Midterm: Who's the Top Dog?

Directions

This assessment is split into two sections. The first sections pertains to a regression problem, and the second pertains to a classification problem. Each section *starts* with an overarching question prompt (like "**What are the most important factors governing the popularity of a dog breed?**") which should guide your investigation throughout each part.

Any question that has "(**challenge**)" after it is *optional*. That is, you can pass the assessment *without* attempting these sections. They may require you to extend beyond explicit taught material, or think in ways that may be challenging or unusual. **I repeat: these questions are optional, and you do not need to attempt them to pass the assessment**. If you do attempt them and do something incorrect or otherwise erroneous, this will not harm you. However, if you want to attempt to reach into the higher ranges of marks for the course, you would need to attempt the challenge questions.

Framing

Many of you probably thought that, in this course, "DS" meant "Data Science." This is a natural assumption, but you would be wrong. Today, it stands for "Dog Science."

The following dataset contains data used by editors and advertisers of the *Modern Dog* quarterly magazine on the different traits and popularity of dog breeds. You've been hired as a data sleuth[1] for a competitor magazine, *Postmodern Dog*, and are tasked with developing a **statistically-justified** explanation for how the editorial board at *Modern Dog* has rated its dog breeds. Each section below will pose one overarching question about the data, and ask a few intermediate questions to help you along the way.

```
import pandas
import seaborn
```

[1] another good "DS" role `r emo::ji("grin")`

```
dogs = pandas.read_csv("./midterm-data.csv")
```

Just to show all the columns you're working with, I'm going to show you the *transpose* of the first two rows.

```
dogs.head(2).T
```

	0	1
affection	5	5
good_with_young_children	3	5
good_with_other_dogs	3	5
shedding	3	5
grooming_needs	1	3
drool	1	3
coat_type	Smooth	Double
coat_length	Short	Medium
openness_to_strangers	3	4
playfulness	4	4
protectiveness	4	3
adaptability	3	4
trainability_level	5	4
energy	5	4
barking	3	3
mental_stimulation_needs	4	3
breed_clean	German Pinschers	Bernese Mountain Dogs
popularity	0.79265	0.922924
editors_choice	1	1

I'll explain the variables here to give you a sense of what the data represents.

- `affection`: integer score from 1 (aloof) to 5 (loving) showing how affectionate dogs in this breed tend to be
- `good_with_children`: integer score from 1 (bad) to 5 (good) showing how good dogs in this breed tend to be with children
- `good_with_other_dogs`: integer score from 1 (bad) to 5 (good) showing how well-behaved dogs in this breed tend to be around other dogs.
- `shedding`: integer score from 1 (sheds little) to 5 (sheds a ton)
- `grooming_needs`: integer score from 1 (needs very little grooming) to 5 (needs a lot of grooming)
- `drool`: integer score from 1 (has no spit) to 5 (leaks like a faucet)

- `coat_type`: categorical variable describing the typical texture of the breed's hair. Can occupy values: Smooth, Double, Wiry, Corded, Curly, Hairless, Wavy, Silky, or Rough.
- `coat_length`: categorical variable describing the typical length of the breed's hair. Can occupy values: Short, Medium, or Long.
- `openness_to_strangers`: integer score from 1 (oppositional) to 5 (would go home with a stranger from the dogpark)
- `playfulness`: integer score from 1 (very serious) to 5 (very playful)
- `protectiveness`: integer score from 1 (completely unterritorial) to 5 (strongly protective of its things)
- `adaptability`: integer score from 1 (needs things always the same) to 5 (can deal with variety)
- `trainability_level`: integer score from 1 (can't learn new things) to 5 (can easily learn new things)
- `energy`: integer score from 1 (a sluggish lump) to 5 (zooming around all day)
- `barking`: integer score from 1 (won't bark) to 5 (is probably barking right now)
- `mental_stimulation_needs`: integer score from 1 (entertained watching paint dry) to 5 (needs a rubiks cube)
- `breed_clean`: string describing the name of the breed, unique for each row, but has patterns (e.g. "Terrier" or "Spaniel" is in every Terrier-type or Spaniel-type dog.)
- `popularity`: continuous value from about .3 to 1 describing how popular the dog breed is.
- `editors_choice`: a binary value, one if the breed has been selected as an Editor's choice breed since 2000, and zero otherwise.

Regression

What are the most important factors governing the popularity of a dog breed?

Regression 1a

Split the data into a test and a train dataset. Fit a linear regression to predict the `popularity` of a dog breed using factors you think will be useful to predict the popularity of the dog breed.[2] **Do not** use the `coat_type` or `coat_length` features. Also, definitely not a good idea to use the `breed_clean` feature!

Regression 1b

What are the most useful features to predict `popularity` in this model? How can you tell?

[2]Include interaction terms if you think they will be useful, but this is not necessary.

Regression 1c

Fit a linear regression to this dataset using only the useful features you identified in **Regression 1b**. What is the out-of-sample mean squared error and R^2?

Regression 1d (challenge)

We have talked about how models may be unbiased on average, but won't always predict groups correctly. Make a new feature called `is_terrier`:

```
dogs['is_terrier'] = dogs.breed_clean.str.contains("Terrier").astype(int)
```

With this new feature, can you tell me if our model predictions are biased against Terrier dogs? Include this new feature in a new separate linear regression model for this question.[3] *Conceptually*, why might we expect to see a change between the two models? *Practically*, is this *new* model biased against Terrier dogs?

Regression 2a

Fit a Random Forest to this dataset using only the useful features from **Regression 1b**. What is the Random Forest's out of sample mean squared error and R^2?

Regression 2b

Are the "good" predictors in this random forest the same as in the linear model?

Regression 2c (challenge)

Is this regression forest biased against terriers?

Regression 3a

Conceptually (before actually fitting it), why might K-Nearest Neighbors be less helpful for this research question? What aspects of the *dataset* make a KNN learner less useful here? What aspects of the *question* make a KNN learner less useful here?

[3] i.e., don't use this feature again in the lab!

Regression 3b

Fit a 10-NN model to this dataset using only the useful features from **Regression 1b**. What is the KNN's out-of-sample mean squared error and R^2?

Regression 3c (challenge)

Can you improve upon the value of k from the preceeding question? Try using a *weighted KNN* (the `weights` option), in addition to increasing the value of k.

Regression 4

Given your answers above, **What are the most important features governing the popularity of a dog breed?** Given your model scores, do you think your *Postmodern Dog* model is good at predicting the popularity scores from *Modern Dog*?

Classification

How can you tell if a dog is an "Editor's choice?"

Classification 1a

Fit a single decision tree with at least 10 dog breeds at each leaf. Use all of the data and features from **Regression 1b** and *also* include the `popularity` feature, too.

Classification 1b

Make a plot of this decision tree using the `plot_tree()` function we have used before in the lab. Describe one of the paths to the bottom of the tree in plain language. Save the tree to a file with high resolution[4] if you have to, so that it can be read in detail.

[4] using something like `dpi=300` in the `plt.savefig()` function

Classification 1c

Leave one third (33%) of the data out as a test dataset, and train a tree with the same parameters on the remaining data using the same features from **Classification 1a**.[5] What is the out of sample confusion matrix for our new decision tree?

Classification 1d

Create a second train/test split from the data in the same was as **Classification 1c** and train a new decision tree. How different is *this* tree's out of sample confusion matrix from the *other* decision tree's out of sample confusion matrix?

Classification 1e (challenge)

In questions & lab, we have talked about model criticism. Here, we'll do some for a classifier by computing the *average* `popularity` *for observations in each quadrant of the confusion matrix!*.

So, create a new column in `dogs` containing the prediction from your decision tree in **Classification 1d**. Then, use a `.groupby(...)` to group the data on *both* the observed `editors_choice` and the predicted value. Then, compute the average `popularity` of dog breeds in each group.

What is the average `popularity` of dog breeds in each `confusion_class`? *Conceptually*, what does this tell you about our decision tree's *flexibility* as a learner?

Classification 2a

Fit a logistic regression on the same train-test split from **Classification 1d**. How does its out-of-sample confusion matrix compare to the decision tree from **Classification 1d**?

[5] Remember: You can use the built-in `sklearn` function for `train_test_split(...)`, or you can use `dogs.sample(...)`. But, this will be slightly different every time you run it, unless you use the `random_state=` option in `sklearn` or use `numpy.random.seed(...)` before `.sample()`.

Classification 2b (challenge)

In the labs, we discussed how to use the Bootstrap to construct confidence intervals. Use a bootstrap to construct 1000 "fake" logistic regressions using the training data from before, and evaluate their performance on the real test dataset. How does the model accuracy[6] for the logistic regression **Classification 2a** compare to the distribution of bootstrapped accuracies? *Conceptually*, why might we expect this? Make a plot to visualise, if you think that will be useful for your explanation.

Classification 3

Fit a Random Forest to predict `editors_choice` using the same features across the same train-test split from **Classification 1d**. How does its out-of-sample confusion matrix compare to the previous models?

Classification 4a

Given the confusion matrices above, which model would you prefer and why? Check to see that your conclusion holds by using `cross_val_score()` to estimate the accuracy across *all* of the three-fold splits.

Classification 4b (challenge)

Check to see that your conclusion holds by using `cross_val_score()` to estimate the accuracy of your top two methods across *all* of the three-fold splits.

[6] that is, correct predictions divided by total predictions! You can use `sklearn.metrics.accuracy_score()` if you like.