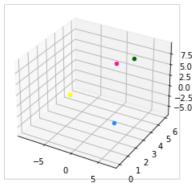
1.3.1 Задание

Задайте 4 точки в трехмерном пространстве, рассчитайте между ними расстояния по описанным в примере выше метрикам (евклидово расстояние, квадрат квклидова расстояния, взвешенное евклидово расстояние,хемингово расстояние, расстояние чебышева). Отобразите точки в трехмерном пространстве.

```
In [15]: import matplotlib.pyplot as plt
                from mpl_toolkits.mplot3d import Axes3D
                from random import randint
                import numpy as np
                fig=plt.figure()
                ax=fig.add_subplot(111,projection='3d')
                coordinates=[randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),randint(-10.10),r
                x=ax.scatter(coordinates[0],coordinates[1],coordinates[2],color="deeppink")# т очки для пло т т ера
                y=ax.scatter(coordinates[3],coordinates[4],coordinates[5],color="dodgerblue")
                z=ax.scatter(coordinates[6],coordinates[7],coordinates[8],color="darkgreen")
                f=ax.scatter(coordinates[9],coordinates[10],coordinates[11],color="yellow")
                point_1=np.array((coordinates[0],coordinates[1],coordinates[2]))#точки для расстояний
                point 2=np.array((coordinates[3],coordinates[4],coordinates[5]))
                point_3=np.array((coordinates[6],coordinates[7],coordinates[8]))
                point_4=np.array((coordinates[9],coordinates[10],coordinates[11]))
                print("Розовая точка: x=",coordinates[0],", y=",coordinates[1],", z=",coordinates[2])
               print("Голубая точка: x=",coordinates[3],", y=",coordinates[4],", z=",coordinates[5]) print("Зеленая точка: x=",coordinates[6],", y=",coordinates[7],", z=",coordinates[8]) print("Желтая точка: x=",coordinates[9],", y=",coordinates[10],", z=",coordinates[11])
                plt.show()
                print("Евклидово расстояние: \nмежду розовой и голубой точками",np.linalg.norm(point 1-point 2),"\nмежду розовой и зеленой точками",np.lina
                print("Квадрат Евклидова расстояния: \пмежду розовой и голубой точками",np.linalg.norm(point_1-point_2)**2,"\пмежду розовой и зеленой точк
                print("Расстояние Чебышева: \nмежду розовой и голубой точками",np.linalg.norm(point_1-point_2,ord=np.inf),"\nмежду розовой и зеленой точка
                print("Расстояние Чебышева: \пмежду розовой и голубой точками", np.linalg.norm(point 1-point 2,ord=1), "\nмежду розовой и зеленой точками",
Розовая точка: x=3, y=3, z=9
Голубая точка: x= 7 , y= 0 , z= 2
3еленая точка: x = 2 , y = 6 , z = 5
Желтая точка: x=-9 , y=5 , z=-6
```



Евклидово расстояние:

между розовой и зеленой точками 5.0990195135927845 между розовой и желтой точкой 19.313207915827967 между голубой и зеленой точкой 8.366600265340756 между зеленой и желтой точкой 15.588457268119896 между голубой и желтой точкой 18.57417562100671 Квадрат Евклидова расстояния: между розовой и голубой точками 74.0 между розовой и зеленой точками 25.9999999999996 между розовой и желтой точками 373.0000000000006 между голубой и зеленой точкой 70.0 между зеленой и желтой точкой 243.0 между голубой и желтой точкой 345.0 Расстояние Чебышева: между розовой и голубой точками 7.0 между розовой и зеленой точками 4.0 между розовой и желтой точкой 15.0

между розовой и голубой точками 8.602325267042627

между зеленой и желтой точкой 11.0 между голубой и желтой точкой 16.0 Расстояние Чебышева: между розовой и голубой точками 14.0 между розовой и зеленой точками 8.0

между голубой и зеленой точкой 6.0

между розовой и желтой точкой 29.0 между голубой и зеленой точкой 14.0 между зеленой и желтой точкой 23.0

между голубой и желтой точкой 29.0

1.3.2 Задание

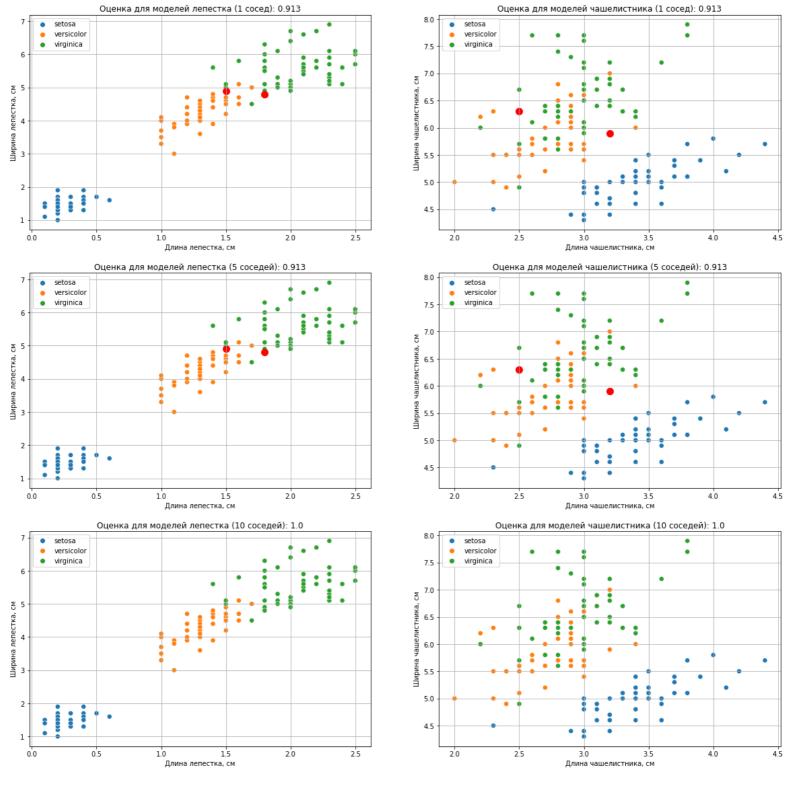
```
Создать 5х5 матрицу со значениями в строках от 0 до 4. Для создания необходимо использовать функцию arange In [17]: import numpy as np Z=np.zeros((5,5)) Z+=np.arange(5) print(Z)

[[0. 1. 2. 3. 4.]
[0. 1. 2. 3. 4.]
[0. 1. 2. 3. 4.]
[0. 1. 2. 3. 4.]
[0. 1. 2. 3. 4.]
[0. 1. 2. 3. 4.]
```

Для предыдущего примера поэксперементируйте с параметрами классификатора:

```
1. Установите другое количество ближайщих соседей (к=1,5,10)
    2. Установите размер тестовой выборки 15% от всего датасета.
    3.Постройте графики и оцените качество моделей, проанализируйте результаты
In [44]: from sklearn.model_selection import train_test_split
       from sklearn.neighbors import KNeighborsClassifier
       import seaborn as sns
       import matplotlib.pyplot as plt
       import numpy as np
       k=[1,5,10]
       iris=sns.load_dataset('iris')
       X_train,X_test,y_train,y_test=train_test_split(iris.iloc[:,:-1],iris.iloc[:,-1],test_size=0.15)
       X train.shape, X test.shape, y train.shape, y test.shape
       X train.head()
       y_train.head()
       fig=plt.figure()
       model=KNeighborsClassifier(n neighbors=k[0])#1 сосед
       model.fit(X_train,y_train)
       y_pred=model.predict(X_test)
       ax1=fig.add_subplot(3,2,1)#добавили график к фигуре
       sns.scatterplot(x='petal_width',y='petal_length',data=iris,hue='species',s=50)#обозначили точки
       plt.xlabel('Длина лепестка, см')#дали имена осям
       plt.ylabel('Ширина лепестка, см')
       plt.legend(loc=2)#сделали легенду графика
       plt.grid()#сделали график в сетку
       for i in range(len(y_test)):
          if np.array(y test)[i]!=y pred[i]:
            plt.scatter(X_test.iloc[i,3],X_test.iloc[i,2],color='red',s=100)
       from sklearn.metrics import accuracy_score
       number=str(round(accuracy score(y test,y pred),3))#вычисляем оценку
       title_in_string="Оценка для моделей лепестка (1 сосед): "+number
       ax1.title.set text(title in string)
       model=KNeighborsClassifier(n_neighbors=k[0])
       model.fit(X train,y train)
       y_pred=model.predict(X_test)
       ax2=fig.add_subplot(3,2,2)
       sns.scatterplot(x='sepal width',y='sepal length',data=iris,hue='species',s=50)
       plt.xlabel('Длина чашелистника, см')
       plt.ylabel('Ширина чашелистника, см')
       plt.legend(loc=2)
       plt.grid()
       for i in range(len(y_test)):
          if np.array(y_test)[i]!=y_pred[i]:
            plt.scatter(X_test.iloc[i,1],X_test.iloc[i,0],color='red',s=100)
       from sklearn.metrics import accuracy_score
       number=str(round(accuracy_score(y_test,y_pred),3))
       title_in_string="Оценка для моделей чашелистника (1 сосед): "+number
       ax2.title.set_text(title_in_string)
       model=KNeighborsClassifier(n_neighbors=k[1])#5 сосед
       model.fit(X_train,y_train)
       y_pred=model.predict(X_test)
       ax3=fig.add_subplot(3,2,3)
       sns.scatterplot(x='petal_width',y='petal_length',data=iris,hue='species',s=50)
       plt.xlabel('Длина лепестка, см')
       plt.ylabel('Ширина лепестка, см')
```

```
plt.legend(loc=2)
plt.grid()
for i in range(len(y_test)):
  if np.array(y_test)[i]!=y_pred[i]:
     plt.scatter(X_test.iloc[i,3],X_test.iloc[i,2],color='red',s=100)
from sklearn.metrics import accuracy_score
number=str(round(accuracy_score(y_test,y_pred),3))
title_in_string="Оценка для моделей лепестка (5 соседей): "+number
ax3.title.set_text(title_in_string)
model=KNeighborsClassifier(n_neighbors=k[1])
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
ax4=fig.add_subplot(3,2,4)
sns.scatterplot(x='sepal_width',y='sepal_length',data=iris,hue='species',s=50)
plt.xlabel('Длина чашелистника, см')
plt.ylabel('Ширина чашелистника, см')
plt.legend(loc=2)
plt.grid()
for i in range(len(y_test)):
  if np.array(y_test)[i]!=y_pred[i]:
     plt.scatter(X test.iloc[i,1],X test.iloc[i,0],color='red',s=100)
from sklearn.metrics import accuracy_score
number=str(round(accuracy_score(y_test,y_pred),3))
title_in_string="Оценка для моделей чашелистника (5 соседей): "+number
ax4.title.set_text(title_in_string)
model=KNeighborsClassifier(n_neighbors=k[2])#10 соседей
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
ax5=fig.add_subplot(3,2,5)
sns.scatterplot(x='petal_width',y='petal_length',data=iris,hue='species',s=50)
plt.xlabel('Длина лепестка, см')
plt.ylabel('Ширина лепестка, см')
plt.legend(loc=2)
plt.grid()
for i in range(len(y_test)):
  if np.array(y_test)[i]!=y_pred[i]:
     plt.scatter(X_test.iloc[i,3],X_test.iloc[i,2],color='red',s=100)
from sklearn.metrics import accuracy_score
number=str(round(accuracy_score(y_test,y_pred),3))
title_in_string="Оценка для моделей лепестка (10 соседей): "+number
ax5.title.set_text(title_in_string)
model=KNeighborsClassifier(n_neighbors=k[2])
model.fit(X_train,y_train)
y_pred=model.predict(X test)
ax6=fig.add subplot(3,2,6)
sns.scatterplot(x='sepal_width',y='sepal_length',data=iris,hue='species',s=50)
plt.xlabel('Длина чашелистника, см')
plt.ylabel('Ширина чашелистника, см')
plt.legend(loc=2)
plt.grid()
for i in range(len(y_test)):
  if np.array(y_test)[i]!=y_pred[i]:
     plt.scatter(X_test.iloc[i,1],X_test.iloc[i,0],color='red',s=100)
from sklearn.metrics import accuracy_score
number=str(round(accuracy_score(y_test,y_pred),3))
title_in_string="Оценка для моделей чашелистника (10 соседей): "+number
ax6.title.set_text(title_in_string)
```



3.3.2 Задание

[0., 2., 0., 0.]])

Опредлите набор признаков человека, по аналогии рабочей тетради 1,-например, цвет глаз и конвертируйте его в матрицу признаков