**Programming Assignment02**

|  |
| --- |
| **Submission guide**  1. Write answer following questions in this file  2. Write your code using provided Jupyter notebook file   * Do not use other packages that are not already imported in the script * After completing your code, run script and submit with the printed results for answering questions in this word file |

**1. Naïve Bayes (35pts)**

The goal of this problem is to build naïve Bayes (NB) models to classify whether each message is spam or not.

The data used in this problem provide frequencies of words for each message.

The column “label” is the target variable, indicating whether each message is spam or not (the label “ham” means that it is not a spam message.).

**Part 1: Bernoulli NB**

To build a Bernoulli NB, if a certain word is used in a message, the values of the variables are set to 1; otherwise, they are set to 0.

1-(1). After the conversion, train a Bernoulli NB using training set (alpha=1). Prior probabilities of classes are proportional to ratios of classes in training set. Then, calculate the overall accuracy and accuracy values corresponding to each target class (spam) for the training and validation sets, respectively. (6pts)

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

1-(2) Find the top 10 most probable words in each class according to the model trained in Question 1-(1) and summarize them with the probability of existence of words in the following tables. (7pts)

[Spam]

|  |  |  |
| --- | --- | --- |
| **Order** | **Word** | **Probability** |
| 1 | get | 0.075633 |
| 2 | go | 0.058522 |
| 3 | know | 0.058179 |
| 4 | ltgt | 0.058179 |
| 5 | like | 0.057153 |
| 6 | got | 0.054415 |
| 7 | call | 0.053046 |
| 8 | come | 0.053046 |
| 9 | time | 0.045517 |
| 10 | day | 0.04449 |

[Ham]

|  |  |  |
| --- | --- | --- |
| **Order** | **Word** | **Probability** |
| 1 | call | 0.25 |
| 2 | mobile | 0.132143 |
| 3 | claim | 0.108929 |
| 4 | txt | 0.103571 |
| 5 | reply | 0.092857 |
| 6 | prize | 0.092857 |
| 7 | text | 0.089286 |
| 8 | free | 0.0875 |
| 9 | contact | 0.083929 |
| 10 | 16 | 0.071429 |

1-(3) Find the top 10 words whose probability of existence is high in spam messages, but low in ham messages according to the model trained in Question 1-(1). In addition, find the top 10 words whose probability of existence is high in ham messages, but low in spam messages. (3pts)

|  |  |  |
| --- | --- | --- |
| Order | High in spam, low in ham | High in ham, low in spam |
| 1 | claim | ltgt |
| 2 | prize | lor |
| 3 | 16 | later |
| 4 | 18 | da |
| 5 | 1000 | cant |
| 6 | awarded | come |
| 7 | tone | already |
| 8 | 150 | ask |
| 9 | 2000 | say |
| 10 | guaranteed | didnt |

1-(4) Describe your opinion related to the results of Question 1-(2) and 1-(3). (5pts)

* First of all, words tend to appear more on Ham class rather than Spam class which means there are less spam mail then real mail. With result of 1-(3), we can get to know that in spam mail, words such as ‘claim’, ‘prize’, ‘awarded’ appear frequently.

**Part 2: Multinomial NB**

2-(1) Train a multinomial NB using training set (x\_train, y\_train) (alpha=1). Prior probabilities of classes are proportional to ratios of classes in training set. Then calculate the overall accuracy and accuracy values corresponding to each class for the training and validation sets, respectively. (3pts)

|  |  |  |
| --- | --- | --- |
| **Dataset** | **Class** | **Accuracy** |
| Training | Overall | 0.958309373 |
| Validation | Overall | 0.959770115 |
| Training | ham | 0.993835616 |
| Training | spam | 0.772401434 |
| Validation | ham | 0.989056088 |
| Validation | spam | 0.805755396 |

2-(2) Find the top 10 most probable words in each class according to the model trained in Question 2-(1) and summarize them with the probability of existence of words in the following tables. (3pts)

[Spam]

|  |  |  |
| --- | --- | --- |
| Order | Term | Probability |
| 1 | call | 0.059529 |
| 2 | mobile | 0.030763 |
| 3 | claim | 0.02517 |
| 4 | txt | 0.023971 |
| 5 | prize | 0.022373 |
| 6 | text | 0.022373 |
| 7 | reply | 0.021974 |
| 8 | free | 0.021175 |
| 9 | contact | 0.018777 |
| 10 | week | 0.015981 |

[Ham]

|  |  |  |
| --- | --- | --- |
| Order | Term | Probability |
| 1 | get | 0.031692966 |
| 2 | ltgt | 0.030088259 |
| 3 | go | 0.024872961 |
| 4 | like | 0.023803156 |
| 5 | know | 0.02366943 |
| 6 | got | 0.022064723 |
| 7 | call | 0.022064723 |
| 8 | come | 0.022064723 |
| 9 | time | 0.018721583 |
| 10 | day | 0.018186681 |

2-(3) Find the top 10 words whose probability of existence is high in spam messages, but low in ham messages according to the model trained in Question 2-(1). In addition, find the top 10 words whose probability of existence is high in ham messages, but low in spam messages. (3pts)

|  |  |  |
| --- | --- | --- |
| Order | High in spam, low in ham | High in ham, low in spam |
| 1 | claim | ltgt |
| 2 | prize | lor |
| 3 | 16 | later |
| 4 | 18 | da |
| 5 | 1000 | cant |
| 6 | awarded | come |
| 7 | tone | already |
| 8 | 150 | ask |
| 9 | entry | say |
| 10 | 2000 | amp |

2-(4) Compare the two NB models trained in Questions 1-(1) and 2-(1), considering the results of Questions 1-(1), (2), and (3) and 2-(1), (2), and (3). (5pts)

* Similar words appear frequently in both models, and the results are almost similar for the frequency of occurrence compared to the other class.

**2. Decision tree (35pts)**

In this question, you have to train decision tree models to classify whether the client has subscribed a term deposit or not (“y” variable). If the client has subscribed a term deposit, the target variable would be 1, and 0 otherwise.

[Variables]

|  |  |
| --- | --- |
| **Variable name** | **Description** |
| **Y** | **Dependent variable, indicating whether the client has subscribed a term deposit (binary: yes – 1 or no – 0)** |
| Age | Age of the individuals in years |
| Job | Type of job (nominal: “admin”, “unknown”, “unemployed”, “management”, “housemaid”, “entrepreneur”, “blue-collar”, “self-employed”, “retired”, “technician”, or “services”) |
| Marital | Marital status (nominal: “married”, “divorced”, or “single”) |
| Education | Education level (nominal: “unknown”, “secondary”, “primary”, or “tertiary”) |
| Default | Has credit in default? (binary: “yes” or “no”) |
| Balance | Average yearly balance in euros |
| Housing | Has housing loan? (binary: “yes” or “no”) |
| Loan | Has personal loan? (binary: “yes” or “no”) |
| Contact | Contact communication type (nominal: “unknown”, “telephone”, or “cellular”) |
| Day | Last contact day of the month |
| Month | Last contact month of year (nominal: “Jan”, “Feb”, … “Nov”, “Dec”) |
| Duration | Last contact duration, in seconds |
| Campaign | Number of contacts performed during this campaign and for this client including last contact. |
| Pdays | Number of days that passed by after the client was last contacted from a previous campaign (numeric: -1 means that client was not previously contacted) |
| Previous | Number of contacts performed before this campaign and for this client |
| Poutcome | Outcome of the previous marketing campaign (nominal: “unknown”, “other”, “failure”, “success”) |

(1) Train decision tree models by different maximum depth (1, 2, 3, 4, 5) with min\_samples\_leaf = 10 and the Gini impurity for the criterion to determine the best split, using a training set (x\_train, y\_train). Then, calculate the accuracy of the models using a validation set (x\_valid, y\_valid) for overall samples and individual classes, and fill the following table. (8pts)

|  |  |  |  |
| --- | --- | --- | --- |
| Depth | overall accuracy | No | Yes |
| 1 | 0.883003428 | 1 | 0 |
| 2 | 0.892292381 | 0.965435191 | 0.34026465 |
| 3 | 0.899259095 | 0.977082029 | 0.311909263 |
| 4 | 0.89903793 | 0.967939887 | 0.379017013 |
| 5 | 0.898485016 | 0.977708203 | 0.300567108 |

(2) Based on the results of Question (1), which model is the best? Describe your rationale. (6pts)

* The best model is depth 4 model. First, the overall accuracy of it is highest. Also it shows highest ‘Yes’ accuracy.

(3) Draw the trained tree with feature names when the maximum depth is set to 3. (5pts)

텍스트, 도표, 라인, 포스트잇 노트이(가) 표시된 사진

자동 생성된 설명

(4) Train decision tree models by different maximum depth (1, 2, 3, 4, 5) with min\_samples\_leaf = 10 and the entropy impurity for the criterion to determine the best split, using a training set (x\_train, y\_train). Then, calculate the accuracy of the models using a validation set (x\_valid, y\_valid) for overall samples and individual classes, and fill the following table. (8pts)

|  |  |  |  |
| --- | --- | --- | --- |
| Depth | overall accuracy | No | Yes |
| 1 | 0.883003 | 1 | 0 |
| 2 | 0.888311 | 0.945648 | 0.455577 |
| 3 | 0.898043 | 0.971947 | 0.340265 |
| 4 | 0.898043 | 0.971947 | 0.340265 |
| 5 | 0.898596 | 0.978585 | 0.294896 |

(5) Compare the two tree models of maximum depth 2 and 3 obtained for Question (4). (8pts)

|  |  |  |  |
| --- | --- | --- | --- |
| **Max Depth** | **Overall Accuracy** | **Class No Accuracy** | **Class Yes Accuracy** |
| 2 | 0.888311401 | 0.94564809 | 0.45557656 |
| 3 | 0.898042685 | 0.971947401 | 0.34026465 |

* Depth 3 gives better overall accuracy. While ‘Yes’ class accuracy is better on Depth 2

**3. -means clustering (30pts)**

This problem uses data generated from several normal distributions to apply k-means clustering.

k-means implemented in sci-kit learn can assign initial centroids through ‘init’. When init is set as by array ( = the number of clusters, = the number of features), each row is used as a centroid.

Ref: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

(1) Select randomly samples from the given data set and use them as initial centroids. This procedure is repeated for 10 times for each . You need to report the average and standard deviation of silhouette coefficient values of 10 iterations for each . (6pts)

For random selection of samples, refer the following page.

Ref: <https://numpy.org/doc/stable/reference/random/generated/numpy.random.choice.html>

|  |  |
| --- | --- |
| The number of clusters | Silhouette coefficient (standard deviation) |
| 3 | 0.63 (0.05) |
| 4 | 0.78 (0.16) |
| 5 | 0.65 (0.11) |
| 6 | 0.61 (0.08) |
| 7 | 0.56 (0.11) |
| 8 | 0.49 (0.12) |
| 9 | 0.44 (0.11) |
| 10 | 0.43 (0.09) |

* Average(standard deviation) (i.e. average = 0.63, standard deviation = 0.05)

(2) Use the default initialization strategy “k-means++”. This procedure is repeated for 10 times. Then, calculate the average and standard deviation silhouette coefficient values of 10 iterations for each . (6pts)

|  |  |
| --- | --- |
| The number of clusters | Silhouette coefficient (standard deviation) |
| 3 | 0.65 (0.00) |
| 4 | 0.78 (0.11) |
| 5 | 0.83 (0.00) |
| 6 | 0.82 (0.05) |
| 7 | 0.85 (0.00) |
| 8 | 0.86 (0.00) |
| 9 | 0.74 (0.07) |
| 10 | 0.55 (0.07) |

(3) Draw scatter plots for the given data with initial centroids and final centroids for the worst and best number of clusters among in Question (1) in terms of silhouette coefficient. The final centroids should be marked as blue ‘X’. (6pts)

스크린샷, 도표, 텍스트이(가) 표시된 사진

자동 생성된 설명

(4) Draw scatter plots for the worst and best number of clusters among of Question (2) in the same way as in Question (3). Do not need to plot initial centroids. (6pts)

스크린샷, 도표, 텍스트이(가) 표시된 사진

자동 생성된 설명

(5) Based on the different results for each case (Questions (1) and (2)), compare two different methods to determine initial centroids. (6pts)

- For the first case, it uses random initialization method which selects k samples from dataset randomly as initial centroids. This method is simple but the quality of clustering depends heavily on these initial centroids, and the algorithm may take longer to converge. In contrast, second case utilizes k-means++ initialization method The method selects the first centroid randomly from the data points, and then each subsequent centroid is chosen based on its distance from the already selected centroids. This process continues until k centroids are selected. The k-means++ method is more sophisticated in choosing initial centroids, leading to faster convergence and better clustering results compared to the random initialization method