UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Probabilistic Model:
Naïve Bayes Classification

Siti Aminah*, Theresia V. R.

CSGE603130: Kecerdasan Artifisial dan Sains Data Dasar

Semester Genap 2021/2022

Outline

- Recall to Bayes Theorem
- Intro to Probabilistic Classifiers
- Naïve Bayes Classification
- Cross Validation to Choose a Model
- Example

References

- Probability and Statistics for Computer Science, David Forsyth, 2018
- Principles of Data Science, Sinan Ozdemir, 2016
- Data Mining The Textbook, Aggarwal, 2015
- Slide Mata Kuliah Data Science & Analytics, Fasilkom UI Sem Genap 2020/2021

Intro to Probabilistic Classification

Probabilistic Classifiers

- Probabilistic classifiers construct a model that quantifies the relationship between the feature variables and the target (class) variable as a probability.

- There are many ways in which such a modeling can be performed. Two of the most popular models are:
  - Bayes Classifiers (generative model)
  - Logistic Regression (discriminative model)`

Probabilistic Classifiers

Bayes classifier:

- The Bayes rule is used to model the probability of each value of the target variable for a given set of feature variables.
- It is assumed that the data points within a class are generated from a specific probability distribution such as the Bernoulli distribution or the multinomial distribution.
- A *naive Bayes assumption* of class-conditioned feature independence is often (but not always) used to simplify the modeling.

Logistic regression:

- The target variable is assumed to be drawn from a Bernoulli distribution whose mean is defined by a parameterized logit function on the feature variables.
- Thus, the probability distribution of the *class* variable is a parameterized function of the feature variables.
- This is in contrast to the Bayes model that assumes a specific generative model of the *feature* distribution of each class.

Discriminative vs Generative Classifiers

Training classifiers involves estimating $f: X \rightarrow Y$, or $P(Y|X)$

Discriminative classifiers :

- Learn the boundary between classes
- Assume some functional form for $P(Y|X)$
- Estimate parameters of $P(Y|X)$ directly from training data

Generative classifiers

- Model the distribution of individual classes
- Assume some functional form for $P(X|Y), P(X)$
- Estimate parameters of $P(X|Y), P(X)$ directly from training data
- Use Bayes rule to calculate $P(Y|X= x_i)$

UNIVERSITAS
INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Recall to Bayes Theorem

Recall: Probability Basics

- We have two six-sided dice. When they are rolled, it could end up with the following occurrence: (A) dice 1 lands on side “3”, (B) dice 2 lands on side “5”, and (C) Two dice sum to eight. Answer the following questions:

$$P(A) = ?$$
$$P(B) = ?$$
$$P(C) = ?$$

$$P(A, B) = ?$$
$$P(A, C) = ?$$

$$P(A|B) = ?$$
$$P(C|A) = ?$$

Recall: Bayes Theorem

- Bayesian Inference
  - $P(A)$: The probability that event A occurs
  - $P(A|B)$: The probability that A occurs, given that B occurred
  - $P(A,B)$: The probability that A and B occurs

  - $P(A,B) = P(A) * P(B|A)$

- From $P(A,B) = P(A) * P(B|A)$
  - $P(B,A) = P(B) * P(A|B)$
  - $P(B) * P(A|B) = P(A) * P(B|A)$
  - $P(A|B) = \frac{P(A)*P(B|A)}{P(B)}$

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London,* **53:370-418**

Recall: Bayes Theorem

- A charitable organization solicits donations from individuals in the population of which 6/11 have age greater than 50. The company has a success rate of 6/11 in soliciting donations, and among the individuals who donate, the probability that the age is greater than 50 is 5/6. Given an individual with age greater than 50, what is the probability that he or she will donate?

Recall: Bayes Theorem

Kita definisikan:

A: Umur > 50 tahun

B: Individu memberikan donasi

Dari soal diketahui

$P(A)$ = 6/11

$P(B)$= 6/11

$P(A|B)$= 5/6

Ditanyakan $P(B|A)$?

Ditanyakan $P(B|A)$?

$$P(B|A) = \frac{P(B) * P(A|B)}{P(A)}$$

$$= \frac{\frac{6}{11} * \frac{5}{6}}{\frac{6}{11}}$$

$$= \frac{5}{6}$$

Recall: Bayes Theorem

- Let's try thinking about Bayes using the terms hypothesis and data.
  - y = your hypothesis about the given data
  - $\boldsymbol{x}$ = the data that you are given.

- Bayes can be interpreted as trying to figure out $P(y|\boldsymbol{x})$, the probability that our hypothesis is correct, given the data at hand.

- To use our terminology from before:

$$P(y|\boldsymbol{x}) = \frac{P(y) * P(\boldsymbol{x}|y)}{P(\boldsymbol{x})}$$

Recall: Bayes Theorem

- $P(y)$ is the probability of the hypothesis before we observe the data, called the ***prior probability*** or just ***prior***
- $P(y|\boldsymbol{x})$ is what we want to compute, the probability of the hypothesis after we observe the data, called the ***posterior***
- $P(\boldsymbol{x}|y)$ is the probability of the data under the given hypothesis, called the ***likelihood*** or ***class/target conditional density***
- $P(\boldsymbol{x})$ is the probability of the data under any hypothesis, called the ***normalizing constant***

$$P(y|\boldsymbol{x}) = \frac{P(y) * P(\boldsymbol{x}|y)}{P(\boldsymbol{x})}$$

Recall: Bayes Theorem

- This concept is not far off from the idea of machine learning and predictive analytics. In many cases, when considering predictive analytics, we use the given data to predict an outcome.

- Using the current terminology,
  - y (our hypothesis) can be considered our outcome
  - $P(y|\boldsymbol{x})$ is another way of saying: what is the chance that my hypothesis is correct, given the data in front of me?

Naïve Bayes Classification

Bayes Classification

- Jika dipenuhi kondisi berikut:
  - $p(y|\mathbf{x})$ untuk sebuah data diketahui.
  - Semua error pada klasifikasi sama pentingnya

  Maka aturan berikut menghasilkan error rate yang terkecil
  - Untuk sampel test $\mathbf{x}$, ambil kelas y dengan nilai $p(y|\mathbf{x})$ tertinggi.
  - Jika terdapat lebih dari satu kelas dengan nilai tertinggi, maka pilih secara acak dari suatu set kelas tersebut.

- Biasanya, nilai $p(y|\mathbf{x})$ tidak diketahui.
  - Jika terdapat **likelihood** atau **class conditional probability** $p(\mathbf{x}|y)$, dan **prior** $p(y)$, dapat digunakan aturan Bayes untuk menghasilkan **posterior:**

$$p(y|\mathbf{x}) = \frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}$$

Bayes Classification

Bayes classification:

$$p(y|\mathbf{x}) \propto p(\mathbf{x}|y)p(y) = p\left(x_i, \dots, x_j \middle| y\right)p(y)$$

- Kesulitan dengan Bayes classification adalah bagaimana mempelajari joint probability
- Solusi: Naïve Bayes Classification

Classifying with Naive Bayes

- Mengasumsikan bahwa features pada suatu kelas bersifat conditionally independent:

$$p(\mathbf{x}|y) = \prod_j p(x_i|y)$$

Dengan menggunakan asumsi tersebut, didapatkan:

$$p(y|\mathbf{x}) = \frac{p(\mathbf{X}|y)p(y)}{p(\mathbf{X})} = \frac{\left(\prod_j p(x_i|y)\right)p(y)}{p(\mathbf{X})} \propto \left(\prod_j p(x_i|y)\right)p(y)$$

Classifying with Naive Bayes

- Untuk membuat keputusan, kita perlu memilih kelas dengan nilai $p(y|\mathbf{x})$ tertinggi ➔ menerapkan Maximum a Posterior (MAP) rule.
- Artinya kita hanya perlu mengetahui nilai posterior pada $\mathbf{x}$, tanpa menghitung estimasi $p(\mathbf{x})$, sehingga diperoleh aturan

$$\text{pilih } y \text{ dimana } \left[\left(\prod_j p(x_i|y)\right)p(y)\right] \text{ adalah tertinggi}$$

- Dalam implementasi, bisa timbul masalah dari aturan ini: perkalian probabilitas (floating point) dalam jumlah besar, hasilnya bisa dianggap nol.

  ➔ perlu ditambahkan log probabilities.

- Fungsi logaritma memiliki properti monotomic:

  $a > b$ sama dengan $\log \mathrm{a} > \log \mathrm{b}$, sehingga aturan dapat disesuaikan menjadi:

$$\text{pilih } y \text{ dimana } \left[\left(\sum_j \log p(x_i|y)\right) + \log p(y)\right] \text{ adalah tertinggi}$$

Classifying with Naive Bayes

- Langkah-langkah Naïve Bayes perlu $p(y)$ dan $p(x_i|y)$ untuk setiap *i*.
  - – untuk $p(y)$ dapat dihitung dengan menghitung jumlah sampel training masing-masing kelas, dibagi dengan jumlah seluruh kelas.
  - – untuk $p(x_i|y)$ dapat menggunakan simple parametric models.
    - penggunaan distribusi normal pada x_iu ntuk setiap kemungkinan nilai y, dengan parameter yang dipilih berdasarkan maximum likelihood.
    - distribusi Poisson ataupun Bernoulli.
    - multinominal model untuk variabel diskret.

Contoh klasifikasi dengan Naïve Bayes

- Kita ingin menentukan $P(C|X)$ yaitu probabilitas bahwa record $X =< x_1, x_2, \dots x_k >$ adalah kelas C.

- Jika, $X = \langle \text{hujan, panas, tinggi, ringan} \rangle$ apakah akan bermain tenis?

Hari	Cuaca	Suhu	Kelembapan	Angin	Main Tenis?
1	Cerah	Panas	Tinggi	Ringan	Tidak
2	Cerah	Panas	Tinggi	Kencang	Tidak
3	Mendung	Panas	Tinggi	Ringan	Ya
4	Hujan	Sedang	Tinggi	Ringan	Ya
5	Hujan	Dingin	Normal	Ringan	Ya
6	Hujan	Dingin	Normal	Kencang	Tidak
7	Mendung	Dingin	Normal	Kencang	Ya
8	Cerah	Sedang	Tinggi	Ringan	Tidak
9	Cerang	Dingin	Normal	Ringan	Ya
10	Hujan	Sedang	Normal	Ringan	Ya
11	Cerah	Sedang	Normal	Kencang	Ya
12	Mendung	Sedang	Tinggi	Kencang	Ya
13	Mendung	Panas	Normal	Ringan	Ya
14	Hujan	Sedang	Tinggi	Kencang	Tidak

Contoh klasifikasi dengan Naïve Bayes

- Problem bisa diformulasikan dengan posterior probability

- $P(C|X)$ = probabilitas bahwa record $X =< x_1, x_2, \ldots x_k >$ adalah kelas C

  Contoh: $\mathrm{P(kelas{=}Ya|cuaca{=}cerah, angin{=}ringan,\ldots)}$
- Tentukan label kelas C untuk sampel X, yang mana $P(C|X)$ adalah maksimal.

- Teorema Bayes:
  - $P(X)$ konstan untuk semua kelas
  - $P(C)$ adalah frekuensi relatif terhadap sampel kelas C
  - C sedemikian hingga $P(C|X)$maksimum = C sedemikian hingga$P(X|C).P(C)$ maksimum
- Problem: menentukan $P(X|C) = P(x_1, \ldots, x_k|C)$ tidak feasible

Naïve Bayes Classifier

- Asumsi Naïve: independensi setiap atribut

$$P(x_1, \dots, x_k | C) = P(x_1|C) \cdot P(x_2|C) \cdot \ \dots \ \cdot P(x_k|C)$$

- Jika atribut *i* adalah kategorikal:
  - $P(x_i|C)$ diestimasi sebagai relative frequency dari sampel yang memiliki nilai x_i sebagai atribut ke-*i* di dalam kelas *C*

- Jika atribut ke-*i* adalah kontinu:
  - $P(x_i|C)$ diestimasi melalui Gaussian density function

Naïve Bayes Classifier

- Asumsi bahwa atribut adalah conditionally independent berpengaruh besar pada pengurangan waktu komputasi.
- Berdasarkan training set sebelumnya, kita dapat menghitung probabilitas berikut.

P(cerah|Main Tenis)

Cuaca	Ya	Tidak
Cerah	2/9	3/5
Mendung	4/9	0
Hujan	3/9	2/5

Play Tennis Example

- Estimasi $P(C)$

Hari	Cuaca	Suhu	Kelembapan	Angin	Main Tenis?
1	Cerah	Panas	Tinggi	Ringan	Tidak
2	Cerah	Panas	Tinggi	Kencang	Tidak
3	Mendung	Panas	Tinggi	Ringan	Ya
4	Hujan	Sedang	Tinggi	Ringan	Ya
5	Hujan	Dingin	Normal	Ringan	Ya
6	Hujan	Dingin	Normal	Kencang	Tidak
7	Mendung	Dingin	Normal	Kencang	Ya
8	Cerah	Sedang	Tinggi	Ringan	Tidak
9	Cerang	Dingin	Normal	Ringan	Ya
10	Hujan	Sedang	Normal	Ringan	Ya
11	Cerah	Sedang	Normal	Kencang	Ya
12	Mendung	Sedang	Tinggi	Kencang	Ya
13	Mendung	Panas	Normal	Ringan	Ya
14	Hujan	Sedang	Tinggi	Kencang	Tidak

P(Ya) = 9/14
P(Tidak) = 5/14

Play Tennis Example

- Estimasi $P(x_i|C)$

Hari	Cuaca	Suhu	Kelembapan	Angin	Main Tenis?
1	Cerah	Panas	Tinggi	Ringan	Tidak
2	Cerah	Panas	Tinggi	Kencang	Tidak
3	Mendung	Panas	Tinggi	Ringan	Ya
4	Hujan	Sedang	Tinggi	Ringan	Ya
5	Hujan	Dingin	Normal	Ringan	Ya
6	Hujan	Dingin	Normal	Kencang	Tidak
7	Mendung	Dingin	Normal	Kencang	Ya
8	Cerah	Sedang	Tinggi	Ringan	Tidak
9	Cerang	Dingin	Normal	Ringan	Ya
10	Hujan	Sedang	Normal	Ringan	Ya
11	Cerah	Sedang	Normal	Kencang	Ya
12	Mendung	Sedang	Tinggi	Kencang	Ya
13	Mendung	Panas	Normal	Ringan	Ya
14	Hujan	Sedang	Tinggi	Kencang	Tidak

Cuaca	
P(Cerah|Ya)=2/9	P(Cerah|Tidak)=3/5
P(Mendung|Ya)=4/9	P(Mendung|Tidak)=0
P(Hujan|Ya)=3/9	P(Hujan|Tidak)=2/5
Suhu	
P(Panas|Ya)=2/9	P(Panas|Tidak)=2/5
P(Sedang|Ya)=4/9	P(Sedang|Tidak)=2/5
P(Dingin|Ya)=3/9	P(Dingin|Tidak)=1/5
Kelembapan	
P(Tinggi|Ya)=3/9	P(Tinggi|Tidak)=4/5
P(Normal|Ya)=6/9	P(Normal|Tidak)=1/5
Angin	
P(Kencang|Ya)=3/9	P(Kencang|Tidak)=3/5
P(Ringan|Ya)=6/9	P(Ringan|Tidak)=2/5

Play Tennis Example

- Klasifikasikan X
- Terdapat sampel yang belum terlihat sebelumnya
- X = <hujan, panas, tinggi, ringan>

$$\begin{aligned} P(\text{Ya} \mid X) &\propto P(X \mid \text{Ya}) \cdot P(\text{Ya}) \\ &= P(\text{Hujan} \mid \text{Ya}) \cdot P(\text{Panas} \mid \text{Ya}) \cdot P(\text{Tinggi} \mid \text{Ya}) \cdot P(\text{Ringan} \mid \text{Ya}) \cdot P(\text{Ya}) \\ &= {}^{3}/_{9} \cdot {}^{2}/_{9} \cdot {}^{3}/_{9} \cdot {}^{6}/_{9} \cdot {}^{9}/_{14} \\ &= 0.010582 \end{aligned}$$

$$\begin{aligned} P(\text{Tidak} \mid X) &\propto P(X \mid \text{Tidak}) \cdot P(\text{Tidak}) \\ &= P(\text{Hujan} \mid \text{Tidak}) \cdot P(\text{Panas} \mid \text{Tidak}) \cdot P(\text{Tinggi} \mid \text{Tidak}) \cdot P(\text{Ringan} \mid \text{Tidak}) \cdot P(\text{Tidak}) \\ &= {}^{2}/_{5} \cdot {}^{2}/_{5} \cdot {}^{4}/_{5} \cdot {}^{2}/_{5} \cdot {}^{5}/_{14} \\ &= 0.018286 > 0.010582 \end{aligned}$$

- Sampel X diklasifikasikan sebagai kelas Tidak Bermain Tenis

The “zero-frequency problem”

- Bagaimana jika, nilai atribut tidak terjadi pada setiap kelas?
  - Contoh: Kelembapan = Tinggi untuk kelas Bermain Tenis
    - $\mathrm{P(Kelembapan{=}Tinggi} \mid \mathrm{Main\ Tenis} = \mathrm{Ya}) = 0$
  - Posteriori probabilities juga akan menghasilkan nilai nol. (Tidak peduli seberapa besar kemungkinan nilai lainnya)
    - $\mathrm{P(Main\ Tenis} \mid \langle \ldots, \mathrm{kelembapan} = \mathrm{tinggi} \rangle) = 0$

- Solusi: menambahkan nilai 1 untuk perhitungan setiap kombinasi nilai atribut dan kelas. (Laplace estimator)
- Hasil: probabilitas tidak akan nol (juga menstabilisasikan estimasi probabilitas)

Modified Probability Estimates

- Pada beberapa kasus, menambahkan nilai konstan selain 1 merupakan kemungkinan yang lebih tepat.
- Contoh: atribut Cuaca untuk kelas Ya (Bermain Tenis)

$$\frac{2+\mu/3}{9+\mu} \qquad \frac{4+\mu/3}{9+\mu} \qquad \frac{3+\mu/3}{9+\mu}$$

Cerah **Mendung** **Hujan**

- Bobot tidak perlu sama, namun sum bobot adalah 1

$$\frac{2+\mu p_1}{9+\mu} \qquad \frac{4+\mu p_2}{9+\mu} \qquad \frac{3+\mu p_3}{9+\mu}$$

Missing Values

- Training: kejadian tidak dimasukkan di dalam perhitungan frekuensi untuk kombinasi nilai atribut dan kelas.
- Klasifikasi: atribut akan diabaikan dalam perhitungan

Cuaca	
P(Cerah|Ya)=2/9	P(Cerah|Tidak)=3/5
P(Mendung|Ya)=4/9	P(Mendung|Tidak)=0
P(Hujan|Ya)=3/9	P(Hujan|Tidak)=2/5
Suhu	
P(Panas|Ya)=2/9	P(Panas|Tidak)=2/5
P(Sedang|Ya)=4/9	P(Sedang|Tidak)=2/5
P(Dingin|Ya)=3/9	P(Dingin|Tidak)=1/5
Kelembapan	
P(Tinggi|Ya)=3/9	P(Tinggi|Tidak)=4/5
P(Normal|Ya)=6/9	P(Normal|Tidak)=1/5
Angin	
P(Kencang|Ya)=3/9	P(Kencang|Tidak)=3/5
P(Ringan|Ya)=6/9	P(Ringan|Tidak)=2/5

- Contoh:

Cuaca	Suhu	Kelembapan	Angin	MainTenis?
?	Dingin	Tinggi	Kencang	?

$$P\,(\text{“Ya”}|X) \propto 3/9 \,.\, 3/9 \,.\, 3/9 \,.\, 9/14 = 0.0238$$

$$P\,(\text{“Tidak”}|X) \propto 1/5 \,.\, 4/5 \,.\, 3/5 \,.\, 5/14 = 0.0343$$

Jadi X diklasifikasikan sebagai tidak bermain tenis.

Numeric Attributes

- Asumsi pada umumnya, atribut memiliki distribusi probabilitas Normal atau Gaussian
- Probability density function untuk distribusi normal didefinisikan dengan 2 parameter:
  - Sample mean μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} \mathrm{x}_i$$

  - Standard deviation σ

$$\sigma = \frac{1}{n-1}\sum_{i=1}^{n} (\mathrm{x}_i - \mu)^2$$

  - Density function $f(x)$ adalah

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Probability Densities

- Relationship antara probability dan density adalah:

$$\Pr\left[c - \frac{\varepsilon}{2} < x < c + \frac{\varepsilon}{2}\right] \approx \varepsilon \times f(c)$$

- Tetapi hal ini tidak akan mempengaruhi perhitungan posteriori probabilities karena ε dihapuskan
- Sehingga, exact relationship adalah:

$$\Pr[a \leq x \leq b] = \int_a^b f(t)dt$$

Statistics for Weather Data

Cuaca			Suhu		Kelembapan		Angin			Bermain	
	Ya	Tidak	Ya	Tidak	Ya	Tidak		Ya	Tidak	Ya	Tidak
Cerah	2	3	64, 68,	65, 71,	65, 70,	70, 85,	False	6	2	9	5
Mendung	4	0	69, 70,	72, 80,	70, 75,	90, 91,	True	3	3		
Hujan	3	2	72, …	85, …	80, …	95, …					
Cerah	2/9	3/5	$\mu = 73$	$\mu = 75$	$\mu = 79$	$\mu = 86$	False	6/9	2/5	9/14	5/14
Mendung	4/9	0/5	$\sigma = 6.2$	$\sigma = 7.9$	$\sigma = 10.2$	$\sigma = 9.7$	True	3/9	3/5		
Hujan	3/9	2/5									

- Contoh nilai density:

$$P(suhu = 66|Ya) \propto f_{73,6.2}(66) = \frac{1}{6.2\sqrt{2\pi}} e^{-\frac{(66-73)^2}{2\times 6.2^2}} = 0.0340$$

Classifying a New Day

- Hari baru X:

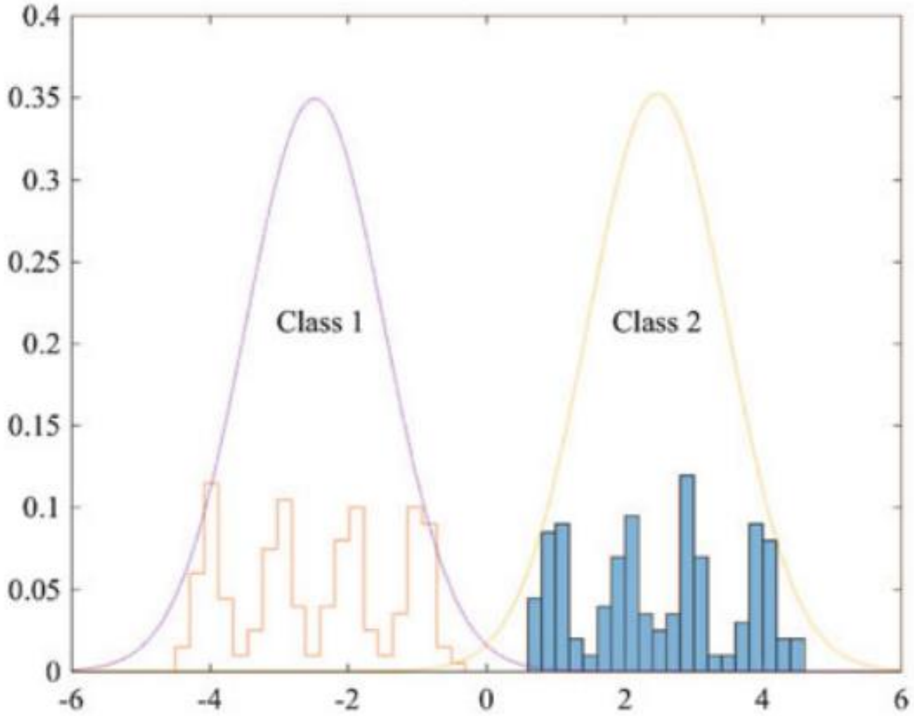
Cuaca	Suhu	Kelembapan	Angin	MainTenis?
Cerah	66	90	True	?

$$\begin{aligned} P(\text{"Ya"}|X) &\propto {}^{2}/_{9}\,.\,(0.0340).\,(0.0221).\,{}^{3}/_{9}.{}^{9}/_{14} = 0.000036 \\ P(\text{"Tidak"}|X) &\propto {}^{3}/_{5}\,.\,(0.0291).\,(0.0380).\,{}^{3}/_{5}.{}^{5}/_{14} = 0.000136 \end{aligned}$$

Jadi X diklasifikasikan sebagai tidak bermain tenis

Classifying with Naive Bayes

- Naïve Bayes classifier yang tidak memiliki model yang fit untuk tiap fitur dapat mengklasifikasikan data secara baik.
  - Ini karena klasifikasi tidak memerlukan model yang bagus untuk $p(\mathbf{x}|y)$ maupun $p(y|\mathbf{x})$.
  - Yang perlu diperhatikan adalah, skor tertinggi pada kelas yang benar dibandingkan kelas lainnya, untuk setiap $\mathbf{x}$.



- Figur ini menunjukkan class conditional histogram dari fitur x untuk 2 kelas yang berbeda.
- Terlihat normal model (superimposed) tidak menunjukkan histogram yang baik.
- Namun, naïve bayes classifier mampu mengklasifikasikan 2 kelas dengan baik

UNIVERSITAS
INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

Cross Validation to Choose a Model

Cross-Validation to Choose a Model

- Pada Naïve Bayes, kita dapat memilih model yang paling sesuai untuk $p(x_i|y)$, seperti model normal atau model Poisson.
- Untuk mengestimasi model terbaik, dapat digunakan cross-validation.
- Pilih *M* jumlah tipe model yang memungkinkan (contoh, dengan melakukan analisis terhadap histogram dari komponen fitur berdasarkan kelas)
- Hitung cross-validated error yang diperoleh untuk masing-masing fold, dan hitung average error.
- Perhatikan bahwa proses fit model untuk setiap fold memiliki nilai parameter yang berbeda, karena kemungkinan data training yang sedikit berbeda.

Cross-Validation to Choose a Model

- Setelah menentukan tipe model, terdapat masalah yang perlu diperhatikan.
  - Nilai parameter dari tipe model yang dipilih tidak diketahui.
  - Tidak adanya estimasi terkait seberapa baik model bekerja.
- Masalah tersebut dapat diatasi jika terdapat dataset yang cukup besar, yaitu dengan memisahkan labelled dataset ke dalam 2 bagian, training set untuk melatih model dan test set untuk evaluasi model final.
- Gunakan hasil cross-validated error untuk memilih tipe model.
- Gunakan keseluruhan training set untuk mengestimasi parameter yang digunakan pada model yang bersangkutan.
- Kemungkinan hasil estimasi akan lebih baik, karena menggunakan sedikit lebih banyak data.
- Terakhir, evaluasi model tersebut terhadap test set.

Cross-Validation to Choose a Model

- Prosedur ini menunjukkan beberapa keuntungan, antara lain:
  - Estimasi dari seberapa baik suatu tipe model tertentu menjadi unbiased, karena dievaluasi berdasarkan data yang tidak muncul pada saat training.
  - Ketika suatu model telah dipilih, estimasi parameter menjadi lebih baik karena menggunakan keseluruhan training set.
  - Estimasi evaluasi performa model yang dipilih juga menjadi unbiased, karena diperoleh berdasarkan data yang tidak muncul saat training maupun saat memilih model.

Variasi Naïve Bayes Classifiers

Reference: https://scikit-learn.org/stable/modules/naive_bayes.html

Variasi Naïve Bayes

- Implementasi Naïve Bayes Classifiers berbeda-beda, tergantung dari asumsi distribusi $P(\boldsymbol{x}|y)$
- Beberapa variasi Naïve Bayes yang telah diimplementasikan pada sk-learn:
  - Gaussian Naïve Bayes
  - Multinomial Naïve Bayes
  - Complement Naïve Bayes
  - Bernoulli Naïve Bayes
  - Categorical Naïve Bayes

Gaussian Naïve Bayes

- 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 *σy* and *μy* are estimated using maximum likelihood.

Multinomial Naïve Bayes (MNB)

- Implements the naive Bayes algorithm for multinomially distributed data,
- one of the two classic naive Bayes variants used in text classification
- The distribution is parametrized by vectors $\theta_y=(\theta_{y1},\dots,\theta_{yn})$ for each class *y*, where *n* is the number of features (in text classification, the size of the vocabulary) and θ_{yi} is the probability $P(x_i|y)$ of feature i appearing in a sample belonging to class *y*.
- The parameters θ_y is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

- Where $N_{yi} = \sum_{x \in T} x_i$ is the number of times feature *i* appears in a sample of class *y* in the training set T, and $N_y = \sum_{i=1}^{n} N_{yi}$ is the total count of all features for class y.
- α is smoothing to prevent zero probabilities

Complement Naïve Bayes (CNB)

- CNB is an adaptation of the standard multinomial naive Bayes (MNB) algorithm that is particularly suited for imbalanced data sets.
- CNB uses statistics from the *complement* of each class to compute the model’s weights.

Bernoulli Naïve Bayes

- Implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions; i.e., there may be multiple features but each one is assumed to be a binary-valued variable.
- The decision rule for Bernoulli naive Bayes is based on

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

- which differs from multinomial NB's rule in that it explicitly penalizes the non-occurrence of a feature *i* that is an indicator for class *y*, where the multinomial variant would simply ignore a non-occurring feature.

Categorical Naïve Bayes

- Designed for categorically distributed data.
- It assumes that each feature, which is described by the index *i*, has its own categorical distribution.
- For each feature *i* in the training set *X*, the classifier estimates a categorical distribution for each feature *i* of *X* conditioned on the class *y*. The index set of the samples is defined as *J* = {1,…,*m*}, with *m* as the number of samples
- The probability of category t in feature i given class c is estimated as:

$$P(x_i = t \mid y = c \,;\, \alpha) = \frac{N_{tic} + \alpha}{N_c + \alpha n_i},$$

- where $N_{tic} = |\{j \in J \mid x_{ij} = t, y_j = c\}|$ is the number of times category *t* appears in the samples x_i, which belong to class *c*, $N_c = |\{j \in J \mid y_j = c\}|$ is the number of samples with class *c*, α is a smoothing parameter and n_i is the number of available categories of feature *i*.

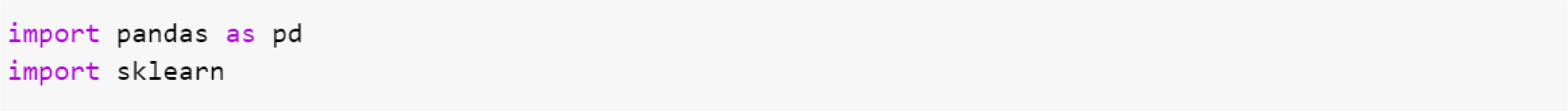
UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

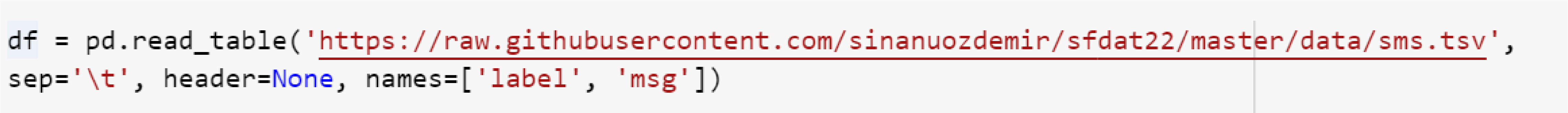
Example with Python Code

Spam Text Classification

- Data set with *n* features, (x_1, x_2, …, x_n) and a class label *C*. Let's take some data involving spam text classification.
- Our features would be words and phrases that are contained within the text samples and our class labels are simply spam or not spam (ham).

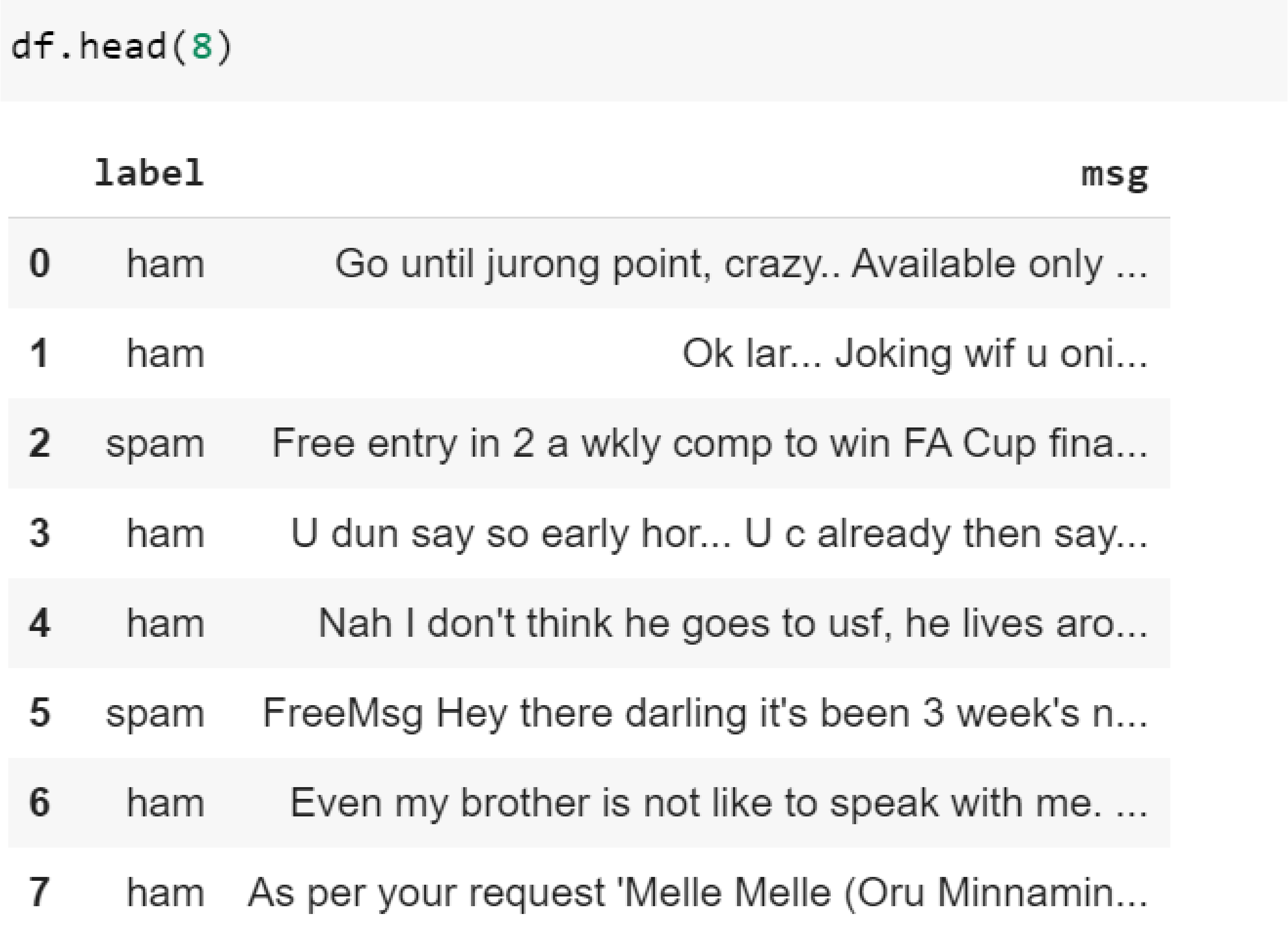


```
import pandas as pd
import sklearn
```



```
df = pd.read_table('https://raw.githubusercontent.com/sinanuozdemir/sfdat22/master/data/sms.tsv',
sep='\t', header=None, names=['label', 'msg'])
```

Spam Text Classification

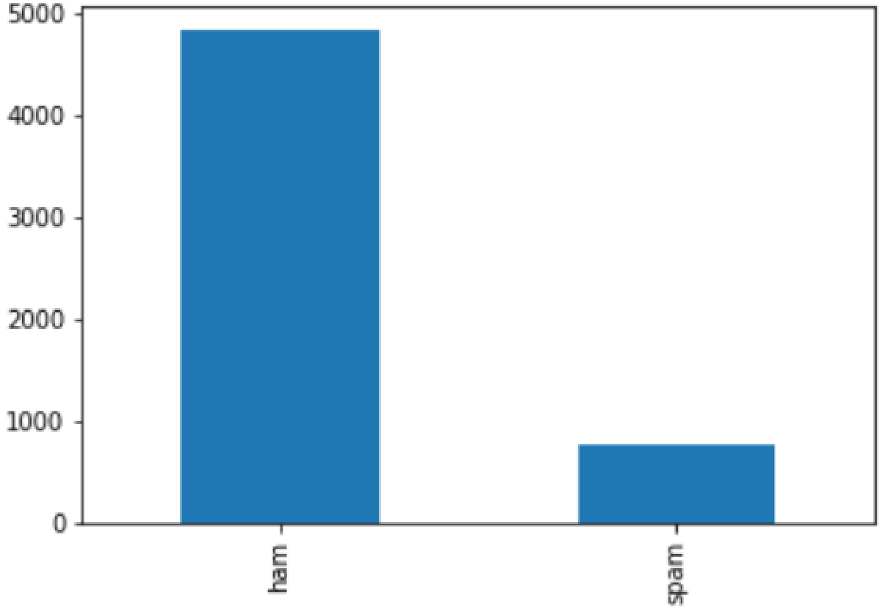


```python
df.head(8)
```

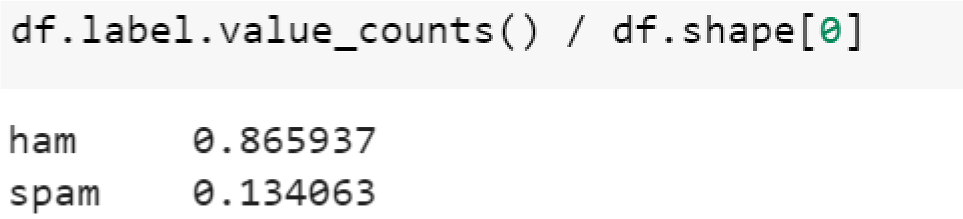
	label	msg
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
5	spam	FreeMsg Hey there darling it's been 3 week's n...
6	ham	Even my brother is not like to speak with me. ...
7	ham	As per your request 'Melle Melle (Oru Minnamin...

Spam Text Classification

```
df.label.value_counts().plot(kind="bar")
```



We have WAY more ham messages than we do spam. Because this is a classification problem, it will be very useful to know our *null accuracy rate* which is the percentage chance of predicting a single row correctly if we keep guessing the most common class, *ham*:



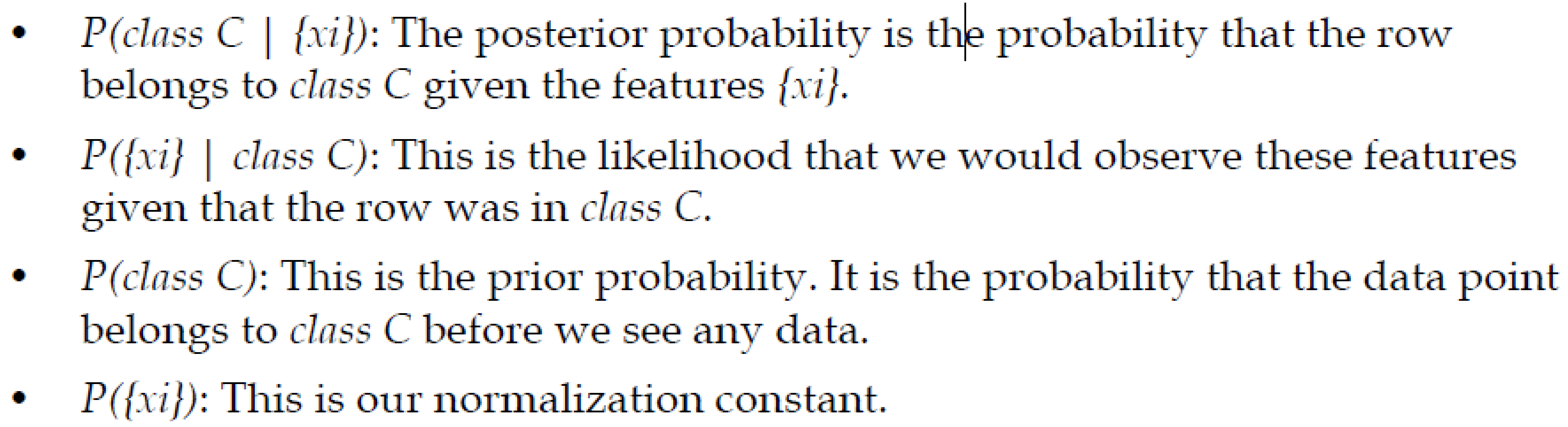
```
df.label.value_counts() / df.shape[0]
```

```
ham     0.865937
spam    0.134063
```

Spam Text Classification

- So if we blindly guessed ham we would be correct about 87% of the time, but we can do better than that. If we have a set of classes, *C*, and a features *xi*, then we can use Bayes theorem to predict the probability that a single row belongs to class *C:*

$$P\left(class\,C|\{x_i\}\right)=\frac{P\left(\{x_i\}\,|\,class\,C\right)\cdot P\left(class\,C\right)}{P\left(\{x_i\}\right)}$$



- *P(class C | {xi})*: The posterior probability is the probability that the row belongs to *class* C given the features *{xi}*.
- *P({xi} | class C)*: This is the likelihood that we would observe these features given that the row was in *class* C.
- *P(class C)*: This is the prior probability. It is the probability that the data point belongs to *class* C before we see any data.
- *P({xi})*: This is our normalization constant.

Spam Text Classification

- F $P(spam \mid send\ cash\ now) = P(send\ cash\ now \mid spam) * P(spam) / P(send\ cash\ now)$ Jsing Naïve Bayes to cl

  $$P(ham \mid send\ cash\ now) = P(send\ cash\ now \mid ham) * P(ham) / P(send\ cash\ now)$$

- We are concerned with the difference of these two numbers. We can use the following criteria to classify any single text sample:

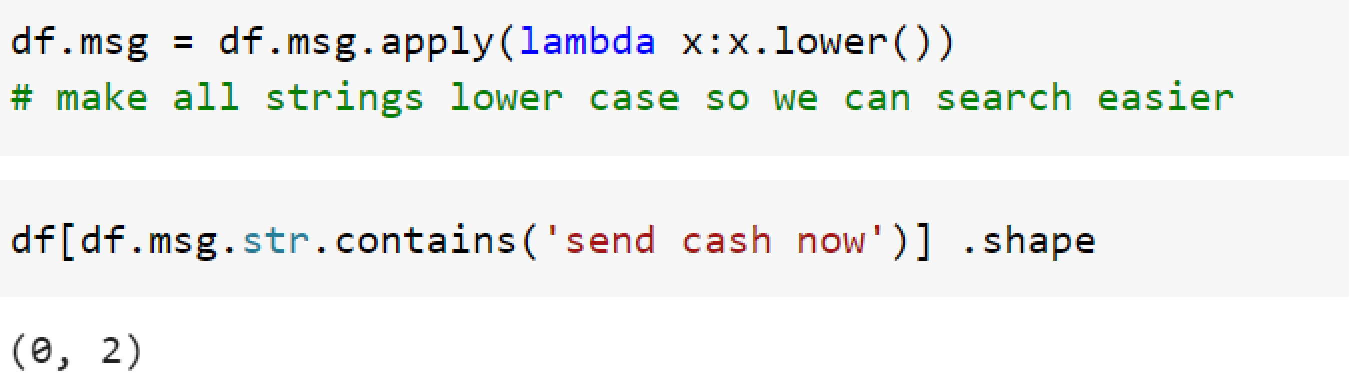
  - If `P(spam | send cash now)` > `P(ham | send cash now)`, then we will classify the text as spam
  - If `P(ham | send cash now)` > `P(spam | send cash now)`, then we will label the text as ham

Let’s figure out the numbers in this equation:

- $P(spam)$ = 0.134063
- $P(ham)$ = 0.865937
- $P(send\ cash\ now \mid spam)$
- $P(send\ cash\ now \mid ham)$

Spam Text Classification

- Let’s count the numbers of spam messages that include the phrase ‘send cash now’ and divi



```
df.msg = df.msg.apply(lambda x:x.lower())
# make all strings lower case so we can search easier
```

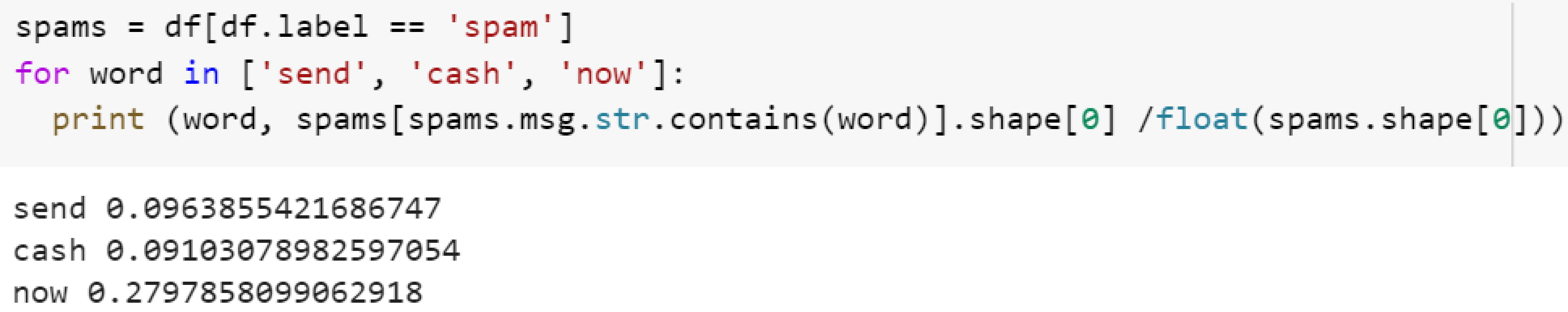
```
df[df.msg.str.contains('send cash now')] .shape
```

```
(0, 2)
```

- Oh No! There are literally 0 texts with the exact phrase ‘send cash now’.
- We can make a *naïve assumption* in our Bayes theorem, assume that the features (words) are conditionally independent.

$$P(send\,cash\,now \mid spam) = P(send \mid spam) * P(cash \mid spam) * P(now \mid spam)$$

Spam Text Classification



```python
spams = df[df.label == 'spam']
for word in ['send', 'cash', 'now']:
  print (word, spams[spams.msg.str.contains(word)].shape[0] /float(spams.shape[0]))
```

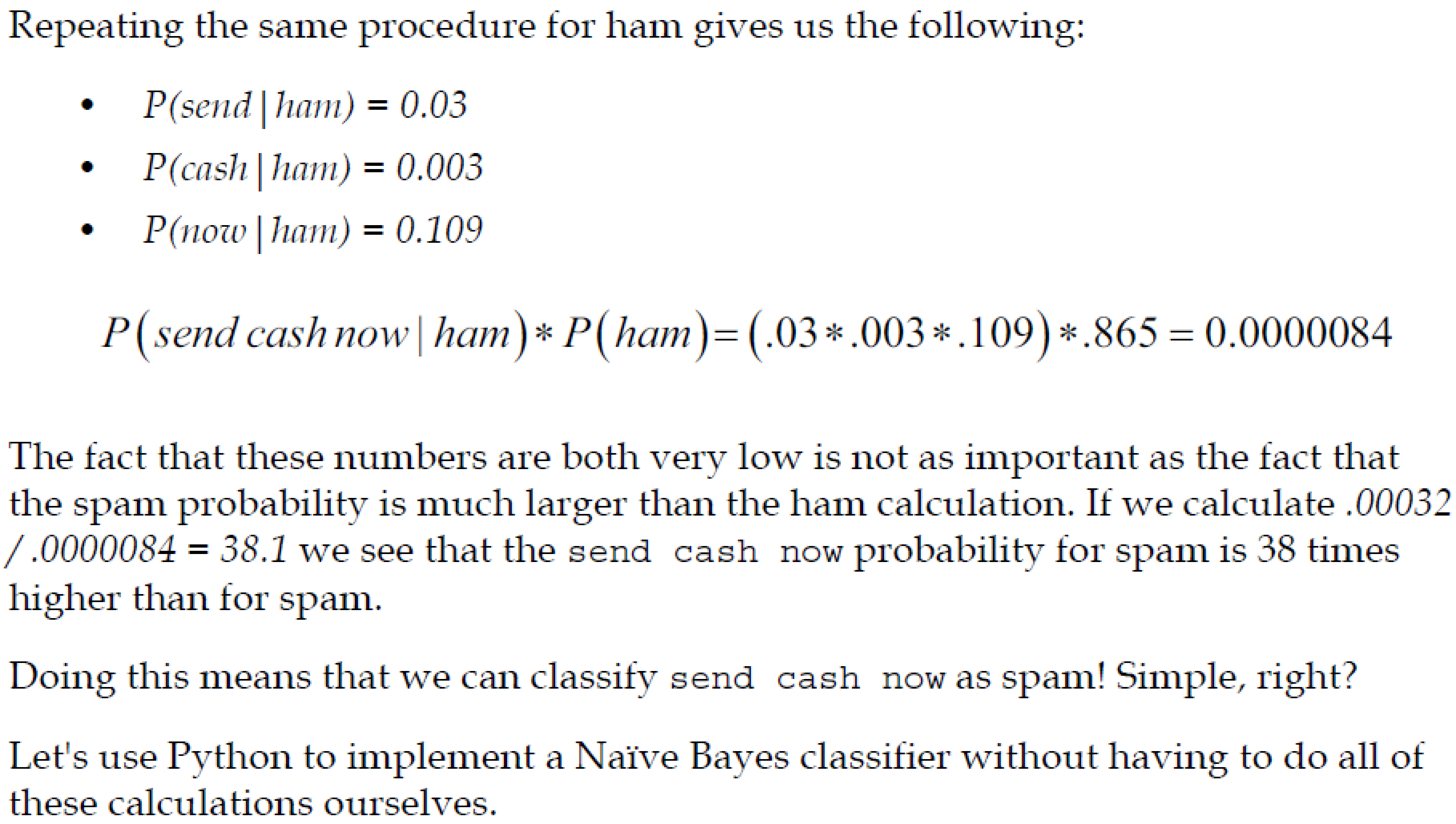
```
send 0.0963855421686747
cash 0.09103078982597054
now 0.2797858099062918
```

- $P(send \mid spam)$ = 0.096
- $P(cash \mid spam)$ = 0.091
- $P(now \mid spam)$ = 0.280

Meaning we can calculate the following:

$$P(send\, cash\, now \mid spam) * P(spam) = (.096 * .091 * .280) * .134 = 0.00032$$

Spam Text Classification



Repeating the same procedure for ham gives us the following:

- *P(send | ham)* = *0.03*
- *P(cash | ham)* = *0.003*
- *P(now | ham)* = *0.109*

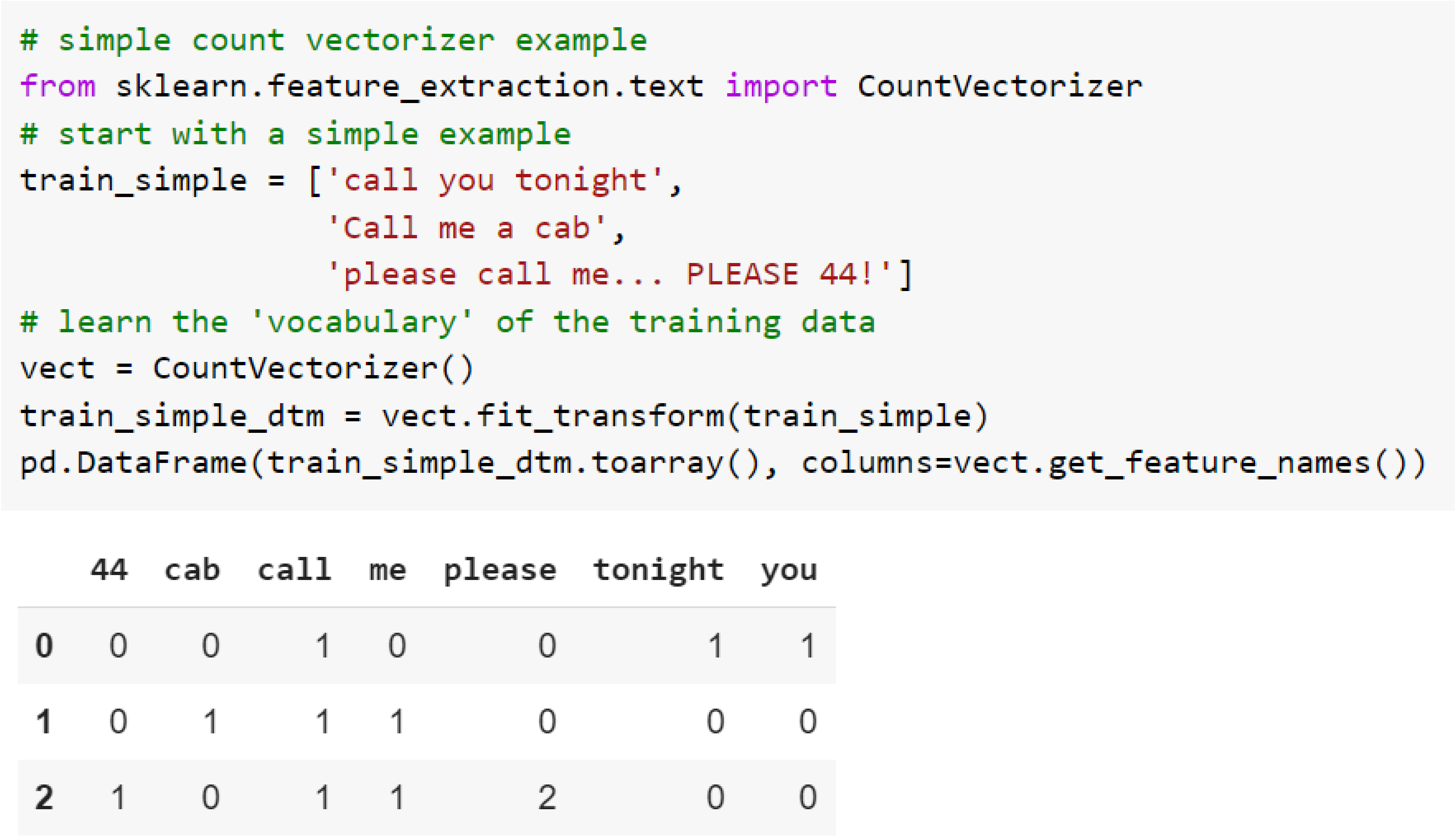
$$P\left(send\;cash\;now \mid ham\right) * P\left(ham\right) = \left(.03 * .003 * .109\right) * .865 = 0.0000084$$

The fact that these numbers are both very low is not as important as the fact that the spam probability is much larger than the ham calculation. If we calculate *.00032 / .0000084* = *38.1* we see that the `send cash now` probability for spam is 38 times higher than for spam.

Doing this means that we can classify `send cash now` as spam! Simple, right?

Let's use Python to implement a Naïve Bayes classifier without having to do all of these calculations ourselves.

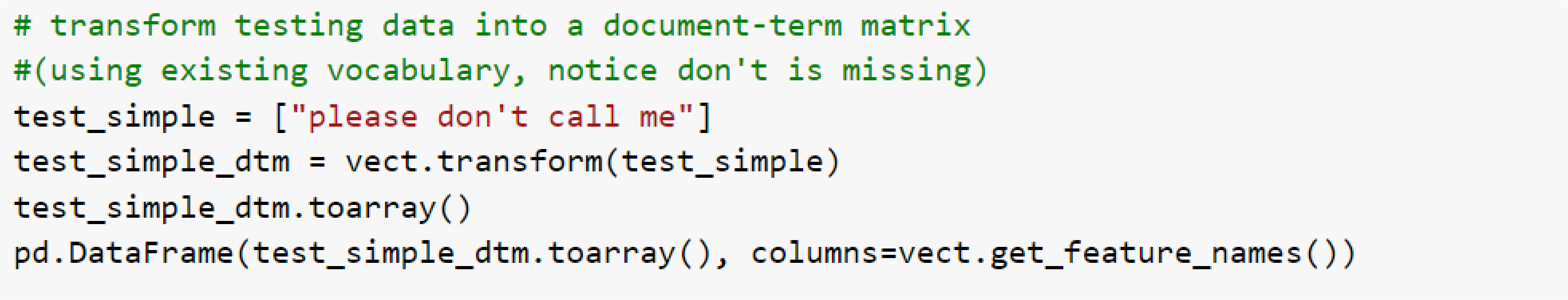
Spam Text Classification



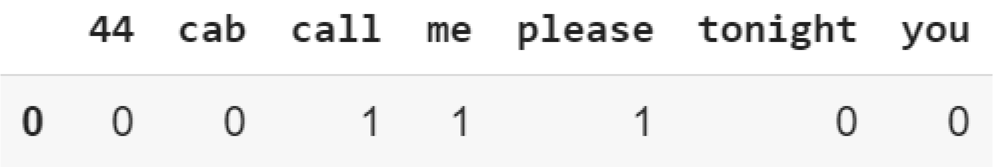
```
# simple count vectorizer example
from sklearn.feature_extraction.text import CountVectorizer
# start with a simple example
train_simple = ['call you tonight',
                'Call me a cab',
                'please call me... PLEASE 44!']
# learn the 'vocabulary' of the training data
vect = CountVectorizer()
train_simple_dtm = vect.fit_transform(train_simple)
pd.DataFrame(train_simple_dtm.toarray(), columns=vect.get_feature_names())
```

	44	cab	call	me	please	tonight	you
0	0	0	1	0	0	1	1
1	0	1	1	1	0	0	0
2	1	0	1	1	2	0	0

Spam Text Classification



```python
# transform testing data into a document-term matrix
#(using existing vocabulary, notice don't is missing)
test_simple = ["please don't call me"]
test_simple_dtm = vect.transform(test_simple)
test_simple_dtm.toarray()
pd.DataFrame(test_simple_dtm.toarray(), columns=vect.get_feature_names())
```



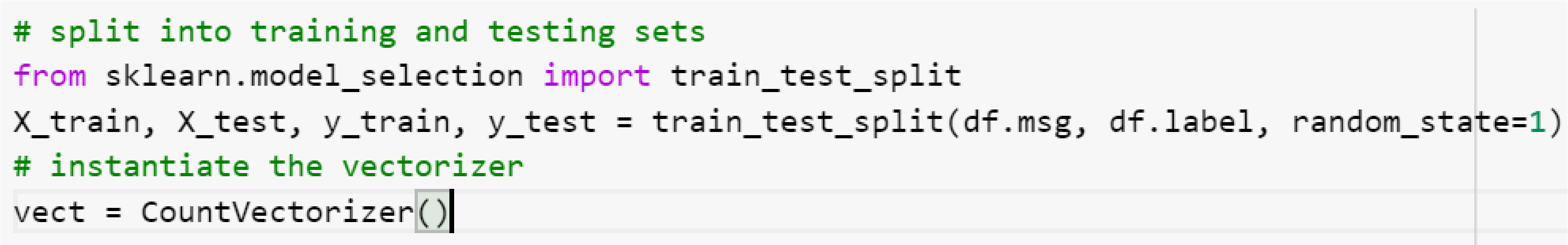
	44	cab	call	me	please	tonight	you
0	0	0	1	1	1	0	0

Our test sentence we had a new word, namely *don't*. When we vectorized it, because we hadn't seen that word previously in our training data, the vectorizer simply ignored it.

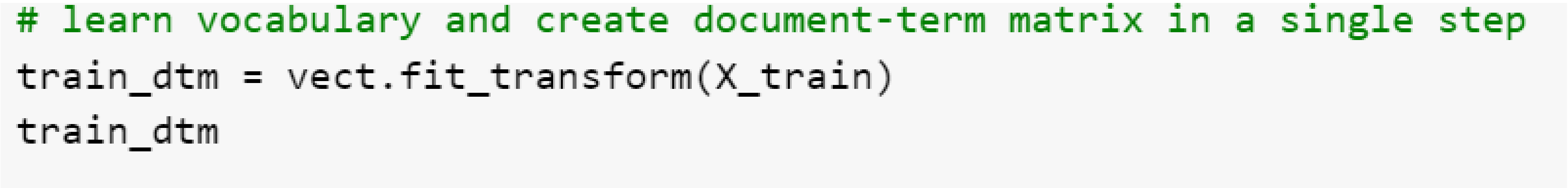
This is important, and incentivizes data scientists to obtain as much data as possible for their training sets.

Now let's do this for our actual data:

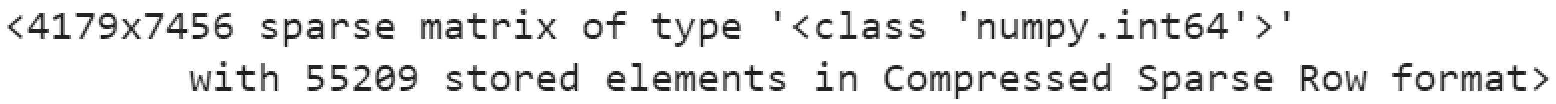
Spam Text Classification



```
# split into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df.msg, df.label, random_state=1)
# instantiate the vectorizer
vect = CountVectorizer()
```



```
# learn vocabulary and create document-term matrix in a single step
train_dtm = vect.fit_transform(X_train)
train_dtm
```



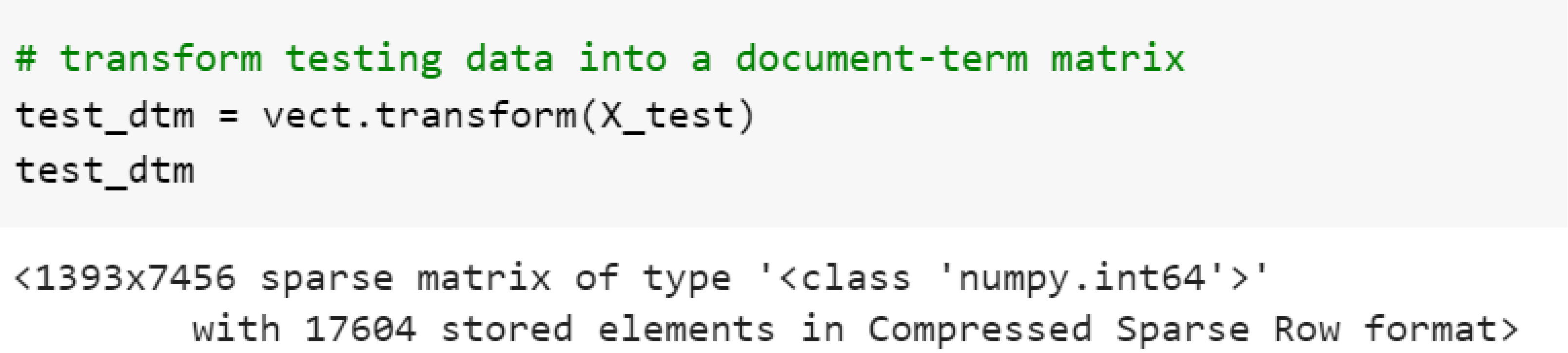
```
<4179x7456 sparse matrix of type '<class 'numpy.int64'>'
        with 55209 stored elements in Compressed Sparse Row format>
```

With 55209 stored elements in compressed sparse matrix. The matrix is so large and full of zeroes, there exists a special format to deal with objects such as this.

Take a look at the number of columns. 7,456 words!!

Spam Text Classification

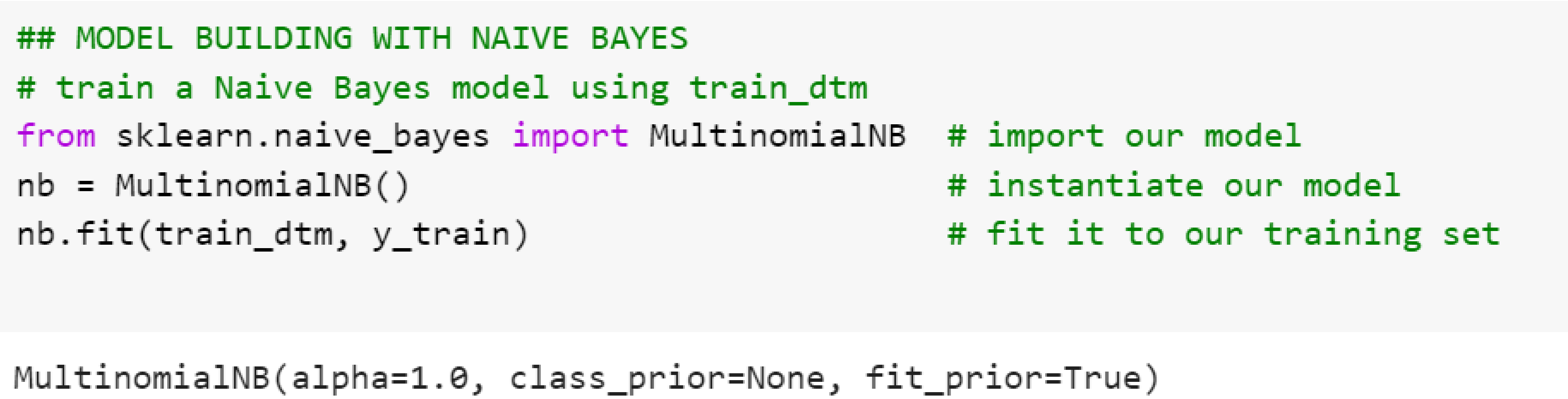
- We can now transform our test data to conform to our vocabulary:



```
# transform testing data into a document-term matrix
test_dtm = vect.transform(X_test)
test_dtm
```

```
<1393x7456 sparse matrix of type '<class 'numpy.int64'>'
	with 17604 stored elements in Compressed Sparse Row format>
```

- Now let's build a Naïve Bayes model

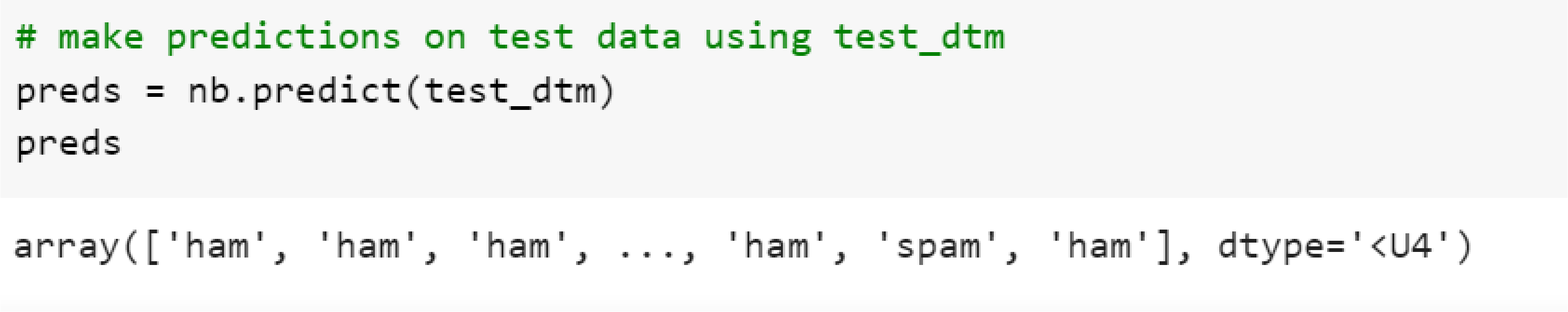


```
## MODEL BUILDING WITH NAIVE BAYES
# train a Naive Bayes model using train_dtm
from sklearn.naive_bayes import MultinomialNB  # import our model
nb = MultinomialNB()                           # instantiate our model
nb.fit(train_dtm, y_train)                     # fit it to our training set
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Spam Text Classification

- Now the variable nb holds our fitted model. The training phase of the model involves computing the likelihood function, which is the conditional probability of each feature given each class:



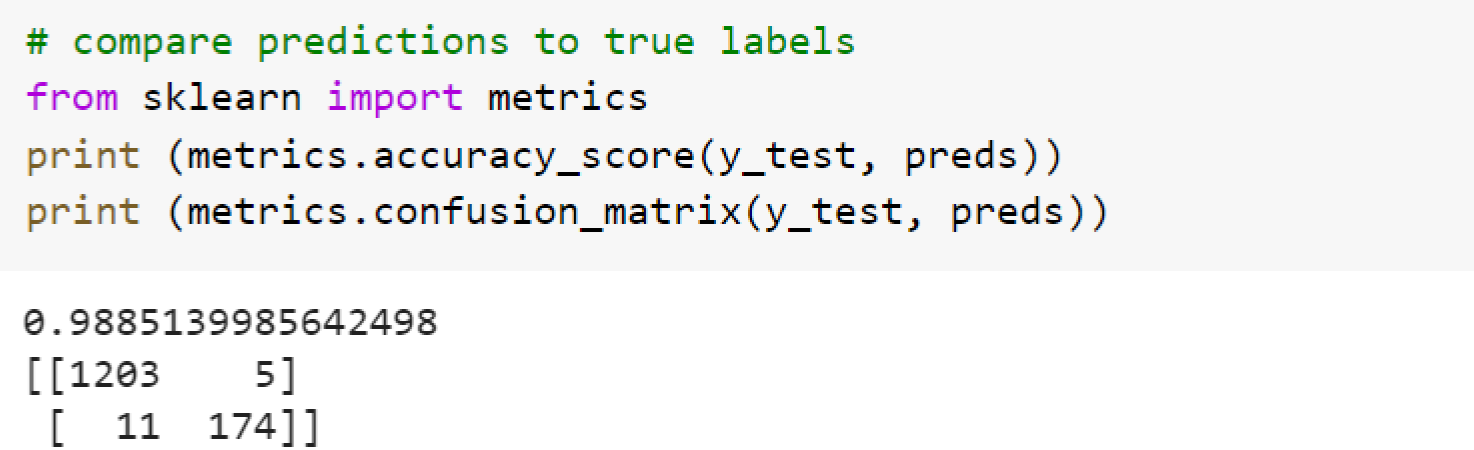
```
# make predictions on test data using test_dtm
preds = nb.predict(test_dtm)
preds
```

```
array(['ham', 'ham', 'ham', ..., 'ham', 'spam', 'ham'], dtype='<U4')
```

- The prediction phase of the model involves computing the posterior probability of each class given the observed features, and choosing the class with the highest probability.

Spam Text Classification

- We will use sklearn's built-in accuracy and confusion matrix to look at how well our Naïve Bayes models are performing:



```
# compare predictions to true labels
from sklearn import metrics
print (metrics.accuracy_score(y_test, preds))
print (metrics.confusion_matrix(y_test, preds))
```

```
0.9885139985642498
[[1203    5]
 [  11  174]]
```

- Our accuracy is great! Compared to our null accuracy (87%), 98.85% is a fantastic improvement.
- Confusion matrix:
  - 1203 refers to true ham predictions and 174 refers to true spam predictions.
  - False spam classification = 5 and false ham classification = 11

UNIVERSITAS INDONESIA
Veritas, Probitas, Iustitia

FAKULTAS
ILMU
KOMPUTER

THANK YOU

Wish you success ☺