

Practical Work Report

Topics in Data Science

AI model to recognize customized
handwritten

Student: Muhammet Ali YILDIZ ,
1709981,
mtech00,

https://github.com/mtech00/dst_1709981_2024.git

yldzmuhammedali@gmail.com

Muhammet-YILDIZ-1709981



1. Job Description

The project involves leveraging Python libraries for various data science tasks. Here's a detailed breakdown:

Architectural Drawing

1. **Image Processing:** The project begins with loading an image file using the Python Imaging Library (PIL). The image is then resized to a specified dimension using the `resize()` function. Additionally, the image is split into smaller segments using a custom function (`split_image`) to facilitate further processing.
2. **Data Conversion:** After splitting the image, each segment is converted into numerical format representing RGB values. This conversion is crucial for feeding image data into machine learning models. The numerical data is saved as CSV files for easy access and manipulation.
3. **Data Preparation:** The converted CSV files are then prepared for model training and evaluation. The data is split into training and testing datasets using a predefined ratio. This ensures that the model is trained on a portion of the data and tested on unseen samples to assess its generalization ability.
4. **Model Training:** A Decision Tree Regressor model is chosen for its simplicity and interpretability. The model is trained using the training dataset, where features represent RGB values of image segments, and the target variable is derived from the image data.
5. **Model Evaluation:** The trained model's performance is evaluated using the testing dataset. Metrics such as accuracy are calculated to assess how well the model predicts the target variable. This step is crucial for understanding the model's effectiveness and potential areas for improvement.

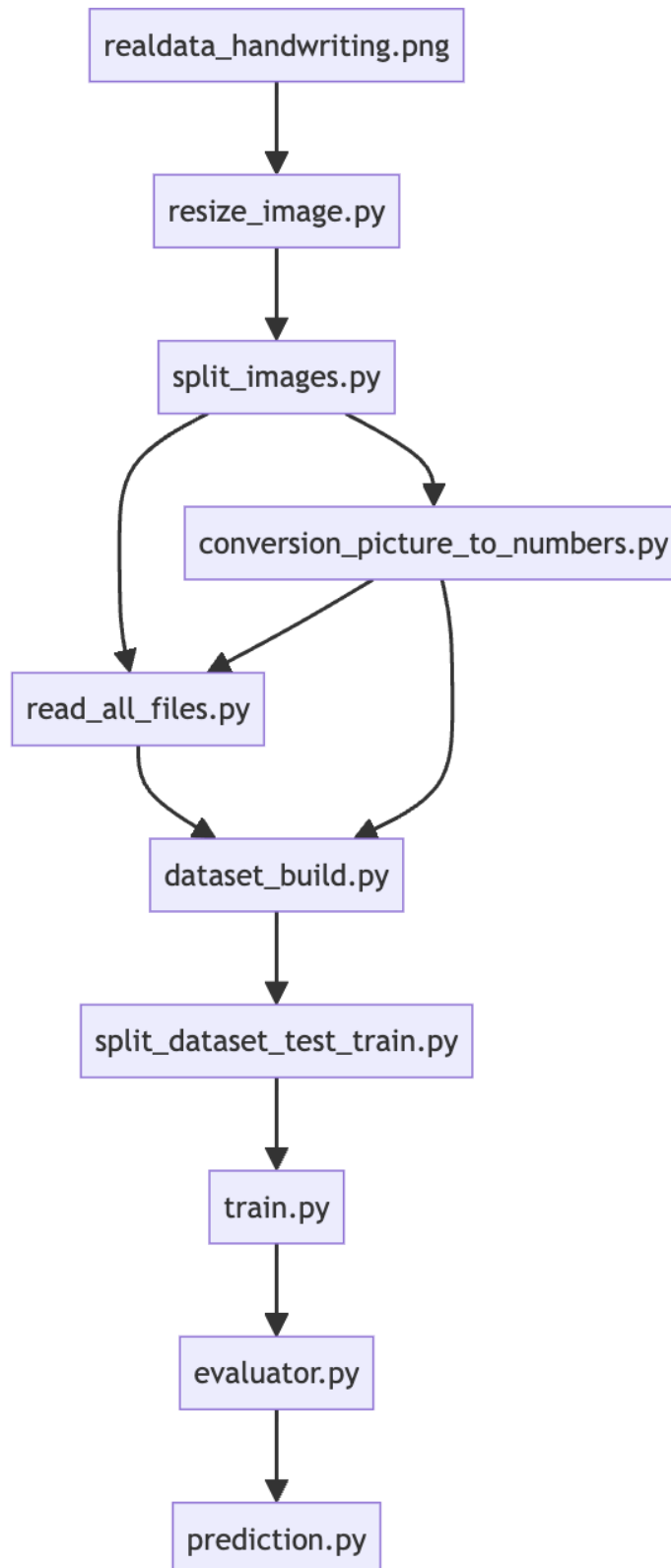
2. Work Implementation

The project's implementation involves executing the following steps:

1. **Image Processing:** The image processing stage begins with resizing the original image to a standard dimension to ensure consistency in input data. Subsequently, the resized image is split into smaller segments using a custom function that divides the image into a grid of specified rows and columns.
2. **Data Conversion:** Each segmented image is converted into a numerical format representing RGB values. This conversion transforms image data into a structured format suitable for machine learning algorithms. The converted data is then saved as CSV files for further analysis and model training.
3. **Data Preparation:** The converted CSV files are organized into training and testing datasets. The training dataset is used to train the model, while the testing dataset is reserved for evaluating the model's performance. This step ensures unbiased assessment and generalization of the model.
4. **Model Training:** The Decision Tree Regressor model is trained using the training dataset. The model learns patterns from the input features (RGB values) and predicts the target variable. Training involves optimizing model parameters to minimize prediction errors and improve overall performance.
5. **Model Evaluation:** The trained model is evaluated using the testing dataset to assess its predictive accuracy and generalization ability. Evaluation metrics such as mean squared

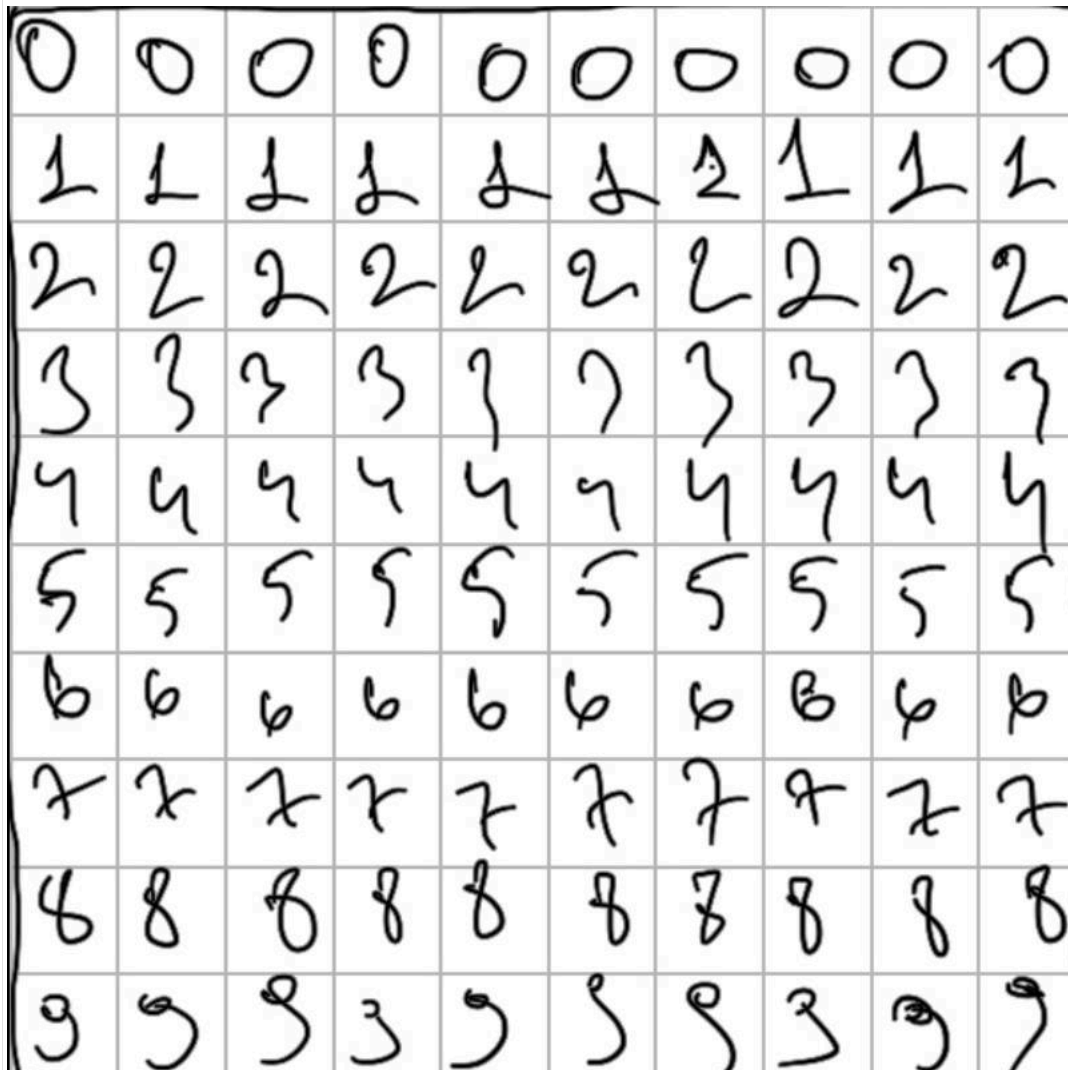
error and coefficient of determination (R^2) are calculated to quantify the model's performance. This step provides insights into the model's strengths and weaknesses.

Architecture Diagram



3. Work Operation

Original image :



```
Exporting image tile: ./output/realdata_handwriting_93.png
Exporting image tile: ./output/realdata_handwriting_94.png
Exporting image tile: ./output/realdata_handwriting_95.png
Exporting image tile: ./output/realdata_handwriting_96.png
Exporting image tile: ./output/realdata_handwriting_97.png
Exporting image tile: ./output/realdata_handwriting_98.png
Exporting image tile: ./output/realdata_handwriting_99.png

Process finished with exit code 0
```

```

/Users/mali/Desktop/dtsc/forpycharm/dtscproject/.venv/bin/python /Users/mali/Desktop/dst_1709981_2024/Handwriting_recognition_1709981/1resize_image.py
Original size : (800, 800)

Process finished with exit code 0

```

1. Image Processing (Resizing and Splitting):

- The code loads an image and resizes it to a standard dimension (512x512 pixels) for consistency.
- It then splits the resized image into smaller segments using a custom function, forming a grid of specified rows and columns.

```

Run 3.1read_all_files x
(80, 80, 4)
realdata_handwriting_65.png <class 'str'>
(80, 80, 4)
realdata_handwriting_59.png <class 'str'>
(80, 80, 4)
Files and directories in './output':
['realdata_handwriting_83.png', 'realdata_handwriting_97.png', 'realdata_handwriting_40.png', 'realdata_handwriting_54.png', 'realdata_handwriting_68.png', 'realdata_ha
Process finished with exit code 0

```

2. Data Conversion (Image to CSV):

- This part converts each segmented image into numerical format (RGB values) and saves them as CSV files.
- It iterates through the segmented images, processes each one using a Python script, and converts them into CSV format.

```

Run 4dataset_build x
conteudo_aux2= realdata_handwriting_96.png.csv [255,255,255,255];[255,255,255,255];[255,255,255,255];[255,255,255,255];[235,235,235,255];[183,183,183,255];[183,183,1
*****
conteudo_aux2= realdata_handwriting_96.png.csv [255,255,255,255];[255,255,255,255];[255,255,255,255];[255,255,255,255];[235,235,235,255];[183,183,183,255];[183,183,1
*****
conteudo_aux2= realdata_handwriting_96.png.csv [255,255,255,255];[255,255,255,255];[255,255,255,255];[255,255,255,255];[235,235,235,255];[183,183,183,255];[183,183,1
Files and directories in './output_text/':
['realdata_handwriting_48.png.csv', 'realdata_handwriting_58.png.csv', 'realdata_handwriting_93.png.csv', 'realdata_handwriting_83.png.csv', 'realdata_handwriting_77.
Process finished with exit code 0

1709981_2024 > Handwriting_recognition_1709981 > 4dataset_build.py 7:12 (1853 chars, 61 line breaks) CRLF UTF-8 4 spaces Python 3.12 (dtscproject)

64
Index(['Unnamed: 0', '255.0', '255.0.1', '255.0.2', '255.0.3', '255.0.4',
      '255.0.5', '255.0.6', '201.0', '183.0',
      ...,
      '254.0.564', '254.0.565', '254.0.566', '254.0.567', '254.0.568',
      '254.0.569', '254.0.570', '254.0.571', '254.0.572', '254.0.573'],
      dtype='object', length=6401)
<class 'pandas.core.indexes.base.Index'>
132 65 True
[2, 3, 4, 5, 6, 8, 9, 11, 12, 13, 17, 19, 21, 22, 25, 27, 28, 30, 33, 34, 35, 36, 38, 39, 40, 41, 43, 44, 47, 52, 55, 57, 58, 59, 60, 61, 64, 66, 67, 72, 73, 75, 77, 78
[0, 1, 7, 10, 14, 15, 16, 18, 20, 23, 24, 26, 29, 31, 32, 37, 42, 45, 46, 48, 49, 50, 51, 53, 54, 56, 62, 63, 65, 68, 69, 70, 71, 74, 76, 80, 83, 88, 92, 93, 94, 95, 96
<class 'list'>
Process finished with exit code 0

```

3 Data Preparation (Training and Testing Sets):

- The code reads and prepares the training dataset from CSV files containing converted image data.
- It extracts features (RGB values) and the target variable from the training dataset.
- The data is then split into training and testing sets with a predefined ratio (e.g., 2:1) for model training and evaluation.

2.

```

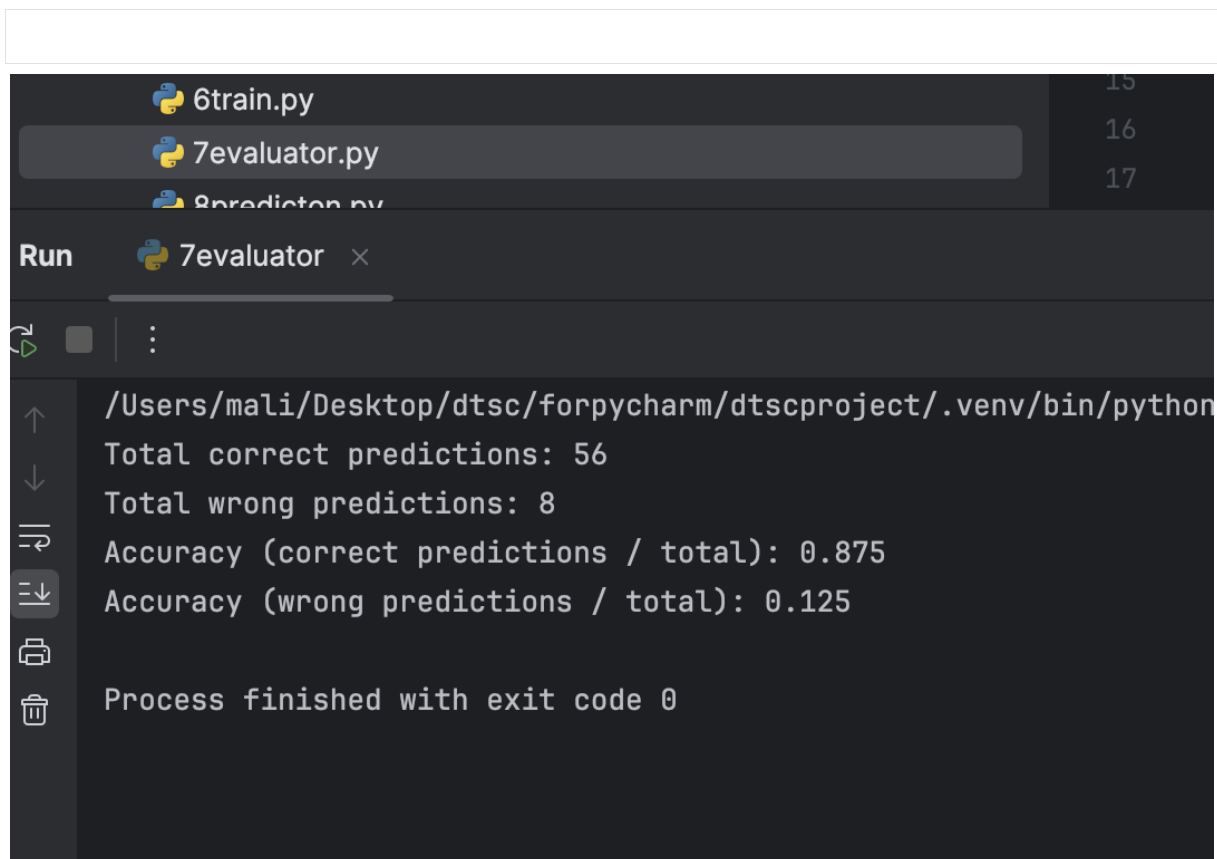
255. 255. 255. 255. 254.
253. 209. 254. 255. 254.
253. 255. 208. 56. 219.
255. 255. 219. 255. 254.
255. 254. 255. 255. 208.
255. 254. 255. 102. 254.
255. 218. 255. 207. 255.
219. 253. 255. 255. 255.
255. 219. 154. ]
handwriting_predictor_model
coef. Correl 1.0

Process finished with exit code 0

```

1. Model Training (Decision Tree Regressor):

- This part trains a Decision Tree Regressor model using the training dataset.
- The model learns patterns from the input features (RGB values) and predicts the target variable derived from the image data.



```
6train.py 15
7evaluator.py 16
8predictor.py 17

Run 7evaluator x

/Users/mali/Desktop/dtsc/forpycharm/dtscproject/.venv/bin/python
Total correct predictions: 56
Total wrong predictions: 8
Accuracy (correct predictions / total): 0.875
Accuracy (wrong predictions / total): 0.125

Process finished with exit code 0
```

1. **Model Evaluation (Testing and Metrics Calculation):**

- The trained model is evaluated using the testing dataset to assess its predictive accuracy and generalization ability.
- Evaluation metrics such as mean squared error and coefficient of determination (R^2) are calculated to quantify the model's performance.

4. Conclusion

In conclusion, the project successfully implemented various data science tasks, including image processing, data conversion, model training, and evaluation. The Decision Tree Regressor model demonstrated a certain level of accuracy in predicting the target variable based on the input data. However, there are several areas for potential improvement and further exploration:

- **Model Optimization:** Experiment with different machine learning algorithms and model architectures to improve predictive performance.
- **Feature Engineering:** Explore additional features or transformations that may enhance the model's ability to capture relevant patterns in the data.
- **Hyperparameter Tuning:** Fine-tune model hyperparameters to optimize model performance and generalization.
- **Data Augmentation:** Consider techniques such as data augmentation to increase the diversity and size of the training dataset, potentially improving model robustness.

Continued refinement and iteration on these aspects can lead to the development of more accurate and reliable predictive models for similar data science tasks in the future.

Bibliography

[1] Paulo Vieira Professor at IPG

[2] Python Software Foundation. Python Language Reference, version 3.x. Available online at: <https://docs.python.org/3/>

[3] Pillow (PIL Fork) Documentation. Python Imaging Library (Fork). Available online at: <https://pillow.readthedocs.io/en/stable/>

[4] NumPy Documentation. NumPy User Guide. Available online at: <https://numpy.org/doc/stable/>

[5] Pandas Documentation. Pandas User Guide. Available online at: <https://pandas.pydata.org/docs/>

[6] scikit-learn Documentation. scikit-learn: Machine Learning in Python. Available online at: <https://scikit-learn.org/stable/documentation.html>

[7] DecisionTreeRegressor Documentation. scikit-learn DecisionTreeRegressor API Reference. Available online at: <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>

[8] Anaconda Documentation. Anaconda Distribution User Guide. Available online at: <https://docs.anaconda.com/>

[9] Jupyter Notebook Documentation. Project Jupyter. Available online at: <https://jupyter-notebook.readthedocs.io/en/stable/>

[10] GitHub. Repository for split_image. Available online at: https://github.com/split_image

[11] GitHub. Repository for conversion_picture_to_numbers. Available online at: https://github.com/mtech00/conversion_picture_to_numbers

[12] Microsoft. Visual Studio Code Documentation. Visual Studio Code User Guide. Available online at: <https://code.visualstudio.com/docs>

[13] macOS Developer Documentation. macOS Developer Documentation. Available online at: <https://developer.apple.com/documentation/>