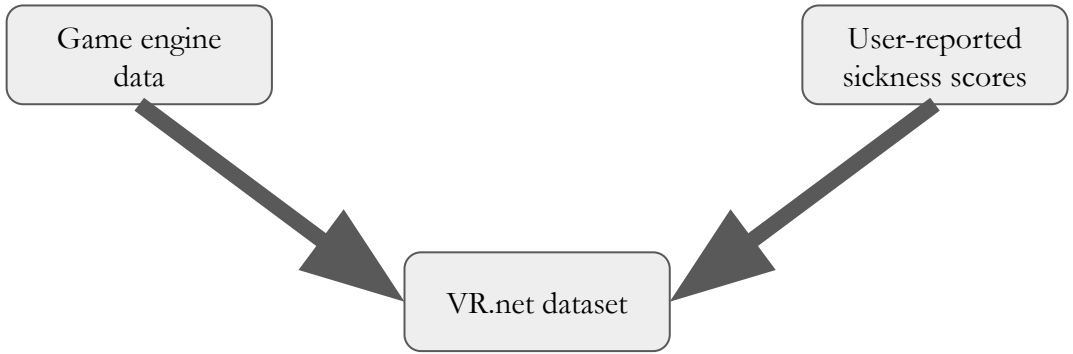
Decoding VR Motion Sickness With The VR.net Dataset

Vinayak Gajjewar, Swathi Bhat

Motivation

- Up to **40%** of users experience VR motion sickness [1]
  - Fatigue
  - Disorientation
  - Nausea
- Continue analysis on games that show promise
- Can we use ML to predict onset of VR sickness?



[1] Wen, Elliott, et al. "VR. net: A Real-world Dataset for Virtual Reality Motion Sickness Research." IEEE Transactions on Visualization and Computer Graphics (2024).

Analysis of games that demonstrated potential

- Beat Saber
- Epic Roller Coasters
- Traffic Cop

Mean sickness score across games

Beat Saber	**Epic Roller Coasters**	Voxel Shot VR	Cartoon Network Journeys VR	VR Monster Awakens
1.79	2.75	1.29	1.57	2.21

Sickness score correlations

Strong negative correlation between sickness score and:

rot6 (*Beat Saber:* ***-0.41****, Epic Roller Coasters:* ***-0.69***)

pos1 (*Beat Saber:* ***-0.52****, Epic Roller Coasters,* ***-0.69***)

Camera movement

Epic Roller Coasters

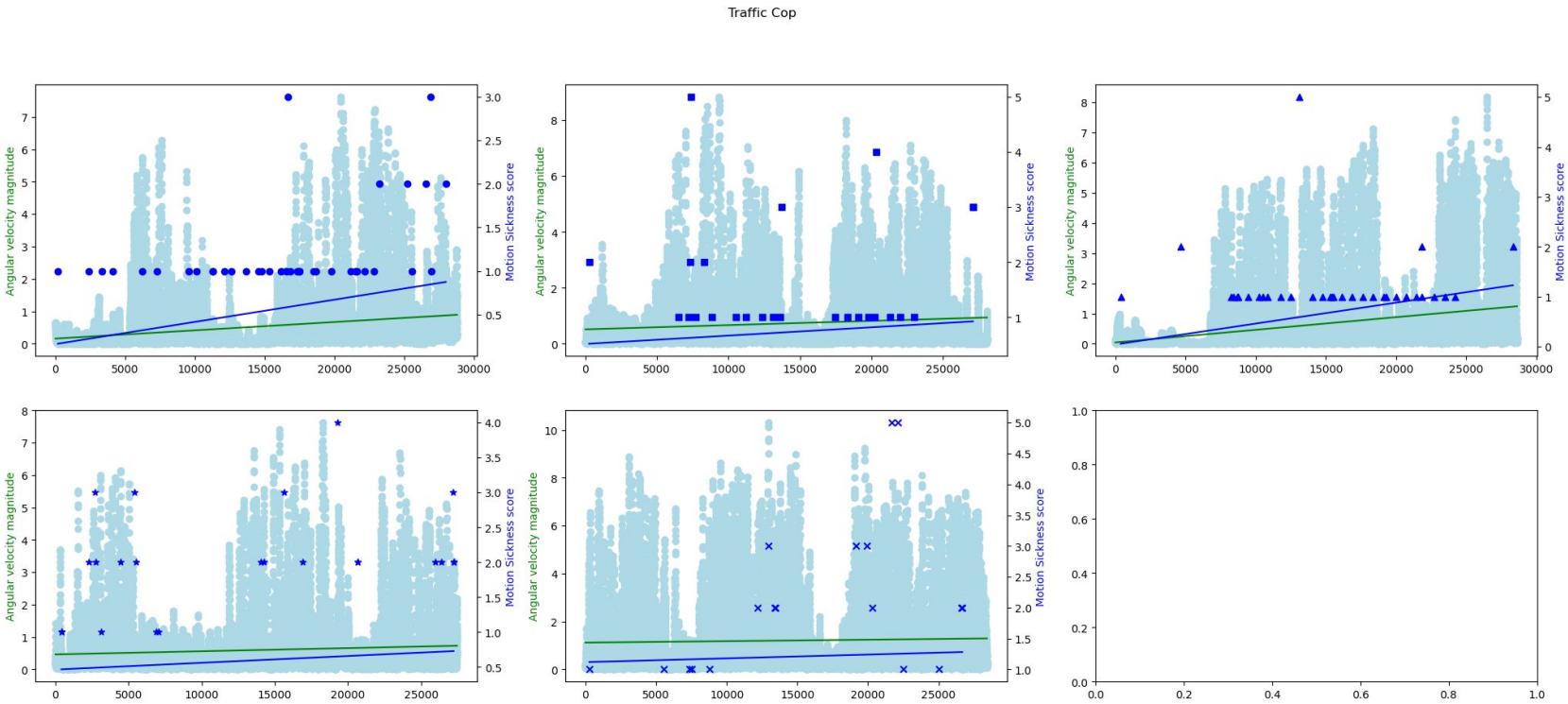
- Motion sickness occurs with:
  - **Sideways tilt** or **horizontal rotations**
  - **Upward/downward tilt** + **horizontal rotation**
- Users experience less motion sickness when we **minimize forward tilt**

Beat Saber

- Significant **pitch** (up/down tilt) could cause motion sickness
- When the camera is **level** (in terms of pitch/yaw/roll), users experience less motion sickness

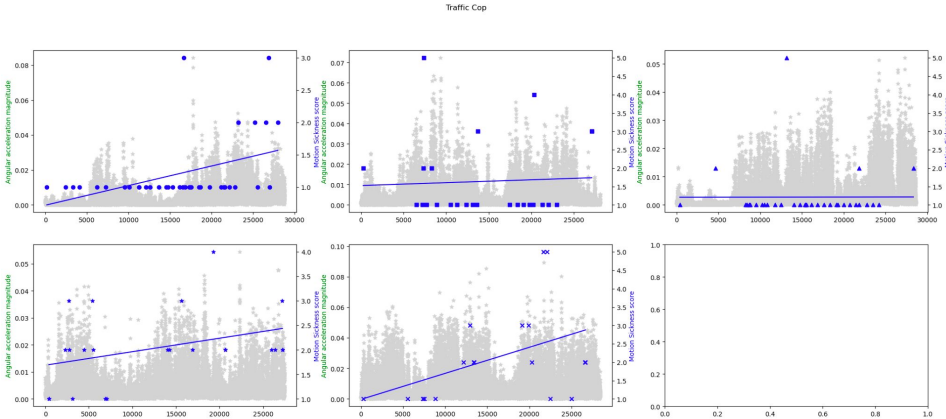
Traffic Cop game in depth analysis

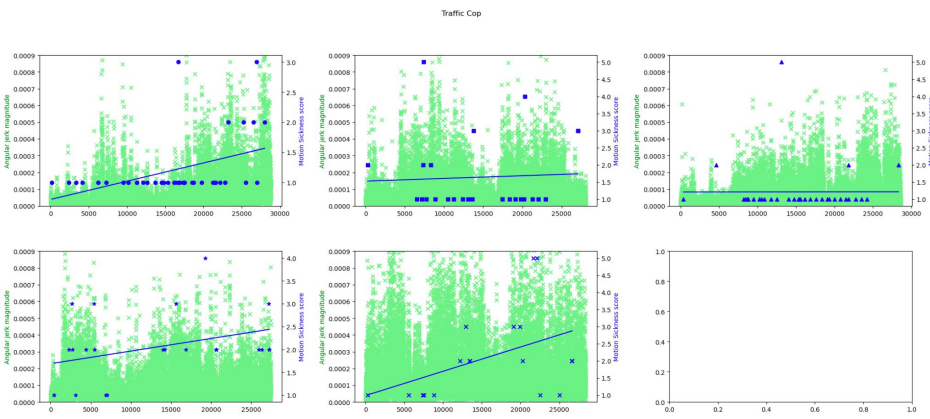
Angular velocity and motion sickness scores



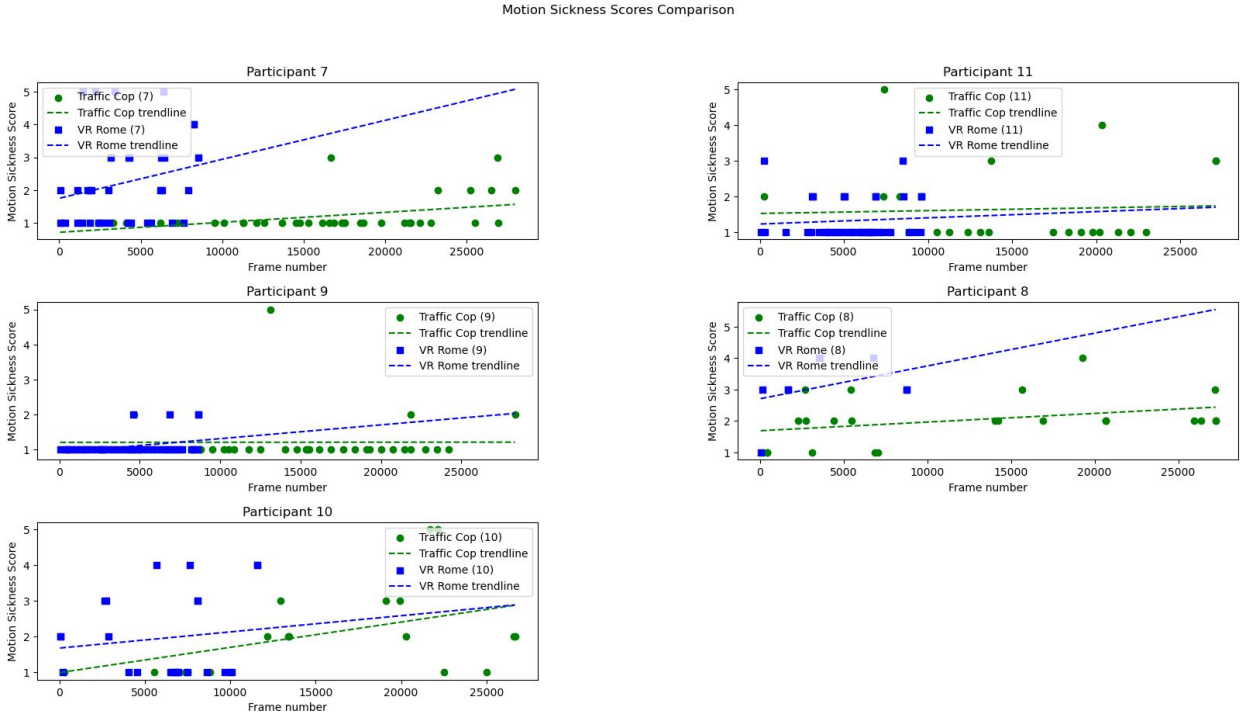
Traffic Cop game in depth analysis

Angular acceleration and angular jerk against motion sickness scores





Comparing motion sickness for same set of participants for Traffic Cop and VR Rome



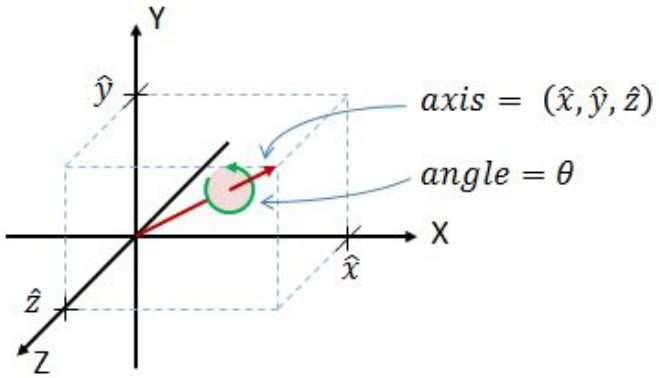
Key takeaways from statistical analysis

- Most games don't cause enough motion sickness
  - Sparse dataset
- But not enough data points for good **time series analysis**!
- Not enough storage/compute for **scene object analysis**

What about ML?

Rotation quaternions

- Alternative to rotation matrices to represent rotations in 3D space
- Closely related to *axis-angle* representation



https://danceswithcode.net/engineeringnotes/quaternions/quaternions.html

Rotation matrix to quaternion

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$|q_0| = \sqrt{\frac{1 + r_{11} + r_{22} + r_{33}}{4}}$$

$$|q_1| = \sqrt{\frac{1 + r_{11} - r_{22} - r_{33}}{4}}$$

$$|q_2| = \sqrt{\frac{1 - r_{11} + r_{22} - r_{33}}{4}}$$

$$|q_3| = \sqrt{\frac{1 - r_{11} - r_{22} + r_{33}}{4}}$$

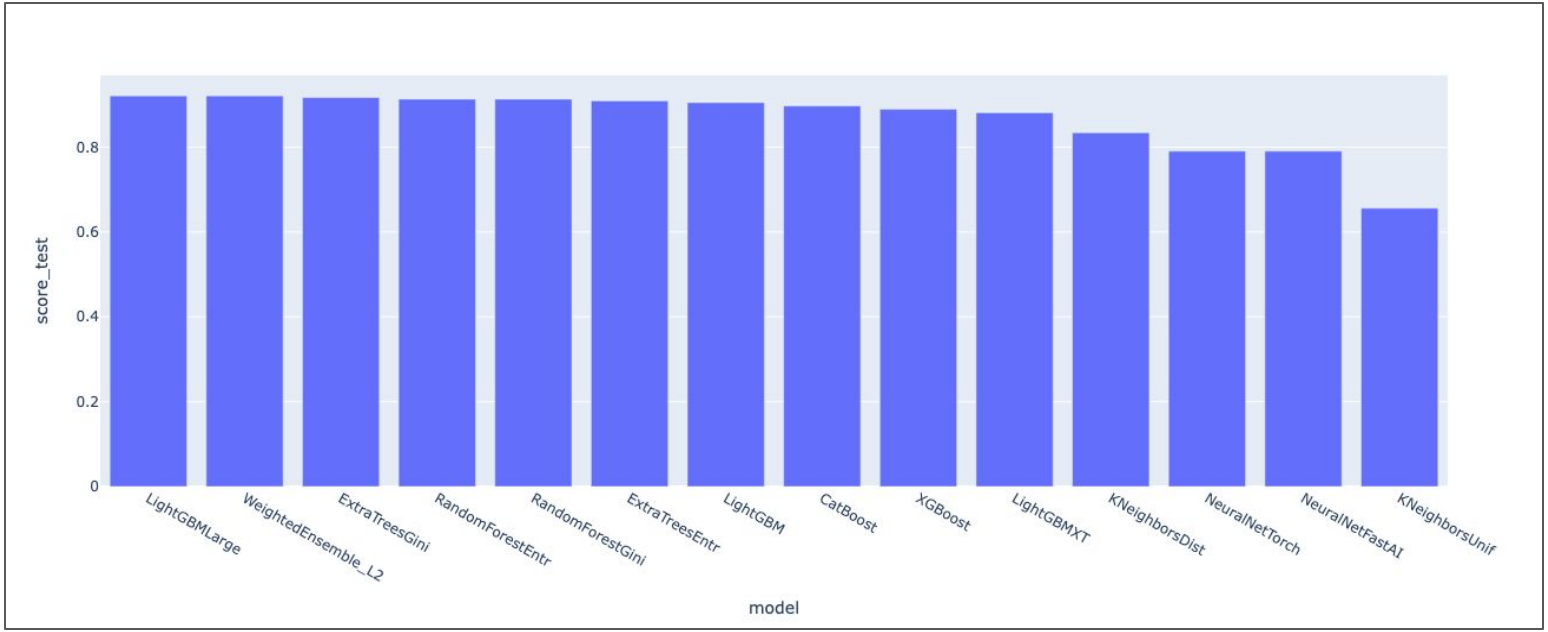
If q_0 is largest:	If q_1 is largest:	If q_2 is largest:	If q_3 is largest:
$q_1 = \frac{r_{32} - r_{23}}{4q_0}$	$q_0 = \frac{r_{32} - r_{23}}{4q_1}$	$q_0 = \frac{r_{13} - r_{31}}{4q_2}$	$q_0 = \frac{r_{21} - r_{12}}{4q_3}$
$q_2 = \frac{r_{13} - r_{31}}{4q_0}$	$q_2 = \frac{r_{12} + r_{21}}{4q_1}$	$q_1 = \frac{r_{12} + r_{21}}{4q_2}$	$q_1 = \frac{r_{13} + r_{31}}{4q_3}$
$q_3 = \frac{r_{21} - r_{12}}{4q_0}$	$q_3 = \frac{r_{13} + r_{31}}{4q_1}$	$q_3 = \frac{r_{23} + r_{32}}{4q_2}$	$q_2 = \frac{r_{23} + r_{32}}{4q_3}$

Source: https://danceswithcode.net/engineeringnotes/quaternions/quaternions.html

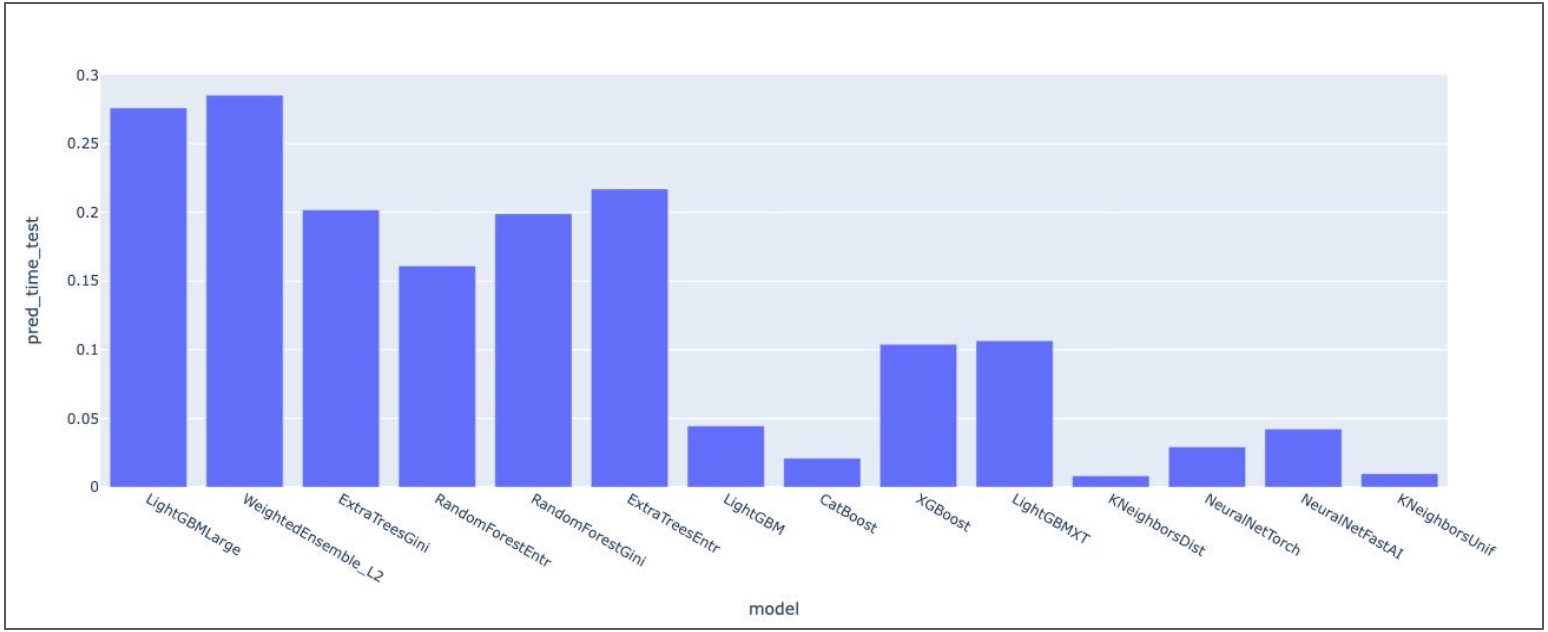
ML model features

Feature name	Description
rotquat[4]	Quaternion representation of in-game camera rotation matrix
pos[3]	View matrix position vector
forward[3]	View matrix forward vector
up[3]	View matrix up vector
rot[9]	In-game camera rotation matrix

Per-model accuracy



Per-model prediction time

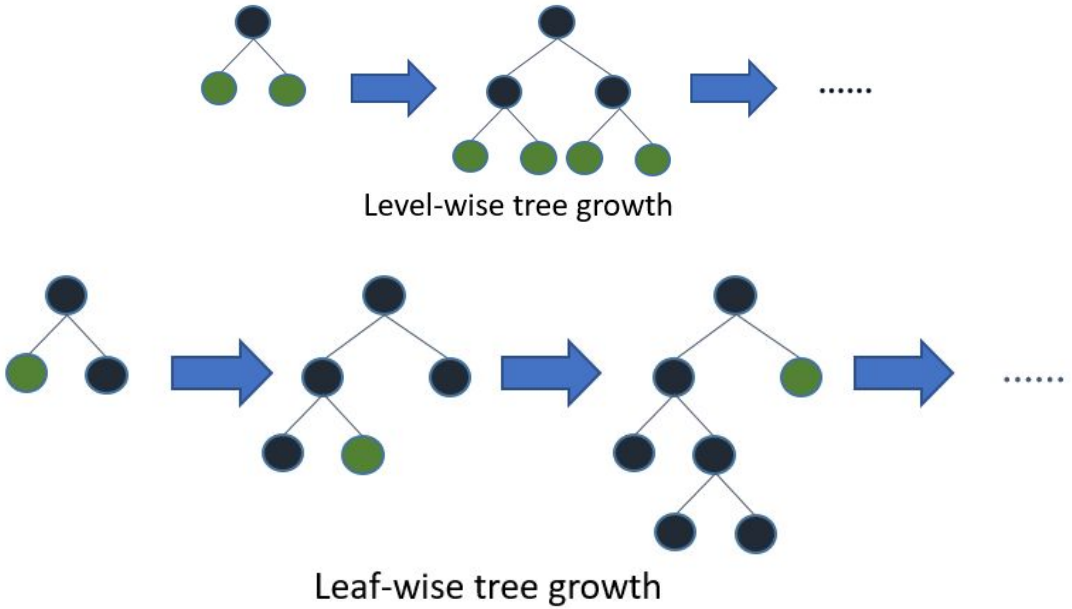


LightGBM

- Decision tree boosting algorithm
- Discretize continuous features to speed up training
- Leaf-wise tree growth

Source: https://lightgbm.readthedocs.io/en/latest/Features.html

LightGBM



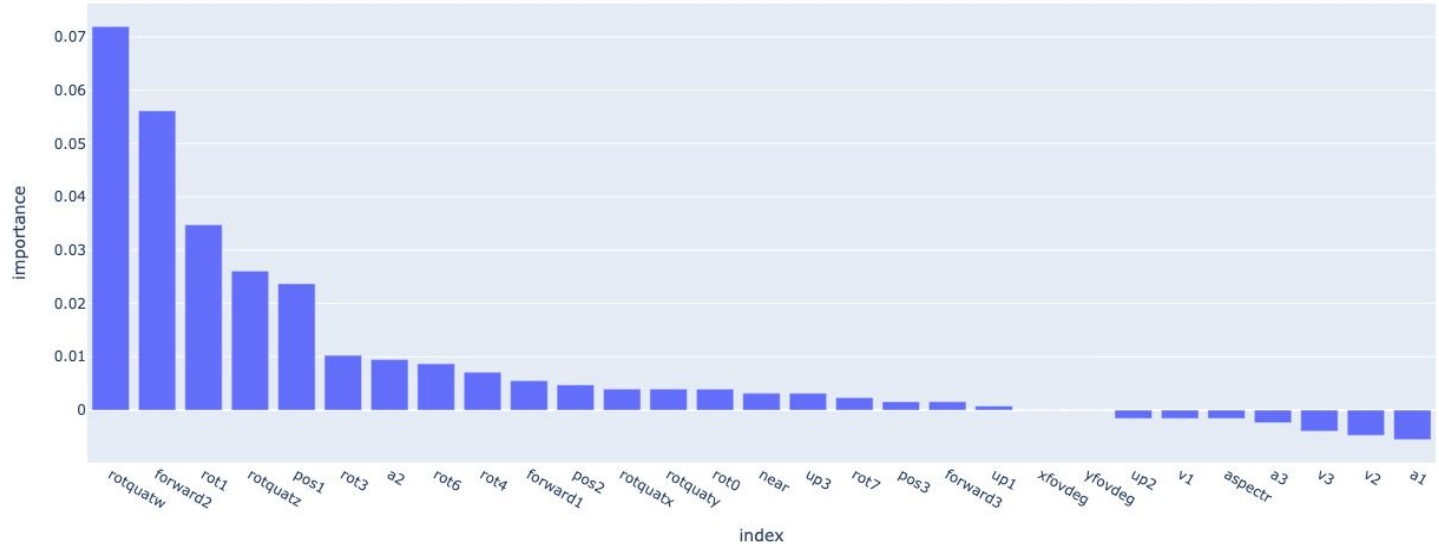
Source: https://lightgbm.readthedocs.io/en/latest/Features.html

Feature importance

After replacing a feature with random values, how much does my model's performance drop?

- More important:
  - In-game camera rotation matrix
- Less important:
  - FOV information
    - Same across games…

Feature importance



Future work

- Time series analysis
- ML generalizability + interpretability
- In-depth scene object analysis

Thanks for listening! Questions?