Тест начат	Воскресенье, 19 ноября 2023, 21:01
Состояние	Завершены
Завершен	Воскресенье, 19 ноября 2023, 21:07
Прошло времени	6 мин. 8 сек.
Оценка	Еще не оценено

Вопрос **1**

Выполнен

Балл: 1,00

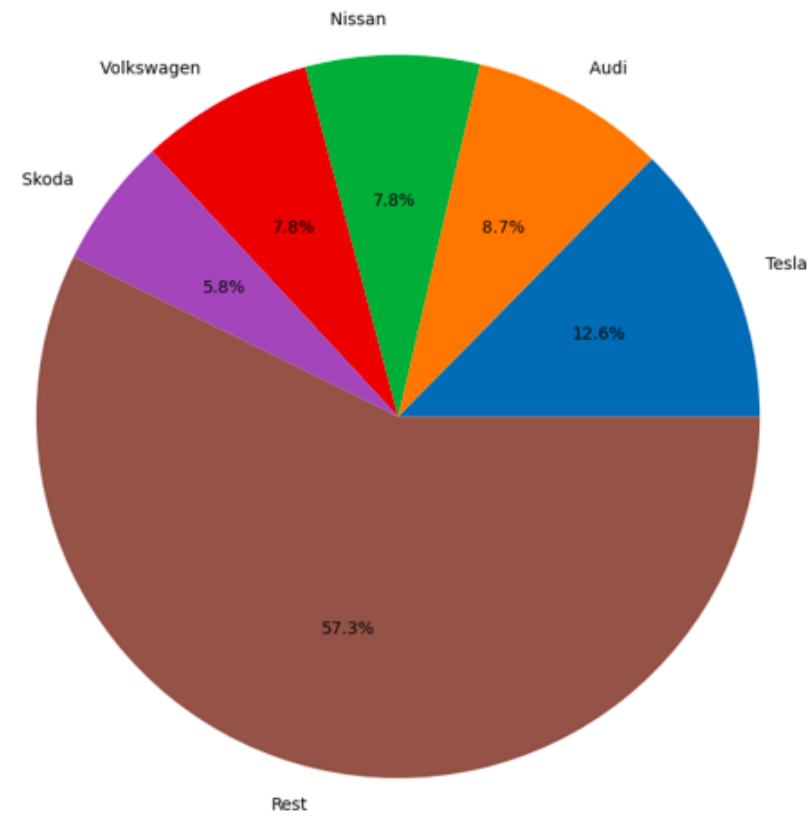
Построение круговой диаграммы

Выполните построение круговой диаграммы по данным файла **Electric_Cars.csv**, содержащим моделях электромобилей.

Для построения диаграммы:

1. Сформируйте данные по количеству моделей (столбец Brand) и выполните их сортировку г TopSpeed_KmH.
2. Используйте библиотеку matplotlib, чтобы построить круговую диаграмму по полученным в пункте данным. Диаграмма должна показывать долю в процентном отношении первых 5 и всех моделей (rest).
3. Покажите на диаграмме наименование брендов и соответствующую им долю в процентах.
4. Увеличьте радиус диаграммы в 3 раза.
5. Сохраните график под именем auto_pie.png.

Как должно получиться:



Напишите код программы в самостоятельно созданном Python-ноутбуке и прикрепите его в LMS начать писать программу, можно использовать следующий код:

```
import matplotlib.pyplot as plt
import pandas as pd

df = pd.read_csv('Electric_Car.csv')

import pandas as pd
import matplotlib.pyplot as plt

# Загрузка данных из CSV-файла
df = pd.read_csv('Electric_Car.csv')

# Сформируйте данные по количеству моделей и выполните сортировку по столбцу TopSpeed_
brand_counts = df['Brand'].value_counts()
sorted_brands = brand_counts.sort_values(ascending=False)

# Выберите первые 5 брендов и объедините остальные в одну категорию
top_brands = sorted_brands.head(5)
rest_brands = pd.Series([sorted_brands.iloc[5:].sum()], index=['rest'])

# Создайте DataFrame для построения круговой диаграммы
pie_data = pd.concat([top_brands, rest_brands])

# Увеличьте радиус диаграммы в 3 раза
plt.figure(figsize=(9, 9))

# Постройте круговую диаграмму
plt.pie(pie_data, labels=pie_data.index, autopct='%1.1f%%', startangle=140,
colors=plt.cm.tab20.colors)

# Добавьте заголовок
plt.title('Доля моделей по брендам (Top 5 и rest)')

# Сохраните график
plt.savefig('auto_pie.png', bbox_inches='tight')
plt.show()
```

Спасибо за работу! Уверены, вы хорошо постарались. Предлагаем свериться с возможным вари
решения:

```
import matplotlib.pyplot as plt

import pandas as pd

df = pd.read_csv('Electric_Car.csv')

# Количество уникальных моделей машин для каждого бренда
sd =
df.groupby(['Brand'])['Model'].count().sort_values(ascending= False)

#Выбор
Model для значений более 5
sd_pd = sd.to_frame().reset_index()
sd_pd_pie=sd_pd[sd_pd['Model']>5]
sd_pd_pie

#Выбор
Model для значений менее или равно 5
sd_rest_count=sd_pd[sd_pd['Model']<=5].sum()
sd_rest_count.iloc[1]

#Создание
данных для новой строки с оставшимся количеством Model
newDict=
{"Brand":'Rest',"Model":sd_rest_count.iloc[1]}

#Создание
Pandas DataFrame для новой строки
print("New row data is:")
print(newDict)
df1=pd.DataFrame([newDict])

#Добавление новой строки
sd_pd_pie =pd.concat([sd_pd_pie,df1])
sd_pd_pie

#Проверка
содержимого столбца Brand
sd_pd_pie.Brand

#Проверка
содержимого столбца Model
sd_pd_pie.Model

#Назначение
переменным данных для построения диаграммы
labels = sd_pd_pie.Brand
sizes = sd_pd_pie.Model
fig = plt.pie(sizes, labels=labels, autopct='%0.1f%%',
radius = 2);

#Сохранение
диаграммы в файл
plt.savefig('auto_pie.png')
```

Вопрос **2**

Выполнен

Балл: 1,00

Построение столбчатой диаграммы

Постройте горизонтальную столбчатую диаграмму по данным файла **Electric_Cars.csv**, которы
сведения о моделях электромобилей.

Для построения диаграммы:

1. Сформируйте данные по количеству различных моделей (столбец Brand) и выполните сорт
количеству.

2. Используйте библиотеку matplotlib, чтобы построить горизонтальную столбчатую диаграмм
данным из первого пункта.

3. При построении диаграммы укажите наименование модели и соответствующее ей количес

4. Сохраните график под именем saved_figure_barh.png.

Как должно получиться:

Напишите код программы в самостоятельно созданном Python-ноутбуке и прикрепите его в LMS

```
import pandas as pd
import matplotlib.pyplot as plt

# Загрузка данных из CSV-файла
df = pd.read_csv('Electric_Cars.csv')

# Формирование данных по количеству различных моделей
brand_counts = df['Brand'].value_counts()

# Сортировка по количеству
sorted_brands = brand_counts.sort_values(ascending=True)

# Построение горизонтальной столбчатой диаграммы
plt.figure(figsize=(10, 6))
sorted_brands.plot(kind='barh', color='skyblue')
plt.title('Количество моделей по брендам')
plt.xlabel('Количество моделей')
plt.ylabel('Бренд')
plt.grid(axis='x')

# Аннотации с количеством моделей
for index, value in enumerate(sorted_brands):
    plt.text(value, index, str(value))

# Сохранение графика
plt.savefig('saved_figure_barh.png', bbox_inches='tight')
plt.show()
```

Спасибо за работу! Уверены, вы хорошо постарались. Предлагаем свериться с возможным вари
решения:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

```
df = pd.read_csv('Electric_Car.csv')
# Группировка Brand по количеству уникальных моделей
sd = df.groupby(['Brand'])['Model'].count().sort_values(ascending= False)
sd.values
sd.index
```

```
#Построение круговой диаграммы количества моделей
#Определенного бренда
labels = sd.index
y = sd.values
y_pos = np.arange(len(labels))
fig, ax = plt.subplots()
ax.barh(y_pos, y,  align='center')
ax.set_yticks(y_pos, labels=labels)
ax.invert_yaxis()  # labels read top-to-bottom
ax.set_title('Количество моделей определенного бренда')
ax.grid()
plt.show()
```

```
#Сохранение диаграммы в файл
plt.savefig('auto_barh.png')
```

Вопрос **3**

Выполнен

Балл: 1,00

Построение и визуализация 3D-цилиндра

Напишите программу для построения и визуализации 3D-цилиндра с помощью библиотеки matp

Шаги выполнения задания:

1. Сформируйте вершины для построения цилиндра из 20 угловых секторов. Сначала по вершины одного основания (круга) — используйте углы поворота и радиус окружности равн

   Координаты вершин основания определяются по формуле:

   ```
   0,cos(2*pi*(i)/N),sin(2*pi*(i)/N),
   ```

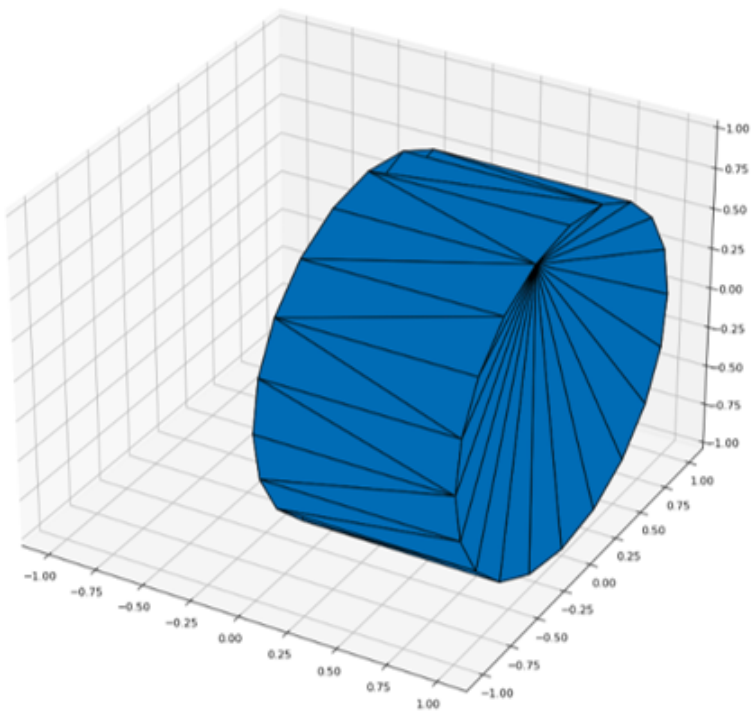
   где выражение (2*pi*(i)/N) задает углы поворота на каждом шаге i.

2. Аналогично постройте вершины второго основания, взяв высоту равную 1. Координаты основания можно найти по формуле:

   ```
   1,cos(2*pi*(i)/N),sin(2*pi*(i)/N)
   ```

3. Используйте библиотеку spatial, чтобы сформировать грани для построения цилиндра угловых секторов.
4. Создайте сетку для построения цилиндра.
5. Визуализируйте полученный цилиндр.
6. Сохраните изображение в формате stl.

Как должно получиться:



```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from scipy.spatial import ConvexHull, convex_hull_plot_2d

def create_cylinder(radius=1, height=1, num_sectors=20):
    # Создаем вершины для первого основания (круга)
    base_vertices = np.array([[0, np.cos(2 * np.pi * i / num_sectors), np.sin(2 * np.p
num_sectors)] for i in range(num_sectors)])

    # Создаем вершины для второго основания (круга)
    top_vertices = np.array([[height, np.cos(2 * np.pi * i / num_sectors), np.sin(2 *
/ num_sectors)] for i in range(num_sectors)])

    # Создаем грани для цилиндра
    cylinder_faces = []
    for i in range(num_sectors):
        next_i = (i + 1) % num_sectors
        # Грани для первого основания
        cylinder_faces.append([i, next_i, num_sectors + i])
        # Грани для второго основания
        cylinder_faces.append([num_sectors + i, next_i + num_sectors, next_i])

    return np.vstack([base_vertices, top_vertices]), np.array(cylinder_faces)

def plot_cylinder(vertices, faces):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    ax.plot_trisurf(vertices[:, 1], vertices[:, 2], faces, vertices[:, 0], shade=True,
color='lightblue')
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
    plt.show()

    # Сохраняем изображение в формате stl
    hull = ConvexHull(vertices[:, 1:])
    plt.figure()
    convex_hull_plot_2d(hull)
    plt.savefig('cylinder.stl', format='stl', bbox_inches='tight')

# Создаем и визуализируем цилиндр
vertices, faces = create_cylinder()
plot_cylinder(vertices, faces)
```

Спасибо за работу! Уверены, вы хорошо постарались. Предлагаем свериться с возможным вари
решения:

```
import numpy as np
from stl import mesh
import matplotlib.pyplot as plt
from matplotlib import pyplot as plt
from mpl_toolkits import mplot3d
from scipy import spatial
```

```
#Функция отображения вершин
def plot_verticles(vertices, isosurf = False):
    #Создание новой графики
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    x = [v[0] for v in vertices]
    y = [v[1] for v in vertices]
    z = [v[2] for v in vertices]
    if isosurf:
        ax.plot_trisurf(x, y, z, linewidth=0.2, antialiased=True)
    else:
        ax.scatter(x, y, z, c='r', marker='o')
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
    #Отображение файла или запись файла
    plt.show()
```

```
#Функция отображения сетки
def plot_mesh(
    your_mesh,
    size_x=10,
    size_y=10,
    dpi=80):
    # Создание нового 3D-отображения
    figure = plt.figure(figsize=(size_x, size_y), dpi=dpi)
    axes = mplot3d.Axes3D(figure)
    axes.add_collection3d(mplot3d.art3d.Poly3DCollection(your_mesh.vectors, edgecolor=
    figure.add_axes(axes)
    #Auto scale к размеру сетки
    scale = your_mesh.points.flatten()
    axes.auto_scale_xyz(scale, scale, scale)
    #Отображение и запись графика
    plt.show()
```

```
#Вершины верхнего основания цилиндра
vertices = np.array([[0,0,0]])
phi = 0
N=20
```

```
#Первый круг
for i in range(0,N):
    vertices_1 = np.array([[0,np.cos(2*np.pi*(i)/N),np.sin(2*np.pi*(i)/N)]])
    vertices =np.append(vertices,vertices_1, axis = 0)

# Второй круг
for i in range(0,N):
    vertices_1 = np.array([[1,np.cos(2*np.pi*(i)/N),np.sin(2*np.pi*(i)/N)]])
    vertices =np.append(vertices,vertices_1, axis = 0)
```

```
#Функция создание граней из вершин
hull = spatial.ConvexHull(vertices)
faces = hull.simplices
#Массив faces содержит описание граней

#Создание сетки
cylinder_mesh = mesh.Mesh(np.zeros(faces.shape[0], dtype=mesh.Mesh.dtype))
for i, f in enumerate(faces):
    for j in range(3):
        cylinder_mesh.vectors[i][j] = vertices[f[j],:]
plot_mesh(cylinder_mesh)
```

```
#Запись сетки в файл "cylinder_mesh.stl"
cylinder_mesh.save('cylinder_mesh.stl')
```