# Amazon ML

**Problem Statement**

**Feature Extraction from Images**

In this hackathon, the goal is to create a machine learning model that extracts entity values from images. This capability is crucial in fields like healthcare, e-commerce, and content moderation, where precise product information is vital. As digital marketplaces expand, many products lack detailed textual descriptions, making it essential to obtain key details directly from images. These images provide important information such as weight, volume, voltage, wattage, dimensions, and many more, which are critical for digital stores.

Data Description:

The dataset consists of the following columns:

1. **index**: A unique identifier (ID) for the data sample.
2. **image_link**: Public URL where the product image is available for download. Example link - https://m.media-amazon.com/images/I/71XfHPR36-L.jpg To download images, use the download_images function from src/utils.py. See sample code in src/test.ipynb.
3. **group_id**: Category code of the product.
4. **entity_name**: Product entity name. For example, “item_weight”.
5. **entity_value**: Product entity value. For example, “34 gram”.

**Note**: For test.csv, you will not see the column entity_value as it is the target variable.

Output Format:

The output file should be a CSV with 2 columns:

1. **index**: The unique identifier (ID) of the data sample. Note that the index should match the test record index.
2. **prediction**: A string which should have the following format: “x unit” where x is a float number in standard formatting and unit is one of the allowed units (allowed

units are mentioned in the Appendix). The two values should be concatenated and have a space between them.
For example: “2 gram”, “12.5 centimetre”, “2.56 ounce” are valid.
**Invalid cases**: “2 gms”, “60 ounce/1.7 kilogram”, “2.2e2 kilogram”, etc.
**Note**: Make sure to output a prediction for all indices. If no value is found in the image for any test sample, return an empty string, i.e., “”. If you have less/more number of output samples in the output file as compared to test.csv, your output won’t be evaluated.

File Descriptions:

**Source Files:**

1. **src/sanity.py**: Sanity checker to ensure that the final output file passes all formatting checks.
   **Note**: The script will not check if fewer/more number of predictions are present compared to the test file. See sample code in src/test.ipynb.
2. **src/utils.py**: Contains helper functions for downloading images from the image_link.
3. **src/constants.py**: Contains the allowed units for each entity type.
4. **sample_code.py**: A sample dummy code that can generate an output file in the given format. Usage of this file is optional.

**Dataset Files:**

1. **dataset/train.csv**: Training file with labels (entity_value).
2. **dataset/test.csv**: Test file without output labels (entity_value). Generate predictions using your model/solution on this file's data and format the output file to match sample_test_out.csv (Refer to the "Output Format" section above).
3. **dataset/sample_test.csv**: Sample test input file.
4. **dataset/sample_test_out.csv**: Sample outputs for sample_test.csv. The output for test.csv must be formatted in the exact same way.
   **Note**: The predictions in the file might not be correct.

Constraints:

1. You will be provided with a sample output file and a sanity checker file. Format your output to match the sample output file exactly and pass it through the sanity checker to ensure its validity.

**Note**: If the file does not pass through the sanity checker, it will not be evaluated. You should receive a message like Parsing successful for file: ...csv if the output file is correctly formatted.

2. You are given the list of allowed units in constants.py and also in the Appendix. Your outputs must be in these units. Predictions using any other units will be considered invalid during validation.

Evaluation Criteria:

Submissions will be evaluated based on the **F1 score**, which is a standard measure of prediction accuracy for classification and extraction problems.

Let **GT** = Ground truth value for a sample and **OUT** be the output prediction from the model for a sample. Then we classify the predictions into one of the 4 classes with the following logic:

1. **True Positives**: If OUT != "" and GT != "" and OUT == GT
2. **False Positives**: If OUT != "" and GT != "" and OUT != GT
3. **False Positives**: If OUT != "" and GT == ""
4. **False Negatives**: If OUT == "" and GT != ""
5. **True Negatives**: If OUT == "" and GT == ""

Then,
**F1 score** = 2 * Precision * Recall / (Precision + Recall)
where:

1. **Precision** = True Positives / (True Positives + False Positives)
2. **Recall** = True Positives / (True Positives + False Negatives)

Submission File:

Upload a test_out.csv file in the portal with the exact same formatting as sample_test_out.csv.

Key Instructions:

- Challenge Window: 12:00 AM IST, 13th September 2024 to 11:59 PM IST, 15th September 2024
- All teams will get access to the problem statement with the data set on day 1 and will have time to build and submit solutions till day 3.
- Teams can track their performance through the leaderboard which will reflect team rankings live over the course of this challenge.
- Please use this link to ask any queries during the hackathon.
- Below mentioned artefacts need to be shared for the best solution submitted by the team:
    - 1-2 pager document explaining the ML approach, ML models used, experiments and conclusion.
    - Source code used for experiments, training and inference, with proper comments describing the functions.
- Each team can make a maximum of 15 submissions over the 3 days of Hackathon. After which the submit button will be disabled.
- Post successful submission of the artefacts, leaderboard score and each team member satisfying the eligibility criteria, the top 50 teams will be announced on 18th September 2024. Top 10 teams will be invited for the virtual grand finale on 24th September 2024.
- Disclaimer: As part of the Amazon ML Challenge hackathon event, we request that you refrain from using publicly or commercially available large language model (LLM) APIs such as those provided by OpenAI, Anthropic, Microsoft, Facebook, Google or other AI companies. The submissions using any of the LLM APIs will be discarded.

Simultaneous Logins and Accessibility:

- You can attempt the assessment on any desktop or laptop only and not on a mobile device.
- Simultaneous logins are not allowed i.e. you can only attempt the assessment from one laptop or desktop.

- In case simultaneous logins are detected, the system may terminate the assessment altogether, and you may only get error messages.

Other instructions:

- If you face any technical problem, clear your browser's cache or try it on a different browser or in incognito mode.
- You may also try changing your internet - mobile hotspot, wifi, etc.
- Please shoot an email to support@unstop.com with a screenshot of the page where you are facing a problem and your registered email ID. Please note that we won't be helping you make decisions and any email asking us to make decisions will not be entertained.