HW 4

Q1

a. See hw4-nhvasan.zip for more details.
b. See hw4-nhvasan.zip for more details.
c. See hw4-nhvasan.zip for more details.
d. See hw4-nhvasan.zip for more details.

Q2

a. See code for implementation details, but here is the model's performance on the Binary and Count Datasets with 200 epochs.

Binary:

```
{0: 2892, 1: 1307, 2: 1307, 3: 187, 4: 268, 5: 198, 6: 226, 7: 154, 8: 142, 9: 116, 10: 114, 11: 110, 12: 102, 13: 104, 14: 85, 15: 81, 16: 74, 17: 77, 18: 66, 19: 68,
 20: 59, 21: 62, 22: 56, 23: 58, 24: 52, 25: 49, 26: 45, 27: 40, 28: 38, 29: 35, 30: 44, 31: 42, 32: 49, 33: 51, 34: 57, 35: 46, 36: 44, 37: 25, 38: 27, 39: 22, 40: 23
, 41: 26, 42: 21, 43: 25, 44: 21, 45: 25, 46: 23, 47: 18, 48: 16, 49: 19, 50: 16, 51: 16, 52: 18, 53: 17, 54: 20, 55: 18, 56: 17, 57: 18, 58: 19, 59: 22, 60: 18, 61: 1
6, 62: 15, 63: 18, 64: 19, 65: 21, 66: 15, 67: 12, 68: 13, 69: 12, 70: 17, 71: 15, 72: 10, 73: 12, 74: 14, 75: 11, 76: 11, 77: 13, 78: 13, 79: 12, 80: 12, 81: 9, 82: 1
0, 83: 9, 84: 11, 85: 10, 86: 8, 87: 9, 88: 8, 89: 5, 90: 8, 91: 7, 92: 15, 93: 7, 94: 7, 95: 8, 96: 8, 97: 10, 98: 6, 99: 7, 100: 7, 101: 6, 102: 7, 103: 8, 104: 10,
105: 19, 106: 9, 107: 6, 108: 7, 109: 8, 110: 7, 111: 7, 112: 6, 113: 5, 114: 7, 115: 5, 116: 5, 117: 7, 118: 5, 119: 6, 120: 5, 121: 5, 122: 4, 123: 5, 124: 4, 125: 4
, 126: 4, 127: 4, 128: 4, 129: 5, 130: 3, 131: 5, 132: 3, 133: 5, 134: 4, 135: 4, 136: 8, 137: 4, 138: 4, 139: 4, 140: 4, 141: 4, 142: 4, 143: 4, 144: 4, 145: 5, 146:
3, 147: 3, 148: 3, 149: 2, 150: 3, 151: 5, 152: 8, 153: 10, 154: 14, 155: 4, 156: 3, 157: 2, 158: 2, 159: 2, 160: 3, 161: 2, 162: 3, 163: 3, 164: 2, 165: 3, 166: 3, 16
7: 2, 168: 3, 169: 3, 170: 2, 171: 2, 172: 2, 173: 1, 174: 3, 175: 2, 176: 3, 177: 2, 178: 1, 179: 2, 180: 2, 181: 1, 182: 4, 183: 2, 184: 2, 185: 2, 186: 2, 187: 1, 1
88: 3, 189: 2, 190: 2, 191: 1, 192: 3, 193: 1, 194: 2, 195: 1, 196: 1, 197: 1, 198: 1, 199: 1}
Number of mistakes on the test dataset
32
```

Count:

```
{0: 2892, 1: 1307, 2: 1268, 3: 2736, 4: 1288, 5: 722, 6: 545, 7: 672, 8: 2690, 9: 1263, 10: 428, 11: 681, 12: 700, 13: 2058, 14: 1193, 15: 337, 16: 348, 17: 308, 18: 3
32, 19: 296, 20: 323, 21: 410, 22: 645, 23: 845, 24: 1909, 25: 1166, 26: 349, 27: 371, 28: 412, 29: 458, 30: 751, 31: 742, 32: 1288, 33: 929, 34: 672, 35: 589, 36: 763
, 37: 621, 38: 741, 39: 603, 40: 660, 41: 548, 42: 564, 43: 467, 44: 356, 45: 230, 46: 233, 47: 219, 48: 214, 49: 210, 50: 207, 51: 205, 52: 211, 53: 200, 54: 207, 55:
 201, 56: 209, 57: 198, 58: 205, 59: 199, 60: 199, 61: 193, 62: 199, 63: 191, 64: 198, 65: 190, 66: 198, 67: 192, 68: 217, 69: 190, 70: 203, 71: 198, 72: 209, 73: 193,
 74: 197, 75: 190, 76: 204, 77: 187, 78: 184, 79: 180, 80: 177, 81: 181, 82: 182, 83: 234, 84: 181, 85: 185, 86: 179, 87: 225, 88: 176, 89: 184, 90: 176, 91: 226, 92:
175, 93: 171, 94: 169, 95: 195, 96: 169, 97: 174, 98: 175, 99: 248, 100: 196, 101: 285, 102: 397, 103: 588, 104: 1421, 105: 1111, 106: 560, 107: 542, 108: 713, 109: 65
2, 110: 968, 111: 741, 112: 709, 113: 522, 114: 409, 115: 308, 116: 221, 117: 179, 118: 175, 119: 174, 120: 165, 121: 164, 122: 171, 123: 156, 124: 172, 125: 155, 126:
 166, 127: 153, 128: 162, 129: 155, 130: 178, 131: 155, 132: 142, 133: 153, 134: 153, 135: 172, 136: 151, 137: 144, 138: 151, 139: 152, 140: 173, 141: 158, 142: 145, 1
43: 144, 144: 142, 145: 142, 146: 138, 147: 140, 148: 139, 149: 141, 150: 140, 151: 142, 152: 142, 153: 142, 154: 139, 155: 140, 156: 138, 157: 141, 158: 138, 159: 143
, 160: 140, 161: 138, 162: 133, 163: 138, 164: 137, 165: 143, 166: 136, 167: 186, 168: 141, 169: 132, 170: 142, 171: 141, 172: 140, 173: 139, 174: 132, 175: 135, 176:
129, 177: 133, 178: 119, 179: 131, 180: 123, 181: 126, 182: 135, 183: 118, 184: 127, 185: 124, 186: 125, 187: 121, 188: 129, 189: 122, 190: 115, 191: 117, 192: 115, 19
3: 118, 194: 113, 195: 124, 196: 123, 197: 118, 198: 113, 199: 117}
Number of mistakes on the test dataset
71
```

The model converges slower on the Count dataset, resulting in a higher number of mistakes on the test data compared to the Binary dataset. However, this is to be expected because the Count (non-binary) data holds more information, and since the number mistakes is on a downward progression on average as the number of epochs increases, this result makes sense.

b. For my K-Fold Cross Validation implementation, I chose k=10 as that is a value that is commonly chosen for k. I first determined the optimal number of epochs for both datasets.

Optimal Epochs

- Binary: The optimal number of epochs based off of number of mistakes is 10,000.

	Epoch	Num_Mistakes
0	1	130.7
1	10	14.6
2	20	10.7
3	50	8.7
4	100	9.7
5	250	9.2
6	500	8.7
7	1000	8.7
8	2500	8.7
9	5000	8.5
10	10000	8.2

	Epoch	Num_Mistakes
10	10000	8.2

- Count: The optimal number of epochs based off of number of mistakes is 2500.

	Epoch	Num_Mistakes
0	1	130.7
1	10	65.8
2	20	67.9
3	50	22.1
4	100	24.7
5	250	16.5
6	500	13.6
7	1000	13.1
8	2500	12.2
9	5000	12.2
10	10000	12.6

	Epoch	Num_Mistakes
8	2500	12.2
9	5000	12.2

I then trained the model using that number of epochs, and reported both the test and training error (number of mistakes) for both datasets.

Binary

- Train: 0
- Test: 32

```
Number of mistakes on the train dataset
0
Number of mistakes on the test dataset
32
```

Count

- Train: 9
- Test: 48

```
Number of mistakes on the train dataset
9
Number of mistakes on the test dataset
48
```

c. See the word lists for the top 15 words with the highest and lowest weights for both the Binary and Count datasets below.

Binary:

- Positive: ['on', 'haven', 'dial', 'might', 'ar', 'the', 'achiev', 'numbertnumb', 'each', 'email', 'without', 'split', 'current', 'gordon', 'like']
- Negative: ['client', 'simpli', 'friend', 'those', 'back', 'specif', 'septemb', 'chat', 'remain', 'should', 'do', 'send', 'link', 'apolog', 'famili']

```
Top 15 Positive Weights:  ['on', 'haven', 'dial', 'might', 'ar', 'the', 'achiev', 'numbertnumb', 'each', 'email', 'without', 'split', 'current', 'gordon', 'like']
Top 15 Negative Weights:  ['client', 'simpli', 'friend', 'those', 'back', 'specif', 'septemb', 'chat', 'remain', 'should', 'do', 'send', 'link', 'apolog', 'famili']
```

Count:

- Positive: ['haven', 'method', 'that', 'on', 'dial', 'numbertnumb', 'keyboard', 'ascii', 'yourself', 'weekli', 'make', 'promot', 'brent', 'pc', 'each']
- Negative: ['flat', 'few', 'request', 'fact', 'next', 'deathtospamdeathtospamdeathtospam', 'septemb', 'accept', 'do', 'those', 'them', 'simpli', 'back', 'specif', 'apolog']

```
Top 15 Positive Weights:  ['haven', 'method', 'that', 'on', 'dial', 'numbertnumb', 'keyboard', 'ascii', 'yourself', 'weekli', 'make', 'promot', 'brent', 'pc', 'each']
Top 15 Negative Weights:  ['flat', 'few', 'request', 'fact', 'next', 'deathtospamdeathtospamdeathtospam', 'septemb', 'accept', 'do', 'those', 'them', 'simpli', 'back',
 'specif', 'apolog']
```

Q3

```
(CS334) nikhitas-mbp-3:hw4-nhvasan niki$ python q3.py
Logit Number of Mistakes [Binary Data] 29
Logit Number of Mistakes [Count Data] 39
Naive Bayes Number of Mistakes [Binary Data] 52
Naive Bayes Number of Mistakes [Count Data] 55
```

a. The Naive Bayes model performs slightly better on the Binary dataset than the count, with 52 mistakes on the test Binary dataset versus 55 on the test Count dataset.
b. The Logistic Regression model performs better on the Binary dataset as well, with 29 mistakes on the test Binary dataset versus 39 on the test Count dataset.