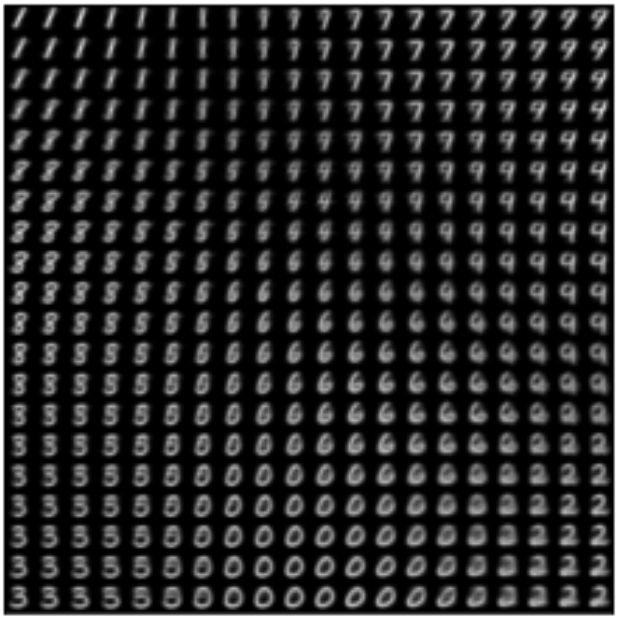
HW2 Report

Fundamentals of Computational Intelligence

Niki Nezakati
98522094
Spring 2022

Kohonen (Q1

Training with 1000 epochs, learning rate = 0.1 :
Time spent : 5m 15s
Final Image :

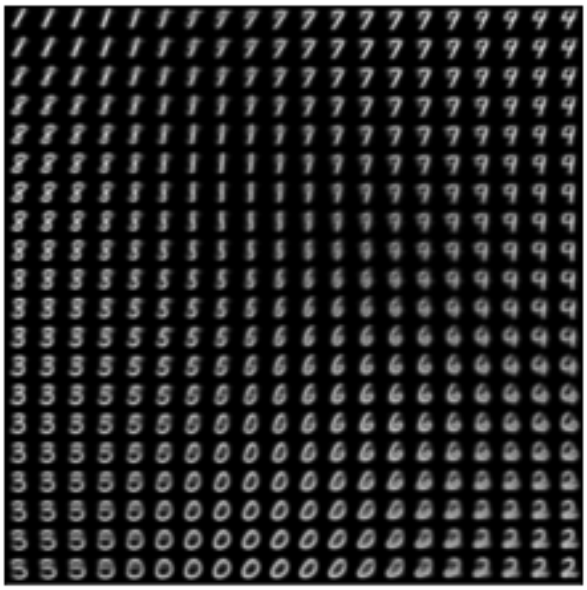


.1.1

If the learning rate is constant, there is always the possibility of **not converging**. This can be fixed by choosing a learning rate that is the best one, but this needs a lot of experiments and trials.
The better solution would be updating the learning rate after each epoch. As it can be seen below, the final image is more clear and numbers are visually more distinguishable and as the converging happened faster, the time spent for training was shorter.

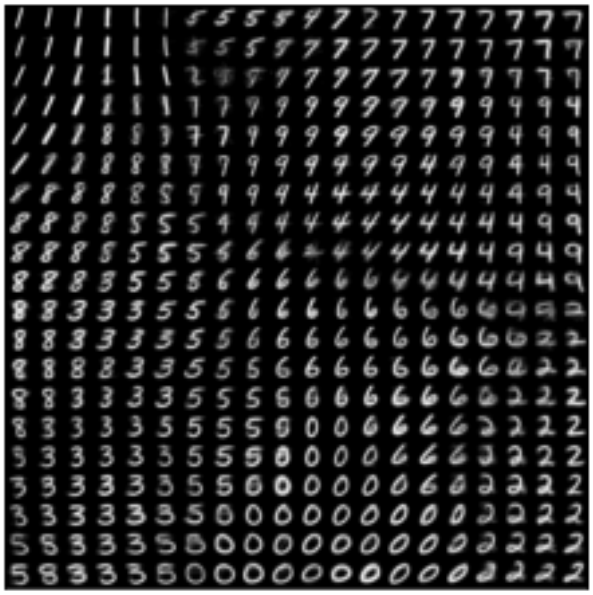
Time spent : 5m 4s

Final Image :



.2.1

The main variable that influences the output is the radius. If we keep the radius constant, whether the learning rate is constant or not, we are still only updating some fixed neurons and it causes **distortion** in our Kohonen map. But if the radius changes and decays over time, no distortion will happen. First the general organizing happens, and then it gets more specific after each epoch.



Q2) سوالات تشریحی Kohonen

1. All the entire learning process occurs without supervision because the nodes are self-organizing. They are also known as feature maps, as they are basically retraining the features of the input data, and simply grouping themselves as indicated by the similarity between each other. That’s why we don’t need labels for our dataset.
2. Kohonen networks are a type of neural network that perform clustering, also known as a knet or a self-organizing map. This type of network can be used to cluster the dataset into distinct groups when you don't know what those groups are at the beginning. This clusters or groups can be a form of classification.
3. A self-organizing map (SOM) is a grid of neurons which adapt to the topological shape of a dataset, allowing us to visualize large datasets and identify potential clusters. An SOM learns the shape of a dataset by repeatedly moving its neurons closer to the data points and finally determining a label for it.

Q3) خوشه بندي (Clustering)

Clustering is an unsupervised machine learning method of identifying and grouping similar data points in larger datasets without concern for the specific outcome. Clustering (sometimes called cluster analysis) is usually used to classify data into structures that are more easily understood and manipulated.

Using a clustering algorithm means you're going to give the algorithm a lot of input data with no labels and let it find any groupings in the data it can. Those groupings are called *clusters*. A cluster is a group of data points that are similar to each other based on their relation to surrounding data points. Clustering is used for things like feature engineering or pattern discovery.

1. K-means clustering algorithm

K-means clustering is the most commonly used clustering algorithm. It's a centroid-based algorithm and the simplest unsupervised learning algorithm.

This algorithm tries to minimize the variance of data points within a cluster. K-means is best used on smaller data sets because it iterates over *all* of the data points. That means it'll take more time to classify data points if there are a large amount of them in the data set.
Algorithm:

Algorithm 1 k-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

2. Gaussian Mixture Model algorithm
One of the problems with k-means is that the data needs to follow a circular format. The way k-means calculates the distance between data points has to do with a circular path, so non-circular data isn't clustered correctly.
This is an issue that Gaussian mixture models fix. You don’t need circular shaped data for it to work well.
The Gaussian mixture model uses multiple Gaussian distributions to fit arbitrarily shaped data.
There are several single Gaussian models that act as hidden layers in this hybrid model. So the model calculates the probability that a data point belongs to a specific Gaussian distribution and that's the cluster it will fall under.

Algorithm:

Input: $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; ...; \mathbf{x}_N] \in \mathbb{R}^{N \times D}$ where $\mathbf{x}_n \in \mathbb{R}^D$

Output: $p(z_k = 1|\mathbf{x}_n)$ and $\boldsymbol{\mu}_k \in \mathbb{R}^D$, $\boldsymbol{\Sigma}_k \in \mathbb{R}^{K \times K}$, $\pi_k \in \mathbb{R}$ which minimise the objective log likelihood

Initialise $\{\boldsymbol{\mu}_k\}$ via the K-means algorithm, $\{\boldsymbol{\Sigma}_k\}$, and $\{\pi_k\}$ uniformly at random and evaluate log likelihood

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})_{new} = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{m}_k, \boldsymbol{\Sigma}_k) \right\}$$

Repeat:

1 $$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})_{old} := \ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})_{new}$$

2 $$p(z_k = 1|\mathbf{x}_n) := \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{m}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{m}_j, \boldsymbol{\Sigma}_j)} \quad \text{(E-Step)}$$

$$\boldsymbol{\mu}_k := \frac{1}{N_k} \sum_{n=1}^{N} p(z_k = 1|\mathbf{x}_n) \mathbf{x}_n$$

3 $$\boldsymbol{\Sigma}_k := \frac{1}{N_k} \sum_{n=1}^{N} p(z_k = 1|\mathbf{x}_n)(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \quad \text{(M-Step)}$$

$$\pi_k := \frac{N_k}{N} \quad \text{where} \quad N_k = \sum_{n=1}^{N} p(z_k = 1|\mathbf{x}_n)$$

4 $$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi})_{new} := \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{m}_k, \boldsymbol{\Sigma}_k) \right\}$$

Until Converged;

3. DBSCAN clustering algorithm

DBSCAN stands for density-based spatial clustering of applications with noise. It's a density-based clustering algorithm, unlike k-means.

This is a good algorithm for finding outliners in a data set. It finds arbitrarily shaped clusters based on the density of data points in different regions. It separates regions by areas of low-density so that it can detect outliers between the high-density clusters.

This algorithm is better than k-means when it comes to working with oddly shaped data.

DBSCAN uses two parameters to determine how clusters are defined: *minPts* (the minimum number of data points that need to be clustered together for an area to be considered high-density) and *eps* (the

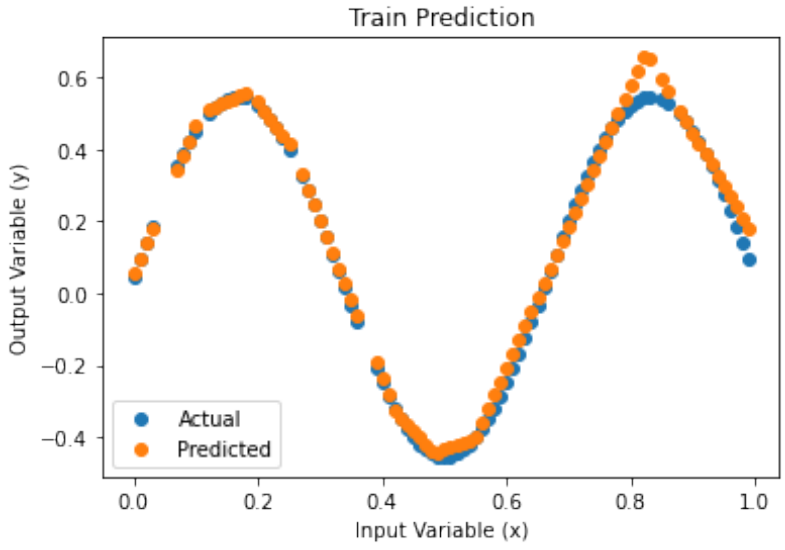
distance used to determine if a data point is in the same area as other data points).
Choosing the right initial parameters is critical for this algorithm to work.

4. BIRCH algorithm
The Balance Iterative Reducing and Clustering using Hierarchies (BIRCH) algorithm works better on large data sets than the k-means algorithm.
It breaks the data into little summaries that are clustered instead of the original data points. The summaries hold as much distribution information about the data points as possible.
This algorithm is commonly used with other clustering algorithm because the other clustering techniques can be used on the summaries generated by BIRCH.
The main downside of the BIRCH algorithm is that it only works on numeric data values. You can't use this for categorical values unless you do some data transformations.

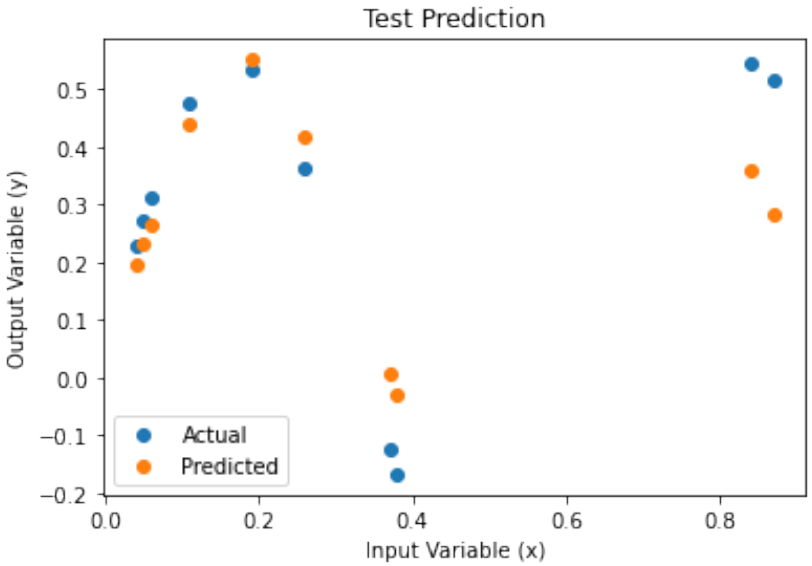
(Q4

MLP .1.4

Model’s prediction based on the training part of the dataset :



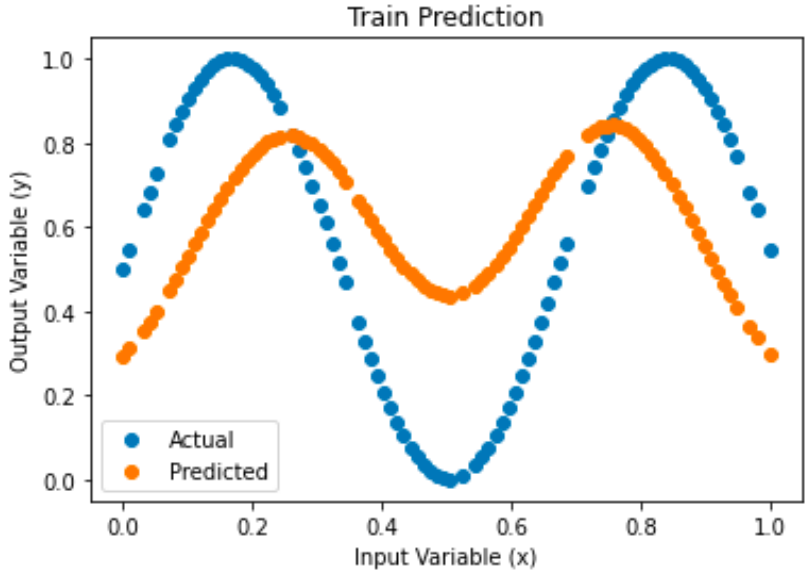
Model’s prediction based on the testing part of the dataset :



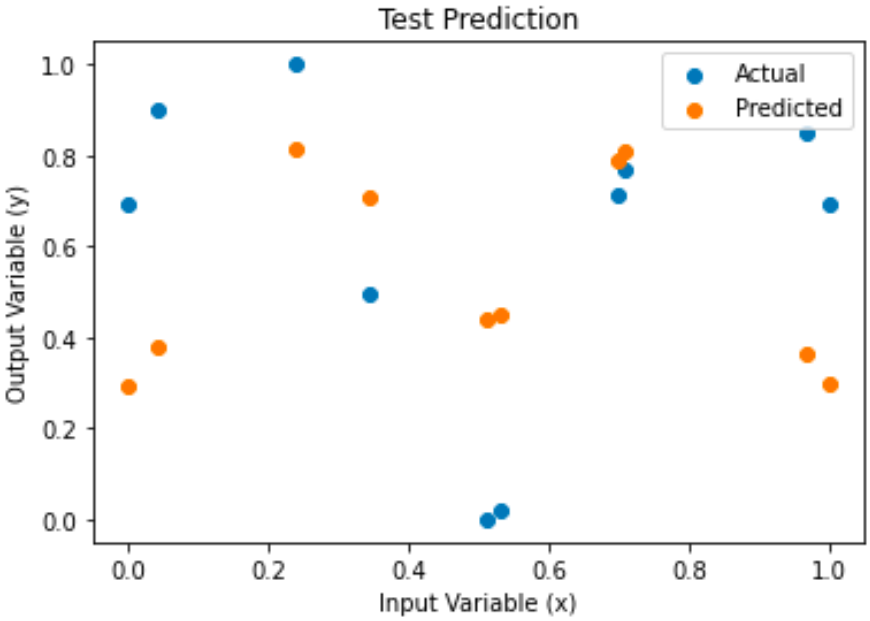
RBF Network .2.4

K-Means .1.2.4

Model’s prediction based on the training part of the dataset :

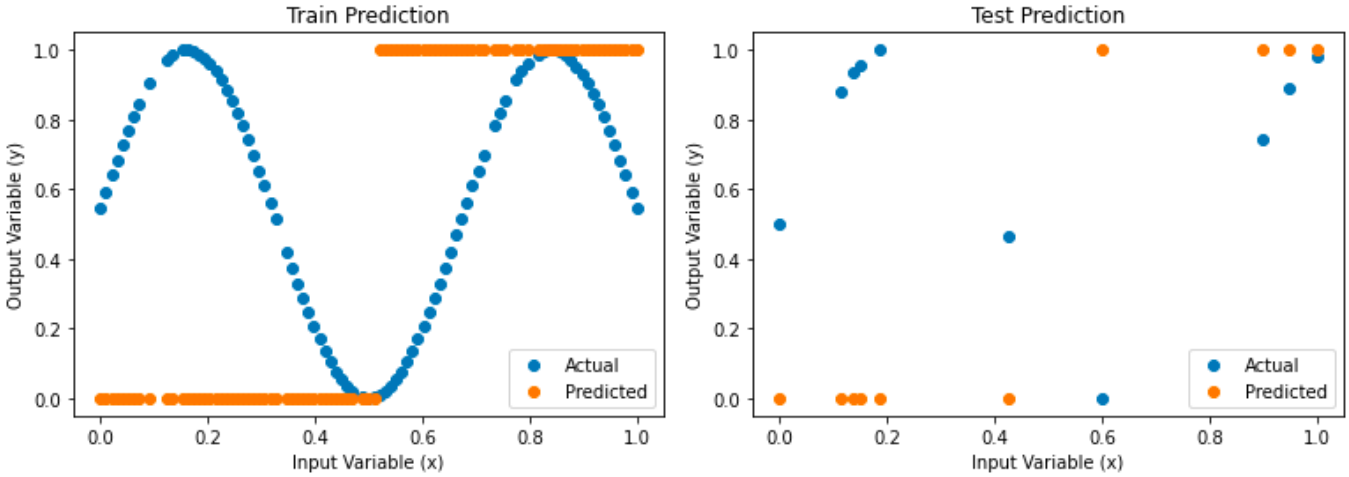


Model’s prediction based on the testing part of the dataset :



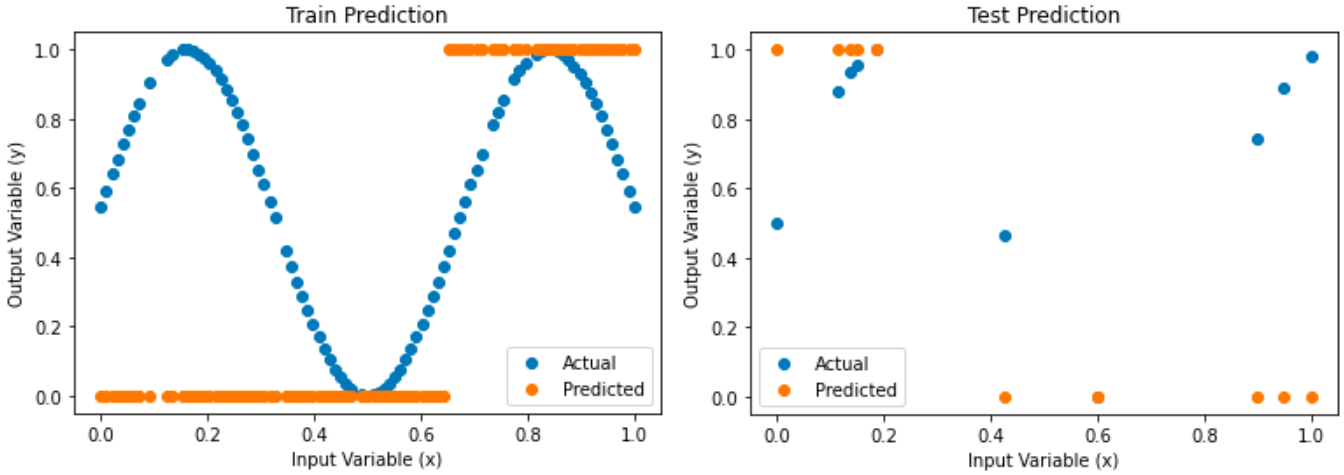
GMM .2.2.4

Model’s prediction based on the training part of the dataset vs the testing part of the dataset :

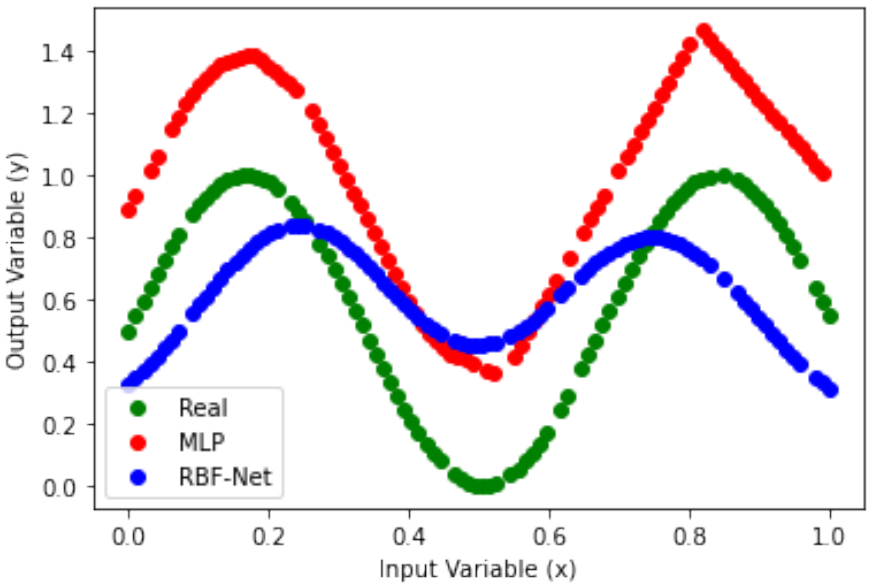


Random (BIRCH) .3.2.4

Model's prediction based on the training part of the dataset vs the testing part of the dataset :



Analysis .3.4



MLP network presents general approach as a whole to handle nonlinear relationship between the input parameter(s) and output parameter(s) while RBF network evaluates the different subspaces of input set as different relationships and produces local solutions.

As we can see in the graph above, the MLP model was able to predict the general pattern of the parameters but the RBF model predicted a more local pattern. Training process was faster using RBF model.

The advantages of MLP are Capability to learn non-linear models
and Capability to learn models in real-time (on-line learning).

The disadvantages of MLP include:
MLP with hidden layers have a non-convex loss function where there exists more than one local minimum.
MLP requires tuning a number of hyper-parameters such as the number of hidden neurons, layers, and iterations.
MLP is sensitive to feature scaling.
When to use MLP:
MLPs are suitable for classification prediction problems where inputs are assigned a class or label. They are also suitable for regression prediction problems where a real-valued quantity is predicted given a set of inputs.

The advantages of RBF:
Radial basis function (RBF) networks have advantages of easy design, good generalization, strong tolerance to input noise, and online learning ability.

The disadvantages of RBF:
Although the training is faster in RBF network but classification is slow in comparison to Multi layer Perceptron due to fact that every node in hidden

layer have to compute the RBF function for the input sample vector during classification.

When to use RBF:
Radial basis function (RBF) networks are commonly used for function approximation problems.

Can we create a network using Perceptron layers and RBF?

In Single Perceptron / Multi-layer Perceptron(MLP), we only have linear separability because they are composed of input and output layers(some hidden layers in MLP).
A non-linearity separable problem(pattern classification problem) is highly separable in high dimensional space than it is in low dimensional space. Our RBF what it does is, it transforms the input signal into another form, which can be then feed into the network to get linear separability. When we increase the dimension of the feature vector, the linear separability of feature vector increases.
So, by using Perceptron layers, and adding hidden layers of RBF, since RBF can change the inputs' dimension or form, our network can improve in classification problems.

Resources:
pythonmachinelearning.pro
freecodecamp.org
machinelearningmastery.com
towardsdatascience.com