Q12. Discuss the strengths and limitations of the K-Nearest Neighbor (K-NN) classifier compared to Fisher's LDA.

K-Nearest Neighbor (K-NN)

- **Idea**: Classify a sample by looking at the majority class among its k nearest neighbors in feature space.
- **Strengths**:
  1. **Simple and intuitive** – no need for complex training.
  2. **Non-parametric** – does not assume any distribution of data.
  3. **Flexible** – can handle non-linear decision boundaries.
  4. **Adaptable** – works for multi-class problems.
- **Limitations**:
  1. **Computationally expensive** – requires storing all data and computing distances for every query.
  2. **Sensitive to irrelevant features** – high-dimensional data may reduce accuracy (curse of dimensionality).
  3. **Choice of k and distance metric matters**.

Fisher's Linear Discriminant Analysis (LDA)

- **Idea**: Finds a projection direction that maximizes class separability by maximizing the ratio of between-class variance to within-class variance.
- **Strengths**:
  1. **Efficient** – gives a linear decision boundary, easy to compute.
  2. **Dimensionality reduction** – projects data into low dimensions while preserving separability.
  3. **Works well with normally distributed data having equal covariance**.
- **Limitations**:
  1. **Assumes linear separability** – not effective if classes overlap non-linearly.
  2. **Assumes equal covariance structure** across classes.
  3. **Sensitive to outliers**.

📌 **Comparison Example**:

- Handwriting recognition:
  - **K-NN** works better since class boundaries are highly non-linear.
  - **LDA** may fail because digits are not linearly separable.

Q13. A single-layer perceptron is trained to classify binary data. Show how decision boundaries are formed.

Single-Layer Perceptron

- A perceptron computes:

$$y = f(w \cdot x + b)$$

where w = weight vector, b = bias, $f(\cdot)$ = step activation function.
- The decision boundary is formed when:

$$w \cdot x + b = 0$$

This boundary is **linear** (line in 2D, plane in 3D, hyperplane in nD).

Illustration

```
Class 1 (●)        Class 2 (○)

● ● ● ● | ○ ○ ○ ○
● ● ● ● | ○ ○ ○ ○
          ^
          decision boundary (line)
```

Key Points

- If data is linearly separable, the perceptron will converge to a correct solution.
- If not linearly separable (like XOR problem), a single-layer perceptron cannot solve it.

📌 **Example**:
- For spam vs non-spam emails, if features like “contains money,” “contains offer” can be linearly separated, a perceptron works.

Q14. Explain how dividing a dataset into training and testing sets helps in evaluating classifier performance.

Why Divide Data?

In machine learning, we want the classifier to **generalize** beyond the training data. If we only check performance on training data, the model might just **memorize** (overfitting).

- **Training set**: Used to adjust model parameters (weights in neural networks, thresholds in decision rules).
- **Testing set**: Kept separate, used to evaluate final model accuracy.

Benefits

1. **Unbiased Evaluation** – prevents over-optimistic accuracy due to memorization.
2. **Generalization Measurement** – ensures the model works on unseen data.
3. **Error Estimation** – provides realistic estimate of misclassification rate.

📌 **Example**:

- Handwritten digit dataset (MNIST).
- If we train on all 60,000 samples and also test on them, accuracy may seem 100%.
- But when tested on unseen 10,000 test digits, accuracy may drop to 95% → indicating real performance.

Best Practices

- Split ratio: 70–80% training, 20–30% testing.
- Use **cross-validation** for robust evaluation.

Q15. How do standardization and normalization of features impact the performance of a multi-layer perceptron classifier?

Standardization

- Transforms features to have **zero mean** and **unit variance**:

$$x' = \frac{x - \mu}{\sigma}$$

Normalization

- Rescales data into a fixed range, often [0,1]:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Impact on MLP

1. **Improves Convergence**: Gradient descent optimizers work better when features are on similar scales.
2. **Avoids Bias Towards Larger Features**: Without scaling, features with larger numerical ranges dominate the weight updates.
3. **Stabilizes Training**: Prevents exploding/vanishing gradients.
4. **Better Generalization**: Produces smoother decision boundaries.

📌 **Example**

- Suppose one feature = *age* (range 20–60) and another = *income* (range 20,000–200,000).
- If left unscaled, income dominates weight updates.
- After normalization, both features contribute fairly, leading to better classification of customer segments.