Open | DSE

Architecting Big Data Solutions with Apache Spark Lecture 6: Machine Learning

Vladimir Osin

Data Scientist/Engineer

Signify Research (formerly known as Philips Lighting)

Outline

- Introduction to Spark
- Machine Learning Process
- Practice
- Spark ML Tips
- Next Steps

Introduction to Spark

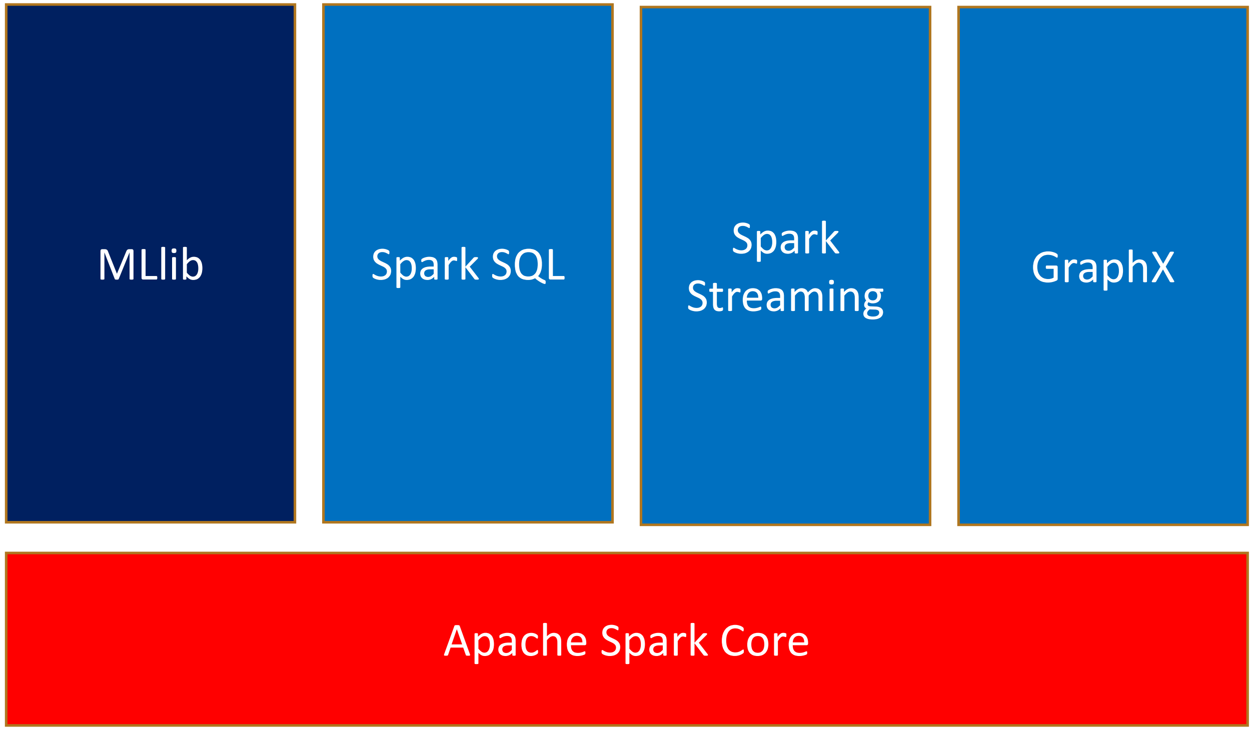
- **Spark** – a distributed, data processing platform for big data
  - Distributed – runs in cluster of servers
  - Processing – performs computations, such as ETL and modeling
  - Big Data – Terabyte and more volumes of data

- **Supports Multiple Languages**

Scala Java python™ R

Introduction to Spark

- **Modular Structure**



MLlib

Spark SQL

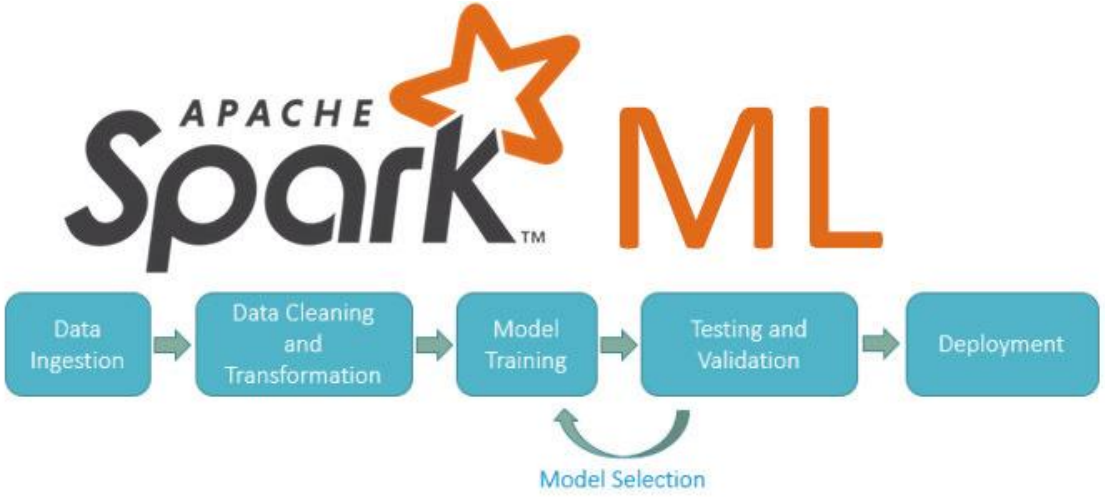
Spark Streaming

GraphX

Apache Spark Core

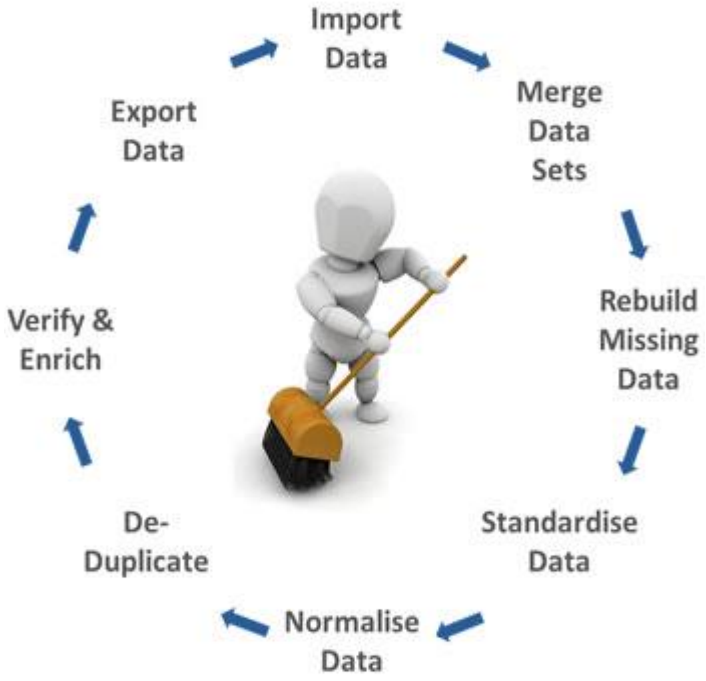
Machine Learning Process

- There are three main steps:
  - Preprocessing – collect, reformat and transform dataset
  - Model Building – apply algorithms to data
  - Validation – measure the quality of the model



Preprocessing

- Extract, transform and load your data
- Check missing/invalid values
- Normalization and Standardization



Building Models

- Model Selection
- Fitting data to the models
- Tuning parameters for models

Validating Models

- Apply models to additional data (**test set**)
- Measuring **quality** of models
  - Accuracy
  - Confusion Matrix
  - etc.

VALIDATED

MLlib Algorithms

- Supervised Learning
  - Regression – [Example Notebook]
    - Linear Regression
    - Decision Tree Regression
    - Gradient-boosted Tree Regression
  - Classification – [Example Notebook]
    - Naive Bayes
    - Decision trees
    - Multilayer perceptron
- Unsupervised Learning
  - Clustering (K-means)

Trees

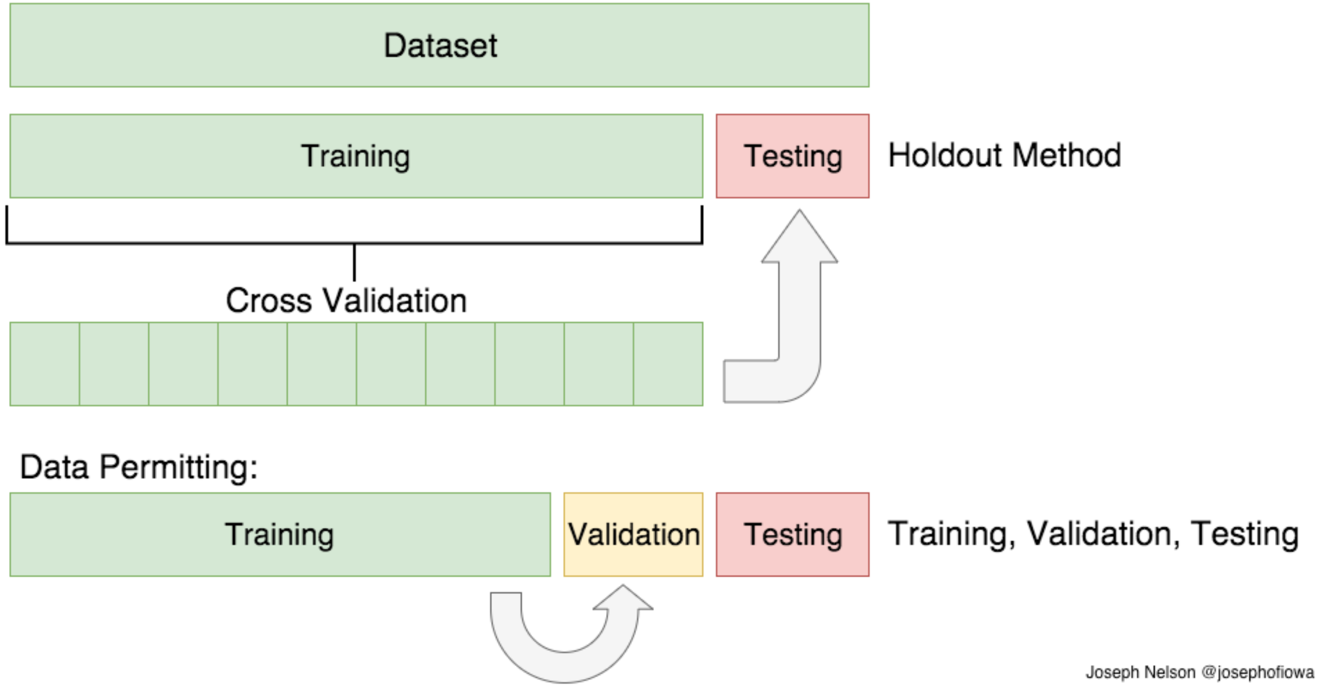
- Decision Tree
  - Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed.

- Random Forest
  - Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

- Gradient-Boosted Tree
  - Classifier is trained on data, taking into account the previous classifiers’ success. After each training step, the weights are redistributed. Misclassified data increases its weights to emphasize the most difficult cases. In this way, subsequent learners will focus on them during their training.

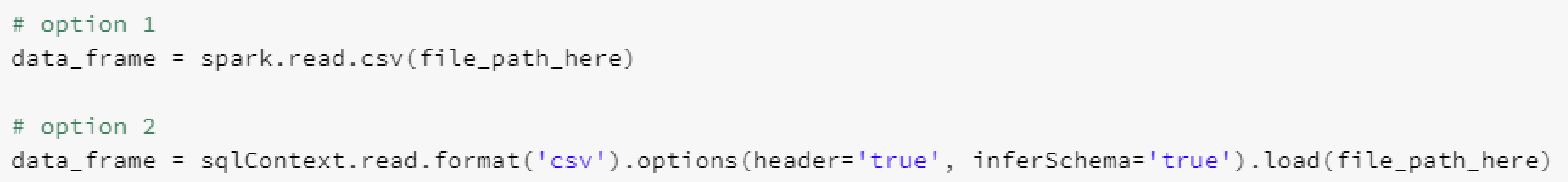
Cross-Validation

- Split your data
- Control overfitting
- Control underfitting
- Hyperparameters tuning



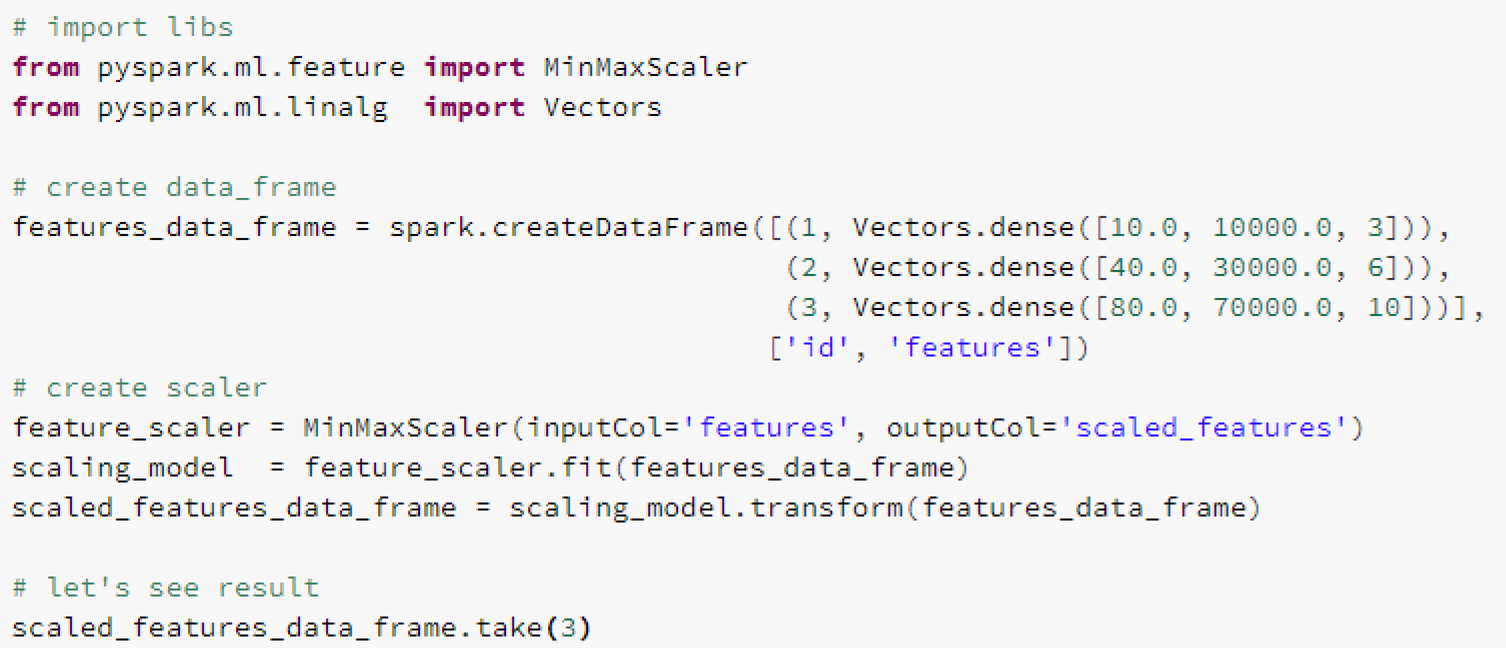
Joseph Nelson @josephofiowa

Practice: Pre-processing



```python
# option 1
data_frame = spark.read.csv(file_path_here)

# option 2
data_frame = sqlContext.read.format('csv').options(header='true', inferSchema='true').load(file_path_here)
```

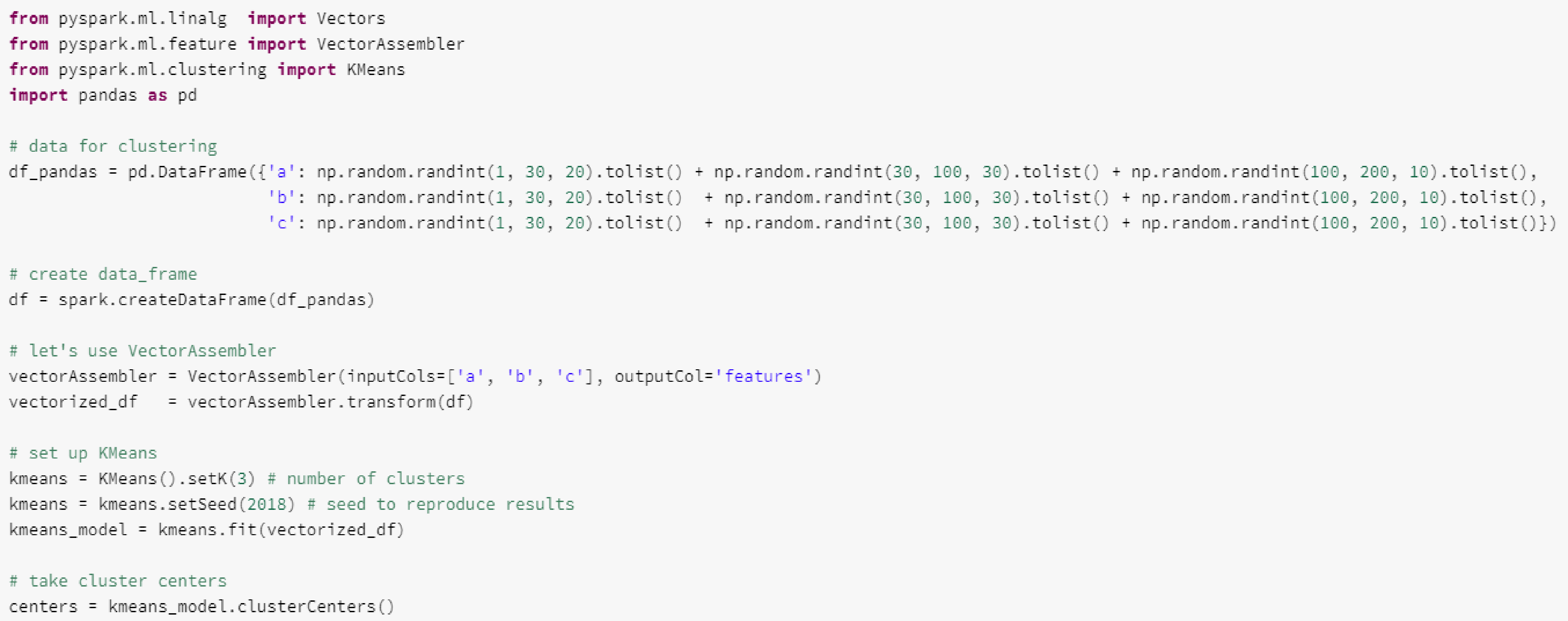


```python
# import libs
from pyspark.ml.feature import MinMaxScaler
from pyspark.ml.linalg  import Vectors

# create data_frame
features_data_frame = spark.createDataFrame([(1, Vectors.dense([10.0, 10000.0, 3])),
                                             (2, Vectors.dense([40.0, 30000.0, 6])),
                                             (3, Vectors.dense([80.0, 70000.0, 10]))],
                                            ['id', 'features'])
# create scaler
feature_scaler = MinMaxScaler(inputCol='features', outputCol='scaled_features')
scaling_model  = feature_scaler.fit(features_data_frame)
scaled_features_data_frame = scaling_model.transform(features_data_frame)

# let's see result
scaled_features_data_frame.take(3)
```

Practice: Clustering



```
from pyspark.ml.linalg  import Vectors
from pyspark.ml.feature import VectorAssembler
from pyspark.ml.clustering import KMeans
import pandas as pd

# data for clustering
df_pandas = pd.DataFrame({'a': np.random.randint(1, 30, 20).tolist() + np.random.randint(30, 100, 30).tolist() + np.random.randint(100, 200, 10).tolist(),
                          'b': np.random.randint(1, 30, 20).tolist()  + np.random.randint(30, 100, 30).tolist() + np.random.randint(100, 200, 10).tolist(),
                          'c': np.random.randint(1, 30, 20).tolist()  + np.random.randint(30, 100, 30).tolist() + np.random.randint(100, 200, 10).tolist()})

# create data_frame
df = spark.createDataFrame(df_pandas)

# let's use VectorAssembler
vectorAssembler = VectorAssembler(inputCols=['a', 'b', 'c'], outputCol='features')
vectorized_df   = vectorAssembler.transform(df)

# set up KMeans
kmeans = KMeans().setK(3) # number of clusters
kmeans = kmeans.setSeed(2018) # seed to reproduce results
kmeans_model = kmeans.fit(vectorized_df)

# take cluster centers
centers = kmeans_model.clusterCenters()
```

Spark ML Tips

- Preprocessing
  - Load data into Data Frame (not RDD)
  - Include column names (headers)
  - Use inferSchema=True
  - Use **VectorAssembler** to create feature vectors
  - Use **StringIndexer** to map from string to numeric indexes

Spark ML Tips

- Building Model
  - Split data into train and test sets
  - Train model using train set
  - Create prediction by applying model to test set
  - Consider Cross Validation during training

Spark ML Tips

- Use MLlib evaluators
  - RegressionEvaluator
  - BinaryClassificationEvaluator
  - MultiClassClassificationEvaluator
- Use different algorithms
- Tune hyperparamers

What’s next?

- Data Bricks documentation

K Keras

mxnet

PYTORCH

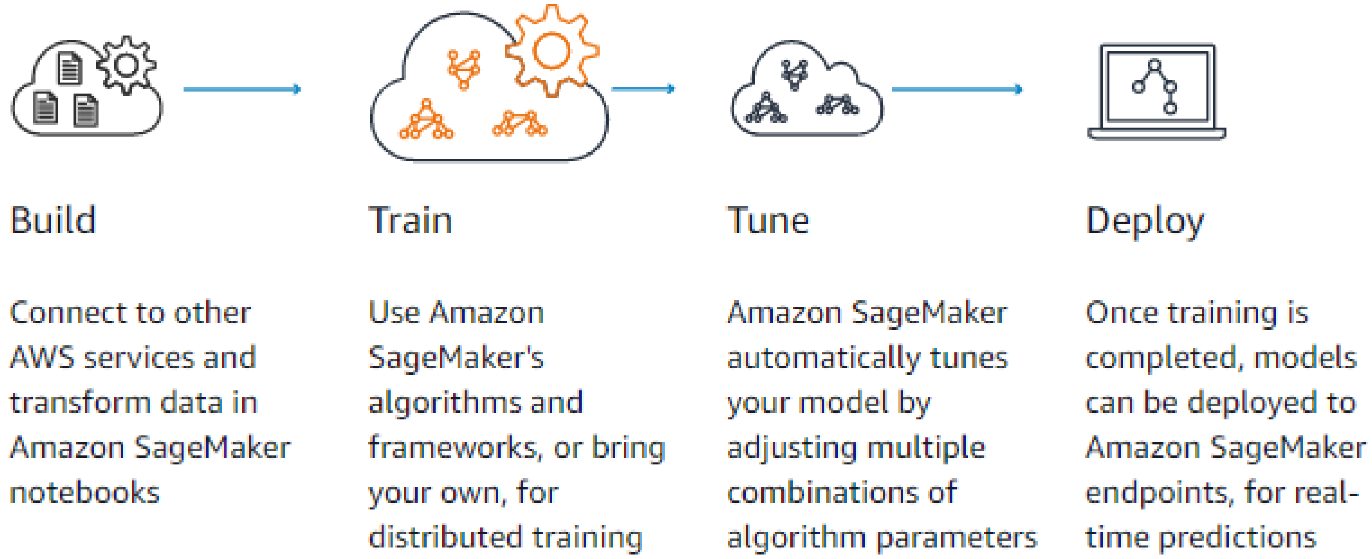
Deep Learning with PyTorch

TensorFlow

aws

Amazon
Machine Learning

Amazon
SageMaker



Build

Connect to other AWS services and transform data in Amazon SageMaker notebooks

Train

Use Amazon SageMaker's algorithms and frameworks, or bring your own, for distributed training

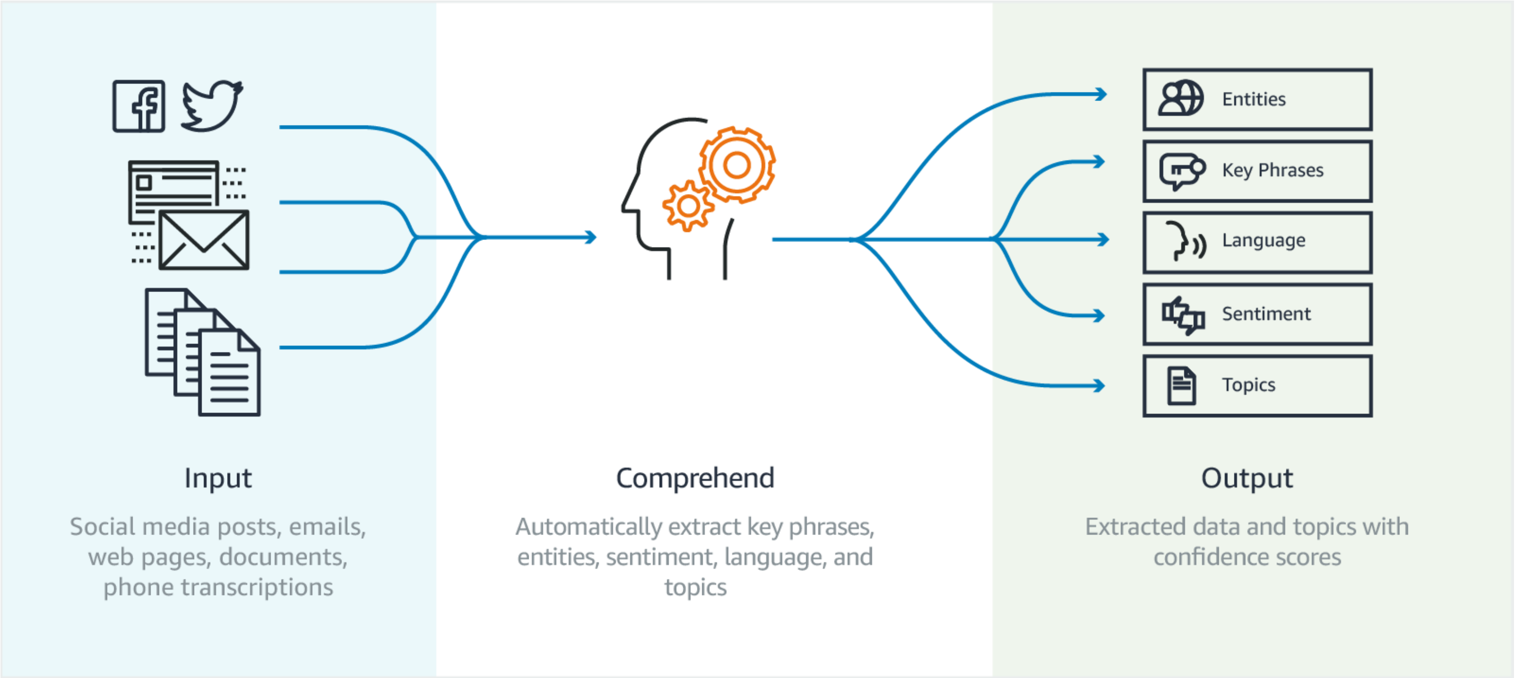
Tune

Amazon SageMaker automatically tunes your model by adjusting multiple combinations of algorithm parameters

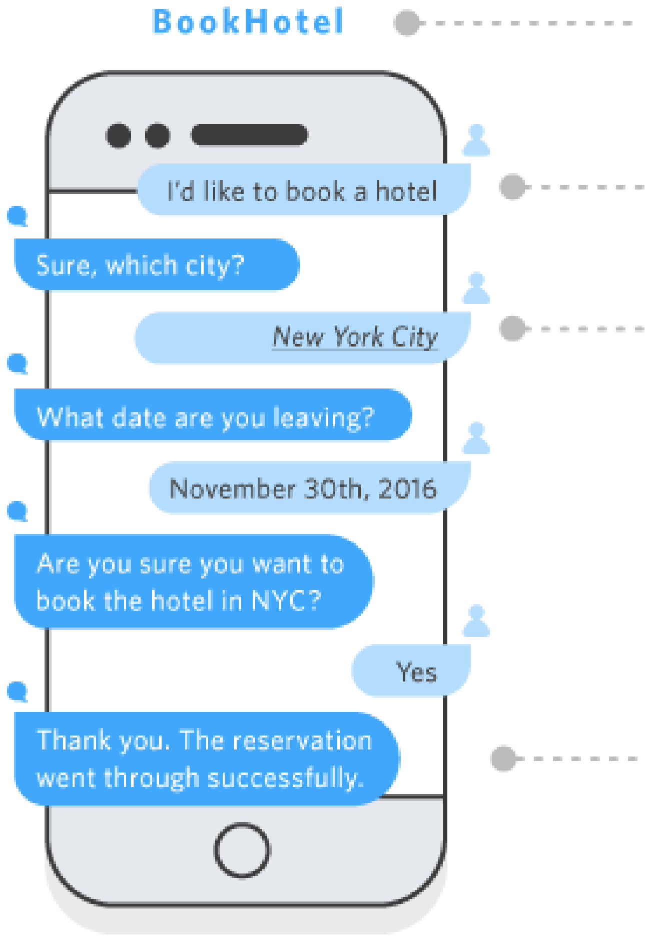
Deploy

Once training is completed, models can be deployed to Amazon SageMaker endpoints, for real-time predictions

Amazon Comprehend


Entities
Key Phrases
Language
Sentiment
Topics
Input
Social media posts, emails, web pages, documents, phone transcriptions
Comprehend
Automatically extract key phrases, entities, sentiment, language, and topics
Output
Extracted data and topics with confidence scores

Amazon Lex

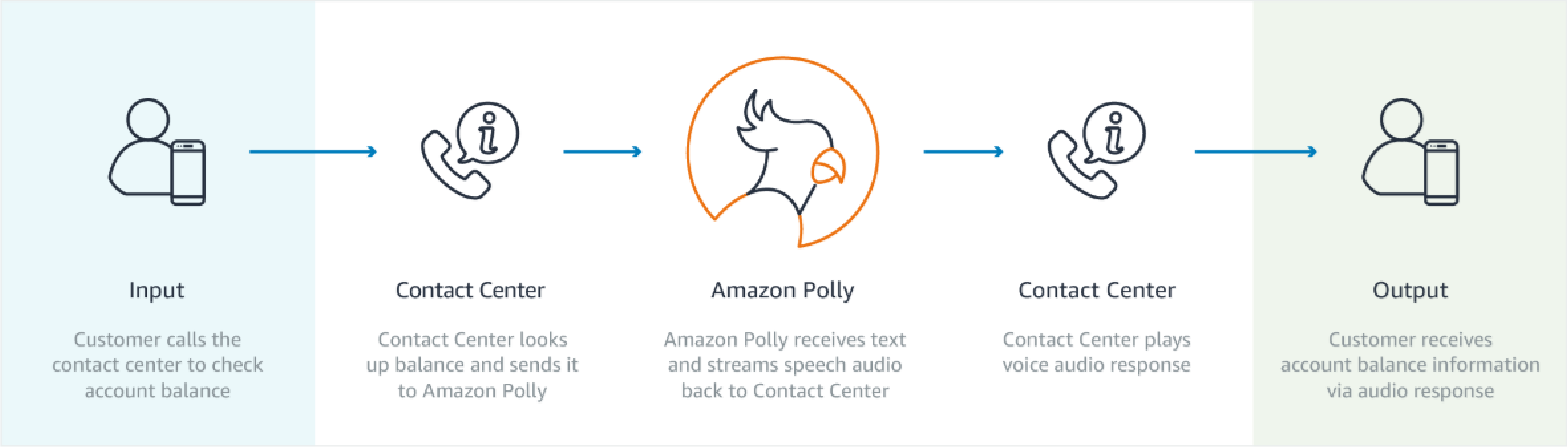

BookHotel
I'd like to book a hotel
Sure, which city?
New York City
What date are you leaving?
November 30th, 2016
Are you sure you want to book the hotel in NYC?
Yes
Thank you. The reservation went through successfully.

Intents
An intent performs an action in response to natural language user input

Utterances
Spoken or typed phrases that invoke your intent

Slots
Slots are input data required to fulfill the intent

Fulfillment
Fulfillment mechanism for your intent



Input

Customer calls the contact center to check account balance

Contact Center

Contact Center looks up balance and sends it to Amazon Polly

Amazon Polly

Amazon Polly receives text and streams speech audio back to Contact Center

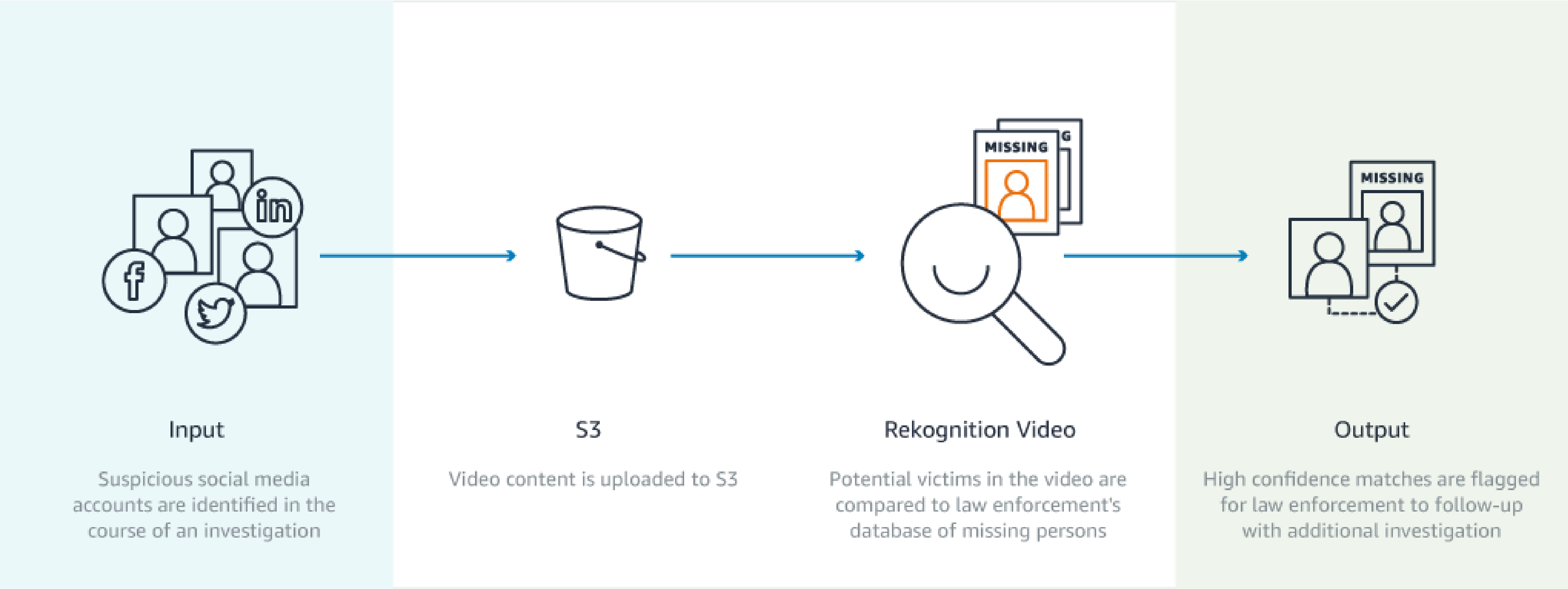
Contact Center

Contact Center plays voice audio response

Output

Customer receives account balance information via audio response

Amazon Poly



Input

Suspicious social media accounts are identified in the course of an investigation

S3

Video content is uploaded to S3

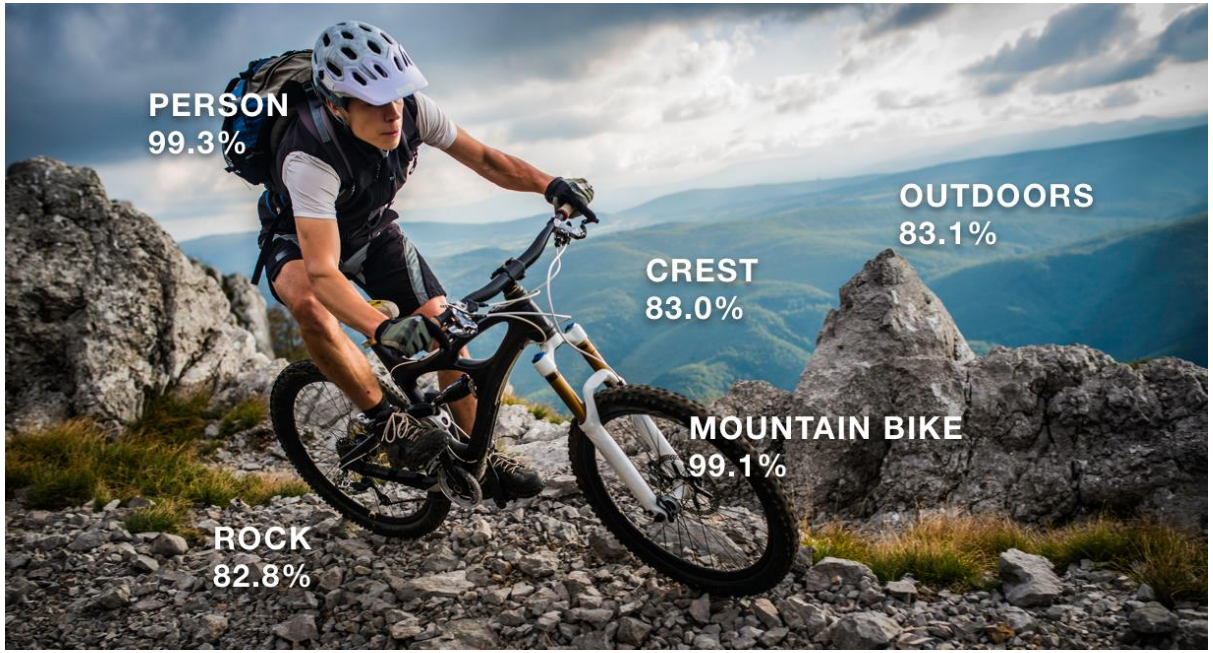
Rekognition Video

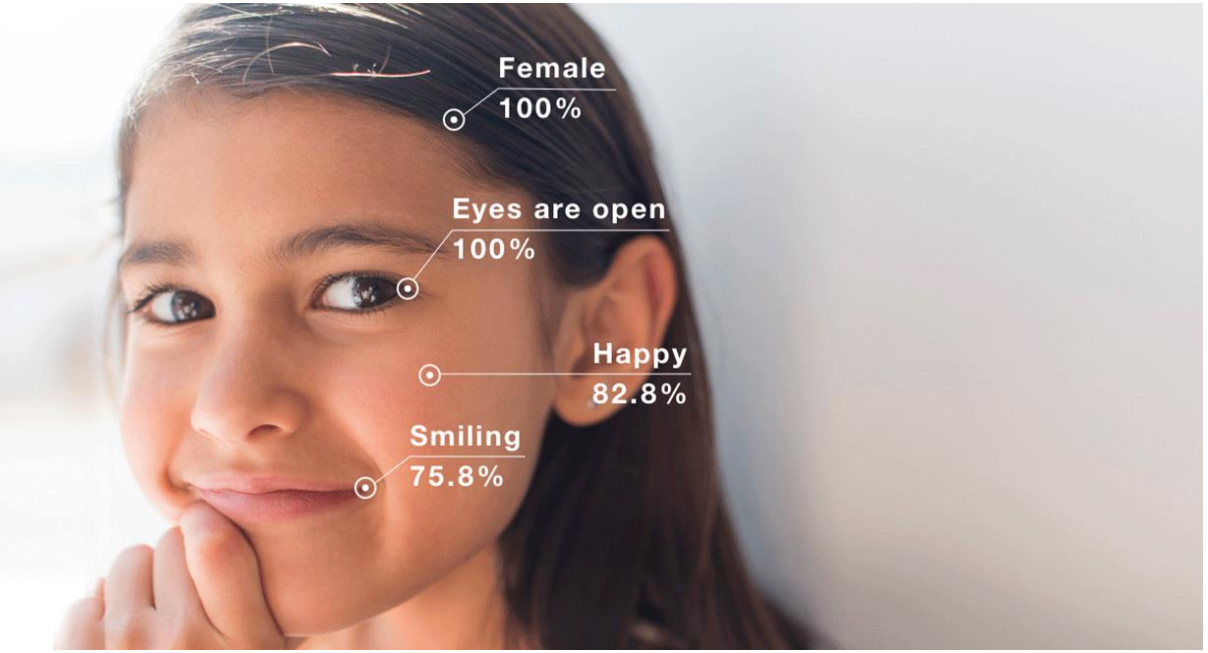
Potential victims in the video are compared to law enforcement's database of missing persons

Output

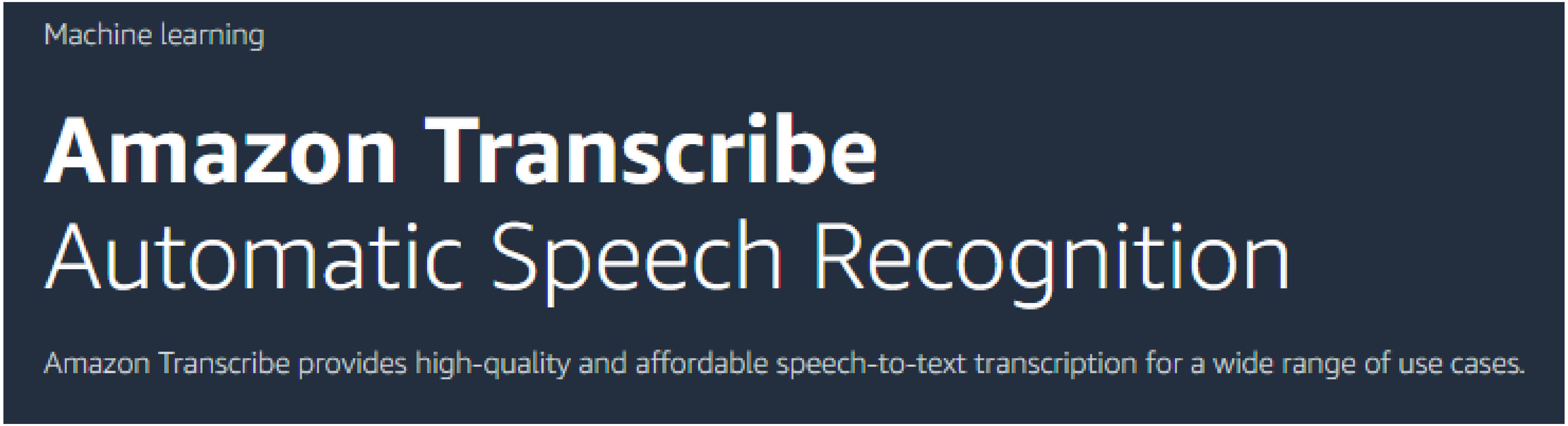
High confidence matches are flagged for law enforcement to follow-up with additional investigation

Amazon Rekognition


PERSON
99.3%
OUTDOORS
83.1%
CREST
83.0%
MOUNTAIN BIKE
99.1%
ROCK
82.8%


Female
100%
Eyes are open
100%
Happy
82.8%
Smiling
75.8%

PERSON 2
PERSON 4
PERSON 3
PERSON 1

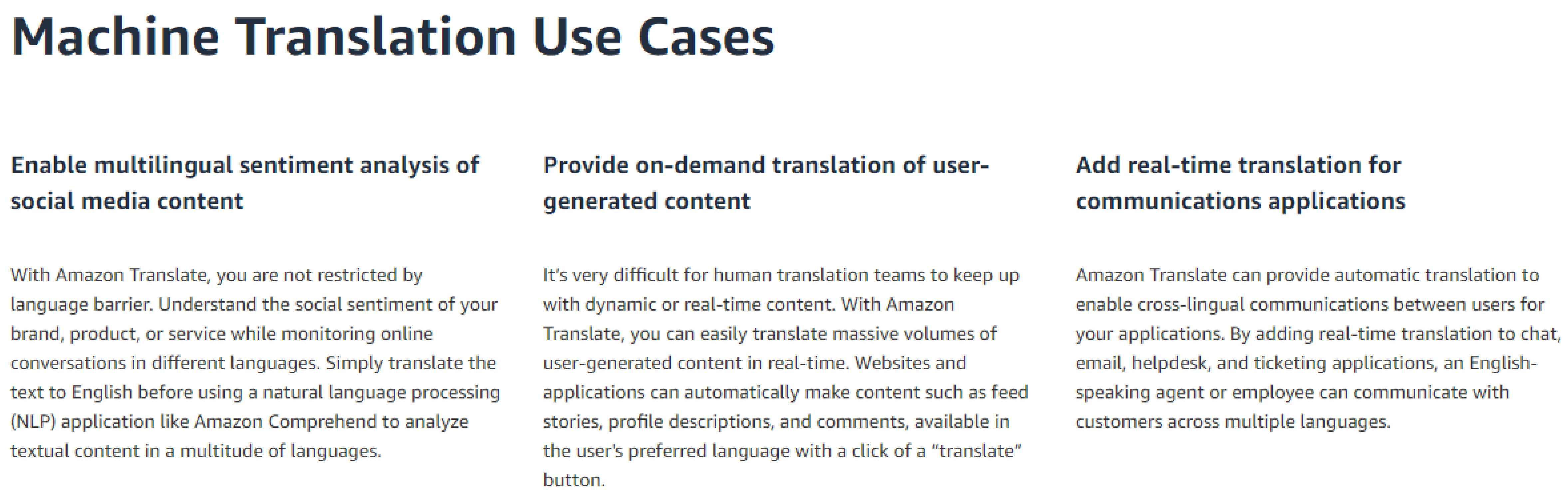


Machine learning

Amazon Transcribe

Automatic Speech Recognition

Amazon Transcribe provides high-quality and affordable speech-to-text transcription for a wide range of use cases.



Machine Translation Use Cases

Enable multilingual sentiment analysis of social media content

With Amazon Translate, you are not restricted by language barrier. Understand the social sentiment of your brand, product, or service while monitoring online conversations in different languages. Simply translate the text to English before using a natural language processing (NLP) application like Amazon Comprehend to analyze textual content in a multitude of languages.

Provide on-demand translation of user-generated content

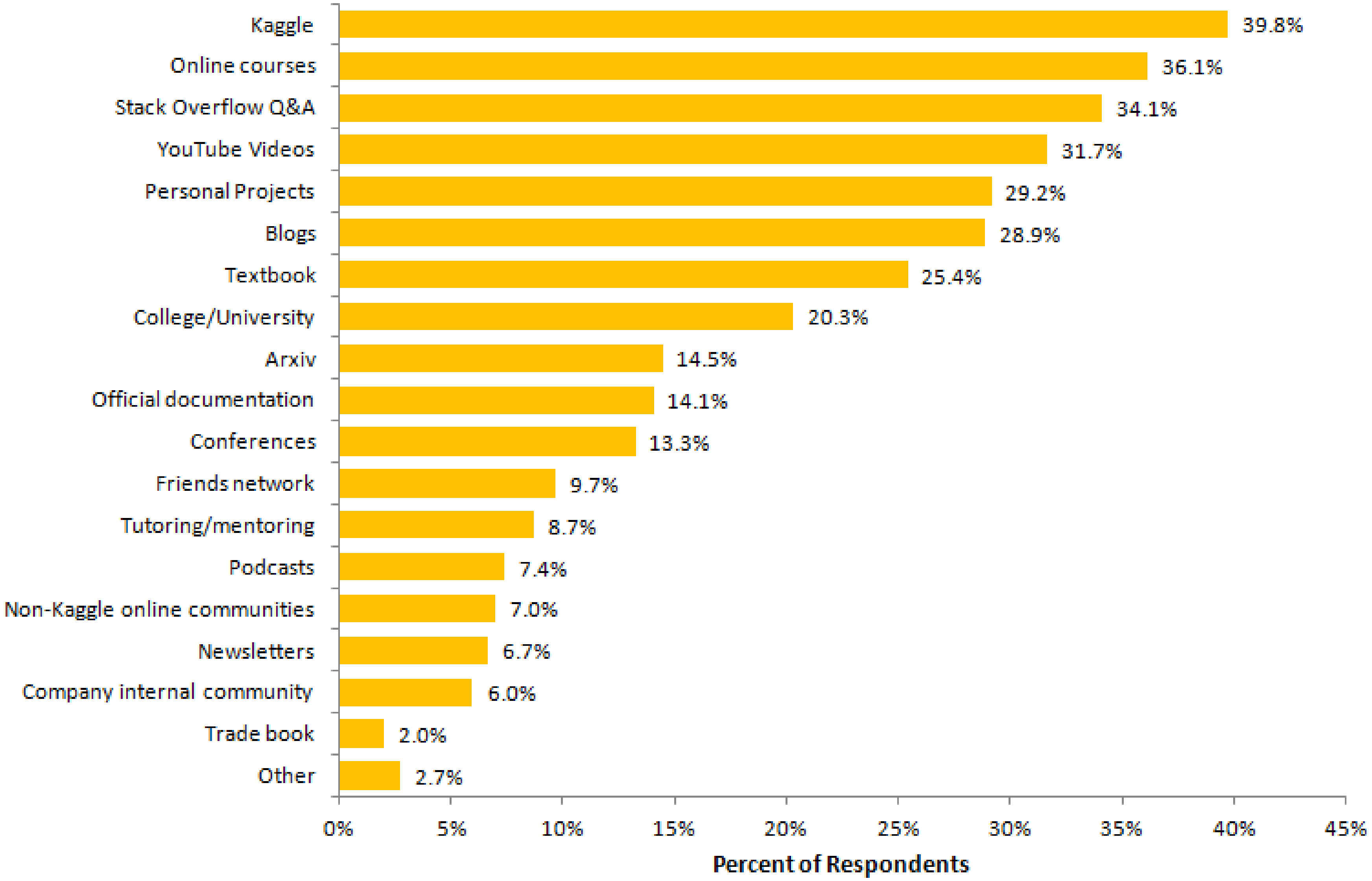
It's very difficult for human translation teams to keep up with dynamic or real-time content. With Amazon Translate, you can easily translate massive volumes of user-generated content in real-time. Websites and applications can automatically make content such as feed stories, profile descriptions, and comments, available in the user's preferred language with a click of a "translate" button.

Add real-time translation for communications applications

Amazon Translate can provide automatic translation to enable cross-lingual communications between users for your applications. By adding real-time translation to chat, email, helpdesk, and ticketing applications, an English-speaking agent or employee can communicate with customers across multiple languages.

kaggle™

Platforms and Resources You Have Used to Continue Learning Data Science Skills



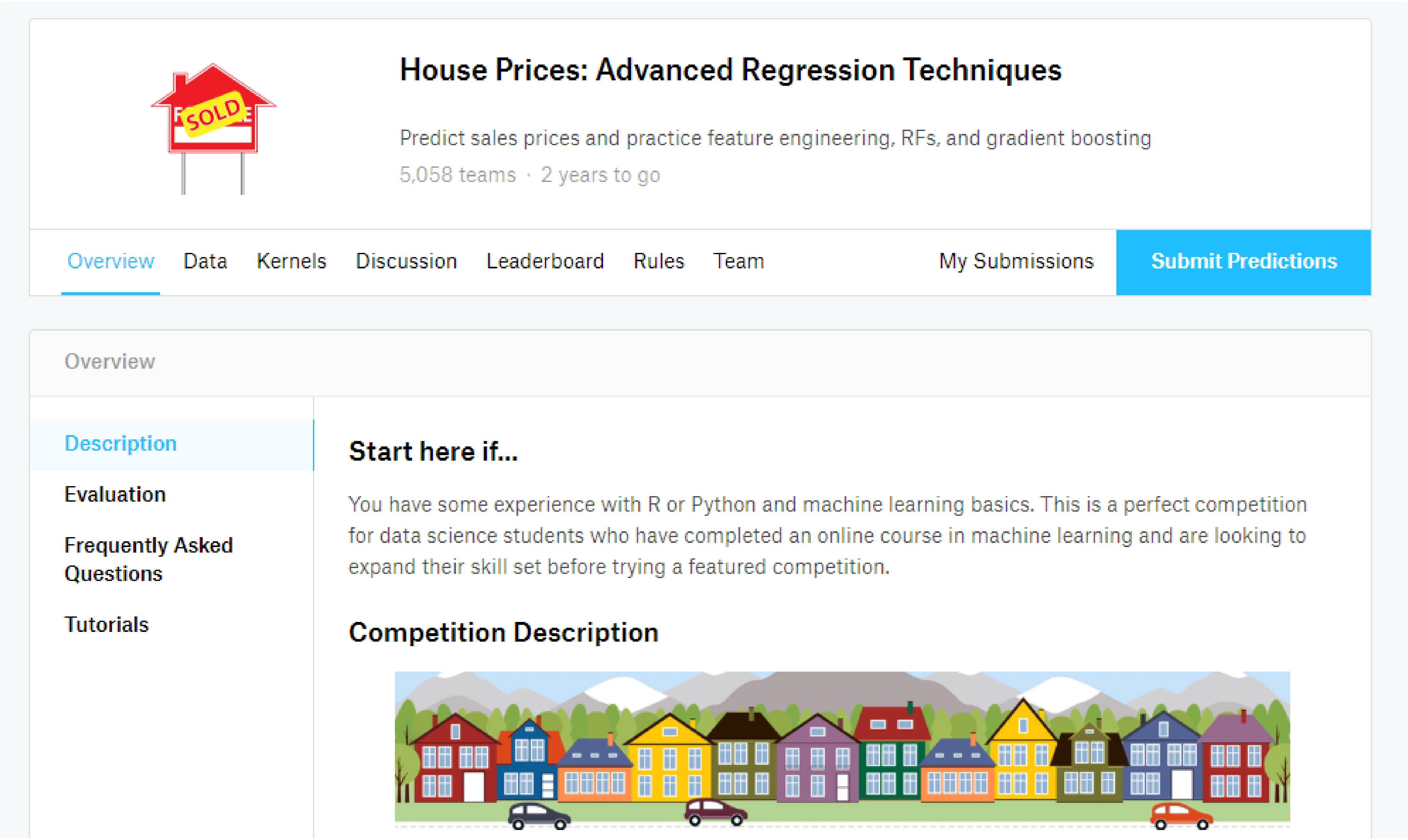
Competitions Platforms

- Kaggle
- DrivenData
- CrowdAnalityx
- CodaLab
- DataScienceChallenge.net
- DataScience.net
- Single-competition sites (like KDD, VizDooM)

Why to participate?

- Great opportunity to **learn** and networking
- Interesting non-trivial tasks and state-of-the-art approaches
- A way to become famous inside data science community
- A way to earn some money

- Wrap up competition results as:
  - LinkedIn project
  - GitHub repository
  - Medium blog



House Prices: Advanced Regression Techniques

Predict sales prices and practice feature engineering, RFs, and gradient boosting

5,058 teams · 2 years to go

Start here if...

You have some experience with R or Python and machine learning basics. This is a perfect competition for data science students who have completed an online course in machine learning and are looking to expand their skill set before trying a featured competition.

Competition Description