ECE 5464 SP24 – Prof. Jones – HW 3

Due Tuesday, March 19, 2024 – 11:59 PM via Canvas

Part 1 (20 points):

In the Datasets folder in the Files section of Canvas, you will find an Excel spreadsheet named "K8.xlsx" (it's a dataset relating "mutant p53 transcriptional activity - active vs inactive - based on data extracted from biophysical simulations" to a large number of anonymous numeric inputs). The file has a single sheet named "K8"; there are 16772 rows and 5409 columns. All columns (except the last) are numeric; the last column is binary (either "active" or "inactive"). Note that the columns do NOT have names in the spreadsheet!

You are to write a Python program to explore the use of nearest-neighbor classification on this dataset. To do this you should do the following:

1. Read in the data set;
2. Perform all of the usual steps of preparing a dataset for modeling;
3. Split the data set into training and test partitions, using a 70-30 split;
4. Train and test a set of KNeighborsClassifier models, using all possible combinations of the following parameters:
    a. both uniform and distance weighting
    b. $nnbrs \in [1, 3, 5, 9, 11, 15, 21]$
    c. dataset (before the train-test split) unbalanced (the original) and balanced to 50-50 in terms of the target class.[1]
5. Write out to the console the training and test model score for each of the possible models, along with all of the parameters (N, balance, weighting type).
6. Create a table presenting this data in the most useful manner.
7. Identify the combination of parameters that gives the best performance on the test set, and the combination of parameters that gives the largest and smallest difference between test set and training set performance.

Part 2 (15 points):

For this part, you are to compare the mean square errors for multivariate linear regression models on the Batting Salaries dataset and on yearly partitions of it. In the Datasets folder in the Files section of Canvas, you will find an Excel spreadsheet named "BattingSalariesData.xlsx" (it's some performance and pay data for professional baseball players). You are to implement a Python program to do the following:

1. Load the dataset, and perform all of the data preprocessing as usual.
2. Use each of the numeric features as predictors – except for Salary, of course.
3. Use one-hot encoding to derive features from the league ID and team ID fields.
4. Implement a linear regression model to predict the continuous variable 'Salary'.
    a. Divide the dataset into training and test sets, using the value 22222 as the random seed value;

[1] A classification dataset is balanced if the number of samples is the same for each possible value of the target variable. For example, a dataset with a binary target [0,1] is balanced if there are as many 0s as 1s.

b. Train the regressor on the entire training set;
c. Measure the accuracy on the test set, as both the r^2 value and the mean square error.

5. For each year present in the dataset:
    a. Write out the year and the size of the dataset for just that year;
    b. Divide the dataset <u>for just that year</u> into training and test sets, using the value 22222 as the random seed value;
    c. Implement a linear regression model to predict the continuous variable 'Salary';
    d. Train the regressor on that year's training set;
    e. Measure the accuracy on that year's test set, as both the r^2 value and the mean square error.
6. Plot the MSE versus year, to determine which year's salary is most "predictable"

HINTS:

- For part 1, it may take some time to load and preprocess the whole dataset (mine took about 45 minutes). I STRONGLY advise you to create a small sample of the dataset (say, all columns but only 100 rows) to use for developer testing of your code.
- When balancing your data, use undersampling (keep all of the minority class and sample without replacement from the majority class to achieve balance).
- Explicitly set the random seed value in your program, so you can run the code multiple times and expect exactly the same result (from functions such as train_test_split that have some random behavior).

Your submission should include:

- A single WORD file including:
    - all of your Python code for part 1 pasted in as plain text (no dark-mode or screen shots)
    - the output of your program for part 1 (as plain text)
    - all of the discussion for part 1 that I request above
    - your part 1 results table
    - all of your Python code for part 2 pasted in as plain text (no dark-mode or screen shots)
    - the output of your program for part 2 (as plain text)
    - the plot of MSE versus year (you can use Tableau or something else).
- Your .py files as attachments (if you use Jupyter, do NOT attach the ipynb file, you must download your code as one or more .py files).