Lecture 4

Naive Bayes Classifier

By: Nazerke Sultanova

Question1:
Supervised Classification Problems:

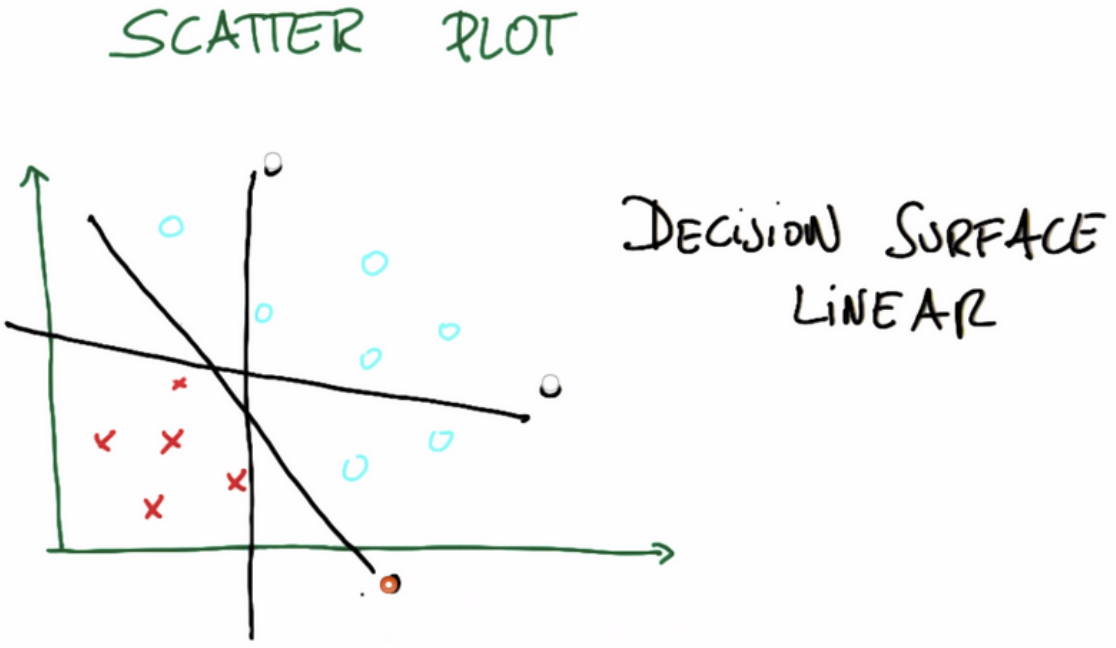
- From an album of tagged pictures, recognize someone in a picture
- Analyze bank data for weird-looking transactions, and flag those for fraud
- Given someone’s music choices, recommend a new song
- Divide student groups into types based on learning styles

Question1:
Supervised Classification Problems:

- From an album of tagged pictures, recognize someone in a picture
- Analyze bank data for weird-looking transactions, and flag those for fraud
- Given someone’s music choices, recommend a new song
- Divide student groups into types based on learning styles

Question2:

- Which line is the best decision boundary?



Naive Bayes Classifier

- Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features.

Example

- P(C) = 0.01

  the probability of having cancer is 1%

- Test:

  90% it is positive if you have C

  90% it is negative if you don’t have C

- Question: Test is positive

  What is the probability of having C?

What do you think the P is?

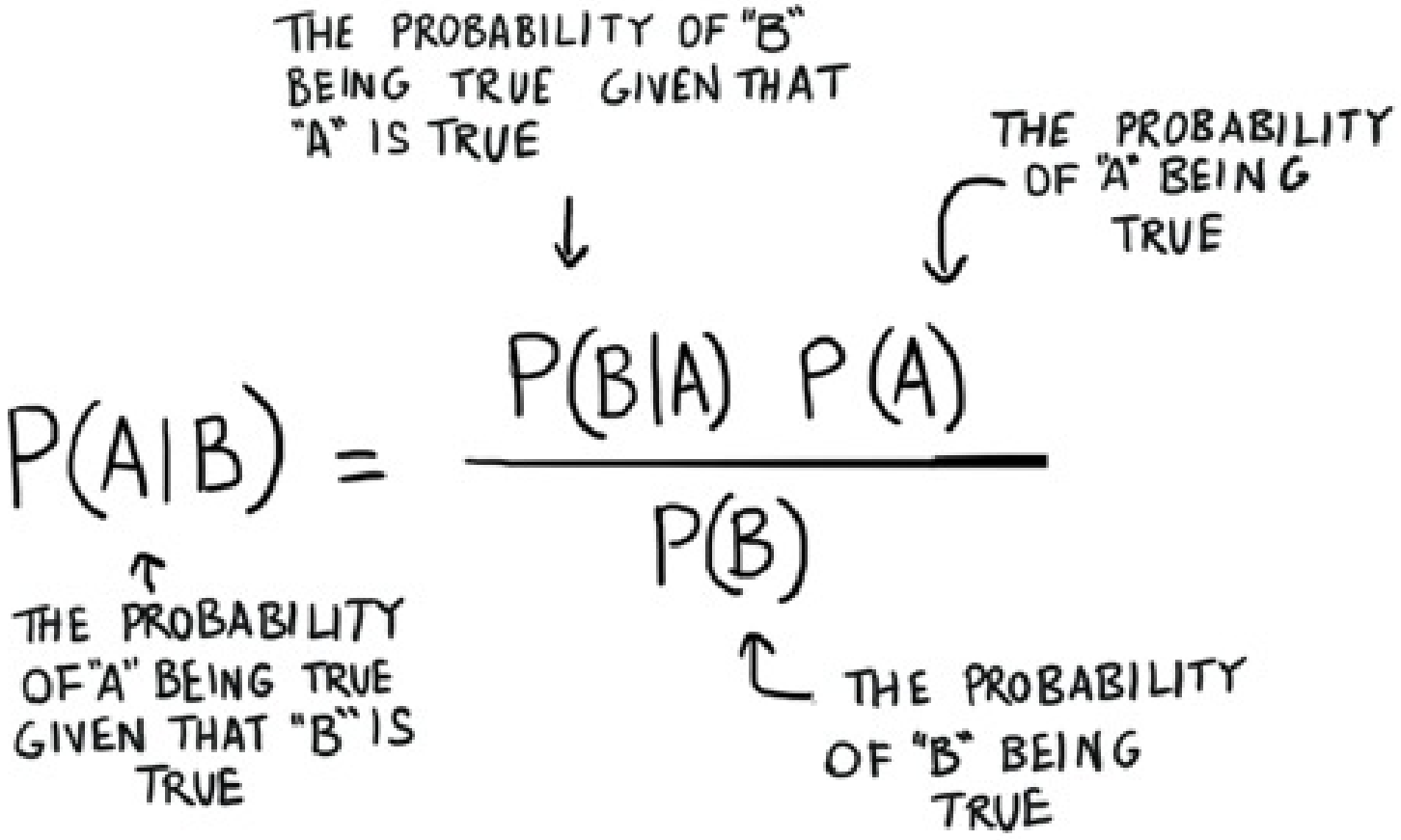
- 90%
- 1%
- 8%

What do you think the P is?

- 90%
- 1%
- 8%

Solution on the blackboard

Bayes Rule


THE PROBABILITY OF "B" BEING TRUE GIVEN THAT "A" IS TRUE
THE PROBABILITY OF "A" BEING TRUE
$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$
THE PROBABILITY OF "A" BEING TRUE GIVEN THAT "B" IS TRUE
THE PROBABILITY OF "B" BEING TRUE

Bayes Rule

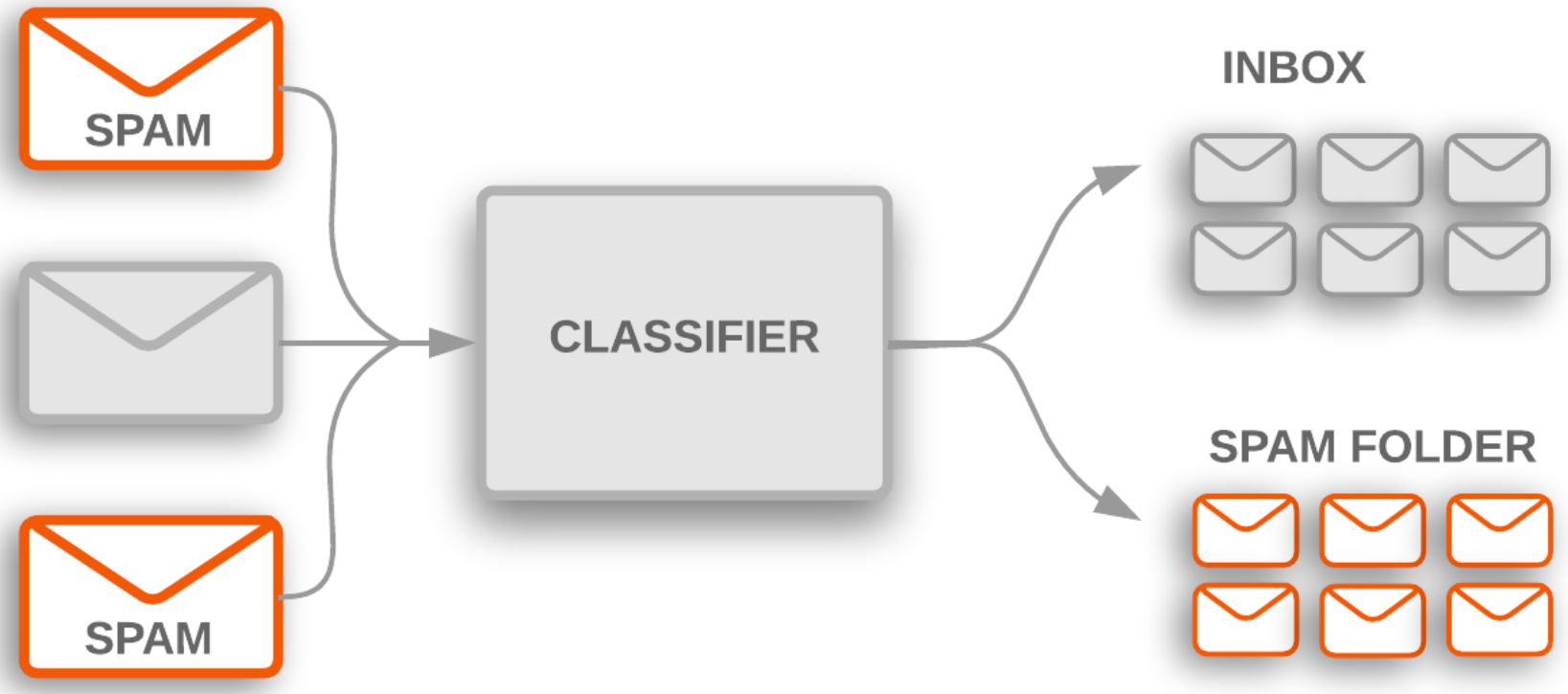
$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

OR

$$P(A/B) = \frac{P(B/A)P(A)}{P(B/A)P(A) + P(B/\sim A)P(\sim A)}$$

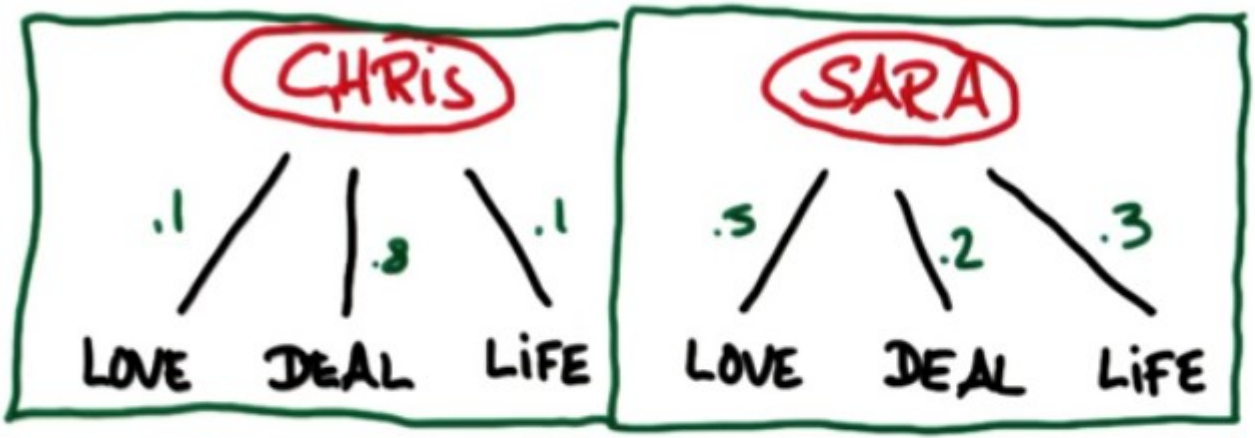
From the "Law of Total Probability"

Naive Bayes for Text Classification



Question 3:

- Who wrote this email?



P(CHRIS) = 0.5

P(SARA) = 0.5

LOVE LIFE !

- CHRIS
- SARA

Why Naive Bayes **Naive?**

- It ignores
  - Words
  - Word order
  - Length of a word

Getting started with **scikit-learn**

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on **NumPy, SciPy**, and **matplotlib**
- Open source, commercially usable - BSD license

1.9. Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features. Given a class variable y and a dependent feature vector x_1 through x_n, Bayes' theorem states the following relationship:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots x_n \mid y)}{P(x_1, \dots, x_n)}$$

Using the naive independence assumption that

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y),$$

for all i, this relationship is simplified to

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$; the former is then the relative frequency of class y in the training set.

1.9.1. Gaussian Naive Bayes

`GaussianNB` implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters σ_y and μ_y are estimated using maximum likelihood.

```
>>> from sklearn import datasets
>>> iris = datasets.load_iris()
>>> from sklearn.naive_bayes import GaussianNB
>>> gnb = GaussianNB()
>>> y_pred = gnb.fit(iris.data, iris.target).predict(iris.data)
>>> print("Number of mislabeled points out of a total %d points : %d"
...       % (iris.data.shape[0],(iris.target != y_pred).sum()))
Number of mislabeled points out of a total 150 points : 6
```

Assignment 4:

- Import breast_cancer datasets using **scikit-learn**
- Divide dataset to train and test as 70:30 ratio
- Import GaussianNB, fit and predict the values of test dataset
- Calculate accuracy of test data. Write your own function for accuracy