Why are the top Users at the top Of r/place 2022

Tyler Pham

1 Who are the top users?

1.1 Why so many pixels?

The top 3 by number of pixel placements placed 795, 781, and 777 pixels respectively. From here onward, the top 3 users will be referred to by their place. I hypothesized that the top users would consist of either bots as they could continuously place pixels without breaks or moderators that could place multiple pixels at once with no wait time. To determine if these users were moderators or bots, I viewed their time between placements. The gaps and variation in placements by hour disprove the possibility that these users were bots and point to their humanness. Additionally, these users never broke 12 placements per hour which suggests these users were not moderators.

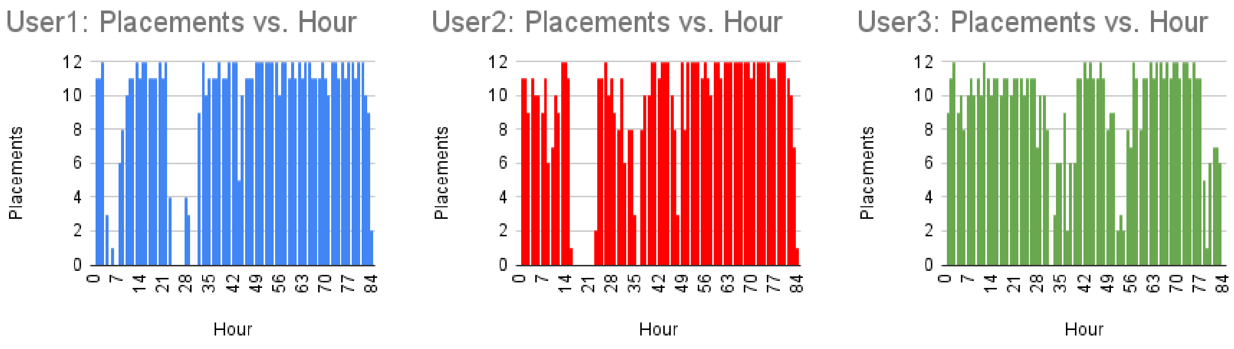


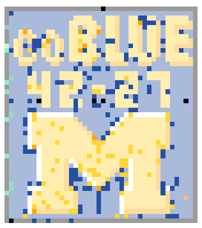
Figure 1: User Placements By Hour Since Start

1.2 Why so much activity?

Given the high probability that these users were not bots or moderators, I assumed that these users must be extremely active members of their communities. With this in mind, the user activity should be focused on specific artwork or community to protect them. Surprisingly, the activity of user 1 and user 2 was somewhat distributed, but user 3 activity was almost solely focused onto one area of the canvas. I plotted the users activity onto the artwork in the area.

1.2.1 User 3: University of Michigan

(a) UofM Logo

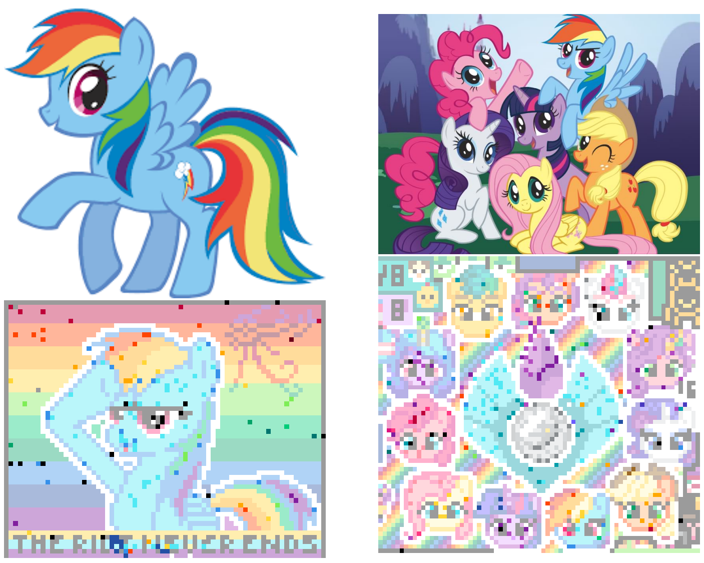


(b) User 3 Pixels

Figure 2: User 3 Pixels In UofM Logo

User 3's pixels are almost all placed in the region of the UofM logo. Additionally user 3's active period, slightly maps to the the day and night cycle when in the EDT time zone, which the state of Michigan uses.

1.2.2 User 1 And 2: My Little Pony



(a) Rainbow Dash (b) Main Ponies

Figure 3: User 1 And 2 Pixels In MLP Art

Both User 1 and User 2 had overlapped areas of focus. From this, I conclude that both users are members of the MLP community. This also explains the wider distribution of both users as the MLP community moved their artwork multiple times and had more art that divded their attention.

2 Comparison Duckdb Vs. PySpark

2.1 Memory

The parquet file containing the data in parquet form was 1.84 Gigs.Looking at the Duckdb version, loading the parquet file into memory took anywhere between 8-10 Gigs of memory, while the PySpark version only took about 2 Gigs of memory.

2.2 CPU

The reverse is true in regards to CPU usage. Running the same queries in Duckdb resulted almost negligible CPU usage compared to idle. On the other hand, PySpark had a much higher CPU demand usage demanded upwards of 50 percent

2.3 Time

Similarly to the CPU section, Duckdb performed almost all queries sub 1 second, but PySpark took at minimum 1 second.