VIETNAM GENERAL CONFEDERATION OF LABOUR

**TON DUC THANG UNIRVERSITY**

FACULTY OF INFORMATION TECHNOLOGY



**MIDTERM PROJECT**

**INTRODUCTION TO MACHINE LEARNING**

*Supervisor*: **Lê Anh Cường**

*Author*: **Nguyễn Công Minh Thành – 520K0215**

**Trịnh Bảo Toàn – 520K0332**

Class **: 20K50301**

School year  **: 24**

**HO CHI MINH CITY, 2022**

VIETNAM GENERAL CONFEDERATION OF LABOUR

**TON DUC THANG UNIRVERSITY**

FACULTY OF INFORMATION TECHNOLOGY



**MIDTERM PROJECT**

**INTRODUCTION TO MACHINE LEARNING**

*Supervisor*: **Lê Anh Cường**

*Author*: **Nguyễn Công Minh Thành – 520K0215**

**Trịnh Bảo Toàn – 520K0332**

Class **: 20K50301**

School year  **: 24**

**HO CHI MINH CITY, 2022**

# ACKNOWLEDGEMENTS

To complete this thesis, I would like to express my deep gratitude to Mr. Le Anh Cuong, for his guidance throughout the process of writing the report.

I am also grateful for the concern and support of my family and friends. It is a great source of encouragement for me to pursue and complete the text.

I sincerely thank the teachers and teachers in the Faculty of Information

Technology, Ton Duc Thang University for their dedication to imparting knowledge during my years of study. With the knowledge acquired in the

course of studying, it is not only the foundation for the thesis research process but also a valuable luggage for me to enter life firmly and confidently.

**PROJECT IS COMPLETED AT TON DUC THANG UNIVERTY**

I commit that this project is my / our own project and is supervised by Dr. Le Anh Cuong; The contents, results in this topic are honest and unpublished in any form before. The data in the tables for analyzing, commenting and evaluating are collected from various sources and by the authors and the citations are specified in References section.

In addition, a number of comments and assessments as well as data from other authors and organizations are also used in the project are referenced and annotated clearly.

**If this project has any cheating or plagiarism, I take full responsibility for the content of my project.** Ton Duc Thang University is not related to infringement of copyright caused by me during the process of this project implementation.

*Ho Chi Minh City, January 7th , 2022 Authors*

*Nguyễn Công Minh Thành – Trịnh Bảo Toàn*

# ABSTRACT

# CONTENTS

[ACKNOWLEDGEMENTS i](#_Toc116847770)

[ABSTRACT iii](#_Toc116847771)

[CONTENTS 1](#_Toc116847772)

[LIST OF SYMBOLS AND ABBREVIATIONS 2](#_Toc116847773)

[LIST OF SYMBOLS 2](#_Toc116847774)

[LIST OF ABBREVIATIONS 2](#_Toc116847775)

[LIST OF TABLES, FIGURES, GRAPHS 3](#_Toc116847776)

[LIST OF FIGURES 3](#_Toc116847777)

[LIST OF TABLES, GRAPHS 3](#_Toc116847778)

[Problem 1: 4](#_Toc116847779)

[Problem 2: 10](#_Toc116847780)

[Problem 3: 14](#_Toc116847781)

[REFERENCE 20](#_Toc116847782)

# LIST OF SYMBOLS AND ABBREVIATIONS

## LIST OF SYMBOLS

## 

## LIST OF ABBREVIATIONS

# LIST OF TABLES, FIGURES, GRAPHS

## LIST OF FIGURES

## 

## LIST OF TABLES, GRAPHS

# Problem 1:

1. **KNN Classifier**

sklearn.neighbors provides functionality for unsupervised and supervised neighbors-based learning methods. Unsupervised nearest neighbors is the foundation of many other learning methods, notably manifold learning and spectral clustering. Supervised neighbors-based learning comes in two flavors: classification for data with discrete labels, and regression for data with continuous labels.

The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples can be a user-defined constant (k-nearest neighbor learning), or vary based on the local density of points (radius-based neighbor learning).

The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice. Neighbors-based methods are known as non-generalizing machine learning methods, since they simply “remember” all of its training data (possibly transformed into a fast indexing structure such as a Ball Tree or KD Tree). Despite its simplicity, nearest neighbors has been successful in a large number of classification and regression problems, including handwritten digits and satellite image scenes.

Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular. The classes in sklearn.neighbors can handle either NumPy arrays or scipy.sparse matrices as input. For dense matrices, a large number of possible distance metrics are supported. For sparse matrices, arbitrary Minkowski metrics are supported for searches. There are many learning routines which rely on nearest neighbors at their core. One example is kernel density estimation, discussed in the density estimation section.

NearestNeighbors implements unsupervised nearest neighbors learning. It acts as a uniform interface to three different nearest neighbors algorithms: BallTree, KDTree, and a brute-force algorithm based on routines in sklearn.metrics.pairwise. The choice of neighbors search algorithm is controlled through the keyword 'algorithm', which must be one of ['auto', 'ball\_tree', 'kd\_tree', 'brute']. When the default value 'auto' is passed, the algorithm attempts to determine the best approach from the training data. For a discussion of the strengths and weaknesses of each option, see Nearest Neighbor Algorithms.



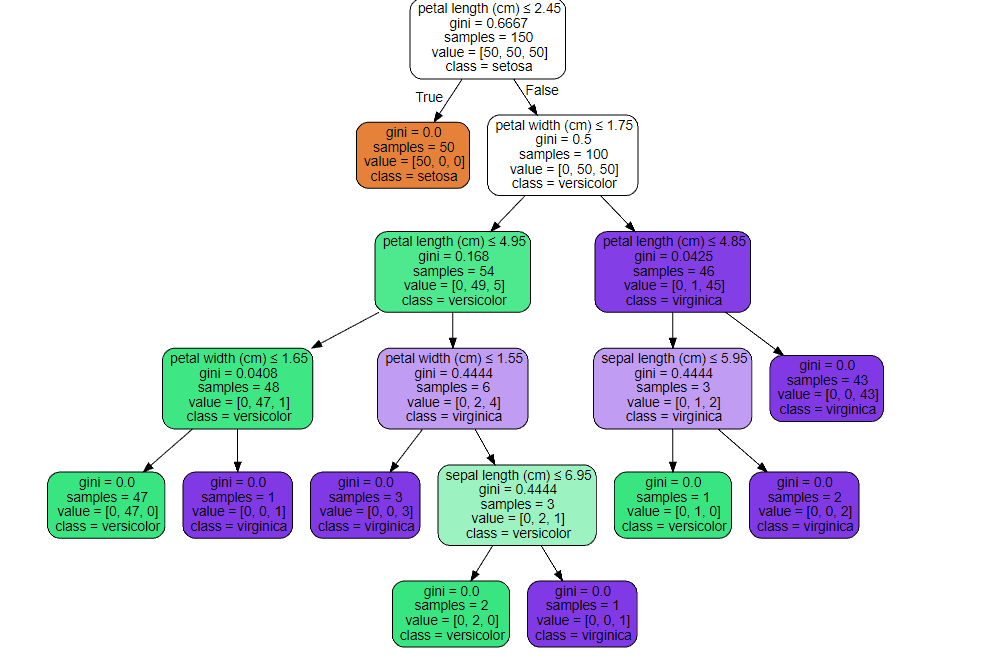
K Nearest Neighbor(KNN) is a very simple, easy to understand, versatile and one of the topmost machine learning algorithms.

KNN used in the variety of applications such as finance, healthcare, political science, handwriting detection, image recognition and video recognition. In Credit ratings, financial institutes will predict the credit rating of customers. In loan disbursement, banking institutes will predict whether the loan is safe or risky. In political science, classifying potential voters in two classes will vote or won’t vote. KNN algorithm used for both classification and regression problems. KNN algorithm based on feature similarity approach.



1. **Decision Tree Classifier**

* Decision Tree Algorithm Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.
* The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data).
* In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.



* Some advantages of decision trees are:
* Simple to understand and to interpret. Trees can be visualized.
* Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
* The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
* Able to handle both numerical and categorical data. However, the scikit-learn implementation does not support categorical variables for now. Other techniques are usually specialized in analyzing datasets that have only one type of variable. See [algorithms](https://scikit-learn.org/stable/modules/tree.html#tree-algorithms) for more information.
* Able to handle multi-output problems.
* Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.
* Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
* Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.
* The disadvantages of decision trees include:
* Decision-tree learners can create over-complex trees that do not generalize the data well. This is called overfitting. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
* Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
* Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations as seen in the above figure. Therefore, they are not good at extrapolation.
* The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
* There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
* Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.



1. **Bayesian classifier**

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset.

Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, Which can be described as:

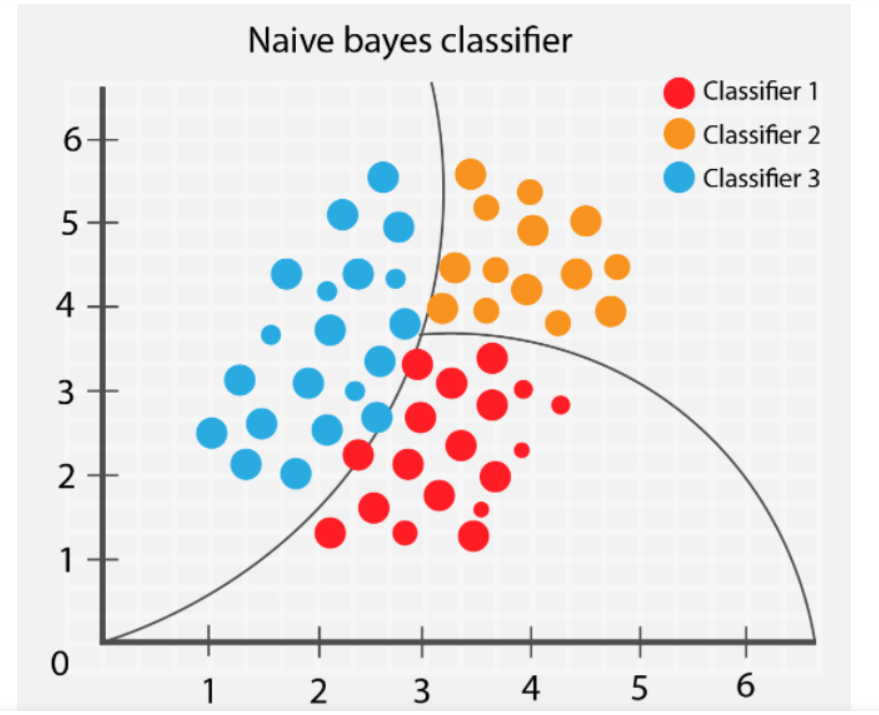
Naïve: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.

Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

* Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:

* P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
* P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
* P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
* P(B) is Marginal Probability: Probability of Evidence.



# Problem 2:

1. **Using Correlation based methods.**

Since the correlation-based feature selection (CFS) technique is a filter method, it is unrelated to the chosen classification model. As implied by the name, correlations are the sole intrinsic qualities used to assess feature subsets.

Finding a feature subset with low feature-feature correlation, which prevents redundancy, and high feature-class correlation, which preserves or boosts predictive power, is the objective.

To do so, the method uses the following equation to estimate the worth of a subset s with k features:

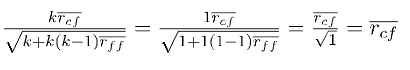


average feature-feature correlation.

 average feature-class correlation.

number of features of that subset.

The search begins with a subset that is empty and assesses each feature's quality in relation to the empty set. The feature-feature correlation may be disregarded for this phase since the denominator of the equation above can be simplified to 1 because k = 1.

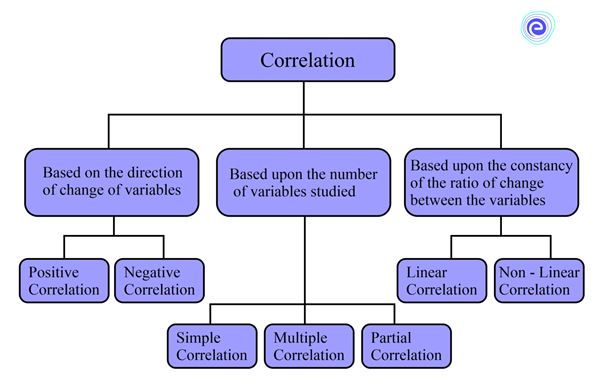


The assessment is therefore purely focused on the feature-class correlation for the initial iteration. The as-yet-empty subset receives the addition of the feature with the highest feature-class correlation. The feature that forms the best subset with the one that was already included is preserved after all features, excluding the one that was already introduced, are once again assessed.

The technique loops back to the next best unexpanded subset whenever a feature expansion does not result in an improvement. This method explores the whole feature subset space without any restrictions. Therefore, the amount of backtracking must be controlled. The feature subset that produced the greatest merit up until this point is returned by the algorithm after it reaches this limit.

The association between the features is found out by using the correlation method. There are two broad categories that can be used to measure the correlation between two random variables. One is based on classical linear correlation and the other is based on information theory. Out of these two, the most familiar measure is linear correlation coefficient. As per the standard literature, for a pair of variables (X, Y), the linear correlation coefficient ‘r’ is given by:



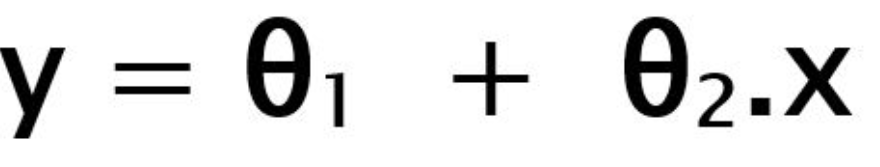


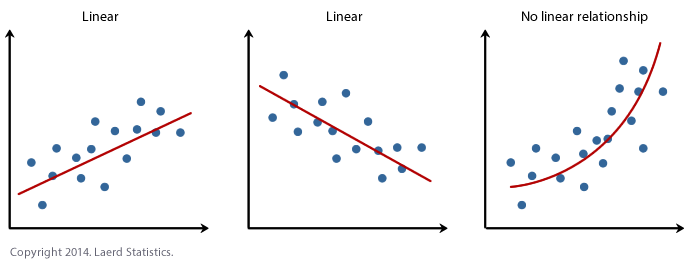
1. **MAE and Linear Regression**
2. Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.

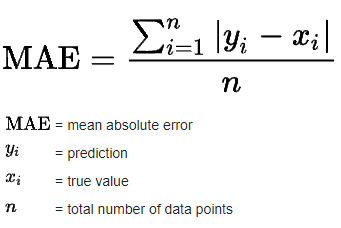
Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression. In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Hypothesis function for Linear Regression :



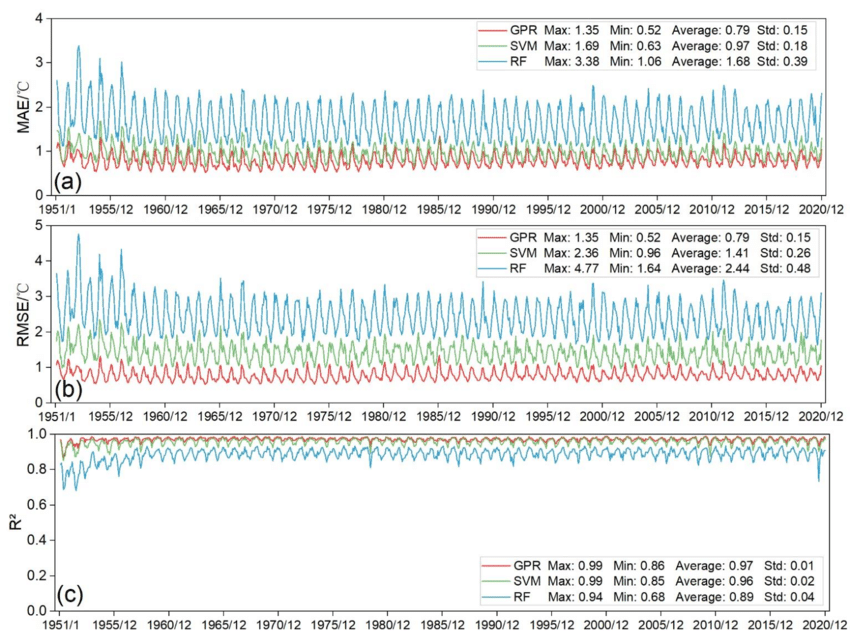


1. MAE

Mean absolute error (MAE) is a popular metric because, as with Root [mean squared error](https://www.sciencedirect.com/topics/engineering/mean-squared-error) (RMSE), see next subsection, the error value units match the predicted target value units. Unlike [RMSE](https://www.sciencedirect.com/topics/engineering/root-mean-square-error), the changes in MAE are linear and therefore intuitive. [MSE](https://www.sciencedirect.com/topics/engineering/mean-square-error) and RMSE penalize larger errors more, inflating or increasing the mean error value due to the square of the error value. In MAE, different errors are not weighted more or less, but the scores increase linearly with the increase in errors. The MAE score is measured as the average of the absolute error values. The Absolute is a mathematical function that makes a number positive. Therefore, the difference between an expected value and a predicted value can be:

The formula may look a little daunting, but the steps are easy:

1. Find all of your absolute errors, xi – x.
2. Add them all up.
3. Divide by the number of errors. For example, if you had 10 measurements, divide by 10.



Compare the difference between feature sets through Mean Absolute Error:

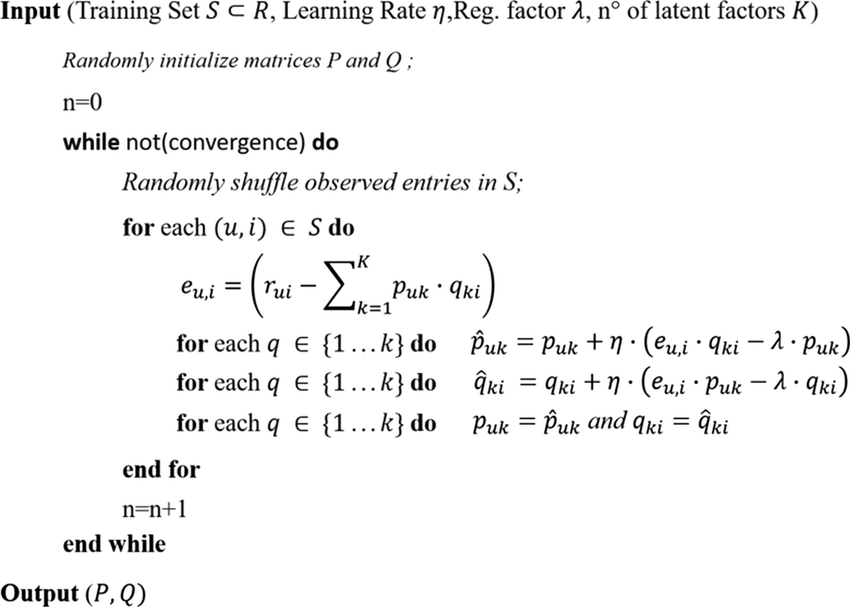
In this session we use ‘Diabetes.csv’ (include these attribute: 'Insulin', 'Glucose', 'BloodPressure', 'SkinThickness', 'BMI', 'DiabetesPedigreeFunction', 'Age')

|  |  |
| --- | --- |
| Attribute | MAE |
| Insulin | 0.412 |
| Glucose | 0.344 |
| Blood Pressure | 0.424 |
| Skin Thickness | 0.421 |
| BMI | 0.391 |
| Diabetes Pedigree Function | 0.409 |
| Age | 0.418 |

# Problem 3:

1. ***Stochastic Gradient Descent Algorithm***

* Stochastic gradient descent is an optimization algorithm often used in machine learning applications to find the model parameters that correspond to the best fit between predicted and actual outputs. It’s an inexact but powerful technique.
* Stochastic gradient descent is widely used in machine learning applications.
* Stochastic gradient descent algorithms are a modification of gradient descent. In stochastic gradient descent, you calculate the gradient using just a random small part of the observations instead of all of them. In some cases, this approach can reduce computation time.
* Online stochastic gradient descent is a variant of stochastic gradient descent in which you estimate the gradient of the cost function for each observation and update the decision variables accordingly. This can help you find the global minimum, especially if the objective function is convex.
* Batch stochastic gradient descent is somewhere between ordinary gradient descent and the online method. The gradients are calculated and the decision variables are updated iteratively with subsets of all observations, called minibatches. This variant is very popular for training neural networks.
* You can imagine the online algorithm as a special kind of batch algorithm in which each minibatch has only one observation. Classical gradient descent is another special case in which there’s only one batch containing all observations.



1. **Minibatches in Stochastic Gradient Descent**

Stochastic gradient descent begins with an initial vector of choice variables and updates it over a number of iterations, much like in the case of the ordinary gradient descent. What takes place during the iterations differs between the two:

* Minibatches are generated at random from the collection of data via stochastic gradient descent.
* The gradient is calculated and the vector is moved for each minibatch.
* You declare that an iteration, or epoch, is complete when all minibatches have been consumed and go on to the next one.

1. **Momentum in Stochastic Gradient Descent**

As you've previously seen, the gradient descent outcome may be significantly influenced by the learning rate. During the execution of the algorithm, you may modify the learning rate using a variety of alternative tactics. Additionally, you may add momentum to your algorithm.

Momentum can be used to offset the impact of learning rate. The concept is to keep track of the vector's most recent update and use it to determine the following one. Although you don't always move the vector in the direction of the negative gradient, you usually maintain the prior move's direction and magnitude.

The decay rate or decay factor parameter determines how significant the preceding update's impact is.

1. **Random Start Values**

For stochastic gradient descent, the beginning point is frequently less crucial than for standard gradient descent. Additionally, it could present an unneeded challenge for a user, especially when there are several potential outcomes. Imagine having to manually establish the parameters for a neural network with thousands of biases and weights to get an idea.

1. ***Adam Optimization Algorithm***

Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data.

The attractive benefits of using Adam on non-convex optimization problems, as follows:

* Straightforward to implement.
* Computationally efficient.
* Little memory requirements.
* Invariant to diagonal rescale of the gradients.
* Well suited for problems that are large in terms of data and/or parameters.
* Appropriate for non-stationary objectives.
* Appropriate for problems with very noisy/or sparse gradients.
* Hyper-parameters have intuitive interpretation and typically require little tuning.

Compared to traditional stochastic gradient descent, Adam is unique.

For all weight updates, stochastic gradient descent maintains a constant learning rate (referred to as alpha), which does not fluctuate throughout training.

As learning progresses, a learning rate is maintained and independently adjusted for each network weight (parameter).

Adam as combining the advantages of two other extensions of stochastic gradient descent. Specifically:

Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems).

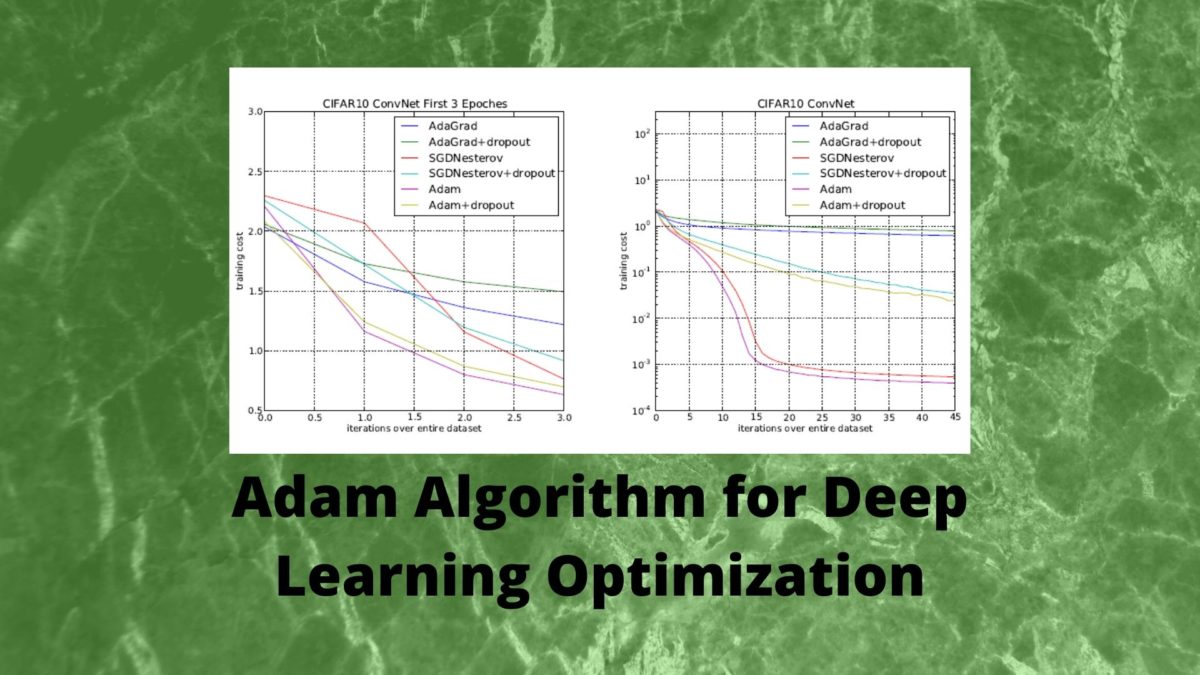
Root Mean Square Propagation (RMSProp) that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy).

Adam realizes the benefits of both AdaGrad and RMSProp.

Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance).

Specifically, the algorithm calculates an exponential moving average of the gradient and the squared gradient, and the parameters beta1 and beta2 control the decay rates of these moving averages.

* Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models.
* Adam combines the best properties of the AdaGrad and RMSProp algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems.
* Adam is relatively easy to configure where the default configuration parameters do well on most problems.



# 

# REFERENCE

<https://machinelearningcoban.com/2017/01/08/knn/>

<https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>

<https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html>

<https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>

<https://scikit-learn.org/stable/modules/tree.html>

<https://proceedings.mlr.press/v70/gal17a>

<https://ieeexplore.ieee.org/abstract/document/1396330>

<https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>

<https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c>

<https://realpython.com/gradient-descent-algorithm-python/#:~:text=Stochastic%20gradient%20descent%20is%20an,used%20in%20machine%20learning%20applications>.

<https://en.wikipedia.org/wiki/Stochastic_gradient_descent>

<https://medium.com/@ewuramaminka/mean-absolute-error-mae-machine-learning-ml-b9b4afc63077>

<https://towardsdatascience.com/comparing-robustness-of-mae-mse-and-rmse-6d69da870828>

<https://datagy.io/mae-python/>

<https://www.geeksforgeeks.org/ml-linear-regression/#:~:text=Linear%20Regression%20is%20a%20machine,relationship%20between%20variables%20and%20forecasting>.

<https://www.analyticsvidhya.com/blog/2021/05/5-regression-algorithms-you-should-know-introductory-guide/>

<https://www.cs.waikato.ac.nz/~mhall/thesis.pdf>

<https://journals.sagepub.com/doi/pdf/10.1260/1748-3018.6.3.385>

https://johfischer.com/2021/08/06/correlation-based-feature-selection-in-python-from-scratch/