1. C) between -1 and 1
2. A) Lasso Regularisation
3. D) polynomial
4. A) Logistic Regression
5. D) Cannot be determined
6. B) increases
7. Random Forests reduce overfitting
8. B) Principal Components are calculated using unsupervised learning techniques

C) Principal Components are linear combinations of Linear Variables.

1. A) Identifying developed, developing and under-developed countries on the basis of factors like GDP, poverty index, employment rate, population and living index

B) Identifying loan defaulters in a bank on the basis of previous years’ data of loan accounts.

C) Identifying spam or ham emails

D) Identifying different segments of disease based on BMI, blood pressure, cholesterol, blood sugar

1. A) max\_depth

B) max\_features

D) min\_samples\_leaf

11.

**Outliers:**  
The outliers may suggest experimental errors, variability in a measurement, or an anomaly. The age of a person may wrongly be recorded as 200 rather than 20 Years. Such an outlier should definitely be discarded from the dataset.  
However, not all outliers are bad. Some outliers signify that data is significantly different from others. For example, it may indicate an anomaly like bank fraud or a rare disease.

**What is Interquartile Range IQR?**

IQR is used to **measure variability** by dividing a data set into quartiles. The data is sorted in ascending order and split into 4 equal parts. Q1, Q2, Q3 called first, second and third quartiles are the values which separate the 4 equal parts.

* Q1 represents the 25th percentile of the data.
* Q2 represents the 50th percentile of the data.
* Q3 represents the 75th percentile of the data.

If a dataset has *2n / 2n+1* data points, then  
Q1 = median of the dataset.  
Q2 = median of n smallest data points.  
Q3 = median of n highest data points.

IQR is the range between the first and the third quartiles namely Q1 and Q3: *IQR = Q3 – Q1*. The data points which fall below *Q1 – 1.5 IQR* or above *Q3 + 1.5 IQR* are outliers.

**Example:**  
Assume the data 6, 2, 1, 5, 4, 3, 50. If these values represent the number of chapatis eaten in lunch, then 50 is clearly an outlier.  
Step by step way to detect outlier in this dataset using **Python**:

**Step 1: Import necessary libraries.**

|  |
| --- |
| import numpy as np  import seaborn as sns |

**Step 2: Take the data and sort it in ascending order.**

|  |
| --- |
| data = [6, 2, 3, 4, 5, 1, 50]  sort\_data = np.sort(data)  sort\_data |

**Output:**

array([ 1, 2, 3, 4, 5, 6, 50])

**Step 3: Calculate Q1, Q2, Q3 and IQR.**

|  |
| --- |
| Q1 = np.percentile(data, 25, interpolation = 'midpoint')  Q2 = np.percentile(data, 50, interpolation = 'midpoint')  Q3 = np.percentile(data, 75, interpolation = 'midpoint')    print('Q1 25 percentile of the given data is, ', Q1)  print('Q1 50 percentile of the given data is, ', Q2)  print('Q1 75 percentile of the given data is, ', Q3)    IQR = Q3 - Q1  print('Interquartile range is', IQR) |

**Output:**

Q1 25 percentile of the given data is, 2.5

Q1 50 percentile of the given data is, 4.0

Q1 75 percentile of the given data is, 5.5

Interquartile range is 3.0

**Step 4: Find the lower and upper limits as Q1 – 1.5 IQR and Q3 + 1.5 IQR, respectively.**

|  |
| --- |
| low\_lim = Q1 - 1.5 \* IQR  up\_lim = Q3 + 1.5 \* IQR  print('low\_limit is', low\_lim)  print('up\_limit is', up\_lim) |

**Output:**

low\_limit is -2.0

up\_limit is 10.0

**Step 5: Data points greater than the upper limit or less than the lower limit are outliers**

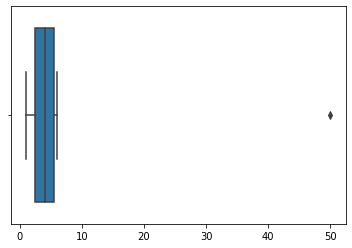
|  |
| --- |
| outlier =[]  for x in data:      if ((x> up\_lim) or (x<low\_lim)):           outlier.append(x)  print(' outlier in the dataset is', outlier) |

**Output:**

outlier in the dataset is [50]

**Step 6: Plot the box plot to highlight outliers.**

|  |
| --- |
| sns.boxplot(data) |



**Step 7: Following code can also be used to calculate IQR**

|  |
| --- |
| from scipy import stats  IQR = stats.iqr(data, interpolation = 'midpoint')  IQR |

**Output:**

3.0

1. **Differences Between Bagging and Boosting**

|  |  |  |
| --- | --- | --- |
| SR | Bagging | Boosting |
| 1 | The simplest way of combining predictions that  belong to the same type. | A way of combining predictions that  belong to the different types. |
| 2 | Aim to decrease variance, not bias | Aim to decrease bias, not variance. |
| 3 | Each model receives equal weight. | Models are weighted according to their performance. |
| 4 | Each model is built independently. | New models are influenced  by the performance of previously built models. |
| 5 | Different training data subsets are selected using row sampling with replacement and random sampling methods from the entire training dataset. | Every new subset contains the elements that were misclassified by previous models. |
| 6 | Bagging tries to solve the over-fitting problem. | Boosting tries to reduce bias. |
| 7 | If the classifier is unstable (high variance), then apply bagging. | If the classifier is stable and simple (high bias) the apply boosting. |
| 8 | In this base classifiers are trained parallelly. | In this base classifiers are trained sequentially. |
| 9 | Example: The Random forest model uses Bagging. | Example: The AdaBoost uses Boosting techniques |

1. **Adjusted R2 :**

Adjusted R2 is **a corrected goodness-of-fit (model accuracy) measure for linear models**. It identifies the percentage of variance in the target field that is explained by the input or inputs. R2 tends to optimistically estimate the fit of the linear regression.

**R-squared**, often written R2, is the proportion of the variance in the [response variable](https://www.statology.org/explanatory-response-variables/) that can be explained by the predictor variables in a [linear regression model](https://www.statology.org/multiple-linear-regression/).

The value for R-squared can range from 0 to 1. A value of 0 indicates that the response variable cannot be explained by the predictor variable at all while a value of 1 indicates that the response variable can be perfectly explained without error by the predictor variables.

The **adjusted R-squared** is a modified version of R-squared that adjusts for the number of predictors in a regression model. It is calculated as:

**Adjusted R2 = 1 – [(1-R2)\*(n-1)/(n-k-1)]**

where:

* **R2**: The R2 of the model
* **n**: The number of observations
* **k**: The number of predictor variables

Since R2 always increases as you add more predictors to a model, adjusted R2 can serve as a metric that tells you how useful a model is, adjusted for the number of predictors in a model.

## Normalisation vs. Standardisation

Normalisation vs. standardisation is an ultimate doubt among machine learning beginners.

Normalisation is suitable to use when the data does not follow Gaussian Distribution principles. It can be used in algorithms that do not assume data distribution, such as K-Nearest Neighbors and Neural Networks.

On the other hand, standardisation is beneficial in cases where the dataset follows the Gaussian distribution. Unlike Normalization, Standardisation is not affected by the outliers in the dataset as it does not have any bounding range.

Applying Normalization or Standardisation depends on the problem and the machine learning algorithm. There are no definite rules as to when to use Normalization or Standardisation. One can fit the normalized or standardized dataset into the model and compare the two.

It is always advisable to first fit the scaler on the training data and then transform the testing data. This would prohibit data leakage during the model testing process, and the scaling of target values is generally not required.

|  |  |
| --- | --- |
| **Normalisation** | **Standardisation** |
| Scaling is done by the highest and the lowest values. | Scaling is done by mean and standard deviation. |
| It is applied when the features are of separate scales. | It is applied when we verify zero mean and unit standard deviation. |
| Scales range from 0 to 1 | Not bounded |
| Affected by outliers | Less affected by outliers |
| It is applied when we are not sure about the data distribution | It is used when the data is Gaussian or normally distributed |
| It is also known as Scaling Normalization | It is also known as Z-Score |

1. Cross Validation:

Cross-validation is a [statistical method](https://www.mygreatlearning.com/academy/learn-for-free/courses/statistical-methods-for-decision-making/?gl_blog_id=17418) used to estimate the performance (or accuracy) of machine learning models. It is used to protect against overfitting in a predictive model, particularly in a case where the amount of data may be limited. In cross-validation, you make a fixed number of folds (or partitions) of the data, run the analysis on each fold, and then average the overall error estimate.

When dealing with a [Machine Learning](https://www.mygreatlearning.com/academy/learn-for-free/courses/basics-of-machine-learning-1/?gl_blog_id=17418) task, you have to properly identify the problem so that you can pick the most suitable algorithm which can give you the best score. But how do we compare the models?

Say, you have trained the model with the dataset available and now you want to know how well the model can perform. One approach can be that you are going to test the model on the dataset you have trained it on, but this may not be a good practice.

So what is wrong with testing the model on the training dataset? If we do so, we assume that the training data represents all the possible scenarios of real-world and this will surely never be the case. Our main objective is that the model should be able to work well on the real-world data, although the training dataset is also real-world data, it represents a small set of all the possible data points(examples) out there.

So to know the real score of the model, it should be tested on the data that it has never seen before and this set of data is usually called testing set. But if we split our data into training data and testing data, aren’t we going to lose some important information that the test [dataset](https://www.mygreatlearning.com/blog/dataset-in-machine-learning/)may hold

**Advantages of Cross Validation**  
  
**1. Reduces Overfitting:** In Cross Validation, we split the dataset into multiple folds and train the algorithm on different folds. This prevents our model from overfitting the training dataset. So, in this way, the model attains the generalization capabilities which is a good sign of a robust algorithm.  
  
**Note:**Chances of overfitting are less if the dataset is large. So, Cross Validation may not be required at all in the situation where we have sufficient data available.  
  
**2. Hyperparameter Tuning:** Cross Validation helps in finding the optimal value of hyperparameters to increase the efficiency of the algorithm.

**Disadvantages of Cross Validation**  
  
**1. Increases Training Time:** Cross Validation drastically increases the training time. Earlier you had to train your model only on one training set, but with Cross Validation you have to train your model on multiple training sets.   
  
For example, if you go with 5 Fold Cross Validation, you need to do 5 rounds of training each on different 4/5 of available data. And this is for only one choice of hyperparameters. If you have multiple choice of parameters, then the training period will shoot too high.  
  
**2. Needs Expensive Computation:** Cross Validation is computationally very expensive in terms of processing power required.