Data analysis was dealt at two levels, one considering player's information given its performance in a single match (match level) and one taking into account player's statistics over multiple games (player level). In order to distinguish between these cases, two distinct datasets were employed, $\mathcal{M}$ and $\mathcal{P}$ respectively. The first one was created with data gathered directly form the Open Dota platform and the second one by aggregating such data on the base of singular users. Moreover, for ours analysis we also employed the $\overline{\mathcal{M}}$ datasets by imposing a constraint on the entries of $\mathcal{M}$ and by adding some features thanks to our domain knowledge. We now discuss the description of the features utilized in each of these datasets.

1 Match Features

The features illustrated in this Section are those used for the $\mathcal{M}$ dataset.

Panoramic

Panoramic features provide an overview of the match at its conclusion. It includes ids of the game, the player and the hero employed, some numbers that supply a shallow summary of the performance statistics, party-related information, and reports about the objects equipped. Instead of directly consider which items are present in player's inventory and backpack, we categorized them on the base of their purposes within the game, highlighting characteristics that we identified as meaningful. Besides that, we also added a *build* feature, expressed as a number between 0 and 3, in order to better understand whether the items utilized are more oriented toward a defensive (value closer to 0) or an offensive (value closer to 3) gaming style. Features comprehended in this section therefor are:

- match_id
- account_id
- hero_id
- player_slot
- level

- kills
- deaths
- assists
- last_hits
- denies
- net_worth
- hero_damage
- hero_healing
- support_cons
- build
- farm
- mobility
- incomplete
- party_id
- party_size

Note that attributes party_id and player_slot are not used in the predictions, since they are only needed to extract more meaningful information about party size and chat analysis respectively.

Benchmarks

This kind of attributes take care of comparison with respect to recent performances on the hero played, considering not just the matches of the player under examination but every match of Dota 2. The "comparison" is achieved by storing two values for each feature, one that indicates the actual player's score (raw value) and one that shows the percentage of recent games with that same hero with a score worse than that (pct value). In here are collected:

- gold_per_min.raw
- gold_per_min.pct
- xp_per_min.raw
- xp_per_min.pct
- kills_per_min.raw
- kills_per_min.pct
- last_hits_per_min.raw
- last_hits_per_min.pct
- hero_damage_per_min.raw
- hero_damage_per_min.pct
- hero_healing_per_min.raw
- hero_healing_per_min.pct
- stuns_per_min.raw
- stuns_per_min.pct
- tower_damage.raw
- tower_damage.pct
- lhten.raw
- lhten.pct

Performance

Performance features are used to evaluate gaming skills by collecting:

- multi_kills
- kill_streaks

- stuns
- camps_stacked
- life_state_dead
- buybacks
- pings

Laning

These attributes are trivially associated with lane role, which can be roaming, safe, mid, or off. Since this aspect in particularly crucial in early phase of the match, lane efficiency (i.e., percentage of lane gold obtained) and denials are considered at ten minutes. Usually also last hits at ten are reported, however we already accounted for such value in the benchmarks section and therefore we avoid to consider that again to prevent duplicates. Thus, the features belonging to this section are simply:

- lane_role
- lane_efficiency
- dn@10

Farm

Farm features provide more specific details about the kills and are respectively:

- hero_kills
- creeps
- neutral_kills
- ancient_kills
- tower_kills
- courier_kills

- roshan_kills
- observer_kills
- necronomicon_kills

Actions

Such attributes lay out the amount of times a player executed a certain kind of action, providing a more elaborated distinction among them. Specifically, they are divided on the base of the type (move, attack or cast spell) and on the target they are aiming to:

- actions_per_minute
- action_move_to_position
- action_move_to_target
- action_attack_move
- action_attack_target
- action_cast_position
- action_cast_target
- action_cast_no_target
- action_hold_position
- action_radar

Objectives

Illustrate the amount of damage dealt to towers, barracks, ancient, and Roshan. In the case of the first two, data is reported for each independent structure, with code names based on their position:

- t1
- m1

- b1
- t2
- m2
- b2
- t3
- m3
- b3
- raxt
- raxm
- raxb
- t4
- ant
- roshan

Runes

Features about runes simply outline the number of times a certain type of rune was used by the player. Each rune has different effects and spawn on fixed positions of the game map. In here are gathered:

- double damage
- haste
- illusion
- invisibility
- regeneration
- bounty
- arcane

Vision

Vision-related attributes are associated with the purchase and usage of two particular items that are the observer and the sentry; Both of them are used to make visible something that normally would not be, that are ground and invisible enemies respectively. Also features about their average duration are gathered, since, despite their time limited usage, they can be eliminated by enemies when discovered:

- purchase_observer
- use_observer
- duration_observer
- purchase_sentry
- use_sentry
- duration_sentry
- purchase_dust
- use_smoke_of_deceit
- pur_smoke_of_deceit
- pur_gem

Cosmetics

Cosmetics prices is the only feature comprised into this section. Note that purchasable items are for customization only (they do not provide any in-game advantage).

Chat

Features about chats assemble numeric information about player's chat usage, differentiating messages between radial (pre-written messages) and typed ones. A variety of values describing the amount times chat is utilized, messages lengths and chat ranking therefore are collected with the purpose to evaluate how much the player interacts with others through chat:

- msg_tot
- msg_radial
- msg_chat
- msg_tot_ratio
- msg_radial_ratio
- msg_chat_ratio
- msg_char_sum
- len_msg_avg
- len_msg_mode
- len_msg_median
- rank_msg_tot
- rank_chat
- rank_radial

Extra Features

In this section are gathered a range of different useful attributes that do not belong to other groups, some information derived by metadata and values slightly re-elaborated from those originally extrapolated by Open Dota, in particular about the hero employed for the match (e.g., its role, complexity, etc.):

- rank
- dota_plus
- randomed
- weekday
- hour

- Carry
- Support
- Nuker
- Disabler
- Pusher
- Initiator
- Escape
- Durable
- Jungler
- hero_primary_attr
- hero_attack_type
- hero_complexity
- hero_gender
- hero_species
- lobby_type
- radiant_win
- win
- teamfight_participation
- perc_gold_spent

2 Match Features with Domain Knowledge

For the creation of dataset $\overline{\mathcal{M}}$, we derived some additional features from chat analysis. We adopted two strategies:

1. **Categorization of chat messages** (i.e., wrote by the players). Here we defined common words used in different contexts, i.e., laugh, slang, bad and good behaviour, and provocative messages at the end of the game (e.g., "gg ez"). We also searched for messages containing only "?" (which is highly provocative for the other team), number of question marks, exclamative marks, capital letters, early messages (sent before the game begins, usually sent to make noise or interact with the other team), and messages after a kill (which can be for complaining, provoking, or similar reasons). We identified such words by exploring websites (e.g., Dota2 forums, urban dictionary), chats of multiple games, and by our gaming experience. Then, we count the occurrences of such words in the chats typed by the player. The features obtained according to this procedure are:

    - counterLaugh
    - counterThank
    - counterDeny
    - ggez_counter
    - bad_counter
    - slang_counter
    - question_mark_counter
    - counter_capital
    - counter_question
    - counter_exclamation
    - earlySpeak
    - spam_kill

2. **Categorization of chatweel messages** (i.e., pick from pre-defined messages, and automatically translated by the game to the receiver's language). Here we categorized chatwheel messages that are pre-defined

by the game. These are useful since we do not incur in translation problems. We manually analyzed the file chat_wheel.json (containing, in english, all the chatwheel messages) and identified various types of messages, i.e., tactics, good behavior, and caster phrases. Moreover, we extracted which of them are reproduced as a sound (audible either in local or global chat), or are sprays left on the ground. Then, there is a second chatwheel, customised for each hero. Such messages contains general information, such as if they are laugh or deny messages, which we extracted as feature. Last, we extracted information such as whether the message was sent locally (same team) or globally (to both teams), and whether the message was an hero message. Therefore, we extracted the features:

- counterTattic
- counterGB
- counterCaster
- counterCasterIn
- counterSounds
- counterSoundsAllChat
- counterSoundsInChat
- counterAllChat
- hero_msg

All the aforementioned features can be of extreme help for our task. For instance, the use of slang could be higher for younger players. Provocative messages can relate to both neurotic and extrovert people. Agreeableness could be related to the use of good behavior messages. Conscious people could use more tactics message, and openess could relate to early messages or laugh messages after a kill. The gender could be affected from several of such typing and hero chatwheel usage. Last, some hero messages are available only on purchase, which can be an indicator of the occupation and buy_content behavior. For this reason, the features reported in this Section were **added** to those presented for $\mathcal{M}$ to create $\overline{\mathcal{M}}$.

3 Player Features

In the following we illustrate the manipulation process we used in order to build the $\mathcal{P}$ dataset. It was created by aggregating data with the same *account_id* within the $\mathcal{M}$ database, described in the precedent Section, and using as new values the mean of such grouped data, therefore the numerical features are the same.

Some adjustments had to be done for **non numerical features**, since it's not possible to compute the average of Booleans nor categorical values:

- For the feature indicating hero_id we kept the most frequently used parameter;
- Dropped match_id, party_id, player_slot, and radiant_win;
- Columns of extra features about hero's role were kept, but Boolean values were substituted with the percentage of matches in which the hero of choice adhered to the respective role in-game. The same was done for dota_plus;
- Lane role, hour, and weekday were expanded in a sort of one-hot encoding fashion. That is, a column was created for each possible value of both features and the ratio between the number of games with such value and the total number of matches made by the player was used as new feature entry. The same transformation was applied to hero's attributes accounted in the extra features section of match level dataset. Therefore, the newly added columns are:
  - lane_role_mid
  - lane_role_off
  - lane_role_roaming
  - lane_role_safe
  - mon
  - tue
  - wed
  - thu
  - fri

  - sat
  - sun
  - night
  - morning
  - afternoon
  - evening
  - hero_pr_attr_agi
  - hero_pr_attr_int
  - hero_pr_attr_str
  - hero_melee_atk
  - hero_complexity_1
  - hero_complexity_2
  - hero_complexity_3
  - hero_gender_A
  - hero_gender_F
  - hero_gender_M
  - hero_species_beast
  - hero_species_human
  - hero_species_humanoid
  - hero_species_spirit

- For each player we considered the three most used characters and report their mode for the values of primary attribute, attack type, complexity, gender, and species. We also added an attribute to represent the percentage of matches played with the top heroes over the total number of games. For limit cases we decided to simply perform the same procedure with the available number of heroes. Therefore, the added features are:

  - top_hero_pr_attr
  - top_hero_atk_type
  - top_hero_complexity

  - top_hero_gender
  - top_hero_species
  - top_usage

- we introduced a new feature to represent the *versatility* of the user, which highlights how much the player is prone to use different heroes and is expressed as a number between 0 and 1;

- information about lobby type (normal or ranked) was unified with the number of wins and, therefore, for every player are reported the amounts of:

  - normal_games
  - normal_win
  - ranked_games
  - ranked_win
  - win
  - games