

Εργασία εξαμήνου στο μάθημα:

# Αποθήκες δεδομένων & Εξόρυξη δεδομένων

Βασίλης Παπαστέργιος [parastva@csd.auth.gr](mailto:parastva@csd.auth.gr) (3651)

Λάζαρος Γώγος [lozarosg@csd.auth.gr](mailto:lozarosg@csd.auth.gr) (3877)



Προεπεξεργασία δεδομένων εισόδου



Εφαρμογή αλγορίθμου K-means



Εντοπισμός ανωμαλιών



Εντοπισμός συστάδων



Κριτική αξιολόγηση  
οδηγίων ChatGPT



# Προεπεξεργασία δεδομένων εισόδου

Η προεπεξεργασία των δεδομένων πραγματοποιήθηκε σε περιβάλλον Apache Spark, χρησιμοποιώντας ως γλώσσα τη Scala. Με βάση τις απαιτήσεις της εκφώνησης, τα στάδια προεπεξεργασίας που εφαρμόστηκαν ήταν τα εξής:

→ Καθαρισμός των εγγραφών με ελλιπείς τιμές ή κενό περιεχόμενο:

```
rawData.filter(line =>
  !line.startsWith(",") &&
  !line.endsWith(",") &&
  line.nonEmpty
)
```

→ Μετασχηματισμός (scaling) των δεδομένων στο διάστημα [0, 1] και για τις δύο διαστάσεις:

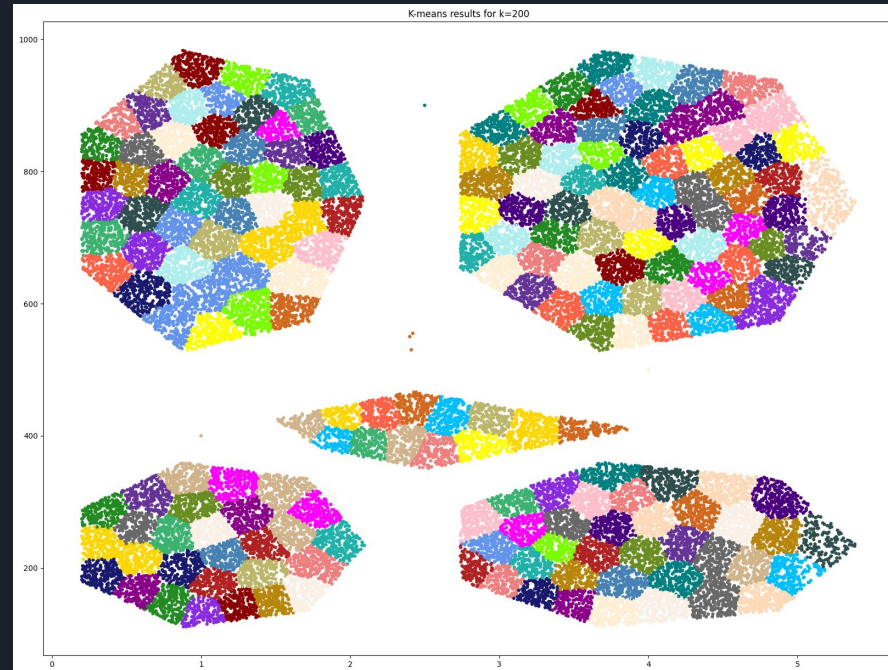
```
unscaledData.map(coordinates => {
  ...
  ((x - xMin) / (xMax - xMin), (y - yMin) / (yMax - yMin))
})
```



# Εφαρμογή αλγορίθμου K-means

Η εφαρμογή του αλγορίθμου K-means πραγματοποιήθηκε σε περιβάλλον Apache Spark, χρησιμοποιώντας ως γλώσσα τη Scala. Χρησιμοποιήθηκαν υλοποιήσεις για τον αλγόριθμο των K-μέσων από τη βιβλιοθήκη ML του Spark, και συγκεκριμένα από το πακέτο

`org.apache.spark.ml.clustering.KMeans`  
Δοκιμάστηκαν διαφορετικές τιμές για την υπερ-παράμετρο  $k$  του αλγορίθμου. Με βάση τις υποθέσεις τις εκφώνησης, ο αριθμός των ομάδων (clusters) είναι γνωστός και ίσος με 5. Γενική μας στόχευση ήταν να διερευνήσουμε τη συμπεριφορά του αλγορίθμου για  $k \gg 5$ . Δοκιμάστηκαν πολλαπλές εκτελέσεις του αλγορίθμου για  $k = \{20, 30, 50, 100, 200, 500\}$ . Για την οπτικοποίηση των αποτελεσμάτων χρησιμοποιήθηκαν οι βιβλιοθήκες Pandas και Matplotlib της γλώσσας Python.





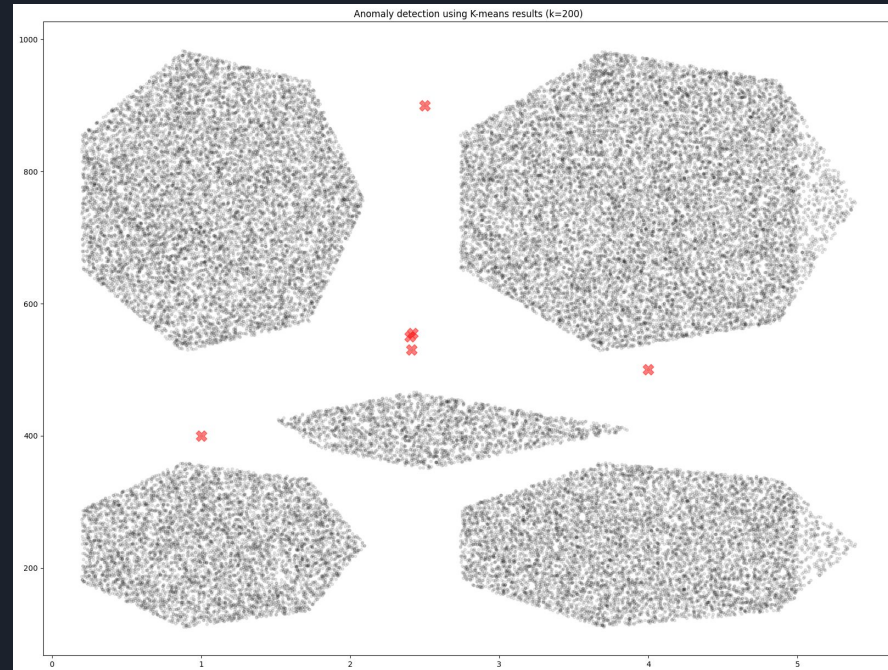
# Εντοπισμός ανωμαλιών

Ο εντοπισμός των ανωμαλιών του συνόλου δεδομένων πραγματοποιήθηκε σε περιβάλλον Apache Spark, χρησιμοποιώντας ως γλώσσα τη Scala. Στόχευσή μας ήταν να εντοπιστούν, με προγραμματιστικό τρόπο, ακραίες/άνωμαλες τιμές χρησιμοποιώντας κλιμακώσιμο αλγόριθμο. Η προσέγγιση που υιοθετήθηκε μπορεί να περιγραφεί από τον ακόλουθο ψευδοκώδικα:

Για κάθε σημείο:  
υπολόγισε την απόσταση από το κέντρο

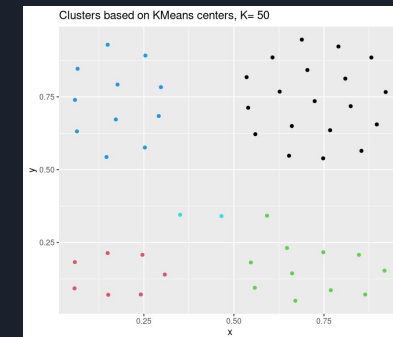
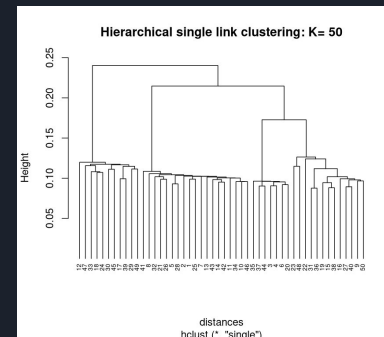
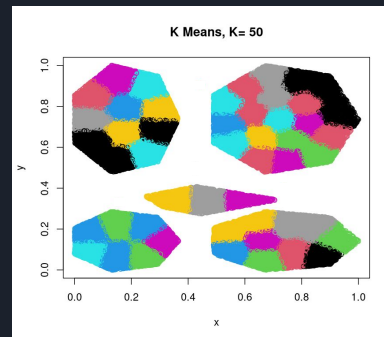
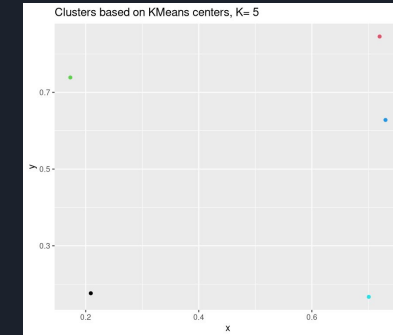
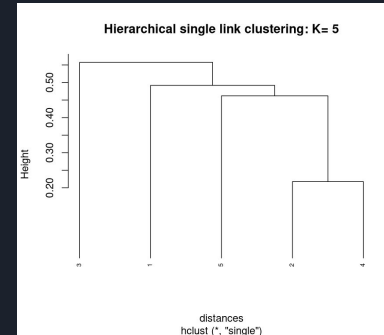
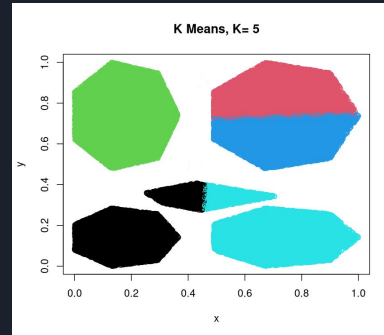
Για κάθε ομάδα:  
υπολόγισε  $\mu$  και  $\sigma$  των αποστάσεων

Θεώρησε ακραίες τιμές όσα σημεία έχουν απόσταση από το κέντρο της ομάδας τους μεγαλύτερη από  $\mu + 3.5\sigma$ .

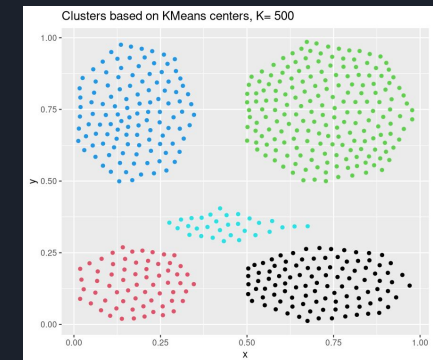
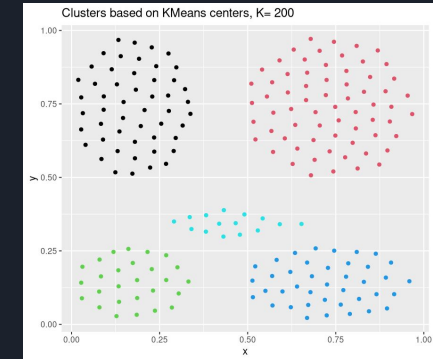
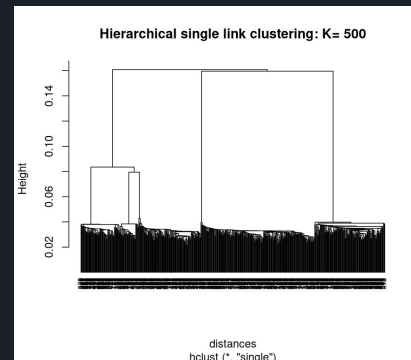
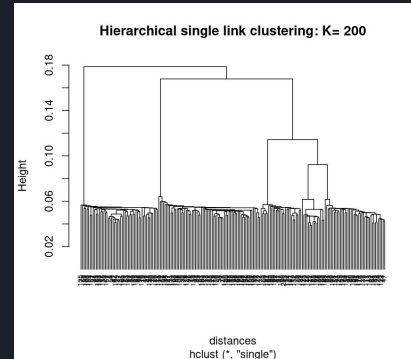
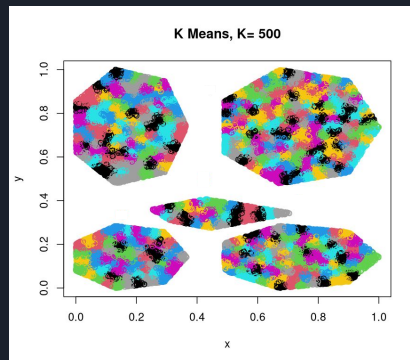
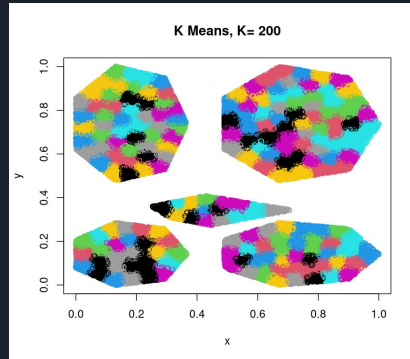


# Εντοπισμός συστάδων <sup>1/2</sup>

Ο εντοπισμός των ανωμαλιών του συνόλου δεδομένων πραγματοποιήθηκε σε γλώσσα R. Χρησιμοποιήθηκε και πάλι ο K-means για διάφορες τιμές του  $k \gg 5$ . Κατόπιν, αξιοποιώντας και τη γνώση από τα αποτελέσματα του K-means σε Spark, χρησιμοποιήθηκε η τιμή  $k=200$ . Για την εύρεση των τελικών (5) συστάδων, εφαρμόστηκε ο αλγόριθμος Ιεραρχικής Ομαδοποίησης Μονής Σύνδεσης (Hierarchical Single Link Clustering). Τα καλύτερα αποτελέσματα παρατηρήθηκαν για την τιμή  $k=200$ .



# Εντοπισμός συστάδων <sup>2/2</sup>





# Κριτική αξιολόγηση οδηγιών ChatGPT <sup>1/2</sup>

Prompt

Solve the following assignment using Spark and the Scala programming language. Our goal is to find clusters and outliers in 2 dimensional data using the kmeans algorithm, knowing beforehand that the final clusters of data are 5.

Task 1: Read the "points.csv" file which consists of x,y pairs.

Task 2: Remove any NA values, or any pairs that do not have either the x or the y value.

Task 3: Scale the data to be in the range 0 to 1 in both dimensions.

Task 4: Apply kmeans using Spark written in scala, using the already existing kmeans functions found in Spark.

Task 5: Detect any outliers that were found in task 4, in a scalable way.

The program will take only one input parameter and that will be the name of the csv file.

The program will print the following:

a) the time needed to execute the main function

b) the outliers that have been detected by the program

```
def cleanData (data: DataFrame): DataFrame = {
  data.na.drop("any", Seq("x", "y"))
}
```



```
def scaleData (data: DataFrame): DataFrame = {
  val assembler = new VectorAssembler()
  .setInputCols(Array("x", "y"))
  .setOutputCol("features")
  val assembledData = assembler.transform(data)
```



```
val scaler = new StandardScaler()
  .setInputCol("features").setOutputCol("scaledFeatures")
  .setWithMean(false)
val scaledData = scaler
  .fit(assembledData).transform(assembledData)

scaledData.select("scaledFeatures")
}
```

Απάντηση





# Κριτική αξιολόγηση οδηγιών ChatGPT <sup>2/2</sup>

```
def applyKMeans(data: DataFrame, k: Int): DataFrame = {
  val kmeans = new KMeans().setK(k).setSeed(1L)
  kmeans.fit(data).transform(data)
}
```



```
def detectOutliers(clusters: DataFrame, data: DataFrame): DataFrame = {
  val withCluster = data.join(clusters, data("scaledFeatures") === clusters("scaledFeatures"))

  val clusteredData = withCluster.select("x", "y", "prediction")
```

```
  val outliers = clusteredData.groupBy("prediction").agg(
    expr("percentile_approx(x, 0.25)").alias("q1"),
    expr("percentile_approx(x, 0.75)").alias("q3")
  ).withColumn("IQR", col("q3") - col("q1"))
  .withColumn("lowerBound", col("q1") - 1.5 * col("IQR"))
  .withColumn("upperBound", col("q3") + 1.5 * col("IQR"))

  val outliersData = clusteredData
  .join(outliers, clusteredData("prediction") === outliers("prediction"))
  .where(
    col("x") < col("lowerBound") || col("x") > col("upperBound") ||
    col("y") < col("lowerBound") || col("y") > col("upperBound")
  )
  .select("x", "y")
  outliersData
}
```

Εκτέλεση (Runtime error)

overloaded method \* with alternatives:

```
(x: Double)Double <and>
(x: Float)Double <and>
(x: Long)Double <and>
(x: Int)Double <and>
(x: Char)Double <and>
(x: Short)Double <and>
(x: Byte)Double
cannot be applied to (org.apache.spark.sql.Column)
.withColumn("lowerBound", col("q1") - 1.5 *
col("IQR"))
```



Απάντηση