Name: Micah Wagner

Dataset link: https://www.kaggle.com/datasets/sidhus/crab-age-prediction

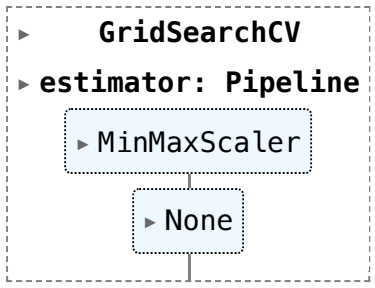
In [40]:
```
import numpy as np
import pandas as pd
from sklearn.pipeline import Pipeline
from sklearn.base import TransformerMixin, BaseEstimator
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import MinMaxScaler
```

In [41]:
```
raw_data = pd.read_csv("CrabAgePrediction.csv")
#select all columns, including categorical features
selected_data = pd.get_dummies(raw_data, dtype=float)
selected_data.fillna(value=0, inplace = True)
selected_data["IsMature"] = selected_data["Age"] > 11
selected_data = selected_data.drop(columns = ["Age"])
xs = selected_data.drop(columns = ["IsMature"])
ys = selected_data["IsMature"]


print(xs, ys)
```

```
      Length  Diameter  Height     Weight  Shucked Weight  Viscera Weight  \
0     1.4375    1.1750  0.4125  24.635715       12.332033        5.584852
1     0.8875    0.6500  0.2125   5.400580        2.296310        1.374951
2     1.0375    0.7750  0.2500   7.952035        3.231843        1.601747
3     1.1750    0.8875  0.2500  13.480187        4.748541        2.282135
4     0.8875    0.6625  0.2125   6.903103        3.458639        1.488349
...      ...       ...     ...        ...             ...             ...
3888  1.4625    1.1375  0.3250  24.819987       11.651644        5.854172
3889  1.5500    1.2125  0.4375  34.458817       15.450477        7.172423
3890  0.6250    0.4625  0.1625   2.012815        0.765436        0.524466
3891  1.0625    0.7750  0.2625  10.347568        4.507570        2.338834
3892  0.7875    0.6125  0.2125   4.068153        1.502523        1.346601

      Shell Weight  Sex_F  Sex_I  Sex_M
0         6.747181    1.0    0.0    0.0
1         1.559222    0.0    0.0    1.0
2         2.764076    0.0    1.0    0.0
3         5.244657    1.0    0.0    0.0
4         1.700970    0.0    1.0    0.0
...            ...    ...    ...    ...
3888      6.378637    1.0    0.0    0.0
3889      9.780577    1.0    0.0    0.0
3890      0.637864    0.0    1.0    0.0
3891      2.976698    0.0    1.0    0.0
3892      1.417475    0.0    1.0    0.0

[3893 rows x 10 columns] 0       False
1       False
2       False
3       False
4       False
        ...
3888    False
3889    False
3890    False
3891    False
3892    False
Name: IsMature, Length: 3893, dtype: bool
```

In [42]:
```
grid_gradient = {
    "classify": [
        GradientBoostingClassifier()
    ],
    "classify__max_depth": [3,4,5,6,7,8,9,10],
    "classify__max_features": ["sqrt", "log2"],
    "classify__learning_rate": [0.025, 0.05, 0.1, 0.2, 0.3, 0.4],

}

steps = [
    ("scale", MinMaxScaler()),
    ("classify", None)
]

pipe = Pipeline(steps)

search_gradient= GridSearchCV(pipe, grid_gradient, scoring='f1', n_jobs=-1)
search_gradient.fit(xs, ys)
```

Out[42]:



In [43]:
```
print(search_gradient.best_score_)
print(search_gradient.best_params_)
```

```
0.5699916942839318
{'classify': GradientBoostingClassifier(learning_rate=0.4, max_features='log2'), 'classify__learning_rate': 0.4, 'cl
assify__max_depth': 3, 'classify__max_features': 'log2'}
```

1. I would expect the chosen metric to decrease because the portion of data I choose to split could unfavorably bias the model. That being said, it could also bias the model the other way, giving us an unrealistically high metric score. Generally, cross-validation gives a more accurate accounting of the performance of the model since it tests each fold of data across the whole dataset and averages the performance at the end (we know that every instance was tested one time, and this eliminates possible unwanted biases produced from the test/train split).

2. Since this is a classification problem, we need to use a metric appropriate for this task. Accuracy could have been okay, but when I looked through my data, it was clear that there were more false values than true values. Using precision and recall alone can lead to very high scores that are misleading, so I decided to use the F1 score which will combine both precision and recall to give one metric score that describes how well our model performed, avoiding the precision / recall edge cases.

3. I think the features that it chose were optimal because a higher max depth value might start to produce overfitting. I'm suprised that it performed so poorly, I thought the relationship between shell thickness / crab size would be enough to accurately predict whether the crab was mature (age 12 months and up, I just looked up what age crabs become mature).