STAT 471: Homework 2

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) . . . 12

2 KNN and bias-variance tradeoff (45 points for correctness; 5 points for presentation) **13**
- Setup: Apple farming . . . 13
- 2.1 A simple rule to predict this season's yield (15 points) . . . 13
- 2.2 K-nearest neighbors regression (conceptual) (15 points) . . . 14
- 2.3 K-nearest neighbors regression (simulation) (15 points) . . . 16

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:

Please list what external references you consulted (e.g. articles, books, or websites):

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`).

```
# import the data from URL
bmd = read_tsv("https://web.stanford.edu/~hastie/ElemStatLearn/datasets/bone.data",
               col_types = "idfd")
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`.
- 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?

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

```
# remove idnum column from bmd
bmd = bmd %>% select(-idnum)
```

```
# determine number of males and females
bmd %>% group_by(gender) %>% count(gender) %>%
  kable(format = "latex", row.names = NA, booktabs = TRUE,
        digits = 2, col.names = c("Gender", "Count"),
        caption = "These are the number of males and females
        in the data set.") %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 1: These are the number of males and females in the data set.

Gender	Count
male	226
female	259

```
# determine median ages of males and females
bmd %>% group_by(gender) %>% summarize(median(age)) %>%
  kable(format = "latex", row.names = NA, booktabs = TRUE,
        digits = 2, col.names = c("Gender", "Median age"),
        caption = "These are the median ages of males and females
        in the data set.") %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 2: These are the median ages of males and females in the data set.

Gender	Median age
male	15.6
female	15.3

We can observe that (ignoring the fact that there are repeated measurements) the number of boys and girls in this dataset is 226 and 259 respectively, as shown in Table 1. The median age of these boys and girls is 15.625 and 15.350 respectively as seen in Table 2.

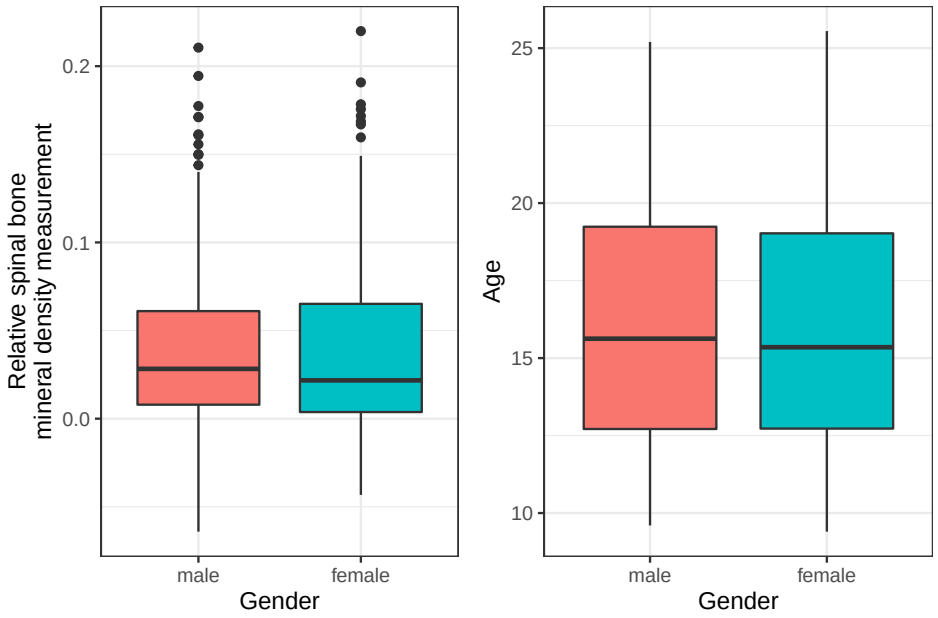

Figure 1: On the left are boxplots for the distribution of spnbmd (relative spinal bone mineral density measurement) by gender, and on the right are boxplots for the distribution of age (average age of child when measurements were taken) by gender. The distributions for both variables appear similar across genders.

From Figure 1, we can observe that the median relative spinal bone mineral density measurement is slightly higher for males than females. However, we can see that the minimum male relative spinal bone mineral density measurement is lower than that for females, with the whisker at the bottom extending further for males than females. The interquartile range for females is a bit larger (i.e., the area in which the middle 50% of the female data lies). Apart from these small differences, the distributions appear quite similar for males and females in terms of relative spinal bone mineral density measurement.

Furthermore, we can observe (from Figure 1) that the median male age in the data is just slightly higher that for females in the data, but the difference is quite marginal. We can also observe that the interquartile ranges for the groups are very similar. Additionally, the overall range for both groups is quite similar, with the female distribution for age having a slightly higher maximum age in the data. Overall, the distributions are quite similar to one another between males and females.

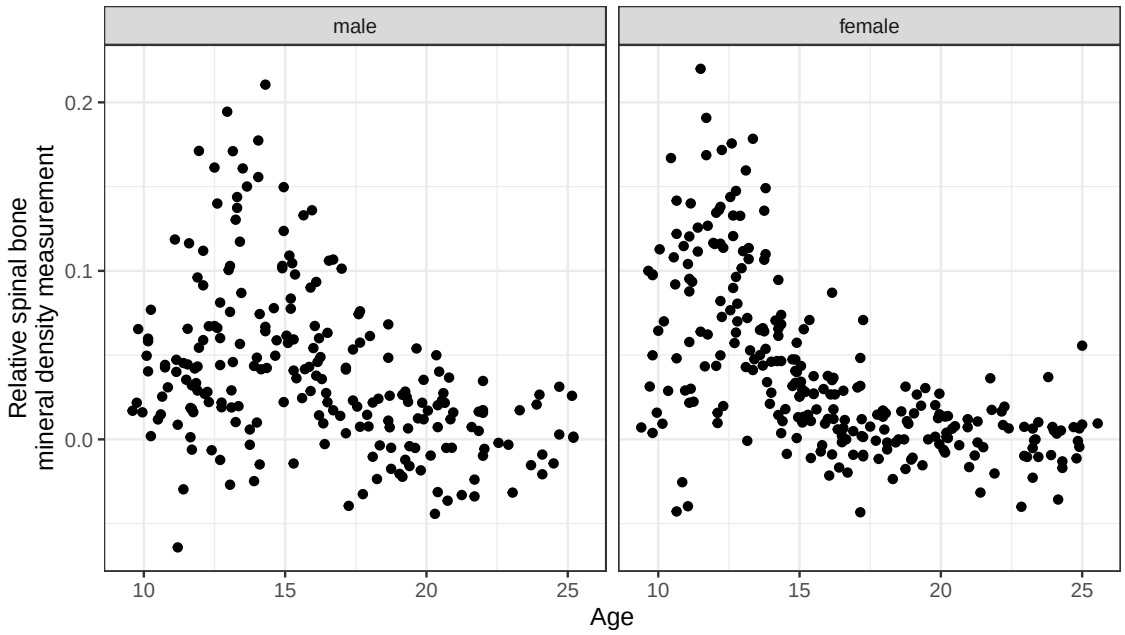

Figure 2: These are scatter plots of relative spinal bone mineral density measurement versus age, broken up by gender (male or female). Both genders have a curve that appears to show exponential decay in terms of the measurement as age increases.

From Figure 2, we can observe that for both males and females, there appears to be a somewhat exponential shaped decay of the relative spinal bone mineral density measurement as age increases. So for older children, they have lower relative spiral bone mineral density measurements, but this measurement plateaus such that children age 20 vs. children age 25, for example, do not tend to significantly differ. Additionally, for children of younger ages (e.g., 10-15), there appears to be more variation in the relative spinal bone mineral density measurement, whereas children who are older (e.g., 15-25) have a relative spinal bone mineral density that seems to concentrate in a smaller range (lying between about -0.05 and 0.05). However, the female data follows this exponential decay shaped trend more closely, as can be seen by the fact that the points exhibit this more well-defined shape (as in Figure 2), whereas there appears to be more variation and deviation from this trend for males. Both genders have similar ranges of relative spinal bone mineral density measurements across the ages (over time).

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)
n = nrow(bmd)
train_samples = sample(1:n, round(0.8*n))
```

```
# split bmd into training and test sets
bmd_train = bmd[train_samples,]
bmd_test = bmd[-train_samples,]
```

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`.
- 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.
- 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?
- 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`.
- Given our intuition for what growth curves look like, which of these two values of the degrees of freedom makes more sense?

```
# separate bmd_train into bmd_train_male and bmd_train_female
bmd_train_male = bmd_train[bmd_train["gender"] == "male",]
bmd_train_female = bmd_train[bmd_train["gender"] == "female",]

# separate bmd_test into bmd_test_male and bmd_test_female
bmd_test_male = bmd_test[bmd_test["gender"] == "male",]
bmd_test_female = bmd_test[bmd_test["gender"] == "female",]
```

```
# perform 10-fold cross-validation on bmd_train_male and bmd_train_female
cv_out_male = cross_validate_spline(bmd_train_male$age,
                                    bmd_train_male$spnbmd,
                      nfolds = 10, df_values = 1:15)
cv_out_female = cross_validate_spline(bmd_train_female$age,
                                      bmd_train_female$spnbmd,
                      nfolds = 10, df_values = 1:15)
```

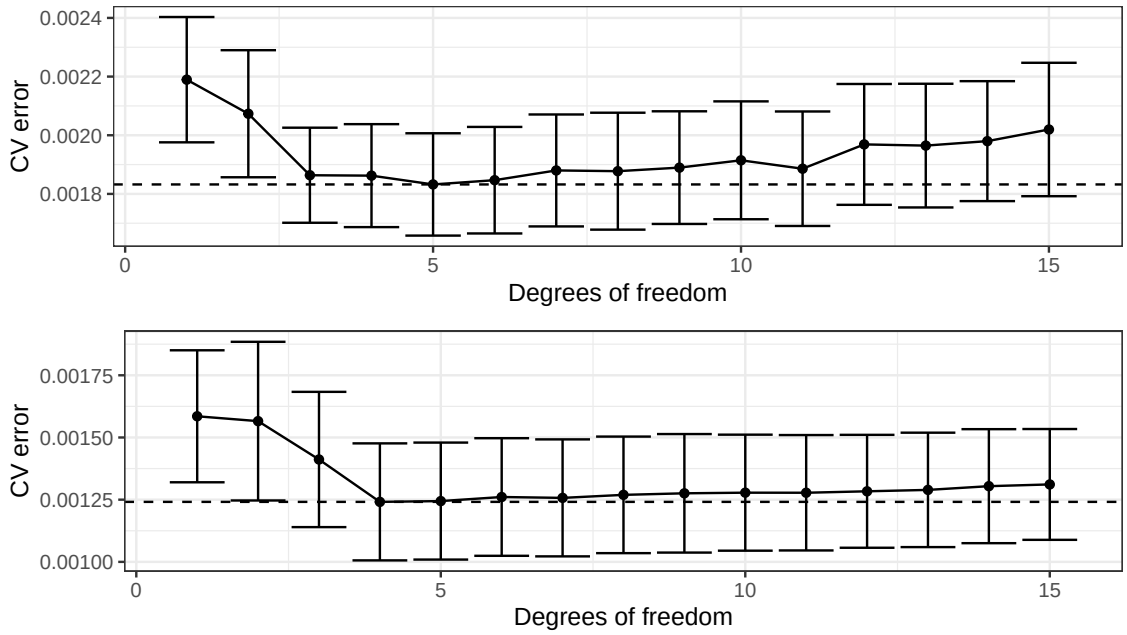

Figure 3: These are the cross validation plots illustrating the error corresponding to various degrees of freedom levels for the boys and girls training data respectively, i.e., the top graph corresponds to boys, and the bottom graph corresponds to girls.

```
# determine the dfs minimizing cv curves
male_min_df = cv_out_male$df.min
female_min_df = cv_out_female$df.min
sprintf("The df minimizing the CV curve for boys is %i", male_min_df)
```

```
## [1] "The df minimizing the CV curve for boys is 5"
```

```
sprintf("The df minimizing the CV curve for girls is %i", female_min_df)
```

```
## [1] "The df minimizing the CV curve for girls is 4"
```

```
# determine the dfs obtained from the one standard error rule
male_1se_df = cv_out_male$df.1se
female_1se_df = cv_out_female$df.1se
sprintf("The df obtained from the one standard error rule for boys is %i",
        male_1se_df)
```

```
## [1] "The df obtained from the one standard error rule for boys is 3"
```

```
sprintf("The df obtained from the one standard error rule for girls is %i",
        female_1se_df)
```

```
## [1] "The df obtained from the one standard error rule for girls is 3"
```

The degrees of freedom values minimizing the CV curve (minimizing the mean CV error) for boys and girls are 5 and 4 degrees of freedom respectively. (These are also the points that exactly meet the horizontal dashed lines on the plots in Figure 3.) The values obtained from the one standard error rule are 3 and 3 degrees of freedom for boys and girls respectively, i.e., the optimal degrees of freedom based on the one standard error rule is the same for the male training data set and the female training data set.

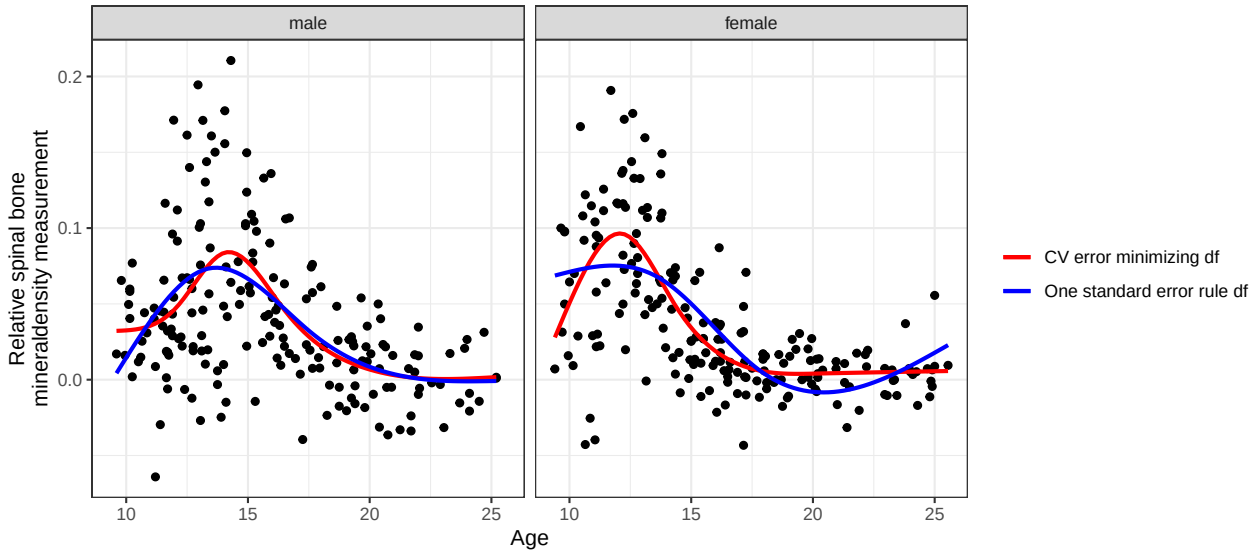

Figure 4: This is a scatter plot of relative spinal bone mineral density measurement versus age faceted by gender with overlaid spline fits. In red, we have the spline fit corresponding to the CV error minimizing degrees of freedom parameter value. In blue, we have the spline fit corresponding to the one standard error rule degrees of freedom parameter value.

The red curve in Figure 4 corresponding to the CV error minimizing degrees of freedom selection for the spline fit appears to make more sense. (That is, the curve with five degrees of freedom appears to make more sense.) This is evident because in the right graph, i.e., for females, the blue curve corresponding to the one standard error degrees of freedom selection for the spline fit tends upwards past the age of 20. This does not make sense given intuition that we do not expect additional increases in growth after a certain point. Thus, the red curve, which in both facets rises during puberty and appears to plateau over time makes the most sense given our intuition for what growth curves look like.

1.3.3 Final fit

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

```
# fit final spline models to bmd_train_male and bmd_train_female
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)
summary(spline_fit_male)
```

```
##
## Call:
## lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_male)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.10760 -0.02393 -0.00384  0.02285  0.12782
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)                      0.03221    0.01687    1.91    0.058 .
## splines::ns(age, df = df.min)1   0.07452    0.01853    4.02  8.4e-05 ***
## splines::ns(age, df = df.min)2 -0.00173    0.02374   -0.07    0.942
## splines::ns(age, df = df.min)3 -0.03524    0.01786   -1.97    0.050 *
## splines::ns(age, df = df.min)4 -0.03089    0.03979   -0.78    0.439
## splines::ns(age, df = df.min)5 -0.03129    0.01802   -1.74    0.084 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0424 on 186 degrees of freedom
## Multiple R-squared:  0.302,  Adjusted R-squared:  0.283
## F-statistic: 16.1 on 5 and 186 DF,  p-value: 3.51e-13
```

```
summary(spline_fit_female)
```

```
##
## Call:
## lm(formula = spnbmd ~ splines::ns(age, df = df.min), data = bmd_train_female)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12307 -0.01655 -0.00046  0.01638  0.10068
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.02787    0.01231    2.26  0.02470 *
## splines::ns(age, df = df.min)1   0.00833    0.01387    0.60  0.54868
## splines::ns(age, df = df.min)2 -0.02502    0.01780   -1.41  0.16158
## splines::ns(age, df = df.min)3 -0.05663    0.01472   -3.85  0.00016 ***
## splines::ns(age, df = df.min)4  0.06308    0.02997    2.11  0.03660 *
## splines::ns(age, df = df.min)5 -0.07475    0.01286   -5.81  2.6e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0342 on 190 degrees of freedom
## Multiple R-squared:  0.511,  Adjusted R-squared:  0.498
## F-statistic: 39.7 on 5 and 190 DF,  p-value: <2e-16
```

Using the degrees of freedom chosen above, i.e., five, we have the summaries of the final fitted spline models to `bmd_train_male` and `bmd_train_female` immediately above.

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.
- How do the training and test errors compare? What does this suggest about the extent of overfitting that has occurred?

```
# determine % of spnbmd variation explained by spline fit in training data
m_r_squared = summary(spline_fit_male)$r.squared
f_r_squared  = summary(spline_fit_female)$r.squared

# determine training RMSE
m_training_RMSE = bmd_train_male %>%
```

```
  mutate(fitted_value = spline_fit_male$fitted.values) %>%
  summarise(training_error = sqrt(mean((spnbmd-fitted_value)^2))) %>%
  pull(training_error)
f_training_RMSE = bmd_train_female %>%
  mutate(fitted_value = spline_fit_female$fitted.values) %>%
  summarise(training_error = sqrt(mean((spnbmd-fitted_value)^2))) %>%
  pull(training_error)

# determine test RMSE
predictions_male = predict(spline_fit_male, newdata = bmd_test_male)
m_test_RMSE = bmd_test_male %>% mutate(prediction = predictions_male) %>%
  summarise(test_error = sqrt(mean((spnbmd-prediction)^2))) %>%
  pull(test_error)

predictions_female = predict(spline_fit_female, newdata = bmd_test_female)
f_test_RMSE = bmd_test_female %>% mutate(prediction = predictions_female) %>%
  summarise(test_error = sqrt(mean((spnbmd-prediction)^2))) %>%
  pull(test_error)

# create table of these three metrics for boys and girls
model_metrics = tribble(
  ~gender, ~r_squared, ~training_rmse, ~test_rmse,
  #------/-----------/---------------/------------
  "Male", m_r_squared, m_training_RMSE, m_test_RMSE,
  "Female", f_r_squared, f_training_RMSE, f_test_RMSE
  )

# print these metrics in nice table
model_metrics %>% kable(format = "latex", row.names = NA,
                        booktabs = TRUE,
                        digits = 2,
                        col.names = c("Gender",
                                      "R-squared",
                                      "Training RMSE",
                                      "Test RMSE"),
                        caption = "These are the three metrics R-squared,
                        training RMSE, and test RMSE for the final models
                        for boys and girls.") %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 3: These are the three metrics R-squared, training RMSE, and test RMSE for the final models for boys and girls.

Gender	R-squared	Training RMSE	Test RMSE
Male	0.30	0.04	0.03
Female	0.51	0.03	0.04

As can be seen in Table 3, for boys and girls, the percent of the variation in `spnbmd` explained by the spline fit in the training data is 30.212% and 51.11% respectively. The training RMSE is 0.042 and 0.034 respectively, and the test RMSE is 0.034 and 0.038 respectively.

We can see in Table 3 that the training error is actually higher than the test error for males. (But the training error and test error are quite close, so this directional difference is not really any reason to be alarmed. Though on average we would expect training error to be less than test error, on a given realization of training and test set it is possible to have test error less than training error, especially if the model being fit is not very complex. However, for females, the training error is lower than the test error as would typically be expected. Given that the RMSE metrics for training and testing are rather close to one another, this does not suggest that there is any gross overfitting occurring. (If the model being fit is not very complex, then there is not much overfitting going on, i.e., we are not fitting the training set too much more closely than the test set. Thus, it is not implausible for the test error to be smaller than the training error.) If there was overfitting, the test error would be significantly higher than the training error, which in such a case would suggest we were building a model that adapted too much to the wiggles in the training data. (And when underfitting, typically both training error and testing error are large.)

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.
- 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?

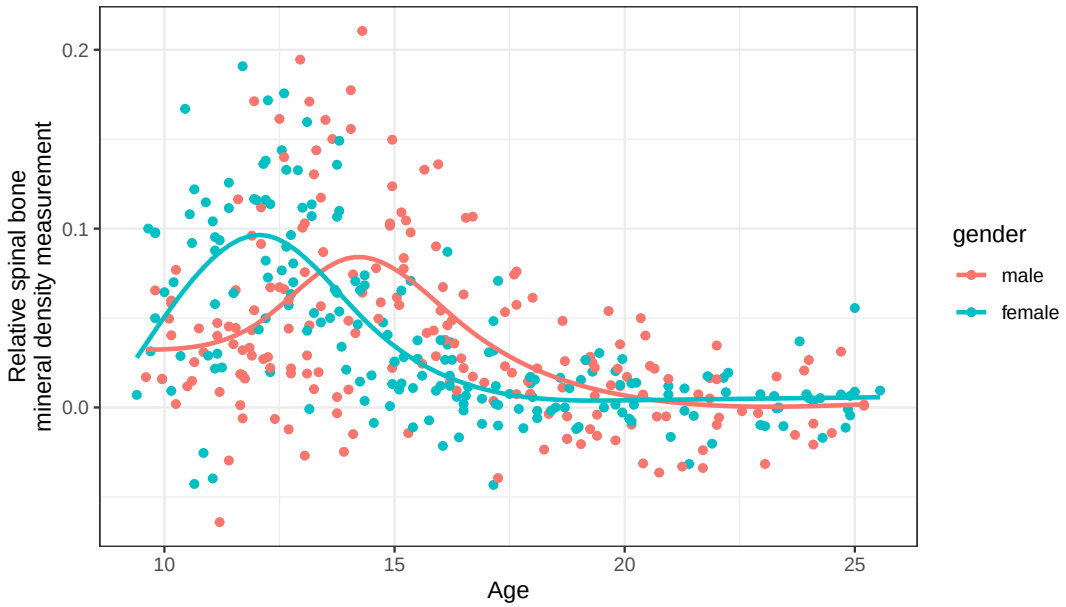

Figure 5: This is a scatter plot of relative spinal bone mineral density measurement versus age with overlaid spline fits. The difference in color of the points allows us to directly compare the spline fits for boys and girls.

Based on Figure 5, we can see that girls tend to reach the peak of their growth spurt at around the age of 12, while boys tend to reach the peak of their growth spurt at around the age of 14. Growth largely levels off for girls at around 17 or 18 years of age, while for boys, growth largely levels off at closer to around 20 years of age. These estimates seem to be in the right ballpark, especially as we expect girls to go through their growing period/puberty earlier than

boys.

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 6.

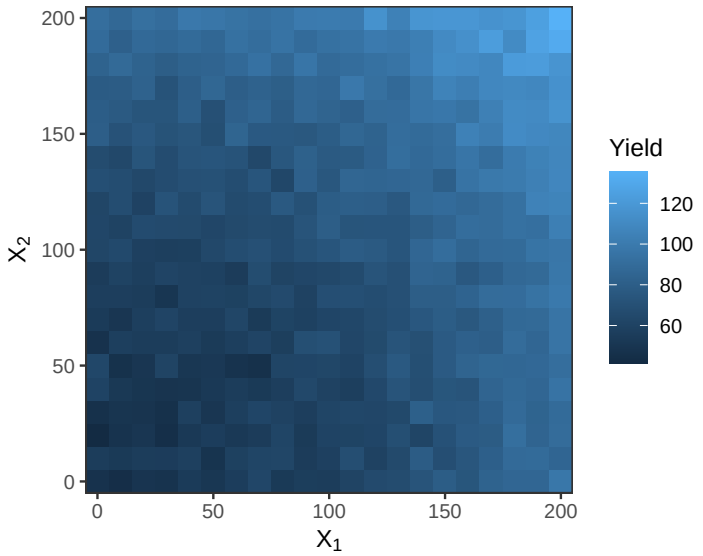

Figure 6: Apple tree yield for each 10m by 10m block of the orchard in a given year.

The underlying trend is depicted in Figure 7, with the top right-hand corner of the orchard being more fruitful.

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? **The expected training error of such a rule is zero. This is because the expected training error is effectively corresponds to the average of the squared differences of the predicted Y train values and the actual Y train values. With this rule, these squared differences are each zero (as predicted = actual), so therefore, the average of these squared differences is zero. Thus, the expected training error is zero. (That is, essentially we are comparing training rule values (prediction) and training values, and ultimately predicting exactly what we have.)**

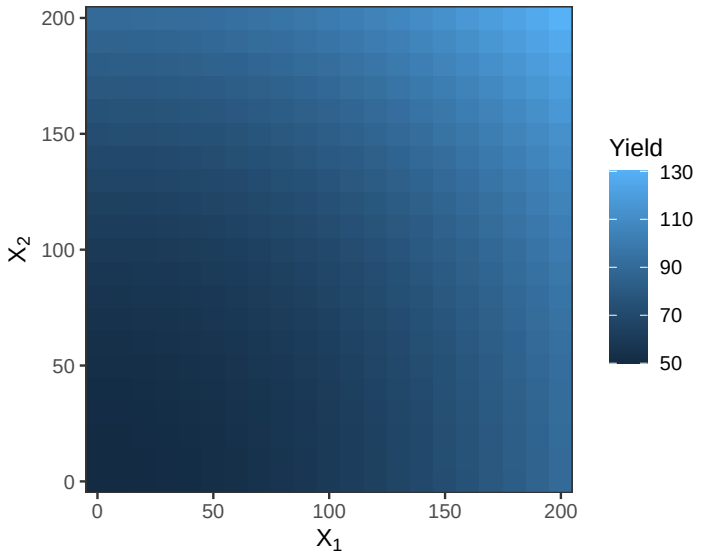

Figure 7: Underlying trend in apple yield for each 10m by 10m block of the orchard.

- Averaged across all trees, what is the squared bias, variance, and ETE of this prediction rule? **The expected test error of this prediction rule is $2\sigma^2 = 2 * 16 = 32$. This expected test error is derived as follows: We know that the ETE is composed of (i.e., is the sum of) the "bias", variance, and irreducible error. There is no reason for us to expect any bias since we have no reason to believe the previous year's yield is biased in any particular direction. (Moreover, the squared bias is effectively the squared difference of the expected value of $\hat{f}(x)$ and $f(x)$, and we have that $\hat{f}(x) = f(x_1, x_2) + \epsilon$ (so the expected value of $\hat{f}(x)$ is equal to the expected value of $f(x_1, x_2)$ plus the expected value of ϵ) and $f(x) = f(x_1, x_2)$. The expected value of $f(x_1, x_2)$ is $f(x_1, x_2)$, and so we obtain $E[f(x_1, x_2)] + E[\epsilon] - f(x_1, x_2) = f(x_1, x_2) + 0 - f(x_1, x_2) = 0$.) So the squared bias is zero. For variance, we take the average of the variances for the Y train values (distributed based on the error of $N(0, \sigma^2)$. Thus, we have n (corresponding to the sample size) such variances (each σ^2) divided by n, so we obtain σ^2. Then, for our irreducible error we have σ^2 (simply based on the error of $N(0, \sigma^2)$). Thus, we obtain $0 + \sigma^2 + \sigma^2 = 2\sigma^2 = 2 * 4^2 = 2 * 16 = 32$.**
- Why is this not the best possible prediction rule? **This is not the best possible prediction rule, as the variance is very high ($\sigma^2 = 16$), contributing to a high expected test error. Moreover, we are using a single tree for prediction, contributing to high variance. (That is, only taking this single tree's yield into account is not very good for prediction.) We could perhaps achieve a lower variance using KNN, as if trees are similar to their neighbors, averaging over such values decreases the variance such that we obtain better prediction. (Also, while not central to this question, from a more conceptual point of view, there may be fluctuations in yield over longer spans of time that are not adequately captured by simply using the previous year's yield for prediction. There also may be more variables such as precise location, weather, and soil fertility that may contribute to the expected yield that could be taken into account to enable better prediction.)**

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? **As K increases, model complexity decreases. When K is larger, you are essentially averaging together larger swaths of data. (This is such that the differences among points within such groups of neighbors are not being captured and are instead assigned the same prediction.) Thus, a value of K that is too large does not have enough complexity to fit the data well, whereas a model with a moderate value of K can capture the trend in the data without overfitting. In the case of grossly high model complexity, i.e., $K = 1$ (fitting one nearest neighbor), the model is overfit to the data, and the model is adapting too much to the noise in the data. Thus, ultimately, for higher values of K, we are looking at bigger neighborhoods, implying a less complex model. In the extreme, taking K to be all the points is sort of the analog of fitting a constant model. If we alternatively have a very low value for K, we are only looking at few nearest neighbors, thus averaging over fewer things, such that we are fitting a more complex model.**
- 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.] **The degrees of freedom for KNN may sometimes be considered n/K, where n is the training set size because if neighborhoods do not overlap, there would be n/K neighborhoods with one label, i.e., parameter, for each. That is, if the neighborhoods are non-overlapping, there would be n/K neighborhoods, and we fit one parameter (mean/average) in each neighborhood.**
- Conceptually, why might increasing K tend to improve the prediction rule? What does this have to do with the bias-variance tradeoff? **Increasing K might tend to improve the prediction rule because it decreases variance. (We can consider the extreme case $K = 1$ (lower bound), in which case we will perfectly predict our training data. In such a case the bias will be 0, but for our test data, there is greater chance of error, increasing variance. When we increase K, the training error increases (increases bias), but the test error may decrease (as variance decreases). A larger K can be associated with a simpler model, which can be associated with higher bias.) For smaller values of K, there are more wiggles/jaggedness, indicating higher variance and illustrates adaptation to the particular training data set (rather than more simply the trend). For larger values of K the curve may be more smooth and there is less variance, but the mismatch with the boundary line indicates higher bias. Thus, increasing K can tend to improve the prediction rule since you are averaging over more neighbors to obtain a more accurate prediction taking more trees into account (such that as those neighbors are similar, we have decreasing variance).**
- Conceptually, why might increasing K tend to worsen the prediction rule? What does this have to do with the bias-variance tradeoff? **Increasing K might tend to worsen the prediction rule because it increases bias. Increasing K results in averaging more points in each prediction, resulting in smoother prediction curves. More distant neighbors are included in the estimates for a particular prediction. (As K grows larger and larger, the distinction between categories becomes more blurred and the boundary line for prediction is not well-matched. That is, we are assigning a single prediction to a group of points, averaging together points that may not be so similar (as we average more and more trees together) such that we end up with an average prediction that is further from the truth.) Ultimately, increasing K will decrease variance and increase bias, while decreasing K will increase variance and decrease bias. Predictions are more variable for lower K, and when K increases, we reduce this variability. When K is increased too much, we move away from the true boundary line and we have higher bias. This reflects the bias-variance tradeoff.**

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 8.

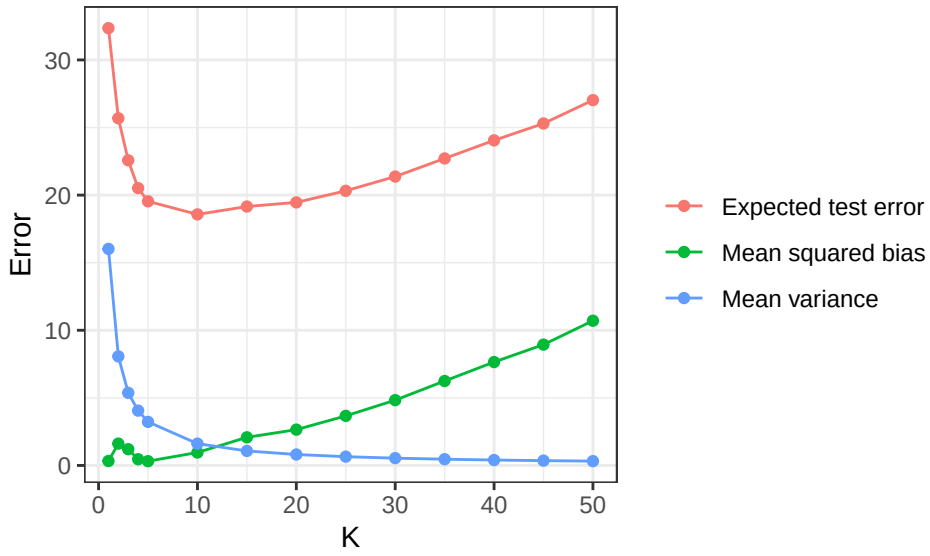

Figure 8: Bias-variance trade-off for KNN regression.

- Based on Figure 8, what is the optimal value of `K`? **Based on Figure 8, the optimal value of K appears to be at around $K = 10$. This is where the expected test error about reaches a minimum (and we can see that the mean squared bias and mean variance (which contribute to the determination of the expected test error) are low).**
- 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? **As we increase K, we are actually effectively decreasing model complexity, and so thus with a simpler model, we have higher bias (as we increase K). With a simpler model, we are simultaneously decreasing variance. We are increasing how many neighbors are considered for predictions, thus averaging over more points for prediction as we increase the value of K, our measure of model complexity, so the bias will increase as more distant neighbors are brought into estimates (and averaging over more points contributes to lower variance). And for smaller values of K, we are averaging over less of the data (and the specific details of the training data including more noise from the data are captured in prediction), highlighting the increased variance for lower values of K.**
- 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.] **In Figure 8, we can see that when we go from $K = 1$ to $K = 2$, we are including one of the trees that is not the focal tree itself. Thus, we are bringing in a more distant tree, and averaging that with the closest tree which is itself, and thus increasing the bias. In this case, we are systematically biased in the direction of the one tree added (whether it be up, down, left or, right). However, when we move from $K = 2$ up to $K = 5$, we are reducing the earlier impact we felt of this distant tree by bringing in more trees that are also distant in the other directions (in the directions up, down, left, and right that are not the focal tree itself), therefore reducing the bias from the point of using $K = 2$ trees for neighbors. When we bring in more than one of these trees that are not the closest tree which is**

itself, we are taking the average and so we are able to essentially cancel out the impacts of different directions from the focal tree, such that we reduce the bias for the prediction. (Once we increase up to $K = 5$, we then have the neighbors as the tree itself and the four next closest trees which are the four neighbors in the four different directions (up, down, left, and right), which are averaged and serve to cancel each other out.)

- 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?

```
# determine which tree and value of $K$ gives overall highest absolute bias
training_results_summary %>% arrange(desc(abs(bias))) %>% head(1) %>%
  kable(format = "latex", row.names = NA, booktabs = TRUE,
        digits = 2,
        col.names = c("K", "X1 location", "X2 location", "Bias", "Variance"),
        caption = "This is the observation, i.e., the particular tree
        and value of K that gives the overall highest absolute bias.") %>%
  kable_styling(position = "center") %>%
  kable_styling(latex_options = "HOLD_position")
```

Table 4: This is the observation, i.e., the particular tree and value of K that gives the overall highest absolute bias.

K	X1 location	X2 location	Bias	Variance
50	200	200	-20.6	0.33

From Table 4 (i.e., highlighting the entry in the dataset corresponding to the highest absolute bias), we have that the tree at location (200, 200) and value of $K = 50$ gives the highest absolute bias, i.e., bias = -20.6 (and thus absolute bias is 20.6). The sign of the bias makes sense, as the negative sign indicates that the actual yield is far greater than the average predicted yield (i.e., the distance from the average fitted model to the true trend is negative). As we saw with the underlying trend depicted in Figure 7, the top right-hand corner of the orchard appeared more fruitful, and that area of the orchard corresponded to points around the furthest point in this corner (200, 200). Thus, it makes sense that the yield here is much higher than the average for the predicted yield such that we have a negative bias. This particular tree and this particular value of `K` gives us the largest absolute bias because of the particularly fruitful yield in actuality that differs largely from the average predicted yield. Also, since this tree is in the far corner, there are not trees above and to the right taken into the average estimate, so we are taking averages of things not in all directions, introducing more bias. (We are taking averages of things below (and left) such that we are more systematically biased, as the point is higher than what our prediction is based off of, as the yields ultimately depend on where you are in the graph.) The value of K chosen (which here we can observe is the largest value in the range of K values explored) decreases model complexity, as we are averaging over 50 nearest trees, thus simplifying the model and reducing our ability to capture more curves and trends in the data, introducing a lot more bias.

- 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.]

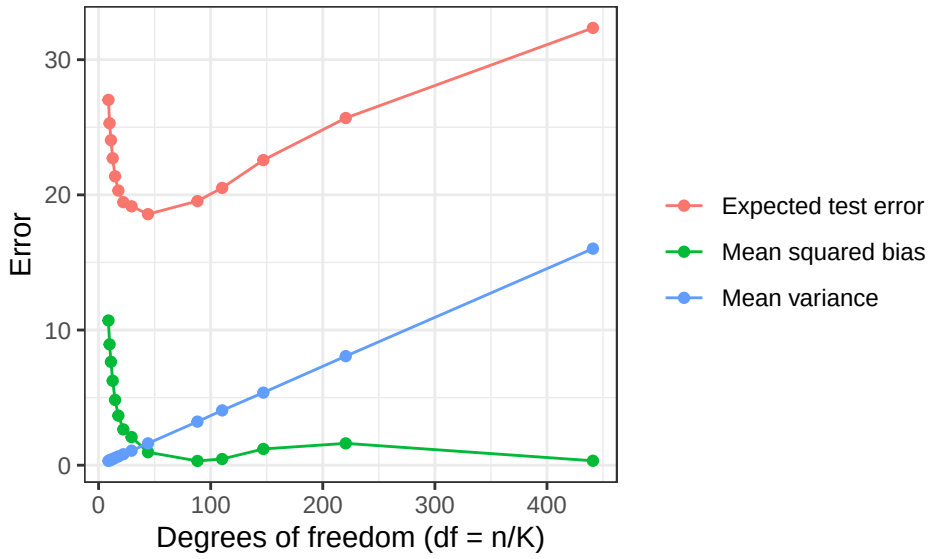

Figure 9: Bias-variance trade-off for KNN regression as a function of df, where $df = n/K$ It is evident that the mean variance curve follows a linear trajectory.

We can observe from Figure 9 that the variance is increasing seemingly linearly as a function of df (where $df = n/K$). We can derive a formula for the KNN Variance. For the prediction of a single tree, we sum the K values of Y (in the training set) (i.e., $Y_1 + Y_2 + ... + Y_K$) and divide by K, as we base a tree's prediction on the average for the K closest trees. Then the Variance for Y-hat is equal to the variance of $\frac{1}{K}\sum_{i=1}^{K} Y_i^{Train}$ (where $Y_i^{Train} = f(x_i) + \epsilon_i$). Considering 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, we obtain $\frac{K\sigma^2}{K^2} = \frac{\sigma^2}{K}$ since the variance for each Y_i is σ^2 and the constant multiple here is $\frac{1}{K}$. Now, we also know that $df = \frac{n}{K}$. So we can multiply df to our result while also multiplying $\frac{K}{n}$ (such that we are ultimately multiplying by 1) to obtain $\frac{\sigma^2}{K} \cdot \frac{K}{n} \cdot df = \frac{\sigma^2}{n} df$. Thus, in summary, we have $\hat{Y} = \frac{1}{K}\sum_{i=1}^{K} Y_{train} = \frac{1}{K}\sum_{i=1}^{K} f(x_i) + \epsilon$. So for the variance, we have $\frac{1}{K^2}\sum_{i=1}^{K} Var(f(x_i) + \epsilon) = \frac{\sigma^2}{K} \rightarrow \frac{\sigma^2}{n} \cdot \frac{n}{K} = \frac{\sigma^2}{n} \cdot df$. Thus, we obtain KNN Variance $= \frac{\sigma^2}{n} \cdot df$. This formula is superimposed onto Figure 10 as a dashed curve.

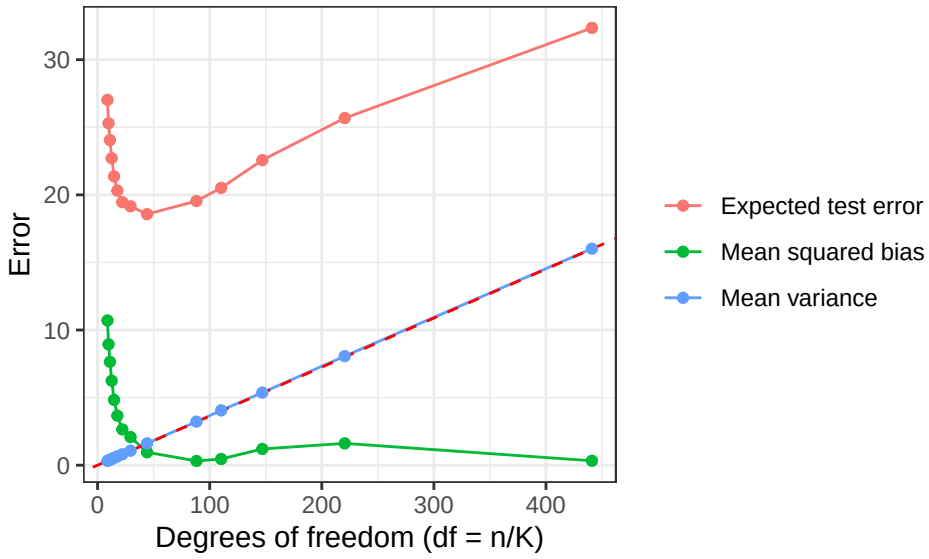

Figure 10: Bias-variance trade-off for KNN regression as a function of df, where $df = n/K$. Superimposed onto the plot as a dashed (red) curve is the derived KNN Variance formula, which matches up with the mean variance curve.

We can observe from Figure 10 that these two variance curves match, as the red dashed line essentially perfectly overlaps with the curve for the mean variance.