Machine Learning: Introduction to Linear Regression, Logistic Regression, and Neural Networks

Copyright Satish Reddy 2020

Chapter 7 Case Studies

Copyright Satish Reddy 2020

Case Studies

So far in this course:

- We have discussed underlying mathematics and algorithms for Linear Regression, Logistic Regression and Neural Network approaches for Supervised Learning
- We have developed a machine learning framework for Supervised Learning that can apply these approaches
- We have discussed tools for improving optimization for the training algorithm, measuring performance, regularization, and addressing Underfitting and Overfitting
- In this chapter, we use the framework to address three case studies

Copyright Satish Reddy 2020

Case Studies

Section	Case Study	Type	Description
7.1	House Price Prediction	Regression	This case study uses Linear Regression to predict house prices. It also shows how to use “exploratory data analysis” and “feature engineering” to prepare the data.
7.2	Spam Classification	Binary Classification	This case study uses a Neural Network to create a spam filter. It shows an approach for converting text into a feature matrix.
7.3	MNIST Digits Classification	Multiclass Classification	This case study uses a Neural Network to identify digits from images. It shows how to convert an image into a feature matrix.

Copyright Satish Reddy 2020

7.1. Case Study: House Price Prediction

Copyright Satish Reddy 2020

Case Study: House Price Prediction

Goal of this Section:

- Introduce Exploratory Data Analysis and Feature Engineering to set up the machine learning problem
- Use Linear Regression for House Price Prediction

Copyright Satish Reddy 2020

House Price Dataset - Citation

- House Price Dataset compiled from data in Sindian Dist., New Taipei City, Taiwan
- Source: (University of California, Irvine, Machine Learning Repository)

https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set

- Paper describing work:

Yeh, I. C., & Hsu, T. K. (2018). Building real estate valuation models with comparative approach through case-based reasoning. Applied Soft Computing, 65, 260-271.

Copyright Satish Reddy 2020

House Price Dataset

- Data located in folder IntroML/Code/Data_House:
  - 414 data samples
  - 1 Value (price-per-unit-area),
  - 3 features (house-age (years), dist-to-nearest-MRT (meters), num-of-stores)
- For this problem, have not included transaction date and coordinates of house

	A	B	C	D	E
1	price-per-unit-area	house-age	dist-to-nearest-MRT	num-of-stores	
2	37.9	32	84.87882	10	
3	42.2	19.5	306.5947	9	
4	47.3	13.3	561.9845	5	
5	54.8	13.3	561.9845	5	
6	43.1	5	390.5684	5	
7	32.1	7.1	2175.03	3	
8	40.3	34.5	623.4731	7	
9	46.7	20.3	287.6025	6	
10	18.8	31.7	5512.038	1	
11	22.1	17.9	1783.18	3	
12	41.4	34.8	405.2134	1	
13	58.1	6.3	90.45606	9	
14	39.3	13	492.2313	5	
15	23.8	20.4	2469.645	4	
16	34.3	13.2	1164.838	4	

Copyright Satish Reddy 2020

Exploratory Data Analysis & Feature Engineering

Exploratory Data Analysis

- Approach for analyzing data sets to understand relationships between variables often using visual tools
- For House Price Prediction, do not expect house prices to be linearly dependent on features
- Use Exploratory Data Analysis to determine relationship between prices and variables to determine transformations to be made

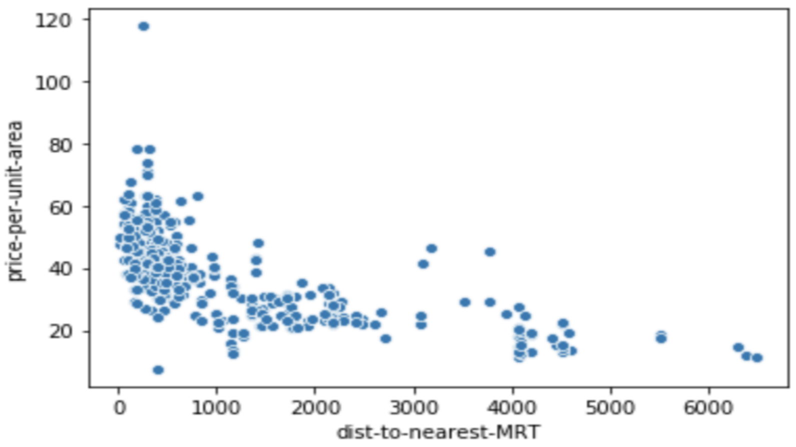
Feature Engineering

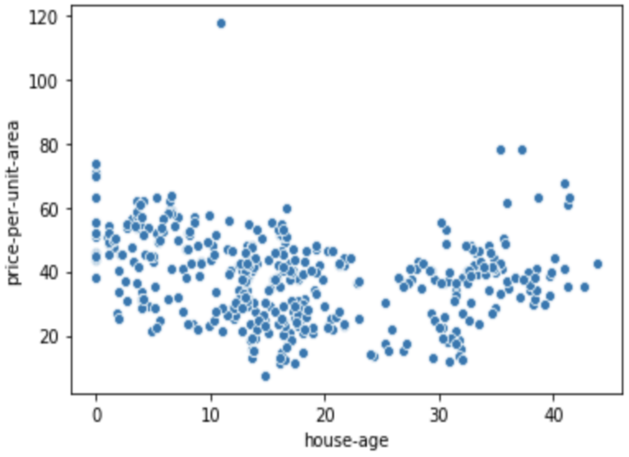
- Apply transformations to feature variables suggested by exploratory data analysis
- “Standardize” or “normalize” variables so they are all of roughly the same size
- Remove outliers

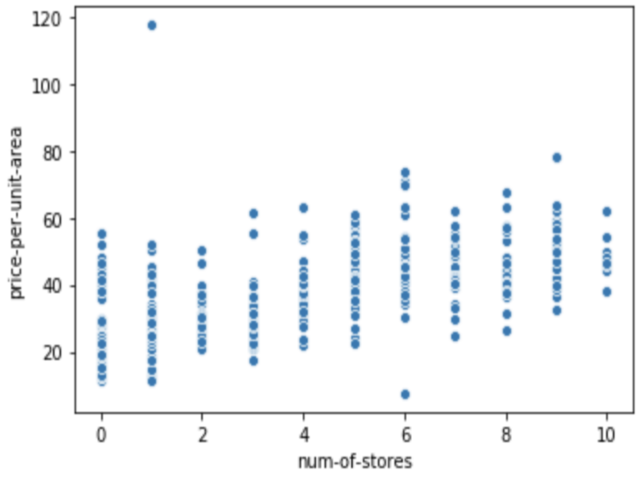
Copyright Satish Reddy 2020

Exploratory Data Analysis

- Scatter plots show relationships between price-per-unit area and each of the features
- Scatter plots suggest non-linear relationships between price-per-unit-area and distance to nearest MRT and house age



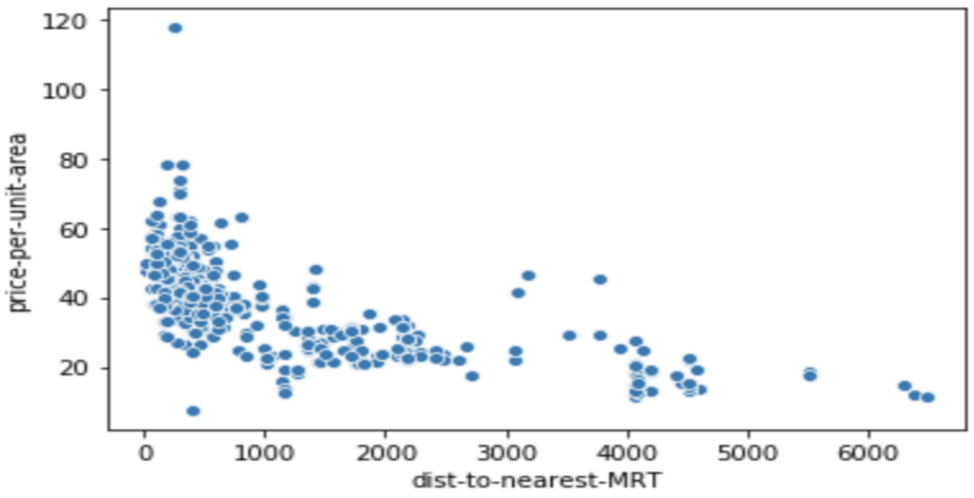


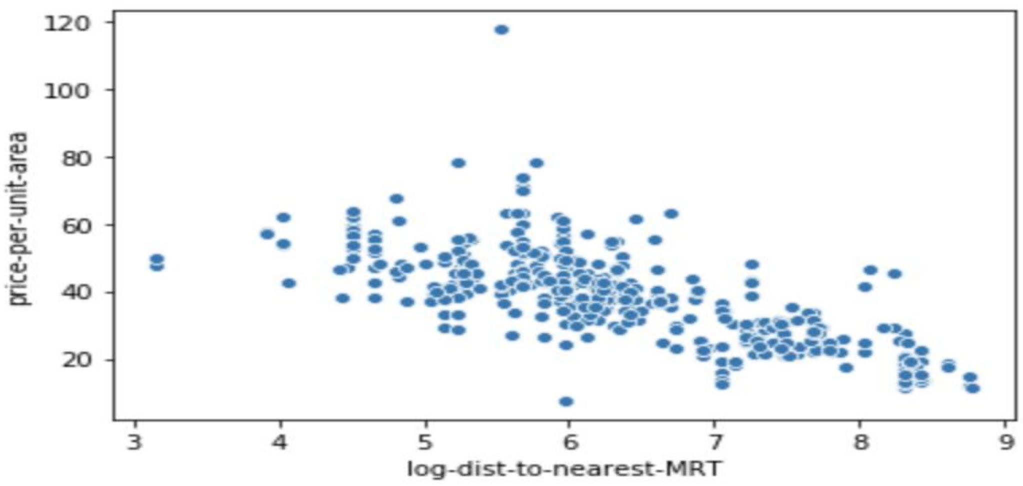


Copyright Satish Reddy 2020

Variable Transformations

- A log transformation is applied to the variable **dist-to-nearest-MRT** (explanatory)
  - Improves linearity between the explanatory variable and price

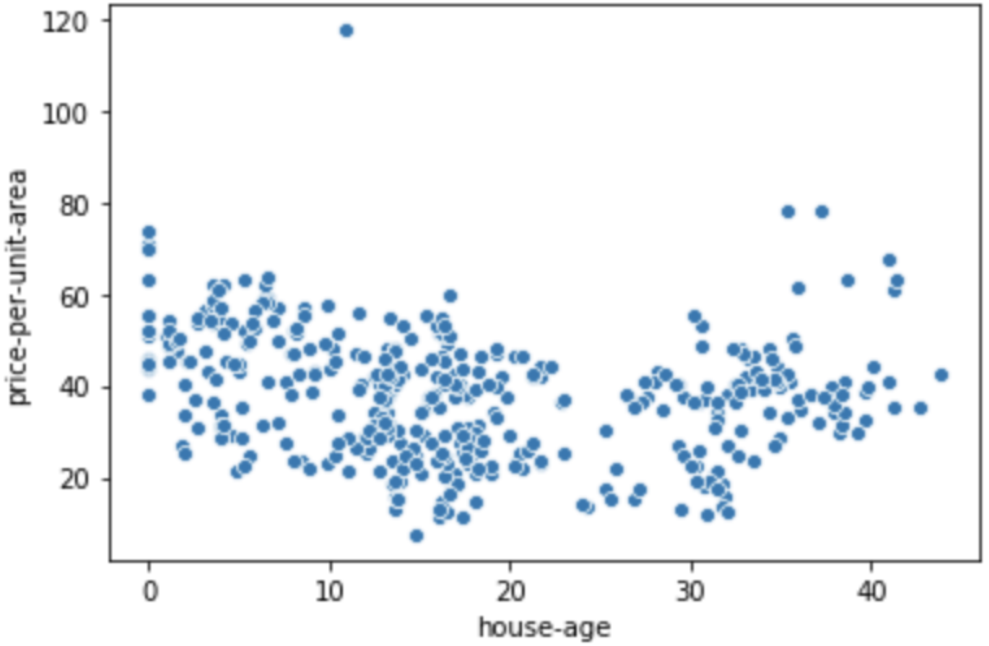


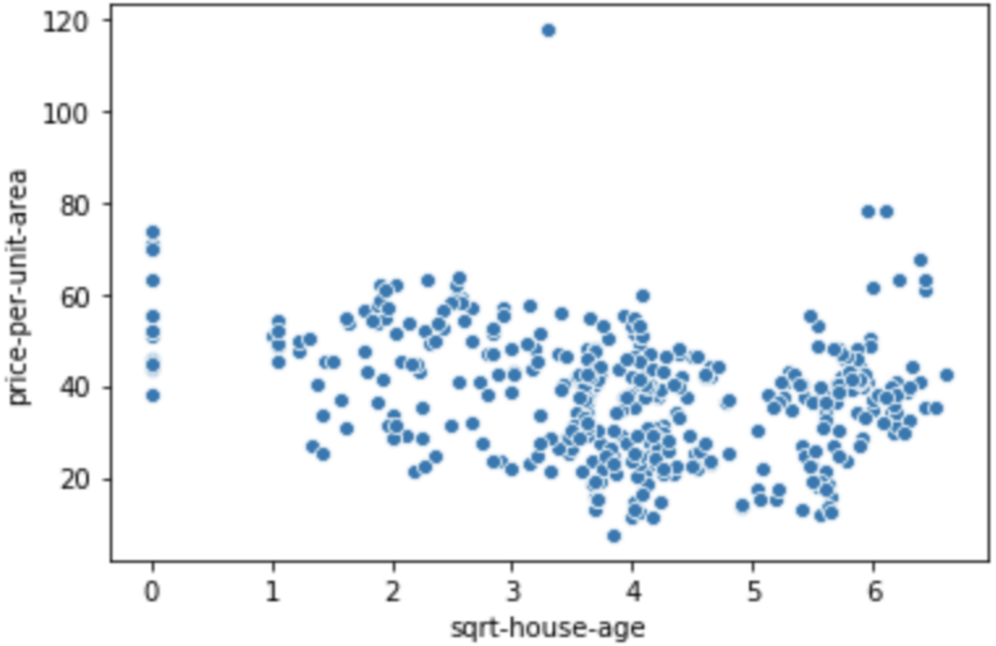


Copyright Satish Reddy 2020

Variable Transformations

- A square root transformation is applied to the variable **house-age** (explanatory)
  - Improves linear relationship between the explanatory variable and the price





Copyright Satish Reddy 2020

Feature Standardization

- Feature values are of different magnitudes:
  - Age: roughly 0 – 50
  - Distance to MRT: roughly 0 – 7000
  - Number of stores: 0 - 10
- Standardize Features:
  - Subtract mean and scale by standard deviation of training data
  - Apply same scaling to validation data
  - Results in feature data having mean 0 and standard deviation 1 so that feature values are the same magnitude
- This can improve convergence of optimizer

Copyright Satish Reddy 2020

Value Standardization

- Can apply same technique as previous page to standardize price-per-unit-area data
- For simplicity, will simply divide price-per-unit-area data by maximum value in training data set
- Apply exact same scaling factor to validation data set
- Note that predictions of the model will need to be multiplied by this factor to get correct price-per-unit-area

Copyright Satish Reddy 2020

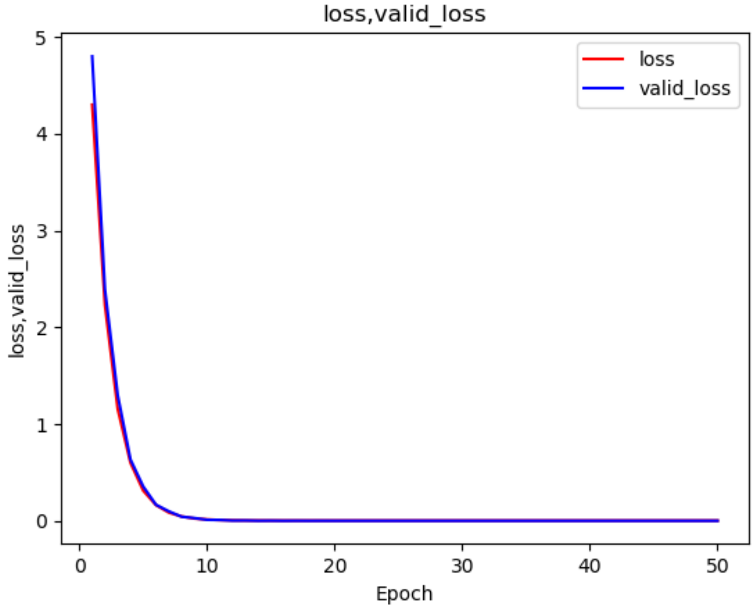
House Price Prediction using Linear Regression

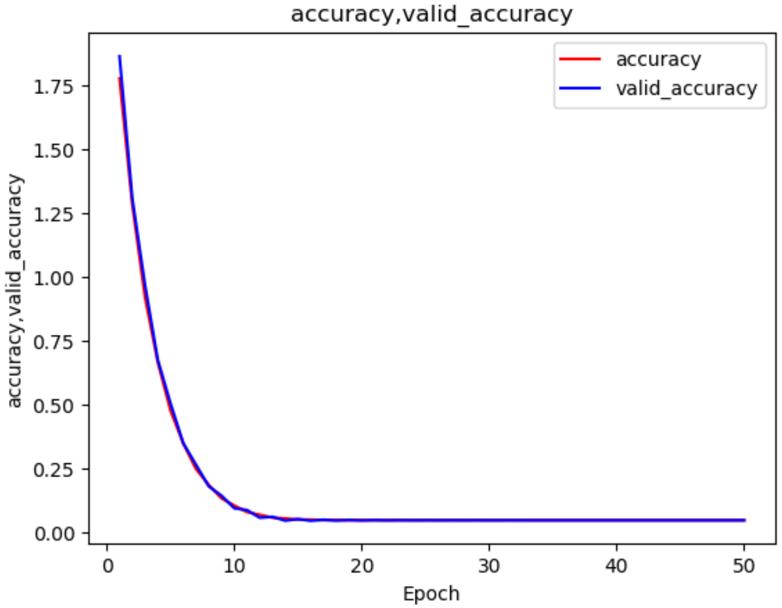
- Example: Dataset
  - 331 samples in training dataset (roughly 80% of samples)
  - 83 samples in validation dataset (roughly 20% of samples)
  - Feature matrix for training (3 x 331)
  - Feature matrix for validation (3 x 83)
- Linear Regression
  - W is 1x3 matrix and b is 1x1 scalar
  - λ = 0.0001 (regularization)
- Optimization:
  - Gradient Descent (α = 0.5)
  - 50 epochs batch

Copyright Satish Reddy 2020

House Price Prediction – Summary of Results

- Run with Transformation, Standardization for X and Y = True
  - Training Loss: 0.0059, Training Accuracy: 0.050
  - Validation Loss: 0.0041, Validation Accuracy: 0.050
  - Loss and Accuracy plots indicate no concerning signs of overfitting





Copyright Satish Reddy 2020

Summary of Results

Run	Transformations/Standardization	Results after 50 epochs
1	Transformation: True Standardization X: True Standardization Y: True	Training Accuracy: 0.050 Validation Accuracy: 0.050
2	Transformation: False Standardization X: True Standardization Y: True	Training Accuracy: 0.056 Validation Accuracy: 0.052
3	Transformation: True Standardization X: False Standardization Y: True	Training Accuracy: Blow up – does not converge Validation Accuracy: Blow up - does not converge

Copyright Satish Reddy 2020

Further Investigations

- Data:
  - Only 3 of 6 features in original dataset are used in this section - investigate if results are improved if additional features are used
- Transformations
  - Investigate other transformations
- Optimization
  - Investigate use of different learning rate for Gradient Descent or use of other optimizers to overcome loss blow up

Copyright Satish Reddy 2020

New Code for House Price Prediction

Function/component	Input	Description
load_house	train_pct (float) transform (boolean) standardizeX (boolean) standardizeY (boolean)	Loads house price data, performs feature engineering/preprocessing. Train_pct ranges from 0.0-1.0 and indicates the proportion of the data to used for training. Inputs transform, standardizeX, and standardizeY, determine what manipulations are made to the data Return: Xtrain, Ytrain, Xvalid, Yvalid
driver_casestudy_ house		Driver for house price prediction

Copyright Satish Reddy 2020

House Price Prediction Walkthrough

- Code for walkthrough located at:

IntroML/Code/Version4.1

- Demo of pandas for loading and processing data:

IntroML/Examples/Chapter2/PandasDemos.ipynb

- You can implement the code additions suggested on the previous page by adding to a clean Version3.3 of the code
  - Have a look at Version4.1 for hints
- We will perform a walkthrough of the code additions for the spam classification

Copyright Satish Reddy 2020

7.2 Case Study: Spam Classification

Copyright Satish Reddy 2020

Case Study: Spam Classification

Goal of this Section:

- Describe approach for using neural networks for spam classification using the SMS dataset

Copyright Satish Reddy 2020

Text Classification

Text Classification is an application of supervised machine learning
Examples include:

- Spam Classification
  - Binary Classification
  - Training Data: messages and labels (spam or not spam)
  - Goal: predict if new message is spam or not (use to filter email, for example)
- Sentiment Classification of Reviews
  - Multiclass classification
  - Training Data: reviews and labels (1, 2, 3, 4 or 5 star, for example)
  - Goal: predict rating for new reviews

Copyright Satish Reddy 2020

SMS Spam Collection Dataset - Citation

- Source: (University of California, Irvine, Machine Learning Repository)

http://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection

- Paper describing work:

Almeida, T.A., Gómez Hidalgo, J.M., Yamakami, A. Contributions to the study of SMS Spam Filtering: New Collection and Results. Proceedings of the 2011 ACM Symposium on Document Engineering (ACM DOCENG'11), Mountain View, CA, USA, 2011. (Under review)

- See website: http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/

Copyright Satish Reddy 2020

SMS Spam Collection Dataset

- Data located in folder IntroML/Code/Data_Spam
  - readme.txt file provides details about dataset
  - SMSSpamCollection.csv contains the data
- Consist of 5574 text messages: 4827 (not spam) 747 (spam)
- Each line has label (ham or spam) in col A and message in col B
- Assign labels: ham = 0 and spam = 1

	A	B	C	D	E	F	G	H	I	J	K
1	label	message									
2	ham	Go until jurong point									
3	ham	Ok lar... Joking wif u oni...									
4	spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry ques									
5	ham	U dun say so early hor... U c already then say...									
6	ham	Nah I don't think he goes to usf									
7	spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still?									
8	ham	Even my brother is not like to speak with me. They treat me like aids patent.									
9	ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertu									
10	spam	WINNER!! As a valued network customer you have been selected to receivea Â£900 prize reward! To clai									
11	spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for									
12	ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight									
13	spam	SIX chance	6days	16+ TsandCs apply Reply HL 4 info							
14	spam	URGENT! You have won a 1 week FREE membership in our Â£100									
15	ham	I've been searching for the right words to thank you for this breather. I promise i wont take your help for									
16	ham	I HAVE A DATE ON SUNDAY WITH WILL!!									
17	spam	XXXMobileMovieClub: To use your credit									
18	ham	Oh k...i'm watching here:)									

Copyright Satish Reddy 2020

Converting Messages into a Feature Matrix

- Rudimentary Approach: CountVectorizer (from sklearn package)

1. Build a vocabulary consisting of all words in all messages
   - Not case sensitive ("My" and "my" and "MY" are same word)
2. Feature matrix entry X_{ij} is number of times word i appears in message j
3. Feature Matrix has dimensions (# of words x # of messages)

- Can adjust settings to not include some words (called "stop words"), such as "the", "to", "and", …., which probably do not impact classification.
- See https://scikit-learn.org/stable/index.html for details

Copyright Satish Reddy 2020

CountVectorizer - Example

- 3 Messages: “Call me soon”, “CALL to win”, “Pick me up soon”
- 7 unique words
- Feature matrix is (7 words) x (3 messages)

Vocabulary: Feature Matrix:

$$\begin{matrix} call \\ me \\ pick \\ soon \\ to \\ up \\ win \end{matrix} \qquad \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Copyright Satish Reddy 2020

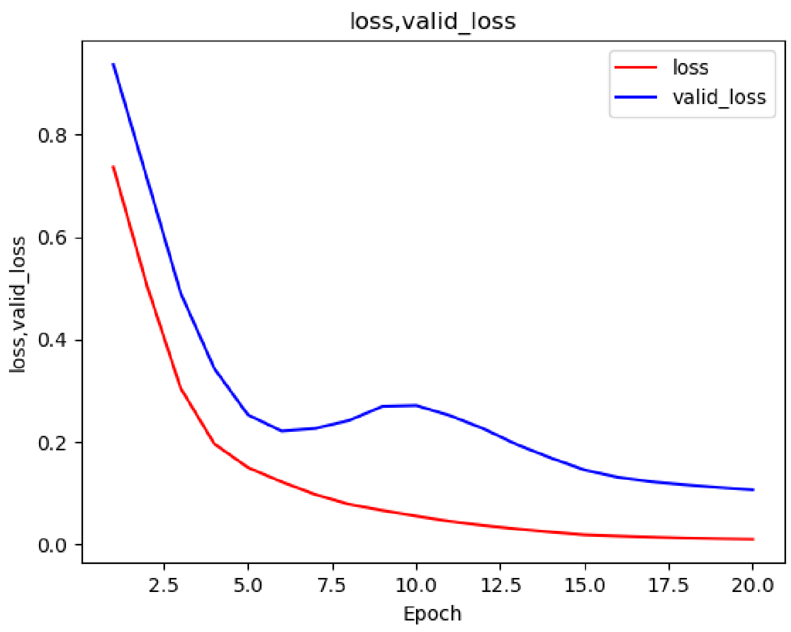
Spam Classification using a Neural Network

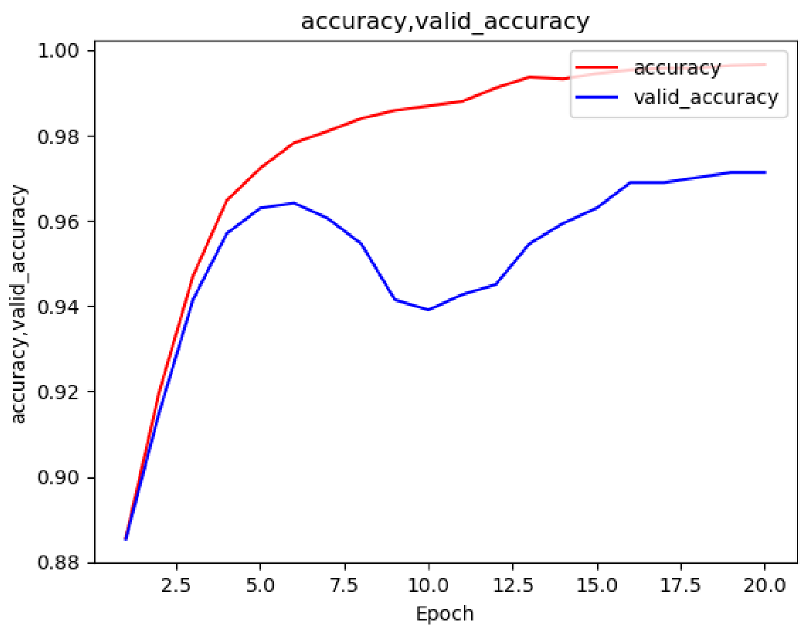
- Example: Dataset
  - 4737 (roughly 85%) messages in training dataset
  - 837 (roughly 15%) messages in validation dataset
  - 7523 words in vocabulary
  - Feature matrix for training is (7523 x 4737)
  - Feature matrix for validation is (7523 x 837)
- Neural Network
  - 3 layer neural network
    - Layer 1: 200 units (tanh activation)
    - Layer 2: 50 units (tanh activation)
    - Layer 3: 1 unit (sigmoid activation)
    - Binary Cross Entropy Loss function
    - 1,514,901 total entries in $W^{[k]}$ and $b^{[k]}$ for k=1,2,3
- Optimization:
  - Adam
    - $\alpha = 0.02$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-7}$
  - 20 epochs (batch_size = 4737 - batch gradient descent)

Copyright Satish Reddy 2020

Spam Classification – Summary of Results

- After 20 epochs:
  - Training Accuracy: 0.997
  - Validation Accuracy: 0.971
  - Precision: 0.864, Recall = 0.927, F1 score = 0.895
- Loss and Accuracy plots indicate an overfitting





Copyright Satish Reddy 2020

Spam Classification – Summary of Results

- Confusion matrix for validation dataset model yields
  - 16 False Positive Messages
  - 8 False Negative Messages

Anaconda Prompt - python driver_casestudy_spam.py

```
                Confusion Matrix
                Actual
                0       1
Predicted  0    711     8
           1    16      102
F1Score: 0.8947368417128347 - Precision: 0.864406779294743 - Recall: 0.9272727268512397
False Positive messages - Actual = not spam - Predicted = spam
Hi.:)technical support.providing assistance to us customer through call and email:)
Thanks  and ! Or bomb and date as my phone wanted to say!
Nokia phone is lovly..
I'm vivek:)i got call from your number.
My mobile number.pls sms ur mail id.convey regards to achan
Yeah so basically any time next week you can get away from your mom &amp; get up before 3
Sorry i missed your call. Can you please call back.
Happy new year to u and ur family...may this new year bring happiness
K k:) sms chat with me.
Hi. Hope ur day * good! Back from walk
Mode men or have you left.
Dhoni have luck to win some big title.so we will win:)
Have you seen who's back at Holby?!
Also fuck you and your family for going to rhode island or wherever the fuck and leaving me all alone the week I have a new bong &gt;:(
Compliments to you. Was away from the system. How your side.
Armand says get your ass over to epsilon
Goal! Arsenal 4 (Henry
Hi
Hi this is Amy
Welcome to Select
This is the 2nd attempt to contract U
If you don't
dating:i have had two of these. Only started after i sent a text to talk sport radio last week. Any connection do you think or coincidence?
Latest News! Police station toilet stolen
```

Further Investigations

- Feature Matrix:
  - Investigate other approaches to create Feature Matrix
    - For example, TdIdfVectorizer (see scikit-learn documentation for more details)
- Overfitting
  - Go through procedures described in previous chapter to address overfitting in this problem
    - More data – this is not feasible
    - Simpler neural network
    - Regularization
    - Perform hyperparameter tuning to find best performance

Copyright Satish Reddy 2020

New Code for Spam Classification

Method	Input	Description
load_spam	train_pct (float)	Loads spam data base and returns train and validation feature matrices (based on CountVectorizer) and label vectors. Also returns original messages in train and validation datasets. Return: Xtrain, Ytrain, Xvalid, Yvalid, Xtrain_raw, Xvalid_raw
data_analysis	X (numpy array) Y (numpy array) nmostcommon (integer) vectorizer (CountVectorizer instance)	Takes in X feature matrix generated by CountVectorizer and label vector Y and prints nmostcommon words in spam and not spam messages Return: nothing
text_results	Y (numpy array) Y_pred (numpy array) X_raw (numpy array)	Given actual Y and predicted Y_pred label vectors and raw messages, this function prints the false positive and false negative messages Return: nothing
driver_casestudy_ spam		Driver for spam classification using a neural network

Copyright Satish Reddy 2020

Spam Classification Walkthrough

- Code for walkthrough located at:

IntroML/Code/Version4.1

- Demo of pandas for loading and processing data in

IntroML/Examples/Chapter2/PandasDemo.ipynb

- Demo of CountVectorizer for text processing in

IntroML/Examples/Chapter2/sklearnDemo.ipynb

- You can implement the code additions suggested on the previous page by adding to a clean Version3.3 of the code
    - Have a look at Version4.1 for hints
- We will perform a walkthrough of the code additions for the spam classification

Copyright Satish Reddy 2020

7.3 Case Study: MNIST Digits Classification

Copyright Satish Reddy 2020

MNIST Digits Classification

Goal of this Section:

- Describe approach for using neural networks for image classification using the MNIST Digits dataset

Copyright Satish Reddy 2020

Machine Learning - Image Classification

- Image classification (cats and dogs, animals, x-rays, scans, etc) is a principal application of supervised machine learning
- Binary or multi-class
- Training data consists of images plus labels
- Goal is to be able to predict label for new images
- Question: how does one convert images into feature matrix to employ neural network approach?

Copyright Satish Reddy 2020

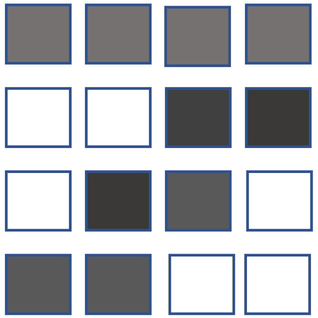
Representation of Images

- Images typically are composed of rectangular arrays of pixels
- For black and white images, intensity of greyscale for each pixel is represented by a number (white = 0 to 255 = black)
- Feature vector for image is vector of intensities for all pixels
- For colour images, each pixel represented by 3 values – intensities of red, blue, and green components for that pixel – feature vector will have 3 x number of pixels as in black and white case

Copyright Satish Reddy 2020

Converting Image to Feature Matrix

Original Image:
Greyscale 4x4 =16 pixels



Intensity Matrix
4x4 (white=0 to 255=black)

$$\begin{bmatrix} 190 & 190 & 190 & 190 \\ 0 & 0 & 220 & 220 \\ 0 & 220 & 200 & 0 \\ 200 & 200 & 0 & 0 \end{bmatrix}$$

Feature Vector 16x1

$$\begin{bmatrix} 190 \\ 190 \\ 190 \\ 190 \\ 0 \\ 0 \\ 220 \\ 220 \\ 0 \\ 220 \\ 200 \\ 0 \\ 200 \\ 200 \\ 0 \\ 0 \end{bmatrix}$$

Copyright Satish Reddy 2020

Choice of Labels

- For Binary Classification, arbitrarily assign 0, 1 to the classes
  - Example: for classification of cats and dogs (assign 0 for cat and 1 for dog)
  - Example: X-rays assign (0 normal and 1 for broken)
  - Choice is arbitrary (can use 1 for cat and 0 for dog) – doesn’t matter
- For Multiclass Classification (c classes) assign 0,1,…,c-1 to classes
  - For digits classification, 10 classes – obviously assign 0 to 0, 1 to 1, …, 9 to 9
  - For pictures of cats, dogs, rabbits, ferrets, ducks (5 classes), assign 0 to cats, 1 to dogs, 2 to rabbits, 3 to ferrets, and 4 to ducks.

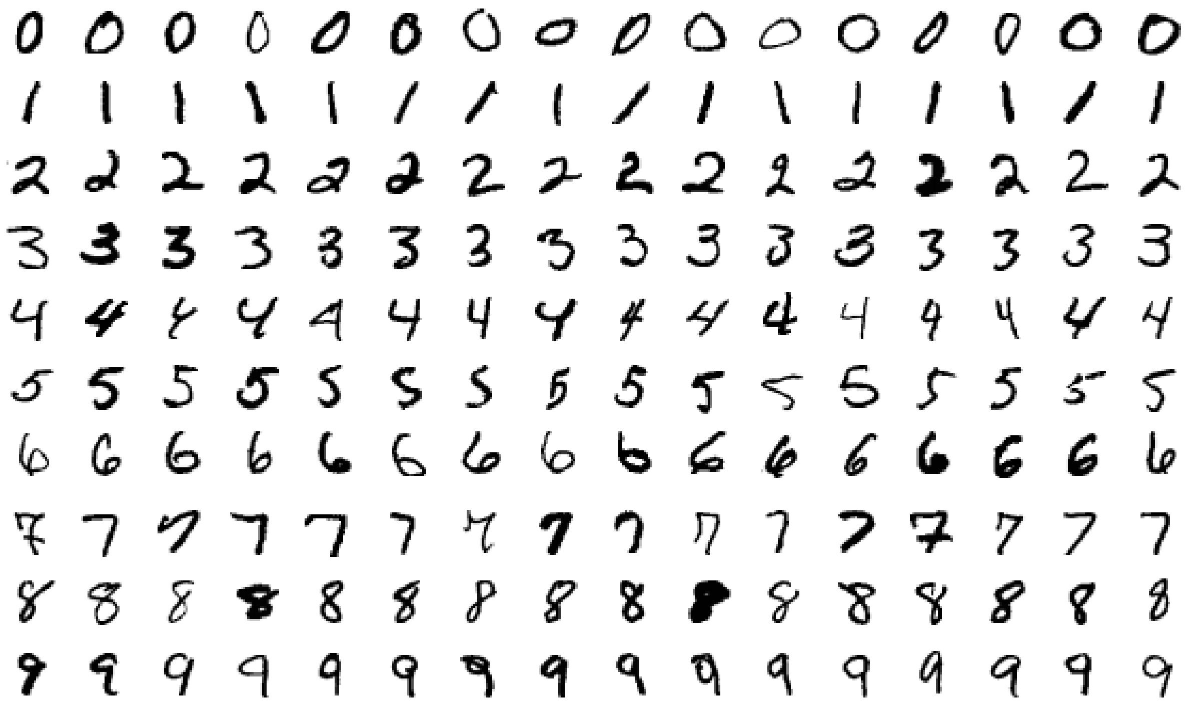
Copyright Satish Reddy 2020

MNIST Digits Database

- NIST is acronym for National Institute of Standards and Technology, which is a physical sciences laboratory and a non-regulatory agency of the United States Department of Commerce
- MNIST (Modified National Institute of Standards and Technology) digits database is a large collection of black and white handwritten digit images used for training and testing of machine learning algorithms
- Digit images are uniform (28x28 resolution = 784 pixels)
- 60,000 individual digit images (0 – 9 with labels) for training
- 10,000 individual digit images (0 – 9 with labels) for testing
- Data Source: http://yann.lecun.com/exdb/mnist/

Copyright Satish Reddy 2020

Sample of Digit Images

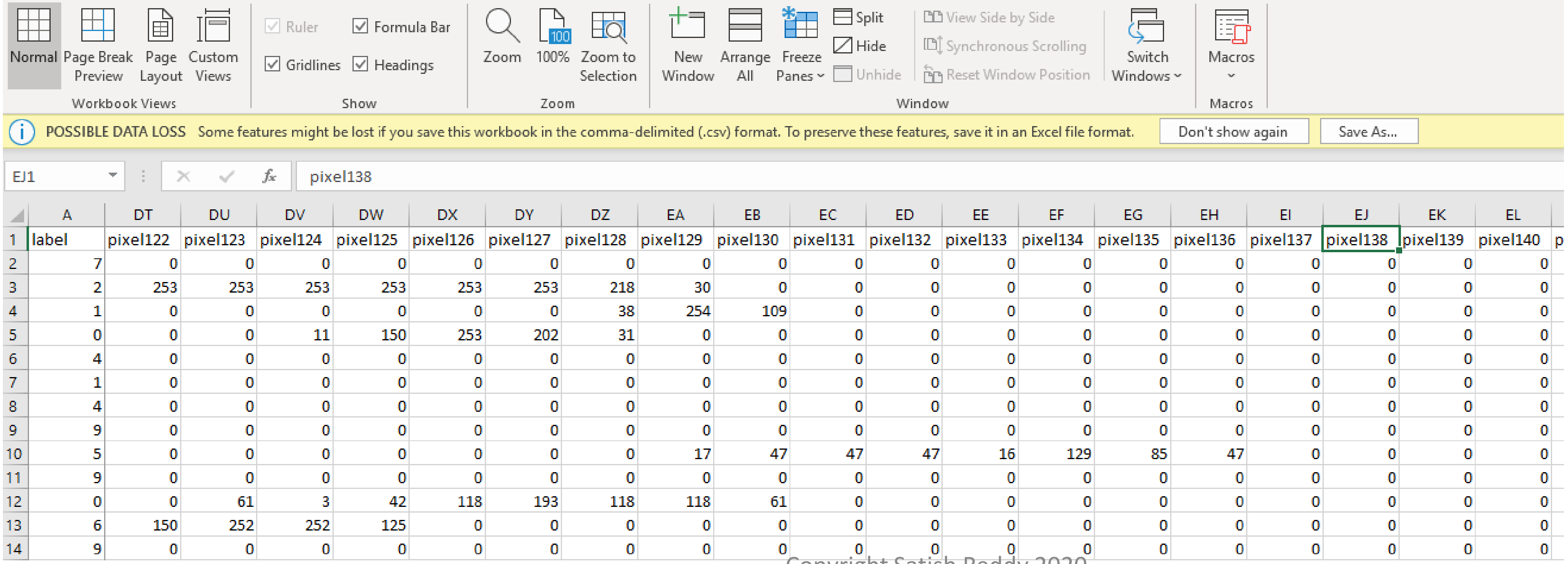


- Collage of 160 individual digit images
- Citation for above image

By Josef Steppan - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=64810040

MNIST Digits – Format of Data Files

- Each row represents label and intensities for one image
  - First column is the digit label (0,1,…,9)
  - Columns 2 – 785 are the intensities
  - Take transpose to convert feature matrix and value vector to correct format
  - Standard practice is to divide pixel values by 255 so between 0 and 1



	A	DT	DU	DV	DW	DX	DY	DZ	EA	EB	EC	ED	EE	EF	EG	EH	EI	EJ	EK	EL
1	label	pixel122	pixel123	pixel124	pixel125	pixel126	pixel127	pixel128	pixel129	pixel130	pixel131	pixel132	pixel133	pixel134	pixel135	pixel136	pixel137	pixel138	pixel139	pixel140
2	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	2	253	253	253	253	253	253	218	30	0	0	0	0	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	38	254	109	0	0	0	0	0	0	0	0	0	0
5	0	0	0	11	150	253	202	31	0	0	0	0	0	0	0	0	0	0	0	0
6	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0	17	47	47	47	16	129	85	47	0	0	0	0
11	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	61	3	42	118	193	118	118	61	0	0	0	0	0	0	0	0	0	0
13	6	150	252	252	125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Copyright Satish Reddy 2020

Training and Validation Data

- Data located in folder IntroML/Code/Data_MNIST:
  - 60000 training data samples split into 2 files (because of Github limitations)
    - MNIST_train_set1_30K.csv
    - MNIST_train_set2_30K.csv
    - 1 data sample for each row consisting of digit label plus 784=28x28 pixel values
  - 10000 validation data samples in file:
    - MNIST_valid_10K.csv
    - 1 data sample for each row consisting of digit label plus 784=28x28 pixel values

Copyright Satish Reddy 2020

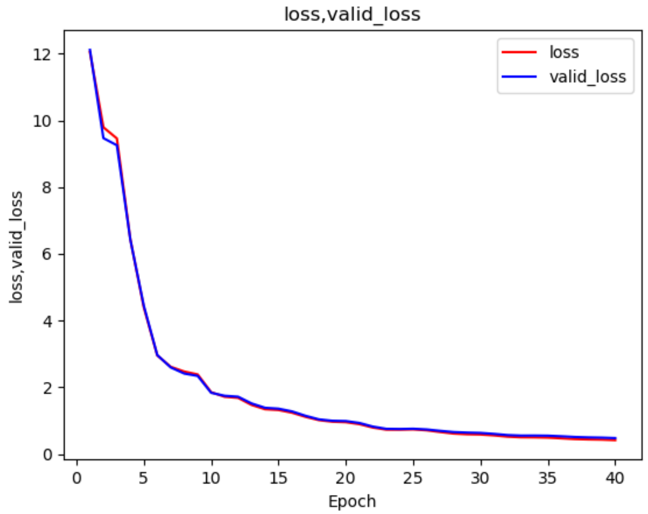
MNIST Digit Classification using a Neural Network

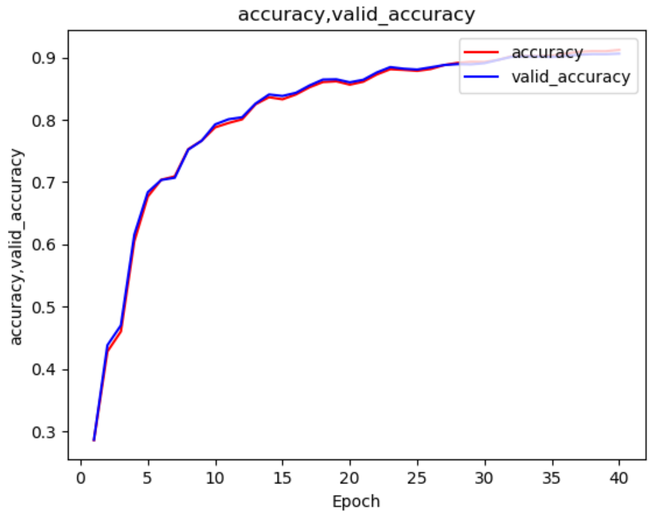
- Example: Dataset
  - 60000 images (28x28 resolution) in training dataset
  - 10000 images in validation dataset
  - Feature matrix for training is (784 x 60000)
  - Feature matrix for validation is (784 x 10000)
- Neural Network
  - 2 layer neural network
    - Layer 1: 128 units (tanh activation)
    - Layer 2: 10 unit (softmax activation)
    - Cross Entropy Loss function
    - 101,770 total entries in $W^{[k]}$ and $b^{[k]}$ for k=1,2
- Optimization:
  - Adam
    - α=0.02, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-7}$
    - 40 epochs (batch_size = 60000 batch gradient descent)

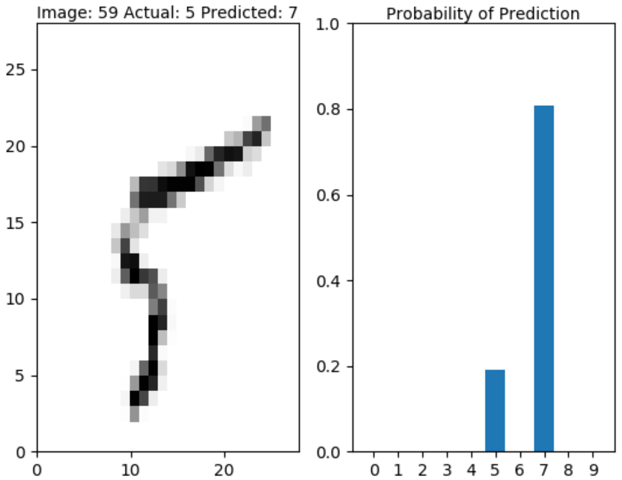
Copyright Satish Reddy 2020

Digit Classification – Summary of Results

- After 40 epochs:
  - Training Accuracy: 0.912
  - Validation Accuracy: 0.906
- Loss and Accuracy plots indicate an underfitting
  - Expect training accuracy to be higher – should be close to 100%
- Plot of Image and Probability:
  - Probability bar chart obtained from final activation for image (datapoint 59)
  - Actual is 5 and Predicted is 7
  - Bar chart shows that probability of prediction of 7 is close to 80% and prediction of 5 is close to 20%







Copyright Satish Reddy 2020

Digit Classification – Summary of Results

- Confusion Matrix:
  - Most difficulty predicting digits 5 and 8
  - Digits 4 and 9 often mistaken for each other
  - Actual 7 is often predicted as 9

Confusion Matrix

		Actual									
		0	1	2	3	4	5	6	7	8	9
Predicted	0	938	0	10	3	1	19	8	0	19	10
	1	0	1103	4	3	1	1	2	6	4	2
	2	6	5	925	26	9	5	6	29	23	3
	3	5	9	36	919	2	48	2	17	45	15
	4	1	1	12	2	909	18	7	6	13	43
	5	4	2	0	17	2	738	14	1	27	3
	6	17	6	11	1	11	10	915	0	14	3
	7	3	0	17	16	2	18	1	908	16	17
	8	1	9	14	15	2	28	2	4	798	5
	9	5	0	3	8	43	7	1	57	15	908

Copyright Satish Reddy 2020

Further Investigations

- Results indicate underfitting has occurred
  - Investigate if adding additional layers and/or units to neural network addresses underfitting

Copyright Satish Reddy 2020

New Code for Digits Classification

Method	Input	Description
load_mnist	ntrain (integer) nvalid (integer)	Loads MNIST database Return: Xtrain, Ytrain, Xvalid, Yvalid
driver_casestudy _mnist		Driver for performing mnist training
plot_results_mnist_an imation	X (numpy array) Y (numpy array) Y_pred (numpy array) Afinal (numpy array) nframe (integer)	Shows animation of digit images (X) and prints actual label (Y) and predicted label (Y_pred), as well as probabilities for each digit (Afinal) for nframe images Return: nothing

Copyright Satish Reddy 2020

MNIST Digits Classification Walkthrough

- Code for walkthrough located at:

IntroML/Code/Version4.1

- Demo of pandas for loading and processing data in

IntroML/Examples/Chapter2/PandasDemo.ipynb

- You can implement the code additions suggested on the previous page by adding to a clean Version3.3 of the code
  - Have a look at Version4.1 for hints
- We will perform a walkthrough of the code additions for MNIST Digit Classification

Copyright Satish Reddy 2020