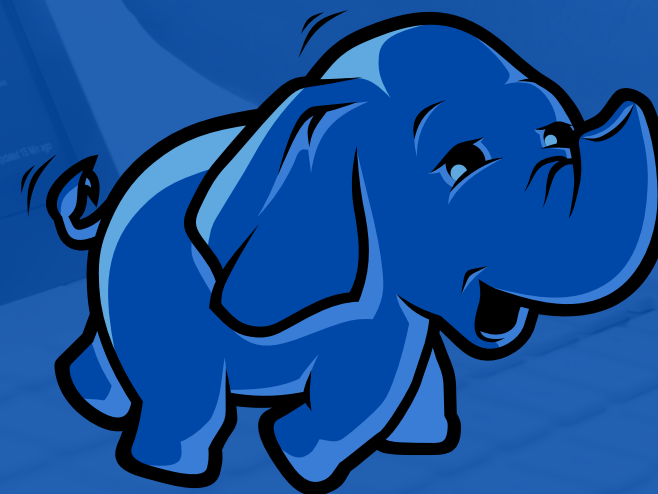


# Први пројекат Denver Mobility



Електронски факултет у Нишу  
Вештачка интелигенција и машинско учење

Професор: **Др. Драган Х. Стојановић**  
Студент: **Петковић Петар**

# Садржај

01

Генерисање  
података

02

Пречишћавање  
података



03

Подешавање  
окружења

04

Апликација



# Генерисање података

Подаци су генерисани путем симулатора урбаног саобраћаја SUMO. Изабран је град Денвер у држави Колорадо. Иницијални скуп података има информације о више од 20.000 возила за временски период од скоро 3 часа. Такође, сваки запис садржи 14 атрибута. Величина фајла је око 1.5gb

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	timestep	id	x	y	angle	type	speed	pos	lane	slope	signals	acceleration	distance	odometer	posLat
2	0	bus0	-104.996667	39.73211	270.14	bus_bus	0	12.1	336167361#0_0	0	0	0	12.1	0	0
3	0	veh0	-104.990763	39.72572	90.39	veh_passenger	0	5.1	427704888#4_4	0	0	0	5.1	0	0
4	1	bus0	-104.996674	39.73211	270.12	bus_bus	0.62	12.72	336167361#0_0	0	0	0.62	12.72	0.62	0
5	1	veh0	-104.990736	39.72572	90.39	veh_passenger	2.34	7.44	427704888#4_4	0	2	2.34	7.44	2.34	0
6	1	veh1	-104.990296	39.72956	328.88	veh_passenger	0	5.1	1121174297_1	0	0	0	5.1	0	0
7	1	veh2	-104.960328	39.72731	269.85	veh_passenger	0	5.1	586800346#3_0	0	0	0	5.1	0	0
8	2	bus0	-104.996692	39.73211	270.1	bus_bus	1.57	14.29	336167361#0_0	0	0	0.95	14.29	2.19	0
9	2	bus1	-105.02491	39.74756	89.86	bus_bus	0	12.1	-628592111#3_0	0	0	0	12.1	0	0
10	2	veh0	-104.990692	39.72572	90.39	veh_passenger	3.78	11.22	427704888#4_4	0	2	1.44	11.22	6.12	0
11	2	veh1	-104.990303	39.72957	329.73	veh_passenger	1.38	6.48	1121174297_1	0	0	1.38	6.48	1.38	0
12	2	veh2	-104.960351	39.72731	269.85	veh_passenger	1.98	7.08	586800346#3_0	0	0	1.98	7.08	1.98	0
13	2	veh3	-105.011973	39.74027	91.65	veh_passenger	0	5.1	628707473_2	0	0	0	5.1	0	0
14	2	veh4	-104.97575	39.7496	0.01	veh_passenger	0	5.1	16985352#3_0	0	0	0	5.1	0	0
15	3	bus0	-104.996719	39.73211	270.1	bus_bus	2.31	16.59	336167361#0_0	0	0	0.74	16.59	4.49	0



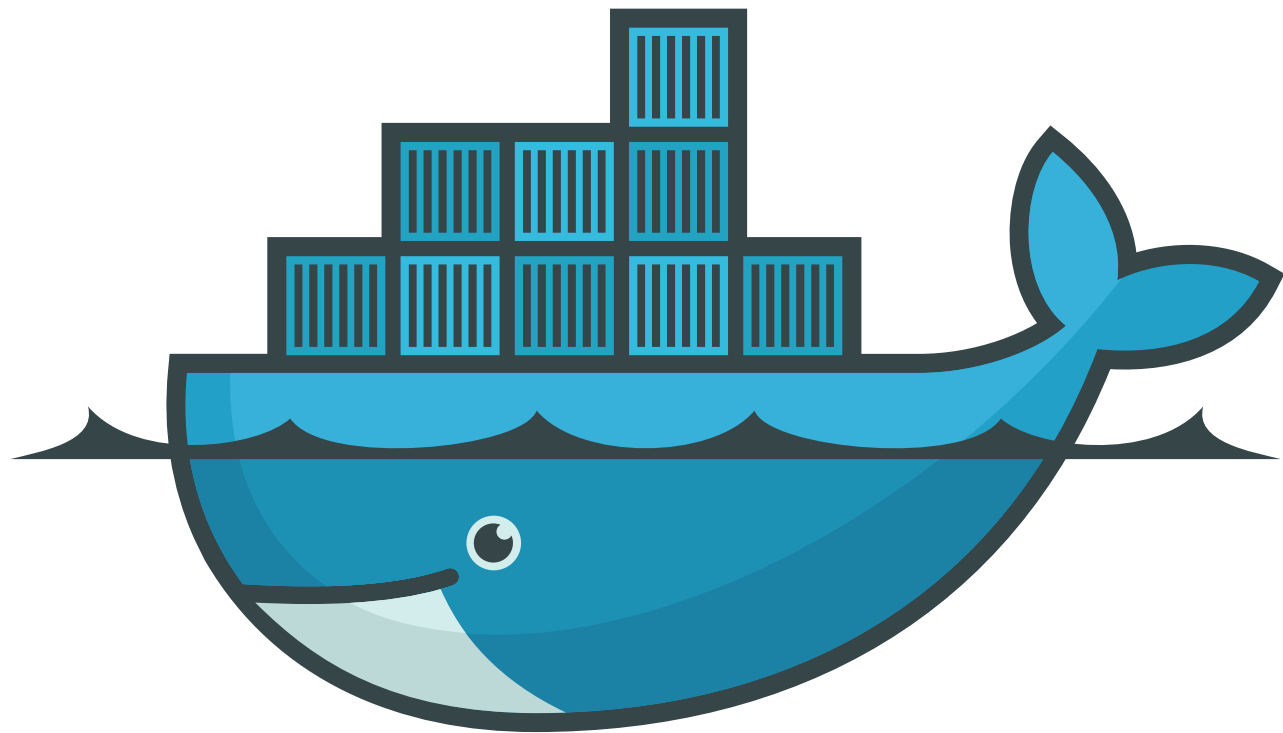
# Пречишћавање података

Генерисан скуп података није био у најбољем формату за анализу па је неопходна била трансформација у погоднији облик.

- 01 Timestep замењен Timestamp-ом.
- 02 Конверзија релативних координати **x** и **y** у географску дужину и ширину
- 03 Конверзија mph у kmh
- 04 Промена назива атрибута

1	timestamp	id	type	latitude	longitude	speed_kmh	acceleration	distance	odometer	pos
2	2023-02-11T08:00:00.000+01:00	bus0	bus	39.732112	-104.996667	0	0	12.1	0	12.1
3	2023-02-11T08:00:00.000+01:00	veh0	car	39.725715	-104.990763	0	0	5.1	0	5.1
4	2023-02-11T08:00:01.000+01:00	bus0	bus	39.732112	-104.996674	2.23	0.62	12.72	0.62	12.72
5	2023-02-11T08:00:01.000+01:00	veh0	car	39.725715	-104.990736	8.42	2.34	7.44	2.34	7.44
6	2023-02-11T08:00:01.000+01:00	veh1	car	39.729562	-104.990296	0	0	5.1	0	5.1
7	2023-02-11T08:00:01.000+01:00	veh2	car	39.727314	-104.960328	0	0	5.1	0	5.1
8	2023-02-11T08:00:02.000+01:00	bus0	bus	39.732112	-104.996692	5.65	0.95	14.29	2.19	14.29
9	2023-02-11T08:00:02.000+01:00	bus1	bus	39.74756	-105.02491	0	0	12.1	0	12.1
10	2023-02-11T08:00:02.000+01:00	veh0	car	39.725715	-104.990692	13.61	1.44	11.22	6.12	11.22
11	2023-02-11T08:00:02.000+01:00	veh1	car	39.729573	-104.990303	4.97	1.38	6.48	1.38	6.48
12	2023-02-11T08:00:02.000+01:00	veh2	car	39.727314	-104.960351	7.13	1.98	7.08	1.98	7.08
13	2023-02-11T08:00:02.000+01:00	veh3	car	39.74027	-105.011973	0	0	5.1	0	5.1
14	2023-02-11T08:00:02.000+01:00	veh4	car	39.749598	-104.97575	0	0	5.1	0	5.1
15	2023-02-11T08:00:03.000+01:00	bus0	bus	39.732112	-104.996719	8.32	0.74	16.59	4.49	16.59

# Подешавање окружења



docker

Кластер контејнера у мрежи BDE - Docker-Compose

Учитавање података на Hadoop

Покретање апликације

Рад апликације































# Кластер контейнера - Мрежа BDE

```
services:
  spark-master:
    image: bde2020/spark-master:3.1.2-hadoop3.2
    container_name: spark-master
    ports:
      - "8070:8070"
      - "7077:7077"
    environment:
      - INIT_DAEMON_STEP=setup_spark
      - SPARK_MASTER_PORT=7077
      - SPARK_MASTER_WEBUI_PORT=8070

  spark-worker-1: <5 keys>

  spark-worker-2: <5 keys>

  namenode:
    image: bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8
    container_name: namenode
    restart: always
    ports:
      - 9870:9870
      - 9000:9000
    volumes:
      - hadoop_namenode:/hadoop/dfs/name
      - $HOME/data:/data
    environment:
      - CLUSTER_NAME=test
    env_file:
      - ./hadoop.env
```



<input type="checkbox"/>	Name	Image	Status	Port(s)	Last started	Actions
<input type="checkbox"/>	 <a href="#">project1</a>		Running (8/8)		0 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">namenode</a> 2ba062ba6368 	<a href="#">bde2020/hadoop-namenode</a>	Running	<a href="#">9000:9000</a>  <a href="#">Show all ports (2)</a>	37 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">datanode</a> 363332c0b30f 	<a href="#">bde2020/hadoop-datanode</a>	Running		37 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">nodemanager</a> 7d0f942a608a 	<a href="#">bde2020/hadoop-nodemanager</a>	Running		37 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">resourcemanager</a> 8146b2901c6b 	<a href="#">bde2020/hadoop-resourcemanager</a>	Running		0 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">historyserver</a> 6cdb54fde1c9 	<a href="#">bde2020/hadoop-historyserver</a>	Running		36 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">spark-master</a> 53f1e2beb56c 	<a href="#">bde2020/spark-master:3</a>	Running	<a href="#">7077:7077</a>  <a href="#">Show all ports (2)</a>	35 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">spark-worker-1</a> 203e1a579de9 	<a href="#">bde2020/spark-worker:3</a>	Running	<a href="#">8071:8071</a> 	31 seconds ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 <a href="#">spark-worker-2</a> c2264fdf2ce3 	<a href="#">bde2020/spark-worker:3</a>	Running	<a href="#">8072:8071</a> 	30 seconds ago	<input type="checkbox"/> ⋮ 

# Учитавање података на HDFS

## Browse Directory

/dir

Show 25 entries

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
<input type="checkbox"/>	<a href="#">-rw-r--r--</a>	<a href="#">root</a>	<a href="#">supergroup</a>	11.55 KB	May 16 00:08	<a href="#">3</a>	128 MB	<a href="#">denverMobility.py</a>	
<input type="checkbox"/>	<a href="#">-rw-r--r--</a>	<a href="#">root</a>	<a href="#">supergroup</a>	1.14 GB	May 16 00:08	<a href="#">3</a>	128 MB	<a href="#">denverVehiclesCleaned.csv</a>	

Showing 1 to 2 of 2 entries

Previous 1 Next

```
@echo off
docker cp denverMobility.py namenode:/data
docker cp data/denverVehiclesCleaned.csv namenode:/data
docker exec -it namenode bash -c "hdfs dfs -mkdir /dir"
docker exec -it namenode bash -c "hdfs dfs -rm -r /dir/denverMobility.py"
docker exec -it namenode bash -c "hdfs dfs -rm -r /dir/denverVehiclesCleaned.csv"
docker exec -it namenode bash -c "hdfs dfs -put /data/denverMobility.py /dir"
docker exec -it namenode bash -c "hdfs dfs -put /data/denverVehiclesCleaned.csv /dir"
```

# Покретање апликације

## 01 Мануелно покретање апликације

```
1. Manually - Start spark container, connect to the bde network, copy application files
(*.py), install requirements-dependencies (if any)
and execute spark-submit commands with appropriate parameters (for execution on a PC/
laptop --executor-memory and --executor-cores are not needed)
docker run -it --network bde --env-file hadoop.env -p 4040:4040 --name spark bde2020/spa
rk-base:3.1.2-hadoop3.2 bash

/spark/bin/spark-submit --master spark://spark-master:7077 --executor-memory 4G --execut
or-cores 4 hdfs://namenode:9000/dir/denverMobility.py "2023-02-11 09:00:00" "2023-02-11
10:00:00" |
```

## 02 Коришћењем Spark template-а

```
FROM bde2020/spark-python-template:3.1.2-hadoop3.2
COPY denverMobility.py /app/

ENV SPARK_MASTER spark://spark-master:7077
ENV SPARK_APPLICATION_PYTHON_LOCATION /app/denverMobility.py
ENV SPARK_APPLICATION_ARGS "2023-02-11 09:30:00 2023-02-11 10:30:00"
```

```
docker build --rm -t bde/spark-app .
docker run --name denverMobility --net bde -p 4040:4040 -d bde/spark-app
```



# Апликација

Апликација је кодирана у Python програмском језику коришћењем PySpark програмског оквира у PyCharm програмском окружењу. Као што је доле наведено, за различит број параметара који се проследи путем командне линије добију се различите информације. Омогућено је филтрирање како по временском тако по географском простору. Могуће је спојити ова два простора и са типом возила а такође и спојити ова два простора као и тип возила. Након филтрирања података на основу аргумената, могућ је приказ статистичких параметара за филтрирани скуп, као и број возила који задовољава одређени критеријум и приказ возила чија је вредност специфициране колоне већа од задате вредности.



Информације у  
одређеном  
временском  
периоду



Информације на  
одређеном  
географском  
простору



Информације о типу  
возила за  
географски и/или  
временски простор



Статистички  
параметри о  
возилима за  
претходно наведе  
случајеве



Филтрирање и  
приказ возила са  
вредностима изнад  
граничне вредности

# Приказ дела кода за иницијализацију

```
def initialization():  
    # Define Spark Configuration  
    conf = SparkConf()  
    conf.setMaster(SPARK_MASTER)  
    spark_session = SparkSession.builder.config(conf=conf).appName(APP_NAME).getOrCreate()  
  
    # Set the log level to ERROR to reduce the amount of output  
    spark_session.sparkContext.setLogLevel("ERROR")  
    data_frame = spark_session.read.csv(DATA_PATH, header=True, inferSchema=True)  
  
    return spark_session, data_frame
```

# Приказ дела кода за парсирање аргумената апликације

```
if len(sys.argv) == 5:
    if "-" in sys.argv[1]:
        first_datetime = sys.argv[1] + " " + sys.argv[2]
        second_datetime = sys.argv[3] + " " + sys.argv[4]
        filtered_df = filter_vehicles_in_timespan(df, first_datetime, second_datetime)
    else:
        filtered_df = filter_vehicles_in_location(df, sys.argv[1], sys.argv[2], sys.argv[3], sys.argv[4])
elif len(sys.argv) == 6:
    if "-" in sys.argv[2]:
        first_datetime = sys.argv[2] + " " + sys.argv[3]
        second_datetime = sys.argv[4] + " " + sys.argv[5]
        filtered_df = filter_vehicles_by_type_in_timespan(df, sys.argv[1], first_datetime, second_datetime)
    else:
        filtered_df = filter_vehicles_by_type_in_location(df, sys.argv[1], sys.argv[2],
                                                         sys.argv[3], sys.argv[4], sys.argv[5])
elif len(sys.argv) == 10:
    first_datetime = sys.argv[2] + " " + sys.argv[3]
    second_datetime = sys.argv[4] + " " + sys.argv[5]
    filtered_df = filter_vehicles_by_type_in_timespan_and_location(df, sys.argv[1], first_datetime, second_datetime,
                                                                    sys.argv[4], sys.argv[5], sys.argv[6], sys.argv[7])
```

# Приказ дела кода са функцијама

1 usage 👤 Petar

```
def filter_vehicles_in_location(data_frame, latitude_1, longitude_1, latitude_2, longitude_2):  
    return data_frame.filter((col("latitude").between(latitude_1, latitude_2)) & (col("longitude").between(longitude_1, longitude_2)))
```

1 usage 👤 Petar

```
def filter_vehicles_by_type_in_location(data_frame, vehicle_type, latitude_1, longitude_1, latitude_2, longitude_2):  
    return data_frame.filter((col("type") == vehicle_type) &  
                             (col("latitude").between(latitude_1, latitude_2)) & (col("longitude").between(longitude_1, longitude_2)))
```

1 usage 👤 Petar

```
def filter_vehicles_by_type_in_timespan_and_location(data_frame, vehicle_type, start_time, end_time,  
                                                    latitude_1, longitude_1, latitude_2, longitude_2):  
    return data_frame.filter((col("type") == vehicle_type) &  
                             (col("timestamp").between(start_time, end_time)) &  
                             (col("latitude").between(latitude_1, latitude_2)) & (col("longitude").between(longitude_1, longitude_2)))
```

# Приказ main дела

```
# Main entry point of the application
if __name__ == '__main__':
    # Check the number of command-line arguments
    if len(sys.argv) < 3 | len(sys.argv) > 8:
        print("Usage: main.py <input folder> ")
        exit(-1)

    # Initialize Spark session and DataFrame
    spark, df = initialization()

    # Get the command-line arguments
    args = sys.argv

    if len(sys.argv) == 5:...
    elif len(sys.argv) == 6:...
    elif len(sys.argv) == 10:...

    # Calculate the statistics for the filtered DataFrame
    calculated_statistics = calculate_statistics(filtered_df, STATISTIC_CRITERIA)

    # Print the statistics
    print_statistics(calculated_statistics)
    print_step("Printing vehicles above threshold")
    print("Number of vehicles above threshold : " + str(count_vehicles_above_threshold(filtered_df, "speed_kmh", 50.00)))
    print_vehicles_above_threshold(df, "speed_kmh", 80.00)

    # Stop the Spark session
    spark.stop()
```



# Приказ резултата у конзоли

```
/app/denverMobility.py
2023-02-11
09:30:00
2023-02-11
10:30:00
*****
----- Printing statistics -----
*****
Mean: 27.10140220190724
Max: 122.47
Min: 0.0
Standard Deviation: 24.838097070931937
*****
----- Printing vehicles above threshold -----
*****
Number of vehicles above threshold :3877
```



# Хвала на пажњи