**CISC 452 Pet Adoption Prediction  
Final Report**

**Garett MacGowan (10197107)  
Areege Chaudhary (10197607)**

**Problem and Motivation**

There are millions of stray animals all around the world. To reduce animal suffering, PetFinder.my, Malaysia’s leading animal welfare platform would like to be able to predict animal adoption rates. Predicting animal adoption rates may allow the platform to optimize the profiles of pets to increase their adoptability. Additionally, it may help in creating projections for the stray animal population as well as animal welfare costs.

**The Data**

PetFinder.my has collected a database of over 150,000 animals that contains images, descriptions, animal attributes such as health, breed, color, etc., administrative attributes such as rescuer id’s, and more general information such as animal location. The dataset which is provided for creating the model contains approximately 15,000 entries in the training set and approximately 4,000 entries in the testing set. Since the data is part of a Kaggle.com competition, the testing set is not labeled. For this reason, the training set provided needs to be split into a training and testing set so that we can evaluate our model.

As a group, we decided to use a 90%/10% split for the training and testing set. To generate sets with good properties, we used stratified sampling. We chose to do this because the distribution of the classes is very skewed. Stratified sampling forces the distribution of the training and testing sets to be the same as the original total set.

The data is available at <https://www.kaggle.com/c/petfinder-adoption-prediction/data> or through GitHub at <https://github.com/Garett-MacGowan/Pet-Adoption-Predictor>.

|  |  |
| --- | --- |
| Data | |
| Adoption Speed | The value we are trying to predict. Ordinal speed of adoption. Lower is faster. 0 means the pet was adopted on the same day it was listed. 1 means the pet was adopted between 1 and 7 days after being listed. 2, pet was adopted between 8 and 30 days. 3, pet was adopted between 31 and 90 days. 4, no adoption after 100 days of being listed. |
| Pet ID | Used to find a sample’s associated images, image metadata, and description sentiment. |
| Type | Whether the pet is a cat or a dog. |
| Name | The name of the pet, if named. |
| Age | The age of the pet when it was listed, in months. |
| Breed 1 | Primary breed of the pet. There are 307 different breeds which can be decoded using the breed labels dictionary file. |
| Breed 2 | The secondary breed of the pet, if the pet is a mixed breed. |
| Gender | The gender of the pet, 1 = male, 2 = female, and 3 = mixed, if the profile is representative of a group of pets. |
| Color 1 | The primary color of the pet. There are 7 different color labels which can be decoded using the color labels dictionary file. |
| Color 2 | The secondary color of the pet. |
| Color 3 | The tertiary color of the pet. |
| Maturity Size | The size of the pet at maturity. 1 = small, 2 = medium, 3 = large, 4 = extra large, 0 = not specified. |
| Fur Length | The fur length of the pet. 1 = short, 2 = medium, 3 = long, 0 = unspecified. |
| Vaccinated | If the pet has been vaccinated. 1 = yes, 2 = no, 3 = unsure. |
| Dewormed | If the pet has been dewormed or not. 1 = yes, 2 = no, 3 = unsure |
| Sterilized | If the pet has been spayed or neutered. 1 = yes, 2 = no, 3 = unsure |
| Health | The health of the animal. 1 = healthy, 2 = minor injury, 3 = serious injury, 0 = unspecified |
| Quantity | The number of pets represented in the profile. |
| Fee | Adoption fee in dollars. 0 = free. |
| State | The location in Malaysia (state). There are 15 different states which can be decoded using the state labels dictionary file. |
| Rescuer ID | The unique hash ID of the rescuer. This attribute will not be used. |
| Video Amount | The number of videos uploaded for the pet. |
| Photo Amount | The number of photos uploaded for the pet. |
| Description | The profile write-up for the pet. The primary language is English, with some Malay or Chinese (Mandarin/Cantonese) |
| Sentiment Magnitude | Magnitude represents the overall strength of emotion (both positive and negative), as determined by sentiment score. |
| Sentiment Score | Score represents the emotional lean of the text. -1 for negative and 1 for positive. |
| Image Metadata Dominant Color | Three RGB values for representing the dominant color of the animal. |

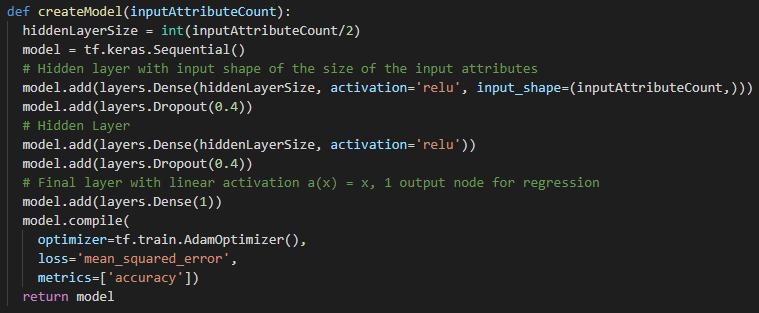
**Pre-Processing**

We need the data to be in numerical format for our neural network. Our group decided that we should normalize the data so that some attributes are not considered more important than others. String data such as the name and description attributes are transformed to numerical attributes by finding their length and normalizing them to a 0 - 1 range. Nominal categorical attributes are one-hot encoded so that each possibility becomes an integer attribute. All ordinal categorical attributes are normalized to a 0-1 range, unless the attribute has a selection for “not recorded”. An improvement in the future would be to separate the “not recorded” component and the ordinal components to retain more information. As a group, we decided upon the below pre-processing procedures.

|  |  |
| --- | --- |
| Specific Pre-Processing Procedure per Attribute | |
| Adoption Speed | None, this is our target. |
| Pet ID | Not used directly |
| Type | One-hot encoded |
| Name | Length(name), normalized |
| Age | Normalized |
| Breed 1 | One-hot encoded |
| Breed 2 | One-hot encoded |
| Gender | One-hot encoded |
| Color 1 | One-hot encoded |
| Color 2 | One-hot encoded |
| Color 3 | One-hot encoded |
| Maturity Size | One-hot encoded (should consider separating ordinal and categorical) |
| Fur Length | One-hot encoded (should consider separating ordinal and categorical) |
| Vaccinated | One-hot encoded |
| Dewormed | One-hot encoded |
| Sterilized | One-hot encoded |
| Health | One-hot encoded (should consider separating ordinal and categorical) |
| Quantity | Normalized |
| Fee | Normalized |
| State | One-hot encoded |
| Rescuer ID | Not used (removed) |
| Video Amount | Normalized |
| Photo Amount | Normalized |
| Description | Length(description), normalized |
| Sentiment Magnitude | Mean of all sentiments |
| Sentiment Score | Mean of all scores |
| Image Metadata Dominant Color | 3 attributes (RGB), normalized |

The data pre-processing flow is as follows. The data is read from the data directory and a NumPy array is created with the proper resultant shape. Currently, there are 689 resultant attributes after pre-processing is complete so the NumPy array has shape (, 689). Each column (attribute) in the unprocessed data is processed for all samples before moving onto another attribute. All attributes are processed according to the procedure above.

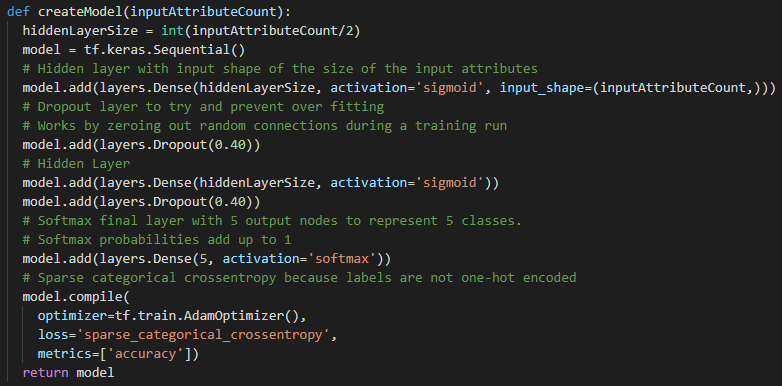
**Network Architecture A**

****

This architecture was developed by Areege Chaudhary. It uses Keras’ sequential model building to create the desired architecture. The approach here is to use two dense (fully connected) hidden layers with 344 nodes in each layer. Each node uses a rectified linear unit activation function, R(z) = max(0, z). Each layer employs the dropout regularization technique which randomly zeros out the activations of 40% of the hidden units at that layer. Dropout is used to prevent overfitting by limiting the ability of nodes to overly contribute to error correction. Without dropout, nodes may overly correct specific errors rather than generally correcting errors for a wide range of network states. The final layer is a single node dense layer with a linear activation function, a(x) = x. The reason for this single node is that in this case, we are trying to perform ordinal regression. That is, we are binning the result of the regression to fit our class labels (0-4). Any regression result which is less than 0 is binned into the 0 class and any result which is greater than 4 is binned into the 4 class. All results within the class range are rounded. Mean squared error is used because we are attempting regression. Finally, the Adam Optimizer is used to perform stochastic gradient descent since it works well for noisy and sparse gradients, which is what we have given the nature of the inputs.

The resulting accuracy of this method was on average, 30% on the test set, with some variation. The accuracy given random class selection is approximately 18%, so the model has acquired some predictive ability.

**Network Architecture B**



This architecture was developed by Garett MacGowan. It also uses Keras’ sequential model building to create the desired architecture. The approach here to use two dense hidden layers with 344 nodes in each layer. In this case, each node uses a sigmoid activation function, S(x) = 1 / (1 + e-x). Sigmoid was used because it seemed to help improve performance. The performance improvements seen here may be a result of sigmoid providing a denser representation since activations will never be zero, except when x is extremely small, and truncation occurs. Each layer employs dropout regularization, zeroing 40% of the hidden units at that layer. The final layer is a five-node dense layer with SoftMax as the activation function. The reason for using SoftMax is that we want to treat the problem as if it were nominal. Since the bins represent different time scales, it may be useful to treat them as independent. The sum of the outputs of the SoftMax output layer is always one. Sparse categorical cross entropy is used as the loss function because the class labels are not one-hot encoded. Finally, the Adam Optimizer is again used to perform stochastic gradient descent.

The resulting accuracy of this method was on average, 38% on the test set, with some variation. The accuracy given random class selection is approximately 18%. This result is relatively much better than architecture A. The increase in performance may be attributed to predicting class probabilities instead of forcing a regression mapping for a skewed ordinal target.

**Comparing to Existing Model**

The highest scoring model submitted so far on Kaggle is an XGBoost model which is a decision tree boosting model. It achieves 45% accuracy and is thus far better than the models generated here. There are no kernels on Kaggle that have an accuracy over about 42% without using ensemble learning techniques. Given that our best model doesn’t use ensemble techniques and has been seen to reach 40% accuracy on certain runs, it is possible that our model could perform much better with the use of ensemble techniques. Our model may be able to perform better by using more of the metadata and sentiment data. Our model might also perform better if a convolutional neural network was employed on pet images to create new predictive features such as cuteness, desperateness, etc.

**Related Works**

*Classification of Ordinal Data Using Neural Networks – (Areege)*

This paper introduces a model for a feedforward neural network for multiclass classification problems, where the classes are ordered. The method assumes that in a supervised classification problem with ordered classes, the random variable class associated with a given query x should be unimodal. The model has one output unit that takes in values in the interval [0,1]. This interval is then subdivided into K sub-intervals (one for each class) according to a parametric model. The parametric model assumes that the output values come from a binomial distribution. For a given query x, the output of the network will be a single numerical value in the range [0,1]. This value is then plugged into a binomial model to calculate P(Ck|x) where k is the number of classes. 

This is somewhat similar to architectures A’s method of binning the linearly activated output of the output node into the 5 target classes.

*Learning to Classify Ordinal Data: The Data Replication Method – (Areege)*

This paper uses data replication methods to reduce the problem of classifying multiple ordered classes to the standard two class problem. The data replication method is then mapped into support vector machines and neural networks. This paper uses an approach that is quite different from ours, so we cannot really compare.

*A Neural Network Approach to Ordinal Regression – (Garett)*

This paper describes a neural network approach for ordinal regression that has the advantages of neural network learning. The method used can be considered a generalization of the perceptron learning into multi layer perceptrons for ordinal regression. The method works on individual data points and uses multiple output nodes to estimate the probabilities of ordinal categories. The network is trained using back-propogation. This paper uses a method similar to the one we used in archicture B. However, instead of using softmax as the output layer activation function, it uses the sigmoid activation function.

*A Simple Approach to Ordinal Classification – (Garett)*

This paper presents a simple method that enables standard classification algorithms to exploit the ordering information in ordinal prediction problems. First, the data is transformed from a k-class ordinal problem to k-1 binary class problems. Training starts by deriving new datasets from the original dataset, one for each of the k-1 new binary class attributes. Each derived dataset contains the same number of attributes as the original, with the same attribute values for each instance – apart from the class attribute. Next, the classification algorithm is applied to generate a model for each of the new datasets.

This approach is also quite different from the methods we used. It is essentially a bagging ensemble technique. This solution would have less variance than our predictor and is analogous to what we have stated in our ideas for next steps.

**Summary and Next Steps**

In this paper, we discussed our data pre-processing approach for which all members assisted in developing, and our two neural network models, architecture A, developed by Areege Chaudhary, and Architecture B, developed by Garett MacGowan. Our best model, architecture B, was able to achieve peak accuracy of approximately 40%, and an average accuracy of approximately 38%. This is not as effective as the current best model on Kaggle which achieves 45% accuracy. However, our group believes that with ensemble techniques such as bagging and boosting, our accuracy variance can be decreased, and our overall accuracy can be increased. By deploying a convolutional neural network to learn new predictive features such as cuteness, desperateness, and other self-generated features from pet images, the model might be able to achieve an even higher accuracy.

**References**

Cardoso, J. S., & Pinto da Costa, J. F. (2007). Learning to Classify Ordinal Data: The Data Replication Method. *Journal of Machine Learning Research 8,*1393-1429. Retrieved from http://www.jmlr.org/papers/volume8/cardoso07a/cardoso07a.pdf

Cheng, J. (2007). A Neural Network Approach to Ordinal Regression. Retrieved from https://arxiv.org/pdf/0704.1028.pdf.

Frank, E., & Hall, M. (2001). A Simple Approach to Ordinal Classification. *Machine Learning: ECML 2001. 12th European Conference, Freiburg, Germany, September 5–7, 2001. Proceedings,*145-156. Retrieved from https://www.cs.waikato.ac.nz/~eibe/pubs/ordinal\_tech\_report.pdf.

Pinto da Costa, J. F., & Cardoso, J. S. (2005). Classification of Ordinal Data Using Neural Networks. *Machine Learning: ECML 2005, 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005, Proceedings,*690-697. doi:10.1007/11564096\_70