SCALE FOR PROJECT FT_SOMMELIER (/PROJECTS/FT_SOMMELIER)

You should evaluate 1 student in this team

★

Git repository

`vogsphere@vgs.42.us.or` 📋

Introduction

To ensure this evaluation goes smoothly, please respect the following set of rules :

- Please remain courteous, polite, respectful, and constructive at all times during this exchange. The trust bond between the school's community and yourself depends on it.

- Should you notice any malfunctions within the submitted project, make sure you take the time to discuss them with the student (or group of students) being graded.

- Keep in mind that some subjects can be interpreted differently. If you come accross a situation where the student you're grading has interpreted the subject differently than you, try and judge fairly whether their interpretation is acceptable or not, and grade them accordingly. Our peer-evaluation system can only work if you both take it seriously.

Guidelines

- You may only evaluate whatever is in the GiT submission directory of the student you are grading.

- Make sure to check that the GiT submission directory belongs to the student (or group) you're grading, and that it's the right project.

- Make sure no mischievous aliases have been used to trick you
into correcting something that is not actually in the official
submitted directory.

- Any script created to make this evaluation session easier - whether
it was produced by you or the student being graded - must be
checked rigorously in order to avoid bad surprises.

- If the student who is grading this project hasn't done the
project him/herself yet, he/she must read the whole topic
before starting the evaluation session.

- Use the flags available to you on this scale in order to report
a submission directory that is empty, non-functional, that contains
norm errors or a case of cheating, etc...
In this case, the evaluation session ends and the final grade is 0
(or -42, in the case of cheating). However, unless the student has
cheated, we advise you to go through the project together in order
for the two (or more) of you to identify the problems that may have
led this project to fail, and avoid repeating those mistakes for
future projects.

Attachments

Subject (https://cdn.intra.42.fr/pdf/pdf/1391/ft_sommelier.en.pdf)

resources (/uploads/document/document/467/resources.zip)

Mandatory Part

General Instructions

Does the python notebook follow the project general instructions?

- Check that no improper modules were used by searching for lines that
start with 'import' or 'from x import y'. As a reminder, high level
modules like numpy, scipy, scikit-learn, tensorflow, Keras, theano,
etc.. are NOT allowed!!

- Throughout the correction, ensure that any matrix math operations
(if they exist) are being implemented by the person being corrected
and not by a library.

✓ Yes   ✕ No

Mandatory Notebook Formatting

Is the python notebook formatted as described in the mandatory part of
the project pdf?

☑ Yes   ✕ No

Exploring the green reds

Plotting a scatter matrix (V1.a)

- Does the scatter matrix plotting function produce a plot similar to
the example shown in the project pdf? (+2 points)

- Does changing the good_threshold and bad_threshold result in
different plots? Do more extreme thresholds result in fewer points
being plotted? (+2 points)

- Is it possible to save the plot as a .png? (+1 point)

Rate it from 0 (failed) through 5 (excellent)

◯

Useful factors (V1.b)

Are the factors chosen *linearly separable*? Can you plausibly draw
a line separating the good and bad wine datapoints for the factors
chosen? Use the plots created from the previous question to check.

☑ Yes   ✕ No

Learning to perceptron

Plotting perceptron performance (V.2c)

To correct this part of the project we're going to do things a bit
out of order. Start by looking at the performance plot for the

perceptron and make sure both of the following are true:

- Does the figure feature a plot that shows number of classification errors over a number of training epochs?

- Does the figure feature a plot that has 'good' and 'bad' wines datapoints plotted together with a 'decision boundary line' and boundary shadings?

- Does specifying different epoch numbers cause the 'decision boundary' plot to change?

If you're unsure about what the two plots should look like, take a look at the example figure in the project pdf.

☑ Yes   ✕ No

Checking the perceptron (V.2a & b)

Rerun the perceptron training function 4-5 times with number of training epochs set to 0.

Plot the output for each run and ensure the following are true:

- Make sure that the 'decision boundary line' in *every* plot separates 'good' from 'bad' wine datapoints. If a 'good' or 'bad' wine datapoint is *clearly* on the wrong side of the boundary line, then mark this question wrong.

- Make sure that the final number of classifications errors on the last epoch of training is 0.

- Make sure that the slope/position of the 'decision boundary line' is different from run to run.

- Make sure that the number of epochs it takes the perceptron to train is different from run to run.

☑ Yes   ✕ No

Checking the perceptron II (V.2d)

In a new code cell, run the perceptron training function with raw (unscaled) wine data. Then in another code cell run the perceptron training function with feature scaled data.

Have the person being corrected show you how to set a specific random seed so you can assess the effects of data feature scaling.

Note: Both training functions should have number of training epochs set to 0. Their learning rates should also be the same.

- You should find that the feature scaled data should take fewer epochs to reach 0 classification errors compared to the unscaled data.

            ☑ Yes   ✕ No

My fair ADALINE

Stop torturing that perceptron! (V.3a)

The answer to this question must touch on the fact that perceptrons can only successfully train on linearly separable problems. Changing the 'good' and 'bad' thresholds no longer allows a clean boundary to be determined. The perceptron will train forever trying to reduce classification errors to 0.

            ☑ Yes   ✕ No

Checking the ADALINE (V.3b & c)

Rerun the ADALINE training function 2-3 times with number of training epochs set to 0 and learning mode set to "online".

Then rerun the ADALINE training function 2-3 times with number of training epochs set to 0 and learning mode set to "batch".

For both sets of runs plot the performance for each run and ensure the following are true:

- Make sure that the 'decision boundary line' in each plot reasonably separates 'good' from 'bad' wine datapoints. The separation will *not* be perfect!

- Make sure that the number of classification errors is much lower at the end of training than the beginning of training.

- Make sure that the 'decision boundary line' for different runs manage to converge on similar solutions. (i.e. the position and slope of the line should not differ too much from run to run)

☑ Yes   ✕ No

Finding good settings for the ADALINE (V.3d)

- The number of classification errors for the "best" settings
should be in the neighborhood of 20 - 35. If the number of
classification errors is higher, then mark this question wrong.

- Are there example plots that support the final settings chosen?

☑ Yes   ✕ No

Advanced wine sampling and resampling

Testing the holdout function (V.4a)

To test whether the holdout function was correctly written,
run the following code snippet below.

Note: Make sure to change functions/variables as necessary

```
"
# Functions/Variables to change!
# You may need to change the function and variable
train, test = holdout_split(train_set, 0.7)

# ============ Do not modify any code below this line! ============
print('Samples in train set: {}\nSamples in validation set: {}'.format(len(train), len(test)))
assert(not any([True for index in train.index if index in test.index]))
# Because we want to be really really sure, plot train and test sets in same plot.
# Visually check that train and test point labels change for a given point if you run holdout many times
# Setting the split_fraction to a low or high number should make the difference very obvious
fig1, ax1 = plt.subplots()
train.plot(x='alcohol', y='volatile acidity', label='Train points', kind='scatter', color='magenta', s = 1, ax = ax1)
test.plot(x='alcohol', y='volatile acidity', label='Test points', kind='scatter', color='green', s = 1, ax = ax1)
"
```

If no assertion error is raised and the plot changes visually then with
repeated runs then mark this question correct.

☑ Yes   ✕ No

Testing the k-fold function (V.4b)

To test whether the k-fold function was correctly written, run the
following code snippet below. Try running it with different values for

.

Note: Make sure to change functions/variables as necessary

```
"
# Functions/Variables to change!
k_folds = 9
folds = kfold_split(train_set, k_folds, shuffle=False)

# ============ Do not modify any code below this line! ============
assert(len(folds) == k_folds)
corr_xval_sizes = [(len(data) // k_folds) + 1 if (i < len(data) % k_folds) else (len(data) // k_folds)
for i in range(k_folds)]
corr_train_sizes = [len(data) - xval_size for xval_size in corr_xval_sizes]
user_train_xval_sizes = [(len(train), len(xval)) for train, xval in [tup for tup in folds]]
assert(list(zip(corr_train_sizes, corr_xval_sizes)) == user_train_xval_sizes)
assert(not any([any([True for index in train.index if index in xval.index]) for train, xval in folds]))
train_lst, xval_lst = zip(*folds)
assert(all([[len(xval.index.intersection(xval_lst[j].index)) == 0 for j in range(len(xval_lst)) if i != j]
for i, xval in enumerate(xval_lst)]))
print('(Train, Xval) lengths for {} folds:\n{}'.format(k_folds, [(len(train), len(xval)) for train, xval in folds]))
"
```

If no assertion error is raised, then mark this question correct.

✓ Yes　　　　　　　　　　　　　　　　　　✕ No

Assessing learning rate and training epoch with k-fold (V.4c)

- Were more than 4 different combinations of learning rate and
training epoch options tried?

- Were conclusions that were drawn supported by the actual results of
the k-folds? Example: If the the person being corrected said that
having a tiny learning rate and small number of epochs led to poor
accuracy, do they have an experiment or experiments with k-fold that
back it up?

✓ Yes　　　　　　　　　　　　　　　　　　✕ No

Adventures in the Nth dimension

How many chemical factors? (V.5a)

- Were there at least 3 different experiments where the number or types of chemical factors used to train were changed?

- Were conclusions made supported by the experiments?

☑ Yes   ✕ No

What do decision boundaries look like? (V.5b)

- A decision boundary for 3 factors is a 2 dimensional plane

- A decision boundary for 7 factors is a 6 dimensional plane

- A decision boundary for 11 factors takes on a 10 dimensional plane

- Mentioning hyperplanes is great, but not necessary

☑ Yes   ✕ No

Marvin's rebuttal

Can you rise to Marvin's rebuttal? (V.6a)

Make sure both of the following are true!

- Was the Pan-Galactic Gargle Blaster dataset successfully classified with a single perceptron or ADALINE?

- Is there a performance plot showing *perfect* classification of good and bad gargle blasters? (i.e. 0 classification errors)

☑ Yes   ✕ No

Bonus

Can you go fast? (VI.1)

- Check that the functions for the perceptron and ADALINE parts of this project use either 'cdef' or 'cpdef' function definitions.

- Check that malloc was used (It's impossible to avoid if this challenge was done properly).

- Was all malloc'ed memory freed? (ALL of it must be manually freed!)

⌢ Yes   ✕ No

Do perceptrons dream of electric sheep? (VI.2)

- Does the animated plot show a changing number of classification errors over time?

- Does the animated plot show a changing decision boundary for a given perceptron or ADALINE?

- You can compare with the provided example which you can download from the correction resources.

⌢ Yes   ✕ No

Dimensional traveler (VI.3)

- Does the plot have 3 axes with each axis representing a different wine chemical factor?

- Is there a 2D decision plane being plotted together with the 'good' and 'bad' wine datapoints?

⌢ Yes   ✕ No

Ratings

Don't forget to check the flag corresponding to the defense

⊘ Forbidden function

Conclusion

Leave a comment on this evaluation

Good work. A week of
effort. Good luck!

Finish evaluation

General term of use of the site (https://signin.intra.42.fr/legal/terms/6)
Privacy policy (https://signin.intra.42.fr/legal/terms/5)
Legal notices (https://signin.intra.42.fr/legal/terms/3)
Declaration on the use of cookies (https://signin.intra.42.fr/legal/terms/2)
Terms of use for video surveillance (https://signin.intra.42.fr/legal/terms/1)
Rules of procedure (https://signin.intra.42.fr/legal/terms/4)