Principal Component Analysis

Principal Component Analysis (PCA) is a statistical technique used for dimensionality reduction and feature extraction, frequently used in exploratory data analysis and machine learning. It aims to transform the original variables into a new set of uncorrelated variables called "principal components," which capture most of the variance in the data. The first principal component accounts for the most variance, the second accounts for the second most, and so on. By selecting a subset of these principal components, one can reduce the dimensionality of the dataset while retaining most of its information.

The Idea of PCA is that it helps us reduce the dimensions of the data so that we can visualize it in 2D or 3D using the first 2 or 3 principal components.

PCA has uses in facial and sound recognition, helping reduce dimensionality of images and sounds and allowing faster processing.

In this project, I aim to visualize song data using a Spotify dataset to visualize and compare differences in song features among different artists. The dataset contains various features related to songs, such as 'acousticness', 'danceability', 'duration_ms', 'energy', 'instrumentalness', 'liveness', 'loudness', 'speechiness', and 'tempo' for around 2000 songs.

Finding the First Principal Component

We begin with n observations and p variables: $[X_1, X_2, ..., X_p]$

The first principal component is Z1, where:

$$Z_1 = \phi_{11}X_1 + \phi_{21}X_2 + ... + \phi_{p1}X_p$$

These coefficients are known as "loadings"

All squares of loadings add up to 1: $\sum \phi_{i1} = 1$

Weights of each loading are chosen based on maximum variance of data. In other words we are looking to find:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j1}^2 = 1.$$

This whole formula represents the variance of Z_1

Think of our facial recognition example:

- Each variable represents, x_i, a facial characteristic
- Because *eyes* vary a lot across faces (and, thus, help us distinguish faces more easily) they would receive a higher weight than *cheeks* or *ears*

Finding the Kth Principal Component

To find kth principal component (k=2,3,4,...), we apply same formula.

However, we need to add an extra constraint. We need to make loadings of kth component such that it is uncorrelated with all previous components. The formula now becomes:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{\text{maximize}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j2} x_{ij} \right)^2 \right\} \text{ subject to } \sum_{j=1}^{p} \phi_{j2}^2 = 1 \text{ and } Corr\{Z_2, Z_1\} = 0$$

- Second principal component is like saying: If two people had identical eyes (i.e., if characteristics of first principal component are identical)
  - which other facial characteristic(s) would we want to focus on?

The following loadings were calculated for all 9 Principal components using the prcomp() function in R:

Rotation (n x k) = (9 x 9):

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
acousticness	0.4726850	-0.12146366	0.30176017	-0.04604459	-0.14742939	0.03275418	-0.10171040	7.820598e-01	0.15448400
danceability	-0.1069035	-0.50473481	-0.55637644	-0.16362952	0.15125970	-0.08848245	-0.58719255	1.350254e-01	0.06123297
duration_ms	0.2198553	0.35349016	-0.53558743	-0.24041537	0.12608663	0.63642097	0.19641051	1.449366e-01	-0.03926424
energy	-0.5273737	0.24042349	-0.13841637	0.09780950	0.02969579	-0.11307934	0.16232364	3.068620e-01	0.70793055
instrumentalness	0.2634923	0.50333184	-0.27098191	-0.05612243	0.24134370	-0.69909138	-0.03099986	1.284988e-01	-0.19026159
liveness	-0.1538311	0.40711885	0.04963455	-0.35677411	-0.67654546	0.02587588	-0.46602128	-6.825979e-02	-0.02490652
loudness	-0.5462300	0.02406496	-0.01467826	0.16952339	-0.02399353	0.05917843	0.07263604	4.816244e-01	-0.65634156
speechiness	-0.1526795	-0.24028835	0.10575094	-0.83481025	0.04419717	-0.18843039	0.41213645	4.296779e-02	-0.04119455
tempo	-0.1546309	0.26518728	0.45395988	-0.21830072	0.64816339	0.21183250	-0.43204697	-7.528077e-05	0.01941226

How to Interpret Principal Components?

In analyzing a given component:

The larger the absolute value, the more it contributed to the component.

- **First component:** The First Principal Component includes the most important features of the data. This means that most of the variability found across songs can be explained by three variables:
    - Acousticness, energy, loudness are significantly higher than the others

	PC1
acousticness	0.4726850
danceability	-0.1069035
duration_ms	0.2198553
energy	-0.5273737
instrumentalness	0.2634923
liveness	-0.1538311
loudness	-0.5462300
speechiness	-0.1526795
tempo	-0.1546309

- The PC1 is telling us that the "vibe" of the song is what distinguishes most songs from each other
    - some songs are energetic, electric and loud
    - Others are acoustic and calm

- **After accounting for the first pc, second component tells us:**
    - Most of variability across songs is explained by how danceable, instrumental, or lively a song is

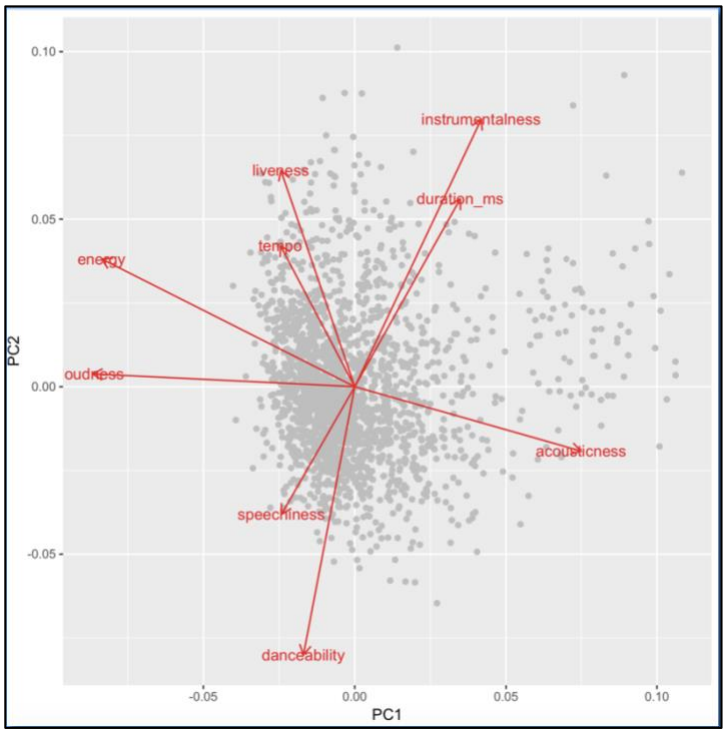
	PC1	PC2
acousticness	0.4726850	-0.12146366
danceability	-0.1069035	-0.50473481
duration_ms	0.2198553	0.35349016
energy	-0.5273737	0.24042349
instrumentalness	0.2634923	0.50333184
liveness	-0.1538311	0.40711885
loudness	-0.5462300	0.02406496
speechiness	-0.1526795	-0.24028835
tempo	-0.1546309	0.26518728

Plotting the Principal Components

We typically plot only the first 2 or 3 principal components to create a 2D or 3D plot of these components.

Let's plot the first 2 components in the Spotify dataset using autoplot() function in the "ggfortify" package:

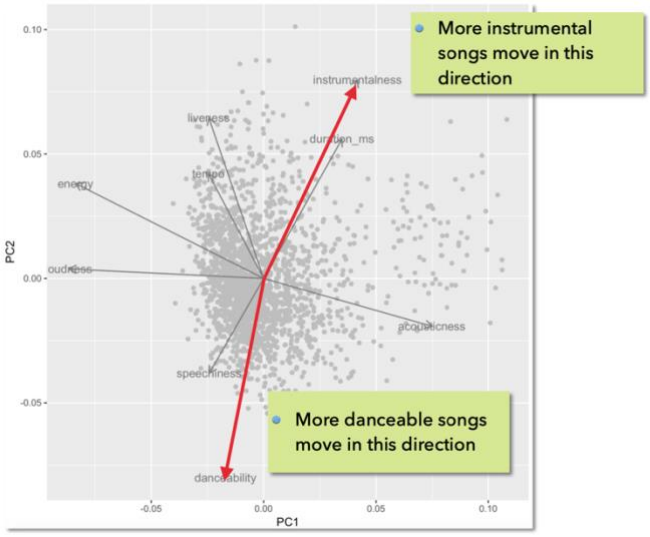
> autoplot(pca, data = spotify_vars, loadings = TRUE, loadings.label = TRUE)



How to Interpret Principal Component Graph?

- Plot is like a map of all songs
- Each gray dot represents a song
- Each song lives in coordinates (z1,z2)

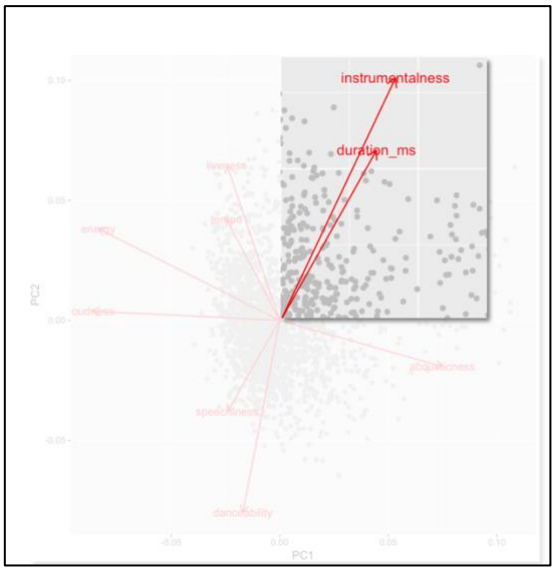
- The arrows represent the variables in our dataset



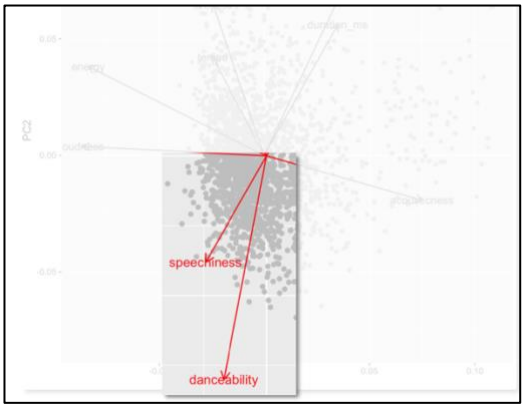
- When arrows are closer together (and have the same direction), variables are highly correlated

Examples:

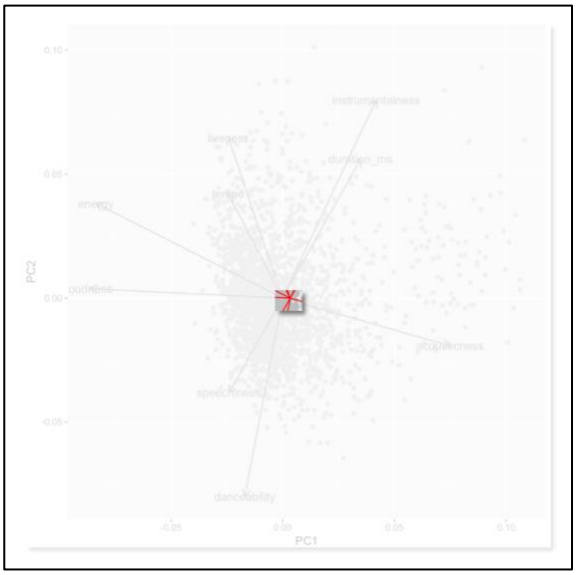
- Instrumentalness and Duration of a song are highly correlated, so songs with lots of instrumentality tend to last longer → Think of classical music!



- Danceability and speechiness move together, so more danceable songs also tend to have lots of lyrics.



- The center of the plot (0,0) means a song is average in all variables



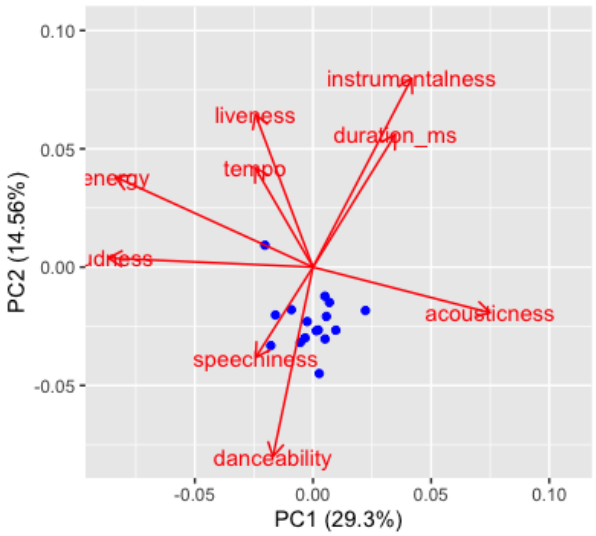
Studying Drake's Songs:

autoplot(pca, data = spotify_vars, loadings = TRUE,
col=ifelse(spotify_labels$artist=="Drake","blue","transparent"), loadings.label = TRUE)

Drake's songs:

- More speech
- More danceable
- Shorter
- Less instrumental
- Less lively
- Shorter tempo

than the average song



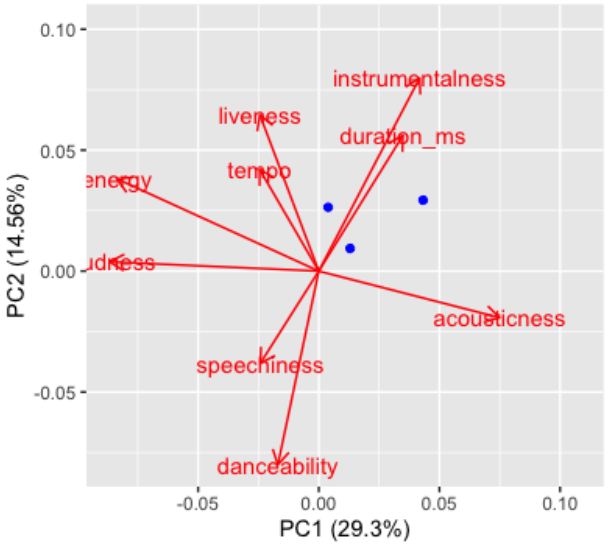
Studying Arcade Fire's Songs:

autoplot(pca, data = spotify_vars, loadings = TRUE,
col=ifelse(spotify_labels$artist=="Arcade Fire","blue","transparent"), loadings.label = TRUE)

Arcade Fire's songs:

- More instrumental
- Longer
- Higher tempo
- More lively
- Less speech (lyrics)
- Less danceable

than the average song



How many Principal Components should we analyze?

Ideally, we would like to use as few components as possible to capture variability in the data with high enough precision for our purpose.

To determine optimal number of components, we calculate the percentage-of-variance-explained and plot it using the following code:

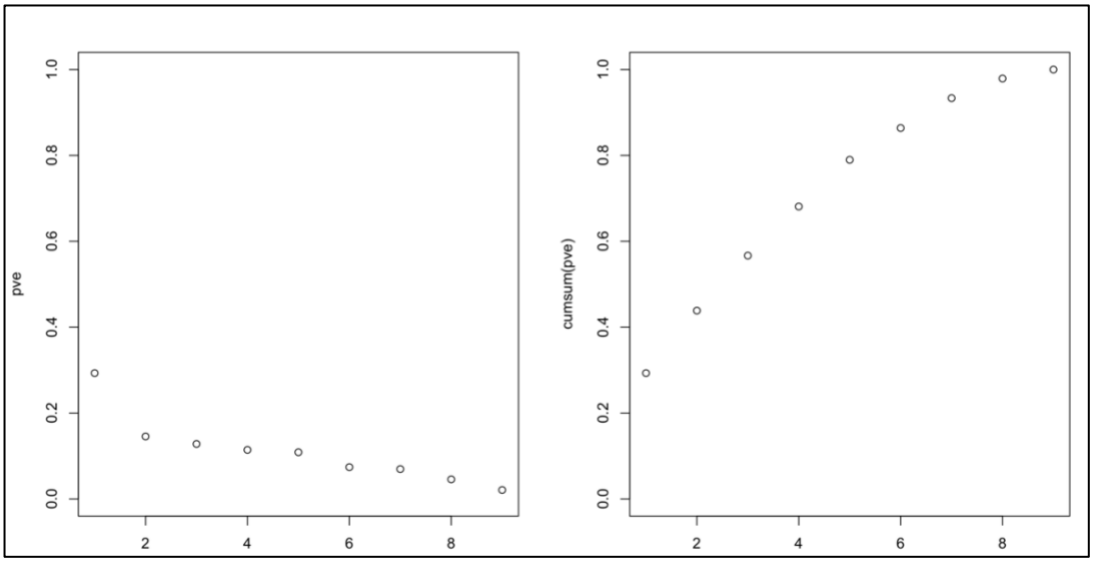
```
>pve=(pca$sdev^2)/sum(pca$sdev^2)
```

-This code calculates the percentage of variance explained (similar to r-squared in regression):

```
> pve
[1] 0.29301523 0.14556826 0.12806939 0.11430633 0.10879495 0.07408999 0.06958550 0.04566166 0.02090868
```

```
>par(mfrow=c(1,2))
>plot(pve, ylim=c(0,1))
>plot(cumsum(pve), ylim=c(0,1))
```

- These lines make the graph



- With 2 components, we only recognize songs with ~43% accuracy
- With 6 components, we recognize it with 80% accuracy

Ultimately, it is our judgement as to how much accuracy we want (i.e., how much variance we want to capture)!

Where is PCA used?

PCA has many applications, especially when it comes to dimensionality reduction:

- Image processing
- Voice recognition
- File compression (Zip files)
- Portfolio management (analyzing factors that affect stocks)
- Genetics

PCA is also an excellent tool to begin a regression analysis. It can help us determine how variables relate to each other and, hence, help us choose predictors.