# Contemporary Art Price Prediction with Machine Learning

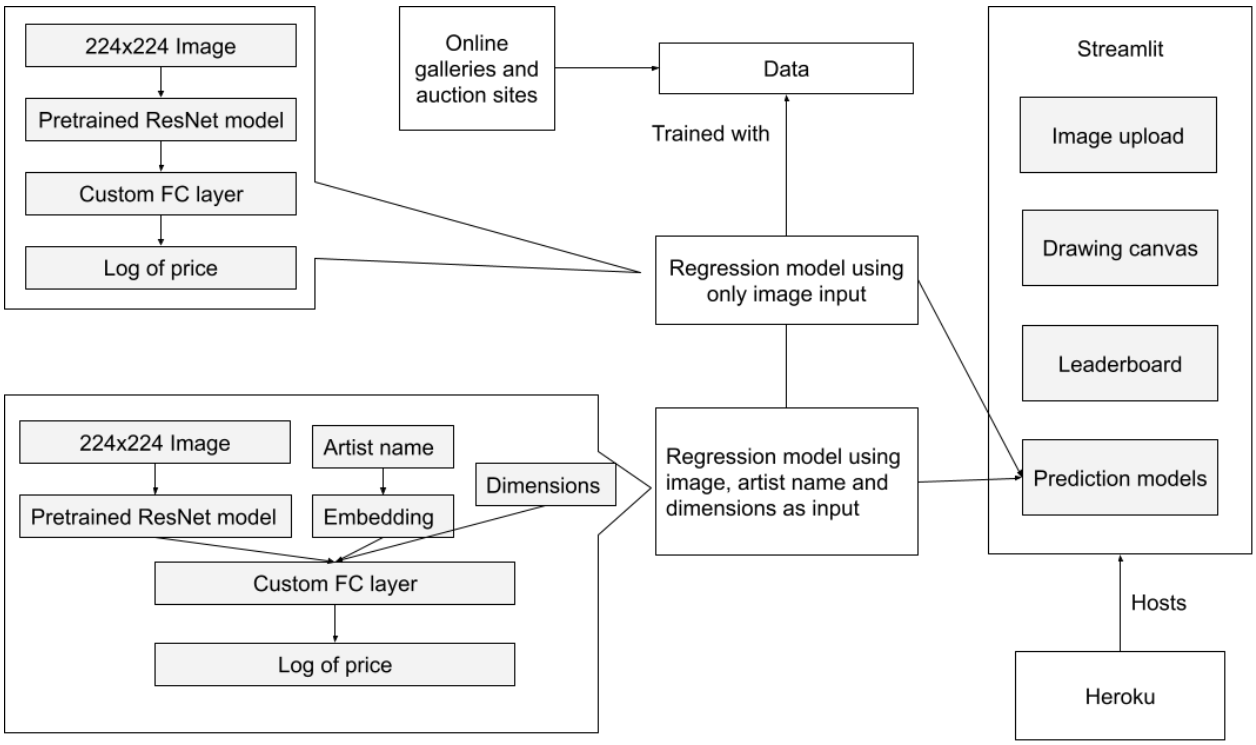
Jared Chan, Dylan Pither, Jane Shi, Eva Wang

## 1. Introduction

For the average person, the value of a contemporary art piece can be hard to estimate and grasp. A simplistic painting can be more expensive than a complex art piece that appears to have taken more skill to create. A seemingly unorderly collection of lines and colors that could have been drawn by a child might sell for millions. Our project looks to answer the question of what makes a contemporary art piece so valuable by evaluating and predicting how much a given art piece might sell for, through analyzing only the visual appearance of the painting, or with added information such as artist names or dimensions of the painting. Our project allows users to either draw or upload an art piece to be evaluated on our website by one of 2 models. The first model predicts a price given only an image of the art piece. The second model judges an art piece given additional information about the artist and size of the art piece. Both models use pre-trained ResNet models to evaluate the images. Additionally, our website has a leaderboard that ranks users on uploaded or drawn art pieces. The front-end web page was created using Streamlit, and is being hosted through Heroku.

## 2. Components

Our final product consists of a web front-end made with Streamlit hosted on Heroku and two models trained using PyTorch with data we collected from various websites.



# A. Data Collection and Cleaning

To train our models, we needed a large dataset. We gathered our data from popular art auction sites such as artsper.com, christies.com, artsy.net, and liveauctioneers.com, and online art galleries such as saatchiart.com, artmajeur.com, and gallerytoday.com. We chose these sites since they provide free information on the prices of artwork, artists, as well as artwork dimensions. We decided to focus on paintings as they could be easily represented in an image and training models with image data could be done in a straightforward manner.

Data was collected from these sites using python notebooks and the Beautiful Soup libraries to scrape image URLs and their prices, as well as artists and dimensions for each art piece. This information was put into pandas data frames and exported as CSV files. We then used another notebook to combine the CSV files and download all the images to form the final dataset. Early on we also found some duplicates and data images that were not art pieces so we took out those data points.

The prices of paintings we collected range from tens of dollars to hundreds of thousands of dollars. We wanted to create a model that judges a painting based mostly on its aesthetics, and since the price tags of paintings that sell for hundreds of thousands of dollars are likely affected by the artist's reputation, we limited our paintings' prices to less than or equal to twenty thousand dollars.
We found that the price of more than a third of our data was at $1710. This skewed the distribution of prices in our dataset and caused our early models to appear to perform better. We downsampled the data in this price range to obtain a more realistic representation of the prices of paintings in the real world.

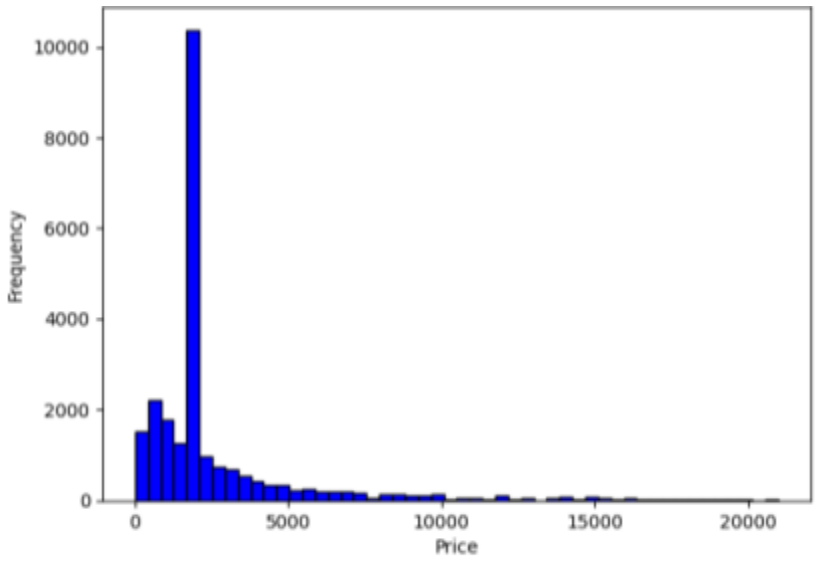



Figure 1. Distribution of price before downsampling

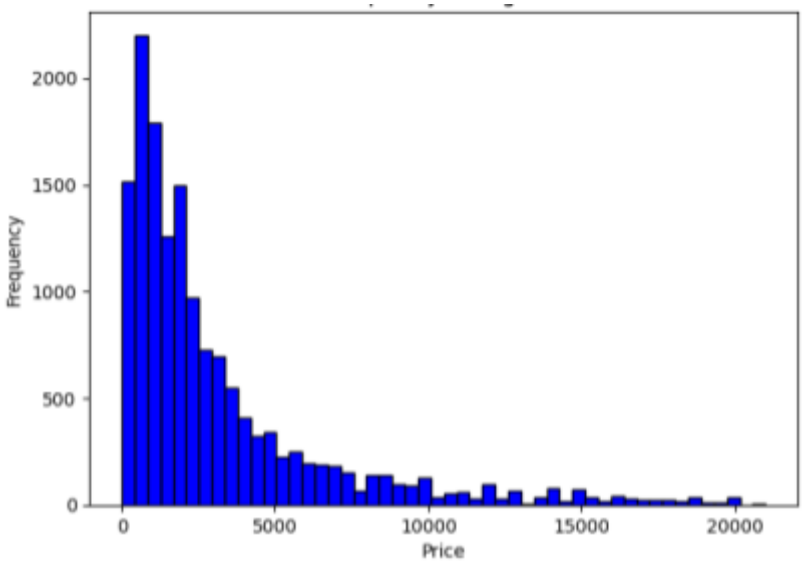


Figure 2. Price distribution after downsampling

We collected a huge quantity of data, but the quality of our data had a lot of room for improvement and there were other challenges that came with having large amounts of data. It took a long time to train a model with all of the images in our dataset, limiting our ability to test different parameters and model layouts. Further, since we primarily used Google Colaboratory to train our models, we were also subject to connection timeouts or slow file loading times. We worked around these problems by training with only subsets of the dataset when testing different training configurations and confirmed our models' performance at the end using the whole dataset. Being able to fully utilize our whole dataset would definitely increase training efficiency.

The quality of our data was limited to what was available on the internet. While some art auction sites and online galleries provide free information, most online databases require subscription fees. Moreover, when we were collecting additional information such as artist names and painting dimensions, we were only able to gather data from less well-known artists. Since one part of our website involves choosing an artist to make a specific prediction, having famous artists as choices would make our website more attractive and useful to people without much exposure to the contemporary art scene.

# B. Models

## I. Regression models

Initially, we tested and trained various models using pre-trained ResNet models such as resnet18, resnet34, and resnet50 connected to a custom fully connected layer. We tested various types of fully connected layers and used ReLU to connect linear layers as well as dropout to reduce overfitting. However, we quickly found that simpler (1 linear module) fully connected layers provided us with the

best loss. Additionally, we found the simpler resnet models (resnet18 and resnet34) worked the best for us.

However, we still were not getting the best loss and so we began experimenting with different models that included information about artists and dimensions. At first, we tried a single fully connected layer that took as an input a concatenation of a single output from a pretrained resnet model that took the image as input, a single output of an embedding that took artists as input, and finally the two dimensions of the painting. We also experimented with increasing the number of outputs from the embedding and ResNet fully connected layer. This proved beneficial for loss, with more outputs from the image module being optimal. We also tried a model which concatenated a one-hot encoding of the artists to the output of a ResNet model which had an identity fully connected layer, this model benefitted from having more linear layers, unlike the previous models we had worked with. Both of these models produced better loss than purely image-based models.

## II. Classification models

We also tried using classification models because classification seemed to be an easier task for machine learning models. We wanted our predictions to be as close to continuous as possible since having a predicted value is more useful than having a predicted range. Therefore, we divided our data into as many classes as possible, to retain the impression that our model's predictions are continuous. We finally tested with models that classify 10, 20, and 50 classes.

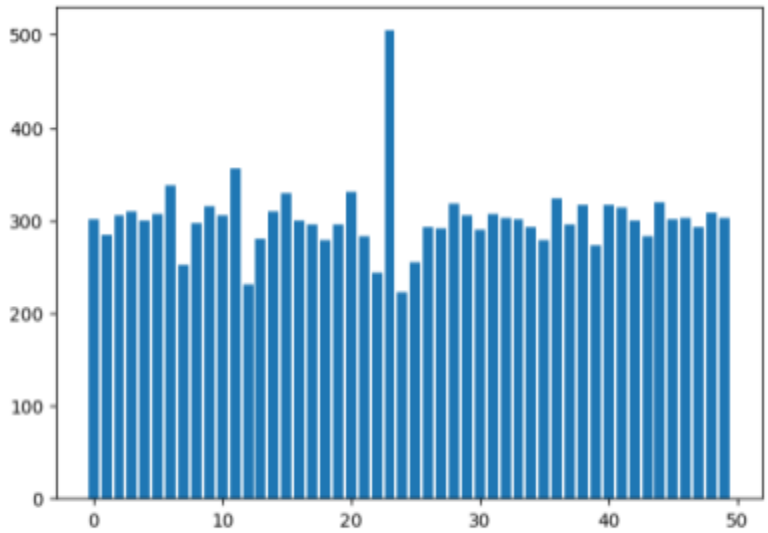



Figure 3. Distribution of data in 50 classes

When dividing our data into classes, we took into account that the prices of paintings were not evenly distributed from $0 to $20,000 (as shown in Figure 2). We made sure that all the classes contained roughly the same amount of data to ensure that our data was not biased.

To compare our different models, we made a wrapper for our classification models since their reported accuracies could not be directly compared with the test losses of our regression models. With fair comparison, we could decide which models we should use for our final product.

Other than the output shapes, the classification models' structures are very similar to our regression models. However, we did not test using additional information such as artist names and dimensions, since our classification models performed worse than our regression ones.

## C. Website

Our website is made up of four components: a direct price prediction, a price prediction that uses additional information, a drawable canvas board, and a competition leaderboard.

When selecting the price prediction tab, users will upload a picture of their painting to the website, and get a predicted price back right away. This serves people who already have a picture and want to quickly evaluate it. Our model only accepts specific inputs, so we always need to resize the uploaded images and switch them into tensor form before letting them enter the model. Moreover, we have also considered the situation when users submit a grayscale image. In that case, the image will first be converted into a regular RGB image, and then transform to feed into the model.

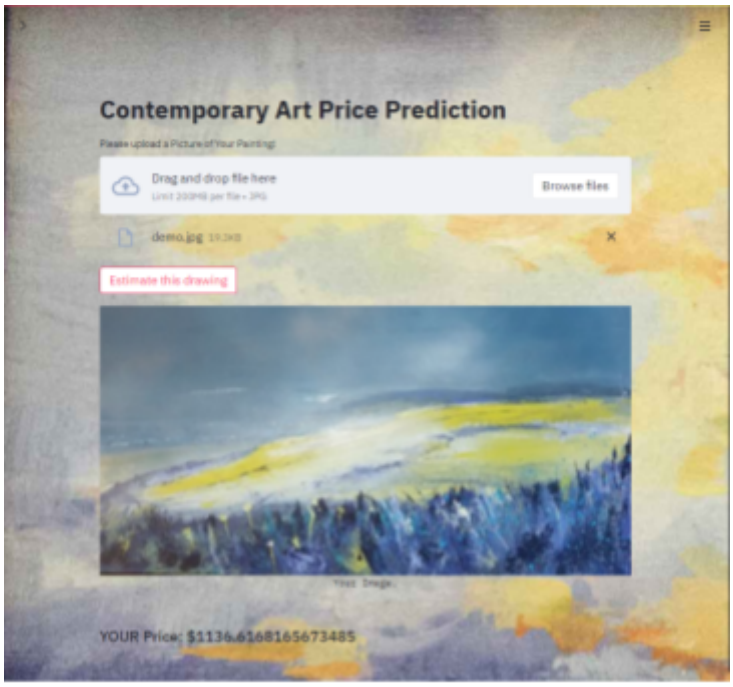



Figure 4. Simple price prediction

Users can also do a more accurate prediction on the website by selecting the second tab. This tab uses a different model than price prediction which takes in two more inputs: artist names and image dimensions. Users can select among more than 5000 artists, and choose dimensions ranging from 10 inches to 100 inches. The resulting price will not only depend on the picture itself but also other important information

about it. Artist names are transformed to numbers based on their index, and dimensions are both in two decimal places to make the prediction more precise. Before feeding into the model, all the inputs are properly converted to their tensor form.

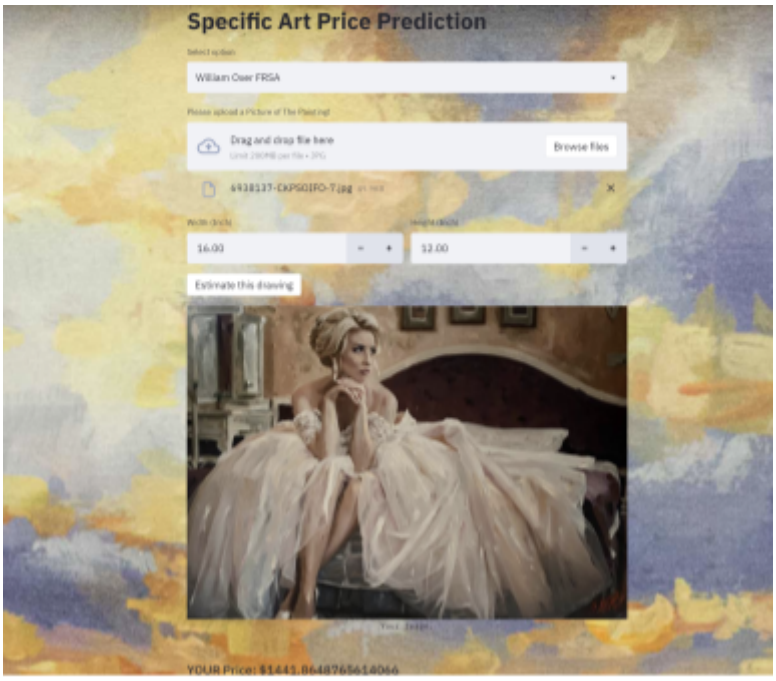



Figure 5. Accurate prediction

We have created another option for users to make their own artwork on a drawable canvas, which they can check the price immediately after they finish. The canvas is very flexible. Users are able to choose the stroke, background, and drawing tools they like. The RGBA picture they draw is then stored as a 4D NumPy array. Since it contains one more channel of information than we need, part of the image data

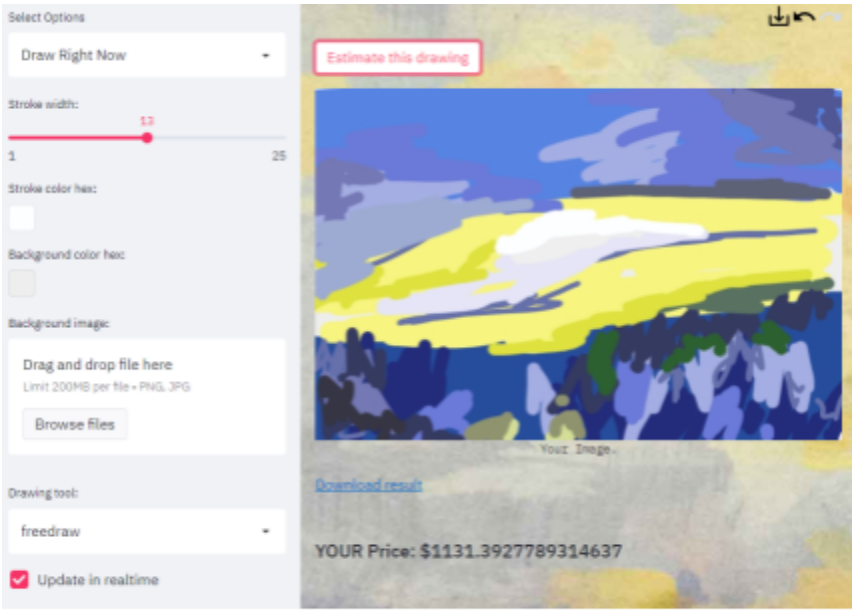



Figure 6. Drawable canvas

related to alpha will be ignored, and then our model could predict the price as usual. We have also added a download feature to the website if the users want to save the images they draw.

The website includes a competition leaderboard that records and ranks the price of people's artworks from high to low. After users finish submitting, they need to refresh the page to see the change that happened in the leaderboard. Each time someone uploads an image to the leaderboard, the image will be reshaped into a 2D matrix and saved as a .csv file in the submission folders. Additionally, the result of the competition will be stored in a file called leaderboard.csv which has four columns containing information such as participants' name, score, and last access time. If a participant uploads multiple times, all of his images will be saved in the submission folder, but only the highest price will be recorded in the leaderboard to be fair, and the number of attempts he tries will be written to the 'entry' column of the leaderboard.csv to better organize it. Since the average price of drawings is about $ 1500, users could also challenge to see if they are above average or not.

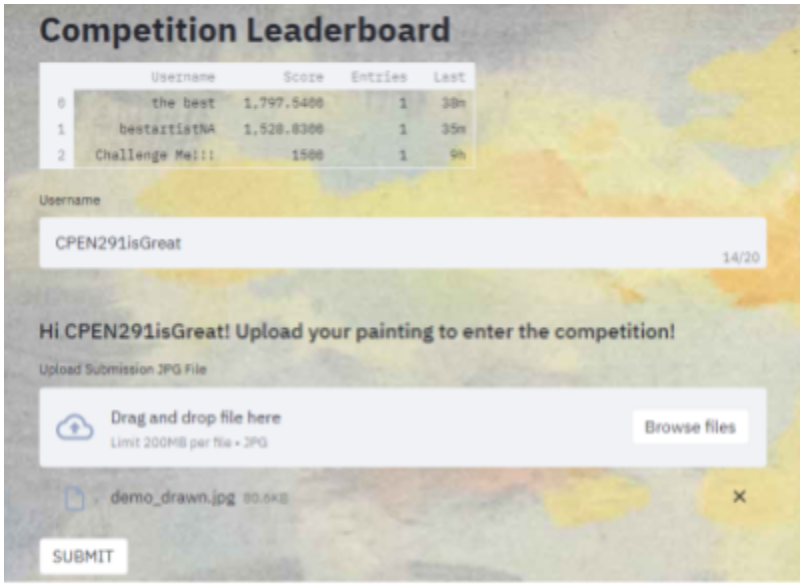



Figure 7. Leaderboard

Finally, for convenience, we made a selection box for users to choose between these different options.

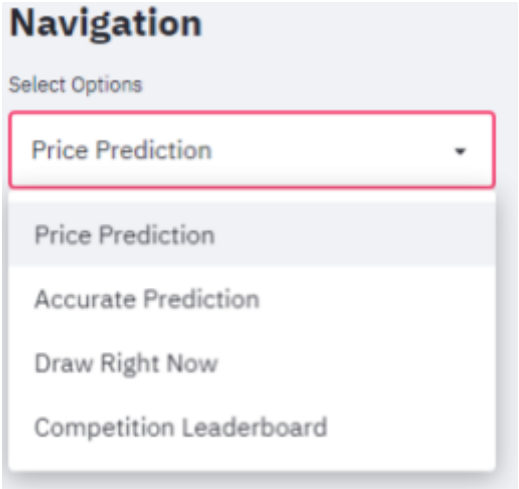



Figure 8. Navigation menu

# 3. Conclusion

What makes a contemporary art piece so valuable? Although our models did not provide an answer, we can conclude that it is not all subjective. Artist reputation does play a big role in determining the value of an art piece, but our models have shown that even when we take all of an artwork's story out of the picture, we can still gauge with moderate accuracy the price of paintings.

# 4. Contributions

Dylan:

- Scraped data from websites (both with artists and dimensions and without)
- Trained both types of models
- Recorded and edited final video

Eva:

- Scraped data from auction and gallery sites (both with artists and dimensions and without)
- Trained models
- Build the web app
- Host on Heroku.com

Jane:

- Scraped data from auction and gallery sites (both with artists and dimensions and without)
- Trained models
- Helped with building the web app

Jared:

- Scraped data from websites (mostly without artists and dimensions)
- Organized and cleaned data into more usable formats
- Trained both types of models
- Helped build the web app