1. **Problem Statement:**

To recognise the handwritten digits with the help of a model built on MNIST dataset.

1. **Practical Applications of this Work:**

a.Bank applications

b.Number plate recognising

c.Medical field

1. **Project Links:**
   1. **Dataset Link:** [**https://www.kaggle.com/competitions/digit-recognizer/data**](https://www.kaggle.com/competitions/digit-recognizer/data)
   2. **Github Repo link:**

https://github.com/Akshitha27304/MNIST-Handwritten-dataset

1. **Exploratory Data Analysis:**
2. **Data Preprocessing techniques:**

* Loading of dataset
* Reshaping the data

X\_train\_flat = X\_train.reshape(X\_train.shape[0], -1) / 255.0

X\_test\_flat = X\_test.reshape(X\_test.shape[0], -1) / 255.0

* Splitting into train and test dataset(considering 20% for test data)

X\_train\_split, X\_val, y\_train\_split, y\_val = train\_test\_split(X\_train\_flat, y\_train, test\_size=0.2, random\_state=42)

1. **ML Models Applied:**

a.KNN :

knn\_model = KNeighborsClassifier(n\_neighbors=3)

knn\_model.fit(X\_train\_split, y\_train\_split)

y\_val\_pred\_knn = knn\_model.predict(X\_val)

knn\_acc = accuracy\_score(y\_val, y\_val\_pred\_knn)

print(f'KNN Validation Accuracy: {knn\_acc:.4f}')

print(classification\_report(y\_val, y\_val\_pred\_knn))

sns.heatmap(confusion\_matrix(y\_val, y\_val\_pred\_knn), annot=True, fmt='d', cmap='Blues')

plt.title("KNN Confusion Matrix")

plt.show()

b.SVC-Linear:

svm\_model = SVC(kernel='linear')

svm\_model.fit(X\_train\_split, y\_train\_split)

y\_val\_pred\_svm = svm\_model.predict(X\_val)

svm\_acc = accuracy\_score(y\_val, y\_val\_pred\_svm)

print(f'SVM Validation Accuracy: {svm\_acc:.4f}')

print(classification\_report(y\_val, y\_val\_pred\_svm))

sns.heatmap(confusion\_matrix(y\_val, y\_val\_pred\_svm), annot=True, fmt='d', cmap='Greens')

plt.title("SVM Confusion Matrix")

plt.show()

C. SVC-rbf:

svm\_model = SVC(kernel='rbf')

svm\_model.fit(X\_train\_split, y\_train\_split)

y\_val\_pred\_svm = svm\_model.predict(X\_val)

svm\_acc = accuracy\_score(y\_val, y\_val\_pred\_svm)

print(f'SVM Validation Accuracy: {svm\_acc:.4f}')

print(classification\_report(y\_val, y\_val\_pred\_svm))

sns.heatmap(confusion\_matrix(y\_val, y\_val\_pred\_svm), annot=True, fmt='d', cmap='Greens')

plt.title("SVM Confusion Matrix")

plt.show()

1. **Table Comparing Performance of different models:**

Accuracy score

KNN=0.9727

SVM-Linear=0.9352

SVM-RBF=0.9741

1. **Best model and why?**

Model -SVM(kernel -rbf)

Accuracy score : 0.9776

1. **Extension and Future Work:**

To make use of neural networks and check for an improvement in the accuracy

Score and build a more effective model.