Name: Janmejay Mohanty **Course:** Applied Machine Learning **Homework Assignment**: 5
CWID: 20009315 **Course Number:** AAI 695

Solutions

Ans1:

1) Reducing the dimensionality and removing irrelevant features of data is important for instance-based learning methods like kNN (k-Nearest Neighbors) for several reasons:

1. **Curse of Dimensionality:** As the number of features in a dataset increases, the volume of the space that the data resides in also increases exponentially. This indicates that for larger dimensions, the data points have a tendency to be farther apart, making it challenging for kNN to precisely locate the closest neighbors. This is referred described as the "curse of dimensionality," as it may cause overfitting and subpar model performance.
2. **Computational Efficiency:** By lowering the number of characteristics that must be taken into account, dimensionality reduction techniques like PCA (Principal Component Analysis) may considerably lower the computing cost of kNN. This can increase the algorithm's scalability and efficiency for huge datasets.
3. **Irrelevant Features:** Removing pointless features might assist the dataset become less noisy and the model's accuracy. Irrelevant features might cause overfitting and impair the model's capacity to generalize, which can cause the model to perform poorly on fresh, untested data.
4. **Better Visualization:** It is simpler to see the connections between the data points when the dimensions of the data are reduced. This makes it easier to spot patterns and connections that may not be obvious in high-dimensional settings.

Overall, reducing the dimensionality and removing irrelevant features can help to improve the accuracy, scalability, and interpretability of instance-based learning methods like kNN, making them more effective for a wide range of applications.

2) The variability issue in K-Means refers to the sensitivity of the algorithm to the initial random placement of the cluster centers. Due to the stochastic nature of the algorithm, different random initializations can lead to different final cluster assignments and different cluster centers. This can result in poor clustering performance and reduced reliability of the results. One way to address the variability issue in K-Means is to use a technique called "multiple random initialization" or "K-Means++ initialization". This technique involves running the K-Means algorithm multiple times with different initializations and selecting the best solution based on a pre-defined criterion, such as the sum of squared distances between the data points and their cluster centers. K-Means++ initialization is a modification of the original K-Means algorithm that attempts to choose initial cluster centers that are far away from each other and likely to represent different clusters. This initialization strategy can reduce the sensitivity of the algorithm to the initial placement of the cluster centers, resulting in more stable and reliable clustering results. Another way to address the variability issue in K-Means is to use a related algorithm called "K-Means with medoids" or "PAM" (Partitioning Around Medoids). Instead of choosing the mean of the data points in each cluster as the cluster center, PAM chooses one of the actual data points in the cluster (i.e., the medoid) as the center. This can make the algorithm less sensitive to outliers and more robust to variations in the initial cluster assignments. In summary, to address the variability issue in K-Means, one can use techniques such as multiple random initialization or K-Means++ initialization, or switch to a related algorithm like PAM that is less sensitive to the initial cluster assignments.

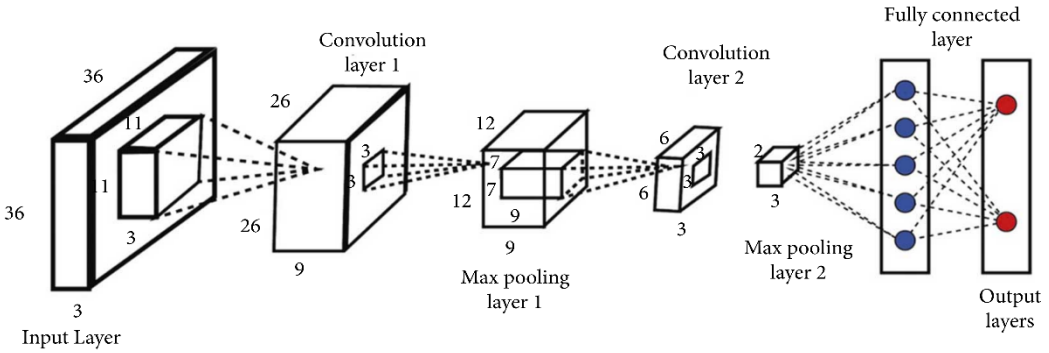
3) A probabilistic model for cluster analysis and density estimation is the Gaussian Mixture Model (GMM). It is assumed that the data are produced by a combination of several Gaussian distributions, each of which is in charge of producing a certain cluster or subpopulation of data. GMM may be used in anomaly detection to find data points that deviate noticeably from the rest of the data. The mean, covariance, and weight of each component are among the Gaussian mixture's parameters that are initially randomly initialized using the GMM method. The Expectation-Maximization (EM) algorithm, which maximizes the likelihood of the data given the current set of parameters, is then used by the algorithm to repeatedly update the parameters. Once the GMM is trained, it can be used for anomaly detection by computing the likelihood of each data point under the model. Data points with low likelihood values are considered to be anomalies or outliers since they are less likely to have been generated by the mixture of Gaussian distributions. The threshold for detecting anomalies can be set by choosing a suitable cut-off value for the likelihood score.

There are different approaches to using GMM for anomaly detection, including:

1. **Unsupervised Anomaly Detection:** This approach involves training the GMM on the entire dataset without any prior knowledge of the anomalies. The data points with low likelihood values under the trained GMM are identified as anomalies.
2. **Semi-Supervised Anomaly Detection:** This approach involves training the GMM on the normal data points and using it to detect anomalies in new, unseen data. The normal data points are used to estimate the parameters of the GMM, and the likelihood of the new data points is computed under the model. Data points with low likelihood values are considered to be anomalies.
3. **Supervised Anomaly Detection:** This approach involves training the GMM on a labeled dataset that includes both normal and anomalous data points. The GMM is trained to model the normal data distribution, and the likelihood of the new data points is computed under the model. Anomalies are identified as data points that have low likelihood values and belong to the anomalous class.

In summary, Gaussian Mixture Model is a probabilistic model that can be used for density estimation and anomaly detection. It assumes that the data is generated by a mixture of Gaussian distributions, and it identifies anomalies by computing the likelihood of each data point under the model. GMM can be used for unsupervised, semi-supervised, or supervised anomaly detection depending on the availability of labeled data.

4) Diagram of Convolutional Neural Network



Convolutional Neural Networks are used for image recognition, image classification, face recognition, etc. Basically, CNN takes in an image, processes it, and then classifies it under

certain categories. Each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers and apply Softmax function to classify an object with probabilistic values between 0 and 1.

Convolutional Layer: This layer performs convolution on the input image using a set of filters. Each filter extracts a specific feature from the image, such as edges or corners. The result of this layer is a set of feature maps that represent different features in the input image.

Activation Layers: These layers apply a non-linear activation function to the output of the convolutional or pooling layers, adding non-linearity to the network, and enabling it to learn more complex representations i.e., ReLU stands for Rectified Linear Unit for a non-linear operation. The output is $f(x) = \max(0, x)$. ReLU's purpose is to introduce non-linearity in our Convolutional network. Since, the real-world data would want our network to learn would be non-negative linear values.

Pooling Layer: This layer reduces the spatial dimensions of the feature maps obtained from the previous layer, by performing down sampling. This helps to reduce the number of parameters in the model and prevent overfitting. Spatial pooling can be of different types:

1. Max Pooling
2. Average Pooling
3. Sum Pooling

Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.

Fully Connected Layer: This layer takes the flattened output from the previous layer and performs classification by applying a set of weights to each feature. This layer is similar to the fully connected layers of a regular neural network.

List of CNN Algorithms:

1. LeNet
2. AlexNet
3. VGGNet
4. InceptionNet/GoogleNet
5. Xception
6. ResNet
7. ZfNet

5) Backpropagation, which is the process of computing the gradients of the loss function with respect to the weights of a neural network, can lead to problems such as the vanishing and exploding gradient problems. These issues can make effectively training deep neural networks difficult or even impossible. The vanishing gradient problem happens when, as they are propagated backwards through the layers of the network, the gradients of the loss function with respect to the weights become exceedingly small. This can cause the weights to be updated too quickly, leading to unstable behavior and divergence during training. Several techniques have been developed to address these problems, including:

1. **Weight initialization:** Proper initialization of the weights can help prevent the gradients from vanishing or exploding. Techniques like Xavier initialization or He initialization have been shown to be effective for this purpose.
2. **Non-saturating activation functions:** Activation functions like ReLU or its variants (e.g., LeakyReLU) do not saturate in the positive region and can help prevent the vanishing gradient problem.
3. **Gradient clipping:** This technique involves clipping the gradients to a maximum value to prevent them from becoming too large during training.
4. **Batch normalization:** This technique involves normalizing the activations of each layer to have zero mean and unit variance, which can help stabilize the gradients and prevent the vanishing gradient problem.
5. **Residual connections:** Residual connections involve adding the input of a layer to its output, allowing the gradients to flow directly through the identity function. This can help prevent the vanishing gradient problem and make it easier to train deep networks.
6. **Long Short-Term Memory (LSTM):** LSTM is a type of recurrent neural network that uses gating mechanisms to control the flow of information and prevent the vanishing gradient problem in sequences of data.

In summary, the vanishing and exploding gradient problems can make it difficult to train deep neural networks effectively. Techniques such as weight initialization, non-saturating activation functions, gradient clipping, batch normalization, residual connections, and LSTM have been developed to address these problems and improve the performance of deep neural networks.

Ans2:

$$Sample\ size\ is\ n = 100$$

$$Number\ of\ incorrectly\ classified\ examples\ is\ x = 100 - 80 = 20$$

$$Let\ p1: Error_D(h)\ rate$$

$$\hat{p} = \frac{x}{n} = \frac{20}{100} = 0.20$$

$$Confidence\ interval\ for\ population\ proportion\ is\ given\ by:$$

$$(\hat{p} - E, \hat{p} + E)$$

$$Here, E = z' * \sqrt{\frac{\hat{p} * (1 - \hat{p})}{n}}$$

$$Here, z' is\ critical\ z\ score\ for\ given\ confidence\ interval.$$

$$Given: Confidence\ level\ (C) = 0.95$$

$$\propto = 1 - C = 1 - 0.95 = 0.05$$

$$\frac{\propto}{2} = \frac{0.05}{2} = 0.025$$

$$For\ a\ 95\%\ confidence\ interval, the\ critical\ value\ z'\ is\ 1.96.$$

$$Now, finding\ the\ 95\%\ confidence\ interval\ for\ the\ true\ error\ rate\ of\ the\ learned\ hypothesis\ h:$$

$$E = 1.96 * \sqrt{\frac{0.20 * (1 - 0.20)}{100}}$$

$$= 1.96 * \sqrt{\frac{0.20 * 0.80}{100}}$$

$$= 1.96 * \sqrt{\frac{0.16}{100}}$$

$$= 1.96 * \sqrt{0.0016}$$

$$= 1.96 * 0.04$$

$$= 0.0784$$

$$\hat{p} - E = 0.20 - 0.0784 = 0.1216$$

$$\hat{p} + E = 0.20 + 0.0784 = 0.2784$$

$Hence, 95\%\ confidence\ that\ the\ true\ error\ rate\ for\ the\ learned\ hypothesis\ h\ lies\ between\ 0.1216\ and\ 0.2784$