Assignment 1

Pedro Sousa Meireles
A15677282

November 18, 2018

1 Task 1: Purchase Prediction

1.1 Information gathered

I started getting information about the items and users. For each item in the dataset I collected its average rating, category, number of purchases and average price, for those that have a price assigned.

For each user I collected every item they bought, the average rating they gave to all items, the amount of items they bought from each category and the average price they paid.

For each category I collected its number of purchases, average rating, average prices and number of different items.

I also computed the average price, average number of purchases and average rating in the whole dataset and sorted the items from the most popular to the less popular.

1.2 Assumptions

Before building a model, I made some reasonable assumptions on the problem:

- Users will most probably buy items from the same category of items they already bought.
- Users will buy items with average rating bigger than the average rating of items in the category/dataset.
- Users will most likely buy popular items in the category they already bought.
- Users will buy items that are cheaper than the average of the category/dataset.

1.3 Building Models

1.3.1 Comparison models

My first approach was to build a model similar to the one in Homework 3. I started with the model we used in Question 3 and started to add more comparisons.

I tried to predict the purchase as true if:

1. The item is cheaper than the average in its category.
2. The item is cheaper than the average in the dataset.

3. The item rating is bigger than the average in its category
4. The item rating is bigger than the average in the dataset
5. The user had already bought some item in the item category.
6. The item category was the user's favorite category.
7. The item was in the 50%/25%/10%/5%/2%/1%/0.5% more popular items in the dataset.
8. The item was in the 50%/25%/10%/5%/2%/1%/0.5% more popular items in its category.

I tried to use all of the conditions above alone and tried lots of them mixed together. I also tried to use weights when comparing ratings and prices, but none of this worked. The best score I got was ∼0.6.

1.3.2 Linear Regression

After giving up on comparison models, I tried to use linear regression to predict the purchases. I tried to use some combinations of the features obtained in section 1.1 in my feature vector, but results were almost always 0.5.

1.3.3 Jaccard, Cosine and Pearson similarities

I tried to implement Jaccard similarity to predict a purchase as 1 if the item was similar to a item the user had already bought, but it took too long to run, so I gave up. As Cosine and Pearson similarities would take even longer, I didn't try them.

1.3.4 SVMs

Since linear regression didn't work, I decided to try using Support Vector Machines. I basically used the same feature vectors that I used in linear regression.

Doing it I noticed that the less features I used, the higher was my score. Even though I always got ∼0.5 accuracy in both train and validation sets, when I submitted my predictions to Kaggle, my results were a bit higher.

So, by decreasing the number of features used, I managed to increase my score from ∼0.41 to 0.628, which is the highest score I could achieve in this task. For this, I used a SVM with a feature vector composed only by the number of times the item was purchased.

2 Task 1: Purchase Prediction

2.1 Information gathered

For this task I used reviews' texts to predict categories. I computed the frequency of each word in the whole dataset. I also did the same for each category independently. I computed the frequencies both considering and not considering stopwords.

Besides, I sorted the words from most frequent to less frequent.

2.2 Building models

2.2.1 KMeans

I tried to use KMeans to cluster my data in 5 groups, each one representing a category. As feature vectors I used booleans representing if each of the 500 most common words in the dataset/category were in the review text. The accuracy was below 20%.

2.2.2 Linear Regression and SVMs

I used the same approach with Linear Regression and SVMs. As they're binary predictors, they would be wrong for some categories. Using the same feature vectors as before and different C parameters (0.01, 0.1, 1, 10 and 100) for the SVMs I got scores around 0.8, with a maximum of 0.82514.

I also tried to add the data collected in section 1.1 to the feature vector, but it didn't improve the score.

I used only 10% of the dataset to train the SVMs as it would take too long to use 100%.

I tried to make a better model that wouldn't only predict two categories, so I used 5 SVMs, one to predict for each category, and chose the SVM with the higher $decision\_function$, as suggested in Homework 3 Question 8. Surprisingly, the obtained score was worst than 0.8.