Project 3

[redacted]

Introduction:

We are looking at the horror movies dataset from TidyTuesday. This dataset contains metadata for horror movies from the Movie Database. The whole dataset is ~36k records, but we will exclude obscure movies that do not have a poster image available or not released yet. We selected the top 10k remaining records by popularity for this analysis.

In this project we only used the year of the movie and the poster image URL from the dataset.

Question:

Are there particular pattern in the color scheme of horror movie posters? How do the popularity of these patterns change over time?

Approach:

Firstly we need to download the poster image from the URL specified in the dataset, and then extract the color features from the image. This part is too computationally intensive to run in this document and was implemented in Rust as an R extension, compiled and ran on the Edu Pod.

We wrote extractors that extracts the following into a CSV file:

- Brightness, by averaging, in the range [0, 255]. In RGB space.
- Contrast, by RMSD. Output is in the range [0, 255]. In RGB space.
- Dominant colors, by an adaptive k-means clustering algorithm which tries to use between 3 and 8 color clusters. The optimal number of clusters is selected using the silhouette score (within sum of squares), when adding another cluster reduces the score by less than 30% the algorithm stops. Each trial is repeated 8 times with different initializations and the best result is selected. The clusters are then sorted by size and the top 3 clusters are output. This is done in LAB space because the LAB space is more "sphere-like", with two components mapping to hue making it easier to interpret as color "tendencies".

If you are not familiar with the LAB colorspace, it has three components:

- L: Lightness, in the range [0, 100].
- A: Green to red, in the range [-90, +90].
- B: Blue to yellow, in the range [-90, +90].

The download and extractions took about 20/5 minutes each on Edu Pod using 32 threads, I included an output CSV file on GitHub gist and we will use that for the analysis.

If you want to generate the CSV file yourself, a sample pipeline is provided at the end of this document.

After we loaded everything into R, we run a second k-means on the extracted features. Then we will interpret the clusters based on their deviation from the mean in each feature.

Lastly, we plot the % of movies in each cluster over time and observe any trends. This is a multiple time-series so we will use a faceted line plot.

```
# Load your dataset here
fetch.csv(horror_movies,
    "https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2022/2022-11-01/horror_m
```

```
    where = global_env()
)
horror_movies %<>%
    # filter out movies with no poster or too low in popularity
    filter(status == "Released") %>%
    filter(!is.na(poster_path)) %>%
    arrange(desc(popularity)) %>%
    slice_head(n = 10000) %>%
    mutate(
        filename = str_remove(poster_path, "^/"),
    )

# load directly from processed output
# image_features <- read_csv("data/horror_movies_image_features.csv") %>%
#    distinct(filename, .keep_all = TRUE)

# data file with extracted features
fetch.csv(image_features,
    "https://gist.githubusercontent.com/eternal-flame-AD/74a23ff3086a3815ebd8216f9ba37cbe/raw/4911db6ec
    where = global_env()
)
image_features %<>%
    distinct(filename, .keep_all = TRUE)

# join the two datasets
horror_movies.joined <- horror_movies %>%
    inner_join(image_features, by = "filename", keep = FALSE)
```

Analysis:

To limit the scope of interpretation, we will only include the top 2 dominant colors along with the brightness and contrast features for clustering.

Cluster number of 6 is selected based on the elbow method on the within sum of squares metric.

```
do.kmeans <- function(centers) {
    set.seed(123)

    horror_movies.joined %>%
        select(
            # first dominant color
            starts_with("color_cluster_0"),
            # second dominant color
            starts_with("color_cluster_1"),
            starts_with("brightness_"),
            contrast_avg
        ) %>%
        scale() %>%
        kmeans(
            # some variables are too similar
            # for the default algorithm
            algorithm = "Lloyd",
            centers = centers,
            iter.max = 500,
            nstart = 25
```

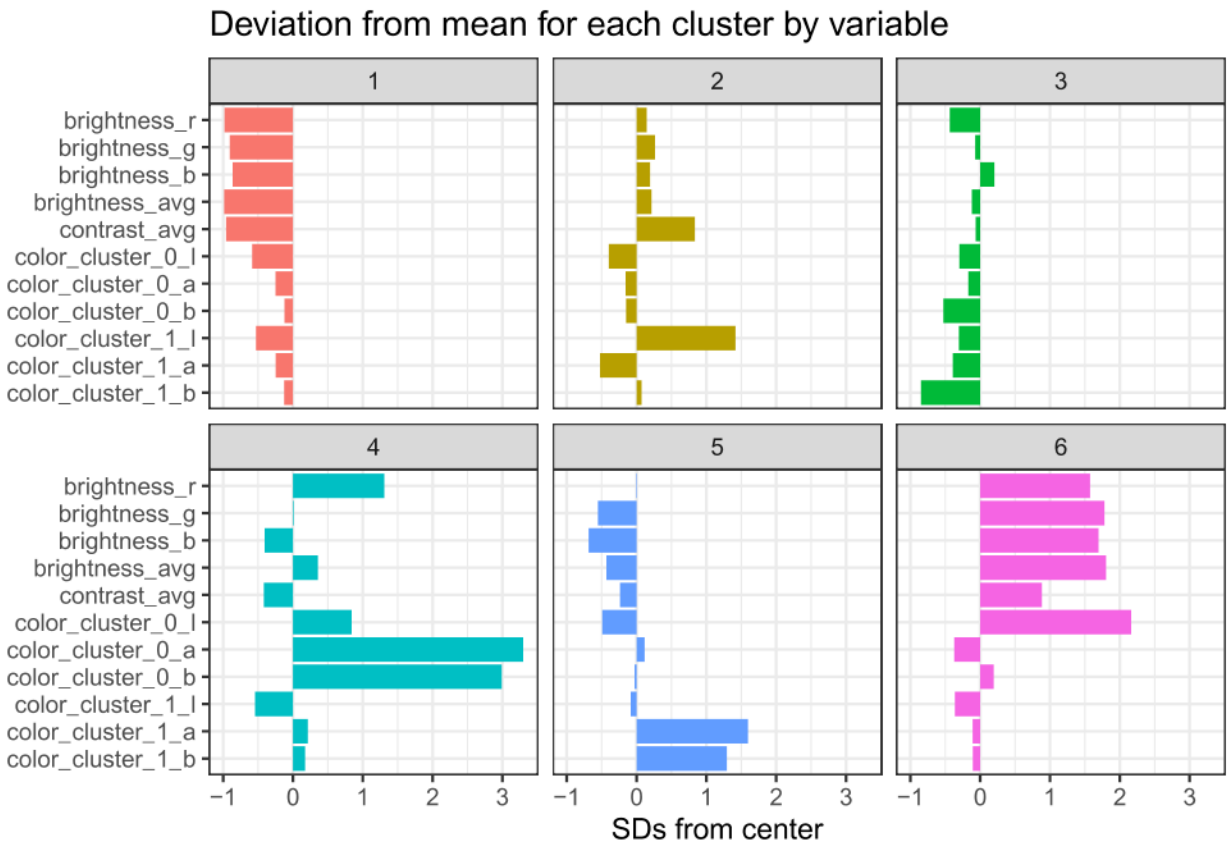
```
        )
}

km_fit <- do.kmeans(6)
```

First we will plot the deviation in each variable from overall center for each cluster.

This will give us an idea of "where" each cluster is and how they each differ from others.

```
km_fit$centers %>%
    as_tibble() %>%
    rownames_to_column("cluster") %>%
    pivot_longer(-cluster, names_to = "parameter", values_to = "value") %>%
    mutate(
        parameter = factor(parameter, levels = c(
            "brightness_r",
            "brightness_g",
            "brightness_b",
            "brightness_avg",
            "contrast_avg",
            "color_cluster_0_l",
            "color_cluster_0_a",
            "color_cluster_0_b",
            "color_cluster_1_l",
            "color_cluster_1_a",
            "color_cluster_1_b"
        ))
    ) %>%
    ggplot(aes(fct_rev(parameter), value)) +
    geom_col(aes(fill = cluster), position = "dodge") +
    coord_flip() +
    facet_wrap(~cluster) +
    labs(
        y = "SDs from center",
        x = NULL,
        title = "Deviation from mean for each cluster by variable"
    ) +
    theme_bw() +
    theme(
        legend.position = "none"
    )
```



We interpreted this plot as follows:

```
# join the cluster labels to the original dataset
horror_movies.clustered <- km_fit %>%
    augment(horror_movies.joined) %>%
    mutate(
        cluster = centroid.as.factors(.cluster)
    )

horror_movies.clustered %>%
    count(cluster) %>%
    mutate(
        prop = n / sum(n)
    ) %>%
    mutate(
        name = as.character(cluster)
    ) %>%
    inner_join(
        centroid.def
    ) %>%
    arrange(idx) %>%
    select(idx, name, human, n, prop) %>%
    knitr::kable(
        caption = "Cluster Naming and Interpreting Cluster Characteristics",
        col.names = c("Cluster", "Shorthand Name", "Description", "Count", "Proportion")
    ) %>%
    kable_styling(
```

```
        full_width = FALSE,
        latex_options = c("striped", "hold_position")
    )
```

```
## Joining with `by = join_by(name)`
```

Table 1: Cluster Naming and Interpreting Cluster Characteristics

Cluster	Shorthand Name	Description	Count	Proportion
1	-Br-Ct	Just Dark	2329	0.2329
2	+Ct 1+L-a	Bright on Dark	1990	0.1990
3	Nil	No significance	2148	0.2148
4	+Br 0+a+b	Bright Red/Yellow	492	0.0492
5	1+a+b	Dark with Warm Secondary Color	1580	0.1580
6	+Br+Ct 0+L	Bright and Vibrant	1461	0.1461

To illustrate the clusters, and visually confirm our interpretation, we will select 12 posters from each cluster and plot them on a tile plot.

```
if (require(ggpattern, quietly = TRUE)) {
    showcase <- horror_movies.clustered %>%
        group_by(cluster) %>%
        mutate(
            ord = row_number()
        ) %>%
        filter(ord <= 12)

    showcase %>%
        ggplot(
            aes(x = cluster, y = ord)
        ) +
        geom_tile_pattern(
            aes(pattern_filename = filename),
            pattern = "image",
            pattern_type = "expand"
        ) +
        coord_flip() +
        scale_pattern_filename_manual(
            values = set_names(showcase$url, showcase$filename)
        ) +
        labs(
            x = NULL, y = NULL,
            title = "Showcase for Clustered Movie Posters"
        ) +
        scale_x_discrete(
            expand = c(0, 0),
            labels = humanize.cluster
        ) +
        scale_y_continuous(
            expand = c(0, 0)
        ) +
        theme(
            axis.title.x = element_blank(),
            axis.text.x = element_blank(),
```

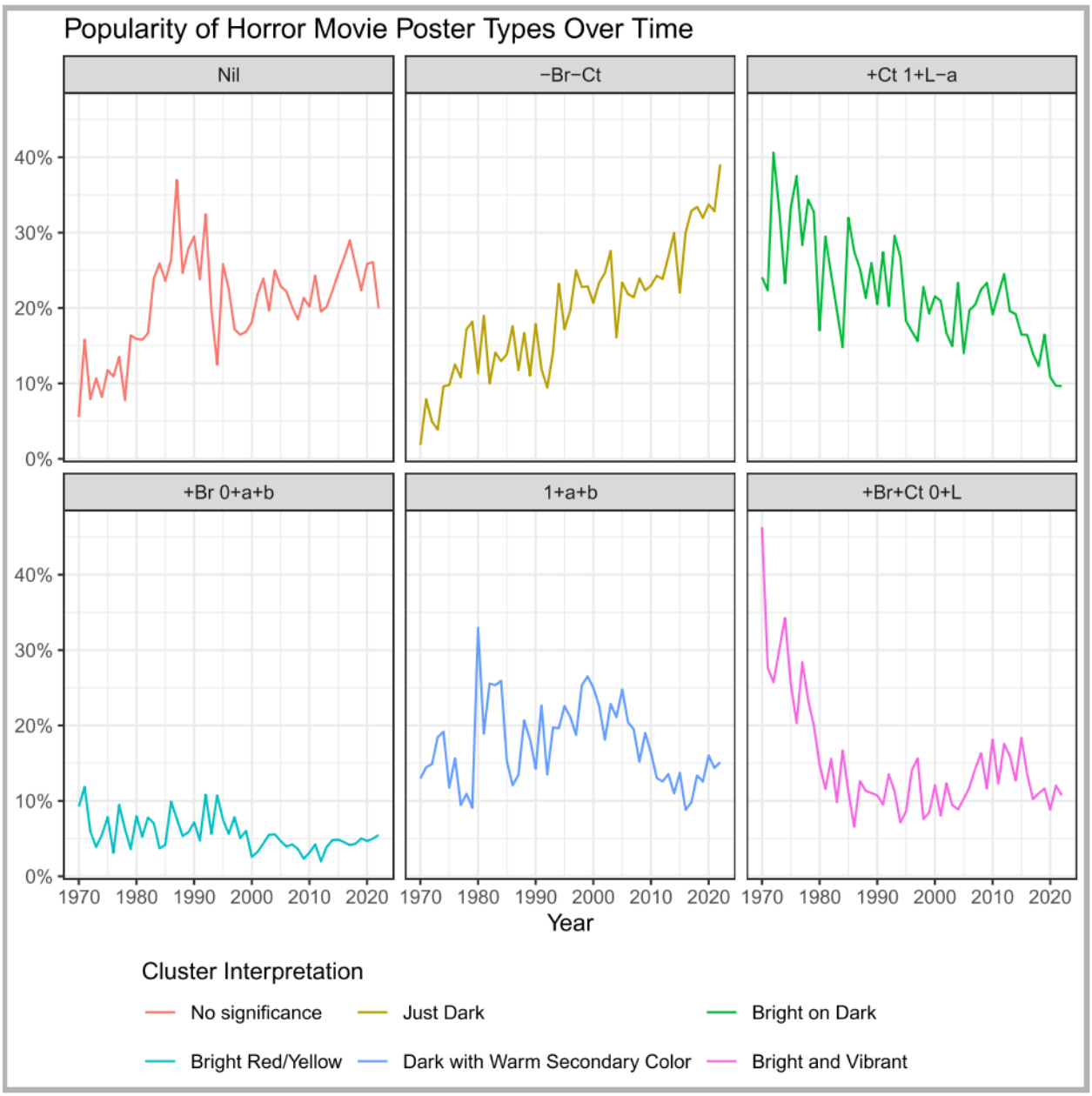
```
            axis.ticks.x = element_blank(),
            axis.text.y = element_text(size = 8),
            axis.ticks.y = element_blank(),
            legend.position = "none"
        )
} else {
    warning("ggpattern not installed, skipping showcase")
}
```

Visually inspecting the output it does seem that the interpretation of centroids is reasonable.

Thus we move on to plot the popularity of each type over year.

```
horror_movies.clustered %>%
    mutate(year = year(release_date)) %>%
    select(cluster, year) %>%
    arrange(year) %>%
    # filter year range and count up by cluster and year
    filter(year >= 1970) %>%
    count(cluster, year) %>%
    # compute proportions
    group_by(year) %>%
    mutate(prop = n / sum(n)) %>%
    ggplot(aes(year, prop, color = cluster)) +
    geom_line() +
    scale_y_continuous(labels = scales::percent) +
    facet_wrap(~cluster) +
```

```
    scale_color_discrete(
        labels = humanize.cluster,
        guide = guide_legend(
            title = "Cluster Interpretation",
            title.position = "top",
            label.hjust = 0,
            nrow = 2,
            byrow = TRUE
        )
    ) +
    labs(
        x = "Year",
        y = NULL,
        title = "Popularity of Horror Movie Poster Types Over Time"
    ) +
    theme_bw() +
    theme(
        plot.background = element_rect(
            fill = "white",
            color = "grey70", linewidth = 2
        ),
        legend.position = "bottom",
        legend.margin = margin(t = 0, r = 0, b = 0, l = -1.5, unit = "cm"),
    )
```



Discussion:

Within the time range of 1970 to 2022, We made the following observations:

- The proportion of "Just Dark" types of posters has been increasing over the years. It was very rare in the 70s but has grown to about 40% after 2020.
- The "Bright on Dark" type used to be popular but has been declining in population ever since.
- The "Bright Red/Yellow" pattern is staying rare at <10% from the 70s and staying at ~5% in the 21st century.
- The "Dark with Warm Secondary Color" has a steady popularity of around 20% for the past 50 years.
- The "Bright and Vibrant" cluster has been popular in the 70s at a rate of about 30% while afterwards it reduced to about 10%.

Supplementary Information:

Image download and extraction pipeline:

A source tarball of the image processor is available here. Extract the tarball in the same directory as this document and run `load.extractor.api()` as defined below. Then run `imaging.download()` to download the images and `imaging.extract()` to extract the features.

```
# workflow for downloading and extracting features
load.extractor.api <- function(loglevel = "INFO") {
    if (exists(".imaging.api.loaded")) {
        return()
    }
    .imaging.api.loaded <<- TRUE

    # generate input for image downloader/feature extractor
    if (!file.exists("data/horror_movies_urls_in.csv")) {
        horror_movies.urls <- horror_movies %>%
            transmute(
                url = paste0("https://image.tmdb.org/t/p/w500", poster_path),
                filename = filename
            )
        write_csv(horror_movies.urls, "data/horror_movies_urls_in.csv")
    }
    if (!file.exists("imaging-lib/target/release/libimaging_lib.so")) {
        message("Compiling Rust library...")
        system2(
            "bash",
            c("-c", shQuote("cd imaging-lib && cargo build --release"))
        )
    }
    dyn.load("imaging-lib/target/release/libimaging_lib.so")
    .Call("imaging_lib_init", loglevel)
    imaging.download <<- function(input_filename = "data/horror_movies_urls_in.csv",
                                  output_filename = "data/horror_movies_images_out.csv",
                                  output_zip_filename = "data/images.zip",
                                  limit = 0L,
                                  overwrite = FALSE) {
        ret <- .Call(
            "imaging_lib_download",
            input_filename, output_filename,
            output_zip_filename, limit, overwrite
        )
        if (is.character(ret)) {
            message(ret)
        }
    }
    imaging.extract <<- function(input_filename = "data/horror_movies_images_out.csv",
                                 output_filename = "data/horror_movies_image_features.csv",
                                 input_zip_filename = "data/images.zip",
                                 num_threads = 32L,
                                 limit = 0L,
                                 overwrite = FALSE,
                                 verbose = FALSE) {
        ret <- .Call(
            "imaging_lib_extract",
            input_filename, output_filename,
            input_zip_filename,
```

```
            num_threads, limit,
            overwrite, verbose
        )
        if (is.character(ret) && ret != "") {
            message(ret)
        }
    }
}

demo.pipeline <- function(limit = 3, n_threads = 1) {
    tmpf1 <- tempfile()
    tmpf2 <- tempfile()
    tmpf3 <- tempfile()
    imaging.download(
        input_filename = "data/horror_movies_urls_in.csv",
        output_filename = tmpf1,
        output_zip_filename = tmpf2,
        limit = limit
    )
    imaging.extract(
        input_filename = tmpf1,
        output_filename = tmpf3,
        input_zip_filename = tmpf2,
        num_threads = n_threads,
        limit = limit,
        verbose = TRUE
    )
    unlink(c(tmpf1, tmpf2, tmpf3))
}
if (!is.selfcontain) {
    tryCatch(
        {
            load.extractor.api()
            demo.pipeline()
        },
        error = function(e) {
            message("Failed to load Rust library, skipping demo pipeline.")
        }
    )
}
```

```
## [INFO] [imaging_lib::rapi] Starting image download.
## [INFO] [imaging_lib::download] Progress: 1 of 3 completed. (9997 skipped, 0 failed) (33.33%)
## [INFO] [imaging_lib::download] Progress: 2 of 3 completed. (9997 skipped, 0 failed) (66.67%)
## [INFO] [imaging_lib::download] Progress: 3 of 3 completed. (9997 skipped, 0 failed) (100.00%)
## [INFO] [imaging_lib::rapi] Starting feature extraction.
## [INFO] [imaging_lib::features] Extracting features for hiaeZKzwsk4y4atFhmncO5KRxeT.jpg
## [INFO] [imaging_lib::features] n_clusters = 3, tot_withinss = 35995730.911893524, ratio = 0.48287119
## [INFO] [imaging_lib::features] n_clusters = 4, tot_withinss = 17381301.647811905, ratio = 0.70331645
## [INFO] [imaging_lib::features] Progress: 1 completed. (0 skipped, 0 failed)
## [INFO] [imaging_lib::features] Extracting features for xIGr7UHsKf0URWmyyd5qFMAq4d8.jpg
## [INFO] [imaging_lib::features] n_clusters = 3, tot_withinss = 50906880.35737884, ratio = 0.797322870
## [INFO] [imaging_lib::features] Progress: 2 completed. (0 skipped, 0 failed)
## [INFO] [imaging_lib::features] Extracting features for pHkKbIRoCe7zIFvqan9LFSaQAde.jpg
```

```
## [INFO] [imaging_lib::features] n_clusters = 3, tot_withinss = 35272703.024885, ratio = 0.57518928352
## [INFO] [imaging_lib::features] n_clusters = 4, tot_withinss = 20288480.781024937, ratio = 0.72323513
## [INFO] [imaging_lib::features] Progress: 3 completed. (0 skipped, 0 failed)
```