Sentiment

Write Code

Binary Classification – Sdca Logistic Regression

Write C# program to predict Sentiment

1. Question and Data
2. Create project
3. Add NuGet packages
4. Add using name space
5. Create data set input/output scheme
6. Set data set path
7. Load data

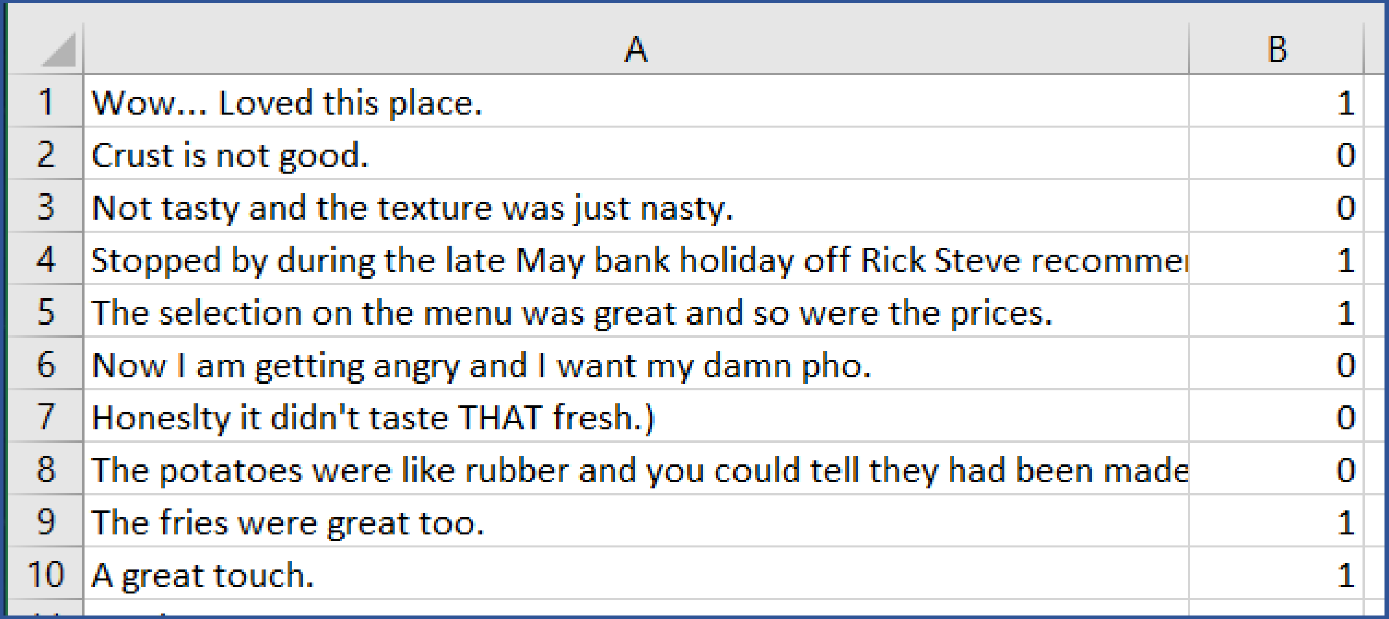
8. Add algorithm
9. Train the model
10. Evaluate the model and show accuracy stats
11. Predict single item
12. Predict multi items
13. Predict multi items from file

Question and Data

Question: The text message has a good or bad meaning?

Data:

- Score is either 1 (for positive) or 0 (for negative)
- The sentences come from three different websites imdb, amazon, yelp .
- For each website, there exist 500 positive and 500 negative sentences.



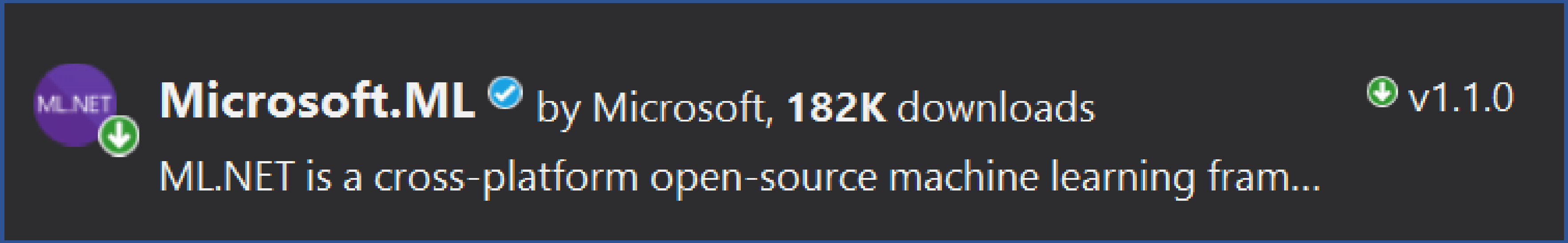
	A	B
1	Wow... Loved this place.	1
2	Crust is not good.	0
3	Not tasty and the texture was just nasty.	0
4	Stopped by during the late May bank holiday off Rick Steve recomme	1
5	The selection on the menu was great and so were the prices.	1
6	Now I am getting angry and I want my damn pho.	0
7	Honeslty it didn't taste THAT fresh.)	0
8	The potatoes were like rubber and you could tell they had been made	0
9	The fries were great too.	1
10	A great touch.	1

Create New Project

Create new .NET CORE console app project name = “Sentiment”

Add NuGet Package

- Microsoft.ML
- Microsoft.ML.Fast



Microsoft.ML by Microsoft, **182K** downloads

v1.1.0

ML.NET is a cross-platform open-source machine learning fram...

Prepare data

Put Data Set in to D:\ml folder

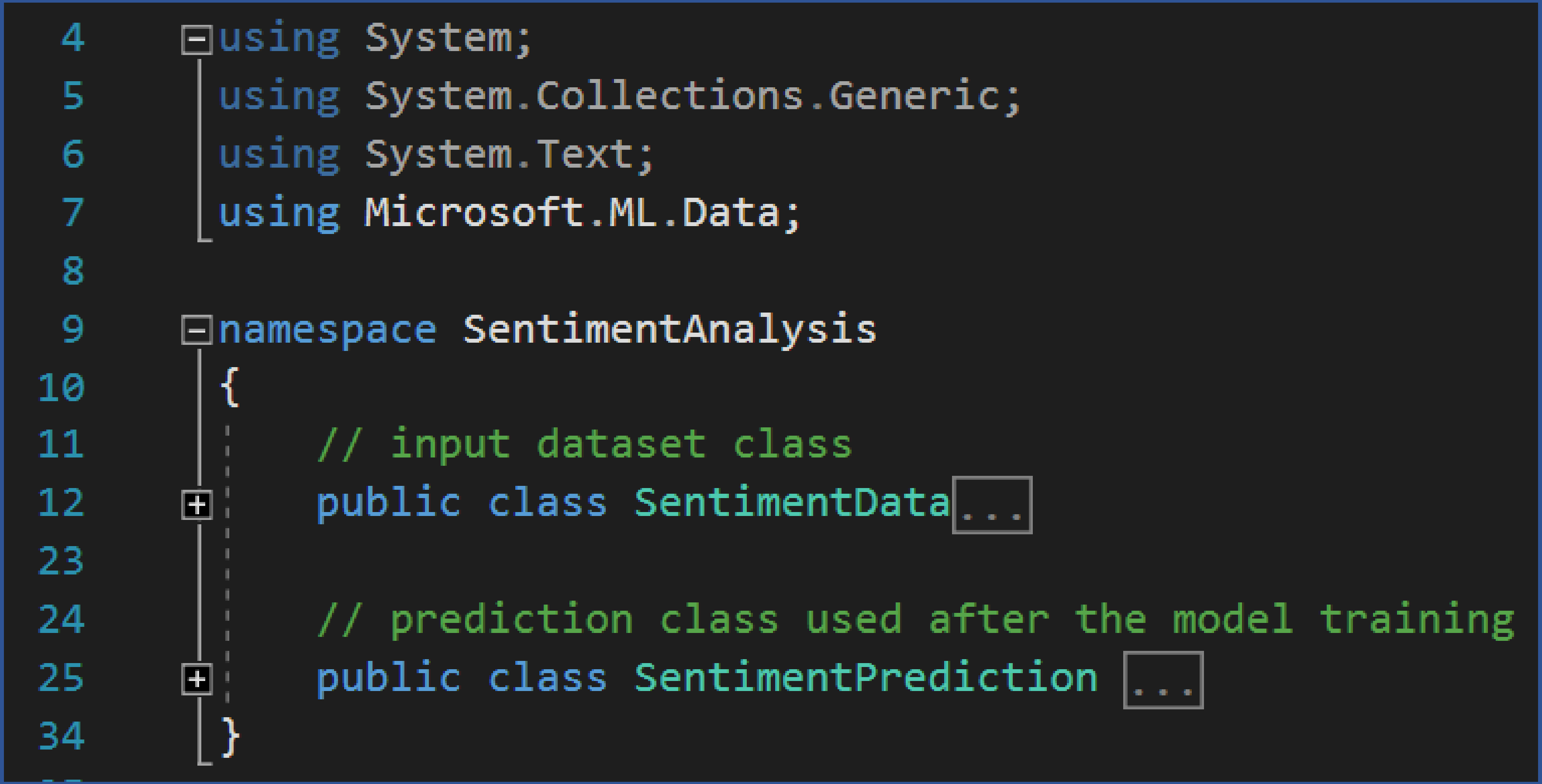
Train Data Set

https://raw.githubusercontent.com/laploy/ML.NET/master/Sentiment/yelp_labelled.txt

Test Data Set

https://raw.githubusercontent.com/laploy/ML.NET/master/Sentiment/BatchSentiments.txt

Create data set input/output scheme



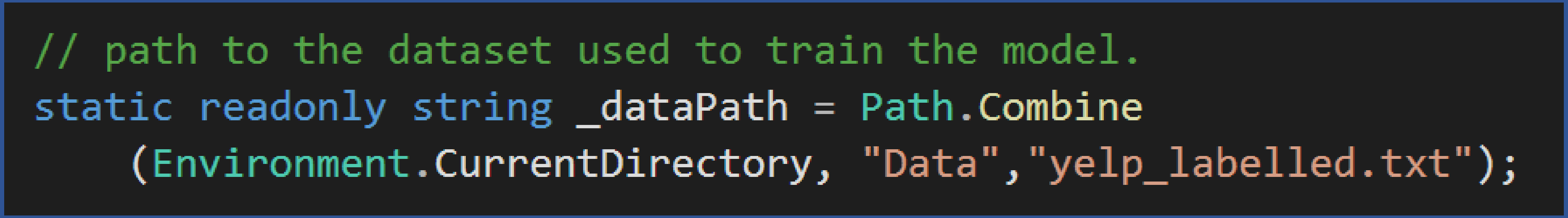
```
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.ML.Data;

namespace SentimentAnalysis
{
    // input dataset class
    public class SentimentData...

    // prediction class used after the model training
    public class SentimentPrediction ...
}
```

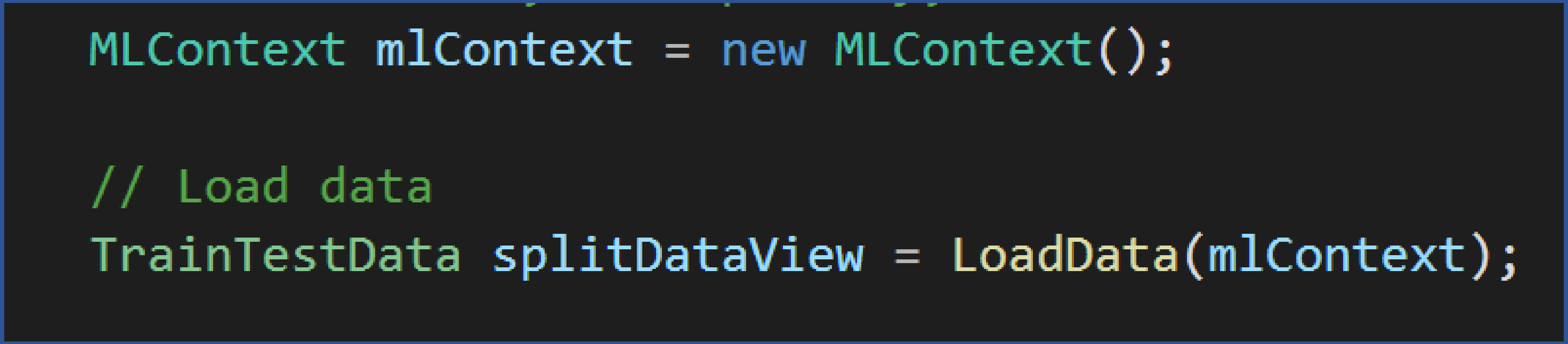
Set data set path

- Add new folder “Data” in to project
- Copy yelp_labelled.txt and BatchSentiments.txt to the new folder
- Set property both file to “Copy if newer”
- Add code to class Program



```
// path to the dataset used to train the model.
static readonly string _dataPath = Path.Combine
    (Environment.CurrentDirectory, "Data","yelp_labelled.txt");
```

Load Data



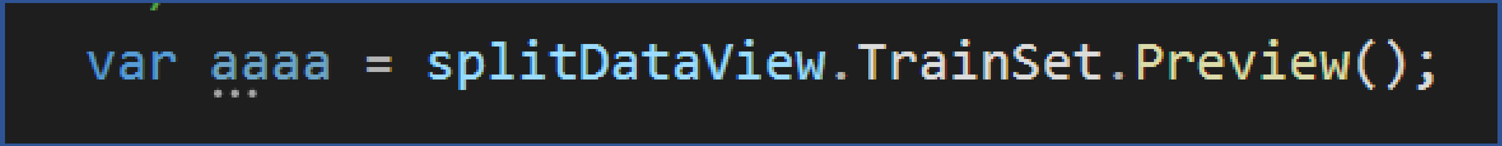
```
MLContext mlContext = new MLContext();

// Load data
TrainTestData splitDataView = LoadData(mlContext);
```

Preview data

Use Preview() method to preview data

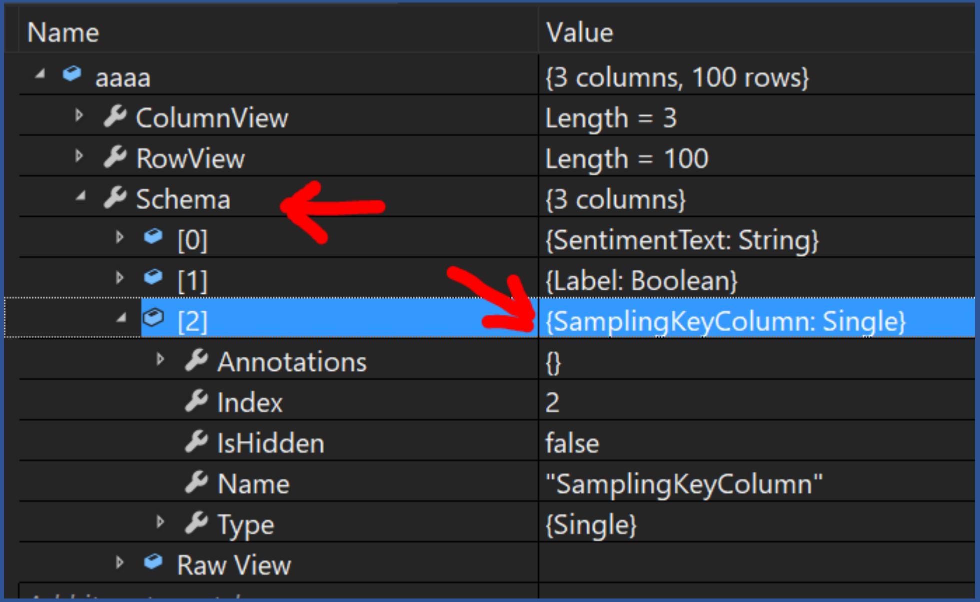
- Add this line



```
var aaaa = splitDataView.TrainSet.Preview();
```

- Set break point after this line
- Run program
- Add variable aaaa to watch

SamplingKeyColumnName

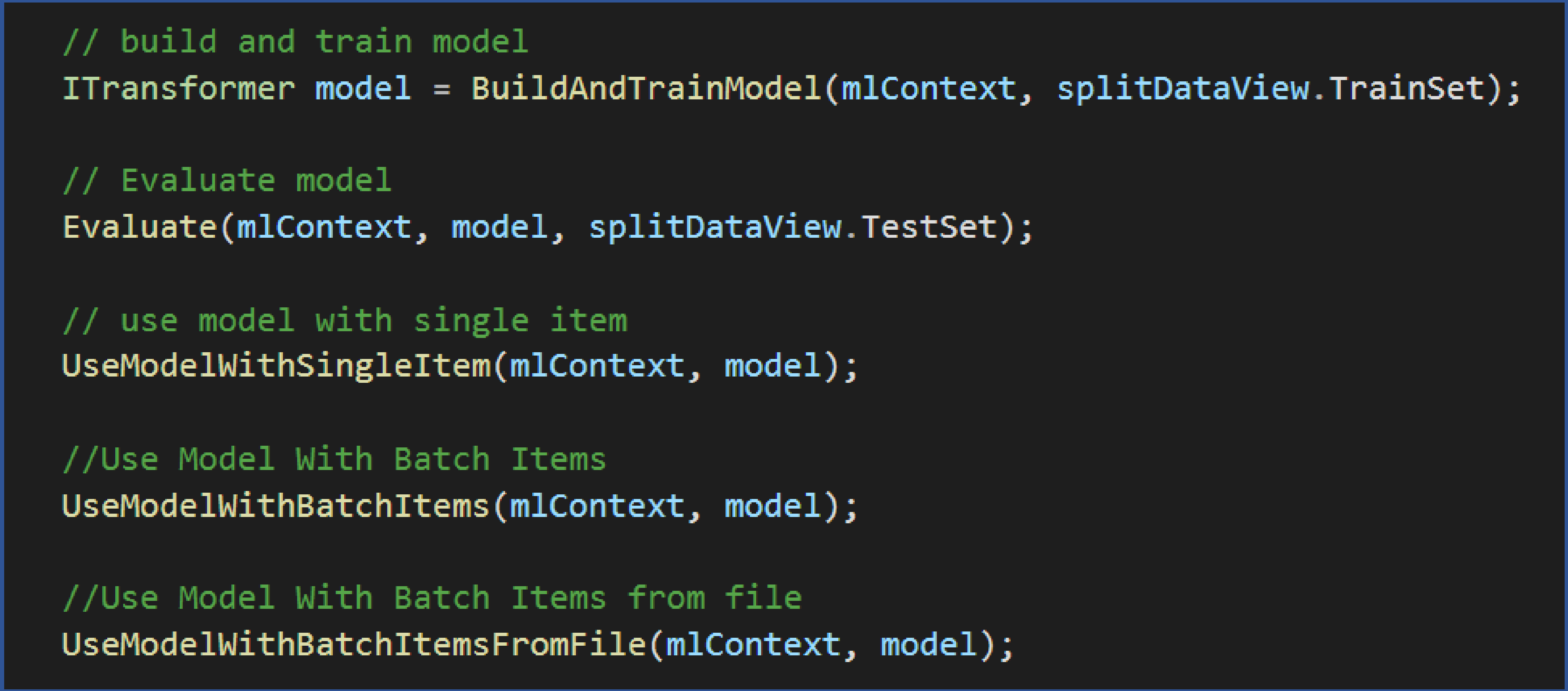


Name	Value
aaaa	{3 columns, 100 rows}
ColumnView	Length = 3
RowView	Length = 100
Schema	{3 columns}
[0]	{SentimentText: String}
[1]	{Label: Boolean}
[2]	{SamplingKeyColumn: Single}
Annotations	{}
Index	2
IsHidden	false
Name	"SamplingKeyColumn"
Type	{Single}
Raw View	

There are 3 columns.

SamplingKeyColumnName: If two rows has the same sampling key column name, they are guaranteed to appear in the same subset (train or test). This can be used to ensure no "label leakage" from the train to the test set.

Build, Evaluate, and Test



```
// build and train model
ITransformer model = BuildAndTrainModel(mlContext, splitDataView.TrainSet);

// Evaluate model
Evaluate(mlContext, model, splitDataView.TestSet);

// use model with single item
UseModelWithSingleItem(mlContext, model);

//Use Model With Batch Items
UseModelWithBatchItems(mlContext, model);

//Use Model With Batch Items from file
UseModelWithBatchItemsFromFile(mlContext, model);
```

Next experiment

GitHub Issue

Multi-class Classification