Assignment Name - Analytics Advanced

Problem Statement - Answer the following questions to the best of your knowledge including the concepts taught to you in the level.

1. On what basis we choose data scaling method (Normalization/Standardization)?
2. If the VIF is 2 then what is value of correlation coefficient (r^2)
3. How do you interpret chi-square result?
4. Why do we choose boxplot method than other for outlier detection and removal?
5. How do we choose best method to impute missing value for a data

Ques 1: On what basis we choose data scaling method (Normalization/Standardization)?

Ans: Feature/Data scaling is a method to limit the variables so that they can compare on common ground. Data Preprocessing is the way to extract the meaningful information out of the data. There are two basic methods

1. Scaling
2. Normalization
3. Standardization/Z score Method.

Scaling: Scale generally means to change the range of the values. The shape of the distribution doesn't change. Think about how a scale model of a building has the same proportions as the original, just smaller. That's why we say ,it is drawn to scale. The range is often set at 0 to 1.

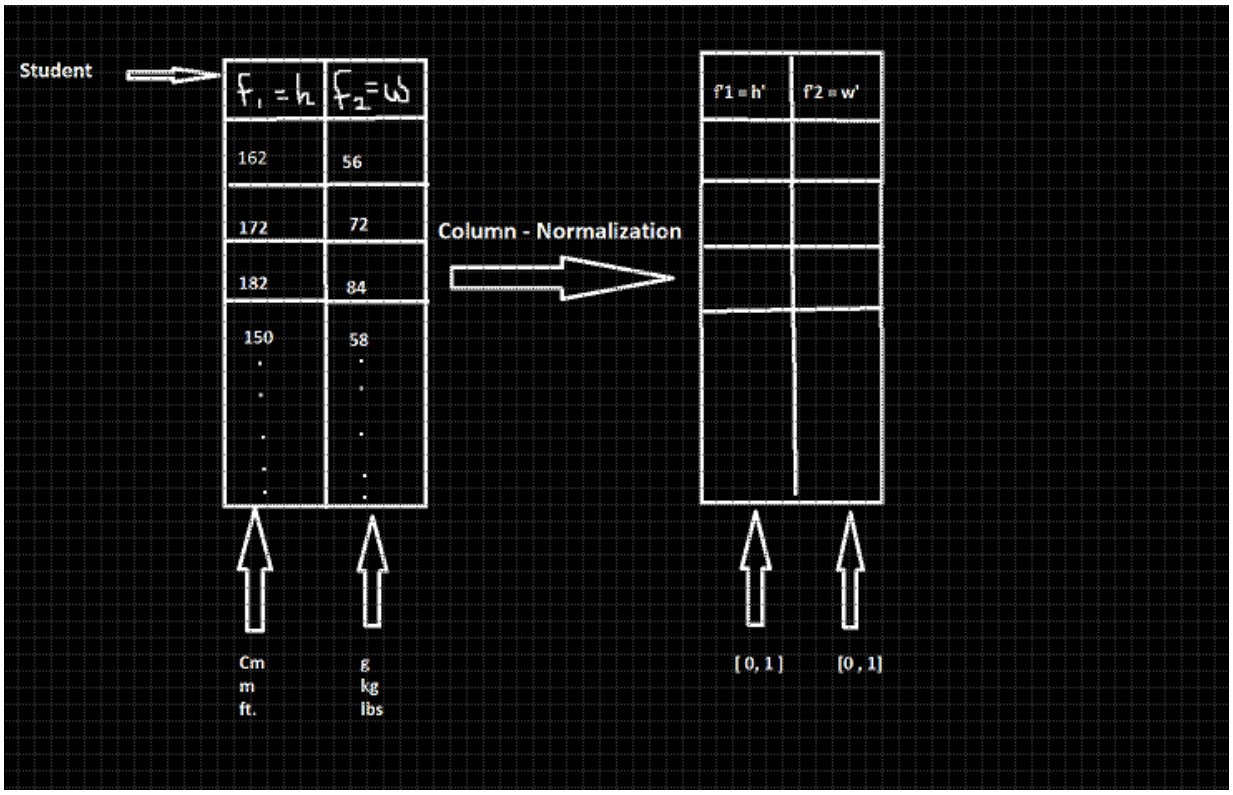
Normalization: Normalizing attribute data is used to rescale components of a feature vector to have the complete vector length of 1. This usually means dividing each component of the feature vector by the Euclidean length of the vector but can also be Manhattan or other distance measurements. This pre-processing rescaling method is useful for sparse attribute features and algorithms using distance to learnsuch as KNN. Sensitive to outliners.

- It is the process of reducing unwanted variation either within or between variables.
- Normalization to bring all of the variables into proportion with one another. In a common scale which ranges between 0 to 1.

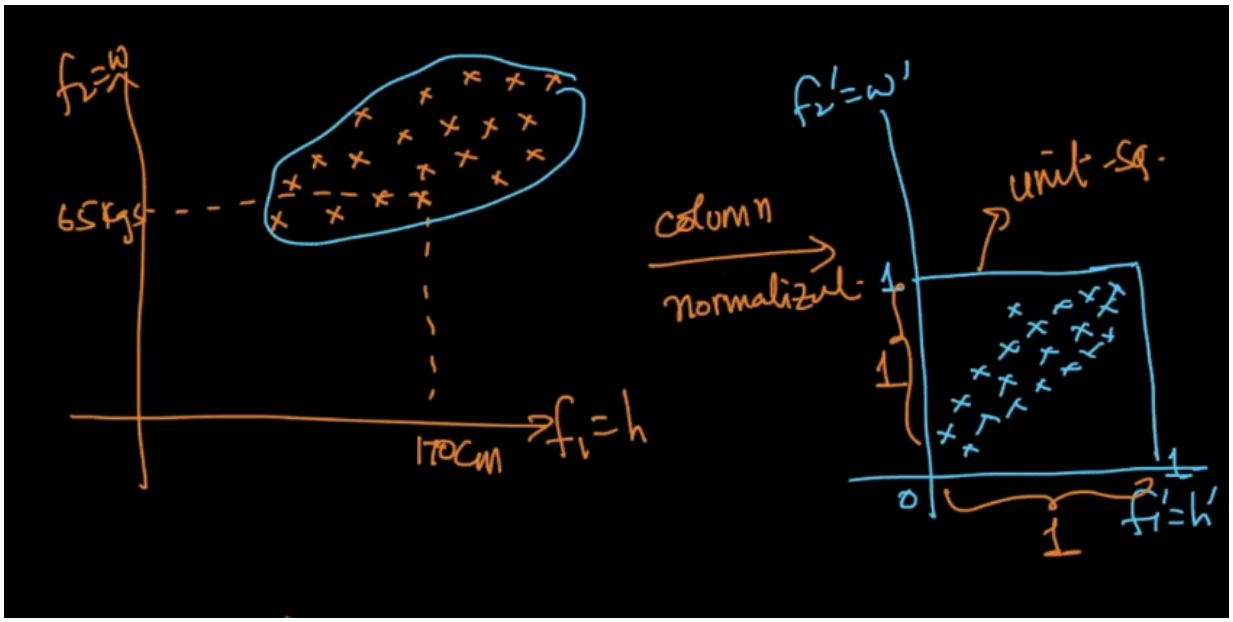
Example: We have a dataset of students in a class. features are like height and weight of the students.

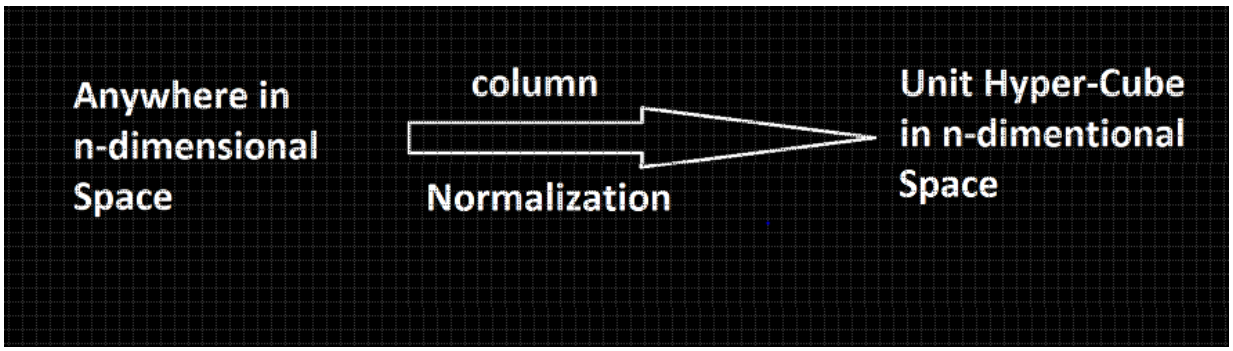
Height (h) are collected in centimetres (cm) and weight are collected in kilograms (kg) and each row represent a student individually. Suppose the same data are collected by other person where height are in feets (ft) and weight are in pounds (lbs). These acn be any values no bound on them, but we can observe the value is changing w.r.t units of features.

In real world, data will be present in any units or scale, using normalization we can give a standard unit so values cannot be varied w. r, t scale/units all are comes under same range.



Every values are now, lies in betwween [0, 1]





Observation:

1. Here, we are collection data from student , featurizing with their heights and weights and plot it geometrically, in 2D space using coordinate geometry.
2. After normalization, we Squished the data into a unit square (because it is a @D space), unit cube (for 3D space) and unit hyper cube (for n-dimentional space)

Standardization/Z score Method: Standardizing attribute data assumes a Gaussian distribution of input features and "standardizes" to a mean of 0 and a standard deviation of 1. This works better with linear regression, logistic regression and linear discriminate analysis. Python StandardScaler class in scikit-learn works for this. Works well with the data which is uniformly distributed.

Z = (point - mean) / standard deviation

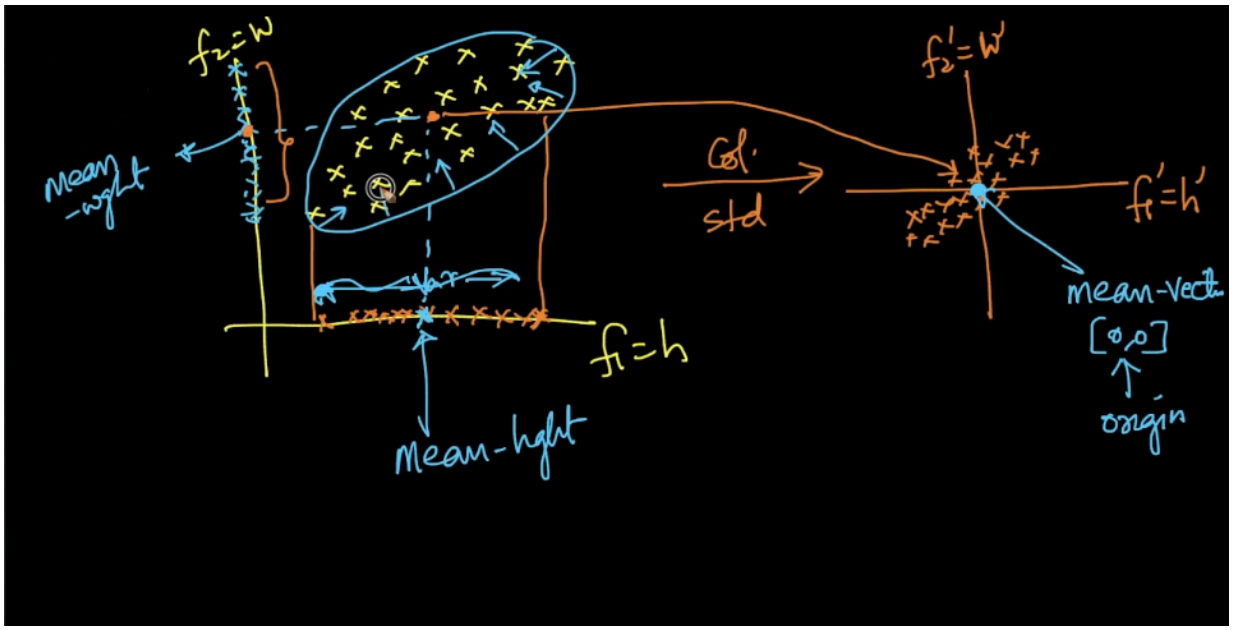
The result of standardization (or Z-score normalization) is that the features will be rescaled so that they'll have the properties of a standard normal distribution with μ=0 and σ=1 where μ, is the mean (average) and σ is the standard deviation from the mean; standard scores (also called z scores) of the samples are calculated as follows:

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

Standardizing the features so that they are centered around 0 with a standard deviation of 1 is not only important if we are comparing measurements that have different units, but it is also a general requirement for many machine learning algorithms. Intuitively, we can think of gradient descent as a prominent example (an optimization algorithm often used in logistic regression, SVMs, perceptrons, neural networks etc.); with features being on different scales, certain weights may update faster than others since the feature values x_j play a role in the weight updates

$\Delta w_j = -\eta * \partial_J / \partial w_j = \eta * \sum_i (t^{(i)} - o^{(i)}) x^{(i)}_j$

- It will convert each data-point to the unit of standard deviation.
- Z represents the difference between raw score and population mean in the units of standard deviation.
- Z is negative when the raw score is below the mean and Z is positive when above mean.



Observation:

1. Move the mean-vector to the origin. Such that mean becomes zero (μ = 0)
2. Squishing/expanding such that standard deviation for any feature becomes one. (σ = 1)

Why Scale, Standardize, or Normalize?

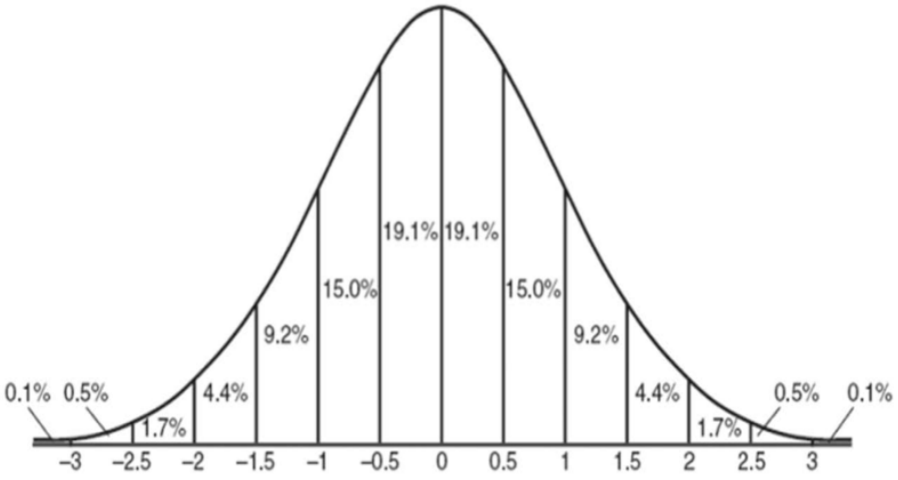
Many machine learning algorithms perform better or converge faster when features are on a relatively similar scale and/or close to normally distributed. Examples of such algorithm families include:

- linear and logistic regression
- nearest neighbors
- neural networks
- support vector machines with radial bias kernel functions
- principal components analysis
- linear discriminant analysis
- Scaling and standardizing can help features arrive in more digestible form for these algorithms.

When to use Normalize, or Standardize?

- **Normalization:** Data should be normalized if there is any predefined assumption of the model which you are going to use else there is no need. ANN doesn't have any assumption about the data.
- Normalization rescales the values into a range of [0,1]. This might be useful in some cases where all parameters need to have the same positive scale. However, the outliers from the data set are lost.
- It is useful to scale the input attributes for a model that relies on the magnitude of values, such as distance measures used in k-nearest neighbors and in the preparation of coefficients in regression. $X_{changed} = X - X_{min} / X_{max} - X_{min}$

4. The bell-shaped curve shown below illustrates a Standard Normal Distribution. Vertical lines have been drawn for every ½ standard deviation away from the mean. Illustrate the calculations you performed in question 3:
[a] draw a vertical line that divides "borderline high" and "high" cholesterol levels for this population,
[b] lightly shade the area representing "borderline high" cholesterol levels for this population, and
[c] darkly shade the area representing "high" cholesterol levels for this population.




```python
# Normalize the data attributes for the Iris dataset.
from sklearn.datasets import load_iris
from sklearn import preprocessing
# load the iris dataset
iris = load_iris()
print(iris.data.shape)
# separate the data from the target attributes
X = iris.data
y = iris.target
# normalize the data attributes
normalized_X = preprocessing.normalize(X)
```

- **Standardization:** When the data is uniformly/Normally distributed use Standardization.
- you need to standardized your data only when you are dealing with multiple variables together (i.e. Multivarite Anaysis). this is also valid for some other models not for ANN.
- you need to transform that nominal (categorical) variable into numeric one, normal method in this case is to use 0-1 binary values for all the k-th category. let say your nominal variable have seven categories(days) then you have to take 7 dummy variable where each of them will take value 0 or 1 depending on their presence or absence.
- It is useful to standardize attributes for a model that relies on the distribution of attributes such as Gaussian processes

```python
Standardize the data attributes for the Iris dataset
from sklearn.datasets import load_iris
from sklearn import preprocessing
# load the Iris dataset
iris = load_iris()
print(iris.data.shape)
# separate the data and target attributes
X = iris.data
y = iris.target
# standardize the data attributes
standardized_X = preprocessing.scale(X)
```

$X_{changed} = (x - \mu) / \sigma$

Ques 2. If the VIF is 2 then what is value of correlation coefficient (r^2)

Ans: **Variance inflation factor (VIF):** Variance inflation factor (VIF) is the quotient of the variance in a model with multiple terms by the variance of a model with one term alone.

It quantifies the severity of multicollinearity in an ordinary least squares regression analysis. It provides an index that measures how much the variance (the square of the estimate's standard deviation) of an estimated regression coefficient is increased because of collinearity.

The variance inflation factor (VIF) quantifies the extent of correlation between one predictor and the other predictors in a model. It is used for diagnosing collinearity/multicollinearity. Higher values signify that it is difficult to impossible to assess accurately the contribution of predictors to a model.

Computation of VIF:

The standard error of an estimate in a linear regression is determined by four things:

- The overall amount of noise (error). The more noise in the data, the higher the standard error.
- The variance of the associated predictor variable. The greater the variance of a predictor, the smaller the standard error (this is a scale effect).
- The sampling mechanism used to obtain the data. For example, the smaller the sample size with a simple random sample, the bigger the standard error.
- The extent to which a predictor is correlated with the other predictors in a model.

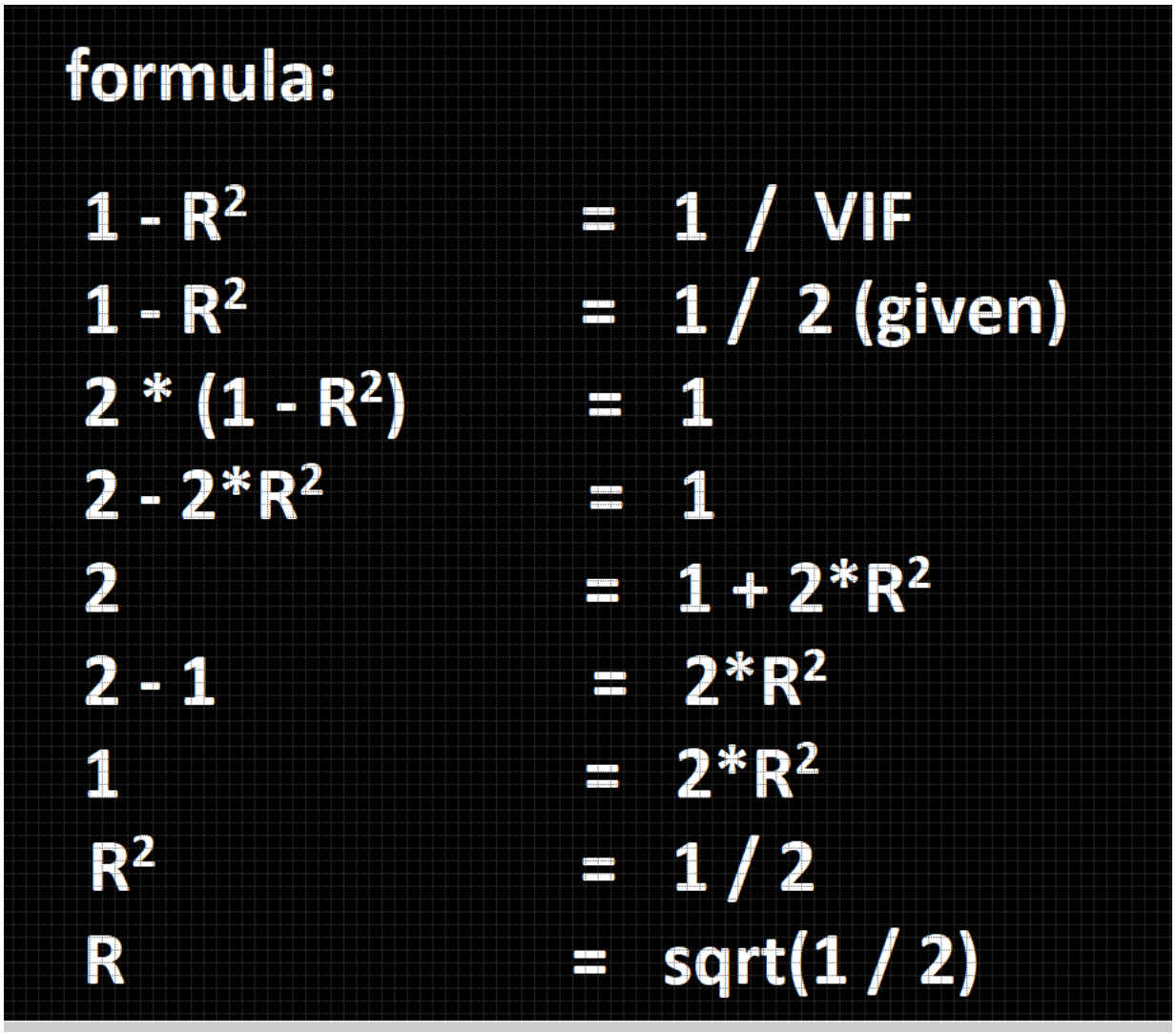
The extent to which a predictor is correlated with the other predictor variables in a linear regression can be quantified as the R-squared statistic of the regression where the predictor of interest is predicted by all the other predictor variables ( ). The variance inflation for a variable is then computed as:

$$VIF = \frac{1}{1 - R^2}$$

Some statistical software use tolerance instead of VIF, where tolerance is:

$$1 - R^2 = \frac{1}{VIF}.$$

The VIF can be applied to any type of predictive model (e.g., CART, or deep learning). A generalized version of the VIF, called the GVIF, exists for testing sets of predictor variables and generalized linear models.



If VIF = 2 then, R^2 is 0.5 or 1/2

A VIF can be computed for each predictor in a predictive model.

A value of 1 means that the predictor is not correlated with other variables. The higher the value, the greater the correlation of the variable with other variables. Values of more than 4 or 5 are sometimes regarded as being moderate to high, with values of 10 or more being regarded as very high.

These numbers are just rules of thumb; in some contexts a VIF of 2 could be a great problem (e.g., if estimating price elasticity), whereas in straightforward predictive applications very high VIFs may be unproblematic.

If one variable has a high VIF it means that other variables must also have high VIFs. In the simplest case, two variables will be highly correlated, and each will have the same high VIF.

Where a VIF is high, it makes it difficult to disentangle the relative importance of predictors in a model, particularly if the standard errors are regarded as being large. This is particularly problematic in two scenarios, where:

The focus of the model is on making inferences regarding the relative importance of the predictors. The model is to be used to make predictions in a different data set, in which the correlations may be different. The higher the VIF, the more the standard error is inflated, and the larger the confidence interval and the smaller the chance that a coefficient is determined to be statistically significant.

Variance inflation factors range from 1 upwards. The numerical value for VIF tells you (in decimal form) what percentage the variance (i.e. the standard error squared) is inflated for each coefficient.

For example, a VIF of 1.9 tells you that the variance of a particular coefficient is 90% bigger than what you would expect if there was no multicollinearity — if there was no correlation with other predictors. A rule of thumb for interpreting the variance inflation factor:

- 1 - not correlated.
- Between 1 and 5 - moderately correlated.
- Greater than 5 - highly correlated.

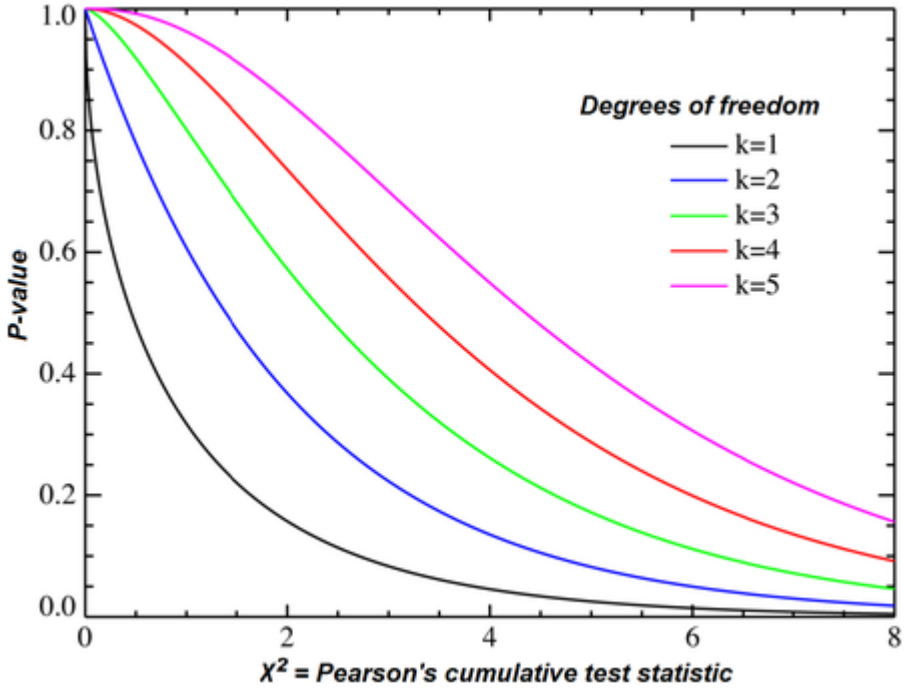
Remedies for VIF

Where VIF is regarded as being too high for variables, the solutions are to:

- Obtain more data, so as to reduce the standard errors.
- Use techniques designed to work better with high VIFs, such as Shapley regression (note that such techniques do not actually solve the VIF problem but instead ensure that the estimates are more reliable -- i.e., consistent).
- Obtain better data, where the predictors are less correlated (e.g., by conducting an experiment)
- Recode the predictors in a way that reduces correlations (e.g., using orthogonal polynomials instead of polynomials).

Ques 3. How do you interpret chi-square result?

Chi-squared (χ2) test: A chi-squared test, also written as χ2 test, is any statistical hypothesis test where the sampling distribution of the test statistic is a chi-squared distribution when the null hypothesis is true. Without other qualification, 'chi-squared test' often is used as short for Pearson's chi-squared test. The chi-squared test is used to determine whether there is a significant difference between the expected frequencies and the observed frequencies in one or more categories.



In the standard applications of this test, the observations are classified into mutually exclusive classes, and there is some theory, or say null hypothesis, which gives the probability that any observation falls into the corresponding class. The purpose of the test is to evaluate how likely the observations that are made would be, assuming the null hypothesis is true.

Chi-squared tests are often constructed from a sum of squared errors, or through the sample variance. Test statistics that follow a chi-squared distribution arise from an assumption of independent normally distributed data, which is valid in many cases due to the central limit theorem. A chi-squared test can be used to attempt rejection of the null hypothesis that the data are independent.

Also considered a chi-squared test is a test in which this is asymptotically true, meaning that the sampling distribution (if the null hypothesis is true) can be made to approximate a chi-squared distribution as closely as desired by making the sample size large enough.

chi-square test for independence is applied when you have two categorical variables from a single population. It is used to determine whether there is a significant association between the two variables.

For example, in an election survey, voters might be classified by gender (male or female) and voting preference (Democrat, Republican, or Independent). We could use a chi-square test for independence to determine whether gender is related to voting preference. The sample problem at the end of the lesson considers this example.

When to Use Chi-Square Test for Independence

The test procedure described in this lesson is appropriate when the following conditions are met:

- The sampling method is simple random sampling.
- The variables under study are each categorical.
- If sample data are displayed in a contingency table, the expected frequency count for each cell of the table is at least 5.

This approach consists of four steps: (1) state the hypotheses, (2) formulate an analysis plan, (3) analyze sample data, and (4) interpret results.

State the Hypotheses

Suppose that Variable A has r levels, and Variable B has c levels. The null hypothesis states that knowing the level of Variable A does not help you predict the level of Variable B. That is, the variables are independent.

Ho: Variable A and Variable B are independent.

Ha: Variable A and Variable B are not independent.

The alternative hypothesis is that knowing the level of Variable A can help you predict the level of Variable B.

Note: Support for the alternative hypothesis suggests that the variables are related; but the relationship is not necessarily causal, in the sense that one variable "causes" the other.

Formulate an Analysis Plan

- The analysis plan describes how to use sample data to accept or reject the null hypothesis. The plan should specify the following elements.
- Significance level. Often, researchers choose significance levels equal to 0.01, 0.05, or 0.10; but any value between 0 and 1 can be used.A Type I error occurs when the researcher rejects a null hypothesis when it is true. The probability of committing a Type I error is called the significance level, and is often denoted by α.

Test method. Use the chi-square test for independence to determine whether there is a significant relationship between two categorical variables.

Analyze Sample Data

Using sample data, find the degrees of freedom, expected frequencies, test statistic, and the P-value associated with the test statistic. The approach described in this section is illustrated in the sample problem at the end of this lesson.

Degrees of freedom. The degrees of freedom (DF) is equal to:

$$DF = (r - 1) * (c - 1)$$

where r is the number of levels for one catagorical variable, and c is the number of levels for the other categorical variable.

Expected frequencies. The expected frequency counts are computed separately for each level of one categorical variable at each level of the other categorical variable. Compute r * c expected frequencies, according to the following formula.

$$E_{r,c} = (n_r * n_c) / n$$

where Er,c is the expected frequency count for level r of Variable A and level c of Variable B, nr is the total number of sample observations at level r of Variable A, nc is the total number of sample observations at level c of Variable B, and n is the total sample size.

Test statistic. The test statistic is a chi-square random variable (X2) defined by the following equation.

$$X2 = \Sigma [(O_{r,c} - E_{r,c})2 / E_{r,c}]$$

where Or,c is the observed frequency count at level r of Variable A and level c of Variable B, and Er,c is the expected frequency count at level r of Variable A and level c of Variable B.

P-value. The P-value is the probability of observing a sample statistic as extreme as the test statistic. Since the test statistic is a chi-square, use the Chi-Square Distribution Calculator to assess the probability associated with the test statistic. Use the degrees of freedom computed above.

Interpret Results

If the sample findings are unlikely, given the null hypothesis, the researcher rejects the null hypothesis. Typically, this involves comparing the P-value to the significance level, and rejecting the null hypothesis when the P-value is less than the significance level.

Example :

Problem Statement:

A public opinion poll surveyed a simple random sample of 1000 voters. Respondents were classified by gender (male or female) and by voting preference (Republican, Democrat, or Independent). Results are shown in the contingency table below.

	Voting Preferences			Row total
	Rep	Dem	Ind	
Male	200	150	50	400
Female	250	300	50	600
Column total	450	450	100	1000

is there a gender gap? Do the men's voting preferences differ significantly from the women's preferences? Use a 0.05 level of significance.

Solution

The solution to this problem takes four steps: (1) state the hypotheses, (2) formulate an analysis plan, (3) analyze sample data, and (4) interpret results. We work through those steps below:

State the hypotheses. The first step is to state the null hypothesis and an alternative hypothesis.

Ho: Gender and voting preferences are independent.

Ha: Gender and voting preferences are not independent.

- **Formulate an analysis plan.** For this analysis, the significance level is 0.05. Using sample data, we will conduct a chi-square test for independence.
- **Analyze sample data.** Applying the chi-square test for independence to sample data, we compute the degrees of freedom, the expected frequency counts, and the chi-square test statistic. Based on the chi-square statistic and the degrees of freedom, we determine the P-value.

$$DF = (r - 1) * (c - 1) = (2 - 1) * (3 - 1) = 2$$

$$E_{r,c} = (n_r * n_c) / n$$

$$E_{1,1} = (400 * 450) / 1000 = 180000/1000 = 180$$

$$E_{1,2} = (400 * 450) / 1000 = 180000/1000 = 180$$

$$E_{1,3} = (400 * 100) / 1000 = 40000/1000 = 40$$

$$E_{2,1} = (600 * 450) / 1000 = 270000/1000 = 270$$

$$E_{2,2} = (600 * 450) / 1000 = 270000/1000 = 270$$

$$E_{2,3} = (600 * 100) / 1000 = 60000/1000 = 60$$

$$X^2 = \Sigma [(O_{r,c} - E_{r,c})^2 / E_{r,c}]$$

$$X^2 = (200 - 180)^2/180 + (150 - 180)^2/180 + (50 - 40)^2/40 + (250 - 270)^2/270 + (300 - 270)^2/270 + (50 - 60)^2/60$$

$$X^2 = 400/180 + 900/180 + 100/40 + 400/270 + 900/270 + 100/60$$

$$X^2 = 2.22 + 5.00 + 2.50 + 1.48 + 3.33 + 1.67 = 16.2$$

where DF is the degrees of freedom, r is the number of levels of gender, c is the number of levels of the voting preference, nr is the number of observations from level r of gender, nc is the number of observations from level c of voting preference, n is the number of observations in the sample, $E_{r,c}$ is the expected frequency count when gender is level r and voting preference is level c, and $O_{r,c}$ is the observed frequency count when gender is level r voting preference is level c.

> The P-value is the probability that a chi-square statistic having 2 degrees of freedom is more extreme than 16.2. We use the Chi-Square Distribution Calculator to find $P(X^2 > 16.2) = 0.0003$.

Interpret results. Since the P-value (0.0003) is less than the significance level (0.05), we cannot accept the null hypothesis. Thus, we conclude that there is a relationship between gender and voting preference.

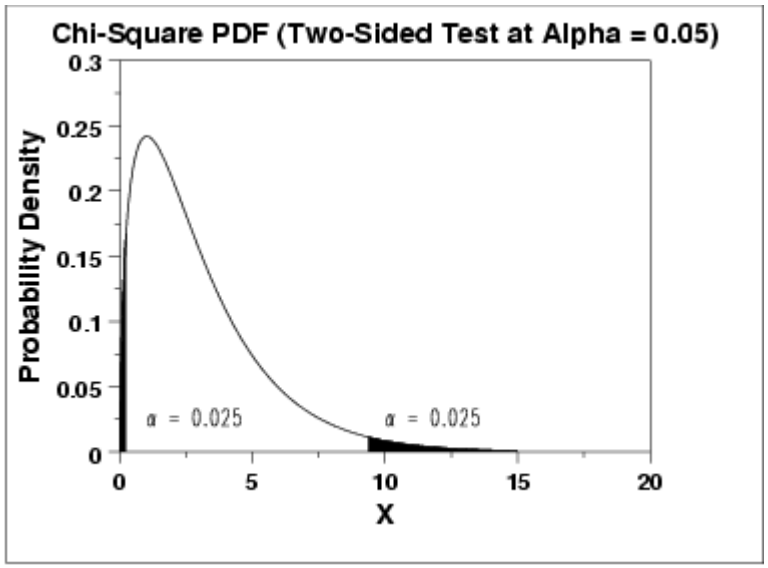
Note: If you use this approach on an exam, you may also want to mention why this approach is appropriate. Specifically, the approach is appropriate because the sampling method was simple random sampling, the variables under study were categorical, and the expected frequency count was at least 5 in each cell of the contingency table.

Critical Values of the Chi-Square Distribution

This table contains the critical values of the chi-square distribution. Because of the lack of symmetry of the chi-square distribution, separate tables are provided for the upper and lower tails of the distribution.

A test statistic with v degrees of freedom is computed from the data. For upper-tail one-sided tests, the test statistic is compared with a value from the table of upper-tail critical values. For two-sided tests, the test statistic is compared with values from both the table for the upper-tail critical values and the table for the lower-tail critical values.

The significance level, α, is demonstrated with the graph below which shows a chi-square distribution with 3 degrees of freedom for a two-sided test at significance level α = 0.05. If the test statistic is greater than the upper-tail critical value or less than the lower-tail critical value, we reject the null hypothesis. Specific instructions are given below.



Given a specified value of α:

1. For a two-sided test, find the column corresponding to 1-α/2 in the table for upper-tail critical values and reject the null hypothesis if the test statistic is greater than the tabled value. Similarly, find the column corresponding to α/2 in the table for lower-tail critical values and reject the null hypothesis if the test statistic is less than the tabled value.

2. For an upper-tail one-sided test, find the column corresponding to 1-α in the table containing upper-tail critical and reject the null hypothesis if the test statistic is greater than the tabled value.

3. For a lower-tail one-sided test, find the column corresponding to α in the lower-tail critical values table and reject the null hypothesis if the computed test statistic is less than the tabled value.

(https://www.itl.nist.gov/div898/handbook/eda/section3/eda3674.htm)

Click Here - Source (https://www.itl.nist.gov/div898/handbook/eda/section3/eda3674.htm)

Upper-tail critical values of chi-square distribution with ν degrees of freedom

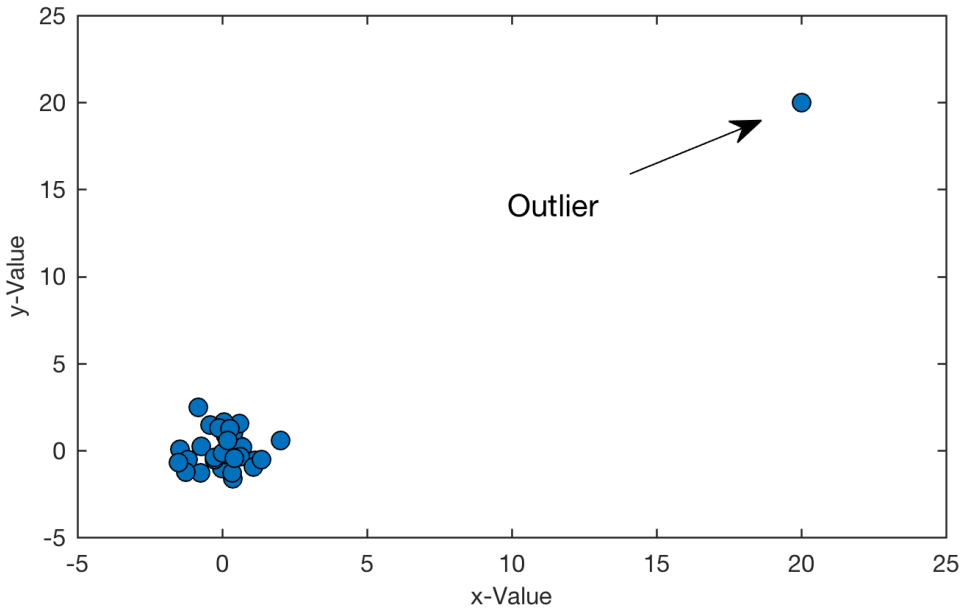
	Probability less than the critical value				
ν	0.90	0.95	0.975	0.99	0.999
1	2.706	3.841	5.024	6.635	10.828
2	4.605	5.991	7.378	9.210	13.816
3	6.251	7.815	9.348	11.345	16.266
4	7.779	9.488	11.143	13.277	18.467
5	9.236	11.070	12.833	15.086	20.515
6	10.645	12.592	14.449	16.812	22.458
7	12.017	14.067	16.013	18.475	24.322
8	13.362	15.507	17.535	20.090	26.125
9	14.684	16.919	19.023	21.666	27.877
10	15.987	18.307	20.483	23.209	29.588
11	17.275	19.675	21.920	24.725	31.264
12	18.549	21.026	23.337	26.217	32.910
13	19.812	22.362	24.736	27.688	34.528
14	21.064	23.685	26.119	29.141	36.123
15	22.307	24.996	27.488	30.578	37.697
16	23.542	26.296	28.845	32.000	39.252
17	24.769	27.587	30.191	33.409	40.790
18	25.989	28.869	31.526	34.805	42.312
19	27.204	30.144	32.852	36.191	43.820
20	28.412	31.410	34.170	37.566	45.315
21	29.615	32.671	35.479	38.932	46.797
22	30.813	33.924	36.781	40.289	48.268
23	32.007	35.172	38.076	41.638	49.728
24	33.196	36.415	39.364	42.980	51.179
25	34.382	37.652	40.646	44.314	52.620
26	35.563	38.885	41.923	45.642	54.052
27	36.741	40.113	43.195	46.963	55.476
28	37.916	41.337	44.461	48.278	56.892
29	39.087	42.557	45.722	49.588	58.301
30	40.256	43.773	46.979	50.892	59.703

Lower-tail critical values of chi-square distribution with ν degrees of freedom

	Probability less than the critical value				
ν	0.10	0.05	0.025	0.01	0.001
1.	.016	.004	.001	.000	.000
2.	.211	.103	.051	.020	.002
3.	.584	.352	.216	.115	.024
4.	1.064	.711	.484	.297	.091
5.	1.610	1.145	.831	.554	.210
6.	2.204	1.635	1.237	.872	.381
7.	2.833	2.167	1.690	1.239	.598
8.	3.490	2.733	2.180	1.646	.857
9.	4.168	3.325	2.700	2.088	1.152
10.	4.865	3.940	3.247	2.558	1.479
11.	5.578	4.575	3.816	3.053	1.834
12.	6.304	5.226	4.404	3.571	2.214
13.	7.042	5.892	5.009	4.107	2.617
14.	7.790	6.571	5.629	4.660	3.041
15.	8.547	7.261	6.262	5.229	3.483
16.	9.312	7.962	6.908	5.812	3.942
17.	10.085	8.672	7.564	6.408	4.416
18.	10.865	9.390	8.231	7.015	4.905
19.	11.651	10.117	8.907	7.633	5.407
20.	12.443	10.851	9.591	8.260	5.921
21.	13.240	11.591	10.283	8.897	6.447
22.	14.041	12.338	10.982	9.542	6.983
23.	14.848	13.091	11.689	10.196	7.529
24.	15.659	13.848	12.401	10.856	8.085
25.	16.473	14.611	13.120	11.524	8.649
26.	17.292	15.379	13.844	12.198	9.222
27.	18.114	16.151	14.573	12.879	9.803
28.	18.939	16.928	15.308	13.565	10.391
29.	19.768	17.708	16.047	14.256	10.986
30.	20.599	18.493	16.791	14.953	11.588

Ques 4. Why do we choose boxplot method than other for outlier detection and removal?

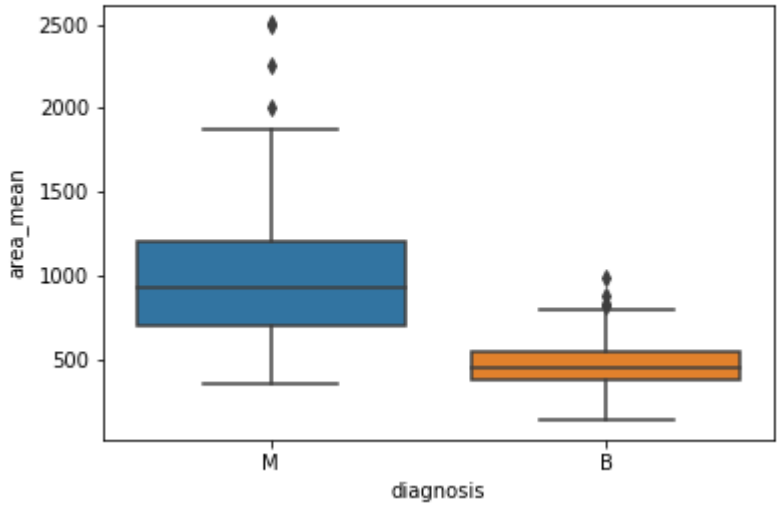
Outliers are observations inconsistent with rest of the dataset Global Outlier. These are due to poor data quality / contamination, Low quality measurement, manual error, malfunction of equipment.

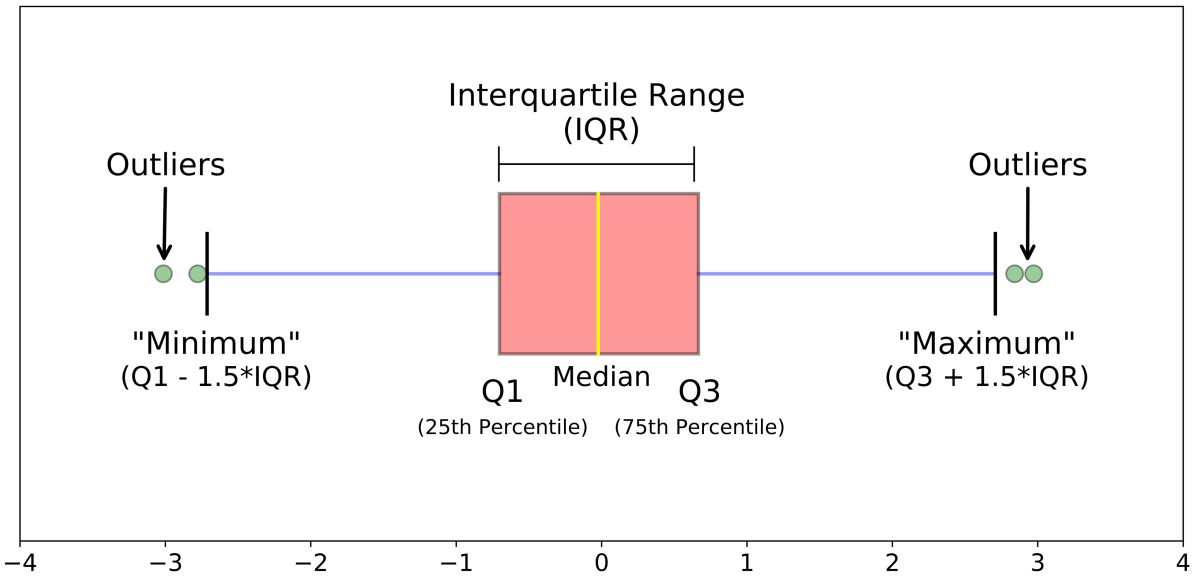


Causes of Outliers

- Poor data quality / contamination
- Low quality measurement, manual error
- Correct but exceptional data

Boxplot is a type of graph that displays a summary of a large amount of data in five numbers. These numbers include the median, upper quartile, lower quartile, minimum and maximum data values.





- Lowest line part is called lower fence.
- lower part of the box plot is called 25th percentile of data.
- middle part of box plot is called 50th percentile or median.
- upper part of the box plot is called 75th percentile of data.
- upper line part is called upper fence and beyond the datapoints either upper or lower fence are the outliers.

Handles Large Data Easily

Due to the five-number data summary, a box plot can handle and present a summary of a large amount of data. A box plot consists of the median, which is the midpoint of the range of data; the upper and lower quartiles, which represent the numbers above and below the highest and lower quarters of the data and the minimum and maximum data values. Organizing data in a box plot by using five key concepts is an efficient way of dealing with large data too unmanageable for other graphs, such as line plots or stem and leaf plots.

Exact Values Not Retained

The box plot does not keep the exact values and details of the distribution results, which is an issue with handling such large amounts of data in this graph type. A box plot shows only a simple summary of the distribution of results, so that it you can quickly view it and compare it with other data. Use a box plot in combination with another statistical graph method, like a histogram, for a more thorough, more detailed analysis of the data.

A Clear Summary

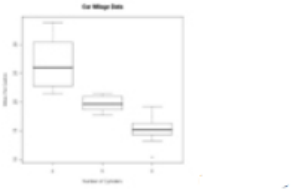
A box plot is a highly visually effective way of viewing a clear summary of one or more sets of data. It is particularly useful for quickly summarizing and comparing different sets of results from different experiments. At a glance, a box plot allows a graphical display of the distribution of results and provides indications of symmetry within the data.

Displays Outliers

A box plot is one of very few statistical graph methods that show outliers. There might be one outlier or multiple outliers within a set of data, which occurs both below and above the minimum and maximum data values. By extending the lesser and greater data values to a max of 1.5 times the inter-quartile range (IQR), the box plot delivers outliers or obscure results. Any results of data that fall outside of the minimum and maximum values known as outliers are easy to determine on a box plot graph

BOX - PLOT in R

```
# Boxplot of MPG by Car Cylinders
boxplot(mpg~cyl,data=mtcars, main="Car Milage Data",
   xlab="Number of Cylinders", ylab="Miles Per Gallon")
```



```
# Notched Boxplot of Tooth Growth Against 2 Crossed Factors
# boxes colored for ease of interpretation
boxplot(len~supp*dose, data=ToothGrowth, notch=TRUE,
  col=(c("gold","darkgreen")),
  main="Tooth Growth", xlab="Suppliment and Dose")
```

BOX - PLOT in Python

Syntax :

```
seaborn.boxplot(x=None, y=None, hue=None, data=None, order=None, hue_order=None,
orient=None, color=None, palette=None, saturation=0.75, width=0.8, dodge=True,
fliersize=5, linewidth=None, whis=1.5, notch=False, ax=None, **kwargs)
```

Parameters:

x = *feature of dataset*

y = *feature of dataset*

hue = *feature of dataset*

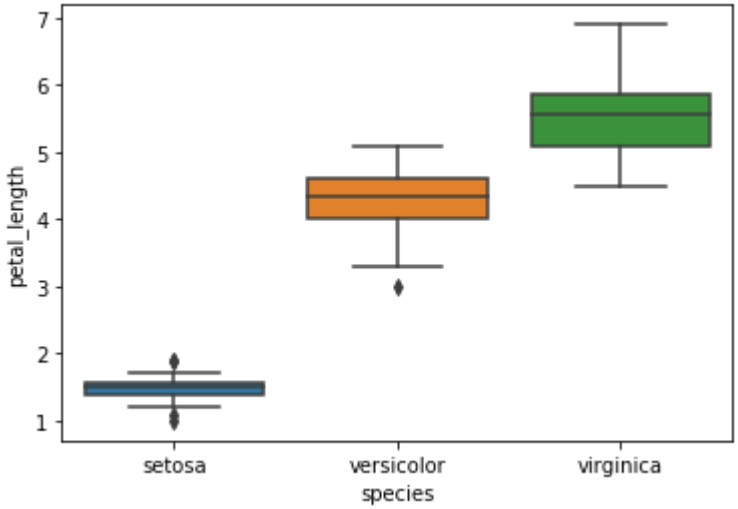
data = *datafram or full dataset*

color = *color name*

Example :

In [2]:

```
# Step 1: Importing and load the Dataset

import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

# Step 2: Load Dataset

iris = pd.read_csv('iris.csv')

# Step 3: Box - plot

sns.boxplot(x = 'species', y = 'petal_length', data=iris)

plt.show()
```



In [3]:
```
print("No. of data points : ",iris.shape[0], "\nNo. of features : ", iris.sh
print()
print(iris['species'].value_counts())
print("\nIt is a balanced dataset")
```

```
No. of data points :  150
No. of features :  5

virginica     50
setosa        50
versicolor    50
Name: species, dtype: int64

It is a balanced dataset
```

Observation:

1. Basically, we use iris dataset which have three class label iris-setosa, iris-versicolor, iris-virginica, we have 150 data-points and 5 columns.
2. It is a balanced dataset, each class-label has equal number of data-points.
3. We can find the 0th (lower wishker/fence) , 25th, 50th (median), 75th and 100th (upper wishker/fence percentile).
4. We can conclude that,

```
if petal-length < 2:
 print("iris-setosa")

elif petal_length > 2 and petal_length < 5:
print("iris-versicolor")

else:
print("iris-virginica")
```

5. We can some points which are beyond the whiskers those are Outliers.

Ques 5. How do we choose best method to impute missing value for a data?

Missing Value: Values or information which are not present/exists in the dataset. Null values are known as missing values, generally they are occurred in dataset due to:

1. Human Error : Missed to filling it up
2. Refuse to answer : Some privacy concern
3. Optional box : When values are not mandatory to filled

> Missing values are handled either dropping that row or imputation.

- Firstly, understand why the particular cell is having missing/null value.
- Plot bar graph
- Drop observation
- Consider the variables to impute whose missing values are less than 30%

In order to select the best method to impute the missing value is keep on trying and experimenting different method on the dataset and select that method which is nearer/closet to estimated result

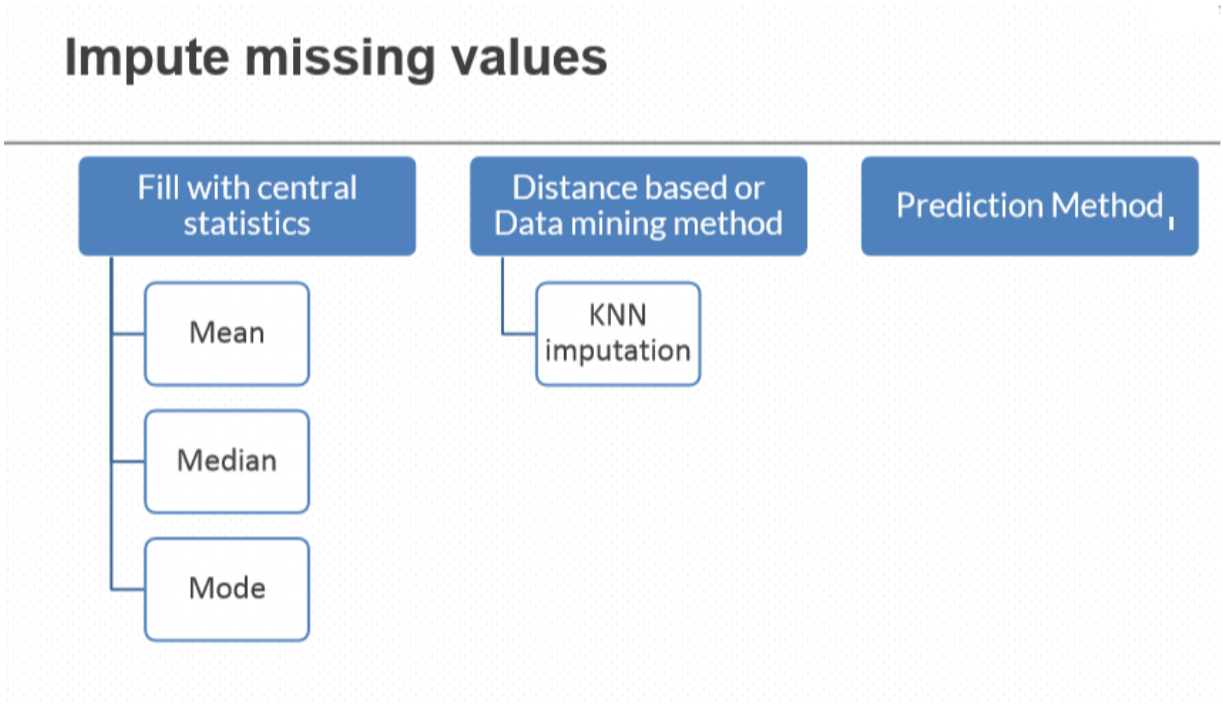
- Create a small subset of data with complete observation.
- Delete some values manually
- Use multiple methods to fill
- See where they are failing
- choose the best method

Though dropping rows and/or columns holding missing values is quite easy to do using numpy and pandas, it is often not appropriate.

Understanding why the data is missing is important before dropping these rows and columns. In this video you saw a number of situations in which dropping values was not a good idea. These included

Dropping data values associated with the effort or time an individual put into a survey. Dropping data values associated with sensitive information. In either of these cases, the missing values hold information. A quick removal of the rows or columns associated with these missing values would remove missing data that could be used to better inform models.

Instead of removing these values, we might keep track of the missing values using indicator values, or counts associated with how many questions an individual skipped.



Many real-world datasets may contain missing values for various reasons. They are often encoded as NaNs, blanks or any other placeholders. Training a model with a dataset that has a lot of missing values can drastically impact the machine learning model's quality. Some algorithms such as scikit-learn estimators assume that all values are numerical and have and hold meaningful value.

One way to handle this problem is to get rid of the observations that have missing data. However, you will risk losing data points with valuable information. A better strategy would be to impute the missing values. In other words, we need to infer those missing values from the existing part of the data. There are three main types of missing data:

- Missing completely at random (MCAR)
- Missing at random (MAR)
- Not missing at random (NMAR)

However, in this article, I will focus on 6 popular ways for data imputation for cross-sectional datasets (Time-series dataset is a different story).

1- Do Nothing:

That's an easy one. You just let the algorithm handle the missing data. Some algorithms can factor in the missing values and learn the best imputation values for the missing data based on the training loss reduction (ie. XGBoost). Some others have the option to just ignore them (ie. LightGBM—use_missing=false). However, other algorithms will panic and throw an error complaining about the missing values (ie. Scikit learn—LinearRegression). In that case, you will need to handle the missing data and clean it before feeding it to the algorithm.

Let's see some other ways to impute the missing values before training:

Note: All the examples below use the California Housing Dataset (https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_california_housing.html) from Scikit-learn.

2- Imputation Using (Mean/Median) Values:

This works by calculating the mean/median of the non-missing values in a column and then replacing the missing values within each column separately and independently from the others. It can only be used with numeric data.

Mean Imputation

Pros:

- Easy and fast.
- Works well with small numerical datasets.

Cons:

- Doesn't factor the correlations between features. It only works on the column level.
- Will give poor results on encoded categorical features (do NOT use it on categorical features).
- Not very accurate.
- Doesn't account for the uncertainty in the imputations.

Mean/Median Imputation

3- Imputation Using (Most Frequent) or (Zero/Constant) Values:

Most Frequent is another statistical strategy to impute missing values and YES!! It works with categorical features (strings or numerical representations) by replacing missing data with the most frequent values within each column.

Pros:

- Works well with categorical features.

Cons:

- It also doesn't factor the correlations between features.
- It can introduce bias in the data.

Most Frequent Imputation

Zero or Constant imputation—as the name suggests—it replaces the missing values with either zero or any constant value you specify

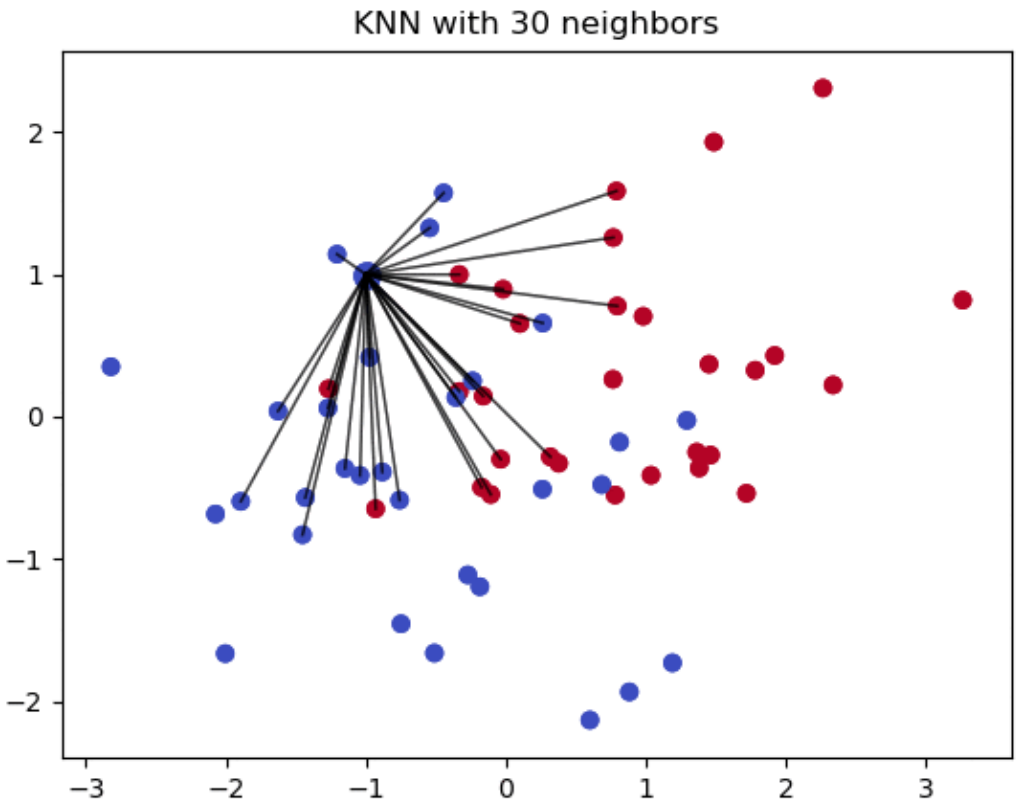
4- Imputation Using k-NN:

The k nearest neighbours is an algorithm that is used for simple classification. The algorithm uses **'feature similarity'** to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set. This can be very useful in making predictions about the missing values by finding the k's closest neighbours to the observation with missing data and then imputing them based on the non-missing values in the neighbourhood. Let's see some example code using Impyute library which provides a simple and easy way to use KNN for imputation:

KNN Imputation for California Housing Dataset

How does it work?

It creates a basic mean impute then uses the resulting complete list to construct a KDTree. Then, it uses the resulting KDTree to compute nearest neighbours (NN). After it finds the k-NNs, it takes the weighted average of them.



Pros:

- Can be much more accurate than the mean, median or most frequent imputation methods (It depends on the dataset).

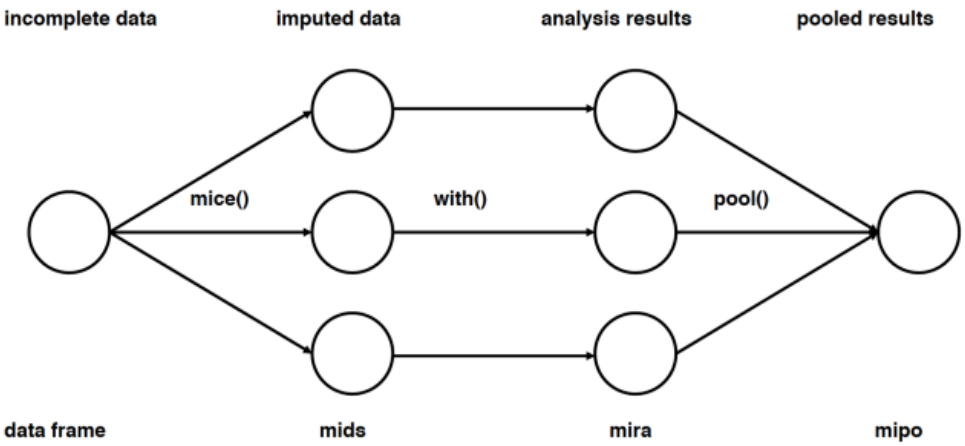
Cons:

- Computationally expensive. KNN works by storing the whole training dataset in memory.
- K-NN is quite sensitive to outliers in the data (unlike SVM)

5- Imputation Using Multivariate Imputation by Chained Equation (MICE)

Main steps used in multiple imputations

This type of imputation works by filling the missing data multiple times. Multiple Imputations (MIs) are much better than a single imputation as it measures the uncertainty of the missing values in a better way. The chained equations approach is also very flexible and can handle different variables of different data types (ie., continuous or binary) as well as complexities such as bounds or survey skip patterns. For more information on the algorithm mechanics, you can refer to the [Research Paper (https://www.jstatsoft.org/article/view/v045i03/v45i03.pdf)](https://www.jstatsoft.org/article/view/v045i03/v45i03.pdf)



MICE imputation using impyute

0	2	5.0	3.0	6.0	NaN
1	9	NaN	9.0	0.0	7.0
2	19	17.0	NaN	9.0	NaN
3	7	10.0	3.0	6.0	4.0
4	2	8.0	10.0	NaN	3.0

mice()

0	2.0	5.0	3.00	6.00	4.666667
1	9.0	10.0	9.00	0.00	7.000000
2	19.0	17.0	6.25	9.00	4.666667
3	7.0	10.0	3.00	6.00	4.000000
4	2.0	8.0	10.00	5.25	3.000000

6- Imputation Using Deep Learning (Datawig (https://github.com/awslabs/datawig)):

This method works very well with categorical and non-numerical features. It is a library that learns Machine Learning models using Deep Neural Networks to impute missing values in a dataframe. It also supports both CPU and GPU for training.

Imputation using Datawig

Pros:

- Quite accurate compared to other methods.
- It has some functions that can handle categorical data (Feature Encoder).
- supports CPUs and GPUs.

Cons:

- Single Column imputation.
- Can be quite slow with large datasets.
- You have to specify the columns that contain information about the target column that will be imputed.

Other Imputation Methods:

Stochastic regression imputation:

It is quite similar to regression imputation which tries to predict the missing values by regressing it from other related variables in the same dataset plus some random residual value.

Extrapolation and Interpolation:

It tries to estimate values from other observations within the range of a discrete set of known data points.

Hot-Deck imputation:

Works by randomly choosing the missing value from a set of related and similar variables.

In conclusion, there is no perfect way to compensate for the missing values in a dataset. Each strategy can perform better for certain datasets and missing data types but may perform much worse on other types of datasets. There are some set rules to decide which strategy to use for particular types of missing values, but beyond that, you should experiment and check which model works best for your dataset.

Example :

In [4]:
```
1 california_housing = pd.read_csv('california_housing.csv')
2
3 california_housing.isnull().sum()
```

Out[4]:
```
longitude               0
latitude                0
housing_median_age      0
total_rooms             0
total_bedrooms        207
population              0
households              0
median_income           0
median_house_value      0
ocean_proximity         0
dtype: int64
```

In [5]:
```
1 california_housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
longitude             20640 non-null float64
latitude              20640 non-null float64
housing_median_age    20640 non-null float64
total_rooms           20640 non-null float64
total_bedrooms        20433 non-null float64
population            20640 non-null float64
households            20640 non-null float64
median_income         20640 non-null float64
median_house_value    20640 non-null float64
ocean_proximity       20640 non-null object
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

In [6]:
```
1 missing_val = pd.DataFrame(california_housing.isnull().sum())
```

In [7]:
```
1 missing_val
```

Out[7]:

	0
longitude	0
latitude	0
housing_median_age	0
total_rooms	0
total_bedrooms	207
population	0
households	0
median_income	0
median_house_value	0
ocean_proximity	0

In [8]:
```
1 missing_val = missing_val.reset_index()
```

In [9]:
```
missing_val = missing_val.rename(columns = {'index':'Variables', 0:'Missing_percentage'})
```

In [10]:
```
missing_val['Missing_percentage'] = (missing_val['Missing_percentage'] / len(california_
```

In [11]:
```
missing_val = missing_val.rename(columns = {'index':'Variables', 0:'Missing_percentage'})
```

In [12]:
```
missing_val = missing_val.sort_values('Missing_percentage', ascending = Fals

Copy_df = missing_val.to_csv('missing_perc.csv', index = False)
```

In [13]:
```
california_housing['total_bedrooms'].loc[70]
```

Out[13]:
```
152.0
```

In [14]:
```
california_housing['total_bedrooms'].loc[70] = np.nan
```

```
C:\Users\ASTHA\Anaconda3\lib\site-packages\pandas\core\indexing.py:205: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stab
le/user_guide/indexing.html#returning-a-view-versus-a-copy (http://pandas.pydat
a.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cop
y)
  self._setitem_with_indexer(indexer, value)
```

In [15]:
```
california_housing['total_bedrooms'].loc[70]
```

Out[15]:
```
nan
```

In [16]:
```
# imputation method
# Actual Value = 152
# Mean         = 537.89
# Median       = 435
# KNN          = Similarly with KNN we cam impute values
```

In [17]:
```
# Impute with mean
california_housing['total_bedrooms'] = california_housing['total_bedrooms'].
```

In [18]:
```
california_housing['total_bedrooms'].loc[70]
```

Out[18]:
```
537.8894381362569
```

In [19]:
```
# Impute with median
california_housing['total_bedrooms'] = california_housing['total_bedrooms'].
#california_housing['total_bedrooms'].loc[70]
```

Conclusion :

To select / choose the best imputation technique / method, we have to keep on experimention on data using different different imputation techniques / methods. Those methods are mentioned

above, and select / choose that method which is efficient and close enough to out Actual / estimated value.