Building a data science solution for an NGO when you don't know what infrastructure it will run on

A case study predicting tutor supply and demand mismatch

DataKind UK

- ➔ Who am I?
- ➔ Who are DataKind UK & The Brilliant Club?
- ➔ What did we set out to do?
- ➔ How we did it!
- ➔ The final result

Adam Hill

tldr;

COMPLY ADVANTAGE

Senior Data Scientist

Financial Crime Fighter

Day Job

Charity

DataKind volunteer

DataDive volunteer; Data Ambassador; Committee member

DataKind

Former Royal Society Entrepreneur in Residence

Recovering astrophysicist

Evangelist

Me

Find me at

@astroadamh

www.linkedin.com/in/adambenhill/

Horsewithapointyhat.com

UNIVERSITY OF Southampton

THE ROYAL SOCIETY

DataKind UK

We're a charity that builds data science capacity in charities, social enterprises, and the public sector


DATA FOR GOOD
Meningitis NOW


DATA FOR GOOD


DATA FOR GOOD


Nicole DKUK (she/her)
Giselle - DKUK - they
Suzy DKUK (She/her)
tyndyll
Joe - checked out
Dulcie DKUK she/her
Jenny CAB she/her
Paul TBC (he/him)
Emiliano TBC (he/him)
Sara TBC (she/her)
danielsharp
Oli TBC he/him
Lottie TBC (she/her)
katy sinka
robertofierimonte
Paolo, BC, troubleshoote...
Tahir Mahmood
Vincent
Yvonne
Fantasy Yap

The Brilliant Club

The Brilliant Club supports less advantaged students to access and succeed in the UK's most competitive universities.

They do this by mobilising the PhD community to support students in schools via their courses and tutoring programme.

Team Effort!

Six dedicated DataKind volunteers

- ➔ Meg
- ➔ Kate
- ➔ Leo
- ➔ Chris
- ➔ Chris

What was the problem we needed to solve?

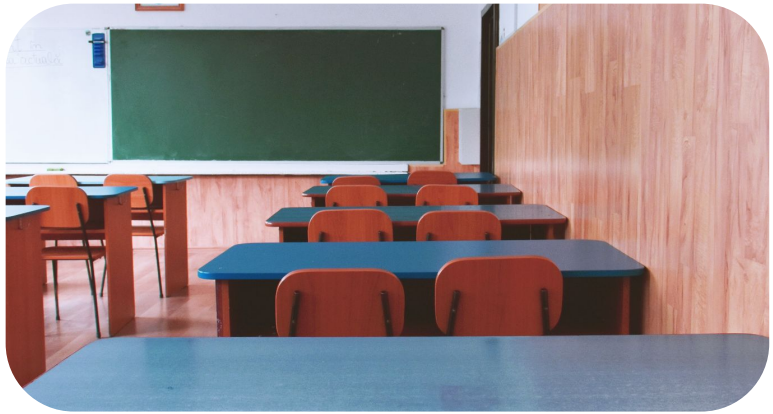
The Challenge

The Brilliant Club's 21-26 strategy focuses on two goals: student access and student success. TBC wants more pupils from less advantaged backgrounds to access university

To achieve this they need:

- ➔ Scalable systems and processes
- ➔ The right number of PhD researcher volunteers in the right areas to meet school demand

How can we estimate the number of new tutors TBC needs to recruit over a coming school year?

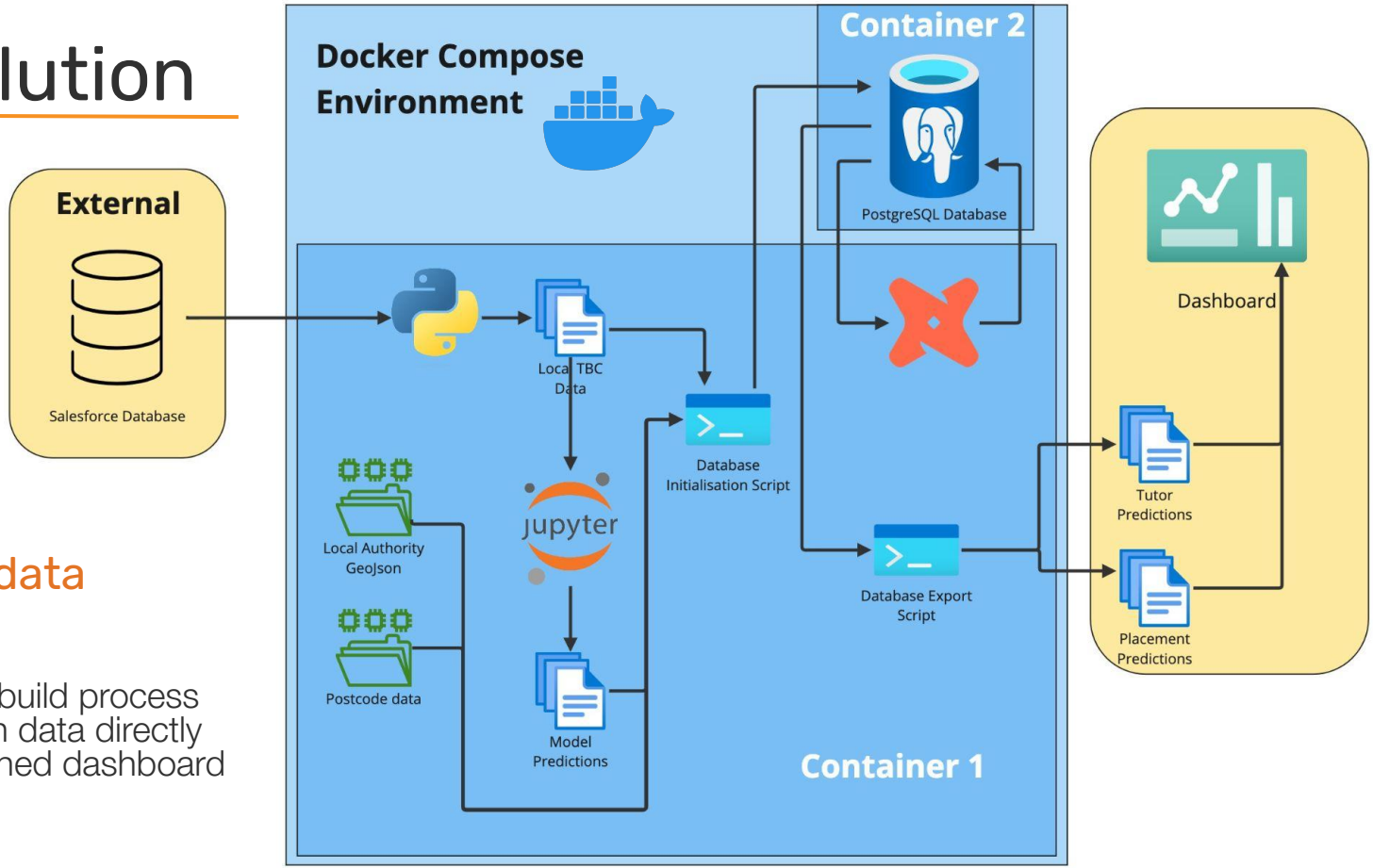


Current Solution

Spreadsheets Galore!

- ➔ Multiple teams, copy and paste snapshot data from database reports
- ➔ Manually build reports and charts, but it's easy to lose track of hard-coded numbers versus calculated figures

tldr;
Our Solution



Easy-to-use data pipeline

- ➔ One click data build process
- ➔ Generates fresh data directly into an established dashboard

Collaboratively working as a volunteer team

Coordination and support

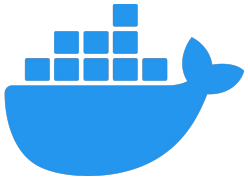
As volunteers working in our spare time we have a challenge to make sure we can move forward coherently but with flexibility to account for everyone's lives.

Our manner of working needs to allow

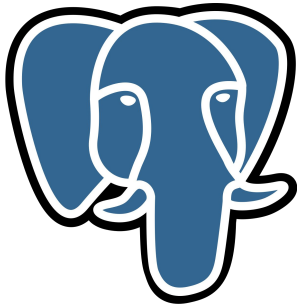
- ➔ Asynchronous work patterns
- ➔ Minimise dependency between individuals
- ➔ Leverage everyone's skills maximally

Leveraging the team skills

Open Source free* Tools



docker



+ a b | e a u

**I know Tableau isn't free but it's what the charity already uses*

Control the Environment

Github

Makes it easy to share the code between us - and with the charity.

Each dev creates their own branch when developing. Allows independent experiment but we require **eventual consistency.**

Poetry

Manages equivalent coding environments for each user.


```
[tool.poetry]
name = "brilliantclub"
version = "0.1.0"
description = "DataCorps tutor forecasting project for the Brilliant Club"
authors = ["TBC DataCorps <brilliantclubproject@datakind.org.uk>"]
license = "CC0 1.0 Universal"

[tool.poetry.dependencies]
python = "^3.9"
pandas = "^1.4.2"
jupyter = "^1.0.0"
jupyterlab = "^3.4.7"
simple-salesforce = "^1.12.1"
PyYAML = "^6.0"
openpyxl = "^3.0.10"
plotly = "^5.9.0"
absl-py = "^1.2.0"
tqdm = "^4.64.0"
dbt-postgres = "^1.3.0"
matplotlib = "^3.5.3"
matplotlib-venn = "^0.11.7"
xlsxwriter = "^3.0.3"
sqlalchemy = "^1.4.42"
psycopg2-binary = "^2.9.5"
icecream = "^2.1.3"
setuptools = "^65.6.0"
scikit-learn = "^1.2.1"
joblib = "^1.2.0"
prefect = "^2.8.0"
prefect-shell = "^0.1.5"
prefect-dbt = "^0.3.1"
prefect-jupyter = "^0.2.0"
```

Control the Environment

Dockerfile > ...

You, 3 months ago | 2 authors (Chris Watkins and others)

```dockerfile
# syntax=docker/dockerfile:1
FROM python:3.9-slim-buster

# Install build-essentials to facilitate installation of Python dependencies
RUN apt-get update --fix-missing && apt-get install -y build-essential
RUN apt-get install -y git postgresql-client

# Skip writing .pyc files
ENV PYTHONDONTWRITEBYTECODE=1 \
    PYTHONUNBUFFERED=1 \
    # Pin a specific Poetry version for maximum compatibility
    POETRY_VERSION=1.2.2

WORKDIR /code/

RUN pip install "poetry==$POETRY_VERSION"
RUN poetry config virtualenvs.create false

COPY poetry.lock pyproject.toml /code/
RUN poetry install --no-interaction

# Expose default jupyter notebook port.
EXPOSE 8888

# Copy Jupyter launch script
COPY launch.py /code/

ENTRYPOINT [ "/bin/bash" ]
```

Docker Compose

docker-compose.yaml > version

docker-compose.yml - The Compose specification establishes a s… of multi-container platform-agnostic app

```yaml
version: "3.9"
services:
  jupyterlab:
    build: .
    tty: true
    env_file:
      - .env
    environment:
      POSTGRES_HOST: postgres
      REFRESH_DATA:
    ports:
      - "8888:8888"
      - "8889:8889"
      - "8080:8080" # For dbt docs serve
    working_dir: /app
    volumes:
      - ./src:/code/src
      - ./notebooks:/code/notebooks
      - ./data:/data
      - ./exported_data:/exported_data
      - ./:/app
    depends_on:
      - postgres
    entrypoint: "python launch.py"
    restart: always

  postgres:
    image: postgres:15-alpine
    environment:
      POSTGRES_USER:
      POSTGRES_PASSWORD:
      POSTGRES_DB :
    volumes:
      - ./data:/data
      - ./exported_data:/exported_data
      - ./init.sql:/init.sql
      - ./export.sql:/export.sql
    ports:
      # NOTE: "5432:5432" makes the Postgres server accessible to both the host
      # developer machine *and* the "app" container in Docker. If you don't want
      # it available on the host machine, change this to simply "5432".
      - 5432:5432
    restart: always
```

Automate the data pull

Connect to Salesforce

Pull the latest TBC data

- ➔ Registered tutors
- ➔ Tutor availability reports
- ➔ Confirmed school placements
- ➔ Candidate school placements

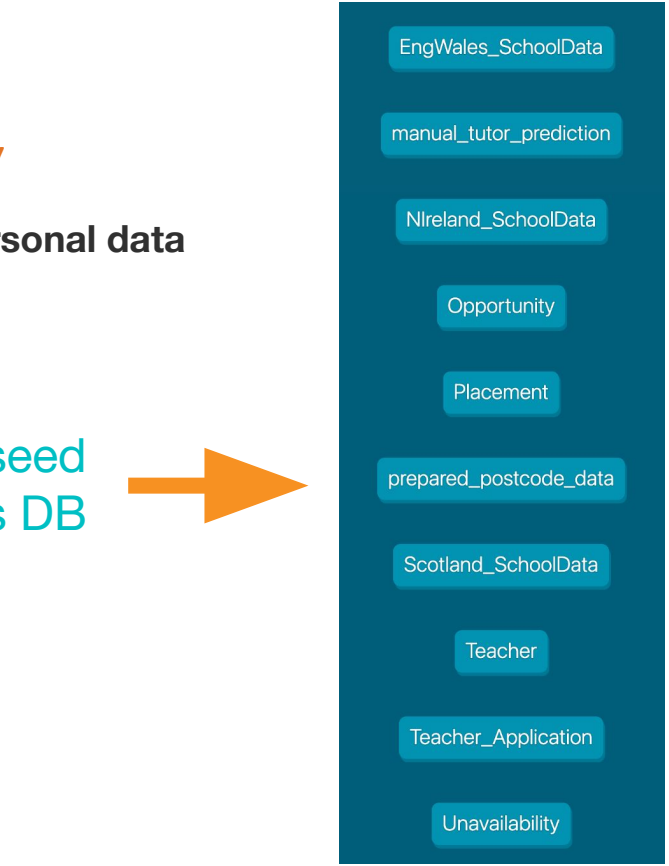
Data Enrichment

- ➔ Local Authority geo boundaries
- ➔ Postcode geo data

PII Data Security

SHAhash256 any personal data

SQL scripts to seed the Postgres DB



EDA, model training & prediction

Jupyter Lab EDA

- ➔ Explored initial dataset which enabled us to identify missing or “dirty” data with TBC
- ➔ Started to experiment with potential features sets
- ➔ Identified the typical return rate(s) when tutors were asked to submit their upcoming availability

Primary Feature Generation

Predicting an individual tutors availability

- ➔ Time to PhD Completion
- ➔ Previous number of placements
- ➔ Mean number of placements per term
- ➔ Number of terms since they registered
- ➔ Whether they did a placement last term.

Model Building

- ➔ Scikit-learn Random Forest Classifier
- ➔ From the trained model extract the individual tutor availability probabilities to facilitate better aggregate population statistics
- ➔ Convolve the availability probability with the population probability that an availability response will be submitted

Final output is a per-tutor availability probability for 1,2 & 3 terms in the future

Reusability of the modelling

Turning notebooks into scripts

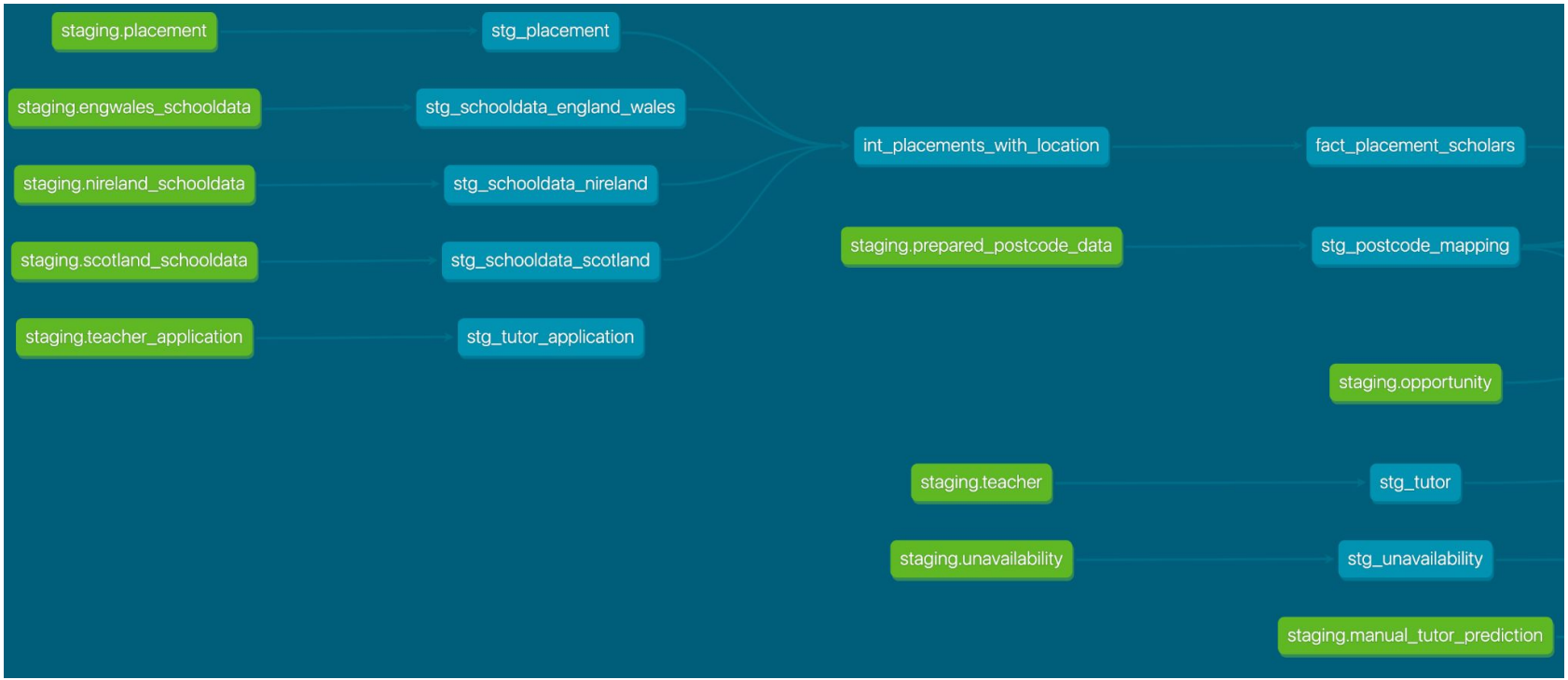
- ➔ Refactored the code into local python files that can be imported into notebooks or scripts
- ➔ Make use of dedicated dataclasses for tutors, school terms etc
- ➔ De-couple the code to make it easier to substitute into different models or add new features

Jupyter Lab Training & Prediction

- ➔ Two notebooks
  a. Train a new model and report accuracy metrics
  b. Load an appropriate model and make new predictions
- ➔ Keep the key details visible to users

```
Running - FutureTerm_1
Predict: 472 tutors in SchoolTerm(year=2023, name='Summer')
Running - FutureTerm_2
Predict: 436 tutors in SchoolTerm(year=2023, name='Autumn')
Running - FutureTerm_3
Predict: 489 tutors in SchoolTerm(year=2024, name='Spring')
```

Transform and load the data



Transform and load the data

Orchestrate the pipeline

```python
import os
from pathlib import Path

from prefect import flow, task
from prefect_shell import ShellOperation
from prefect_dbt.cli.commands import DbtCoreOperation
from prefect_jupyter import notebook

from icecream import ic

ic.configureOutput(prefix='Pipeline | ')

# Setup env variables
## Postgres
PGUSER = os.environ.get("PGUSER")
PGHOST = os.environ.get("PGHOST")
PGPORT = os.environ.get("PGPORT")
PGDB = os.environ.get("PGDB")

# Jupyer Modelling
NOTEBOOKS_PATH = Path("/app/notebooks/pipeline")
MODEL_NOTEBOOK_NAME = "02.Tutor-Modelling"
PREDICT_NOTEBOOK_NAME = "03.Tutor-predictions"
MODEL_YEAR_CUTOFF = os.environ.get("MODEL_YEAR_CUTOFF", 2019)
MODEL_PREDICT_START_YEAR = os.environ.get("MODEL_PREDICT_START_YEAR")
MODEL_PREDICT_START_TERM = os.environ.get("MODEL_PREDICT_START_TERM")

@task(name="Validate Model Variables",
      description="Task to validate that the model variables are all set with reasonable values",
      task_run_name="validate-model-variables",
      retries=0)
def env_validation_task():…

@task(name="Check location files",
      description="Task that checks the pre-downlaoaded location files are where they are expected",
      task_run_name="check-location-files-present",
      retries=0)
def check_presence_location_files():…
```

```python
@flow(name="TBC-DKUK End-to-End modelling pipeline",…
def end_to_end_pipeline() -> str:
    """ Runs the complete TBC data modelling pipeline"""

    # Validate the environment
    _ = env_validation_task()

    # Check for presence of location files
    _ = check_presence_location_files()

    # Download the data
    with ShellOperation(commands = ["python -m src.utils.salesforce_to_csv --reports=false --write_to /data"],
                        working_dir="./") as download_data:
        download_data_process = download_data.trigger()
        download_data_process.wait_for_completion()
        output_lines = download_data_process.fetch_result()
    msg = "Completed Data Download!"
    ic(msg)

    # Run the Jupyter training notebook
    modelling_nb = notebook.execute_notebook(…
    body = notebook.export_notebook(modelling_nb)
    output_path = NOTEBOOKS_PATH / f"{MODEL_NOTEBOOK_NAME}_latestResult.ipynb"
    with open(output_path, "w") as outfile:…
    msg = "Completed Rebuild of Tutor Model"
    ic(msg)

    # Run the Jupyter prediction notebook
    prediction_nb = notebook.execute_notebook(…
    body = notebook.export_notebook(prediction_nb)
    output_path = NOTEBOOKS_PATH / f"{PREDICT_NOTEBOOK_NAME}_latestResult.ipynb"
    with open(output_path, "w") as f:…
    msg = f"Completed prediction of future Tutor availability starting from {MODEL_PREDICT_START_YEAR}-{MODEL_PREDICT_START_TERM}"
    ic(msg)

        You, now • Uncommitted changes
    # Initialise the database
    with ShellOperation(commands = [f"psql -U {PGUSER} -h {PGHOST} -p {PGPORT} -d {PGDB} -f 'init.sql'",],…
        init_database_process = init_database.trigger()
        init_database_process.wait_for_completion()
        output_lines = init_database_process.fetch_result()
    msg = "Completed PostgreSQL database initialisation ..."
    ic(msg)

    # DBT Commands
    dbt_result = DbtCoreOperation(…
    msg = "Completed DBT data pipeline build ..."
    ic(msg)

    # Export data files
    with ShellOperation(commands = [f"psql -U {PGUSER} -h {PGHOST} -p {PGPORT} -d {PGDB} -f 'export.sql'",],…
        export_database_process = export_database.trigger()
        export_database_process.wait_for_completion()
        output_lines = export_database_process.fetch_result()
    msg = "Completed PostgreSQL prediction data export"
    ic(msg)

    return "Pipeline Run completed!"

if __name__ == "__main__":
    # Run the complete pipeline
    end_to_end_pipeline()
```

Present the data visually

Dashboarding

Some parts of The Brilliant Club business are already using Tableau so they chose that component. Alternative choices could have been:

- ➔ Streamlit
- ➔ Plotly
- ➔ PowerBI

The aim of the dashboard is to easily surface insight into the predicted and actual number of tutors/placements across the country and within different geographic boundaries

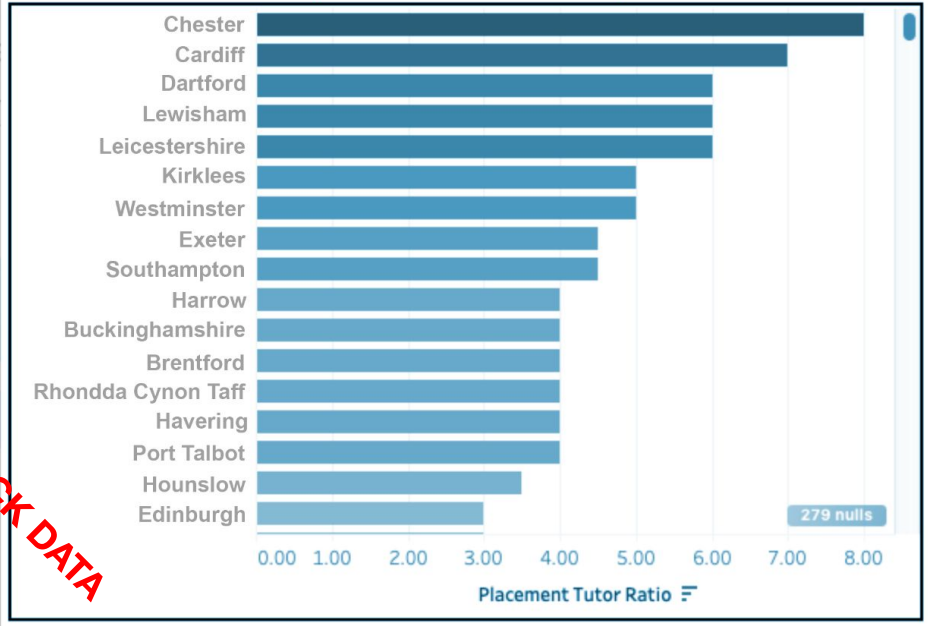
Overall Summary

Placement : Tutor Ratio

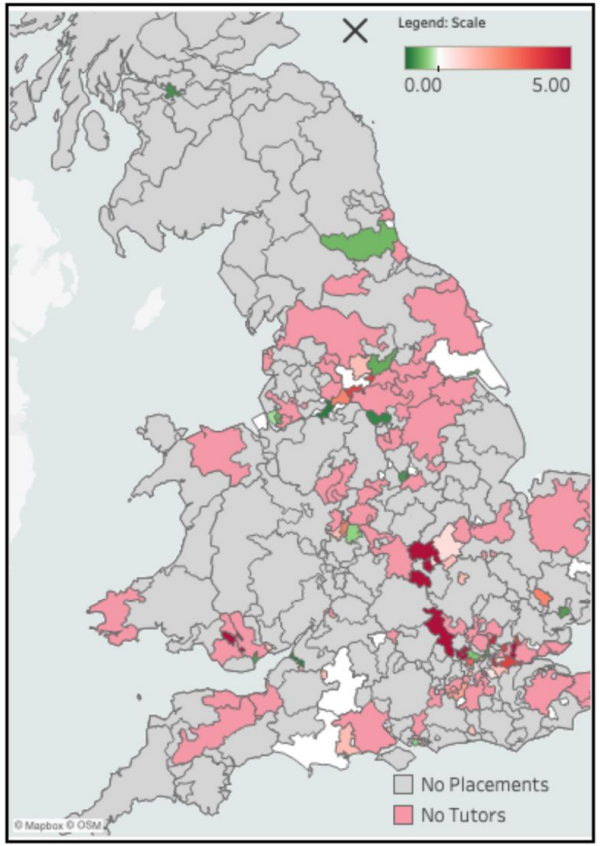
MOCK DATA

Term/Year	Select Metric	Select Geographic View	Region Name	Use placement prediction?	Use tutor prediction?
Spring - 2021-22	Placement : Tutor Ratio	Local Authority	(All)	Yes / No	Yes / No

Total Tutors	Total Placements	Placement/Tutor Ratio
423	467	0.91

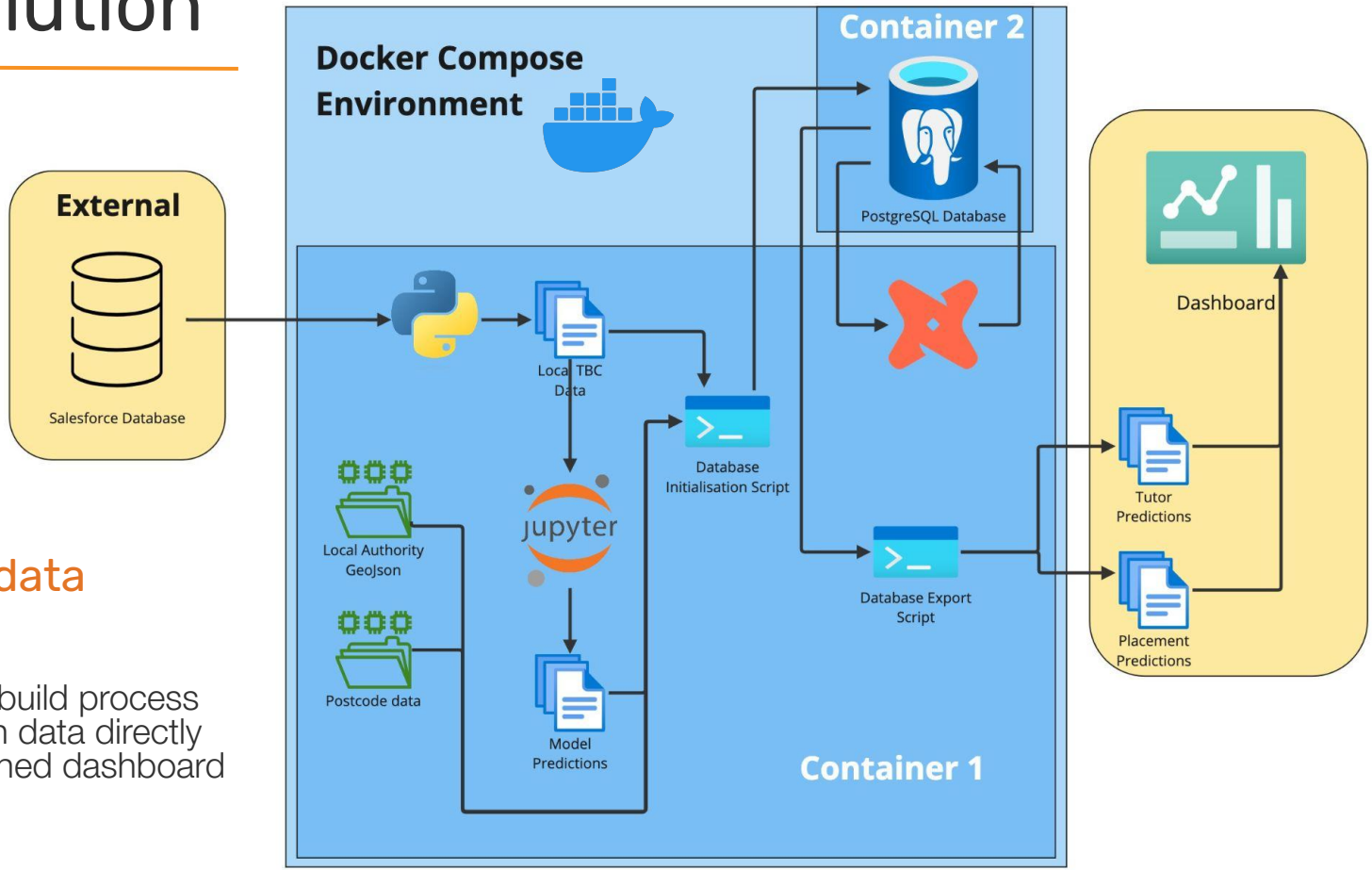


MOCK DATA



Note: "Use placement prediction" and "Use tutor prediction" will use placement/tutor predictions respectively, if selected. Placement predictions are based on confirmed placements, plus a percentage of likely Opportunities. Tutor predictions are based on past tutor behaviour.

The Solution



Easy-to-use data pipeline

- ➔ One click data build process
- ➔ Generates fresh data directly into an established dashboard

What impact will this have?

The goal

Use data on tutor availability and on sales placements to forecast and visualise the mismatch between tutor supply and demand, regionally to enable The Brilliant Club to decide which areas to prioritise for recruitment, so that sufficient tutors are available to meet the needs of partner schools.

Desired outcomes

1. School sales are easier to manage with less back and forth conversation with the programme team about how placing is going/whether we have tutors available in a given area
2. TBC more accurately meet recruitment targets regionally, securing tutors in the areas with highest school demand.
3. TBC need to reduce fewer school sales due to being unable to predict tutor need.
4. TBC is able to meet placement demands without undue burden on tutors who already have a placement or The Brilliant Club staff

An impact assessment will happen at a future date

Questions?

contact@datakind.org.uk
www.datakind.org.uk
@datakinduk

www.linkedin.com/in/adambenhill/
Horsewithapointyhat.com
@astroadamh