Fundamental Mathematical Understanding of Machine Learning Techniques and Its Basic Applications

Ginni Garg[1*] *and Arti Garg*[2]

1. *National Institute of Technology Kurukshetra India 136119*
2. *Giani Zail Singh Campus College of Engineering & Technology India 151001*

**Corresponding Author: gargginni01@gmail.com*

Abstract: *We interact with machine learning in our everyday life. If we analyze deeply we can understand that from birth itself we are using Machine Learning. Our eyes do Image Processing, which helps us in extracting features from Image and our brain does classification based on extracted features. At a very small age our parents start telling us this is a cat, this is a cow, this is a dog, and so on, so this is like labels given to our Neural Network and we are simply training our Human Neural Network. In the real-life scenario, we are using Machine Learning everywhere such as Recommendation Systems, Earth Observations using Satellite Imagery, Bio-medical Domain, Agriculture Sector, Speech Recognition, and so on. Machine Learning comes under the category of Soft Computing techniques. Anyone with basic knowledge of Mathematics and programming can understand Machine Learning implementations. In this chapter, we have discussed different techniques of Machine Learning such as Supervised Learning (Feature Scaling, Data Augmentation, Fundamental Mathematics, Linear Regression, Logistic Regression, Gradient Descent, NOT, OR, AND Implementation), Unsupervised Learning (Principal Component Analysis, K-means based clustering), strategy for solving basic problems using Machine Learning techniques, the study of various evaluation metrics to understand the robustness of any model – F1 score, Recall, Precision, Youden-Index, and Sensitivity. We have discussed solutions of some of the applications based on Machine Learning such as Plant disease detection and classification, Land Use Land Cover Classification Problems over Satellite Imagery and many other remote sensing applications.*

Keywords: Machine Learning, Supervised Learning, Unsupervised Learning, ML Applications for remote sensing.

1. *What is Machine Learning?*

It is the ability of the computer to fit the data, so that we have good mathematical curves, which can be further used for prediction of complex problems outputs based on input feature given to that trained model. Machine learning can be classified into Supervised Learning and Unsupervised Learning. Supervised Learning have label of training data whereas Unsupervised learning donot have labels of data used to train Model.

1.1. Supervised Learning:

We can understand working of supervised learning as shown in Fig.1.

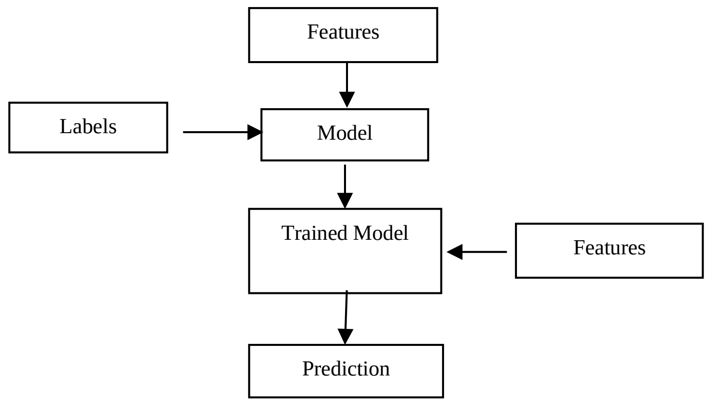

Fig.1 Represents basic model of Supervised Learning.

1.1.1. Dataset Preparation:

Usually, dataset involves features and labels. We do feature scaling to reduce the dominance of large valued feature over small valued features. There are various techniques to do feature scaling such as Normalization, and Standardization.

Normalization involves scaling all the features between zero to one as in (1).

$X' = (X - X_{min})/(X_{max} - X_{min})$, where X = original feature to normalize, X_{min} and X_{max} = min and max value of feature respectively. (1)

Standardization involves scaling all the features around mean with unit standard deviation as in (2).

$X' = (X - \mu)/\sigma$, Here X is original feature, μ = mean of all the features, σ = Standard deviation of all the features. (2)

1.1.2. Model:

We have analysis here different Machine Learning techniques Linear Regression, Logistic Regression, Gradient Descent algorithms, NOT, AND, and OR Implementations. We can understand them mathematically here to have better clarity how Machine Learning works internally.

Representation of Linear Regression Model [1]

Hypothesis: $H_\theta(x) = \theta^T x$, where $\theta = [\ \theta_0\ \ \theta_1\ \ \theta_2 . . .\ ]^T$, $x = [\ X_0\ \ X_1\ \ X_2 \ldots]^T$

Parameters: θ

Cost Function: $J(\theta) = \sum_{i=1}^{m} (H(x^i) - y^i)^2 / (2*m)$

Gradient Descent:

Repeat:

$\theta_i = \theta_j - \alpha *$ partial_derivation $(J(\theta))$

(simultaneously update of every $j = 0 \ldots. n$)

Here θ = weights, α = Learning Rate

Representation of Logistic Regression Model [1]

Hypothesis: $H_\theta(x) = g(\theta^T x)$, where $\theta = [\ \theta_0\ \ \theta_1\ \ \theta_2 . . .\ ]^T$, $x = [\ X_0\ \ X_1\ \ X_2 \ldots]^T$, $g = 1/(1 + e^{-z})$

Parameters: θ

Cost Function: $J(\theta) = -[\ \sum_{i=1}^{m} \log H(x^i) * y^i + (1 - y^i) * \log(1 - H_\theta(x^i))] / m$

Gradient Descent:

Repeat:

$\theta_i = \theta_j - \alpha *$ partial_derivation $(J(\theta))$

(simultaneously update of every $j = 0 \ldots. n$)

Here θ = weights, α = Learning Rate

While we are training our model we go through these basic mathematical equations for Linear and Logistic Regression. Initialization of weights is also very important. If we initialize weights with zero then it can train neurons of hidden layers with same features instead of training models with all the features, so this problem can be solved using randomly initializing weights in model. To check correctness of gradient descent algorithm we choose proper learning rate and plot graph between cost and number of iterations, it should be in this form as shown in Fig. 2.

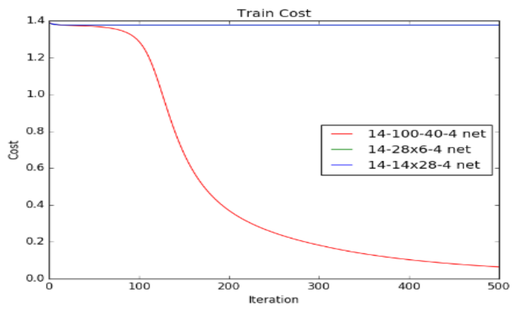

Fig. 2 Represents the graph between Cost and iterations for model.

If $\boldsymbol{\alpha}$ is too small leads to slow convergence. If $\boldsymbol{\alpha}$ is too large then $J(\theta)$ may not decrease on every iteration, may not converge. To choose α try values such as 0.001, 0.01, 0.1 and so on.

AND function representation using Logistic Regression [1] as in Table. 1.

x1, x2 lies in {0 , 1} and y = x1 AND x2

$H_\theta(x) = g(-30 + 20 * x1 + 20 * x2)$, where $g = 1/(1+e^{-z})$

Table. 1. Represents computation of AND logic using Logistic Regression.

x1	x2	$H_\theta(x)$
0	0	g(-30) = 0
0	1	g(-10) = 0
1	0	g(-10) = 0
1	1	g(10) = 1

OR function representation using Logistic Regression [1] as in Table. 2.

x1, x2 lies in {0 , 1} and y = x1 OR x2

$H_\theta(x) = g(-10 + 20 * x1 + 20 * x2)$, where $g = 1/(1+e^{-z})$

Table. 2. Represents computation of OR logic using Logistic Regression

x1	x2	$H_\theta(x)$
0	0	g(-10) = 0
0	1	g(10) = 1
1	0	g(10) = 1
1	1	g(30) = 1

Negation function representation using Logistic Regression [1] as in Table. 3.

x1 lies in {0 , 1} and y = Not x1

$H_\theta(x) = g(10 - 20 * x1)$, where $g = 1/(1+e^{-z})$

Table. 3. Represents computation of Negation logic using Logistic Regression

x1	$H_\theta(x)$
0	g(10) = 1
1	g(-10) = 0

Overfitting simply means, if we have too many features then it can train our model well on training data but its accuracies are very poor on testing data. It leads to high variance and low bias. There are some ways to address the overfitting problem.

1. Reduce number of features.
   Manually select which features to keep.
   Model Selection algorithm.
2. Regularization
   Keep all the features, but reduce values of weights θ_j

Equations of Linear and Logistic Regression after adding Regularization parameters is shown in (3) and (4).

Cost Function: $J(\theta) = [\sum_{i=1}^{m}(H(x^i) - y^i)^2 + \lambda * \sum_{j=1}^{n}\theta^2_j] / (2*m)$ (3)

Cost Function: $J(\theta) = -[\sum_{i=1}^{m}\log H(x^i) * y^i + (1 - y^i) * \log(1 - H_\theta(x^i))] / m + \lambda * \sum_{j=1}^{n}\theta^2_j / (2*m)$ (4)

Underfitting simply means, our training and testing accuracies are very poor. Model is not trained well nor its prediction is good on new data. It usually happens when we have less number of features. It leads to high bias and low variance. Simple solution to this problem is to increase the number of features. Good Fit hypothesis leads to low bias and low variance as in Fig. 3.

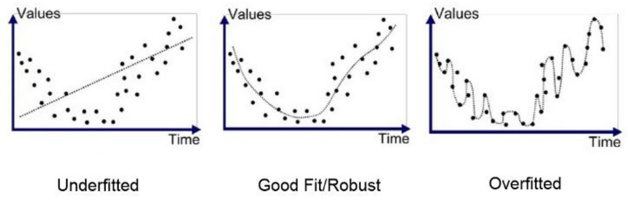

Fig. 3. Represents the diagram for Underfitting and Overfitting.

Evaluation Metrics to check robustness of any model is as follow:

1. Recall = $\frac{TP}{TP+FN}$ (5)
   Higher the Recall lower can be False Negative and vice versa.
2. Precision = $\frac{TP}{TP+ FP}$ (6)
   Higher the Precision lower can be False Positive and vice versa.
3. Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$ (7)
4. Specificity = $\frac{TN}{TN+FP}$ (8)
5. Youden-Index = Recall + Specificity – 1 (9)
6. F1 – Score = $\frac{2*Precision*Recall}{Precision+Recall}$ (10)
   F1-Score represents the relationships between Precision and Recall. Mostly Higher Precision means lower recall and vice versa.
   Here TP = True Positive, FP = False Positive, FN = False Negative, and TN = True Negative.

1.2. Unsupervised Learning:

Here, we have discussed the algorithms Principal Component Analysis, and K-means Algorithm.

Principal Component Analysis: It is required to compress data, speed up training model, reduce space and memory requirements and visualization.

Algorithm:

1. Mean Normalize Feature X' = (X – μ)/σ, Here X is original feature, μ = mean of all the features, σ = Standard deviation of all the features.
2. Covariance Matrix = $[\sum_{i=1}^{m} x^i (x^i)^T] / m$, where x is feature of order n, so that we find here covariance matrix of order n*n.

   It represents every feature correlation with every other feature.

3. Eigen Value and Eigen Vectors from Covariance Matrix such as PC1, PC2, PC3, …PCn
   Max Variance is represented by Eigen Vector corresponding to maximum Eigen value and followed for others. Max Variance represents most significant information.
4. Now suppose we have taken k eigen vectors then we have a n****k*** matrix representing features in K-dimensions.
5. Original Data = m*n dimension. Matrix multiplication of (m*n) * (n****k***) gives us (m****k***), which is final dataset casted along ***k*** PC axes.

Working of K-means can be explained as below. It is used for segmentation of Image based on Clustering technique.

Algorithm:

1. Firstly, choose K, number of clusters, it can be taken randomly or one can choose Elbow method to find the number of clusters.
2. Define K random centroids.
3. Map each data point closet to any centroids.
4. Calculate variance for each centroid, and make a new centroid based on computed variance for a cluster.
5. Repeat step 3.
6. If any re-assignment occurs then go to step 4 else, we get required clusters.

Techniques to improve performance of Multilayer Perceptron, Artificial Neural Network. There are various techniques can be used to improve Neural Network based model performance.

1. Decrease Learning Rate: Usually higher learning rate leads to increase in cost. Cost usually starts increasing as it jumps out of global/local minima. So small and appropriate value of learning rate is good to have low cost.
2. Increase height and width of hidden layers: By having proper number of neurons in hidden layers and proper number of hidden layers we can improve the performance of our model.
3. Data Augmentation: There is various data augmentation techniques such as Rotating Images by different Angles, Flipping, cropping, adding noise, which helps us increasing our dataset to train model properly.
4. Ensemble techniques such as Majority Voting Methods, Weight methods, and Hybridization of different model such as MLP-SVM, CNN-SVM, helps in improving accuracies of Neural Network based models. [3]

2. Applications of Machine Learning

In this section we have discussed some of application of Machine Learning such as Plant Disease Identification, Land Use Land Cover Classification, Wildfires, Volcanoes, Climate Change Monitoring, Tracking Deforestation, Recording Artic Sea Ice Losses, Rural to Urban Mapping.

Step by step solution of machine learning based problems is as shown in Fig. 4.

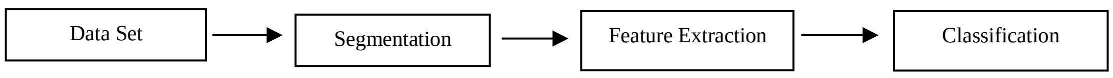

Fig. 4. Represents general steps to solve problem using Computer Vision and Machine learning

First steps involve collection of data for required problems. Segmentation involves extracting the lesion region of Image. Feature Extraction involves calculating features from lesion region of Image which are used for classification by classifiers. This is mostly standard process of processing an image and its classification. There are number of ways feature can be extracted, which involves Texture Features, Gabor Filters, KNN or ANN based Feature Extraction. Texture Feature involves contrast, correlation, energy, homogeneity, mean, standard deviation, kurtosis, skewness, variance, smoothness, IDM, RMS, entropy.

In Land Use Land Cover classification (LULC) [2] problem involves same set of steps. In LULC problems we can use Pixel wise Classification, patching concepts to improve accuracies. The satellite images before and after from natural disasters can be used to track scale of devastation caused. We can even segment area of land undergo deforestation due to wildfires. Then we can calculate total area as each pixel in satellite images represents x*x on some scale. One of another interesting use of remote sensing is in rural to urban mapping [4]. We can track the land area which is changed from rural to urban in some span of time x years.

3. Conclusion

Overall we studied different fundamental machine learning techniques which involves data set preparation, feature extraction, supervised learning, and unsupervised learning techniques. We studied different methods to improve our machine learning models such as Decrease Learning Rate, Increase height and width of hidden layers, Data Augmentation, Ensemble and Hybrid techniques. Finally we discussed some of machine learning applications in remote sensing such as LULC problems, natural disaster monitoring, rural to urban mapping.

References

1. https://www.coursera.org/learn/machine-learning
2. Garg, G., Kumar, D., Sonker, Y., & Garg, R. (2021). A Hybrid MLP-SVM Model for Classification using Spatial-Spectral Features on Hyper-Spectral Images. *arXiv preprint arXiv:2101.00214*.
3. Opitz, D., & Maclin, R. (1999). Popular ensemble methods: An empirical study. Journal of artificial intelligence research, *11*, 169-198.Carneiro, E., Lopes, W., & Espindola, G. (2021).
4. Urban Land Mapping Based on Remote Sensing Time Series in the Google Earth Engine Platform: A Case Study of the Teresina-Timon Conurbation Area in Brazil. *Remote Sensing*, *13*(7), 1338.