Homework 9
Applications of Machine Learning

CMU 10-601: Machine Learning (Spring 2017)
https://piazza.com/cmu/spring2017/10601
OUT: April 24, 2017
DUE: May 3, 2017 11:59 PM
Authors: Rui Sun, Prakruthi Prabhakar, Simon Du, Sarah Schultz

1 Introduction

In this assignment you will implement techniques you have learned in this course on a real dataset. Previously, in Homework 5, you read a paper related to one of three topics: Natural Language Processing, Computer Vision, or Recommender Systems. For this assignment, we have provided you with the dataset on each of the three topics, which you were asked to describe in Homework 5. These datasets can be found on Autolab. You are given a labeled training dataset and an unlabeled test set. Your task is to implement and train a model using the training data and use it to classify the test data. You will submit your learned labels on the test data through Autolab.

2 Specifications

- You may use any code that you've previously implemented in this course as well as any libraries previously used in those implementations and any of the libraries listed at https://piazza.com/class/ixs4v2xr1cz10d?cid=2226.

  Note: Any other library or code which you require from external sources have to be approved by the course staff. You may NOT use an external library for Recommender Systems.
- You may work in groups of no more than two (2). If you are working with a partner, you are expected to work together collaboratively.
- You are encouraged, but not required, to work on the same topic that you chose in Homework 5. If you decide to work on a different topic, you must read at least one of the recommended papers on that topic from Homework 5.
- The purpose of this assignment is to give you the experience of implementing an end-to-end machine learning model. The idea is to implement fundamental techniques learnt in the course, do thorough experimentation and analyze the results. You are free to experiment with any model for the task. The purpose of this assignment is NOT that you design a state-of-the-art algorithm, as this would require a large amount of effort. However, if you are interested in doing so. See section 5.5

3 Guidelines

The below points are some questions to help analyze the model and experiment with it. We have also provided further information about how it would be applicable in a real world setting.

- What does the provided dataset contain? Does it have missing data? Does it require pre-processing? *Analysis of data is the first step in any machine learning problem. Understanding the provided data,* *performing data cleaning and imputation and generating simple statistics will help in identifying the information value in the dataset.*

- What is the problem? Can the problem be reduced to regression or classification? What are the models which could be trained for solving the problem?
  The next step is to analyze what models are applicable for solving the problem. Understanding the model complexity and availability of training data will help in choosing the right model for solving the problem.
- What are the features currently present in the dataset? Does the data require feature engineering? Are there interesting features which could be generated from the available data to better model the problem?
  Feature engineering is an essential component of real-world machine learning problems. Sometimes, domain knowledge will help in identifying which features could be more relevant to the task than other. Sometimes, several features are generated and experimentally determined which contribute the most to the model.
- What are the model parameters and hyper-parameters?
  Tuning the parameters and hyper-parameters will help you to better fit your model
- How to split the data into train and validation dataset?
  Performing n-fold cross validation will help analyze model performance (if the model is overfitting / underfitting). This will also help identify if the model is able to learn well from the available dataset or if there is a need for a complex / simpler model.
- What are the evaluation metrics? Can the parameters be tuned to optimize performance?
  Evaluation is the last crucial step. Defining standard evaluation criterion is very important for justifying the model performance. For instance, if there is class imbalance in the data, will accuracy alone suffice? Evaluation also helps in parameter tuning. In most of the previous assignments, we had provided parameter values such as learning rate to experiment with. In this assignment, you are required to evaluate your model and optimize performance by choosing best configuration of model parameters. You are required to analyze how different model parameters affect the time complexity as well as model performance.

4 Submission

Autolab: Three assignments(NLP,CV,RS) for homework9 have been set up on Autolab. You will submit a tar file containing your code as well as the predictions given by your machine learning system on the test data through the corresponding assignment on Autolab. You will put your code in a folder named code and result in a .csv file named as results.csv. Your results file should contain just the learned labels for the test data with each label **on a new line**. The handin tar file structure looks like:

- \code
- results.csv

Use `tar cvf hw9.tar results.csv code` command to build your handin file. You will only allow 20 submissions in total.

Gradescope: You will submit one write-up not exceeding 2 pages, containing short descriptions of the following:

- Problem Statement
- Data Pre-processing and Feature Engineering
- Model Implemented (including choice of parameters and how they were chosen)
- Analysis of Results (including why the metric used for analysis of this task was appropriate)

- If you are in a group of two, please briefly state how you shared the workload.
- Any other collaboration with students outside of your group (as detailed in the course policy).
- Time spent

Groups of 2: If you are working in a group of 2, please use Autolab and Gradescope's group functionality to submit as a group. In Autolab click "Group Options" under "Options" and in Gradescope see https://youtu.be/aLesBkeL5zk?t=34s for a demonstration of how to submit as a group.

5 Grading

The model performance will be evaluated for a total of 75 points and the write-up is worth 25 points. We have provided a leaderboard on Autolab, where you can compare your model accuracy with others (who are working on the same task as yours) along with some of our baselines. Please refer to the "finalScore" column on Autolab to find your score on this portion of the assignment. The "Total Score" column on Autolab should be ignored. Your learned labels will be compared to some benchmarks we've provided on a metric appropriate to each task and graded as follows (The percentages are accounted for a total of 75 points):

5.1 NLP

For the NLP chunking task, the metric will be the F1 score. F1 score is calculated as a combination of precision (the ratio of how many predictions are correct) and recall (the ratio of how many of the correct labels were predicted). This score will be a number from 0 to 100, with higher numbers being better. Perl code for calculating the F1 score for this dataset can be found at:

http://www.cnts.ua.ac.be/conll2000/chunking/conlleval.txt

As this code requires one file containing both the correct and guessed labels as input, we provide Python code that will concatenate your correctly labeled data and guesses and call this script. This can be found at https://piazza.com/class/ixs4v2xr1cz10d?cid=2225

The baselines for this task are:

- **Random Guessing**: This baseline was obtained by simply randomly guessing one of the possible chunks for each data point. If your method does at least as well as this benchmark, but not as well as the next, you will get 10% on the Auotlab portion of the assignment. (F1 $\geq$ 2.18)
- **Simple Counting Algorithm**: This was the baseline for the CoNLL 2000 task. This "was obtained by selecting the chunk tag which was most frequently associated with the current part-of-speech tag." If your classifier does as well as this benchmark, but not as well as the next, you will get 30% on the Autolab portion of the assignment. (F1 $\geq$ 76.06)
- **Linear SVM with Relatively Few Features, Using SGD, L2 regularized, No Tuning**: This baseline performs a little better than the Simple Counting Algorithm. If your classifier does as well as this baseline, but not as well as the next, you will get 50% on the Autolab portion of the assignment. (F1 $\geq$ 77.78)
- **Linear SVM with More Features, SGD, L2, Careful Tuning**: This baseline performs much better than the previous one. If your classifier does as well as this baseline, but not as well as the next, you will get 80% on the Autolab portion of the assignment. (F1 $\geq$ 89)
- **Neural Network with Many Features**: This is the highest baseline included for NLP. To get 100% on this assignment your classifier should do at least as well as this baseline. This baseline was obtained using the same features as the previous one and a small neural network. (F1 $\geq$ 92)

5.2 CV

For this task, you will be working with a sample of CIFAR-10, called CIFAR-3, which we have created to simplify the task. The provided dataset contains three classes, with training and test examples being 32x32 (x3 for RGB) color images.(More details of the data format can be found on Autolab, dataformat_discription) The metric for evaluation would be accuracy, calculated as the percentage of correct labels among all predicted labels. The baselines for this task are:

- **Majority Class**: This baseline was obtained by selecting the output class to be equal to the class label occuring majority of the times in the training data. To get 10% on this assignment, you need to be more accurate than this baseline. (Accuracy $\geq$ 33.33%)
- **Random Guessing**: This baseline was obtained by randomly guessing one of the three output classes. To get 20% on this assignment, you need to be more accurate than random guessing. (Accuracy $\geq$ 34.63%)
- **Simple Classifier**: This baseline was obtained by training a simple SVM classifier on the training dataset. To get 50% on this assignment, you need to be more accurate than this baseline. (Accuracy $\geq$ 75.8%)
- **Simple Classifier with parameter tuning**: This baseline was obtained by training a simple SVM classifier on the training dataset along with some hyperparameter tuning. To get 60% on this assignment, you need to be more accurate than this baseline. (Accuracy $\geq$ 79.33%)
- **Deep Learning with simple ConvNet**: This baseline was obtained by training a simple convolutional neural network. To get 85% on this assignment, you need to be more accurate than this baseline. (Accuracy $\geq$ 93.3%)
- **Deep Learning with Deeper Convnet and Data Augmentation**: This baseline was obtained by training a deeper convolutional neural network with some tricks like data augmentation and ZCA whitening. To get 100% on this assignment, you need to be more accurate than this baseline. (Accuracy $\geq$ 95.8%)

5.3 Recommendation System

Let Ω_{test} denote the indices of testing ratings. For the Recommendation System task, the metric will be RMSE:

$$\sqrt{\frac{\sum_{(i,j)\in\Omega_{test}} (\hat{r}_{i,j} - r_{i,j})^2}{|\Omega_{test}|}}$$

where $r_{i,j}$ is the true rating and $\hat{r}_{i,j}$ is your prediction.

The training data and testing indices are included in the handout files. Please check Section 6 for details.

- **Rating Mean Prediction**: to get 30% on this assignment, your predictions need to be more accurate than using the mean of all ratings (RMSE $\leq$ 1.1196).
- **Movie / User Mean Prediction**: to get 50% on this assignment, your predictions need to be more accurate than using the mean of each movie (RMSE $\leq$ 0.9800).
- **Matrix Factorization with Simple Tuning**: to get 80% on this assignment, your predictions need to be more accurate than using the matrix factorization model with some simple tuning on number of latent factor (RMSE $\leq$ 0.9000).
- **Matrix Factorization with Carefully Tuned Parameters**: to get 100% on this assignment, your predictions need to be more accurate than using the matrix factorization model with some careful tuning on number of latent factor, regularization and adding some constraints like non-negativity (RMSE $\leq$ 0.8800)

5.4 Chocolate

This assignment focuses on how to effectively utilize machine learning to solve challenging real-world problems. The goal is not to achieve state-of-the-art performance or outperform your peers, but to learn about the nuances and tricks that are required to get machine learning to work in the wild. That said, we recognize that some of you may go above and beyond our expectations. The student(s) with the highest scores on each challenge will be awarded with chocolates.

6 Helpful Hints and Suggestions

6.1 NLP

Pre-processing: watch out for commas in the data if you are using csv! You might want to convert them to another character that is not present in the data already.

It might be helpful to use a bag-of-words style sparse feature matrix. For example, if your training data consisted of only the sentence 'the cat sat on the mat' and you wanted to take the current word into account, you might represent the word 'cat' by the vector [0, 1, 0, 0, 0, 0] where the 1 indicates that the word 'cat' is present and the 0s indicate the other words are not. You can also represent the part-of-speech tag and context similarly. For example, if 'cat' was tagged 'noun,' your feature vector might look like [0, 1, 0, 0, 0, 0, 0, 1, 0], where the first 1 indicates the word 'cat' and the second indicated the tag 'noun' while the 0s indicate words or tags not present for this sample.

Another approach is to convert words to vector representations using GloVe. The two approaches can also be combined such that a vector representation of the word and a sparse vector representation are both used in the features.

6.2 Computer Vision

- ZCA_whitening on each of the three channels of the image.
- Data Augmentation by rotation, cropped windows of images, etc will help.

6.3 Recommendation System

The data files have the following formats:

```
movies.dat: movie_id :: genres
users.dat: user_id :: sex :: age group : occupation
training_rating.dat: user_id :: movie_id :: rating
testing.dat: user_id movie_id
```

See `http://files.grouplens.org/datasets/movielens/ml-1m-README.txt` for detailed description. This data is not the same data as on the MovieLens website, so please use our provided data.

For submission, please use the same format as the results.csv we provide to you in the handout files. Note that your predictions can be non-integer.

Suggestions:

- The rating data has some missing data, i.e. a (user,movie,rating) tuple missing user id or movie id. You need to do some pre-processing to ignore them. We recommend first try to use movie / user mean prediction to see if you have processed data correctly.
- We recommend to use matrix factorization algorithms but you are welcome to use other methods.

- For matrix factorization, you can try classical approach taught in class or adding non-negative constraints to user and movie latent factors.
- We recommend to use a validation dataset to tune the hyper-parameters like number of latent factors, magnitude of regularizations, etc.