1 . Design a complete k-NN based recommendation system for an online bookstore.

Include:

- **Feature selection and preprocessing**
- **Distance metric selection and justification**
- **Algorithm implementation details**
- **Performance optimization techniques**

Solution

Feature Selection & Preprocessing

- User Features: Age, gender, past purchases, browsing history, ratings given.
- Book Features: Genre, author, price, publication year, average rating, number of pages.
- Preprocessing:
  - Normalize numerical features (e.g., Min-Max scaling for price, pages).
  - One-hot encode categorical features (e.g., genre, author).
  - Handle missing ratings using mean imputation.
  - Use TF-IDF for book descriptions (if included).

Distance Metric Selection

- Cosine Similarity: Best for high-dimensional sparse data (e.g., user-book interactions).
- Euclidean Distance: Suitable for normalized numerical features.
- Jaccard Similarity: Useful for binary features (e.g., liked/disliked).

Algorithm Implementation

1. User-Item Matrix: Construct a matrix where rows are users and columns are books (values = ratings).
2. Neighbor Selection: For a target user, find the top-kk similar users using the chosen distance metric.
3. Recommendation: Predict ratings for unread books by averaging neighbors' ratings (weighted by similarity).

Performance Optimization

- KD-Trees/Ball Trees: For efficient nearest-neighbor search in high dimensions.
- Dimensionality Reduction: PCA for reducing feature space.
- Locality-Sensitive Hashing (LSH): Approximate nearest neighbors for scalability.
- Parallelization: Use Spark for distributed k-NN on large datasets.

2. Implement a locally weighted regression algorithm for predicting house prices.

Design should include:

- **Kernel function selection**
- **Feature engineering**
- **Weight calculation method**
- **Cross-validation approach**

Solution

Kernel Function Selection

- **Gaussian Kernel**: $w_i = \exp\left(-\frac{(x_i - x)^2}{2\tau^2}\right)$, where τ (bandwidth) controls weighting.
- **Justification**: Smoothly weights nearby points more heavily.

Feature Engineering

- **Numerical**: Square footage, number of bedrooms/bathrooms, age of property.
- **Categorical**: Neighborhood (one-hot encoded), proximity to amenities.
- **Polynomial Features**: Include interaction terms (e.g., bedrooms × bathrooms).

Weight Calculation

- Weights are computed per query point x, emphasizing training points near x.

Cross-Validation Approach

- **Leave-One-Out (LOO) CV**: Fit the model on all but one data point and validate on the held-out point.
- **Bandwidth Tuning**: Use grid search to optimize τ for minimal MSE.

3. Create a case-based reasoning system for medical diagnosis with:

- **Case representation structure**
- **Similarity metrics**
- **Case adaptation rules**
- **Case base maintenance strategy**

Solution

Case Representation

- Attributes: Symptoms (fever, cough), lab results (WBC count), patient demographics (age, gender).
- Outcome: Diagnosis (e.g., "flu", "pneumonia").

Similarity Metrics

- Numerical: Euclidean distance for lab results.

- Categorical: Overlap coefficient for symptoms.
- Composite Similarity: Weighted sum of individual similarities.

Case Adaptation Rules

- Reuse: If a past case matches closely, reuse its diagnosis.
- Revise: Adjust dosage based on patient weight differences.
- Retain: Add new cases to the case base.

Case Base Maintenance

- Forgetting: Remove outdated cases (e.g., old treatment protocols).
- Clustering: Group similar cases to speed up retrieval.
- Validation: Regularly audit cases for accuracy.

4. Design a hybrid system combining k-NN and radial basis functions for time series prediction. Include architecture diagram and implementation details.

Solution

Architecture

1. **k-NN Layer**: Identify similar historical subsequences.
2. **RBF Layer**: Fit a radial basis function network to interpolate between neighbors.
   - **RBF Centers**: Selected neighbors from k-NN.
   - **Weights**: Learned via least squares or gradient descent.

Implementation

- **Distance Metric**: Dynamic Time Warping (DTW) for k-NN to handle temporal shifts.
- **RBF Kernel**: Gaussian $\phi(r) = e^{-(\epsilon r)^2}$.
- **Optimization**: Use gradient descent to minimize prediction error.

5 . i) Differentiate between generative and discriminative learning models. In a multinational company, there are people speaking different languages of their own mother tongue. The auto teller engine hosted by the company has a task of determining the language that someone is speaking by determining the linguistic differences without learning any language. Which learning model it has to follow ? Why ?

ii) For the application of your choice, explain the machine learning process indicating

1) Type of machine learning model

2) Dataset needed and how much ?

3) input parameters and expected outcome

4) Possible evaluation strategy.

Solution

5.i) Generative vs. Discriminative Models

- **Generative Models** (e.g., Naive Bayes, GMMs): Learn joint distribution $P(X, Y)$. Useful for generating synthetic data.
- **Discriminative Models** (e.g., Logistic Regression, SVM): Learn $P(Y|X)$ directly. Better for classification tasks.
- **Language Identification**: Use a **generative model** (e.g., GMM/HMM) because it can model linguistic features (phonemes, syntax) without explicit labels by clustering acoustic patterns.

5.ii) Example: Spam Detection

1. **Model**: Discriminative (Logistic Regression/SVM).
2. **Dataset**: 10,000 labeled emails (spam/ham).
3. **Input Features**: TF-IDF of words, sender domain.
4. **Evaluation**: Precision/recall, F1-score on a held-out test set.