Intro to AI

CS 3368

Jose Miguel Loguirato, Maria Elena Leizaola, Andres Aguilar

**Fashion MNIST exploration with multiple ML models**

First of all, we began by loading the raw data files and inspecting their structure to understand the type and format of the data. We performed the necessary cleaning steps, such as handling missing values and normalizing pixel values, to make sure the dataset was consistent and ready for analysis. This pre-processing step was crucial to know with accuracy that the data could be effectively used for training machine learning models.

We then conducted an initial exploration of the dataset to understand its characteristics. We calculated the number of examples available for each class to check for class balance and identify any potential biases, we analyzed the distribution of pixel values across the dataset to detect any anomalies or patterns, and we generated centroid images, both overall and per-class, to visualize the central tendencies and typical examples within the dataset, which provided valuable insights for further analysis.

A screenshot of a computer screen

Description automatically generated

As seen above, all the classes in the training data had about 1000 examples each, meaning it is almost completely balanced. This was great news since we knew the dataset was clean and had no biases.

A screen shot of a computer

Description automatically generated

For all of these classes, the distribution of pixels was very similar. Most of the images has 100% white pixels with some distribution of pixels all the way to 100% black. It’s interesting to notice that there are more values at about 0.8 pixel darkness, which makes sense because most of the pixels will be solid or almost solid, with some being “edge” pixel and most being white.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Above we can see the Centroid images of the data. To recall, centroid images are images that represent the mean of a dataset or a specific class. It’s incredible seeing the model decide what an average shoe or t-shirt looks like, and how quickly it can reach this conclusion.

Then, we selected a diverse set of classifiers to evaluate like: Naïve Bayes, Ridge Regression, Softmax Regression and Random Forest. We implemented each classifier using Scikit-Learn and performed a thorough parameter tuning process to identify the optimal settings for each model. Our goal was to maximize the performance of each classifier and determine which one provided the best results for the given dataset.

We tracked the performance of each classifier by maintaining a scorecard.

|  |  |
| --- | --- |
| METHOD | ACCURACY |
| NAÏVE BAYES | 0.490 |
| RIDGE REGRESSION | 0.811 |
| SOFTMAX REGRESSION | 0.842 |
| RANDOM FOREST | 0.870 |

We extracted a set of images from the test dataset that were incorrectly classified by the best-performing model, which was Random Forest. We analyzed these mis-classified examples to understand the potential reasons behind the classifier's errors. This analysis offered valuable insights for improving the model's performance and addressing specific challenges in the dataset.

A screenshot of a computer

Description automatically generated

Even if our model is very good (over 80% accuracy), it will always misclassify some data, but it is interesting to see that the images that look the same to the human eye also fail “correctly” with our model. What we mean is that images like sweaters and long sleeved shirts look very similar and thus it makes sense that our model gets confused. Also, sneakers and shoes get mixed up. A model that has those types of misclassifications and a 87% accuracy is a pretty solid model.

A screenshot of a computer

Description automatically generated

We then investigated the importance of individual pixels in the decision-making process of the best-performing classifier. We visualized these importance scores as a heatmap or similar image, highlighting the areas of the input data that had the most significant impact on the classifier's predictions. This visualization helped us understand the model's focus and potentially uncover areas for improvement in the dataset or model.

A screen shot of a computer

Description automatically generated

Lastly, we assessed how varying the image resolution impacted the performance of the best classifier. We resized the images to different resolutions, such as 28x28, 10x10, 8x8, and 6x6 and retrained the classifier on these modified datasets. As we can tell from this graph and the results in the file, as the resolution got worse the actual accuracy of the predictor didn’t really shift, it was higher. On the other hand, the precision definitely did decrease proportionally to the resolution.

This project was a really eye opening experience on our whole group. This project helped us realize how fast and simple it really is training effective models when the data is correct. We realized that most of the work done to training machine learning models is actually in the data and how you handle that.