Project: The War with Star Wars.

Author: Gabriel Sanahuano

Elevator pitch

This has been an incredible project that has helped me to learn what Data Science means and its contributions to our society nowadays. I could say that it was the project in which I learned the most during this semester because it was the one in which I encountered the most obstacles. I think obstacles help us to build our knowledge and character. During the information analysis, I was able to understand a little more the importance of each element and tool of the data preprocessing and its contribution to the final product, which is the machine learning model. I would like to detail a little more about my experience when answering the grand questions of this project.

TECHNICAL DETAILS

GRAND QUESTION 1

- Shorten the column names and clean them up for easier use with pandas.

I was able to do this by applying the "replace" and "assign" functions from Python, and thus modifying the column names to fit better the data in the columns. I used the variable "variables_replace" to store the changes and then introduced them in the code.

GRAND QUESTION 2

- Filter the dataset to those that have seen at least one film.

I filtered the data to display only those respondents who saw at least one film. I was able to do that by implementing the following code.

```
haveSeenAtLeast1film = dat['seen_any'] == "Yes"
dataset = dat[haveSeenAtLeast1film]
print(dataset)
```

GRAND QUESTION 3

- Please validate that the data provided on GitHub lines up with the article by recreating 2 of their visuals and calculating 2 summaries that they report in the article.

GRAND QUESTION 4

- Clean and format the data so that it can be used in a machine learning model. Please achieve the following requests and provide examples of the table with a short description the changes made in your report.

One-hot encode all columns that have categories.
I used the following code to One-Hot encode the "view_" columns. I also implemented it with other categorical columns. It was very important to use the Pandas function "get_dummies()"

```
dat_view = dataset.filter(regex = 'view__')
(dat_view
.fillna(value = "Missing")
.apply(lambda x: pd.factorize(x)[0]))
dummies1 = pd.get_dummies(dat_view)

dummies1

dataset_view = dataset.drop(['view__han_solo', 'view__luke_skywalker', 'view__lando_calrissian',
'view__princess_leia_organa', 'view__anakin_skywalker', 'view__obi_wan_kenobi',
'view__emperor_palpatine', 'view__darth_vader', 'view__boba_fett', 'view__c-3p0', 'view__r2_d2',
'view__jar_jar_binks', 'view__padme_amidala', 'view__yoda'], axis='columns')

dataset_view.head(3)
```

Convert all yes/no responses to 1/0 numeric.

```
variables_replace = {
'Yes': 1,
'No': 0,
'Unfamiliar (N/A)' : np.nan
}
dataset = dataset.replace(variables_replace)

dataset.head()
```

Create an additional column that converts the income ranges to a number.
I was able to do this by renaming the column to "min_income" and replacing the symbols included in

the original values.

```
salary_stuff = (data1.household_income
.str.split(' - ', expand=True)
.rename(columns={0: "min_income", 1: "max_income"})
.min_income
.str.replace("$|,|+", "", regex=True)
.astype('float')
)
```

Create an additional column that converts the age ranges to a number.
I used the function the "concat()" function to join all the values encoded from the get_dummies() function.

```
age_range = (data1.age
.str.split('-', expand=True)
.rename(columns={0: "min_age", 1: "max_age"})
.min_age
.str.replace("> ", "")
.astype('float')
)
```

```
merged = pd.concat([data1, onehot_stuff, onehot_stuff2, onehot_stuff3], axis="columns")
```

Create an additional column that converts the school groupings to a number.

```
onehot_stuff2 = pd.get_dummies(data1.education)
```

GRAND QUESTION 5

- Build a machine learning model that predicts whether a person makes more than $50k.

This was the interesting part of the project and where I found most of the obstacles. I was able to convert all of the categorical values into numerical data by encoding them, and also I min-max scaled the features to prepare them for the model. But for some reason, I wasn't able to get a high accuracy score even though I ran two models, the decision tree and Kneighbors classifier. My score was 30% accurate, which is not good at all. I will have to request some feedback here to know the reason why I couldn't get higher accuracy.
I had provided evidence of all of my work and code in the google Colab notebook that I attach the link below.

APPENDIX A (PYTHON SCRIPT)

https://gist.github.com/gabecastri/bcf77407c958409c12ab9115ba92a6e4