Twitter US Airline Sentiment Analyzer using Machine Learning

Loïc XU
TDT4310 / 2024
loicx@stud.ntnu.no

Abstract

Passengers' sentiment of the airline during their flight has a considerable influence on their likelihood of remaining loyal to it. In an industry that faces aggressive competition to attract customers, it's important to please the customer. The former involves analyzing people's feelings and find the negative ones to take into account for the future and therefore improve the quality of the flight. To do so, I am going to try to classify sentiment of tweets about airlines with Machine Learning such as Multinomial Logistic Regression, Naive Bayes, Random Forest or Recurrent Neural Networks and compare it to the State-of-the-Art. The result showed that Multinomial Logistic Regression with TF-IDF and n-gram is the most accurate model and is quite good even though it can't beat the State-of-the-Art. It's interesting as Multinomial Logistic Regression is less expensive in terms of computation to train the model than most of the other models presented previously.

1 Introduction

The goal of this project is to perform sentiment analysis on tweets related to airline experiences. Basically, try to predict the sentiment felt by the author of the tweet about US Airlines, 3 possible cases : positive, neutral, negative. This idea came from the fact that recently, a door of Alaska Boeing 737 plane was ripped-off in mid-air and passengers immediately turned to Twitter to share their sentiment about it and I thought that airline companies may use social media to gauge the satisfaction of their customers. My methodology will consist of testing different features extracted from the tweets and Machine Learning models to achieve the best possible accuracy while minimizing calculation costs. Besides, I will compare the performance of each model.

2 Background

In this section, I am going to define and describe the complex terms that I will use in the rest of the sections.

2.1 Terms related to Natural Language Processing

Tokenization : process of breaking a piece of text into smaller units called tokens.
Stopwords : frequent words that we use in our sentences to link the words e.g. "a", "the".
Term Frequency (TF) : a measure of how frequent is a term in a document.

$$\text{TF}(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

Inverse Document Frequency (IDF) : inverse of how many documents in a collection of documents appears a word.

$$\text{IDF}(t, D) = \log\left(\frac{\text{Total number of documents in the collection } D}{\text{Number of documents containing term } t}\right)$$

TF-IDF (Term Frequency-Inverse Document Frequency) : a statistical measure used to evaluate the importance of a word in a document relative to a collection of documents.

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D)$$

N-gram : a continuous sequence of N words in a sentence e.g. 2-gram of "Twitter is the best social medium." would give us "Twitter is", "is the", "the best", "best social", "social medium".
Lemmatization : process of reducing words to their root called lemma e.g. "flying" to "fly".

2.2 Models

2.2.1 Multinomial Logistic Regression

Multinomial Logistic Regression (MLR) is a supervised machine learning algorithm that is an extension of binary Logistic Regression useful in multi classification tasks. It uses the softmax function to obtain the probability that the observation belongs to each class. In our case, we have 3 classes (negative, positive, neutral), so let's define for k between (0, 1, 2) :

$$z_k = \sum_{i=1}^{n} w_{ki} \cdot x_i + b_k$$

with :

- w_{ki} : the weight for the i-th feature in class k.
- x_i : the i-th feature.
- b_k : the bias term for class k.

The probability that the observation belongs to class k is :

$$P(y = k \,|\, x_1, x_2, ..., x_n) = \frac{e^{z_k}}{\sum_{j=1}^{3} e^{z_j}}$$

Traininig the Multinomial Logistic Regression model means to find the best set of parameters w_{ki} which accurately predict the correct class. It is possible by minimizing the loss function through gradient descent.

2.2.2 Naive Bayes

The Multinomial Naive Bayes model is based on Bayes' theorem, which describes the probability of a hypothesis given the observed evidence. In our case, the hypothesis represents the sentiment class y, and the evidence represents the observed features $(x_1, x_2, ..., x_n)$:

$$P(y = k|x_1, x_2, ..., x_n) = \frac{P(y = k) \cdot P(x_1, x_2, ..., x_n|y = k)}{P(x_1, x_2, ..., x_n)}$$

2.2.3 Random Forest

Random Forest is like a squad of Decision Trees teaming up for better predictions. Decision Trees recursively split data based on features using criteria like the maximization of Information Gain (quantifies uncertainty reduction from a split). Trees are built until stopping criteria, e.g., max depth. Random Forest randomly selects data and features for each tree. Final prediction average tree outputs.

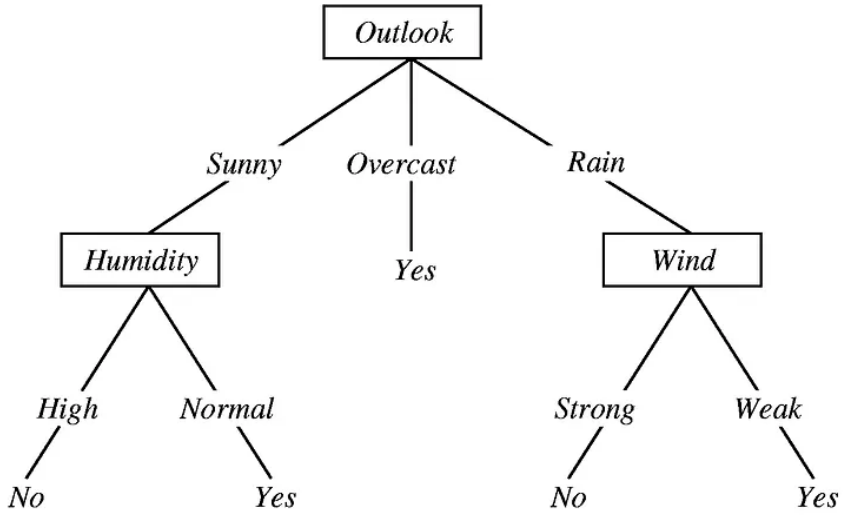

Figure 1: Example of Decision Tree from https://towardsdatascience.com/decision-tree-in-machine-learning-e380942a4c96

2.2.4 VADER

Valence Aware Dictionary and sEntiment Reasoner (VADER) is a rule-based sentiment analysis library that is specifically fine-tuned to sentiments expressed in social media. It has its own score the "compound score" (normalized sum of the valence scores of each word in the lexicon) between -1 for extreme negative and 1 for extreme positive. According to their scale : positive sentiment has a score greater than 0.05, neutral sentiment has a score between -0.05 and 0.05 and negative sentiment has a score lower than -0.05.

2.2.5 BERT

Bidirectional Encoder Representations from Transformers (BERT) is a State-of-the-Art pre-trained language model from Google that is both bidirectional and context-sensitive. In concrete terms, unlike unidirectional models, BERT will read a sequence of words in both directions, from left to right and from right to left, simultaneously. This makes it possible to capture more complex turns of phrase, much closer to human language. To do so, it uses a Transformer architecture. It therefore features the attention mechanism that enables it to make a contextual analysis of words. This makes it easier to handle ambiguities and polysemous terms It has derivatives such as :

- Robustly Optimized BERT Pretraining Approach (RoBERTa) : hyperparameters are different
- DistilBERT (Distilled version of BERT) : 40% less parameters so run faster and keep almost the same performance as BERT.

2.3 Performance metrics

For the performance metrics, I will use Accuracy, Precision, Recall and F1-Score :

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\_Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

with :

- TP = True Positives
- TN = True Negatives
- FP = False Positives
- FN = False Negatives

2.4 Tools

I will use several Python libraries to do feature extraction and use implemented models. For the Natural Language Processing part, I will use NLTK, SpaCy and emoji. And for the models part, I will use Scikit-learn, VADER, Transformers and TensorFlow.

3 Related Work

There are a lot of research papers on Sentiment Analysis on almost every subject. I focused on the ones about social media because it's more related to my project.
In (Cliche, 2017), the researcher tried to create a State-of-the-Art Twitter sentiment classifier using Deep Learning such as Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) models. More precisely, for the CNN, he used word embeddings and convolutional layers followed by max-pooling, dropout layers to avoid overfitting, fully connected layers and softmax layer for classification.
For the RNN, he stacked two LSTM layers to do a bidirectional LSTM and then used as previously dropout, fully connected and softmax layers. One LSTM reads the sentence forward, and the other LSTM reads it backward which gives more context as the model has access to the next words. The original ideas about his pre-precessing are the conversion of emojis into text and that he didn't removed stopwords. His work did achieve State-of-the-Art as he outperformed most of the previous best historical scores (before BERT's release).
In (George B. Alima et al. , 2022), they tried to use Machine Learning such as Naive Bayes, Linear Support Vector Classifier, Stochastic Gradient Descent Classifier and Logistic Regression to classify tweets as positive or negative. They found out that the best model for their dataset was Logistic Regression with an accuracy of 81%. They collected their own data from Twitter API.
In (Chiorrini Andrea, Claudia Diamantini, Alex Mircoli and Domenico Potena. "Emotion and sentiment analysis of tweets using BERT." EDBT/ICDT Workshops , 2021), they fine-tuned BERT to classify tweets between happiness, anger, sadness and fear and achieved an accuracy of 92% as BERT achieved State-of-the-Art performance across various NLP tasks.

4 Data

The dataset used in this project is Twitter US Airline Sentiment in 2015 from Kaggle containing tweets about various airlines, along with their associated sentiments (positive, negative, or neutral). This one stands out from the other on Kaggle as it is said to be human labelled which is important to avoid impurity in the dataset which will affect the performance of the models. The dataset contains 14640 entries and has 15 columns : (tweet_id, airline_sentiment, airline_sentiment_confidence, negativereason, negativereason_confidence, airline, airline_sentiment_gold, name, negativereason_gold retweet_count, text, tweet_coord, tweet_created, tweet_location, user_timezone). The ones that we are interested in are "airline _sentiment" and "text". I do not think that the other columns are relevant in our context as we do not want the model to overfit on for instance Twitter's username or airline name. As shown in Table 1, the dataset is unbalanced with a majority of negative tweets about airlines. I think it's pretty normal as people are most likely going to say something when it's bad then when it's good. When it's good, it's "normal" for the passenger and doesn't require to say anything about it. But when it's really really good, they are going to say it. It's called the negativity bias (tendency for humans to pay more attention to negative events or experiences compared to positive ones).

Table 1: Number of tweets for each sentiment

Negative	Positive	Neutral
9082	3069	2334

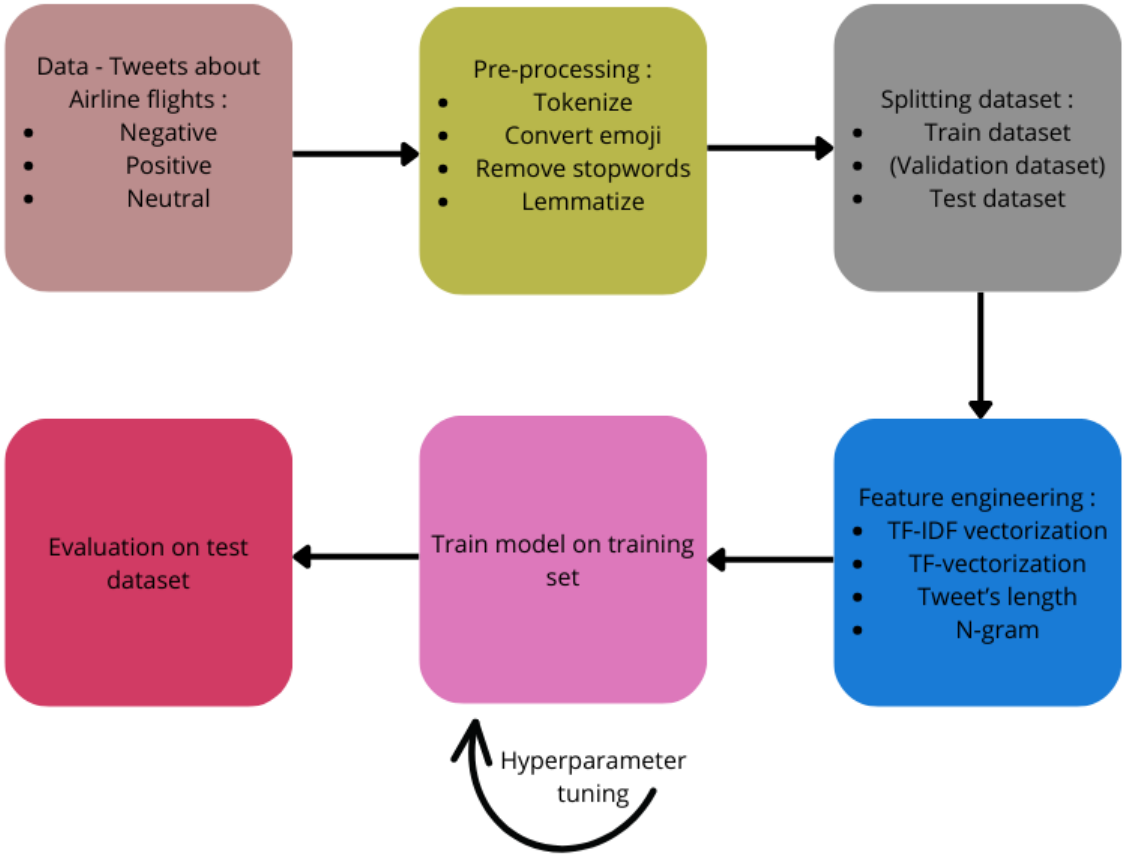

Figure 2: Pipeline for sentiment analysis

5 Architecture

Figure 2 shows the pipeline used in the project. I used different models such as : Logistic Regression, Random Forest, Naive Bayes, RNN with different combinations of features such as TF-IDF vectorization, TF-vectorization, tweet's length and N-gram.
For the State-of-the-Art, I used BERT and its derived versions. Since BERT is made to extract features in NLP, I had to add a classification layer as the output layer. Besides, BERT has its own pre-processing tool that I used. Therefore, as input it was only the raw tweet without any pre-processing.

6 Experiments and Results

In this section, I will explain the setup used in the project to reproduce the results such as pre-processing, hyperparameters tuning and also give the results of my different models.

6.1 Experimental Setup

For the **pre-processing** (Logistic Regression, Naive Bayes, Random Forest), I did :

- remove duplication of tweets thanks to unique "tweet_id"
- remove every columns except "text" and "airline_sentiment"
- convert emojis to text with demojize (emoji library)

- tokenize the tweets (NLTK library)
- remove stopwords and symbols as they don't contain information (NLTK library)
- remove every company's name as I believe it can lead to overfitting
- lemmatize tokens (SpaCy library because NLTK is not good enough if you do not precise the Parts-of-Speech tagging e.g. flying -> flying)
- transform sentiment into numerical values (negative 0, positive 1, neutral 2)

For the **feature engineering**, I chose to use :

- TF-IDF because it allows to find the words that are not frequent in all the tweets but still frequent in one specific tweet.
- Term Frequency as often people have the same lexicon in Twitter so they also may to use the same words to describe their sentiment.
- Tweet's length, during data analysis I found out that positive tweets are shorter then negative tweets. In average, negative tweets have 20 words for 114 characters and positive tweets have 14 words for 86 characters.
- N-gram because it gives more context than a single token.
- VADER compound score as it is fine-tuned on social media it may capture information that the other features can't.

For the **hyperparameters tuning**, I used GridSearchCV library to find the best set of hyperparameters for Logistic Regression and Random Forest with 5 stratified folds. So, you define the grid of parameters that you want to try and then it tries out all possible combinations of these hyperparameters using cross-validation (technique where the data is split into multiple parts, and the model is trained on all the part except one and tested on the only one. This helps prevent overfitting and gives a more reliable estimate of how well the model will perform on new data).
However, for BERT, I used a validation set to fine-tune the hyperparameters as cross-validation require a lot of computations for BERT.

It showed that the best hyperparameters are :

- for (LR TF-IDF, LR TF-IDF + tweet length, LR TF 2-gram, LR TF 2-gram + TF-IDF 4-gram + TF-IDF) : L2 regularization with C = 5.
- for LR TF 2-gram + TF-IDF 4-gram + TF-IDF + Vader score : L2 regularization with C = 0.5
- for the BERT models : learning rate = 1e-5 and batch size = 16
- for RF TF-IDF 2-gram : max depth = None, min samples leaf = 1, min samples split = 10, n-estimators = 100
- for (RF TF 2-gram + TF-IDF 4-gram + TF-IDF + tweet length, RF TF-IDF 2-gram, RF TF 2-gram + TF-IDF 4-gram + TF-IDF + VADER score) : max depth = None, min samples leaf = 1, min samples split = 2, n-estimators = 200

6.2 Experimental Results

The results are displayed in the Table 2 for Baseline models, Table 3 for Logistic Regression models, Table 4 for BERT models, Table 5 for Random Forest models. In the results displayed in the Table 3 and Table 5, "2-gram" means for instance that we used in range 1 to 2 for grams.

Table 2: Classification Report for the baseline models

Model	Class	Precision	Recall	F1-Score
Random Guess	0	0.65	0.34	0.45
	1	0.14	0.32	0.19
	2	0.21	0.34	0.26
	Accuracy on test set			**0.3379**
	Accuracy on train set			0.3416
VADER	0	0.90	0.43	0.58
	1	0.26	0.92	0.41
	2	0.38	0.33	0.35
	Accuracy on test set			**0.4911**
	Accuracy on train set			0.4811
Always predicting negative	0	0.65	1.00	0.78
	1	0	0	0
	2	0	0	0
	Accuracy on test set			**0.6454**
	Accuracy on train set			0.6223
Naive Bayes TF-IDF	0	0.72	0.99	0.83
	1	0.85	0.32	0.46
	2	0.73	0.23	0.35
	Accuracy on test set			**0.7314**
	Accuracy on train set			0.7714

7 Evaluation and Discussion

As the results shown in Table 2, the Random Guess model is not accurate with only 34% accuracy, but the most frequent class model works well with 65% accuracy. It highlights that a high number doesn't really mean anything. VADER is deceiving (49%) as it was fine-tuned for social media. The Naive Bayes TF-IDF is a good start with 73% but we can see that the model is lacking to generalize well as the F1-score of the positive and neutral class and are really low.

For the family of Logistic Regression (Table 3), we start to see an improvement and allows us to get to 82% accuracy as its peak with LR TF 2-gram + TF-IDF 4-gram + TF-IDF + VADER score. It generalizes better for the positive and neutral class. However, there is an overfitting as the train accuracy is at 99%. Moreover, a classic Logistic Regression with TF-IDF as a feature seems to be interesting with 78% accuracy without using too much computation, it reinforces the accuracy found by the study (George B. Alima et al. , 2022) with 81% even though it's not the same dataset.

We also see that using 2-gram as a feature helps to improve the accuracy. Besides, the feature tweet length didn't really work as expected, I guess it's because of the neutral tweets that prevent from discriminating the length of the positive and negative class.

For the Random Forest models (Table 5), there are a lot of overfitting and require too much computation for an average accuracy.

For the State-of-the-Art BERT models (Table 4), the best accuracy is for RoBERTa fine-tuned on the dataset with 85% accuracy. It shows a great generalization overall of the model to predict. A little bit deceiving as I expected an higher accuracy as said the study (Chiorrini Andrea, Claudia Diamantini, Alex Mircoli and Domenico Potena. "Emotion and sentiment analysis of tweets using BERT." EDBT/ICDT Workshops , 2021) with 91% accuracy but again it's not the same dataset.

Table 3: Classification Report for the best Logistic Regression models

Model	Class	Precision	Recall	F1-Score
LR TF-IDF	0	0.83	0.91	0.87
	1	0.74	0.63	0.68
	2	0.62	0.51	0.56
	Accuracy on test set			**0.7849**
	Accuracy on train set			0.9005
LR TF-IDF	0	0.84	0.90	0.87
+ tweet length	1	0.75	0.64	0.69
	2	0.62	0.53	0.57
	Accuracy on test set			**0.7880**
	Accuracy on train set			0.8992
LR TF 2-gram	0	0.86	0.90	0.88
	1	0.75	0.65	0.70
	2	0.64	0.61	0.62
	Accuracy on test set			**0.8046**
	Accuracy on train set			0.9981
LR TF 2-gram	0	0.87	0.91	0.89
+ TF-IDF 4-gram + TF-IDF	1	0.76	0.66	0.71
	2	0.65	0.62	0.63
	Accuracy on test set			**0.8101**
	Accuracy on train set			0.9984
LR TF 2-gram	0	0.87	0.92	0.89
+ TF-IDF 4-gram + TF-IDF	1	0.76	0.69	0.72
+ VADER score	2	0.68	0.61	0.64
	Accuracy on test set			**0.8194**
	Accuracy on train set			0.9909

For the LSTM, the accuracy is around 77% which is pretty deceiving for Deep Learning model, I think the issue comes from the fact that I decided to use TF-IDF as an input and not word embeddings like suggested in the study of (Cliche, 2017) and that I din't try enough different architecture's layers.

8 Conclusion and Future Work

To conclude, the best model is RoBERTa for the pre-trained model but not so far there is LR TF 2-gram + TD-IDF 4-gram + TD-IDF + VADER score with only 3% accuracy less. However, it costs less to compute with a Logistic Regression ! Random Forest and LSTM were pretty deceiving as it doesn't generalize well.

For future work, it should be interesting to check the part of Deep Learning about LSTM and CNN. Especially, using Word Embeddings such as GloVe or Word2Vec instead of TF-IDF as TF-IDF doesn't really represent the words. To improve performance, it is important to try to remove ambiguity around the words as we only use the Term Frequency that can't really take care of that. I believe that using RoBERTa as a feature extractor can help to find characteristics in the text and use it to improve the current high accuracy of RoBERTa on sentiment analysis. Another issue that should be tackled is the acronym used in tweets that most of the model can not understand.

Table 4: Classification Report for the BERT models

Model	Class	Precision	Recall	F1-Score
DistilBERT fine-tuned on SST2	0	0.77	0.90	0.83
	1	0.51	0.86	0.64
	2	0.00	0.00	0.00
	Accuracy on test set			**0.7079**
	Accuracy on train set			X
DistilBERT	0	0.73	0.02	0.04
	1	0.12	0.55	0.20
	2	0.22	0.34	0.27
	Accuracy on test set			**0.1615**
	Accuracy on train set			X
DistilBERT fine-tuned on the dataset	0	0.89	0.91	0.90
	1	0.77	0.81	0.79
	2	0.70	0.63	0.66
	Accuracy on test set			**0.8394**
	Accuracy on train set			*0.4591*
BERT fine-tuned on the dataset	0	0.91	0.92	0.91
	1	0.81	0.75	0.78
	2	0.67	0.69	0.68
	Accuracy on test set			**0.8453**
	Accuracy on train set			*0.4598*
RoBERTa fine-tuned on the dataset	0	0.93	0.89	0.91
	1	0.71	0.88	0.79
	2	0.71	0.68	0.70
	Accuracy on test set			**0.8463**
	Accuracy on train set			*0.4535*

References

Chiorrini Andrea, Claudia Diamantini, Alex Mircoli and Domenico Potena. "Emotion and sentiment analysis of tweets using BERT." EDBT/ICDT Workshops . Emotion and sentiment analysis of tweets using BERT. 2021.

Mathieu Cliche. BB_twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs. *Proceedings of the 11th International Workshop on Semantic EvaluationZ*, page 573–580, 2017.

George B. Alima et al. . Sentiment Analysis using Logistic Regression. *Journal of Computational Innovations and Engineering Applications*, pages 35–40, 2022.

Table 5: Classification Report for the best Random Forest models

Model	Class	Precision	Recall	F1-Score
RF TF-IDF 2-gram	0	0.78	0.94	0.85
	1	0.79	0.44	0.56
	2	0.63	0.43	0.51
	Accuracy on test set			**0.7604**
	Accuracy on train set			0.9985
RF TF-IDF	0	0.79	0.92	0.85
	1	0.72	0.54	0.62
	2	0.62	0.43	0.51
	Accuracy on test set			**0.7621**
	Accuracy on train set			0.9861
RF TF 2-gram + TF-IDF 4-gram + TF-IDF + tweet length	0	0.78	0.95	0.86
	1	0.81	0.45	0.58
	2	0.66	0.43	0.52
	Accuracy on test set			**0.7694**
	Accuracy on train set			0.9989
RF TF 2-gram + TF-IDF 4-gram + TF-IDF + VADER score	0	0.77	0.94	0.85
	1	0.78	0.46	0.57
	2	0.64	0.41	0.50
	Accuracy on test set			**0.7680**
	Accuracy on train set			0.9989