A CLOUD GURU

Other Feature Engineering

Brock Tubre
INSTRUCTOR

Images

Image Feature Engineering

Extracting useful information from images before using them with ML algorithms.

Transforming images to find out useful information.

Images

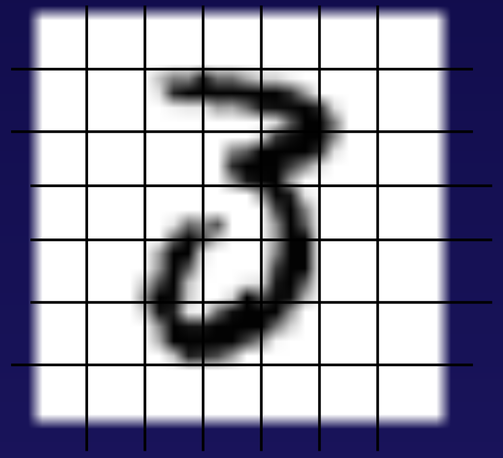
Problem

Is this character a number?

Images

Problem

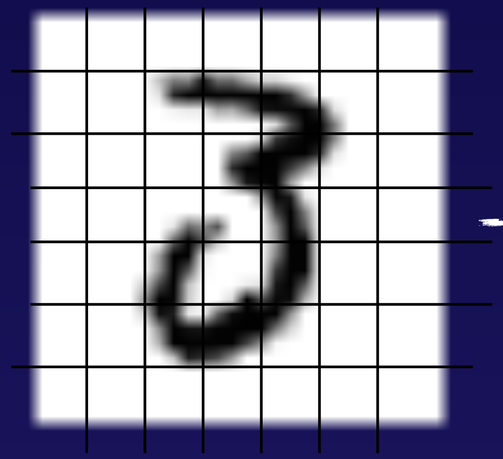
Is this character a number?



Images

Problem

Is this character a number?

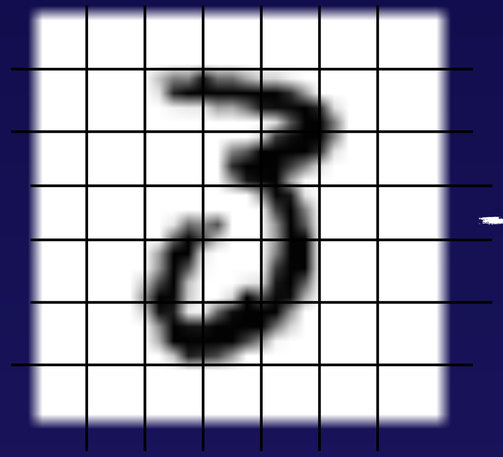


0	0	0	0	0	0	0
0	0	1	1	1	1	0
0	0	0	1	1	0	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

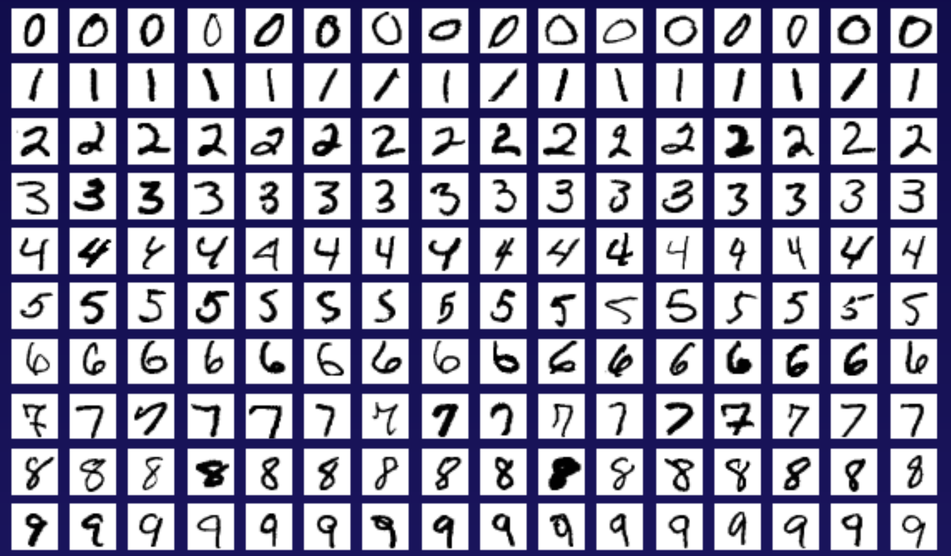
Images

Problem

Is this character a number?



0	0	0	0	0	0	0
0	0	1	1	1	1	0
0	0	0	1	1	0	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0



MNIST data

Compare against other images…

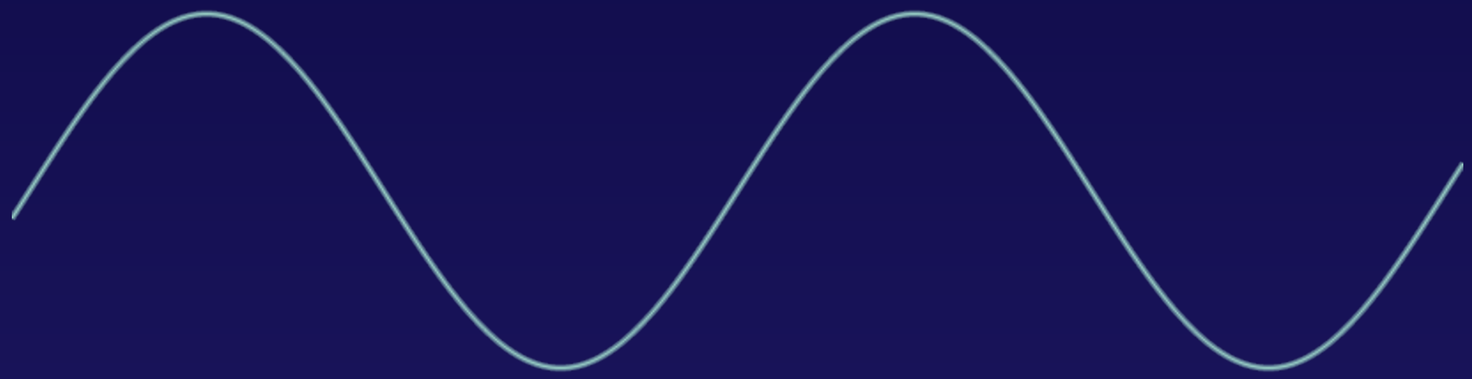
Audio Feature Engineering

Extracting useful information from sounds and audio before using them with ML algorithms.

Transforming audio data into something useful.

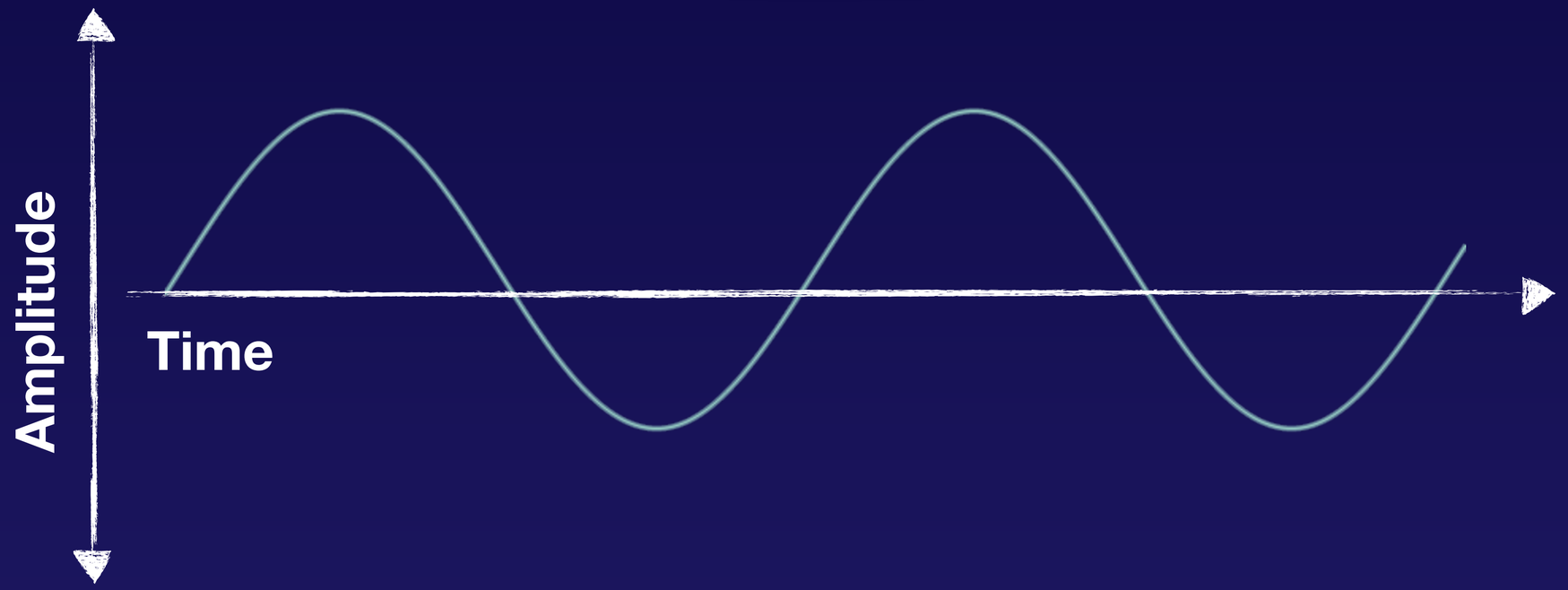
Audio Stream Example

Analog Audio Stream



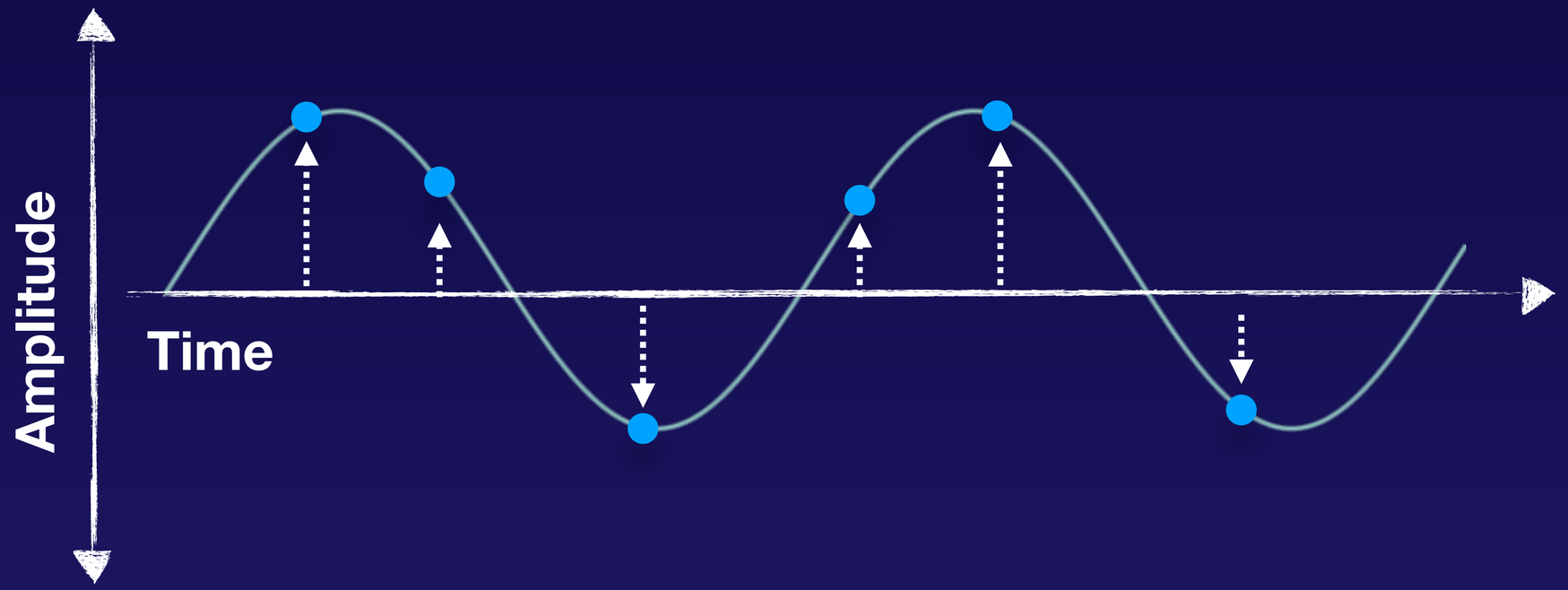
Audio Stream Example

Analog Audio Stream



Audio Stream Example

Analog Audio Stream



Time	Amplitude
0:03	4
0:06	2
0:09	-4
0:12	1
0:14	4
0:16	-3
...	...

Dataset Formats

File

Pipe

Dataset Formats

File

Pipe

- Loads all of the data from S3 directly onto the training instance volumes.
- CSV
- JSON
- Parquet
- Image files (.png or .jpg)

Dataset Formats

File

- Loads all of the data from S3 directly onto the training instance volumes.
- CSV
- JSON
- Parquet
- Image files (.png or .jpg)

Pipe

- Your datasets are streamed directly from Amazon S3.
- recordIO-protobuf (creates tensor)

Dataset Formats


```
from sagemaker.amazon.common import write_numpy_to_dense_tensor
import io
import boto3

bucket = 'bucket-name' # Use the name of your s3 bucket here
data_key = 'kmeans_lowlevel_example/data'
data_location = 's3://{}/{}'.format(bucket, data_key)

# Convert the training data into the format required by the algorithm
buf = io.BytesIO()
write_numpy_to_dense_tensor(buf, train_set[0], train_set[1])
buf.seek(0)

# location to upload to recordIO-protobuf data
boto3.resource('s3').Bucket(bucket).Object(data_key).upload_fileobj(buf)
```

Dataset Formats


```
from sagemaker.amazon.common import write_numpy_to_dense_tensor
import io
import boto3

bucket = 'bucket-name' # Use the name of your s3 bucket here
data_key = 'kmeans_lowlevel_example/data'
data_location = 's3://{}/{}'.format(bucket, data_key)

# Convert the training data into the format required by the algorithm
buf = io.BytesIO()
write_numpy_to_dense_tensor(buf, train_set[0], train_set[1])
buf.seek(0)

# location to upload to recordIO-protobuf data
boto3.resource('s3').Bucket(bucket).Object(data_key).upload_fileobj(buf)
```

Dataset Formats


```
from sagemaker.amazon.common import write_numpy_to_dense_tensor
import io
import boto3

bucket = 'bucket-name' # Use the name of your s3 bucket here
data_key = 'kmeans_lowlevel_example/data'
data_location = 's3://{}/{}'.format(bucket, data_key)

# Convert the training data into the format required by the algorithm
buf = io.BytesIO()
write_numpy_to_dense_tensor(buf, train_set[0], train_set[1])
buf.seek(0)

# location to upload to recordIO-protobuf data
boto3.resource('s3').Bucket(bucket).Object(data_key).upload_fileobj(buf)
```

Imports

Where we want
to store recrodIO-protobuf

Dataset Formats


```
from sagemaker.amazon.common import write_numpy_to_dense_tensor
import io
import boto3

bucket = 'bucket-name' # Use the name of your s3 bucket here
data_key = 'kmeans_lowlevel_example/data'
data_location = 's3://{}/{}'.format(bucket, data_key)

# Convert the training data into the format required by the algorithm
buf = io.BytesIO()
write_numpy_to_dense_tensor(buf, train_set[0], train_set[1])
buf.seek(0)

# location to upload to recordIO-protobuf data
boto3.resource('s3').Bucket(bucket).Object(data_key).upload_fileobj(buf)
```

Imports

Where we want to store recrodIO-protobuf

Data format conversion

Dataset Formats


```
from sagemaker.amazon.common import write_numpy_to_dense_tensor
import io
import boto3

bucket = 'bucket-name' # Use the name of your s3 bucket here
data_key = 'kmeans_lowlevel_example/data'
data_location = 's3://{}/{}'.format(bucket, data_key)

# Convert the training data into the format required by the algorithm
buf = io.BytesIO()
write_numpy_to_dense_tensor(buf, train_set[0], train_set[1])
buf.seek(0)

# location to upload to recordIO-protobuf data
boto3.resource('s3').Bucket(bucket).Object(data_key).upload_fileobj(buf)
```

Imports

Where we want to store recrodIO-protobuf

Data format conversion

Upload to S3

Layering Feature Engineering

Feature Engineering is like a layered cake. Usually there are multiple layers of transformations done to properly prepare your data.

Layering Feature Engineering

Feature Engineering is like a layered cake. Usually there are **multiple** layers of transformations done to properly **prepare** your **data**.