\*All input variables of the classification model are included in the previous Data-Feature table. There’s no new variables.

\*Several features in the Data-Feature table is not necessary now, for details please check the latest Data-Feature table in the end of this file.

**Input description of the classification model**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input variables** | | | **Index** | **Shape/type** | **Transfer to X\_features** | **Numbers of labels** | **Feature importance** |
| 1st gain order sequence | | | 0 | [A list of 18 int] | float (list->a clustered label) | 3 (can be 0,1, or 2) | 0.041 |
| 1st gain time sequence | | | 1 | [A list of 18 int] | float (list->a clustered label) | 3 (can be 0,1, or 2) | **0.049** |
| Accumulated amount of each item | | | 2 | [A list of 18 int] | float (list->a clustered label) | 3 (can be 0,1, or 2) | 0.018 |
| ~~Sparse reward accumulation sequence~~ | | | 3 | [A list of int]  ~~length= duration\_steps~~ | float (list->a clustered label) | 3 (can be 0,1, or 2) | **0.050** |
| ~~Dense reward accumulation sequence~~ | | | 4 | [A list of int]  ~~length= duration\_steps~~ | float (list->a clustered label) | 3 (can be 0,1, or 2) | 0.026 |
| If useless tools was crafted (3 booleans) | | iron\_axe | 5 | boolean | float (1 if True else 0) |  | 0 |
| stone\_axe | 6 | boolean | float (1 if True else 0) |  | 0.003 |
| wooden\_axe | 7 | boolean | float (1 if True else 0) |  | 0.015 |
| Sparse total reward | | | 8 | int | float |  | **0.055** |
| Dense total reward | | | 9 | int | float |  | **0.120** |
| Attack efficiency | | | 10 | float | same |  | **0.097** |
| Attack ratio | | | 11 | float | same |  | **0.102** |
| Equipped attack ratio | | | 12 | float | same |  | **0.100** |
| Camera moving ratio | | | 13 | float | same |  | **0.111** |
| Position moving ratio | | | 14 | float | same |  | **0.100** |
| Placed\_items (4 items) | torch\_placed | | 15 | int | float |  | 0.038 |
| cobblestone\_placed | | 16 | int | float |  | 0.028 |
| dirt\_placed | | 17 | int | float |  | 0.021 |
| stone\_placed | | 18 | int | float |  | 0.019 |
| If\_smelt\_coal | | | 19 | boolean | float (1 if True else 0) |  | 0.008 |

\*The order of 18 items in inventory list and how to calculate reward :

|  |  |  |  |
| --- | --- | --- | --- |
| **Order index (from 0)** | **Item** | **Can get reward or not?**  **(‘x’ for not)** | **How many reward?**  **Sparse** reward: The reward will be accumulated whenever one item has been obtained at the first time;  **Dense** reward: The reward will be accumulated every time when one item has been obtained, even if there are already same items. |
| 0 | 'coal' | x |  |
| 1 | 'cobblestone' |  | 16 |
| 2 | 'crafting\_table' |  | 4 |
| 3 | 'dirt' | x |  |
| 4 | 'furnace' |  | 32 |
| 5 | 'iron\_axe' | x |  |
| 6 | 'iron\_ingot' |  | 128 |
| 7 | 'iron\_ore' |  | 64 |
| 8 | 'iron\_pickaxe' |  | 256 |
| 9 | 'log' |  | 1 |
| 10 | 'planks' |  | 2 |
| 11 | 'stick' |  | 4 |
| 12 | 'stone' | x |  |
| 13 | 'stone\_axe' | x |  |
| 14 | 'stone\_pickaxe' |  | 32 |
| 15 | 'torch' | x |  |
| 16 | 'wooden\_axe' | x |  |
| 17 | 'wooden\_pickaxe' |  | 8 |

**Output description of the classification model:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function name** | **Operation number** | **Description** | **Shape/type of return value** | **Other things** |
| print\_decision\_tree | 1 | Print structure of decision tree | / |  |
| check\_input\_tuple | 2 | Check if there are errors in input values and print | / |  |
| **predict** | 3 | Prediction of the evaluated label (0 for under average ,1 for around average, 2 for over average) | A list of labels | Do the prediction of one input |

**Data-Feature table**

|  |  |  |
| --- | --- | --- |
| **Original Data** | **Features** | **Possible or not** |
| **'duration\_steps'**  \*All the other actions, inventory sequences, reward sequences and other features are based on the state of each “step”. | Total time steps.  1 time step is 0.05 s (50 ms) |  |
| **'inventory\_seq'** -> **Inventory information** | **Inventory keeping sequence**  ~~The amount of each item in the inventory in each step. This~~ **~~Inventory keeping sequence~~** ~~is basic and very important, and has the highest priority.~~  ~~Type: a list of Ordered Dictionary~~  ~~[ inventory1, inventory2, …]~~  ~~len(list)= duration\_steps~~  ~~Here is an example of inventory:~~  ~~\*~~ **~~'inventory'~~** ~~is an ordered dictionary looks like below (include 18 items, please keep this specific order, value of each key is the current amount of each item):~~  ~~('coal', 0),~~  ~~('cobblestone', 0),~~  ~~('crafting\_table', 0),~~  ~~('dirt', 0),~~  ~~('furnace', 0),~~  ~~('iron\_axe', 0),~~  ~~('iron\_ingot', 0),~~  ~~('iron\_ore', 0),~~  ~~('iron\_pickaxe', 0),~~  ~~('log', 7),~~  ~~('planks', 0),~~  ~~('stick', 0),~~  ~~('stone', 0),~~  ~~('stone\_axe', 0),~~  ~~('stone\_pickaxe', 0),~~  ~~('torch', 0),~~  ~~('wooden\_axe', 0),~~  ~~('wooden\_pickaxe', 0)])),~~  ~~<- So, this inventory means at this time step, the player has 7 logs, other items are all 0.~~ |  |
| Gain **order** sequence of the 1st of each item  Type: list of int  [1, 8, 0 , …] -> 18 items in same order as above  item=0 if it wasn’t gained in the episode;  item=i(1<=i<=18) means it was the i\_th gained item |  |
| Gain **time\_step** sequence of the 1st of each item  Type: list of int  [100, 805, 0 , …] -> 18 items in same order as above  item=0 if it wasn’t gained in the episode;  item=t(1<=i<= duration\_steps) means it was gained at the t\_th time\_step |  |
| ~~Gain~~ **~~time\_step~~** ~~sequence of~~ **~~all items~~**  ~~Type: 2-dimensional array of int~~  ~~[100, 805, 0 , …] -> 18 items in same order as above (columns)~~  ~~[200, 1000, 0,…]~~  ~~[………………………]~~  ~~->1562 rows (1562 is upper limit of the inventory. But the real data usually will not achieve the upper limit)~~  ~~item=0 if it wasn’t gained in the episode;~~  ~~item=t(1<=i<= duration\_steps) means it was gained at the t\_th time\_step;~~  ~~the r\_th row shows when the r\_th same item was gained~~ |  |
| If useless tools was crafted  Type: a list of 3 booleans  [boolean1, boolean2, boolean3]  (Assume the following 3 tools are useless)  boolean1: Whether the player crafted the **iron\_axe**? True = yes, craft  boolean2: Whether the player crafted the **stone\_axe**? True = yes, craft  boolean3: Whether the player crafted the **wooden\_axe**? True = yes, craft |  |
| Accumulated amount of each item  List of int: [10, 5, 8,…] -> 18 items in same order as above  i.e. how many amount of each item the player got in total, include the items that was used.  \*the amount of each inventory item might decrease during the game, so the state of the last time step is NOT what we want. |  |
| **Sparse\_reward information** | **Sparse reward accumulation sequence**  i.e. how many rewards the player get up till each step. The reward will be accumulated whenever one item has been obtained **at the first time** (e.g. when getting the first log, reward+=1). If the rewards don’t change in step t then reward(t)= reward(t-1)  list of float (or int): [0.0,…, 1.0, 1.0, 2.0, 3.0,…]  len(list)= duration\_steps  \*Please check the following list for ‘how much reward to give the player for each item’. Only items shown in this list will get a reward.  <Item reward="1" type="log" />  <Item reward="2" type="planks" />  <Item reward="4" type="stick" />  <Item reward="4" type="crafting\_table" />  <Item reward="8" type="wooden\_pickaxe" />  <Item reward="16" type="cobblestone" />  <Item reward="32" type="furnace" />  <Item reward="32" type="stone\_pickaxe" />  <Item reward="64" type="iron\_ore" />  <Item reward="128" type="iron\_ingot" />  <Item reward="256" type="iron\_pickaxe" /> |  |
| sparse\_total\_reward:  The total reward = the last value in the above reward accumulated sequence.  i.e. if duration\_steps = ds,  Total reward = reward\_seq(ds)  (if ds counts from 0, then it’s ds-1) |  |
| **Dense\_reward information** | **Dense reward accumulation sequence**  \*Similar with **Sparse\_reward** above EXCEPT that the reward will be accumulated **every time** when one item has been obtained, even if there are already same items (e.g. when getting the first log, reward+=1; when getting the second log, reward+=1…).  list of float (or int): [0.0,…, 1.0, 1.0, 2.0, 3.0,…]  len(list)= duration\_steps  \*Please check the following list for ‘how much reward to give the player for each item’. Only items shown in this list will get a reward.  <Item reward="1" type="log" />  <Item reward="2" type="planks" />  <Item reward="4" type="stick" />  <Item reward="4" type="crafting\_table" />  <Item reward="8" type="wooden\_pickaxe" />  <Item reward="16" type="cobblestone" />  <Item reward="32" type="furnace" />  <Item reward="32" type="stone\_pickaxe" />  <Item reward="64" type="iron\_ore" />  <Item reward="128" type="iron\_ingot" />  <Item reward="256" type="iron\_pickaxe" |  |
| dense\_total\_reward:  The total reward = the last value in the above reward accumulated sequence.  i.e. if duration\_steps = ds,  Total reward = reward\_seq(ds)  (if ds counts from 0, then it’s ds-1) |  |
| **'attack' information** | attack\_steps (int)  If an attack action is performed: this is an attack\_step. (i.e. how many steps the player did attack action) |  |
| Attack efficiency (float)  = total\_excavable\_inventory / attack\_steps  \*total\_excavable\_inventory = the total amount of **log**, **cobblestone** and **iron\_ore** the player got. This value can be abstracted from ‘**Accumulated amount of each item**’. |  |
| Attack ratio (float)  =attack\_steps/ duration\_steps |  |
| equipped\_and\_attack\_steps (int)  Number of steps that the player equipped **wooden\_pickaxe** or **stone\_pickaxe** and did attack actions at the same time |  |
| Equipped attack ratio (float)  = equipped\_and\_attack\_steps /attack\_steps  \*If a **wooden\_pickaxe** or **stone\_pickaxe** is equipped and an ‘attack’ action is performed, this is an equipped\_and\_attack\_step. |  |
| **'camera' information** | Camera position sequence  Type: list of 2d array of float  [ array([0., 0.]), array([0., 0.1]), …]  len(list)= duration\_steps  \* **'camera'** is an array of 2 floats range between [-180.0, 180.0] |  |
| Camera moving ratio (float)  =camera\_moving\_steps/ duration\_steps  if (camera\_t) – (camera\_t-1) ≠(0., 0.), it means the camera moved at t\_th step |  |
| 7 move actions: **back, forward, left, right, jump, sneak, sprint** | move\_steps (int)  \*It is a move\_step, when the action is one (or more than one) of ‘**back, forward, left, right, jump, sneak, sprint’.** move\_steps is the number of move\_step. |  |
|  | Position moving ratio  = move\_steps/ duration\_steps |  |
| **~~Misuse action information~~** | **~~misuse\_action\_steps (int)~~**  ~~\* if a player uses wooden\_pickaxe to dig iron\_ore, or uses stone\_pickaxe to dig log, this will be regarded as misuse of equipment.~~  **~~\*~~** ~~misuse\_action\_steps is the number of steps that the player did equipment misuse~~ |  |
| ~~The ratio of equipment misuse.~~  ~~=misuse\_action\_steps/ duration\_steps~~  ~~PS: In~~ [~~Action Space~~](https://minerl.io/docs/environments/index.html#id23) ~~and~~ [~~Observation Space~~](https://minerl.io/docs/environments/index.html#id22) ~~of MineRL:~~  **~~- 'equip'~~**~~(wooden\_pickaxe, stone\_pickaxe)~~  **~~- 'inventory'~~**~~(log, iron\_ore)~~  **~~- 'attack'~~**~~=true~~ |  |
| **Place item information** | **placed\_items**  Type: a list of 4 integers  [torch\_placed, cobblestone\_placed, dirt\_placed, stone\_placed ]  i.e. Total number of **torch, cobblestone**, **dirt**, and **stone** that were placed.  PS: In [Action Space](https://minerl.io/docs/environments/index.html#id23) of MineRL:  **'place'** (torch, cobblestone, dirt, stone) |  |
| **'if\_smelt\_coal' information** | if\_smelt\_coal=True (Boolean type),  i.e. did the player once smelt coal? if the player once smelt coal, instead of dig the coal from mineral. |  |