Deep Learning techniques applied to prediction from images. Use case: pet's adoption

David Mora Garrido

Undergraduate Dissertation

Bachelor’s Degree in Computer Science and Engineering, Major in Computing

29th July, 2021

Index

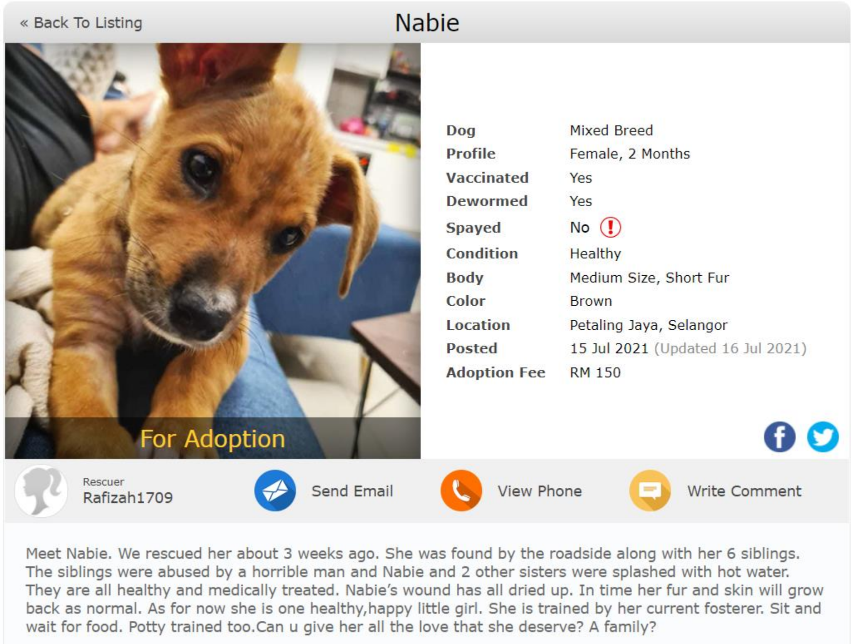
1. Introduction
2. Exploratory Data Analysis
3. Feature Engineering
4. Model creation and evaluation
5. Conclusions and future work

Index

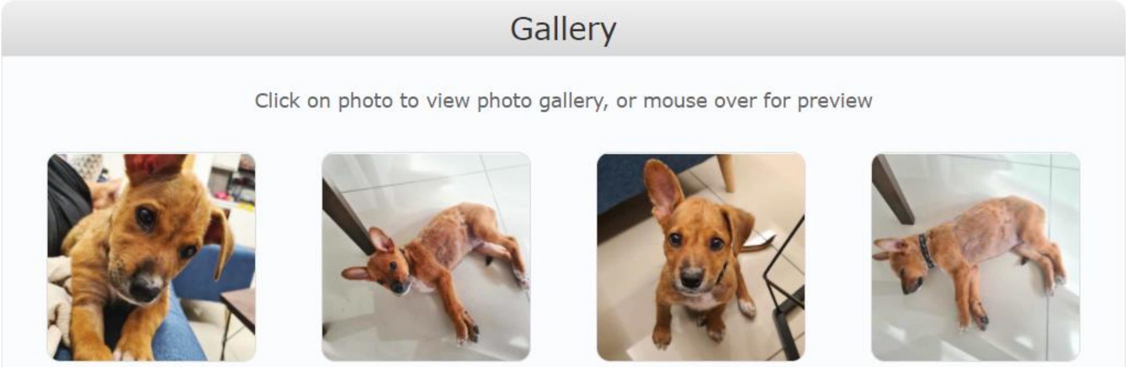
1. Introduction
    - Objectives
    - Problem domain
2. Exploratory Data Analysis
3. Feature Engineering
4. Model creation and evaluation
5. Conclusions and future work

Introduction

- Kaggle Competition (already finished)
- Objective: estimate the adoption speed of a pet profile
- Great variety and formats of data



PetFinder.my



Objectives

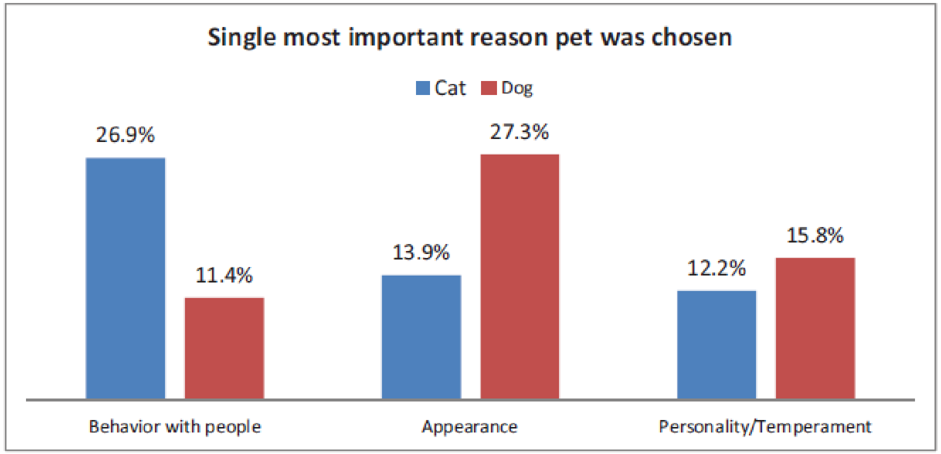
Conducting a **data mining process**, as a sequence of stages (subobjectives):

1. Understand the problem domain.
2. Analyse the data and get early feature engineering ideas.
3. Clean, transform and extract information from the data, focusing on image feature extraction with Deep Learning techniques.
4. Create several models based on different paradigms and evaluate their performance.

Problem domain: studies

Malaysia:

- Better than surrounding countries in animal welfare policy
- Cats are preferred over dogs: religion and strict regulation
- More particularities: local cat breed, overpopulation of stray animals, dogs as tools rather than pets



Extracted from [Weiss et al., 2012]

Images

- Not only pet appearance (mouth, ears, position), background can also give information (being in a kennel or not, grass, etc.)

Text

- Name is not relevant
- Descriptions: third person or first person, presence or absence of certain terms

Index

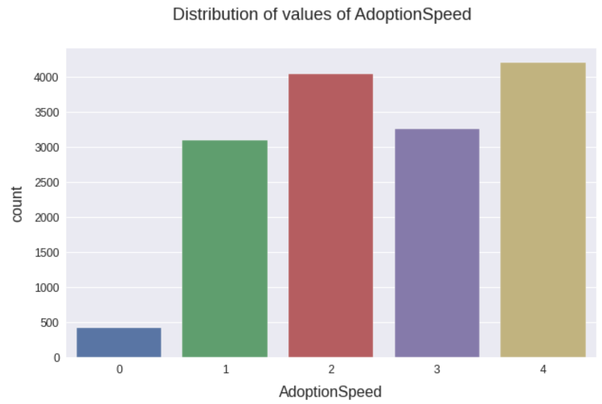
1. Introduction
2. Exploratory Data Analysis
   - Base tabular data
   - Text data
   - Text metadata
   - Image data
   - Image metadata
   - Image properties
3. Feature Engineering
4. Model creation and evaluation
5. Conclusions and future work

Base tabular data

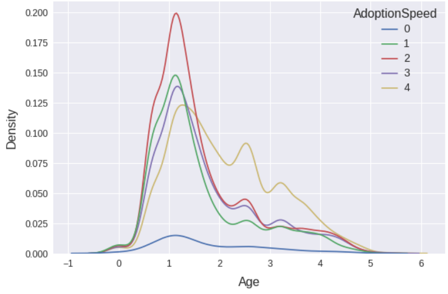
Complete list of variables:

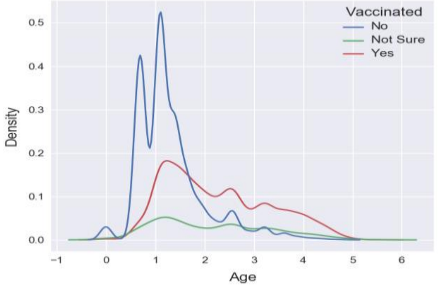
- ***AdoptionSpeed***
- *PetID*
- *Type*
- *Age*
- *Breed1, Breed2*
- *Gender*
- *Color1, Color2, Color3*
- *MaturitySize*
- *FurLength*
- *Health*
- *Vaccinated, Dewormed, Sterilized*
- *Quantity*
- *Fee*
- *State*
- *RescuerID*
- *VideoAmt*
- *PhotoAmt*

- Categorical target, ordinal relationship:
  - 0: adoption took place on the same day as the profile was published.
  - 1: between 1 and 7 days.
  - 2: between 8 and 30 days.
  - 3: between 31 and 90 days.
  - 4: no adoption took place 100 days after the publication (there are no pets that waited between 90 and 100 days in the training data).
- Imbalaced, but not worrying



Base tabular data





Complete list of variables:

- *AdoptionSpeed*
- *PetID*
- ***Type***
- ***Age***
- ***Breed1, Breed2***
- *Gender*
- *Color1, Color2, Color3*
- *MaturitySize*
- *FurLength*
- *Health*
- ***Vaccinated, Dewormed, Sterilized***
- *Quantity*
- *Fee*
- *State*
- ***RescuerID***
- *VideoAmt*
- *PhotoAmt*

- *Age*: young pets (less than 6 months approx.) were adopted earlier than older pets.
- *Breed* variables: very high cardinality (175 + 135 unique values). ‘Pure’ breed pets were adopted earlier.
- Health-related variables:
  - *Vaccinated*: pets that were not vaccinated were generally adopted earlier. This is mainly explained by *Type* and *Age.*
  - *Dewormed*: similar to *Vaccinated*, ‘Not Sure’ values significantly increased the proportion of ‘4’ cases.
  - *Sterilized*: the proportion of not sterilized pets with a ‘4’ outcome significantly decreases (20%).
- *RescuerID*: very high cardinality (5595). Replaced by count encoding as a proxy of the type of rescuer. Profiles published by novice rescuers are more likely to end up not being adopted.

Text data and metadata

Sentiment analysis:
- Magnitude
- Score

Entity analysis:
- Salience

Other variables:
- Language

- *Description*: valuable source of information
  - First or third grammatical person
  - Information about the context of the pet
  - Past, present, future tenses
  - 6.3% of texts are repeated (‘For adoption’, templates)
- *Magnitude* and *Score* give an estimation of the text’s overall strength of emotion and whether it is positive or not, respectively. For example:
  - *Magnitude* = 10.0 and *Score* ≈ 0.0 → Mixed emotions
  - *Magnitude* = 1.0 and *Score* ≈ 0.0 → ‘Neutral’ text
- Not all the texts could be analysed → those written in Malay or mix of Malay and English
  - Significant difference on *AdoptionSpeed* when the language is Malay or Chinese.

Image data

- 58311 training images, 14652 of them are the ‘profile image’.
- Different aspect ratios, lighting conditions, collage or text, pet’s face is not shown or only a portion is occupied by the pet, different positions, etc.

- Not necessarily bad, could give information (environmental or background elements too, like a kennel, cage, etc.).
- Profile images showing a single pet and then giving up for adoption multiple pets, or vice versa (collages).
- Extra images can provide useful information.

Photos of pet 23b3f793e (index 4115)

Image metadata and properties

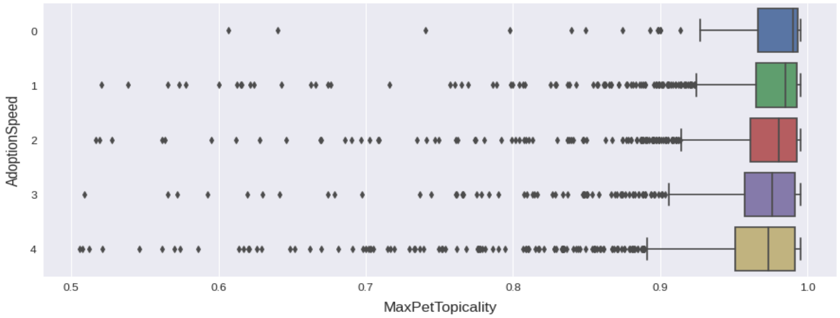
Metadata:

- Presence of human faces, detection confidence and probability of showing certain emotions (discarded)
- Text annotations in the image
- Dominant colors and their pixel fraction
- **Label annotations (entities), with their Topicality**

Extracted properties:

- Dullness
- Whiteness
- Blurriness
- Size
- Width
- Height
- Aspect ratio

- Entities:
    - Maximum pet (entity containing ‘cat’ or ‘dog’) topicality.

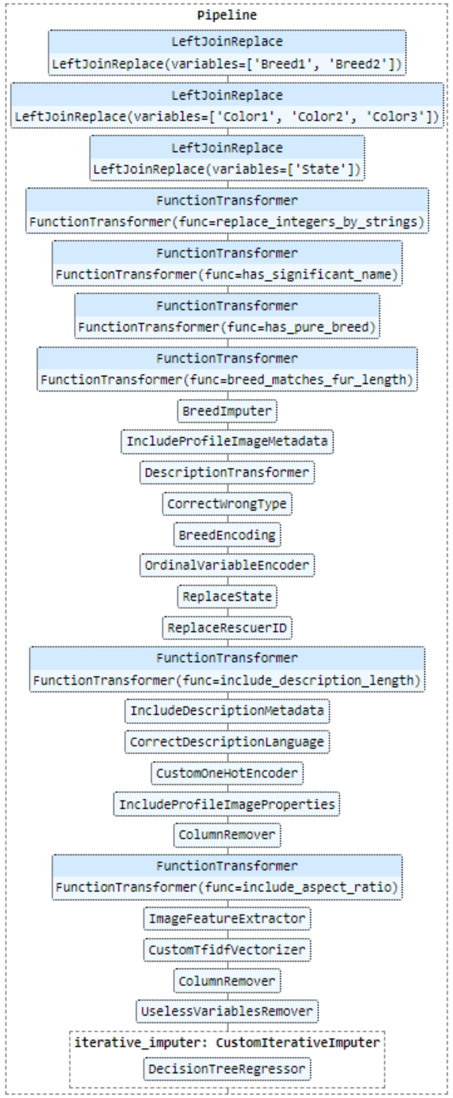


    - The smaller the number of entities, the greater proportion of ‘4’ cases.
    - Description or labels can be used to fix *Type* anomalies.
- The probability of ending up without adoption is higher if the profile image has an aspect ratio smaller than 1.0.
- The proportion of ‘4’ cases with a very dull profile image is greater.

Index

1. Introduction
2. Exploratory Data Analysis
3. Feature Engineering
   - Base and additional tabular data
   - Image features extraction
   - Text features extraction
4. Model creation and evaluation
5. Conclusions and future work

Feature Engineering



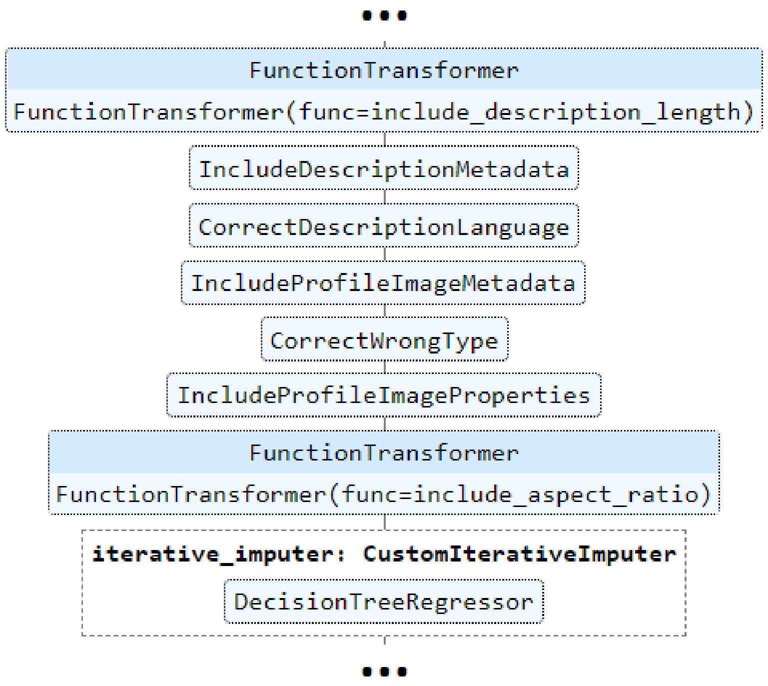
Base tabular data

```
LeftJoinReplace
LeftJoinReplace(variables=['Breed1', 'Breed2'])

LeftJoinReplace
LeftJoinReplace(variables=['Color1', 'Color2', 'Color3'])

LeftJoinReplace
LeftJoinReplace(variables=['State'])

FunctionTransformer
FunctionTransformer(func=replace_integers_by_strings)

FunctionTransformer
FunctionTransformer(func=has_significant_name)

FunctionTransformer
FunctionTransformer(func=has_pure_breed)

FunctionTransformer
FunctionTransformer(func=breed_matches_fur_length)

BreedImputer

...
```

- Breed variables…
  - New variables: whether the pet has a ‘pure’ breed and whether the specified *FurLength* value matches the *Breed1* value
  - Very high cardinality
  - Important variables (especially *Breed1*)
  - Retaining as much information as possible while representing them in a manageable way?

Option 1: **Target encoding**
Option 2**: One-hot encoding + SVD**

Additional tabular data



- Include *Description* metadata.
  - Language is corrected: those below a minimum sample size are grouped and null values are replaced by "ms" (Malay).
- Image metadata and properties variables are included.
  - Hyperparameter to specify whether to add only the profile image metadata/properties, or an aggregation of all the images.
  - Image description is used to correct *Type* anomalies (involves checking *Name*, *Description* and external breeds file).
- Need to impute the values of the *Description* metadata variables for those texts written in Malay + some image properties of profiles without any image.

Iterative imputation

Image features extraction

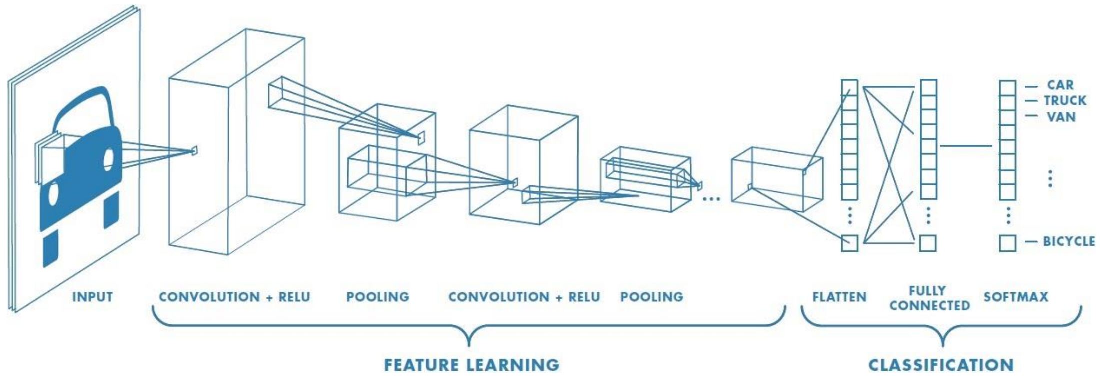
- Image metadata and properties: extracted without considering the target
- What we want: extracting the best possible features for predicting *AdoptionSpeed*

↓

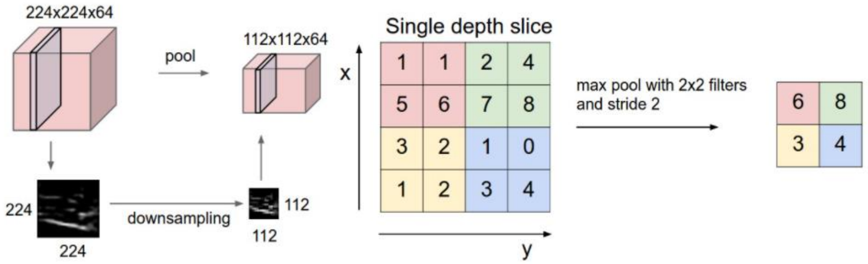
Deep Learning

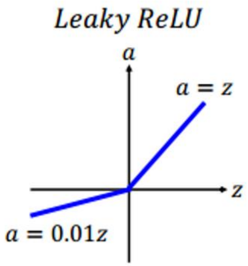
Deep Neural Networks: CNNs

Feature extraction: from lower-level to higher-level features for the intended task

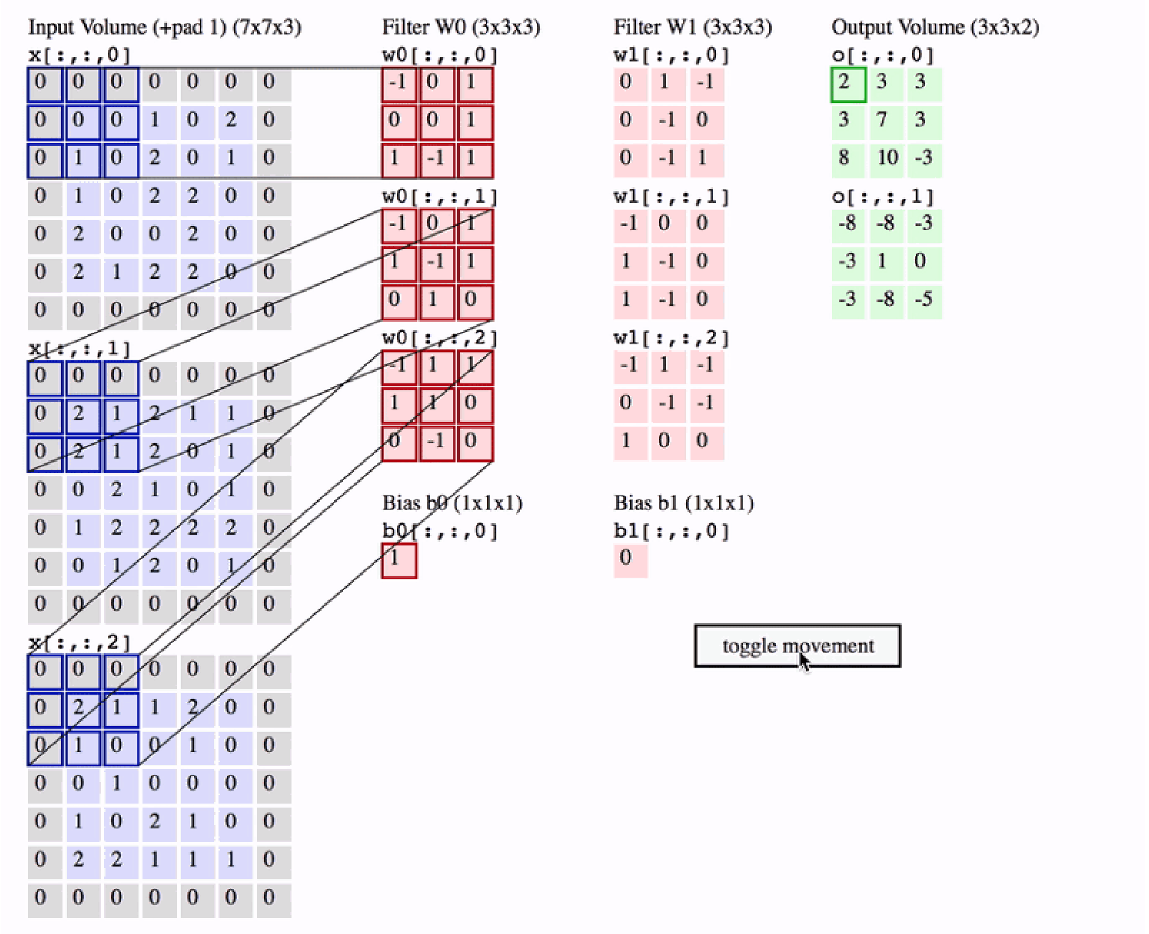


Other components for downsampling, regularization, reducing the vanishing gradient effect…





Main operation: convolution



CNNs: initial approach

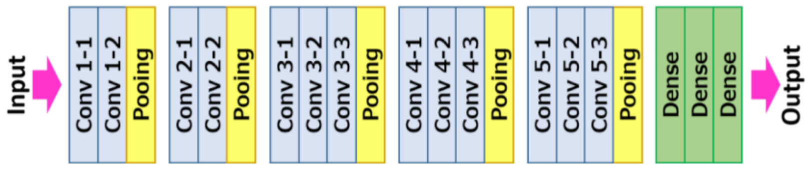
We could train a CNN on *AdoptionSpeed* and then extract a summary of the last volume of feature maps… However:

- What are we looking for? Background or surrounding elements? ‘Cuteness’? What defines ‘cuteness’?
- Predicting *AdoptionSpeed* is definitively not the same as predicting a hand-written digit, nor the *Type* of pet…

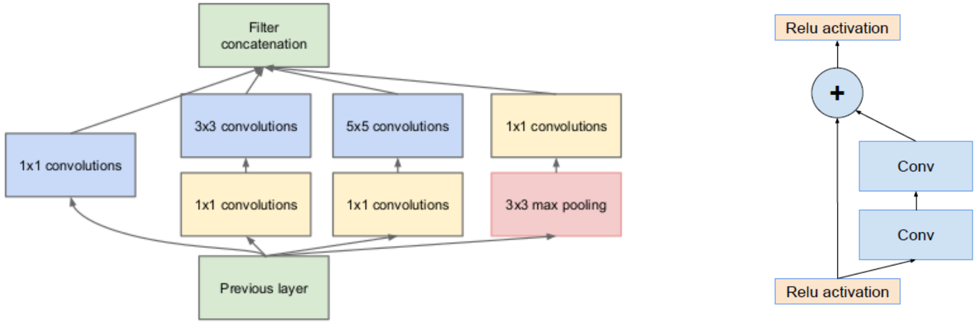
Pre-trained CNNs

Pre-trained CNNs

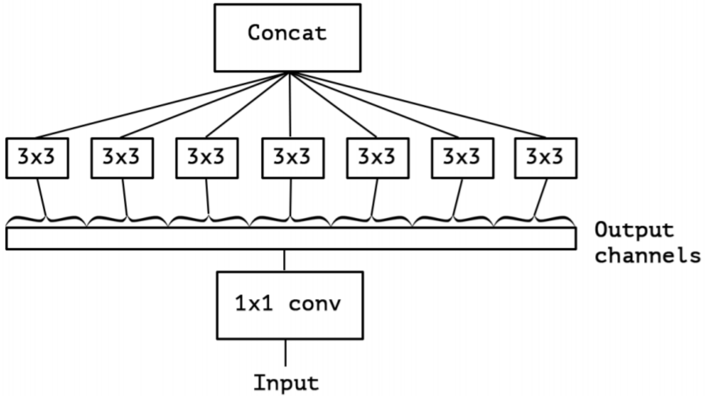
VGG16



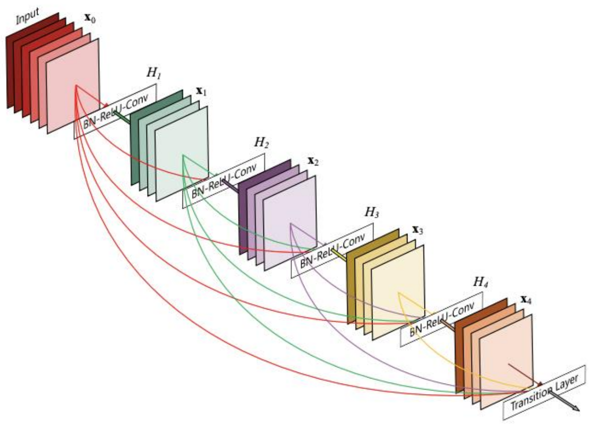
InceptionResNetV2



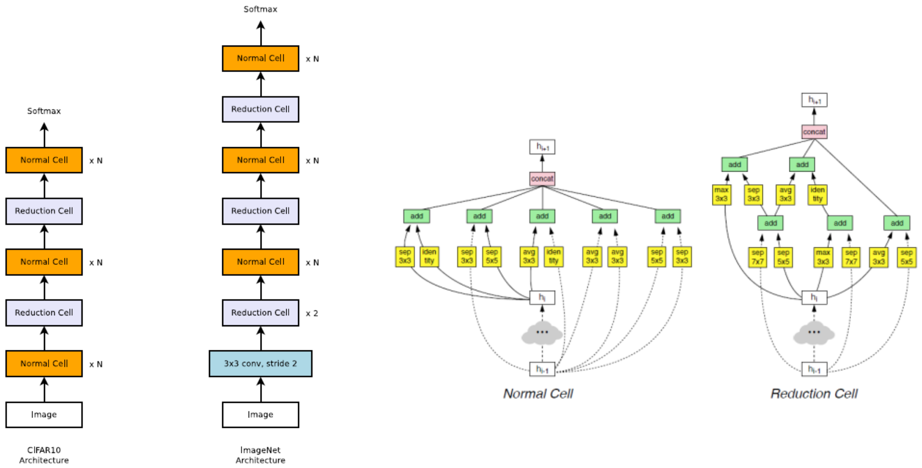
Xception



DenseNet121 and 201



NASNetLarge



Selecting the most suitable backbone

Simple preprocessing of images:

- Resize to 256x256
- Pad them inside the square so that the aspect ratio is maintained
- Apply CNN preprocessing function


Kaggle restrictions: cannot train too many models (9/30 hours limit)

- We need to evaluate the most suitable CNN: extract features + include them in the pipeline + evaluate using XGBoost. Problem: too many features.
  - Global average pooling to reduce number of features (still too many)
  - Applying truncated SVD (16, 32, 64 components)
- Train on top of the selected backbone

Selecting the most suitable backbone

CNN	SVD	Average fit time (s)	Average accuracy	Average QWK
DenseNet121	16	32.302502	0.422797	0.371695
NASNetLarge	16	51.264992	0.423198	0.373589
VGG16	16	28.980260	0.421264	0.369433

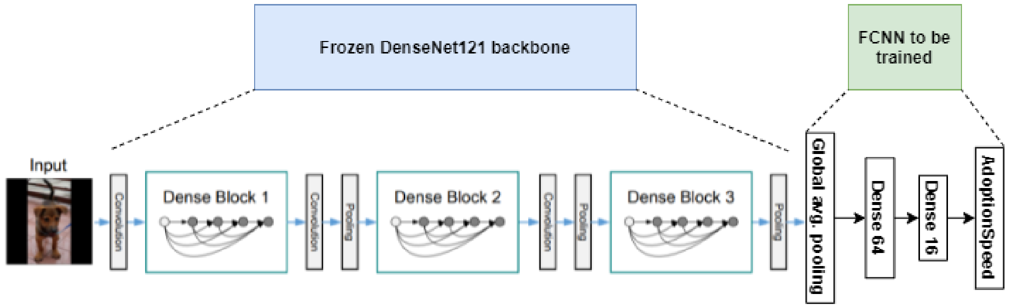
The pre-trained backbone of DenseNet121 is selected considering the results.

Some other configurations that we have evaluated:

- Smaller number of SVD components
- Different image size (224, 384, 512)
- Averaging again groups of features.

Transfer Learning

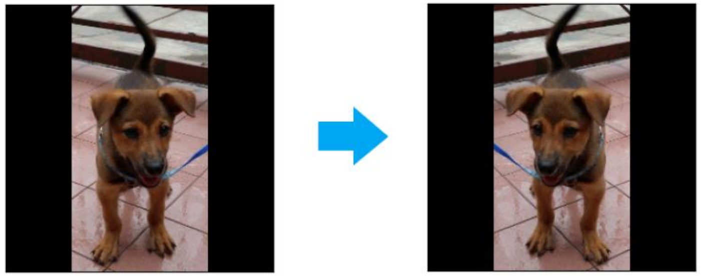
Transfer Learning: using DenseNet121 backbone to extract high-level features and train a fully-connected neural network on top to predict *AdoptionSpeed*.



Different evaluation strategy: 5-CV not affordable, single training-validation split instead

- Training: 46617 images (11994 profiles)
- Validation: 11694 images (2999 profiles)

All the images of each profile are either in the training or in the validation set



Setting up the FCNN:

- Different configurations (16, 32, 64, 64-32, 64-16, 32-16)
- Little data augmentation (small number of epochs)
- *AdoptionSpeed* as a classification problem

Transfer Learning

Best average validation results (loss, accuracy, QWK) over 5 epochs: 64-16 → two options to extract features

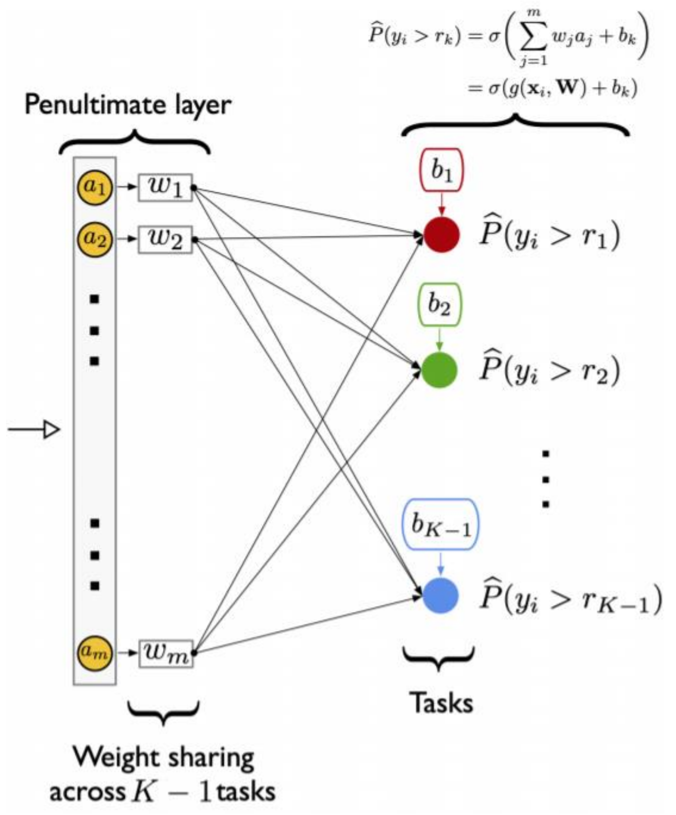
Then, we trained 3 different configurations with that set-up:

- Classification (softmax + categorical cross-entropy)
- Regression (linear activation + mean squared error)
- Ordinal regression: Consistent Rank Logits (CORAL) approach ([Cao et al., 2019])

[0] -> [0, 0, 0, 0]
[1] -> [1, 0, 0, 0]
[2] -> [1, 1, 0, 0]
[3] -> [1, 1, 1, 0]
[4] -> [1, 1, 1, 1]

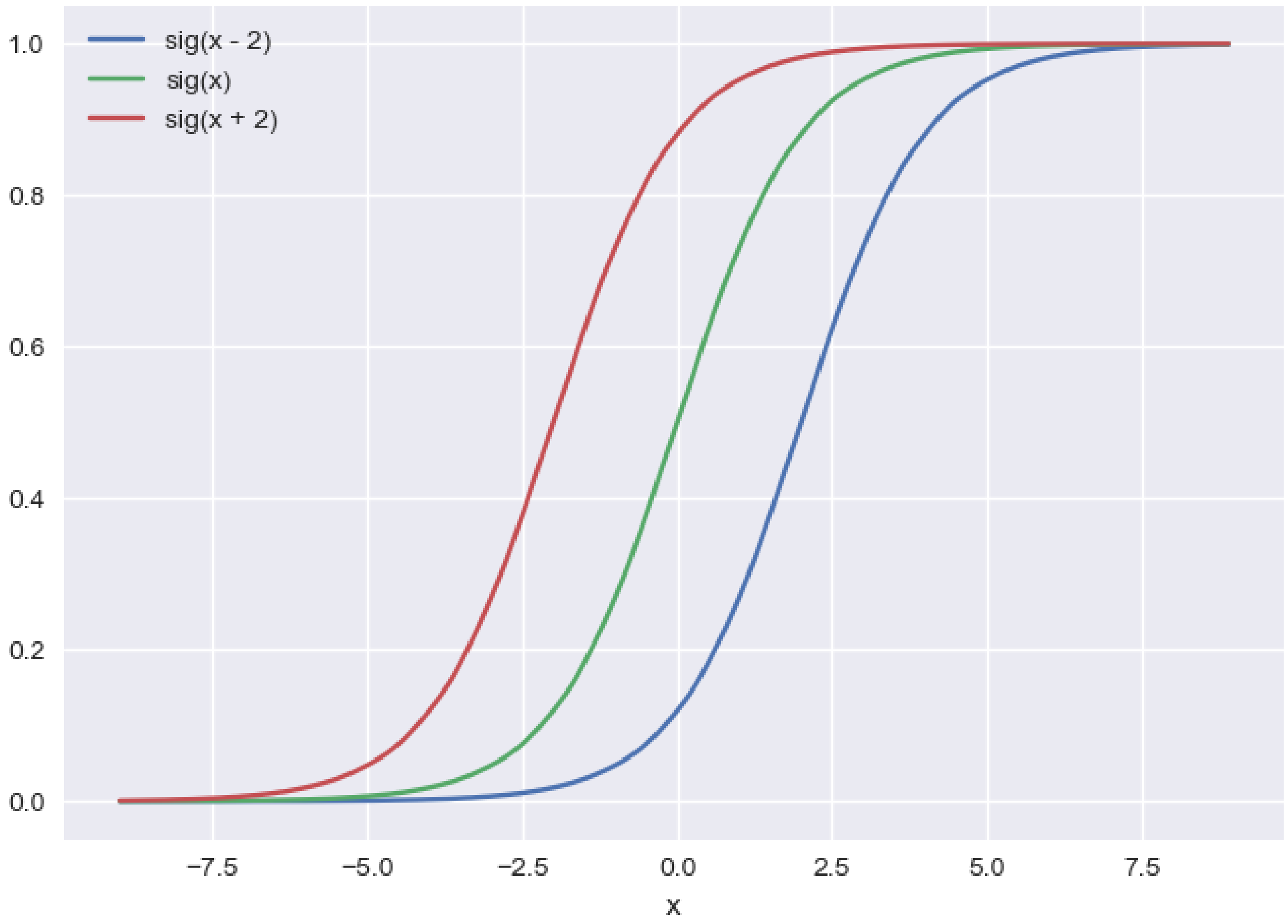
(sigmoid instead of softmax)
Output of i^{th} neuron: probability of being > V_i
Problem: inconsistencies → [1, 0, 0, 1] ?

SOLUTION



Same shared weights, but independent bias units
Custom loss function allowing rank consistence

Ordered bias units

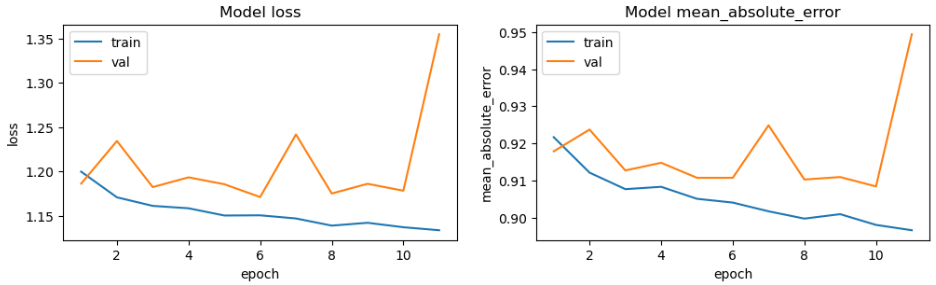


Transfer Learning

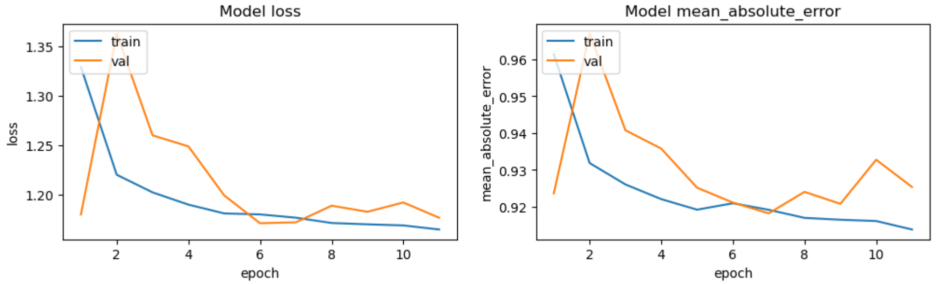
The regression CNN seemed to have more room for improvement.

Improvements (regularization, generalization):

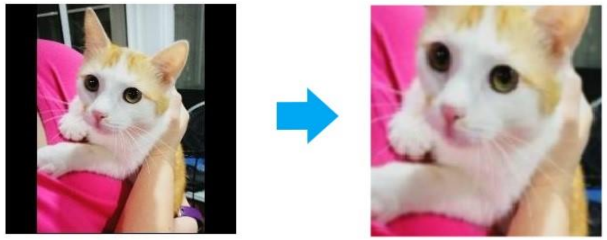
- Different set-ups of Dropout layers
- More data augmentation methods



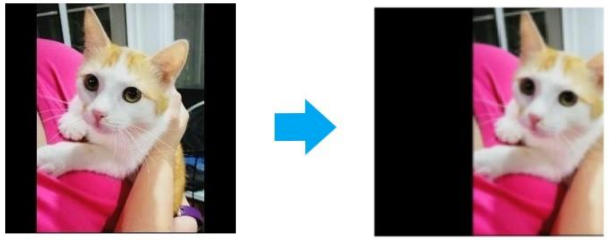
(no dropout)



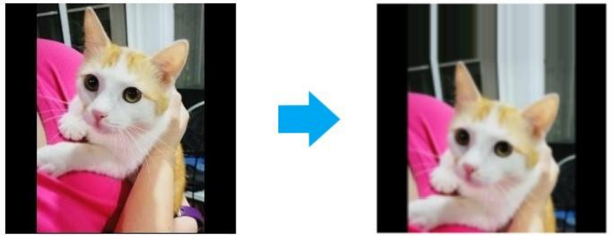
(with dropout)



Zoom



Width shift



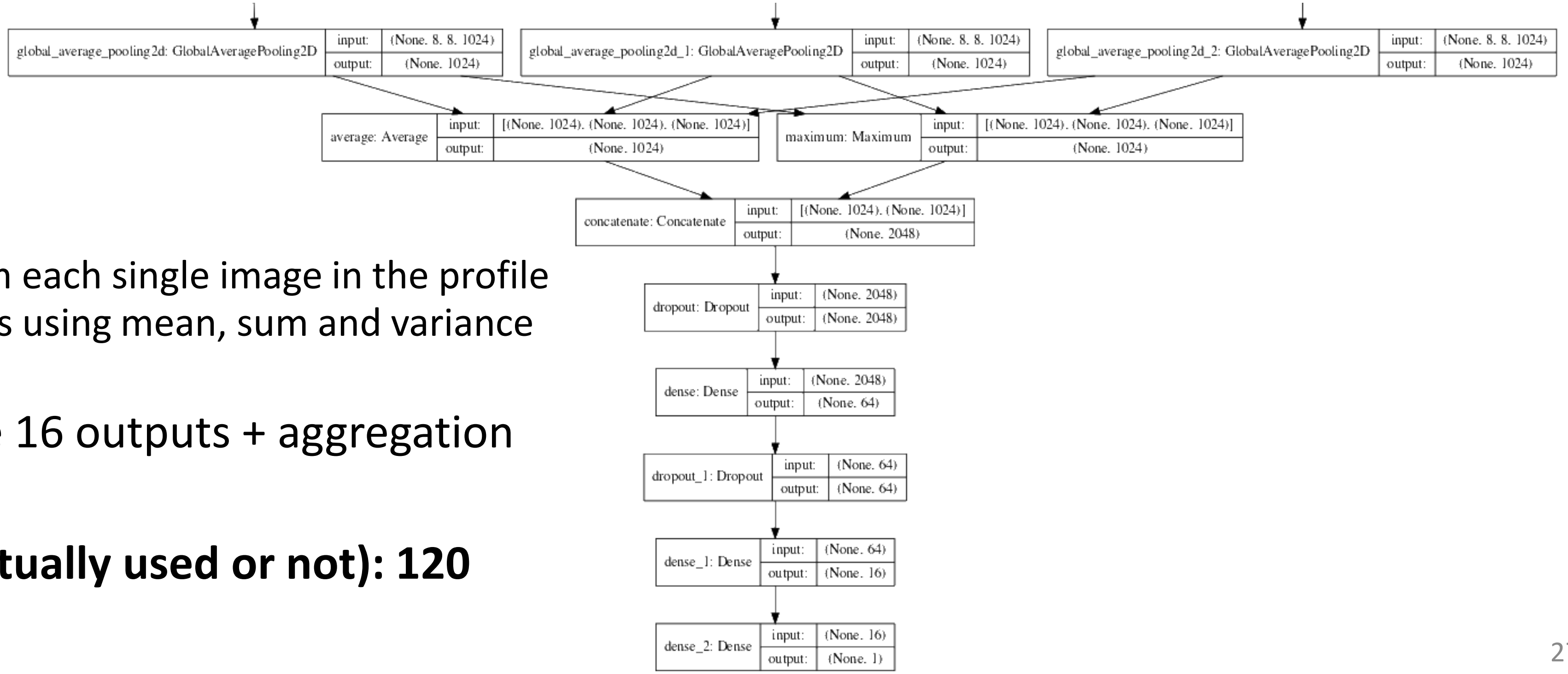
Height shift

As we expected, the number of useful features from the regression CNN increased.

Final CNN model

Fine tuning with a smaller learning rate (unfreezing last dense block). Applied on best model, didn't improve one of the options. Instead:

- Single input ensemble:
  - Three DenseNet121 backbones
  - Each one with a different number of unfrozen dense blocks
  - First FCNN Dropout rate increased



- Using all the images:
  1. Extract the features from each single image in the profile
  2. Aggregate those features using mean, sum and variance

Final model: ensemble Dense 16 outputs + aggregation

Training all the models (eventually used or not): 120 hours, 4 Kaggle GPU weeks

Text features extraction

Description texts needs to be cleaned and transformed. Steps:

1. Remove emojis (no clear influence on *AdoptionSpeed*)
2. Translate non-English descriptions into English (required for the next models)
3. Expand English contractions (*pycontractions*):
   - Word embeddings (GloVe, Twitter)
   - Word Mover's Distance
4. Replace punctuation symbols by whitespace for correct tokenization

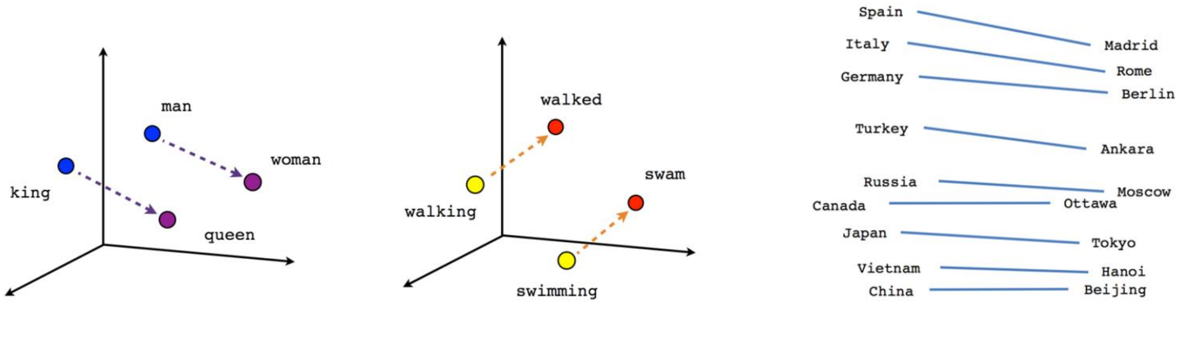
Expanding contractions

pycontractions approach: generate several possible combinations and use the closest one to the original text according to the WMD measure

”I’d like to travel if I’d money”:

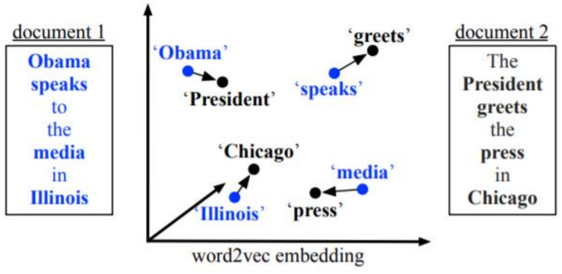
- ”I had... I had...”,
- ”I had... I would...”,
- ”I would... I had...”,
- ”I would... I would”

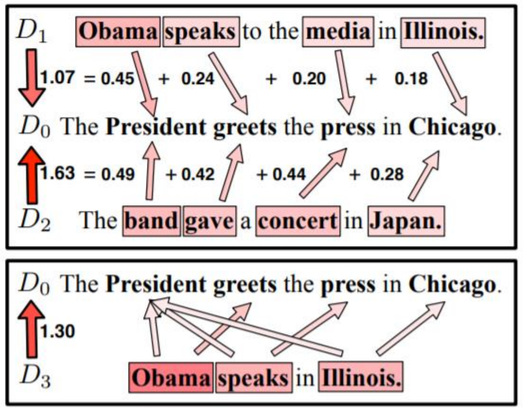
- Word embeddings



Male-Female | Verb tense | Country-Capital

- Word Mover’s Distance





Text features extraction

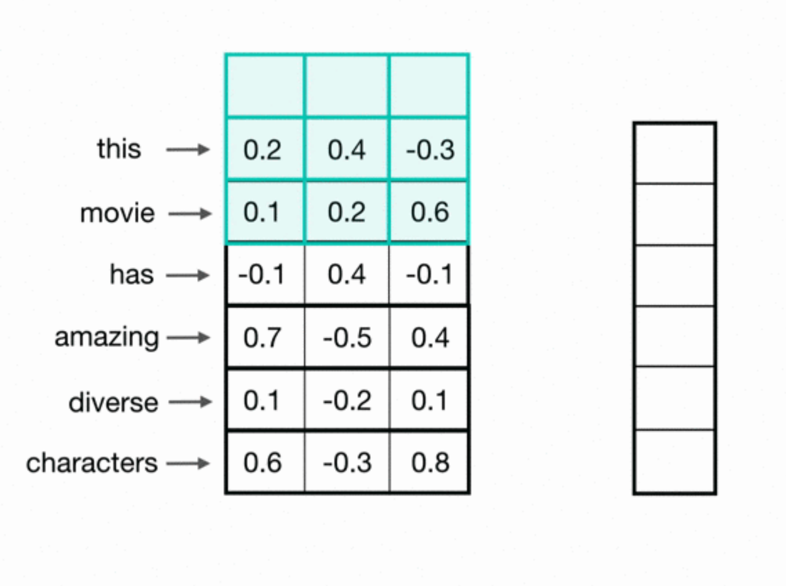
🎈 ADOPTION 领养贴 🎈 Area : PENANG (If other areas whatsapp me.) - Simple requests（小條件）: (if you agree with all of this, hit me up ~) 1)Spaying (結紮） 2)Keep indoors, Avoid caging too frequently. （養家裡, 不關籠） 3)No supermarket/minimarket catfood.（不餵超市貓糧） 4)Regular vet check ups. 5)Family is fine with you keeping a cat .（家人允許） -宾士宝贝, 母的, 4月29日出生(差不多6个月大), 打完预防针 - Tuxedo kitty, female, almost 6 mnths old, got all her vaccines as a kitten . -性格 : 一开始会很胆小, 久了会很好相处, 不挑食, 独立聪明厉害观察奴才在干嘛😂 - Personality: Shiki is a scaredy kitten at first, she'll be amazing company after she gets used to you :), Not a picky-eater , Really independent. 希望会遇到有缘人 💞 Are you willing to give this little penguin a chance? Whatsapp me at

↓

ADOPTION Adoption stickers Area PENANG If other areas whatsapp me Simple requests small conditions if you agree with all of this hit me up 1 Spaying ligation 2 Keep indoors Avoid caging too frequently raising the family not closing the cage 3 No supermarket minimarket catfood 4 Regular vet check ups 5 Family is fine with you keeping a cat Family allowed Benz Baby mother born on April 29th almost 6 months old after the vaccination Tuxedo kitty female almost 6 mnths old got all her vaccines as a kitten Character Very timid at first After a long time I will get along well not picky eaters independent smart and powerful observing what the minions are doing Personality Shiki is a scaredy kitten at first she will be amazing company after she gets used to you Not a picky eater Really independent Are you willing to give this little penguin a chance Whatsapp me at

Text feature extraction models

- **TF-IDF**
- CNN with word embeddings:
  - Convolutions over a fixed-length input (200 words) of pre-trained word embeddings (GloVe)
  - Kernels move along just 1 dimension, covers the entire vectors
  - Obtained feature maps = 1-D vectors, concatenated and forwarded to the next pooling or convolutional layer



Extracted from https://cezannec.github.io/CNN_Text_Classification/

Problems of this model (eventually discarded):

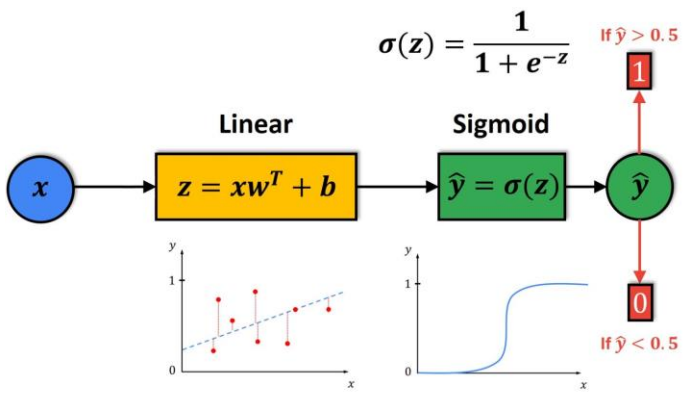
- Did overfit (with regularization techniques too)
- Not the state-of-the-art way to extract text features

Index

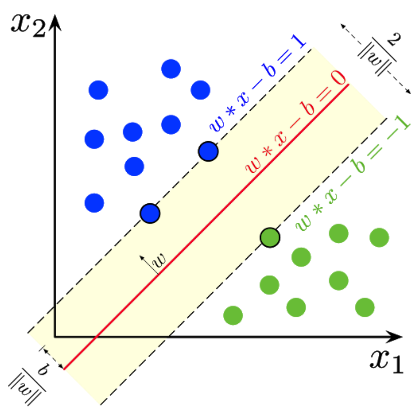
1. Introduction
2. Exploratory Data Analysis
3. Feature Engineering
4. Model creation and evaluation
   - Preliminary results
   - Classification or Regression?
   - Hyperparameter search
   - Final models
5. Conclusions and future work

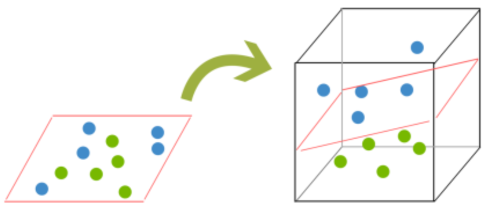
Learning algorithms

Logistic Regression

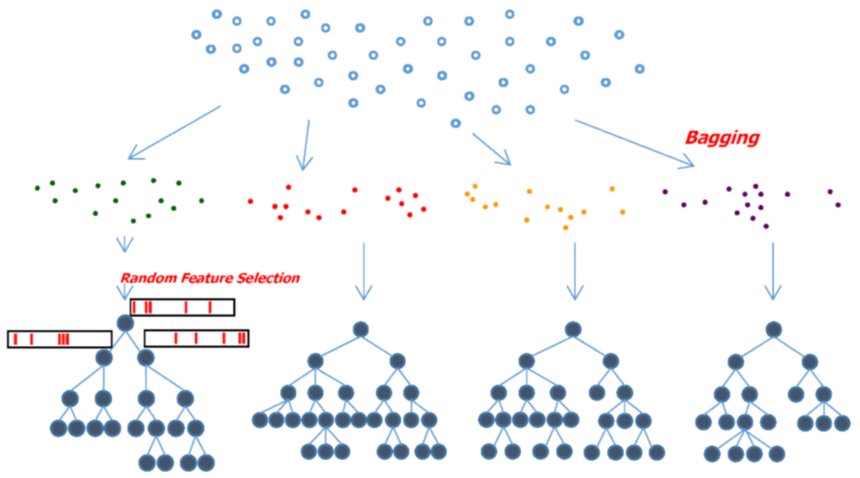


Support Vector Machines

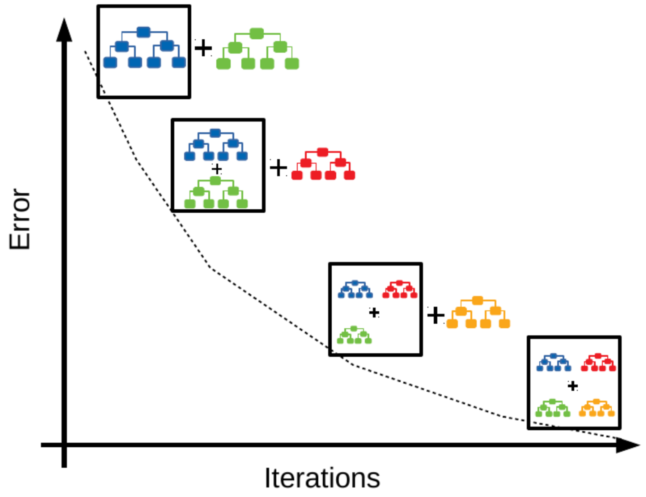




Random Forest



Gradient Boosting



dmlc XGBoost

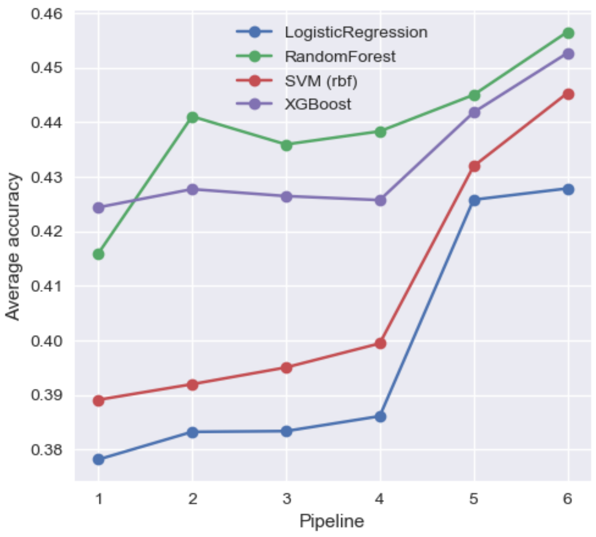
- Splitting criteria:

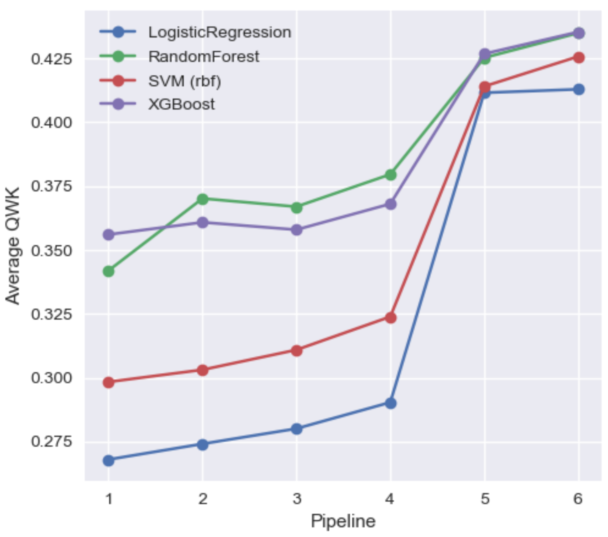
$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

- Weighted Quantile Sketch
- Parallel processing
- Column subsampling
- Cache-aware access

Preliminary results

Pipeline	Model	Average fit time	Average accuracy	Average QWK
1	XGBClassifier	7.027791	0.424397	0.356105
	RandomForestClassifier	2.654937	0.415927	0.342006
	SVC (rbf kernel)	91.539028	0.389114	0.298415
	Logistic Regression	4.030118	0.378176	0.268038
2	XGBClassifier	9.140597	0.427733	0.360878
	RandomForestClassifier	3.255018	0.441072	0.370197
	SVC (rbf kernel)	105.852106	0.391982	0.303153
	Logistic Regression	4.802423	0.383245	0.274153
3	XGBClassifier	11.879944	0.426465	0.357981
	RandomForestClassifier	4.904582	0.435936	0.366991
	SVC (rbf kernel)	97.261596	0.395050	0.310946
	Logistic Regression	6.232931	0.383378	0.280119
4	XGBClassifier	15.188047	0.425731	0.368089
	RandomForestClassifier	5.962002	0.438337	0.379648
	SVC (rbf kernel)	111.837167	0.399452	0.323938
	Logistic Regression	7.300400	0.386113	0.290413





Pipeline	Model	Avg. fit time	Avg. accuracy	Avg. QWK	1 split accuracy	1 split QWK
5	XGBClassifier	22.974383	0.441806	0.426829	0.438146	0.402173
	RForestClassifier	6.793313	0.445006	0.425277	0.437479	0.401443
	SVC (rbf kernel)	112.041703	0.431934	0.414075	0.425475	0.387992
	Log. Regression	8.006603	0.425797	0.411592	0.413471	0.387436
6	XGBClassifier	40.896393	0.452677	0.435379	0.464155	0.417941
	RForestClassifier	15.361865	0.456546	0.434955	0.441147	0.391791
	SVC (rbf kernel)	134.074037	0.445340	0.425769	0.434478	0.390644
	Log. Regression	16.713814	0.427864	0.412946	0.405802	0.372089

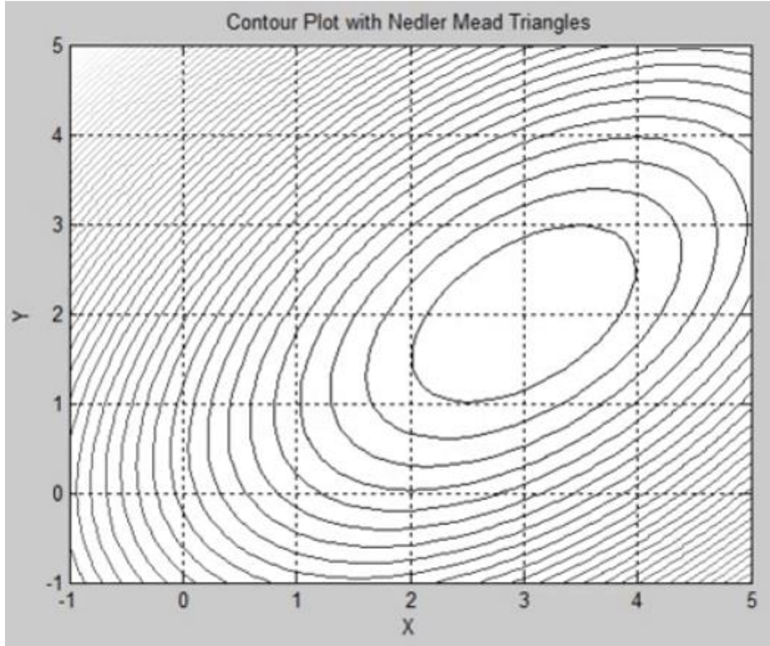
Classification or Regression?

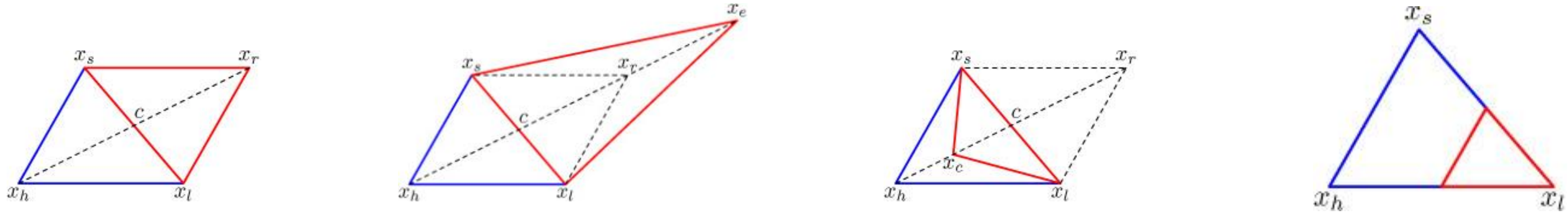
So far, we have used classifiers, but we can set out the prediction of *AdoptionSpeed* as a regression problem, and then round up or down each prediction to the corresponding class (0, 1, 2, 3 or 4).

Naïve approach (bad results): (-inf, 0.5] → 0, (0.5, 1.5] → 1, (1.5, 2.5] → 2, (2.5, 3.5] → 3, (3.5, +inf) → 4

We can optimize them using the training data: **Nelder-Mead method**

- Heuristic optimization, from a simplex of n+1 vertices, where n is the number of input parameters.
- **Function to be optimized**: the one that rounds the predictions according to the thresholds and computes the QWK value.
- Initial guess: naïve thresholds + another 4 near random points in the 4-D space to obtain the simplex.
- Iterative process (pre-defined number of iterations), new points are obtained through several ways:





Extracted from https://codesachin.wordpress.com/2016/01/16/nelder-mead-optimization/

Classification or Regression?

Regression

```
Average accuracy: 0.39411740760373537
Average QWK: 0.44870653969599034

Classification report:
              precision    recall  f1-score   support

           0       0.22      0.01      0.03       410
           1       0.42      0.33      0.37      3090
           2       0.34      0.39      0.36      4037
           3       0.27      0.36      0.31      3259
           4       0.61      0.50      0.55      4197

    accuracy                           0.39     14993
   macro avg       0.37      0.32      0.32     14993
weighted avg       0.41      0.39      0.40     14993
```

Classification

```
Average accuracy: 0.4545445465586596
Average QWK: 0.4314061862496381

Classification report:
              precision    recall  f1-score   support

           0       0.60      0.04      0.07       410
           1       0.42      0.40      0.41      3090
           2       0.38      0.40      0.39      4037
           3       0.40      0.28      0.33      3259
           4       0.55      0.73      0.63      4197

    accuracy                           0.45     14993
   macro avg       0.47      0.37      0.36     14993
weighted avg       0.45      0.45      0.44     14993
```

Moreover, the Nelder-Mead method takes even more time than fitting the regressor to the training data, so we cannot use the QWK metric in the hyperparameter search.

Thus, the hyperparameter search is done for classifiers.

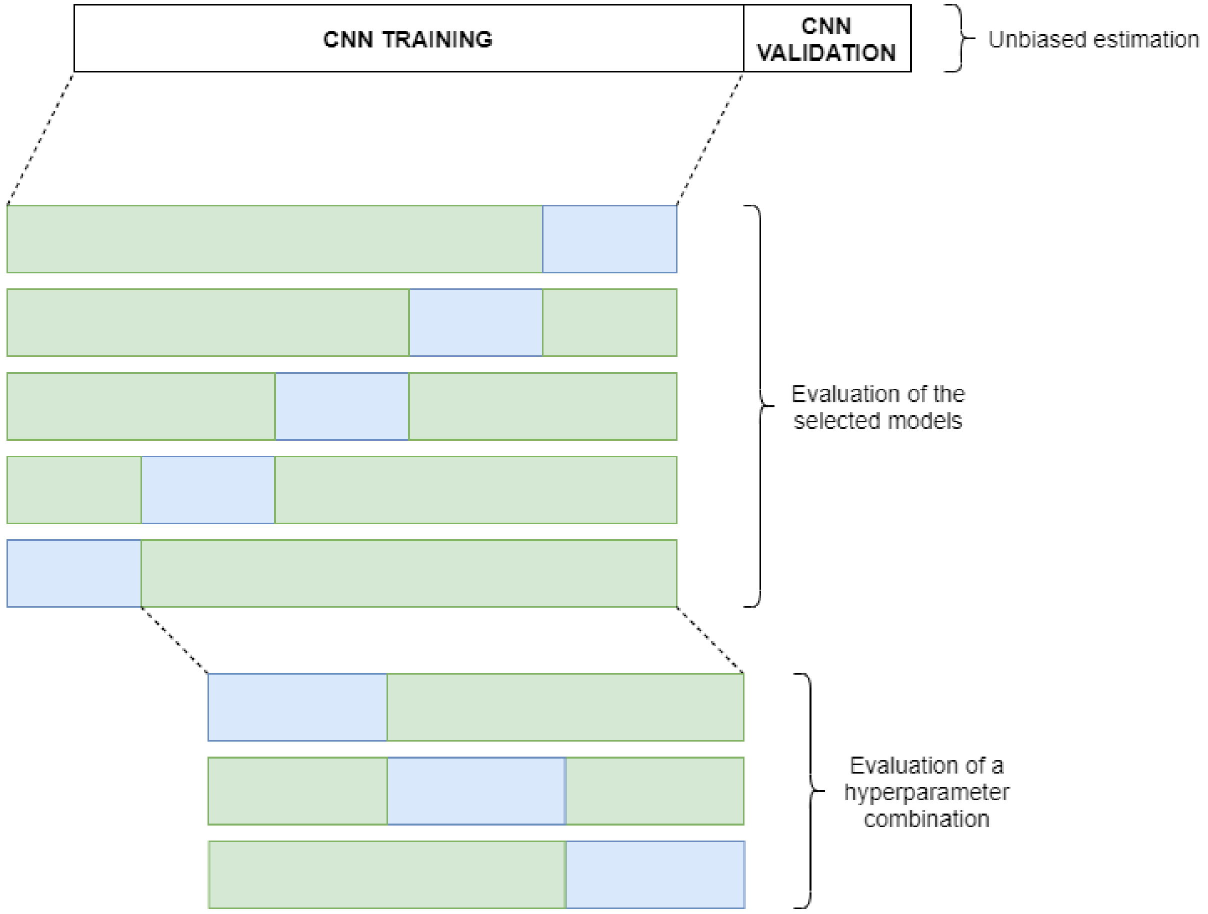
Hyperparameter search

We have to **decouple** the hyperparameter search from the evaluation of the selected model in that search.

Thus, inside the CNN training dataset, a **5-CV** process is conducted to evaluate the selected combination of hyperparameters. Inside each 5-CV iteration, a **3-CV** is conducted for the search, evaluating each combination of hyperparameters.

From the outer 5-CV, we can get different hyperparameter combinations: we select the one with the **highest average rank (QWK)** in the rest of searches (outer iterations).

Last filter: CNN validation dataset.



Hyperparameter search

Model	Avg. accuracy score	Avg. QWK score	S. split accuracy	S. split QWK
RandomForest	0.460147	0.447716	0.444481	0.412748
XGBoost	0.457562	0.440089	0.453151	0.419098
LightGBM	0.450976	0.436128	0.440480	0.397288

Majority vote and stacking ensembles didn't improve both metrics.

Thus, the final model is the pipeline with XGBoost and the following hyperparameters:

```
tfidf_vectorizer__sublinear_tf                          True
tfidf_vectorizer__ngram_range                         (1, 1)
tfidf_vectorizer__max_df                                 1.0
model__reg_lambda                                        5.0
model__n_estimators                                      120
model__max_depth                                           5
model__learning_rate                                     0.1
model__gamma                                             1.0
include_prof_im_properties__aggregate_properties        True
include_prof_im_metadata__aggregate_metadata           False
encode_breed__enc_type                           one-hot_svd
```

Final model

Two problems in the final XGBoost model (trained on the complete training dataset):

- Too high proportion of predicted ‘4’ cases in the test dataset

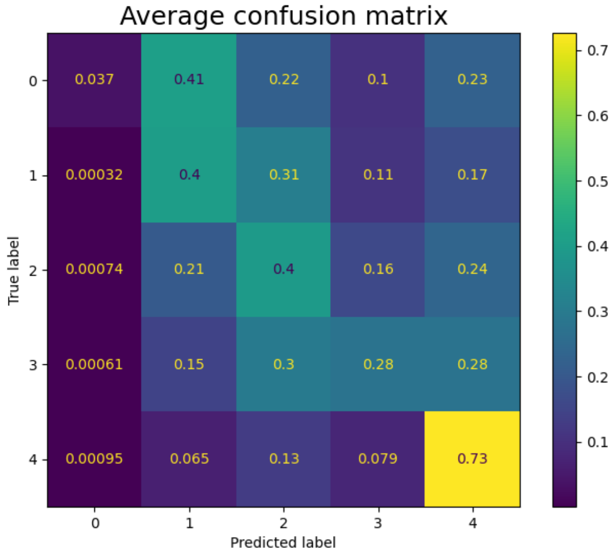
Classification

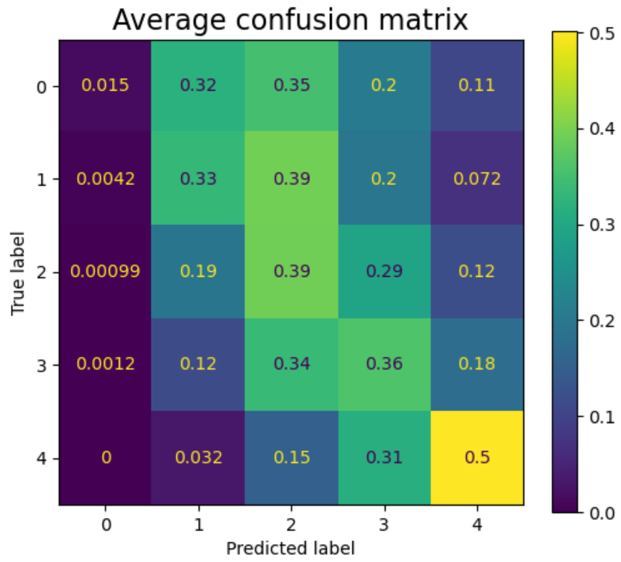
4	0.645267
3	0.038016
2	0.110020
1	0.206697

Regression

4	0.347684
3	0.319235
2	0.214502
1	0.118580

Training confusion matrices:

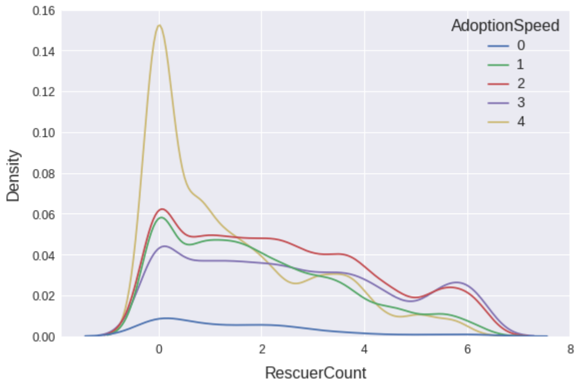


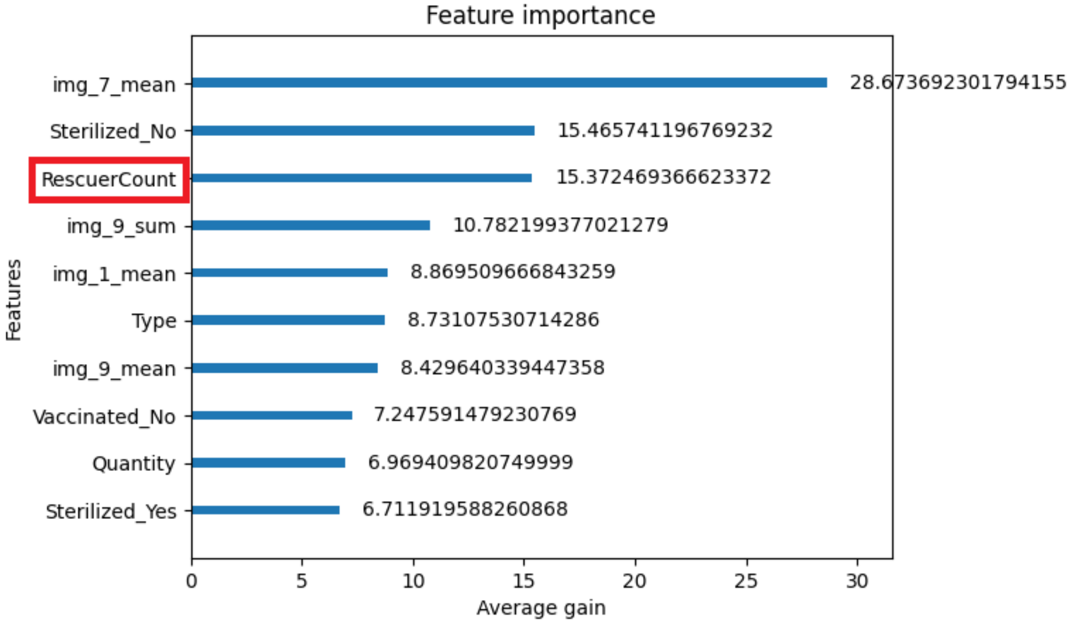


Final model

Test QWK score of the regression XGBoost pipeline with the selected hyperparameters: 0.36761 (we expected over 0.42…). This would be the final score.

- Second problem (looking at the test dataset): none of its *RescuerID* values coincide with the ones seen in the training dataset.
  - All of the test instances are assigned a *RescuerCount* value of 1.
  - Hence, the probability of being a '4' case increases significantly.





Test predictions of the XGBoost regressor trained without *RescuerCount* (QWK score of 0.39003):

```
4    0.285247
3    0.265861
2    0.248238
1    0.200655
```

Similar to the training distribution (possible influence of threshold optimization based on QWK).

Index

1. Introduction
2. Exploratory Data Analysis
3. Feature Engineering
4. Model creation and evaluation
5. Conclusions and future work

Conclusions

We have:

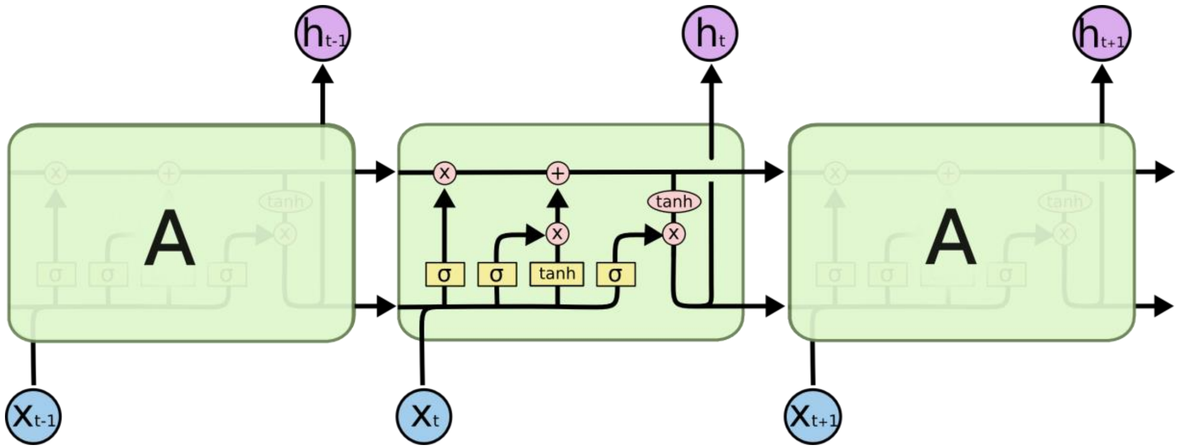
- conducted a data mining process in several stages,
- applied a variety of techniques to extract information and represent it in the most suitable way ([CM5]),
- used different state-of-the-art learning algorithms and paradigms ([CM7]),
- after understanding their fundamentals and how they work ([CM4]).

Yet:

- We have experienced how a minor decision (*RescuerCount*) can induce an optimistic estimation or just make our final model to not perform as we would like.
- The work can be improved.

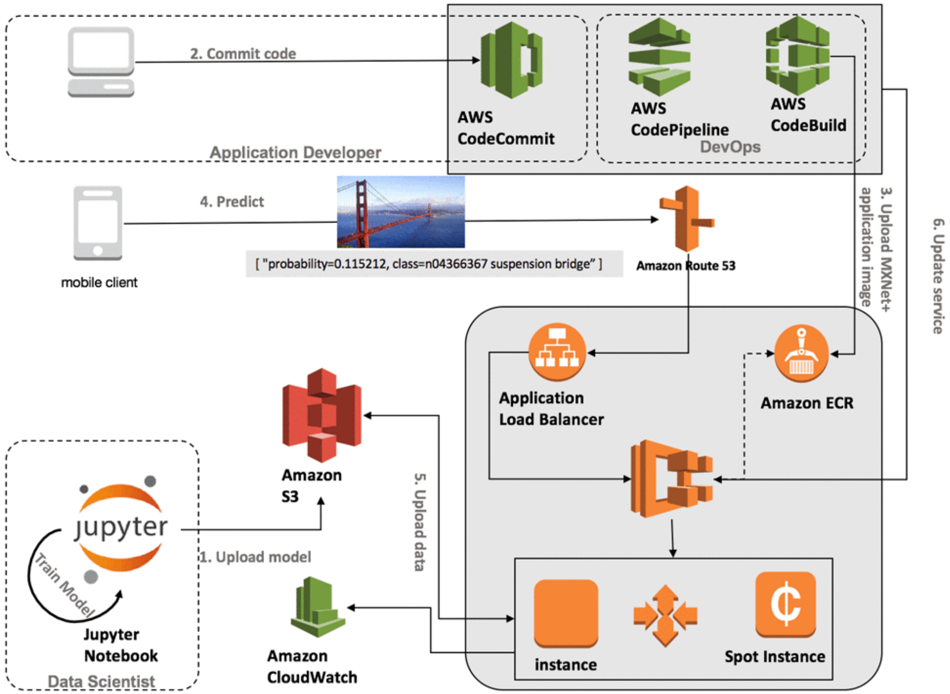
Future work

Use state-of-the-art text feature extraction models



Extracted from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Deploy the model in the cloud ([CM6])



Extracted from https://aws.amazon.com/es/blogs/machine-learning/deploy-deep-learning-models-on-amazon-ecs/

Source code

Notebook 1 (EDA):

https://www.kaggle.com/davidmora/tfg-pet-adoption-eda

Utilities of Notebook 1:

https://www.kaggle.com/davidmora/utils-tfg-pet-adoption-eda

Functions to extract image properties:

https://www.kaggle.com/shivamb/ideas-for-image-features-and-image-quality/notebook

Notebook 2 (Feature Engineering):

https://www.kaggle.com/davidmora/tfg-pet-adoption-fe-fss

Transformers extracted from Notebook 2:

https://www.kaggle.com/davidmora/transformers-tfg-pet-adoption

Nelder-Mead optimization to round predictions:

https://www.kaggle.com/c/petfinder-adoption-prediction/discussion/76107

Notebook 3 (hyperparameter tuning and final models):

https://www.kaggle.com/davidmora/tfg-pet-adoption-hpt-models

Generated data and intermediate results of all the stages:

https://www.kaggle.com/davidmora/tfg-pet-adoption-data

References

[Cao et al., 2019] Cao, W., Mirjalili, V., and Raschka, S. (2019). Consistent rank logits for ordinal regression with convolutional neural networks. *CoRR*, abs/1901.07884.

[Weiss et al., 2012] Weiss, E., Miller, K., Mohan-Gibbons, H., and Vela, C. (2012). Why did you choose this pet?: Adopters and pet selection preferences in five animal shelters in the united states. *Animals*, 2(2):144–159.

Images used in Slide 33:

- https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html
- https://cs231n.github.io/convolutional-networks/

Images used in Slide 36:

- https://neurohive.io/en/popular-networks/vgg16/
- Chollet, F. (2016). Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357.
- Huang, G., Liu, Z., and Weinberger, K. Q. (2016). Densely connected convolutional networks. *CoRR*, abs/1608.06993.
- Szegedy, C., Ioffe, S., and Vanhoucke, V. (2016). Inceptionv4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261.
- Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. (2017). Learning transferable architectures for scalable image recognition. *CoRR*, abs/1707.07012.

References

Images used in Slide 47:

- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space.
- Kusner, M. J., Sun, Y., Kolkin, N. I., and Weinberger, K. Q. (2015). From word embeddings to document distances. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning* - Volume 37, ICML'15, page 957–966. JMLR.org.

Images used in Slide 52:

- http://datahacker.rs/005-pytorch-logistic-regression-in-pytorch/
- http://albertotb.com/curso-ml-R/Rmd/07-svm/07-svm.html#1
- https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html
- Berriri, M., Djema, S., Rey, G., and Dartigues-Pallez, C. (2021). Multi-class assessment based on random forests. *Education Sciences*, 11(3).
- https://medium.com/swlh/gradient-boosting-trees-for-classification-a-beginners-guide-596b594a14ea

Deep Learning techniques applied to prediction from images. Use case: pet's adoption

David Mora Garrido

Undergraduate Dissertation

Bachelor’s Degree in Computer Science and Engineering, Major in Computing

29th July, 2021