AI5213/EC4213: Machine Learning and Deep Learning **Student ID: 20195024**
Homework 3 (Resampling, Model Selection) **Name: Kim, Minjun**

Due: Sunday, Oct 15, 11:59PM

This homework comprises a set of conceptual problems and one coding exercise. Some problems are trivial, while others will require a lot of thought. Start this homework early!

> Guideline for those new to data analysis using Python:
> We recommend you to review the Lab section within each chapter (e.g., p. 215 of Ch. 5 or https://islp.readthedocs.io/en/latest/labs/Ch05-resample-lab.html) prior to tackling the programming tasks. Additionally, find datasets and Jupyter notebooks at https://github.com/intro-stat-learning/ISLP_labs/ and https://islp.readthedocs.io/en/latest/.
>
> Visit https://www.statlearning.com/forum — a dedicated forum created by and for the ISL community. Whether you have a question or encounter issues with ISLP labs, this platform is your go-to resource for assistance and collaborative discussions.

Deliverables:

1. Submit a PDF of your homework, with an appendix listing all your code, to the Gradescope assignment entitled "HW3 Write-Up". You may typeset your homework in LaTeX or Word or submit neatly handwritten and scanned solutions. Please start each question on a new page. If there are graphs, include those graphs in the correct sections. Do not put them in an appendix. We need each solution to be self-contained on pages of its own.

   • On the first page of your write-up, please sign your signature next to the following statement. (Mac Preview, PDF Expert, and Foxit PDF Reader, among others, have tools to let you sign a PDF file.) We want to make extra clear the consequences of cheating.

   "I certify that all solutions are entirely in my own words and that I have not looked at another student's solutions. I have given credit to all external sources I consulted."

   • On the first page of your write-up, please list students who helped you or whom you helped on the homework. (Note that sending each other code is not allowed.)

2. Submit all the code needed to reproduce your results to the Gradescope assignment entitled "HW3 Code". You must submit your code twice: once in your PDF write-up (above) so the readers can easily read it, and again in compilable/interpretable form so the readers can easily run it. Do NOT include any data files we provided. Please include a short file named README listing your name, student ID, and instructions on how to reproduce your results. Please take care that your code doesn't take up inordinate amounts of time or memory.

For staff use only

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Total
/ 8	/ 6	/ 9	/ 10	/ 9	/ 15	/ 13	/ 30	/ 100

Honor Code

Declare and sign the following statement:

"I certify that all solutions in this document are entirely my own and that I have not looked at anyone else's solution. I have given credit to all external sources I consulted."

Signature:

We welcome group discussions, but the work you submit should be entirely your own. If you use any information or pictures not from our lectures or readings, make sure to say where they came from. Please note that breaking academic rules can lead to severe penalties.

(a) Did you receive any help whatsoever from anyone in solving this assignment? If your answer is 'yes', give full details (e.g. "Junho explained to me what is asked in Q2-a")

(b) Did you give any help whatsoever to anyone in solving this assignment? If your answer is 'yes', give full details (e.g. "I pointed Josh to Ch. 2.3 since he didn't know how to proceed with Q2")

(c) Did you find or come across code that implements any part of this assignment? If your answer is 'yes', give full details (book & page, URL & location within the page, etc.).

I learned how to use bootstrap function in python through Lab session in "Introduction to statistical learning with python", from p220 to p223. And this knowledge was used in Q7.

I referred to https://forecastegy.com/posts/time-series-cross-validation-python/ when studying the preprocessing of time series data while working on the Kaggle project. And that knowledge was used in Q6(c).

Q1. Statistical properties of the variance [8 pts]

Prove that $\alpha = \dfrac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$ mimimizes $\mathrm{Var}(\alpha X + (1-\alpha)Y)$, where $\sigma_X^2 = \mathrm{Var}(X)$, $\sigma_Y^2 = \mathrm{Var}(Y)$, and $\sigma_{XY} = \mathrm{Cov}(X, Y)$.

$$\mathrm{Var}(\alpha X + (1-\alpha)Y) = \mathrm{Var}(\alpha X) + \mathrm{Var}((1-\alpha)Y) + 2\cdot \mathrm{Cov}(\alpha X, (1-\alpha)Y)$$

$$= \alpha^2 \mathrm{Var}(X) + (1-\alpha)^2 \mathrm{Var}(Y) + 2\alpha(1-\alpha)\mathrm{Cov}(X,Y)$$

$$= \alpha^2 \sigma_X^2 + (1-\alpha)^2 \sigma_Y^2 + 2\alpha(1-\alpha)\sigma_{XY}$$

$$\frac{\partial}{\partial \alpha}\mathrm{Var}(\alpha X + (1-\alpha)Y) = 2\alpha\sigma_X^2 - 2(1-\alpha)\sigma_Y^2 + (2-4\alpha)\sigma_{XY}$$

Let's find α that makes $\dfrac{\partial}{\partial \alpha}\mathrm{Var}(\alpha X + (1-\alpha)Y) = 0$.

$$\alpha(2\sigma_X^2 + 2\sigma_Y^2 - 4\sigma_{XY}) + (-2\sigma_Y^2 + 2\sigma_{XY}) = 0.$$

$$\therefore \alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}.$$

Let's compute $\dfrac{\partial^2}{\partial \alpha^2}\mathrm{Var}(\alpha X + (1-\alpha)Y)$ to check if the point when $\alpha = \dfrac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$ is maximum of minimum.

$$\frac{\partial^2}{\partial \alpha^2}\mathrm{Var}(\alpha X + (1-\alpha)Y) = 2\sigma_X^2 + 2\sigma_Y^2 - 4\sigma_{XY} = 2\mathrm{Var}(X-Y) \geq 0.$$

$$\therefore \alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - \sigma_{XY}} \text{ minimizes } \mathrm{Var}(\alpha X + (1-\alpha)Y).$$

Q2. Best subset selection vs. Forward stepwise selection [6 pts]

What could be the reasons that the fourth model in best subset selection and forward stepwise selection differ as shown in the table below?

# Variables	Best subset selection	Forward stepwise selection
One	rating	rating
Two	rating, income	rating, income
Three	rating, income, student	rating, income, student
Four	**cards, income, student, limit**	**rating, income, student, limit**

Let M_k as a model that contains k predictors (variables).

① For the best subset selection case, we don't consider M_{k-1} to generate M_k.

That is, when we find M_k, we find every combination of k predictors and pick the best among those all models that contains exactly k predictors

So, if we see the table above, 'rating' is in M_3 but not in M_4.
And 'cards' and 'limit' are in M_4, which are not in M_3.
It is clear, because we find the single 'best' model by just considering the smallest RSS or largest R^2 when we choose M_k among many model containing exactly k predictors.

② Unlike the best subset selection, for the forward stepwise selection, we consider M_{k-1} to generate M_k. This means that M_k contains every predictors in M_{k-1} and add one additional predictor. By comparing models, of which the only difference is just one additional predictor, we find best M_k having smallest RSS or highest R^2 among them.

Therefore, M_4 have all predictors that M_3 have : 'rating, income, student', and one new additional predictor : 'limit'.

In a nutshell, the reasons that the fourth models in best subset selection and forward stepwise selection are different is one is a selection that does not consider the predictors of the third model and the other is a selection that consider.

Q3. Subset selection [9 pts]

We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain $p + 1$ models, containing $0, 1, 2, \cdots, p$ predictors. Explain your answers:

(a) Which of the three models with k predictors has the smallest *training* RSS? [2 pts]

Best subset selection model with k predictors would have the smallest training RSS. This is because for best subset selection, we choose M_k with the smallest RSS by comparing with "all" models which contain exactly k predictors. However, forward stepwise selection consider (k-1) predictors in M_{k-1} to make M_k, and backward stepwise selection consider (k+1) predictors in M_{k+1} and eliminate one predictor to make M_k. So they consider smaller subset having k predictors than the best subset selection model.

(b) Which of the three models with k predictors has the smallest *test* RSS? [2 pts]

Best subset model probably have the smallest test RSS since it consider all subsets of k-predictors, but that's not always the case. What I mean is that if k and p is large, the best subset selection model may overfit the train set. So it would show bad performance on new test data.

Therefore, we need more information about data to know what model have the smallest test RSS.

(c) True or False [5 pts; 1 pts each]:

i. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the $(k + 1)$-variable model identified by forward stepwise selection.

True. M_{k+1} contains all k predictors in M_k and one additional predictor for the forward stepwise selection

ii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the $(k + 1)$-variable model identified by backward stepwise selection.

True, when we make M_k, we eliminate one predictor among k+1 predictors in M_{k+1} for the backward stepwise selection.

iii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the $(k + 1)$-variable model identified by forward stepwise selection.

False, there is no relationship between them.

iv. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the $(k + 1)$-variable model identified by backward stepwise selection.

False, there is no relationship between them.

v. The predictors in the k-variable model identified by best subset are a subset of the predictors in the $(k + 1)$-variable model identified by best subset selection.

False, it could be subset in some special cases, but when we find the M_k, best single model, we compare all models containing exactly k predictors. So, M_k and M_{k+1} is not relevant. They are independently computed.

Q4. Regression with regularization [10 pts; 2 pts each]

Suppose we estimate the regression coefficients in a linear regression model by minimizing

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 \text{ subject to } \sum_{j=1}^{p}|\beta_j| \leq s$$

for a particular value of s. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

(a) As we increase s from 0, the training RSS will:

i. Increase initially, and then eventually start decreasing in an inverted U shape.
ii. Decrease initially, and then eventually start increasing in a U shape.
iii. Steadily increase.
(iv.) Steadily decrease.
v. Remain constant.

iv. Steadly decrease.

As s increases, restrictions on B_j are relaxed, so the model become more flexible. Therefore, it will fit train data closely as s increases, then RSS decreases.

(b) Repeat (a) for test RSS.

ii. Decrease initially, and then eventually start increasing in a U shape.

At first, as s increases, the model became more flexible and test RSS decreases.

If the model became too flexible, then it will overfit the train data and test RSS will increase.

(c) Repeat (a) for variance.

iii. steadily increase

As s increases, restrictions on B_j are relaxed, it will more closely fit train data. Then, flexibility of model and variance will steadily increase.

(d) Repeat (a) for (squared) bias.

iv. steadily decrease

As s increases, restrictions on B_j are relaxed and we get flexible model. Since, the model is closely fit the train data, the bias also become lower continuously.

(e) Repeat (a) for the irreducible error.

V. remain constant

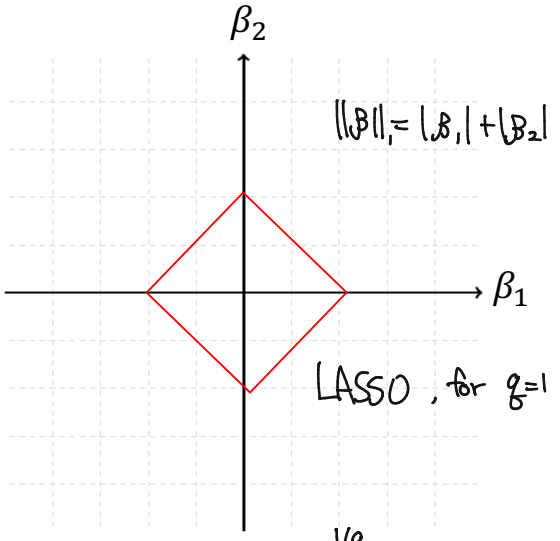
Irreducible error is not relevant to the model's flexibility. It would remain constant regardless of s.

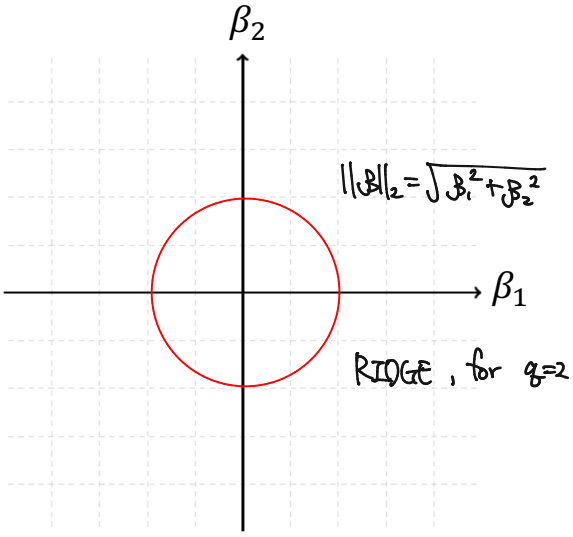
Q5. Drawing contours [9 pts]

In *lasso* and *ridge* regression, the term $\sum_j |\beta_j|^q$, is known as the L_q norm (or quasi-norm when $0 < q < 1$) of the vector β. This term is used in various regularization techniques in statistics and machine learning. When visualized in two dimensions (i.e., considering only β_1 and β_2), the lines of equal value (contours) for this regularization term assume different shapes based on the q value. Sketch how these contours transform with varying values of q.

[Note] Do not use any software.

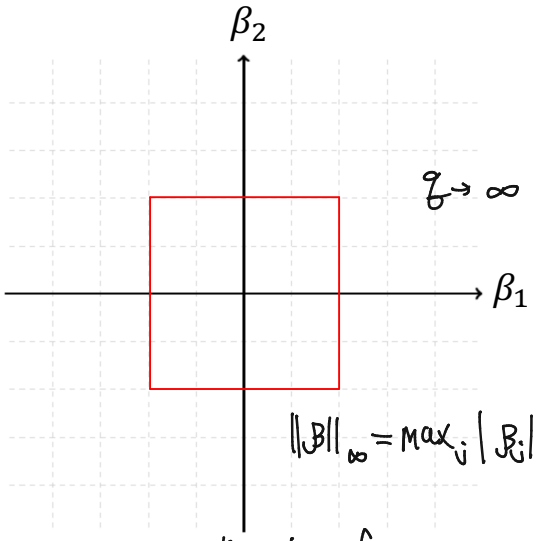
(a) Draw contours when q=1 (lasso) and q=2 (ridge), respectively. [3 pts]



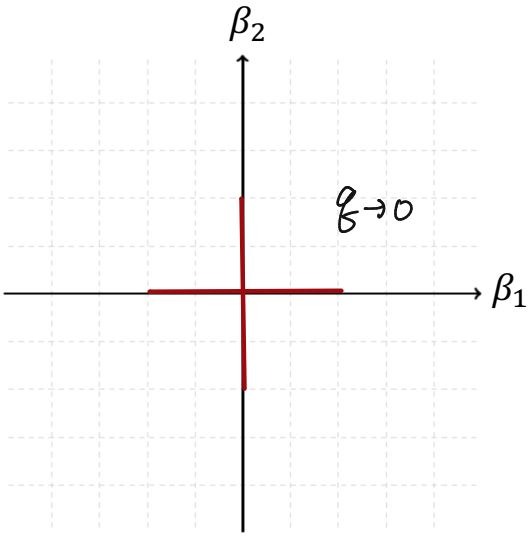


$\|\beta\|_q = \left(\sum_j |\beta_j|^q\right)^{1/q}$. we only consider only β_1 and β_2, since it's 2D space.

(b) How the contour change when $q \to \infty$? and $q \to 0$? [6 pts]



L_∞ norm gives the box forms.

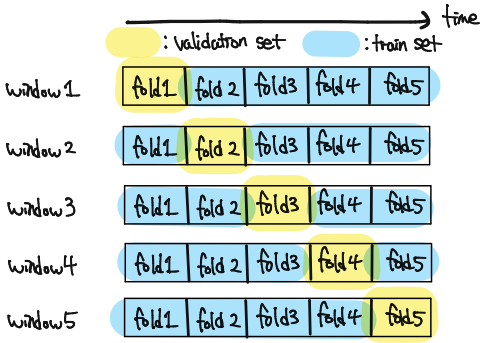


L_0 doesn't have a apparent geometrical shape.

Q6. Thoughts on k-fold cross-validation [15 pts]

Consider the challenges of using k-fold cross-validation for time series or temporal data. Why might standard k-fold cross-validation be inappropriate in this context?

(a) How does the temporal ordering of data points in a time series differ from the assumptions made by standard k-fold cross-validation regarding data independence? [3 pts]



In temporal data, for example oil price varying every day, the data points are correlated with their past or future data. So, if we just take validation set randomly using k-fold cross-validation like the picture on the left, it will cause problems.

Standard k-fold cross-validation assumes that the data points are independent of each other and well-distributed. This fact make choosing a randomly devided 'fold' for validation set appropriate. But in temporal data, datas are usually dependent with each other.

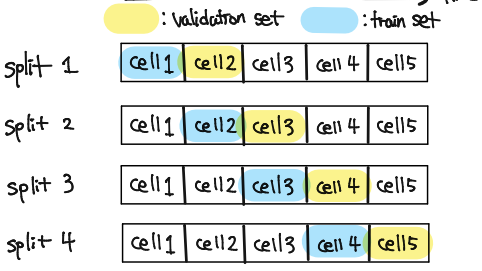
(b) If we were to use standard k-fold cross-validation on a time series dataset, what potential problems might arise during the training and testing phases? [3 pts]

For example, if we see 'window 2', train set might already contains information of validation set (fold2), as future data is influenced by past data. So using validation set would be not meaningful, because it will probably return very optimistic evaluation.

Therefore, we cannot choose random folds to become train or test set, because predicting past values by fitting future values are useless.

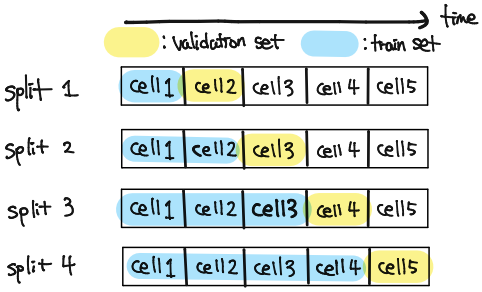
(c) What modifications or alternative cross-validation techniques could be employed to ensure that the temporal structure of the data is preserved during the validation process? [3 pts]

① Slicing Window Validation



In slicing window validation, the size of window are fixed. And it always ensure the order of train and validation data, past for train, future for validation. Therefore, the temporal structure is considered and preserved.

② Expanding Window Validation



In expanding window validation, training window becomes bigger by expanding. But this method also ensures the order of train and validation set in window for each split.

(d) Can you think of a practical scenario in which forecasting future outcomes based on historical data is essential? In your chosen scenario, describe how employing standard k-fold cross-validation could lead to unreliable model predictions. What specific challenges might arise from not respecting the temporal order of the data? [6 pts]

I think predicting oil prices is practical scenario that historical data is essential. Oil prices fluctuate continuously over time, with previous prices potentially impacting subsequent prices based on temporal dependencies. There

① Since k-fold cross-validation disregards the temporal order while making each 'window' and divide into train set and validation set, it potentially use future oil price in model training. This can reflect an unrealistic scenario, thereby model would learn patterns improperly. And also, when it use future data(recent data, relatively future data compared to validation set) in training set, the model can overfit to the training set, which leads to the serious problem.

② If we neglect the seasonal pattern by using k-fold cross-validation, for example oil demand increased especially winter, the model's performance would be low.

③ k-fold cross-validation cannot reflect the big events in real-world, for example, COVID issues, Russo-Ukrainian War, which can significantly impact on subsequent data point.

Q7. Bootstrap [13 pts]

We will now consider the Boston housing data set, from the ISLP library.

(a) Based on this data set, provide an estimate for the population mean of medv data. Call this estimate $\hat{\mu}$. [1 pts]

(a) Let's compute $\hat{\mu}$, an estimate for the population mean of medv data

```
medv_mean = Boston['medv'].mean()
print(f"An Estimate for the Population Mean of medv data: {medv_mean:.5f}")
```

[65] ✓ 0.0s Python

```
An Estimate for the Population Mean of medv data: 22.53281
```

(b) Provide an estimate of the standard error of $\hat{\mu}$. Interpret this result. [2 pts]

Hint: We can compute the standard error of the sample mean by dividing the sample standard deviation by the square root of the number of observations.

(b) Let's compute an estimate of the standard error of $\hat{\mu}$,which is $SE(\overline{X}) = \frac{s}{\sqrt{n}}$

The standard error reflect how accurate an estimate of the mean is.
To be more specific, a smaller standard error indicates that the sample mean is closer to the population mean.
In this case, a value of 0.40886 so we can that the estimate of the mean is relatively accurate.

```
medv_se = Boston['medv'].std() / np.sqrt(Boston['medv'].count())
print(f"An Estimate of the Standard Error of mu hat: {medv_se:.5f}")
```

[90] ✓ 0.0s Python

```
An Estimate of the Standard Error of mu hat: 0.40886
```

(c) Now estimate the standard error of $\hat{\mu}$ using the bootstrap. How does this compare to your answer from (b)? [2 pts]

(c) Let's estimate the standard error of $\hat{\mu}$ using boostrap

First, this generates bootstrap samples from the original data D, the Boston data.
And I applied a function sample_mean to each sample, which returns mean of random samples.
Finally, the boot_SE returns an estimate of standard error of mu hat by calculating the standard error from the results.

So, an estimate of standard error of $\hat{\mu}$ by using boostrap is bigger than those gained from (b).
This is because the bootstrap make samples randomly from the original data, which can lead to additional variability.

```python
def sample_mean(D, idx):
    return D['medv'].loc[idx].mean()

def boot_SE(func, D, n=None, B=1000, seed=0):
    rng = np.random.default_rng(seed)
    first_, second_ = 0, 0
    n = n or D.shape[0]
    for _ in range(B):
        idx = rng.choice(D.index, n, replace=True)
        value = func(D, idx)
        first_ += value
        second_ += value**2
    return np.sqrt(second_ / B - (first_ / B)**2)

medv_se_bootstrap = boot_SE(sample_mean, Boston, n=None, B=1000, seed=0)


print(f"An Estimate of the Standard Error of mu hat by using Bootstrap: {medv_se_bootstrap:.5f}")
print(f"An Estimate of the Standard Error of mu hat from (b): {medv_se:.5f}")
```

[110] ✓ 0.1s Python

```
An Estimate of the Standard Error of mu hat by using Bootstrap: 0.41253
An Estimate of the Standard Error of mu hat from (b): 0.40886
```

(d) Based on your bootstrap estimate from (c), provide a 95% confidence interval for the mean of medv. Compare it to the results obtained by using Boston['medv'].std() and the two standard error rule. [2 pts]

Hint: You can approximate a 95% confidence interval using the formula $[\hat{\mu} - 2SE(\hat{\mu}), \hat{\mu} + 2SE(\hat{\mu})]$.

Two standard error rule: For linear regression, the 95% confidence interval for β_1 approximately takes the form

$$\hat{\beta}_1 \pm 2 \cdot \text{SE}(\hat{\beta}_1). \tag{1}$$

(d) Let's provide 95% confidence interval for $\hat{\mu}$, the mean of medv.

The confidence intervals from both methods are quite alike, showing that the bootstrap method might be a good way to estimate confidence intervals for this data.
Also, because both intervals include $\hat{\mu} = 22.53281$, an estimate for the population mean of medv, it suggests that those intervals are meaningful.

```python
#95% confidence interval
def bootstrap_mean(func, D, n=None, B=1000, seed=0):
    rng = np.random.default_rng(seed)
    means = []
    n = n or D.shape[0]
    for _ in range(B):
        idx = rng.choice(D.index, n, replace=True)
        means.append(func(D, idx))
    return np.mean(means)

medv_mean_bootstrap = bootstrap_mean(sample_mean, Boston, n=None, B=1000, seed=0)

conf_interval_bootstrap = [medv_mean_bootstrap - 2*medv_se_bootstrap, medv_mean_bootstrap+ 2*medv_se_bootstrap]
conf_interval_by_std = [medv_mean - 2*medv_se, medv_mean + 2*medv_se]

print(f"95% Confidence Interval by using Bootstrap: [{conf_interval_bootstrap[0]:.5f}, {conf_interval_bootstrap[1]:.5f}]")
print(f"95% Confidence Interval by using Boston['medv'].std(): [{conf_interval_by_std[0]:.5f}, {conf_interval_by_std[1]:.5f}]")
```

[11] ✓ 0.1s Python

```
95% Confidence Interval by using Bootstrap: [21.72403, 23.37417]
95% Confidence Interval by using Boston['medv'].std(): [21.71508, 23.35053]
```

(e) Based on this data set, provide an estimate $\hat{\mu}_{med}$, for the median value of medv in the population. [1 pts]

(e) Let's estimate $\hat{\mu}_{med}$ for the median value of medv in the population

```python
medv_med = Boston['medv'].median()
print(f"An estimate of the median value of medv data: {medv_med:.5f}")
```

[98] ✓ 0.0s Python

```
An estimate of the median value of medv data: 21.20000
```

(f) We now would like to estimate the standard error of $\hat{\mu}_{med}$. Unfortunately, there is no simple formula for computing the standard error of the median. Instead, estimate the standard error of the median using the bootstrap. Comment on your findings. [2 pts]

(f) Let's estimate the standard error of $\hat{\mu}_{med}$ using boostrap

Since I already made boot_SE function previously for (c), all I need to do is just making a new definition of sample_median function that returns the median value of random samples.

I found that an estimate of the standard error of median $\hat{\mu}$ by using Bootstrap is 0.36945, which is significantly small.
This means that the sample median is likely near the median of the population.

```
def sample_median(D, idx):
    return D['medv'].loc[idx].median()

# Using boot_SE to estimate the standard error of the median.
medv_median_se_bootstrap = boot_SE(sample_median, Boston, n=None, B=1000, seed=0)

print(f"An Estimate of the Standard Error of median mu hat by using Bootstrap: {medv_median_se_bootstrap:.5f}")
```

[111] ✓ 0.1s Python

```
An Estimate of the Standard Error of median mu hat by using Bootstrap: 0.36945
```

(g) Based on this data set, provide an estimate for the tenth percentile of medv in Boston census tracts. Call this quantity $\hat{\mu}_{0.1}$. (You can use the np.percentile() function. [1 pts]

(g) Let's compute an estimate for $\hat{\mu}_{0.1}$, the tenth percentile of medv

```
medv_10th_percentile = np.percentile(Boston['medv'], 10)
print(f"An Estimate of the 10th Percentile of medv: {medv_10th_percentile:.5f}")
```

[105] ✓ 0.0s Python

```
An Estimate of the 10th percentile of medv: 12.75000
```

(h) Use the bootstrap to estimate the standard error of $\hat{\mu}_{0.1}$. Comment on your findings. [2 pts]

(h) Let's estimate the standard error of $\hat{\mu}_{0.1}$, the tenth percentile of medv

This is also similar to the problem (f), so all I need to do is just making a new definition of sample_10th_percentile function that returns the tenth percentile of medv for random samples.

After running this code, I found that an estimate of the standard error of $\hat{\mu}_{0.1}$ by using Bootstrap is 0.50345, which is significantly small.
The interpretation of this is silmilar to the previous problems, so this means that the 10th percentile of medv in our data set is close to the 10th percentile in the population.

```
def sample_10th_percentile(D, idx):
    return np.percentile(D['medv'].loc[idx], 10)

# Using boot_SE to estimate the standard error of the 10th percentile.
medv_10th_percentile_se_bootstrap = boot_SE(sample_10th_percentile, Boston, n=None, B=1000, seed=0)

print(f"An estimate of the Standard Error of the 10th Percentile using Bootstrap: {medv_10th_percentile_se_bootstrap:.5f}")
```

[109] ✓ 0.1s Python

```
An estimate of the Standard Error of the 10th Percentile using Bootstrap: 0.50345
```

Q8. Multi-variate linear regression with NYC taxi dataset [30 pts]

Please complete the exercises in the following Google Colab notebook: https://bit.ly/mldl23f-hw3-nyc-taxi and submit your .ipynb file.

I could not convert ipynb to pdf in Macbook. So I attached the html file instead.