

Путь к файлу

from google.colab import files

uploaded = files.upload()

Параметры тренировки

Установка библиотеки fitcode

!pip install fitdecode

Чтение файла

f1 = open ('/content/2\_5303101742206956504.fit','r')

Структура записей

record = ['latitude', 'longitude','altitude','timestamp','heart\_rate','cadence','speed','distance']

Подключение пакетов

import fitdecode

import pandas as pd

import os

from datetime import datetime, timedelta

from typing import Dict,Union,Optional,NamedTuple

Функция, извлекающая данные о координатах и преобразующая их в десятичные градусы

def get\_fit\_point\_data(frame: fitdecode.records.FitDataMessage) -> Optional[Dict[str, Union[float, int, str, datetime]]]:

  data: Dict[str, Union[float, int, str, datetime]] = {}

  if not (frame.has\_field('position\_lat') and frame.has\_field('position\_long')):

    return None

  else:

        lat\_=frame.get\_value('position\_lat')

        long\_=frame.get\_value('position\_long')

        data['latitude']=lat\_

        data['longitude'] = long\_

        if (lat\_ is not None) and (long\_ is not None):

          data['latitude'] = float(lat\_) / ((2\*\*32) / 360)

          data['longitude'] = float(long\_) / ((2\*\*32) / 360)

  for field in record[2: ]:

    if frame.has\_field(field):

      data[field] = frame.get\_value(field)

  return data

функция, которая извлекает данные о параметрах тренировки

def get\_fit\_other\_data(col, frame: fitdecode.records.FitDataMessage) -> Optional[Dict[str, Union[float, int, str, datetime]]]:

  data: Dict[str, Union[float, int, str, datetime]] = {}

  for field in col:

    if frame.has\_field(field):

      data[field] = frame.get\_value(field)

  return data

Чтение файла и вывод функции

record\_data = []

with fitdecode.FitReader('/content/2\_5303101742206956504.fit') as fit\_file:

  for frame in fit\_file:

    if isinstance(frame, fitdecode.records.FitDataMessage):

      if frame.name == 'record':

        single\_point\_data = get\_fit\_point\_data(frame)

        if single\_point\_data is not None:

          record\_data.append(single\_point\_data)

record\_df = pd.DataFrame(record\_data, columns=record)

record\_df = record\_df.dropna().reset\_index()

record\_df

Подключение пакетов для обработки геометрии

from pyproj import CRS

import geopandas as gpd

from shapely.geometry import LineString

Получаем границы экстента

line = gpd.GeoDataFrame (crs = CRS('EPSG:4326'), geometry=[LineString(zip(record\_df.longitude, record\_df.latitude))])

line\_centroid = line.centroid

line\_bounds = line.bounds

Показываем на карте openstreetmap

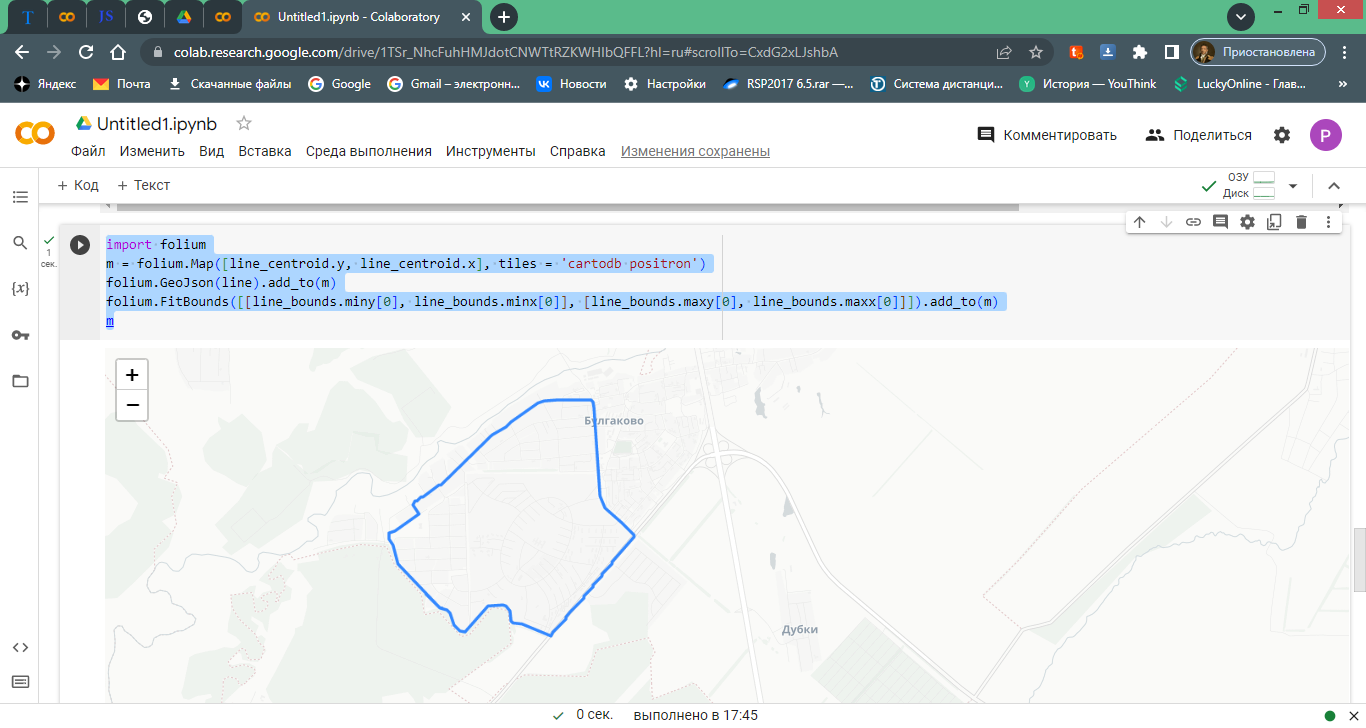
import folium

m = folium.Map([line\_centroid.y, line\_centroid.x], tiles = 'openstreetmap ')

folium.GeoJson(line).add\_to(m)

folium.FitBounds([[line\_bounds.miny[0], line\_bounds.minx[0]], [line\_bounds.maxy[0], line\_bounds.maxx[0]]]).add\_to(m)

m



**Показываем на карте длину маршрута**

import folium

end = record\_df.shape[0]

dist = record\_df['distance'][end - 1]

dist = str(dist)

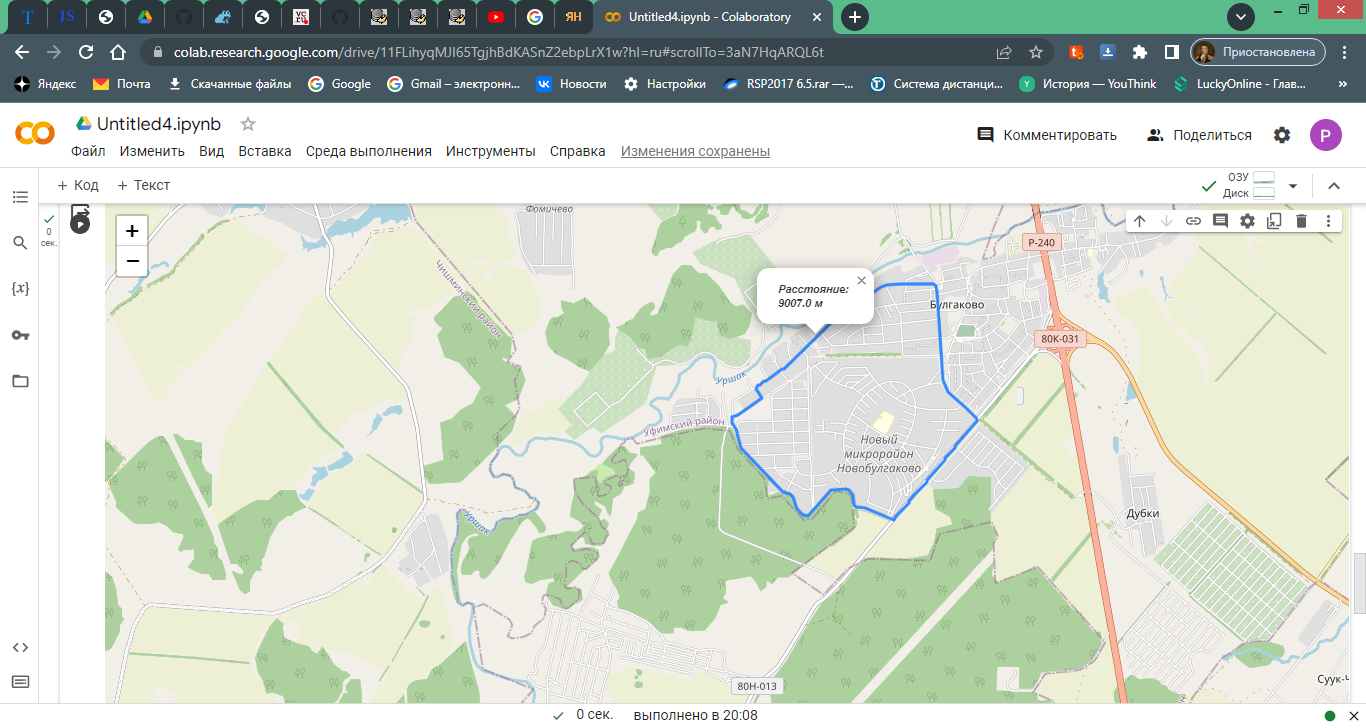
m = folium.Map([line\_centroid.y, line\_centroid.x], tiles = openstreetmap ')

lin = folium.GeoJson(line).add\_to(m)

folium.FitBounds([[line\_bounds.miny[0], line\_bounds.minx[0]], [line\_bounds.maxy[0], line\_bounds.maxx[0]]]).add\_to(m)

folium.Popup('<i><b>Расстояние:<b><i> '+dist+' <i>м<i>').add\_to(lin)

m



Строим график

import altair as alt

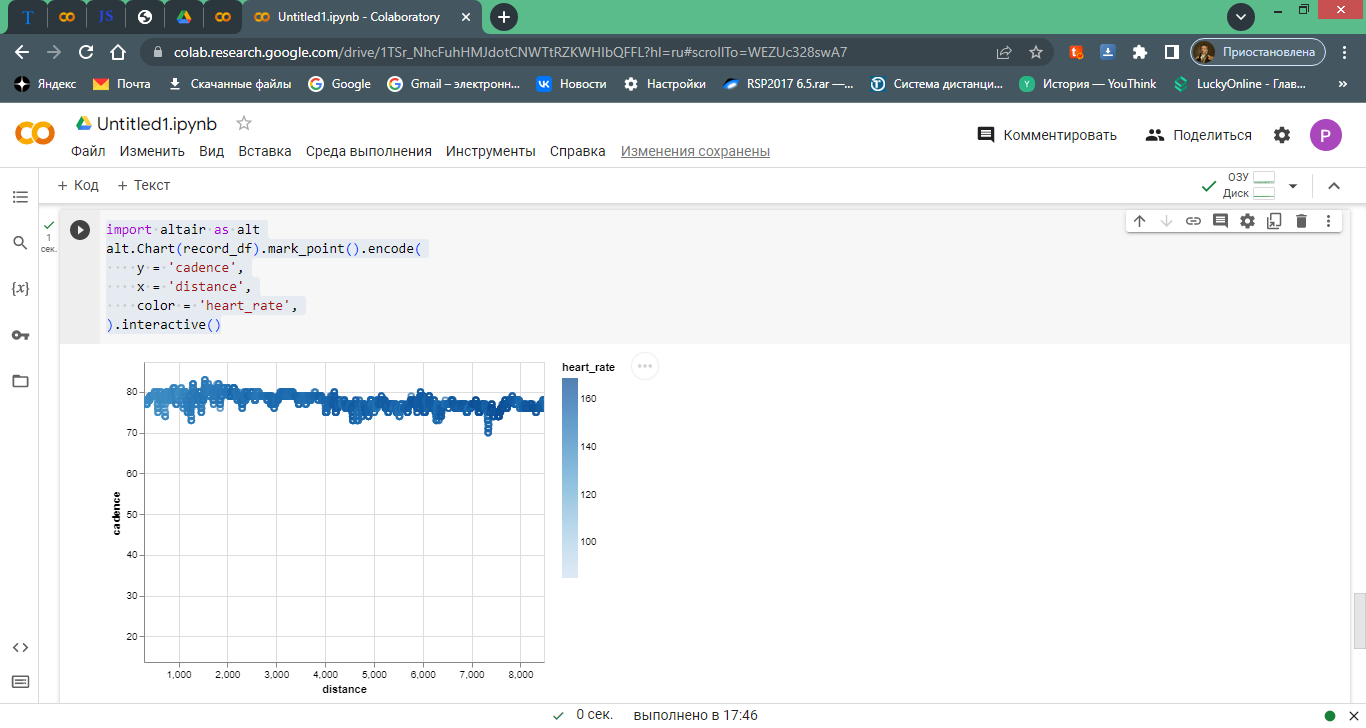
alt.Chart(record\_df).mark\_point().encode(

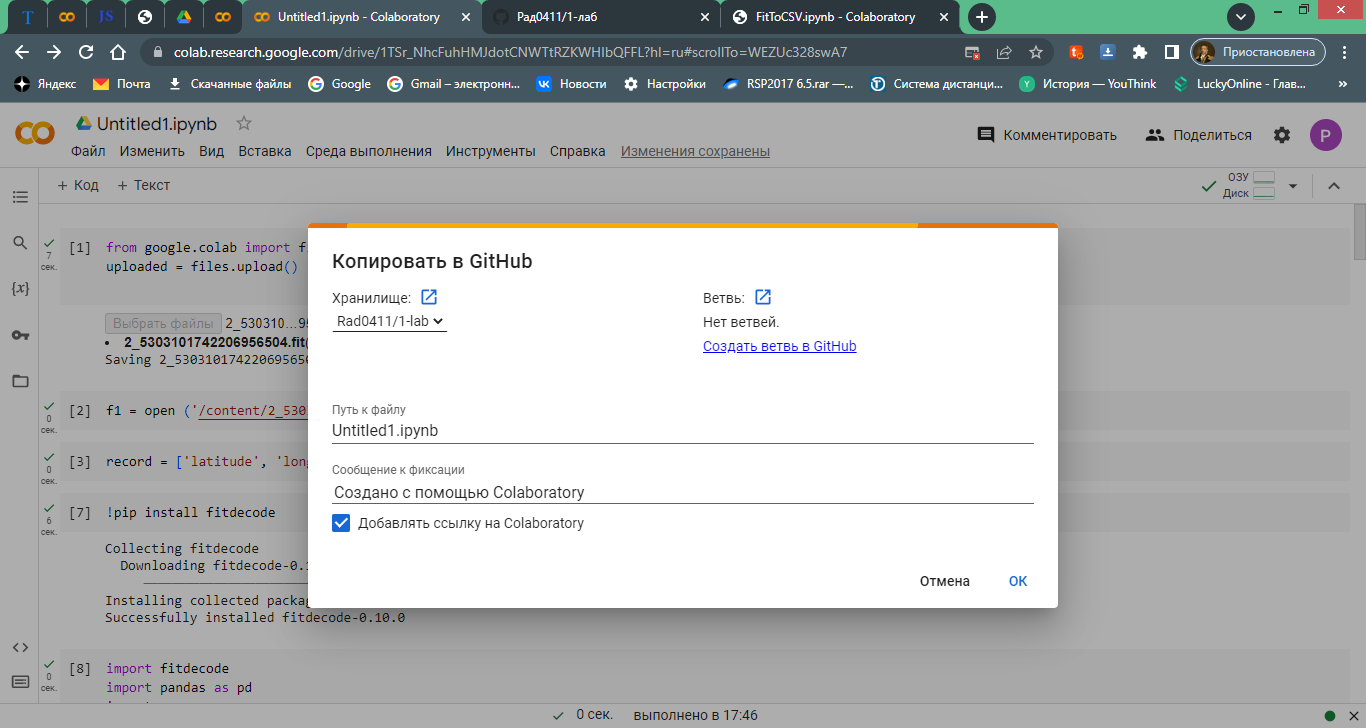
    y = 'cadence',

    x = 'distance',

    color = 'heart\_rate',

).interactive()





**Грфик 2**

import altair as alt

alt.Chart(record\_df).mark\_tick(

    filled = True

).encode(

    y = 'speed',

    x = 'distance',

    tooltip = 'heart\_rate'

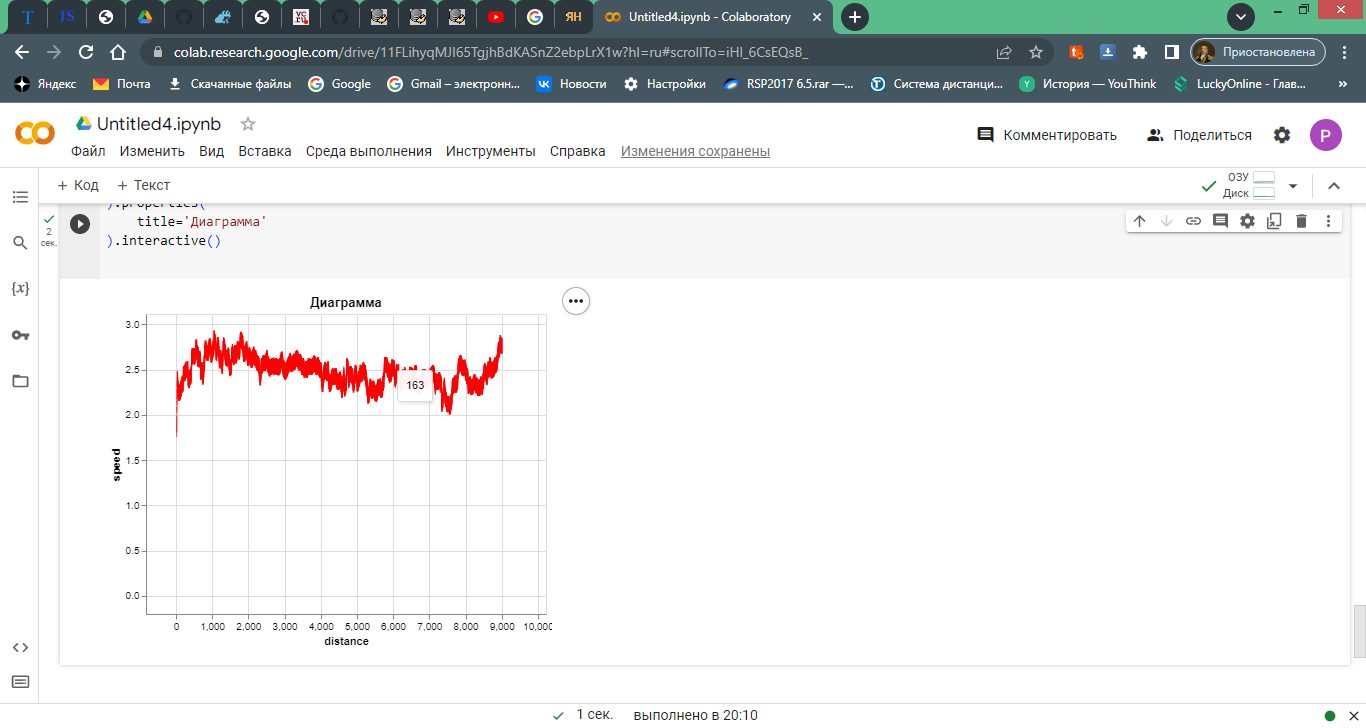
).configure\_mark(

    color='red',  # Изменяем цвет диаграммы

).properties(

    title='Диаграмма'

).interactive()



line1=line.to\_crs(3395)

line\_len = line1.length

line\_len