Global AI Community

Machine Learning using C# on Jupyter Notebook!

Introduction

- Cloud Architect @ Harman, A Samsung Company
- Domain: Professional Audio, Video and Control
- Area of Expertise: Cloud, Distributed computing
- Area of Interest: AI/ML, Cloud and IoT
- Location: Bangalore
- Member: .Net Foundation

Agenda

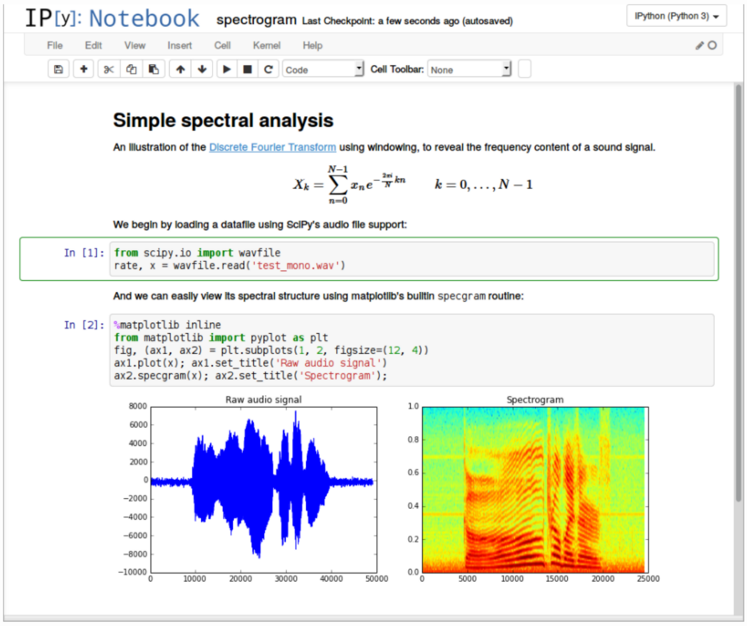
- Jupyter Notebook
- .Net on Juypter Notebook
- Prerequisites & Installation
- Machine Learning – ML.Net
- Demo – Sentiment Analysis

Jupyter Notebook

A notebook = Code + Output (Visualizations/text/equations/media)

- Open source web application maintained by [Project Jupyter]
- Live code
- Easy to share notebooks
- Stores results from previous execution
- Mainly used by Data scientists

Jupyter Notebook



Simple spectral analysis

An illustration of the Discrete Fourier Transform using windowing, to reveal the frequency content of a sound signal.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

We begin by loading a datafile using SciPy's audio file support:

In [1]:
```
from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin `specgram` routine:

In [2]:
```
%matplotlib inline
from matplotlib import pyplot as plt
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
ax1.plot(x); ax1.set_title('Raw audio signal')
ax2.specgram(x); ax2.set_title('Spectrogram');
```

.Net on Jupyter Notebook

```
public class Employee
{
    public Employee(string firstName, string lastName)
    {
        FirstName = firstName;
        LastName = lastName;
    }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string FullName => $"{FirstName}_{LastName}";
}
var developer = new Employee("Praveen", "Raghuvanshi");
display(developer.FullName);
```

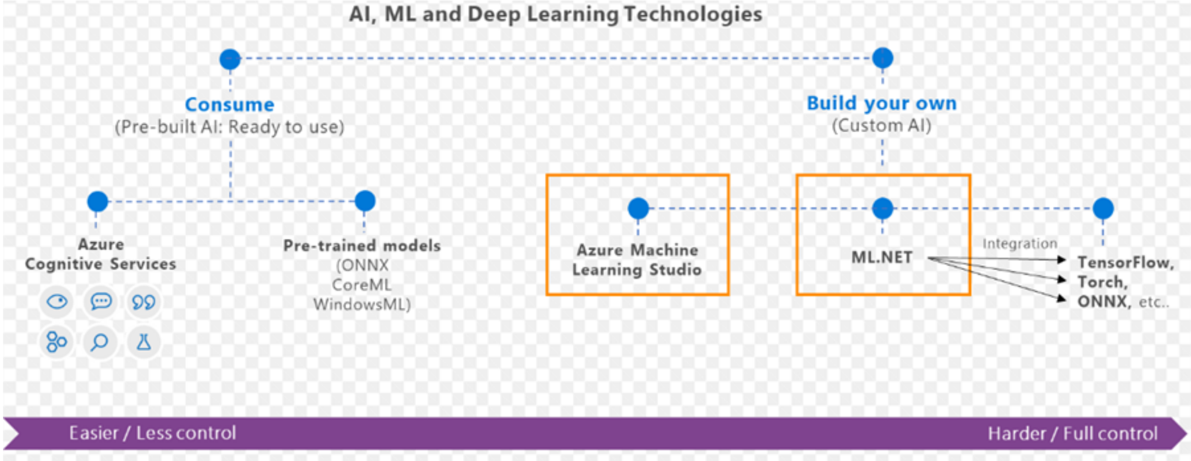
```
Praveen_Raghuvanshi
```

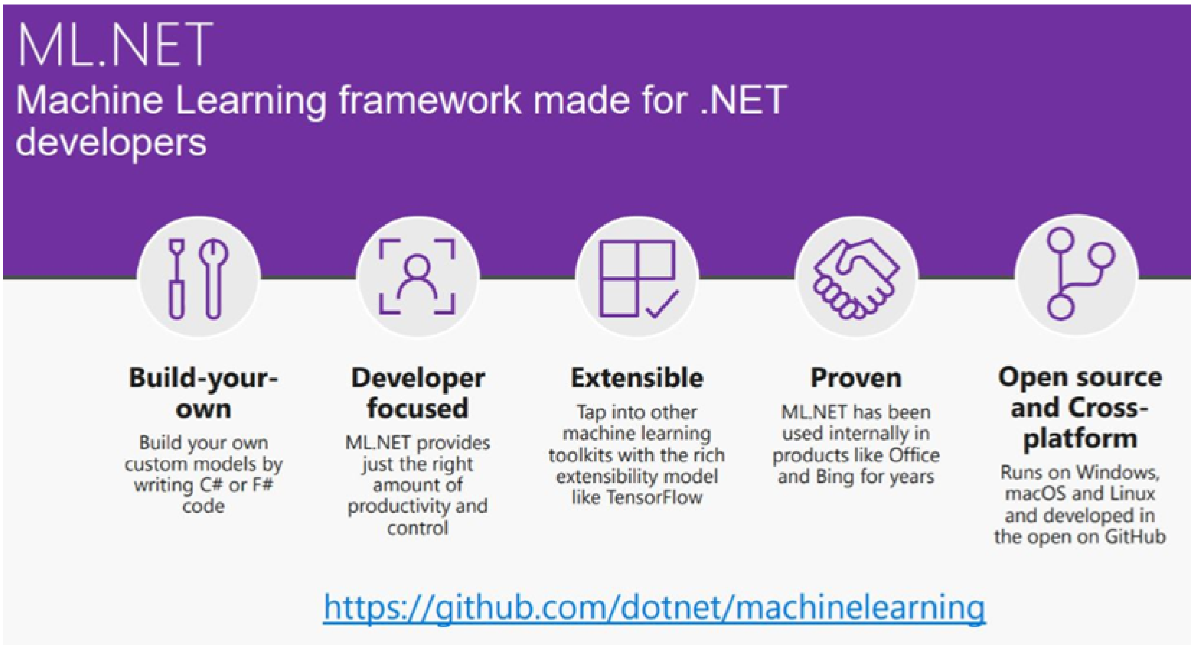
Prerequisites and Installation

- Jupyter (Easiest way is to install Anaconda)
- Latest .Net core
- dotnet interactive : Installation
- Enable the .NET kernel for Jupyter

ML.Net

ML framework from Microsoft for developing Custom AI/ML applications. Originated in 2002 as part of Microsoft Research project





ML.NET

Machine Learning framework made for .NET developers

Build-your-own

Build your own custom models by writing C# or F# code

Developer focused

ML.NET provides just the right amount of productivity and control

Extensible

Tap into other machine learning toolkits with the rich extensibility model like TensorFlow

Proven

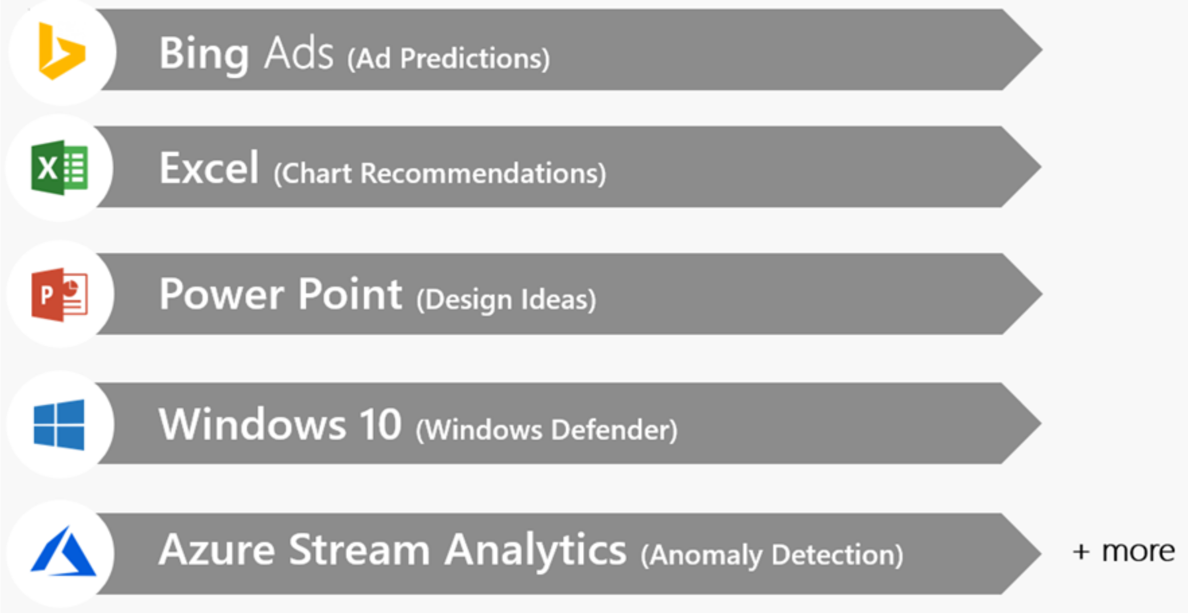
ML.NET has been used internally in products like Office and Bing for years

Open source and Cross-platform

Runs on Windows, macOS and Linux and developed in the open on GitHub

https://github.com/dotnet/machinelearning

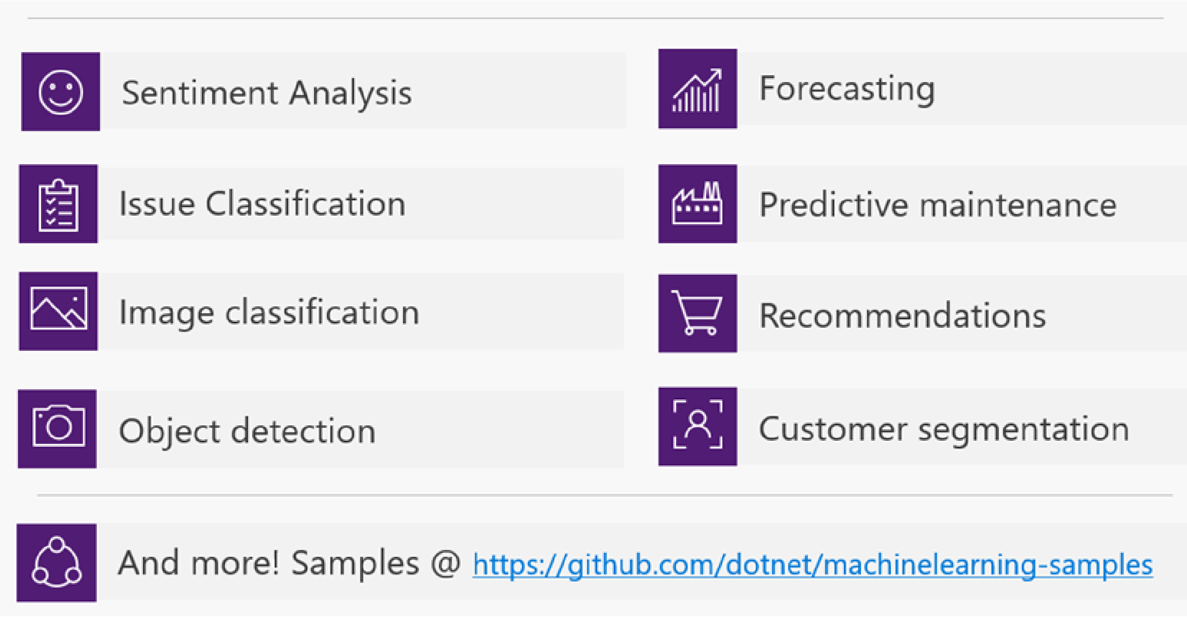
Proven at scale, Enterprise ready



- Bing Ads (Ad Predictions)
- Excel (Chart Recommendations)
- Power Point (Design Ideas)
- Windows 10 (Windows Defender)
- Azure Stream Analytics (Anomaly Detection)

+ more

Possibilities



Sentiment Analysis

Forecasting

Issue Classification

Predictive maintenance

Image classification

Recommendations

Object detection

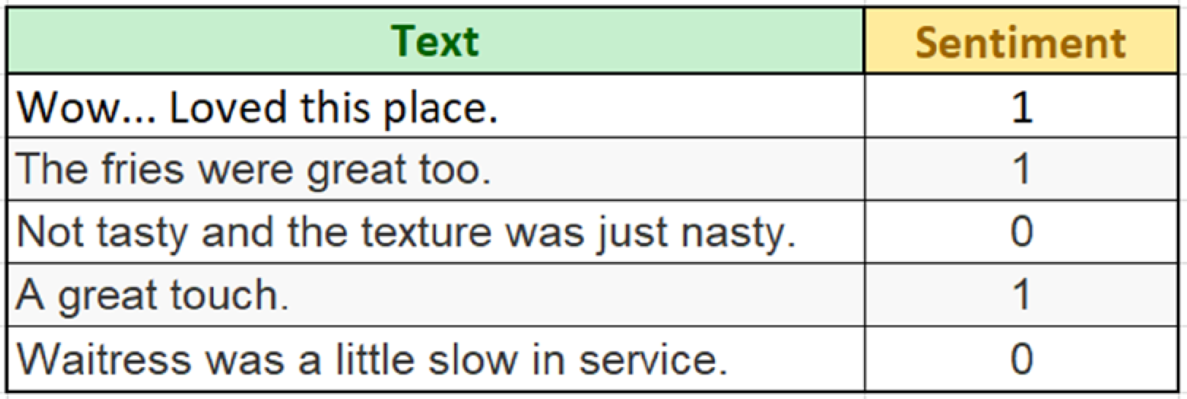
Customer segmentation

And more! Samples @ https://github.com/dotnet/machinelearning-samples

Demo - Sentiment Analysis

- Positive(+ve) , Negative(-ve)
- Sentiment(Label) and Text(Features)
- Train model using ML.Net
- ML Pipeline : Load -> Transform -> Train -> Evaluate -> Predict

Dataset - Yelp Reviews



Text	Sentiment
Wow... Loved this place.	1
The fries were great too.	1
Not tasty and the texture was just nasty.	0
A great touch.	1
Waitress was a little slow in service.	0

1. Define Application wide Items

Nuget Packages

- Microsoft.ML

```
// ML.NET Nuget packages installation
#r "nuget:Microsoft.ML"
```

Installed package Microsoft.ML version 1.5.1

Namespaces

```
using Microsoft.ML;
using Microsoft.ML.Data;
using static Microsoft.ML.DataOperationsCatalog;
using System;
using System.IO;
```

2. Load data

Set the dataset path

```
var dataPath = Path.Combine(Environment.CurrentDirectory, "Data",
"yelp_labelled.txt");
display(dataPath)
```

```
D:\Praveen\sourcecontrol\github\praveenraghuvanshi\tech-sessions\03092020-Global-
AI-Community\Data\yelp_labelled.txt
```

Define Schema(classes) for input data and predictions

```
/// Input
public class SentimentData
{
    [LoadColumn(0)]
    public string SentimentText;

    [LoadColumn(1), ColumnName("Label")]
    public bool Sentiment;
}

/// Prediction
public class SentimentPrediction : SentimentData
{

    [ColumnName("PredictedLabel")]
    public bool Prediction { get; set; }

    public float Probability { get; set; }

    public float Score { get; set; }
}
```

```
// Initialize ML Context
MLContext mlContext = new MLContext();
```

```
// Load : Split it into 80% training and 20% test data
public TrainTestData LoadData(MLContext mlContext)
{
    IDataView dataView = mlContext.Data.LoadFromTextFile<SentimentData>(dataPath, hasHeader: false);
    TrainTestData splitDataView = mlContext.Data.TrainTestSplit(dataView, testFraction: 0.2);
    return splitDataView;
}
```

```
TrainTestData splitDataView = LoadData(mlContext);
```

3. Transform data and choose algorithm

```
// Transform : Converts the text column(SentimentText) into numeric type Features column using FeaturizeText
var estimator = mlContext.Transforms.Text.FeaturizeText(outputColumnName: "Features", inputColumnName: nameof(SentimentData.SentimentText))

.Append(mlContext.BinaryClassification.Trainers.SdcaLogisticRegression(labelColumnName: "Label", featureColumnName: "Features"));
```

4. Train Model

```
// Train/Fit model
display("=============== Create and Train the Model ===============");
var model = estimator.Fit(splitDataView.TrainSet);
display("=============== End of training ===============");
```

```
=============== Create and Train the Model ===============
```

```
=============== End of training ===============
```

5. Evaluate Model

```
// Evaluate : Evaluate performance of the model using Test set
Console.WriteLine("=============== Evaluating Model accuracy with Test data===============");
IDataView predictions = model.Transform(splitDataView.TestSet);
CalibratedBinaryClassificationMetrics metrics = mlContext.BinaryClassification.Evaluate(predictions, "Label");
```

```
=============== Evaluating Model accuracy with Test data===============
// Display Metrics
display("Model quality metrics evaluation");
display("--------------------------------");
display($"Accuracy: {metrics.Accuracy:P2}");
display($"Auc: {metrics.AreaUnderRocCurve:P2}");
display($"F1Score: {metrics.F1Score:P2}");
display("=============== End of model evaluation ===============");
```

```
Model quality metrics evaluation
```

```
--------------------------------
```

```
Accuracy: 83.96%
```

```
Auc: 90.05%
```

```
F1Score: 84.54%
```

```
=============== End of model evaluation ===============
```

6. Prediction

```
// Create PredictionEngine passing above model
var predictionFunction = mlContext.Model.CreatePredictionEngine<SentimentData,
SentimentPrediction>(model);
```

```
// Create sample text
SentimentData sampleStatement = new SentimentData
{
    SentimentText = "This was a very bad steak"
};

// Predict
var resultPrediction = predictionFunction.Predict(sampleStatement);
```

```
// Display Prediction
display("=============== Prediction Test of model with a single sample and test
dataset ===============");
display($"Sentiment: {resultPrediction.SentimentText} | Prediction:
{(Convert.ToBoolean(resultPrediction.Prediction) ? "Positive" : "Negative")} |
Probability: {resultPrediction.Probability} ");
display("=============== End of Predictions ===============");
```

```
=============== Prediction Test of model with a single sample and test dataset
===============
```

```
Sentiment: This was a very bad steak | Prediction: Negative | Probability:
0.031075107
```

```
=============== End of Predictions ===============
```

7. Save Model

```
// Save Model
mlContext.Model.Save(model,
splitDataView.TrainSet.Schema,"SentimentAnalysisModel.zip");
```

Resources

- Github
- Deck + Notebook
- Source

References

- How to Use Jupyter Notebook in 2020: A Beginner's Tutorial
- .Net interactive
- Using ML.NET in Jupyter notebooks
- RISE-Jupyter/IPython Slideshow Extension

Thank you

Contact

Twitter : @praveenraghuvan\
LinkedIn : https://in.linkedin.com/in/praveenraghuvanshi \
Github : https://github.com/praveenraghuvanshi \
dev.to : https://dev.to/praveenraghuvanshi

I am running an unofficial **telegram** group for ML.Net enthusiasts, please feel free to join it at https://t.me/joinchat/IifUJQ_PuYT757Turx-nLg