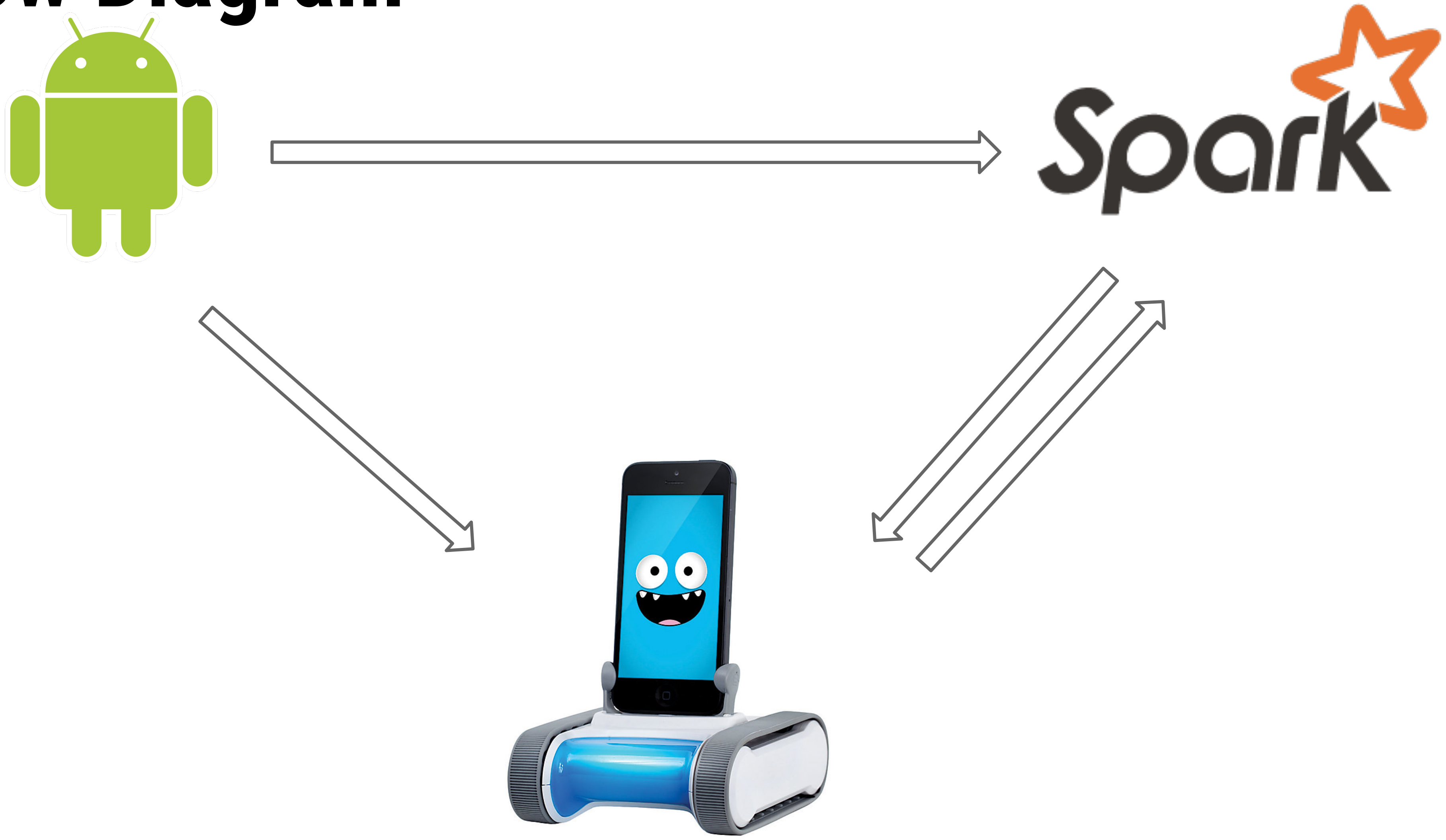
Full WorkFlow

CS590BD: Big Data Analytics and Apps

Flow Diagram



Topics Covered

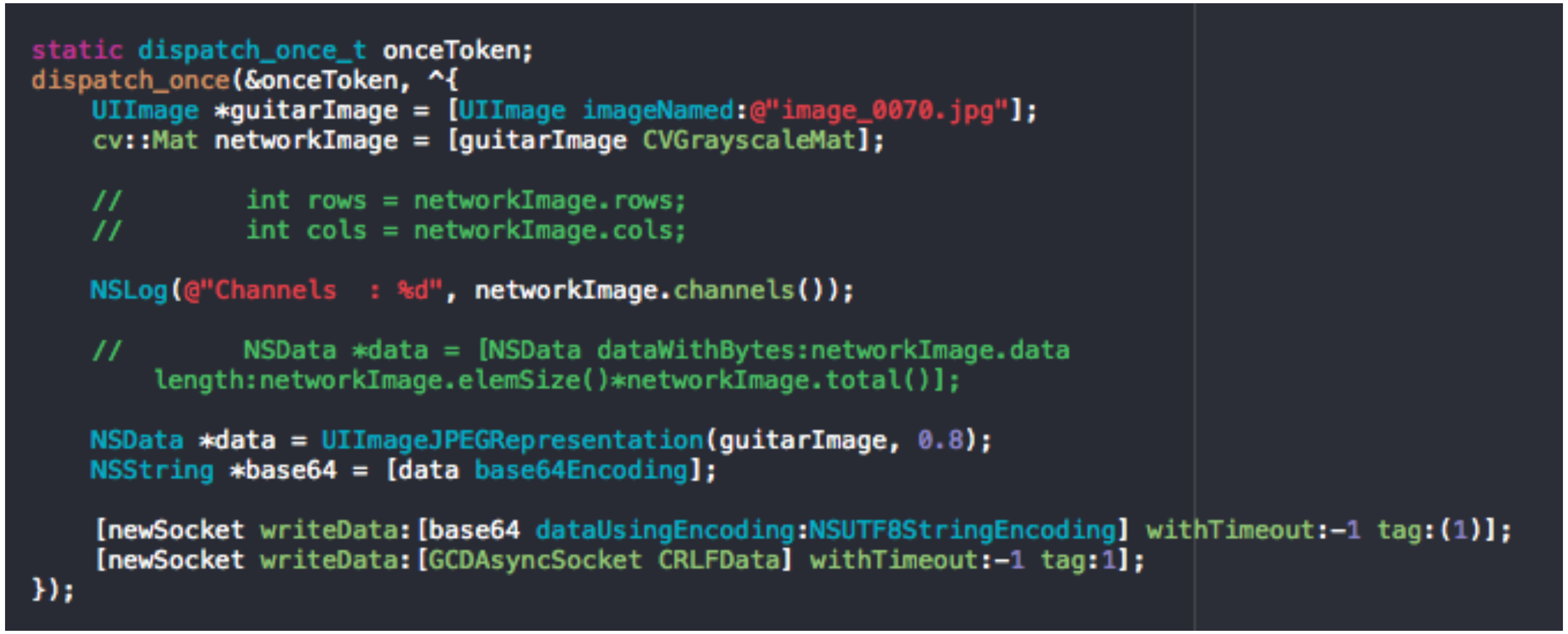
- Image Classification using Streaming Input from iOS
- Text Classification using Streaming Input from Android and Command given to iOS
- Recommendation using Streaming Input from Android
- Sentiment Analysis using Stanford Core NLP

Image Classification using Streaming Input from iOS

Image Classification with Streaming

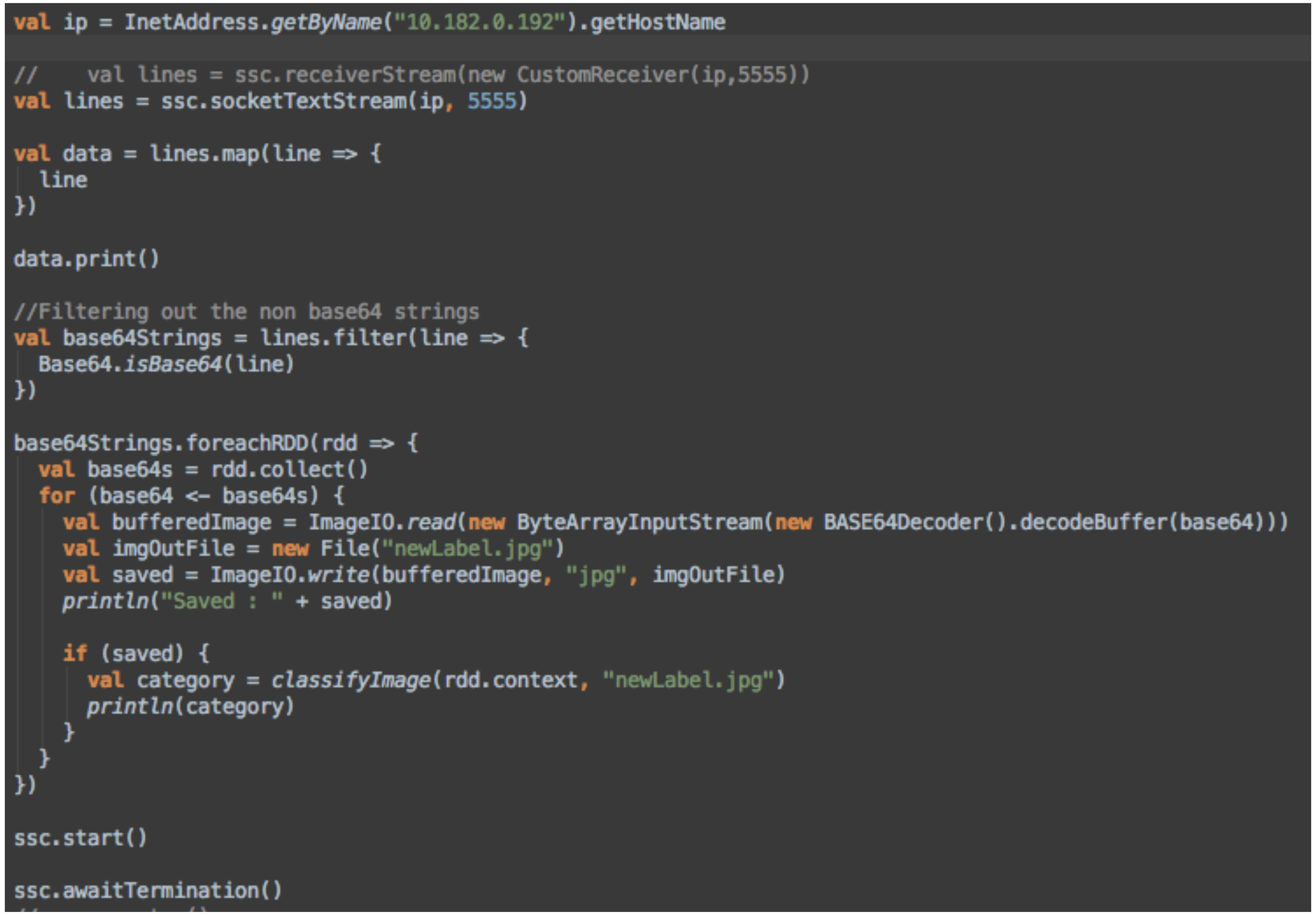
- Image is sent from the Client (Base64 format)
- The Base64 image is saved as a jpg on Spark server.
- This image is sent for the classification

Sending image from iOS



```
static dispatch_once_t onceToken;
dispatch_once(&onceToken, ^{
    UIImage *guitarImage = [UIImage imageNamed:@"image_0070.jpg"];
    cv::Mat networkImage = [guitarImage CVGrayscaleMat];

    //        int rows = networkImage.rows;
    //        int cols = networkImage.cols;

    NSLog(@"Channels  : %d", networkImage.channels());

    //        NSData *data = [NSData dataWithBytes:networkImage.data
        length:networkImage.elemSize()*networkImage.total()];

    NSData *data = UIImageJPEGRepresentation(guitarImage, 0.8);
    NSString *base64 = [data base64Encoding];

    [newSocket writeData:[base64 dataUsingEncoding:NSUTF8StringEncoding] withTimeout:-1 tag:(1)];
    [newSocket writeData:[GCDAsyncSocket CRLFData] withTimeout:-1 tag:1];
});
```

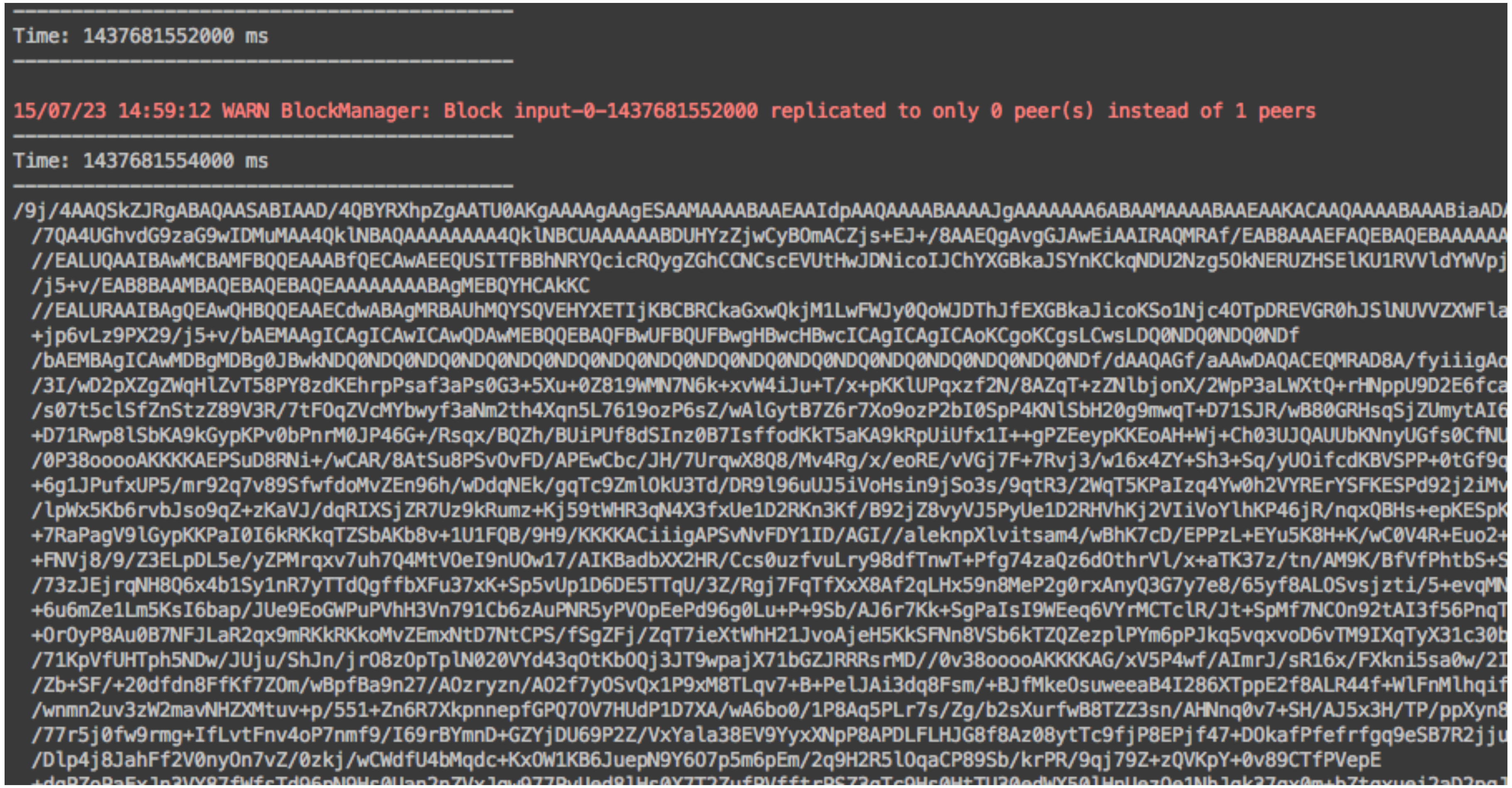
Receiving image on Spark



```
val ip = InetAddress.getByName("10.182.0.192").getHostName

//    val lines = ssc.receiverStream(new CustomReceiver(ip,5555))
val lines = ssc.socketTextStream(ip, 5555)

val data = lines.map(line => {
  line
})

data.print()

//Filtering out the non base64 strings
val base64Strings = lines.filter(line => {
  Base64.isBase64(line)
})

base64Strings.foreachRDD(rdd => {
  val base64s = rdd.collect()
  for (base64 <- base64s) {
    val bufferedImage = ImageIO.read(new ByteArrayInputStream(new BASE64Decoder().decodeBuffer(base64)))
    val imgOutFile = new File("newLabel.jpg")
    val saved = ImageIO.write(bufferedImage, "jpg", imgOutFile)
    println("Saved : " + saved)

    if (saved) {
      val category = classifyImage(rdd.context, "newLabel.jpg")
      println(category)
    }
  }
})

ssc.start()

ssc.awaitTermination()
```

Output

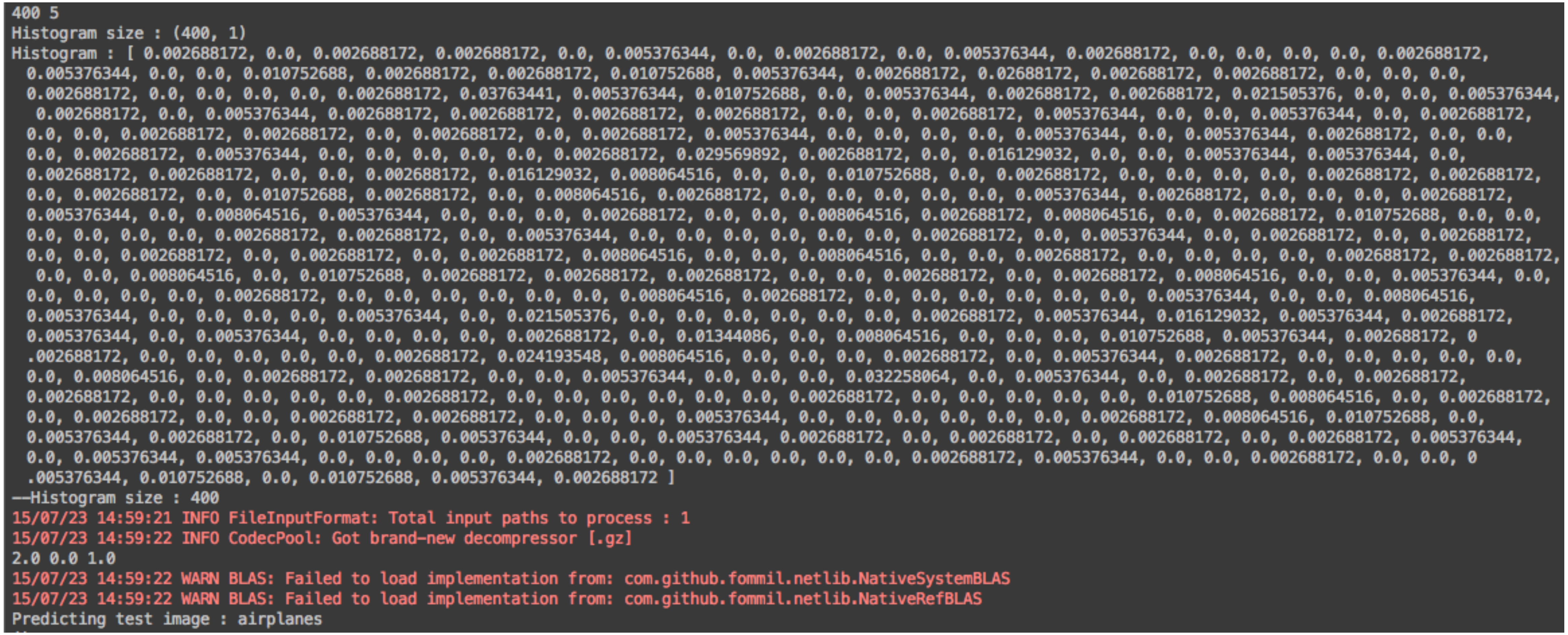
Base64 from the iOS



```
-------------------------------------------
Time: 1437681552000 ms
-------------------------------------------

15/07/23 14:59:12 WARN BlockManager: Block input-0-1437681552000 replicated to only 0 peer(s) instead of 1 peers
-------------------------------------------
Time: 1437681554000 ms
-------------------------------------------
```

```
/9j/4AAQSkZJRgABAQAASABIAAD/4QBYRXhpZgAATU0AKgAAAAgAAgESAAMAAAABAAEAAIdpAAQAAAABAAAAJgAAAAAAA6ABAAMAAAABAAEAAKACAAQAAAABAAABiaADA
  /7QA4UGhvdG9zaG9wIDMuMAA4QklNBAQAAAAAAAA4QklNBCUAAAAAABDUHYzZjwCyBOmACZjs+EJ+/8AAEQgAvgGJAwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAA
  //EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpj
  /j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC
  //EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFla
  +jp6vLz9PX29/j5+v/bAEMAAgICAgICAwICAwQDAwMEBQQEBAQFBwUFBQUFBwgHBwcHBwcICAgICAgICAoKCgoKCgsLCwsLDQ0NDQ0NDQ0NDf
  /bAEMBAgICAwMDBgMDBg0JBwkNDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDQ0NDf/dAAQAGf/aAAwDAQACEQMRAD8A/fyiiigAo
  /3I/wD2pXZgZWqHlZvT58PY8zdKEhrpPsaf3aPs0G3+5Xu+0Z819WMN7N6k+xvW4iJu+T/x+pKKlUPqxzf2N/8AZqT+zZNlbjonX/2WpP3aLWXtQ+rHNppU9D2E6fca
  /s07t5clSfZnStzZ89V3R/7tFOqZVcMYbwyf3aNm2th4Xqn5L7619ozP6sZ/wAlGytB7Z6r7Xo9ozP2bI0SpP4KNlSbH20g9mwqT+D71SJR/wB80GRHsqSjZUmytAI6
  +D71Rwp8lSbKA9kGypKPv0bPnrM0JP46G+/Rsqx/BQZh/BUiPUf8dSkKA9kRpUiUfx1I++gPZEeypKEoAH+Wj+Ch03UJQAUbKNnyUGfs0CfNU
  /0P38ooooAKKKKAEPSuD8RNi+/wCAR/8AtSu8PSvOvFD/APEwCbc/JH/7UrqwX8Q8/Mv4Rg/x/eoRE/vVGj7F+7Rvj3/w16x4ZY+Sh3+Sq/4yUOifcdKBVSPP+0tGf9q
  +6g1JPufxUP5/mr92q7v89SfwfdoMvZ1En96h/wDdqNEk/gqTc9Z2kyzVaIz4W9F1ZtrfwfVIq5Wi/ZjHzVp2iMv93IEaKjDT4/9Ot+lxZaDp92j2iMv
  /lpWx5Kb6rvbJso9qZ+zKaVJ/dqRIXXSjZR7Uz9kRumz+Kj59tWHWHR3qN4X3fxlhKP46jR/nqxQBHs+Sj5vkrepYV5PyUe1D2RHRHVhkIoYlhKP46jR/nqxQBHs+Sj5vkrpKESpK
  +7RaPagV9lGypKPaI0I6kRKaTZSbAKb8v+1U1FQB/9H9H9/KKKKACiiigAPSvzNvFDY1ID/APGI/wBhK7cD/Eg8/Mv4Rg/x/eoRE/vVGj7F+7Rvj3/w16x4ZY+Sh3+Sq/4yUOifcdKBVSPP+0tGf9q
  +FNVj8/9/Z3ELpDL5e/yPZMrqxv7uh7Q7AVTeF/Vs9ZhPBWZhe1q6+W8Ns8iffUjfPrh/S0Q+Jw+B4jCaR4p1ccthtN4W6oNX/ALfjU76HNCtAfNK/BfVfPhtbS+S
  /73zJEjrqNH8Q6x4b1Sy1nR7yTTdQgffU5TTqU/3Z/Rgj7FqTfXxX8Af2qLHx59n8MeP2g0rxAb7jyAnyTkH3AvL3zj+VVXSK+/5+evqMN/Bkc23vkjYwYg9Ltw5u3wGUqs5vt1xNXyA3dCbVWRXu/+/kcCCu4I1X6/RC/AIyPz7y/d/tc4UJ8ePqLWWMWHPWJ5kBdXWRn/w5O52CXq5+ZY9sBz/ItqiiiCoQrq
  +6u6Zejdp6j7rbZmUrWjvuvq/xxWq/9pm/xXBPKTX/N/yUoGi7Lnr9qr9t49I0V3SO28b7dVgtGgLEDRfD/fXwfq51rA7EQXynwR/PmqhPeIkBj8Sq/w/zRJTZnZ9XWxB4bLj/5KdzLmp9oc9jHrfcy85/CD8Sdq/+9bX1qF/eVn4V7/
  /73zJEjrqNH8Q6x4b1Sy1nR7yTTdQgffbXFu37xK+Sp5vUp1D6Dz8evqMN/ZZf/65yf8ALOSs8dAuBdtTo5wlOWMIkBHb0Ar7jYXdy4Or3G7ywHRaWdZHd3rf1B7
  +6u6mZe1Lm5KsI6bap/JUe9EoGWPuVhH3Vn791zZEn96h/wDdqNEk/gqTc9Z2kyzVaIz4W9F1ZtrfwfVIq5Wi/ZjHzVp2iMv93IEaKjDT4/9Ot+lxZaDp92j2iMv
  +0rOyP8Au0B7NFJLaR2qx9mRKkRKkoMvZBHzVp2iMv93IEaKjDT4/9Ot+lxZaDp92j2iMv93IEaKjDT4/9Ot+lxZaDp92j2iMv93IEaKjDT4/9Ot+lxZaDp92j2iMv
  /71KpVfUHTph5NDw/JUju/ShJn/jr08zTplN020VYd43q0tKbOQj3JT9wpajX71bGZJRRsrMD//vx4oooAKKKKAG/xV5P4wr/AI16d
  /Zb+SF/+20dfdn8FfKf7ZOm/wBpa9zryzn/AOzaJzzWMB6ia/w6ABOsvQx1P9xM8TLqv7+Pq3+B+PelJAi3dq8Fsm/+BJfMkfPwgkwvoaFseKR4fB5KqULBGH5U1hqi
  /wnmn2uv3zW2mavNHZ/551+Zn6R7Xdk7ZKpwFm8aGq0V7HUdP1D7XA/+/wA5rt7hPP7PHonfZ87B4A1S92C0E00f7T0P9E7D0ZF1m7u9Rz0kPw7ShdphqQSsTPpxsV
  /77r5j0fw9mg+IfLvtFnv4oP7nmfB9YJDUd69P2Z/9EPf47p2IMV96oNUdOqPHAgwOFHZQEHRw4lxb7vPm2dq4yNvTN1YGsHu8HVcjWZ3f9jHu8EzWt4+r9UpuPEKvDPtaZ
  /Dlp4j8JahFf2V0nyOn7vZ/0zkj/wCWdfU4bMqdc+Kx0W1KB6JuepN9Y67p5m6pEm/2q9H2R5z15CP89SeSb/krkrPRR/9qj79Z+ZaY+0vZePVep
  +dqBZoBaFxJn3VY87fWfcTd96pN9Hs0Uan2nZVxJgw977RvUed8lHs0X7T27ufBVfftrRS73qTc9Hs0HtTU30edWY50lHnUezOe1NbJqk37qx0m+b7tqxuei2aD2pqJ
```

Output : Prediction



```
400 5
Histogram size : (400, 1)
Histogram : [ 0.002688172, 0.0, 0.002688172, 0.002688172, 0.0, 0.005376344, 0.0, 0.002688172, 0.0, 0.005376344, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.002688172,
  0.005376344, 0.0, 0.0, 0.010752688, 0.002688172, 0.002688172, 0.010752688, 0.005376344, 0.002688172, 0.02688172, 0.002688172, 0.002688172, 0.0, 0.0, 0.0,
  0.002688172, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.03763441, 0.005376344, 0.010752688, 0.0, 0.005376344, 0.002688172, 0.002688172, 0.021505376, 0.0, 0.0, 0.005376344,
   0.002688172, 0.0, 0.005376344, 0.002688172, 0.002688172, 0.002688172, 0.002688172, 0.0, 0.0, 0.002688172, 0.005376344, 0.0, 0.0, 0.005376344, 0.0, 0.002688172,
  0.0, 0.0, 0.002688172, 0.002688172, 0.0, 0.002688172, 0.0, 0.002688172, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.005376344, 0.0, 0.005376344, 0.002688172, 0.0, 0.0,
  0.0, 0.002688172, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.029569892, 0.002688172, 0.0, 0.016129032, 0.0, 0.0, 0.005376344, 0.005376344, 0.0,
  0.002688172, 0.002688172, 0.0, 0.0, 0.002688172, 0.016129032, 0.008064516, 0.0, 0.0, 0.010752688, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.002688172,
  0.0, 0.002688172, 0.0, 0.010752688, 0.002688172, 0.0, 0.008064516, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.005376344, 0.002688172, 0.0, 0.0, 0.0, 0.002688172,
  0.005376344, 0.0, 0.008064516, 0.005376344, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.008064516, 0.002688172, 0.008064516, 0.0, 0.002688172, 0.010752688, 0.0, 0.0,
  0.0, 0.0, 0.0, 0.0, 0.002688172, 0.002688172, 0.0, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.005376344, 0.0, 0.002688172, 0.0, 0.002688172,
  0.0, 0.0, 0.002688172, 0.0, 0.002688172, 0.0, 0.002688172, 0.008064516, 0.0, 0.0, 0.008064516, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.002688172,
   0.0, 0.0, 0.008064516, 0.0, 0.010752688, 0.002688172, 0.002688172, 0.002688172, 0.0, 0.0, 0.002688172, 0.0, 0.002688172, 0.008064516, 0.0, 0.0, 0.005376344, 0.0,
  0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.008064516, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.005376344, 0.0, 0.0, 0.008064516,
  0.005376344, 0.0, 0.0, 0.0, 0.0, 0.005376344, 0.0, 0.021505376, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.005376344, 0.016129032, 0.005376344, 0.002688172,
  0.005376344, 0.0, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.01344086, 0.0, 0.008064516, 0.0, 0.0, 0.0, 0.010752688, 0.005376344, 0.002688172, 0
  .002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.024193548, 0.008064516, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.005376344, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0,
  0.0, 0.008064516, 0.0, 0.002688172, 0.002688172, 0.0, 0.0, 0.005376344, 0.0, 0.0, 0.0, 0.032258064, 0.0, 0.005376344, 0.0, 0.002688172, 0.0, 0.002688172,
  0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.010752688, 0.008064516, 0.0, 0.002688172,
  0.0, 0.002688172, 0.0, 0.0, 0.002688172, 0.002688172, 0.0, 0.0, 0.0, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.008064516, 0.010752688, 0.0,
  0.005376344, 0.002688172, 0.0, 0.010752688, 0.005376344, 0.0, 0.0, 0.005376344, 0.002688172, 0.0, 0.002688172, 0.0, 0.002688172, 0.0, 0.002688172, 0.005376344,
  0.0, 0.005376344, 0.005376344, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.002688172, 0.005376344, 0.0, 0.0, 0.002688172, 0.0, 0.0, 0
  .005376344, 0.010752688, 0.0, 0.010752688, 0.005376344, 0.002688172 ]
--Histogram size : 400
15/07/23 14:59:21 INFO FileInputFormat: Total input paths to process : 1
15/07/23 14:59:22 INFO CodecPool: Got brand-new decompressor [.gz]
2.0 0.0 1.0
15/07/23 14:59:22 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
15/07/23 14:59:22 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
Predicting test image : airplanes
```

Text Classification using Streaming Input from Android and Command given to iOS

Android Controller

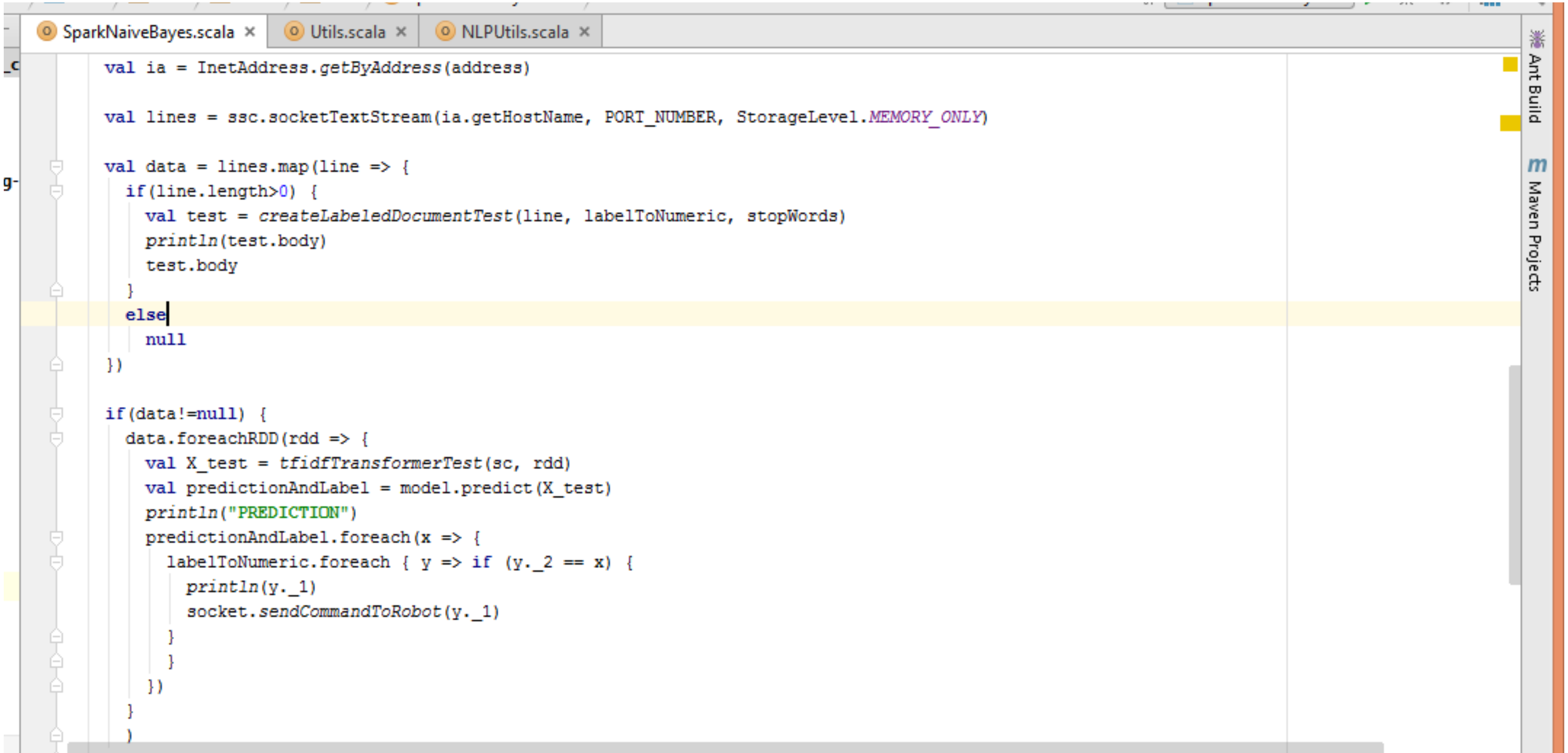
- Socket Server is initialized in Android.

- The Commands from Android are set to the Spark SocketStream

iOS

- Socket Server is established to receive the continuous commands from the Spark Processing.
- The output of the Spark Processing are given as commands

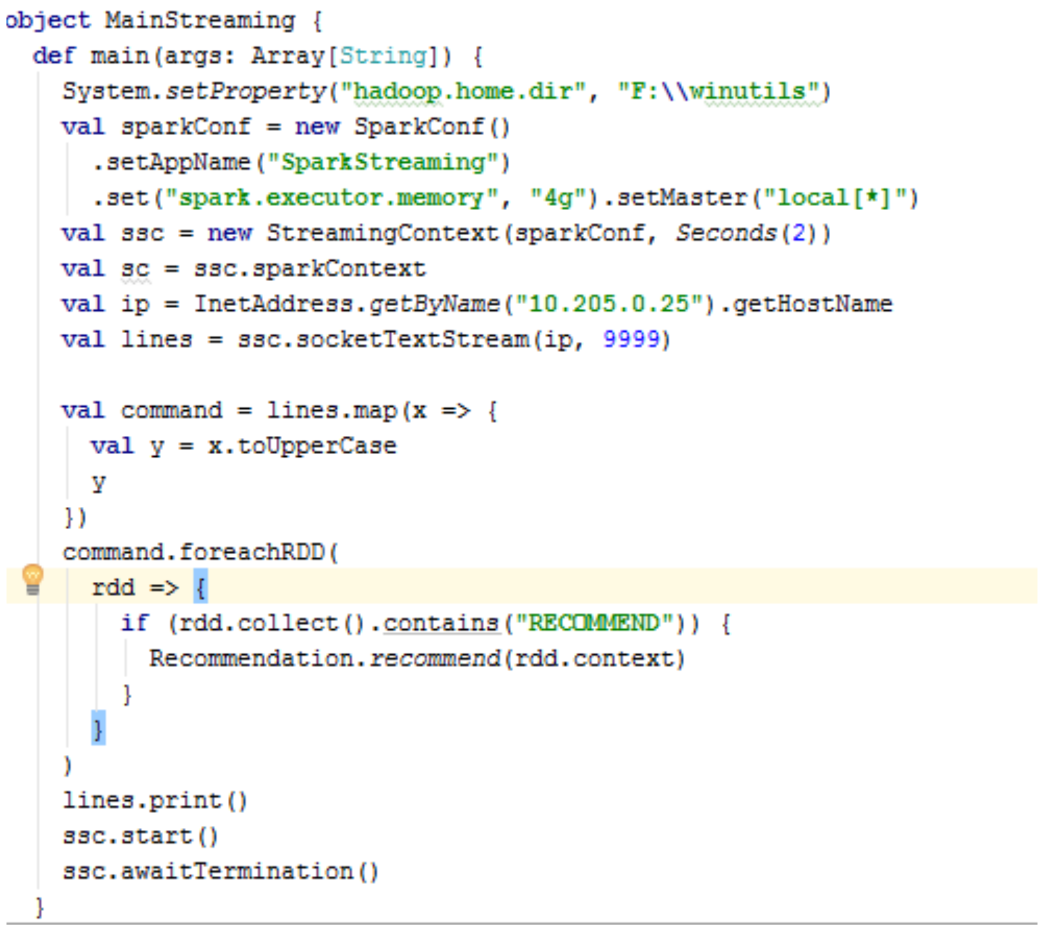
Spark Processing



```
val ia = InetAddress.getByAddress(address)

val lines = ssc.socketTextStream(ia.getHostName, PORT_NUMBER, StorageLevel.MEMORY_ONLY)

val data = lines.map(line => {
  if(line.length>0) {
    val test = createLabeledDocumentTest(line, labelToNumeric, stopWords)
    println(test.body)
    test.body
  }
  else
    null
})

if(data!=null) {
  data.foreachRDD(rdd => {
    val X_test = tfidfTransformerTest(sc, rdd)
    val predictionAndLabel = model.predict(X_test)
    println("PREDICTION")
    predictionAndLabel.foreach(x => {
      labelToNumeric.foreach { y => if (y._2 == x) {
        println(y._1)
        socket.sendCommandToRobot(y._1)
      }
      }
    })
  }
  )
```

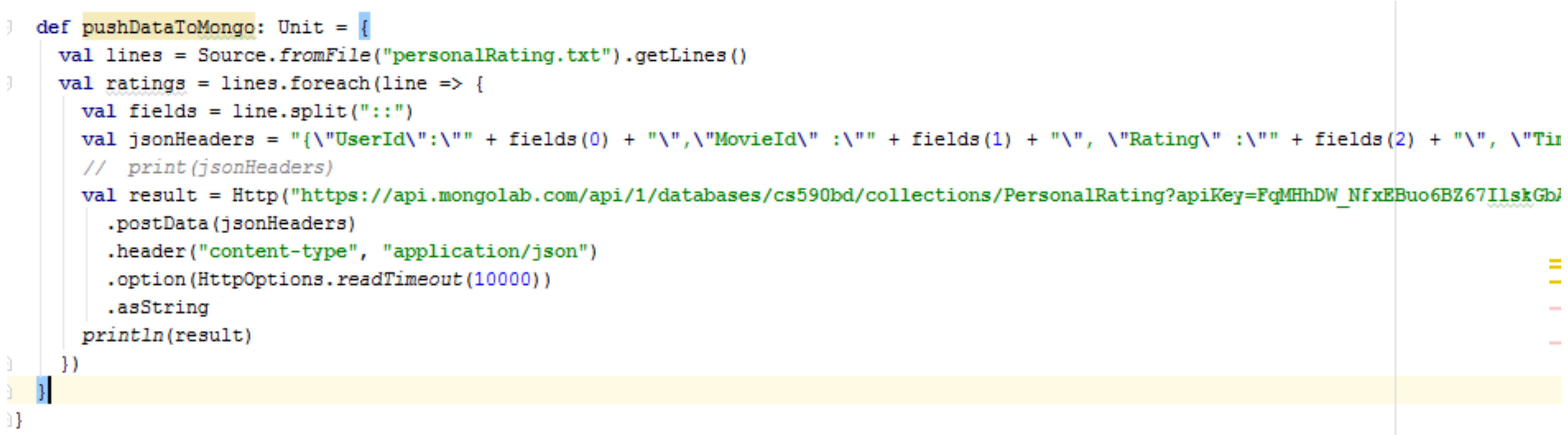
Recommendation using Streaming Input from Android

Spark Processing



```
object MainStreaming {
  def main(args: Array[String]) {
    System.setProperty("hadoop.home.dir", "F:\\winutils")
    val sparkConf = new SparkConf()
      .setAppName("SparkStreaming")
      .set("spark.executor.memory", "4g").setMaster("local[*]")
    val ssc = new StreamingContext(sparkConf, Seconds(2))
    val sc = ssc.sparkContext
    val ip = InetAddress.getByName("10.205.0.25").getHostName
    val lines = ssc.socketTextStream(ip, 9999)

    val command = lines.map(x => {
      val y = x.toUpperCase
      y
    })
    command.foreachRDD(
      rdd => {
        if (rdd.collect().contains("RECOMMEND")) {
          Recommendation.recommend(rdd.context)
        }
      }
    )
    lines.print()
    ssc.start()
    ssc.awaitTermination()
  }
```

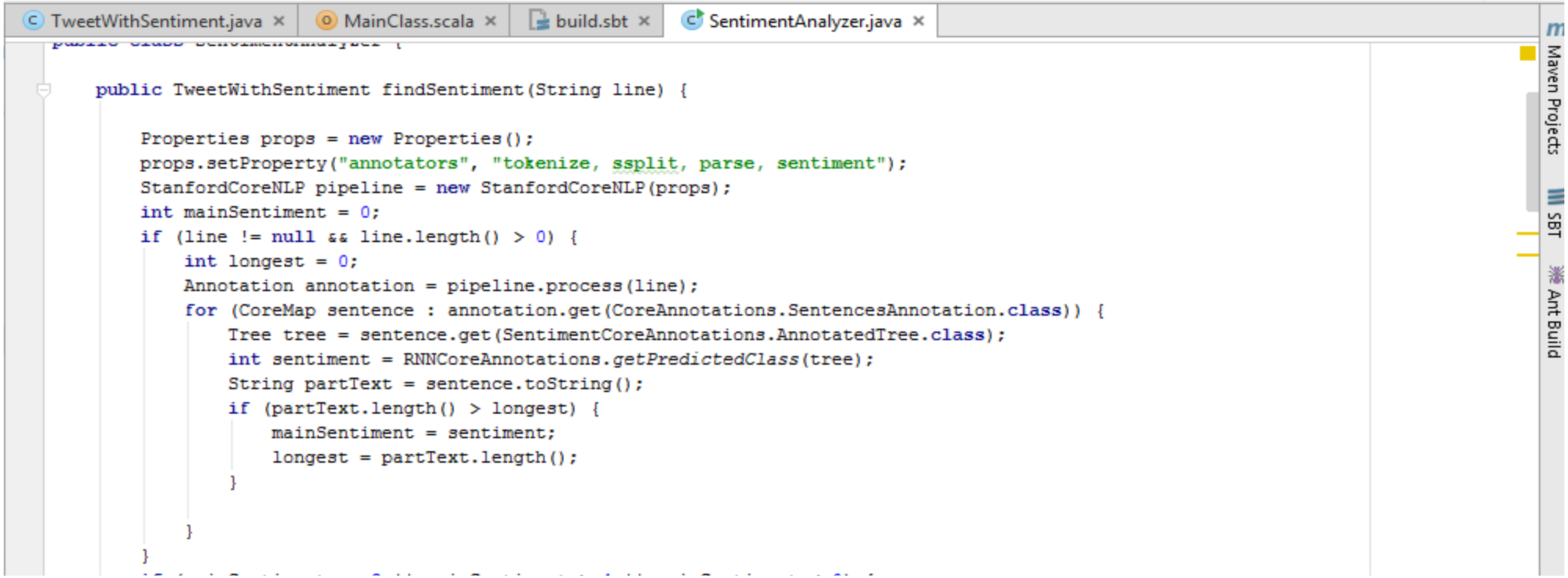
MongoLab Retrieval



```
def pushDataToMongo: Unit = {
  val lines = Source.fromFile("personalRating.txt").getLines()
  val ratings = lines.foreach(line => {
    val fields = line.split("::")
    val jsonHeaders = "{\"UserId\":\"" + fields(0) + "\",\"MovieId\" :\"" + fields(1) + "\", \"Rating\" :\"" + fields(2) + "\", \"Tim
    //  print(jsonHeaders)
    val result = Http("https://api.mongolab.com/api/1/databases/cs590bd/collections/PersonalRating?apiKey=FqMHhDW_NfxEBuo6BZ67IlskGbA
      .postData(jsonHeaders)
      .header("content-type", "application/json")
      .option(HttpOptions.readTimeout(10000))
      .asString
    println(result)
  })
}
}
```

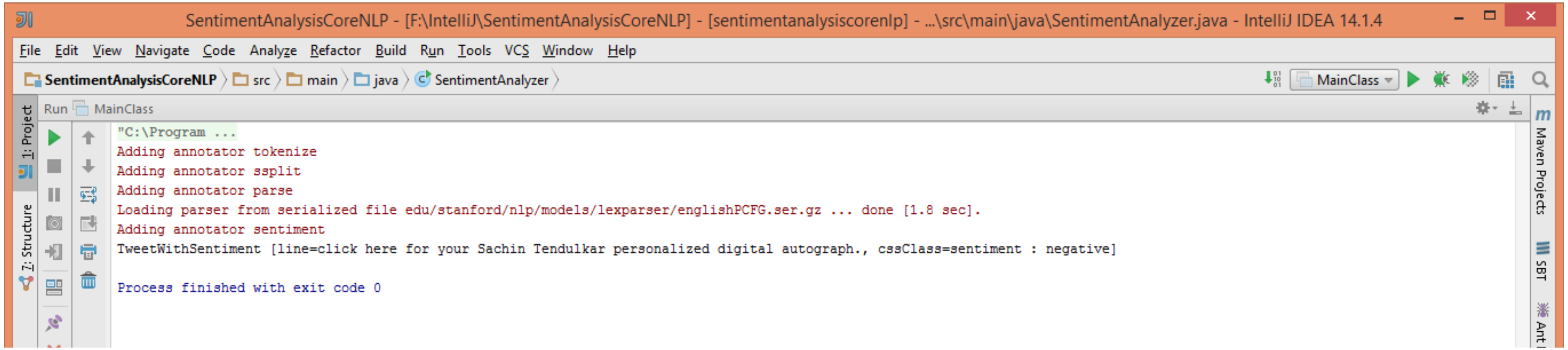
Sentiment Analysis using Stanford Core NLP

Sentiment Analysis



```
    public TweetWithSentiment findSentiment(String line) {

        Properties props = new Properties();
        props.setProperty("annotators", "tokenize, ssplit, parse, sentiment");
        StanfordCoreNLP pipeline = new StanfordCoreNLP(props);
        int mainSentiment = 0;
        if (line != null && line.length() > 0) {
            int longest = 0;
            Annotation annotation = pipeline.process(line);
            for (CoreMap sentence : annotation.get(CoreAnnotations.SentencesAnnotation.class)) {
                Tree tree = sentence.get(SentimentCoreAnnotations.AnnotatedTree.class);
                int sentiment = RNNCoreAnnotations.getPredictedClass(tree);
                String partText = sentence.toString();
                if (partText.length() > longest) {
                    mainSentiment = sentiment;
                    longest = partText.length();
                }

            }
        }
```

OUTPUT


SentimentAnalysisCoreNLP - [F:\IntelliJ\SentimentAnalysisCoreNLP] - [sentimentanalysiscorenlp] - ...\src\main\java\SentimentAnalyzer.java - IntelliJ IDEA 14.1.4
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
SentimentAnalysisCoreNLP src main java SentimentAnalyzer
MainClass
Run MainClass
1: Project
7: Structure
Maven Projects
SBT
Ant

```
"C:\Program ...
Adding annotator tokenize
Adding annotator ssplit
Adding annotator parse
Loading parser from serialized file edu/stanford/nlp/models/lexparser/englishPCFG.ser.gz ... done [1.8 sec].
Adding annotator sentiment
TweetWithSentiment [line=click here for your Sachin Tendulkar personalized digital autograph., cssClass=sentiment : negative]

Process finished with exit code 0
```

Thank you