**Reflective Essay: Exploring Image Classification with SVM**

**First Impressions**

When I first encountered the image classification exercise using the Support Vector Machine (SVM) algorithm, I felt both excited and slightly intimidated. I was eager to dive into hands-on image classification using the CIFAR-10 dataset, but there were some challenges. My computer, equipped with a powerful video card and CPU, was more than capable of handling AI training, but the revised assignment lacked the dataset folder. After attempting to download CIFAR-10 from Keras, I found that it required formatting to fit the code in the provided notebook "2024 Image Processing and ML for CV with creating dataset.ipynb." As a result, I chose to work on the original Jupyter notebook, "0924Subset\_Classical\_ML\_Image\_Classification\_with\_CIFAR\_10\_Subset\_of\_Dataset\_.ipynb." I had some prior experience with image classification from two previous projects during a Udacity course: the [**Dog-Breed-Image-Classifier**](https://github.com/Hishamdmacaraya/NanoDegree-AI-Programming/tree/main/Dog-Breed-Image-Classifier) and [**Flower-Image-Classifier**](https://github.com/Hishamdmacaraya/NanoDegree-AI-Programming/tree/main/Flower-Image-Classifier) projects. Additionally, I completed an image classifier project during my machine learning class at HCC. These experiences made me more comfortable with the workflow, though I knew working with SVM on image data would present new challenges.

**Learning Process**

As I worked through the original Jupyter notebook, the process of loading, reshaping, and normalizing the CIFAR-10 subset was familiar thanks to my previous experience. However, when I began applying SVM, I faced some challenges.

Representing image data for SVM was tricky since the algorithm required me to flatten the images into vectors, which felt counterintuitive given the spatial nature of images. Additionally, selecting the right kernel was another hurdle—understanding how the radial basis function (RBF) kernel transforms non-linear data was initially difficult, but the visualization in the notebook helped clarify this concept.

The training process itself took 3 minutes and 8.6 seconds, which my computer handled smoothly due to its high performance. After tuning the hyperparameters, as specified in the markdown, I achieved an accuracy of 0.547. While not perfect, this result was satisfactory given the complexity of the dataset and the limitations of SVM for image classification.

**Challenges and Triumphs**

One of the biggest challenges I faced was working with the dataset, particularly due to the missing dataset folder in the revised notebook. Fortunately, I could use the original file to keep moving forward. Despite this, training the SVM on image data was computationally intensive, but my fully loaded computer managed it efficiently. Overfitting was another issue, especially with the RBF kernel, but using cross-validation and grid search helped optimize the model. The most rewarding moment was seeing the SVM classify images with reasonable accuracy. While SVM is not the most conventional tool for image classification, achieving **0.547 accuracy** after hyperparameter tuning felt like a victory. It reinforced my understanding of how machine learning models learn from data, even when dealing with non-linear and high-dimensional inputs.

**Personal Growth**

This exercise expanded my understanding of image classification and SVMs. My prior work on the Dog-Breed-Image-Classifier and Flower-Image-Classifier projects, both of which utilized Convolutional Neural Networks (CNNs), helped me grasp the basics of image classification. However, applying SVM added a new dimension to my learning, particularly in terms of handling image data that’s represented as feature vectors. The exercise also built on the knowledge I gained in my HCC machine learning class, where I worked on a similar image classification project. The hands-on experience of applying SVM and optimizing it for performance enhanced my skills in hyperparameter tuning, cross-validation, and data preprocessing.

**Looking Ahead**

After completing this exercise, I’m eager to explore more advanced image classification techniques, particularly CNNs, which are better suited for handling spatial data in images. Comparing the performance of SVM and CNN on similar tasks would be a valuable learning experience. In my future studies and career, I see this exercise as a foundational experience that will help me apply machine learning to more complex datasets. I’m also interested in exploring ensemble methods that combine SVM with other algorithms to further improve accuracy. The skills I’ve gained here, such as data handling, optimization, and model evaluation, will be critical for future projects.

**Conclusion**

In conclusion, this exercise was both challenging and rewarding. Despite initial obstacles with the dataset, I was able to successfully train an SVM model with a training time of **3 minutes and 8.6 seconds** and an accuracy rate of **0.547**. The hands-on experience deepened my understanding of image classification and reinforced my confidence in tackling future machine learning tasks. This work, combined with my previous projects, has prepared me for more complex challenges in the field of machine learning and AI.

**References:**

Scikit-learn. (n.d.). Support vector machines. Retrieved from https://scikit-learn.org/stable/modules/svm.html