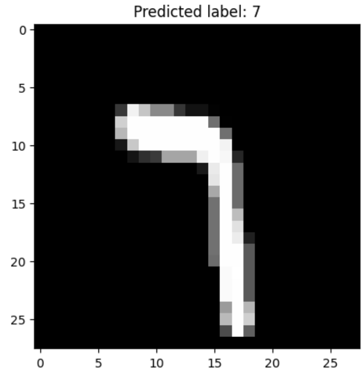
Miguel J. Garrido

Digit Recognizer

Introduction

In this project different machine learning models are going to be trained on the MNIST digit dataset. Then it is going to be tested how well do these models generalize to different datasets and lastly a webpage with *streamlit* is going to be created where users can upload an image with a digit (from 0-9) or write themselves a digit with the cursor and our model is going to predict the number uploaded or written.



Training model dataset: MNIST

MNIST (LeCun, Cortes, & Burges, 1998) is a subset of the NIST dataset downloaded from *Kaggle* (hojjatk, 2018). It is a database containing images and labels for handwritten digits. It has a training set of 60,000 examples, and a test set of 10,000 examples.

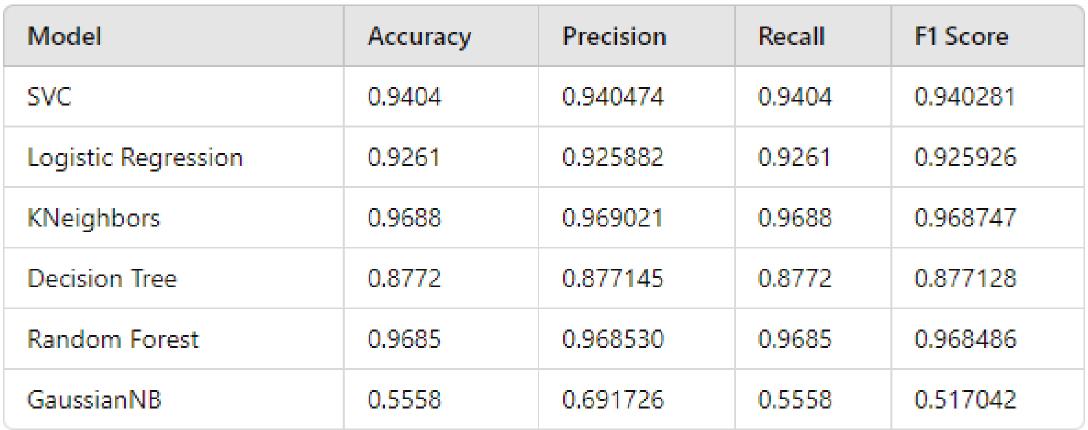
Digits in this database are size-normalized and centered in a fixed-size image.

MNIST data pre-processing

For the models training, some preprocessing had to be done to ensure data was in a suitable format.

- Normalization: Each pixel value in the image was scaled between 0 and 1. For doing this we divide them by 255 since each channel (red, green, and blue) is 8 bits so limited at the 256 values, in this case 255 since 0 is included.
- Reshaping: Images are 2D, 28x28 pixels, so they were flattened into 1D vectors of size 784 (28x28).

Model training and evaluation



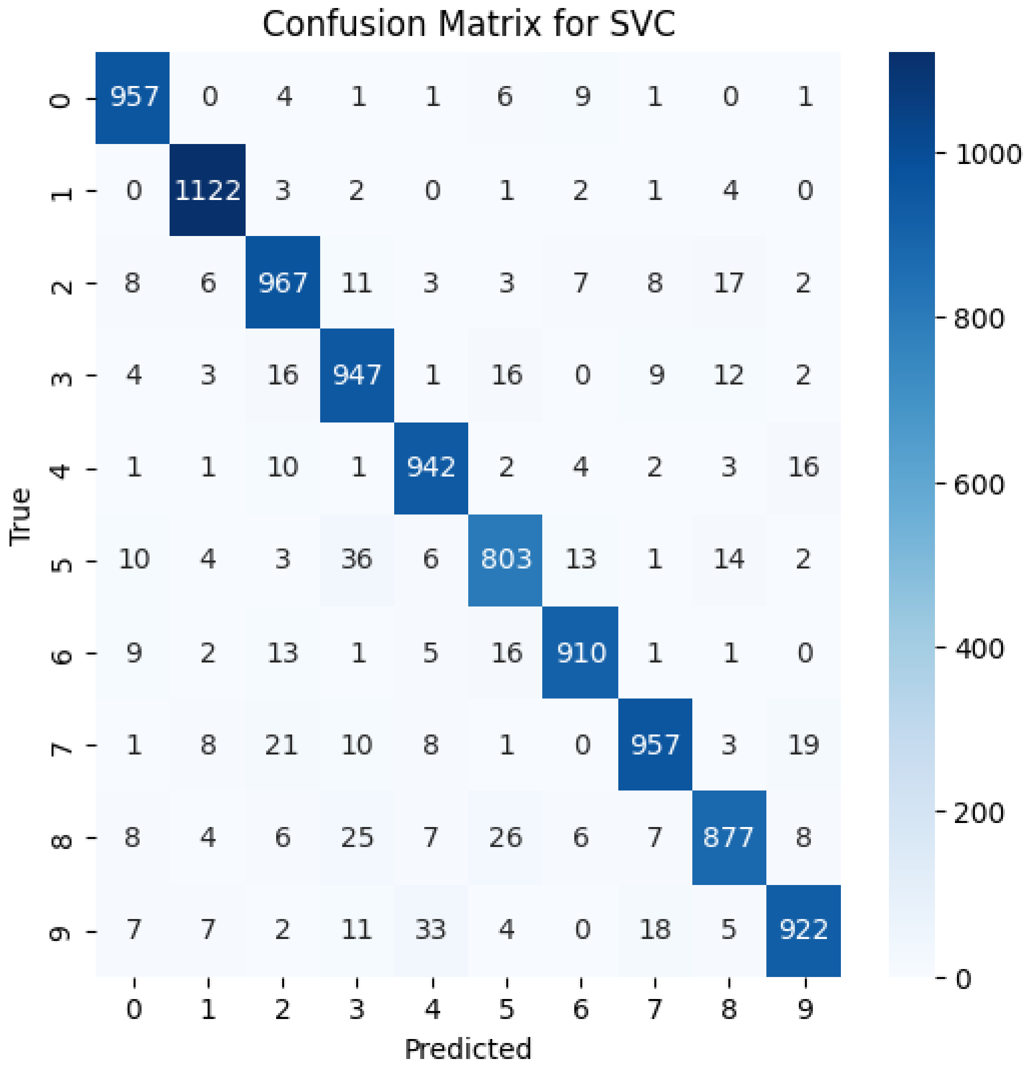
Model	Accuracy	Precision	Recall	F1 Score
SVC	0.9404	0.940474	0.9404	0.940281
Logistic Regression	0.9261	0.925882	0.9261	0.925926
KNeighbors	0.9688	0.969021	0.9688	0.968747
Decision Tree	0.8772	0.877145	0.8772	0.877128
Random Forest	0.9685	0.968530	0.9685	0.968486
GaussianNB	0.5558	0.691726	0.5558	0.517042

Support Vector Classifier (SVC)

- Accuracy: 94.04%
- Precision: 94.05%
- Recall: 94.04%
- F1-Score: 94.03%

Performs decently with all metric around 94%, this suggests a good balance between precision and recall.

This was expected from a model robust in high-dimensional spaces having multi-classes with normalized images.

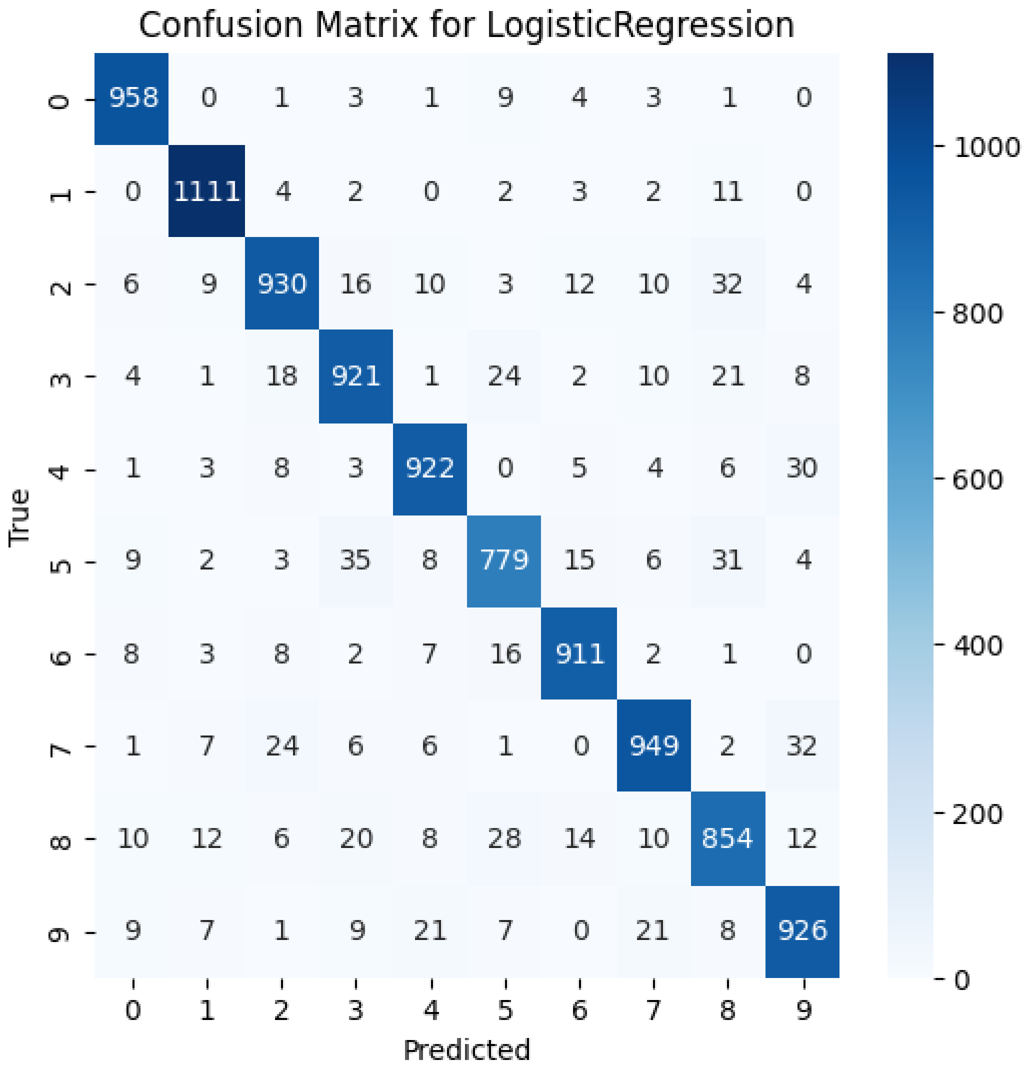


We can see a balanced confusion matrix where most of the predictions equal the labels, this seems logical since we have high accuracy and precision values.

Logistic Regression

- Accuracy: 92.61%
- Precision: 92.59%
- Recall: 92.61%
- F1-Score: 92.59%

Not as strong compared to SVC but still decent performance. This is not a flexible model like KNN or Random Forest due to its linear decision boundary which limits its performance in complex data patterns.



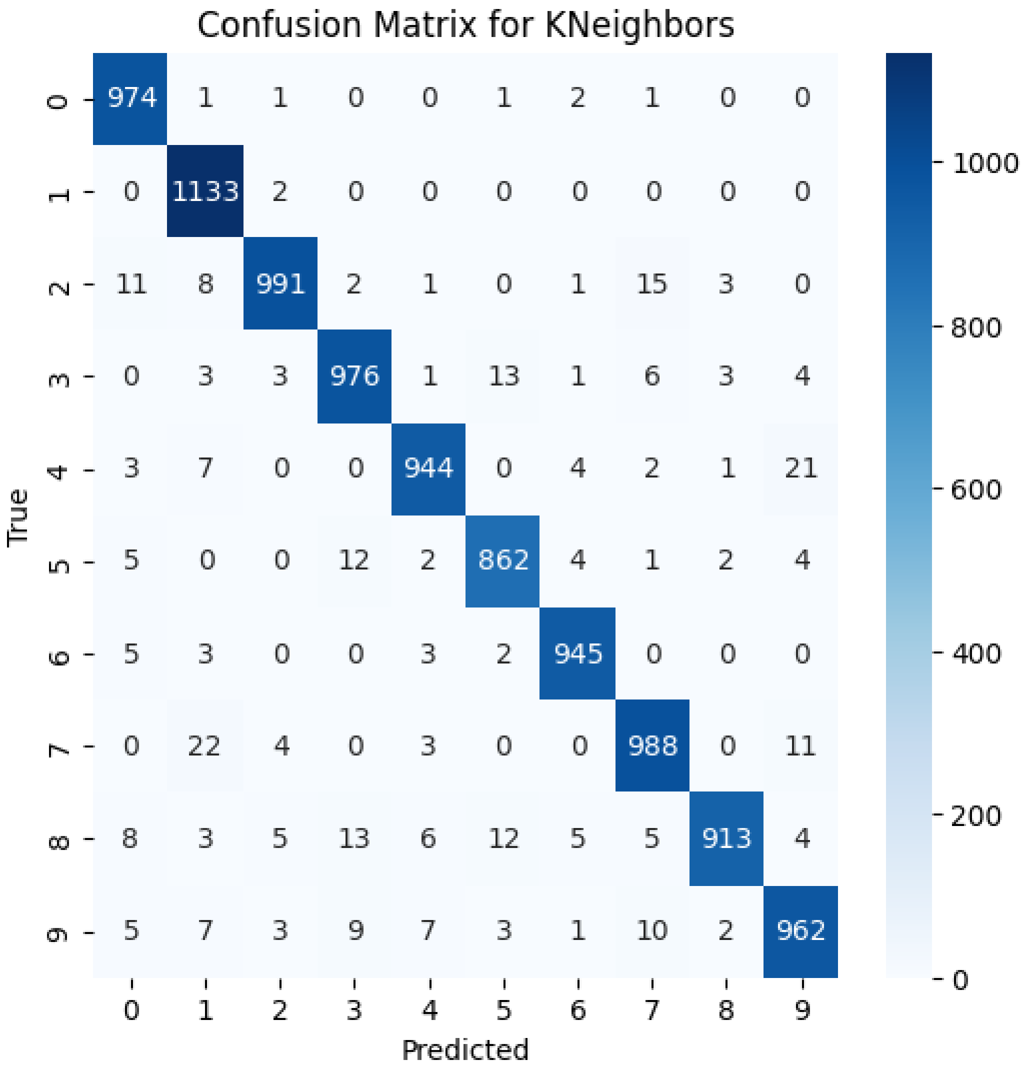
	0	1	2	3	4	5	6	7	8	9
0	958	0	1	3	1	9	4	3	1	0
1	0	1111	4	2	0	2	3	2	11	0
2	6	9	930	16	10	3	12	10	32	4
3	4	1	18	921	1	24	2	10	21	8
4	1	3	8	3	922	0	5	4	6	30
5	9	2	3	35	8	779	15	6	31	4
6	8	3	8	2	7	16	911	2	1	0
7	1	7	24	6	6	1	0	949	2	32
8	10	12	6	20	8	28	14	10	854	12
9	9	7	1	9	21	7	0	21	8	926

We can see a balanced confusion matrix where most of the predictions equal the labels, this seems logical since we have high accuracy and precision values.

KNeighbors

- Accuracy: 96.88%
- Precision: 96.90%
- Recall: 96.88%
- F1-Score: 96.87%

Really good performance with the highest accuracy of all models. The proximity-based learning method typically excels with sufficient data, and in this case, the performance indicates that it's capable of capturing local data patterns effectively. The only downside is computational cost.



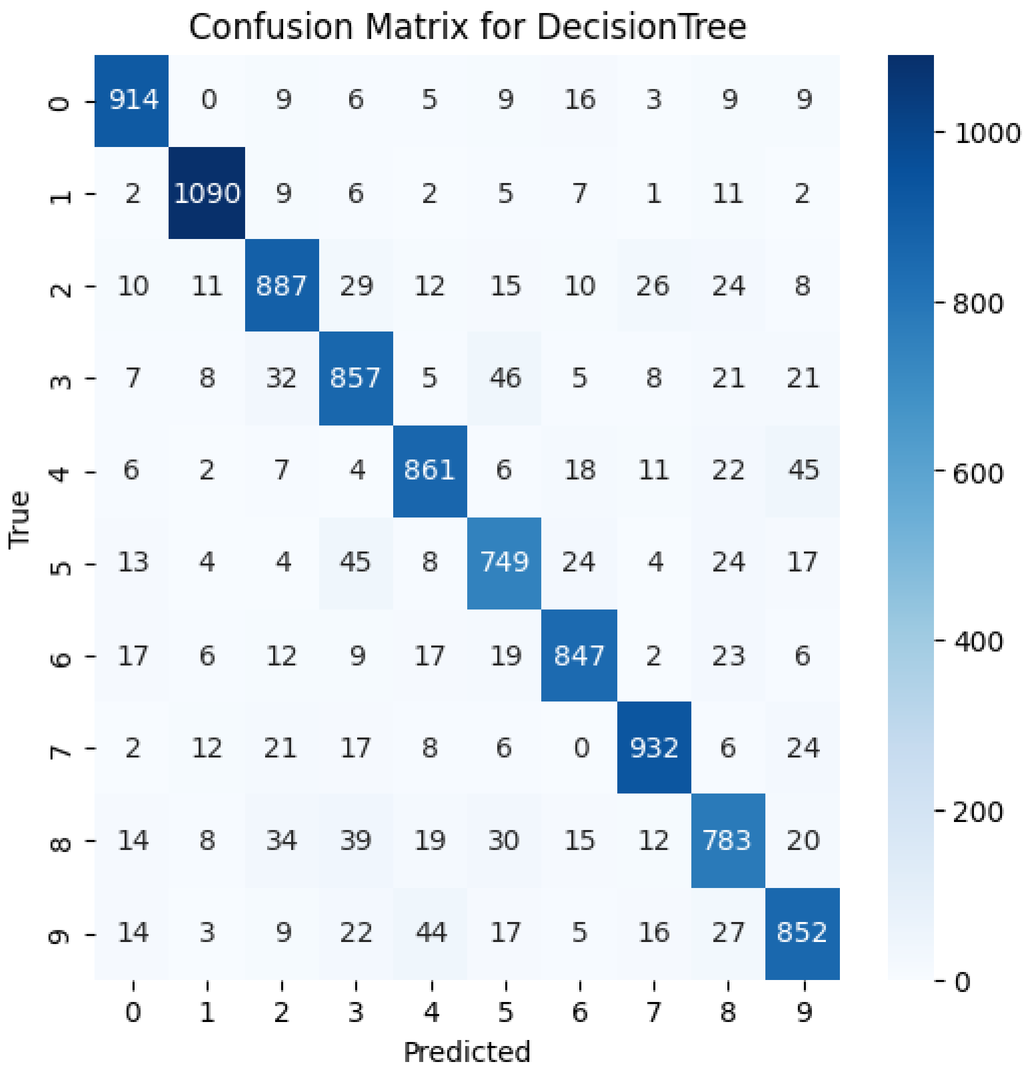
True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	974	1	1	0	0	1	2	1	0	0
1	0	1133	2	0	0	0	0	0	0	0
2	11	8	991	2	1	0	1	15	3	0
3	0	3	3	976	1	13	1	6	3	4
4	3	7	0	0	944	0	4	2	1	21
5	5	0	0	12	2	862	4	1	2	4
6	5	3	0	0	3	2	945	0	0	0
7	0	22	4	0	3	0	0	988	0	11
8	8	3	5	13	6	12	5	5	913	4
9	5	7	3	9	7	3	1	10	2	962

We can see a balanced confusion matrix where most of the predictions equal the labels, this seems logical since we have high accuracy and precision values.

Decision tree

- Accuracy: 87.72%
- Precision: 87.71%
- Recall: 87.72%
- F1-Score: 87.71%

Lower performance with an accuracy of 87.72%. The hypothesis was that it would overfit and not generalize well with other datasets without pruning and overfitting of noise, inthis case the absence of it.



Confusion Matrix for DecisionTree

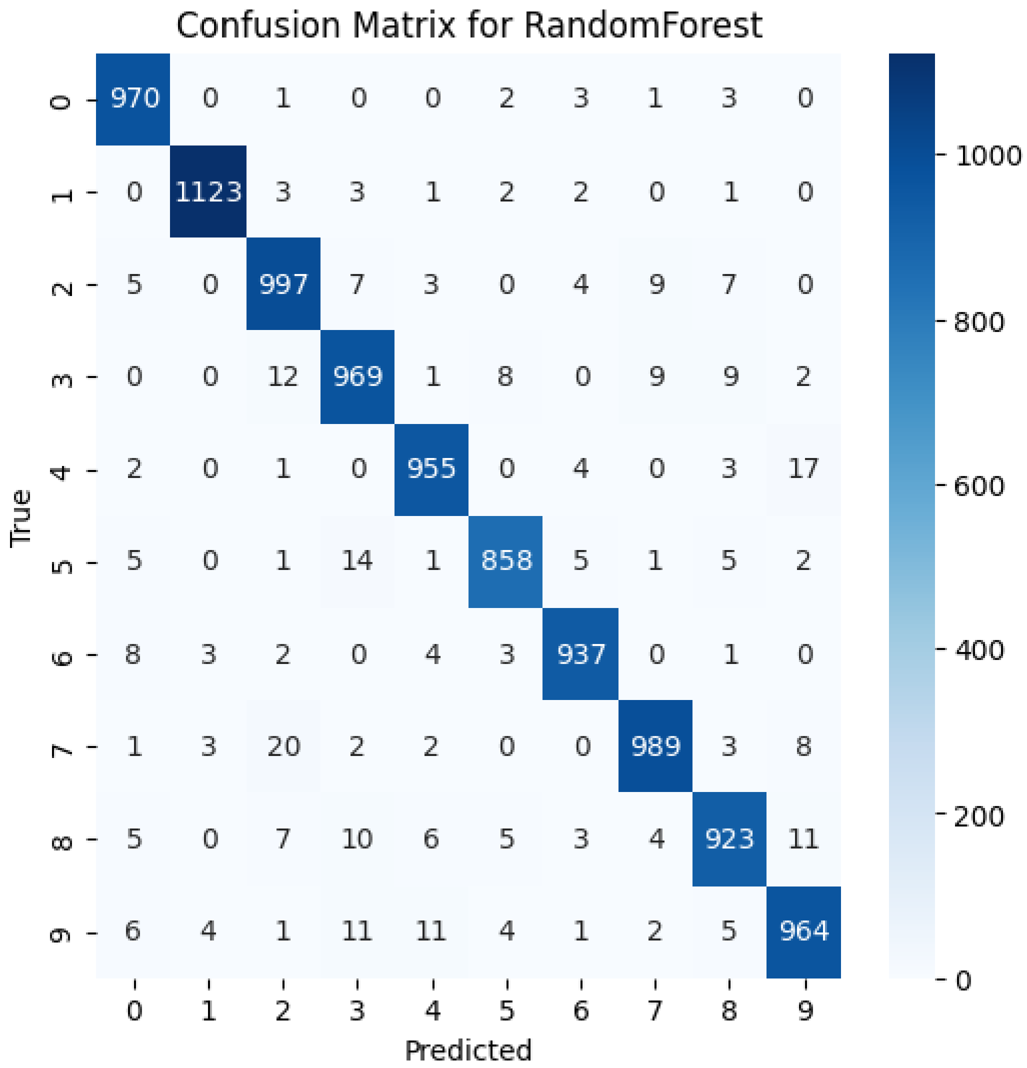
True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	914	0	9	6	5	9	16	3	9	9
1	2	1090	9	6	2	5	7	1	11	2
2	10	11	887	29	12	15	10	26	24	8
3	7	8	32	857	5	46	5	8	21	21
4	6	2	7	4	861	6	18	11	22	45
5	13	4	4	45	8	749	24	4	24	17
6	17	6	12	9	17	19	847	2	23	6
7	2	12	21	17	8	6	0	932	6	24
8	14	8	34	39	19	30	15	12	783	20
9	14	3	9	22	44	17	5	16	27	852

Random Forest

- ➢ Accuracy: 96.85%
- ➢ Precision: 96.85%
- ➢ Recall: 96.85%
- ➢ F1-Score: 96.85%

Similar performance to KNeighbors with a high accuracy of 96.85%. Strong performance across all metrics suggests it has successfully captured complex data patterns in the data.

Hypothesis is that because of it being an ensemble model, it benefits from reducing variance by averaging multiple decision trees, which improves generalization and prevents overfitting.



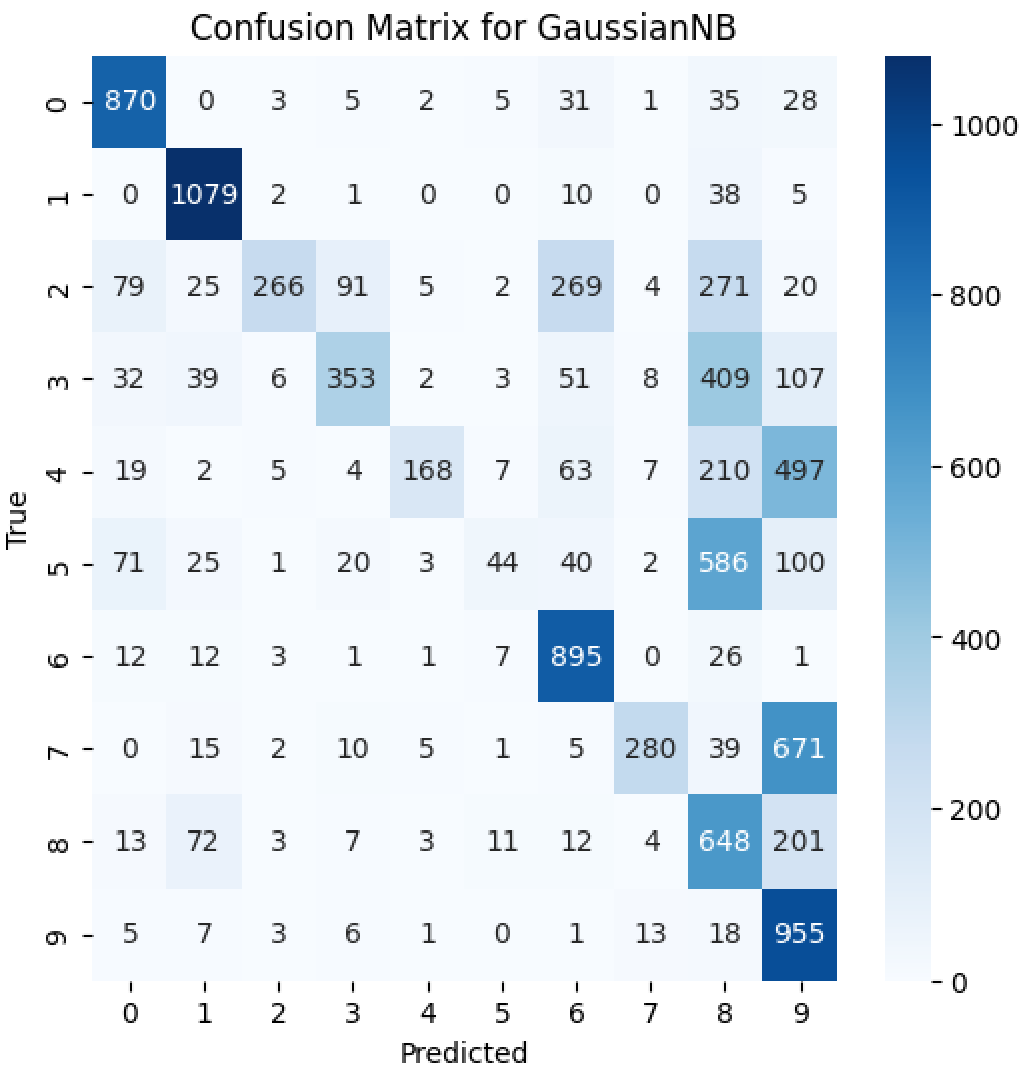
True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	970	0	1	0	0	2	3	1	3	0
1	0	1123	3	3	1	2	2	0	1	0
2	5	0	997	7	3	0	4	9	7	0
3	0	0	12	969	1	8	0	9	9	2
4	2	0	1	0	955	0	4	0	3	17
5	5	0	1	14	1	858	5	1	5	2
6	8	3	2	0	4	3	937	0	1	0
7	1	3	20	2	2	0	0	989	3	8
8	5	0	7	10	6	5	3	4	923	11
9	6	4	1	11	11	4	1	2	5	964

Gaussian Naïve Bayes

- Accuracy: 55.58%
- Precision: 69.17%
- Recall: 55.58%
- F1-Score: 51.70%

Performs the worst with an accuracy of 55.58%. It depends on feature independence and in this case features in the dataset can be similar amongst them.

Hypothesis is that due to relatively higher precision compared to recall suggests that it is conservative in predicting positive cases but misses many, leading to a low overall effectiveness in generalization.



Confusion Matrix for GaussianNB

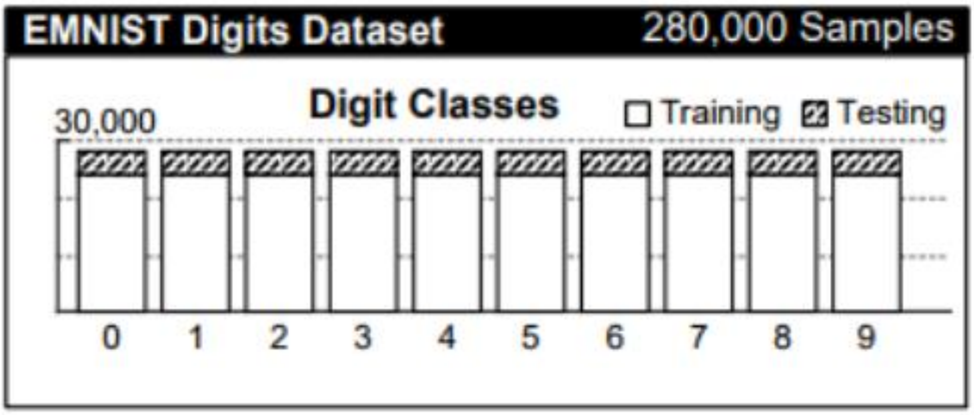
True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	870	0	3	5	2	5	31	1	35	28
1	0	1079	2	1	0	0	10	0	38	5
2	79	25	266	91	5	2	269	4	271	20
3	32	39	6	353	2	3	51	8	409	107
4	19	2	5	4	168	7	63	7	210	497
5	71	25	1	20	3	44	40	2	586	100
6	12	12	3	1	1	7	895	0	26	1
7	0	15	2	10	5	1	5	280	39	671
8	13	72	3	7	3	11	12	4	648	201
9	5	7	3	6	1	0	1	13	18	955

Models generalization

EMNIST

EMNIST dataset

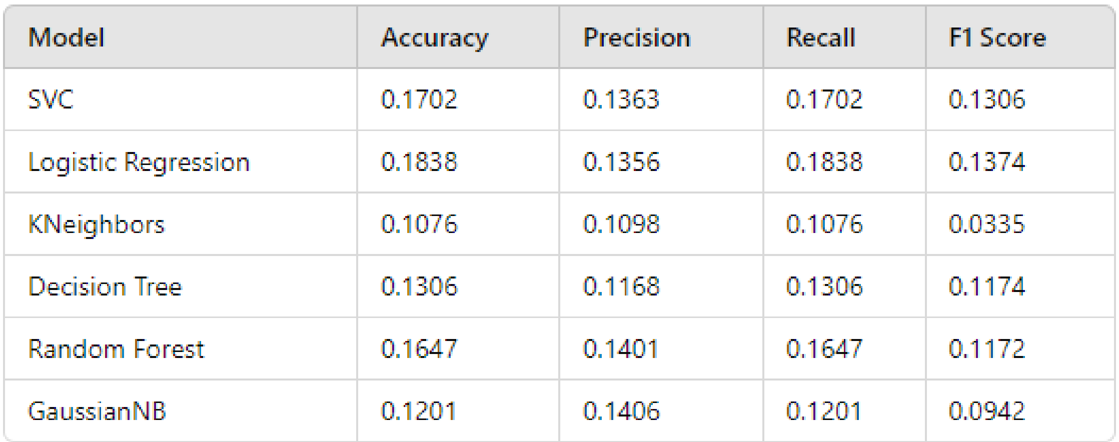
"The EMNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset. " (Crawford, 2017). Six different splits are provided but only the digits dataset was used. It was used only the test images and labels, the test dataset consisted of 40,000 digits.



EMNIST preprocessing

Same as with MNIST.

Evaluation

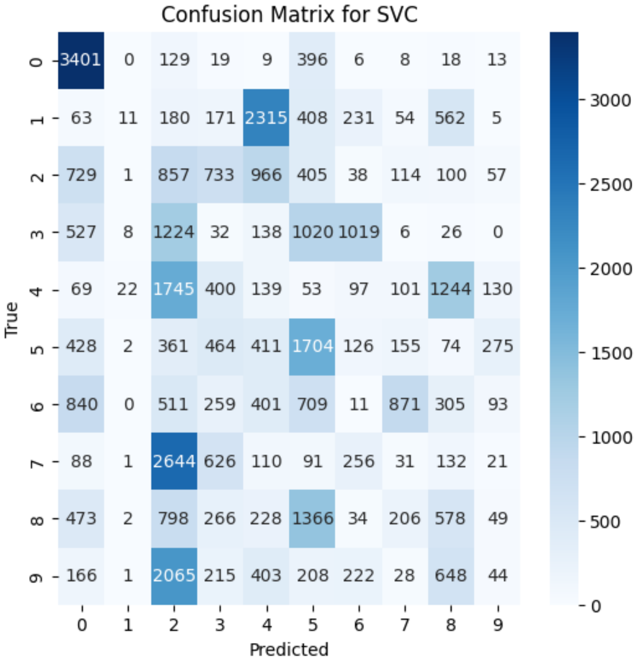


Model	Accuracy	Precision	Recall	F1 Score
SVC	0.1702	0.1363	0.1702	0.1306
Logistic Regression	0.1838	0.1356	0.1838	0.1374
KNeighbors	0.1076	0.1098	0.1076	0.0335
Decision Tree	0.1306	0.1168	0.1306	0.1174
Random Forest	0.1647	0.1401	0.1647	0.1172
GaussianNB	0.1201	0.1406	0.1201	0.0942

All models exhibit very low accuracy, with none exceeding 20%. This is a huge performance drop when changing from MNIST to EMNIST. The precision, recall, and F1 scores are also low across the board, suggesting that the models are struggling to effectively identify patterns.

Models may be overfitting in the MNIST dataset. Low precision, meaning lots of false positives and recall values similarly low.

This may be due to noise. EMNIST contains more variations in handwriting and image quality differs.



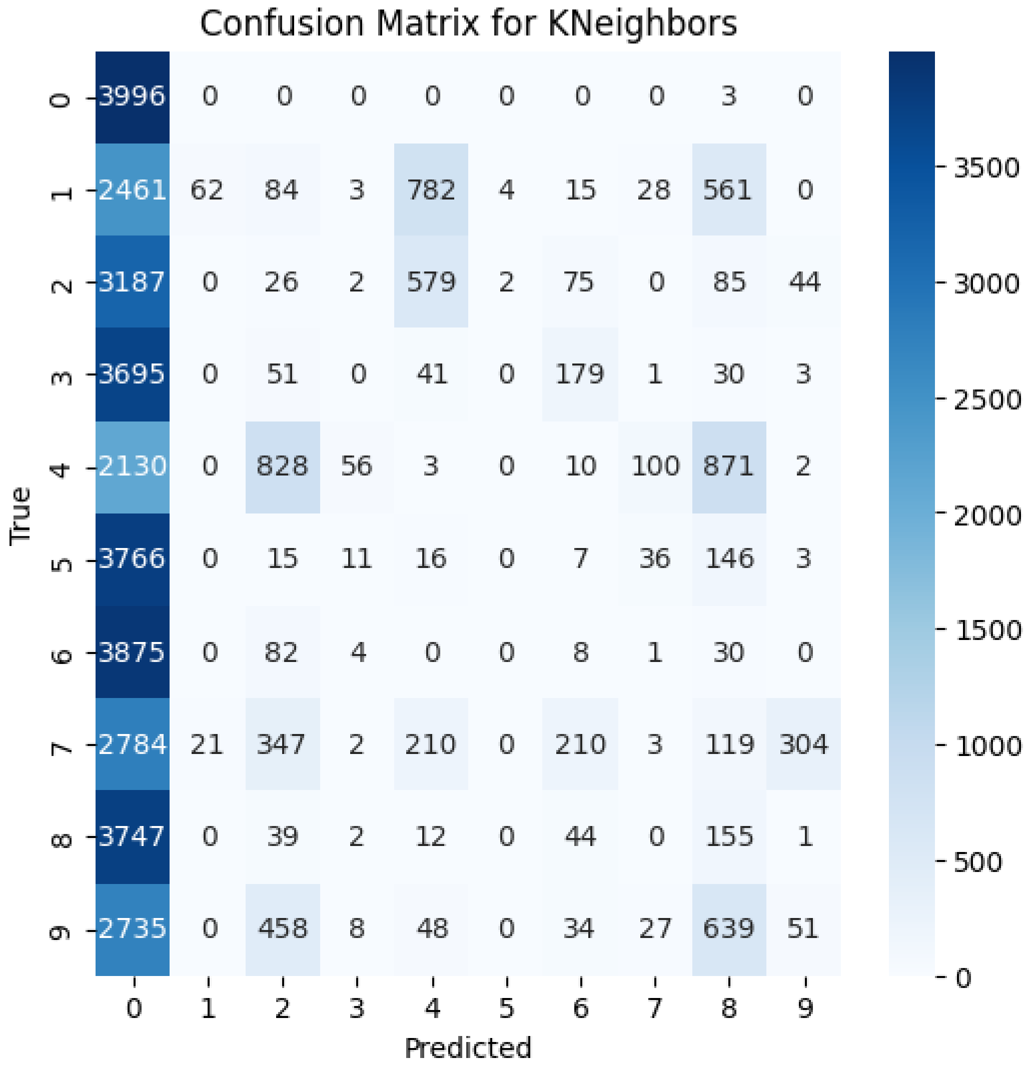
	0	1	2	3	4	5	6	7	8	9
0	3401	0	129	19	9	396	6	8	18	13
1	63	11	180	171	2315	408	231	54	562	5
2	729	1	857	733	966	405	38	114	100	57
3	527	8	1224	32	138	1020	1019	6	26	0
4	69	22	1745	400	139	53	97	101	1244	130
5	428	2	361	464	411	1704	126	155	74	275
6	840	0	511	259	401	709	11	871	305	93
7	88	1	2644	626	110	91	256	31	132	21
8	473	2	798	266	228	1366	34	206	578	49
9	166	1	2065	215	403	208	222	28	648	44

The models SVC, DecisionTree and random forest had scarce results which resulted in a disperse confusion matrix.

In SVC may be due to the classes not being linearly separable or having considerable overlap in feature space.

With Decision Trees this may be due to data being noisy, the tree might create convoluted decision boundaries, leading to misclassifications and less distinct patterns.

Lastly Random Forest struggles with overlapping class distributions. Its individual trees may make different decisions based on random subsets of features, resulting in varied predictions that do not follow a clear pattern.


Confusion Matrix for KNeighbors

True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	3996	0	0	0	0	0	0	0	3	0
1	2461	62	84	3	782	4	15	28	561	0
2	3187	0	26	2	579	2	75	0	85	44
3	3695	0	51	0	41	0	179	1	30	3
4	2130	0	828	56	3	0	10	100	871	2
5	3766	0	15	11	16	0	7	36	146	3
6	3875	0	82	4	0	0	8	1	30	0
7	2784	21	347	2	210	0	210	3	119	304
8	3747	0	39	2	12	0	44	0	155	1
9	2735	0	458	8	48	0	34	27	639	51

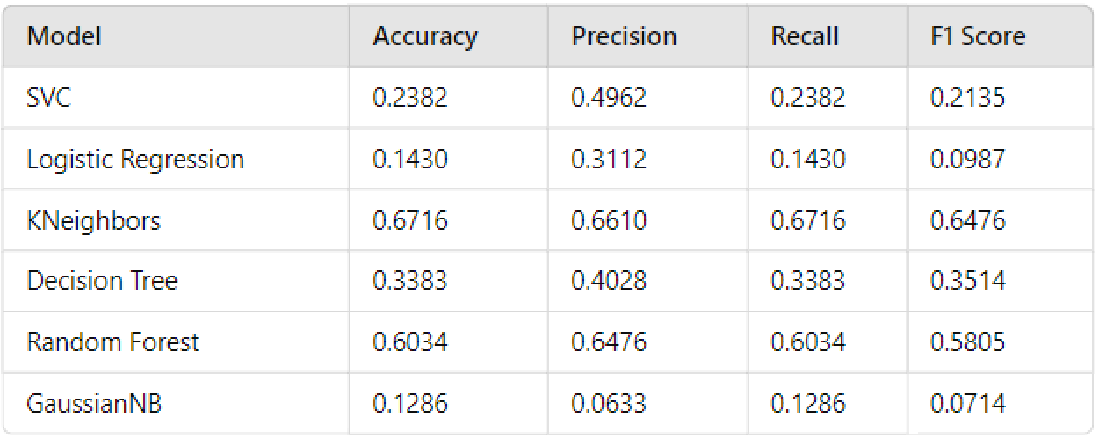
Models like KNN and Logistic Regression can leverage local structures and probabilistic interpretations, respectively, leading to better classification outcomes and clearer predictions. Overall, these factors contribute to the model's capacity to learn and generalize from the training data effectively, thereby influencing the distinguishability of patterns observed in their confusion matrices.

USPS

USPS dataset

Images had to be reshaped from 16x16 to 28x28 then flattened. This reshaping may cause quality and feature changes, affecting the models performances.

Evaluation



Model	Accuracy	Precision	Recall	F1 Score
SVC	0.2382	0.4962	0.2382	0.2135
Logistic Regression	0.1430	0.3112	0.1430	0.0987
KNeighbors	0.6716	0.6610	0.6716	0.6476
Decision Tree	0.3383	0.4028	0.3383	0.3514
Random Forest	0.6034	0.6476	0.6034	0.5805
GaussianNB	0.1286	0.0633	0.1286	0.0714

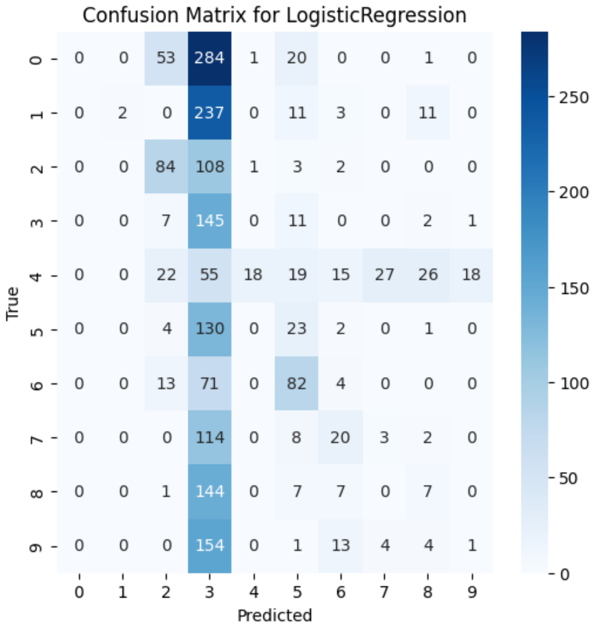
The models KNeighbors and Random Forest outperformed the rest of them with accuracies of 67.2% and 60.3%. Other models performed poorly but these two demonstrated a decent generalization ability.

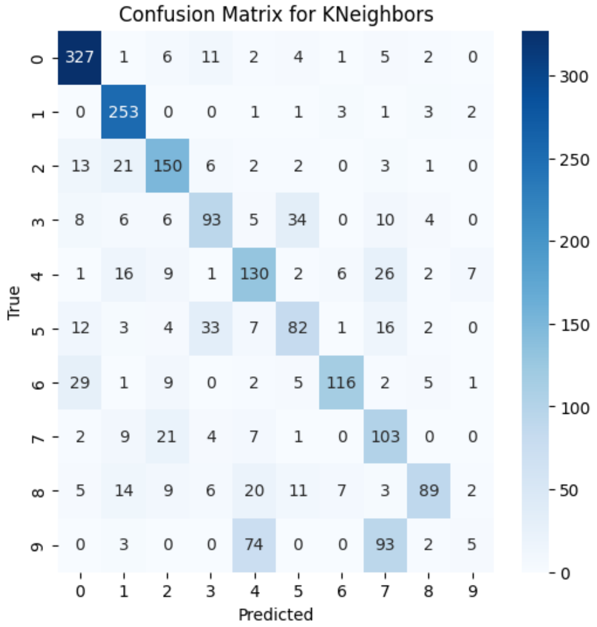
KNeighbors model ability to achieve high accuracy suggests that it effectively captures local patterns in the USPS dataset, likely due to its instance-based learning nature, which allows it to classify based on the nearest neighbors.

Random Forest model's ensemble approach helps improve accuracy by averaging out predictions, making it robust against overfitting while maintaining good performance.

Precision scores for SVC (49.63%) and Random Forest (64.76%) suggest that while they have decent accuracy, they still face issues with false positives, especially given the relatively high precision of KNeighbors at (66.1%).

All this may be due to datasets characteristics being somewhat similar but still different. Having different quality, different original size…





Same explanation as before.

The Street View House Numbers (SVHN) Dataset

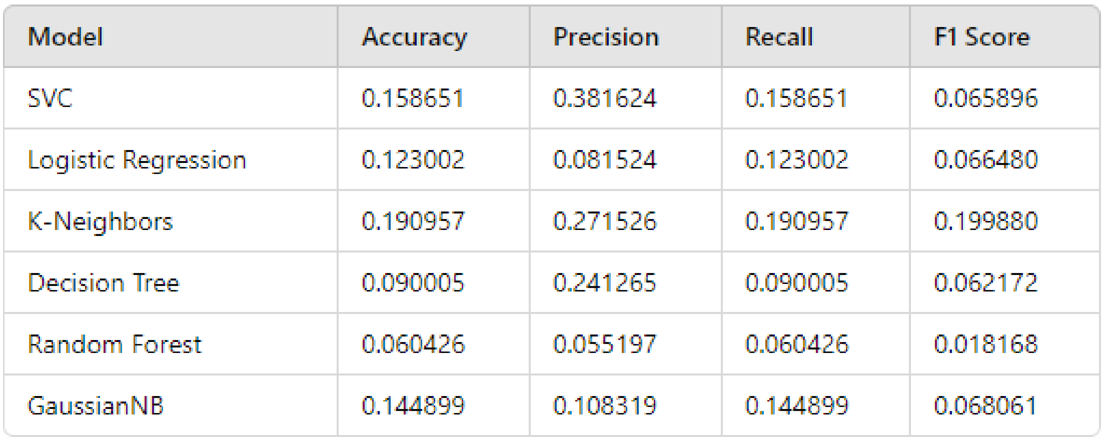
"SVHN is a real-world image dataset for developing machine learning and object recognition algorithms with minimal requirement on data preprocessing and formatting. It can be seen as similar in flavor to MNIST (e.g., the images are of small cropped digits), but incorporates an order of magnitude more labeled data (over 600,000 digit images) and comes from a significantly harder, unsolved, real world problem (recognizing digits and numbers in natural scene images). SVHN is obtained from house numbers in Google Street View images."(Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng, 2011)

"Character level ground truth in an MNIST-like format. All digits have been resized to a fixed resolution of 32-by-32 pixels. The original character bounding boxes are extended in the appropriate dimension to become square windows, so that resizing them to 32-by-32 pixels does not introduce aspect ratio distortions. Nevertheless this preprocessing introduces some **distracting** digits to the sides of the digit of interest. Loading the .mat files creates 2 variables: **X** which is a 4-D matrix containing the images, and **y** which is a vector of class labels. To access the images, X(:,:,:,i) gives the i-th 32-by-32 RGB image, with class label y(i)."

Preprocessing

Original images are 32x32 pixels and in RGB. They had to be converted to greyscale with *rgb2gray()* and then resized to 28x28. After that once they were flattened the label 10 was set to 0, since the digits are from 0 to 9 and number 10 in the labels in this dataset represented the 0.

Evaluation



Model	Accuracy	Precision	Recall	F1 Score
SVC	0.158651	0.381624	0.158651	0.065896
Logistic Regression	0.123002	0.081524	0.123002	0.066480
K-Neighbors	0.190957	0.271526	0.190957	0.199880
Decision Tree	0.090005	0.241265	0.090005	0.062172
Random Forest	0.060426	0.055197	0.060426	0.018168
GaussianNB	0.144899	0.108319	0.144899	0.068061

Overall, a poor performance with low accuracy. Models struggle to classify instances correctly with the highest accuracy being 19.1% for K-Neighbors and Random Forest now having a 6%. This is due to data quality and noise. This dataset is really different to the MNIST dataset the models were trained on and includes significant amount of noise, irrelevant features and outliers. All this leads to diminished model performance, as models learn from the misleading patterns present in the data.

Conclusion

The evaluation of different machine learning models on the MNIST dataset and other different datasets reveals significant discrepancies in performance, with some models demonstrating overfitting to the MINST dataset and overfitting to some features.

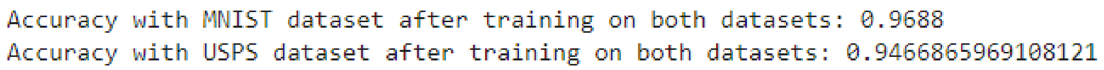
Although in the USPS dataset some models performed decently demonstrating their ability to generalize.

Some fine tuning could be implemented to increase performance but arguably the best improvement that could be made would be training the models on a new dataset that mixes all the different datasets into one, so it can learn lots of features.

Upgrading the models

Finally, for so the web app could make better predictions, more robust models were needed.

A KNeighbors model was trained again with a new dataset. This dataset consisted of a mix of the MNIST and the USPS datasets. KNeighbors was choses because it was the model that showed the best performance and generalization.



```
Accuracy with MNIST dataset after training on both datasets: 0.9688
Accuracy with USPS dataset after training on both datasets: 0.9466865969108121
```

With this a better accuracy and precision was obtained when testing on both datasets with executions of 94.66% and 96.88%.

Although the EMNIST and the SVHN datasets precision and accuracy was lower this doesn't matter for our objective. Performance decreased even more because MNIST and USPS features are somewhat similar and very different from EMNIST and SVHN features. But where the final user would draw the digit in the webapp will have more similar features to MNIST and USPS.

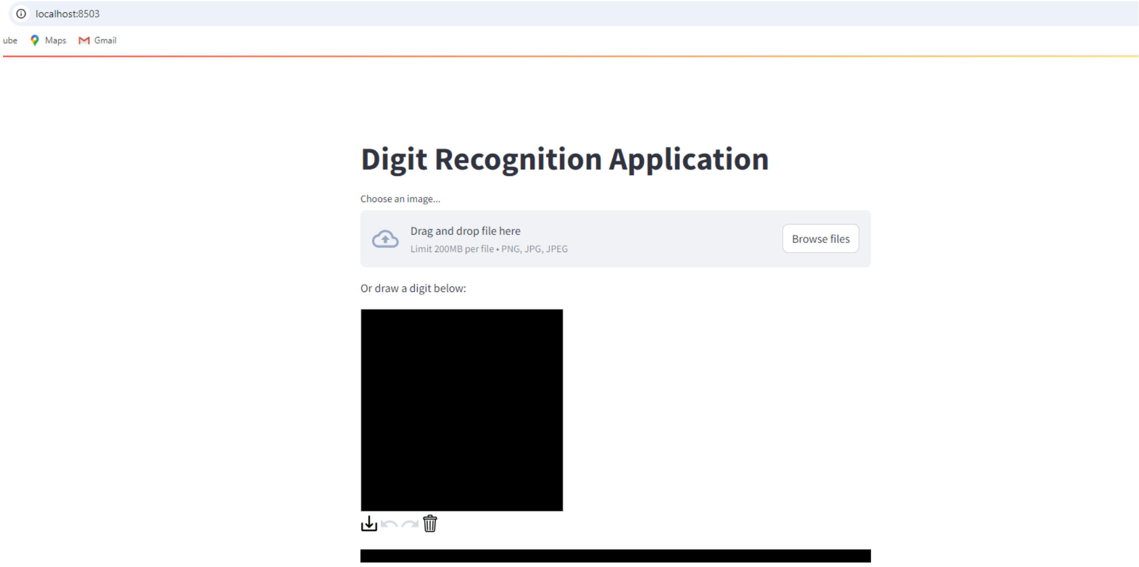
Web app implementation with *streamlit*

Lastly a web app implementation was made for making predictions using the KNeighbors model since it is the one that generalized the best

The application enables users to upload images of handwritten digits or draw their own digits directly on a canvas.

Once an image is provided, the application processes it by converting it to grayscale, resizing it to the appropriate dimensions (28x28 pixels, which is standard for MNIST), normalizing the pixel values, and reshaping the image into a format suitable for the model. After preprocessing, the application uses the trained model to make predictions about the digit depicted in the image.

The predictions are displayed to the user along with a loading spinner to indicate that the model is processing the input. If any errors occur during the prediction process, they are caught and displayed to the user. This application is built using Streamlit, which allows for an interactive web interface, enhancing user experience by providing real-time feedback.



localhost:8503

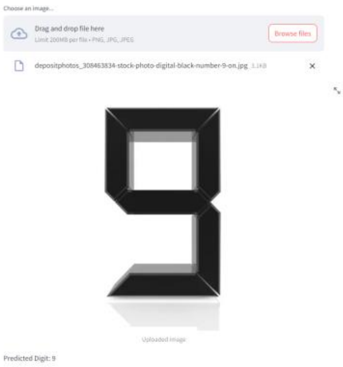
ube Maps Gmail

Digit Recognition Application

Choose an image...

Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG

Or draw a digit below:



Choose an image...

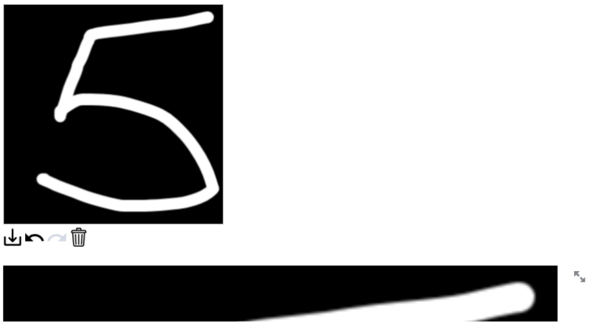
Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG

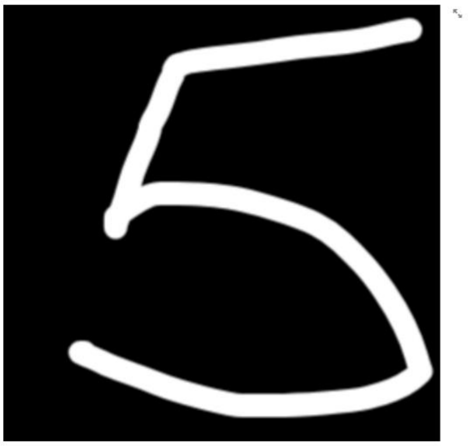
depositphotos_308463834-stock-photo-digital-black-number-9-on.jpg 3.1KB

Uploaded image

Predicted Digit: 9

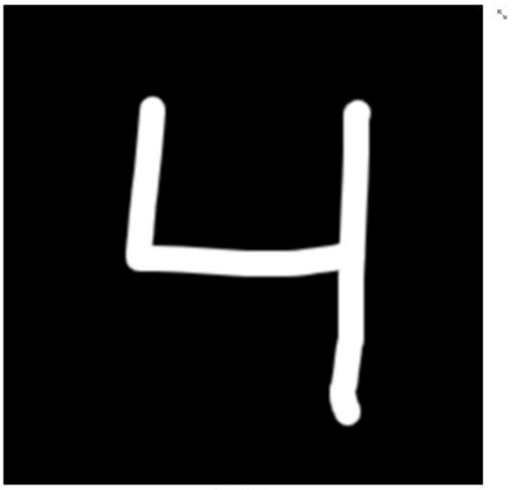
Or draw a digit below:





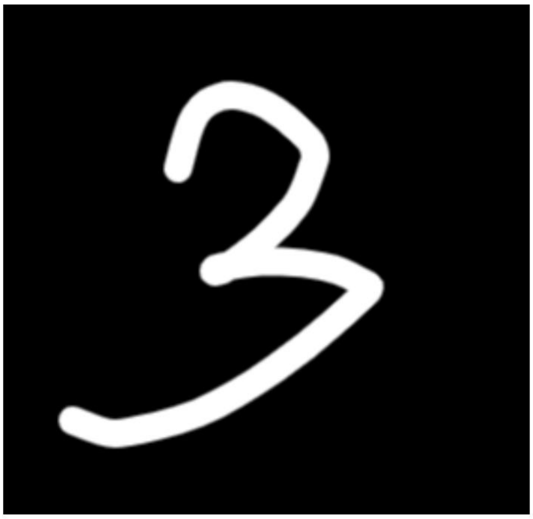
Drawn Image

Predicted Digit: 5



Drawn Image

Predicted Digit: 4



Drawn Image

Predicted Digit: 3

We can see the app works fine and the predictions are accurate. Thanks to using the upgraded model with the MNIST and USPS combined we get a high precision of predictions. Only some digits will not be predicted correctly and “confused” by the model. For example, a narrow “9” may be confused with a 1.