



Machine Learning made it easy with ML.NET

BY RICCARDO TERRELL - @TRIKACE

Agenda

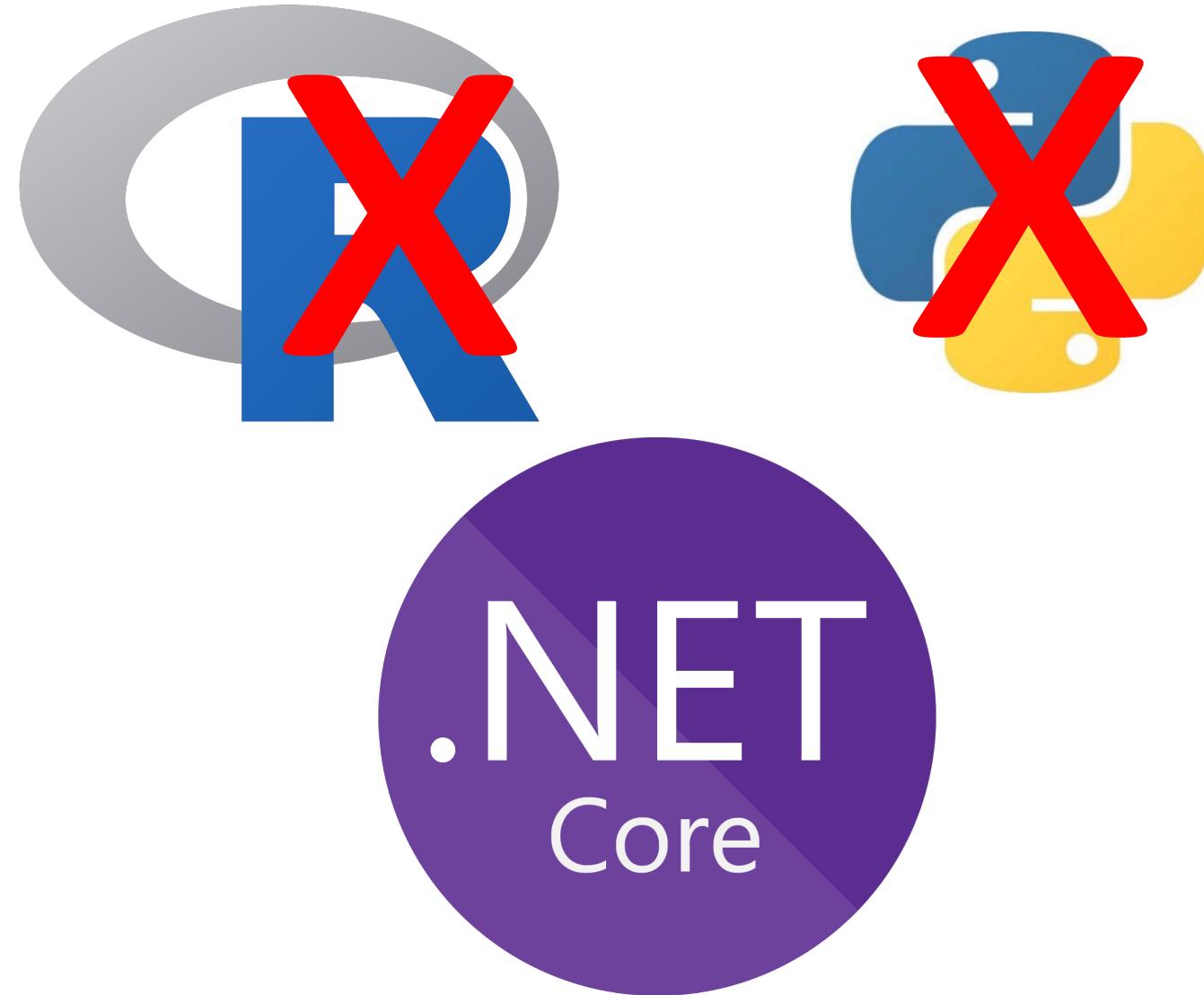
What are we going to cover?



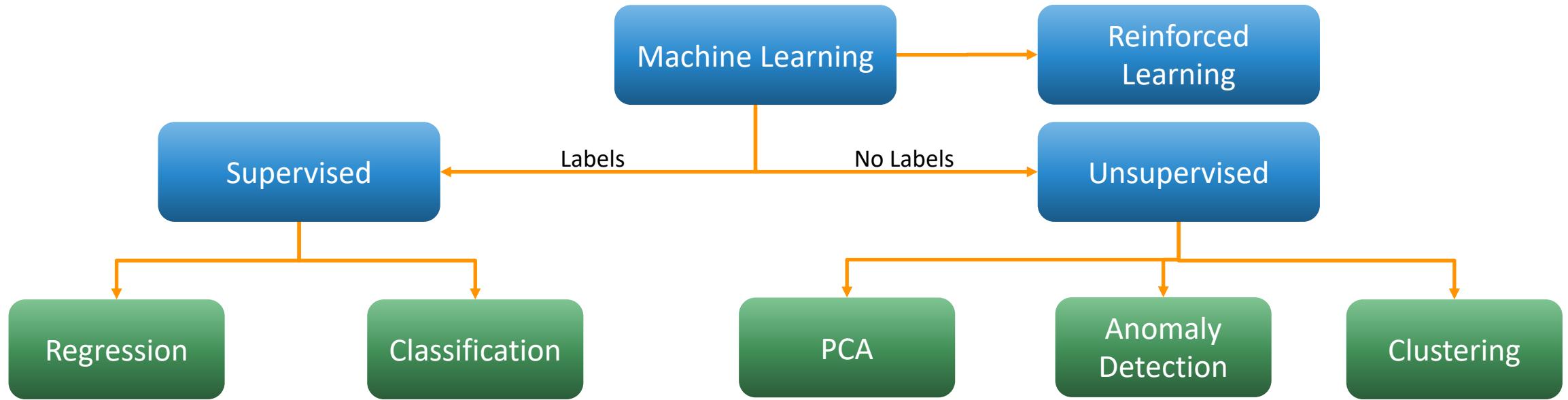
1. What is Machine Learning? A brief introduction to machine learning
2. What is ML.NET?
3. The ML.NET workflow
4. ML.NET in action
 - a. Regression
 - b. Classification
 - c. Recommendation
 - d. Clustering
5. Deep learning in ML.NET (integration with TensorFlow)
6. AutoML

Why should you use ML.NET?

- Want to stay in .NET ecosystem for Machine Learning
- Don't want to worry about low-level complexities of ML
- Want to train a custom model
- Want to consume a pre-trained model

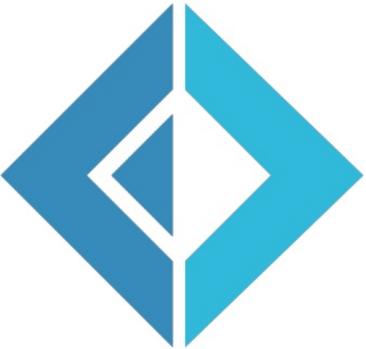


Machine Learning Types



Why F# for machine learning

- Exploratory programming
- Interactive environment
- Data Pipelines
- Functions Composition
- Algebraic data types
- Advanced pattern matching
- Strong and Duck typing



$$y = f(x_1, x_2 \dots x^n)$$

```
let trainData = context.Data.LoadFromTextFile<Passenger>(trainDataPath, hasHeader = true, separatorChar = ',')  
// set up a training pipeline  
let pipeline = EstimatorChain()  
    .Append(context.Transforms.Conversion.ConvertType("Age", outputKind = DataKind.Single))  
    .Append(context.Transforms.Concatenate("Features", "Age", "Pclass", "SibSp", "Parch", "Sex", "Embarked"))  
    .Append(context.BinaryClassification.Trainers.FastTree())  
  
// train the model  
let model = trainData |> pipeline.Fit  
  
// make predictions and compare with ground truth  
let metrics = testData |> model.Transform |> context.BinaryClassification.Evaluate  
  
// set up a prediction engine  
let engine = context.Model.CreatePredictionEngine model  
  
// create sample records  
let passenger = { Pclass = 1.0f; Sex = "female"; RawAge = "48"; Ticket = "B"; Fare = 30.0f }  
// make the prediction  
let prediction = engine.Predict passenger
```

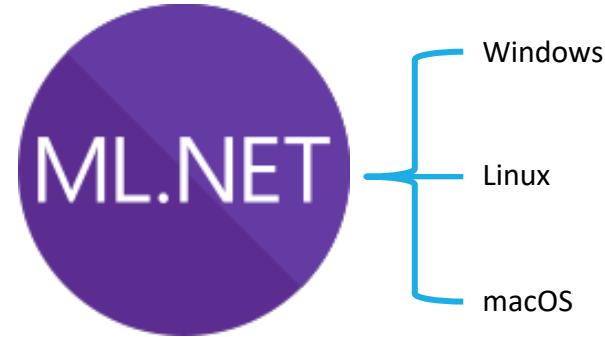
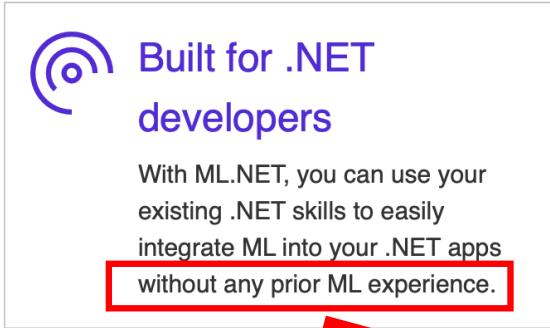
Where ML.NET is used?

Trusted and Proven at Scale



What is ML.NET

- Open Source & Cross Platform
- Machine learning SDK that works offline
- Simple and powerful
- Support other ML models like TensorFlow and ONNX
- Great documentation available online
- Reuse .Net Skills

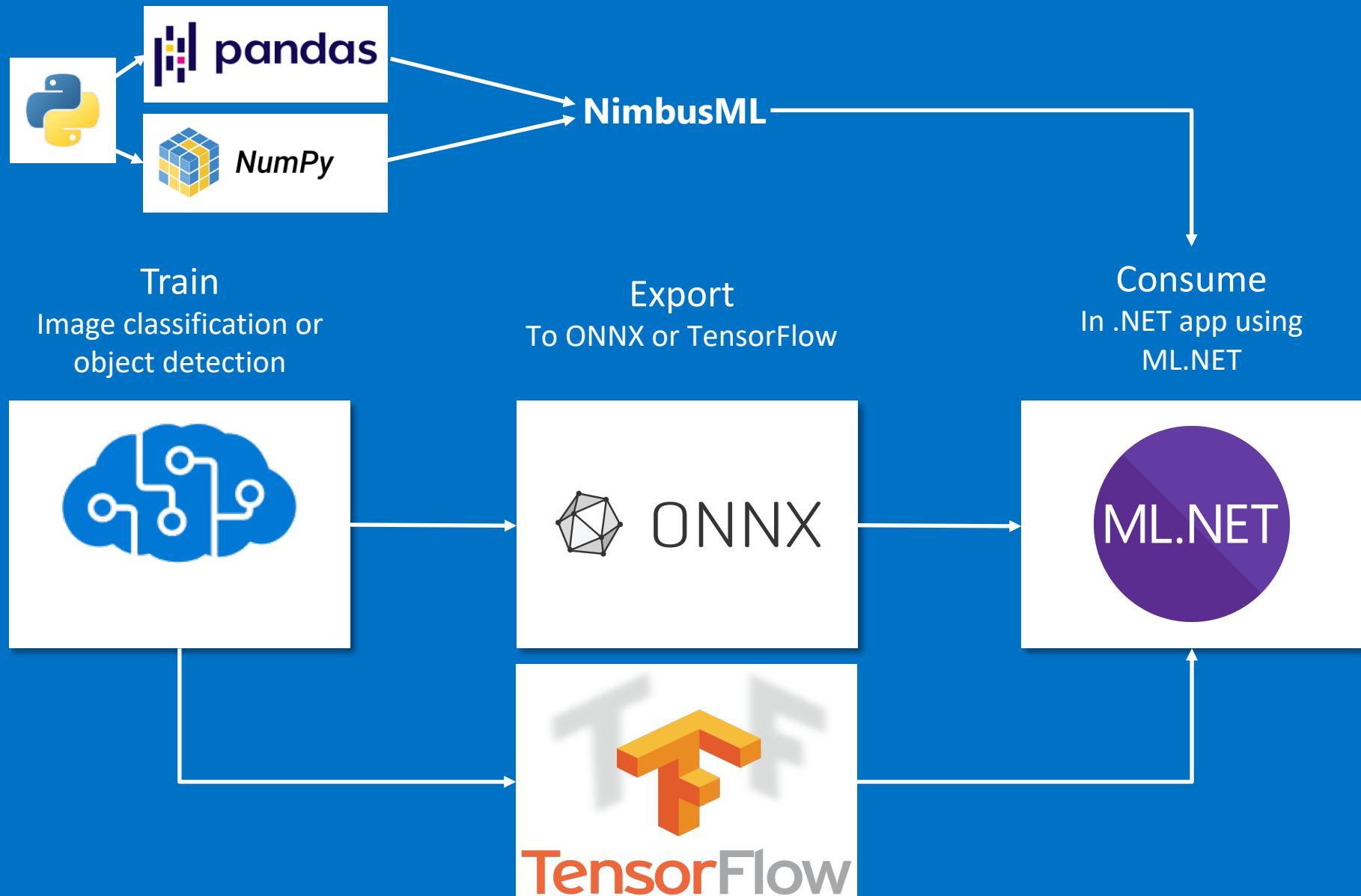


Built for
.NET Developers

Can use existing C# and F# skills to integrate ML into .NET apps

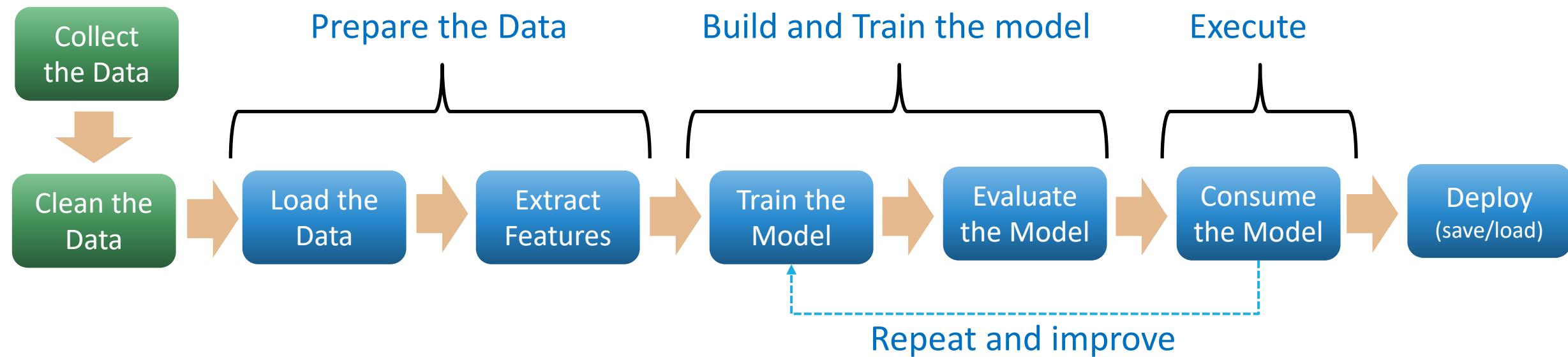
Data science & ML experience not required

ML.NET integration with other ML techs



The ML.NET workflow

Building an ML Model involves few steps



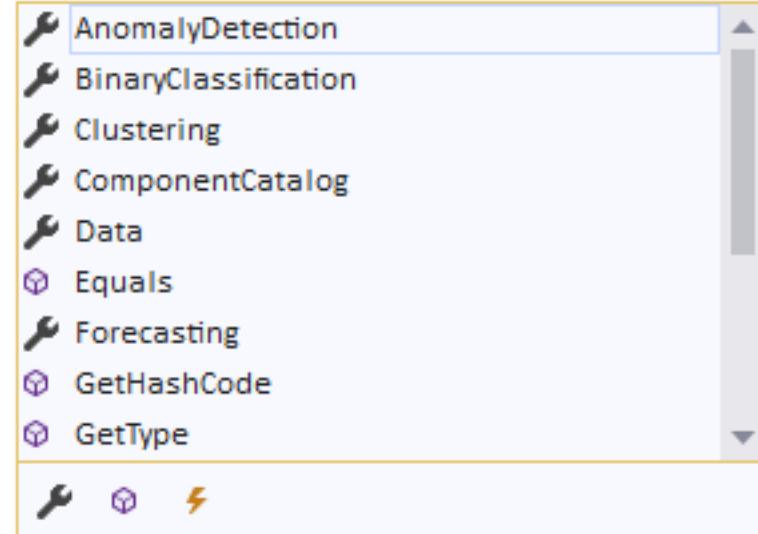
ML.NET Pipeline step by step

- Create ML.NET Context
- Load data
- Transform data
- Train model
- Evaluate model
- Consume model

Create ML.NET Context

```
// Create MLContext to be shared across the model  
// Set a random seed for repeatable/deterministic  
// results across multiple trainings.  
let mlContext = MLContext(seed = Nullable 1)
```

```
var mlContext = new MLContext();  
mlContext.|
```



Create the Data schema

- Create ML.NET Context
 - **Load data**
 - Transform data
 - Train model
 - Evaluate model
 - Consume model

Dataset

FlowerType	SepalLength	SepalWidth	PetalLength	PetalWidth
Iris-setosa	5.1	3.5	1.4	0.2
Iris-versicolor	7.0	3.2	4.7	1.4
Iris-setosa	4.9	3.0	1.5	0.1
...				

Labels

Features

Class definition of schema

```
[<CLIMutable>]  
type IrisData = {  
    [<>LoadColumn(0)</>] SepalLength : float32  
    [<>LoadColumn(1)</>] SepalWidth : float32  
    [<>LoadColumn(2)</>] PetalLength : float32  
    [<>LoadColumn(3)</>] PetalWidth : float32  
    [<>LoadColumn(4)</>] Label : string  
}
```

```
[<CLIMutable>]
type IrisPrediction = {
    [<ColumnName("PredictLabel")>]
    PredictedLabel : uint32
    Score : float32[]
}
```

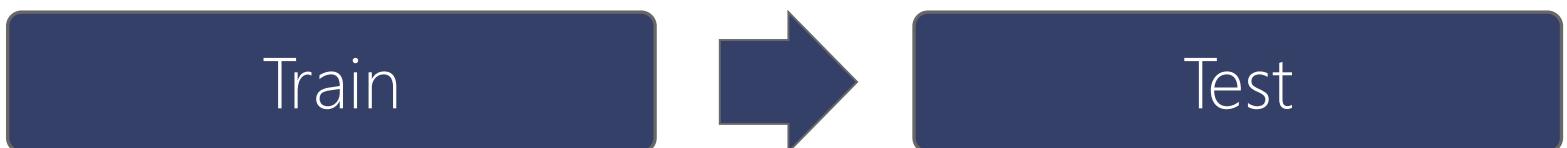
ML.NET Pipeline step by step

- Create ML.NET Context
- **Load data**
- Transform data
- Train model
- Evaluate model
- Consume model

Load the Data

```
let trainData : IDataView =  
  
    context.Data.LoadFromTextFile<Passenger>("train.csv",  
        hasHeader = true, separatorChar = ',')
```

```
let testData : IDataView =  
  
    context.Data.LoadFromTextFile<Passenger>("test.csv",  
        hasHeader = true, separatorChar = ',')
```



Use ~80%

Use ~20%

Clean and Normalize Data

- Create ML.NET Context
- Load data
- **Transform data**
- Train model
- Evaluate model
- Consume model

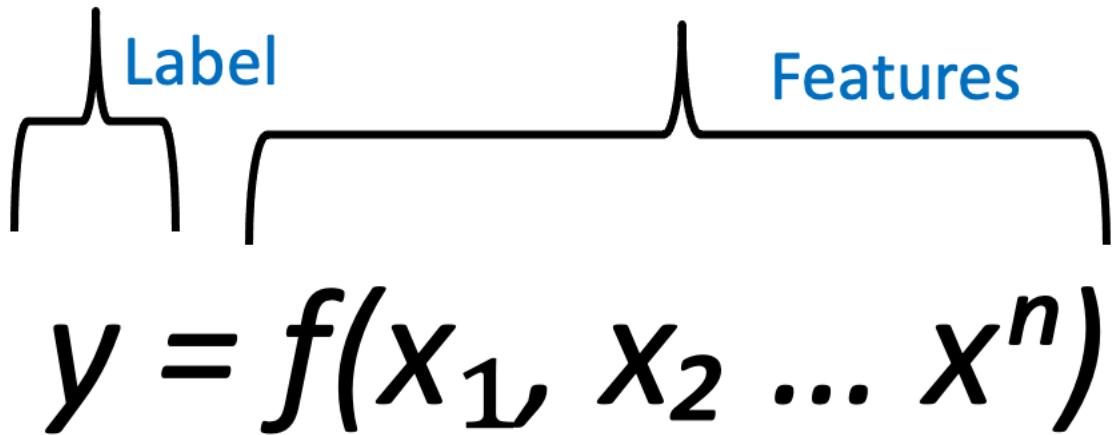


```
// one-hot encode all text features
.Append(context.Transforms.NormalizeMinMax("Price"))
.Append(context.Transforms.Categorical.OneHotEncoding("Make"))
.Append(context.Transforms.Categorical.OneHotEncoding("Model"))
.Append(context.Transforms.Conversion.ConvertType("Year", outputKind = DataKind.Single))
.Append(context.Transforms.Conversion.ConvertType("Mileage", outputKind = DataKind.Single))
```

ML.NET Pipeline step by step

- Create ML.NET Context
- Load data
- Transform data
- **Train model**
- Evaluate model
- Consume model

Training the Model



ML.NET Pipeline step by step

- Create ML.NET Context
- Load data
- Transform data
- Train model
- **Evaluate model**
- Consume model

Evaluate the Model and improve accuracy

```
let metrics =  
    trainData  
    |> model.Transform  
    |> context.MulticlassClassification.Evaluate  
  
// show evaluation metrics  
printfn " Accuracy:      %f" metrics.Accuracy  
printfn " Auc:          %f" metrics.AreaUnderRocCurve  
printfn " Auprc:         %f" metrics.AreaUnderPrecisionRecallCurve  
printfn " F1Score:        %f" metrics.F1Score  
printfn " LogLoss:        %f" metrics.LogLoss  
printfn " LogLossReduction: %f" metrics.LogLossReduction  
printfn " PositivePrecision: %f" metrics.PositivePrecision  
printfn " PositiveRecall:   %f" metrics.PositiveRecall  
printfn " NegativePrecision: %f" metrics.NegativePrecision  
printfn " NegativeRecall:   %f" metrics.NegativeRecall
```

Cross-Validating Regression Models

- Regression CrossValidate divides a dataset into “folds” and scores each fold separately
- R-squared, Mean Absolute Error (MAE), Mean Squared Error (MSE), etc.
- Use mean of scores from all folds to get a more robust measure of accuracy

// Evaluate the model using cross-validation

```
let scores = context.BinaryClassification.CrossValidate(data = data,  
estimator = pipeline, numberOfFolds = 5)
```

```
let mean = scores |> Seq.averageBy(fun x -> x.Metrics.F1Score)
```

```
Common.printRed (sprintf "Mean cross-validated F1 score: %A" mean)
```

ML.NET Pipeline step by step

- Create ML.NET Context
- Load data
- Transform data
- Choose algorithm
- Train model
- Evaluate model
- **Consume model**

Consume the model

```
// train the model on the training set
let model = partitions.TrainSet |> pipeline.Fit

// get predictions and compare them to the ground truth
let metrics = partitions.TestSet |> model.Transform |> context.MulticlassClassification.Evaluate

// set up a prediction engine
let engine = context.Model.CreatePredictionEngine model

// create sample messages
let messages =
    { Message = "Why pay more for expensive meds when you can order them online and save $$$?" }
    { Message = "free price $250 weekly competition just text the word WIN to 80086 NOW" }
    { Message = "Home in 30 mins. Need anything from store?" }

let run () =
    // make the predictions
    printfn "Model predictions:"
    messages |> List.iter(fun m ->
        let p = engine.Predict m
        printfn " %f %s - Classification %s" p.Probability m.Message (if p.IsSpam then "Spam" else "Ham"))
```

Online Datasets

<http://archive.ics.uci.edu/ml/datasets.php>



The UCI Machine Learning Repository logo features a yellow 'UCI' monogram above a black and white illustration of an anteater. Below the logo is the text 'Machine Learning Repository' and 'Center for Machine Learning and Intelligent Systems'. To the right of the logo are links for 'About', 'Citation Policy', 'Donate a Data Set', and 'Contact'. A search bar and a 'Search' button are also present. At the bottom right are buttons for 'Repository' (selected) and 'Web', and a 'Google' search link.

Browse Through:

Default Task
Classification (417)
Regression (129)
Clustering (112)
Other (56)

Attribute Type
Categorical (38)
Numerical (375)
Mixed (55)

Data Type
Multivariate (434)
Univariate (27)
Sequential (55)
Time-Series (112)
Text (63)
Domain-Theory (23)

557 Data Sets

Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes
 Abalone	Multivariate	Classification	Categorical, Integer, Real	4177	8
 Adult	Multivariate	Classification	Categorical, Integer	48842	14
 Annealing	Multivariate	Classification	Categorical, Integer, Real	798	38
 Anonymous Microsoft Web Data		Recommender-Systems	Categorical	37711	294

<https://www.kaggle.com/datasets>



The Kaggle logo is a large, stylized blue word 'kaggle' with a drop shadow.

Public Your Datasets Favorites Sort by: Hottest

Trending YouTube Video Statistics	3005
 Trending YouTube Video Statistics Mitchell J 1 year 201 MB 7.9 20 Files (CSV, JSON) 5 Tasks	3005
 TMDB 5000 Movie Dataset The Movie Database (TMDb) 3 years 9 MB 8.2 2 Files (CSV) 2 Tasks	2391
 World Happiness Report Sustainable Development Solutions Network 1 year 37 KB 8.5 5 Files (CSV) 2 Tasks	2232
 Netflix Movies and TV Shows Shivam Bansal 9 months 971 KB 10.0 1 File (CSV) 4 Tasks	1922

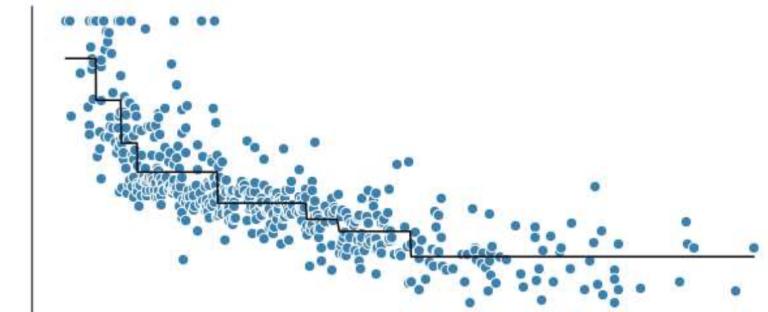
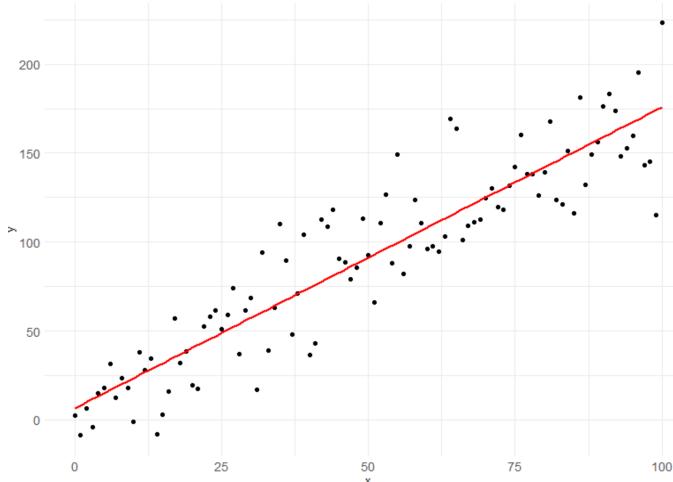
Regression Decision Trees and Random Forests

How much / how many

Examples:

- Predict the temperature
- Predict house prices
- Predict next month's sales

Predict price of used car



Classification

SMS Spam Detection



Dataset

SMS Spam Collection Dataset

by UCI Machine Learning

4 years ago • 211 KB • ▲ 669

SMS [Spam](#) Collection Dataset



Dataset

Email Spam

by Wessel van Lit

2 years ago • 12 MB • ▲ 54

Email [Spam](#)



Comment

Reply to Email spam or not spam classification

by KUNTAL SARKAR

f

2 years ago • Getting Started

Email [spam](#) or not [spam](#) classification



Dataset

Spam or Not Spam Dataset

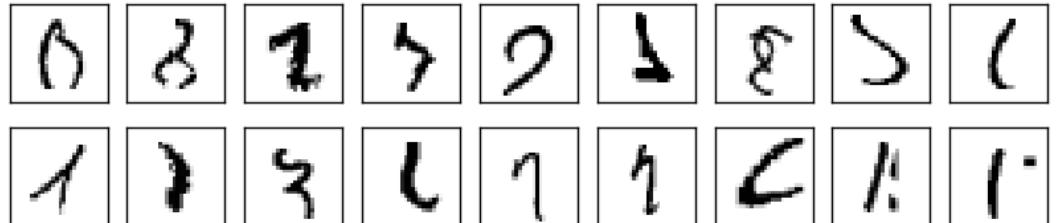
by Hakan Ozler

2 years ago • 1 MB • ▲ 21

Spam or Not [Spam](#) Dataset

MultiClass Classification

Optical Character Recognition



Recommendation

Movie recommendation

kaggle

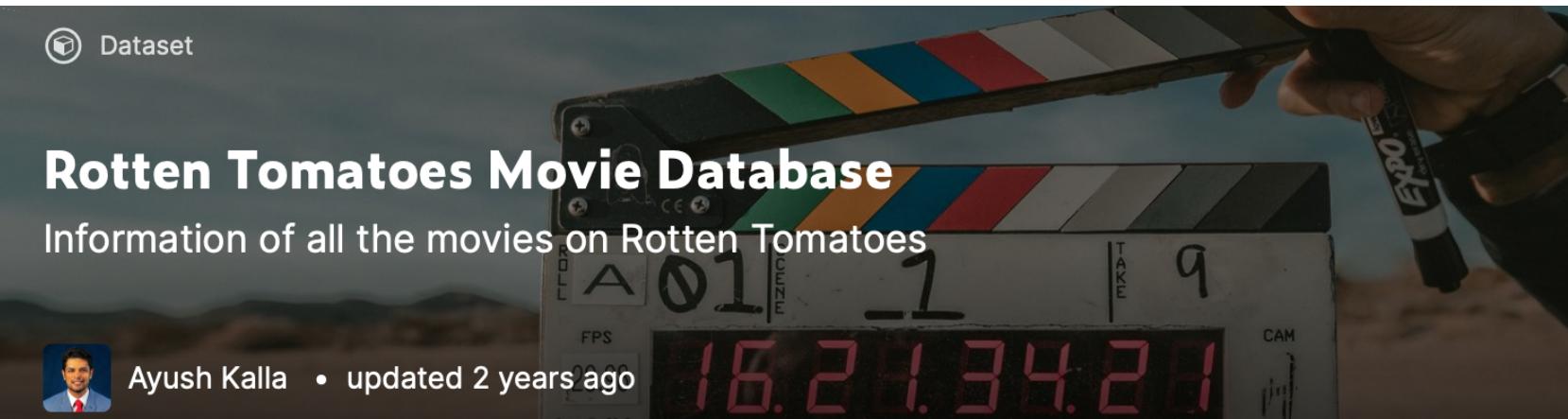
Dataset

Rotten Tomatoes Movie Database
Information of all the movies on Rotten Tomatoes

Ayush Kalla • updated 2 years ago

16.21.34.21

Data Tasks Notebooks (1) Discussion Activity Metadata Download (30 MB)

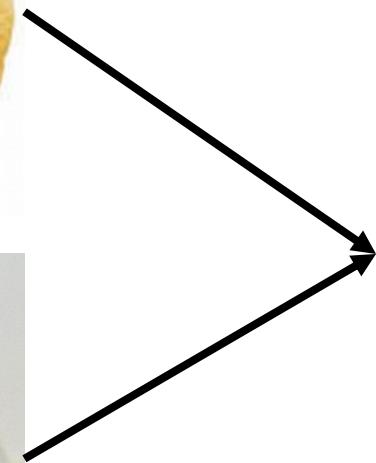


ML.NET integration with TensorFlow Transfer Learning



Image classification based on deep neural network





Bread
or
Pug?

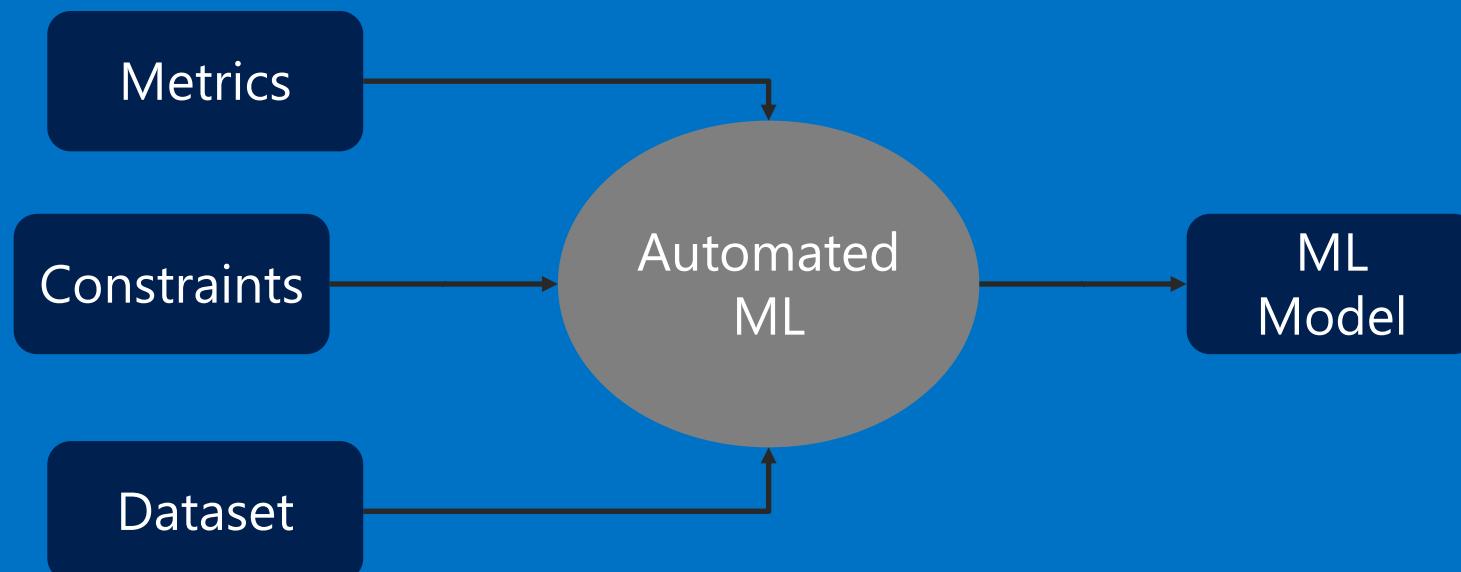


Bread



Pug





AutoML

Automated Machine Learning

Available in three from factors

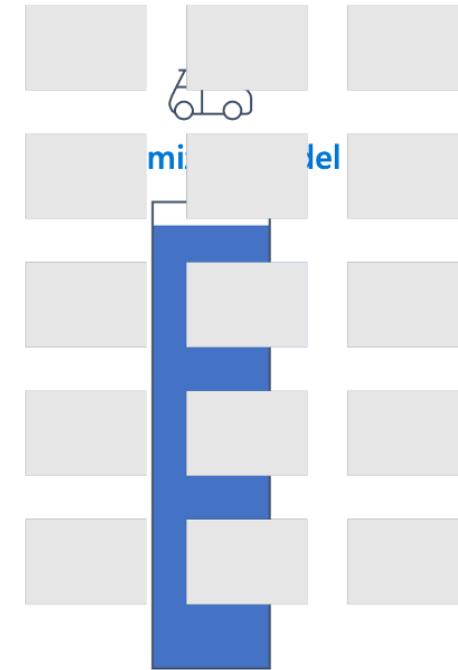
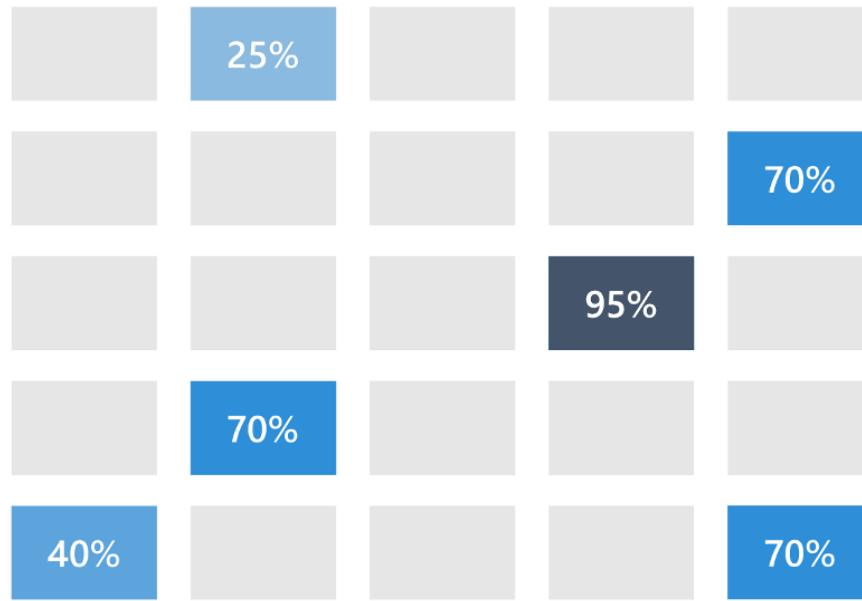
- ML.NET CLI (Command Line)
- ML.NET Model Builder (Graphical User Interface)
- ML.NET AutoML API (Application)

Input

Intelligently test multiple models in parallel

Output

- 101010
- 010101
- 101010
- Enter data**
-  **Define goals**
-  **Apply constraints**

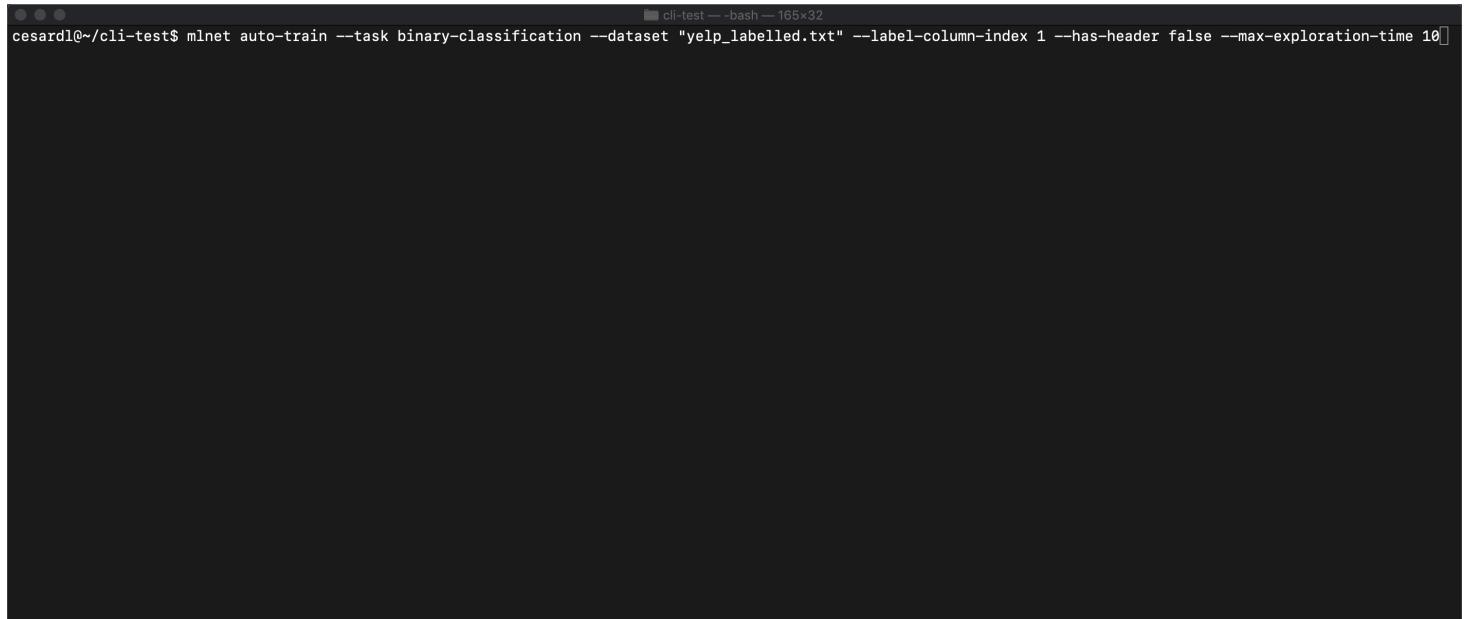


What is automated machine learning?

Automated ML Accelerates Model Development

ML.NET CLI

- Easily build custom ML models with AutoML
- Generates code for training and consumption
- Model Builder
 - Currently in Visual Studio only
 - Integration with Azure ML
- ML.NET CLI
 - Cross platform



A screenshot of a terminal window titled "cli-test — -bash — 165x32". The command entered is:

```
cesardl@~/cli-test$ mlnet auto-train --task binary-classification --dataset "yelp_labelled.txt" --label-column-index 1 --has-header false --max-exploration-time 10
```