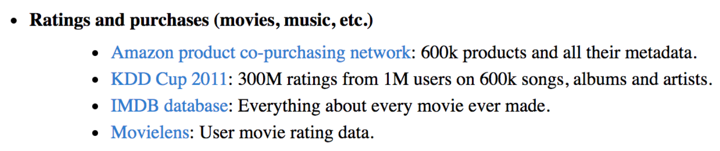
AI训练营-数据挖掘-第一周

Day 1

1. Learn Python
   a. Python Basics
   b. Install/Learn Python Libraries: Numpy/SciPy/Matplotlib/IPython Notebook/Pandas
   c. Reading:
      i. http://cs231n.github.io/python-numpy-tutorial/
      ii. https://github.com/kuleshov/cs228-material/blob/master/tutorials/python/cs228-python-tutorial.ipynb
      iii. https://github.com/sul-cidr/python_workshops/blob/master/data_manipulation.ipynb
2. Review of Probability Theory
   a. Random variable; Cumulative distribution function (CDF); Probability density function (PDF); Expectation; Variance; Joint and marginal distribution; Conditional distribution; Bayes rule.
   b. Reading: (can skip the reading if you know all the concept above; if not, only read the revelant part to learn the concepts.) http://cs229.stanford.edu/section/cs229-prob.pdf
3. Review of Linear Algebra
   a. Vector; Matrix; Matrix Inverse; Trace; Rank; Eigenvalues; Lease Square
   b. Reading: (can skip the reading if you know all the concept above; if not, only read the revelant part to learn the concepts.) http://cs229.stanford.edu/section/cs229-linalg.pdf
4. Play with Data
   a. Datasets: Go to https://web.stanford.edu/class/cs341/data.html



- **Ratings and purchases (movies, music, etc.)**
  - Amazon product co-purchasing network: 600k products and all their metadata.
  - KDD Cup 2011: 300M ratings from 1M users on 600k songs, albums and artists.
  - IMDB database: Everything about every movie ever made.
  - Movielens: User movie rating data.

   b. **Pick one out of four datasets listed above for your Week1 project!**
   c. Data profiling (compute any statistics interested you most). Some inspiring examples: median ratings, rating distribution, user distribution, average song per albums/artistist, histogram of users/ratings/songs, etc. **Try to understand the data!**

Day 2

1. Recommendation Basics
   a. Content-based recommendations (Content)
   b. Collaborative Filtering (CF)
   c. Reading:
      i. https://web.stanford.edu/class/cs345a/slides/03-recomsys.pdf
      ii. http://blog.ethanrosenthal.com/2015/11/02/intro-to-collaborative-filtering/
2. Recommendation Evaluations
   a. Top K recall
   b. Root mean square error
   c. Mean absolute error
   d. Reading:
      i. https://stackoverflow.com/questions/33697625/recall-recall-ratek-and-precision-in-top-k-recommendation
      ii. https://en.wikipedia.org/wiki/Root-mean-square_deviation

        iii. https://en.wikipedia.org/wiki/Mean_absolute_error

3. Implement Your First Recommendation Algorithm. Use the data picked at Day 1 and implement the following recommendation algorithms.
    a. Always recommend the most popular item
    b. Content-based recommendations Algorithm: Recommend based on content, i.e., TFIDF
    c. Collaborative Filtering Algorithm: User-to-User Collaborative Filtering
    d. Collaborative Filtering Algorithm: Item-to-Item Collaborative Filtering
4. Test above recommendations with multiple metrics. In terms of testing, **you need to hide some unknown rating during the training and only test on those unknown ratings.**
5. Pick several examples and look at the actually recommendation results from above algorithms and see whether they make sense. **Human evaluation!**

Day 3

1. Recap and re-think what is done on Day 2 and make sure you know how Content and CF work. Keep in mind that both Content and CF rely on finding similiar "things".
2. KNN: k nearest neighbor search.
    a. Reading: http://scikit-learn.org/stable/modules/neighbors.html
    b. Read above link to fully understand KNN. (KD-tree and Ball tree parts can be skipped).
    c. Play with examples in above doc and think about the limitation of KNN.
    d. Actually, the Content-based recommendations Algorithm we implemented in 3(b) in Day 2 is using KNN.
3. In 2., we do exact KNN search, which means for each item, we need to search the entire item space to figure out those similiar items. Think about when you have millions of movies, how can you use KNN to do recommendation. There are both algorithm and infrastructure solutions to solve this "big data" problem.

    我们做了精确的KNN搜索，这意味着对于每个项目，我们需要搜索整个项目空间来找出那些相似的项目。 想想你有几百万部电影，怎么可以用KNN做推荐。 有算法和基础设施解决方案来解决这个“大数据”问题。

    a. Algorithm solution: Approximated KNN.
        i. Reading: https://en.wikipedia.org/wiki/Nearest_neighbor_search#Approximation_methods
        ii. Facebook has already open sourced their approximate KNN library (**FAISS**). https://github.com/facebookresearch/faiss Download and run the package on your real data set. Repeat the experiment you did in Day 2 3(b) with FAISS.
        iii. Benchmark the results. Compare the running time of recommending items with/without FAISS.
    b. Infrastructure solution (**OPTIONAL**): Sharding
        i. Reading: https://en.wikipedia.org/wiki/Shard_(database_architecture)

Day 4

1. Matrix Factorization (MF).
    a. What is MF? Reading: http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python/
    b. How to use MF? Reading: http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html
    c. Think about what is the matrix in the dataset you pick on Day 1 and use sklearn function to factroize your matrix!
2. Test MF performance. On Day 2, you have learned how to evaluate a recommendation algorithm. Use the evaluation metrics you learned on Day 2 and benchmark the performance of MF. Compare the results with Content and CF.

Day 5

1. Summarize what you learned so far.

    a. Content based recommendation by using exact KNN search.
    b. Content based recommendation by using approximate KNN search.
    c. User to User CF
    d. Item to Item CF
    e. Matrix Factorization
2. Refactor your code so that all five methods above have the same interface and by giving different data sets, results and relevant evaluation metrics can be computed easily.
3. Think about how to make improvement? Hint: when using above methods, what information you use? what are those information you forget to put into the algorithm? Can you make the recommendation results better?
4. Brainstorm and come up with your own recommendation algorithm! Google can be very helpful! You don't need to invent a super cool algorithm by yourself. Find a paper that excites you most and go ahead and implement it.
    a. How to find a paper. Google "X accept papers in Y" where X = {KDD, WSDM, AAAI, IJCAI, SIGIR, WWW} and Y = {2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017}.

Day 6

1. Data profiling on TAL data.
2. Play with your refactored, clean recommendation algorithms on TAL data.
3. Implement your own recommendation algorithm or published algorithm and test it on both the data you picked on Day 1 and the TAL data.
4. Benchmark both the recommendation accuracy and running time.
5. Prepare the report to summarize what you learned in this week
    a. Data statistics (feature distributions, etc.)
    b. Recommendation performance (recall, RMSE, MAE, etc)
    c. Running time analysis
    d. **Any interesting stuff you find**

Prepare the presentation slides