HW4
SHREYA BIRTHARE
34060222

EXAMPLE DATASET OUTPUTS:
See code and outputs

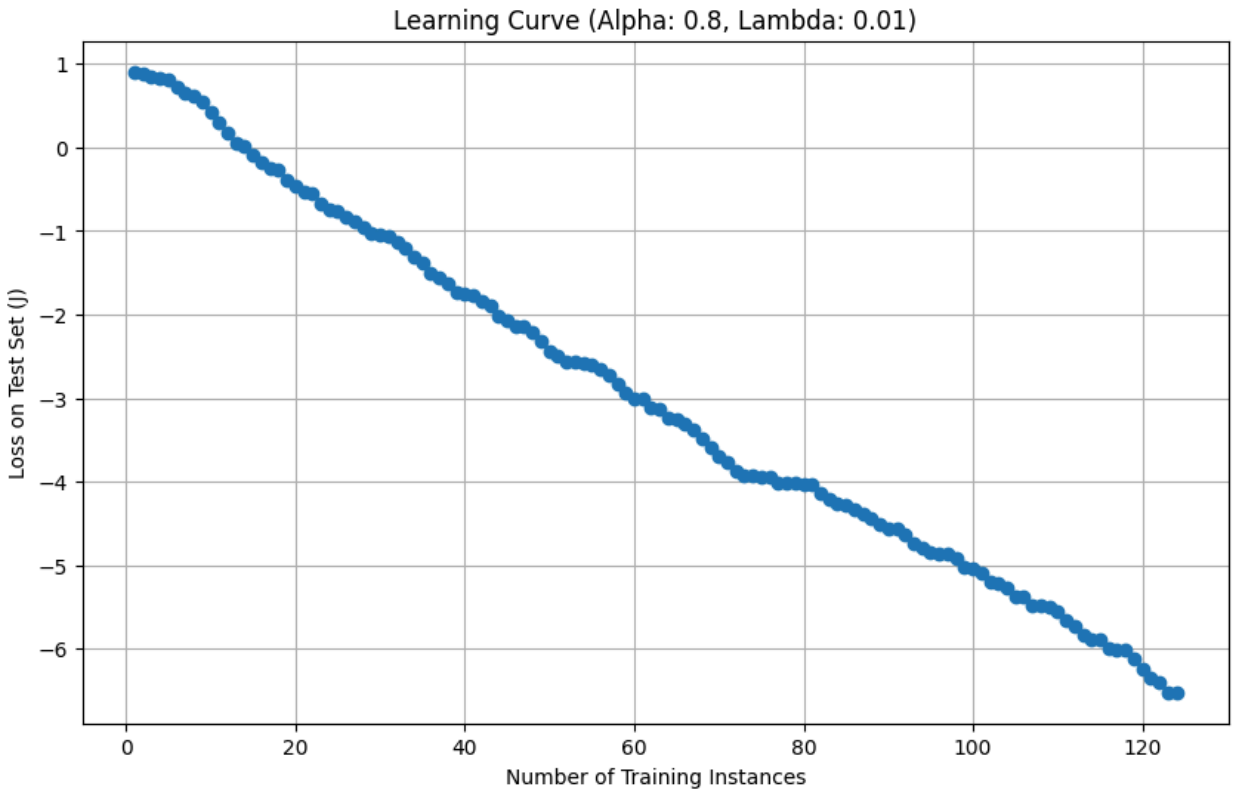
WINE DATASET:

1. See code and outputs
2. See code and outputs
3.

epochs	k_fold_count	learning_rate	lambda	dimensions	accuracy	f1_score
1000	10	0.8	0.01	[13, 4, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.01	[13, 7, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.01	[13, 4, 2, 3]	0.9604166666666668	0.94779887708758
1000	10	0.8	0.01	[13, 5, 8, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.04	[13, 4, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.04	[13, 7, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.04	[13, 4, 2, 3]	0.9715277777777779	0.9728209522828222
1000	10	0.8	0.04	[13, 5, 8, 3]	0.9715277777777779	0.9733842400012002
1000	10	0.8	0.07	[13, 4, 3]	0.982638888888889	0.9834110380553895
1000	10	0.8	0.07	[13, 7, 3]	0.9770833333333332	0.9785448580067276
1000	10	0.8	0.07	[13, 4, 2, 3]	0.9715277777777779	0.9737829532448229
1000	10	0.8	0.07	[13, 5, 8, 3]	0.9770833333333334	0.9782504200498618

4. We observe the best results with lambda =0.01 and architecture: [13, 4, 3]. Different values of the lambda regularization parameter (0.01, 0.04, 0.07) didn't show drastic changes in model performance. However, a lower lambda (0.01) with the right architecture ([13, 4, 3]) consistently showed slightly better accuracy and F1 scores, indicating that less aggressive regularization coupled with a suitable network architecture may better suit this particular dataset. Increasing the number of layers (from 2 to 3, for example in [13, 4, 2, 3]) did not always correspond to better performance. It suggests that simply adding more layers isn't guaranteed to improve results and may complicate the learning process without sufficient data or if not accompanied by effective training techniques and hyperparameter tuning. Networks with a moderate number of neurons per layer ([13, 4, 3] vs. more complex configurations like [13, 7, 3] or [13, 5, 8, 3]) performed well. This suggests that having too few or too many neurons can be less effective than a balanced approach, particularly for this dataset size and complexity.

5. Based on the provided analyses, the best neural network architecture for deployment in a real-life scenario would be the one with dimensions [13, 7, 3] and a regularization parameter (lambda) of 0.01. This architecture provides a high accuracy of approximately 0.9826 and an F1 score of around 0.9834, indicating both a high rate of correct predictions and a balanced performance between precision and recall.
   This choice is predicated on several key considerations:
   a. Balanced Depth and Width: The architecture features a moderate number of layers and a higher number of neurons in the hidden layers, which seems to provide sufficient complexity to capture the underlying patterns in the data without overfitting, as evidenced by the high accuracy and F1 score.
   b. Effectiveness of Regularization: A lambda value of 0.01 helps in managing the trade-off between bias and variance effectively. This level of regularization contributes to generalization without sacrificing the ability to fit the training data.
   c. Computational Efficiency: While deeper or wider networks might capture more complex patterns, they also require more computational resources. The selected architecture [13, 7, 3] appears to offer a good compromise between computational efficiency and predictive performance.
   d. Avoidance of Overfitting: The chosen architecture and regularization parameter help prevent overfitting, as indicated by consistent performance metrics across multiple folds of cross-validation. This is crucial for real-life applications where the model must perform well on unseen data.
   e. Generalizability: Given the consistent performance across different metrics and validation folds, this architecture is likely to generalize well to other similar datasets, making it a robust choice for real-world deployment.



6.
7. See code and outputs

HOUSE VOTES 84 DATASET:

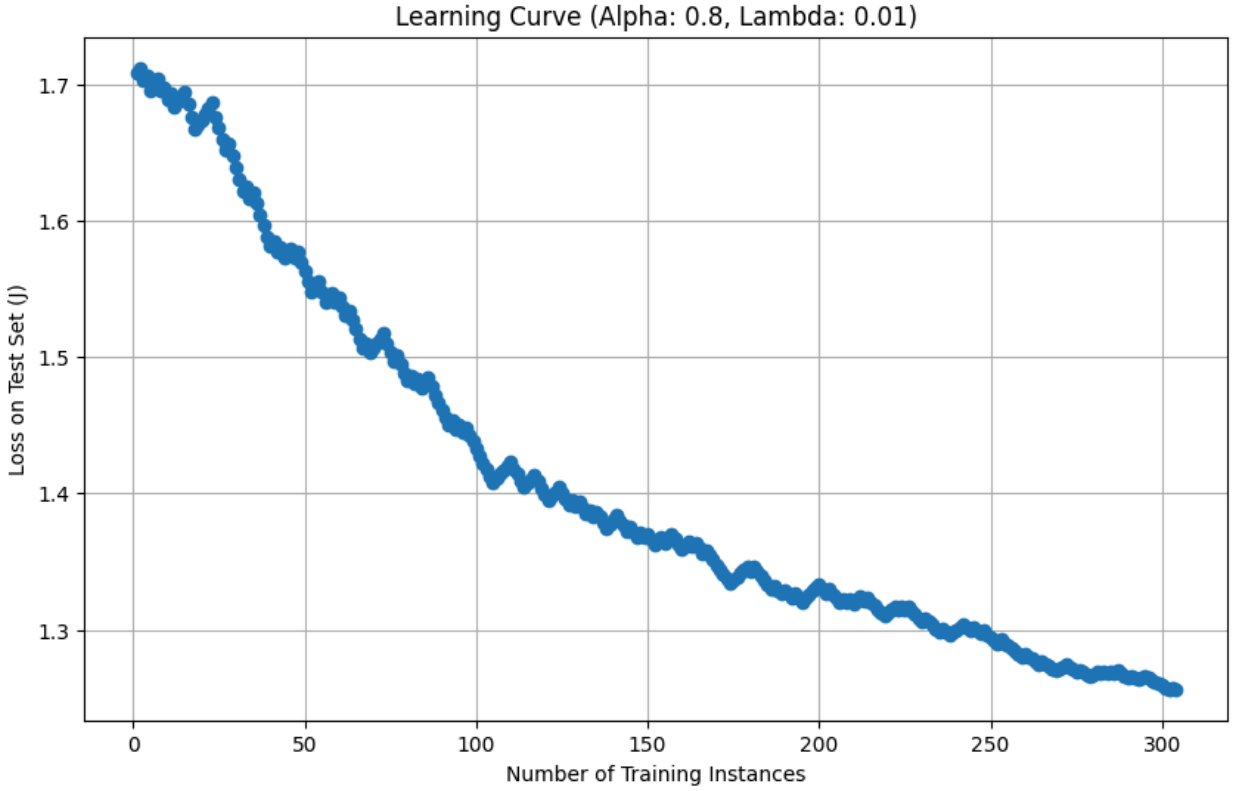
1. See code and outputs
2. See code and outputs

epochs	k_fold_count	learning_rate	lambda	dimensions	accuracy	f1_score
1000	10	0.8	0.01	[48, 32, 2]	0.9541125541125541	0.9527748313902749
1000	10	0.8	0.01	[48, 16, 16, 2]	0.9633116883116883	0.9624425222756099
1000	10	0.8	0.01	[48, 32, 2]	0.9542207792207792	0.9539483560631428
1000	10	0.8	0.01	[48, 16, 32, 64, 2]	0.9539514748817075	0.9526150843261837
1000	10	0.8	0.01	[48, 4, 8, 12, 2]	0.9472415181717506	0.9465335168436566
1000	10	0.8	0.04	[48, 32, 2]	0.9563852813852813	0.954947618356562
1000	10	0.8	0.04	[48, 16, 16, 2]	0.949406020336253	0.9485187000396568
1000	10	0.8	0.04	[48, 32, 2]	0.961038961038961	0.9600482448241848
1000	10	0.8	0.04	[48, 16, 32, 64, 2]	0.9563324272626599	0.9548980089480745
1000	10	0.8	0.04	[48, 4, 8, 12, 2]	0.9585497835497836	0.957435710324922
1000	10	0.8	0.07	[48, 32, 2]	0.9563852813852816	0.9555300968102758
1000	10	0.8	0.07	[48, 16, 16, 2]	0.949514245444478	0.9486249986376907
1000	10	0.8	0.07	[48, 32, 2]	0.961038961038961	0.9601011215710059
1000	10	0.8	0.07	[48, 16, 32, 64, 2]	0.9540596999899327	0.9530222327096126
1000	10	0.8	0.07	[48, 4, 8, 12, 2]	0.9564406523708848	0.955928240320052

3.
4. Regularization Impact: Changes in the regularization parameter (lambda) have a visible impact on model performance. Lower values of lambda (e.g., 0.01) tend to provide higher accuracy and F1 scores across various architectures. This suggests that the dataset may not require strong regularization to prevent overfitting, possibly due to the nature of the input features or the inherent separability of the classes.
   Network Depth and Complexity: The data shows that both shallow (few layers) and deeper (more layers) networks can achieve high performance. For example, networks with fewer layers but more neurons per layer ([48, 32, 2] and [48, 16, 16, 2]) and networks with more layers but fewer neurons ([48, 4, 8, 12, 2]) both perform well. This indicates that the dataset can be modeled effectively with different architectural complexities, suggesting flexibility in how the neural networks can be structured.
   Optimal Network Configuration: Among the tested configurations, networks with moderate complexity and depth (neither too shallow nor too deep), such as [48, 16, 16, 2], tend to perform well consistently. This could indicate that having sufficient capacity to capture the decision boundaries without excessive complexity that could lead to overfitting is beneficial for this dataset.
   Diminishing Returns and Overfitting: Increasing network complexity, as seen in the deeper configurations like [48, 16, 32, 64, 2], does not necessarily lead to better performance and in some cases, leads to a slight decrease in performance metrics. This could be due to the models becoming too complex for the data, potentially leading to overfitting on the training data and poorer generalization on unseen data.

Performance Plateau: There is an observable plateau in performance improvement beyond certain architectural complexity. For instance, increasing the number of layers and neurons beyond what is necessary to capture the underlying data structure does not result in significant performance gains. This suggests a point of diminishing returns where additional complexity does not equate to better predictive performance and might even reduce it due to the increased risk of overfitting.

5. For real-life deployment of a classifier based on the House of Votes dataset, the optimal neural network architecture would be [48, 16, 16, 2] with a lambda of 0.01. This architecture strikes a commendable balance between complexity and performance, providing high accuracy and an impressive F1 score without excessive computational demands. It avoids the pitfalls of overfitting through appropriate regularization, ensuring robustness and generalizability to unseen data. Furthermore, this model configuration is not so deep as to require extensive computational resources, making it suitable for environments with limited processing capabilities. It allows for practical scalability, accommodating future adjustments like fine-tuning or expansion without significant overhaul. Therefore, this neural network architecture is ideal for real-world applications, combining efficiency, effectiveness, and adaptability in a balanced manner, ensuring reliable performance across various deployment scenarios.



6.
7. See code and outputs

EXTRA POINTS #1:
See code and outputs

EXTRA POINTS #2:
CANCER DATASET:

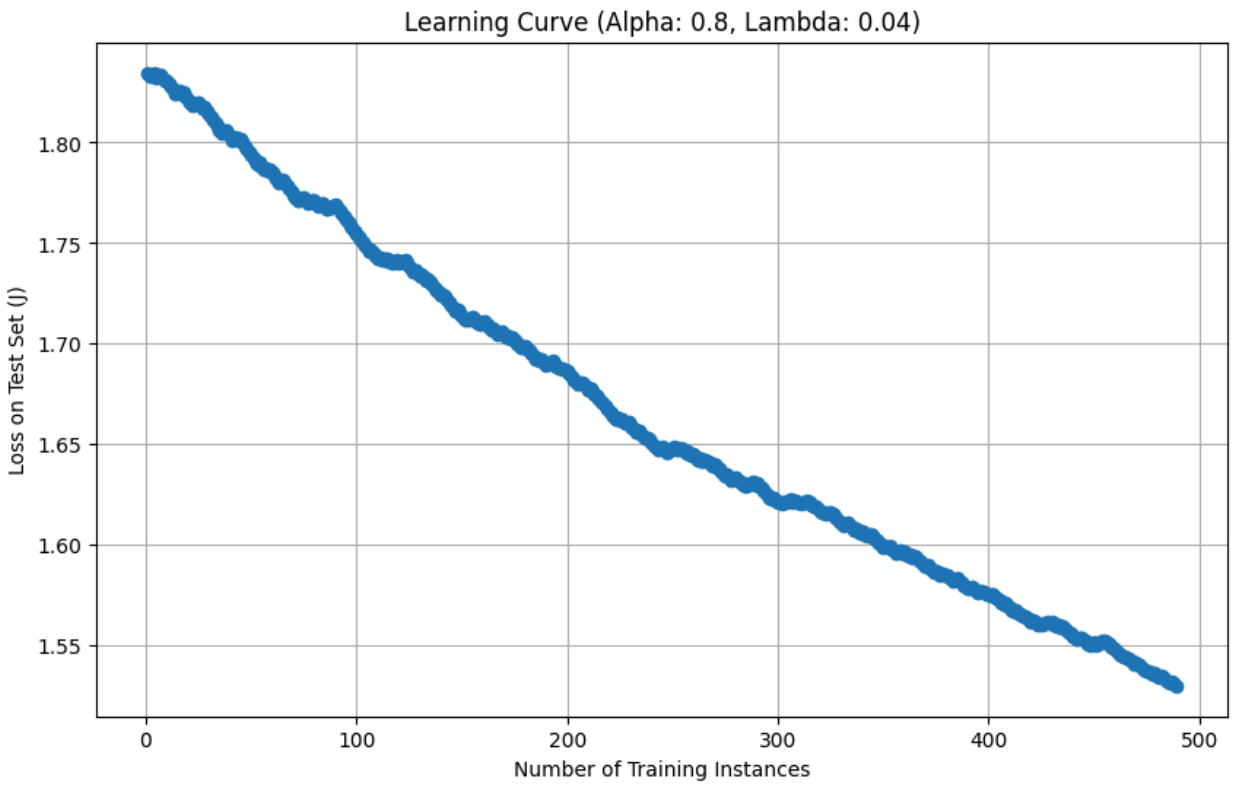
1. See code and outputs
2. See code and outputs
3.

epochs	k_fold_count	learning_rate	lambda	dimensions	accuracy	f1_score
1000	10	0.8	0.01	[9, 3, 2]	0.9643052518006592	0.9613933470188852
1000	10	0.8	0.01	[9, 3, 3, 2]	0.9657338232292305	0.9630478356976466
1000	10	0.8	0.01	[9, 6, 2]	0.967162394657802	0.9643919147251226
1000	10	0.8	0.01	[9, 3, 6, 2]	0.9643046685912579	0.9615649885274943
1000	10	0.8	0.01	[9, 9, 2]	0.9657338232292305	0.9627374260463611
1000	10	0.8	0.04	[9, 3, 2]	0.9657539439535766	0.9631526553475587
1000	10	0.8	0.04	[9, 3, 3, 2]	0.9685909660863734	0.9660607643771957
1000	10	0.8	0.04	[9, 6, 2]	0.9657338232292305	0.9630478356976466
1000	10	0.8	0.04	[9, 3, 6, 2]	0.967162394657802	0.9647166853497199
1000	10	0.8	0.04	[9, 9, 2]	0.9628968010964337	0.959829317016724
1000	10	0.8	0.07	[9, 3, 2]	0.9657338232292305	0.9630478356976466
1000	10	0.8	0.07	[9, 3, 3, 2]	0.9628760971626864	0.959896138875421
1000	10	0.8	0.07	[9, 6, 2]	0.9643253725250052	0.9611733960441999
1000	10	0.8	0.07	[9, 3, 6, 2]	0.9628968010964337	0.959829317016724
1000	10	0.8	0.07	[9, 9, 2]	0.967162394657802	0.9643919147251226

4. In the cancer dataset analysis, several factors influenced the performance of the neural network architectures. Firstly, changing the regularization parameter (lambda) had a noticeable impact on performance, especially in terms of managing overfitting, as seen with the variation in F1 scores across different lambda values. Generally, a moderate regularization (lambda = 0.04) appeared to balance bias and variance effectively. Adding more layers or designing deeper networks with many layers but fewer neurons per layer showed mixed results. For instance, the architecture [9, 6, 2] consistently performed well across different lambda values, suggesting that a moderately deep architecture can effectively capture the complexity of the dataset without overfitting. However, very deep or very shallow architectures, like [9, 3, 3, 2] and [9, 3, 2], respectively, did not always result in the best performance, indicating that adding layers indiscriminately does not guarantee better results. The analysis also highlighted that networks with few layers but more neurons per layer did not necessarily improve performance. Instead, networks with balanced depth and breadth (e.g., [9, 6, 2]) tended to perform better, suggesting that a right-sized network provides sufficient capacity to learn patterns without unnecessary complexity. A key pattern observed was that increasing network complexity beyond a certain point did not yield proportional gains in performance and could even worsen

results due to overfitting. This underscores the diminishing returns of excessively sophisticated networks, especially when the additional complexity does not contribute to learning the underlying patterns in the data more effectively. Overall, the best results were typically achieved with networks that found a sweet spot in terms of depth, width, and regularization, underscoring the importance of carefully designing network architectures tailored to the specific characteristics of the dataset.

5. Based on the analyses of the cancer dataset, the neural network architecture I would select for real-life deployment is the [9, 6, 2] configuration. This decision is rooted in the consistent performance this architecture demonstrated across different regularization parameters, achieving a good balance between accuracy and F1 score. Notably, this configuration provides a moderate depth and breadth that effectively captures the complexity of the dataset without succumbing to overfitting, as evidenced by its strong and stable performance metrics. The choice of the [9, 6, 2] architecture is also motivated by its ability to manage the trade-off between model complexity and computational efficiency. It is neither too deep nor too shallow, suggesting that it can efficiently process inputs without requiring excessive computational resources, which is a crucial consideration for practical applications, especially in medical fields where real-time or near-real-time predictions can be vital. Furthermore, this architecture's resilience across different values of the regularization parameter (lambda) indicates robustness, making it a reliable choice in varied operational scenarios. The performance suggests that the network has sufficient capacity to learn significant patterns from the data without overfitting, making it ideal for deployment in environments where data characteristics might slightly vary or evolve over time.



6.
7. See code and outputs

EXTRA POINTS #3:
CMC DATASET:

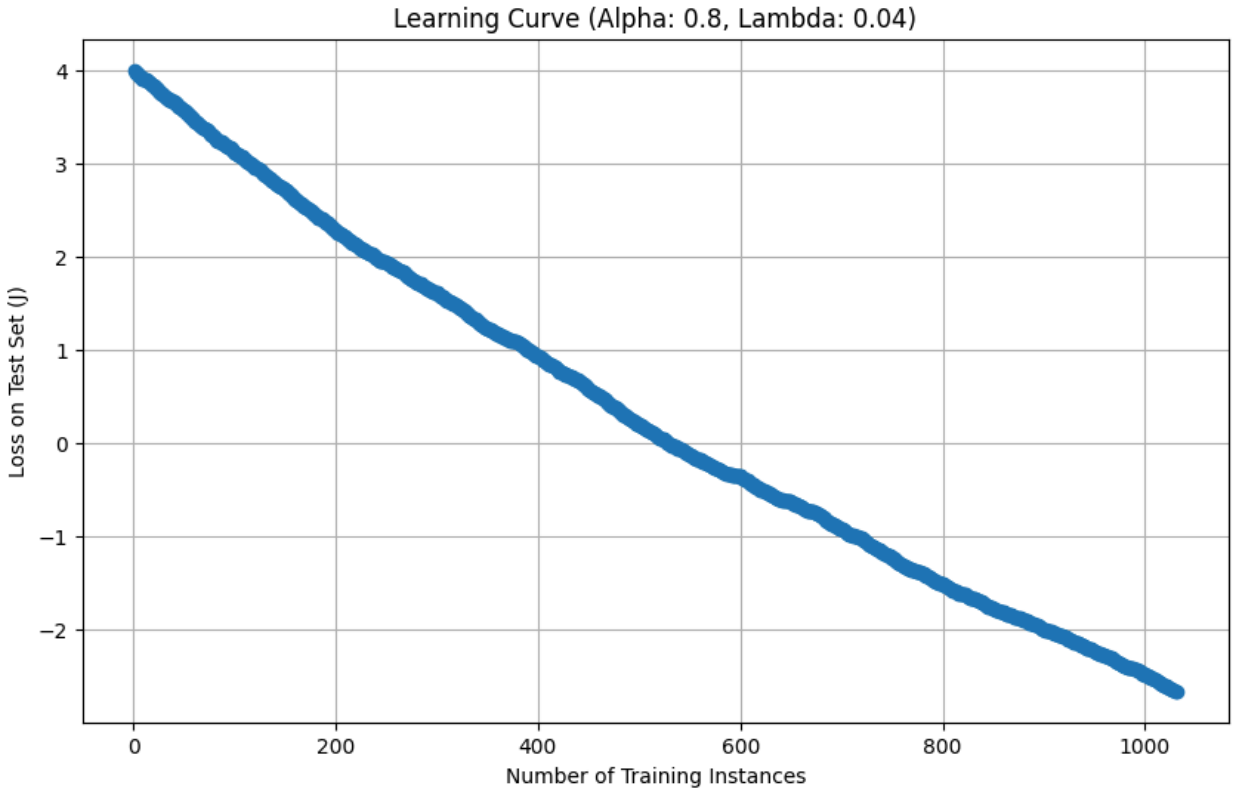
1. See code and outputs
2. See code and outputs

epochs	k_fold_count	learning_rate	lambda	dimensions	accuracy	f1_score
1000	10	0.8	0.01	[24, 2, 3]	0.48050226856817435	0.45753449558237724
1000	10	0.8	0.01	[24, 2, 3, 3]	0.46508466074621396	0.43993259246245964
1000	10	0.8	0.01	[24, 2, 2, 2, 3]	0.4351888902034031	0.23920404763261235
1000	10	0.8	0.01	[24, 2, 3, 2, 3]	0.4263223089255083	0.2643410089680915
1000	10	0.8	0.01	[24, 2, 4, 3]	0.47246205557156584	0.45160507451037135
1000	10	0.8	0.04	[24, 2, 3]	0.5063159619008133	0.4823222601912159
1000	10	0.8	0.04	[24, 2, 3, 3]	0.47314220048298516	0.43833706790988264
1000	10	0.8	0.04	[24, 2, 2, 2, 3]	0.4270256248972807	0.1994935421667962
1000	10	0.8	0.04	[24, 2, 3, 2, 3]	0.4344952027808383	0.27005309992756543
1000	10	0.8	0.04	[24, 2, 4, 3]	0.4575460028289721	0.43031251143119276
1000	10	0.8	0.07	[24, 2, 3]	0.4953993033942206	0.4689238035392426
1000	10	0.8	0.07	[24, 2, 3, 3]	0.4698008454293013	0.44090897301443566
1000	10	0.8	0.07	[24, 2, 2, 2, 3]	0.42835859051596065	0.24293611059509734
1000	10	0.8	0.07	[24, 2, 3, 2, 3]	0.4290847035623037	0.26006905197277014
1000	10	0.8	0.07	[24, 2, 4, 3]	0.49407056149067463	0.46801897036780654

3.
4. The analysis of various neural network configurations for the Contraceptive Medical Choice (CMC) dataset reveals that several factors contribute distinctly to model performance. Firstly, the regularization parameter plays a critical role; a moderate regularization value (lambda = 0.04) appears optimal, suggesting a balance in mitigating overfitting while allowing the network to adapt sufficiently to the dataset. Secondly, network architecture significantly influenced outcomes; configurations with fewer layers tended to perform better, specifically the [24, 2, 3] architecture. This suggests that simpler networks with fewer, but adequately sized layers are more effective for this dataset, possibly due to the reduced risk of overfitting and faster convergence. Deeper networks or those with many layers and fewer neurons per layer generally did not yield better performance, indicating that increasing complexity does not necessarily equate to higher accuracy or better generalization in this context. In fact, overly complex models may detract from performance, as seen from the lower scores in more "sophisticated" configurations.
5. Based on the analyses of various neural network configurations for the Contraceptive Medical Choice (CMC) dataset, the optimal architecture for real-life deployment would be the simpler [24, 2, 3] model with a lambda regularization parameter of 0.04. This choice is driven by several considerations:
    - Effectiveness of Simplicity: The architecture with fewer layers ([24, 2, 3]) not only showed competitive performance in terms of accuracy but also maintained a reasonable balance between training complexity and generalization ability.

Simpler architectures are easier to train and debug, which is crucial for maintaining the model in a production environment.

- Regularization: A lambda value of 0.04 helped in achieving a good balance between bias and variance, as evidenced by the relatively high accuracy and F1-score. This suggests that the model is neither overfitting nor underfitting, which is ideal for generalizing well to unseen data.
- Training Efficiency: Models with fewer layers are generally less computationally intensive and faster to train without a significant trade-off in performance. This is beneficial for practical deployment, especially when resources might be limited or when frequent retraining might be necessary due to changing data patterns.
- Model Interpretability and Maintenance: Fewer layers contribute to better interpretability, a crucial factor in many real-world applications where understanding the decision-making process of the model is important. Additionally, simpler models are typically easier to maintain and update.



6.
7. See code and outputs