Introduction to Machine Learning Project
Adaption of a Pet

Ufuk Cem Birbiri

uccbirbiri@gmail.com

Contents

1 Data Analysis

This report is about data analysis and machine learning prediction of an animal shelter where we predict if the animals in your possession can be adopted within 30 days or not.

The dataset contains information about cats and dogs with other features like age, gender, color, maturity size, fur length, vaccination, dewormed, sterilization, health, fee, description, breed or image. Training data has 8168 examples with 16 features. The Age and Fee columns have numeric type and the others have categorical variables as 'object' type.
Before applying machine learning algorithms, first we will do feature engineering and have a better understanding of the dataset and how features would affect the performance.

1.1 Type

The dataset has two types: Cat and Dog. Fig1 shows the number of cats and dogs in the training and testing datasets. We see that the rate of dogs in the both set is a bit higher. Though, it is not as much as we worry.

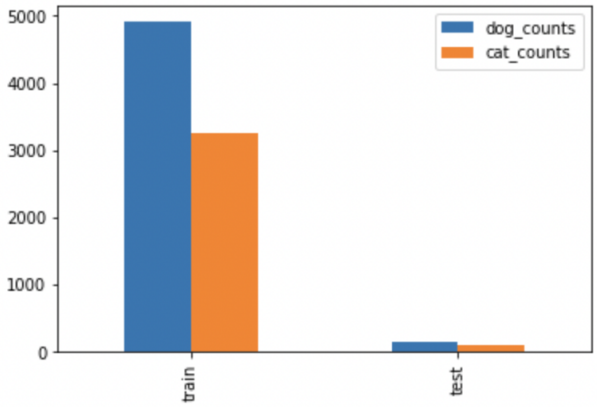


Figure 1: Number of cats and dogs in the train and test data.

Now let's analyse the proportion of how likely pets will get adopted in 30 days. 'True' means the pet will be adopted in 30 days and false is the opposite.

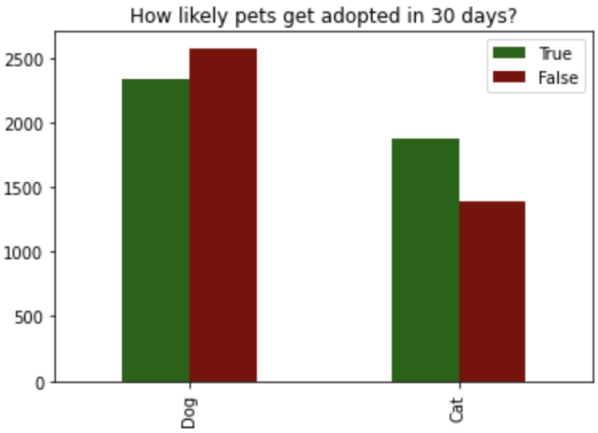


Figure 2: How likely pets will get adopted in 30 days according to Type feature.

We see that cats are more likely to be adopted than dogs because 'True' count is greater than 'False' count in cats.

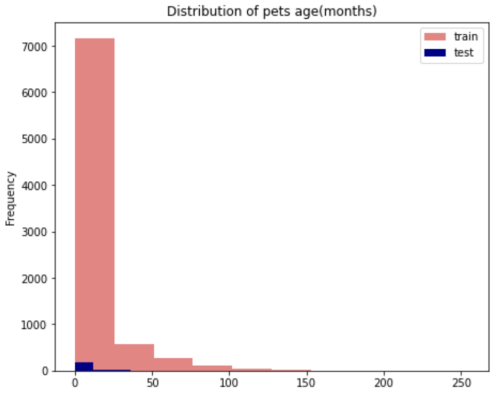
1.2 Age

The age is written in months. Fig3 shows the value counts of ages in testing and traning data. We can see that most pets are young. Interestingly, most ages are multiple of 12 that means people are not sure about pet's age and they indicated as multiples of 12.

```
train.Age.value_counts()
```

```
2.0      1769
3.0      1030
1.0      1004
4.0       618
12.0      599
24.0      407
5.0       337
6.0       336
36.0      261
8.0       199
Name: Age, dtype: int64
```

(a)



(b)

```
test.Age.value_counts()
```

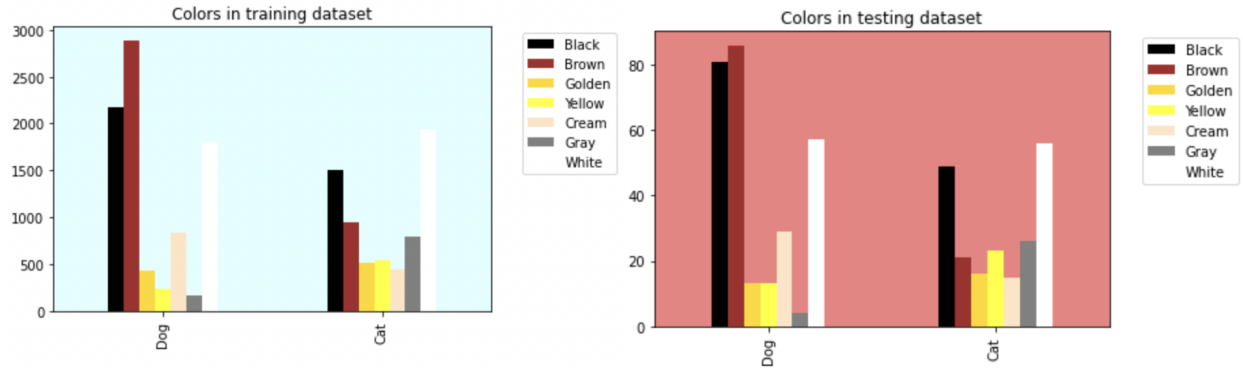
```
2.0      54
3.0      32
4.0      31
12.0     17
24.0     13
5.0      13
1.0      12
6.0      11
8.0       8
7.0       6
Name: Age, dtype: int64
```

(c)

Figure 3: Value counts of ages in months: (a) and (c). The Age distribution in train and test set: (b).

1.3 Color

We have 3 color columns in data dataset and 7 different colors: White, Cream, Gray, Golden, Yellow, Black, Brown. If the value is 'Unknow', it means there is no color.



(a) (b)

Figure 4: Color distribution for train and test data respectively.

We see that brown, white and black are most popular colors in both cats and dogs.

IDEA: If the Color1, Color2 and Color3 are correlated with each other, we can remove the most correlated one. Let's test this idea and see the correlation matrix of colors:

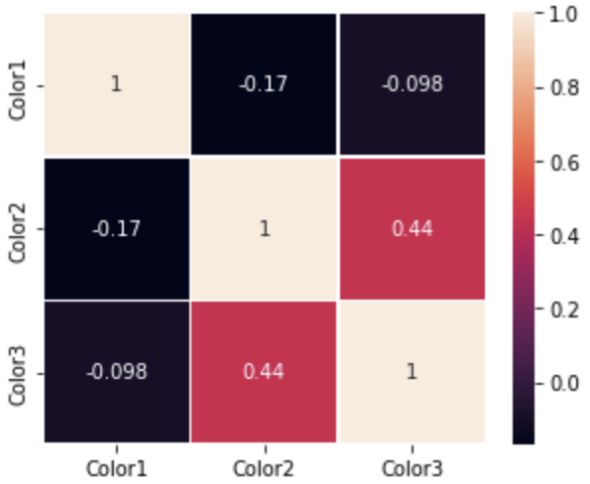


Figure 5: Correlation matrix of colors.

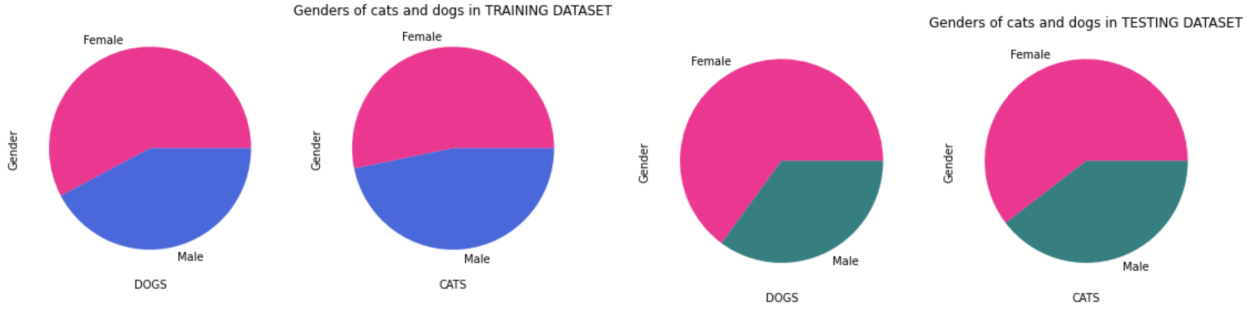
Color2 and Color3 are correlated as 0.44 but that is not enough. We should get at least 0.80 for a strong correlation. Colors are not correlated with each other as I expected so, we cannot remove any of them.

Adding Color Count as a feature

Maybe there is a relationship between color count a pet has and adaptation. To apply this idea, I count the number of colors f or each pet and added to the dataset as a new feature. "color_count_transformer" does this job in the pipeline.

1.4 Gender

Make a quick analysize for Gender. Fig 6 shows the distribution of genders in cats and dogs in training and testing sets.



(a) (b)

Figure 6

Females and males are almost equal in cats and there are more female in dogs but not in a problematic way in training dataset. There are more female in both cats and dogs in testing set also.

1.5 Vaccinated, Dewormed, Sterilized and Health

Vaccination status, being dewormed,sterilization and health are important features for adoption. They can be analysed together since they are about pet's health situation. Let's look at the correlation matrix of these

features in Fig 7

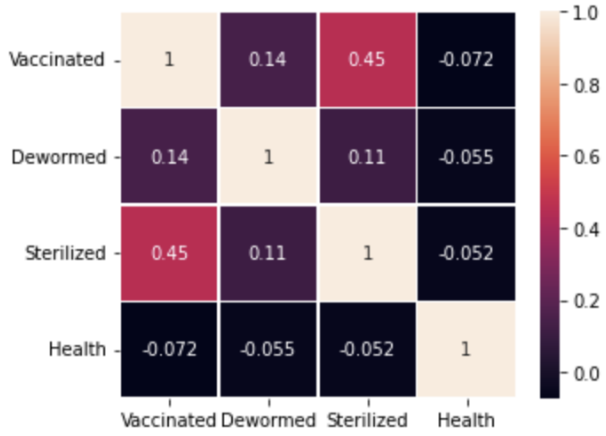
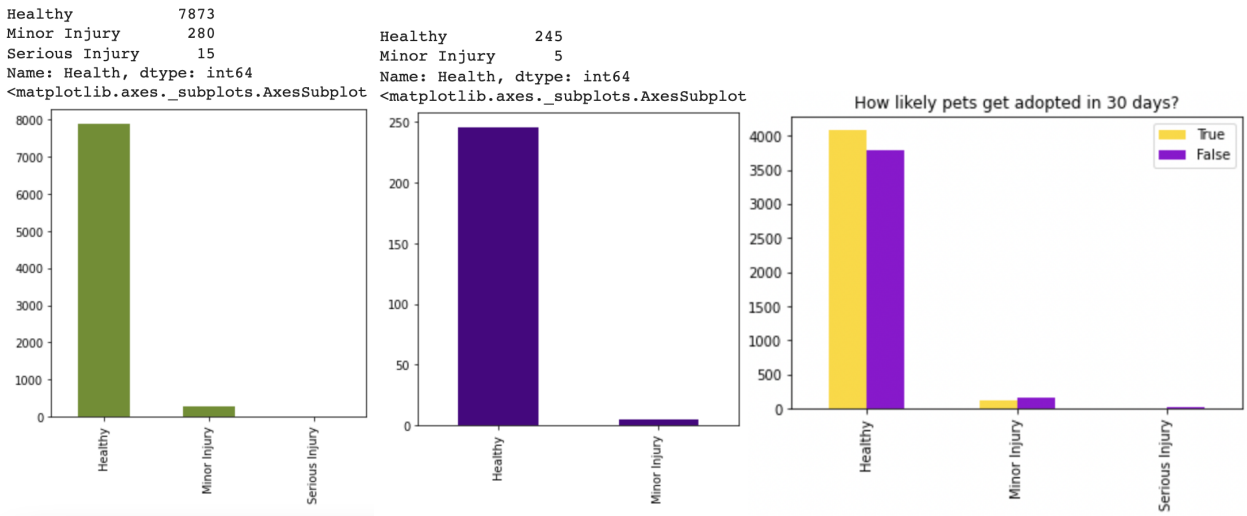


Figure 7: Correlation matrix of Vaccinated, Dewormed, Sterilized and Health columns in the training data.

They are not correlated with each other at all. Vaccination and Sterilized features are correlated as 0.45 however this is not enough for a strong correlation.

There are three options in Health column: Healthy, Minor Injury and Serious Injury. Now let's see the 'Health' column distribution in training and testing datasets.



(a) (b) (c)

Figure 8: 'Health' column distribution in training(a) and testing(b) sets. The third bar plot(c) shows the relationship between health condition and the adaptation.

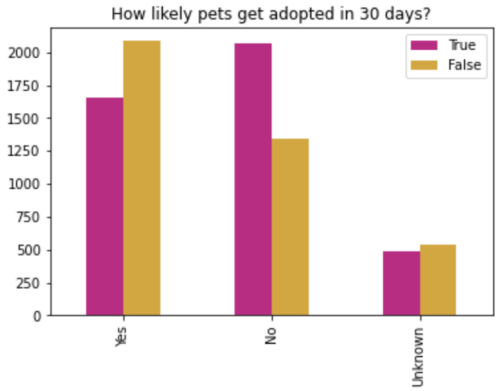
Most pets are healthy in the training dataset. There is no 'Serious Injury' category in test data. It is not OK because we must have data points belonging to every class in testing set. If the testing set does not represent every class label, then we could have problems in accuracy score and performance. Results show that if a pet is healthy, there is no guarantee that it will get adopted.

Let's make a quick analysis about vaccination. There are 3 types in this column: Unknown, No, Yes.

```
train.Vaccinated.value_counts()
```

```
Yes         3742
No          3409
Unknown     1017
Name: Vaccinated, dtype: int64
```

(a)



(b)

Figure 9: (a) shows the vaccination status of pets in the training data. (b) shows the vaccination status vs adaptation possibility relation.

According to Fig 9, it is interesting to see that people prefer not-vaccinated pets more. Maybe they want to start vaccination from zero and take them to vets themselves.

1.6 Breed

Let's visualize the breed names with a word cloud and see the popular breeds in the dataset.

```
train.Breed.value_counts().head(10)
```

```
Mixed_Breed             3437
Domestic_Short_Hair     1694
Domestic_Medium_Hair     599
Tabby                    174
Siamese                  159
Shih_Tzu                 132
Domestic_Long_Hair       129
Labrador_Retriever       129
Persian                  124
Poodle                   118
Name: Breed, dtype: int64
```

(a)

(b)

Figure 10: Breed counts in training data is on the left. Word count representation is on the right.

Fig 10 shows the breed count and word cloud plot in training data. If a word is bigger in the word cloud, its frequency is greater than others. Breed type varies among pets and most of the pets have 'Mixed Breed' because it has the biggest size in word cloud.

1.7 Fee

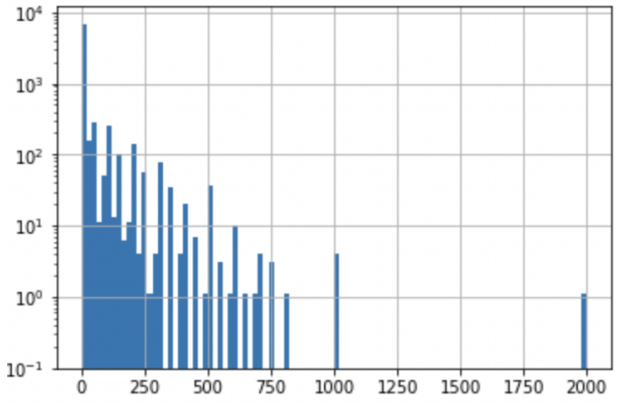
Fig 14 shows the fee of animals in the training data.

Most of the pets are free, and there are also expensive pets.

```
train.Fee.value_counts().head(10)
```

```
0.0      6754
50.0      265
100.0     253
200.0     140
150.0     103
20.0       98
300.0      76
250.0      55
30.0       54
80.0       42
Name: Fee, dtype: int64
```

(a)



(b)

Figure 11: Fee of animals. Counts are on left and log scaled bar plot is on right.

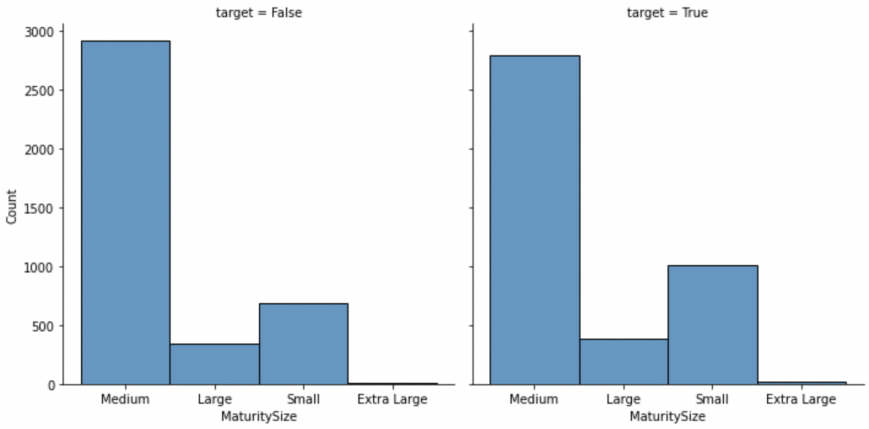
1.8 Maturity Size

Fig 12 shows the maturity size counts and its affect to adaptation.

```
train.MaturitySize.value_counts()
```

```
Medium         5711
Small          1700
Large           736
Extra Large      21
Name: MaturitySize, dtype: int64
```

(a)



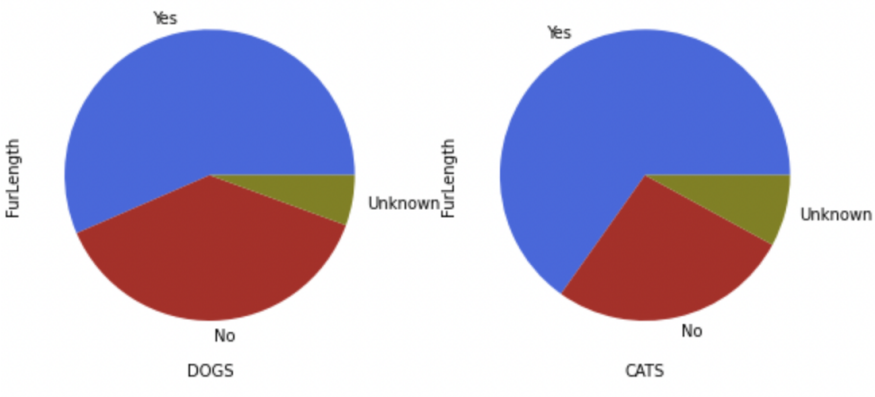
(b)

Figure 12: Maturity size of animals in the training data is shown. Value counts are on left side. The adaptation possibility and maturity size relationship is on the right hand side.

According to the two bar plots in the image, maturity size does not have a huge effect on adaptation since the 'Medium', 'Large', 'Small' and 'Extra Large' value counts are close to each other for both target class.

1.9 Fur Length

Interestingly, the fur length column does not have lengths, it has 'Yes' , 'No' and 'Unknown' types. I am not sure about what 'FurLength' represents because I expect it to be a real length in the metric centimeters. Let's investigate the distribution of this feature. Fig 13 shows the distribution of fur length for cats and dogs in training data.



(a)

Figure 13: Furlength distribution of cats and dogs in training data

The answers for fur length have close distributions in cats and dogs.

1.10 Description Column

This feature has a text description of the pet written by its owner. Since this feature is not categorical or numerical feature, we need to make a different analysis. Here, I did two different analysis. The first one is counting the number of words in the description. The second one is to understand whether the description is positive, negative or neutral.

1. NUMBER OF WORDS IN THE DESCRIPTION

Maybe there is a relationship between the number of words in the description and adaptation possibility. To test this, I created another columns and add it to the dataset as a new feature. This columns has numeric type and includes the number of words in each description. I added this process to the pipeline, so when someone runs my model and pipeline, it automatically adds this new column to the dataset. In the pipeline, the "word_count_transformer" makes this job.

2. SENTIMENT ANALYSIS

Step 1: Cleaning the text

In this step, we need to remove the special characters and numbers from the text. We can use the regular expression operations library of Python.

Step 2:

This step has various phases:

- Stopwords removal: Stopwords in English are words that carry very little useful information. We need to remove them as part of text preprocessing. nltk has a list of stopwords of every language.
- Tokenization is the process of breaking the text into smaller pieces called Tokens. It can be performed at sentences(sentence tokenization) or word level(word tokenization). I will be performing word-level tokenization using nltk tokenize function word_tokenize().
- Parts of Speech (POS) tagging is a process of converting each token into a tuple having the form (word, tag). POS tagging essential to preserve the context of the word and is essential for Lemmatization. This can be achieved by using the nltk pos_tag function.

Step 3: Lemmaziation I used WordNetLemmatizer in nltk library for lemmaziation. Fig 10 shows examples of these steps and the final form of the description.

	Description	Cleaned Desc	POS tagged	Lemma
0	We got Luna when she was a kitten in Feb 15'. ...	We got Luna when she was a kitten in Feb She i...	[(got, v), (Luna, n), (kitten, n), (Feb, n), (...	get Luna kitten Feb calm friendly cat gentle...
1	Ginger Boy was found starving and hungry so I ...	Ginger Boy was found starving and hungry so I ...	[(Ginger, n), (Boy, n), (found, v), (starving,...	Ginger Boy find starving hungry start feed r...
2	An indoor cat with nice green/ yellowish eyes....	An indoor cat with nice green yellowish eyes A...	[(indoor, a), (cat, n), (nice, a), (green, a),...	indoor cat nice green yellowish eye first lo...
3	My dog name called boo. He is a male. I feedin...	My dog name called boo He is a male I feeding ...	[(dog, n), (name, n), (called, v), (boo, n), (...	dog name call boo male feed around year
4	1) Foxy is a stray cat which I feed regularly,...	Foxy is a stray cat which I feed regularly sh...	[(Foxy, n), (stray, a), (cat, n), (feed, v), (...	Foxy stray cat fee regularly spay July spay ...

(a)

Figure 14: The sentiment analysis steps and the final result of description.

a. Sentiment Analysis using TextBlob: TextBlob is a Python library for processing textual data. It provides a consistent API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, and more.

TextBlob gives us the Polarity and Subjectivity values:

- Polarity is about how positive or negative or neutral the text is. It ranges from -1 to 1 (1 is more positive, 0 is neutral, -1 is more negative)
- Subjectivity is about how subjective the text is. It ranges from 0 to 1 (0 being very objective and 1 being very subjective)

b. Sentiment Analysis using VADER: VADER stands for Valence Aware Dictionary and Sentiment Reasoner. Vader sentiment not only tells if the statement is positive or negative along with the intensity of emotion.

The sum of positive, negative, neutral intensities gives 1. Compound ranges from -1 to 1 and is the metric used to draw the overall sentiment.

- positive if compound ≥ 0.5
- neutral if -0.5 $<$ compound $<$ 0.5
- negative if -0.5 $\geq$ compound

c. Sentiment Analysis using SentiWordNet: SentiWordNet uses the WordNet database. It is important to obtain the POS, lemma of each word. We will then use the lemma, POS to obtain the synonym sets(synsets). We then obtain the positive, negative, objective scores for all the possible synsets or the very first synset and label the text.

- if positive score $>$ negative score, the sentiment is positive
- if positive score $<$ negative score, the sentiment is negative
- if positive score $=$ negative score, the sentiment is neutral

Fig 15 shows the sentiment analysis results as a visualization. The first, second and third columns belong to TextBlob analysis. The fourth and fifth columns belong to Vader analysis and the last column is for SentiWordNet analysis.

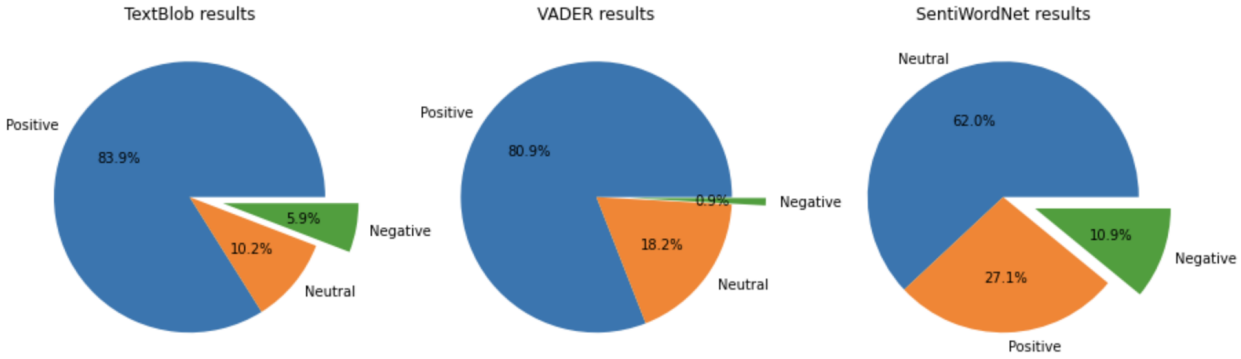
Now let's see the results of these sentiment analysis methods in pie charts. These results are done for training dataset description column.

According to Fig 16, SentiWordNet analized most of the words as neutral. VADER analysis has the least negative results for descriptions.

	Subjectivity	Polarity	Analysis	Vader Sentiment	Vader Analysis	SWN analysis
0	0.733333	0.179167	Positive	0.8271	Positive	Positive
1	0.570833	0.180587	Positive	0.9382	Positive	Neutral
2	0.475000	0.066667	Positive	0.7845	Positive	Positive
3	0.100000	0.000000	Neutral	0.0000	Neutral	Neutral
4	0.225641	-0.166667	Negative	-0.0258	Neutral	Negative

(a)

Figure 15: After the sentiment analysis methods, the final data frame looks like this. 'Subjectivity', 'Polarity' and 'Analysis' columns belong to TextBlob method. 'Vader Sentimet' and 'Vader Analysis' columns belong to Vader method. 'SWN analysis' column belongs to SentiWordNet method.



(a)

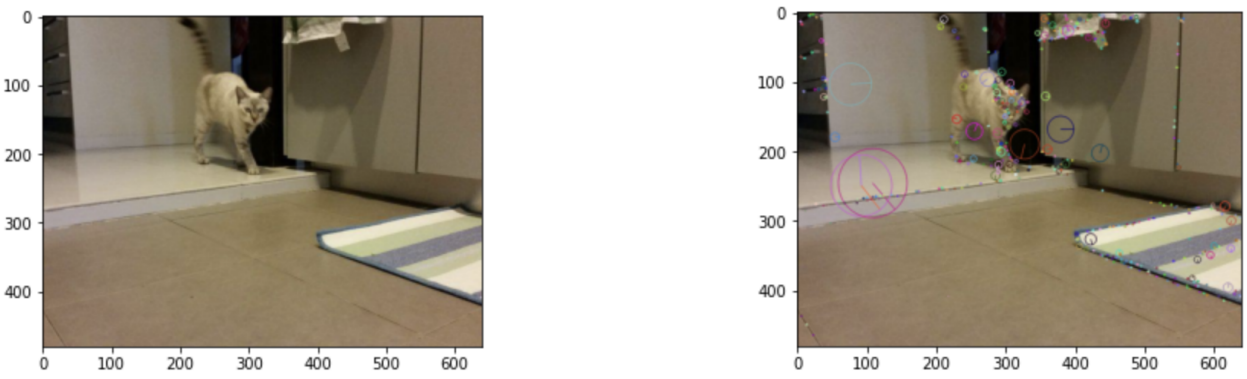
Figure 16: Sentiment analysis results for training data description column.

I used these analysis and add them to the dataset separately. "text_transformer" is used in the pipeline to transform the description column to sentiment analysis results.

For TextBlob the 'Subjectivity' and 'Polarity' columns are added to the training dataset as numeric columns. For the VADER analysis, only 'Vader Analysis' column added to the training dataset as categorical column. For the SentiWordNet analysis, 'SWN analysis' column is used and added to the traning dataset as categorical column. Then both VADER and SWN analysis results are converted to numeric values with OrdinalEncoder function in sklearn.preprocessing library. Again, these sentiment analysis methods were kept as hyper-parameters and I tested their effects on accuracy results separately.

1.11 Image Column

I used SIFT algorithm to create Bag of Features(BOF). It was already implemented and I didn't change anything on this implementation. It took a lot of time to extract features from all of images in the training data. Fig 17 shows the raw image and features after SIFT detector.



(a) (b)

Figure 17: .

2 Machine Learning Part

Before showing results, let me talk about the pipeline.

2.1 Pipeline

Different methods are used in the pipeline considering the columns types. Column types of the dataset and the processing types are given in the following.

- **Nominal columns:** Type, Gender, Breed, Color1, Color2, Color3, FurLength, Vaccinated, Dewormed, Sterilized, Health. Processed by OneHotEncoder in sklearn.preprocessing.
- **Ordinal columns:** MaturitySize. Processed by OrdinalEncoder in sklearn.preprocessing
- **Numerical columns:** Age, Fee. Processed by StandardScaler in sklearn.preprocessing
- **Text columns:** Description. Processed by "text_transformer" that is done with sentiment analysis.
- **Image columns:** Image. Processed by SIFT and BOF implementation.

Description column also processed by "word_count_transformer" that add the number of word counts in the description to the dataset. Color columns are also processed by "color_count_transformer" in order to add the number of colors in each pet to the dataset.
Each process is done in a different pipeline and with a different transformer. ColumnTransformer function in sklearn.compose is used to concatenate the pipelines. Fig 18 shows the image of the pipeline.

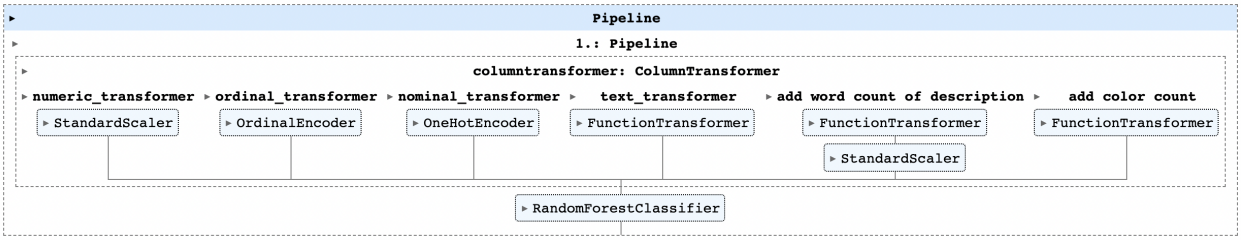


Figure 18: My pipeline

2.2 Dimension Reduction Techniques

I applied two methods: PCA and Feature Agglomeration.
PCA is a linear dimension reduction method that uses Singular Value Decomposition(SVD) to project the data to a lower dimension. PCA centers the input data but it does not scale for each feature before applying the SVD. On the other hand, feature agglomeration merges pair of clusters of features recursively try to group similar features. I applied both of them at the end of the pipeline.
The shape of the traning dataset before and after dimension reduction methods are:

BEFORE PCA X_train.shape= (8168, 199)
AFTER PCA X_train.shape= (8168, 28)

BEFORE Feature Agglomeration X_train.shape= (8168, 198)
AFTER Feature Agglomeration X_train.shape= (8168, 54)

2.3 Classifiers

I will first mention about the tested hyper-parameters and the best hyper-parameter group in each classifier. Then show the results of best hyper-parameters.

2.3.1 Random Forest

Random Forest is tested with the following hyper-parameters:

- n_estimators: [200, 500]
- max_features: 'auto', 'sqrt', 'log2'
- max_depth : 4, 5, 10, 20
- criterion : 'gini', 'entropy'
- Dimension reduction: PCA, Feature Agglomeration
- Sentiment Analysis: Textblob, Vader Analysis, SWN Analysis

The best hyper-parameters are max_depth=20, max_features = 'sqrt', n_estimators=500, 'criterion': 'gini', sentiment analysis = Textblob without PCA or Feature Agglomeration. The highest test accuracy was 0.672
The training accuracy was 0.845.

NOTE: I tested sentiment analysis methods with random forest and its best hyper-parameters. The results are:
Textblob test accuracy = 0.672
VADER Analysis test score = 0.664
SWN Analysis test score = 0.648

According to results, Textblob got the highest scores. So, I continued my experiments in other classifiers with Textblob sentiment analysis.

2.3.2 Logistic Regression

Logistic regression is tested with the following hyper-parameters:

- C: 0.01, 0.1, 1.0, 10, 100
- solver : newton-cg, lbfgs, liblinear
- penalty : l1, l2
- Dimension reduction: PCA, Feature Agglomeration
- Sentiment Analysis: Textblob

The best hyper-parameters are C=1, penalty= l1 , solver= liblinear, sentiment analysis = Textblob, Dimension reduction= PCA.

The highest accuracy on test set is 0.592
The accuracy on train set was 0.631.

2.3.3 Decision Tree

Decision Tree is tested with the following hyper-parameters:

- criterion: gini, entropy
- max_features: auto, sqrt, log2
- max_depth : 5, 10, 20, 50, 100, 200
- Dimension reduction: PCA, Feature Agglomeration
- Sentiment Analysis: Textblob

The best hyper-parameters are criterion: gini, max_features: sqrt,max_depth = 200, sentiment analysis = Textblob, Dimension reduction = PCA.

The highest accuracy on test set is 0.631
The accuracy on train set was 0.669

2.3.4 SVC

SVC is tested with the following hyper-parameters:

- C: 0.1, 1.0,10,100
- kernel: linear, poly, rbf, sigmoid

The best hyper-parameters are C = 1 and kernel = rbf.

The highest accuracy on test set is 0.52
The accuracy on train set was 0.681

2.3.5 K-Means

KMeans is tested with the following hyper-parameters:

- max_iter: 300, 500
- n_clusters: 2
- init: k-means++, random

The best hyper-parameters are max_iter = 500, n_clusters=2 and init = k-means++.

The highest accuracy on test set is 0.364
The accuracy on train set was 0.38
KMeans is very unefficient for this classification problem.