Project 1 – Whitepaper Draft
Jerica Tripp – DSC680
July 4, 2022

Business Problem

Can a site know their likelihood of winning 'Site of the Day' on Awwwards.com based on their site design before applying for an award? The reason web development and web design companies seek out awards is so they can be recognized as an expert and thus have credibility with their customers. GoDaddy lists Awwwards.com as first on their list of the most popular awards among freelance web developers and designers (Bobriakov, 2022). "Awwwards.com" (n.d.) states on their site: "Every month, more than 2,000,000 unique users who specialize in web design, development and the latest web technologies, visit our site looking for inspiration, trends and talent on the web."

Applying for these awards is not free, so having a way to predict likelihood of winning would help to save potential applicants money. Knowing the likelihood of winning and knowing the features of historical winners could help designers make better design choices as well.

Background/History

Awwwards.com awards 'Site of the Day' to one submitted website every weekday. A jury of 18 jury members review the sites submitted. The process of voting can take up to 5 days or longer based on weekends & holidays. Scores are given in the following categories: design, usability, creativity, and content. Users of Awwwards.com can also vote on winners. The scores are only revealed to the winners. It costs $65 to submit once or you can pay $165 a year a Professional account that allows users to submit for free.

Data Explanation

Data Gathering

There is no export option on Awwwards.com to collect the winning and nominated site images and data. In order to gather this information, I had to web scrape the site. I wrote a small script in JavaScript that iterated over the pages of winners and downloaded the site image, site name, site country, site URL, and date. I wrote the data to a csv and saved the images with a hash name. I used hashes so I could relate the row of data in the CSV back to the image saved. I performed the same actions for the nominee's data as well, but I saved the nominees data in a new folder.

Data Prep

My data preparation began after I had downloaded all the data available for winners and nominees. My first step was to compare the CSV data of the two datasets (winners and nominees) to ensure I had similar date ranges. I found that I had more dates in my winner's dataset than my nominees, so I trimmed the winners to match the nominees date range.

The second step was to remove winners from the nominee's dataset. Winners would still show up on the Awwwards.com site in the nominee's section even if they had gone on to win site of the day. To

remove them, I first identified matching records in the CSV for winner and nominee where the site name was the same. I felt safe doing this, because even if I removed data for a site that really submitted more than once, the nominee's dataset was still sufficiently large. After this step I had 8,946 records in the nominee's dataset.

The third step was combining the data into one dataframe. I added an indicator on the winner's dataset called 'winner' and set it to 1. I add this same indicator on the nominee's dataset and set it to 0. I then concatenated the dataframes into 1.

The fourth step was to remove any images that did not download successfully. Some images that I attempted to download returned with '404' errors. I created a new field in my dataset called '404_error' and set it to 1 only when the returned image type was 'None', indicating an image error. I identified 210 images with errors. I created a new dataframe with only records in my dataset where 404_error equaled 1.

The last step was to create 50 pixels by 50 pixels thumbnails of my images and saved those to my machine. I did this because the images varied in their size, and I needed a smaller image to be able to iterate over each pixel. I used the thumbnail method from PIL (Python Imaging Library) to do this. I used the default 'NEAREST image resampling option to generate the thumbnails using k-nearest neighbors.

Feature Engineering & Class Balancing

The remaining steps of my data preparation was to create new features to help in my model. The first feature I created was the count of unique colors used in each image. For every thumbnail, I iterated over every pixel to create a unique list of the images RGB values and the count of the occurrences of each. Figure 1 below illustrates this new feature.

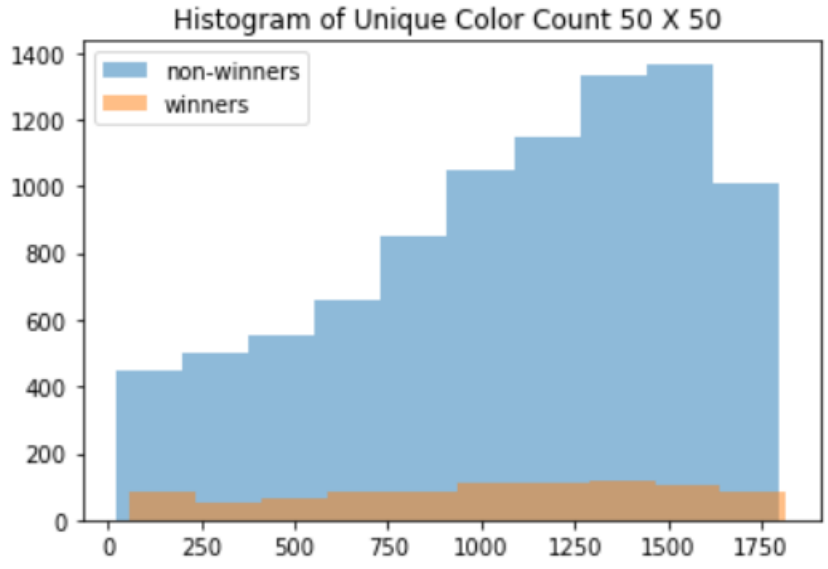

Figure 1

I also added a feature that was the average color of the thumbnails. To do this, I used numpy to average the total numeric R, G, & B values of the image. I converted the returned tuple to a hexadecimal color value. Figure 2 shows the distribution of the average color's values of the images.

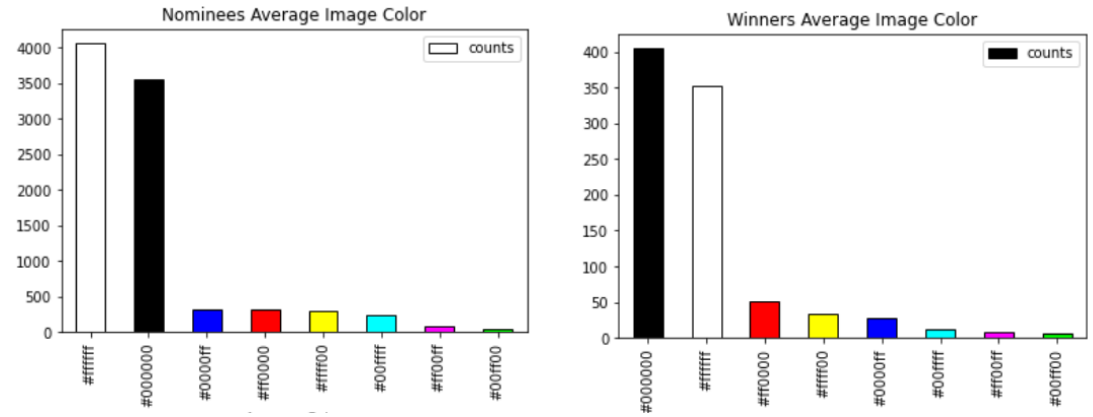

Figure 2

The next feature I added was 'Dominant Color'. The most dominant color is the most frequent color in the given image. I used the library 'ColorThief' which had a method called 'get_color' which returns an array of the RGB value of the most frequent color in an image. Again, I converted the returned RGB array into the hexadecimal value of the color.

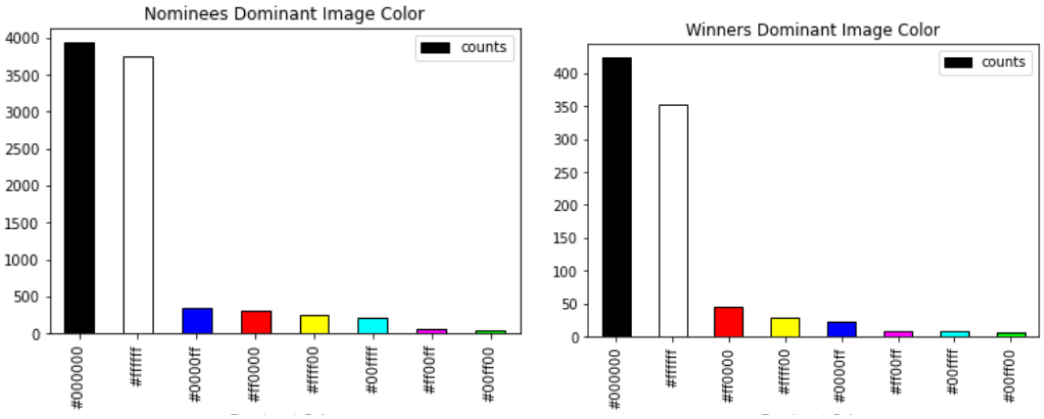

Figure 3

I also converted the date field into weekday, day, year, and month values. I also converted the hexadecimal color values to encoded integers. I had to remove data for Saturday and Sunday as there was a low volume with a high distribution from the winner's dataset.

My winner and nominees classes were imbalanced, so I had sampled with replacement from my winners until my classes were balanced. Figure 3 illustrates the before and after balancing.

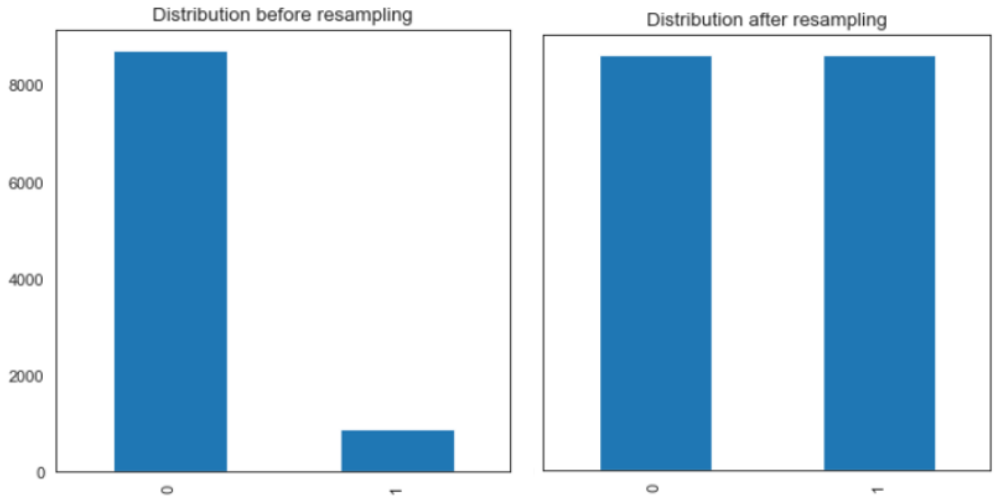

Figure 4

Data Dictionary

Main Data

1. Hash: Unique hex key to link image to record. Hash was created using site name and date.
2. Site_nm: Name of the website.
3. Date: Date site identified as a nominee or date site won site of the day.
4. Day: Day value of Date field.
5. Weekday: Weekday value of Date field.
6. Month: Month value of Date field
7. Year: Year value of Date field.
8. Country: Geographic country of site.
9. Img_url: Full URL to site image submitted.
10. Img_full_nm: Full name of image file.
11. Winner: Indicator of if the site won site of the day (1) or not (0)
12. 404_error: Indicates if image has an error. (0-no error, 1- error)
13. Unique_color_count_50: Count of unique colors in the image thumbnail.
14. Average_color: Average color of the image thumbnail.
15. Dominant_color: Most frequent color of the image thumbnail.
16. Average_color_enc: Encoded value of average color field.
17. Dominant_color_enc: Encoded value of average color field.

Methods

For my prediction model, I will be using K-Nearest Neighbors. My problem is essentially a classification problem, I'm attempting to categorize unknown images as either winners or not. "The k-nearest neighbors (KNN) algorithm is a data classification method for estimating the likelihood that a data point will become a member of one group or another based on what group the data points nearest to it belong to" (Joby, 2021). I will be separating my data into 25% validation and 75% training populations. This will help to gauge how well (or not) my model is performing.

Analysis

My analysis of the model will be conducted using the output from the training and testing runs. Figure 5 below shows the output from one such test. 2 Seems to be the optimal number of nearest neighbors for the algorithm. In the appendix, I've added the full classification report as well as the feature analysis.

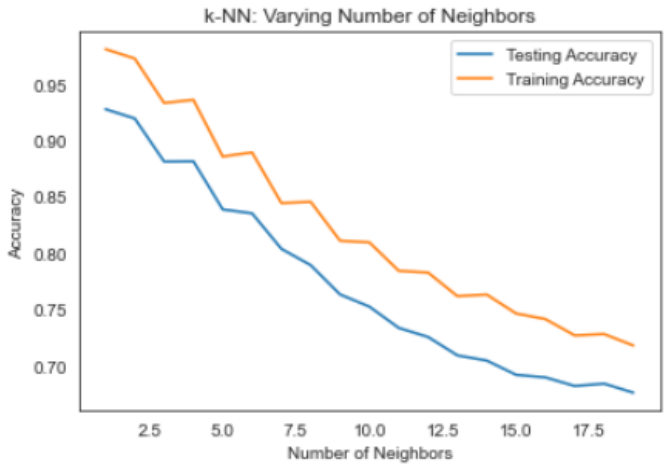

Figure 5

Conclusion & Implementation

After reviewing the impact of the features on the model, I've determined that they are all useful but unique color count seems to have the highest impact on likelihood to win. The next step in this project would be to build an application where users can upload images of their sites and get suggestions on how to improve before submitting to Awwwards.com

Ethical Assessment

This project did use data that was scraped without permission. It would not be ethical to attempt to profit off of this data. If I did create an application, it would only be for academic purposes. It may not be ethical to allow users to attempt to "game" the Awwwards site in order to win.

Sources

Awwwards.com. Choose an option to submit your site. (n.d.). Retrieved June 25, 2022, from https://www.awwwards.com/submit/

Bobriakov, I. (2022, March 11). *How to apply for awards & stand out like A pro*. GoDaddy Blog. Retrieved June 25, 2022, from https://www.godaddy.com/garage/apply-for-awards/#:~:text=Winning%20a%20web%20design%20award,also%20for%20web%20design%20inspiration.

Joby, A. (2021, July 19). *What is K-nearest neighbor? an ML algorithm to classify data*. Learn Hub. Retrieved June 25, 2022, from https://learn.g2.com/k-nearest-neighbor#:~:text=The%20k%2Dnearest%20neighbors%20(KNN,nearest%20to%20it%20belong%20to.

Appendix

Classification Report

	precision	recall	f1-score	support
0.0	1.00	0.91	0.95	2204
1.0	0.92	1.00	0.96	2251
accuracy			0.95	4455
macro avg	0.96	0.95	0.95	4455
weighted avg	0.96	0.95	0.95	4455

Feature Impact Analysis

Features Used							Max Score
day	month	year	country_enc	average_color_enc	dominant_color_enc	unique_color_count	0.9549
	month	year	country_enc	average_color_enc	dominant_color_enc	unique_color_count	0.903
day		year	country_enc	average_color_enc	dominant_color_enc	unique_color_count	0.9526
day	month		country_enc	average_color_enc	dominant_color_enc	unique_color_count	0.9549
day	month	year		average_color_enc	dominant_color_enc	unique_color_count	0.9549
day	month	year	country_enc			unique_color_count	0.9567
day	month	year	country_enc				0.8727
				average_color_enc	dominant_color_enc	unique_color_count	0.8563
			country_enc	average_color_enc	dominant_color_enc	unique_color_count	0.8851
				average_color_enc			0.5367

					dominant_color_enc		0.5192
						unique_color_count	0.7495
day							0.5015
	month						0.4949
		year					0.5048
			country_enc				0.6159