Εργασία στο μάθημα Διαχείριση Μεγάλων Δεδομένων - HDFS

Στοιχεία Φοιτητών:

ΠΑΣΣΟΣ ΔΗΜΗΤΡΙΟΣ 2024201000083
ΜΠΡΑΟΥΝΙ ΓΙΟΑΝΙ 2022201800131

Μελέτη των Δεδομένων

Numeric :

Year_Birth,Income,Kidhome,Teenhome,Dt_Customer,Recency,Όλα τα στοιχεία αγορών και τόπος αγορών,NumDealsPurchases

Nominal :

ID,Education,Marital_Status,Complain,Όλα τα στοιχεία προσφορών εκτός από το NumDealsPurchases

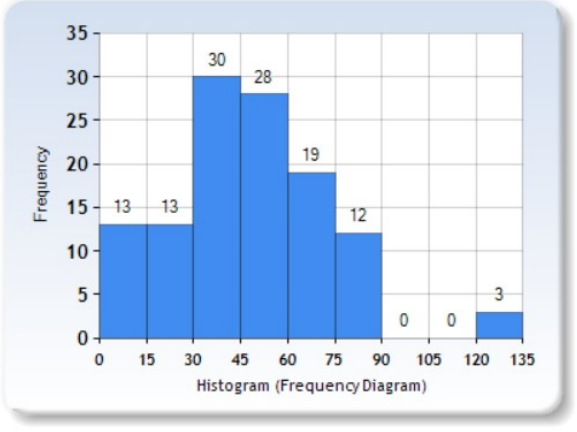
Μελέτη των Δεδομένων

- Outliers Ηλικίες : Οι ακραίες τιμές βρίσκονται στην ηλικιακή ομάδα 120-135

Result

Success! We have edited your histogram, and updated the frequency table and histogram information, based on the values you provided via the edit tool.

Frequency Table	
Class	*Count*
0-14	13
15-29	13
30-44	30
45-59	28
60-74	19
75-89	12
90-104	0
105-119	0
120-134	3



Your Histogram	
Mean	47.05932
Standard Deviation (s)	25.35737
Skewness	0.38862
Kurtosis	0.75045
Lowest Score	1
Highest Score	128
Distribution Range	127
Total Number of Scores	118
Number of Distinct Scores	74
Lowest Class Value	0
Highest Class Value	134
Number of Classes	9
Class Range	15

Μελέτη των Δεδομένων

- Outliers Ποσά Εισοδημάτων : Δεν υπάρχουν ακραίες τιμές.

  Στην πραγματικότητα τόσο οι ηλικίες οσο τα εισοδήματα , δεν μας επηρεάζουν στον υπολογισμό των δεδομένων στα παρακάτω ερωτήματα.

Μελέτη των Δεδομένων

Δεν έχει γινεί κάποια προ-επεξεργασία ή καθαρισμός των δεδομενών , αλλά στο κώδικα γίνεται ο κατάλληλος έλεγχος όπου χρείαζεται κατά το διάβασμα του αρχείου.

Εκπαιδευτικό υπόβαθρο πελατών

α) Ψευδοκώδικας

```
void Map(FILE fileName, String line)
{
    word = line[2];
    emit(word,1)
}
```

```
void Reduce(Word w, int[] counts)
{
        int sum = 0;
        foreach (int i in counts)
        {
            sum += i;
        }
       emit(w, sum);
 }
```

Εκπαιδευτικό υπόβαθρο πελατών

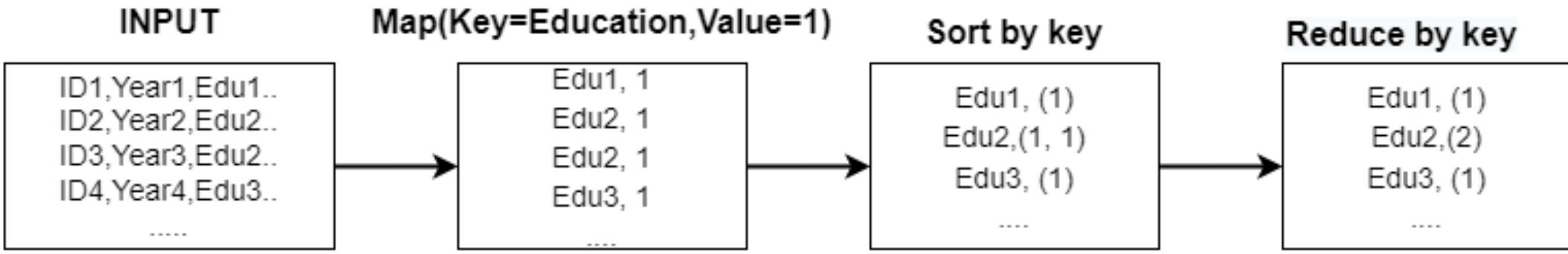
β) Παράδειγμα

Map		**Reduce**	
Graduation,	1	Graduation ,	1
Phd ,	1	Phd ,	2
Master ,	1	Master ,	1
Phd ,	1		

Ομαδοποιούμε τους πελάτες , ανάλογα με το εκπαιδευτικό τους υπόβαθρο.

Εκπαιδευτικό υπόβαθρο πελατών

γ) Σχηματική εκτέλεση



Ανάδειξη των πελατών που είναι πιο πιθανό να αγοράσουν ένα νέο προϊόν κρασιού

α) Ψευδοκώδικας για τον υπολογισμό του μέσου όρου για κρασί

```
void Map(FILE fileName, String line)
{
        mntWine = line[9];
        emit(text,mntWine)
}

void Reduce(Text t, double[] counts)
{
        double sum = 0;
        int count=0;
        double averageTotalSpent=0;

        foreach (double i in counts)
        {
          sum += i;
          count++;
        }
        averageTotalSpent = sum/count;
        emit(t, averageTotalSpent);
}
```

Ανάδειξη των πελατών που είναι πιο πιθανό να αγοράσουν ένα νέο προϊόν κρασιού

β) Παράδειγμα υπολογισμού του μέσου όρου για κρασί

Map

“” , 40

“” , 20

“” , 35

“” , 50

Reduce

“Average , Count” ,

(40+20+35+50)/4

Προσθέτουμε το **MntWines** απο κάθε πελάτη και το διαιρούμε με το πλήθος των πελατών για να βρούμε το μέσο όρο

Ανάδειξη των πελατών που είναι πιο πιθανό να αγοράσουν ένα νέο προϊόν κρασιού

α) Ψευδοκώδικας τελικού προγράμματος

```
void Map(FILE fileName, String line)
{
        mntWine = line[9];
        diffAge = 2021 - line_tokens[1];
        ID = line_tokens[0];
        Age = diffAge;
        Education = line_tokens[2];
        Marital_Status = line_tokens[3];
        Income = line_tokens[4];
        MntWines = line_tokens[9];

        if(mntWine>=(average*1.5)
        {
           emit(mntWine,ID + Age + Education + Marital_Status + Income + MntWines);
        }
}

int count=0;
void Reduce(Int key, Text[] text)
{
        count++;
        foreach (double i in counts)
        {
          sum += i;
          count++;
        }
        emit(count, text);
}
```

Ανάδειξη των πελατών που είναι πιο πιθανό να αγοράσουν ένα νέο προϊόν κρασιού

β) Παράδειγμα

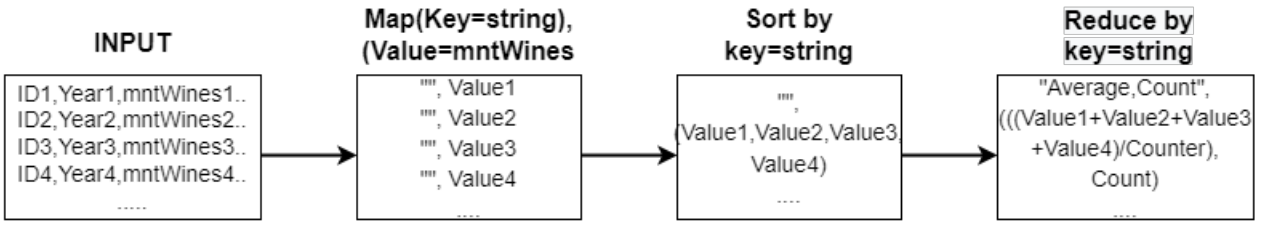
Map	**Reduce**
1492 , 7431 62 Graduation Together 87771	1 , 7431 62 Graduation Together 87771
1478 , 5547 39 PhD Married 84169	2 , 6000 40 Master Together 75000
1396 , 11088 50 PhD Together 78642	3 , 5547 39 PhD Married 84169
1492 , 6000 40 Master Together 75000	4 , 11088 50 PhD Together 78642

Ομαδοποιούμε τις εγγραφές με το ίδιο **MntWines** και τις κατατάσουμε σε φθίνουσα σειρά ως προς τα συνολικά χρήματα που ξόδεψαν για κρασί.

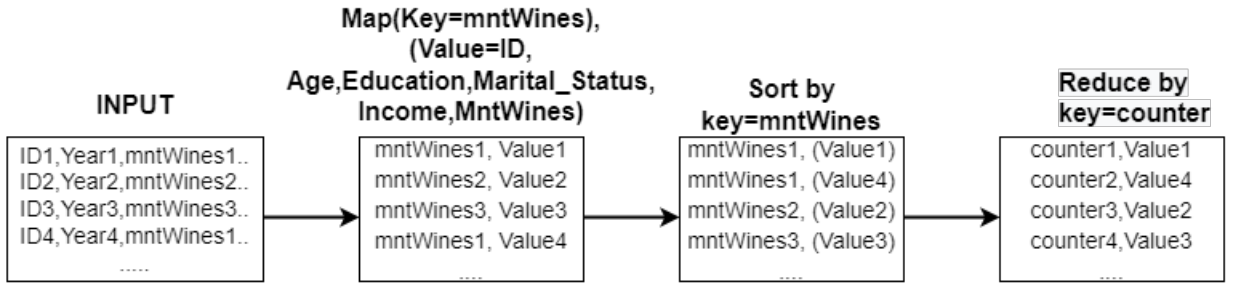
Ανάδειξη των πελατών που είναι πιο πιθανό να αγοράσουν ένα νέο προϊόν κρασιού

γ) Σχηματική εκτέλεση μέσου όρου και τελικού προγραμάτος

MAP1⟶ REDUCE1



MAP2⟶ REDUCE2



Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική

α) Ψευδοκώδικας υπολογισμού ΜΟ όλων των αγορών

```
void Map(FILE fileName, String line)
{
        wines = line[9];
        fruits = line[10];
        meat = line[11];
        fish = line[12];
        sweet = line[13];
        gold = line[14];

        totalSpent = (Wines + Fruits + Meat + Fish + Sweet + Gold);
        emit(Text,totalSpent);
}

double sum = 0;
int count=0;
double averageTotalSpent=0;

void Reduce(Text t, double[] values)
{
        foreach (double i in values)
        {
          sum += i;
          count++;
        }

        averageTotalSpent = sum/count;
        emit(t, averageTotalSpent);
}
```

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική

β) Παράδειγμα

Map	**Reduce**
“” , 40	“Average , Count” , (40+100+150+230)/4
“” , 100	
“” , 150	
“” , 230	

Προσθέτουμε το ποσό που έχει ξοδέψει κάθε πελάτης για τις αγορές του και το διαιρούμε με το πλήθος των πελατών για να βρούμε το μέσο όρο

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική

α) Ψευδοκώδικας τελικού προγράμματος

```
void Map(FILE fileName, String line)
{
        id = line[0];
        wines = line[9];
        fruits = line[10];
        meat = line[11];
        fish = line[12];
        sweet = line[13];
        gold = line[14];
        income = line[4];
        date = line[7];

        totalSpent = (Wines + Fruits + Meat + Fish + Sweet + Gold);
        if(totalSpent>=average*1.5)
        {
                if(income>69500)
                {
                        if(date==21)
                                emit(Gold,id);
                        else
                                emit(Silver,id);
                }
        }
}
```

```
double sum = 0;
int count=0;
double averageTotalSpent=0;

void Reduce(Text t, long[] values)
{
        ArrayList list;
        foreach (long i in values)
        {
          list.add(i);
        }
        sort.list
        emit(t, list);
}
```

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική

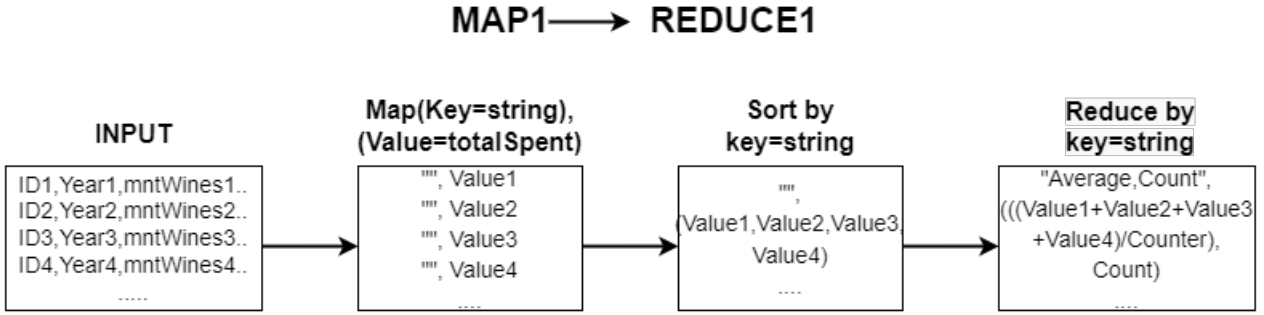
β) Παράδειγμα

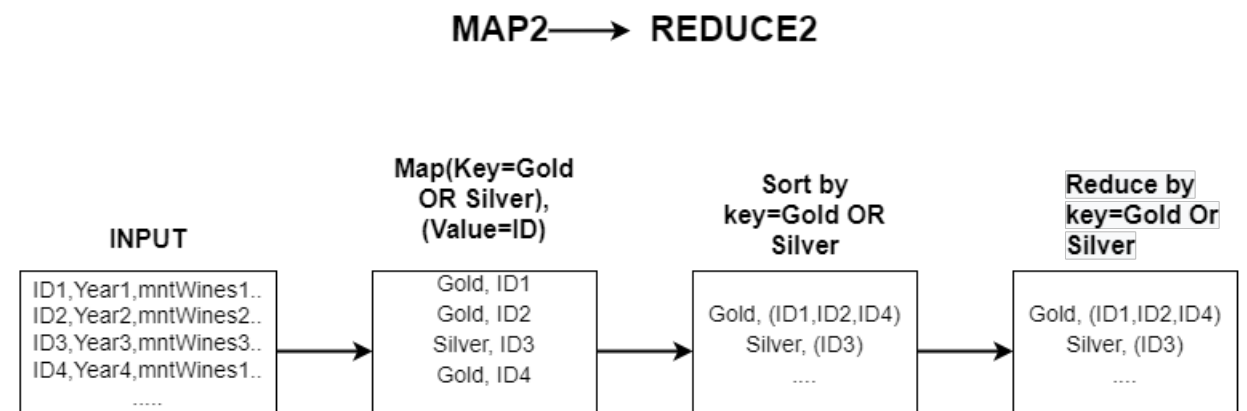
Map	**Reduce**
Gold , 40	Gold , (40 , 150)
Silver , 100	Silver, (100 , 230)
Gold , 150	
Silver , 230	

Ομαδοποιούμε τα ID των πελατών σε gold ή silver

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική

γ) Σχηματική εκτέλεση μέσου όρου και τελικού προγραμάτος





Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική(Bonus)

α) Ψευδοκώδικας υπολογισμού ΜΟ του εισοδήματος

```
void Map(FILE fileName, String line)
{
        income = line[4];
        emit(Text,income);
}

double sum = 0;
int count=0;
double averageIncome=0;

void Reduce(Text t, double[] values)
{
        foreach (double i in values)
        {
          sum += i;
          count++;
        }
        averageIncome = sum/count;
        emit(t, averageIncome);
}
```

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική(Bonus)

β) Παράδειγμα

Map	**Reduce**
“” , 15000	“Average,Count“,(15000+17500+18000+35000)/4
“” , 17500	
“” , 18000	
“” , 35000	

Προσθέτουμε το εισόδημα όλων των πελατών και το διαιρούμε με το πλήθος των πελατών για να βρούμε το μέσο όρο

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική(Bonus)

α) Ψευδοκώδικας τελικού προγράμματος

```
void Map(FILE fileName, String line)
{
        id = line[0];
        wines = line[9];
        fruits = line[10];
        meat = line[11];
        fish = line[12];
        sweet = line[13];
        gold = line[14];
        income = line[4];
        date = line[7];

        totalSpent = (Wines + Fruits + Meat + Fish + Sweet + Gold);
        if(totalSpent<=averageTotalSpent/4)
        {
                if(income<51687.45)
                {
                        if(date==21)
                                emit(Bronze,id);
                        else
                                emit(Papeer,id);
                }
        }
}
```

```
double sum = 0;
int count=0;
double averageTotalSpent=0;

void Reduce(Text t, long[] values)
{
        ArrayList list;
        foreach (long i in values)
        {
          list.add(i);
        }
        sort.list
        emit(t, list);
}
```

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική(Bonus)

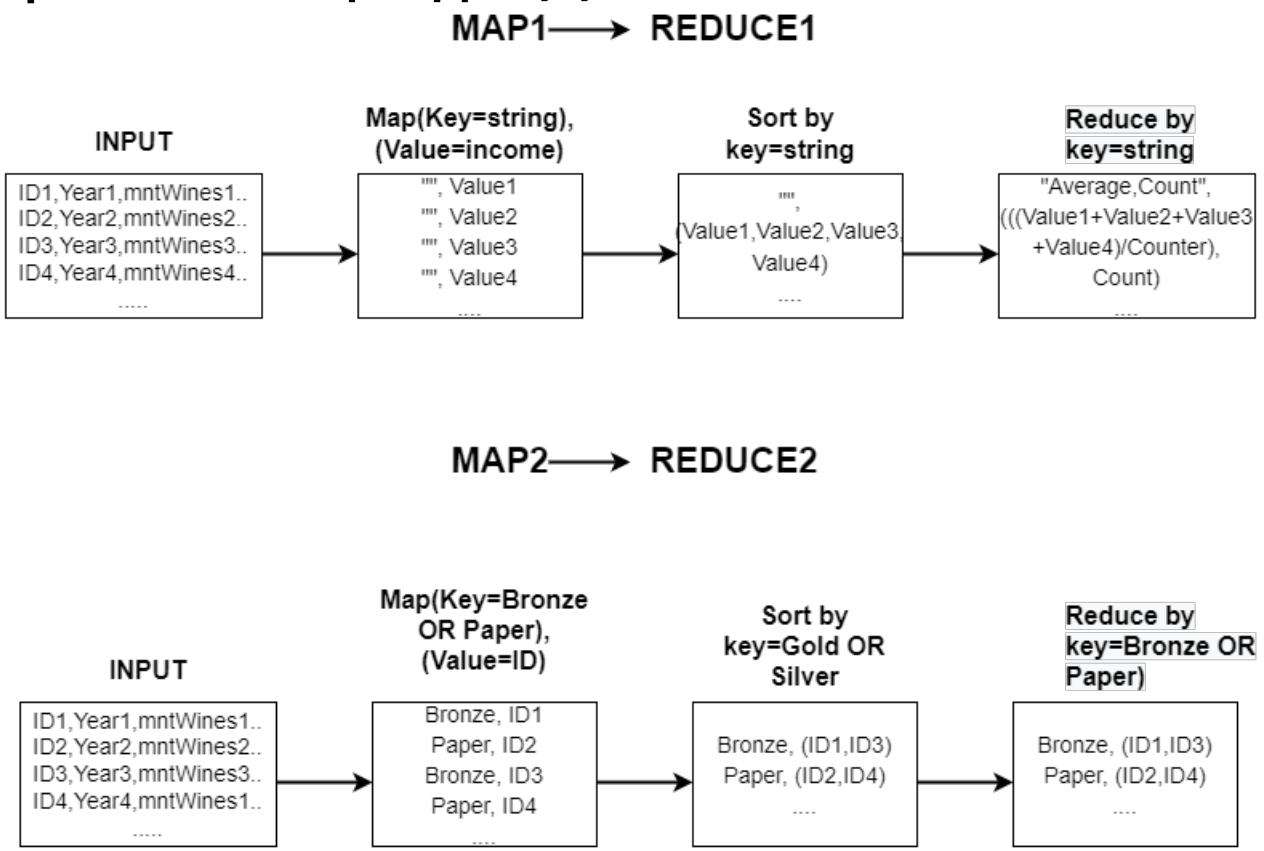
β) Παράδειγμα

Map	**Reduce**
Bronze , 400	Bronze , (400 , 1500)
Paper , 1000	Paper, (1000 , 2000)
Bronze , 1500	
Paper , 2000	

Ομαδοποιούμε τα ID των πελατών σε bronze ή paper

Κατηγοριοποίηση των πελατών ανάλογα με την αγοραστική τους δυναμική(Bonus)

γ) Σχηματική τελικού προγράμματος



Επιπρόσθετες λεπτομέριες

- Σχετικά με τις ημερομηνίες , για κάθε ημερομηνία κρατάμε το έτος και κάνουμε την σύγκριση , για να δούμε εάν ένας πελάτης έχει αποκτήσει κωδικό το τελευταίο ή προηγούμενο έτος.

- Τα map reduce για τον υπολογισμό του μέσου όρου , βρίσκονται σε ξεχωριστό αρχείο απο το τελικό πρόγραμμα.

User Manual

Κρατάμε σε μια μεταβλητή το path που έχουμε αποθηκεύσει στον υπολογιστή μας.

1) export HADOOP_CLASSPATH=$(hadoop classpath)

Δημιουργούμε ένα φάκελο με όνομα Ergasia στο hadoop

2) hadoop fs -mkdir /Ergasia

Ανεβάζουμε το αρχείο εισόδου στο hadoop

3)hadoop fs -put personality_analysis.csv /*Ergasia*/Input

User Manual

Κάνουμε compile το αρχείο java

4)javac -Xlint -classpath ${HADOOP_CLASSPATH} -d '/home/kali/Downloads/compiledFIles' '/home/kali/Downloads/app.java'

Μετατρέπουμε το μεταγλωττισμένο αρχείο σε εκτελέσιμο

5)jar -cvf app.jar -C '/home/kali/Downloads/compiledFIles' .

Τρέχουμε το εκτελέσιμο αρχείο στο hadoop

6)hadoop jar '/home/kali/Downloads/app.jar' app /Ergasia/Input /Ergasia/Output