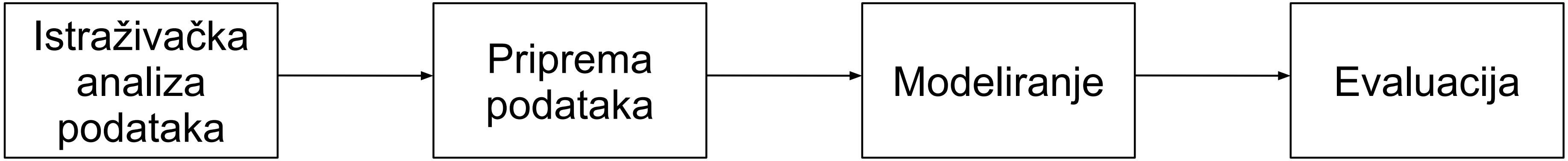
Hadoop implementacija algoritama za klasifikaciju podataka

Miloš Veljković 1174

Algoritmi za klasifikaciju

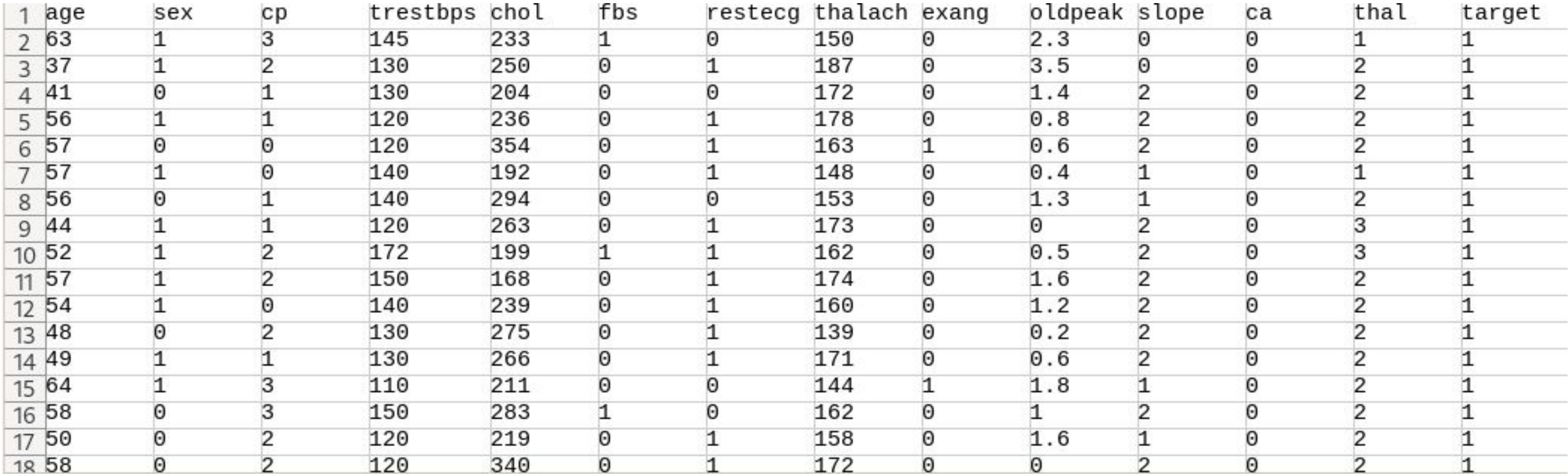
- KNN (K-nearest neighbors)
- Naive-Bayes



Faze klasifikacije

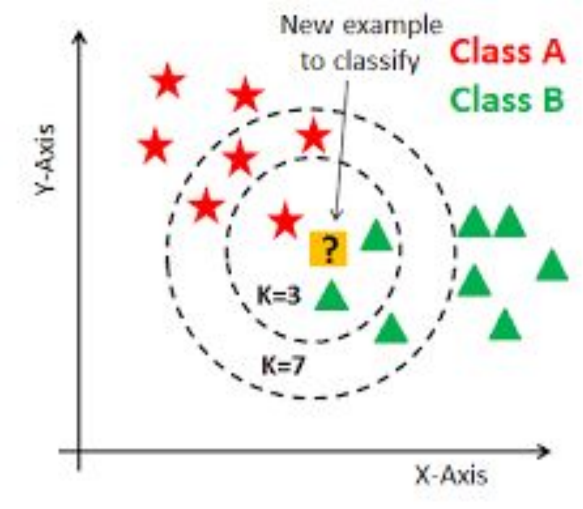
Koji problem rešavamo?

- Određivanje da li osoba ima srčanu bolest ili ne !?



1	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
2	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
11	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
12	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
13	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
14	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
15	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
16	58	0	3	150	283	1	0	162	0	1	2	0	2	1
17	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
18	58	0	2	120	340	0	1	172	0	0	2	0	2	1

KNN (K-nearest neighbors) ?



$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

TEST

3	2	?

TRAINING

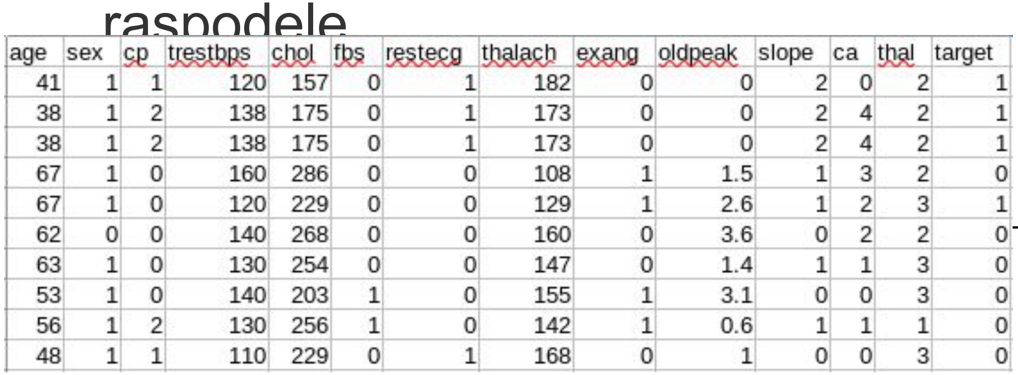
1	2	1
3	6	0
2	4	1
4	5	0

2	1
2.23	1
3,16	0
4	0

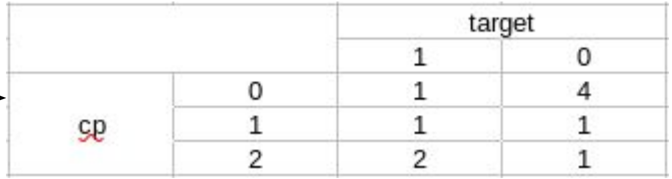
K=3

Naive-Bayes ?

- Bazira se na verovatnoći i statistici
- Kreće od pretpostavke da su svi podaci podjednako važni
- Predviđen za rad sa diskretnim podacima
- Kontinualni podaci podrazumevaju primenu Gausove normalne raspodele



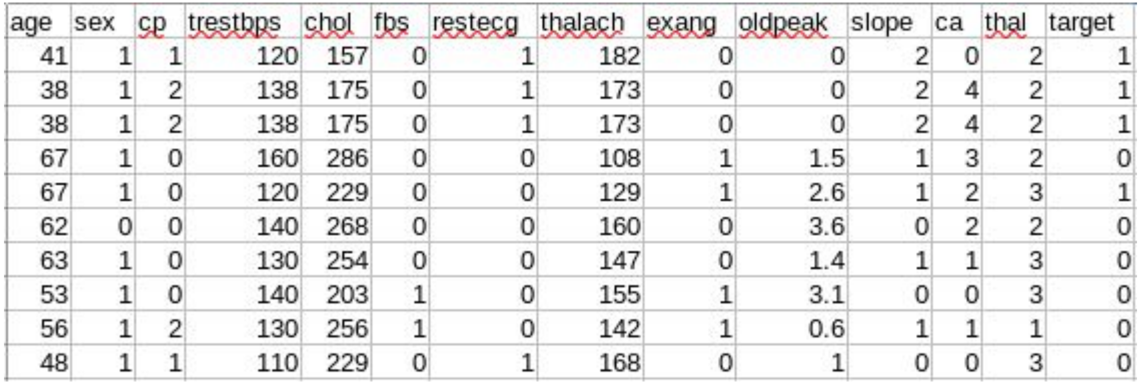
age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
41	1	1	120	157	0	1	182	0	0	2	0	2	1
38	1	2	138	175	0	1	173	0	0	2	4	2	1
38	1	2	138	175	0	1	173	0	0	2	4	2	1
67	1	0	160	286	0	0	108	1	1.5	1	3	2	0
67	1	0	120	229	0	0	129	1	2.6	1	2	3	1
62	0	0	140	268	0	0	160	0	3.6	0	2	2	0
63	1	0	130	254	0	0	147	0	1.4	1	1	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
56	1	2	130	256	1	0	142	1	0.6	1	1	1	0
48	1	1	110	229	0	1	168	0	1	0	0	3	0



		target	
		1	0
cp	0	1	4
	1	1	1
	2	2	1

1/4 4/6

Kontinualni podaci ?
Gausova normalna raspodela !



age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
41	1	1	120	157	0	1	182	0	0	2	0	2	1
38	1	2	138	175	0	1	173	0	0	2	4	2	1
38	1	2	138	175	0	1	173	0	0	2	4	2	1
67	1	0	160	286	0	0	108	1	1.5	1	3	2	0
67	1	0	120	229	0	0	129	1	2.6	1	2	3	1
62	0	0	140	268	0	0	160	0	3.6	0	2	2	0
63	1	0	130	254	0	0	147	0	1.4	1	1	3	0
53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
56	1	2	130	256	1	0	142	1	0.6	1	1	1	0
48	1	1	110	229	0	1	168	0	1	0	0	3	0

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Mean

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)^2\right]^{0.5}$$

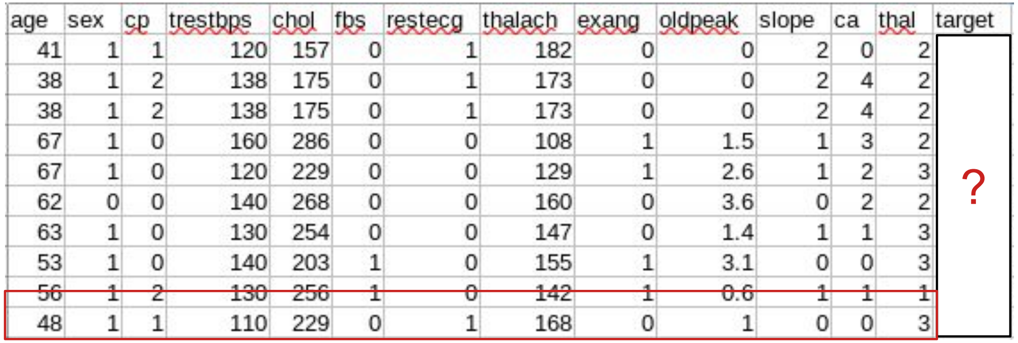
Standard deviation

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

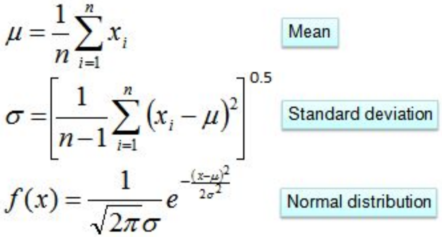
Normal distribution

Kako odrediti pripadnos klasi kod Naive-Bayes?

TEST



age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
41	1	1	120	157	0	1	182	0	0	2	0	2	?
38	1	2	138	175	0	1	173	0	0	2	4	2	
38	1	2	138	175	0	1	173	0	0	2	4	2	
67	1	0	160	286	0	0	108	1	1.5	1	3	2	
67	1	0	120	229	0	0	129	1	2.6	1	2	3	
62	0	0	140	268	0	0	160	0	3.6	0	2	2	
63	1	0	130	254	0	0	147	0	1.4	1	1	3	
53	1	0	140	203	1	0	155	1	3.1	0	0	3	
~~56~~	~~1~~	~~2~~	~~130~~	~~256~~	~~1~~	~~0~~	~~142~~	~~1~~	~~0.6~~	~~1~~	~~1~~	~~1~~	
48	1	1	110	229	0	1	168	0	1	0	0	3	



$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

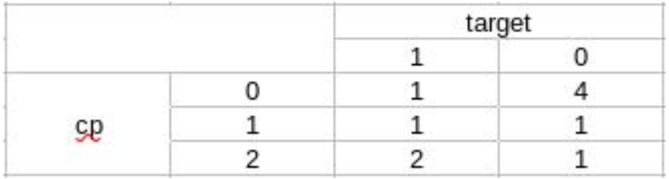
Mean

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu)^2\right]^{0.5}$$

Standard deviation

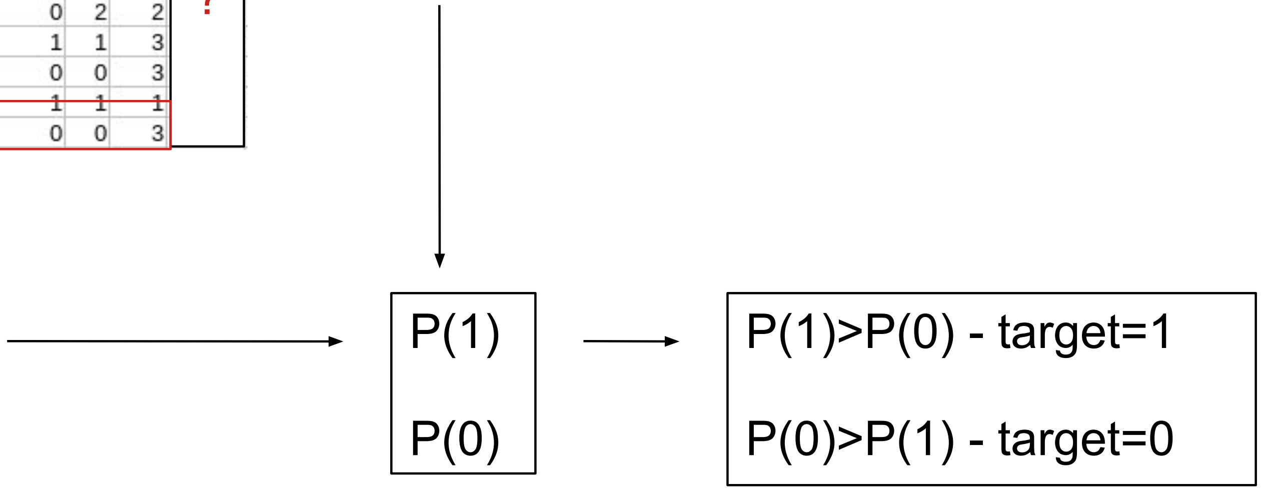
$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Normal distribution



		target	
		1	0
cp	0	1	4
	1	1	1
	2	2	1

1/4 1/6



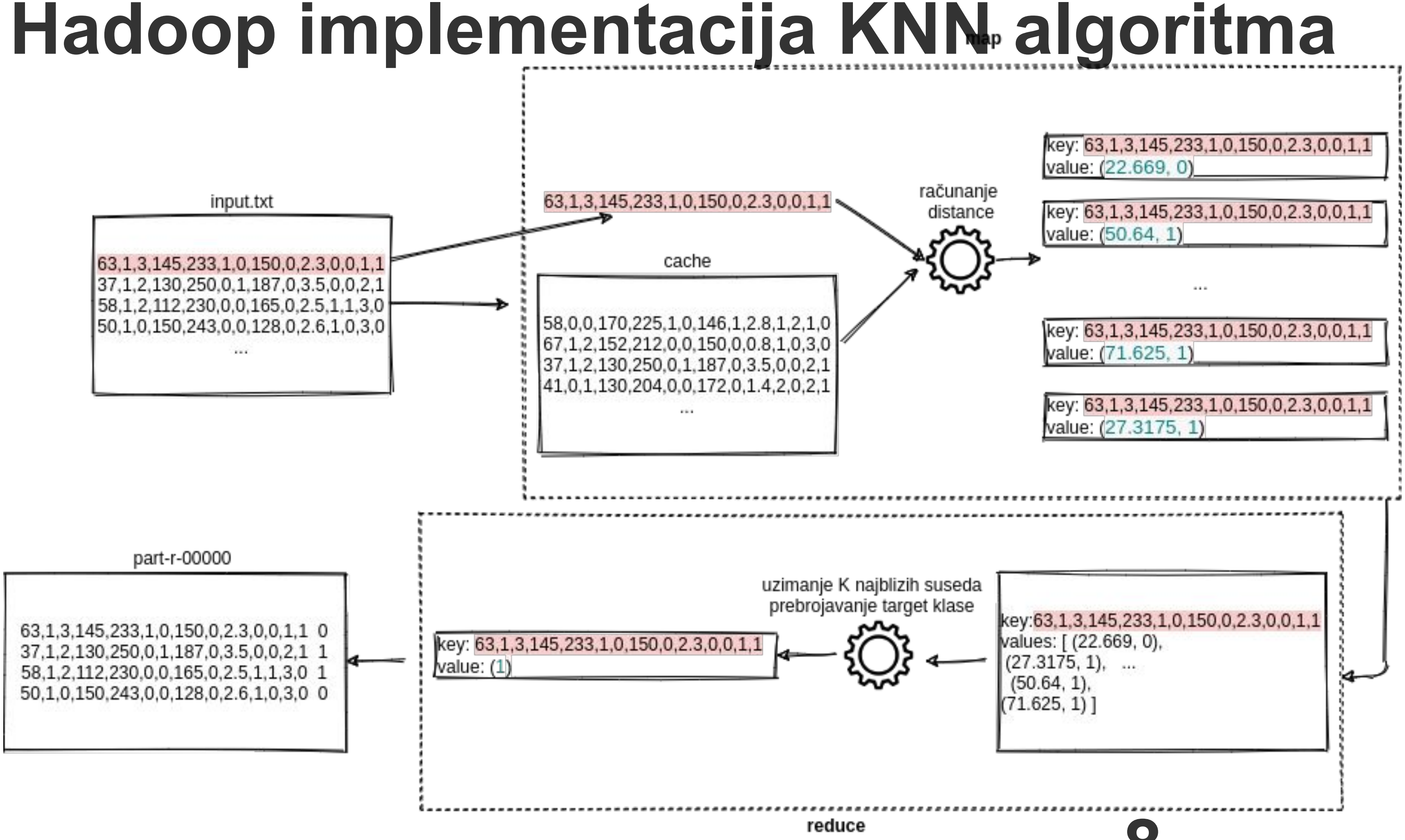
P(1)

P(0)

P(1)>P(0) - target=1

P(0)>P(1) - target=0

Hadoop implementacija KNN algoritma



map

input.txt

```
63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
37,1,2,130,250,0,1,187,0,3.5,0,0,2,1
58,1,2,112,230,0,0,165,0,2.5,1,1,3,0
50,1,0,150,243,0,0,128,0,2.6,1,0,3,0
...
```

63,1,3,145,233,1,0,150,0,2.3,0,0,1,1

cache

```
58,0,0,170,225,1,0,146,1,2.8,1,2,1,0
67,1,2,152,212,0,0,150,0,0.8,1,0,3,0
37,1,2,130,250,0,1,187,0,3.5,0,0,2,1
41,0,1,130,204,0,0,172,0,1.4,2,0,2,1
...
```

računanje distance

key: 63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
value: (22.669, 0)

key: 63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
value: (50.64, 1)

...

key: 63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
value: (71.625, 1)

key: 63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
value: (27.3175, 1)

key:63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
values: [(22.669, 0),
(27.3175, 1), ...
(50.64, 1),
(71.625, 1)]

uzimanje K najblizih suseda
prebrojavanje target klase

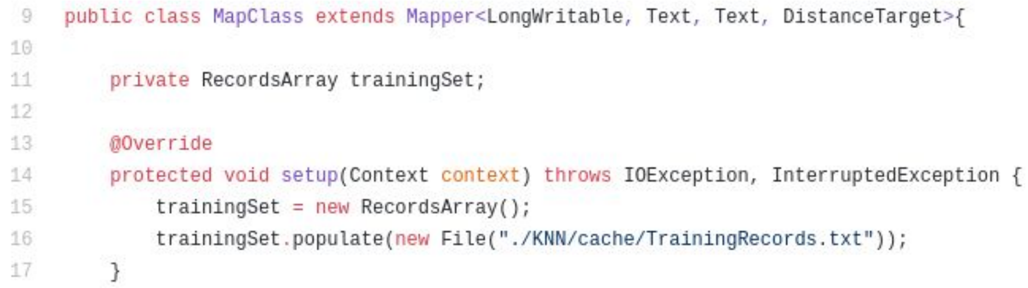
key: 63,1,3,145,233,1,0,150,0,2.3,0,0,1,1
value: (1)

part-r-00000

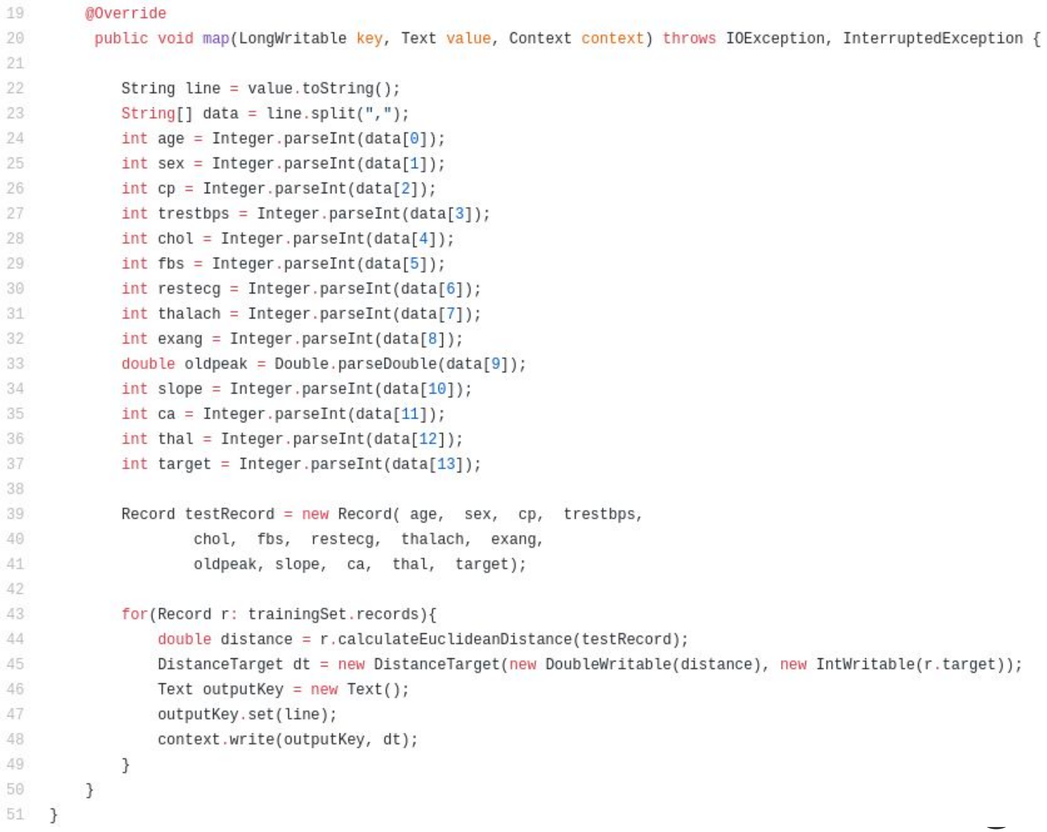
```
63,1,3,145,233,1,0,150,0,2.3,0,0,1,1  0
37,1,2,130,250,0,1,187,0,3.5,0,0,2,1  1
58,1,2,112,230,0,0,165,0,2.5,1,1,3,0  1
50,1,0,150,243,0,0,128,0,2.6,1,0,3,0  0
```

reduce

Mapper

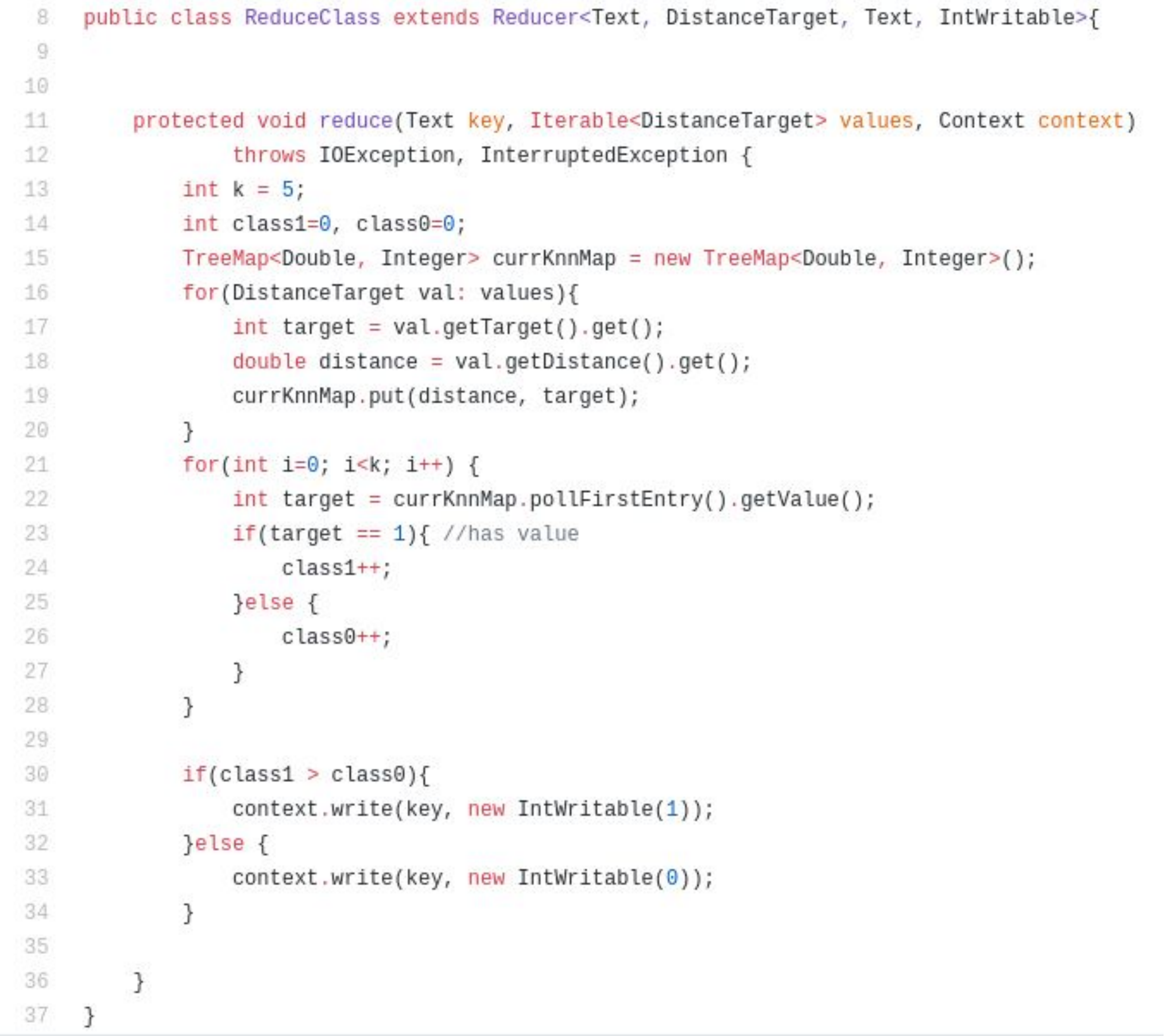


```java
public class MapClass extends Mapper<LongWritable, Text, Text, DistanceTarget>{

    private RecordsArray trainingSet;

    @Override
    protected void setup(Context context) throws IOException, InterruptedException {
        trainingSet = new RecordsArray();
        trainingSet.populate(new File("./KNN/cache/TrainingRecords.txt"));
    }
```



```java
    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        String line = value.toString();
        String[] data = line.split(",");
        int age = Integer.parseInt(data[0]);
        int sex = Integer.parseInt(data[1]);
        int cp = Integer.parseInt(data[2]);
        int trestbps = Integer.parseInt(data[3]);
        int chol = Integer.parseInt(data[4]);
        int fbs = Integer.parseInt(data[5]);
        int restecg = Integer.parseInt(data[6]);
        int thalach = Integer.parseInt(data[7]);
        int exang = Integer.parseInt(data[8]);
        double oldpeak = Double.parseDouble(data[9]);
        int slope = Integer.parseInt(data[10]);
        int ca = Integer.parseInt(data[11]);
        int thal = Integer.parseInt(data[12]);
        int target = Integer.parseInt(data[13]);

        Record testRecord = new Record( age,  sex,  cp,  trestbps,
                chol,  fbs,  restecg,  thalach,  exang,
                oldpeak, slope,  ca,  thal,  target);

        for(Record r: trainingSet.records){
            double distance = r.calculateEuclideanDistance(testRecord);
            DistanceTarget dt = new DistanceTarget(new DoubleWritable(distance), new IntWritable(r.target));
            Text outputKey = new Text();
            outputKey.set(line);
            context.write(outputKey, dt);
        }
    }
}
```

Reducer

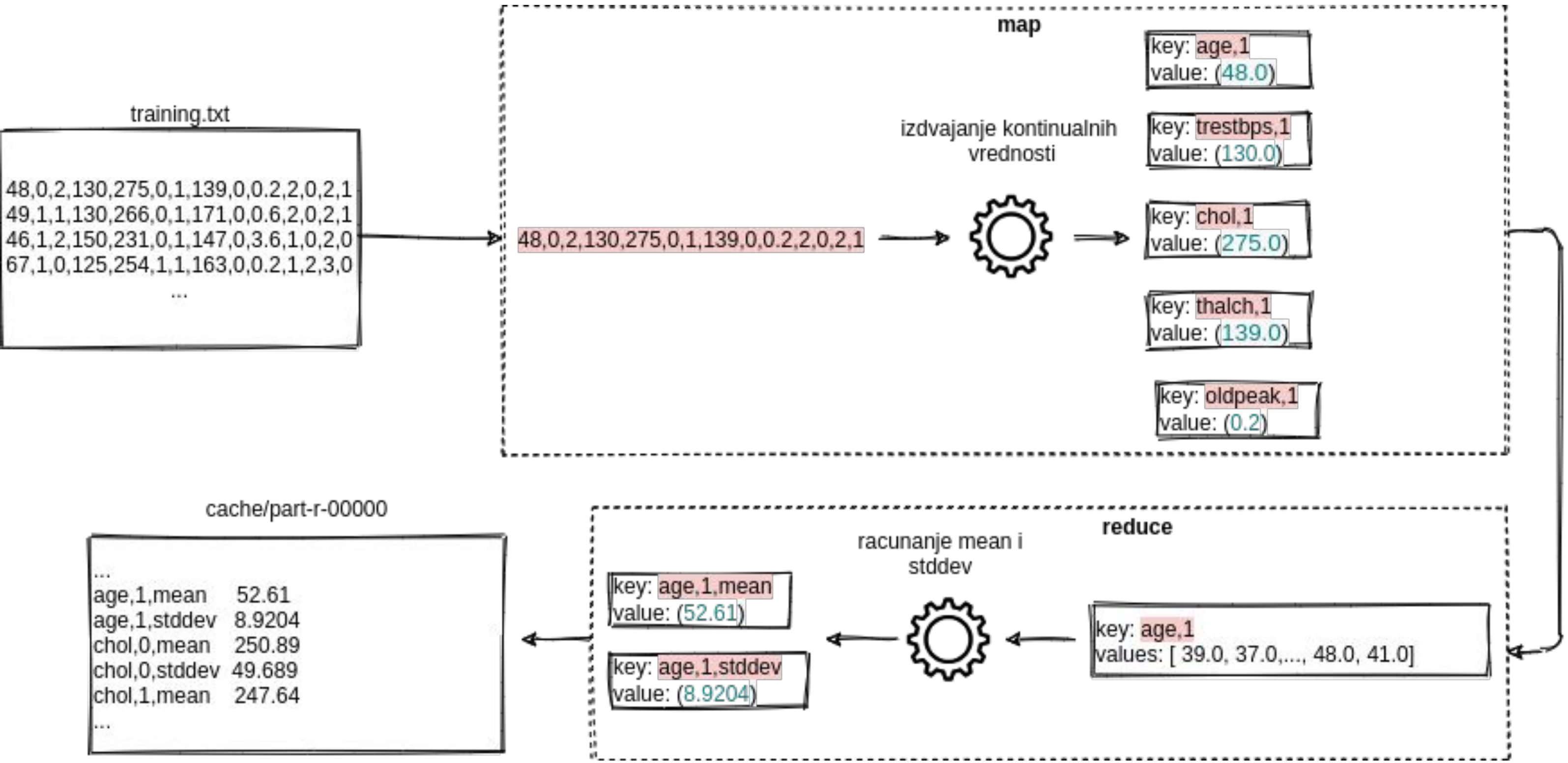


```
public class ReduceClass extends Reducer<Text, DistanceTarget, Text, IntWritable>{


    protected void reduce(Text key, Iterable<DistanceTarget> values, Context context)
            throws IOException, InterruptedException {
        int k = 5;
        int class1=0, class0=0;
        TreeMap<Double, Integer> currKnnMap = new TreeMap<Double, Integer>();
        for(DistanceTarget val: values){
            int target = val.getTarget().get();
            double distance = val.getDistance().get();
            currKnnMap.put(distance, target);
        }
        for(int i=0; i<k; i++) {
            int target = currKnnMap.pollFirstEntry().getValue();
            if(target == 1){ //has value
                class1++;
            }else {
                class0++;
            }
        }

        if(class1 > class0){
            context.write(key, new IntWritable(1));
        }else {
            context.write(key, new IntWritable(0));
        }

    }
}
```

Hadoop implementacija Naive-Bayes algoritma

- 3 faze
  - Faza 1: Prepocesiranje-Računanje standardne devijacije i srednje vrednosti
  - Faza 2: Kreiranja frekvencione tabele za diskretne podatke
  - Faza 3: Određivanje target klase

Faza 1: Računanje standardne devijacije i srednje vrednosti



training.txt

```
48,0,2,130,275,0,1,139,0,0.2,2,0,2,1
49,1,1,130,266,0,1,171,0,0.6,2,0,2,1
46,1,2,150,231,0,1,147,0,3.6,1,0,2,0
67,1,0,125,254,1,1,163,0,0.2,1,2,3,0
...
```

map

48,0,2,130,275,0,1,139,0,0.2,2,0,2,1

izdvajanje kontinualnih vrednosti

key: age,1
value: (48.0)

key: trestbps,1
value: (130.0)

key: chol,1
value: (275.0)

key: thalch,1
value: (139.0)

key: oldpeak,1
value: (0.2)

reduce

key: age,1
values: [39.0, 37.0,..., 48.0, 41.0]

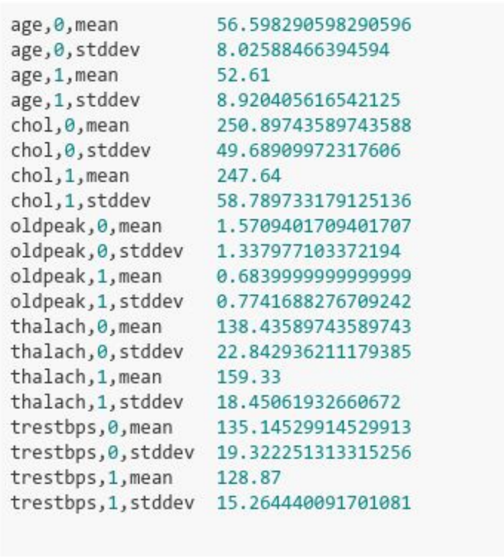
racunanje mean i stddev

key: age,1,mean
value: (52.61)

key: age,1,stddev
value: (8.9204)

cache/part-r-00000

```
...
age,1,mean     52.61
age,1,stddev   8.9204
chol,0,mean    250.89
chol,0,stddev  49.689
chol,1,mean    247.64
...
```



```
age,0,mean          56.598290598290596
age,0,stddev        8.02588466394594
age,1,mean          52.61
age,1,stddev        8.920405616542125
chol,0,mean         250.89743589743588
chol,0,stddev       49.68909972317606
chol,1,mean         247.64
chol,1,stddev       58.789733179125136
oldpeak,0,mean      1.5709401709401707
oldpeak,0,stddev    1.337977103372194
oldpeak,1,mean      0.6839999999999999
oldpeak,1,stddev    0.7741688276709242
thalach,0,mean      138.43589743589743
thalach,0,stddev    22.842936211179385
thalach,1,mean      159.33
thalach,1,stddev    18.45061932660672
trestbps,0,mean     135.14529914529913
trestbps,0,stddev   19.322251313315256
trestbps,1,mean     128.87
trestbps,1,stddev   15.264440091701081
```

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Mean

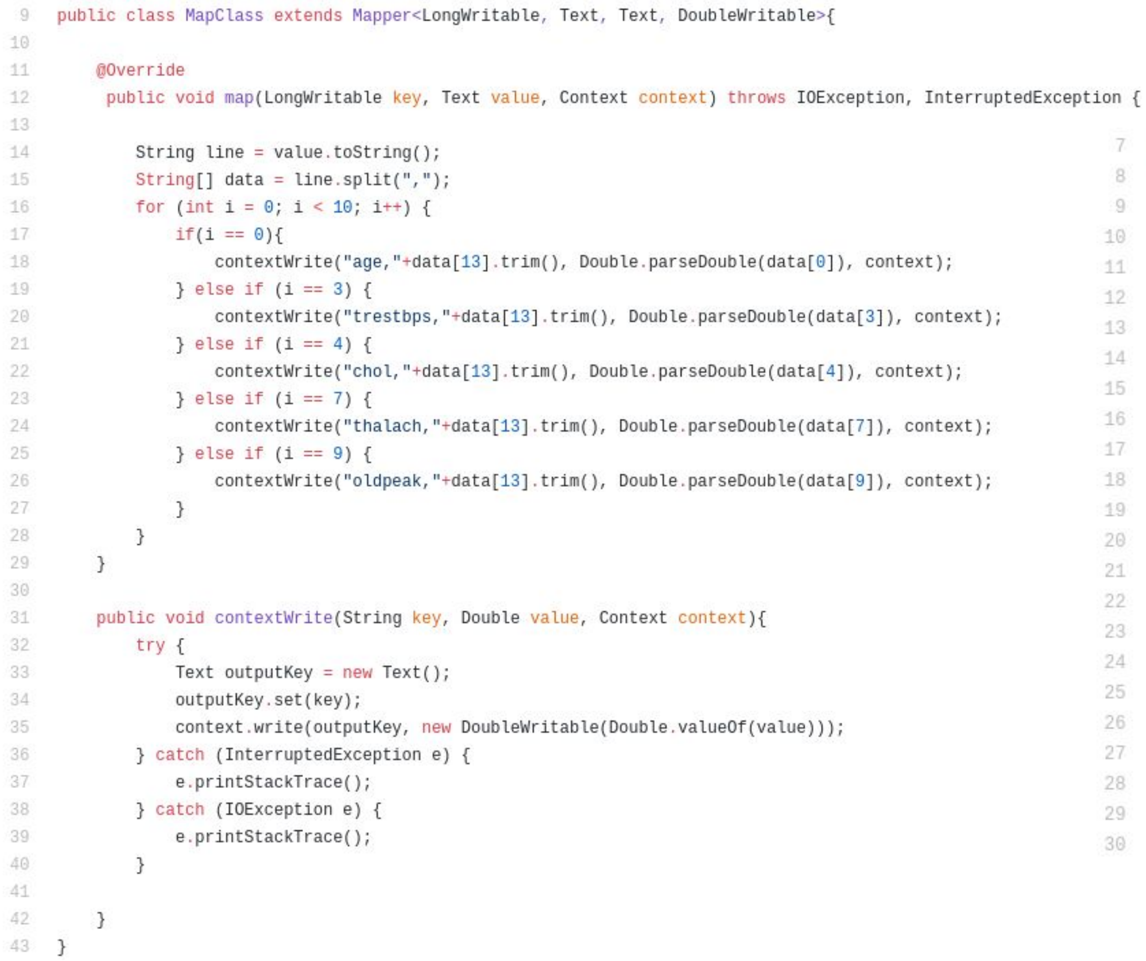
$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2\right]^{0.5}$$

Standard deviation

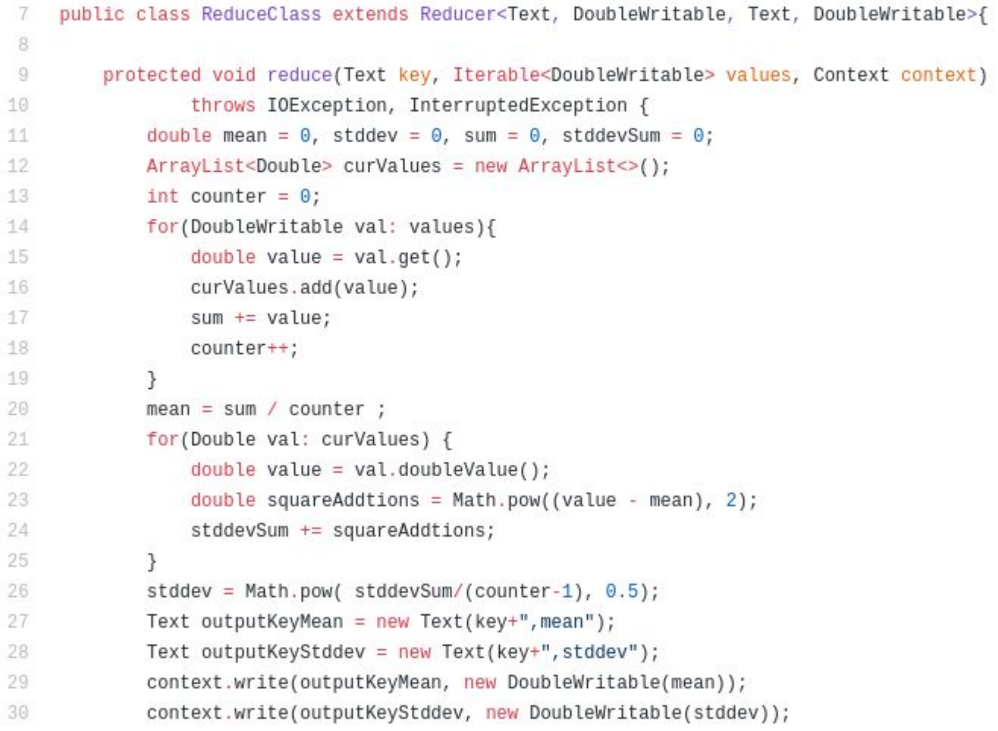
$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Normal distribution

Faza 1: Hadoop implementacija

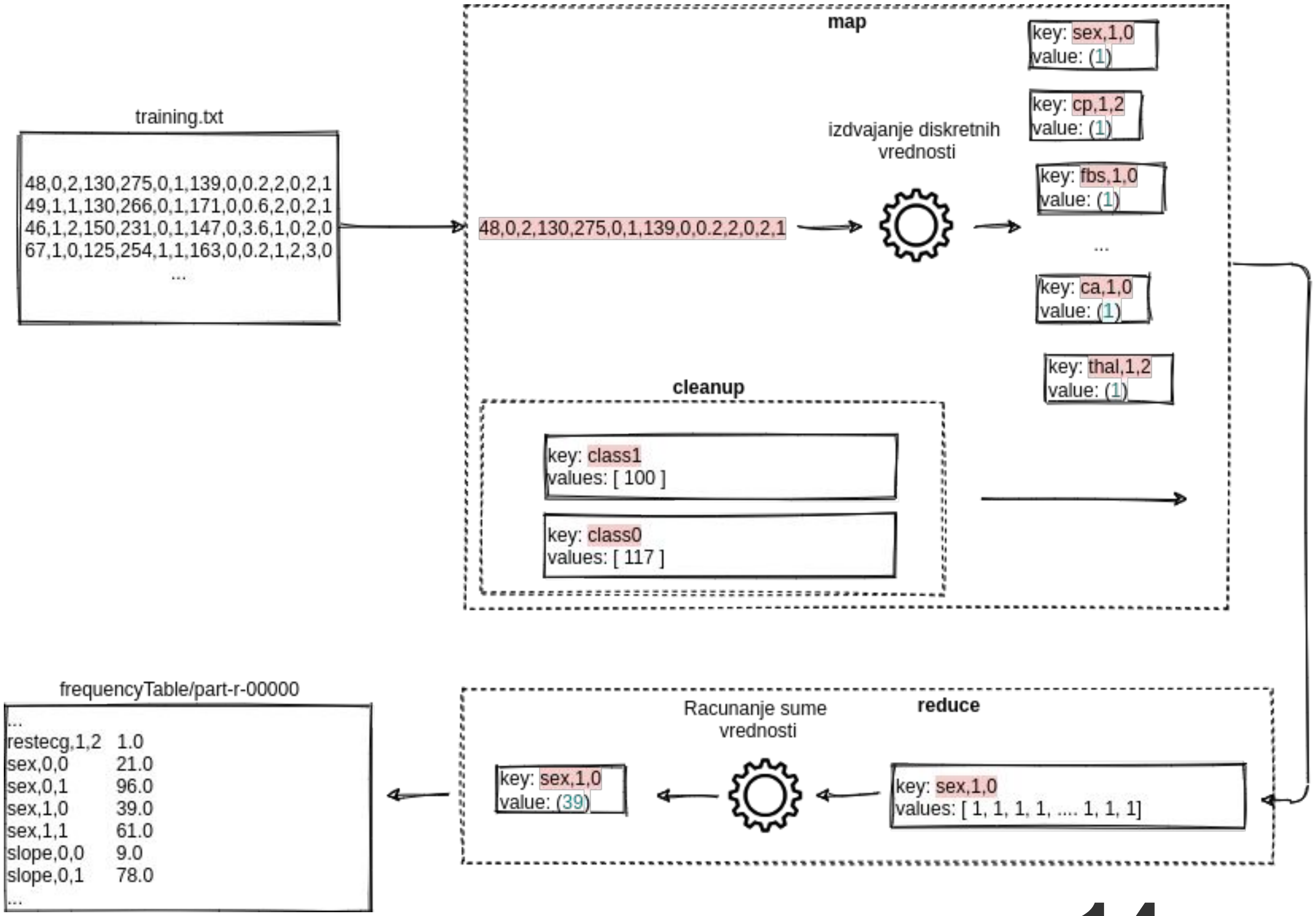


```java
public class MapClass extends Mapper<LongWritable, Text, Text, DoubleWritable>{

    @Override
     public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        String line = value.toString();
        String[] data = line.split(",");
        for (int i = 0; i < 10; i++) {
            if(i == 0){
                contextWrite("age,"+data[13].trim(), Double.parseDouble(data[0]), context);
            } else if (i == 3) {
                contextWrite("trestbps,"+data[13].trim(), Double.parseDouble(data[3]), context);
            } else if (i == 4) {
                contextWrite("chol,"+data[13].trim(), Double.parseDouble(data[4]), context);
            } else if (i == 7) {
                contextWrite("thalach,"+data[13].trim(), Double.parseDouble(data[7]), context);
            } else if (i == 9) {
                contextWrite("oldpeak,"+data[13].trim(), Double.parseDouble(data[9]), context);
            }
        }
    }

    public void contextWrite(String key, Double value, Context context){
        try {
            Text outputKey = new Text();
            outputKey.set(key);
            context.write(outputKey, new DoubleWritable(Double.valueOf(value)));
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
}
```



```java
public class ReduceClass extends Reducer<Text, DoubleWritable, Text, DoubleWritable>{

    protected void reduce(Text key, Iterable<DoubleWritable> values, Context context)
            throws IOException, InterruptedException {
        double mean = 0, stddev = 0, sum = 0, stddevSum = 0;
        ArrayList<Double> curValues = new ArrayList<>();
        int counter = 0;
        for(DoubleWritable val: values){
            double value = val.get();
            curValues.add(value);
            sum += value;
            counter++;
        }
        mean = sum / counter ;
        for(Double val: curValues) {
            double value = val.doubleValue();
            double squareAddtions = Math.pow((value - mean), 2);
            stddevSum += squareAddtions;
        }
        stddev = Math.pow( stddevSum/(counter-1), 0.5);
        Text outputKeyMean = new Text(key+",mean");
        Text outputKeyStddev = new Text(key+",stddev");
        context.write(outputKeyMean, new DoubleWritable(mean));
        context.write(outputKeyStddev, new DoubleWritable(stddev));
```

Faza 2: Kreiranje frekvencione tabele



```
48,0,2,130,275,0,1,139,0,0.2,2,0,2,1
49,1,1,130,266,0,1,171,0,0.6,2,0,2,1
46,1,2,150,231,0,1,147,0,3.6,1,0,2,0
67,1,0,125,254,1,1,163,0,0.2,1,2,3,0
...
```

```
key: sex,1,0
value: (1)

key: cp,1,2
value: (1)

key: fbs,1,0
value: (1)

...

key: ca,1,0
value: (1)

key: thal,1,2
value: (1)
```

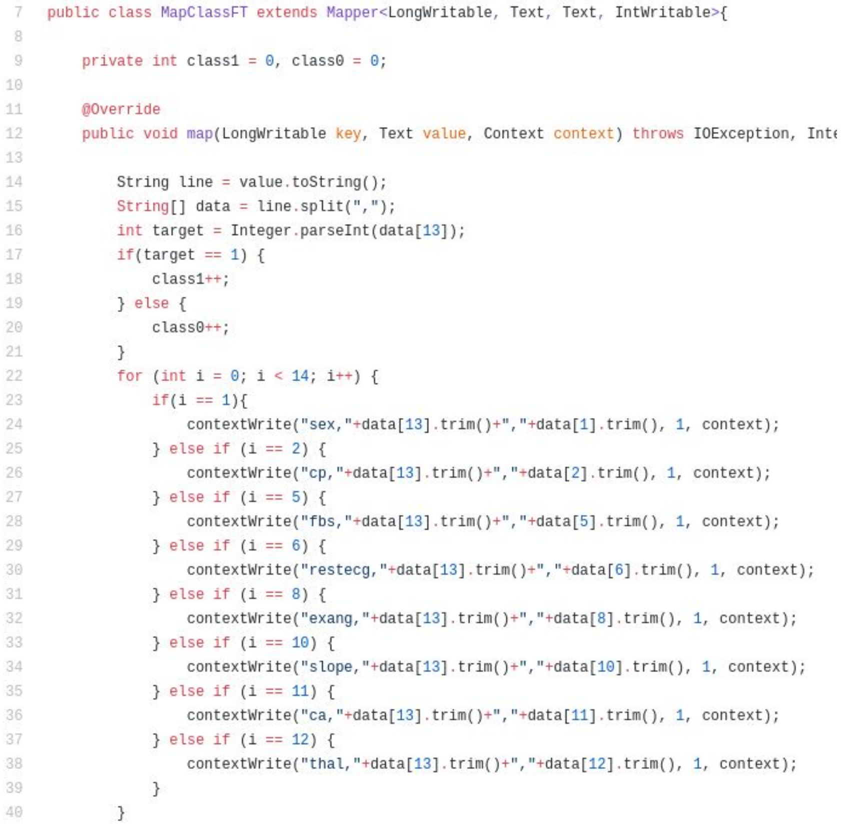
```
key: class1
values: [ 100 ]

key: class0
values: [ 117 ]
```

```
key: sex,1,0
values: [ 1, 1, 1, 1, .... 1, 1, 1]

key: sex,1,0
value: (39)
```

```
...
restecg,1,2   1.0
sex,0,0       21.0
sex,0,1       96.0
sex,1,0       39.0
sex,1,1       61.0
slope,0,0     9.0
slope,0,1     78.0
...
```

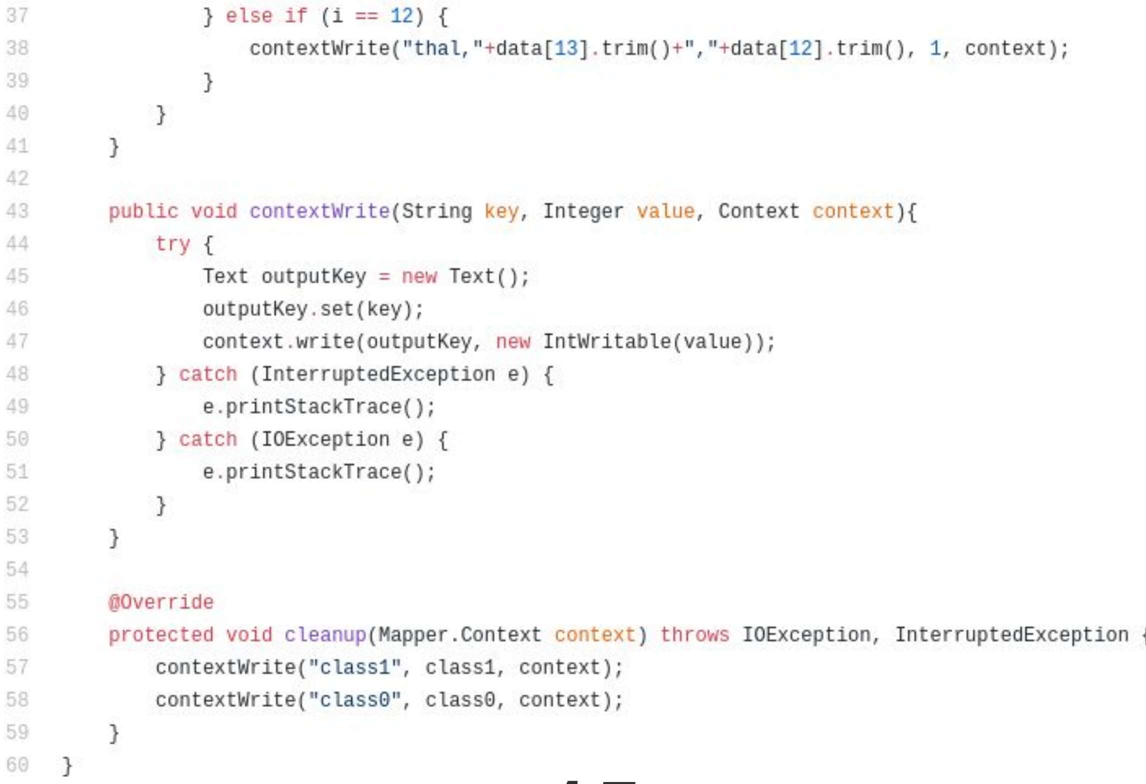
Faza 2: Hadoop implementacija

Part1



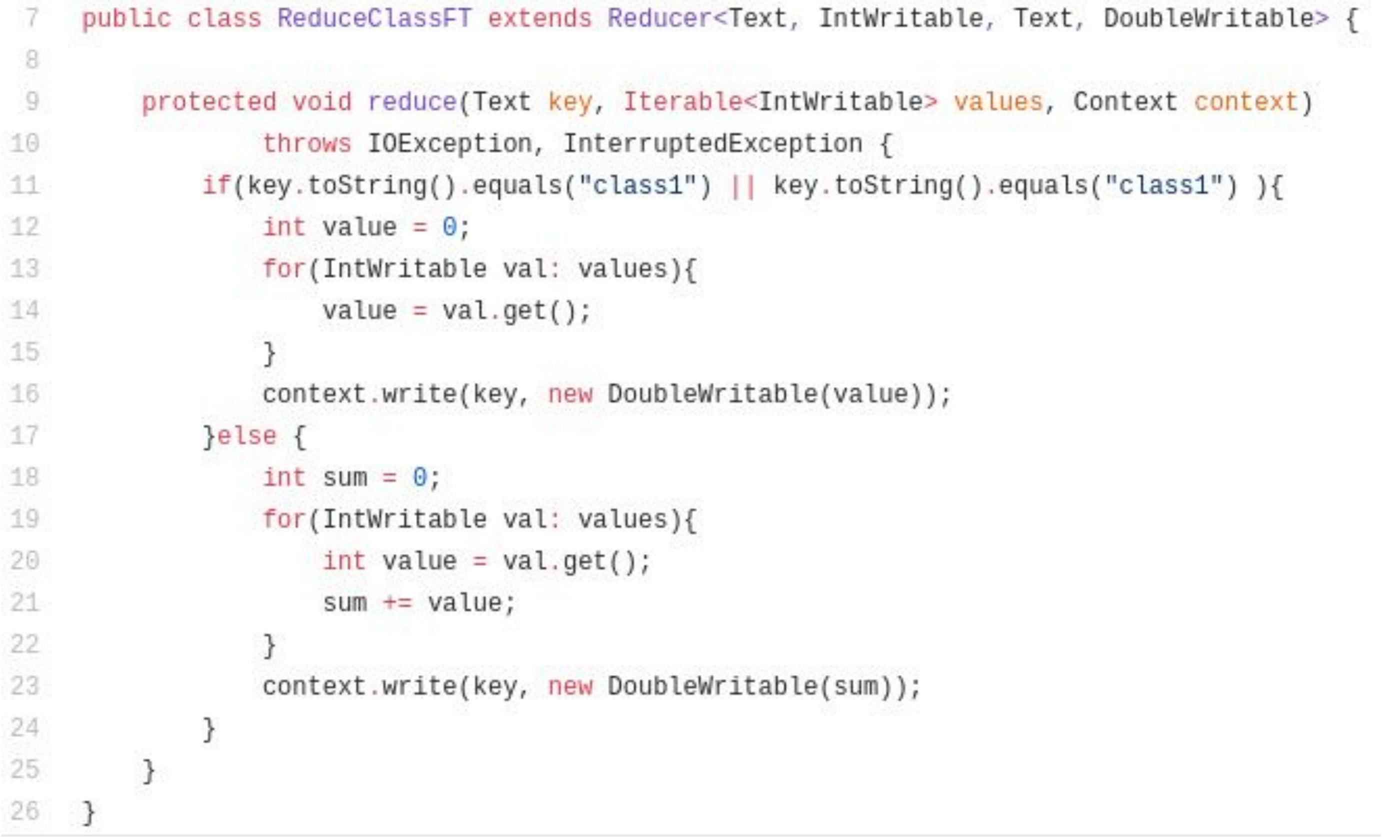
```java
public class MapClassFT extends Mapper<LongWritable, Text, Text, IntWritable>{

    private int class1 = 0, class0 = 0;

    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException, Int

        String line = value.toString();
        String[] data = line.split(",");
        int target = Integer.parseInt(data[13]);
        if(target == 1) {
            class1++;
        } else {
            class0++;
        }
        for (int i = 0; i < 14; i++) {
            if(i == 1){
                contextWrite("sex,"+data[13].trim()+","+data[1].trim(), 1, context);
            } else if (i == 2) {
                contextWrite("cp,"+data[13].trim()+","+data[2].trim(), 1, context);
            } else if (i == 5) {
                contextWrite("fbs,"+data[13].trim()+","+data[5].trim(), 1, context);
            } else if (i == 6) {
                contextWrite("restecg,"+data[13].trim()+","+data[6].trim(), 1, context);
            } else if (i == 8) {
                contextWrite("exang,"+data[13].trim()+","+data[8].trim(), 1, context);
            } else if (i == 10) {
                contextWrite("slope,"+data[13].trim()+","+data[10].trim(), 1, context);
            } else if (i == 11) {
                contextWrite("ca,"+data[13].trim()+","+data[11].trim(), 1, context);
            } else if (i == 12) {
                contextWrite("thal,"+data[13].trim()+","+data[12].trim(), 1, context);
            }
        }
```

Part2



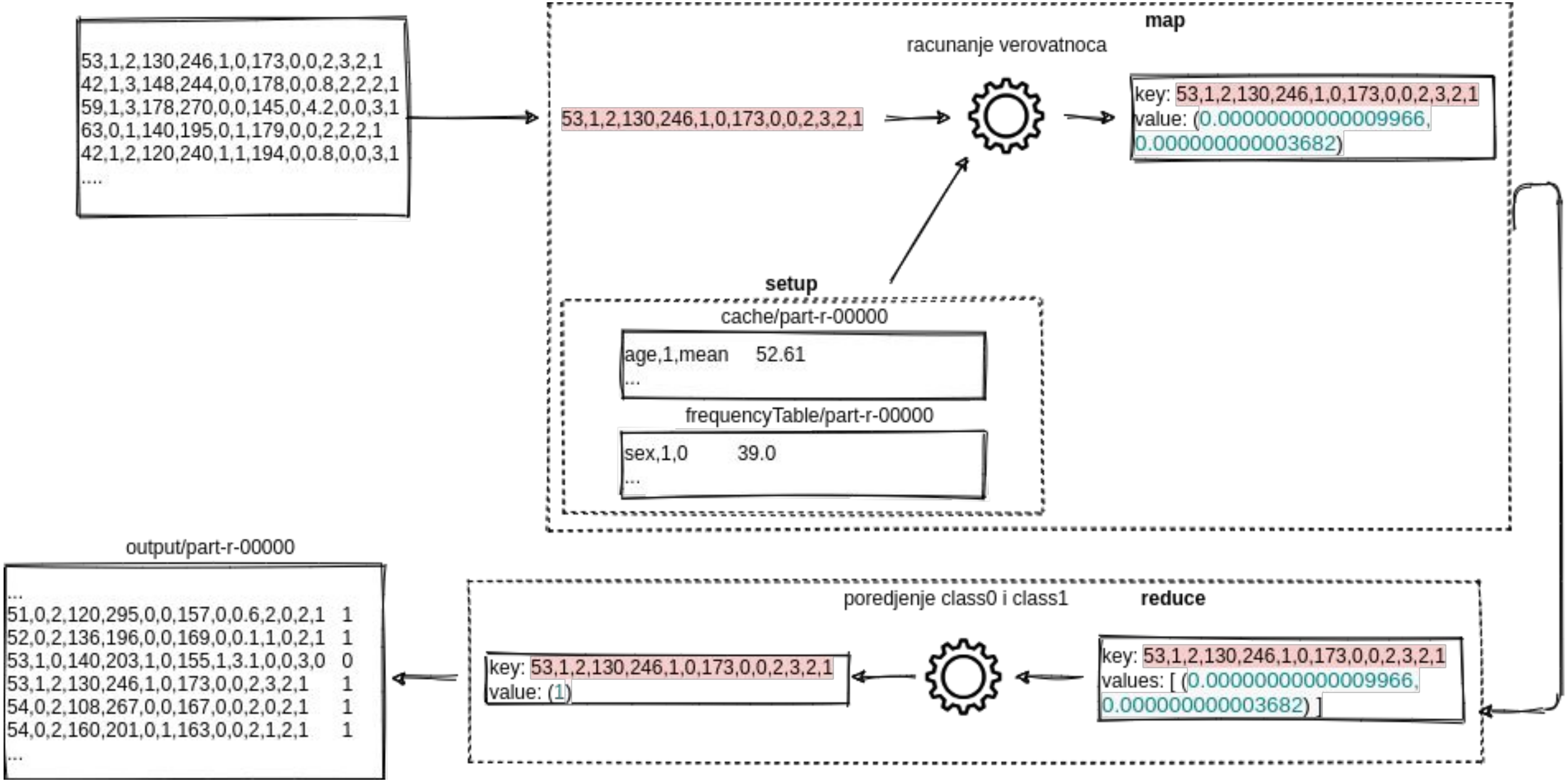
```java
            } else if (i == 12) {
                contextWrite("thal,"+data[13].trim()+","+data[12].trim(), 1, context);
            }
        }
    }

    public void contextWrite(String key, Integer value, Context context){
        try {
            Text outputKey = new Text();
            outputKey.set(key);
            context.write(outputKey, new IntWritable(value));
        } catch (InterruptedException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Override
    protected void cleanup(Mapper.Context context) throws IOException, InterruptedException {
        contextWrite("class1", class1, context);
        contextWrite("class0", class0, context);
    }
}
```

Faza 2: Reducer



```java
public class ReduceClassFT extends Reducer<Text, IntWritable, Text, DoubleWritable> {

  protected void reduce(Text key, Iterable<IntWritable> values, Context context)
      throws IOException, InterruptedException {
    if(key.toString().equals("class1") || key.toString().equals("class1") ){
      int value = 0;
      for(IntWritable val: values){
        value = val.get();
      }
      context.write(key, new DoubleWritable(value));
    }else {
      int sum = 0;
      for(IntWritable val: values){
        int value = val.get();
        sum += value;
      }
      context.write(key, new DoubleWritable(sum));
    }
  }
}
```

Faza 3: Određivanje target klase



```
53,1,2,130,246,1,0,173,0,0,2,3,2,1
42,1,3,148,244,0,0,178,0,0.8,2,2,2,1
59,1,3,178,270,0,0,145,0,4.2,0,0,3,1
63,0,1,140,195,0,1,179,0,0,2,2,2,1
42,1,2,120,240,1,1,194,0,0.8,0,0,3,1
....
```

map

racunanje verovatnoca

53,1,2,130,246,1,0,173,0,0,2,3,2,1

```
key: 53,1,2,130,246,1,0,173,0,0,2,3,2,1
value: (0.00000000000009966,
0.000000000003682)
```

setup

cache/part-r-00000

```
age,1,mean     52.61
...
```

frequencyTable/part-r-00000

```
sex,1,0        39.0
...
```

poredjenje class0 i class1

reduce

```
key: 53,1,2,130,246,1,0,173,0,0,2,3,2,1
values: [ (0.00000000000009966,
0.000000000003682) ]
```

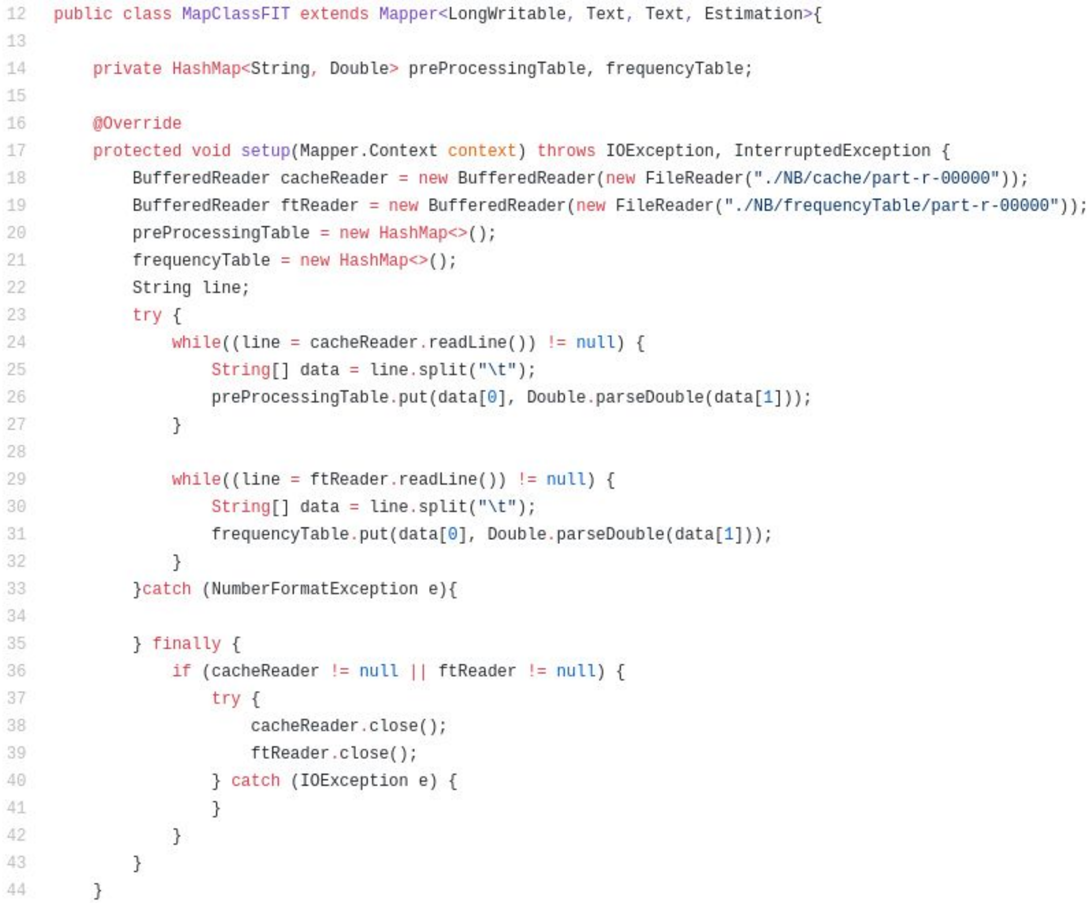
```
key: 53,1,2,130,246,1,0,173,0,0,2,3,2,1
value: (1)
```

output/part-r-00000

```
...
51,0,2,120,295,0,0,157,0,0.6,2,0,2,1   1
52,0,2,136,196,0,0,169,0,0.1,1,0,2,1   1
53,1,0,140,203,1,0,155,1,3.1,0,0,3,0   0
53,1,2,130,246,1,0,173,0,0,2,3,2,1     1
54,0,2,108,267,0,0,167,0,0,2,0,2,1     1
54,0,2,160,201,0,1,163,0,0,2,1,2,1     1
...
```

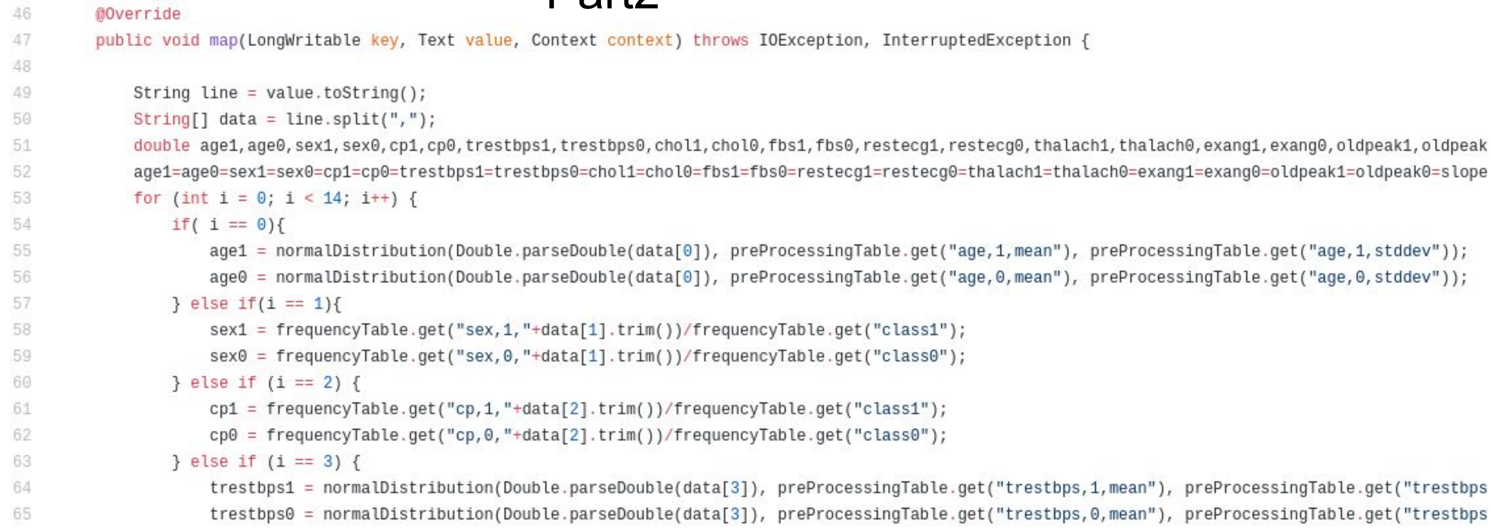
Faza 3: Hadoop implementacija

Part1



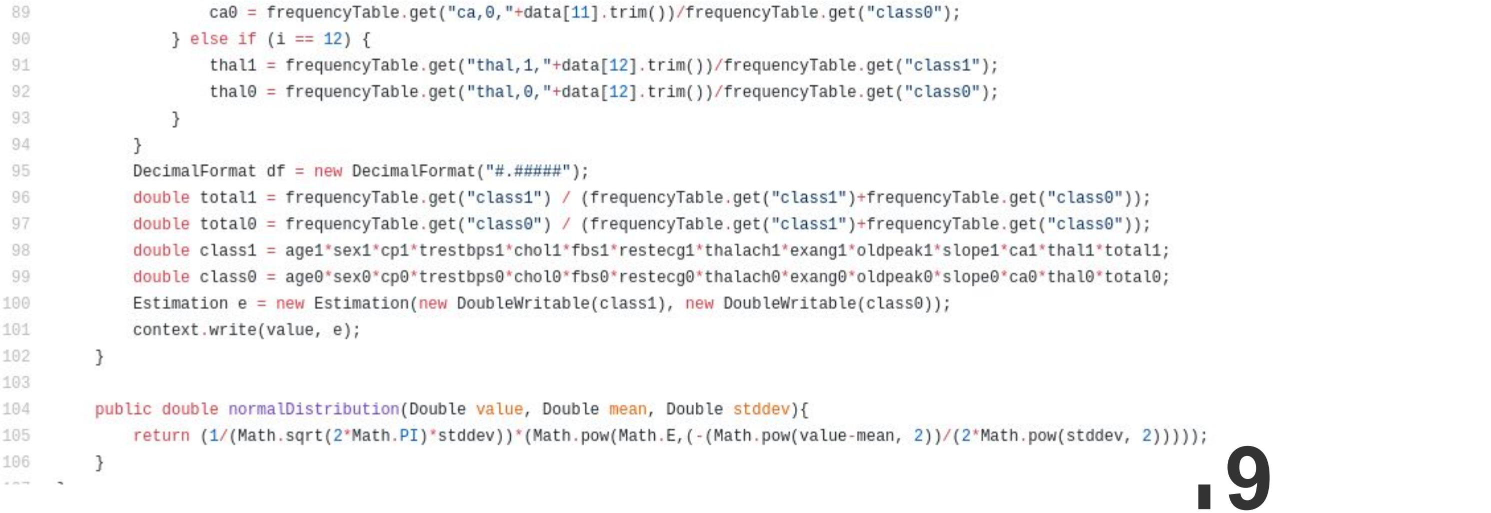
```java
public class MapClassFIT extends Mapper<LongWritable, Text, Text, Estimation>{

    private HashMap<String, Double> preProcessingTable, frequencyTable;

    @Override
    protected void setup(Mapper.Context context) throws IOException, InterruptedException {
        BufferedReader cacheReader = new BufferedReader(new FileReader("./NB/cache/part-r-00000"));
        BufferedReader ftReader = new BufferedReader(new FileReader("./NB/frequencyTable/part-r-00000"));
        preProcessingTable = new HashMap<>();
        frequencyTable = new HashMap<>();
        String line;
        try {
            while((line = cacheReader.readLine()) != null) {
                String[] data = line.split("\t");
                preProcessingTable.put(data[0], Double.parseDouble(data[1]));
            }

            while((line = ftReader.readLine()) != null) {
                String[] data = line.split("\t");
                frequencyTable.put(data[0], Double.parseDouble(data[1]));
            }
        }catch (NumberFormatException e){

        } finally {
            if (cacheReader != null || ftReader != null) {
                try {
                    cacheReader.close();
                    ftReader.close();
                } catch (IOException e) {
                }
            }
        }
    }
```

Part2



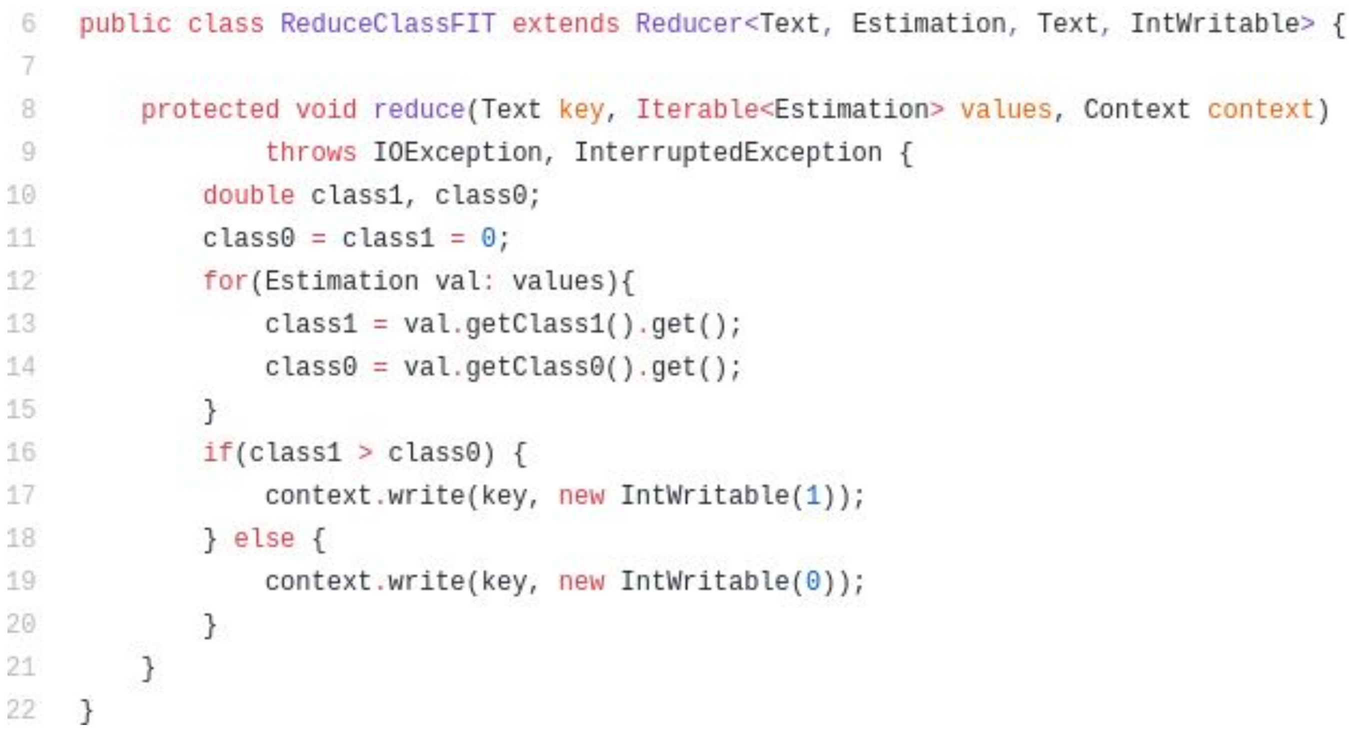
```java
    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        String line = value.toString();
        String[] data = line.split(",");
        double age1,age0,sex1,sex0,cp1,cp0,trestbps1,trestbps0,chol1,chol0,fbs1,fbs0,restecg1,restecg0,thalach1,thalach0,exang1,exang0,oldpeak1,oldpeak
        age1=age0=sex1=sex0=cp1=cp0=trestbps1=trestbps0=chol1=chol0=fbs1=fbs0=restecg1=restecg0=thalach1=thalach0=exang1=exang0=oldpeak1=oldpeak0=slope
        for (int i = 0; i < 14; i++) {
            if( i == 0){
                age1 = normalDistribution(Double.parseDouble(data[0]), preProcessingTable.get("age,1,mean"), preProcessingTable.get("age,1,stddev"));
                age0 = normalDistribution(Double.parseDouble(data[0]), preProcessingTable.get("age,0,mean"), preProcessingTable.get("age,0,stddev"));
            } else if(i == 1){
                sex1 = frequencyTable.get("sex,1,"+data[1].trim())/frequencyTable.get("class1");
                sex0 = frequencyTable.get("sex,0,"+data[1].trim())/frequencyTable.get("class0");
            } else if (i == 2) {
                cp1 = frequencyTable.get("cp,1,"+data[2].trim())/frequencyTable.get("class1");
                cp0 = frequencyTable.get("cp,0,"+data[2].trim())/frequencyTable.get("class0");
            } else if (i == 3) {
                trestbps1 = normalDistribution(Double.parseDouble(data[3]), preProcessingTable.get("trestbps,1,mean"), preProcessingTable.get("trestbps
                trestbps0 = normalDistribution(Double.parseDouble(data[3]), preProcessingTable.get("trestbps,0,mean"), preProcessingTable.get("trestbps
```

...



```java
                ca0 = frequencyTable.get("ca,0,"+data[11].trim())/frequencyTable.get("class0");
            } else if (i == 12) {
                thal1 = frequencyTable.get("thal,1,"+data[12].trim())/frequencyTable.get("class1");
                thal0 = frequencyTable.get("thal,0,"+data[12].trim())/frequencyTable.get("class0");
            }
        }
        DecimalFormat df = new DecimalFormat("#.#####");
        double total1 = frequencyTable.get("class1") / (frequencyTable.get("class1")+frequencyTable.get("class0"));
        double total0 = frequencyTable.get("class0") / (frequencyTable.get("class1")+frequencyTable.get("class0"));
        double class1 = age1*sex1*cp1*trestbps1*chol1*fbs1*restecg1*thalach1*exang1*oldpeak1*slope1*ca1*thal1*total1;
        double class0 = age0*sex0*cp0*trestbps0*chol0*fbs0*restecg0*thalach0*exang0*oldpeak0*slope0*ca0*thal0*total0;
        Estimation e = new Estimation(new DoubleWritable(class1), new DoubleWritable(class0));
        context.write(value, e);
    }

    public double normalDistribution(Double value, Double mean, Double stddev){
        return (1/(Math.sqrt(2*Math.PI)*stddev))*(Math.pow(Math.E,(-(Math.pow(value-mean, 2))/(2*Math.pow(stddev, 2)))));
    }
```

Faza 3: Reducer



```java
public class ReduceClassFIT extends Reducer<Text, Estimation, Text, IntWritable> {

    protected void reduce(Text key, Iterable<Estimation> values, Context context)
            throws IOException, InterruptedException {
        double class1, class0;
        class0 = class1 = 0;
        for(Estimation val: values){
            class1 = val.getClass1().get();
            class0 = val.getClass0().get();
        }
        if(class1 > class0) {
            context.write(key, new IntWritable(1));
        } else {
            context.write(key, new IntWritable(0));
        }
    }
}
```

Hvala na pažnji!