|  |  |  |
| --- | --- | --- |
| **UMBC – CSEE Department** |  | Dr. Waleed Youssef |
| **Data Science Program** |  | youssef1@umbc.edu |
| **Spring 2023** |  |  |
| **DATA 603 – Big Data Platforms** | | |
|  |  |  |
| **Homework #11 – ML and NN** | | |
|  |  |  |

1. **[10 Points]** Describe the regression learning mechanism?

Regression is a method for understanding the relationship between independent variables or features and a dependent variable or outcome. Outcomes can then be predicted once the relationship between independent and dependent variables has been estimated. Regression learning is a type of supervised learning in machine learning, where the goal is to predict a continuous target variable based on one or more input features. These models will learn the relationship between input and output data from labelled training data. It can then forecast future trends or predict outcomes from unseen input data, or be used to understand gaps in historic data.

Linear regression finds the linear relationship between the dependent variable and one or more independent variables using a best-fit straight line. Generally, a linear model makes a prediction by simply computing a weighted sum of the input features, plus a constant called the bias term (also called the intercept term). In this technique, the dependent variable is continuous, the independent variable(s) can be continuous or discrete, and the nature of the regression line is linear.

To evaluate predictions there are two important mechanisms: Variance, and Bias.

**Variance**: Variance is the amount by which the estimate of the target function changes if different training data were used. To avoid false predictions, we need to make sure the variance is low.

**Bias:** Bias is the algorithm’s tendency to consistently learn the wrong thing by not taking into account all the information in the data. For the model to be accurate, bias needs to be low.

The following steps are involved in the regression learning mechanism:

Data preparation: The first step is to prepare the data for training. This involves collecting,cleaning the data, removing any outliers, and formatting the data in a way that the regression algorithm can understand.

Model selection: The next step is to select a regression algorithm. There are many different regression algorithms available, such as Linear, Logistic. others. The choice of algorithm will depend on the problem being solved.

Model training: Once a regression algorithm has been selected, it needs to be trained on the data. This involves finding the parameters of the model that best fit the data.

Model evaluation: Once the model has been trained, it needs to be evaluated to see how well it performs on new data. This is done by using a test data that was not used to train the model. Usually, we train with 80% data and keep the remaining 20% untouched for testing the model.

Model deployment: Once the model has been evaluated and found to be satisfactory, it can be deployed to production. This involves making the model available to users so that they can use it to make predictions.

**Reference:** [Machine Learning Regression Explained - Seldon](https://www.seldon.io/machine-learning-regression-explained)

[Regression in Machine Learning: What It Is & Examples | Built In](https://builtin.com/data-science/regression-machine-learning)

1. **[10 Points]** Explain how Neural Networks identify weights and factors in supervised learning environments?

Neural networks are a type of machine learning algorithm that can be used to solve a wide variety of problems, including classification, regression, and natural language processing. Neural networks are inspired by the human brain, and they work by learning from data.

In a supervised learning environment, the neural network is given a set of training data that includes both the input and the desired output. The neural network then uses this data to learn the weights and factors that are needed to make accurate predictions.

The weights and factors are learned using a process called backpropagation. Backpropagation is an iterative process that starts with the output layer of the neural network and works its way back to the input layer. At each layer, the weights and factors are updated to minimize the error between the predicted output and the desired output.

The neural network continues to learn until it converges on a set of weights and factors that produce low error. Once the neural network has converged, it can be used to make predictions on new data.

Here is how back propagation works:

**Initialization**: The weights of the neural network are initialized randomly.

**Forward Propagate**: We start from the input we have, we pass them through the network layer and calculate the actual output of the model straightforwardly. This step is called forward-propagation, because the calculation flow is going in the natural forward direction from the input -> through the neural network -> to the output.

**Loss Function**: The output of the neural network is compared to the actual output label, and the difference between them is calculated using a loss function. The loss function measures how well the neural network is performing on the training data.

**Back Propagate**: The weights of the neural network are adjusted based on the gradient of the loss function with respect to the weights. This process is known as backpropagation. The gradient tells us in which direction and by how much the weights need to be adjusted to minimize the loss.

**Weight Update**: The weights of the neural network are updated based on the gradient computed in the previous step. The learning rate is used to control the size of the weight updates.

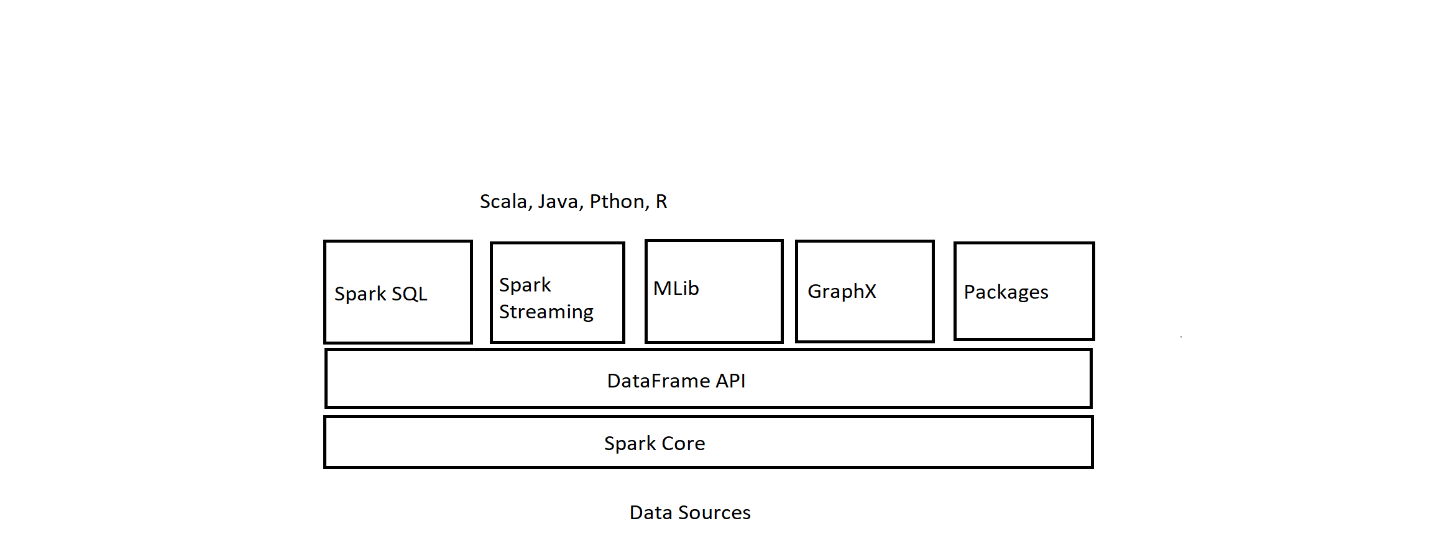
**Iterate until convergence**: Above steps are repeated for each record in the training data until the neural network converges to a solution.

**Reference:** <https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e>

1. **[10 Points]** Perform the following:

* Watch this video:
  + <https://www.youtube.com/watch?v=DBxcua0Vmvk>
* Develop a quick report summarizing what you learned in this video. Limit your report to no more than one page

Spark is a unified data processing engine. It is based on data frame API.



The goal of MLib component is to

* Help make practical machine learning scalable and easy by leveraging the distributed nature of Spark to run the required tasks.
* Simplify the development and deployment of scalable machine learning pipelines.

MLib package has four main Components they are:

1. Algorithms – Classification, Regression, Clustering, etc.
2. Pipeline - Constructing, Evaluating, Tuning, Persistence.
3. Featurization – Extraction, Transformation.
4. Utilities – Linear algebra, Statistics.

Machine Learning Pipeline and its corresponding MLib feature:

1. Load/Clean Data – Load/Clen Data
2. Feature Extraction - Transformer
3. Model Training - Estimator
4. Model Evaluation – Evaluator

Transformer: It is a preprocessor step for feature extraction. It transforms data into a format required/understandable to the algorithm. Some of the transformations are:

* Normalize the data
* Tokenization – Converting the sentences into words.
* Converting categorical values into numbers.

Estimator: It is the Machine Learning algorithm that we develop. It trains or fits the data.

Evaluator: Evaluates the model performance based on certain metrics such as:

Root Mean Square Error (RMSE), Receiver Operating Characteristic(ROC)

It helps with automating the model tuning process such as:

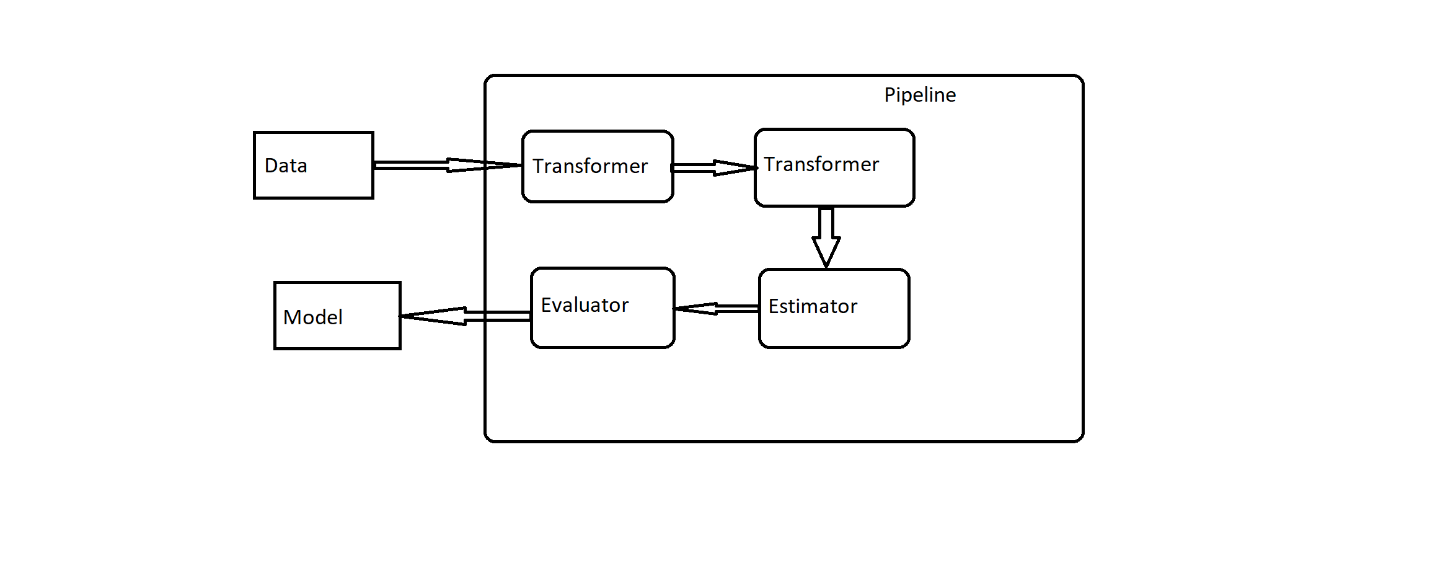
* Comparing model performance.
* Select the best model for generating predictions.

E.g.: BinaryClassificationEvaluator, CrossValidator

Using Transformer, Estimator, Evaluator we can create a pipeline which can be used for:

* To represent a ML workflow.

It consists of set of stages which has to be executed in an order.



1. **[30 Points]** Programming assignment:

Write a Spark program to load images continuing dogs, cats and horses, then use the developed model to identify 10 different pictures presented to your program including animals not in your training data set (such as a rabbit)?

import os

import sys

from pyspark.sql import SparkSession

from pyspark.ml import Pipeline

from pyspark.ml.image import ImageSchema

from pyspark.ml.clustering import KMeans

# Set up a SparkSession

spark = SparkSession.builder.appName("Image Classification").getOrCreate()

# Define the paths to the training and test data

train\_path = os.path.abspath("/content/sample\_data/Images")

test\_path = os.path.abspath("/content/sample\_data/Images\_test")

# Define a pipeline to process the data

train\_data = spark.read.format("image").load(train\_path).alias("train\_data")

kmeans\_model = KMeans().setK(3).setSeed(1).fit(train\_data.select("image")).alias("model")

test\_data = spark.read.format("image").load(test\_path).alias("test\_data")

predictions = kmeans\_model.transform(test\_data).alias("predictions")

pipeline = Pipeline(stages=[train\_data, kmeans\_model, test\_data, predictions])

# Run the pipeline and display the results

predictions = pipeline.fit().transform()

predictions.select("image", "prediction").show(truncate=False)

* Your program should use an image data set of 100 images for training
* Your validation data set is 10 images
* Show illustrations and graphs of your results