0.0.1 Question 6a

What does each row in `df_clean` represent?

df.iloc[:, -15:]:Choose last 15 columns

.drop(['Unnamed: 60'], axis = 1): Drop Column name with 'Unnamed: 60'

.rename(columns = {"2000 ‡": "2000", "2016 ‡": "2016", "State.1": "State"}) :Rename column "2000 ‡" to "2000", "2016 ‡" to "2016", "State.1" to "State"

.drop([51]) : Drop row 51

.set_index("State"): Set column "State" as index

Unfortunately, we have two problems:

1. There is a lot of overplotting, with only 28 distinct dots (out of 104 points). This means that at least some states voted exactly alike in these elections.
2. We don't know which state is which because the points are unlabeled.

0.0.2 Question 7a: Jitter

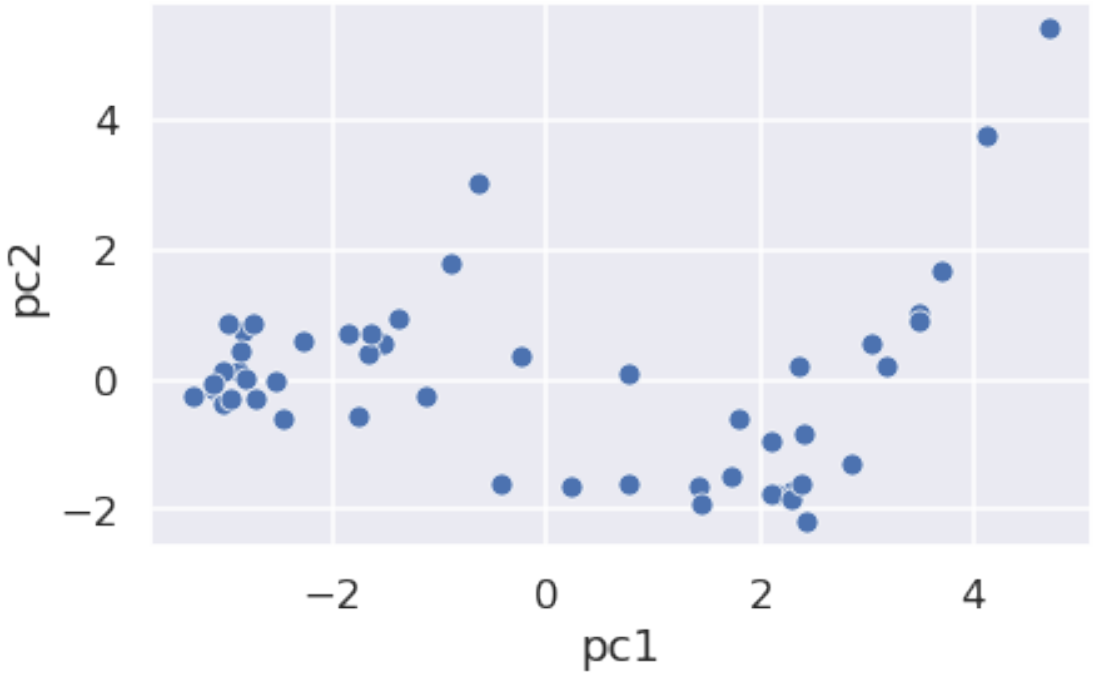
Let's start by addressing problem 1.

In the cell below, create a new dataframe `first_2_pcs_jittered` with a small amount of random noise added to each principal component. In this same cell, create a scatterplot.

To reduce overplotting, we **jitter** the first two principal components: * Add a small amount of random, unbiased Gaussian noise to each value using `np.random.normal` (documentation) with mean 0 and standard deviation less than 1. * Don't get caught up on the exact details of your noise generation; it's fine as long as your plot looks roughly the same as the original scatterplot, but without overplotting. * The amount of noise you add *should not significantly affect* the appearance of the plot; it should simply serve to separate overlapping observations.

```
In [274]: np.random.seed(42)
          # first, jitter the data
          first_2_pcs_jittered = first_2_pcs + np.random.normal(0,0.2, first_2_pcs.shape)

          # then, create a scatter plot
          sns.scatterplot(data = first_2_pcs_jittered, x = "pc1", y = "pc2");
```


4
2
0
−2
pc2
−2
0
2
4
pc1

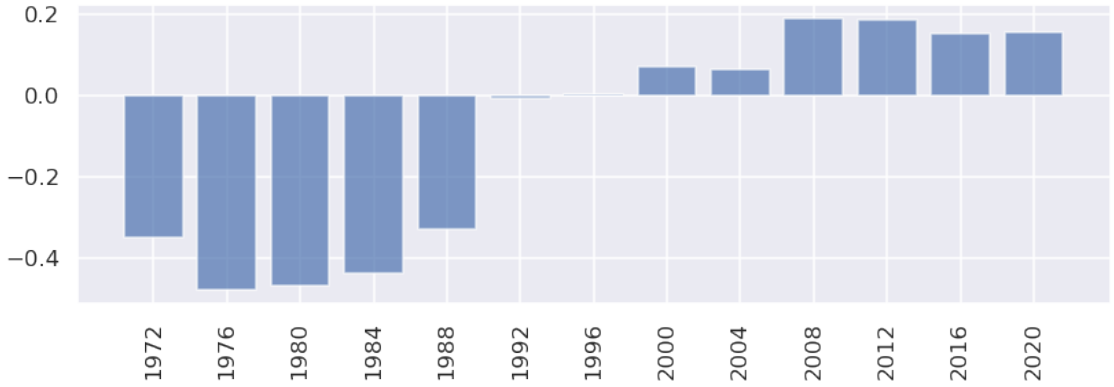
Analyze the above plot. In the below cell, address the following two points: 1. Give an example of a cluster of states that vote a similar way. Does the composition of this cluster surprise you? If you're not familiar with U.S. politics, it's fine to just say "No, I'm not surprised because I don't know anything about U.S. politics." 1. Include anything interesting that you observe. You will get credit for this as long as you write something reasonable that you can takeaway from the plot.

1) No, I'm not surprised because I don't know anything about U.S. politics.
2) For majorrity of States, when PC1 is large, PC2 always small, and vice versa. But for some certain states, PC1 and PC2 are large at same time, which may be due to some special local politic issue which complicated the voting situation of this state.

0.0.3 Question 8a

In the cell below, plot the the 2nd row of V^T, i.e., the row of V^T that correpsonds to `pc2`.

```
In [278]: plt.figure(figsize=(12, 4)) # adjusts size of plot
          plot_pc(list(df_standardized.columns), vt, 1);
```



0.0.4 Question 8b

Using the two above plots of the rows of V^T as well as the original table, give a description of what it means for a point to be in the top-right quadrant of the 2-D scatter plot from Question 7.

In other words, what is generally true about a state with relatively large positive value for `pc1` (right side of 2-D scatter plot)? For a large positive value for `pc2` (top side of 2-D scatter plot)?

Notes: * `pc2` is pretty hard to interpret, and the staff doesn't really have a consensus on what it means either. We'll be very nice when grading as long as your answer is reasonable - there is no correct answer necessarily. * Principal components beyond the first are often hard to interpret (but not always; see the lab).

For PC1: High PC1 means this state is more democratic(Blue) and low PC1 means this state is more republican(Red)

For PC2: It is likely to be something related with variance. For states with small population(like D.C), the variance could be high. And for states like CA, which has large population, the variance is genreally low, which indicated by a low PC2.

0.0.5 Question 8c

To get a better sense of whether our 2D scatterplot captures the whole story, create a **scree plot** for this data. In other words, plot the fraction of the total variance (y-axis) captured by the ith principal component (x-axis).

Hint: Be sure to label your axes appropriately! You may find `plt.xticks()` (documentation) helpful for formatting. Also check out the lab for more on scree plots.

```
In [280]: first_6_pcs = df_standardized@np.transpose(vt[0:6])
          plt.xticks([1,2,3,4,5,6])
          plt.xlabel("Principal Component")
          plt.ylabel("Fraction of the total variance")
          plt.title("Scree Plot of first 6 Principal Components")
          plt.plot([1,2,3,4,5,6], s[0:6] ** 2/sum(s[0:6] ** 2))
```

```
Out[280]: [<matplotlib.lines.Line2D at 0x7fc638162430>]
```