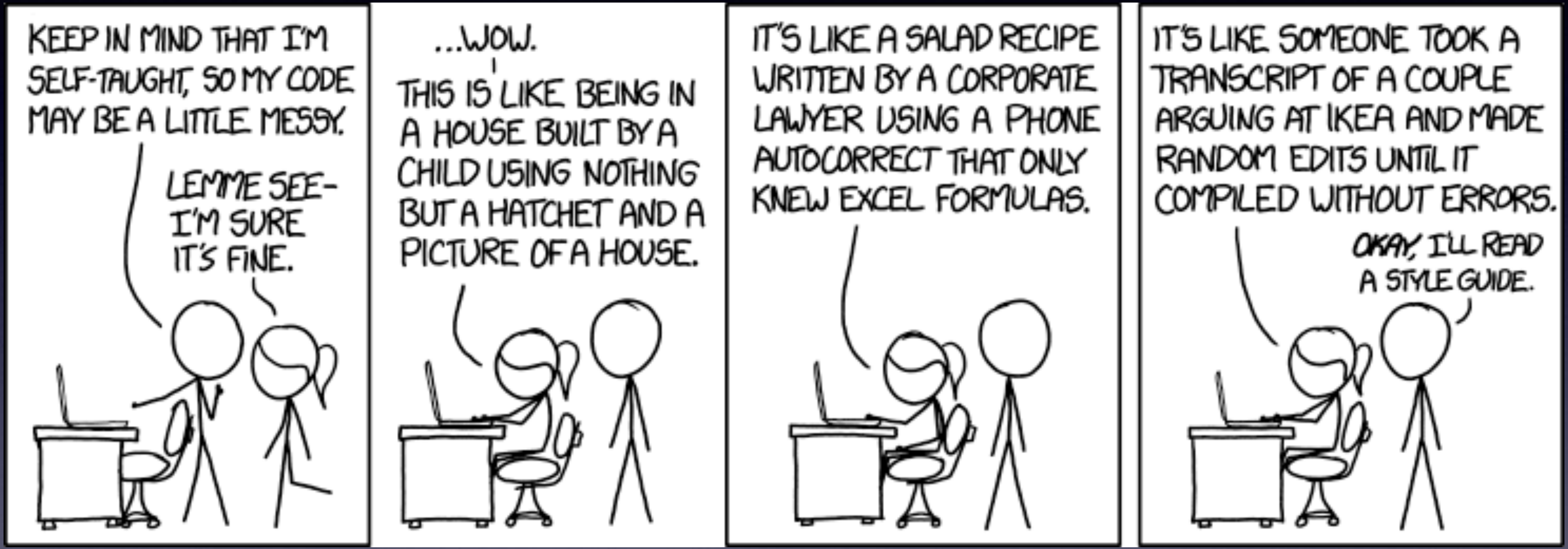
Software craftsmanship

Brendan Herger


KEEP IN MIND THAT I'M SELF-TAUGHT, SO MY CODE MAY BE A LITTLE MESSY.
LEMME SEE– I'M SURE IT'S FINE.
...WOW.
THIS IS LIKE BEING IN A HOUSE BUILT BY A CHILD USING NOTHING BUT A HATCHET AND A PICTURE OF A HOUSE.
IT'S LIKE A SALAD RECIPE WRITTEN BY A CORPORATE LAWYER USING A PHONE AUTOCORRECT THAT ONLY KNEW EXCEL FORMULAS.
IT'S LIKE SOMEONE TOOK A TRANSCRIPT OF A COUPLE ARGUING AT IKEA AND MADE RANDOM EDITS UNTIL IT COMPILED WITHOUT ERRORS.
OKAY, I'LL READ A STYLE GUIDE.

https://xkcd.com/1513/

Craft professional, maintainable code

Why?

Style Guides

Tools

Why?

Goals: Why?

Using style guides makes it easier to collaborate, by creating internally consistent, maintainable code bases.

Why?

- **Money:** DS w/ strong coding skills make $15k/year more
- **Time:** Crafted code bases take less time to build and maintain
- **Professionalism:** Software engineering is our primary tool

Style Guides

Style Guides: Code

- Google style guide: A practical, well used style guide. Particularly good for its naming conventions
- PEP-8: Python's official 'style guide', though it is not actively maintained

Style Guides: Docstring

In addition to style guides for code, there are a few different ways to format code comments. The goals of docstrings usually include:

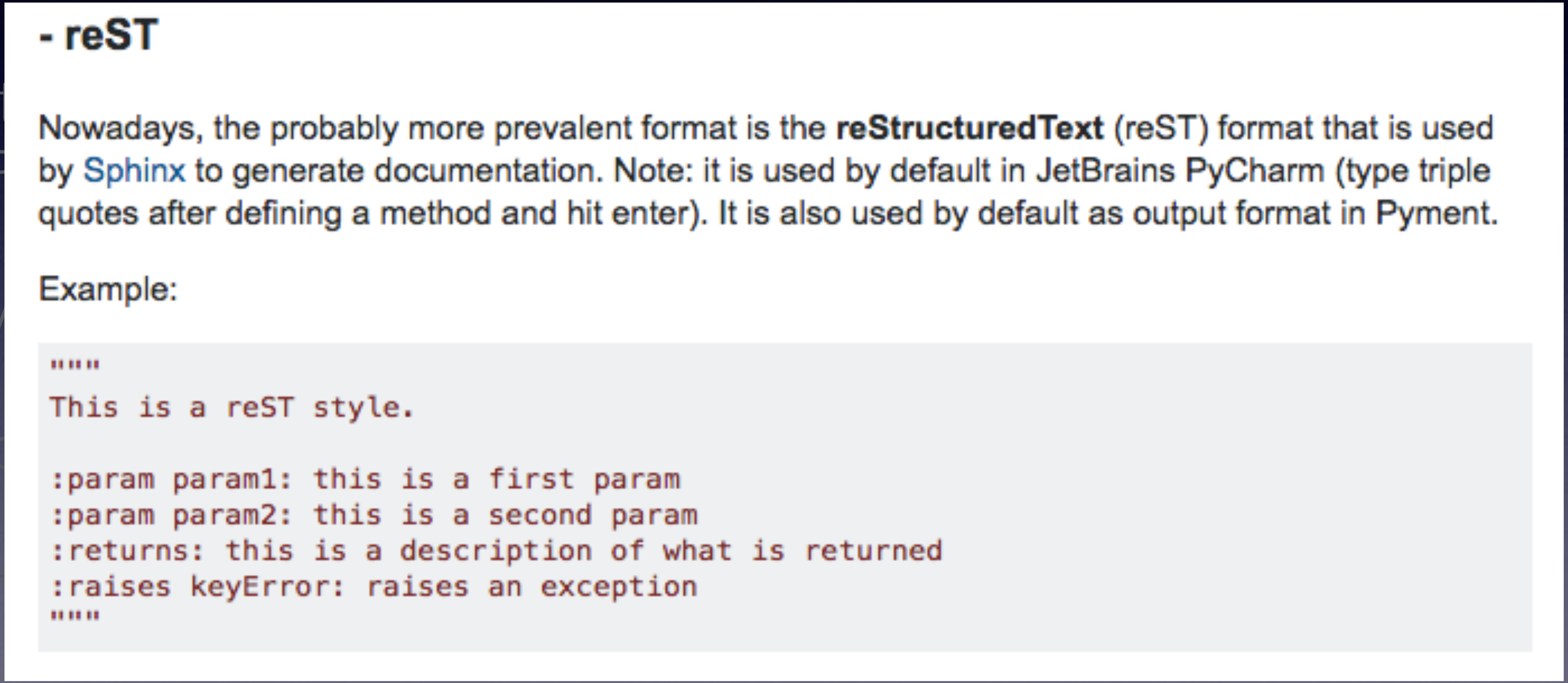
- Describing functionality
- Describing inputs and outputs
- Providing method signature (variable name and type)

Style Guides: Docstring

- reStructured Text (rST): Officially preferred guide, recommended in PEP-287
- Epytest: Mirroring JavaDoc style
- Google: Convenient for longer descriptions
- Numpy-style: Incredibly popular

Docstrings

- reSt
PEF
- Epy
- Goo
- Nur



- reST

Nowadays, the probably more prevalent format is the **reStructuredText** (reST) format that is used by Sphinx to generate documentation. Note: it is used by default in JetBrains PyCharm (type triple quotes after defining a method and hit enter). It is also used by default as output format in Pyment.

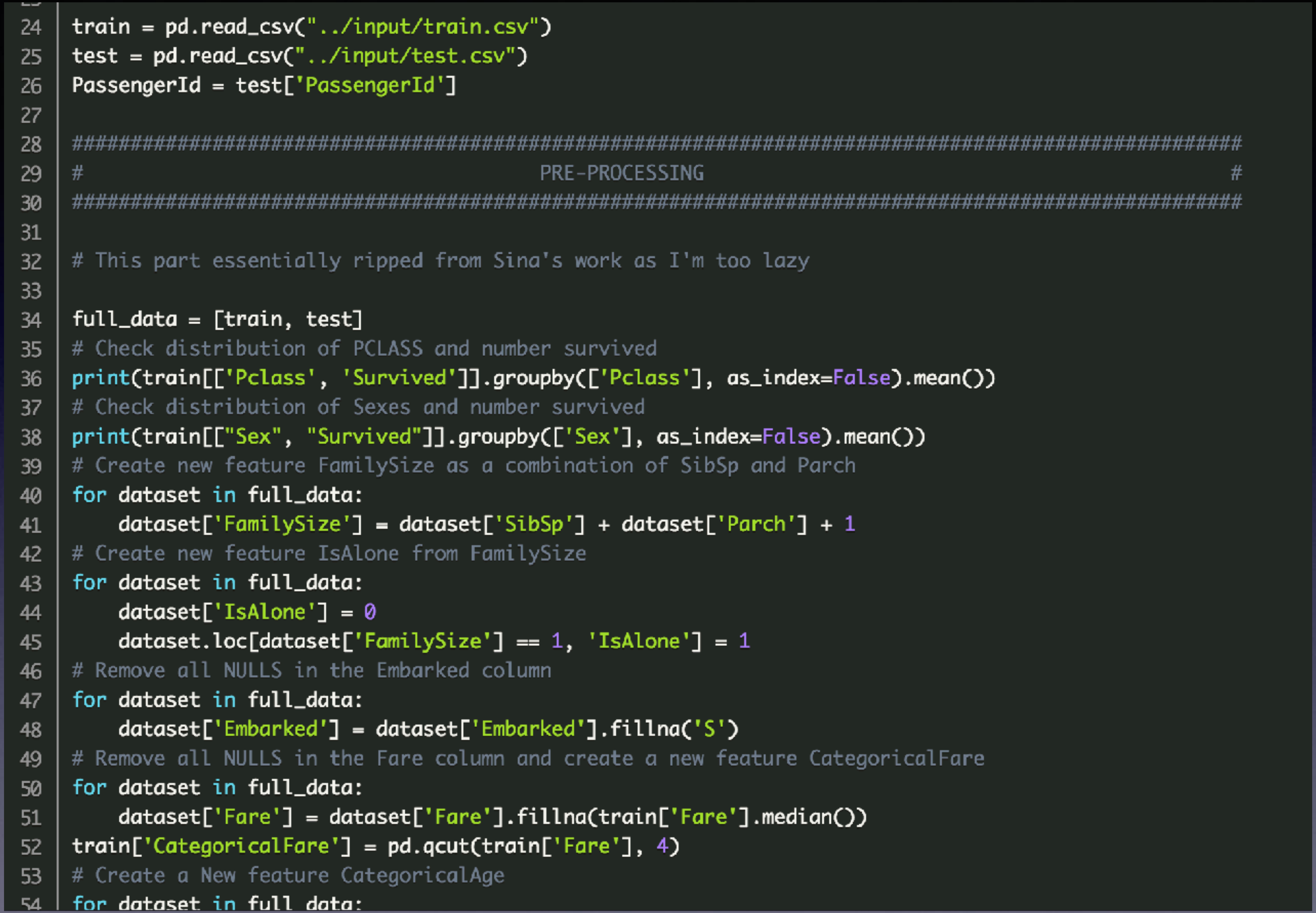
Example:

```
"""
This is a reST style.

:param param1: this is a first param
:param param2: this is a second param
:returns: this is a description of what is returned
:raises keyError: raises an exception
"""
```

Tools

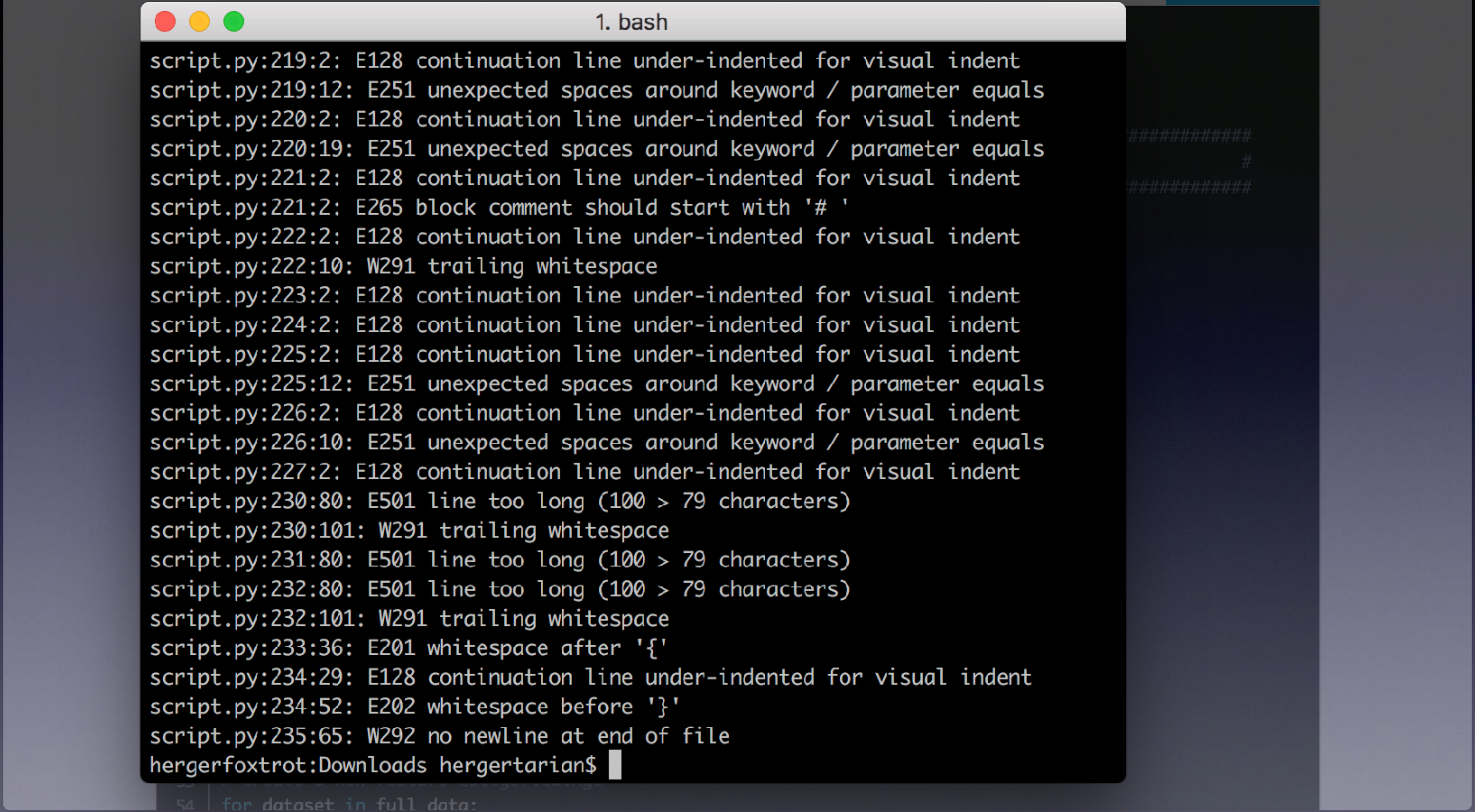
Tools: Style Checker

- PyLint: CLI, provides code score and list of which rules have been broken (w/ line number)
- PEP8 tool: CLI Implementation of PEP8 rules, provides list of which rules have been broken (w/ line number). Sometimes different than PyLint



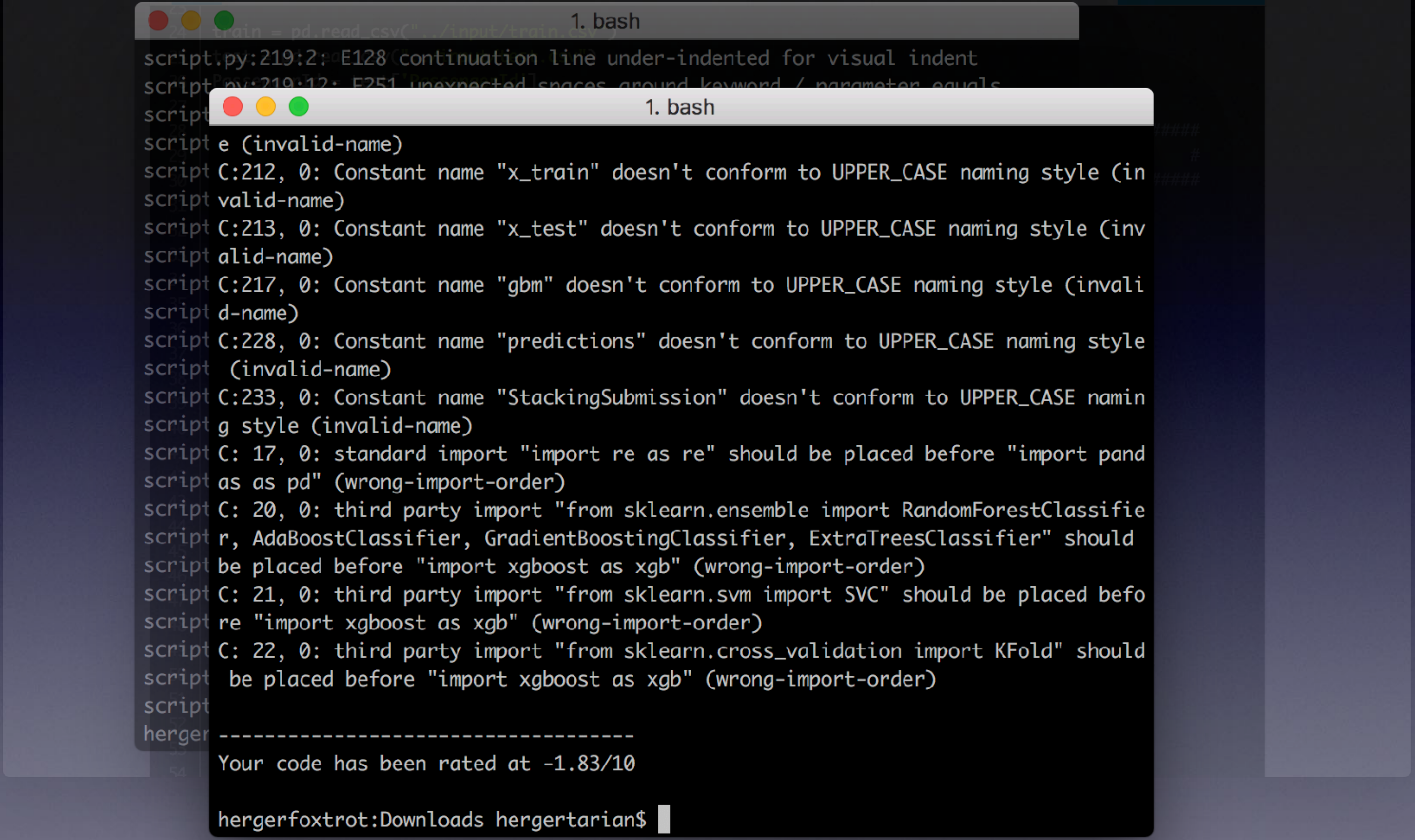
```
train = pd.read_csv("../input/train.csv")
test = pd.read_csv("../input/test.csv")
PassengerId = test['PassengerId']

#########################################################################################
#                                       PRE-PROCESSING                                  #
#########################################################################################

# This part essentially ripped from Sina's work as I'm too lazy

full_data = [train, test]
# Check distribution of PCLASS and number survived
print(train[['Pclass', 'Survived']].groupby(['Pclass'], as_index=False).mean())
# Check distribution of Sexes and number survived
print(train[["Sex", "Survived"]].groupby(['Sex'], as_index=False).mean())
# Create new feature FamilySize as a combination of SibSp and Parch
for dataset in full_data:
    dataset['FamilySize'] = dataset['SibSp'] + dataset['Parch'] + 1
# Create new feature IsAlone from FamilySize
for dataset in full_data:
    dataset['IsAlone'] = 0
    dataset.loc[dataset['FamilySize'] == 1, 'IsAlone'] = 1
# Remove all NULLS in the Embarked column
for dataset in full_data:
    dataset['Embarked'] = dataset['Embarked'].fillna('S')
# Remove all NULLS in the Fare column and create a new feature CategoricalFare
for dataset in full_data:
    dataset['Fare'] = dataset['Fare'].fillna(train['Fare'].median())
train['CategoricalFare'] = pd.qcut(train['Fare'], 4)
# Create a New feature CategoricalAge
for dataset in full_data:
```

https://www.kaggle.com/arthurtok/0-808-with-simple-stacking



```
script.py:219:2: E128 continuation line under-indented for visual indent
script.py:219:12: E251 unexpected spaces around keyword / parameter equals
script.py:220:2: E128 continuation line under-indented for visual indent
script.py:220:19: E251 unexpected spaces around keyword / parameter equals
script.py:221:2: E128 continuation line under-indented for visual indent
script.py:221:2: E265 block comment should start with '# '
script.py:222:2: E128 continuation line under-indented for visual indent
script.py:222:10: W291 trailing whitespace
script.py:223:2: E128 continuation line under-indented for visual indent
script.py:224:2: E128 continuation line under-indented for visual indent
script.py:225:2: E128 continuation line under-indented for visual indent
script.py:225:12: E251 unexpected spaces around keyword / parameter equals
script.py:226:2: E128 continuation line under-indented for visual indent
script.py:226:10: E251 unexpected spaces around keyword / parameter equals
script.py:227:2: E128 continuation line under-indented for visual indent
script.py:230:80: E501 line too long (100 > 79 characters)
script.py:230:101: W291 trailing whitespace
script.py:231:80: E501 line too long (100 > 79 characters)
script.py:232:80: E501 line too long (100 > 79 characters)
script.py:232:101: W291 trailing whitespace
script.py:233:36: E201 whitespace after '{'
script.py:234:29: E128 continuation line under-indented for visual indent
script.py:234:52: E202 whitespace before '}'
script.py:235:65: W292 no newline at end of file
hergerfoxtrot:Downloads hergertarian$
```

https://www.kaggle.com/arthurtok/0-808-with-simple-stacking



```
script.py:219:2: E128 continuation line under-indented for visual indent
```

```
e (invalid-name)
C:212, 0: Constant name "x_train" doesn't conform to UPPER_CASE naming style (in
valid-name)
C:213, 0: Constant name "x_test" doesn't conform to UPPER_CASE naming style (inv
alid-name)
C:217, 0: Constant name "gbm" doesn't conform to UPPER_CASE naming style (invali
d-name)
C:228, 0: Constant name "predictions" doesn't conform to UPPER_CASE naming style
 (invalid-name)
C:233, 0: Constant name "StackingSubmission" doesn't conform to UPPER_CASE namin
g style (invalid-name)
C: 17, 0: standard import "import re as re" should be placed before "import pand
as as pd" (wrong-import-order)
C: 20, 0: third party import "from sklearn.ensemble import RandomForestClassifie
r, AdaBoostClassifier, GradientBoostingClassifier, ExtraTreesClassifier" should
be placed before "import xgboost as xgb" (wrong-import-order)
C: 21, 0: third party import "from sklearn.svm import SVC" should be placed befo
re "import xgboost as xgb" (wrong-import-order)
C: 22, 0: third party import "from sklearn.cross_validation import KFold" should
 be placed before "import xgboost as xgb" (wrong-import-order)

-----------------------------------
Your code has been rated at -1.83/10

hergerfoxtrot:Downloads hergertarian$
```

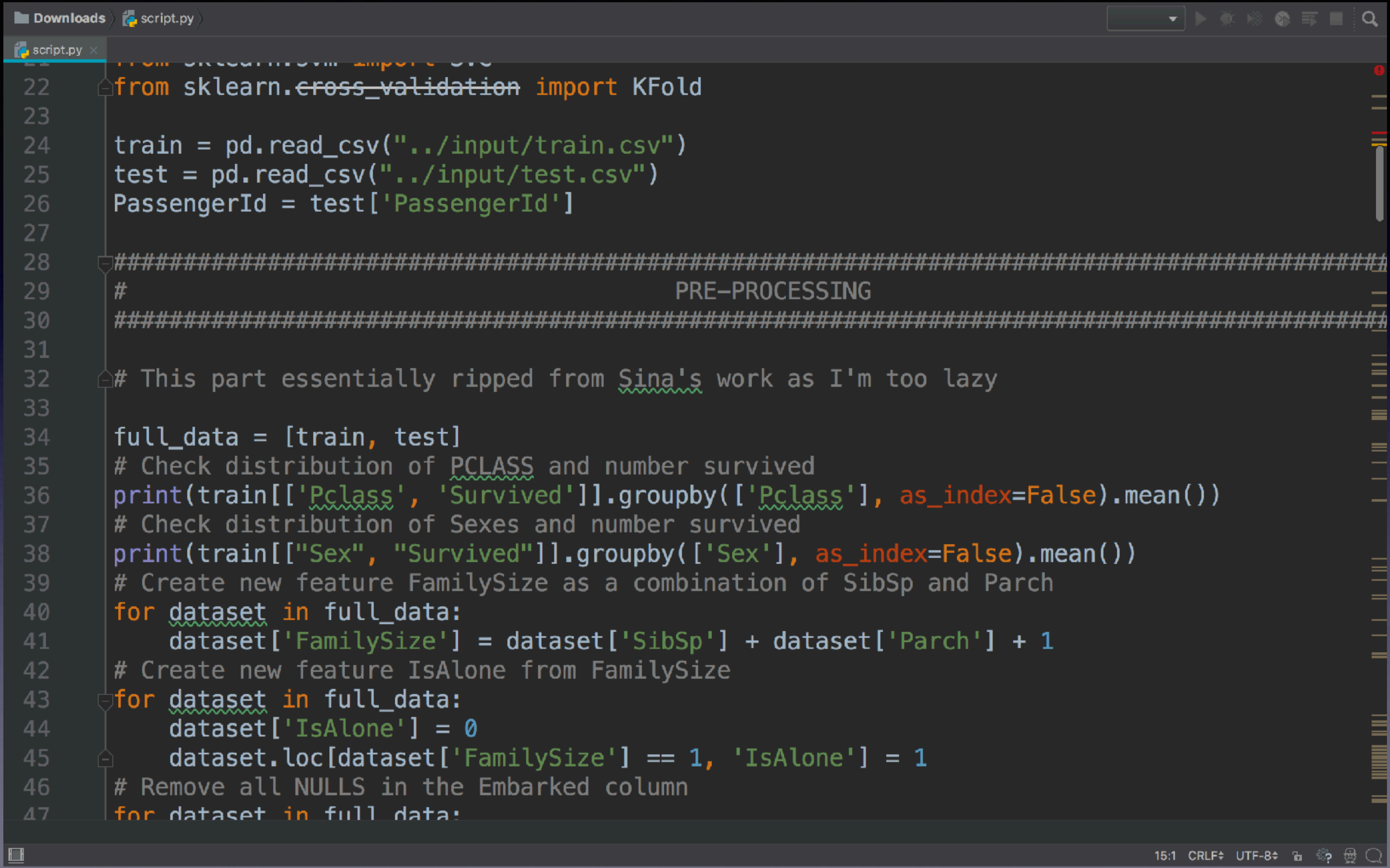
https://www.kaggle.com/arthurtok/0-808-with-simple-stacking

Tools: IDEs

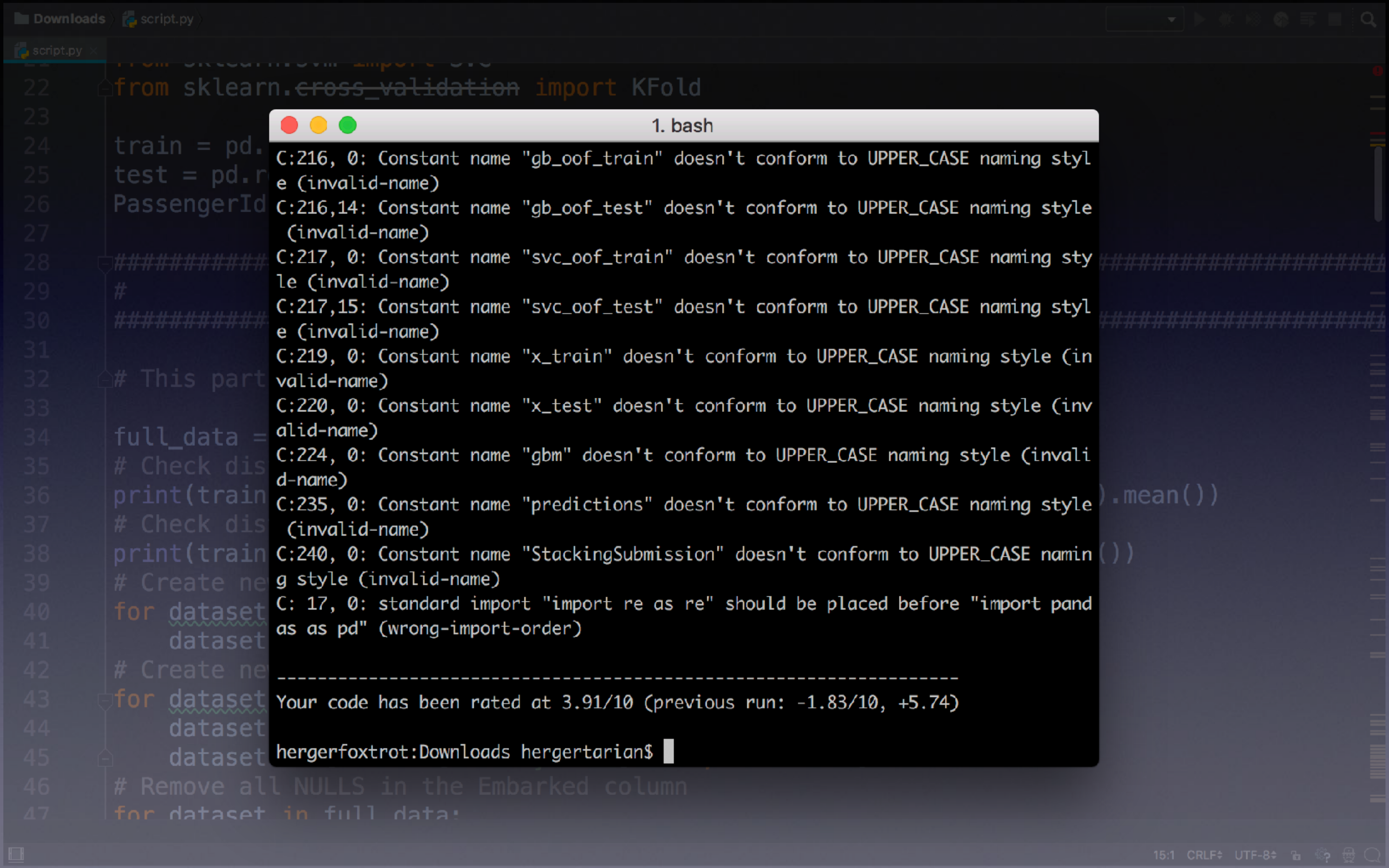
Integrated Development Environments (IDEs) help software engineers craft beautiful code, by

- Automatically highlighting and correcting style issues
- Providing rapid access into underlying source code
- Providing fully featured debugging tools

A current industry standard IDE for python is PyCharm



```
from sklearn.svm import SVC
from sklearn.~~cross_validation~~ import KFold

train = pd.read_csv("../input/train.csv")
test = pd.read_csv("../input/test.csv")
PassengerId = test['PassengerId']

###########################################################################################
#                                       PRE-PROCESSING
###########################################################################################

# This part essentially ripped from Sina's work as I'm too lazy

full_data = [train, test]
# Check distribution of PCLASS and number survived
print(train[['Pclass', 'Survived']].groupby(['Pclass'], as_index=False).mean())
# Check distribution of Sexes and number survived
print(train[["Sex", "Survived"]].groupby(['Sex'], as_index=False).mean())
# Create new feature FamilySize as a combination of SibSp and Parch
for dataset in full_data:
    dataset['FamilySize'] = dataset['SibSp'] + dataset['Parch'] + 1
# Create new feature IsAlone from FamilySize
for dataset in full_data:
    dataset['IsAlone'] = 0
    dataset.loc[dataset['FamilySize'] == 1, 'IsAlone'] = 1
# Remove all NULLS in the Embarked column
for dataset in full_data:
```



```
C:216, 0: Constant name "gb_oof_train" doesn't conform to UPPER_CASE naming styl
e (invalid-name)
C:216,14: Constant name "gb_oof_test" doesn't conform to UPPER_CASE naming style
 (invalid-name)
C:217, 0: Constant name "svc_oof_train" doesn't conform to UPPER_CASE naming sty
le (invalid-name)
C:217,15: Constant name "svc_oof_test" doesn't conform to UPPER_CASE naming styl
e (invalid-name)
C:219, 0: Constant name "x_train" doesn't conform to UPPER_CASE naming style (in
valid-name)
C:220, 0: Constant name "x_test" doesn't conform to UPPER_CASE naming style (inv
alid-name)
C:224, 0: Constant name "gbm" doesn't conform to UPPER_CASE naming style (invali
d-name)
C:235, 0: Constant name "predictions" doesn't conform to UPPER_CASE naming style
 (invalid-name)
C:240, 0: Constant name "StackingSubmission" doesn't conform to UPPER_CASE namin
g style (invalid-name)
C: 17, 0: standard import "import re as re" should be placed before "import pand
as as pd" (wrong-import-order)

-------------------------------------------------------------------
Your code has been rated at 3.91/10 (previous run: -1.83/10, +5.74)

hergerfoxtrot:Downloads hergertarian$
```

Why?

Style Guides

Tools

Craft professional, maintainable code