CSCI_B 565 DATA MINING
Homework Number 2
Morning Class Computer Science Core
Spring
Indiana University,
Bloomington, IN

Puneet Loya
ploya

Feb 16,2015

All the work here is solemnly mine

1 Answer for Question 1

a. Sample space Ω =
{(1,1),(1,2),(1,3),(1,4),(1,5),(1,6),
(2,1),(2,2),(2,3),(2,4),(2,5),(2,6),
(3,1),(3,2),(3,3),(3,4),(3,5),(3,6),
(4,1),(4,2),(4,3),(4,4),(4,5),(4,6),
(5,1),(5,2),(5,3),(5,4),(5,5),(5,6),
(6,1),(6,2),(6,3),(6,4),(6,5),(6,6)}

b. The Random variable X = {2,3,4,5,6,7,8,9,10,11,12} parititions the data based on a throw.

c. Function of the random variable = g(X)
g(x) = { 9.99 if sum = 9
-1.00 if sum $\neq$ 9

d. Mass function over X

X	2	3	4	5	6	7	8	9	10	11	12
p(X)	0.027	0.056	0.083	0.11	0.138	0.16	0.13	0.111	0.083	0.056	0.027

e. Mass function over g(X)

g(X)	9.99	-1
p(g(X))	0.111	0.88

f. Expected Value of X

   E[X] = $\Sigma x.p(x)$

   $E[X] = 2 * 0.027 + 3 * 0.056 + 4 * 0.083 + 5 * 0.11 + 6 * 0.138 + 7 * 0.16$
   $+ 8 * 0.13 + 9 * 0.111 + 10 * 0.083 + 11 * 0.056 + 12 * 0.027$

   E[X] = 7

g. E[g(X)] = $\Sigma g(x).p(g(x))$

   E[g(X)] = 0.22

h. Probability of Winning Coffee = 3 * 0.111 = 0.33
   Probability of Winning Cookies = 1 * 0.88 = 0.88

   The winner is more likely to have cookies.

2 Answer for Question 2

2.1 Part 1

The uses of Linear Regression referred from [3]:

- Pure Description : Getting a well defined description of the response variable by fitting the curve of data using least squares method. Many prefer looking at the R^2 value as it is easier to interpret the pure description.
- Prediction and Estimation: In regression, the difference between predicting of future response and estimating mean response must be recognized. The prediction variance is small due to variability in estimation of coefficients but largely dependent on the variability of the system as a whole.
- Extrapolation: This may result in two disadvantages. The model may no longer apply. Secondly, if the model applies the predictor may not predict outside the range as good as it does within the range.
- Estiamtion of parameters: Estimated variance and the bias resulting from eliminating the variable should be examined.
- Control: The output must be controlled by varying the level of inputs.
- Model Building: Developing a model for the response by evaluating subset models(regression) which may reveal relationship between the variables involved in the regression.

  Ridge regression: It is a way of penalizing variables in a multivariate regression setup which suffer from Multicollinearity. Hence ploting a ridge regression gives the variable subsets that must be considered and that can be ignored.

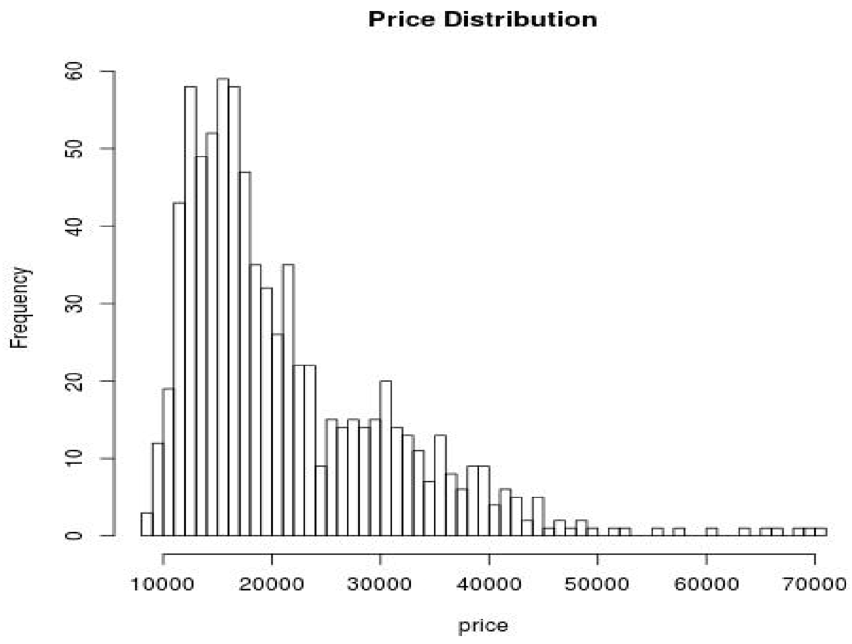
2.2 Part 2

I have created all the tables. The tables and the preliminary queries are attached in the appendix.

2.3 Part 3

- Histograms for all Attributes

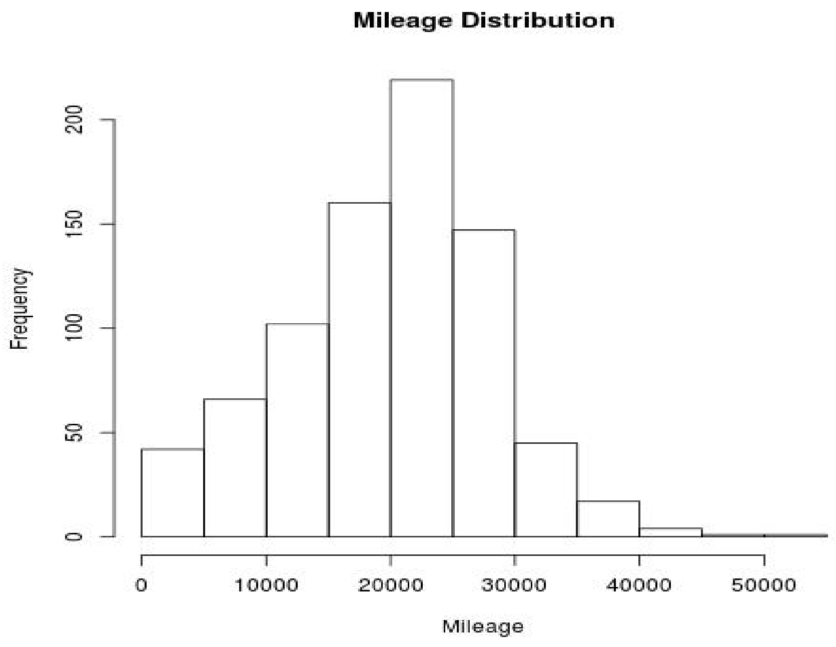
  Histogram for Price:



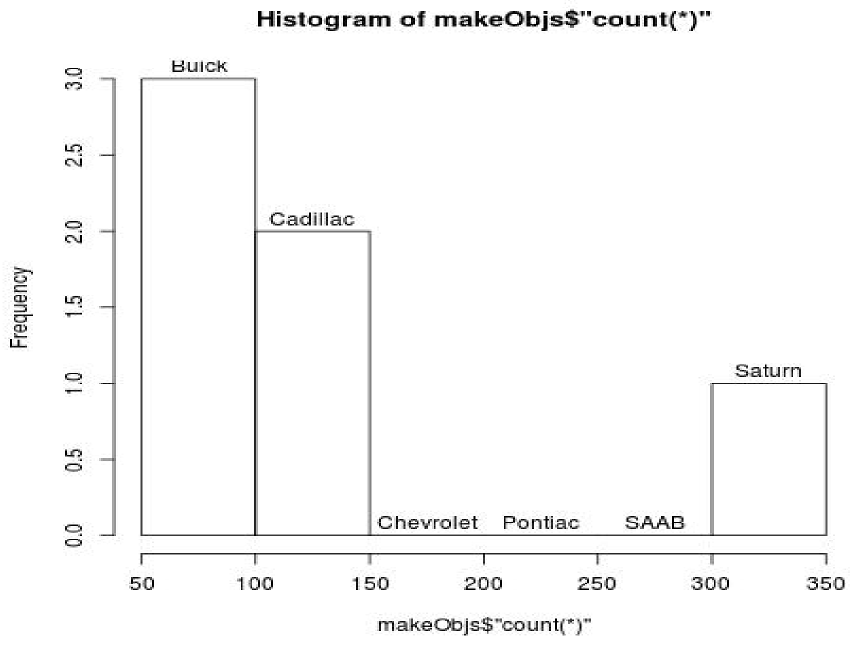
The price distribution shows that frequency of buying used cars reduces with increase in price atleast when the price is above 30k.

Histogram for mileage:

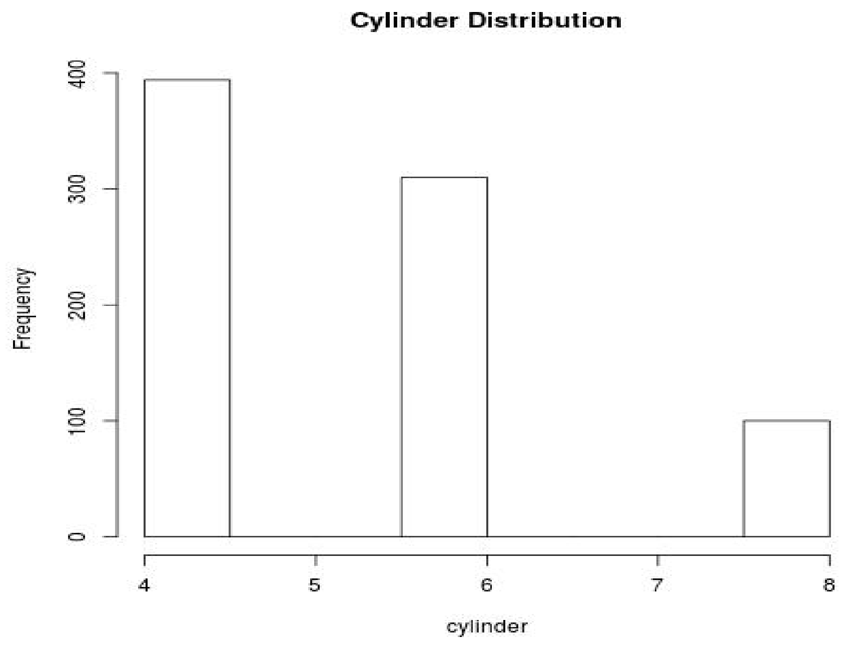
The mileage distribution shows that



Histogram for make:

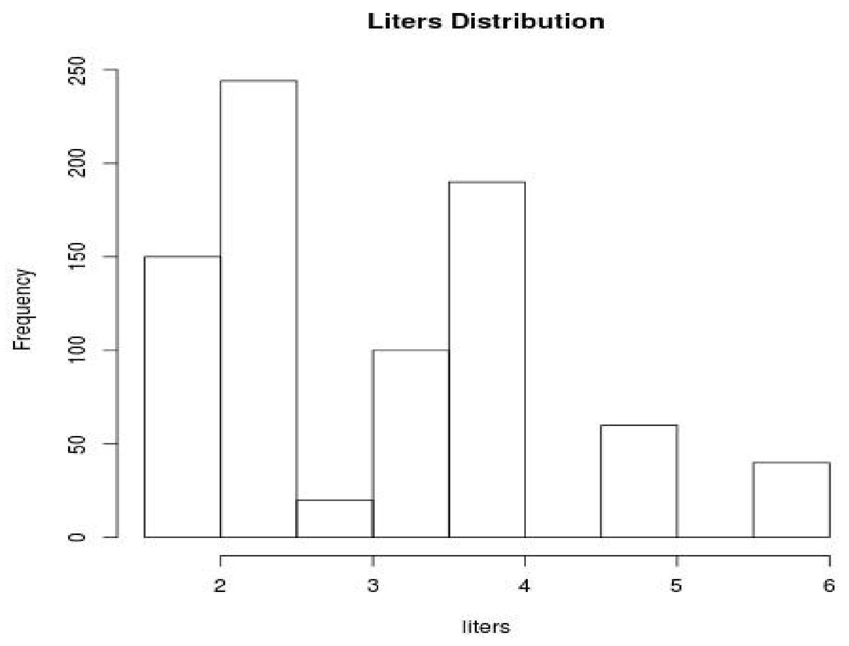


Histogram for Cylinder:

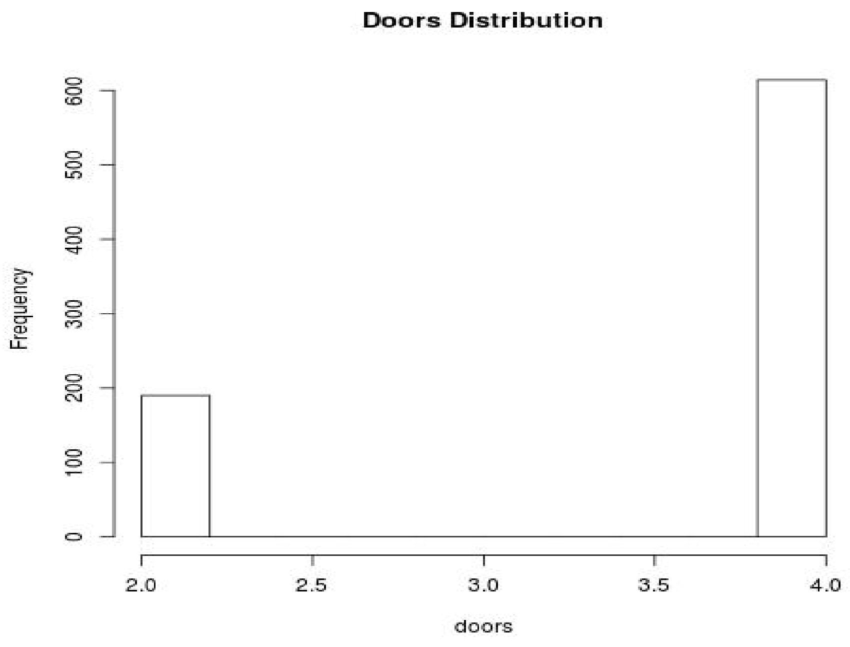


The Cylinders distribution shows that maximum number of used cars have 4 cylinders and number of user cars decreases with increase in number of cylinders.

Histogram for liters:

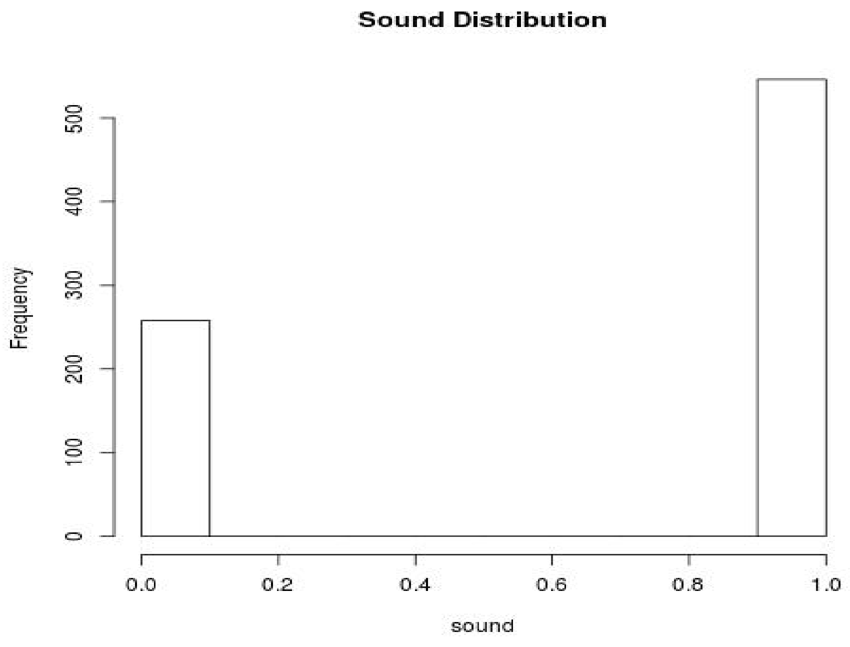


Histogram for Doors:



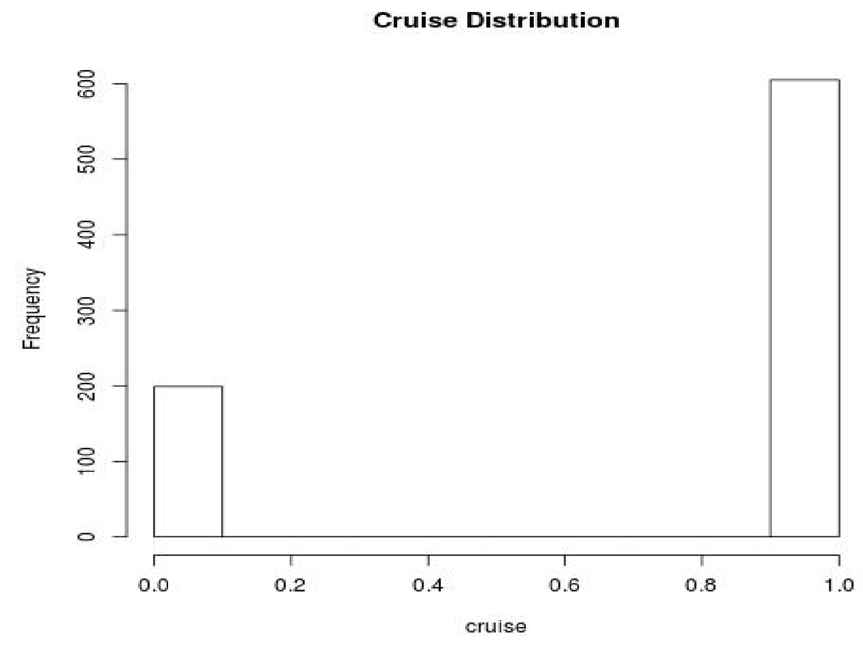
The door distribution shows that the cars have 4-doors and 2-doors. The number of cars with 2-doors are lot less than 4-doors.

Histogram for Sound:



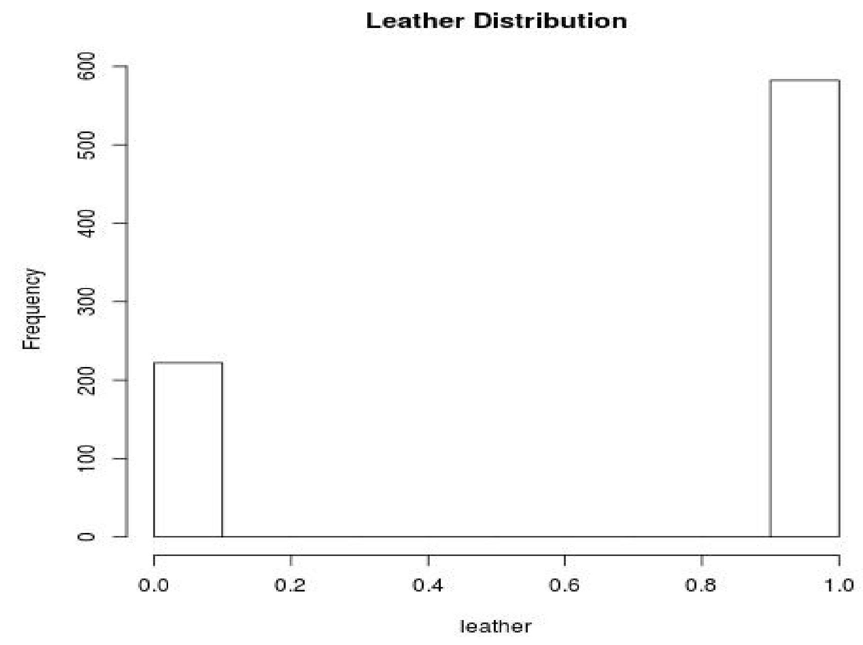
Sound is a binary distributed variable, indicatings its presence or absence.

Histogram for Cruise:



Cruise is a binary distributed variable, indicatings its presence or absence.

Histogram for leather:

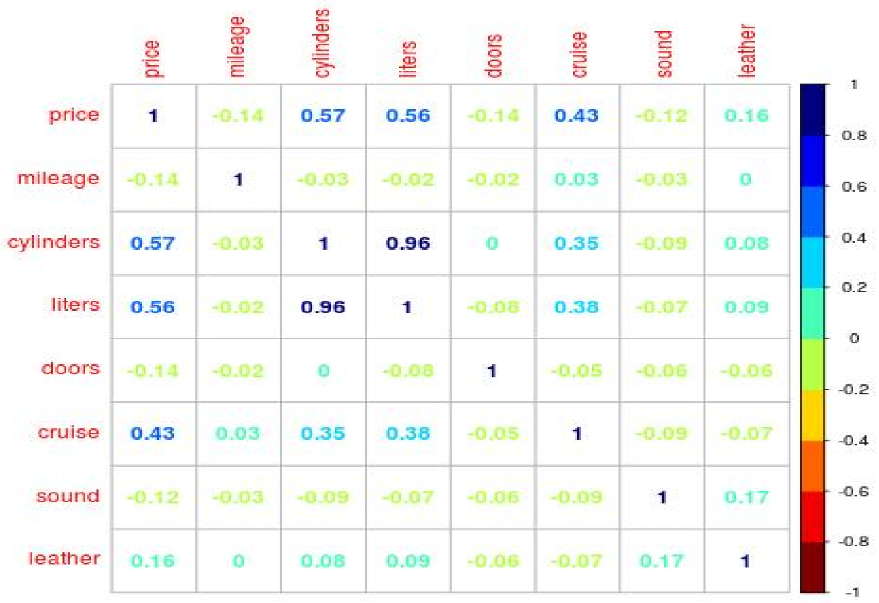


Cruise is a binary distributed variable, indicatings its presence or absence.

- Correlation for each pair of attributes:

For all non-categorical data, the Peasrson's correlation can be applied.

The following figure shows the Pearson Correlation for all the non-categorical attributes.



The correlation between price and cylinders is 0.56 and Price and liters is 0.57. Hence they can be said to be very much positively correlated, i.e, with the increase in cylinders there can be increase in price and vice-versa.

The highest correlation of 0.96 is between cylinders and liters which naturally is the case.

For correlation between a categorical and non-categorical attribute, I have used the dummy coding approach. Depending the number of categories, the same number of columns are created.

This technique can be used in R by contr.treatment(). Following snippet is given below:

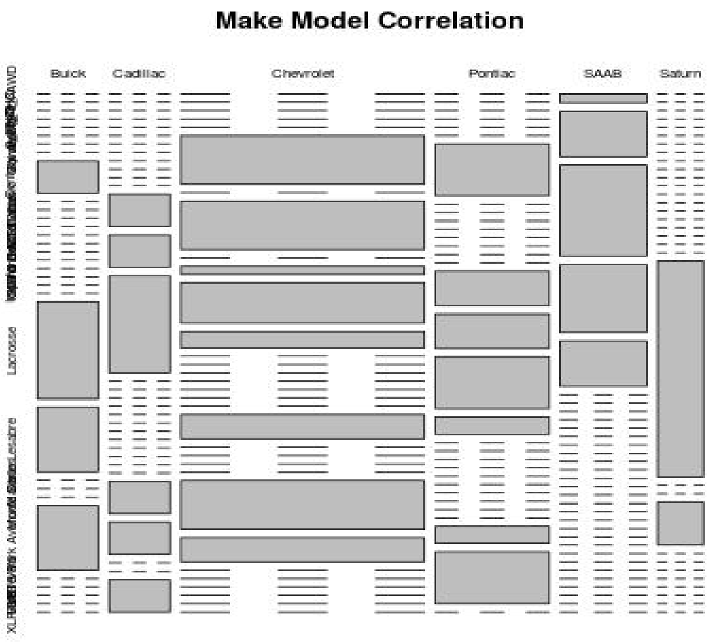
```
priceTrim = vehicles[,c(”price”,”trim”)]
priceTrim$trim = factor(priceTrim$trim)
contrasts(priceTrim$trim) ¡- c̈ontr.treatment¨
summary(lm(priceTrim$price    priceTrim$trim,data=priceTrim))
```

The v́ehiclesís the variable which contains the complete data.

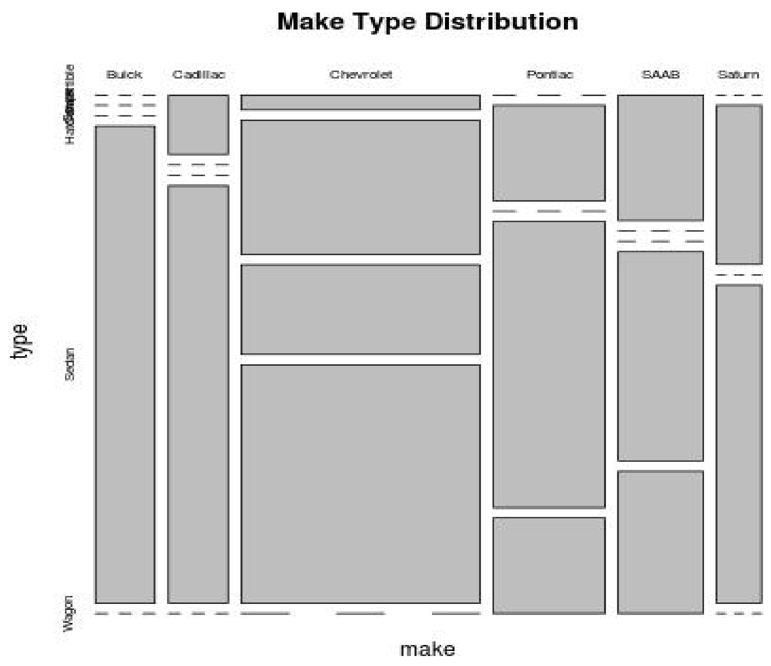
The above commands returns the adjusted R^2 to be

Similarly, other categorical and non-categorical variables are compared.

The correlation between two categorical variables are compared using a mosaic plot. The mosaic plot can indicate whether two nominal variables are correlated. It does not give a numerical value but the correlation can be inferred. Following is an example:



Visualizing a mosaic plot can be started either from the bottom or top. If there are multiple overlaps at levels (categories) of an attribute with respect to another attribute then such attributes are non-correlated. If there are lesser overlaps or no overlaps then the attributes can be termed as correlated or highly correlated respectively. Below is the example for less correlation



From the mosaic plot following are the observed results:

  - Model is correlated to Make and Trim and not correlated to Type.
  - Make is correlated to Trim but not Type.
  - Type and Trim show a correlation.

- Building a table for every possible model:

  The attributes considered for the models are:

  - **Model**
  - **Type**
  - **Cylinder**
  - **Cruise**

  The main principle followed in the above feature selection is from [2].
  "The Good feature subsets are highly correlated to the classification, yet uncorrelated to each other".
  The other reference for variable selection is [4].

  The reasons for choosing these model are:

  - Model was chosen because the it has very high Adjusted R^2 value by dummy coding. Eliminated the attribute make because it shows a less adjusted R^2 with price and it is correlated to the model.
  - Cylinder is highly correlated to price. The litre attribute is not considered because it has a correlation of 0.96 with cylinder.
  - Type is considered because it is uncorrelated to model and has a good Adjusted R^2 value with price.
  - cruise It shows a promising correlation of 0.43 with price.
  - Other attributes which show a very low correlation with price like Sound,door are not considered.

  Referred the webpage [1] to create all the possible models for a set of attributes. Following are the observed results.

Model	adjustedR2	p-value
price $\tilde{1}$ + model + cruise + type + cylinders	0.9571197	0
price $\tilde{1}$ + cruise + type + cylinders	0.639919	2.80E-174
price $\tilde{1}$ + model + type + cylinders	0.9571712	0
price $\tilde{1}$ + type + cylinders	0.6207491	2.41E-166
price $\tilde{1}$ + model + cruise + cylinders	0.9474086	0
price $\tilde{1}$ + cruise + cylinders	0.3824109	5.51E-85
price $\tilde{1}$ + model + cylinders	0.9474725	0
price $\tilde{1}$ + cylinders	0.3230163	3.45E-70
price $\tilde{1}$ + model + cruise + type	0.9566213	0
price $\tilde{1}$ + cruise + type	0.4080609	1.82E-89
price $\tilde{1}$ + model + type	0.9566740	0
price $\tilde{1}$ + type	0.30358190	2.833E-62
price $\tilde{1}$ + model + cruise	0.94685580	0
price $\tilde{1}$ + cruise	0.18461759	1.12E-37
price $\tilde{1}$ + model	0.9469184	0

  The p-value indicates the current considered sample of data is consistent with a null hypothesis.

The R^2 value is the measure of closeness(variance) of data to the fitted regression line.

Low p-value and low R^2: A low R^2 may be favorable in cases where a little change in variance matters. A low-p value indicates the null hypothesis may be true.

High p-value and low R^2: The results obtained may have to be either further examined using residual plots or plane reject the hypothesis.

High p-value and High R^2: This gives a clue that the alternate hypothesis is correct.

Low p-value and high R^2: It is the best possible situation where in we can get the evidence of the hypothesis with a high R^2.

3 Answer for Question 3

3.1 Part 1

- The errors in the data include issues like Pin codes are entered as less than five digits or six digits and greater without a hyphen(-).

  Eg:

  Using the below query we can spot the errors:

  SELECT pincode
  FROM User
  WHERE (LEN(pincode) ¿ 5 AND pincode NOT LIKE 'OR (LEN(pincode) ¡ 5)

  There are few pin codes which have hyphen(-) in them. All of them are nine-digit codes and hence seem correct.

  Resolution: Truncate the pincode data with excess length to a five digit number.

- Another issue spotted was a User of age around 18 had **retired** and similar age occupation discrepancies of others.

  Resolution: Changing the occupation of the user to **others**

3.2 Part 2

Assuming Scientists to be computer scientists.

The queries written are:

- SELECT rating,count(*) FROM Users M
  inner join ratings on ratings.userId = M.userId
  WHERE M.occupation=15 and M.Gender='M'
  AND ratings.MovieID IN (SELECT r.MovieID
  FROM Users F inner join ratings r on r.userId = F.userId
  WHERE F.occupation=15 and F.Gender='F')
  group by rating;

rating	count
1	498
2	1402
3	3676
4	6004
5	4169

- SELECT rating,count(*) FROM Users M
  inner join ratings on ratings.userId = M.userId
  WHERE M.occupation=15 and M.Gender='F'
  AND ratings.MovieID IN (SELECT r.MovieID
  FROM Users F inner join ratings r on r.userId = F.userId
  WHERE F.occupation=15 and F.Gender='M')
  group by rating;

rating	count
1	104
2	284
3	942
4	1610
5	1108

  These queries give the number of ratings that were given by male and female computer scientists.

- Analysing the above two table, Following is the distribution:

Rating	Male	Female
1	3.16%	2.5%
2	8.89%	7.01%
3	23.34%	23.2%
4	38.11%	39.77%
5	26.4%	27.37%

  Looks like the male and female scientists have given roughly the similar number of ratings to the movies they both watch.

3.3 Part 3

SELECT count(*)
FROM ratings INNER JOIN Users
ON ratings.userId = Users.userId
WHERE Gender='M';

OUTPUT : 753769

SELECT count(*)
FROM ratings INNER JOIN Users
ON ratings.userId = Users.userId
WHERE Gender='F';

OUTPUT : 246440

Men have rated more movies than woman.

3.4 Part 4

```
SELECT count(*)
FROM ratings INNER JOIN Movies
ON ratings.MovieId = Movies.id
WHERE genres LIKE '%Comedy%'
```

Comedy : 355080

The genre **comedy** has been rated the most.

3.5 Part 5

- Clustering Algorithm:

  Trees are used to implement the Agglomerative algorithm.

  - Initially, all the leaf nodes of the tree are considered as clusters.
  - Slowly clusters are grown converging towards a single node in the tree .
  - Each cluster has a cluster representative which are decided using the centroid approach, i.e, calculating the average of the (x,y points).
  - Depending on the number of iterations, 'n' number of clusters are formed. Number of iterations acts as a threshold

- Degree Of Similarity:

  Clustering on Age,profession gives us the insight that how profession and age affects a user's choice of Genre and rating.

  Degree Of Similarity:

  Clustering on genre and rating gives the popularity of each genre within the community(age,profession,sex).

  For Eg:

```
Cluster Representation :
Mystery,
The average rating :3.162109375
Total Males in the cluster : 527
Total Females in the cluster : 145
For age range 1
People in the age range 12
For age range 18
People in the age range 134
For age range 50
People in the age range 63
For age range 35
People in the age range 118
For age range 56
People in the age range 25
For age range 25
People in the age range 232
```

```
For age range 45
People in the age range 88
```

The above example shows that the 'Mystery' genre is very famous in the age range 45 and the overall average rating is 3.16 in the Cluster.

- Referred Clustering approach from the textbook [5]

  The distance function for clustering age,profession is a cosine distance between age and profession as these values represent a range and not the actual age and the other field occupation just represents a profession.

  The distance function for clustering are genre and rating. The distance mesaurement for genre is Jaccard similarity and rating is directly subtracted and its reciprocal is taken because lesser the difference in rating greater is the similarity. Cumulative the sum of these two is taken as the distance.

4 Appendix A

```
create table Movies(id int, name varchar(100), genres varchar(50))


LOAD DATA LOCAL INFILE '/media/jus-mine/OS/Study/Data Mining/Assignment2/ml-1m/ml-1m/movies.dat' INTO TA


create table Users(UserID integer,Gender char(1),Age integer,Occupation integer,ZipCode varchar(11))

LOAD DATA LOCAL INFILE '/media/jus-mine/OS/Study/Data Mining/Assignment2/ml-1m/ml-1m/users.dat' INTO TAB


create table ratings(UserID int,MovieID int,rating int,Timestamp int);

LOAD DATA LOCAL INFILE '/media/jus-mine/OS/Study/Data Mining/Assignment2/ml-1m/ml-1m/ratings.dat' INTO T
LINES TERMINATED BY '\n';


I had also created partitions later on the table for speeding up the queries.

Also have indexed the keys later.

Male Scientists

Select rating,count(*) from ratings inner join Users on ratings.userId = Users.userId  where occupation=

Female Scientists

Select rating,count(*) from ratings inner join Users on ratings.userId = Users.userId  where occupation=

mysql> Select rating,count(*) from Users M inner join ratings on ratings.userId = M.userId
-> WHERE M.occupation=15 and M.Gender='M'
-> AND ratings.MovieID IN ( select r.MovieID
-> from Users F inner join ratings r on r.userId = F.userId
-> WHERE F.occupation=15 and F.Gender='F')
-> group by rating;
```

```
+--------+----------+
| rating | count(*) |
+--------+----------+
|      1 |      498 | -> 3.16%
|      2 |     1402 | -> 8.9%
|      3 |     3676 | -> 23.34%
|      4 |     6004 | -> 38.12%
|      5 |     4169 | -> 26.4%
+--------+----------+



5 rows in set (47.03 sec)

mysql> Select rating,count(*) from Users M inner join ratings on ratings.userId = M.userId WHERE M.occu
+--------+----------+
| rating | count(*) |
+--------+----------+
|      1 |      104 | -> 2.5%
|      2 |      284 | -> 7.01%
|      3 |      942 | -> 23.2%
|      4 |     1610 | -> 39.77%
|      5 |     1108 | -> 27.37%
+--------+----------+
5 rows in set (13.41 sec)
```

5 Appendix B

```
LOAD DATA LOCAL INFILE '/media/jus-mine/OS/Study/Data Mining/Assignment2/vehicle2.txt' INTO TABLE vehic



desc vehicles
-> ;
+-----------+-----------------------+------+-----+---------+-------+
| Field     | Type                  | Null | Key | Default | Extra |
+-----------+-----------------------+------+-----+---------+-------+
| price     | decimal(8,2)          | YES  |     | NULL    |       |
| mileage   | mediumint(8) unsigned | YES  |     | NULL    |       |
| make      | varchar(20)           | YES  |     | NULL    |       |
| model     | varchar(20)           | YES  |     | NULL    |       |
| trim      | varchar(20)           | YES  |     | NULL    |       |
| type      | varchar(20)           | YES  |     | NULL    |       |
| cylinders | tinyint(3) unsigned   | YES  |     | NULL    |       |
| liters    | decimal(3,1)          | YES  |     | NULL    |       |
| doors     | tinyint(3) unsigned   | YES  |     | NULL    |       |
| cruise    | tinyint(1)            | YES  |     | NULL    |       |
| sound     | tinyint(1)            | YES  |     | NULL    |       |
| leather   | tinyint(1)            | YES  |     | NULL    |       |
+-----------+-----------------------+------+-----+---------+-------+
mysql> Select cylinders,count(*)
-> from vehicles
-> group by cylinders
-> ;
+-----------+----------+
| cylinders | count(*) |
+-----------+----------+
|         4 |      394 |
|         6 |      310 |
|         8 |      100 |
+-----------+----------+
3 rows in set (0.06 sec)
```

6 Appendix C

```
The Agglomerative Code for Clustering:

The entity which holds the data obtained from the database:

package dataMining.AssignmentTwo;

import java.util.Arrays;
import java.util.List;

public class Entity {

int UserID,Age,Occupation,movieId;
char sex;
double rating;

List<String> genres;

public Entity(int userID, int age, int occupation, double rating,
String genres,int movieID) {

UserID = userID;
Age = age;
Occupation = occupation;
this.rating = rating;
this.genres = Arrays.asList(genres.split("\\|"));
this.movieId = movieID;
}



public Entity(double rating, List<String> genres) {
super();
this.rating = rating;
this.genres = genres;
}



public Entity(int age, int occupation, double rating, String genres,char sex) {
super();
Age = age;
Occupation = occupation;
this.rating = rating;
this.genres = Arrays.asList(genres.split("\\|"));
this.sex = sex;
}



public Entity(int age, int occupation) {
Age = age;
Occupation = occupation;
}

public int getUserID() {
return UserID;
}

public int getAge() {
return Age;
}

public int getOccupation() {
return Occupation;
}

public double getRating() {
return rating;
}

public List<String> getGenres() {
return genres;
}

public int getMovieId() {
return movieId;
}
}
```

The common mysql connector for both clustering algorithms:

```
package dataMining.AssignmentTwo;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;
```

```
public class MysqlConnector {

Connection con = null;
Statement st = null;
ResultSet rs = null;
String url = "", user = "",passwd = "";

public MysqlConnector(){

url = "jdbc:mysql://localhost:3306/MovieLens";
user = "root";
passwd = "quititdude";
}

public void connect() throws SQLException{

con = DriverManager.getConnection(url,user,passwd);

}

public List<Cluster> getData() throws SQLException {

List<ResultSet> rsList = new ArrayList<ResultSet>();
String query;
for(int i = 0; i < 50; i++) {
query = String.format("Select age,occupation,rating,genres,Gender from CongregatedData LIMIT 1000 OFFSE
st = con.createStatement();
rs = st.executeQuery(query);
rsList.add(rs);
}
return getListFromResultSet(rsList);
}

public List<Cluster> getListFromResultSet(List<ResultSet> rsList) throws SQLException{

List<Cluster> items = new ArrayList<Cluster>();
Entity entity;Cluster cluster;
for(ResultSet rs1: rsList) {
while (rs1.next()) {
//entity = new Entity(rs1.getInt(1),rs1.getInt(2),rs1.getInt(3),rs1.getInt(4),rs1.getString(5),rs1.getI
entity = new Entity(rs1.getInt(1),rs1.getInt(2),rs1.getDouble(3),rs1.getString(4),rs1.getString(5).char
cluster = new Cluster();
cluster.clusterRepresentaive = entity;
items.add(cluster);
}
}
return items;
}


public void disconnect() throws SQLException{
con.close();
}
}
```

The Clustering Class used for both clustering algorithms:

```
package dataMining.AssignmentTwo;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

public class Cluster {

Entity cluster Representaive;
Cluster left;
Cluster right;

}
```

Agglomerative Clustering for Age and Profession

```
package dataMining.AssignmentTwo;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

public class Cluster {

Entity clusterRepresentaive;
Cluster left;
Cluster right;

}
```

The Clustering Algorithm for Age and Profession:

```
package dataMining.AssignmentTwo;


import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;

public class AgglomerativeForAgeProfession {
```

```
class XYPairs {

int index1;
int index2;

public XYPairs(int x,int y) {
index1 = x;
index2 = y;
}
}


void calculateCosineDistancesOfClusters(List<Cluster> clusters) {

List<XYPairs> xyPairs = new ArrayList<XYPairs>();
double max = Double.MIN_VALUE,newMax;
int i=0,j=0,index = -1;
Map<Integer,Boolean> visited = new HashMap<Integer,Boolean>();
Entity ent1,ent2;
for(Cluster cl1: clusters.subList(0,clusters.size() - 2)) {
j=i+1;
ent1 = cl1.clusterRepresentaive;
max = Double.MIN_VALUE;
for(Cluster cl2 : clusters.subList(i+1,clusters.size() - 1)) {
ent2 = cl2.clusterRepresentaive;
newMax = calculateCosineDistances(ent1,ent2);
if(newMax > max && (!visited.containsKey(j) && !visited.containsKey(i))) {
max = newMax;
index = j;
}
j++;
}
if( !visited.containsKey(index) && !visited.containsKey(i) && index != -1) {
visited.put(i,true);
visited.put(index,true);
xyPairs.add(new XYPairs(i,index));
}
i++;
}
prepareCluster(clusters,xyPairs);
}

void prepareCluster(List<Cluster> clusters,List<XYPairs> xyPairs) {

List<Cluster> removalList = new ArrayList<Cluster>();
for(XYPairs pair: xyPairs){

Cluster cluster1 = clusters.get(pair.index1);
Cluster cluster2 = clusters.get(pair.index2);
int ageMedian = (cluster1.clusterRepresentaive.Age + cluster2.clusterRepresentaive.Age)/2;
int occMedian = (cluster1.clusterRepresentaive.Occupation + cluster2.clusterRepresentaive.Occupation)/2
Entity en = new Entity(ageMedian,occMedian);
```

```
Cluster newCluster = new Cluster();
newCluster.clusterRepresentaive = en;
newCluster.left = cluster1;
newCluster.right = cluster2;
clusters.add(newCluster);
removalList.add(cluster1);removalList.add(cluster2);
}
clusters.removeAll(removalList);
}

void getAllLeavesInACluster(Cluster cl,List<Entity> entities) {

if(cl == null)
return;
if(cl.left == null && cl.right == null){
entities.add(cl.clusterRepresentaive);
}
getAllLeavesInACluster(cl.left,entities);
getAllLeavesInACluster(cl.right,entities);
}


double calculateCosineDistances(Entity it,Entity it2) {//,List<XYPairs> xyPairs) {

double distanceMat;
distanceMat = ((it.Age * it2.Age) + (it.Occupation * it2.Occupation))/(Math.sqrt(Math.pow(it.Age,2)+ Ma
return distanceMat;
}

void analyseCluster(Cluster cluster){

List<Entity> entities = new ArrayList<Entity>();
getAllLeavesInACluster(cluster,entities);
Map<String,Map<Double,Integer>> dict = new HashMap<String,Map<Double,Integer>>();
int count;
Map<Double,Integer> ratingsDict = new HashMap<Double,Integer>();

for(Entity en : entities){

for(String genre:en.genres) {

if(dict.containsKey(genre)) {
ratingsDict = dict.get(genre);
if(ratingsDict != null && ratingsDict.containsKey(en.rating)){
count = ratingsDict.get(en.rating);
ratingsDict.put(en.rating,count++);
}
else{
ratingsDict.put(en.rating,1);
}
}
else {
ratingsDict = new HashMap<Double,Integer>();
ratingsDict.put(en.rating,1);
dict.put(genre, ratingsDict);
}
}
}
int sum;
Double maxRating = Double.MIN_VALUE;
for(Entry<String, Map<Double,Integer>> entry: dict.entrySet()) {
System.out.println(" Genre: " + entry.getKey());
sum = 0;count = 0;
for(Entry<Double, Integer> innerEntry: entry.getValue().entrySet()) {
sum += innerEntry.getKey() * innerEntry.getValue();
count += innerEntry.getValue();
if(maxRating < innerEntry.getKey())
maxRating = innerEntry.getKey();
}
System.out.println(" No of People Rated the genre : " + sum);
System.out.println(" The average rating : " + (sum/count));
System.out.println(" The Max rating : " + maxRating);
}
}


List<Cluster> getData() throws SQLException {

MysqlConnector mysql = new MysqlConnector();
mysql.connect();
List<Cluster> items = mysql.getData();
mysql.disconnect();
return items;
}

public static void main(String[] args) throws SQLException{
AgglomerativeForAgeProfession agg = new AgglomerativeForAgeProfession();
List<Cluster> items = agg.getData();
for(int i = 0; i < 13 ; i++){
agg.calculateCosineDistancesOfClusters(items);
}
System.out.println("No of clusters :" + items.size());
for( Cluster cl : items )
agg.analyseCluster(cl);
}
}

The Clustering Algorithm for Genre and rating:

package dataMining.AssignmentTwo;


import java.sql.SQLException;
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;

public class Agglomerative {

class XYPairs {

int index1;
int index2;

public XYPairs(int x,int y) {
index1 = x;
index2 = y;
}
}

void calculateCosineDistancesOfClusters(List<Cluster> clusters) {

List<XYPairs> xyPairs = new ArrayList<XYPairs>();
double max = Double.MIN_VALUE,newMax;
int i=0,j=0,index = -1;
Map<Integer,Boolean> visited = new HashMap<Integer,Boolean>();
Entity ent1,ent2;
for(Cluster cl1: clusters.subList(0,clusters.size() - 2)) {
j=i+1;
ent1 = cl1.clusterRepresentaive;
max = Double.MIN_VALUE;
for(Cluster cl2 : clusters.subList(i+1,clusters.size() - 1)) {
ent2 = cl2.clusterRepresentaive;
//newMax = calculateCosineDistances(ent1,ent2);
newMax = calculateDistancesForGenreRatings(ent1,ent2);
if(newMax > max && (!visited.containsKey(j) && !visited.containsKey(i))) {
max = newMax;
index = j;
}
j++;
}
if( !visited.containsKey(index) && !visited.containsKey(i) && index != -1) {
visited.put(i,true);
visited.put(index,true);
xyPairs.add(new XYPairs(i,index));
}
i++;
}

prepareClusterForGenreRating(clusters,xyPairs);
}


void prepareClusterForGenreRating(List<Cluster> clusters,List<XYPairs> xyPairs) {

List<Cluster> removalList = new ArrayList<Cluster>();
```

```
for(XYPairs pair: xyPairs){

Cluster cluster1 = clusters.get(pair.index1);
Cluster cluster2 = clusters.get(pair.index2);
List<String> clusteredGenreList = new ArrayList<String>(cluster1.clusterRepresentaive.genres);
clusteredGenreList.retainAll(cluster2.clusterRepresentaive.genres);
double avgRating = (cluster1.clusterRepresentaive.rating + cluster2.clusterRepresentaive.rating)/2; //Ma
Entity en = new Entity(avgRating,clusteredGenreList);
Cluster newCluster = new Cluster();
newCluster.clusterRepresentaive = en;
newCluster.left = cluster1;
newCluster.right = cluster2;
clusters.add(newCluster);
removalList.add(cluster1);removalList.add(cluster2);
}
clusters.removeAll(removalList);
}


int getAllClusterEntities(List<Cluster> clusters) {

List<Entity> temp = new ArrayList<Entity>();
int count = 0;
for(Cluster cl: clusters) {
temp = new ArrayList<Entity>();
getAllLeavesInACluster(cl,temp);
count += temp.size();
}
return count;
}

void getAllLeavesInACluster(Cluster cl,List<Entity> entities) {

if(cl == null)
return;
if(cl.left == null && cl.right == null) {
entities.add(cl.clusterRepresentaive);
}
getAllLeavesInACluster(cl.left,entities);
getAllLeavesInACluster(cl.right,entities);
}

double calculateDistancesForGenreRatings(Entity it,Entity it2) {
int similarGenrecount = 0;
List<String> common = new ArrayList<String>(it.genres);
//Collections.copy(common,it.genres);
common.retainAll(it2.genres);
similarGenrecount = common.size();
double ratingDiff = Math.abs(it.rating - it2.rating);
if(similarGenrecount == 0) return 0;
double score =  similarGenrecount/(it.genres.size() + it2.genres.size() - similarGenrecount);
if(ratingDiff == 0){
score += 1.3; //kept as a high score because similar rating for same genre can be a top priority
}
else
score += (1/ratingDiff);
return score;
}

void analyseClusterForGenreRating(Cluster cluster){

List<Entity> entities = new ArrayList<Entity>();
getAllLeavesInACluster(cluster,entities);

System.out.println(" Cluster Representation :");

Entity ent = cluster.clusterRepresentaive;

for(String genre: ent.genres){
System.out.print(genre+ ",");
}
System.out.println();
System.out.println("The average rating :" + ent.rating);

Map<String,Map<Integer,Integer>> dict = new HashMap<String,Map<Integer,Integer>>();
Map<Integer,Integer> occDict = new HashMap<Integer,Integer>();
Map<Integer,Integer> ageDict = new HashMap<Integer,Integer>();
int maleCount = 0,femaleCount = 0;
int occCount = 0,ageCount = 0;
for(Entity en : entities) {

if(en.sex == 'M')
maleCount++;
else
femaleCount++;

if(occDict.containsKey(en.Occupation)){
occCount = occDict.get(en.Occupation);
occDict.put(en.Occupation,++occCount);
}
else
occDict.put(en.Occupation,1);

if(ageDict.containsKey(en.Age)) {
ageCount = ageDict.get(en.Age);
ageDict.put(en.Age,++ageCount);
}
else
ageDict.put(en.Age,1);
}
System.out.println(" Total Males in the cluster : " + maleCount);
System.out.println(" Total Females in the cluster : " + femaleCount);
for(Entry<Integer, Integer> entry: ageDict.entrySet()){
System.out.println(" For age range " +  entry.getKey());
System.out.println(" People in the age range " +  entry.getValue());
}
```

```
}

List<Cluster> getData() throws SQLException {

MysqlConnector mysql = new MysqlConnector();
mysql.connect();
List<Cluster> items = mysql.getData();
mysql.disconnect();
return items;
}

public static void main(String[] args) throws SQLException{
Agglomerative agg = new Agglomerative();
List<Cluster> items = agg.getData();

for(int i = 0; i < 13 ; i++){
agg.calculateCosineDistancesOfClusters(items);
}
System.out.println("No of clusters :" + items.size());
for( Cluster cl : items )
agg.analyseClusterForGenreRating(cl);

System.out.println(agg.getAllClusterEntities(items));
}

}
```

7 Appendix D

Example Anova output:

```
Analysis of Variance Table
Model 1: price ~ 1 + model + cruise + type + cylinders Model
2: price ~ 1 + model + cruise + type + cylinders   Res.Df        RSS Df Sum of Sq F Pr(>F)
1    766 3209420957
2    766 3209420957  0         0
Analysis of Variance Table
Model 1: price ~ 1 + model + cruise + type + cylinders Model 2: price ~ 1 + cruise + type + cylinders
1    766 3.2094e+09                                   2    797 2.8041e+10 -31 -2.4832e+10 191.18 < 2
*** --- Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1
```

8 Credits

Have discussed the algorithm with **Aniket Gaekwad**

References

[1] R: Calculating all possible linear regression models for a given set of predictors — ”r” you ready? https://ryouready.wordpress.com/2009/02/06/

`r-calculating-all-possible-linear-regression-models-for-a-given-set-of-predictors/`. Accessed: 2015-02-15.

[2] Mark A Hall. *Correlation-based feature selection for machine learning.* PhD thesis, The University of Waikato, 1999.

[3] Ronald R Hocking. A biometrics invited paper. the analysis and selection of variables in linear regression. *Biometrics*, pages 1–49, 1976.

[4] Shonda Kuiper. Introduction to multiple regression: How much is your car worth? *Journal of Statistics Education*, 16(3), 2008.

[5] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, et al. *Introduction to data mining*, volume 1. Pearson Addison Wesley Boston, 2006.