**Worksheet 7 MACHINE LEARNING**

1. Which of the following in sk-learn library is used for hyper parameter tuning?

Answer:- D) All of the above

2.In which of the below ensemble techniques trees are trained in parallel?

Answer:- D) All of the above

3.In machine learning, if in the below line of code: sklearn.svm.SVC (C=1.0, kernel='rbf', degree=3) we increasing the C hyper parameter, what will happen? A) The regularization will increase B) The regularization will decrease C) No effect on regularization D) kernel will be changed to linear

Answer:- A) The regularization will increase

4. Check the below line of code and answer the following questions: sklearn.tree.DecisionTreeClassifier(\*criterion='gini',splitter='best',max\_depth=None, min\_samples\_split=2) Which of the following is true regarding max\_depth hyper parameter?

Answer:- both A & B

5.Which of the following is true regarding Random Forests?

Answer- A) It's an ensemble of weak learners

6. What can be the disadvantage if the learning rate is very high in gradient descent?

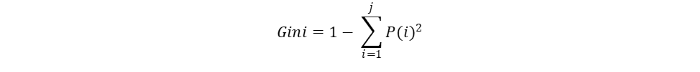
Answer:- C) Both of them

7. As the model complexity increases, what will happen?

Answer:- B) Bias will decrease, Variance increase

9. Suppose we have a dataset which have two classes A and B. The percentage of class A is 40% and percentage of class B is 60%. Calculate the Gini index and entropy of the dataset.

Answer:-The formula for Gini Index



1-(40\100\*2)^2+-(60\100\*2)^2=1.09

**Formula for Entropy**

The formula for entropy, in order to find out the uncertainty or the high disorder, goes as follows:

E(S)=c∑i=1−pilog2pi

E(A)=-40\100\*2 log2(40\100) - 50\100\*2 log2(50\100\*2) =0.7752

10. What are the advantages of Random Forests over Decision Tree?

1. Answer:- mproved accuracy: Random forests typically produce more accurate predictions than individual decision trees because they combine the predictions of multiple trees.
2. Reduced overfitting: Because a random forest is composed of multiple decision trees, each tree is less likely to overfit the training data, resulting in improved generalization to new data.
3. Handling missing values: Random forests can handle missing values in the input data without the need for imputation.
4. Handling outliers: Random forests are less sensitive to outliers in the input data than decision trees.
5. Feature importance: Random forests can provide feature importance measures, which can be used to identify the most important features in the input data.
6. Handle high dimensional data: Random forests are less prone to the curse of dimensionality, they handle high dimensional data well.
7. Robustness: Random forests are more robust to noise in the data and can generalize well from noisy data.

11. . What is the need of scaling all numerical features in a dataset? Name any two techniques used for scaling.

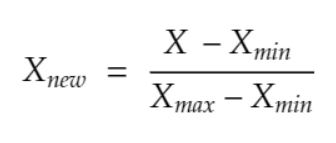
Answere:-

In Data Processing, we try to change the data in such a way that the model can process it without any problems. And Feature Scaling is one such process in which we transform the data into a better version. Feature Scaling is done to normalize the features in the dataset into a finite range.

Real Life Datasets have many features with a wide range of values like for example let’s consider the house price prediction dataset. It will have many features like no. of. bedrooms, square feet area of the house, etc.

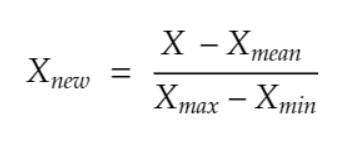
As you can guess, the no. of bedrooms will vary between 1 and 5, but the square feet area will range from 500-2000. This is a huge difference in the range of both features.

## 1.Min Max Scaling



2.Normalization

Instead of using the min() value in the previous case, in this case, we will be using the average() value.



12. Write down some advantages which scaling provides in optimization using gradient descent algorithm.

Answer:- **Gradient Descent** is the most common optimization algorithm in *machine learning* and *deep learning*. It is a first-order optimization algorithm. This means it only takes into account the first derivative when performing the updates on the parameters. On each iteration, we update the parameters in the opposite direction of the gradient of the objective function *J(w)* w.r.t the parameters where the gradient gives the direction of the steepest ascent. The size of the step we take on each iteration to reach the local minimum is determined by the learning rate α. Therefore, we follow the direction of the slope downhill until we reach a local minimum.

In this article, we’ll cover gradient descent algorithm and its variants: *Batch Gradient Descent, Mini-batch Gradient Descent, and Stochastic Gradient Descent*.

Let’s first see how gradient descent works on logistic regression before going into the details of its variants. For the sake of simplicity, let’s assume that the logistic regression model has only two parameters: weight *w* and bias *b*.

1. Initialize weight *w* and bias *b* to any random numbers.

2. Pick a value for the learning rate α. The learning rate determines how big the step would be on each iteration

The main advantages:

* We can use fixed learning rate during training without worrying about learning rate decay.
* It has straight trajectory towards the minimum and it is guaranteed to converge in theory to the global minimum if the loss function is convex and to a local minimum if the loss function is not convex.
* It has unbiased estimate of gradients. The more the examples, the lower the standard error.

13.

Answer:- In [statistical](https://en.wikipedia.org/wiki/Statistics) analysis of [binary classification](https://en.wikipedia.org/wiki/Binary_classification), the **F-score** or **F-measure** is a measure of a test's [accuracy](https://en.wikipedia.org/wiki/Accuracy_and_precision#In_binary_classification). It is calculated from the [precision](https://en.wikipedia.org/wiki/Precision_(information_retrieval)) and [recall](https://en.wikipedia.org/wiki/Recall_(information_retrieval)) of the test, where the precision is the number of true positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive. Precision is also known as [positive predictive value](https://en.wikipedia.org/wiki/Positive_predictive_value), and recall is also known as [sensitivity](https://en.wikipedia.org/wiki/Sensitivity_and_specificity) in diagnostic binary classification.

The **F1** score is the [harmonic mean](https://en.wikipedia.org/wiki/Harmonic_mean) of the precision and recall. The more generic �� score applies additional weights, valuing one of precision or recall more than the other.

The traditional F measure is calculated as follows:

* F-Measure = (2 \* Precision \* Recall) / (Precision + Recall)

For example, a perfect precision and recall score would result in a perfect F-Measure score:

* F-Measure = (2 \* Precision \* Recall) / (Precision + Recall)
* F-Measure = (2 \* 1.0 \* 1.0) / (1.0 + 1.0)
* F-Measure = (2 \* 1.0) / 2.0
* F-Measure = 1.0

15.in this article, we will discuss the difference between ‘transform’ and ‘fit\_transform’ in [sklearn](https://www.geeksforgeeks.org/learning-model-building-scikit-learn-python-machine-learning-library/) using [Python](https://www.geeksforgeeks.org/python-programming-language/).

In Data science and machine learning the methods like fit(), transform(), and fit\_transform() provided by the [scikit-learn](https://www.geeksforgeeks.org/how-to-install-scikit-learn-in-windows/) package are one of the vital tools that are extensively used in data preprocessing and model fitting. The task here is to discuss what is the difference between **fit()**, **transform, and fit\_transform()**and how they are implemented using in-built functions that come with this package.

* The **fit(data)** method is used to compute the mean and std dev for a given feature to be used further for scaling.
* The**transform(data)**method is used to perform scaling using mean and std dev calculated using the .fit() method.
* The **fit\_transform()** method does both fits and transform.

All these 3 methods are closely related to each other. Before understanding them in detail, we will have to split the dataset into training and testing datasets in any typical machine learning problem. All the data processing steps performed on the training dataset apply to the testing dataset as well but in a slightly different format. This difference could be understood well when we understand these three methods.

## The fit() Method

The fit function computes the formulation to transform the column based on [Standard scaling](https://www.geeksforgeeks.org/data-pre-processing-wit-sklearn-using-standard-and-minmax-scaler/) but doesn’t apply the actual transformation. The computation is stored as a fit object. The fit method doesn’t return anything.

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

scaler.fit(data['sepal width'])

## ****The transform() Method****

The transform method takes advantage of the fit object in the fit() method and applies the actual transformation onto the column. So, fit() and transform() is a **two-step process** that completes the transformation in the second step. Here, Unlike the fit() method the transform method returns the actually transformed array.

scaler.transform(data['sepal width'])

## ****The fit\_transform() Method****

As we discussed in the above section, fit() and transform() is a two-step process, which can be brought down to a one-shot process using the fit\_transform method. When the fit\_transform method is used, we can compute and apply the transformation in a **single step**.

scaler.fit\_transform(X\_train)