Chap2-Exercises_LuisCorreia-745724_v4

August 30, 2025

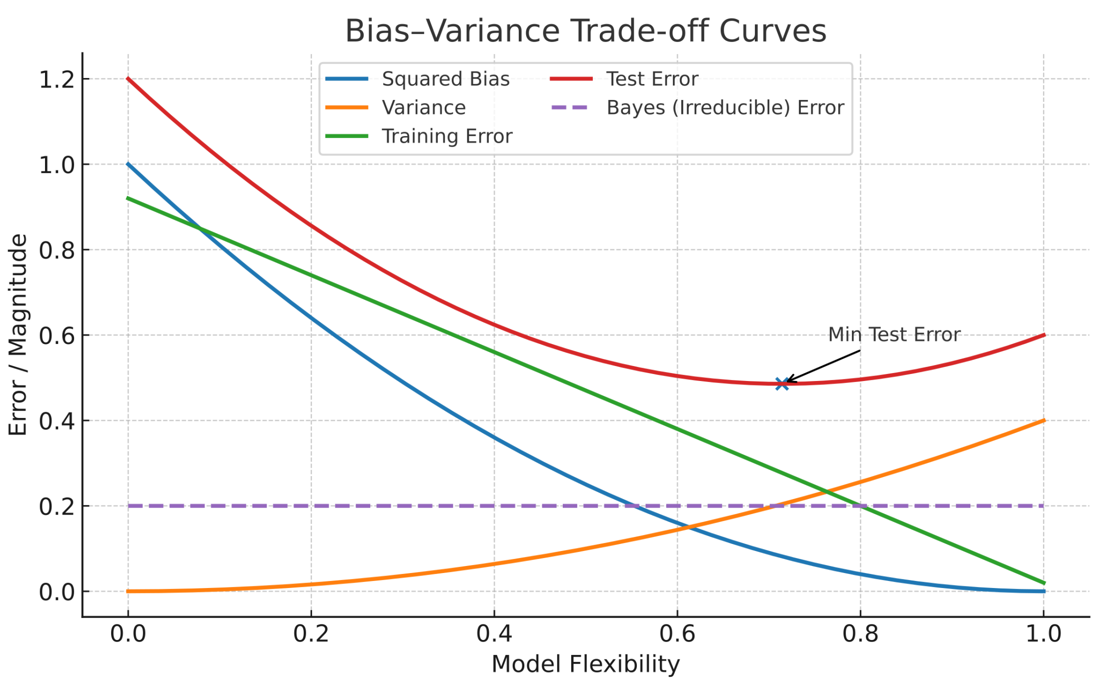
1 MAP5935 - Statistical Learning (Chapter 2 - Statistical Learning)

Prof. Christian Jäkel

https://www.statlearning.com/

1.1 Conceptual Questions

1.1.1 Question 3



(a) Provide a sketch of typical (squared) bias, variance, training error, test error, and Bayes (or irreducible) error curves, on a single plot, as we go from less flexible statistical learning methods towards more flexible approaches. The x-axis should represent the amount of flexibility in the method, and the y-axis should represent the values for each curve. There should be five curves. Make sure to label each one.

Bias–Variance Decomposition — Conceptual Solution *On one plot:* x-axis = **Model flexibility** (left = rigid, right = very flexible).
y-axis = **Error / magnitude**.

- **Squared Bias**: **monotonically decreasing** curve (starts high on the left, drops toward ~0 on the right).
- **Variance**: **monotonically increasing** curve (starts low on the left, rises on the right).
- **Training Error**: **monotonically decreasing** curve (typically very low near the most flexible end).
- **Test Error**: **U-shaped** curve (decreases at first—sweet spot—then increases as overfitting kicks in).
- **Bayes (Irreducible) Error**: **flat horizontal line** (constant across all flexibility).

Measuring Model Flexibility

The x-axis in the bias–variance tradeoff plot represents **model flexibility**.
This is not a universal numeric scale, but rather a measure of how complex or adaptable the model is to fit patterns in the data.
Different model families control flexibility in different ways, as summarized below:

Model Family	Flexibility Control Parameter(s)	Low Flexibility (Rigid)	High Flexibility (Flexible)
Polynomial Regression	Polynomial degree (d)	Small (d) (e.g., linear, quadratic) → high bias	Large (d) (e.g., 15th degree) → low bias, high variance
k-Nearest Neighbors	Number of neighbors (k)	Large (k): smoother predictions, underfitting	Small (k): very wiggly, may overfit noise
Decision Trees	Tree depth / number of leaves	Shallow tree: underfits, high bias	Deep tree: fits training data closely, high variance
Regularized Models	Regularization strength ( ) (Ridge, Lasso)	Large ( ): strong shrinkage, simple model	Small ( ): weak shrinkage, complex model
Neural Networks	# Layers, # Units, Regularization	Few layers/units, strong regularization	Many layers/units, weak regularization → highly flexible

Key idea:
- Moving **left → right** on the x-axis means allowing the model more capacity to adapt to the data.
- More flexibility reduces **bias** but increases **variance**.
- The optimal point balances this trade-off, minimizing **test error**.

(b) Explain why each of the five curves has the shape displayed in part (a). **Key Ideas from the Bias–Variance Tradeoff Graph**

- **Squared Bias** decreases as model flexibility increases:
  Rigid models cannot capture complex patterns (high bias), but more flexible models approximate the true function better.

- **Variance** increases with flexibility:
  Flexible models adapt strongly to the training data, making them sensitive to noise and

sample variation.

- **Training Error** always decreases with flexibility:
  More flexible models fit the training data better, often driving training error close to zero.
- **Test Error** follows a U-shape:
  - Initially, increasing flexibility reduces error (bias reduction dominates).
  - After a point, variance dominates and test error rises due to overfitting.
- **Bayes (Irreducible) Error** is constant:
  Represents inherent noise in the data that no model can reduce.

In Summary:
- The best generalization is achieved near the minimum of the **test error curve**, where bias and variance are balanced.

1.1.2 Question 6

Describe the differences between a parametric and a non-parametric statistical learning approach. What are the advantages of a parametric approach to regression or classification (as opposed to a nonparametric approach)? What are its disadvantages?

Parametric Approaches

- **Definition:** Assume a specific functional form for the relationship between predictors and response.
  Example: Linear regression assumes $Y \approx \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon$.
- **Process:**
  1. Specify the form of the model (e.g., linear, logistic).
  2. Estimate a finite set of parameters (e.g., coefficients ( )) from the data.

Non-Parametric Approaches

- **Definition:** Do not assume a predetermined functional form.
  Instead, they allow the data to determine the model's shape.
  Example: k-Nearest Neighbors (kNN), decision trees, splines, kernel methods.
- **Process:**
  - The model complexity grows with the size of the dataset.
  - The method adapts more flexibly to complex, nonlinear patterns.

Advantages of Parametric Approaches

- **Simplicity & Interpretability:** Results are easy to understand and explain (e.g., linear coefficients).

- **Efficiency with Small Data:** Require estimating only a few parameters, so they perform well when the sample size is limited.

- **Faster Computation:** Training and prediction are typically computationally inexpensive.

- **Less Risk of Overfitting (with correct form):** When the assumed form is approximately correct, parametric models generalize well.

Disadvantages of Parametric Approaches

- **Model Misspecification Risk:** If the true relationship deviates from the assumed form, the model may suffer from high bias.

- **Limited Flexibility:** They struggle to capture highly nonlinear or complex structures in the data.

- **Rigid Assumptions:** Performance depends heavily on whether assumptions (linearity, normality, etc.) hold.

Summary:
- **Parametric models**: simple, interpretable, and efficient with small data, but prone to bias if the functional form is wrong.
- **Non-parametric models**: more flexible and data-driven, but require large sample sizes and can be harder to interpret.

1.2 Applied Questions

1.2.1 Question 8

This exercise relates to the College data set, which can be found in the file College.csv on the book website. It contains a number of variables for 777 different universities and colleges in the US. The variables are:

- **Private** : Public/private indicator
- **Apps** : Number of applications received
- **Accept** : Number of applicants accepted
- **Enroll** : Number of new students enrolled
- **Top10perc** : New students from top 10 % of high school class
- **Top25perc** : New students from top 25 % of high school class
- **F.Undergrad** : Number of full-time undergraduates
- **P.Undergrad** : Number of part-time undergraduates
- **Outstate** : Out-of-state tuition
- **Room.Board** : Room and board costs
- **Books** : Estimated book costs
- **Personal** : Estimated personal spending
- **PhD** : Percent of faculty with Ph.D.s
- **Terminal** : Percent of faculty with terminal degree
- **S.F.Ratio** : Student/faculty ratio

- **perc.alumni** : Percent of alumni who donate
- **Expend** : Instructional expenditure per student
- **Grad.Rate** : Graduation rate

(a) Use the ***pd.read_csv()*** *function to read the data into Python. Call the loaded data college. Make sure that you have the directory set to the correct location for the data.*

[2]:
```python
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
college = pd.read_csv("../Data/College.csv")

# Show basic structure
print(college.shape)
college.head()
```

```
(777, 19)
```

[2]:
```
                     Unnamed: 0 Private  Apps  Accept  Enroll  Top10perc  \
0  Abilene Christian University     Yes  1660    1232     721         23
1            Adelphi University     Yes  2186    1924     512         16
2                Adrian College     Yes  1428    1097     336         22
3           Agnes Scott College     Yes   417     349     137         60
4     Alaska Pacific University     Yes   193     146      55         16

   Top25perc  F.Undergrad  P.Undergrad  Outstate  Room.Board  Books  Personal  \
0         52         2885          537      7440        3300    450      2200
1         29         2683         1227     12280        6450    750      1500
2         50         1036           99     11250        3750    400      1165
3         89          510           63     12960        5450    450       875
4         44          249          869      7560        4120    800      1500

   PhD  Terminal  S.F.Ratio  perc.alumni  Expend  Grad.Rate
0   70        78       18.1           12    7041         60
1   29        30       12.2           16   10527         56
2   53        66       12.9           30    8735         54
3   92        97        7.7           37   19016         59
4   76        72       11.9            2   10922         15
```

(b) Look at the data used in the notebook by creating and running a new cell with just the code college in it. You should notice that the first column is just the name of each university in a column named something like Unnamed: 0. We don't really want pandas to treat this as data. However, it may be handy to have these names for later. Try the following commands and similarly look at the resulting data frames:

[3]:
```python
college2 = pd.read_csv('..\\Data\\College.csv', index_col =0)
college3 = college.rename({'Unnamed: 0': 'College'},
axis=1)
college3 = college3.set_index('College')
```

This has used the first column in the file as an index for the data frame. This means that pandas has given each row a name corresponding to the appropriate university. Now you should see that the first data column is Private. Note that the names of the colleges appear on the left of the table. We also introduced a new python object above: a dictionary, which is specified by dictionary (key, value) pairs. Keep your modified version of the data with the following:

[4]:
```python
college = college3
```

(c) Use the `describe()` *method of to produce a numerical summary of the variables in the data set.*

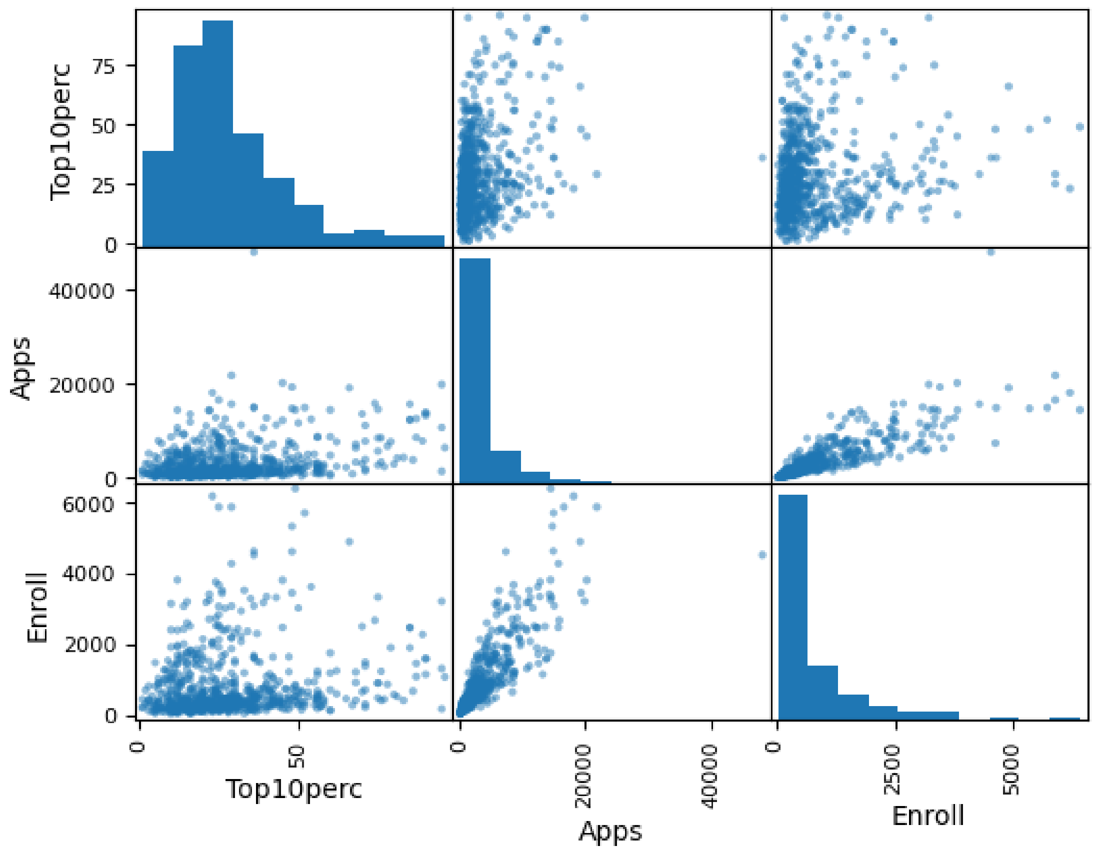
[5]:
```python
college.describe()
```

[5]:
```
               Apps        Accept       Enroll   Top10perc   Top25perc  \
count    777.000000    777.000000   777.000000  777.000000  777.000000
mean    3001.638353   2018.804376   779.972973   27.558559   55.796654
std     3870.201484   2451.113971   929.176190   17.640364   19.804778
min       81.000000     72.000000    35.000000    1.000000    9.000000
25%      776.000000    604.000000   242.000000   15.000000   41.000000
50%     1558.000000   1110.000000   434.000000   23.000000   54.000000
75%     3624.000000   2424.000000   902.000000   35.000000   69.000000
max    48094.000000  26330.000000  6392.000000   96.000000  100.000000

        F.Undergrad   P.Undergrad      Outstate   Room.Board        Books  \
count    777.000000    777.000000    777.000000   777.000000   777.000000
mean    3699.907336    855.298584  10440.669241  4357.526384   549.380952
std     4850.420531   1522.431887   4023.016484  1096.696416   165.105360
min      139.000000      1.000000   2340.000000  1780.000000    96.000000
25%      992.000000     95.000000   7320.000000  3597.000000   470.000000
50%     1707.000000    353.000000   9990.000000  4200.000000   500.000000
75%     4005.000000    967.000000  12925.000000  5050.000000   600.000000
max    31643.000000  21836.000000  21700.000000  8124.000000  2340.000000

          Personal         PhD    Terminal   S.F.Ratio  perc.alumni  \
count   777.000000  777.000000  777.000000  777.000000   777.000000
mean   1340.642214   72.660232   79.702703   14.089704    22.743887
std     677.071454   16.328155   14.722359    3.958349    12.391801
min     250.000000    8.000000   24.000000    2.500000     0.000000
25%     850.000000   62.000000   71.000000   11.500000    13.000000
50%    1200.000000   75.000000   82.000000   13.600000    21.000000
75%    1700.000000   85.000000   92.000000   16.500000    31.000000
max    6800.000000  103.000000  100.000000   39.800000    64.000000

             Expend   Grad.Rate
```

```
count     777.000000  777.00000
mean     9660.171171   65.46332
std      5221.768440   17.17771
min      3186.000000   10.00000
25%      6751.000000   53.00000
50%      8377.000000   65.00000
75%     10830.000000   78.00000
max     56233.000000  118.00000
```

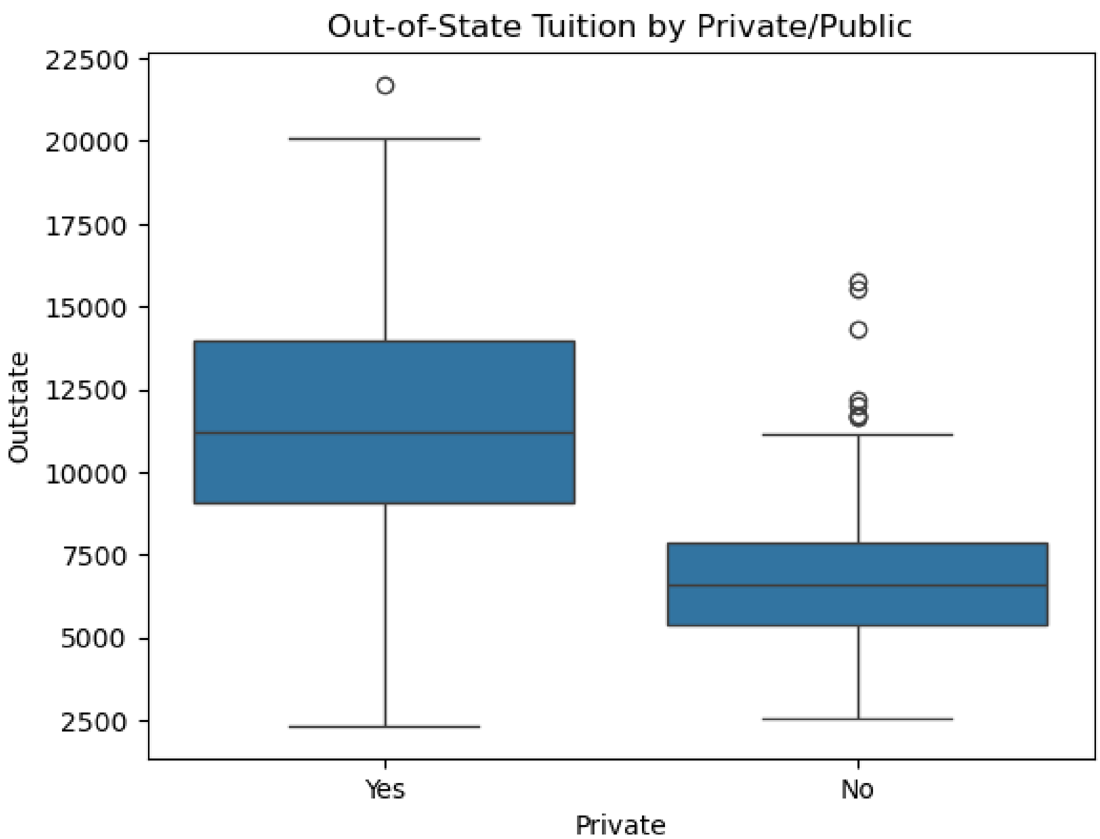
(d) Use the ***pd.plotting.scatter_matrix()*** *function to produce a scatterplot matrix of the first columns [****Top10perc****,* ***Apps****,* ***Enroll****]. Recall that you can reference a list C of columns of a data frame A using A[C].*

[6]:
```python
_ = pd.plotting.scatter_matrix(college[['Top10perc', 'Apps', 'Enroll']])
```



(e) Use the ***boxplot()*** *method of college to produce side-by-side boxplots of Outstate versus Private.*

[7]:
```python
sns.boxplot(x="Private", y="Outstate", data=college)
plt.title("Out-of-State Tuition by Private/Public")
plt.show()
```



(f) Create a new qualitative variable, called Elite, by binning the Top10perc variable into two groups based on whether or not the proportion of students coming from the top 10% of their high school classes exceeds 50%.

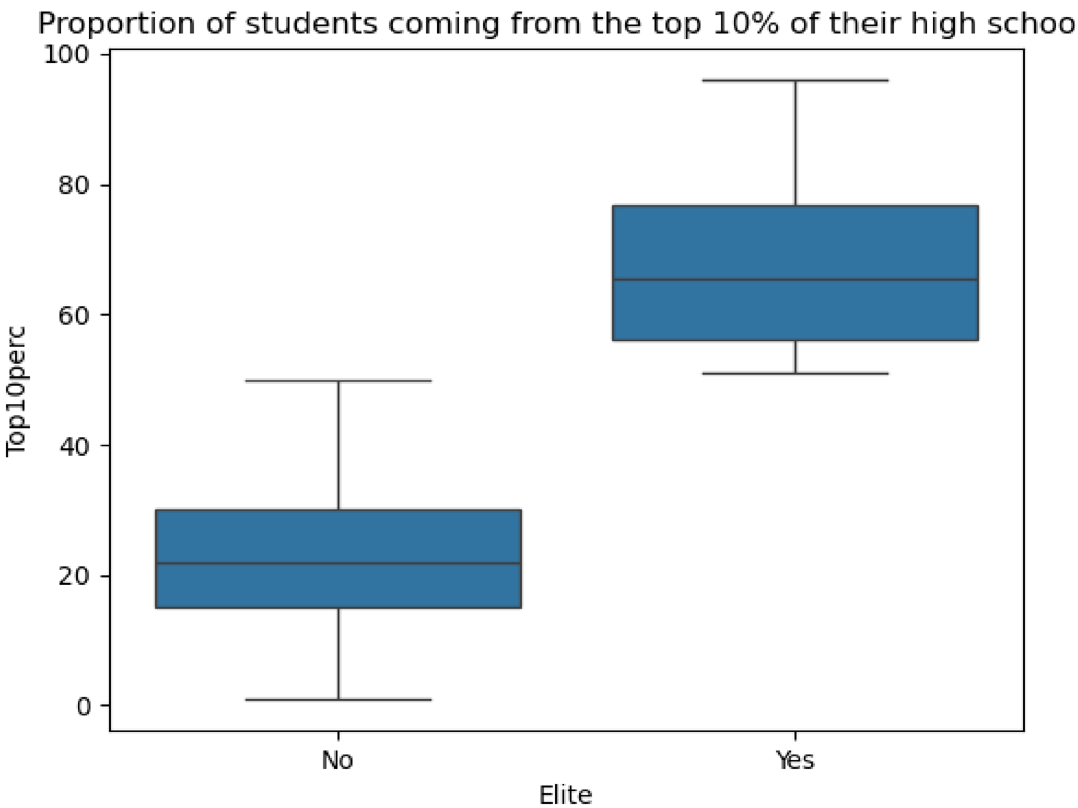
[8]:
```
# pd.cut bins (splits) a numeric variable into intervals, doing (0, 50] → "No"
↪and (50, 100] → "Yes"

college['Elite'] = pd.cut(college['Top10perc'], [0,50,100], labels=['No',
↪'Yes'])
```

Use the `value_counts()` method of `college['Elite']` to see how many elite universities there are. Finally, use the `boxplot()` method again to produce side-by-side boxplots of Outstate versus Elite.

[9]:
```
# Frequency table (counts)
print(college['Elite'].value_counts())
```

```
Elite
No     699
Yes     78
Name: count, dtype: int64
```

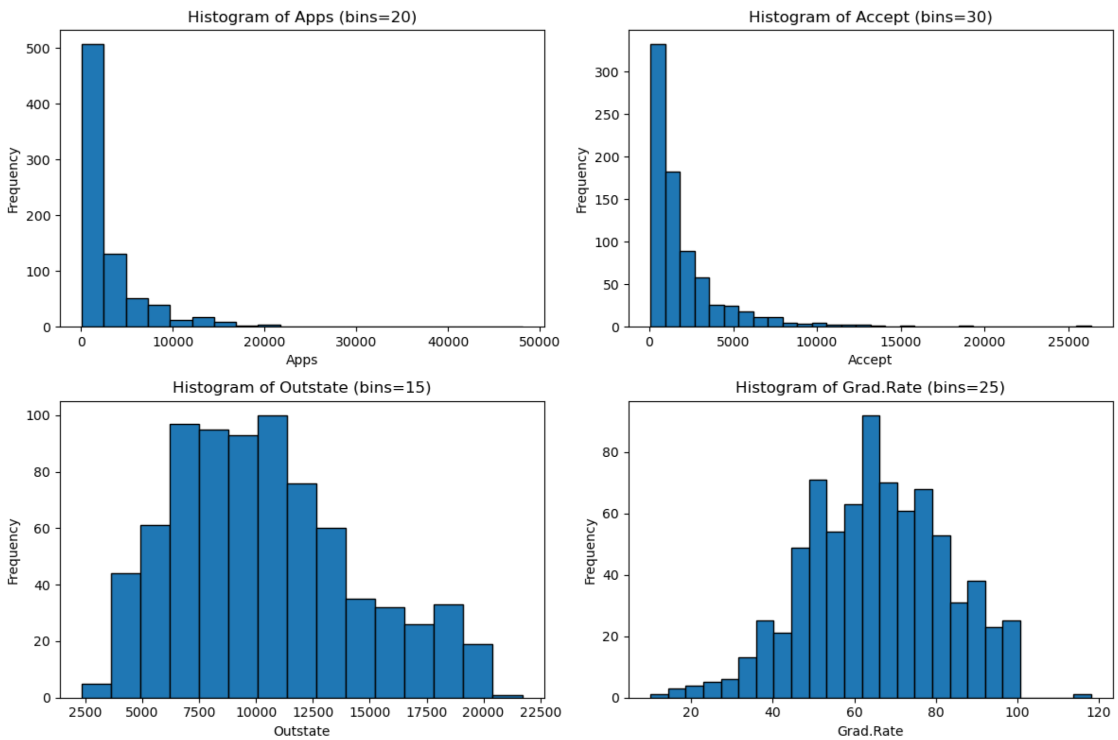
[10]:
```
sns.boxplot(x="Elite", y="Top10perc", data=college)
plt.title("Proportion of students coming from the top 10% of their high school␣
↪")
plt.show()
```



(g) Use the ***plot.hist()*** *method of college to produce some histograms with differing numbers of bins for a few of the quantitative variables. The command* ***plt.subplots(2, 2)*** *may be useful: it will divide the plot window into four regions so that four plots can be made simultaneously. By changing the arguments you can divide the screen up in other combinations.*

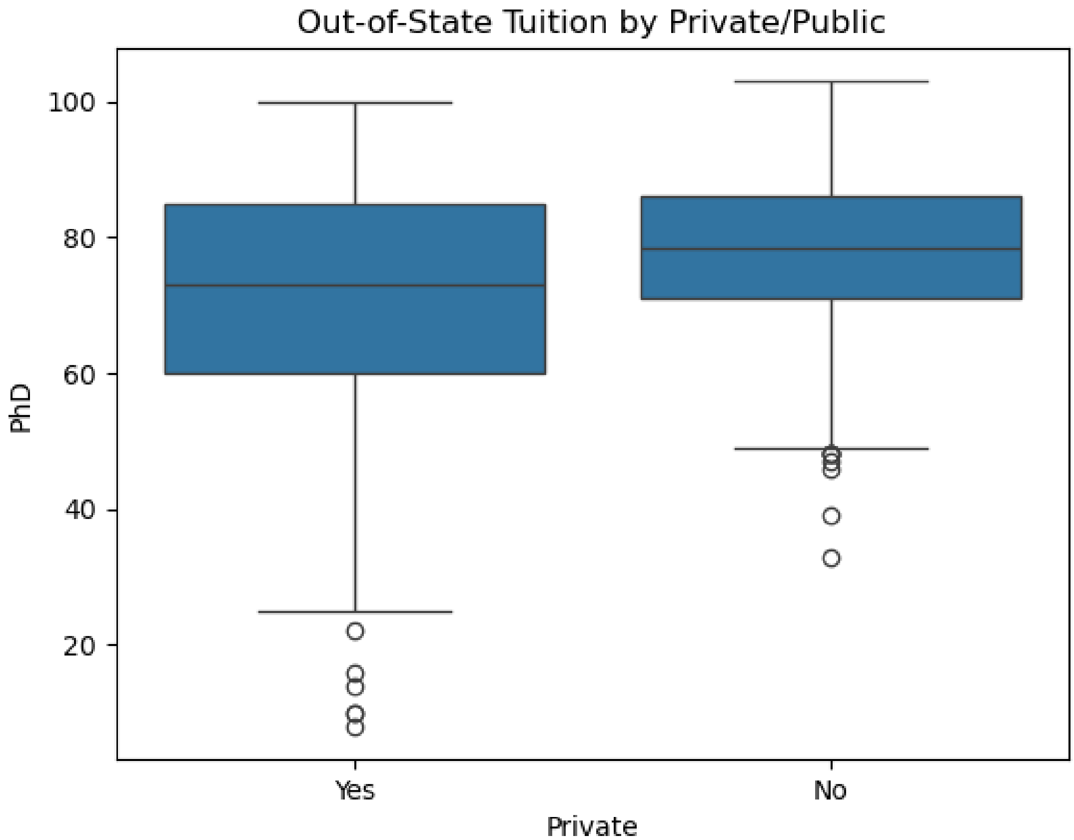
[11]:
```
# Choose 4 quantitative variables to explore
vars_to_plot = ['Apps', 'Accept', 'Outstate', 'Grad.Rate']

# Create a 2x2 subplot grid
fig, axes = plt.subplots(2, 2, figsize=(12, 8))

# Loop through variables and plot histograms with different bin sizes
# Method: .ravel() flattens the array into 1D.

for ax, var, bins in zip(axes.ravel(), vars_to_plot, [20, 30, 15, 25]):
    college[var].plot.hist(bins=bins, ax=ax, edgecolor='black')
```

```
    ax.set_title(f"Histogram of {var} (bins={bins})")
    ax.set_xlabel(var)

plt.tight_layout()
plt.show()
```



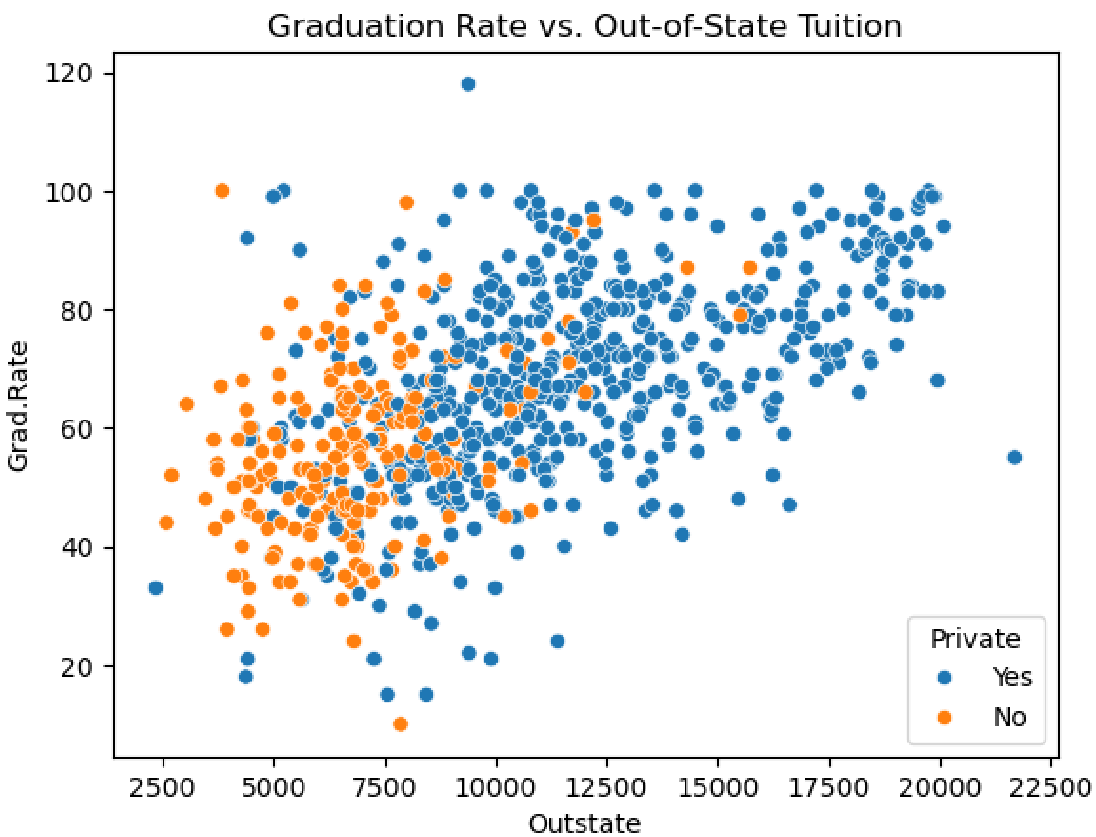
(h) Continue exploring the data, and provide a brief summary of what you discover.

[12]:
```
# Percentage of PhD per Public and Private institutions
sns.boxplot(x="Private", y="PhD", data=college)
plt.title("Out-of-State Tuition by Private/Public")
plt.show()
```



- **Both sectors**: high PhD levels
- **Public**: higher median, less spread
- **Private**: wider variation, some very low outliers

[13]:
```
# Outstate Tuition vs. Graduation Rate (Scatterplot)
# Check if Schools with higher tuition may have better resources and student␣
↪support → possibly higher graduation rates.
sns.scatterplot(x="Outstate", y="Grad.Rate", hue="Private", data=college)
plt.title("Graduation Rate vs. Out-of-State Tuition")
plt.show()
```

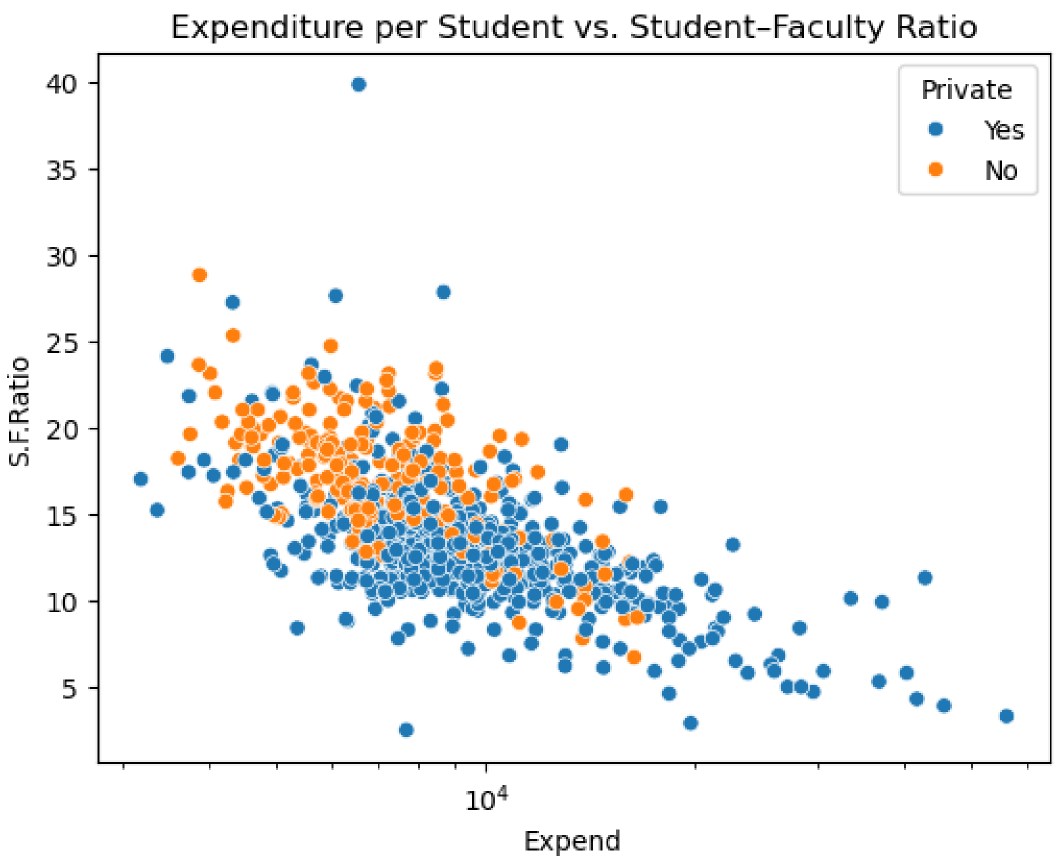


- There is a **positive relationship** between out-of-state tuition and graduation rate: colleges with higher tuition generally have higher graduation rates.
- **Private institutions (blue)** cluster at higher tuition levels and often show higher graduation rates compared to public ones.
- **Public institutions (orange)** tend to concentrate in the lower tuition range (roughly below \$8,000), with graduation rates more widely spread.
- A few schools show **unusual graduation rates above 100%**, likely data entry or reporting anomalies in the dataset.
- Overall, tuition appears to be an important differentiator: private colleges charge more and, on average, achieve higher graduation rates than public colleges.

- **Positive relation**: higher tuition → higher grad rate
- **Private schools**: higher tuition, higher grad rates
- **Public schools**: lower tuition, wider grad rate spread
- **Outliers**: some grad rates > 100%

[14]:
```
# Expenditure per Student vs. Student-Faculty Ratio (Scatterplot, maybe
↪log-scale for Expend)
```

```
# The Student-Faculty Ratio (S.F.Ratio) is the number of students per faculty␣
↪member at an institution.
# Check if Instructional expenditure (Expend) should relate to teaching␣
↪resources; schools that spend more might have smaller student-faculty ratios

sns.scatterplot(x="Expend", y="S.F.Ratio", hue="Private", data=college)
plt.xscale("log")  # log-scale often helps since Expend is skewed
plt.title("Expenditure per Student vs. Student-Faculty Ratio")
plt.show()
```



- Clear negative relationship: higher instructional expenditures per student are associated with lower student–faculty ratios.
- Private schools (blue) generally spend more per student and achieve lower ratios, reflecting smaller class sizes.
- Public schools (orange) cluster at lower expenditures and higher ratios, meaning larger class sizes.
- A few outliers exist, with unusually high expenditure but still moderate ratios.

- **Higher expend $\rightarrow$ lower S.F. ratio**
- **Private**: higher spending, smaller ratios
- **Public**: lower spending, larger ratios
- **Some high-expend outliers**