Accuracy Report for Digit Recognition Models

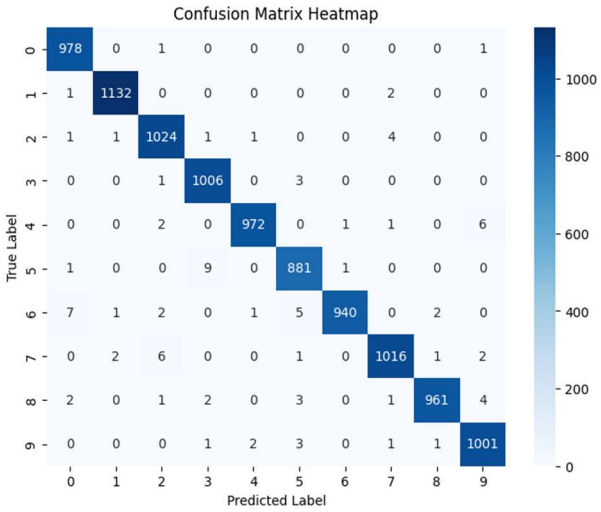
1. Convolutional Neural Network (CNN)

Model Architecture:

```
model = models.Sequential([
    layers.Conv2D(filters=16, kernel_size=(3,3),activation='relu', input_shape=(28,28,1)),
    layers.MaxPooling2D((2,2)),

    layers.Conv2D(filters=32, kernel_size=(3,3),activation='relu'),
    layers.MaxPooling2D((2,2)),

    layers.Conv2D(filters=64, kernel_size=(3,3),activation='relu'),

    layers.Flatten(),
    layers.Dense(64,activation='relu'),
    layers.Dense(10,activation='softmax')
])
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
model.fit(X_train, y_train, epochs=10)
```

Performance Metrics:

Heatmap:



Classification Report:

	precision	recall	f1-score	support
0	0.99	1.00	0.99	980
1	1.00	1.00	1.00	1135
2	0.99	0.99	0.99	1032
3	0.99	1.00	0.99	1010
4	1.00	0.99	0.99	982
5	0.98	0.99	0.99	892
6	1.00	0.98	0.99	958
7	0.99	0.99	0.99	1028
8	1.00	0.99	0.99	974
9	0.99	0.99	0.99	1009
accuracy			0.99	10000
macro avg	0.99	0.99	0.99	10000
weighted avg	0.99	0.99	0.99	10000

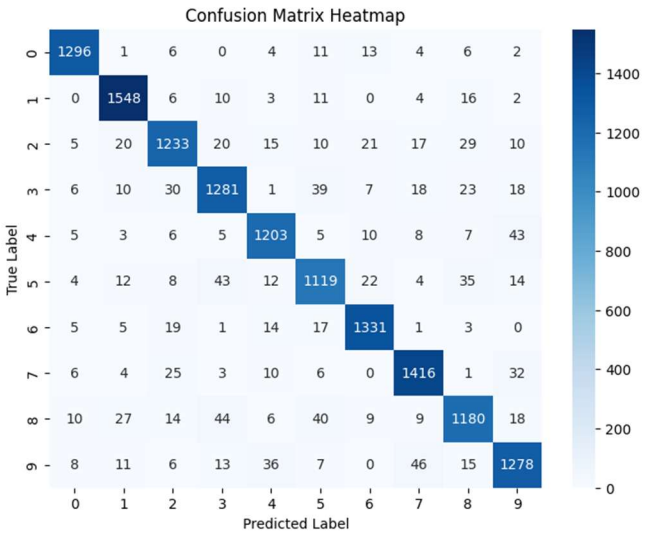
2. Logistic Regression

Model Training:

```python
MODEL_FILENAME = "mnist_logistic_regression2.pkl"
try:
    print(f"Loading pre-trained model from {MODEL_FILENAME}...")
    model = joblib.load(MODEL_FILENAME)
except FileNotFoundError:
    print("No pre-trained model found. Training a new model...")
    model = LogisticRegression(solver='lbfgs', max_iter=500, random_state=42)
    model.fit(X_train, y_train)
    joblib.dump(model, MODEL_FILENAME)  # Save model for later use
    print(f"Model saved as {MODEL_FILENAME}")
```

Performance Metrics:

Heatmap:



Classification Report:

	precision	recall	f1-score	support
0	0.9636	0.9650	0.9643	1343
1	0.9433	0.9675	0.9553	1600
2	0.9113	0.8935	0.9023	1380
3	0.9021	0.8939	0.8980	1433
4	0.9225	0.9290	0.9257	1295
5	0.8846	0.8790	0.8818	1273
6	0.9420	0.9534	0.9477	1396
7	0.9273	0.9421	0.9347	1503
8	0.8973	0.8696	0.8832	1357
9	0.9019	0.9000	0.9010	1420
accuracy			0.9204	14000
macro avg	0.9196	0.9193	0.9194	14000
weighted avg	0.9201	0.9204	0.9202	14000

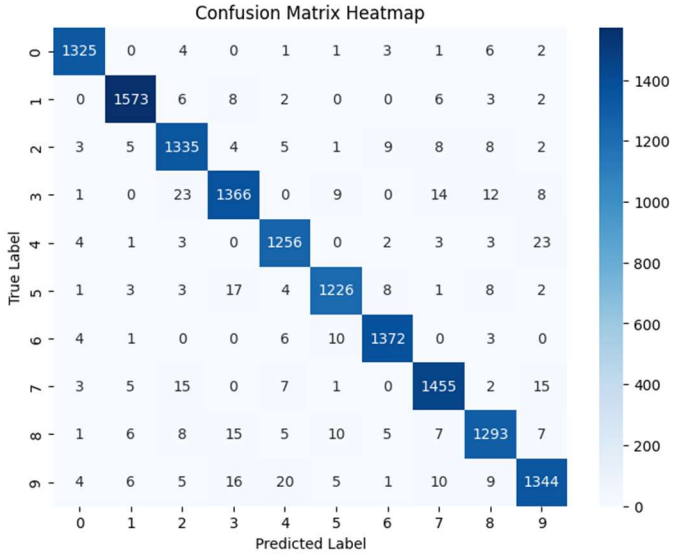
3. Random Forest

Model Training:

```
MODEL_FILENAME = "mnist_random_forest2.pkl"
try:
    print(f"Loading pre-trained model from {MODEL_FILENAME}...")
    model = joblib.load(MODEL_FILENAME)
except FileNotFoundError:
    print("No pre-trained model found. Training a new model...")
    model = RandomForestClassifier(n_estimators=75, random_state=42, n_jobs=-1)
    model.fit(X_train, y_train)
    joblib.dump(model, MODEL_FILENAME)  # Save model for later use
    print(f"Model saved as {MODEL_FILENAME}")
```

Performance Metrics:

Heatmap:



Classification Report:

	precision	recall	f1-score	support
0	0.9844	0.9866	0.9855	1343
1	0.9831	0.9831	0.9831	1600
2	0.9522	0.9674	0.9597	1380
3	0.9579	0.9532	0.9556	1433
4	0.9617	0.9699	0.9658	1295
5	0.9707	0.9631	0.9669	1273
6	0.9800	0.9828	0.9814	1396
7	0.9668	0.9681	0.9674	1503
8	0.9599	0.9528	0.9564	1357
9	0.9566	0.9465	0.9515	1420
accuracy			0.9675	14000
macro avg	0.9673	0.9674	0.9673	14000
weighted avg	0.9675	0.9675	0.9675	14000

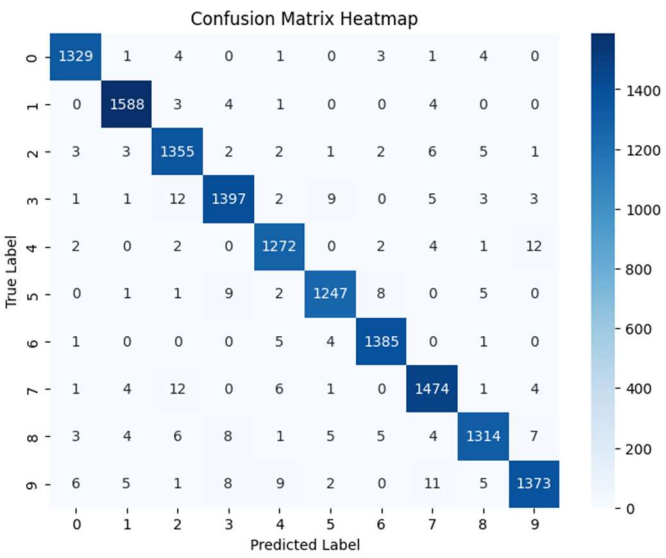
4. Support Vector Machine (SVM)

Model Training:

```
MODEL_FILENAME = "mnist_svm.pkl"
try:
    print(f"Loading pre-trained model from {MODEL_FILENAME}...")
    model = joblib.load(MODEL_FILENAME)
except FileNotFoundError:
    print("No pre-trained model found. Training a new model...")
    model = SVC(kernel='rbf', C=10, gamma=0.01, random_state=42)
    model.fit(X_train, y_train)
    joblib.dump(model, MODEL_FILENAME)  # Save model for later use
    print(f"Model saved as {MODEL_FILENAME}")
```

Performance Metrics:

Heatmap:



Classification Report:

	precision	recall	f1-score	support
0	0.9874	0.9896	0.9885	1343
1	0.9882	0.9925	0.9903	1600
2	0.9706	0.9819	0.9762	1380
3	0.9783	0.9749	0.9766	1433
4	0.9777	0.9822	0.9800	1295
5	0.9827	0.9796	0.9811	1273
6	0.9858	0.9921	0.9889	1396
7	0.9768	0.9807	0.9788	1503
8	0.9813	0.9683	0.9748	1357
9	0.9807	0.9669	0.9738	1420
accuracy			0.9810	14000
macro avg	0.9809	0.9809	0.9809	14000
weighted avg	0.9810	0.9810	0.9810	14000

Here's the summary table comparing the models based on Accuracy, Precision, Recall, and F1-score:

Model	Accuracy	Precision	Recall	F1-score
CNN	99.0%	99.0%	99.0%	99.0%
SVM	98.1%	98.1%	98.1%	98.1%
Random Forest	96.75%	96.73%	96.74%	96.73%
Logistic Regression	92.04%	91.96%	91.93%	91.94%

Observations:

- CNN outperforms all models with the highest accuracy (99%) and consistent precision, recall, and F1-score.
- SVM achieves the second-best performance (98.1%), making it a solid alternative.
- Random Forest performs decently (96.75%) but is slightly behind SVM.
- Logistic Regression has the lowest accuracy (92.04%), making it the least effective for this task.
