Burnout Play Style

```
%%HTML
<style>
div.prompt {display:none}
</style>
```

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from config import test_data_loc, color_mapping
from helper import get_test_data, get_plot_data
```

Loading and Processing Data

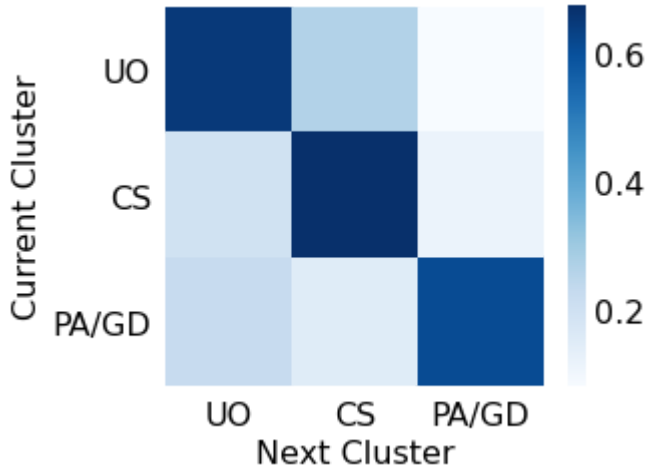
```
df = get_test_data(test_data_loc, "burnout")
dfg = df.groupby("key")
```

Transitions Heatmap

```
temp = dfg.agg({"cluster":"unique", "part_id":"unique"})
temp["cluster"] = temp["cluster"].apply(lambda x: int(x[0]))
temp["part_id"] = temp["part_id"].apply(lambda x: int(x[0]))
temp = temp.sort_values(by="part_id")
temp = temp["cluster"].tolist()
```

```
mapping = {
    0: 0,
    3: 1,
    4: 2,
    6: 3,
}

heatmap = np.zeros((len(set(temp)),len(set(temp))))
i = 0
while i < len(temp)-1:
    heatmap[mapping[temp[i]]][mapping[temp[i+1]]] += 1
    i += 1

heatmap = np.delete(np.delete(heatmap, (1), axis=0), (1), axis=1)
heatmap = np.divide(heatmap, heatmap.sum(axis=1).reshape(-1,1))
```

```
fig, ax = plt.subplots(figsize=(4.5,3.5))
_ = sns.heatmap(
    heatmap,
    robust=True,
    square=True,
    cmap="Blues",
    xticklabels=["UO","CS","PA/GD"],
    yticklabels=["UO","CS","PA/GD"]
)
_ = plt.xlabel("Next Cluster", size=16)
_ = plt.ylabel("Current Cluster", size=16)
_ = plt.tick_params(left=False, bottom=False)
_ = plt.xticks(rotation=0, size=16)
_ = plt.yticks(rotation=0, size=16)

cbar = ax.collections[0].colorbar
cbar.ax.tick_params(length=0, labelsize=16)
cbar.set_ticks(np.linspace(0,1,6))
plt.tight_layout()
plt.show()
```

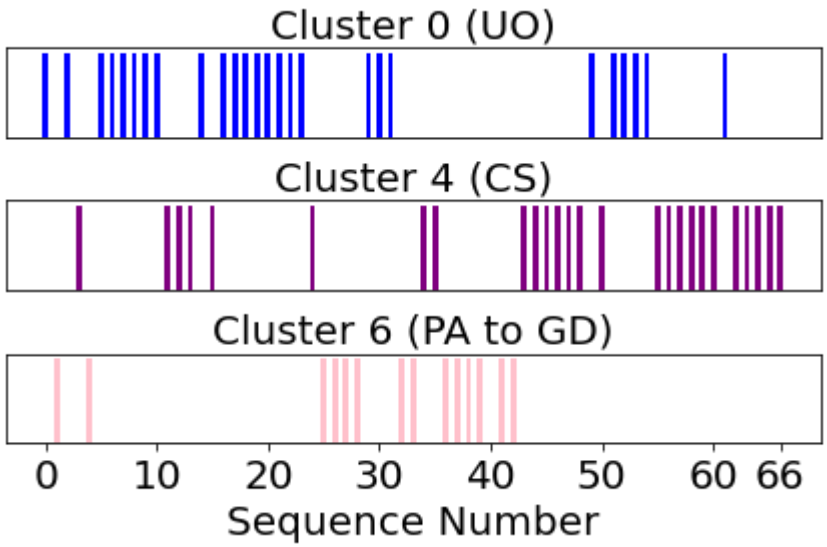


Temporal Cluster Allocation Plot

```
temp = df.groupby("cluster").agg({"part_id":"unique"})
```

```
max_x = max(
    max(temp.iloc[0]).max(),
    max(temp.iloc[1]).max(),
    max(temp.iloc[2]).max()
)
```

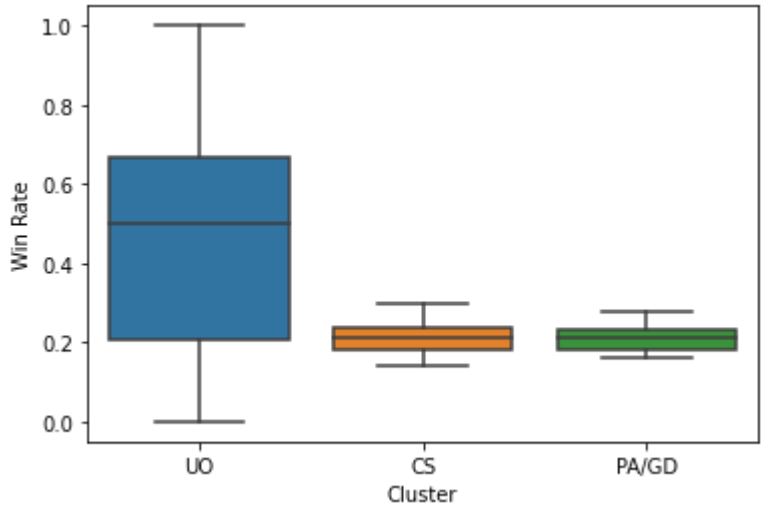
```
fig, ax = plt.subplots(3,1, sharex=True);

ax[0].bar(
    temp.loc[0,"part_id"],
    [1]*temp.loc[0,"part_id"].shape[0],
    color=color_mapping[0],
    width=0.5
)
ax[0].set_title("Cluster 0 (UO)", size=20)
ax[0].xaxis.set_ticks_position('none')
ax[0].get_yaxis().set_visible(False)
ax[1].bar(
    temp.loc[4,"part_id"],
    [1]*temp.loc[4,"part_id"].shape[0],
    color=color_mapping[4],
    width=0.5
)
ax[1].set_title("Cluster 4 (CS)", size=20)
ax[1].xaxis.set_ticks_position('none')
ax[1].get_yaxis().set_visible(False)
ax[2].bar(
    temp.loc[6,"part_id"],
    [1]*temp.loc[6,"part_id"].shape[0],
    color=color_mapping[6],
    width=0.5
)
ax[2].set_title("Cluster 6 (PA to GD)", size=20)
ax[2].set_xlabel("Sequence Number", size=20)
ax[2].xaxis.set_ticks([*range(0, max_x, 10), max_x])
ax[2].get_yaxis().set_visible(False)

plt.xticks(size=20)
plt.tight_layout()
plt.show()
```



Graphical Analysis - Aggregate Statistics

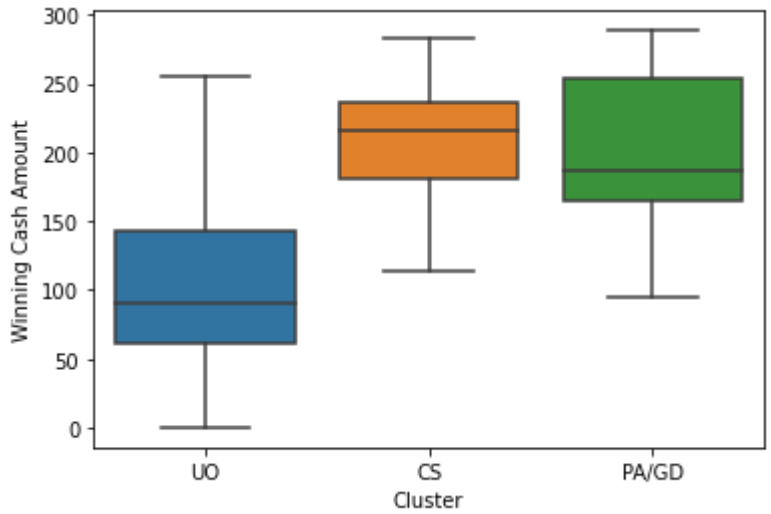
Win Rate

```
_ = sns.boxplot(
    data=get_plot_data(dfg, "win_status", "mean", cluster_idx={0,4,6}),
    x="cluster",
    y="win_status",
    showfliers=False,
)
_ = plt.xticks(range(3), ["UO","CS","PA/GD"])
_ = plt.xlabel("Cluster")
_ = plt.ylabel("Win Rate")
```



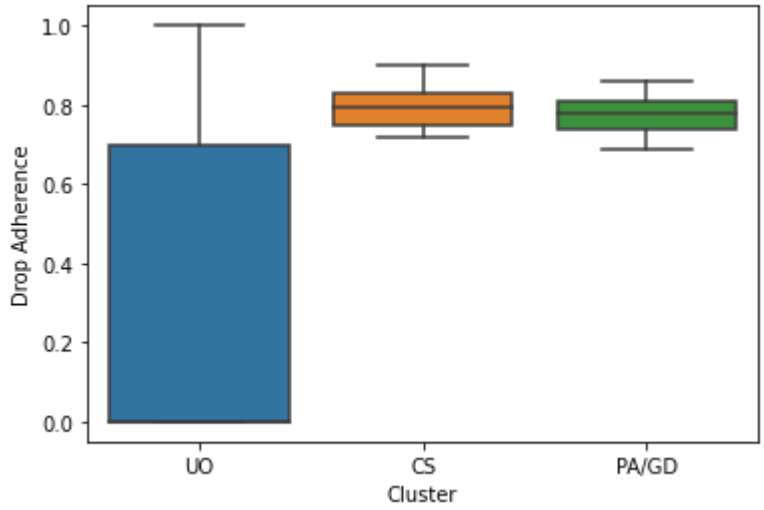
Winning Cash Amount

```
_ = sns.boxplot(
    data=get_plot_data(dfg, "winning_cash_amt", "mean", cluster_idx={0,4,6}),
    x="cluster",
    y="winning_cash_amt",
    showfliers=False,
)
_ = plt.xticks(range(3), ["UO","CS","PA/GD"])
_ = plt.xlabel("Cluster")
_ = plt.ylabel("Winning Cash Amount")
```



Drop Adherence

```
_ = sns.boxplot(
    data=get_plot_data(dfg, "drop_Adherance", "mean", cluster_idx={0,4,6}),
    x="cluster",
    y="drop_Adherance",
    showfliers=False
)
_ = plt.xticks(range(3), ["UO","CS","PA/GD"])
_ = plt.xlabel("Cluster")
_ = plt.ylabel("Drop Adherence")
```



Mean Rumble Games

```
_ = sns.boxplot(
    data=get_plot_data(dfg, "is_rumble", "mean", cluster_idx={0,4,6}),
    x="cluster",
    y="is_rumble",
    showfliers=False
)
_ = plt.xticks(range(3), ["UO","CS","PA/GD"])
_ = plt.xlabel("Cluster")
_ = plt.ylabel("Mean Rumble Games")
```