Gender classification by first name

Konstantin Lekomtsev
Data Science Career Track

Mentor: Nishant Gupta

Objective

The main objective of the project was to classify first names as male or female. Another important goal was to practice applying variety of classifiers and natural language processing techniques as well as deep learning methodologies, such as recurrent neural networks.

Dataset and Approach to modeling

The dataset contained 5000 first names. 2662 names were male and 2365 were female. The names and their gender were obtained by web scraping.

The following techniques were applied:

- Applied Naive Bayes classifier where the names' last letter, last two letters, and the first letter were features.
- Used term-frequency times inverse document-frequency (tf-idf) vectorization with 2-gram and 3-gram sequences to create features for support vector machine (SVM) classifier and multilayer perceptron (MLP) classifier.
- Used simple integer encoding and applied a recurrent neural network (RNN) with one bi-directional long short-term memory (LSTM) layer, and two bi-directional LSTM layers together with dropout layers.

Models training and evaluation

It was expected that Naïve Bayes classifier would perform well on the name's last letter or the last two letters. In many languages, the name's last few letters define a person's gender. To have an additional reference case Naïve Bayes classifier was also applied to the first letters. The results were consistent with the expectations: AUC score on the test data for the last letter was 0.85, it increased to 0.88 for the last two letters, for the first letter classification the result was close to random guessing with AUC on test data around 0.55.

Another approach was to consider relationships between letters and combinations of 2 and 3 letters within a name. It was possible to do by using tf-idf vectorization, a technique that puts more weight on the "important" letters or combinations of letters for gender classification. Tf-idf vectors were fed into support vector machine (SVM) and multilayer perceptron (MLP). Default SVM overfitted the data, with train AUC 0.99 and test AUC 0.82. The MLP with four dense layers initially overfitted the data, applying 3 dropout layers allowed to control overfitting, but as a

result, the classifier performance suffered (Fig. 1). The AUC score on the test data was 0.73.

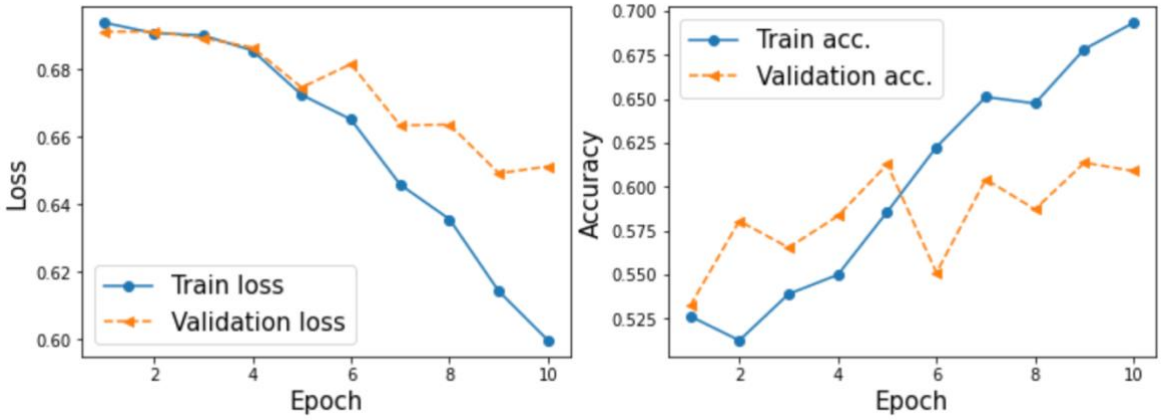

Fig. 1 Training of the MLP using tf-idf vectorization.

The final approach was to use RNNs with integer character encoding. Two RNN architectures were applied.

One LSTM network: RNN with an embedding layer, one bi-directional LSTM layer, and one dense output layer. The training-validation curves for this network are shown in Fig. 2.

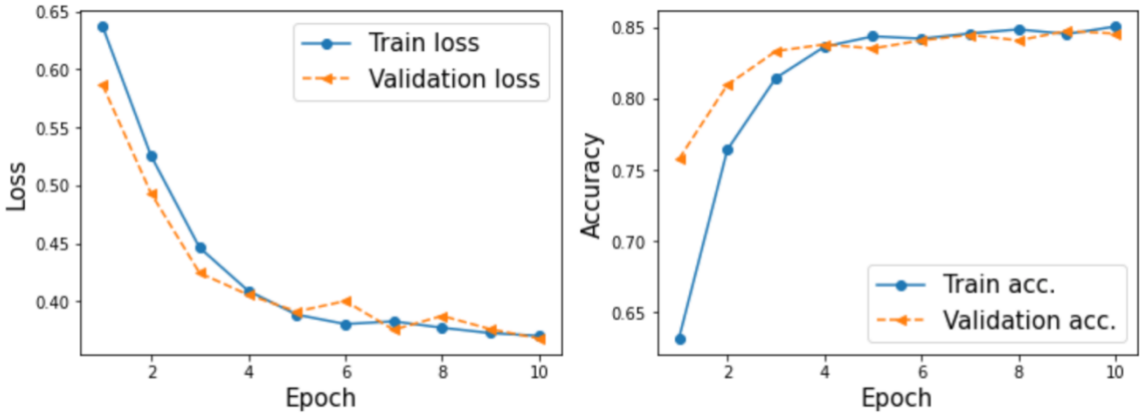

Fig. 2 Training of the RNN with one bi-directional LSTM.

Two LSTMs network with dropout layers:

-> Embedding layer
-> Dense layer
-> 1st dropout layer
-> 1st bi-directional LSTM
-> 2nd dropout layer
-> 2nd bi-directional LSTM
-> 3rd dropout layer
-> Output dense layer

The training-validation curves for the second network are shown in Fig. 3. For both architectures, AUC on the test data was 0.9. Despite a lot more complex architecture for the network with two LSTMs, the training time was just twice longer, probably due to the dataset being small.

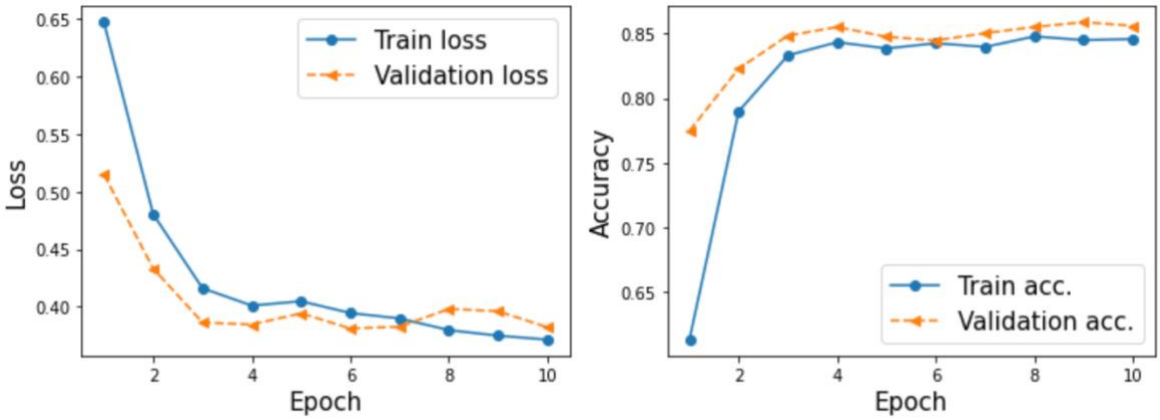

Fig. 3 Training of the RNN with 2 bi-directional LSTMs and dropout layers.

To look closer into the networks' performance, the classification reports for both architectures were compared (Table 1 and Table 2). The results on the test were very similar, however for the architecture with one LSTM precision and recall were more balanced.

1 LSTM	precision	recall
male	0.85	0.84
female	0.83	0.84
AUC (test)		0.9

Table 1. Classification report for the network with one LSTM.

2 LSTMs	precision	recall
male	0.88	0.81
female	0.81	0.87
AUC (test)		0.9

Table 2. Classification report for the network with two LSTMs and dropout layers.

Recall can be interpreted as: if I pick a random positive example, what is the probability of making the right prediction. Precision can be interpreted as: if I take a positive prediction example, what is the probability that it is indeed a positive example.

Conclusions

Naïve Bayes classifiers on the first, last and last two letters of the first names were applied. Intuitively the last two letters are good predictors of a person's gender, which was confirmed by the classifiers. SVM and MLP

classifiers were applied on tf-idf vectors that took into account the importance of a certain character or combinations of characters for gender prediction. Finally, RNNs with bi-directional LSTMs were applied to take into account relationships between letters or combinations of letters, when "reading" a name from left to right and in reverse, right to left.

An overview of the classifiers' performance is shown in Table 3.

Classifier	AUC on test data
Naïve Bayes on last letter	0.85
Naïve Bayes on last two letters	0.88
Naïve Bayes on first letter	0.55
SVM, tf-idf vectors	0.82
MLP, tf-idf vectors	0.73
1 LSTM, integer encoding	0.9
2 LSTMs + dropouts, integer encoding	0.9

Table 3. Classifiers' AUC score on the test data.

To summarize:

- The top performers were RNNs with one and two bi-directional LSTMs.
- Naive Bayes classifier applied to the last two letters was the second-best based on the AUC metric, however, its computation time was significantly shorter compared to RNNs.