

**Amirkabir University of Technology**

**(Tehran Polytechnic)**

**Computer Engineering Department**

**Machine Learning Course**

**CE5501**

**Homework 3**

**Student**

**Reza Sajedi**

**400131072**

**r.sajedi@aut.ac.ir**

**Teacher**

**Dr. Nazerfard**

**Fall 2022**

Table of Contents

[Problem 1: Why and how 1](#_Toc120819794)

[Part a 1](#_Toc120819795)

[Part b 1](#_Toc120819796)

[Part c 1](#_Toc120819797)

[Part d 2](#_Toc120819798)

[Part e 2](#_Toc120819799)

[Part f 3](#_Toc120819800)

[Problem 2: Breast Cancer 3](#_Toc120819801)

[Part a 3](#_Toc120819802)

[Part b 3](#_Toc120819803)

[Part c 4](#_Toc120819804)

[Part d 5](#_Toc120819805)

[Problem 3: Email Spam Classifier 6](#_Toc120819806)

[Part a 6](#_Toc120819807)

[Part b 6](#_Toc120819808)

[Part c 7](#_Toc120819809)

# Problem 1: Why and how

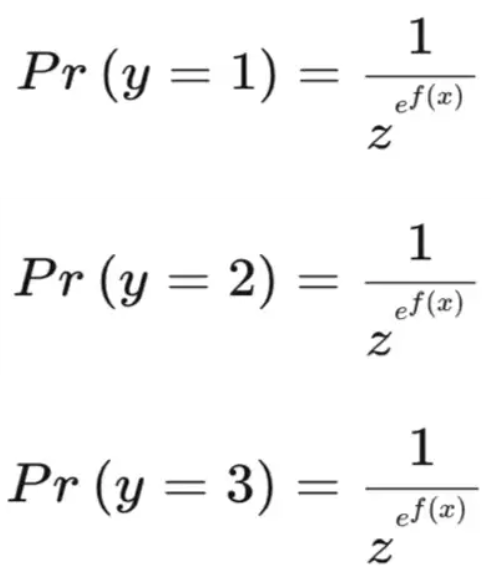
## Part a

The ROC curve has been widely used to adjust an appropriate threshold for binary classification. This curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. In this way, the point closest to the top left corner of the ROC curve is chosen as the threshold, and the threshold used to generate this point should be the optimal one. Another method is applying the Precision-Recall curve. It can be more informative than the ROC curve when the dataset is imbalanced.

## Part b

We will treat each class as a binary classification problem. This approach is called the one vs all method. In the one vs all method, when we work with a class, that class is denoted by 1 and the rest of the classes becomes 0.

The other way is to use multinomial logistic regression. To apply this, the logistic equation is transformed to allow probability for more than two classes. The probability equations with multinomial logistic regression for a classification task with three classes can be formulated as follows[[1]](#footnote-1):

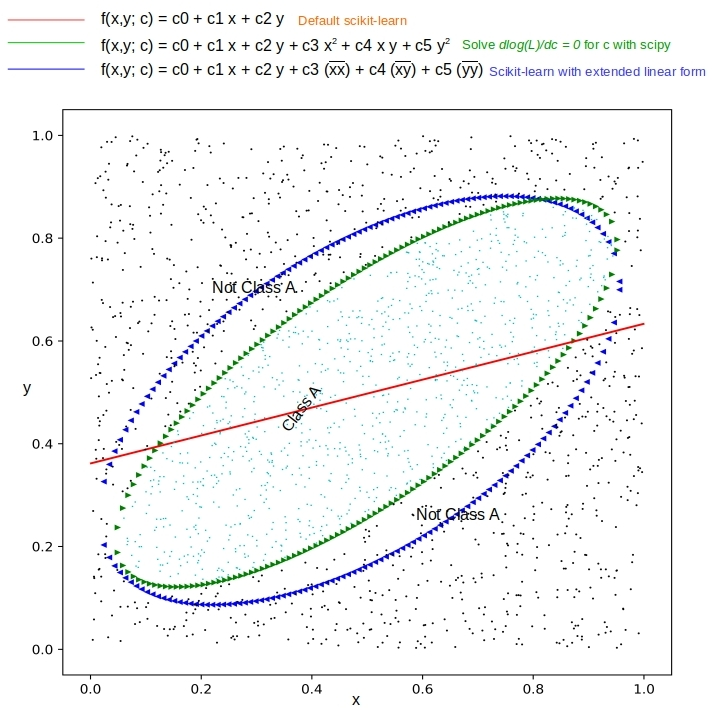


Where z is the sum of the e^f(x) for all classes in the model.

## Part c

Logistic Regression has traditionally been used as a linear classifier. So, the decision boundary can be formulated as follows:

We can have a non-linear decision boundary by considering a higher-order polynomial for f and transforming the dataset[[2]](#footnote-2):



## Part d

## Part e

**e.1)**

**e.2)**

**e.3)**

## Part f

Pr(p3) = Pr(p3 | p2) Pr(p2) + Pr(p3 | ~p2) Pr(~p2)

Pr(p2) = Pr(p2 | p1) Pr(p1) + Pr(p2 | ~p1) Pr(~p1) = 0.8 \* 0.4 + 0.5 \* 0.6 = 0.62

Pr(~p2) = 1 - Pr(p2) = 0.38

Pr(p3) = 0.2 \* 0.62 + 0.3 \* 0.38 = 0.238

Pr(~p3) = 1 - Pr(p3) = 0.762

Pr(p1 | p2, ~p3) = Pr(p2, ~p3 | p1) Pr(p1) / Pr(p2, ~p3)

= Pr(~p3 | p2) Pr(p2 | p1) Pr(p1) / Pr(~p3 | p2) Pr(p2)

= 0.8 \* 0.8 \* 0.4 / 0.8 \* 0.62 = 0.516

# Problem 2: Breast Cancer

## Part a

solver: Algorithm to use in the optimization problem. For small datasets, ‘liblinear’ is a good choice, whereas ‘sag’ is faster for large ones.

max\_iter: Maximum number of iterations taken for the solvers to converge.

param\_grid={

'solver': ['lbfgs', 'newton-cg', 'liblinear', 'sag', 'saga'],

'max\_iter': [100, 1000, 2500, 5000],

}

Best Parameters for Logistic Regression: {'max\_iter': 100, 'solver': 'lbfgs'}

Best Train Accuracy Score for Logistic Regression: 0.9802

Best Test Accuracy Score for Logistic Regression: 0.9649

Confusion Matrix for Logistic Regression:

[[34 0]

[ 4 76]]

## Part b

The results show that Logistic Regression’s accuracy is higher than Naïve Bayes (about 3%). LR and GNB are very similar. when the size of train data is small, GNB performs better because of the independency assumption between features, and LR performs better when the train data’s size is large.

var\_smoothing: Portion of the largest variance of all features that are added to variances for calculation stability. is a stability calculation to widen (or smooth) the curve and therefore account for more samples that are further away from the distribution mean. In this case, np.logspace returns numbers spaced evenly on a log scale, starts from 0, ends at -9, and generates 100 samples.

param\_grid={

'var\_smoothing': np.logspace(0, -9, 100)

}

Best Parameters for GaussianNB: {'var\_smoothing': 0.0533669923120631}

Best Train Accuracy Score for GaussianNB: 0.9362

Best Test Accuracy Score for GaussianNB: 0.9298

Confusion Matrix for GaussianNB:

[[32 2]

[ 6 74]]

## Part c

In this case, feature selection has improved the accuracy slightly for both LR and GNB (about 1%). It is worth it because when the dimension has been decreased, we have a simpler model with lower computation time.

|  |  |
| --- | --- |
| **Feature** | **AUC Score** |
| mean concave points | 0.9415 |
| worst concave points | 0.9290 |
| mean concavity | 0.8890 |
| mean area | 0.8765 |
| mean radius | 0.8640 |
| worst concavity | 0.8599 |
| worst radius | 0.8555 |
| worst area | 0.8555 |
| worst perimeter | 0.8515 |
| mean perimeter | 0.8430 |
| area error | 0.8239 |
| radius error | 0.8176 |
| perimeter error | 0.8176 |
| mean compactness | 0.7989 |
| worst compactness | 0.7676 |
| concave points error | 0.6982 |
| worst smoothness | 0.6813 |
| worst symmetry | 0.6768 |
| mean smoothness | 0.6415 |
| mean texture | 0.6397 |
| compactness error | 0.6309 |
| worst fractal dimension | 0.6099 |
| concavity error | 0.5886 |
| mean symmetry | 0.5868 |
| worst texture | 0.5724 |
| mean fractal dimension | 0.5000 |
| texture error | 0.5000 |
| smoothness error | 0.5000 |
| symmetry error | 0.5000 |
| fractal dimension error | 0.5000 |

Accuracy Score for Logistic Regression: 0.9737

Improvement for Logistic Regression Compared to Part a: 0.0088

Confusion Matrix for Logistic Regression:

[[34 0]

[ 3 77]]

Accuracy Score for GaussianNB: 0.9386

Improvement for GaussianNB Compared to Part b: 0.0088

Confusion Matrix for GaussianNB:

[[32 2]

[ 5 75]]

## Part d

In this case, the results show a decrement in accuracy. It depends on the selected train data because we are splitting the data set randomly, sometimes feature selection has good effects but in this case most of the time it leads to poor performance .

Accuracy Score for Logistic Regression: 0.9298

AUC Score for Logistic Regression: 0.95

Improvement for Logistic Regression Compared to Part a: -0.0351

Improvement for Logistic Regression Compared to Part c: -0.0439

Confusion Matrix for Logistic Regression:

[[34 0]

[ 8 72]]

Accuracy Score for GaussianNB: 0.886

AUC Score for GaussianNB: 0.9103

Improvement for GaussianNB Compared to Part b: -0.0439

Improvement for GaussianNB Compared to Part c: -0.0526

Confusion Matrix for GaussianNB:

[[33 1]

[12 68]]

Accuracy Score for MultinomialNB: 0.9123

AUC Score for MultinomialNB: 0.8868

Confusion Matrix for MultinomialNB:

[[28 6]

[ 4 76]]

# Problem 3: Email Spam Classifier

## Part a

Preprocessing steps of textual data can be listed as follows:

* Lowercasing all of the words
* Removing punctuation characters
* Removing single character words, because usually they have no meaning.
* Removing stop words (like 'the', 'any', 'such', 'this') because they are common words in every context and don’t give us information
* Removing non-English words
* Stemming (words like 'go', 'goes', 'going' will be replaced by a single 'go')



## Part b

We use the Bayes theorem to find out the likelihood of an email being spam. Each email consists of multiple words. We should calculate the probability of each word being considered spam word.

We calculate spam probability of an email by combining the spam probabilities of all contained words in that email.

It is better to use the alternative form of the above formula to avoid floating point underflow.

Rare words can cause some problems in our calculations. We can handle that by using the corrected probability.

Where s is the strength we give to background information about incoming spam and n is the number of occurrences of this word during the learning phase.

## Part c

TP: 244

TN: 790

FP: 79

FN: 32

Accuracy: 0.9031

F1-score: 0.8147

Precision: 0.7554

TPR (Recall, Sensitivity): 0.8841

TNR (Specificity): 0.9091

1. <https://pub.towardsai.net/logistic-regression-for-multi-class-classification-hands-on-with-scikit-learn-bcc0bbad1def> [↑](#footnote-ref-1)
2. <https://xplordat.com/2019/03/13/logistic-regression-as-a-nonlinear-classifier/> [↑](#footnote-ref-2)