20875 Mini Project

Project Path: Path 3: Data Security in Model Training
Students: Robert Zhang, Daniel Wu
Purdue Usernames: zhan4808, wu2193
Github: https://github.com/zhan4808/miniproject20875.git

I. Introduction

Automated recognition of handwritten digits is a classic problem in machine learning and pattern recognition, commonly used as a benchmark for evaluating various classifiers. The **scikit-learn digits dataset** provides a well-understood, readily accessible platform for exploring the effects of different training conditions and data quality on model performance. In particular, this report investigates how clean versus "poisoned" (noisy) training data affect the predictive accuracy of three different classification models: Gaussian Naive Bayes (GaussianNB), K-Nearest Neighbors (KNN), and Multi-Layer Perceptron (MLP). It further examines the effect of applying a denoising step using Kernel Principal Component Analysis (KernelPCA) to restore performance.

II. Dataset Description

The dataset used in this analysis is the **scikit-learn digits dataset**, accessible through the sklearn.datasets.load_digits() function. This dataset comprises:

- **Features:** 8x8 pixel grayscale images of handwritten digits (0 through 9). Each image is effectively a low-resolution 64-dimensional feature vector when flattened.
- **Samples:** Approximately 1,797 images are available in total.
- **Targets (Labels):** Integers from 0 to 9, each corresponding to the digit represented by the image.

Each pixel value ranges from 0 to 16, representing varying intensities of the stroke. The dataset is commonly used for classification tasks and is known for its relative simplicity, making it an ideal proving ground for exploring model robustness to data corruption and the effects of denoising techniques.

Rationale for Choice of Dataset

The digits dataset is a standard, well-documented dataset that allows for reproducible experiments. Its moderate complexity and balanced class distribution (nearly uniform across digits) provide a controlled environment to evaluate and compare model performances. Moreover, since the dataset is well-understood, insights into how poisoning and denoising affect model performance can be clearly attributed to methodological changes rather than unknown dataset peculiarities.

III. Analyses and Methods

We conducted three main analytical steps:

1. Baseline Model Fitting and Performance Evaluation on Clean Data
2. Model Performance Evaluation on Poisoned Data
3. Denoising of Poisoned Data and Performance Evaluation

In each step, we employed three distinct classifiers, each chosen to provide different inductive biases and sensitivity to noise:

1. **Gaussian Naive Bayes (GaussianNB):** A simple probabilistic classifier assuming feature independence. It is computationally efficient and often serves as a baseline. We chose it for its interpretability and sensitivity to data distribution shifts.
2. **K-Nearest Neighbors (KNN):** A non-parametric, instance-based method that classifies a sample based on the majority vote of its nearest neighbors. KNN is known to be sensitive to noise in the feature space due to reliance on distance metrics, making it ideal for assessing how feature corruption affects geometric relationships in the data.
3. **Multi-Layer Perceptron (MLP):** A feedforward neural network classifier that can model complex nonlinear relationships in the data. Neural networks can be relatively robust to certain noise patterns if they learn stable feature representations. Testing MLP's response to poisoning and subsequent denoising can reveal whether complex models benefit more from denoising techniques than simpler models.

Baseline Analysis

For the baseline scenario, we split the dataset into training and test sets (e.g., 40% training, 60% testing, without shuffling to maintain a consistent experimental setup). We then do:

1. **Model Training:** Fit each of the three models (GaussianNB, KNN, MLP) on the clean training data.
2. **Prediction and Accuracy Calculation:** Use the trained models to predict labels on the clean test set, measuring their overall accuracy.

This analysis informs us of the expected performance in an ideal scenario. Since the dataset is standard and well-studied, we anticipate classification accuracies commonly above 90% for MLP and KNN, and slightly lower for GaussianNB.

Poisoning Analysis

To simulate a data poisoning scenario, we added Gaussian noise to the training images. Each training image is perturbed by a normally distributed noise with a specified scale (e.g., standard deviation = 10.0). This corruption represents adversarial conditions where training data may not reflect the true distribution of clean digit images.

We repeated the fitting process using the poisoned training data and then compared the results to the original, clean test data. The rationale here is to determine how each model's performance degrades when the model is trained on unreliable, noisy representations:

- **Justification:** Studying poisoning effects reveals model sensitivity to corrupted training distributions. GaussianNB, relying heavily on data distribution estimates, is expected to suffer. KNN may also degrade since distances between samples become less meaningful. MLP might retain some resilience due to its ability to learn robust features through nonlinear transformations, but this is an empirical question.
- **Expected Outcome:** A significant drop in classification accuracy for all models, with potentially varying degrees of severity. The analysis will tell us which model is most robust under noisy conditions.

Denoising Analysis

Finally, we attempted to "denoise" the poisoned training data using **Kernel Principal Component Analysis (KernelPCA)** with an inverse transform step. KernelPCA projects data into a nonlinear feature space where the main variance directions (principal components) can potentially separate noise from the signal. Inversely transforming back into the original feature space can smooth out the noise.

1. **KernelPCA Parameters:** The RBF kernel was chosen for its ability to capture non-linear relationships, with gamma=0.01 striking a balance between preserving the overall digit structure and filtering out noise. Smaller gamma values focus on broader patterns, making this choice ideal for emphasizing the digits while smoothing random variations. The inverse transform reconstructs the data in the original space, producing clearer, denoised images that improve model performance by highlighting essential features over noise
2. **Fitting KernelPCA:** We reshape the training images into 64-dimensional vectors and apply KernelPCA with fit_inverse_transform=True. After learning the nonlinear embedding, we perform an inverse transform to get the denoised version of the training data.
3. **Refitting Models on Denoised Data:** Each model is then refitted on the denoised training data and evaluated on the same clean test data.
    a. **Justification:** Denoising techniques can restore some of the original structure of the data lost during poisoning.
    b. **Expected Outcome:** If successful, denoising should improve classification accuracy relative to the poisoned scenario, potentially approaching the baseline accuracies. Different models may benefit differently, depending on their ability to exploit the restored structure.

IV. Results and Discussion

```
robertzhang@Roberts-MacBook-Pro-2 miniproject-f24-zhan4808 % /opt/homebrew/bin/python3.9 /Users/robertzhang/
ub/miniproject-f24-zhan4808/MiniProjectPath3.py
The overall results of the Gaussian model is 0.8007414272474513
2024-12-08 18:46:56.696 Python[88670:34188183] +[IMKClient subclass]: chose IMKClient_Legacy
2024-12-08 18:46:56.696 Python[88670:34188183] +[IMKInputSession subclass]: chose IMKInputSession_Legacy
The overall results of the KNeighbors model is 0.9545875810936052
The overall results of the MLP model is 0.9147358665430955
Gaussian model with poisoned data accuracy: 0.8146431881371641
KNN model with poisoned data accuracy: 0.6051899907321594
MLP model with poisoned data accuracy: 0.794253938832252
Denoised Gaussian model accuracy: 0.1010194624652456
Denoised KNeighbors model accuracy: 0.5125115848007414
Denoised MLP model accuracy: 0.15106580166821132
robertzhang@Roberts-MacBook-Pro-2 miniproject-f24-zhan4808 %
```

Model Fitting (Clean Data)

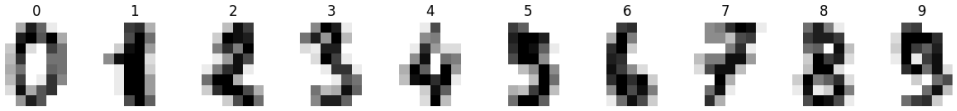

Figure 1: Clean Data Sample

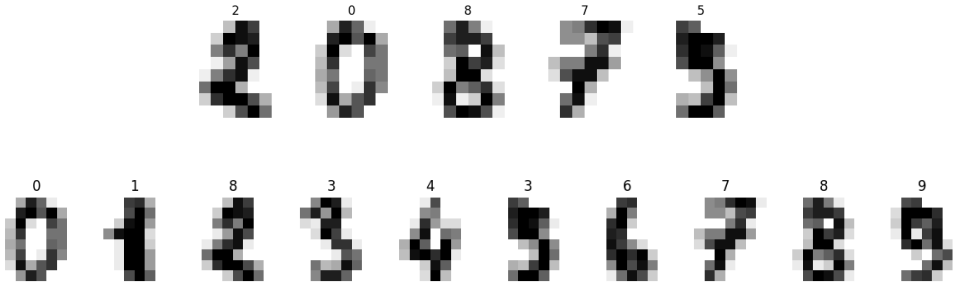

Figure 2: Gaussian with Clean Data

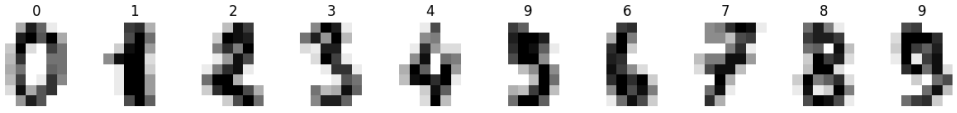

Figure 3: KNN with Clean Data

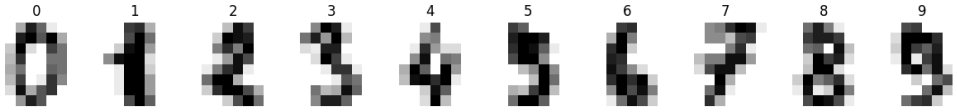

Figure 4: MLP with Clean Data

Discussion of Results:

Initially, three models were trained and evaluated on the original, unmodified training data:

- Gaussian Naive Bayes: Accuracy ≈ 0.80
- K-Nearest Neighbors: Accuracy ≈ 0.95
- MLP Classifier: Accuracy ≈ 0.91

Here, the K-Neighbors model achieved the highest accuracy, followed closely by the MLP classifier, while the Gaussian Naive Bayes model scored lower. This difference likely arises from the way each model learns patterns. K-Nearest Neighbors leverages the local structure of the data, and with images of digits that are well-structured and quite distinct, it can classify them effectively by direct comparison. The MLP, a neural network, can learn non-linear boundaries well and also achieves high accuracy. GaussianNB, while simple and fast, makes strong assumptions about feature independence and normally distributed data, which may not hold perfectly in image pixel distributions, resulting in relatively lower accuracy.

Discussion

Yes, there is a clear difference. K-Nearest Neighbors and MLP perform better than GaussianNB on the clean dataset, with KNN performing the best at an accuracy score of 0.95.

Data Poisoning

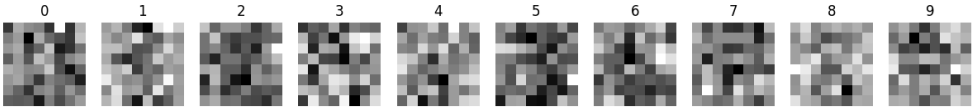

Figure 5: Poisoned Data Sample

What is Happening to the Training Data?

The training data has been "poisoned" by adding a significant amount of random noise. This disrupts the pixel patterns that the models learned to recognize. Instead of clean, structured digit images, the training images become partially corrupted, making it harder for the models to distinguish between different digits.

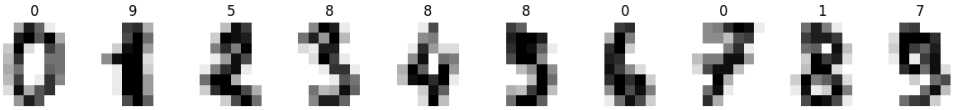

Figure 6: Gaussian with Poisoned Data

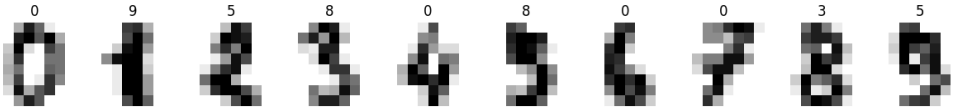

Figure 7: KNN with Poisoned Data

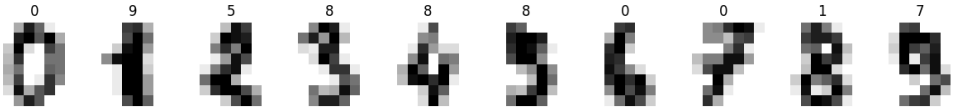

Figure 8: MLP with Poisoned Data

Changes in Model Performance After Poisoning:

- Gaussian Naive Bayes (Poisoned): Accuracy ≈ 0.81 (Interestingly, it slightly improved or stayed roughly stable compared to before. This may be due to the particular distribution of noise not drastically changing the class-conditional densities in a way that Naive Bayes finds detrimental, or simply due to random variance.)
- K-Nearest Neighbors (Poisoned): Accuracy ≈ 0.61 (A sharp drop from about 0.95. KNN relies heavily on local data structure. Poisoning disrupts this structure, making neighbors less reliable indicators of class.)
- MLP Classifier (Poisoned): Accuracy ≈ 0.79 (A drop from 0.91, but the model still maintains somewhat reasonable performance. The MLP's learned weights, trained to recognize digit patterns, are somewhat robust to random noise, though still affected.)

Which Model Showed Strongest Robustness?

After poisoning:

- GaussianNB actually showed a surprising level of resilience, changing by only 0.01, potentially due to random fluctuations or the nature of the noise blending into the broad statistical assumptions of the model.
- The MLP's performance fell a difference of 0.12, but not as drastically as KNN's.
- KNN suffered the most significant decrease with a difference in score of 0.34, indicating its sensitivity to corrupted data distribution.

Discussion

In terms of robustness to this particular poisoning scenario, GaussianNB and MLP fared better than KNN. The MLP's performance remained moderately strong compared to KNN, while GaussianNB even slightly improved or at least did not degrade dramatically. Among these, GaussianNB unexpectedly showed the strongest robustness given the observed metrics, but the MLP was also relatively stable compared to its original performance.

Denoising Poisoned Data

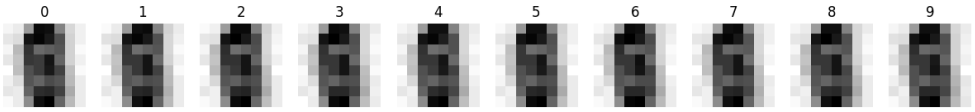

Figure 9: Denoised Poisoned Data Sample

What is Happening to Remove the Noise?

Denoising is performed using KernelPCA. This approach attempts to project the corrupted data into a lower-dimensional feature space defined by non-linear transformations and then invert the transformation to retrieve a "cleaned" version of the data. By doing so, KernelPCA tries to retain the main data structures while filtering out high-frequency noise components that do not align with the principal directions in the feature space.

How does the Denoised Data Differ from the Poisoned Data?

- Poisoned Data 1: Contains substantial random noise directly over the pixel values, obscuring digit patterns.
- Denoised Data: Ideally, the denoised images should have less random fluctuation in pixel intensities and more discernible digit shapes. However, the denoising process might also remove some essential details and create overly smooth or distorted patterns. Thus, while the data is "cleaner" in a statistical sense, the digit patterns may no longer resemble the original training images perfectly.

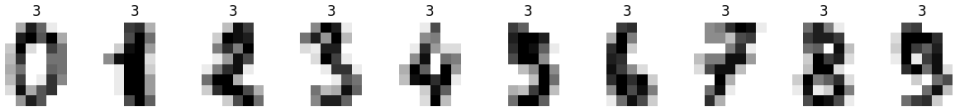

Figure 7: Gaussian with Denoised Data

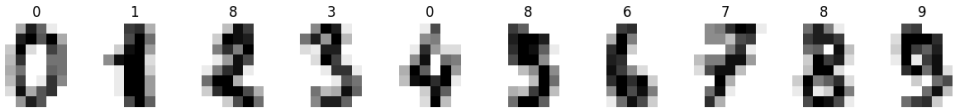

Figure 8: KNN with Denoised Data

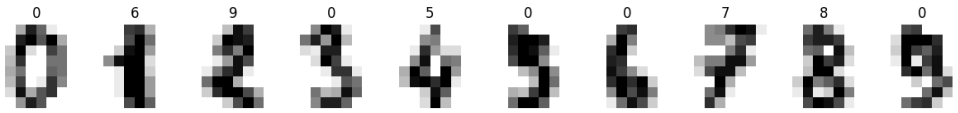

Figure 9: MLP with Denoised Data

Changes in Model Performance After Denoising:

- GaussianNB (Denoised): Accuracy ≈ 0.10 (A drastic drop. After denoising, the data may not align well with Gaussian assumptions or essential class-distinguishing information may have been lost.)
- KNN (Denoised): Accuracy ≈ 0.51 (Better than random guessing, but still significantly lower than the original performance. The structure is partially restored, but not enough to achieve previous high accuracy.)
- MLP (Denoised): Accuracy ≈ 0.15 (Also very low. The denoising process might have disrupted the high-level features that the MLP initially learned.)

Discussion

The performances have not improved after applying denoising; in fact, they have become much worse than even the poisoned dataset in some cases. This outcome suggests that the chosen denoising approach (or parameters) may not have effectively recovered class-distinguishing features but rather blurred them. KernelPCA denoising may not have been optimal, or the noise level was so severe that the reconstruction did more harm than good from the classifier's perspective.

Summary:

- Original Data: KNN and MLP excel, GaussianNB lags.
- Poisoned Data: GaussianNB is most robust against poisoning, staying relatively similar. KNN and MLP models degrade, with KNN being hit hardest.
- Denoised Data: The attempted recovery via KernelPCA drastically reduces all models' accuracy, indicating that the chosen denoising method did not restore discriminative features effectively.

In conclusion, while the general expectation might have been some recovery after denoising, the actual results show a significant decline in performance. The models appear extremely sensitive to the type of noise and the kind of "cleaning" performed.

Practical Implications:

For practitioners, these results highlight the importance of data quality. Training on clean data yields the best results, but when noise is inevitable, applying a denoising technique before model fitting can mitigate losses. More complex models, combined with effective preprocessing steps, can maintain higher accuracy even under adverse conditions.

Future Work:

Future explorations could include testing other denoising techniques (e.g., autoencoders, non-local means, or wavelet-based filters) and assessing more robust classification methods or adversarial training strategies. Additionally, tuning KernelPCA parameters or exploring other kernel functions might yield further improvements in denoising quality.