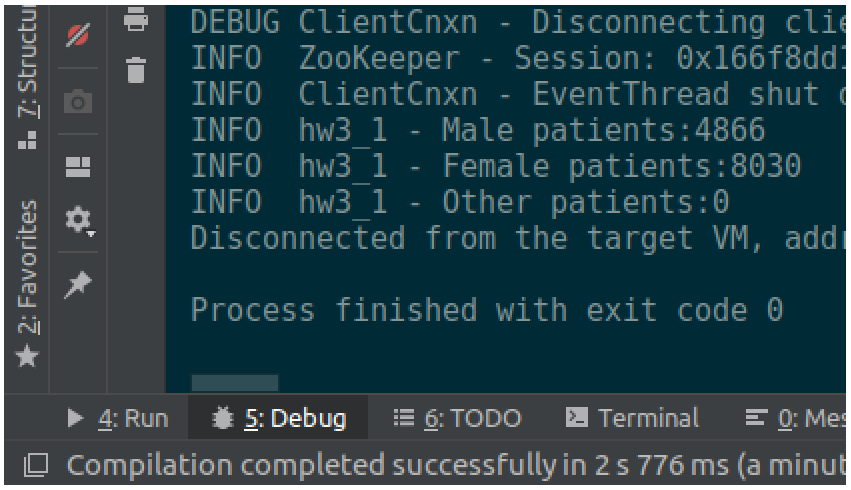
R1.1,R1.3

```
    static void loadPatients(Configuration conf) throws IOException {
        /*R1.1,R1.3*/
        Map<String,String> p1Gender = new HashMap<>();
        Map<String,String> p2Gender = new HashMap<>();
        Map<String,String> p1Asthma = new HashMap<>();
        Map<String,String> p2Asthma = new HashMap<>();
        /*Patient1.csv mappings*/
        //1-> male
        p1Gender.put("1","0");
        //2->female
        p1Gender.put("2","1");
        //0->no
        p1Asthma.put("0","0");
        //1->yes
        p1Asthma.put("1","1");
        //9->unknown
        p1Asthma.put("8","-9");

        /*Patient2.csv mappings*/
        //0-> female
        p2Gender.put("0","1");
        //1->male
        p2Gender.put("1","0");
        //1->Yes
        p2Asthma.put("1","1");
        //2->No
        p2Asthma.put("2","0");

        Connection conn;
        conn = ConnectionFactory.createConnection(conf);
        Table patients = conn.getTable(TableName.valueOf("patients"));
        putRows("patients1.csv",patients,p1Gender,p1Asthma);
        putRows("patients2.csv",patients,p2Gender,p2Asthma);
        patients.close();
        conn.close();
    }
    /*R1.1*/
    static void putRows(String resourceName, Table table,Map<String,String>
genderMap,Map<String,String> asthmaMap) throws IOException{
        BufferedReader reader = new BufferedReader(new
FileReader(Main.class.getResource(resourceName).getFile()));
        reader.readLine();//skip header line
        while (reader.ready()) {
            String rawline = reader.readLine();
            String[] line = rawline.split(",");
            Put row = prepareRow(line, genderMap, asthmaMap);
            table.put(row);
        }
        reader.close();
    }
    /*R1.1*/
    static Put prepareRow(String[] line, Map<String,String> genderMap,
Map<String,String>asthmaMap){
        Put row = new Put(Bytes.toBytes(line[0]));
        /*R1.3*/
        /*Set NULL to unknown for all fields*/
        for(int i=0;i<line.length;i++){
            if(line[i].equals("NULL")){
                line[i] = "-9";
            }
        }
        row.addColumn(Bytes.toBytes("demographics"), Bytes.toBytes("gender"),
Bytes.toBytes(genderMap.getOrDefault(line[1],"-9")));
        row.addColumn(Bytes.toBytes("demographics"),Bytes.toBytes("race"),Bytes.toBytes(line[2]));
        row.addColumn(Bytes.toBytes("anthropometry"),Bytes.toBytes("height"),Bytes.toBytes(line[3]));
        row.addColumn(Bytes.toBytes("anthropometry"),Bytes.toBytes("weight"),Bytes.toBytes(line[4]));

row.addColumn(Bytes.toBytes("medical_history"),Bytes.toBytes("asthma"),Bytes.toBytes(asthmaMap.
getOrDefault(line[5],"-9")));

row.addColumn(Bytes.toBytes("medical_history"),Bytes.toBytes("hypertension"),Bytes.toBytes(line[6
]));
        row.addColumn(Bytes.toBytes("other"),Bytes.toBytes("pid"),Bytes.toBytes(line[0]));
        row.addColumn(Bytes.toBytes("other"),Bytes.toBytes("year"),Bytes.toBytes(line[7]));
        return row;
    }
    /*R1.1*/
    public static void createPatientsTable(Configuration conf) throws IOException{
            Connection conn = ConnectionFactory.createConnection(conf);
            Admin admin = conn.getAdmin();
            List<ColumnFamilyDescriptor> columnFamilies =
                    Stream.of("demographics","anthropometry","medical_history","other")
                            .map(Bytes::toBytes)
                            .map(ColumnFamilyDescriptorBuilder::newBuilder)
                            .map(ColumnFamilyDescriptorBuilder::build)
                            .collect(Collectors.toList());
            TableDescriptor patientsDesc =
                    TableDescriptorBuilder.newBuilder(TableName.valueOf("patients"))
                            .setColumnFamilies(columnFamilies)
                            .build();
            admin.createTable(patientsDesc);
    }
```

R1.2

```
Connection conn;
        long maleCount = 0;
        long femaleCount = 0;
        long otherCount = 0;
        try {
            conn = ConnectionFactory.createConnection(conf);
            Table patients = conn.getTable(TableName.valueOf("patients"));
            loadPatients(conf);
            Scan patientScan = new Scan();
            patientScan.addColumn(Bytes.toBytes("demographics"),Bytes.toBytes("gender"));
            ResultScanner patientScanner = patients.getScanner(patientScan);
            for(Result patient: patientScanner){
                String gender = 
Bytes.toString(patient.getValue(Bytes.toBytes("demographics"),Bytes.toBytes("gender")));
                if(gender.equals("0")){
                    maleCount++;
                }
                else if(gender.equals("1")){
                    femaleCount++;
                }
                else{
                    otherCount++;
                }
            }
            log.info("Male patients:"+maleCount);
            log.info("Female patients:"+femaleCount);
            log.info("Other patients:"+otherCount);
        }
        catch(IOException ex){
            log.error(ExceptionUtils.getStackTrace(ex));
        }
```



```
DEBUG ClientCnxn - Disconnecting clie
INFO  ZooKeeper - Session: 0x166f8dd1
INFO  ClientCnxn - EventThread shut d
INFO  hw3_1 - Male patients:4866
INFO  hw3_1 - Female patients:8030
INFO  hw3_1 - Other patients:0
Disconnected from the target VM, addr

Process finished with exit code 0
```

Compilation completed successfully in 2 s 776 ms (a minut

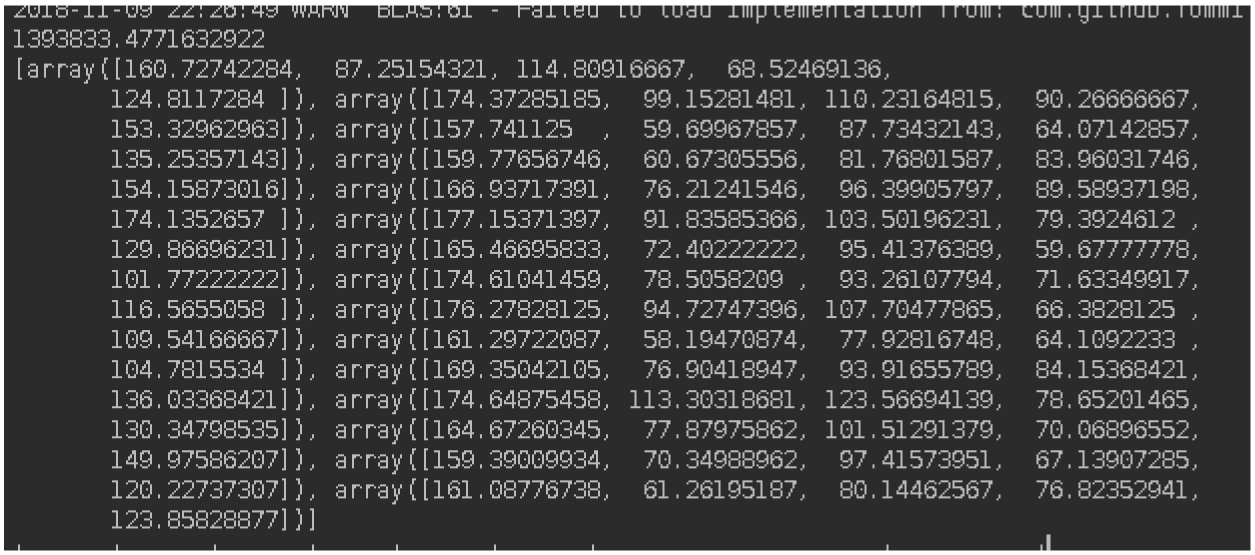
R2.1
```
patients =
spark.read.csv("/home/nima/code/cs626/hw3/src/main/resources/patients3.csv",header=True,infe
rSchema=True)
assembler =
VectorAssembler(inputCols=['height','weight','waist','diasbp','systbp'],outputCol='features'
)
patients_feats = assembler.transform(patients)
kmeans = KMeans(maxIter=100, k=15)
model = kmeans.fit(patients_feats)
```

R2.2
```
costs = model.computeCost(patients_feats)
print(costs)
```

R2.3
```
centers = model.clusterCenters()
print(centers)
```

R2.4
```
transformed = model.transform(patients_feats)
transformed.show()
```

Screenshot with within-set sum of squared errors (number at the top) and cluster centers (arrays):



```
2018-11-09 22:26:49 WARN  BLAS:61 - Failed to load implementation from: com.github.fommil
1393833.4771632922
[array([160.72742284,  87.25154321, 114.80916667,  68.52469136,
       124.8117284 ]), array([174.37285185,  99.15281481, 110.23164815,  90.26666667,
       153.32962963]), array([157.741125  ,  59.69967857,  87.73432143,  64.07142857,
       135.25357143]), array([159.77656746,  60.67305556,  81.76801587,  83.96031746,
       154.15873016]), array([166.93717391,  76.21241546,  96.39905797,  89.58937198,
       174.1352657 ]), array([177.15371397,  91.83585366, 103.50196231,  79.3924612 ,
       129.86696231]), array([165.46695833,  72.40222222,  95.41376389,  59.67777778,
       101.77222222]), array([174.61041459,  78.5058209 ,  93.26107794,  71.63349917,
       116.5655058 ]), array([176.27828125,  94.72747396, 107.70477865,  66.3828125 ,
       109.54166667]), array([161.29722087,  58.19470874,  77.92816748,  64.1092233 ,
       104.7815534 ]), array([169.35042105,  76.90418947,  93.91655789,  84.15368421,
       136.03368421]), array([174.64875458, 113.30318681, 123.56694139,  78.65201465,
       130.34798535]), array([164.67260345,  77.87975862, 101.51291379,  70.06896552,
       149.97586207]), array([159.39009934,  70.34988962,  97.41573951,  67.13907285,
       120.22737307]), array([161.08776738,  61.26195187,  80.14462567,  76.82352941,
       123.85828877])]
```