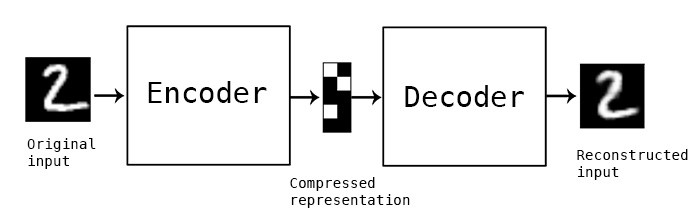
<https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/?utm_source=blog&utm_medium=auc-roc-curve-machine-learning>

https://sagemaker-workshop.com/introduction/concepts.html

[**https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726**](https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726)

**Autoencoder**, by design, reduces data dimensions by learning how to ignore the noise in the data.

Autoencoder is an unsupervised artificial neural network that learns how to efficiently compress and encode data then learns how to reconstruct the data back **from** the reduced encoded representation **to** a representation that is as close to the original input as possible.



## **Autoencoder Components:**

Autoencoders consists of 4 main parts:

1- **Encoder**: In which the model learns how to reduce the input dimensions and compress the input data into an encoded representation.

2- **Bottleneck**: which is the layer that contains the compressed representation of the input data. This is the lowest possible dimensions of the input data.

3- **Decoder**: In which the model learns how to reconstruct the data from the encoded representation to be as close to the original input as possible.

4- **Reconstruction** **Loss**: This is the method that measures measure how well the decoder is performing and how close the output is to the original input.

The training then involves using back propagation in order to minimize the network’s reconstruction loss.

Autoencoder Architecture: FeedForward network, LSTM network or Convolutional Neural Network

# 1- Autoencoder for Anomaly Detection: There are many ways and techniques to detect anomalies and outliers

2- Image Denoising:

<https://towardsdatascience.com/tf-term-frequency-idf-inverse-document-frequency-from-scratch-in-python-6c2b61b78558>

Table of Contents:

Terminology .

* Term Frequency(TF) .
* Document Frequency .
* Inverse Document Frequency .

Implementation in Python .

1 - Terminology :

t — term (word)

d — document (set of words)

N — count of corpus

corpus — the total document set

the number of times a term occurs in a document is called its term frequency

tf(t,d) = count of t in d / number of words in d

df(t) = occurrence of t in documents

Idf(t) = log(N/(df + 1))

tf-idf(t, d) = tf(t, d) \* log(N/(df + 1))

Feature engineering Techniques:-

1.Imputation

2.Handling Outliers

3.Binning

4.Log Transform

5.One-Hot Encoding

6.Grouping Operations

7.Feature Split

8.Scaling

9.Extracting Date

The most simple solution to the missing values is to drop the rows or the entire column. There is not an optimum threshold for dropping but you can use **70%** as an example value and try to drop the rows and columns which have missing values with higher than this threshold.

threshold = 0.7

**#Dropping columns with missing value rate higher than threshold**  
 data = data[data.columns[data.isnull().mean() < threshold]]  
  
**#Dropping rows with missing value rate higher than threshold**  
 data = data.loc[data.isnull().mean(axis=1) < threshold]

Numerical Imputation

Categorical Imputation

https://elitedatascience.com/machine-learning-interview-questions-answers

LDA :- In natural language processing, the Latent Dirichlet Allocation is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar.

**Data Preprocessing**

Box-Cox transformation is a generalized "power transformation" that transforms data to make the distribution more normal

It's used to stabilize the variance (eliminate heteroskedasticity) and normalize the distribution.

3 data preprocessing techniques to handle outliers :-

Winsorize (cap at threshold).

Transform to reduce skew (using Box-Cox or similar).

Remove outliers if you're certain they are anomalies or measurem

3 ways of reducing dimensionality :-

Removing collinear features.

Performing PCA, ICA, or other forms of algorithmic dimensionality reduction.

Combining features with feature engineering

**Decision trees:-**

Advantages: Decision trees are easy to interpret, nonparametric (which means they are robust to outliers), and there are relatively few parameters to tune.

Disadvantages: Decision trees are prone to be overfit. However, this can be addressed by ensemble methods like random forests or boosted trees.

**Neural networks:-**

Advantages: Neural networks (specifically deep NNs) have led to performance breakthroughs for unstructured datasets such as images, audio, and video. Their incredible flexibility allows them to learn patterns that no other ML algorithm can learn.

Disadvantages: However, they require a large amount of training data to converge. It's also difficult to pick the right architecture, and the internal "hidden" layers are incomprehensible.

**ROC** (receiver operating characteristic) the performance plot for binary classifiers of True Positive Rate (y-axis) vs. False Positive Rate (x-

axis)

It's equivalent to the expected probability that a uniformly drawn random positive is ranked before a uniformly drawn random negative.

**AUROC** :- robust to class imbalance, unlike raw accuracy.

For example, if you want to detect a type of cancer that's prevalent in only 1% of the population, you can build a model that achieves 99% accuracy by simply classifying everyone has cancer-free.

raw accuracy as an out-of- sample evaluation metric

**Bias** is the simplifying assumptions made by the model to make the target function easier to approximate.

**Variance –** model performance is different on different dataset.

Trade-off is tension between the error introduced by the bias and the variance.

Linear machine learning algorithms often have a high bias but a low variance. Nonlinear machine learning algorithms often have a low bias but a high variance

**Bias Error**

Bias are the simplifying assumptions made by a model to make the target function easier to learn.

- Have lower predictive performance on complex problems

Low Bias: Suggests less assumptions about the form of the target function.

High-Bias: Suggests more assumptions about the form of the target function.

**Variance Error**

Variance is the amount that the estimate of the target function will change if different training data was used.

low-variance / high-bias(Linear) - Linear Regression, Linear Discriminant Analysis and Logistic Regression.

high-variance/ low-bias (non linear) - Decision Trees, k-Nearest Neighbors and Support Vector Machines.

The k-nearest neighbors algorithm has low bias and high variance, but the trade-off can be changed by increasing the value of k which increases the number of neighbors that contribute t the prediction and in turn increases the bias of the model.

The support vector machine algorithm has low bias and high variance, but the trade-off can be changed by increasing the C parameter that influences the number of violations of the margin allowed in the training data which increases the bias but decreases the variance.

**Weights and Bias with Cricket eg :-**

**Factors** like the weather or temperature might have a higher weight, and other factors like equipment would have a lower weight.

**Bias** lets you assign some threshold which helps you activate a decision-point (or a neuron) only when that threshold is crossed.

* need 1 ball and 6 wickets

**Forward propagation:**  calculate the output of the activation at each node .

**Backpropagation :** minimize the cost function by its understanding of how it changes with changing the weights and biases in a neural network.

Change obtained by gradient descent.

**Batch Normalization** is one of the techniques used for reducing the training time of our deep learning algorithm. Just like normalizing our input helps improve our logistic regression model, we can normalize the activations of the hidden layers in our deep learning model as well

### CNN :- it just identifies curves and edges. Thus, instead of looking at the entire image, it helps to just read the image in parts.

### Element wise operations:-

### CNN_Deep_learning_questions

### Artificial Neural Network (ANN) :- each input is given the same weight and fed to the network at the same time

### I saw the movie and hated it

### Difficult to identify how it is attached to movie to RNN is comes into picture.

### RNN :- The addition of a loop is to denote preserving the previous node’s information for the next node, better for sequential data.

### Valid Padding: When we do not use any padding. The resultant matrix after convolution will have dimensions (n – f + 1) X (n – f + 1)

### Same padding: Adding padded elements all around the edges such that the output matrix will have the same dimensions as that of the input matrix.

### Gradient Descent algorithm tries to minimize the error by taking small steps towards the minimum value. These steps are used to update the weights and biases in a neural network.

### exploding gradient :- the steps become too large and this results in larger updates to weights and bias terms

### vanishing gradient : steps are too small and this leads to minimal changes in the weights and bias terms – even negligible changes at times.

### LSTM – Input gate, Output gate, forget gate

### GRU :- Update Gate, Ouput gate

### GRU less complex and faster than LSTM

### Transformers use what is called the attention mechanism. This basically means mapping dependencies between all the parts of a sentence

<https://www.edureka.co/blog/interview-questions/deep-learning-interview-questions/>

### Ridge Regression:

* **Linear regression** in which a small amount of bias is introduced
* **lambda \* the squared weight** of each individual features
* Linear or polynomial regression will **fail** if there is high **collinearity** between the independent variables

Regression Analysis in Machine learning

### Lasso Regression

* Technique to reduce the complexity of the model
* Penalty term contains only the **absolute weights instead** of a square of weights

Regression Analysis in Machine learning

### Decision Tree Regression:

* A **supervised** learning algorithm
* Decision Tree regression builds a tree-like structure
* Random forest is one of the most **powerful** supervised learning algorithms
* The Random Forest regression is an ensemble learning
* **Ensemble learning** - combines multiple decision trees and predicts the final output based on the average of each tree output

Random forest uses **Bagging and Boosting**

**Bagging – Mostly used**

* **Random set**
* **Parrallel training**
* **Aggragate all tree**

### Support Vector Regression

* **Kernel:**  function used to map a lower-dimensional data into higher dimensional data.
* **Hyperplane:** **SVM**, it is a **separation** line between two classes, but in **SVR**, it is a line which helps to **predict** the continuous.
* **Boundary line:**  the two lines apart from hyperplane, which creates a margin for datapoints.
* **Support vectors:**  the datapoints which are nearest to the hyperplane and opposite class.

**Linear Regression in Machine Learning:-**

* It is used for **predictive analysis**
* **Goal is to find the best fit line**
* **Error** between predicted values and actual values should be **minimized**
* For Linear Regression, we use the **Mean Squared Error (MSE)**

**Residuals:** The distance between the actual value and predicted values is called residual.

Linear Regression in Machine Learning

**Residuals:** The distance between the actual value and predicted values is called residual.

**Gradient Descent:**

* Used to minimize the MSE by calculating the gradient of the cost function.
* A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
* It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

Model Performance:

The Goodness of fit determines how the line of regression fits the set of observations

**1. R-squared method:**

* R-squared is a statistical method that determines the goodness of fit.Linear Regression in Machine Learning

## **Evaluating a Classification model:**

**1. Log Loss or Cross-Entropy Loss:**

* + value between the 0 and 1.
  + lower log loss -> higher accuracy of the mode

**2. Confusion Matrix:**

**3. AUC-ROC curve**

### Feature Selection :-

* **Filtering Methods**
* **Wrapper Methods**
* **Embedded Methods**

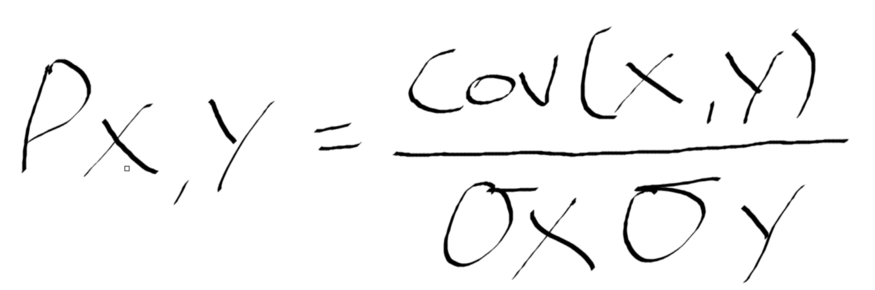
https://www.kdnuggets.com/2020/08/getting-started-feature-selection.html

**Filtering Methods**

Statistically significant relationship from each input feature(x) to the target feature(y)

An example of a filtering method is Pearson's correlation coefficient

It ranges from +1 to -1



 Cov is the covariance,

σX is the standard deviation of X,

σY is the standard deviation of Y.

1. Chi-Squared Test
2. Information Gain
3. F Test
4. ANOVA
5. Correlation Coefficient Scores (eg. Pearson’s Correlation Coefficient)

**2. Wrappers Methods**

some features are fed to the ML model, and evaluate the performance.

The performance decides whether to add those features or remove to increase the accuracy of the model.

* Forward Selection
* Backward Selection
* Bi-directional Elimination

**3. Embedded Methods:** Embedded methods check the different training iterations of the machine learning model and evaluate the importance of each feature. Some common techniques of Embedded methods are:

* **LASSO**
* **Elastic Net**
* **Ridge Regression, etc.**

**Techniques handle imbalanced data :**

<https://www.kdnuggets.com/2017/06/7-techniques-handle-imbalanced-data.html>

<https://www.kdnuggets.com/2021/07/distinguish-yourself-hundreds-other-data-science-candidates.html>

https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/

### 1. Use the right evaluation metrics

* **Precision/Specificity**: how many selected instances are relevant.
* **Recall**/Sensitivity: how many relevant instances are selected.
* **F1 score:** harmonic mean of precision and recall.
* **MCC**: correlation coefficient between the observed and predicted binary classifications.
* **AUC**: relation between true-positive rate and false positive rate.

### 2. Resample the training set

**2.1. Under-sampling :**

* **R**educing the size of the abundant class.
* used when quantity of data is sufficient.

**2.2. Over-sampling**

* It tries to balance dataset by increasing the size of rare samples.
* New rare samples are generated by using e.g. **repetition, bootstrapping or SMOTE (Synthetic Minority Over-Sampling Technique)**

### 3. Use K-fold Cross-Validation in the right way

* Used over-sampling method to address imbalance problems.
* It is applied after **over-sampling**
* oversampling can lead to **overfitting**
* cross-validation should always be **done before** over-sampling the data.
* Only by resampling the data repeatedly, **randomness** can be introduced into the dataset to make sure that there won’t be an overfitting problem.

### 4. Ensemble different resampled datasets

### Happen in logistic regression or random forest tend to generalize by discarding the rare class

### Best to use all the samples of the rare class and n-differing samples of the abundant class

### you just split the 10.000 cases with 1 rare case in 10 chunks and train 10 different models.

### 7. Design your own models

### The famous XGBoost is already a good starting point if the classes are not skewed too much, because it internally takes care that the bags it trains on are not imbalanced. But then again, the data is resampled, it is just happening secretly.

### By designing a cost function that is penalizing wrong classification of the rare class more than wrong classifications of the abundant class, it is possible to design many models that naturally generalize in favour of the rare class. For example, tweaking an SVM to penalize wrong classifications of the rare class by the same ratio that this class is underrepresented.

### A discriminative model makes predictions based on conditional probability and is either used for classification or regression.

### A generative model revolves around the distribution of a dataset to return a probability for a given example.

### https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

Loss Functions is divided into three parts; they are:

1. Regression Loss Functions
   1. Mean Squared Error Loss
   2. Mean Squared Logarithmic Error Loss
   3. Mean Absolute Error Loss
2. Binary Classification Loss Functions
   1. Binary Cross-Entropy
   2. Hinge Loss
   3. Squared Hinge Loss
3. Multi-Class Classification Loss Functions
   1. Multi-Class Cross-Entropy Loss
   2. Sparse Multiclass Cross-Entropy Loss
   3. Kullback Leibler Divergence Loss

Feature engineering Techniques:-

1.Imputation

2.Handling Outliers

3.Binning

4.Log Transform

5.One-Hot Encoding

6.Grouping Operations

7.Feature Split

8.Scaling

9.Extracting Date

**https://towardsdatascience.com/feature-engineering-for-machine-learning-3a5e293a5114**

**1.Imputation:-**

The most simple solution to the missing values is to drop the rows or the entire column. There is not an optimum threshold for dropping but you can use **70%** as an example value and try to drop the rows and columns which have missing values with higher than this threshold.

threshold = 0.7

#Dropping columns with missing value rate higher than threshold

data = data[data.columns[data.isnull().mean() < threshold]]

#Dropping rows with missing value rate higher than threshold

data = data.loc[data.isnull().mean(axis=1) < threshold]

**Numerical Imputation**

The best imputation way is to use the medians

#Filling all missing values with 0

data = data.fillna(0)

#Filling missing values with medians of the columns

data = data.fillna(data.median())

**Categorical Imputation**

#Max fill function for categorical columns

data['column\_name'].fillna(data['column\_name'].value\_counts()

.idxmax(), inplace=True)

# **2.Handling Outliers**

the best way to detect the outliers is to demonstrate the data visually. Two different ways of handling outliers are **standard deviation**, and **percentiles.**

If a value has a distance to the average higher than **x \* standard deviation,**it can be assumed as an outlier : 2 and 4

#Dropping the outlier rows with standard deviation

factor = 3

upper\_lim = data['column'].mean () + data['column'].std () \* factor

lower\_lim = data['column'].mean () - data['column'].std () \* factor

data = data[(data['column'] < upper\_lim) & (data['column'] > lower\_lim)]

**Z-score** (or standard score) standardizes the distance between a value and the mean using the standard deviation.

## **Outlier Detection with Percentiles**

We can assume a certain percent of the value from the top or the bottom as an outlie.

**#Dropping the outlier rows with Percentiles**  
upper\_lim = data['column'].quantile(.95)  
lower\_lim = data['column'].quantile(.05)  
  
data = data[(data['column'] < upper\_lim) & (data['column'] > lower\_lim)]

# **3. Log Transform**

# Logarithm transformation (or log transform) is one of the most commonly used mathematical transformations in feature engineering

* It helps to handle skewed data and after transformation, the distribution becomes more approximate to normal.
* In most of the cases the magnitude order of the data changes within the range of the data

A critical note: The data you apply log transform must have only positive values, otherwise you receive an error. Also, you can add 1 to your data before transform it. Thus, you ensure the output of the transformation to be positive.

**4. One-hot encoding :- Categorical Column Grouping**

**5. Feature Split**

**6. Scaling**

Algorithms based on **distance** calculations such as **k-NN**or **k-Means** need to have scaled continuous features as model input.

Basically, there are two common ways of scaling:

**Normalization**



all values in a fixed range between **0** and **1**

Normalization (or **min-max normalization**) scale all values in a fixed range between **0** and **1**. This transformation does not change the distribution of the feature and due to the decreased standard deviations, the effects of the **outliers** increases. Therefore, before normalization, it is recommended to handle the outliers

data = pd.DataFrame({'value':[2,45, -23, 85, 28, 2, 35, -12]})  
  
data['normalized'] = (data['value'] - data['value'].min()) / (data['value'].max() - data['value'].min()) **value normalized**  
0 2 0.23  
1 45 0.63  
2 -23 0.00  
3 85 1.00  
4 28 0.47  
5 2 0.23  
6 35 0.54  
7 -12 0.10

## **Standardization**

Standardization (or **z-score normalization**) scales the values while taking into account standard deviation. If the standard deviation of features is different, their range also would differ from each other. This reduces the effect of the outliers in the features.

In the following formula of standardization, the **mean** is shown as **μ** and the **standard** **deviation** is shown as **σ**.



data = pd.DataFrame({'value':[2,45, -23, 85, 28, 2, 35, -12]})  
  
data['standardized'] = (data['value'] - data['value'].mean()) / data['value'].std() **value standardized**  
0 2 -0.52  
1 45 0.70  
2 -23 -1.23  
3 85 1.84  
4 28 0.22  
5 2 -0.52  
6 35 0.42  
7 -12 -0.92

[**https://www.javatpoint.com/deep-learning**](https://www.javatpoint.com/deep-learning)

### Autoencoders

An autoencoder neural network is another kind of unsupervised machine learning algorithm. Here the number of hidden cells is merely small than that of the input cells. But the number of input cells is equivalent to the number of output cells. An autoencoder network is trained to display the output similar to the fed input to force AEs to find common patterns and generalize the data. The autoencoders are mainly used for the smaller representation of the input. It helps in the reconstruction of the original data from compressed data. This algorithm is comparatively simple as it only necessitates the output identical to the input.

* **Encoder:** Convert input data in lower dimensions.
* **Decoder:** Reconstruct the compressed data.

**Applications:**

* Classification.
* Clustering.
* Feature Compression.