Assignment 3: Adapt to Change

AI 539, Winter 2023

Due: Feb. 22, 2023, at 11:59 p.m.

What happens when the real world changes over time? This assignment gives you the opportunity to explore a method for **adapting machine learning models to domain shift**.

Learning objectives:

- You will apply Black Box Shift Correction (BBSC) to update classifier predictions when the underlying class frequencies have changed.
- You will explain adaptations for different classifiers in different settings.
- You will describe your results in a **Findings** document.

Background: We are working with a data set that consists of car accident records along with weather and location information. The data we use is a subset from the larger data set which was compiled by Sobhan Moosavi (https://www.kaggle.com/sobhanmoosavi/us-accidents).

What to do:

1. Download the data set files from Canvas:
   A. **train-TX.csv:** random subset of Texas car accidents (Jan. 1 – March 31, 2020)
   B. **val-TX.csv:** (different) random subset of Texas car accidents (Jan. 1 – Mar. 31, 2020)
   C. **test1-TX.csv:** Texas car accidents (April 1 – October 31, 2020)
   D. **test2-FL.csv**: Florida car accidents (Jan. 1 – March 31, 2020)
   E. **test3-FL.csv**: Florida car accidents (April 1 – October 31, 2020)

2. Review the documentation at the Kaggle link above so you understand the features.
   A. The class label is "Severity" (1, 2, 3, or 4).
   B. Use these 21 features: Distance(mi), Temperature(F), Wind_Chill(F), Humidity(%), Pressure(in), Visibility(mi), Wind_Speed(mph), Precipitation(in), Amenity, Bump, Crossing, Give_Way, Junction, No_Exit, Railway, Roundabout, Station, Stop, Traffic_Calming, Traffic_Signal, Turning_Loop

3. In Python, write a program called **`shift_adapt.py`** to train **four classifiers** (RandomForest, GaussianProcess, 3-NearestNeighbor, and 9-NearestNeighbor), plus the two baselines below, on the accidents in the training data set (**train-TX.csv).** Baselines to use:
   A. Predict based on training set majority class (e.g., scikit-learn's `DummyClassifier(strategy='most_frequent', random_state=0)`
   B. Predict based on training set class distribution (e.g., scikit-learn's `DummyClassifier(strategy='stratified', random_state=0)`

Describe each baseline and classifier in your **Findings** document.

4. Generate predictions for each baseline and classifier on the validation data set and each test set. Control the random state so your results are reproducible.

5. Expand your program to perform **label shift adaptation** on the classifier predictions using the Black Box Shift Correction (BBSC) implemention in **label_shift_adaptation.py** (from Canvas). This method was proposed by Lipton et al. (2018). The paper is available at http://proceedings.mlr.press/v80/lipton18a.html. The implementation also incorporates some ideas from Alexandari et al. (2020), http://proceedings.mlr.press/v119/alexandari20a.html.
   A. Use `analyze_val_data()` to compute adaptation weights.
   B. Use `update_probs()` to adapt classifier predictions. You will need to pass in the classifier's predicted probabilities as well as class predictions. Most classifiers in scikit-learn provide probabilities with the `predict_proba()` function.
   C. Treat this file as a library. Do not modify it or move the code into your script. *Example usage: See comments (doctest) for each function.*

6. Report results in your **Findings** document. (Your program should output this table.)

Accuracy	**Val-TX**	**Test1-TX**	**Test2-FL**	**Test3-FL**
Baseline: most_frequent				
Baseline: stratified				
RandomForest		[a1,a2]		
GaussianProcess				
3-NearestNeighbor				
9-NearestNeighbor				

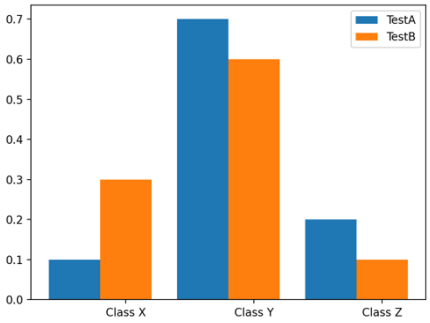
   - Each of the shaded cells should include two numbers:
     [a1: test accuracy of original predictions, a2: test accuracy after using BBSC]
   - Report only two digits after the decimal, e.g. 83.76% accuracy.

7. Show the adaptation weights that were calculated for each classifier and each test set in your **Findings** document. (Your program should output this table.)

Adaptation weights	**Test1-TX**	**Test2-FL**	**Test3-FL**
RandomForest	[w1,w2,w3,w4]		
GaussianProcess			
3-NearestNeighbor			
9-NearestNeighbor			

   - Each of the shaded cells should include four numbers (one per class).
   - Report only two digits after the decimal, e.g. weight of 0.76.

8. Put a figure in your **Findings** document showing the **true class label** distribution of all **four data sets** (validation plus three test sets) on the same plot (e.g., grouped bar plot). Rather than showing class counts, normalize each data set's counts by the size of the data set so that the distributions are comparable (each value should range from 0 to 1). Hypothetical example (3 classes, 2 data sets):



9. Answer these questions in your **Findings** document:
   A. In your own words, explain "label shift" and give an example.
   B. In your own words, explain how BBSC works.
   C. In your experiments, in which cases did BBSC adaptation improve test set accuracy? How was better accuracy achieved?
   D. Did BBSC ever yield lower accuracy for a classifier after adaptation? If so, why do you think this happened? (Look at the classifier's validation set confusion matrix, its distribution of predictions on the test set where worse results were seen after adaptation, and the BBSC adaptation weights that were computed.)
   E. What do you learn by looking at the distribution of **true class labels** in the three test sets? How did car accident severity change in each data set (vs. validation)?
   F. What is one reason that would cause all BBSC adaptation weights to be 0.0? Use your understanding of BBSC and inspection of the methods in **label_shift_adaptation.py** to answer this question.
   G. **Reflection**: What did you learn from completing this assignment? How long did it take (in hours)? What was the hardest part? What might you use in the future?

What to submit (see rubric on Canvas for point breakdown):

- **Good programming style is required (readable code with good use of whitespace and variable names; header with author, date, assignment; internal comments to help the reader understand the program).**
- **Attribution must given when code is re-used from another source.**
- A .zip file containing **`shift_adapt.py`** and README file.
  - You do not need to submit the data files.
- Your **Findings** document (pdf, doc, docx, rtf, or txt), not in the .zip.

Extra credit (optional; include your answers in your Findings document):

10. Rank the features by their importance/relevance for the random forest classifier. (In scikit-learn, you can inspect `clf.feature_importances_` after training a RandomForestClassifier in `clf`). Show the ranked list. Are the most important features what you would expect? Are there any surprises?

11. (a) Which of the three test sets do you think exhibit covariate shift?
    (b) What is one method for detecting covariate shift?
    (c) What is one method for addressing covariate shift (to correct predictions and improve accuracy)?

Do you have questions? Please submit your questions here:
https://discord.com/channels/1061031671010447380/1061031672184844400