Detecting a Programming Language with Machine Learning

WGU Computer Science Capstone

David Zentner
1/1/2023

SECTION A

Letter of Transmittal

Dec 26th, 2022

Mr. Paul Dixon
Pastebin.com
3000 C St, Ste 301
Anchorage, Alaska, 99503

Mr. Dixon,

Your website offers a quick and easy way to share code snippets between users. As you know, this ability to post and share a new code snippet within a few seconds is one of the biggest advantages your website has. However, because users are in such a hurry to post a code snippet, they seldom select the programming language the code snippet is written in, as this is not a requirement to post the snippet. While this does expedite the process of posting the code snippet, the user they share the code snippet with is left to read an unformatted code block which does not offer language specific syntax highlighting or other features that are available when the language is identified. So the tradeoff of not requiring a poster to select the language is a faster posting experience, at the expense of readability of the code snippet.

I would like to propose a system which will automatically identify the programming language of the code snippet, thus eliminating the manual step of identifying the language. This will provide a better user experience for those reading the code snippet, and will offer pastebin.com a competitive advantage over other similar snippet sharing websites.

There are 2 main considerations for this feature addition: first, the accuracy of the automatic language detection tool, and the scalability of this feature. For this proposal, I will demonstrate the accuracy of this tool, and make it available for testing against your current database of code snippets. If the service meets your standards, then we can proceed with implementing a scaled version of this new feature in another proposal.

Regards,
David Zentner
Chief Software Architect
Dazcode Programming Solutions

Project Recommendation

Problem Summary

Snippets of code are often shared between programmers in instant messages, emails, blogs, and documentation. Unless the person who posts the code specifically tags the code with the language used, it will be shared as simply a generic unformatted code block. These generic code blocks lack many language specific features which can be helpful in understanding the code block.

By automatically identifying the programming language of these code snippets using machine learning, the code snippets can then be formatted automatically without the user having to specify the language. This does provide a benefit that may not be apparent: which is that in most cases when users post code snippets, in order to manually specify the code language, they must be familiar with and use mark-down (https://en.wikipedia.org/wiki/Markdown) format, which not all programmers are well versed in. Consequently, it is common for code snippets to be posted without specifying the language, and they are posted as generic unformatted code blocks. For this project, only the code prediction will be performed, and the formatting itself is out of scope, and will be done with a 3rd party tool.

Application Benefits

The benefits of code identification include: language specific formatting, syntax highlighting for readability, syntax error checking to check for code errors, and additional features such as code completion, code analysis, and enabling linking of code fragments to external help resources. I propose a system which will input a code block and identify and output the programming language of the code block in order to enable these features.

Application Description

The project is written initially with a web browser based user interface which takes as input a single code fragment and will output the predicted programming language of the code fragment. This is mainly to function as a proof of concept, and afterwards a scalable REST based service will be created which will allow this programming language prediction to operate independently from any given user interface. This scalable REST service is out of scope for this current project and will be produced as a followup project.

Data Description

The data used to train the machine learning model used by this solution is an open source dataset located at: https://github.com/TheAlgorithms . The code located at this endpoint is a set of common computer algorithms, with each repository representing algorithms in a specific

language.The code is listed under the MIT license, so it is freely available for use on this project. 5 total language classifications will be used for this project: Python,Java,JavaScript,Go, and C++ .

Objective and Hypothesis

The objective of this project is to take a given computer code fragment and accurately predict the programming language that the code fragment is written in. The hypothesis is that by training a machine learning classifier model based on open source code, the model can then be used to accurately predict the programming language.

Methodology

The selected development methodology for this project is the waterfall approach. This is a structured approach to software project management where the development lifecycle steps are completed in sequential order. For this specific project, the requirements are well known in advance, which makes the waterfall approach preferable over an iterative approach such as agile. While the waterfall approach may be criticized for being an outdated traditional approach which is more rigid and less flexible than agile, when the requirements are known in advance, waterfall makes it easier to plan and communicate the timeline, budget, milestones and deliverables.

Funding Requirements

Developer costs: 1 developer for 1 month	$15,000.00
Hardware costs: 1 Laptop computer	$1,500.00
Hosting costs: Project is run locally	$0.00
Software Licensing costs: None (Open source/Free to use software is used)	$0.00
Total:	$16,500

Stakeholders Impact

The impact for customers is a better experience using the application as a result of saving time by not having to select a language when they are posting, and more snippets will automatically be tagged with the correct language, which will allow customers reading the code snippet to be able to comprehend it faster due to the language specific syntax highlighting. As a result, the service will have an advantage over the competition and will experience more repeat customer usage. Since the service usage will be used more frequently by customers, the impact for investors, and employees will be more revenue and job security.

Data Precautions

There are ethical and legal considerations which must be handled. Since the input to the system is often proprietary code, it is often owned or copyrighted by the user and may also contain unique identifiers or system access tokens. Because of this, no user code submissions will be stored in any way by the language predictor service.

Developer Expertise

I have over 10 years experience developing successful full stack web applications and databases at startups,academic institutions, corporate environments, and on a contract basis. I will be graduating with a bachelor's degree in computer science from WGU by the end of 2022.

SECTION B

Project Proposal

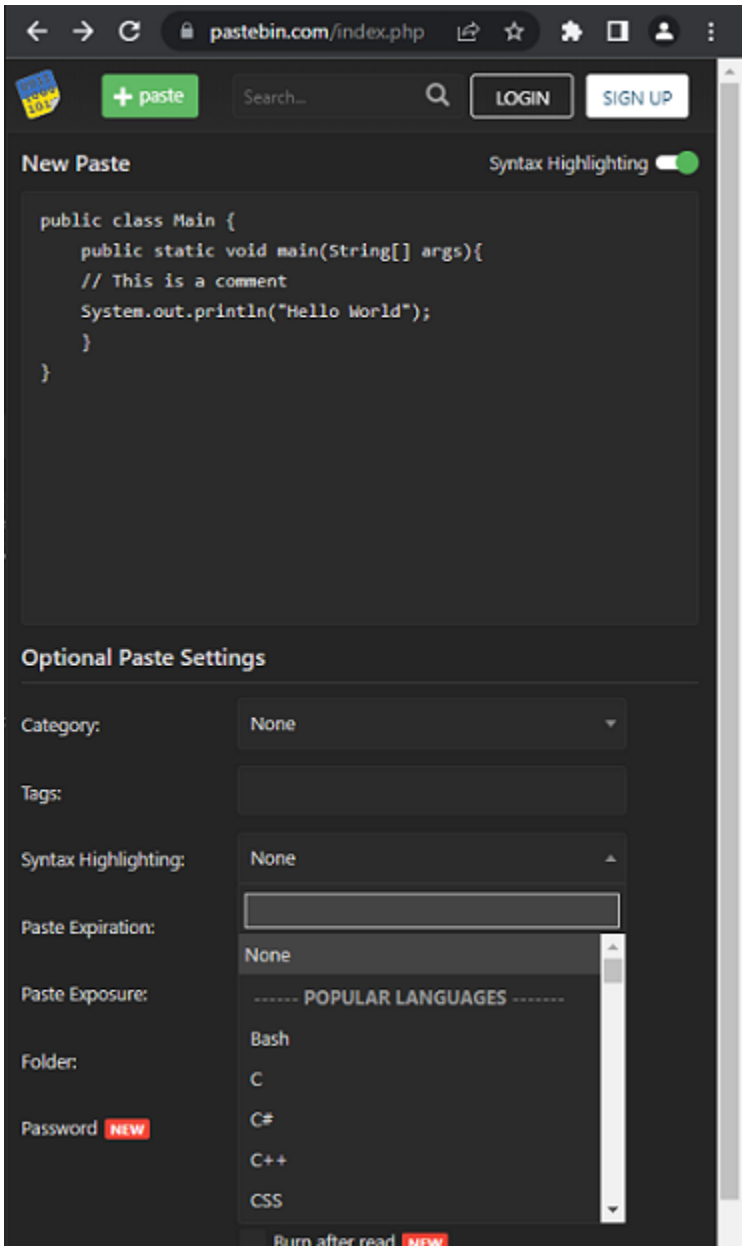
Problem Statement

The programming language of a source code file is typically identified by the file extension. For example Java programs typically use .java, Javascript files typically use .js, and Python files typically use .py . While this convention works effectively to identify the type of code in a file, not all source code resides in files. Source code is often transmitted in 'snippets' over chat programs such as Slack/Discord, or posted on blogs, tutorials,email, and online documentation. In these cases, the programming language is not known, unless explicitly identified by the user or implicitly by examining the code.

Customer Summary

Pastebin.com was founded in 2002 by Paul Dixon as a text storage site where users can post text and code snippets up to 512kb in size per paste for free. It has become a popular site for programmers to post code snippets, and as of 2019 the site reported 17 million unique monthly users. The code snippets typically represent a small block of code, such as an individual algorithm, function, or a single code file.

Existing System Analysis

Currently the existing system requires the user to manually select the programming language from a dropdown list.


Data

Repository	Language	# of files	size
https://github.com/TheAlgorithms/Python	Python	765	13MB
https://github.com/TheAlgorithms/Java	Java	536	2MB
https://github.com/TheAlgorithms/JavaScript	JavaScript	552	2MB
https://github.com/TheAlgorithms/Go	GoLang	309	1MB
https://github.com/TheAlgorithms/C-Plus-Plus	C++	351	3MB

Project Methodology

The project methodology follows the waterfall model as specified below:

1. Requirements
The first step in the waterfall model is to gather and define the project requirements and to determine the cost and timeline. These documents must be approved by project stakeholders before proceeding to the next phase. The capstone project approval form contains the high level project requirements and was approved by the course instructor and is used to set the agreed upon project outcome.

2. Design
During the design phase, the high level design of the software system is produced. The output of this phase are visual system design diagrams which can be used as a guide to produce the project code. For this project user interface mockups were produced as well as basic system design diagrams. These visual design documents guide development during the coding phase as well as being useful in communicating a high level view of the system to project stakeholders.

3. Coding
At this stage the project code will be written, and since it has a simple single page interface and limited functional scope, I will be able to complete the project in a reasonable amount of time as the sole developer. Since this is a data science project, writing the code in the Python language is a good fit, as there are extensive machine learning resources in this language. The development environment is set up by creating a new project in Pycharm, and the algorithm and application code is developed to meet the documented requirements. The project code is maintained in a local git repository in order to track history, changes, and version control.

4. Verification

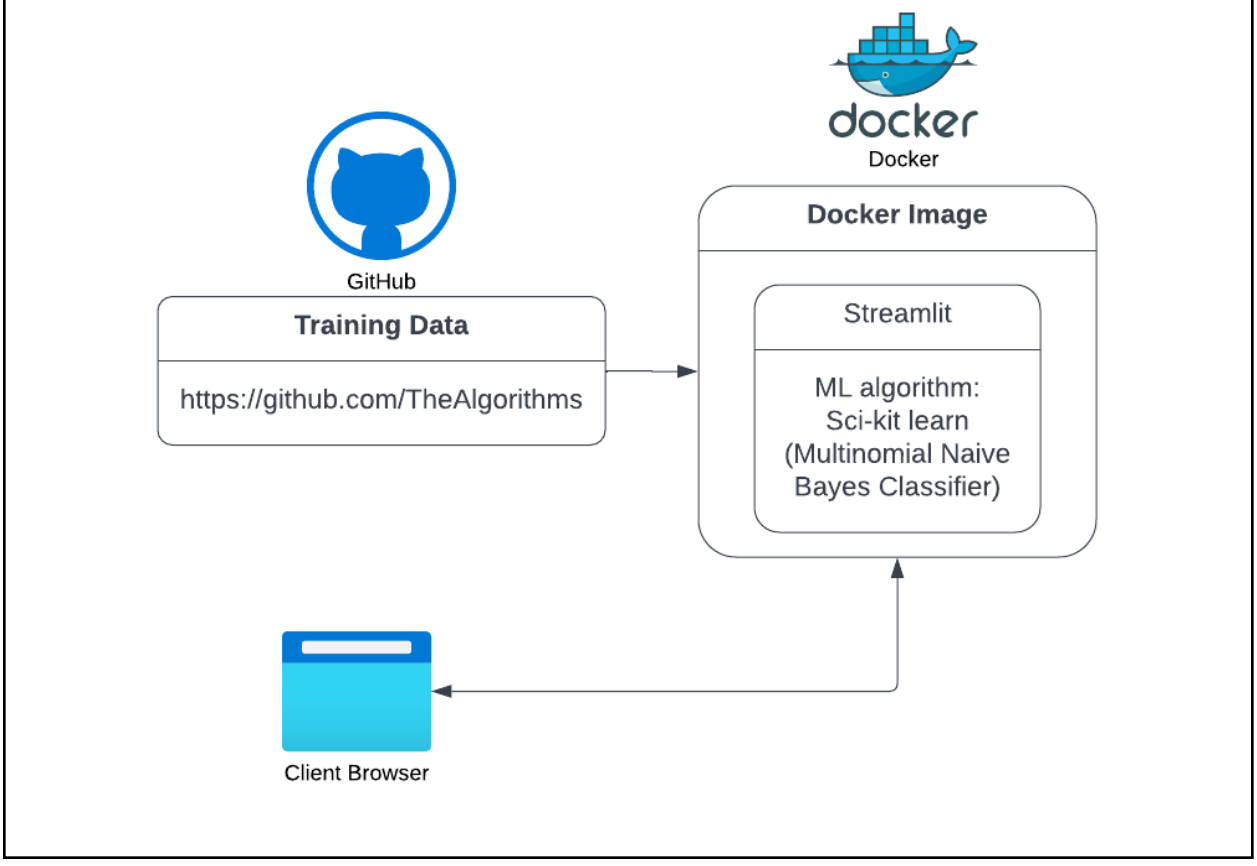
After the code is developed, each feature is individually tested against a variety of manual input and scenarios to make sure it functions as intended and meets the requirements. In addition to the manual testing, machine learning model validation analysis is done. After local testing and confirming the project works on the development machine, the software is then deployed to a test environment accessible to testing and acceptance by project stakeholders.

5. Maintenance
After the project is verified and deployed, all bugs and user reports are recorded and handled according to their severity. To facilitate the maintenance process, software such as Jira can be used to track bugs and feature requests from stakeholders.

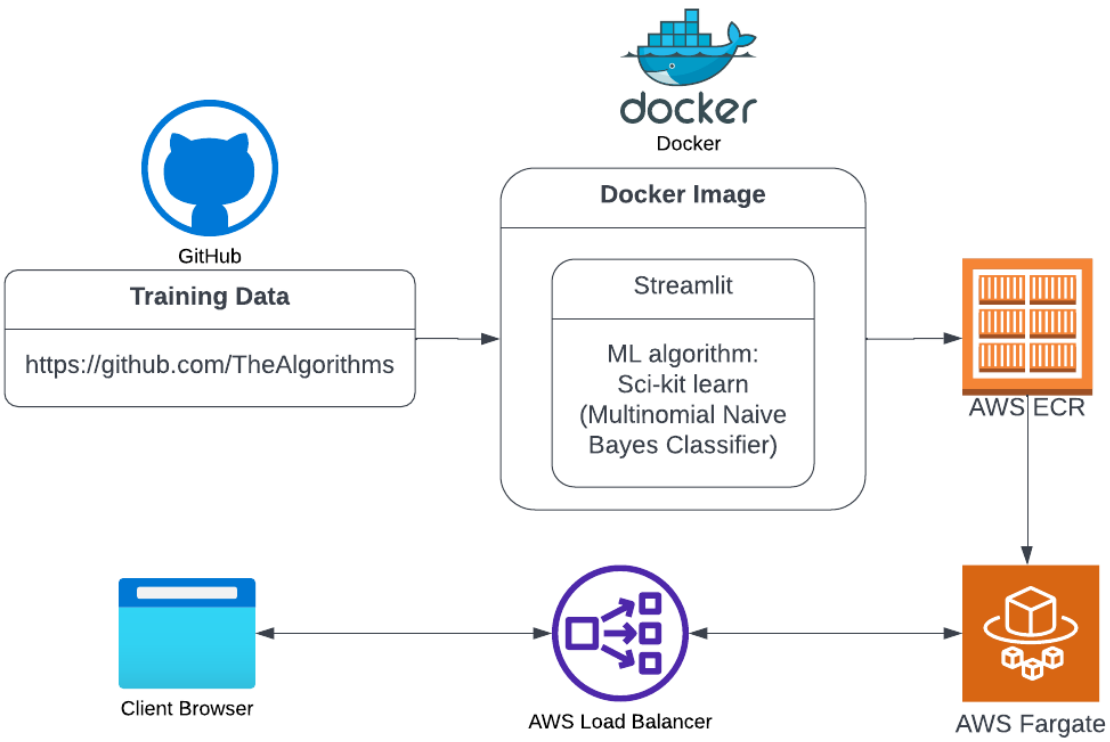
Project Outcomes

The project outcome is a streamlit based web application which can detect the input code language given a code snippet. This application can be run locally, or by accessing a publicly available url. The application can be accessed with an up to date web browser.

Local application: Application running on a local docker image



Online application: Application running on AWS cloud service



Implementation Plan

1. Create initial code in Jupyter notebook
   - Research, identify,import training datasets
   - Research and implement a machine learning algorithm
   - Test the ML model using multiple folds and determine accuracy
   - Print visual graphs of training dataset, and model accuracy.
   - Clean and modify dataset based on what was learned on the model accuracy
2. Build Streamlit web interface
   - Create streamlit web controls and application flow
   - Integrate code from Jupyter notebook into the streamlit application
3. Deploy application to a web server
   - Research web server hosts, create deployment plans and deploy.
4. Get project feedback from stakeholders
   - Provide a local copy of the project zip files
   - Provide a web accessible link to the project

Evaluation Plan

The machine learning algorithm is evaluated by using Sci-kit learn's K-Folds cross-validator library (sklearn.model_selection.KFold). Rather than splitting the model datasets into a training and test set, the K Fold validation method rotates the train/test sets. This method of model accuracy verification splits the training dataset into an arbitrary number of "K" groups, where each of the data groups is set aside as a test set to validate against the remaining K-1 groups. To accomplish this, the KFold library from Sci-kit learn was used with a chosen value K=5 groups.

Resources and Costs

Development Resources	
Docker Desktop	$0.00
Pycharm Community Edition	$0.00
Jupyter Notebook	$0.00
Training Datasets (open source) https://github.com/TheAlgorithms	$0.00

Deployment Resources	
Fargate Task 1vCPU 3GB RAM	$25.00/month
Load Balancer	$7.50/month

Timeline and Milestones

Milestone	Start	End
Define project requirements	11/1/2022	11/10/2022
Create initial machine learning algorithm in Jupyter notebook	11/12/2022	11/18/2022

Design Streamlit web app layout	11/20/2022	11/24/2022
Transfer Jupyter notebook code to Streamlit web app	11/26/2022	11/28/2022
Test and Validate	12/1/2022	12/4/2022
Deploy application to cloud	12/6/2022	12/10/2022

SECTION C

Application Files

main.py	The main application file.
functions.py	Supporting functions
Dockerfile	The docker config
.dockerignore	Used by docker to ignore certain directories when copying
requirements.txt	List of the project's python dependencies
install.txt	Docker commands to install/run
examples.txt	Example code snippets
.streamlit/config.toml	Streamlit settings file

SECTION D

Post-implementation Report

Project Purpose

The purpose is to identify the programming language of a code snippet. Code identification enables the code to then be processed by a code formatter which provides the following benefits: language specific formatting, syntax highlighting for readability, syntax error checking to check for code errors, and additional features such as code completion, code analysis, and enabling linking of code fragments to external help resources.

Datasets

The datasets used to train the machine learning algorithm come from an open source project called TheAlgorithms (https://github.com/TheAlgorithms). The datasets from this github project were chosen because it represents self contained code files similar to snippets shared by programmers. For this project 5 languages were selected to train the machine learning classifier: Java, Javascript, C++, Python, and Golang.

Example Java code file from:
TheAlgorithms/Java/blob/master/src/main/java/com/thealgorithms/strings/ReverseString.java

```
package com.thealgorithms.strings;

/**
 * Reverse String using different version
 */
public class ReverseString {

    public static void main(String[] args) {
        assert reverse("abc123").equals("321cba");
        assert reverse2("abc123").equals("321cba");
    }

    /**
     * easiest way to reverses the string str and returns it
     *
     * @param str string to be reversed
     * @return reversed string
     */
    public static String reverse(String str) {
        return new StringBuilder(str).reverse().toString();
```

```
    }

    /**
     * second way to reverses the string str and returns it
     *
     * @param str string to be reversed
     * @return reversed string
     */
    public static String reverse2(String str) {
        if (str == null || str.isEmpty()) {
            return str;
        }

        char[] value = str.toCharArray();
        for (int i = 0, j = str.length() - 1; i < j; i++, j--) {
            char temp = value[i];
            value[i] = value[j];
            value[j] = temp;
        }
        return new String(value);
    }
}
```

Data Product Code

The project written in Python 3.11 (https://www.python.org/) , uses a multinomial naive bayes classifier machine learning algorithm from Sci-kit learn (https://scikit-learn.org/) , a web based interface using Streamlit (https://streamlit.io/) , and is ran in a Docker container (https://www.docker.com/) .

The first step in the program logic is to download the data the algorithm will train on. This is done automatically when building the docker image.

code_detect/Dockerfile

```
# Download data used to train the algorithm
RUN git clone https://github.com/TheAlgorithms/Python ./data/python
RUN git clone https://github.com/TheAlgorithms/Java ./data/java
RUN git clone https://github.com/TheAlgorithms/C-Plus-Plus ./data/cplusplus
RUN git clone https://github.com/TheAlgorithms/JavaScript ./data/javascript
RUN git clone https://github.com/TheAlgorithms/Go ./data/golang
```

This data is then loaded into a pandas dataframe when the Streamlit application is run.

code_detect/main.py

```
# List of supported languages. Including location,extension, and name.
load_data=[]
load_data.append(["data/python", ".py", "python"])
load_data.append(["data/javascript", ".js", "javascript"])
load_data.append(["data/java", ".java", "java"])
load_data.append(["data/cplusplus", ".cpp", "c++"])
load_data.append(["data/golang", ".go", "golang"])
supported_languages=",".join([str(item[2]).capitalize() for item in
load_data])

# Load supported language data into pandas dataframe
data = DataFrame({'code': [], 'language': []})
for location,extension,language in load_data:
   data = pd.concat([data, functions.data_frame_from_directory(location,
extension, language)])
```

The dataframes are sent to a count vectorizer. Then transformed and processed by multinomial naive bayes classifier for training the model.

code_detect/main.py

```
vectorizer = CountVectorizer(stop_words=stop_words, max_features=max_features)
X = vectorizer.fit_transform(data['code'].values)
classifier = MultinomialNB()
y = data['language'].values
classifier.fit(X, y)
```

Once the model is trained, the user can enter a code sample and click ‘detect language’. The user code sample is converted to vectorized tokens which are passed to the machine learning model and used to predict the language classification. The detected language is then displayed for the user.

code_detect/main.py

```
# The user clicks 'Detect code language' button
if st.button('Detect code language'):

   # Vectorize the user input text
   predict_code = [code]
   vector_counts = vectorizer.transform(predict_code)
```

```
# Predict the detected language classification
predictions = classifier.predict(vector_counts)
detected_language=str(predictions[0])

# Output the result
st.success(f'Language: {detected_language.upper()}')
```

Hypothesis Verification

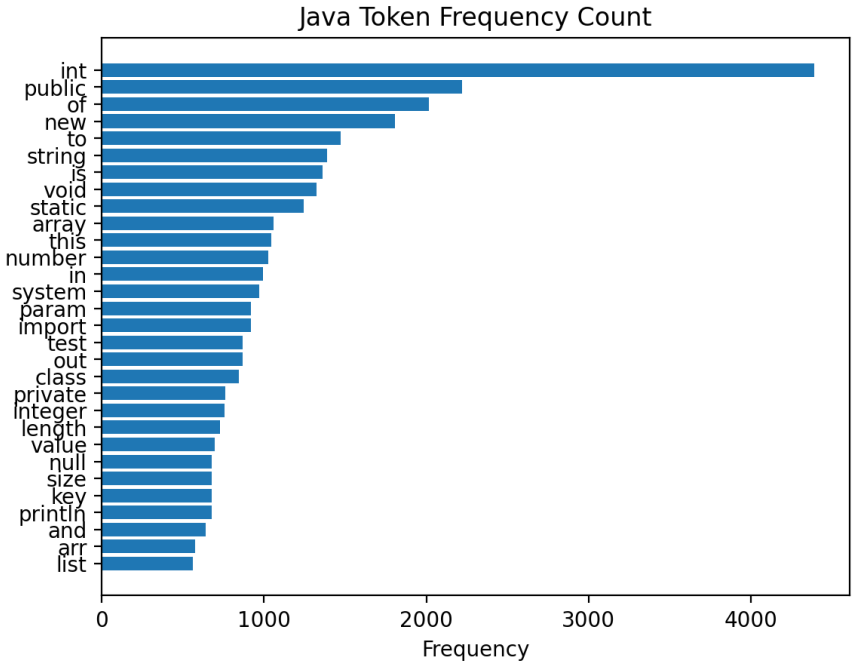
The hypothesis of this project is that by training a machine learning classifier model based on open source code can be used to accurately detect the programming language of a given code snippet. It turns out that this hypothesis can be verified quite well given the results from the accuracy tests. This hypothesis does currently hold true when 5 languages are supported for classification, and additional testing can be done after adding more supported languages.

Effective Visualizations and Reporting

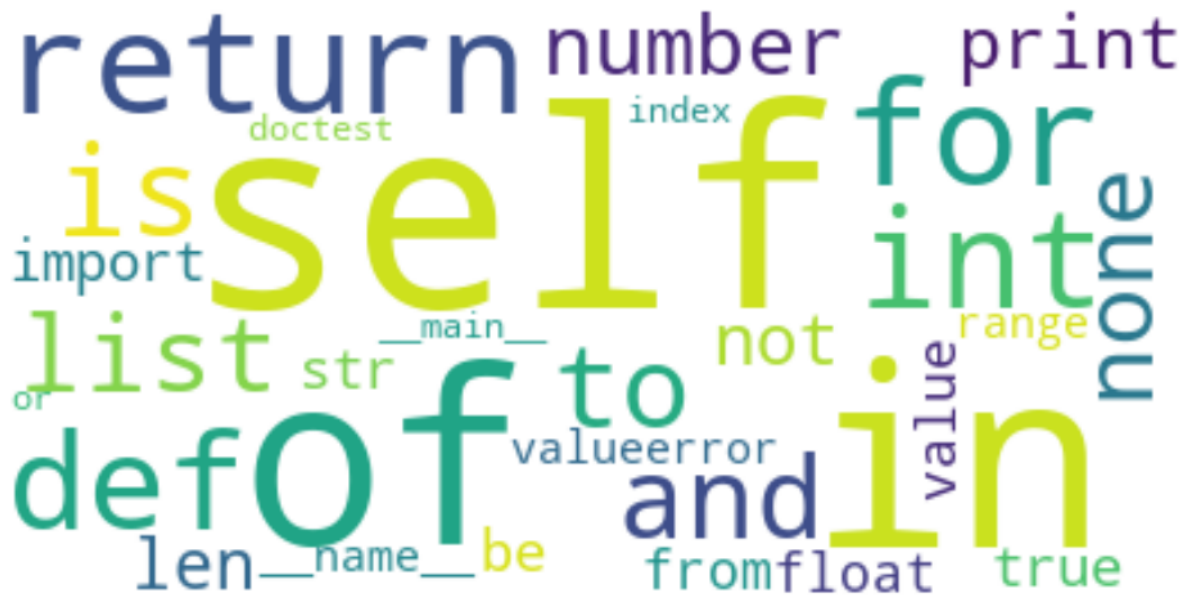
In order to gain insight on the dataset, a graph and word cloud of the vectorized tokens for each language was created. By comparing these visuals it was possible to identify certain tokens which should be added to the ‘stop_words’ list and not be used to train the model. This includes non programming related tokens such as ‘The’ and tokens common to each language such as ‘if’ and ‘return’.

In addition to the token visual diagrams, a confusion matrix was generated showing details of the classification accuracy of this supervised machine learning algorithm. This visual makes it easy to see which percentages of individual classification are accurately predicted or are wrongly predicted as another classification.

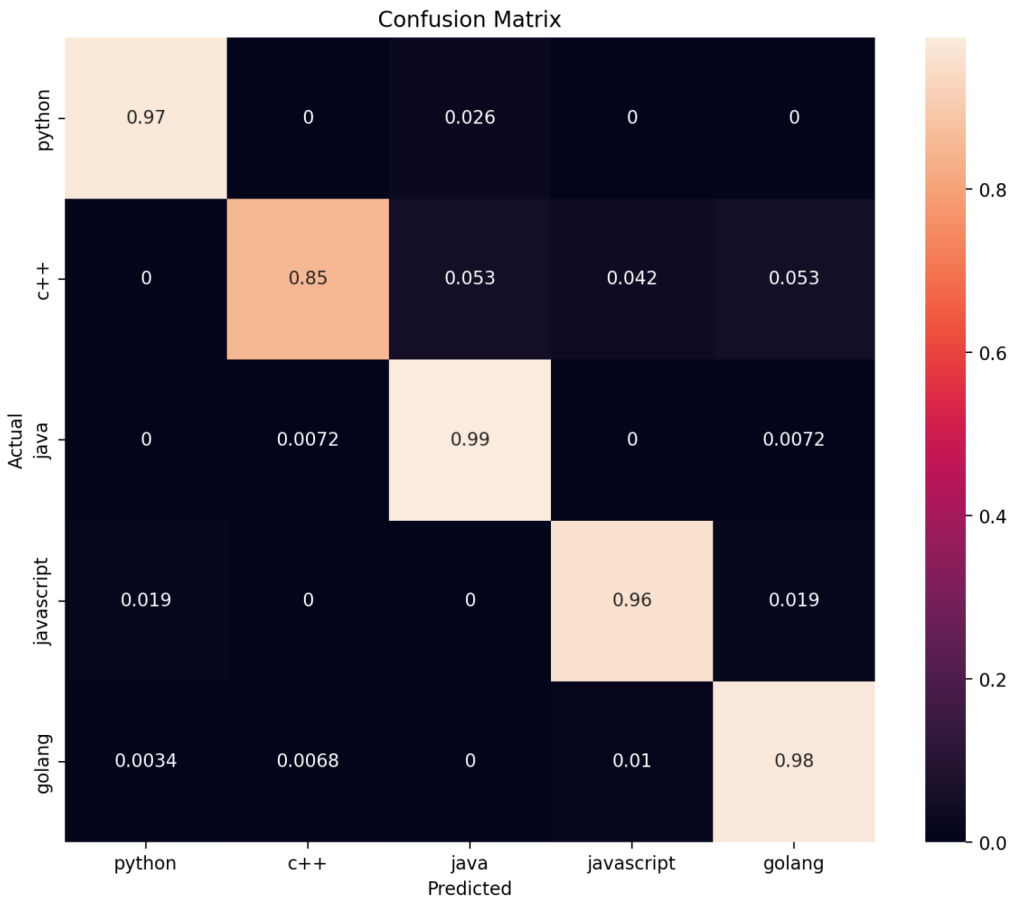
Java Token Frequency Chart Graph



Python Token Word Cloud



Confusion Matrix



The confusion matrix shows that the most accurately predicted language is Java at 99%, and the least accurately predicted language is C++ at 85%.

Accuracy Analysis

K-Fold cross-validation was used to verify the trained models accuracy. This method of model accuracy verification splits the training dataset into an arbitrary number of "K" groups, where each of the groups is set aside as a test set to validate against the remaining K-1 groups. To accomplish this, the KFold library from Sci-kit learn was used with a chosen value K=5 groups.

The rounded classification accuracy for each of the 5 folds are: .97, .96, .96, .97, .97. Which gives an average classification accuracy score of over 96% . This is a very good accuracy, and manual testing using code snippets from other open source software does also appear accurate. Testing specifically with code fragments which are deliberately ambiguous seems to produce a good, but less accurate response. However, in the majority of common code snippets, the accuracy is high.

Application Testing

In addition to the accuracy analysis using K-Fold cross validation, acceptance testing for the application was done on a windows machine using the chrome web browser. The application successfully passes all tests using the inputs specified in the ‘code_detect/examples.txt’ file.

Appendices

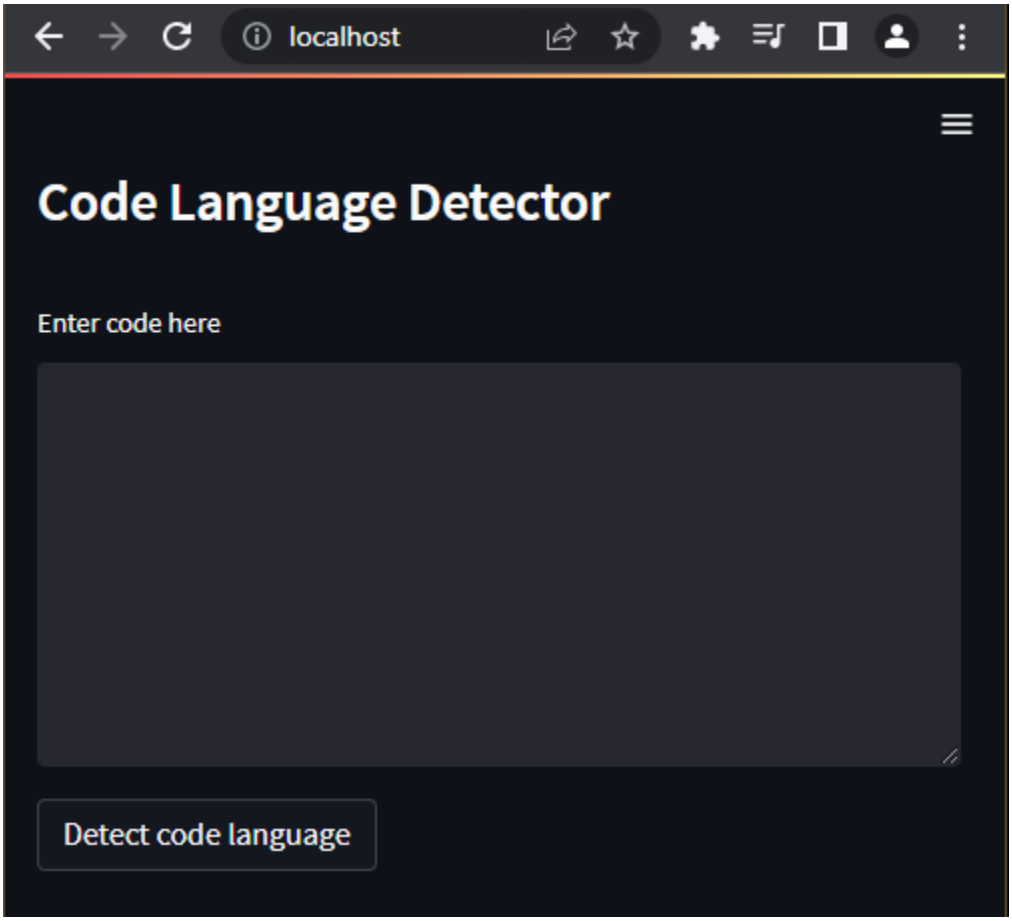
Installation Guide

The application can be accessed online at https://detect.dazcode.com , or by running a local copy by following the installation steps below.

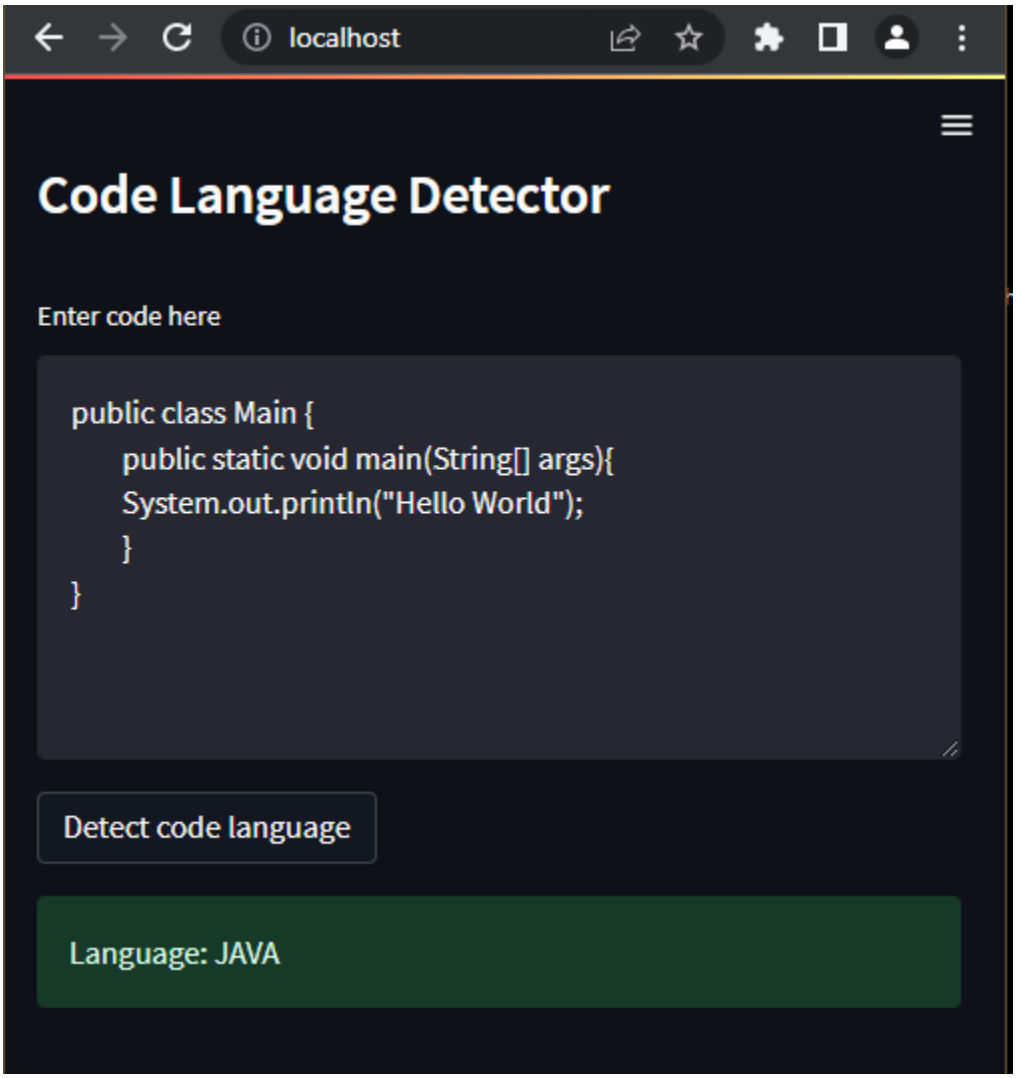
1. Download and install Docker: https://docs.docker.com/engine/install/
2. Download the project source code zip file: https://dazcode.com/code_detect.zip
3. Unzip project source code to the following folder: c:\code_detect
4. Open a command prompt, and change to project directory: `cd c:\code_detect`
5. Build the project docker image: `docker build -t streamlit .`
6. Run the docker image: `docker run -p 80:80 streamlit`
7. Open a web browser and visit http://localhost

User Guide

Step 1: Open the application in a web browser



Step 2: Enter a code snippet in a supported language and click ‘Detect code language’. The detected programming language will display below. There are example code snippets in the “examples.txt” file included in the project.



Summation of Learning Experience

This project was an excellent learning experience for me as I was able to learn the fundamentals of machine learning as well as practice technical writing. Working with the latest machine learning tools such as Jupyter notebook, Sci-kit learn, and Streamlit allowed me to implement this project solution much more effectively and with much less code than attempting to solve this classification problem by simply relying on my previous full-stack development experience. This project also gave me the opportunity to practice Dockerizing the application and deploying to a modern scalable cloud infrastructure on AWS. The skills I developed working on this project should prove very useful for my career in software development.