ASSIGNMENT 2

Machine Learning

ALEXANDER CROLL

Advanced Artificial Intelligence, IT714A

Data Science, University of Skövde

Vicenç Torra

December 11, 2016

Table of Contents

1. Introduction

This document describes the 2nd assignment in the course Advanced Artificial Intelligence. The assignment focuses on Machine Learning (ML) and both the regression experiment and analysis as well as the code will be included in this paper.

1.1. Task Description

This assignment is to analyze a supervised machine learning algorithm and study the bias-variance trade-off. The report should contain

- a description on the experiments done (e.g., design of the experiments, software used, machine learning algorithm, what complexity means in the selected representation formalism)
- a discussion of the results obtained
- the code implemented to obtain the results
- the references used.

.

2. Experiment Description

The following chapter will describe the experiment setup and solution for the given task from above. This includes the data set, software and ML algorithm being used.

2.1. Dataset

The data set used for this regression experiment was published by *Data-World* on www.data.world. The data set includes global climate change data with different temperature variables over the time series starting in 1750.[1] The data set is described as following:

> Date: starts in 1750 for average land temperature and 1850 for max and min land temperatures and global ocean and land temperatures
>
> LandAverageTemperature: global average land temperature in celsius
>
> LandAverageTemperatureUncertainty: the 95% confidence interval around the average
>
> LandMaxTemperature: global average maximum land temperature in celsius
>
> LandMaxTemperatureUncertainty: the 95% confidence interval around the maximum land temperature
>
> LandMinTemperature: global average minimum land temperature in celsius
>
> LandMinTemperatureUncertainty: the 95% confidence interval around the minimum land temperature
>
> LandAndOceanAverageTemperature: global average land and ocean temperature in celsius
>
> LandAndOceanAverageTemperatureUncertainty: the 95% confidence interval around the global average land and ocean temperature

Here is a sample excerpt from the .csv file:

```
1     1750-01-01,3.0340000000000003,3.574,,,,,,
2     1750-02-01,3.083,3.702,,,,,,
3     1750-03-01,5.626,3.076,,,,,,
4     1750-04-01,8.49,2.451,,,,,,
5     1750-05-01,11.573,2.072,,,,,,
6     1750-06-01,12.937000000000001,1.724,,,,,,
7     1750-07-01,15.868,1.911,,,,,,
8     1750-08-01,14.75,2.231,,,,,,
9     1750-09-01,11.412999999999998,2.637,,,,,,
10    1750-10-01,6.366999999999999,2.668,,,,,,
11    1750-11-01,,,,,,,,
12    1750-12-01,2.772,2.97,,,,,,
```

[1] https://data.world/data-society/global-climate-change-data

There is one record per month for every year since 1750. Before starting with the regression experiment, the data set is cleaned by

- Creating a one-dimensional time series of type (date, double) with the first column being the first day of each month of every year and the second column the land average temperature at that point in time
- Removing all empty values from the data set
- Creating the average temperature per year

Finally, the data set in use has one record with the average temperature for every year since 1750.

2.2. Software

The programming language used for this assignment is R. These libraries are imported:

```
1     library(readr) ## to load data into R
2     library(tidyr) ## cleaning data
3     library(dplyr) ## cleaning data
4     library(ggplot2) ## plotting results from regression
5     library(sparklyr) ##  comparison of standard polynomial regression with Spark
      mllib random forest
```

R includes a standard package called “stats”. This package includes statistical functions and its function lm() can be used for both linear and polynomial regression.

```
lm(formula, data, subset, weights, na.action,
   method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE,
   singular.ok = TRUE, contrasts = NULL, offset, ...)
```

2.3. Machine Learning Algorithm

As mentioned in the previous section, the algorithm used to conduct this regression experiment, is polynomial regression. In polynomial regression, the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial.[2] Polynomial regression itself is nonlinear and the number of turns is indicated by the degree n (n=2 has n-1 turns etc.). However, it is considered to be a special case of linear regression.[3]

[2] https://en.wikipedia.org/wiki/Polynomial_regression
[3] http://www.theanalysisfactor.com/regression-modelshow-do-you-know-you-need-a-polynomial/

Using the lm() regression function from R's stats library, a polynomial model (e.g. degree n = 2) can be fitted in the following way:

```
polyFit <- lm(y ~ poly(x, 2, raw = TRUE), data = train)
```

This will result in a polynomial function that automatically minimizes the error in the following form:

$$y_i = b_0 + b_1 * x_i + b_2 x_i^2 + \ldots + b_n * x_i^n$$

To illustrate that the lm() function can use poly() to fit polynomial regression models, we can rewrite the R code into

```
polyFit <- lm(y ~ x + I(x²), data = train)
```

or correspondingly into

```
polyFit <- lm(y ~ x + I(x²) + I(x³), data = train)
```

Hence, the poly() function makes fitting polynomials of higher degrees easier, because there is no need to write out the entire equation.[4]

2.4. Complexity

The polynomial degree influences the complexity of the regression model. Generally speaking, the higher the degree, the better the approximation. However, a model needs to be found that does not allow for too high of a variance.

This experiment starts with a polynomial degree of n = 1 which is then increased incrementally by 1 up to a degree of n = 20. Each of these fitted models represents different complexity.

For each polynomial degree, different models were built with different training sets. The different training sets were obtained by applying cross-validation (with i = 10), hence 10 models were built at each polynomial degree.

To reduce visualization complexity a bit in this experiment, the original data set was reduced to a simple one-dimensional time series. If using the original multivariate time series, this would help to further increase complexity of the regression model. Also, the number of data points

[4] https://www.r-bloggers.com/polynomial-regression-techniques/

was reduced from monthly to yearly basis. Using a higher number of data points would increase complexity as well and potentially lead to more accuracy of the model.

3. Results

This chapter will showcase the results of the experiment and elaborate on them.

First, let's have a look at the original data set and how the observations of yearly average temperature from 1750 until today are plotted. While there were some outliers in the early years, the distance between observations became less and over time and a clear and somewhat continuous increase in temperature can be observed.

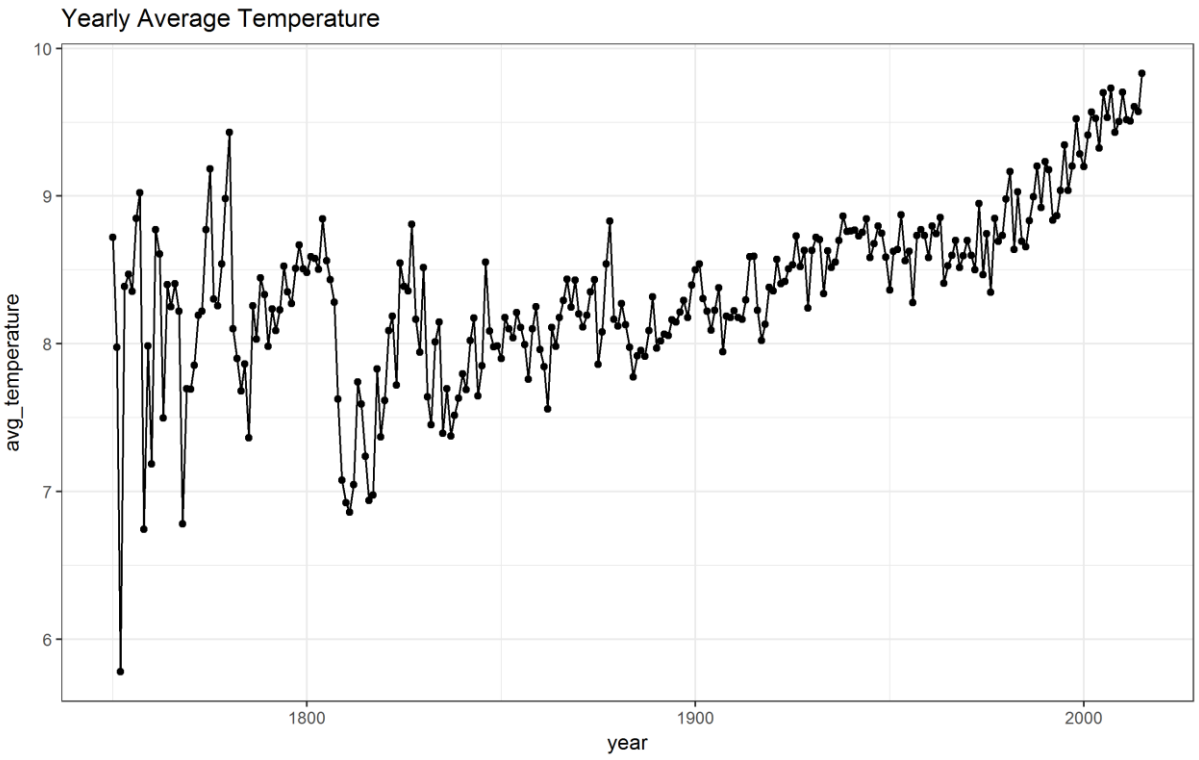

Fig. 1: Yearly Average Global Temperature (from original data set)

Next, three polynomial models of degrees 1, 2 and 3 are plotted in figure 2. As the models advance from linear to quadratic and finally to cubic, it can be seen that accuracy is improved as the polynomial degree increases. Now, it should be evaluated which polynomial degree proves to be the optimal fit for the given data set.

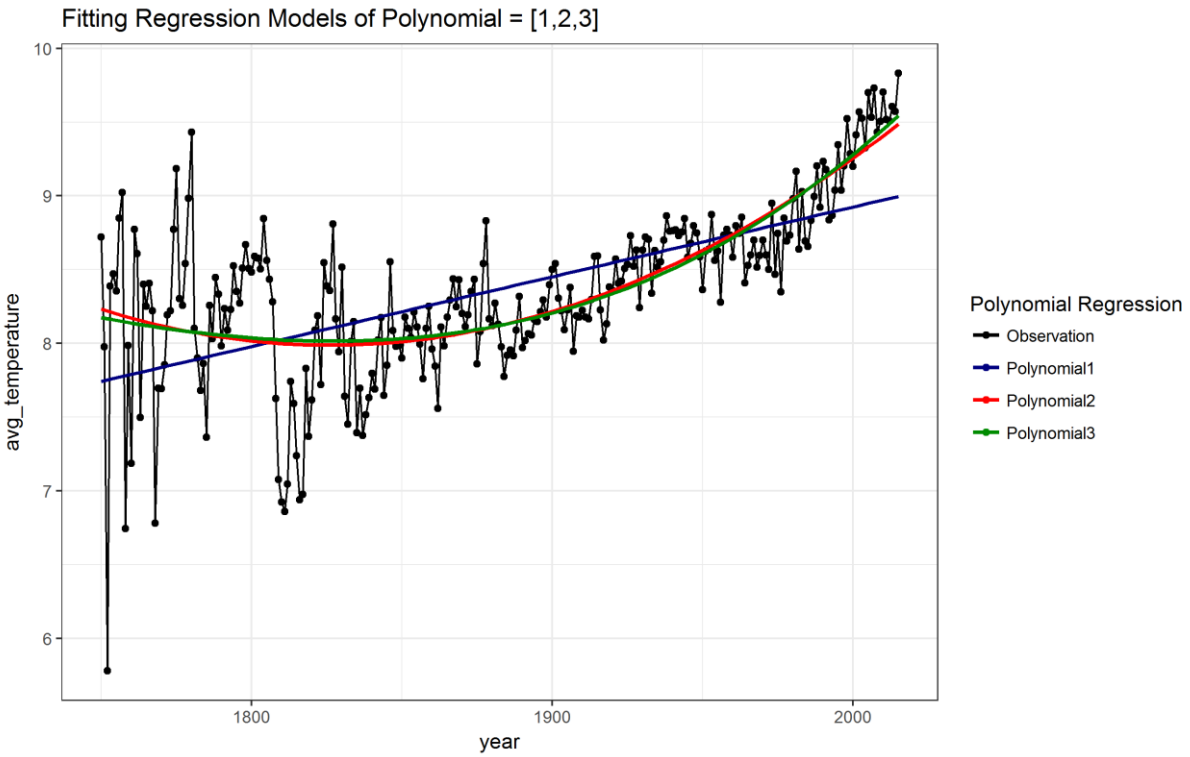

Fig. 2: Polynomial Regression Models of Degree 1, 2 and 3 fitted to observed data

As a next step, regression models for polynomials from n = 1 to n =20 are built by applying cross-validation (with i = 10), creating 10 models for each polynomial degree. For these, the bias is calculated on the basis of the training set. Additionally, the variance is calculated on the basis of the test set. The results are visualized in figure 3-5.

	polynomial	model	bias	variance
1	1	1	0.3369256	0.4627709
2	1	2	0.3422042	0.4559391
3	1	3	0.3586702	0.3864826
4	1	4	0.3410986	0.3995721
5	1	5	0.3531715	0.4516387
6	1	6	0.3369393	0.3911641
7	1	7	0.3514358	0.4592988
8	1	8	0.3398901	0.4043398
9	1	9	0.3423502	0.4497601
10	1	10	0.3364427	0.4302979
11	2	1	0.2789385	0.4895138
12	2	2	0.2731355	0.4402601
13	2	3	0.2851716	0.4828532
14	2	4	0.2768317	0.4251004
15	2	5	0.2834708	0.4934356
16	2	6	0.2679418	0.4412356
17	2	7	0.2743903	0.4513175
18	2	8	0.2745743	0.4723649
19	2	9	0.2735317	0.4566826

Fig. 3: Bias and Variance for polynomial degrees 1-20 and different models

	polynomial	bias	variance
1	1	0.3439128	0.4291264
2	2	0.2763041	0.4619971
3	3	0.2755886	0.4522634
4	4	0.2747908	0.4538643
5	5	0.2747908	0.4538643
6	6	0.2677800	0.4784583
7	7	0.2677800	0.4784583
8	8	0.2677800	0.4784583
9	9	0.2689951	0.4829944
10	10	0.2689951	0.4829944
11	11	0.2689951	0.4829944
12	12	0.2689951	0.4829944
13	13	0.2689951	0.4829944
14	14	0.2687142	0.4843672
15	15	0.2687142	0.4843672
16	16	0.2687142	0.4843672
17	17	0.2687142	0.4843672
18	18	0.2687142	0.4843672
19	19	0.2687142	0.4843672
20	20	0.2687142	0.4843672

Fig. 4: Bias and Variance for each polynomial degree from 1-20

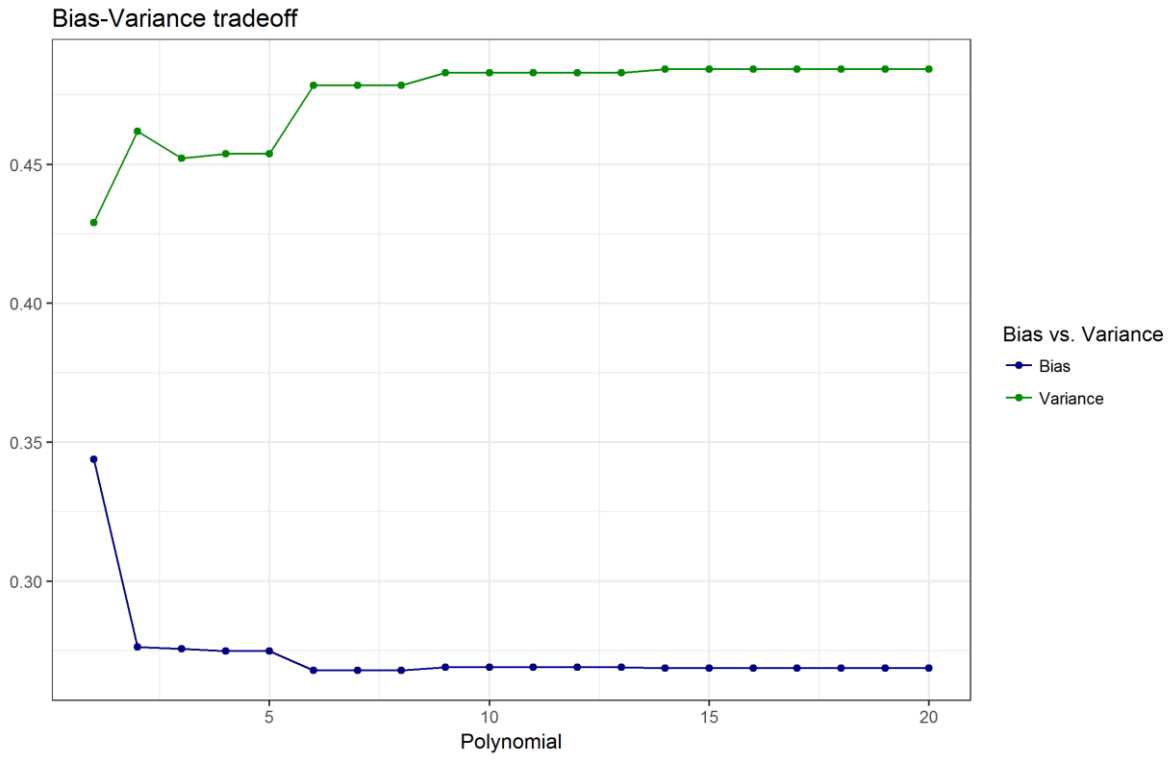

Fig. 3: Bias and variance with increasing complexit

Based on the obtained results, and given the principle of Occam's razor, which states to select the simpler of two methods, the polynomial regression model with polynomial = 6 should be chosen. This model gives a good representation of the real world and more complex models do not provide better bias/variance values.

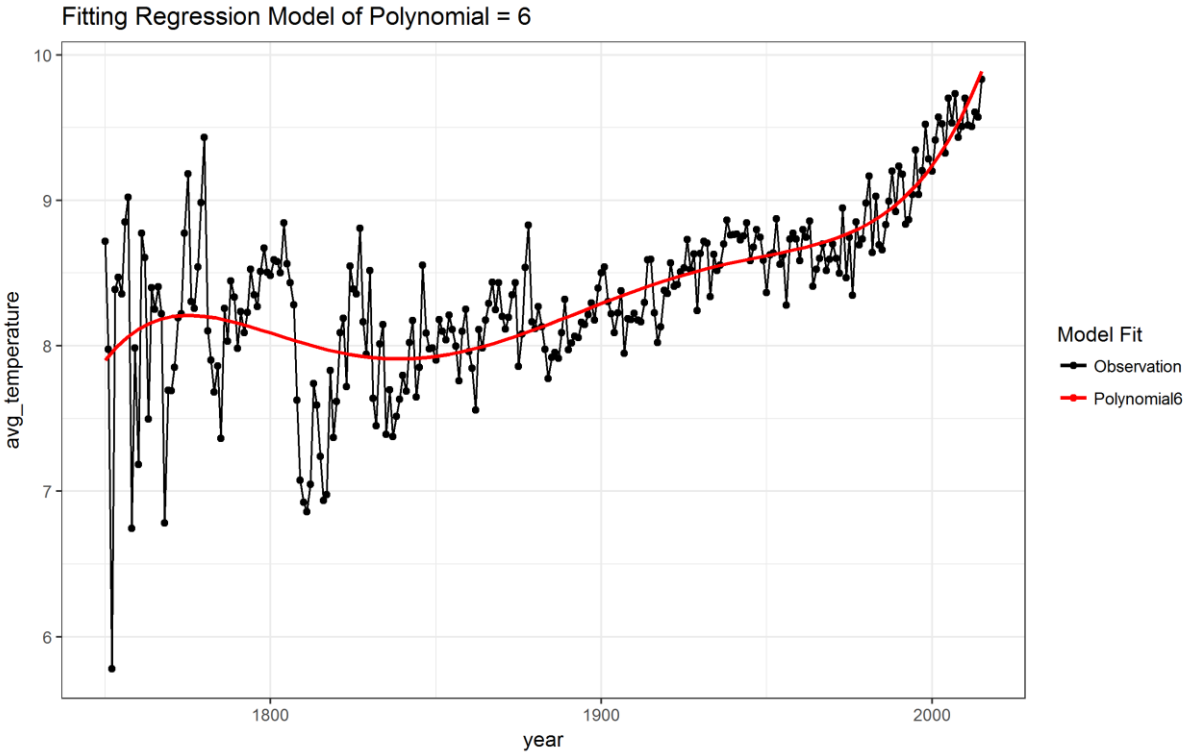

Fig. 6: Polynomial Regression Model of n = 6

4. Code

```
## ==================== AI - MACHINE LEARNING ASSIGNMENT ==================== ##

## Loading libraries
library(readr)
library(tidyr)
library(dplyr)
library(ggplot2)
library(sparklyr)

## Setting working directory
setwd("C:/Users/alcroll/OneDrive - Deloitte (O365D)/Documents/R workspace/AI")

## Reading input file (red wines) as ;-delimited file
weather_raw <- read_csv("GlobalLandTemperatures-GlobalTemperatures.csv")

## Cleaning dataset
## Average temperature per year
weather <- weather_raw %>%
  filter(!is.na(LandAverageTemperature)) %>%
  select(dt, LandAverageTemperature) %>%
  mutate(dt = as.integer(substring(dt, 1,4))) %>%
  group_by(dt) %>%
  summarize(avg_temperature = mean(LandAverageTemperature)) %>%
  rename(year = dt)

## Plotting the observations of average yearly temperature from 1750-today
ggplot(weather, aes(x = year, y = avg_temperature)) +
  geom_line(col = "black",  lwd = 0.5) +
  geom_point(col = "black") +
  theme_bw() +
  ggtitle("Yearly Average Temperature")
ggsave("yearly_average.png")

## Creating random indexes for weather normalized dataset
## 20% of weather dataset
indexes <- sample(1:nrow(weather), size = 0.2*nrow(weather))

## Creating test and training datasets
## Test set will have 20% of original dataset
## Training set will have 80%of original dataset
test <- weather[indexes,]
train <- weather[-indexes,]

## Fitting regression models of polynomials 1,2,3
fit1 <- lm(avg_temperature ~ year, data = train)
fit2 <- lm(avg_temperature ~ poly(year, 2, raw = TRUE), data = train)
fit3 <- lm(avg_temperature ~ poly(year, 3, raw = TRUE), data = train)

## Plotting regression models
ggplot(weather, aes(x = year, y = avg_temperature, color = "Observation")) +
  geom_point() +
```

```
  geom_line() +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, aes(color =
      "Polynomial1")) +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ poly(x, 2, raw =
  TRUE), aes(color = "Polynomial2")) +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ poly(x, 3, raw =
  TRUE), aes(color = "Polynomial3")) +
  ggtitle("Fitting Regression Models of Polynomial = [1,2,3]") +
  scale_color_manual(name = "Polynomial Regression",
                     values = c(Polynomial1 = "navy", Polynomial2 = "red",
                            Polynomial3 = "green4", Observation = "black")) +
  theme_bw()
ggsave("regression-models1+2+3.png")

## Bias-variance tradeoff
## Initializing tradeoff data frame with columns polynomial, bias, variance,
error
tradeoff_all <- data.frame(polynomial = numeric(0), model = numeric(0), bias
= numeric(0), variance = numeric(0))

## Bias Function
f_bias_new <- function(x){
  sum(abs(predict(lm(avg_temperature ~ poly(year, k, raw = TRUE), data = x),
                  newdata = x)/nrow(x) - x$avg_temperature/nrow(x)))
}

## Variance Function
## Building model for entire dataset
f_variance_new <- function(x, y){
sum(abs(predict(lm(avg_temperature ~ poly(year, k, raw = TRUE), data = x),
newdata = y)/nrow(x) -
      predict(lm(avg_temperature ~ poly(year, k, raw = TRUE), data = x),
 newdata = x)/nrow(x)))
}

# K-folds validation
weatherKFolds <- weather[sample(nrow(weather)),]

#Create 10 folds
kfolds <- cut(seq(1, nrow(weatherKFolds)), breaks = 10, labels = FALSE)

#Perform 10 fold cross validation
tradeoff_all <- tradeoff_all[0,]
for(k in 1:20){
  for(i in 1:10){
    #Segement your data by fold using the which() function
    testIndexes <- which(kfolds==i, arr.ind=TRUE)
    testData <- weatherKFolds[testIndexes, ]
    trainData <- weatherKFolds[-testIndexes, ]
    tradeoff_all[nrow(tradeoff_all)+1,] <- c(k, i, f_bias_new(trainData),
    f_variance_new(trainData, testData),0)
  }
}

tradeoff <- tradeoff[0,]
tradeoff <- tradeoff_all %>%
```

```
100  group_by(polynomial) %>%
101  select(polynomial, bias, variance) %>%
102  summarize(bias = mean(bias),
103            variance = mean(variance))
104
105## Plotting bias and variance as complexity increases
106ggplot(tradeoff, aes(x = polynomial)) +
107  geom_line(aes(y = bias, color = "Bias")) +
108  geom_point(aes(y = bias, color = "Bias")) + theme_bw() +
109  geom_line(aes(y = variance, color = "Variance")) +
110  geom_point(aes(y = variance, color = "Variance")) +
111  ggtitle("Bias-Variance tradeoff") +
112  xlab("Polynomial") +
113  ylab("") +
114  scale_color_manual(name = "Bias vs. Variance", values = c(Bias = "navy",
     Variance = "green4"))
115ggsave("bias-variance-tradeoff.png")
116
117## Plotting regression model
118ggplot(weather, aes(x = year, y = avg_temperature, color = "Observation")) +
119  geom_point() +
120  geom_line() +
121  geom_smooth(method = "lm", se = FALSE, formula = y ~ poly(x, 6, raw =
     TRUE), aes(color = "Polynomial6")) +
122  ggtitle("Fitting Regression Model of Polynomial = 6") +
123  theme_bw() +
124  scale_color_manual(name = "Model Fit", values = c(Observation = "black",
     Polynomial6 = "red"))
125ggsave("regression-model6.png")
126
127## ==================== END ====================
```