Machine Learning(Supervise):-

In machine learning we need to divide dataset into 3 types:-
!)Training datasets:-
In this case we know both i/p & o/p as well, we just train the ml model using this known things.If it gives same o/p as actual o/p our model is accuracy.

2)Validation dataset:-
It is a subset of training dataset, It will use only after the training datasets ececutes.It checks what model is gives more accurate result.It used to reduce the errors.

3)Test Datasets:-
Here we know only i/p datasets without lables (or o/p) we check the prediction of o/p through this model.And compair with actual o/p to check the accuracy.

Regularization: A Fix To Overfitting In Machine Learning:-
https://www.enjoyalgorithms.com/blog/regularization-in-machine-learning

Overfitting and Underfitting:-
1)Overfitting:-low Bias and Less Variable
The training dataset does well(more accuracy) but test datasets are not well(less accuracy)

what causes Overfitting:-
*The ML model learn noise/invalid data_sets during training period.
*The dataset is too large/complex.

2)Underfitting:-
Both training dataset not does well(less accuracy) and test datasets are also not well(less accuracy).

what causes Underfitting:-
*Underfitting occurs when a model is too simple.
*informed by too few features or regularized too much.
*which makes it inflexible in learning from the dataset.
Simple learners tend to have less variance in their predictions but more bias towards wrong outcomes.

Linear Regrission:-

Mean square error(MSR) or mean squared deviation (MSD) or cost function:-

Why square is added in that?this is my question?

The mean squared error (MSE)/cost function tells you how close a regression line is to a set of points(which are not lied on regresion line). It does this by taking the distances from the points to the regression line (these distances are the “errors or Deviations”) and squaring them.The squaring is critical to reduce the complexity with negative signs. To minimize MSE, the model could be more accurate, which would mean the model is closer to actual data. The squaring is necessary to remove any negative signs.This is for only one point.

So in order to find overall error we need to take average.

Threrfore formula for MSE is:-

Average of square of distance b/w actual and predicted points(i,e,,diffrence b/w actual and predicted value)

Gradient Desent:-

Read this for complete understanding

"https://www.analyticsvidhya.com/blog/2020/10/how-does-the-gradient-descent-algorithm-work-in-machine-learning/"

Gradient descent(decent means slop towards negative direction) is an iterative optimization algorithm for finding the local minimum of a function. To find the local minimum(Local minimum refers to a minimum within some neighborhood and it may not be a global minimum.) of a function using gradient descent, we must take steps proportional to the negative of the gradient (move away from the gradient) of the function at the current point.

Similarly Gradient Assent is there for positive direction

Difference b/w local minima and golbal minima:-A local minimum of a function is a point where the function value is smaller than at nearby points, but possibly greater than at a distant point. A global minimum is a point where the function value is smaller than at all other feasible points.

Gradient:-The gradient is a fancy word for derivative, or the rate of change of a function.

Difference b/w Gradient and Slop:-Gradient: (Mathematics) The degree of steepness(inclined) of a graph at any point. Slope: The gradient of a graph at any point.

Model Performance:-

Here we need to find on which co-efficience gives perfect rigression, in order to compair this perfect models with different co-efficiences , we need model performance.

R2/R_square/Person's_square/co-efficient of Determination is the method to find perfect regresion model.

Correlation Coefficient/R:-only R

Correlation coefficients are used to measure how strong a relationship is between two variables. There are several types of correlation coefficient, but the most popular is Pearson’s. Pearson’s correlation (also called Pearson’s R) is a correlation coefficient commonly used in linear regression.
"https://www.statisticshowto.com/probability-and-statistics/correlation-coefficient-formula/"

R2:-R-squared is a goodness-of-fit measure for linear regression models. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively. ... After fitting a linear regression model, you need to determine how well the model fits the data.It is the square of correlation co-efficiency in order to overcome negative value and complexity
"https://www.statisticshowto.com/probability-and-statistics/coefficient-of-determination-r-squared/"read this

R2=sum of squared regrissio error/sum of squared tatal error , read this (https://towardsdatascience.com/anova-for-regression-fdb49cf5d684)

Backward Elimination:-

We can see that our predictions our close enough to the test set but how do we find the most important factor contributing to the profit.
Here is a solution for that.
We know that the equation of a multiple linear regression line is given by
y=b1+b2*x+b3*x’+b4*x”+…….
where b1, b2, b3, … are the coefficients and x, x’, x” are all independent variables.
Since we don’t have any ‘x’ for the first coefficient we assume it can be written as a product of b and 1 and hence we append a column of ones. There are libraries that take care of it but since we are using the stats model library we need to explicitly add the column.

```
import statsmodels.regression.linear_model as sm
# add a column of ones as integer data type
x = np.append(arr = np.ones((50, 1)).astype(int),
              values = x, axis = 1)                                    ###(i,e,,
Here  the first letter in y=b1+b2*x
       +b3*x’+b4*x”+…… i,e,, b1 had no value of x.Therefore we
                                      need to assume x for b1 is 1.So we need add
column of one's
representing x value of 1)
# choose a Significance level usually 0.05, if p>0.05
#  for the highest values parameter, remove that value
x_opt = x[:, [0, 1, 2, 3, 4, 5]]
ols = sm.OLS(endog = y, exog = x_opt).fit()            ##endog:-Endogeneous, that
is model from within itself(dependent)
ols.summary()                                                  ##exog:- Exogeneous,
that is model from other(independent)
```

LOGISTIC REGRESSION:-

Visualizing the training set result in Logistic Regression:-

After making the regression model we need to visualize the training dataset

```
#Visualizing the training set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train
                                ####Here we make X_train as X_set

x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:,
0].max() + 1, step  =0.01),
                     nm.arange(start = x_set[:, 1]step.min() - 1, stop = x_set[:,
1].max() + 1, step = 0.01))
```

####Prepare the grid so that we can color ever pixel in the frame. We start by taking the minimum age of the age values (minus 1 so the points aren't touchng the axis) and we stop at the max age of the set in order to get the range of our frame. We do the same for the y axis (salary in this example) and we make to set a step argument to choose the resolution.

##"meshgrid function":-Is used to create a rectangular grid(i,e, here we use this for creating pixels)
##arange:-The arange() function is used to get evenly spaced values within a given interval.Values are generated within the half-open interval [start, stop]. For integer arguments the function is equivalent to the Python built-in range function, but returns an ndarray rather than a list.results will often not be consistent. It is better to use linspace for these cases.
##Syntax:For arange
numpy.arange([start,]stop, [step,]dtype=None) ##read this"https://www.w3resource.com/numpy/array-creation/arange.php"
##start:- Start of interval. The interval includes this value. The default start value is 0.
stop:- End of interval. The interval does not include this value, except in some cases where step is not an integer and floating point round-off affects the length of out.
step:- Spacing between values. For any output out, this is the distance between two adjacent values, out[i+1] - out[i]. The default step size is 1. If step is specified as a position argument, start must also be given.
##min() & max() is the function return the min and max value.
##step:-step argument to choose the resolution.

```
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(),
x2.ravel()]).T).reshape(x1.shape),
alpha = 0.75, cmap = ListedColormap(('purple','green' )))
```

##The numpy.reshape() function shapes an array without changing the data of the array.

Syntax:

```
numpy.reshape(array, shape, order = 'C')
```

##ravel():- in Python. The numpy module of Python provides a function called numpy. ravel, which is used to change a 2-dimensional array or a multi-dimensional array into a contiguous flattened array.
##T:-the transpose of the array(converting rows into columns)
##alpha:-alpha is a value of 0 means that the pixel is fully transparent and the color in the pixel beneath will show through. A value of 1 means that the pixel is fully opaque.
##contour and contourf draw contour lines and filled contours, respectively. Except as noted, function signatures and return values are the same for both versions

```
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
```

##It plot x-axis b/w x1.min() & x1.max() values and y-axis b/w x2.min() & x2.max(). To set the limits for X-axis only, We could use xlim() and set_xlim() methods. Similarly to set the limits for Y-axis, we could use ylim() and set_ylim() methods. We can also use axis() method which can control the range of both axes.lim() makes number of division of axis on x-axis and y-axis(suppose we assume xlim(4,8) it creates an axis of 4,5,6,7,8) read this for more"https://www.delftstack.com/howto/matplotlib/how-to-set-limit-for-axes-in-matplotlib/"

```
for i, j in enumerate(nm.unique(y_set)):  mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],  c = ListedColormap(('purple', 'green'))(i), label = j)
```

##To generate the scatter plot we loop through each value and assign a color depending on whether the value is 1 or 0
##read this for enumerate "https://www.geeksforgeeks.org/enumerate-in-python/"
##unique:-get unique values from the list.
##syntax:-enumerate(iterable, start=0)
##scatter(x,y),where x=x_set[y_set == j, 0].it means y_set in x_set with the values of 0(i,e No/Notspam,,ect) & x=x_set[y_set == j, 1].it means y_set in x_set with the values of 1(i,e Yes/Spam,,ect)
##C:-color map

```
mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary')
mtp.legend()
mtp.show()
```

KNN

ALGORITHM:-

```
from sklearn.neighbors import KNeighborsClassifier
classifier= KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2 )
```

##IF P=1--->Manhattan distance

```
classifier.fit(x_train, y_train)
        ##IF P=2--->Euclidean distance
```

The Minkowski distance:- generalizes the "Euclidean and the Manhatten distance" in one distance metric. If we set the parameter p in the following formula to 1 we get the manhattan distance an using the value 2 gives us the euclidean distance.

SVM

ALGORITHM:-

```
from sklearn.svm import SVC # "Support vector classifier"
classifier = SVC(kernel='linear', random_state=0)
classifier.fit(x_train, y_train)
```

Kernel Function is a method used to take data as input and transform into the required form of processing data. “Kernel” is used due to set of mathematical functions used in Support Vector Machine provides the window to manipulate the data.

Types of Kernel function:-

1)Gaussian Kernel: It is used to perform transformation, when there is no prior knowledge about data.

K (x, y) = e ^ - (\frac{||x - y||^2} {2 \sigma^2})

2)Gaussian Kernel Radial Basis Function (RBF) : Same as Gaussian kernel function, adding radial basis method to improve the transformation.

3)Sigmoid Kernel: this function is equivalent to a two-layer, perceptron model of neural network, which is used as activation function for artificial neurons.

4)Polynomial Kernel: It represents the similarity of vectors in training set of data in a feature space over polynomials of the original variables used in kernel.

5)Linear Kernel: used when data is linearly separable.

Decision Tree

Classification Algorithm:-

1)Attribute Selection Measures(ASM):-

Attribute is nothing but future i,e,, independent variables(ex:-age,salary,,,ect)

But in the case of decision tree we need to calculate which attribute/Future may come first/become_root and other become branches.

Through ASM we can calculate this, there are 2 types:-

1)Information gain/ Entropy:-

2)GINI Index:-

1)Information gain/ Entropy:-

*Entropy:-

Entropy can be defined as a measure of the purity of the sub split. Entropy always lies between 0 to 1.Suppose take an example

A((10yes/5NO)(it is impured bcz it has 10 yes and 8 no)----------B((5yes/5no)(it is completly impure because it contains equal number of both yes and no)
: (Therefore decision is continue,it split into A & B)
 (Therefore decision is continue,it split into others branches)
: (Therefore it becomes node))
 (Therefore it becomes node))
:
:
:
:
C(5yes/0no)(it is, pure bcz it has only yes not contains no)(Therefore decision is stop)(it become the Leaf)

Formula for Entropy=P(Yes)Plog2-P(No)Plog2 ,Where P(yes):-probability of yes & P(No):-Probability of No

 *Information Gain:-
Problem in Entropy is How to choose the root value I,e,, either A or B or C, So In order to choose we need to use Information Gain.
It compairs the Entropy of sets assuming (A as root & B/C as subset, B as root and C/A as subset or C as root and A/B as subsets)
Based on this we select the attribute.

Example:-
```
#Fitting Decision Tree classifier to the training set
From sklearn.tree import DecisionTreeClassifier
classifier= DecisionTreeClassifier(criterion='entropy', random_state=0)
classifier.fit(x_train, y_train)
```

2)GINI Index:-
Formula=1-[P*2] Where P:-Probability of Yes or NO
*Compair to Entropy it makes task easier bcz in entropy we have log functions it makes program difficuilt to exicute and it is simple compair to Emtropy.
*In GINI ,if the value decreased accuracy increased,inverse to Entropy.
*For an example Consider completly impured means(5YES/5NO)
In this case Entropy value is 1
In this case GINI index value is 0.5

2)Pruning: Getting an Optimal Decision tree:-It is very important for overcome overfitting problem.(read the Errors in ML in notebook)
Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.
There are 2 types:-
1)Cost Complexity Pruning
2)Reduced Error Pruning.

Cost Complexity pruning:-(alpha) / (ccp_alpha):-
Syntax:-
```
tree=DecisionTreeClassifier(ccp_alpha=0.02,random_state=0)
```

```
tree.fit(x_train,y_train)
y_pred=tree.predict(X_test)
```

It assigns alpha valuse for each decision trees.
Alpha value is less for weak tree and more for strong tree.
It removes weak tree first.
Depending upon alpha value it will remove trees.

Reduced Error Pruning:-
breaks the samples into a training set and a test set. The tree is induced completely on the training set. –Working backwards from the bottom of the tree, the subtree starting at each nonterminal node is examined. If the error rate on the test cases improves by pruning it, the subtree is removed. The process continues until no improvement can be made by pruning a subtree, The error rate of the final tree on the test cases is used as an estimate of the true error rate.

