STAT 471: Homework 2

Name

Due: October 4, 2021 at 11:59pm

Contents

Instructions **2**
- Setup . . . 2
- Collaboration . . . 2
- Writeup . . . 2
- Programming . . . 2
- Grading . . . 2
- Submission . . . 2

1 Case study: Bone mineral density (40 points for correctness; 10 points for presentation) **3**
- 1.1 Import (2 points) . . . 3
- 1.2 Explore (10 points) . . . 3
- 1.3 Model (15 points) . . . 6
- 1.4 Evaluate (6 points) . . . 10
- 1.5 Interpret (7 points) . . . 11

2 KNN and bias-variance tradeoff (45 points for correctness; 5 points for presentation) **12**
- Setup: Apple farming . . . 12
- 2.1 A simple rule to predict this season's yield (15 points) . . . 14
- 2.2 K-nearest neighbors regression (conceptual) (15 points) . . . 14
- 2.3 K-nearest neighbors regression (simulation) (15 points) . . . 15

Instructions

Setup

Pull the latest version of this assignment from Github and set your working directory to `stat-471-fall-2021/homework/homework-2`. Consult the getting started guide if you need to brush up on `R` or `Git`.

Collaboration

The collaboration policy is as stated on the Syllabus:

> "Students are permitted to work together on homework assignments, but solutions must be written up and submitted individually. Students must disclose any sources of assistance they received; furthermore, they are prohibited from verbatim copying from any source and from consulting solutions to problems that may be available online and/or from past iterations of the course."

In accordance with this policy,

Please list anyone you discussed this homework with: Zach Bradlow and Paul Heysch de la Borde

Please list what external references you consulted (e.g. articles, books, or websites): https://liangf.wordpress.com/2019/08/18/degree-of-freedom-for-knn/ https://medium.com/30-days-of-machine-learning/day-3-k-nearest-neighbors-and-bias-variance-tradeoff-75f84d515bdb

Writeup

Use this document as a starting point for your writeup, adding your solutions after "**Solution**". Add your R code using code chunks and add your text answers using **bold text**. Consult the preparing reports guide for guidance on compilation, creation of figures and tables, and presentation quality.

Programming

The `tidyverse` paradigm for data wrangling, manipulation, and visualization is strongly encouraged, but points will not be deducted for using base `R`.

We'll need to use the following `R` packages:

```
library(tidyverse)  # tidyverse
library(kableExtra) # for printing tables
library(cowplot)    # for side by side plots
library(FNN)        # for K-nearest-neighbors regression
```

We'll also need the `cross_validate_spline` function from Unit 2 Lecture 3:

```
source("../../functions/cross_validate_spline.R")
```

Grading

The point value for each problem sub-part is indicated. Additionally, the presentation quality of the solution for each problem (as exemplified by the guidelines in Section 3 of the preparing reports guide will be evaluated on a per-problem basis (e.g. in this homework, there are three problems). There are 100 points possible on this homework, 85 of which are for correctness and 15 of which are for presentation.

Submission

Compile your writeup to PDF and submit to Gradescope.

1 Case study: Bone mineral density (40 points for correctness; 10 points for presentation)

In this exercise, we will be looking at a data set (available online) on spinal bone mineral density, a physiological indicator that increases during puberty when a child grows.

Below is the data description:

> "Relative spinal bone mineral density measurements on 261 North American adolescents. Each value is the difference in spnbmd taken on two consecutive visits, divided by the average. The age is the average age over the two visits."
>
> Variables:
>
> `idnum`: identifies the child, and hence the repeat measurements
>
> `age`: average age of child when measurements were taken
>
> `gender`: male or female
>
> `spnbmd`: Relative Spinal bone mineral density measurement

The goal is to learn about the typical trends of bone mineral density during puberty for boys and girls.

1.1 Import (2 points)

- Using `readr`, import the data from the above URL into a tibble called `bmd`. Specify the column types using the `col_types` argument.
- Print the imported tibble (no need to use `kable`).

```
bmd <- read_tsv("https://web.stanford.edu/~hastie/ElemStatLearn/datasets/bone.data", col_types = "idfd")
print(bmd)
```

```
## # A tibble: 485 x 4
##    idnum   age gender   spnbmd
##    <int> <dbl> <fct>     <dbl>
##  1     1  11.7 male    0.0181
##  2     1  12.7 male    0.0601
##  3     1  13.8 male    0.00586
##  4     2  13.2 male    0.0103
##  5     2  14.3 male    0.211
##  6     2  15.3 male    0.0408
##  7     3  11.4 male   -0.0296
##  8     3  12.4 male   -0.00643
##  9     3  13.4 male    0.0566
## 10     4  10.6 female  0.108
## # ... with 475 more rows
```

1.2 Explore (10 points)

- To keep things simple, let's ignore the fact that we have repeated measurements on children. To this end, remote the `idnum` column from `bmd`.

```
bmd <- bmd %>%
  select(-c("idnum")) #removes idnum column
```

- What is the number of boys and girls in this dataset (ignoring the fact that there are repeated measurements)? What are the median ages of these boys and girls?

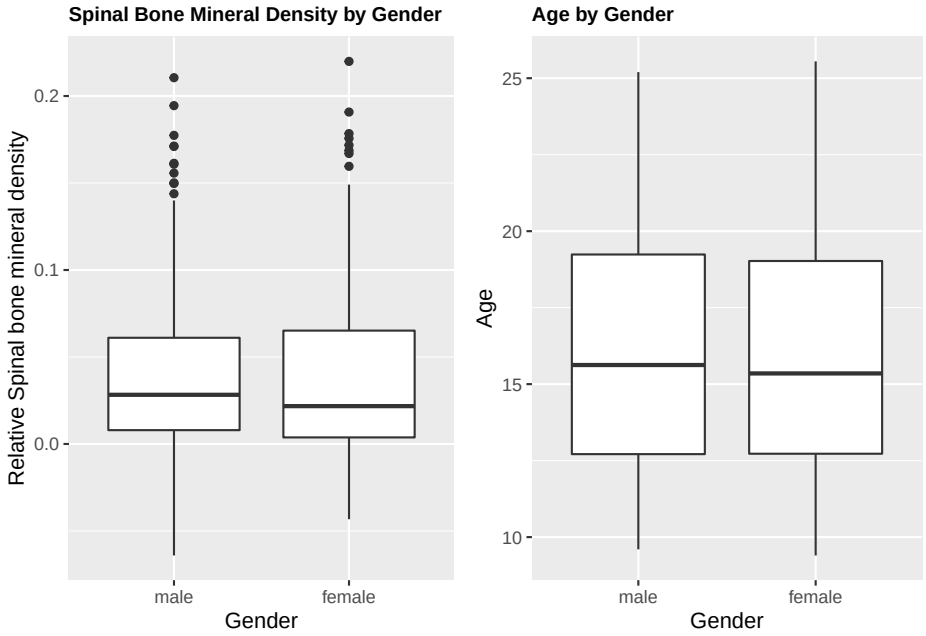
Table 1: Boys and Girls in Sample

gender	count	median_age
male	226	15.6
female	259	15.3

```
boy_girl <- bmd %>%
  group_by(gender) %>%  #groups by gender
  summarise(count = n(), median_age = median(age)) #calculates observations and median age by gender

boy_girl  %>%
  kable(format = "latex", row.names = NA,
      booktabs = TRUE, digits = 2,
      caption = "Boys and Girls in Sample") %>%
  kable_styling(position = "center")
```

- Produce boxplots to compare the distributions of `spnbmd` and `age` between boys and girls (display these as two plots side by side, one for `spnbmd` and one for `age`). Are there apparent differences in either `spnbmd` or `age` between these two groups?

```
spnbmd_box <- bmd %>%
  ggplot() +
  geom_boxplot(aes(x = gender, y = spnbmd)) + #creates boxplot
  labs(
    x = "Gender",
    y = "Relative Spinal bone mineral density"
  ) +
  ggtitle("Spinal Bone Mineral Density by Gender") + #adds title
  theme(plot.title = element_text(size = 10, face = "bold")) #formats title

age_box <- bmd %>%
  ggplot() +
  geom_boxplot(aes(x = gender, y = age)) + #creates boxplot
  labs(
    x = "Gender",
    y = "Age"
  ) +
  ggtitle("Age by Gender") + #adds title
  theme(plot.title = element_text(size = 10, face = "bold")) #formats title

#plots grids next to each other
plot_grid(spnbmd_box, age_box)
```

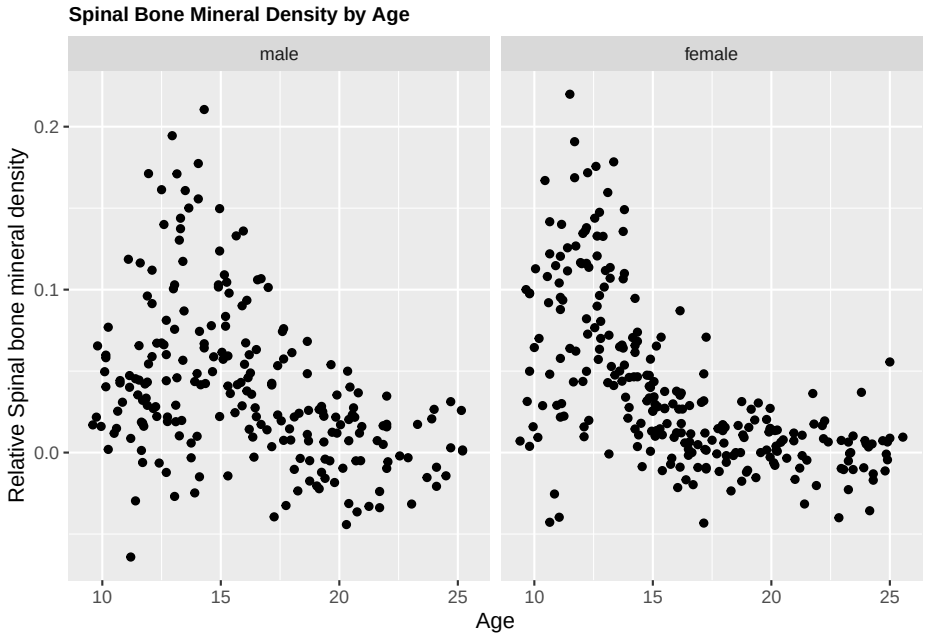


For spinal bone mineral density, the median is approximately equal for boys and girls. While both distibutions are skewed to the left, the male distribution is less skewed. Also, the male distribution has observations that are below -.05, while the female distrubtion does not.

The median age for males is slightly higher than the median for females. The box plots suggest that the ages of males are distributed farily normally, while the distribution of ages for females is slightly skewed to the right.

- Create a scatter plot of `spnbmd` (y axis) versus `age` (x axis), faceting by `gender`. What trends do you see in this data?

```
bmd %>%
  ggplot() +
  geom_point(aes(x = age, y = spnbmd)) + #creates scatter plot
  labs(
    x = "Age",
    y = "Relative Spinal bone mineral density"
  ) +
  facet_wrap(~ gender) + # split into facets based on age
  ggtitle("Spinal Bone Mineral Density by Age") + #adds title
  theme(plot.title = element_text(size = 10, face = "bold")) #formats title
```



The is not one clear trend. However, spinal bone density appears to peak at around the age of 12.5 for both males and females and then returns to around 0 0 between the ages of 17.5 and 20. Before age 12.5 bone mineral density appears to be increasing while after age 12.5, it appears to be on the decline. This trend is more clearly seen with females, where there appears to be less variation in the data

1.3 Model (15 points)

There are clearly some trends in this data, but they are somewhat hard to see given the substantial amount of variability. This is where splines come in handy.

1.3.1 Split

To ensure unbiased assessment of predictive models, let's split the data before we start modeling it.

- Split `bmd` into training (80%) and test (20%) sets, using the rows in `train_samples` below for training. Store these in tibbles called `bmd_train` and `bmd_test`, respectively.

```
set.seed(5) # seed set for reproducibility (DO NOT CHANGE)
smp_size <- floor(nrow(bmd)*.8)
train_id <- sample(seq_len(nrow(bmd)), size = smp_size)
bmd_train <- bmd[train_id, ]
bmd_test <- bmd[-train_id, ]
n = nrow(bmd)
train_samples = sample(1:n, round(0.8*n))
```

1.3.2 Tune

- Since the trends in `spnbmd` look somewhat different for boys than for girls, we might want to fit separate splines to these two groups. Separate `bmd_train` into `bmd_train_male` and `bmd_train_female`, and likewise for `bmd_test`.

```
bmd_train_male <- bmd_train %>%
  filter(gender == "male") # filters for only males
bmd_train_female <- bmd_train %>%
```

```
  filter(gender == "female") # filters for only females
bmd_test_male <- bmd_test %>%
  filter(gender == "male") # filters for only males
bmd_test_female <- bmd_test %>%
  filter(gender == "female") # filters for only females
```

- Using `cross_validate_spline` from Lecture 3, perform 10-fold cross-validation on `bmd_train_male` and `bmd_train_female`, trying degrees of freedom 1,2,...,15. Display the two resulting CV plots side by side.

```
#source function from lecture 3
source("~/Desktop/STAT471/stat-471-fall-2021/functions/cross_validate_spline.R")

#perform cross validation
cv_male = cross_validate_spline(bmd_train_male$age,
                                bmd_train_male$spnbmd,
                                nfolds = 10, #specify 10 fold
                                df_values = 1:15) #specify 1-15 df
```

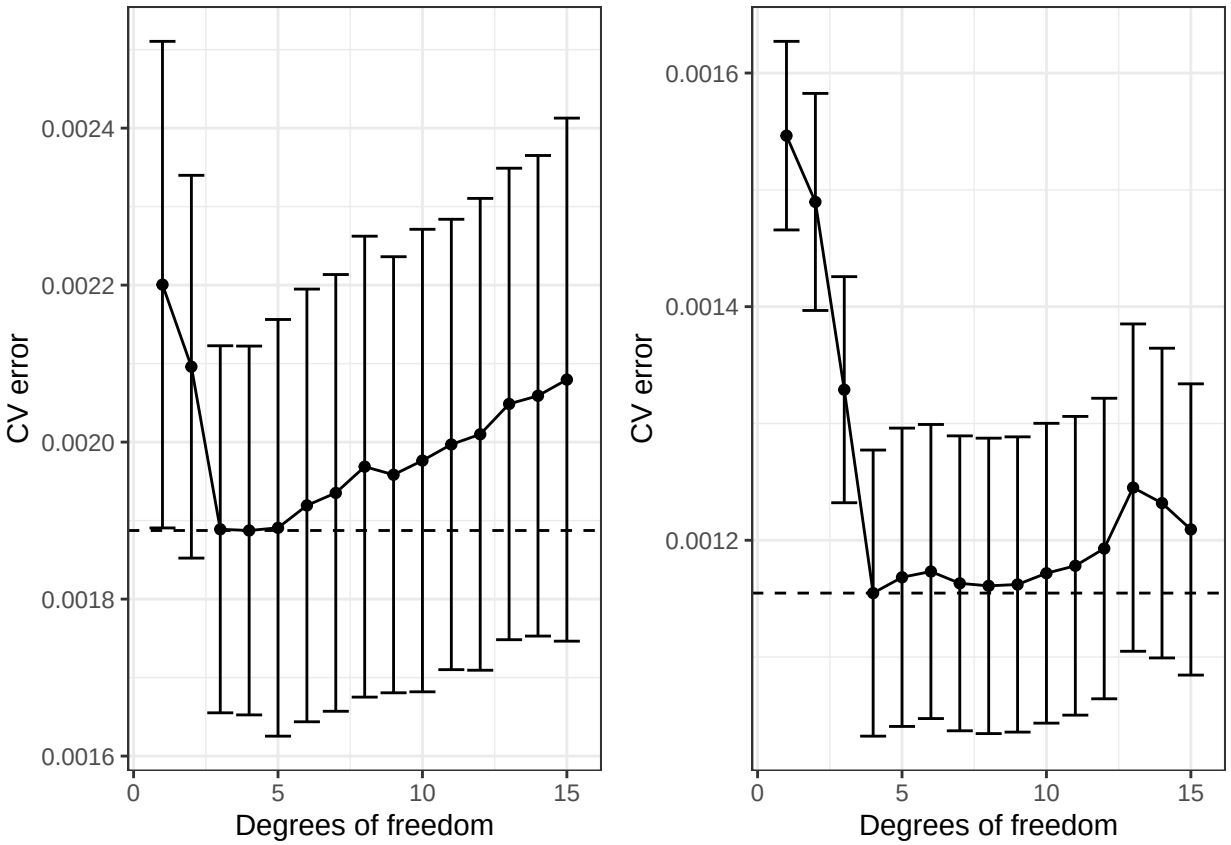
```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if `.name_repair` is
## Using compatibility `.name_repair`.
```

```
## `summarise()` has grouped output by 'df'. You can override using the `.groups` argument.
```

```
cv_female = cross_validate_spline(bmd_train_female$age,
                                bmd_train_female$spnbmd,
                                nfolds = 10, #specify 10 fold
                                df_values = 1:15) #specify 1-15 df
```

```
## `summarise()` has grouped output by 'df'. You can override using the `.groups` argument.
```

```
# plot 2 cv plots
plot_grid(cv_male$cv_plot, cv_female$cv_plot)
```



- What are the degrees of freedom values minimizing the CV curve for boys and girls, and what are the values obtained from the one standard error rule?

```
#male CV
print(cv_male$df.min) #value minimizng the CV curve
```

```
## [1] 4
```

```
print(cv_male$df.1se) #value from one standard error rule
```

```
## [1] 2
```

```
#female CV
print(cv_female$df.min) #value minimizng the CV curve
```

```
## [1] 4
```

```
print(cv_female$df.1se) #value from one standard error rule
```

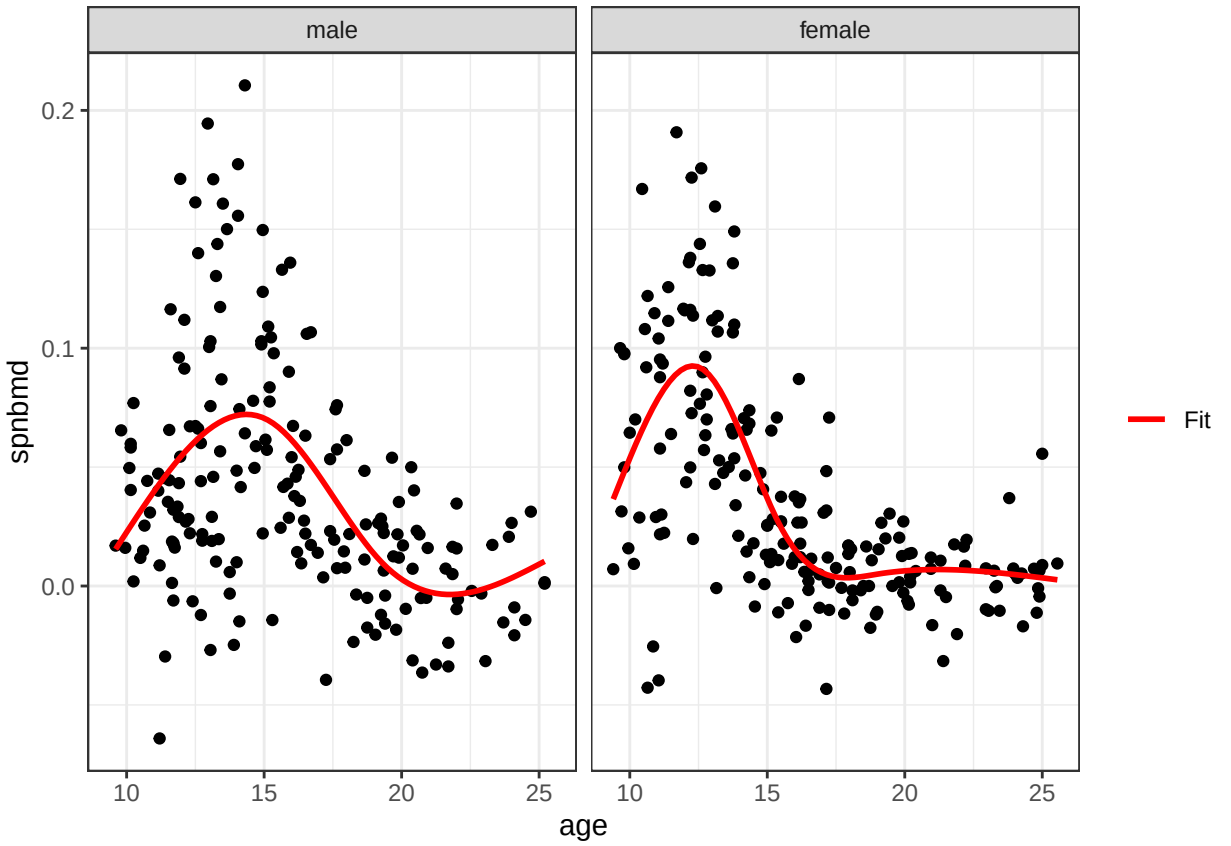
```
## [1] 4
```

For males, 5 df minimizes the CV curve, but 3 df is obtained from the one standard error rule
For females, 8 df minimizes the CV curve, but 3 df is obtained from the one standard error rule

- For the sake of simplicity, let's use the same degrees of freedom for males as well as females. Define `df.min` to be the maximum of the two `df.min` values for males and females, and define `df.1se` likewise. Add these two spline fits to the scatter plot of `spnbmd` (y axis) versus `age` (x axis), faceting by `gender`.

```
#max of df.min for males and females
df.min <- max(cv_male$df.min, cv_female$df.min)
#max of df.1se for males and females
```

```r
df.1se <- max(cv_male$df.1se, cv_female$df.1se)

# Creates split fit
bmd_train %>%
  ggplot(aes(age, spnbmd)) +
  geom_point() +
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.min)",
              aes(colour = "Fit"), se = FALSE) +
  scale_colour_manual(values = c("red", "blue")) +
  theme_bw() + theme(legend.title = element_blank()) +
  facet_wrap(~gender)
```



- Given our intuition for what growth curves look like, which of these two values of the degrees of freedom makes more sense? **3 degrees of freedom makes the most sense because growth curves an exponential phase, transitonal phase, and plateau phase. Thus, df are needed to represent each phase**

1.3.3 Final fit

- Using the degrees of freedom chosen above, fit final spline models to `bmd_train_male` and `bmd_train_female`.

```r
# Creates split fit for each gender with df.min degrees of freedom
spline_fit_male = lm(spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_male)
spline_fit_female = lm(spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_female)
```

Table 2: Traning vs. Test RMSE for Male Spline Fit

r_squared	RMSE_train	RMSE_test
0.28	0	0

1.4 Evaluate (6 points)

- Using the final models above, answer the following questions for boys and girls separately: What percent of the variation in `spnbmd` is explained by the spline fit in the training data? What is the training RMSE? What is the test RMSE? Print these three metrics in a nice table.

```
#calculates training and test predictions based on model
y_hat_train = predict(spline_fit_male, newdata = bmd_train_male)
y_hat_test = predict(spline_fit_male, newdata = bmd_test_male)

#calculates RMSE by taking mean of residuals for train
validation_error_train = bmd_train_male %>%
    cbind(y_hat_train) %>%
    summarise(RMSE_train = mean((y_hat_train-spnbmd)^2))

#calculates RMSE by taking mean of residuals for test
validation_error_test = bmd_test_male %>%
    cbind(y_hat_test) %>%
    summarise(RMSE_test = mean((y_hat_test-spnbmd)^2))

#combines train and test error and pull r^2 value from summary
male_model <- cbind(validation_error_train, validation_error_test) %>%
  summarise(r_squared = summary(spline_fit_male)$r.squared, RMSE_train, RMSE_test )

#formats chart
male_model %>%
  kable(format = "latex", row.names = NA,
      booktabs = TRUE, digits = 2,
      caption = "Traning vs. Test RMSE for Male Spline Fit") %>%
  kable_styling(position = "center")
```

The male RMSE for the train date is higher than RMSE for test data, which suggests that the model for males does not overfit

```
#calculates training and test predictions based on model
y_hat_train_fem = predict(spline_fit_female, newdata = bmd_train_female)
y_hat_test_fem = predict(spline_fit_female, newdata = bmd_test_female)

#calculates RMSE by taking mean of residuals for train
validation_error_train_fem = bmd_train_female %>%
    cbind(y_hat_train_fem) %>%
    summarise(RMSE_train = mean((y_hat_train_fem-spnbmd)^2))

#calculates RMSE by taking mean of residuals for test
validation_error_test_fem = bmd_test_female %>%
    cbind(y_hat_test_fem) %>%
    summarise(RMSE_test = mean((y_hat_test_fem-spnbmd)^2))

#combines train and test error and pull r^2 value from summary
```

Table 3: Traning vs. Test RMSE for Female Spline Fit

r_squared	RMSE_train	RMSE_test
0.51	0	0

```
female_model <- cbind(validation_error_train_fem, validation_error_test_fem) %>%
  summarise(r_squared = summary(spline_fit_female)$r.squared, RMSE_train, RMSE_test)

#formats chart
female_model %>%
  kable(format = "latex", row.names = NA,
      booktabs = TRUE, digits = 2,
      caption = "Traning vs. Test RMSE for Female Spline Fit") %>%
  kable_styling(position = "center")
```

- How do the training and test errors compare? What does this suggest about the extent of overfitting that has occurred? **The female RMSE for the train date is ~40% lower than RMSE for test data, which suggests that the model is overfitting**

1.5 Interpret (7 points)

- Using the degrees of freedom chosen above, redo the scatter plot with the overlaid spline fits, this time without faceting in order to directly compare the spline fits for boys and girls. Instead of faceting, distinguish the genders by color.

```
bmd_train %>%
  ggplot(aes(age, spnbmd, color = gender)) +
  geom_point() +
  # shows spline fit for boys and girls with df.min degrees of freedom
  geom_smooth(method = "lm",
              formula = "y ~ splines::ns(x, df = df.min)", se = FALSE) +
  #sets colors to blue and pink
  scale_colour_manual(values = c("blue", "magenta")) +
  theme_bw() + theme(legend.title = element_blank()) +
  labs(
    x = "Age",
    y = "Relative Spinal Bone Mineral Density"
  ) +
  ggtitle("Bone Density by Age")  #adds title
```

- The splines help us see the trend in the data much more clearly. Eyeballing these fitted curves, answer the following questions. At what ages (approximately) do boys and girls reach the peaks of their growth spurts? At what ages does growth largely level off for boys and girls? Do these seem in the right ballpark?

Girls rech their peak growth spurts at around age 12 while boys reach the peaks of their growth spurts at around age 14.These numbers seem accurate because they are relatively close to the age each gender reaches puberty at

2 KNN and bias-variance tradeoff (45 points for correctness; 5 points for presentation)

Setup: Apple farming

You own a square apple orchard, measuring 200 meters on each side. You have planted trees in a grid ten meters apart from each other. Last apple season, you measured the yield of each tree in your orchard (in average apples per week). You noticed that the yield of the different trees seems to be higher in some places of the orchard and lower in others, perhaps due to differences in sunlight and soil fertility across the orchard.

Unbeknownst to you, the yield Y of the tree planted X_1 meters to the right and X_2 meters up from the bottom left-hand corner of the orchard has distribution

$$Y = 50 + 0.001X_1^2 + 0.001X_2^2 + \epsilon, \quad \epsilon \sim N(0, \sigma^2), \quad \sigma = 4.$$

The data you collected are as in Figure 1.

The underlying trend is depicted in Figure 2, with the top right-hand corner of the orchard being more fruitful.

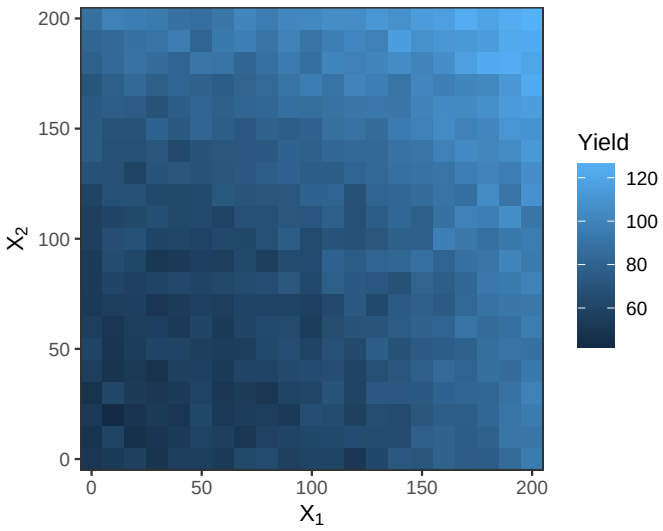


Figure 1: Apple tree yield for each 10m by 10m block of the orchard in a given year.

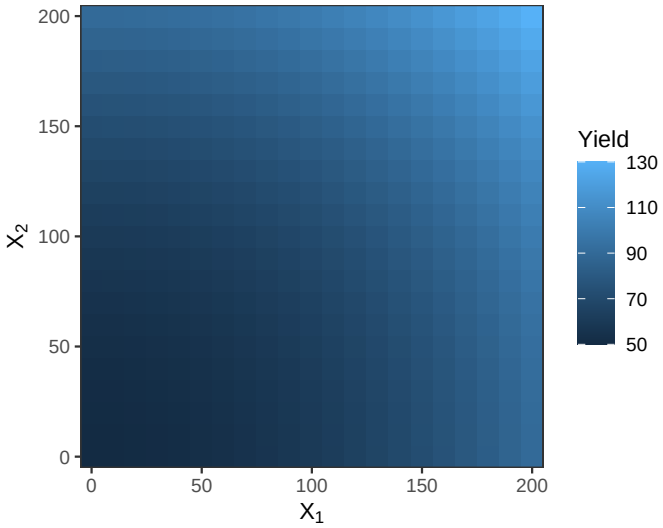


Figure 2: Underlying trend in apple yield for each 10m by 10m block of the orchard.

2.1 A simple rule to predict this season's yield (15 points)

This apple season is right around the corner, and you'd like to predict the yield of each tree. You come up with perhaps the simplest possible prediction rule: predict this year's yield for any given tree based on last year's yield from that same tree. Without doing any programming, answer the following questions:

- What is the expected training error of such a rule? **Training error will be zero because the model has essentially fit every data point to itself**
- Averaged across all trees, what is the squared bias, variance, and ETE of this prediction rule? **Squared bias is 0. Variance is (10+20+30 DID NOT FINISH THIS ONE DID NOT FINISH THIS ONE ALERT**
- Why is this not the best possible prediction rule? **While it works well on the training data set, it is not useful in predicting on the test data set because the values of each individual point are not known. Thus, the model is generally useless in making predicitons.**

2.2 K-nearest neighbors regression (conceptual) (15 points)

As a second attempt to predict a yield for each tree, you average together last year's yields of the K trees closest to it (including itself, and breaking ties randomly if necessary). So if you choose $K = 1$, you get back the simple rule from the previous section. This more general rule is called *K-nearest neighbors (KNN) regression* (see ISLR p. 105).

KNN is not a parametric model like linear or logistic regression, so it is a little harder to pin down its degrees of freedom.

- What happens to the model complexity as K increases? Why? **Model complexity is how close the mdoel fits the traning data. Model complexity increases exponentially because for every 1 unit increase in K, one more tree's yield is used in the computation of each tree's yield. This means 400 more trees are used in the computation of the yields for all 400 trees. If K was increased by 5 units, 2000 more trees would be used in the computation.**
- The degrees of freedom for KNN is sometimes considered n/K, where n is the training set size. Why might this be the case? [Hint: consider a situation where the data are clumped in groups of K.] **Since the predicted value is equal to the average of K y-values and the sample point must be one of the included y-values, it makes sense that df = n/K. When K = n, the model will give just one prediction, since every data point in the cluster is averaged, which is the same as having just one parameter. When K = 1, the model outputs n different predicitons, which is the same as a model with n predictions.**
- Conceptually, why might increasing K tend to improve the prediction rule? What does this have to do with the bias-variance tradeoff? **Larger K values tend to improve the prediction rule because they will have smoother decision boundaries, which means lower variance but increased bias. On the extreme side, if K = n, the model will predict the same y-value for every tree, which would give a variance of 0 but have high bias**
- Conceptually, why might increasing K tend to worsen the prediction rule? What does this have to do with the bias-variance tradeoff? **Bias is the difference between the true label and the prediction, while variance is the expectation of the squared deviation of a random variable from its mean. Therefore, increasing K would reduce variance since predictions would become closer to each other. However, it would increase bias because the model now has a higher potential to overfit and predict values that are far from their true value. Therefore, when K increases, the training error will increase (which increases bias), but the test error may decrease at the same time (which decreases variance).**

2.3 K-nearest neighbors regression (simulation) (15 points)

Now, we try KNN for several values of K. For each, we compute the bias, variance, and ETE for each value based on 50 resamples. The code for this simulation, provided for you below (see Rmd file; code omitted from PDF for brevity), results in Figure 3.

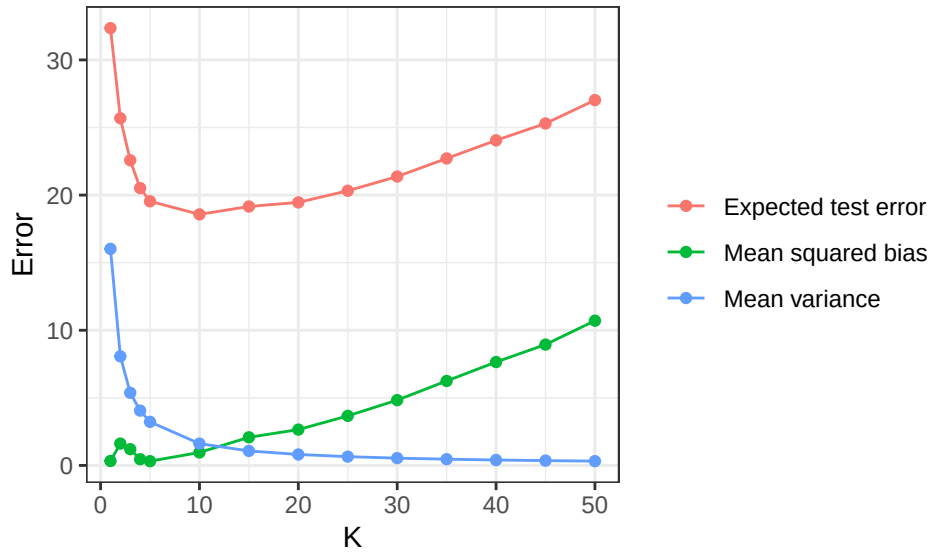


Figure 3: Bias-variance trade-off for KNN regression.

- Based on Figure 3, what is the optimal value of `K`?

```
#identifies which K has the highest ETE
overall_results$K[which.max(overall_results$expected_test_error)]
```

```
## [1] 1
```

The optimal value of K is 10 because ETE is minimized when K=10

- We are used to the bias decreasing and the variance increasing when going from left to right in the plot. Here, the trend seems to be reversed. Why is this the case? **When K increases, the predicted y-values get closer to each other because more points in the cluster are used to compute the y-value, which is an average of the K-nearest points. Therefore, when K increases, the training error will increases because the y-values are further away from their true values (which increases bias), but variance decreases because the predicted points become closer to each other. In fact, if K=n, mean variance will be 0 for the train data set**

- The squared bias has a strange bump between `K` = 1 and `K` = 5, increasing from `K` = 1 to `K` = 2 but then decreasing from `K` = 2 to `K` = 5. Why does this bump occur? [Hint: Think about the rectangular grid configuration of the trees. So for a given tree, the closest tree is itself, and then the next closest four trees are the ones that are one tree up, down, left, and right from it.] **When K=2, one tree out of the four surrounding point is randomly selected to be averaged with y-value of the tree itself. If the surrounding data point that is most different from the point we are trying to pick is chosen, bias will increase. In constrastm when 3-5 data points are averaged to predict the y-value, the difference between the selected data point and the data points used for prediction decreaes.**

- The data frame `training_results_summary` contains the bias and variance for every tree in the orchard, for every value of `K`. Which tree and which value of `K` gives the overall highest absolute bias? Does the sign of the bias make sense? Why do this particular tree and this particular value of `K` give us the

largest absolute bias?

```
#identifies which row has the highest absolute sq bias
training_results_summary[which.max(abs(overall_results$mean_sq_bias)), ]
```

```
## # A tibble: 1 x 5
##       K    X1    X2   bias variance
##   <int> <dbl> <dbl>  <dbl>    <dbl>
## 1     1     0   130 -0.793     20.7
```

The tree located at (0,130) when K = 1 gives the largest absolute bias. The bias is -0.793. The sign of the bias is negative because the predicted y-value was lower than the actual y-value. This particular tree has the largest bias because I DO NOT KNOW I DO NOT KNOW

- Redo the bias-variance plot above, this time putting `df = n/K` on the x-axis. What do we notice about the variance as a function of `df`? Derive a formula for the KNN variance and superimpose this formula onto the plot as a dashed curve. Do these two variance curves match? [Hint: To derive the KNN variance, focus first on the prediction of a single tree. Recall the facts that the variance of the sum of independent random variables is the sum of their variances, and that the variance of a constant multiple of a random variable is the square of that constant times its variance.]

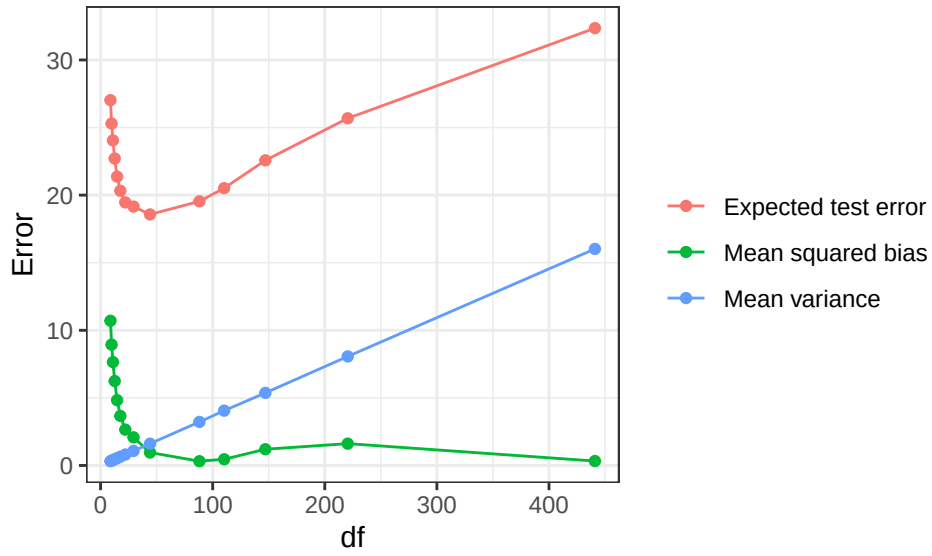


Figure 4: Bias-variance trade-off for KNN regression.

Yes, the two curves match. Since df = n/K, when the df is approaching 0, K is increasing and when df is increasing, K is decreasing. We see that increasing K increases bias and the expected test error. Also, when K increases, the mean variance approaches zero beacuse K = n. Thus, all estimations are the same. When K decreases, we see ETE and bias increase. As K decereases, bias decreases because K approaches 1.