CS 229, Fall 2017, Project Writeup

Huy Le

December 15, 2017

- **Title**: Inference of subsurface properties by machine learning
- **Category**: Physical sciences (Geophysics)
- **Student**: Huy Le (SUNet ID 05802881)

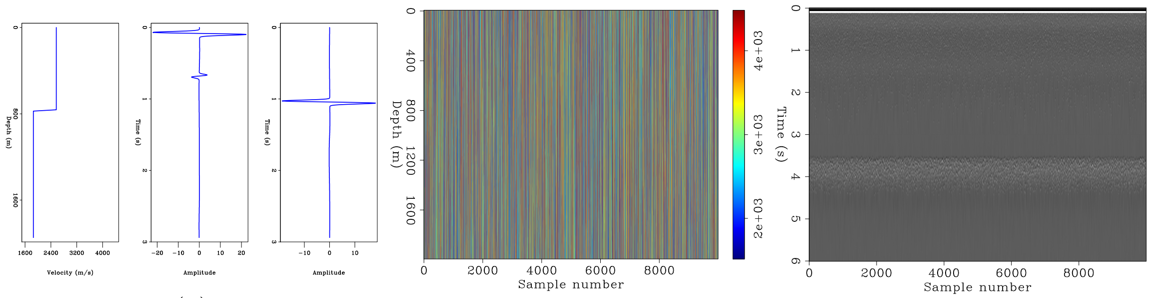
1 Motivation

Seismic waves that travel through the Earth carry useful information about its interior such as acoustic velocity of rocks, porosity, or location of reflection interfaces. Geophysicists use sensors to record seismic waves' waveform and arrival times at the surface and use these data to infer subsurface properties to, for example, better understand the Earth's history, study earthquakes, and explore energy resources. Inference of the Earth's properties from seismic data is conventionally performed by solving an optimization problem that minimizes the difference between observed data and simulated data using gradient descent methods. In this project, I propose to use machine learning to infer subsurface properties, specifically rocks' acoustic velocity and interfaces' depth, from recorded seismic data.

2 Data and labels

For simplicity, I consider one dimension only, the depth dimension, and the velocity profile consists of two layers down to depth of two kms. The seismic data is generated by solving an one dimensional wave equation by finite differences. Two sensors are used to record seismic waves: one at the surface and one at the bottom of the velocity profile. Figure 1a shows a sample velocity profile (left most panel), and two seismic traces: direct and reflected waves recorded by the sensor at the surface (middle panel) and transmitted wave recorded by the sensor at the bottom (right most panel). 10000 unique velocity profiles are generated randomly (Figure 1b) and their corresponding seismic data are simulated (Figure 1c). 100 profiles are set aside for testing purposes.

The seismic traces are my raw data. I later processed this data to obtain two arrival times, one for reflection and one for transmission, and one reflection coefficient. I also labeled these data in two ways: first, using only three parameters (two velocities and one depth) and second, 160 discrete velocity values at every 12.5 m. Later sections show experiments' results using different types of data (raw or processed) and different types of labels (three parameters or discrete).



(a) (b) (c)

Figure 1: (a) One sample velocity profile and its data. (b) 10000 unique randomly generated one-dimensional velocity profiles. (c) Recorded seismic data: direct and reflection waves from 0-3 seconds and transmitted waves from 3-6 seconds.

3 Method

I have experimented with neural networks using Google's tensorflow library. Given that there is a relationship between the my labels (the velocity profiles) and my data (the seismic traces) through the wave equation, neural networks seem to be a good method to approximate this relationship. Since my setting is really simple with only one dimensional velocity profiles and two layers, I chose a dense neural network with only one hidden layer of 100 neurons.

First, I formulate the problem as a classification one with two depth classes and three velocity classes. For this I use sigmoid activation function for the hidden layer and softmax function for the output layer with cross-entropy loss function. Second, I consider the problem as a regression. For this case, I use relu activation function for the hidden layer and none for the output layer with least-square loss function. For both cases, I use mini-batch gradient descent to minimize the loss functions.

4 As a classification problem

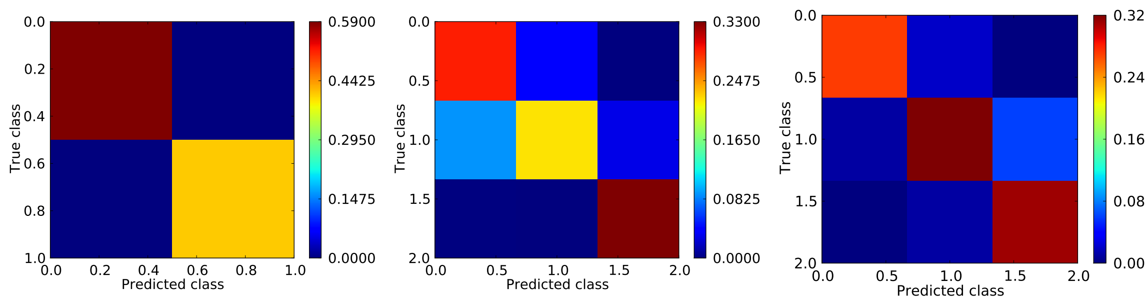
I divide the total depth range (0-2 km) into two classes and velocity range (1.5-4.5 km/s) into three classes (Table 1). Since my parameters consist of one depth and two velocities, total number of classes is $2 \times 3 \times 3 = 18$. Figure 2 shows the confusion matrices for depth (panel a), top velocity (panel b), and bottom velocity (panel c) when processed data (arrival times and reflection coefficient) are used as features. These matrices are diagonally dominant, stating that there are more correct classifications than misses. Note that the confusion matrix for depth is actually diagonal, meaning depth is classified perfectly. Table 2 details all the mis-classified test samples. Out of 100 test sample, classification with processed data misses 26. Interestingly, depth is classified perfectly and mis-classifications occur either in velocity above or below.

Figure 3 shows the confusion matrices for depth (panel a), top velocity (panel b), and bottom velocity (panel c) when the raw waveform data are used as features. Similar to classification with processed data, these matrices are also diagonally dominant. The depth confusion matrix is, however, not actually diagonal, meaning that depth is not classified perfectly as in the case with processed data. Table 3 details all the mis-classified test samples. Classification with raw data misses 22, slightly less than with processed data. However, mis-classifications occur all parameters.

Figure 4 shows the loss function (panel a) and accuracy (panle b) of training data and test data for classification experiments. I observe that when feature is the raw waveform, training loss reduces and accuracy improves compared to processed data as features, but test loss and accuracy don't differ much for both types of data. Another observation is the loss function curves and accuracy curves seem to mimic each other.

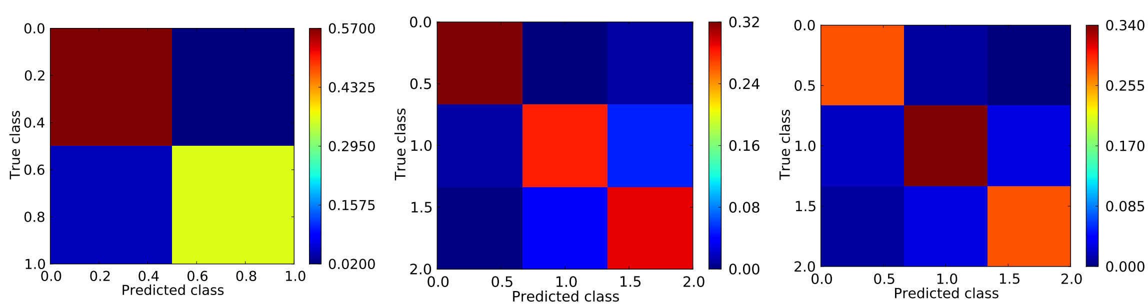
Depth classes	0-1 km	1-2 km	
Velocity classes	1.5-2.5 km/s	2.5-3.5 km/s	3.5-4.5 km/s

Table 1: Two classes of depth and three classes of velocity.



(a) (b) (c)

Figure 2: Confusion matrices for depth (a), v_1 (b), and v_2 (c) when processed data are used as features.



(a) (b) (c)

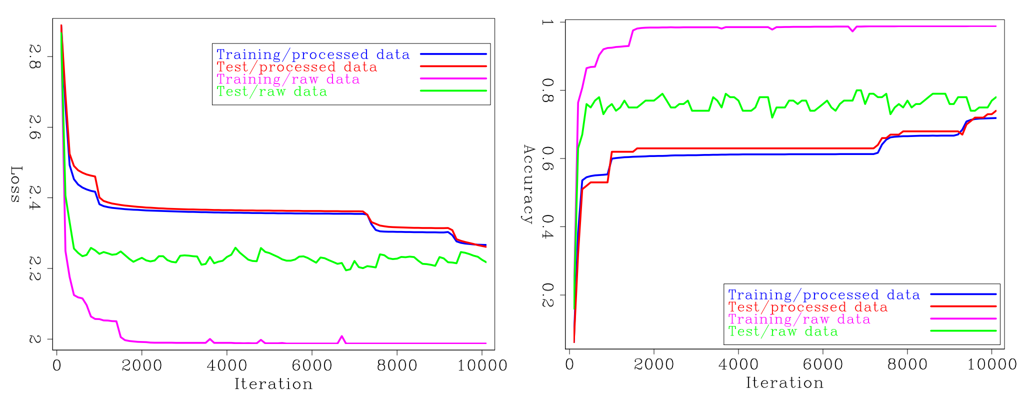
Figure 3: Confusion matrices for depth (a), v_1 (b), and v_2 (c) when raw waveform data are used as features.

Test sample number	Depth miss	v_1 miss	v_2 miss
11		x	
13		x	
14		x	
16			x
21		x	
22		x	
31		x	
33		x	
38		x	
39		x	
45			x
46		x	
66			x
69		x	
70		x	
73		x	
76			x
78			x
79			x
84			x
86			x
87		x	
88			x
94		x	
95			x
97		x	
Total miss 26	0	16	10

Table 2: Summary of classification results when processed data are used as features.

Test sample number	Depth miss	v_1 miss	v_2 miss
14		x	
15		x	
19		x	
21		x	
35		x	x
37		x	x
39		x	
48		x	
49		x	
50			x
52	x		
53			x
59		x	
61	x		
62	x		
64			x
66	x		x
67			x
68	x	x	x
74			x
90	x		x
91	x		
Total miss 22	7	11	10

Table 3: Summary of classification results when the raw waveforms are used.



(a) (b)

Figure 4: Loss function (a) and accuracy (b) of training data and test data for classification problem.

5 As a regression problem

I have performed four experiments using different types of data and labels. For each of these experiments, I trained the network on 20%, 40%, 60%, 80%, and 100% of training data. Table 4 summarizes the test errors of my experiments when all training data is used. Among all experiments, the one that used processed data and three parameter label type perform best and the one that used raw data with three parameter label type performs worst.

Data type	Label type	Average velocity error	Color
processed	three parameters	23.54 m/s	red
processed	discrete	75.04 m/s	green
raw	three parameters	140.15 m/s	magenta
raw	discrete	104.94 m/s	cyan

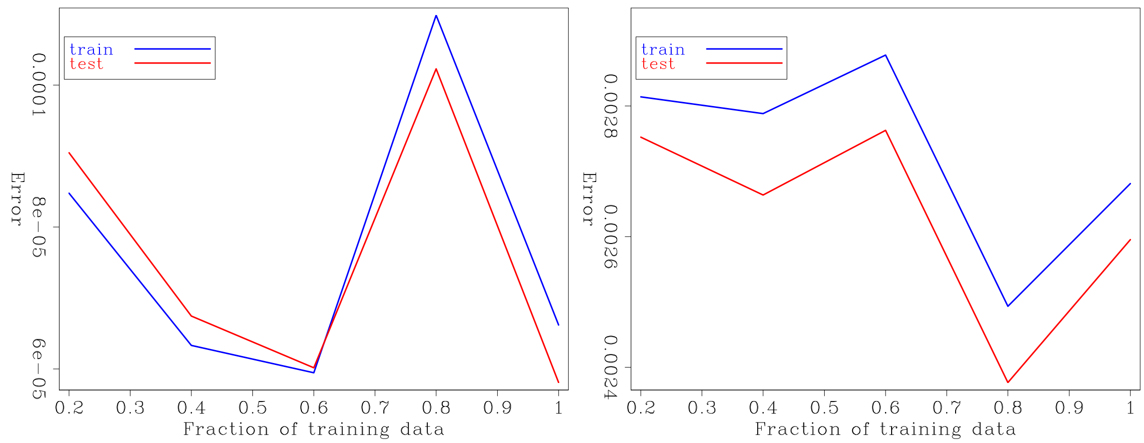
Table 4: Summary of regression results.

Figures 5a and 5a show respectively the learning curves for three parameter label type and discrete value label type using processed data. Similar learning curves for raw data are shown in Figures 6a and 6a. From these figures I notice that when processed data is used, there is not much improvement in increasing training data size and the test error seems to follow the training error closely (Figure 5). On the other hand, when raw data is used, the test error reduces as training size increases. Figure 7 shows how training error decreases with iteration. Most significant reduction occurs after the first 2000 iterations.

Figure 8 shows five good predictions from my experiments and Figure 9 shows the corresponding simulated data. For these test profiles, the network is able to output correct values for the two velocities and close values for the interface's depth. The simulated data match the test data in both reflection and transmission. I notice that when processed data is used, the network predicts smooth velocity profiles (green curves in Figure 8) even when the data is labeled with discrete values. On the other hand, when the raw data is used, the predicted velocity profiles are much more noisy.

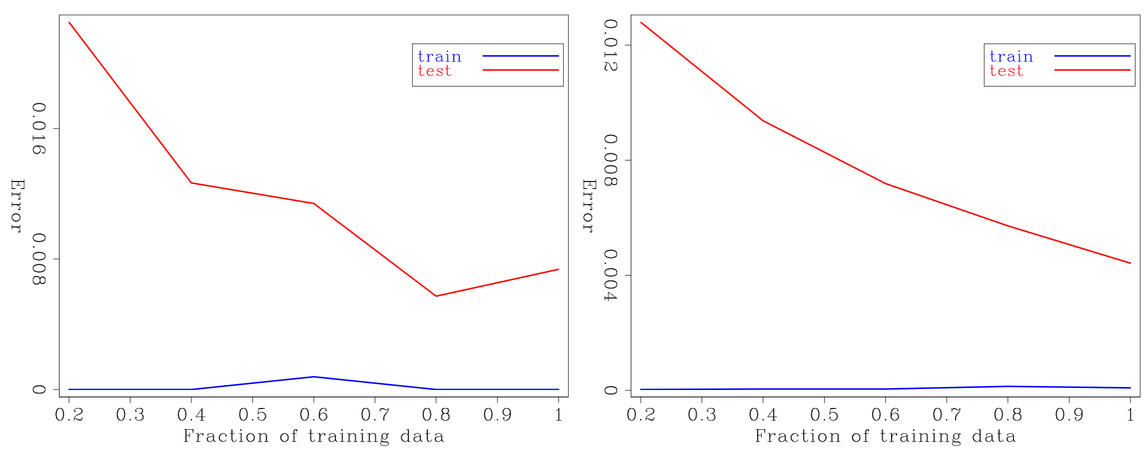
Figure 10 shows five profiles that the network performs poorly on and Figure 11 shows the corresponding simulated data. On these test samples, the network missed either or both depth of the interface and velocity. Interestingly, the simulated data show that most of the transmission events are matched well with the test data. The predictions with processed data seem to be better than those with raw data.

Figure 12 shows all the test samples and their simulated data. Figures 13, 14, 15, and 16, show predicted velocity profiles and corresponding simulated data from different experiments. In general, the predicted velocity profiles/data match the test profiles/data relatively well. I observe that when velocity profile is labeled with three parameters, the simulated data from predictions agree well with the test data in both reflection and transmission (Figures 13b and 15b). On the other hand, when trying to predict discrete velocity values, the simulated data agree with test data only in transmission events while reflection energy is smeared (Figures 14b and 16b).



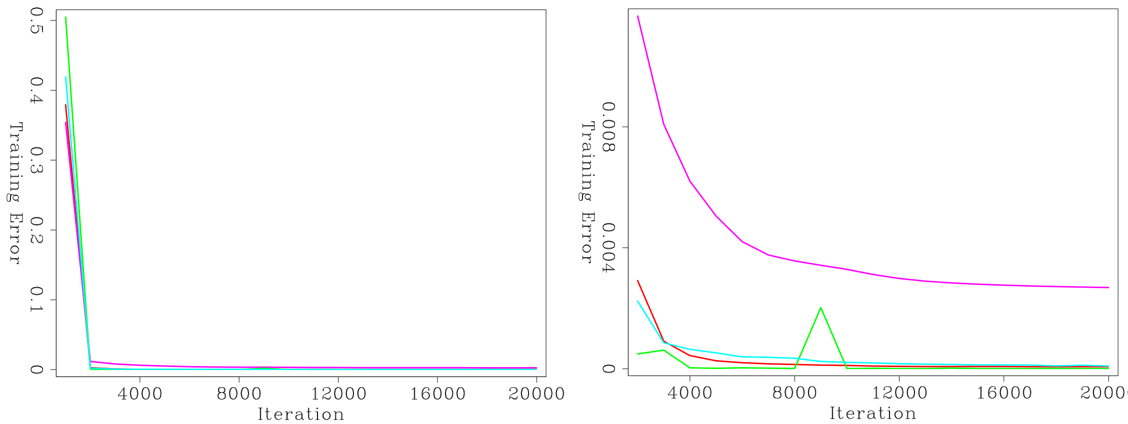
(a) (b)

Figure 5: Learning curves for processed data with three parameter labels (left) and discrete value label (right).



(a) (b)

Figure 6: Learning curves for raw data with three parameter labels (left) and discrete value label (right).



(a) (b)

Figure 7: Training error with iteration for different experiments. See Table 4 for color code.

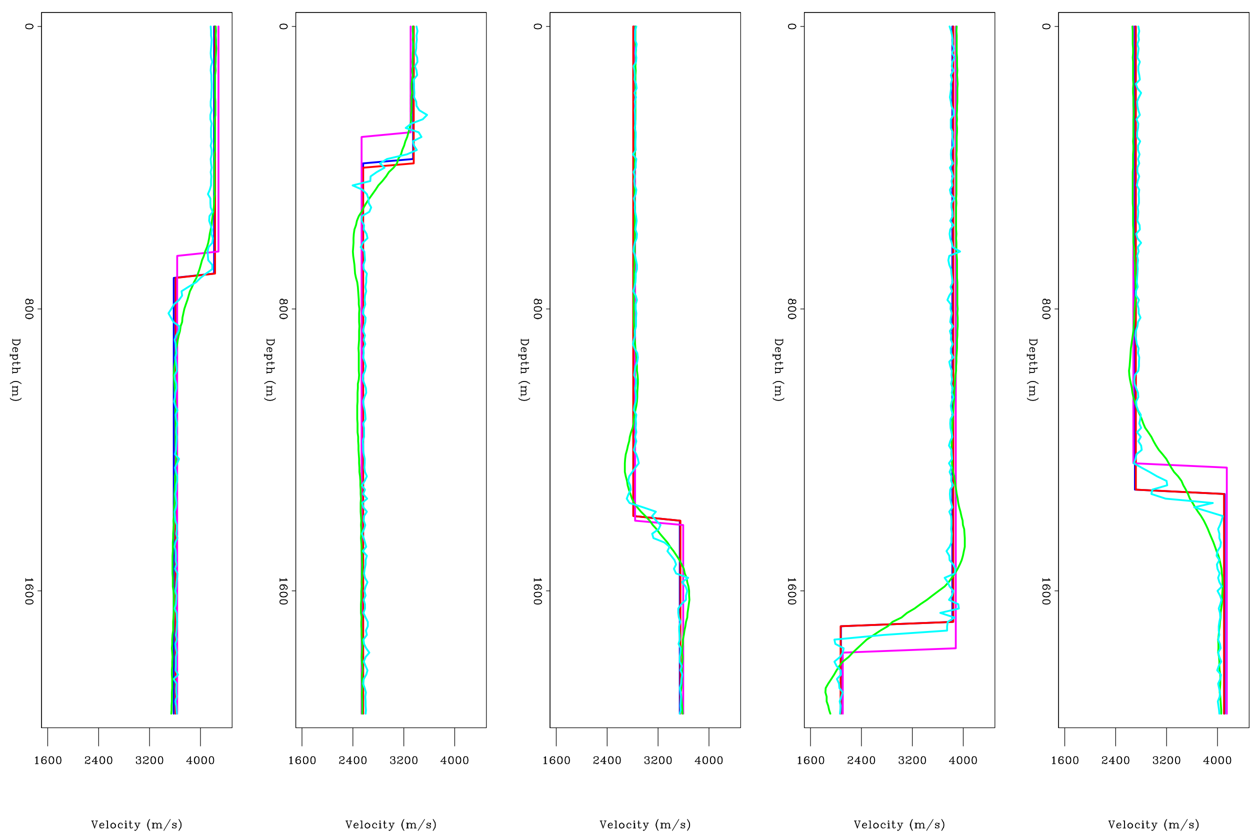


Figure 8: Five good predictions from different experiments. See Table 4 for color code. True profile is shown in blue.

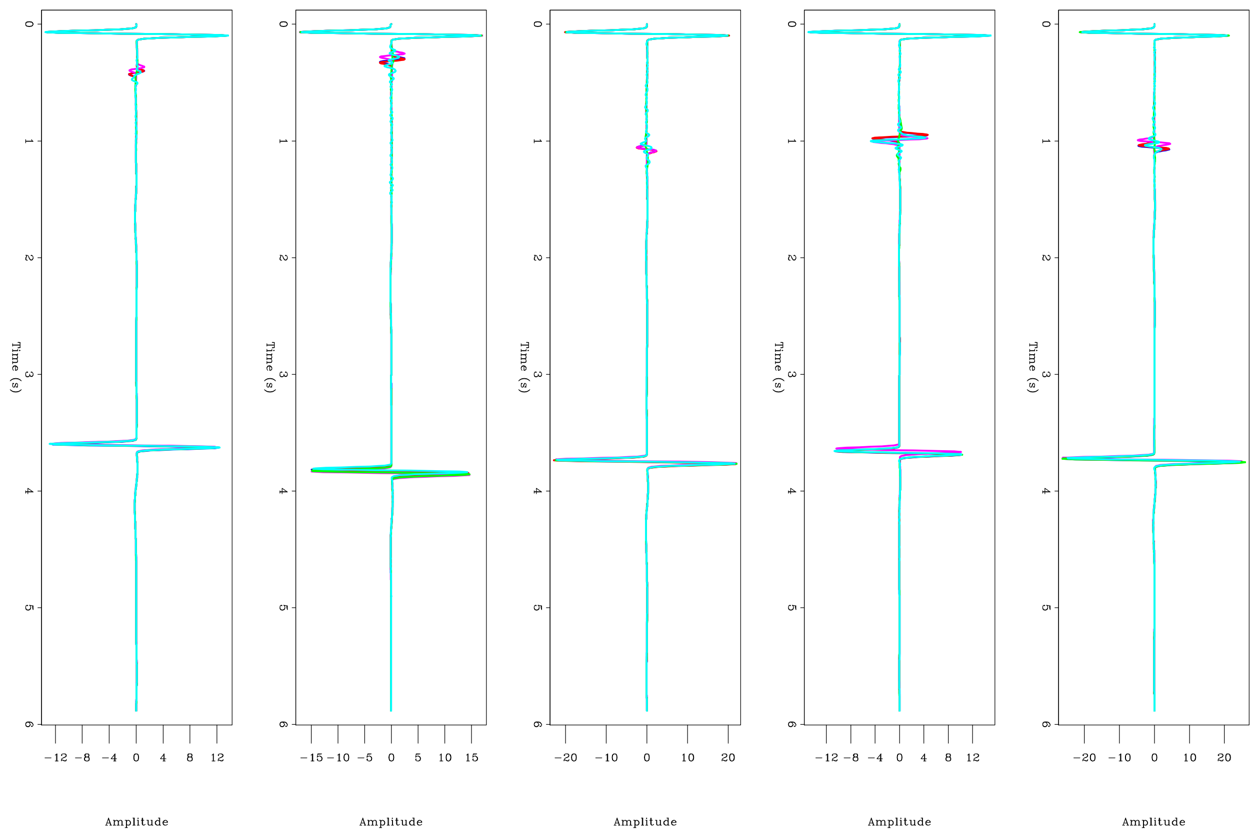


Figure 9: Simulated data from five good predictions from different experiments. See Table 4 for color code. Data simulated from true profile is shown in blue.

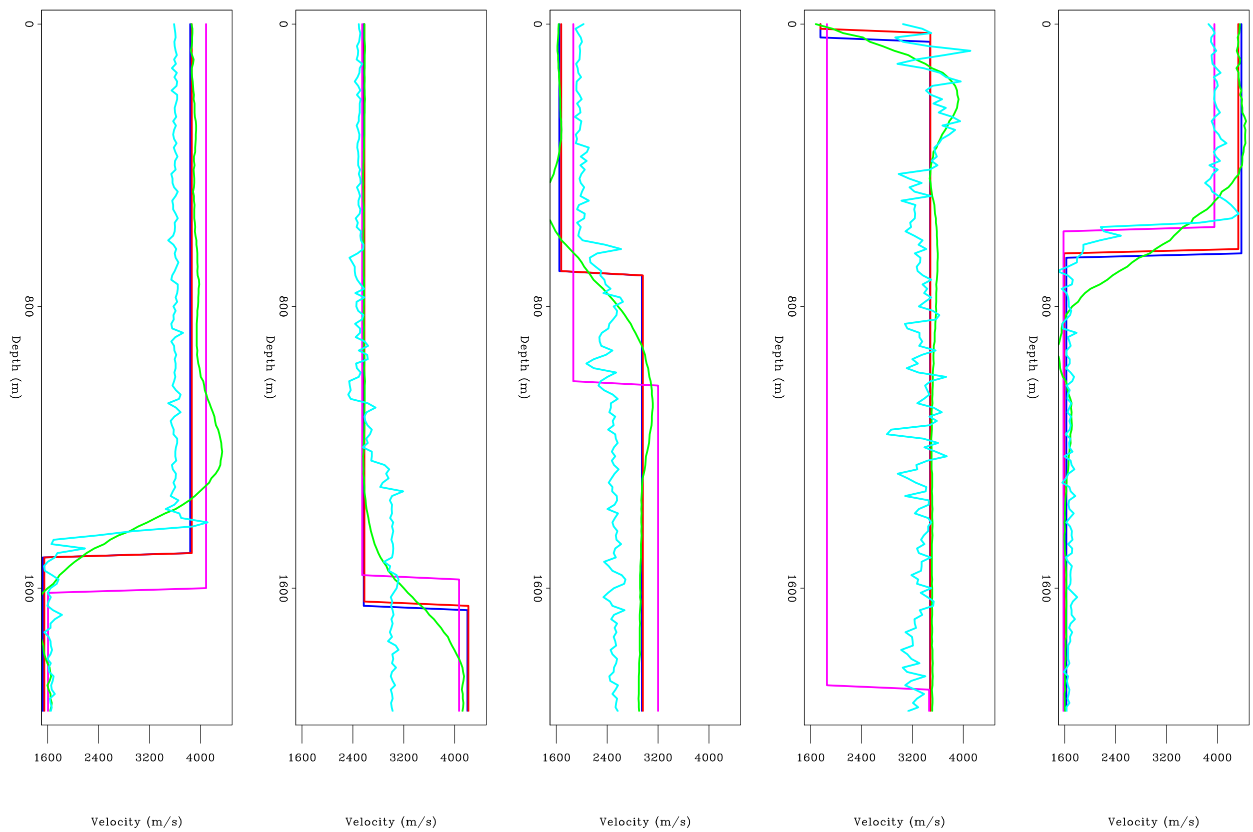


Figure 10: Five poor predictions from different experiments. See Table 4 for color code. True profile is shown in blue.

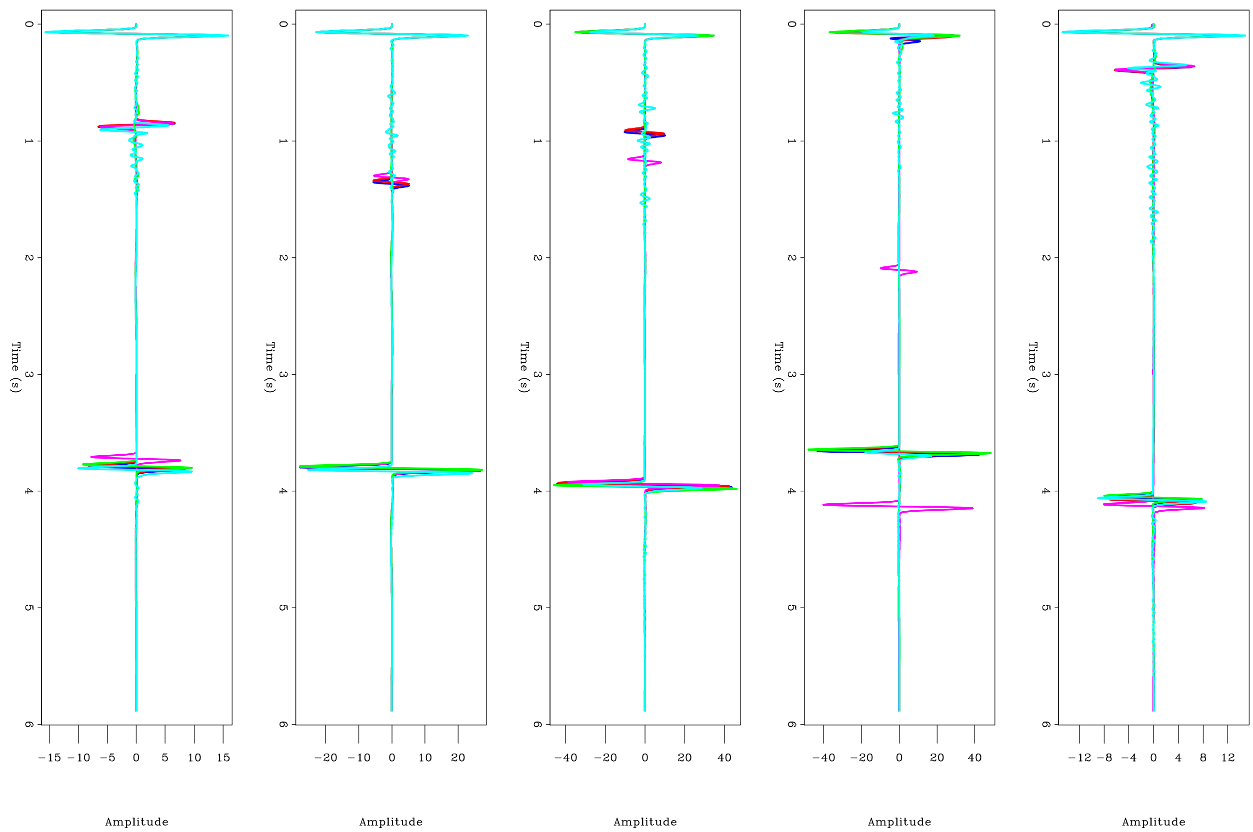
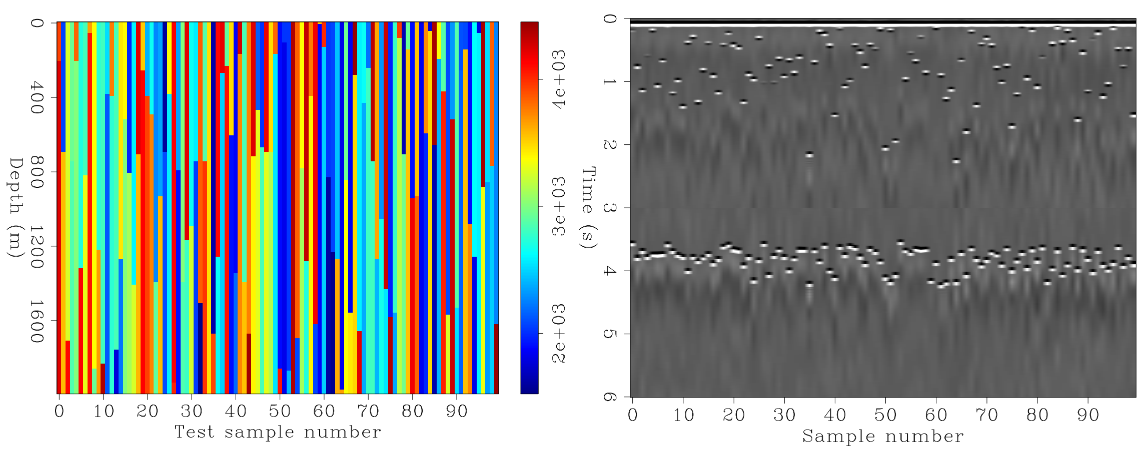
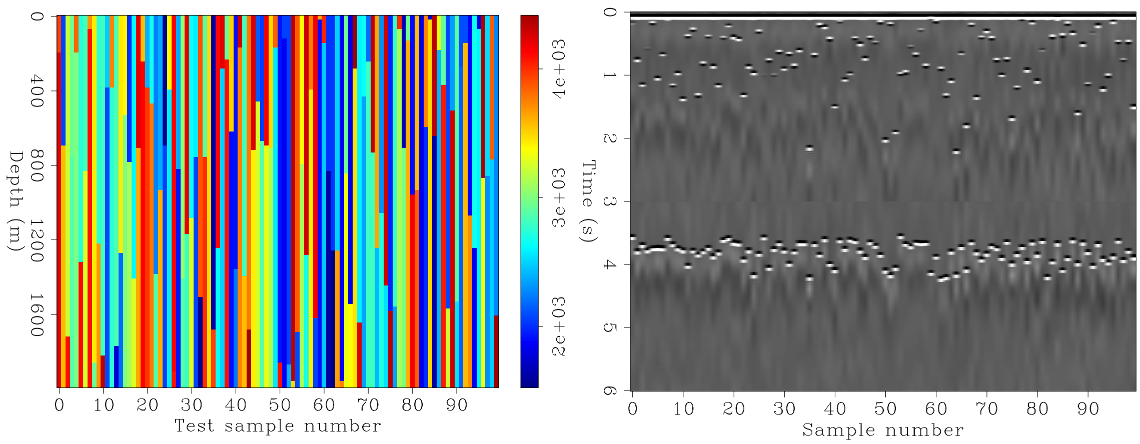


Figure 11: Simulated data from five unsuccessful predictions from different experiments. See Table 4 for color code. Data simulated from true profile is shown in blue.



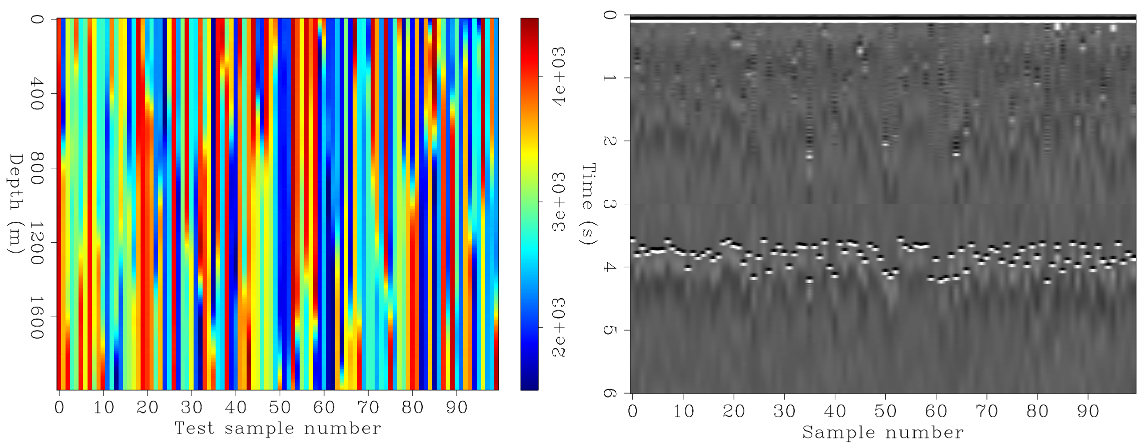
(a) (b)

Figure 12: (a) Test profiles and (b) simulated data.



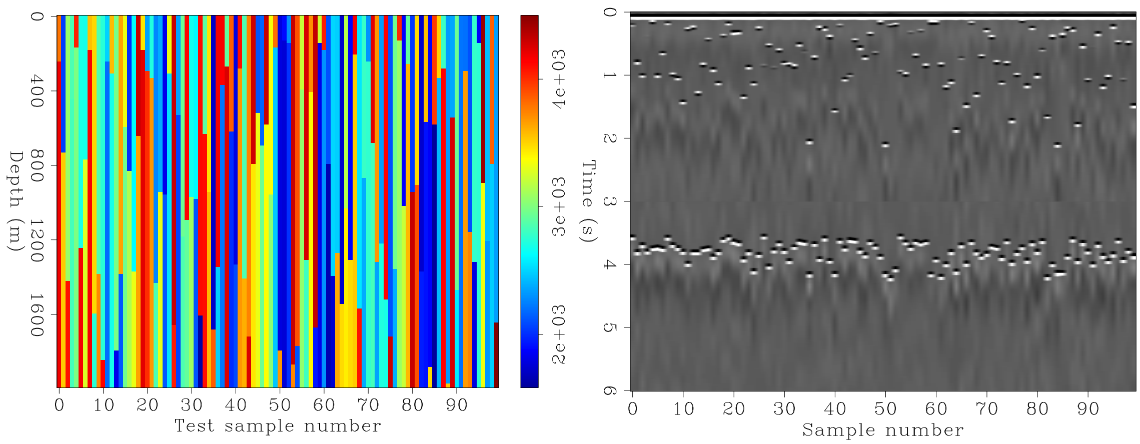
(a) (b)

Figure 13: (a) Predicted profiles and (b) simulated data when using processed data and three-parameter label. Compare with Figure 12.



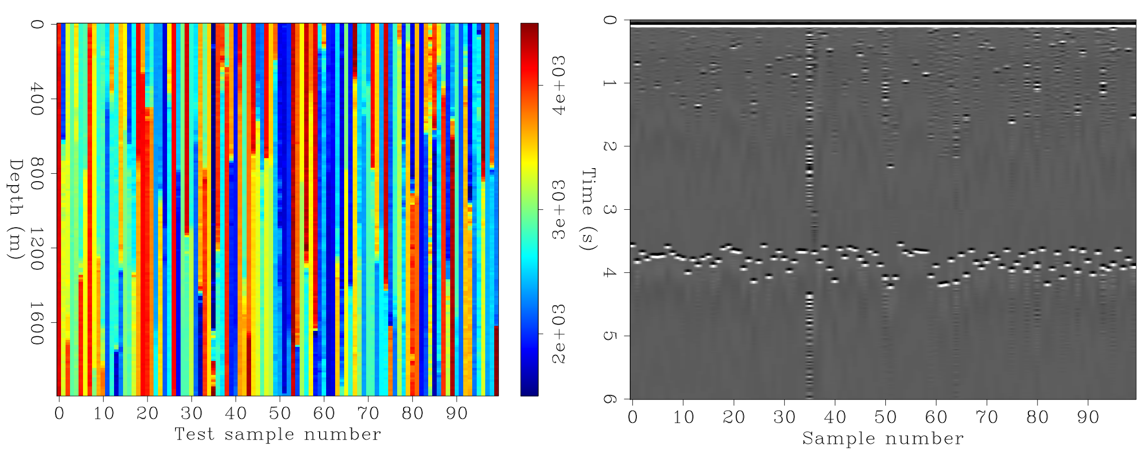
(a) (b)

Figure 14: (a) Predicted profiles and (b) simulated data when using processed data and discrete-velocity label. Compare with Figure 12.



(a) (b)

Figure 15: (a) Predicted profiles and (b) simulated data when using raw data and three-parameter label. Compare with Figure 12.



(a) (b)

Figure 16: (a) Predicted profiles and (b) simulated data when using raw data and discrete-velocity label. Compare with Figure 12.

6 Conclusions

For the classification problem, both types of data have similar performance on the test set, roughly 75% accuracy. When processed data is used as features, the network classifies reflector's depth perfectly and only misses either one of two velocities, while when raw waveform is used, mis-classifications occur in all parameters. For the regression problem, prediction with processed data and three-parameter label gives the best result while using raw waveforms to predict three-parameter label is worst. When processed data and discrete velocity label are used, the network predicts smooth velocity profiles. When velocity profile is labeled with three parameters, the simulated data from predictions agree well with the test data in both reflection and transmission, while trying to predict discrete velocity values, the simulated data agree with test data only in transmission events while reflection energy is smeared.

References

[1] Aaron Moya and Kojiro Irikura, 2010, Inversion of a velocity model using artificial neural networks, Computer and Geosciences, Vo. 36, 1474-1483.

[2] Carlos Calderón-Macías, Mrinal K. Sen, and Paul L. Stoffa, 2000, Artificial neural networks for parameters estimation in geophysics, Geophysical Prospecting, Vol. 48, 21-47.

[3] C. Baronian, M. A. Riahi, and C. Lucas, 2009, Applicability of artificial neural networks for obtaining velocity models from synthetic seismic data, International Journal of Earth Sciences, Vol. 89, 1173-1184.

[4] Gunter Roth and Albert Tarantola, 1994, Neural networks and inversion of seismic data, Journal of Geophysical Research, Vol. 99, No. B4, 6753-6768.

[5] Horst Langer, Giuseppe Nunnari, and Luigi Occhipinti, 1996, Estimation of seismic waveform governing parameters with neural networks, Journal of Geophysical Research, Vol. 101, 20109-20118.