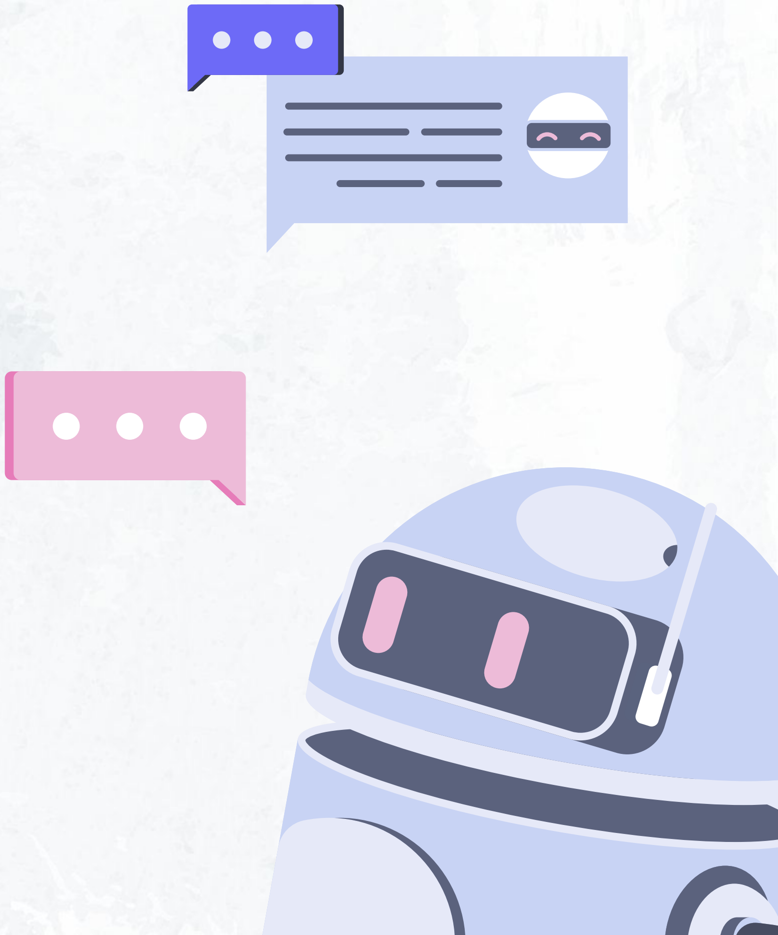
CSD3185 Group Project

Image Classifier

Done by Team 5



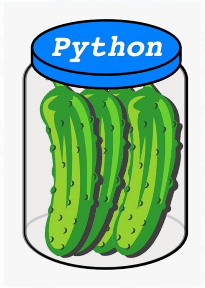
Chiok Wei Wen Gabriel
Yin Shengkai
Jed Goh Yujie
Seow Kai Jun
Dennys Tay Khaj Tjong
Huang Wei Jhin

Introduction

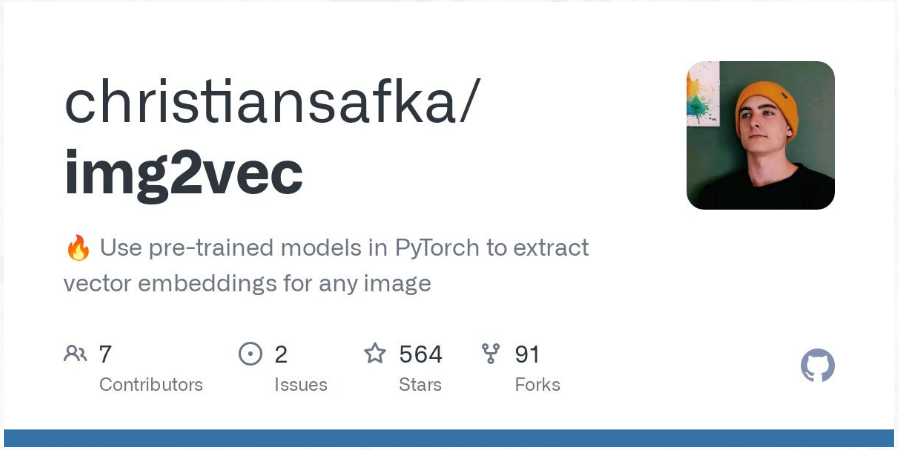
- **Image classifier** is an application to classify images into designated folders automatically
- Utilizes the following libraries:

scikit learn

NumPy



TKINTER
GUI FOR PYTHON



christiansafka/
img2vec

🔥 Use pre-trained models in PyTorch to extract vector embeddings for any image

7 Contributors · 2 Issues · 564 Stars · 91 Forks

Data Collection

Data consists of animal images which are retrieved from Kaggle

Images are split into 4 folders:

- **Training Folder** - used to store images for training
- **Validation Folder** - used to store images for validating models accuracy
- **Testing Folder - **used to test the model classification ability
- **Output Folder** - used to store images based on the model predictions

Models Used

KNN

Appreciated for its simplicity and its direct approach

Random Forest

Used for its robust handling of complex data

SVM

Stands out for its ability to perform well in high dimension spaces

Pre-processing

Resizing

All images were **standardized** to 224x224 pixels without alpha channel. **Streamlined** feature extraction but also **improved** computation times.

Flipping

Added **flipped** versions of image to **augment** our dataset and **enhance** model robustness

Normalisation

Ensures a uniform feature space, facilitating more **accurate** distance measurements and slightly **boosting** the accuracy of models

Brightness and Contrast Adjustment

Modified the brightness and contrast of images across the board which **enhanced** model accuracy

Feature Extraction

Edge Detection

Accentuated the contours within images to spotlight the shapes and boundaries of objects

Grayscale Conversion

Converted images to grayscale to remove color data

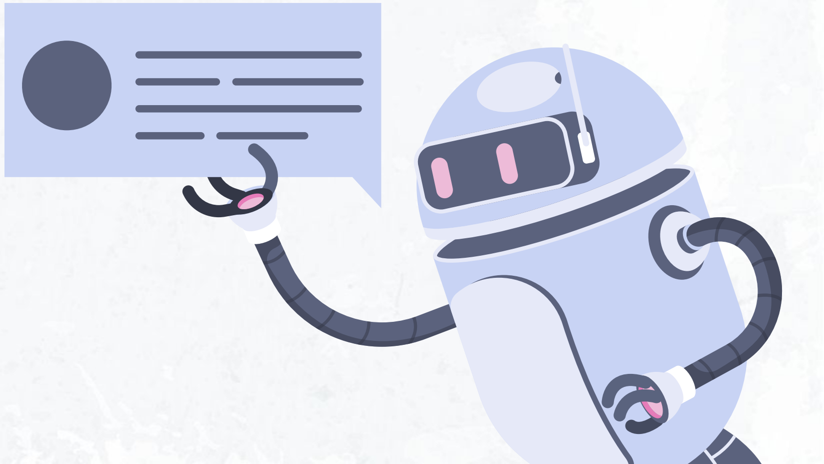
Histogram Equalization

Modified image contrast to spread the most common intensity values

Problems Encountered

Problem	Result	Solution
Bias in predictions towards categories with more training data	Overrepresentation of categories in predictions	Balanced dataset by resampling images and introduced crops and rotations
Images that do not belong to any category were classified into one	Skewed the results	Implement Probability drop-off mechanism

Application Demo



Optimisation Methods

Made use of an **automatic** optimisation method through the following steps:

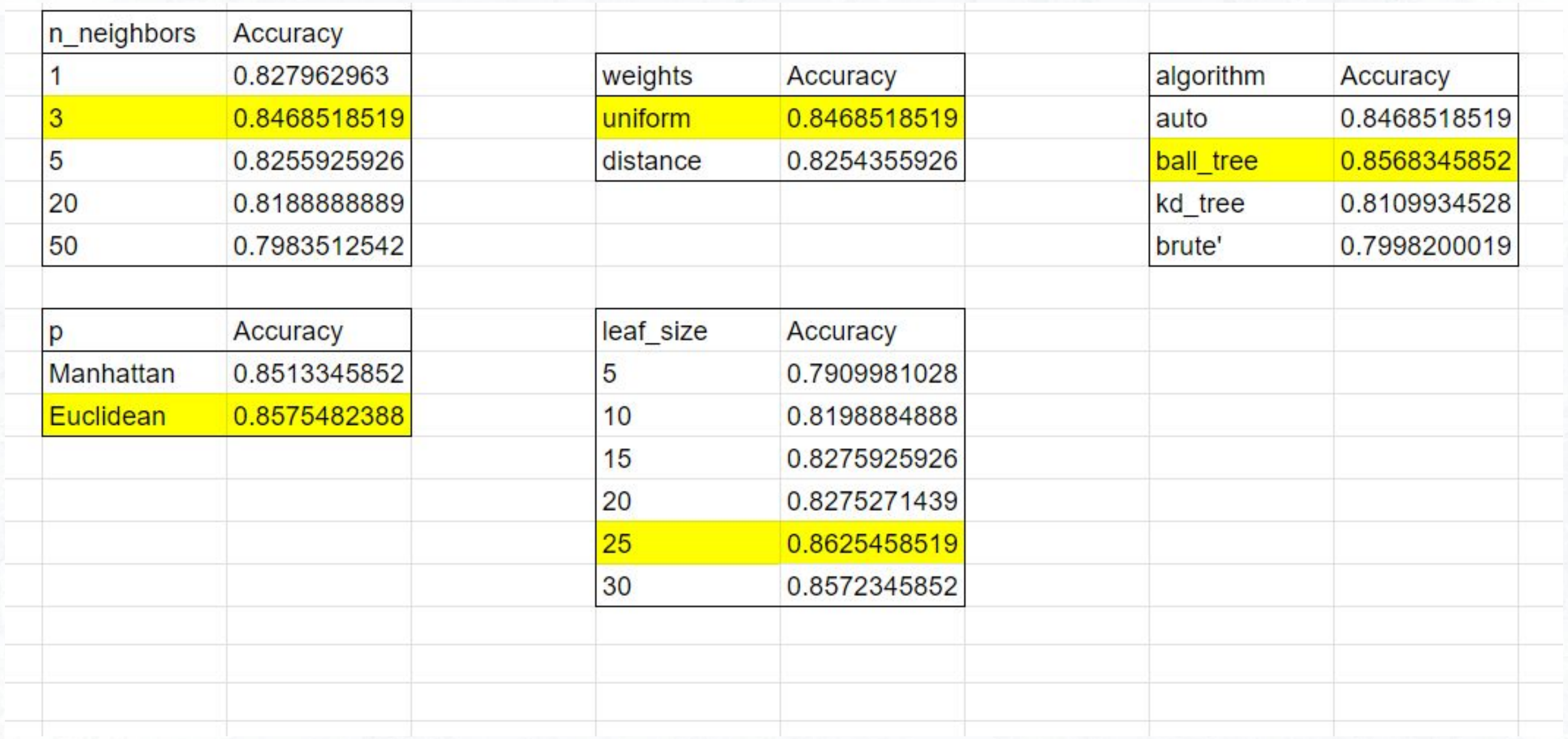
Step 1 Initiate the optimization process by establishing a configuration for each model type	**Step 2** Load the training data and labels, then apply StratifiedKFold, and maintain the percentage of samples for each class.
Step 3 Calculate the average accuracy across all folds for each parameter value introduce a size score	**Step 4** Select the parameter value that results in the highest combined score for each model

Optimisation Methods

How the Optimisation Method Works

Step 1 Split our data into training and validation sets	**Step 2** For settings with predefined choices, test each option one by one
Step 3 For numerical settings, use 3 different settings and repeatedly change their values until the best value is obtained	**Step 4** Set our model based on the best numbers or options to get the best performance

KNN Results



n_neighbors	Accuracy
1	0.827962963
3	0.8468518519
5	0.8255925926
20	0.8188888889
50	0.7983512542

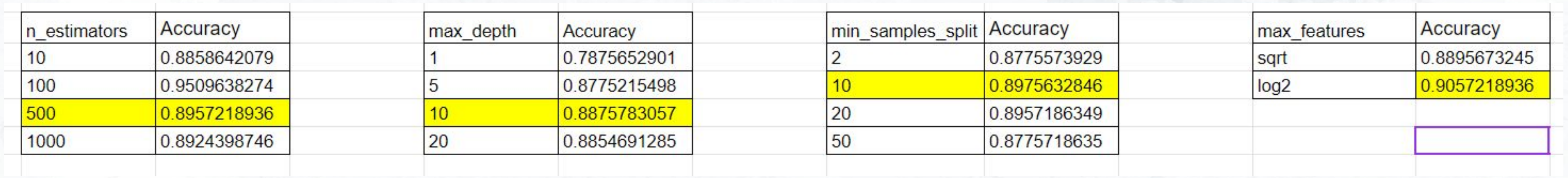
weights	Accuracy
uniform	0.8468518519
distance	0.8254355926

algorithm	Accuracy
auto	0.8468518519
ball_tree	0.8568345852
kd_tree	0.8109934528
brute'	0.7998200019

p	Accuracy
Manhattan	0.8513345852
Euclidean	0.8575482388

leaf_size	Accuracy
5	0.7909981028
10	0.8198884888
15	0.8275925926
20	0.8275271439
25	0.8625458519
30	0.8572345852

Random Forest Results



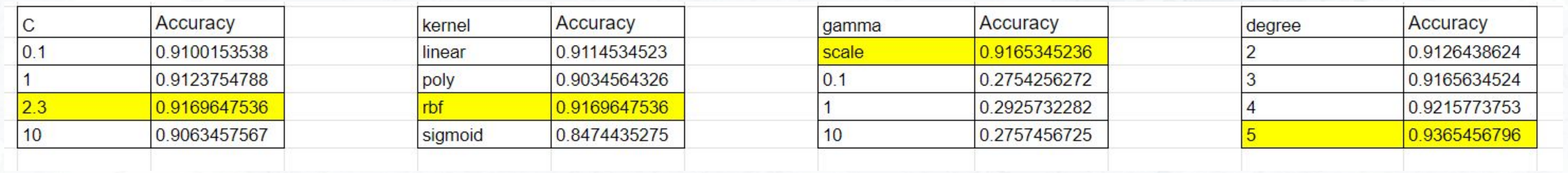
n_estimators	Accuracy
10	0.8858642079
100	0.9509638274
500	0.8957218936
1000	0.8924398746

max_depth	Accuracy
1	0.7875652901
5	0.8775215498
10	0.8875783057
20	0.8854691285

min_samples_split	Accuracy
2	0.8775573929
10	0.8975632846
20	0.8957186349
50	0.8775718635

max_features	Accuracy
sqrt	0.8895673245
log2	0.9057218936

SVM Results



C	Accuracy
0.1	0.9100153538
1	0.9123754788
2.3	0.9169647536
10	0.9063457567

kernel	Accuracy
linear	0.9114534523
poly	0.9034564326
rbf	0.9169647536
sigmoid	0.8474435275

gamma	Accuracy
scale	0.9165345236
0.1	0.2754256272
1	0.2925732282
10	0.2757456725

degree	Accuracy
2	0.9126438624
3	0.9165634524
4	0.9215773753
5	0.9365456796

Why SVM?

High-dimensional Data Handling

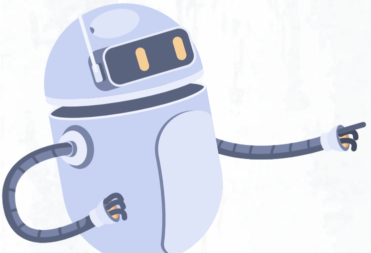
SVM excelled in in managing the image data

Generalization

Results in effective classification

Efficiency in Multi-class Classification

Proved more adaptable and efficient



Thanks! →

Chiok Wei Wen Gabriel
Yin Shengkai
Jed Goh Yujie
Seow Kai Jun
Dennys Tay Khaj Tjong
Huang Wei Jhin