Interpolating the number or crimes

Imports

```
library(tidyverse)
library(data.table)
library(lubridate)
library(sf)

devtools::load_all("../chigcrim/")
```

Interpolating number of crimes using kernel ridge regression

Estimating the number of crimes is a regression task. Here we will consider estimating the the number of crimes in a given week using kernel ridge regression.The kernel ridge regression implementation includes a linear kernel (equivalent to ridge regression), a polynomial kernel, and a radial basis function kernel. As the relationship complex and non-linear, only the radial basis function feature transform will be considered here, which implicitly induces an infinite dimensional feature transform. First we will consider a simple, single dimensional example, using just the number of crimes each week as the feature.

Predicting the number of crimes each week

Read and prepare data

Due to speed limitations, it is convenient to consider summarising the data into crimes per week.

```
df <- fread("../data/crimes-2001-present.csv", select = c("Date"))

df <- df %>%
  mutate(date = as.Date(mdy_hms(Date))) %>%
  select(-Date) %>%
  mutate(week = week(date),
         year = year(date)) %>%
  filter(week != 53)  # 53rd "week" does not contain 7 days

df <- df %>%
  group_by(week, year) %>%
  mutate(week_start = min(date)) %>%
  group_by(week, week_start, year) %>%
  tally(name = "n_crimes")

df <- df[order(df$year, df$week),]
df$cumulative_week <- 1:nrow(df)
```

Cross-validation

Below I will use 5-fold cross validation to choose the bandwidth hyperparameter for the radial basis function kernel.

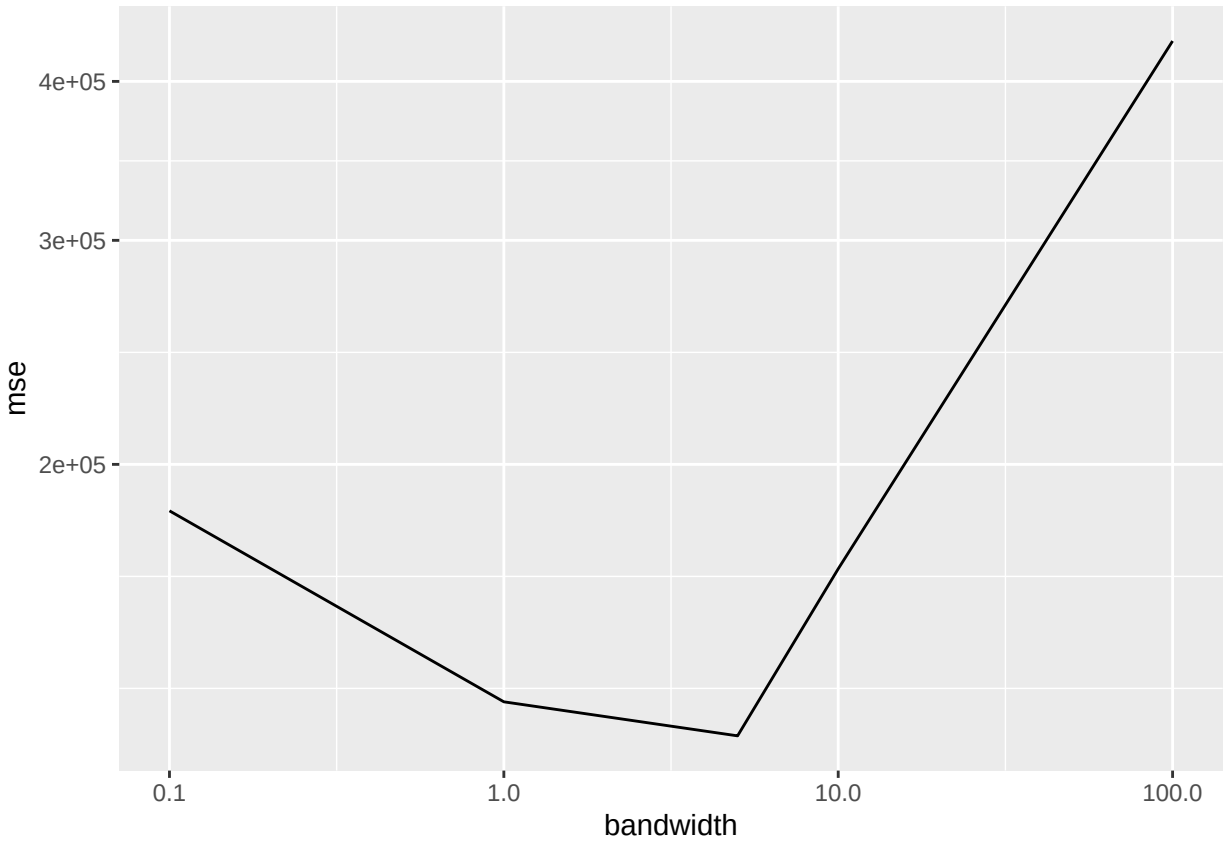
```r
X <- as.matrix(df$cumulative_week)
X <- scale(X)
y <- df$n_crimes

bandwidth <- c(0.1, 1, 5, 10, 100)

mse_vec <- c()
set.seed(2)
for (i in 1:length(bandwidth)){
  kr = KernelRidge$new("rbf", lambda = 0.001, bandwidth[i])
  mse <- cv_R6_k_fold(kr, X, y, squared_error_loss, 5)
  mse_vec <- c(mse_vec, mse)

  # Save best model
  if (mse == min(mse_vec)){
    kr_best <- kr$clone(deep=TRUE)
  }
}

rbf_results <- data.frame(bandwidth, mse = mse_vec)

rmse <- sqrt(rbf_results$mse[rbf_results$bandwidth == kr_best$A])
sprintf("Optimal bandwidth found: %s", kr_best$A)
```

```
## [1] "Optimal bandwidth found: 5"
```

```r
sprintf("Which had root mean square error: %s", rmse)
```

```
## [1] "Which had root mean square error: 349.804006309564"
```

```r
ggplot(rbf_results) +
  geom_line(aes(x = bandwidth, y = mse)) +
  scale_x_log10() +
  scale_y_log10()
```

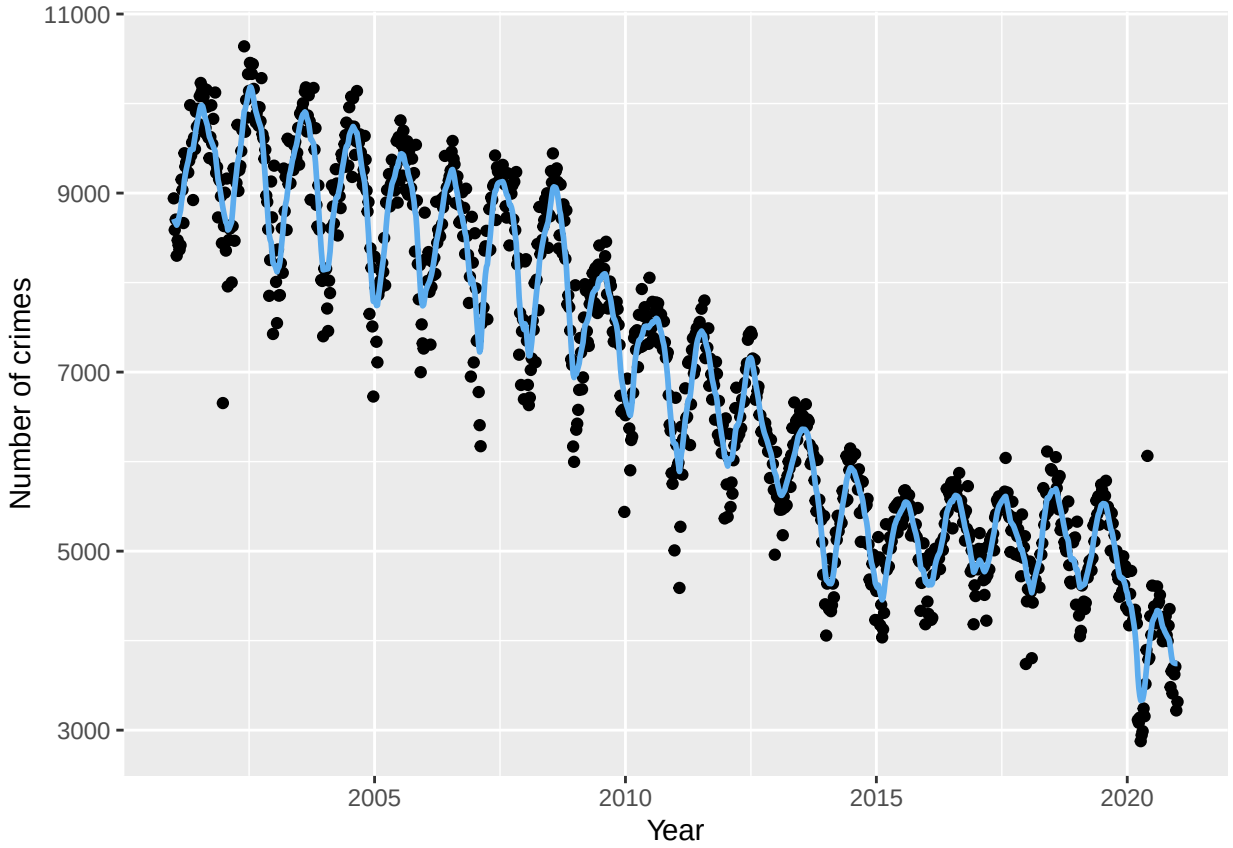


A bandwidth of 5 gave the lowest cross-validation error.

Plotting the prediction function

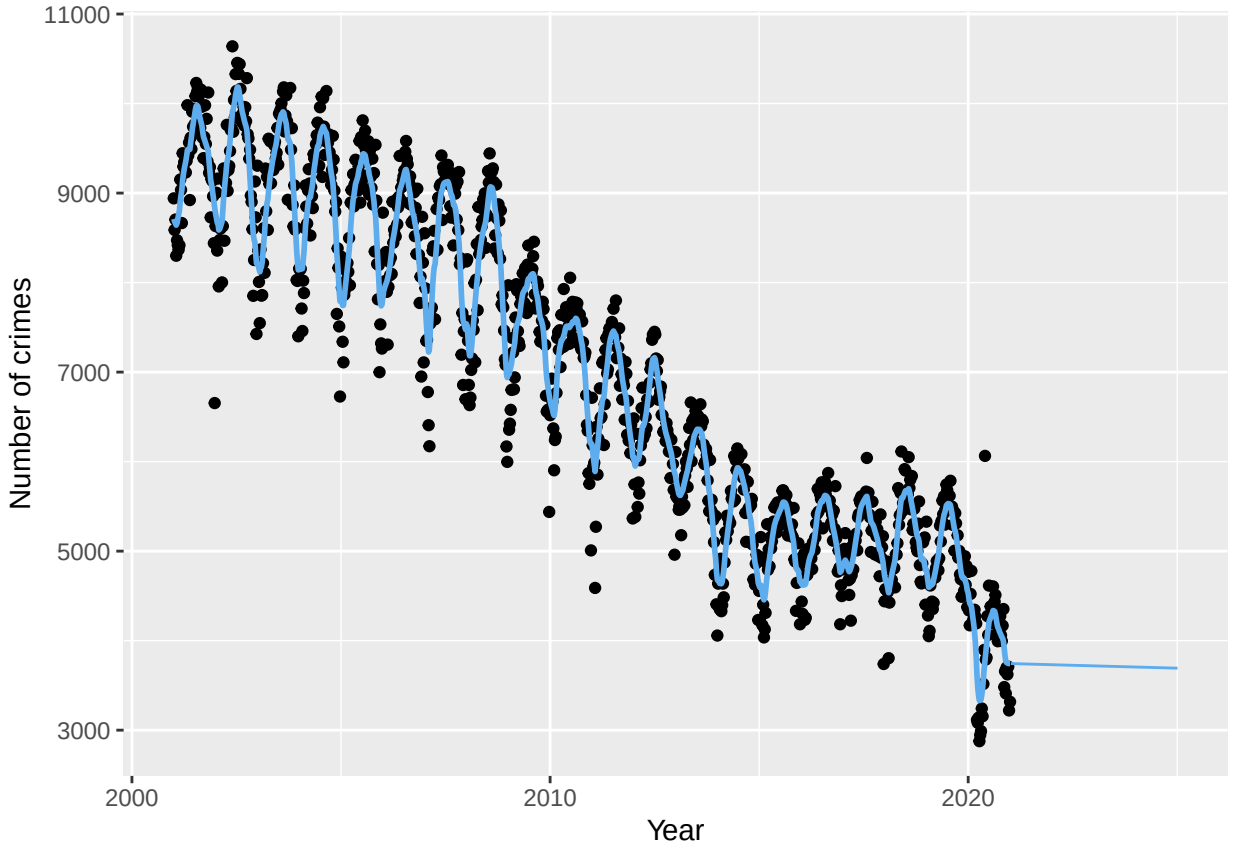
As this is single dimensional we can plot the prediction function for the best model:

```
y_hat = kr_best$predict(X)

# Plot output
df$y_hat <- y_hat

p <- ggplot(df) +
  geom_point(aes(x = week_start, y = n_crimes)) +
  geom_line(aes(x = week_start, y = y_hat), colour = "steelblue2", size = 1) +
  labs(x = "Year", y = "Number of crimes")
p
```



Another option that has not been considered here is using a trigonometric transform of the data (with a period of a year), to try and capture the cycles with a simpler model. It is worth noting that whilst useful for interpolation, kernel ridge regression with a radial basis function kernel would have limited utility for forecasting crimes for into the future. We can see why this is if we try to extrapolate using these this model.

```
start_week_date <- max(df$week_start) + 7
start_week_int <- max(df$cumulative_week) + 1

extra_dates <- seq(ymd(start_week_date),ymd(start_week_date + 365*4),
                   by = '1 week')

extra_X <- start_week_int:(start_week_int+length(extra_dates)-1)
extra_X <- (extra_X - attributes(X)$`scaled:center`) /
  attributes(X)$`scaled:scale`

extra_y_hat <- kr_best$predict(as.matrix(extra_X))

extra_df <- tibble(extra_dates, extra_y_hat)
p +
  geom_line(data = extra_df, aes(extra_dates, extra_y_hat), colour = "steelblue2")
```

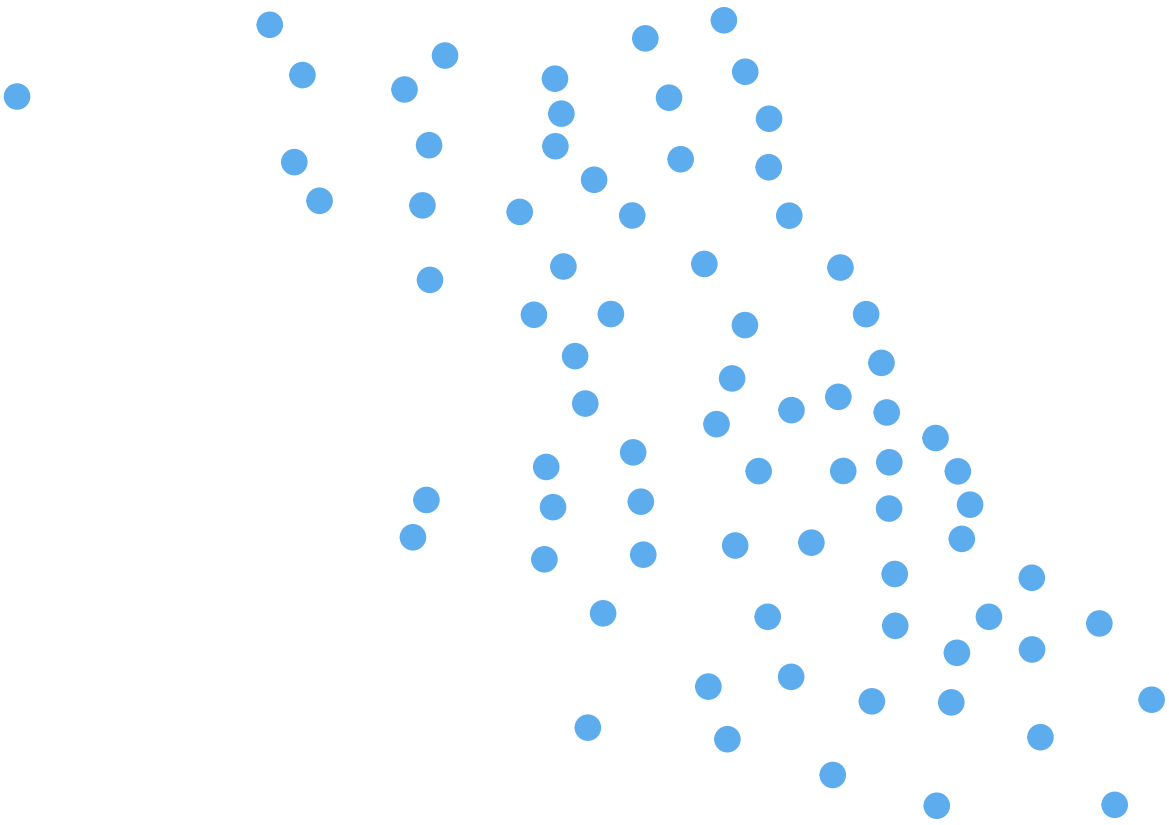


The cycles are captured solely due to being able to interpolate locally between points. Hence, when extrapolating with a radial basis function kernel, no cycles are predicted, so the out-of-sample accuracy would drop substantially. The cross-validation error found is only applicable if the new data followed the same distribution, and that is not the case when considering future data. For this reason, other models, such as generalised additive models, will be considered for forecasting crimes in subsequent sections.

Predict number of crimes in a particular month for a particular community area

The previous example used a single dimensional feature vector. We can also apply it to more complicated problems. Here we will consider the problem of predicting the number of crimes in a given month, for different community areas. The positions of the community areas will be represented by their centroids, as shown below:

```
centroids <- st_read("../data/community_areas.shp", quiet = TRUE) %>%
  arrange(as.integer(area_numbe)) %>%
  st_centroid() %>%
  st_coordinates() %>%
  as_tibble() %>%
  mutate(community_area = 1:nrow(.))

ggplot(centroids, aes(X, Y)) +
  geom_point(size = 4, colour = "steelblue2") +
  theme_void()
```



Read and prepare data

```
# Can use below but slow so better to download data set from website
# df <- load_data(strings_as_factors = FALSE)

df <- fread("../data/crimes-2001-present.csv",
            select = c("Date", "Community Area"))

colnames(df) <- str_replace(tolower(colnames(df)), " ", "_")
```

```
df <- df %>%
  mutate(date = as.Date(mdy_hms(date)),
         year = year(date),
         month = month(date)) %>%
  filter(date >= "2015-01-01", year != 2021) %>%  # Not enough data on 2021 yet
  drop_na()

df <- df %>%
  group_by(year, month) %>%
  mutate(date = min(date)) %>%
  group_by_all() %>%
  summarise(n_crimes = n()) %>%
  left_join(centroids, by = "community_area")
```

Cross-validation

As before, 5-fold cross validation will be used to choose the bandwidth parameter:

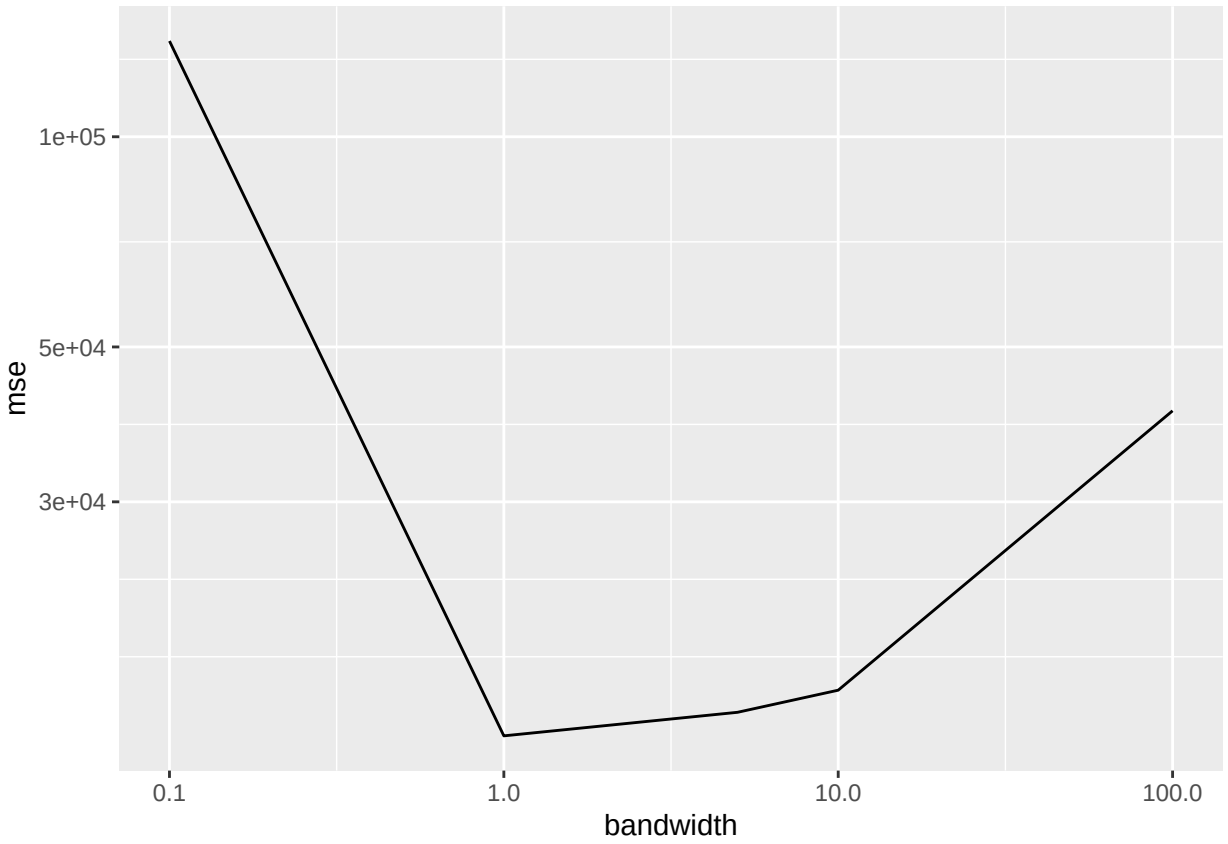
```r
X <- df %>%
  ungroup() %>%
  select(-community_area, -date, -n_crimes) %>%
  as.matrix()

X <- scale(X)
y <- df$n_crimes

bandwidth <- c(0.1, 1, 5, 10, 100)

mse_vec <- c()
set.seed(3)
for (i in 1:length(bandwidth)){
  kr = KernelRidge$new("rbf", lambda = 0.001, bandwidth[i])
  mse <- cv_R6_k_fold(kr, X, y, squared_error_loss, 5)
  mse_vec <- c(mse_vec, mse)

  # Save best model
  if (mse == min(mse_vec)){
    kr_best <- kr$clone(deep=TRUE)
  }
}

rbf_results <- data.frame(bandwidth, mse = mse_vec)

rmse <- sqrt(rbf_results$mse[rbf_results$bandwidth == kr_best$A])
sprintf("Optimal bandwidth found: %s", kr_best$A)
```

```
## [1] "Optimal bandwidth found: 1"
```

```r
sprintf("Which had root mean square error: %s", rmse)
```

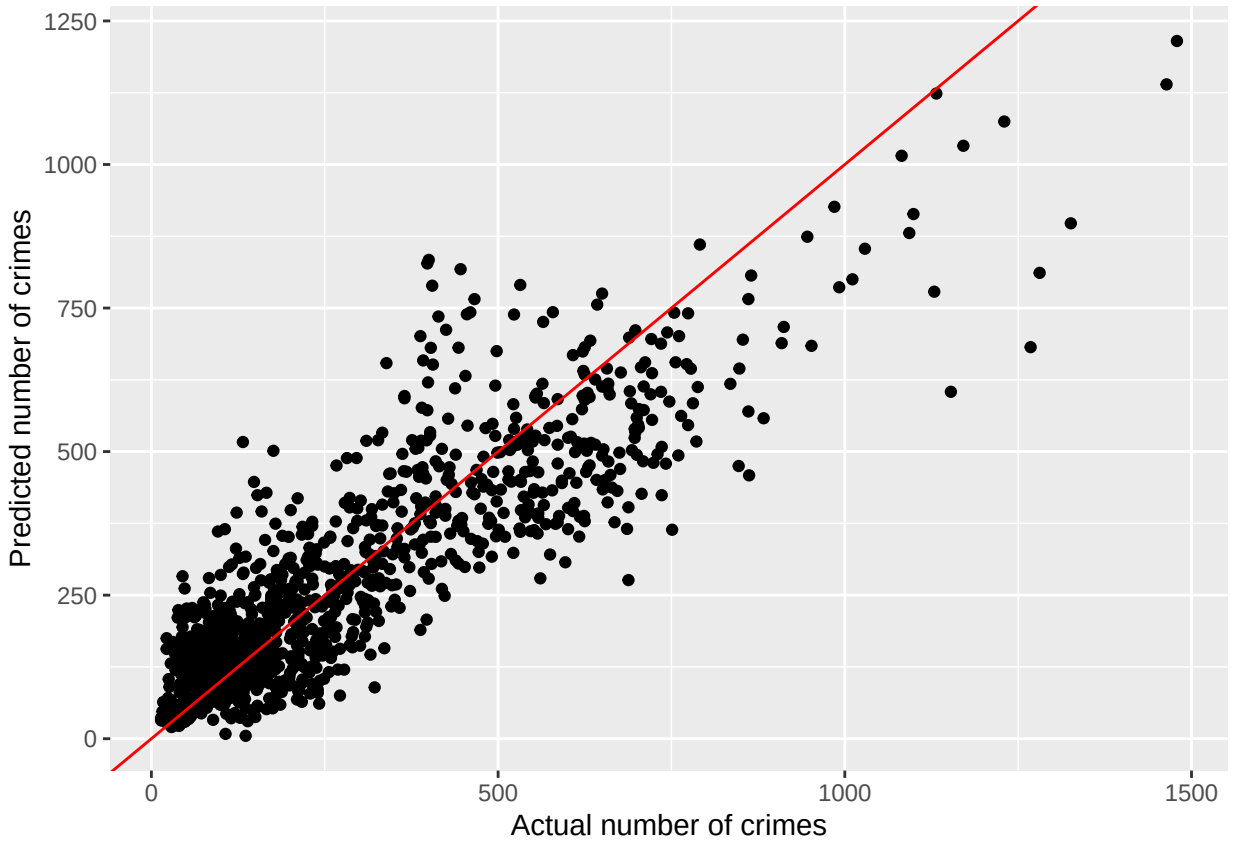
```
## [1] "Which had root mean square error: 117.757445924084"
```

```r
ggplot(rbf_results) +
  geom_line(aes(x = bandwidth, y = mse)) +
  scale_x_log10() +
  scale_y_log10()
```



Although being higher dimensional so difficult to plot, we can plot the predicted values against the observed values on a held out data set:

```
test_idxs <- sample(1:nrow(X), round(0.2*nrow(X)))

X_train <- X[-test_idxs, ]
y_train <- y[-test_idxs]
X_test <- X[test_idxs, ]
y_test <- y[test_idxs]

# Refit best model
kr_best$fit(X_train, y_train)
y_hat <- kr_best$predict(X_test)

qplot(y_test, y_hat, geom = "point") +
  geom_abline(slope = 1, intercept = 0, colour = "red") +
  labs(x = "Actual number of crimes", y = "Predicted number of crimes")
```



We can see a good association between the actual number of crimes in a given month and community area, and the predicted number of crimes on this held-out data. Once again, this method would primarily be useful for interpolation, rather than extrapolation, for similar reasons as have been mentioned above.