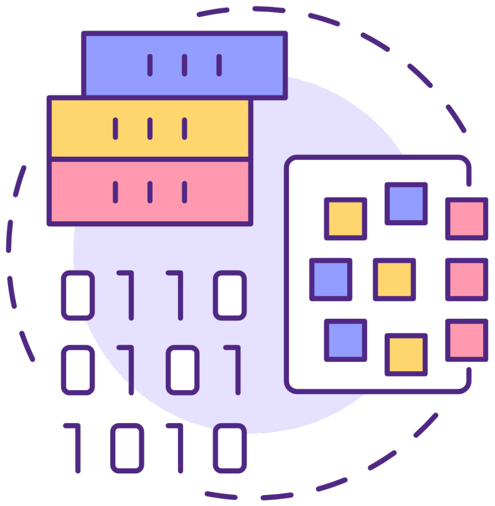
Assignment Lesson 4 - 5

- Classification Algorithms -

Classification is at the core of Data Science itself–to predict whether data belong to one class or another.



In this assignment, you will try to perform some Classification techniques and give some reasoning and analysis about your choice of parameter and the obtained result.
Again, you are **free to use any dataset you are interested in** as long as it satisfies the following conditions:

- Contains at least **5 regular attributes** (can be numerical or categorical (before encoding). ID-like data (such as name) or time series data is not counted.)
- Contains at least **400 examples.**
- Contains **a binary categorical label attribute** (you can always discretize to make it binary)

Feel free to priorly adjust the data (discretization or just select a few attributes) so it will fit those requirements. Finding a perfect dataset can be challenging this time. Some cleaning might be needed. You can look at kaggle.com for some datasets. If the dataset you used in the previous assignment still satisfied those conditions, you can use that dataset again. To complete this assignment feel free to choose any statistical tool you prefer.

[!] ***Three students*** *who submit their assignment by the deadline with the most interesting dataset and prediction will receive* **free non-fiction book(s) of their choice totaling at most 5000 yen**. *As long as you can find them here: https://www.amazon.co.jp, the book can be in English, Japanese, or Chinese. No need to worry about the delivery fees. A book related to your study is a better choice! Winners will be announced approximately two weeks after the deadline.*

Book's Amazon Links (if not sure, leave it blank):

1. JLPT N3 Grammar
2. Kanji Flash Cards

Prior Knowledge

● Data Science Objective 5/5

Using the data you have, write down **what kind of prediction** you will do and **why it matters**.

I want to see if we can predict the gender of a video game character based on one or multiple of the following criteria:

- If we can be the character (i.e. the character is playable)
- How important to the story the character is
- If the character is sexualized or linked with romance
- If the character is supportive, neutral or negative towards the protagonist.

I am interested to see if the characters in video games have any gender biases. It is essential because video games belong to our culture and indirectly influence us. Children are likely to identify themselves with video game characters and, more generally, get inspired by video games. So, there must be no gender biases in video games.

Upload the data to your Google Drive and paste the link here. It can be in a **csv or xls** format (not compressed like zip). Please make sure the sharing setting allows Tohoku University members to see it.

Link: https://drive.google.com/file/d/1ESeGIPH_rcg3mSWBJs5y8IlfAZT1hKG-/view?usp=share_link

Original data: https://www.kaggle.com/datasets/br33sa/gender-representation-in-video-games
I only used the file characters.grivg.csv. From this file, I removed the non-binary characters and information which discretizes a character, i.e. the columns Id, Age, Game, Name, and Species.
Legend:

Chosen a binary categorical label attribute: Is_Female:

- 0: No (male character)
- 1: Yes (female character)

Relevance: (One-hot Encoded in our dataset)

- SC - secondary character
- MC - the main character
- PA - Protagonist
  MA - Main antagonisy
- DA - Deuteragonist
- SK - Sidekick

Side: (One-hot Encoded in our dataset)

- P - On the protagonist side or neutral side
- A - Antagonists
- B - characters that are or can be on both of the above sides.

Romantic Interest:

- 0: No - this character is no romantic interest of the protagonist
- 0.5: Opt - this character can be dated or can have romantic or sexual interactions with the protagonist based on the player's choice.
- 1: Yes - this character is unavoidably the romantic interest or partner of the main character.

Age Range:

- 0: Unknown
- 1: Infant - 0 to 5 years old
- 2: Child - 6 to 14 years old
- 3: Teenager - 15 to 17 years old
- 4: Young-Adult - 18 to 24 years old
- 5: Adult - 25 to 39 years old
- 6: Middle-Aged - 40 to 64 years old
- 7: Elderly - older than 65 years old

Playable:

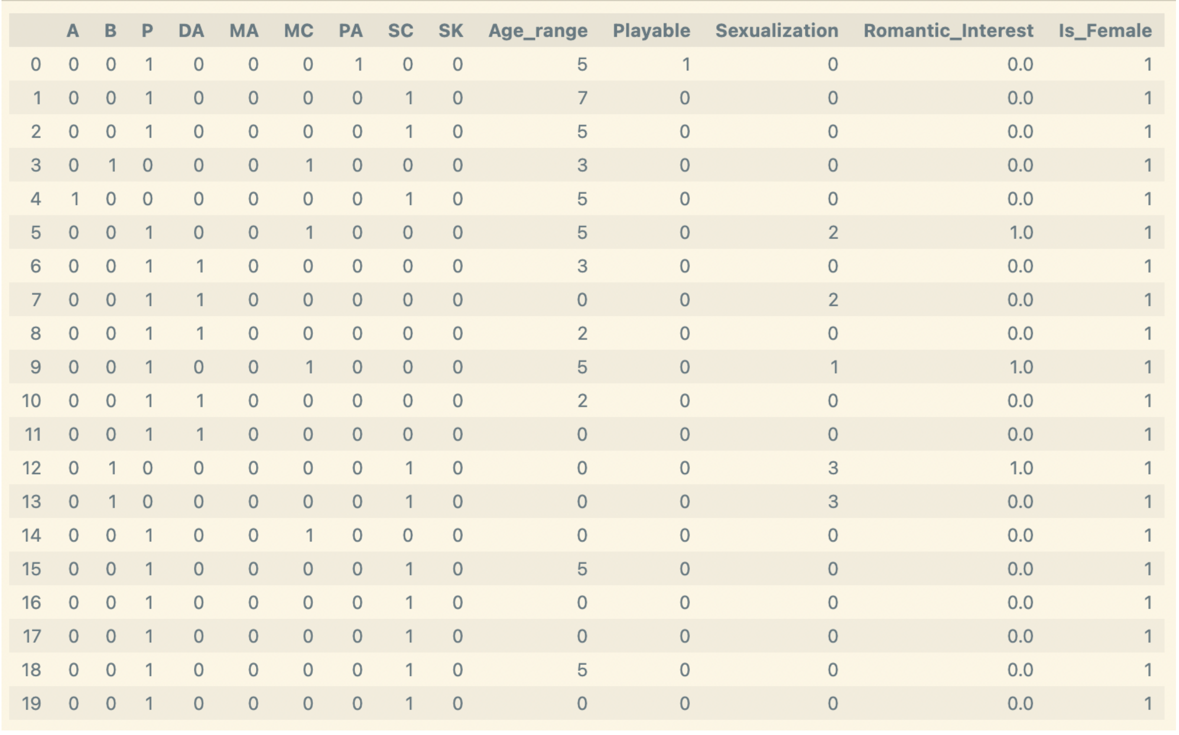
- 0: Not playable
- 1: Playable

Data Preparation & Exploration

● Dataset

Give a screenshot of the **top 10 - 30 examples** of your dataset.

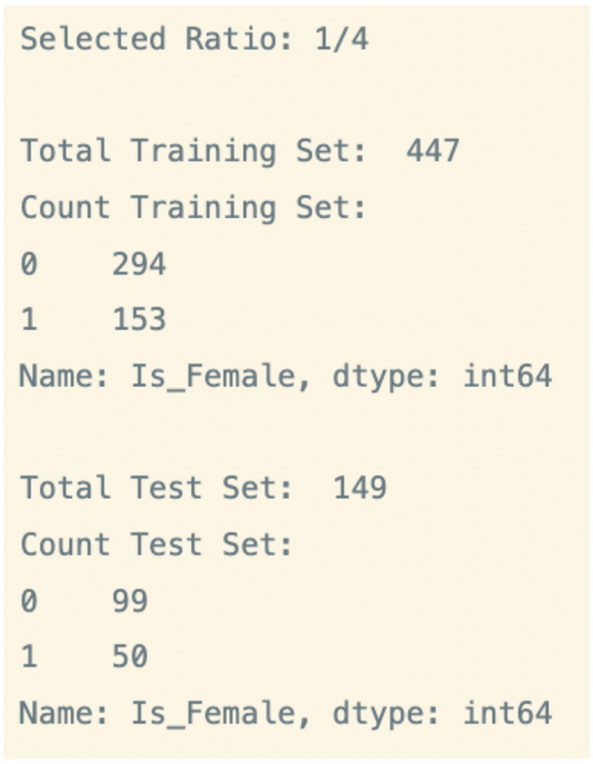
Top 20



	A	B	P	DA	MA	MC	PA	SC	SK	Age_range	Playable	Sexualization	Romantic_Interest	Is_Female
0	0	0	1	0	0	0	1	0	0	5	1	0	0.0	1
1	0	0	1	0	0	0	0	1	0	7	0	0	0.0	1
2	0	0	1	0	0	0	0	1	0	5	0	0	0.0	1
3	0	1	0	0	0	1	0	0	0	3	0	0	0.0	1
4	1	0	0	0	0	0	0	1	0	5	0	0	0.0	1
5	0	0	1	0	0	1	0	0	0	5	0	2	1.0	1
6	0	0	1	1	0	0	0	0	0	3	0	0	0.0	1
7	0	0	1	1	0	0	0	0	0	0	0	2	0.0	1
8	0	0	1	1	0	0	0	0	0	2	0	0	0.0	1
9	0	0	1	0	0	1	0	0	0	5	0	1	1.0	1
10	0	0	1	1	0	0	0	0	0	2	0	0	0.0	1
11	0	0	1	1	0	0	0	0	0	0	0	0	0.0	1
12	0	1	0	0	0	0	0	1	0	0	0	3	1.0	1
13	0	1	0	0	0	0	0	1	0	0	0	3	0.0	1
14	0	0	1	0	0	1	0	0	0	0	0	0	0.0	1
15	0	0	1	0	0	0	0	1	0	5	0	0	0.0	1
16	0	0	1	0	0	0	0	1	0	0	0	0	0.0	1
17	0	0	1	0	0	0	0	1	0	0	0	0	0.0	1
18	0	0	1	0	0	0	0	1	0	5	0	0	0.0	1
19	0	0	1	0	0	0	0	1	0	0	0	0	0.0	1

Preparing Training and Testing 5/5

Split the data for training and testing using **class-stratified sampling**. The suggested splitting ratio is ⅓, ¼, or ⅕. Give a screenshot of the **count of both datasets** (how many examples each has) and also the **count of each class in the label of both datasets.**



```
Selected Ratio: 1/4

Total Training Set:  447
Count Training Set:
0    294
1    153
Name: Is_Female, dtype: int64

Total Test Set:  149
Count Test Set:
0    99
1    50
Name: Is_Female, dtype: int64
```

Summary Statistics 5/5

Give a screenshot of the summary statistics of your data (**mean**, **median, standard deviation, min, max, count)**.

A:

Mean	: 0.22
Median	: 0
Standard Deviation	: 0.41
Min	: 0
Max	: 1
Count 0	: 467
Count 1	: 129

```
B:
        Mean            : 0.12
        Median          : 0
        Standard Deviation: 0.33
        Min             : 0
        Max             : 1
          Count 0       : 523
          Count 1       : 73
P:
        Mean            : 0.66
        Median          : 1
        Standard Deviation: 0.47
        Min             : 0
        Max             : 1
          Count 1        : 394
          Count 0        : 202
DA:
        Mean            : 0.03
        Median           : 0
        Standard Deviation: 0.17
        Min            : 0
        Max             : 1
          Count 0   578
          Count 1    18
MA:
        Mean            : 0.082
        Median          : 0
        Standard Deviation: 0.27
        Min              : 0
        Max              : 1
          Count 0        : 547
          Count 1        : 49
MC:
        Mean            : 0.33
        Median          : 0
        Standard Deviation: 0.47
        Min             : 0
        Max             : 1
          Count 0       : 398
          Count 1       : 198
```

```
PA:
          Mean             : 0.1
          Median          : 0
          Standard Deviation: 0.3
          Min               : 0
          Max              : 1
             Count 0       : 535
             Count 1       : 61
SC:
          Mean             : 0.43
          Median            : 0
          Standard Deviation: 0.5
          Min             : 0
          Max             : 1
             Count 0      : 339
             Count 1     : 257
SK:
          Mean             : 0.022
          Median            : 0
          Standard Deviation: 0.15
          Min             : 0
          Max              : 1
             Count 0     :  583
             Count 1     : 13
Age_range:
          Mean             : 2.8
          Median            : 3
          Standard Deviation: 2.5
          Min             : 0
          Max              : 7
             Count 5     : 259
             Count 0     :  228
             Count 3     :   42
             Count 2     :   35
             Count 7     :   29
             Count 1     :     3
Playable:
          Mean             : 0.18
          Median            : 0
          Standard Deviation: 0.38
          Min             : 0
          Max              : 1
             Count 0     :  491
             Count 1     :  105
```

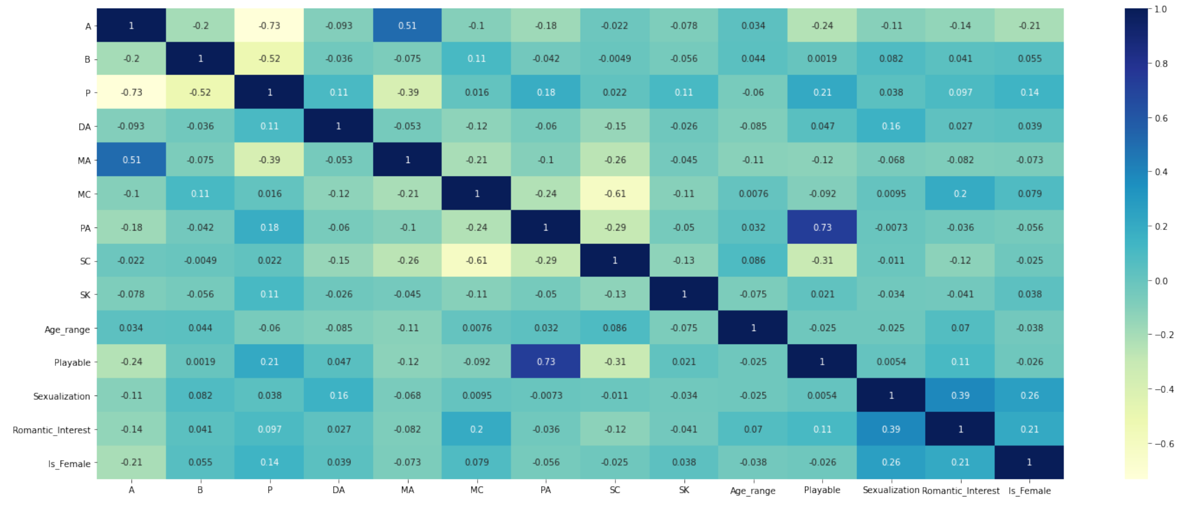
```
Sexualization:
            Mean            : 0.091
            Median           : 0
            Standard Deviation: 0.4
            Min             : 0
            Max              : 3
               Count 0    562
               Count 1     17
               Count 2     14
               Count 3      3

Romantic_Interest:
            Mean            : 0.054
            Median           : 0
            Standard Deviation: 0.2
            Min             : 0
            Max              : 1
               Count 0.0    549
               Count 0.5     30
               Count 1.0     17

Is_Female:
            Mean            : 0.34
            Median           : 0
            Standard Deviation: 0.47
            Min             : 0
            Max              : 1
               Count 0    :  393
               Count 1    :  203
```

Correlation Table 5/5

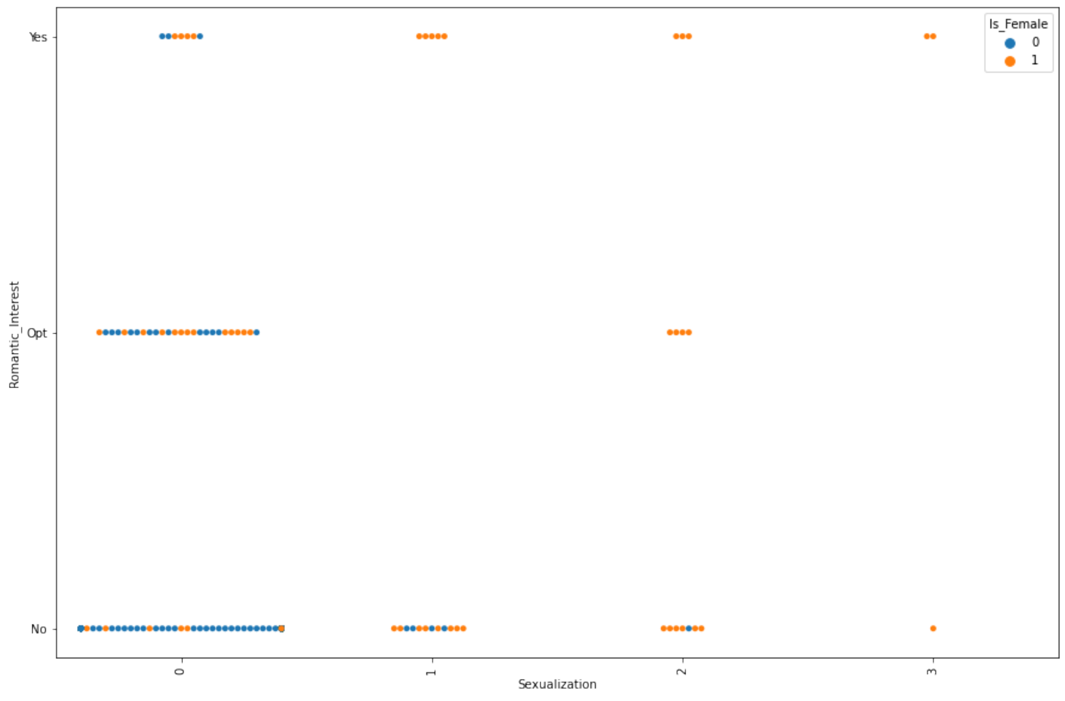
Give a screenshot of a **correlation matrix** of the numerical attributes.



	A	B	P	DA	MA	MC	PA	SC	SK	Age_range	Playable	Sexualization	Romantic_Interest	Is_Female
A	1	-0.2	-0.73	-0.093	0.51	-0.1	-0.18	-0.022	-0.078	0.034	-0.24	-0.11	-0.14	-0.21
B	-0.2	1	-0.52	-0.036	-0.075	0.11	-0.042	-0.0049	-0.056	0.044	0.0019	0.082	0.041	0.055
P	-0.73	-0.52	1	0.11	-0.39	0.016	0.18	0.022	0.11	-0.06	0.21	0.038	0.097	0.14
DA	-0.093	-0.036	0.11	1	-0.053	-0.12	-0.06	-0.15	-0.026	-0.085	0.047	0.16	0.027	0.039
MA	0.51	-0.075	-0.39	-0.053	1	-0.21	-0.1	-0.26	-0.045	-0.11	-0.12	-0.068	-0.082	-0.073
MC	-0.1	0.11	0.016	-0.12	-0.21	1	-0.24	-0.61	-0.11	0.0076	-0.092	0.0095	0.2	0.079
PA	-0.18	-0.042	0.18	-0.06	-0.1	-0.24	1	-0.29	-0.05	0.032	0.73	-0.0073	-0.036	-0.056
SC	-0.022	-0.0049	0.022	-0.15	-0.26	-0.61	-0.29	1	-0.13	0.086	-0.31	-0.011	-0.12	-0.025
SK	-0.078	-0.056	0.11	-0.026	-0.045	-0.11	-0.05	-0.13	1	-0.075	0.021	-0.034	-0.041	0.038
Age_range	0.034	0.044	-0.06	-0.085	-0.11	0.0076	0.032	0.086	-0.075	1	-0.025	-0.025	0.07	-0.038
Playable	-0.24	0.0019	0.21	0.047	-0.12	-0.092	0.73	-0.31	0.021	-0.025	1	0.0054	0.11	-0.026
Sexualization	-0.11	0.082	0.038	0.16	-0.068	0.0095	-0.0073	-0.011	-0.034	-0.025	0.0054	1	0.39	0.26
Romantic_Interest	-0.14	0.041	0.097	0.027	-0.082	0.2	-0.036	-0.12	-0.041	0.07	0.11	0.39	1	0.21
Is_Female	-0.21	0.055	0.14	0.039	-0.073	0.079	-0.056	-0.025	0.038	-0.038	-0.026	0.26	0.21	1

Distribution Visualization

Give a screenshot of a **scatter matrix** (color-coded by the label)



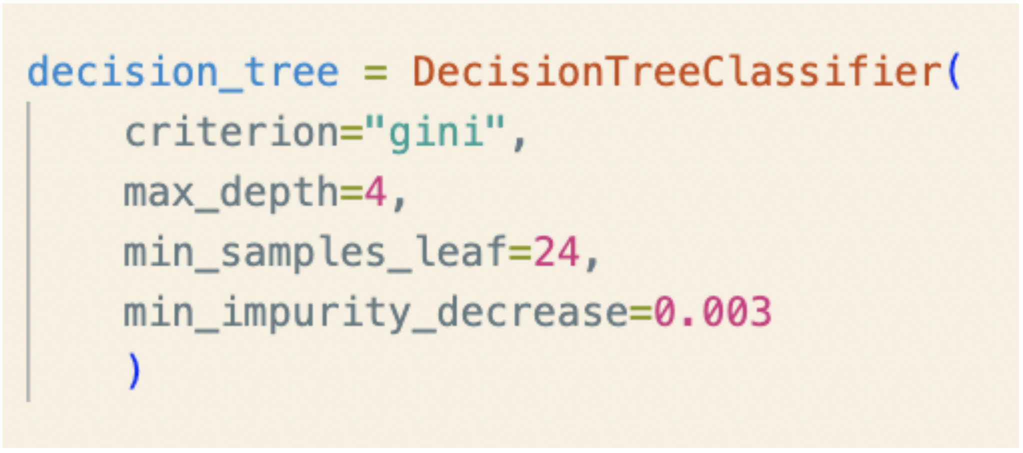
Classification Modeling

[!] In this assignment, the effort of getting high accuracy and the analysis is more important than the value of the accuracy itself. Getting almost 100% accuracy does not mean that you will get the full grade.

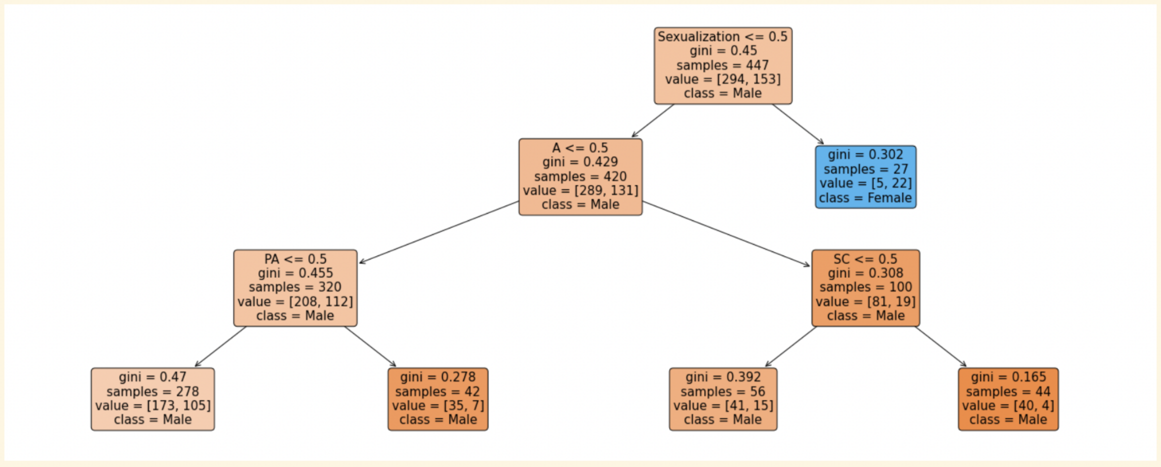
Base Model Preparation

Decision Tree 5/5 5/5

Perform classification on your model using the decision tree method. Adjust the pruning parameter to obtain the best performance. Give a screenshot of the model (the chosen parameters) and the resulting decision tree.



```
decision_tree = DecisionTreeClassifier(
    criterion="gini",
    max_depth=4,
    min_samples_leaf=24,
    min_impurity_decrease=0.003
    )
```



Best Model Accuracy: **71.141%**

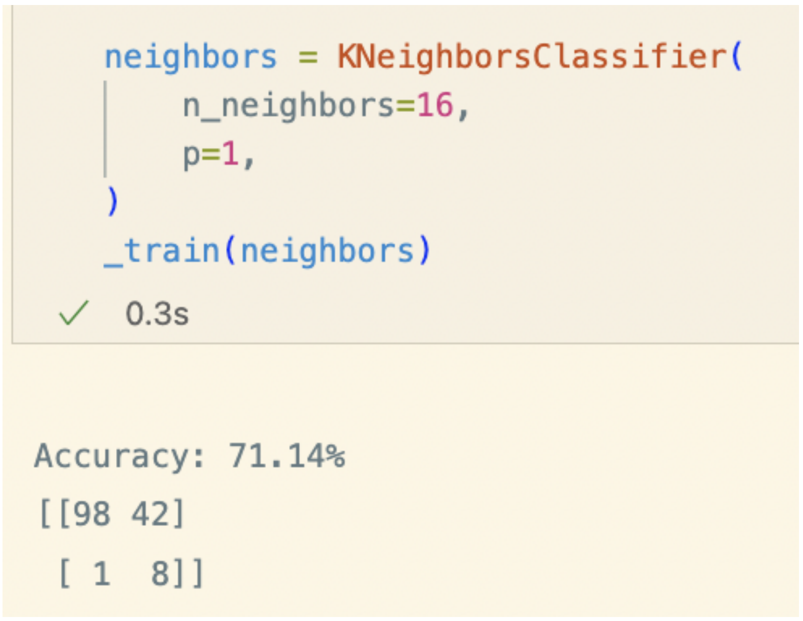
Analysis (30 - 100 words)

(Hint: Why do you choose those parameters for pruning?)

After multiple tries, I decided on these parameters because they gave the best accuracy while giving a simple tree. If we set the min_impurity_decrease to a lower value, the tree will become more complex, but the accuracy doesn't improve. If we increase the max_depth, the same behavior will happen. On the other hand, if we decrease max_depth, the tree will become too simple, and the accuracy will drastically reduce. If the min_samples_leaf is too high, we will obtain a tree of only one layer depth. When there is only one layer, every row is labeled "Male", even though with our final tree, we cannot get the caption "Female" after the first layer. We can observe that it is still a pretty bad representation.

● *k* Nearest Neighbor 5/5 5/5

Perform classification on your model using the *k*NN method using at least 3 different values of *k*. You can change the other parameter (distance for example) as well. Give a screenshot of those models (the chosen parameters) and their accuracy.



```
neighbors = KNeighborsClassifier(
    n_neighbors=16,
    p=1,
)
_train(neighbors)
✓ 0.3s

Accuracy: 71.14%
[[98 42]
 [ 1  8]]
```

Best Model Accuracy: **71.141%**

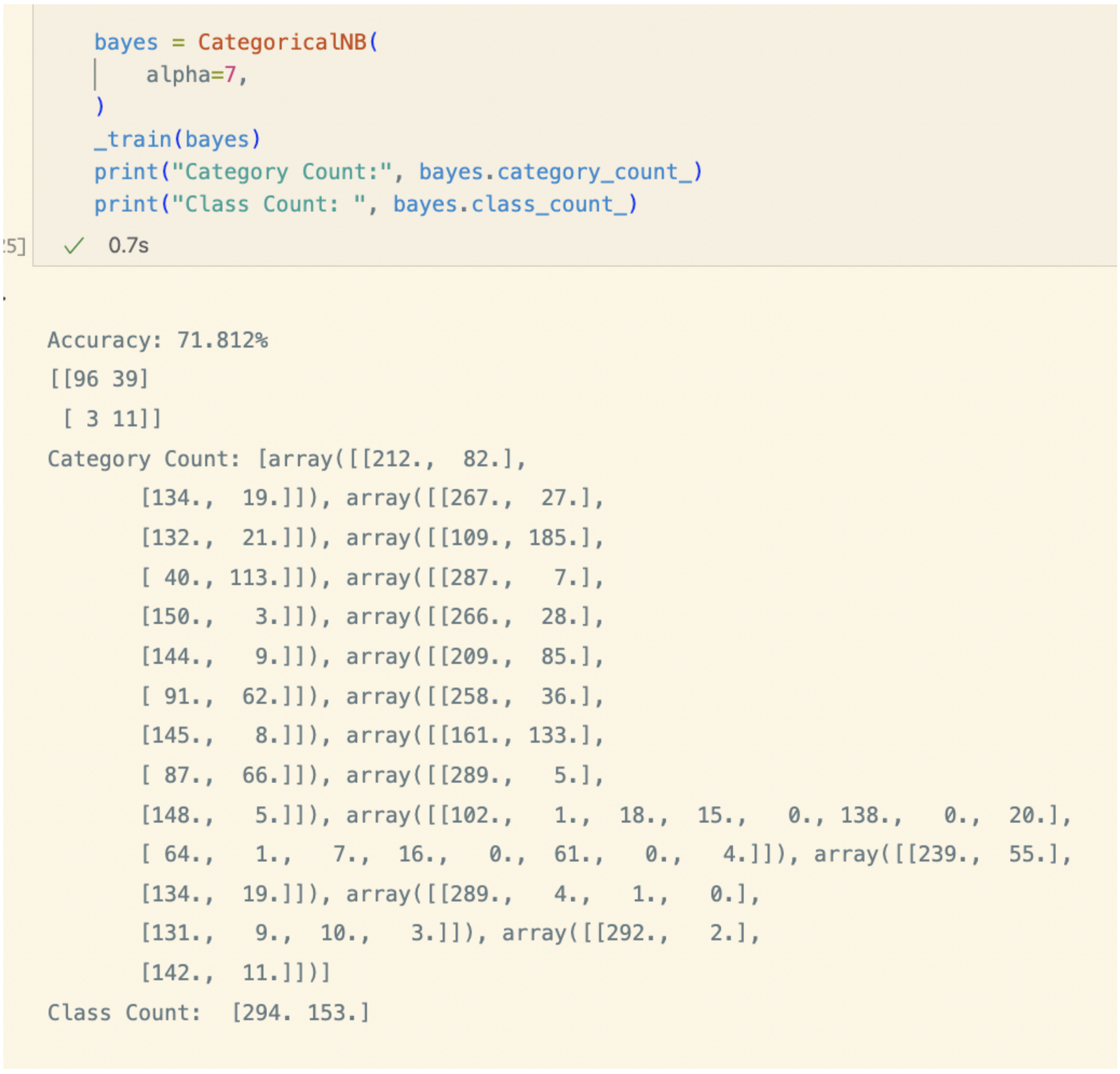
Analysis (30 - 100 words)

(Hint: Why do you think that the value of *k* of your choosing gives good prediction?)

As all our data are discrete and separated from a step-value equal to one, I fought that the Manhattan distance would be more optimal for this situation. I choose this value of n after testing all the integers between 1 and 100. A lot of values appeared to have the same accuracy. I decided then to take the lowest one, which seemed to be 16.

Naive Bayes 5/5 5/5

Perform classification on your model using the Naive Bayes method with Laplace correction. Give a screenshot of the distribution table (if you are using Python, showing category_count_ and class_count_ would suffice).



```
bayes = CategoricalNB(
    alpha=7,
)
_train(bayes)
print("Category Count:", bayes.category_count_)
print("Class Count: ", bayes.class_count_)
```

```
5]  ✓ 0.7s
```

```
Accuracy: 71.812%
[[96 39]
 [ 3 11]]
Category Count: [array([[212.,  82.],
       [134.,  19.]]), array([[267.,  27.],
       [132.,  21.]]), array([[109., 185.],
       [ 40., 113.]]), array([[287.,   7.],
       [150.,   3.]]), array([[266.,  28.],
       [144.,   9.]]), array([[209.,  85.],
       [ 91.,  62.]]), array([[258.,  36.],
       [145.,   8.]]), array([[161., 133.],
       [ 87.,  66.]]), array([[289.,   5.],
       [148.,   5.]]), array([[102.,   1.,  18.,  15.,   0., 138.,   0.,  20.],
       [ 64.,   1.,   7.,  16.,   0.,  61.,   0.,   4.]]), array([[239.,  55.],
       [134.,  19.]]), array([[289.,   4.,   1.,   0.],
       [131.,   9.,  10.,   3.]]), array([[292.,   2.],
       [142.,  11.]])]
Class Count:  [294. 153.]
```

Best Model Accuracy: **71.812%**

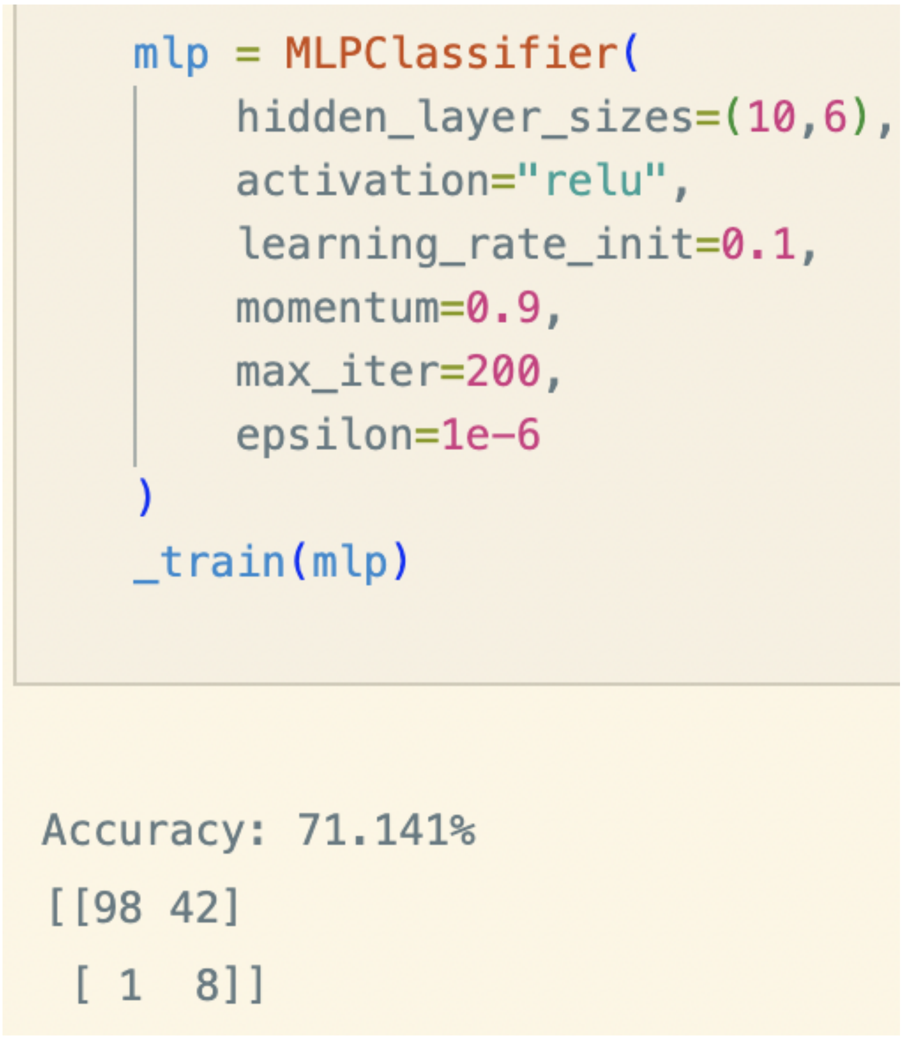
Analysis (30 - 100 words)

(Hint: Compared to Decision Tree and kNN, speculate why you get that accuracy)

This time, I got a little higher accuracy than the two previous models. Still, apriori this difference doesn't seem to be that significant. I think Naive Bayes performed better because it is better suited for categorical input variables than numerical ones. However, as we can see on the matrix correlation matrix, not all variables seemed to be 100% independent (for example, being the Main Antagonist (i.e., MA=1) necessarily implies that it is also an Antagonist (i.e., A=1)).

Artificial Neural Network 5/5 5/5

Perform classification on your model using the Artificial Neural Network. Adjust the parameter to obtain the best performance (at least set 3 parameters different than the default values). Give a screenshot of the model (the chosen parameters) and the visualization of the network, if any.



```
mlp = MLPClassifier(
    hidden_layer_sizes=(10,6),
    activation="relu",
    learning_rate_init=0.1,
    momentum=0.9,
    max_iter=200,
    epsilon=1e-6
)
_train(mlp)
```

```
Accuracy: 71.141%
[[98 42]
 [ 1  8]]
```

Best Model Accuracy: **71.141%**

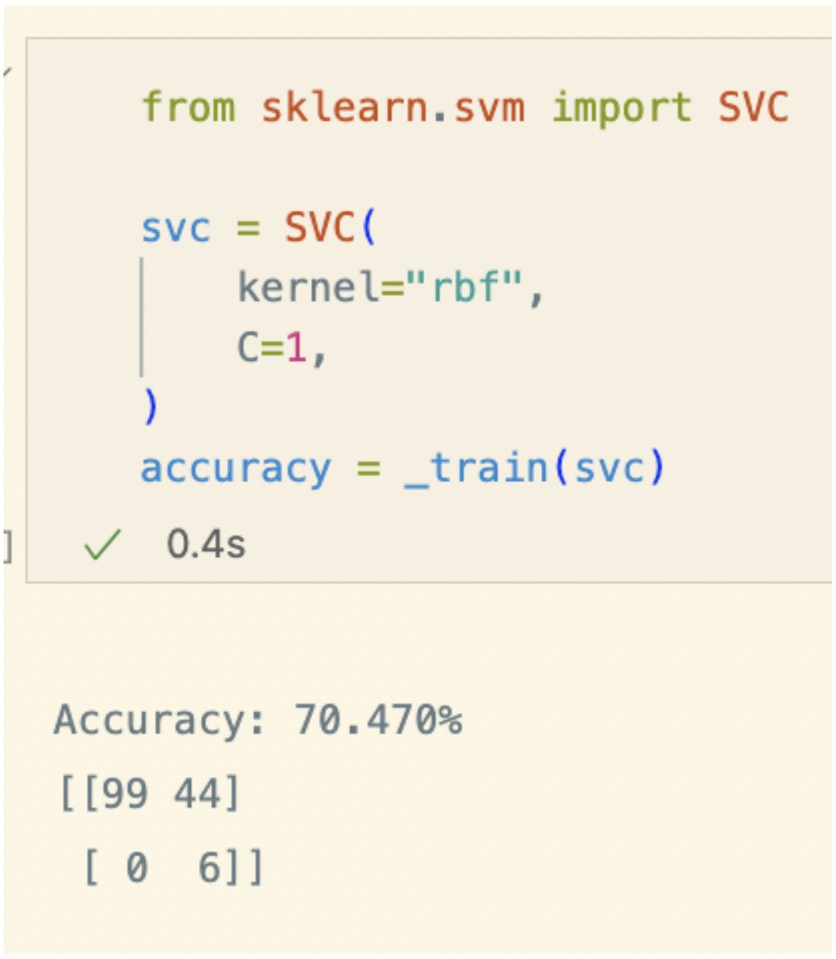
Analysis (30 - 100 words)

(Hint: Why do you choose those parameters for this ANN? Why do you think those numbers work in getting good accuracy?)

I chose to have two hidden layers because more would have added too much complexity to the model and reduced the performance. Moreover, the dataset isn't so complex, so two hidden layers should be enough. The input layer is 14 (as we have 14 features) and the output layer is 1 (Is_Female is True or False) so 10 and 6 as intermediate values appeared to be a good choice for the hidden layers. I choose to have a smaller epsilon to reduce the error. I also changed the activation function. Indeed, the model used ReLU, which is a better option than sigmoid and tanh in this context because the ReLU doesn't have the vanishing gradient problem and is a bit faster. I tried other settings, but this one gave me the best result.

Support Vector Machine 2/2 3/3

Perform classification on your model using the SVM model. Adjust the available parameters to obtain the best performance. Give a screenshot of the model (the chosen parameters).



```
from sklearn.svm import SVC

svc = SVC(
    kernel="rbf",
    C=1,
)
accuracy = _train(svc)
✓ 0.4s

Accuracy: 70.470%
[[99 44]
 [ 0  6]]
```

Best Model Accuracy: **70.470%**

Analysis (30 - 100 words)

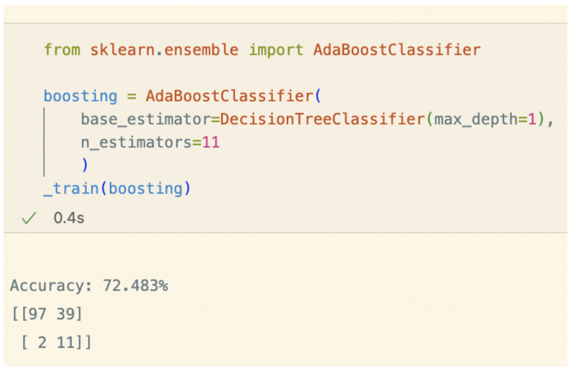
(Hint: Why do you think those parameters work for your problem? Does different kernel function give different results?)

After using different configurations, the default configuration works the best. The linear and rbf kernels gave the same result. However, it is not the case with the poly and sigmoid kernels, even when changing the degree or gamma attributes. We can conclude that our dataset is more likely to be linearly separable. However, the accuracy remains too low to assert that a Support Vector Machine is a good model.

Ensemble Modeling

Boosting 5/5 5/5

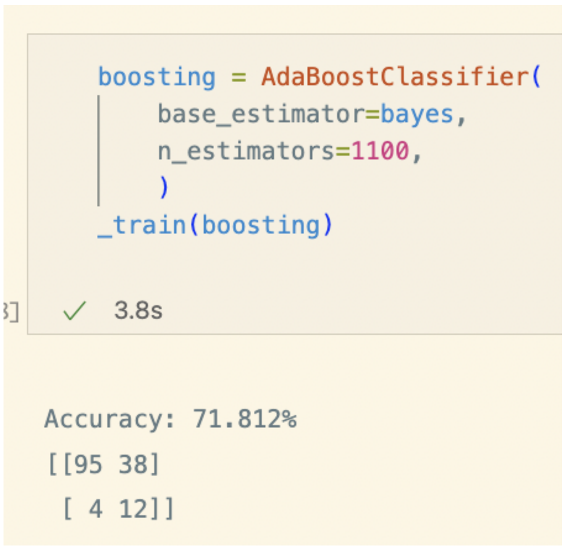
Perform classification on your model using any boosting method. Make two models: one using weak learner (such as Decision Tree with max_depth=1), and the other using the model you've made above that gives the best accuracy. Give a screenshot of the models (the chosen parameters) and their accuracy.



```
from sklearn.ensemble import AdaBoostClassifier

boosting = AdaBoostClassifier(
    base_estimator=DecisionTreeClassifier(max_depth=1),
    n_estimators=11
    )
_train(boosting)
✓  0.4s

Accuracy: 72.483%
[[97 39]
 [ 2 11]]
```



```
boosting = AdaBoostClassifier(
    base_estimator=bayes,
    n_estimators=1100,
    )
_train(boosting)
✓  3.8s

Accuracy: 71.812%
[[95 38]
 [ 4 12]]
```

Best Model Accuracy: **72.483%**

Analysis (30 - 100 words)

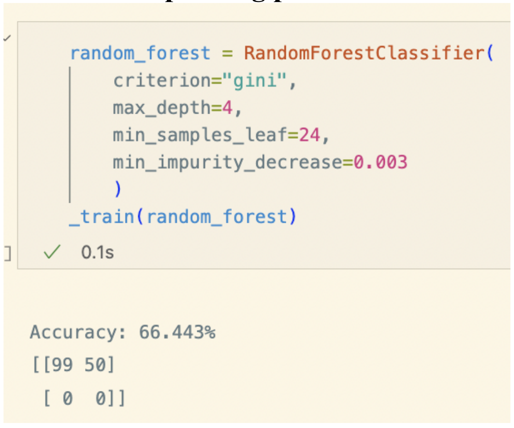
(Hint: How does your model perform compare to the weak learner?)

The model I made above performs poorly compared to the weak learner. It even performs the same way without the Boosting. Even though I explored different values of n_estimators, I remained with the same conclusion. This is probably due to the fact that we simply cannot predict the gender of a character based on our features. I will discuss this matter in the last section.

● Random Forest 5/5 5/5

Perform classification on your model using the random forest method. Make two models: one using the pruning parameter you've made in the Decision Tree, and the other one using a default/other value of your liking. Give a screenshot of the models (the chosen parameters) and their accuracy.

Same pruning parameters as in the Decision Tree

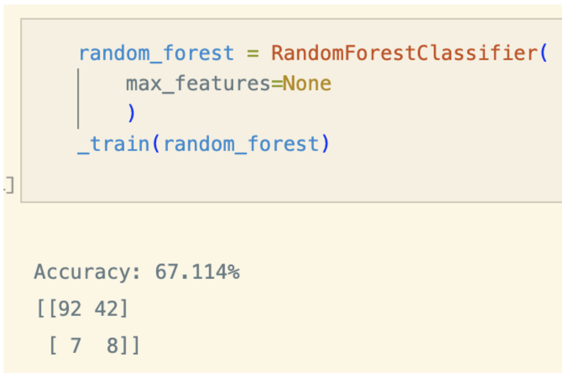


```
random_forest = RandomForestClassifier(
    criterion="gini",
    max_depth=4,
    min_samples_leaf=24,
    min_impurity_decrease=0.003
    )
_train(random_forest)
```

✓ 0.1s

```
Accuracy: 66.443%
[[99 50]
 [ 0  0]]
```

Other pruning parameters



```
random_forest = RandomForestClassifier(
    max_features=None
    )
_train(random_forest)
```

```
Accuracy: 67.114%
[[92 42]
 [ 7  8]]
```

Best Model Accuracy: **67.114%**

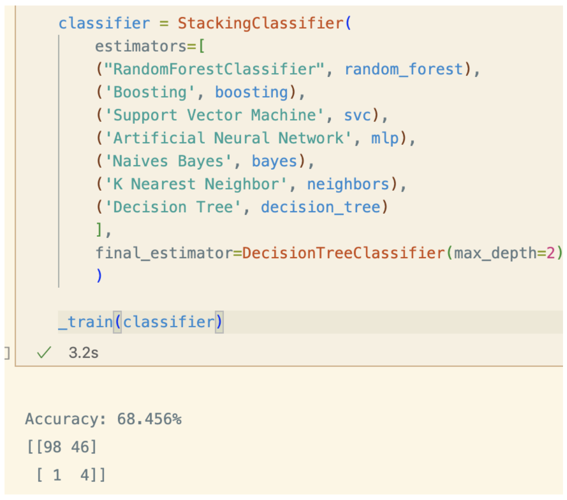
Analysis (30 - 100 words)

(Hint: Why do you choose those parameters for pruning?)

After changing and trying different parameters, I realized that this model was poorly performing. The accuracy oscillated between 62% and 66%. However, it was the highest when the max_features was set to None, which means `max_features=n_features`, which means the search for a split inspects all the features.

Stacking 5/5 5/5

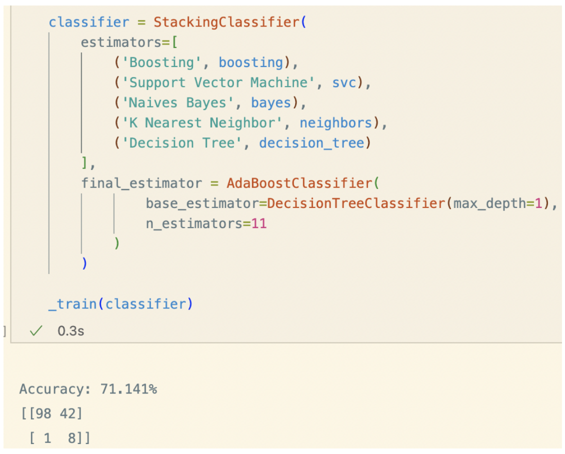
Perform classification on your model using the stacking method. Make two models: one using at least 4 models you have made in the "Base Model Preparation", and the other one is free, whatever it takes to get the best accuracy. The choice of the final estimator/meta-learner is up to you. Give a screenshot of the model (the chosen models with their parameters) and their accuracy.



```
classifier = StackingClassifier(
    estimators=[
    ("RandomForestClassifier", random_forest),
    ('Boosting', boosting),
    ('Support Vector Machine', svc),
    ('Artificial Neural Network', mlp),
    ('Naives Bayes', bayes),
    ('K Nearest Neighbor', neighbors),
    ('Decision Tree', decision_tree)
    ],
    final_estimator=DecisionTreeClassifier(max_depth=2)
    )

_train(classifier)
✓ 3.2s

Accuracy: 68.456%
[[98 46]
 [ 1  4]]
```



```
classifier = StackingClassifier(
    estimators=[
        ('Boosting', boosting),
        ('Support Vector Machine', svc),
        ('Naives Bayes', bayes),
        ('K Nearest Neighbor', neighbors),
        ('Decision Tree', decision_tree)
    ],
    final_estimator = AdaBoostClassifier(
        base_estimator=DecisionTreeClassifier(max_depth=1),
        n_estimators=11
        )
    )

_train(classifier)
✓ 0.3s

Accuracy: 71.141%
[[98 42]
 [ 1  8]]
```

Best Model Accuracy: 71.141%

Analysis (30 - 100 words)

(Hint: Does combining some different algorithms really gives a better prediction?)

In this case, combining different algorithms did not provide a better prediction. All the algorithms might not be independent or of enough good quality. However, I think it is more due to the fact that it is hard to predict the gender of a character in those conditions.

Final Conclusion (50 - 150 words) 3/5

(Hint: How did all the models perform? Which one gives the best result and why? What's your suggestion to improve the accuracy of the model.)

Even if all the different models' accuracy were close (mostly between 70 and 72), the model which performs the best is Boosting while using a weak learner. Indeed boosting provides a solution to combine many weak learners into one strong learner, by minimizing bias due to training records. I did this study to see if it was possible to identify the gender of a character based on other features such as their role in the story, age, sexualization, and representation of a love interest. We observed that it was hard for a model to answer this question, which means that the gender-biased in video games is lower than my expectations. To see, if this is really impossible, we should extend the database, which is rather small.

Best Model Accuracy (among all models): 72.483%