Report on Homework 3

Exploring Classification using Naive Bayes

Course : CS725 - Foundations of Machine Learning

Submitted By:

Ankush Mitra - 23M0755
Shariq Faraz - 23M0779

Parameters

1. Gaussian (Features: X1, X2):

		Classes		
Feature	**Parameter**	**0**	**1**	**2**
X1	**Mean (μ)**	2.02	0.02	8.02
	Variance (σ^2)	9.05	25.16	35.67
X2	**Mean (μ)**	3.91	0.86	-0.02
	Variance (σ^2)	78.43	230.03	4.01

2. Bernoulli (Features: X3, X4):

		Classes		
Feature	**Parameter**	**0**	**1**	**2**
X3	**p**	0.20	0.60	0.91
X4	**p**	0.10	0.80	0.19

3. Laplace (Features: X5, X6):

		Classes		
Feature	**Parameter**	**0**	**1**	**2**
X5	**μ**	0.06	0.38	0.75
	b	78.42	230.03	4.01
X6	**μ**	0.87	0.29	0.21
	b	78.42	230.03	4.01

4. Exponential (Features: X6, X7):

		Classes		
Feature	**Parameter**	**0**	**1**	**2**
X7	λ	1.97	2.96	8.86
X8	λ	3.92	7.92	14.47

5. Multinomial (Features: X9, X10)

		Classes		
Feature	**Parameter**	**0**	**1**	**2**
X9	p_0	0.202	0.098	0.205
	p_1	0.203	0.198	0.299
	p_2	0.204	0.405	0.103
	p_3	0.197	0.158	0.341
	p_4	0.194	0.141	0.051
X10	p_0	0.121	0.101	0.197
	p_1	0.124	0.051	0.048
	p_2	0.128	0.051	0.048
	p_3	0.127	0.010	0.105
	p_4	0.127	0.152	0.155
	p_5	0.127	0.149	0.152
	p_6	0.124	0.200	0.098
	p_7	0.123	0.097	0.194

Results

Data	Performance Metrics			
	Accuracy (%)	F1 score		
		Class 0	Class 1	Class 2
Training	90.14 %	0.881	0.879	0.943
Validation	90.20 %	0.881	0.878	0.946

Approach

The following is a description of the approach taken in the implementation of Naive Bayes classifier:

- The `__init__` method of the class takes an optional parameter `smoothing_alpha`, which is a small positive value that is added to the counts or estimates of probabilities to avoid zero probabilities and improve generalization. The default value is 0.001.

For each class `c` in `self.classes`, we do the following:

- Filters the rows of `X` that belong to class `c` and stores them in `class_data`.
- Calculates the **prior probability** of class `c` using **Laplace smoothing** and stores it in `self.class_priors[c]`.

- The model fits different distributions for each feature given each class, depending on the feature index. For each feature index, the model assumes a certain distribution (Gaussian, Bernoulli, Laplace, Exponential, or Multinomial) and estimates its parameters using maximum likelihood estimation with smoothing. The model then appends a tuple of the distribution name and the parameters to a list that stores the distributions and parameters for each class.

To make a prediction we do the following:

- Initializes a variable `posterior` to store the **logarithm of the posterior probability** of class `c` given `x`. This is done to avoid numerical underflow when multiplying many small probabilities.
- Add the logarithm of the prior probability of class `c` to `posterior`.
- For each feature index `feature_idx` in range from 0 to `self.features`, it does the following:

    Extrats the distribution and parameters of feature index `feature_idx` given class `c` from `self.distributions[c][feature_idx]`.

- For each feature, the model uses a different distribution (Gaussian, Bernoulli, Laplace, Exponential, or Multinomial) and applies the corresponding formula to compute the logarithm of the likelihood. The model then adds the logarithm of the likelihood to the posterior probability of the class.
- It then adds the logarithm of the likelihood to `posterior`.

It then finds the class `predicted_class` that has the **maximum posterior probability** and appends it to `predictions`.