Machine Learning - Assignment 1

Joel Grimmer

November 2021

1 Make Your Own

1. First find a set of metrics/attributes which are present for all students. One such set of metrics would be unit scores for all required modules for the unit, as students should have taken these units before beginning Machine Learning. The sample space X is a vector of real numbers from 0 to 100, each axis corresponding to a score for a required unit. The real numbers correspond to the percentage mark obtained for that unit.

2. The label space Y could either be a regression label, from 0 to 100, representing the students percentage score at the end of the unit, or, a multiclass classificaiton accross the set $\{-2, 0, 2, 4, 7, 10, 12\}$ corresponding to the Danish 7-point grading system. The former is chosen at it allows for more fine grained traning of our function.

3. The square loss function is chosen with Y' corresponding to an expected score for the unit, and Y being the actual score for that unit.

4. The distance measure is the Euclidean distance between a student's vector of scores and another student's vector of scores.

5. The performance of the algorithm could be assessed via the use of 5-fold cross-validaiton.

6. The algorithms do not take into account that different students progress at different rates and thus students with similar marks in some units may progress at different rates later on in the course. It also does not take into account individual cases such as illness which may unfairly penalize students who have had to deal with unfornute circumstances.

2 Digits Classification with K Nearest Neighbors

Task #1

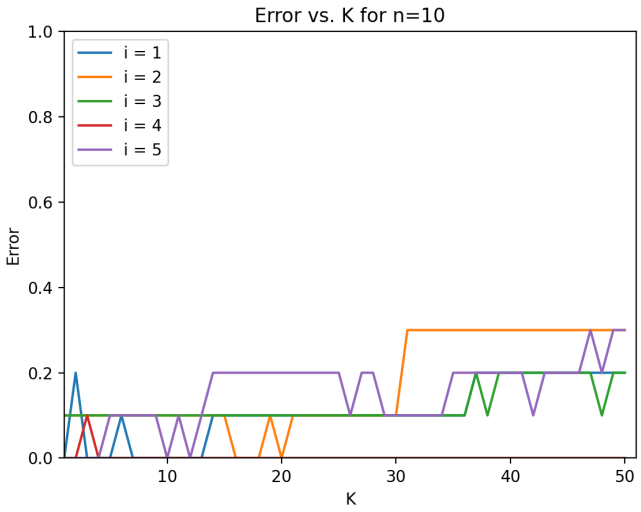

Figure 1: Error / K for $n = 10$

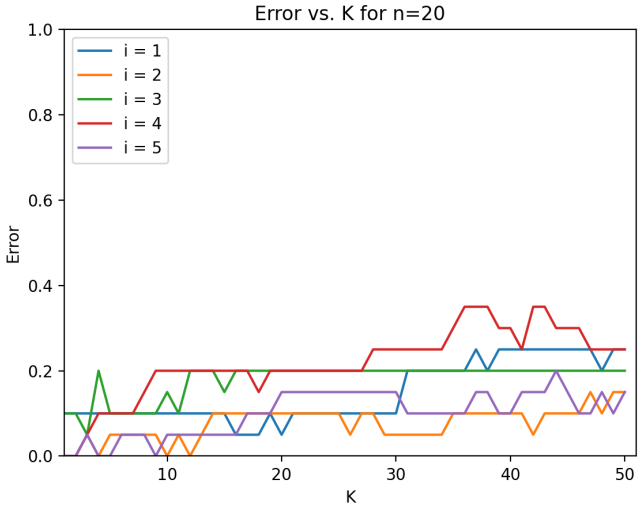

Figure 2: Error / K for $n = 20$

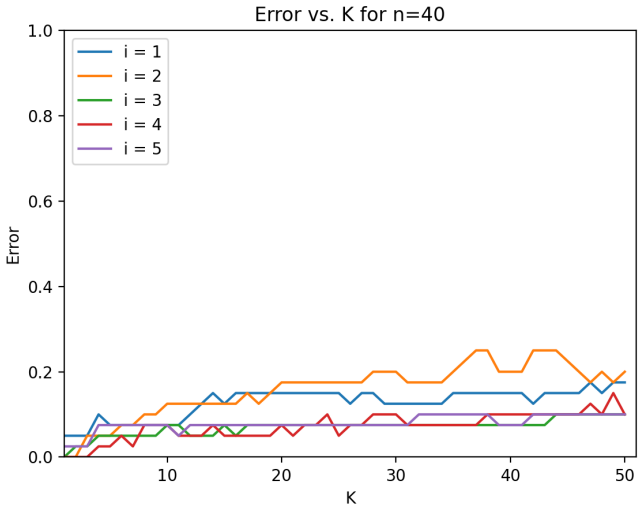

Figure 3: Error / K for $n = 40$

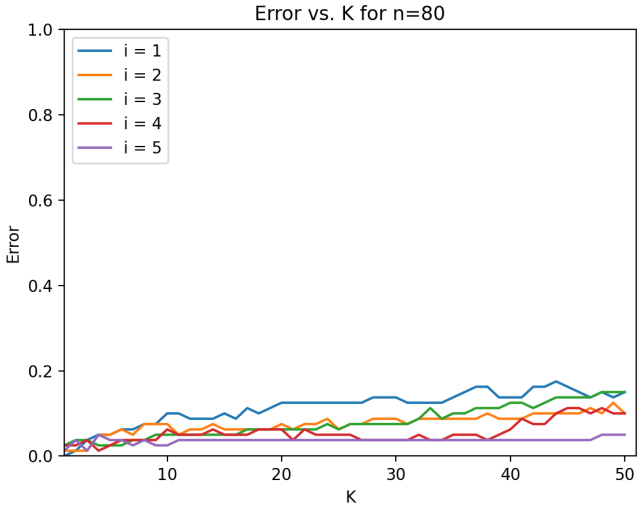

Figure 4: Error / K for $n = 80$

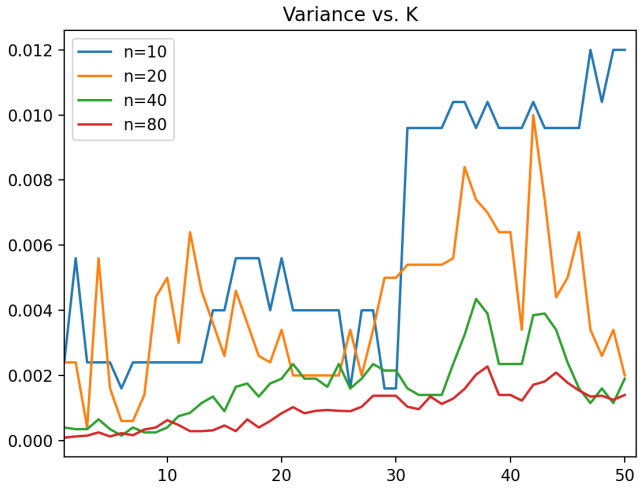

Figure 5: Variance / K

Implementation

The model and labels are loaded as two seperate np arrays. The training model is the first 100 elements of the model, and the training labels are the first 100

labels from the labels. We iterate over the list $n \in [10, 20, 40, 80]$, and produce plots for each in order to automate the process of graph creation. The function validation_sets_index(n, i off) takes an n representing the size of each set, an i representing the number of validation sets wanted, and an off representing the offset of the validation sets. This produces an np.ndarray of indices for the validation sets from an overall model of data (an array of arrays of indexes for each validation set of i).

We iterate over each validation set for each value of i. For each value of i, create an array k_diff_ts representing the error for each value of K, instantiated as 50 zeros. Iterate over each element in the ith validation set, and use the knn_pred(t_set, t_levels, x, max_k) function to produce an array of the knn predictions for each value of K. Compare this to the actual label, tiled K times, and use the absolute difference element wise between the two arrays to find the validation error $e \in \{0, 1\}$ for each value of K. Add this array to the k_diff_ts array (matrix-add).

This method is incredibly fast as knn can be calculated for all values of K in one go, using the knn_pred function. This function works by taking the training set, training labels, a given element x, and a max value of K (which must be smaller or equal to the size of the training set). The difference between the training set and the training set are calculated using matrices (and as such all in one go), and the sorted indices from the sort of the differences is used to sort the labels. Produce an array of k between 1 and 50 inclusive, and map this array to be the majority label for this k for the sorted labels. Return this array of k predictions.

Once all elements in the ith validation set have been iterated over, the error for each value of K is just k_diff_ts divided by n (matrix-division). Plot this array against $x \in \{1, ..., 50\}$ to create the line of error vs. K for a given i of size n. Repeat for all values of i, putting each ith plot on the same graph for a given value of n. Repeat with a new graph for each value of n.

Data Analysis from Plots

As n increases the fluctuations of validation error decrease. Having more comparisons results in less noise in the function from K to error. As n increases the difference in accuracy between validation sets for each i decreases. As K increases the validation error also increases, in a logarithmic shape. As n increases, the variance for all K decreases. The variance of the variance for K also decreases.

Task #2

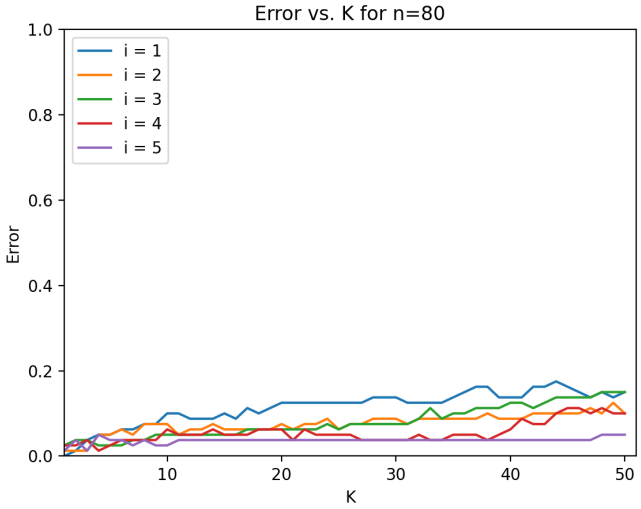

Figure 6: Error / K for $n = 80$

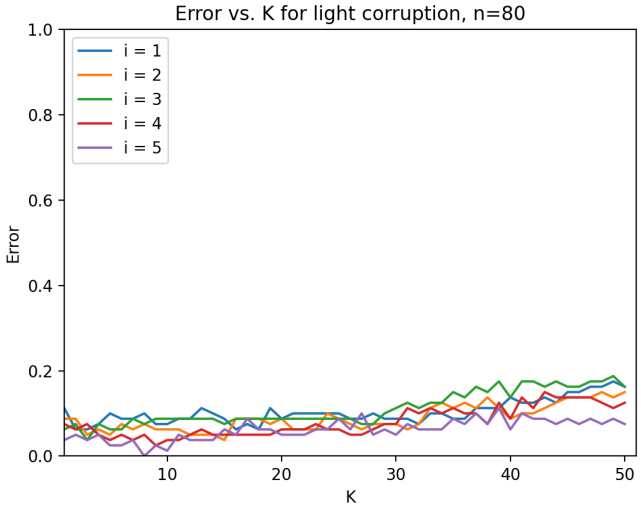

Figure 7: Error / K for $n = 80$ with light corruption

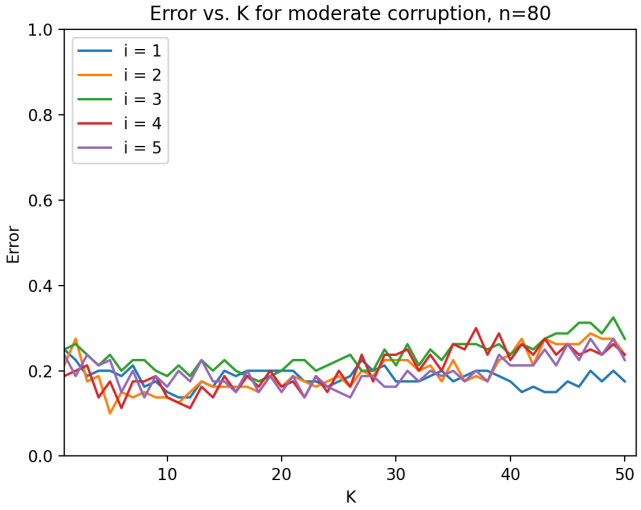

Figure 8: Error / K for $n = 80$ with moderate corruption

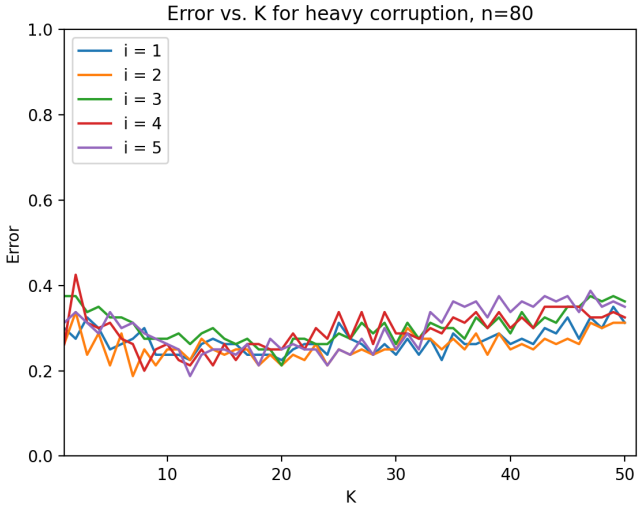

Figure 9: Error / K for $n = 48$ with heavy corruption

A training set with higher levels of corruption results in greater validation error (i.e. the prediction accuracy is lower for greater levels of corruption). For all levels of corruption the optimal value of K sits near $K = 10$, which is the square root of the number of items in the training set.

3 Illustration of Markov's, Chebyshev's, and Hoeffding's Inequalities

Part 1

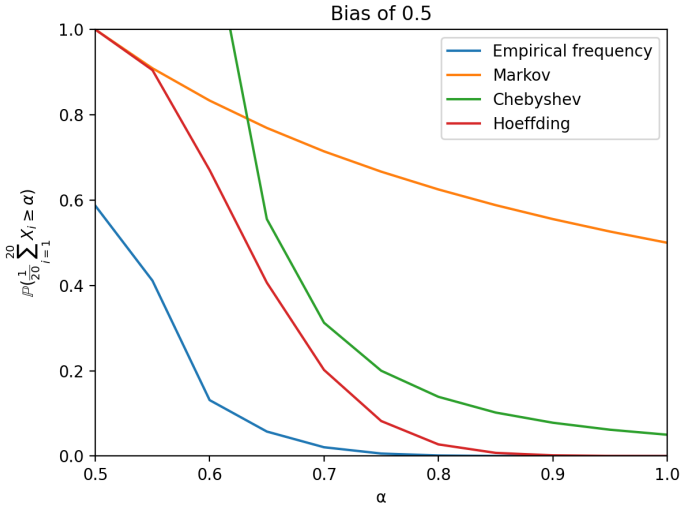

Figure 10: Probability vs α for bias of 0.5

It is sufficient to take a granularity of 0.05 for α, as the random variable $\frac{1}{20}\sum_{i=1}^{20} X_i$ produces outputs with the same granularity. Thus, selecting a higher granularity does not yield any more information as, $\mathbb{P}(\frac{1}{20}\sum_{i=1}^{20} X_i \geq 0.55)$ is equal to the probability that $\mathbb{P}(\frac{1}{20}\sum_{i=1}^{20} X_i \geq 0.51)$.

The empirical frequency maps the tightest. Markov's inequality initially has a superior binding to Chebyshev's inequality, although this changes between $\alpha = 0.65$ and $\alpha = 0.7$. Chebyshev's inequality is initially greater than 1.0, but is less than 1.0 between $\alpha = 0.60$ $\alpha = 0.65$. Hoeffding's inequality is either equal or tighter than the two other inequalities for all values of α.

We can use the binomial distribution to calculate $\mathbb{P}(\frac{1}{20}\sum_{i=1}^{20} X_i \geq \alpha)$.

For $\alpha = 1$, $\binom{20}{20} * (\frac{1}{2})^{20-20}(\frac{1}{2})^{20} = \binom{20}{20} * (\frac{1}{2})^{0}(\frac{1}{2})^{20} = 1/1048576 = 9.53674316 * 10^{-7}$

For $\alpha = 0.95$, $\binom{20}{19} * (\frac{1}{2})^{20-19}(\frac{1}{2})^{19} = \binom{20}{19} * (\frac{1}{2})^{1}(\frac{1}{2})^{19} = 5/262144 = 1.90734863 * 10^{-5}$

Part 2

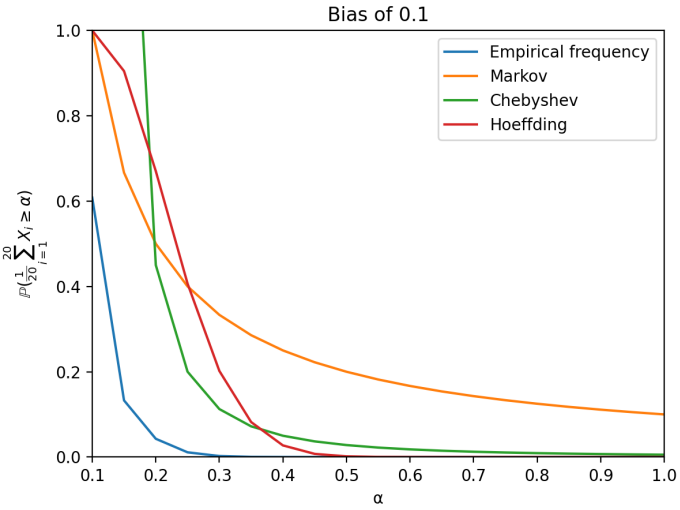

Figure 11: Probability vs α for bias of 0.1

The same principles of granularity apply as above, as there are still only 20 possible outputs from the random variable.

The empirical frequency maps the tightest. Markov's inequality initially has a superior binding to Chebyshev's inequality and Hoeffding's inequality, although it becomes worse than Chebyshev's at $\alpha = 0.2$ and Hoeffding's at $\alpha = 0.25$. Chebyshev's inequality is initially greater than 1.0, but is less than 1.0 between $\alpha = 0.20$ $\alpha = 0.25$. Hoeffding's inequality is tighter than the two other after $\alpha = 0.3$.

We can use the binomial distribution to calculate $\mathbb{P}(\frac{1}{20}\sum_{i=1}^{20} X_i \geq \alpha)$.

For $\alpha = 1$, $\binom{20}{20} * (\frac{1}{10})^{20-0}(\frac{9}{10})^{20-20} = \binom{20}{20} * (\frac{1}{10})^{20}(\frac{9}{10})^{0} = 1 * 10^{-20}$

For $\alpha = 0.95$, $\binom{20}{20} * (\frac{1}{10})^{20}(\frac{9}{10})^{0} + \binom{20}{20} * (\frac{1}{10})^{19}(\frac{9}{10})^{1} = 1.89 * 10^{-18}$

4 Basic Linear Algebra

We have a hyperplane defined by $\{x : w^T x + b = 0\}$. We need a vector wish points from the origin to a point in the hyperplane (a point where $w^T x + b = 0$). The vector which goes from this x to the origin should be perpendicular to the hyperplane, as this makes it the nearest point on the plane to the origin. This vector is w, and as such the distance is $||proj_w(origin - x)||$. This simplifies to $\frac{|origin \cdot w + b|}{||w||} = \frac{|b|}{||w||}$.

5 Regression

We wish to find values $w = (a, b, c)$ for $y = ax^2 + bx + c$. We know that the cannonball passes through $(0, 0)$, and as such $c = 0$.

$$X = \begin{bmatrix} 0^2 & 0 \\ 1^2 & 1 \\ \dots & \dots \\ 5^2 & 5 \end{bmatrix} \quad Y = \begin{bmatrix} 0 \\ 14 \\ 21 \\ 25 \\ 35 \\ 32 \end{bmatrix}$$

The we fit the parameters of X and Y using $w = (X \cdot X^T)^{-1} \cdot X^T \cdot Y$ to produce $w = (-1.37, 13.4)$, which maps to $y = -1.37x^2 + 13.4x$. We solve this using the quadratic formula to find that the line intersects the x axis at 0 (which we already knew), and 9.81 (where the ball reaches the ground).

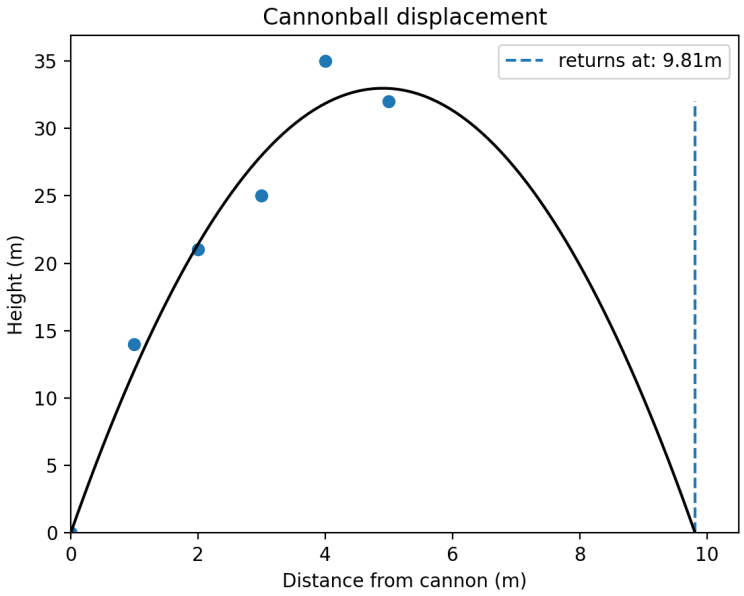

Figure 12: Cannonball height / distance from cannon