一、环境配置

1.1 Hadoop

```
root@hadoop01:~# start-all.sh
Starting namenodes on [101.42.45.176]
Starting datanodes
Starting secondary namenodes [hadoop02]
Starting resourcemanager
Starting nodemanagers
root@hadoop01:~# jps
11522 DataNode
12103 NodeManager
12279 Jps
```

1.2 Hive

metastore

```
root@hadoop01:~# 2023-01-16 19:20:27: Starting Hive Metastore Server
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.1.3/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
```

hiveserver2

```
root@hadoop01:~# hive --service hiveserver2 &
[1] 6470
root@hadoop01:~# 2023-01-16 19:20:39: Starting HiveServer2
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.1.3/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 986a0d6e-665f-4cca-8879-41d27c93ea9c
Hive Session ID = 17f085be-8bad-4855-bcb9-d57bac42367a
Hive Session ID = 5c2b7ca6-6d35-4a81-86ca-05c8457705b8
Hive Session ID = a07d68d2-1008-4c4f-abfe-eb558dc5b23e
OK
```

二、导入数据

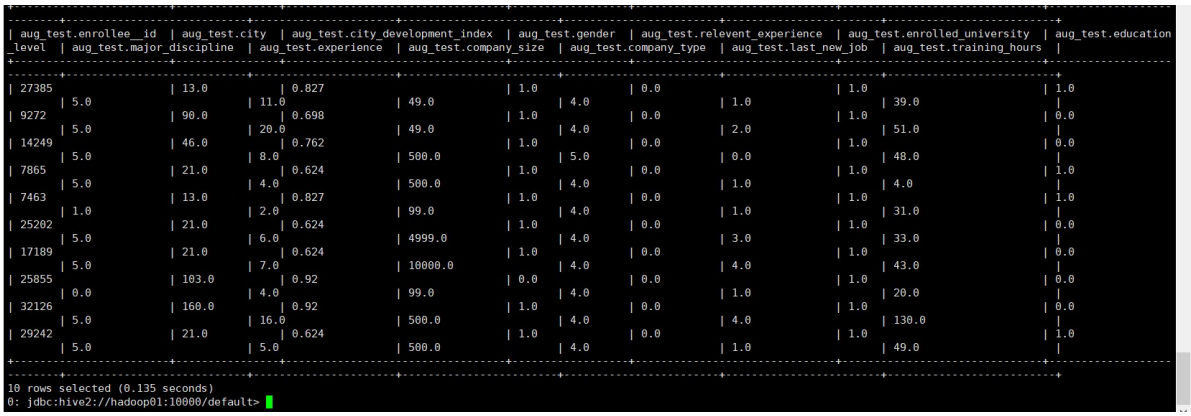
2.1 建表语句

```sql
CREATE TABLE `human`.`aug_train` (
    `enrollee__id` INT,
    `city` DOUBLE,
     `city_development_index` DOUBLE,
    `gender` DOUBLE,
    `relevent_experience` DOUBLE,
    `enrolled_university` DOUBLE,
    `education_level` DOUBLE,
    `major_discipline` DOUBLE,
    `experience` DOUBLE,
     `company_size` DOUBLE,
     `company_type` DOUBLE,
    `last_new_job` DOUBLE,
   `training_hours` DOUBLE, `target` DOUBLE)
    ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
   STORED AS TEXTFILE;
```

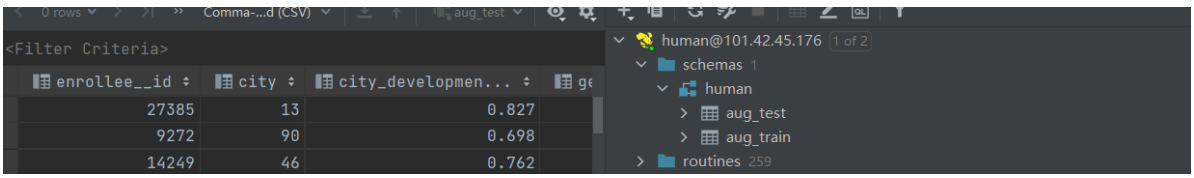
```
CREATE TABLE `human`.`aug_test` (
    `enrollee__id` INT,
    `city` DOUBLE,
     `city_development_index` DOUBLE,
    `gender` DOUBLE,
    `relevent_experience` DOUBLE,
    `enrolled_university` DOUBLE,
    `education_level` DOUBLE,
    `major_discipline` DOUBLE,
    `experience` DOUBLE,
     `company_size` DOUBLE,
     `company_type` DOUBLE,
    `last_new_job` DOUBLE,
   `training_hours` DOUBLE)
    ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
   STORED AS TEXTFILE;
```

2.2 导入数据

```
load data local inpath '/usr/local/aug_test.csv' overwrite into table aug_test
load data local inpath '/usr/local/aug_train.csv' overwrite into table aug_train
```



2.3 测试idea连接



三、SparkMllib建模

3.1 连接hive，读取数据

```
    var data_test = spark.read.table("human.aug_test")
    var data_train_source = spark.read.table("human.aug_train")
```

3.2 提取需要的字段

```
    var df =
data_train_source.select("target","city","city_development_index","gender",
 "relevent_experience","enrolled_university","education_level","major_discipl
ine"
,"experience","company_size","company_type","last_new_job","training_hours")
```

3.3 转成labeledPoint类型

```
      var labeledPointRDD = df.rdd.map(row=>{
        var label = row.getAs[Double]("target")
        var city = row.getAs[Double]("city")
        var city_development_index = row.getAs[Double]
("city_development_index")
        var gender = row.getAs[Double]("gender")
        var relevent_experience = row.getAs[Double]("relevent_experience")
        var enrolled_university = row.getAs[Double]("enrolled_university")
        var education_level = row.getAs[Double]("education_level")
        var major_discipline = row.getAs[Double]("major_discipline")
        var experience = row.getAs[Double]("experience")
        var company_size = row.getAs[Double]("company_size")
        var company_type = row.getAs[Double]("company_type")
        var last_new_job = row.getAs[Double]("last_new_job")
        var training_hours = row.getAs[Double]("training_hours")

        LabeledPoint(label,Vectors.dense(city,city_development_index,gender
,relevent_experience,enrolled_university,education_level,major_discipline,
          experience,company_size,company_type,last_new_job,training_hours))
      })
```

3.4 持久化

```
labeledPointRDD.cache()
```

3.5 建模部分

3.5.1 支持向量机

```
var svmModel:SVMModel = SVMWithSGD.train(trainRDD,100)

var svmPredictAndActualRDD:RDD[(Double,Double)] = testRDD.map{
  case LabeledPoint(label,features)=>(svmModel.predict(features),label)
}
// 评价roc
var svmMetrics = new BinaryClassificationMetrics(svmPredictAndActualRDD)
var svmRoc = svmMetrics.areaUnderROC()

val nbTotalCorrect = testRDD.map { point =>
  if (svmModel.predict(point.features) == point.label) 1 else 0
}.sum
```

```
val numData = testRDD.count()
val svmpre = nbTotalCorrect/numData
```

3.5.2 逻辑回归

```
var lr = new LogisticRegressionWithLBFGS().setNumClasses(2)

var lrModel:LogisticRegressionModel = lr.run(trainRDD)
var lrPredictAndActualRDD:RDD[(Double,Double)] = testRDD.map{
  case LabeledPoint(label,features)=>(lrModel.predict(features),label)
}
var lrMetrics = new BinaryClassificationMetrics(lrPredictAndActualRDD)
var lrRoc = lrMetrics.areaUnderROC()

val LrnbTotalCorrect = testRDD.map { point =>
  if (lrModel.predict(point.features) == point.label) 1 else 0
}.sum
val LrnumData = testRDD.count()
val LrAuc = LrnbTotalCorrect/LrnumData
```

3.5.3 决策树

```
var dtModel:DecisionTreeModel = DecisionTree.trainClassifier(trainRDD,2
    ,Map[Int,Int](),"gini",6,2)

    var dtPredictAndActualRDD:RDD[(Double,Double)] = testRDD.map{
      case LabeledPoint(label,features) =>(dtModel.predict(features),label)
    }
    val dtMetrics = new BinaryClassificationMetrics(dtPredictAndActualRDD)
    val dtRoc = dtMetrics.areaUnderROC()

    val DtnbTotalCorrect = testRDD.map { point =>
      if (dtModel.predict(point.features) == point.label) 1 else 0
    }.sum
    val DtnumData = testRDD.count()
    val Dtauc = DtnbTotalCorrect/DtnumData
```

3.5.4 朴素贝叶斯

```
    var nbModel = NaiveBayes.train(trainRDD,1.0)
    var nbPredictAndActualRDD:RDD[(Double,Double)] = testRDD.map{
      case LabeledPoint(label,features)=>(nbModel.predict(features),label)
    }
    var nbMetrics = new BinaryClassificationMetrics(nbPredictAndActualRDD)
    val nbRoc = nbMetrics.areaUnderROC()

    val NbnbTotalCorrect = testRDD.map { point =>
      if (nbModel.predict(point.features) == point.label) 1 else 0
    }.sum
    val NbnumData = testRDD.count()
    var nbauc = NbnbTotalCorrect/NbnumData
```

运行结果

```
"C:\Program Files\Java\jdk1.8.0_131\bin\java.exe" "-javaagent:F:\IntelliJ IDEA 20
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/F:/maven_repository/org/slf4j/slf4j-log4j12/1.
SLF4J: Found binding in [jar:file:/F:/maven_repository/org/apache/logging/log4j/l
SLF4J: Found binding in [jar:file:/F:/maven_repository/ch/qos/logback/logback-cla
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
支持向量机 准确度 = 0.8022508038585209
支持向量机 ROC = 0.5140573284206582

逻辑回归 准确度 = 0.8303858520900321
逻辑回归 ROC = 0.6016299256853807

决策树 准确度 = 0.8062700964630225
决策树 ROC = 0.49900497512437814

朴素贝叶斯 准确度 = 0.6254019292604501
朴素贝叶斯 ROC = 0.4954932450717126
```

1 我们发现:四个模型的ROC都接近0.5，其中决策树的准确度和roc都是不错的

3.6 模型评估

为了避免模型偶然性造成的损失，我们对每个模型进行10次训练求每个模型的平均准确度和ROC

3.6.1 支持向量机

```
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/apache/logging/log4j/log4j-slf4j-im
SLF4J: Found binding in [jar:file:/root/.m2/repository/ch/qos/logback/logback-classic/1.2.3/lo
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
支持向量机10次平均 ROC = 0.5
支持向量机10次平均准确度 = 0.815970386039133

Process finished with exit code 0
```

3.6.2 逻辑回归

```
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/slf4j/slf4j-log4j12/1.7.30
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/apache/logging/log4j/log4j
SLF4J: Found binding in [jar:file:/root/.m2/repository/ch/qos/logback/logback-classic
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
逻辑回归10次平均 ROC = 0.5276959860673766
逻辑回归10次平均准确度 = 0.8125714285714286

Process finished with exit code 0
```

3.6.3 决策树

```
/usr/java/jdk1.8.0_281-amd64/bin/java ...
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/slf4j/slf4
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/apache/log
SLF4J: Found binding in [jar:file:/root/.m2/repository/ch/qos/logback
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an e
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
决策树10次平均 ROC = 0.6077480009265518
决策树10次平均准确度 = 0.8080752212389382
```

3.6.4 朴素贝叶斯

```
/usr/java/jdk1.8.0_281-amd64/bin/java ...
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/slf4j/s
SLF4J: Found binding in [jar:file:/root/.m2/repository/org/apache/
SLF4J: Found binding in [jar:file:/root/.m2/repository/ch/qos/logb
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for a
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactor
朴素贝叶斯10次平均 ROC = 0.505493627151857
朴素贝叶斯10次平均准确度 = 0.6243469174503656

Process finished with exit code 0
```

1 我们发现：经过10次模型训练，朴素贝叶斯不适合上述数据集，其他三个均取得不错的效果

3.7 模型优化：决策树

- maxDepth：决策树最大深度
- maxBins：决策树每个节点的最大分支数

根据决策树的最大深度和每个节点的最大分支数进行调参，每个参数运行30次求准确度的平均值

```
1 "C:\Program Files\Java\jdk1.8.0_131\bin\java.exe" "-javaagent:F:\IntelliJ
  IDEA 2020.3.1\lib\idea_rt.jar=56251:F:\IntelliJ IDEA 2020.3.1\bin" -
  Dfile.encoding=UTF-8 -classpath
  C:\Users\DELL\AppData\Local\Temp\classpath61966627.jar
  com.program.DTModelPlus
2 SLF4J: Class path contains multiple SLF4J bindings.
3 SLF4J: Found binding in [jar:file:/F:/maven_repository/org/slf4j/slf4j-
  log4j12/1.7.30/slf4j-log4j12-
  1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
4 SLF4J: Found binding in
  [jar:file:/F:/maven_repository/org/apache/logging/log4j/log4j-slf4j-
  impl/2.10.0/log4j-slf4j-impl-
  2.10.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]
```

```
SLF4J: Found binding in [jar:file:/F:/maven_repository/ch/qos/logback/logback-classic/1.2.3/logback-classic-1.2.3.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
maxDepth is : 3 maxBins is : 2
 决策树30次平均准确度 = 0.8125530110262933

maxDepth is : 3 maxBins is : 3
 决策树30次平均准确度 = 0.8142493638676845

maxDepth is : 3 maxBins is : 4
 决策树30次平均准确度 = 0.8320610687022904

maxDepth is : 3 maxBins is : 5
 决策树30次平均准确度 = 0.8371501272264628

maxDepth is : 3 maxBins is : 6
 决策树30次平均准确度 = 0.8439355385920277

maxDepth is : 4 maxBins is : 2
 决策树30次平均准确度 = 0.8125530110262933

maxDepth is : 4 maxBins is : 3
 决策树30次平均准确度 = 0.815945716709076

maxDepth is : 4 maxBins is : 4
 决策树30次平均准确度 = 0.8227311280746398

maxDepth is : 4 maxBins is : 5
 决策树30次平均准确度 = 0.8227311280746398

maxDepth is : 4 maxBins is : 6
 决策树30次平均准确度 = 0.8396946564885501

maxDepth is : 5 maxBins is : 2
 决策树30次平均准确度 = 0.8125530110262933

maxDepth is : 5 maxBins is : 3
 决策树30次平均准确度 = 0.8193384223918575

maxDepth is : 5 maxBins is : 4
 决策树30次平均准确度 = 0.8210347752332484

maxDepth is : 5 maxBins is : 5
 决策树30次平均准确度 = 0.8269720101781174

maxDepth is : 5 maxBins is : 6
 决策树30次平均准确度 = 0.8379983036471581

maxDepth is : 6 maxBins is : 2
 决策树30次平均准确度 = 0.8100084817642069

maxDepth is : 6 maxBins is : 3
```

```
 决策树30次平均准确度 = 0.8184902459711622

maxDepth is : 6 maxBins is : 4
 决策树30次平均准确度 = 0.8320610687022904

maxDepth is : 6 maxBins is : 5
 决策树30次平均准确度 = 0.8329092451229851

maxDepth is : 6 maxBins is : 6
 决策树30次平均准确度 = 0.8337574215436805

maxDepth is : 7 maxBins is : 2
 决策树30次平均准确度 = 0.8057675996607294

maxDepth is : 7 maxBins is : 3
 决策树30次平均准确度 = 0.8074639525021208

maxDepth is : 7 maxBins is : 4
 决策树30次平均准确度 = 0.8235793044953351

maxDepth is : 7 maxBins is : 5
 决策树30次平均准确度 = 0.8320610687022904

maxDepth is : 7 maxBins is : 6
 决策树30次平均准确度 = 0.825275657336726


Process finished with exit code 0

```