Assignment 1

2023-09-29

Introduction

In this assignment, we will examine 3 different datasets, split into small, large and test, and construct polynomial regression models accordingly. The models will have this form:

$$y = f(x) = \beta_0 + \sum_{k=1}^{K} \beta_k x^k$$

In the report below, we will visualize the data, evaluate the MSE values, as well as make conclusions on what polynomial regression models best generalize the unseen data.

Furthermore, we will explore underfitting, overfitting and model generalization, and how to improve these issues.

Import required libraries.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.1     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)
library(readxl)
library(tinytex)
```

Import the datasets

We import 3 different datasets, with 9 sheets in total.

```
filename = "/Users/nguyennhatle/Desktop/UTM/STA/STA314/ASSIGNMENTS/A1/Dataset_1.xlsx"
small_train1 = read_excel(filename, sheet=1)
large_train1 = read_excel(filename, sheet=2)
test_data1 = read_excel(filename, sheet=3)
small_train2 = read_excel(filename, sheet=4)
```

```
large_train2 = read_excel(filename, sheet=5)
test_data2 = read_excel(filename, sheet=6)
small_train3 = read_excel(filename, sheet=7)
large_train3 = read_excel(filename, sheet=8)
test_data3 = read_excel(filename, sheet=9)
# Store all datasets in 1 list for future use
dataset_list = list()
for (i in 1:9) {
  dataset_list[[i]] = read_excel(filename, sheet=i)
}
dataset_names = list()
dataset_names[[1]] = 'small_train1'
dataset_names[[2]] = 'large_train1'
dataset_names[[3]] = 'test_data1'
dataset_names[[4]] = 'small_train2'
dataset_names[[5]] = 'large_train2'
dataset_names[[6]] = 'test_data2'
dataset_names[[7]] = 'small_train3'
dataset_names[[8]] = 'large_train3'
dataset_names[[9]] = 'test_data3'
```

Build the models

The model of each dataset is fitted using polynomial regression, and its MSE values that evaluate model predictions with unseen data (test dataset) are displayed and visualized. Moreover, the parameter vectors

$$\beta_K = [\beta_0, \beta_1, .., \beta_K]^T$$

are estimated.

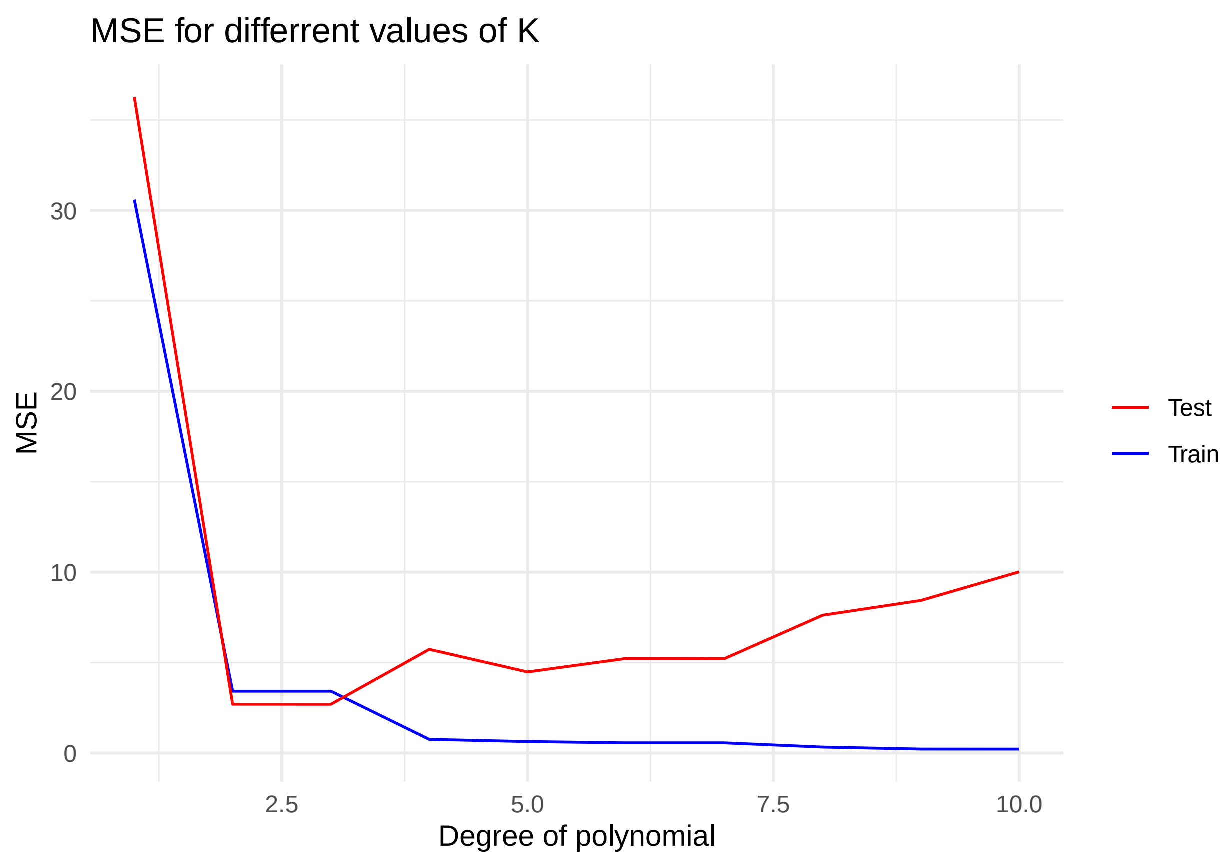
```
# This is a loop that generates the MSE graphs and tables, as well as the parameter vector of each K, f
for (i in 1:9) {
  dataset <- dataset_list[[i]]
  mse_train <- c()
  mse_test <- c()
  # Vector that stores the coefficients/parameters of the model
  models_coefs <- c()
  if (i %% 3 != 0) {
    for (k in 1:10) {
      model <- lm(y ~ poly(x, k, raw=TRUE), data=dataset)
      train_fit <- model %>% predict(dataset)
      train_mse <- mean((dataset$y - train_fit)^2)
      mse_train[k] <- train_mse
      test_data <- dataset_list[[(floor(i/3) + 1)*3]]
      test_fit <- model %>% predict(test_data)
      test_mse <- mean((test_data$y - test_fit)^2)
      mse_test[k] <- test_mse
      models_coefs[[k]] <- as.vector(coef(model))
    }
    mse <- data.frame(
        x = 1:length(mse_train),
        y1 = mse_train,
        y2 = mse_test
```

```r
      )
    print(dataset_names[[i]])
    print("PARAMETERS VECTORS FOR EACH VALUE OF K")
    print(models_coefs)
    print("Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset")
    print("MSE")
    print(mse)
    # Plot the graph
    print(ggplot(mse, aes(x=x)) +
      geom_line(aes(y = y1, color="Train"), linetype = "solid") +
      geom_line(aes(y = y2, color="Test"), linetype = "solid") +
      labs(
        title = "MSE for differrent values of K",
        x = "Degree of polynomial",
        y = "MSE"
      ) + scale_color_manual(values = c("Train" = "blue", "Test" = "red")) +
      theme_minimal() + theme(legend.title = element_blank()))


  }
}
```

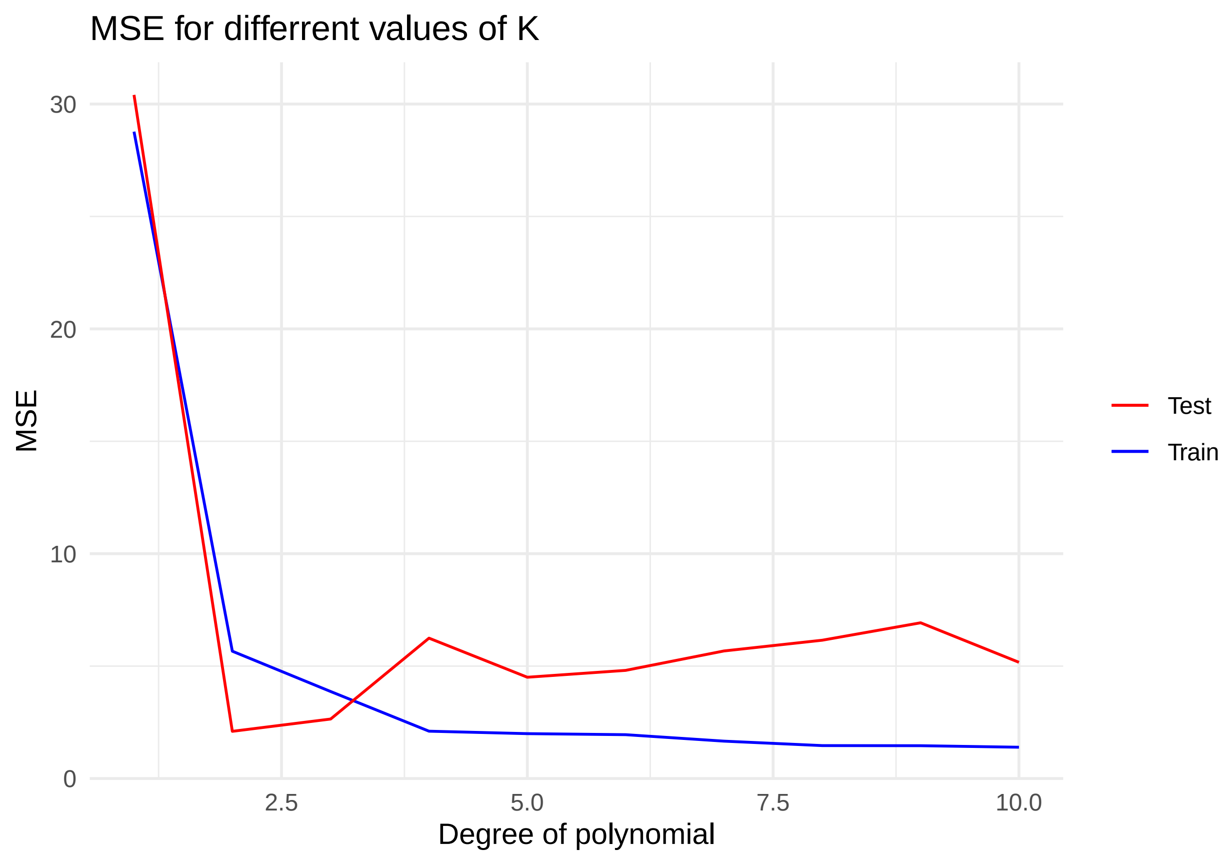
```
## [1] "small_train1"
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
## [[1]]
## [1] 8.498693 1.487141
##
## [[2]]
## [1] 3.536406 1.635826 3.952697
##
## [[3]]
## [1]  3.535467192  1.656643935  3.953634239 -0.007861508
##
## [[4]]
## [1]  4.6136689  3.2749222 -0.7431671 -0.7397551  1.4544078
##
## [[5]]
## [1]  4.5857050  2.0098484 -0.8197144  0.6461968  1.5135348 -0.3052741
##
## [[6]]
## [1]  4.3508736  1.6656219  1.4744648  0.9050007 -0.1978830 -0.3698159  0.3075375
##
## [[7]]
## [1]  4.34178976  1.72127109  1.60267716  0.69359628 -0.29375742 -0.23756962
## [7]  0.32559885 -0.02193175
##
## [[8]]
## [1]   5.32643543   1.05742956 -11.93475835   1.95899221  17.38554328
## [6]  -0.60176814  -7.08281899  -0.02807419   0.97752090
##
## [[9]]
##  [1]   6.166544  -1.936728 -22.653643  25.860749  29.271144 -27.822327
##  [7] -11.804461  10.540615   1.622073  -1.331551
```

```
##
## [[10]]
##  [1]   6.1925532  -2.5110963 -22.1125348  31.8667662  23.0413675 -34.6772747
##  [7]  -4.6572169  13.2436777  -1.1431818  -1.6850332   0.3483338
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x        y1        y2
## 1   1 30.5970661 36.263945
## 2   2  3.4179651  2.697821
## 3   3  3.4178977  2.694149
## 4   4  0.7532015  5.730822
## 5   5  0.6325159  4.480249
## 6   6  0.5597886  5.225318
## 7   7  0.5594646  5.212097
## 8   8  0.3269160  7.615242
## 9   9  0.2154632  8.433212
## 10 10  0.2139145 10.016358
```



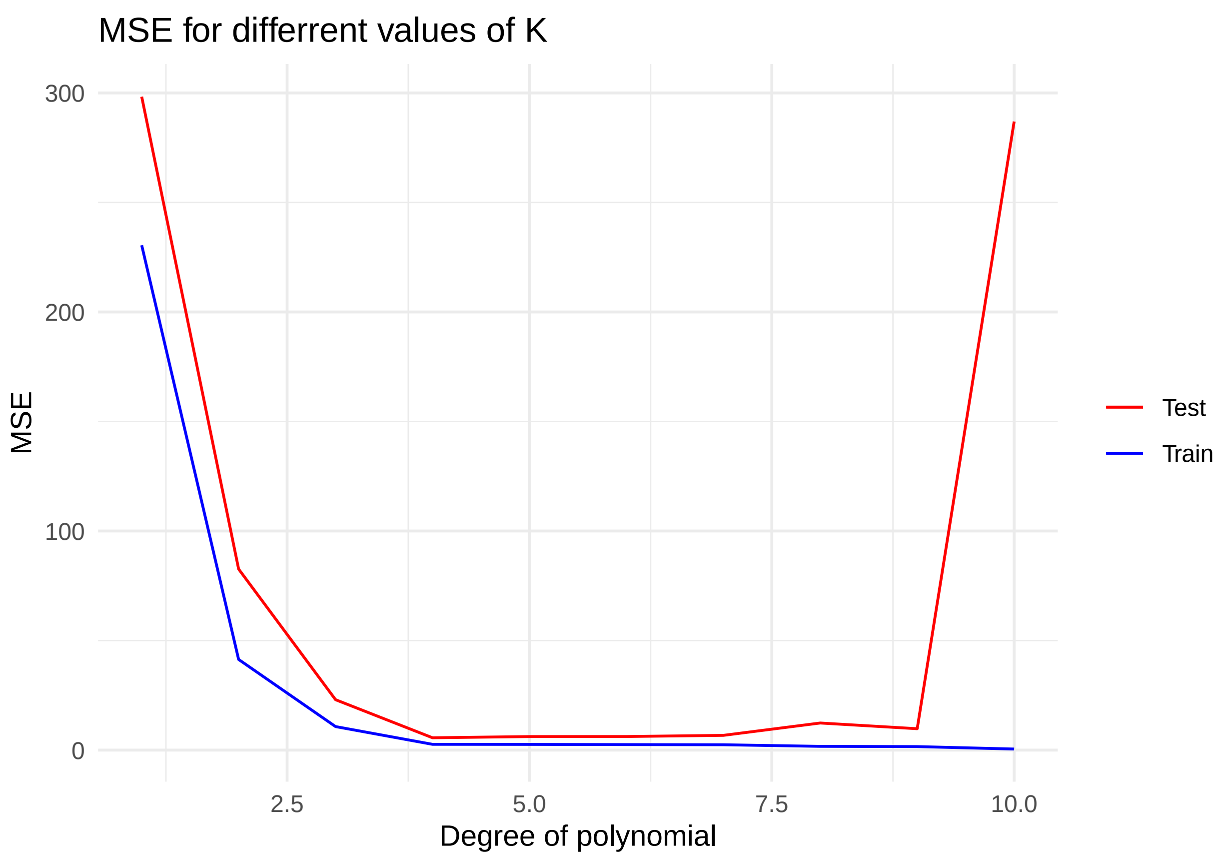

```
## [1] "large_train1"
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
## [[1]]
## [1] 7.080136 2.493103
##
## [[2]]
## [1] 3.004289 1.963497 3.948467
```

```
##
## [[3]]
## [1]  2.815056  4.851338  4.342502 -1.179337
##
## [[4]]
## [1]  3.98263928  4.62127210  0.08785232 -1.36341712  1.25835292
##
## [[5]]
## [1]  4.01844415  3.76035875  0.07040849 -0.16179133  1.29600924 -0.28286659
##
## [[6]]
## [1]  3.7805036  3.5498333  1.6308555  0.2141229  0.1139721 -0.3880057  0.2178299
##
## [[7]]
## [1]  3.9007992  5.5961288  1.0884126 -5.9906332 -0.0903635  3.2662043  0.3413909
## [8] -0.5832798
##
## [[8]]
## [1]  4.6947039  4.9737081 -7.4393261 -2.8740167 13.9561224  1.8214827 -6.1951845
## [8] -0.4328830  0.9054629
##
## [[9]]
##  [1]  4.6828047  5.2197060 -7.3509739 -4.4811927 14.0287118  3.9847674
##  [7] -6.2264894 -1.3597730  0.9040720  0.1230315
##
## [[10]]
##  [1]   5.2310248   4.8206989 -14.2112099  -2.5713021  30.0738110   2.5518955
##  [7] -19.2987359  -1.0681459   5.1993202   0.1200515  -0.4872513
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x        y1        y2
## 1   1 28.771679 30.407982
## 2   2  5.663763  2.099428
## 3   3  3.870950  2.647117
## 4   4  2.105443  6.244531
## 5   5  1.996164  4.505936
## 6   6  1.950295  4.810027
## 7   7  1.662841  5.674785
## 8   8  1.465245  6.152186
## 9   9  1.460570  6.927631
## 10 10  1.391774  5.169417
```



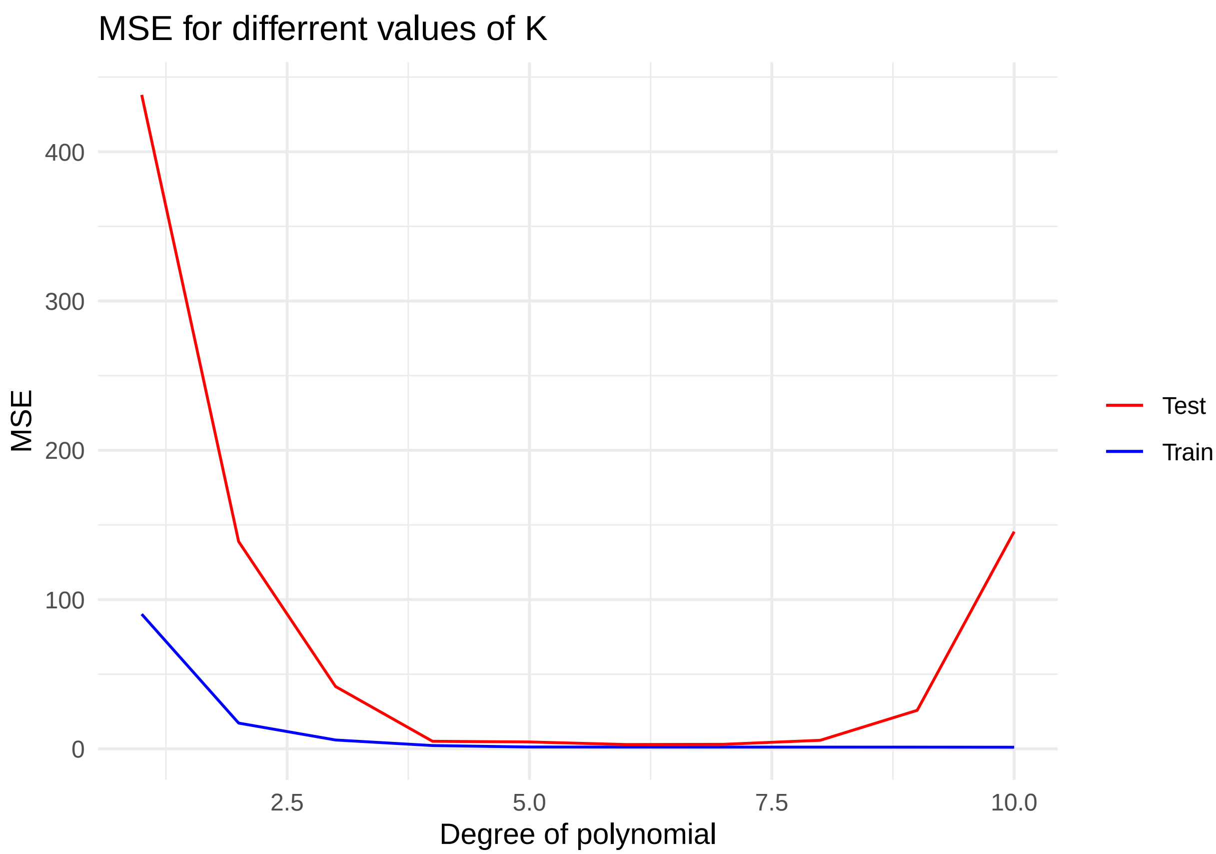

```
## [1] "small_train2"
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
## [[1]]
## [1] 17.28644 17.08568
##
## [[2]]
## [1]  5.527188 15.712988 10.308757
##
## [[3]]
## [1] 5.454692 3.272884 9.576782 4.601331
##
## [[4]]
## [1] 7.564425 2.185361 1.475406 4.718731 2.281718
##
## [[5]]
## [1]  7.5662653  1.5122462  1.5105481  5.5292915  2.2778980 -0.1752305
##
## [[6]]
## [1]  7.8187654  1.6135826 -0.5984112  5.3266644  3.9076900 -0.1220670 -0.2928723
##
## [[7]]
## [1]  7.74505857  2.43187352  0.05441751  2.85196654  3.27698887  1.39381750
## [7] -0.16485190 -0.24539199
##
## [[8]]
## [1]   9.0864940   3.2130723 -15.2281830  -1.1033002  24.2760159   4.5915548
```

```
## [7]  -9.2723656  -0.8467187   1.2214483
##
## [[9]]
##  [1]   8.8113504   5.2077393 -11.8008198 -14.8914096  17.8953634  21.1902507
##  [7]  -5.6573093  -7.5706294   0.6383209   0.8661229
##
## [[10]]
##  [1]   11.6637025   -0.2235431  -58.0775124   14.8885799  158.0099396
##  [6]   -2.4499743 -136.8767163   -3.6006087   47.0093644    0.9960223
## [11]   -5.4631005
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x          y1         y2
## 1   1 230.4641021 298.281854
## 2   2  41.3686368  82.644572
## 3   3  10.7219987  22.988521
## 4   4   2.6314235   5.629123
## 5   5   2.5868202   6.167429
## 6   6   2.5028055   6.199278
## 7   7   2.4282766   6.724248
## 8   8   1.6970697  12.356811
## 9   9   1.5913511   9.757250
## 10 10   0.4771781 286.947548
```



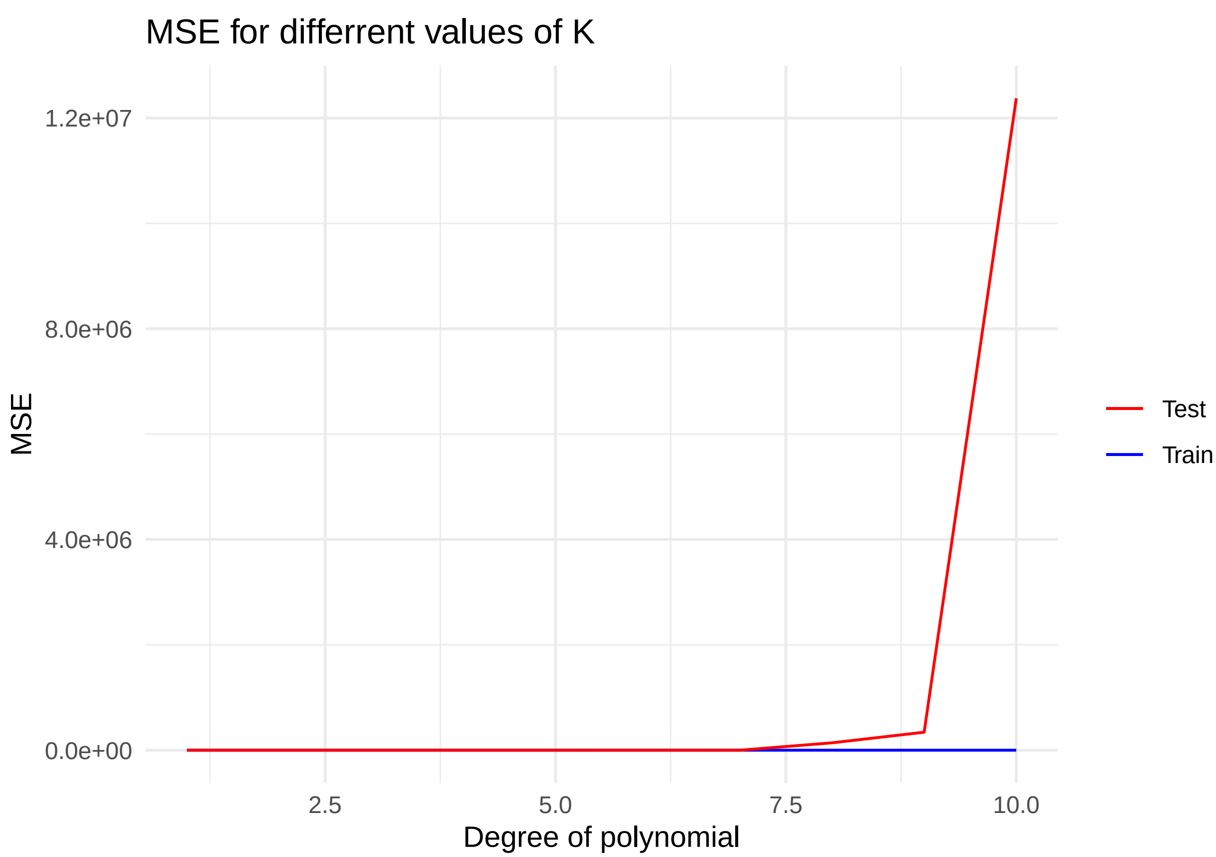

```
## [1] "large_train2"
```

```
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
## [[1]]
## [1] 15.49925 11.57116
##
## [[2]]
## [1]  6.753970 13.108130  6.583517
##
## [[3]]
## [1] 6.217797 3.860847 7.536654 3.588779
##
## [[4]]
## [1] 7.669666 3.175305 1.033885 4.191767 2.102142
##
## [[5]]
## [1]  7.9790208  6.2221442 -0.2923934  0.2487080  2.6292920  0.9908631
##
## [[6]]
## [1]  8.3410445  5.9864062 -3.2654784  0.8954613  4.9062001  0.7793007 -0.4399983
##
## [[7]]
## [1]  8.3368767  5.8876472 -3.2265726  1.3123251  4.8318777  0.4986687 -0.4175722
## [8]  0.0505258
##
## [[8]]
## [1]  8.6123841  5.5850266 -5.9675059  2.4248231  8.5550573 -0.3127534 -2.1075857
## [8]  0.2114292  0.2471555
##
## [[9]]
##  [1]  8.5903286  6.5059059 -6.1993779 -3.8521685 10.0420474  6.7958727
##  [7] -3.1994606 -2.6309178  0.4563880  0.3781869
##
## [[10]]
##  [1]  8.2780414  7.1068373 -2.0088395 -7.4053720 -1.9787026 11.9561617
##  [7]  7.8047435 -5.1763910 -3.5416301  0.7784600  0.5044435
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x        y1         y2
## 1   1 90.240691 438.073761
## 2   2 17.248005 138.863935
## 3   3  5.918811  41.748396
## 4   4  2.201385   5.068467
## 5   5  1.236560   4.612989
## 6   6  1.167758   2.909228
## 7   7  1.166510   3.046615
## 8   8  1.148167   5.728867
## 9   9  1.124857  25.799732
## 10 10  1.070496 145.482754
```



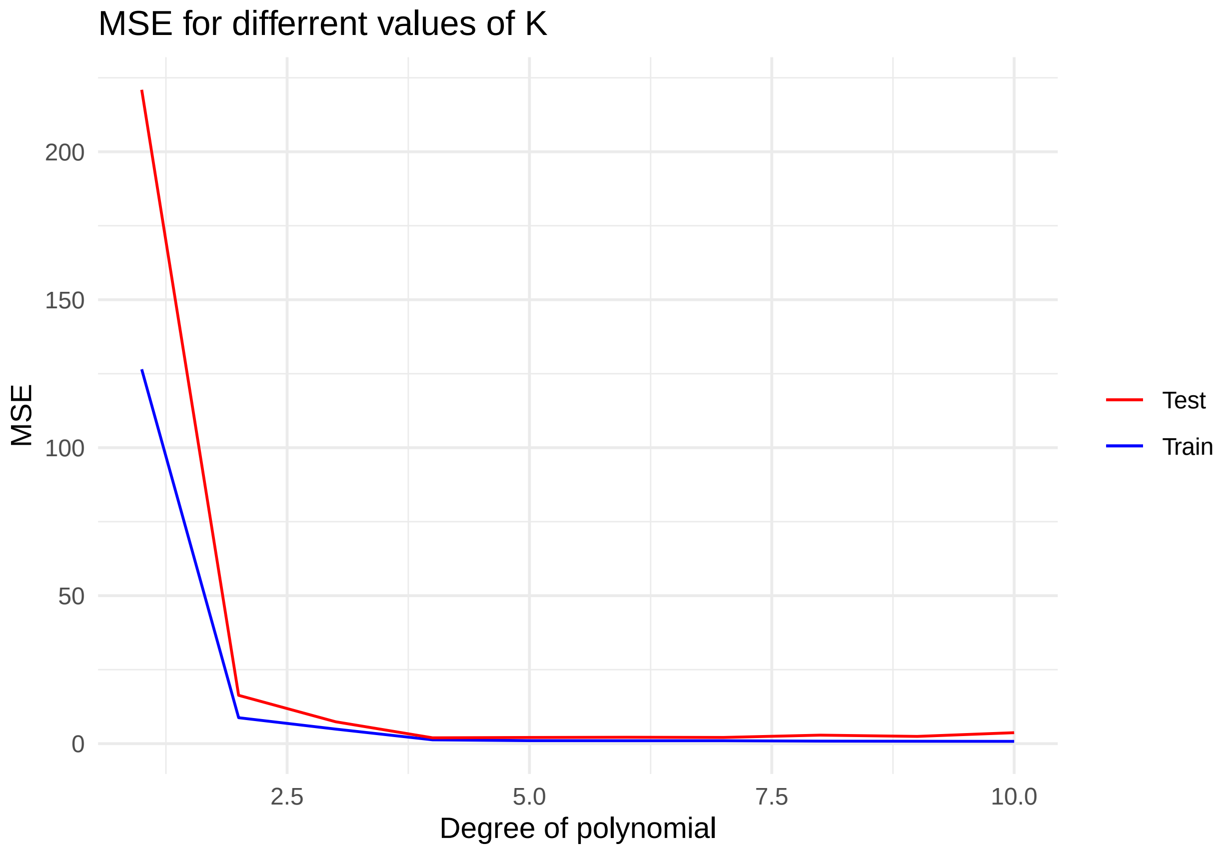

```
## [1] "small_train3"
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
## [[1]]
## [1] 8.914003 2.718080
##
## [[2]]
## [1] 4.311399 7.029707 5.888304
##
## [[3]]
## [1] 4.30673127 6.99609264 5.90605301 0.02149452
##
## [[4]]
## [1] 4.882050 4.606732 3.290349 2.125693 1.504044
##
## [[5]]
## [1]  4.4601454  2.6124416  7.1237251  5.2646751 -0.7274763 -1.3389315
##
## [[6]]
## [1]  4.707015  1.478363  4.018898  9.352491  2.897134 -3.296693 -1.234775
##
## [[7]]
## [1]  4.410216  1.132979  8.176701 10.981817 -3.504036 -5.870698  1.147742
## [8]  1.072189
##
## [[8]]
## [1]   4.192527   5.584118  12.532240 -23.683858 -14.293415  38.561387  16.547841
```

```
## [8] -14.065796  -6.332985
##
## [[9]]
##  [1]   4.120382   5.616312  14.860787 -23.404855 -25.958766  36.615473
##  [7]  30.305729 -10.019797 -10.886669  -1.804690
##
## [[10]]
##  [1]   3.910547   4.093823  22.556114   0.496646 -65.056273 -49.402265
##  [7]  66.468134  87.667891  -3.140282 -34.493870 -10.277509
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x         y1           y2
## 1   1 27.9602339 4.457262e+02
## 2   2  2.0936462 8.300451e+01
## 3   3  2.0934227 8.188083e+01
## 4   4  1.0210157 2.813325e+00
## 5   5  0.6336013 1.700593e+02
## 6   6  0.4772344 9.593558e+02
## 7   7  0.4295020 1.810553e+01
## 8   8  0.2164968 1.405368e+05
## 9   9  0.2135462 3.418990e+05
## 10 10  0.1342066 1.237566e+07
```




```
## [1] "large_train3"
## [1] "PARAMETERS VECTORS FOR EACH VALUE OF K"
```

```
## [[1]]
## [1] 12.09655 10.01560
##
## [[2]]
## [1]  2.466580 10.205676  8.653536
##
## [[3]]
## [1] 2.655854 5.541946 8.524401 1.845059
##
## [[4]]
## [1] 3.995074 5.009388 3.178168 1.997241 1.641315
##
## [[5]]
## [1]  3.9048783  6.8653423  3.1872974 -0.1465299  1.6475343  0.4807962
##
## [[6]]
## [1] 3.79269810 6.70194901 4.31755768 0.05079736 0.74977298 0.43634663 0.16797230
##
## [[7]]
## [1]  3.79731239  6.83633604  4.26671033 -0.35479708  0.77398433  0.68807077
## [7]  0.16537925 -0.04190847
##
## [[8]]
## [1]  3.3384101  6.4490450 11.0792999 -1.0189252 -9.5467526  1.3045309  4.9728317
## [8] -0.1518086 -0.6848162
##
## [[9]]
##  [1]  3.3406632  7.2951141 11.3670012 -7.0952675 -9.4101196  8.9604225
##  [7]  4.7505081 -3.3924850 -0.6406549  0.4357818
##
## [[10]]
##  [1]  3.6014097  7.3853124  6.6218736 -6.0792847  2.3709563  7.3185849
##  [7] -5.4223633 -2.6229194  2.9048455  0.3267835 -0.4252773
##
## [1] "Let y1 be the mse values for training dataset, y2 be the mse values for testing dataset"
## [1] "MSE"
##     x          y1         y2
## 1   1 126.4979416 220.937512
## 2   2   8.7625695  16.324396
## 3   3   4.9184891   7.392283
## 4   4   1.2860318   1.970621
## 5   5   1.0110131   2.086343
## 6   6   0.9916978   2.158495
## 7   7   0.9904718   2.099509
## 8   8   0.8572659   2.871519
## 9   9   0.8078484   2.445929
## 10 10   0.7686227   3.696438
```



It can be noticed that the mean squared errors of the test dataset using the model trained by the smaller dataset are generally larger than that trained by the larger one. This might be due to the fact that it is harder for small dataset to capture the pattern of an unseen dataset, as it has high bias and low variance (underfitting). The difference in size may also result in relatively different parameters and best model polynomial degree.

After training the model on both small and large datasets, and looking at the results for the mean squared errors of the test datasets, it can be observed that the MSE values are always lowest at K = 2 for dataset 1, K = 6 for dataset 2, and K = 4 for dataset 3. In this case, overfitting might be the problem, as the increase of model complexity results in higher MSE. Therefore, it can be concluded that the unseen dataset (test) is best generalized with a polynomial model with degree 2, 6 and 4 for datasets 1, 2 and 3 respectively.

Using the results of the models' parameter vectors above, we have the following models that best generalized the data:

Dataset 1:

$$y = f(x) = 3.004289 + 1.963497x + 3.948467x^2$$

Dataset 2:

$$y = f(x) = 8.3410445 + 5.9864062x - 3.2654784x^2 + 0.8954613x^3 + 4.9062001x^4 + 0.7793007x^5 - 0.4399983x^6$$

Dataset 3:

$$y = f(x) = 3.995074 + 5.009388x + 3.178168x^2 + 1.997241x^3 + 1.641315x^4$$

During the training of the models, underfitting and overfitting are inevitable, as analyzed above. In order to avoid the underfitting issue, we can either increase the models' complexity (for example, the degree), or increase the size of the datasets (for example, the model with large_train1 dataset performs better). On the other hand, overfitting can be prevented by reducing the model complexity and choosing the right polynomial degree.

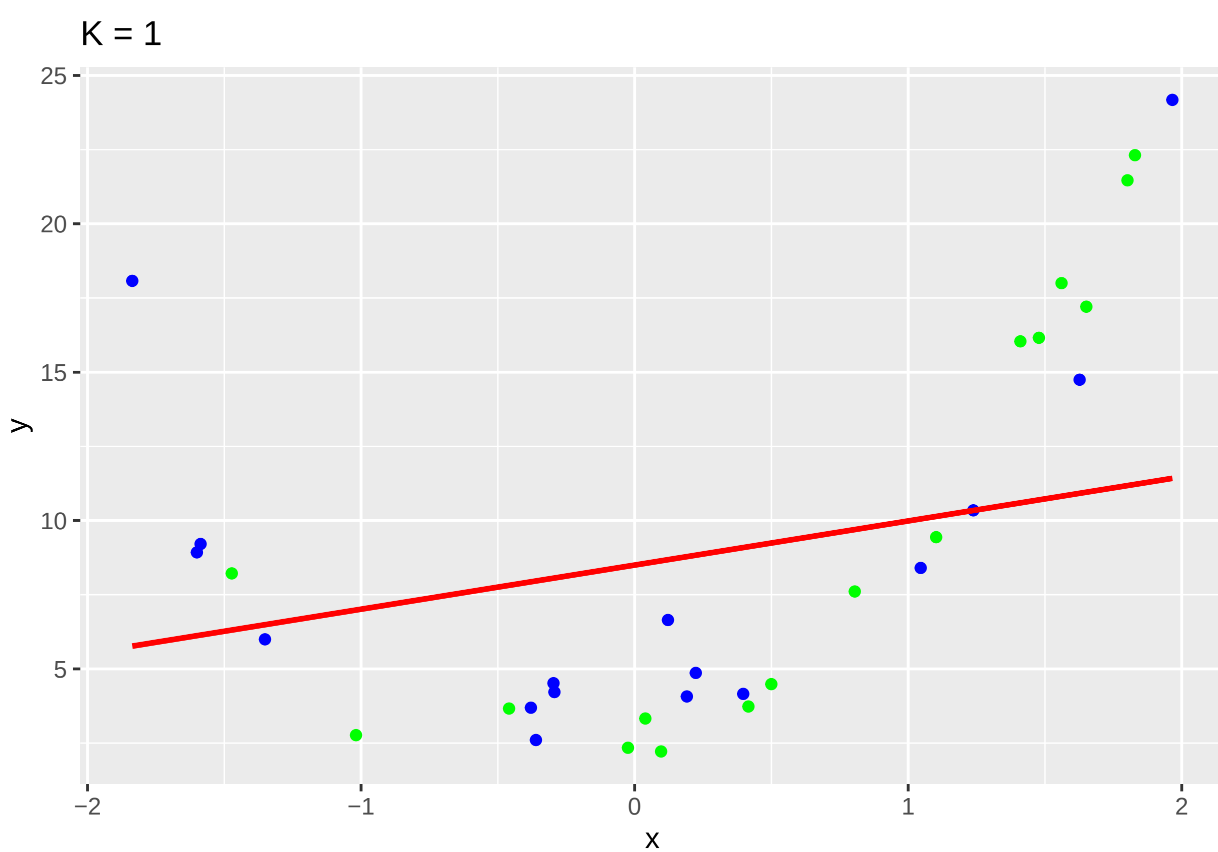
```
mse_train <- c()
mse_test <- c()
for (k in 1:10) {
  model <- lm(y ~ poly(x, k, raw=TRUE), data = small_train3)
  train_fit <- model %>% predict(small_train3)
  train_mse <- mean((small_train3$y - train_fit)^2)
  mse_train[k] <- train_mse
  test_fit <- model %>% predict(test_data1)
  test_mse <- mean((test_data1$y - train_fit)^2)
}
mse_train
```

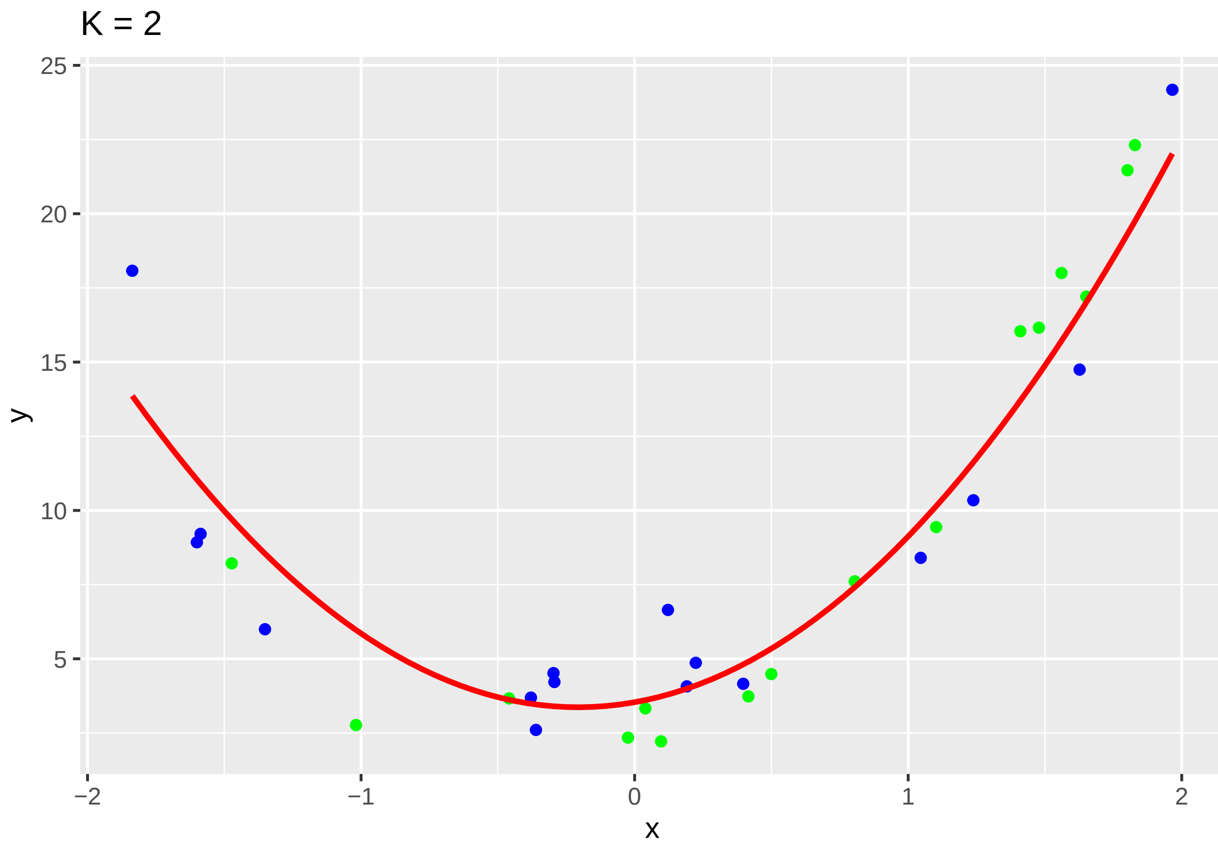
```
##  [1] 27.9602339  2.0936462  2.0934227  1.0210157  0.6336013  0.4772344
##  [7]  0.4295020  0.2164968  0.2135462  0.1342066
```

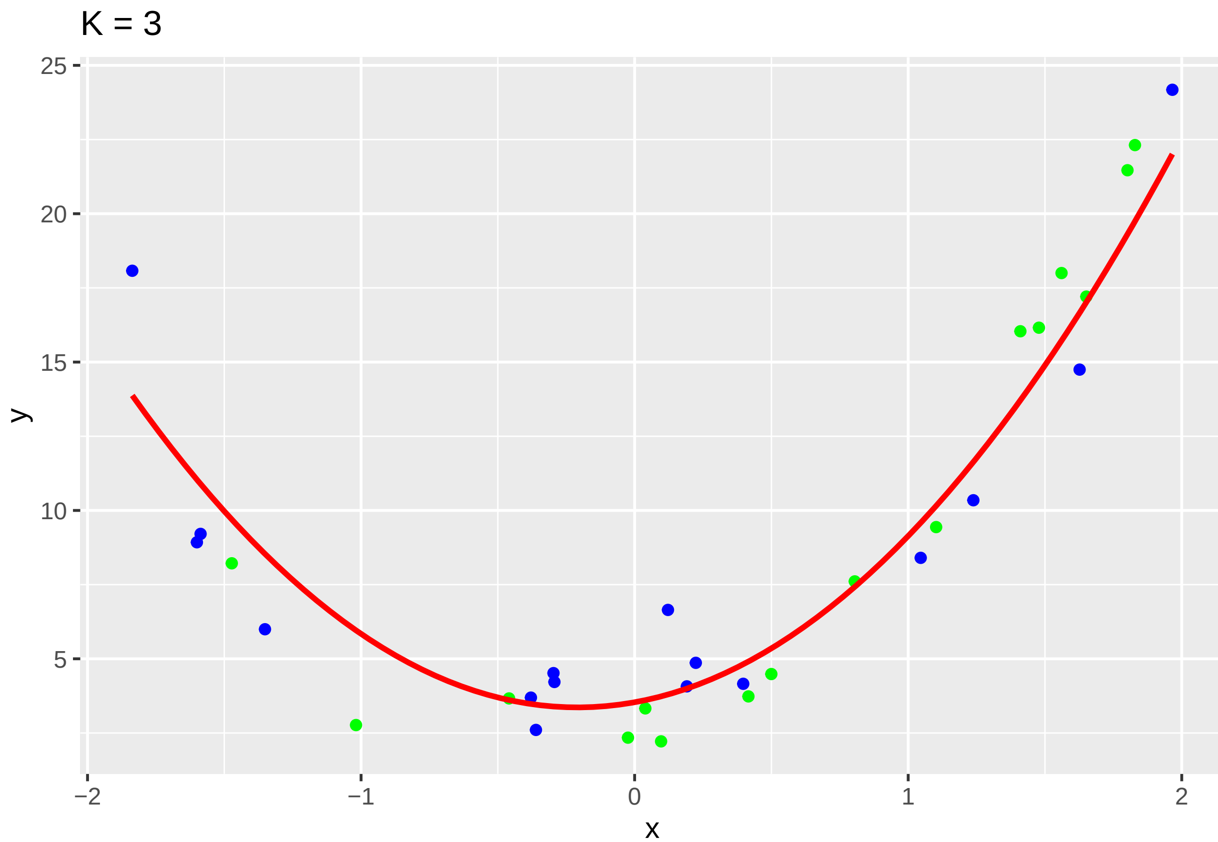
Appendix: Compare model fits for different values of K:

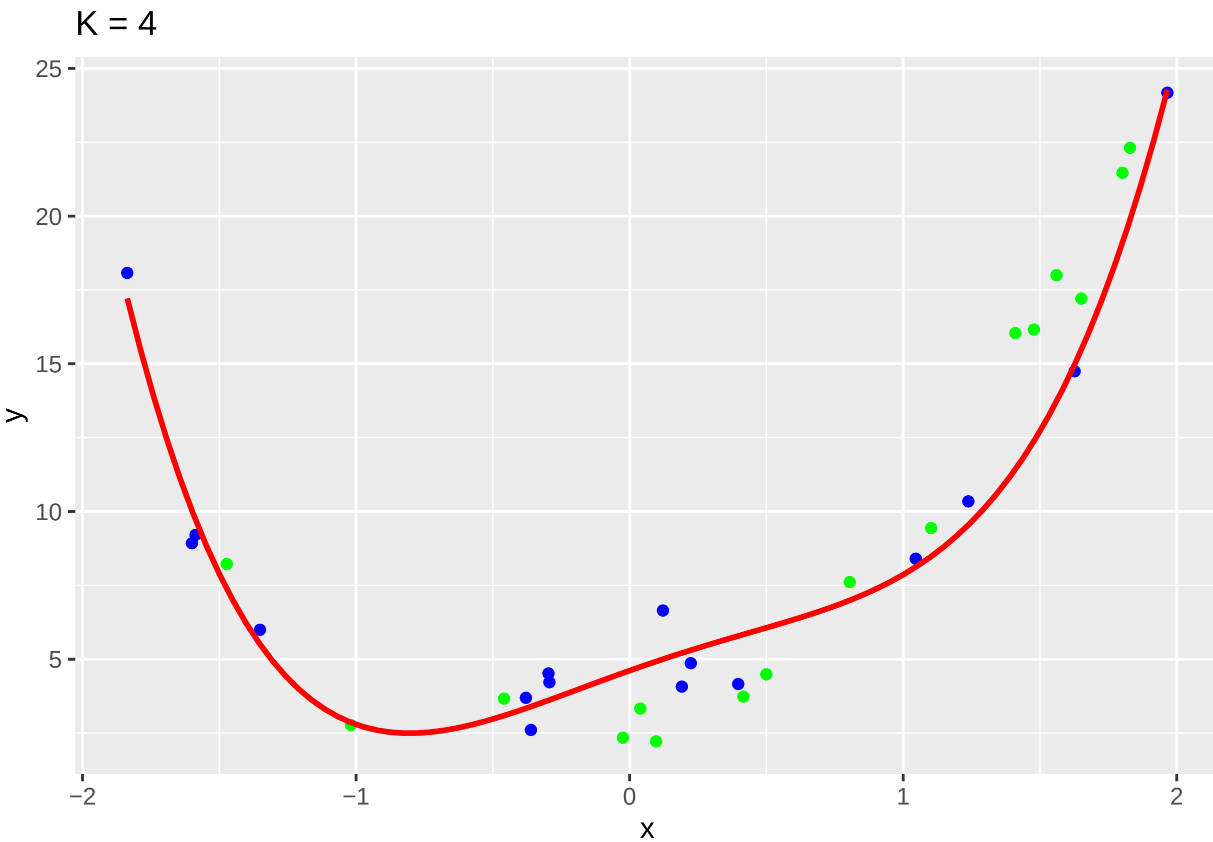
Model fits for K from 1 to 10 for small_train1 dataset:

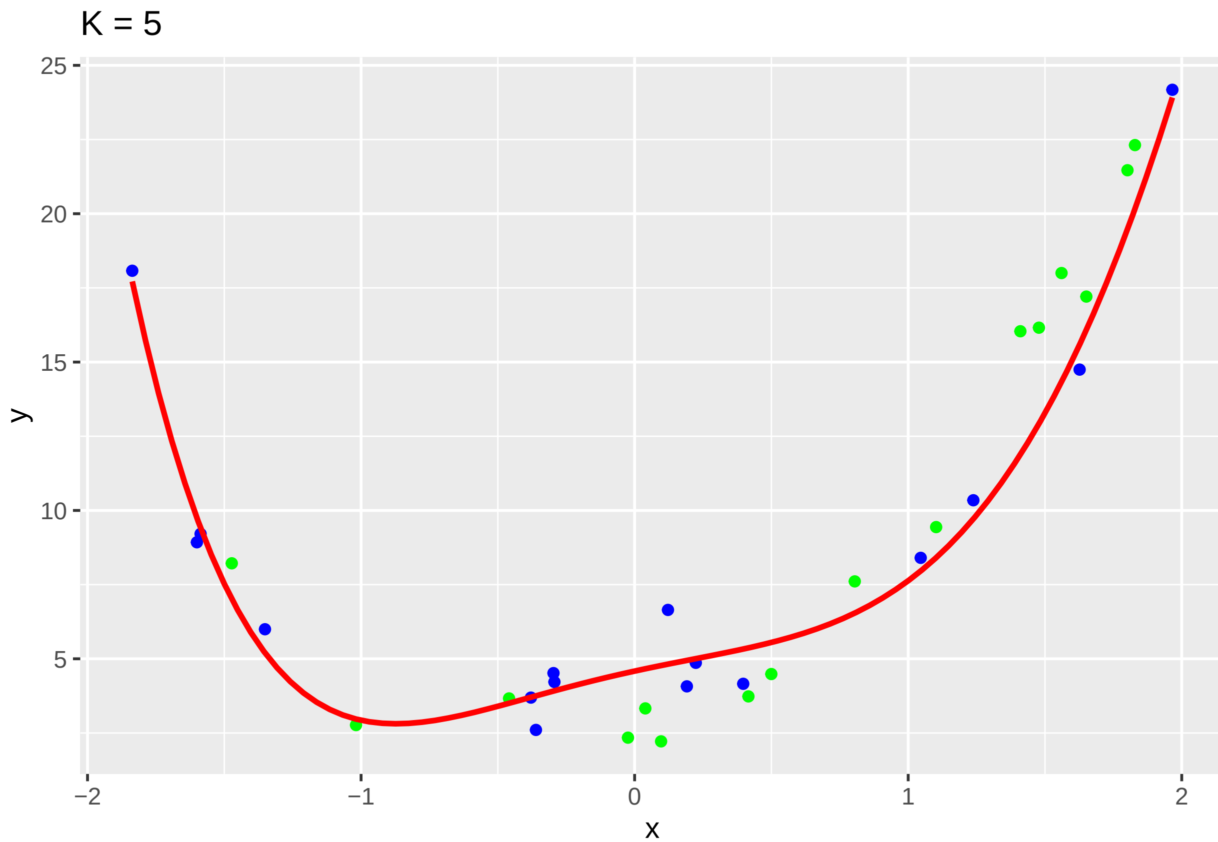
```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=small_train1, color="blue") + geom_point(data=
  print(g)
}
```

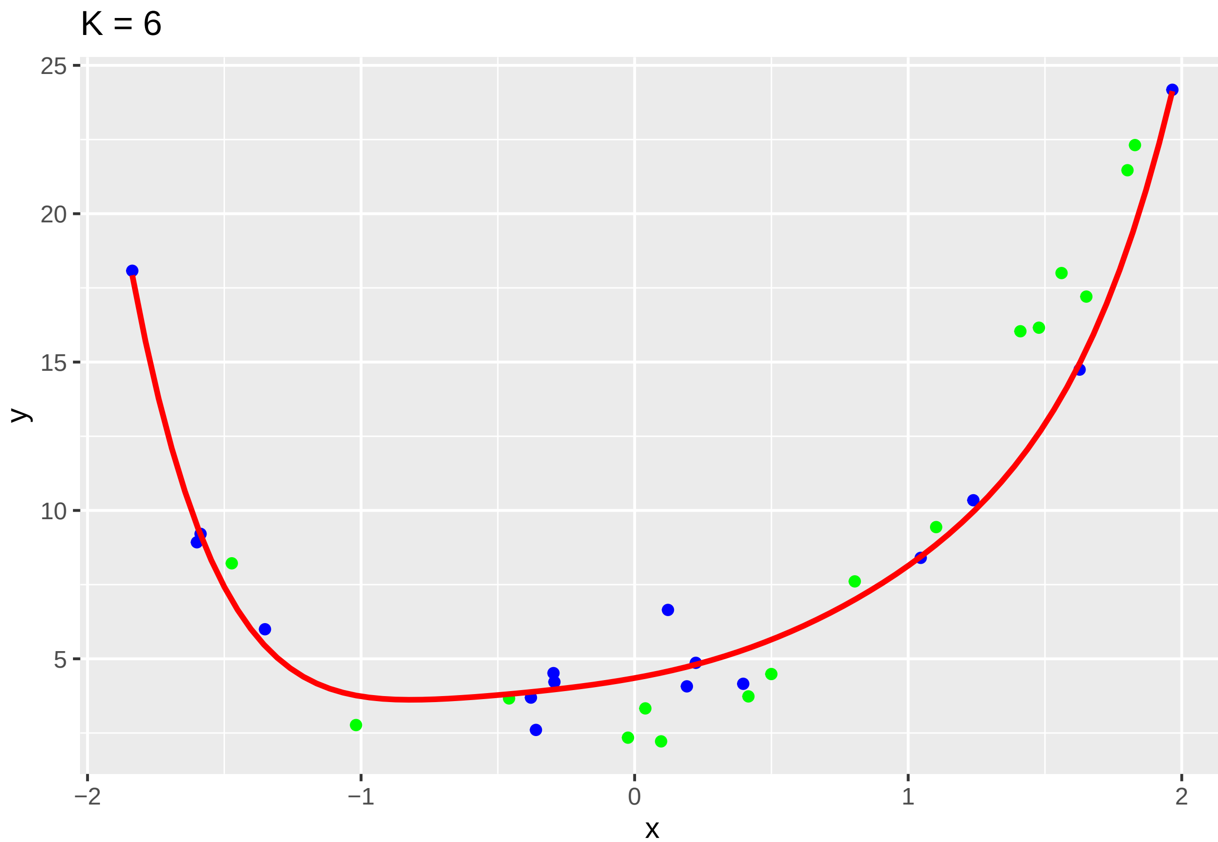



K = 2
25
20
15
10
5
y
−2
−1
0
1
2
x

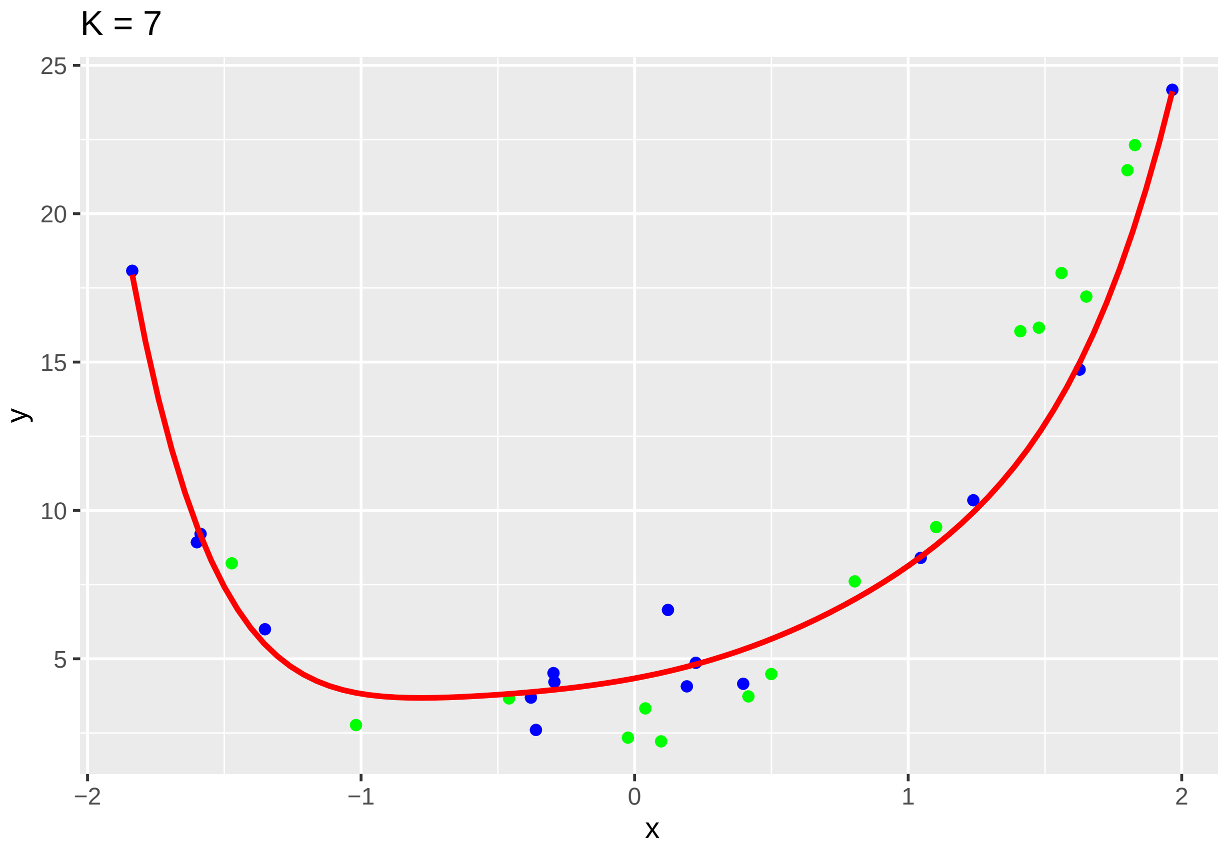

K = 3
y
x
25
20
15
10
5
−2
−1
0
1
2


K = 4
25
20
15
10
5
y
−2
−1
0
1
2
x

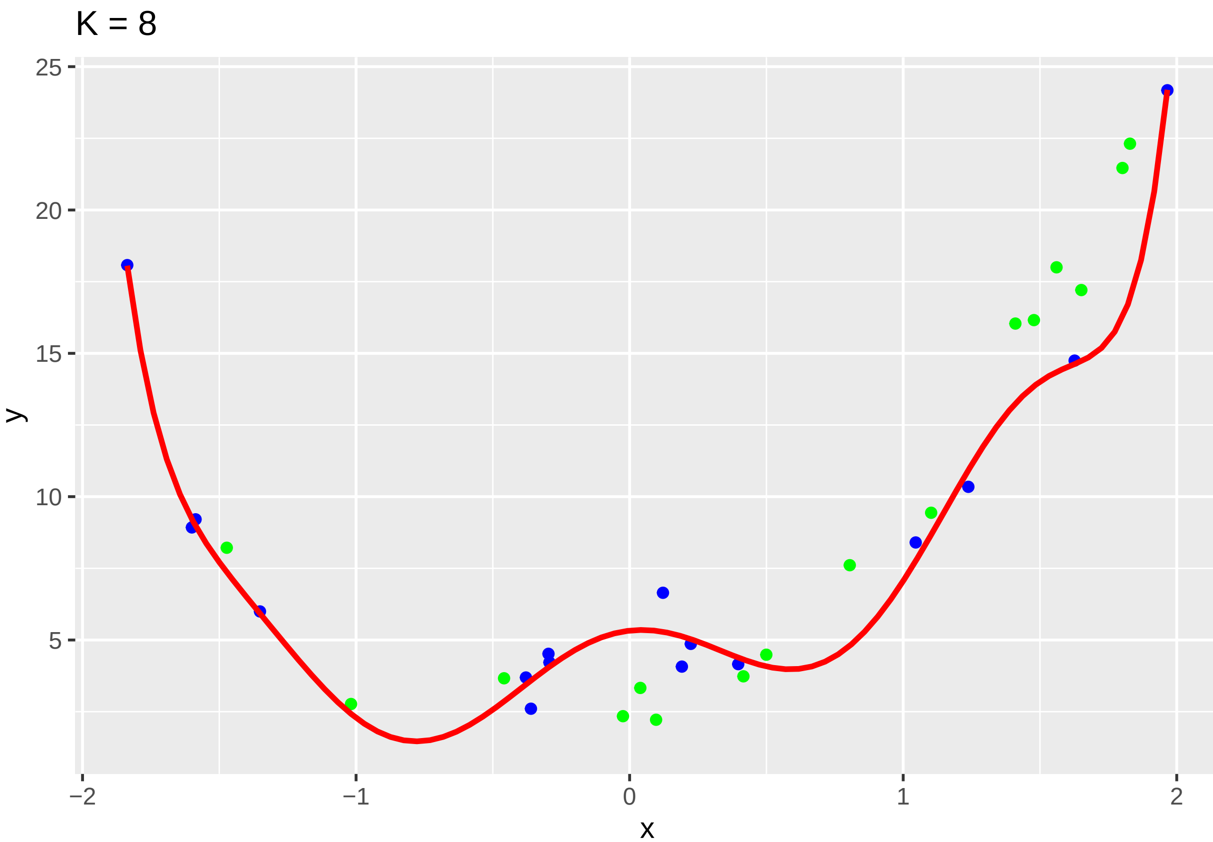

K = 5
25
20
15
10
5
y
−2
−1
0
1
2
x



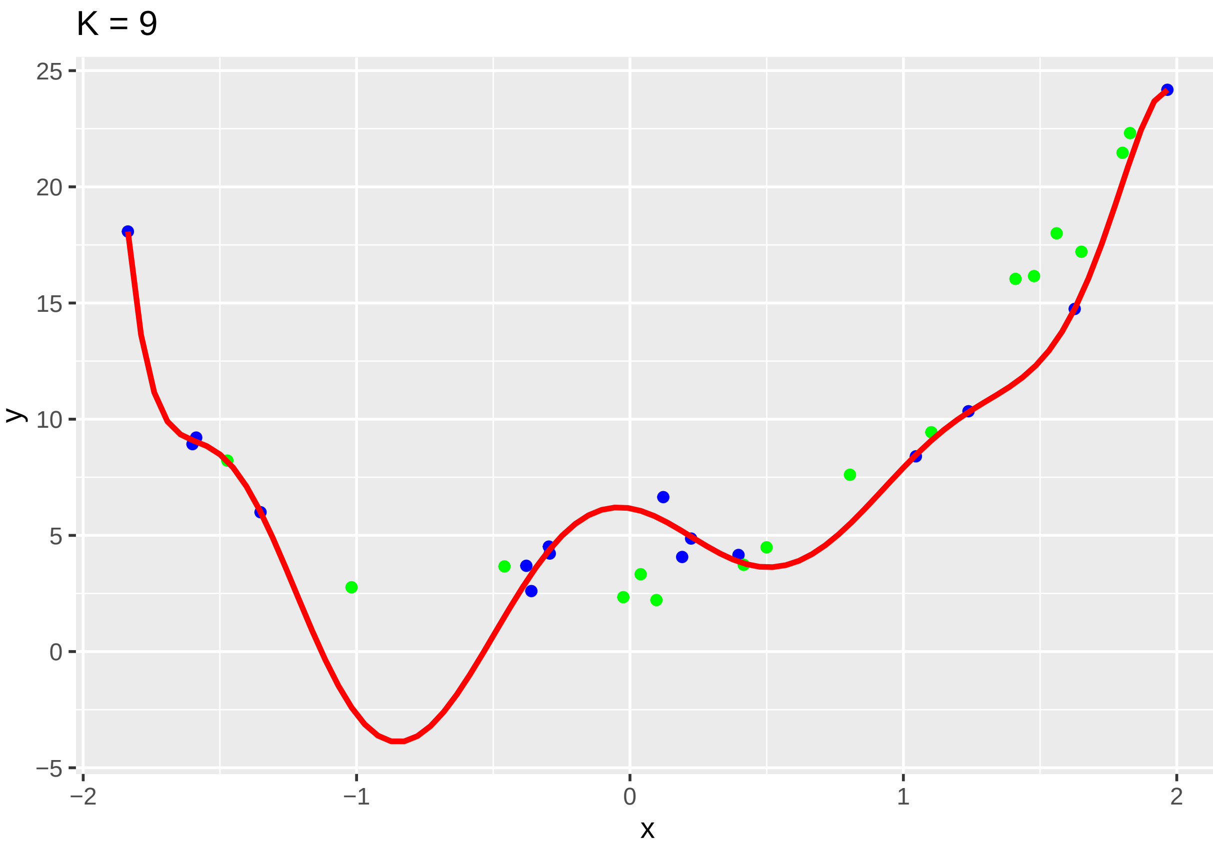
K = 6
25
20
15
10
5
y
−2
−1
0
1
2
x



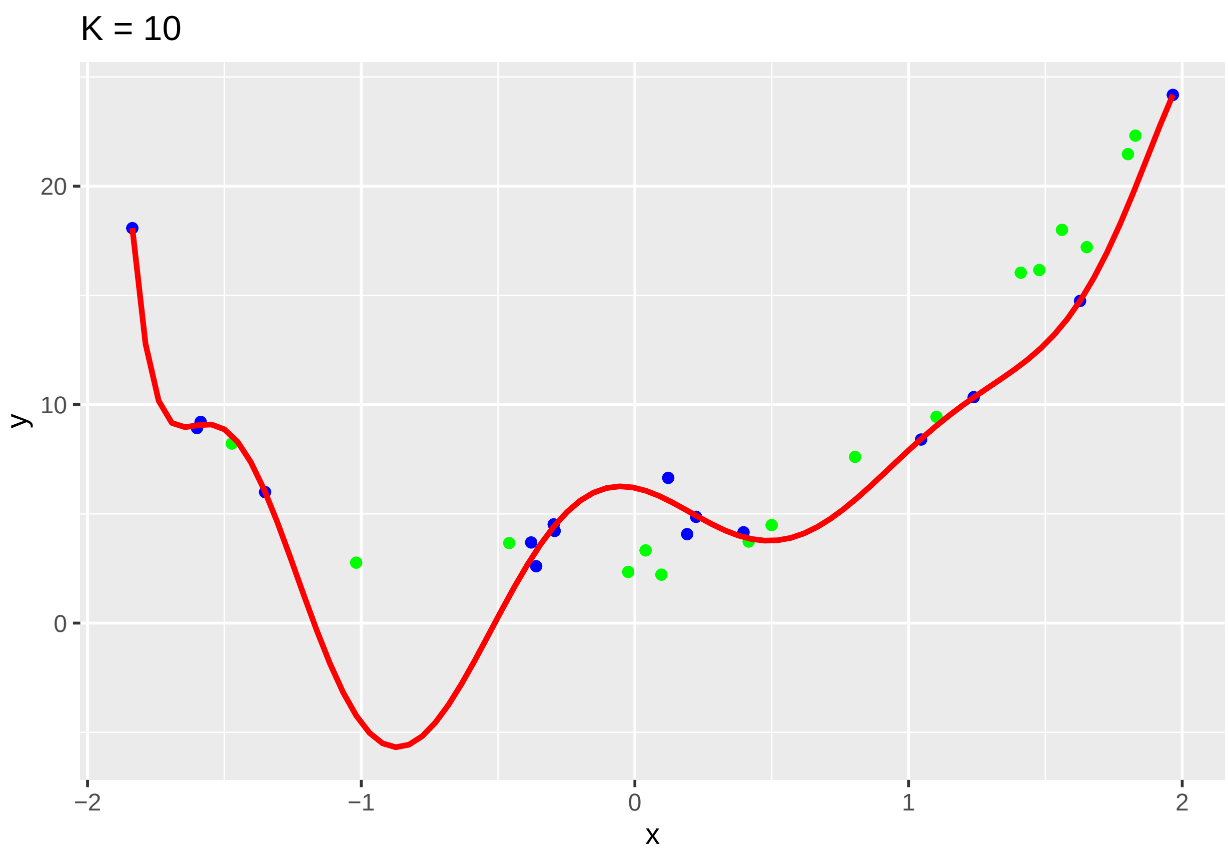
K = 7
25
20
15
10
5
y
−2
−1
0
1
2
x



K = 8
25
20
15
10
5
y
−2
−1
0
1
2
x

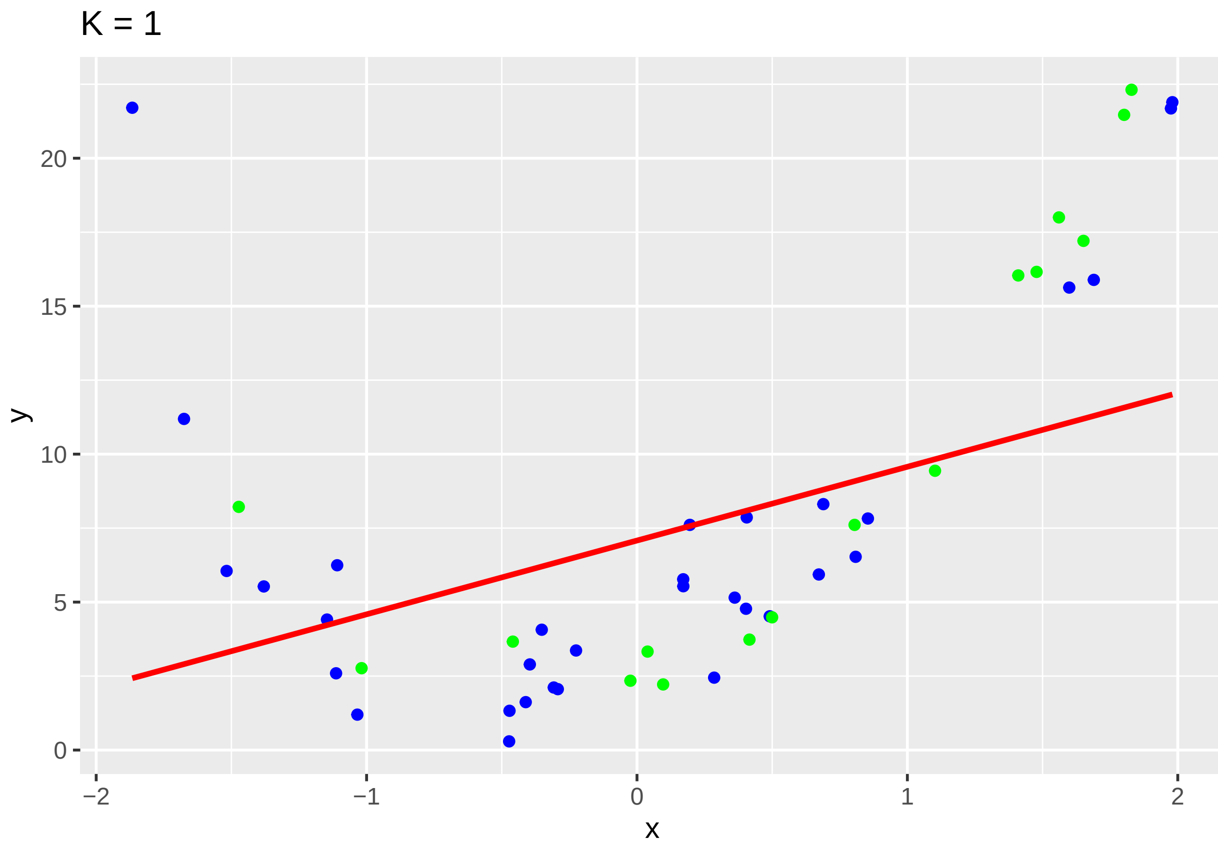


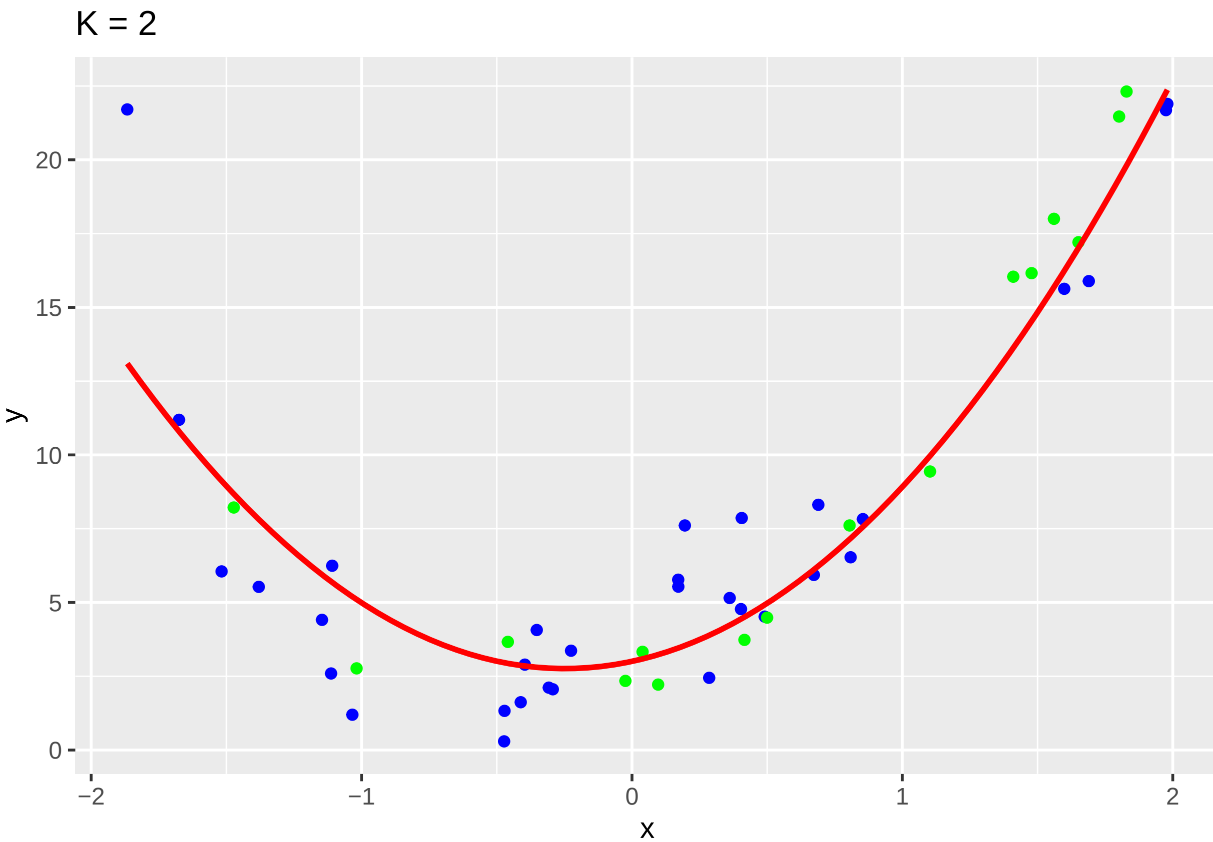
K = 9
25
20
15
10
5
0
−5
y
−2
−1
0
1
2
x

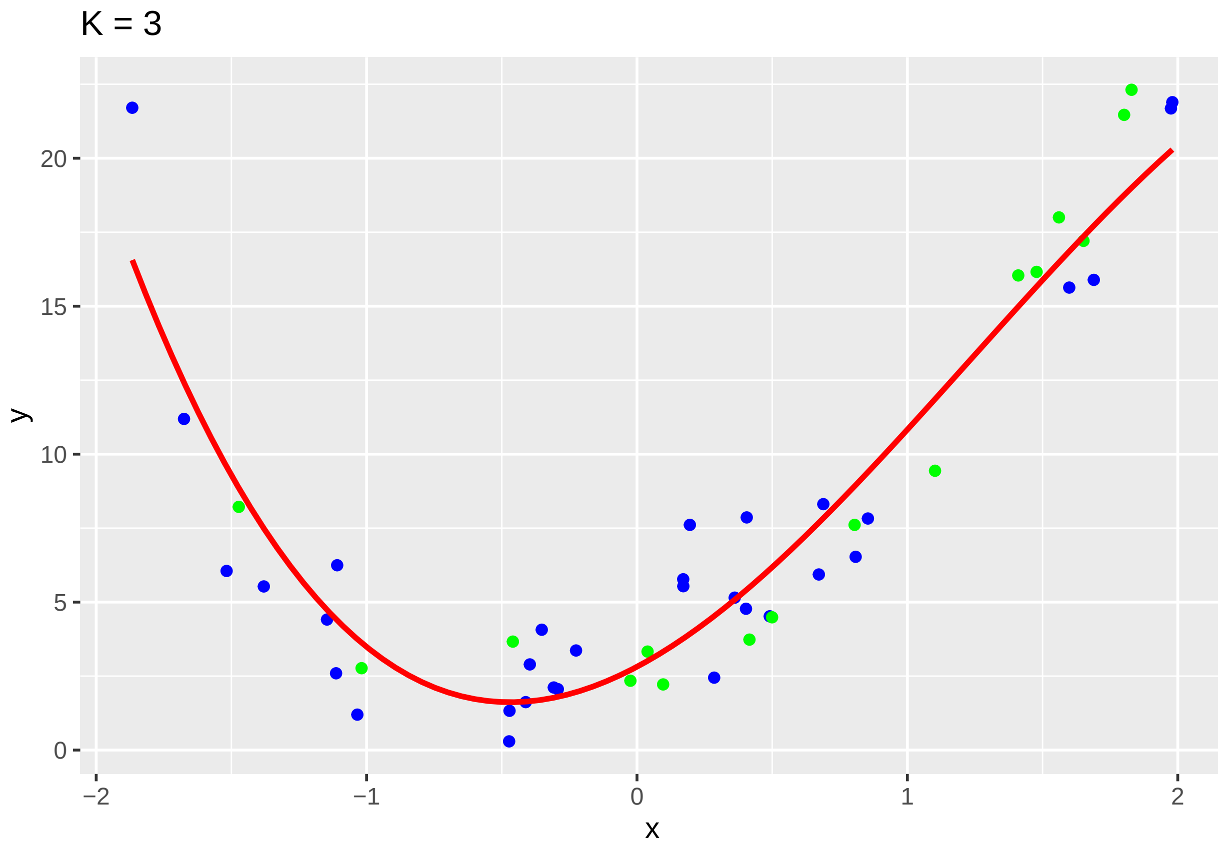


Model fits for K from 1 to 10 for large_train1 dataset:

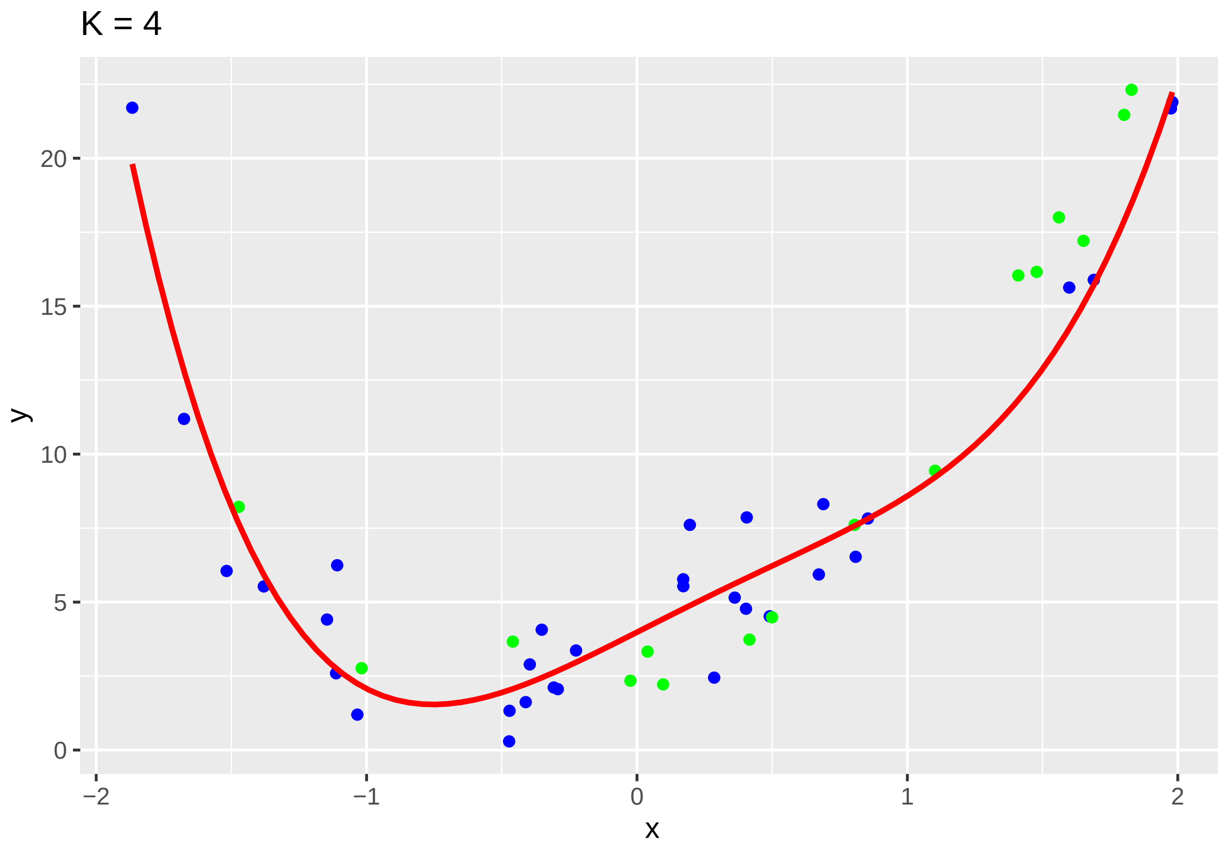
```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=large_train1, color="blue") + geom_point(data=
  print(g)
}
```

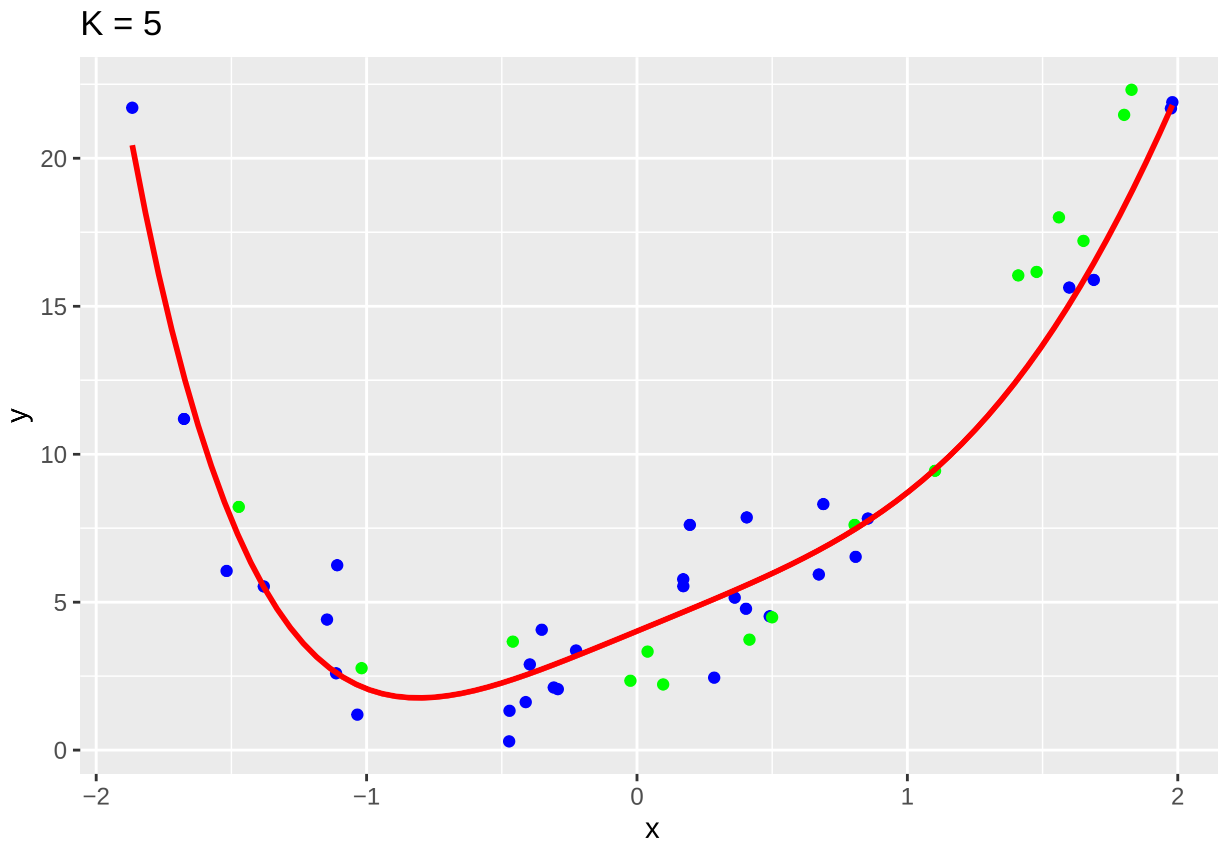

K = 1
y
20
15
10
5
0
−2
−1
0
1
2
x


K = 2
y
20
15
10
5
0
−2
−1
0
1
2
x

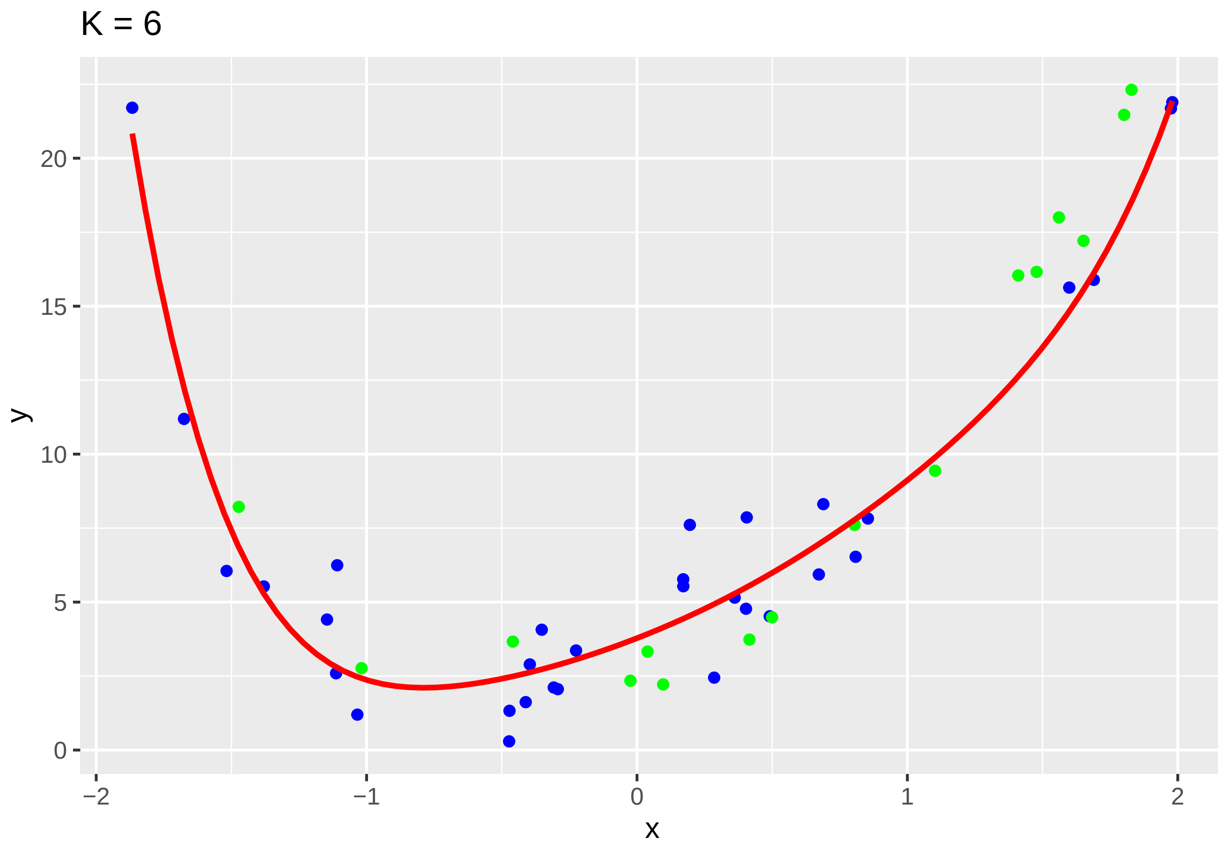


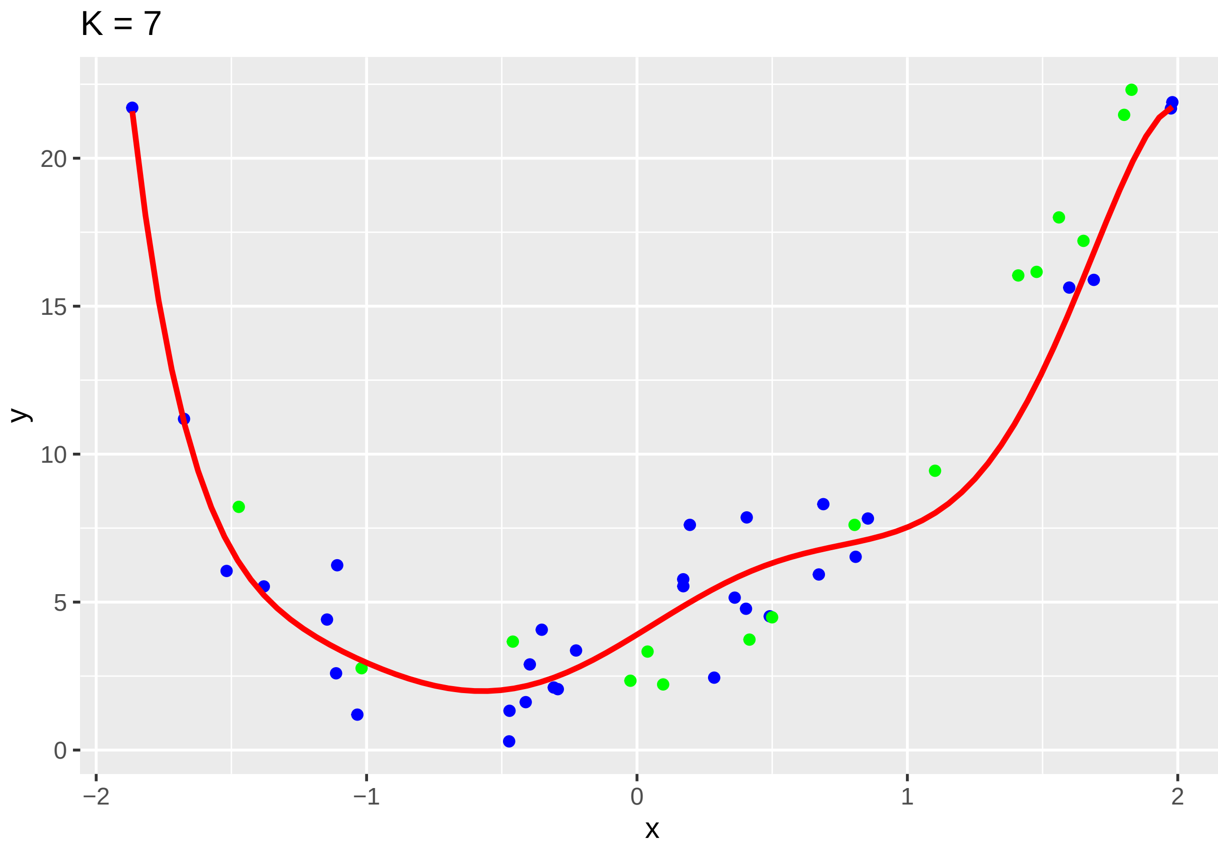
K = 3
y
x
0
5
10
15
20
−2
−1
0
1
2


K = 4
y
x
0
5
10
15
20
−2
−1
0
1
2

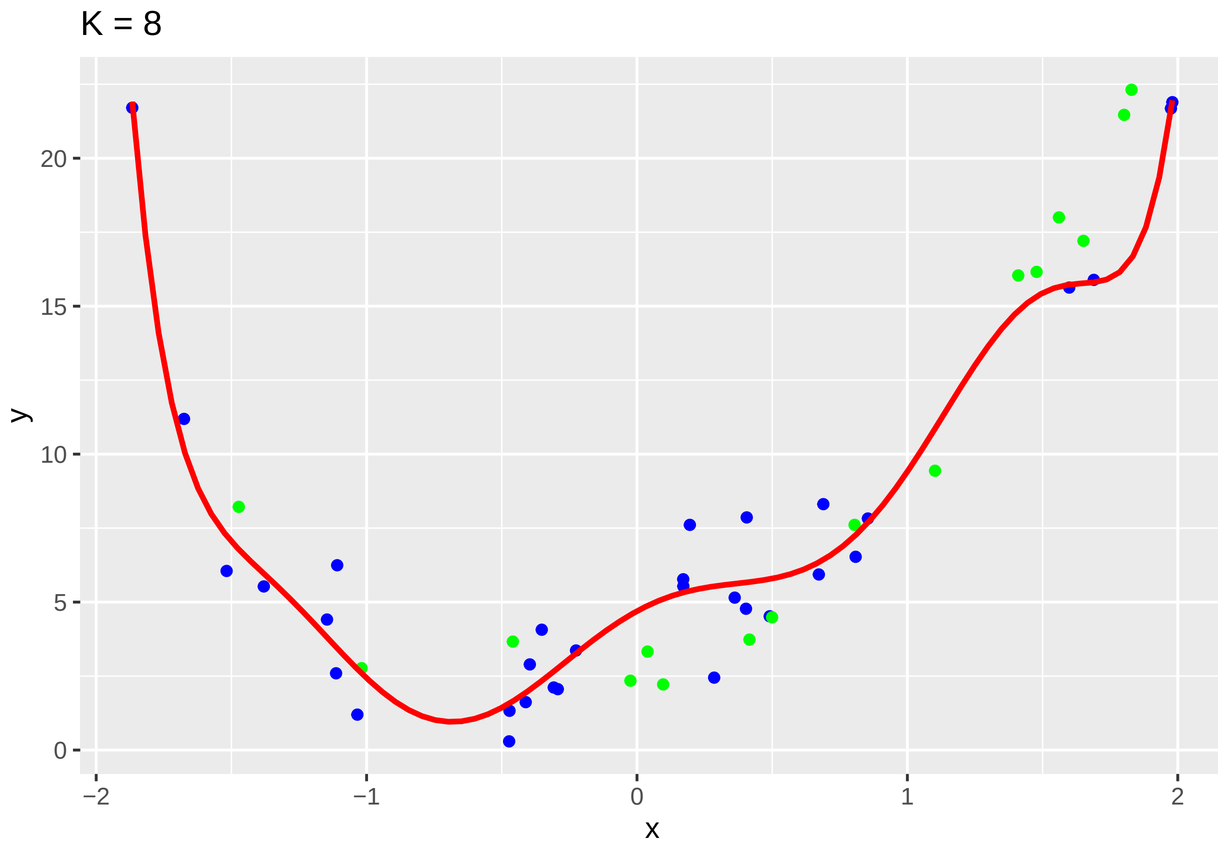


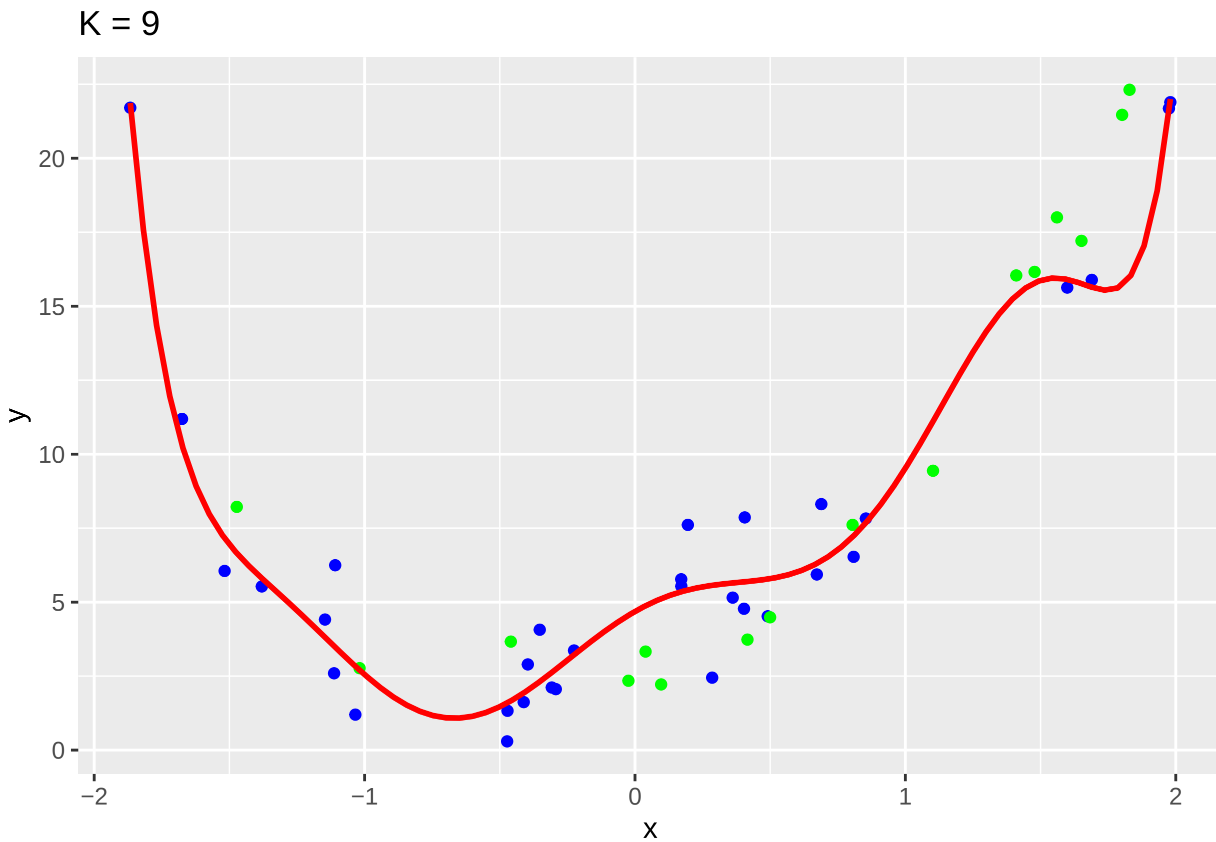
K = 5
y
x
20
15
10
5
0
−2
−1
0
1
2

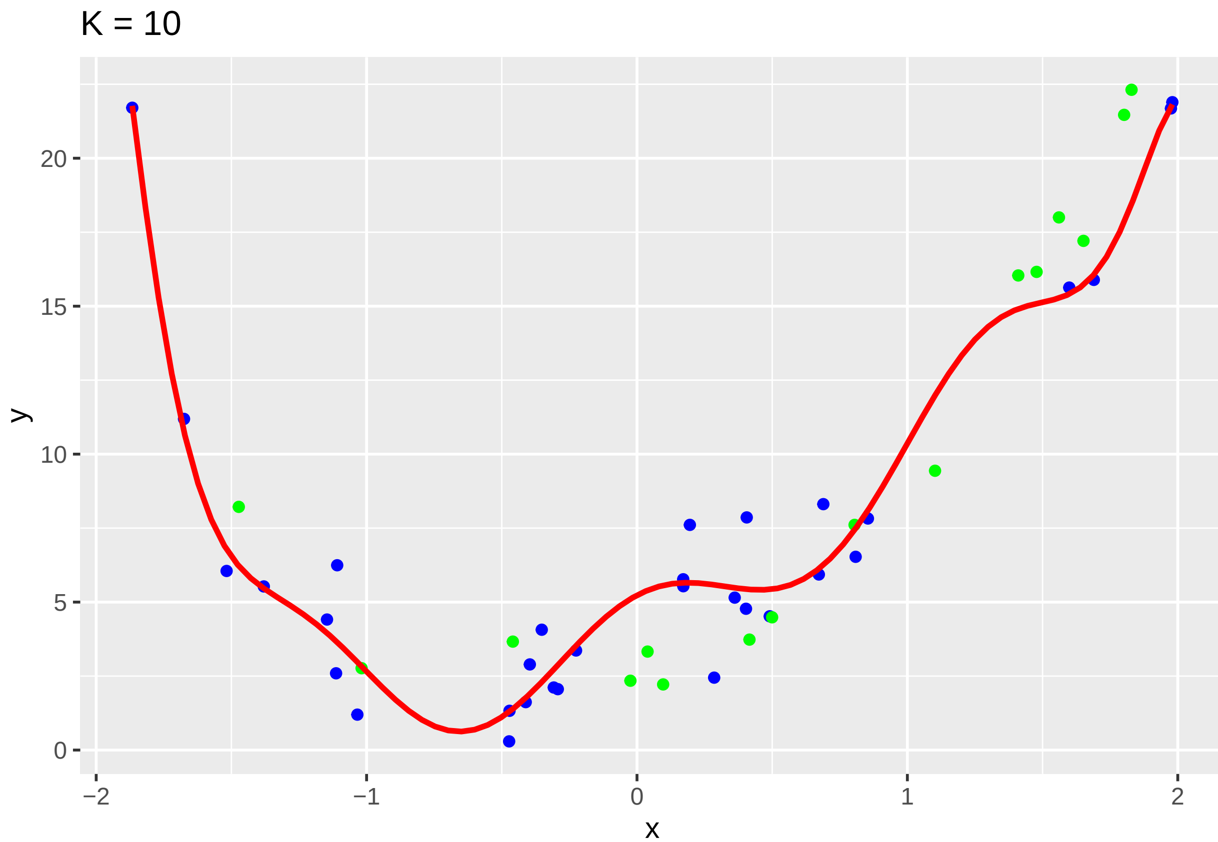

K = 6
y
20
15
10
5
0
−2
−1
0
1
2
x



K = 7
20
15
10
5
0
y
−2
−1
0
1
2
x

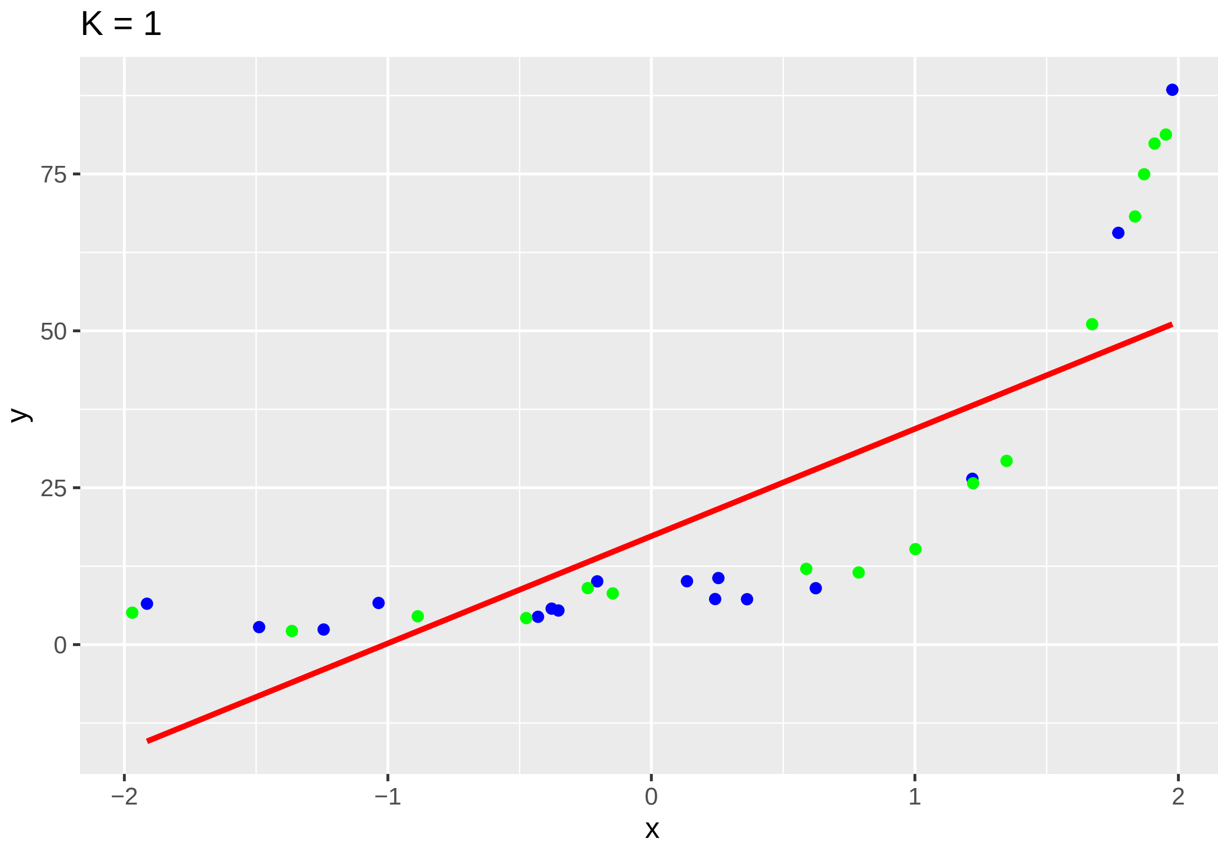

K = 8
y
x
20
15
10
5
0
−2
−1
0
1
2

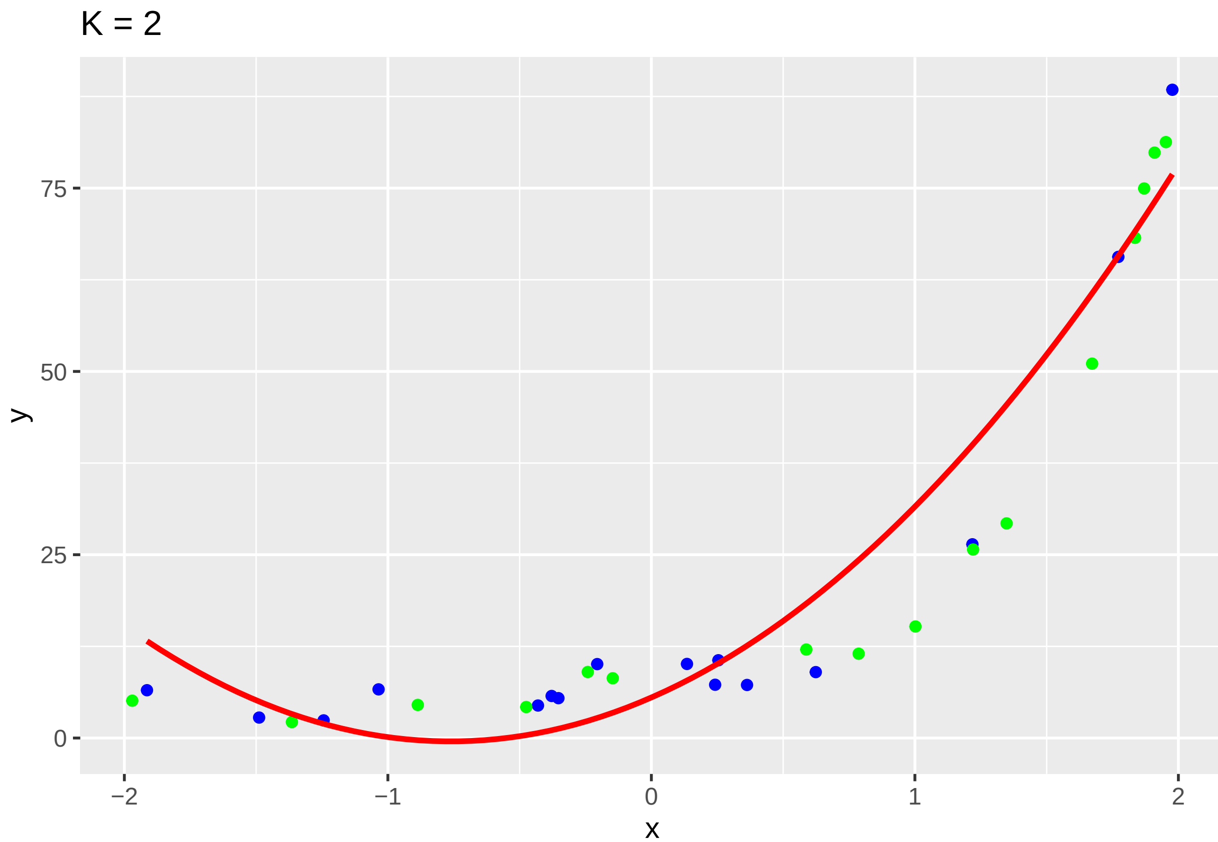

K = 9
y
x
0
5
10
15
20
−2
−1
0
1
2



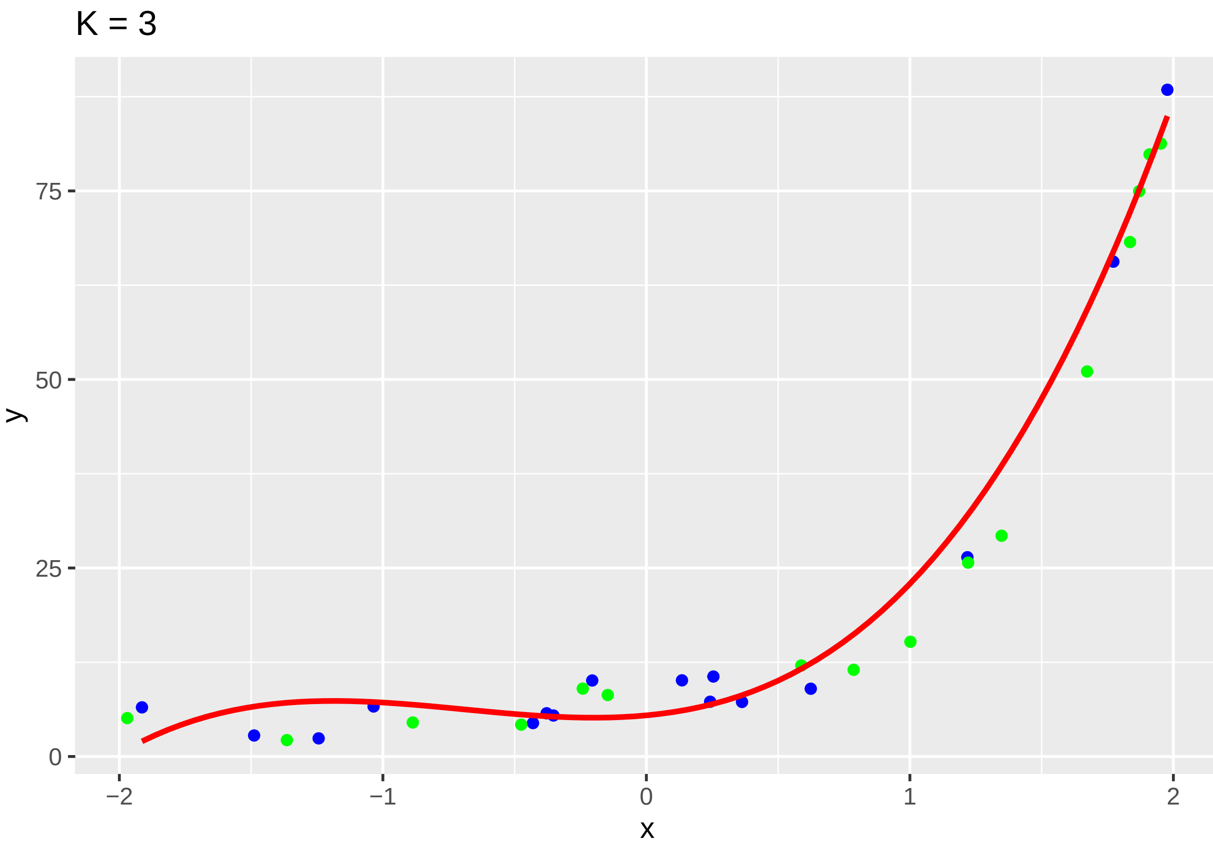
Model fits for K from 1 to 10 for small_train2 dataset:

```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=small_train2, color="blue") + geom_point(data=
    ggtitle(paste("K =", k))
  print(g)
}
```

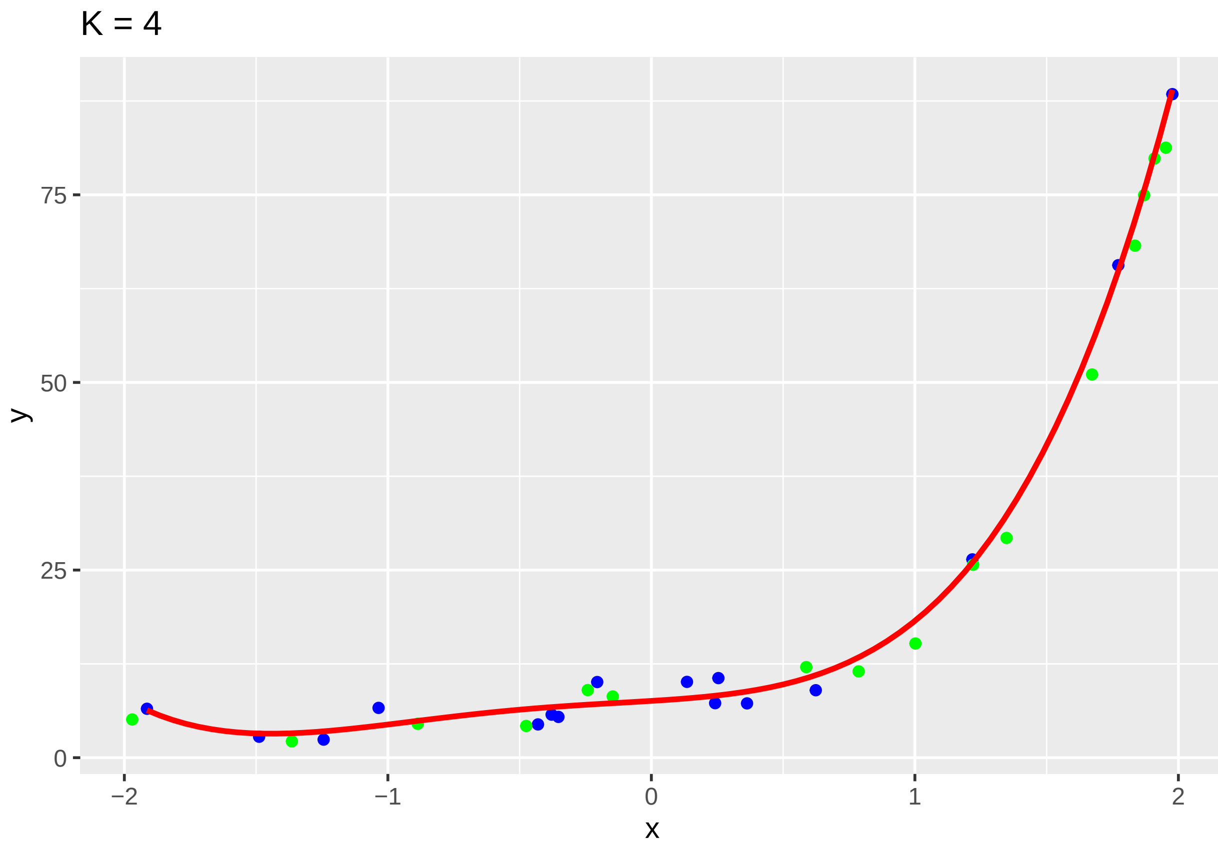

K = 1
75
50
25
0
y
−2
−1
0
1
2
x

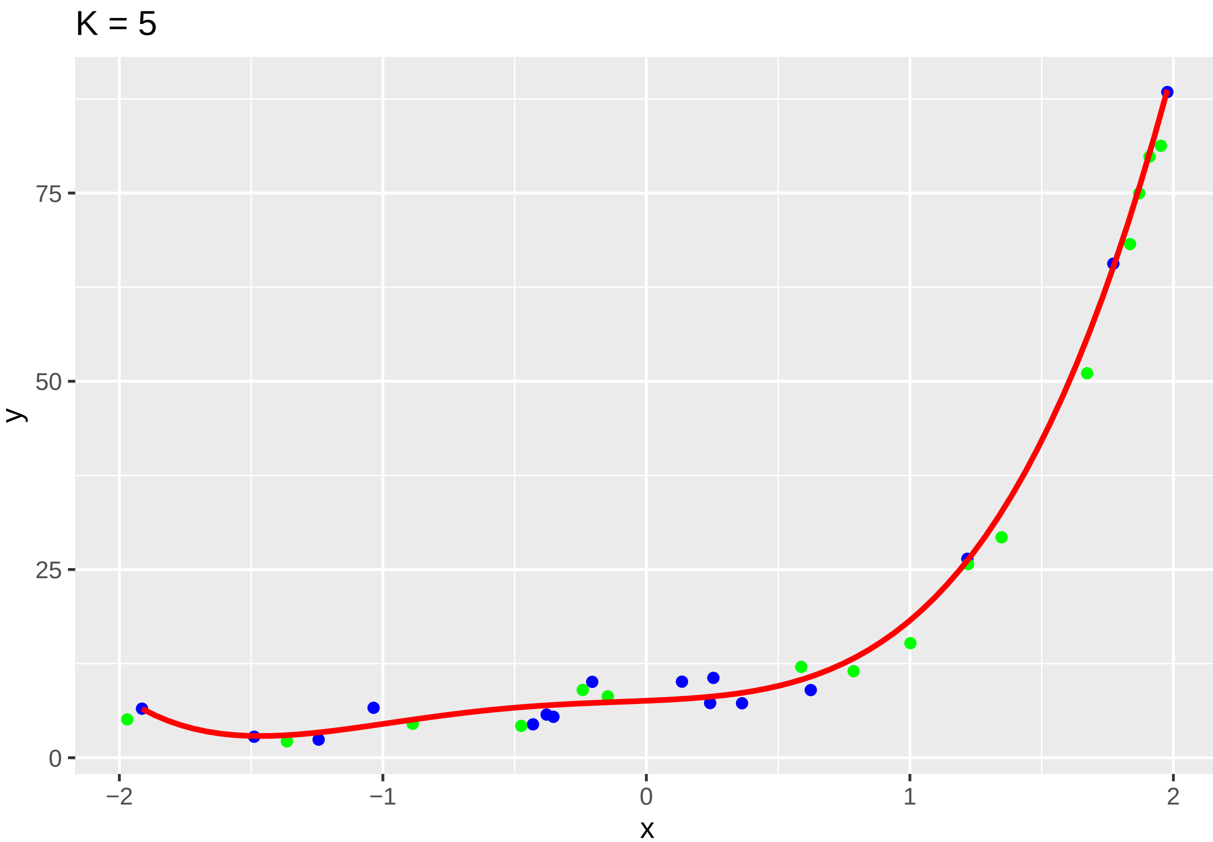


K = 2
y
75
50
25
0
−2
−1
0
1
2
x

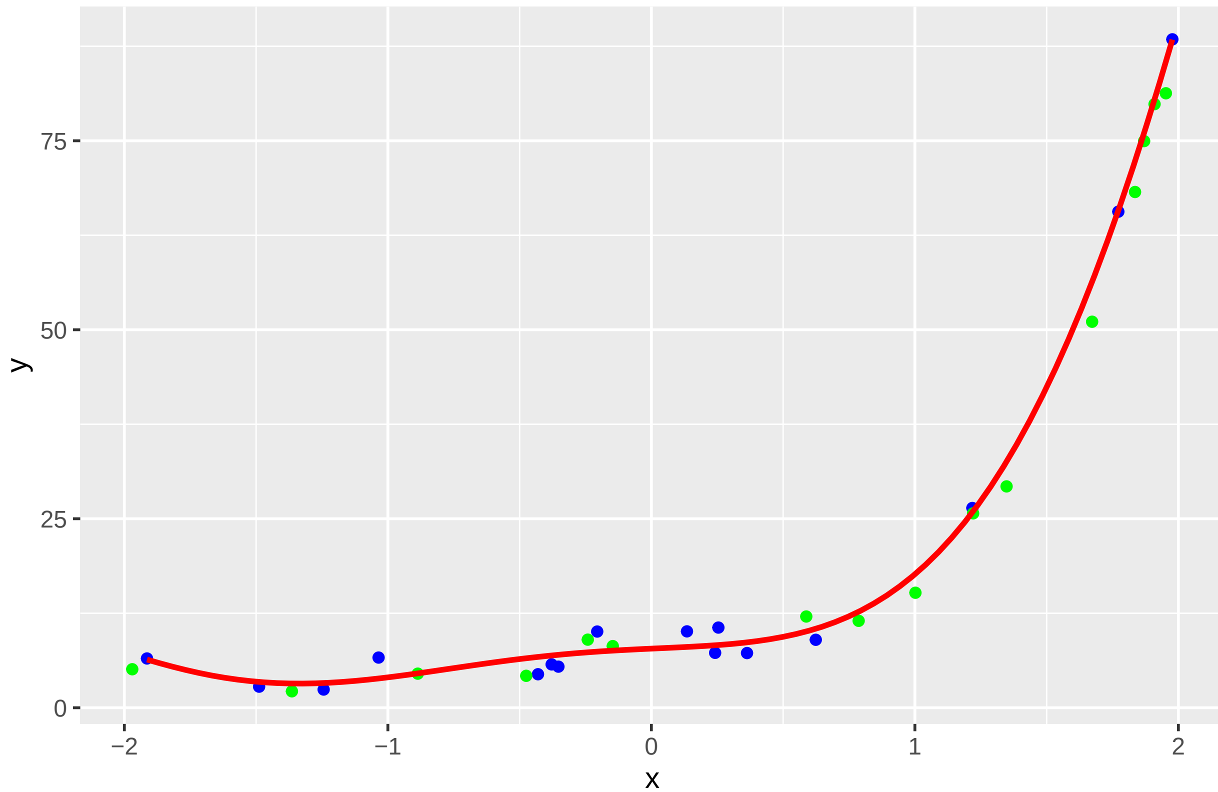


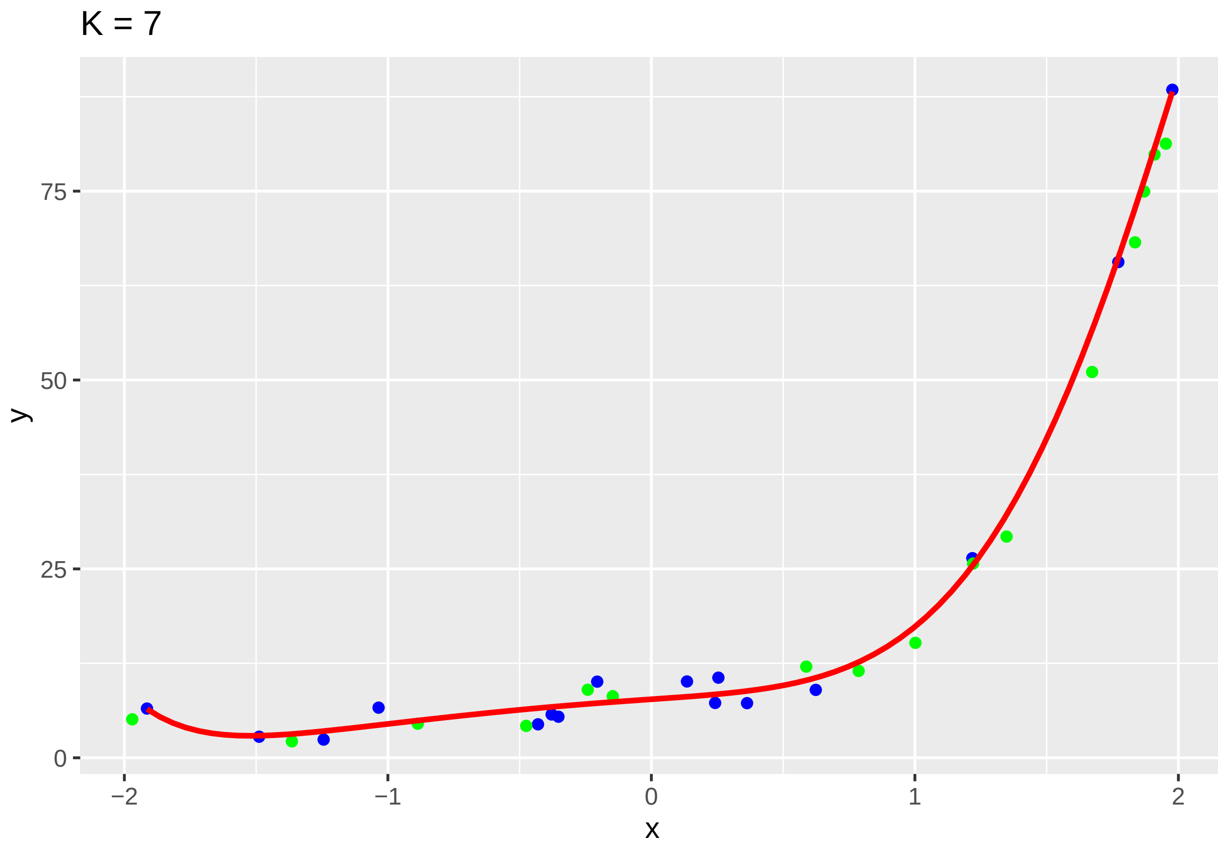
K = 3
75
50
25
0
y
−2
−1
0
1
2
x

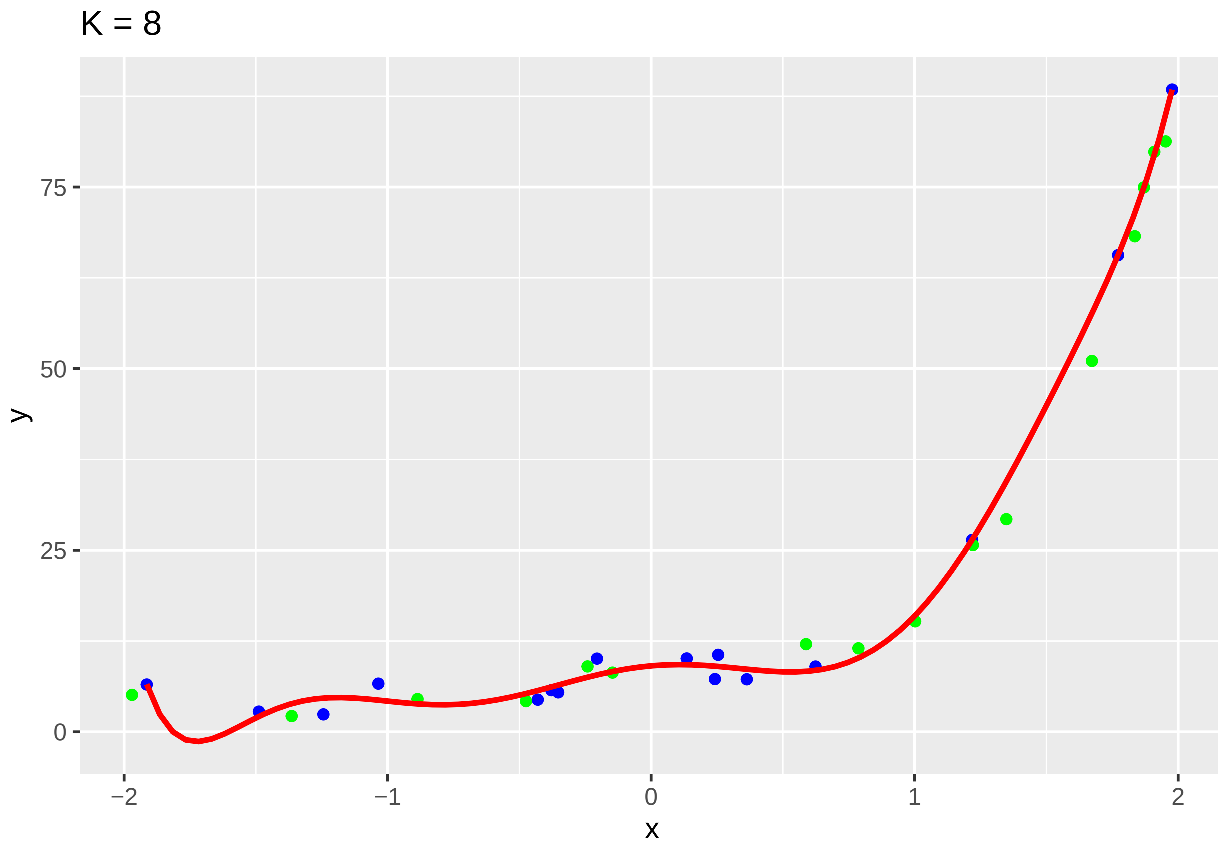

K = 4
y
75
50
25
0
−2
−1
0
1
2
x


K = 5
y
75
50
25
0
−2
−1
0
1
2
x

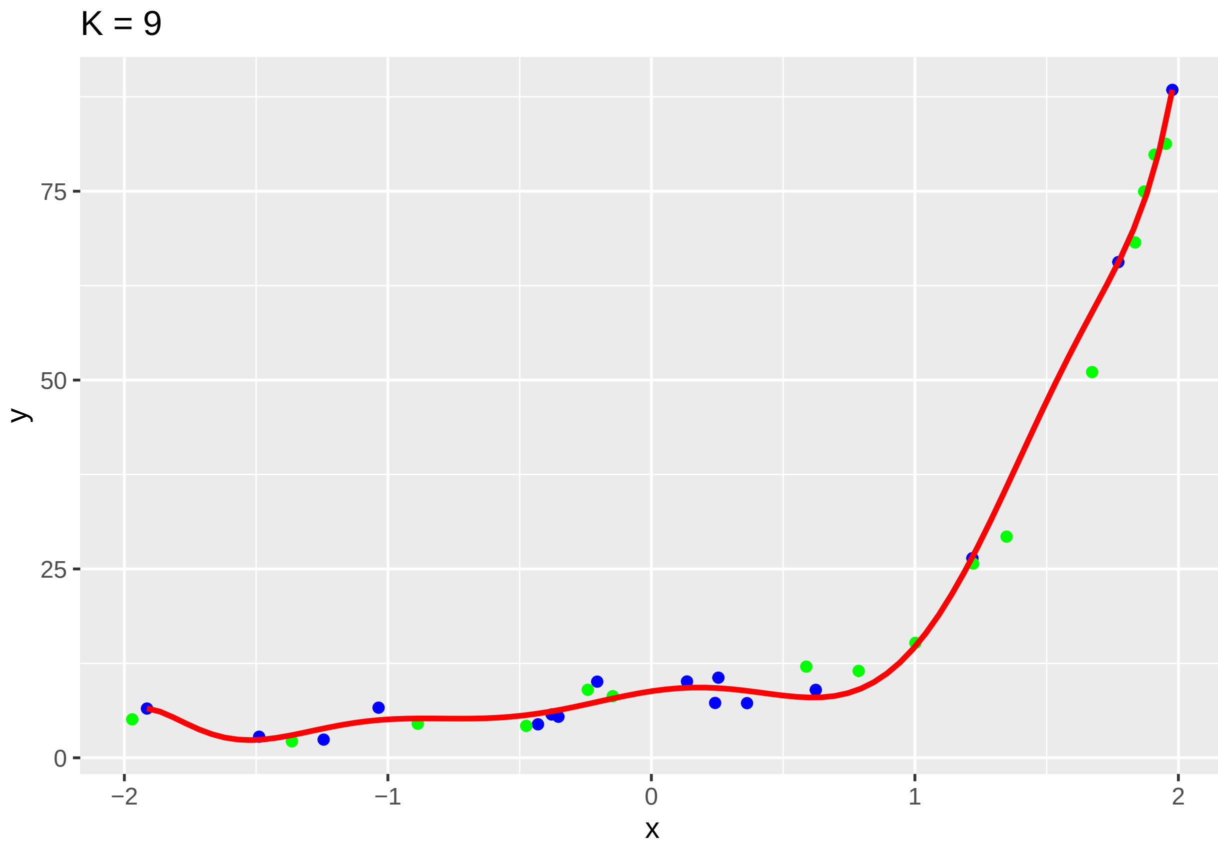
K = 6

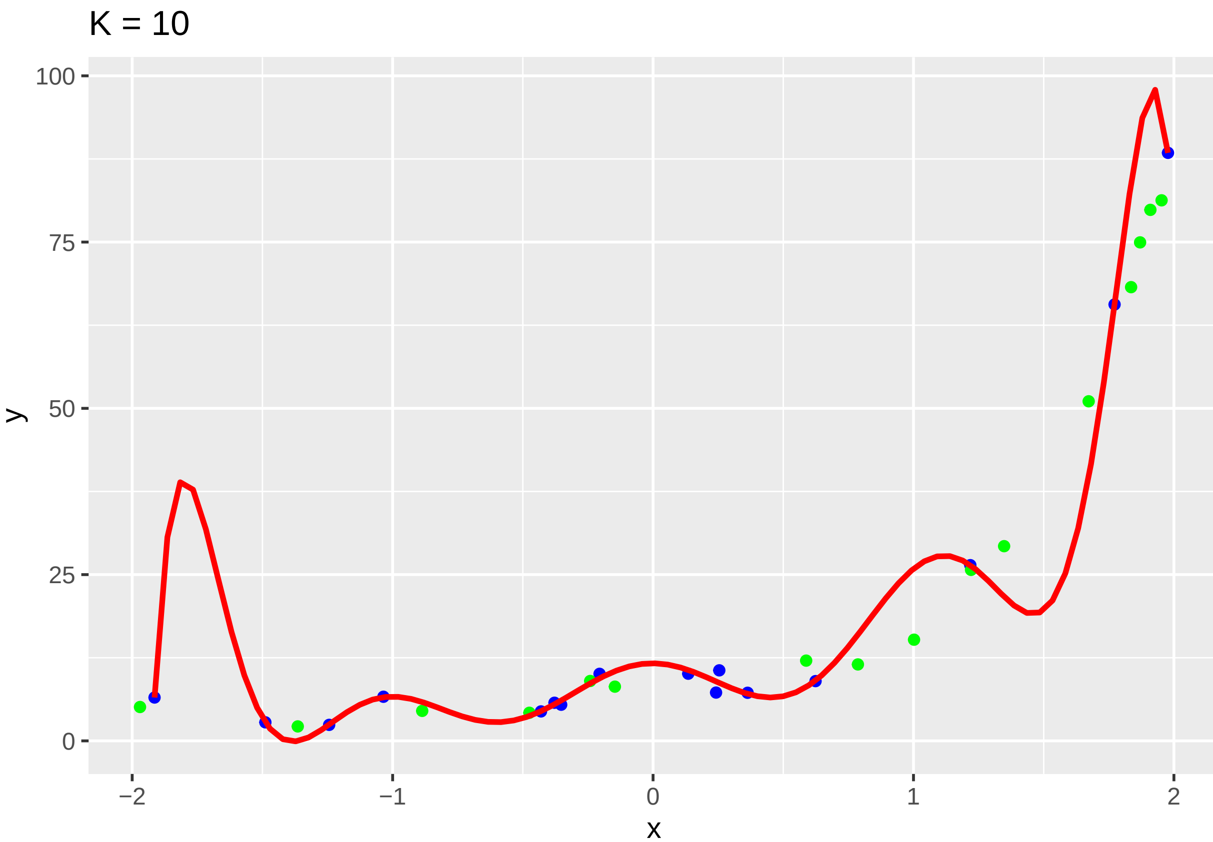



K = 7
75
50
25
0
y
−2
−1
0
1
2
x



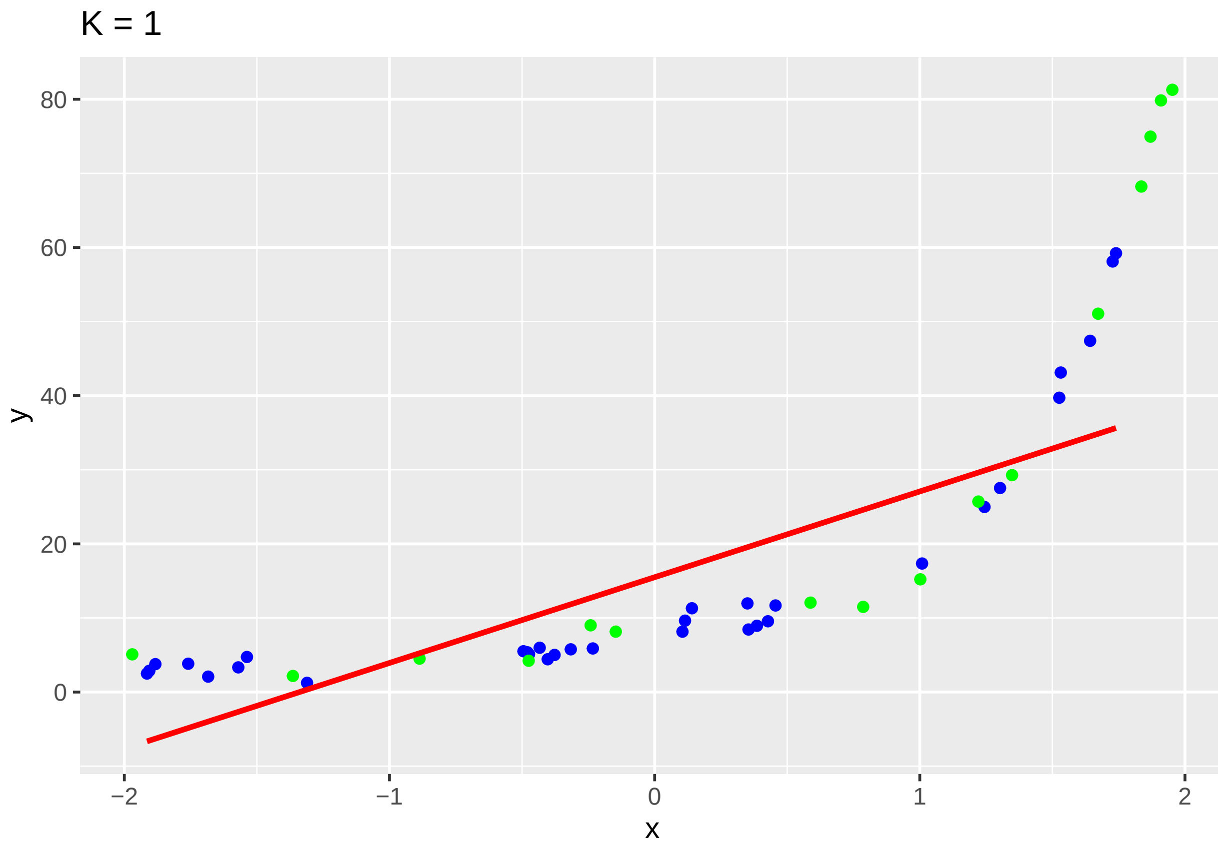
K = 8
y
75
50
25
0
−2
−1
0
1
2
x


K = 9
y
x
0
25
50
75
−2
−1
0
1
2

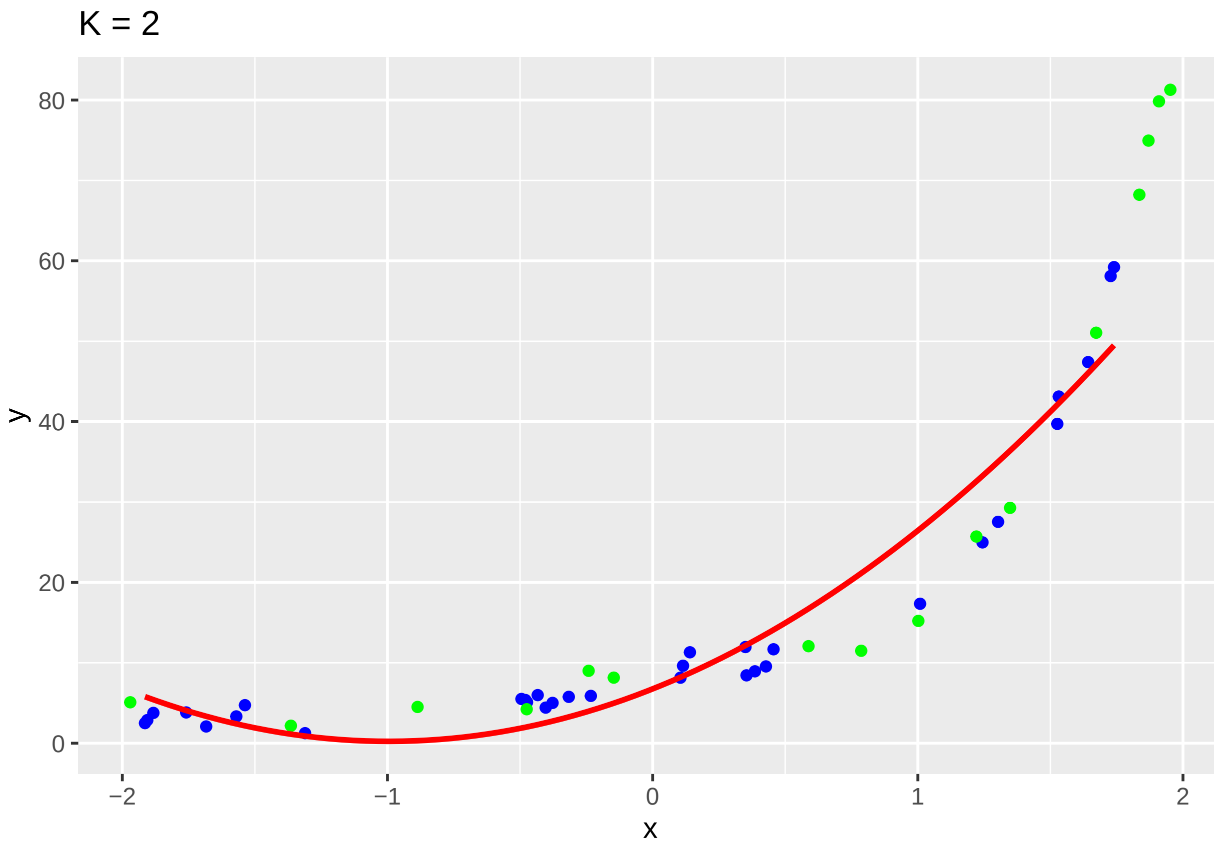


Model fits for K from 1 to 10 for large_train2 dataset:

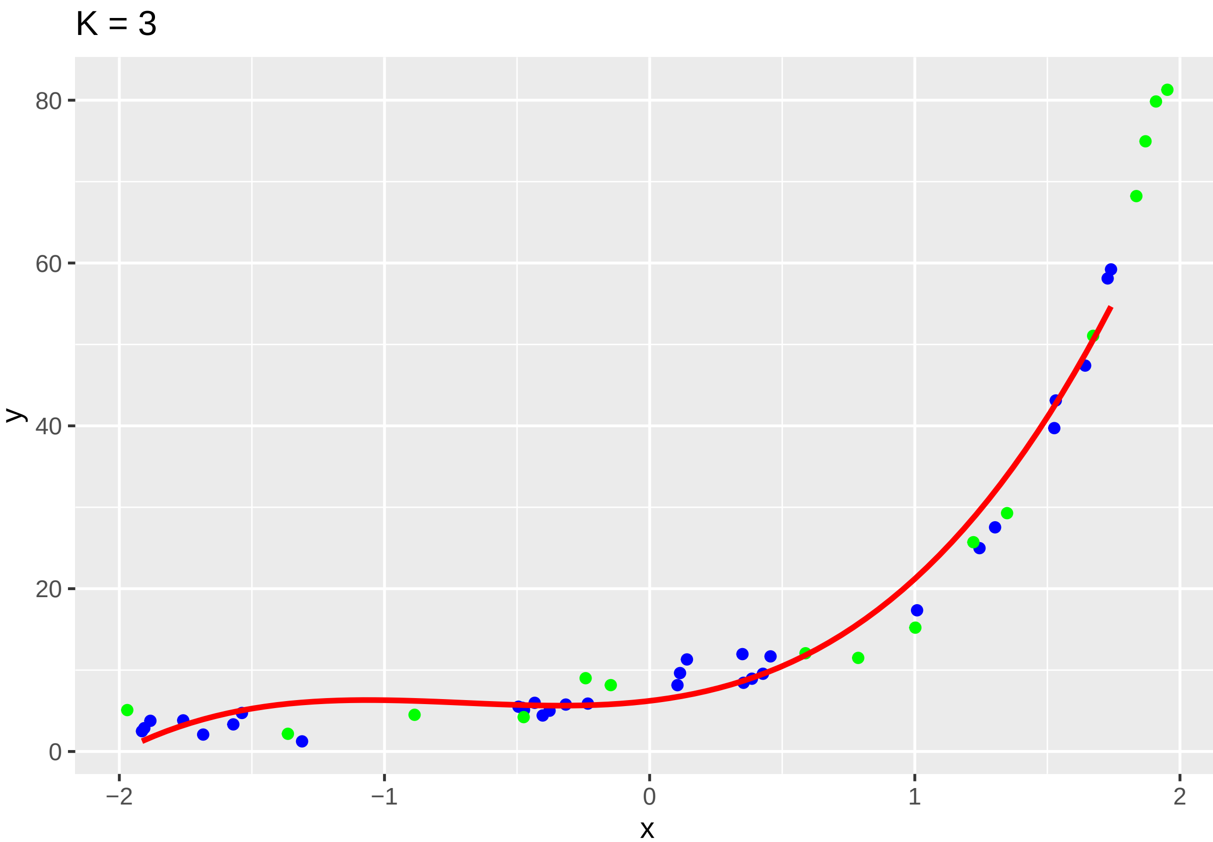
```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=large_train2, color="blue") + geom_point(data=
    ggtitle(paste("K =", k))
  print(g)
}
```

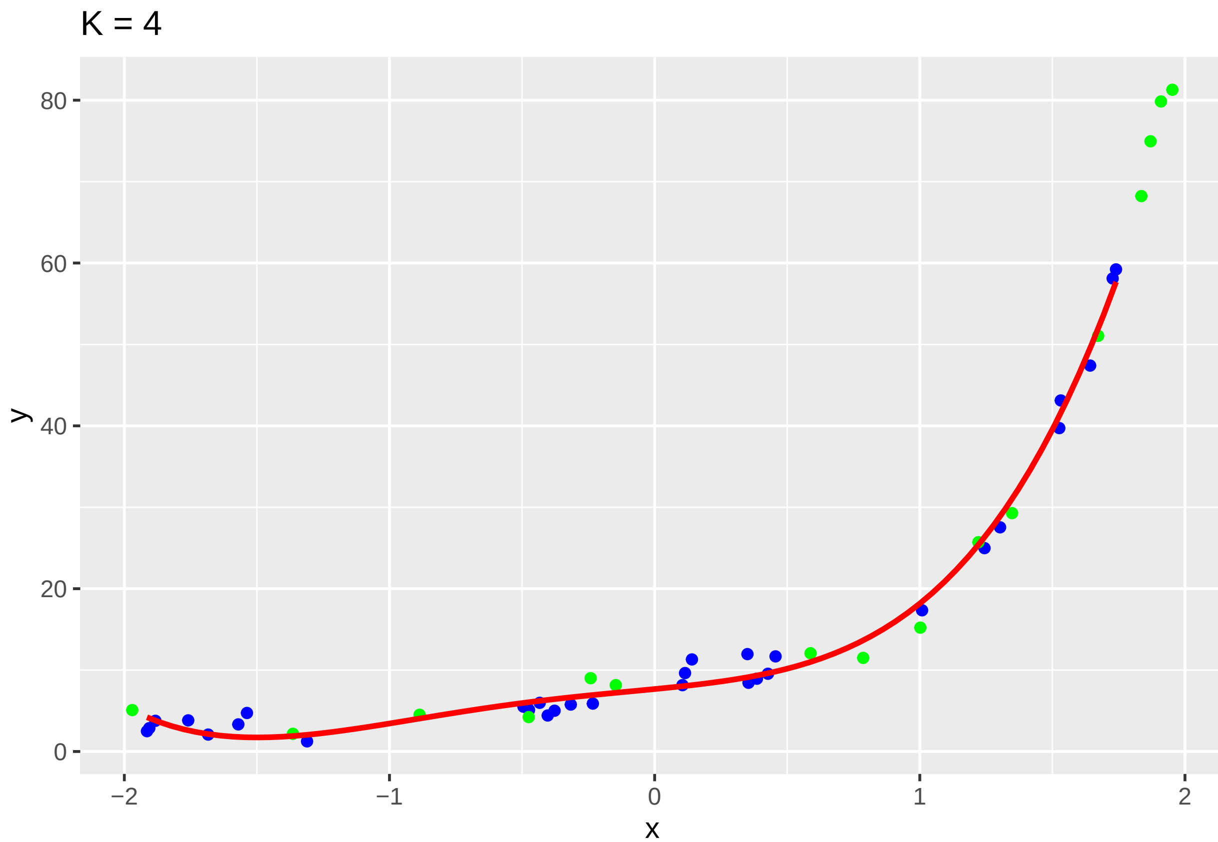


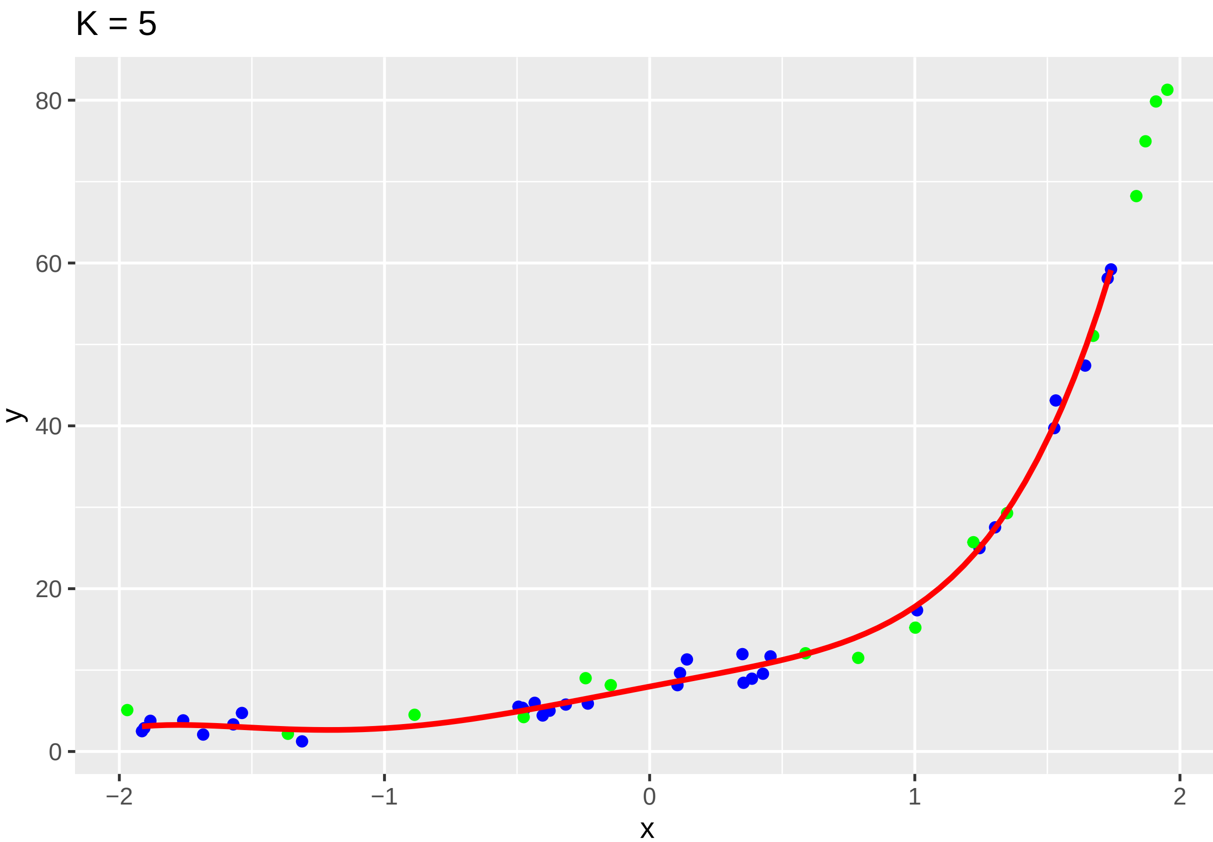
K = 1
80
60
40
20
0
y
−2
−1
0
1
2
x

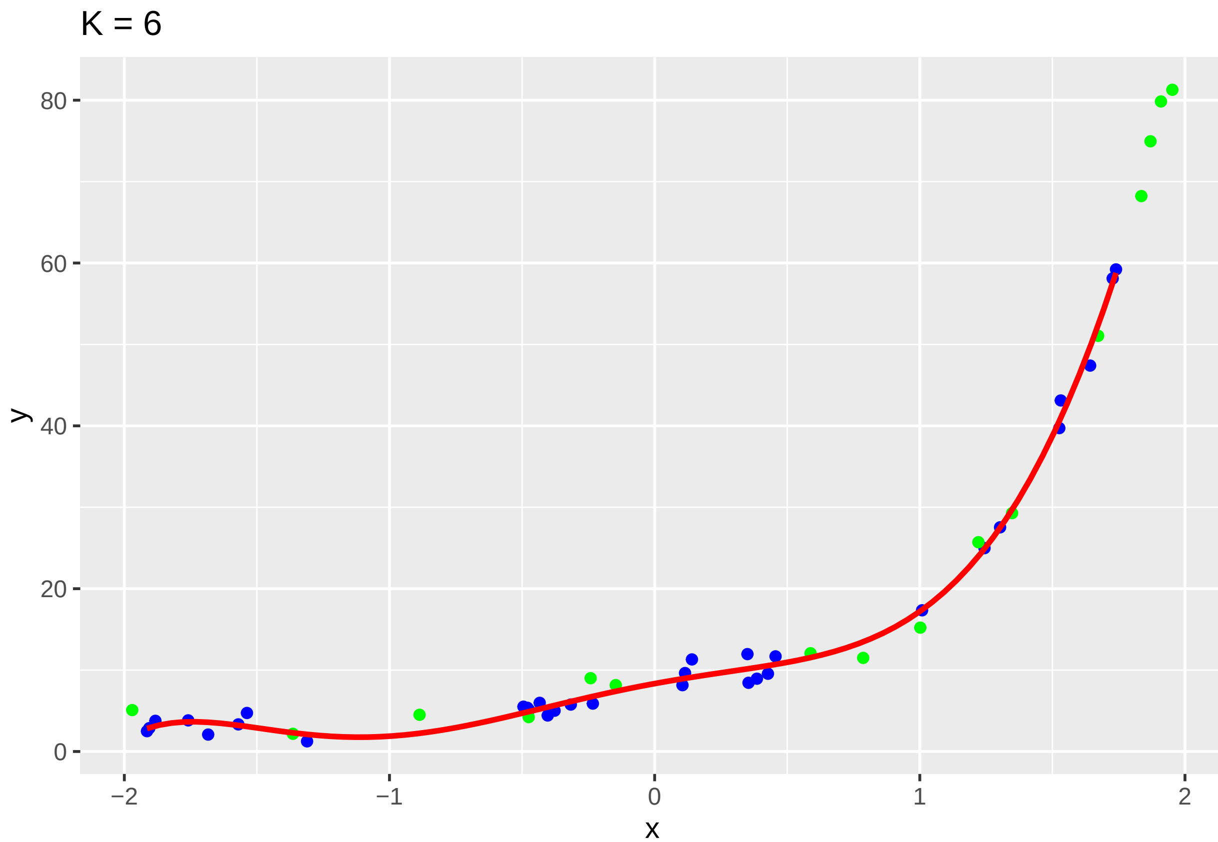


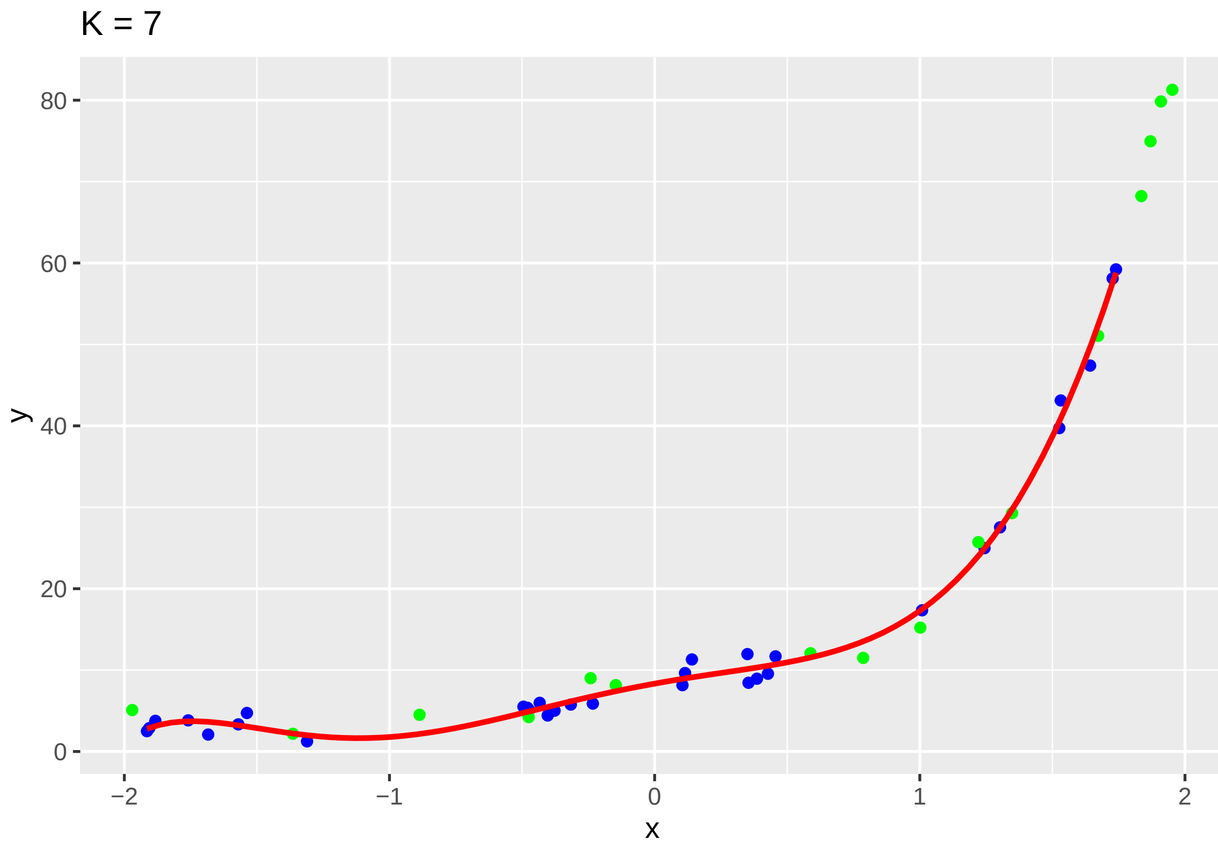
K = 2
80
60
40
20
0
y
−2
−1
0
1
2
x

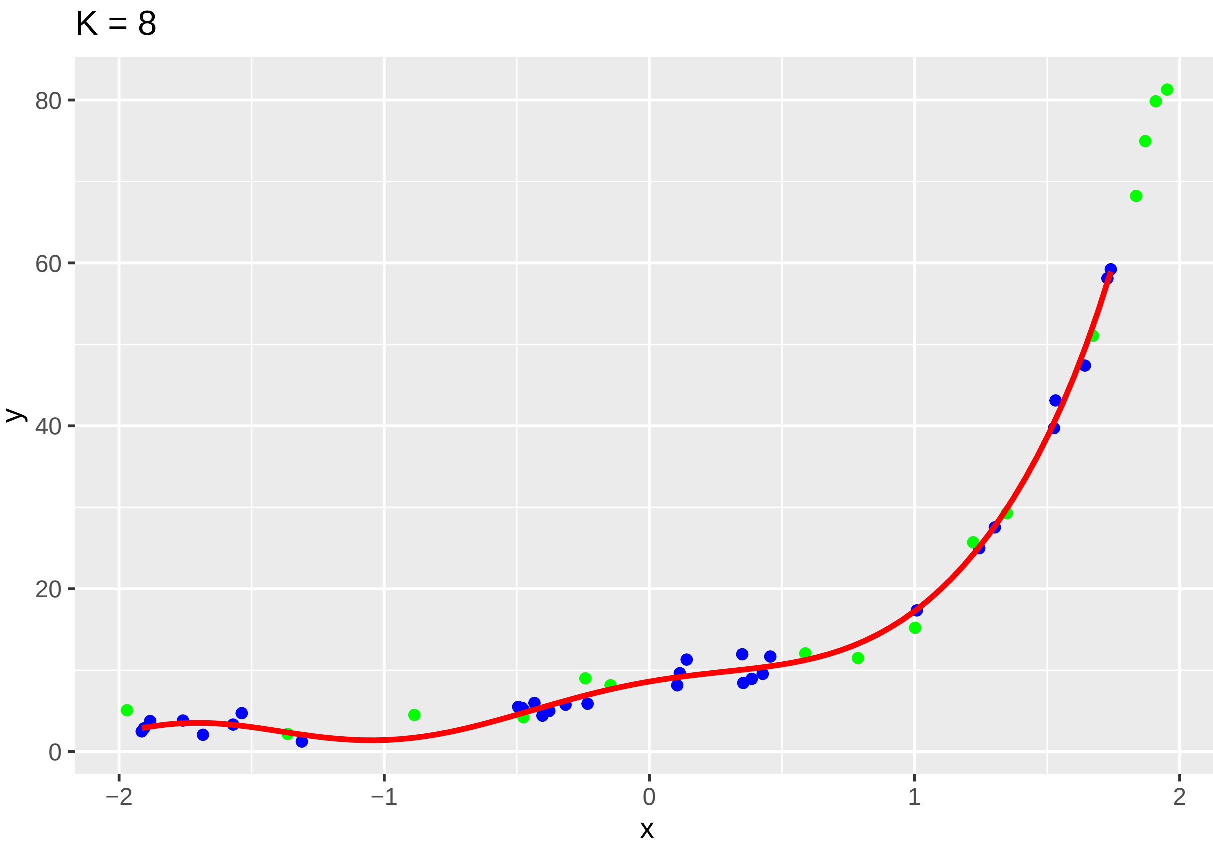

K = 3
80
60
40
20
0
y
−2
−1
0
1
2
x

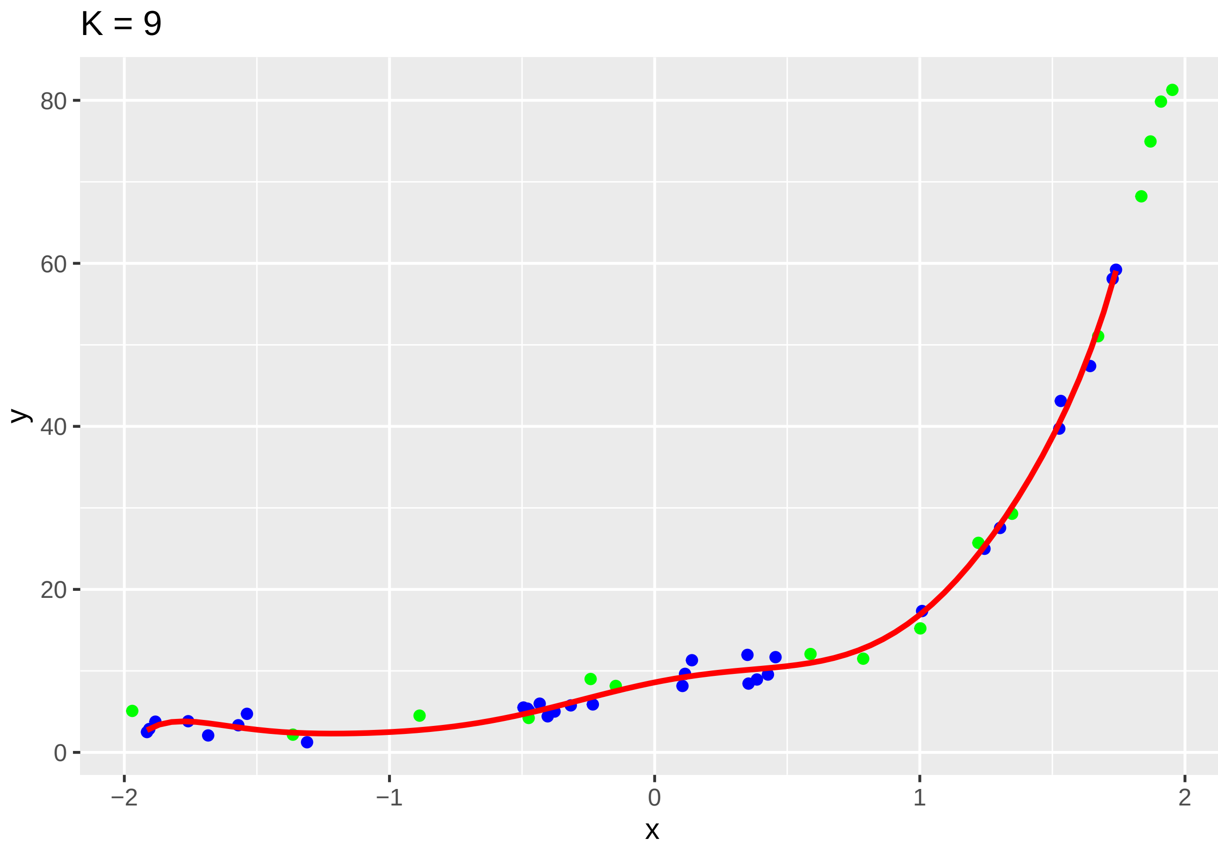

K = 4
y
80
60
40
20
0
−2
−1
0
1
2
x

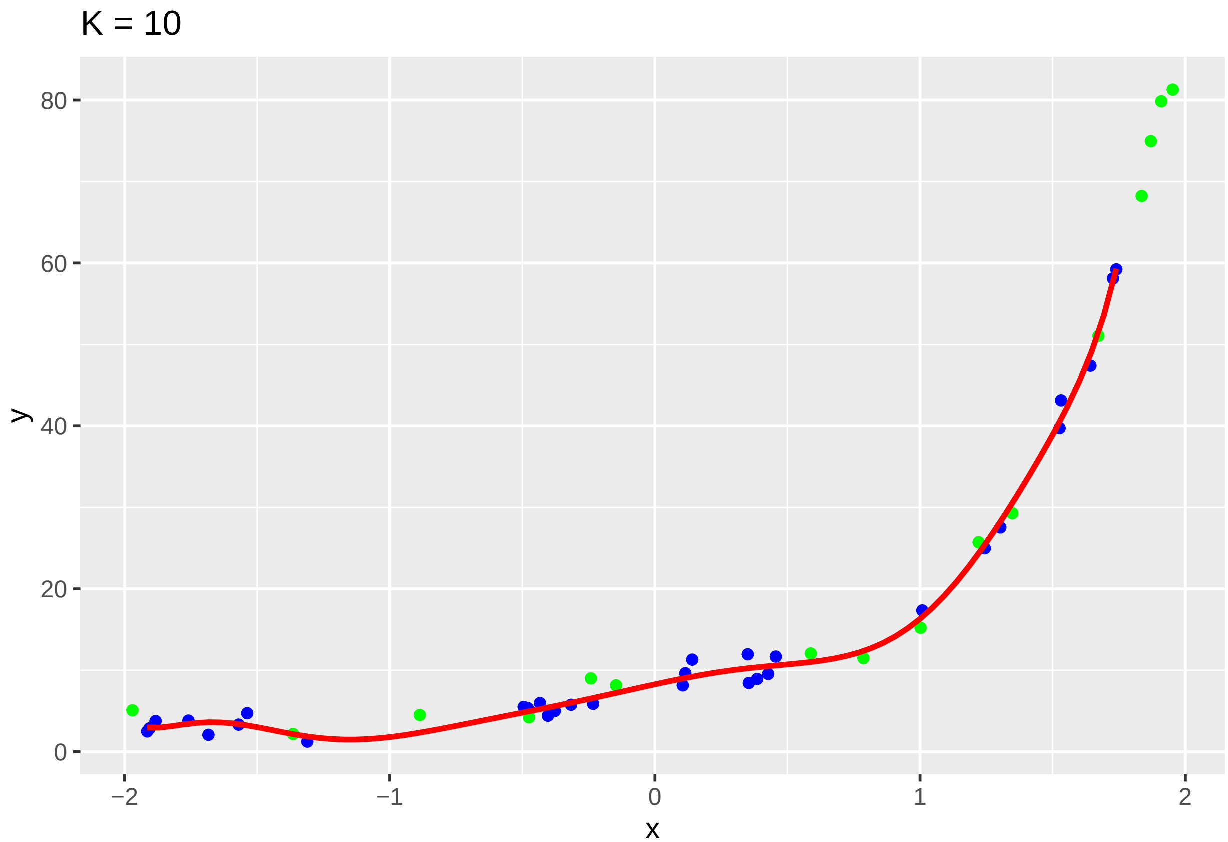

K = 5
y
80
60
40
20
0
−2
−1
0
1
2
x


K = 6
y
x
80
60
40
20
0
−2
−1
0
1
2


K = 7
80
60
40
20
0
y
−2
−1
0
1
2
x

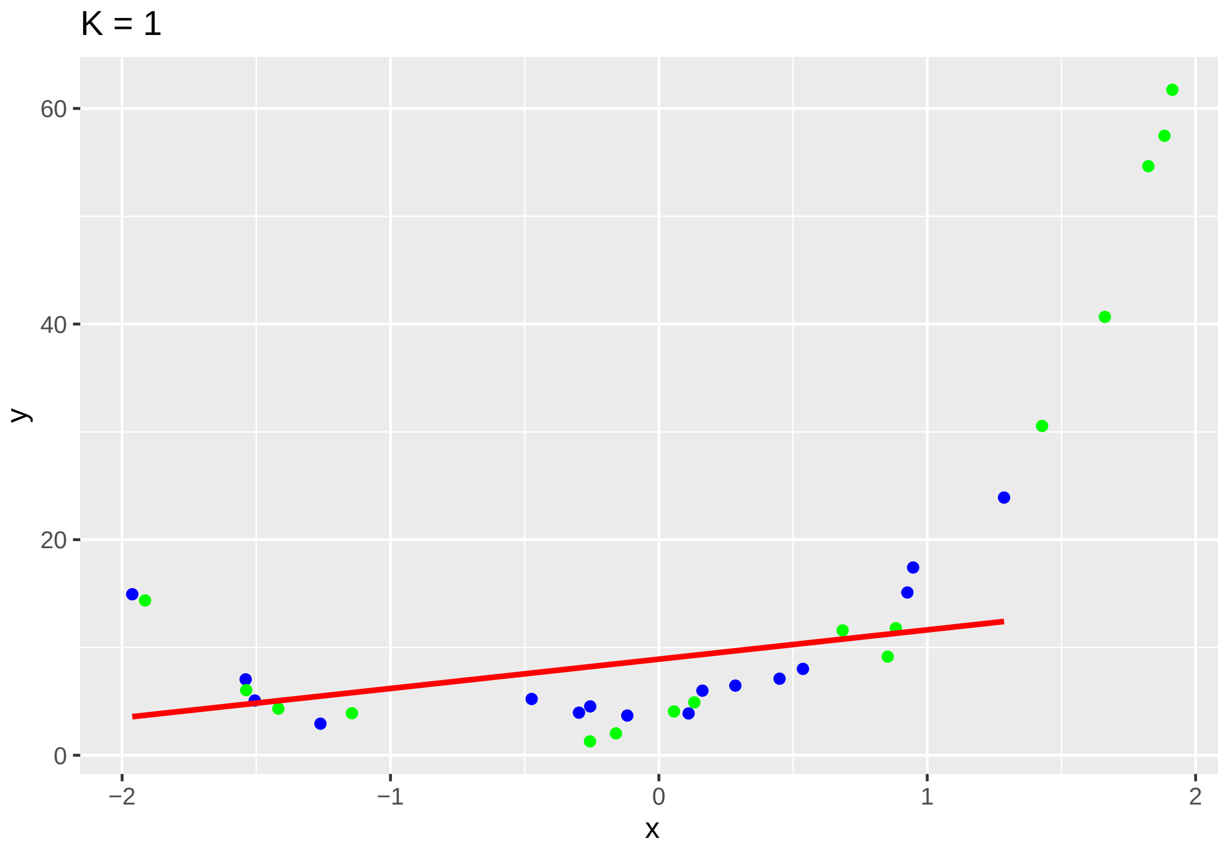

K = 8
80
60
40
20
0
y
−2
−1
0
1
2
x


K = 9
y
80
60
40
20
0
−2
−1
0
1
2
x

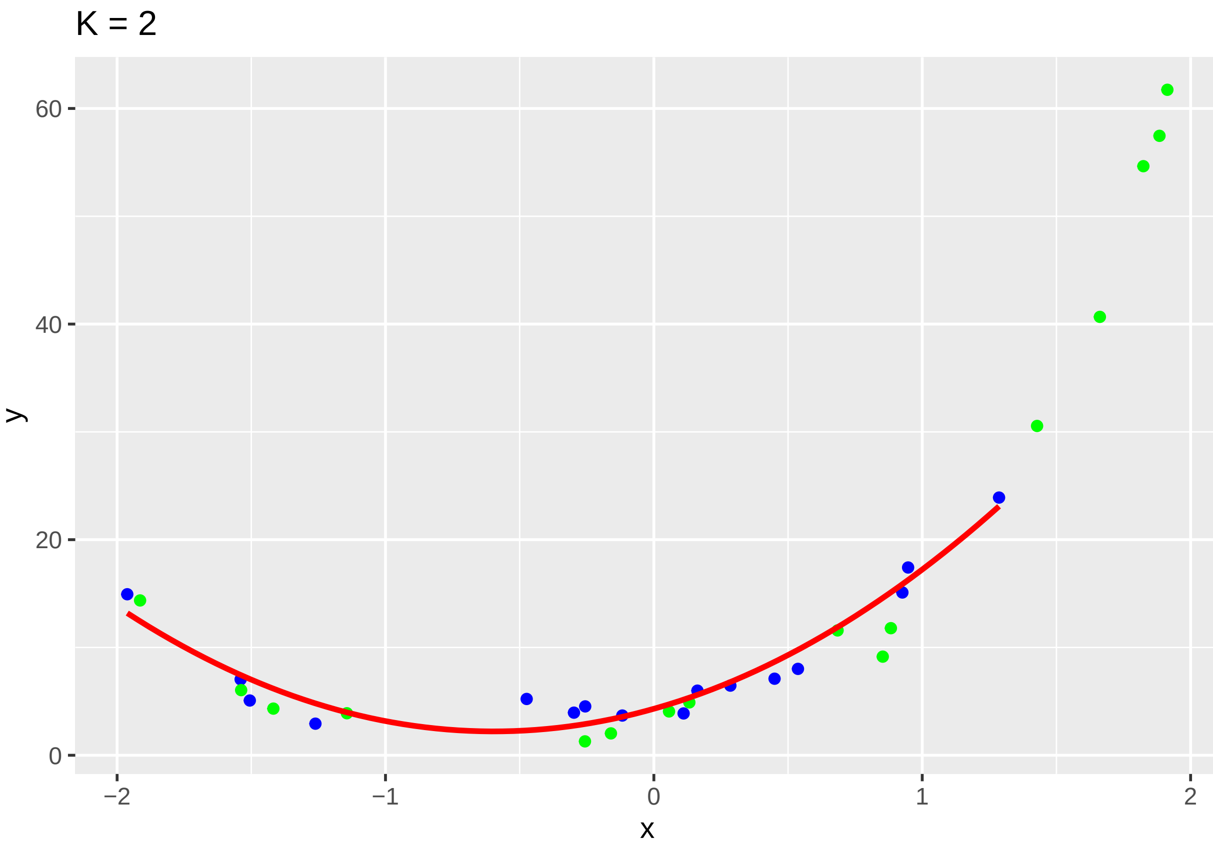


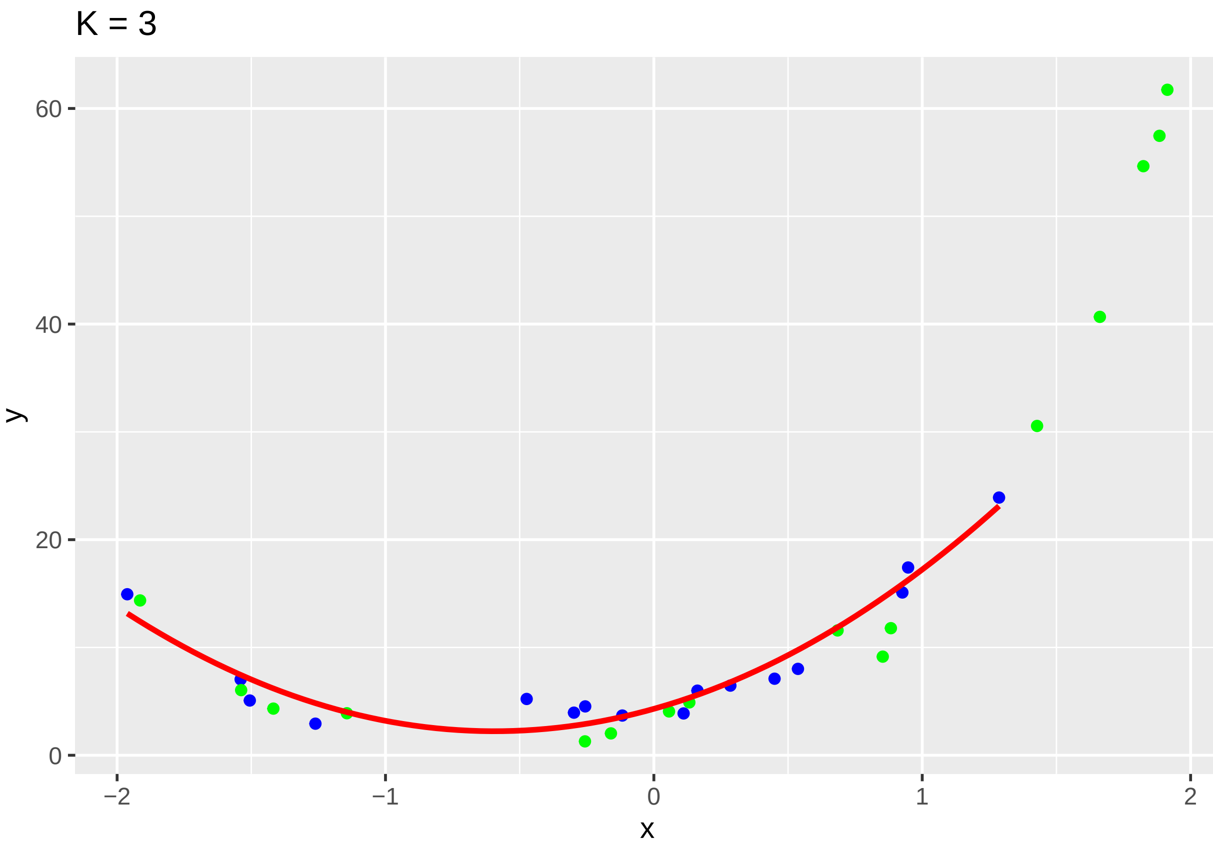
Model fits for K from 1 to 10 for small_train3 dataset:

```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=small_train3, color="blue") + geom_point(data=
    ggtitle(paste("K =", k))
  print(g)
}
```

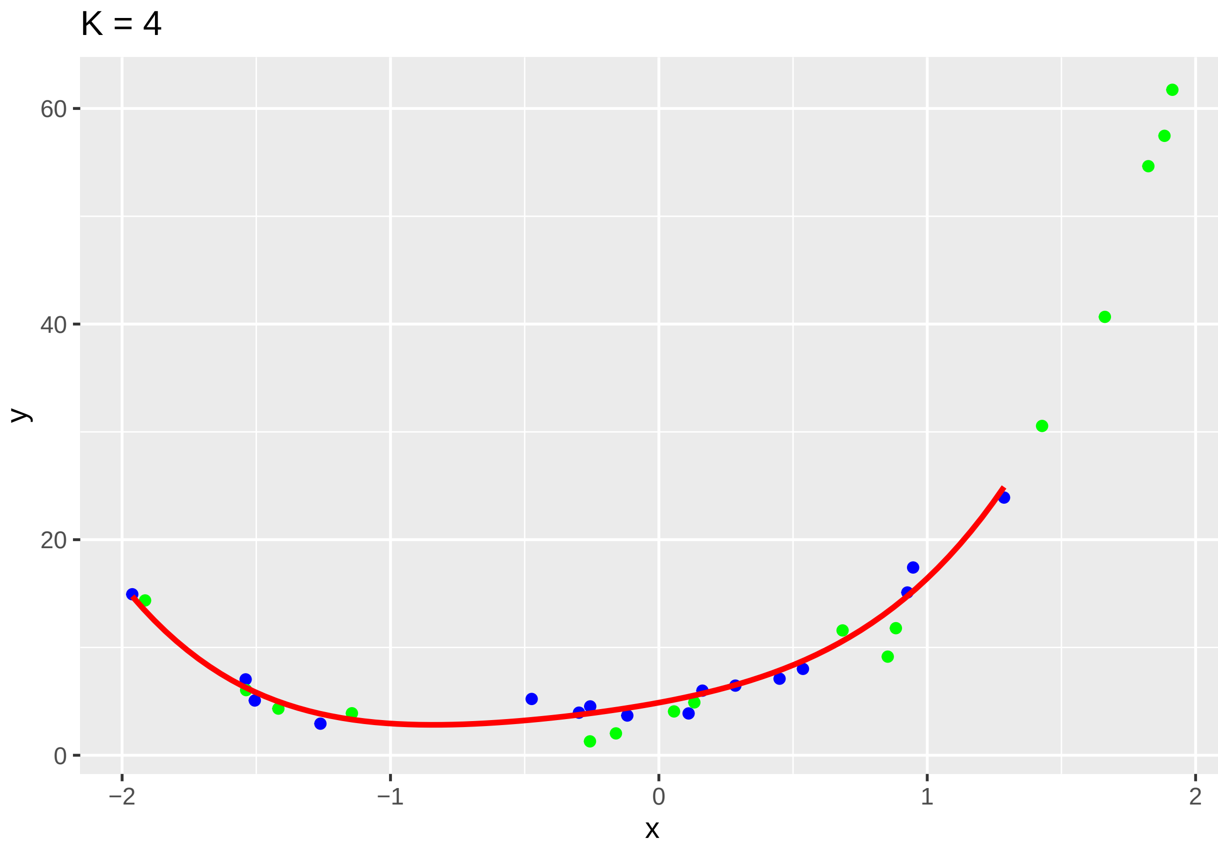


K = 1
60
40
20
0
y
−2
−1
0
1
2
x

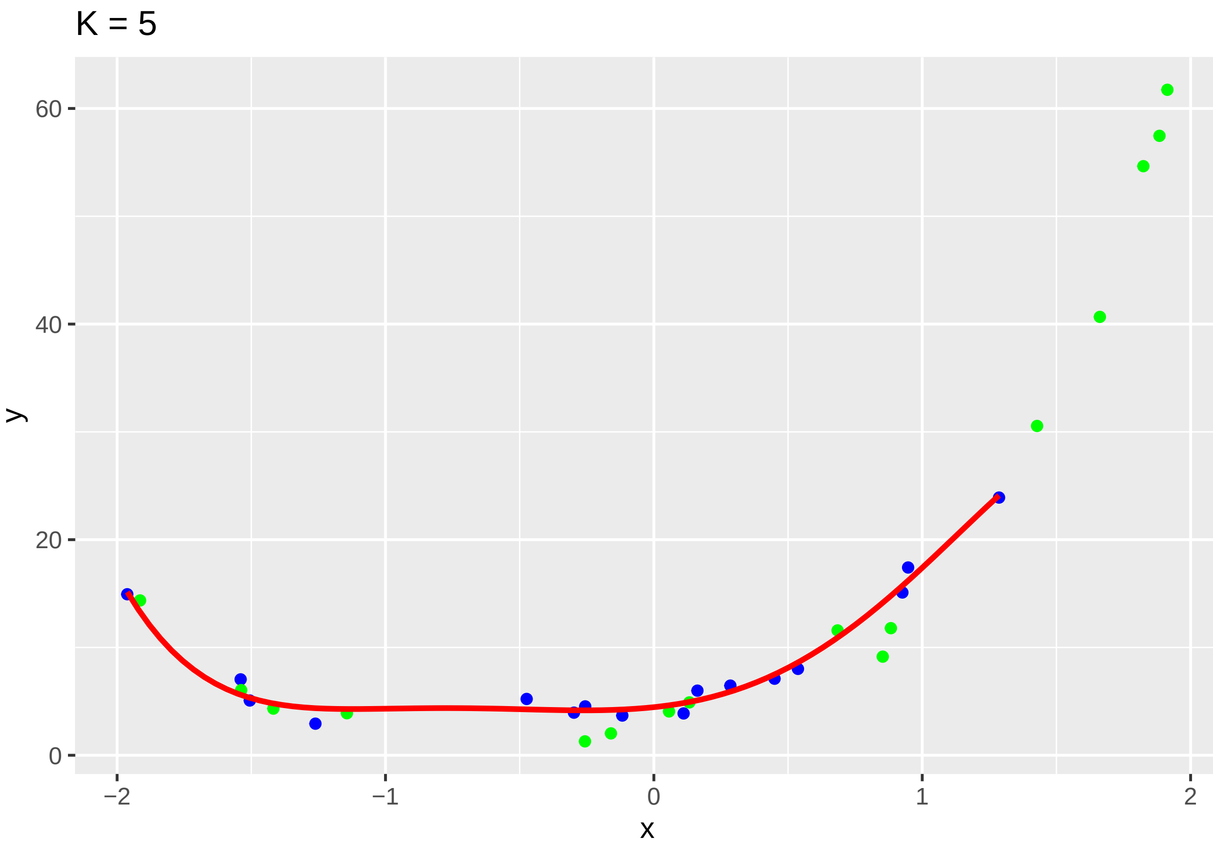

K = 2
60
40
20
0
y
−2
−1
0
1
2
x

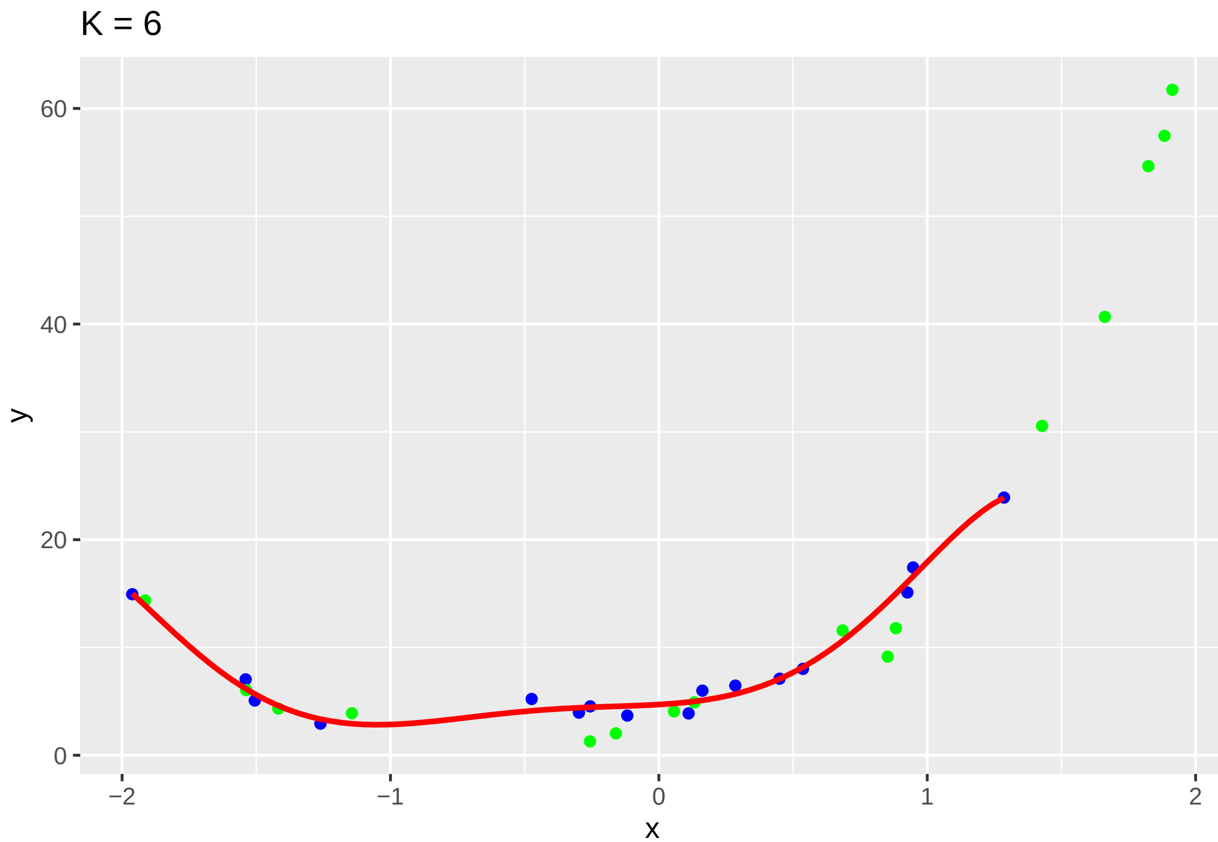


K = 3
60
40
20
0
y
−2
−1
0
1
2
x

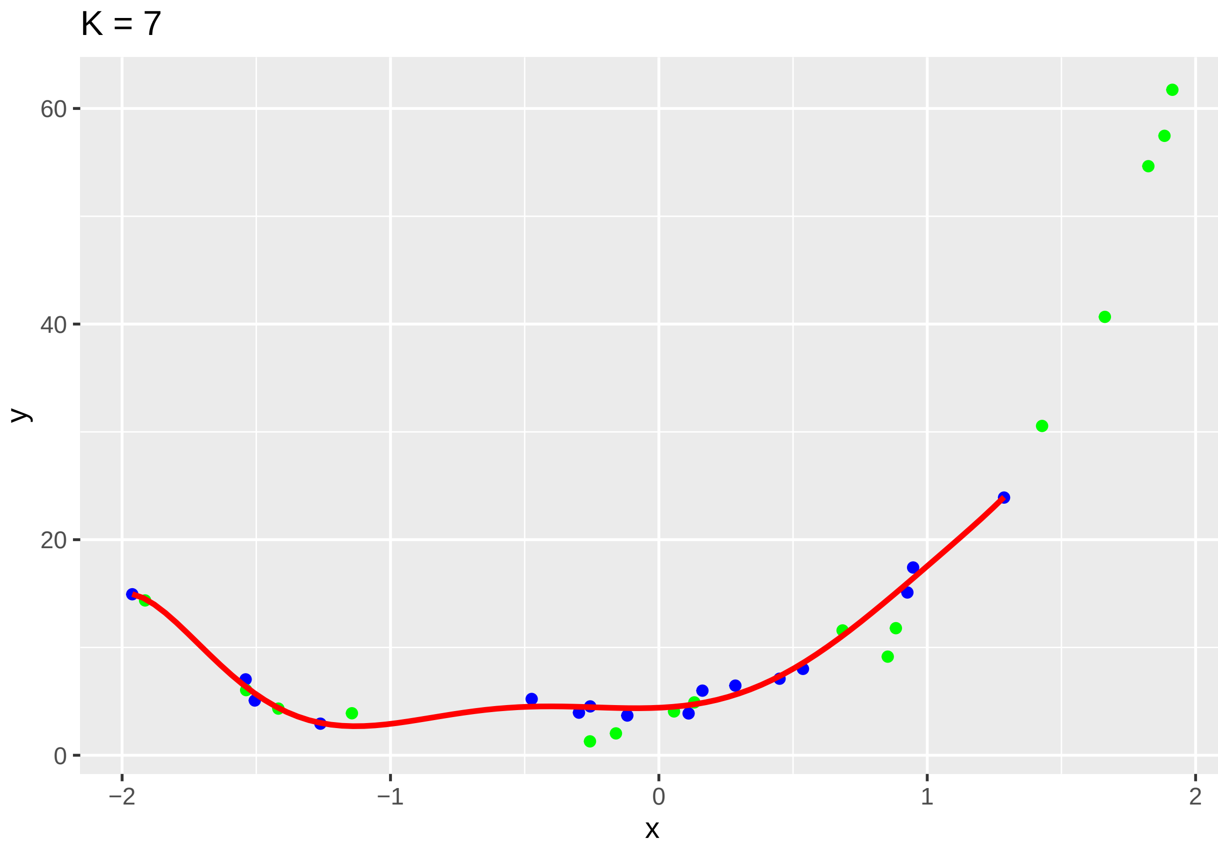


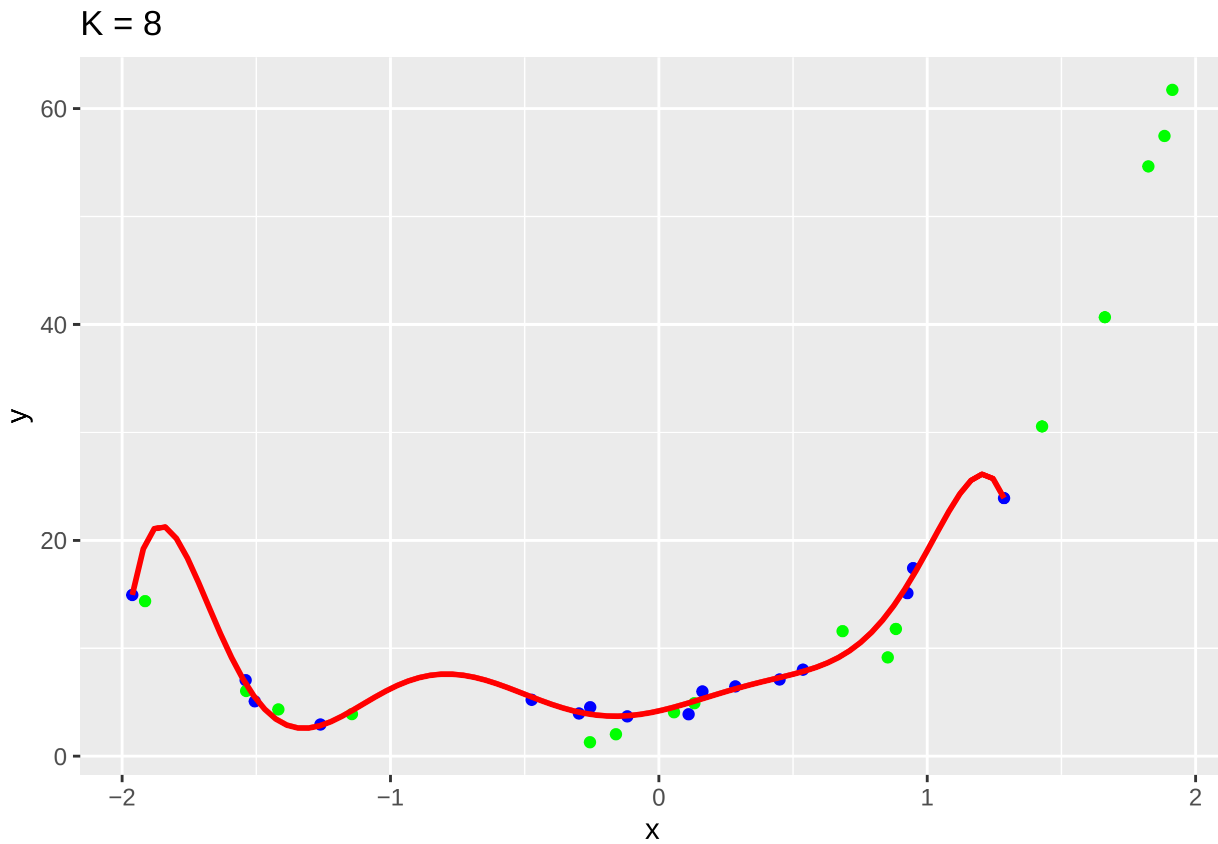
K = 4
60
40
20
0
y
−2
−1
0
1
2
x

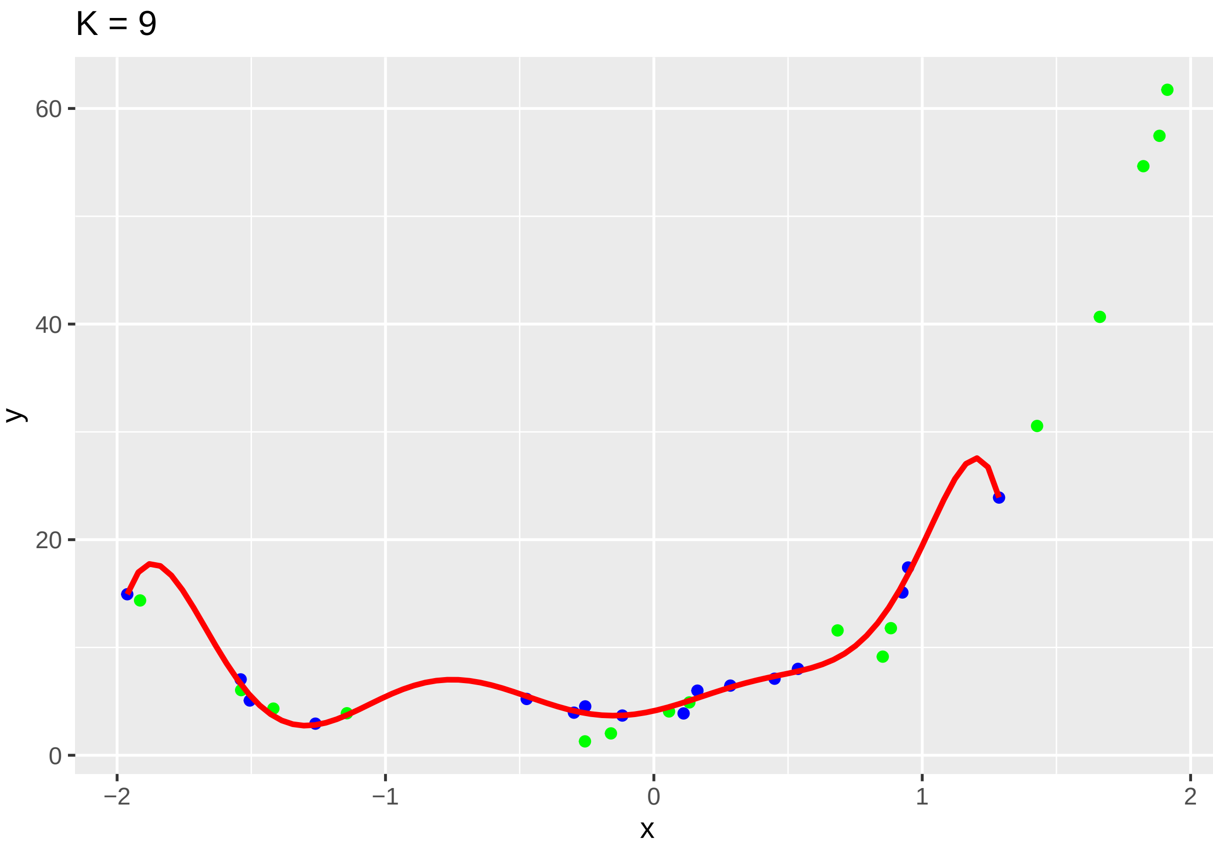

K = 5
60
40
20
0
y
−2
−1
0
1
2
x

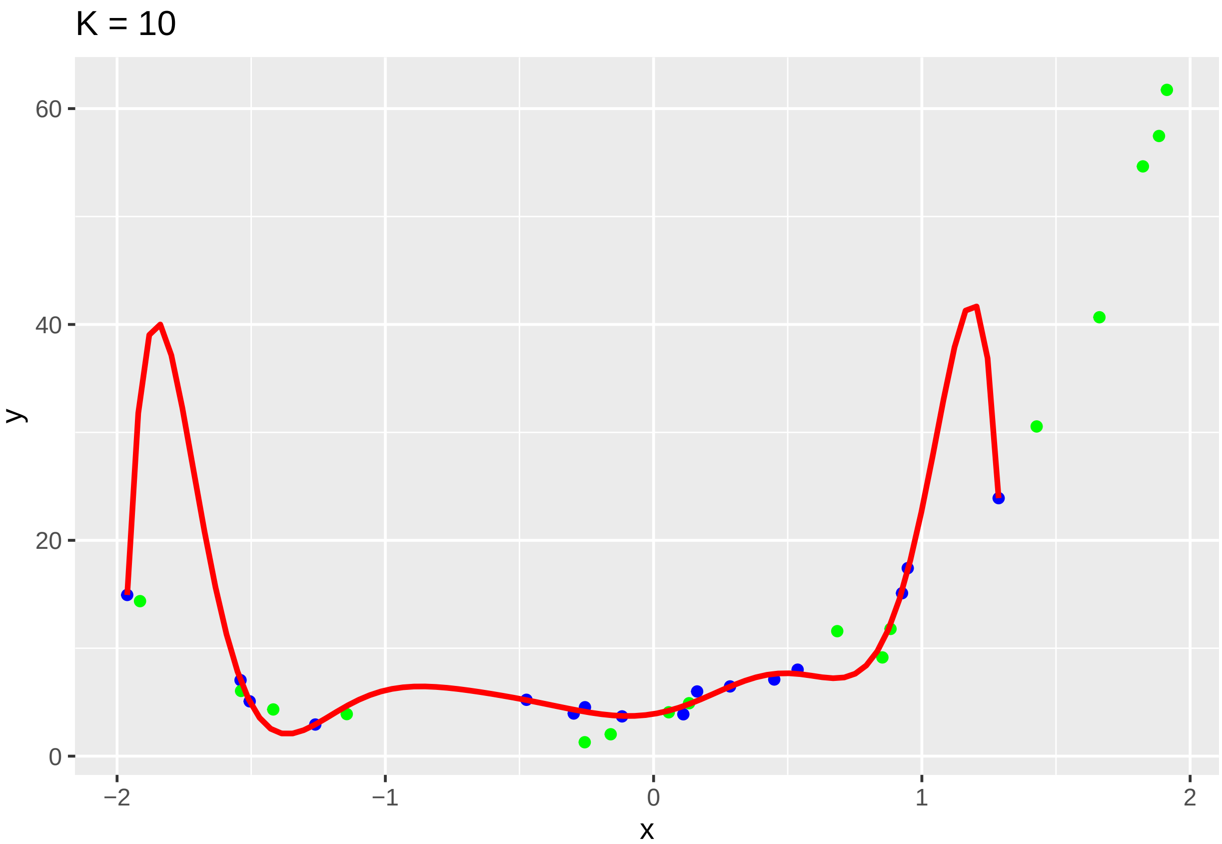


K = 6
60
40
20
0
y
−2
−1
0
1
2
x


K = 7
60
40
20
0
y
−2
−1
0
1
2
x

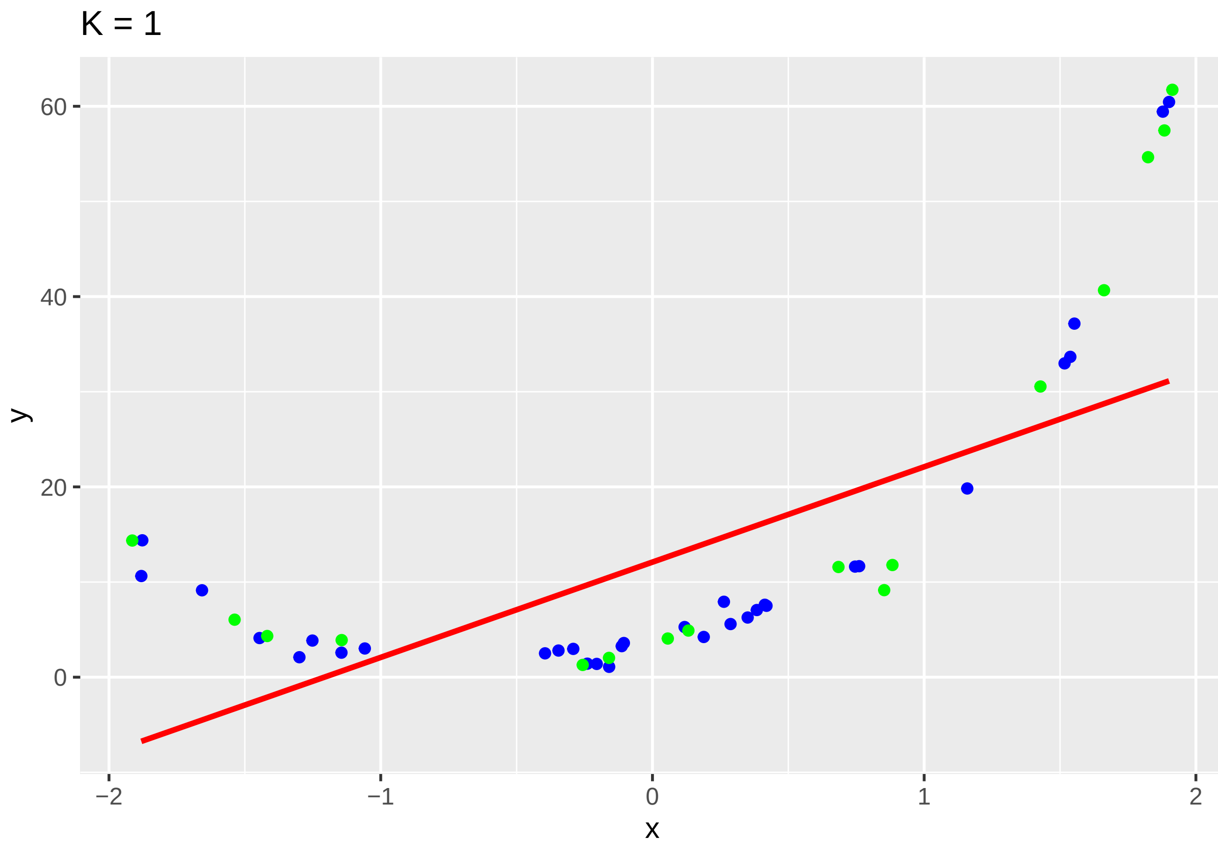

K = 8
60
40
20
0
y
−2
−1
0
1
2
x

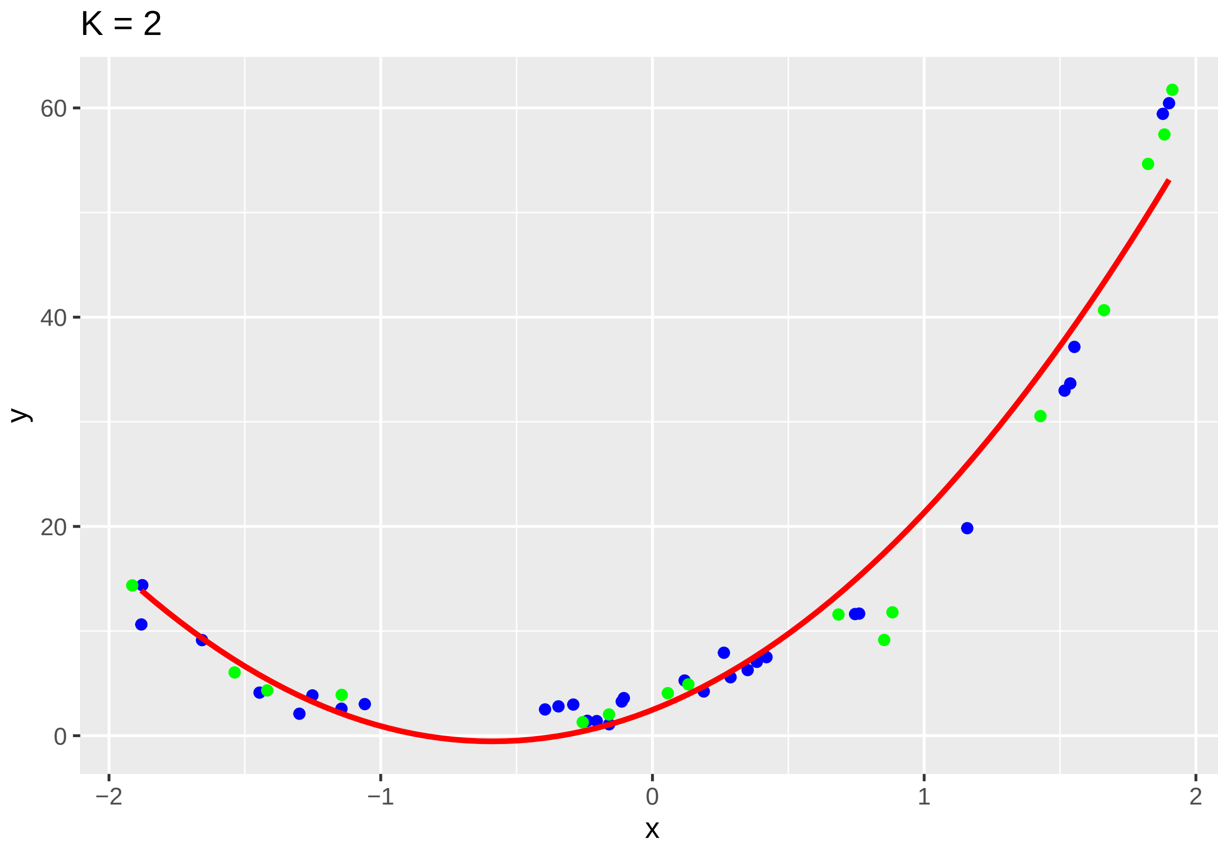

K = 9
60
40
20
0
y
−2
−1
0
1
2
x



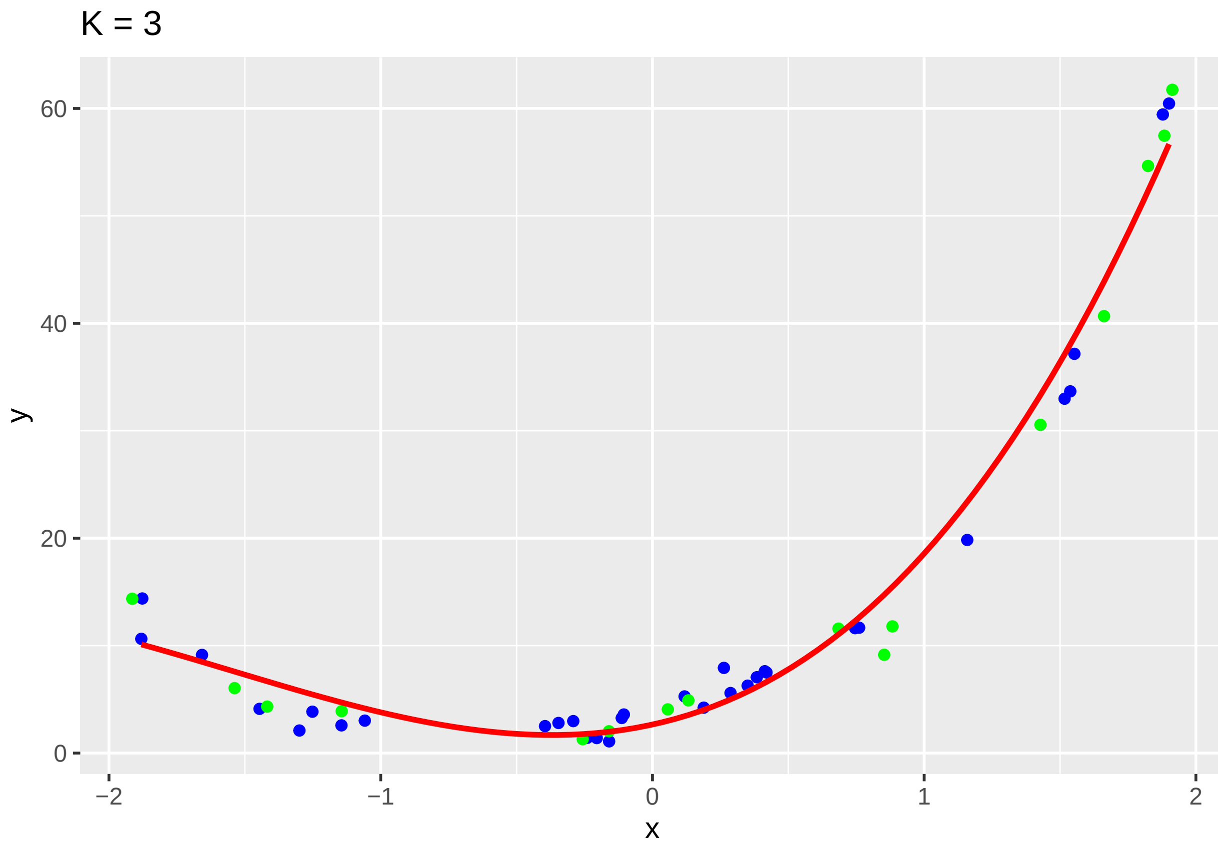
Model fits for K from 1 to 10 for large_train3 dataset:

```
for (k in 1:10) {
  g <- ggplot(data=NULL, aes(x=x, y=y)) + geom_point(data=large_train3, color="blue") + geom_point(data=
    ggtitle(paste("K =", k))
  print(g)
}
```

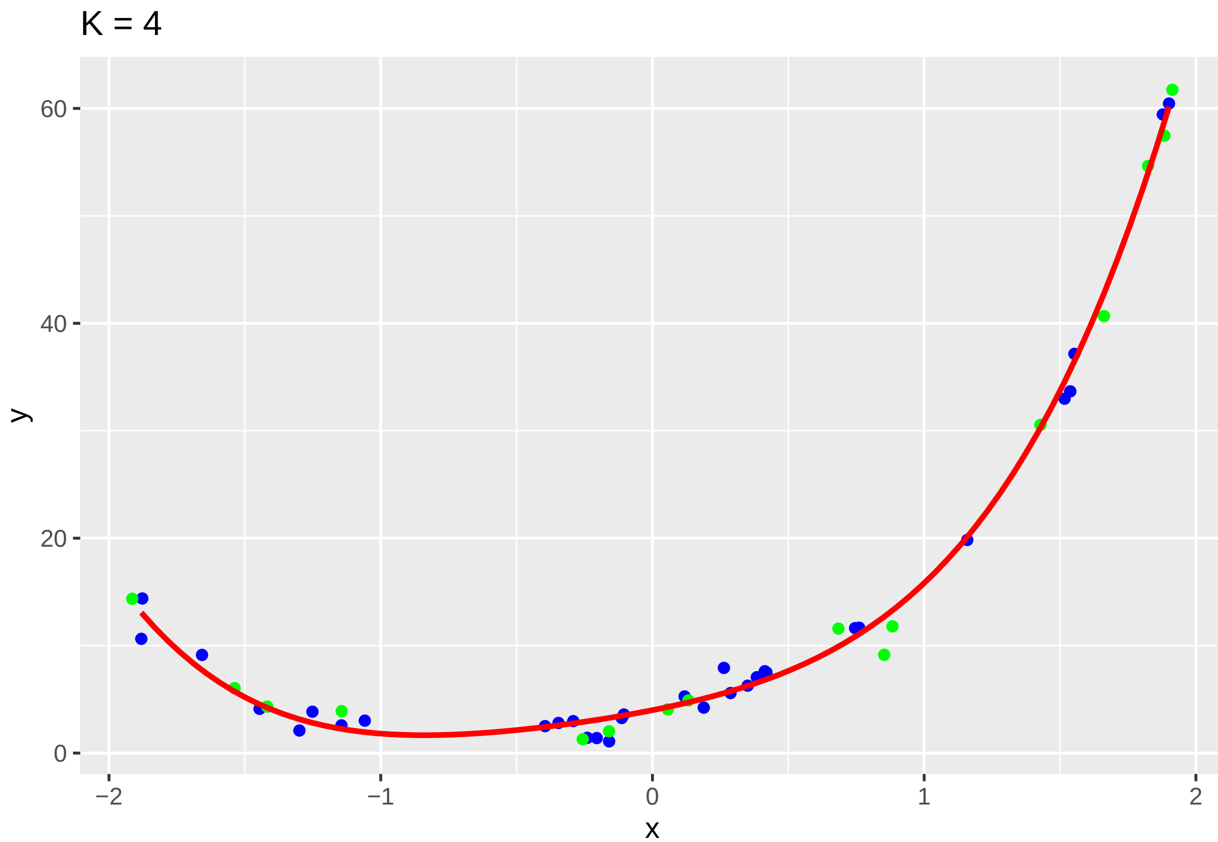

K = 1
60
40
20
0
y
−2
−1
0
1
2
x

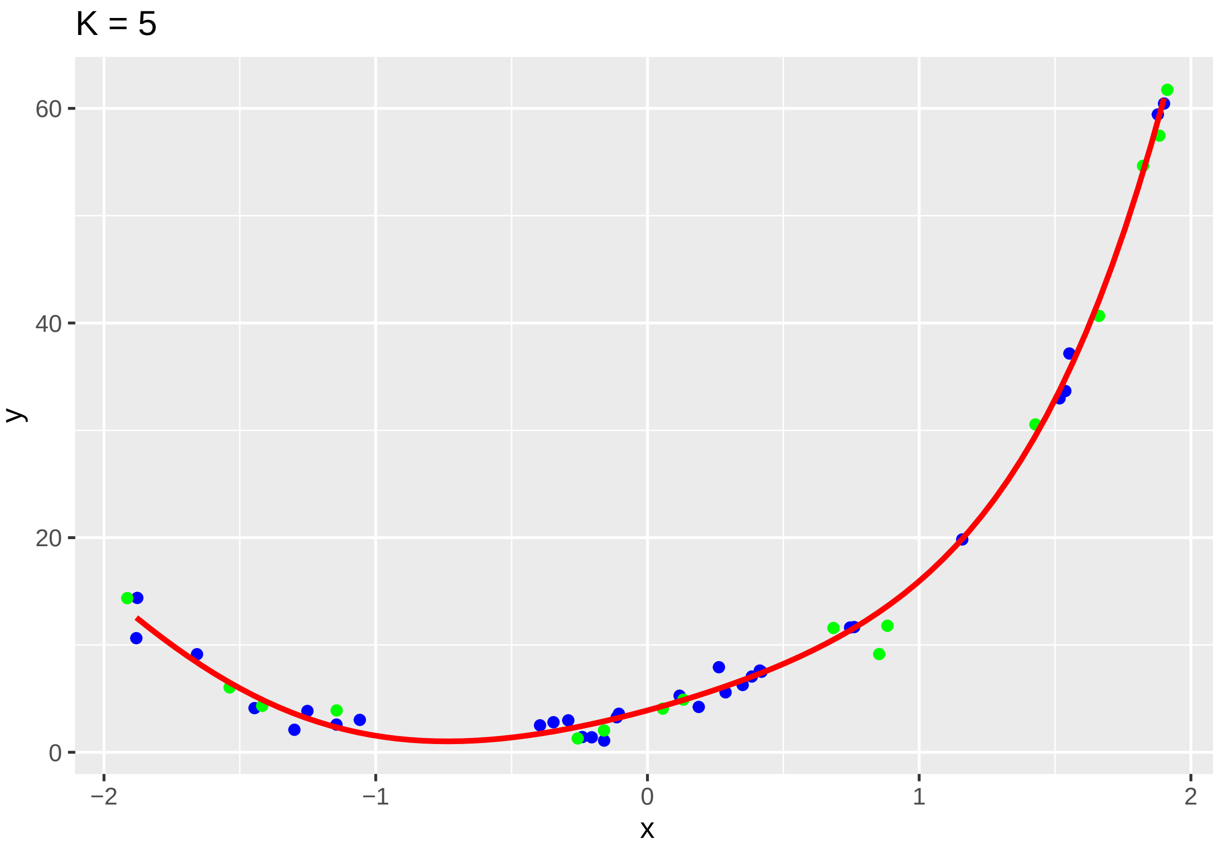


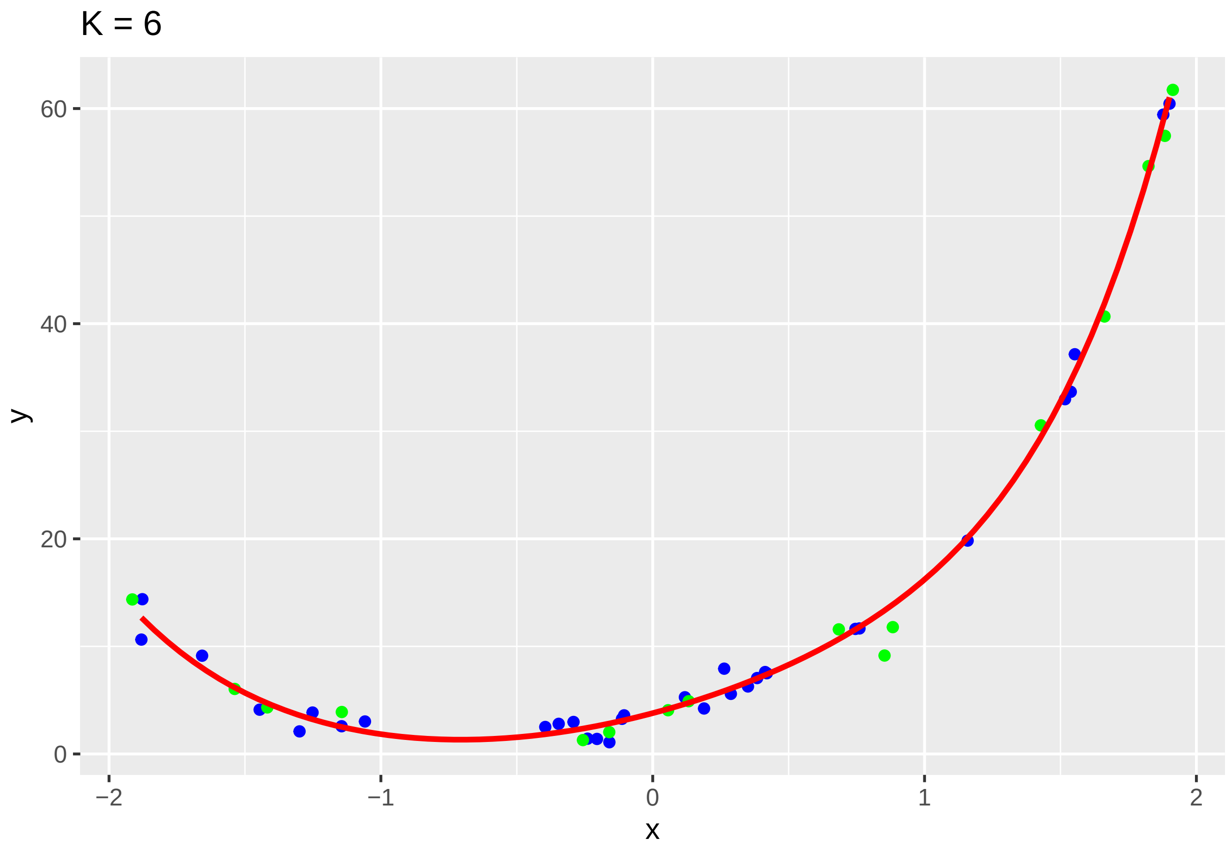
K = 2
60
40
20
0
y
−2
−1
0
1
2
x

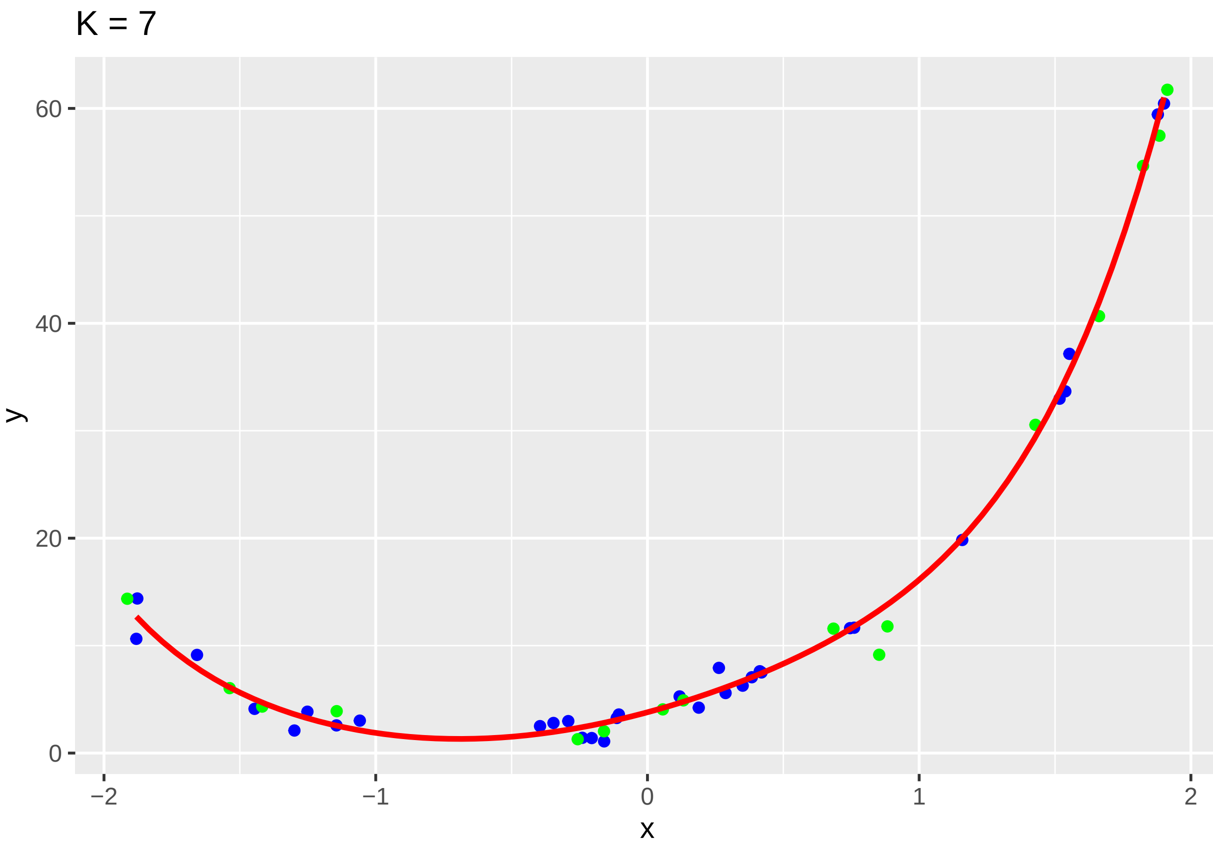


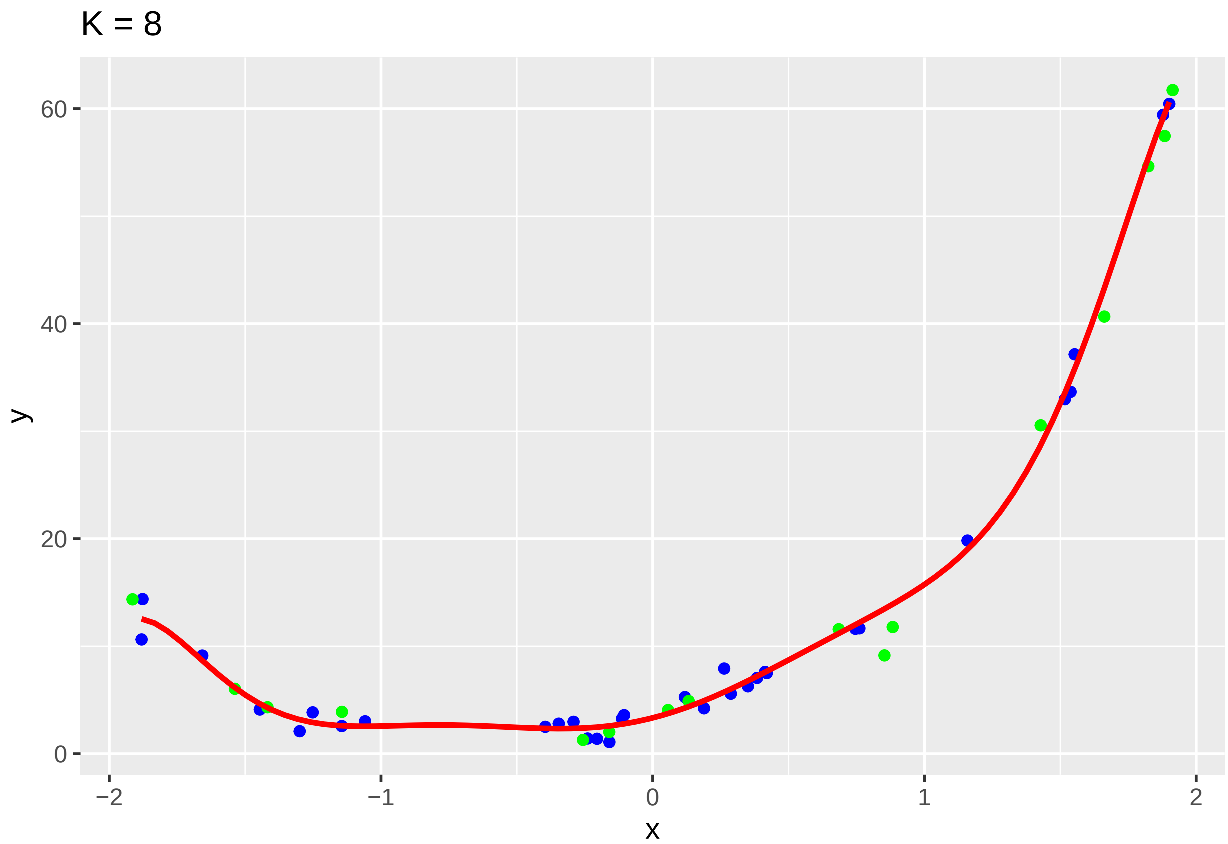
K = 3
60
40
20
0
y
−2
−1
0
1
2
x

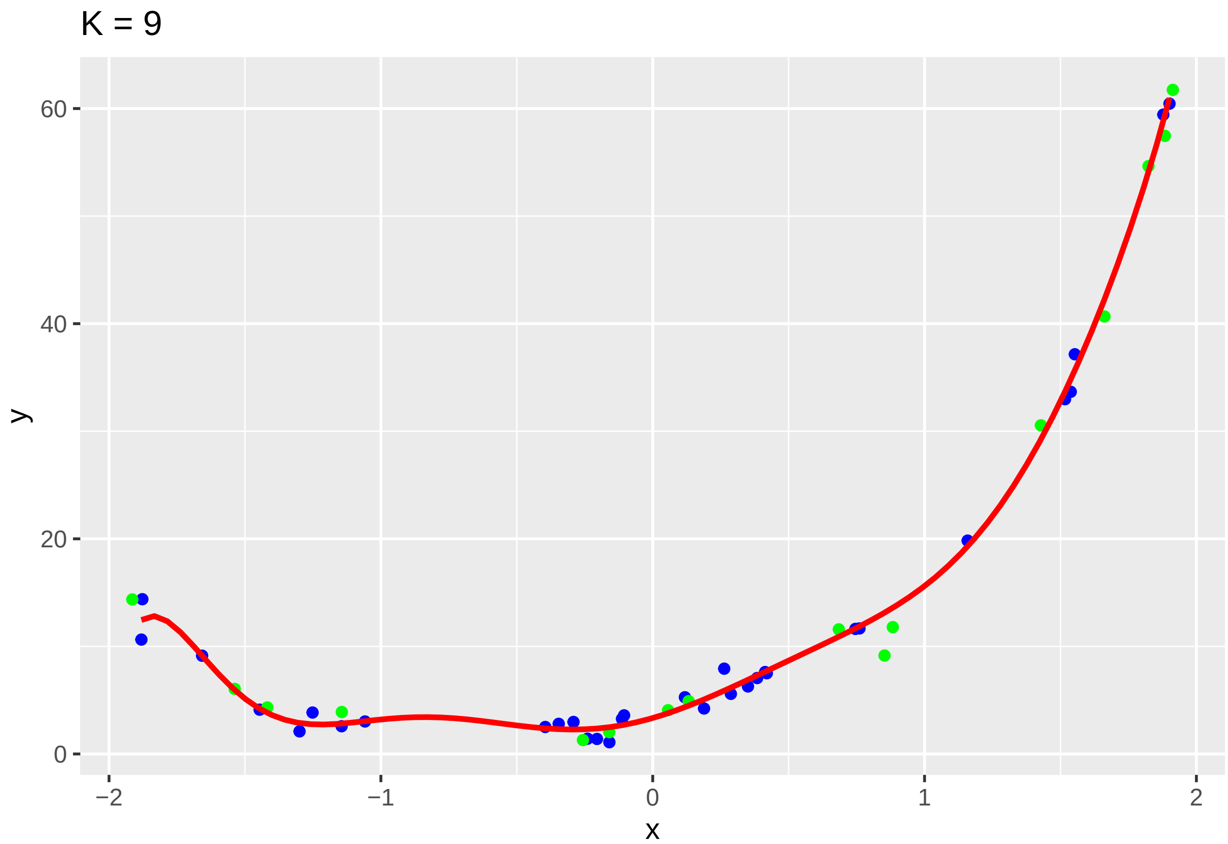

K = 4
60
40
20
0
y
−2
−1
0
1
2
x

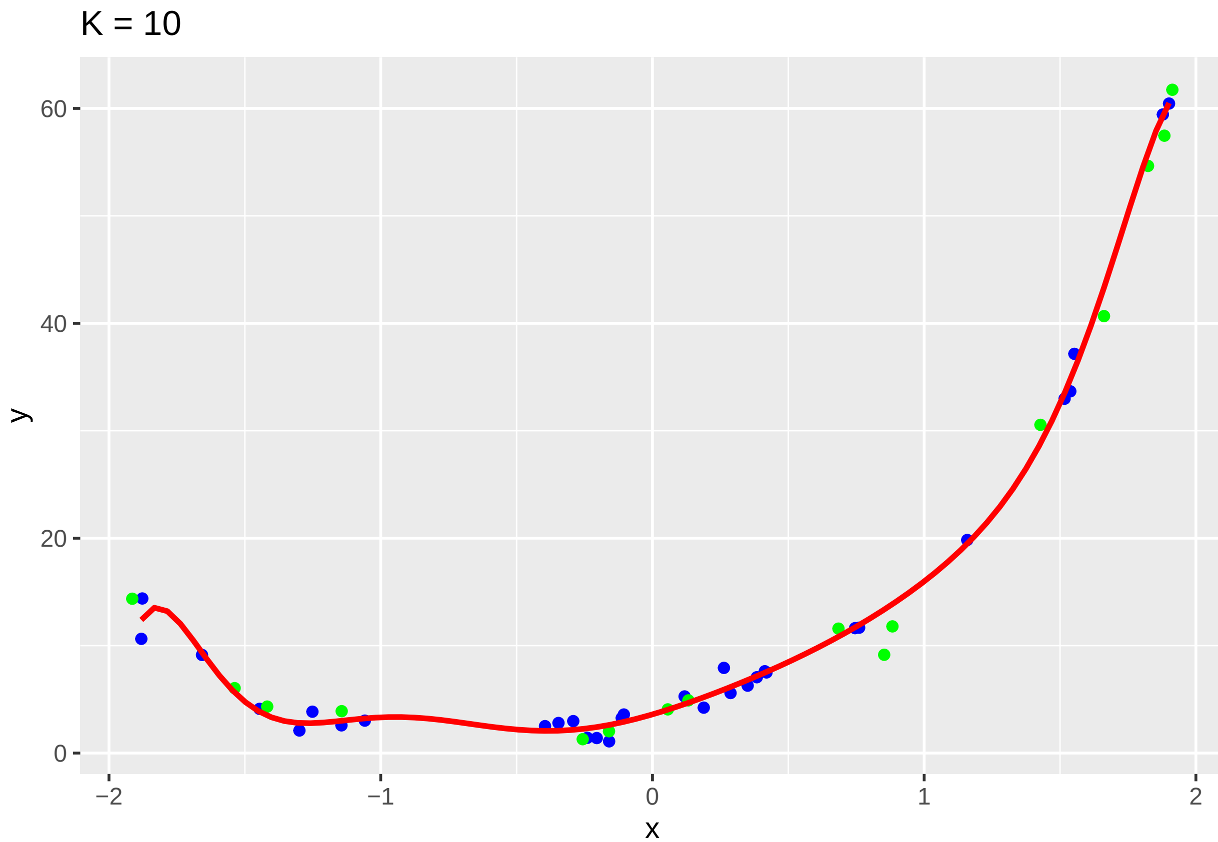

K = 5
60
40
20
0
y
−2
−1
0
1
2
x


K = 6
60
40
20
0
y
−2
−1
0
1
2
x


K = 7
60
40
20
0
y
−2
−1
0
1
2
x


K = 8
y
60
40
20
0
x
−2
−1
0
1
2


K = 9
60
40
20
0
y
−2
−1
0
1
2
x



K = 10
60
40
20
0
y
−2
−1
0
1
2
x