Assignment 1

Due on November 01, 2019 (23:59:59)

Click here to accept your Assignment 1

Instructions. There are two parts in this assignment. The first part involves a series of theory questions and the second part involves coding. The goal of this problem set is to make you understand and familiarize with K-Nearest Neighbor Classifier and Kernel Regression Algorithm.

Part I: Theory Questions

k-Nearest Neighbor Classification

1. Assume that you have a large training dataset. Specify a disadvantage of the k-Nearest Neighbor method when using it during testing. State also your reason about your answer.
2. Create a 1-Dimensional classification dataset in which the 1-Nearest Neighbors method always gives a leave-one out cross validation error value of 1 (In other words, the method can't guess correct class for a specific point in the dataset). State also a proper explanation about your reasoning.
3. Assume that you have the following training set of positive (+), negative (-) instances and a single test instance (o) in the figure below (Figure 1). Assume also that the Euclidean metric is used for measuring the distance between instances. Finally consider that every nearest neighbor instance affects the final vote equally.

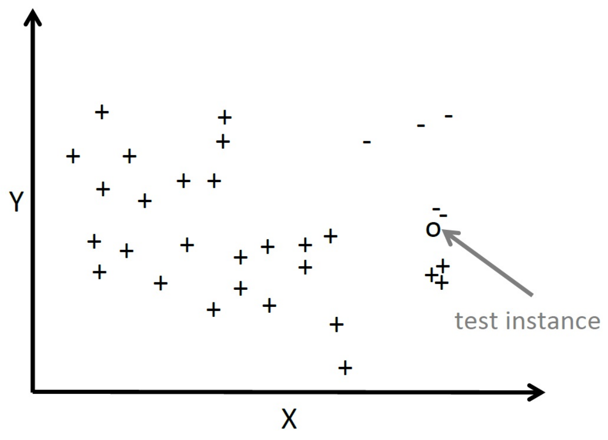

Figure 1: Example KNN Case

- What is the class appointed to the test instance for K=1? State also reason behind your answer.
- What is the class appointed to the test instance for K=3? State also reason behind your answer.
- What is the class appointed to the test instance for K=5? State also reason behind your answer.

4. Fill the blanks with T (True) or F (False) for the statements below:
   - If all instances of the data have the same scale then k-Nearest Neighbor's performance increases drastically. (_)
   - While k-Nearest Neighbor performs well with a small number of input variables, it's performance decreases when the number of inputs becomes large. (_)
   - k-Nearest Neighbor makes no assumption about the functional form of the problem it handles. (_)

Linear Regression

1. Assume that you have five students have registered to a class and the class have a midterm and the final exam. You have obtained a set of their marks on two exams, which is in the table below:

Student	**Midterm Exam**	**Midterm exam (Squared)**	**Final Exam**
$x^{(1)}$	87	7569	94
$x^{(2)}$	70	4900	72
$x^{(3)}$	92	8464	85
$x^{(4)}$	67	4489	76
$x^{(5)}$	45	2025	51

   You plan to a model which form's is $f_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ for fitting the data above. The x_1 shows midterm exam score while x_2 shows square of the midterm score. Besides you plan to use feature scaling (using divide operation by the "max-min", or range, of a feature) and mean normalization. What is the normalized value of the feature $x_2^{(4)}$?

2. Considering the figure below (Figure 2), which of the offsets used in linear regressions least square line fit? Assume that horizontal axis represents independent variable and vertical axis represents dependent variable. State your answer with your proper explanation.

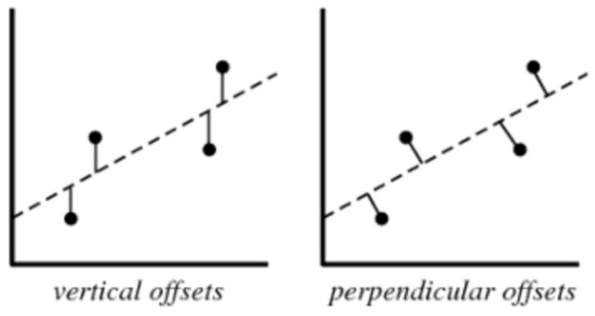

Figure 2: Different Alternative Offsets

3. Considering the table below (Table 1), consisting of four training examples:

x	**y**
1	0.5
2	1
4	2
0	0

Table 1

   Assume that you are trying to fit the data above to the linear regression model $f_\theta(x) = \theta_0 + \theta_1 x_1$. Find the θ_0 and θ_1 values by using closed form solution ($\theta = (X^T X)^{-1} X^T y$). Also state dimension values of X, y and θ matrices. Finally show your calculations step by step.

4. State a valid reason for feature scaling and explain why it is a valid reason with respect to your reasoning.

PART II: Movie Recommendation System

In this part, you will implement a nearest neighbor algorithm to recommend movies to viewers best suited to their tastes and traits. Similarly, your algorithm will also be able to recommend movies to a user based on user-item ratings.

Specifically, you are going to implement a KNN algorithm to find sets of similar users based on common movie ratings, and make predictions using the average rating of top-k nearest neighbors. You will also extend your implementation as weighted k-NN algorithm.

You are provided with a movie ratings dataset of 100.000 ratings of 9.000 movies by 600 users. The ratings are on a scale between 0 to 5. Your algorithm will try to make predictions using the average rating of top-k nearest neighbors.

Collaborative Filtering

Collaborative filtering based systems collect and analyze users' behavioral information in the form of their feedback, ratings, preferences and activities. Based on this information, these systems then exploit similarities amongst several users/ items to predict missing ratings and hence make suitable recommendations. They can discover and learn features on its own without the need of explicit features to profile items or users. Types of collaborative filtering are usually grouped into two as:

- User-based collaborative filtering
- Item-based collaborative filtering

A dataset is provided for your training phase in the Piazza. The validation set will be provided later and announced from Piazza group. Since validation set will be provided later, you should use a subset of the training set to test the performance of your model. In other words, you should split your training dataset into two : the training set which will be used to learn model and the test set which will be used to measure the success of your model. You can use k-fold cross-validation method which is explained in the class. **You should also state your details about each model's accuracy, model accuracies with respect to different k parameters you chosen and how you choose the final accuracy in your report.**

Dataset

MovieLens is a movie rating dataset compiled by GroupLens Resource Team [1]. As mentioned before, you have given the small version of the dataset which contains 100.000 ratings of 9.000 movies by 600 users. The ratings are on a scale from 0 to 5.

- The dataset consists of four tables: ratings, movies, links and tags info.
- The ratings data set provides a list of ratings that users have given to books. It includes 100.837 records and 4 fields: "userId", "movieId", "rating", "timestamp".
- The movies dataset provides movie details. It includes 9.743 records and 3 fields: "movieId", "title" and "genres".
- The links dataset provides id details. It includes 9.743 records and 3 fields: "movieId", "imdbId" and "tmdbId".
- The tags dataset provides tag details. It includes 3.684 records and 4 fields: "userId", "movieId", "tag" and "timestamp".

Bonus (Kaggle Competition)

As a bonus task, you can try different feature sets/combinations or additional link informations for movies from the shared files, in order to improve your results.

Similarity Functions

- There is no limitation about which similarity function you use in your implementation. You can try using the following similarity functions :
  - **Cosine-based Similarity** In this case, two items are thought of as two vectors in m dimensional user-space. The similarity between them is measured by computing cosine angle between two vectors. Similarity between vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, denoted by

    $$sim_{cosine}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\left\|\mathbf{x}_i\right\|_2 \left\|\mathbf{x}_j\right\|_2}$$
  - **Correlation-based Similarity** Similarity between two items is measured by computing correlation $corr_{i,j}$. Denoting the set of users who both rate i and j as U, the correlation similarity is given by

    $$sim_{correlation}(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_i})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_j})^2}}$$

    Where $R_{u,i}$ represents the rating of user u on item i and $\overline{R_i}$ is the average rating of the i-th item.
  - **Adjusted Cosine Similarity** Computing similarity by using basic cosine measure in item-based case has one obvious drawback, and it is the difference in rating scale between different users. We can subtract this kind of bias, so the similarity using this scheme is given by

    $$sim_{adj.cosine}(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_u})^2}}$$

    Where again, U represents the set of users who both rate i and j, $R_{u,i}$ represents the rating of user u on item i and $\overline{R_u}$ is the average of the u-th user's ratings.

Steps to follow

1. Combine movie data with rating data. You can extract more than one data.
2. Use user-based collaborative filtering.
3. Calculate similarities (cosine-based, correlation-based, adjusted cosine, or your own choice) between rating vectors to find the nearest neighbors.
4. For a given test sample, you will try to recommend movies.
5. Finally you will measure your recommender algorithm performance for each setting you have used:

Mean Absolute Error (MAE) $= \frac{1}{n}\sum_{i=1}^{n} |d_i| - |\hat{d_i}|$
d_i is the actual rating
$\hat{d_i}$ is the predicted rating
n is the amount of ratings

Submit

You are required to submit all your code (*all your code should be written in Jupyter notebook* long with a report in ipynb format (should be prepared using Jupyter notebook). The codes you will submit should be well commented. Your report should be self-contained and should contain a brief overview of the problem and the details of your implemented solution. You can include pseudocode or figures to highlight or clarify certain aspects of your solution. Finally, prepare a ZIP file named **name-surname-pset1.zip** containing

- report.ipynb (PDF file containing your report)
- code/ (directory containing all your codes as Python file .py)

The ZIP file will be submitted via Github Classroom. Click here to accept your Assignment 1

NOTE: To enter the competition, you have to register kaggle in Class with your department email account. The webpage of the competition will be announced later. Top 5 assignment will earn extra points (10p).

Grading

- Code (60p): k-NN: 20p, Weighted k-NN: 40p
- Report(40p): Theory part: 20p, Analysis of the results for prediction: 20p.

> **Note**: Preparing a good report is important as well as the correctness of your solutions! You should explain your choices and their effects to the results. You can create a table to report your results.

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

[1] Harper, F. Maxwell, and Joseph A. Konstan. ”The movielens datasets: History and context.” Acm transactions on interactive intelligent systems (tiis) 5.4 (2016): 19.