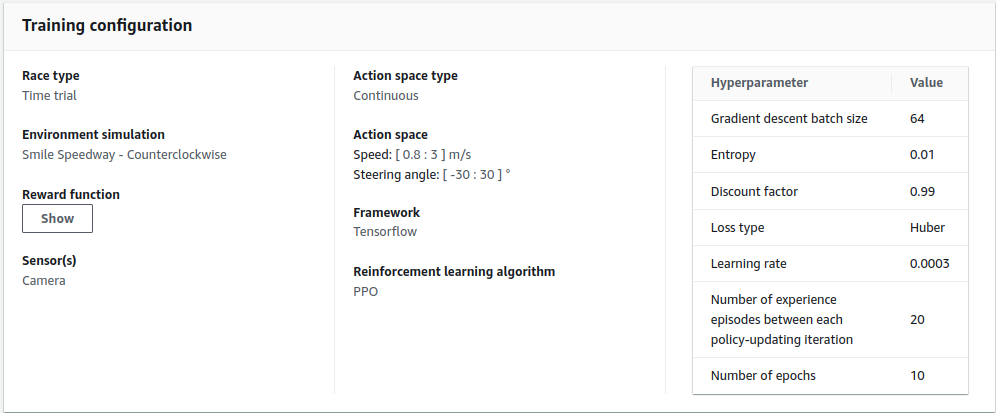
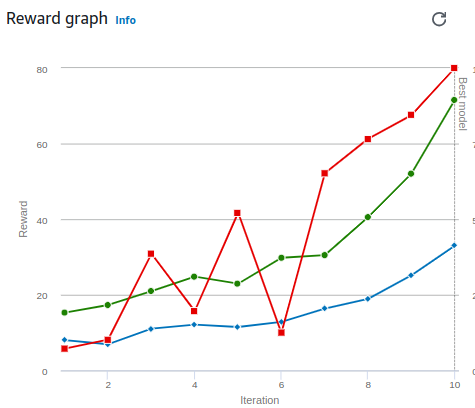
# Modelo 1

## Función de recompensa:

| def reward\_function(params):  '''  Example of rewarding the agent to follow center line  '''    # Read input parameters  track\_width = params['track\_width']  distance\_from\_center = params['distance\_from\_center']    # Calculate 3 markers that are at varying distances away from the center line  marker\_1 = 0.1 \* track\_width  marker\_2 = 0.25 \* track\_width  marker\_3 = 0.5 \* track\_width    # Give higher reward if the car is closer to center line and vice versa  if distance\_from\_center <= marker\_1:  reward = 1.0  elif distance\_from\_center <= marker\_2:  reward = 0.5  elif distance\_from\_center <= marker\_3:  reward = 0.1  else:  reward = 1e-3 # likely crashed/ close to off track    return float(reward) |
| --- |

## Datos entrenamiento





## 

## Evaluación

| 1º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | SmileRaceway | No | 0 | 00:16.201 |
|  |  |  | 1 | 00:16.770 |
|  |  |  | 0 | 00:16.267 |
|  | | | | |

| 2º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | ForeverRaceway | No | 1 | 00:18.803 |
|  |  |  | 2 | 00:20.193 |
|  |  |  | 7 | 00:33.350 |
|  | | | | |

# Modelo 2

## Función de recompensa:

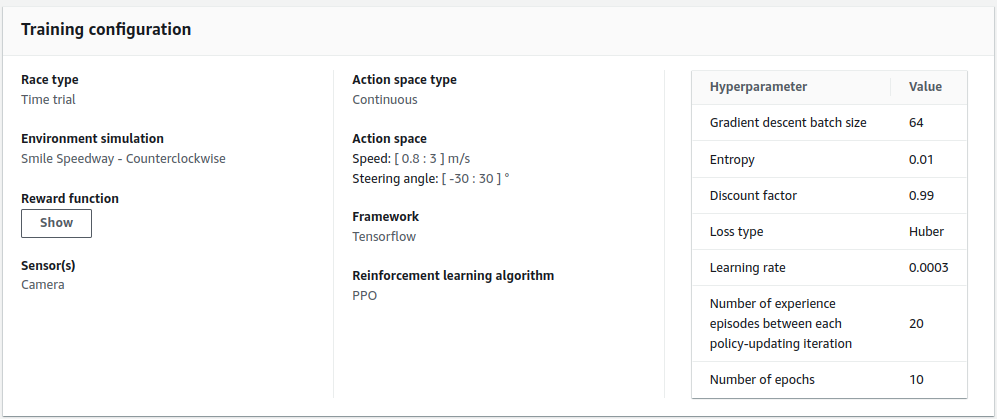
| import math def reward\_function(params):    # Read input parameters  all\_wheels\_on\_track = params['all\_wheels\_on\_track']  distance\_from\_center = params['distance\_from\_center']  track\_width = params['track\_width']  abs\_steering = abs(params['steering\_angle']) # Only need the absolute steering angle  speed = params['speed']  waypoints = params['waypoints']  closest\_waypoints = params['closest\_waypoints']  heading = params['heading']  progress = params['progress']    # Calculate 3 markers that are at varying distances away from the center line  marker\_1 = 0.1 \* track\_width  marker\_2 = 0.25 \* track\_width  marker\_3 = 0.5 \* track\_width    # Give higher reward if the car is closer to center line and vice versa  if distance\_from\_center <= marker\_1:  reward = 1.0  elif distance\_from\_center <= marker\_2:  reward = 0.5  elif distance\_from\_center <= marker\_3:  reward = 0.1  else:  reward = 1e-3 # likely crashed/ close to off track    #Salirse de pista e ir rapido  # Set the speed threshold based your action space  SPEED\_THRESHOLD = 2.5   if not all\_wheels\_on\_track:  # Penaliza salirse  reward = 1e-3  elif speed < SPEED\_THRESHOLD:  # Penaliza ir lento  reward = 0.5  else:  # recompensa ir rapido  reward = 1.0    #Evitar zig-zag  ABS\_STEERING\_THRESHOLD = 20    # Penaliza girar mucho  if abs\_steering > ABS\_STEERING\_THRESHOLD:  reward \*= 0.8    #Que el coche siempre vaya en direccion correcta  # Calcular la dirección de la pista entre los puntos más cercanos  next\_point = waypoints[closest\_waypoints[1]]  prev\_point = waypoints[closest\_waypoints[0]]   # Calculate the direction in radius, arctan2(dy, dx), the result is (-pi, pi) in radians  track\_direction = math.atan2(next\_point[1] - prev\_point[1], next\_point[0] - prev\_point[0])  # Convert to degree  track\_direction = math.degrees(track\_direction)   # Calculate the difference between the track direction and the heading direction of the car  direction\_diff = abs(track\_direction - heading)  if direction\_diff > 180:  direction\_diff = 360 - direction\_diff   # Penaliza la gran diferencia  DIRECTION\_THRESHOLD = 10.0  if direction\_diff > DIRECTION\_THRESHOLD:  reward \*= 0.5    #Recompensar el casi terminar  if progress > 95:  reward += 5.0 # Recompensa adicional por casi completar la vuelta   return float(reward) |
| --- |

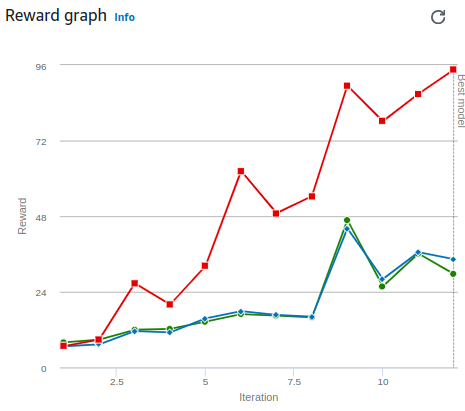
Hemos dejado el código por defecto, que nos indica que el modelo debe ir siempre lo más centrado posible y hemos añadido lo siguiente:

1. Le hemos añadido una penalización por sacar las 4 ruedas de la pista y también le hemos puesto que siempre intente ir a una velocidad de 2.5.
2. Hemos añadido que el modelo trate de evitar el zig-zag, mirando el ángulo de giro del volante.
3. Miramos en el siguiente bloque de código que el coche no se haya girado y vaya para atrás, mirando los parámetros de heading y waypoints.
4. Hemos puesto una gran recompensa al modelo cuando completa el 95% de la vuelta, porque esto significa que ha conseguido casi terminar la vuelta sin salirse.

\*Viendo el vídeo del tiempo de la carrera, podemos ver que el coche para la pista en la que se prueba frena mucho cuando llega a una curva.

## Datos de entrenamiento





## Evaluación

| 1º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | SmileRaceway | No | 0 | 00:11.381 |
|  |  |  | 0 | 00:11.469 |
|  |  |  | 1 | 00:14.928 |
|  | | | | |

| 2º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | ForeverRaceway | No | 1 | 00:15.597 |
|  |  |  | 1 | 00:16.279 |
|  |  |  | 4 | 00:23.403 |
|  | | | | |

# Modelo 3 (clon del 1)

## Función de recompensa:

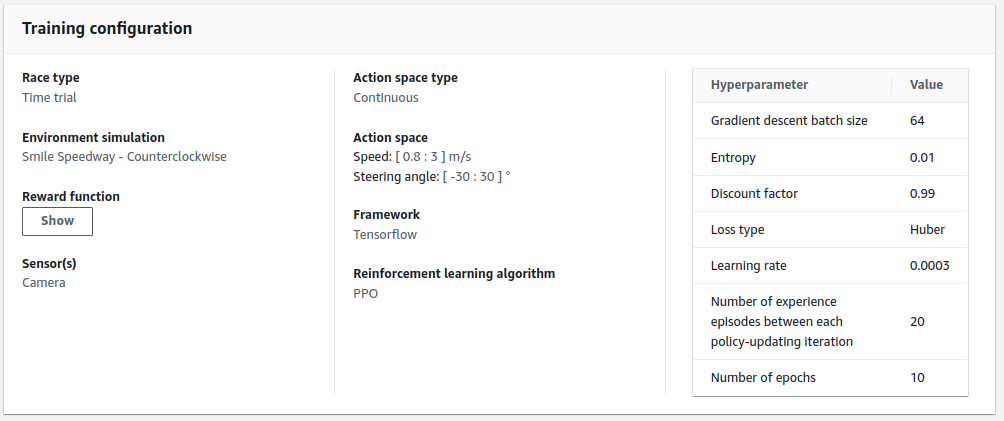
| import math def reward\_function(params):    # Read input parameters  track\_width = params['track\_width']  distance\_from\_center = params['distance\_from\_center']  progress = params['progress']  closest\_waypoints = params['closest\_waypoints']  waypoints = params['waypoints']  heading = params['heading']  speed = params['speed']    # Calculate 3 markers that are at varying distances away from the center line  marker\_1 = 0.1 \* track\_width  marker\_2 = 0.25 \* track\_width  marker\_3 = 0.5 \* track\_width    # Give higher reward if the car is closer to center line and vice versa  if distance\_from\_center <= marker\_1:  reward = 1.0  elif distance\_from\_center <= marker\_2:  reward = 0.5  elif distance\_from\_center <= marker\_3:  reward = 0.1  else:  reward = 1e-3 # likely crashed/ close to off track    next\_point = waypoints[closest\_waypoints[1]]  prev\_point = waypoints[closest\_waypoints[0]]  # Calcular la dirección de la pista entre los puntos más cercanos  track\_direction = math.degrees(math.atan2(next\_point[1] - prev\_point[1], next\_point[0] - prev\_point[0]))    direction\_diff = abs(track\_direction - heading)    SPEED\_THRESHOLD = 2.5 # Objetivo de velocidad en rectas  in\_curve = direction\_diff > 10 # Detecta si está en una curva  if in\_curve:  if speed < SPEED\_THRESHOLD:  reward += 3.0 # Recompensa por reducir velocidad en curva  else:  reward \*= 0.5 # Penalización por ir demasiado rápido en curva  else:  if speed >= SPEED\_THRESHOLD:  reward += 5.0 # Recompensa por mantener alta velocidad en recta   # Progreso en la pista  reward += progress \* 0.1 # Recompensa basada en progreso total     return float(reward) |
| --- |

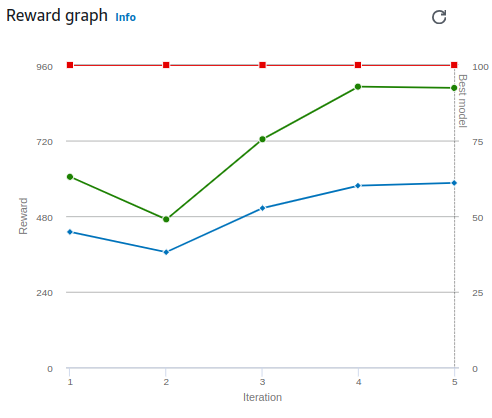
Hemos añadido los 3 parámetros que pedía en la función de recompensa.

Hemos utilizado los waypoints para saber si lo que se viene es una curva o no y en base a ello que el coche frene o no usando speed.

Por último, hemos añadido un reward mayor a cada porcentaje de pista que el modelo complete.

## Datos de entrenamiento





## Evaluación

| 1º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | SmileRaceway | No | 0 | 00:14.923 |
|  |  |  | 0 | 00:14.543 |
|  |  |  | 1 | 00:16.351 |
|  | | | | |

| 2º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | ForeverRaceway | No | 11 | 00:42.888 |
|  |  |  | 6 | 00:30.259 |
|  |  |  | 10 | 00:40.674 |
|  | | | | |

# Modelo 4 (clon del 2)

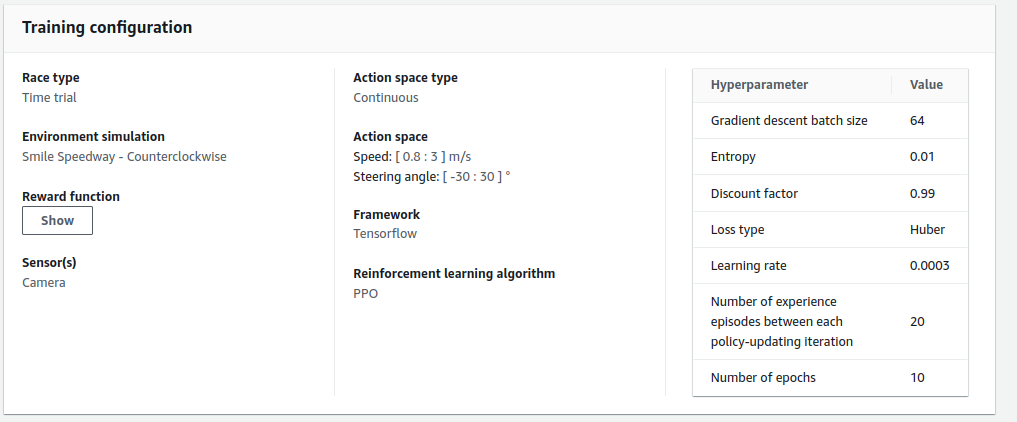
## Función de recompensa

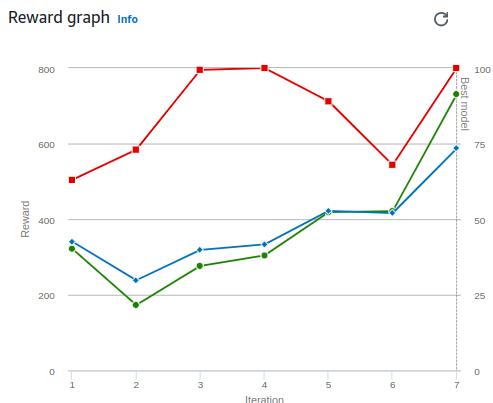
| import math  def reward\_function(params):    # Read input parameters  all\_wheels\_on\_track = params['all\_wheels\_on\_track']  distance\_from\_center = params['distance\_from\_center']  track\_width = params['track\_width']  abs\_steering = abs(params['steering\_angle']) # Only need the absolute steering angle  speed = params['speed']  waypoints = params['waypoints']  closest\_waypoints = params['closest\_waypoints']  heading = params['heading']  progress = params['progress']    #Evitar salirse de pista e ir rapido  # Set the speed threshold based your action space  SPEED\_THRESHOLD = 2.5   if not all\_wheels\_on\_track:  # Penalize if the car goes off track  reward = 1e-3  elif speed < SPEED\_THRESHOLD:  # Penalize if the car goes too slow  reward = 0.5  else:  # High reward if the car stays on track and goes fast  reward = 1.0    #Evitar zig-zag  # Steering penality threshold, change the number based on your action space setting  ABS\_STEERING\_THRESHOLD = 20    # Penalize reward if the car is steering too much  if abs\_steering > ABS\_STEERING\_THRESHOLD:  reward \*= 0.8    #Que el coche siempre vaya en direccion correcta  # Calcula la dirección del centro basado en el waypoint más cercano  next\_point = waypoints[closest\_waypoints[1]]  prev\_point = waypoints[closest\_waypoints[0]]   # Calculate the direction in radius, arctan2(dy, dx), the result is (-pi, pi) in radians  track\_direction = math.atan2(next\_point[1] - prev\_point[1], next\_point[0] - prev\_point[0])  # Convert to degree  track\_direction = math.degrees(track\_direction)   # Calcula la diferencia entre la pista y hacia donde va el coche  direction\_diff = abs(track\_direction - heading)  if direction\_diff > 180:  direction\_diff = 360 - direction\_diff   # Penaliza la diferencia grande  DIRECTION\_THRESHOLD = 10.0  if direction\_diff > DIRECTION\_THRESHOLD:  reward \*= 0.5    # Progreso en la pista  reward += progress \* 0.1 # Recompensa basada en progreso total   #Recompensar el casi terminar  if progress > 95:  reward += 5.0 # Recompensa adicional por casi completar la vuelta   return float(reward) |
| --- |

Eliminamos el ir por el centro de la pista ya que queremos que el coche recorte lo máximo posible en la curvas, además ya le teníamos añadido el que no sacase la 4 ruedas fuera, lo cual era un poco redundante.

Añadimos la recompensa por porcentaje de progresión para así asegurarnos de que el mejor modelo sea el que más lejos ha llegado entrenando.

## Datos de entrenamiento





## 

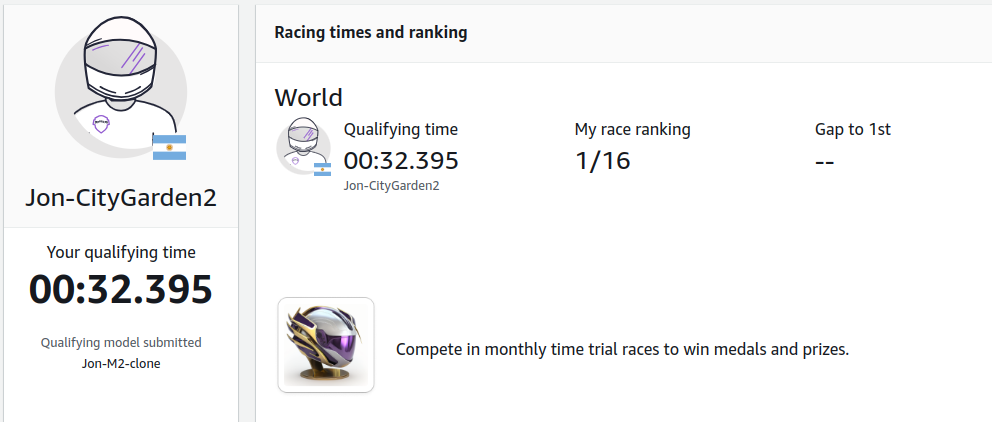
## 

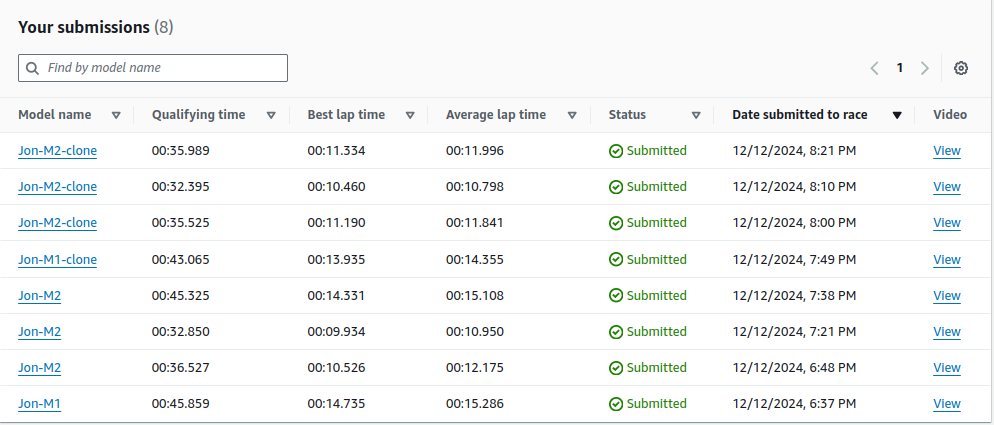
## Evaluación

| 1º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | SmileRaceway | No | 0 | 00:11.068 |
|  |  |  | 0 | 00:12.010 |
|  |  |  | 0 | 00:10.935 |
|  | | | | |

| 2º evaluación | | | | |
| --- | --- | --- | --- | --- |
| **Tipo** | **Pista evaluación** | **Clockwise** | **Salidas de pista** | **Tiempo** |
| Contrarreloj | ForeverRaceway | No | 3 | 00:20.733 |
|  |  |  | 2 | 00:17.994 |
|  |  |  | 2 | 00:16.546 |
|  | | | | |

# MIAR Race





# 

# 

# Conclusiones

Si miramos las gráficas de entrenamiento del modelo 1 y del 3 podemos ver un caso curioso, el primer modelo casi termina consiguiendo una vuelta entera pero su versión clonada desde el primer momento de entrenamiento consigue el 100% de la vuelta y la gráfica es una línea horizontal.

Evaluación 1

| **Modelo 1** | **Modelo 3** |
| --- | --- |
| **Tiempo** | **Tiempo** |
| 00:16.201 | 00:14.923 |
| 00:16.770 | 00:14.543 |
| 00:16.267 | 00:16.351 |

Podemos comprobar que el modelo con más entrenamiento y alguna mejora en la función de recompensa mejora en la misma pista donde entreno.

Evaluación 2

| **Modelo 1** | **Modelo 3** |
| --- | --- |
| **Tiempo** | **Tiempo** |
| 00:18.803 | 00:42.888 |
| 00:20.193 | 00:30.259 |
| 00:33.350 | 00:40.674 |

Viendo los tiempos en otro circuito, podemos ver como el modelo clonado está sobreentrenado y a la hora de ir a otros circuitos hace muy mal tiempos porque se sale mucho, mientras que el primero también tiene algún caso en el que se sale pero generaliza mejor.

Evaluación 1

| **Modelo 2** | **Modelo 4** |
| --- | --- |
| **Tiempo** | **Tiempo** |
| 00:11.381 | 00:11.068 |
| 00:11.469 | 00:12.010 |
| 00:14.928 | 00:10.935 |

Si comparamos los tiempos de ambos en la pista donde han entrenado podemos ver que el modelo clonado es algo más rápido y además no tiene ninguna salida de pista. Es decir, los cambios han podido surgir efecto o el seguir entrenando el modelo o el conjunto de ambos.

Evaluación 2

| **Modelo 2** | **Modelo 4** |
| --- | --- |
| **Tiempo** | **Tiempo** |
| 00:15.597 | 00:20.733 |
| 00:16.279 | 00:17.994 |
| 00:23.403 | 00:16.546 |

Cuando cambiamos de pista podemos ver que el modelo más entrenado en la anterior pista consigue peores resultados, ya que está sobreentrenado en ella y a la hora de generalizar se sale mas de pista.

Tras analizar el video de el envío a la carrera del modelo 4 podemos ver que aunque le hayas quitado el que vaya por el medio, como el modelo previamente había entrenado con ello siempre trata de hacerlo. Aunque en los momentos que no lo hace si consigue mejorar el tiempo, sobre todo cuando intenta hacer dos curvas enlazadas rectas si sacar las 4 ruedas y no por la línea central.

Si comparamos todos los modelos, podemos sacar de conclusión que el modelo 2 es el que mejor generaliza mientras que el modelo 4 es el más rápido en la carrera.