Predicting Gender Based on First Names

Anais Corona Perez, Jason Lee, Gyuho Shim, S. Sudhir, Yuki Zhou

August 9, 2023

Abstract

This paper is intended to explain and report the supervised machine learning models used to predict gender on the basis of first name. The data includes 25,595 names with labels `male`, `female`, and `uni-sex` respectively. Feature engineering was implemented to convert the names in the dataset with special characters to ASCII equivalent, and reduce the dataset to only account for binary targets `male` and `female`. One-hot encoding was used to embed these names into 1-dimensional vectors. Finally, after splitting the dataset into training, testing, and validating sets, we used grid-search and gradient-boosting to evaluate and find the best model.

1 About the Dataset

The website `behind the name` allows users to find the history and etymological meaning of names in various languages. We used the `given names + gender` dataset to train our model, which can be found here.

	Name	Gender
0	Aabraham	m
1	Aada	f
2	Aadan	m
3	Aadolf	m
4	Aafje	f
...	...	...
25590	Žydrė	f
25591	Žydrūnas	m
25592	Zygfryd	m
25593	Zygmunt	m
25594	Zyta	f

25595 rows × 2 columns

Figure 1: Snapshot of dataframe with column "Names" and associated "Gender".

2 Feature Engineering

We first removed all rows with labels `uni-sex`, as we were only concerned with binary targets `male` and `female`, which we labelled `0` and `1` respectively. Then, we mapped each name with ISO Latin 1 characters to the equivalent ASCII characters using the unidecode package (further details can be found here). This left us with 24,595 names to be used to train our data.

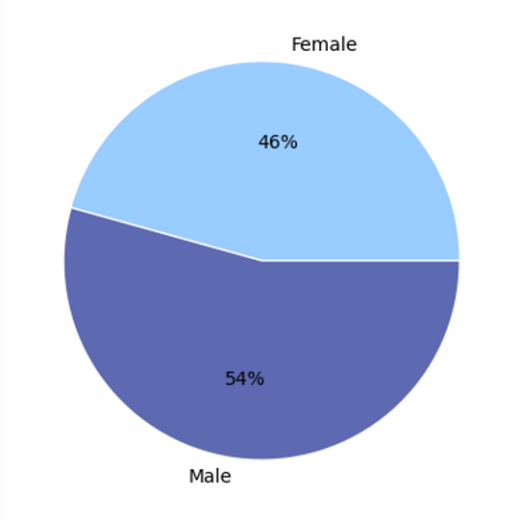


Figure 2: Proportions of male and female names post feature engineering.

3 One-Hot Encoding Names

Consider a 62 length vector of the following form:

$$\Big(\text{a, b, ..., z} \quad | \quad \text{A, B, ..., Z} \quad | \text{ ', , AE, ae, Th, th, @, -, *, ‘}\Big)$$

Each letter of a name can be one-hot encoded in the vector above. In order to make our one-hot encoding sci-kit usable, we pad enough rows with zeros so that the total number of rows is equals the length of the largest name in the

dataset. In our data, the largest name is 21 characters long, i.e., each name in our dataset is encoded as a (21×62) 2-dimensional array, which after flattening gives us a 1-dimensional vector of size 1302. An interactive-3d PCA projection can be found here.

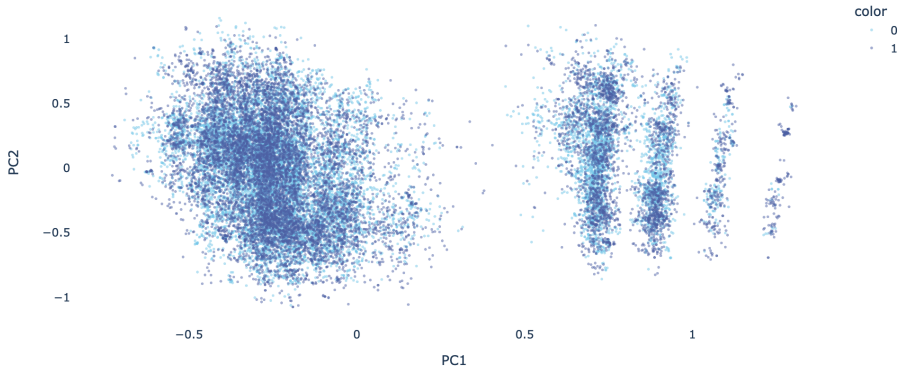


Figure 3: 2-dimensional PCA projection of one-hot encoded names

4 Algorithm to Find the Best Classifier

We used `GridSearchCV` with multicore `n_jobs = 5` to find the best hyper-parameters for binary classification using Knn and Logistic Regression. We found that `metric = cosine` and `n_neighbors = 3` were the best hyper-parameters for knn, and `C = 360` was the best hyper-parameter for logistic regression. Comparing validation accuracy's for both those models, along with decision tree classifier with `criterion = entropy` we found that decision tree classifier had the best performance.

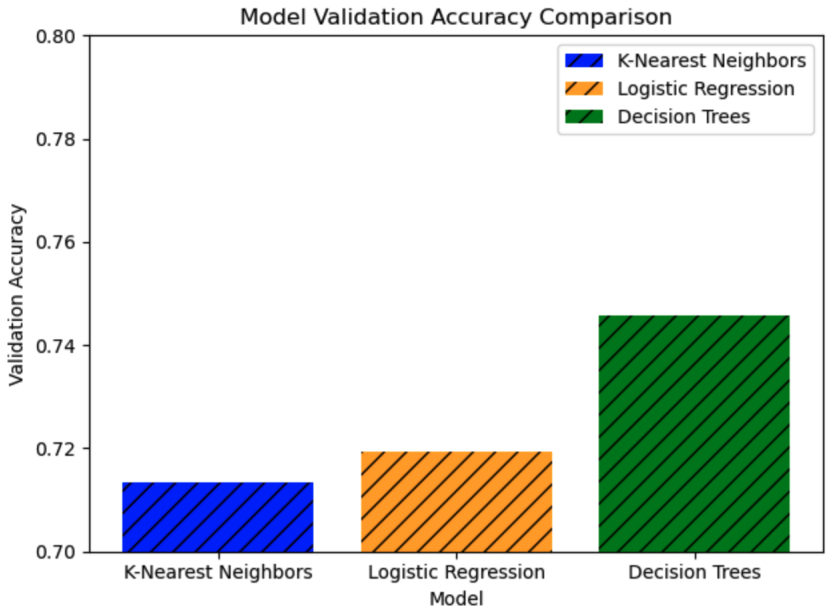


Figure 4: Comparing validation accuracy for Knn, Logistic Regression, and Decision Trees

5 Gradient Boosting

As the validation accuracy for all our models is ranged from 0.71 to 0.74, we decided to fit an ensemble classifier model to maximize accuracy. As decision trees had the highest validation accuracy, we decided to go for the Gradient Boosting classifying model as it is an ensemble model that fits boosted decision trees by minimizing error gradient. After brute-forcing, we found that `n_estimators = 10000` and `learning_rate = 1.0` had the best validation accuracy at about 0.794, which was considerably better than the previous three models.

6 Model Performance Assessment

After plotting Accuracy, Precision, Recall, and Area under the ROC curve, we can see that only Decision Trees and Gradient Boosting models have all scores over the threshold 0.7. Additionally, Gradient Boosting out performs every model by a decent margin, thus making it the final model that we choose for predicting gender based on names.

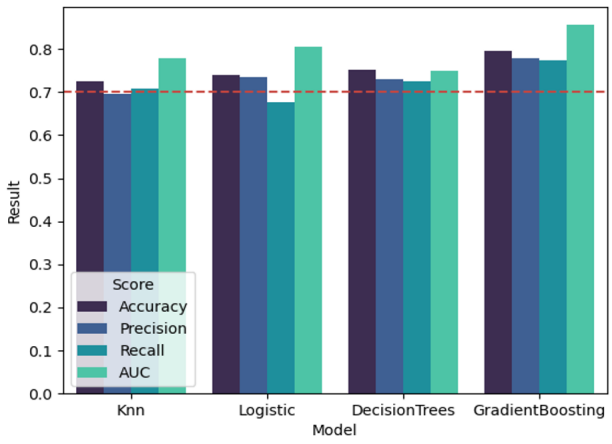
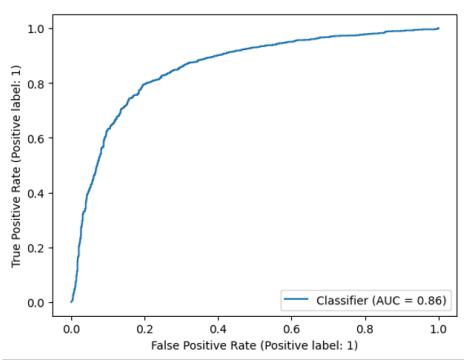
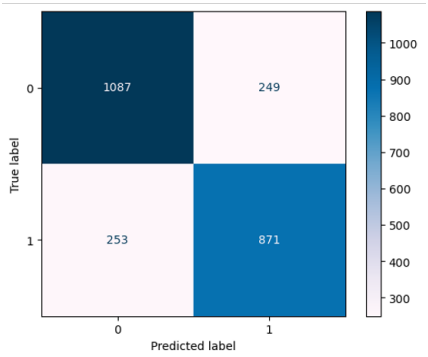


Figure 5: Comparing performance assessment between all models

7 Conclusion

In order to predict gender based on names, we used the `behind the name` data-set, conducted rigorous feature engineering, one-hot encoding, and splitting to make the data-set usable in `scikit`. After running a grid search and plotting the Knn, Logistic Regression, and Decision Trees with the most optimal hyper-parameters, we found that Decision trees had the best validation performance. Following this, we used gradient boosting ensemble classifier with 10,000 boosting stages and learning rate 1.0 to fit the best model that outperforms the previous three models by a considerable margin.





Names	Predicted Gender	Actual Gender
Shrivats	0	0
Anais	1	1
Gyuho	0	0
Jason	0	0
Yuqi	0	0

Figure 6: Assessment Figures of Gradient Boosting and Predicting Gender of Contributors of this report

8 References and Appendix

Campbell, Mike. "The Meaning And History Of First Names - Behind The Name". Behindthename.Com, 2023, https://www.behindthename.com/.

Gupta Sukrat, Kanchinadam Teja, Conathan Devin, Fung Glenn (2020). "Task-Optimized Word Embeddings for Text Classification Representations." *Frontiers in Applied Mathematics and Statistics.* Volume 5. doi: 10.3389/fams.2019.00067. ISSN: 2297-4687. https://www.frontiersin.org/articles/10.3389/fams.2019.00067.

Junting Ye, Steven Skiena (2019). "The Secret Lives of Names? Name Embeddings from Social Media." *arXiv.* eprint: 1905.04799. https://arxiv.org/pdf/1905.04799.pdf

Natekin A, Knoll A (2013). "Gradient boosting machines, a tutorial." *Front Neurorobot.* doi: 10.3389/fnbot.2013.00021. PMID: 24409142; PMCID: PMC3885826.

"NLP From Scratch: Classifying Names With A Character-Level RNN". Pytorch, 2023, https://pytorch.org/tutorials/intermediate/char_rnn_classification_tutorial.html.

Wong KO, Zaïane OR, Davis FG, Yasui Y (2020). "A machine learning approach to predict ethnicity using personal name and census location in Canada." PLOS ONE 15(11): e0241239. https://doi.org/10.1371/journal.pone.0241239.

Note: We believe in open-source information and knowledge. All relevant codes and materials can be found in this GitHub repository: https://github.com/Stochastic1017/Predicting_Gender.git