National University of Singapore (NUS)

Mult-Lingual Text Classification Explanation

THE NEXT FRONTIER FOR CLASSIFICATION: EXPLANATION

User Guide Document

Content Page

Executive Summary

This user guide document contains all the information required for a user to utilise the multi-lingual text classification model explainer, Langsplain.

Langsplain is designed to accurately explain predictions in multiple languages. The model uses Local Interpretable Model-Agnostic Explanations (LIME) technology to explain predictions made by text classification models, allowing X to validate these models and gain insights into public sentiment through social media posts, feedback, and other forms of written communication.

One of the key benefits of Langsplain is its ability to explain models across multiple languages. This is particularly important for X which operates in a multilingual Singaporean society and needs to understand communication mediums across different languages.

Overall, Langsplain provides a powerful tool for X to understand and validate their text classification models.

Introduction

Langsplain is a web application developed as an explainer aimed to explain predictions of machine learning (ML) text classification models, including any pre-trained models from SimpleTransformers library and Transformers library, for stance and emotion. The explainer also interprets and explains an original ML model, Double-headed classifier, developed by the other NUS group.

Langsplain offers comprehensive functionality by considering the classification type, such as stance or emotion, and providing detailed explanations of ML model predictions in relation to language - English, Mandarin, or Malay - and Named-Entity-Recognition (NER) categories. With Langsplain, users can easily navigate the complexities of machine learning and gain valuable insights into the outputs of their text classification models.

Underlying Technology

Local Interpretable Model-Agnostic Explanations (LIME)

Langsplain utilises LIME as the technology to output explanations for ML models.It works by training a simpler, interpretable model (called an "explainer") that approximates the behaviour of the complex model in a local region around the prediction of interest. The explainer is then used to generate explanations for individual predictions by highlighting the most important features that contributed to the prediction. LIME is model-agnostic, meaning it can be applied to any type of machine learning model, and has been used in various domains to improve the interpretability and trustworthiness of complex models. By providing explanations for the predictions made by a machine learning model, LIME can help build trust in the model and improve its transparency.

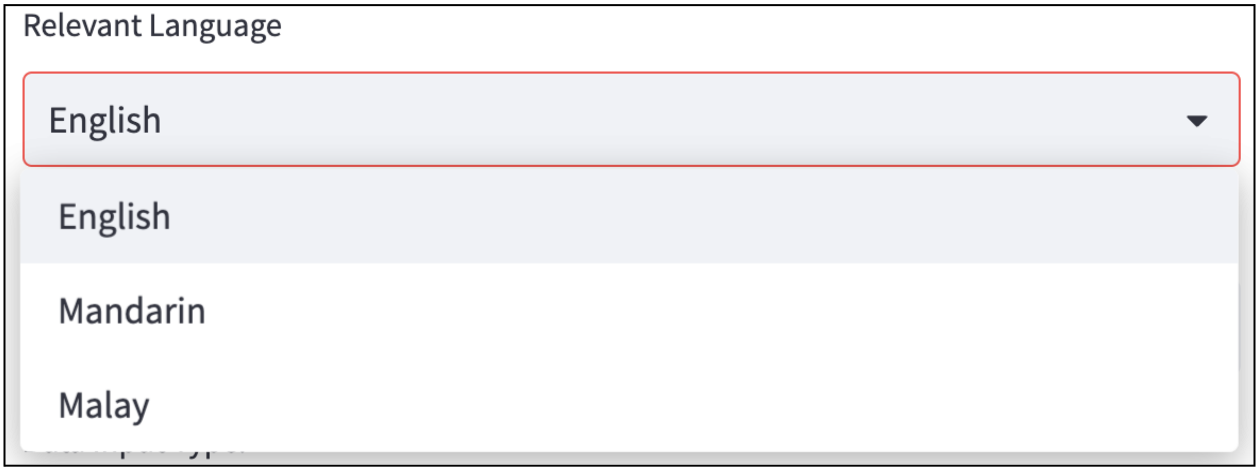
Sentence-Pair Model

For the machine learning model, Langsplain utilises the sentence-pairs paradigm. Sentence-pairs models take in two associated pieces of text (“Text A” and “Text B”) to output a prediction, and can be used for a wide range of tasks such as translation as well as question and answering. For our purposes, “Text A” was the full text while “Text B” was the named entity in question that appears in the full text. Our sentence-pairs model was trained to classify the emotion and/or stance based on these two inputs.

Key Features of Langsplain

Feature 1: Language Selection

Langsplain is available in three different languages, English, Mandarin and Malay. From the dropdown menu, the user can choose English, Mandarin, or Malay.
Note: The Double-headed classifier developed by the other NUS group is only available to predict texts with NERs in English and Mandarin, and does not support Malay.

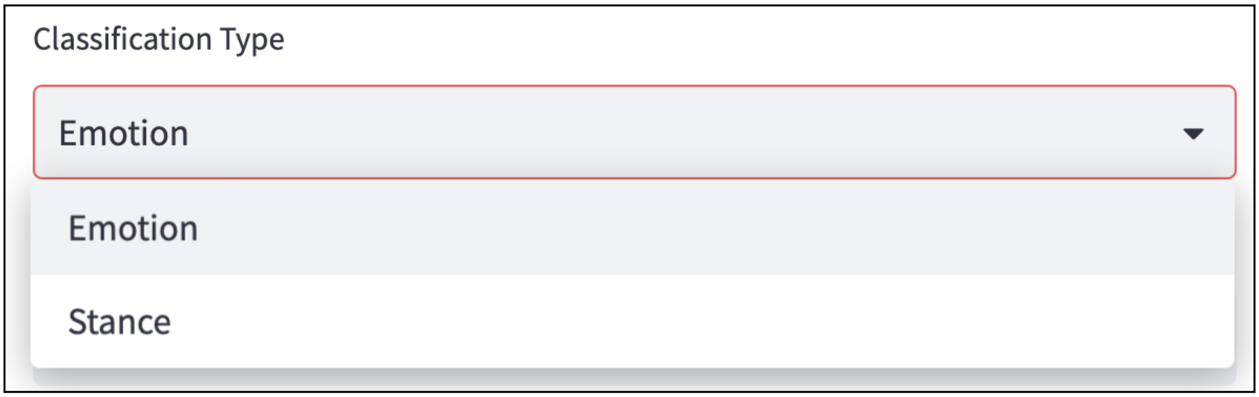


Relevant Language

English

Feature 2: Classification Type

Langsplain supports both Emotion and Stance classifications. From the dropdown menu, the user can choose either Stance or Emotion classifications to be applied on the input text and NER for prediction explanations.

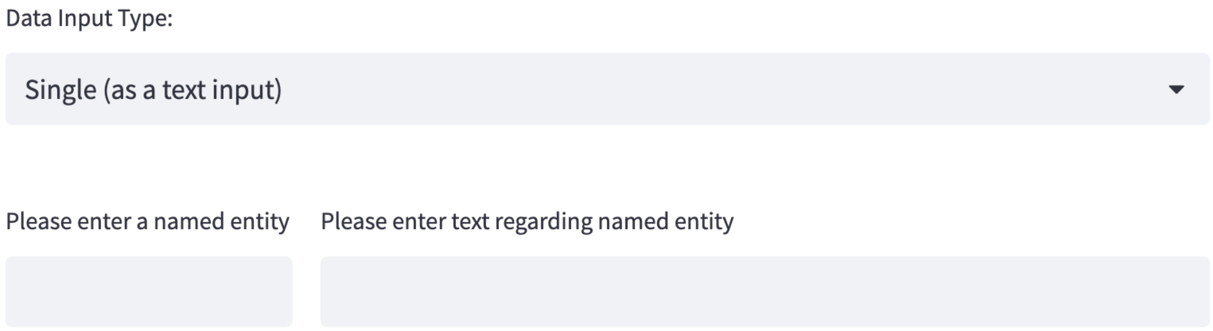


Classification Type

Emotion

Feature 3: Data Input Type

Langsplain offers users the ability to select the desired format for inputting data entries for prediction explanation. For single data entries, the user can input text and NER directly on the application. For multiple data entries, the user can input a CSV file containing multiple entries.



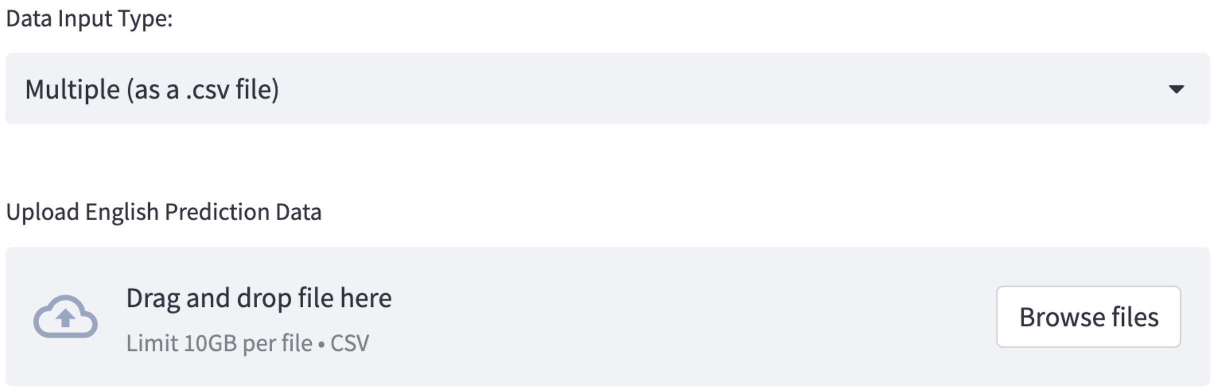
Data Input Type:

Single (as a text input)

Please enter a named entity

Please enter text regarding named entity

Single Input Data Entry



Data Input Type:

Multiple (as a .csv file)

Upload English Prediction Data

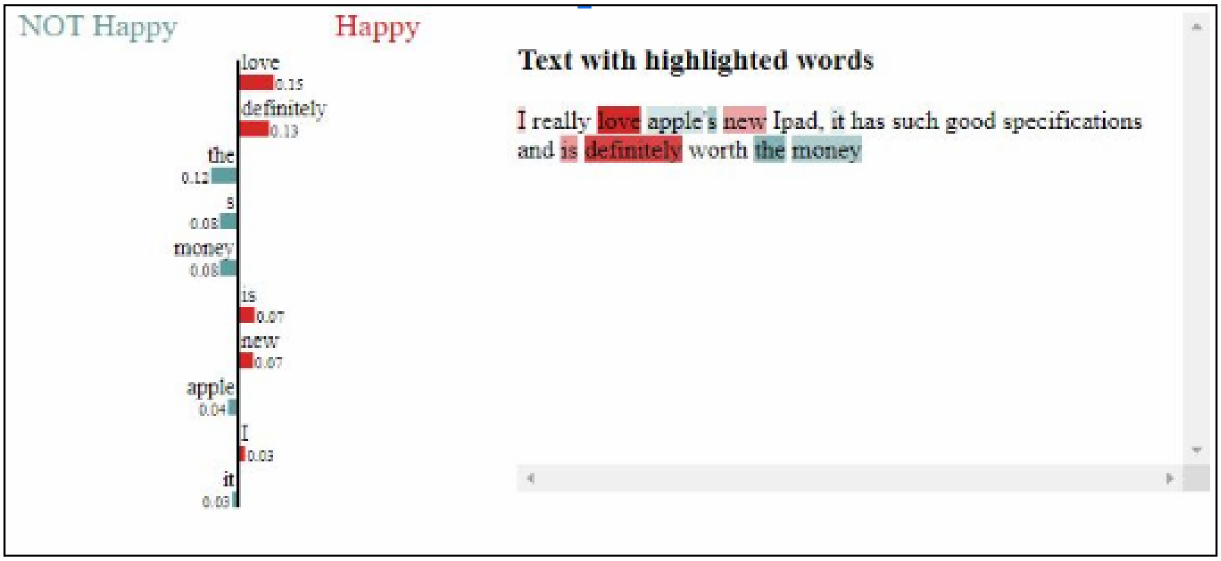
Drag and drop file here

Limit 10GB per file • CSV

Multiple Input Data Entries

Feature 4: Feature Importance and Local Explanation

Langsplain utilises LIME which helps the user to understand the prediction by highlighting the most important words/tokens in driving the prediction for the instance of interest.



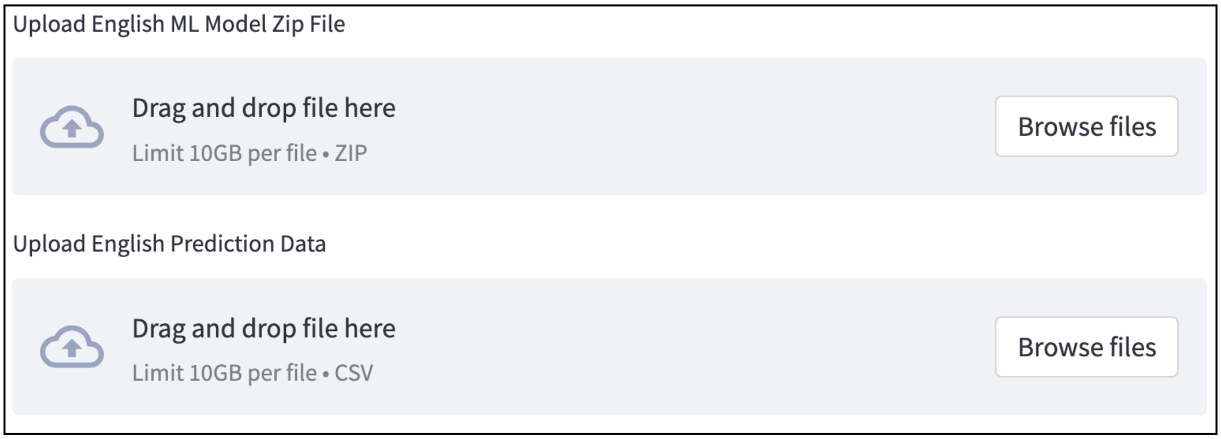
Text with highlighted words

I really love apple's new Ipad, it has such good specifications and is definitely worth the money

Regarding local explanations, LIME does not require any background context of the inputs. It explains the predictions for a specific instance or data point. Langsplain only requires text and one NER as input for emotion and stance prediction explanation.

Feature 5: Large Input Capacity

Langsplain supports files up to a limit of 10GB. This allows for large ML text classification models and data entries to be loaded onto the application. While langsplain can support large numbers of data entries, we recommend only loading 20 data entries for each prediction explanation. Any more than this could result in long loading times.



Upload English ML Model Zip File

Drag and drop file here
Limit 10GB per file • ZIP

Upload English Prediction Data

Drag and drop file here
Limit 10GB per file • CSV

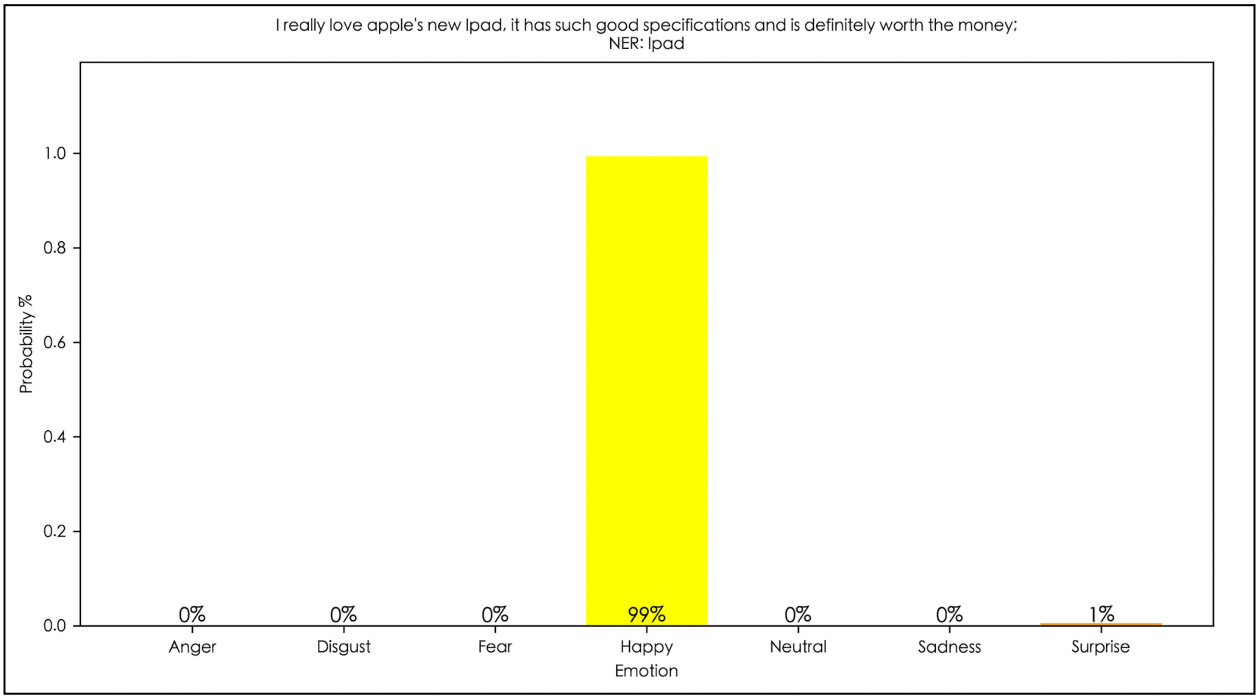
Feature 6: Graphical Results and Downloadability

After Langsplain has completed the generation of explanation results from the input CSV file, a download button will be available to save all the results as images in JPG format in the user's local computer.

There will be 3 types of files that will be output to the user in a ZIP file.

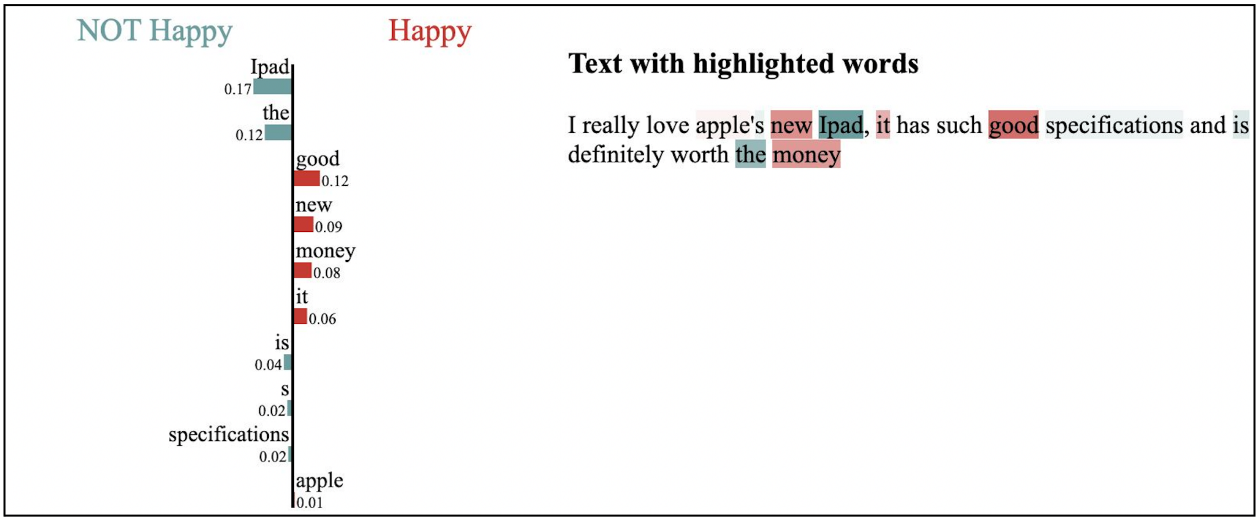
Output 1: Graphical Prediction Probabilites for each Data Entry (JPG)

The graphical prediction probabilities will provide users a graph plot of the probability distribution for each data entry. This will allow the users to know how confident the model predicts the data entry for each emotion/stance label.



Output 2: Graphical Explanation of tokens for each Data Entry (JPG)

The graphical explanation of tokens will provide users a graph plot of the probability distribution of each token for each data entry associated with each emotion or stance label. This will allow the users to know how the model used each token to predict.

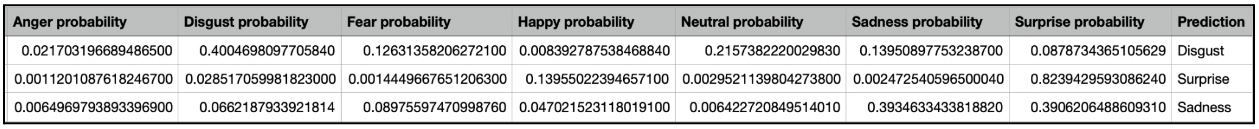


Text with highlighted words

I really love apple's new Ipad, it has such good specifications and is definitely worth the money

Output 3: Class Prediction Probability Distribution File (CSV)

The class prediction probability distribution file will provide users an overall summary of the explanation results. In this probability distribution file, the user will be able to view the exact probability that the model has predicted for each emotion or stance (e.g. “Sadness: 0.06543896437, Joy: 0.856432549830”), which will be saved to a CSV file.



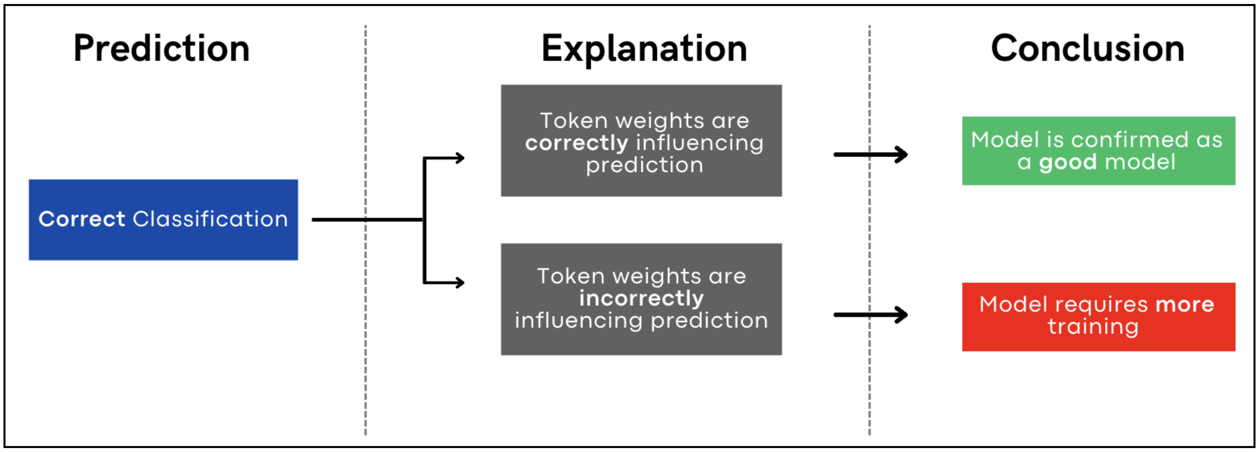
Anger probability	Disgust probability	Fear probability	Happy probability	Neutral probability	Sadness probability	Surprise probability	Prediction
0.021703196689486500	0.4004698097705840	0.12631358206272100	0.008392787538468840	0.2157382220029830	0.13950897753238700	0.0878734365105629	Disgust
0.0011201087618246700	0.028517059981823000	0.0014449667651206300	0.13955022394657100	0.0029521139804273800	0.002472540596500040	0.8239429593086240	Surprise
0.0064969793893396900	0.0662187933921814	0.08975597470998760	0.047021523118019100	0.006422720849514010	0.3934633433818820	0.3906206488609310	Sadness

Use-Case Scenario

As a ML text classification model explainer, Langsplain is able to help validate the ML model prediction process. This is important as this will help build trust in the model and facilitate confident data-driven decision-making. Also, in the event that MHA is required to explain to stakeholders (such as regulators) why such decisions were made by ML models, they are able to use these results to justify their decisions.

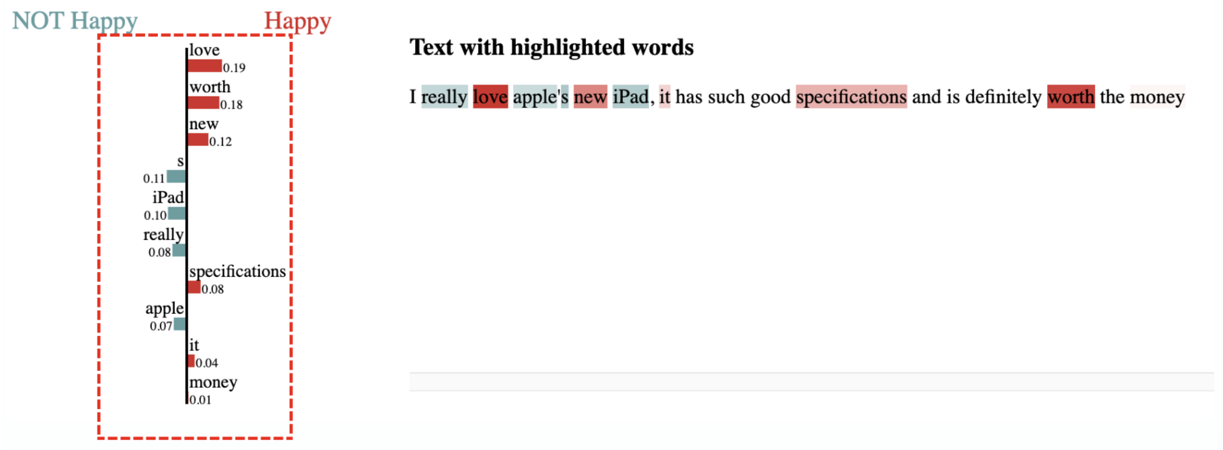
Scenario 1: ML Model Predicts a Correct Classification

In the event that the ML model classification is correct, there is still a need to validate the results by understanding how each token (word) in the data entry is influencing the prediction.


Prediction
Explanation
Conclusion
Correct Classification
Token weights are correctly influencing prediction
Token weights are incorrectly influencing prediction
Model is confirmed as a good model
Model requires more training

Scenario Flowchart of Correct Classification

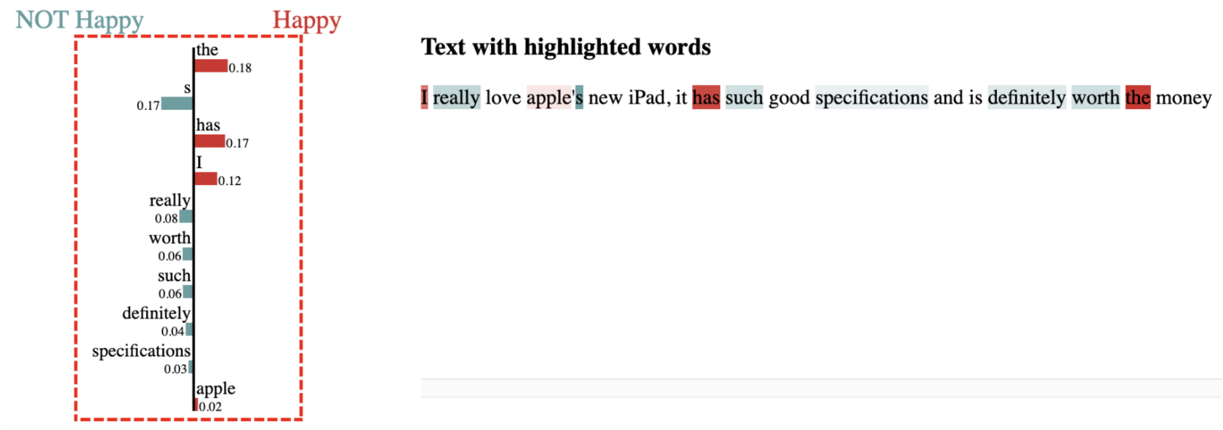
There are two outcomes for a correctly classified ML model. Either the token weights are accurately influencing the prediction, or they are inaccurately influencing it. A correct influence indicates a good model, while an incorrect influence suggests that the model needs further training.



Text with highlighted words

I really love apple's new iPad, it has such good specifications and is definitely worth the money

Correct Token Weight Influencing Prediction - Good Model



Text with highlighted words

I really love apple's new iPad, it has such good specifications and is definitely worth the money

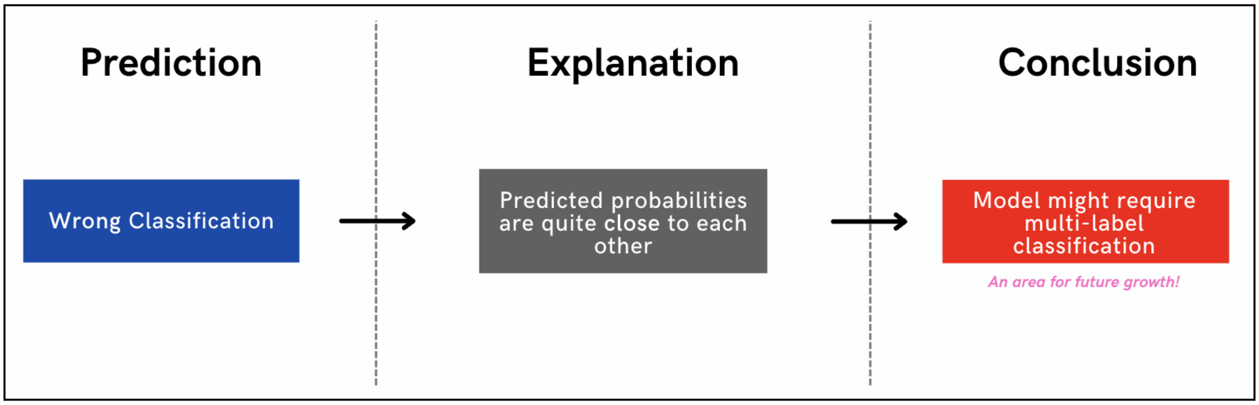
Incorrect Token Weight Influencing Prediction - Model Requires More Training

This is an example of how to use Langsplain to identify if the ML text classification model is truly a good model or if it requires more training. While both examples classify the prediction as Happiness, the first example shows that the model uses accurate tokens to classify Happiness while the second example shows that the model uses inaccurate tokens to classify Happiness.

If Langsplain highlights a particular word token as being highly important for a model's prediction, you can examine that feature more closely to see if it is relevant to the problem at hand. Conversely if it highlights a word token that is known to be biased or discriminatory, you can investigate ways to address that bias in the ML model.

Scenario 2: Wrong Classification

In the event that the ML model classification is wrong, there is still a need to validate the results by understanding whether there is still useful information to be gained from this classification. Or if the definition of the classification problem is unsuitable.



Prediction

Explanation

Conclusion

Wrong Classification

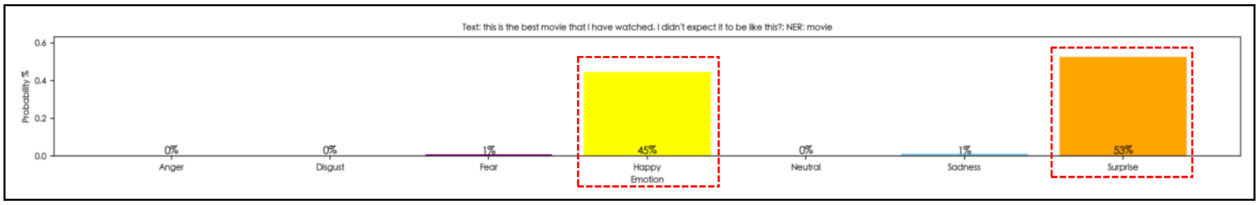
Predicted probabilities are quite close to each other

Model might require multi-label classification

An area for future growth!

Scenario Flowchart of Wrong Classification

For example, let us consider a text input of “This is the best movie that I have watched, I didn’t expect it to be like this?” with a NER tag of “movie”. Despite the correct labeled emotion being “Happy”, the ML text classifier predicts the emotion as “Surprise”.



With Langsplain’s explanatory capabilities, we can delve into the model’s inner workings and determine that for this particular data entry, both “Happy” and “Surprise” emotions received high scores. However, “Surprise” was slightly favoured over “Happy”, with probabilities of 53% and 45% respectively. Looking at the data entry, it also makes sense that the text could be classified as either “Happy” or “Surprise” or even both.

With this information where the predicted probabilities are quite close to each other, we can identify areas for future growth where a multi-label classification model (where data entries can be predicted with multiple emotions) might be more applicable.

Hands-on Guide

Setting up

1. Ensure you have the Langsplain folder saved on your computer. If you do not already have it, clone the github repo using the following steps

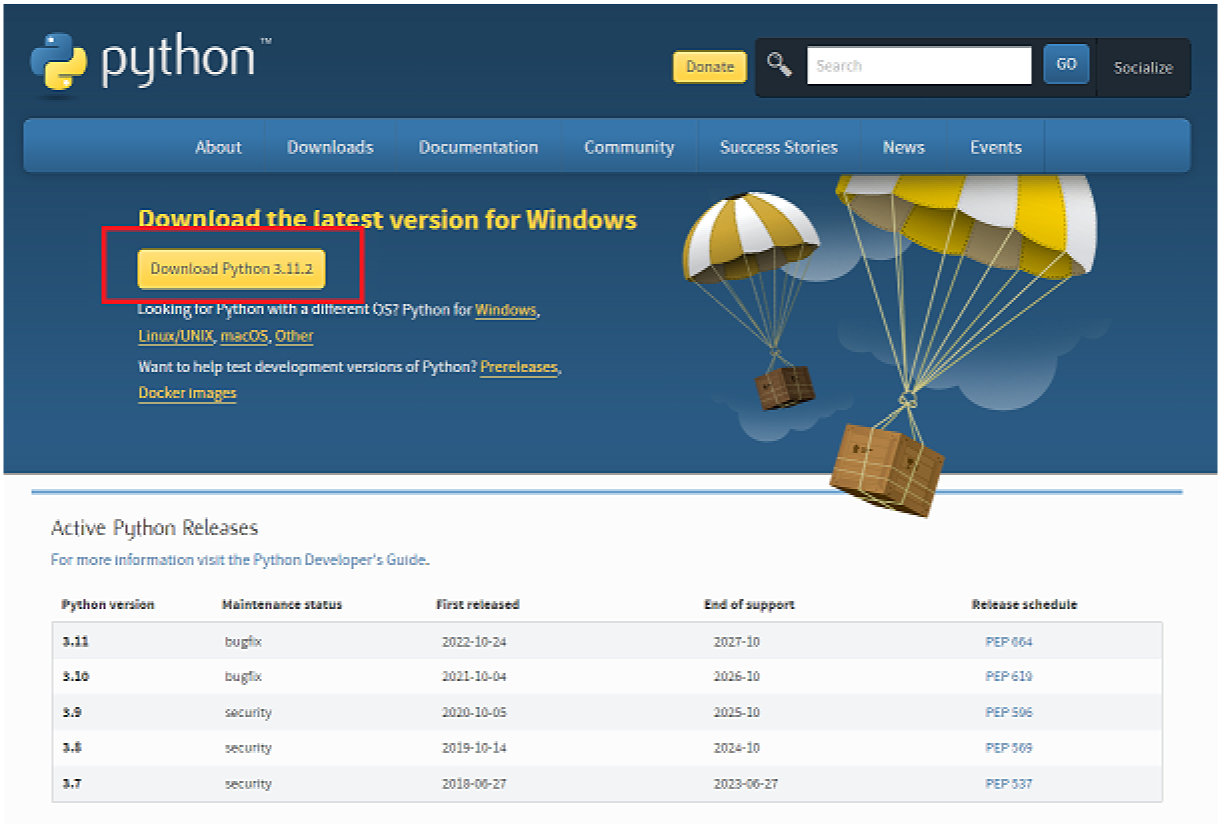
    a. Go to https://github.com/darellelogram/langsplain

    b. Click on the green button saying “<> Code” and select Download ZIP

    c. Extract the ZIP file to a folder of your choice

2. Ensure you have Python 3 installed on your computer. If you do not already have Python 3 installed, use the following link to install it: https://www.python.org/downloads/

    a. Install Python for Windows. You can download the latest stable version.



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

Active Python Releases

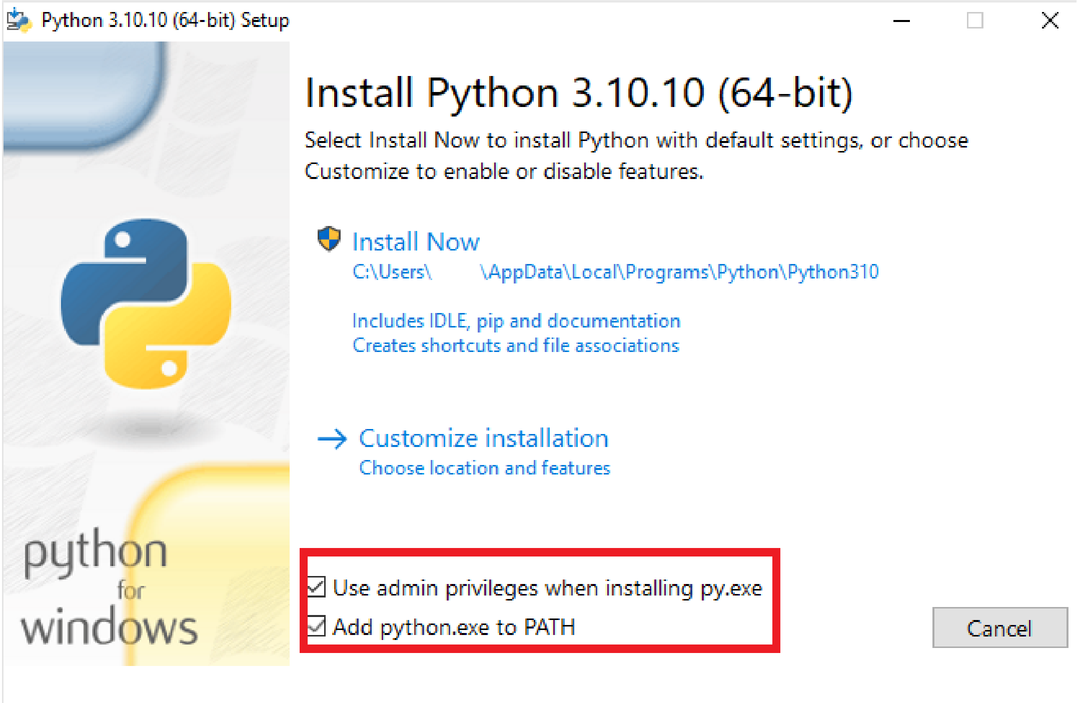
For more information visit the Python Developer's Guide.

Python version	Maintenance status	First released	End of support	Release schedule
3.11	bugfix	2022-10-24	2027-10	PEP 664
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569
3.7	security	2018-06-27	2023-06-27	PEP 537

    b. After installer has been downloaded, double-click the `.exe` file

c. Select the Install launcher for all users checkbox, which enables all users of the computer to access the Python launcher application

d. Select the Add python.exe to PATH, which enables users to launch Python from the command line.



Python 3.10.10 (64-bit) Setup

Install Python 3.10.10 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

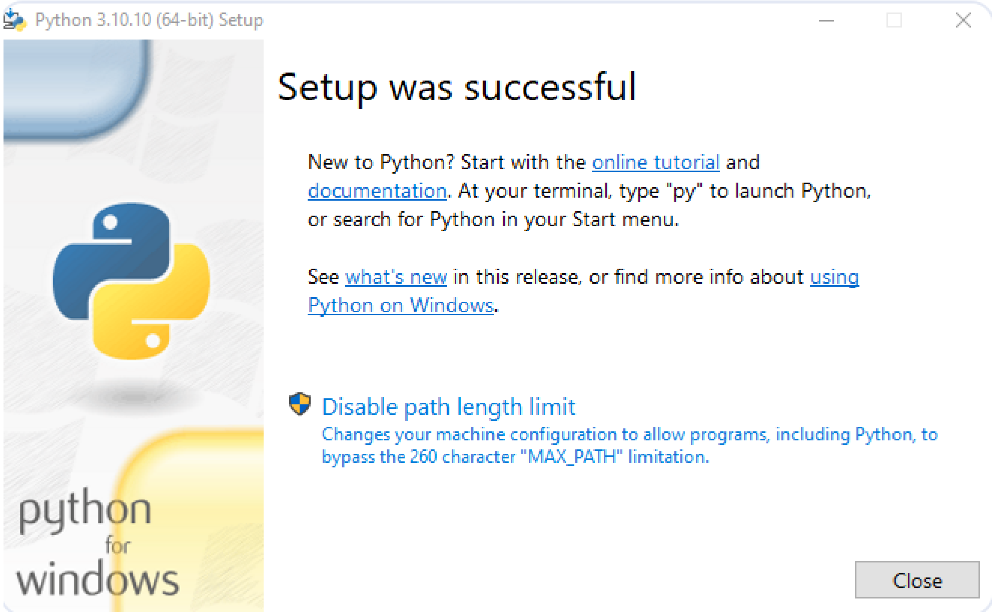
Install Now
C:\Users\ \AppData\Local\Programs\Python\Python310

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Use admin privileges when installing py.exe
☑ Add python.exe to PATH

e. You should see this installation complete screen.



Python 3.10.10 (64-bit) Setup

Setup was successful

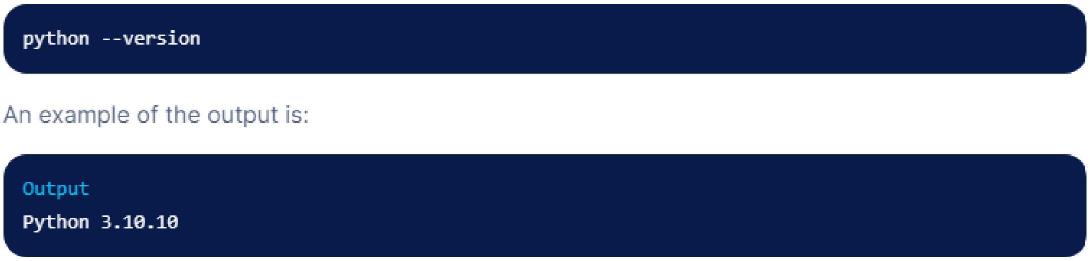
New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.

See what's new in this release, or find more info about using Python on Windows.

Disable path length limit
Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

f. Verify you have installed python by going to **Start** and enter `cmd`. Click **Command Prompt**

g. Enter the following command in the command prompt: (python version may be different)

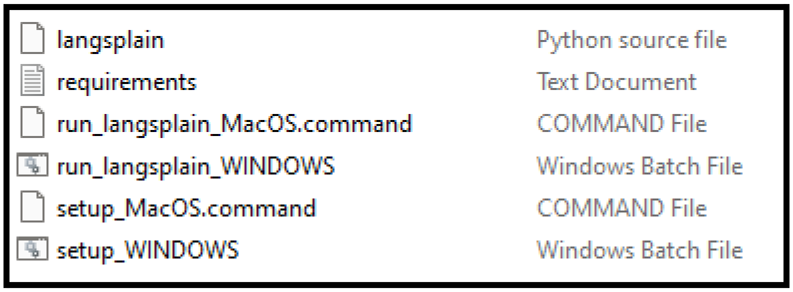


```
python --version
```

An example of the output is:

```
Output
Python 3.10.10
```

3. Once you have ensured that Python is installed, go to the folder that you downloaded from GitHub. You should see the following files in the folder:
    a. `langsplain.py`
    b. `requirements.txt`
    c. `setup_WINDOWS.bat`
    d. `setup_MacOS.command`
    e. `run_langsplain_WINDOWS.bat`
    f. `run_langsplain_MacOS.command`



langsplain	Python source file
requirements	Text Document
run_langsplain_MacOS.command	COMMAND File
run_langsplain_WINDOWS	Windows Batch File
setup_MacOS.command	COMMAND File
setup_WINDOWS	Windows Batch File

Windows Folder Structure

You should also have a `.streamlit` folder with a `config.toml` file in it.

4. Double click `setup_WINDOWS.bat` for Windows users or `setup_MacOS.command` for MacOS users. This should install all the required Python packages needed to run Langsplain. It might take several minutes. For windows, you may have to click on “More info” and “Run anyway”. Refer to **Common Issues and Error Messages** under **Troubleshooting**

Launching the Langsplain App

Option 1: Using File Explorer

1. Open the folder that you downloaded from GitHub.

2. Double click `run_langsplain_WINDOWS.bat` for Windows users or `run_langsplain_MacOS.command` for MacOS users

3. This should launch the Langsplain web app in your Internet browser with the address `https://localhost:8501/`
   A code interface should also pop-up on your computer. Do not close this window.
   Note: You do NOT need an Internet connection to do this.

Option 2: Using Command Line

1. Open Command Line using Command Prompt for Windows or Terminal for MacOS

2. Navigate to the folder that you downloaded from GitHub on the Command Line

3. Ensure that the required Python libraries are installed using this command

   ```
   pip install -r requirements.txt
   ```

4. Run these two lines of code

   ```
   python -m spacy download zh_core_web_sm
   streamlit run langsplain.py
   ```

5. This should launch the Langsplain web app in your Internet browser with the address `https://localhost:8501/`
   A code interface should also pop-up on your computer. Do not close this window.
   Note: You do NOT need an Internet connection to do this.

Running Explanations Using Langsplain

Required Files

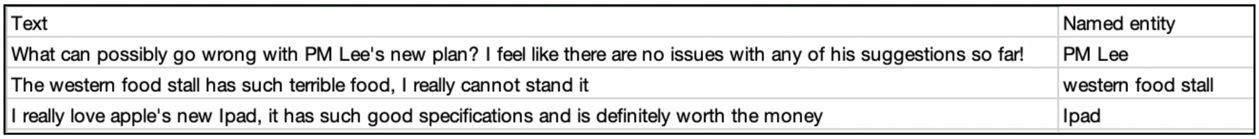
The files that you will require are:

1. A **zip file** containing the machine learning model. For now, Langsplain is only able to take in models created by the NUS Capstone groups.
   The ML text classification model zip file has to be uploaded exactly as they were provided to the stakeholder.

2. A **CSV file** (i.e. a file with an extension .csv) containing two columns:

   a. “Text”
      This is the full text that you want to use to explain model predictions on it

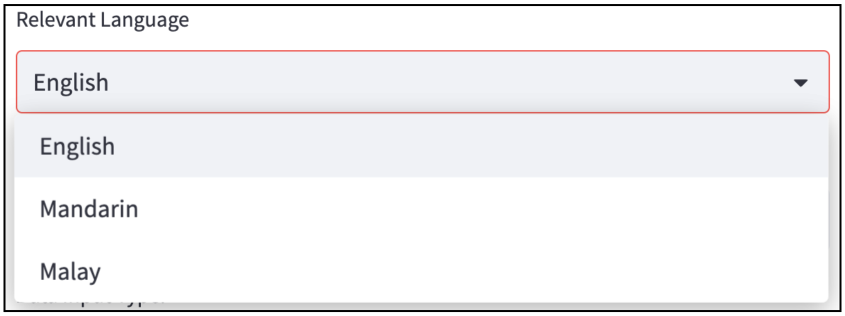
   b. “Named entity”
      This is the named entity within the text regarding which is the target of prediction of either emotion or stance. This can be a person, a place, an organisation, a geopolitical entity or even a sociocultural group, etc. Do ensure that the named entity actually appears inside the full text.



Text	Named entity
What can possibly go wrong with PM Lee's new plan? I feel like there are no issues with any of his suggestions so far!	PM Lee
The western food stall has such terrible food, I really cannot stand it	western food stall
I really love apple's new Ipad, it has such good specifications and is definitely worth the money	Ipad

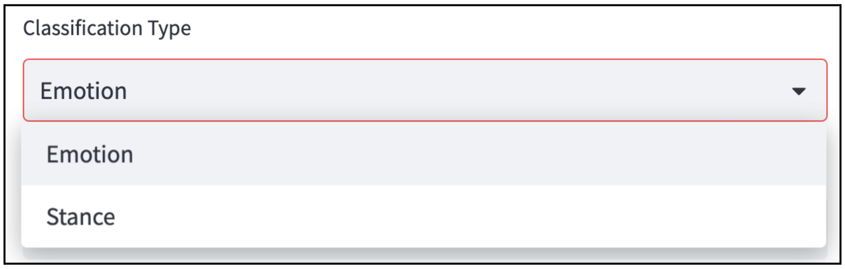
Producing Explanations

1. Select the language


Relevant Language

English

2. Select the type of classification - either “Emotion” or “Stance”

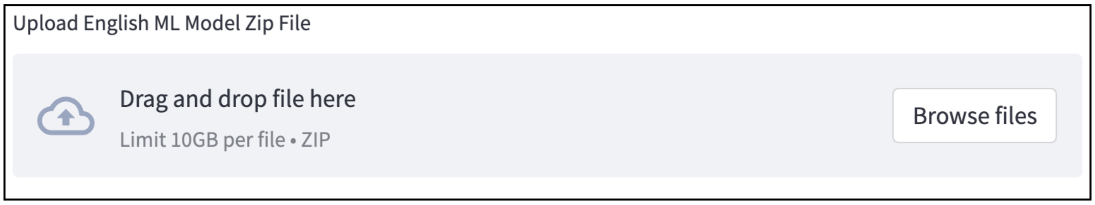

Classification Type

Emotion

3. Upload the machine learning model using either of the following methods:

   a. Drag and drop the zip file containing the machine learning model you would like to use to the upload box saying “Upload <Language> ML Model Zip File”

   b. Click on “Browse files” on the right of the upload box and select the zip file containing the machine learning model you would like to use


Upload English ML Model Zip File

Drag and drop file here

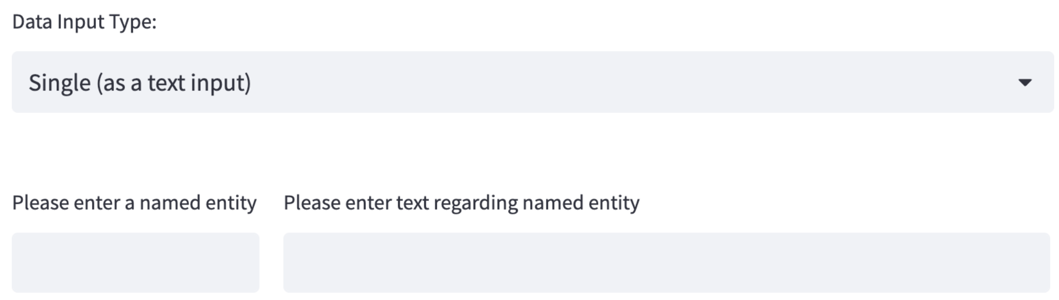
Limit 10GB per file • ZIP

4. Input the data to predict and explain by either keying in the data yourself, or by uploading a CSV file. **Do ensure that the data is in CSV format.** If the file is in XLSX format, remember to convert it first. We advise a 20-row limit for CSV files. Beyond this limit, Langsplain will take much longer to run.

a. Single Input

   i. Select “Single” under “Data Input Type”.

   ii. Type the full text into the box and the named entity.



Data Input Type:

Single (as a text input)

Please enter a named entity

Please enter text regarding named entity

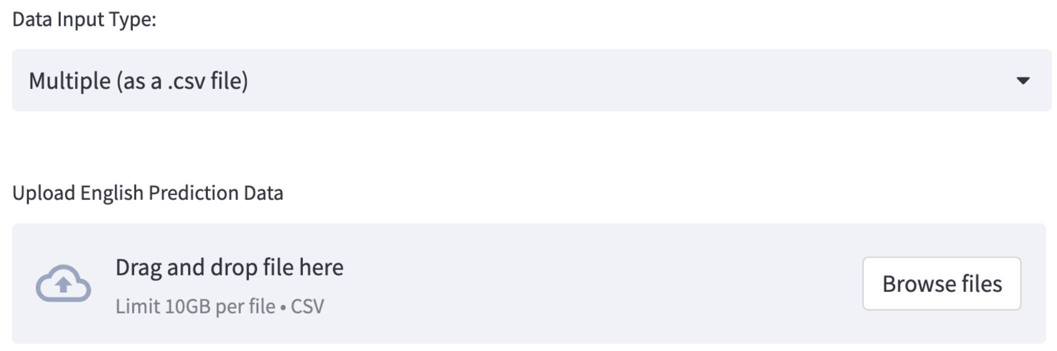
Single Input Data Entry

b. Multiple Input (CSV file)

   i. Select “Multiple” under “Data Input Type”.

   ii. Drag and drop the CSV file containing the data you wish to use to the upload box saying “Upload <Language> Prediction Data”

   iii. Click on “Browse files” and select the CSV file containing the data you wish to use



Data Input Type:

Multiple (as a .csv file)

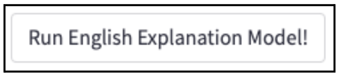
Upload English Prediction Data

Drag and drop file here

Limit 10GB per file • CSV

Multiple Input Data Entries

5. Once all the above steps are complete, click “Run <Language> Explanation Model!” to produce the explanations

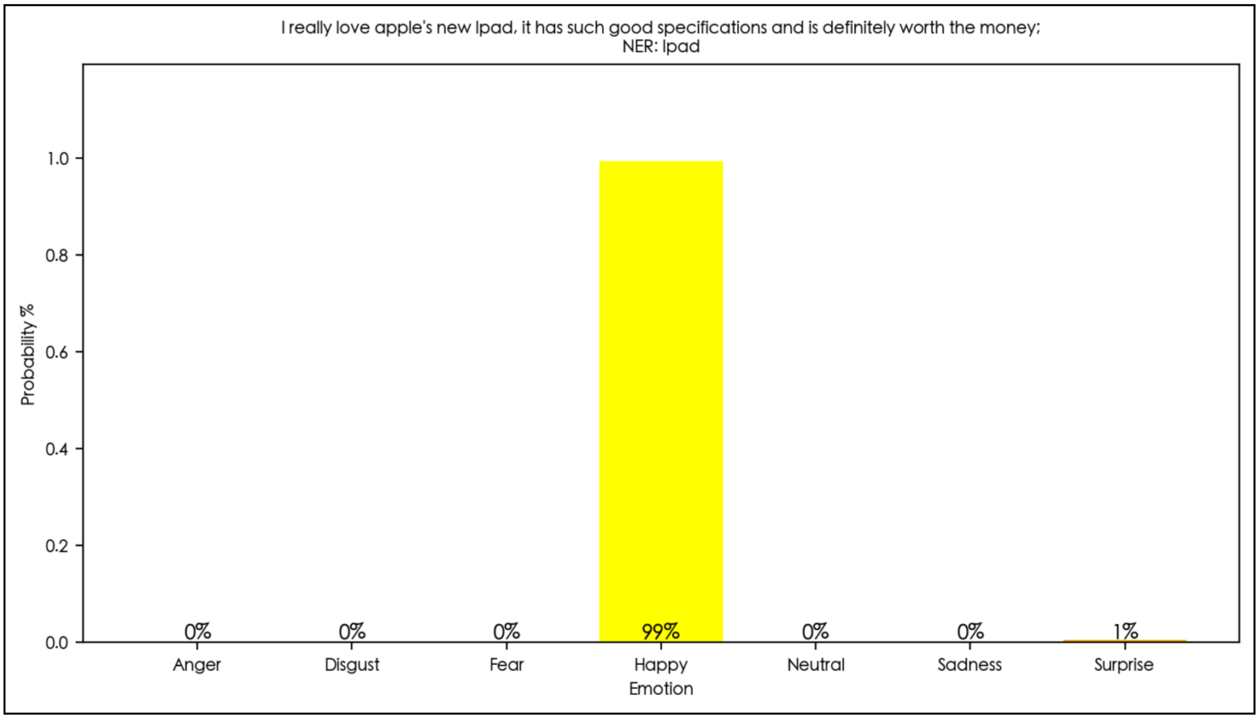


Interpreting Output

Predictions

For each input (a pair of text and named entity), Langsplain will output a list of emotions and stances, along with the prediction probabilities for each one. These probabilities represent how likely each emotion and or stance is being expressed in the text with respect to the named entity.

For example, in the example below, the model is 99% confident that the emotion expressed in “I really love apple's new Ipad, it has such good specifications and is definitely worth the money” towards the named entity “Ipad” is happiness. It also gives a 1% probability that the emotion is surprise. The other emotions are given a probability of 0%, meaning that it is highly unlikely that the text expresses any of those emotions towards “Ipad”.



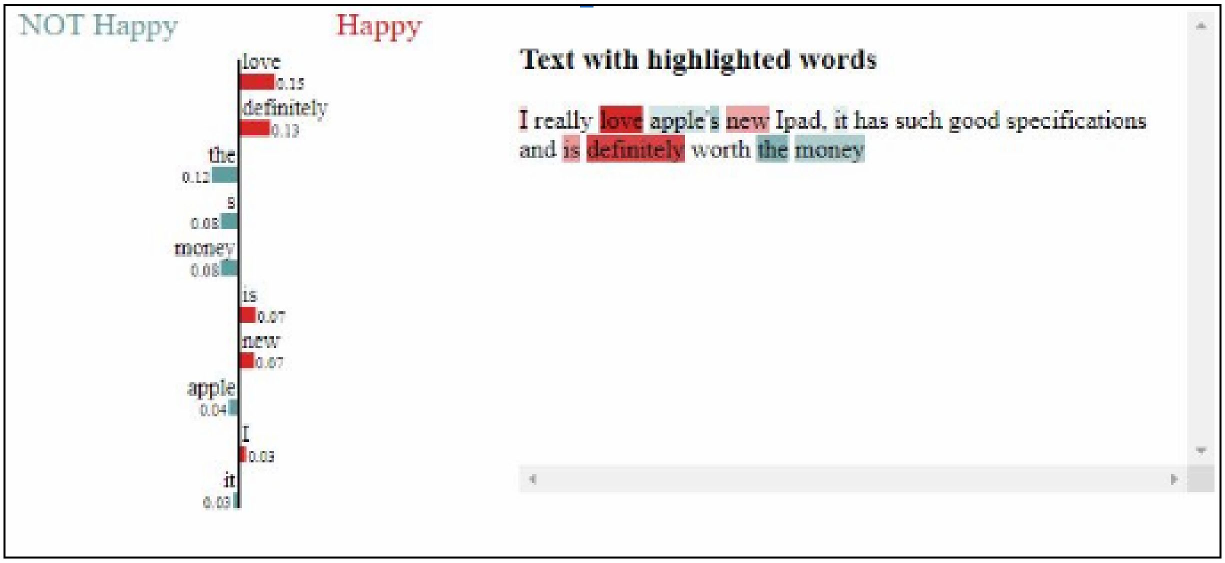
Explanations

The previous portion describes what the machine learning model is doing. After that is where the explainable AI comes in.

In order to help you understand which words in the text are contributing to each emotion/stance prediction, Langsplain highlights these words with a colour corresponding to the intensity of the emotion/stance. Words that contribute more strongly to the prediction will be highlighted more intensely.

Finally, Langsplain gives you a score on the left side to quantify the importance of each word in the prediction. This score is based on how much the word contributes to the overall prediction for that emotion/stance. So if a word has a high score, it means that it is particularly important in helping to determine that emotion/stance for that named entity.

In the example below, “love” has the highest score of 0.15 and is highlighted the deepest red, showing that it is the most important word in predicting the emotion “Happy”.



Text with highlighted words

I really love apple's new Ipad, it has such good specifications and is definitely worth the money

System Requirements for Langsplain

Please check the following as Langsplain has several system requirements. Firstly, ensure that Python 3 is installed on your computer. The instructions in the Setting Up portion of the Hands-On Guide will ensure that these Python packages have been installed locally. A `requirements.txt` file will be included in the Github repository.

streamlit	numpy
simpletransformers	scipy
lime	pandas
html2image	re
webdriver_manager	os
spacy	zipfile
torch	selenium
zh_core_web_sm*	

Required Python Packages

Langsplain also requires spacy's Chinese package `zh_core_web_sm`, which will be installed by default with our `setup_WINDOWS.bat` for Windows users or `setup_MacOS.command` for MacOS users.

Limitations

Limitations of Langsplain Application

Limited Language Support
Langsplain currently only supports three languages - English, Mandarin, and Bahasa Melayu. This means that it may not be suitable for users who need to analyze text in other languages.

Limited Model Support
Langsplain can only explain the predictions made by specific machine learning models, including pre-trained models from the SimpleTransformers and Transformers libraries, as well as a PyTorch-based Double-headed classifier developed and optimized by the sister group. This means that it may not be suitable for users who need to analyze predictions made by other models.

Limited NER Support
Langsplain requires input text to include at least one named entity recognition (NER) tag, which may not be present in all types of text. This means that it may not be suitable for users who need to analyze text that does not include named entities. When explaining predictions on Malay language text, Langsplain is also likely to have poorer performance on texts that contain multiple named entities due to limitations in its Malay training data.

Limited Explanation Granularity
Langsplain provides local explanations that highlight the most important features for a specific instance or data point. While this can be useful for understanding the reasoning behind a single prediction, it may not capture the overall behavior of the machine learning model across all instances or data points.

It is important to note that while Langsplain has some limitations, it still provides a useful and accessible tool for users who need to analyze predictions made by specific machine learning models in supported languages. As with any tool or application, it is important to understand its limitations and use it appropriately for the specific task at hand.

Limitations of LIME

To reiterate, Langsplain is built using LIME as the underlying medium. Here are some general limitations of using LIME as an explainer:

Local Explanations May Not Capture Global Patterns
LIME generates local explanations that highlight the most important features for a specific instance or data point. While this can be useful for understanding the reasoning behind a single prediction, it may not capture the overall behavior of the machine learning model across all instances or data points.

LIME Explanations May Not Be Consistent: LIME is Non-Deterministic
This means that for feeding it the same input twice might result in different outputs. Since LIME generates local explanations based on a perturbation of the input features, the explanations may vary depending on the specific perturbation applied. Different perturbations may result in different feature importance scores and explanations, which can make it difficult to compare and interpret the results.

Max Number of Tokens at 200 Samples
Currently our explainer uses the recommended default of 200 samples to approximate the model's behavior. This means that if the input text contains more than 200 tokens, the effect of some tokens may not be fully explained by LIME. To improve the accuracy of LIME's approximation, users can customize the number of samples used by LIME, at the expense of runtime. It is recommended to use a number of samples greater than some multiple of the number of tokens in the input text, such as 5 times the number of tokens, to ensure that each token has its effect quantified multiple times.

It is important to note that while LIME has some limitations, it is still a useful and widely-used technique for explaining the predictions made by machine learning models. Other techniques, such as SHAP (SHapley Additive exPlanations) and Integrated Gradients, may be explored as a means of addressing some of these limitations by future groups.

Troubleshooting

Common Issues and Error Messages

- If the input file is too large or the system does not have enough memory, this error message may occur.
- If the uploaded model is of type or language different from the selected input.
- If the uploaded files contain different content than expected, such as uploading Chinese data to be explained when an English model is selected.
- “Windows protected your PC!”: If you encounter this error while running any of the setup or run_langsplain programs, just select “More Info” then “Run anyway”

Troubleshooting Steps and Solutions:

- Check the input language: Make sure that the input language is supported by Langsplain (English, Mandarin, or Bahasa Melayu).
- Check the input file format: Make sure that the input file is in a supported format, such as CSV or zip.
- Check the input model: Make sure that the model you have uploaded is of the type (Stance/Emotion) and language you have selected.
- Try a smaller input file: If the input file is too large, try reducing its size or splitting it into smaller files.
- Be patient: Make sure files are fully loaded with the message indicating that the application is ready to accept the next inputs or instructions before proceeding.

Appendix

Definition of Key Terms and Concepts

- Emotion Explainer: A feature of Langsplain that predicts the emotional state conveyed by a given text input. This can be useful for tasks such as sentiment analysis or understanding the emotional impact of a piece of writing.
- Stance Explainer: A feature of Langsplain that predicts the writer's stance on a given topic, such as whether they are in favour of or opposed to a particular policy or idea.
- Feature Importance: A measure of the contribution of each input feature to the prediction made by a machine learning model. In Langsplain, feature importance is used to highlight the most important words or tokens in driving the prediction for a given input instance.

Resources for Further Reading or Support

- LIME Github repository: https://github.com/marcotcr/lime
- Transformers library documentation: https://huggingface.co/transformers/