A logo with a yellow and blue design

AI-generated content may be incorrect.

**National College of Ireland**

**Project Submission Sheet**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name:** | **DHEERAJ GOPALKRISHNA**  **AHMED OZAIR CHAKARI** | | |
| **Student ID:** | **24135739 | 24212113** | | |
| **Programme:** | **MSc in Artificial Intelligence** | **Year:** | **2025** |
| **Module:** | **Engineering and Evaluating AI Systems (H9EEAI)** | | |
| **Lecturer:** | **Jaswinder Singh** | | |
| **Submission Due Date:** | **15/07/2025** | | |
| **Project Title:** | **Continuous Assessment** | | |
| **Word Count:** |  | | |

**I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.**

**ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.**

|  |  |
| --- | --- |
| **Signature:** | Dheeraj Gopalkrishna | AHMED OZAIR CHAKARI |
| **Date:** | 15/07/2025 |

**PLEASE READ THE FOLLOWING INSTRUCTIONS:**

1. Please attach a completed copy of this sheet to each project (including multiple copies).

2. Projects should be submitted to your Programme Coordinator.

3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.

4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**

5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

|  |  |
| --- | --- |
| **Office Use Only** | |
| Signature: |  |
| Date: |  |
| Penalty Applied (if applicable): |  |

**AI Acknowledgement Supplement**

**Data Analytics for Artificial Intelligence**

**MSc in Artificial Intelligence**

|  |  |  |
| --- | --- | --- |
| **Your Name/Student Number** | **Course** | **Date** |
| Dheeraj Gopalkrishna / 24135739 | MSc in Artificial Intelligence | 14/07/2025 |
| AHMED OZAIR CHAKARI / 24212113 | MSc in Artificial Intelligence | 14/07/2025 |

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click [here](https://libguides.ncirl.ie/useofaiinteachingandlearning/studentguide).

**AI Acknowledgment**

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

|  |  |  |
| --- | --- | --- |
| **Tool Name** | **Brief Description** | **Link to tool** |
| **N/A** | N/A | N/A |
|  |  |  |

**Description of AI Usage**

This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

|  |  |
| --- | --- |
| **[Insert Tool Name]** | |
| [Insert Description of use]**- N/A** | |
| [Insert Sample prompt]**– N/A** | [Insert Sample response] – N/A |

**Evidence of AI Usage**

This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

**Additional Evidence:**

[Place evidence here] – N/A

**Additional Evidence:**

[Place evidence here] – N/A

**Introduction**

The project implements a Chained Multi-Output Architecture for multiple label email classification on top of the Single Output Architecture that was provided. The new architecture will predict messages tone and resolution thus expanding the single architecture functionality that predicted only intent. The Multi Output model is designed and implemented to ensure modularity and scalability by choosing a pipeline chain.

**Architecture**

**EVALUATION MODULE**

Precision, Recall, F1, Support

* **Model: Random Forest**
* **Train on Input Data**
* **Predict/Classify**

**MODEL BLOCK**

Classification Result

**E + Y2**

**E**

**PIPELINE**

**MODEL BLOCK Y4**

**MODEL BLOCK Y2**

**MODEL BLOCK Y3**

**FEATURE COMBINER**

**EMBEDDINGS**

**+**

**MODEL OUTPUT**

**DATA PREPROCESSOR**

Generate clean Data (DataFrames)

* **Clean Duplicate Texts**
* **Remove Word Noise**
* **Translate Text to English**

**EMBEDDING**

Generate Embeddings using TF-IDF Vectorization

**DATA LOADER**

Loads CSV Files and returns unified DataFrames

**AppGallery.csv | Purchasing.csv**

**E**

**Y3 TONE**

**Y2 INTENT**

**E + Y2 + Y3**

**Y4 RESOLUTION**

This architecture design will classify emails into multiple inter-dependent categories such as Intent (Type 2), Tone (Type 3) and Resolution (Type 4). A sequential pipeline is created where the prediction of each model is fed into the next model. The system begins with the Data Loader block where the customer messages from CSV files are loaded. This is followed by a Data Preprocessor block that cleans the data, normalizes it by removing duplicate texts, word noise and then translates the cleaned text into English language. The Embedding block transforms the cleaned text into numerical embeddings using TF-IDF technique, a popular one in NLP. A pipeline is created which manages the chaining of Random Forest Classification models and ensures proper data flow. The first model is fed only with embedding data to predict Intent (Type 2). The output (Y2) thus generated is then sent to Feature Combiner block which combines embeddings and the Y2, which is fed into the second model to predict Tone (Type 3). The result (Y3) generated is then again combined with embeddings, Y2 and Y3 and fed into the third model to predict Resolution (Type 4). Between each model stage, the Feature Combiner block will merge the embeddings with the previous model outputs to make sure the models have the context from previous outcomes. Finally, the Evaluation Module will compute performance metrics such as Precision, Recall, F1 Score and Support.

**Modifiability**

Since the architecture consists of multiple modules such as data loading, preprocessing, embedding, and also a pipeline, the architecture is modular and hence easily modifiable. It is easy to replace embedding techniques such as TF0IDF or experiment on different machine learning models without affecting other blocks. Additional layers can also be added to the pipeline with minimal modification.

**Performance**

Chained model architecture is easy to implement, and the accuracy can be greatly improved as the errors get filtered as the context passes down the model chain. However, misclassification in earlier stages will propagate the errors down the chain, thus affecting the prediction of later models. Running several models in a chain can increase computing time.

**Scalability**

The architecture can be greatly scalable since the models operate individually in the pipeline. With minimal modification, additional model blocks can be added to the pipeline without much effort. This will help in increasing the functionalities provided by the architecture. As the chain grows larger, it becomes difficult to maintain as failure in one model block will affect the rest following the chain.

**Component Identification**

1. **Data Loader/Ingestion Block**

Roles & Responsibility:

* Its main role is to read CSV files.
* It is used to merge them into a unified dataframe to feed into AI models.

1. **Preprocessing Block**

Roles & Responsibility:

* It is used to process the data by cleaning and normalizing the data.
* It will mainly process duplicate contents, noises and redundant data.
* It will also convert the text in other languages into English.

1. **Embedding Block**

Roles & Responsibility:

* It will convert text into numerical vectors for models to process.
* Main embedding tool used is TF-IDF vectorization.

Reusability:

* This block is used in each and every models in the pipeline blocks.

1. **Pipeline Block**

Roles & Responsibility:

* This block manages the chaining of Random Forest Classification models and ensures proper data flow from one model to the next.
* It contains multiple AI model blocks connected in chain.
* This block will ensure models are executed sequentially in the chain.

Reusability:

* This block can be scaled to add more models for different functionalities or modified to use different algorithms.

1. **Feature Combiner Block**

Roles & Responsibility:

* It is used to merge the embeddings with output from earlier models.
* It is responsible for maintaining context throughout the execution process.

Reusability:

* This block is reused at every stage in the pipeline.
* The models output is combined with embedding at every stage of the pipeline.

1. **Model Block**

Roles & Responsibility:

* It contains AI models, especially RandomForest classifier.
* Training of the model is carried out.
* Prediction on test data is also carried out.

Reusability:

* This block is reused by the pipeline model blocks to implement RandomForest model for classification.
* The algorithms can be easily replaced by other Algorithms with minimal modifications without impacting the other stages.

1. **Evaluation Block**

Roles & Responsibility:

* It is used to evaluate performance and measurement metrics of AI models.
* It contains features such as Precision, Recall, F1 Score and Support and Confusion Matrices.

Reusability:

* This block is reused by each Model Block to calculate and print the results.
* Advanced Statistical analysis can also be added in the future.

**Connector Identification**

Connections and Interactions between the blocks in the architecture.

1. **Data Loader and Preprocessor**

Data loader will load the raw data from CSV files and pass it preprocessor block. It is a synchronous operation containing raw text data and labels such as y1, y2, y3, y4. Both the blocks only share the data formats which could be modified without affecting the internal logic, thus making it loosely coupled.

1. **Preprocessor and Embedding Block**

The preprocessor will send cleaned and processed data with normalized text to the embedding block for vectorization. Since the preprocessor does not depend on the embedding process, it is loosely coupled and synchronized as it only has to pass the text as input to embedding block.

1. **Embedding Block and Pipeline Chain**

Embedding Block communicated with the several AI model blocks inside the Pipeline chain. Since Pipeline Chain is a virtual division, there are no direct communication.

1. **Embedding Block and Model Block (y2)**

Embedding block will send vectorized data to the first AI model block (y2) in the pipeline chain for training and prediction. Numeric data of the features are passed. It’s a loosely coupled and synchronous operation.

1. **Embedding Block and Feature Combiner**

Feature Combiner block will receive updated embeddings from the embedding block which will then be combined before being fed into the next model blocks. It is a asynchronous operation and tightly coupled as the feature combiner is dependent on the embedding block.

1. **Model Block (y2) and Feature Combiner**

Feature Combiner block will also receive output from model block (y2) in order to be combined with embeddings to generate input data for the models in next stages of the pipeline chain. It is an synchronous and tightly coupled operation.

1. **Feature Combiner and Model Block (y3)**

The updated feature matrix containing original embeddings and previous model predictions is sent as input to model block (y3), the next in the chain. It’s a loosely coupled operation since the models will process any numeric data as long as the data dimensions are compatible.

1. **Model Block (y3) and Feature Combiner**

Output (y3) is sent to the feature combiner where previous embeddings are added to retain context. It is a synchronous and tightly coupled operation.

1. **Feature Combiner and Model Block (y4)**

The final updated data containing embeddings and outputs from all previous models is fed into model block (y4) which will predict the final output containing resolution.

1. **Model Blocks and Pipeline Chain**

The model blocks has no communication with the pipeline chain, as the pipeline only ensures proper flow of data into the next stages.

1. **Model Blocks and Evaluation Block:** The output generated from the model blocks are passed on to evaluation block that computes performance metrics such as support, precision, accuracy and confusion matrix. It’s a synchronous operation

**Data Elements and Format Consistency**

**Different Kinds of Data Elements Passed**

1. Raw Text: This is the original data from CSV files that contains email fields like Ticker summary, emails, interaction content, and categories.
2. Processed Text: This is the text derived after processing original texts. This data is the result of removing redundancies and noises.
3. Embeddings: Embeddings are the data produced after the processed texts in string format are represented as numerical format using techniques such as TF-IDF.
4. Model Labels: These are the feature variables, some that are also produced by the AI models in the pipeline chain which will be fed to next models as inputs in the form of training data.
5. Model Predictions: These are the output predictions generated by the AI models in the pipeline chain.
6. Combined Features: This data is the combination of embeddings merged with predictions of the previous AI models.

**Methods to Maintain Consistent Input/Output Formats**

1. **Use of even Pandas Dataframe**: Pandas Dataframe has been used throughout to maintain consistency in data. Storing the processed data in columns, labelling the columns and appending the model predictions as new columns helps in managing and tracking data via consistent column names.
2. **Numeric Data using Vectorization**: AI models only accept data in numeric format and not in strings. Thus before passing that data for training, the data is embedded using TF-IDF vectorization technique. This will thus ensure numeric format is passed to the models.
3. **Numerical encoding of target variables**: In a single chain prediction, the target variable can be in non-numeric format. But, in case of Multi Chain prediction, since the output of previous model needs to be passed as input to the future models in the chain, it is essential to encode the data into numeric format before being forwarded. Label encoders are used for numerical conversion and reshape method will format the data dimensions into 2D arrays.
4. **Abstract Classes**: Use of abstract classes with abstract methods has provided with common interface for all model classes and has aided in reusability.
5. **Inheritance**: Extension of abstract classes through inheritance particularly while implementing AI models such as RandomForest has prevented code duplication while also allowing to add new models easily in the near future.
6. **Encapsulation**: Use of setters and getters methods while defining the Data class has allowed each objects to store and manage its own internal starts and has made setting and fetching of data uniform.
7. **Modularity and Coupling**: Code modularity has increased maintainability of the code and loose coupling approach has been used to maximum extent to decrease dependencies between each architecture blocks.
8. **Polymorphism**: Unified Model Interface has allowed certain method names to be reused while providing different functionalities as required by the context.

**Evaluation, Interpretation and Report of the results**

**Group: AppGallery & Games**

* y2 Classification
  + Accuracy: 94.26%
  + “Others” has a good precision of 0.83 but the lower recall 0.71 indicates may missing of some examples.
  + “Problem/Fault” has a precision of 0.75 and recall of 0.86.
  + “Suggestion displays underperformance with low precision.
* y3 Classification
  + Accuracy: 89.34%
  + “General” and “Others” have zero recall which suggeste that the model is not learning these classes due to few samples.
  + “Coupon/Gifts/Points Issues” has a good performance with f1 being 0.89.
* y4 Classification
  + Accuracy: 83.60%
  + Some classes have been predicted perfectly such as “Offers / Vouchers / Promotions” and “Cooperated campaign issue”.
  + “Personal data” and “Can't update Apps” have 0 recall because of few examples.
  + “Nan” has large count of data issues and needs fixing.

**Group: In-App Purchase**

* y2 Classification
  + Accuracy: 94.05%
  + “Problem/Fault is not detected at all.
  + “Suggestion” is predicted strongly with precision 0.88, recall 0.93.
* y3 Classification
* Accuracy: 90.48%
* “Other “ is completely missed due to class imbalance.
* Payment-related issues is predicted well.
  + - y4 Classification
    - Accuracy: 85.71%
    - “Invoice related request” and “Query deduction details” have zero recall.
    - “Subscription cancellation” predicted well.

**Conclusion**

The Random Forest models in the pipeline chain that are trained on the data produces good accuracy of 84% to 94%. However, because of the minorities classes there is a class imbalance, and some of them have zero recall. In “AppGallery & Games,” y3 and y4 labels shows good performance, where some of the classes are predicted perfectly while some are missed. “In-App Purchase,” is predicted well, but some classes remain undetected.