DM2024 ISA2810 - Lab2 Homework

112034574 陳彥瑋

Contents

Introduction

- **The competition aim to understanding the emotions behind texts.**
- **It's important to devise a strategy that addresses various aspects of the task, like data cleaning etc.**
- **Twitter data often contains a lot of noise (e.g., URLs, @mentions, hashtags), and how to deal with these noise matters.**

Preprocess

- Data Cleaning

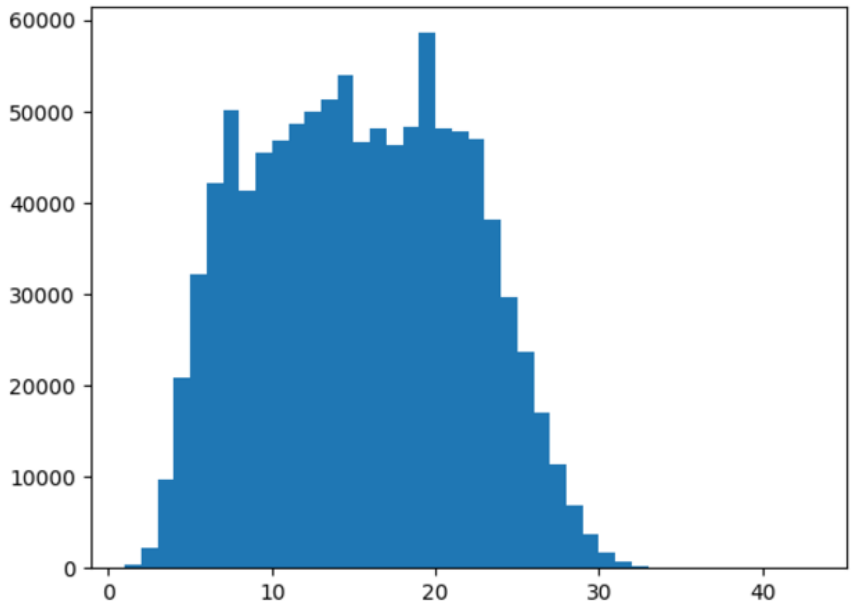
Combine all csv files into one dataframe according to tweet id.

- Feature Engineering

Use TF-IDF and Tokenizer to acquire the frequency of each word.
Pad the sequence of words with maxLen. (got from EDA)
Do one-hot encoder on emotion label.

- Other thing can do

Filter the noise in tweets, like emoji, urls, etc.
Ignore some common words in token, like prop..



The distribution of frequency

Training & Evaluations

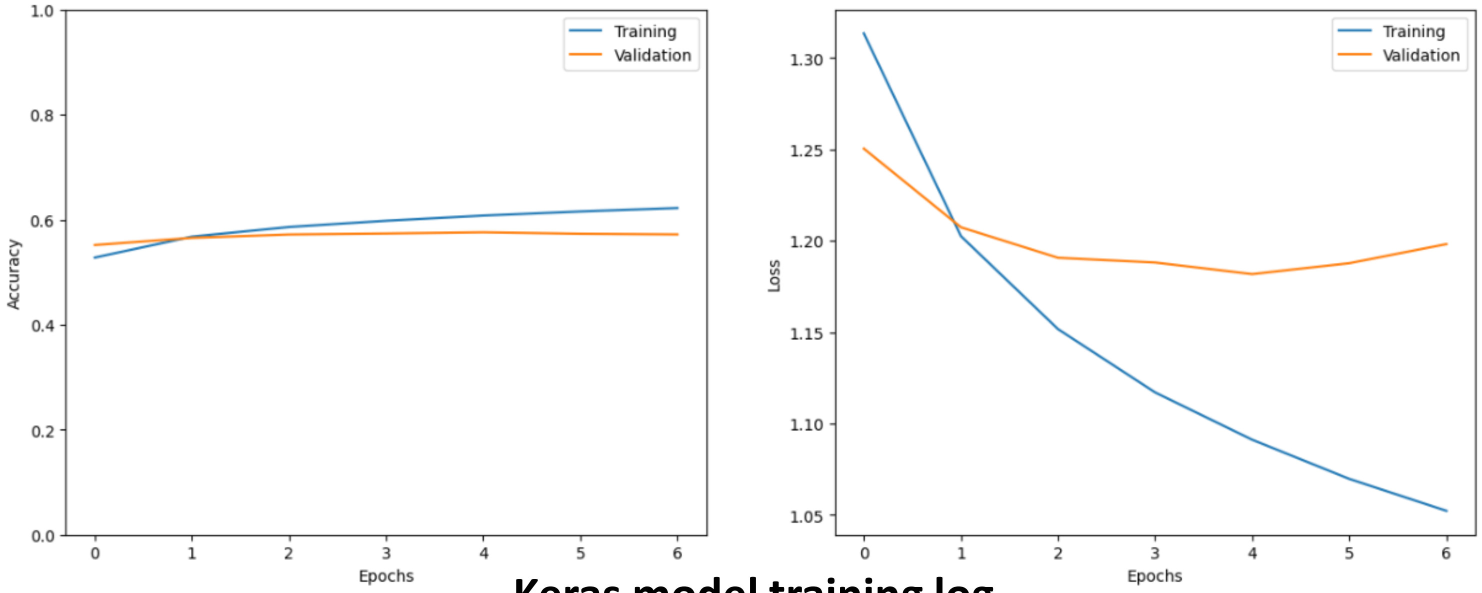
Choose DecisionTree and Keras as training model

- **DecisionTree**
- **Keras**

Decision tree validation result

	precision	recall	f1-score	support
anger	0.05	0.00	0.00	12085
anticipation	0.79	0.05	0.09	74497
disgust	0.10	0.96	0.18	41742
fear	0.37	0.04	0.07	19159
joy	0.76	0.03	0.06	154772
sadness	0.53	0.01	0.02	57923
surprise	0.24	0.00	0.01	14683
trust	0.26	0.09	0.14	61808
accuracy			0.13	436669
macro avg	0.39	0.15	0.07	436669
weighted avg	0.55	0.13	0.08	436669

The parameters setting goal is to tell which model is more suitable for this dataset. Therefore the number of layer is not really big.



Keras model training log

Model Parameters

```
model = Sequential()
model.add(Embedding(input_dim=len(tokenizer.word_index) + 1,
                    output_dim=32,  # Smaller embedding dimension for more efficiency
                    input_length=max_length))

# Global Average Pooling instead of Flatten to reduce parameter count
model.add(GlobalMaxPooling1D())

# Dense layer with fewer units for efficiency
model.add(Dense(units=32, activation="relu"))

# Dropout layer to prevent overfitting (optional, but recommended for large datasets)
model.add(Dropout(0.5))

# Output layer with softmax activation (for multi-class classification)
model.add(Dense(units=len(one_hot_labels[0]), activation="softmax"))

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.001), loss="categorical_crossentropy", metrics=["accuracy"])
early_stopping = EarlyStopping(monitor="val_accuracy", patience=2, restore_best_weights=True)
# Print model summary to see the architecture and number of parameters
model.summary()
model.fit(x_train, y_train, epochs=3, batch_size=32, validation_data=(x_validate, y_validate), callbacks=[early_stopping])
```

Future Works

- Try some more powerful models, like LSTM, BERT.

But the cost of try-and-error is an important factor.

- Enhance the feature engineering

Not only analysis the frequency feature, but some sentiment lexicons, like VADER.

- Adopt the model stacking technique

Through multiple result from different model, vote out the prediction.