Thought Process and Design Decisions

Objective (Givens)

Challenge

- Create a dataset for training an ML model using the airFRANS dataset.
  - The dataset should provide a sequence of points with their SDF (distance from the airfoil) value as the input (x,y,sdf) and the velocity (x,y,vx,vy) as the target. Package the data such that it can be quickly loaded for training a model.
- Provide some dataset statistics to help users understand the data.
- Document your design decisions.

Plan

▼ 1 - Problem: Define constraints and objectives

1. Define constraints and objectives to make open-ended problem approach an obvious, specific solution

▼ 2 - Approach: Define tools

1. Read through all given resources, nothing has been given by accident.
2. The 'open-endedness' isn't really open-ended - airfrans has recommendations, and the problem clues on what tools to use.
   a. In fact, the airfrans docs say pyvista is installed with airfrans - meaning the problem explicitly gives pyvista as a direct clue on what to use, the explicit additional installation is not necessary, PLUS its dependencies (correction. Pyvista is needed for to read in data)

```
!pip install airfrans --quiet
!pip install pyvista --quiet
!sudo apt install libgl1-mesa-glx xvfb
```

▼ 3 - Execution: Read in the data

1. Keep it simple. Clone the problem and see if anything comes up when running it - no need for Kaggle or anything like that except for some reference.
2. AirFRANS references two datasets, but since instructions are for loading preprocessed, assume that.

▼ 4 - Execution: Explain Data through Visualization

1. Most critical part. Knowing the physics is good but knowing the dataframe, how to load it and visualize it is critical for the objective.
2. One of the objectives is "provide some dataset statistics to help users understand the data", meaning *you* must first understand the data.
3. Really, the purpose of the assignment is:
   a. Figure out from the givens what unknowns are actually not unknown.
   b. Teach the user about the dataset through stat and raw data visualizations
   c. Prepare data for training a model in a clean, modular and scalable way
   d. Present thought process during the whole assignment
   e. Present data in a production format
4. Understand what the data represents, its format, some standard graphs for ML, and some visualization to show what the data is even in before you process it.
5. So show what each data point represents, then show data relations in processing, then remove columns and export

▼ 5 - Execution: Exporting the Data

1. Run standard ML preparation to clean dataset, even if it's the preprocessed form
2. Engineer dataset to match the system requirements of "The dataset should provide a sequence of points with their SDF (distance from the airfoil) value as the input (x,y,sdf) and the velocity (x,y,vx,vy) as the target."
3. Export the data optimized for Pytorch for efficiency (maybe numpy?) and csv for universal compatibility

▼ 6 - Presentation

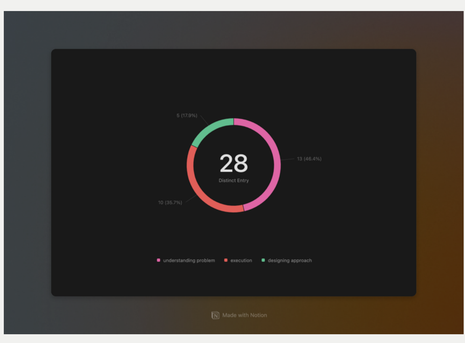
1. The file structure is "up to you" but base it on industry standards (UDEMY production code file structure plus this guide).
   a. So structure is:
      i. root.README.md (why does this exist) and root.requirements.txt (packages to be reflected earlier)
      ii. root.data.processed and (possibly) root.data.raw
      iii. root.notebooks.01_Data_Preparation.ipynb (processing) and root.notebooks.02_EDA.ipynb (visualization)
      iv. root.projectname.setup.py (possibly. may be better to just do this in readme if involving bash commands, though this would be clean.)
         1. make sure to include virtual env ifi applicable
2. The main objective is to see how you code, so set up braindump in such a way to have more data than just entries, answers, and exploration without burdening bandwidth or allotted time for problem.

Design decisions

- Using PyTorch Geometric over Deep Graph Library for data structure as Cameron mentioned the model being a PyTorch model.
- Chose between PyTorch Geometric over Deep Graph Library in the first place over pandas or something else because airFRANS documentation highly recommended one or the other
- File structure based on standards from UDEMY download structure and optimized with this well-known guide
- Used code based on interview emphasis of python ETL, so follow PEP 8 protocol, specifically with functions, & zen of python where applicable
- Features of input columns (x,y,sdf) and target columns (x,y,vx,vy) explicitly defined as objective
- Only loading preprocessed data from airfrans considering documentation only shows to download that set of the two (as opposed to raw).
- Normalized data as it's generally standard for Physics-Informed Neural Networks (PINNs) and Implicit Neural Representations (INRs).
- Assuming Pytorch implementation of Physics-Informed Neural Networks (PINNs) and Implicit Neural Representations (INRs) from ML Engineer Navier job announcements.

Statistics

Generated automatically from thought stream - loose time estimate based on entry number.



Braindump

data likely needs to be normalized given span of values. no other prep seems applicable	execution
you could load the torch dataset directly, as another approach, but to not risk the data being different will stick with give	execution
also redundancy in the ask. making x,y, vx, vy as targets really is just vx and vy as targets.	execution
i was wrong about pyvista. it's just needed for reading these files. the redundancy was just a mistake.	execution
definitely need to just copy the documentaion and go from there. get a foothold against a proper source of truth.	execution
need to run small checks before scaling. will save a chunk of the foil array to prevent memory erasing during image crast	execution
right, no random numpy array, can use pyvista's. some trouble with rendering the interactive points in collab, i can disabl	execution
First visualization should be based on: airfrans.dataset.load contains point clouds defined as the nodes of the simulation	execution
trouble displaying the point cloud in google collab. wonder if that's a non-kaggle issue or pyvista	execution
okay, displaying a random numpy array with similiar dimensions in pyvista to see how i'd view any data in point cloud. sta	execution
These were all run on OpenFOAM solver, keep that in mind for formatting later	understanding problem
Nvm, simplest solution first. No need to reinvent anything. Copy and run the notebook instead of trying Kaggle, let's see	execution
Based on the docs there's a bool that dictates if the point is on the airfoil. Maybe simplest solution to hit first goal of visu	understanding problem
Okay, above now written. It seems this is mainly an Easter egg hunt, finding what's required from resources.	designing approach
The env memory needed is proportional to the % of dataset needed, if it can be sliced prior to downloading	understanding problem
Airfrans suggests pytorch geometric or deep graph library for data manipulation.	understanding problem
Yep. The airfrans dataset includes a preprocessed set and raw set. Likely just need preprocessed, but will run checks to	designing approach
None of the givens are by accident. pyvista is for 3D visualization, the airFrans docs likely give structure to this otherwis	designing approach
pyvista was explicitly listed in the downloads despite being redundant with airfrans dataset according to docs. Likely imp	designing approach
From the docs the data is NACA defined airfoils. Each point has a distance to the airfoil surface. As sanity check expect a	understanding problem
Writing in python so follow PEP 8, Zen of python, and setup virtual env if use case allows. A README and notebook also !	designing approach
Objective states dataset statistics to understand data. Reminds me of my titanic neural net. Likely will preprocess if not a	understanding problem
This is open ended, but there's definitely a non-wrong way to do this. Common processes always have a standard. Assur	understanding problem
These are NACA defined airfoils for imcompressible flow, likely because all flight in 1915 was so slow it could be assume	understanding problem
Data actually too big to download. Moving to Google space. Yeah, figuring this now. Hm. Okay, the dataset is on kaggle b	understanding problem
How we export this project as a whole matters. Thinking this stream of thought + notebook + Readme. I remember UDEM	understanding problem
Planning on using Kaggle to have dedicated ML ecosystem with projects for ref - dataset is gigantic, 20 min to download	understanding problem
Cameron mentioned using pytorch for the model - optimize the final dataset with efficient format for this, and csv if they	understanding problem
Okay, open ended project. Need first to assign system reqs and constraints to make next steps more obvious	understanding problem