ECE 174 Linear and Nonlinear Optimization
Prof. Piya Pal

Mini Project 1: Least Squares Based Supervised Classification

November 6, 2022

Conner Hsu (A16665092)

Contents

1 Solutions for $\min||\mathbf{y} - A\mathbf{x}||$

I will prove that $\hat{\mathbf{x}}$ solves $\min||\mathbf{y} - A\mathbf{x}||$ if it is a solution to the normal equation,

$$A^TA\mathbf{x} = A^T\mathbf{y}$$

Let $\hat{\mathbf{x}}$ be a solution to the above equation. Let $\mathbf{v}$ be any vector in $\mathbb{R}^n$.

$$\begin{aligned}
||\mathbf{y} - A(\hat{\mathbf{x}} + \mathbf{v})||^2 &= ||\mathbf{y} - A\hat{\mathbf{x}} - A\mathbf{v}||^2 \\
&= (\mathbf{y} - A\hat{\mathbf{x}} - A\mathbf{v})^T(\mathbf{y} - A\hat{\mathbf{x}} - A\mathbf{v}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - \mathbf{y}^T(A\hat{\mathbf{x}}) - \mathbf{y}^T(A\mathbf{v}) - (A\hat{\mathbf{x}})^T\mathbf{y} - (A\mathbf{v})^T\mathbf{y} + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - 2\mathbf{y}^T(A\hat{\mathbf{x}}) - 2\mathbf{y}^T(A\mathbf{v}) + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - 2(A^T\mathbf{y})^T\hat{\mathbf{x}} - 2(A^T\mathbf{y})^T\mathbf{v} + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - 2(A^TA\hat{\mathbf{x}})^T\hat{\mathbf{x}} - 2(A^TA\hat{\mathbf{x}})^T\mathbf{v} + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - 2\hat{\mathbf{x}}^TA^TA\hat{\mathbf{x}} - 2\hat{\mathbf{x}}^TA^TA\mathbf{v} + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 + ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2 - 2||A\hat{\mathbf{x}}||^2 - 2(A\hat{\mathbf{x}})^T(A\mathbf{v}) + 2(A\mathbf{v})^T(A\hat{\mathbf{x}}) \\
&= ||\mathbf{y}||^2 - ||A\hat{\mathbf{x}}||^2 + ||A\mathbf{v}||^2
\end{aligned}$$

Picking $\mathbf{v}$ such that $||A\mathbf{v}||^2 = 0$ minimizes this expression. Any $\mathbf{v}$ not in $N(A)$ will cause $||\mathbf{y} - A(\hat{\mathbf{x}} + \mathbf{v})||^2$ to be greater than $||\mathbf{y} - A\hat{\mathbf{x}}||^2$. Thus, the solution to this least squares problem is $\hat{\mathbf{x}} + \mathbf{x}_0$ where $\hat{\mathbf{x}}$ is a solution to the normal equation and $\mathbf{x}_0 \in N(A)$. Note that this is also a solution to the normal equation.

$$\begin{aligned}
A^TA(\hat{\mathbf{x}} + \mathbf{x}_0) &= A^T\mathbf{y} \\
A^TA\hat{\mathbf{x}} + A^TA\mathbf{x}_0 &= A^T\mathbf{y} \\
A^TA\hat{\mathbf{x}} + A^T\mathbf{0} &= A^T\mathbf{y} \\
A^TA\hat{\mathbf{x}} &= A^T\mathbf{y}
\end{aligned}$$

In conclusion, any vector that is a solution to the normal equation is also a solution for the least squares minimization problem. Additionally, when A is not full rank, there can be infinite solutions to the problem.

2 The normal equation, $A^TA\mathbf{x} = A^T\mathbf{y}$

2.1 Preliminary argument 1: $R(A) \cap N(A^T) = \{\mathbf{0}\}$

Let $A \in \mathbb{R}^{m \times n}$ where $A = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n\end{bmatrix}$. Let $\mathbf{y}_1 \in \mathbb{R}^m$ such that $\mathbf{y}_1 \in R(A)$. This implies $\exists \mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{y}_1 = A\mathbf{x}$.

$$\mathbf{y}_1 = A\mathbf{x}$$

Let $\mathbf{y}_2 \in \mathbb{R}^m$ such that $\mathbf{y}_2 \in N(A^T)$. This implies that $A^T\mathbf{y}_2 = \mathbf{0}$ Now, let's see what happens when $\langle \mathbf{y}_1, \mathbf{y}_2 \rangle$ is computed.

$$\mathbf{y}_1^T\mathbf{y}_2 = (A\mathbf{x})^T\mathbf{y}_2 = x^TA^T\mathbf{y}_2 = x^T\mathbf{0} = \mathbf{0}$$

This means that $\mathbf{y}_1 \perp \mathbf{y}_2$. Since $\mathbf{y}_1$ is any vector in $R(A)$ and $\mathbf{y}_2$ is any vector in $N(A^T)$, we can say that all the vectors in $R(A)$ are orthogonal to all the vectors in $N(A^T)$, i.e. $R(A) \perp N(A^T)$.

Now suppose that $\mathbf{y}_3 \in \mathbb{R}^m$ such that $\mathbf{y}_3 \in R(A)$ and $\mathbf{y}_3 \in N(A^T)$. Since $R(A) \perp N(A^T)$,

$$\mathbf{y}_3^T\mathbf{y}_3 = 0 \implies \mathbf{y}_3 = \mathbf{0}$$

Thus, it can be said that

$$R(A) \cap N(A^T) = \{\mathbf{0}\}$$

2.2 Preliminary argument 2: Rank-nullity theorem

Let $A \in \mathbb{R}^{m\times n}$. Let $\dim(R(A)) = r$. This means that there are at most r columns in A that are linearly independent. This implies that there are $n-r$ columns in A that are linearly dependent on the other columns of A. Let the columns of A be rearranged such that its first r columns are the linearly independent ones and the rest are linearly dependent such that

$$A = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_r & \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_{(n-r)}\end{bmatrix} = \begin{bmatrix}A_1 & \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_{(n-r)}\end{bmatrix}$$

Note that rearranging the columns of A has no effect on $R(A)$. Since each $\mathbf{b}$ column is linearly dependent on the columns of A_1. Thus,

$$\mathbf{b}_i = A_1\mathbf{v}_i, \quad i = 1, 2, \dots, (n-r)$$

where $\mathbf{v}_i \in \mathbb{R}^r$. Thus,

$$A = \begin{bmatrix}A_1 & A_1\mathbf{v}_1 & A_1\mathbf{v}_2 & \dots & A_1\mathbf{v}_{(n-r)}\end{bmatrix}$$

Let $V = \begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2 & \dots \mathbf{v}_{(n-r)}\end{bmatrix}$.

$$A = \begin{bmatrix}A_1 & A_1V\end{bmatrix}$$

Let $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix}^T$ where $\mathbf{x}_1 \in \mathbb{R}^r$ and $\mathbf{x}_2 \in \mathbb{R}^{(n-r)}$

$$\begin{aligned} A\mathbf{x} &= \mathbf{0} \\ \begin{bmatrix}A_1 & A_1V\end{bmatrix}\begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = A_1\mathbf{x}_1 + A_1V\mathbf{x}_2 &= \mathbf{0} \\ A_1(\mathbf{x}_1 + V\mathbf{x}_2) &= \mathbf{0} \end{aligned}$$

Since A_1 is full rank,

$$\implies \mathbf{x}_1 + V\mathbf{x}_2 = \mathbf{0} \implies \mathbf{x}_1 = -V\mathbf{x}_2$$

Thus we can generalize any vector in $N(A)$ to have the form

$$\begin{bmatrix}-V \\ I_{(n-r)}\end{bmatrix}\mathbf{x}_2$$

In other words, any vector in $N(A)$ is in the span of the columns of $\begin{bmatrix}-V^T & I_{(n-r)}^T\end{bmatrix}^T$. Additionally, the columns of $\begin{bmatrix}-V^T & I_{(n-r)}^T\end{bmatrix}^T$ are linearly independent. Let $\mathbf{c} \in \mathbb{R}^{n-r}$

$$\begin{bmatrix}-V \\ I_{(n-r)}\end{bmatrix}\mathbf{c} = \mathbf{0}$$
$$\begin{bmatrix}-V\mathbf{c} = \mathbf{0} \\ I_{(n-r)}\mathbf{c} = \mathbf{0}\end{bmatrix}$$

Since $I_{(n-r)}\mathbf{c} = \mathbf{0} \implies \mathbf{c} = \mathbf{0}$, this means that the columns of $\begin{bmatrix}-V^T & I_{(n-r)}^T\end{bmatrix}^T$ are linearly independent. Thus, $\begin{bmatrix}-V^T & I_{(n-r)}^T\end{bmatrix}^T$ is also a basis for $N(A)$. Since $\begin{bmatrix}-V^T & I_{(n-r)}^T\end{bmatrix}^T$ is $n \times (n-r)$,

$$\dim(N(A)) = n - r$$

In other words,

$$\dim(N(A)) + \dim(R(A)) = n$$

Additionally,

$$\dim(N(A^T)) + \dim(R(A^T)) = m$$

2.3 Preliminary argument 3: $\dim(R(A)) + \dim(N(A^T)) = m$

Let $A \in \mathbb{R}^{m\times n}$ and let $\dim(R(A)) = r$. Since $\dim(R(A)) = \dim(R(A^T))$, by rank-nullity theorem $\dim(N(A^T)) = m - r$. As a result,

$$\dim(R(A)) + \dim(N(A^T)) = r + (m - r) = m$$

2.4 Preliminary argument 4: Every $\mathbf{v} \in \mathbb{R}^m$ can be written as the sum of vectors in $R(A)$ and in $N(A^T)$

Let B_1 be a matrix whose columns form a basis for $R(A)$. Let B_2 be a matrix whose columns form a basis for $N(A^T)$. If $\dim(R(A)) = r$ then $B_1 \in \mathbb{R}^{m\times r}$ and $B_2 \in \mathbb{R}^{m\times(m-r)}$. Let $\mathbf{x}_1 \in \mathbb{R}^r$ and $\mathbf{x}_2 \in \mathbb{R}^{m-r}$. I will prove the matrix $\begin{bmatrix} B_1 & B_2 \end{bmatrix}$ has m linearly independent columns.

$$\begin{aligned} \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} &= \mathbf{0} \\ B_1\mathbf{x}_1 + B_2\mathbf{x}_2 = \mathbf{0} &\implies B_1\mathbf{x}_1 = -B_2\mathbf{x}_2 = B_2(-\mathbf{x}_2) \end{aligned}$$

$B_1\mathbf{x}_1 = B_2(-\mathbf{x}_2)$ is satisfied when $B_1\mathbf{x}_1 \in R(B_2)$ and $B_2(-\mathbf{x}_2) \in R(B_1)$. Since $R(A) \cap N(A^T) = \{\mathbf{0}\}$, $R(B_1) \cap R(B_2) = \{\mathbf{0}\}$. This means that $B_1\mathbf{x}_1 = B_2(-\mathbf{x}_2)$ is only possible if $B_1\mathbf{x}_1 = B_2(-\mathbf{x}_2) = \mathbf{0}$. This implies that $\mathbf{x}_1, \mathbf{x}_2 = \mathbf{0}$. As a result,

$$\begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \mathbf{0} \implies \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \mathbf{0}$$

Thus $\begin{bmatrix} B_1 & B_2 \end{bmatrix}$ has m linearly independent columns and thus it spans $\mathbb{R}^m$. Furthermore, the expression

$$\begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = B_1\mathbf{x}_1 + B_2\mathbf{x}_2$$

shows that any vector in $\mathbb{R}^m$ can be expressed as the sum of a vector in $R(A)$ and a vector in $N(A^T)$.

2.5 Existence and uniqueness of solutions for $A^TA\mathbf{x} = A^T\mathbf{y}$

Let $A \in \mathbb{R}^{m\times n}$ and let $\mathbf{y} \in \mathbb{R}^m$ such that

$$\mathbf{y} = \mathbf{y}_1 + \mathbf{y}_2$$

where $\mathbf{y}_1 \in R(A)$ and $\mathbf{y}_2 \in N(A^T)$. Using results from earlier, it is known that $\mathbf{y}$ could be any vector in $\mathbb{R}^m$. Plugging $\mathbf{y}$ into the normal equation yields

$$\begin{aligned} A^TA\mathbf{x} &= A^T(\mathbf{y}_1 + \mathbf{y}_2) \\ A^TA\mathbf{x} &= A^T\mathbf{y}_1 + A^T\mathbf{y}_2 \\ A^TA\mathbf{x} &= A^T\mathbf{y}_1 + \mathbf{0} \\ A^TA\mathbf{x} &= A^T\mathbf{y}_1 \end{aligned}$$

Since $\mathbf{y}_1 \in R(A)$, a solution for this equation can always be obtained by row reducing

$$A\mathbf{x} = \mathbf{y}_1$$

A few things can be noted about the number of solutions from this result:

- If A is full rank, then the normal equation has one solution.
- If A is not full rank, then the normal equation can have infinite solutions.

2.6 Computing solutions for $A^T A\mathbf{x} = A^T\mathbf{y}$

I will prove that if A is a full rank matrix, $A^T A$ is invertible. By the invertible matrix theorem, $A^T A$ is invertible if and only if $N(A^T A) = \{\mathbf{0}\}$. Let $\mathbf{x} \in \mathbb{R}^n$.

$$\begin{aligned} A^T A\mathbf{x} &= \mathbf{0} \\ \mathbf{x}^T A^T A\mathbf{x} &= 0 \\ (A\mathbf{x})^T(A\mathbf{x}) &= 0 \\ ||A\mathbf{x}||^2 &= 0 \end{aligned}$$

Since the norm of a vector can only equal zero if the vector itself is zero, $A\mathbf{x} = \mathbf{0}$ and thus $\mathbf{x} \in N(A)$. However, because A is full rank, only $\mathbf{x} = \mathbf{0}$ satisfies the above equations.

$$\begin{aligned} A^T\mathbf{y} &= A^T A\mathbf{x} \\ \implies \mathbf{x} &= (A^T A)^{-1}A^T\mathbf{y} \end{aligned}$$

The above shows that an explicit formula for $\mathbf{x}$ can be found if A is full rank. However, the MNIST training data is not full rank. A method for finding a basis for the training data will be needed. We can work towards finding a basis for the training data by removing columns that satisfy the following conditions.

1. the entries of the column are all zero
2. the column is a duplicate of another column
3. the column is a scalar multiple of another column

It is worth noting that two of these algorithms have $O(n^2)$ time complexity, so this would take a decently long time to run. More than that, however, these algorithms don't even guarantee a basis for A! This means that a fully realized implementation that finds a basis for a matrix would run even longer! As a result, in my code I have used numpy's pinv() function to compute the solution to the normal equation for the sake of convienience.

3 Least squares classifiers

3.1 One-versus-all classifier

Table 1: Confusion matrix for one-versus-all based multiclassifier.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	944	0	1	2	2	7	14	2	7	1	980	3.67 %
1	0	1107	2	2	3	1	5	1	14	0	1135	2.47 %
2	18	54	813	26	15	0	42	22	37	5	1032	21.22 %
3	4	17	23	880	5	17	9	21	22	12	1010	12.87 %
4	0	22	6	1	881	5	10	2	11	44	982	10.29 %
5	23	18	3	72	24	659	23	14	39	17	892	26.12 %
6	18	10	9	0	22	17	875	0	7	0	958	8.66 %
7	5	40	16	6	26	0	1	884	0	50	1028	14.01 %
8	14	46	11	30	27	40	15	12	759	20	974	22.07 %
9	15	11	2	17	80	1	1	77	4	801	1009	20.61 %
All	1041	1325	886	1036	1085	747	995	1035	900	950	10000	13.97 %

From table 1, the total error of the one-versus-all classifier can be calculated to be 13.97%. Looking at the entries on the off-diagonal, we can notice a few interesting results.

- When the outcome was 9, the 1vAll model would often classify it as a 4 (80 occurrences) or as a 7 (77 occurences). This is fairly sensible given that many 9s look much like 4 or 7.
- Similarly, when the outcome was a 4 or 7, the model classified it as a 4, 7, or 9.
- 5s are mistaken as 3s pretty often (72 occurences).
- 2s are mistaken as 1s fairly often, however ones are rarely mistaken as 2s.

3.2 One-versus-one classifier

Table 2: Confusion matrix for one-versus-one based multiclassifier.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	961	0	1	1	0	6	8	3	0	0	980	1.94 %
1	0	1120	3	3	1	1	4	1	2	0	1135	1.32 %
2	9	18	936	12	10	5	10	10	22	0	1032	9.30 %
3	9	1	19	926	2	19	1	7	21	5	1010	8.32 %
4	2	4	6	1	932	1	7	3	3	23	982	5.09 %
5	7	5	3	30	8	800	17	2	15	5	892	10.31 %
6	6	5	12	0	5	19	908	1	2	0	958	5.22 %
7	1	17	17	3	10	1	0	955	1	23	1028	7.10 %
8	7	17	8	23	10	36	10	10	840	13	974	13.76 %
9	6	5	1	12	30	11	0	21	3	920	1009	8.82 %
All	1008	1192	1006	1011	1008	899	965	1013	909	989	10000	7.02 %

From table 2, the total error of the one-versus-one classifier can be calculated to be 7.02%. Again, interesting results can be realized by looking at the off-diagonal entries.

- 9 is often misclassified as a 4 or 7. Similarly, 4 and 7 are often misclassified as a 9.
- 5s are still mistaken as 3s (30 occurences).
- 2s are now mistaken as 8s more often than they are mistaken as 1s.

4 Randomized feature based least square classifiers

4.1 Performance of different feature spaces

4.1.1 Identity function

The feature space using the identity function has similar performance to the classifiers without the feature mapping. For the one-versus-one classifier, performance increased slightly.

Table 3: Confusion matrix for one-versus-all based multiclassifier with an identity feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	944	0	1	2	2	7	14	2	7	1	980	3.67 %
1	0	1107	2	2	3	1	5	1	14	0	1135	2.47 %
2	18	54	813	26	15	0	42	22	37	5	1032	21.22 %
3	4	17	23	880	5	17	9	21	22	12	1010	12.87 %
4	0	22	6	1	881	5	10	2	11	44	982	10.29 %
5	23	18	3	72	24	659	23	14	39	17	892	26.12 %
6	18	10	9	0	22	17	875	0	7	0	958	8.66 %
7	5	40	16	6	26	0	1	884	0	50	1028	14.01 %
8	14	46	11	30	27	40	15	12	759	20	974	22.07 %
9	15	11	2	17	80	1	1	77	4	801	1009	20.61 %
All	1041	1325	886	1036	1085	747	995	1035	900	950	10000	13.97 %

Table 4: Confusion matrix for one-versus-one based multiclassifier with an identity feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	960	0	2	1	0	6	8	3	0	0	980	2.04 %
1	0	1118	4	3	1	1	4	1	3	0	1135	1.50 %
2	9	18	938	12	10	3	10	9	23	0	1032	9.11 %
3	10	1	19	926	2	18	1	7	21	5	1010	8.32 %
4	2	3	7	1	932	1	8	3	3	22	982	5.09 %
5	7	6	3	29	8	799	17	2	15	6	892	10.43 %
6	6	5	12	0	5	19	908	1	2	0	958	5.22 %
7	1	17	17	3	8	1	0	957	1	23	1028	6.91 %
8	7	18	8	23	10	36	10	10	839	13	974	13.86 %
9	6	5	1	12	29	11	0	21	3	921	1009	8.72 %
All	1008	1191	1011	1010	1005	895	966	1014	910	990	10000	7.02 %

4.1.2 Sigmoid function

The feature space using the sigmoid function improves the performance of both classifiers significantly.

Table 5: Confusion matrix for one-versus-all based multiclassifier with a sigmoid feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	962	0	2	2	0	3	6	2	3	0	980	1.84 %
1	0	1119	3	3	1	3	3	0	3	0	1135	1.41 %
2	13	2	944	14	13	1	8	13	24	0	1032	8.53 %
3	0	2	12	933	0	24	6	15	16	2	1010	7.62 %
4	1	2	3	0	922	2	14	2	5	31	982	6.11 %
5	8	2	2	27	9	796	19	9	15	5	892	10.76 %
6	8	3	4	0	11	13	917	0	2	0	958	4.28 %
7	1	14	17	6	10	1	1	945	4	29	1028	8.07 %
8	5	3	6	18	11	16	15	7	881	12	974	9.55 %
9	6	7	3	14	32	9	2	21	7	908	1009	10.01 %
All	1004	1154	996	1017	1009	868	991	1014	960	987	10000	6.73 %

Table 6: Confusion matrix for one-versus-one based multiclassifier with a sigmoid feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	968	0	2	3	0	3	2	0	2	0	980	1.22 %
1	0	1125	3	1	0	2	2	0	2	0	1135	0.88 %
2	8	1	987	4	3	1	5	9	14	0	1032	4.36 %
3	3	0	9	960	0	13	2	8	13	2	1010	4.95 %
4	1	0	4	0	948	0	6	2	2	19	982	3.46 %
5	8	1	1	17	4	842	10	1	5	3	892	5.61 %
6	7	2	2	0	5	8	929	0	5	0	958	3.03 %
7	1	7	15	4	7	1	0	973	3	17	1028	5.35 %
8	3	2	5	10	5	13	6	4	920	6	974	5.54 %
9	7	5	1	14	22	3	0	8	5	944	1009	6.44 %
All	1006	1143	1029	1013	994	886	962	1005	971	991	10000	4.04 %

4.1.3 Sinusoidal function (degrees)

The sinusoidal feature space makes both classifiers perform better than the identity function, but not as good as the sigmoid function. I thought that the performance of the sinusoidal function should be similar to the identity since at small values of x, $\sin(x) \approx x$. It is important to note that the sinusoidal function used is in degrees, not radians.

Table 7: Confusion matrix for one-versus-all based multiclassifier with a sinusoidal feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	959	0	1	1	0	7	7	1	3	1	980	2.14 %
1	0	1121	3	1	1	2	4	0	2	1	1135	1.23 %
2	8	23	914	16	9	3	16	14	27	2	1032	11.43 %
3	3	8	17	912	4	18	4	17	19	8	1010	9.70 %
4	2	13	4	1	908	1	10	1	10	32	982	7.54 %
5	10	9	2	29	10	777	18	10	12	15	892	12.89 %
6	11	4	3	0	9	17	910	0	4	0	958	5.01 %
7	2	27	16	3	14	0	0	936	2	28	1028	8.95 %
8	8	18	7	24	9	25	13	14	847	9	974	13.04 %
9	5	11	3	12	43	12	2	22	3	896	1009	11.20 %
All	1008	1234	970	999	1007	862	984	1015	929	992	10000	8.20 %

Table 8: Confusion matrix for one-versus-one based multiclassifier with a sinusoidal feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	963	0	1	0	0	7	4	2	3	0	980	1.73 %
1	0	1124	2	2	0	1	3	1	2	0	1135	0.97 %
2	9	4	972	10	5	1	8	11	12	0	1032	5.81 %
3	2	1	12	970	0	6	2	6	9	2	1010	3.96 %
4	2	2	6	1	939	0	6	3	2	21	982	4.38 %
5	8	3	3	18	2	830	11	2	9	6	892	6.95 %
6	8	3	2	0	3	10	929	0	3	0	958	3.03 %
7	0	10	18	3	4	2	0	970	3	18	1028	5.64 %
8	3	1	11	17	7	12	9	6	904	4	974	7.19 %
9	7	7	1	11	21	5	0	11	6	940	1009	6.84 %
All	1002	1155	1028	1032	981	874	972	1012	953	991	10000	4.59 %

4.1.4 Rectified linear unit function (ReLU)

The ReLU feature space has the best performance for both the 1vAll and 1v1 classifiers.

Table 9: Confusion matrix for one-versus-all based multiclassifier with a ReLU feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	962	1	0	1	0	2	9	1	4	0	980	1.84 %
1	0	1122	2	2	1	0	5	1	2	0	1135	1.15 %
2	4	2	956	9	9	2	12	12	25	1	1032	7.36 %
3	0	2	5	962	0	7	3	13	11	7	1010	4.75 %
4	1	5	4	0	929	1	8	3	6	25	982	5.40 %
5	8	2	1	24	6	818	18	4	7	4	892	8.30 %
6	8	3	0	1	8	13	922	1	2	0	958	3.76 %
7	1	14	14	3	10	3	0	958	4	21	1028	6.81 %
8	6	5	3	16	8	15	8	5	897	11	974	7.91 %
9	5	7	1	14	28	6	1	11	9	927	1009	8.13 %
All	995	1163	986	1032	999	867	986	1009	967	996	10000	5.47 %

Table 10: Confusion matrix for one-versus-one based multiclassifier with a ReLU feature mapping applied.

Outcome	Prediction										Total	Error
	0	1	2	3	4	5	6	7	8	9		
0	968	0	1	0	0	3	4	2	2	0	980	1.22 %
1	0	1126	3	2	1	0	1	0	2	0	1135	0.79 %
2	5	1	1006	1	3	1	3	5	7	0	1032	2.52 %
3	0	0	10	981	0	2	0	7	7	3	1010	2.87 %
4	1	2	5	0	953	0	3	2	2	14	982	2.95 %
5	7	2	0	11	1	859	9	1	1	1	892	3.70 %
6	6	3	0	0	3	11	931	0	4	0	958	2.82 %
7	2	5	18	1	4	0	0	985	2	11	1028	4.18 %
8	4	1	4	11	4	7	3	5	930	5	974	4.52 %
9	5	4	1	11	18	1	0	4	6	959	1009	4.96 %
All	998	1144	1048	1018	987	884	954	1011	963	993	10000	3.02 %

4.2 Performance with respect to number of features

Figure 1 shows that as more features are added, the error rate decreases exponentially. At a certain point, however, adding more features doesn't have much effect on the error rate. Compared to the rest of the feature mappings, the identity doesn't improve much from increased features after around $L = 500$.

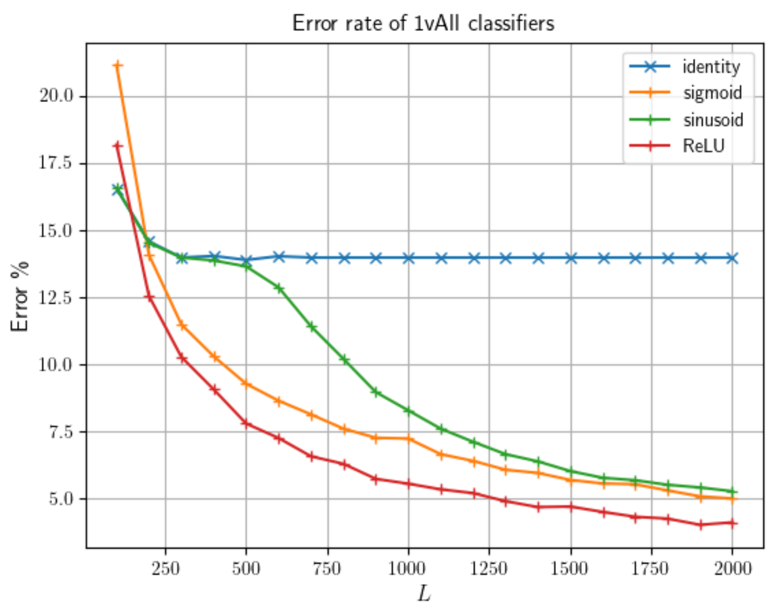

Figure 1: Plot of error rate of 1vAll classifiers with respect to L

4.3 Performance with noisy data

A noise vector will be generated to add noise to the data. The noise vector should satisfy the condition $||\mathbf{n}_i|| \leq \epsilon$ where ϵ is the noise level. To make such a vector, a random vector, $\hat{\mathbf{n}}_i$ will be generated with entries drawn from $\mathcal{N}(0,1)$. Then, the properties of the norm will be used to make sure that $||\mathbf{n}_i|| \leq \epsilon$ is true. Let $c \in \mathbb{R}^n$.

$$||c\hat{\mathbf{n}}_i||^2 = (c\hat{\mathbf{n}}_i)^T(c\hat{\mathbf{n}}_i) = c^2(\hat{\mathbf{n}}_i)^T(\hat{\mathbf{n}}_i) = c^2||\hat{\mathbf{n}}_i||^2 \implies ||c\hat{\mathbf{n}}_i|| = |c|||\hat{\mathbf{n}}_i||$$

Now, let $\mathbf{n}_i = c\hat{\mathbf{n}}_i$

$$\begin{aligned} ||\mathbf{n}_i|| &\leq \epsilon \\ |c|||\hat{\mathbf{n}}_i|| &\leq \epsilon \\ |c| &\leq \frac{\epsilon}{||\hat{\mathbf{n}}_i||} \end{aligned} \tag{1}$$

The above equation gives a formula for what to scale $\hat{\mathbf{n}}_i$ by in order to ensure $||\mathbf{n}_i|| < \epsilon$. The 1vAll classifiers from earlier were all tested to see how they perform with noisy test data.

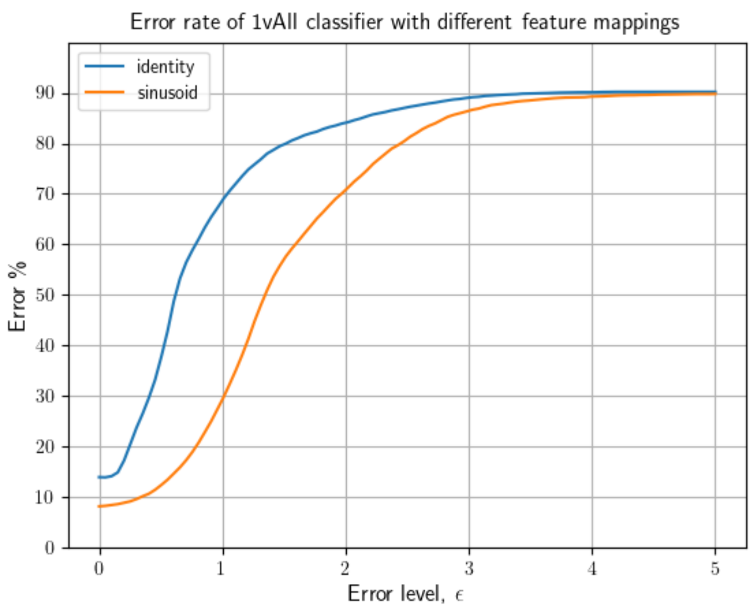

Figure 2: Plot of error rate of 1vAll classifiers with noisy test images

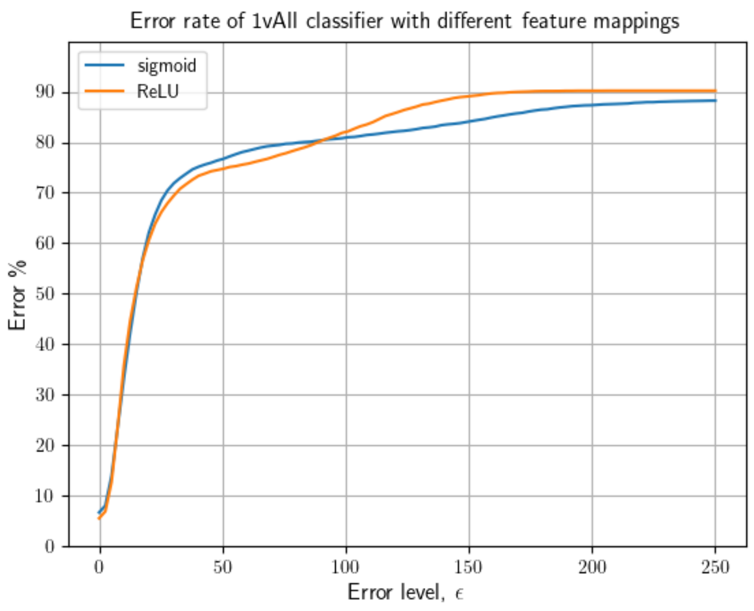

Figure 3: Plot of error rate of 1vAll classifiers with noisy test images

Figure 2 shows that the sinusoidal feature space is better at dealing with noise than the identity feature space. However, figure 3 shows that the sigmoid and ReLU feature spaces perform significantly better than both the identity and sinusoidal feature space. It is particularly interesting to look at when each feature space reaches 90% error because that's when the classifier performs just as well as a classifier that chooses randomly.

- Identity breaks at $\epsilon \approx 3$

- Sinusoidal breaks at $\epsilon \approx 4$
- Sigmoid breaks at $\epsilon \approx 250$
- ReLU breaks at $\epsilon \approx 150$

4.4 Varying probability distributions on the feature map

Table 11: Error rates for varying normal distributions on W and $\mathbf{b}$.

Classifier	$g(x)$	Error rate $\mathcal{N}(0,1)$	$\mathcal{N}(0,2)$	$\mathcal{N}(0,10)$	$\mathcal{N}(0,20)$	$\mathcal{N}(0,50)$	$\mathcal{N}(0,100)$	$\mathcal{N}(3,1)$
1vAll	Identity	13.97 %	13.97 %	13.97 %	13.97 %	13.97 %	13.97 %	13.97 %
	Sigmoid	6.73 %	7.45 %	8.26 %	8.47 %	8.13 %	7.78 %	88.65 %
	Sinusoid	8.20 %	7.83 %	6.07 %	20.34 %	84.42 %	89.80 %	31.36 %
	ReLU	5.47 %	5.53 %	5.79 %	5.26 %	5.48 %	5.63 %	13.97 %
1v1	Identity	7.02 %	6.98 %	7.00 %	7.00 %	6.99 %	7.00 %	7.01 %
	Sigmoid	4.04 %	4.47 %	4.93 %	5.03 %	5.03 %	4.79 %	88.65 %
	Sinusoid	4.59 %	4.22 %	3.73 %	19.92 %	84.43 %	89.66 %	27.85 %
	ReLU	3.02 %	3.47 %	3.26 %	3.42 %	3.36 %	3.21 %	7.01 %

When the normal distributions used to generate W and $\mathbf{b}$ were changed, some feature spaces were affected more than others. A few trends are notable.

- The identity mapping remained the same regardless of (μ, σ).
- Changing the mean of the normal distrubition seemed to have a negative effect on on all of the classifiers except the identity mapping. This is probably because the sigmoid, sinusoidal, and ReLU function have some significant point at 0, so shifting all the data away from 0 is not ideal for these functions.
- The sinusoidal mapping had increasing error with increasing σ.
  - The exception to this is $\mathcal{N}(0, 10)$. Perhaps with finer increments of σ, we could notice that the errorrate of the sinusoidal mapping doesn't strictly increase with σ.

It's also interesting to try W and $\mathbf{b}$ with each drawing from a different normal distribution. I tested three, each of which reveal how feature maps work.

- $W \sim \mathcal{N}(1, 0), b \sim \mathcal{N}(0, 0)$ is when W and $\mathbf{b}$ have no effect on the data.
- $W \sim \mathcal{N}(0, 1), b \sim \mathcal{N}(0, 0)$ is when only $\mathbf{b}$ has no effect on the data.
- $W \sim \mathcal{N}(1, 0), b \sim \mathcal{N}(0, 1)$ is when only W has no effect on the data.

$W \sim \mathcal{N}(0, 1), b \sim \mathcal{N}(0, 1)$ is also shown just for reference.

Table 12: Error rates for different normal distributions each on W and $\mathbf{b}$.

Classifier	$g(x)$	Error rate $W \sim \mathcal{N}(0,1)$, $b \sim \mathcal{N}(0,1)$	$W \sim \mathcal{N}(1,0)$, $b \sim \mathcal{N}(0,0)$	$W \sim \mathcal{N}(0,1)$, $b \sim \mathcal{N}(0,0)$	$W \sim \mathcal{N}(1,0)$, $b \sim \mathcal{N}(0,1)$
1vAll	Identity	13.97 %	80.68 %	13.97 %	80.72 %
	Sigmoid	6.73 %	88.65 %	7.03 %	88.65 %
	Sinusoid	8.20 %	84.66 %	8.43 %	78.83 %
	ReLU	5.47 %	80.68 %	5.70 %	80.72 %
1v1	Identity	7.02 %	78.01 %	7.01 %	78.01 %
	Sigmoid	4.04 %	88.65 %	4.49 %	88.65 %
	Sinusoid	4.59 %	82.89 %	4.76 %	77.74 %
	ReLU	3.02 %	78.01 %	3.35 %	78.01 %

It appears that W is the most important for feature maps because when W has no effect on the data, the error rates are really high.