

Name:

Matthew Tang

Netid:

mhtang2

## CS 441 - HW2: PCA and Linear Models

Complete the sections below. You do not need to fill out the checklist.

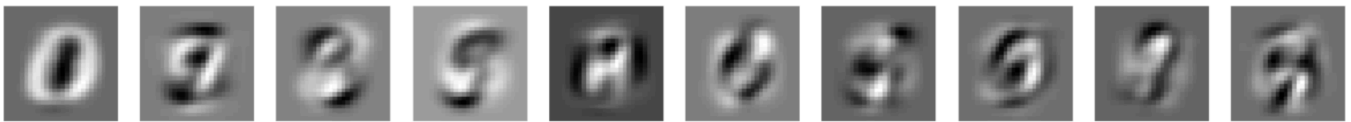
Total Points Available

[ ] / 160

1. PCA on MNIST
  - a. Display 10 principal component vectors [ ] / 5
  - b. Display scatterplot [ ] / 5
  - c. Plot cumulative explained variance [ ] / 5
  - d. Compression and 1-NN experiment [ ] / 15
2. MNIST Classification with Linear Models
  - a. LLR / SVM error vs training size [ ] / 20
  - b. Error visualization [ ] / 10
  - c. Parameter selection experiments [ ] / 15
3. Temperature Regression
  - a. Linear regression test [ ] / 10
  - b. Feature selection results [ ] / 15
4. Stretch Goals
  - a. PR and ROC curves [ ] / 10
  - b. Visualize weights [ ] / 10
  - c. Other embeddings [ ] / 15
  - d. One city is all you need [ ] / 15
  - e. SVM with RBF kernel [ ] / 10

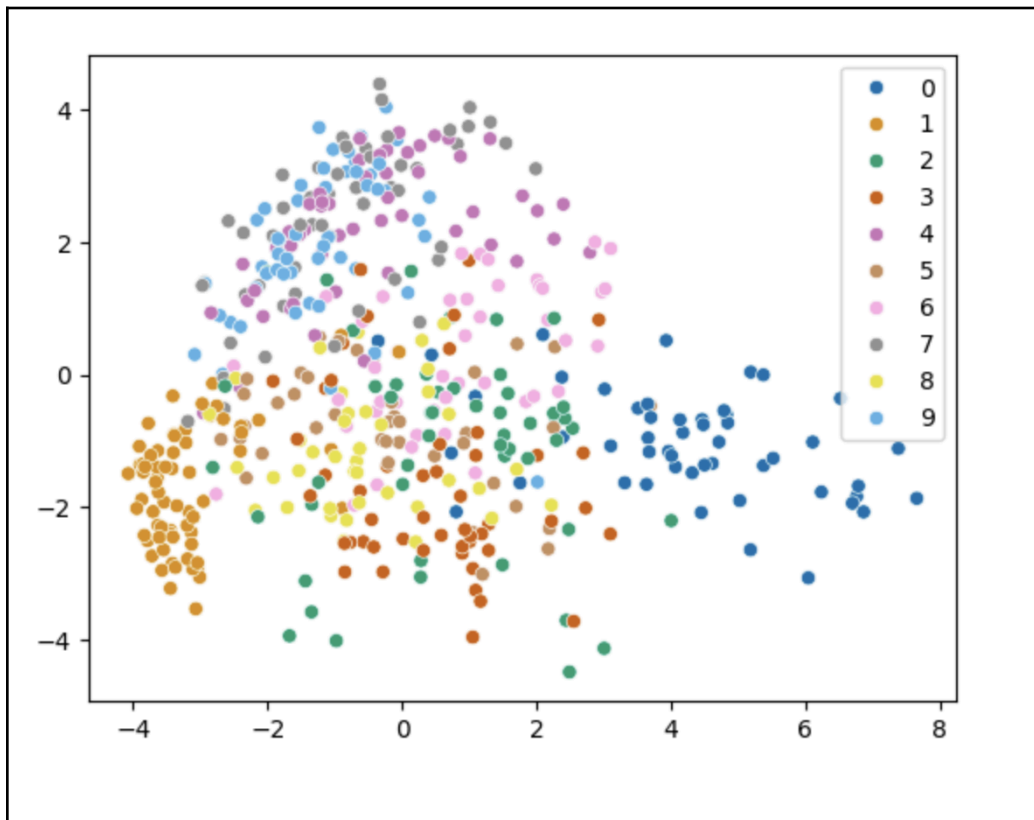
### 1. PCA on MNIST

#### a. Display 10 principal component vectors

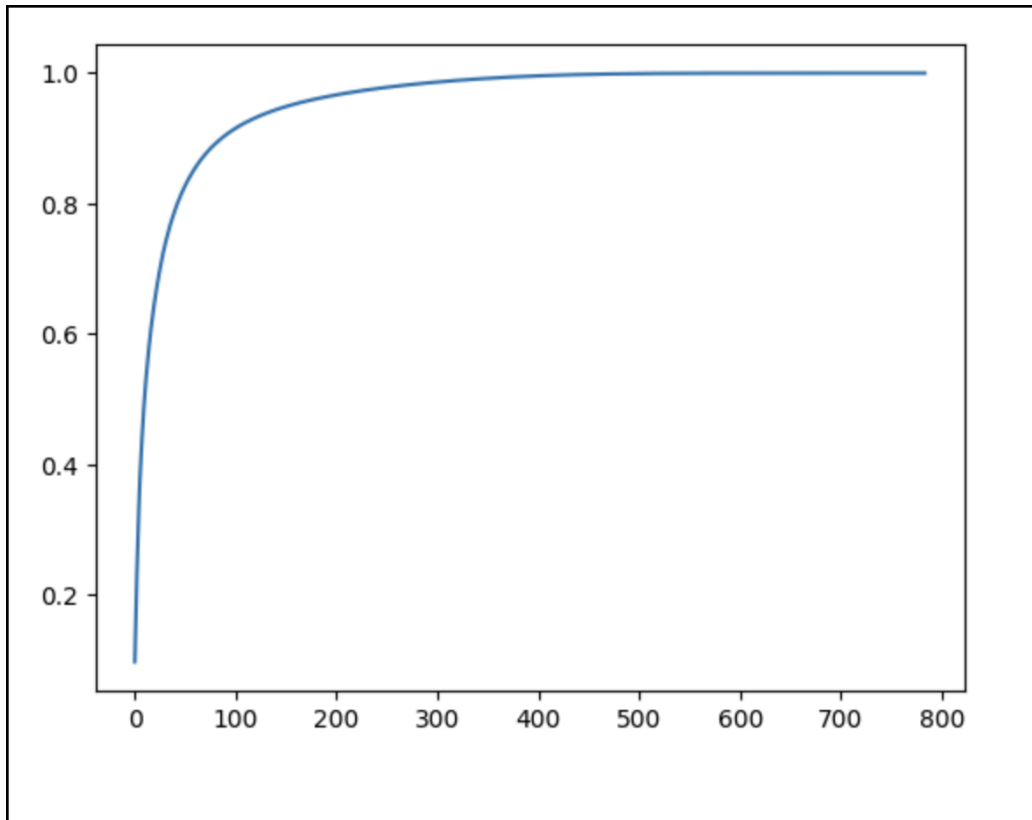


**b. Display scatterplot**

Scatterplot `x_train[:500]` for the first two PCA dimensions. Show a different color for each label.



**c. Plot cumulative explained variance**



#### d. Compression and 1-NN experiment

Number of components selected

	Total Time (s)	Test Error (%)	Dimensions
Brute Force (PCA)	4.545	2.73%	87
Brute Force	36.031	3.09%	784

## 2. MNIST Classification with Linear Models

#### a. LLR / SVM error vs training size



Test error (%)



# training samples	LLR	SVM
100	32.5%	32.4%
1,000	13.64 %	16.11 %

10,000	9.5 %	11.12 %
60,000	7.38 %	8.17 %











**b. Error visualization**

LLR















LLR Most confident correct:



LLR Most confident incorrect:



SVM

Most confident correct:



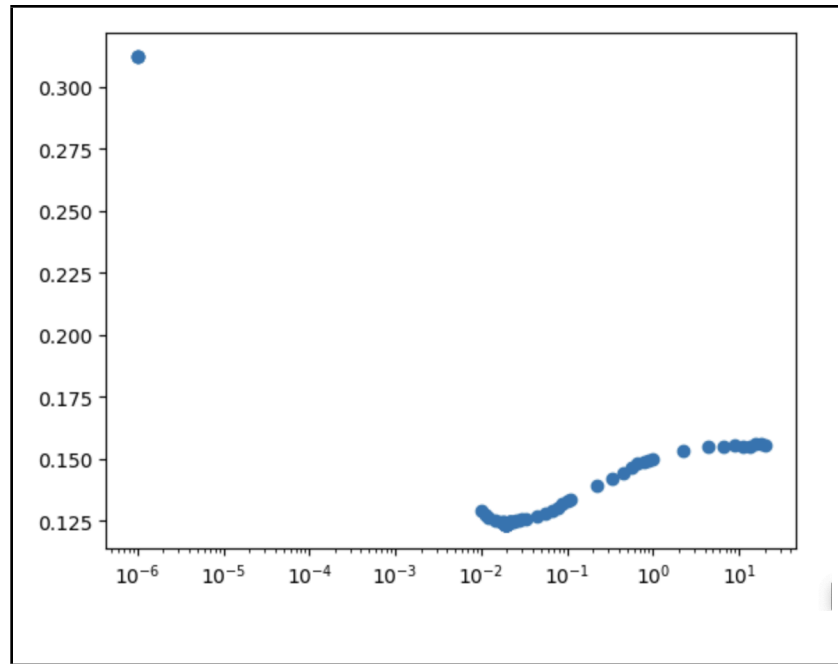
Most confident incorrect:



**c. Parameter selection experiments**

	SVM
Best C value	0.0196
Validation error (%)	12.35 %
Test error (%)	13.58 %

Plot C value vs validation error for values tested



### 3. Temperature Regression

#### a. Linear regression test

Test RMSE

	Linear regression
Original features	2.1608
Normalized features	2.1631

Why might normalizing features in this way not be as helpful as it is for KNN?

Because KNN uses distances, it is sensitive to the scale of features. Linear regression can learn weights to scale features so it is less sensitive. This normalization essentially centers/rescales features.

#### b. Feature selection results

Feature Rank	Feature number	City	Day
1	334	Chicago	-1
2	347	Minneapolis	-1
3	405	Grand Rapids	-1
4	366	Kansas City	-1
5	361	Cleveland	-1
6	307	Omaha	-2
7	367	Indianapolis	-1
8	264	Minneapolis	-2
9	9	Boston	-5
10	236	Springfield	-3

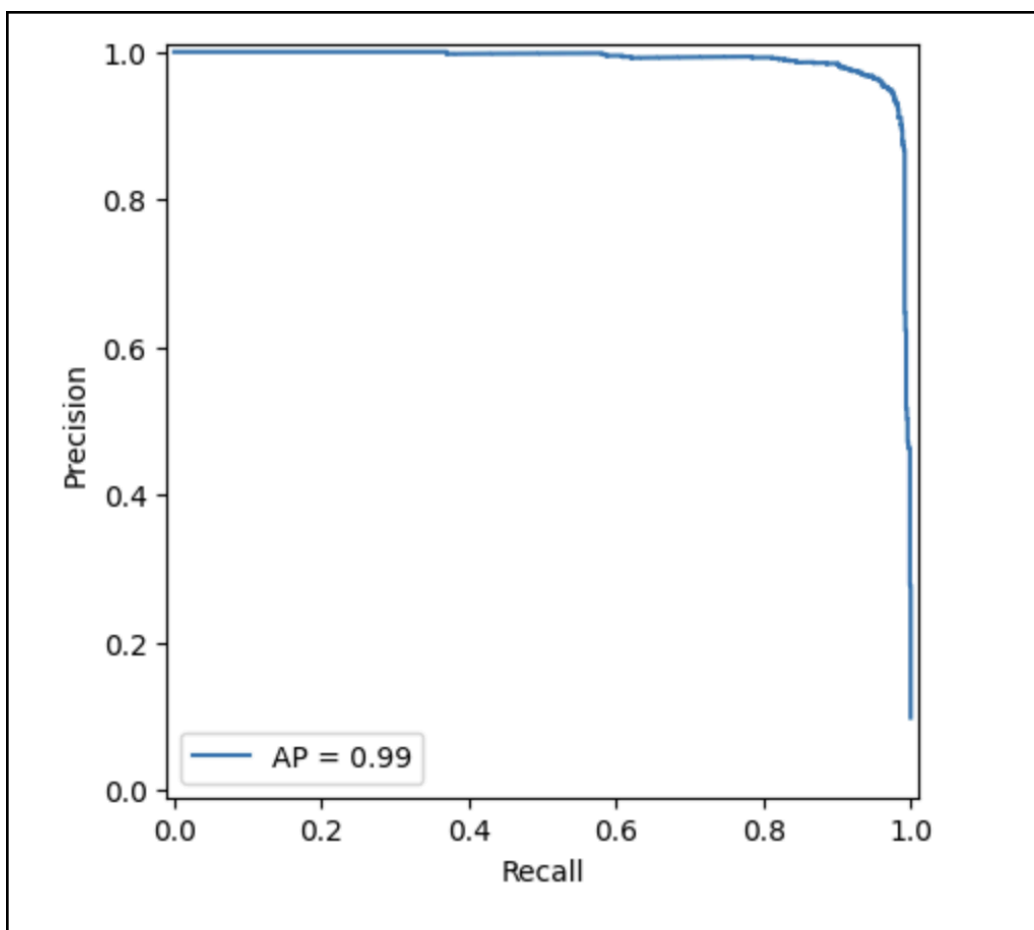
Test error using only the 10 most important features for regression

	Linear Regression
RMS Error	2.0621

#### 4. Stretch Goals

##### a. PR and ROC curves

PR plot

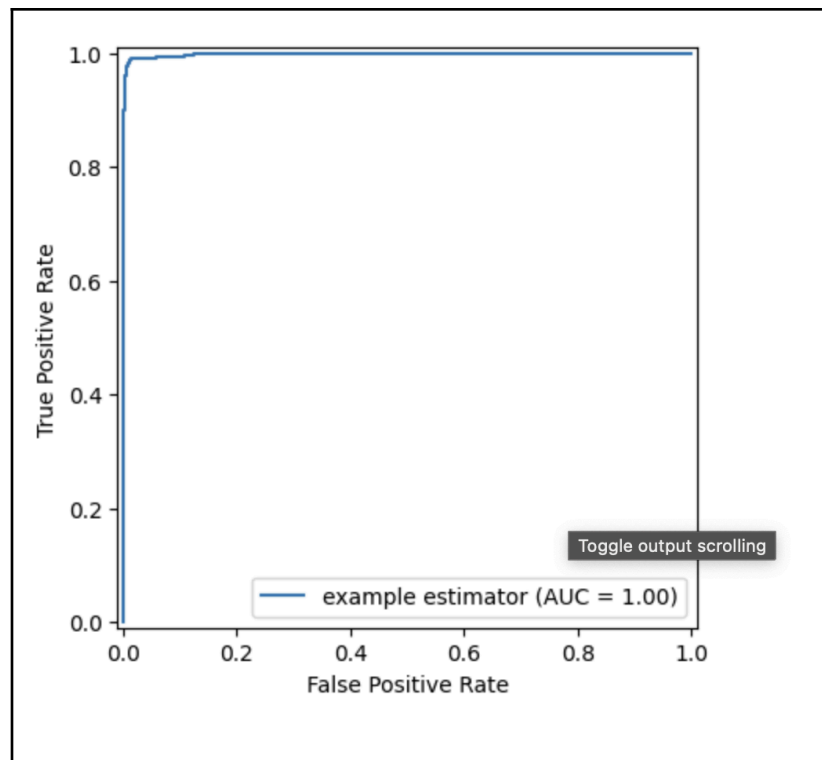


Average Precision

0.98883

ROC plot



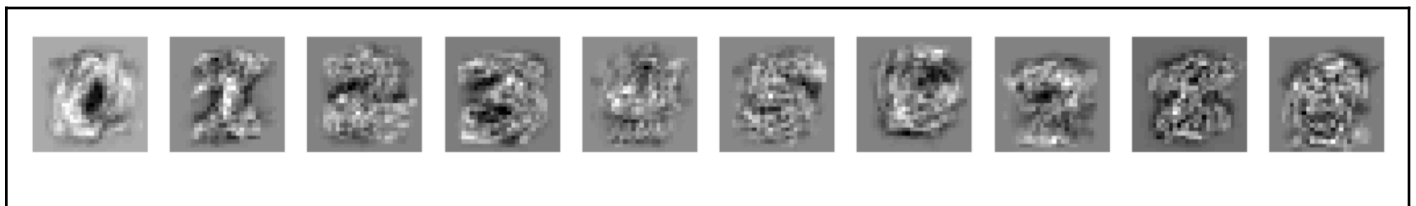


Area under the curve (AUC)

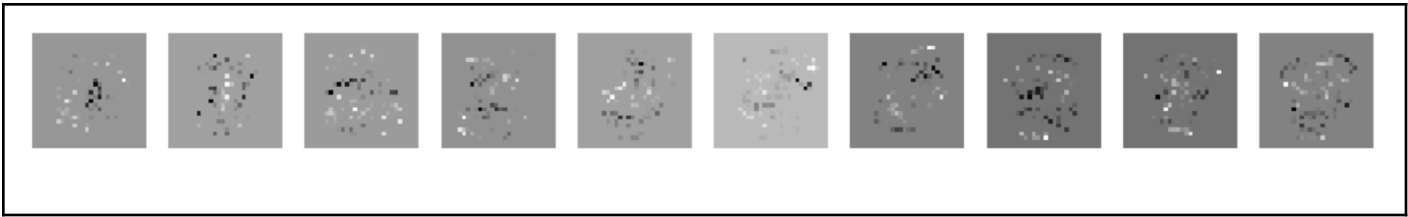
0.9983

**b. Visualize weights**

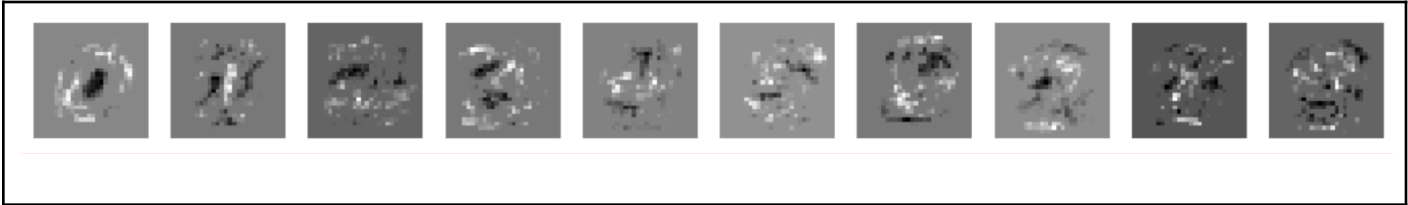
LLR - L2



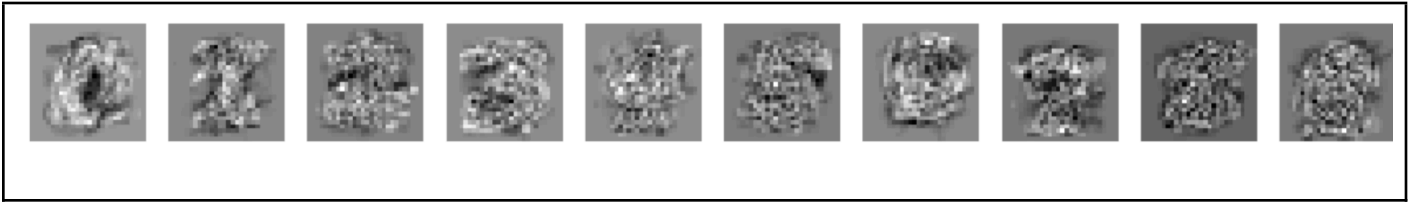
LLR - L1



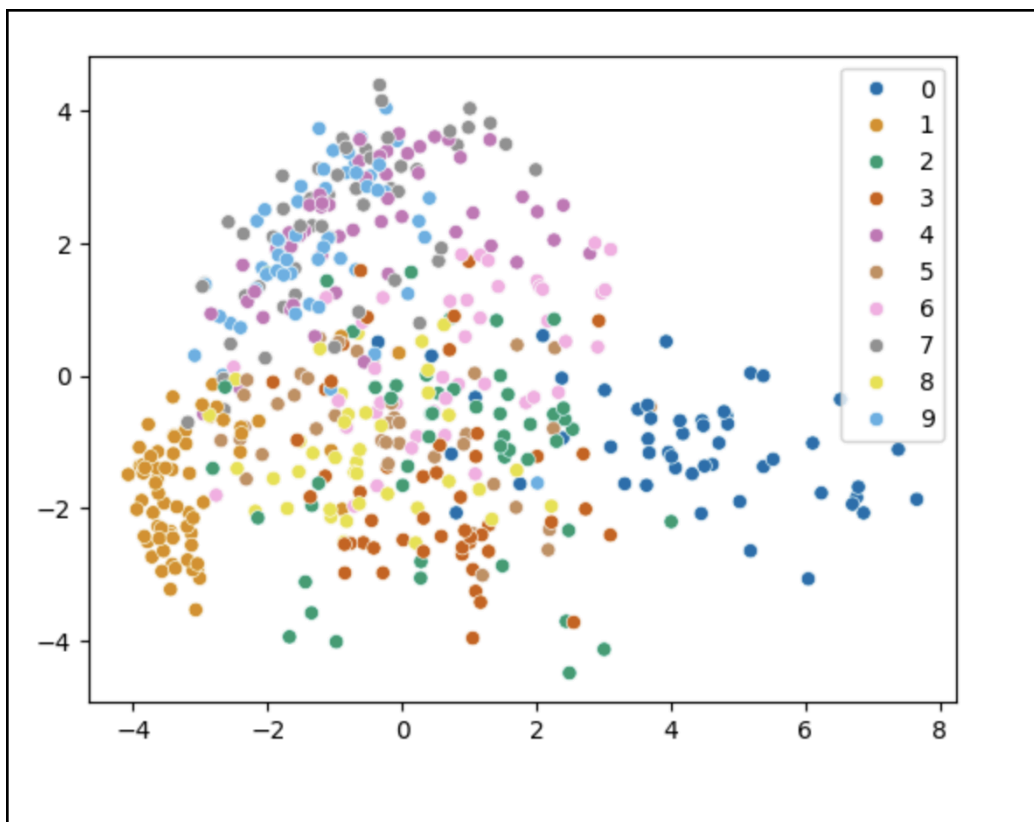
LLR - elastic



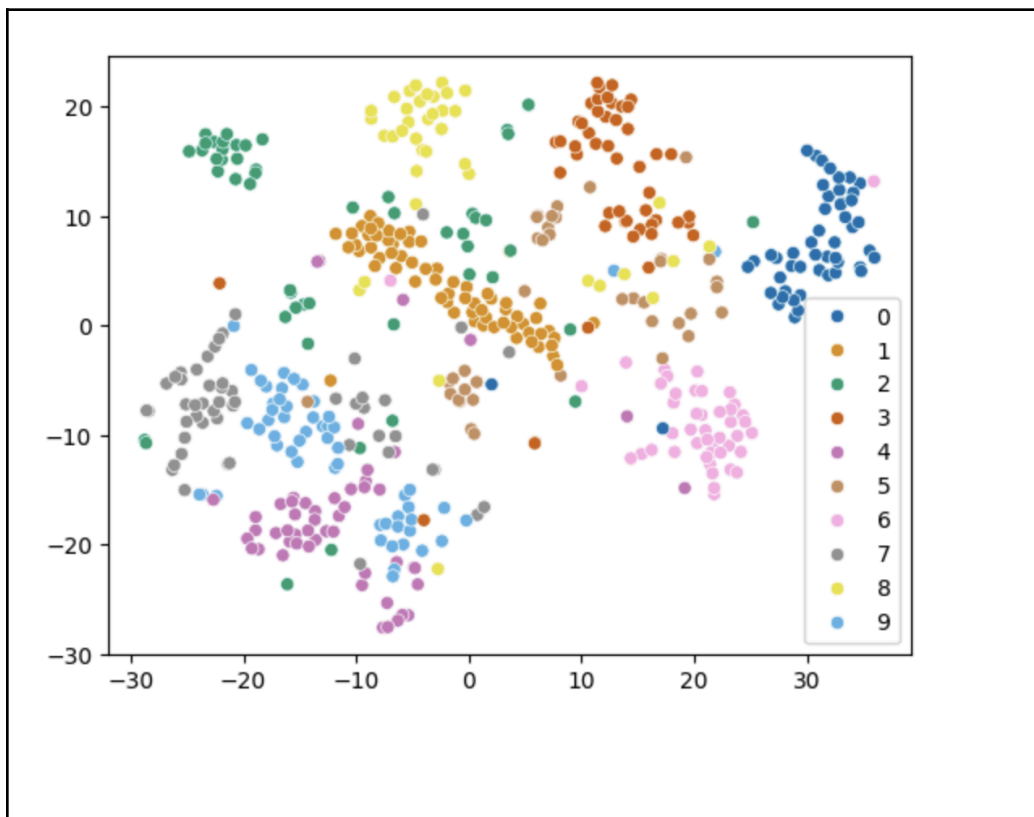
SVM



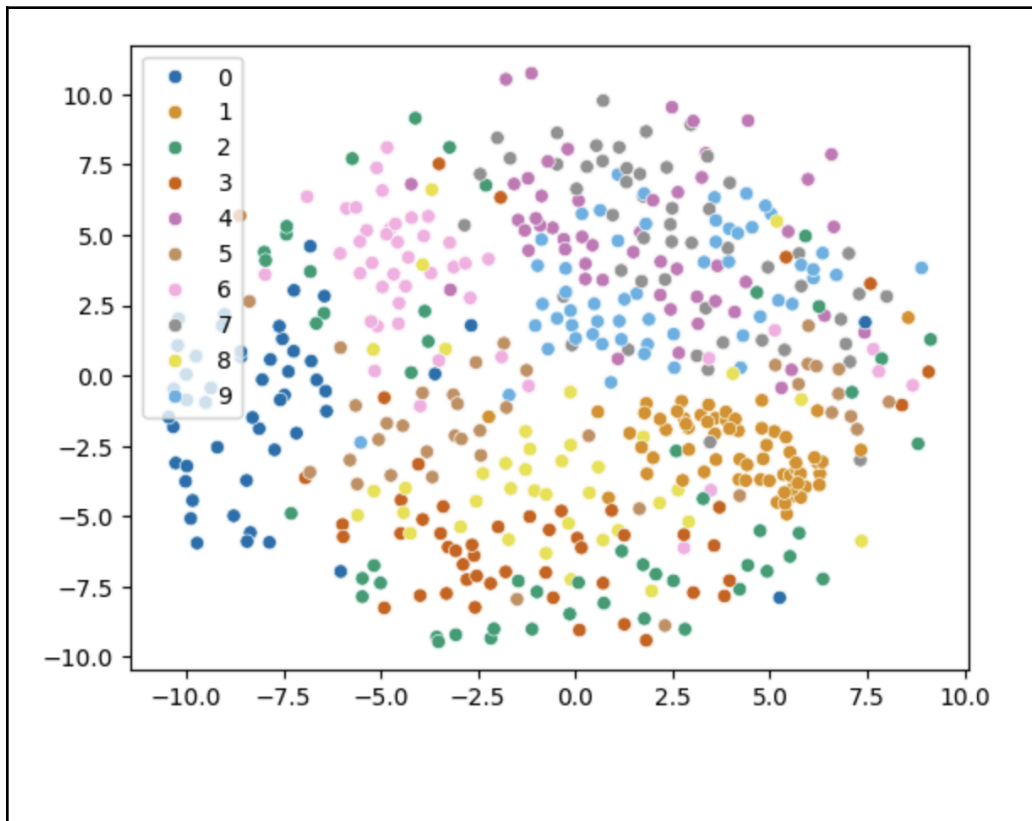
**c. Other embeddings**  
Display 2+ plots for TSNE, MDA, and/or LDA, and copy PCA plot from 1b here.  
PCA



TNSE



MDS



**d. One city is all you need**

City

St. Louis

Test error using features only from that city

3.1263 RMSE

Explain your process (in words):

Iterated through every city, and trained a Ridge model (no feature normalization, default parameters) using only days from that city as input features. Then I took the validation error for each of these models, and took the city with the minimum validation error.

#### e. Compare linear SVM and SVM with RBF kernel

Test accuracy (%)

# training samples	SVM-Linear	SVM-RBF
100	32.4%	34.4%
1,000	16.11 %	9.17%
10,000	11.12 %	4.06%
60,000	8.17 %	2.08%

#### Acknowledgments / Attribution

List any outside sources for code or ideas or "None".  
Stackoverflow + sklearn docs