UNIT-1
(2Marks)

To get basic idea see- https://www.youtube.com/playlist?list=PLeo1K3hjS3uvCeTYTeyfe0-rN5r8zn9rw

https://youtu.be/eIicc0UTd4M

https://www.youtube.com/watch?v=Y4qO9unerGs&list=PLYwpaL_SFmcBhOEPwf5cFwqo5B-cP9G4P

Category – 1 (Easy)

1.What are the different classification algorithms?
Classification can be performed on structured or unstructured data. Classification is a technique where we categorize data into a given number of classes. The main goal of a classification problem is to identify the category/class to which a new data will fall under.

Various types of classification algorithms:

- Logistic Regression.
- Naïve Bayes.
- Stochastic Gradient Descent.
- K-Nearest Neighbours.
- Decision Tree.
- Random Forest.
- Support Vector Machine.
  - **Logistic Regression**: Logistic regression is a machine learning algorithm for classification. In this algorithm, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function.
  - **Naïve Bayes:** Naive Bayes algorithm based on Bayes' theorem with the assumption of independence between every pair of features. Naive Bayes classifiers work well in many real-world situations such as document classification and spam filtering.
  - **Stochastic Gradient Descent:** Stochastic gradient descent is a simple and very efficient approach to fit linear models. It is particularly useful when the number of samples is very large. It supports different loss functions and penalties for classification.
  - **K-Nearest Neighbours**: Neighbours based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbours of each point.
  - **Decision Tree**: Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.
  - **Random Forest:** Random forest classifier is a meta-estimator that fits a number of decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement.
  - **Support Vector Machine:** Support vector machine is a representation of the training data as points in space separated into categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

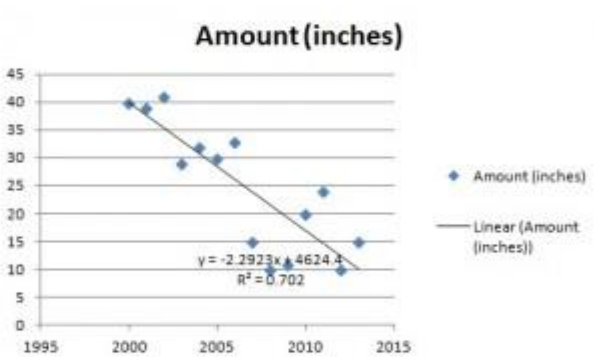
~~~~~~
~~~~~~

2. Write the cost function for linear regression in one variable?

Simple Linear Regression

You're probably familiar with plotting line graphs with one X axis and one Y axis. The X variable is sometimes called the independent variable and the Y variable is called the dependent variable. Simple linear regression plots one independent variable X against one dependent variable Y. Technically, in regression analysis, the independent variable is usually called the predictor variable and the dependent variable is called the criterion variable. However, many people just call them the independent and dependent variables. More advanced regression techniques (like multiple regression) use multiple independent variables.

Regression analysis can result in *linear* or *nonlinear* graphs. A linear regression is where the relationships between your variables can be described with a straight line. Non-linear regressions produce curved lines.(**)



Simple linear regression for the amount of rainfall per year.

Cost function= 1/2m summation (yi-y prediction)2

~~~~~~

3.What are the commonly used convergence criteria in gradient descent algorithm?

Strongly convex f. In contrast, if we assume that f is strongly convex, we can show that gradient descent converges with rate O(ck) for 0 <c< 1. This means that a bound of f(x(k)) − f(x∗) ≤ ϵ can be achieved using only O(log(1/ϵ)) iterations. This rate is typically called "linear convergence."

~~~~~~

4. Name some of the distance based algorithms in machine learning?

- **Hamming Distance:**

Hamming Distance measures the similarity between two strings of the same length. The Hamming Distance between two strings of the same length is the number of positions at which the corresponding characters are different.

Look carefully – seven characters are different whereas two characters (the last two characters) are similar:

euclidean and manhattan

Hence, the Hamming Distance here will be 7. Note that larger the Hamming Distance between two strings, more dissimilar will be those strings (and vice versa).

Euclidean Distance: *Euclidean Distance represents the shortest distance between two points.*

he formula for Euclidean Distance:

$$d = ((p_1 - q_1)^2 + (p_2 - q_2)^2)^{1/2}$$

We use this formula when we are dealing with 2 dimensions. We can generalize this for an n-dimensional space as:

$$D_e = \left(\sum_{i=1}^{n} (p_i - q_i)^2 \right)^{1/2}$$

Manhattan Distance: *Manhattan Distance is the sum of absolute differences between points across all the dimensions.* the Manhattan distance in a 2-dimensional space is given as:

$$d = |p_1 - q_1| + |p_2 - q_2|$$

And the generalized formula for an n-dimensional space is given as:

$$D_m = \sum_{i=1}^{n} |p_i - q_i|$$

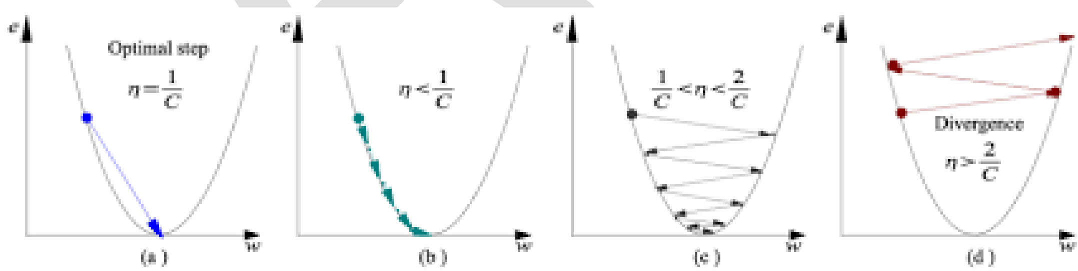
Minkowski Distance: *Minkowski Distance is the generalized form of Euclidean and Manhattan Distance.*
The formula for Minkowski Distance is given as:

$$D = \left(\sum_{i=1}^{n} |p_i - q_i|^p \right)^{1/p}$$

---

5. What is the influence of learning rate on convergence of gradient descent algorithm?

- If the learning rate is too high, we might **OVERSHOOT** the minima and keep bouncing, without reaching the minima
- If the learning rate is too small, the training might turn out to be too long



a) Learning rate is optimal, model converges to the minimum
b) Learning rate is too small, it takes more time but converges to the minimum
c) Learning rate is higher than the optimal value, it overshoots but converges ($1/C < \eta < 2/C$)
d) Learning rate is very large, it overshoots and diverges, moves away from the minima, performance decreases on learning.

Category – 2 (Moderate)

1. Differentiate between regression and classification?

Classification	Regression
• Classification is the task of predicting a discrete class label	• Regression is the task of predicting a continuous quantity
• In a classification problem data is labelled into one of two or more classes	• A regression problem requires the prediction of a quantity
• A classification problem with two classes is called binary, more than two classes is called a multi-class classification	• A regression problem with multiple input variables is called a multivariate regression problem
• Classifying an email as spam or non-spam is an example of a classification problem	• Predicting the price of a stock over a period of time is a regression problem

~~~~~~

2. Is a large margin classifier better than others?

SVM is a type of classifier which classifies positive and negative examples, here blue and red data points

As shown in the image, the largest margin is found in order to avoid overfitting ie,.. the optimal hyperplane is at the maximum distance from the positive and negative examples(Equal distant from the boundary lines).

To satisfy this constraint, and also to classify the data points accurately, the margin is maximised, that is why this is called the large margin classifier.

~~~~~~

3. Should the data be normalized before feeding to machine learning model?

Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as Min-Max scaling.

It is better to normalize data as it reduces the range difference between various data points and improves model accuracy.

Here’s the formula for normalization:

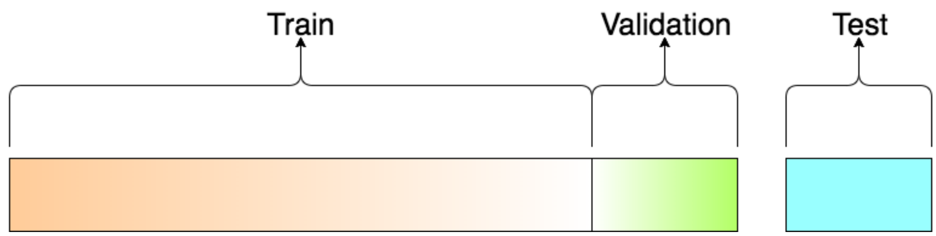
$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Here, Xmax and Xmin are the maximum and the minimum values of the feature respectively.

- When the value of X is the minimum value in the column, the numerator will be 0, and hence X’ is 0
- On the other hand, when the value of X is the maximum value in the column, the numerator is equal to the denominator and thus the value of X’ is 1
- If the value of X is between the minimum and the maximum value, then the value of X’ is between 0 and 1

~~~~~~

4. Dataset is split into train, test and validation sets? What is the purpose?
~~~~~~



This mainly depends on 2 things. First, the total number of samples in your data and second, on the actual model you are training.

Training Dataset*: The sample of data used to fit the model.* The actual dataset that we use to train the model (weights and biases in the case of a Neural Network). The model *sees* and *learns* from this data.

Validation Dataset*: The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.*

Test Dataset*: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.*

~~~~~~

(8marks)

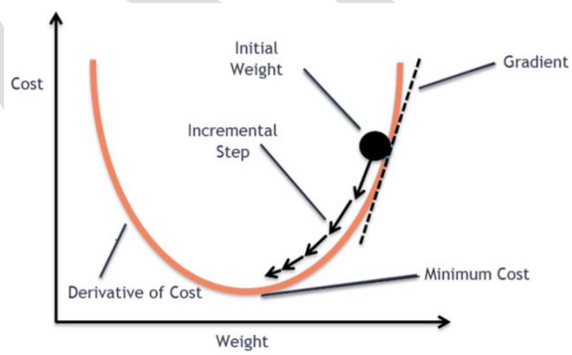
1) What are the differentiate methods to find parameters of linear regression?

Linear equation method:

Gradient descent method:

Gradient Descent Algorithm:

Gradient descent is an optimization algorithm that's used when training a machine learning model. It's based on a convex function and tweaks its parameters iteratively to minimize a given function to its local minimum.



To find the local 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.

After making a hypothesis with initial parameters, we calculate the Cost function. And with a goal to reduce the cost function, we modify the parameters by using the Gradient descent algorithm over the given data. Here's the mathematical representation for it:

$$\text{Hypothesis:} \quad h_\theta(x) = \theta_0 + \theta_1 x$$

$$\text{Parameters:} \quad \theta_0, \theta_1$$

$$\text{Cost Function:} \quad J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

$$\text{Goal:} \quad \underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$$

The goal of the gradient descent algorithm is to minimize the given function (say cost function). To achieve this goal, it performs two steps iteratively:

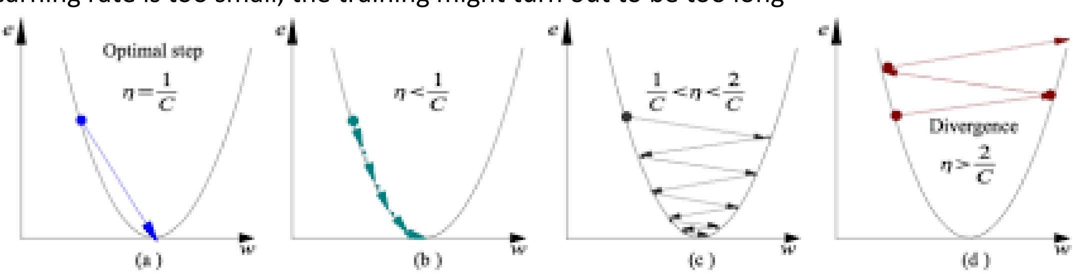
1. **Compute the gradient** (slope), the first order derivative of the function at that point
2. **Make a step (move) in the direction opposite to the gradient**, opposite direction of slope increase from the current point by alpha times the gradient at that point

Gradient descent algorithm

$$\begin{aligned} &\text{repeat until convergence } \{ \\ &\quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \\ &\quad \quad (\text{for } j = 1 \text{ and } j = 0) \\ &\} \end{aligned}$$

Alpha is called **Learning rate** – a tuning parameter in the optimization process. It decides the length of the steps. **It must be chosen carefully to end up with local minima.**

- If the learning rate is too high, we might **OVERSHOOT** the minima and keep bouncing, without reaching the minima
- If the learning rate is too small, the training might turn out to be too long



- a) Learning rate is optimal, model converges to the minimum
- b) Learning rate is too small, it takes more time but converges to the minimum
- c) Learning rate is higher than the optimal value, it overshoots but converges (1/C < η <2/C)
- d) Learning rate is very large, it overshoots and diverges, moves away from the minima, performance decreases on learning.

~~~~~~

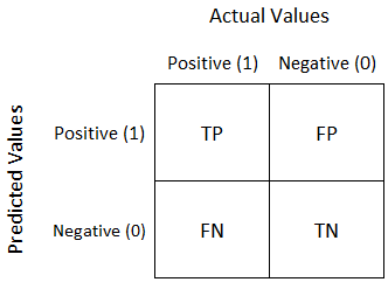
2) Brief the evaluation metrics for classification? Differentiate between precision and recall?

1. Confusion Matrix
2. Precision
3. Recall
4. Accuracy
~~~~~~

5. F1-Score
6. AUC ROC

i) Confusion Matrix:

A confusion matrix is a table with four different combinations of predicted and actual values. This is a great way to visualize the different outputs and to calculate the **Precision, Recall, Accuracy, and F-1 Score** as well as the **AUC-ROC.** We will go over each of these in more detail later in this blog. First, let's break down the TP, FP, FN, and TN in a confusion matrix.



Confusion Matrix

- **True Positives (TP)** : The number of times our model predicted YES and the actual output was also YES.
- **True Negatives (TN)**: The number of times our model predicted NO and the actual output was NO.
- **False Positives (FP)**: The number of times our model predicted YES and the actual output was NO. This is known as a Type 1 Error.
- **False Negatives (FN)**: The number of times our model predicted NO and the actual output was YES. This is known as a Type 2 Error.

ii) Precision —Precision can be described as the fraction of relevant instances among the retrieved instances. This answers the question "What proportion of positive identifications was actually correct?" The formula is as follows:

$$\text{Precision} = \frac{\text{Number of True Positives}}{\text{Number of Predicted Positives}}$$

In the terms of our confusion matrix, the equation can be represented as:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Precision expresses the proportion of the data points our model says was relevant actually were relevant.

iii) Recall- Recall, also known as *sensitivity.* This answers the question "What proportion of actual positives was classified correctly?" This can be represented by the following equation:

$$\text{Recall} = \frac{\text{Number of True Positives}}{\text{Number of Actual Total Positives}}$$

In our confusion matrix, it would be represented by:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Recall expresses which instances are relevant in a data set. It is important to examine both the Precision AND Recall when evaluating a model because they often have an inverse relationship. When precision increases, recall tends to decrease and vice versa.

iv) Accuracy: Accuracy is determining out of all the classifications, how many did we classify correctly? This can be represented mathematically as:

$$\text{Accuracy} = \frac{\text{Number of True Positives + True Negatives}}{\text{Total Observations}}$$

Using our confusion matrix terms, this equation is written as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

We want the accuracy score to be **as high as possible**. It is important to note that accuracy may not always be the best metric to use, especially in cases of a **class-imbalanced data set.** This is when the distribution of data is not equal across all classes.

For example, let's say that we are looking to build a model to help diagnose people with brain cancer. If our model just classified every single person as not having brain cancer, we would be correct the overwhelming majority of the time, but the cost of someone not being diagnosed when they do, in fact, have brain cancer is devastating. Depending on the industry, these costs may outweigh the accuracy of the model. In imbalanced cases like these, it is better to use the **F1-Score** instead.

v) F1-Score- The F1 Score is a function of precision and recall. It is used to find the correct balance between the two metrics. It determines how many instances your model classifies correctly without missing a significant number of instances. This score can be represented by the following equation:

$$\text{F1 score} = 2\,\frac{Precision\ x\ Recall}{Precision + Recall}$$

Having an imbalance between precision and recall, such as a high precision and low recall, can give you an extremely accurate model, but classifies difficult data incorrectly. We want the F1 Score to be as high as possible for the best performance of our model.

vi) AUC (Area under Curve) ROC (Receiver Operating Characteristics) Curve: The AUC ROC curve is a graph which shows the performance of a classification model at all thresholds. ROC is a probability curve and AUC represents degree of separability. ROC plots the following parameters:

True Positive Rate (TPR), also known as recall or sensitivity, which was explained in the previous section.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

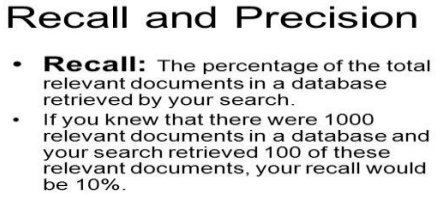
False Positive Rate (FPR), also known as Fall-out, the ratio of the false positive predictions compared to all values that are actually negative.

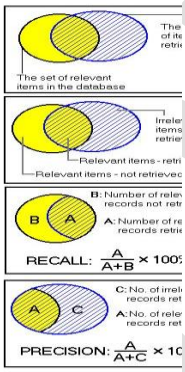
$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

Both the RPR and FPR are within the range [0, 1]. The curve is the FPR vs TPR at different points in the range [0, 1]. The best performing classification models will have a curve similar to the green line in the graph below. The green line has

the largest Area under the Curve. The higher the AUC, the better your model is performing. A classifier with only 50–50 accuracy is realistically no better than randomly guessing, which makes the model worthless (red line).

Difference b/w precision and recall:





For example, for a text search on a set of documents, precision is the number of correct results divided by the number of all returned results.

Precision takes all retrieved documents into account, but it can also be evaluated at a given cut-off rank, considering only the topmost results returned by the system. This measure is called *precision at n* or *P@n*.

Precision is used with recall, the percent of *all* relevant documents that is returned by the search. The two measures are sometimes used together in the F_1 Score (or f-measure) to provide a single measurement for a system.

Note that the meaning and usage of "precision" in the field of information retrieval differs from the definition of accuracy and precision within other branches of science and technology.

Precision

In the field of information retrieval, precision is the fraction of retrieved documents that are relevant to the query:

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved document}|}{|\{\text{retrieved documents}\}|}$$

Recall

In information retrieval, recall is the fraction of the relevant documents that are successfully retrieved.

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

For example, for a text search on a set of documents, recall is the number of correct results divided by the number of results that should have been returned.

In binary classification, recall is called sensitivity. It can be viewed as the probability that a relevant document is retrieved by the query.

It is trivial to achieve recall of 100% by returning all documents in response to any query. Therefore, recall alone is not enough but one needs to measure the number of non-relevant documents also, for example by also computing the precision.

---

3) Why are kernel functions used in machine learning? Write different popular kernels used in SVM.

In machine learning, a “kernel” is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. ... The kernel function is what is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable.

<u>Types of Kernel and methods in SVM</u>

Let us see some of the kernel function or the types that are being used in SVM:

1. Linear Kernel

Let us say that we have two vectors with name x1 and Y1, then the linear kernel is defined by the dot product of these two vectors:

K(x1, x2) = x1 . x2

2. Polynomial Kernel

A polynomial kernel is defined by the following equation:

K(x1, x2) = (x1 . x2 + 1)d,

Where,

d is the degree of the polynomial and x1 and x2 are vectors

3. Gaussian Kernel

This kernel is an example of a radial basis function kernel. Below is the equation for this:

$$k(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma\|\mathbf{x_i} - \mathbf{x_j}\|^2)$$

The given sigma plays a very important role in the performance of the Gaussian kernel and should neither be overestimated and nor be underestimated, it should be carefully tuned according to the problem.

4. Exponential Kernel

This is in close relation with the previous kernel i.e. the Gaussian kernel with the only difference is – the square of the norm is removed.

The function of the exponential function is:

$$k(x, y) = \exp\left(-\frac{\|x - y\|}{2\sigma^2}\right)$$

This is also a radial basis kernel function.

5. Laplacian Kernel

This type of kernel is less prone for changes and is totally equal to previously discussed exponential function kernel, the equation of Laplacian kernel is given as:

$$k(x, y) = \exp\left(-\frac{\|x - y\|}{\sigma}\right)$$

6. Hyperbolic or the Sigmoid Kernel

This kernel is used in neural network areas of machine learning. The activation function for the sigmoid kernel is the bipolar sigmoid function. The equation for the hyperbolic kernel function is:

$$k(x, y) = \tanh(\alpha x^T y + c)$$

This kernel is very much used and popular among support vector machines.

7. Anova radial basis kernel

This kernel is known to perform very well in multidimensional regression problems just like the Gaussian and Laplacian kernels. This also comes under the category of radial basis kernel.

The equation for Anova kernel is :

$$k(x, y) = \sum_{k=1}^{n} \exp(-\sigma(x^k - y^k)^2)^d$$

There are a lot more types of Kernel Method and we have discussed the mostly used kernels. It purely depends on the type of problem which will decide the kernel function to be used.

~~~~~~

4) Brief different normalization techniques used in Machine learning?

Four common normalization techniques may be useful:

- scaling to a range
- clipping
- log scaling
- z-score

i) <u>Scaling to a range</u>: Scaling means converting floating-point feature values from their natural range (for example, 100 to 900) into a standard range—usually 0 and 1 (or sometimes -1 to +1). Use the following simple formula to scale to a range:

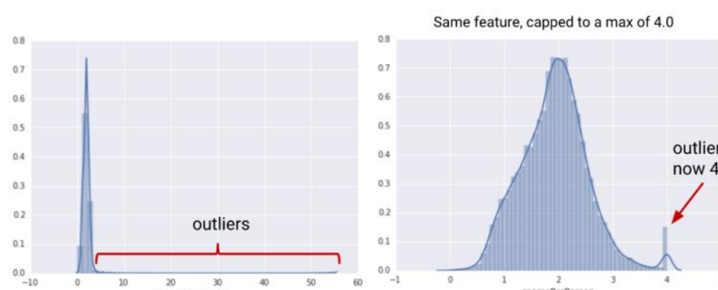
x′=(x−xmin)/(xmax−xmin)

Scaling to a range is a good choice when both of the following conditions are met:

- You know the approximate upper and lower bounds on your data with few or no outliers.
- Your data is approximately uniformly distributed across that range.

A good example is age. Most age values falls between 0 and 90, and every part of the range has a substantial number of people. In contrast, you would *not* use scaling on income, because only a few people have very high incomes. The upper bound of the linear scale for income would be very high, and most people would be squeezed into a small part of the scale.

ii) <u>Feature Clipping</u>: If your data set contains extreme outliers, you might try feature clipping, which caps all feature values above (or below) a certain value to fixed value. For example, you could clip all temperature values above 40 to be exactly 40. You may apply feature clipping before or after other normalizations.



Formula: Set min/max values to avoid outliers.
~~~~~~

Figure 2. Comparing a raw distribution its clipped version.

Another simple clipping strategy is to clip by z-score to +-Nσ (for example, limit to +-3σ). Note that *σ* is the standard deviation.

iii) <u>Log Scaling</u>: Log scaling computes the log of your values to compress a wide range to a narrow range.

$$x' = \log(x)$$

Log scaling is helpful when a handful of your values have many points, while most other values have few points. This data distribution is known as the *power law* distribution. Movie ratings are a good example. In the chart below, most movies have very few ratings (the data in the tail), while a few have lots of ratings (the data in the head). Log scaling changes the distribution, helping to improve linear model performance.

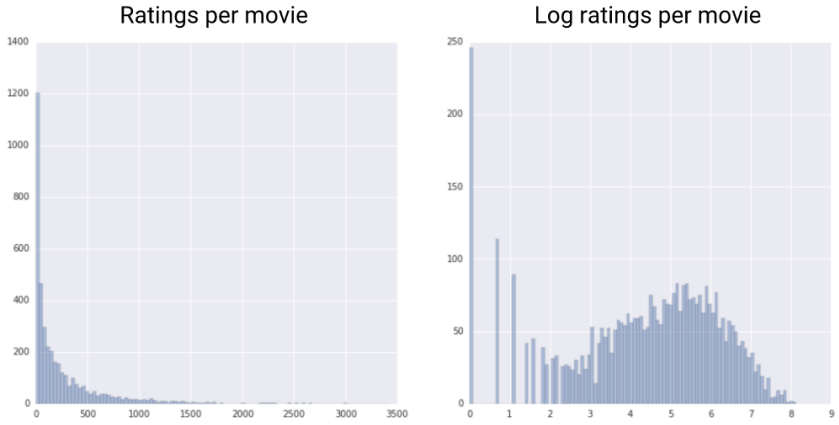
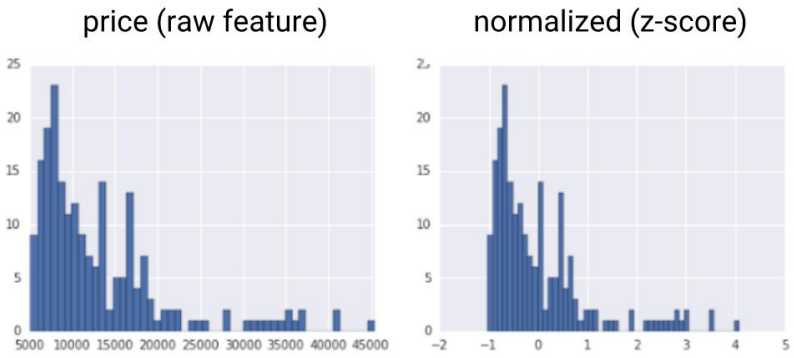


Figure 3. Comparing a raw distribution to its log.

iv) <u>Z-Score</u>: Z-score is a variation of scaling that represents the number of standard deviations away from the mean. You would use z-score to ensure your feature distributions have mean = 0 and std = 1. It's useful when there are a few outliers, but not so extreme that you need clipping. The formula for calculating the z-score of a point, *x*, is as follows:

$$x' = (x - \mu)/\sigma$$

Note: *μ* is the mean and *σ* is the standard deviation.



Normalization Technique	Formula	When to Use
Linear Scaling	x'=(x−xmin)/(xmax−xmin)	When the feature is more-or-less uniformly distributed across a fixed range.
Clipping	if x > max, then x' = max. if x < min,	When the feature contains some extreme outliers.

	then x' = min	
Log Scaling	x' = log(x)	When the feature conforms to the power law.
Z-score	x' = (x - μ) / σ	When the feature distribution does not contain extreme outliers.

~~~~~~

(Moderate)

1) How is a hyper parameter different from parameters of a model? List the hyper parameters of linear regression?

A **model parameter** is a configuration variable that is internal to the model and whose value can be estimated from data.

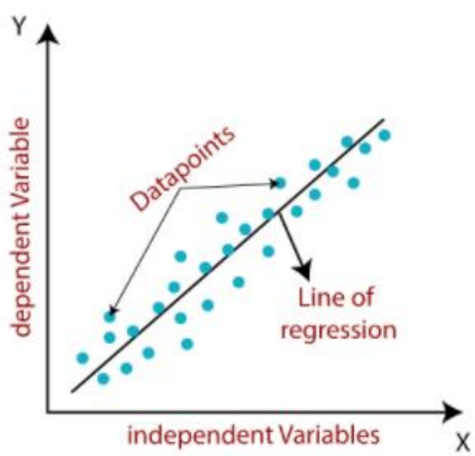
- They are required by the model when making predictions.
- They values define the skill of the model on your problem.
- They are estimated or learned from data.
- They are often not set manually by the practitioner.
- They are often saved as part of the learned model.

A **model hyperparameter** is a configuration that is external to the model and whose value cannot be estimated from data.

- They are often used in processes to help estimate model parameters.
- They are often specified by the practitioner.
- They can often be set using heuristics.
- They are often tuned for a given predictive modelling problem.

Linear Regression:

- Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression.
- It finds how the value of the dependent variable is changing according to the value of the independent variable.
- The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

$$y = a_0 + a_1 x + \varepsilon$$

Y= Dependent Variable (Target Variable)
X= Independent Variable (predictor Variable)
a0= intercept of the line (Gives an additional degree of freedom)
a1 = Linear regression coefficient (scale factor to each input value).
ε = random error

Types of Linear Regression:

- **Simple Linear Regression:**
  If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
~~~~~~

- **Multiple Linear regression:**
  If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

Hyperparameters for Linear regression:

Numerical:

- Learning Rate
- Number of Iterations

Function:

- Mean Squared Error

~~~~~~

2) Is Naive Bayes classifier a probabilistic model? Write the pros and cons of the classifier.

In statistics, **naive Bayes classifiers** are a family of simple "**probabilistic classifiers**" based on applying **Bayes**' theorem with strong (**naïve**) independence assumptions between the features. They are among the simplest Bayesian network models, but coupled with kernel density estimation, they can achieve higher accuracy levels.
Naïve Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, naive Bayes models are known under a variety of names, including **simple Bayes** and **independence Bayes**. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naïve Bayes is not (necessarily) a Bayesian method.

Advantages:-

- When the **independent assumption holds** then this classifier gives outstanding accuracy.
- **Easy to implement** as only the probability is to be calculated
- It works well with high dimensions such as text classification.

Disadvantages:-

- If the **independent assumption does not hold** then performance is very low.
- **Smoothing turns out to be a over-head** and a must to do step when probability of a feature turns out to be zero in a class.
- **Vanishing value** is also a problem due to product of many small probability(eg. 0.05^3).

~~~~~~

3) Compare Gradient descent algorithm with normal equations method for linear regression?

Gradient Descent:

Gradient descent is an optimization algorithm used to find the values of parameters of a function that minimizes a cost function. It is an iterative algorithm. We use gradient descent to update the parameters of the model. Parameters refer to coefficients in Linear Regression and weights in neural networks.

Repeat until convergence {

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

} (α: Learning rate)

Gradient descent can also converge even if the learning rate is kept fixed.

Normal Equation:

Normal Equation, an analytical approach used for optimization. It is an alternative for Gradient descent. Normal equation performs minimization without iteration.

Normal equations are equations obtained by setting equal to zero the partial derivatives of the sum of squared errors or cost function; normal equations allow one to estimate the parameters of multiple linear regression.

$$\Theta = (X^T X)^{-1} X^T y$$

(Θ: Hypothesis Parameters)

Where
X = input features value
y = output value

If the term X is non-invertible or singular then we can use regularization.

Difference between Gradient Descent and Normal Equation.

S.NO.	Gradient Descent	Normal Equation
1.	In gradient descenet , we need to choose learning rate.	In normal equation , no need to choose learning rate.
2.	It is an iterative algorithm.	It is analytical approach.
3.	Gradient descent works well with large number of features.	Normal equation works well with small number of features.
4.	Feature scaling can be used.	No need for feature scaling.

~~~~~~

(Standard)

1)Deduce the Gradient Descent algorithm for a dataset with two features?

https://towardsdatascience.com/gradient-descent-algorithm-and-its-variants-10f652806a3

Gradient Descent with Linear Regression

Gradient descent is a name for a generic class of computer algorithms which minimize a function. These algorithms achieve this end by starting with initial parameter values and iteratively moving towards a set of parameter values that minimize some cost function or metric—that are the descent part. The movement toward best-fit is achieved by taking the derivative of the variable or variables involved, towards the direction with the lowest (calculus-defined) gradient—that's the gradient part.

Ordinary linear regression is a good and simple way of demonstrating how gradient descent works. We start with some error function. We could use any metric we want, but in OLS the obvious one is the residual sum of squares.
~~~~~~

Given a sequence of points, yiyi, and a sequence of points predicted by our model, y^iy^i, RSS is:

error(m,b)=∑i=1n(yi−yi^)error(m,b)=∑i=1n(yi−yi^)

Our objective is to minimize this value. Inserting our linear regression model in for the y^iy^i predictions, and assuming (for the sake of simplicity) that we're doing regression on only one variable, we get:

RSS=∑i=1n(yi−(mxi+b))2RSS=∑i=1n(yi−(mxi+b))2

Where bb is the intercept and mm is the slope of the line of best fit.

Now we need to take the gradient. Since this is an equation of two variables (bb and mm) the gradient will consist of two partial derivatives. Hence the gradient is:

⟨∂∂b(RSS),∂∂m(RSS)⟩=⟨−2∑i=1n(yi−(mxi+b)),−2∑i=1nxi(yi−(mxi+b))⟩⟨∂∂b(RSS),∂∂m(RSS)⟩=⟨−2∑i=1n(yi−(mxi+b)),−2∑i=1nxi(yi−(mxi+b))⟩

To solve, take a step in the negative gradient direction every iteration. Eventually we will have something that converges.

Gradient Descent Algorithm For Linear Regression:

Cost Function

$$J(\Theta_0, \Theta_1) = \frac{1}{2m}\sum_{i=1}^{m}[h_\Theta(x_i) - y_i]^2$$

($h_\Theta(x_i)$: Predicted Value; y_i: True Value)

Gradient Descent

$$\Theta_j = \Theta_j - \alpha\frac{\partial}{\partial\Theta_j}J(\Theta_0, \Theta_1)$$

(α: Learning Rate)

Now,

$$\frac{\partial}{\partial\Theta}J_\Theta = \frac{\partial}{\partial\Theta}\frac{1}{2m}\sum_{i=1}^{m}[h_\Theta(x_i) - y]^2$$

$$= \frac{1}{m}\sum_{i=1}^{m}(h_\Theta(x_i) - y)\frac{\partial}{\partial\Theta_j}(\Theta x_i - y)$$

$$= \frac{1}{m}(h_\Theta(x_i) - y)x_i$$

Therefore,

$$\Theta_j := \Theta_j - \frac{\alpha}{m}\sum_{i=1}^{m}[(h_\Theta(x_i) - y)x_i]$$

-> **?$_j$:** Weights of the hypothesis.
-> **h$_?$(x$_i$) :** predicted y value for ith input.
-> **j :** Feature index number (can be 0, 1, 2, ..., n).
-> **? :** Learning Rate of Gradient Descent.

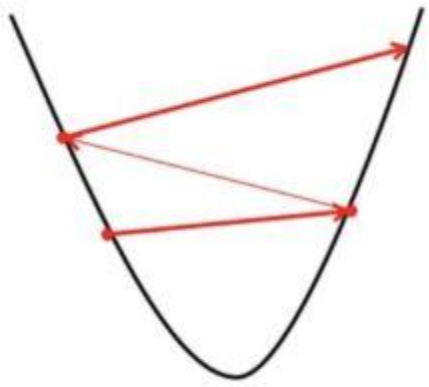
We graph cost function as a function of parameter estimates i.e. parameter range of our hypothesis function and the cost resulting from selecting a particular set of parameters. We move downward towards pits in the graph, to find the minimum value. Way to do this is taking derivative of cost function as explained in the above figure. Gradient Descent step downs the cost function in the direction of the steepest descent. Size of each step is determined by parameter? Known as **Learning Rate**.

In the Gradient Descent algorithm, one can infer two points :

1. **If slope is +ve** : ?$_j$ = ?$_j$ – (+ve value). Hence value of ?$_j$ decreases.
2. **If slope is -ve** : ?$_j$ = ?$_j$ – (-ve value). Hence value of ?$_j$ increases.

The choice of correct learning rate is very important as it ensures that Gradient Descent converges in a reasonable time. :

1. If we choose **? to be very large**, Gradient Descent can overshoot the minimum. It may fail to converge or even diverge.



2. If we choose ? to be very small, Gradient Descent will take small steps to reach local minima and will take a longer time to reach minima.

~~~~~~

2) Differentiate between logistic regression and decision tree classifier? What is the different convergence criteria used in building a decision tree classifier?

Logistic Regression and Decision Tree classification are two of the most popular and basic classification algorithms being used today. None of the algorithms is better than the other and one's superior performance is often credited to the nature of the data being worked upon.

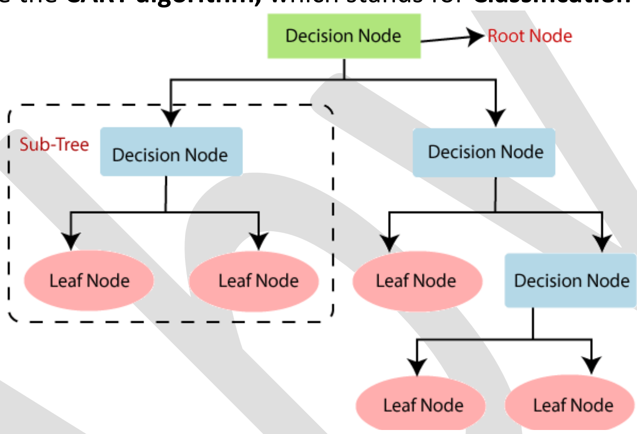
We can compare the two algorithms on different categories –

Criteria	Logistic Regression	Decision Tree Classification			
Interpretability	Less interpretable	More interpretable			
Decision Boundaries	Linear and single decision boundary	Bisects the space into smaller spaces			
Ease of Decision Making	A decision threshold has to be set	Automatically handles decision making	**Overfitting**	Not prone to overfitting	Prone to overfitting
Robustness to noise	Robust to noise	Majorly affected by noise			
Scalability	Requires a large enough training set	Can be trained on a small training set			

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes**
~~~~~~

represent the features of a dataset, branches represent the decision rules and **each leaf node represents the outcome.**

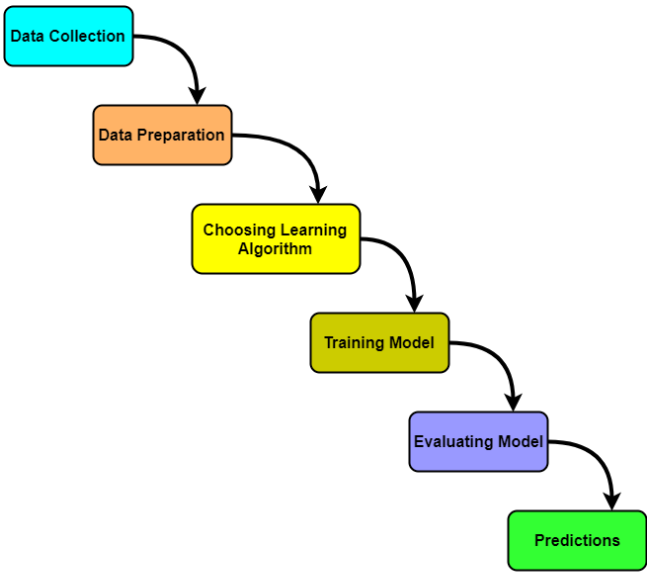
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**



3) Draw the machine learning model development process with functionality of each step?

<u>Machine Learning Workflow-</u>

Machine learning workflow refers to the series of stages or steps involved in the process of building a successful machine learning system. The various stages involved in the machine learning workflow are-



Machine Learning Workflow

1. Data Collection
2. Data Preparation
3. Choosing Learning Algorithm
4. Training Model
5. Evaluating Model
6. Predictions

Let us discuss each stage one by one.

1. Data Collection-

In this stage,

- Data is collected from different sources.
- The type of data collected depends upon the type of desired project.
- Data may be collected from various sources such as files, databases etc.
- The quality and quantity of gathered data directly affects the accuracy of the desired system.

2. Data Preparation-

In this stage,

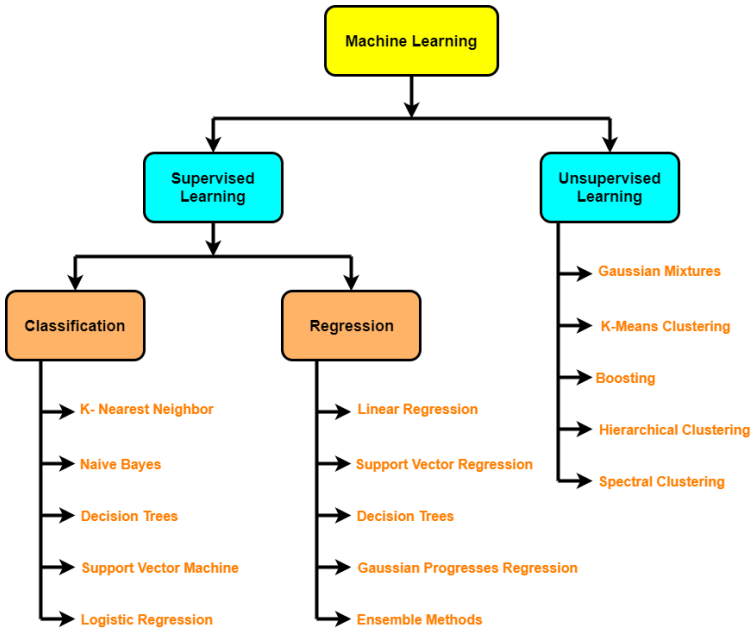
- Data preparation is done to clean the raw data.
- Data collected from the real world is transformed to a clean dataset.
- Raw data may contain missing values, inconsistent values, duplicate instances etc.
- So, raw data cannot be directly used for building a model.

Different methods of cleaning the dataset are-

- Ignoring the missing values
- Removing instances having missing values from the dataset.
- Estimating the missing values of instances using mean, median or mode.
- Removing duplicate instances from the dataset.
- Normalizing the data in the dataset.

This is the most time consuming stage in machine learning workflow.

3. Choosing Learning Algorithm- In this stage,The best performing learning algorithm is researched.



- It depends upon the type of problem that needs to solved and the type of data we have.
- If the problem is to classify and the data is labeled, classification algorithms are used.
- If the problem is to perform a regression task and the data is labeled, regression algorithms are used.
- If the problem is to create clusters and the data is unlabeled, clustering algorithms are used.

The following chart provides the overview of learning algorithms-

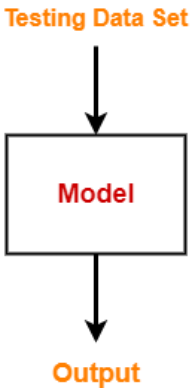
4. Training Model-

In this stage,

- The model is trained to improve its ability.
- The dataset is divided into training dataset and testing dataset.
- The training and testing split is order of 80/20 or 70/30.
- It also depends upon the size of the dataset.
- Training dataset is used for training purpose.
- Testing dataset is used for the testing purpose.
- Training dataset is fed to the learning algorithm.
- The learning algorithm finds a mapping between the input and the output and generates the model.

5. Evaluating Model- In this stage,

- The model is evaluated to test if the model is any good.
- The model is evaluated using the kept-aside testing dataset.
- It allows testing the model against data that has never been used before for training.
- Metrics such as accuracy, precision, recall etc are used to test the performance.
- If the model does not perform well, the model is re-built using different hyper parameters.
- The accuracy may be further improved by tuning the hyper parameters.



6. Predictions-

In this stage,

- The built system is finally used to do something useful in the real world.
- Here, the true value of machine learning is realized.

~~~~~~
~~~~~~