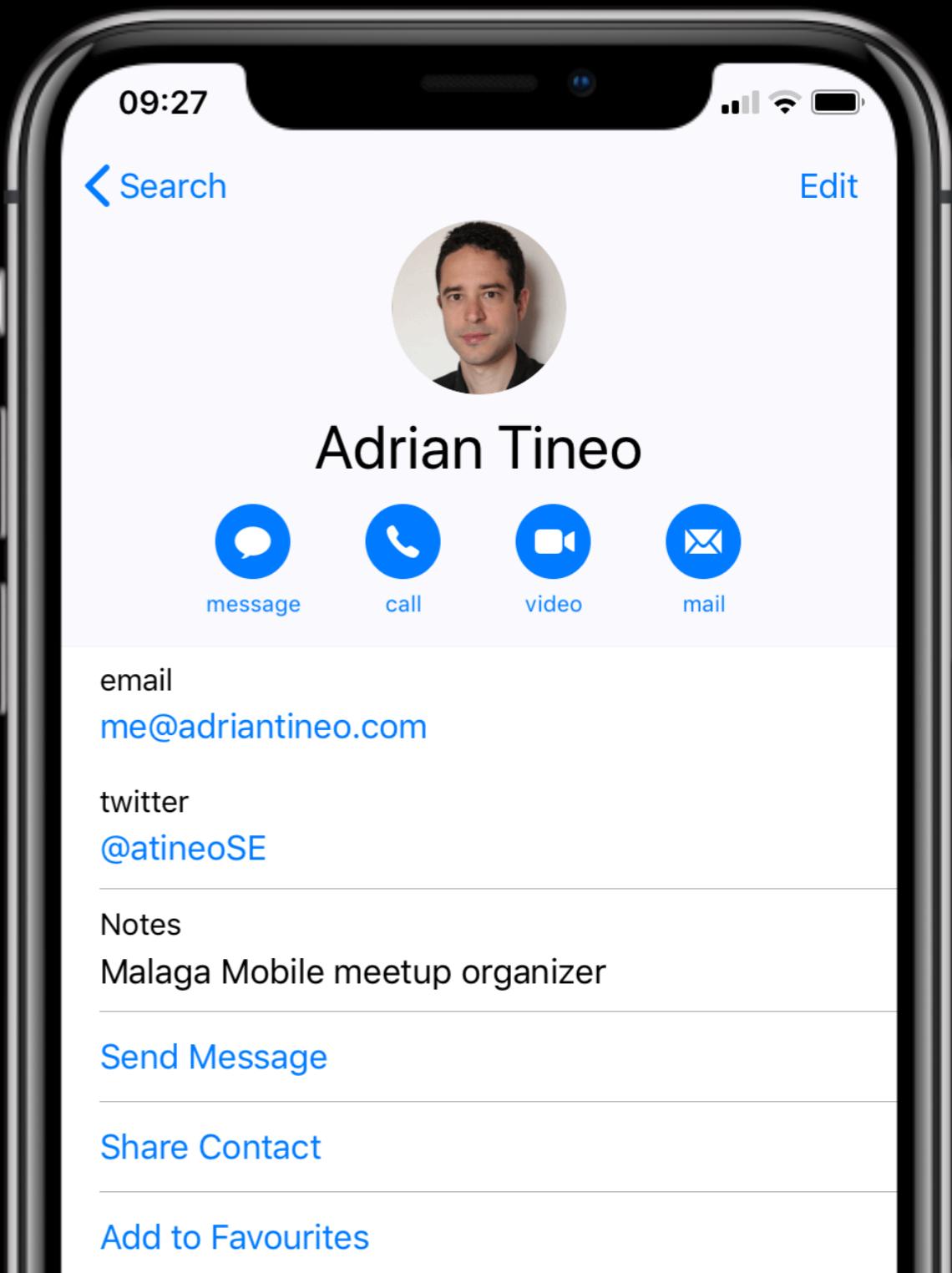


Enabling your iOS apps with ML: opportunities and challenges

Adrian Tineo

About me

- Freelance iOS Consultant
- Ph.D. in CS
- Background:
 - research scientist
 - parallel computing
 - test automation



Core ML is announced at WWDC'17



Lots of possible applications

Sentiment Analysis

That was totally
awesome Leo! → 😊

Handwriting Recognition

7 → 7

Translation

I love you mom → 사랑해 엄마

Scene Classification



Style Transfer



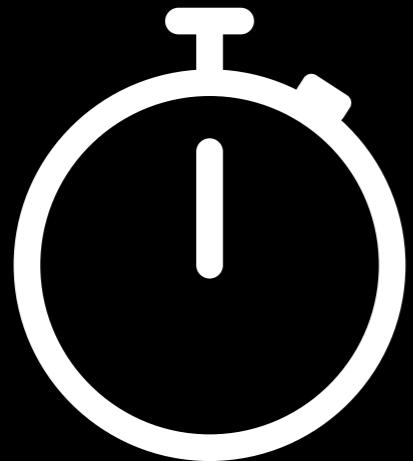
Music Tagging



Predicting Text

Do you know the way to → San Jose

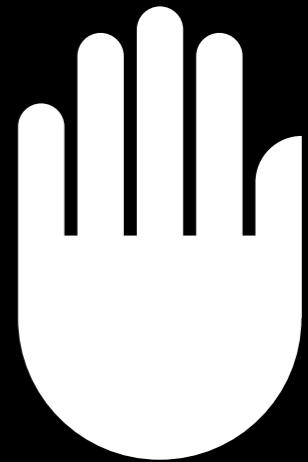
Model runs on device



Performance



Server costs



Privacy

Initial workflow



+

Caffe

dmlc
XGBoost

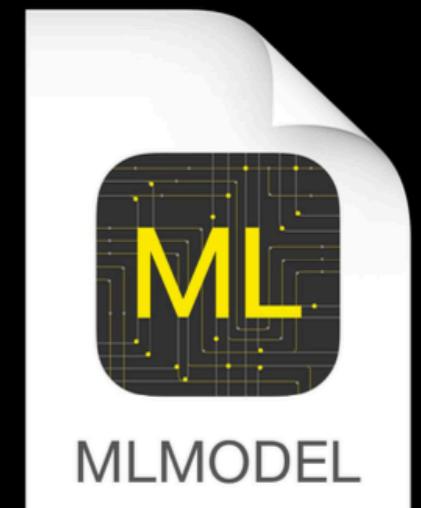
turi 

K Keras

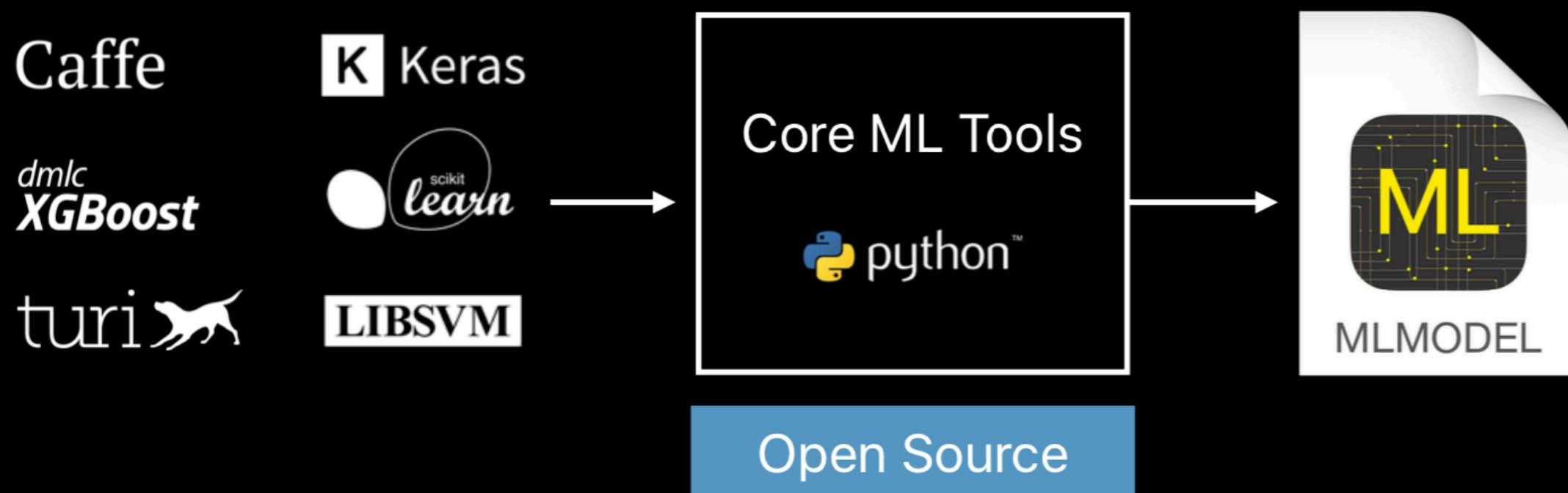


LIBSVM

Convert
→



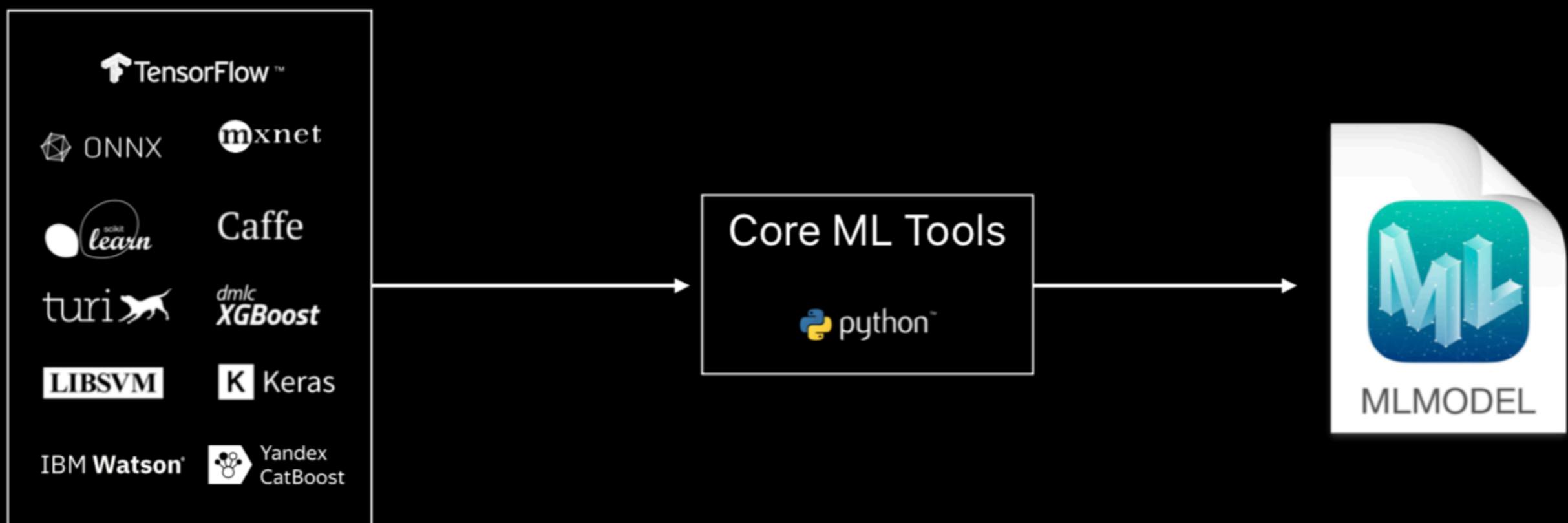
Model conversion tool



Fast-forward to WWDC'18



More formats are supported



Create ML is announced

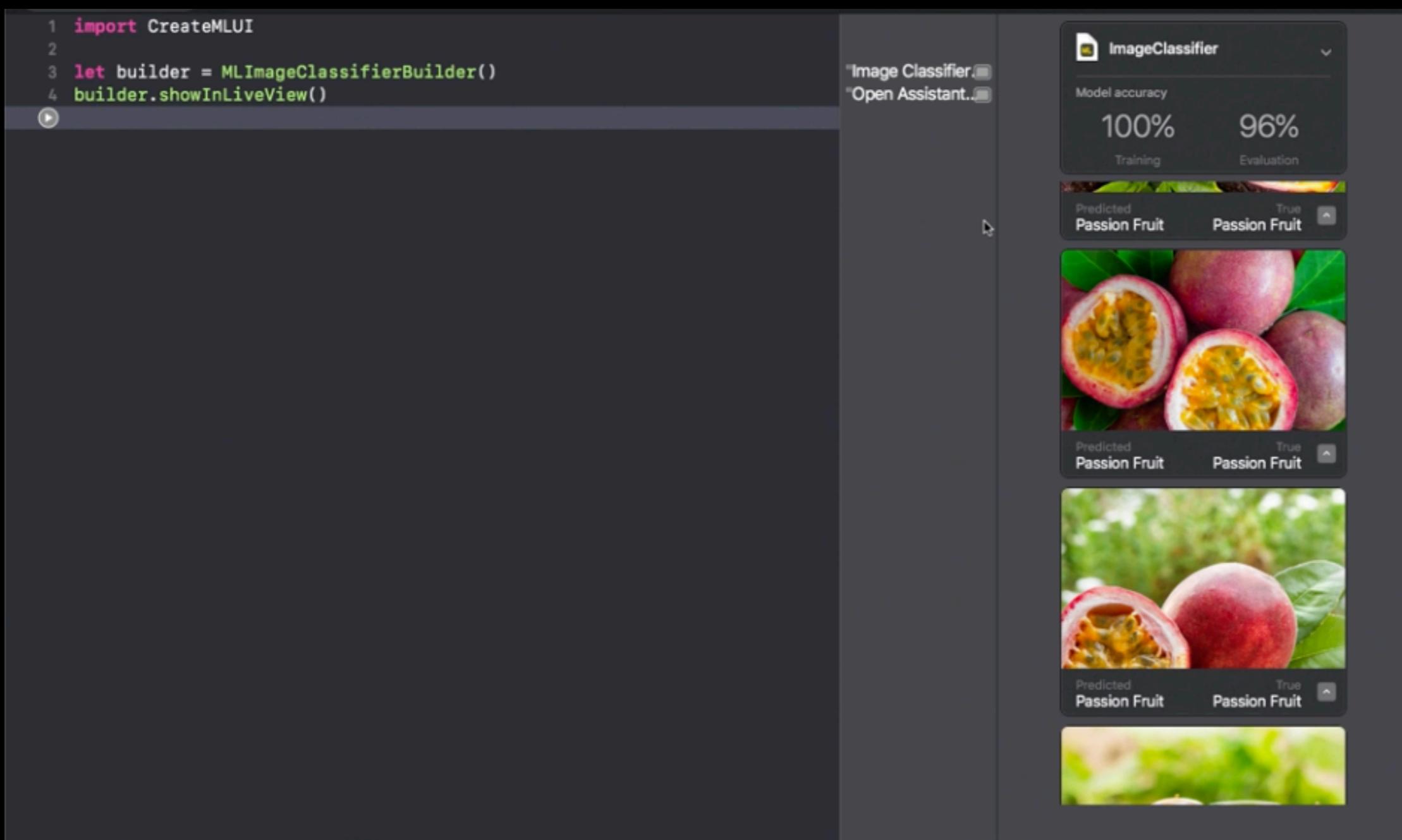


Alternative workflow



All within the Apple developer ecosystem (i.e. Xcode)

Train from a playground



Three major cases supported



Images



Text



Tabular Data

Is it really so easy?



For common cases



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 ⓘ](#)



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 ⓘ](#)



Places205-GoogLeNet

Detects the scene of an image from 205 categories such as an airport terminal, bedroom, forest, coast, and more.

[View original model details >](#)

[Download Core ML Model ⓘ](#)



ResNet50

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 ⓘ](#)



Inception v3

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 ⓘ](#)



VGG16

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 ⓘ](#)

How about some challenging problems?



Competitions

Documentation InClass

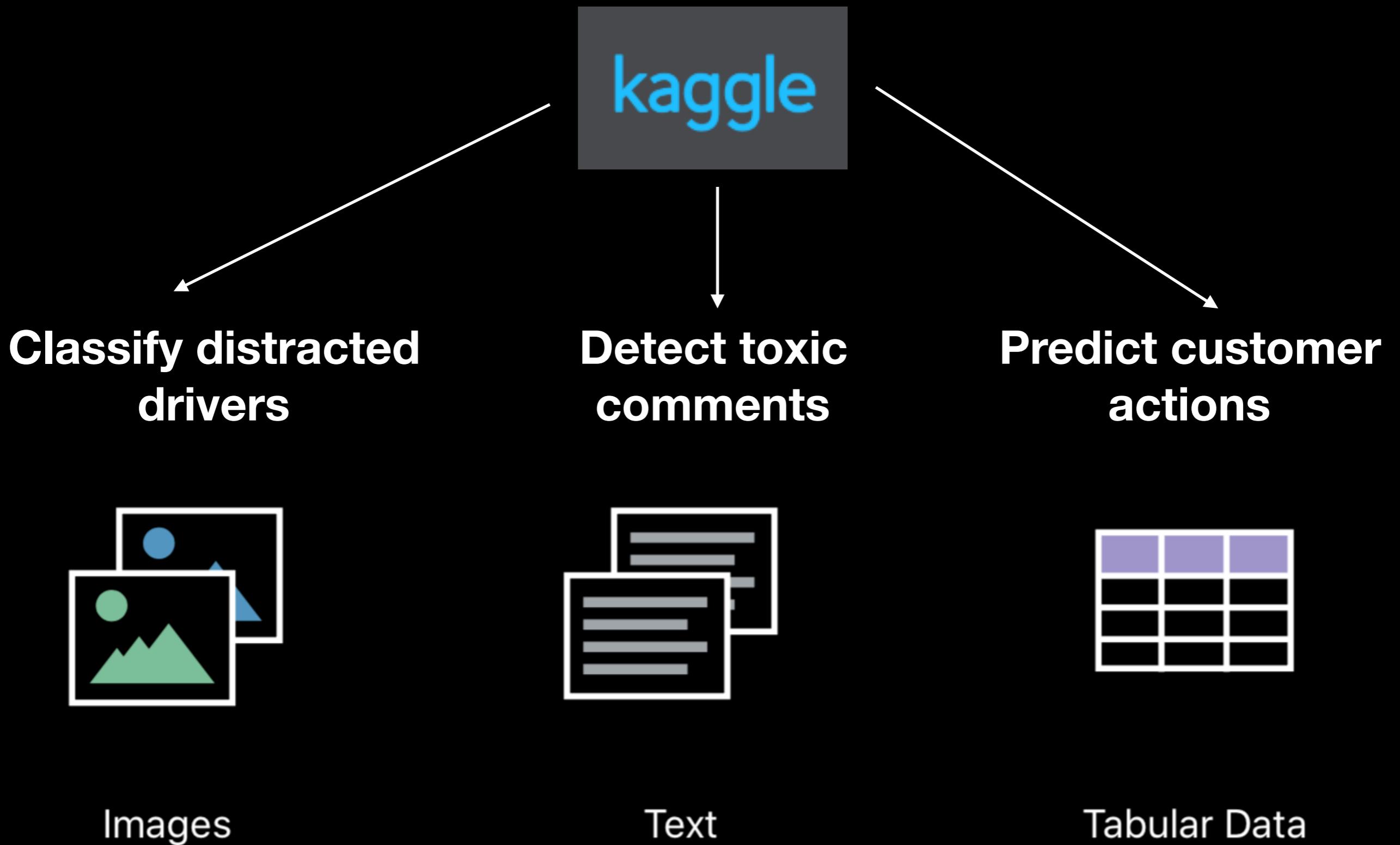
General InClass Sort by Grouped

Featured Search competitions

4 Active Competitions

Competition	Description	Prize	Teams
Two Sigma: Using News to Predict Stock Movements Use news analytics to predict stock price performance <small>Featured · Kernels Competition · 2 months to go · news agencies, time series, finance, money</small>	\$100,000	2,927 teams	
Jigsaw Unintended Bias in Toxicity Classification Detect toxicity across a diverse range of conversations <small>Featured · Kernels Competition · a month to go · biases, nlp, text data</small>	\$65,000	2,395 teams	
Predicting Molecular Properties Can you measure the magnetic interactions between a pair of atoms? <small>Featured · 3 months to go · chemistry, tabular data, regression</small>	\$30,000	190 teams	
Instant Gratification A synchronous Kernels-only competition <small>Featured · Kernels Competition · 21 days to go · tabular data, binary classification</small>	\$5,000	866 teams	

Three problems



Direct approach



Multidisciplinary approach

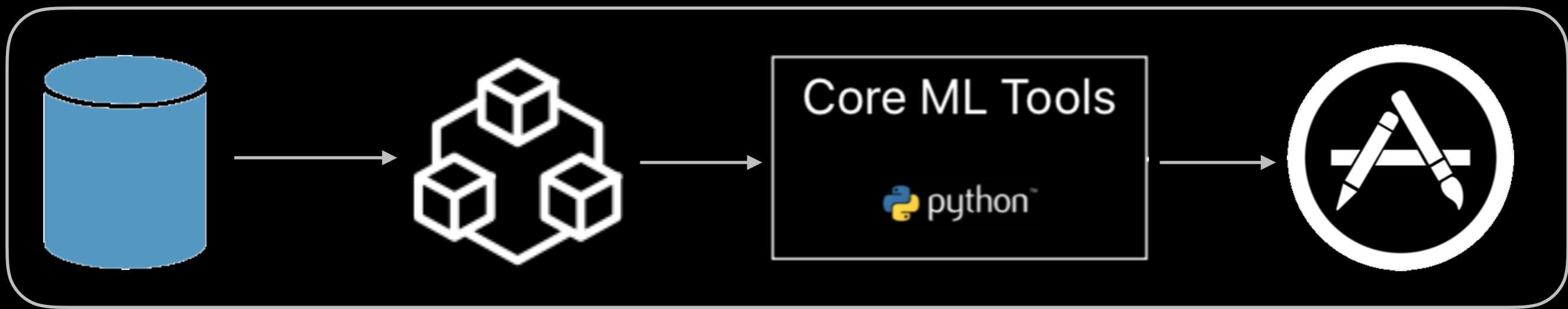


Image Classification



State Farm Distracted Driver Detection

State Farm™ Can computer vision spot distracted drivers?
\$65,000 · 1,440 teams · 3 years ago

[Overview](#) [Data](#) [Kernels](#) [Discussion](#) [Leaderboard](#) [Rules](#) [Team](#) [My Submissions](#) [Late Submission](#)

Overview

Description We've all been there: a light turns green and the car in front of you doesn't budge. Or, a previously unremarkable vehicle suddenly slows and starts swerving from side-to-side.

Evaluation When you pass the offending driver, what do you expect to see? You certainly aren't surprised when you spot a driver who is texting, seemingly enraptured by social media, or in a lively hand-held conversation on their phone.

Prizes

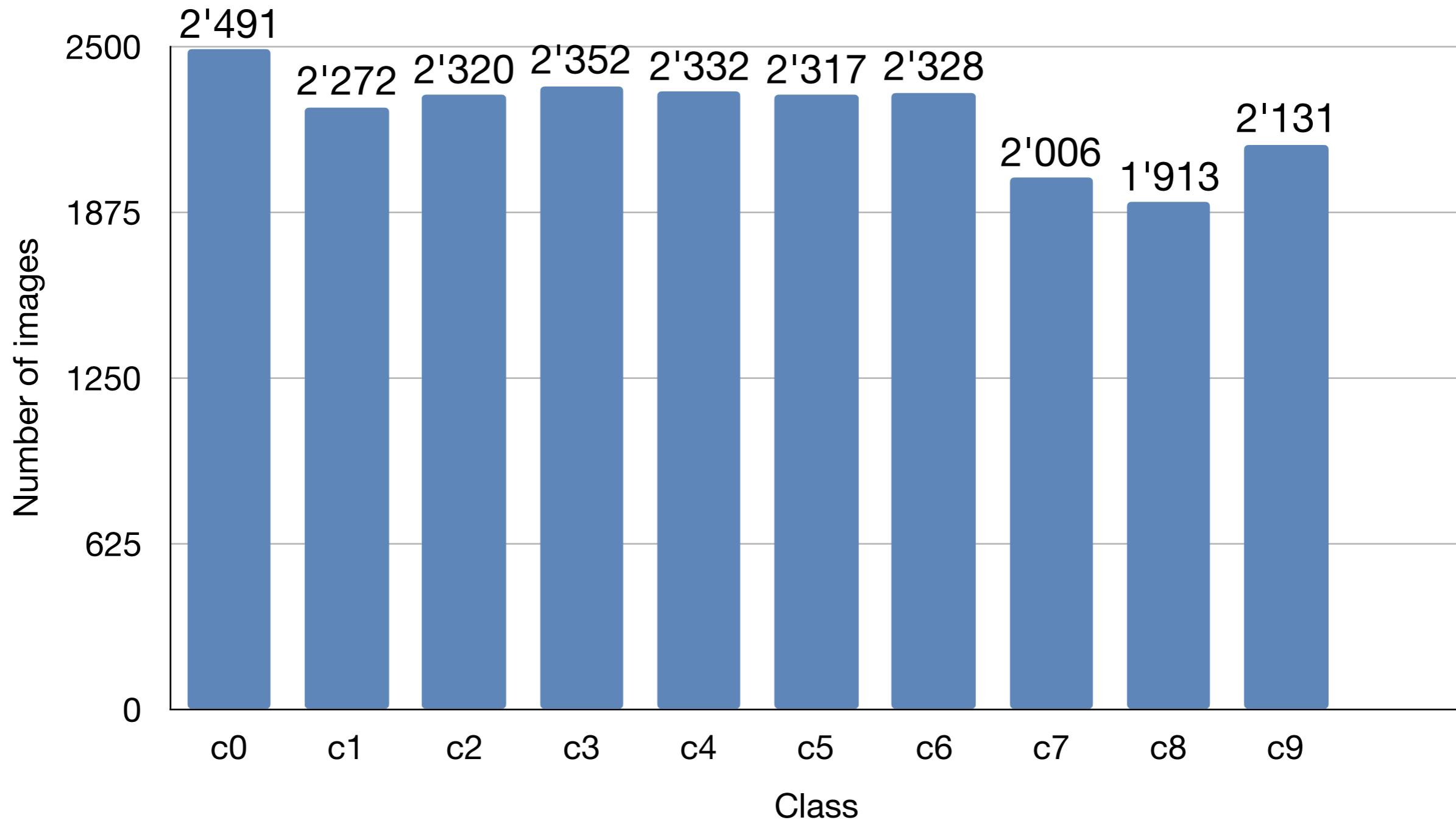
Timeline



Training data distribution

22,462 images

Distracted Driver Training Set Class Distribution



Distracted Driver Dataset



c0 - safe driving



c1 - texting right



c2 - talking right



c3 - texting left



c4 - talking left



c5 - radio



c6 - drinking



c7 - reach behind



c8 - grooming



c9 - talking

Create image classifier



```
// Define parameters for training
let parameters = MLImageClassifier.ModelParameters(featureExtractor: .scenePrint(revision: 1),
    validationData: nil,
    maxIterations: 150,
    augmentationOptions: [] )
```

A large red arrow points from the 'parameters' variable in the first code block to the 'parameters' variable in the second code block.

```
// Train classifier
let classifier = try? MLImageClassifier(trainingData: trainingData,
    parameters: parameters)
```

Optimal solution



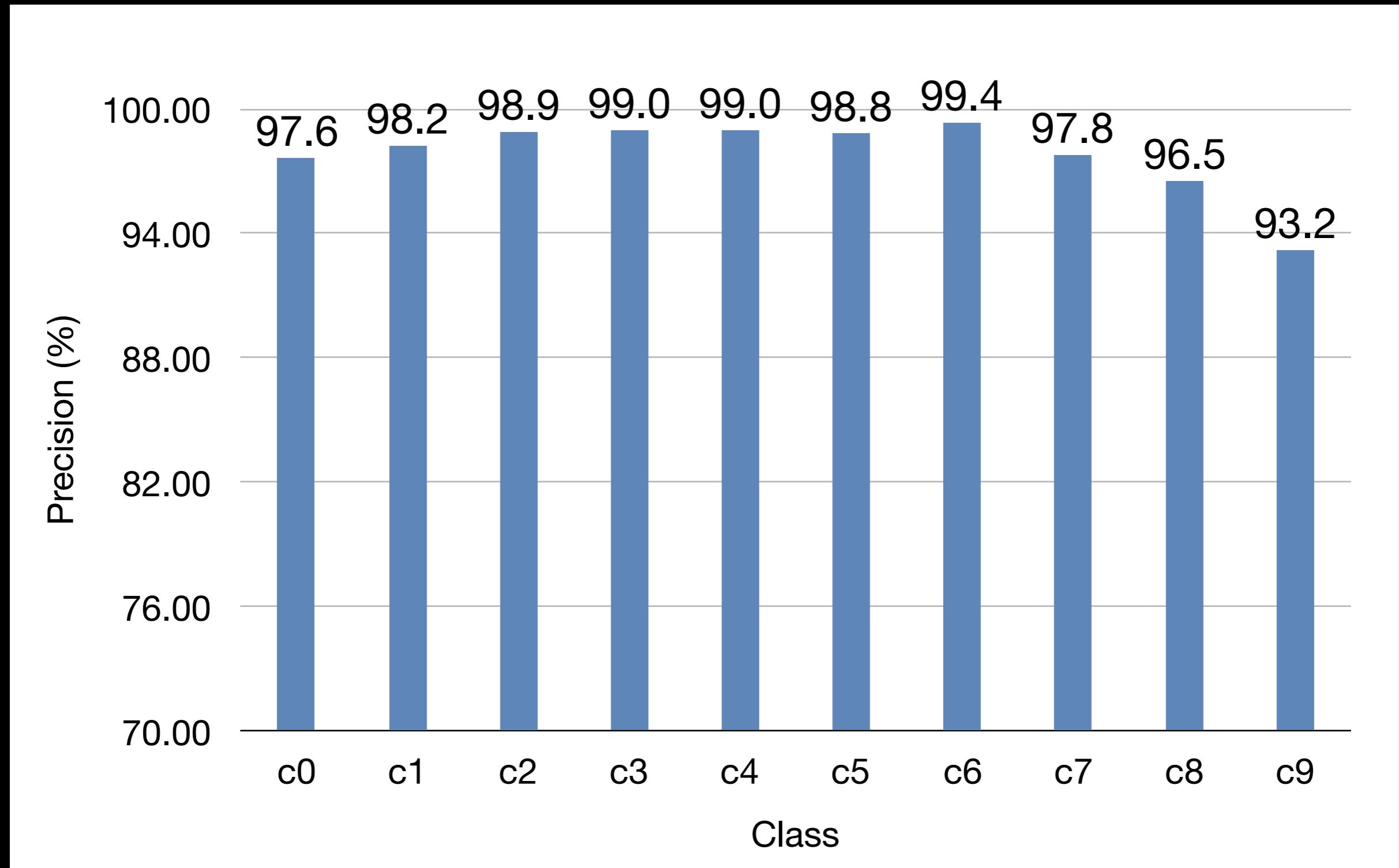
1000 it

85 min

Iteration	Elapsed Time	Training Accuracy	Validation Accuracy
...			
710	911.891504	1.000000	0.976035
715	917.239272	1.000000	0.976035
720	923.040021	1.000000	0.977124
725	928.810305	1.000000	0.977124
730	934.036429	1.000000	0.977124
735	941.578091	1.000000	0.977124

SUCCESS: Optimal solution found.

Classifier precision



Model size



178 Kb

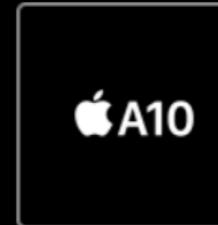
Performance on device



iPhone XR

Neural engine
8 cores

Avg 33 images/second



iPhone 7

No dedicated HW
4 cores

Avg 8 images/second

Measured over ~19,000 images

Accuracy on device



Fails 909 out of 4'467 sample images (20%)



Accuracy on device



▼ Model Evaluation Parameters			
Name	Type	Flexibility	Description
▼ Inputs			
image	Image Color 299 x 299)	299... x 299...	Input image to be classified
▼ Outputs			
classLabelProbs	Dictionary (String → Double)		Probability of each category
classLabel	String		Most likely image category

Accuracy on device



`request.imageCropAndScaleOption`



Dataset image (c7)
640x480



.centerCrop
299x299



.scaleFill
299x299



.scaleFit
299x299

Accuracy on device



- Several failed strategies
 - Change crop and scale option -> same result
 - Add flexible sizes to the model -> same result
 - Try to manipulate images manually -> crashes
 - Train the model with augmentation (cropped images) -> worse precision

Model size

K Keras + TensorFlow™



8 hours training time



4 epochs

111.5 MB

Accuracy: 95%

iMac Late 2013, Intel Core i7, 4 cores, 16 GB

TensorFlow built with CPU vectorized operations (AVX2)

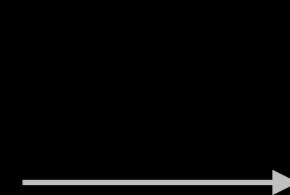
Convert model



```
from coremltools.converters.keras import convert  
from keras.models import load_model
```

```
keras_model = load_model("./Model/distracted_driver.h5")  
coreml_model = convert(keras_model, image_input_names = "input1")  
coreml_model.save("./Model/DistractedDriverKeras.mlmodel")
```

K Keras



Core ML Tools
python™



111.5 MB

37.1 MB

Accuracy on device



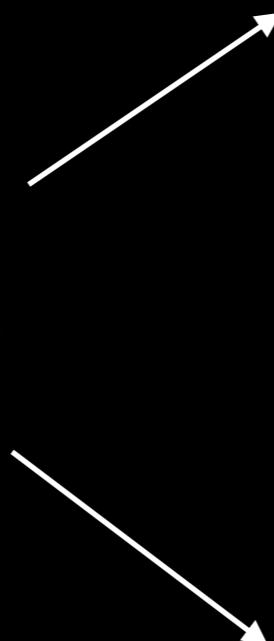
Fails 2'352 out of 4'467 sample images (+50%)



Image resizing



Dataset image (c7)
640x480



Default

.centerCrop
360x480



.scaleFill/.scaleFit
360x480

Accuracy on device



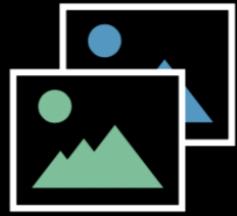
```
kerasClassificationRequest.imageCropAndScaleOption = .scaleFill
```

Fails 404 out of 4'467 sample images (~9%)





VS.



Precision:
98% in lab
80% on device

Precision:
95% in lab
91% on device

Training time:
1.4 hours

Training time:
8 hours

Performance:
iPhone 7: 8 sec/img
iPhone XR: 33 sec/img

Performance:
iPhone 7: 9 img/sec
iPhone XR: 45 img/sec



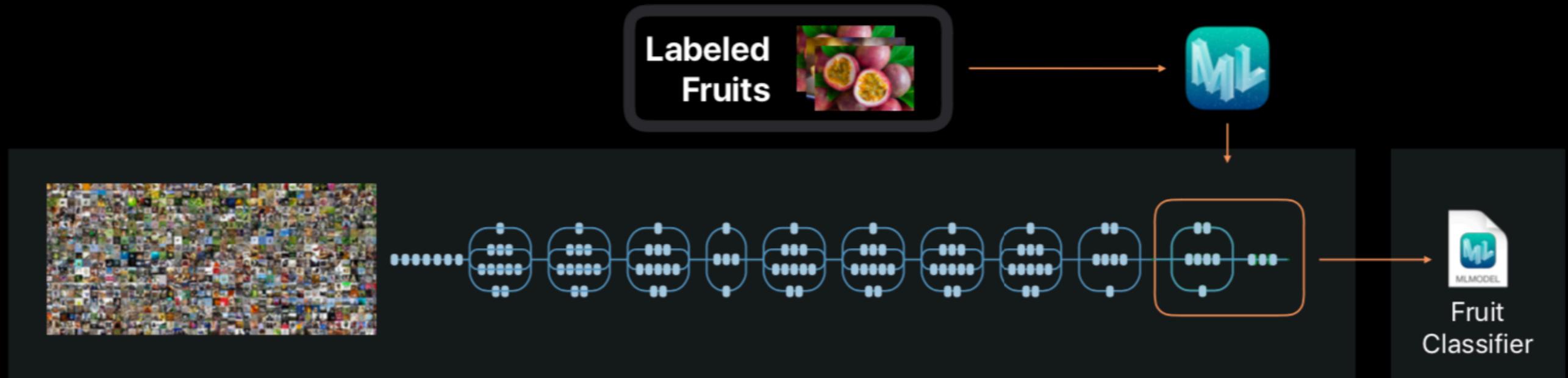
178 KB



37.1 MB

Not to

With Create ML, the model is really just the last layer over a pre-trained deep neural network



In a nutshell



- Image pre-processing can cause significant precision lost
 - Leverages already excellent iOS support
 - Real-time performance
-



- Complex architecture, long training times and heavy models
- When matched with the input resolution, the solution can render very good precision
- Can perform faster, due to highly-tuned architecture

Text Classification



Featured Prediction Competition

Toxic Comment Classification Challenge

Identify and classify toxic online comments

Jigsaw/Conversation AI · 4,551 teams · a year ago

\$35,000 Prize Money

Overview Data Kernels Discussion Leaderboard Rules Team My Submissions Late Submission

Data Description

You are provided with a large number of Wikipedia comments which have been labeled by human raters for toxic behavior. The types of toxicity are:

- toxic
- severe_toxic
- obscene
- threat
- insult
- identity_hate

You must create a model which predicts a probability of each type of toxicity for each comment.



Offensive language ahead

Toxic

“ You are a know it all....Jimbo! Why don't you get a life instead of sitting in front of your computer all day. The high priestess of knowledge, indeed!

“ I don't mean to startle anyone but God is dead. We should not worry about him anymore. Just thought I would let everyone know. Well, goodbye and good luck with your newfound crisis of faith!

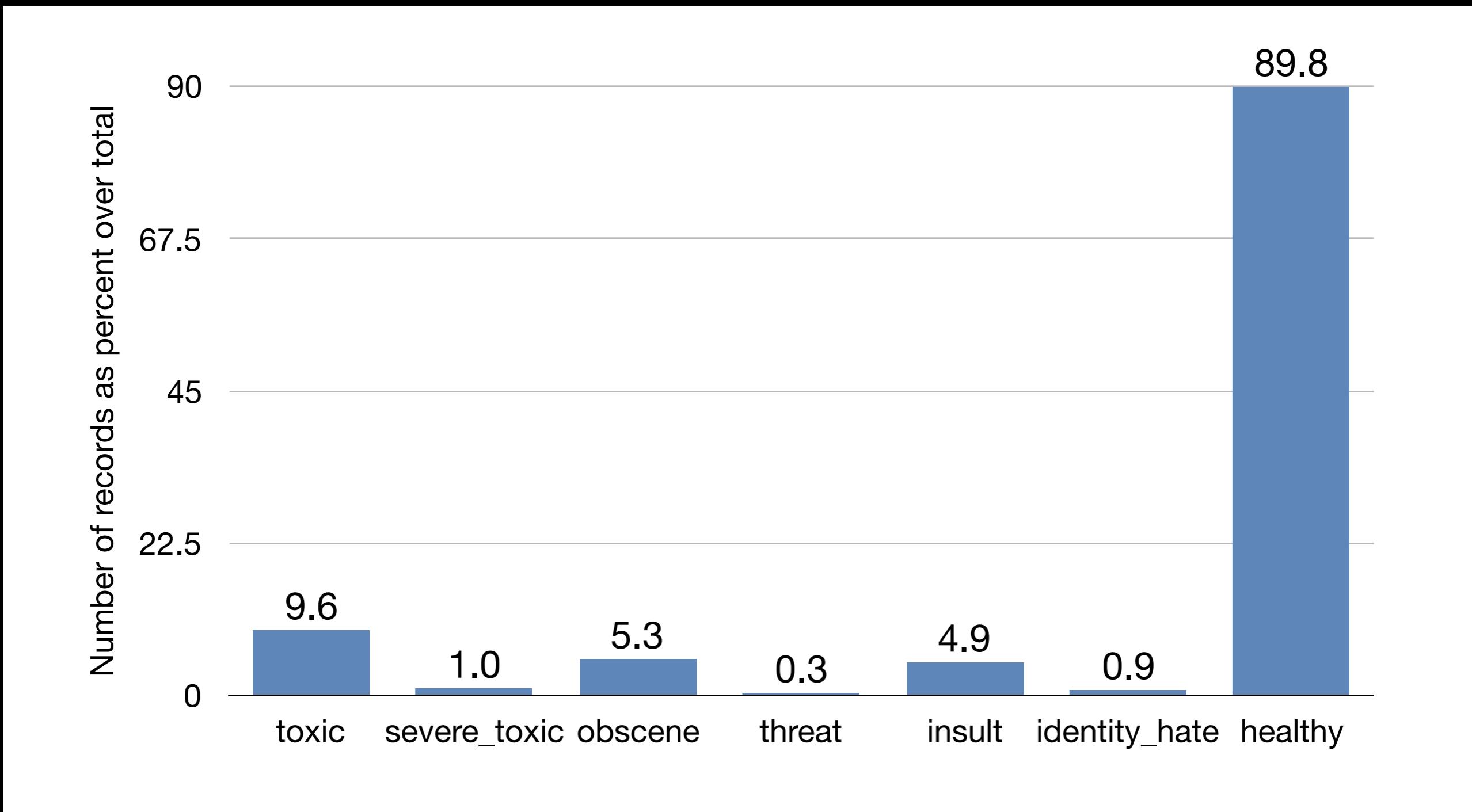
Identity hate

“ Black americans have a hive mind mentality and automatically switch political party preferences just like that.

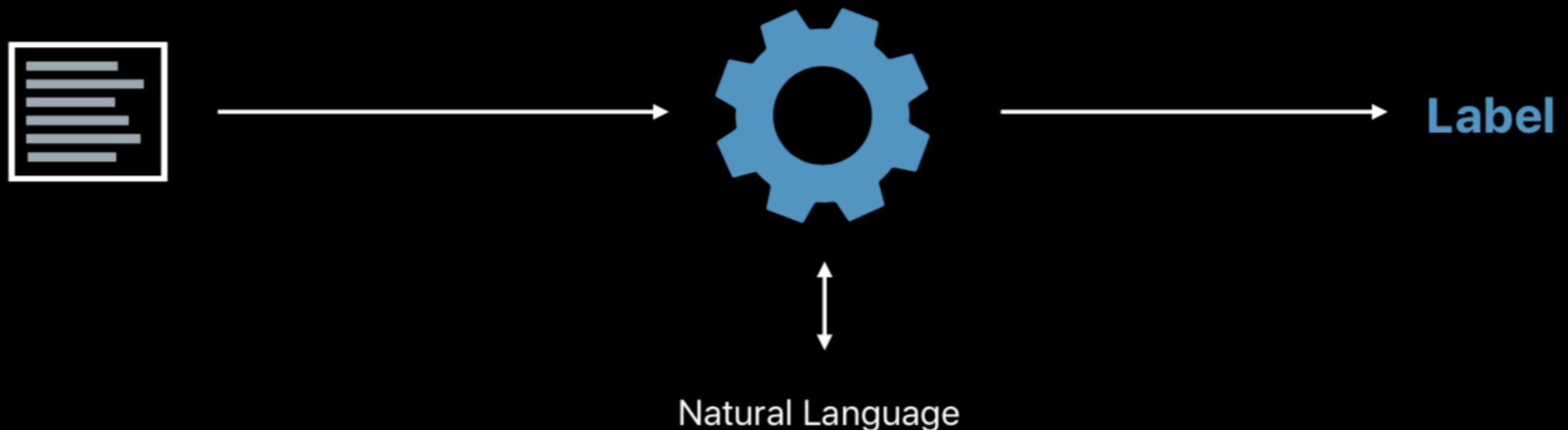
“ Ossmann, are you jew or your father was an SS officer?

Data distribution

~160,000 comments



Stages for text classification



Training model



```
let parameters =  
MLTextClassifier.ModelParameters(validationData: nil,  
algorithm: .maxEnt(revision: 1),  
language: .english)
```

```
let classifier = try? MLTextClassifier(trainingData: trainingData,  
textColumn: "comment_text",  
labelColumn: targetColumn,  
parameters: parameters)
```

Training model



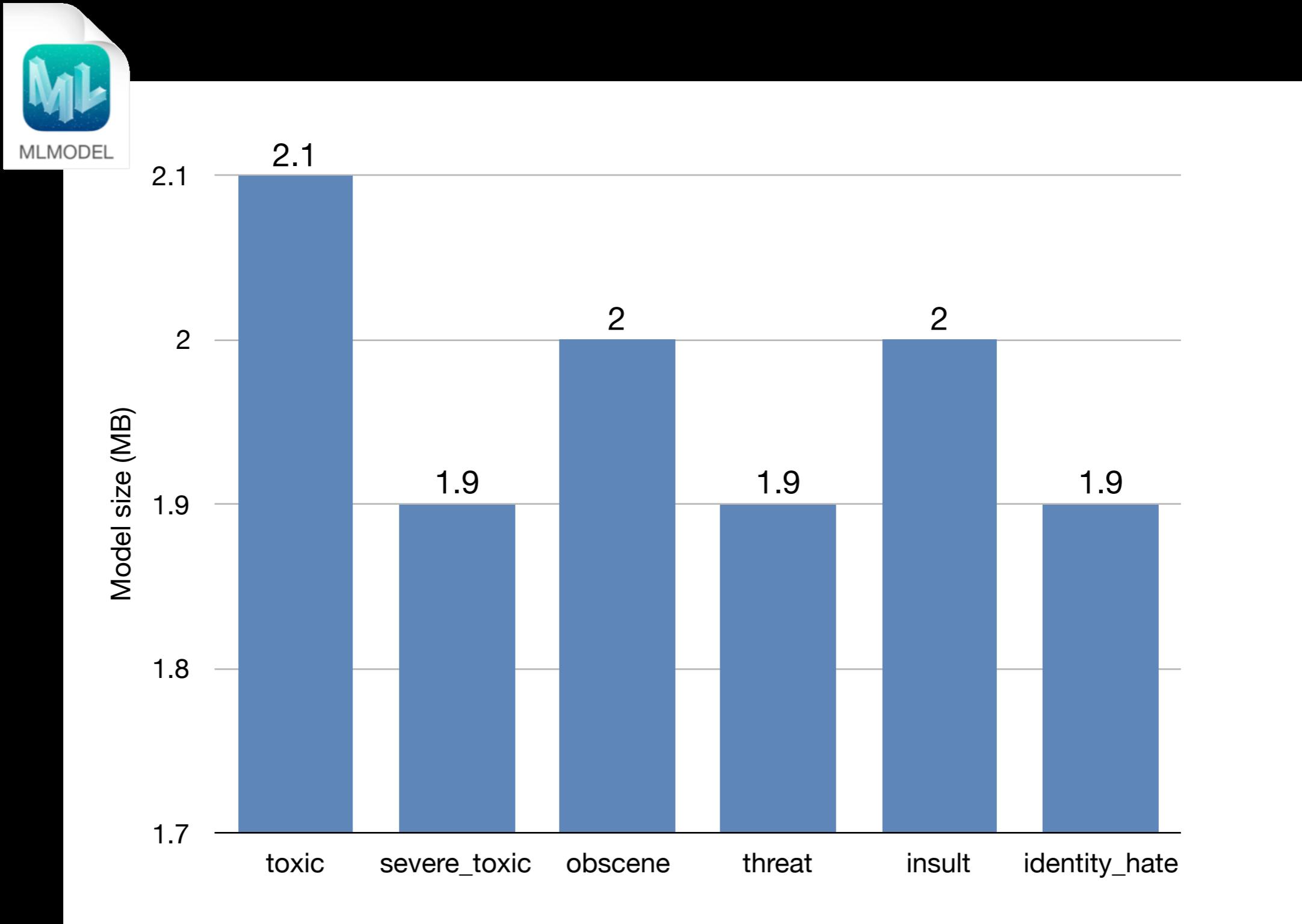
```
Automatically generating validation set from 5% of the data.  
Tokenizing data and extracting features  
10% complete  
20% complete  
30% complete  
40% complete  
50% complete  
60% complete  
70% complete  
80% complete  
90% complete  
100% complete  
Starting MaxEnt training with 121170 samples  
Iteration 1 training accuracy 0.904143  
Iteration 2 training accuracy 0.909672  
Iteration 3 training accuracy 0.941735  
Iteration 4 training accuracy 0.952719  
Iteration 5 training accuracy 0.960535  
Iteration 6 training accuracy 0.967665  
Iteration 7 training accuracy 0.974730
```

...

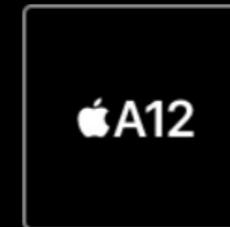
```
Iteration 42 training accuracy 0.997549  
Iteration 43 training accuracy 0.997565  
Iteration 44 training accuracy 0.997598  
Iteration 45 training accuracy 0.997615  
Finished MaxEnt training in 48.01 seconds  
Classifier for "toxic":  
TextClassifier
```

Accuracy ~ 95 %

Model size



Performance on device



iPhone XR

**Neural engine
8 cores**

>4,500 comments/second



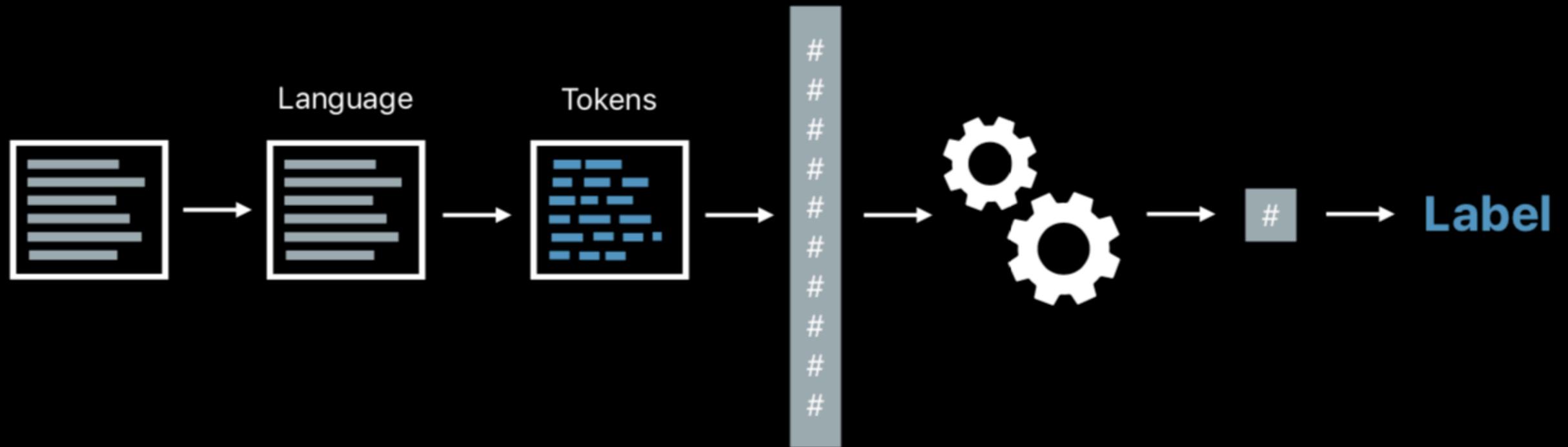
iPhone 7

**No dedicated HW
4 cores**

~2,300 comments/second

Measured over ~80,000 comments

Stages for text classification



Training models



Clean up texts -> Tokenize -> Make bag of words -> Create tf-idf matrix -> Logistic regression with Naive-Bayes -> Probabilities

“

... situation where a linear model is pretty close to the state of the art.

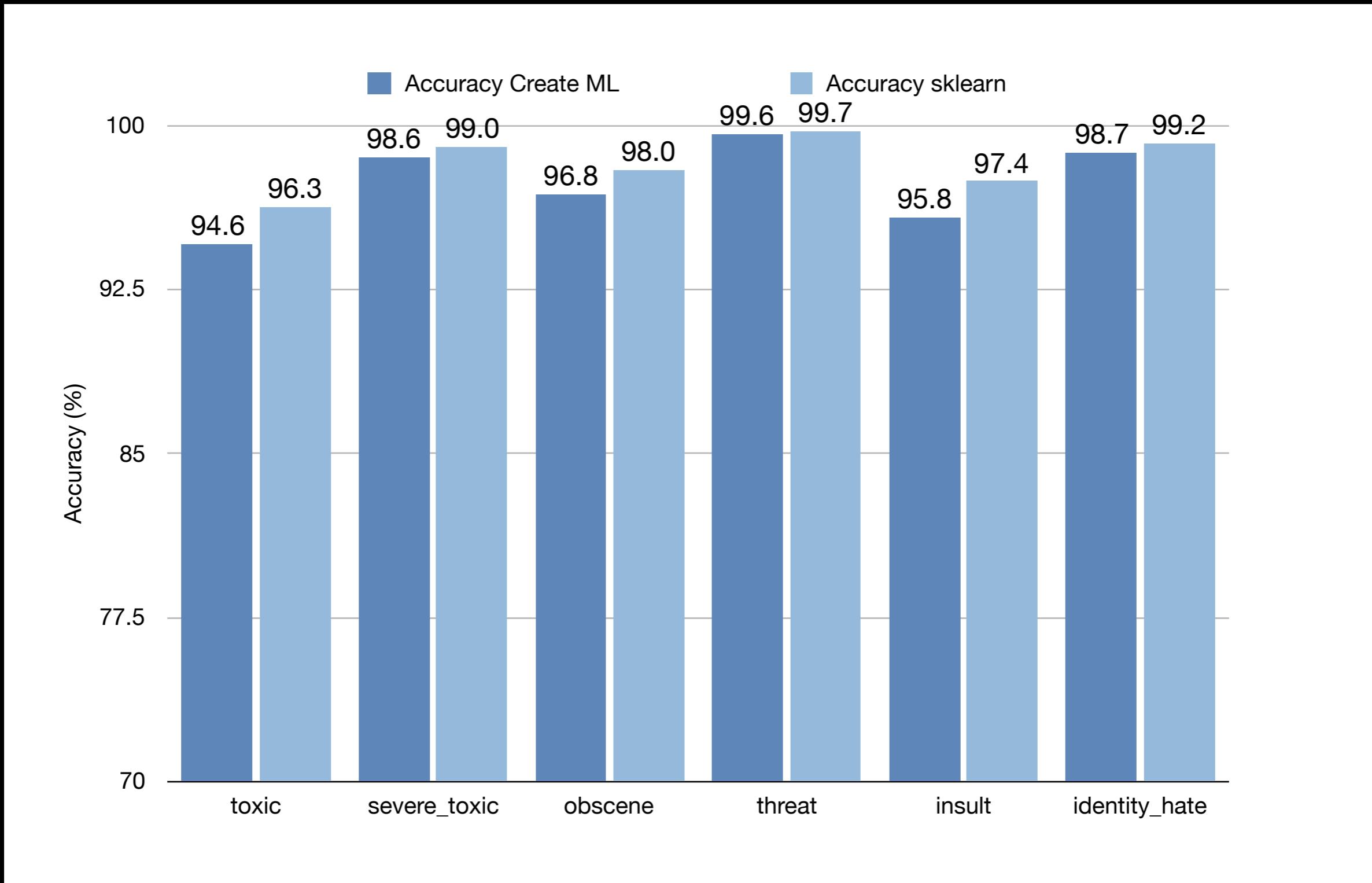
– Jeremy Howard, from fast.ai



VS.



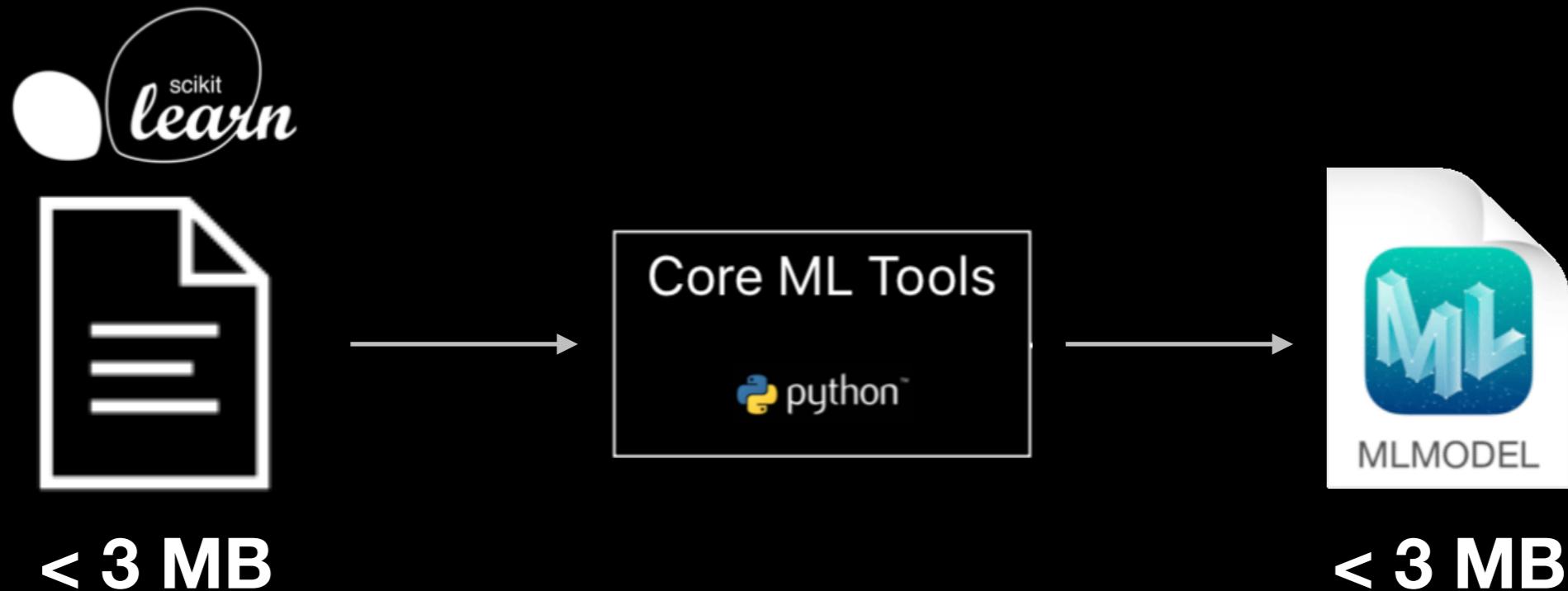
Accuracy



Convert models



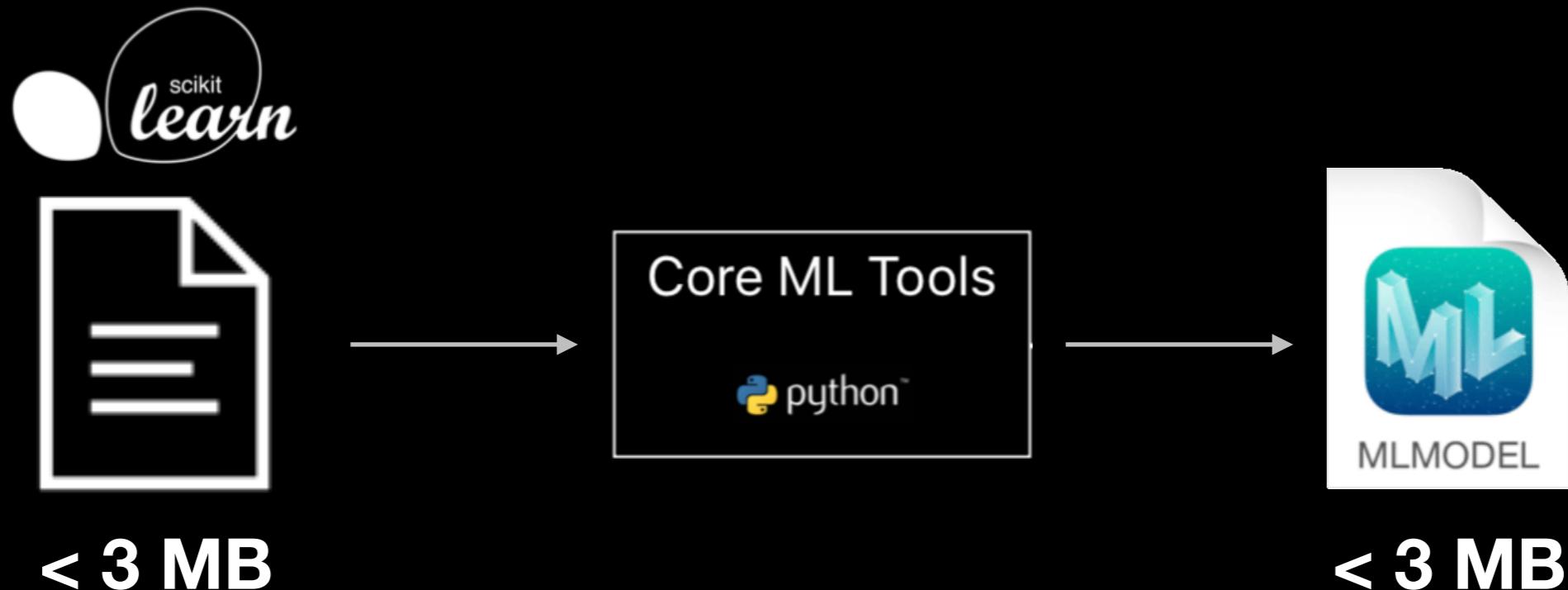
Logistic Regressor (sklearn)



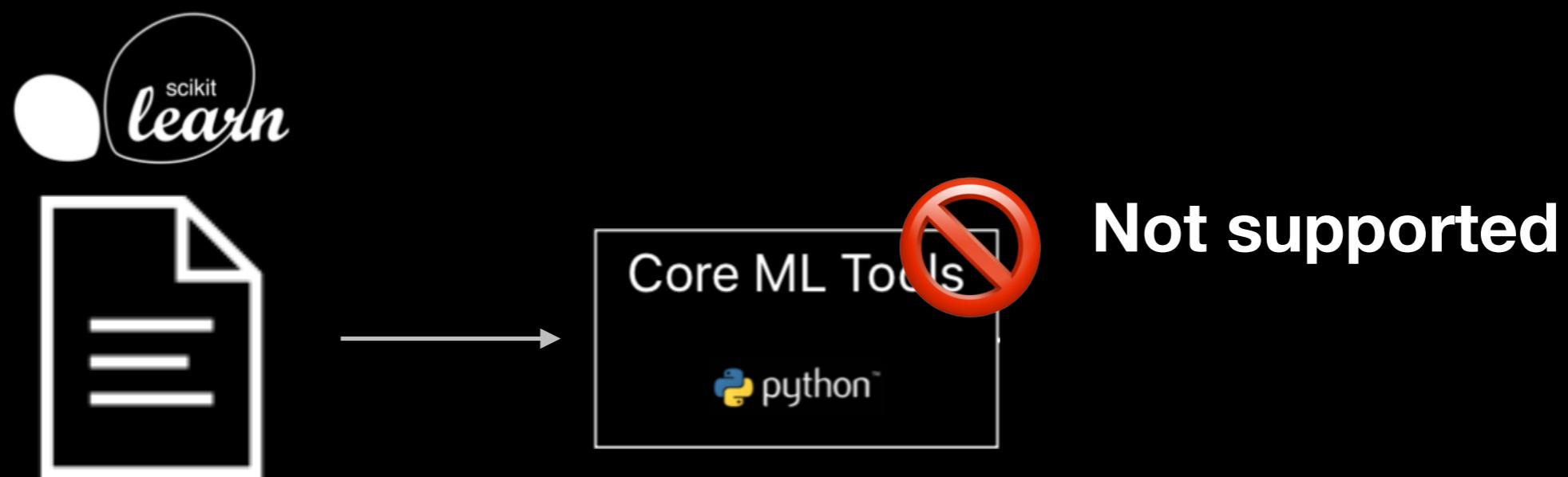
Convert models



Logistic Regressor (sklearn)



TF-IDF Vectorizer (sklearn)



In a nutshell



- Streamlined solution
- High accuracy, low model size, fast performance, fast training



- Need to code feature extraction on device
- Lacking support for typical algorithms

Tabular data



Featured Prediction Competition

Santander Customer Transaction Prediction

Can you identify who will make a transaction?

Banco Santander · 8,802 teams · 2 months ago

\$65,000 Prize Money

Overview Data Kernels Discussion Leaderboard Rules Team My Submissions Late Submission

Data Description

You are provided with an anonymized dataset containing numeric feature variables, the binary `target` column, and a string `ID_code` column.

The task is to predict the value of `target` column in the test set.

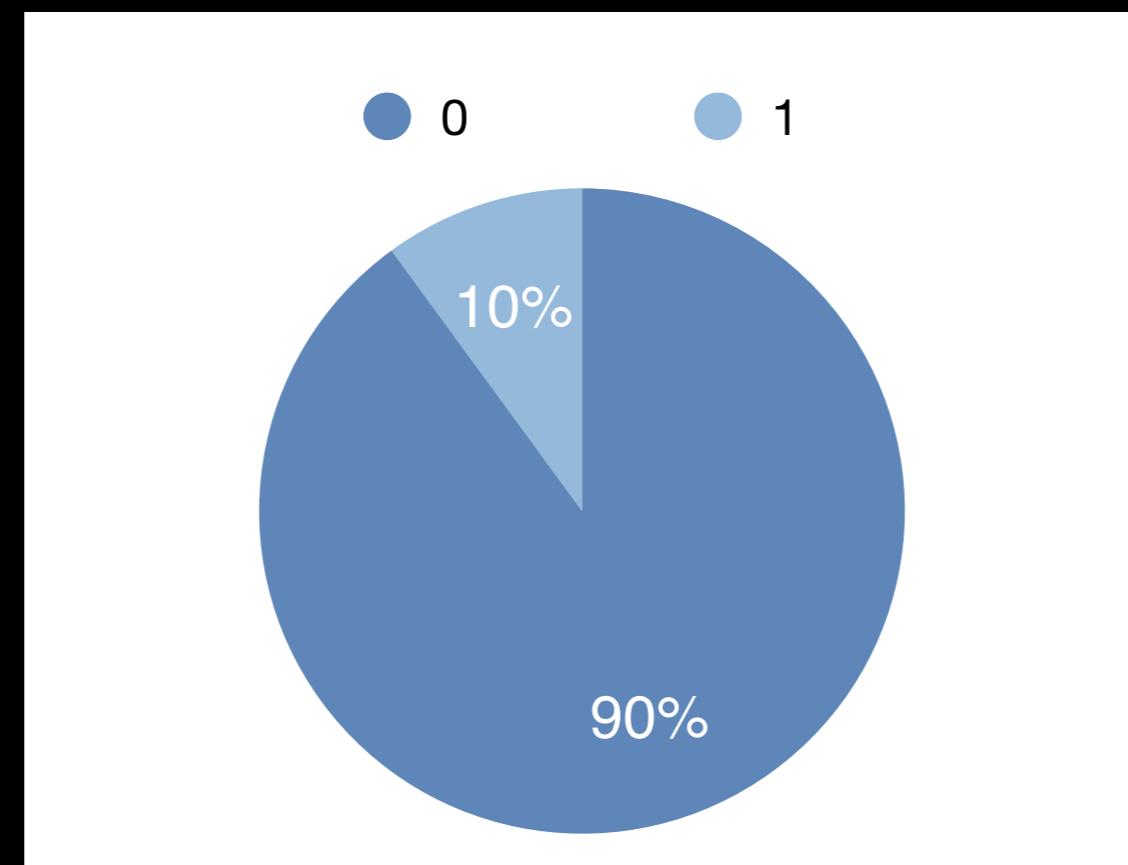
File descriptions

- `train.csv` - the training set.
- `test.csv` - the test set. The test set contains some rows which are not included in scoring.
- `sample_submission.csv` - a sample submission file in the correct format.

Tabular data



- 200,000 records
- 200 numerical features
- 1 binary target: customer will make a transaction or not



Model training



Regressor Cases

The regressor case picked automatically when initializing an `MLRegressor`.

`case linear(MLLinearRegressor)`

A regressor that estimates the target as a linear function of the features.

`case decisionTree(MLDecisionTreeRegressor)`

A regressor that estimates the target by learning rules to split the data.

`case boostedTree(MLBoostedTreeRegressor)`

A regressor based on a collection of decision trees combined with gradient boosting.

`case randomForest(MLRandomForestRegressor)`

A regressor based on a collection of decision trees trained on subsets of the data.



Model training



Model training



maxDepth

The maximum depth of the tree. Must be a value of at least 1.

The default value is 6.

maxIterations

The maximum number of passes through the data. Each iteration creates an extra tree.

The default value is 10.

minLossReduction

The minimum amount of reduction in the loss function that is required to make another split to the data. Larger values help prevent overfitting.

The default value is 0.

minChildWeight

Determines the minimum weight of each leaf node of the tree. Larger values help prevent overfitting.

The default value is 0.1.

randomSeed

A seed for internal random operations. Set this value to ensure reproducible results.

The default value is 42.

Model training



When used to minimize the above function, a standard (or "batch") [gradient descent](#) method would perform the following iterations :

$$w := w - \eta \nabla Q(w) = w - \eta \sum_{i=1}^n \nabla Q_i(w)/n,$$

where η is a step size (sometimes called the [learning rate](#) in machine learning).

In classical statistics, sum-minimization problems arise in [least squares](#) and in [maximum-likelihood estimation](#) (for independent observations). The general class of [estimators](#) that arise as minimizers of sums are called [M-estimators](#). However, in statistics, it has been long recognized that requiring even local minimization is too restrictive for some problems of maximum-likelihood estimation.^[2] Therefore, contemporary statistical theorists often consider [stationary points](#) of the [likelihood](#) function (or zeros of its derivative, the [score function](#), and other [estimating equations](#)).

The [convergence](#) of stochastic gradient descent has been analyzed using the theories of [convex minimization](#) and of [stochastic approximation](#). Briefly, when the [learning rates](#) η decrease with an appropriate rate, and subject to relatively mild assumptions, stochastic gradient descent converges [almost surely](#) to a global minimum when the objective function is [convex](#) or [pseudoconvex](#), and otherwise converges almost surely to a local minimum.^{[4][5]} This is in fact a consequence of the [Robbins-Siegmund theorem](#).^[6]

Model training



Choosing solution



Stratified k-folds to prevent overfitting
5 models trained with partitioned training set
Output averaged over 5 models

Choosing solution



Best solution with LightGBM



Converters

Automatically convert models from popular machine learning libraries such as Keras, Caffe, scikit-learn, LibSVM, and XGboost to the Core ML format.

caffe.convert	Convert a Caffe model to Core ML format.
keras.convert	Convert a Keras model to Core ML protobuf specification (.mlmodel).
libsvm.convert	Convert a LIBSVM model to Core ML format.
sklearn.convert	Convert scikit-learn pipeline, classifier, or regressor to Core ML format.
xgboost.convert	Convert a trained XGBoost model to Core ML format.

Converting model



Try with **XGBoostClassifier**

```
raise TypeError("Unexpected type. Expecting XGBoost model.")  
TypeError: Unexpected type. Expecting XGBoost model.
```



Choosing solution



apple / [coremltools](#)

Used by 169 Watch 80 Star 1,368 Fork 169

Code Issues 125 Pull requests 14 Projects 0 Wiki Security Insights

Tree: cf0dcfc8d1 ▾ [coremltools](#) / [coremltools](#) / [converters](#) / [xgboost](#) / [_tree_ensemble.py](#) Find file Copy path

 [gustavla](#) XGBoost 0.7 name change fix d9e1bbc on 19 Jan 2018

2 contributors  

157 lines (121 sloc) | 5.56 KB Raw Blame History   

```
105 import json
106 import os
107 feature_map = None
108 if isinstance(model, (_xgboost.core.Booster, _xgboost.XGBRegressor)):
109
```

Converting model

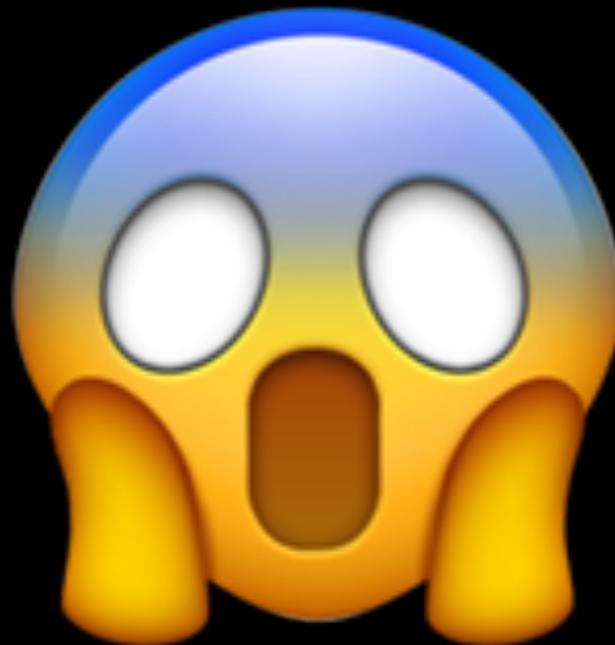


Try with **XGBoostRegressor**



RMSE = 0.082

RMSE = 3.17



Converting model



apple / coremltools

Used by 170 Watch 80 Star 1,370 Fork 169

Code Issues 126 Pull requests 14 Projects 0 Wiki Security Insights

Filters is:issue is:open Labels 11 Milestones 0 New issue

126 Open ✓ 78 Closed Author Labels Projects Milestones Assignee Sort

apple / coremltools

Used by 170 Watch 80 Star 1,370 Fork 169

Code Issues 126 Pull requests 14 Projects 0 Wiki Security Insights

New issue

XGBoost model in CoreMLTools produces completely different results #174

Open stylianos-kampakis opened this issue on 16 Apr 2018 · 8 comments

Converting model



Xgboost classifier conversion #293

[Open](#) zaccharieramzi wants to merge 23 commits into [apple:master](#) from [zaccharieramzi:xgboost-classifier-conversion](#)

Conversation 2

Commits 23

Checks 0

Files changed 4

+390 -27



zaccharieramzi commented on 20 Nov 2018 • edited

+ ...

Reviewers

No reviews

Assignees

No one assigned

This solves [#174](#)

It will allow users to convert an xgboost classifier model to a coreml one.

Converting model



Back to **XGBoostClassifier**



RMSE = 0.082

RMSE = 0.11





VS.



RMSE: 0.15

RMSE: 0.11

**Training time (20_000 it):
10 hours**

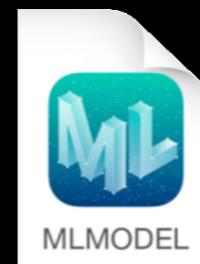
**Training time:
16.8 hours**

Performance:
iPhone 7: 440 pred/sec
iPhone XR: 453 pred/sec

Performance:
iPhone 7: 253 pred/sec
iPhone XR: 194 pred/sec



40.7 MB



**14.7 MB
(all 5 variations)**

In a nutshell



- Params are hard to choose
- Longer training time means larger architecture and bigger models



- Limited (and buggy) support of model formats
- Compact size and fast



Take-away



- Create ML is a seamless entry into ML for iOS developers
- Good results out of the box
- Create ML may lack flexibility
- iOS lacks support for scientific algorithms
- Model conversion is still very much a WIP
- Performance is real-time in every case



Bridging the gap



turi A white silhouette of a dog running to the right.



Turi Create



I want to...

Machine Learning Task

Label Images Image Classification

Recognize objects within images Object Detection

Find similar images Image Similarity

Create stylized avatars / profile images Style Transfer

Personalize choices for users Recommender

Detect an activity using sensors Activity Classification

Analyze sentiment of messages Text Classifier

Predict a label Classifiers

Predict numeric values Regression

Group similar datapoints together Clustering

Sources

- WWDC images property of Apple (Fair Use)
- Image classification kernel by Robert Ruzzo
- Text classification kernel by Jeremy Howard
- Tabular data kernel by Vladislav Bogorod

Thank you

@atineoSE
me@adriantineo.com

