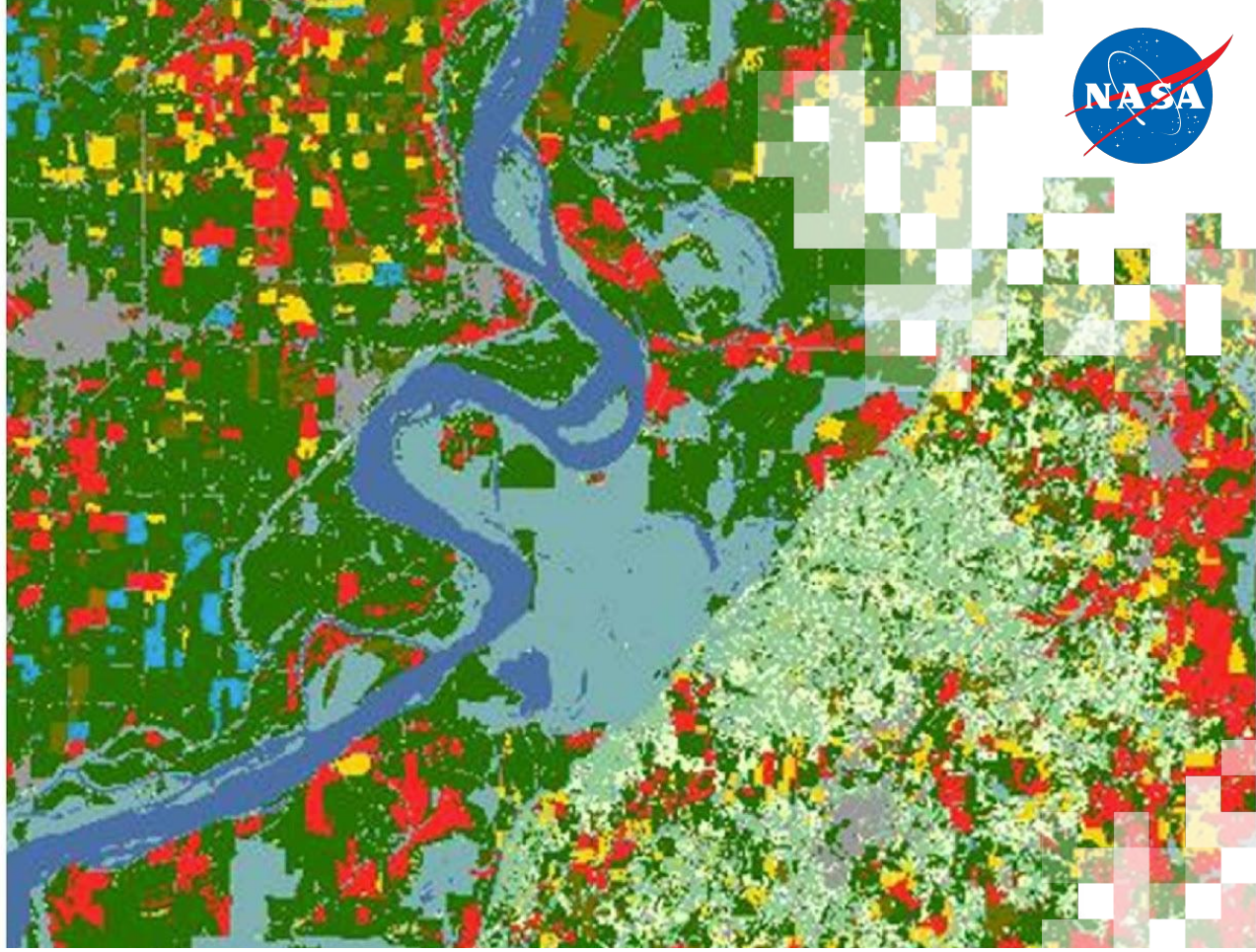
National Aeronautics and Space Administration

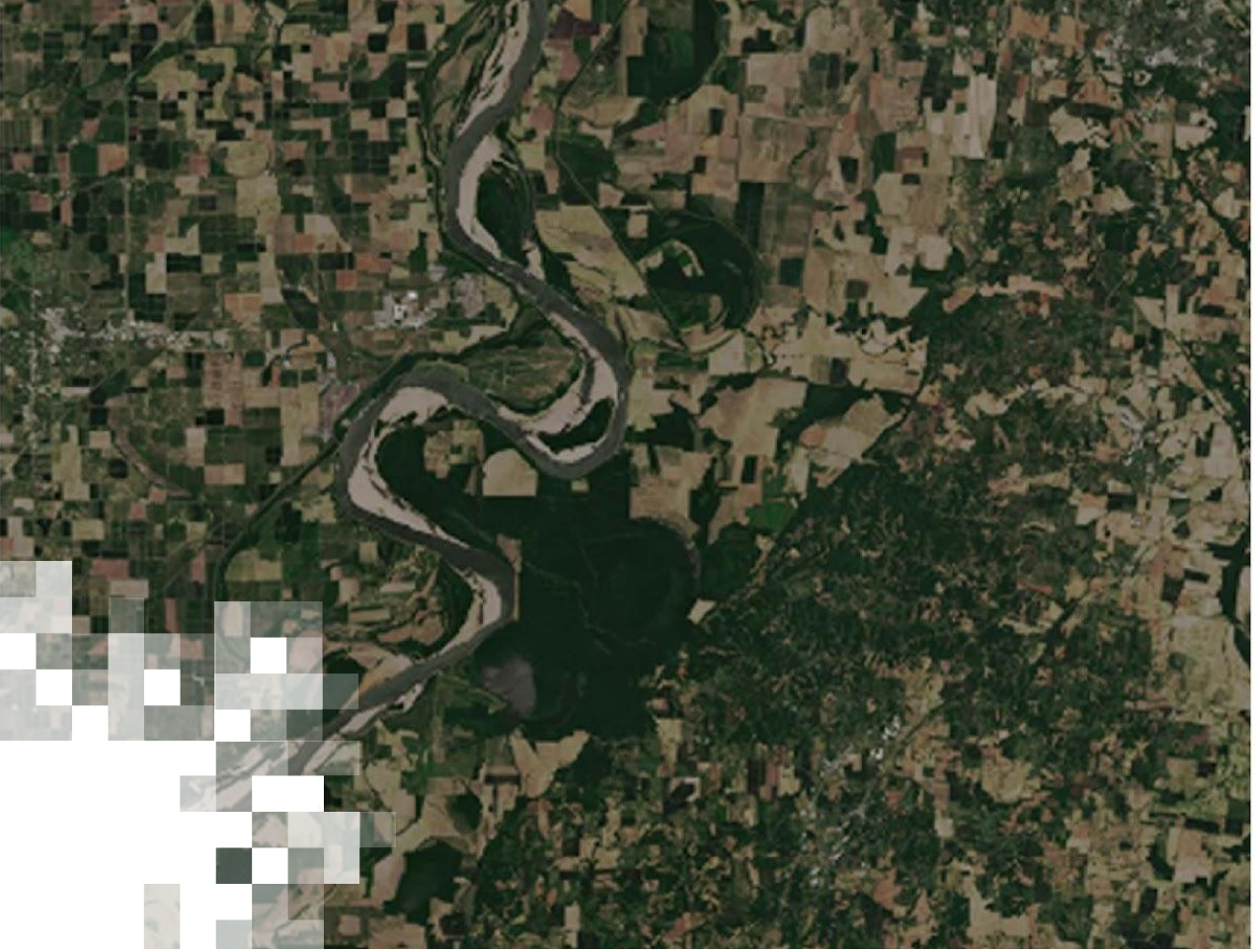


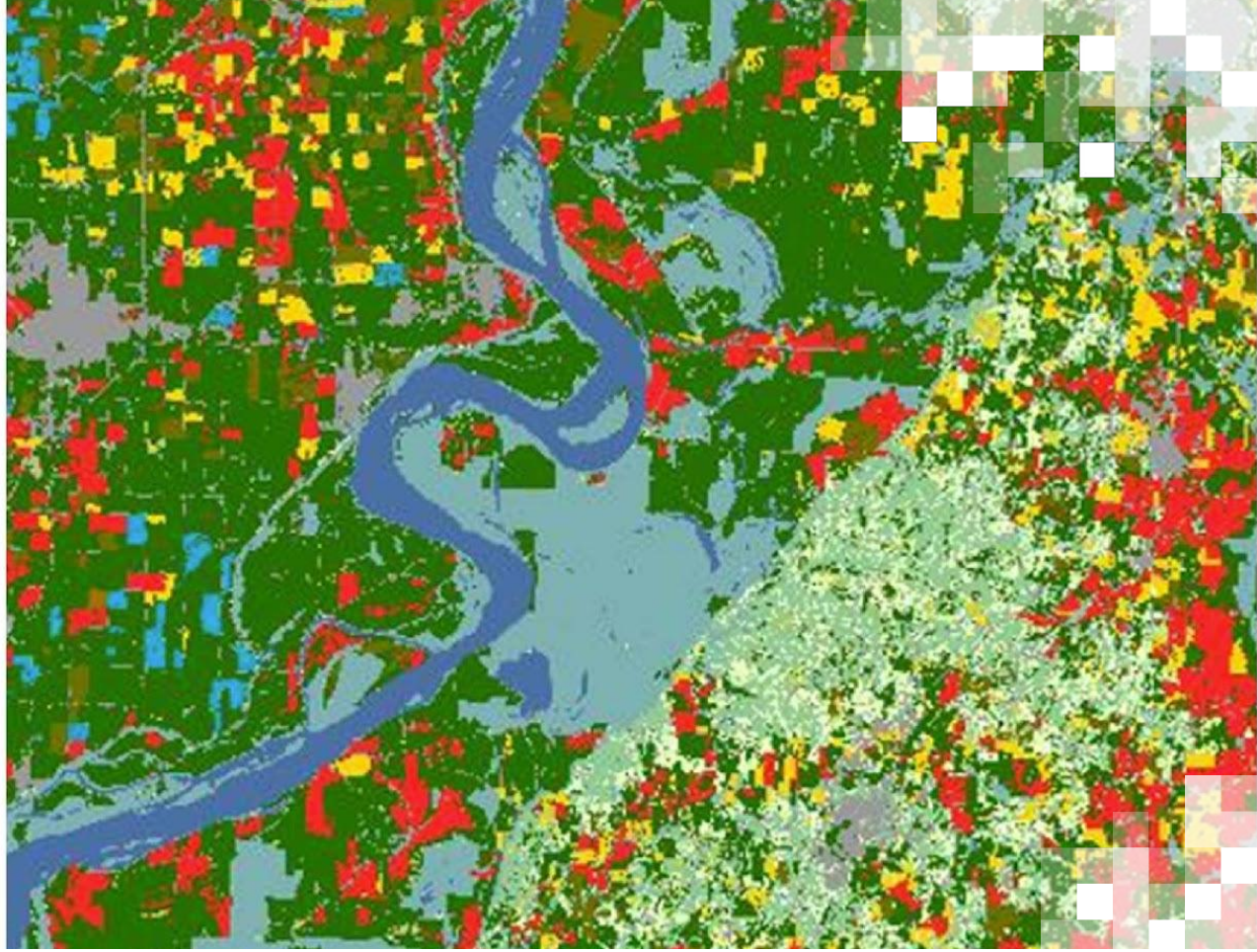
Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

2da Parte: Cargadores de Datos para Entrenar Modelos de Aprendizaje Automático sobre Series Temporales de Imágenes Espaciadas Irregularmente

John Just (Deere y Cia., Universidad Estatal de Iowa), Erik Sorensen (Deere y Cia.)

12 de marzo de 2024





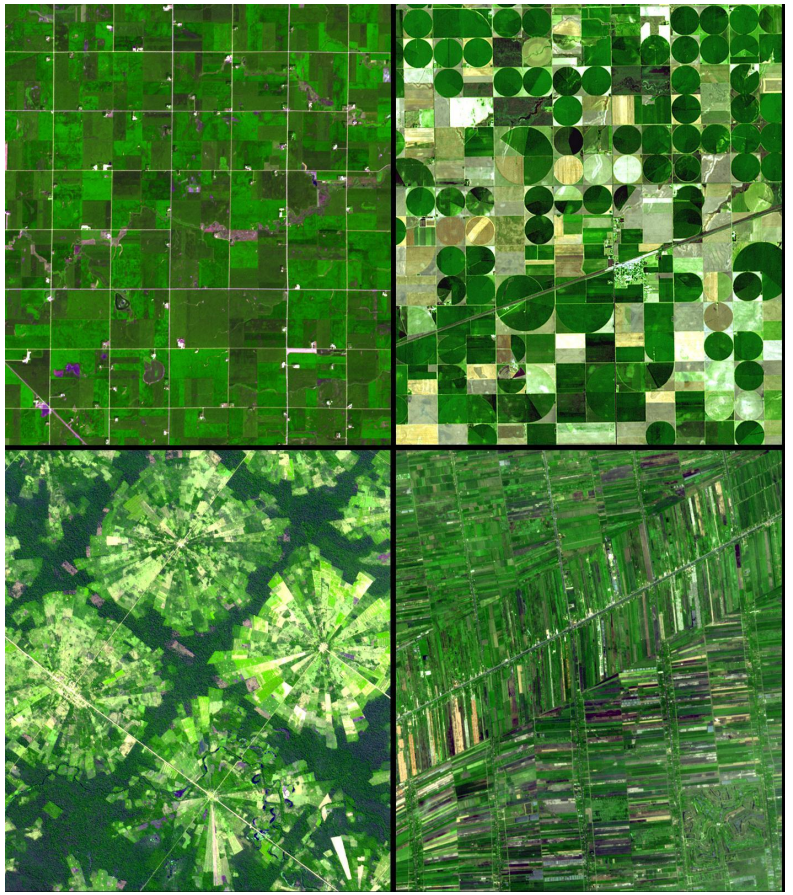
Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

Resumen General

Motivación para Esta Capacitación

- Los mapas de cultivos oportunos y precisos dentro de la temporada, a escala local y regional, son cruciales para la toma de decisiones y la gestión agrícola.
  Las series temporales espaciadas irregularmente son comunes en las imágenes satelitales ópticas.
  El entrenamiento robusto de modelos con datos de teledetección a menudo requiere datos muy grandes, pero el procesamiento y el entrenamiento son complejos.

- La capa Cropland Data Layer* (CDL, USDA–NASS) solo proporciona estimaciones de los tipos de cultivos que se dan a conocer al público unos meses después del final de la temporada de crecimiento, y no su secuencia o cronología (por ejemplo, para cultivos dobles)



Montaje de imágenes mostrando diferencias en la geometría y el tamaño de los campos agrícolas en diferentes partes del mundo. Fuente de la imagen: NASA (Instrumento: Terra – ASTER)

*Capa de datos de tierras de cultivo, en inglés

Objetivos de Aprendizaje para Esta Capacitación

Al final de esta serie, las/los participantes habrán desarrollado la capacidad para:

- Utilizar las técnicas recomendadas para descargar y procesar datos de teledetección de Sentinel-2 y la capa Cropland Data Layer (CDL) a gran escala (> 5 GB) con herramientas en la nube (Amazon Web Services [AWS] Simple Storage Service [S3], Databricks, Spark/Pyspark, Parquet)
  Producir gráficos interactivos de mapas, tablas, series temporales etc. para la investigación y verificación de datos y modelos.
  Filtrar datos de los dominios medidos (imágenes satelitales) y el objetivo (CDL) para cumplir con los objetivos de modelación en base a factores de calidad, clasificación de tierras, superposición de áreas de interés (AOI, por sus siglas en inglés) y ubicación geográfica.
  Crear canalizaciones de entrenamiento en TensorFlow para entrenar algoritmos de aprendizaje automático en conjuntos de datos geoespaciales/de teledetección a gran escala para el monitoreo agrícola
  Utilizar técnicas de muestreo aleatorio para crear solidez en un algoritmo predictivo y, al mismo tiempo, evitar la fuga de información en las divisiones de entrenamiento/ validación/prueba

Prerrequisitos

- Entendimiento básico/general de la programación con Python en Databricks de la 1ra Parte.
- Acceso a los datos asociados de la 1ra parte.
- Inscribirse y acceder a Databricks Community Edition

Esquema de la Capacitación

1ra Parte

Preparación de Datos para Imágenes y Etiquetas para la Modelación con Aprendizaje Automático a Gran Escala

5 de marzo de 2024

2da Parte

Cargadores de Datos para Entrenar Modelos de Aprendizaje Automático sobre Series Temporales de Imágenes Espaciadas Irregularmente

12 de marzo de 2024

3ra Parte

Entrenamiento y Prueba de Modelos de Aprendizaje Automático para Series Temporales de Imágenes Espaciadas Irregularmente

19 de marzo de 2024

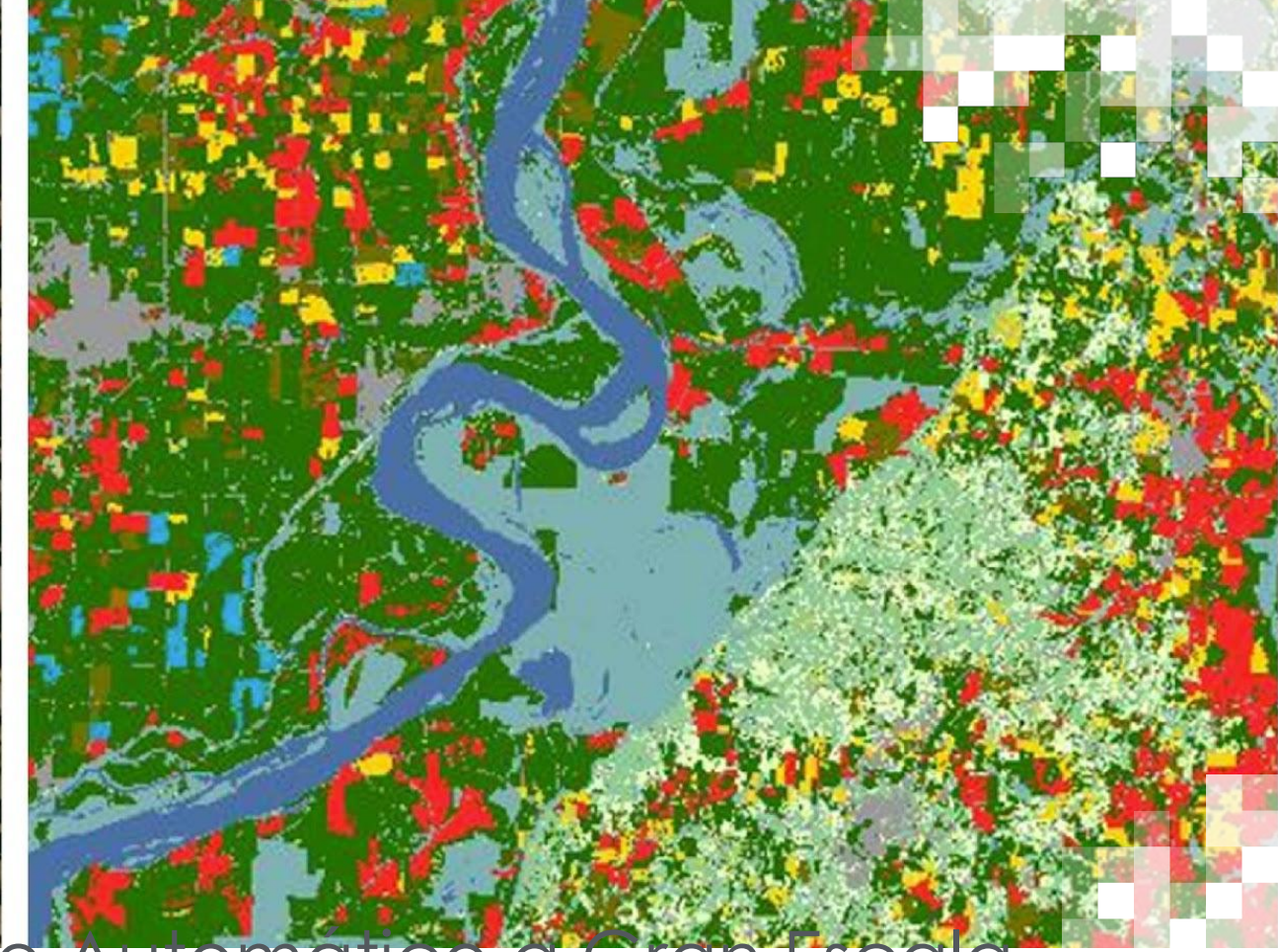
Tarea

Abre el 19 de marzo – **Fecha Límite: 1ro de abril** – Publicada en la Página Web de la Capacitación

Se otorgará un certificado de finalización de curso a quienes asistan a todas las sesiones en vivo y completen la tarea asignada antes de la fecha estipulada.

Cómo Hacer Preguntas

- Por favor escriba sus preguntas en la casilla denominada “Questions” y las responderemos al final de este webinar.
- No dude en escribir sus preguntas mientras vayamos avanzando. Intentaremos responder todas las preguntas durante la sesión para preguntas y respuestas después del webinar.
- Las demás preguntas las responderemos en el documento de preguntas y respuestas, el cual será publicado en la página web de la capacitación aproximadamente una semana después de esta.



Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

2da Parte: Cargadores de Datos para Entrenar Modelos de Aprendizaje Automático sobre Series Temporales de Imágenes Espaciadas Irregularmente

2da Parte – Formadores

John Just

Científico Informático Principal

John Deere

Erik Sorensen

Científico Informático Sénior

John Deere

Objetivos – 2da Parte

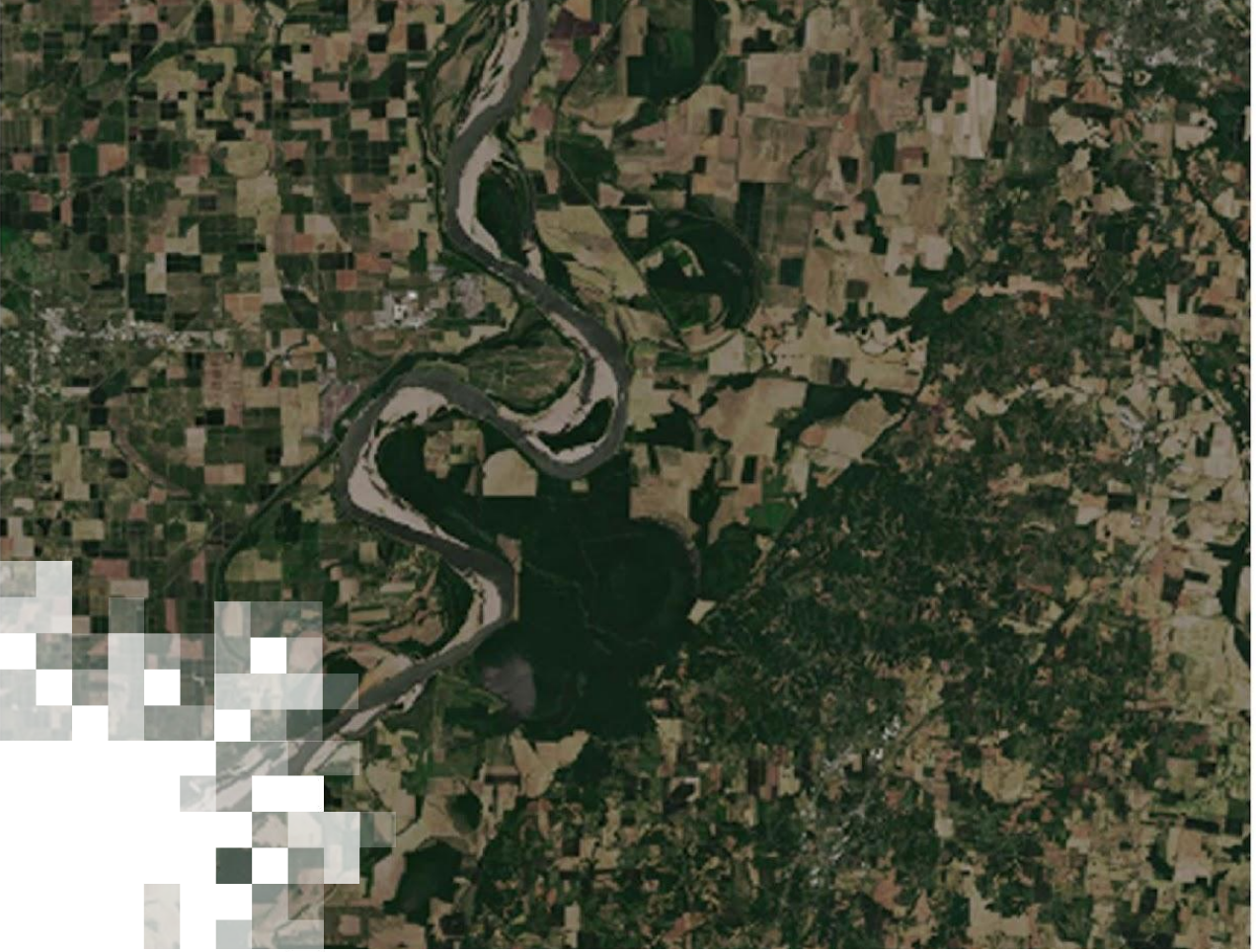
Al final de la 2dda Parte, las/los participantes podrán usar Python en Databricks Community Edition para hacer lo siguiente:

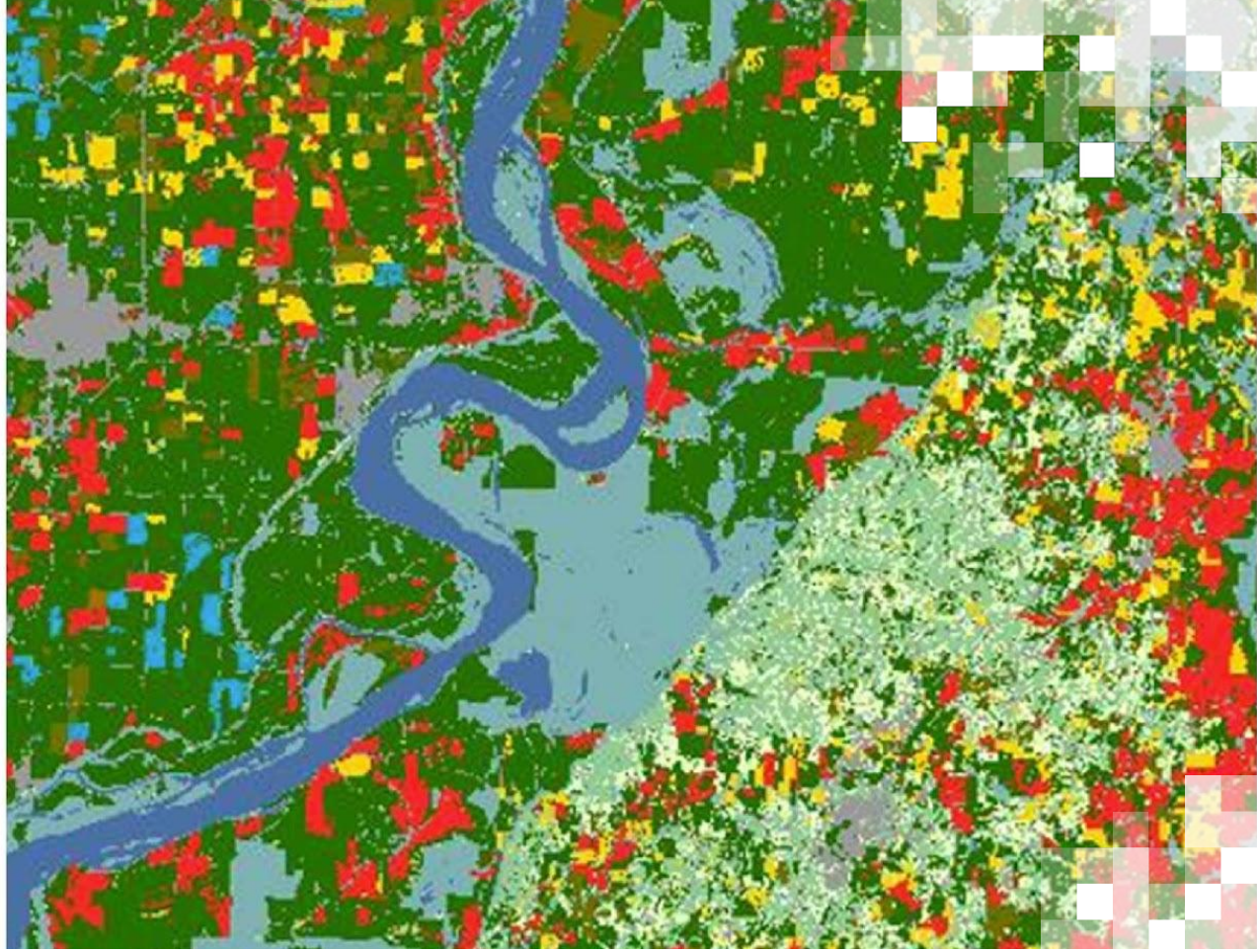
- Dividir el conjunto de datos en grupos de train/val/test* para evitar la pérdida de datos a través del tiempo y el espacio.
  Crear una cola especializada/optimizada (cargador de datos de TensorFlow):
  - Extraer/leer archivos Parquet en series temporales de píxeles (con selección aleatoria).
    Filtrar cualquier dato que queramos ignorar (por ejemplo, cultivos dobles).
    Aplicar aumentos (por ejemplo, selecciones aleatorias de fechas para cada píxel).
    Agrupar series temporales espaciadas irregularmente por píxel en intervalos de tiempo regulares.
    Normalizar valores para facilitar la convergencia durante la optimización.
- Optimizar la canalización de datos para ganar velocidad mediante la paralelización y los beneficios de la vista.

*entrenamiento/validación/prueba

Repaso de Conocimiento Adquirido Anteriormente

- Las imágenes satelitales tienen una irregularidad elevada en cuanto a series temporales debido a fenómenos como la nubosidad.
  La capa de datos de tierras de cultivo (CDL, por sus siglas en inglés) es un modelo que proporciona etiquetas de los cultivos que crecen en los EE. UU. contiguos y se publica a finales de año.
  El almacenamiento de datos como cadenas de bytes en archivos Parquet permite un almacenamiento eficiente de datos de series temporales grandes.





2da Parte, Sección 1:

Acerca de TensorFlow y el Flujo de Datos

TensorFlow

Biblioteca de fuente abierta de Google para crear canalizaciones de aprendizaje automático escalables de principio a fin: Tensorflow.org

2da Parte (esta parte)

3ra Parte (la próxima)

Otras capacidades (no usadas en la demo)

Prepare data

Use TensorFlow tools to process and load data.

Build ML models

Use pre-trained models or create custom ones.

Deploy models

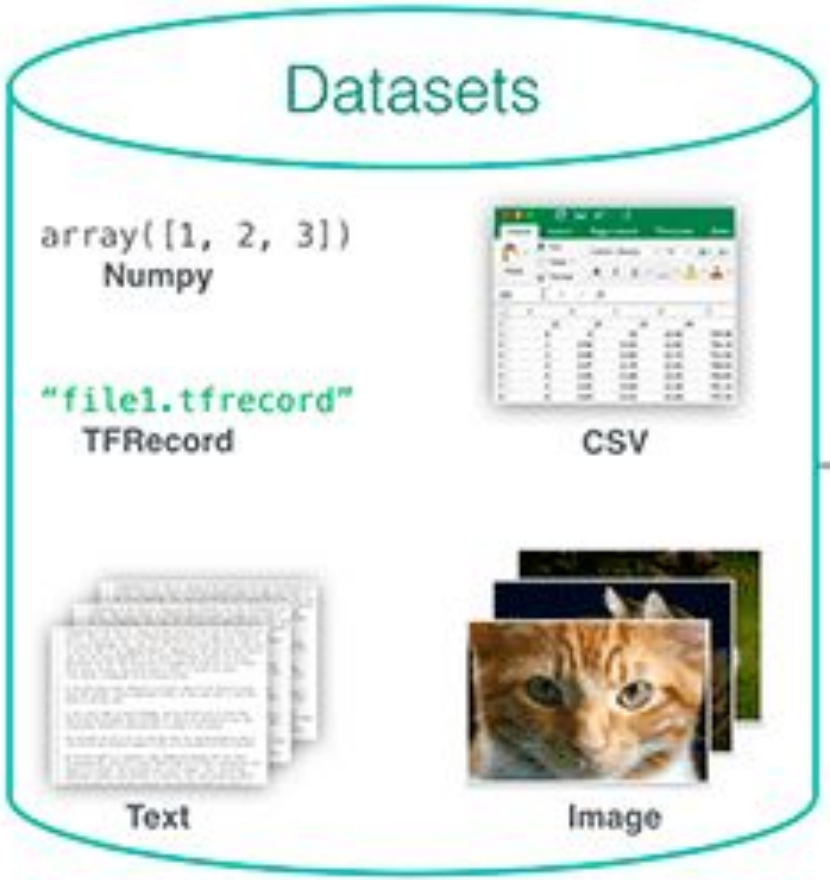
Run on-prem, on-device, in the browser, or in the cloud.

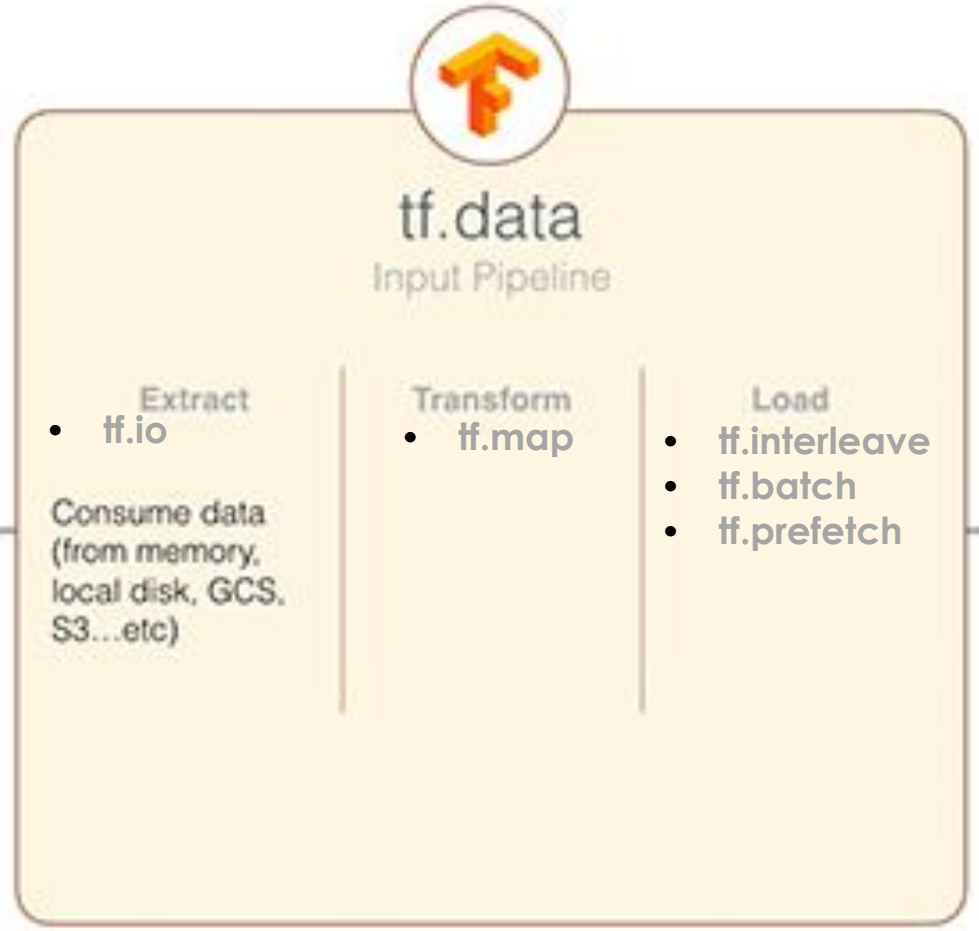
Implement MLOps

Run models in production and keep them performing.

Flujo de Datos

Nos encontramos aquí

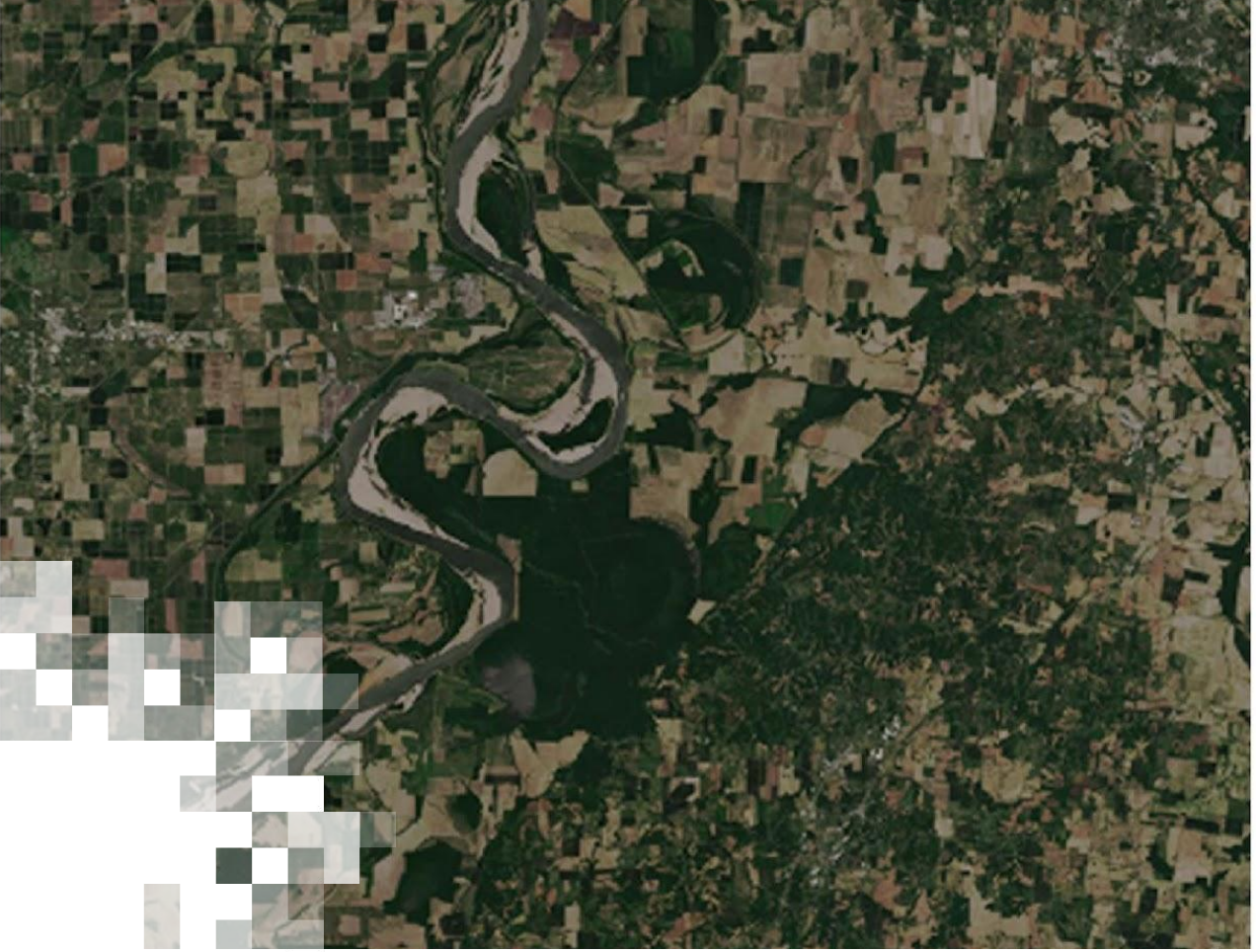


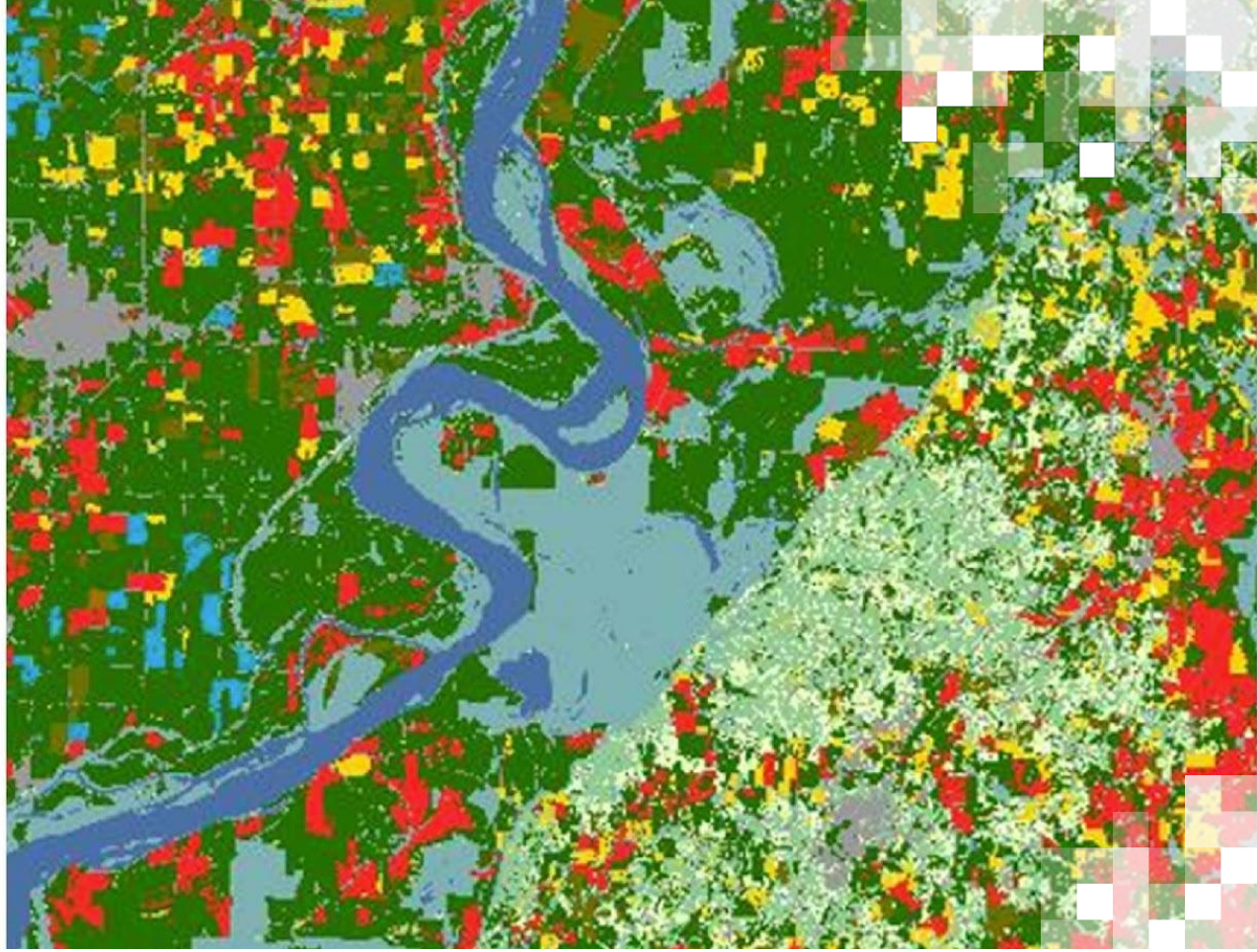


1ra Parte (construir un conjunto de datos)

2da Parte (construir un cargador de datos)

3ra Parte (entrenamiento del modelo e inferencia)





2da Parte, Sección 2:

Canalizaciones de Datos en TensorFlow

Resumen General de la Canalización de Datos

Existen varias buenas explicaciones para las canalizaciones de datos en Tensorflow (TF), p.ej., el Stanford CS230 (de Andrew Ng) y directamente de los documentos de TF. Según las sugerencias del CS230, lo que sigue es un orden de operaciones recomendable:

1. Adquirir una lista de ubicaciones de rutas de archivos
2. Leer el contenido de cada archivo como conjunto de datos de TensorFlow. Puede ser un generador (carga diferida de archivos) o en memoria (filas de una tabla). Para esta demostración, cargamos los datos en memoria debido al rendimiento en DB Community
3. .filter()
   - Filtra cualquier dato que no queramos durante en entrenamiento (p.ej., etiquetas de doble cultivo)
4. .shuffle()
   - Mezcla aleatoriamente el orden de los datos para diversificar las muestras de cada lote y, en última instancia, encontrar actualizaciones de gradiente más sólidas
5. .map(parse function)
   - Se aplica a cada píxel/año para muestrear, agrupar los valores de la imagen y prepararlos junto con las etiquetas para la modelación
6. .batch (datos de entrenamiento y salidas de etiquetas)
   - Carga "N" (tamaño de lote) número de ejemplos del mapa cada vez que se exigen datos (call)
7. .prefetch
   - Hace todo lo anterior en procesos en segundo plano (cola paralela "productor") desde el proceso principal ("consumidor", donde se produce el entrenamiento del modelo) para minimizar el tiempo de espera para nuevos lotes de entrenamiento

Ejemplo de Código para Cargar Datos a TensorFlow

```
train_ds = train_files_ds \
        .filter(filter_double_croppings) \
        .shuffle(BATCH_SIZE*10) \
        .map(lambda x: parse(x, norm=True), num_parallel_calls = tf.data.AUTOTUNE) \
        .batch(BATCH_SIZE) \
        .prefetch(1) \
```

NOTE: **num_parallel_calls** controla el número de CPUs que se utilizan para la paralelización de la función de mapeo

.map(función)

Map es donde se realiza la mayor parte del trabajo en el cargador de datos. Puede paralelizar la aplicación de una función [personalizada] a los datos de insumo. En nuestro caso, aplicamos una función de análisis que:

- Lee en cada fila de los archivos Parquet,
  Convierte características de datos de bytes sin procesar en tf.tensors usandotf.io.decode_raw()
  - La lectura de datos como bytes garantiza que los datos se carguen correctamente,
- Agrupa datos (selecciona aleatoriamente 1 imagen si hay más de 1 disponible por intervalo) y los forma en una matriz (num buckets)*(num bands),
  Realiza la normalización (importante para el proceso de optimización),
  Realiza codificación "one-hot" sobre las etiquetas (la función de pérdida espera que las etiquetas estén en este formato), y
  Devuelve tupla de entidades y etiquetas de datos.

Lectura de Cadenas de Bytes con Tensorflow.IO

- Recuerden de la 1ra Parte
  - Las series temporales de imágenes se agruparon por cada píxel CDL/año Estas series temporales se convirtieron en cadenas de bytes para un almacenamiento más eficiente
- La mayoría de los tipos de datos (por ejemplo, imágenes) implican decodificación. TF tiene funciones para decodificar las cadenas de bytes usando tf.io.decode_raw()
  decode_raw() solo necesita el **tipo de datos** anticipado de (por ejemplo, float32, string) y convertirá la cadena de bytes de nuevo en la matriz multidimensional original
  - Forma de arreglo: [num_images, num_bands]

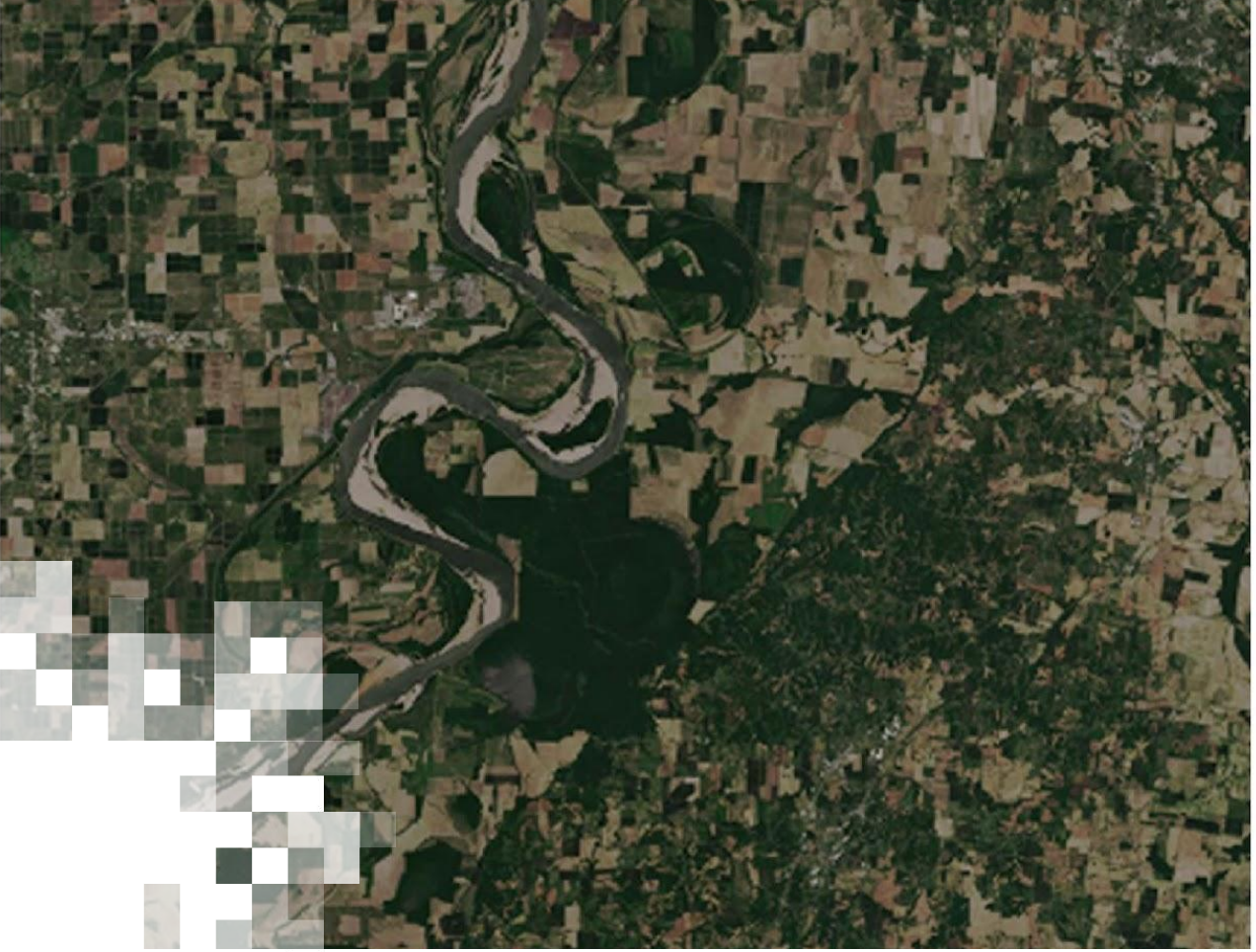
Cadenas de Bytes de Insumo

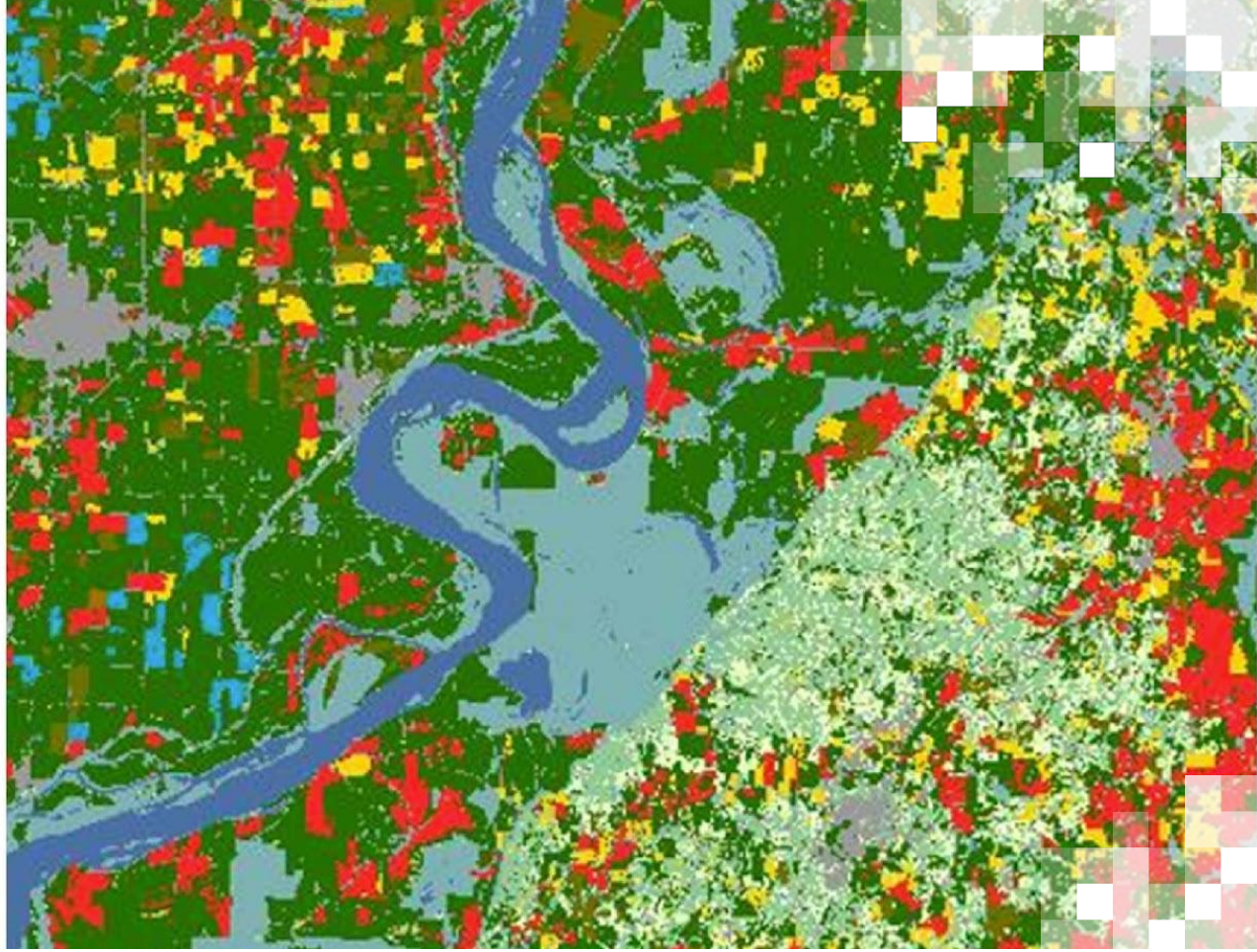
lon	lat	# scenes	bands	tiles	img dates	scl_vals	bbox	year	CDL
-89.2973	36.85473	38	AUECmAOEBM	MTZTQkZfMCw	Re5GAkYgRjRG	BQUKBQUFCgoF	549309,	2019	Corn
-89.3087	36.95602	71	AEYBMQHnAq4	MTZTQkZfMCw	Re5F7kYCRgJGI	BQUFBQUFBQUF	549309,	2019	Corn
-89.3206	36.88018	36	ATACXAOZBCw	MTZTQkZfMCw	Re5GAkYgRjRG	BQUKBQUFCgUF	549309,	2019	Corn
-89.3322	36.86472	33	ARcB4QKxA4M	MTZTQkZfMCw	Re5GIEY0RjIGP	BQoFBQUKBQUF	549309,	2019	Corn
-89.3382	36.79661	37	AQQBugJiA1ID	MTZTQkZfMCw	Re5GAkYgRjRG	AgIKBQUFCgoFB	549309,	2019	Corn
-89.3394	36.84096	37	ACgBcwHOArk	MTZTQkZfMCw	Re5GIEY0RjIGP	BQoFBQUKCgUF	549309,	2019	Corn
-89.3493	36.9019	37	AO4B5gJSAvgD	MTZTQkZfMCw	Re5GAkYgRjRG	BQUKBQUFCgUF	549309,	2019	Corn
-89.3552	36.95054	72	Ak8DYAQ4Bbg	MTZTQkZfMCw	Re5F7kYCRgJGI	BQUFBQUFBQUF	549309,	2019	Corn
-89.3577	36.74937	37	ANIAmQFTAfo	MTZTQkZfMCw	Re5GAkYgRjRG	BQUKBQUFCgoF	549309,	2019	Corn
-89.3632	36.97514	70	ACsBSwIVAIUC	MTZTQkZfMCw	Re5F7kYCRgJGI	BQUFBQUFBQUF	549309,	2019	Corn

tf.io.decode_raw(byte_string, tf.int32)

Datos Decodificados Multidimensionales

decoded band vals	decoded tiles	decoded img dates	decoded scl_vals
321,664,900,1228,1415	16SBF_0,16SBF_0,16SE	2019-01-06,2019-01-26,201	5,5,10,5,5,5,10,10,
70,305,487,686,843,98	16SBF_0,16SBG_0,16S	2019-01-06,2019-01-06,201	5,5,5,5,5,5,5,5,5,5,
304,604,921,1068,147	16SBF_0,16SBF_0,16SE	2019-01-06,2019-01-26,201	5,5,10,5,5,5,10,5,5
279,481,689,899,1075,	16SBF_0,16SBF_0,16SE	2019-01-06,2019-02-25,201	5,10,5,5,5,10,5,5,5
260,442,610,850,999,1	16SBF_0,16SBF_0,16SE	2019-01-06,2019-01-26,201	2,2,10,5,5,5,10,10,
40,371,462,697,744,81	16SBF_0,16SBF_0,16SE	2019-01-06,2019-02-25,201	5,10,5,5,5,10,10,5,
238,486,594,760,872,9	16SBF_0,16SBF_0,16SE	2019-01-06,2019-01-26,201	5,5,10,5,5,5,10,5,5
591,864,1080,1464,174	16SBF_0,16SBG_0,16S	2019-01-06,2019-01-06,201	5,5,5,5,5,5,5,5,5,5,
210,153,339,506,738,9	16SBF_0,16SBF_0,16SE	2019-01-06,2019-01-26,201	5,5,10,5,5,5,10,10,
43,331,533,597,755,11	16SBF_0,16SBG_0,16S	2019-01-06,2019-01-06,201	5,5,5,5,5,5,5,5,5,5,





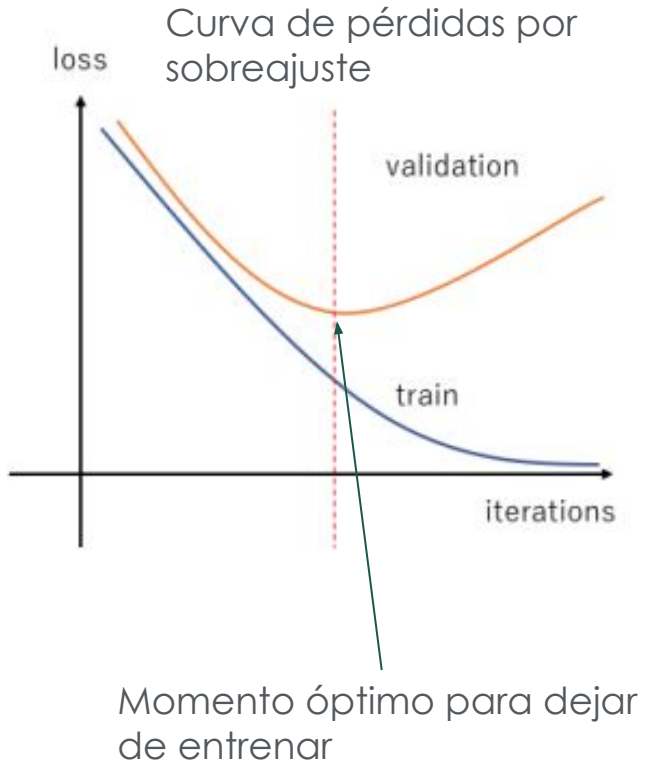
2da Parte, Sección 3:

Preparación del Conjunto de Datos para el Entrenamiento

Train/Val/Test* con Datos de Series Temporales

Antes de entrenar un modelo, necesitamos reservar algunos datos para usarlos para la validación y pruebas (no se usan para el entrenamiento del modelo).

- El conjunto de datos de Validación se usa principalmente para
  - evitar sobreajustes
  - afinar hiperparámetros
- El conjunto de datos de prueba se utiliza principalmente para evaluar cómo el modelo podría funcionar durante la producción/situaciones de la vida real (prueba)
- La siguiente es una distribución común del conjunto de datos:
    - Entrenamiento: 80%
    - Validación: 10%
    - Prueba: 10%

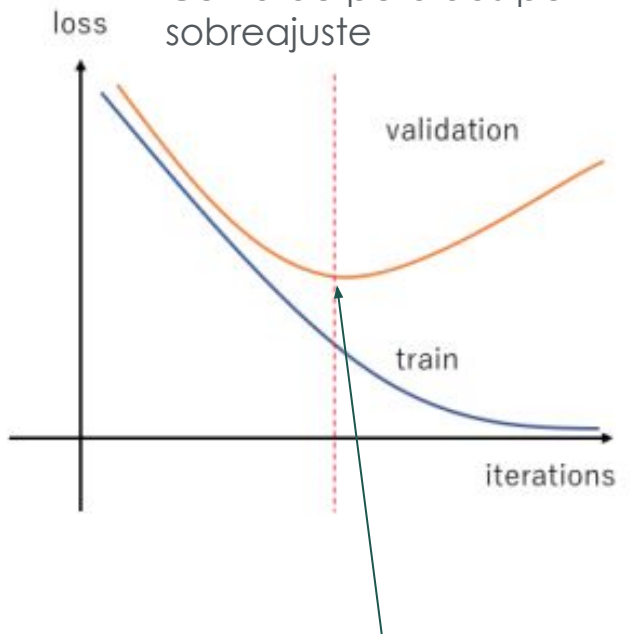


*entrenamiento/validación/prueba

Train/Val/Test* para la Predicción de Tipos de Cultivo

- Debemos tener cuidado para no "filtrar" ninguna información a través del tiempo/espacio del conjunto de entrenamiento a los conjuntos de validación o prueba, de lo contrario, el rendimiento de nuestro modelo puede parecer mejor de lo que realmente puede funcionar.
- **Para nuestra tarea de predecir las etiquetas de la capa CDL mediante imágenes, realizamos las divisiones en train/val/test utilizando el año en que se tomaron las imágenes. Esto garantiza que no se comparta información entre las divisiones de datos**.
  - **2021 entrenamiento, 2020 validación, 2019 prueba**

Curva de pérdidas por sobreajuste



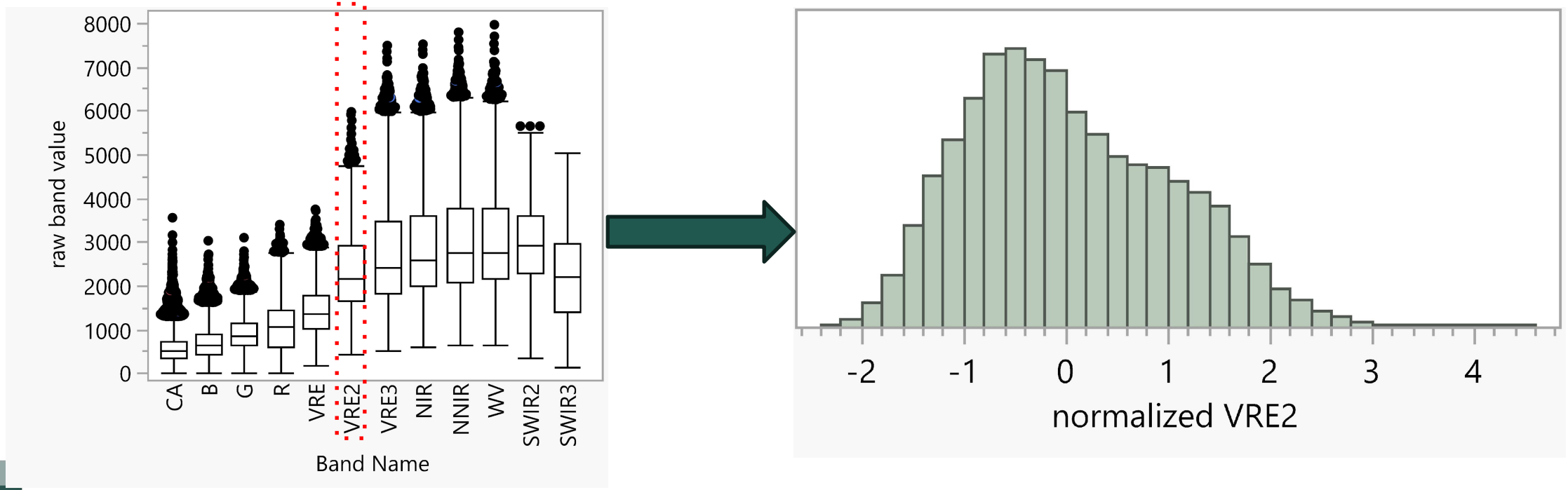
Momento óptimo para dejar de entrenar

*entrenamiento/validación/prueba

Normalización

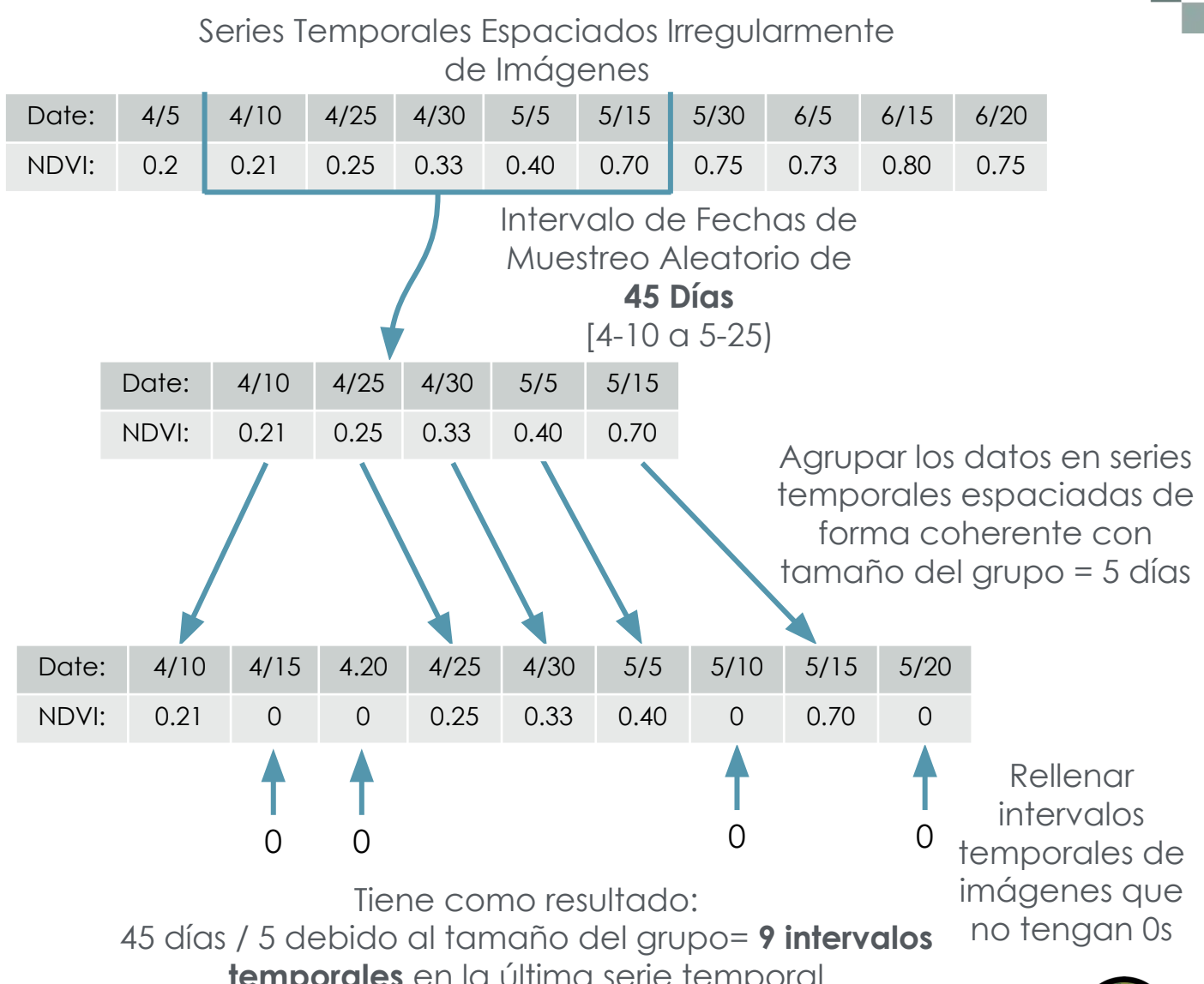
- Los insumos de nuestro modelo son las 12 bandas de Sentinel-2, cada una de las cuales tiene un rango de valores diferente. Esto no es propicio para los procedimientos de optimización típicos. La normalización es el proceso de hacer que cada característica tenga una media de 0 y un estándar de 1
  - Este paso es típico de los procedimientos de optimización en el modelado estadístico o pueden surgir problemas de convergencia.

    X_normalized = (X – mean_of_features) / std_of_features



Agrupación de Datos de Series Temporales Espaciados Irregularmente

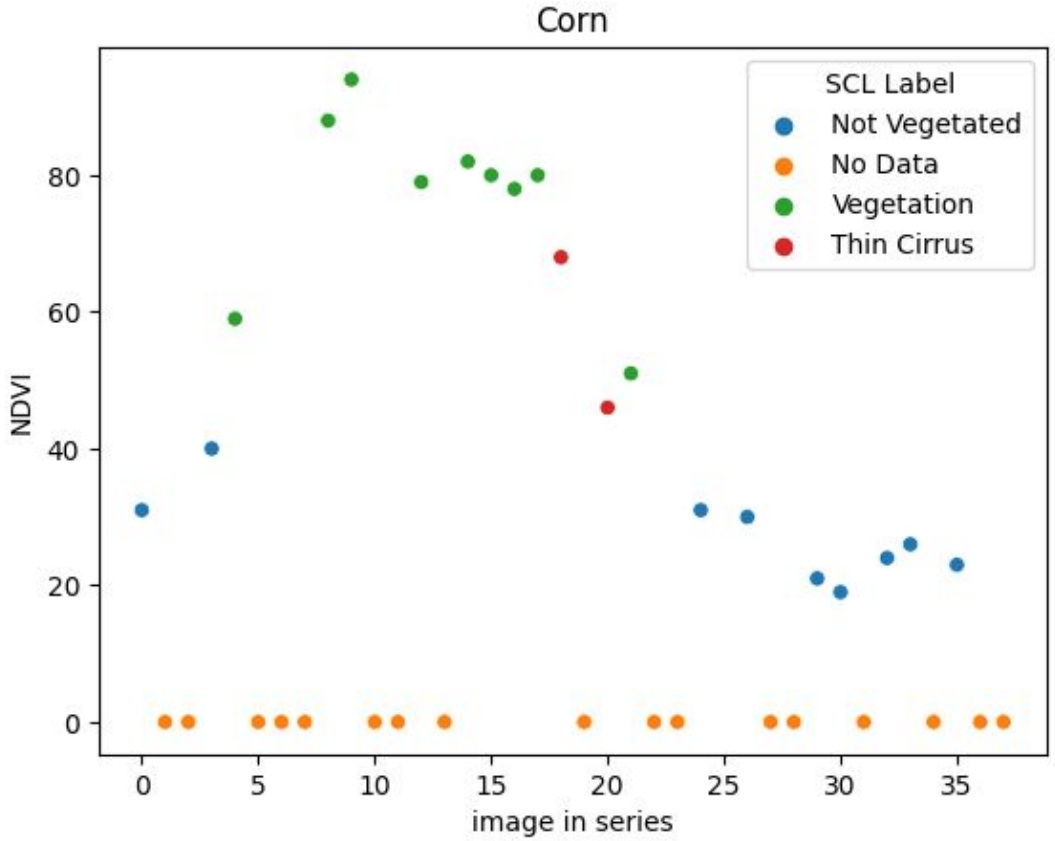
- La mayoría de las arquitecturas de modelos requieren un tamaño de entrada estándar para el entrenamiento de modelos, por lo que tener series temporales con un número diferente de imágenes no funcionará.
  - p.ej., 1D-CNN (Convolutional Neural Network)
- Necesitamos estandarizar la longitud de las series temporales de entrada.
  Para ello, tomamos imágenes dentro de un intervalo de **fechas muestreado aleatoriamente, agrupamos** las series temporales con un tamaño constante para cada cubo (por ejemplo, 5 días) y **rellenamos** los cubos sin imágenes con valores de 0.
  - El muestreo aleatorio también agiliza el rendimiento de la generalización del modelo.
- Esto garantiza que cada serie temporal tenga la misma longitud para el entrenamiento del modelo.
  El cargador de datos final utiliza un intervalo de tiempo de 120 días.



Resultados de Agrupar los Datos (Discretización en Dimensión de Tiempo)

Ejemplo:

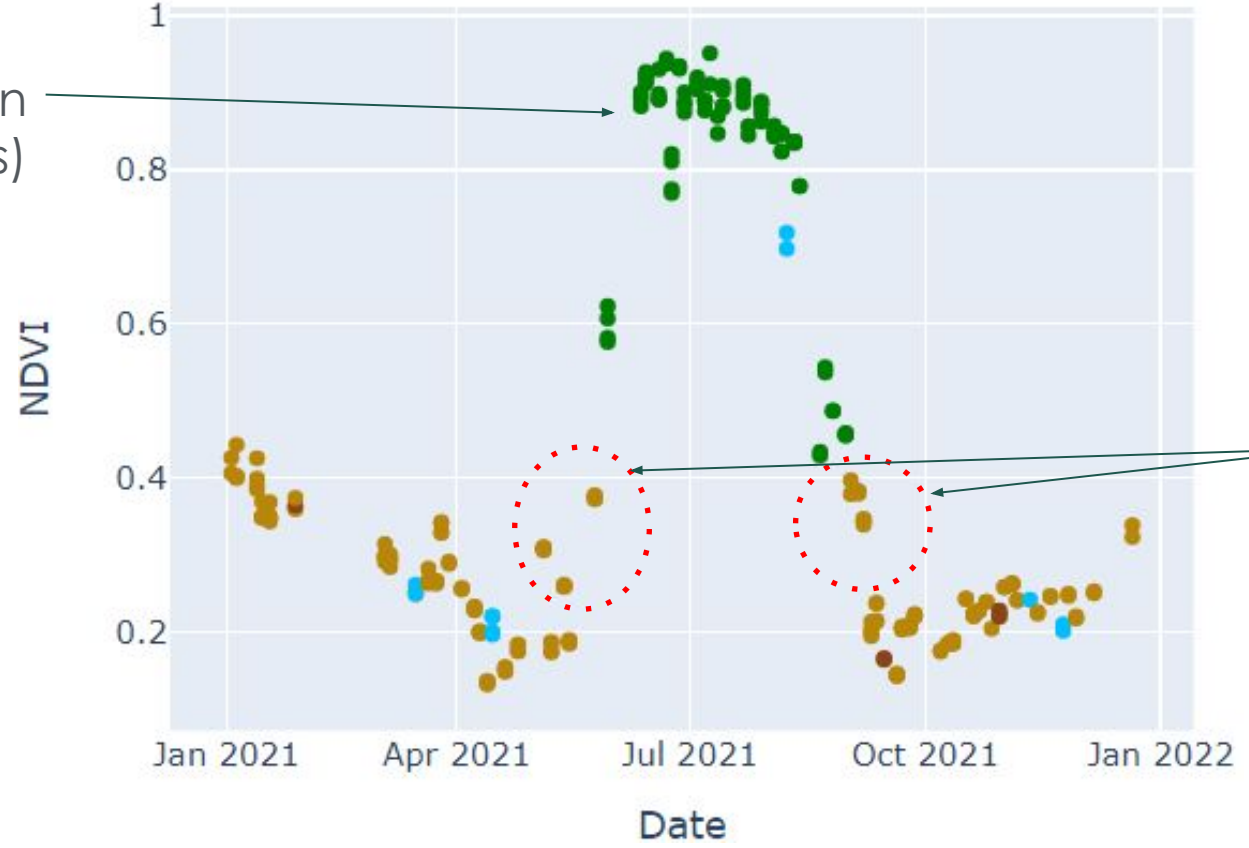
- 38 imágenes en total
  Plazo/lapso de 180 días
  Tamaño de "cubo" de intervalo de tiempo discreto de 5 días
  A los cubos/intervalos sin imagen se les asigna un valor predeterminado de "0".
  - Similar a sustituir un valor "medio" para cada banda cuando todos los datos están normalizados.



Filtrado de Capas de Clasificación de Escenas

De todas las categorías, la capa de clasificación de escenas (SCL, por sus siglas en inglés) es la más fiable para identificar cuándo algo está cubierto de vegetación (es decir, tiene una alta precisión). Por lo tanto, podemos usar esta salida particular de la SCL para señalar cuándo es un momento razonable para predecir el tipo de cultivo (por ejemplo, al menos una imagen de las 2 últimas (aprox. 10 días) en la serie temporal debe contener vegetación).

Vegetación (puntos verdes)



Estas áreas durante el crecimiento inicial del cultivo y la senescencia tardía tienen vegetación, pero no son detectadas por el clasificador SCL como vegetación. Por lo tanto, necesitamos requerir vegetación dentro de un cierto período de tiempo de la última imagen para hacer una predicción.

Modificación de las Variables Objetivo para Alinearlas con los Objetivos de Entrenamiento

- Limitar el espacio objetivo para alinearlo con nuestros objetivos para reducir la complejidad del problema..
  - Por ejemplo, concéntranos en los marcos de tiempo con vegetación.
- En este ejemplo, queremos identificar
  - 4 tipos principales de cultivos: **maíz, soya, algodón y arroz**
    1 etiqueta para otros cultivos (cultivos generales)
    1 etiqueta para áreas no cultivadas (por ejemplo, áreas urbanas)
    0 para cualquier área sin cultivos en crecimiento que normalmente se cultiva
- Reducir el espacio de destino de la CDL de más de 100 etiquetas a 6 etiquetas.
  - Esto se puede personalizar en función de tus propios objetivos de entrenamiento.

Resumen de Grupos de Variables Objetivo

Primary CDL Labels
Corn
Soybeans
Cotton
Rice

"Other Crops" CDL Labels	
Other Hay/Non-Alfalfa	Winter Wheat
Pop or Orn Corn	Alfalfa
Peanuts	Potatoes
Sorghum	Peas
Oats	Herbs
Peaches	Rye
Clover/Wildflowers	Cantaloupes
Pecans	Sunflower
Sod/Grass Seed	Watermelons
Other Crops	Sweet Corn
Dry Beans	Sweet Potatoes

Uncultivated CDL Labels
All other labels

Final Training Label List
Corn
Soybeans
Cotton
Rice
Cultivated
Uncultivated
No Crops Growing

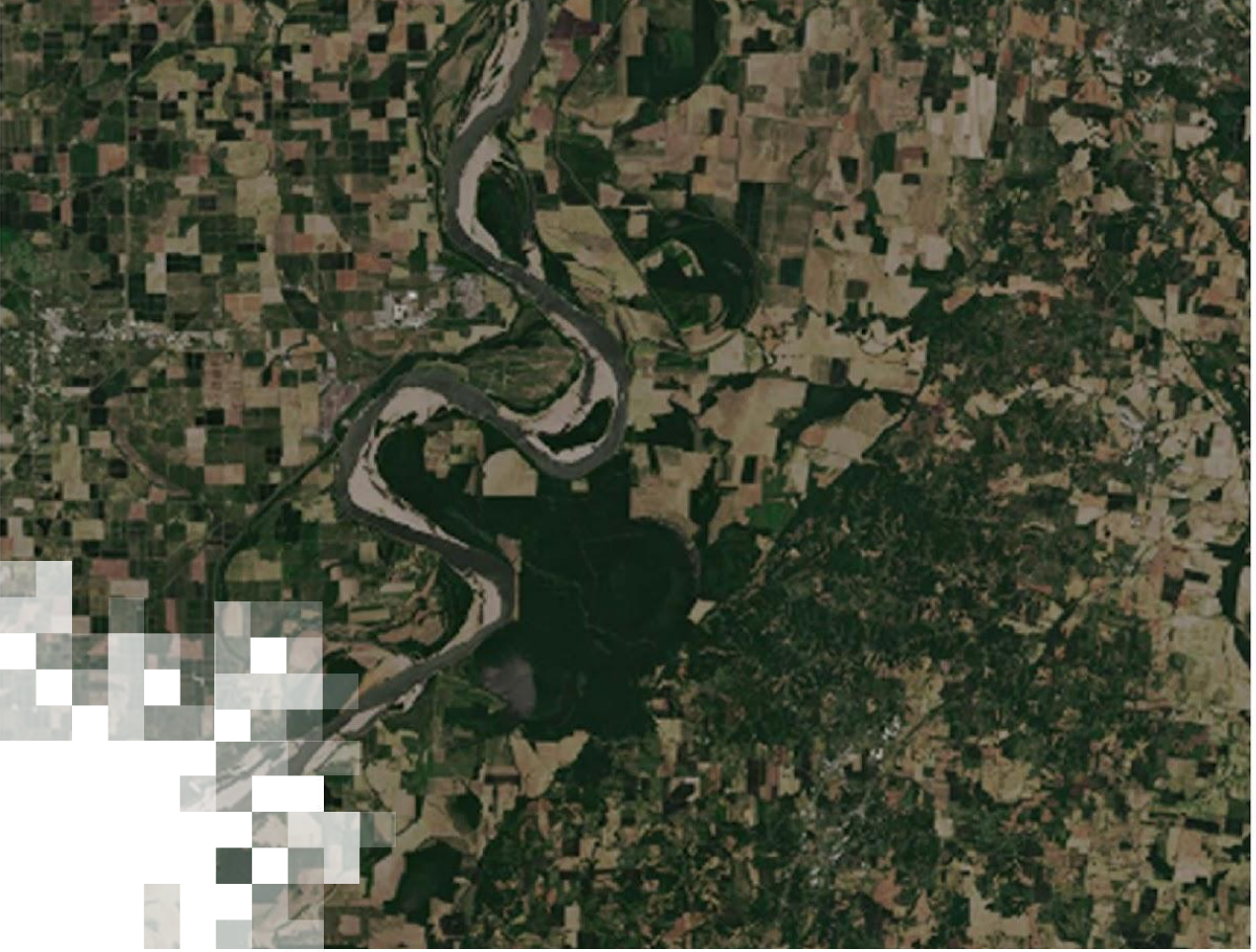
Etiquetas de Codificación One-hot

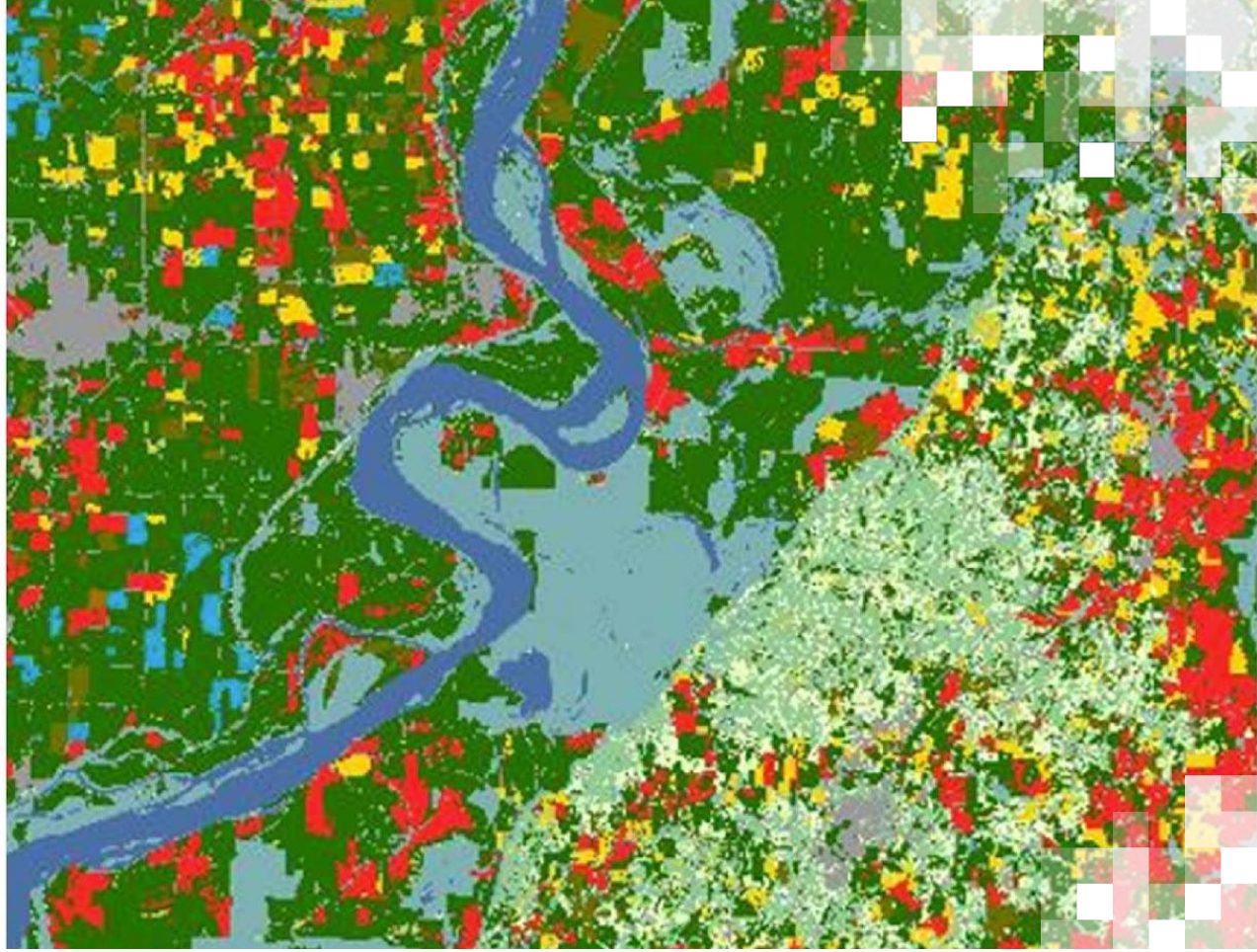
- La codificación one-hot-encoding es un método para transformar el espacio de etiquetas en valores de 0 y 1. El índice en el que el valor 1 indica el tipo de etiqueta (por ejemplo, un 1 en el primer índice indica Cultivado) Por lo tanto, la longitud de la etiqueta codificada one-hot es el número de etiquetas que se van a clasificar. En nuestro caso, la longitud es de 6.
  Esto es necesario si se utiliza una función de activación Softmax con pérdida de entropía cruzada categórica.
  - Entraremos en los detalles de estos en la 3rd Parte.

Nuestras Etiquetas
Uncultivated
Cultivated
No Crops Growing
Corn
Soybeans
Cotton
Rice

One-hot-encoded Labels

No Crops Growing	Uncultivated	Cultivated	Corn	Soybeans	Cotton	Rice
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	0	1
0	0	0	0	0	1	0
0	1	0	0	0	0	0
0	0	1	0	0	0	0





2da Parte:
Resumen

Resumen

- Se creó una cola personalizada y altamente eficiente para cargar datos a través de conjuntos de datos de TensorFlow utilizando los archivos Parquet de la 1ra Parte como insumos.
  Mapeamos, mezclamos aleatoriamente, por lotes y captura previa para optimizar el rendimiento del conjunto de datos de TensorFlow con paralelización.
  - Usamos módulos como **tf.io.decode_raw()** para convertir las **byte strings** (cadenas de bytes) de vuelta en datos útiles.
- Analizamos algunos pasos de preprocesamiento:
  - Distribuimos los datos en divisiones train/val/test correctamente para evitar la "**fuga de datos**" y realizamos un seguimiento si es que el modelo está sobreajustado.
  - **Normalizamos** los datos de insumo para mejorar la estabilidad del entrenamiento del modelo.
  - **Agrupamos** las imágenes de series temporales espaciadas irregularmente para prepararlas para el entrenamiento del modelo.
    Modificamos la variable objetivo para alinearla con **los objetivos de entrenamiento**.

Mirando Hacia la 3ra Parte

- Entrenamiento de un 1D-CNN (One-Dimensional – Convolutional Neural Network*) para predecir el tipo de cultivo a partir de imágenes de Sentinel-2.
- Supervisión del rendimiento del modelo en Databricks mediante Tensorboard. Prueba y visualización de los resultados del modelo.

*Red neuronal convolucional unidimensional

Tarea y Certificados

- **Tarea:**
  - Habrá una tarea asignada
  - Abre el 19 de marzo de 2024
  - Acceso desde la página web de la capacitación
  - Debe enviar sus respuestas vía Formularios de Google
  - **Fecha límite: 1ro de abril de 2024**

- **Certificado de Finalización de Curso:**
  - Asista a las tres sesiones en vivo (la asistencia se registra automáticamente)
  - Complete la tarea dentro del plazo estipulado
  - Recibirá un certificado por correo electrónico aproximadamente dos meses después de la conclusión del curso.

Datos de Contacto

Formadores:

- John Just (John Deere)
  - JustJohnP@JohnDeere.com
- Erik Sorensen
  - SorensenErik@JohnDeere.com
- Sean McCartney
  - Sean.McCartney@nasa.gov

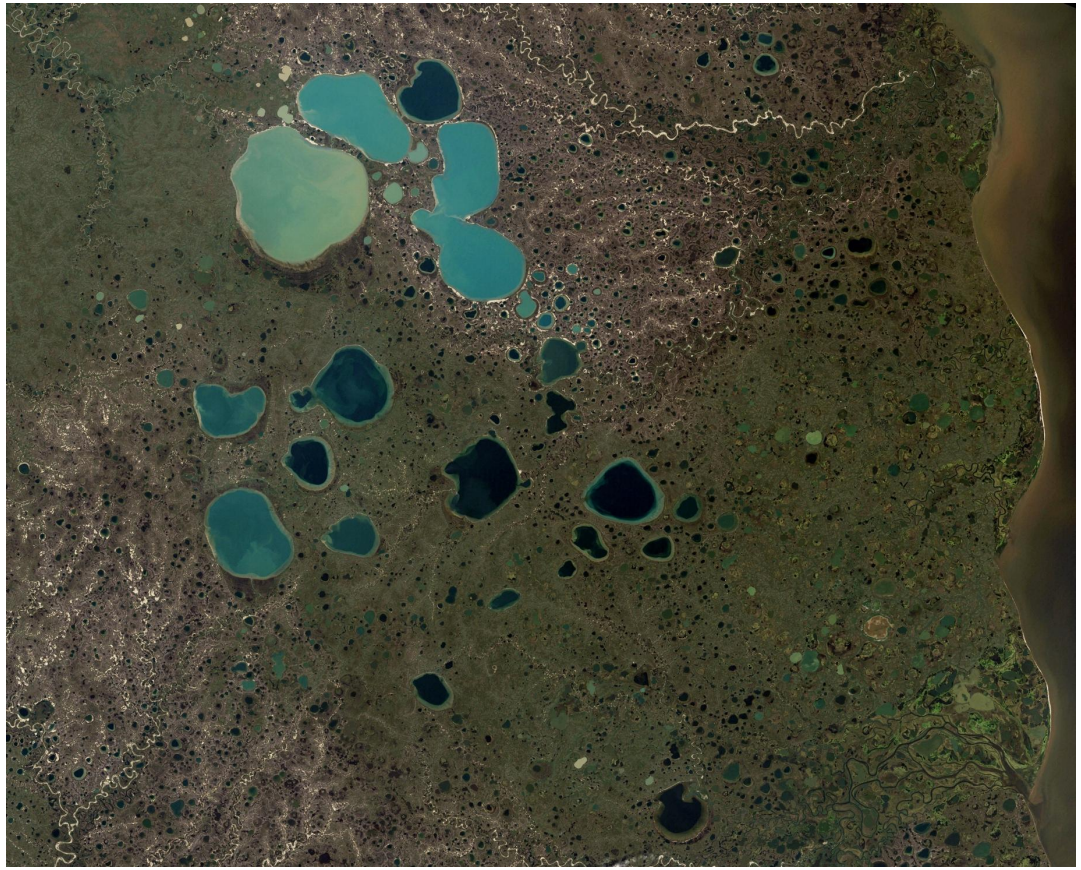
- Página web de ARSET
- ¡Síganos en X (antiguamente Twitter)!
  - @NASAARSET
- ARSET YouTube

Visite nuestros Programas Hermanos:

- DEVELOP
- SERVIR

¿Preguntas?

- Por favor escriba sus preguntas en la casilla denominada “Questions”. Las responderemos en el orden en que fueron recibidas.
- Publicaremos las preguntas y respuestas a la página web de la capacitación después de la conclusión del webinar.



https://earthobservatory.nasa.gov/images/6034/pothole-lakes-in-siberia


APPLIED REMOTE SENSING TRAINING PROGRAM
TRAIN - EMPOWER - ADVANCE

¡Gracias!