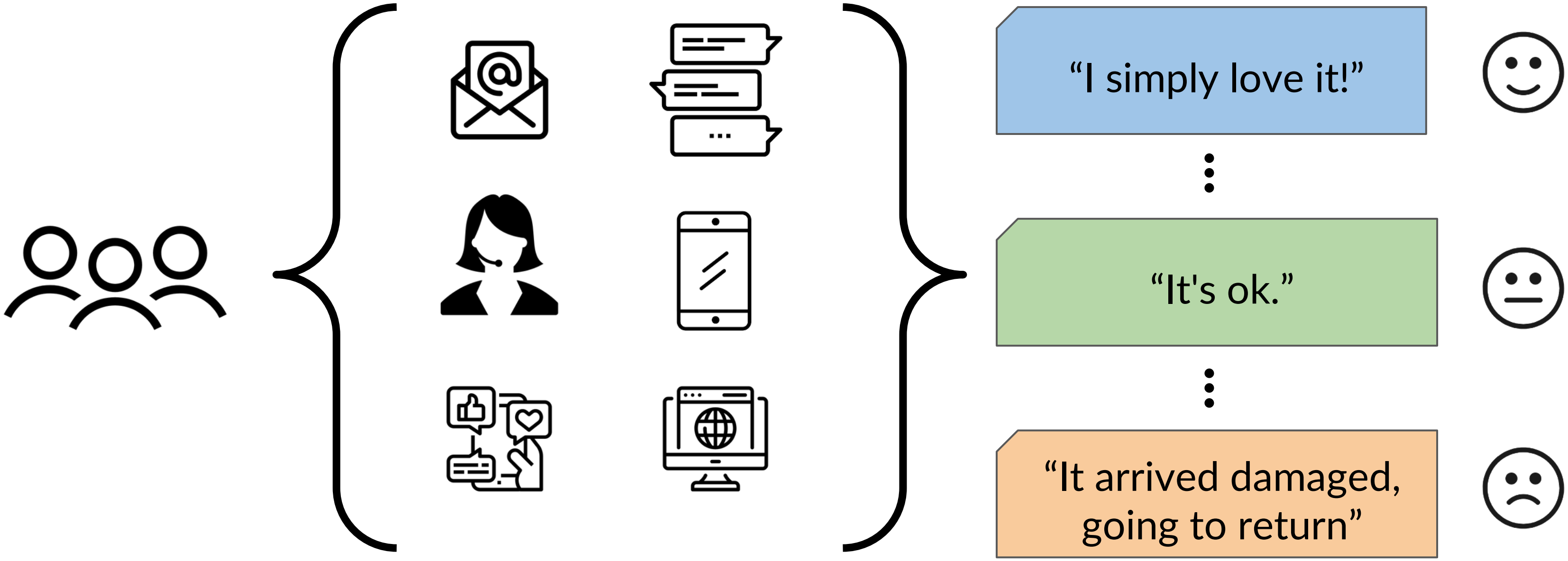
Copyright Notice

Practical Data Science

Use AutoML to Train a Text Classifier

Multi-class classification for sentiment analysis of product reviews



Machine Learning Workflow

Ingest & Analyze → **Prepare & Transform** → **Train & Tune** → **Deploy & Manage**

Ingest & Analyze	Prepare & Transform	Train & Tune	Deploy & Manage
Data exploration	Feature engineering	Automated ML	Model deployment
Bias detection	Feature store	Model train and tune	Automated pipelines
Amazon S3 & Amazon Athena	Amazon SageMaker Data Wrangler	Amazon SageMaker Autopilot	Amazon SageMaker Endpoints
AWS Glue	Amazon SageMaker Processing Jobs	Amazon SageMaker Training & Debugger	Amazon SageMaker Batch Transform
Amazon SageMaker Data Wrangler & Clarify	Amazon SageMaker Feature Store	Amazon SageMaker Hyperparameter Tuning	Amazon SageMaker Pipelines

Automated Machine Learning

(AutoML)

Model Building Challenges

Ability to reduce Time-to-Market

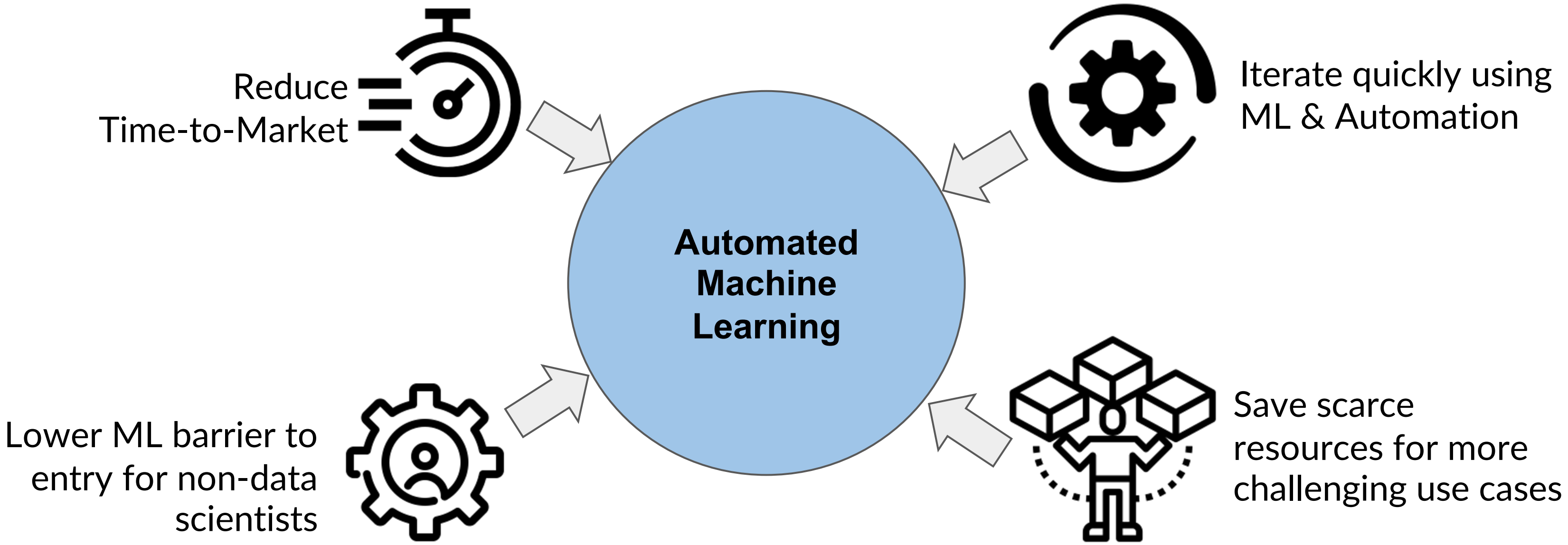
Ability to iterate quickly

Lack of ML skillsets

Ability to optimize scarce resources & skillsets

Automated Machine Learning

Automated Machine Learning (or AutoML) uses machine learning to automate many of the tasks in the machine learning workflow, allowing you to address some of those challenges



Machine Learning Workflow

Ingest & Analyze → **Prepare & Transform** → **Train & Tune** → **Deploy & Manage**

Ingest & Analyze	Prepare & Transform	Train & Tune	Deploy & Manage
Data exploration	Feature engineering	Automated ML	Model deployment
Bias detection	Feature store	Model train and tune	Automated pipelines

Use a specific combination of:

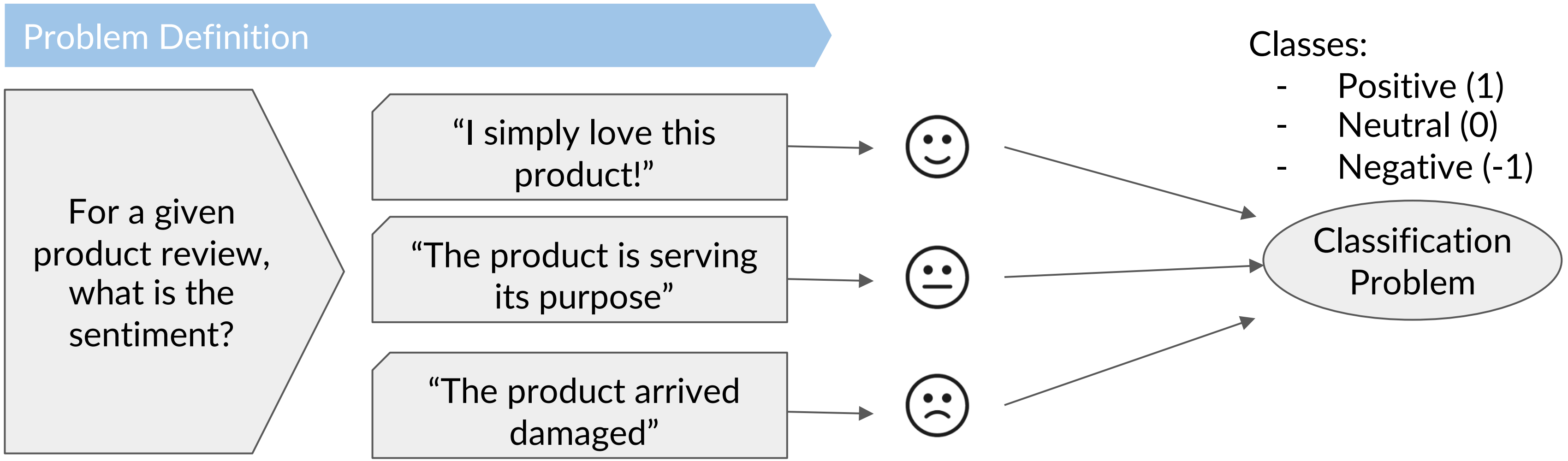
- algorithm,
- data transformations, and
- hyper parameters

Data Preparation

Ingest & Analyze → **Prepare & Transform** → **Train & Tune** → **Deploy & Manage**

Data Analysis:

Collecting statistics, such as missing entries, quantiles, skewness, correlation with the target.

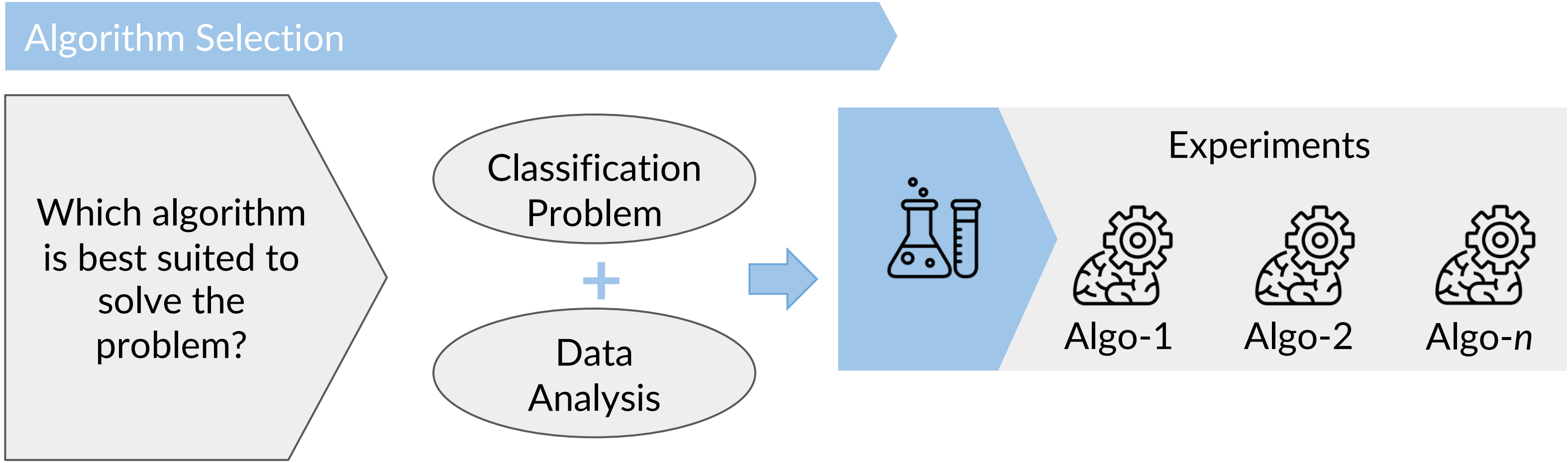


Data Preparation

Ingest & Analyze → Prepare & Transform → Train & Tune → Deploy & Manage

Data Analysis:

Collecting statistics, such as missing entries, quantiles, skewness, correlation with the target.



Data Preparation

Ingest & Analyze → Prepare & Transform → Train & Tune → Deploy & Manage

Data Analysis:

Collecting statistics, such as missing values, quantiles, skewness, correlation with the target.

Dataset Schema Detection

review_id (Numeric)	review_text (Categorical)	sentiment (Numeric)
001	"I simply love this product!"	1
002	"The product is serving its purpose"	0
003	"The Product arrived damaged"	-1

Data Preparation

Ingest & Analyze → **Prepare & Transform** → Train & Tune → Deploy & Manage

Data Transformation:

How should data be transformed so that the model can predict as accurately as possible?

review_id	review_text	sentiment
001	"I simply love this product!"	1
002	"The product is serving its purpose"	0
003	"The Product arrived damaged"	-1

Too Many Unique Values
= Treat as Text

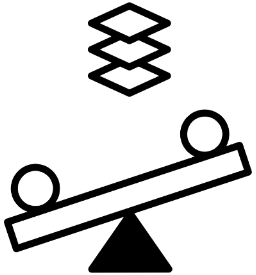
Text Transformation
- TD IDF
- Word to Vec
- Doc to Vec, etc.

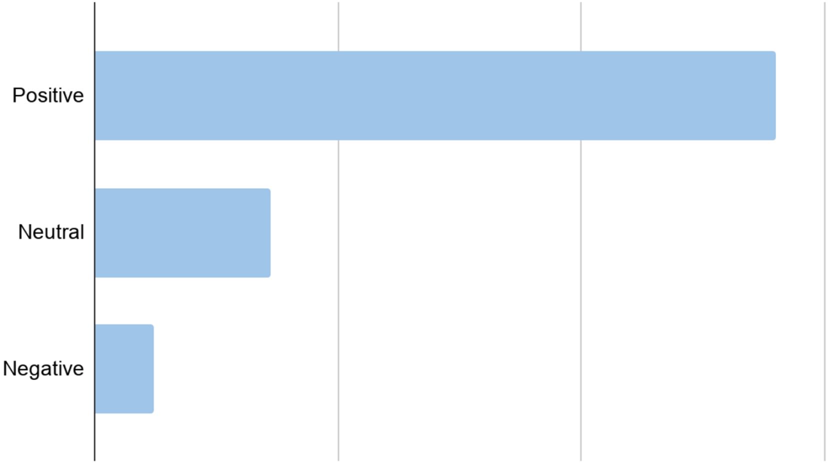
Data Preparation

Ingest & Analyze → **Prepare & Transform** → Train & Tune → Deploy & Manage

Class Imbalance:

How to identify and handle potential class imbalance?



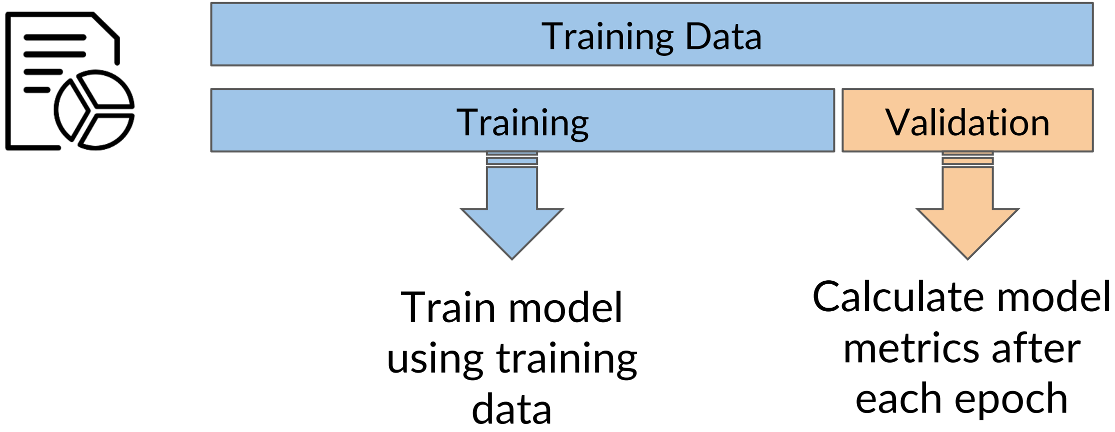


Data Preparation: Train and Validation Data Splits

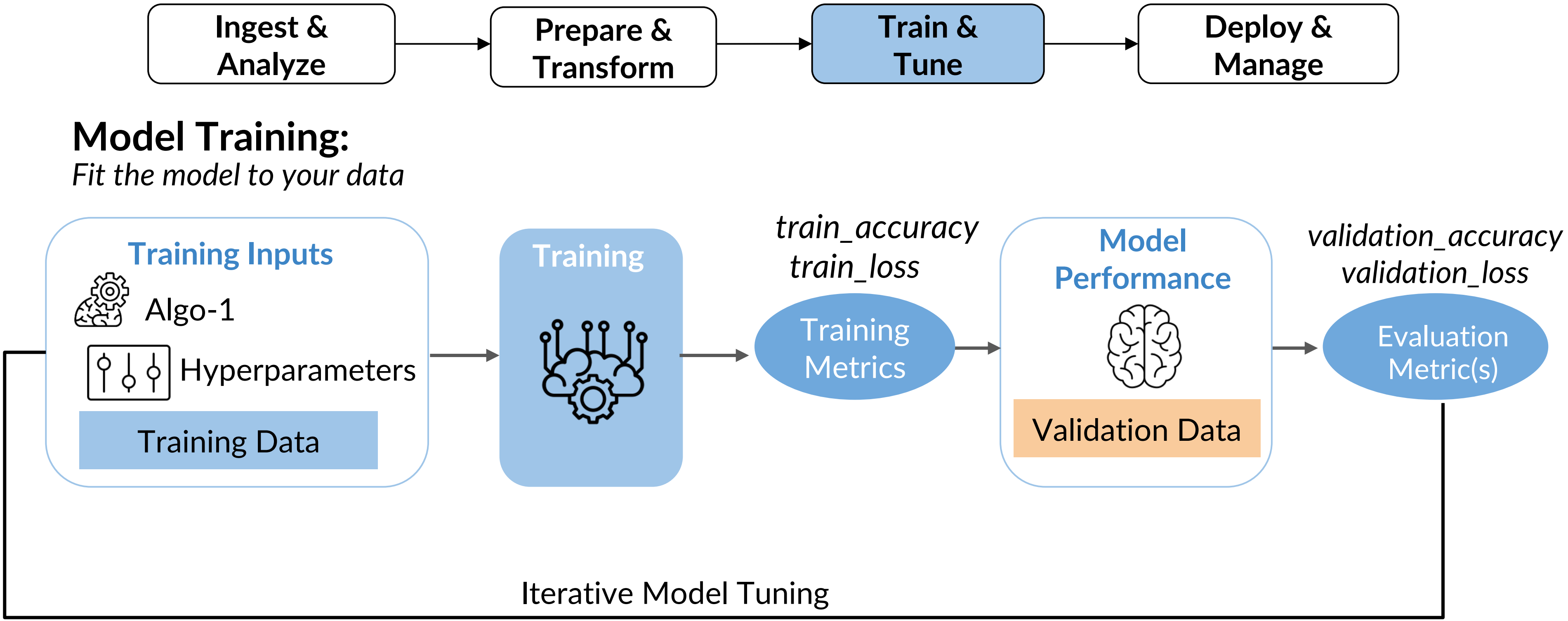
Ingest & Analyze → **Prepare & Transform** → Train & Tune → Deploy & Manage

Train-Validation Splits:

Splitting prepared data for model training, model performance, and final model evaluation



Model Training & Tuning



AutoML

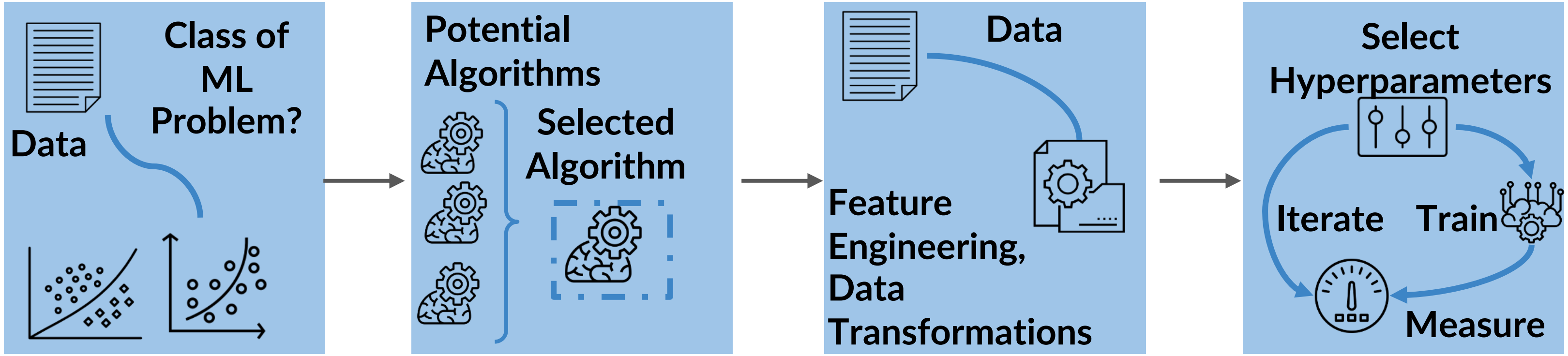
Ingest & Analyze → Prepare & Transform → Train & Tune → Deploy & Manage

AutoML aims at automating the process of building a model

AutoML

Ingest & Analyze → Prepare & Transform → Train & Tune → Deploy & Manage

AutoML aims at automating the process of building a model



Problem Identification | Algorithm Selection | Data Preprocessing | Hyperparameter Tuning

Scenarios for AutoML

Build models without any ML expertise

- Empower more people in your organization: software developers, business people
- Let experts focus on **hard problems**

Experiment and build models at scale

- Thousands of data sets can be modeled without human intervention
- Let experts focus on **new problems**

Automate the majority of the work, then tweak

- Data cleaning, feature engineering, feature selection, etc.
- Let experts focus on high value tasks such as **domain knowledge**, and **error analysis**.

Transparency and Control are Important

Get the **best model** only

- Hard to understand it
- Hard to reproduce it manually

Get the **best model**, **all candidates**, **full source code**

- Understand how the model was built
- Keep tweaking for extra performance

AutoML

with
Amazon SageMaker Autopilot

Introduction

Machine Learning Workflow

Ingest & Analyze	Prepare & Transform	Train & Tune	Deploy & Manage
Data exploration	Feature engineering	Automated ML	Model deployment
Bias detection	Feature store	Model train and tune	Automated pipelines
Amazon S3 & Amazon Athena	Amazon SageMaker Data Wrangler	Amazon SageMaker Autopilot	Amazon SageMaker Endpoints
AWS Glue	Amazon SageMaker Processing Jobs	Amazon SageMaker Training & Debugger	Amazon SageMaker Batch Transform
Amazon SageMaker Data Wrangler & Clarify	Amazon SageMaker Feature Store	Amazon SageMaker Hyperparameter Tuning	Amazon SageMaker Pipelines

AutoML with Amazon SageMaker Autopilot

Amazon SageMaker Autopilot covers all steps:

Ingest & Analyze → Prepare & Transform → Train & Tune → Deploy & Manage

SageMaker Autopilot is fully transparent

Automatically Generated

Notebooks

Code

Amazon SageMaker Autopilot at a High-Level

Share your tabular dataset in a S3 bucket

Dataset,
Target Attribute

Amazon Simple Storage Service
- Amazon S3

Ingest & Analyze

Pre-Processing → Candidate Definitions Generated

SageMaker Autopilot automatically...

Analyzes the data → Calculates Statistics → Identifies ML Problem → Chooses Algorithm → Defines Model Candidate Pipelines → Generates Feature Engineering Code → Generates Notebooks

Problems covered:
- Regression
- Binary classification
- multi-class classification

Linear Learner,
XGBoost
and a deep learning algorithm

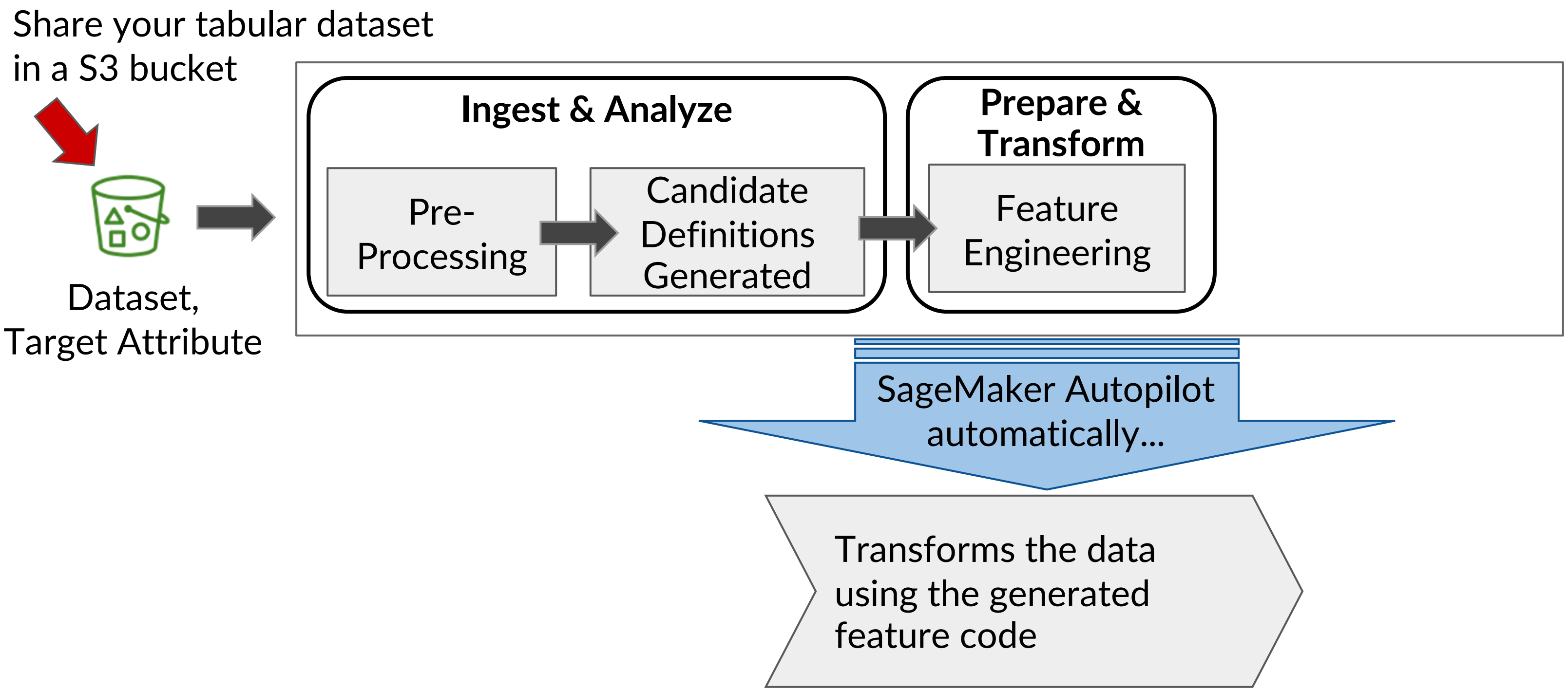
Data Exploration Notebook
- what autopilot learned about your data
- issues in your source data

Candidate generation notebook
- each suggested data preprocessing step
- the algorithm and the hyper parameter ranges that will be used for tuning job.

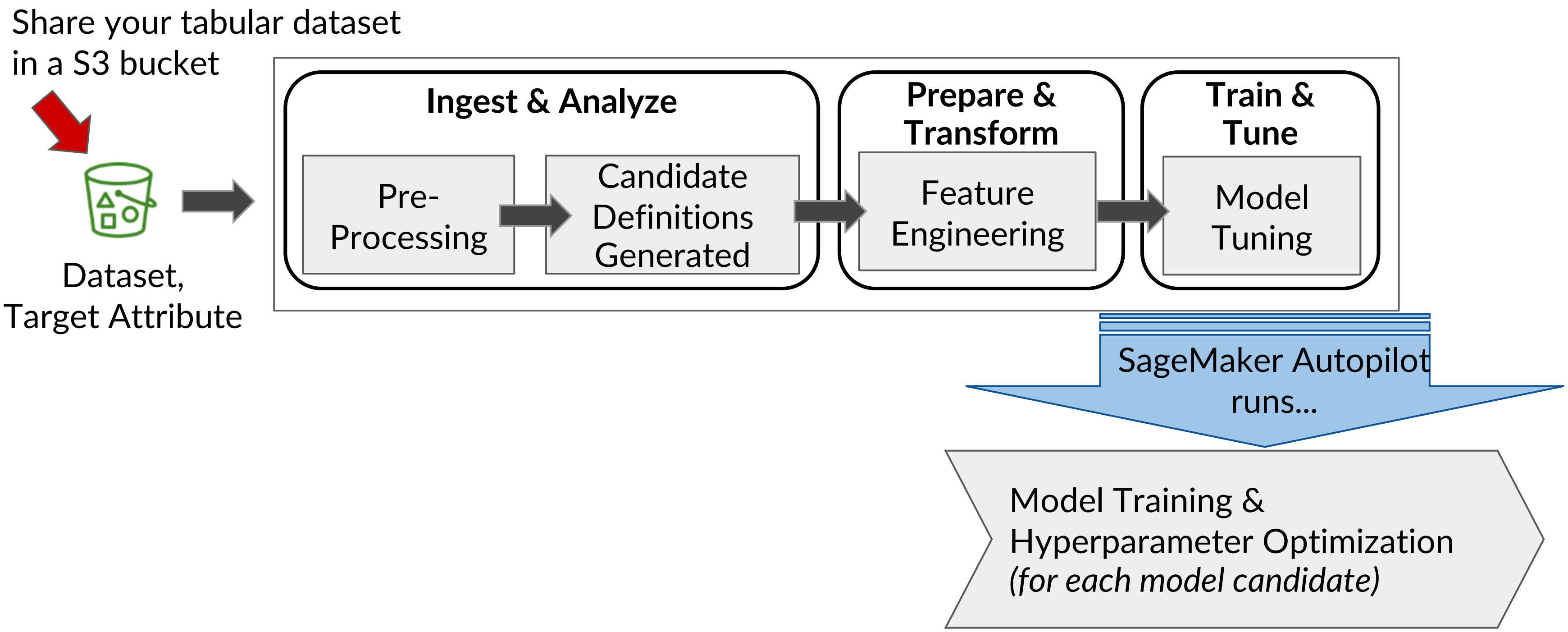
Autopilot can provide models which are more accurate, even when the data sets are highly imbalanced and has few as 500 data points.

Further you can use the SageMaker Autopilot to use the area under the curve, or the area under the receiver operating characteristic curve, as the objective metric, to create even more accurate models

Amazon SageMaker Autopilot at a High-Level

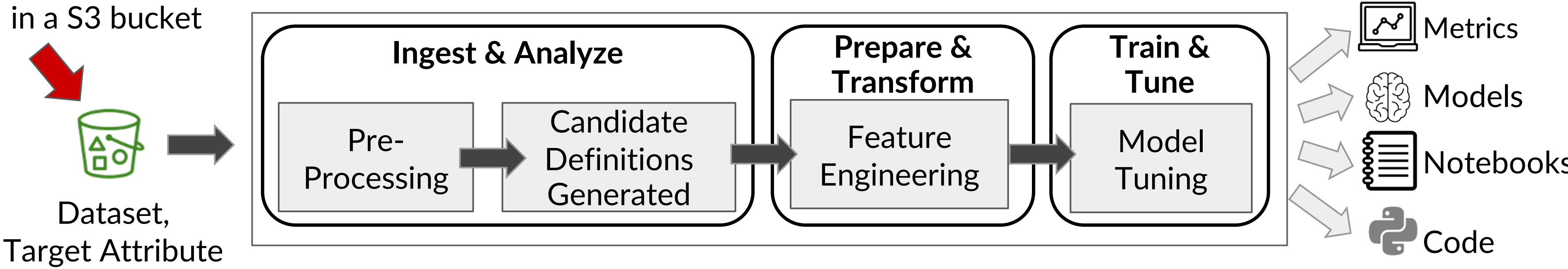


Amazon SageMaker Autopilot at a High-Level



Amazon SageMaker Autopilot at a High-Level

Share your tabular dataset
in a S3 bucket



SageMaker Autopilot shares...

- All metrics
- Leaderboard of model candidates
- Notebooks
- Code

Metrics with a ranked list of recommendations, to determine the best-performing model.

Provides complete visibility into the feature engineering code, the algorithm, and the optimized hyper parameters that were used, which is designed to give you that control and transparency.

AutoML

with
Amazon SageMaker Autopilot

Running Experiments

Amazon SageMaker Autopilot Notebook Overview

Use Case: Analyze Customer Sentiment

Goal: Use SageMaker Autopilot to find the optimal feature transformations, algorithm, and hyperparameters to produce a best performing model allowing us to predict our label (sentiment) based on product reviews (review_body)

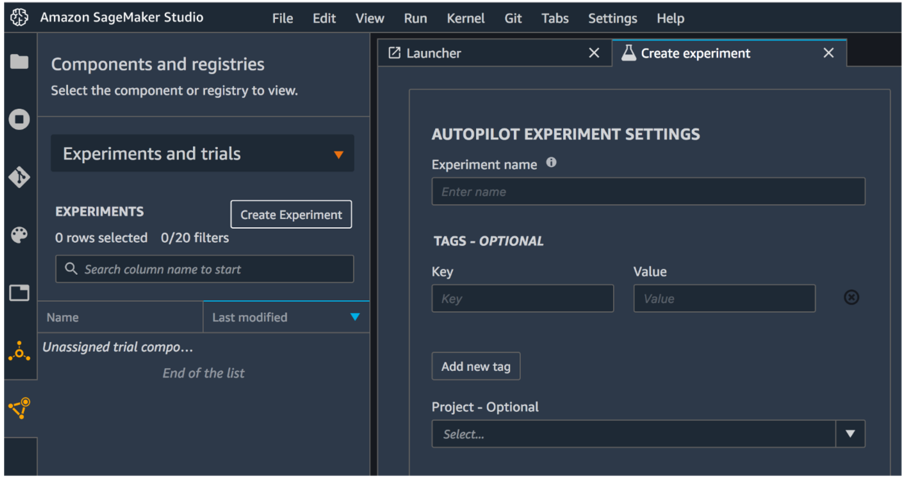
sentiment	review_body
-1	This is bad.
0	This is OK.
1	This is great!

Interacting with Amazon SageMaker Autopilot

~OR~

Programmatically:

1. AWS CLI
2. AWS SDK
3. Amazon SageMaker Python SDK



Amazon SageMaker Studio

Launch the Amazon SageMaker Autopilot Job

```
automl = sagemaker.automl.automl.AutoML(
    target_attribute_name=...
    output_path=..,
    max_candidates=3,
    role=role,
    max_runtime_per_training_job_in_seconds=1200,
    total_job_runtime_in_seconds=7200 # max automl job runtime in seconds
)

automl.fit(
    inputs=...,
)
```

Attribute to predict

Job completion criteria

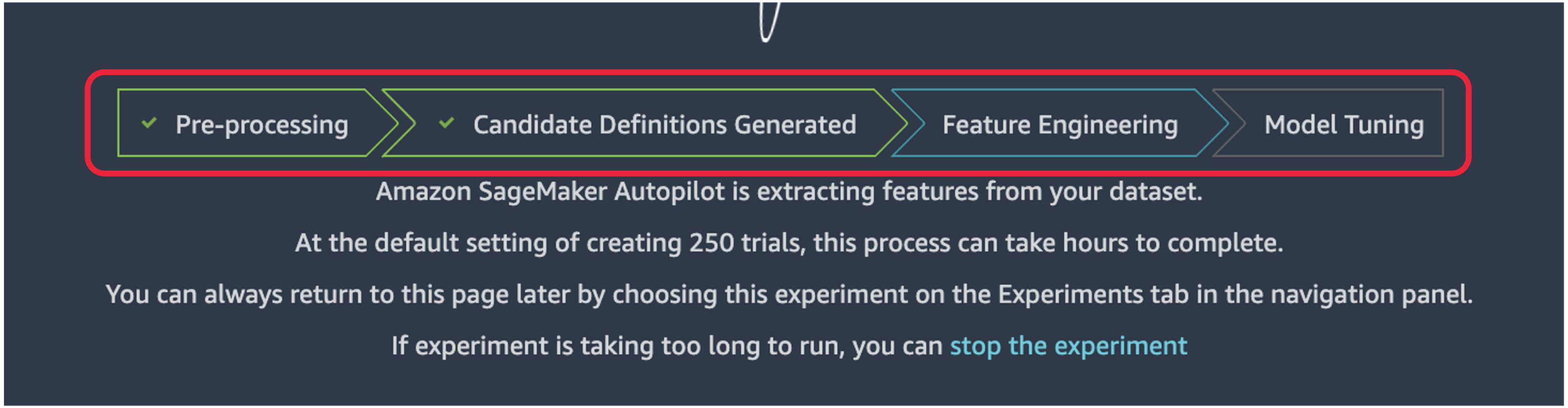
Max. training job run time

Max. AutoML job runtime

Specify input data

specify the S3 bucket that you want to use for your output artifacts. this will contain:
- generated notebooks
- model artifacts
- generate candidate definitions only

Monitor Progress in Amazon SageMaker Studio



API → *DescribeAutoMLJob*

Generated Code for Feature Engineering

- SageMaker Autopilot automatically performs data exploration and prepares the data for the problem type

Example(s):

Transform text features using MultiColumnTfidfVectorizer

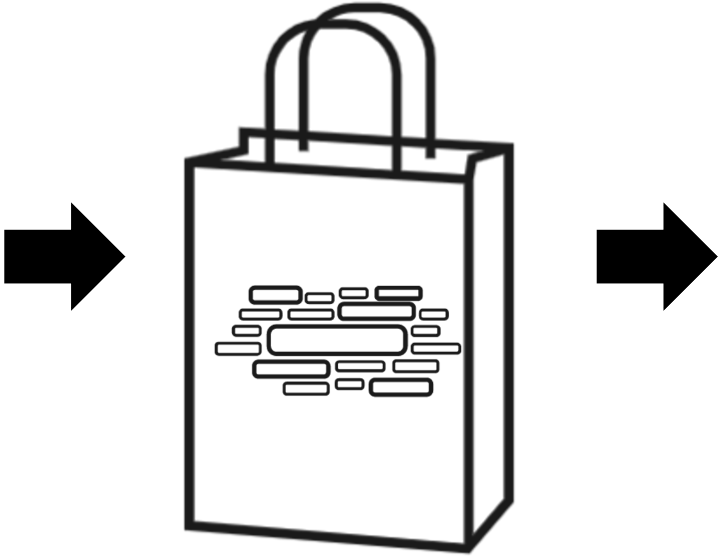
“The Product arrived damaged”	“This product is awesome”	“Broke the first day”

0	.9	0	.6	0	0	0	0	0

- SageMaker Autopilot will automatically tune MultiColumnTfidfVectorizer parameters

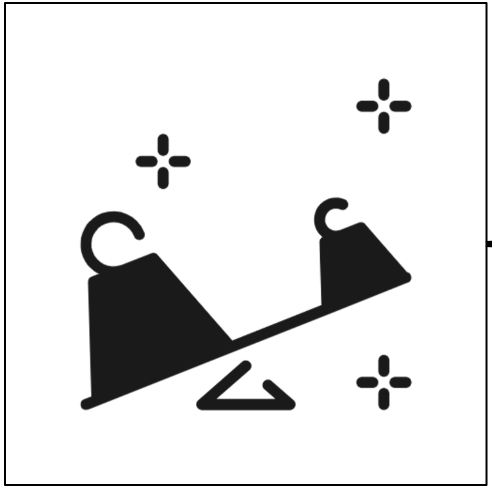
Bag-of-Words: Text as Vectors

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



Term	Term Count
it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
...	...

Text Mining: Measuring Word Importance



Statistical measure of word importance in corpus

↑ proportionally to the number of times a word appears in document

↓ by the frequency of the word in the corpus

tf-idf: term frequency-inverse document frequency

Computing Term Frequency (TF)

t	term
d	document
D	corpus

$$\mathrm{tf}(t,d) = \frac{\boxed{f_{t,d}}}{\boxed{\sum_{t' \in d} f_{t',d}}}$$

Computing Inverse Document Frequency (IDF)

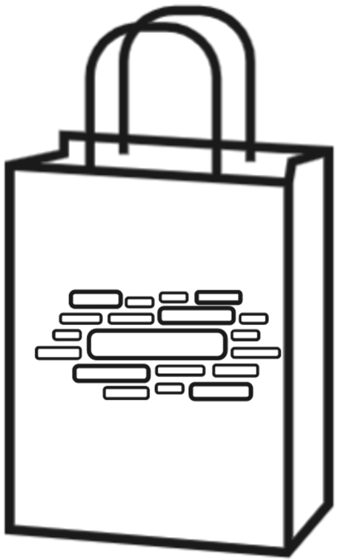
t	term
d	document
D	corpus

$$\mathrm{idf}(t, D) = \log\left(\frac{\boxed{|D|}}{\boxed{|\{d \in D : t \in d\}|}}\right)$$

Putting It All Together: TF-IDF

$$\text{tf-idf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D)$$

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



Term	TF / IDF
it	0.06
I	0.05
the	0.01
to	0.03
and	0.03
seen	0.04
yet	0.01
...	...

AutoML

with
Amazon SageMaker Autopilot

Evaluating Output

SageMaker Autopilot Generates Resources & Artifacts

Code

- Data Transformation Code
- Job Configuration Code

Notebooks

- Data Exploration
- Candidate Generation

Transformed Data

Train

Validation

Processed data used for training

Models

Candidate Models

Metrics

Job #	Objective Metric
tuning-job-1	0.618
tuning-job-2	0.613
tuning-job-3	0.446

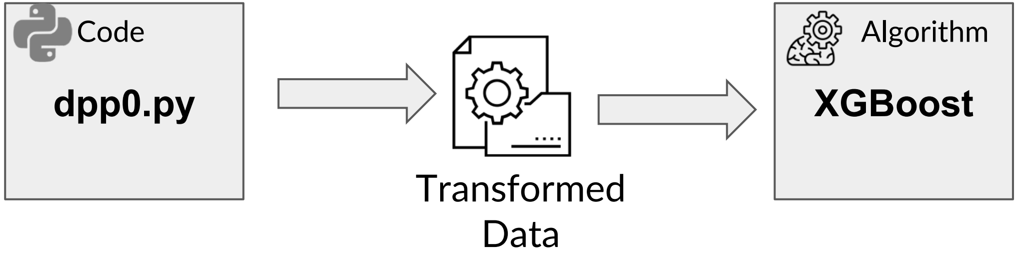
Resources & Artifacts All Accessible in Amazon S3

Model Candidate Pipelines

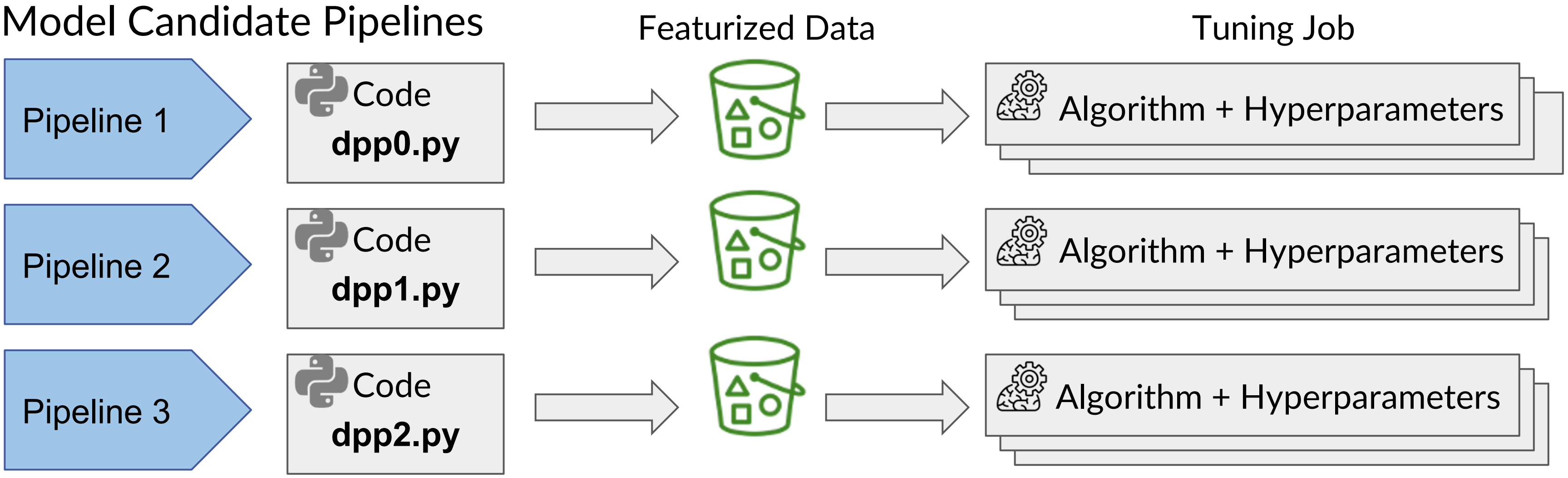
Autopilot generates multiple model candidate pipelines.
An automatically created pipeline contains:
- the feature engineering code.
- the algorithm,
- and the algorithm hyper parameter ranges into the model candidate pipelines.

A model candidate pipeline is composed of

- the feature engineering code (i.e. dpp0.py)
- and an algorithm (i.e. XGBoost).



Executing Model Candidate Pipelines




```
automl_interactive_runner.fit_data_transformers(parallel_jobs=7)
```

Screen recording (Shelbee to insert)

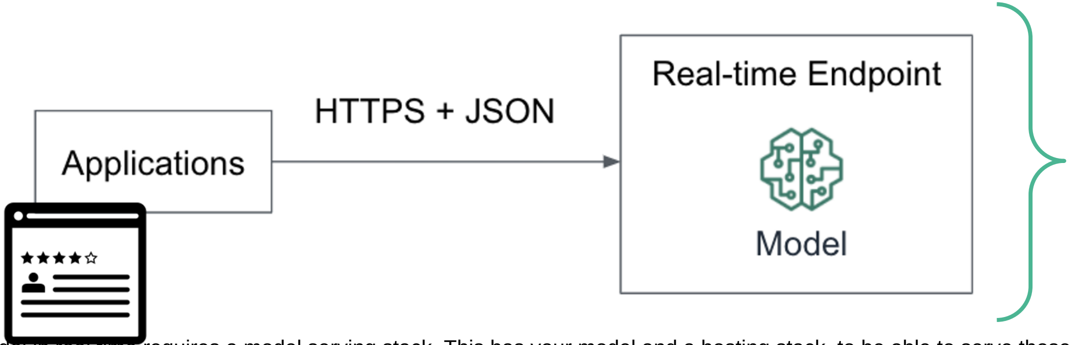
Model Hosting

Introduction

Host a Model Endpoint

Deploy the model to serve predictions in real-time.

- Optimized for **low latency** of model predictions
- Example: As product reviews are coming in through various online channels, you want to predict the sentiment



You choose:

- Instance Type/Size
- Number of Instances
- Autoscaling Options

Serving a model in real time requires a model serving stack. This has your model and a hosting stack, to be able to serve those predictions.
This typically involves some sort of a proxy, a web server that can interact with your loaded serving code and your trained model.
Your model can then be consumed by client applications through real time, invoke endpoint API requests.

With Sagemaker model hosting, you can choose the instance type, as well as the count, combined with the docker/container image that you want use for inference and then Sagemaker takes care of creating the endpoint and deploying that model to the endpoint. You can configure auto scaling.

Deploy Inference Pipeline

```
pipeline_model.deploy(initial_instance_count=1,
                      instance_type='ml.m5.2xlarge',
                      endpoint_name=pipeline_model.name,
                      wait=True)
```

Congratulations! Now you could visit the sagemaker endpoint console page to find the deployed endpoint (it'll take a few minutes to be in service).

The `PipelineModel` has multiple containers of the following:

- **Data Transformation Container:** a container built from the model we selected and trained during the data transformer sections Transformations of the data.
- **Algorithm Container:** a container built from the trained model we selected above from the best HPO training job. container that contains the trained model artifact that was selected as the best-performing model, based on your hyper parameter tuning jobs.
- **Inverse Label Transformer Container:** a container that converts numerical intermediate prediction value back to non-numerical label value.
  Post-process your prediction into a readable value by your application that consumes the output.

Inference Pipeline

When you choose to deploy a candidate pipeline generated by Autopilot, it gets deployed using a Sagemaker hosting feature, called inference pipeline. With inference pipeline, you are able to host your data transformation model, your product classification model, and your inverse label transformer, behind the same endpoint.
This allows you to keep your training and inference code in sync and allows you to abstract your transformations for consuming applications.