Assignment 3: kNN and SVM

UVA CS 6316 :
Machine Learning (Fall 2019)

Out: Oct. 10 2019
Due: Oct. 22, Sun midnight 11:59pm, @ Collab

a *The assignment should be submitted in the PDF format through Collob. If you prefer hand-writing QA parts of answers, please convert them (e.g., by scanning or using an app) into PDF form.*

b *For questions and clarifications, please post on piazza.*

c *Policy on collaboration:*

Homework should be done individually: each student must hand in their own answers. It is acceptable, however, for students to collaborate in figuring out answers and helping each other solve the problems. We will be assuming that, with the honor code, you will be taking the responsibility to make sure you personally understand the solution to any work arising from such collaboration.

d *Policy on late homework: Homework is worth full credit at the midnight on the due date. Each student has three extension days to be used at his or her own discretion throughout the entire course. Your grade would be discounted by 15% per day when you use these 3 late days. You could use the 3 days in whatever combination you like. For example, all 3 days on 1 assignment (for a maximum grade of 55%) or 1 each day over 3 assignments (for a maximum grade of 85% on each). After you've used all 3 days, you cannot get credit for anything turned in late.*

e *Policy on grading:*

***1**: 50 points in total. 20 points for code submission (and able to run). 30 points for displaying results in the report.*
***2**: 40 points in total. 20 points for code submission (and able to run). 10 point for result submission. 10 points for the result table in your report.*
***3**: 10 points in total. 5 points for answering each question.*
***4**: 10 extra points if you wins the competition! (Get Top 20 accuracy in leaderboard)*
The overall grade will be divided by 10 and inserted into the grade book. Therefore, you can earn 11 out of 10.

1 KNN and Model Selection (K) (programming)

- Task 1: To implement a very simple classifier, k-nearest neighbors (KNN), from scratch.
- Task 2: To implement a 4-folds Cross Validation strategy for selecting the best K for KNN classification. Feel free to reuse the cross-validation code in the previous HWs.

We will use Euclidean distance as the measurement of distance in KNN. We will test your implementation on the *"Movie_Review_Data.txt"*

- Please use the "knn.py" template to guide your work. Please follow the instructions and the function names/descriptions in the template. Feel free to cross-check your implementation against sci-kit's KNN. Other requirements or recommendations are the same as HW1 and HW2.

- 2.1 Implement the CV method:

$$x\_train, y\_train, x\_test, y\_test = fold(x, y, i, nfolds)$$

So that when i is iterated from 0 to $nfolds - 1$, the folds of train and test data will be seperated.

ATT: Feel free to try other folds of CV, kfold=3-fold or kfold=10-fold.

- 2.2 Fill the blank part inside function $findBestK()$ following the guidance in the template.
- 2.3 Implement the classify method:

$$y\_predict = classify(x\_train, y\_train, x\_test, K)$$

Where K is the number of data points that KNN takes into consideration when labeling an unlabeled data point. Note here the training data is part of the testing (classify) algorithm. In KNN, we label a new data point based on the k points that are the closest to it in our train dataset. In this function, for each testing point, please find its K nearest neighbor in the training set (those having the smallest Euclidian distance to the testing point). Then we label the testing point according to the majority voting rule.

- 2.4 Implement the calc_accuracy method:

$$acc = calc\_accuracy(y\_predict, y)$$

Where *y_predict* is the list of 0 or 1 predictions obtained from the classify method and y is the true label for the testing points (the last column of the test data).

(Hint1: If your accuracy is below 50%. Please consider how the order of the samples are dictated by the class.)

- 2.5 Run the code with $K \in \{3, 5, 7, 9, 11, 13\}$. Report the accuracy and the best K. Discuss why some values work better than others.
- 2.6 Implement function *barplot*. A bar graph is recommended to show the change of accuracy with K. Use K as x-axis and accuracy as y-axis. Put the bar plot in your report.
- Att: we will not consider the computational speed in grading your kNN code.

2 Support Vector Machines with Scikit-Learn and preprocessing

- (1) For this assignment, you will implement a SVM classifier. Given a proper set of attributes, your program will be able to determine whether an individual makes more than 50,000 USD/year.
- (2) you will use scikit-learn's C-Support Vector Classifier: sklearn.svm.SVC. Install the latest stable version of scikit-learn following directions available at
http://scikit-learn.org/stable/install.html
- (3) Two dataset files are provided, labeled sample set "salary.labeled.csv" and unlabeled sample set "salary.2Predict.csv". Make sure to download them from Collab! The unlabeled sample set "salary.2Predict.csv" is a text file in the same format as the labeled dataset "salary.labeled.csv", **except that its last column includes a fake field for class labels**. You are required to generate/ predict labels for samples in "salary.2Predict.csv" (including about $10k$ samples).

- (4) We will evaluate your output 'predictions' - an array of strings (">50K" or "<=50K") according to the true labels of these test samples (ATT: you don't have these labels !!!). This simulates a Kaggle-competition in which test labels are always held out and only team-ranking will be released after all teams have submitted their predictions. When grading this assignment, we will rank all students' predictions. So please try to submit the best performing model that you can! **Best groups in the tournament will receive bonus credit.**
- (5) You need to include the CV classification accuracy results in your pdf report by performing 3-fold cross validation (CV) on the labeled set "salary.labeled.csv" (including about $38k$ samples) using at least three different SVM kernels you pick. Please provide details about the kernels you have tried and their performance (e.g. 3CV classification accuracy) including results on both train and test folds into the writing. For instance, you can summarize the results into a table with each row containing kernel choice, kernel parameter, CV train accuracy and CV test accuracy.

Detail Steps:

1. Load and Preprocess your data:
   - Implement load_data function in svm_template.py
   - Use techniques mentioned in class!
   - Be creative. A good data preprocessing can help you win the tournament.
   - To start, you can try regularization on continuous feature and one-hot encoding on categorical data.
2. Write a wrapper code for cross-validation, and iterate through different hyperparemters (just like we did in HW1/2).
   - hyperparameters you can choose include SVM kernels (basic linear kernel / polynomial kernel / RBF kernel), C value, ...
   - Call sklearn.svm.SVC with your parameter to get a classification!
3. Select the best hyperparameter, train a model and save the output, following the steps in svm_template.py. **Remember not to shuffle your prediction outputs!**

Submission Instructions: You are required to submit the following :
(The starting code, 'svm_template.py', has been provided in Collab.)

1. A python program svm.py:

   It should be able to train and select a model using a set of hyperparameters on the training data, these hyperparameters should be hard coded.

   Next, we should be able to use the learned/ trained_model to classify samples in an unlabeled test set using the following function:

   ```
   predictions = predict(``salary.2Predict.csv", trained_model)
   ```

2. A file "predictions.txt" generated by:

   ```
   output_results(predictions)
   ```

   Please do not archive the file or change the file name for the automated grading.

3. In your PDF submission,
   - A table reporting classification accuracy (score) averaged over the test folds, along with details of the kernels, best performing hyperparameter C for each case etc.
   - Some writing about how you preprocess the data, and how you choose your SVM hyperparameters

Classes: >50K, <=50K.
Attributes:
age: continuous. workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked. fnlwgt: continuous. education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool. education-num: continuous. marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse. occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces. relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried. race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black. sex: Female, Male. capital-gain: continuous. capital-loss: continuous. hours-per-week: continuous. native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

Table 1: About the data in SVM coding.

3 Sample QA Questions:

Question 1. Support Vector Machine

Soft-margin linear SVM can be formulated as the following constrained quadratic optimization problem:

$$argmin_{\{w,b\}} \frac{1}{2} w^T w + C \sum_{i=1}^{m} \epsilon_i$$

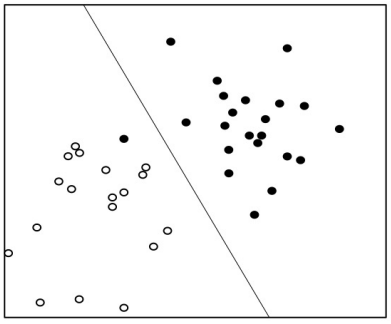
such that

$$\begin{aligned} y_i(w^T x_i + b) &\geq 1 - \epsilon_i \\ \epsilon_i &\geq 0 \qquad\qquad \forall i \end{aligned}$$

where C is the regularization parameter, which balances the margin (smaller $w^T w$) and the penalty of mis-classification (smaller $\sum_{i=1}^{m} \epsilon_i$).

(a) (**True/False**) Number of support vectors do not depend on the selected value of C. Please provide a one-sentence justification.

In the following figure, $n_{C=0}$ is the number of support vectors for C getting close to 0 ($\lim_{C \to 0}$) and $n_{C=\infty}$ is the number of support vectors for C gets close to ∞.



C=1

(b) Select the correct option:( )

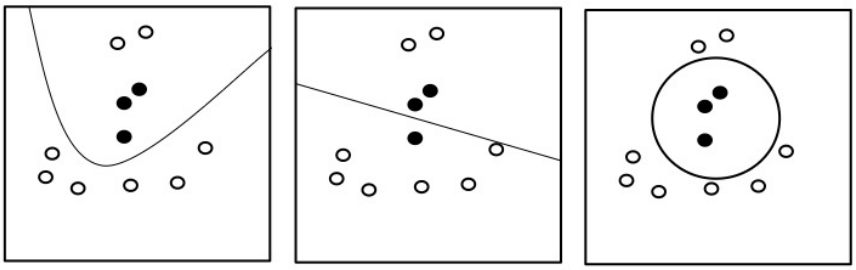
(1) $n_{C=0} > n_{C=\infty}$

(2) $n_{C=0} < n_{C=\infty}$

(3) $n_{C=0} = n_{C=\infty}$

(4) Not enough information provided

(c) Match the following list of kernels used for SVM classification with the following figures:



_____ _____ _____

(1) Linear Kernel : $K(x, x') = x^T x'$

(2) Polynomial Kernel (order = 2) : $K(x, x') = (1 + x^T x')^2$

(3) Radial Basis Kernel : $K(x, x') = exp(-\frac{1}{2}||x - x'||^2)$