COMS 4771 Machine Learning
Problem Set #3

Jun Hu - jh3846@columbia.edu
Discussants:

July 12, 2017

Problem 1

(a) Execute:

```
library(MASS)
attach(Boston)
mu.hat = mean(medv)
mu.hat
```

```
[1] 22.53281
```

The mean of **medv**, the estimate $\hat{\mu}$ is 22.53281.

(b) By the definition:

```
sem.mu.hat = sd(medv)/sqrt(length(medv))
sem.mu.hat
```

```
[1] 0.4088611
```

The standard error of the $\hat{\mu}$ is 0.4088611. It is the standard deviation of the **medv**-sample-mean's estimate of the its population-mean.

(c) By bootstrap:

```
library(boot)
set.seed(1)
boot.fn = function(data, index){
return(mean(data[index]))
}
boot(medv, boot.fn, 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
original      bias    std. error
t1* 22.53281 0.008517589   0.4119374
```

The standard error of the $\hat{\mu}$ is 0.4119374 by bootstrap, which is very close to the $\hat{\mu}$ obtain in **(b)**.

(d) The 95% confidence interval by bootstrap:

```
ci.bootstrap = c(22.53281-2*0.4119374, 22.53281+2*0.4119374)
ci.bootstrap
```

```
[1] 21.70894 23.35668
```

The 95% confidence interval by **t.test(medv)**:

```
t.test(medv)
```

```
One Sample t-test

data:  medv
t = 55.111, df = 505, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
21.72953 23.33608
sample estimates:
mean of x
22.53281
```

They are very close to each other (only about 0.02 difference for the lower/higher bound).

(e) The median of **medv** $\hat{\mu}_{med}$:

```
med.hat = median(medv)
med.hat
```

```
[1] 21.2
```

The $\hat{\mu}_{med}$ is 21.2.

(f) The standard error of $\hat{\mu}_{med}$:

```
boot.fn = function(data, index){
return(median(data[index]))
}
boot(medv, boot.fn, 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
original  bias    std. error
t1*     21.2 -0.0098   0.3874004
```

The estimated median of **medv** is 21.2 which is exactly the same as $\hat{\mu}_{med}$ in **(e)**, and the standard error found by bootstrap is 0.3874004, which is relatively small. Furthermore, the standard error of $\hat{\mu}_{med}$ found by bootstrap is also smaller than the standard error of mean in this case.

(g) Let's compute the $10th$ percentile of **medv** $\hat{\mu}_{0.1}$:

```
pct.hat = quantile(medv, c(0.1))
pct.hat
```

```
  10%
12.75
```

The $\hat{\mu}_{0.1}$ is 12.75%

(h) The standard error of $\hat{\mu}_{0.1}$ by bootstrap:

```
boot.fn = function(data, index){
return(quantile(data[index], c(0.1)))
}
boot(medv, boot.fn, 1000)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
original  bias    std. error
t1*    12.75 0.00515   0.5113487
```

By using bootstrap, we obtain the estimated value of $10th$ percentile of $\hat{\mu}_{0.1}$ is the same as in **(g)**, and the standard error 0.5113487 is relatively small. This has proven that bootstrap analysis can be applied in lots of situations.

Problem 2

Let $D(\mathbf{x}, \mathbf{x_n})$ to be the distance from $\mathbf{x}$ to $\mathbf{x_n}$. By decomposition of $D(\mathbf{x}, \mathbf{x_n})$ into inner products and substitution of $K(\mathbf{x_i}, \mathbf{x_j}) = \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})$ for the inner products, we have

$$
\begin{aligned}
D(\mathbf{x}, \mathbf{x_n}) &= \|\mathbf{x} - \mathbf{x_n}\|^2 \\
&= (\mathbf{x} - \mathbf{x_n})(\mathbf{x} - \mathbf{x_n})^{\mathrm{T}} \\
&= (\mathbf{x} - \mathbf{x_n})(\mathbf{x}^{\mathrm{T}} - \mathbf{x_n}^{\mathrm{T}}) \\
&= \mathbf{x}\mathbf{x}^{\mathrm{T}} - \mathbf{x}\mathbf{x_n}^{\mathrm{T}} - \mathbf{x_n}\mathbf{x}^{\mathrm{T}} + \mathbf{x_n}\mathbf{x_n}^{\mathrm{T}} \\
&= \mathbf{x} \cdot \mathbf{x} - \mathbf{x} \cdot \mathbf{x_n} - \mathbf{x_n} \cdot \mathbf{x} + \mathbf{x_n} \cdot \mathbf{x_n} \\
&= \mathbf{x} \cdot \mathbf{x} - 2\mathbf{x} \cdot \mathbf{x_n} + \mathbf{x_n} \cdot \mathbf{x_n} \\
&= K(\mathbf{x}, \mathbf{x}) - 2K(\mathbf{x}, \mathbf{x_n}) + K(\mathbf{x_n}, \mathbf{x_n}) \qquad \textbf{(1)}
\end{aligned}
$$

Now $K(\mathbf{x_i}, \mathbf{x_j})$ in the above expression can be substituted by an arbitrary nonlinear function, such as:

$$
\begin{aligned}
K(\mathbf{x_i}, \mathbf{x_j}) &= (1 + \mathbf{x_i} \cdot \mathbf{x_j})^p \\
K(\mathbf{x_i}, \mathbf{x_j}) &= \exp\left(-\frac{\|\mathbf{x_i} - \mathbf{x_j}\|^2}{\sigma^2}\right) \\
K(\mathbf{x_i}, \mathbf{x_j}) &= \tanh(\alpha \mathbf{x_i} \cdot \mathbf{x_j} + \beta)
\end{aligned}
$$

Therefore, **(1)** is the formulated nearest-neighbour classifier for a general nonlinear kernel.

Problem 3

Express the middle factor as a power series as:

$$\begin{aligned}
\exp\left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}'}{\sigma^2}\right) &= \sum_{n=0}^{\infty}\left(\frac{\left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}'}{\sigma^2}\right)^n}{n!}\right) \\
&= \sum_{n=0}^{\infty}\frac{(\mathbf{x}^{\mathrm{T}}\mathbf{x}')^n}{\sigma^{2n}n!} \\
&= \sum_{n=0}^{\infty}\phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x'})
\end{aligned}$$

So the middle factor is expressed as the inner product of an infinite-dimensional feature vector, now we substitute this middle part back into the expanded Gaussian kernel:

$$\begin{aligned}
k(\mathbf{x},\mathbf{x'}) &= \exp\left(-\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}}{2\sigma^2}\right)\exp\left(\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}'}{\sigma^2}\right)\exp\left(-\frac{\mathbf{x'}^{\mathrm{T}}\mathbf{x}'}{2\sigma^2}\right) \\
&= \exp\left(-\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}}{2\sigma^2}\right)\sum_{n=0}^{\infty}\phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x'})\exp\left(-\frac{\mathbf{x'}^{\mathrm{T}}\mathbf{x}'}{2\sigma^2}\right) \\
&= \sum_{n=0}^{\infty}\exp\left(-\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}}{2\sigma^2}\right)\phi(\mathbf{x})^{\mathrm{T}}\phi(\mathbf{x'})\exp\left(-\frac{\mathbf{x'}^{\mathrm{T}}\mathbf{x}'}{2\sigma^2}\right) \\
&= \sum_{n=0}^{\infty}\left[\exp\left(-\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}}{2\sigma^2}\right)\phi(\mathbf{x})\right]^{\mathrm{T}}\left[\exp\left(-\frac{\mathbf{x'}^{\mathrm{T}}\mathbf{x}'}{2\sigma^2}\right)\phi(\mathbf{x'})\right] \\
&= \sum_{n=0}^{\infty}\psi(\mathbf{x})^{\mathrm{T}}\psi(\mathbf{x'})
\end{aligned}$$

So, as shown $\exists\ \psi(\mathbf{x}) = \exp\left(-\frac{\mathbf{x}^{\mathrm{T}}\mathbf{x}}{2\sigma^2}\right)\phi(\mathbf{x})$, the Gaussian Kernel can be expressed as the inner product of an infinite-dimensional vector.