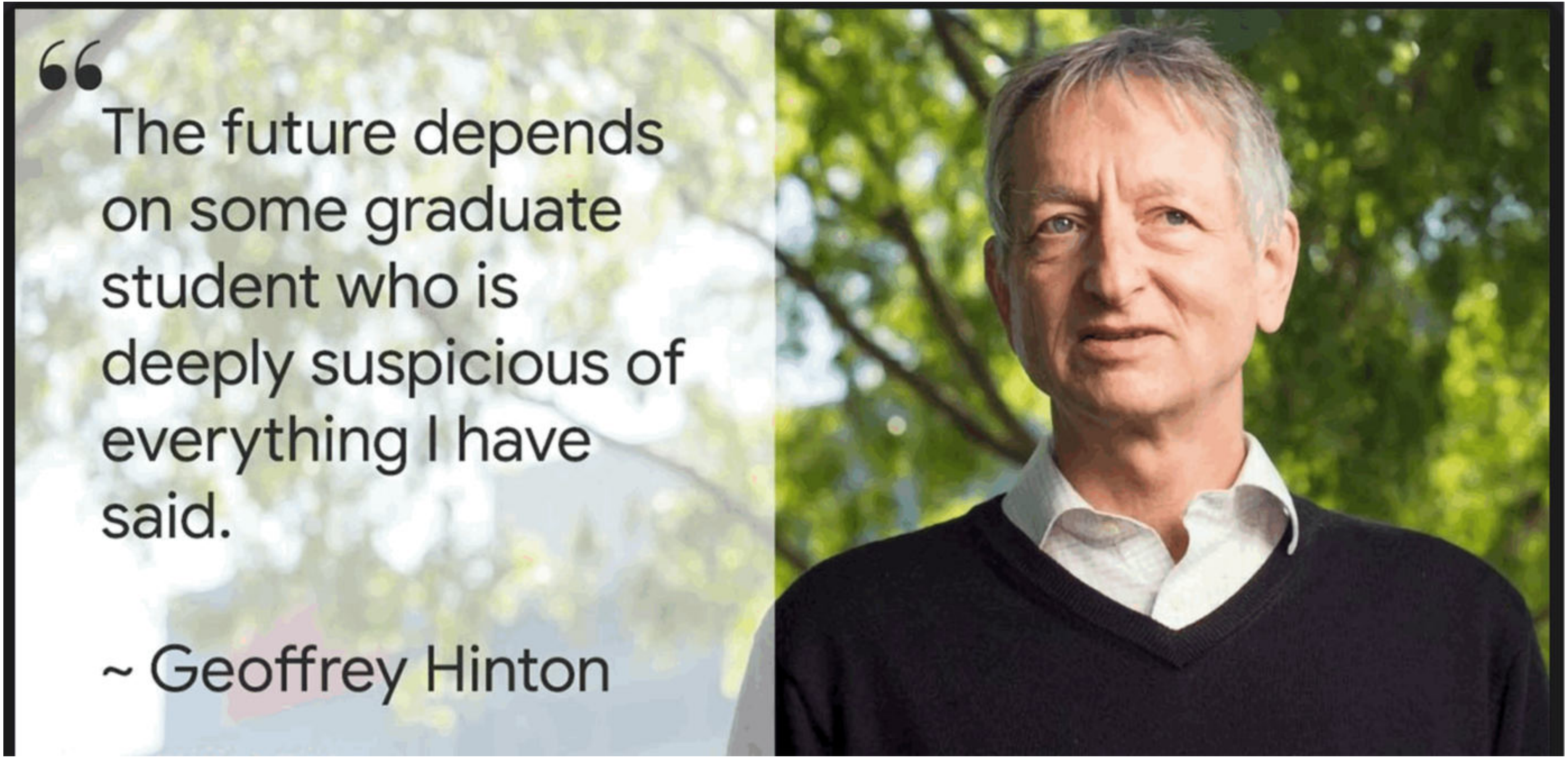
Team Hinton

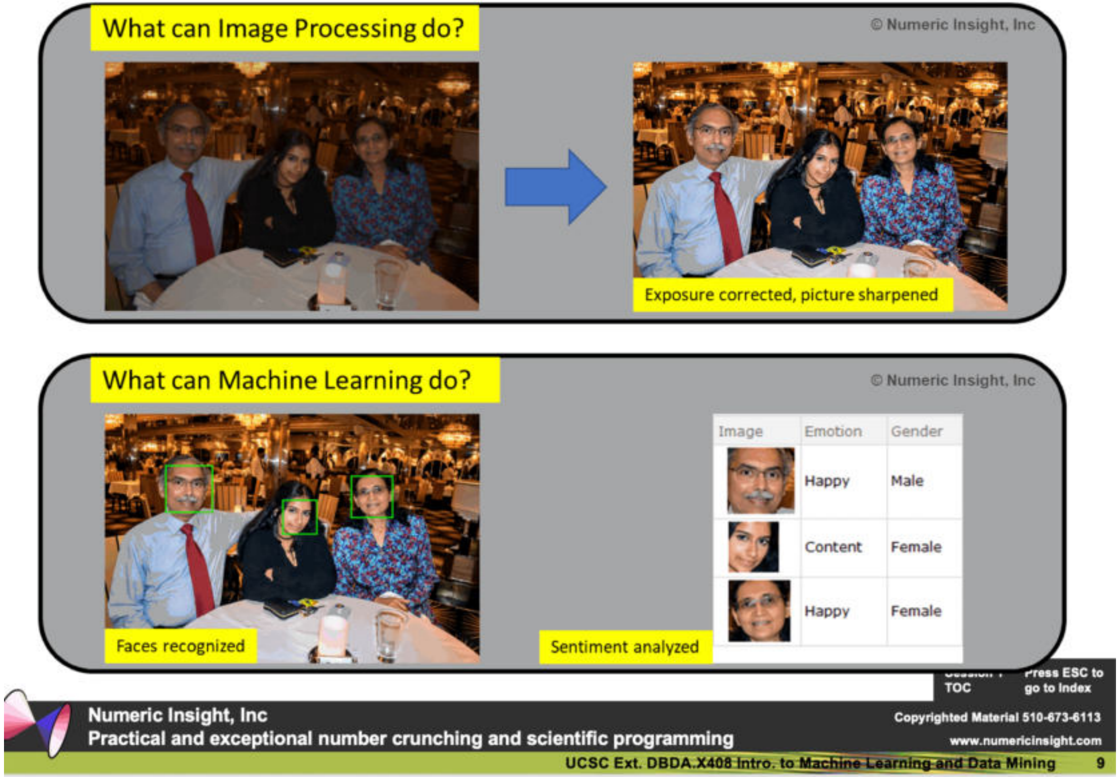
Optical Digit Recognition (ODR)

Carisa Yao
Chandramouly Rathinasababathy
Ganesh Jothikumar
Hyunjung Byun
Jong Lin
Ravisankar Videla



> The future depends on some graduate student who is deeply suspicious of everything I have said.
>
> ~ Geoffrey Hinton

Machine Learning versus Other things



What can Image Processing do?

© Numeric Insight, Inc

Exposure corrected, picture sharpened

What can Machine Learning do?

© Numeric Insight, Inc

Image	Emotion	Gender
	Happy	Male
	Content	Female
	Happy	Female

Faces recognized

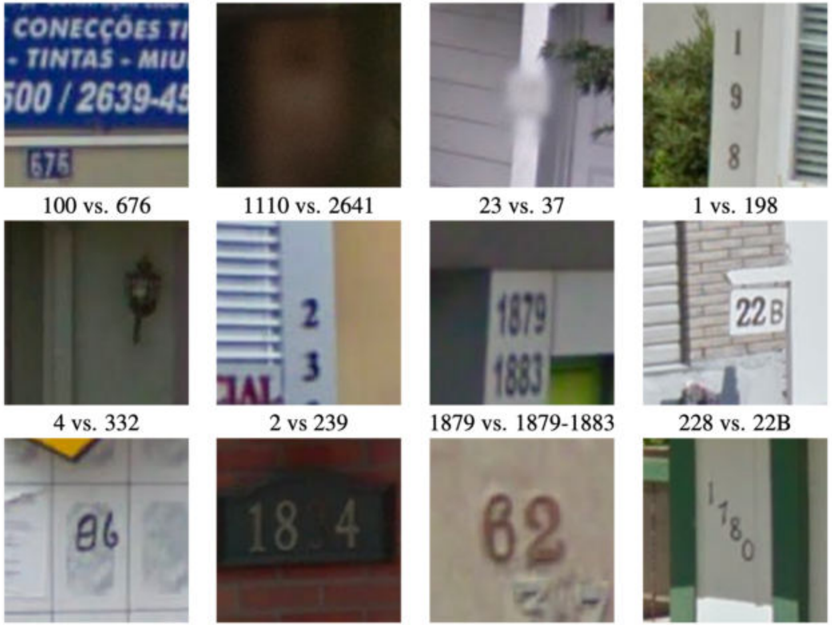
Sentiment analyzed

Problem Statement

- Recognizing handwritten / printed numbers from an image

- Practical applications include street view number identification

Some more patterns in street view



100 vs. 676

1110 vs. 2641

23 vs. 37

1 vs. 198

4 vs. 332

2 vs 239

1879 vs. 1879-1883

228 vs. 22B

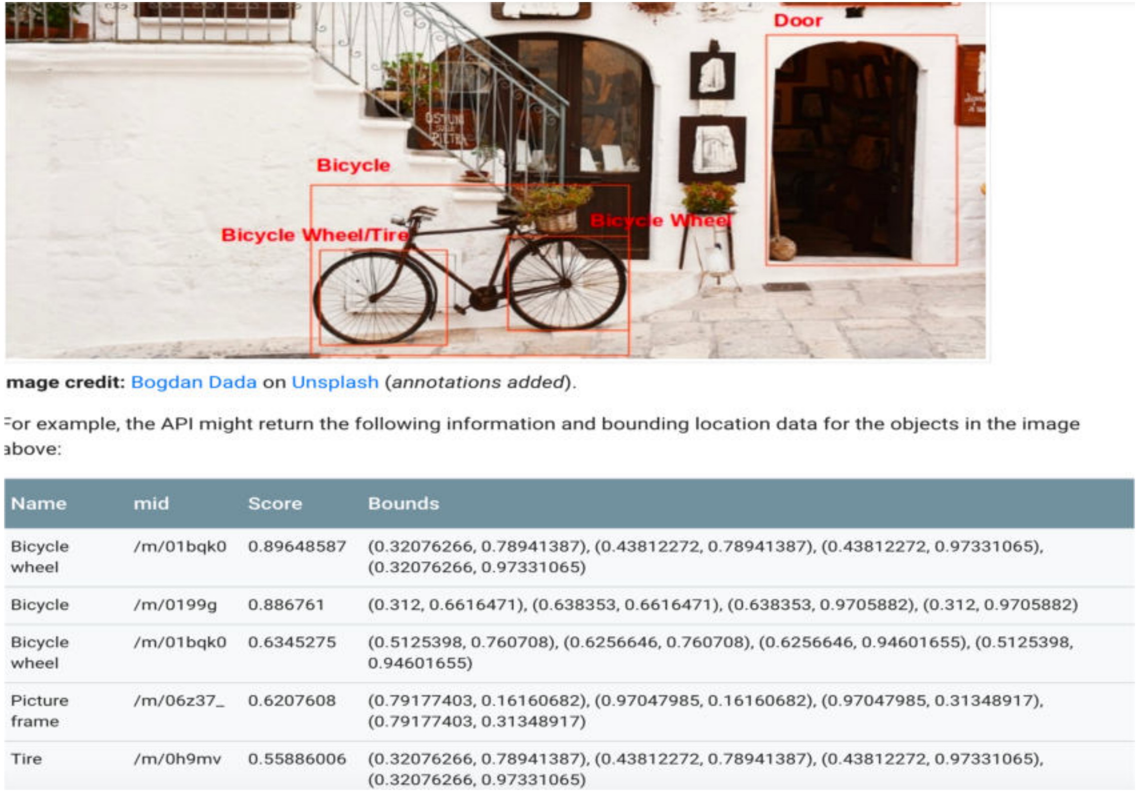
96 vs. 86

1844 vs. 184

62 vs. 62-37

1180 vs. 1780

Extended Problem Space



mage credit: Bogdan Dada on Unsplash (*annotations added*).

or example, the API might return the following information and bounding location data for the objects in the image bove:

Name	mid	Score	Bounds
Bicycle wheel	/m/01bqk0	0.89648587	(0.32076266, 0.78941387), (0.43812272, 0.78941387), (0.43812272, 0.97331065), (0.32076266, 0.97331065)
Bicycle	/m/0199g	0.886761	(0.312, 0.6616471), (0.638353, 0.6616471), (0.638353, 0.9705882), (0.312, 0.9705882)
Bicycle wheel	/m/01bqk0	0.6345275	(0.5125398, 0.760708), (0.6256646, 0.760708), (0.6256646, 0.94601655), (0.5125398, 0.94601655)
Picture frame	/m/06z37_	0.6207608	(0.79177403, 0.16160682), (0.97047985, 0.16160682), (0.97047985, 0.31348917), (0.79177403, 0.31348917)
Tire	/m/0h9mv	0.55886006	(0.32076266, 0.78941387), (0.43812272, 0.78941387), (0.43812272, 0.97331065), (0.32076266, 0.97331065)

Objective

Overall we tried to achieve the following through the course of this project

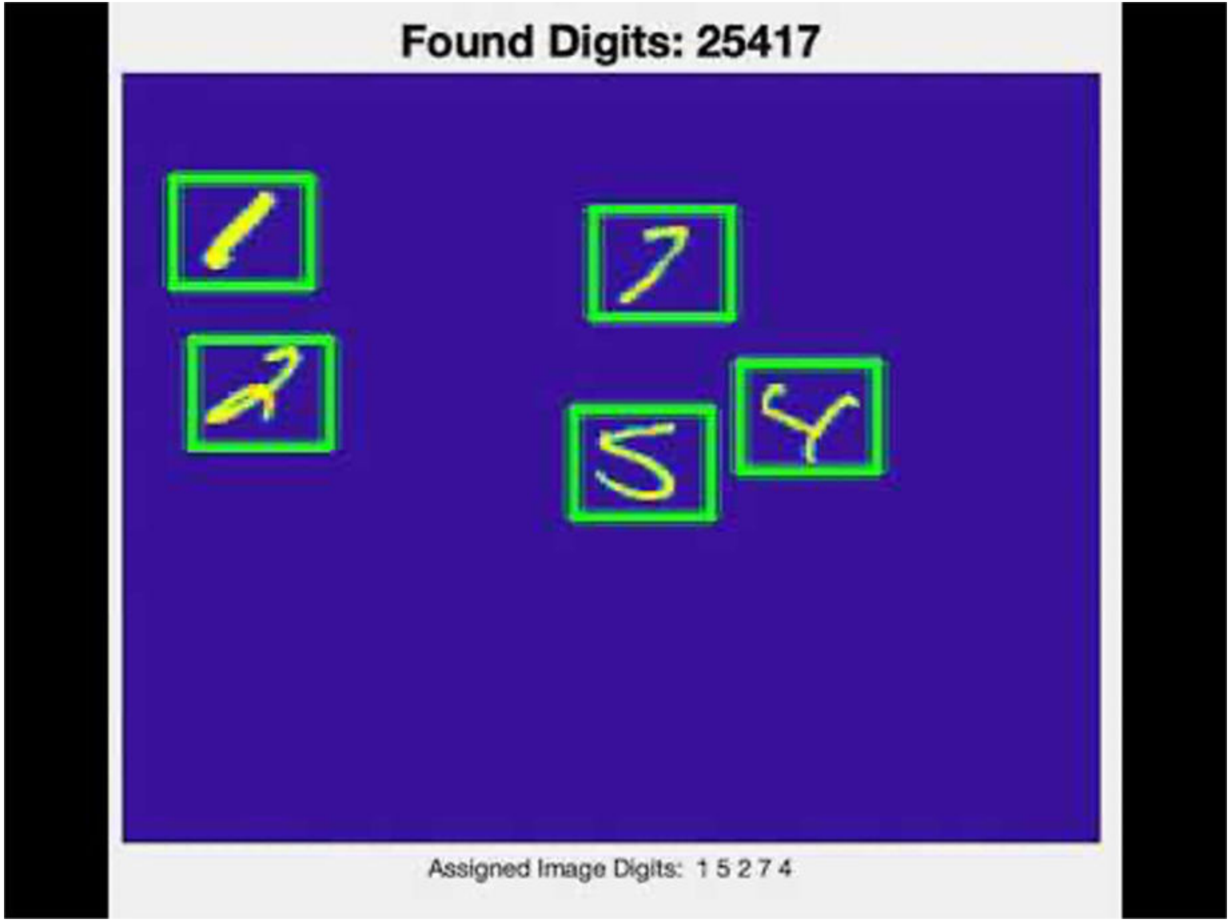
- A technique to do the multi number detection with its bounding box from a given image.
- Evaluate few classifiers and derive interesting findings

Dataset Challenges & Using MNIST

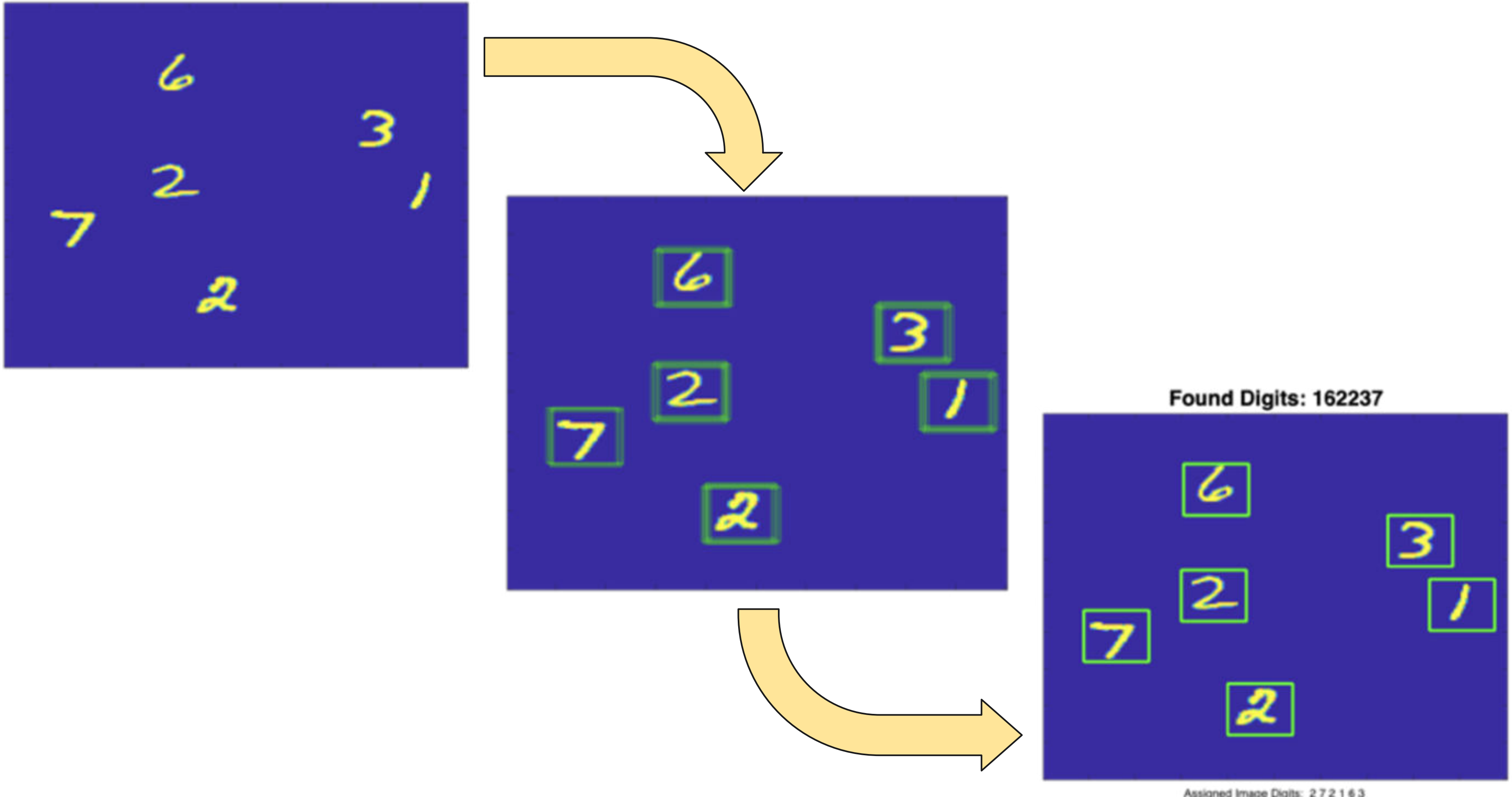
- Realworld images from Google Street view imagery
  - Images not readily available for processing
  - Heavy preprocessing needed for consumption

- Why we chose MNIST?
  - Core of the problem is number detection
  - MNIST is a rich resource for numbers
  - Can focus more on the problem and classification

Demo - Video



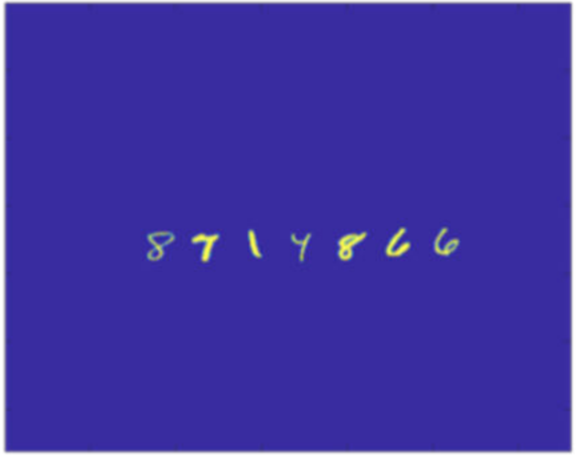
Demo - Screenshots



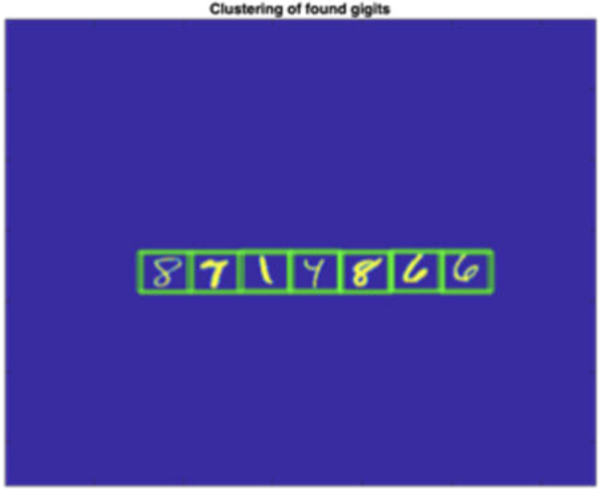
Found Digits: 162237

Assigned Image Digits: 2 7 2 1 6 3

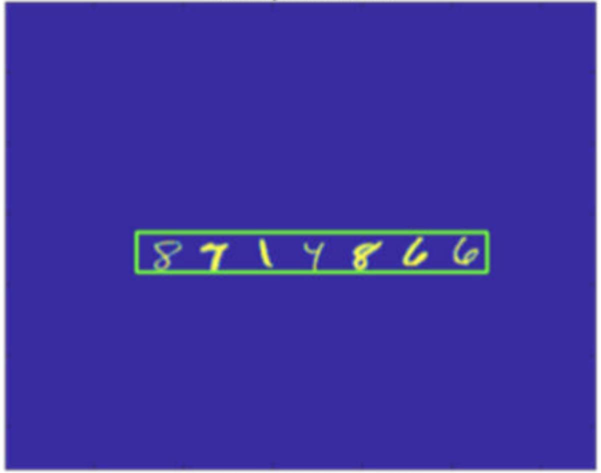
Demo - Screenshots



Clustering of found gigits

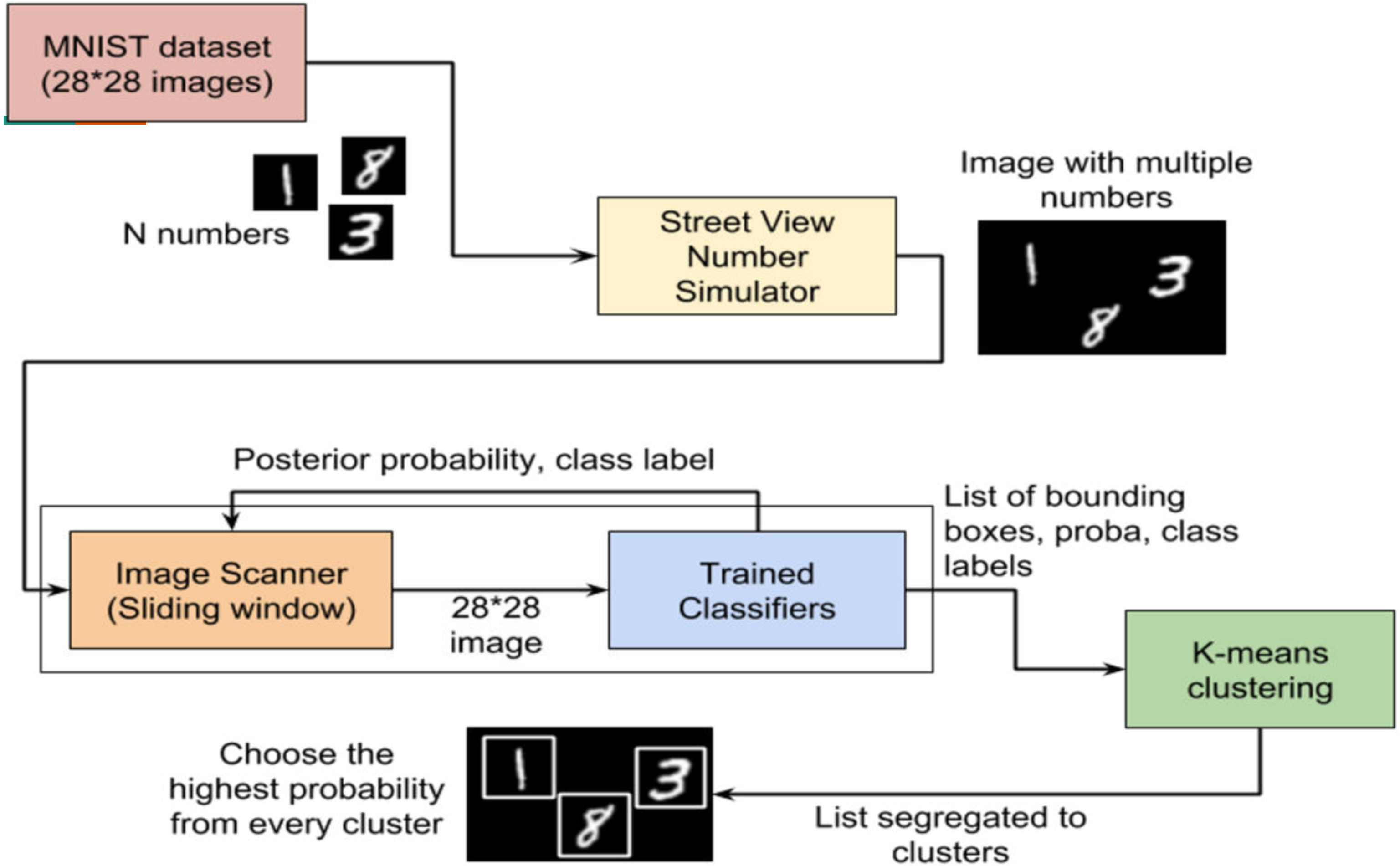


Found Zip Code: 8714866



Assigned Image Digits: 8 7 1 4 8 6 6

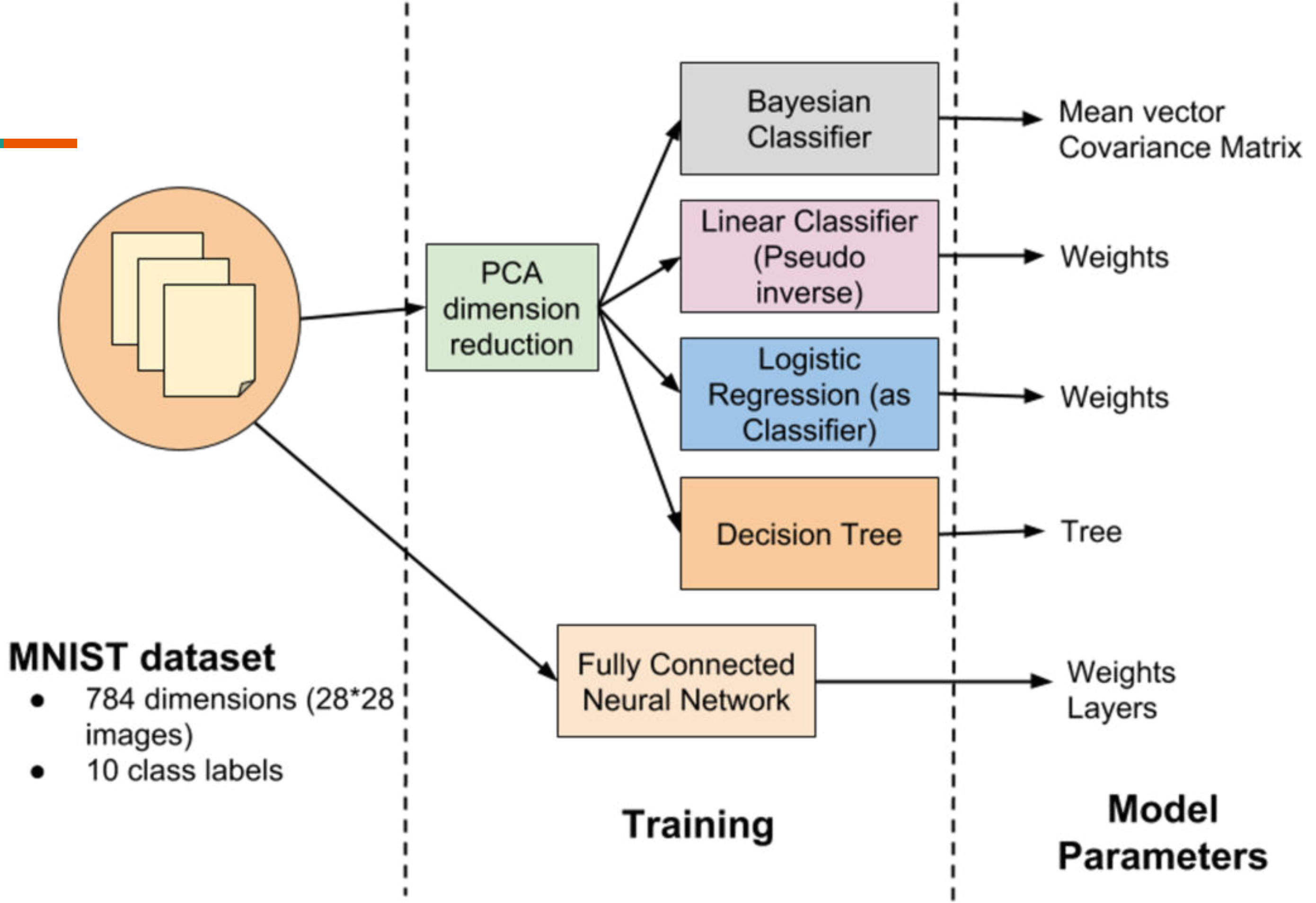
Overall Approach


MNIST dataset (28*28 images)
N numbers
Street View Number Simulator
Image with multiple numbers
Posterior probability, class label
Image Scanner (Sliding window)
28*28 image
Trained Classifiers
List of bounding boxes, proba, class labels
K-means clustering
List segregated to clusters
Choose the highest probability from every cluster

Techniques Used

- Image generator with random number of digits or address/zip code
- Sliding window to search through image
- Supervised learning classification to recognize digit, collect cluster of found digits
- K-means and posterior probability to report address/zip code or digits with bounding box

Supervised Learning - Classifiers



PCA dimension reduction

Bayesian Classifier

Mean vector
Covariance Matrix

Linear Classifier (Pseudo inverse)

Weights

Logistic Regression (as Classifier)

Weights

Decision Tree

Tree

Fully Connected Neural Network

Weights
Layers

MNIST dataset

- 784 dimensions (28*28 images)
- 10 class labels

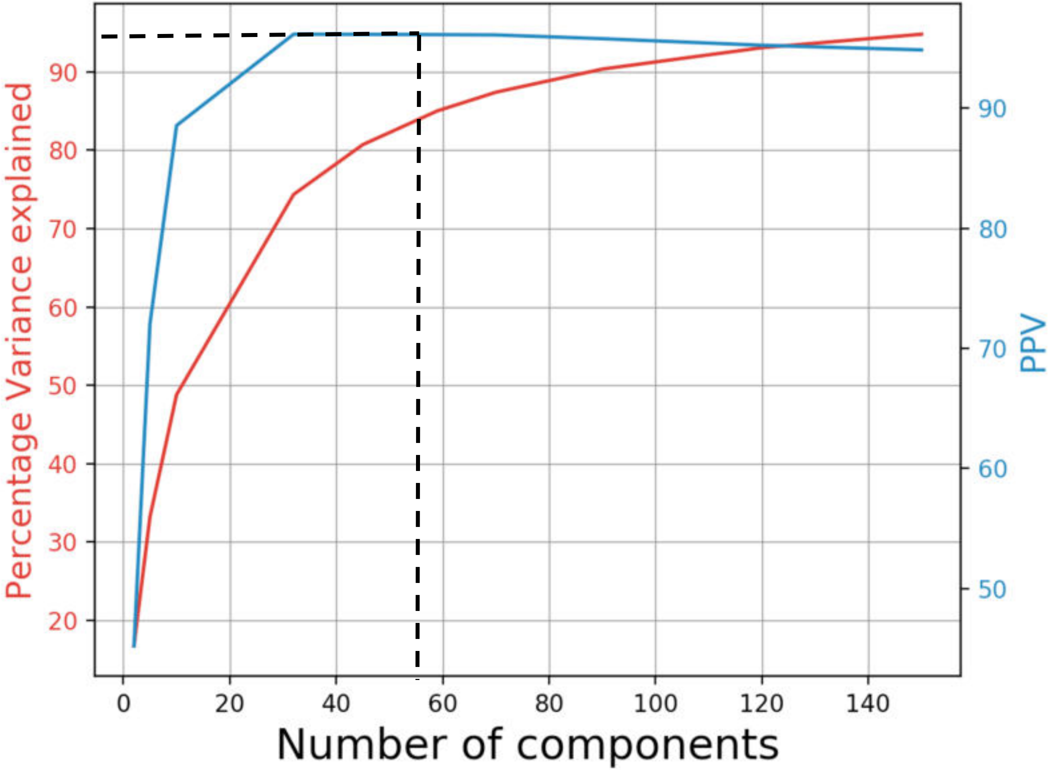
Training

Model Parameters

PCA Analysis

- PCA determination through percentage of variance explained

- Choice of optimal number of components based on PPV

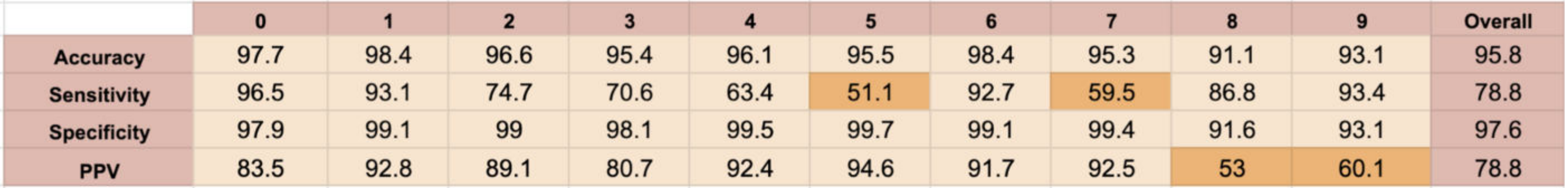


Astounding Insights!

- Accuracy is not reliable. We know this! High false positives get hidden by very high false negatives

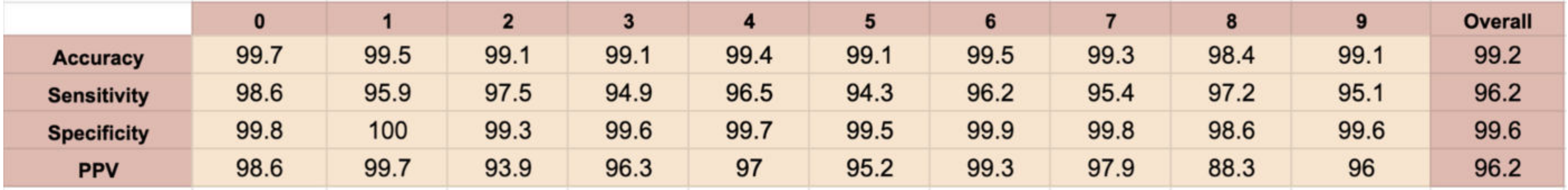
- The PPV with **Bayes classification after PCA** is significantly higher than PPV before PCA

Bayes on 784 dimension



	0	1	2	3	4	5	6	7	8	9	Overall
Accuracy	97.7	98.4	96.6	95.4	96.1	95.5	98.4	95.3	91.1	93.1	95.8
Sensitivity	96.5	93.1	74.7	70.6	63.4	51.1	92.7	59.5	86.8	93.4	78.8
Specificity	97.9	99.1	99	98.1	99.5	99.7	99.1	99.4	91.6	93.1	97.6
PPV	83.5	92.8	89.1	80.7	92.4	94.6	91.7	92.5	53	60.1	78.8

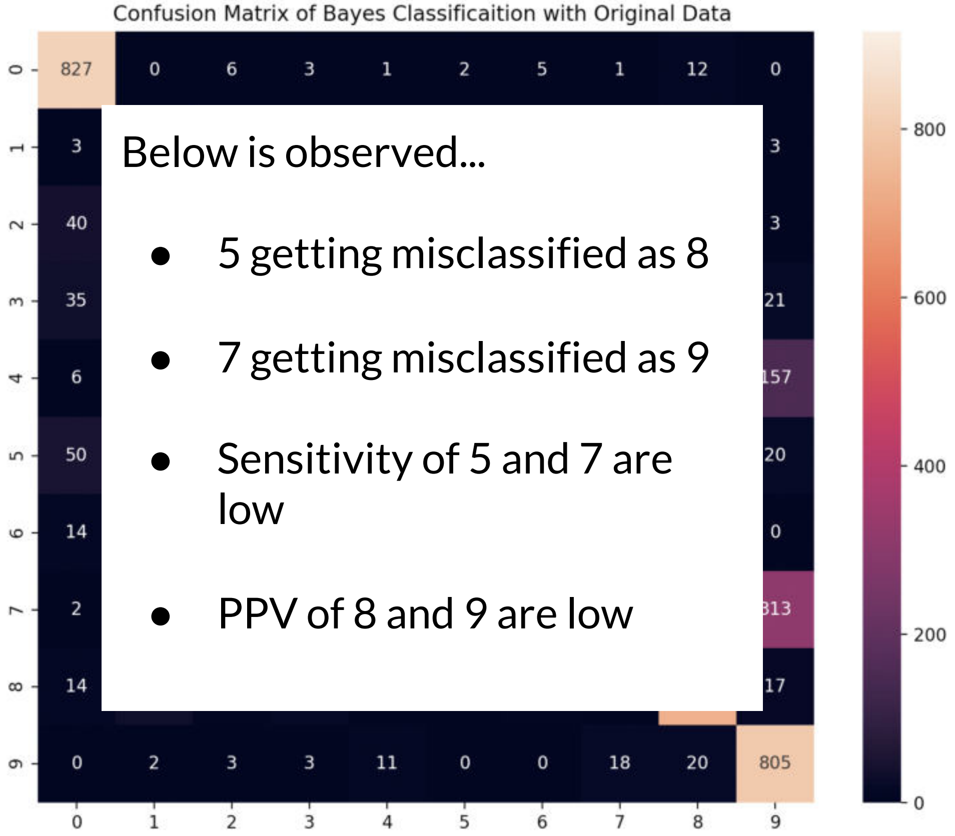
Bayes after PCA with 59 components



	0	1	2	3	4	5	6	7	8	9	Overall
Accuracy	99.7	99.5	99.1	99.1	99.4	99.1	99.5	99.3	98.4	99.1	99.2
Sensitivity	98.6	95.9	97.5	94.9	96.5	94.3	96.2	95.4	97.2	95.1	96.2
Specificity	99.8	100	99.3	99.6	99.7	99.5	99.9	99.8	98.6	99.6	99.6
PPV	98.6	99.7	93.9	96.3	97	95.2	99.3	97.9	88.3	96	96.2

Astounding Insights!

Bayes on 784 dimension

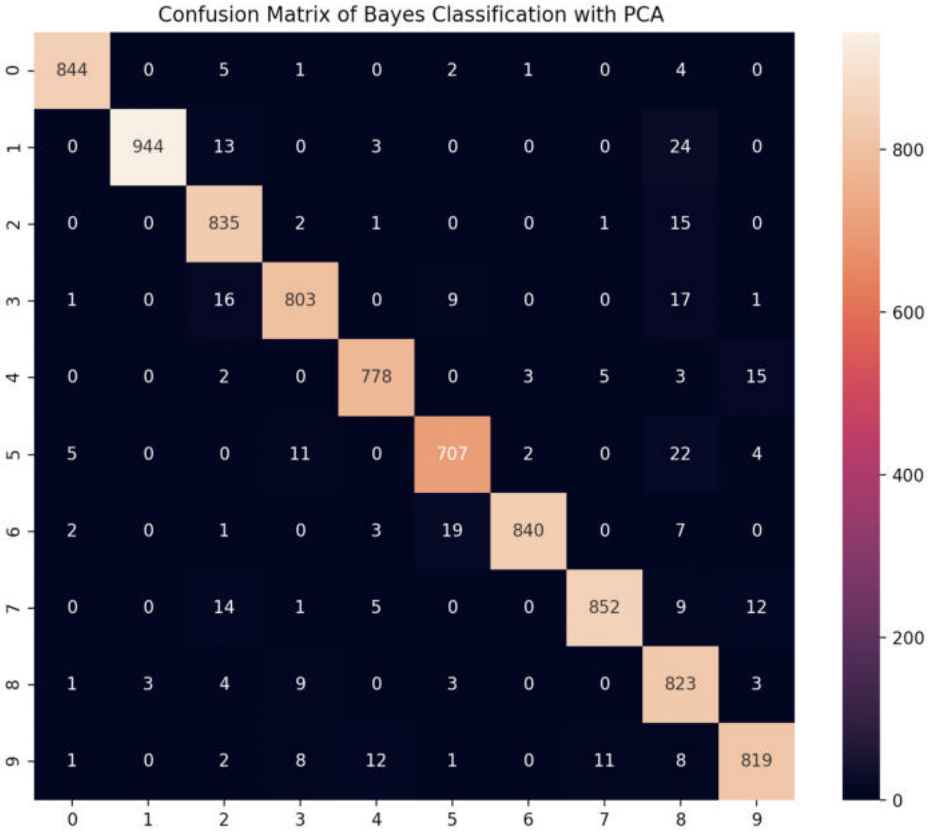


	0	1	2	3	4	5	6	7	8	9
0	827	0	6	3	1	2	5	1	12	0
1	3									3
2	40									3
3	35									21
4	6									157
5	50									20
6	14									0
7	2									313
8	14									17
9	0	2	3	3	11	0	0	18	20	805

Below is observed...

- 5 getting misclassified as 8
- 7 getting misclassified as 9
- Sensitivity of 5 and 7 are low
- PPV of 8 and 9 are low

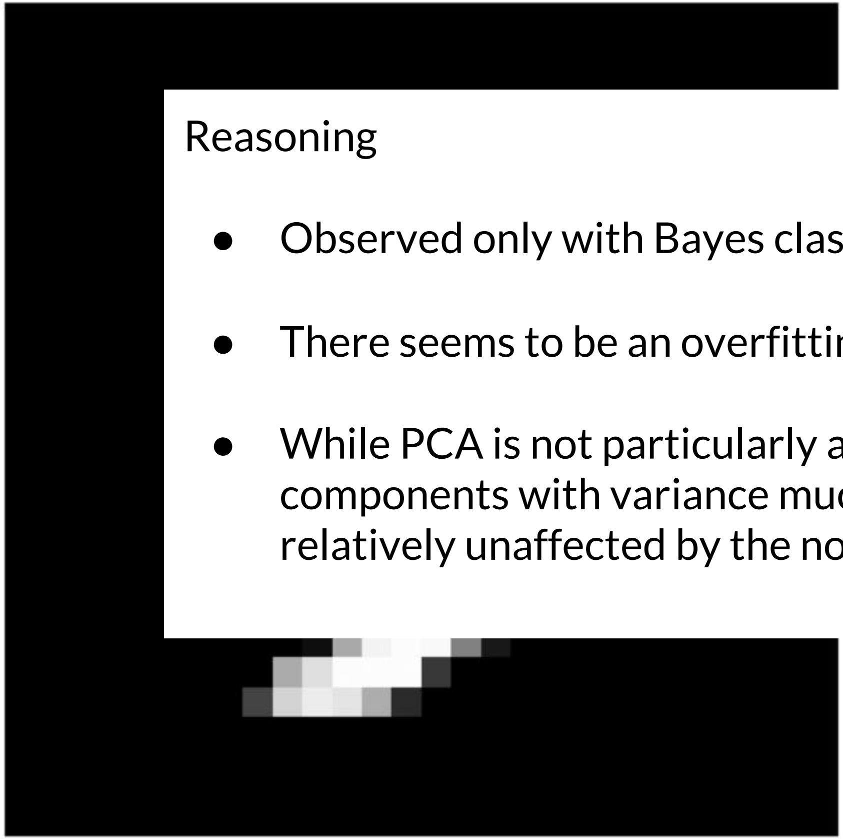
Bayes after PCA with 59 components



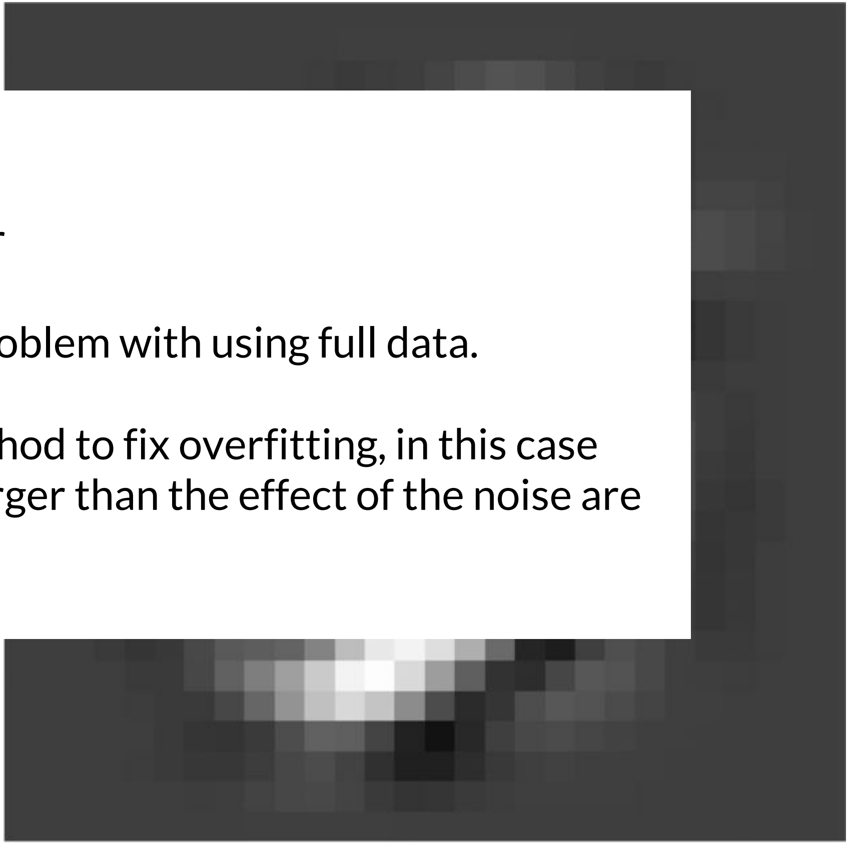
	0	1	2	3	4	5	6	7	8	9
0	844	0	5	1	0	2	1	0	4	0
1	0	944	13	0	3	0	0	0	24	0
2	0	0	835	2	1	0	0	1	15	0
3	1	0	16	803	0	9	0	0	17	1
4	0	0	2	0	778	0	3	5	3	15
5	5	0	0	11	0	707	2	0	22	4
6	2	0	1	0	3	19	840	0	7	0
7	0	0	14	1	5	0	0	852	9	12
8	1	3	4	9	0	3	0	0	823	3
9	1	0	2	8	12	1	0	11	8	819

Astounding Insights!

Bayes on 784 dimension
5 misclassified as 8



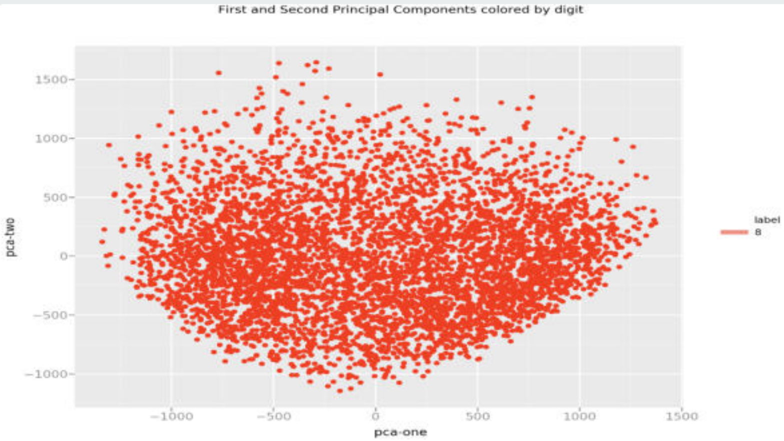
Bayes after PCA with 59 components
5 classified as 5



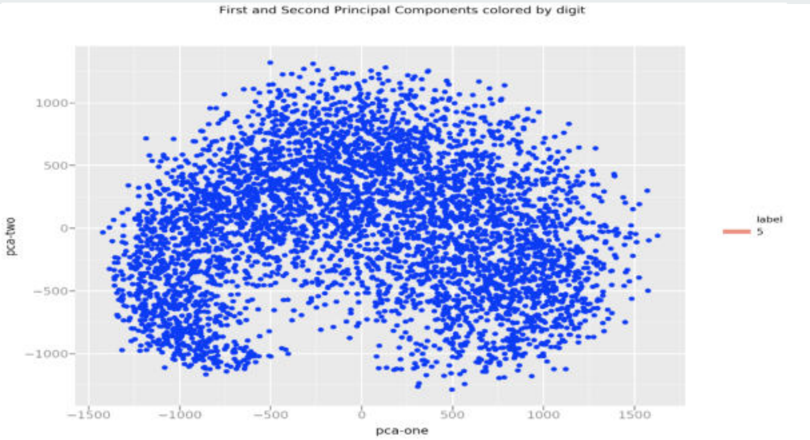
Reasoning

- Observed only with Bayes classifier
- There seems to be an overfitting problem with using full data.
- While PCA is not particularly a method to fix overfitting, in this case components with variance much larger than the effect of the noise are relatively unaffected by the noise

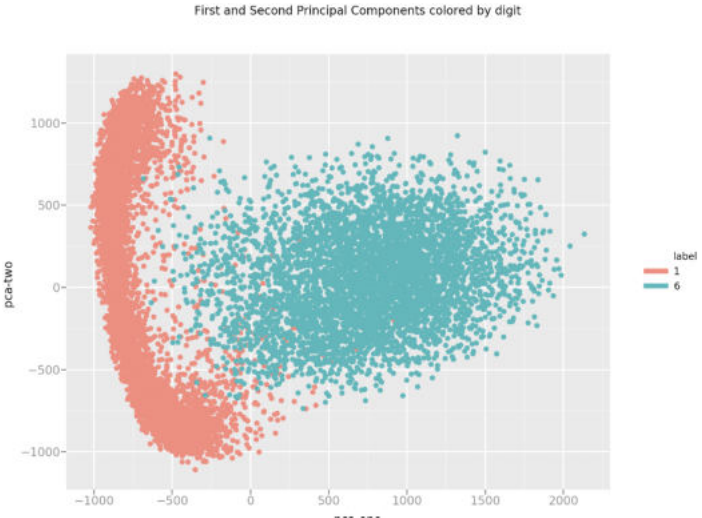
PCA Scatter plot - Some examples..



Digits
5 & 7



Digits 1 & 6



Interesting Facts

- Linear classifier (pseudo inverse) gave a PPV of 55% clearly indicating lack of linear separability

- Adding non linear quadratic dimension to the power 2 yielded a bump in PPV to 90%

	Linear Classifier (Pseudo Inverse)	Linear Classifier quadratic power 2 (Pseudo Inverse)
PPV	56%	87%
Dimensions	784	1830
Training time (ms)	75	900

- When tried for quadratic power 3, the training did not complete even after 15 minutes

Performance Metrics

	Accuracy	Sensitivity	Specificity	PPV
Full Bayesian	99.2	96.17	99.57	96.22
Logit	98.24	91.02	99.02	91.08
Neural Networks*	91.37	80.15	81.09	73.21
Decision Tree	96.75	83.42	98.2	83.48

** 784 dimensions, rest of the classifiers are run on the reduced dimensionality-PCA(59)*

Conclusion on classifiers

- By no means we have tried all possible tuning across all classifiers

- Goal was to bring out some interesting insights

- With the minimal combinations tried out, we feel Bayes classification on over PCA worked best for the given MNIST dataset for a 10 class classification.

- The reasoning thus far is that the distribution is highly gaussian in nature and PCA helped reduce noise to a certain extent leading to the highest PPV.

What we enjoyed ?

- The main goal from the beginning was to try something more deeper.

- We really enjoyed coming up with the technique for detecting multiple numbers and their bounding boxes.

- We thoroughly enjoyed going one step further with some of the concepts studied in class and applying / learning in practice.

- Definitely the outcome of Bayes classifier with PCA took us by surprise, main learning being when to use and not use them.