Lecture 2:

Naïve Bayes and Parameter Estimation

Markus Hohle

University California, Berkeley

Bayesian Data Analysis and Machine Learning for Physical Sciences

Bayesian Data Analysis and Machine Learning for Physical Sciences

Course Map

Module 1	Maximum Entropy and Information, Bayes Theorem
Module 2	**Naive Bayes, Bayesian Parameter Estimation, MAP**
Module 3	Model selection: Comparing Distributions vs Frequentist Methods
Module 4	Model Selection: Bayesian Signal Detection
Module 5	Variational Bayes, Expectation Maximization
Module 6	Stochastic Processes
Module 7	Monte Carlo Methods
Module 8	Markov Models, Graphs
Module 9	Machine Learning Overview, Supervised Methods
Module 10	Unsupervised Methods
Module 11	ANN: Perceptron, Backpropagation
Module 12	ANN: Basic Architecture, Regression vs Classification, Backpropagation again
Module 13	Convolution and Image Classification and Segmentation
Module 14	TBD (GNNs)
Module 15	TBD (RNNs and LSTMs)
Module 16	TBD (Transformer and LLMs)

Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

Naïve Bayes

- Idea
- Example

Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

Naïve Bayes

- Idea
- Example

Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP:

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

Idea

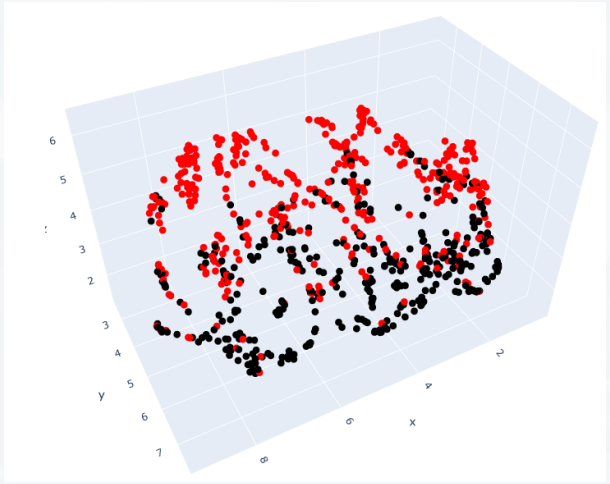
Example

$\vec{x}$: vector with all model parameters (or features)

Index	molecular_weight	electronegativity	bond_lengths	num_hydrogen_bonds	logP	label
0	413.228	2.94416	3.41991	1	10.4335	Toxic
1	447.945	3.55371	3.66831	7	10.3475	Toxic
2	309.199	3.19761	2.84841	0	7.88825	Non-Toxic
3	382.554	3.8653	3.46237	8	9.59041	Toxic
4	310.904	3.18141	2.87774	6	7.85477	Non-Toxic
5	353.857	3.12105	3.32724	6	8.58887	Non-Toxic

K different classes
(here K = 2)

UMAP projection
(**toxic**, **non-toxic**)



Naive Bayes, Bayesian Parameter Estimation, MAP:

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

Idea
Example

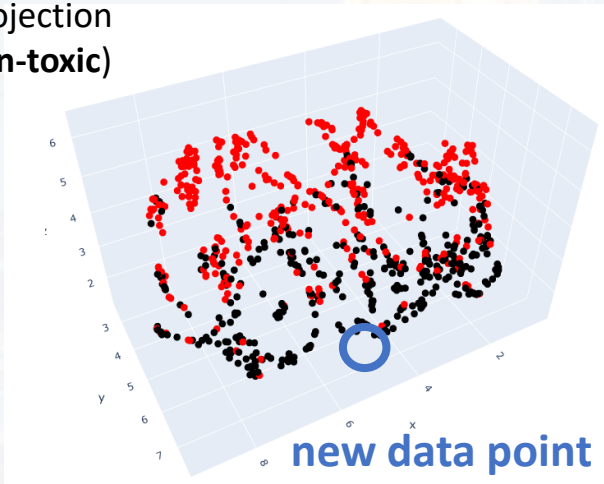
goal: predicting class C_k of a new datum, given $\vec{x}$

$P(C_k|\vec{x})$: probability that datapoint belongs to class C_k, given $\vec{x}$
$P(\vec{x}|C_k)$: probability that datapoint has features $\vec{x}$, given class C_k

$$P(C_k|\vec{x}) = \frac{P(\vec{x}|C_k)P(C_k)}{P(\vec{x})} = \frac{P(C_k)}{P(\vec{x})}\prod_{i=1}^{I} P(x_i|C_k) \sim P(C_k)\prod_{i=1}^{I} P(x_i|C_k) \qquad \sum_{k=1}^{K} P(C_k|\vec{x}) = 1$$

Naïve Bayes:
- all features are **mutually independent**
- i. e.: no **correlation** between features
- features can be factorized

UMAP projection
(**toxic**, **non-toxic**)



$$k_{new} = \underset{k}{argmax}\left\{P(C_k)\prod_{i=1}^{I} P(x_i|C_k)\right\}$$

from the training data → supervised learning (see later)

Naive Bayes, Bayesian Parameter Estimation, MAP:

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

Idea
Example

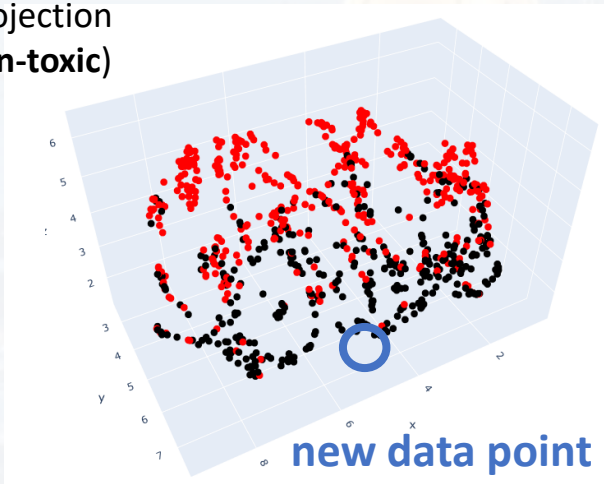
goal: predicting class C_k of a new datum, given $\vec{x}$

$P(C_k|\vec{x})$: **probability that datapoint belongs to class C_k, given $\vec{x}$**

$P(\vec{x}|C_k)$: **probability that datapoint has features $\vec{x}$, given class C_k**

$$P(C_k|\vec{x}) = \frac{P(\vec{x}|C_k)P(C_k)}{P(\vec{x})} = \frac{P(C_k)}{P(\vec{x})}\prod_{i=1}^{I} P(x_i|C_k) \sim P(C_k)\prod_{i=1}^{I} P(x_i|C_k) \qquad \sum_{k=1}^{K} P(C_k|\vec{x}) = 1$$

UMAP projection
(**toxic**, **non-toxic**)



$$k_{new} = \underset{k}{argmax}\left\{P(C_k)\prod_{i=1}^{I} P(x_i|C_k)\right\}$$

from the training data → supervised learning (see later)

different models for $P(x_i|C_k)$

- multinomial
- Gaussian
- …

Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

Naïve Bayes

- Idea
- Example

Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP:

from the training data → supervised learning

1) creating the model:

```
my_model = library.method(argument1 = 'arg1', … )
```

2) training the model

```
out = my_model.fit(xtrain, ytrain)
```

3) evaluation

```
ypred = out.predict(xeval)
accur = (ypred == yeval).sum()/len(yeval)
```

4) prediction (actual application)

```
ypred = out.predict(xnew)
```

Naive Bayes, Bayesian Parameter Estimation, MAP:

Python:

```
from sklearn.naive_bayes import *
from sklearn.preprocessing import MinMaxScaler
```

importing methods for naïve bayes

scaling/normalizing data

Idea
Example

```
Train = pd.read_csv('molecular_train_gbc_cat.csv')
Test  = pd.read_csv('molecular_test_gbc_cat.csv')

XTrain = Train.drop('label', axis = 1).values
YTrain = Train['label']

XTest = Test.drop('label', axis = 1).values
YTest = Test['label']
```

```
print(YTrain[:10])
```

```
0        Toxic
1        Toxic
2    Non-Toxic
3    Non-Toxic
4    Non-Toxic
5        Toxic
6    Non-Toxic
7    Non-Toxic
8        Toxic
9    Non-Toxic
Name: label, dtype: object
```

Naive Bayes, Bayesian Parameter Estimation, MAP:

Idea

Example

```
scaler  = MinMaxScaler(feature_range = (0, 1))
XTrainS = scaler.fit_transform(XTrain)
```

scaling the data to mean = 0 and std = 1

```
gnb = GaussianNB()
Fit = gnb.fit(XTrainS, YTrain)
```

the actual fit

applying the model to the test data set

```
XTestS = scaler.transform(XTest)
```

scaling the test set **without** fitting

```
Ypred  = Fit.predict(XTestS)
Probs  = Fit.predict_proba(XTestS)
```

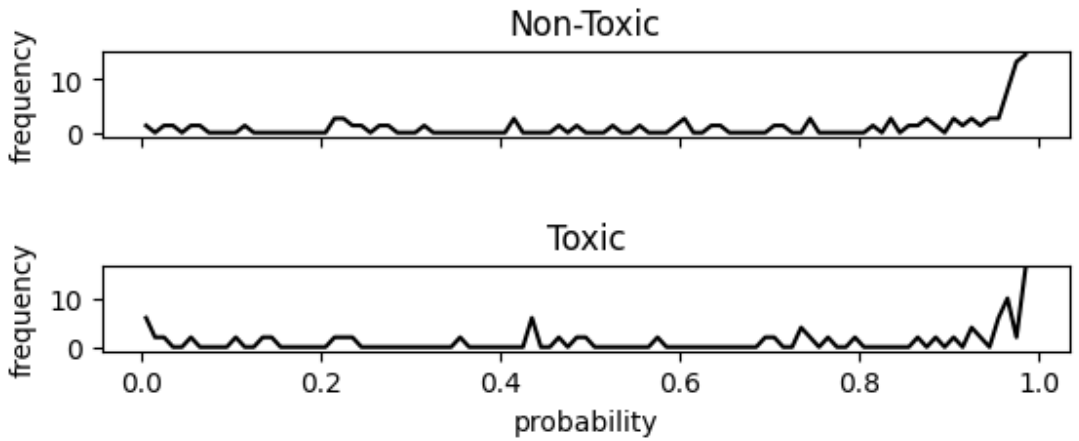
predicting the class C_k and calculating the probabilities

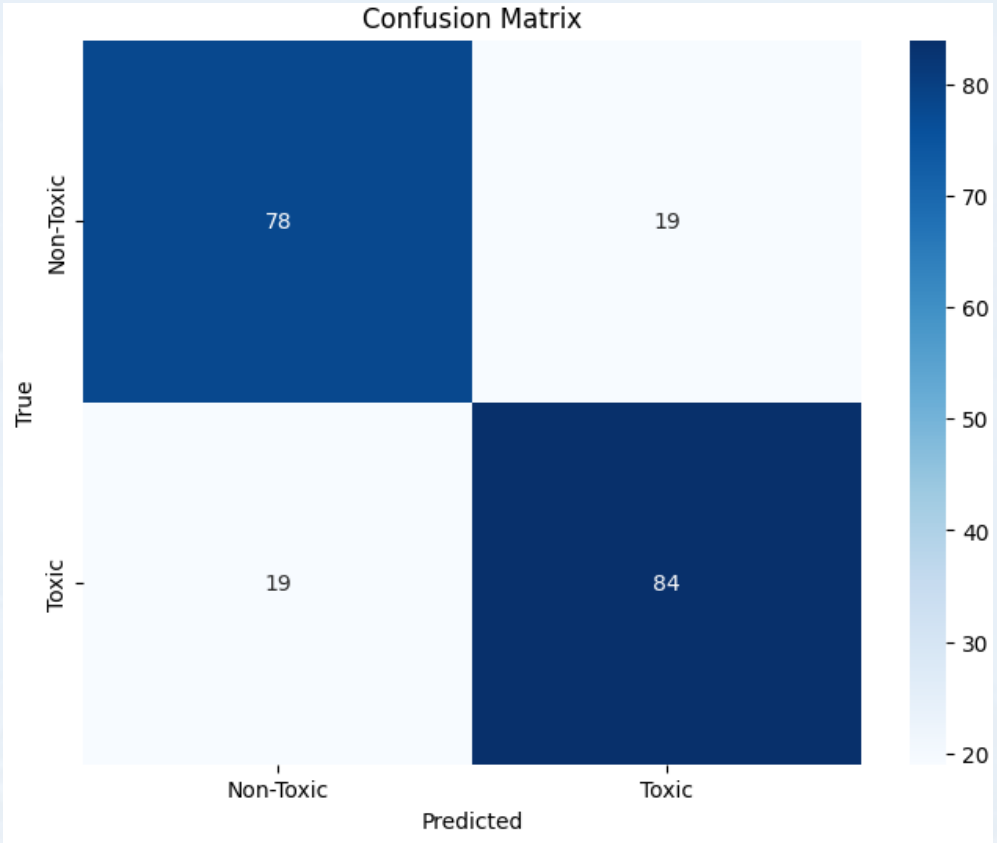
Idea
Example

```
XTestS  = scaler.transform(XTest)

Ypred   = Fit.predict(XTestS)
Probs   = Fit.predict_proba(XTestS)
```

evaluation





see
Walk_Through_NaiveBayes.ipynb

Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

Naïve Bayes

- Idea
- Example

Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP:

D:	data set
q:	parameter
L=L(D|q):	**known** likelihood function

Idea

Sharing new Information

Laplace Approximation and MAP

goal: finding the parameter *q* as *P(q|D)* such that
- entropy of *P(q|D)* **decreases** for **increasing** amount of *D*
- we derive the actual *P(q|D)*

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

likelihood function **(eg: binomial)**

$$P(q|D) = \frac{P(D|q)P(q)}{P(D)}$$ prior

evidence **(const wrt q)**

$$\int P(q|D)\ dq = 1$$

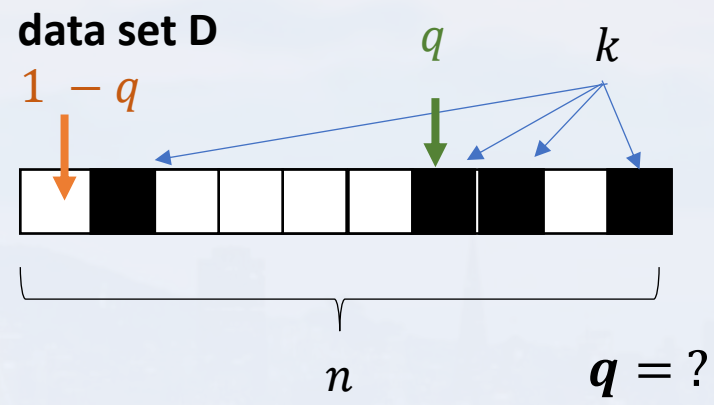
Naive Bayes, Bayesian Parameter Estimation, MAP:

D:	data set
q:	parameter
L=L(D|q):	**known** likelihood function

Idea

Sharing new Information

Laplace Approximation and MAP



likelihood function **(eg: binomial)**

$$P(q|D) = \frac{P(D|q)P(q)}{P(D)}$$

prior

evidence **(const wrt q)**

in **this** example: $P(k|n,q) = \binom{n}{k} q^k (1-q)^{n-k}$

$$P(q|D) = \frac{\binom{n}{k} q^k (1-q)^{n-k}}{P(D)} P(q)$$

$P(D)$ and $\binom{n}{k}$ are no functions of q

$$\sim q^k (1-q)^{n-k} P(q)$$

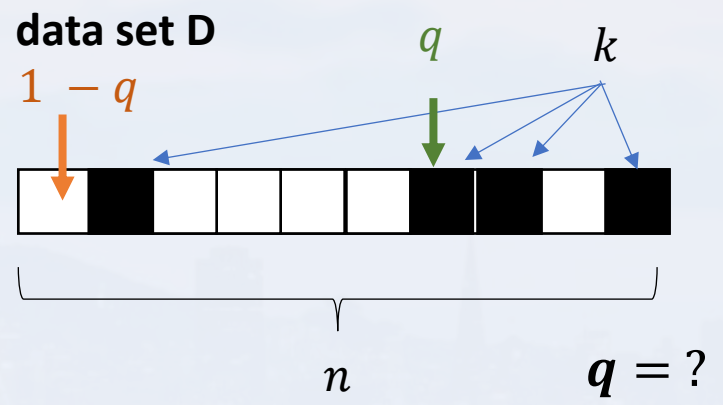
Naive Bayes, Bayesian Parameter Estimation, MAP:

D:	data set
q:	parameter
L=L(D|q):	**known** likelihood function

Idea

Sharing new Information

Laplace Approximation and MAP



in **this** example: $P(k|n,q) = \binom{n}{k} q^k (1-q)^{n-k}$

$$P(q|D) = \frac{\binom{n}{k}\, q^k (1-q)^{n-k}}{P(D)} P(q)$$

$$\sim q^k (1-q)^{n-k}\, P(q)$$

max. entropy: $P(q) = const$
if no prior information about q

$$\sim q^k (1-q)^{n-k}$$

$$P(q|D) = \frac{q^k (1-q)^{n-k}}{\int_0^1 q^k (1-q)^{n-k}\, dq}$$

Naive Bayes, Bayesian Parameter Estimation, MAP:

check out `bayesian_bino.py`

```
n1 = 4
k1 = np.random.binomial(n1, 0.25)
```

creating artificial data set
note: in reality ***q*** is unknown!

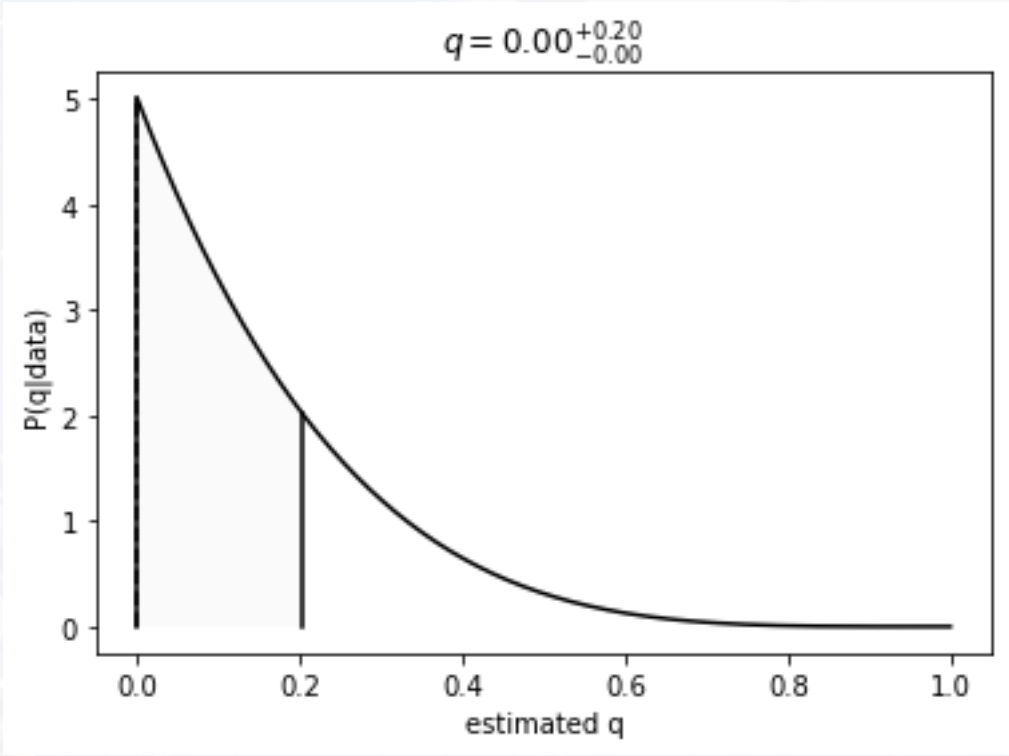
```
[q1, b, _] = bayesian_bino(n1, k1)
```

Idea

Sharing new Information

Laplace Approximation and MAP

$$P(q|data\ set) = \frac{q^k(1-q)^{n-k}}{\int_0^1 q^k(1-q)^{n-k}\ dq}$$



Naive Bayes, Bayesian Parameter Estimation, MAP:

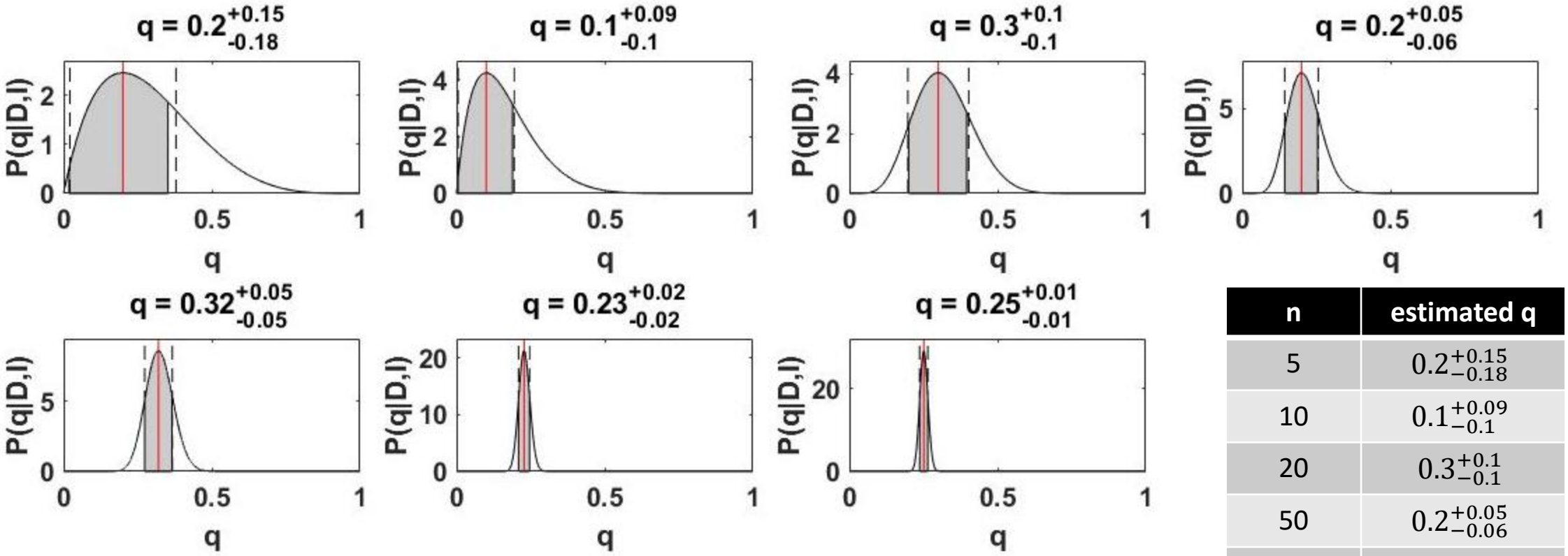
check out `bayesian_bino.py`

Idea

Sharing new Information

Laplace Approximation and MAP

increasing n



n	estimated q
5	$0.2^{+0.15}_{-0.18}$
10	$0.1^{+0.09}_{-0.1}$
20	$0.3^{+0.1}_{-0.1}$
50	$0.2^{+0.05}_{-0.06}$
100	$0.32^{+0.05}_{-0.05}$
500	$0.23^{+0.02}_{-0.02}$
1,000	$0.25^{+0.01}_{-0.01}$
infinity	0.25

Naive Bayes, Bayesian Parameter Estimation, MAP:

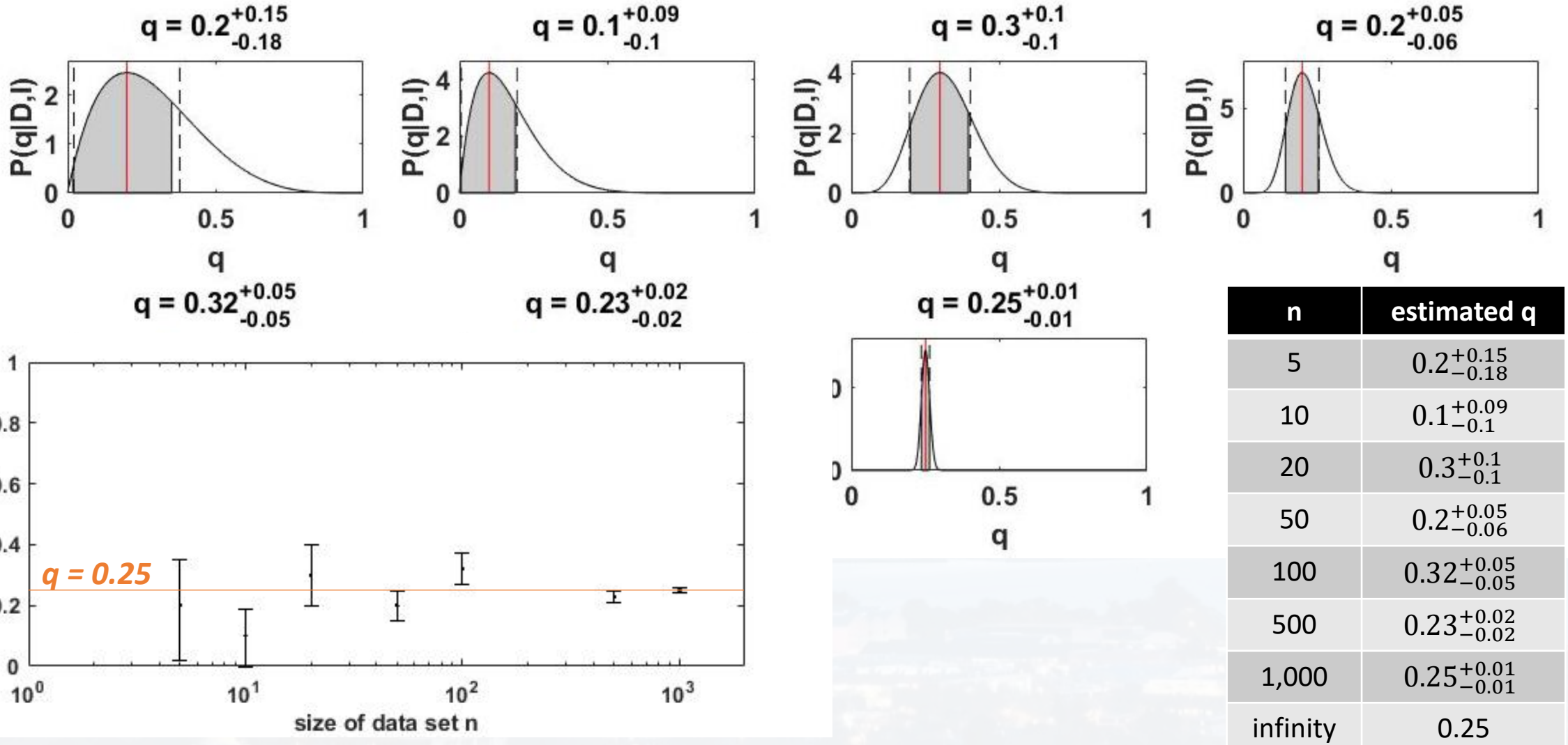
check out `bayesian_bino.py`

Idea

Sharing new Information

Laplace Approximation and MAP

increasing n



n	estimated q
5	$0.2^{+0.15}_{-0.18}$
10	$0.1^{+0.09}_{-0.1}$
20	$0.3^{+0.1}_{-0.1}$
50	$0.2^{+0.05}_{-0.06}$
100	$0.32^{+0.05}_{-0.05}$
500	$0.23^{+0.02}_{-0.02}$
1,000	$0.25^{+0.01}_{-0.01}$
infinity	0.25

Naive Bayes, Bayesian Parameter Estimation, MAP:

Bayesian Parameter Estimation works with **any other pdf**

What is the average number of WhatsUp messages I get every day?

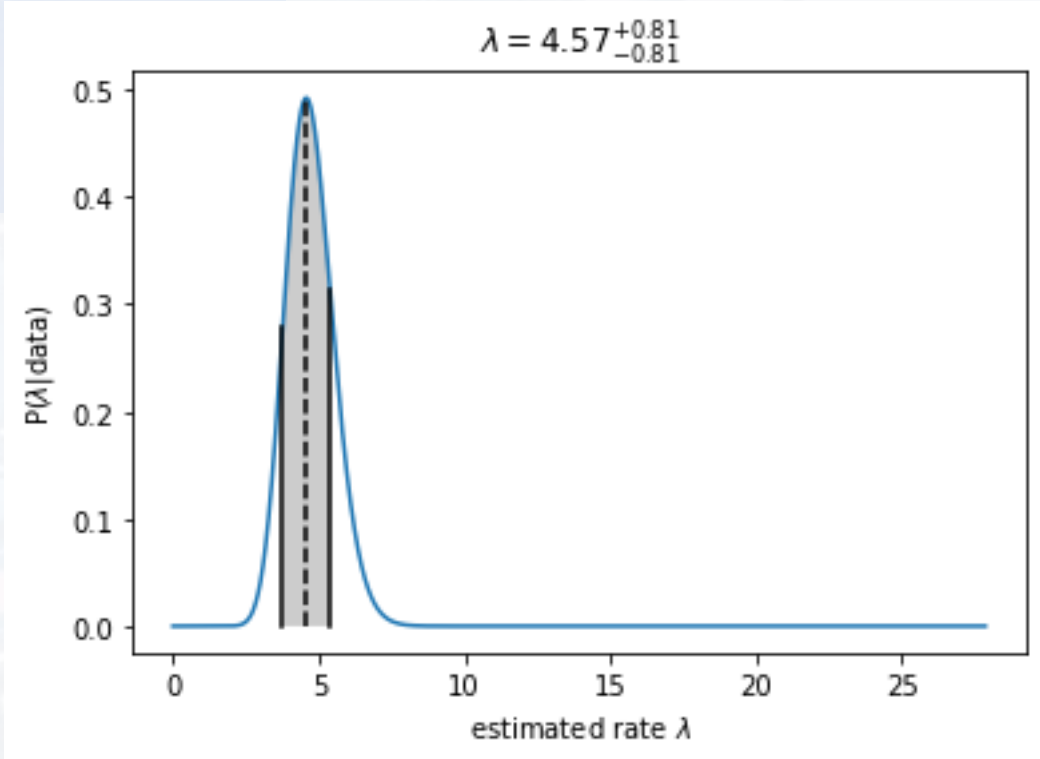
Mon:	5
Tue:	7
Wed:	1
Thu:	3
Fri:	9
Sat:	2
Sun:	5

event
- has no duration
- is rare

→ Poissonian

likelihood function $P(k|\lambda) = \frac{\lambda^k}{k!} e^{-\lambda}$

$$P(q|D) = \frac{P(D|q)P(q)}{P(D)}$$



Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

Naïve Bayes

- Idea
- Example

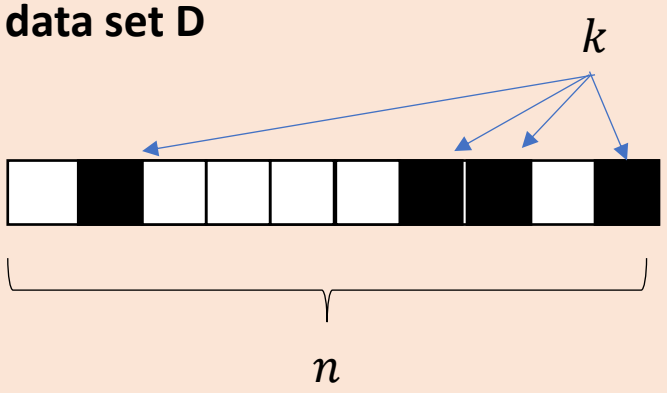
Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP:

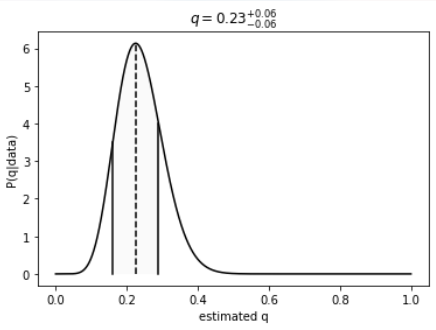
What if there is new data?

data set D

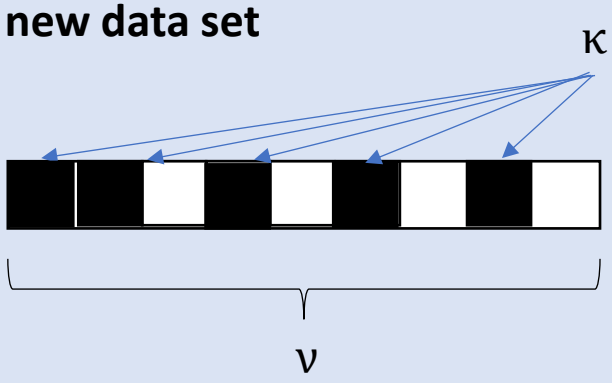


~~$q = ?$~~

$$P(q|D) = \frac{q^k(1-q)^{n-k}}{\int_0^1 q^k(1-q)^{n-k}\ dq}$$



new data set



if there **is** prior information ***I*** about q:

$$P(q|new\ data\ set, I) = \frac{P(new\ data\ set|q, I)\ \boldsymbol{P(q, I)}}{P(new\ data\ set)}$$

Naive Bayes, Bayesian Parameter Estimation, MAP:

What if there is new data?

$$P(q|new\ data\ set, I) = \frac{P(new\ data\ set|q, I)\ \boldsymbol{P(q, I)}}{P(new\ data\ set)}$$

$$P(q|D) = \frac{q^k(1-q)^{n-k}}{\int_0^1 q^k(1-q)^{n-k}\ dq}$$

$$= \frac{q^{\kappa}(1-q)^{\nu-\kappa}\ q^k(1-q)^{n-k}}{\int_0^1 q^{\kappa}(1-q)^{\nu-\kappa}\ q^k(1-q)^{n-k}\ dq}$$

$$= \frac{q^{k+\kappa}(1-q)^{\nu-\kappa+n-k}}{\int_0^1 q^{k+\kappa}(1-q)^{\nu-\kappa+n-k}\ dq}$$

often: $\kappa = \alpha - 1$

$\beta = \nu - \kappa - 1$

$$= \frac{q^{k+\alpha-1}(1-q)^{n-k+\beta-1}}{\int_0^1 q^{k+\alpha-1}(1-q)^{n-k+\beta-1}\ dq}$$

Beta function **(conjugate prior of the binomial distribution)**

Naive Bayes, Bayesian Parameter Estimation, MAP:

What if there is new data?

Likelihood $p(x_i \mid \theta)$	Model parameters θ	Conjugate prior (and posterior) distribution $p(\theta \mid \Theta), p(\theta \mid \mathbf{x}, \Theta) = p(\theta \mid \Theta')$	Prior hyperparameters Θ
Bernoulli	p (probability)	Beta	$\alpha, \beta \in \mathbb{R}$
Binomial with known number of trials, m	p (probability)	Beta	$\alpha, \beta \in \mathbb{R}$
Negative binomial with known failure number, r	p (probability)	Beta	$\alpha, \beta \in \mathbb{R}$
Poisson	λ (rate)	Gamma	$k, \theta \in \mathbb{R}$
			α, β[note 4]
Categorical	$\boldsymbol{p}$ (probability vector), k (number of categories; i.e., size of $\boldsymbol{p}$)	Dirichlet	$\boldsymbol{\alpha} \in \mathbb{R}^k$
Multinomial	$\boldsymbol{p}$ (probability vector), k (number of categories; i.e., size of	Dirichlet	$\boldsymbol{\alpha} \in \mathbb{R}^k$

Idea

Sharing new Information

Laplace Approximation and MAP

see example “Model Selection” about discrete signals

Naive Bayes, Bayesian Parameter Estimation, MAP:

What if there is new data?

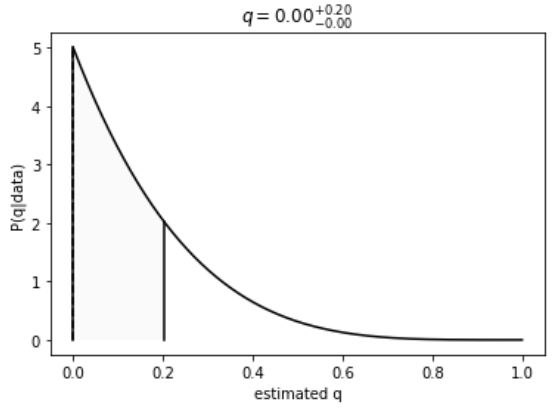
$$P(q|new\ data\ set, I) = \frac{q^{\kappa}(1-q)^{\nu-\kappa}\ q^{k}(1-q)^{n-k}}{\int_0^1 q^{\kappa}(1-q)^{\nu-\kappa}\ q^{k}(1-q)^{n-k}\ dq}$$

Idea

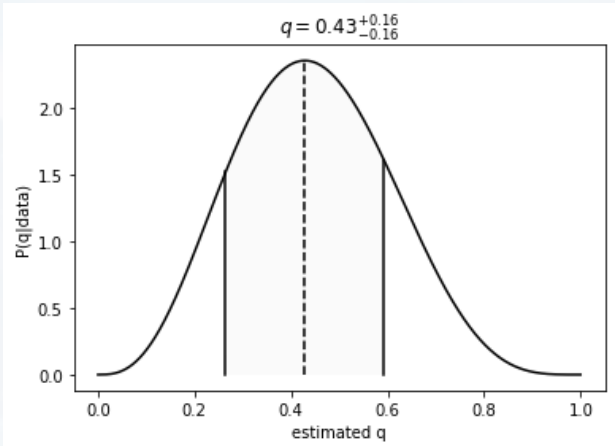
Sharing new Information

Laplace Approximation and MAP

```
n1 = 4
k1 = np.random.binomial(n1, q = 0.2)
[_, _, Prior] = bayesian_bino(n1, k1)
```

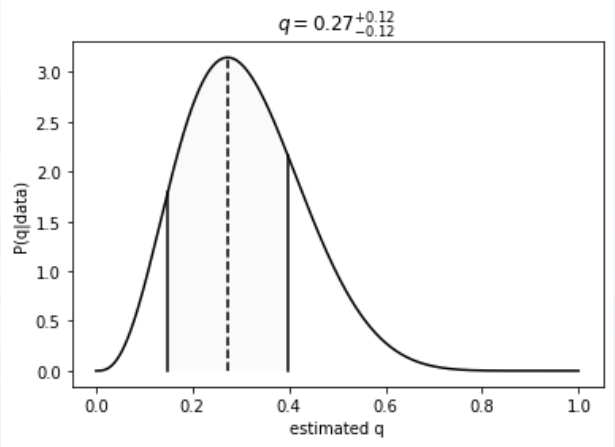



```
n2 = 7
k2 = np.random.binomial(n2, q = 0.2)
[_, _, _] = bayesian_bino(n2, k2)
```



$$P(q, I) = \frac{q^{k}(1-q)^{n-k}}{\int_0^1 q^{k}(1-q)^{n-k}\ dq}$$

```
[_ , _, _] = bayesian_bino(n2, k2, Prior = Prior)
```

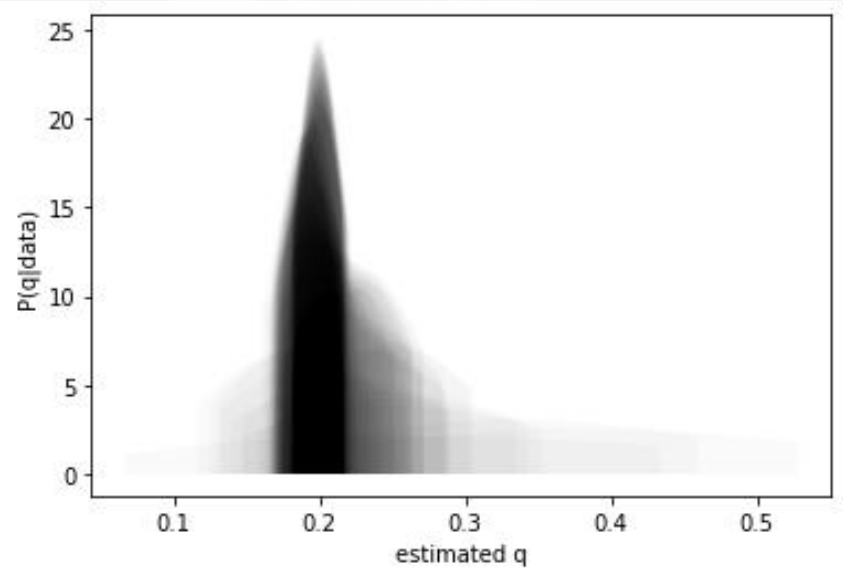


Naive Bayes, Bayesian Parameter Estimation, MAP:

What if there is new data?

$$P(q|new\ data\ set, I) = \frac{q^{\kappa}(1-q)^{\nu-\kappa}\ q^{k}(1-q)^{n-k}}{\int_0^1 q^{\kappa}(1-q)^{\nu-\kappa}\ q^{k}(1-q)^{n-k}\ dq}$$

The **posterior** from the **previous** experiment is the **prior** of the **next** experiment



→ we become more certain about the model parameters
→ learning!

→ see e.g. **V**ariational **A**uto **E**ncoders

Naive Bayes, Bayesian Parameter Estimation, MAP

Outline

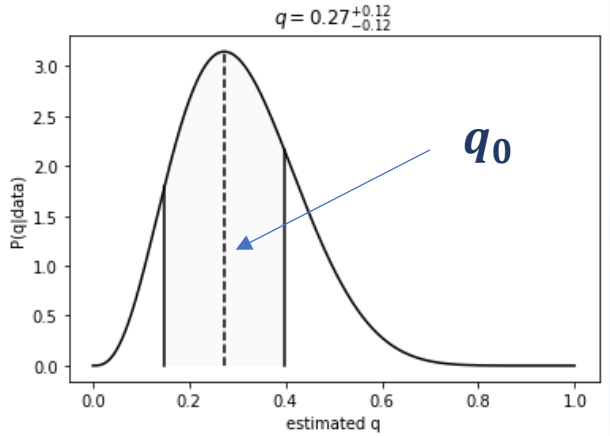
Naïve Bayes

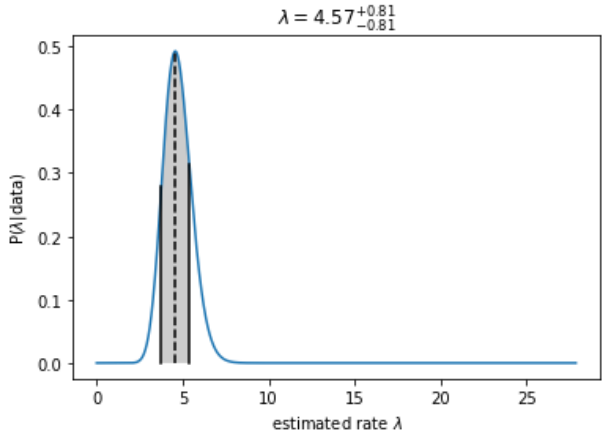
- Idea
- Example

Parameter Estimation

- Idea
- Sharing new Information
- Laplace Approximation and MAP

Naive Bayes, Bayesian Parameter Estimation, MAP:





Idea
Sharing new Information
Laplace Approximation and MAP

***D*:**	**data set**
***q*:**	**parameter**
q_0**:**	**q for which *P(q|D)* reaches max**

for large *D*, the posterior seems to approach a Gaussian

Laplace approximation: 2nd order Taylor approximation of $ln[P(q|D)]$

$$L = ln[P(q|D)] \approx ln[P(q_0|D)] + \frac{d}{dq} ln[P(q|D)]|_{q=q_0}(q - q_0) + \frac{1}{2}\frac{d^2}{dq^2} ln[P(q|D)]|_{q=q_0}(q - q_0)^2$$

= 0 (evaluated at maximum)

$$P(q|D) \approx P(q_0|D) \exp\left\{\frac{1}{2}\frac{d^2}{dq^2} ln[P(q|D)]|_{q=q_0}(q - q_0)^2\right\}$$

Naive Bayes, Bayesian Parameter Estimation, MAP:

$$P(q|D) \approx P(q_0|D)\exp\left\{\frac{1}{2}\frac{d^2}{dq^2}ln[P(q|D)]|_{q=q_0}(q-q_0)^2\right\}$$

$$\mu = q_0$$

$$\sigma^2 = \frac{1}{\frac{d^2}{dq^2}ln[P(q|D)]|_{q=q_0}}$$

for **example**, if $P(D|q)$ is a **binomial** distribution:

$$P(q|D) \sim q^k(1-q)^{n-k}$$

$$q_0 = \frac{k}{n}$$

$$\sigma = \sqrt{\frac{q_0(1-q_0)}{n}}$$

$$q = \frac{k}{n} \pm \sqrt{\frac{q_0(1-q_0)}{n}}$$

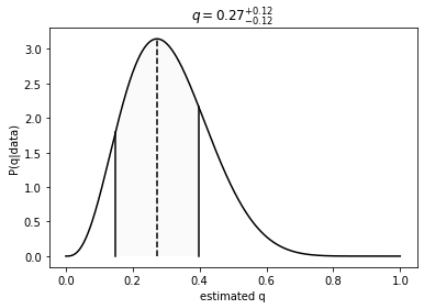
for large n:
approaches frequentist results
(like maximum likelihood estimation, MLE)

Idea
Sharing new Information
Laplace Approximation and MAP

***D*:**	**data set**
***q*:**	**parameter**
q_0**:**	**q for which *P(q|D)* reaches max**

Naive Bayes, Bayesian Parameter Estimation, MAP:

$$P(q|D) \approx P(q_0|D) \exp\left\{\frac{1}{2}\frac{d^2}{dq^2} ln[P(q|D)]|_{q=q_0} (q - q_0)^2\right\}$$



Idea
Sharing new Information
Laplace Approximation and MAP

***D*:**	**data set**
***q*:**	**parameter**
q_0**:**	**q for which *P(q|D)* reaches max**

note:

- using Bayesian Parameter Estimation, we find the **pdf** $P(q|D)$ of the desired parameter q
- the $P(q|D)$ is flat for small *D*, but becomes more defined for large *D* → **reflects the exact amount of information with have about q**
- **integration** around the maximum of $P(q|D)$ provides **confidence intervals** about q
- for large *D*, we can apply the **Laplace approximation**, which is equal to the frequentist result
- finding the maximum of $P(q|D)$, given a prior is called **maximum a posteriori (MAP)** estimate
- prior information acts as optimization weight

Thank you very much for your attention!