Lecture 5: Classification and Naive Bayes classifier
Statistical Methods for Data Science

Yinan Yu

Department of Computer Science and Engineering

November 16, 2020

Today

Learning outcome

- Be able to explain classification related terminology: classification, binary/multi-class classification, annotation, training, validation and testing, etc
- Be able to calculate and interpret TP, TN, FP, FN, accuracy, precision, recall, specificity, F1 score
- Understand basic concepts of performance evaluation and comparison of different classifiers
- Be able to explain the Bayes' rule for both multinomial and Gaussian naive Bayes classifiers
- Be able to explain the differences between the multinomial naive Bayes classifier and the Gaussian naive Bayes classifier
- For a given problem, be able to formulate and implement a naive Bayes classifier

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Today

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Modeling for classification

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**;

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**;

Classification　　**Modeling for classification**
Naive Bayes classifier　　Training, validation and test
Summary　　Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**
    - Typically, x is a vector denoted by $\boldsymbol{x} = [x_1, \cdots, x_d]$; sometimes the notations x and $\boldsymbol{x}$ are used interchangeably

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**
    - Typically, x is a vector denoted by $x = [x_1, \cdots, x_d]$; sometimes the notations x and x are used interchangeably
    - x is called a **feature vector**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**
    - Typically, x is a vector denoted by $\boldsymbol{x} = [x_1, \cdots, x_d]$; sometimes the notations x and $\boldsymbol{x}$ are used interchangeably
    - $\boldsymbol{x}$ is called a **feature vector**
  - g: **classification model**, e.g. naive Bayes classifiers, support vector machines, decision trees, etc

Classification Modeling for classification
Naive Bayes classifier Training, validation and test
Summary Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**
    - Typically, x is a vector denoted by $\boldsymbol{x} = [x_1, \cdots, x_d]$; sometimes the notations x and $\boldsymbol{x}$ are used interchangeably
    - $\boldsymbol{x}$ is called a **feature vector**
  - g: **classification model**, e.g. naive Bayes classifiers, support vector machines, decision trees, etc
  - θ (parameters) and h (hyperparameters) depend on g

Classification | Modeling for classification
Naive Bayes classifier | Training, validation and test
Summary | Performance evaluation

Classification

- Recall (cf. lecture 3): modeling is to *describe* a system using mathematics in order to solve a range of problems. The description has this form:

$$y = g(x; \theta \mid h)$$

- Classification:
  - y: **categorical (nominal)**, scalar - each category is called a **class**; when there are only two classes, i.e. $y \in \{0, 1\}$, the classification problem is called **binary classification**; if there are more than two classes, i.e. $y \in \{1, \cdots, C\}$ for $C > 2$, the classification problem is called **multi-class classification**
  - x: **categorical** or **numerical**
    - Typically, x is a vector denoted by $\boldsymbol{x} = [x_1, \cdots, x_d]$; sometimes the notations x and $\boldsymbol{x}$ are used interchangeably
    - $\boldsymbol{x}$ is called a **feature vector**
  - g: **classification model**, e.g. naive Bayes classifiers, support vector machines, decision trees, etc
  - θ (parameters) and h (hyperparameters) depend on g

  Note: if you are a "bottom-up" kind of thinker and have a hard time interpreting g, just imagine it as

  $$y = g(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases}, \quad \text{for two classes 0 and 1}$$

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms
- Parameter estimation is called **training**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms
- Parameter estimation is called **training**
- Given an estimate $\hat{\theta}$, $g(x; \hat{\theta} \mid h)$ is a (trained) **classifier**

Classification | **Modeling for classification**
Naive Bayes classifier | Training, validation and test
Summary | Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms
- Parameter estimation is called **training**
- Given an estimate $\hat{\theta}$, $g(x; \hat{\theta} \mid h)$ is a (trained) **classifier**
- In supervised learning, the parameters are estimated from a data set called the **training data set** which corresponds to the 20 ducks from lecture 4

Classification | Modeling for classification
Naive Bayes classifier | Training, validation and test
Summary | Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms
- Parameter estimation is called **training**
- Given an estimate $\hat{\theta}$, $g(x; \hat{\theta} \mid h)$ is a (trained) **classifier**
- In supervised learning, the parameters are estimated from a data set called the **training data set** which corresponds to the 20 ducks from lecture 4
- The training data set contains paired data $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, e.g.

$$\{(\text{}, scoter), (\text{}, goldeneye), \cdots, (\text{}, scoter)\}$$

where $\boldsymbol{x}_i =$ pixel values in a picture, $y_i \in \{scoter, goldeneye\}$ is called the **ground truth labels**; the data set is called a **labeled data set**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Parameter estimation

- Classification models are **supervised learning** algorithms
- Parameter estimation is called **training**
- Given an estimate $\hat{\theta}$, $g(x; \hat{\theta} \mid h)$ is a (trained) **classifier**
- In supervised learning, the parameters are estimated from a data set called the **training data set** which corresponds to the 20 ducks from lecture 4
- The training data set contains paired data $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, e.g.

$$\{(\text{■}, scoter), (\text{■}, goldeneye), \cdots, (\text{■}, scoter)\}$$

where $\boldsymbol{x}_i = $ pixel values in a picture, $y_i \in \{scoter, goldeneye\}$ is called the **ground truth labels**; the data set is called a **labeled data set**
- The targets y_i's in the training set are typically created by humans. The process of creating the ground truth labels is called **annotation** or **labeling**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Training, validation and test

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Problem solving in data science
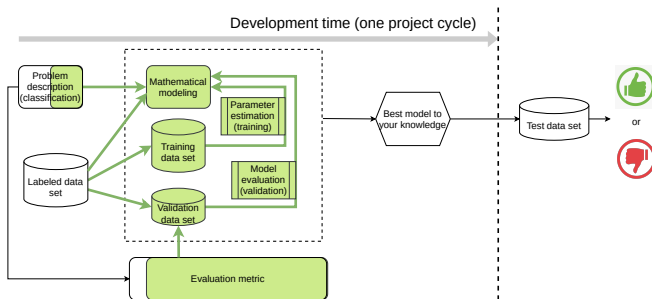
Figure: The (green) shadowed boxes are the actions performed by the data scientist.

Classification   Modeling for classification
Naive Bayes classifier   **Training, validation and test**
Summary   Performance evaluation

During development: training and validation

During development: split the available data set into a training data set and a validation data set

- **Training data set**: to estimate the parameters
- **Validation data**: to evaluate the performance of one or more classifiers
  What is being evaluated:
    - g (the selection of a family of models with the functional form g)
      Philosophies:
        - **Occam's razor**: if multiple models are showing similar performances, the simplest model is preferred
        - **All models are wrong, but some are useful**
    - $\hat{\theta}$ (parameter estimation method)
    - h (hyperparameter tuning)

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

After development: testing

After development:

- **Test data set**: the real deal - you DO NOT have access to it during development

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

After development: testing

After development:

- **Test data set**: the real deal - you DO NOT have access to it during development

For the project, do not use the "test data set" for development! Don't even look at it before testing your model!!! 😠 Because in reality, you do not have access to it!

Classification
Naive Bayes classifier
Summary
Modeling for classification
Training, validation and test
Performance evaluation

Performance evaluation

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to split the data

Given a labeled data set $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$, split the data set into a **training data set** and a **validation data set**

- Training-validation split
  - $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$, $\{(x_5, y_5), (x_6, y_6)\}$
- K-fold cross validation, e.g. 3-fold
  - $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$, $\{(x_5, y_5), (x_6, y_6)\}$
  - $\{(x_1, y_1), (x_2, y_2), (x_5, y_5), (x_6, y_6)\}$, $\{(x_3, y_3), (x_4, y_4)\}$
  - $\{(x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$, $\{(x_1, y_1), (x_2, y_2)\}$
- Leave-one-out cross validation
  - $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5)\}$, $\{(x_6, y_6)\}$
  - $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_6, y_6)\}$, $\{(x_5, y_5)\}$
  - $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_5, y_5), (x_6, y_6)\}$, $\{(x_4, y_4)\}$
  - $\{(x_1, y_1), (x_2, y_2), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$, $\{(x_3, y_3)\}$
  - $\{(x_1, y_1), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$, $\{(x_2, y_2)\}$
  - $\{(x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$, $\{(x_1, y_1)\}$

Classification　　Modeling for classification
Naive Bayes classifier　　Training, validation and test
Summary　　**Performance evaluation**

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
    - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
    - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
    - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
    - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec)

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec) **TP**+**FN**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec) **TP**+**FN**
- What is count($y = 0$)? (15 sec)

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec) **TP**+**FN**
- What is count($y = 0$)? (15 sec) **TN**+**FP**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec) **TP**+**FN**
- What is count($y = 0$)? (15 sec) **TN**+**FP**
- What is size of the entire data set, i.e. count($y = 1$)+count($y = 0$)? (15 sec)

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Validation: four outcomes

- Given a **binary classification** problem, a trained classifier $g(x; \hat{\theta} \mid h)$ and a validation data set containing pairs (x, y)
- Positive: $y = 1$; negative: $y = 0$
- Compute $\hat{y} = g(x; \hat{\theta} \mid h)$ on the validation data set
  - **True Positive (TP):** count(ground truth $y = 1$, classifier output $\hat{y} = 1$)
  - **False Positive (FP):** count(ground truth $y = 0$; classifier output $\hat{y} = 1$)
  - **True Negative (TN):** count(ground truth $y = 0$; classifier output $\hat{y} = 0$)
  - **False Negative (FN):** count(ground truth $y = 1$; classifier output $\hat{y} = 0$)

Confusion matrix (contingency table)

	$y = 1$	$y = 0$
$\hat{y} = 1$	**TP**	**FP** (Type I error)
$\hat{y} = 0$	**FN** (Type II error)	**TN**

- What is count($y = 1$)? (15 sec) **TP**+**FN**
- What is count($y = 0$)? (15 sec) **TN**+**FP**
- What is size of the entire data set, i.e. count($y = 1$)+count($y = 0$)? (15 sec) **TP**+**FN**+**TN**+**FP**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Evaluation metric

- Accuracy:

$$accuracy = \frac{\mathbf{TP} + \mathbf{TN}}{count(y = 1) + count(y = 0)} = \frac{\mathbf{TP} + \mathbf{TN}}{\mathbf{TP} + \mathbf{TN} + \mathbf{FP} + \mathbf{FN}}$$

- True Positive Rate (recall, sensitivity):

$$TPR = \frac{\mathbf{TP}}{count(y = 1)} = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FN}}$$

- True Negative Rate (specificity):

$$TNR = \frac{\mathbf{TN}}{count(y = 0)} = \frac{\mathbf{TN}}{\mathbf{TN} + \mathbf{FP}}$$

- Precision:

$$precision = \frac{\mathbf{TP}}{count(\hat{y} = 1)} = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FP}}$$

- F1 score:

$$F = 2 \times \frac{precision \times recall}{precision + recall}$$

- More from scikit-learn:
  `www.scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics`

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Evaluation metric (cont.)

- **Use with caution** especially for **imbalanced** data set, e.g. data from medical measurements typically contains more negative data (90% healthy volunteers) than positive data (10% patients); the data set is then imbalanced
  - Terrible metric: accuracy
  - Okay metric:
    - Precision vs recall
    - Sensitivity vs specificity
    - F1 score

  Reference: read the data science design manual, section 7.4.1
- In this lecture, we only consider **binary classification** $c \in \{0, 1\}$
- In the multi-class case $c \in \{1, \cdots, C\}$:
  - **Macro**: the metrics are computed for each class c and then the average is calculated
  - **Micro**: the metrics are computed globally for all classes

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   **Performance evaluation**

Finding a good classifier

- **Given**: a labeled data set with $y_i \in \{0, 1\}$

$$\{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5), (x_6, y_6)\}$$

- **Task**: use this data set to build a good classifier
- **Method**:
  1) Come up with M models g_m, $m = 1, \cdots, M$. Let $\hat{y}_i^m = g_m(x_i; \theta \mid h)$

     For the sake of simplicity (not to overwhelm you with complex notations), we neglect the different choices of h. In practice, when you choose a different set of hyperparameters h, the model needs to be evaluated as a different model.
  2) Split the data set (cf. page 13) into a **training data set** and a **validation data set**
     - Training-validation split
     - K-fold cross validation
     - Leave-one-out cross validation
  3) Estimate parameters $\hat{\theta}$ using the **training data set**
     - Similar to the choice of h, different parameter estimation techniques will give a different $\hat{\theta}$, which in turn needs to be evaluated.
  4) **Evaluate** $g_m(x; \hat{\theta} \mid h)$ on the **validation data set**
     - $M = 1$: evaluation of one classifier
     - $M > 1$: comparison of multiple classifiers

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to evaluate a classifier?

Given a classifier $g(x; \hat{\theta} \mid h)$,

- **Goal:** to evaluate the performance of the classifier $g(x; \hat{\theta} \mid h)$
- **Steps:**
  - Step 1: compute $\hat{y}_i = g(x_i; \hat{\theta} \mid h)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16), denoted by s
- **Interpretation:**
  - Case 1: one validation data set (e.g. training-validation split, leave-one-out cross validation) - only one s, e.g. $s = 0.92$
  - Case 2: multiple validation data sets (e.g. K-fold cross validation) - a set of s, e.g. for 3-fold cross validation, we have 3 validation datasets, which gives us 3 results $\{s_1, s_2, s_3\}$. We can then compute sample statistics (e.g. sample mean, sample standard deviation) from this set.

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$ Example: say we choose the accuracy as the metric

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
    Example: say we choose the accuracy as the metric
    - Question: if $s^1 = 0.92$ and $s^2 = 0.52$, which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$?

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
    Example: say we choose the accuracy as the metric
    - Question: if $s^1 = 0.92$ and $s^2 = 0.52$, which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$?
    - Answer: probably $g_1(x; \hat{\theta} \mid h)$ since $s^1 \gg s^2$

Classification Modeling for classification
Naive Bayes classifier Training, validation and test
Summary **Performance evaluation**

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
    Example: say we choose the accuracy as the metric
    - Question: if $s^1 = 0.92$ and $s^2 = 0.52$, which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$?
    - Answer: probably $g_1(x; \hat{\theta} \mid h)$ since $s^1 \gg s^2$
    - Question: if $s^1 = 0.92$ and $s^2 = 0.91$, now which classifier is better?

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
  Example: say we choose the accuracy as the metric
    - Question: if $s^1 = 0.92$ and $s^2 = 0.52$, which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$?
    - Answer: probably $g_1(x; \hat{\theta} \mid h)$ since $s^1 \gg s^2$
    - Question: if $s^1 = 0.92$ and $s^2 = 0.91$, now which classifier is better?
    - Still $g_1(x; \hat{\theta} \mid h)$ with $s^1 \approx s^2$?

Classification   Modeling for classification
Naive Bayes classifier   Training, validation and test
Summary   **Performance evaluation**

How to compare two classifiers?

Given two classifiers $g_1(x; \hat{\theta} \mid h)$ and $g_2(x; \hat{\xi} \mid t)$ (note: I use different symbols $\hat{\theta}$ and $\hat{\xi}$ to indicate that they might have different parameters; likewise, h and t for different hyperparameters)

- **Goal:** to check which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
- **Steps:**
  - Step 1: compute $\hat{y}_i^1 = g_1(x_i; \hat{\theta} \mid h)$ and $\hat{y}_i^2 = g_2(x_i; \hat{\xi} \mid t)$ on the validation data set $\{(x_1, y_1), \cdots, (x_n, y_n)\}$
  - Step 2: compute an evaluation metric (e.g. page 16) for each classifier, denoted by s^j for $j = 1, 2$
  - Step 3: compare s^1 and s^2 to choose which classier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$
    Example: say we choose the accuracy as the metric
    - Question: if $s^1 = 0.92$ and $s^2 = 0.52$, which classifier is better, $g_1(x; \hat{\theta} \mid h)$ or $g_2(x; \hat{\xi} \mid t)$?
    - Answer: probably $g_1(x; \hat{\theta} \mid h)$ since $s^1 \gg s^2$
    - Question: if $s^1 = 0.92$ and $s^2 = 0.91$, now which classifier is better?
    - Still $g_1(x; \hat{\theta} \mid h)$ with $s^1 \approx s^2$?
    We need to use **hypothesis testing** (lecture 7) to determine which classifier is better.
    - Case 1: one validation data set (e.g. training-validation split, leave-one-out cross validation) - **McNemar's test**
    - Case 2: multiple validation data sets (e.g. K-fold cross validation) - **paired t-test**

Classification
Naive Bayes classifier
Summary

Modeling for classification
Training, validation and test
Performance evaluation

Summary

- Classification, binary/multi-class classification
- Training, validation and test data set
- TP, TN, FP, FN
- Evaluation metrics
- Basic concepts of performance evaluation and comparison of different classifiers

Today

1. Classification

2. Naive Bayes classifier
   - Multinomial naive Bayes classifier
   - Gaussian naive Bayes classifier

3. Summary

Naive Bayes classifier

- Multinomial naive Bayes classifier (categorical y, categorical x)
- Gaussian naive Bayes classifier (categorical y, continuous x)

Naive Bayes classifier

- Multinomial naive Bayes classifier (categorical y, categorical x)
- Gaussian naive Bayes classifier (categorical y, continuous x)

Bayes' rule and MAP for parameter estimation

- In parameter estimation:

$$f_{\Theta|data}(\theta \mid data) = \frac{\overbrace{f_{data|\Theta}(data \mid \theta)}^{\text{likelihood}} \overbrace{f_{\Theta}(\theta)}^{\text{prior}}}{\underbrace{f_{data}(data)}_{\text{normalization constant}}}$$

$$\hat{\theta}_{MAP} = \arg \max_{\theta} f_{data|\Theta}(data \mid \theta) f_{\Theta}(\theta)$$

Bayes' rule and MAP for naive Bayes classifier

Let x be the input variables and y the target,

- In multinomial naive Bayes classifier (categorical y, categorical x):

$$P(Y = y \mid X = x) = \frac{\overbrace{P(X = x \mid Y = y)}^{\text{likelihood}}\overbrace{P(Y = y)}^{\text{prior}}}{\underbrace{P(X = x)}_{\text{normalization constant}}}$$

$$\hat{y}_{MAP} = \arg\max_{y} P(X = x \mid Y = y)P(Y = y)$$

- In Gaussian naive Bayes classifier (categorical y, continuous x):

$$P(Y = y \mid X = x) = \frac{\overbrace{f_{X|Y=y}(x \mid Y = y)}^{\text{likelihood}}\overbrace{P(Y = y)}^{\text{prior}}}{\underbrace{f_X(x)}_{\text{normalization constant}}}$$

$$\hat{y}_{MAP} = \arg\max_{y} f_{X|Y=y}(x \mid Y = y)P(Y = y)$$

Classification
Naive Bayes classifier
Summary
Multinomial naive Bayes classifier
Gaussian naive Bayes classifier

Multinomial naive Bayes classifier

Example 1: spam filter

An email server would like to build a spam filter for its clients

- **Input variables x**: the body of an email
- **Training data**: there are 1000 emails labeled either "spam" or "not spam"
- **Prediction task**: for a new email, the server would like to identify if it is a spam

Example 1: spam filter

An email server would like to build a spam filter for its clients

- **Input variables x**: the body of an email
- **Training data**: there are 1000 emails labeled either "spam" or "not spam"
- **Prediction task**: for a new email, the server would like to identify if it is a spam
- **Model g**: **Multinomial naive Bayes classifier**

Modeling for spam filter

- **Prediction** y: spam or not spam
- **Variables** x_i, $i = 1, \cdots, n$: the body of an email with n words
  - **Assumptions:**
    - the words are independent - a bag of words - **NAIVE!**
    - the words are generated from a categorical distribution; each category is a word from a vocabulary
- **Model** g: **multinomial naive Bayes classifier**

$$\hat{y} = g(x_1, \cdots, x_n) = \underset{c \in \{spam,\ not\ spam\}}{\arg\max} \ P(c) \prod_{i=1}^{n} P(x_i \mid c)$$

where $P(c)$ is the prior and $\prod_{i=1}^{n} P(x_i \mid c)$ is the likelihood under the assumptions
Note: it is the maximum a posteriori estimation of the label (spam or not spam)
- **Hyperparameters** h: smoothing factor α (explained soon)
- **Parameters** θ: $P(c)$, a vocabulary (if not given) and $P(word \mid c)$ for all words in the vocabulary

Parameter estimation (training)

In this demo, we only consider 7 training emails for illustration purposes

- **Training data**: there are 7 emails labeled either "spam" or "not spam"
  - Email 1 (not spam): "Hi see you at dinner."
  - Email 2 (spam): "Buy lottery!"
  - Email 3 (not spam): "Hi, wanna have dinner?"
  - Email 4 (not spam): "Hi you, nice dinner today!"
  - Email 5 (spam): "Wanna get rich today?"
  - Email 6 (not spam): "Lottery dinner?"
  - Email 7 (spam): "Win lottery; get rich today!"

Parameter estimation (training) (cont.)

- Step 1: Estimate the likelihood $P(word_i \mid spam)$ and $P(word_i \mid not\ spam)$

  1.1 Build a vocabulary containing all unique words from the 7 emails:

  $V = \{$buy, lottery, wanna, get, rich, today, win, hi, see, you, at, dinner, have, nice$\}$

  1.2 Count how many times each word appears in both spams and not spams:

	buy	lottery	wanna	...	dinner	have	nice
Spam	1	2	1	...	0	0	0
Not spam	0	1	1	...	4	1	1
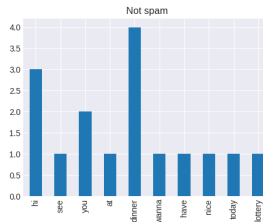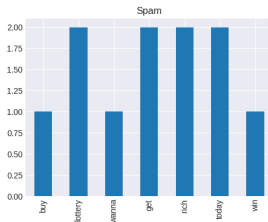
Parameter estimation (training) (cont.)

- Step 1 (cont.):
  - 1.3 Count how many words in total for each class:
    - Spam: 11 words
    - Not spam: 16 words
  - 1.4 Estimate the **likelihood** $P(word_i \mid spam)$ **and** $P(word_i \mid not\ spam)$ for all $w_i \in V$:

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1}{11}$	$\frac{2}{11}$	...	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0}{16}$	$\frac{1}{16}$	...	$\frac{2}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

- Step 2: Estimate the **prior** $P(spam)$ **and** $P(not\ spam)$
  - Spam $P(spam)$: $\frac{\#\ of\ spams}{\#\ of\ total\ emails} = \frac{3}{7}$
  - Not spam $P(not\ spam)$: $\frac{\#\ of\ not\ spams}{\#\ of\ total\ emails} = \frac{4}{7}$

Classify a new email

- Construct the multinomial naive Bayes classifier
  The naive Bayes classifier is a function of a given email. Let s_{spam} and $s_{not\ spam}$ be the posterior without the normalization constant

$$s_{spam} = P(spam) \prod_{\forall word \in email} P(word \mid spam)$$

$$s_{not\ spam} = P(not\ spam) \prod_{\forall word \in email} P(word \mid not\ spam)$$

  - If $s_{spam} > s_{not\ spam}$: the email is spam
  - If $s_{spam} \leq s_{not\ spam}$: the email is not spam

Classify a new email (cont.)

- Compute the posterior of this email
  - Spam: $P(spam \mid an\ email) = \frac{s_{spam}}{s_{spam} + s_{not\ spam}}$
  - Not spam: $P(not\ spam \mid an\ email) = \frac{s_{not\ spam}}{s_{spam} + s_{not\ spam}}$

  These are the probability of the email being a spam and not a spam, respectively.

One problemo

Say, the email is "You! Lottery! Lottery! Lottery!!" and it is clearly a spam.
But when we compute the likelihood (cf. page 31),

$$s_{spam} = P(spam)P("you" \mid spam)P("lottery" \mid spam)^3$$

$$s_{not\ spam} = P(not\ spam)P("you" \mid not\ spam)P("lottery" \mid not\ spam)^3$$

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1}{11}$	$\frac{2}{11}$	...	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0}{16}$	$\frac{1}{16}$	...	$\frac{2}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

One problemo

Say, the email is "You! Lottery! Lottery! Lottery!!" and it is clearly a spam.
But when we compute the likelihood (cf. page 31),

$$s_{spam} = P(spam)P("you" \mid spam)P("lottery" \mid spam)^3$$

$$s_{not\ spam} = P(not\ spam)P("you" \mid not\ spam)P("lottery" \mid not\ spam)^3$$

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1}{11}$	$\frac{2}{11}$	...	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0}{16}$	$\frac{1}{16}$	...	$\frac{2}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

- $s_{spam} = \frac{3}{7} \times \frac{0}{11} \times \frac{2}{11}^3 = \mathbf{0}$

One problemo

Say, the email is "You! Lottery! Lottery! Lottery!!" and it is clearly a spam.
But when we compute the likelihood (cf. page 31),

$$s_{spam} = P(spam)P("you" \mid spam)P("lottery" \mid spam)^3$$

$$s_{not\ spam} = P(not\ spam)P("you" \mid not\ spam)P("lottery" \mid not\ spam)^3$$

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1}{11}$	$\frac{2}{11}$	...	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0}{16}$	$\frac{1}{16}$	...	$\frac{2}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

- $s_{spam} = \frac{3}{7} \times \frac{0}{11} \times \frac{2}{11}^3 = \mathbf{0}$
- $s_{not\ spam} = \frac{4}{7} \times \frac{2}{16} \times \frac{1}{16}^3 > s_{spam}$

One problemo

Say, the email is "You! Lottery! Lottery! Lottery!!" and it is clearly a spam.
But when we compute the likelihood (cf. page 31),

$$s_{spam} = P(spam)P("you" \mid spam)P("lottery" \mid spam)^3$$

$$s_{not\ spam} = P(not\ spam)P("you" \mid not\ spam)P("lottery" \mid not\ spam)^3$$

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1}{11}$	$\frac{2}{11}$	...	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$	$\frac{0}{11}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0}{16}$	$\frac{1}{16}$	...	$\frac{2}{16}$	$\frac{1}{16}$	$\frac{4}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

- $s_{spam} = \frac{3}{7} \times \frac{0}{11} \times \frac{2}{11}^3 = \mathbf{0}$
- $s_{not\ spam} = \frac{4}{7} \times \frac{2}{16} \times \frac{1}{16}^3 > s_{spam}$

This email will be classified as not a spam simply because the word "you" has
never appeared in spam emails.

Solution to the problemo

Smoothing or discounting with **hyperparameter** α: we need to alter Step 1.4

Let $|V|$ be the size of the vocabulary

	buy	lottery	...	you	at	dinner	have	nice
Spam $P(word_i \mid spam)$	$\frac{1+\alpha}{11+\alpha|V|}$	$\frac{2+\alpha}{11+\alpha|V|}$	...	$\frac{0+\alpha}{11+\alpha|V|}$	$\frac{0+\alpha}{11+\alpha|V|}$	$\frac{0+\alpha}{11+\alpha|V|}$	$\frac{0+\alpha}{11+\alpha|V|}$	$\frac{0+\alpha}{11+\alpha|V|}$
Not spam $P(word_i \mid not\ spam)$	$\frac{0+\alpha}{16+\alpha|V|}$	$\frac{1+\alpha}{16+\alpha|V|}$	...	$\frac{2+\alpha}{16+\alpha|V|}$	$\frac{1+\alpha}{16+\alpha|V|}$	$\frac{4+\alpha}{16+\alpha|V|}$	$\frac{1+\alpha}{16+\alpha|V|}$	$\frac{1+\alpha}{16+\alpha|V|}$

Let $\alpha = 1$,

- $s_{spam} = \frac{3}{7} \times \frac{1}{25} \times \frac{3}{25}^3 = 0.0000296$
- $s_{not\ spam} = \frac{4}{7} \times \frac{3}{30} \times \frac{2}{30}^3 = 0.0000169 < s_{spam}$

Note: these are very small values due to the product of small values. Typically, we apply the logarithm function to avoid underflow as in MAP and MLE (cf. lecture 4).

Summary: Bayes' rule for multinomial naive Bayes classifier

- **Data**: categorical y, categorical x
- **Random variable**: discrete Y, discrete X

$$P(Y = y \mid X = x) = \frac{P(X = x \mid Y = y)P(Y = y)}{P(X = x)}$$

$$\hat{y}_{MAP} = \arg\max_y P(X = x \mid Y = y)P(Y = y)$$

Summary: multinomial naive Bayes classifier

- **Prediction** y: categorical data $y \in \{1, \cdots, C\}$
- **Variables** x_i, $i = 1, \cdots, n$: categorical data $x_i \in V$, where V is the vocabulary $V = \{w_1, \cdots, w_K\}$ given K unique categories
  - **Assumptions:**
    - x_i's are independent - **NAIVE!**
    - x_i follows a categorical distribution

  Note: here n is the size of the input data, e.g. the length of a document
- **Model** g:

$$\hat{y} = g(x_1, \cdots, x_n) = \underset{c \in \{1, \cdots, C\}}{\arg\max} P(c) \prod_{i=1}^{n} P(x_i \mid c)$$

  where $P(c)$ is the prior and $\prod_{i=1}^{n} P(x_i \mid c)$ is the likelihood under the assumptions
- **Hyperparameters** h: smoothing factor α
- **Parameters** θ: $P(c)$, $P(x_j \mid c)$ and V (if not given)

Summary: multinomial naive Bayes classifier (cont.)

- **Parameter estimation (training)**:
  Given the vocabulary $V = \{w_k\}_{k=1}^K$ and a training data set
  $\{(b_1, y_1), \cdots, (b_N, y_N)\}$, where each b_j contains a list of words.
  Let $N_c = count(y_j = c)$.
  - Likelihood $P(w_i \mid c)$ for each w_i:

  $$P(w_i \mid c) = \frac{count(\text{occurrences of } w_i \text{ in all } b_j \text{ for } y_j = c) + \alpha}{count(\text{all words from class } c) + \alpha K}$$

  - Prior $P(c)$:

  $$P(c) = \frac{N_c}{N}$$

Gaussian naive Bayes classifier

Example 2: real-time customer insight

An online shop is selling a new gaming computer

- **Prediction task** y: for a customer browsing this computer, the shop would like to predict if the customer will complete the transaction. If the prediction says no, the shop will perform certain actions, such as
  - proposing a discount to the customer
  - threatening the customer by showing an irritating message, e.g. "there are 20 people looking at this item right now"
  - offering a free item to encourage the transaction

Example 2: real-time customer insight

An online shop is selling a new gaming computer

- **Prediction task** y: for a customer browsing this computer, the shop would like to predict if the customer will complete the transaction. If the prediction says no, the shop will perform certain actions, such as
  - proposing a discount to the customer
  - threatening the customer by showing an irritating message, e.g. "there are 20 people looking at this item right now"
  - offering a free item to encourage the transaction
- **Input variables** x: the shop has the following information about the customers who are browsing this computer:
  - All kinds of personal information from different sources (Google, Facebook, via e.g. cookies, IP address, the version of your browser, etc)
  - In this example, they choose the following features as the input variables: **1)** average time they stay on Facebook everyday; **2)** how much money they spend on games (yes they have access to their steam account); **3)** daily active time on average (and yes they have access to their smart watch)

Example 2: real-time customer insight

An online shop is selling a new gaming computer

- **Prediction task** y: for a customer browsing this computer, the shop would like to predict if the customer will complete the transaction. If the prediction says no, the shop will perform certain actions, such as
  - proposing a discount to the customer
  - threatening the customer by showing an irritating message, e.g. "there are 20 people looking at this item right now"
  - offering a free item to encourage the transaction
- **Input variables** x: the shop has the following information about the customers who are browsing this computer:
  - All kinds of personal information from different sources (Google, Facebook, via e.g. cookies, IP address, the version of your browser, etc)
  - In this example, they choose the following features as the input variables: **1)** average time they stay on Facebook everyday; **2)** how much money they spend on games (yes they have access to their steam account); **3)** daily active time on average (and yes they have access to their smart watch)
- **Training data**:
  - The aforementioned personal information recorded from 1000 customers
  - If they have completed the transaction of purchasing the computer or not

Modeling for real-time customer insight

- **Prediction** y: complete transaction or drop out before paying
- **Variables** $\boldsymbol{x} = [x_1, x_2, x_3]$: $x_1 =$ duration (hour) on Facebook, $x_2 =$ money (k dollar) spent on games, $x_3 =$ active time (hour)
  - **Assumptions:**
    - x_1, x_2, x_3 are independent - **NAIVE!**
    - given data from class c, x_i is generated from a Gaussian distribution with PDF

$$f_i(x_i \mid c) = \frac{1}{\sqrt{2\pi\sigma_{c,i}^2}} e^{-\frac{(x_i - \mu_{c,i})^2}{2\sigma_{c,i}^2}}$$

- Model g: **Gaussian naive Bayes classifier**

$$\hat{y} = \underset{c \in \{complete,\ drop\ out\}}{\arg\max} \ P(c) \prod_{i=1}^{d} f_i(x_i \mid c)$$

where $P(c)$ is the prior and $\prod_{i=1}^{d} f_i(x_i \mid c)$ is the likelihood under the assumptions
Note: it is the maximum a posteriori estimation of the label (complete or drop out).
- **Parameters** θ: $P(c)$ and $\mu_{c,i}$, $\sigma_{c,i}$ in the likelihood $f_i(x_i \mid c)$ for all variable i and all classes c

CHALMERS | GÖTEBORGS UNIVERSITET

Parameter estimation (training)

In this demo, we only consider 5 customers in the training data for illustration purposes

- **Training data**: there are 5 customers:
    - $\boldsymbol{x}_1 = [x_1^1, x_2^1, x_3^1] = [2.44, 2.48, 2.64]$, $y_1 = 1$ drop out
    - $\boldsymbol{x}_2 = [x_1^2, x_2^2, x_3^2] = [9.77, 6.82, 0.55]$, $y_2 = 0$ complete
    - $\boldsymbol{x}_3 = [x_1^3, x_2^3, x_3^3] = [2.15, 8.05, 3.11]$, $y_3 = 1$ drop out
    - $\boldsymbol{x}_4 = [x_1^4, x_2^4, x_3^4] = [1.96, 3.78, 3.75]$, $y_4 = 1$ drop out
    - $\boldsymbol{x}_5 = [x_1^5, x_2^5, x_3^5] = [8.31, 7.93, 0.16]$, $y_5 = 0$ complete

Parameter estimation (training) (cont.)

- Step 1: Estimate $\mu_{c,i}$, $\sigma_{c,i}$ in the likelihood

$$f_i(x_i \mid c) = \frac{1}{\sqrt{2\pi\sigma_{c,i}^2}} e^{-\frac{(x_i - \mu_{c,i})^2}{2\sigma_{c,i}^2}}$$

for all variable i and $c \in \{complete, drop\ out\}$

- For each $i = 1, 2, 3$, collect all x_i^j for $y_j = drop\ out$. Compute the sample mean $\hat{\mu}_{drop\ out,i}$ and the sample standard deviation $\hat{\sigma}_{drop\ out,i}$.
- For each $i = 1, 2, 3$, collect all x_i^j for $y_j = complete$. Compute the sample mean $\hat{\mu}_{complete,i}$ and the sample standard deviation $\hat{\sigma}_{complete,i}$.

Parameter estimation (training) (cont.)

Estimated likelihood functions
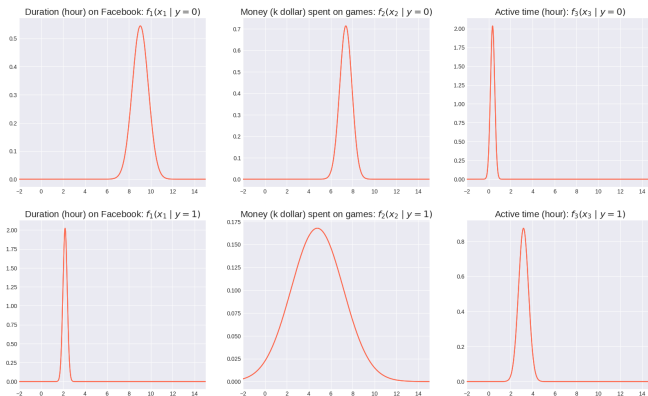
Parameter estimation (training) (cont.)

- Step 2: Estimate the **prior** $P(complete)$ **and** $P(drop\ out)$
  - Customers who have completed transaction $P(complete)$:

$$P(complete) = \frac{\#\ of\ complete}{\#\ of\ customers} = \frac{2}{5}$$

  - Customers who have dropped out before paying $P(drop\ out)$:

$$P(drop\ out) = \frac{\#\ of\ drop\ out}{\#\ of\ customers} = \frac{3}{5}$$

Classify a new customer

- Construct the Gaussian naive Bayes classifier
  The Gaussian naive Bayes classifier is a function of a given customer,
  i.e. $\mathbf{x} = [x_1, x_2, x_3]$. Let $s_{complete}$ and $s_{drop\ out}$ be the posterior
  without the normalization constant

$$s_{complete} = P(complete) \prod_{i=1}^{3} f_i(x_i \mid complete)$$

$$s_{drop\ out} = P(drop\ out) \prod_{i=1}^{3} f_i(x_i \mid drop\ out)$$

  - If $s_{complete} > s_{drop\ out}$: the customer will complete the transaction
  - If $s_{complete} \leq s_{drop\ out}$: the customer will drop out

CHALMERS | GÖTEBORGS UNIVERSITET

Classify a new customer (cont.)

- Compute the posterior of this customer
  - Complete: $P(complete \mid a\ customer) = \frac{s_{complete}}{s_{complete} + s_{drop\ out}}$
  - Drop out: $P(drop\ out \mid a\ customer) = \frac{s_{drop\ out}}{s_{complete} + s_{drop\ out}}$

  These are the probability of the customer completing the
  transaction and dropping out, respectively.

Classify a new customer (cont.)

For a new customer: hours spent on Facebook $x_1 = 2.51$; money spent on games $x_2 = 4.38$; active time $x_3 = 2.51$

- The likelihood of this customer completing a transaction:

- The likelihood of this customer dropping out before paying:

Classify a new customer (cont.)

- Compute the scores:

$$
\begin{aligned}
s_{complete} &= P(complete) \prod_{i=1}^{3} f_i(x_i \mid complete) \\
&= \frac{2}{5} f_1(2.51 \mid complete) f_2(4.38 \mid complete) f_3(2.51 \mid complete) \\
&\approx 0
\end{aligned}
$$

$$
\begin{aligned}
s_{drop\ out} &= P(drop\ out) \prod_{i=1}^{3} f_i(x_i \mid drop\ out) \\
&= \frac{3}{5} f_1(2.51 \mid drop\ out) f_2(4.38 \mid drop\ out) f_3(2.51 \mid drop\ out) \\
&= 0.016
\end{aligned}
$$

Classify a new customer (cont.)

- Compute the scores:

$$
\begin{aligned}
s_{complete} &= P(complete)\prod_{i=1}^{3} f_i(x_i \mid complete) \\
&= \frac{2}{5} f_1(2.51 \mid complete) f_2(4.38 \mid complete) f_3(2.51 \mid complete) \\
&\approx 0
\end{aligned}
$$

$$
\begin{aligned}
s_{drop\ out} &= P(drop\ out)\prod_{i=1}^{3} f_i(x_i \mid drop\ out) \\
&= \frac{3}{5} f_1(2.51 \mid drop\ out) f_2(4.38 \mid drop\ out) f_3(2.51 \mid drop\ out) \\
&= 0.016
\end{aligned}
$$

- $s_{complete} < s_{drop\ out}$: the customer will drop out before paying

Classify a new customer (cont.)

- Compute the scores:

$$
\begin{aligned}
s_{complete} &= P(complete) \prod_{i=1}^{3} f_i(x_i \mid complete) \\
&= \frac{2}{5} f_1(2.51 \mid complete) f_2(4.38 \mid complete) f_3(2.51 \mid complete) \\
&\approx 0
\end{aligned}
$$

$$
\begin{aligned}
s_{drop\ out} &= P(drop\ out) \prod_{i=1}^{3} f_i(x_i \mid drop\ out) \\
&= \frac{3}{5} f_1(2.51 \mid drop\ out) f_2(4.38 \mid drop\ out) f_3(2.51 \mid drop\ out) \\
&= 0.016
\end{aligned}
$$

- $s_{complete} < s_{drop\ out}$: the customer will drop out before paying
- Therefore, the online shop will send a message to threaten this customer.

Classify a new customer (cont.)

- Compute the scores:

$$
\begin{aligned}
s_{complete} &= P(complete) \prod_{i=1}^{3} f_i(x_i \mid complete) \\
&= \frac{2}{5} f_1(2.51 \mid complete) f_2(4.38 \mid complete) f_3(2.51 \mid complete) \\
&\approx 0
\end{aligned}
$$

$$
\begin{aligned}
s_{drop\ out} &= P(drop\ out) \prod_{i=1}^{3} f_i(x_i \mid drop\ out) \\
&= \frac{3}{5} f_1(2.51 \mid drop\ out) f_2(4.38 \mid drop\ out) f_3(2.51 \mid drop\ out) \\
&= 0.016
\end{aligned}
$$

- $s_{complete} < s_{drop\ out}$: the customer will drop out before paying
- Therefore, the online shop will send a message to threaten this customer.

$$\mathcal{THE}\ \ \mathcal{END}$$

Summary: Bayes' rule for Gaussian naive Bayes classifier

- **Data**: categorical y, continuous x
- **Random variable**: discrete Y, continuous X

$$P(Y = y \mid X = x) = \frac{f_{X|Y=y}(x \mid Y = y)P(Y = y)}{f_X(x)}$$

$$\hat{y}_{MAP} = \arg \max_y f_{X|Y=y}(x \mid Y = y)P(Y = y)$$

Summary: Gaussian naive Bayes classifier

- **Prediction** y: categorical data $y \in \{1, \cdots, C\}$
- **Variables** x_i, $i = 1, \cdots, d$: continuous numerical data $x_i \in \mathbb{R}$
  - **Assumption:**
    - x_i's are independent - **NAIVE!**
    - x_i follows a Gaussian distribution
- **Model** g:

$$
\begin{aligned}
\hat{y} &= g(x_1, \cdots, x_d) \\
&= \underset{c \in \{1, \cdots, C\}}{\arg\max} \, P(c) \prod_{i=1}^{d} f_i(x_i \mid y = c)
\end{aligned}
$$

where $P(c)$ is the prior and $\prod_{i=1}^{d} f_i(x_i \mid y = c)$ is the likelihood under the

assumptions with $f_i(x_i \mid y = c) = \frac{1}{\sqrt{2\pi\sigma_{c,i}^2}} e^{-\frac{(x_i - \mu_{c,i})^2}{2\sigma_{c,i}^2}}$

- **Parameters** θ: $P(c)$, $\mu_{c,i}$, $\sigma_{c,i}$ in $f_i(x_i \mid y = c)$ for all c and i

Summary: Gaussian naive Bayes classifier (cont.)

- **Parameter estimation (training)**:
  Given a training data set $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_N, y_N)\}$, where each
  $\boldsymbol{x}_j = [x_1^j, \cdots, x_d^j]$ is a vector containing all the features for one data point.
  Let $N_c = count(y_j = c)$.
  - $\mu_{c,i}$, $\sigma_{c,i}$ in the likelihood $f_i(x_i \mid y = c)$ for all variable i and all classes c:

  $$\hat{\mu}_{c,i} = \frac{1}{N_c} \sum_{t=1}^{N_c} x_i^t$$

  $$\hat{\sigma}_{c,i} = \sqrt{\frac{1}{N_c - 1} \sum_{t=1}^{N_c} (x_i^t - \hat{\mu}_{c,i})^2}$$

    for all $t \in$ class c
  - Prior $P(c)$:

  $$P(c) = \frac{N_c}{N}$$

Naive Bayes: pros and cons

- Pros:
  - Highly scalable
  - Simple
  - Interpretable
  - Easy to implement
  - Working fine for some use cases (e.g. spam filter)
- Cons:
  - Too simple for most use cases
  - Assumptions are too naive

A word on model complexity

- Models with high complexity:
  - Smart-ass models: they usually suffer from overfitting, i.e. working well on the training data set, but generalizing poorly on unseen data
  - Low bias (good)
  - High variance (bad)
  - Regularization is needed during training (cf. lecture 4 MLE vs MAP)
- Simple models:
  - They usually suffer less from overfitting, i.e. they might not work very well on the training data set; they are not performing much worse on unseen data
  - High bias (bad)
  - Low variance (good)

Today

Yinan Yu   Lecture 5: Classification and Naive Bayes classifier

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier

Not yet:

- Evaluation of parameter estimation

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier

Not yet:

- Evaluation of parameter estimation

Next:

- Interval estimation, confidence interval

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier

Not yet:

- Evaluation of parameter estimation

Next:

- Interval estimation, confidence interval

Before next lecture:

- Gaussian distribution, sample mean, CDF, quantiles

Only in this one lecture! Sorry!