# A Hybrid Recommendation Engine for Food Joints

## Prototype : APOGEE 2015

## Category : Economic Modelling and Finance

By:
Shubham Gupta (2012A7PS086P)
Kapil Singhal (2013A7PS038P)
Prangav Singhal (2014A8PS332P)
Rusheen Rathore (2012A1PS387P)
Shrey Gupta (2012A1PS412P)

Birla Institute of Technology and Science
Pilani

# Objective

To create a hybrid recommendation engine which is able to recommend products to customer that complements the items already purchased by them in a restaurant or a food joint

"A lot of times, people don't **know** what they want until you **show** it to them."

- Steve Jobs

# What is Recommendation Engine

It is an information filtering system that seek to predict the 'rating' or 'preference' that user would give to an item and therefore is able to **suggest items** to the customers

- **Collaborative filtering** methods are based on collecting and analysing a large amount of information on users' behaviours, activities or preferences and predicting what users will like based on their similarity to other users.

- **Content-based filtering** methods are based on a description of the item and a profile of the user's preference.

# HYBRID APPROCH

- **Hybrid Recommendation Systems** is the one combining collaborative filtering and content-based filtering approaches.

# ADVANTAGES

- Increased sales
- Greater margins
- Customer Satisfaction
- Menu Management
- Consumer Awareness

# Timeline

- Data Collection and Pre-Processing
- Designing and developing the recommendation engine
- Integrating the engine with a user application
- Testing the output and taking user feedback

# Data Sets

- We have a computer generated data of a large food joint with about 3.25 lakhs entries collected over the last two years. It has data that is categorised under the following variables :-
  1. Customer ID
  2. Product Category
  3. Product Subcategory
  4. Size
  5. Mode of Order
  6. Net Sales
  7. Quantity
  8. Gender
  9. Zip Code
  10. Age Group

# Timeline

Data Collection and Pre-Processing

- Designing and developing the recommendation engine
- Integrating the engine with a user application
- Testing the output and taking user feedback

# Collaborative Approach

**Customer Ratings**

- CustomerID
- Items
- Ratings

# Sample Data (User Ratings)

| RATINGS | | | | |
|---|---|---|---|---|
| | **Item 1** | **Item 2** | **Item 3** | **Item 4** |
| **User 1** | 2 | 2 | 1 | 4 |
| **User 2** | 0 | 5 | 4 | 0 |
| **User 3** | 0 | 1 | 2 | 2 |
| **User 4** | 1 | 0 | 3 | 4 |
| **User 5** | 1 | 4 | 5 | 3 |
| **User 6** | 1 | 0 | 1 | 3 |
| **User 7** | 1 | 3 | 2 | 0 |
| **User 8** | 5 | 2 | 5 | 2 |
| **User 9** | 0 | 0 | 5 | 5 |
| **User 10** | 5 | 1 | 5 | 4 |

# Recommendation Algo. (Python Code)

```
import numpy
import csv
"""
@INPUT:
    R     : a matrix to be factorized, dimension N x M
    P     : an initial matrix of dimension N x K
    Q     : an initial matrix of dimension M x K
    K     : the number of latent features
    steps : the maximum number of steps to perform the optimisation
    alpha : the learning rate
    beta  : the regularization parameter
@OUTPUT:
    the final matrices P and Q
"""
```

```
def matrix_factorization(R, P, Q, K, steps=5000, alpha=0.0002, beta=0.02):
    Q = Q.T
    for step in xrange(steps):
        for i in xrange(len(R)):
            for j in xrange(len(R[i])):
                if R[i][j] > 0:
                    eij = R[i][j] - numpy.dot(P[i,:],Q[:,j])
                    for k in xrange(K):
                        P[i][k] = P[i][k] + alpha * (2 * eij * Q[k][j] - beta * P[i][k])
                        Q[k][j] = Q[k][j] + alpha * (2 * eij * P[i][k] - beta * Q[k][j])
        eR = numpy.dot(P,Q)
        e = 0
        for i in xrange(len(R)):
            for j in xrange(len(R[i])):
                if R[i][j] > 0:
                    e = e + pow(R[i][j] - numpy.dot(P[i,:],Q[:,j]), 2)
                    for k in xrange(K):
                        e = e + (beta/2) * ( pow(P[i][k],2) + pow(Q[k][j],2) )
        if e < 0.001:
            break
    return P, Q.T
```

```
if __name__ == "__main__":
    reader=csv.reader(open("/home/kapil/rating1.csv","rb"),delimiter=',')
    x=list(reader)
    R=numpy.array(x).astype('int')
    R = numpy.array(R)
    N = len(R)
    M = len(R[0])
    K = 2
    P = numpy.random.rand(N,K)
    Q = numpy.random.rand(M,K)

    nP, nQ = matrix_factorization(R, P, Q, K)
    nR=numpy.dot(nP,nQ.T)
    numpy.savetxt("foo.csv", nR, delimiter=",")
```

# Predicted Ratings

| RATINGS | | | | |
|---|---|---|---|---|
| | **Item 1** | **Item 2** | **Item 3** | **Item 4** |
| **User 1** | 1.15 | 2.52 | 2.23 | 2.93 |
| **User 2** | 1.87 | 4.95 | 4.03 | 4.63 |
| **User 3** | 1.61 | 1.25 | 2.05 | 1.74 |
| **User 4** | 1.18 | 3.61 | 2.78 | 4.06 |
| **User 5** | 2.05 | 3.47 | 3.49 | 4.15 |
| **User 6** | 0.50 | 2.48 | 1.63 | 2.70 |
| **User 7** | 0.85 | 2.92 | 2.15 | 3.26 |
| **User 8** | 4.80 | 1.30 | 4.95 | 2.69 |
| **User 9** | 3.49 | 3.88 | 4.99 | 4.98 |
| **User 10** | 4.92 | 1.73 | 4.27 | 3.17 |

# Content Based Approach

1. **Customers Purchase History**
   - CustomerID
   - Products
   - No. of Times Purchased
   - Quantity
2. **Customer Combos**
   - CustomerID
   - All Products purchased together
3. **Based on Demographics**
   - CustomerID
   - ZipCode
   - AgeGroups
   - Gender
4. **Product Details**
   - Products
   - NetSales

# Timeline

Data Collection and Pre-Processing

Designing and developing the recommendation engine

- Integrating the engine with a user application
- Testing the output and taking user feedback

# Scope of the Project

- Suggesting Items to the Customers
- Fixing better prices for items
- Scrapping unpopular merchandise
- Rewarding Customers with special offers and discounts


Birla Institute of Technology and Science
Pilani

innovate achieve lead