Task (Rating prediction)

In [1]:
```
import numpy
import urllib.request
import scipy.optimize
import random
from collections import defaultdict
import nltk
import string
import os
from nltk.stem.porter import *
from sklearn import linear_model
import matplotlib.pyplot as plt
```

In [2]:
```
def parseData(fname):
    for l in urllib.request.urlopen(fname):
        yield eval(l)
```

In [3]:
```
print("Reading data...")
data = list(parseData("file:train.json"))
print("done")
```

```
Reading data...
done
```

Problem 5

In [4]:
```
train_data = data[:100000]
valid_data = data[100000:]
```

In [5]:
```
allRatings_train = []
allRatings_valid = []
reviewer_item = defaultdict(list)
item_reviewer = defaultdict(list)
pair_rating = defaultdict(list)
i=0
for l in train_data:
    reviewer,item = l['reviewerID'],l['itemID']
    allRatings_train.append(l['rating'])
    reviewer_item[reviewer].append(item)
    item_reviewer[item].append(reviewer)
    pair_rating[reviewer + item].append(l['rating'])
for l in valid_data:
    allRatings_valid.append(l['rating'])

Average = sum(allRatings_train)*1.0/len(allRatings_train)
print ("Alpha: ", Average )
```

```
Alpha:  4.232
```

In [6]:
```
MSE = 0
for x in allRatings_valid:
    MSE = MSE + (Average-x) **2
MSE = MSE / len(allRatings_valid)
print ("MSE on validation set is: ", MSE)
```

```
MSE on validation set is:  1.222481119999121
```

In [ ]:

Problem 6

In [20]:
```
lamda = 1
alpha = 0
beta_reviewer = defaultdict(int)
beta_item = defaultdict(int)

i=0
while i < 500:
    i += 1

    for reviewer in reviewer_item.keys():
        beta_reviewer[reviewer]=sum((pair_rating[reviewer + x][0]-Average -
    for item in item_reviewer.keys():
        beta_item[item]=sum((pair_rating[x + item][0]-Average-beta_reviewer

for reviewer in reviewer_item.keys():
    for item in reviewer_item[reviewer]:
        alpha += ((pair_rating[reviewer+item][0]-beta_item[item]-beta_revie
print ("alpha", alpha)

MSE=0
for l in valid_data:
    reviewer,item = l['reviewerID'],l['itemID']
    rate_predict=beta_reviewer[reviewer]+beta_item[item]+alpha
    MSE = MSE + (rate_predict - l['rating']) ** 2
MSE=MSE/100000
print ("MSE:", MSE)
```

```
alpha 4.231400766370532
MSE: 1.281143227020166
```

In [21]:
```
lamda = 1
alpha = 0


beta_reviewer = defaultdict(int)
beta_item = defaultdict(int)
for reviewer in reviewer_item.keys():
    for item in reviewer_item[reviewer]:
        beta_reviewer[reviewer] = sum((pair_rating[reviewer + x][0]-Average
for item in item_reviewer.keys():
    for reviewer in item_reviewer[item]:
        beta_item[item]=sum((pair_rating[x + item][0]-Average-beta_reviewer

for reviewer in reviewer_item.keys():
    for item in reviewer_item[reviewer]:
        alpha += ((pair_rating[reviewer+item][0]-beta_item[item]-beta_revie
print ("alpha", alpha)


MSE=0
for l in valid_data:
    reviewer,item = l['reviewerID'],l['itemID']
    rate_predict=beta_reviewer[reviewer]+beta_item[item]+alpha
    MSE = MSE + (rate_predict - l['rating']) ** 2
MSE=MSE/100000
print ("MSE:", MSE)
```

```
alpha 4.231707482679271
MSE: 1.2605827693662364
```

In [ ]:

Problem 7

In [8]:
```
target_max = max(beta_reviewer.values())
target_min = min(beta_reviewer.values())
for x in beta_reviewer.keys():
    if beta_reviewer[x] == target_max:
        print("reviewerID with max_beta: ", x)
    if beta_reviewer[x] == target_min:
        print("reviewerID with min_beta: ", x)
```

```
reviewerID with max_beta:  U495776285
reviewerID with min_beta:  U204516481
```

In [9]:
```
target_max = max(beta_item.values())
target_min = min(beta_item.values())
for x in beta_item.keys():
    if beta_item[x] == target_max:
        print("itemID with max_beta: ", x)
    if beta_item[x] == target_min:
        print("itemID with min_beta: ", x)
```

```
itemID with min_beta:  I511389419
itemID with max_beta:  I809804570
```

In [ ]:

Problem 8

In [11]:
```
def train(lamda, Average, reviewer_item, item_reviewer, pair_rating):
    alpha = 0
    beta_reviewer = defaultdict(int)
    beta_item = defaultdict(int)

    i=0
    while i < 500:
        i += 1

        for reviewer in reviewer_item.keys():
            beta_reviewer[reviewer]=sum((pair_rating[reviewer + x][0]-Avera
                                        for x in reviewer_item[reviewer])/(
        for item in item_reviewer.keys():
            beta_item[item]=sum((pair_rating[x + item][0]-Average-beta_revi
                                or x in item_reviewer[item])/(lamda+len(ite

    for reviewer in reviewer_item.keys():
        for item in reviewer_item[reviewer]:
            alpha += ((pair_rating[reviewer+item][0]-beta_item[item]-beta_r
    print ("alpha", alpha)

    MSE=0
    for l in valid_data:
        reviewer,item = l['reviewerID'],l['itemID']
        rate_predict=beta_reviewer[reviewer]+beta_item[item]+alpha
        MSE = MSE + (rate_predict - l['rating']) ** 2
    MSE=MSE/100000
    print("lamda is: ", lamda)
    print("MSE is: ", MSE)
    return alpha, beta_reviewer, beta_item
```

In [12]:
```
lamda_test=[1, 4, 5, 6, 7, 8, 10, 100]
for lamda in lamda_test:
    train(lamda, Average, reviewer_item, item_reviewer,pair_rating)
```

```
alpha 4.231388674091933
lamda is:  1
MSE is:  1.28113923201379
alpha 4.230918478095691
lamda is:  4
MSE is:  1.1454069139152079
alpha 4.230876700210596
lamda is:  5
MSE is:  1.1399110617720556
alpha 4.230854964744218
lamda is:  6
MSE is:  1.1379377877593821
alpha 4.23084569302904
lamda is:  7
MSE is:  1.1377804626335801
alpha 4.23084440310799
lamda is:  8
MSE is:  1.1386031650822064
alpha 4.230855668883374
lamda is:  10
MSE is:  1.1416124042480875
alpha 4.231466168529679
lamda is:  100
MSE is:  1.1998254049208708
```

In [18]:
```
alpha, beta_reviewer, beta_item = train(6.7, Average, reviewer_item, item_r
predictions = open("predictions_Rating.csv", 'w')
for l in open("pairs_Rating.txt"):
    if l.startswith("reviewerID"):
        predictions.write(l)
        continue
    reviewer, item = l.strip().split('-')
    rating_pred = alpha + beta_reviewer[reviewer] + beta_item[item]
    predictions.write(reviewer + '-' + item + "," + str(rating_pred) + '\n'
predictions.close()
```

```
alpha 4.2308474742046585
lamda is:  6.7
MSE is:  1.1376969305466105
```

Kaggle Username: Macchiato

In [ ]: