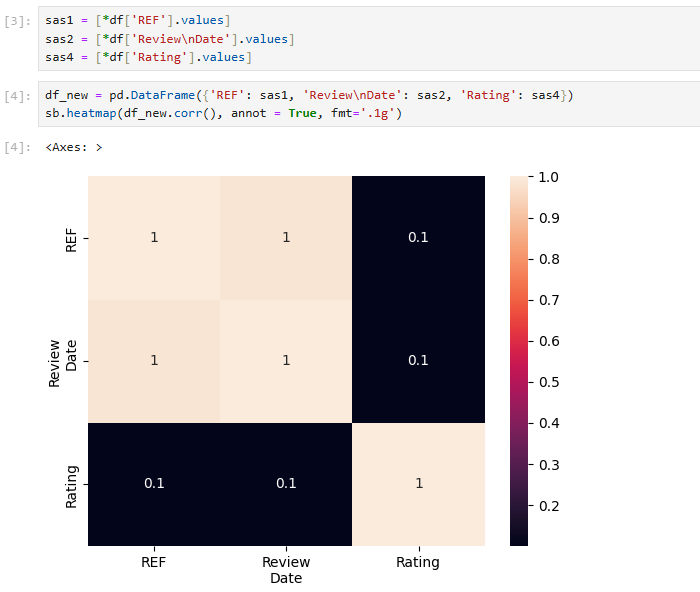
**

Рисунок 3 – Цветовая карта матрицы корреляции.

Для визуального изображения собранных данных используется библиотека matplotlib. На рисунке 4 изображена диаграмма выборки.

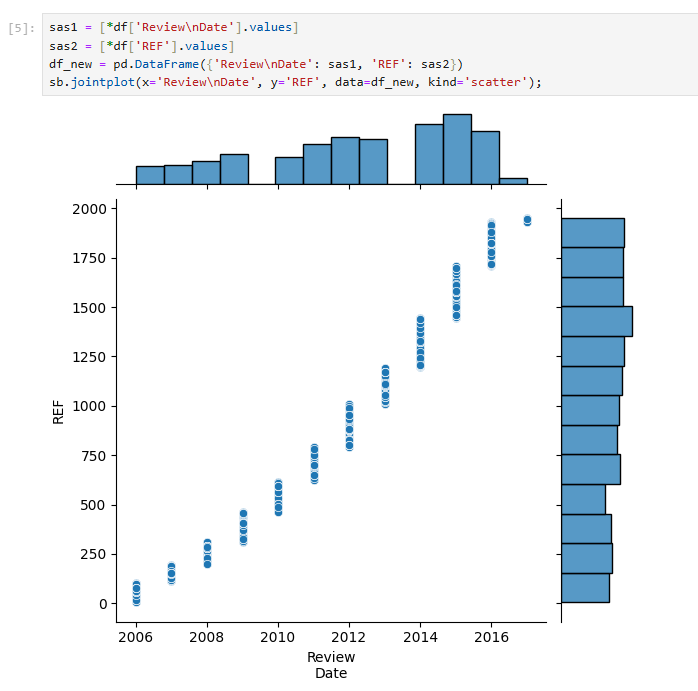


Рисунок 3 – Диаграмма выборки

Матрица диаграмм рассеивания изображается благодаря библиотеке seaborn и matplotlib.pyplot. На рисунке 4 изображена кластеризация качества батончика на три кластера.

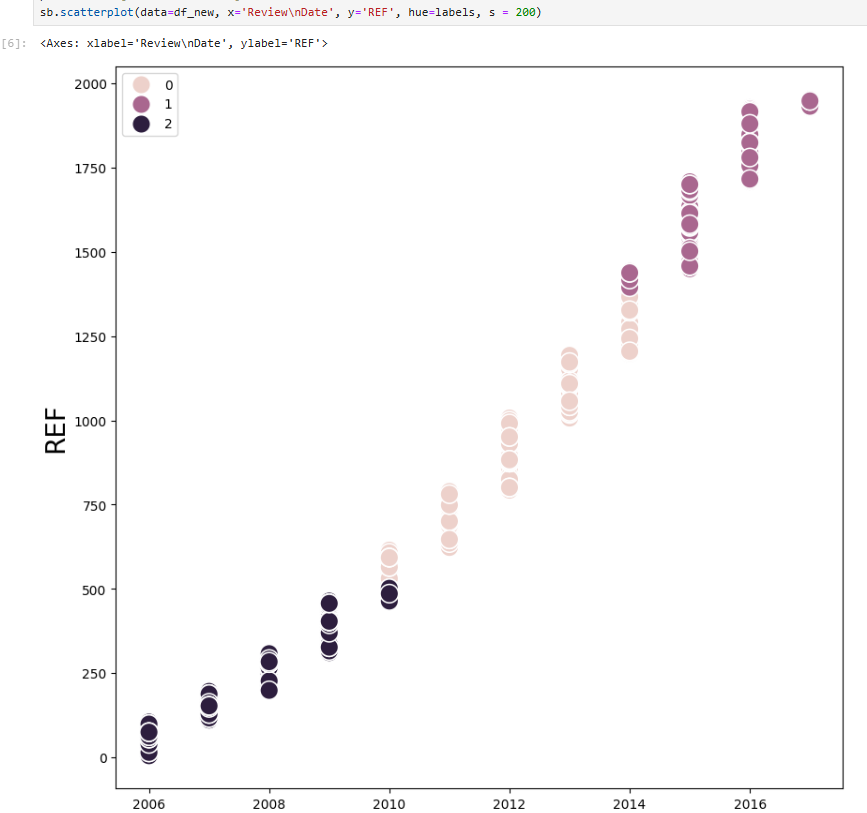


Рисунок 4 – Кластеризация качества батончика

На рисунке 5 изображен график плотности и гостограмма всех задержек.

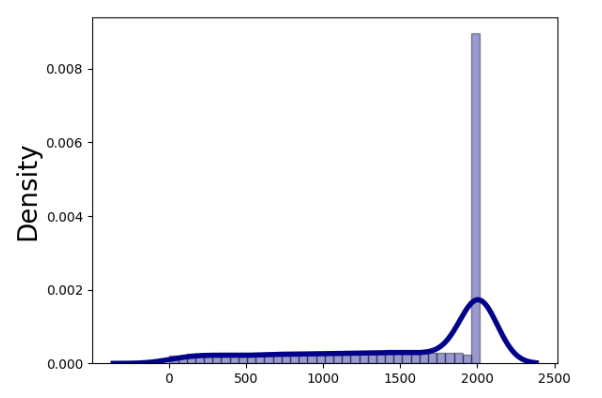


Рисунок 5 – Диаграмма задержек

На рисунке 6 изображен графический подбор метода распределения.

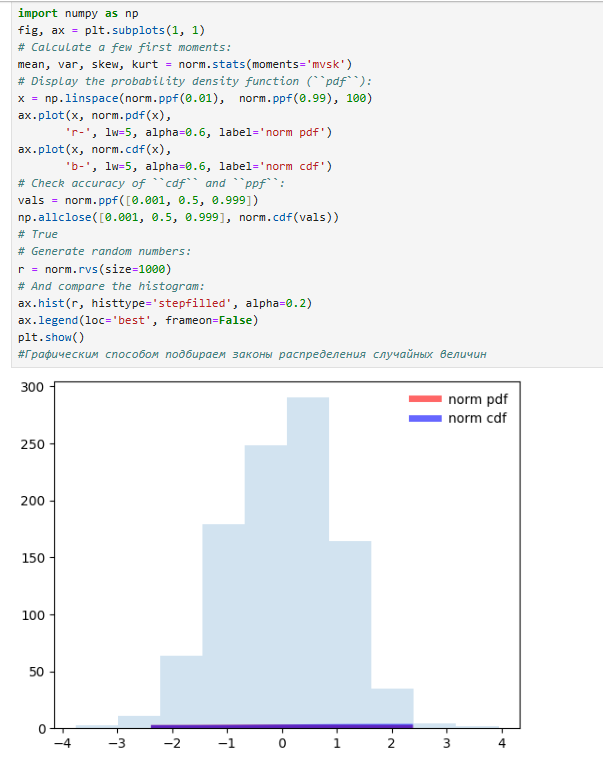


Рисунок 6 – Графический подбор метода распределения

На рисунке 7 изображена проверка корреляции текущего распределения с известными.

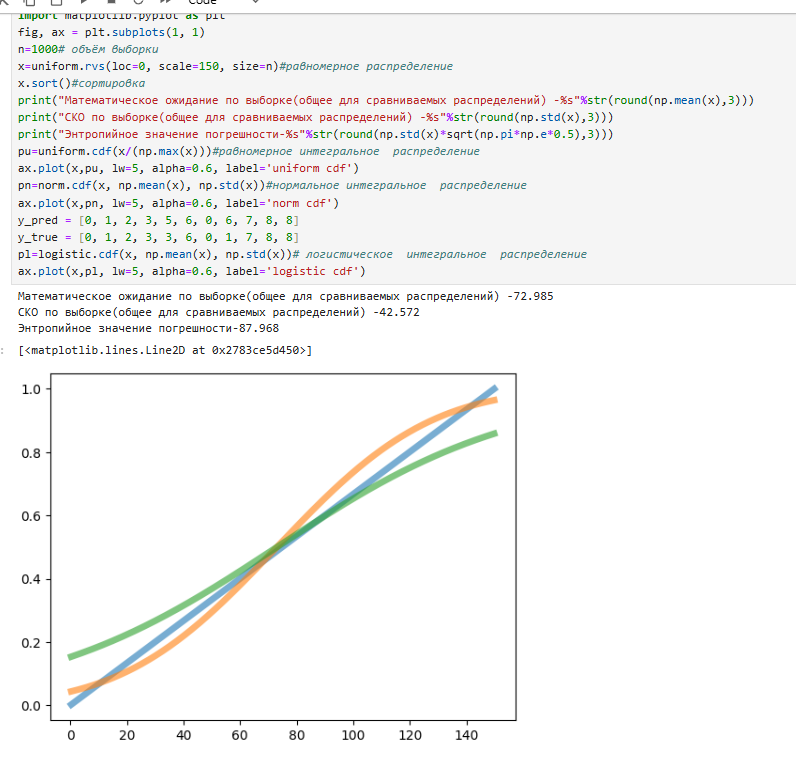


Рисунок 7 – Корреляция распределения

На рисунке 8 изображена реализация ЕМ-алгоритма в контексте задачи

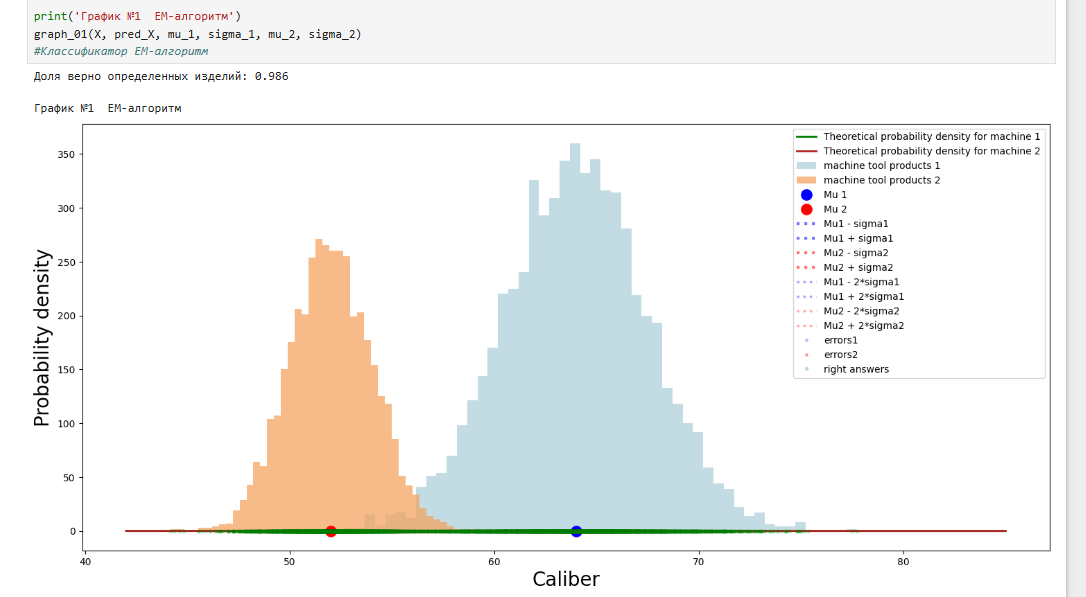


Рисунок 8 – Реализация ЕМ-алгоритма

На рисунке 9 изображена ROC-кривая и вычисление площади под ней

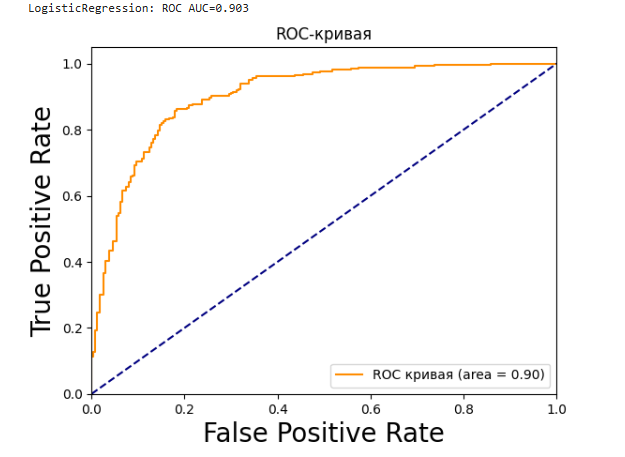


Рисунок 9 - ROC-кривая и вычисление площади под ней

**Вывод:** в результате выполнения лабораторной работы выполнен анализ данных в csv файле, изображены графики зависимости данных друг от друга, из собранных данных составлены кластеры и произведен анализ функциональности. Были получены три кластера путем реализации метода Прима и пять кластеров методом к-средних, а так же проверена принадлежность некого объекта к кластеру.

Приложение А

Код

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import cm

from mpl\_toolkits.mplot3d import Axes3D

from mlxtend.plotting import scatterplotmatrix

import seaborn as sb

import plotly.express as px

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics.cluster import completeness\_score

import datetime

import math

import plotly.express as px

from sklearn import tree

df = pd.read\_csv("C:/Users/Victus/Desktop/flavors\_of\_cacao.csv", delimiter=',', low\_memory=False)

print(df.info())

df = df.dropna()

print(df)

sas1 = [\*df['REF'].values]

sas2 = [\*df['Review\nDate'].values]

sas4 = [\*df['Rating'].values]

df\_new = pd.DataFrame({'REF': sas1, 'Review\nDate': sas2, 'Rating': sas4})

sb.heatmap(df\_new.corr(), annot = True, fmt='.1g')

y\_pred = [0, 1, 2, 3, 5, 6, 0, 6, 7, 8, 8]

y\_true = [0, 1, 2, 3, 3, 6, 0, 1, 7, 8, 8]

sb.pairplot(df\_new)

plt.tight\_layout()

sas1 = [\*df['Review\nDate'].values]

sas2 = [\*df['REF'].values]

df\_new = pd.DataFrame({'Review\nDate': sas1, 'REF': sas2})

sb.jointplot(x='Review\nDate', y='REF', data=df\_new, kind='scatter');

from scipy.cluster.hierarchy import linkage, dendrogram

cluster\_ar = linkage(df\_new, method='ward', metric='euclidean')

link\_df = pd.DataFrame(cluster\_ar, index=[f' step {i+1}' for i in range(cluster\_ar.shape[0])],columns=['cluster', 'cluster 2', 'dist', 'number elements'])

fig = plt.figure(figsize=(25, 10))

row\_dendr = dendrogram(link\_df)

from sklearn.cluster import AgglomerativeClustering

cl = AgglomerativeClustering(n\_clusters=3, linkage = 'ward')

labels = cl.fit\_predict(df\_new)

plt.figure(figsize=(10,10))

plt.rc('axes', labelsize=20)

sb.color\_palette("Paired")

plt.rcParams['scatter.marker'] = 'o'

sb.scatterplot(data=df\_new, x='Review\nDate', y='REF', hue=labels, s = 200)

from sklearn.cluster import AgglomerativeClustering

cla = AgglomerativeClustering(n\_clusters=3, linkage = 'complete')

labels1 = cla.fit\_predict(df\_new)

plt.figure(figsize=(10,10))

plt.rc('axes', labelsize=20)

sb.color\_palette("Paired")

plt.rcParams['scatter.marker'] = 'o'

sb.scatterplot(data=df\_new, x='Review\nDate', y='REF', hue=labels, s = 200)

import sklearn.metrics

import numpy

r = sklearn.metrics.confusion\_matrix(labels1, labels)

r = numpy.flip(r)

print(r)

from sklearn.datasets import make\_classification

from sklearn.linear\_model import LogisticRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import roc\_curve, auc

from sklearn.metrics import roc\_auc\_score

from matplotlib import pyplot as plt

# генерируем датасет на 2 класса

X, y = make\_classification(n\_samples=1000, n\_classes=2, random\_state=1)

# разделяем его на 2 выборки

trainX, testX, trainy, testy = train\_test\_split(X, y, test\_size=0.5, random\_state=2)

# обучаем модель

model = LogisticRegression(solver='lbfgs')

model.fit(trainX, trainy)

# получаем предказания

lr\_probs = model.predict\_proba(testX)

# сохраняем вероятности только для положительного исхода

lr\_probs = lr\_probs[:, 1]

# рассчитываем ROC AUC

lr\_auc = roc\_auc\_score(testy, lr\_probs)

print('LogisticRegression: ROC AUC=%.3f' % (lr\_auc))

# рассчитываем roc-кривую

fpr, tpr, treshold = roc\_curve(testy, lr\_probs)

roc\_auc = auc(fpr, tpr)

# строим график

plt.plot(fpr, tpr, color='darkorange',

label='ROC кривая (area = %0.2f)' % roc\_auc)

plt.plot([0, 1], [0, 1], color='navy', linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC-кривая')

plt.legend(loc="lower right")

plt.show()

import numpy as np

def mean\_squared\_error(act, pred):

diff = pred - act

differences\_squared = diff \*\* 2

mean\_diff = differences\_squared.mean()

return mean\_diff

act = np.array([1.1,2,1.7])

pred = np.array([1,1.7,1.5])

print(mean\_squared\_error(act,pred))

import sklearn

from sklearn.metrics import accuracy\_score

print(accuracy\_score(y\_true, y\_pred))