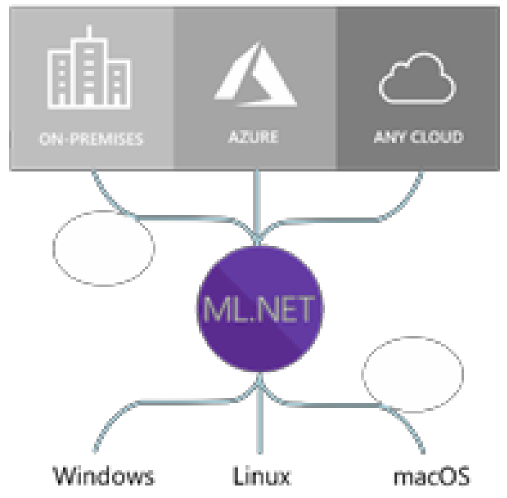
ML.NET

Conf.dr. Cristian Kevorchian

University of Bucharest

ML.NET runs anywhere



Supported Frameworks:
- .NET Core *(Natively)*
- .NET Framework *(Natively)*
- Python with *NimbusML* *(Python bindings)*

Supported processor arch.
- x64
- x86

ML.NET is enterprise-ready battle proven

Microsoft Products
(Using ML.NET internally for > 8 years)

- Bing - Ad Predictions
- Excel - Chart Recommendations
- Power Point - Design Ideas
- Microsoft Defender – Antivirus Threat Protection
- Azure Stream Analytics - Anomaly Detection

Customers
(ML.NET v1 since May 2019)

- Evolution Software – Hazelnut drying time prediction
- Williams Mullen – **Law document classification**
- Sig Parser – **E-mail spam detection**
- Brenmor – Medical patient survey classification
- endjin – Newsletter article classification

+ more

Machine Learning complexity?

Supervised ML (Infers label)

Linear Discriminant Analysis

Structured prediction

Regression

Naïve Bayes

Linear regression

Logistic regression

Decision Trees

Binary Classification

Multi-class Classification

k-nearest neighbor

Neural Networks

(MultiLayer Perception, etc.)

Support Vector Machines

Unsupervised ML (Infers structure)

Clustering

(K-means
Mixture models
Hierarchical clustering)

Topic Modeling

Dimensionality Reduction

Spike detection

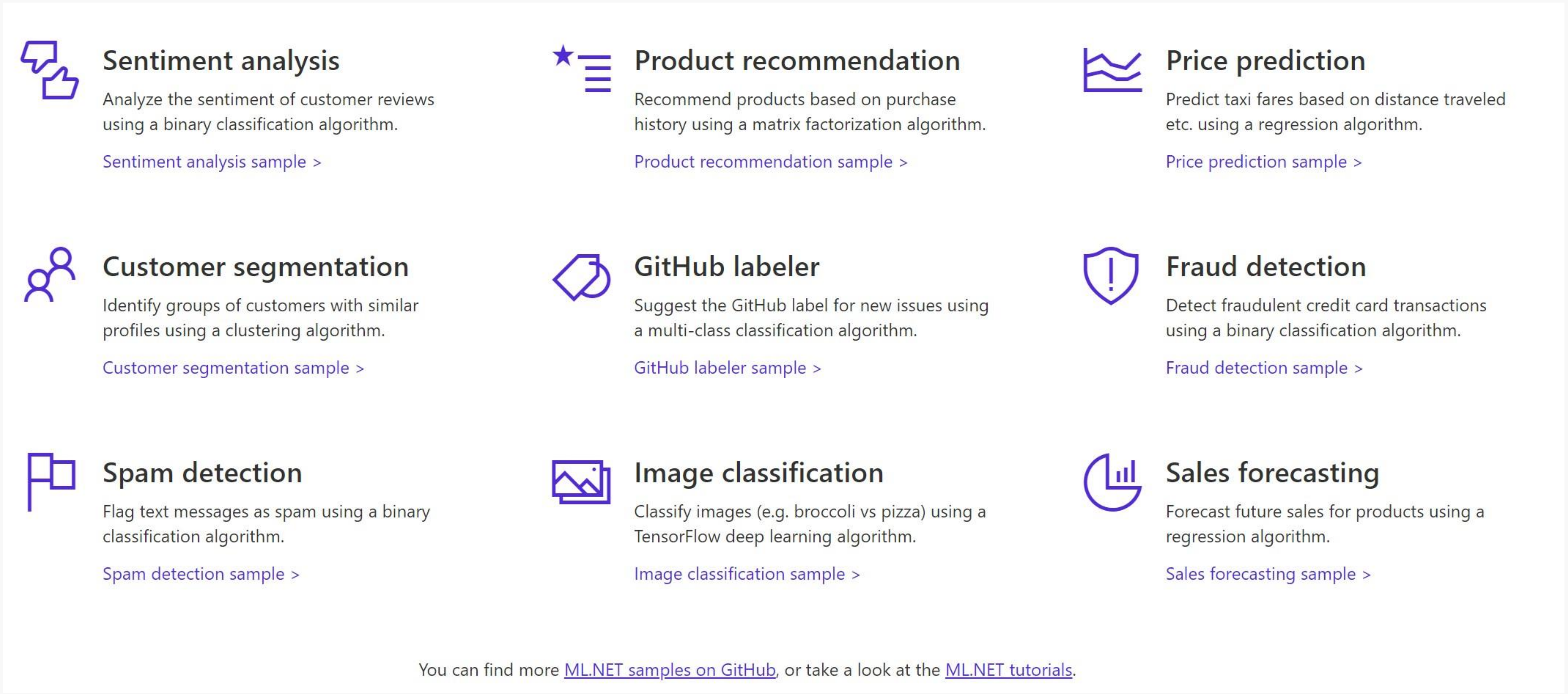
Latent variable models

Topic modeling

Neural Networks

(Autoencoders,
Self-organizing maps, etc.)

A few things you can do with ML.NET …



Sentiment analysis

Analyze the sentiment of customer reviews using a binary classification algorithm.

Sentiment analysis sample >

Product recommendation

Recommend products based on purchase history using a matrix factorization algorithm.

Product recommendation sample >

Price prediction

Predict taxi fares based on distance traveled etc. using a regression algorithm.

Price prediction sample >

Customer segmentation

Identify groups of customers with similar profiles using a clustering algorithm.

Customer segmentation sample >

GitHub labeler

Suggest the GitHub label for new issues using a multi-class classification algorithm.

GitHub labeler sample >

Fraud detection

Detect fraudulent credit card transactions using a binary classification algorithm.

Fraud detection sample >

Spam detection

Flag text messages as spam using a binary classification algorithm.

Spam detection sample >

Image classification

Classify images (e.g. broccoli vs pizza) using a TensorFlow deep learning algorithm.

Image classification sample >

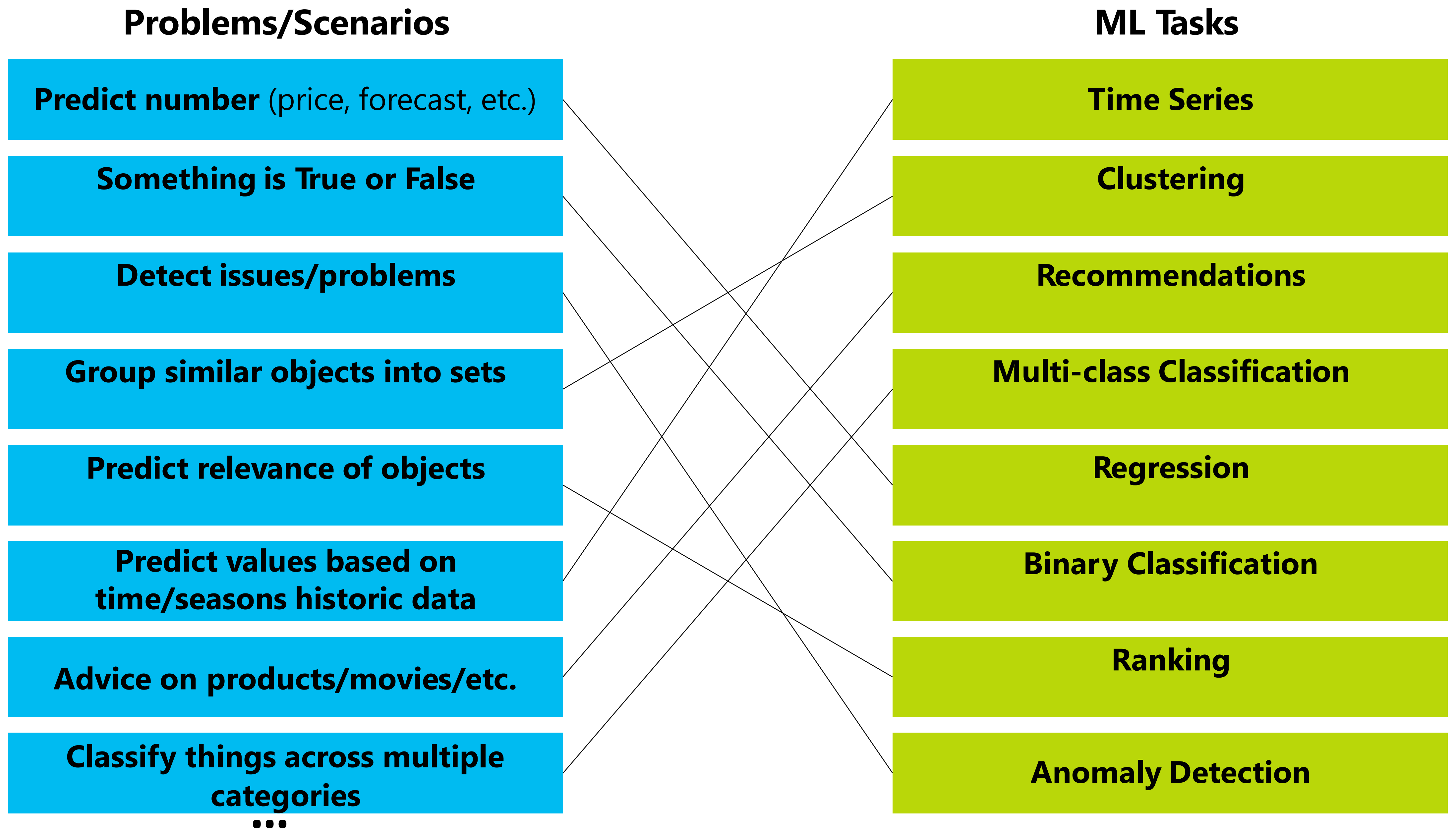
Sales forecasting

Forecast future sales for products using a regression algorithm.

Sales forecasting sample >

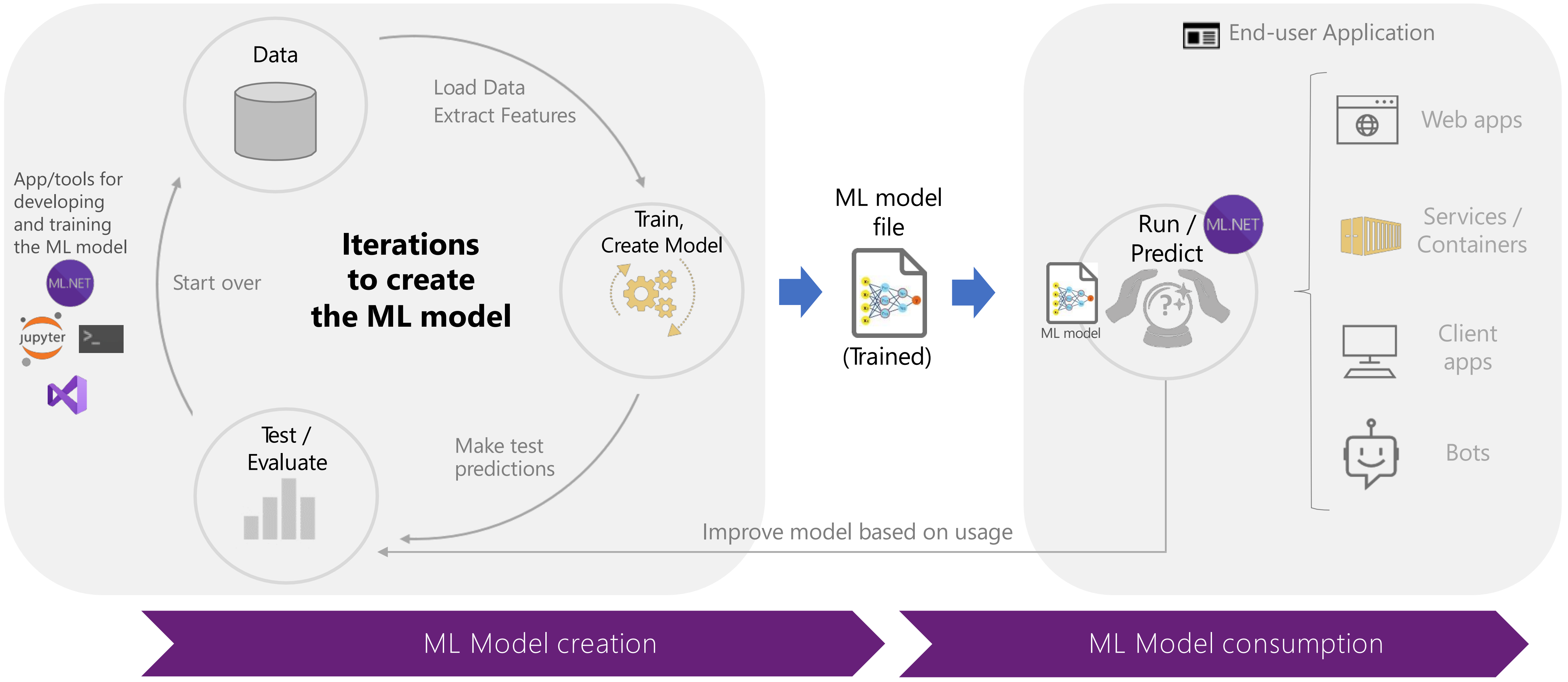
You can find more ML.NET samples on GitHub, or take a look at the ML.NET tutorials.

Mapping from Scenarios/Problems to ML Tasks


Problems/Scenarios
ML Tasks
Predict number (price, forecast, etc.)
Something is True or False
Detect issues/problems
Group similar objects into sets
Predict relevance of objects
Predict values based on time/seasons historic data
Advice on products/movies/etc.
Classify things across multiple categories
…
Time Series
Clustering
Recommendations
Multi-class Classification
Regression
Binary Classification
Ranking
Anomaly Detection

The Machine Learning journey

ML model lifecycle



Three ways to use ML.NET…
…for **training** an ML model

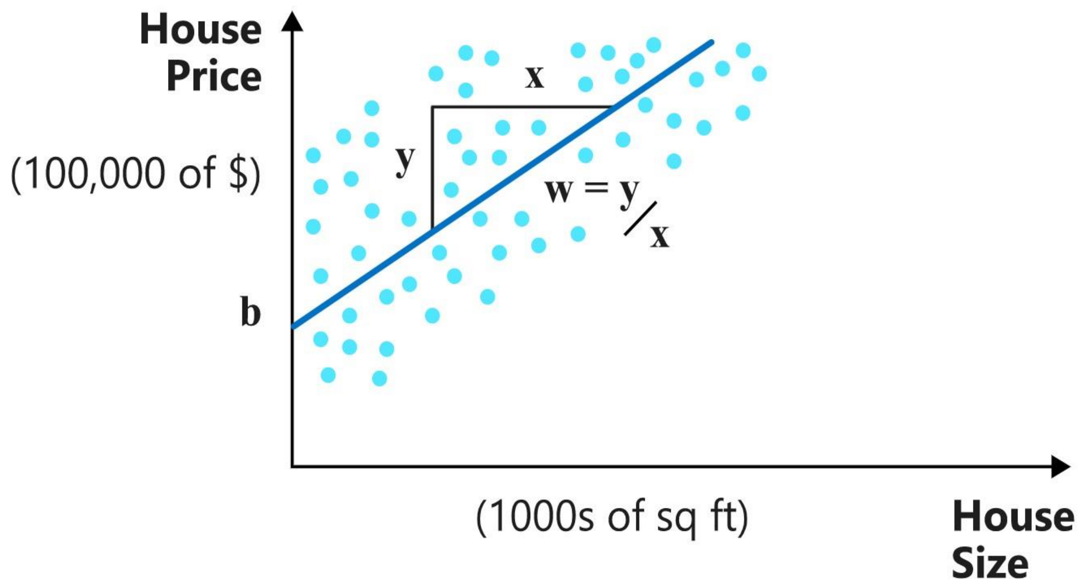
ML.NET
API
(Code)

ML.NET
Model Builder
(Visual Studio UI)

ML.NET
CLI
(Command-Line Interface)

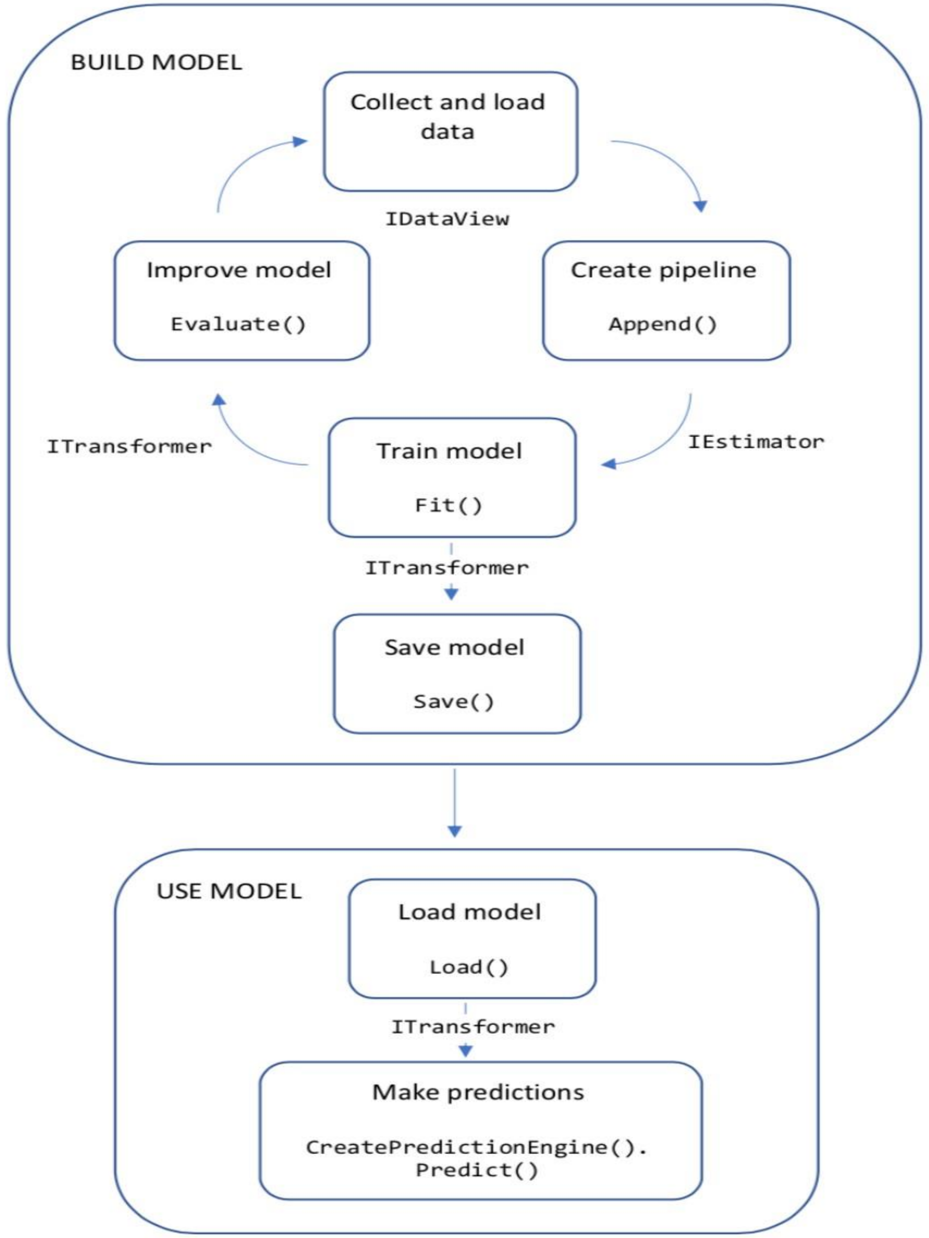
Recap: Machine Learning Models

- ML.NET model is an object that contains transformations to perform on your input data to arrive at the predicted output



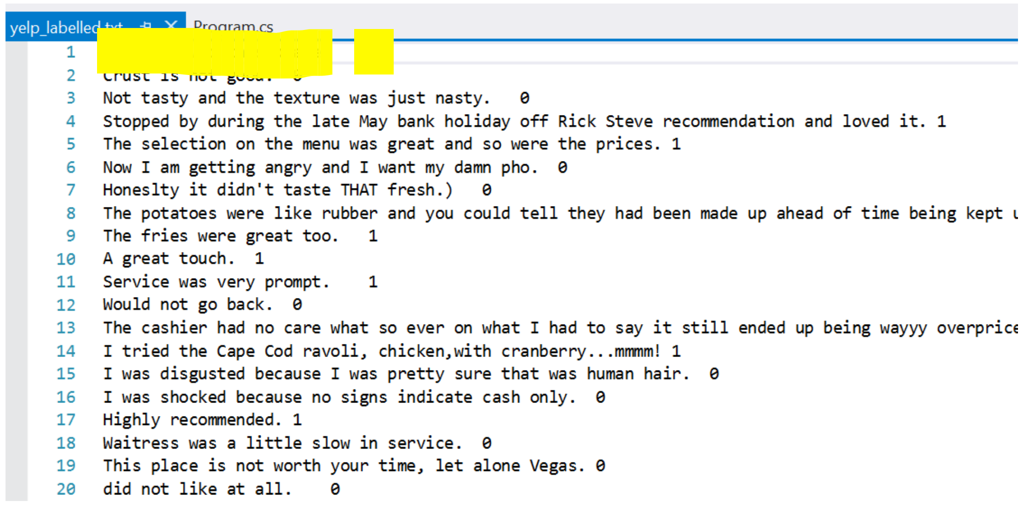
- Most basic model is two-dimensional linear regression
- **Label** is the output, in this case Price
- **Features** are the input to the model that we will process, in this case there is only one, Size

ML.NET API Pattern

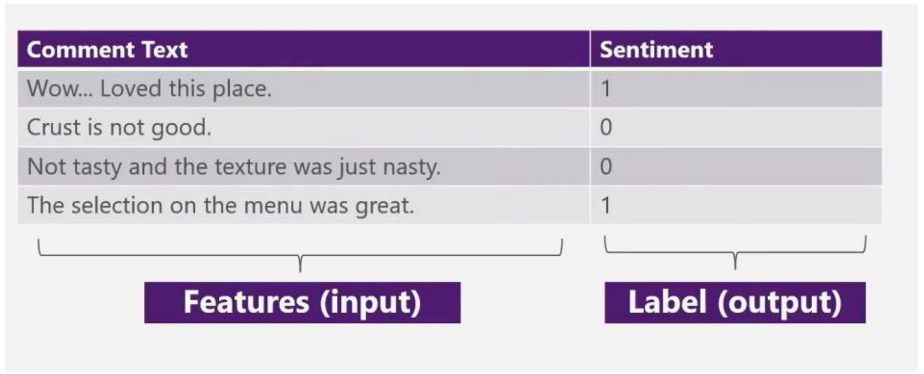


```
//Step 1. Create a ML Context
var ctx = new MLContext();

//Step 2. Read in the input data for model training
IDataView dataReader = ctx.Data
    .LoadFromTextFile<MyInput>(dataPath, hasHeader: true);

//Step 3. Build your estimator
IEstimator<ITransformer> est = ctx.Transforms.Text
    .FeaturizeText("Features", nameof(SentimentIssue.Text))
    .Append(ctx.BinaryClassification.Trainers
        .LbfgsLogisticRegression("Label", "Features"));

//Step 4. Train your Model
ITransformer trainedModel = est.Fit(dataReader);

//Step 5. Make predictions using your model
var predictionEngine = ctx.Model
    .CreatePredictionEngine<MyInput, MyOutput>(trainedModel);

var sampleStatement = new MyInput { Text = "This is a horrible movie" };

var prediction = predictionEngine.Predict(sampleStatement);
```

Example - Analyse Sentiment



```
[redacted]
Crust is not good.  0
Not tasty and the texture was just nasty.   0
Stopped by during the late May bank holiday off Rick Steve recommendation and loved it. 1
The selection on the menu was great and so were the prices. 1
Now I am getting angry and I want my damn pho.  0
Honeslty it didn't taste THAT fresh.)   0
The potatoes were like rubber and you could tell they had been made up ahead of time being kept u
The fries were great too.   1
A great touch.  1
Service was very prompt.    1
Would not go back.  0
The cashier had no care what so ever on what I had to say it still ended up being wayyy overprice
I tried the Cape Cod ravoli, chicken,with cranberry...mmmm! 1
I was disgusted because I was pretty sure that was human hair.  0
I was shocked because no signs indicate cash only.  0
Highly recommended. 1
Waitress was a little slow in service.  0
This place is not worth your time, let alone Vegas. 0
did not like at all.    0
```



Comment Text	Sentiment
Wow... Loved this place.	1
Crust is not good.	0
Not tasty and the texture was just nasty.	0
The selection on the menu was great.	1

```
IDataView dataView = ctx.Data.LoadFromTextFile<SentimentData>(_dataPath, hasHeader: false);
TrainTestData splitDataView = ctx.Data.TrainTestSplit(dataView, testFraction: 0.2);  // create a split set
```

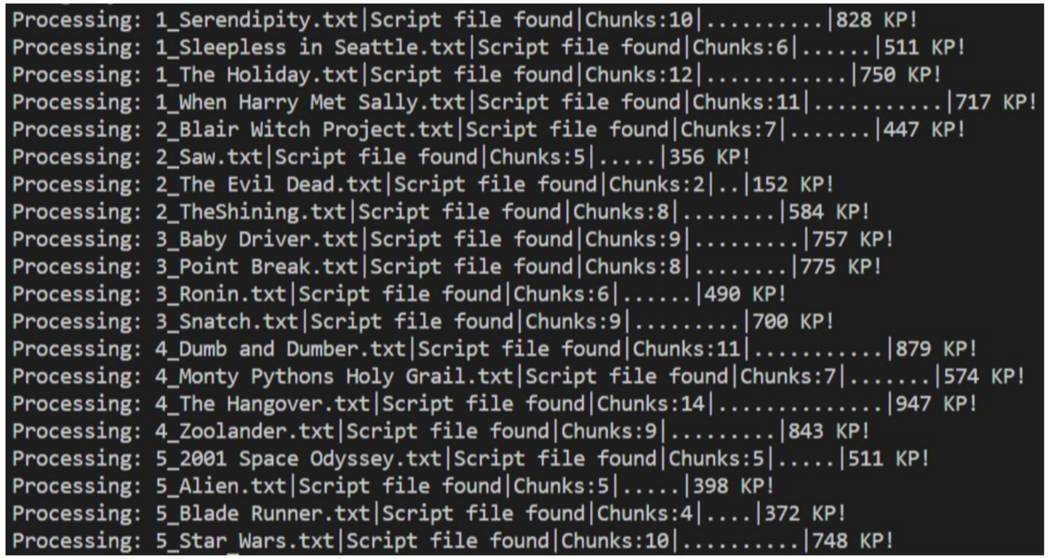
```
1 reference
public class SentimentData
{
    [LoadColumn(0)]
    public string SentimentText;

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

0 references
public class SentimentPrediction : SentimentData
{

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

    0 references
    public float Probability { get; set; }

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

The [redacted] method in the previous code converts the text column (`SentimentText`) into a numeric key type `Features` column used by the machine learning algorithm and adds it as a new dataset column:

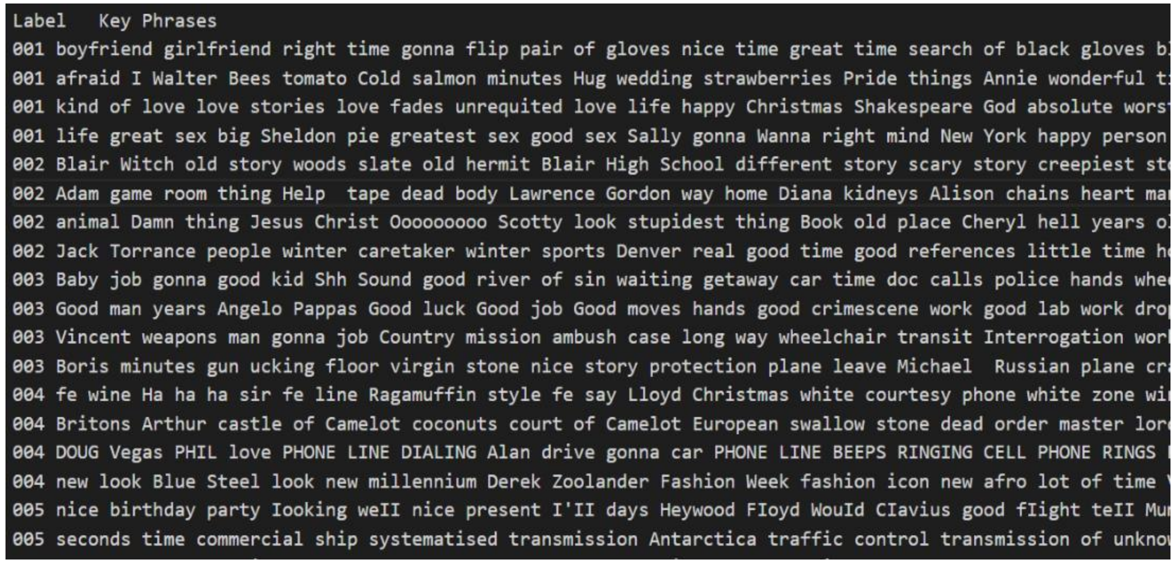
SentimentText	Sentiment	Features
Waitress was a little slow in service.	0	[0.76, 0.65, 0.44, …]
Crust is not good.	0	[0.98, 0.43, 0.54, …]
Wow... Loved this place.	1	[0.35, 0.73, 0.46, …]
Service was very prompt.	1	[0.39, 0, 0.75, …]

```
IEstimator<ITransformer> est = ctx.Transforms.Text[redacted]outputColumnName: "Features", inputColumnName: nameof(SentimentData.SentimentText))
                .Append(ctx.BinaryClassification.T[redacted]sticRegression(labelColumnName: "Label", featureColumnName: "Features"));
```

Step #1 use Azure Text Analytics to extract key phrases, named entities, etc from movie scripts

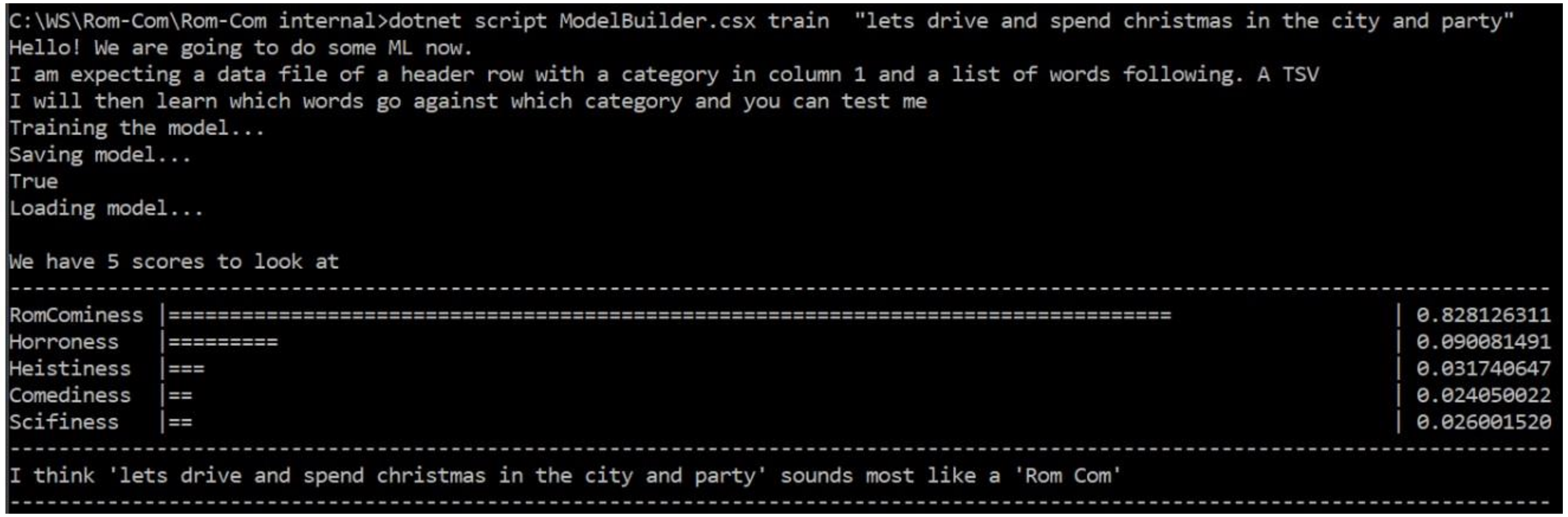


```
Processing: 1_Serendipity.txt|Script file found|Chunks:10|..........|828 KP!
Processing: 1_Sleepless in Seattle.txt|Script file found|Chunks:6|......|511 KP!
Processing: 1_The Holiday.txt|Script file found|Chunks:12|............|750 KP!
Processing: 1_When Harry Met Sally.txt|Script file found|Chunks:11|...........|717 KP!
Processing: 2_Blair Witch Project.txt|Script file found|Chunks:7|.......|447 KP!
Processing: 2_Saw.txt|Script file found|Chunks:5|.....|356 KP!
Processing: 2_The Evil Dead.txt|Script file found|Chunks:2|..|152 KP!
Processing: 2_TheShining.txt|Script file found|Chunks:8|........|584 KP!
Processing: 3_Baby Driver.txt|Script file found|Chunks:9|.........|757 KP!
Processing: 3_Point Break.txt|Script file found|Chunks:8|........|775 KP!
Processing: 3_Ronin.txt|Script file found|Chunks:6|......|490 KP!
Processing: 3_Snatch.txt|Script file found|Chunks:9|.........|700 KP!
Processing: 4_Dumb and Dumber.txt|Script file found|Chunks:11|...........|879 KP!
Processing: 4_Monty Pythons Holy Grail.txt|Script file found|Chunks:7|.......|574 KP!
Processing: 4_The Hangover.txt|Script file found|Chunks:14|..............|947 KP!
Processing: 4_Zoolander.txt|Script file found|Chunks:9|.........|843 KP!
Processing: 5_2001 Space Odyssey.txt|Script file found|Chunks:5|.....|511 KP!
Processing: 5_Alien.txt|Script file found|Chunks:5|.....|398 KP!
Processing: 5_Blade Runner.txt|Script file found|Chunks:4|....|372 KP!
Processing: 5_Star Wars.txt|Script file found|Chunks:10|..........|748 KP!
```



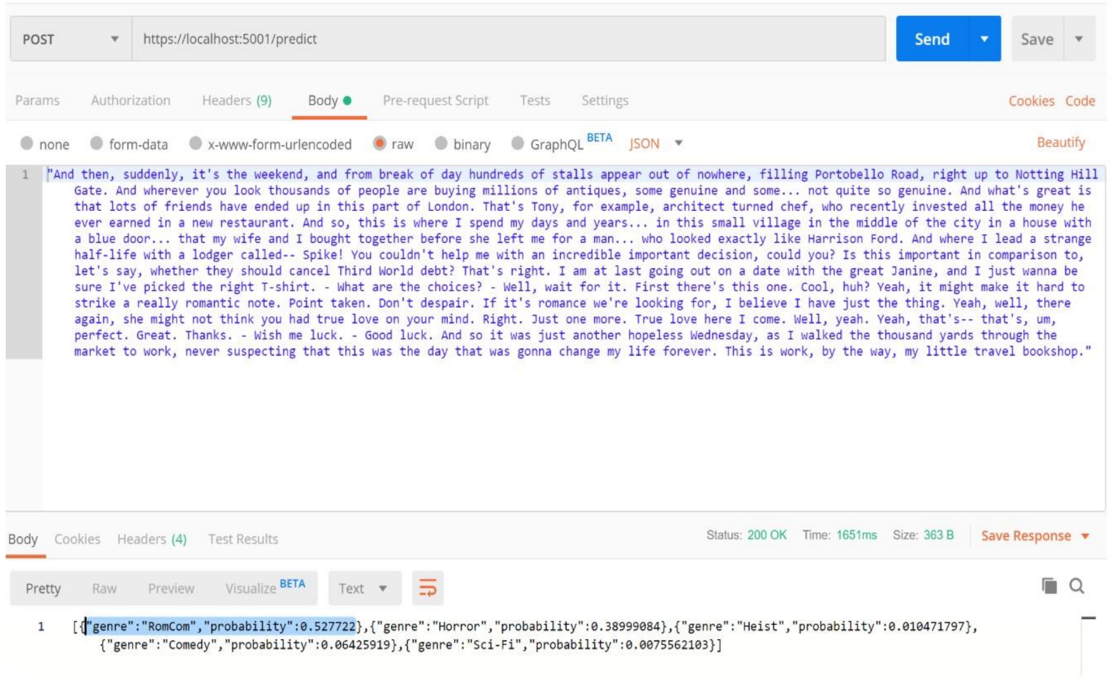
```
Label   Key Phrases
001 boyfriend girlfriend right time gonna flip pair of gloves nice time great time search of black gloves bl
001 afraid I Walter Bees tomato Cold salmon minutes Hug wedding strawberries Pride things Annie wonderful t
001 kind of love love stories love fades unrequited love life happy Christmas Shakespeare God absolute worst
001 life great sex big Sheldon pie greatest sex good sex Sally gonna Wanna right mind New York happy person
002 Blair Witch old story woods slate old hermit Blair High School different story scary story creepiest st
002 Adam game room thing Help  tape dead body Lawrence Gordon way home Diana kidneys Alison chains heart ma
002 animal Damn thing Jesus Christ Ooooooooo Scotty look stupidest thing Book old place Cheryl hell years o
002 Jack Torrance people winter caretaker winter sports Denver real good time good references little time h
003 Baby job gonna good kid Shh Sound good river of sin waiting getaway car time doc calls police hands whe
003 Good man years Angelo Pappas Good luck Good job Good moves hands good crimescene work good lab work dro
003 Vincent weapons man gonna job Country mission ambush case long way wheelchair transit Interrogation wor
003 Boris minutes gun ucking floor virgin stone nice story protection plane leave Michael  Russian plane cr
004 fe wine Ha ha ha sir fe line Ragamuffin style fe say Lloyd Christmas white courtesy phone white zone wi
004 Britons Arthur castle of Camelot coconuts court of Camelot European swallow stone dead order master lor
004 DOUG Vegas PHIL love PHONE LINE DIALING Alan drive gonna car PHONE LINE BEEPS RINGING CELL PHONE RINGS
004 new look Blue Steel look new millennium Derek Zoolander Fashion Week fashion icon new afro lot of time
005 nice birthday party Iooking weII nice present I'II days Heywood FIoyd WouId CIavius good fIight teII Mu
005 seconds time commercial ship systematised transmission Antarctica traffic control transmission of unkno
```

Step #2 Train a multi-classifier ML.NET model



```
C:\WS\Rom-Com\Rom-Com internal>dotnet script ModelBuilder.csx train  "lets drive and spend christmas in the city and party"
Hello! We are going to do some ML now.
I am expecting a data file of a header row with a category in column 1 and a list of words following. A TSV
I will then learn which words go against which category and you can test me
Training the model...
Saving model...
True
Loading model...

We have 5 scores to look at
------------------------------------------------------------------------------------------------------------------------------
RomCominess |==================================================================================                  | 0.828126311
Horroness   |=========                                                                                           | 0.090081491
Heistiness  |===                                                                                                 | 0.031740647
Comediness  |==                                                                                                  | 0.024050022
Scifiness   |==                                                                                                  | 0.026001520
------------------------------------------------------------------------------------------------------------------------------
I think 'lets drive and spend christmas in the city and party' sounds most like a 'Rom Com'
------------------------------------------------------------------------------------------------------------------------------
```

Step #3 Deploy behind dotnet core WebAPI

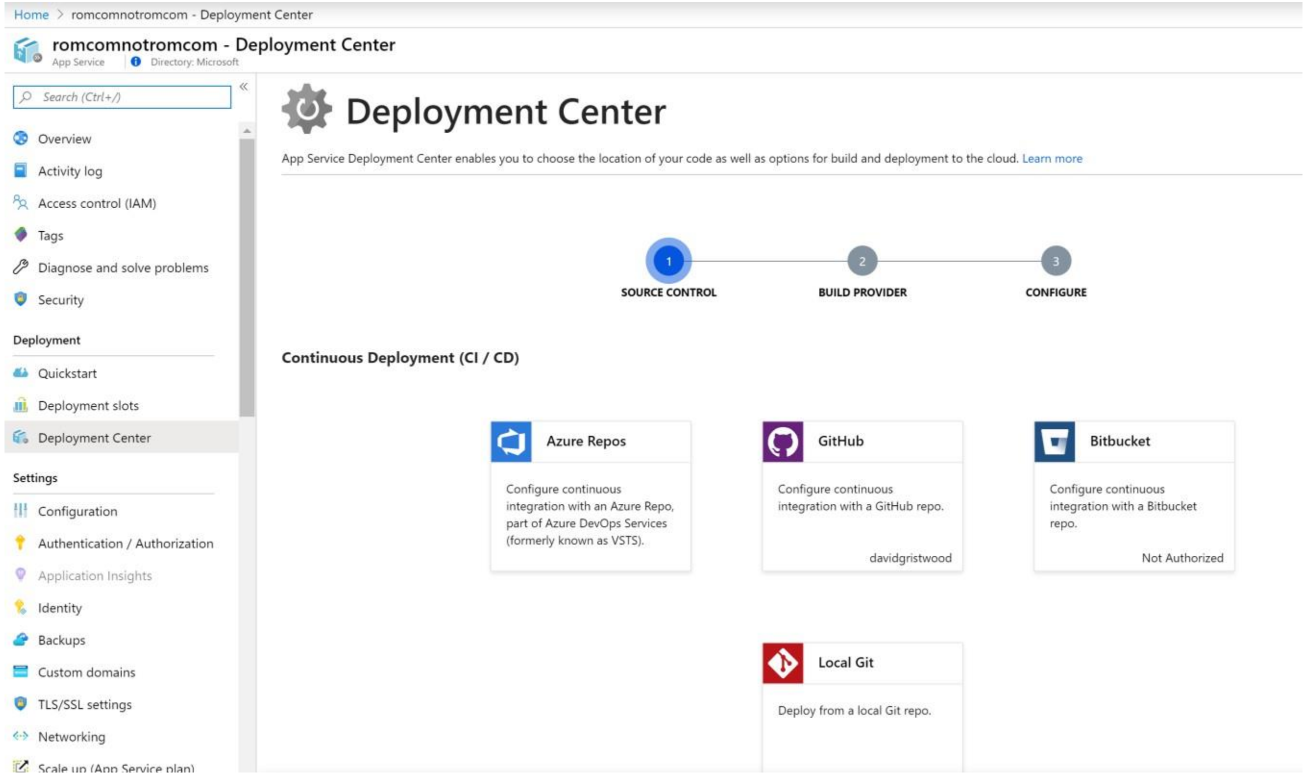


POST https://localhost:5001/predict

```
"And then, suddenly, it's the weekend, and from break of day hundreds of stalls appear out of nowhere, filling Portobello Road, right up to Notting Hill
    Gate. And wherever you look thousands of people are buying millions of antiques, some genuine and some... not quite so genuine. And what's great is
    that lots of friends have ended up in this part of London. That's Tony, for example, architect turned chef, who recently invested all the money he
    ever earned in a new restaurant. And so, this is where I spend my days and years... in this small village in the middle of the city in a house with
    a blue door... that my wife and I bought together before she left me for a man... who looked exactly like Harrison Ford. And where I lead a strange
    half-life with a lodger called-- Spike! You couldn't help me with an incredible important decision, could you? Is this important in comparison to,
    let's say, whether they should cancel Third World debt? That's right. I am at last going out on a date with the great Janine, and I just wanna be
    sure I've picked the right T-shirt. - What are the choices? - Well, wait for it. First there's this one. Cool, huh? Yeah, it might make it hard to
    strike a really romantic note. Point taken. Don't despair. If it's romance we're looking for, I believe I have just the thing. Yeah, well, there
    again, she might not think you had true love on your mind. Right. Just one more. True love here I come. Well, yeah. Yeah, that's-- that's, um,
    perfect. Great. Thanks. - Wish me luck. - Good luck. And so it was just another hopeless Wednesday, as I walked the thousand yards through the
    market to work, never suspecting that this was the day that was gonna change my life forever. This is work, by the way, my little travel bookshop."
```

Status: 200 OK Time: 1651ms Size: 363 B

```
[{"genre":"RomCom","probability":0.527722},{"genre":"Horror","probability":0.38999084},{"genre":"Heist","probability":0.010471797},
  {"genre":"Comedy","probability":0.06425919},{"genre":"Sci-Fi","probability":0.0075562103}]
```



App Service Deployment Center enables you to choose the location of your code as well as options for build and deployment to the cloud. Learn more

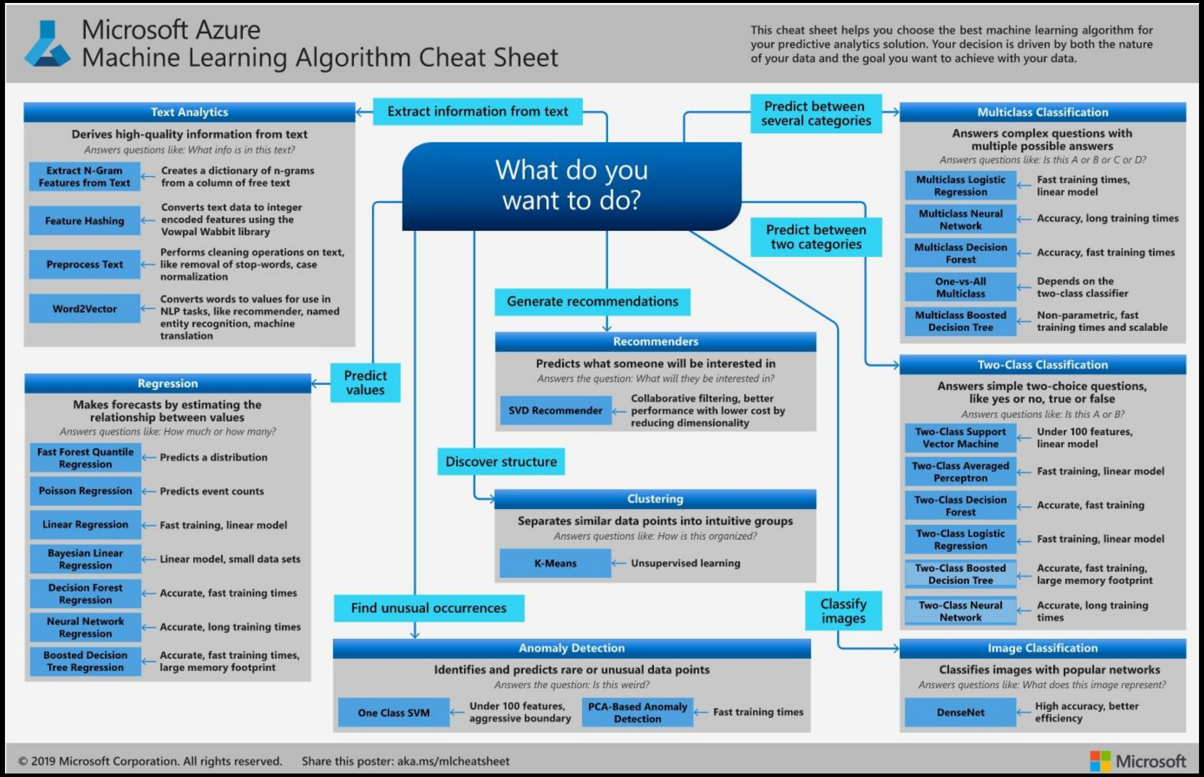
Azure Repos: Configure continuous integration with an Azure Repo, part of Azure DevOps Services (formerly known as VSTS).

GitHub: Configure continuous integration with a GitHub repo. davidgristwood

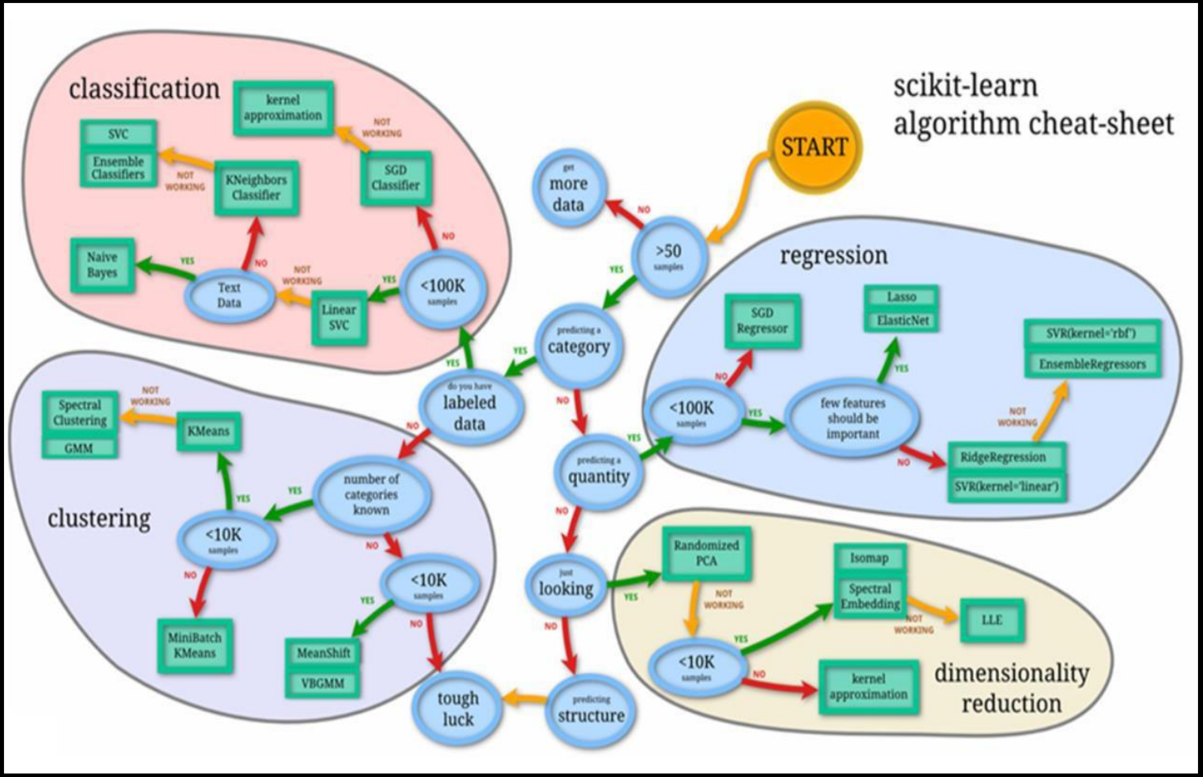
Bitbucket: Configure continuous integration with a Bitbucket repo. Not Authorized

Local Git: Deploy from a local Git repo.

The need for AutoML



Microsoft Azure
Machine Learning Algorithm Cheat Sheet
This cheat sheet helps you choose the best machine learning algorithm for your predictive analytics solution. Your decision is driven by both the nature of your data and the goal you want to achieve with your data.
What do you want to do?
Extract information from text
Text Analytics
Derives high-quality information from text
Answers questions like: What info is in this text?
Extract N-Gram Features from Text
Creates a dictionary of n-grams from a column of free text
Feature Hashing
Converts text data to integer encoded features using the Vowpal Wabbit library
Preprocess Text
Performs cleaning operations on text, like removal of stop-words, case normalization
Word2Vector
Converts words to values for use in NLP tasks, like recommender, named entity recognition, machine translation
Predict values
Regression
Makes forecasts by estimating the relationship between values
Answers questions like: How much or how many?
Fast Forest Quantile Regression
Predicts a distribution
Poisson Regression
Predicts event counts
Linear Regression
Fast training, linear model
Bayesian Linear Regression
Linear model, small data sets
Decision Forest Regression
Accurate, fast training times
Neural Network Regression
Accurate, long training times
Boosted Decision Tree Regression
Accurate, fast training times, large memory footprint
Generate recommendations
Recommenders
Predicts what someone will be interested in
Answers the question: What will they be interested in?
SVD Recommender
Collaborative filtering, better performance with lower cost by reducing dimensionality
Discover structure
Clustering
Separates similar data points into intuitive groups
Answers questions like: How is this organized?
K-Means
Unsupervised learning
Find unusual occurrences
Anomaly Detection
Identifies and predicts rare or unusual data points
Answers the question: Is this weird?
One Class SVM
Under 100 features, aggressive boundary
PCA-Based Anomaly Detection
Fast training times
Predict between several categories
Multiclass Classification
Answers complex questions with multiple possible answers
Answers questions like: Is this A or B or C or D?
Multiclass Logistic Regression
Fast training times, linear model
Multiclass Neural Network
Accuracy, long training times
Multiclass Decision Forest
Accuracy, fast training times
One-vs-All Multiclass
Depends on the two-class classifier
Multiclass Boosted Decision Tree
Non-parametric, fast training times and scalable
Predict between two categories
Two-Class Classification
Answers simple two-choice questions, like yes or no, true or false
Answers questions like: Is this A or B?
Two-Class Support Vector Machine
Under 100 features, linear model
Two-Class Averaged Perceptron
Fast training, linear model
Two-Class Decision Forest
Accurate, fast training
Two-Class Logistic Regression
Fast training, linear model
Two-Class Boosted Decision Tree
Accurate, fast training, large memory footprint
Two-Class Neural Network
Accurate, long training times
Classify images
Image Classification
Classifies images with popular networks
Answers questions like: What does this image represent?
DenseNet
High accuracy, better efficiency
© 2019 Microsoft Corporation. All rights reserved.
Share this poster: aka.ms/mlcheatsheet
Microsoft



ML.NET AutoML

AutoML performs automatic model selection and training

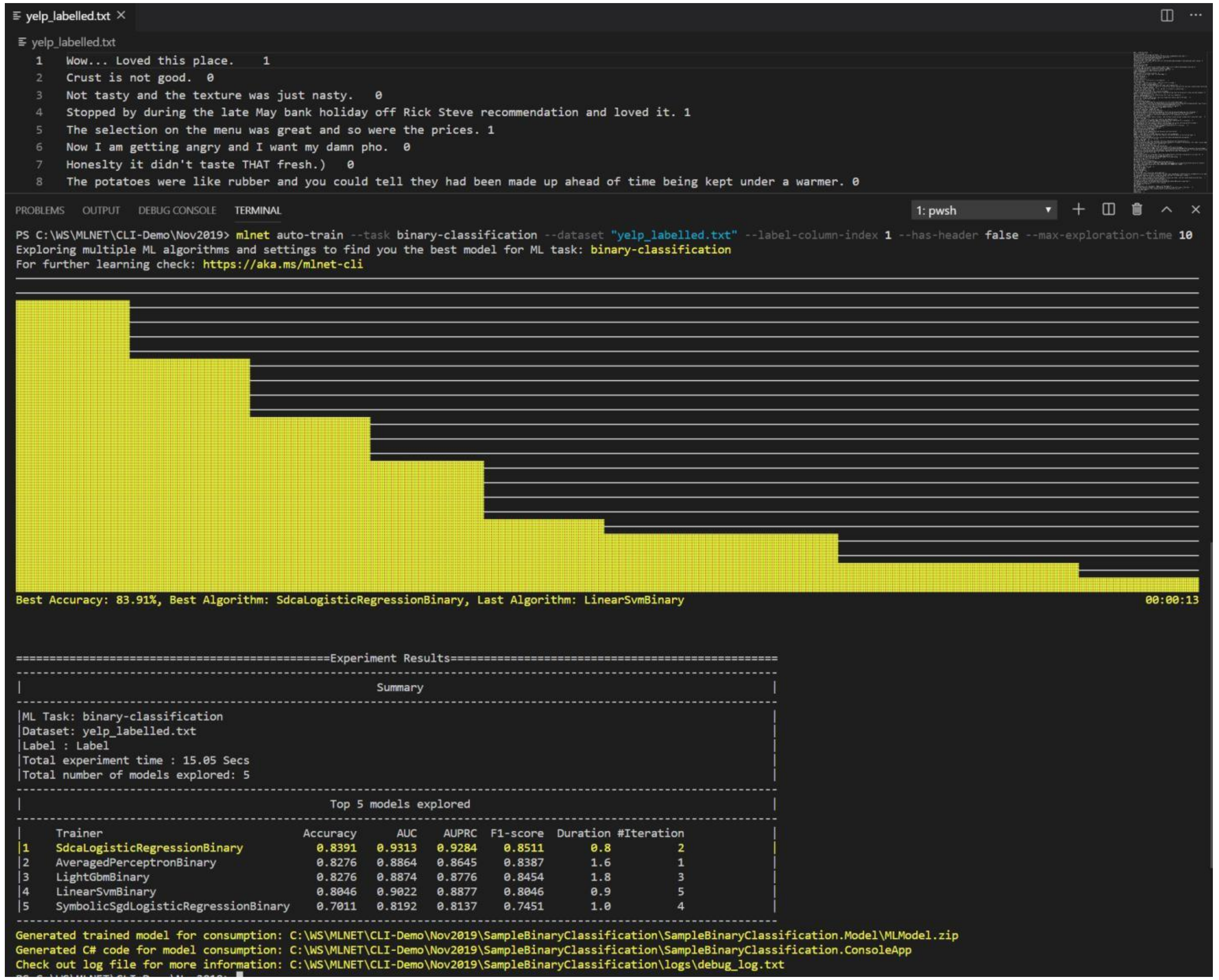
You supply machine learning task and supply a dataset, it outputs:

- a model file that can be loaded into your prediction application
- application code to make predictions
- the source code used for feature selection and model training (to understand the model)

Available in 3 form factors:

- ML.NET CLI
- ML.NET Model Builder
- ML.NET AUTO API

MLNET CLI



```
Wow... Loved this place.	1
Crust is not good.	0
Not tasty and the texture was just nasty.	0
Stopped by during the late May bank holiday off Rick Steve recommendation and loved it. 1
The selection on the menu was great and so were the prices. 1
Now I am getting angry and I want my damn pho.	0
Honeslty it didn't taste THAT fresh.)	0
The potatoes were like rubber and you could tell they had been made up ahead of time being kept under a warmer. 0
```

```
PS C:\WS\MLNET\CLI-Demo\Nov2019> mlnet auto-train --task binary-classification --dataset "yelp_labelled.txt" --label-column-index 1 --has-header false --max-exploration-time 10
Exploring multiple ML algorithms and settings to find you the best model for ML task: binary-classification
For further learning check: https://aka.ms/mlnet-cli

Best Accuracy: 83.91%, Best Algorithm: SdcaLogisticRegressionBinary, Last Algorithm: LinearSvmBinary                00:00:13


==============================================Experiment Results=================================================
-----------------------------------------------------------------------------------------------------------------
|                                                     Summary                                                   |
-----------------------------------------------------------------------------------------------------------------
|ML Task: binary-classification                                                                                 |
|Dataset: yelp_labelled.txt                                                                                     |
|Label : Label                                                                                                  |
|Total experiment time : 15.05 Secs                                                                             |
|Total number of models explored: 5                                                                             |
-----------------------------------------------------------------------------------------------------------------
|                                              Top 5 models explored                                            |
-----------------------------------------------------------------------------------------------------------------
|     Trainer                              Accuracy      AUC    AUPRC  F1-score  Duration #Iteration            |
|1    SdcaLogisticRegressionBinary           0.8391   0.9313   0.9284    0.8511       0.8          2            |
|2    AveragedPerceptronBinary               0.8276   0.8864   0.8645    0.8387       1.6          1            |
|3    LightGbmBinary                         0.8276   0.8874   0.8776    0.8454       1.8          3            |
|4    LinearSvmBinary                        0.8046   0.9022   0.8877    0.8046       0.9          5            |
|5    SymbolicSgdLogisticRegressionBinary    0.7011   0.8192   0.8137    0.7451       1.0          4            |
-----------------------------------------------------------------------------------------------------------------
Generated trained model for consumption: C:\WS\MLNET\CLI-Demo\Nov2019\SampleBinaryClassification\SampleBinaryClassification.Model\MLModel.zip
Generated C# code for model consumption: C:\WS\MLNET\CLI-Demo\Nov2019\SampleBinaryClassification\SampleBinaryClassification.ConsoleApp
Check out log file for more information: C:\WS\MLNET\CLI-Demo\Nov2019\SampleBinaryClassification\logs\debug_log.txt
```

```
mlnet auto-train
--task binary-classification
--dataset "yelp_labelled.txt"
--label-column-index 1
--has-header false
--max-exploration-time 10
```

Current supported ML Tasks

- binary-classification
- multiclass-classification
- Regression

Future:

- recommendation
- anomaly-detection
- clustering

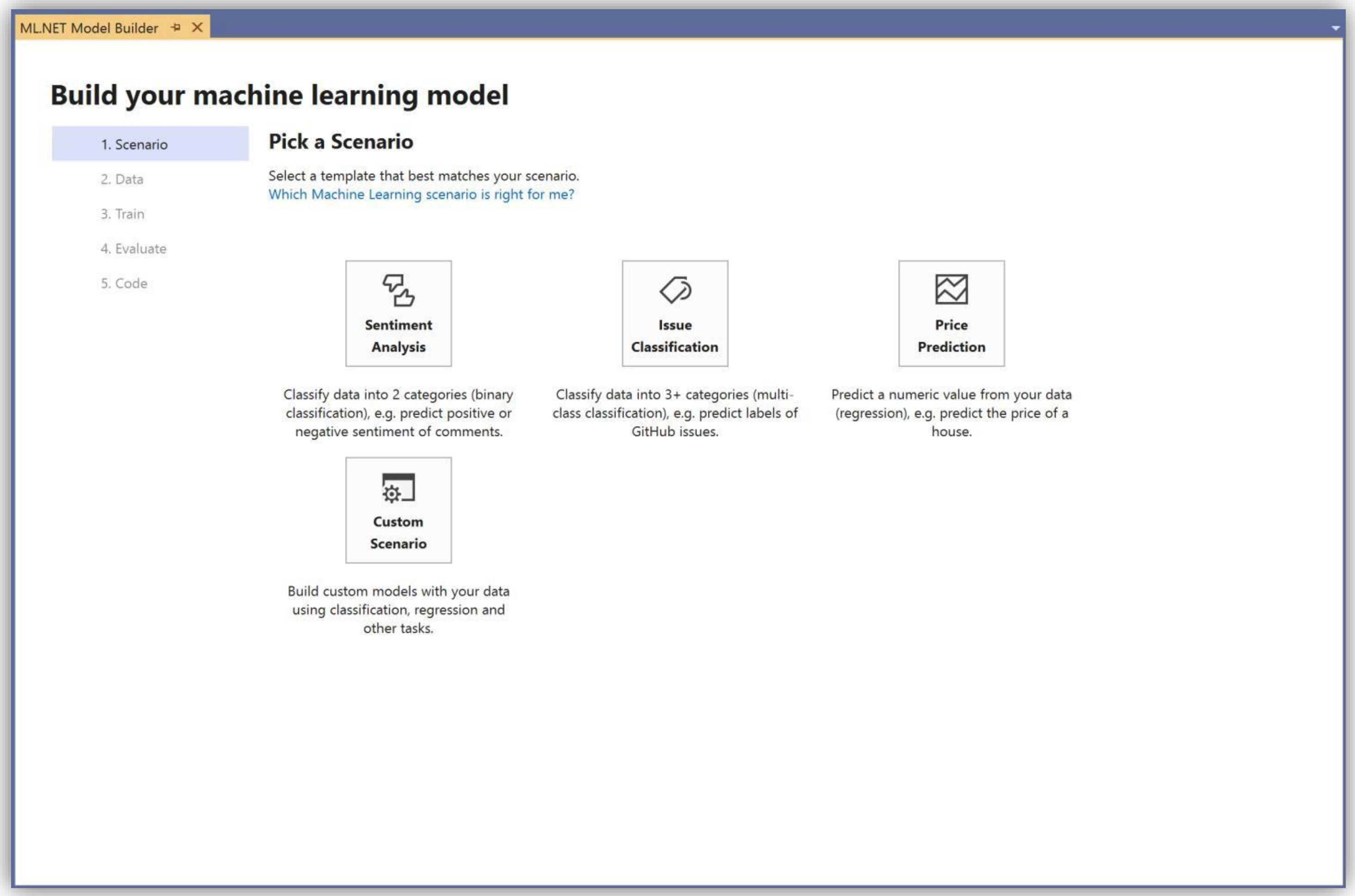
ML.NET Model Builder

Approachable machine learning in Visual Studio

- A simple UI to easily build custom ML models with Automated ML
- Load from files and databases
- Generate code for training and consumption
- Run everything local

Download VS vsix:

http://aka.ms/mlnetmodelbuilder



ML.NET Model Builder

Build your machine learning model

1. Scenario
2. Data
3. Train
4. Evaluate
5. Code

Pick a Scenario

Select a template that best matches your scenario.
Which Machine Learning scenario is right for me?

Sentiment Analysis
Classify data into 2 categories (binary classification), e.g. predict positive or negative sentiment of comments.

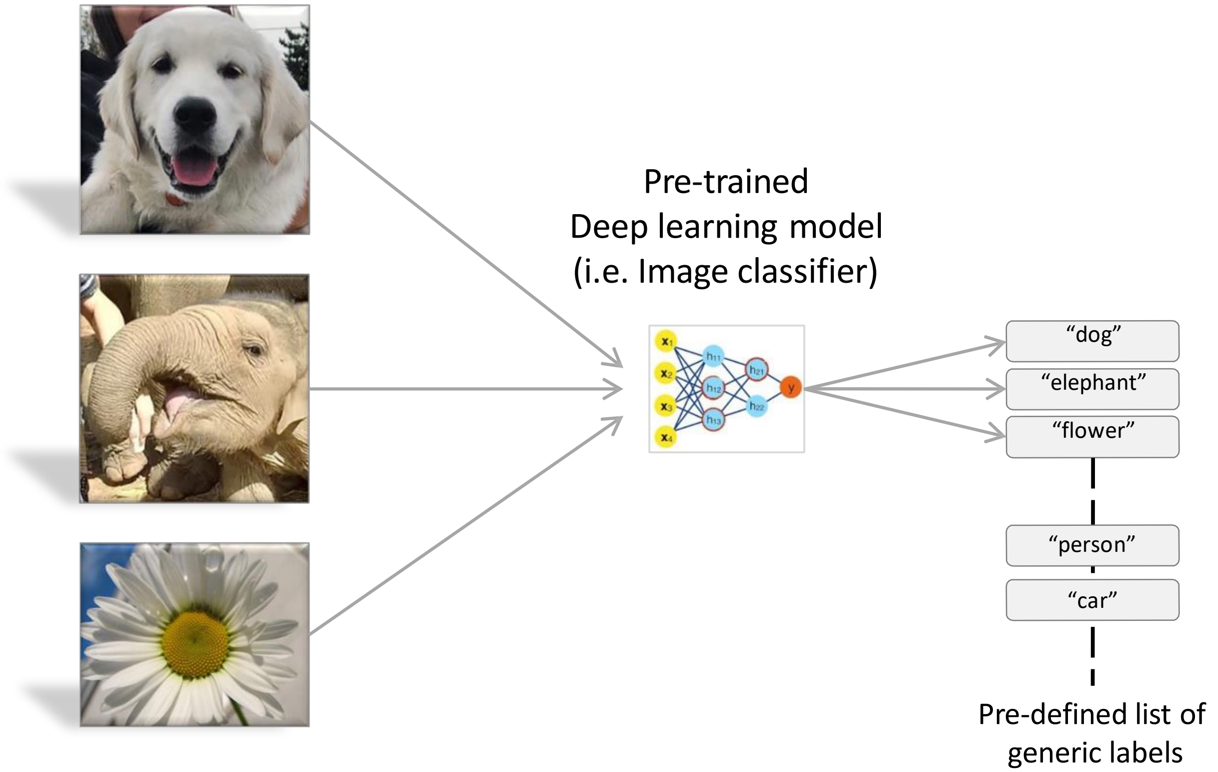
Issue Classification
Classify data into 3+ categories (multi-class classification), e.g. predict labels of GitHub issues.

Price Prediction
Predict a numeric value from your data (regression), e.g. predict the price of a house.

Custom Scenario
Build custom models with your data using classification, regression and other tasks.

Consuming pre-trained deep learning models with ML.NET

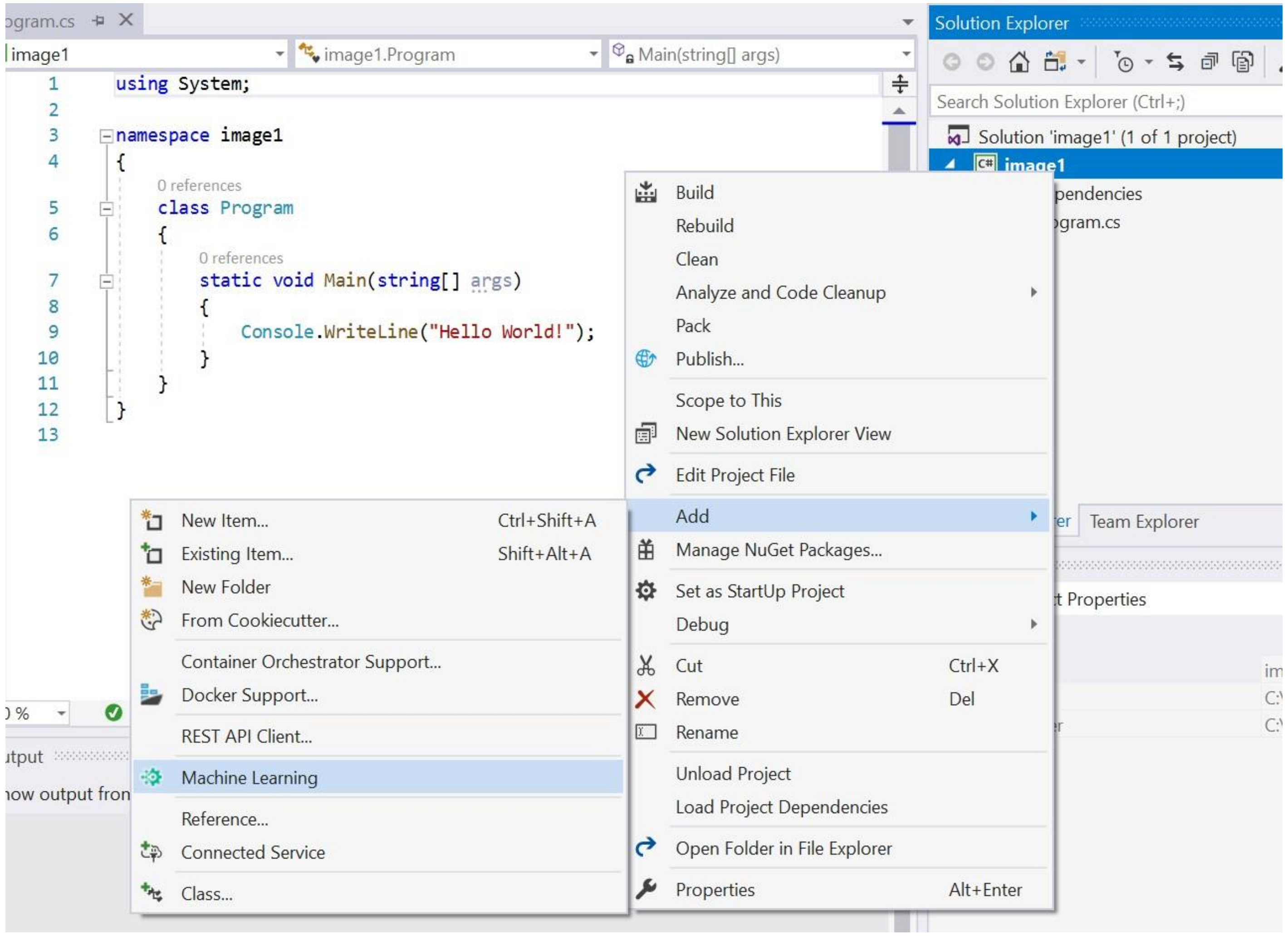
Scenario: **Image classifier** (Consuming the model)



Examples of pre-trained models (Image classifiers):

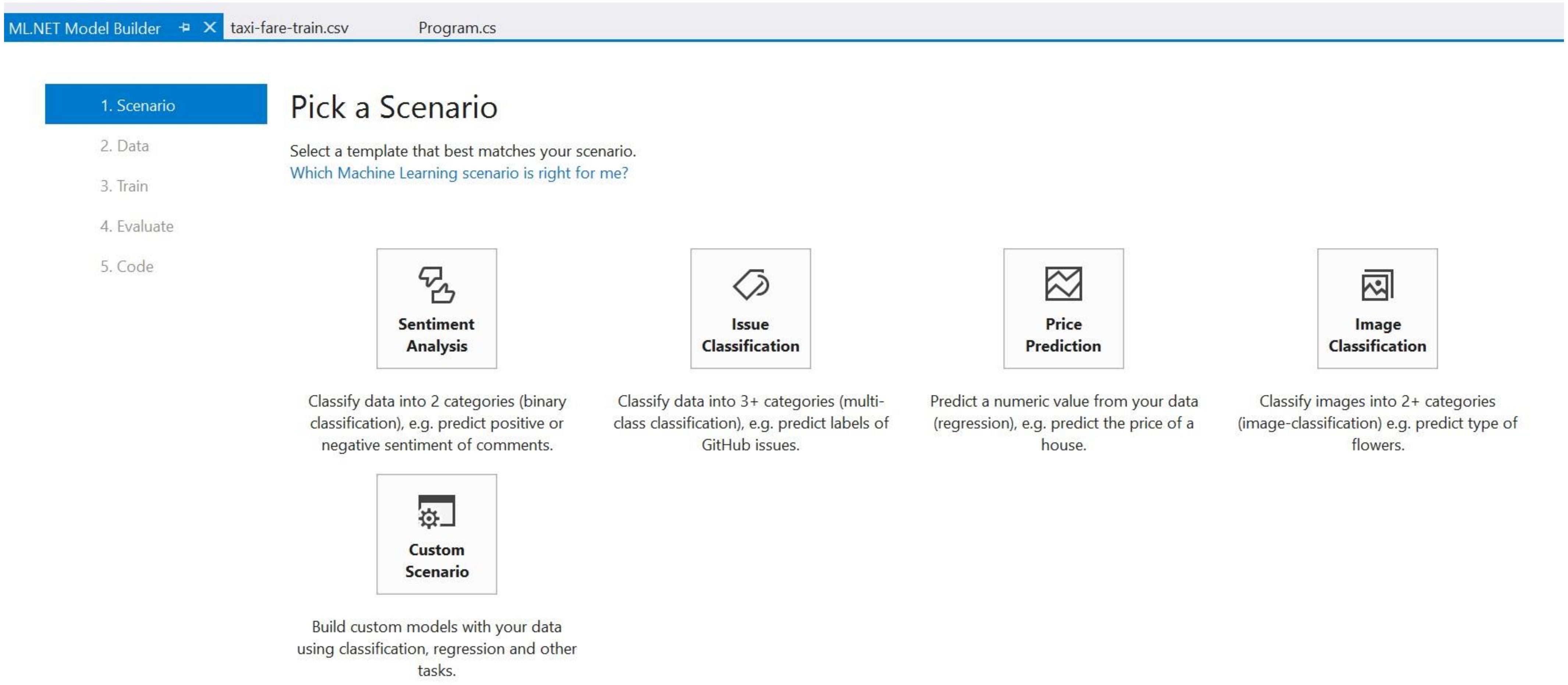
- Google **Inception v3**, **NASNet**
- Microsoft **ResNet**
- Oxford **VGG** Model, etc.

Add model to project



```
using System;

namespace image1
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

#1 Chose scenario



Pick a Scenario

Select a template that best matches your scenario.
Which Machine Learning scenario is right for me?

Sentiment Analysis

Classify data into 2 categories (binary classification), e.g. predict positive or negative sentiment of comments.

Issue Classification

Classify data into 3+ categories (multi-class classification), e.g. predict labels of GitHub issues.

Price Prediction

Predict a numeric value from your data (regression), e.g. predict the price of a house.

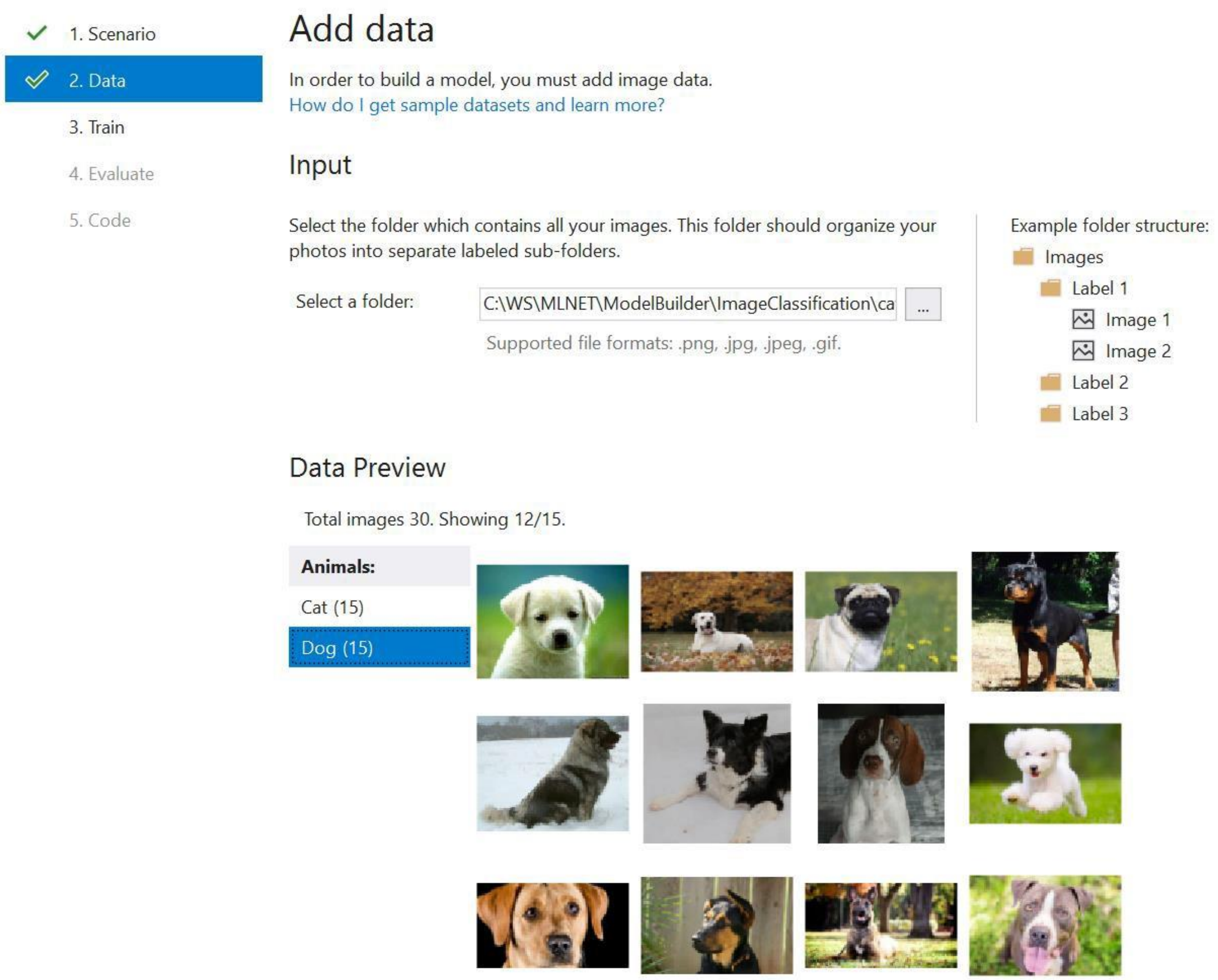
Image Classification

Classify images into 2+ categories (image-classification) e.g. predict type of flowers.

Custom Scenario

Build custom models with your data using classification, regression and other tasks.

#2 Add data



1. Scenario
2. Data
3. Train
4. Evaluate
5. Code

Add data

In order to build a model, you must add image data.
How do I get sample datasets and learn more?

Input

Select the folder which contains all your images. This folder should organize your photos into separate labeled sub-folders.

Select a folder: C:\WS\MLNET\ModelBuilder\ImageClassification\ca

Supported file formats: .png, .jpg, .jpeg, .gif.

Example folder structure:

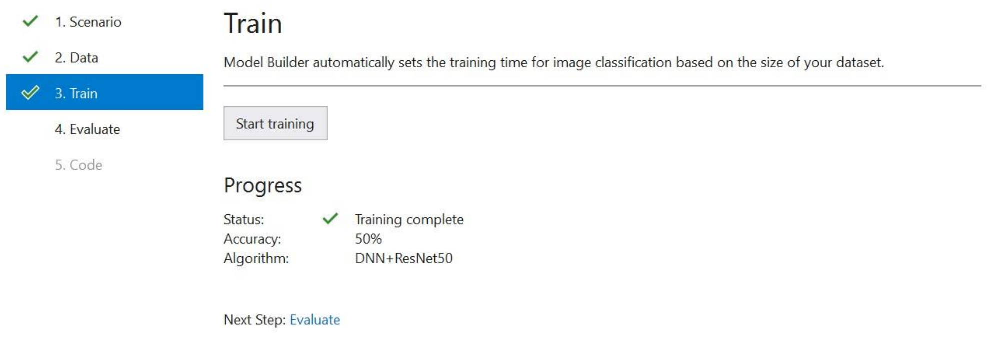
```
Images
  Label 1
    Image 1
    Image 2
  Label 2
  Label 3
```

Data Preview

Total images 30. Showing 12/15.

Animals:
Cat (15)
Dog (15)

#3 Train model



✓ 1. Scenario
✓ 2. Data
✓ 3. Train
4. Evaluate
5. Code

Train

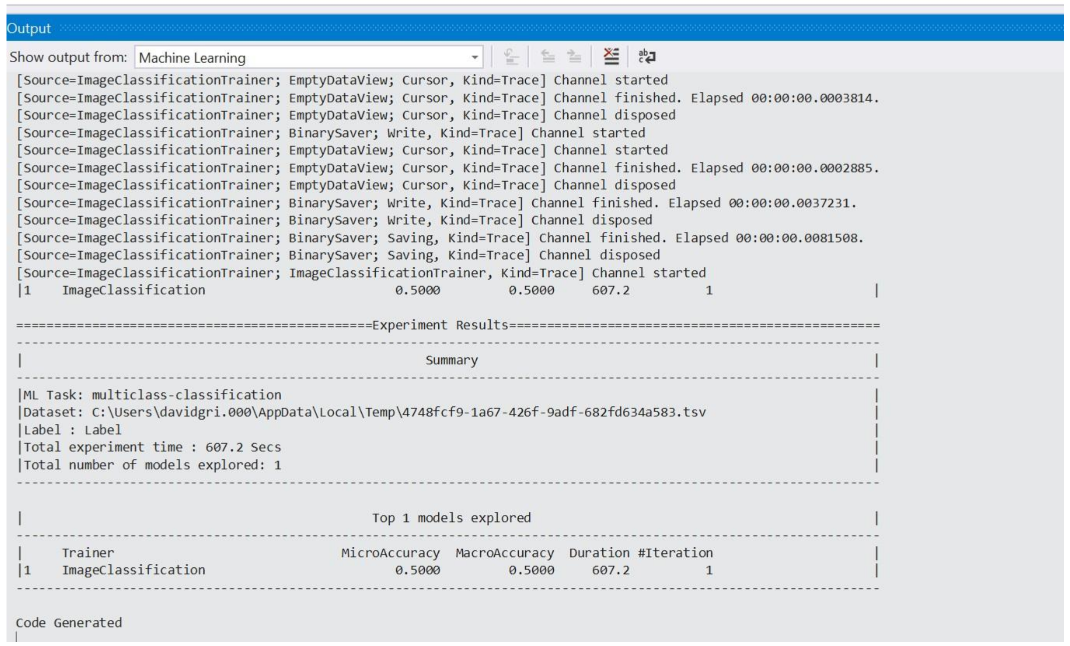
Model Builder automatically sets the training time for image classification based on the size of your dataset.

Start training

Progress

Status:	✓ Training complete
Accuracy:	50%
Algorithm:	DNN+ResNet50

Next Step: Evaluate

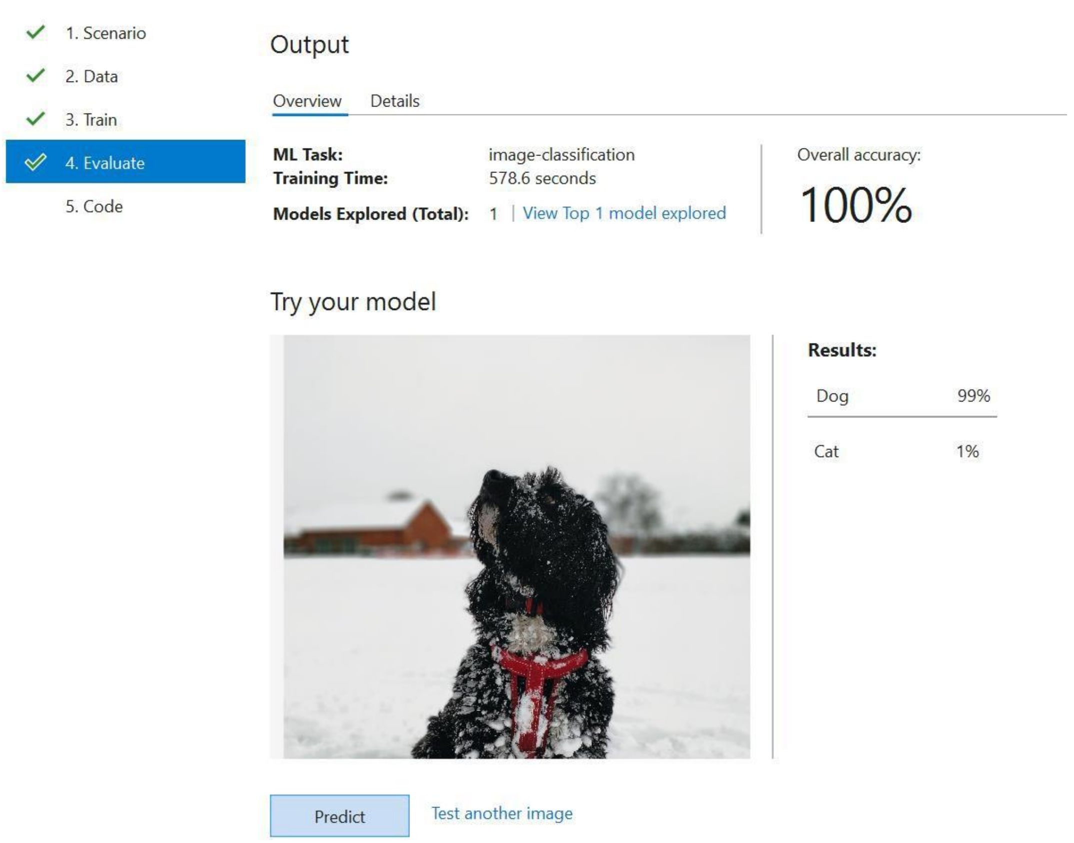


Output

Show output from: Machine Learning

```
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel started
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel finished. Elapsed 00:00:00.0003814.
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel disposed
[Source=ImageClassificationTrainer; BinarySaver; Write, Kind=Trace] Channel started
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel started
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel finished. Elapsed 00:00:00.0002885.
[Source=ImageClassificationTrainer; EmptyDataView; Cursor, Kind=Trace] Channel disposed
[Source=ImageClassificationTrainer; BinarySaver; Write, Kind=Trace] Channel finished. Elapsed 00:00:00.0037231.
[Source=ImageClassificationTrainer; BinarySaver; Write, Kind=Trace] Channel disposed
[Source=ImageClassificationTrainer; BinarySaver; Saving, Kind=Trace] Channel finished. Elapsed 00:00:00.0081508.
[Source=ImageClassificationTrainer; BinarySaver; Saving, Kind=Trace] Channel disposed
[Source=ImageClassificationTrainer; ImageClassificationTrainer, Kind=Trace] Channel started
|1    ImageClassification                         0.5000         0.5000     607.2          1                     |

===============================================Experiment Results=================================================
------------------------------------------------------------------------------------------------------------------
|                                                     Summary                                                    |
------------------------------------------------------------------------------------------------------------------
|ML Task: multiclass-classification                                                                              |
|Dataset: C:\Users\davidgri.000\AppData\Local\Temp\4748fcf9-1a67-426f-9adf-682fd634a583.tsv                      |
|Label : Label                                                                                                   |
|Total experiment time : 607.2 Secs                                                                              |
|Total number of models explored: 1                                                                              |
------------------------------------------------------------------------------------------------------------------

|                                              Top 1 models explored                                             |
------------------------------------------------------------------------------------------------------------------
|     Trainer                              MicroAccuracy  MacroAccuracy  Duration #Iteration                     |
|1    ImageClassification                         0.5000         0.5000     607.2          1                     |
------------------------------------------------------------------------------------------------------------------

Code Generated
|
```

#4 Evaluate model



Output

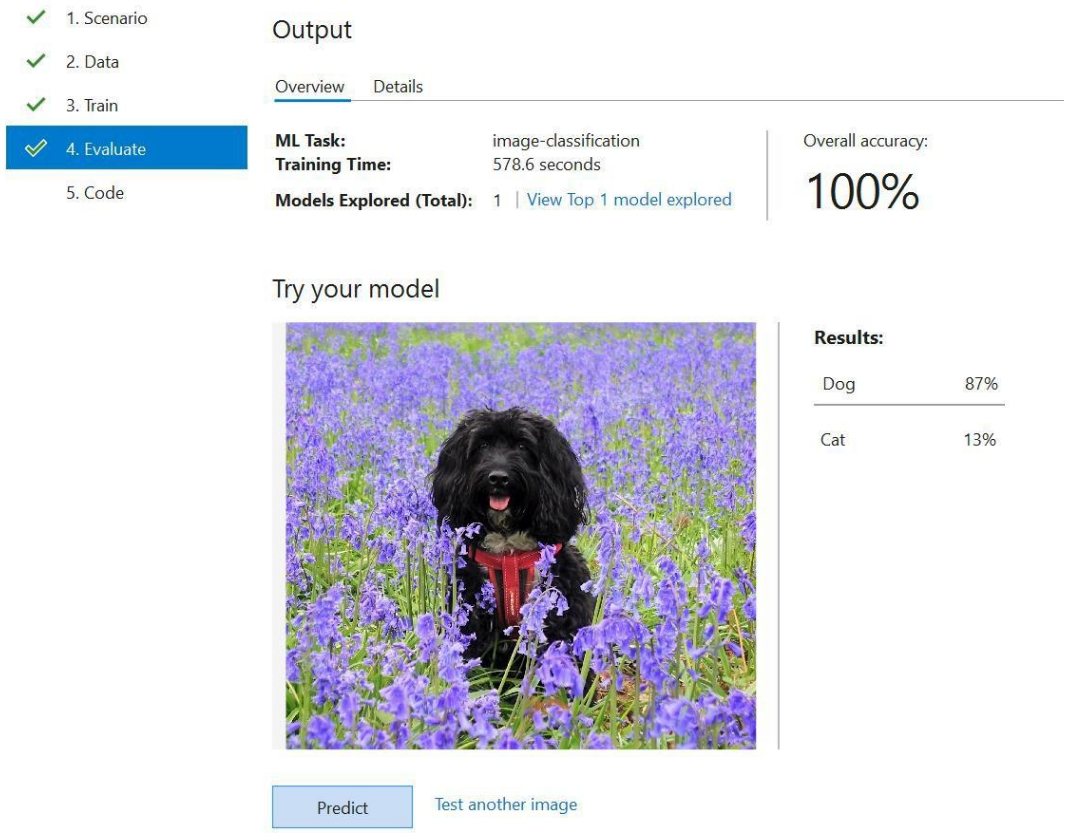
ML Task: image-classification
Training Time: 578.6 seconds
Models Explored (Total): 1 | View Top 1 model explored

Overall accuracy:
100%

Try your model

Results:

Dog	99%
Cat	1%



Output

ML Task: image-classification
Training Time: 578.6 seconds
Models Explored (Total): 1 | View Top 1 model explored

Overall accuracy:
100%

Try your model

Results:

Dog	87%
Cat	13%

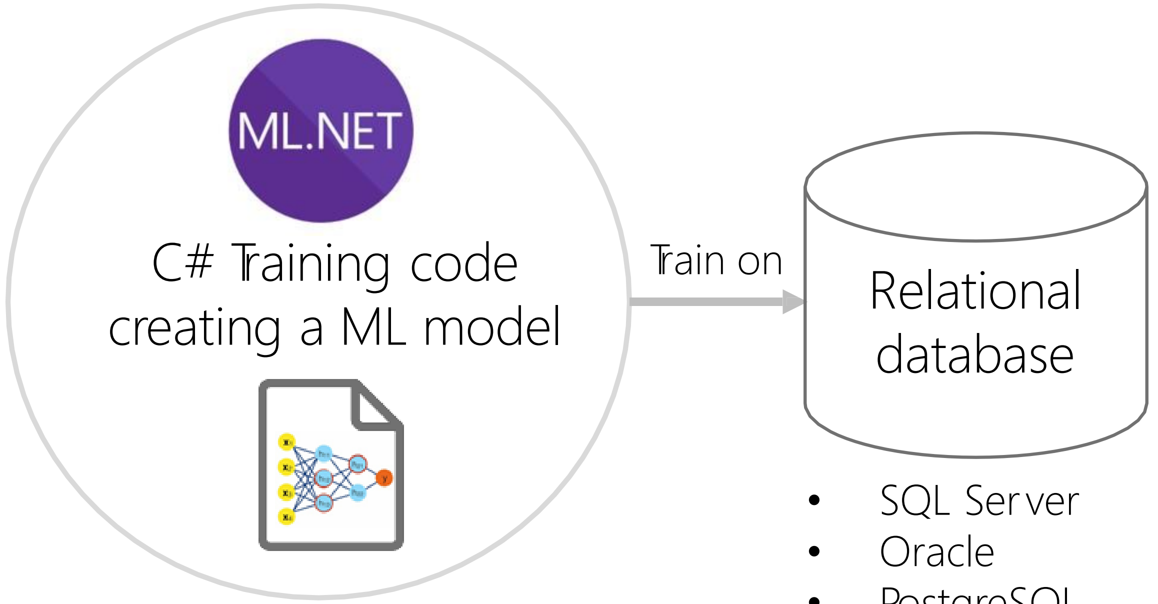
Latest updates…

Database Loader

Enabled scenarios:

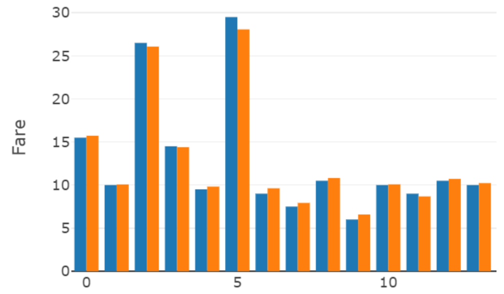
- Training directly against relational databases.
- Simple and out-of-the-box code
- Supports any RDBMS supported by System.Data

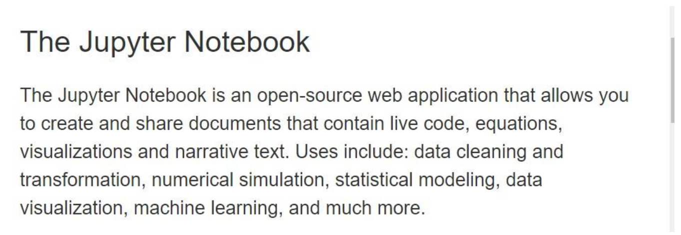
- v1.4-preview release



- SQL Server
- Oracle
- PostgreSQL
- MySQL
- etc.

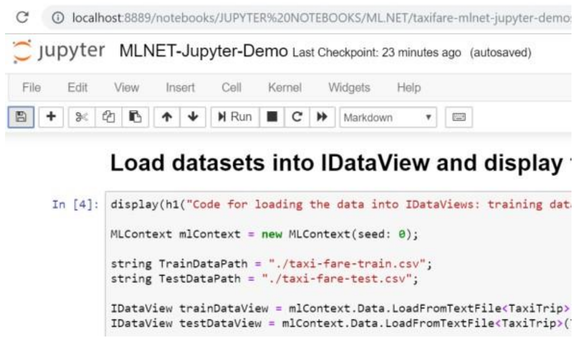
ML.NET and C# in Jupyter Notebooks!





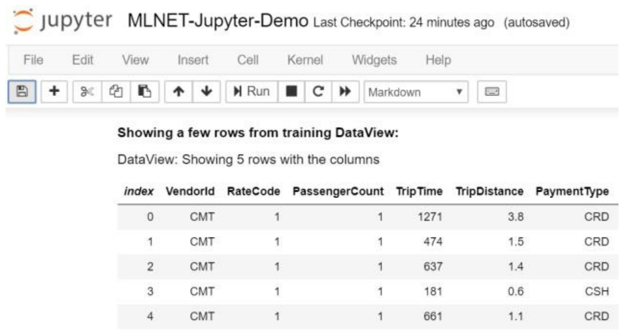
The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.


localhost:8889/notebooks/JUPYTER%20NOTEBOOKS/ML.NET/taxifare-mlnet-jupyter-demo
jupyter MLNET-Jupyter-Demo Last Checkpoint: 23 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Markdown

Load datasets into IDataView and display

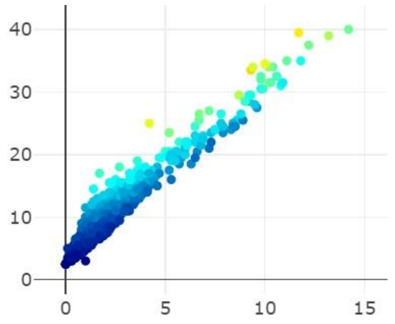
```
In [4]: display(h1("Code for loading the data into IDataViews: training dat

        MLContext mlContext = new MLContext(seed: 0);

        string TrainDataPath = "./taxi-fare-train.csv";
        string TestDataPath = "./taxi-fare-test.csv";

        IDataView trainDataView = mlContext.Data.LoadFromTextFile<TaxiTrip>
        IDataView testDataView = mlContext.Data.LoadFromTextFile<TaxiTrip>(
```


jupyter MLNET-Jupyter-Demo Last Checkpoint: 24 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run
Markdown

Showing a few rows from training DataView:

DataView: Showing 5 rows with the columns

index	VendorId	RateCode	PassengerCount	TripTime	TripDistance	PaymentType
0	CMT	1	1	1271	3.8	CRD
1	CMT	1	1	474	1.5	CRD
2	CMT	1	1	637	1.4	CRD
3	CMT	1	1	181	0.6	CSH
4	CMT	1	1	661	1.1	CRD



Roadmap ahead

- Scale-out on Azure for training and consumption
- Improve tooling in Visual Studio and .NET (Model Builder & CLI)
  - Improve AutoML.NET experience for all ML scenarios
- Object Detection training (Deep Learning based)
- Text Analysis support (Deep Learning based, i.e. BERT)
- ARM / full ONNX support (Enablers for Xamarin & IoT scenarios)

Bibiography

Get started at **http://dot.net/ml**

Try the samples at **http://aka.ms/mlnetsamples**

Read the docs at **http://aka.ms/mlnetdocs**

Watch ML.NET videos at **https://aka.ms/mlnetyoutube**

Request features or contribute at **http://aka.ms/mlnet**