

School of Computer Science and IT Department of Master of Computer Applications

Jain Knowledge Campus

#44/4 District Fund Road Jayanagar 9th Block Behind Big Bazaar, Bengaluru, Karnataka 560069

[**C# and .Net Technologies**](https://classroom.google.com/u/1/c/MTI2NDMzMzM1NTQ1)

**Activity: 1**

**Project On:**

**Feedback Model Using C# Machine Learning**

**Team Members: Tadesse Mega, Yogesh Kumar**

**Name: Yogesh Kumar**

**USN: 19MCAR0035**

**Semester: 3rd Semester**

**Branch: MCA**

**Academic Year: 2020**

**Acknowledgement**

The successful completion of any task would be incomplete without thanking the people who made it possible whose constant guidance and encouragement motivated my effort with success.

It is my great pleasure to thank my guide “**Prof.Kathireshan V.** “for his excellent guidance, constant encouragement, support, constructive suggestions.

I, also thank all the faculties of the Computer Application Department for their suggestions enabled us to surpass many of the seemingly impossible hurdles.

Finally, I would like to show my gratitude to my family members and all friends for advice and kind co-operation for without which this project would have been just a dream.

Yogesh Kumar

* **Analyze sentiment of website comments/feedback/ with binary classification in ML.NET**

For this project , first we create/gather the following

* Prepare data
* a console application
* **Create** 2 classes , FeedbackTrainingData  class and FeedbackPrediction class to test predictions data
* Load the data in main class
* Build and train the model
* Evaluate the model
* Use the model to make a prediction
* See the results

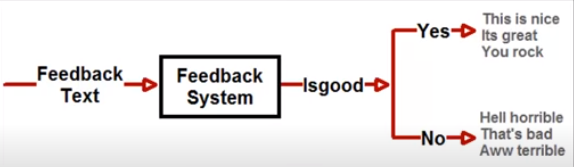
**IDE used**

* Visual Studio 2017 version 15.6 or later with the ".NET Core cross-platform development" workload installed

**1. data Source**

* The datasets for this project are taken from the 'From Group to Individual Labels using Deep Features', Kotzias et. al,. KDD 2015, and hosted at the UCI Machine Learning Repository - Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

**General Sturcture a project**

****

### data preparation

Total collected data

| **total collected data given to algo.** | |
| --- | --- |
| **Feed back text(features)**   * **Input given to algo.** | **Sentiment (Label)**   * **Output we expected from algo** |
| I did’t like it | 0(negative) |
| It’s horrible | 0 |
| Wow... Loved this | 1(positive) |
| Wow super | 1 |
| I did’t understand | 0(negative) |
| Soo nice | 1(positive) |
| It’s nice | 1(positive) |
| This is good website | 1(positive) |
| Not good | 0(negative) |
| Complicated website | 0(negative) |
| It’s cool | 1(positive) |
| Noting good | 0(negative) |
| It’s best | 1(positive) |
| I understand a lot thanks | 1(positive) |
| great | 1(positive) |

**Training data /80%**

| **Training data given to algo.** | |
| --- | --- |
| **Sentiment Text /Feed back text(features)**   * **Input given to algo.** | **Sentiment (Label)**   * **Output we expected from algo** |
| I did’t like it | 0(negative) |
| It’s horrible | 0 |
| Wow... Loved this | 1(positive) |
| Wow super | 1 |
| I did’t understand | 0(negative) |
| Soo nice | 1(positive) |
| It’s nice | 1(positive) |
| This is good website | 1(positive) |
| Not good | 0(negative) |
| Complicated website | 0(negative) |
| It’s cool | 1(positive) |
| Noting good | 0(negative) |

**Test data**

| **Training data given to algo.** | |
| --- | --- |
| **Sentiment Text /Feed back text(features)**   * **Input given to algo.** | **Sentiment (Label)**   * **Output we expected from algo** |
| I did’t like it | 0(negative) |
| It’s horrible | 0 |
| It’s nice | 1(positive) |

**2.Create a console application**

1. Create a **.NET Core Console Application**  called "SentimentAnalysis".
2. Install the **Microsoft.ML NuGet Package**:

NuGet( New Get)—Is a free and open source package manager designed for the Microsoft development platform

* It helps developer to create ,share ,and consume useful .NET libraries
* **Steps to install**

In Solution Explorer, right-click on your project and select **Manage NuGet Packages**. and then select the **Browse** tab. Search for **Microsoft.ML**, select the package want, and then select the **Install** button. Proceed with the installation by agreeing to the license terms for the package choose.

1. **Create classes**

* 2 classes are created for input data Training and for test data predictions.:
* two classes  are

FeedbacKTrainingData

* we define input data structure for algo..
* has two properties IsGood Which store the out put what we are expected from Algo. And FeedBackText Which stores input Which are given to algo.

FeedbackPrediction

* output data structure for algo..
* has properties IsGood Which store the Predication output result

class FeedBackTrainingData

{

public bool IsGood { get; set; }

public string FeedBackText { get; set; }

}

class FeedBackPrediction

{

public bool IsGood{ get; set; }

}

FeedbackPrediction is the prediction class used after model training.

* that the input SentimentText can be displayed along with the output prediction.
* The Prediction boolean is the value that the model predicts when supplied with new input  SentimentText.

1. create two generic list and inside it fill trainingdata And TestData using functions

static List<FeedBackTrainingData> trainingdata = new List<FeedBackTrainingData>();

static List<FeedBackTrainingData> testData = new List<FeedBackTrainingData>();

static void LoadTestData()

{

testData.Add(new FeedBackTrainingData()

{

FeedBackText = "i hate that one",

IsGood = false

});

testData.Add(new FeedBackTrainingData()

{

FeedBackText = "this is so bad",

IsGood = false

});}

static void LoadTrainingData()

{

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "waw good ",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "very bad",

IsGood =false

});}

1. In the Main Function

Step 1 –the 1st step in ML.net Machine Learning is We need to Load training Data

**Load the data**

static void Main(string[] args)

{

LoadTrainingData();

Step 2---create object for MLContext class

var mlContext = new MLContext();

step 3---- convert data into IDataView

IDataView dataView = mlContext.CreateStreamingDataView<FeedBackTrainingData>(trainingdata);

Step 4---we need to create the pipe line and define the work in it

var pipeline = mlContext.Transforms.Text.FeaturizeText("FeedBackText", "Features")

.Append(mlContext.BinaryClassification.Trainers.FastTree

(numLeaves: 50, numTrees: 50, minDatapointsInLeaves: 1));

Step 5-- Traing the algorithm and we want the model out

Console.WriteLine("=============== the accurcy of model is ===============");

Step 6-Load the our test data and run the test data to check our models accuracy

LoadTestData();

IDataView dataView1 = mlContext.CreateStreamingDataView<FeedBackTrainingData>(testData);

var model = pipeline.Fit(input: dataView);

var predictions = model.Transform(dataView1);

Console.WriteLine();

var metrics = mlContext.BinaryClassification.Evaluate(predictions, "Label");

Console.WriteLine($"Accuracy: {metrics.Accuracy:P2}");

Console.ReadLine();

Step 7..use the model

string strcont = "y";

while (strcont == "y")

{

Console.WriteLine("Enter a FeedBack text");

string feedbackstring = Console.ReadLine().ToString();

var predictionFunction = model.MakePredictionFunction

<FeedBackTrainingData, FeedBackPrediction>

(mlContext);

var feedbackinput = new FeedBackTrainingData();

feedbackinput.FeedBackText = feedbackstring;

var feedbackpredicted = predictionFunction.Predict(feedbackinput);

//Console.WriteLine("Predicted result shows:-" + $"predication:{(Convert.ToBoolean(feedbackpredicted.IsGood) ?"positive":"Negative")}");

//Console.WriteLine("Predicted result shows:-" + feedbackpredicted.IsGood);

Console.WriteLine("Predicted result shows:");

Console.WriteLine(Convert.ToBoolean(feedbackpredicted.IsGood) ? "positive" : "Negative");

}

Console.ReadLine();

}

}

}

Code for the project

using System;

using System.Collections.Generic;

using Microsoft.ML.Runtime.Data;

using Microsoft.ML;

using Microsoft.ML.Runtime.Api;

namespace Lab1

{

class FeedBackTrainingData

{

[Column(ordinal: "0", name: "Label")]

public bool IsGood { get; set; }

[Column(ordinal: "1")]

public string FeedBackText { get; set; }

}

class FeedBackPrediction

{

[ColumnName("PredictedLabel")]

public bool IsGood{ get; set; }

}

class Program

{

static List<FeedBackTrainingData> trainingdata = new List<FeedBackTrainingData>();

static List<FeedBackTrainingData> testData = new List<FeedBackTrainingData>();

static void LoadTestData()

{

testData.Add(new FeedBackTrainingData()

{

FeedBackText = "i did’t like it",

IsGood = false

});

testData.Add(new FeedBackTrainingData()

{

FeedBackText = "it’s horriable",

IsGood = false

});

testData.Add(new FeedBackTrainingData()

{

FeedBackText = "it’s nice",

IsGood = true

});

}

static void LoadTrainingData()

{

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "wow super",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "I did’t like it",

IsGood =false

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "it’s horriable",

IsGood = false

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = " I did’t understand",

IsGood = false

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "sooo nice",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "it’s nice",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "this is good website ",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = " Wow... Loved this ",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "not good",

IsGood = false

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "complicated website",

IsGood = false

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "it’s cool",

IsGood = true

});

trainingdata.Add(new FeedBackTrainingData()

{

FeedBackText = "noting good",

IsGood = false

});

}

static void Main(string[] args)

{

//step 1 we need to load training data

LoadTrainingData();

//create object for MLContext

var mlContext = new MLContext();

// convert your data into IDataView

IDataView dataView = mlContext.CreateStreamingDataView<FeedBackTrainingData>(trainingdata);

//we need to create the pipe line and define the work in it

var pipeline = mlContext.Transforms.Text.FeaturizeText("FeedBackText", "Features")

.Append(mlContext.BinaryClassification.Trainers.FastTree

(numLeaves: 50, numTrees: 50, minDatapointsInLeaves: 1));

// Traing the algorithem and we want the model out

Console.WriteLine("=============== the accuracy of model is ===============");

// Load the our test data and run the test data to check our models accuracy

LoadTestData();

IDataView dataView1 = mlContext.CreateStreamingDataView<FeedBackTrainingData>(testData);

var model = pipeline.Fit(input: dataView);

var predictions = model.Transform(dataView1);

Console.WriteLine();

var metrics = mlContext.BinaryClassification.Evaluate(predictions, "Label");

Console.WriteLine($"Accuracy: {metrics.Accuracy:P2}");

// Console.WriteLine(metrics.Accuracy);

Console.ReadLine();

//use the model

string strcont = "y";

while (strcont == "y")

{

Console.WriteLine("Enter a FeedBack text");

string feedbackstring = Console.ReadLine().ToString();

var predictionFunction = model.MakePredictionFunction

<FeedBackTrainingData, FeedBackPrediction>

(mlContext);

var feedbackinput = new FeedBackTrainingData();

feedbackinput.FeedBackText = feedbackstring;

var feedbackpredicted = predictionFunction.Predict(feedbackinput);

//Console.WriteLine("Predicted result shows:-" + $"predication:{(Convert.ToBoolean(feedbackpredicted.IsGood) ?"positive":"Negative")}");

//Console.WriteLine("Predicted result shows:-" + feedbackpredicted.IsGood);

Console.WriteLine("Predicted result shows:");

Console.WriteLine(Convert.ToBoolean(feedbackpredicted.IsGood) ? "positive" : "Negative");

}

Console.ReadLine();

}

}

}

Output:

