DATA 101: INTRODUCTION TO DATA AND DATA PROCESSING

SPONSORED BY PURDUE WOMEN IN HPC

SARAH RODENBECK, SENIOR RESEARCH DATA SCIENTIST
Sign up for our newsletter

SCHOLARSHIPS
Present your research at conferences

CAREER DEVELOPMENT
Attend trainings and events

COMMUNITY
Join the conversation on Slack
OUTCOMES

- Understand how data is used in machine learning and data science
- Data types and data collection
- Topics to consider when doing an Exploratory Data Analysis
- Basic data processing using Pandas
DATA BASICS
MODEL DEVELOPMENT CYCLE
WHAT IS DATA?

1 Introduction

The asynchronous form of Q-learning, which is a stochastic approximation paradigm that applies to Markovian non-i.i.d. samples, has found applicability in an abundance of reinforcement learning (RL) applications (Silver et al., 2014; Amsaleg et al., 1994; Borkar, 1992). The input data takes the form of a Markovian sample trajectory induced by a policy called the behavior policy; in each turn, asynchronous Q-learning only updates the Q-function estimate of a single state-action pair along the trajectory rather than updating all pairs at once — and hence the terminology “asynchronous” (Bertsekas and Tsitsiklis, 1996; Tsitsiklis, 1994). This classical algorithm has the virtue of being off-policy, allowing one to learn the optimal policy even when the behavior policy is suboptimal. Recent years have witnessed a resurgence of interest in understanding the performance of asynchronous Q-learning, due to a shift of attention from classical asymptotic analysis to the non-asymptotic counterpart. By and large, non-asymptotic results bear important and clear implications for the impacts of relevant parameters (e.g., model capacity, batch length) in large-dimensional RL problems.

1.1 Motivation

A central consideration in modern RL applications is data efficiency: the limited availability of data samples places increasing demands on sample-efficient RL solutions, and in turn, calls for reexamining classical algorithms like Q-learning. When it comes to asynchronous Q-learning, recent theoretical advances have led to sharped sample complexity analyses (Li et al., 2021a; Qiu and Wierman, 2020). For concreteness, consider a γ-discounted infinite-horizon Markov decision process (MDP) and a stationary behavior policy; asynchronous Q-learning provably yields ε-accuracy as soon as the sample size exceeds the order of $\varepsilon^2 (\ln(1/\varepsilon))^{-2}$.

$$\frac{1}{\ln(1 - \gamma)^{2g} + \frac{1}{2}}$$  (11)
WHY DOES THIS MATTER?

Intended model dictates data needs

Available data dictates possible models
STRUCTURED VS UNSTRUCTURED DATA

Structured Data

What you find in a DB (typically)

Unstructured Data

What you find in the 'wild' (text, images, audio, video)
LABELLED VS UNLABELLED DATA

Unlabeled Data

Je m’appelle Sarah.

Labeled Data

Je m’appelle Sarah = My name is Sarah
SUPERVISED VS UNSUPERVISED LEARNING

- Most Machine Learning and Statistical Models
  - Image Recognition
  - Neural Machine Translation
  - Loan Default Prediction
- Relies on Labelled Data as the "ground truth"

- Data preparation and limited models
  - Clustering for anomaly detection
  - Dimensionality reduction
  - Association and Recommender systems
- Does not require labelled data
SUPERVISED LEARNING

- More data preparation
- More performant
- More types of models
ON LABELED DATA

- You need a lot of labeled data for most ML systems!
  - Typically on the scale of thousands or tens of thousands of data points

- **Transfer learning** reduces data needs but you still need enough to fine tune the model for your specific task.
DATA ANNOTATION

- Sometimes labelled data is naturally collected (e.g. engineers marking if a test worked or not)
- Other times you can use pre-existing labelled data (e.g. Kaggle dataset)
- Most of the time you will need to go through a data annotation process

**Example:**

I am now part of the group of people for which gabapentin is the devil. 600mg was fine but useless. 900mg: I am now the useless one 😞
DATA ANNOTATION DIFFICULTIES

- Clear and consistent guidelines are key
  - Inconsistent labelling can confuse the model’s training
  - It can be helpful to think about whether false positives or false negatives are more acceptable for your model and advise your annotators to err on that side
- Annotation can be a tedious and error prone process but is one of the most important
- Garbage in, garbage out!

Coming off Zoloft is $%&@*#! weird
BEST PRACTICES FOR DATA ANNOTATION

- If multiple annotators:
  - Create common guidelines
  - Hold an alignment meeting to go through examples together and calibrate
  - Have everyone complete a pre-test by annotating a limited number of examples and assess inter-annotator accuracy and analyze common misconceptions
- Tools like Doccano (open-source) can make data annotation easier
DATA PRE-PROCESSING
MODEL DEVELOPMENT CYCLE
DATA CLEANING AND DATA PREPARATION ACCOUNTS FOR AS MUCH AS 80% OF A DATA SCIENTIST'S TIME
An extremely common tool used for working with data are dataframes

- Contains both data and metadata

- Dataframes are structures that organizes data into a two-dimensional data

- Doesn’t have size limitations like Excel!

*All provided commands in this presentation are for use with the Pandas DataFrames package in Python*
MISSING DATA

What you do with missing outliers depends on your use case and the columns.

If you are missing data in the label column you may be best to delete the row.

For other columns you may be able to interpolate data:
  - e.g. in a time series analysis you could average the previous and next data point.
ACCOUNT FOR OUTLIERS

- What you do with outliers depends on your use case
- Are outliers important in your data (e.g. anomaly detection) or are they representative of someone not labelling the data well?
  - With the annotation data visualized here, I might want to not use annotator 11’s data for model training
If you have a column with the subject name or identifiable information you can:

- Delete the column
- Substitute an anonymized value instead
DATA SETS

Training Data (70%)
Data used by the model for training

Validation Data (20%)
Used to assess model performance during training

Test Data (10%)
Data used after training is complete for evaluation metrics
OTHER POTENTIAL STEPS IN PRE-PROCESSING

- Ensure consistent descriptors/categories (e.g. ‘female’ and ‘woman’ should be the same)
- Rename columns
- Data encoding
  - Data must be in a specific format for training. For example, with text data you will need to get the word embeddings
- Feature engineering
  - Using domain knowledge to manipulate raw data into a format that better captures key characteristics of the data
- Binning data for easier analysis
- Feature Selection/Dimensionality reduction
  - Reduces the data used to help the model identify what’s most important
- Etc...
EXPLORATORY DATA ANALYSIS
EXPLORATORY DATA ANALYSIS

- Understand and summarize the main characteristics of a data set prior to model training
WHY PERFORM AN EDA?

- Better understand data and data patterns
- Can uncover data issues
- Help select the right model for your data
COMMON ASPECTS OF AN EDA

- Average, median, high and low values for each column
- Relationships between columns (correlations)
- Explore data quality trends
- Identify unnecessary columns
- Null and outlier analysis
- Visualize data
- Identify data biases
VISUALIZATIONS

Recommended Introductory Python Visualization Libraries:

- matplotlib
- seaborn
DATA BIAS
CONCLUSION

- Preparation for building a model often takes longer than building the model itself
- Steps are not strictly linear
- The quality of each of these preparation steps directly impacts the quality of your outcome
GETTING STARTED WITH PANDAS DATAFRAMES

Standard import statement

```python
import pandas as pd
pd.read_csv("Data_to_Annotate_25/3_of_25_project_data.csv")
```

Reads the specified CSV file into a DataFrame
COMBINE DATASETS AND REMOVE DUPLICATES

This concatenates the given dataframes (essentially stacks them on top of one another)

```
import pandas as pd

df = pd.concat([df1, df2, df3], ignore_index=True)
df.drop_duplicates(subset=None, keep='first', inplace=False)
```
Sometimes you want to merge dataframes on certain values as well.

```python
merged = pd.merge(left=df1, right=df2, left_on='ID', right_on='ID', how='outer')
```

Merges df1 and df2 on the ID column in each and keeps all values.
INFORMATION

Gives summary information about the dataframe

def info()
FILTERING DATA

Shows only data matching the specified condition

```python
df[df['PoolQC'].isna() == False]
```
DROP OR FILL NULL VALUES

- Deletes rows where there is a null value in the ‘Utilities’ column
  
  ```python
  df = df.dropna(subset=['Utilities'])
  df = df.fillna({'Functional':0})
  ```

- Fills null values with zeroes
INFORMATION ABOUT THE DATA

Output a statistical description of the column

```
print(df['GrLivArea'].describe())
df['GrLivArea'].hist()
```
STANDARDIZE/NORMALIZE DATA

Standardize data to have mean of 0 and standard deviation of 1

```python
df['number_tweets'] = (df['number_tweets'] - df['number_tweets'].mean()) / df['number_tweets'].std()
```

Normalize data so all values are between 0 and 1

```python
df['number_tweets'] = df['number_tweets'] / df['number_tweets'].abs().max()
```
TRAIN-VAL-TEST SPLIT

- Common package for working with arrays

```python
import numpy as np
train, validate, test = np.split(df.sample(frac=1), [int(.7*len(df)), int(.9*len(df))])
```

- Splits the data into three data sets
- Randomizes data before splitting
- Specifies where the splits should happen (i.e. between train and val at the 70% and between val and test at 90%)