PetFinder Challenge: Predicting Pet Adoption Speed

Kaylee Zhang
Department of Statistics
Stanford University
kayleez@stanford.edu

Sherine Zhang
Department of Computer Science
Stanford University
sherinez@stanford.edu

1 Introduction

Many stray animals throughout the world unfortunately do not get the opportunity to find the loving home that they deserve. In this classification task, we look to develop an algorithm to predict the speed of pet adoptions. The input to our algorithm includes animal type (dog or cat), breed, gender, color, profile picture online and the descriptions about the pet, etc. We then use both traditional machine learning techniques (Logistic Regression, Naive Bayes, SVM, Decision Tree, Random Forest and Gradient Boosting) and neural networks (fully connected neural networks and long short-term memory Model) to predict the adoption rate. In particular, we build feature vectors using information extracted from description scripts and feed them into neural network models. We are hoping to examine the results to develop strategies to help improve the overall adoption rate (i.e. what features lead to faster adoption). The methods are described briefly in Section 4 and experiments are included in Section 5. Result and performance of these models are compared in Section 6.

2 Related Works

In contrast of general classification task, the four classes of adoption speed in our dataset retains ordinal relationships. Frank and Hall argue that standard approach may fail to consider the ordering information and suggest decision tree approach [1]. We therefore explore decision tree methods, as well as deep learning techniques. In the original Kaggle challenge, the evaluation metric is quadratic weighted kappa, which measures the agreement between two ratings, and it takes error cost into account as in [2]. Considering the non-linear time intervals of different classes, however, we choose classification accuracy as our evaluation metric.

Word2Vec [3] and GloVe [4] are two word embedding models that compute vector representations of words. The goal is to embed semantically similar words into similar vector space. Word2Vec is a prediction-based method, trained on a large dataset with rather low computational requirement. It outperforms many traditional methods. GloVe, on the other hand, is a count-based method. In the paper [4], Pennington et al argue that both methods share fundamental ideas and capture similar sementic information, but GloVe is more efficient in performing the task. Thus we use GloVe as the embedding model to obtain embedding vectors.

A recent work done by Mirsamadi et al [5] on speech emotion recognition suggests that using a simple local attention scheme helps with identifying emotionally salient words. Based on this attention scheme, they then propose a new method that enables the model to put more weights on the learned words in order to achieve better classification accuracy. Inspired by this work, we incorporate a local attention layer in the model in the hope of capturing emotionally important information from the text inputs.

Conference on Neural Information Processing Systems (NIPS), .

3 Dataset and Features

3.1 Dataset

The dataset is provided by PetFinder and Kaggle. It consists of three parts. The first is a CSV file with detailed information including animal type (indicator for dog or cat), breed, gender, color, fur length, sterilized, health, and description. The second is JSON files of descriptions with sentiment score, and the third is a large collection of image files. In total, we have 14,993 pet records and 23 features. We drop a few non-essential variables such as PetID and RescuerID. Image files are matched with every pet entry but each image can include more than one pet. This creates an additional challenge of identifying the specific pet corresponding to the record. The response variable, adoption rate, is comprised of five categories:

0 - same day as listed
1 - between 1 and 7 days (1st week)
2 - between 8 and 30 days
3 - between 31 and 90 days (2nd and 3rd month)
4 - No adoption after 100 days of being listed

Note that there are no pets in this dataset that waited between 90 and 100 days. Also, the time duration of each category is not the same. For example, class 0 includes a 1-day time interval while class 3 includes a 60-day time interval.

3.2 Features

The dataset includes many categorical variables and we convert them into one-hot-encoding variables so that we can feed those into machine learning models and neural networks. We dropped the pet name in our models. Taking account of the number of levels in Breed variable, we only consider the primary breed type, which already introduced over 300 one-hot-encoding features.

We use GloVe [4] to obtain feature embeddings for the LSTM model (discussed later in the Experiment section). For any missing word, we append <unk> token and its corresponding embedding vector to the input data. We then pad zeros to the end of each vector to match the longest description length in the entire dataset so that the input length is consistent across all examples.

4 Methods

The first step was randomly splitting the data into training, validation and testing sets (72%, 18% and 10% respectively). We trained the models on the training set, then adjusted the hyperparameters and regulation strengths based on the results from validation set. We then experimented with deep learning models and fit on the test set to get the test accuracy score. We only did this once, after we constructed the final model to avoid overfitting to the test set.

4.1 Logistic Regression

We used multi-class logistic regression to model the 5-class response variable. The conditional distribution of the response given the input and parameters is given by

$$\Pr(Y_i = c|\mathbf{X}; \boldsymbol{\beta}) = \frac{e^{\boldsymbol{\beta}_c \cdot \mathbf{X}_i}}{\sum_{k=1}^{K} e^{\boldsymbol{\beta}_k \cdot \mathbf{X}_i}} \tag{1}$$

Multi-class logistic regression is an extension of the binary two-class case. The model predicts the probability of assigning examples to every class. Based on the confusion matrix result, we found that the model falsely classified many class 2 and class 3 data as class 5. Logistic regression particularly did not do a good job of predicting class 0 or class 3. In fact, the model did not predict class 0 at all. We added regularization to the model by penalizing the size of $\boldsymbol{\beta}$. L1 penalty was chosen over L2 penalty since it yielded a better result on the validation set.

4.2 Naive Bayes and Support Vector Machines

We wanted to explore a variety of classification algorithms to see what works well for our task. Gaussian Naive Bayes is often a very good initial algorithm to try and SVMs are among the best "off-the-shelf" supervised learning algorithms, per the lecture notes. However, in our task, Naive Bayes performed the worst in terms of validation set accuracy score.

After fitting a C-support SVM model, and tuning the hyperparameters, we found that a penalty parameter (C) of 1.0, and the radial basis function kernel performed best on the validation set. SVMs performed better than all other non-ensemble methods.

4.3 Decision Trees

The decision tree, or classification tree method, at each step chooses the variable which makes the best split and then decides the split point. In terms of best splits, we interpret it as the split that best reduces the risk (error) and improves the purity of leaf nodes [6]. For trees, many class 5 examples were misclassified as class 3. Decision trees allow us to calculate variable importance [6]. Age, breed, and color were the most important feaures but health and animal type were not.

4.4 Random Forest and Gradient Boosting

Random forest and gradient boosting algorithm are ensemble methods to improve the performance of decision trees (by achieving higher predictive accuracy). Trees have high variance and are very unstable. Random forests reduce variance by performing bootstrap aggregation (bagging) by building a large collection of de-correlated trees and then averaging them [7]. Boosting combines weak classifiers (trees) and produces a more powerful model. The function space is much larger than that of a single tree and the boosting function is a lot smoother [8]. The reader may refer to [7] for more information.

4.5 Neural Networks

Neural Networks are a group of algorithms inspired by the structure of human brain, using neurons as the basic unit. They are used in various fields to solve problems such as clustering, classification and regression. For the purpose of this project, we introduce two types of networks, fully connected and long short-term memory (LSTM) nets. A fully connected network consists of multiple fully connected layers, i.e., each neuron in the current layer is connected to each neuron in the previous layer, and each link between two neurons has a weight to itself. Because of this, fully connected nets are expensive, but they are general purpose networks which are suitable for common usage. LSTM [9] is a type of Recurrent Neural Network (RNN) consisting cells and gates. The gates are used to decide which and how much information to pass on to the next cell. The cell takes into consideration both previous and new information to give the final output. This structure of LSTM makes it possible for the net to handle sequential data well.

5 Experiments

5.1 Fully Connected Network

The constructed Fully Connected network consists of three groups of layers, as shown on the bottom left in Figure 1. There are in total 3 sets of hyperparameters that we could tune, namely, learning rate, batch size, and the output sizes of the intermediate fully connected layers. We observed the behavior of the validation set under different combinations of hyperparameters, and chose the combination that gave the best validation accuracy as our final parameters. Since we used Adam optimizer, we also changed the weight decay parameter along with the learning rate. For the best-performing model, we used a learning rate of 10^{-4}, a weight decay of 10^{-5}, a batch size of 200, and output dimension of 32, 16, 8 for each layer respectively. We used Dropout layer to mitigate the effect of overfitting.

5.2 LSTM Network

The LSTM model is consisted of a linear encoding layer, an LSTM layer, a local attention layer and a linear decoding layer. In addition to the learning rate, weight decay, batch size, we also tried different hidden layer dimensions. The final set of hyperparameters in use is the same as that in the Fully Connected Network except a hidden layer size of 128.

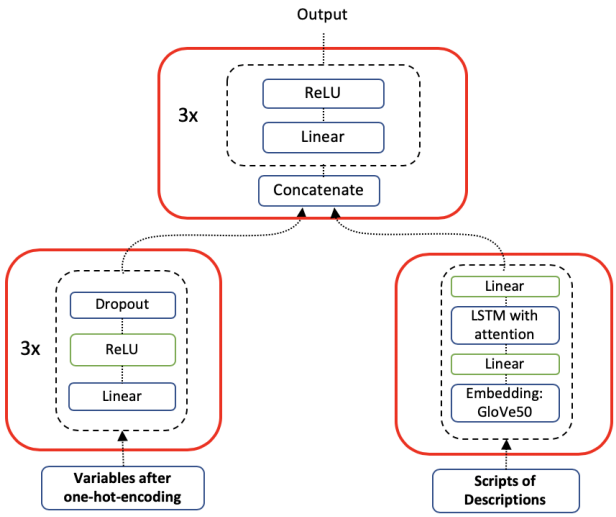

Figure 1: The combined model pipeline. Bottom left shows the structure of the Fully Connected network. Bottom right shows the structure of the LSTM model.

5.3 Combined Model

After obtaining results from both the Fully Connected and LSTM model, we merged them into a single model to see if this boosts classification accuracy. We first extracted the second last layer output from both models, concatenated them into a single vector and fed into another set of linear layers to get the final prediction. Then we tried to take full advantage of both models and simply use the concatenation of their final output as the input to the linear layers. Result from the final layer concatenation is slightly better than intermediate layer extraction but they do not differ much, so we only included the better one in our result table.

5.4 Evaluation

We use accuracy as the evaluation metric for the classification task.

6 Results and Discussion

Among machine learning models, Random Forest and Gradient Boosting give the best accuracy. As we can see from the table, the performance of the ensemble methods exceed decision trees because of the aforementioned reduction of variance. The table also suggests that our models are insufficient for the prediction task as they do not incorporate the information from the profile images or the animal descriptions (text data). Without visual information extracted from animal photos and textual information extracted from the descriptions, the models are unable to predict the adoption rate accurately.

Among deep learning models, Fully Connected model with only categorical data inputs performs the best. It also has a slightly higher accuracy than the best-performing machine learning model under the same inputs. LSTM model with text inputs has relatively low accuracy and does not show a learning trend. Reasons might include the large variation of description length in the dataset and the informality of the texts, which introduce many unknown words to the GloVe pretrained model.

		Statistics		
		Precision	F1-score	Accuracy
Machine Learning Models	Logistic Regression	0.32	0.31	0.335
	Naïve Bayes	0.33	0.29	0.311
	SVM	0.34	0.33	0.359
	Decision Tree	0.33	0.32	0.321
	Random Forest	0.38	0.38	0.392
	Gradient Boosting	0.37	0.36	0.385
		Cross Entropy Loss	Evaluation Accuracy	Test Accuracy
Deep Learning Models	Fully Connected	0.00101	0.393	0.396
	LSTM	0.00097	0.294	0.322
	Combined	0.00091	0.383	0.384

Figure 2: Result table that lists all models and their performance on the classification task.

As shown in the left plot of Figure 3, evaluation accuracy increases as epoch number increases. The Fully Connected model converges the fastest among the three.

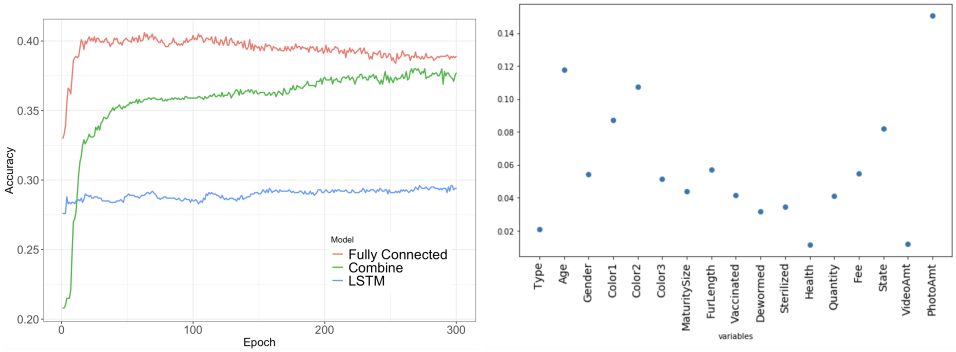

Figure 3: The left plot shows the evaluation accuracy of three neural network models. The right plots shows the variable importance score using decision tree

Though our response variable, adoption rate, is categorical, it has an ordinal relationship. Falsely assigning class 0 to class 5 should generate a greater penalty or cost than assigning it to class 2. By examining the confusion matrices, we found that Logistic Regression and SVM never classify input examples as class 0, mainly because class 0 only occupy a small proportion of total dataset as showed in Figure 2. Thus, simple models always favor other classes to achieve better overall accuracy.

The right plot shows the variable importance score using decision tree method. It is calculated by the reduction of errors and increase of node purity introduced by splits using this variable. The importance of the amount of photos confirmed the use of visual input would help boost the result.

7 Conclusion and Future Work

The task for this project is to predict pet adoption speed given related information such as biological details, descriptions. We use traditional machine learning techniques such as logistic regression, Naive Bayes, SVM, decision trees, random forest and gradient boosting on the categorical data, as well as deep learning algorithms on the combination of categorical and text data. Among all models, the Fully Connected model on only the categorical data achieves the highest accuracy of 0.396 on the test set. Based on the result, visual information such as photos and videos of the pet seems to have the most influence on the adoption decision.

For future work, we plan to experiment with more machine learning models, and try different ensemble methods in the hope of an increase in accuracy. Given the importance of visuals and the available image data, we would like to take full advantage of the dataset and incorporate these images into our deep learning models using computer vision algorithms. Different model structures and hyperparameter combinations can be tested and adjusted accordingly if more time and computational resource are available.

8 Contributions

Carson Zhao (czhao333@stanford.edu), Kaylee Zhang and Sherine Zhang contributed equally for the milestone. Carson switched his group for final project. For this project, two of us contributed equally and together after milestone. We both did general project brainstorming, fitting machine learning models and neural networks to the data, and writing the final project.

References

[1] Eibe Frank and Mark Hall. A simple approach to ordinal classification. In *European Conference on Machine Learning*, pages 145–156. Springer, 2001.

[2] Arie Ben-David. Comparison of classification accuracy using cohen's weighted kappa. *Expert Systems with Applications*, 34(2):825–832, 2008.

[3] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[4] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[5] Seyedmahdad Mirsamadi, Emad Barsoum, and Cha Zhang. Automatic speech emotion recognition using recurrent neural networks with local attention. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2227–2231. IEEE, 2017.

[6] Leo Breiman. *Classification and regression trees*. Routledge, 2017.

[7] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[8] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.