

DATA MINING REPORT 2

- M Nimesh Reddy
- K Manpreeth Sai
- Akshay Babu
- Bobby Aloysius Johnson
- Balmukund Sinha

Aim: To Apply Data Mining Techniques (Clustering, Classification, Association) on the Pre processed data set

1. Classification:

Classification is a form of data analysis that extracts models describing important data classes. Such models, called classifiers, predict categorical (discrete, unordered) class labels.

Classification is usually described as a two step process. In the first step, we build a classification model based on previous data. In the second step, we determine if the model's accuracy is acceptable, and if so, we use the model to classify new data.

In order to build a classification model, we need to analyze the training set which is made up of the database tuples and their associated class labels.

After preprocessing , our data set doesn't contain any missing class label for any of the tuples. So , we are using Supervised Learning .

1.1 DECISION TREE :

Decision tree induction is the learning of decision trees from class-labeled training tuples. A decision tree is a flowchart-like tree structure, where each internal node (nonleaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or *terminal node*) holds a class label.

The topmost node in a tree is the root node. Internal nodes are denoted by rectangles, and leaf nodes are denoted by ovals.

An attribute selection measure is a heuristic for selecting the splitting criterion that "best" separates a given data partition D of class-labeled training tuples into individual classes.

1.1.1 Decision Tree Advantages:

- The construction of decision tree classifiers does not require any domain knowledge or parameter setting, and therefore is appropriate for exploratory knowledge discovery.
- Decision trees can handle multidimensional data.
- Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
- The learning and classification steps of decision tree induction are simple and fast.
- Decision tree classifiers have good accuracy.

1.1.2 Decision Tree Induction:

Most algorithms for decision tree induction also follow a top-down approach, which starts with a training set of tuples and their associated class labels. The training set is recursively partitioned into smaller subsets as the tree is being built.

Attribute selection method employs an attribute selection measure such as information gain or the Gini index or the Gain Ratio.

Attribute selection measures are also known as splitting rules because they determine how the tuples at a given node are to be split. The attribute having the best score for the measure is chosen as the splitting attribute for the given tuples.

Three Popular Attribute Selection methods are used for the data set namely

- Information Gain
- Gain Ratio
- Gini Index

Among the three popular methods, Information Gain is biased towards multivalued attributes, so we cannot use this for our data set as it has a multivalued attribute (i.e., color of the glasses). Gini Index is also biased towards multivalued attributes and has a difficulty when the number of classes is large. Here the number of classes is large, so we cannot use Gini Index as attribute selection measure.

The Gain Ratio adjusts the bias towards multivalued attributes. The decision tree depth was set as 20 initially and the following decision tree is obtained as shown in the screenshot. Later the height was reduced to 8 (for visualizing the tree) and it is also shown below. Cross validation is applied for model with partition size as 10.

Pruned trees tend to be smaller and less complex and, thus, easier to comprehend. They are usually faster and better at correctly classifying independent test data. There are two common approaches to tree pruning:

- prepruning
- postpruning.

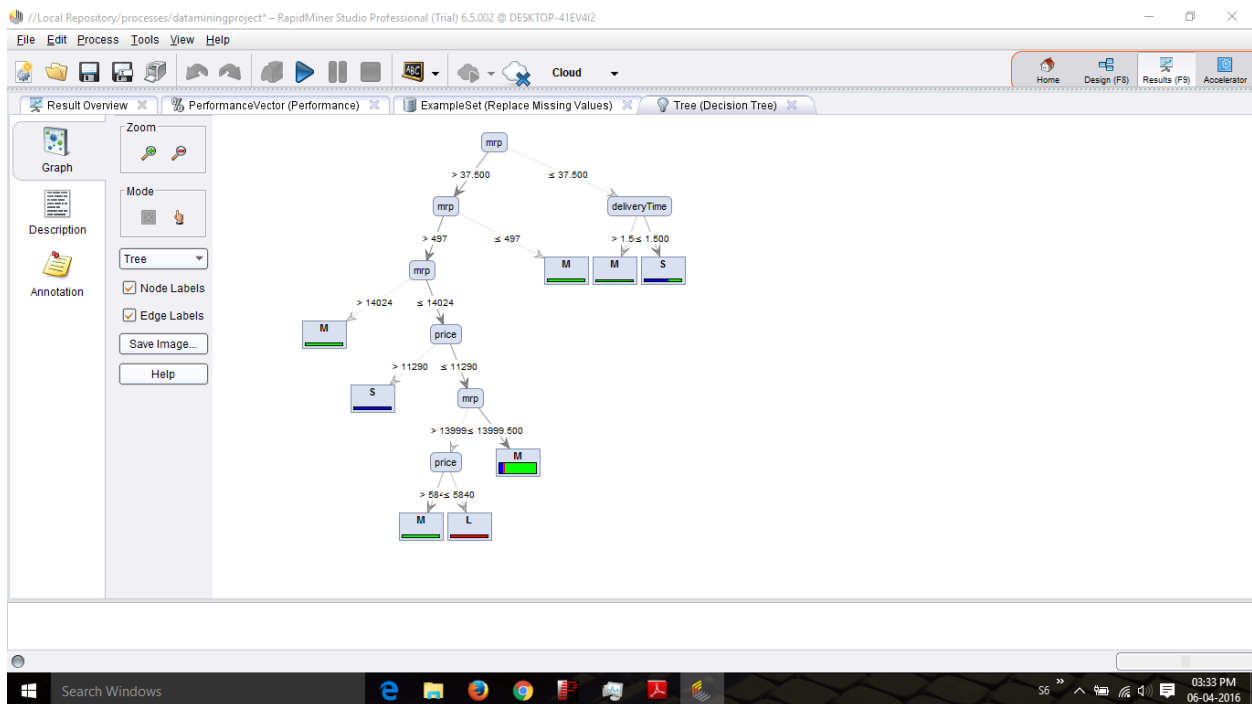
Prepruning:

In the **prepruning** approach, a tree is “pruned” by halting its construction early. Upon halting, the node becomes a leaf. The leaf may hold the most frequent class among the subset tuples or the probability distribution of those tuples.

There are difficulties, however, in choosing an appropriate threshold. High thresholds could result in oversimplified trees, whereas low thresholds could result in very little simplification. This yielded a very small tree when applied with our tool .

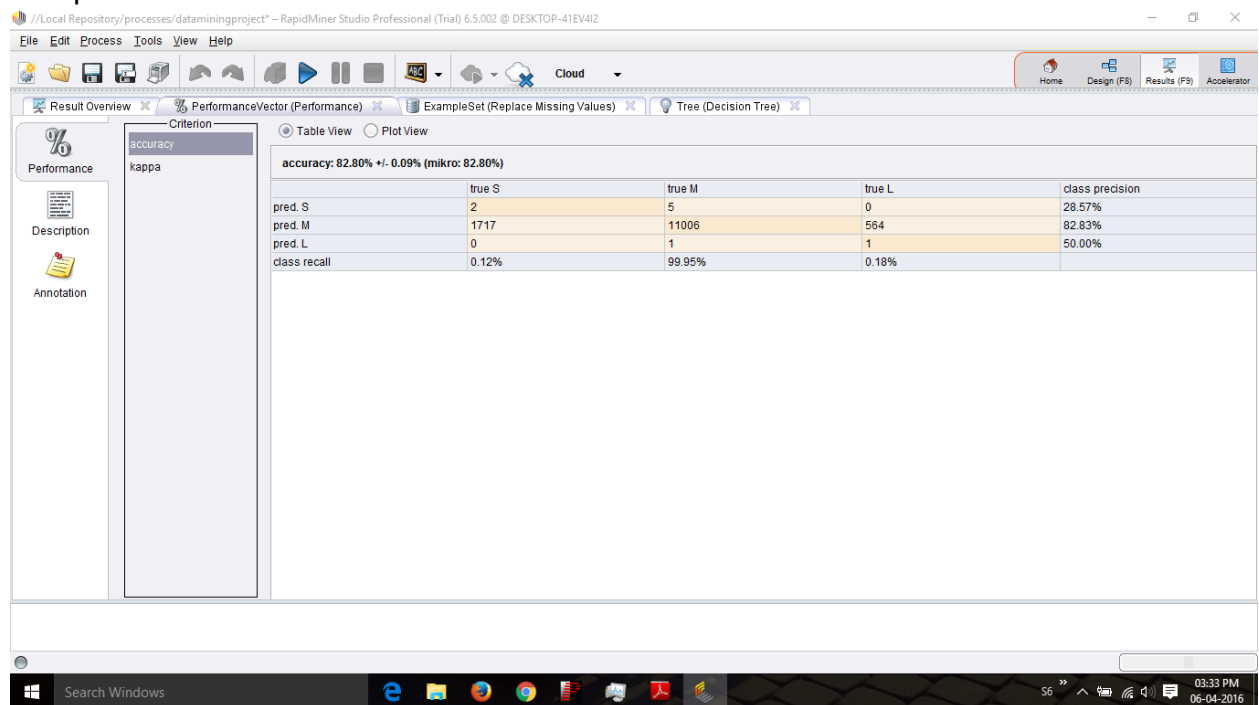
Postpruning:

The second and more common approach is **postpruning**, which removes subtrees from a “fully grown” tree. A subtree at a given node is pruned by removing its branches and replacing it with a leaf. The leaf is labeled with the most frequent class among the subtree being replaced. The postpruning method is applied and a decision tree as shown in the screenshot below is obtained.



1.1.4 Model Evaluation

The performance of the decision tree is obtained through performance operator in rapidminer tool . The results obtained are as shown below.



1.2 Naïve Bayes' Theorem:

Naïve Bayes' theorem is based on probability and decision theory to build a classification model. Let X be a data tuple. In Bayesian terms, X is considered evidence. As usual, it is described by measurements made on a set of n attributes. Let H be some hypothesis such as that the data tuple X belongs to a specified class C . For classification problems, we want to determine $P(H|X)$, the probability that the hypothesis H holds given the "evidence" or observed data tuple X . In other words, we are looking for the probability that tuple X belongs to class C , given that we know the attribute description of X .

$P(H)$, $P(X/H)$, and $P(X)$ may be estimated from the given data, as we shall see next. Bayes' theorem is useful in that it provides a way of calculating the posterior probability, $P(H/X)$, from $P(H)$, $P(X/H)$, and $P(X)$. Bayes' theorem is

$$P(H/X) = P(X/H) * P(H) / P(X)$$

The top screenshot shows the 'SimpleDistribution' model description in RapidMiner Studio. The model is a distribution model for label attribute size. The description lists three classes: Class S (0.129) with 11 distributions, Class M (0.813) with 11 distributions, and Class L (0.042) with 11 distributions.

The bottom screenshot shows the performance metrics for the 'SimpleDistribution' model. The 'Table View' is selected, displaying the following data:

	true S	true M	true L	class precision
pred. S	978	488	35	65.16%
pred. M	677	10042	298	91.15%
pred. L	64	281	232	40.21%
class recall	56.89%	92.89%	41.06%	

2. ASSOCIATION

Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository.

2.1 ADVANTAGES

In data mining, association rules are useful for analyzing and predicting customer behavior. They play an important part in shopping basket data analysis, product clustering, catalog design and store layout.

An association rule is an implication of the form

$$A \Rightarrow B$$

where $A \subset I$, $B \subset I$, $A \neq \phi$, $B \neq \phi$, and $A \cap B = \phi$. where I is itemset.

The rule $A \Rightarrow B$ holds in the transaction set D with support s and confidence c . where D is collection of itemset I .

- **SUPPORT** for $A \Rightarrow B$, s is the percentage of transactions in D that contain $A \cup B$ (i.e., the union of sets A and B say, or, both A and B). This is taken to be the probability, $P(A \cup B)$.

$$\text{support}(A \Rightarrow B) = P(A \cup B)$$

- **CONFIDENCE** for $A \Rightarrow B$, c is the percentage of transactions in D containing A that also contain B . This is taken to be the conditional probability, $P(B|A)$

$$\text{confidence}(A \Rightarrow B) = P(B|A).$$

- **Rules**

In general, association rule mining can be viewed as a two-step process:

1. Find all frequent itemsets: By definition, each of these itemsets will occur at least as frequently as a predetermined minimum support count, min sup .
2. Generate strong association rules from the frequent itemsets: By definition, these rules must satisfy minimum support and minimum confidence.

if $\text{support}(A \Rightarrow B) = P(A \cup B) > \text{minimum support}$

$\text{confidence}(A \Rightarrow B) = P(B|A) > \text{minimum confidence}$

rule $A \Rightarrow B$ is strong

2.2 FREQUENT PATTERN-GROWTH

FP-Growth method adopts a divide and conquer strategy. It compresses the database representing frequent items into a frequent-pattern tree, or FP-tree, which retains the itemset association information. It then divides the compressed database into a set of *conditional databases* (a special kind of projected database), each associated with one frequent item or “pattern fragment,” and mines each such database separately.

The data set cannot be directly used to generate rules using FP-Growth since there are many non-binary attributes. So, using “NOMINAL TO BINOMIAL” operator, the attributes are modified, and FP-Growth is applied followed by “create association rules” operator. The following rules were obtained as shown in the

screenshot.

The screenshot displays the RapidMiner Studio Professional interface. The main window is titled "AssociationRules (Create Association Rules)". The left sidebar contains icons for Data, Graph, Description, and Annotation. The central pane shows a list of association rules with their confidence values. The rules are as follows:

```
Association Rules
[size = M --> [discount] (confidence: 0.965)
[size = M --> [deliveryTime, discount] (confidence: 0.965)
[deliveryTime, size = M --> [discount] (confidence: 0.965)
[size = M --> [price, discount] (confidence: 0.965)
[size = M --> [mrp, discount] (confidence: 0.965)
[size = M --> [deliveryTime, price, discount] (confidence: 0.965)
[deliveryTime, size = M --> [price, discount] (confidence: 0.965)
[size = M --> [deliveryTime, mrp, discount] (confidence: 0.965)
[deliveryTime, size = M --> [mrp, discount] (confidence: 0.965)
[size = M --> [price, mrp, discount] (confidence: 0.965)
[size = M --> [deliveryTime, price, mrp, discount] (confidence: 0.965)
[deliveryTime, size = M --> [price, mrp, discount] (confidence: 0.965)
[price, size = M --> [discount] (confidence: 0.965)
[mrp, size = M --> [discount] (confidence: 0.965)
[price, size = M --> [deliveryTime, discount] (confidence: 0.965)
[deliveryTime, price, size = M --> [discount] (confidence: 0.965)
[mrp, size = M --> [deliveryTime, discount] (confidence: 0.965)
[deliveryTime, mrp, size = M --> [discount] (confidence: 0.965)
[price, size = M --> [mrp, discount] (confidence: 0.965)
[mrp, size = M --> [price, discount] (confidence: 0.965)
[price, mrp, size = M --> [discount] (confidence: 0.965)
[price, size = M --> [deliveryTime, mrp, discount] (confidence: 0.965)
```

To make the rules more precise and accurate, first discretization (binning) was performed on the numeric attributes (price, discount, mrp and cashback). All the attributes are converted to numeric and then to binomial which produced transaction data set with all distinct values in the dataset. FP-Growth operator was applied and create association rules was attached for which rules are obtained as shown below.

The screenshot displays the RapidMiner Studio Basic interface. The main window is titled "AssociationRules (Create Association Rules)". The left sidebar contains icons for Data, Graph, Description, and Annotation. The central pane shows a list of association rules with their confidence values. The rules are as follows:

```
Association Rules
[mrp = range1 [-∞ - 4628.571], codAvailable = TRUE, size = M --> [price = range1 [-∞ - 1985.714]] (confidence: 0.950)
[mrp = range1 [-∞ - 4628.571], codAvailable = TRUE --> [price = range1 [-∞ - 1985.714]] (confidence: 0.953)
[mrp = range1 [-∞ - 4628.571], inStock = TRUE --> [price = range1 [-∞ - 1985.714]] (confidence: 0.954)
[mrp = range1 [-∞ - 4628.571], codAvailable = TRUE, categories = GENTS --> [price = range1 [-∞ - 1985.714]] (confidence: 0.954)
[mrp = range1 [-∞ - 4628.571], codAvailable = TRUE, cashBack = range1 [-∞ - 4.429] --> [price = range1 [-∞ - 1985.714]] (confidence: 0.957)
[mrp = range1 [-∞ - 4628.571], codAvailable = TRUE, inStock = TRUE --> [price = range1 [-∞ - 1985.714]] (confidence: 0.961)
[price = range1 [-∞ - 1985.714], size = M, cashBack = range1 [-∞ - 4.429] --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.986)
[price = range1 [-∞ - 1985.714], cashBack = range1 [-∞ - 4.429] --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.988)
[price = range1 [-∞ - 1985.714], size = M, categories = GENTS --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.989)
[price = range1 [-∞ - 1985.714], size = M --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.990)
[price = range1 [-∞ - 1985.714], categories = GENTS --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.991)
[price = range1 [-∞ - 1985.714] --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.991)
[price = range1 [-∞ - 1985.714], deliveryTime = range2 --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.992)
[price = range1 [-∞ - 1985.714], inStock = TRUE --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.997)
[price = range1 [-∞ - 1985.714], codAvailable = TRUE, size = M --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.997)
[price = range1 [-∞ - 1985.714], codAvailable = TRUE --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.997)
[price = range1 [-∞ - 1985.714], codAvailable = TRUE, cashBack = range1 [-∞ - 4.429] --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.998)
[price = range1 [-∞ - 1985.714], codAvailable = TRUE, categories = GENTS --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.998)
[price = range1 [-∞ - 1985.714], codAvailable = TRUE, inStock = TRUE --> [mrp = range1 [-∞ - 4628.571]] (confidence: 0.998)
```


2.3 APRIORI ALGORITHM

The name of the algorithm is based on the fact that the algorithm uses *prior knowledge* of frequent itemset properties. Apriori employs an iterative approach known as a level-wise search, where k -itemsets are used to explore $(k+1)$ itemsets. First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support.

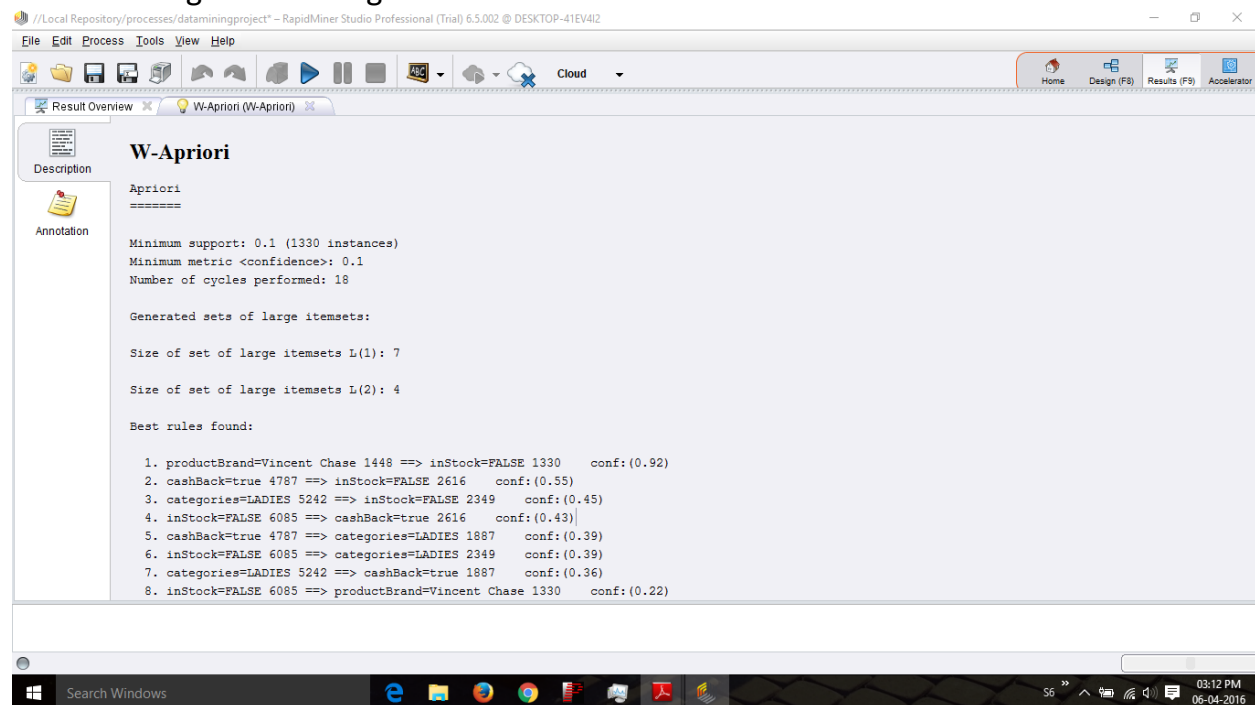
Apriori property: All nonempty subsets of a frequent itemset must also be frequent.

Mainly two steps are present in the apriori property 1) Join Step 2) Prune Step

Apriori operator was not included in the packages of the rapidminer. So additional package for apriori was imported from weka. Then transaction data set's attributes were converted from numeric to nominal and weka –apriori operator was applied .

There is no need of special create association rules as w-apriori itself would generate the association rules. The apriori conditions were confidence :0.1 and min-support :0.5 .

The following rules were generated.



W-Apriori

Apriori
=====

Minimum support: 0.1 (1330 instances)
Minimum metric <confidence>: 0.1
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 7

Size of set of large itemsets L(2): 4

Best rules found:

1. productBrand=Vincent Chase 1448 ==> inStock=FALSE 1330 conf:(0.92)
2. cashBack=true 4787 ==> inStock=FALSE 2616 conf:(0.55)
3. categories=LADIES 5242 ==> inStock=FALSE 2349 conf:(0.45)
4. inStock=FALSE 6085 ==> cashBack=true 2616 conf:(0.43)
5. cashBack=true 4787 ==> categories=LADIES 1887 conf:(0.39)
6. inStock=FALSE 6085 ==> categories=LADIES 2349 conf:(0.39)
7. categories=LADIES 5242 ==> cashBack=true 1887 conf:(0.36)
8. inStock=FALSE 6085 ==> productBrand=Vincent Chase 1330 conf:(0.22)

3. CLUSTERING

Cluster analysis or simply **clustering** is the process of partitioning a set of data objects (or observations) into subsets. Each subset is a **cluster**, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters. The set of clusters resulting from a cluster analysis can be referred to as a **clustering**

The simplest and most fundamental version of cluster analysis is partitioning, which organizes the objects of a set into several exclusive groups or clusters.

3.1 K-MEANS: A Centroid-Based Technique

A centroid-based partitioning technique uses the centroid of a cluster, C , to represent that cluster. The quality of cluster C_i can be measured by the **withincluster**

variation, which is the sum of squared error between all objects in C_i and the centroid c_i , defined as k-means clustering operator in rapidminer is used to perform clustering on the attributes of the data set . Since all of the attributes are non-numerical , the possible attributes are chosen and k-means clustering was performed setting the value of the k as 7 (No. of Clusters). The following clusters are obtained as shown in the screenshot below.

Result Overview Cluster Model (Clustering)

Attribute	cluster_0	cluster_1	cluster_2	cluster_3	cluster_4	cluster_5	cluster_6	cluster_7
deliveryTime	0	0	0	0	0	0	0	0
deliveryTime	0.508	0.535	0.545	0.531	1	0.502	0	0.545
deliveryTime	0.357	0.340	0.353	0.352	0	0.346	0.782	0.319
deliveryTime	0.134	0.125	0.102	0.117	0	0.152	0.218	0.136
deliveryTime	0	0	0	0	0	0	0	0
mnp = range	0.978	0.975	0.975	0	0.985	0.984	0.983	0.990
mnp = range	0.022	0.024	0.002	0.956	0.004	0.002	0.008	0.003
mnp = range	0.001	0.001	0.015	0.010	0.001	0.003	0	0.004
mnp = range	0	0.007	0.006	0.006	0.004	0.001	0.006	0
mnp = range	0	0.001	0	0.002	0	0.001	0	0.003
mnp = range	0	0	0	0.019	0.001	0	0.001	0
mnp = range	0	0	0.000	0.007	0.004	0	0.002	0
price = rangi	0.936	0.935	0.928	0.005	0.933	0.952	0.934	0.949
price = rangi	0.058	0.062	0.048	0.237	0.055	0.043	0.057	0.042
price = rangi	0.005	0.002	0.020	0.523	0.009	0.001	0.006	0.003
price = rangi	0.001	0	0.003	0.228	0.001	0.003	0.002	0.004
price = rangi	0	0.001	0.000	0.005	0.001	0	0.001	0
price = rangi	0	0	0	0	0	0	0	0
price = rangi	0	0.001	0	0.002	0.001	0.001	0	0.003
discount = r2	0.081	0.045	0.011	0.185	0.042	0.021	0.023	0.031
discount = r2	0.020	0.021	0.055	0.272	0.016	0.031	0.021	0.033
discount = r2	0.147	0.133	0.187	0.233	0.181	0.105	0.201	0.113
discount = r2	0.231	0.223	0.271	0.134	0.169	0.390	0.173	0.386
discount = r2	0.230	0.248	0.302	0.041	0.177	0.244	0.207	0.233
discount = r2	0.243	0.261	0.150	0.103	0.343	0.163	0.324	0.155
discount = r2	0.080	0.060	0.000	0.000	0.000	0.000	0.000	0.000