Machine Learning Data Lifecycle in Production

MLOps Specialization Course 2

In the second course of Machine Learning Engineering for Production Specialization, you will build data pipelines by gathering, cleaning, and validating datasets and assessing data quality; implement feature engineering, transformation, and selection with TensorFlow Extended and get the most predictive power out of your data; and establish the data lifecycle by leveraging data lineage and provenance metadata tools and follow data evolution with enterprise data schemas.

Understanding machine learning and deep learning concepts is essential, but if you're looking to build an effective AI career, you need production engineering capabilities as well. Machine learning engineering for production combines the foundational concepts of machine learning with the functional expertise of modern software development and engineering roles to help you develop production-ready skills.

Week 4: Advanced Labeling, Augmentation and Data Preprocessing

Combine labeled and unlabeled data to improve ML model accuracy and augment data to diversify your training set.

- Discuss direct, semi-supervised, weak supervision and active learning methods for labeling data
- Increase the diversity of your training set by data augmentation
- Perform advanced data preparation and transformation on different structured and unstructured data types

Compiled By	**Peter Boshra**
LinkedIn	**https://www.linkedin.com/in/peterboshra/**
GitHub	**https://github.com/PeterBushra**
Email	**Dev.PeterBoshra@gmail.com**

Outline

- Overview of advanced labeling techniques:
  - Semi-supervised learning
  - Active learning
  - Weak supervision with Snorkel

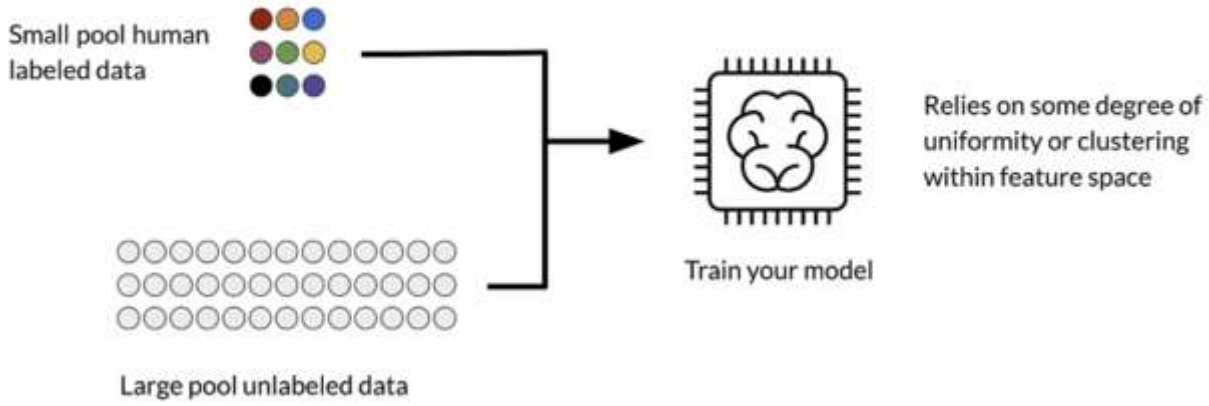
Why is Advanced Labeling Important?

ML use is growing everywhere

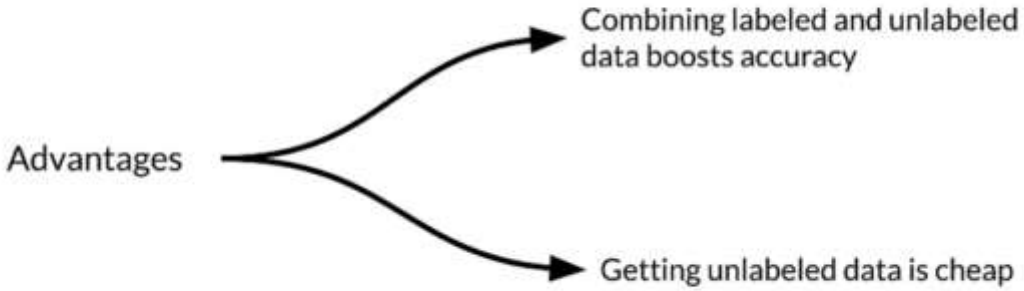
...as is the need for labeled training sets

- Manually labeling of data is expensive
- Unlabeled data is usually cheap and easy to get
- Unlabeled data contains a lot of information that can improve our model

Human labeling, Semi-supervised



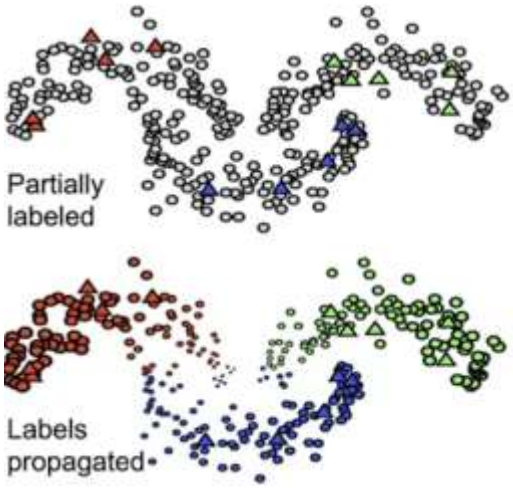
Human labeling, Semi-supervised



Label propagation

- Semi-supervised ML algorithm
- A subset of the examples have labels
- Labels are propagated to the unlabeled points:
  - Based on similarity or "community structure"

Label propagation - Graph based



Unlabeled examples can be assigned labels based on their neighbors

Active learning

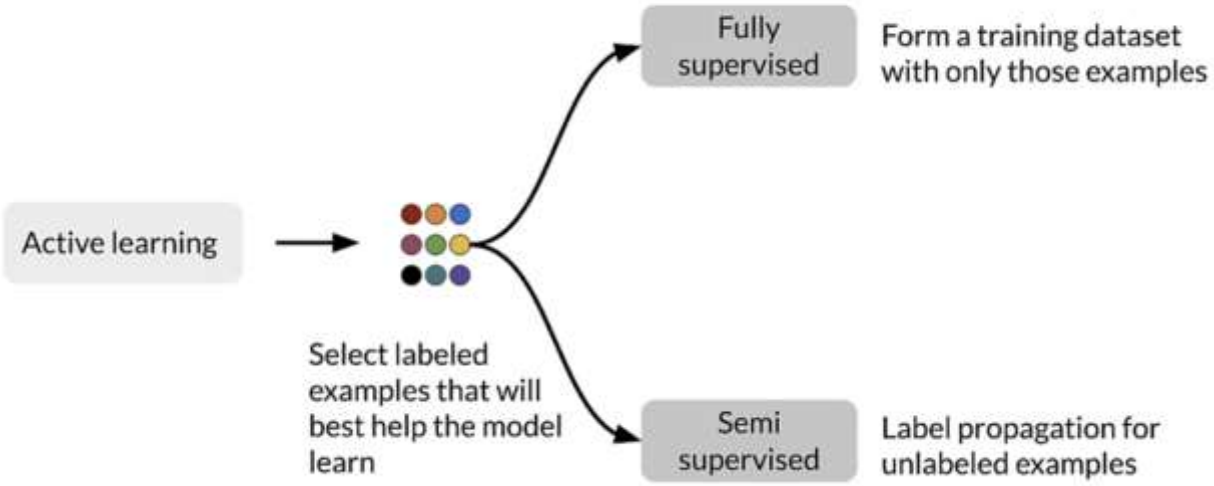
- A family of algorithms for intelligently sampling data
- Select the points to be labeled that would be most informative for model training
- Very helpful in the following situations:


Constrained data budgets: you can only afford labeling a few points

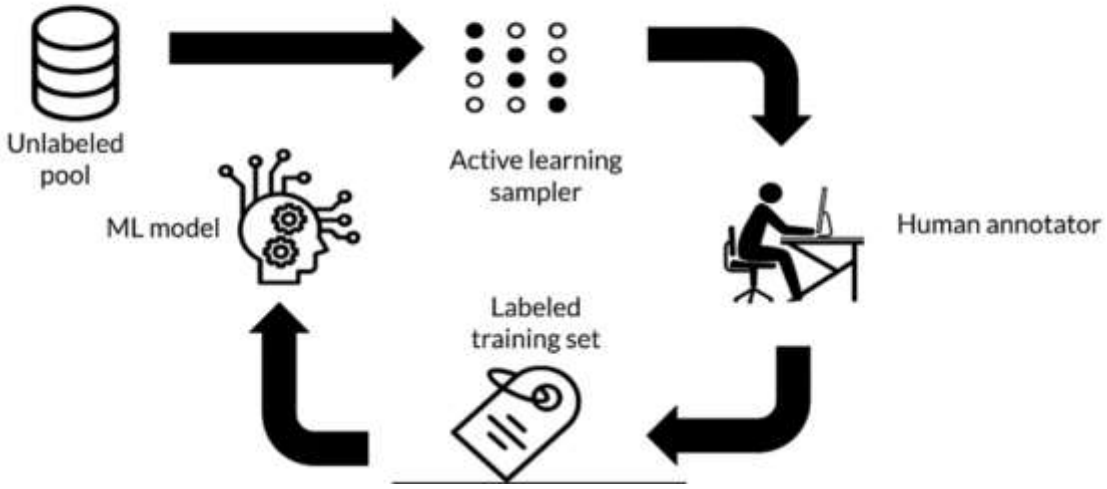

Imbalanced dataset: helps selecting rare classes for training


Target metrics: when baseline sampling strategy does not improve selected metrics

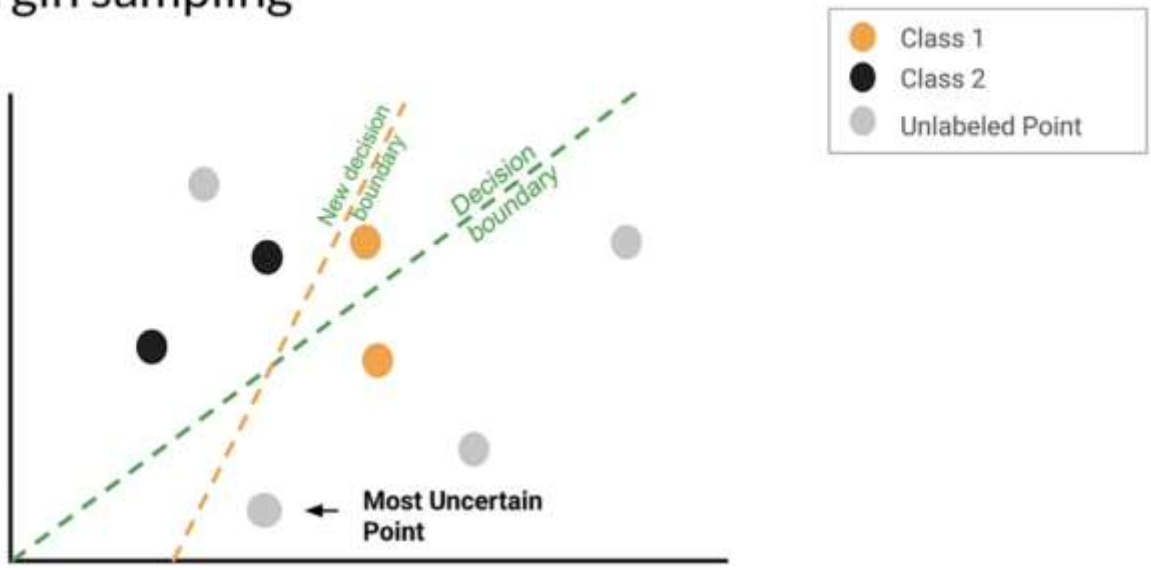
Active learning strategies



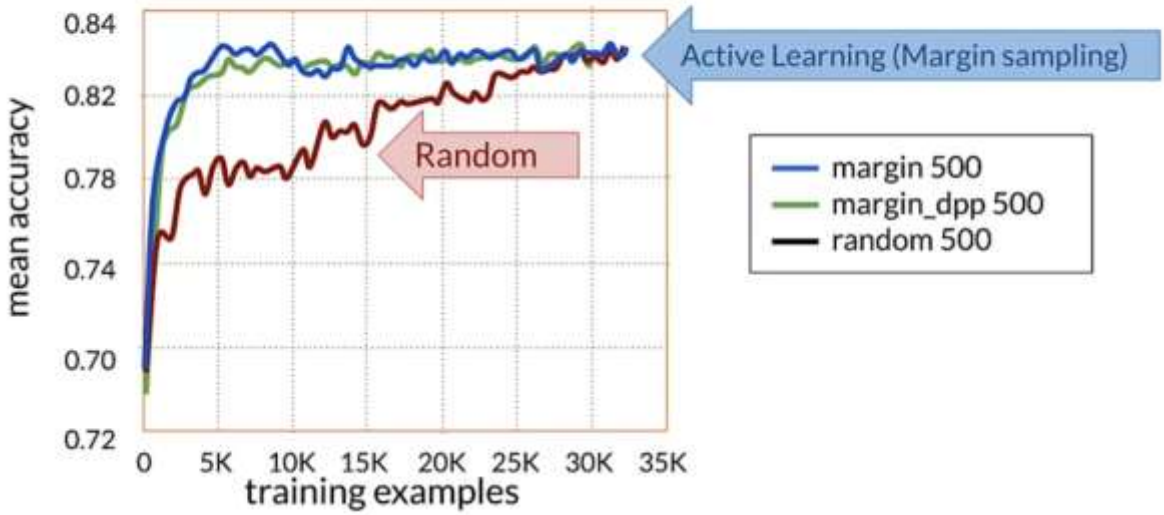
Active learning cycle



Margin sampling



Example results - Different Sampling Techniques



Active learning sampling techniques

Margin sampling: Label points the current model is least confident in.

Cluster-based sampling: sample from well-formed clusters to "cover" the entire space.

Query-by-committee: train an ensemble of models and sample points that generate disagreement.

Region-based sampling: Runs several active learning algorithms in different partitions of the space.

Hand labeling: intensive labor

"Hand-labeling training data for machine learning problems is effective, but very labor and time intensive. This work explores how to use algorithmic labeling systems relying on other sources of knowledge that can provide many more labels but which are noisy."

Jeff Dean, March 14, 2019

Weak supervision

"Weak supervision is about leveraging higher-level and/or noisier input from subject matter experts (SMEs)."

-Weak Supervision: The New Programming Paradigm for Machine Learning
Blog post by Ratner, Varma, Hancock, Re, and Hazy Lab

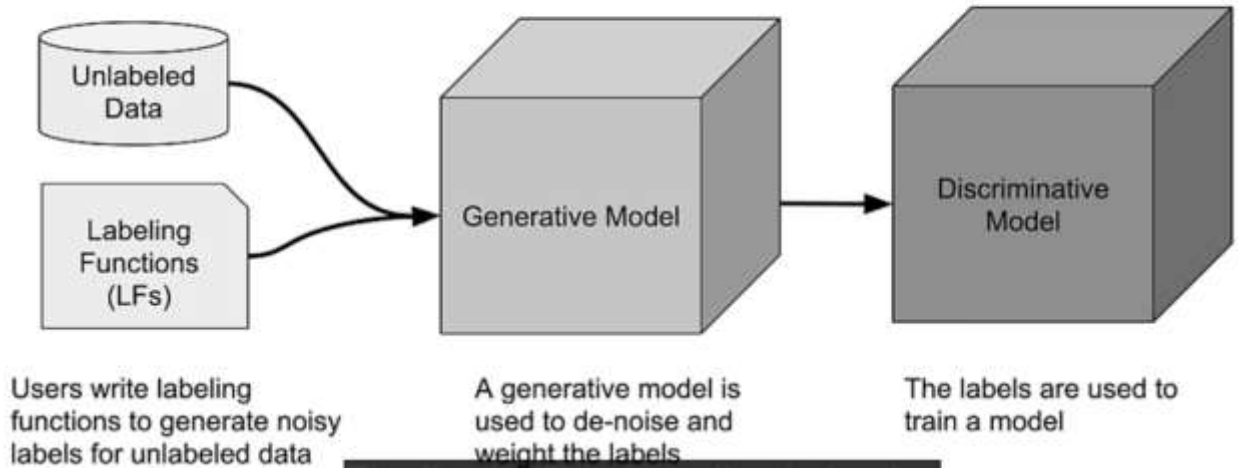
Weak supervision

- Unlabeled data, without ground-truth labels
- One or more weak supervision sources
  - A list of heuristics that can automate labeling
  - Typically provided by subject matter experts
- Noisy labels have a certain probability of being correct, not 100%
- Objective: learn a generative model to determine weights for weak supervision sources

Snorkel

- Project started at Stanford in 2016
- Programmatically building and managing training datasets without manual labeling
- Automatically: models, cleans, and integrates the resulting training data
- Applies novel, theoretically-grounded techniques
- Also offers data augmentation and slicing

Data programming pipeline in Snorkel



Snorkel labeling functions

```
from snorkel.labeling import labeling_function

@labeling_function()
def lf_keyword_my(x):
    """Many spam comments talk about 'my channel', 'my video', etc."""
    return SPAM if "my" in x.text.lower() else ABSTAIN

@labeling_function()
def lf_short_comment(x):
    """Non-spam comments are often short, such as 'cool video!'."""
    return NOT_SPAM if len(x.text.split()) < 5 else ABSTAIN
```

Key points

- Semi-supervised learning:
  - Applies the best of supervised and unsupervised approaches
  - Using a small amount of labeled data boosts model accuracy
- Active learning:
  - Selects the most important examples to label
  - Improves predictive accuracy
- Weak supervision:
  - Uses heuristics to apply noisy labels to unlabeled examples
  - Snorkel is handy framework for weak supervision

Outline

- Generating synthetic data
- Augmenting an image dataset: CIFAR-10 example
- Other advanced techniques

How do you get more data?

- Augmentation as a way to expand datasets
- One way is introducing minor alterations
- For images: flips, rotations, etc.

How does augmentation data help?

- Adds examples that are similar to real examples
- Improves coverage of feature space
- Beware of invalid augmentations!

An example: CIFAR-10 data set

- 60,000 32x32 color images
- 10 classes of objects
  (6,000 images per class)

Data augmentation on CIFAR-10

Let's augment the CIFAR-10 dataset with the following steps:

1. Pad the image with a black, four-pixel border
2. Randomly crop a 32 x 32 region from the padded image
3. Flip a coin to determine if the image should be flipped horizontally left/right

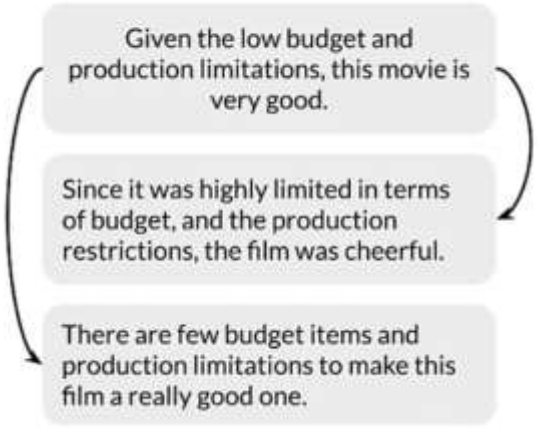
Defining the augment operation

```
def augment(x, height, width, num_channels):
    x = tf.image.resize_with_crop_or_pad(x, height + 8, width + 8)
    x = tf.image.random_crop(x, [height, width, num_channels])
    x = tf.image.random_flip_left_right(x)
    return x
```

Augmented examples

Other Advanced techniques

- Semi-supervised data augmentation e.g., UDA, semi-supervised learning with GANs
- Policy-based data augmentation e.g., AutoAugment



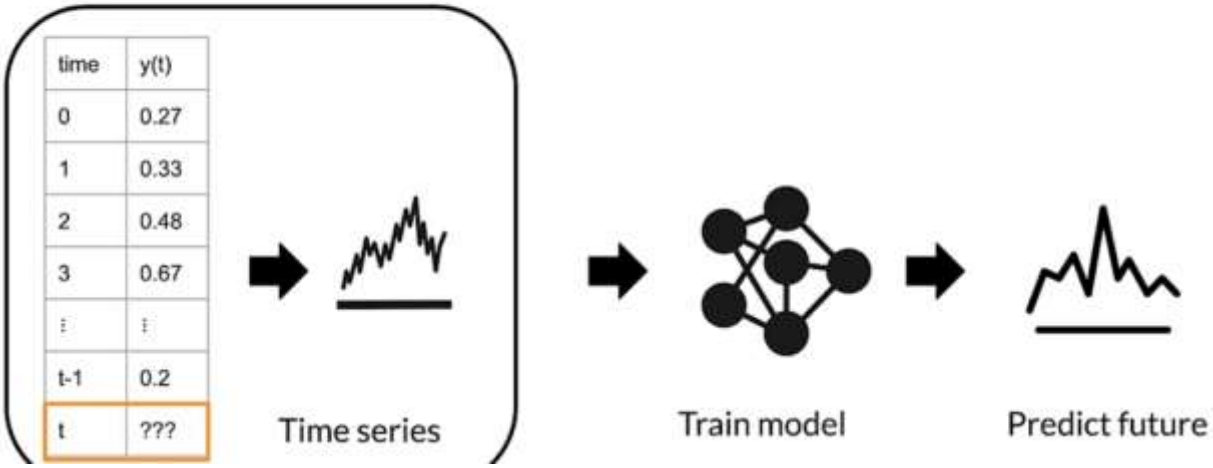
Key points on data augmentation

- It generates artificial data by creating new examples which are variants of the original data
- It increases the diversity and number of examples in the training data
- Provides means to improves accuracy, generalization, and avoiding overfitting

A note on different types of data

- TFX pre-processing capabilities for multiple data types:
  - Images
  - Video
  - Text
  - Audio
  - Time series
- Optional notebook on images
- Two optional notebooks on time series

Time series data



"It is difficult to make predictions, especially about the future."

- Karl Kristian Steincke

Time series forecasting

- Time Series forecasting predicts future events by analyzing data from the past
- Time series forecasting makes predictions on data indexed by time
- Example:
    - Predict future temperature at a given location based on historical meteorological data

Time series dataset: Weather prediction

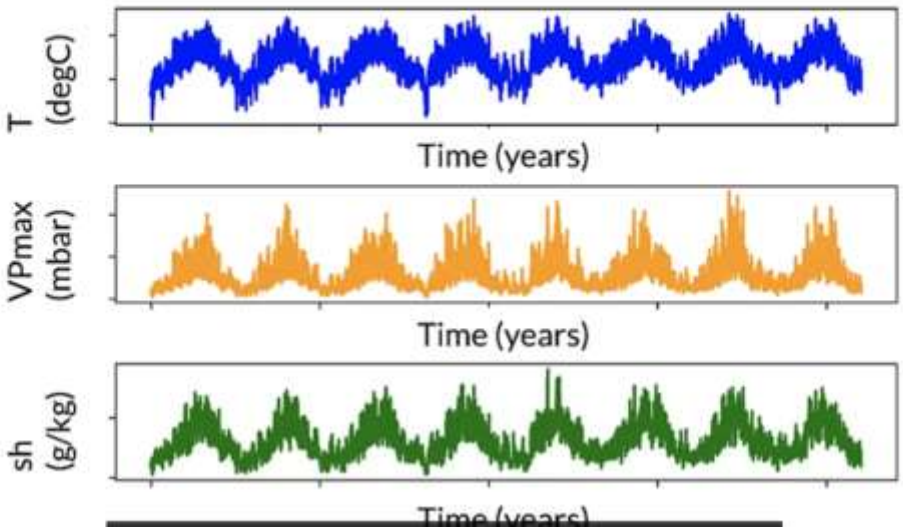
We will use the weather time series dataset recorded by the Max Planck Institute for Biogeochemistry:

- to preprocess time series data with TensorFlow Transform
- to convert data into sequences of time steps:
    - Making data ready to train a long short-term memory (LSTM) recurrent neural network (RNN)

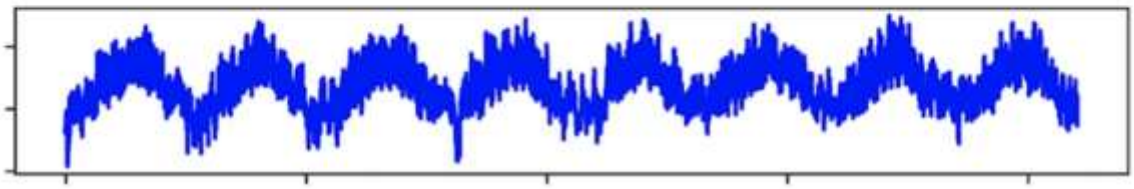
Weather time series dataset

- There are 14 variables:
    - P(mbar), T (degC), Tdew (degC), rh (%), VPmax (mbar), VPact (mbar), VPdef (mbar), sh (g/kg), H20C (mmol/mol), rho (g/m**3), wv (m/s), max.xv (m/s), wd (deg)
    - Target is T (degC)
- Observations recorded every 10 minutes
    - 6 observations per hour
    - 144 (6 X 24) observations in a day.

Data visualizations



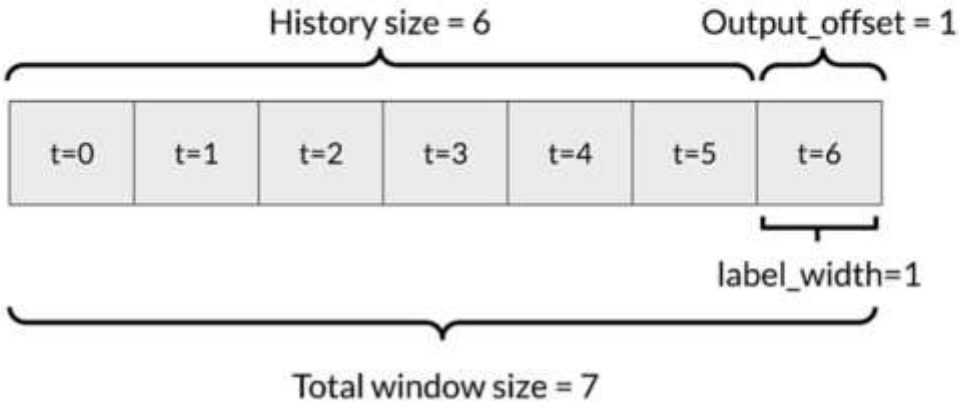
Windowing data



- Windowing makes sense.
- tf.data.Datasets.window() converts times series data to depend on past observations.

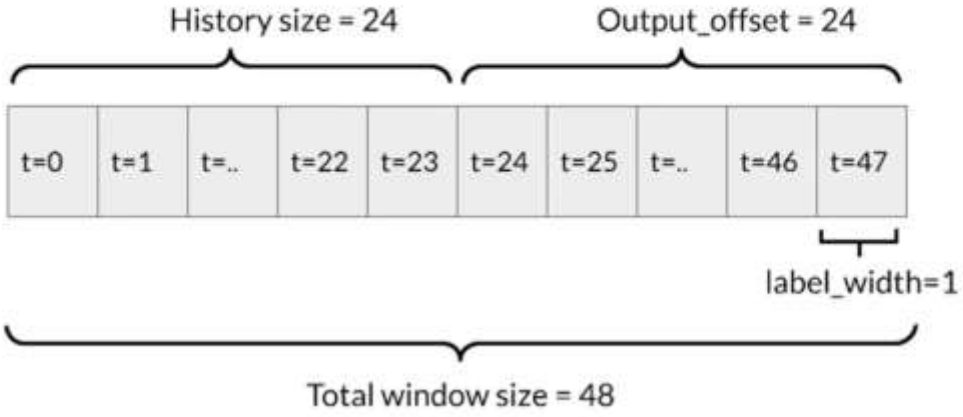
Windowing strategies in single step time series

A model that makes a prediction 1h into the future, given 6h of history would need a window like this:



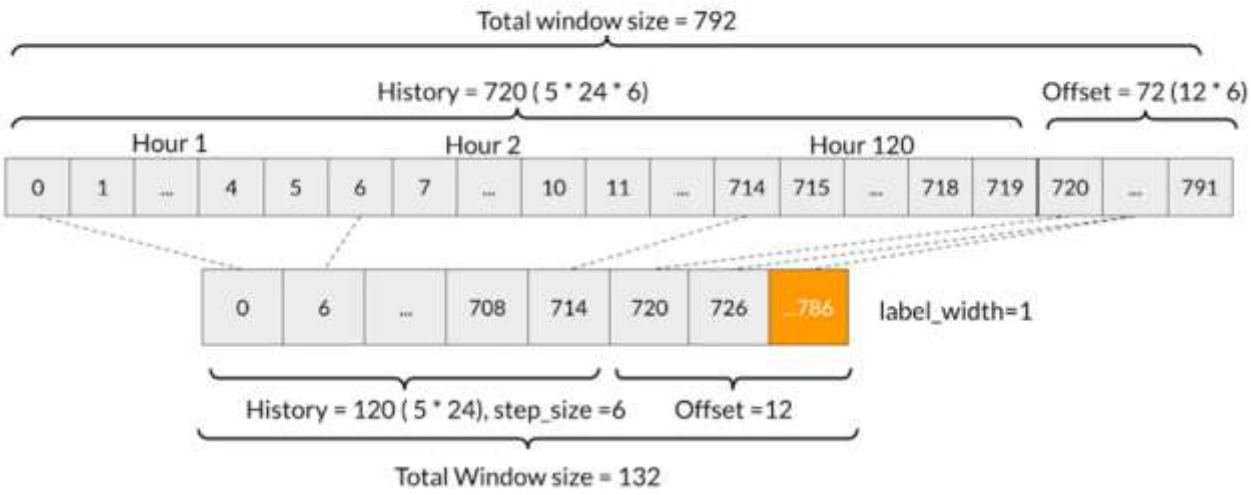
Windowing strategies in single step time series

Predict next 24 hours given 24 hours of history



Windowing strategy for the problem

Sample one observation every hour with step size = 6



Optional notebook: what will you do?

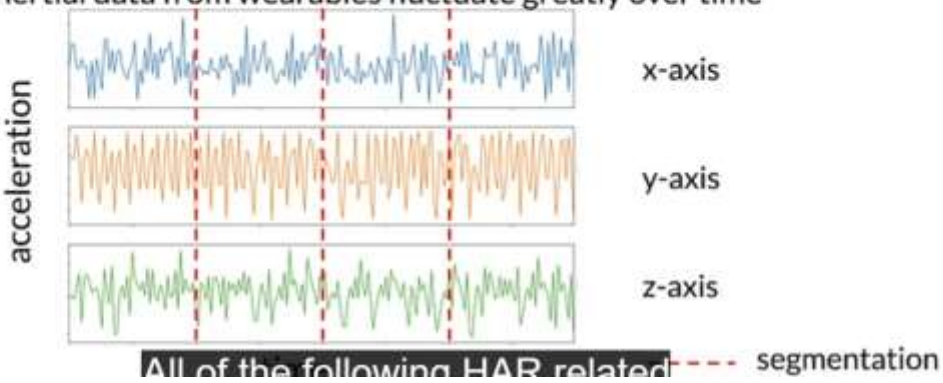
- Data processing with TFX to extract features
- Segment data into windows
- Save data in TFRecord format
- Make it ready for training an LSTM model

Sensors and Signals

- Signals are sequences of data collected from real time sensors
- Each data point is indexed by a timestamp
- Sensors and signals data is thus time series data
- Example: classify sequences of accelerometer data recorded by the sensors on smartphones to identify the associate activity

Human activity recognition (HAR)

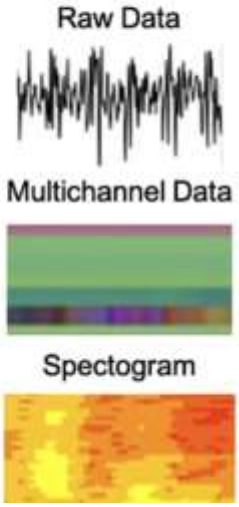
- HAR tasks require segmentation operations
  - Raw inertial data from wearables fluctuate greatly over time



All of the following HAR related

Human activity recognition (HAR)

- Segmented data should be transformed for modeling
- Different methods of transformation:
  - Spectrograms (commonly used)
  - Normalization and encoding
  - Multichannel
  - Fourier transform



Optional notebook: what will you do?

- Work with Human Activity Recognition Dataset (WISDM):
  - Preprocess with TensorFlow Transform
  - Use tf.data.Datasets.window() for converting times series data to depend on past observations