Dask

Joseph John

National Computational Infrastructure, Australia

Outline

We will cover:

- Dask
- Dask-ML
- Dask-distributed

Cluster Support

- **project**: vp91
- **modules**: python3/3.9.2 cuda/12.0.0
- **venv**: /scratch/vp91/AAPP2023/dask-python3.9-venv
- **repo**: https://github.com/nci900/Dask-Data-Analytics

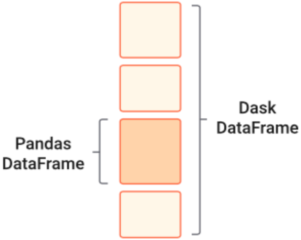
Dask

- Parallel and distributed computing library for python
- Dask scale up to your full laptop capacity and out to a cloud cluster
- Multi-core and distributed+parallel execution on larger-than-memory datasets

Dask Collection

- **High-level collections**: Mimic NumPy, lists, and pandas but can operate in parallel on datasets that don't fit into memory
  - Array
  - DataFrame
  - Bag
- **Low-level collections**: Give finer control to build custom parallel and distributed computations.
  - Delayed
  - Futures

Dask Dataframe

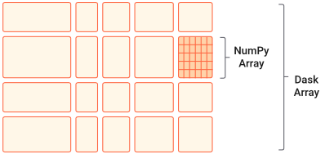


- One Dask DataFrame is comprised of many in-memory pandas DataFrames separated along the index
- One operation on a Dask DataFrame triggers many pandas operations on the constituent pandas DataFrames
- These operations are mindful of potential parallelism and memory constraints

Lazy Evaluation

- Dask constructs the logic (called task graph) of your computation immediately
- Evaluates them only when necessary

Dask Arrays



- Dask Array implements a subset of the NumPy ndarray interface using **blocked** algorithms
- Large array is cut into many small arrays
- Large computations are performed by combining many smaller computations

Dask Delayed Decorator

- A Block of code can have operations that can happen in parallel
- Normally in python this would happen sequentially or the user will identify the parallel section and write parallel codes
- The Dask **delayed** function decorates your functions so that they operate lazily
- Dask will defer execution of the function, placing the function and its arguments into a task graph
- Dask will then identify opportunities for parallelism in the task graph
- The Dask schedulers will exploit this parallelism, generally improving performance

Dask Future

- We can submit individual functions for evaluation
- The call returns immediately, giving one or more future
  - whose status begins as “pending”
  - later becomes “finished”
- There is no **blocking** of the local Python session.
- Difference between futures and delayed
  - delayed is lazy (it just constructs a graph)
  - futures are eager
- With futures, as soon as the inputs are available and there is compute available, the computation starts

Compute Vs Persist

- Dask executes the computations transformation to the distributed data.
- Compute: Converts it to a local object.
- Persist: The object remains distributed.

Dask ML

1. Dask-ML provides scalable machine learning in Python.
2. Works with ML libraries like Scikit-Learn, XGBoost.
3. Dask-ML API is similar to Scikit-Learn API.

Data Prepossessing

The first process step in building a machine learning model is data cleaning.
Data cleaning mainly involves:

1. Remove any unnecessary observations from your dataset
2. Remove redundant information
3. Remove duplicate information
4. Remove structural errors in data collection
5. Remove unwanted outliers - outliers can result in overfitting
6. Handle missing data:
   1. Remove observations with values missing
   2. Infer the missing values

Data Prepossessing

We are taking the easiest method to address missing values. We are removing any dataframe row that has missing values. , we end up not getting the entire picture. Inferring data is also not always a good idea as we may add some bias to the inference.

1. This is not always advisable as we are losing a lot of information and in the end.
2. Inferring data is also not always a good idea as we may add some bias to the inference.

.

Drop Missing Data

```python
import dask.dataframe as dd

ddf = dd.read_csv("weatherAUS.csv",
    dtype={'Humidity3pm': 'float64',
    'Humidity9am': 'float64',
    'WindGustSpeed': 'float64',
    'WindSpeed3pm': 'float64',
    'WindSpeed9am': 'float64'})

ddf_clean = ddf.dropna()
    ...
```

Types of data

Data cleaning mainly involves:

1. Categorical data groups information (usually text) with similar characteristics.
2. Numerical data expresses information in the form of numbers

Most machine learning algorithms cannot handle categorical variables unless it is converted to numerical data. This process is called **encoding**.

Encoding

Data cleaning mainly involves:

1. One-hot encoding
2. Label encoding
3. Target encoding
4. Frequency
5. Binary encoding
6. Feature Hashing

We will be using **One-hot** encoding (also called Dummy encoding).

One-hot Encoding

Team	**Colour**
1	Red
2	Blue
3	Green

One-hot Encoding

Team	Red_colour	Blue_colour	Green_colour
1	1	0	0
2	0	1	0
3	0	0	1

Categorical data in Dask

There are two types of categorical data in Dask

1. Known: categories are known statically (from the metadata).
2. Unknown: categories are not known statically (from the metadata).

One-hot Encoder

```
from dask_ml.preprocessing import Categorizer, DummyEncoder

de = DummyEncoder()
ddf_features_preproc = de.fit_transform(ddf_features.categorize())
    ...
```

Normalization

1. Normalization is the process of translating data into the range.
2. It is a good practice to normalize the data - especially useful when different features have different value ranges.
3. Normalization ensures that one feature does not overtly influence the model.

Normalization

```
from dask_ml.preprocessing import StandardScaler

scaler = StandardScaler()
scalar_norm = scaler.fit(ddf_features_preproc)
ddf_features_norm = scaler.fit_transform(ddf_features_preproc)
    ...
```

Correlation matrix

Correlation is often used in machine learning to identify multi-collinearity.

1. Two or more predictor variables are highly correlated with each other.
2. Multicollinearity can adversely affect the accuracy of predictive models.
3. The coefficients become very sensitive to small changes in the model.
4. Multicollinearity reduces the precision of the estimated coefficients, which weakens the statistical power of your regression model.
5. Multicollinearity can be addressed by removing one of the correlated variables.

Correlation matrix

1. Pearson’s correlation
2. Spearman’s correlation
3. Kendall’s Tau correlation

We will me using **Pearson’s correlation.**

Correlation matrix

```
corr_matrix = ddf_features_norm.corr(method='pearson',
    min_periods=None, numeric_only='__no_default__',
    split_every=False)
    ...
```

Principal Component Analysis

Principal component analysis, or PCA, is a that is often used to reduce the dimensionality of large data sets, by that .

1. Dimensionality reduction method.
2. Transform a large set of variables into a smaller one.
3. The reduced variable set will still contains most of the information in the large set.

Correlation matrix

```
from dask_ml.decomposition import PCA

pca = PCA(n_components=3)
pca.fit(ddf_features_norm.to_dask_array(lengths=True))
PCA(copy=True, iterated_power='auto', n_components=3,
    random_state=None, svd_solver='auto', tol=0.0, whiten=False)
```

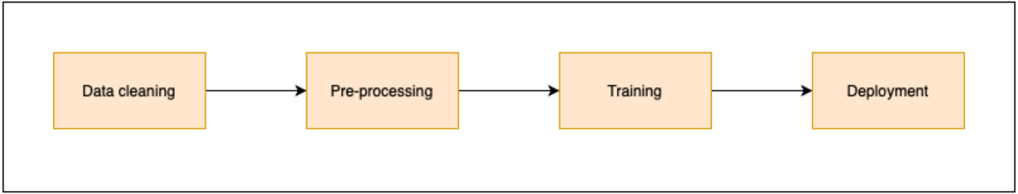
Cross Validation

Cross-validation is a method for evaluating ML models by training several ML models on subsets of the data and evaluating another subset of the data.
The advantages of cross validation are :

1. Identify Overfitting
2. Comparison between different models
3. Hyperparameter tuning
4. Efficiency : Allows the use of data for both training and validation

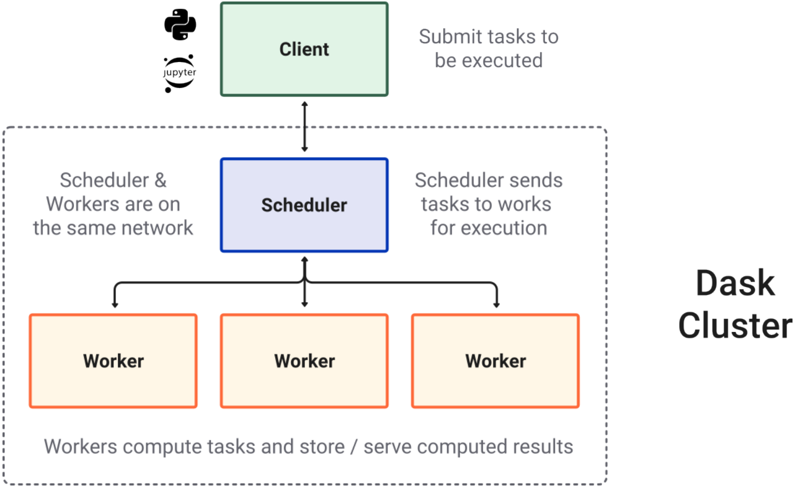
```
X_train, X_test, y_train, y_test=
    train_test_split(ddf_features_norm, ddf_target, shuffle=False)
```

ML Pipelines



1. Automate the workflows.
2. Pipeline links all steps of data manipulation.
3. Shortens code.
4. Improves code quality.

Dask Cluster



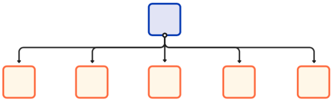
Dask Cluster

1. A cluster managers deploy a scheduler and the necessary workers.
2. Dask by default use single-machine scheduler.

Dask Cluster & Resource Manager

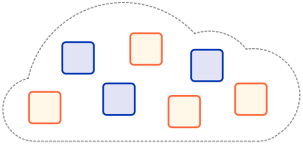
Cluster Manager

Deploys one Scheduler and many Workers by talking to the Resource Manager



Resource Manager

Kubernetes/Yarn/SLURM/PBS/Abstract pods/jobs on top of Physical Hardware



Physical Hardware

Physical CPUs, GPUs, networking and storage; either on-prem or on the cloud

Dask Cluster & Resource Manager

1. Cluster manager communicate with the resource manager to determine where the workers are running.
2. In Gadi we use PBS as the resource manager.
3. Other popular resource managers are Slurm and SGE.
4. Dask also works with Kubernetes.

Dask Jobqueue

1. Interfaces HPC resource manager with Dask Cluster manager.
2. Dask workers are automatically allocated physical hardware resources.

Dask Jobqueue

```
os.environ['DASK_PYTHON'] =
    '/scratch/vp91/AAPP2023/dask-python3.9-venv/bin/python'
setup_commands =
    ["module load python3/3.9.2",
    "source /scratch/vp91/AAPP2023/dask-python3.9-venv/bin/activate"]
extra = ['-q normal',
 '-P vp91',
 '-l ncpus=48',
 '-l mem=192GB']
```

Dask Jobqueue

```
cluster = PBSCluster(walltime="00:50:00",
             cores=48,
             memory="192GB",
             #processes=48,
             shebang='#!/usr/bin/env bash',
             job_extra=extra,
             local_directory='$TMPDIR',
             header_skip=["select"],
             interface="ib0",
             env_extra=setup_commands,
             python=os.environ["DASK_PYTHON"])
```

Dask Jobqueue

```
#!/usr/bin/env bash

#PBS -N dask-worker
#PBS -l walltime=00:50:00
#PBS -q normal
#PBS -P vp91
#PBS -l ncpus=48
#PBS -l mem=192GB
module load python3/3.9.2
source /scratch/vp91/AAPP2023/dask-python3.9-venv/bin/activate
/scratch/vp91/Training/python3.9-venv/bin/python -m distributed.cli.dask_worker
tcp://10.6.41.71:36145 --nthreads 6 --nprocs 8 --memory-limit 22.35GiB
--name dummy-name --nanny --death-timeout 60 --local-directory $TMPDIR
--interface ib0 --protocol tcp://
```

Thank You!