Programming Assignment: Naive Bayes Classifier

Background

For this assignment, we will be utilizing data samples drawn from the Zoo Animal Classification dataset from the UCI Machine Learning Repository. You can find further details here: https://archive.ics.uci.edu/ml/datasets/Zoo.
The dataset consists of 101 animals from a zoo. There are 16 variables describing various traits related to the animals. The 7 Class Labels are 'Mammal', 'Bird', 'Reptile', 'Fish', 'Amphibian', 'Bug', and 'Invertebrate'.

Naive Bayes Classifier

We will quickly review the inner working of a Naive Bayes Classifier.

Let us denote the features as X and the label as y. A Multinomial Naive Bayes classifier makes predictions based on an estimate of the joint probability $P(X, y)$.

For each datapoint, the predicted label y' is determined as $y' = argmax_y P(X, y) = argmax_y P(y)P(X|y)$.

Also, note the underlying independence assumption: $P(X|y) = \Pi_i P(x_i|y)$

Therefore, to make a prediction, we need to utilize $P(y)$ and $P(x_i|y)$

For this assignment, we will be smoothing out probabilities $P(y)$ and $P(X|y)$ via Laplacian correction with a pseudo-count of 0.1.

Therefore,

$$P(y = c) = \frac{\#\ training\ examples\ of\ class\ c\ +\ 0.1}{N + 0.1|C|}$$

Here, N would be the total number of data points and C would be the set of classes.

$$P(x_i = f|y = c) = \frac{\#\ training\ examples\ where\ class = c\ and\ x_i = f\ +\ 0.1}{\#\ of\ training\ examples\ where\ class = c\ +\ 0.1 \times \#\ unique\ values\ for\ attribute\ x_i}$$

Test case format

The very first line indicates the type of test case. `0` indicates a test case where you are supposed to compute and return the set of prior probabilities. `1` indicates a test case where you compute the class labels for the test data points.

The following lines represent a CSV file with the header line being:

```
animal_name,hair,feathers,eggs,milk,airborne,aquatic,predator,toothed,backbone,breathes,venomous,fins,legs,tail,domestic,catsize,class_type
```

Note here that we do **NOT** use `animal_name` as a feature.

`class_type` is the target label.

The lines are split into training and testing data points. For the training data points, the `class_type` field will have an integer value between 1 and 7 (both inclusive). For the testing data points, the `class_type` field will have the value -1.

Finally, these are the set of possible values taken by each attribute

- animal_name ~ string (NOT used for classification, provided for reference only)
- hair ~ {0, 1}
- feathers ~ {0, 1}
- eggs ~ {0, 1}
- milk ~ {0, 1}
- airborne ~ {0, 1}
- aquatic ~ {0, 1}
- predator ~ {0, 1}
- toothed ~ {0, 1}

- backbone ~ {0, 1}
- breathes ~ {0, 1}
- venomous ~ {0, 1}
- fins ~ {0, 1}
- legs ~ {0, 2, 4, 5, 6, 8}
- tail ~ {0, 1}
- domestic ~ {0, 1}
- catsize ~ {0, 1}

Here is a sample input file with the expected output being `[0.2897, 0.1028, 0.1028, 0.1028, 0.0093, 0.1963, 0.1963]`.

```
0
animal_name,hair,feathers,eggs,milk,airborne,aquatic,predator,toothed,backbone,breathes,venomous,fins,legs,tail,domestic,catsize,class_type
aardvark,1,0,0,1,0,0,1,1,1,1,0,0,4,0,0,1,1
worm,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,7
piranha,0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0,4
gnat,0,0,1,0,1,0,0,0,0,1,0,0,6,0,0,0,6
oryx,1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1,1
moth,1,0,1,0,1,0,0,0,0,1,0,0,6,0,0,0,6
skimmer,0,1,1,0,1,1,1,0,1,1,0,0,2,1,0,0,2
crab,0,0,1,0,0,1,1,0,0,0,0,0,4,0,0,0,7
vampire,1,0,0,1,1,0,0,1,1,1,0,0,2,1,0,0,1
slowworm,0,0,1,0,0,0,1,1,1,1,0,0,0,1,0,0,3
bass,0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0,-1
```

And here is another sample input file with the expected output being `[4]`.

```
1
animal_name,hair,feathers,eggs,milk,airborne,aquatic,predator,toothed,backbone,breathes,venomous,fins,legs,tail,domestic,catsize,class_type
aardvark,1,0,0,1,0,0,1,1,1,1,0,0,4,0,0,1,1
worm,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,7
piranha,0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0,4
gnat,0,0,1,0,1,0,0,0,0,1,0,0,6,0,0,0,6
oryx,1,0,0,1,0,0,0,1,1,1,0,0,4,1,0,1,1
moth,1,0,1,0,1,0,0,0,0,1,0,0,6,0,0,0,6
skimmer,0,1,1,0,1,1,1,0,1,1,0,0,2,1,0,0,2
crab,0,0,1,0,0,1,1,0,0,0,0,0,4,0,0,0,7
vampire,1,0,0,1,1,0,0,1,1,1,0,0,2,1,0,0,1
slowworm,0,0,1,0,0,0,1,1,1,1,0,0,0,1,0,0,3
bass,0,0,1,0,0,1,1,1,1,0,0,1,0,1,0,0,-1
```

Template Code and Autograder

You are allowed to use one of **three** programming languages for this assignment: Python (3.10), C++ (14), or Java (JDK version 17). For each language, only standard built-in libraries would be usable. For example, if using Python, you would **NOT** be able to use libraries like NumPy or Scikit-Learn.

Note that the test case format is provided for you to understand the test case files that will be used. For whichever language you use, you would NOT have to read in test cases from STDIN. That will be handled by the grader. You would need to complete the required functions in the appropriate template code file.

Also, do **NOT** include any print statements within the code that you submit as the autograder will judge the printed lines which would lead to a 'Wrong Answer' outcome.

Lastly, you do **NOT** have to round your answers for test case type `0` (computing prior probabilities). The grader will handle that for you. Premature rounding of floating point numbers could lead to inaccurate results.

Similar to past gradescope assignments, you are provided with the template code for each language containing descriptions for the expected input and output for each function that you would have to write. You **ONLY** need to submit the completed template code file: `submission.py` or `submission.cpp` or `Submission.java`. Note the capitalization for the `.java` file.

Descriptions for input variables in the template files:

1. `X_train`: Matrix of shape N times d where N is the number of training data points and d is the number of attributes. Each row represents a training data point.
2. `Y_train`: List of size N holding the true class labels for the training data points in `X_train`
3. `X_test`: Matrix of shape N times d where N is the number of testing data points and d is the number of attributes. Each row represents a testing data point.

- For `X_Train` and `X_test`, the possible values for each attribute have been provided above.

On gradescope, your code will be tested on a list of public test cases, for which, you would be able to view both the input and the expected output. You may use these test cases to debug and verify your implementation.

At the end of the submission deadline, you would be able to see the results produced by your code on a list of hidden test cases. Your final grade will be based on both the public and the hidden test cases.

All the best.