A Python Companion to ISLR

Naresh Gurbuxani

May 24, 2019

Contents

1 Introduction

Figure 1 shows graphs of Wage versus three variables.

Figure 2 shows boxplots of previous days' percentage changes in S&P 500 grouped according to today's change `Up` or `Down`.

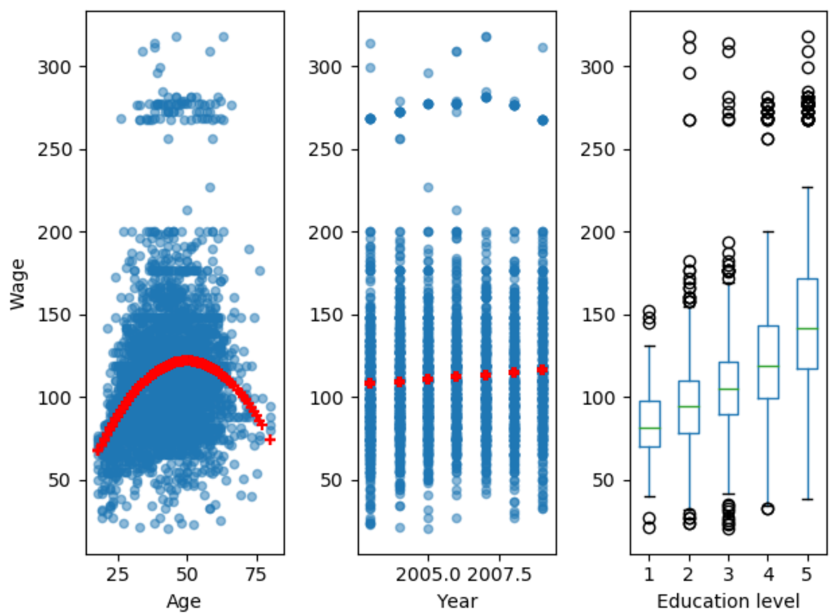

Figure 1: `Wage` data, which contains income survey information for males from the central Atlantic region of the United States. Left: `wage` as a function of `age`. On average, `wage` increases with `age` until about 60 years of age, at which point it begins to decline. Center: `wage` as a function of `year`. There is a slow but steady increase of approximately $10,000 in the average `wage` between 2003 and 2009. Right: Boxplots displaying `wage` as a function of `education`, with 1 indicating the lowest level (no highschool diploma) and 5 the highest level (an advanced graduate degree). On average, `wage` increases with the level of `education`.

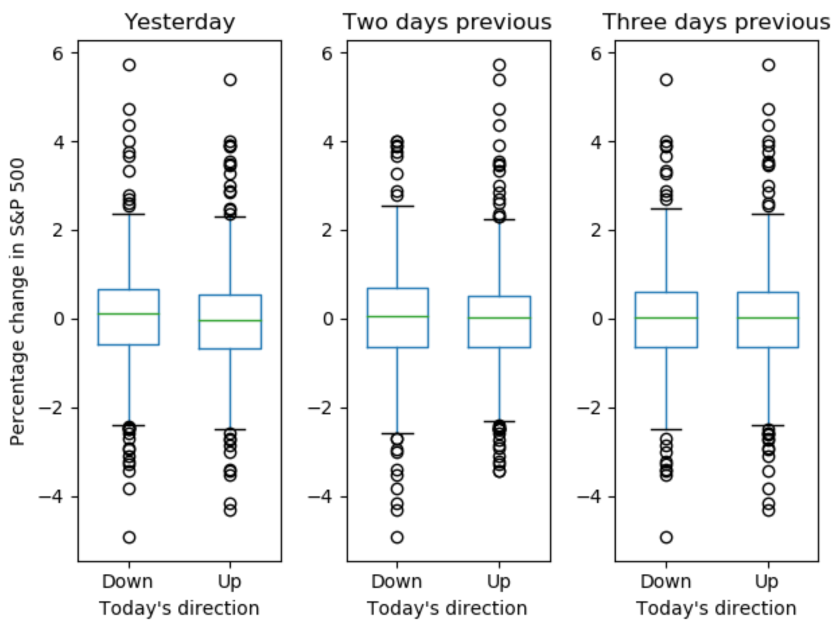

Figure 2: Left: Boxplots of the previous day's percentage change in the S&P 500 index for the days for which the market increased or decreased, obtained from the `Smarket` data. Center and Right: Same as left panel, but the percentage changes for two and three days previous are shown.

2 Statistical Learning

2.1 What is Statistical Learning?

Figure 3 shows scatter plots of `sales` versus `TV`, `radio`, and `newspaper` advertising. In each panel, the figure also includes an OLS regression line.

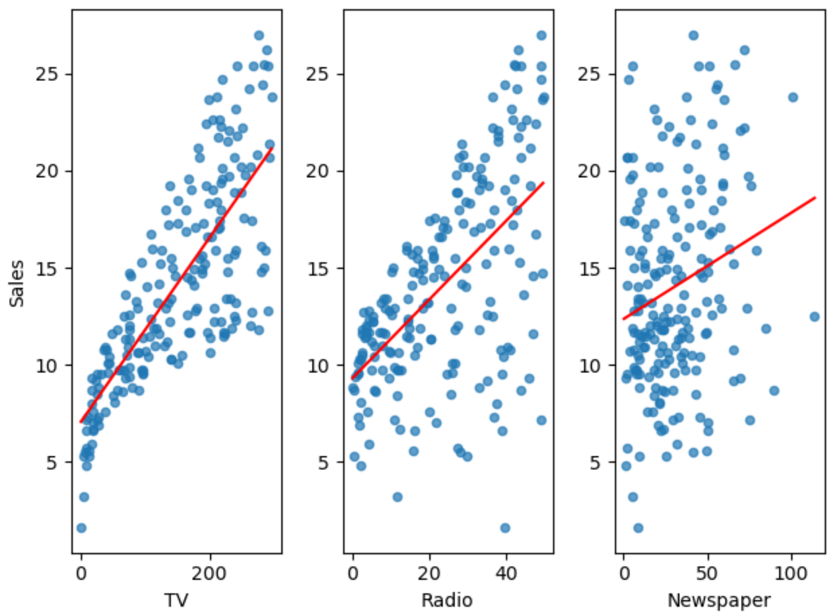

Figure 3: The `Advertising` data set. The plot displays `sales`, in thousands of units, as a function of `TV`, `radio`, and `newspaper` budgets, in thousands of dollars, for 200 different markets. In each plot we show the simple least squares fit of `sales` to that variable. In other words, each red line represents a simple model that can be used to predict `sales` using `TV`, `radio`, and `newspaper`, respectively.

Figure 4 is a plot of `Income` versus `Years of Education` from the Income data set. In the left panel, the "true" function (given by blue line) is actually my guess.

Figure 5 is a plot of `Income` versus `Years of Education` and `Seniority` from the `Income` data set. Since the book does not provide the true values of `Income`, "true" values shown in the plot are actually third order polynomial fit.

Figure 6 shows an example of the parametric approach applied to the

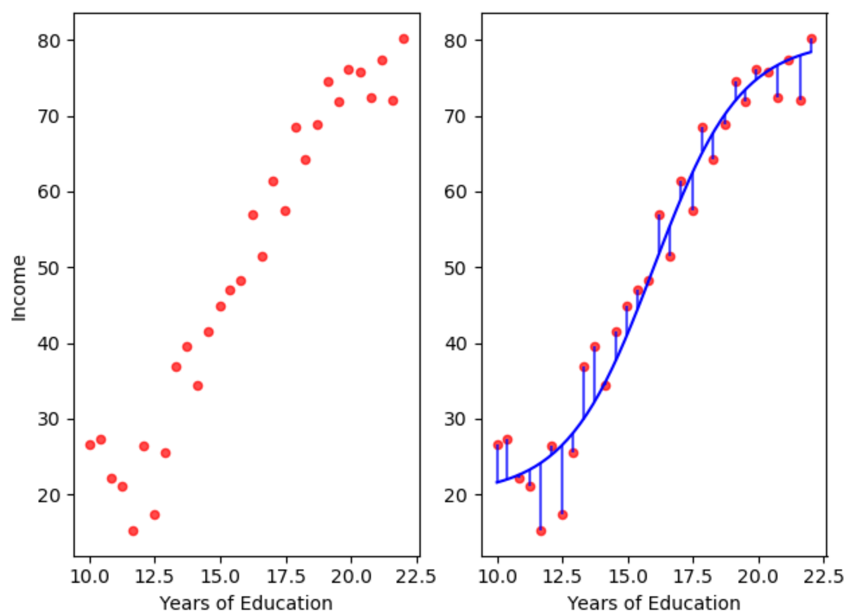

Figure 4: The `Income` data set. Left: The red dots are the observed values of `income` (in tens of thousands of dollars) and `years of education` for 30 individuals. Right: The blue curve represents the true underlying relationship between `income` and `years of education`, which is generally unknown (but is known in this case because the data are simulated). The vertical lines represent the error associated with each observation. Note that some of the errors are positive (when an observation lies above the blue curve) and some are negative (when an observation lies below the curve). Overall, these errors have approximately mean zero.

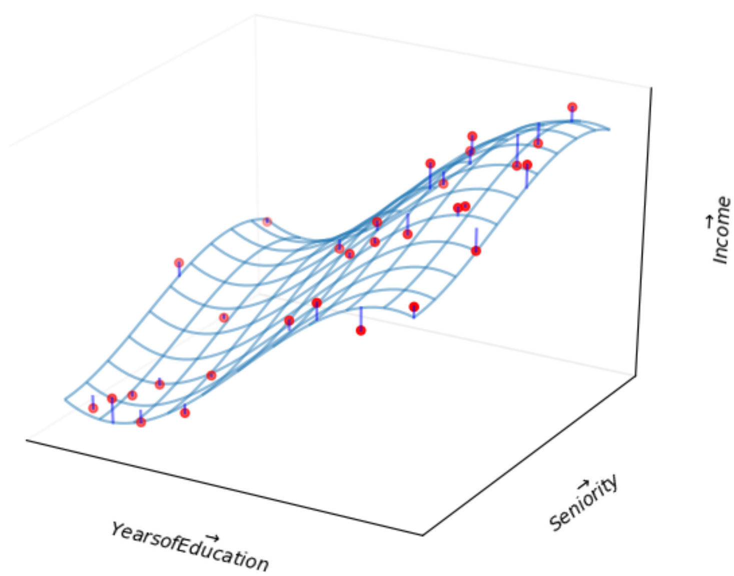

Figure 5: The plot displays `income` as a function of `years of education` and `seniority` in the `Income` data set. The blue surface represents the true underlying relationship between `income` and `years of education` and `seniority`, which is known since the data are simulated. The red dots indicate the observed values of these quantities for 30 individuals.

`Income` data from previous figure.

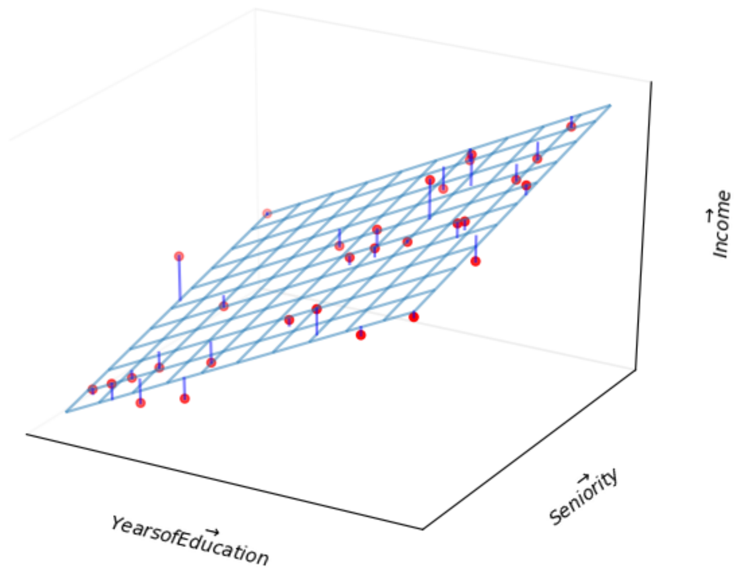

Figure 6: A linear model fit by least squares to the `Income` data from figure 5. The observations are shown in red, and the blue plane indicates the least squares fit to the data.

Figure 7 provides an illustration of the trade-off between flexibility and interpretability for some of the methods covered in this book.

Figure 8 provides a simple illustration of the clustering problem.

2.2 Assessing Model Accuracy

Figure 9 illustrates the tradeoff between training MSE and test MSE. We select a "true function" whose shape is similar to that shown in the book. In the left panel, the orange, blue, and green curves illustrate three possible estimates for f given by the black curve. The orange line is the linear regression fit, which is relatively inflexible. The blue and green curves were produced using *smoothing splines* from `UnivariateSpline` function in `scipy` package. We obtain different levels of flexibility by varying the parameter `s`, which affects the number of knots.

For the right panel, we have chosen polynomial fits. The degree of polynomial represents the level of flexibility. This is because the function

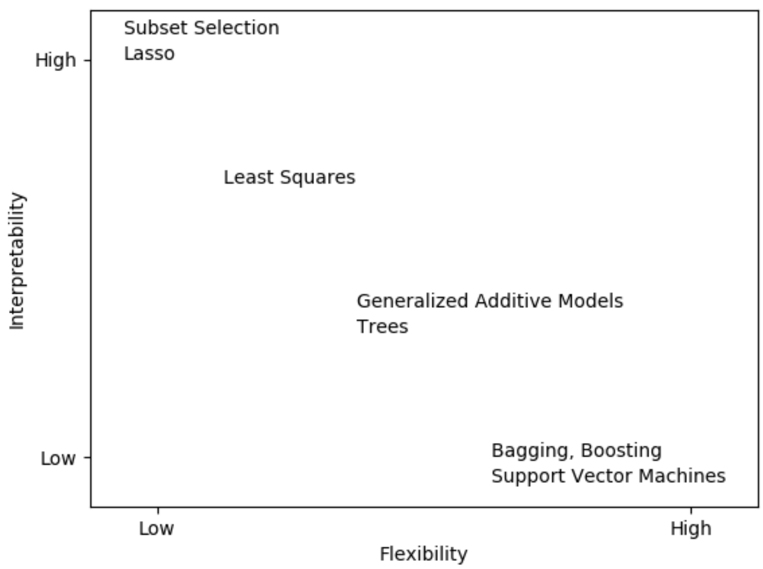

Figure 7: A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods. In general, as the flexibility of a method increases, its interpretability decreases.

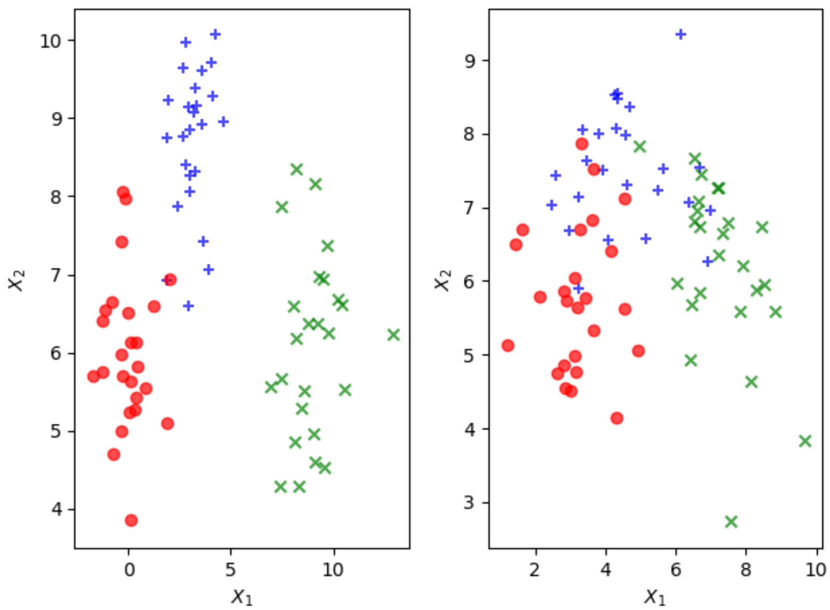

Figure 8: A clustering data set involving three groups. Each group is shown using a different colored symbol. Left: The three groups are well-separated. In this setting, a clustering approach should successfully identify the three groups. Right: There is some overlap among the groups. Now the clustering taks is more challenging.

`UnivariateSpline` does not more than five degrees of freedom.

When we repeat the simulations for figure 9, we see considerable variation in the right panel MSE plots. But the overall conclusion remains the same.

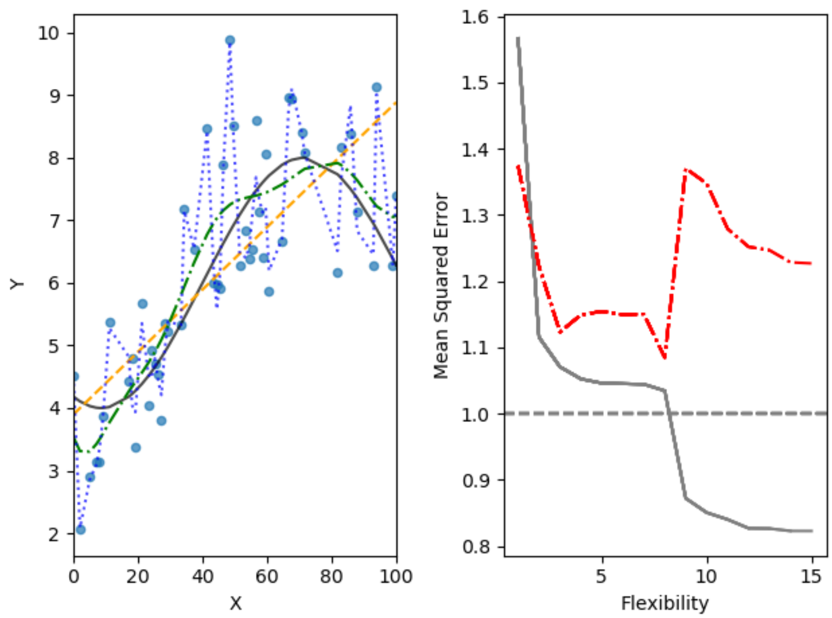

Figure 9: Left: Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline fits (blue and green curves). Right: Training MSE (grey curve), test MSE (red curve), and minimum possible test MSE over all methods (dashed grey line).

Figure 10 provides another example in which the true f is approximately linear.

Figure 11 displays an example in which f is highly non-linear. The training and test MSE curves still exhibit the same general patterns.

Figure 12 displays the relationship between bias, variance, and test MSE. This relationship is referred to as *bias-variance trade-off*. When simulations are repeated, we see considerable variation in different graphs, especially for MSE lines. But overall shape remains the same.

Figure 13 provides an example using a simulated data set in two-dimensional space consisting of predictors X_1 and X_2.

Figure 14 displays the KNN decision boundary, using $K = 10$, when

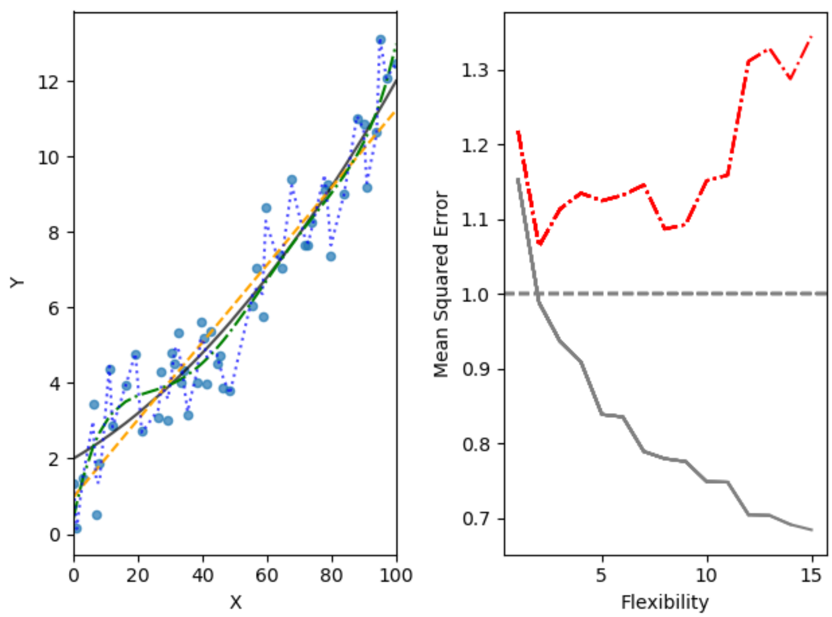

Figure 10: Details are as in figure 9 using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.

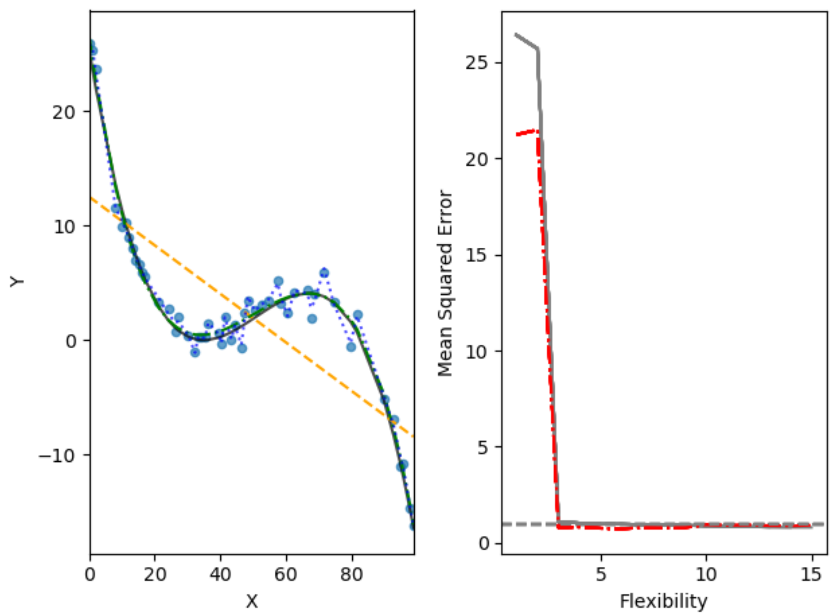

Figure 11: Details are as in figure 9, using a different f that is far from linear. In this setting, linear regression provides a very poor fit to the data.

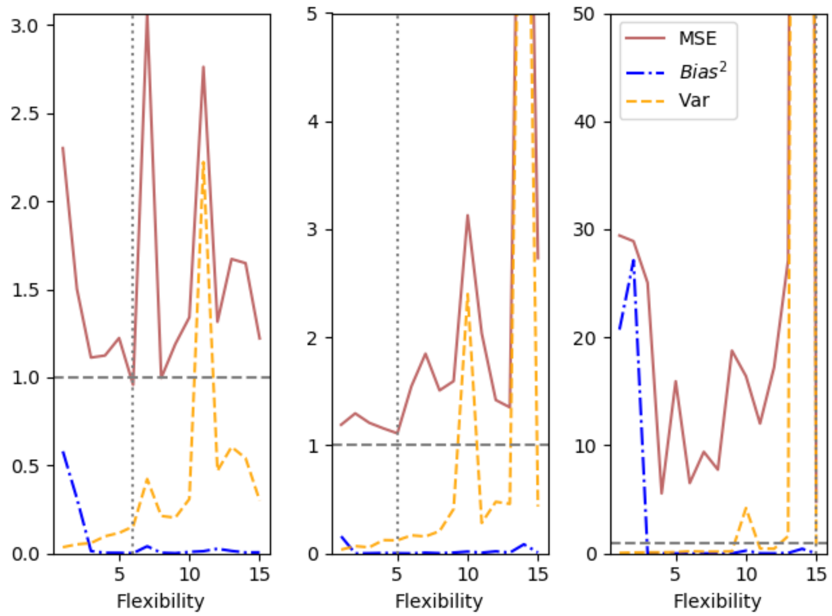

Figure 12: Squared bias (blue curve), variance (orange curve), $Var(\epsilon)$ (dashed line), and test MSE (red curve) for the three data sets in figures 9 - 11. The vertical dotted line indicates the flexibility level corresponding to the smallest test MSE.

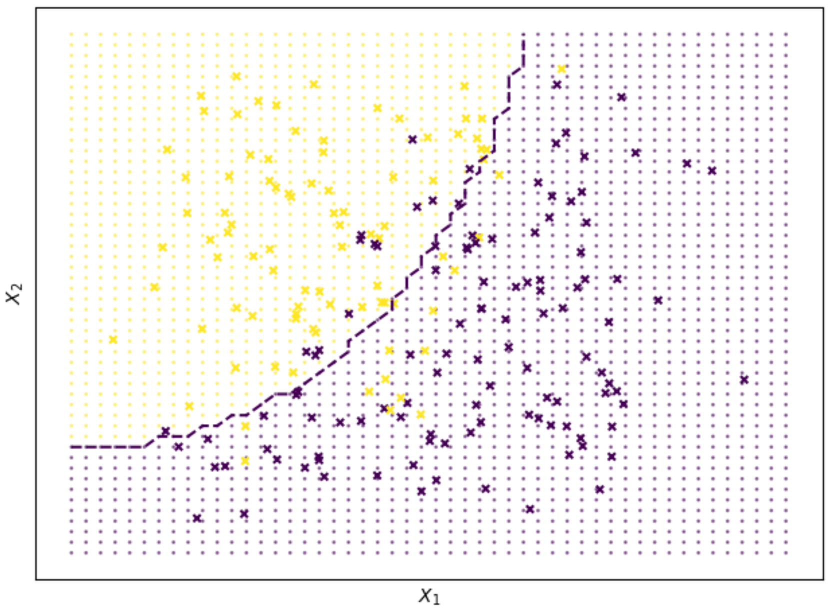

Figure 13: A simulated data set consisting of 200 observations in two groups, indicated in blue and orange. The dashed line represents the Bayes decision boundary. The orange background grid indicates the region in which a test observation will be assigned to the orange class, and blue background grid indicates the region in which a test observation will be assigned to the blue class.

applied to the simulated data set from figure 13. Even though the true distribution is not known by the KNN classifier, the KNN decision making boundary is very close to that of the Bayes classifier.

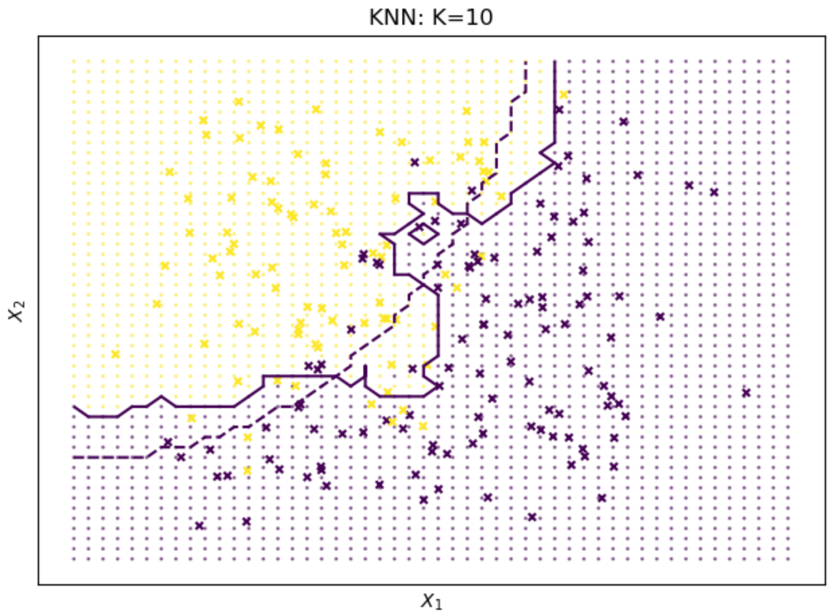

Figure 14: The firm line indicates the KNN decision boundary on the data from figure 13, using $K = 10$. The Bayes decision boundary is shown as a dashed line. The KNN and Bayes decision boundaries are very similar.

In figure 16 we have plotted the KNN test and training errors as a function of $\frac{1}{K}$. As $\frac{1}{K}$ increases, the method becomes more flexible. As in the regression setting, the training error rate consistently declines as the flexibility increases. However, the test error exhibits the characteristic U-shape, declining at first (with a minimum at approximately $K = 10$) before increasing again when the method becomes excessively flexible and overfits.

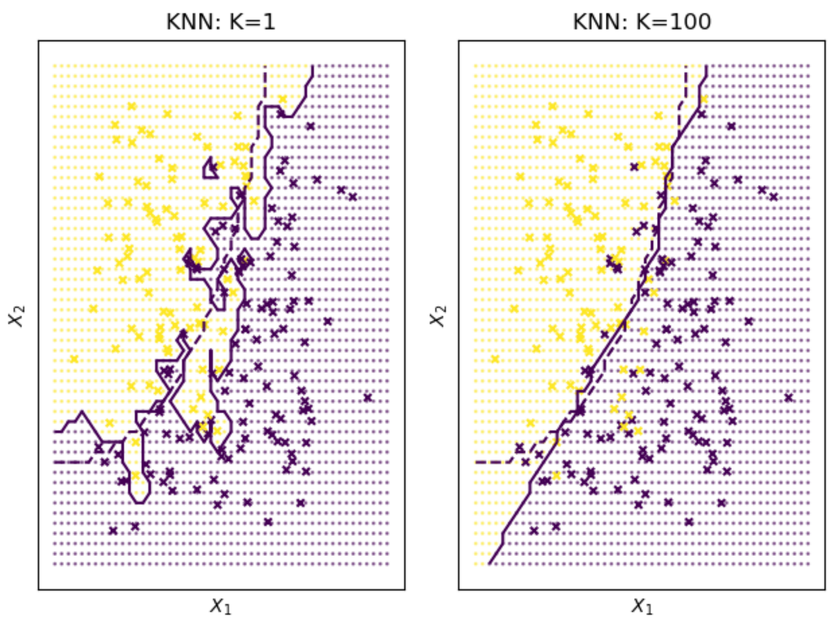

Figure 15: A comparison of the KNN decision boundaries (solid curves) obtained using $K = 1$ and $K = 100$ on the data from figure 13. With $K = 1$, the decision boundary is overly flexible, while with $K = 100$ it is not sufficiently flexible. The Bayes decision boundary is shown as dashed line.

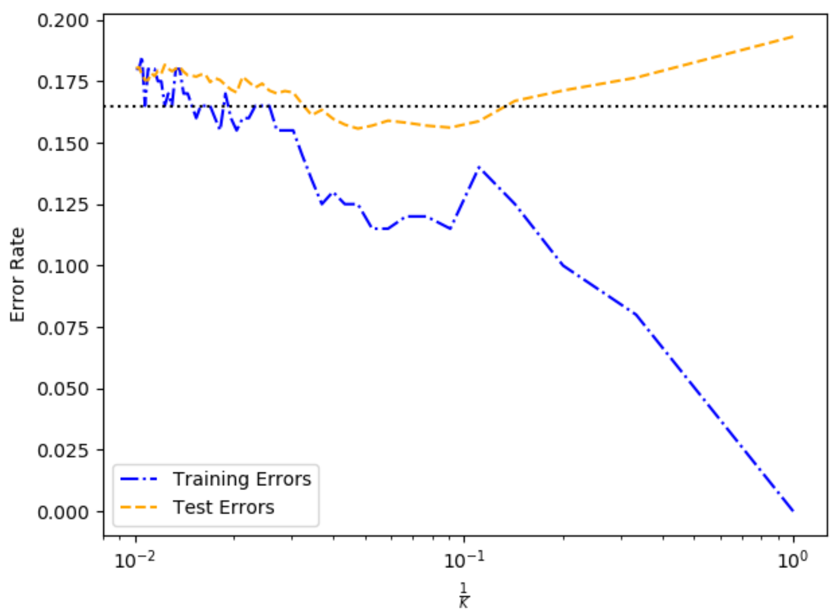

Figure 16: The KNN training error rate (blue, 200 observations) and test error rate (orange, 5,000 observations) on the data from figure 13 as the level of flexibility (assessed using $\frac{1}{K}$) increases, or equivalently as the number of neighbors K decreases. The black dashed line indicates the Bayes error rate.

2.3 Lab: Introduction to Python

2.3.1 Basic Commands

In `Python` a list can be created by enclosing comma-separated elements by square brackets. Length of a list can be obtained using `len` function.

```
x = [1, 3, 2, 5]
print(len(x))
y = 3
z = 5
print(y + z)
```

```
4
8
```

To create an array of numbers, use `array` function in `numpy` library. `numpy` functions can be used to perform element-wise operations on arrays.

```
import numpy as np
x = np.array([[1, 2], [3, 4]])
y = np.array([6, 7, 8, 9]).reshape((2, 2))
print(x)
print(y)
print(x ** 2)
print(np.sqrt(y))
```

```
[[1 2]
 [3 4]]
[[6 7]
 [8 9]]
[[ 1  4]
 [ 9 16]]
[[2.44948974 2.64575131]
 [2.82842712 3.        ]]
```

`numpy.random` has a number of functions to generate random variables that follow a given distribution. Here we create two correlated sets of numbers, `x` and `y`, and use `numpy.corrcoef` to calculate correlation between them.

```
import numpy as np
np.random.seed(911)
x = np.random.normal(size=50)
y = x + np.random.normal(loc=50, scale=0.1, size=50)
print(np.corrcoef(x, y))
print(np.corrcoef(x, y)[0, 1])
print(np.mean(x))
print(np.var(y))
print(np.std(y) ** 2)
```

```
[[1.         0.99374931]
 [0.99374931 1.        ]]
0.9937493134584551
-0.020219724397254404
0.9330621750073689
0.9330621750073688
```

2.3.2 Graphics

`matplotlib` library has a number of functions to plot data in `Python`. It is possible to view graphs on screen or save them in file for inclusion in a document.

```
import numpy as np
import matplotlib               # only if we need to save figure in file
matplotlib.use('Agg')           # only to save figure in file
import matplotlib.pyplot as plt

x = np.random.normal(size=100)
y = np.random.normal(size=100)
plt.plot(x, y)
plt.xlabel('This is x-axis')
plt.ylabel('This is y-axis')
plt.title('Plot of X vs Y')

plt.savefig('xyPlot.png')       # only to save figure in a file
```

`numpy` function `linspace` can be used to create a sequence between a start and an end of a given length.

```
import numpy as np
```

```
import matplotlib.pyplot as plt

x = np.linspace(-np.pi, np.pi, num=50)
y = x
xx, yy = np.meshgrid(x, y)
zz = np.cos(yy) / (1 + xx ** 2)

plt.contour(xx, yy, zz)

fig, ax = plt.subplots()
zza = (zz - zz.T) / 2.0
CS = ax.contour(xx, yy, zza)
ax.clabel(CS, inline=1)
```

2.3.3 Indexing Data

To access elements of an array, specify indexes inside square brackets. It is possible to access multiple rows and columns. `shape` method gives number of rows followed by number of columns.

```
import numpy as np

A = np.array(np.arange(1, 17))
A = A.reshape(4, 4, order='F')  # column first, Fortran style
print(A)
print(A[1, 2])
print(A[(0,2),:][:,(1,3)])
print(A[range(0,3),:][:,range(1,4)])
print(A[range(0, 2), :])
print(A[:, range(0, 2)])
print(A[0,:])
print(A.shape)
```

```
[[ 1  5  9 13]
 [ 2  6 10 14]
 [ 3  7 11 15]
 [ 4  8 12 16]]
10
[ 5 15]
[ 5 10 15]
[[ 1  5  9 13]
```

```
 [ 2  6 10 14]]
[[1 5]
 [2 6]
 [3 7]
 [4 8]]
(4, 4)
```

2.3.4 Loading Data

pandas library provides read_csv function to read files with data in rectangular shape.

```
import pandas as pd
Auto = pd.read_csv('data/Auto.csv')
print(Auto.head())
print(Auto.shape)
print(Auto.columns)
```

```
    mpg  cylinders  displacement  ... year  origin                       name
0  18.0          8         307.0  ...   70       1  chevrolet chevelle malibu
1  15.0          8         350.0  ...   70       1          buick skylark 320
2  18.0          8         318.0  ...   70       1         plymouth satellite
3  16.0          8         304.0  ...   70       1              amc rebel sst
4  17.0          8         302.0  ...   70       1                ford torino

[5 rows x 9 columns]
(397, 9)
Index(['mpg', 'cylinders', 'displacement', 'horsepower', 'weight',
       'acceleration', 'year', 'origin', 'name'],
      dtype='object')
```

To load data from an R library, use get_rdataset function from statsmodels. This function seems to work only if the computer is connected to the internet.

```
from statsmodels import datasets
carseats = datasets.get_rdataset('Carseats', package='ISLR').data
print(carseats.shape)
print(carseats.columns)
```

```
(400, 11)
Index(['Sales', 'CompPrice', 'Income', 'Advertising', 'Population', 'Price',
       'ShelveLoc', 'Age', 'Education', 'Urban', 'US'],
      dtype='object')
```

2.3.5 Additional Graphical and Numerical Summaries

`plot` method can be directly applied to a `pandas` dataframe.

```
import pandas as pd
Auto = pd.read_csv('data/Auto.csv')
Auto.boxplot(column='mpg', by='cylinders', grid=False)
```

`hist` method can be applied to plot a histogram.

```
import pandas as pd
Auto = pd.read_csv('data/Auto.csv')
Auto.hist(column='mpg')
Auto.hist(column='mpg', color='red')
Auto.hist(column='mpg', color='red', bins=15)
```

For pairs plot, use `scatter_matrix` method in `pandas.plotting`.

```
import pandas as pd
from pandas import plotting
Auto = pd.read_csv('data/Auto.csv')
plotting.scatter_matrix(Auto[['mpg', 'displacement', 'horsepower', 'weight',
      'acceleration']])
```

On `pandas` dataframes, `describe` method produces a summary of each variable.

```
import pandas as pd
Auto = pd.read_csv('data/Auto.csv')
print(Auto.describe())
```

```
              mpg   cylinders  ...        year      origin
count  397.000000  397.000000  ...  397.000000  397.000000
mean    23.515869    5.458438  ...   75.994962    1.574307
std      7.825804    1.701577  ...    3.690005    0.802549
min      9.000000    3.000000  ...   70.000000    1.000000
25%     17.500000    4.000000  ...   73.000000    1.000000
50%     23.000000    4.000000  ...   76.000000    1.000000
75%     29.000000    8.000000  ...   79.000000    2.000000
max     46.600000    8.000000  ...   82.000000    3.000000

[8 rows x 7 columns]
```

3 Linear Regression

3.1 Simple Linear Regression

Figure 17 displays the simple linear regression fit to the `Advertising` data, where $\hat{\beta}_0 = 0.0475$ and $\hat{\beta}_1 = 7.0326$.

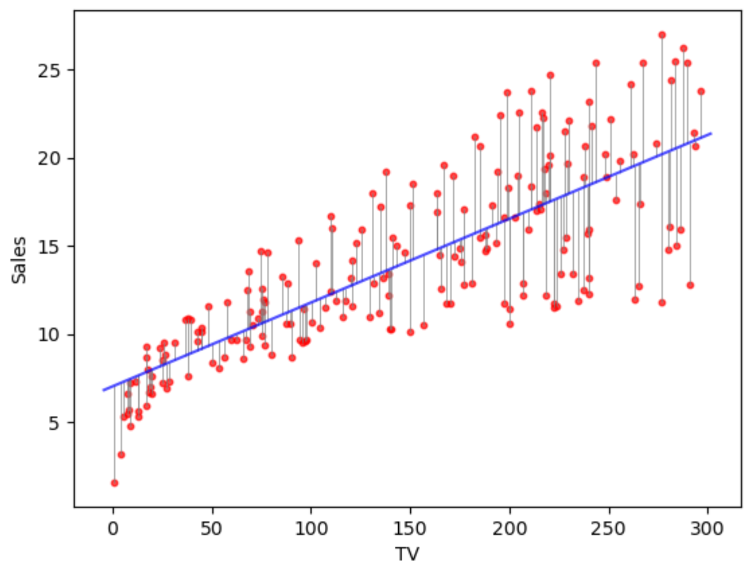

Figure 17: For the `Advertising` data, the least squares fit for the regression of `sales` onto `TV` is shown. The fit is found by minimizing the sum of squared errors. Each grey line represents an error, and the fit makes a compromise by averaging their squares. In this case a linear fit captures the essence of the relationship, although it is somewhat deficient in the left of the plot.

In figure 18, we have computed RSS for a number of values of β_0 and β_1, using the advertising data with `sales` as the response and `TV` as the predictor.

The left-hand panel of figure 19 displays *population regression line* and *least squares line* for a simple simulated example. The red line in the left-hand panel displays the *true* relationship, $f(X) = 2 + 3X$, while the blue line is the least squares estimate based on observed data. In the right-hand panel of figure 19 we have generated five different data sets from the model $Y = 2 + 3X + \epsilon$ and plotted the corresponding five least squares lines.

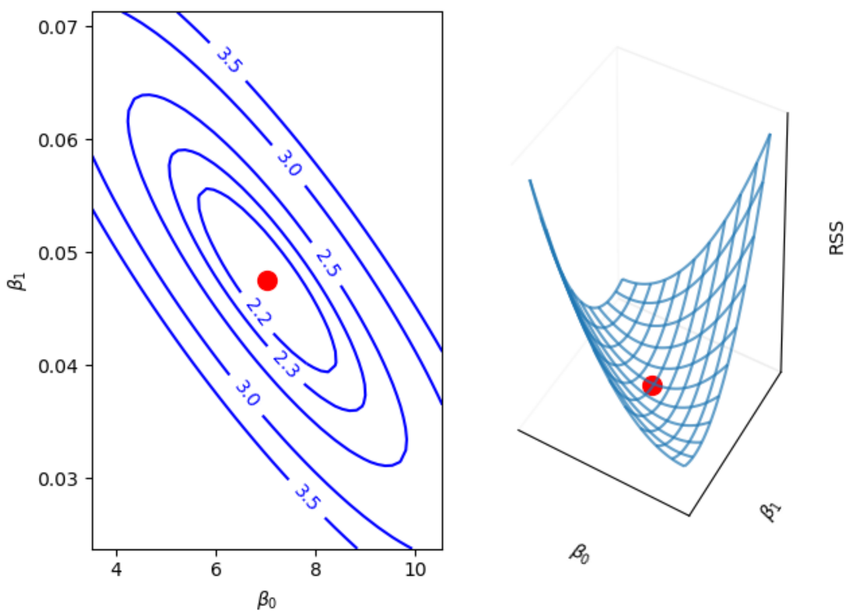

Figure 18: Contour and three-dimensional plots of the RSS on the `Advertising` data, using `sales` as the response and `TV` as the predictor. The red dots correspond to the least squares estimates $\hat{\beta}_0$ and $\hat{\beta}_1$.

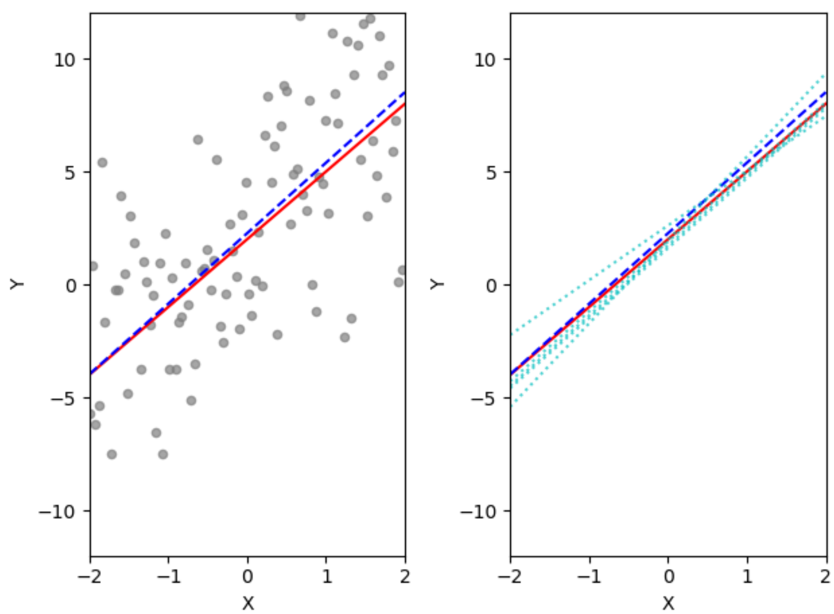

Figure 19: A simulated data set. Left: The red line represents the true relationship, $f(X) = 2 + 3X$, which is known as the population regression line. The blue line is the least squares line; it is the least squares estimate for $f(X)$ based on the observed data, shown in grey circles. Right: The population regression line is again shown in red, and the least squares line in blue. In cyan, five least squares lines are shown, each computed on the basis of a separate random set of observations. Each least squares line is different, but on average, the least squares lines are quite close to the population regression line.

For `Advertising` data, table 1 provides details of the least squares model for the regression of number of units sold on TV advertising budget.

	Coef.	Std.Err.	t	$P >\mid t \mid$
Intercept	7.0326	0.4578	15.3603	0.0
TV	0.0475	0.0027	17.6676	0.0

Table 1: For `Advertising` data, the coefficients of the least squares model for the regression of number of units sold on TV advertising budget. An increase of $1,000 on the TV advertising budget is associated with an increase in sales by around 50 units.

Next, in table 2, we report more information about the least squares model.

Quantity	Value
Residual standard error	3.259
R^2	0.612
F-statistic	312.145

Table 2: For the `Advertising` data, more information about the least squares model for the regression of number of units sold on TV advertising budget.

3.2 Multiple Linear Regression

Table 3 and the next table show results of two simple linear regressions, each of which uses a different advertising medium as a predictor. We find that a $1,000 increase in spending on radio advertising is associated with an increase in sales by around 202 units. A $1,000 increase in advertising spending on on newspapers increases sales by approximately 55 units.

	Coef.	Std.Err.	t	$P >\mid t \mid$
Intercept	12.351	0.621	19.876	0.0
newspaper	0.055	0.017	3.3	0.001

	Coef.	Std.Err.	t	$P >\| t \|$
Intercept	9.312	0.563	16.542	0.0
radio	0.202	0.02	9.921	0.0

Table 3: More simple linear regression models for `Advertising` data. Coefficients of the simple linear regression model for number of units sold on radio advertising budget. a $1,000 increase in spending on radio advertising is associated with an average increase sales by around 202 units.

Figure 20 illustrates an example of the least squares fit to a toy data set with $p = 2$ predictors.

Table 4 displays multiple regression coefficient estimates when TV, radio, and newspaper advertising budgets are used to predict product sales using `Advertising` data.

	Coef.	Std.Err.	t	$P >\| t \|$
Intercept	2.939	0.312	9.422	0.0
TV	0.046	0.001	32.809	0.0
radio	0.189	0.009	21.893	0.0
newspaper	-0.001	0.006	-0.177	0.86

Table 4: For the `Advertising` data, least squares coefficient estimates of the multiple linear regression of number of units sold on radio, TV, and newspaper advertising budgets.

Table 5 shows the correlation matrix for the three predictor variables and response variable in table 4.

	TV	radio	newspaper	sales
TV	1.0	0.0548	0.0566	0.7822
radio	0.0548	1.0	0.3541	0.5762
newspaper	0.0566	0.3541	1.0	0.2283
sales	0.7822	0.5762	0.2283	1.0

Table 5: Correlation matrix for `TV`, `radio`, and `sales` for the `Advertising` data.

Figure 21 displays a three-dimensional plot of `TV` and `radio` versus `sales`.

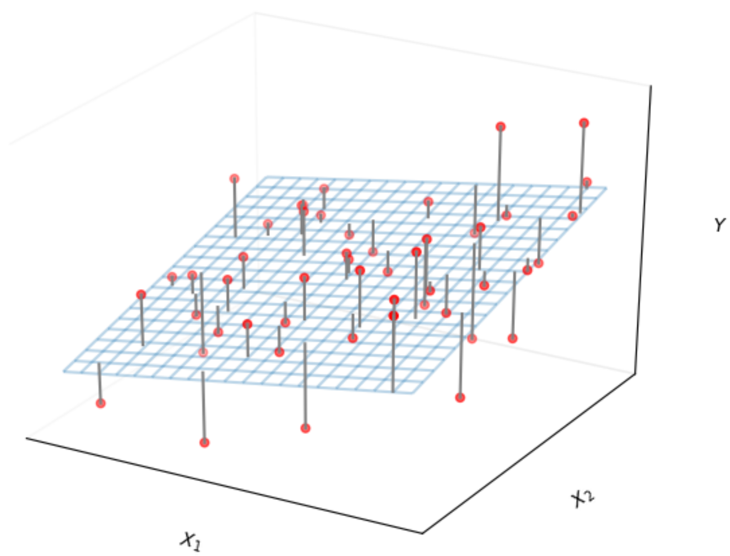

Figure 20: In a three-dimensional setting, with two predictors and one response, the least squares regression line becomes a plane. The plane is chosen to minimize the sum of the squared vertical distances between each observation (shown in red) and the plane.

Quantity	Value
Residual standard error	1.69
R^2	0.897
F-statistic	570.0

Table 6: More information about the least squares model for the regression of number of units sold on TV, newspaper, and radio advertising budgets in the `Advertising` data. Other information about this model was displayed in table 4.

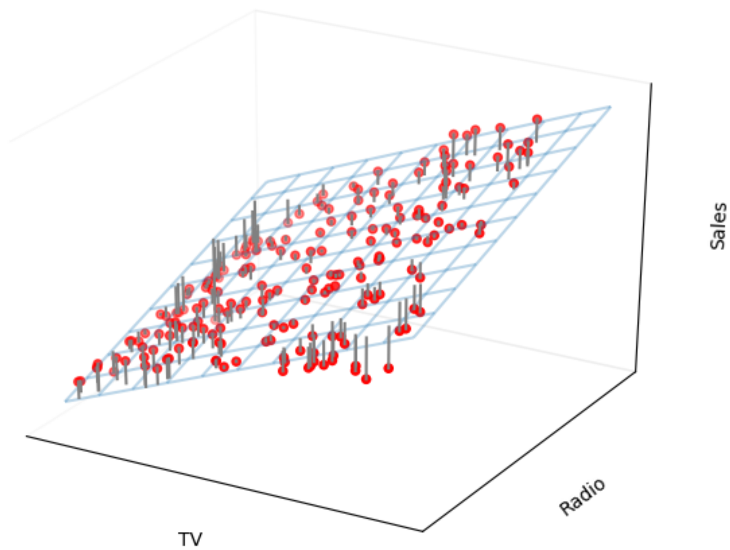

Figure 21: For the `Advertising` data, a linear regression fit to `sales` using `TV` and `radio` as predictors. From the pattern of the residuals, we can see that there is a pronounced non-linear relationship in the data. The positive residuals tend to lie along the 45-degree line, where TV and Radio budgets are split evenly. The negative residuals tend to lie away from this line, where budgets are more lopsided.

3.3 Other Considerations in the Regression Model

`Credit` data set displayed in figure 22 records `balance` (average credit card debt for a number of individuals) as well as several quantitative predictors: `age`, `cards` (number of credit cards), `education` and `rating` (credit rating).

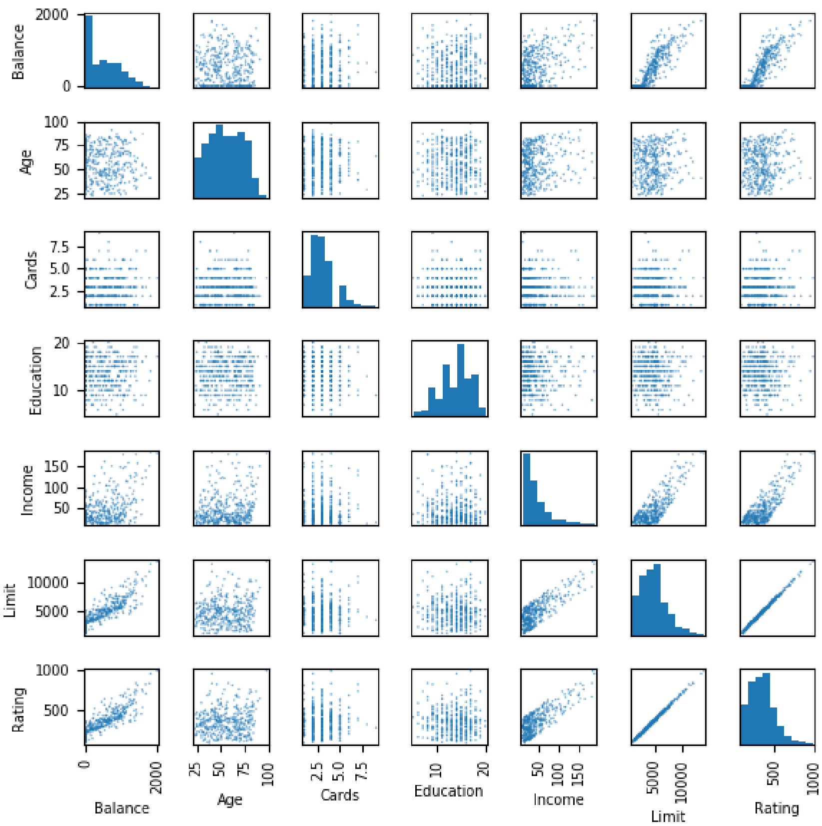

Figure 22: The `Credit` dataset contains information about `balance`, `age`, `cards`, `education`, `income`, `limit`, and `rating` for a number of potential customers.

Table 7 displays the coefficient estimates and other information associated with the model where `gender` is the only explanatory variable.

From table 8 we see that the estimated `balance` for the baseline, African American, is \$531.0. It is estimated that the Asian category will have an additional \$-18.7 debt, and that the Caucasian category will have an additional \$-12.5 debt compared to Africna American category.

	Coef.	Std.Err.	t	$P >\| t \|$
Intercept	509.803	33.128	15.389	0.0
Gender[T.Female]	19.733	46.051	0.429	0.669

Table 7: Least squares coefficient estimates associated with the regression of `balance` onto `gender` in the `Credit` data set.

	Coef.	Std.Err.	t	$P >\| t \|$
Intercept	531.0	46.319	11.464	0.0
Ethnicity[T.Asian]	-18.686	65.021	-0.287	0.774
Ethnicity[T.Caucasian]	-12.503	56.681	-0.221	0.826

Table 8: Least squares coefficient estimates associated with the regression of `balance` onto `ethnicity` in the `Credit` data set.

Table 9 shows results of regressing `sales` and `TV` and `radio` when an interaction term is included. Coefficient of interaction term `TV:radio` is highly significant.

In figure 23, the left panel shows least squares lines when we predict `balance` using `income` (quantitative) and `student` (qualitative variables). There is no interaction term between `income` and `student`. The right panel shows least squares lines when an interaction term is included.

	Coef.	Std.Err.	t	$P >\| t \|$
Intercept	6.75	0.248	27.233	0.0
TV	0.019	0.002	12.699	0.0
radio	0.029	0.009	3.241	0.001
TV:radio	0.001	0.0	20.727	0.0

Table 9: For `Advertising` data, least squares coefficient estimates associated with the regression of `sales` onto `TV` and `radio`, with an interaction term.

Figure 24 shows a scatter plot of `mpg` (gas mileage in miles per gallon) versus `horsepower` in the `Auto` data set. The figure also includes least squares fit line for linear, second degree, and fifth degree polynomials in `horsepower`.

Table 10 shows regression results of a qudratic fit to explain `mpg` as a function of `horsepower` and $horsepower^2$.

The left panel of figure 25 displays a residual plot from the linear regression of `mpg` onto `horsepower` on the `Auto` data set. The red line is a smooth fit to the residuals, which is displayed in order to make it easier to identify

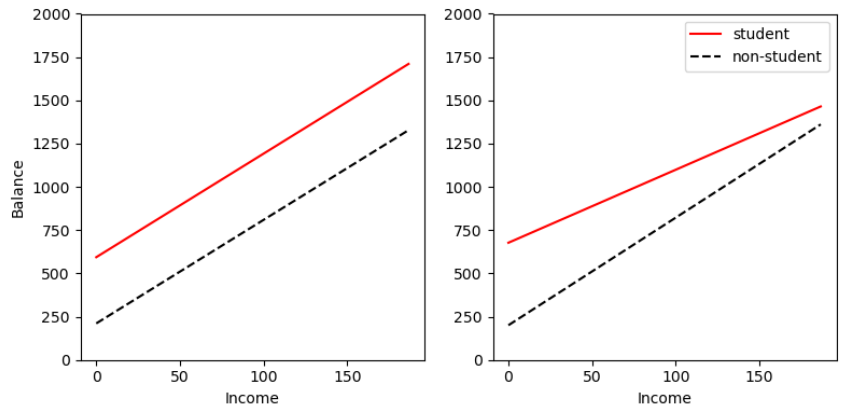

Figure 23: For the `Credit` data, the least squares lines are shown for prediction of `balance` from `income` for students and non-students. Left: There is no interaction between `income` and `student`. Right: There is an interaction term between `income` and `students`.

	Coef.	Std.Err.	t	$P >\mid t \mid$
Intercept	56.9001	1.8004	31.6037	0.0
horsepower	-0.4662	0.0311	-14.9782	0.0
$horsepower^2$	0.0012	0.0001	10.0801	0.0

Table 10: For the `Auto` data set, least squares coefficient estimates associated with the regression of `mpg` onto `horsepower` and

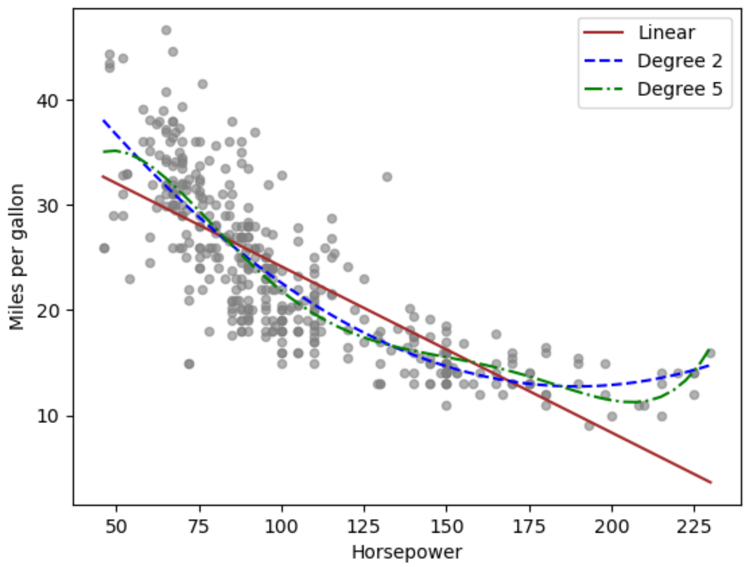

Figure 24: The `Auto` data set. For a number of cars, `mpg` and `horsepower` are shown. The linear regression fit is shown in orange. The linear regression fit for a model that includes first- and second-order terms of `horsepower` is shown as blue curve. The linear regression fit for a model that includes all polynomials of `horsepower` up to fifth-degree is shown in green.

any trends. The residuals exhibit a clear U-shape, which strongly suggests non-linearity in the data. In contrast, the right hand panel of figure25 displays the residual plot results from the model which contains a quadratic term in `horsepower`. Now there is little pattern in residuals, suggesting that the quadratic term improves the fit to the data.

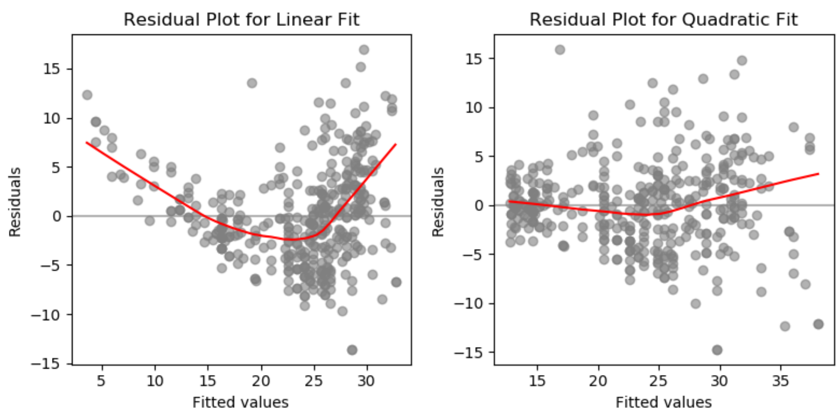

Figure 25: Plots of residuals versus predicted (or fitted) values for the `Auto` data set. In each plot, the red line is a smooth fit to the residuals, intended to make it easier to identify a trend. Left: A linear regression of `mpg` on `horsepower`. A strong pattern in the residuals indicates non-linearity in the data. Right: A linear regression of `mpg` on `horsepower` and square of `horsepower`. Now there is little pattern in the residuals.

Figure 26 provides an illustration of correlations among residuals. In the top panel, we see the residuals from a linear regression fit to data generated with uncorrelated errors. There is no evidence of time-related trend in the residuals. In contrast, the residuals in the bottom panel are from a data set in which adjacent errors had a correlation of 0.9. Now there is a clear pattern in the residuals - adjacent residuals tend to take on similar values. Finally, the center panel illustrates a more moderate case in which the residuals had a correlation of 0.5. There is still evidence of tracking, but the pattern is less pronounced.

In the left-hand panel of figure 27, the magnitude of the residuals tends to increase with the fitted values. The right hand panel displays residual plot after tranforming the response using $\log(Y)$. The residuals now appear to have constant variance, although there is some evidence of a non-linear relationship in the data.

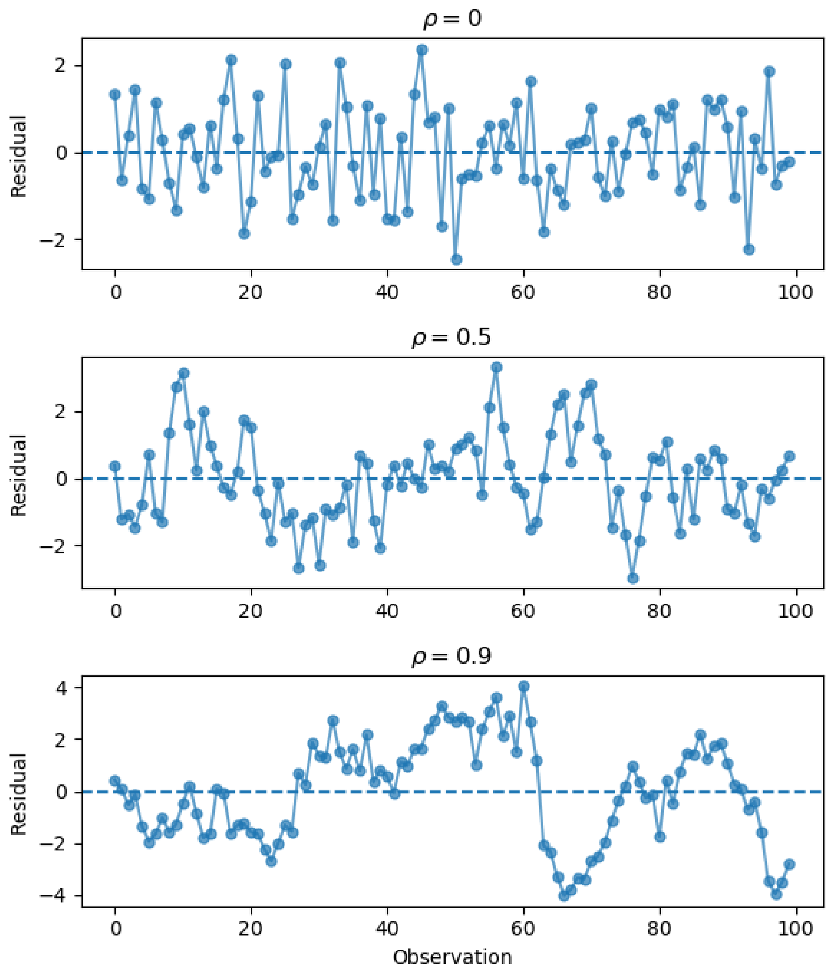

Figure 26: Plots of residuals from simulated time series data sets generated with differeing levels of correlation ρ between error terms for adjacent time points.

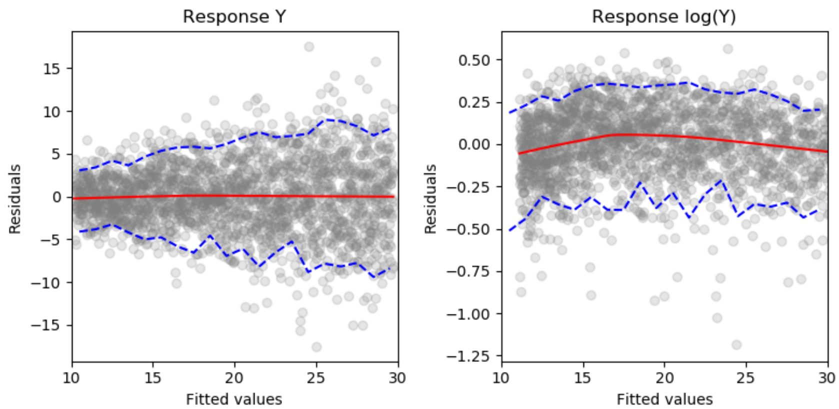

Figure 27: Residual plots. The red line, a smooth fit to the residuals, is intended to make it easier to identify a trend. The blue lines track 5^{th} and 95^{th} percentiles of the residuals, and emphasize patterns. Left: The funnel shape indicates heteroscedasticity. Right: the response has been log transformed, and now there is no evidence of heteroscedasticity.

The red point (observation 20) in the left hand panel of figure 28 illustrates a typical outlier. The red solid line is the least squares regression fit, while the blue dashed line is the least squares fit after removal of the outlier. In this case, removal of outlier has little effect on the least squares line. In the center panel of figure 28, the outlier is clearly visible. In practice, to decide if the outlier is sufficiently big to be considered an outlier, we can plot *studentized residuals*, computed by dividing each residual ϵ_i by its estimated standard error. These are shown in the right hand panel.

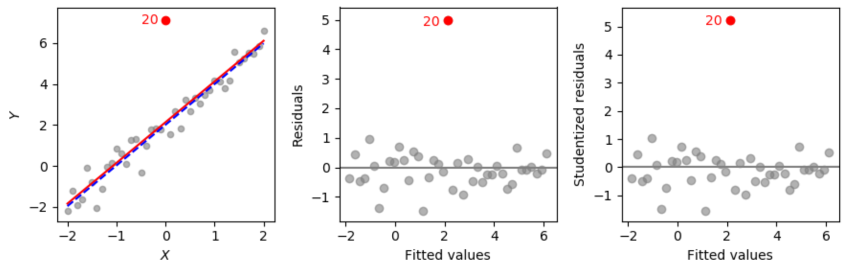

Figure 28: Left: The least squares regression line is shown in red. The regression line after removing the outlier is is shown in blue. Center: The residual plot clearly identifies the outlier. Right: The outlier has a studentized residual of 6; typically we expect values between -3 and 3.

3.4 Lab: Linear Regression

It is possible to call R from Python and *vice versa*.