EVALUATING MULTINOMIAL LOGISTIC REGRESSION AND SUPPORT VECTOR MACHINE IN HANDWRITTEN DIGIT RECOGNITION

Stylianos Psara 2140527

Panagiotis Andrikopoulos 1780743

Christos Papageorgiou 9114343

December 8, 2023

ABSTRACT

Advancing the accuracy of Handwritten digit recognition proves to be a crucial task, particularly within the domain of document analysis. By applying machine learning algorithms, we can provide a better understanding of handwritten texts, facilitating the automation of tasks that previously required human supervision. In this project we aim to classify handwritten digits ranging from 0 to 9. Our approach involves the application of a multinomial logistic regression based on two distinct features both individually and combined. Moreover, we evaluate and compared the performances of both multinomial logistic regression and support vector machine models based on the raw pixels extracted from our datasets. Our experiments seek to assess each feature's contribution to digit recognition and the effectiveness of the applied models.

1 Introduction

Handwritten digit recognition (HDR) is a key aspect of various applications, such as optical character recognition (OCR) which plays a vital role in information extraction from images and scanned documents [1]. With the advent of machine learning, HDR has seen substantial improvements in both accuracy and efficiency. This paper aims to explore the use of two machine learning classifiers, logistic regression, and support vector machines (SVM), for classifying handwritten digits in the range of 0 to 9.

Logistic regression and SVM are popular machine learning classifiers that have been widely used in various applications, including HDR. Logistic regression is a simple and efficient method for classification problems, while SVM is a powerful technique for handling both linear and non-linear relationships between data points.

The rest of the paper is organized as follows. In the "Data" section, we will describe the dataset used for the experiment and some preprocessing techniques that was applied. In The "Classification" section, we will present the implementation details and the methodology used for the logistic regression and SVM classifiers. Moreover the classification performance of logistic regression using different features will be analyzed, and a comparison between logistic regression and support vector machines will be discussed. Next, a statistical test will be performed in order to discover if there is a significant statistical difference between SVM and Logistic regression classifiers.In the "Conclusion and Discussion" section, we will summarize the findings of our experiment and we will discuss the observed differences in performance between logistic regressions and Support vector machines.In the last section "Implementation tools" the tools and libraries that are used will be mentioned.

2 Data

The dataset for this project was claimed from the MNIST database, a widely recognized benchmark in machine learning for handwritten digit recognition. It consists of 42000 rows and 785 columns. Each row represents a grayscale image of hand-drawn digits ranging from 0 to 9 with 28 pixels height and 28 pixels width. The first column provides information about the class of each observation and the rest, the pixel-values of the associated image. Each pixel represents lightness

or darkness with an integer between 0 and 255, inclusive. The smaller pixel value corresponds to a lighter shade and the larger to a darker shade.

2.1 Useless variables

After summing the values of each pixel for all the observations together, we located the “useless” pixels. A pixel is called “useless” if it has the value 0 in all the observations, that means that it doesn’t contribute any information for the images in the dataset. Excluding the “useless” pixels would potentially result in reduced complexity of the dataset.

In Figure 1 we can see, with black color, all the 76 “useless” pixels. As we can observe the pixels are on the border of the image which lead us to conclude that the digits must be centralized. Other than this, the figure shows the nonexistence of a constant "useless" frame of pixels that can be evenly cropped off all images to reduce dimensionality.

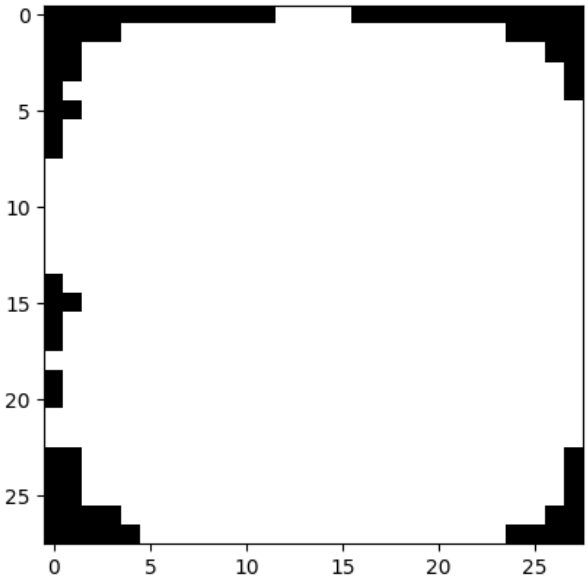


Figure 1: A plot of the pixels that are always empty in all 42,000 images of the dataset (black) and those who experience ink in at least one digit (white).

2.2 Majority class prediction

In pattern recognition, establishing a baseline performance is crucial. One such baseline is the ’majority class prediction,’ where the most frequently occurring class in the dataset is used as the prediction for all instances. Table 1 presents the distribution of each digit in our dataset. The digit ’1’ appears as the majority class. However, the class distribution is relatively balanced, with the majority class encompassing only 11.2% of the data. Consequently, a model predicting solely based on the majority class would achieve an accuracy of 11.2%, barely above random chance. This sets a fundamental benchmark for the performance of more sophisticated models.

	0	1	2	3	4	5	6	7	8	9	Total
Observations	4132	4684	4177	4351	4072	3795	4137	4401	4063	4188	**42000**
Percentage	9.8%	11.2%	9.9%	10.4%	9.7%	9.0%	9.8%	10.5%	9.7%	10%	**100%**

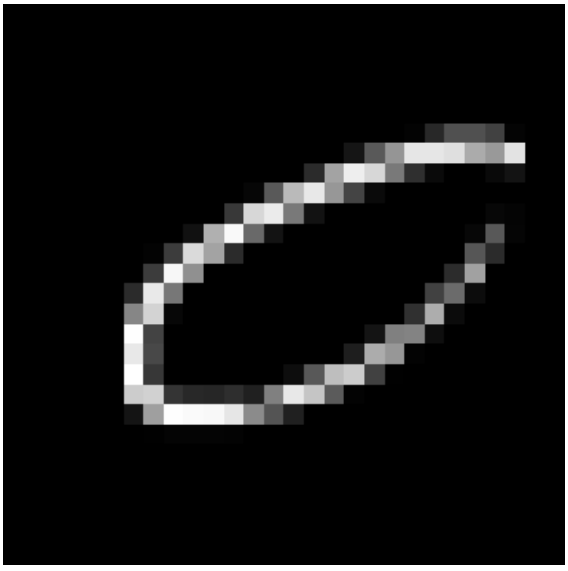
Table 1: Number and percentage of observations for each unique digit.

3 Classification

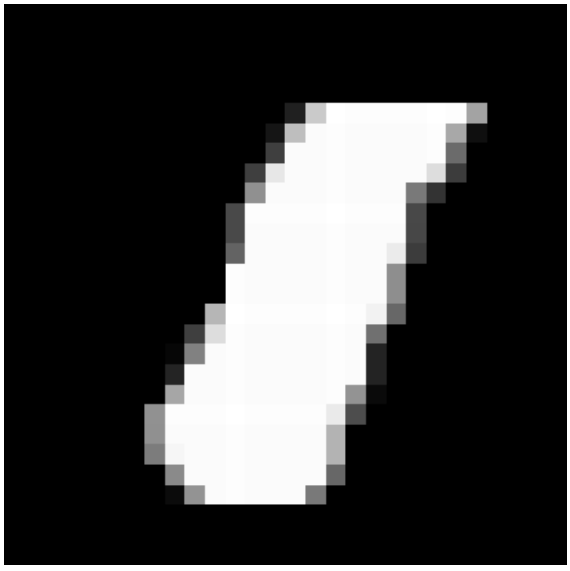
3.1 Ink feature

The ink feature refers to the sum of the values of all pixels in an image. This is the simplest feature to extract from image data with just an iterative sum function across all entries. Disregarding its non-complex nature, ink offers an

important insight on the variety of our data. We calculated the mean ink value of each class of digit, to get an estimate about their 'mass', as well as the standard deviation, that indicates how each instance within one class can vary. Table 2, displays the results of these two metrics. It is obvious from the table that the digit with the highest mean value of ink is '0' (34632.41), whereas '1' presents a notably low value (15188.47), less than half of the former digit. Due to this difference it will be easier to distinguish their classes, however, we can still observe some odd cases where the ink value of a digit '1' is almost quadruple the ink value of a digit '0' (Figure 2). Moreover, there are interesting occurrences of digits, for instance '4' and '9', that have almost identical mean values for ink (24232.72 and 24553.75 respectively). In this case the feature ink does not provide any useful information, which makes it hard to distinguish their class. Standard deviation appears consistent with the mean values, meaning that digits with low levels of ink are characterized by lower deviation and digits with higher values of ink count higher deviation, maintaining a stable ratio, similar to the other metric.



(a) Image of digit "0" that uses 11,994 of total ink.



(b) Image of digit "1" that uses 43,682 of total ink.

Figure 2: Odd cases of the digits "0" and "1". Specifically the thinnest "0" and the thickest "1" included in the dataset.

	0	1	2	3	4	5	6	7	8	9
Mean	34632.41	15188.47	29871.10	28320.19	24232.72	25835.92	27734.92	22931.24	30184.15	24553.75
St.d.	8461.89	4409.46	7653.01	7574.10	6374.63	7526.60	7530.50	6168.34	7777.40	6465.231

Table 2: Mean value and standard deviation of ink for each unique digit of the dataset.

3.1.1 Classification using ink

For classification using solely the ink feature, we employed a multinomial logistic regression model. We additionally added grid search with cross-validation, in order to classify the images based on their amount of ink. We tested various 'C' values, particularly those between 0 and 10 with step value 0.1 including 100, to explore the impact of diverse hyperparameters on model accuracy. The leave-one-out cross-validation process was specified with 'cv=5' dividing the dataset into five parts, training it on the four and validating it on the remaining one. After training, the best hyperparameter value and the best cross-validation score were extracted. This procedure identified C=2.3 as the optimal hyperparameter, with the maximum accuracy score of 22.7% (0.2268), which is double than the accuracy resulting from predicting the majority class. A new multinomial logistic regression model was trained with this optimal value of 'C', resulting to the confusion matrix seen on Figure 3.

The digit '1' stands out as being classified most accurately, evidenced by the highest diagonal count (3823) for label '1'. This could be because the digit '1' generally requires less ink compared to other digits, making it particularly distinguishable among the others. Similarly, '0' is also classified with considerable accuracy, with 2420 correct classifications. The round shape of '0', in most cases leads to a consistent amount of ink that aids in its recognition. Exceptions for these two digits are discussed in section Ink feature. Digits such as '2', '3', and '9' show notable confusion with several other digits. For example, '2' and '3' are often misclassified as '0' and '9' is often missclassified

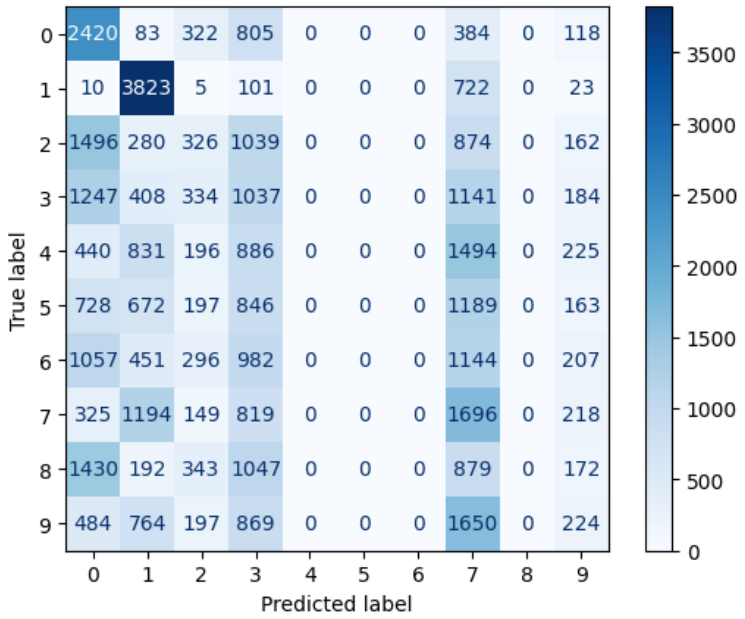


Figure 3: Confusion Matrix for the results of the multinomial logistic regression using the "ink" feature.

as ’7’. It appears that the model never predicts ’4’, ’5’, ’6’ or ’8’ as indicated by their columns being entirely zeroes. This suggests that the model is unable to distinguish these digits based on the ink feature alone. The aforementioned issues, both missclassifications and the lack of them, can be interpreted by observing the mean ink values on Figure 4. Digits ’4’, ’5’ and ’9’ have similar mean value of ink with ’7’, something that can be seen in the confusion matrix where the former three are heavily missclassified as the latter. It appears that the model formed four strong classification boundaries: digits with ink value close digit ’0’ are classified as ’0’, digits with ink value close to digit ’3’ are classified as ’3’, those with ink values close to digit ’7’ are classified as ’7’ and the last one being the digit ’1’ which is significantly recognizable and was classified correctly most of the time.

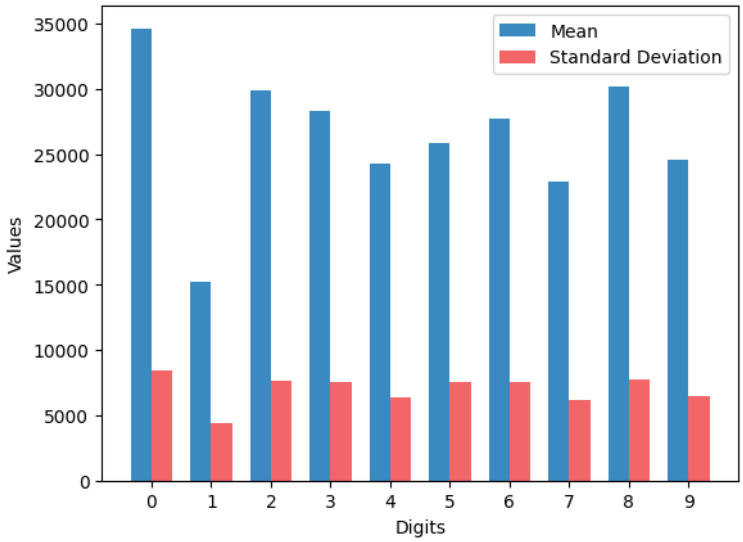


Figure 4: Bar graph of the mean (blue) and standard deviation (red) values regarding the ink feature of each digit.

3.2 Center mass feature

As a second feature, we used the 'center mass' of each image. The center mass, is a point representing the average position of all the weighted pixel intensities in the image. This provides a measure of where the 'mass' or information in the image is concentrated.

The digits data, originally in a shape of (42000, 784), is reshaped to (42000, 28, 28). This is done to convert each digit from a flat array of 784 pixels into a 2D array of 28x28 pixels. This reshaping is essential to process the image in a two-dimensional spatial context, which is required for calculating the center of mass.
We created a grid of coordinates for each pixel in the 28x28 image, using the Numpy function *np.meshgrid*. This grid was used to determine the distance of each pixel from the center of the image. For a 28x28 image, with each dimension ranging from 0 to 27, the center was calculated as (13.5,13.5). Then, we computed the Euclidean distance of each pixel from this center point. Subsequently, we multiplied each image pixel with the distance grid using element-by-element multiplication and summed these values across the entire image. The idea is to give more weight to pixels that are farther from the center, since it is a less common area to appear, hence a better feature for classification purposes. The weighted sums were scaled to have a mean of 0, in order to be used more effectively in logistic regression.

3.2.1 Classification using mass

Similarly with the ink feature, we employed a multinomial logistic regression model, enhanced by a grid search with cross-validation, to classify data points based on the center of mass feature. The same hyperparameters were used for 'C' (0 to 10 with step value 0.1 and 100). Leave-one out cross-validation process was set out, again with 'cv=5' . This time, the search indicated C=0.7 as the best hyperparameter, with the highest accuracy score being 25.3% (0.253), which is again more than twice the accuracy of the majority class.
Then, we trained a multinomial logistic regression model using the best 'C' value and plotted the confusion matrix seen on Figure 5.

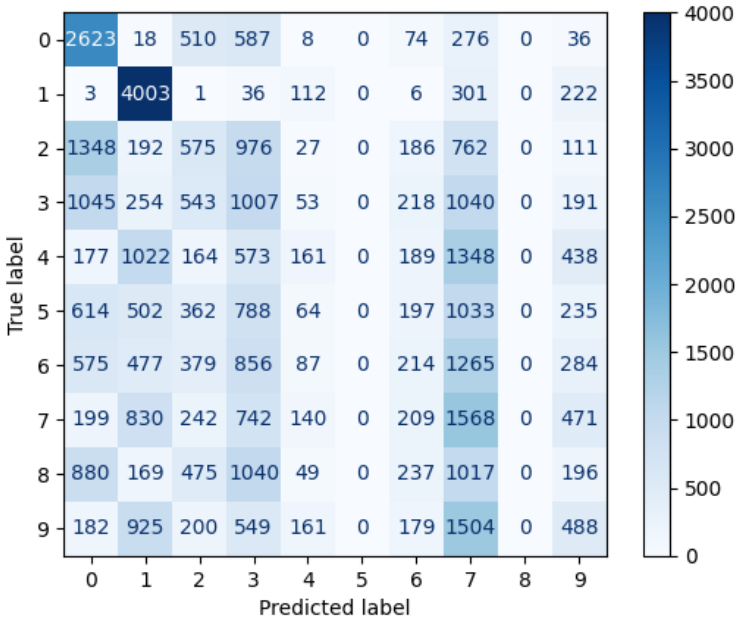


Figure 5: Confusion Matrix for the results of the multinomial logistic regression using the "center mass" feature.

It can be seen from the matrix that the strongest diagonal value is for the digit '1', which suggests that this model is highly accurate in predicting the digit '1'. This could be due to the distinctive center mass that '1' typically has, which is different from other digits. Additionally, the matrix indicates that the digit '0' has been classified with a relatively high degree of accuracy. This suggests that the center mass feature is particularly effective for recognizing this digit, which could be attributed to its circular and usually fairly symmetrical shape that extends equally to all the directions on an image. There are noticeable off-diagonal numbers, such as for digits '0' and '2', '0' and '3' and '7' and '9'. These suggest common misclassifications, likely because the center mass feature does not sufficiently distinguish between these digits' shapes. The model does not perform well for digits '5' and '8' that can be noticed from their completely

empty columns. Digit '5' is often missclassified as '7', '3' or '0', likely due to similar distribution of mass across the image. Likewise, the model confuses the digit '8' with '3', because of similarities in the shape and center mass distribution.
The significant number of misclassifications indicates that while the center mass feature may provide some useful information for classification, it is not sufficient on its own to achieve high accuracy across all digits.

3.3 Classification using Combination of features

We combined the ink and center-mass features and created an array that includes both for every image on the dataset. Now, unlike the previous two attempts, the new model contains information from two independent features, aiming to achieve a stronger understanding of the data. We trained the model using logistic regression and a grid search with cross-validation, applying the same hyperparameters as on the previous models. The search found C=2.1 as the best hyperparameter, with score 32.74% (0.3274).
Additionally, we trained a multinomial logistic regression model using the best C, provided by the grid search and plotted the confusion matrix seen on Figure 6. The model predicts the digit '1' with great accuracy, as indicated by the high number on the diagonal for the label '1'. This is often the case because the digit '1' has a distinctive ink value and distance from the center. The digits '5' and '8' seem to be confused with other digits more frequently than others. This can be seen from the off-diagonal numbers in their respective rows and columns, which indicates a higher number of misclassifications. The digit '7' is frequently misclassified as '1', which occurs due to the the fact that both of them have low average ink values and usually similar distribution of ink across the image.

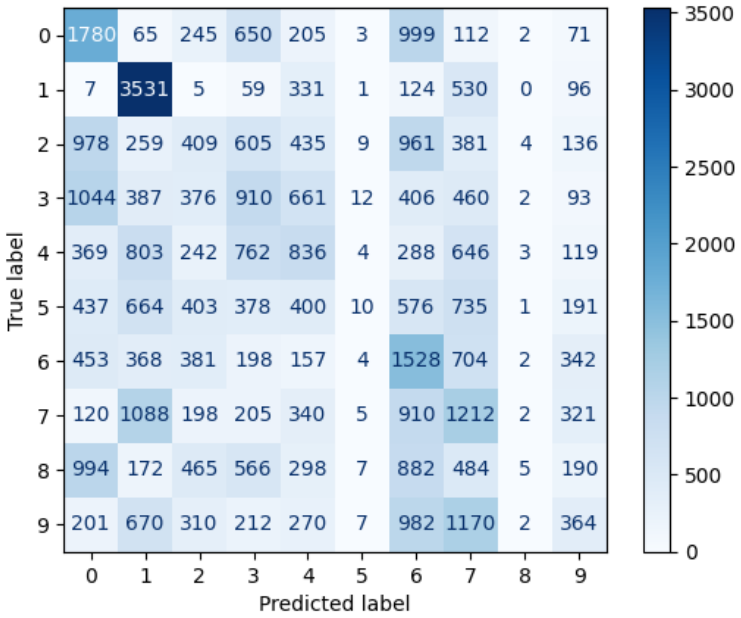


Figure 6: Confusion Matrix for the results of the multinomial logistic regression using the combination of 'ink' and 'center mass' features.

3.4 Classification using raw pixels as features

3.4.1 Data preprocessing

For the purpose of classification using the raw pixel data as features, we begun the process by reducing the size of our images to half on both axes. This way, we minimized the complexity of the training data, along with the time needed for the completion of the training process. The resizing was applied with the use of the *OpenCV* library, by downsizing by a factor of two, particularly taking the mean of each 2x2 block from the original image and assigning this mean value to a single pixel in the output image.

3.4.2 Classification using Multinomial

In this approach we used 5000 examples of the digits for training the model and the remaining 37000 for testing it. This time, on the multinomial logistic regression model we applied a grid search with 5-fold cross validation keeping the same range for 'C' values (0.01 - 100) but reducing the amount of samples. Precisely the values we chose to test for the hyperparameter 'C' were 0.001, 0.01, 0.1, 1, 10 and 100. This approach resulted in maintaining the variety and keeping the search time at standard levels. In addition, for this model we applied 'LASSO' penalty, for variable selection and regularization. The grid search indicated 'C=1' as the best hyperparameter value with the respective accuracy being 90% (0.90). Next, we trained a new model using '1' for the value of 'C' from our grid search and plotted the confusion matrix presented in Figure 7.

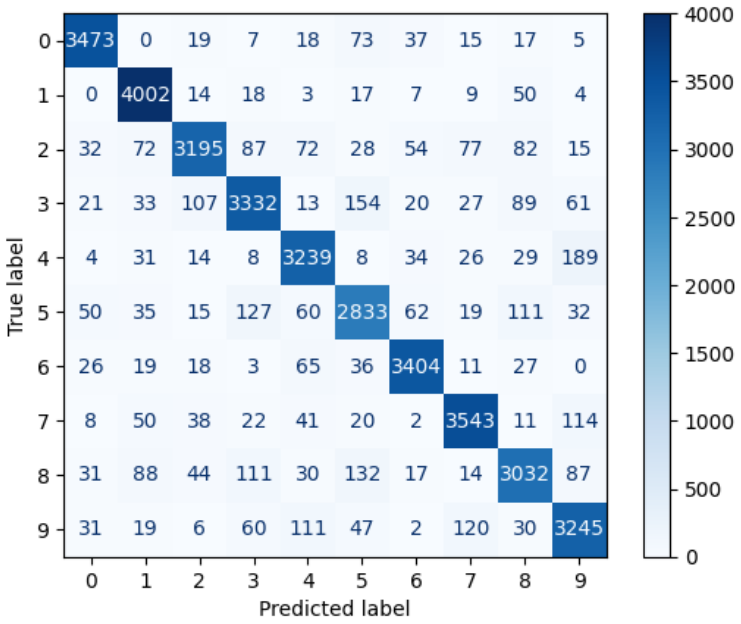


Figure 7: Confusion Matrix for the results of the multinomial logistic regression using pixels as features.

The matrix shows remarkably improved results compared to the previous attempts. The correct predictions of digit '1' continue to prevail against the others, with '5' and '8' again being the weakest links. Overall, the model's performance is relatively high and may appear adequate for a variety of applications.

An important aspect of understanding the Multinomial Logistic Regression model is identifying the importance of each pixel in classifying the digits. To this end, we visualized the model's coefficients as heatmaps (Figure 8), providing an intuitive representation of pixel importance for each digit class. In this visualization, the coefficients of the model, which represent the feature importance, are reshaped to match the dimensions of the input images (14x14 pixels). Red shades in the heatmap indicate pixels that significantly influence the classification of a particular digit, whereas whiter shades represent pixels with minimal influence. Blue pixels on the other hand, suggest a negative influence on the prediction of the corresponding digit. For example, blue pixels at the center of the heatmap for digit '0' imply that the presence of ink in these regions is unlikely in images of '0', aiding the model in distinguishing '0' from other digits. This analysis not only sheds light on how the MLR model makes its decisions but also helps in understanding the distinguishing features of each digit according to the model.

3.4.3 Classification using SVM

For the classification using Support Vector Machine, we used the downsized images from the multinomial logistic regression model, as well as the the same train and test sets. The purpose for this, was to analyze the two models in a similar environment, in order to produce a fair comparison between them. To optimize the SVM parameters, we constructed a pipeline comprising a standard scaler and an SVM classifier. The hyperparameters to be optimized included the regularization parameter C, the kernel type, the degree of the polynomial kernel, and the kernel coefficient. A randomized grid search was employed to explore a range of hyperparameter values. The search covered various SVM kernels (linear, polynomial, RBF, sigmoid), degrees for the polynomial kernel, and values for C and gamma. The

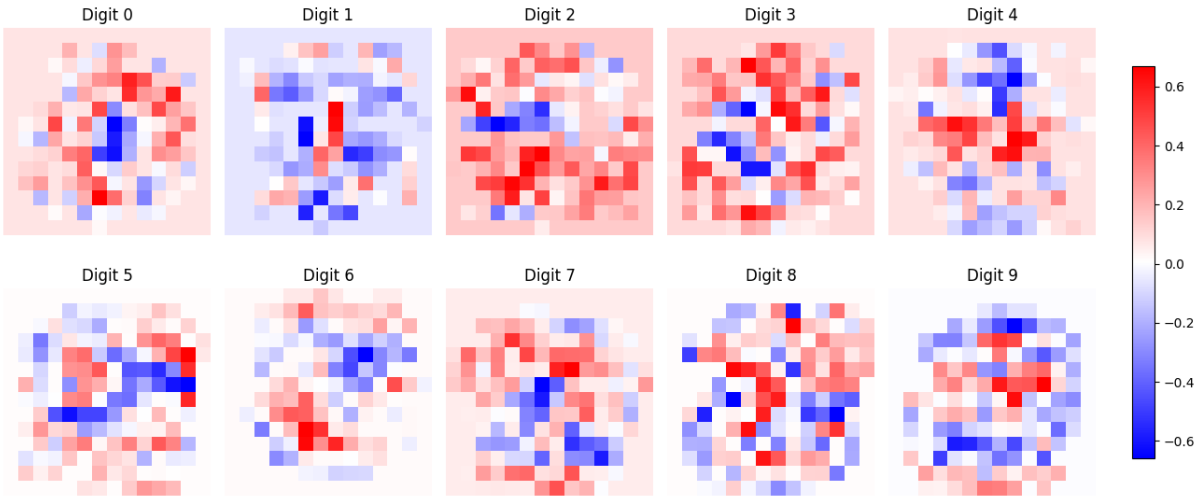


Figure 8: Heatmaps of the MLR's coefficients of each digit, marking the importance of pixels during the classification process.

randomized search was conducted over 10 iterations with a 5-fold cross-validation strategy, optimizing for accuracy. After identifying the best hyperparameters ('C': 70, 'degree': 3, 'gamma': 0.1, 'kernel': 'poly'), we trained a new SVM pipeline using these parameters on the training dataset. The model's performance was then evaluated on the test set. The evaluation focused on the accuracy metric and the confusion matrix. The optimized SVM model demonstrated an accuracy of 93.90% (0.939) on the test set, exceeding the multinomial logit model's accuracy that amounted at 90%. Figure 9 we present the confusion matrix resulting from the SVM classification.

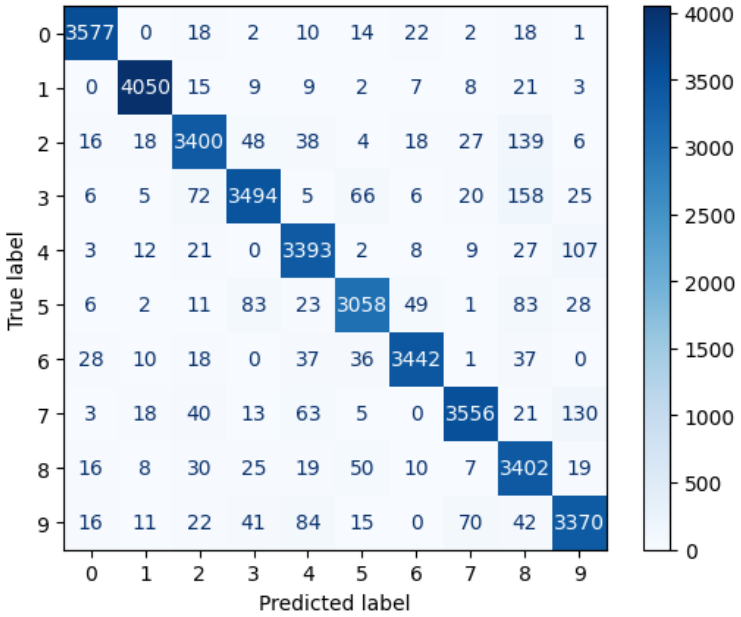


Figure 9: Confusion Matrix for the results of the SVM classification model.

3.5 Multinomial vs SVM

The SVM classifier performed better than the Logistic Regression classifier, achieving higher accuracy by a few percentage points. Specifically, the SVM model achieved an overall accuracy of close to 93%, while the Logistic regression one reached an overall accuracy of approximately 90%. Furthermore, when examining the main diagonal of

		Digits									
		0	1	2	3	4	5	6	7	8	9
Models	Mult. Logit	0.9478	0.9704	0.8599	0.8638	0.9042	0.8471	0.9431	0.9204	0.8455	0.8842
	SVM	0.9762	0.9820	0.9154	0.9058	0.9472	0.9144	0.9537	0.9238	0.9486	0.9180

Table 3: Accuracy for each class of SVM and MLR models.

their confusion matrices for each class (0-9), the values of the SVM classifier prevail over the Logistic Regression's. The accuracy of each independent class is also illustrated in Table 3. Moreover, a significant time difference is observed in the training time of the two classifiers, with the SVM classifier reaching a faster training time of approximately 4 seconds, compared to the Logistic Regression classifier, which required around 15 seconds for the whole process.

To further examine the differences in the performance of the two classifiers, we conducted a statistical analysis using the McNemar test [2]. This test is suitable for comparing two paired classifiers on a binary outcome. In our case, the outcome was whether each prediction by the Support Vector Machine (SVM) and the Multinomial Logistic Regression (MLR) models was correct or incorrect [3].Our null hypothesis is that there is no significant statistical difference between SVM and MLR, and our threshold is $\alpha = 0.05$. For the null hypothesis to be rejected, the p-value in the McNemar test should be lower than 0.05. The value of 0.05, is a commonly used significance threshold, and its selection is designed to achieve a balance between precision and sensitivity. We transformed the multiclass predictions into a binary format by comparing them against the true labels, and constructed a 2x2 contingency table (Table 4) reflecting the instances where both models were either correct or incorrect in their predictions. The McNemar test, applied to this table, helped us determine if there is a significant statistical difference between the predictions of our two models. The results, as shown in Table 5, indicate a significant difference, with a McNemar's test statistic of 726.813 and a p-value of approximately 4.42e-160. Therefore the null hypothesis is rejected. The results indicate a statistically significant difference in the error patterns of the two models, suggesting that they perform differently on certain subsets of the dataset. However, it is important to note that this test examines the binary correctness of the models' predictions and does not delve into the specifics of multiclass classification performance.

		Mult. logit		
		Correct	Incorrect	Total
SVM	Correct	32591	2141	34732
	Incorrect	706	1562	2268
	Total	33297	3703	37000

Table 4: Contingency table of the MLR and SVM models.

	T-Statistic	P-value
SVM vs MLR	726.81	4.42e-160

Table 5: McNemar test results.

In favor of comparing the performance of our two models, we also calculated the precision, recall and F1 score of each unique digit in the dataset for both classifiers. Precision, indicates the proportion of true positives among all instances classified as positive, whereas recall signals the proportion of actual positives that were correctly classified. The harmonic mean of precision and recall is called F1 score and it provides a single metric that balances the former two.

In Table 6 we observe that that the SVM model shows high precision and recall across most classes, especially for digits such as '0', '1', and '6', indicating a strong ability to correctly identify these digits with few false positives or negatives. The relatively lower precision for digit '8' and lower recall for digit '3', suggest that these digits are more challenging for the SVM model to classify correctly. It is interesting that even though the digit '5' possesses the lowest accuracy among the others (see Figures 9 and 7) it doesn't have the lowest precision or recall. A higher precision for '5' suggests that when the model predicts a '5', it is often correct, even if it misses several actual '5's (false negatives). On the other hand, the fact that it doesn't show the lowest recall implies that the model still captures a reasonable proportion of all '5's in the dataset, despite the lower number of true positives.

The MLR model generally has lower precision and recall compared to the SVM model, particularly for digits like '5' and '8', indicating more challenges in correctly classifying these digits. Despite these challenges, the model still performs reasonably well, with decent F1 scores, especially for digits like '0' and '1'.

Overall, regarding precision, the SVM model prevailed against the MLR model in all classes, except '8' (see Figure 10), whereas for recall the former exhibited raised values in all digits (see Figure 11).

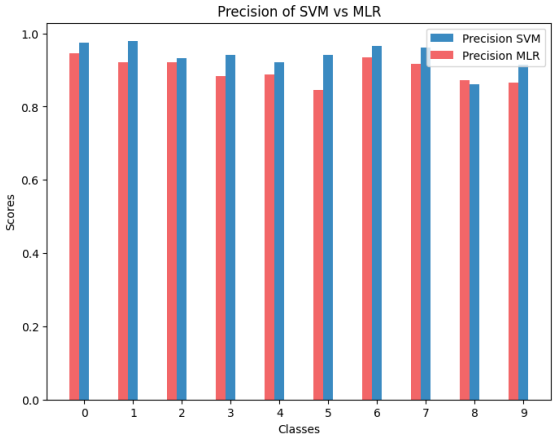


Figure 10: Comparison between the precision values for the SVM and the MLR models.

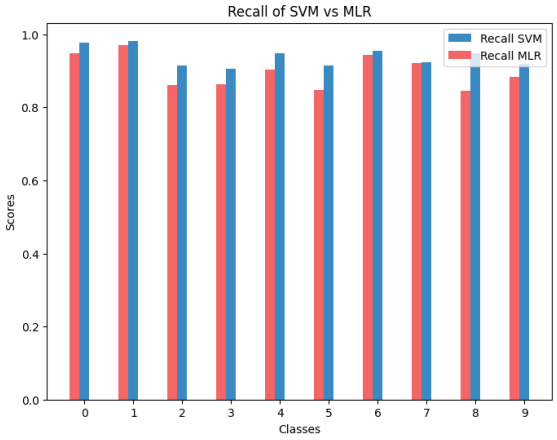


Figure 11: Comparison between the recall values for the SVM and the MLR models.

		0	1	2	3	4	5	6	7	8	9	Mean
SVM	Precision	0.9744	0.9797	0.9323	0.9405	0.9218	0.9403	0.9663	0.9608	0.8617	0.9135	0.9391
	Recall	0.9763	0.9821	0.9155	0.9059	0.9472	0.9145	0.9537	0.9239	0.9487	0.9180	0.9386
	F1	0.9753	0.9809	0.9238	0.9229	0.9343	0.9272	0.9600	0.9420	0.9031	0.9157	0.9385
MLR	Precision	0.9448	0.9202	0.9210	0.8829	0.8867	0.8462	0.9354	0.9177	0.8718	0.8651	0.8992
	Recall	0.9479	0.9704	0.8603	0.8639	0.9042	0.8472	0.9432	0.9208	0.8458	0.8840	0.8988
	F1	0.9463	0.9446	0.8896	0.8733	0.8954	0.8467	0.9393	0.9192	0.8586	0.8744	0.8987

Table 6: Precision, Recall and F1 score of each digit in the dataset for SVM and MLR model.

4 Conclusion and Discussion

In this paper, the evaluation of multinomial logistic regression and support vector machine (SVM) in handwritten digit recognition has provided valuable insights into the effectiveness of these machine learning models. Our study utilized a dataset of 42,000 grayscale images of handwritten digits ranging from 0 to 9, extracted from the MNIST database. The analysis focused on the classification performance of logistic regression and Support vector machines. First, different features, including ink and center mass, and a combination of them, were utilized to test the performance of multinomial logistic regression in these features. Then, the raw pixels of the images were used as features in order to compare the multinomial logistic regression with the SVM classifier. The same dataset was employed for this experiment, with 5000 items for the training process and 37000 for testing.

The results of the study revealed that SVM outperformed MLR in recognizing handwritten digits, achieving an overall accuracy of close to 93%, compared to approximately 90% for the latter. Moreover, the McNemar test revealed a significant statistical difference in the error patterns of the two models. While this result supports the finding that the SVM and logistic regression models perform differently on specific subsets of the dataset, it does not alone constitute evidence of overall higher accuracy for SVM. The observed difference in overall accuracy is supported by the direct comparison of accuracy scores rather than the McNemar test alone.

Additionally, the analysis highlighted the strengths and weaknesses of each model in recognizing specific digits, with SVM generally performing better across most classes. The study also emphasized the importance of feature selection, the impact of different features, and the tuning of the hyperparameters on the classification performance of the models.

Overall, the evaluation of multinomial logistic regression and support vector machines in handwritten digit recognition has provided valuable insights for advancing machine learning applications in this domain, contributing to the ongoing efforts to improve the automation of optical character recognition (OCR) and document analysis tasks. The findings of this study have significant implications for the aforementioned tasks. The effectiveness of SVM in recognizing handwritten digits suggests its potential for enhancing accuracy and efficiency in OCR applications.

5 Implementation Tools

The analysis was conducted using Python 3.10.12 with the following libraries:

- **Numpy (v1.23)**: Used for efficient numerical computations and array manipulations, essential for data processing.
- **Pandas (v1.5)**: Provided data structures for easy data manipulation and preprocessing.
- **Scikit-learn (v1.2)**:
  - **sklearn.preprocessing.scale**: Standardized the dataset, ensuring feature values had a mean of zero and unit variance.
  - **sklearn.metrics**: Offered tools like confusion matrices for assessing and visualizing model performance.
  - **sklearn.linear_model.LogisticRegression**: Applied for creating our Multinomial Logistic Regression model.
  - **sklearn.svm.SVC**: Implemented our Support Vector Machine classifier.
  - **sklearn.model_selection.train_test_split**: Split the data into training and testing sets for model evaluation.
  - **sklearn.model_selection.RandomizedSearchCV**: It is used to find the optimal hyperparameters of an extensive machine learning model through a process of random search.
  - **sklearn.model_selection.GridSearchCV**: Performed systematic hyperparameter tuning to optimize the Multinomial Logistic Regression model's performance.
  - **sklearn.pipeline.Pipeline**: Utilized for the creation of a sequence of data preprocessing steps and model application in a streamlined workflow for the SVM model.
- **Matplotlib (v3.7)**: Enabled the creation of various data visualizations, including images, performance charts, and confusion matrices.
- **OpenCV (v4.8)**: For the processing task of resizing and the images from size 28x28 to 14x14 before using them for training the models.
- **Scipy (v1.11)**:
  - **scipy.stats.uniform**: Provided uniform continuous random variables for hyperparameter tuning the 'C' value of the SVM model.

References

[1] Ayushi Sharma et al. "A Machine Learning and Deep Learning Approach for Recognizing Handwritten Digits". In: *Computational Intelligence and Neuroscience* 2022 (2022), pp. 1–7.

[2] Thomas G Dietterich. "Approximate statistical tests for comparing supervised classification learning algorithms". In: *Neural Computation* 10.7 (1998), pp. 1895–1923.

[3] Hassaan Malik et al. "CDC$_Net : multi-classificationconvolutionalneuralnetworkmodelfordetectionofCOVID-19, pneumothorax, pneumonia, lungCancer, andtuberculosisusingchestX-rays$". In: ***Multimedia Tools and Applications*** 82.9 (Sept. 2022), pp. 13855–13880.