Interpolating the number or crimes

Imports

```
library(tidyverse)
library(data.table)
library(lubridate)
library(sf)
# library(rgdal)
# library(ggmap)

devtools::load_all("../chigcrim/")
set.seed(1)
```

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 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 is complex and non-linear, only the radial basis function feature transform will be considered here, which implicitly induces an extremely flexible infinite dimensional feature transform.

Assessing performance

In these regression tasks, two metrics will be used to assess performance, root mean square error, and the R^2 value.

```
metrics <- list(rmse = rmse_loss, r2 = r_squared)
```

Repeated k-fold cross validation is used, and hyperparameters are chosen based on the root mean square error metric. Specifically, we will use 5 repetitions of 5-fold cross validation at each parameter value (ran in parallel).

Predicting the number of crimes each week

First we will consider a simple, single dimensional example, using just the number of crimes each week as the feature.

Read and prepare data

```
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
```

```r
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)

df
```

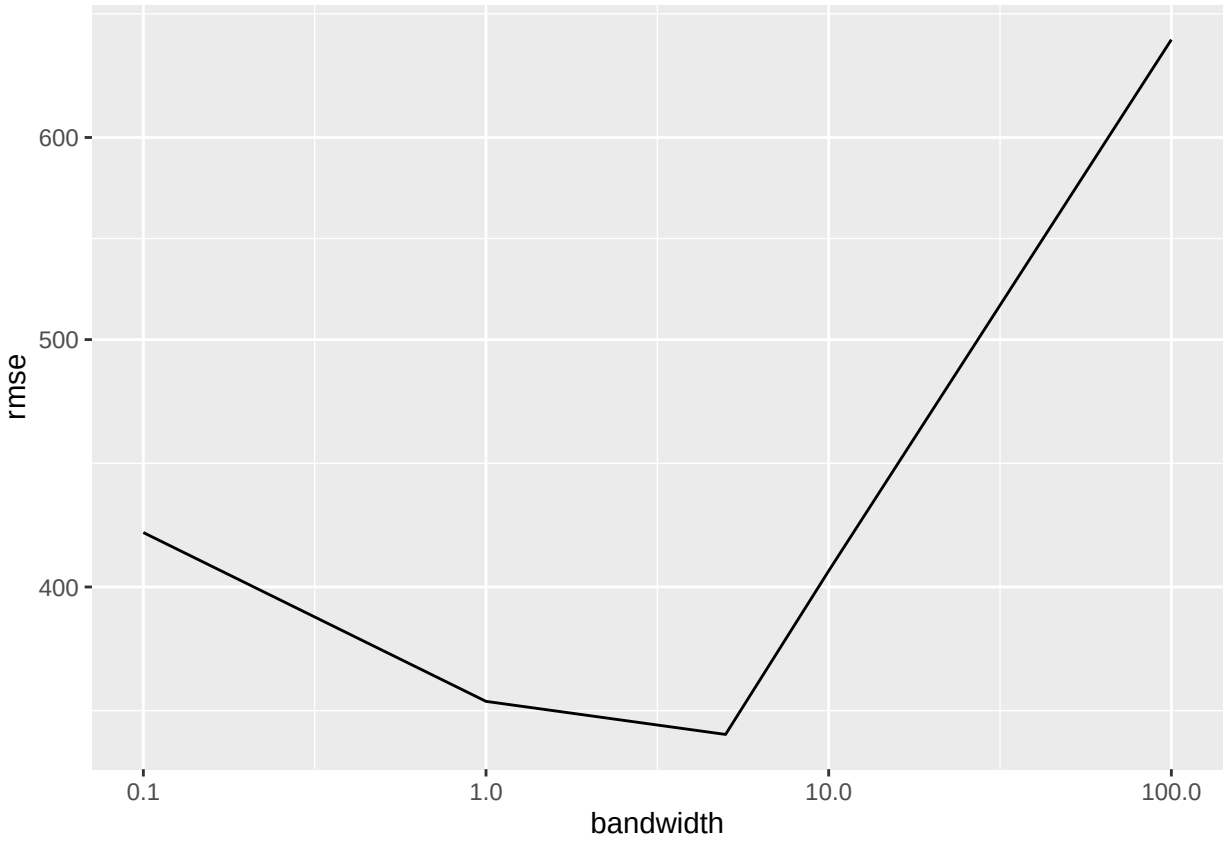
```
## # A tibble: 1,041 x 5
## # Groups:   week, week_start [1,041]
##     week week_start  year n_crimes cumulative_week
##    <dbl> <date>     <dbl>    <int>           <int>
##  1     1 2001-01-01  2001     8941               1
##  2     2 2001-01-08  2001     8588               2
##  3     3 2001-01-15  2001     8704               3
##  4     4 2001-01-22  2001     8300               4
##  5     5 2001-01-29  2001     8473               5
##  6     6 2001-02-05  2001     8418               6
##  7     7 2001-02-12  2001     8368               7
##  8     8 2001-02-19  2001     8417               8
##  9     9 2001-02-26  2001     9151               9
## 10    10 2001-03-05  2001     9035              10
## # ... with 1,031 more rows
```

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)

rmse_vec <- c()
for (i in 1:length(bandwidth)){
  kr = KernelRidge$new("rbf", lambda = 0.001, bandwidth[i])
  cv_results <- kfold_cv(kr, X, y, metrics, 5, n_reps = 5,
                         parallel = TRUE, n_threads = 5)

  mean_rmse <- mean(cv_results$rmse)
  rmse_vec <- c(rmse_vec, mean_rmse)

  # Save best model
  if (mean_rmse == min(rmse_vec)){
    kr_best <- kr$clone(deep=TRUE)
    cv_results_best <- cv_results
  }
}

rbf_results <- data.frame(bandwidth, rmse = rmse_vec)
```

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



We can see that a bandwidth parameter value of 5 gave the lowest root mean square error. We can look at the results for the three repeats of 5-fold cross-validation for this model:

```
as_tibble(cv_results_best)
```

```
## # A tibble: 5 x 2
##    rmse    r2
##   <dbl> <dbl>
## 1  351. 0.964
## 2  351. 0.964
## 3  353. 0.964
## 4  346. 0.964
## 5  349. 0.964
```

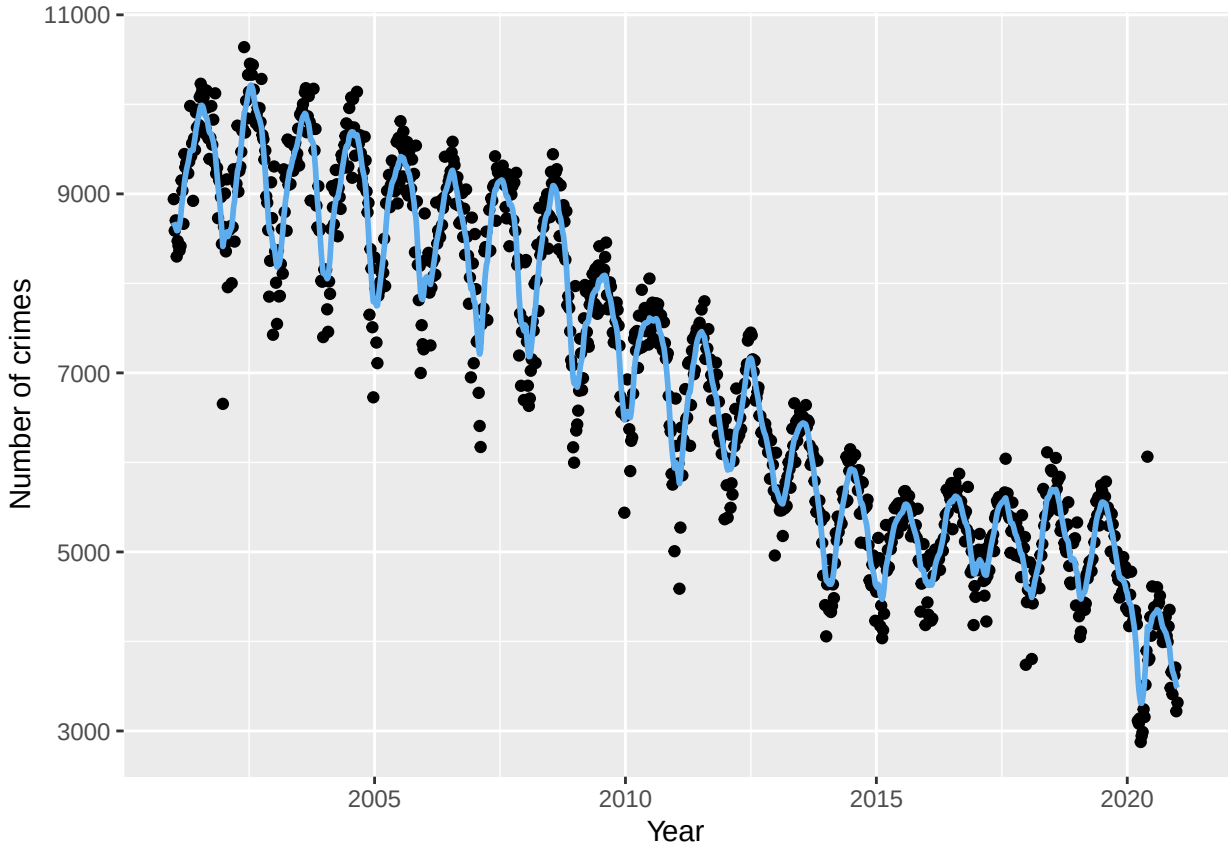
The model performs very well, explaining 96.37% of the variance on average.

Plot results

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

```
y_hat = kr_best$fit(X, y)
y_hat = kr_best$predict(X)

# Plot output
```

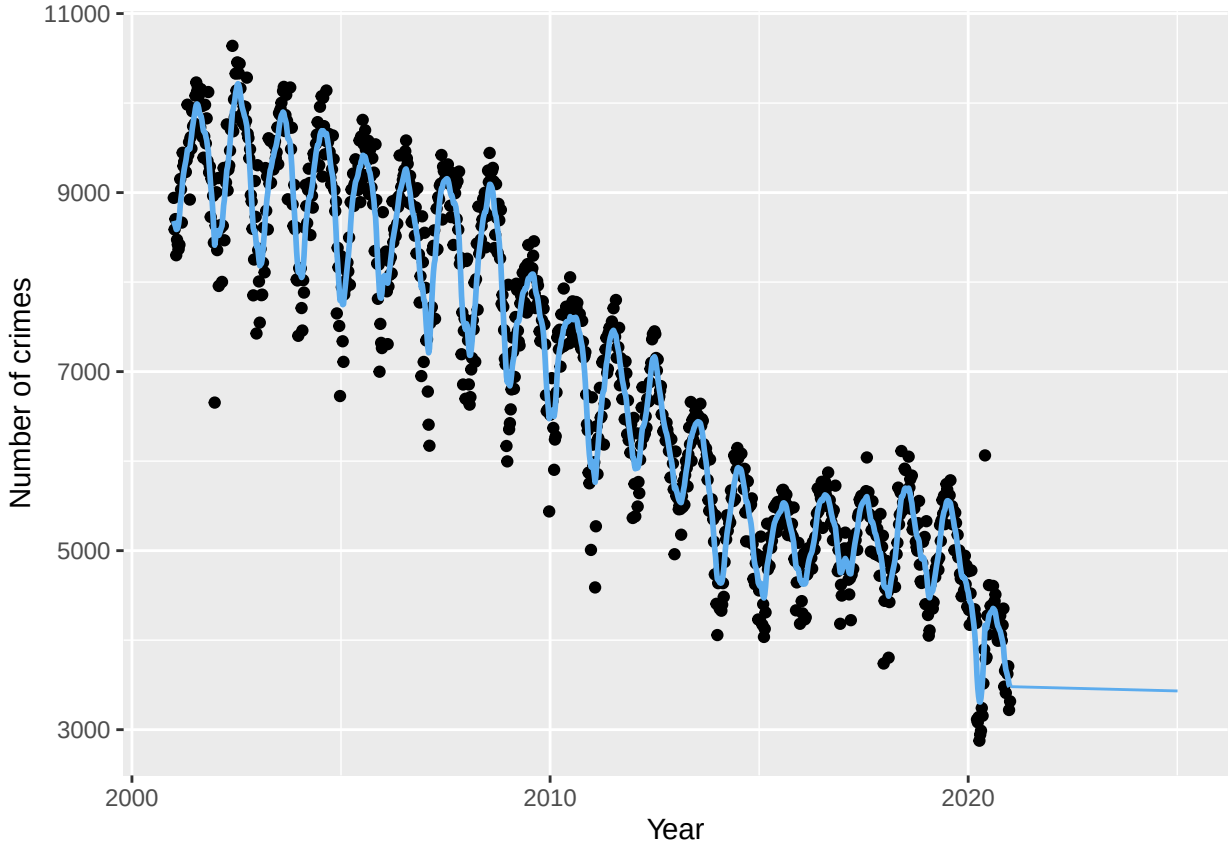
```
df$y_hat <- y_hat

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.

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

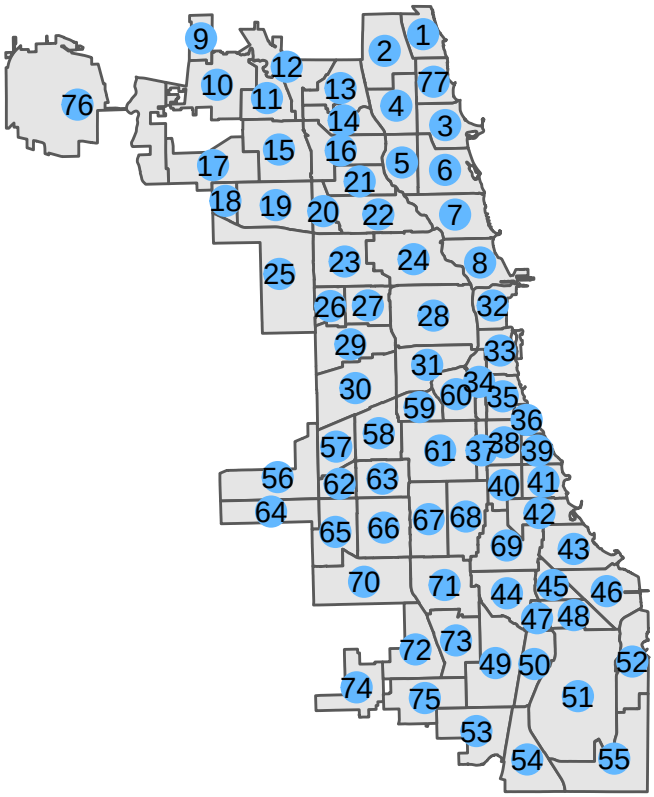
The previous example used a single dimensional feature vector. We can also apply it to more complicated problems. Here we will consider the problem of predicting the number of crimes in a given month, for different community areas.

Read and prepare data

The positions of the community areas will be represented by their centroids, as shown below:

```
centroids <- community_bounds %>%
  arrange(as.integer(area_numbe)) %>%
  st_centroid() %>%
  st_coordinates() %>%
  as_tibble() %>%
  mutate(community_area = 1:nrow(.))

p2 <- ggplot() +
  geom_sf(data = community_bounds) +
  geom_point(data = centroids, aes(X, Y), size = 5, colour = "steelblue1") +
  geom_text(data = centroids, aes(X, Y, label = community_area)) +
```

```
  theme_void()
p2
```



We will consider the time period from 2016 to 2020 (inclusive):

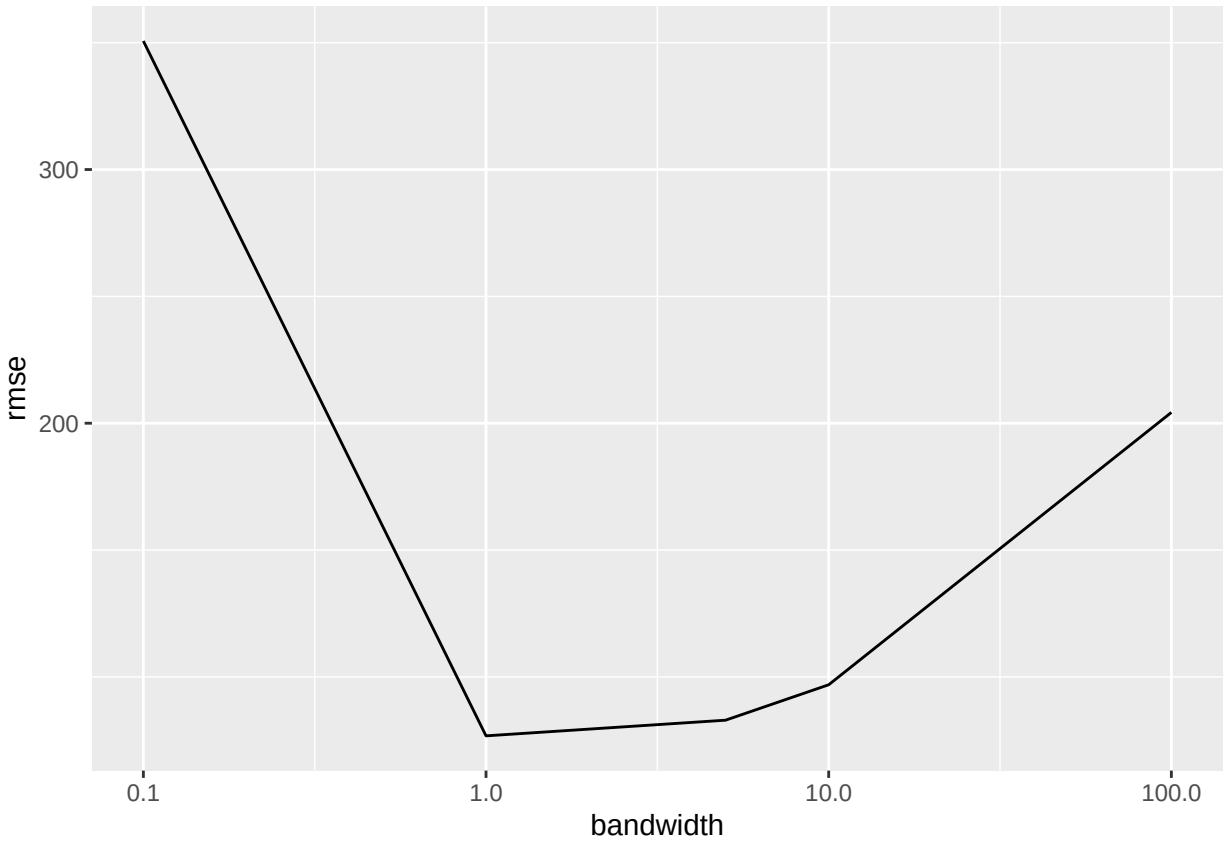
```
# Can use below but slow so better to download data set from website
# df <- load_data(strings_as_factors = FALSE)

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 >= "2016-01-01", 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")

df
```

```
## # A tibble: 4,620 x 7
## # Groups:   date, community_area, year [4,620]
##    date       community_area  year month n_crimes     X     Y
##    <date>              <int> <dbl> <dbl>    <int> <dbl> <dbl>
##  1 2016-01-01              1  2016     1      309 -87.7  42.0
##  2 2016-01-01              2  2016     1      253 -87.7  42.0
##  3 2016-01-01              3  2016     1      280 -87.7  42.0
##  4 2016-01-01              4  2016     1      140 -87.7  42.0
##  5 2016-01-01              5  2016     1      108 -87.7  41.9
##  6 2016-01-01              6  2016     1      477 -87.7  41.9
##  7 2016-01-01              7  2016     1      306 -87.6  41.9
##  8 2016-01-01              8  2016     1      801 -87.6  41.9
##  9 2016-01-01              9  2016     1       19 -87.8  42.0
## 10 2016-01-01             10  2016     1      103 -87.8  42.0
## # ... with 4,610 more rows
```

Cross-validation

As before, 3 times repeated 5-fold cross validation will be used to choose the bandwidth parameter:

```
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)

rmse_vec <- c()
set.seed(3)
for (i in 1:length(bandwidth)){
  kr = KernelRidge$new("rbf", lambda = 0.001, bandwidth[i])
  cv_results <- kfold_cv(kr, X, y, metrics, 5, n_reps = 5,
                         parallel = TRUE, n_threads = 5)
  mean_rmse <- mean(cv_results$rmse)
  rmse_vec <- c(rmse_vec, mean_rmse)

  # Save best model
  if (mean_rmse == min(rmse_vec)){
    kr_best <- kr$clone(deep=TRUE)
    cv_results_best <- cv_results
  }
}

rbf_results <- data.frame(bandwidth, rmse = rmse_vec)

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



We can see that a bandwidth parameter value of 1 gave the lowest root mean square error. Again, we can look at the results for the three repeats of 5-fold cross-validation for this model:

```
as_tibble(cv_results_best)
```

```
## # A tibble: 5 x 2
##    rmse    r2
##   <dbl> <dbl>
## 1  119. 0.765
## 2  121. 0.759
## 3  123. 0.750
## 4  121. 0.758
## 5  123. 0.750
```

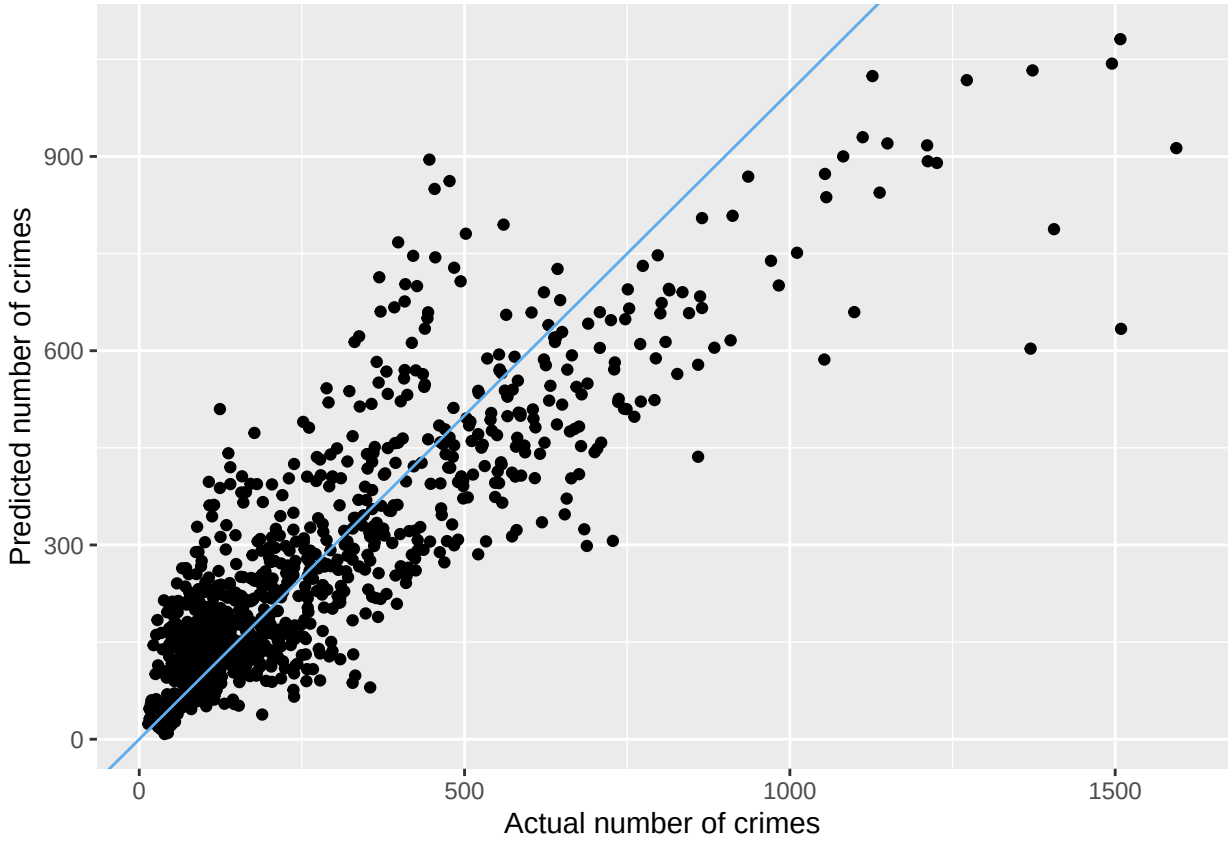
The model explained 75.63% of the variance on average. The lower R^2 value is unsurprising for this model, as in contrast to the first model, observations were not averaged over the whole of Chicago.

Plot results

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]
```

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

p3 <- qplot(y_test, y_hat, geom = "point") +
  geom_abline(slope = 1, intercept = 0, colour = "steelblue2") +
  labs(x = "Actual number of crimes", y = "Predicted number of crimes")
p3
```



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.

Conclusion

We can see that kernel ridge regression provides prediction functions that explain a high proportion of the variance. A major limitation of this method for predicting the number of crimes is that it would be primarily be useful for interpolation, rather than extrapolation, for the reasons outlined above. In the subsequent section, we explore generalised additive models, which form a more principled approach for forecasting the number of crimes.