UDACITY

‹ Return to Classroom

Disaster Response Pipeline

REVIEW

CODE REVIEW 5

HISTORY

Meets Specifications

Congratulations 👏✨✨

Dear Learner,

You have passed the project "**Disaster Response Pipeline**" with flying colors.🎯
This is an amazing piece of work you have done here.

I went through all your modules and it clearly demonstrates all the functionalities.
The way you have implemented your logic was awesome. Great job.
I really appreciate your hard work and dedication to this project.

Keep Learning and keep doing the good work.
Good luck !!!

Github & Code Quality

All project code is stored in a GitHub repository and a link to the repository has been provided for reviewers. The student made at least 3 commits to this repository.

Nice job!

- ✅ All the project code is pushed to GitHub.
- ✅ The link is available for review.
- ✅ There are a significant number of commits with meaningful commit messages.

Resources

- Git Branching
- 5 types of GitWork flows

The README file includes a summary of the project, how to run the Python scripts and web app, and an explanation of the files in the repository. Comments are used effectively and each function has a docstring.

The README file includes:-

- ✅ A summary of the project
- ✅ How to run the Python scripts and web app,
- ✅ An explanation of the files in the repository.
- ✅ Comments are used effectively and each function has a docstring.

Resources

- Anatomy of a README
- Python docstrings- PEP257

Scripts have an intuitive, easy-to-follow structure with code separated into logical functions. Naming for variables and functions follows the PEP8 style guidelines.

Nice job, the entire code is properly organized.

- ✅ Scripts have an intuitive, easy-to-follow structure.
- ✅ The code is separated into logical functions.
- ✅ Variables and functions follow the PEP8 style guidelines.

ETL

The ETL script, process_data.py, runs in the terminal without errors. The script takes the file paths of the two datasets and database, cleans the datasets, and stores the clean data into a SQLite database in the specified database file path.

The ETL script, process_data.py, runs in the terminal without errors. ✅
The script:-

- ✅ Takes the file paths of the two datasets and database
- ✅ Cleans the datasets
- ✅ Stores the clean data into an SQLite database

The script successfully follows steps to clean the dataset. It merges the messages and categories datasets, splits the categories column into separate, clearly named columns, converts values to binary, and drops duplicates.

Nice job here!

The script nicely follows steps to clean the dataset:-

- ✅ messages and categories datasets are merged
- ✅ The categories column is splitted into separate, properly named columns
- ✅ Converts values to binary
- ✅ Drops duplicates

Machine Learning

The machine learning script, train_classifier.py, runs in the terminal without errors. The script takes the database file path and model file path, creates and trains a classifier, and stores the classifier into a pickle file to the specified model file path.

The machine learning script, train_classifier.py, runs in the terminal without errors. ✅
The script takes:-

- ✅ The database file path and model file path.
- ✅ Creates and trains a classifier, and stores the classifier into a pickle file.

Further reading

- Sklearn pipeline
- Using countvectorizer to extracting features
- Multioutput classification with multioutputclassifier

The script uses a custom tokenize function using nltk to case normalize, lemmatize, and tokenize text. This function is used in the machine learning pipeline to vectorize and then apply TF-IDF to the text.

Your tokenize function succesfully:

- Case normalizes the text. ✓
- Uses a lemmatizer ✓
- Tokenizes the text using NLTK's word_tokenize. ✓

Resources

Removing stop words

The script builds a pipeline that processes text and then performs multi-output classification on the 36 categories in the dataset. GridSearchCV is used to find the best parameters for the model.

- ✅ The script builds a pipeline that processes text and then performs multi-output classification on the 36 categories in the dataset.
- ✅ GridSearchCV is used to find the best parameters for the model.

The TF-IDF pipeline is only trained with the training data. The f1 score, precision and recall for the test set is outputted for each category.

- ✅ The TF-IDF pipeline is only trained with the training data.
- ✅ The f1 score, precision and recall for the test set is outputted for each category.

Deployment

The web app, run.py, runs in the terminal without errors. The main page includes at least two visualizations using data from the SQLite database.

Awesome job!

The web app:-

- ✅ Runs in the terminal without errors.
- ✅ Shows the required plots on the home page.

When a user inputs a message into the app, the app returns classification results for all 36 categories.

Brilliant job 💥💥

The application returns classification results for all 36 categories. ✅

DOWNLOAD PROJECT

5 CODE REVIEW COMMENTS ›

RETURN TO PATH