MACHINE LEARNING BEGINNER TUTORIAL

Supriya Suresh

Contents

Abstract

This document describes some basic concepts of Machine Learning for beginners. This tutorial has been made assuming that target audience has no prior knowledge on Machine Learning. Each Algorithm has been explained along with basic syntax in Python as well as in R.

INTRODUCTION

What is Machine Learning?

As most of the websites site *"Machine Learning is the science that gives machines the capacity to learn and do something without being explicitly programmed"*, However according to me, the idea behind machine learning is to write an computer algorithms that will automatically improve themselves by finding patterns in existing data.

The efficiency of an algorithm relies completely on data. The more data we have for training, the more accurate it becomes. Machine learning algorithm becomes more and more accurate as the quantity of data grows over time. This is one of the reason why google offers unlimited storage for photos so that they get lots of data!

MOTIVATION

Figure 1. Applications:Google search engine, Instagram

It might sound strange but we probably use ML based applications dozens of times a day without even knowing it. Each time we do a web search on Google or Bing, that works so well because their machine learning software has figured out how to rank what pages or be it Youtube, it not only shows best results for our search but also recommends what videos we might like to watch. Facebook With it's "Face Recognition for tagging" or "Post's which we may like" options are all driven by AI. While Apple's Siri with voice recognition systems is also because of Machine Learning.

Each time we read your emails and a spam filter saves you from having to wade through tons of spam, again, that's because our computer has learned to distinguish spam from non-spam emails.

Figure 2. Identification of Spam mails

There's a lot of Machine Learning based applications hence a reason why we should start learning about how Machine Learning Algorithms work and implement our own algorithms to create some innovative product for ourselves or probably just build a better version of Jarvis!

Figure 3. Application of ML: Fitness bands

CLASSIFICAION OF MACHINE LEARNING

Machine learning is broadly classified as follows;

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

Supervised Learning

This type of learning is used when we have a data with labelled output.
So basically it consist of a target/outcome variable (or dependent variable) which is to be predicted from a given set of independent variables which is nothing but predictors/input variables. Using these set of variables, we generate a function that will map inputs to desired outputs. So whenever the model is given a new set of predictors, based on the function generated during training, it will predict the outcome.

To summarise, all data is labelled and the algorithms learn to predict the output from the input data.

Supervised learning problems can be further grouped into Regression and classification problems.

- **Classification:** A classification problem is when the type of output variable is categorical, such as "white" and "black" or "male" and "female".
- **Regression:** A regression problem is when the output variable is a real value, such as "rupees" or "weights".

Some common types of problems of classification and regression include Recommendation and Time series prediction respectively.
Some popular algorithms of supervised machine learning algorithms are:

- **Linear Regression** used for Regression problems.
- **Logistic Regression** used for Classification problems.
- **Decision Tree** used for Classification problems.
- **Random Forest** used for Classification and Regression problems.
- **Support Vector Machines** used for Classification problems.

Unsupervised Learning

Unlike Supervised Learning, this algorithm does not have any target or outcome variable to predict/estimate. It is mainly used for clustering into different groups, which can be further used for deriving some useful insights from them .

In short, Unsupervised learning is where we only have input data (X) and no corresponding output variables(Y).

Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering:** A clustering problem is used to discover the inherent groupings in the data, such as grouping customers based on their choice of purchasing.
- **Association:** An association rule learning problem used to discover rules that describe large portions of your data, such as people who buy X also tend to buy Y.

Some popular examples of unsupervised learning algorithms are:

- **k-Means** for clustering problems.
- **Apriori Algorithm** for association rule learning problems.

Reinforcement Learning

Using this algorithm, the machine is trained to make specific decisions. So here the machine is exposed to an environment where it trains itself continously using trial and error.

MOST POPULARLY USED SUPERVISED ALGORITHMS:

1). Linear Regression:

It is used for estimation of real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= m *X + c.
In this equation:

- Y – Dependent Variable/Output
- m – Slope
- X – Independent variable/predictors/input variables
- c – Intercept

Linear Regression is of mainly two types: **Simple Linear Regression** and **Multiple Linear Regression**. Simple Linear Regression is characterized by one independent variable. While Multiple Linear Regression is characterized by multiple independent variables (more than 1).

Basic Syntax of Linear regression for Python and R programmers:
***Note:** We will be using Scikit learn, so make sure you have installed Scikit learn library and other necessary libraries like Numpy, Matplotlib.pyplot.
Windows users, type the below command in your command prompt for installing Scikit-Learn,

```
pip install scikit-learn
```

Ubuntu users do the following:

```
sudo pip install scikit-learn.
```

If you don't yet have the pip tool, you can get it by following these instructions. For both Windows and Ubuntu users you can refer scikit-learn for installations.
For more Information about Scikit-learn refer this link and go through this link for tutorials.

Python code:

```
#Import other necessary libraries like pandas, numpy...
from sklearn import linear_model
#Load Train and Test datasets
x_train=input_variables_values_training_datasets
y_train=target_variables_values_training_datasets
x_test=input_variables_values_test_datasets
# Create linear regression model
linear = linear_model.LinearRegression()
# Train the model using the training sets
linear.fit(x_train, y_train)
linear.score(x_train, y_train)
#Equation coefficient and Intercept
print('Coefficient: \n', linear.coef_)
print('Intercept: \n', linear.intercept_)
#Predict Output
predicted= linear.predict(x_test)
```

R Code:

```
#Load Train and Test datasets
#Assign a variable for train and test data sets
x_train <- input_variables_values_training_datasets
y_train <- target_variables_values_training_datasets
x_test <- input_variables_values_test_datasets
#Joining the input and target values to make a complete data
x <- cbind(x_train,y_train)
# Train the model using the training sets and also study summary
linear <- lm(y_train ~ ., data = x)
summary(linear)
#Predict Output
predicted= predict(linear,x_test)
```

In the code above:

- y_train – represents dependent variable.
- x_train – represents independent variable
- x – represents training data.

2). Logistic Regression

It is used to estimate discrete values like 0/1, yes/no, true/false based on given set of independent variable(s).
Important Point: Logistic regression is used only for Classification type of problems.

Basic syntax of Logistic Regression for Python and R programmers:

Python code:

```
#Import Library
from sklearn.linear_model import LogisticRegression
#Load training and test data set like previous regression
# Create logistic regression model
model = LogisticRegression()
# Train the model using the training sets
model.fit(X, y)
model.score(X, y)
#Equation coefficient and Intercept
print('Coefficient: \n', model.coef_)
print('Intercept: \n', model.intercept_)
#Predict Output
predicted= model.predict(x_test)
```

R code:

```
#import train and test data set and follow initial steps similar to
    Linear regression.
x <- cbind(x_train,y_train)
# Train the model using the training sets
# Remember to keep the "family" as "binomial" as we are doing
    logistic regression!
logistic <- glm(y_train ~ ., data = x,family='binomial')
summary(logistic)
#Predict Output
predicted= predict(logistic,x_test)
```

3). Decision Tree

Decision Tree algorithm is mostly used for Classification kind of problems. It works for both Categorical and Continuous type of variables.

Example:- Let's say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/no). Here we know that income of customer is a significant variable but insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product and various other variables. In this case, we are predicting values for continuous variable.

Basic Syntax for Python and R programmers :

Python Code:

```
#Import Library
#Import other necessary libraries like pandas, numpy...
from sklearn import tree
#Follow steps similar to linear regression
# Create tree model
```

```
model = tree.DecisionTreeClassifier(criterion='gini') # for
    classification, here you can change the algorithm as gini or
    entropy (information gain) by default it is gini
# model = tree.DecisionTreeRegressor() for regression
# Train the model using the training sets
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

R code:

```
library(rpart)
x <- cbind(x_train,y_train)
#set some random seed
set.seed(101)
# grow tree
fit <- rpart(y_train ~ ., data = x,method="class")
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

4). Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have). Then, we perform classification by finding the hyperplane that differentiate the two classes very well.
To add a Hyperplane, SVM has a technique called the kernel trick which converts not separable problem into a separable problem, these functions are called kernels.

There are two parameters associated with SVM:

a). Gamma
b). Cost (**C**)

gamma: Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. Higher the value of gamma, will try to exact fit the as per training data set i.e. generalization error and cause over-fitting problem.
where;
rbf - radial basis function
poly - polynomial

Example: Let's see understand what happens if we have different gamma values like 0, 10 or 100. svc = svm.SVC(kernel='rbf', C=1,gamma=0).fit(X, y)

Figure 4. SVC plots for different values of gamma

C: According to official website of scikit-learn, "The C parameter trades off misclassification of training examples against simplicity of the decision surface. A low

C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors"

Figure 5. SVC plots for different values of Cost parameter

Basic syntax for Python and R programmers:

Python code:

```
#Import Library
from sklearn import svm
# Follow steps similar to linear regression
# Create SVM classification model
model = svm.svc() # there is various option associated with it, this
    is simple for classification.
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

R code:

```
#Library used for svm models is e1071
library(e1071)
# Follow initial loading steps similar to linear regression
x <- cbind(x_train,y_train)
# Fitting model
fit <-svm(y_train ~ ., data = x)
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

Watch this video for better understanding this concept.

5). Random Forest Algorithm

Random Forest is capable of performing both regression and classification tasks. It also treats missing values, outlier values and other essential steps required for data exploration.

Basic syntax for Python and R programmers:

Python code:

```
#Import Library
from sklearn.ensemble import RandomForestClassifier
# Follow initial steps for loading data sets like previous one
# Create Random Forest model
model= RandomForestClassifier()
# Train the model using the training sets
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

R code:

```
library(randomForest)
x <- cbind(x_train,y_train)
#set some random seed
set.seed(101)
```

```
# Fitting model, note that number of trees depend on the number of
    inputs and also size of data, here for instance assumed 500 trees
fit <- randomForest(Species ~ ., x,ntree=500)
summary(fit)
#Predict Output
predicted= predict(fit,x_test)
```

MOST COMMONLY USED UNSUPERVISED ALGORITHM

Clustering

In Clustering, data points are divided into a number of groups such that data points in the same groups have similar traits. In simple words, the aim is to segregate groups with similar features/properties and club them into clusters.

Though Clustering has many advanced and higher application but let's understand this with a simple example. Suppose, you are the head of a rental store and wish to understand preferences of your costumers to scale up your business. Is it possible for you to look at details of each costumer and devise a unique business strategy for each one of them? Definitely not. But, what you can do is to cluster all of your costumers into say 10 groups based on their purchasing habits and use a separate strategy for customers in each of these 10 groups.

Applications of Clustering:

Clustering has a large number of applications spread across various domains. Some of the most popular applications of clustering are:

1. Recommendation Engines
2. Market Segmentation
3. Social Network Analysis
4. Search Result Grouping

TYPES OF CLUSTERING

Most commonly used Clustering algorithms:

***Note that here two of its types have been mentioned but there are other types as well, refer this and this for more details.*

1). K-Means Clustering

K-Means is an iterative clustering algorithm that aims to find local maxima in each iteration. This algorithm works in the folowing steps :

Step 1: Plot the data points in 2-D space.

Figure 6. Plot of data points

Step 2: Specify the number of clusters K. Here for the above Example let's assume K=2.

Step 3: Specify the number of Iterations. Here let's see what happens for 5 iterations.

On first Iteration: Two cluster centroids are formed (one in Blue colour and other in red colour).Hence conveying that two clusters are formed.

Figure 7. Clusters formed on first iteration

On Second iteration: Cluster centroids move to form more likely classified cluster.

Figure 8. Clusters formed on second iteration

On third iteration: Again the cluster centroids move.

Figure 9. Clusters formed on third iteration

Fourth Iteration

Figure 10. Clusters formed on fourth iteration

Similarly, compute for other iteration as well, it's possible that you will reach a stage where even by increasing the iterations, no improvements are possible and which is an indication that we have reached the global optima which means there will be no further changes between two clusters for successive iterations. It will mark the termination of the algorithm.

For more information regarding this type of algorithm refer this link.

2). Hierarchial Clustering

Hierarchical clustering, as the name suggests is an algorithm that builds hierarchy of clusters. This algorithm starts with all the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, this algorithm terminates when there is only a single cluster left.

Here is some more indepth Information regarding Hierarchial Clustering.

END NOTES

In this tutorial we learnt about some basic Machine Learning Algorithms and also several links are mentioned along with corresponding article for more information. I suggest practice on these algorithms and work on projects and also participate in competitions hosted on kaggle, infact you can find millions of data set for practicing on kaggle. Moreover, there is a famous course on Machine Learning taught by **Andrew Ng** on Coursera.He has described each algorithm very briefly and covers almost everything.
Other resources that I would recommend is a Youtube channel sentdex.
And incase if you are not familiar with Python then go through this, and also a youtube channel thenewboston.
For mastering in R programming language I would recommend 100 free tutorials for learning R and R programming tutorial playlist.This is the best platform to learn Basic and Advanced R. It covers every aspect of data analytics from Data Cleaning, Manipulation, Visualization, Machine Learning Algorithms.

References:

- https://www.analyticsvidhya.com/blog/2013/11/getting-clustering-right/
- http://www.aihorizon.com/essays/generalai/supervised_unsupervised_machine_learning.htm

- https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/
- https://www.datacamp.com/community/tutorials/machine-learning-python#gs.4f2UCnU
- https://www.datacamp.com/community/tutorials/machine-learning-python
- http://dataconomy.com/2017/03/beginners-guide-machine-learning/
- https://www.toptal.com/machine-learning/machine-learning-theory-an-introductory-primer
- https://discuss.analyticsvidhya.com/t/decision-tree-with-continuous-variables/201
- http://www.listendata.com/p/r-programming-tutorials.html
- http://machinelearningmastery.com/start-here/
- http://machinelearningmastery.com/self-study-guide-to-machine-learning/
- https://sites.google.com/site/dataclusteringalgorithms/hierarchical-clustering-algorithm