Task 1: Lab 1: Block 1

```
## Warning: package 'kknn' was built under R version 4.4.2
```

```
## Warning: package 'readr' was built under R version 4.4.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- 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
## New names:
## Rows: 3822 Columns: 65
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (65): 0...1, 1...2, 6...3, 15...4, 12...5, 1...6, 0...7, 0...8, 0...9, 7...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

2. Use training data to fit 30-nearest neighbor classifier with function kknn() and kernel="rectangular" from package kknn and estimate • Confusion matrices for the training and test data (use table()) • Misclassification errors for the training and test data Comment on the quality of predictions for different digits and on the overall prediction quality:

```
## [1] "Training Misclassification Error: 0.0424"
```

```
## [1] "Test Misclassification Error: 0.0492"
```

The misclassification level for both the train and test data seem to be at a fairly low level, at 0.0424 and 0.0492 respectively

Based from the calculations of the error rate for each digit on the training and testing data, the digits that appears the easiest to classify correctly for the knn model are 0, 2, 3, 6 and 7

The digits that seem to be the harder to predict are 1, 4, 5, 8, 9

```
## [1] "Training Confusion Matrix:"
```

```
##    train_pred
##       0   1   2   3   4   5   6   7   8   9
##   0 177   0   0   0   1   0   0   0   0   0
```

```
##   1   0 174   9   0   0   0   1   0   1   3
##   2   0   0 170   0   0   0   0   1   2   0
##   3   0   0   0 197   0   2   0   1   0   0
##   4   0   1   0   0 166   0   2   6   2   2
##   5   0   0   0   0   0 183   1   2   0  11
##   6   0   0   0   0   0   0 200   0   0   0
##   7   0   1   0   1   0   1   0 192   0   0
##   8   0  10   0   1   0   0   2   0 190   2
##   9   0   3   0   4   2   0   0   2   4 181
```

```
## [1] "Test Confusion Matrix:"
```

```
##    test_pred
##      0  1  2  3  4  5  6  7  8  9
##   0 97  0  0  0  0  0  1  0  0  0
##   1  0 91  3  0  0  0  0  0  0  3
##   2  0  0 93  1  0  0  0  0  1  0
##   3  0  0  0 95  0  0  0  2  1  0
##   4  1  0  0  0 89  0  1  5  1  3
##   5  0  1  0  1  0 79  1  0  0  5
##   6  0  0  0  0  0  0 94  0  0  0
##   7  0  2  0  0  0  1  0 91  1  0
##   8  0  3  0  1  0  0  1  0 86  0
##   9  0  0  0  4  0  0  0  2  1 94
```

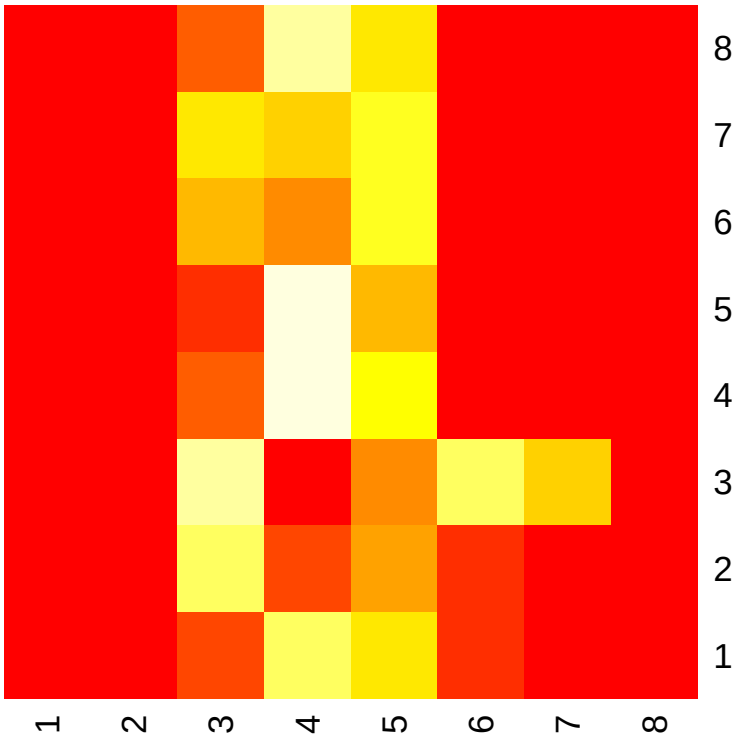
```
##           0           1           2           3           4           5
## 0.005617978 0.074468085 0.017341040 0.015000000 0.072625698 0.071065990
##           6           7           8           9
## 0.000000000 0.015384615 0.073170732 0.076530612
```

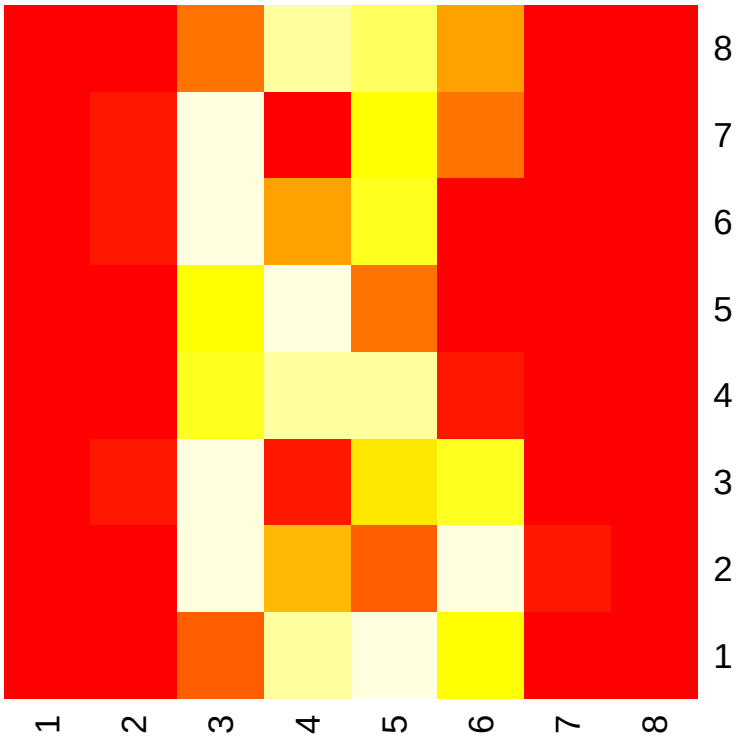
```
##          0          1          2          3          4          5          6
## 0.01020408 0.06185567 0.02105263 0.03061224 0.11000000 0.09195402 0.00000000
##          7          8          9
## 0.04210526 0.05494505 0.06930693
```

3. Find any 2 cases of digit "8" in the training data which were easiest to classify and 3 cases that were hardest to classify (i.e. having highest and lowest probabilities of the correct class). Comment on whether these cases seem to be hard or easy to recognize visually.

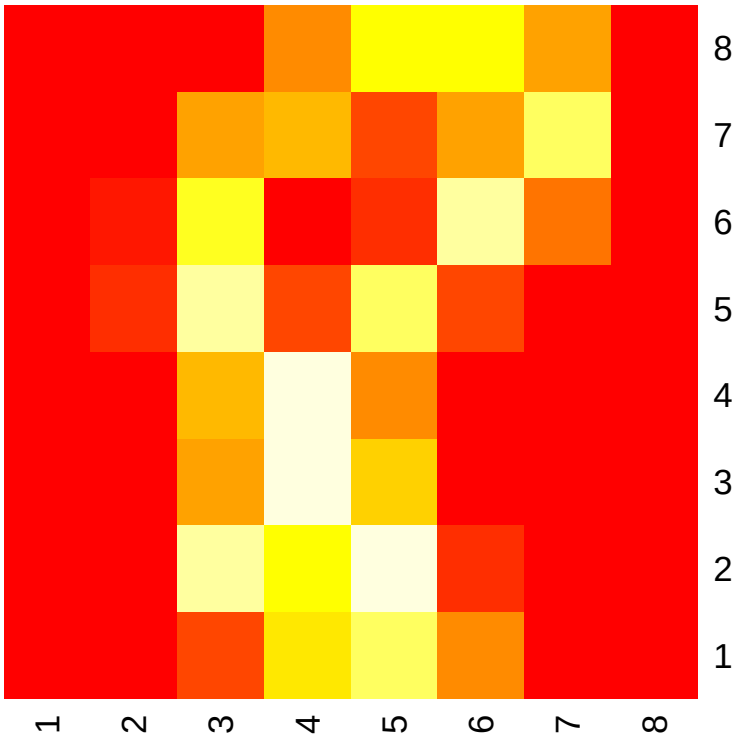
The first two digits seemed pretty definitely easier to recognize as an eight (altought the first one of them was maybe a little difficult to recognize as well). The other did appear much less recognizable as symbolizing the digit eight. In fact, looking at the probabilities for the predictions for them, you get the result down in the table below, indicating that althought the probability for the datapoints being 8 were not zero, the model found it much more likely for the datapoints to resemble other digits:

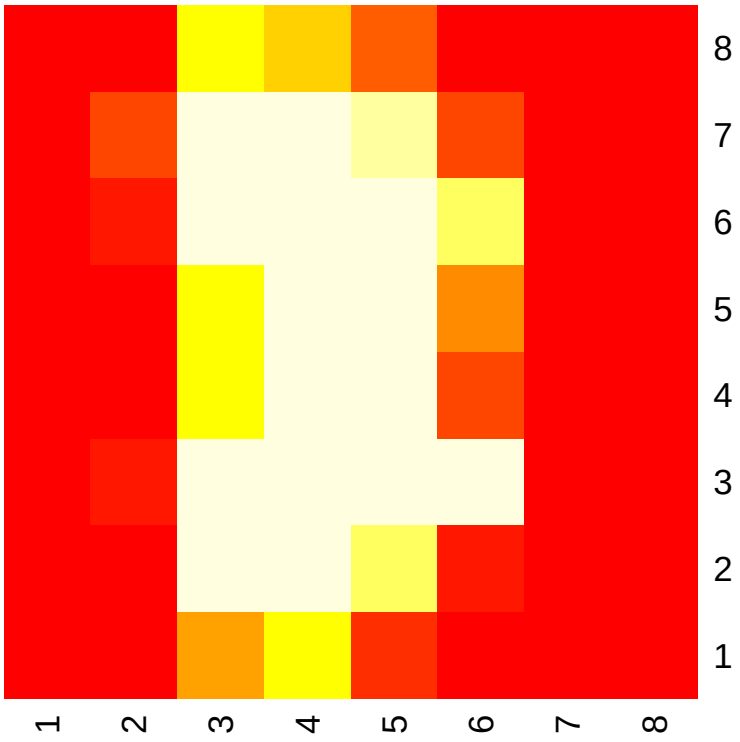
```
## Easiest cases of digit '8':
```

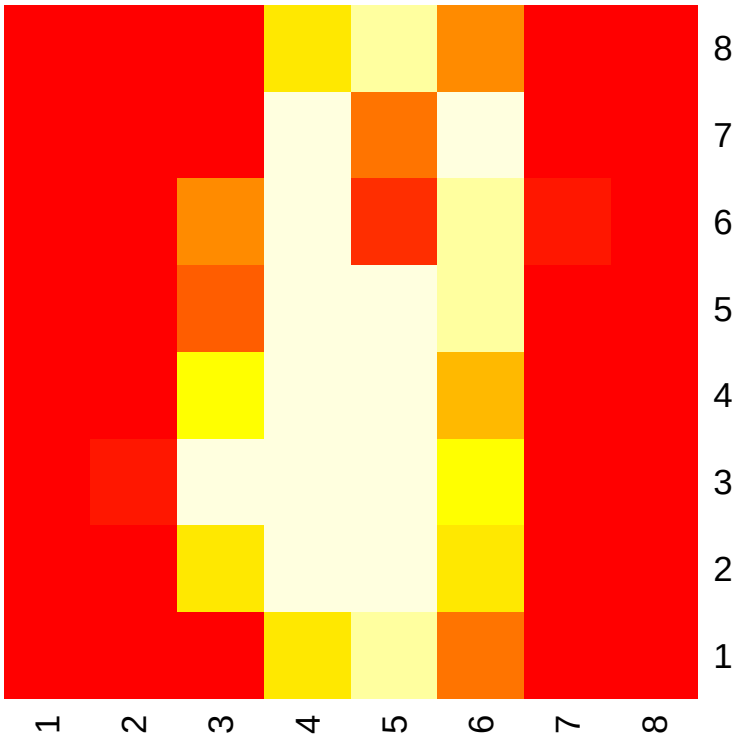

8
7
6
5
4
3
2
1
1
2
3
4
5
6
7
8




```
## Hardest cases of digit '8':
```


8
7
6
5
4
3
2
1
1
2
3
4
5
6
7
8

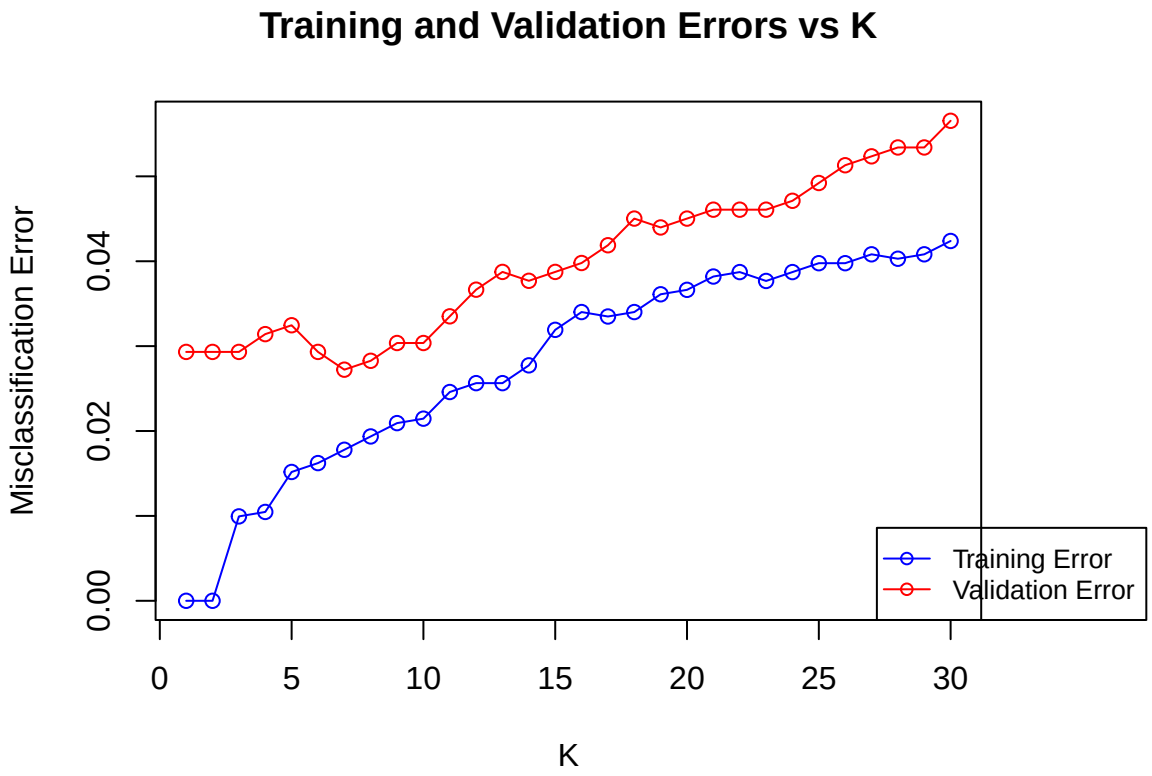

8
7
6
5
4
3
2
1
1
2
3
4
5
6
7
8




```
## probabilities for hardest cases of digit '8':

##      0         1 2 3 4 5   6 7         8 9
## [1,] 0 0.0000000 0 0 0 0 0.9 0 0.1000000 0
## [2,] 0 0.8333333 0 0 0 0 0.0 0 0.1666667 0
## [3,] 0 0.7666667 0 0 0 0 0.0 0 0.2333333 0

## probabilities for easiest cases of digit '8':

##      0 1 2 3 4 5 6 7 8 9
## [1,] 0 0 0 0 0 0 0 0 1 0
## [2,] 0 0 0 0 0 0 0 0 1 0
```

4. How does the model complexity change when K increases and how does it affect the training and validation? estimate the test error for the model having the optimal K, compare it with the training and validation errors and make necessary conclusions about the model quality:

As observed from the graph below, the misclassification seems to be generally lower for predictions on the training data than the validation data, which is to be expected since the model is trained to minimize the error on the training data. The error appears to only increase with K for the training data. This is simply because the model becomes more sensitive to local anomalies and is more likely to overfit to the training data for few K's. The validation error decreases a little bit from 1 to 7 K, but later grows as K also grows. The model was probably slightly overfit for the lower values of K, but reached a local optimum at around K = 7. As K grew, it probably began to become underfitted as it averages to many neighbors, including those far away from the test point, which explains why the validation error began to grow after K > 7.




```
## Optimal K: 7
```

```
## Optimal K: 7
```

```
## Test Error for optimal K: 0.03870293
```