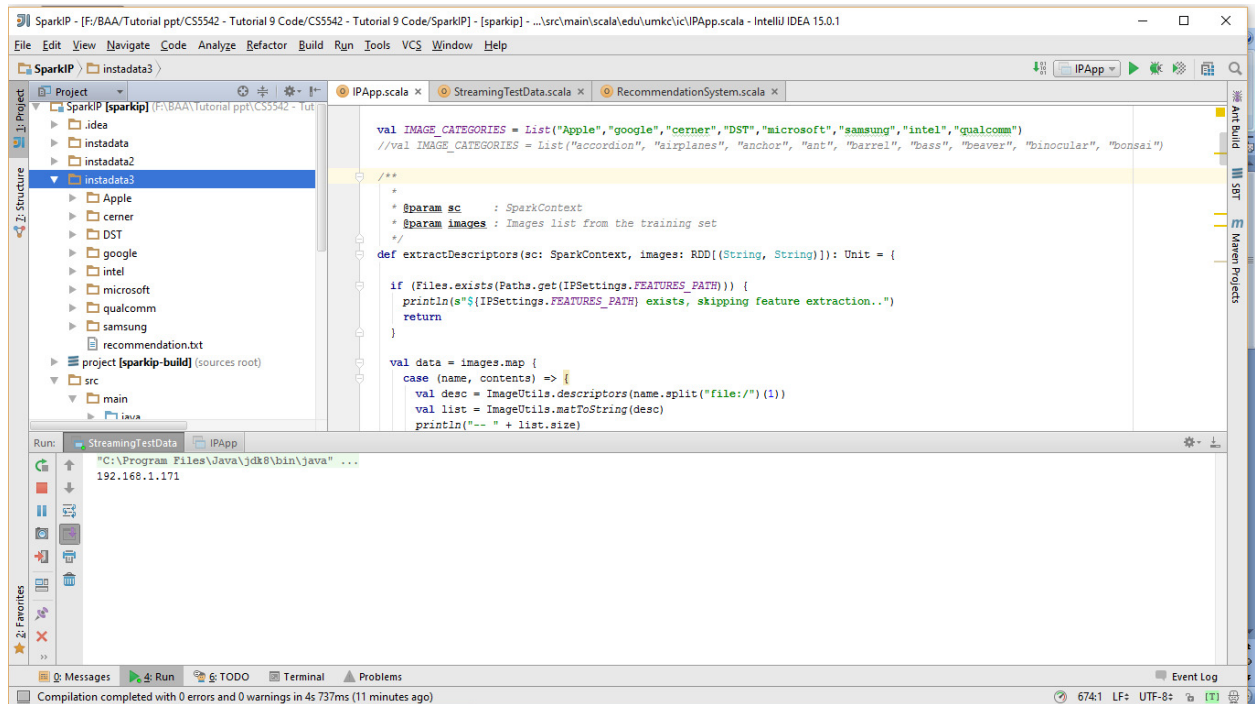
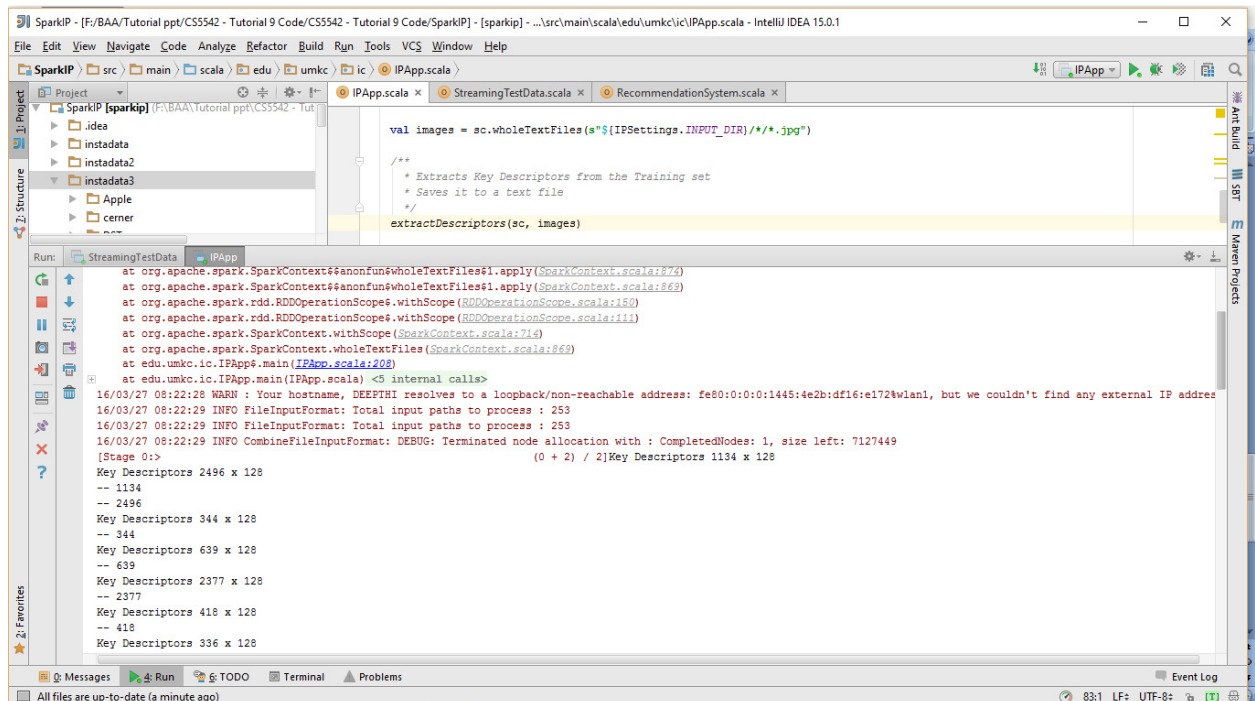
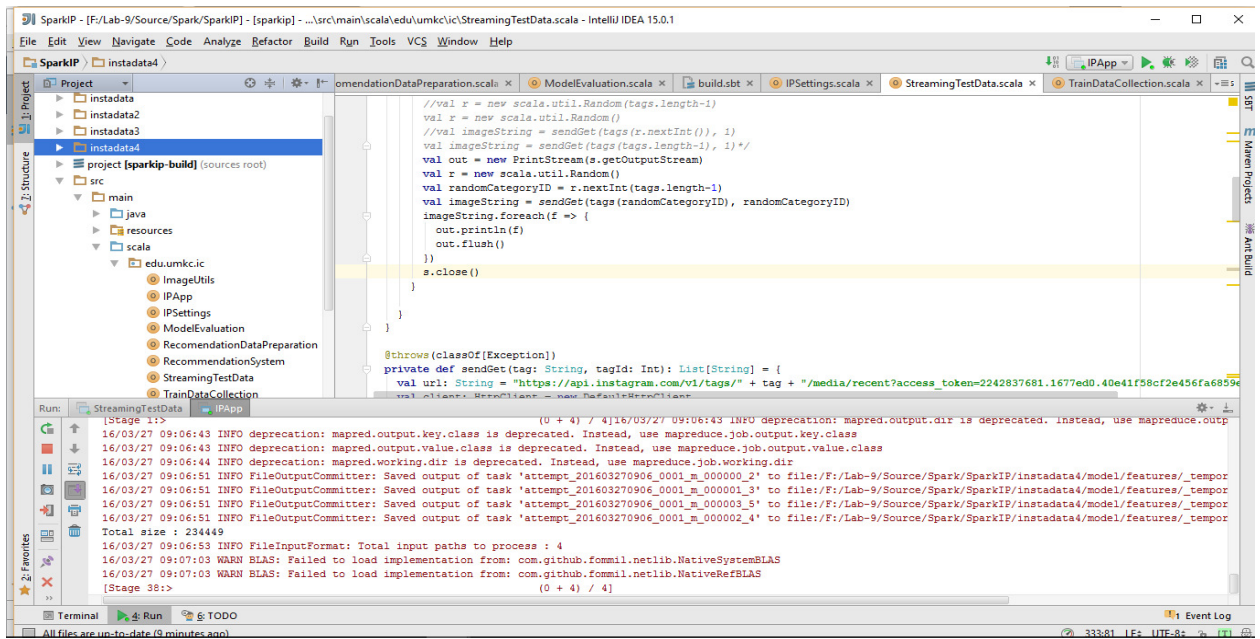


Live stream Data to predict the current image for the particular tags:

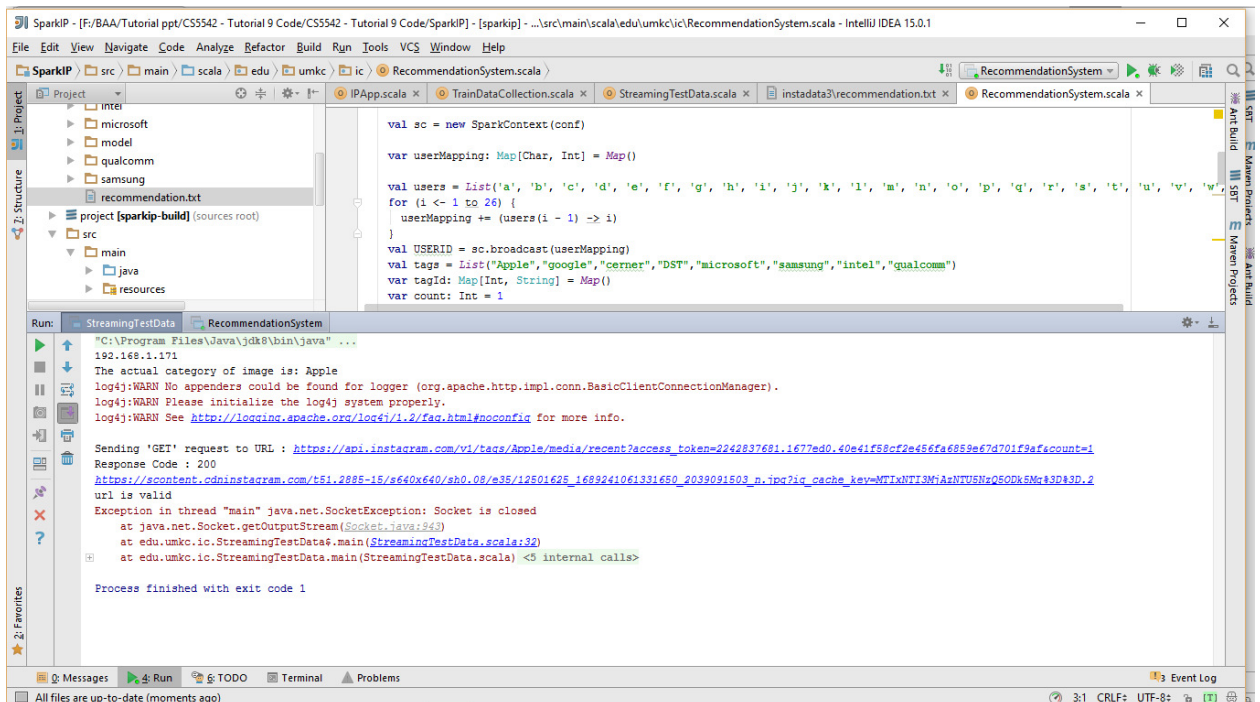


2. Image Classification based on categories:





Actual category of image:



Predicted category of image:

```

SparkIP - [F:\Lab-9\Source\Spark\SparkIP] - [sparkip] - ...src\main\scala\edu\umkc\ic\IPApp.scala - IntelliJ IDEA 15.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

SparkIP | src | main | scala | edu | umkc | ic | IPApp.scala
Project: SparkIP
Structure: intel, microsoft, model, clusters, features, histograms, nbmodel, qualcomm, samsung, recommendation.txt
Run: StreamingTestData, IPApp

object IPApp {
  val featureVectorsCluster = new mutable.MutableList[String]
  val IMAGE_CATEGORIES = List("Apple", "google", "cerner", "DST", "microsoft", "samsung", "intel", "qualcomm")
  //val IMAGE_CATEGORIES = List("rice", "tempura", "toast", "Bibimap", "sushi", "spaghetti", "sausage", "oden", "cmelet", "jiaozi")
  //val IMAGE_CATEGORIES = List("accordion", "airplanes", "anchor", "ant", "barrel", "bass", "beaver", "binocular", "bonsai")
}

Predicting test image : google
Time: 1459092148000 ms

```

```

SparkIP - [F:\Lab-9\Source\Spark\SparkIP] - [sparkip] - ...src\main\scala\edu\umkc\ic\RecommendationSystem.scala - IntelliJ IDEA 15.0.1
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

SparkIP | src | main | scala | edu | umkc | ic | RecommendationSystem.scala
Project: SparkIP
Structure: main, java, resources, scala
Run: StreamingTestData, IPApp

var userMapping: Map[Char, Int] = Map()
val users = List('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w')
for (i <- 1 to 26) {
  userMapping += (users(i - 1) -> i)
}

Predicting test image : google
Time: 1459092148000 ms
Time: 1459092150000 ms

```


3. Image Recommendation:

The screenshot shows the IntelliJ IDEA interface with the `RecommendationSystem.scala` file open. The code defines a Spark application that generates recommendations based on a user mapping and a list of companies. The Run console displays the output of the application.

```

val sc = new SparkContext(conf)
var userMapping: Map[Char, Int] = Map()

val users = List('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w')
for (i <- 1 to 26) {
  userMapping += (users(i - 1) -> i)
}

val USERID = sc.broadcast(userMapping)
val tags = List("Apple", "google", "cerner", "DST", "microsoft", "samsung", "intel", "qualcomm")
var tagId: Map[Int, String] = Map()
var count: Int = 1
  
```

Run Output:

```

2.0
Companies recommended to you:
Rating(1,8,2.5321572872855556)
1: qualcomm
Rating(1,8,2.5321572872855556)
2: qualcomm
Rating(1,1,2.725021763621964)
3: Apple
Rating(1,5,2.5667114464428)
4: microsoft
Rating(1,6,2.4852530818947196)
5: samsung
Rating(1,6,2.4852530818947196)
6: samsung
Rating(1,6,2.4852530818947196)
7: samsung
Rating(1,2,2.472655850609989)
8: google
Rating(1,2,2.472655850609989)
9: google
Rating(1,2,2.472655850609989)
10: google
  
```

The screenshot shows the IntelliJ IDEA interface with the `StreamingTestData.scala` file open. The code defines a simple HTTP server that responds to requests with a random recommendation from a predefined list. The Run console displays the output of the server.

```

def main(args: Array[String]) {
  val tags = List("Apple", "google", "cerner", "DST", "microsoft", "samsung", "intel", "qualcomm")
  val server = new ServerSocket(1001)
  println(InetAddress.getLocalHost.getHostAddress)
  while (true) {
    val s = server.accept()
    while (s.isConnected) {
      val out = new PrintStream(s.getOutputStream)
      val r = new scala.util.Random
      val randomCategoryID = r.nextInt(tags.length-1)
      println("The actual category of image is: " + tags(randomCategoryID))
    }
  }
}
  
```

Run Output:

```

Companies recommended to you:
Rating(1,4,2.399741505166628)
1: DST
Rating(1,4,2.399741505166628)
2: DST
Rating(1,8,2.464784211743371)
3: qualcomm
Rating(1,8,2.464784211743371)
4: qualcomm
Rating(1,1,2.144053946041083)
5: Apple
Rating(1,1,2.144053946041083)
6: Apple
Rating(1,1,2.144053946041083)
7: Apple
Rating(1,1,2.144053946041083)
8: Apple
Rating(1,6,2.2963357258022716)
9: samsung
Rating(1,6,2.2963357258022716)
10: samsung
  
```

4. Mobile Recommendation through smartphone

