Raymond Lin
304937942

Homework 3

1. **Instructions**
    a. Partners: Derek Hu

Raymond Lin
304937942

2. Overfitting

a. A model is overfitting when there is a high discrepancy between the training error and testing error. The training error is extremely low, while the testing error is extremely high.

b. Yes, deploying an overfitted model is extremely bad, because the overfitted model is not representative of the data's trends. As a result, predictions will have extremely poor accuracy.

c. We typically tune the hyperparameters of a particular model or, in other words, tune the complexity of the model and find the best hyperparameter, where the training and testing errors are roughly the same and at their lowest points. At low complexities, the model usually underfits, and both train and test errors are high. As we increase the complexity of the model, we find the train and test errors to decrease until a certain point. This is our ideal complexity for the model. As we increase the complexity even more, the model usually overfits, and the train error starts to decrease, while the test error starts to increase.

   Furthermore, if we have too many features or have irrelevant features in our dataset, overfitting can also occur. This adds noise to our training model and thus overfits the model.

d. Dropout is a method of regularization typically used in neural networks. At each layer, a random number of activations are dropped during training. While this may sound counter-intuitive, it actually strengthens the model by requiring all activations to carry the same information. In other words, if some of them are dropped, other ones can still carry the same information, making the model more robust and preventing overfitting.

   Early stopping is another method of regularization, where we stop training before the maximum number of iterations or epochs is reached. This is because machine learning models typically tend to overfit the data the longer it spends training on the dataset. As a result, we stop training whenever we see the performance on the test set start to degrade.

   L2 or weight decay is another method of regularization. A new regularization term that is the product of a hyperparameter and the L2-norm of the weights is added to the cost function. The hyperparameter controls the magnitude of the weights. As the hyperparameter increases, the weights decrease, thus leading to a simpler model and the reduction of overfitting.

Raymond Lin
304937942

3. Overfitting Mitigation

a. No, using a smaller dataset will not mitigate overfitting. If we use a smaller dataset, it is less likely to be representative of the population that we are trying to predict and thus less likely to generalize. As a result, any model that is trained on this dataset will be more likely to be overfitted to just the dataset itself. Thus, it will not perform well when given new data.

b. Yes, restricting the maximum value a parameter can take on will reduce overfitting. Limiting the value of the parameters results in lower weights, which in turn results in a less complex model. Less complex models are less likely to overfit the data.

c. No, training the model for longer will not mitigate overfitting. The model will actually become overfitted if we train it for too long. It will start to memorize the data. As a result, we must look at the test error and train error in order to decide when to stop training. We stop training when the test error improvement starts to stagnate, and this is called early stopping. If we continue to train the model, the test error will actually start to increase, thus resulting in overfitting.

d. No, training a model with more parameters does not reduce overfitting. More parameters results in a more complex model, which in turn results in a model that is more likely to overfit.

e. Yes, randomly setting 50% of the nodes in a neural network to 0 will likely reduce overfitting. This is known as the process of dropout in neural network training. While it may sound counterintuitive, dropping some nodes in a neural network actually strengthens it, because the network must learn some of the lost information in the dropped nodes, through the nodes that are still alive.

f. No, sparse features will not reduce overfitting. Adding additional features in general will add complexity to the model. Spare features are defined as those where most of the dataset's feature values are zero or null values. In particular, sparse features add little predictive power to the model but instead add complexity to the model, thus leading to overfitting.

g. Yes, initializing parameters randomly would prevent overfitting. Typically, a model is trained multiple times and the best results are selected to become the final model. Training a model typically involves using an optimization process known as gradient descent, where it finds a particular function's local minimum in order to find the best weights. However, there may be many local minimums,

and some may be much better than others. If we only start training with the initial weights at zero, we may not find a good model because it will always end up at the same local minimum. If we randomize the weights, we may discover better local minimums and thus a better model that is less likely to overfit.

h. No, using a GPU or a CPU does not affect whether or not a model will overfit. While a GPU is more helpful by making the training process more efficient, it has no effect on preventing overfitting.

Raymond Lin
304937942

4. K-Nearest Neighbors
    a. It is important to normalize data because we need features to be weighted the same. For example, if the range of one feature was from 0 to 1, and the range of another feature was from 0 to 1000, then the KNN algorithm would depend much more on the feature with the larger spread. We must normalize the data so that each feature is weighted the same amount and, thus the prediction will depend equally on both features.

    b. An odd value is typically used to avoid ties. If we use even values for K, some data points will have the same number of samples with a positive label as the number of samples with a negative label.

    c.

Data Point	**x**	**y**	**L**
1	0.167	0.461	0
2	0.083	0.987	0
3	0.250	0.921	0
4	0.083	0.658	0
5	0.167	0.658	0
6	1.000	0.526	1
7	0.750	0.539	1
8	0.917	0.513	1
9	0.083	1.000	1

    d.

Test Data Point	**x**	**y**	**x_normalized**	**y_normalized**
a	0.1	760	0.083	1.000
b	0.2	700	0.167	0.921
c	0.6	200	0.500	0.263

Raymond Lin
304937942

Test Data Point	**x_normalized**	**y_normalized**	**Train Data Point**	**Distance**
a	0.083	1.000	1	0.5455061869
			2	0.013
			3	0.1847430648
			4	0.342
			5	0.3521647342
			6	1.032262079
			7	0.8108082387
			8	0.9657768893
			9	0
b	0.167	0.921	1	0.46
			2	0.1068269629
			3	0.083
			4	0.2760887538
			5	0.263
			6	0.9219078045
			7	0.6970028694
			8	0.8537938861
			9	0.1153126186
c	0.500	0.263	1	0.3874183785
			2	0.8355028426
			3	0.7038920372
			4	0.5743814064
			5	0.5166372035
			6	0.5649504403

			7	0.3723922663
			8	0.4861985191
			9	0.8467927728

K = 3

Test Data Point	**Closest Data Points**	**Closest Data Point Labels**	**Prediction**
a	9, 2, 3	1, 0, 0	0
b	3, 2, 9	0, 0, 1	0
c	7, 1, 8	1, 0, 1	1

e. K = 1

Test Data Point	**Closest Data Point**	**Closest Data Point Labels**	**Prediction**
a	9	1	1
b	3	0	0
c	7	1	1

f. K = 5

Test Data Point	**Closest Data Point**	**Closest Data Point Labels**	**Prediction**
a	9, 2, 3, 4, 5	1, 0, 0, 0, 0	0
b	3, 2, 9, 5, 4	0, 0, 1, 0, 0	0
c	7, 1, 8, 5, 6	1, 0, 1, 0, 1	1

g. The issue with using a low K value is that it has the potential to overfit. Each data point has its own region space. It's possible that a single data point has a particular label, while every neighbour around that point has a different label. Even if a test data point fell into the region of the data point with a different label, it would be classified the same as that data point, when in reality that point probably should have been classified the same as all of its neighbours.

Raymond Lin
304937942

h. The issue with using a high K value is that it will underfit the data. A higher K value results in smoother decision boundaries. It will try to generalize too much, and thus it will result in a model that is not predictive enough. Take the extreme case as an example. If we use a K value that is equal to the number of data points in the entire dataset, all test points will be classified as the majority class and not the other class. Thus, this model is underfitted and is not expressive enough of the trend of the population.

i. We could quantify the performance of the KNN by using binary classification metrics, such as accuracy, precision, recall and F1 score. These values give us quantitative values that can evaluate the performance of the model.

j. We could test the model with different values of K, in order to see which value performs the best on the dataset. We can compare the classification metrics mentioned above for each of the different values of K. Then, we can select the model with the K value that had the highest metric scores.

   Furthermore, we could use cross validation for each value of K. We split the dataset into N partitions, and then we train on N-1 partitions and test on the left out partition. We do this until all N partitions have been used as a test set. We can take the average of all the scores as the final score for the particular value of K.

Raymond Lin
304937942

5. Principal Component Analysis

a. Yes, PCA would work well on data with a linear structure. This is because we can reduce the dimensionality of the data by taking the principal component along which the data lies. Because there is low variation on the other features, we can eliminate them and have the data only depend on the most important feature.

b. No, PCA would not work well on data with a hyperbolic structure. PCA is used to eliminate features that have a low impact on the prediction of the label. In the linear case, we can see that features that don't contribute to the prediction are the unwanted features. However, in the case of hyperbolic structured data, more than two features impact the prediction. As a result, we should not use PCA.

c. No, PCA should not be used on a non-normalized dataset. If the data is not normalized, we cannot be certain which features have more impact or influence than others. For example, a feature with a very high range of values will dominate a feature with a low range of values.

d. No, PCA should not be used on a dataset where each of the features are independent of one another. This means that each feature provides some new information that cannot be extracted from other features. Thus, every feature is important and we cannot discard any one of them.

Raymond Lin
304937942

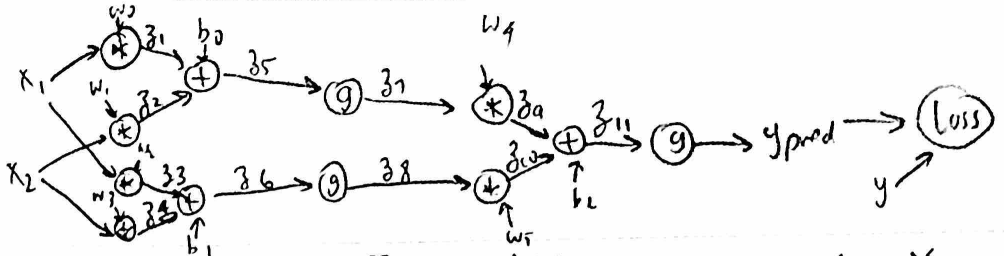
6. Artificial Neural Networks

a.

Test Points	x1	x2	y
1	0	0	0
2	1	1	1
3	0	1	1

Parameter/ Weight	Initial Value	After 1	After 2	After 3 (Final)
b0	1	1.000225236	0.9976907078	0.9972529405
b1	-6	-5.999994349	-6.000352854	-6.012685799
b2	-3.93	-3.929427207	-3.934501555	-3.946849881
w0	3	3	2.997465472	2.997465472
w1	4	4	3.997465472	3.997027705
w2	6	6	5.999641495	5.999641495
w3	5	5	4.999641495	4.98730855
w4	2	2.000418745	1.997791795	1.985666628
w5	4	4.000001416	3.995018336	3.988846369

(work on next page)



6. a)

Parameter	Initial Value
b_0	1
b_1	-6
b_2	-3.93
w_0	3
w_1	4
w_2	6
w_3	5
w_4	2
w_5	4

	x_1	x_2	y
①	0	0	0
②	1	1	1
③	0	1	1

$$g(z) = \frac{1}{1+e^{-z}}$$

$$g((g(x_1 \cdot w_0 + x_2 \cdot w_1 + b_0)) \cdot w_4 + (g(x_1 \cdot w_2 + x_2 \cdot w_3 + b_1)) \cdot w_5 + b_2) = y_{pred}$$

Data point:

① $\frac{\partial Loss}{\partial w_5} = \frac{\partial loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial z_{10}} \cdot \frac{\partial z_{10}}{\partial w_5}$

$= (y - y_{pred}) \cdot y_{pred} \cdot (1 - y_{pred}) \cdot 1 \cdot z_8 = -1.15 \times 10^{-5}$

$w_5 = w_5 - \alpha \frac{\partial Loss}{\partial w_5} = 4.00000115$

$\frac{\partial Loss}{\partial w_4} = \frac{\partial loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial z_9} \cdot \frac{\partial z_9}{\partial w_4}$

$= (y - y_{pred}) \cdot y_{pred} \cdot (1 - y_{pred}) \cdot 1 \cdot z_7 =$

$w_4 = w_4 - \alpha \frac{\partial Loss}{\partial w_4} = 2.00042$

$\frac{\partial Loss}{\partial w_0} = \frac{\partial loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial z_9} \cdot \frac{\partial z_9}{\partial z_7} \cdot \frac{\partial z_7}{\partial z_5} \cdot \frac{\partial z_5}{\partial z_1} \cdot \frac{\partial z_1}{\partial w_0}$

$= 0$

$z_1 = 0$
$z_2 = 0$
$z_3 = 0$
$z_4 = 0$
$z_5 = 1$
$z_6 = -6$
$z_7 = 0.731$
$z_8 = 0.002$
$z_9 = 1.462$
$z_{10} = 0.009$
$z_{11} = -2.459$
$y_{pred} = 0.079$

w_0, w_1, w_2, w_3 remain constant

$\frac{\partial z_8}{\partial z_6} = 0.0019$ $\frac{\partial z_7}{\partial z_5} = z_5(1-z_5)\frac{\partial z_5}{\partial z_1} = 0.197$ $\frac{\partial z_5}{\partial z_1} = 1$ $\frac{\partial z_{11}}{\partial z_{10}} = \frac{\partial z_{11}}{\partial z_9} = 1$ $\frac{\partial loss}{\partial y_{pred}} = -0.079$

$\frac{\partial y_{pred}}{\partial z_{11}} = 0.073$

$\frac{\partial z_1}{\partial b_2} \frac{\partial z_6}{\partial b_1} = \frac{\partial z_5}{\partial b_0} = 1$ $\frac{\partial z_1}{\partial w_0} = 0$ $\frac{\partial z_7}{\partial z_7} = 2$ $\frac{\partial z_{10}}{\partial z_8} = 4$

$z_7 = w_4$

$$\frac{\partial Loss}{\partial b_0} = \frac{\partial Loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial z_9} \cdot \frac{\partial z_9}{\partial z_7} \cdot \frac{\partial z_7}{\partial z_5} \cdot \frac{\partial z_5}{\partial b_0}$$

$$= -0.079 \cdot 0.073 \cdot 1.2 \cdot 0.197 \cdot 1$$

$$= -0.0023$$

$$b_0 = b_0 - \alpha \frac{\partial Loss}{\partial b_0} = 1 + \alpha 0.0023 = 1.00023$$

$$\frac{\partial Loss}{\partial b_1} = \frac{\partial Loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial z_{10}} \cdot \frac{\partial z_{10}}{\partial z_8} \cdot \frac{\partial z_8}{\partial z_6} \cdot \frac{\partial z_6}{\partial b_1}$$

$$= -0.079 \cdot 0.073 \cdot 1.4 \cdot 0.0019 \cdot 1$$

$$= -0.000044$$

$$b_1 = b_1 - \alpha \frac{\partial Loss}{\partial b_1} = -5.999956$$

$$\frac{\partial Loss}{\partial b_2} = \frac{\partial Loss}{\partial y_{pred}} \cdot \frac{\partial y_{pred}}{\partial z_{11}} \cdot \frac{\partial z_{11}}{\partial b_2}$$

$$= 0.0058$$

$$b_2 = b_2 - \alpha \frac{\partial Loss}{\partial b_2} = -3.9306$$

New Weights

$b_0 = 1.00023$

$b_1 = -5.999956$

$b_2 = -3.9306$

$w_0 = 3$

$w_1 = 4$

$w_2 = 6$

$w_3 = 5$

$w_4 = 2.00042$

$w_5 = 4.0000115$

Raymond Lin
304937942

More Work From Google Sheets

Helpers for Training

(Each z value represents the output after a particular node - see previous page)

Values	Data point 1	Data point 2	Data point 3
z1	0	-3.929427207	0
z2	0	3	2.997465472
z3	0	4	0
z4	0	6	5.999641495
z5	1	0.07079802849	3.99515618
z6	-6	4.000005651	-0.0007113589166
z7	0.7310585786	0.5176921178	0.981928035
z8	0.002472623157	0.9820138899	0.4998221603
z9	1.462117157	1.035601017	1.961687772
z10	0.009890492627	3.92805695	1.996798695
z11	-2.45799235	1.03423076	0.02398491123
ypred (raw)	0.07885604513	0.7377352943	0.5059959404

Loss

(This table represents the loss after back propagation of each weight/parameter)

Parameter	After 1	After 2	After 3
b0	-0.002252357686	0.02534527965	0.004377673119
b1	-0.0000565119427 6	0.003585050555	0.1233294512
b2	-0.005727927213	0.05074348139	0.1234832548
w0	0	0.02534527965	0
w1	0	0.02534527965	0.004377673119
w2	0	0.003585050555	0
w3	0	0.003585050555	0.1233294512
w4	-0.004187450327	0.02626950035	0.1212516698
w5	-0.0000141630054 7	0.04983080355	0.06171966718

Raymond Lin
304937942

Predictions

(This table was used to predict the label of each data point after training with the same three data points)

Values	Data point 1	Data point 2	Data point 3
z1	0	-3.946849881	0
z2	0	2.997465472	2.997465472
z3	0	3.997027705	0
z4	0	5.999641495	5.999641495
z5	0.9972529405	0.04786853163	3.994718413
z6	-6.012685799	3.983983401	-0.01304430404
z7	0.7305181313	0.5119648483	0.981920265
z8	0.002441530177	0.9817286987	0.4967389702
z9	1.450565474	1.016591514	1.949766302
z10	0.009738888782	3.915964955	1.981415438
z11	-2.486545518	0.9857065884	-0.01566814139
ypred (raw)	0.07680678826	0.7282390596	0.4960830448
ypred	0	1	0
(threshold = 0.5)			

b. Confusion Matrix

	Predicted 0	**Predicted 1**
Actual 0	1 (True negative)	0 (False positive)
Actual 1	1 (False negative)	1 (True positive)

$$Accuracy = \frac{True\ negatives + True\ positives}{True\ negatives + False\ negatives + True\ positives + False\ positives} = \frac{1+1}{1+1+1+0} = \frac{2}{3} = 66.667\%$$

c. $$Precision = \frac{True\ positives}{True\ positives + False\ positives} = \frac{1}{1+0} = 1 = 100\%$$

Raymond Lin
304937942

d. $Recall = \frac{True\ positives}{True\ positives + False\ negatives} = \frac{1}{1+1} = \frac{1}{2} = 50\%$

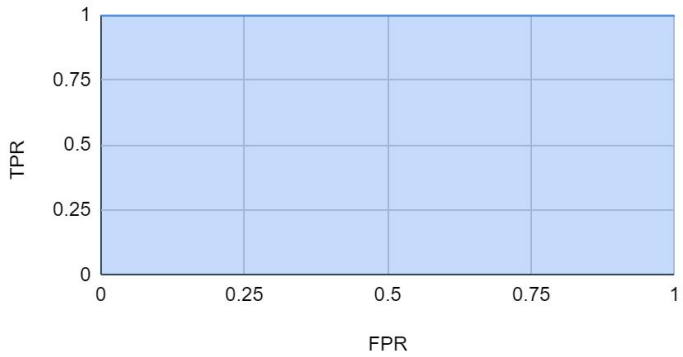
e. $F1\ Score = 2 \times (\frac{Precision \times Recall}{Precision + Recall}) = \frac{2 \times 0.5}{1.5} = 66.67\%$

f. True Positive Rate - False Positive Rate

Threshold	**True Positive Rate**	**False Positive Rate**
0.0	1.0	1.0
0.1	1.0	0.0
0.2	1.0	0.0
0.3	1.0	0.0
0.4	0.5	0.0
0.5	0.5	0.0
0.6	0.5	0.0
0.7	0.5	0.0
0.8	0.5	0.0
0.9	0.0	0.0
1.0	0.0	0.0

$$True\ Positive\ Rate = \frac{True\ positives}{True\ positives + False\ negatives}$$
$$False\ Positive\ Rate = \frac{False\ positives}{False\ positives + True\ negatives}$$

ROC Curve



Raymond Lin
304937942

g. $AUROC = 1$

7. N/A

Raymond Lin
304937942

8. (Money) ball So Hard

a. The movie, *Moneyball*, depicts the Oakland Athletics baseball team of the Major League Baseball league in the 2002 season and how the front office (mainly the general manager, Billy Beane) constructed a very successful team, after losing a number of key players from the previous season. This process was especially difficult because the Oakland A's organization had a much lower budget than many other teams in the league, and it caused many conflicts between Beane and the other members of the front office. Beane hired Peter Brand, who was a Yale graduate with a degree in economics. Beane chose to listen to Peter's advice over his front office in constructing a team, because of Peter's statistical analysis of the available free agents. The analysis mainly included how to get the maximum number of runs in the season for the lowest price, among other factors, such as the ability to get to base etc… The front office disagreed with Beane, because they believed that years of baseball scouting experience and intuition was more valuable than a statistical analysis. bThey argued that many of these new players had other issues which could potentially become detrimental. For example, some of them were old, some of them were in poor health or had a history of injuries, some of them had a history of off-court issues, and some of them seemed like they simply weren't talented enough. While the team did not perform well during the early parts of the season, it surprisingly came back and ended the season as one of the top teams in the league.

The way I would go about solving this problem is to first gather data on strong baseball teams from the last couple of seasons. We could then use a machine learning algorithm to find the factors that are the best predictors of a strong baseball team. According to the movie, the number one factor in a strong baseball team is the number of runs that it is able to achieve over the course of a season. After that, we could gather a dataset of all the players in the league and determine how much each player is worth based on their ability to score runs. For example, a possible metric we could use is price per run. As a result, we can see which free agents are currently undervalued, or in other words, worth a good price. From here, we can pick players that fit under our budget. Furthermore, we can confirm our team's successes by running an algorithm on the team as a dataset where each data point is a player and the features are the player's statistics (runs per game, outs per game, bats per game etc…). We can use linear regression to predict the team's record. We first run the same algorithm on existing teams as a baseline, with their records as the label. Then, we can run the algorithm on our existing team and predict their record. We can repeat this using a different combination of free agents in order to construct the best possible team.

b. This method has its weaknesses, because it is only based on numbers. In the movie, other members of the Oakland A's front office are skeptical of Beane's approach because it does not take into account the off-court history of the players and other intangibles. This is still a legitimate concern, because if players get into trouble outside of the league, it could impact the performance of the team. We could theoretically gather data on each player's off-court history. We could gather statistics such as how many times he has been in trouble with the law, how many times he has gotten into drug or alcohol related incidents etc… These statistics could then be added as features into the machine learning algorithms we listed above. There are many other factors that simply cannot be predicted. For example, players can simply have a bad season. Additionally, team chemistry matters. If the team does not get along with each other socially, they may also not work well together on the court. This is something that cannot be predicted.