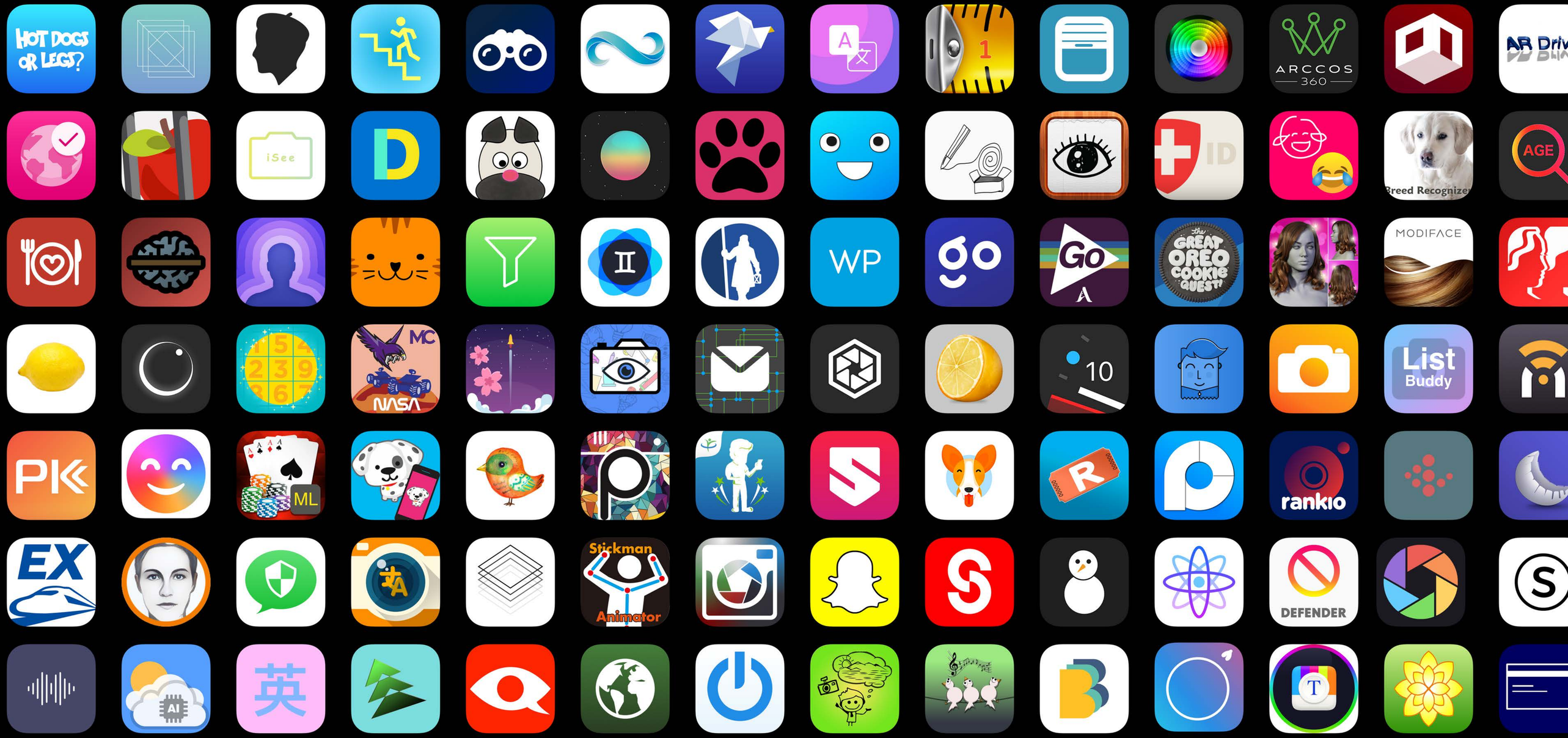
#WWDC18

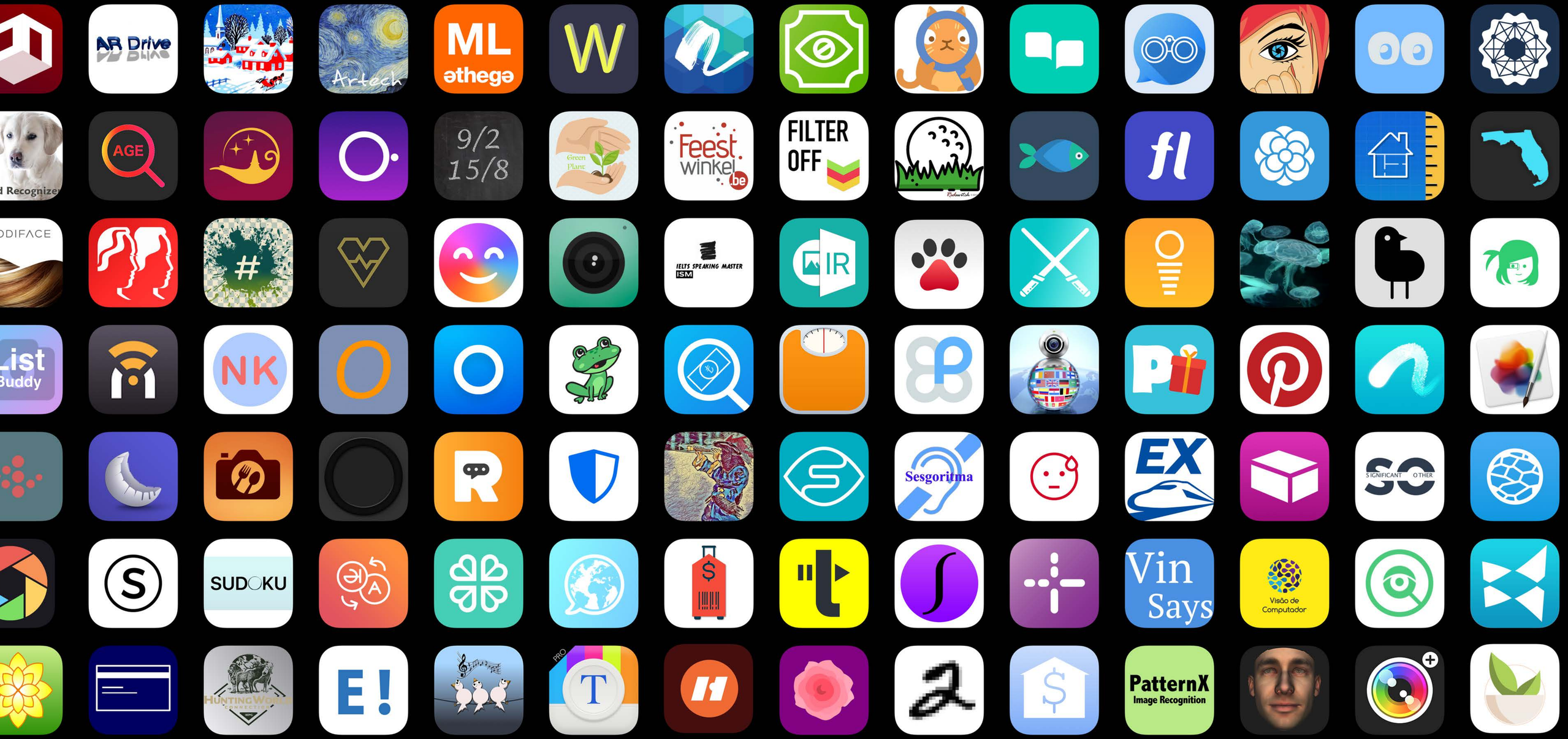
# Introducing Create ML

Gaurav Kapoor, Core ML
Lizi Ottens, Core ML
Tao Jia, Core ML

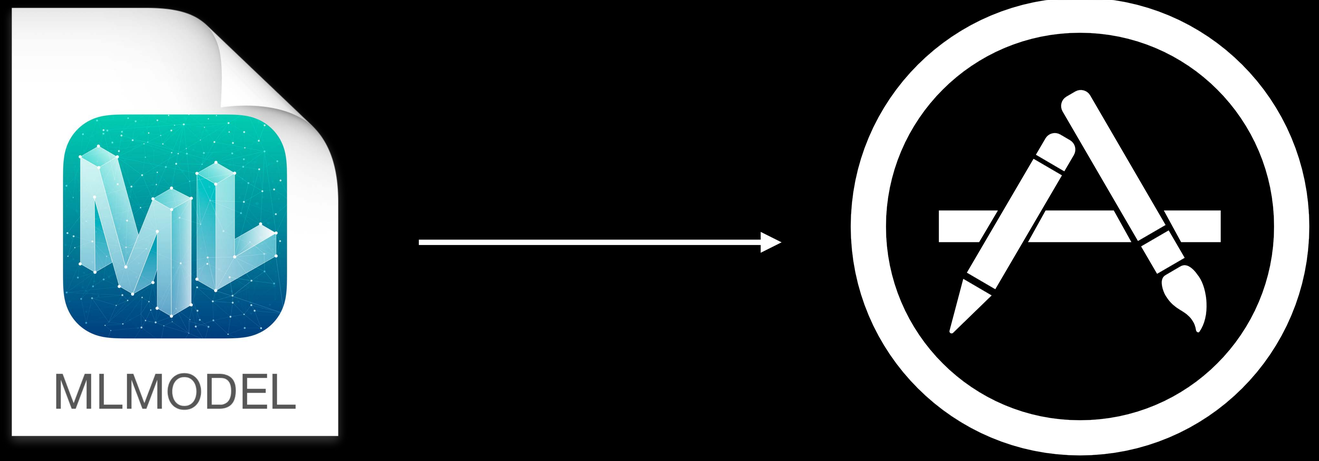
© 2018 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.

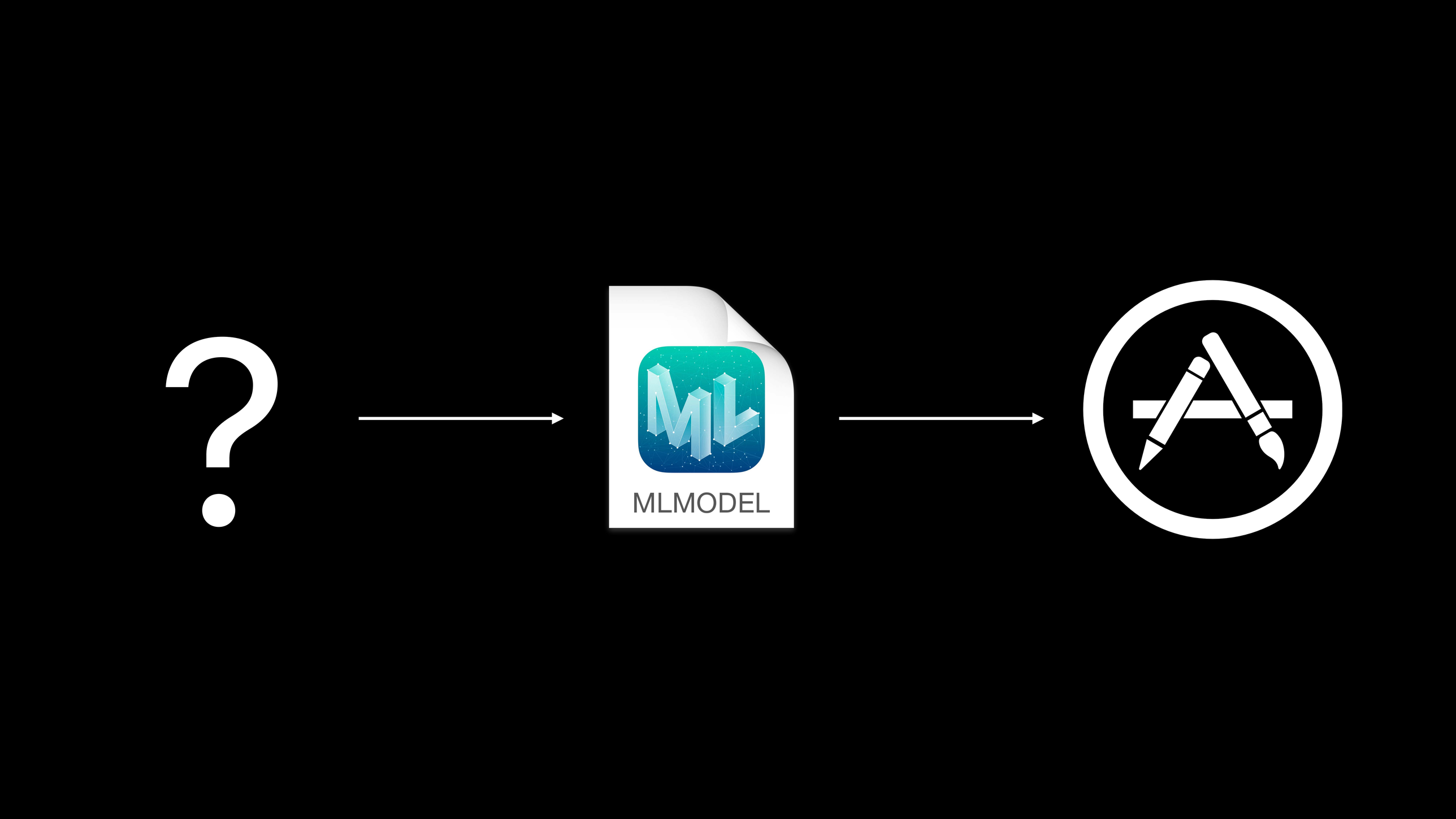

Core ML


HOT DOGS OR LEGS?
ARCCOS 360
iSee
Breed Recognizer
AGE
WP
Go
GREAT OREO COOKIE QUEST!
MODIFACE
List Buddy
NASA
MC
ML
rankio
DEFENDER
Stickman Animator
AI
英


AR Drive
Artech
ML
athega
W
AGE
9/2
15/8
Green
Plant
Feest
winkel
be
FILTER
OFF
fl
IELTS SPEAKING MASTER
ISM
IR
NK
Sesgoritma
EX
SIGNIFICANT
OTHER
SUDOKU
Vin
Says
Visão de
Computador
HUNTING WORLD
E!
PRO
PatternX
Image Recognition

Thank you!


MLMODEL


?
MLMODEL


# Working with Models

Build your apps with the ready-to-use Core ML models below, or use Core ML Tools to easily convert models into the Core ML format.

## Models

### MobileNet

MobileNets are based on a streamlined architecture that have depth-wise separable convolutions to build lightweight, deep neural networks.

Detects the dominant objects present in an image from a set of 1000 categories such as trees, animals, food, vehicles, people, and more.

View original model details
Download Core ML Model (17.1 MB)

### SqueezeNet

Detects the dominant objects present in an image from a set of 1000 categories such as trees, animals, food, vehicles, people, and more.

With an overall footprint of only 5 MB, SqueezeNet has a similar level of accuracy as AlexNet but with 50 times fewer parameters.

View original model details
Download Core ML Model (5 MB)

## Model Converters

### Core ML Tools

Core ML Tools is a python package that can be used to convert models from machine learning toolboxes into the Core ML format.

Get Core ML Tools ↗

### Apache MXNet

MXNet helps you train machine learning models and convert them into the Core ML format.

Get MXNet model converter ↗

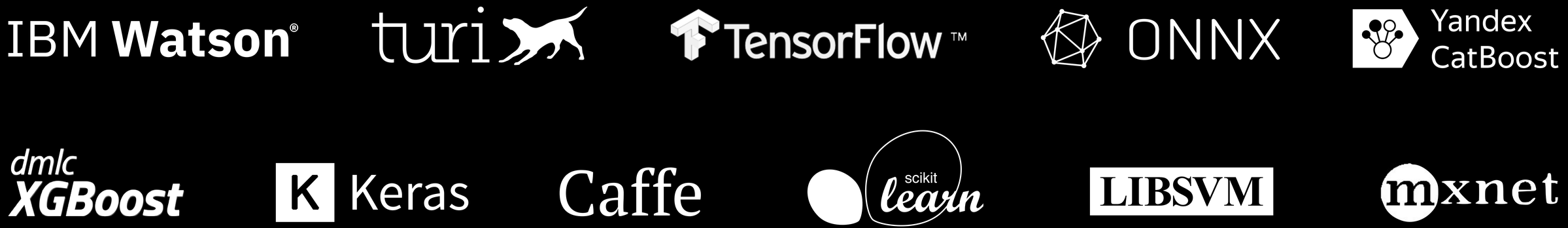
### TensorFlow

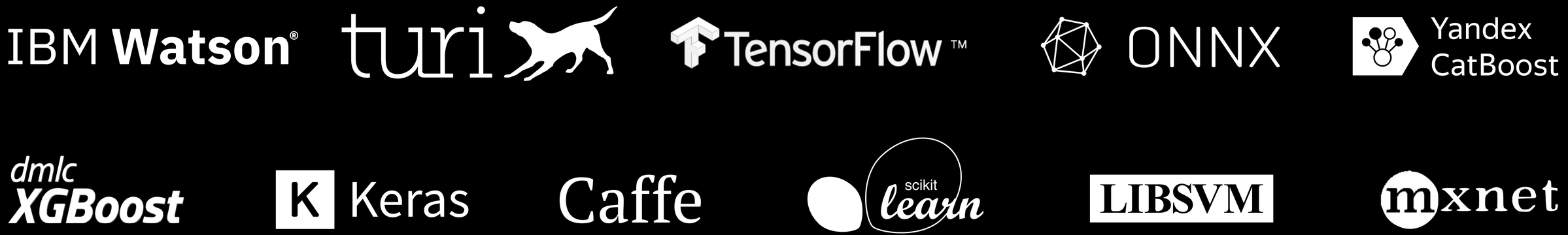
Train machine learning models in TensorFlow and easily convert them to the Core ML Model format.

Get TensorFlow Converter ↗

### ONNX New

Convert ONNX models you have created to the Core ML Model format.


IBM Watson®
turi
TensorFlow™
ONNX
Yandex
CatBoost
dmlc
XGBoost
K
Keras
Caffe
scikit
learn
LIBSVM
mxnet


IBM Watson®
turi
TensorFlow™
ONNX
Yandex
CatBoost
dmlc
XGBoost
K
Keras
Caffe
scikit
learn
LIBSVM
mxnet

Continuing our journey...

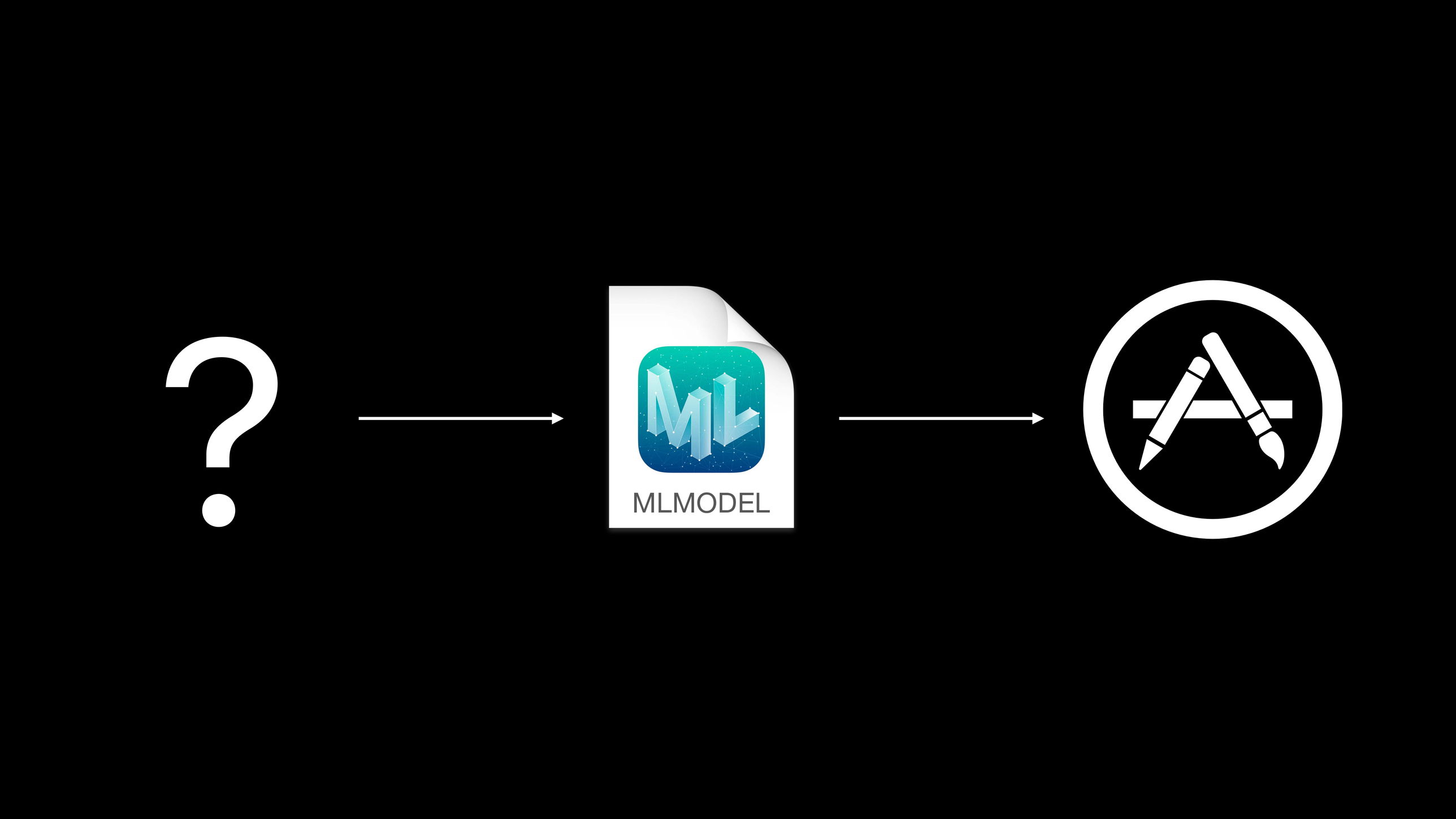
Native


XCODE


WWDC18
You


Create ML

# Machine Learning in Swift


?
MLMODEL


MLMODEL

Simple model creation

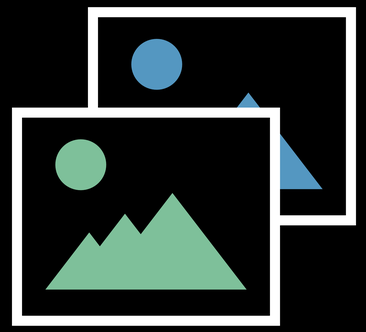
Tailored to your app

Leverages core Apple technologies

Powered by Mac

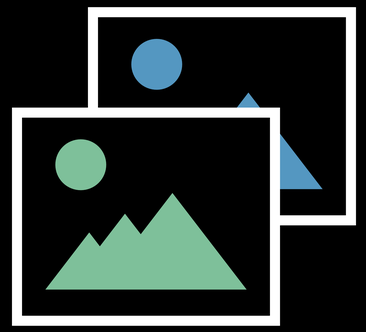
# Data

# Data



Images

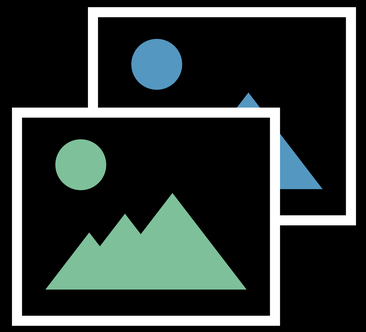
# Data



Images

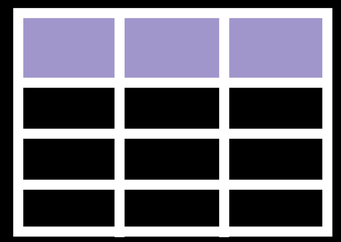
Text

# Data



Images

Text

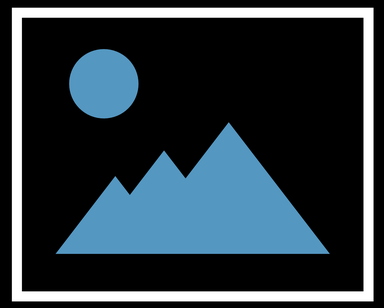


Tabular Data

# Image Classifier

Input

Output

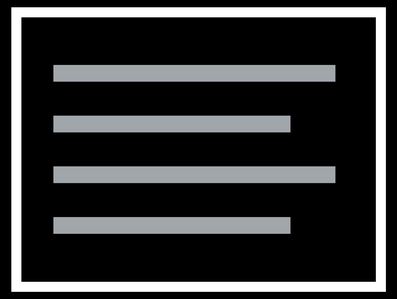


**Label**

# Text Classifier

Input

Output

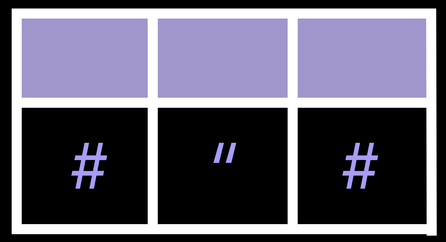


**Label**

# Tabular Data

Input

Output



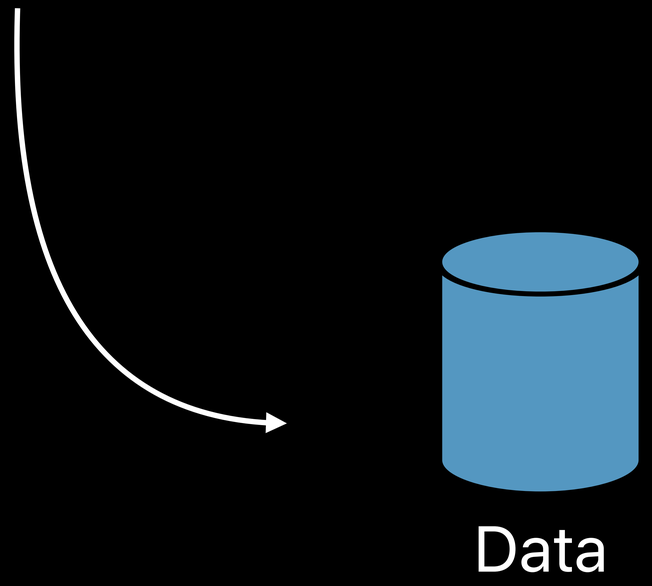

**Number**

# Work Flow

Problem

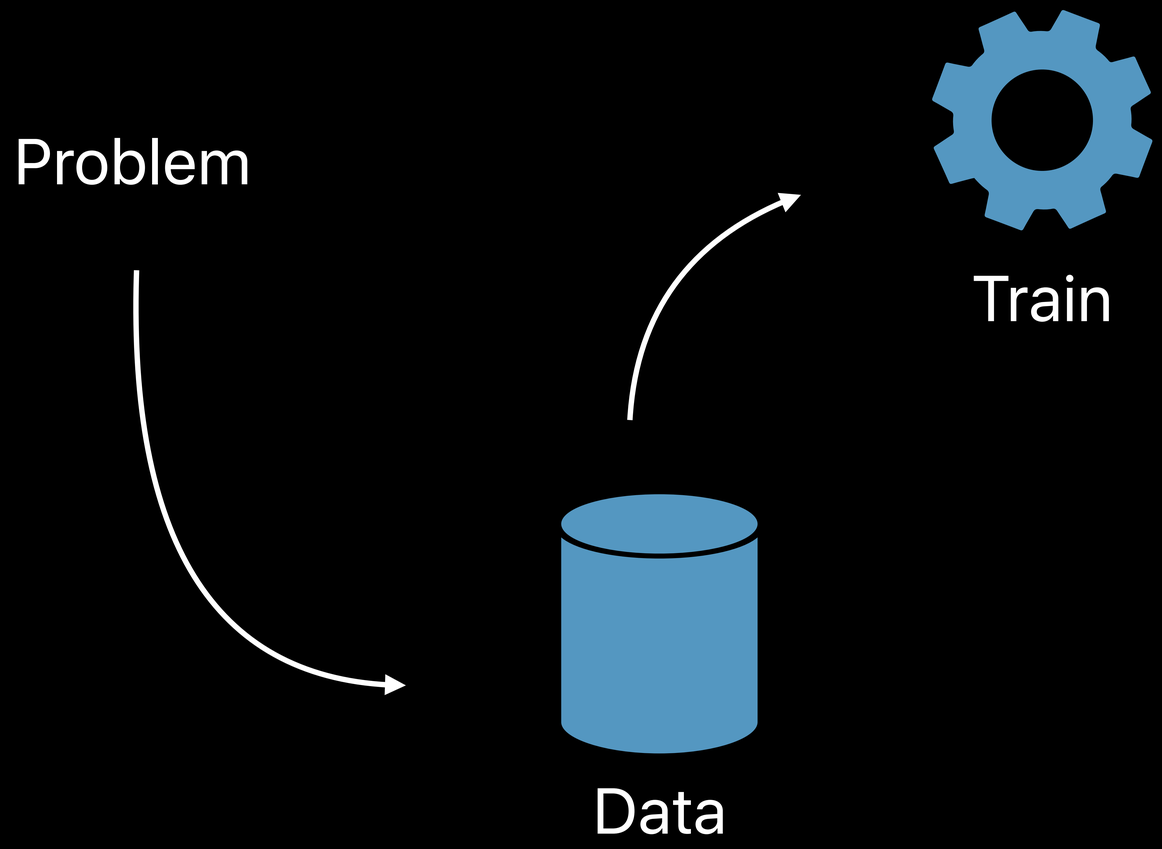
# Work Flow

Problem

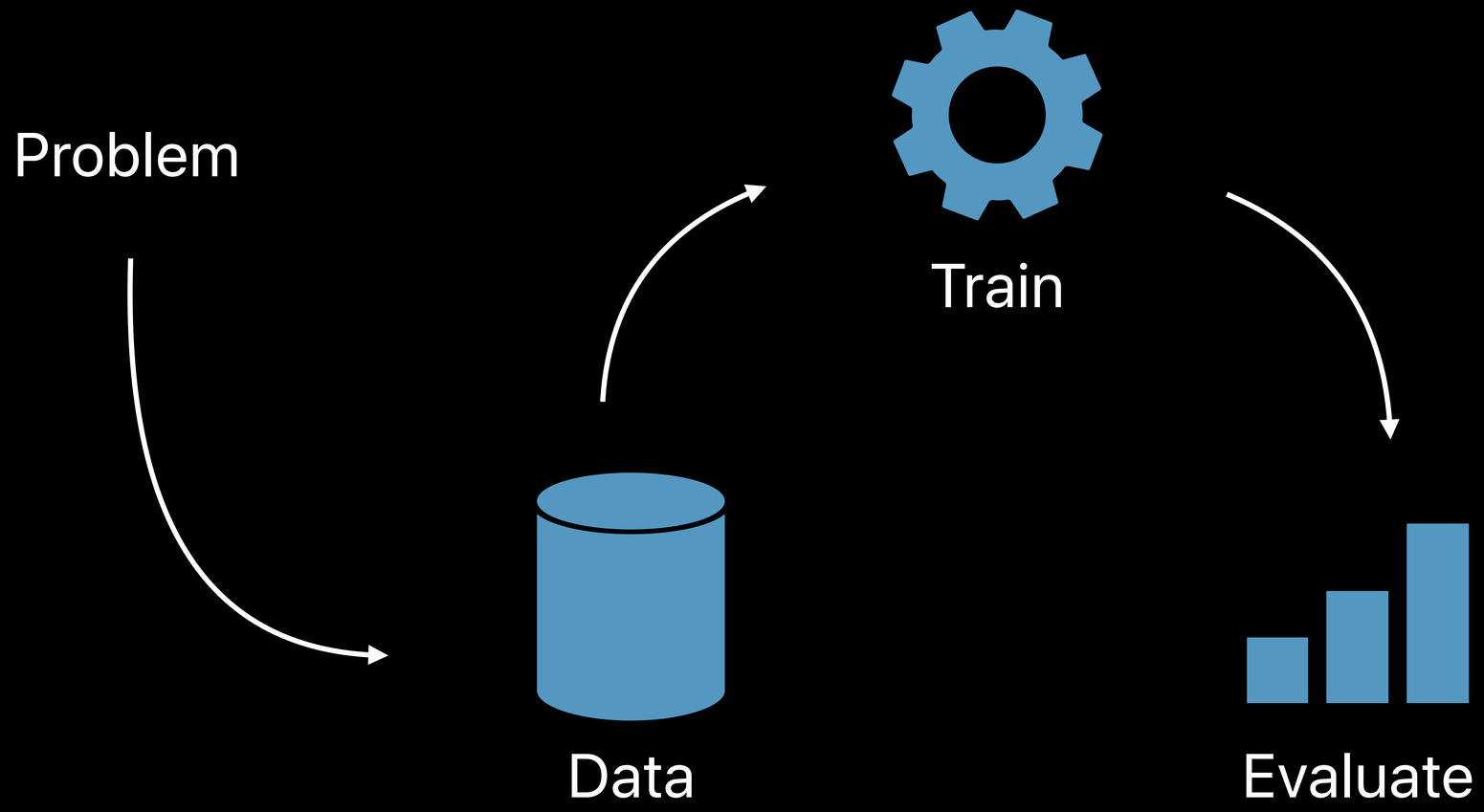


Data

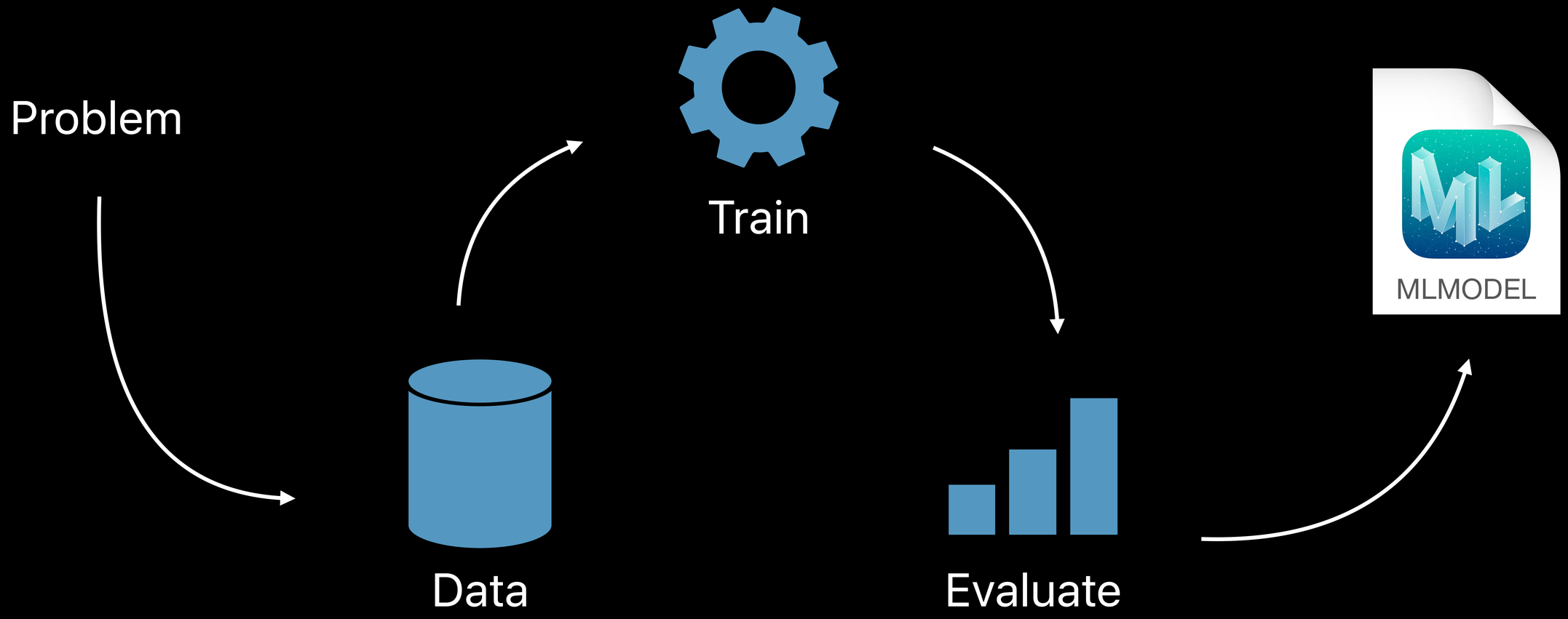
# Work Flow



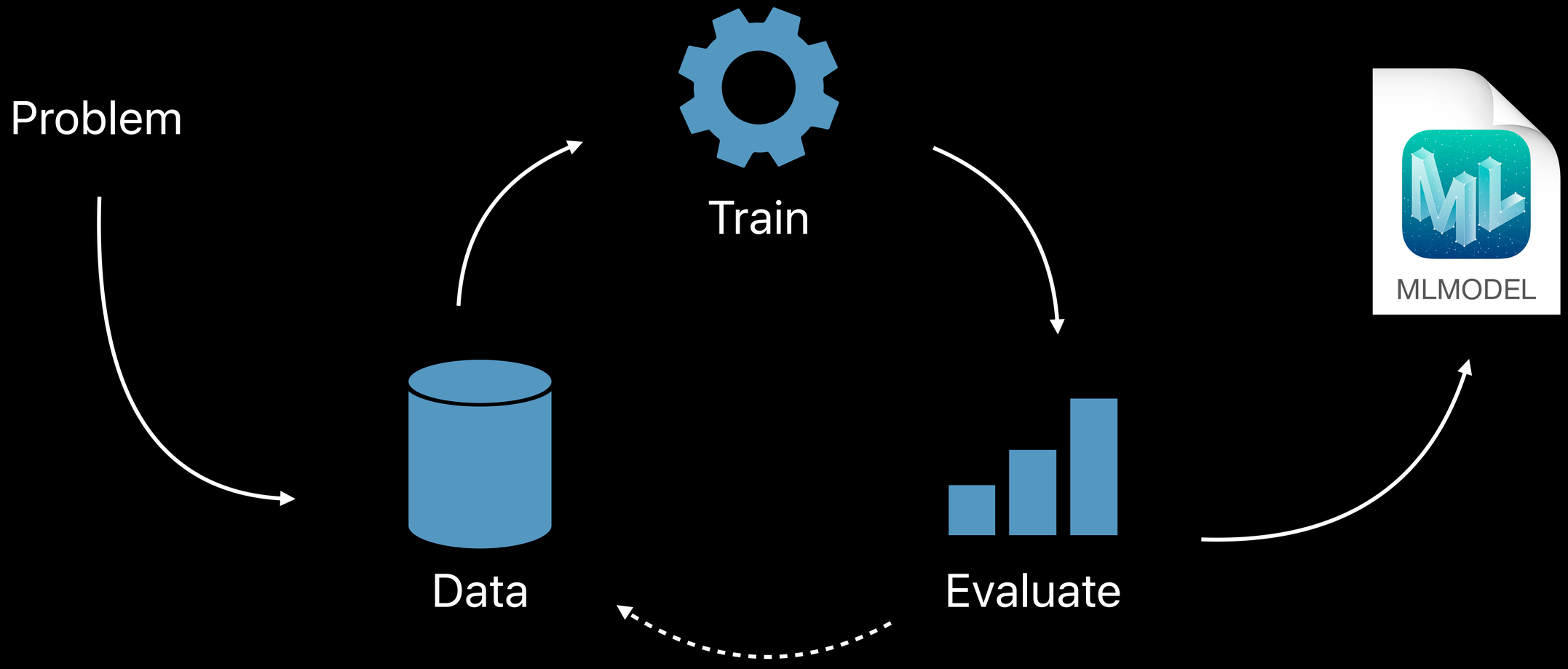
# Work Flow



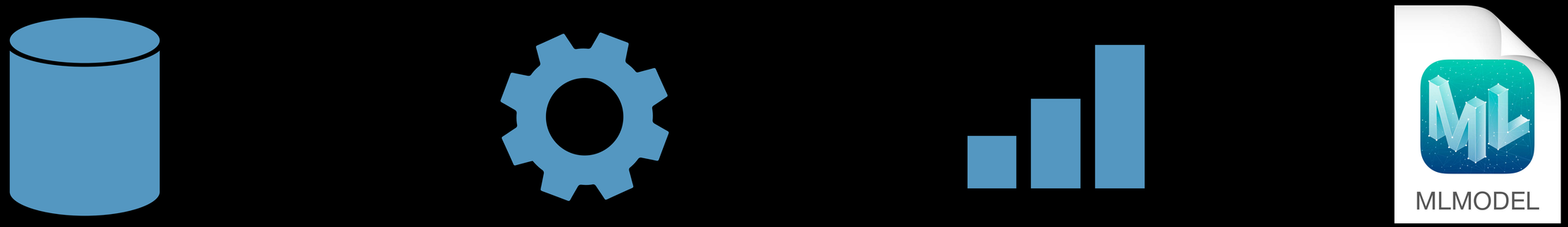
# Work Flow



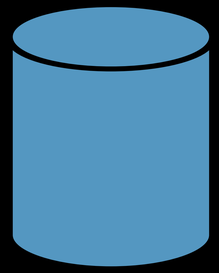
# Work Flow

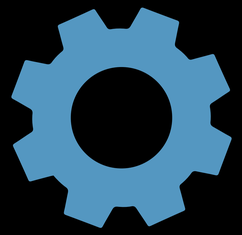


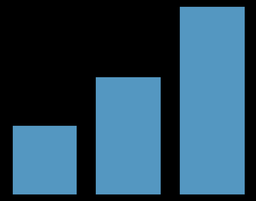
# Work Flow



# Work Flow





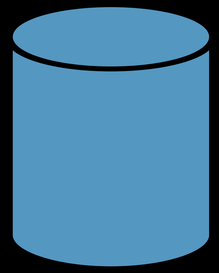



DataSource

Swift Primitives

DataTable

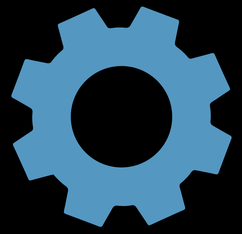
# Work Flow



DataSource

Swift Primitives

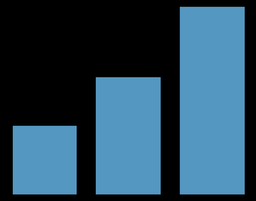
DataTable



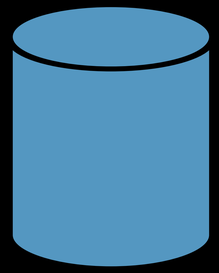
One line of Code

Automatic Selection

Optimized



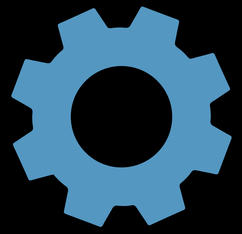
# Work Flow



DataSource

Swift Primitives

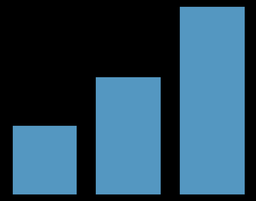
DataTable



One line of Code

Automatic Selection

Optimized

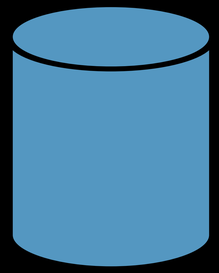


Evaluation Built-in

Classification Metrics

Regression Metrics

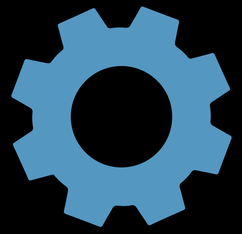
# Work Flow



DataSource

Swift Primitives

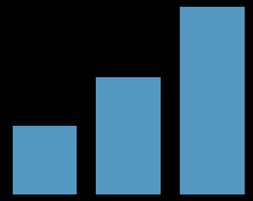
DataTable



One line of Code

Automatic Selection

Optimized

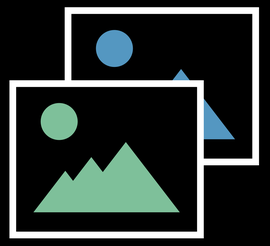


Evaluation Built-in

Classification Metrics

Regression Metrics

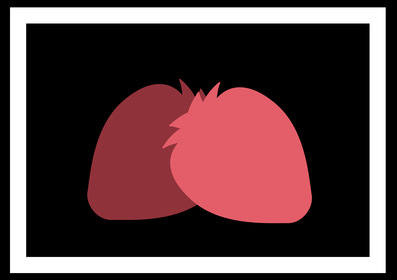

write()



# Images

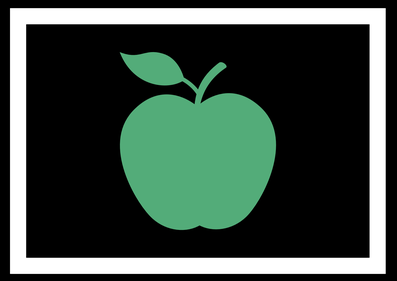
Lizi Ottens, Core ML

# Image Classification

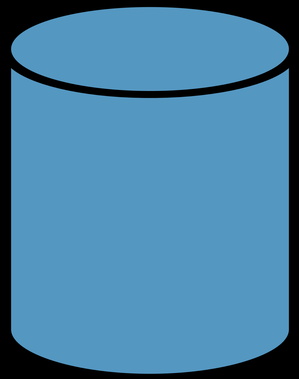


Strawberry

Orange



Apple



Data

Passion Fruit

Blueberry

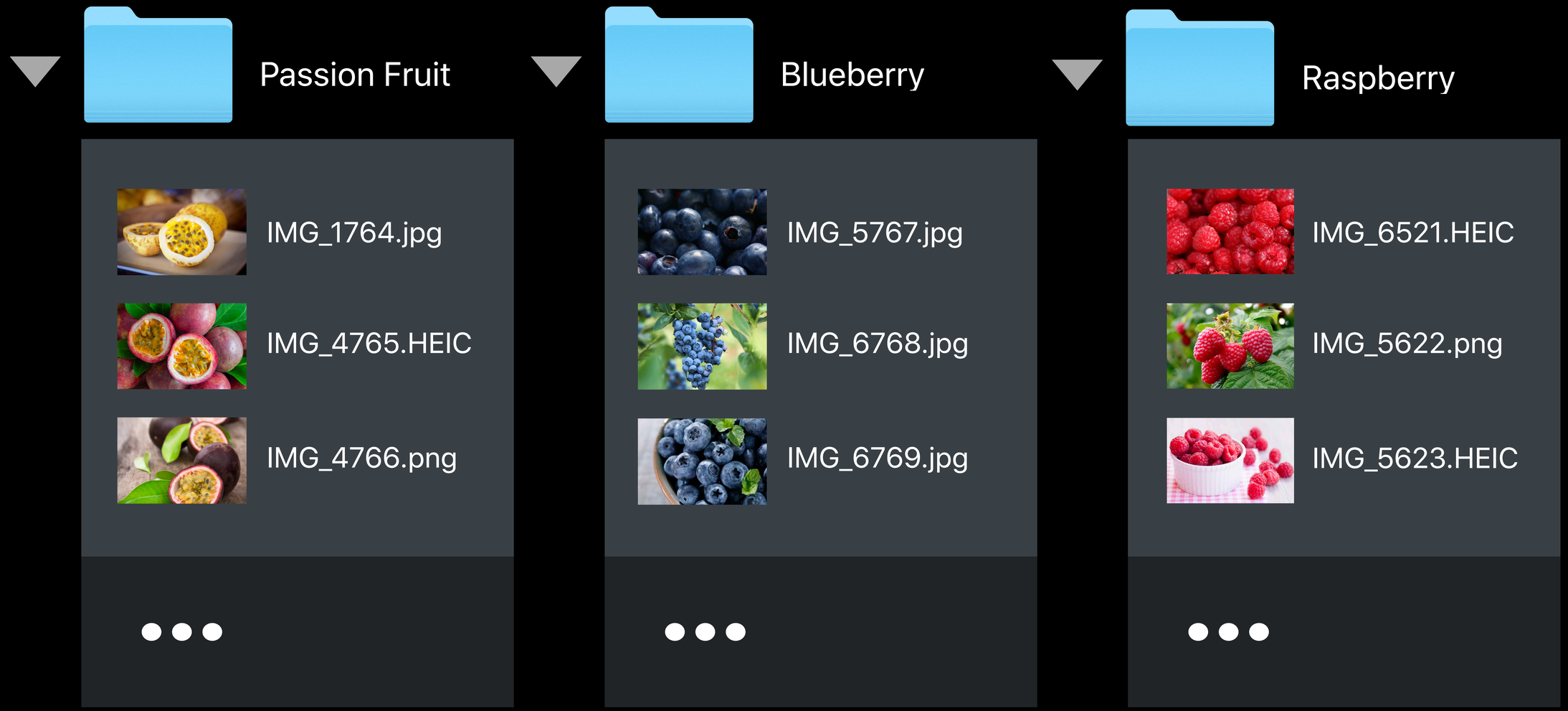
Raspberry

Passion Fruit

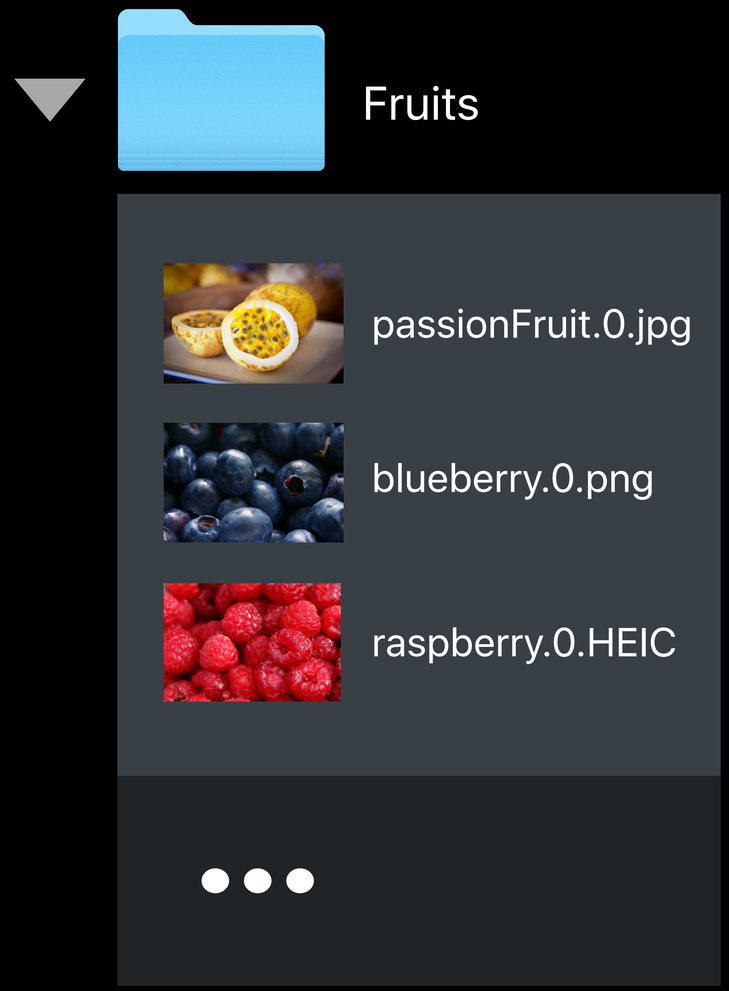
Blueberry

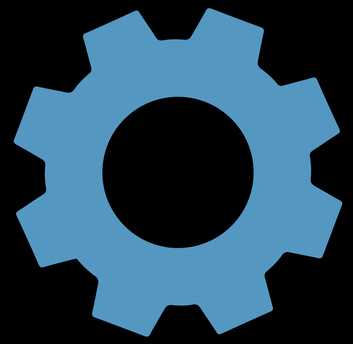
Raspberry

# Data Source

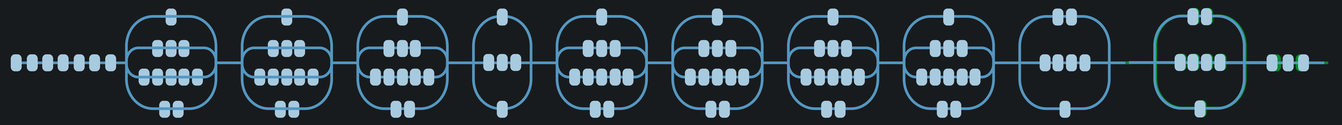

Passion Fruit
IMG_1764.jpg
IMG_4765.HEIC
IMG_4766.png
Blueberry
IMG_5767.jpg
IMG_6768.jpg
IMG_6769.jpg
Raspberry
IMG_6521.HEIC
IMG_5622.png
IMG_5623.HEIC

# Data Source





# Training



# Transfer Learning

# Transfer Learning

# Faster

# Faster

# Hours

# Minutes

10,000 images

# Minutes

10,000 images

# Seconds

## 100 images

# Smaller Models

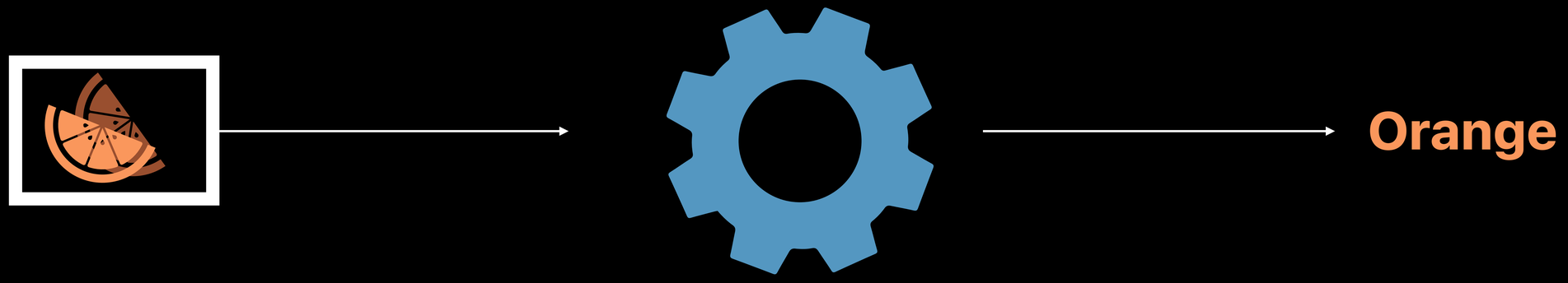
# Smaller Models

# 100s of MB


# MB


# MB

# KB




Vision in Create ML

# *Demo*

Fruit Classifier

# *Demo Recap*

Fruit Classifier

Ready to continue FruitClassifierUI

FruitClassifierUI

```
1  import CreateMLUI
2
3  let builder = MLImageClassifierBuilder()
4  builder.showInLiveView()
```

FruitClassifier...und (Live View)

ImageClassifier

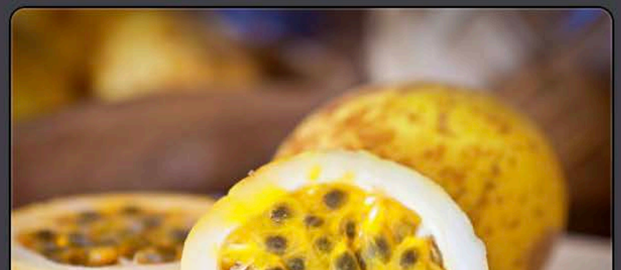
Model accuracy

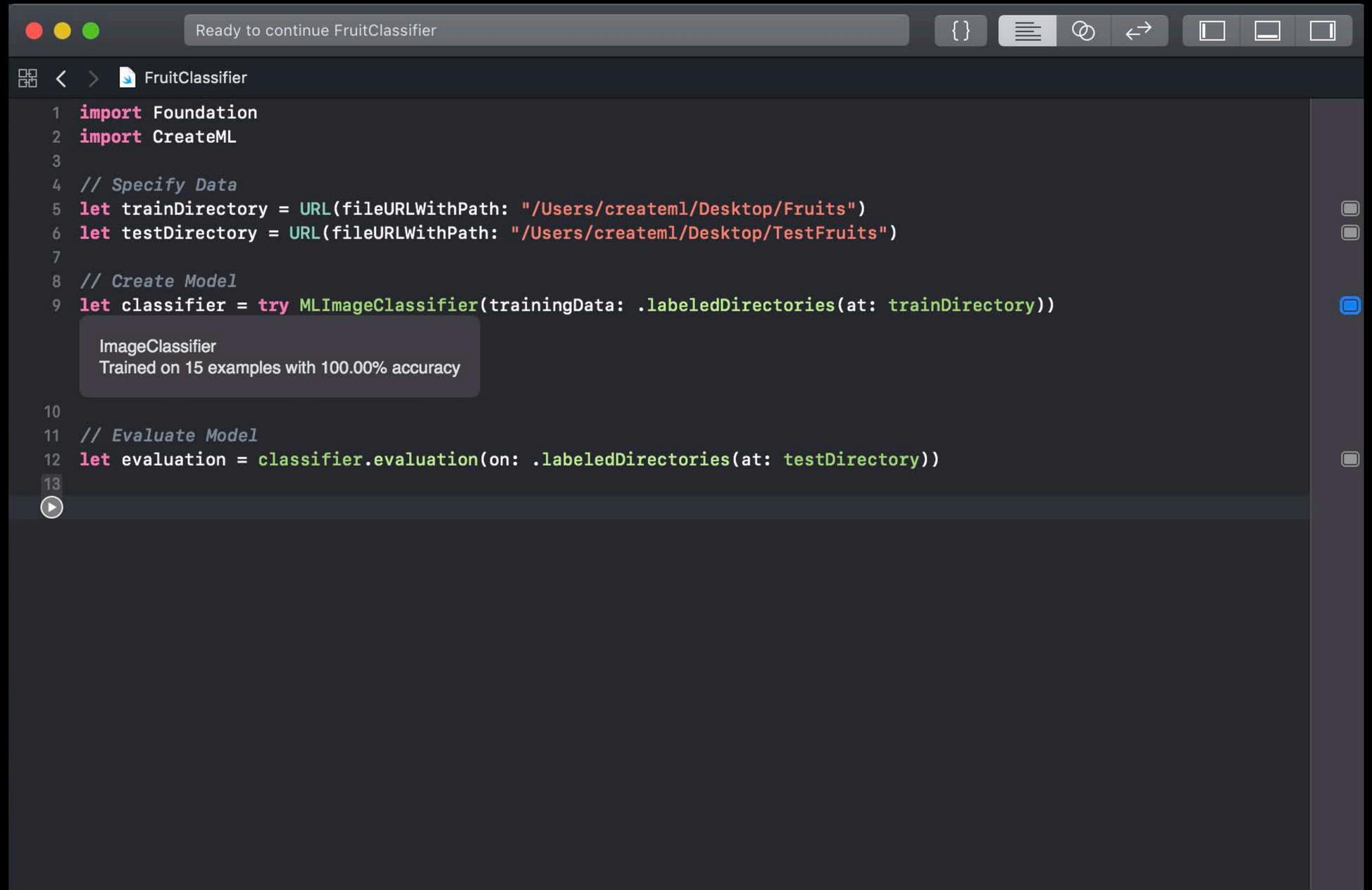
100%
Training

92%
Evaluation

Predicted
Passion Fruit

True
Passion Fruit




Ready to continue FruitClassifier
FruitClassifier
import Foundation
import CreateML
// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/Fruits")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/TestFruits")
// Create Model
let classifier = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))
ImageClassifier
Trained on 15 examples with 100.00% accuracy
// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/Fruits“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/TestFruits“)

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: “/Users/createml/Desktop/FruitClassifier.mlmodel“))
```

```
#!/usr/bin/swift

import Foundation
import CreateML

// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/Fruits")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/TestFruits")

// Create Model
let model = try MLImageClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = model.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
try model.write(to: URL(fileURLWithPath: "/Users/createml/Desktop/FruitClassifier.mlmodel"))
```


Terminal — -bash

```
$ swift fruitClassifier.swift

$ chmod u+x ./fruitClassifier.swift
$ ./fruitClassifier.swift
```


Terminal — -bash

```
$ swift
Welcome to Apple Swift version 4.2.
    1 >  import CreateML
    2 >
```

# Text

Tao Jia, Core ML

| | | |
|---|---|---|
| Introducing Natural Language Framework | Hall 3 | Wednesday 4:00PM |

# Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

**Label**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Label**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Label**

## Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Label

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Label

## Word Tagging

| Lorem | ipsum | dolor | sit | amet |
|---|---|---|---|---|
| Label | Label | Label | Label | Label |

## Text Classification

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

**Label**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Label**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

**Label**

## Word Tagging

| Lorem | ipsum | dolor | sit | amet |
|---|---|---|---|---|
| **Label** | **Label** | **Label** | **Label** | **Label** |

# The energy of developers is amazing!

Sentiment Analysis

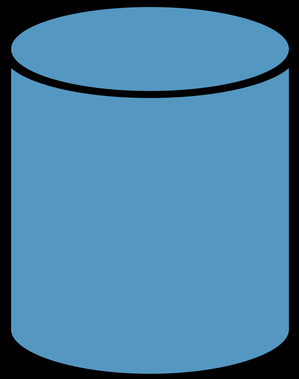
# Deposed Prince wants to give you money SPAM

Spam Analysis

# The Warriors just had an amazing comeback win

# SPORT

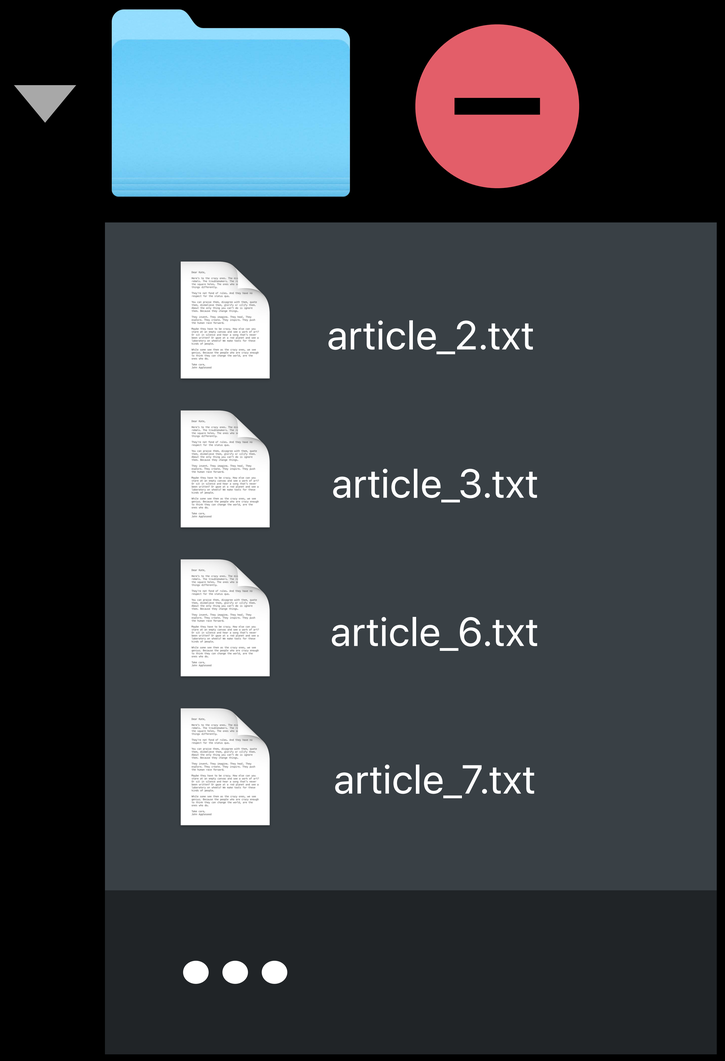
Topic Analysis



Data

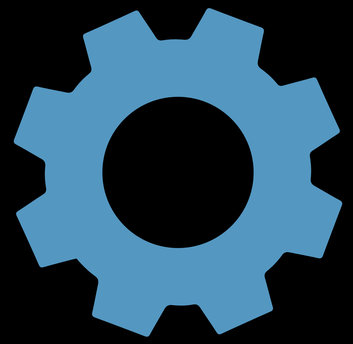
# Data Source


article_1.txt
article_4.txt
article_5.txt
article_8.txt


article_2.txt
article_3.txt
article_6.txt
article_7.txt


QueriesWithDomainLabels.csv

```
I’m looking for a place to stay in Barcelona,HOTELS
Where can I get good Mexican food on a Sunday?,RESTAURANTS
Find me an inexpensive round trip flight to London.,FLIGHTS
…
```



# Training

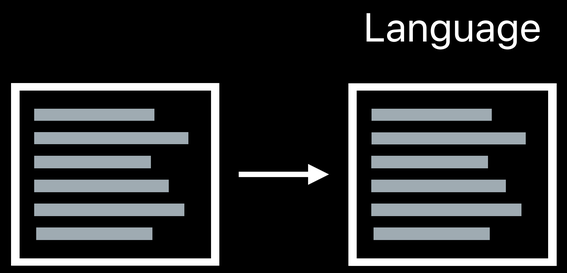
# Text Classification

(Typical work flow)

**Label**

# Text Classification

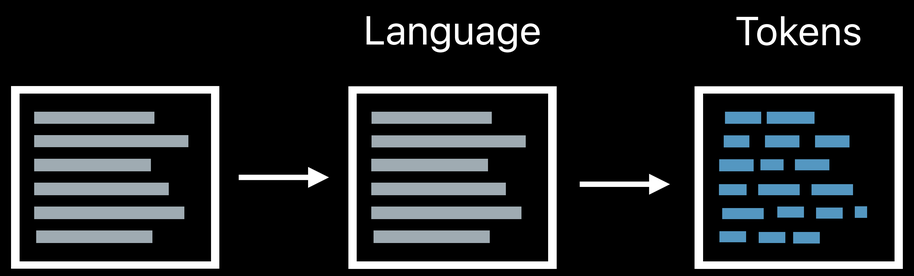
(Typical work flow)




**Label**

# Text Classification

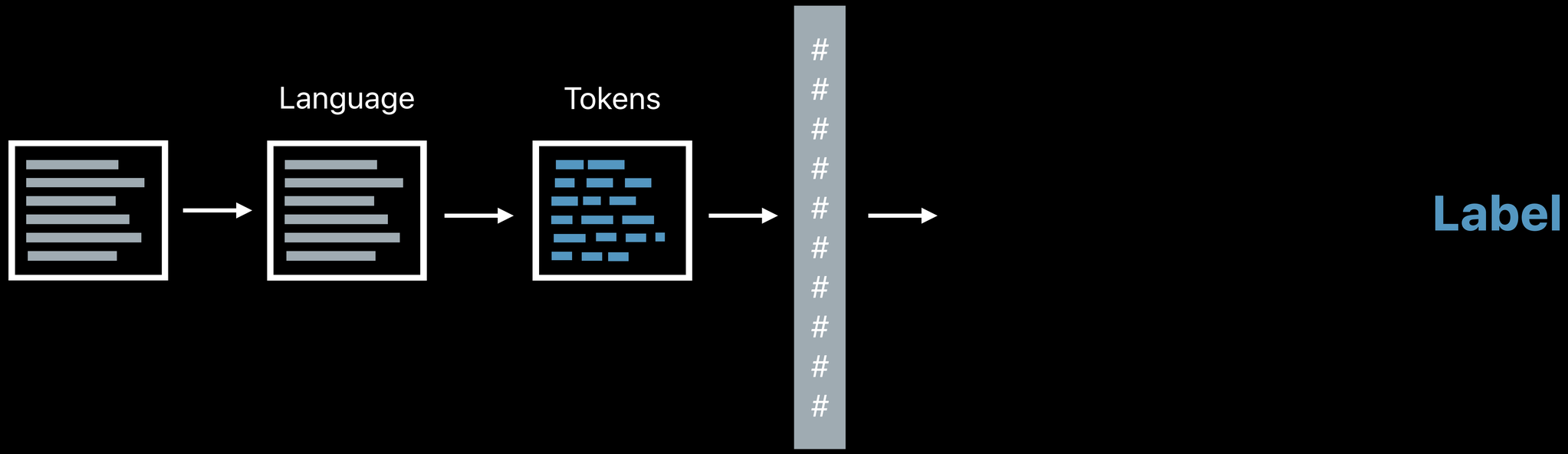
(Typical work flow)




**Label**

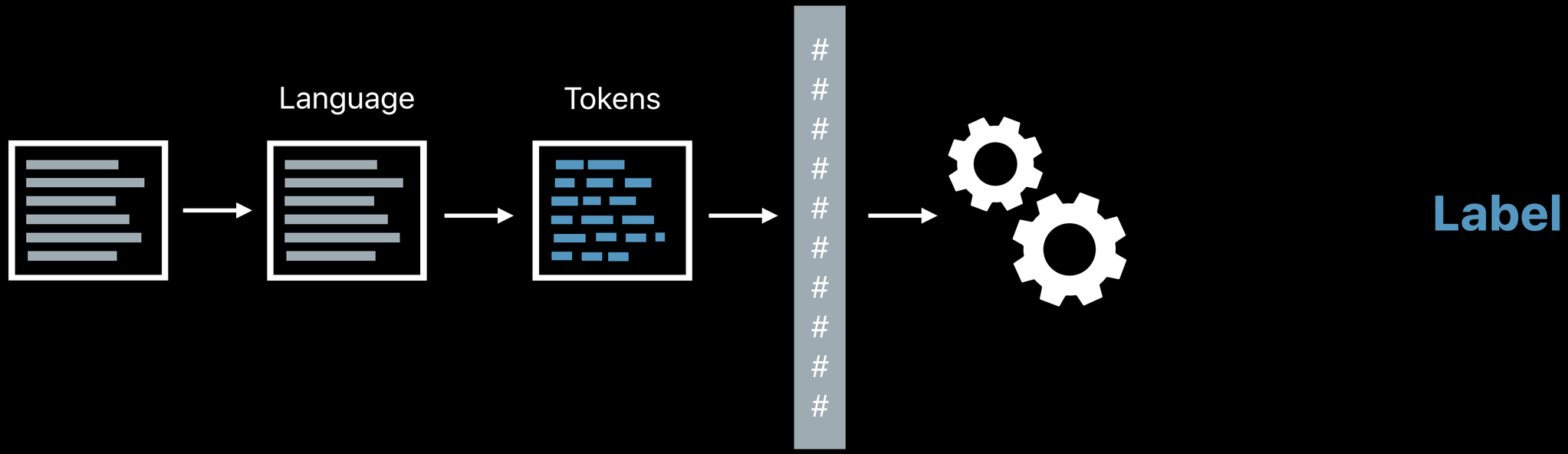
# Text Classification

(Typical work flow)



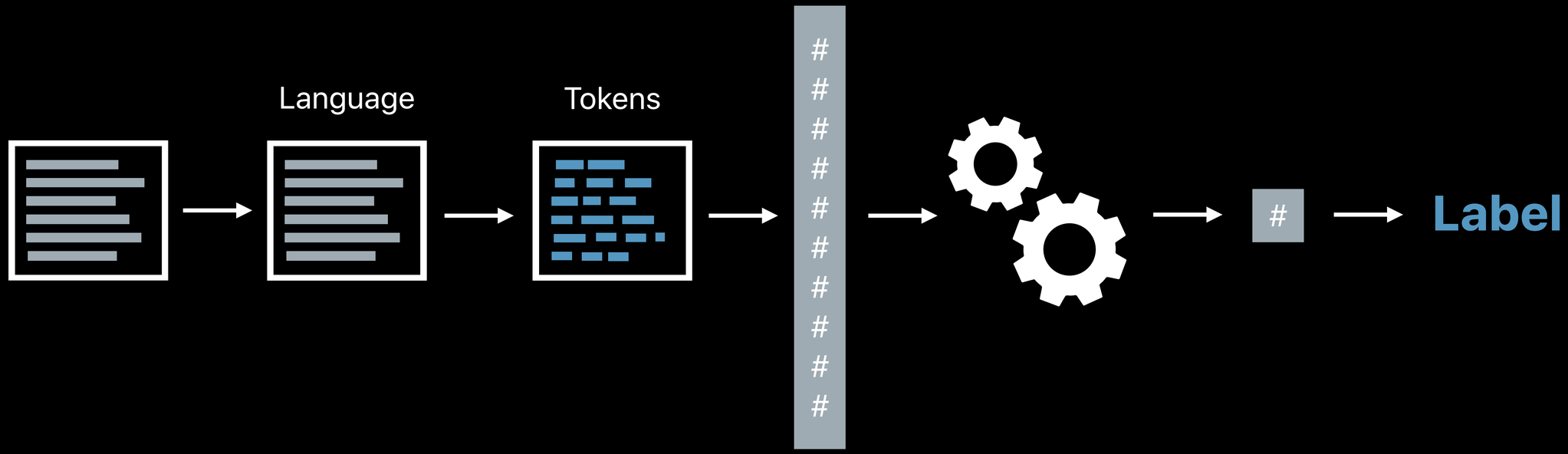
# Text Classification

(Typical work flow)



# Text Classification

(Typical work flow)



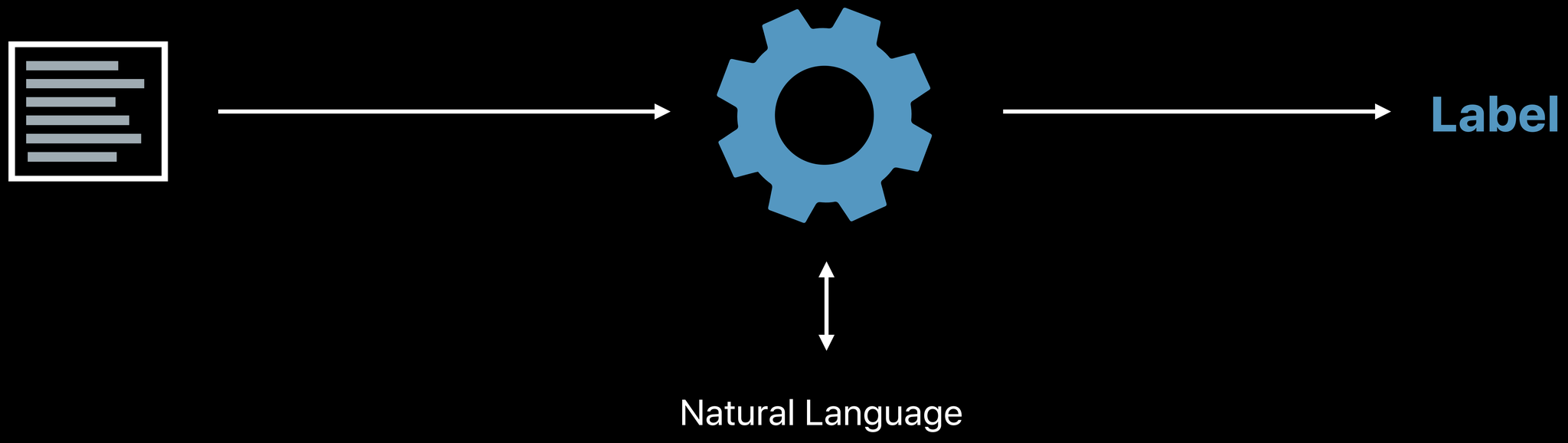
# Text Classification with Create ML

| | | |
|---|---|---|
| Introducing Natural Language Framework | Hall 3 | Wednesday 4:00PM |

# Text Classification with Create ML

Label

# Text Classification with Create ML



# Example

# Stay Positive

Encouraging positive posts

# Stay Positive

Encouraging positive posts

# Stay Positive

Encouraging positive posts

# *Demo*

Text Classifier

# *Demo Recap*

Text Classifier

```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/train“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/test“)

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: “/Users/createml/Desktop/TextClassifier.mlmodel“)
try classifier.write(to: modelPath)
```

```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/train")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/test")

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: "/Users/createml/Desktop/TextClassifier.mlmodel")
try classifier.write(to: modelPath)
```

```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/train")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/test")

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: "/Users/createml/Desktop/TextClassifier.mlmodel")
try classifier.write(to: modelPath)
```

```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/train")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/test")

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: "/Users/createml/Desktop/TextClassifier.mlmodel")
try classifier.write(to: modelPath)
```

```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/train")
let testDirectory = URL(fileURLWithPath: "/Users/createml/Desktop/test")

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: "/Users/createml/Desktop/TextClassifier.mlmodel")
try classifier.write(to: modelPath)
```

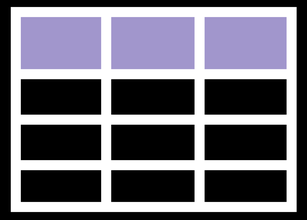
```
import CreateML
import Foundation

// Specify Data
let trainDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/train“)
let testDirectory = URL(fileURLWithPath: “/Users/createml/Desktop/test“)

// Create Model
let classifier = try MLTextClassifier(trainingData: .labeledDirectories(at: trainDirectory))

// Evaluate Model
let evaluation = classifier.evaluation(on: .labeledDirectories(at: testDirectory))

// Save Model
let modelPath = URL(fileURLWithPath: “/Users/createml/Desktop/TextClassifier.mlmodel“)
try classifier.write(to: modelPath)
```



# Tabular Data

Gaurav Kapoor, Core ML

Predict house price using

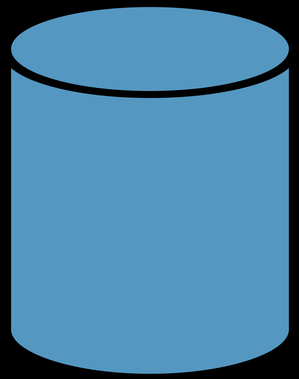
- No. of bedrooms
- No. of baths
- Location
- Sq. ft
- Lot size

Wine Quality using

- Acidity
- Sugar
- pH
- Alcohol
- Citric

Where to go tonight

- Rating
- Price
- Happy hour
- Distance



Data

# Tabular Data

`MLDataTable`

| beds | baths | squareFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

# Tabular Data

`MLDataTable`

Example

| beds | baths | squareFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

# Tabular Data

`MLDataTable`

Feature

| beds | baths | squareFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

# Tabular Data

`MLDataTable`

Feature

| beds | baths | squareFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

# Tabular Data

`MLDataTable`

| beds | baths | squareFt | Price (Target) |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

# Common Sources

## Files


```
beds,baths,squareFt,price
2,2,2000,400000
4,3,2500,500000
3,2,1800,450000
3,2,1500,300000
```


```
[ {“beds”: 2,
   “baths”: 2,
   “squareFt”: 2000,
   “price” : 400000},
   …
]
```

## Code

```
func loadData() ->
[String:[Any]]
```

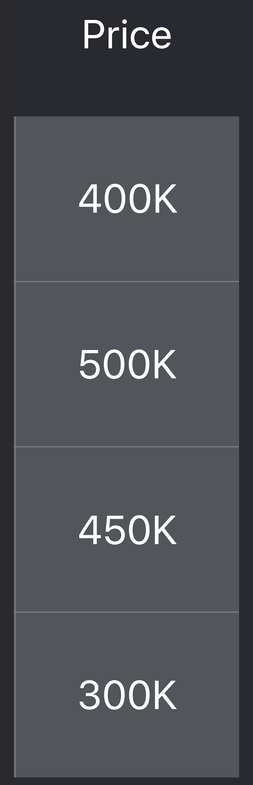
| bedrooms | bathrooms | sqFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

```
// You can read data
let houseData = try MLDataTable(contentsOf: mycsv)
```

| bedrooms | bathrooms | sqFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

```
// You can read data
let houseData = try MLDataTable(contentsOf: mycsv)

// You can access the Column
houseData[“price”]
```



```
// You can add, subtract, multiply, divide two columns
```

| sqFt | Price |
| --- | --- |
| 2000 | 400K |
| 2500 | 500K |
| 1800 | 450K |
| 1500 | 300K |

```
// You can add, subtract, multiply, divide two columns
let pricePerSqft = houseData["price"] / houseData["sqft"]
```

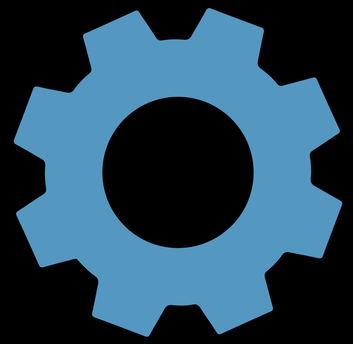
| sqFt | Price | pricePerSqFt |
|---|---|---|
| 2000 | 400K | 200 |
| 2500 | 500K | 200 |
| 1800 | 450K | 250 |
| 1500 | 300K | 200 |

```
// You can filter on a column
```

| bedrooms | bathrooms | sqFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |
| 3 | 2 | 1800 | 450K |
| 3 | 2 | 1500 | 300K |

```
// You can filter on a column
let indexOfBigHouses = houseData["sqft"] >= 2000
let largeHouseData = houseData[indexOfBigHouses]
```

| bedrooms | bathrooms | sqFt | Price |
|---|---|---|---|
| 4 | 2 | 2000 | 400K |
| 4 | 3 | 2500 | 500K |



# Training

Decision Tree Classifier

Random Forest Classifier

Boosted Tree Classifier

Logistic Regression

Support Vector Machines

Boosted Trees Regression

Linear Regression

Decision Tree Regression

Random Forest Regression

MLDecisionTreeClassifier

MLRandomForestClassifier

MLBoostedTreeClassifier

MLLogisticRegressionClassifier

MLSupportVectorClassifier

MLBoostedTreeRegressor

MLLinearRegressor

MLDecisionTreeRegressor

MLRandomForestClassifier

```
// You can make a model
let classifier = try          MLLinearRegressor(trainingData: houseData, targetColumn: “price”)
```

```
// You can make a model
let classifier = try    MLBoostedTreeRegressor(trainingData: houseData, targetColumn: “price”)
```

```
// You can make a model
let classifier = try   MLRandomForestRegressor(trainingData: houseData, targetColumn: “price”)
```

```
// You can make a model
let classifier = try                    (trainingData: houseData, targetColumn: “price”)
```

```
// You can make a model
let classifier = try   MLRegressor(trainingData: houseData, targetColumn: “price”)
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

```
import CreateML

// Specify Data
let trainingCSV = URL(fileURLWithPath: “/Users/createml/HouseData.csv”)
let houseData = MLDataTable(contentsOf: trainingCSV)
let (trainingData,testData) = houseData.randomSplit(by: 0.8, seed: 0)

// Create Model
let pricer = try MLRegressor(trainingData: houseData, targetColumn: “price")

// Evaluate Model
let metrics = try pricer.testingMetrics(on: testData)

// Save Model
try pricer.write(to: URL(fileURLWithPath: “/Users/createml/HousePricer.mlmodel”))
```

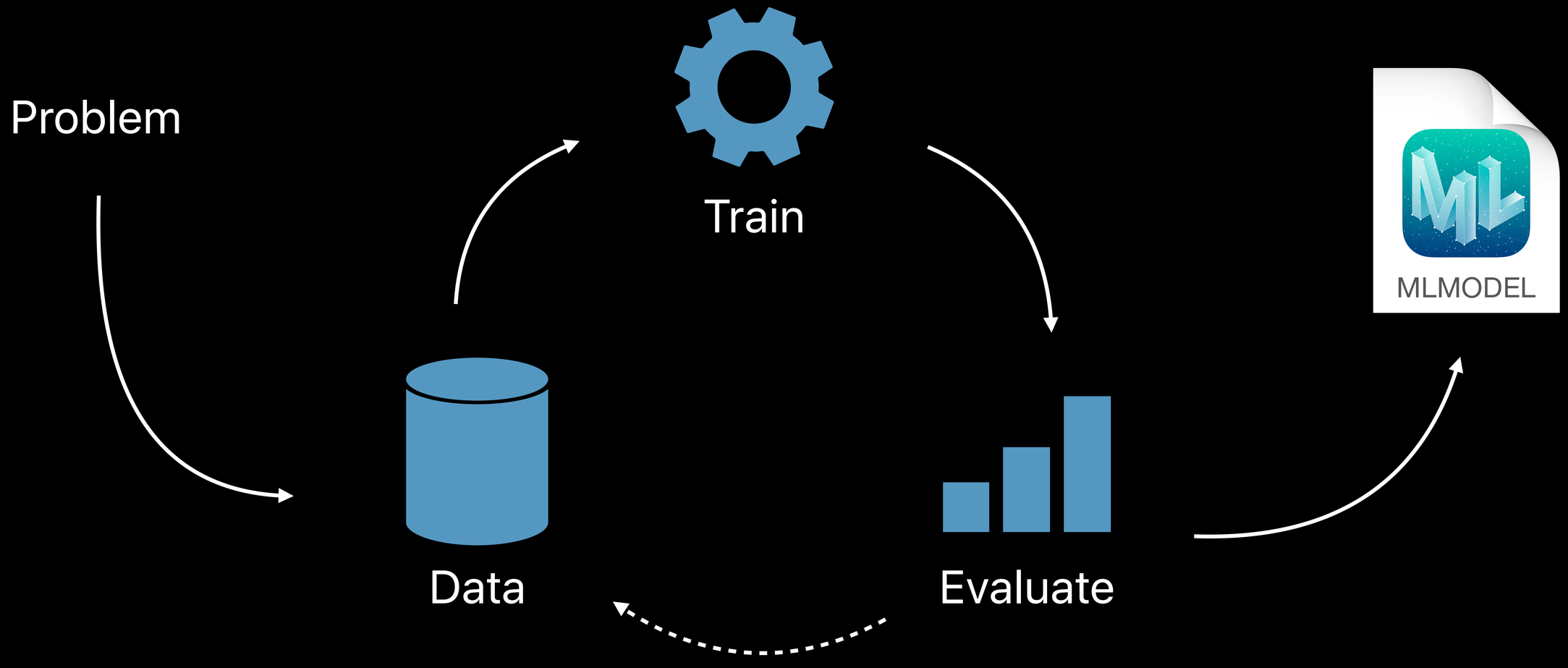
# Summary


ML framework in Swift

Simple and Powerful

Leverages core Apple technologies

# Work Flow



Available on **macOS Mojave**


PLAYGROUND


SWIFT


>_

Try it out

Machine Learning Get Together

Labs

Related Sessions in the WWDC App

# More Information

https://developer.apple.com/wwdc18/703

| | | |
|---|---|---|
| Machine Learning Lab | Technology Lab 2 | Wednesday 4:00PM |
| Machine Learning Lab | Technology Lab 12 | Friday 2:00PM |


WWDC18