Native Language Identification

Using Bert vectors for Text Classification

Chirag Bhansali

(A20436467)

04/23/3030

—

Natural Language Processing

—

Prof. Derrick Higgins

Instructions to Reproduce the Results

Requirements

1. Anaconda version 4.7.12 with Python version 3.7.4.
2. Python Libraries: a. NumPy

b. Pandas

c. Seaborn

d. Matplotlib

e. Sci-Kit Learn

Directory Structure

```
BERT
|-- BERT_BASE_DIR (BERT Repo cloned from GitHub)
|-- bert_input_data
|    |-- eval.txt
|    |-- test.txt
|    |-- train.txt
|-- bert_output_data
|    |-- eval.jsonlines
|    |-- test. jsonlines
|    |-- train. jsonlines
|-- handout
|    |-- command.sh
|    |-- run_bert_fv.sh
|    |-- data
|    |    |-- lang_id_eval.csv
|    |    |-- lang_id_test.csv
|    |    |-- lang_id_train.csv
|-- solution
|   |-- Assignment4ChiragBhansaliSubmission.pdf
|   |-- Assignment_4_Chirag_Bhansali_Submission.ipynb
|   |-- neural_network_final.csv
```

Steps to reproduce the results:

1. Clone BERT Repo from GitHub and store it in folder *BERT_BASE_DIR*
2. Download pre-trained BERT model and decompress it into *BERT_BASE_DIR*.
3. Create the *bert_input_directory* that contains the data files from the given handout material.
4. Programmatically re-format the datafiles from the handout materials so that they can be processed by the BERT extract_features.py script
5. The file contains the bash commands that will reformat the files.

```
sed 's/\([^,]*\),\(.*\)/\2/' ./data/lang_id_eval.csv > ../bert_input_data/eval.csv
sed 's/\([^,]*\),\(.*\)/\2/' ./data/lang_id_train.csv > ../bert_input_data/train.csv
sed 's/\([^,]*\),\(.*\)/\2/' ./data/lang_id_test.csv > ../bert_input_data/test.csv

sed 1d ../bert_input_data/eval.csv > ../bert_input_data/eval.txt
sed 1d ../bert_input_data/train.csv > ../bert_input_data/train.txt
sed 1d ../bert_input_data/test.csv > ../bert_input_data/test.txt

rm ../bert_input_data/eval.csv
rm ../bert_input_data/train.csv
rm ../bert_input_data/test.csv
```

6. Since I am using Anaconda, I have modified run_bert_fv.sh file in order to run it using the activated conda base.
7. Execute jupyter notebook *Assignment_4_Chirag_Bhansali_Submission.ipynb* to get the misclassification rates and the observations

Summary of findings:

We begin by using accuracy as our metric to measure the overall performance on the test set. Accuracy is defined as fraction of correct predictions = correct predictions / total number of predictions

Using Logistic regression model we get an accuracy of 48% (approx.) in our model. Although accuracy gives the overall performance of the model it alone is not enough to measure the performance of a model. The other important metrics to evaluate a model are precision, recall and f1 score which are an accurate measure of the performance of the model.

Have applied logistic regression model and 36 different neural network models using different classifier and evaluated each one of them on following points:

1. Accuracies
2. Confusion Matrix
3. Evaluation metrics for each class
4. Within Class Misclassifications
5. Short Summary on Misclassifications

For neural networks, we build multiple different models and use the evaluation data accuracy as a metric for fine tuning the model. Out of all the neural network models, below are the best models based on misclassification and lowest loss.

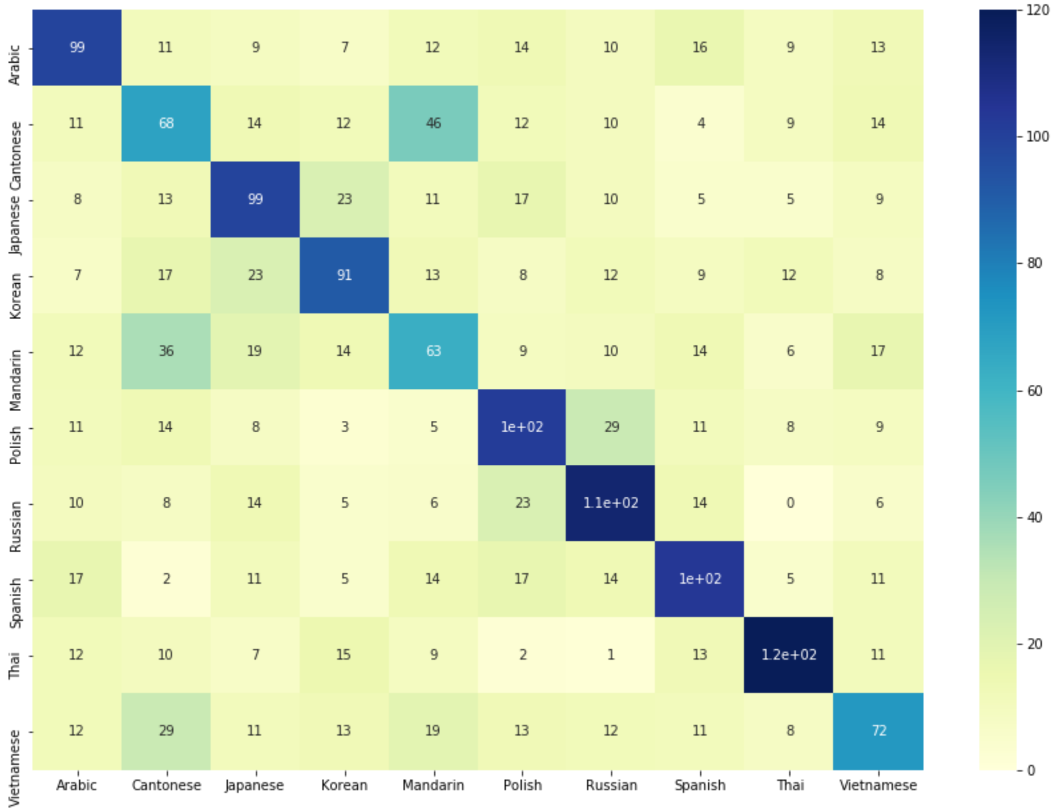
1. Logistic Regression
2. Neural Network with lowest loss (Activation=tanh)
3. Neural Network with lowest misclassification (Activation = identity)

Evaluation of Logistic Regression Model

1) Accuracies:

Training Accuracy: 0.7345
Evaluation Accuracy: 0.4855
Testing Accuracy: 0.466

2) Confusion Matrix:



3) Evaluation Metrics for each language:

Language	Misclassification	Precision	Recall	F1 Score
Arabic	10.05	0.497487	0.495	0.496241
Cantonese	13.6	0.326923	0.34	0.333333
Japanese	10.85	0.460465	0.495	0.477108
Korean	10.3	0.484043	0.455	0.469072
Mandarin	13.6	0.318182	0.315	0.316583
Polish	10.65	0.470046	0.51	0.489209
Russian	9.7	0.513514	0.57	0.540284
Spanish	9.65	0.517413	0.52	0.518703
Thai	7.1	0.659341	0.6	0.628272
Vietnamese	11.3	0.423529	0.36	0.389189

4) Within class Misclassifications:

Language	Predicted	Misclassification
Arabic	Cantonese	11
Arabic	Japanese	9
Arabic	Korean	7
Arabic	Mandarin	12
Arabic	Polish	14
Arabic	Russian	10
Arabic	Spanish	16
Arabic	Thai	9
Arabic	Vietnamese	13
Cantonese	Arabic	11
Cantonese	Japanese	14
Cantonese	Korean	12
Cantonese	Mandarin	46
Cantonese	Polish	12
Cantonese	Russian	10
Cantonese	Spanish	4
Cantonese	Thai	9
Cantonese	Vietnamese	14
Japanese	Arabic	8
Japanese	Cantonese	13
Japanese	Korean	23
Japanese	Mandarin	11
Japanese	Polish	17
Japanese	Russian	10
Japanese	Spanish	5
Japanese	Thai	5
Japanese	Vietnamese	9
Korean	Arabic	7
Korean	Cantonese	17
Korean	Japanese	23
Korean	Mandarin	13
Korean	Polish	8
Korean	Russian	12
Korean	Spanish	9
Korean	Thai	12
Korean	Vietnamese	8
Mandarin	Arabic	12
Mandarin	Cantonese	36
Mandarin	Japanese	19
Mandarin	Korean	14

Mandarin	Polish	9
Mandarin	Russian	10
Mandarin	Spanish	14
Mandarin	Thai	6
Mandarin	Vietnamese	17
Polish	Arabic	11
Polish	Cantonese	14
Polish	Japanese	8
Polish	Korean	3
Polish	Mandarin	5
Polish	Russian	29
Polish	Spanish	11
Polish	Thai	8
Polish	Vietnamese	9
Russian	Arabic	10
Russian	Cantonese	8
Russian	Japanese	14
Russian	Korean	5
Russian	Mandarin	6
Russian	Polish	23
Russian	Spanish	14
Russian	Thai	0
Russian	Vietnamese	6
Spanish	Arabic	17
Spanish	Cantonese	2
Spanish	Japanese	11
Spanish	Korean	5
Spanish	Mandarin	14
Spanish	Polish	17
Spanish	Russian	14
Spanish	Thai	5
Spanish	Vietnamese	11
Thai	Arabic	12
Thai	Cantonese	10
Thai	Japanese	7
Thai	Korean	15
Thai	Mandarin	9
Thai	Polish	2
Thai	Russian	1
Thai	Spanish	13
Thai	Vietnamese	11
Vietnamese	Arabic	12
Vietnamese	Cantonese	29

Vietnamese	Japanese	11
Vietnamese	Korean	13
Vietnamese	Mandarin	19
Vietnamese	Polish	13
Vietnamese	Russian	12
Vietnamese	Spanish	11
Vietnamese	Thai	8

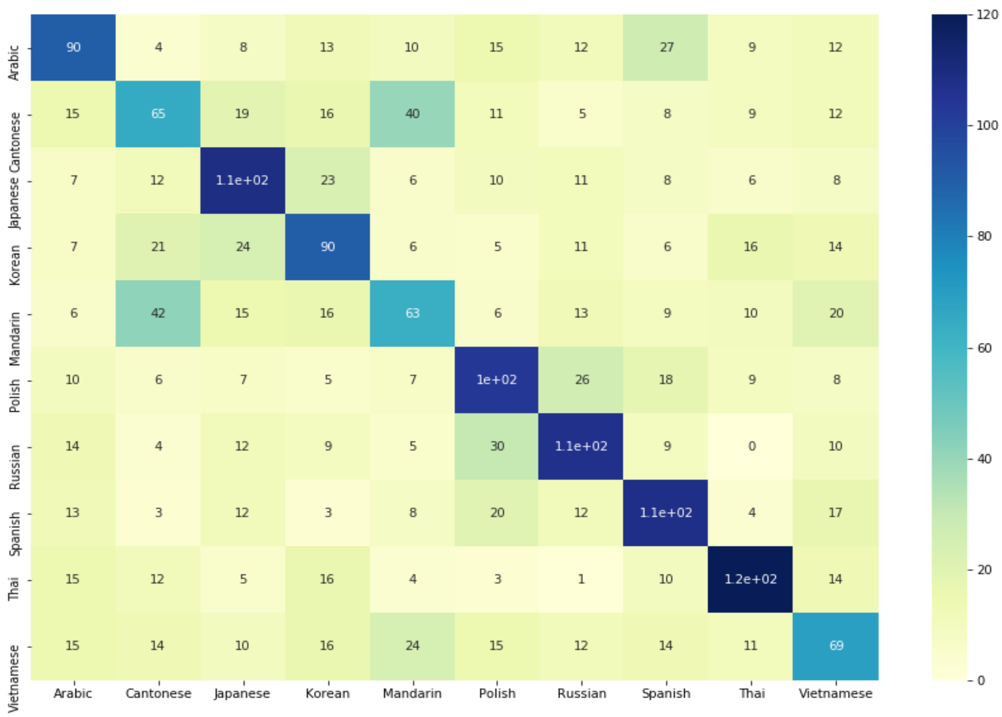
5) Short Summary on Misclassifications:

Total data: 2000
Total predicted incorrect: 1068
Total predicted correct: 932

Evaluation of Neural Network with Lowest Misclassification Model

1) **Accuracies:**
   Training Accuracy: 0.5656666666666667
   Evaluation Accuracy: 0.488
   Testing Accuracy: 0.4625

2) **Confusion Matrix:**



3) **Evaluation Metrics for each language:**

Language	Misclassification	Precision	Recall	F1 Score
Arabic	10.6	0.46875	0.45	0.459184
Cantonese	12.65	0.355191	0.325	0.339426
Japanese	10.15	0.493213	0.545	0.517815
Korean	11.35	0.434783	0.45	0.44226
Mandarin	12.35	0.364162	0.315	0.337802
Polish	10.55	0.474886	0.52	0.49642
Russian	9.8	0.509524	0.535	0.521951
Spanish	10.05	0.497696	0.54	0.517986
Thai	7.7	0.618557	0.6	0.609137
Vietnamese	12.3	0.375	0.345	0.359375

4) Within class Misclassifications:

Language	Predicted	Misclassification
Arabic	Cantonese	4
Arabic	Japanese	8
Arabic	Korean	13
Arabic	Mandarin	10
Arabic	Polish	15
Arabic	Russian	12
Arabic	Spanish	27
Arabic	Thai	9
Arabic	Vietnamese	12
Cantonese	Arabic	15
Cantonese	Japanese	19
Cantonese	Korean	16
Cantonese	Mandarin	40
Cantonese	Polish	11
Cantonese	Russian	5
Cantonese	Spanish	8
Cantonese	Thai	9
Cantonese	Vietnamese	12
Japanese	Arabic	7
Japanese	Cantonese	12
Japanese	Korean	23
Japanese	Mandarin	6
Japanese	Polish	10
Japanese	Russian	11
Japanese	Spanish	8
Japanese	Thai	6
Japanese	Vietnamese	8
Korean	Arabic	7
Korean	Cantonese	21
Korean	Japanese	24
Korean	Mandarin	6
Korean	Polish	5
Korean	Russian	11
Korean	Spanish	6
Korean	Thai	16
Korean	Vietnamese	14
Mandarin	Arabic	6
Mandarin	Cantonese	42
Mandarin	Japanese	15

Mandarin	Korean	16
Mandarin	Polish	6
Mandarin	Russian	13
Mandarin	Spanish	9
Mandarin	Thai	10
Mandarin	Vietnamese	20
Polish	Arabic	10
Polish	Cantonese	6
Polish	Japanese	7
Polish	Korean	5
Polish	Mandarin	7
Polish	Russian	26
Polish	Spanish	18
Polish	Thai	9
Polish	Vietnamese	8
Russian	Arabic	14
Russian	Cantonese	4
Russian	Japanese	12
Russian	Korean	9
Russian	Mandarin	5
Russian	Polish	30
Russian	Spanish	9
Russian	Thai	0
Russian	Vietnamese	10
Spanish	Arabic	13
Spanish	Cantonese	3
Spanish	Japanese	12
Spanish	Korean	3
Spanish	Mandarin	8
Spanish	Polish	20
Spanish	Russian	12
Spanish	Thai	4
Spanish	Vietnamese	17
Thai	Arabic	15
Thai	Cantonese	12
Thai	Japanese	5
Thai	Korean	16
Thai	Mandarin	4
Thai	Polish	3
Thai	Russian	1
Thai	Spanish	10
Thai	Vietnamese	14
Vietnamese	Arabic	15

Vietnamese	Cantonese	14
Vietnamese	Japanese	10
Vietnamese	Korean	16
Vietnamese	Mandarin	24
Vietnamese	Polish	15
Vietnamese	Russian	12
Vietnamese	Spanish	14
Vietnamese	Thai	11

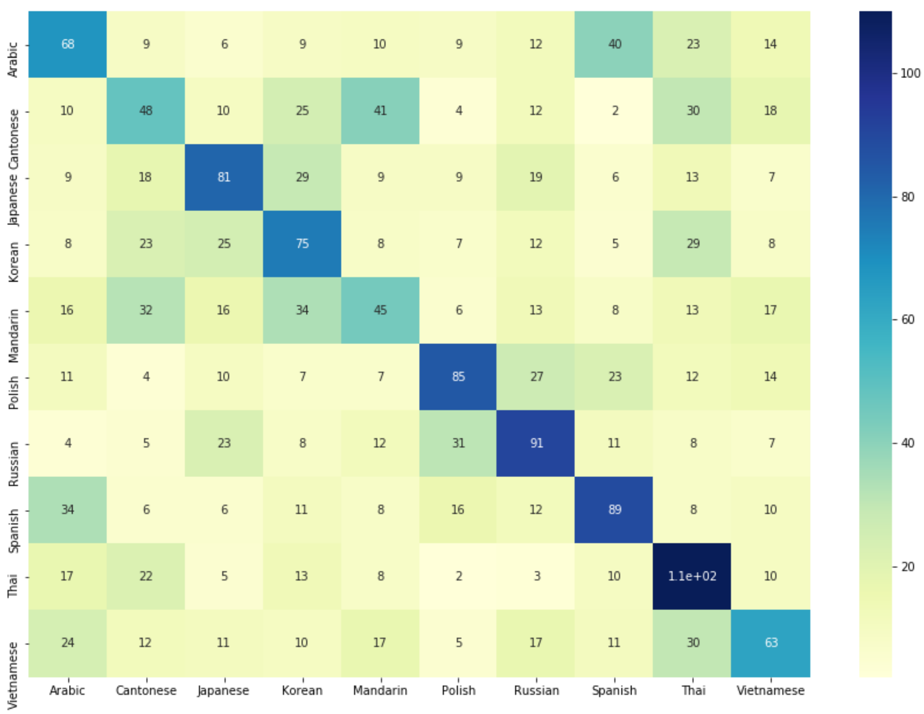
5) Short Summary on Misclassifications:

Total data: 2000
Total predicted incorrect: 1075
Total predicted correct: 925

Evaluation of Neural Network with Lowest Loss Model

1) **Accuracies:**
   Training Accuracy: 0.9958333333333333
   Evaluation Accuracy: 0.3825
   Testing Accuracy: 0.3775

2) **Confusion Matrix:**



3) **Evaluation Metrics for each language:**

Language	**Misclassification**	**Precision**	**Recall**	F1 Score
Arabic	13.25	0.338308	0.34	0.339152
Cantonese	14.15	0.268156	0.24	0.253298
Japanese	11.55	0.419689	0.405	0.412214
Korean	13.55	0.339367	0.375	0.356295
Mandarin	13.75	0.272727	0.225	0.246575
Polish	10.2	0.488506	0.425	0.454545
Russian	11.8	0.417431	0.455	0.435407
Spanish	11.35	0.434146	0.445	0.439506
Thai	12.8	0.398551	0.55	0.462185
Vietnamese	12.1	0.375	0.315	0.342391

4) Within class Misclassifications:

Language	Predicted	Misclassification
Arabic	Cantonese	9
Arabic	Japanese	6
Arabic	Korean	9
Arabic	Mandarin	10
Arabic	Polish	9
Arabic	Russian	12
Arabic	Spanish	40
Arabic	Thai	23
Arabic	Vietnamese	14
Cantonese	Arabic	10
Cantonese	Japanese	10
Cantonese	Korean	25
Cantonese	Mandarin	41
Cantonese	Polish	4
Cantonese	Russian	12
Cantonese	Spanish	2
Cantonese	Thai	30
Cantonese	Vietnamese	18
Japanese	Arabic	9
Japanese	Cantonese	18
Japanese	Korean	29
Japanese	Mandarin	9
Japanese	Polish	9
Japanese	Russian	19
Japanese	Spanish	6
Japanese	Thai	13
Japanese	Vietnamese	7
Korean	Arabic	8
Korean	Cantonese	23
Korean	Japanese	25
Korean	Mandarin	8
Korean	Polish	7
Korean	Russian	12
Korean	Spanish	5
Korean	Thai	29
Korean	Vietnamese	8
Mandarin	Arabic	16
Mandarin	Cantonese	32
Mandarin	Japanese	16

Mandarin	Korean	34
Mandarin	Polish	6
Mandarin	Russian	13
Mandarin	Spanish	8
Mandarin	Thai	13
Mandarin	Vietnamese	17
Polish	Arabic	11
Polish	Cantonese	4
Polish	Japanese	10
Polish	Korean	7
Polish	Mandarin	7
Polish	Russian	27
Polish	Spanish	23
Polish	Thai	12
Polish	Vietnamese	14
Russian	Arabic	4
Russian	Cantonese	5
Russian	Japanese	23
Russian	Korean	8
Russian	Mandarin	12
Russian	Polish	31
Russian	Spanish	11
Russian	Thai	8
Russian	Vietnamese	7
Spanish	Arabic	34
Spanish	Cantonese	6
Spanish	Japanese	6
Spanish	Korean	11
Spanish	Mandarin	8
Spanish	Polish	16
Spanish	Russian	12
Spanish	Thai	8
Spanish	Vietnamese	10
Thai	Arabic	17
Thai	Cantonese	22
Thai	Japanese	5
Thai	Korean	13
Thai	Mandarin	8
Thai	Polish	2
Thai	Russian	3
Thai	Spanish	10
Thai	Vietnamese	10
Vietnamese	Arabic	24

Vietnamese	Cantonese	12
Vietnamese	Japanese	11
Vietnamese	Korean	10
Vietnamese	Mandarin	17
Vietnamese	Polish	5
Vietnamese	Russian	17
Vietnamese	Spanish	11
Vietnamese	Thai	30

5) Short Summary on Misclassifications:

Total data: 2000
Total predicted incorrect: 1245
Total predicted correct: 755

I have used several combinations of neurons and hidden layers using neural networks. I used a threshold value of 0.1 as each class has probability of 0.1(200/2000)

Below are the different neural network models:

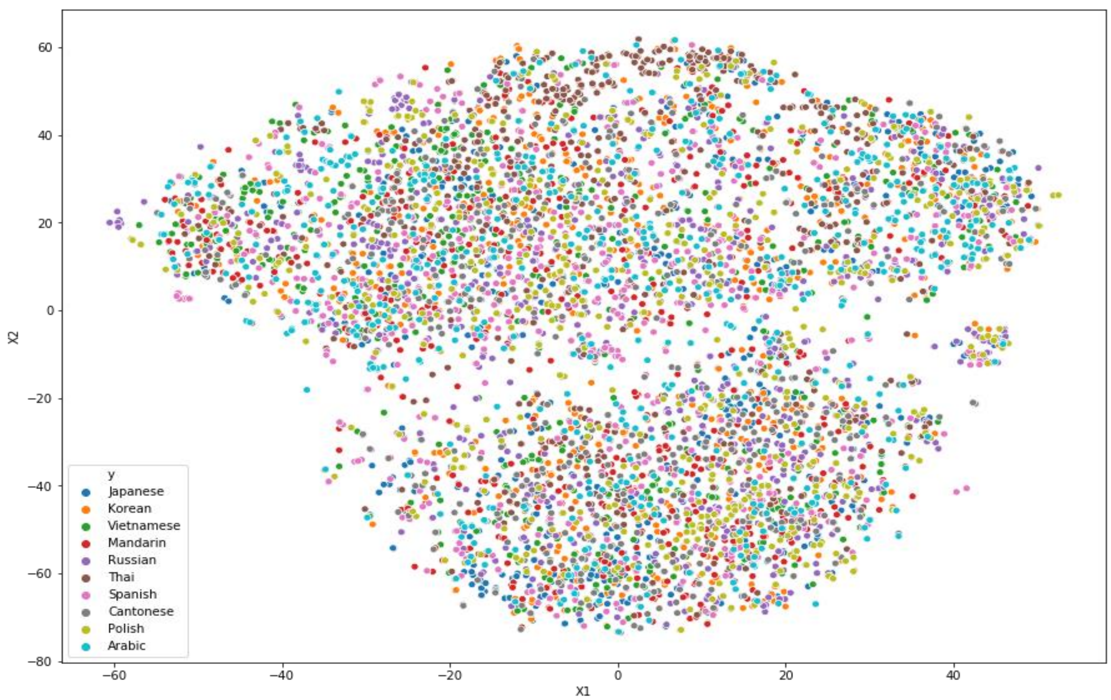
Activation	nLayer	nHiddenNeuron	nIter	Loss	Misclassification
logistic	50	80	6	2.302596	0.9
logistic	50	90	6	2.302598	0.9
logistic	50	100	6	2.302599	0.9
logistic	60	80	6	2.302599	0.9
logistic	60	90	6	2.3026	0.9
logistic	60	100	5	2.302602	0.9
logistic	70	80	6	2.302601	0.9
logistic	70	90	9	2.302603	0.9
logistic	70	100	6	2.302605	0.9
relu	50	80	7	2.302619	0.9
relu	50	90	8	2.302624	0.9
relu	50	100	7	2.302628	0.9
relu	60	80	8	2.302626	0.9
relu	60	90	7	2.302631	0.9
relu	60	100	7	2.302636	0.9
relu	70	80	9	2.302633	0.9
relu	70	90	7	2.302639	0.9
relu	70	100	7	2.302644	0.9
tanh	50	80	5002	1.584751	0.638
tanh	50	90	5001	0.893529	0.611
tanh	50	100	5001	0.024614	0.6175
tanh	60	80	5001	1.763256	0.7395
tanh	60	90	5001	1.896288	0.763
tanh	60	100	5001	0.638188	0.6655
tanh	70	80	5001	0.1399	0.6365
tanh	70	90	9	2.302623	0.902
tanh	70	100	5001	1.061502	0.6605
identity	50	80	5001	0.595219	0.588
identity	50	90	5002	1.19338	0.5195
identity	50	100	5001	1.240108	0.512
identity	60	80	5001	0.58828	0.5875
identity	60	90	5001	0.547011	0.569
identity	60	100	5001	0.710188	0.5635
identity	70	80	5001	0.742467	0.581
identity	70	90	5001	0.91574	0.5755
identity	70	100	5001	1.257233	0.523

From this table, we ca infer that the two models highlighted (tanh (50,100) and identity(50,90)) has lowest loss and lowest evaluation misclassification respectively. We can see that the logistic and ReLU models produces high misclassification as they get stuck in local minimum and hence the convergence is not enough for these models.

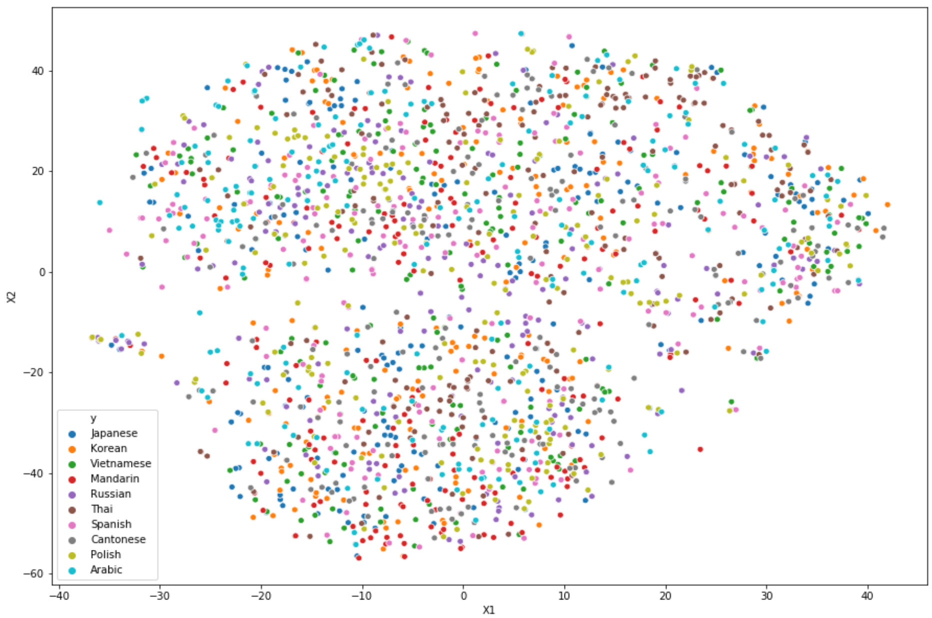
Furthermore, I also tried out t-SNE (t Stochastic Neighborhood Embedding) which is a technique to visualize higher dimensional objects into lower dimensions. It performs better than PCA in dimensionality reduction as t-SNE can solve the swiss roll problem.

The two-dimensional data for multiple classes can be shown as:

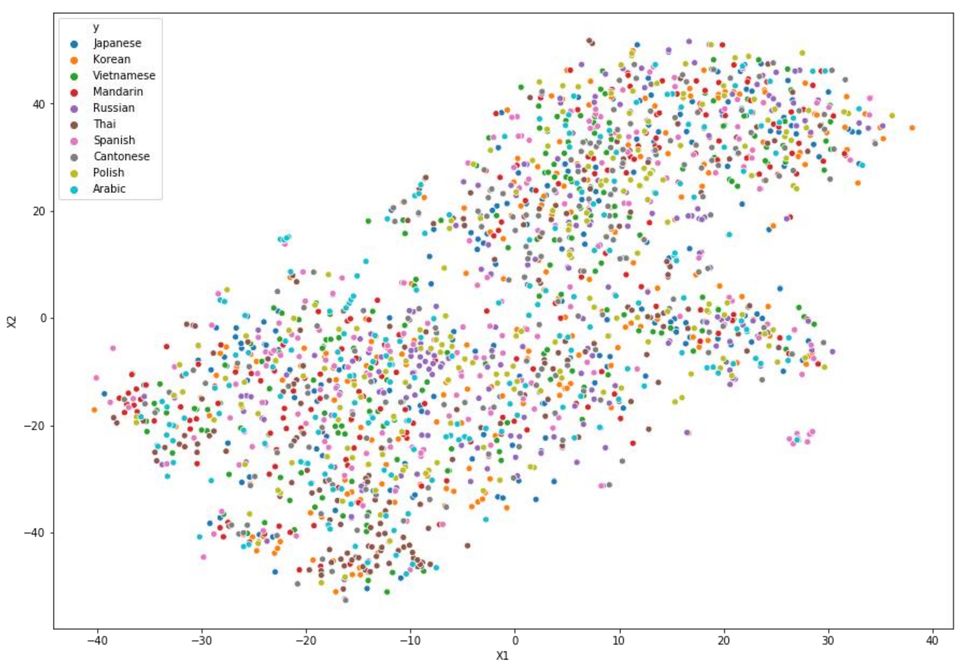
Training Data:



Evaluation Data:



Testing Data:



By using dimensionality reduction, we can reduce the model training time because it reduces the model complexity. However, in my case, it did not perform well because the training accuracy for logistic regression is 0.139. Hence, we do not utilize these results for further prediction.