I. **Univariate, Bivariate and Multivariate Analysis**

Univariate Analysis: We just pick up one feature and try to see / classify those points w.r.t output. In the y-axis, we would have nothing as there is just one feature and the points would appear in a line parallel to the x-axis.
For example: we choose weight feature - then we get the clusters in line as slim, fit and obese just with the help of weight.
But, these easily classifiable points will not always be there; there would be many overlaps.
To tackle that, we would go with bivariate and multivariate analysis.

Bivariate Analysis: For example, we take height and weight features. Based on these points are classified. We see certain clusters with some overlaps. This overlap will also help us understand which machine learning algorithm we can use to classify. For example if there is lesser overlap, we can use logistic regression because in it we are using sigmoid function and we would get higher errors. So, in case of higher overlaps, we would choose non-linear algorithms such as decision tree, KNN, random forest etc.

Multivariate analysis: What if we have multiple features. So, in order to analyze such data, we use multivariate analysis. Seaborn pairplot helps us to visualize such data. For example, features: age, height, weight. This pairplot can lead to correlation. Whenever we see age and height, can we find out if age is increasing, if height is increasing. This would show positive correlation. If we are not able to find positive or negative correlation, it might have zero correlation. We can use Pearson correlation (value ranges from -1 to +1). Should I use some non-linear classifying plane/line to classify? - all these questions would be answered by a pairplot.

II. **Histograms**

Histograms help us to visualize the number of points within a particular category in univariate analysis as those points are not quite evident in a univariate analysis. Default bin count is 10. Y-axis shows the count of the number of values in a particular range. We can use the Matplotlib hist() function or seaborn histogram function. It would be respect to one feature. This figure would look like a curve (if bell, then the distribution might be normal / gaussian). The Bell curve is called the Probability density function (PDF). Then , we would be converting to PDF function - at this point of time what percentage of distribution is within that particular range.

III. **Z-Score statistics**

In a Gaussian or normal curve, when we go to the right of the mean, we get 1 standard deviation, then 2 standard deviation. On the left of the mean, we see -1 standard deviation, -2 standard deviation and so on. Within the first standard deviation of a Gaussian distribution, we have around 68% of our information, and within the second standard deviation we have around 95% information. And, if we want to convert this entire information to our standard normal distribution where mean is 0 and standard deviation is 1, we use Z-Score.

Z-Score = (x(i) - mean)/ s.d.

If I have {1,2,3,4,5} in our distribution, then mean = 3, s.d. = 1
Z-score = 3 - 3 / 1 = 0 for x(i) = 3
And so on.

If I want to find out 1.5 s.d. Away from the mean, then we use standard normal distribution and for getting that we use Z-score and Z-score table.

Example (Student data):
Mean = 75, s.d. 10, P(x > 60) = ?
Now, when we convert it to standard normal distribution, we see that our mean 75 gets converted to mean = 0; 65 gets converted to -1 and 60 gets converted to -.15. And, we need to find the region under the standard normal curve where s.d. > -1.5

For this we would use the Z-score table, which would give us the left hand side value of area under the curve for s.d. < -1.5.

Now, our curve has 3 regions, Z-score basically gives region 3 where s.d. < -1.5. Region 1 is for the region between s.d >= -1.5 and s.d. <= 0. Region 2 is the s.d. > 0.

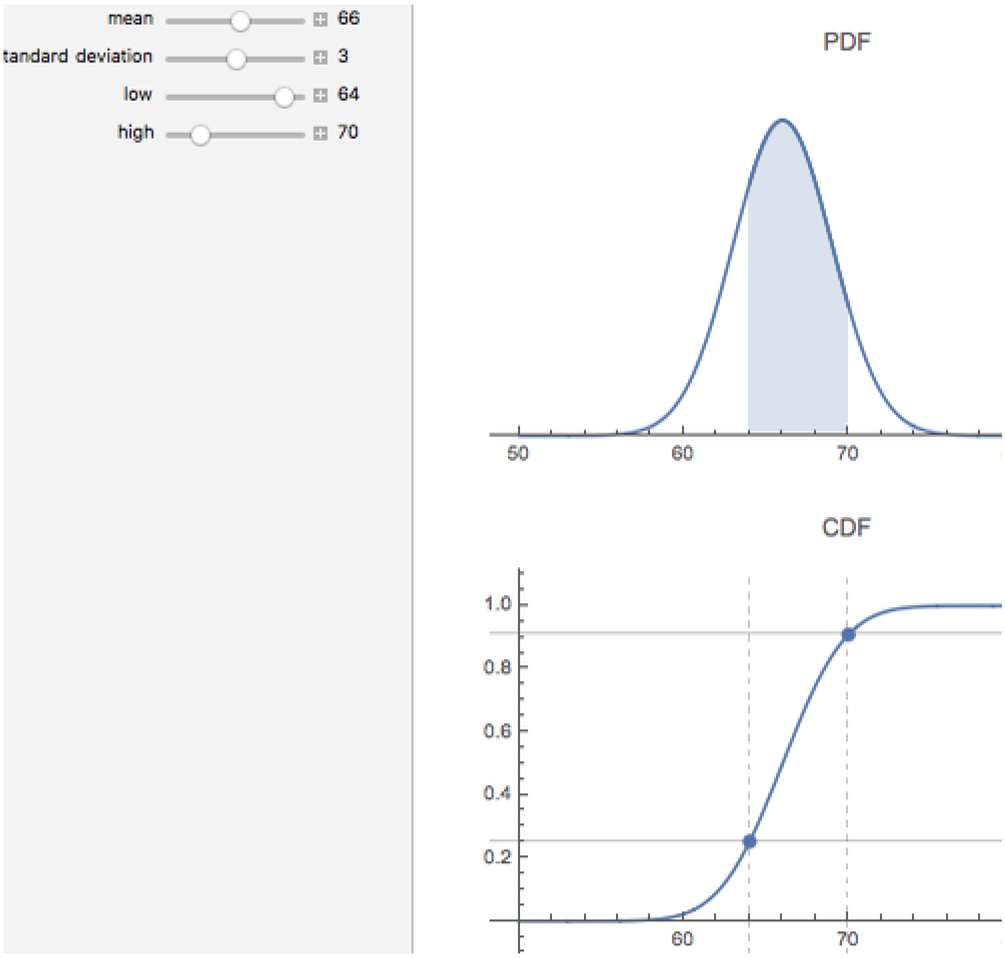
Region 3 value from the Z-score table is 0.0668 (6.68%). Now, we know that for a standard normal distribution, the curve is symmetrical around the mean, so region 2 is 50% of the overall value under the curve. Then, region 1 is (50 - 6.668)% i.e. 43.37% approximately. Then, the value of Region 1 and Region 2 combined would be around 94% approx.

IV. Probability Density Function:

Plotting all the points in the form of histograms shows on y-axis how many points are within a particular bin / range. If I want to convert it into a PDF or smoothen the histogram, it resembles a bell curve. As soon as we draw a bell curve, the count value gets replaced by % of the distribution on the y-axis.

It basically says that within this range, what is the % of the distribution present in the particular region. If that particular value is 0.2, which means 20% of the points distribution are in that particular region.

Cumulative density function (CDF) is different from PDF. In this , we basically add the percentage distribution for these points and the curve looks different from PDF.



Suppose, we take a point x(j) and we get 90% on the y-axis corresponding to it. This indicates that 90% of the distribution is less than ….x(j)…kgs. That also indicates that 10% of the distribution is greater than x(j) kgs.

V. Ridge and Lasso Regression

Regularization hypertuning techniques.
Sum of residuals in linear regression: $\sum(y' - y)^2$

Suppose there are two parameters: Experience and Salary. We try to create a best fit line using linear regression.

Suppose my training data has only two points which are far and fall almost on a line. We create a line and then calculate the sum of residuals to reduce the cost function. Now, in this case, for the training dataset, the points lie exactly on the line we made, the residuals would be zero. But, we wish to create a generalized model. Now, the test data might show high errors due to the huge distance from this line. So, this case is of overfitting.

For my training dataset, we have got a low bias / low error result. But, for test data, we see that the model is less generalized and shows high errors.

Now, we can use Ridge and Lasso regression to convert this high variance condition to a low variance condition.

How does Ridge and Lasso regression solve these problems?

Usually, in a linear regression model, Sum of residuals is given by $\sum(y' - y)^2$. But, in ridge regression, we add one more parameter. Cost function for ridge regression = $\sum(y' - y)^2 + \lambda*(slope)^2$.
So, if we see a steep curve in our linear regression model, we can deduce that it might lead to overfitting as with a unit increase in the experience, there is a quite high increase in the salary. In Ridge regression, we add $\lambda*(slope)^2$ to this model where we already see that the slope is quite high. We can assign 0 to any positive value to λ. Let us take it as 1 for now. Let us assume that the slope is 1.3 at present. If we do the calculation $\lambda*(slope)^2 = 1 *(1.3)(1.3) = 1.69$. Now, we will see the same value for other lines possible and see if they can reduce it.

Suppose residuals for this line might be a smaller value and slope might also be a smaller value as steepness has gone, thus, $\lambda*(slope)^2$ would also be a smaller number. Hence, cost reduces on the whole. So, we can use this line as the new best fit line.

We are basically penalizing features which have higher slopes to make the best fit line less steeper. If there is just one feature slope would be m. If we have 2 features, $\lambda*(slope)^2 = \lambda*(m1^2 + m2^2)$.

This will lead to less variance and better generalization. There would be slightly higher bias for the training dataset now but the test accuracy improves.

Cost function for Lasso regression = $\sum(y' - y)^2 + \lambda*|slope|$. It not only helps in regularization, but also helps in feature selection.

Suppose, we have y = m1x1 + m2x2 + m3x3 + m4x4 + c1. Now, as per Lasso regression, |slope| = |m1 + m2 + m3 + m4|. Wherever, the slope is very less, those features would get removed as they are not important / significant in making the prediction. This leads to feature selection as well. For Ridge regression, on the other hand, the $(slope)^2$ shrinks but never reduces to zero.

VI. Hypothesis Testing

P Values, T Test, Anova Test, Z Test, Type I and Type II Error

What exactly is hypothesis testing? Any data in statistics is useful only if you analyze it and deduce conclusions or inferences. In hypothesis testing, we actually evaluate 2 or more exclusive statements on a population using a sample of data.

Statement 1: The person is guilty
Statement 2: The person is innocent

Steps of hypothesis testing:

a. Make an initial assumption (Ho - null hypothesis - The person is innocent)
b. Collect data / evidences
c. Gather evidence to reject or not reject the null hypothesis

Alternate hypothesis, H1 - Opposite of null hypothesis. If we are able to prove Ho, then we consider it as true else due to lack of evidence, we consider H1 as true.

Suppose if a null hypothesis is actually true (defendant is innocent) but I do not have enough evidence. Then Ho gets rejected and H1 would be considered as true (the defendant will be treated as guilty). If we look at the confusion matrix as follows:

	Ho	**H1**
Do not reject	Ok	***Type 2 Error***: we took H1 as true as per evidence, but actually Ho was true
Reject	***Type I Error***: Due to the lack of evidence, I have to reject Ho even though it might be true.	Ok

Suppose. Ho: the market is going to crash and H1: the market is not going to crash.
I collected various pieces of evidence and found that H1 is true but actually Ho was true : the market crashed. This is type II error.

If P < 0.05 (significance value), then we reject Ho and consider H1.

VII. P Value

Based upon the number of touches in the center of the spacebar key, we can make a normal bell curve. It is also called a 2-tailed test.

Suppose for a particular point, P value is 0.01. This means that if we repeat this experiment 100 times, then the number of times we will be touching the space key in this area will be 1. Suppose the P value at a point is 0.8, then if we repeat this experiment 100 times, then the number of times we will be touching the space key in this area will be 80 times.

P-value is the probability for the null hypothesis to be true.
Null hypothesis: It treats everything same or equal

Consider I have a fair coin. If I am performing a specific toss experiment 100 times, based on fair nature at least 50 times head or tail should be coming up.

Null hypothesis, Ho - the coin is fair
H1: the coin is not fair
Now, when we perform an experiment of toss (100 times). Out of that we achieve: 67 times head, 33 times tail. 50 would be the mean value of the bell curve and 67 would be on its right. It would be good to get this value nearer to the mean. We see that the P-value is falling in the tail region , away from the mean. SO, we need to reject the null hypothesis and we cannot say that it is treating all the same. The coin is not fair as Ho is rejected.

5% of data is usually divided in two parts under the tail region and the rest 95% is there within the two tails.

Now, suppose, P value is 0.05 (significance). If my experiment comes with a P value and it lies in the tail region, then we will reject Ho definitely because it is likely not possible to be true.

If suppose , we get 55% head in 100 tosses, we notice that P value does not come in the extreme tail regions, so we accept the null hypothesis and say that it is really a fair coin.

VIII. Naive Bayes Classifier

Preparation Video:

https://www.youtube.com/watch?v=mlumJPFvooQ&list=PLZoTAELRMXVPkl7oRvzyNnyj1HS4wt2K-&index=2&ab_channel=KrishNaik

Theoretical Understanding:

1. Tutorial 48th : https://www.youtube.com/watch?v=jS1CKhALUBQ
2. Tutorial 49th: https://www.youtube.com/watch?v=temQ8mHpe3k

1. What Are the Basic Assumptions?

Features Are Independent

2. Advantages

1. Work Very well with many number of features

   Explanation: Because of the class independence assumption, naive Bayes classifiers can quickly learn to use high dimensional features with limited training data compared to more sophisticated methods. This can be useful in situations where the dataset is small compared to the number of features, such as images or texts. Naive Bayes implicitly treats all features as being independent of one another, and therefore the sorts of curse-of-dimensionality problems which typically rear their head when dealing with high-dimensional data do not apply.

   If your data has k dimensions, then a fully general ML algorithm which attempts to learn all possible correlations between these features has to deal with 2^k possible feature interactions, and therefore needs on the order of 2^k many data points to be performant.

However because Naive Bayes assumes independence between features, it only needs on the order of k many data points, exponentially fewer.

However this comes at the cost of only being able to capture much simpler mappings between the input variables and the output class, and as such Naive Bayes could never compete with something like a large neural network trained on a large dataset when it comes to tasks like image recognition, although it might perform better on *very* small datasets.

2. Works Well with Large training Dataset
3. It converges faster when we are training the model

    Explanation: Unlike other machine learning models, naive bayes require little to no training. When trying to make a prediction that involves multiple features, we simply use the maths by making the *naive* assumption that the features are independent.

4. It also performs well with categorical features

    Explanation: For a Naive Bayes classifier, categorical values are the easiest to deal with. All you are really after is P(Feature | Class). This should be easy for the days of the week. Compute P(Monday | Class=Yes) and so on.

3. Disadvantages

1. Correlated features affects performance

4. Whether Feature Scaling is required?

No. *In fact, any Algorithm which is NOT distance based, is not affected by Feature Scaling.* As Naive Bayes algorithm is based on probability not on distance, so it doesn't require feature scaling.

5. Impact of Missing Values?

Naive Bayes can handle missing data. Attributes are handled separately by the algorithm at both model construction time and prediction time. As such, if a data instance has a missing value for an attribute, it can be ignored while preparing the model, and ignored when a probability is calculated for a class value tutorial : https://www.youtube.com/watch?v=EqjyLfpv5oA

6. Impact of outliers?

It is usually robust to outliers.

One potential issue with outliers is that unseen observations can lead to 0 probabilities. And we know in naive bayes we multiply probab of words lying in that particular class and results zero. For example, Bernoulli Naive Bayes applied to word features will always produce 0 probabilities when it encounters a word that wasn't seen in the training data. Outliers in this sense can be a problem.

However, all these and similar issues of Naive Bayes have well-known solutions (like Laplace smoothing, i.e. adding an artificial count for every word) and are routinely implemented. In Gaussian Naive Bayes, outliers will affect the shape of the Gaussian distribution and have the usual effects on the mean etc. So depending on your use case, it still makes sense to remove outliers.

Different Problem statement you can solve using Naive Bayes

1. Sentiment Analysis
2. Spam classification
3. twitter sentiment analysis
4. document categorization

IX. Linear Regression Algorithm

Preparation Video:

Interview Prep Day 2- Linear Regression Interview Question-The Most Importa…

Theoretical Understanding:

1. https://www.youtube.com/watch?v=1-OGRohmH2s&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=29
2. https://www.youtube.com/watch?v=5rvnlZWzox8&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=34
3. https://www.youtube.com/watch?v=NAPhUDjgG_s&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=32
4. https://www.youtube.com/watch?v=WuuyD3Yr-js&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=35
5. https://www.youtube.com/watch?v=BqzgUnrNhFM&list=PLZoTAELRMXVPBTrWtJkn3wWQxZkmTXGwe&index=33

Theoretical Concepts:

$y=mx + c$; m = slope, c = intercept

We need to find the best fit line. If my x is 0, then y is c i.e. Y-intercept. Within a unit change in X-axis, the change in the y-value is called slope or m. $m = (y2 - y1) / (x2 - x1)$

The summation of squared error should be minimized by the requisite values of m and c obtained. Here, cost function = distance b/w the best fit point and the actual point should be minimum. Thus, cost function = $1/2n \sum(y' - y)^2$ where i ranges from 1 to m. Here, m is the number of points.

Predicted points: y' and actual points: y. But, if we just use the above equation, we might get several best fit lines, and choosing just one might be time consuming.

So, what can we do?

Example: $y = x$; $y' = mx + c$; here, we can assume the best fit line passes through origin and $c = 0$; thus $y' = mx$

Now, we can substitute $x = 1$, and $m = 1$; then $y' = 1$;

For $x = 2$, $y' = 2$ and $y = 2$ and so on. This is actually my best fit line when my slope is 1. After getting this equation, we calculate our cost function and try to reduce it.

Here, cost function, $J(m) = 1/2n((1-1)^2 + (2-2)^2 + (3-3)^2) = 0$

With respect to every m value, we can plot our cost function. For $m = 1$, $J(m) = 0$.

For $m = 0.5$, $x = 1$; $y = 0.5$,

For $x= 2$, $y = 1$,

For $x = 3$, $y = 1.5$

Then our cost function, $J(m) = 1/2n((1-0.5)^2 + (2-1)^2 + (3-1.5)^2) = 0.58$. Thus, a curvature of J(m) versus m can be plotted and we can see the gradient descent. How do we arrive at the global minimum?

Based on some m value, we get some initial J(m) value. In order to move downwards, we should use the convergence theorem.

Convergence theorem: $m = m - \alpha \partial J/\partial m$ where α is the learning rate

If the slope ($\partial J/\partial m$) is negative, the curve points downwards. When we get a negative slope, then m = m + (+ive smaller value). This step would be very very small and it would move slowly towards global minimum. If we take a larger alpha, then the jumps might be bigger and it might not converge after several iterations and keep oscillating.

At the global minima, the slope will be zero. This would be the slope of the best fit line. Until then, we would keep following the convergence theorem.

If I have multiple independent features, then each of the features will try to reach a global minimum.

For multiple linear regression, $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5$; here, β_0 is the y-intercept; β_1, β_2 and β_3 are changes in y with unit changes in x_1 , x_2 and x_3 respectively.

Multicollinearity: In regression, "multicollinearity" refers to predictors that are correlated with other predictors. Multicollinearity occurs when your model includes multiple factors that are correlated not just to your response variable, but also to each other. In other words, it results when you have factors that are a bit redundant. To handle such a multicollinearity situation, one solution is to remove highly correlated feature (check for P value in model summary and remove the one for which it is higher).

R Square and Adjusted R Square:

$R^2 = 1 - SSres/SStot$

Here, $SSres$ is the sum of squares of residuals or errors = $\sum(y' - y)^2$

$SStot$ is the sum of average totals = $\sum(y' - y_{mean})^2$

The closer the value of R^2 to 1, the better the model is.

It will be less than zero only when the best fit line is worse than the average value.

As we go on adding new independent features, our R^2 value usually increases. This is because the model then tries to apply some coefficient value such that our $SSres$ value decreases. Then, resultantly, our R^2 value increases. We should also note that this value will never decrease when we keep on adding independent features to it. But, there might be a scenario that the independent feature being added might not be correlated to the target output. Even though R^2 value might show an increase in such a case, but it is basically not penalizing the newly added features which do not have any correlation with the target. For this reason, we use adjusted R square.

$$Adjusted\ R^2 = 1 - \frac{(1 - R^2)(N - 1)}{N - p - 1}$$

Where

R^2 Sample R-Squared

N Total Sample Size

p Number of independent variable

From the above formula, we can note that as the value of p increases when independent features which are not correlated to the target variable are added, N - p - 1 value decreases, leading to an overall decrease in the Adjusted R square value as a higher term is subtracted from 1 to achieve it. In case, the added features have correlation with the target variable, then R^2 value would be higher such that the rest of the multiplication factor will become overwhelmed and would not make much of a difference to the overall R^2 value.

Differences between R^2 value and Adjusted R^2 value:

Every time an independent variable is added to a model, R^2 value increases, even if the independent variable is insignificant. It never declines, whereas adjusted R^2 value increases only when the independent variable is significant and affects the dependent variable.

Adjusted R^2 value is always less than or equal to R^2 value.

Interview Question on Multicollinearity

1. https://www.youtube.com/watch?v=tcaruVHXZwE

Removing the highly correlated feature would work when we have quite fewer numbers of features and a smaller dataset. We also lose certain information when we remove some data. In case, we have a large dataset, we would go for Ridge or Lasso correlation.

1. What Are the Basic Assumptions?(favorite)

There are four assumptions associated with a linear regression model:

1. ***Linearity***: The relationship between X and the mean of Y is linear.
2. ***Homoscedasticity***: The variance of residual is the same for any value of X.
3. ***Independence***: Observations are independent of each other.
4. ***Normality***: For any fixed value of X, Y is normally distributed.

2. Advantages

1. Linear regression performs exceptionally well for linearly separable data
2. Easy to implement and train the model
3. It can handle overfitting using dimensionality reduction techniques and cross validation and regularization

3. Disadvantages

1. Sometimes Lot of Feature Engineering Is required

2. If the independent features are correlated it may affect performance
3. It is often quite prone to noise and overfitting

4. Whether Feature Scaling is required?

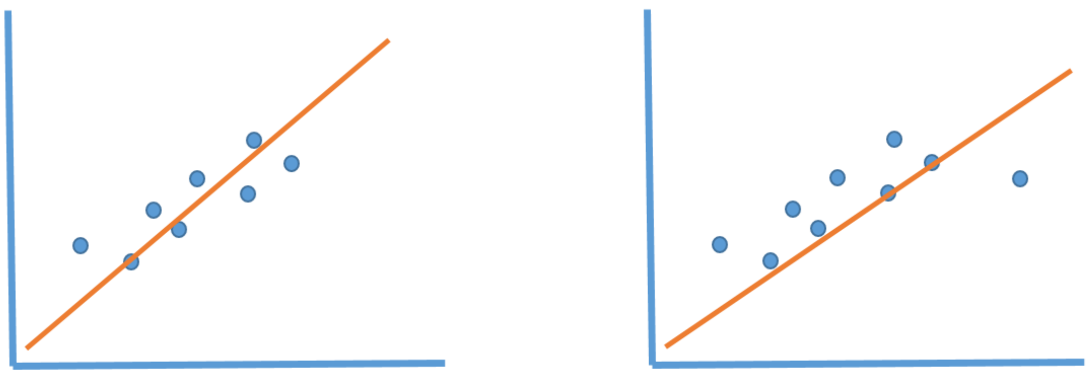
Yes (Whenever we talk about gradient descent, we need to do feature scaling because that will help us to reach the optimum solution / global minima very quickly.)

5. Impact of Missing Values?

It is sensitive to missing values (feature engineering is used to handle it)

6. Impact of outliers?

Linear regression needs the relationship between the independent and dependent variables to be linear. It is also important to check for outliers since linear regression is sensitive to outlier effects. Just to reduce the mean squared error (MSE), the line is changing in the below figure with the appearance of an outlier. To handle it, ridge and lasso regression are used.



Types of Problems it can solve(Supervised)

1. Regression

Overfitting And Underfitting

We will use polynomial linear regression to explain bias and variance tradeoff as well as overfitting and underfitting. Please note that if the degree of the polynomial is one, then the curve is a straight line. The sum of mean square error (cost function) is higher in that scenario when compared to a polynomial curve with degree greater than one. When error is very high for a training dataset, then the particular scenario is called *underfitting*. Suppose, we use a quite higher order polynomial such that almost all the training data points are fitted quite well by the model, this scenario is called *overfitting*. This is due to the fact that the accuracy would go down for the test data even though it is very high for the training data.

Our main aim is to achieve an optimum solution such that accuracy is high for both training and test data. That model will give us low bias and low variance.

In an underfitting scenario, we basically have high bias (error of the training data) and high variance (generalizability - error on the test data). For overfitting scenario, we have low bias but high variance.

Different Problem statement you can solve using Linear Regression

1. Advance House Price Prediction
2. Flight Price Prediction

Practical Implementation
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

X. Support vector Machines (SVM)

Preparation Video:
https://www.youtube.com/watch?v=eYRBWM9Mvuc&list=PLZoTAELRMXVM0zN0cgJrfT6TK2ypCpQdY&index=8&ab_channel=KrishNaik

Theoretical Concepts:

The main aim of SVM is to create a hyperplane and two separating margin lines such that there are two more hyperplanes in parallel to the particular hyperplane and these hyperplanes pass through the nearest neighbouring point in the two clusters.

This whole distance between the r=two parallel hyperplanes - d+ive and d-ive is called margin. We aim to get a generalised model so that it is easily applicable on unseen points. This margin and hyperplane helps in such generalisation.

Now, we can create multiple hyperplanes separating the two clusters apart from this one hyperplane. Then, why choose this hyperplane. Our main aim is to select the hyperplane which gives us the maximum marginal distance. This linear hyperplane is for linearly separable points.

What if the points are non-linearly separable?
A linearly separable hyperplane cannot be made for such a dataset. For tackling this we use kernels which convert 2 -dimension to a higher dimension. Then, between the higher dimensional dataset, we can construct linearly separable hyperplanes

What is a support vector?
The nearest positive point and nearest negative point that passes through the parallel marginal hyperplanes are called support vectors. It may have multiple support vectors.

SVM Maths Intuition:

Let us consider a simple example of logistic regression where we have two points (4,4) and (-4,0) in a 2-D plane. We wish to separate them using a linear hyperplane. Now, suppose the slope of such a linear plane is -1. The equation of this line is given as
$w'x + b = 0 \; where \, ' \, denotes \; the \; transpose.$
We also know this equation: y = mx + c; m = -1 (assumption made).
c = b = 0 as the line has been assumed to be passing through origin.Thus, y = $w'x$
$y = [-1\ 0]'[-4\ 0] = 4 \; (this\ value\ is\ always\ going\ to\ be\ positive)$

Anytime, we calculate y for points below the linear separator, y is going to be positive. And, when we calculate it for points above the linear separator, y is going to be negative. Now, we can take this value as + 1 and -1 for two clusters respectively.

In SVM, this hyperplane equation can also be given similar to logistic regression:
$w'x + b = 0.$
I am going to find the nearest points for the hyperplane in the two clusters. Suppose, it's at a distance -1 and + 1 respectively as assumed above.
So, the equations can be $w'x + b = -1 \, and \, w'x + b = 1$ respectively.

Now, here b will not be zero as the hyperplane is not passing through zero. Now, we basically wish to compute the distance between two points x1 and x2 lying on these two parallel hyperplanes. We can write the equations as follows:
$w'x1 + b = -1 \, and \, w'x2 + b = 1$. So, we can subtract the two equations:
$w'(x2 - x1) = 2.$
To remove w' , we are going to divide by ||w|| on both sides. So, 2/||w|| would be our optimization function and we need to maximise this.
We need to update (w,b) to maximise the optimization function such that y = 1 when $w'x + b$ >= 1 and y = -1 when $w'x + b$ <= -1.

We can also write it as: y*($w'x + b$) >= 1

If it is not greater than equal to 1, then it means that it is a misclassification.

One point to note is that in a real world scenario, we will not have such separable data points. There would be a lot of overlap.

We have to change (w*, b*) such that min(||w||/2) + c∑**ζ**i where i ranges from 1 to n. 'c' represents how many errors the model can consider; **ζ** is the value of the error. This value 'c' is called regularisation and is obtained by hyperparameter tuning. This particular SVM is called linear SVM and the margin is a ***hard margin***.
(w*, b*) = min(||w||/2) + c∑**ζ**i where i ranges from 1 to n

SVM Kernels:

1. Polynomial Kernels
2. RBF Kernels
3. Sigmoid Kernels

In case of soft margin, there would be certain consideration for error. Now, non-linearly separable data can be handled using the SVM kernel which will convert a lower dimensional non-linearly separable dataset into a linearly separable higher dimensional dataset. This transformation is done by SVM Kernel from lower dimension to higher dimension. It does it using some mathematical formulas.

Let us consider one-dimensional points for example. —------*****-----****-------******------

To classify these points, in order to use SVM, we first need to draw a hyperplane but they are 1-dimensional. So, we would use a kernel to make this dataset 2-dimensional. Let us suppose, we are using a Polynomial kernel ($y = f(x) = x^2$). So, these points get projected as a parabola. Then, we can draw a hyperplane to separate these points.

Polynomial Kernel: Let us consider we have elliptical data points which are non-linearly separable in x1 and x2 plane. We will use a polynomial kernel to make the data points linearly separable in a higher dimension.

The formula for such a polynomial kernel would be: f(x1, x2) = (x1'.x2 + 1)^d where d is the higher order dimension.

The unique elements in x1 and x2 multiplication would be $x1^2$ and $x2^2$ whereas the repetitive parts would be x1x2. So, we see that our 2-dimensional points are becoming higher-dimensional - x1, x2, $x1^2$, $x2^2$, x1x2 (axes: x1, x2 and x1x2) such that we can obtain linearly separable data points and hyperplane.

Theoretical Understanding:

1. https://www.youtube.com/watch?v=H9yACitf-KM
2. https://www.youtube.com/watch?v=Js3GLb1xPhc

1. What Are the Basic Assumptions?

There are no such assumptions

2. Advantages

1. SVM is more effective in high dimensional spaces. : the reason that SVMs work well with high-dimensional data is that **they are automatically regularized**, and regularization is a way to prevent overfitting with high-dimensional data.
2. SVM is relatively memory efficient. (It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.)
3. SVM's are very good when we have no idea on the data.
4. Works well with even unstructured and semi structured data like text, Images and trees.
5. The kernel trick is the real strength of SVM. With an appropriate kernel function, we can solve any complex problem.
6. SVM models have generalisation in practice, the risk of overfitting is less in SVM.

3. Disadvantages

1. More Training Time is required for larger dataset
2. It is difficult to choose a good kernel function https://www.youtube.com/watch?v=mTyT-oHoivA
3. The SVM hyperparameters are Cost -C and gamma. It is not that easy to fine-tune these hyper-parameters. It is hard to visualise their impact

4. Whether Feature Scaling is required?

Yes - Feature scaling is crucial for some machine learning algorithms, which **consider distances between observations because the distance between two observations differs for non-scaled and scaled cases**. As we've already stated, the decision boundary maximises the distance to the nearest data points from different classes.

5. Impact of Missing Values?
Although SVMs are an attractive option when constructing a classifier, SVMs do not easily accommodate missing covariate information. Similar to other prediction and classification methods, in-attention to missing data when constructing an SVM can impact the accuracy and utility of the resulting classifier.

6. Impact of outliers?
It is usually sensitive to outliers
https://arxiv.org/abs/1409.0934#:~:text=Despite%20its%20popularity%2C%20SVM%20has,causes%20the%20sensitivity%20to%20outliers.

The penalty on misclassification is defined by a convex loss called the hinge loss, and the unboundedness of the convex loss causes sensitivity to outliers.

Types of Problems it can solve(Supervised)

1. Classification
2. Regression

Overfitting And Underfitting
In SVM, to avoid overfitting, we choose a Soft Margin, instead of a Hard one i.e. we let some data points enter our margin intentionally (but we still penalise it) so that our classifier don't overfit on our training sample

https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Different Problem statement you can solve using SVM

1. We can use SVM with every ANN use cases
2. Intrusion Detection
3. Handwriting Recognition

Practical Implementation

1. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
2. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

Performance Metrics

Classification

1. Confusion Matrix
2. Precision,Recall, F1 score

Regression

1. R2,Adjusted R2
2. MSE,RMSE,MAE

Which all algorithms are impacted by an imbalance dataset?

Machine Learning Algorithms such as Logistic Regression, KNN,SVM (basically which includes Gradient Descent and Euclidean Distance Computation) are affected by Imbalanced Datasets.

XI. Decision Tree Classifier And Regressor

Preparation Video:

Interview Prep Day 4-How To Learn Machine Learning Algorithms For Interview…

Interview Questions:

1. Decision Tree
2. Entropy, Information Gain, Gini Impurity
3. Decision Tree Working For Categorical and Numerical Features
4. What are the scenarios where Decision Tree works well
5. Decision Tree ***Low Bias And High Variance***- Overfitting
6. Hyperparameter Techniques
7. Library used for constructing decision tree
8. Impact of Outliers Of Decision Tree
9. Impact of missing values on Decision Tree
10. Does Decision Tree require Feature Scaling

Theoretical Understanding:

1. Tutorial 37:Entropy In Decision Tree https://www.youtube.com/watch?v=1IQOtJ4NI_0

   Entropy: f1 , f2, f3 etc. features; o/p is yes or no

   ID3 Algorithm → which node to select for split up

   Here, is when entropy comes into picture to select the right attribute for splitting purposes.

   Entropy helps us to measure the purity of the split.

   Suppose we split f1. Initially, we had 9 yes and 5 No’s. After the split, at f2 we had 3 yes, 2 No and at f3 we had 6 yes / 3 No. Ideally, after the split we should have all the Yes on one side and all the No on another.

   Again the split happens, this time we get 3 Yes, 0 No ; 2 No and 0 Yes on one side. We get 6 Yes, 0 No; 3 No, 0 Yes on the other side after the split. In order to get the correct leaf nodes fast, we need to select the right parameters / features of the dataset for split.

   We have to go and calculate the purity split at each split.

   Entropy: $H(S) = -P_{+}\log_2(P_{+}) - P_{-}\log_2(P_{-})$

P_+/P_- : % of +ive cases / % of -ive cases

S: subset of training sample

Entropy for above example:

$H(S) = -3/5\log_2(3/5) -2/5\log_2(2/5)$

H(S) = 0.78 bits APPROX.

Note: If we have a completely impure subset (3 Yes, 3 No), then its entropy is 1 bit. And, for a pure split (4 yes, 0 no) , entropy is 0 bits.

So, we would check for entropy at f2 and f3 for above example and select the one for which entropy is better. But, this calculation is just for one node. We should also consider other attributes which are needed for reaching pure leaf nodes. For that, we use Information gain. Basically, it calculates total entropy value from top to the leaf node to decide which path is better. Entropy ranges between 0 and 1.

2. Tutorial 38:Information Gain https://www.youtube.com/watch?v=FuTRucXB9rA

   Information Gain: In short, we can say that the average of all the entropy is based on a specific split. Suppose, based on f2 we wish to split into f1 and f3; or based on f1 into f2 and f3. We need to decide which split would be better to reach leaf nodes earlier. For that we need information gain.

   $Gain(S,A) = H(S) - \sum |S_v| / |S| (H(S_v))$ where v belongs to val

   Here, S_v is the subset after the split, S is the total subset, H(S) is the entropy; $H(S_v)$ is the entropy after the splitting for the subset

   Suppose, f1 (9Y, 5N) is splitting into f2 and f3. For f2, we get (6Y, 2N) and for f3 (3Y, 3N).

   Here, H(f1) = H(S) i.e. entropy before the split = 0.94

   $H(S_v)$ is the entropy after the split

   H(f2) = 0.81 ; H(f3) = 1

   Thus, gain = 0.91 - (8/14 * 0.81) - (6/14 * 1) = 0.049

   Instead, I can also go from f2 splitting into f1 and f3. So, we will check which one gives a higher information gain and we will select that particular way of splitting.

3. Tutorial 39:Gini Impurity https://www.youtube.com/watch?v=5aIFgrrTqOw

   Gini Impurity: Both entropy and gini impurity do the same task which is to calculate the purity of the split. But, in most cases Gini impurity is preferred. Let us see why?

   This is computationally more efficient and takes a lesser amount of time. For entropy, logarithmic calculations take some amount of time whereas in GI , we do not have any logarithmic calculations.

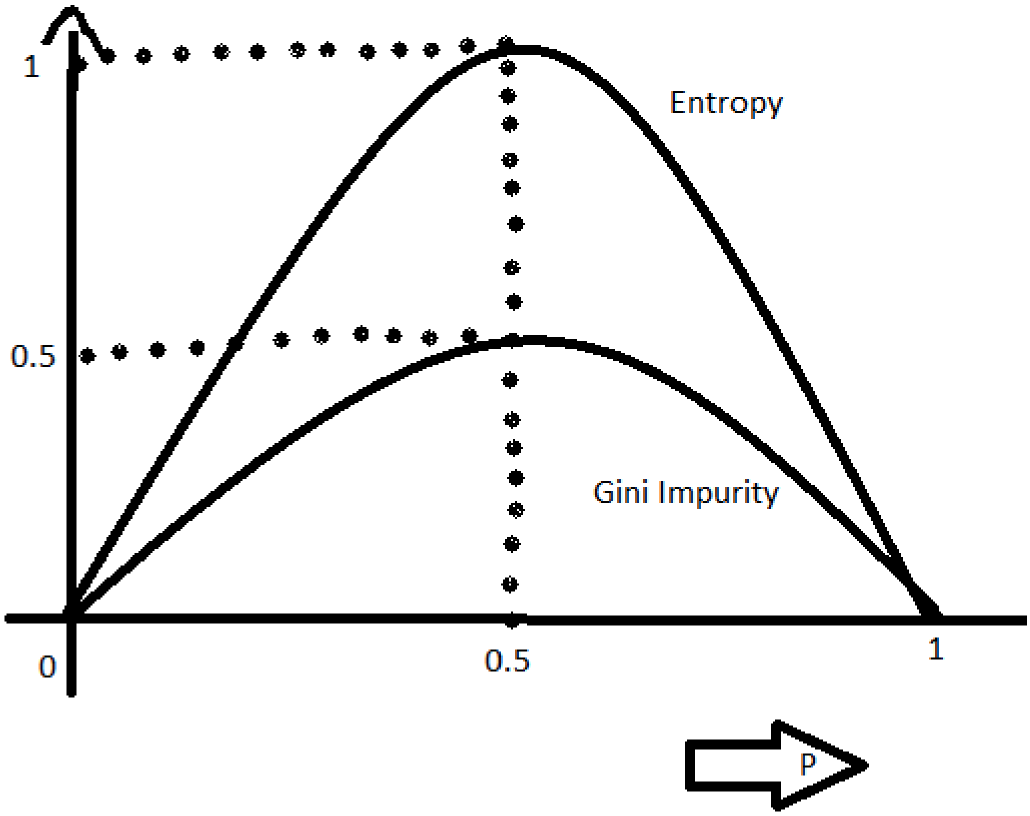
   F1 (6Y/3N) → f2 (3Y/3N) and f3 (3Y/0N)

Entropy → 0.5 → 1 and 0 respectively

Now, Gini Impurity, GI = 1 - $\sum(P^2)$ where i ranges from 1 to n

Where P^2 is the summation of squares of P_+ and P_-

GI = 1 - [(3/6)^2 + (3/6)^2] = 1 - [0.25 + 0.25] = 0.5 , whereas entropy for f2 was 1. So, we are getting less GI when compared to entropy.



$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

$$Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$$

4. Tutorial 40: Decision Tree For Numerical Features: https://www.youtube.com/watch?v=5O8HvA9pMew

If your feature is a numerical variable, the first step decision tree algorithm does is sorting all the values. It will try to consider some threshold values and is going to check for each and every record for a particular feature. If that xi is less than or equal the threshold, it will create a branch for it

In the given example, f1 (4yes, 4 no) , threshold is 2.3; and it is split; on one side there is just 1 Yes. On the other side (>2.3) we have 3Yes / 4No. The next threshold would be considered the next value, say 3.6. Again the split will happen for <= 3.6 (2Yes / 0No) and > 3.6 (1Yes/4 No). For this entropy and information gain is calculated and then decided based on which has the better value and that split is taken.

There is a disadvantage here. Suppose, we have millions of records. The time complexity would keep on increasing with so many samples. Decision trees with numerical features will take more time for training. The same disadvantage is true for ensemble methods.

5. How To Visualize DT: https://www.youtube.com/watch?v=ot75kOmpYjI

1. What Are the Basic Assumptions?
There are no such assumptions

2. Advantages
Advantages of Decision Tree

1. ***Clear Visualization*****:** The algorithm is simple to understand, interpret and visualize as the idea is mostly used in our daily lives. Output of a Decision Tree can be easily interpreted by humans.
2. ***Simple and easy to understand***: Decision Tree looks like simple if-else statements which are very easy to understand.
3. Decision Tree can be used for ***both classification and regression problems.***
4. The Decision Tree can handle ***both continuous and categorical variables***.
5. ***No feature scaling required***: No feature scaling (standardization and normalization) required in case of Decision Tree as it ***uses rule based approach instead of distance calculation.***
6. ***Handles non-linear parameters*** efficiently: Non linear parameters don't affect the performance of a Decision Tree unlike curve based algorithms. So, if there is high non-linearity between the independent variables, Decision Trees may outperform as compared to other curve based algorithms.
7. Decision Tree can ***automatically handle missing values***.
8. The Decision Tree is usually ***robust to outliers*** and can handle them automatically.
9. ***Less Training Period***: Training period is less as compared to Random Forest because it generates only one tree unlike forest of trees in the Random Forest.

3. Disadvantages
Disadvantages of Decision Tree

1. ***Overfitting***: This is the main problem of the Decision Tree. It generally leads to overfitting of the data which ultimately leads to ***wrong predictions***. In order to fit the data (even noisy data), it keeps generating new nodes and ultimately the ***tree becomes too complex to interpret.*** In this way, it ***loses its generalization capabilities***. It performs very well on the trained data but starts making a lot of mistakes on the unseen data.
2. ***High variance***: As mentioned in point 1, Decision Tree generally leads to the overfitting of data. Due to the overfitting, there are very high chances of ***high variance*** in the output

which leads to many errors in the final estimation and shows high inaccuracy in the results. In order to achieve zero bias (overfitting), it leads to high variance.

3. ***Unstable***: Adding a new data point can lead to re-generation of the overall tree and all nodes need to be recalculated and recreated.
4. ***Not suitable for large datasets***: If data size is large, then one single tree may grow complex and lead to overfitting. So in this case, we should use Random Forest instead of a single Decision Tree.

4. Whether Feature Scaling is required?

No

6. Impact of outliers?

It is ***not sensitive to outliers***.Since, extreme values or outliers, never cause much reduction in RSS, they are never involved in split. Hence, tree based methods are insensitive to outliers.

Types of Problems it can solve(Supervised)

1. Classification
2. Regression

Overfitting And Underfitting

Ho to avoid overfitting

https://www.youtube.com/watch?v=SLOyyFHbiqo

Practical Implementation

1. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
2. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

Performance Metrics

Classification

1. Confusion Matrix
2. Precision,Recall, F1 score

Regression

1. R2,Adjusted R2
2. MSE,RMSE,MAE

XII. Natural Language Processing(NLP)

Machine Learning Libraries: SpaCy, NLTK
Deep learning Libraries: PyTorch, Keras, TensorFlow
BERT, Transformers: Hugging Face Libraries