UNSUPERVISED MACHINE LEARNING

"The World's Billionaires"

TING CHONG NA

28.05.2023

DATASET

Topic:

“The World's Billionaires”

Objective:

To determine if any patterns exist among the richest people in the world.

Rumor has it that the ultra-wealthy community consists of either investment bankers or entrepreneurs in the tech industry that dropped out of college. Is that stereotype really true? Ever wonder if the top billionaires in the world share anything in common?

Dataset:

"The World's Billionaires" is an annual ranking documenting the net worth of the wealthiest billionaires in the world, compiled and published in March, annually, by the American business magazine - Forbes.

Dataset containing the list of 2500+ people with fortunes valued at least 1 Billion USD.

DATASET

	rank	name	networth	age	country	source	industry
0	1	Elon Musk	$219 B	50	United States	Tesla, SpaceX	Automotive
1	2	Jeff Bezos	$171 B	58	United States	Amazon	Technology
2	3	Bernard Arnault & family	$158 B	73	France	LVMH	Fashion & Retail
3	4	Bill Gates	$129 B	66	United States	Microsoft	Technology
4	5	Warren Buffett	$118 B	91	United States	Berkshire Hathaway	Finance & Investments

2600 observations

7 features

The features available from the dataset are:

1. **Rank**
2. **Name**
3. **Net Worth** - their net worth in billions USD
4. **Age**
5. **Country**
6. **Source** - their source of income
7. **Industry** - sector/industry/market segment in which each billionaire has made their fortune

DATA ANALYSIS

```
for col in df:
    print(str.format("{} has {} unique values.", col, len(df[col].unique())))
```

```
rank has 228 unique values.
name has 2598 unique values.
networth has 228 unique values.
age has 76 unique values.
country has 75 unique values.
source has 895 unique values.
industry has 18 unique values.
```

```
df[-100:-1]
```

	rank	name	networth	age	country	source	industry
2500	2448	Koo Bon-sik	$1.1 B	63	South Korea	LG	Technology
2501	2448	Suresh Krishna	$1.1 B	85	India	auto parts	Automotive
2502	2448	Nancy Lerner	$1.1 B	61	United States	banking, credit cards	Finance & Investments
2503	2448	Norma Lerner	$1.1 B	86	United States	banking	Finance & Investments
2504	2448	Randolph Lerner	$1.1 B	60	United States	banking, credit cards	Finance & Investments
...	...	...	...	...	...	...	...
2594	2578	Fu Gang	$1 B	51	China	pharma retailing	Healthcare
2595	2578	Jorge Gallardo Ballart	$1 B	80	Spain	pharmaceuticals	Healthcare
2596	2578	Nari Genomal	$1 B	82	Philippines	apparel	Fashion & Retail
2597	2578	Ramesh Genomal	$1 B	71	Philippines	apparel	Fashion & Retail
2598	2578	Sunder Genomal	$1 B	68	Philippines	garments	Fashion & Retail

228 unique *rank* values

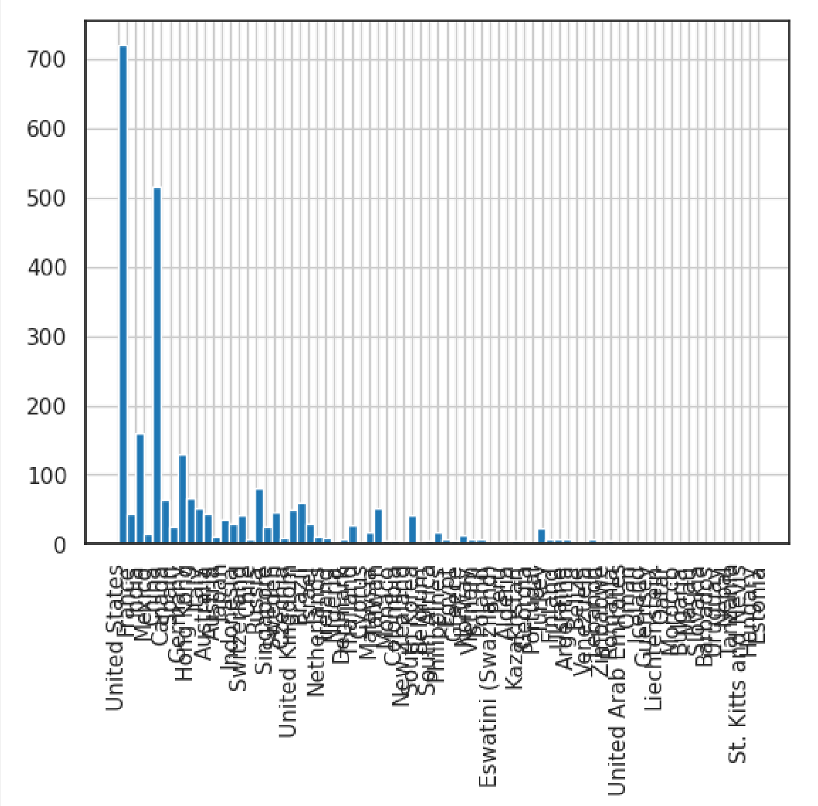
- due to many ties in the rankings

2298 unique *name* values

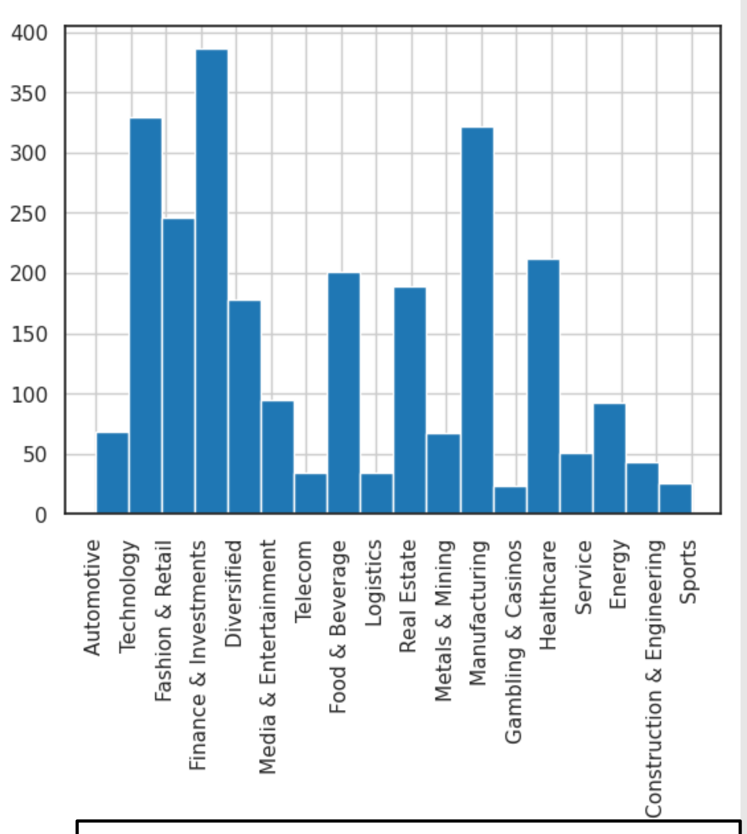
895 unique *source* values

- possibly won't help with *rank* prediction
- can be excluded

DATA ANALYSIS For Categorical variables



Histogram for *country*



Histogram for *industry*

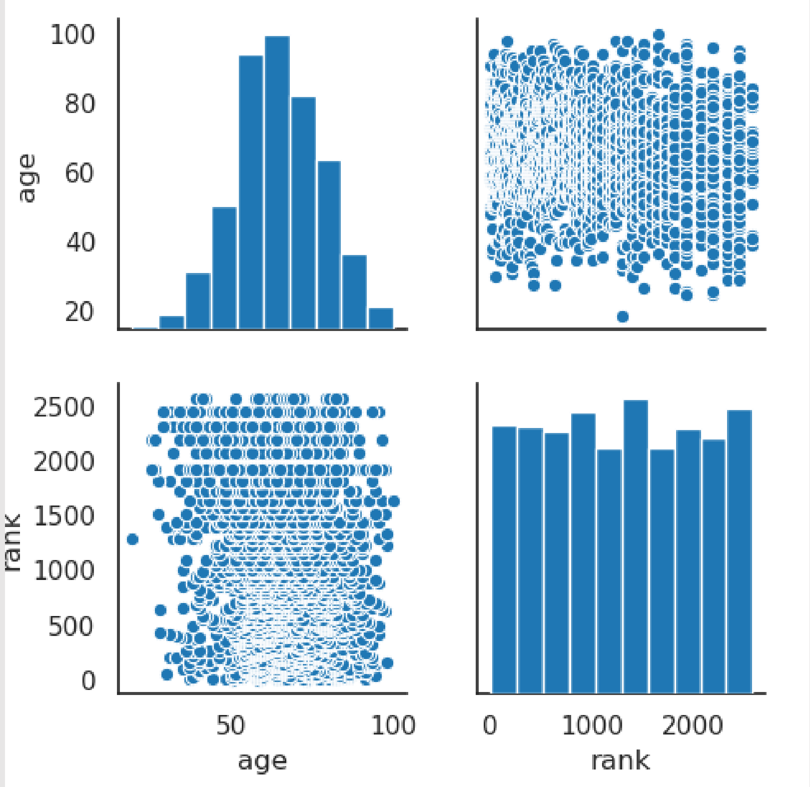
Certain *countries* contain more billionaires than others

- United State (27.6%)
- China (19.8%)
- India (6.2%)

Certain *industries* contain more billionaires than others

- Finance & Investments (14.8%)
- Technology (12.6%)
- Manufacturing (12.3%)

DATA ANALYSIS For Numerical variables



Pairwise plot (*age* and *rank*)

```
sns.pairplot(df[['age','rank']])
df[['age','rank']].corr()
```

	age	**rank**
age	1.000000	-0.124947
rank	-0.124947	1.000000

Correlation coefficient (*age* and *rank*)

2 features *age* and *rank* are Negatively Correlated

DATA ANALYSIS

```
y=df['rank']
```

- For prediction

```
df.drop(columns=['name','networth','source'],inplace=True)
```

- drop all the features not going to use

	rank	name	networth	age	country	source	industry
0	1	Elon Musk	$219 B	50	United States	Tesla, SpaceX	Automotive
1	2	Jeff Bezos	$171 B	58	United States	Amazon	Technology
2	3	Bernard Arnault & family	$158 B	73	France	LVMH	Fashion & Retail
3	4	Bill Gates	$129 B	66	United States	Microsoft	Technology
4	5	Warren Buffett	$118 B	91	United States	Berkshire Hathaway	Finance & Investments

7 features

	rank	age	country	industry
0	1	50	United States	Automotive
1	2	58	United States	Technology
2	3	73	France	Fashion & Retail
3	4	66	United States	Technology
4	5	91	United States	Finance & Investments

4 features

DATA ANALYSIS For Categorical variables

Categorical variables *country* and *industry* are not ordinal, meaning that the categories don't have a specific order.
Apply **One-Hot Encoding** to convert their levels into dummy variables.

	rank	age	country	industry
0	1	50	United States	Automotive
1	2	58	United States	Technology
2	3	73	France	Fashion & Retail
3	4	66	United States	Technology
4	5	91	United States	Finance & Investments

Before
One-Hot Encoding

	Algeria	Argentina	Australia	Austria	Barbados	Belgium	Belize	Brazil	Bulgaria	Canada	...	Manufacturing	Media & Entertainment	Metals & Mining	Real Estate	Service	Sports	Technology	Telecom	rank	age
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	50.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	2.0	58.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	73.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	4.0	66.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	91.0

After
One-Hot Encoding

Model: PCA vs KernelPCA

```
pca = PCA()
score_pca = pca.fit_transform(new_data)
score_pca
```

```
array([[-1.26853519e+03,  1.71468528e+01,  5.72335490e-01, ...,
        -7.10658790e-04,  2.44011905e-12,  4.47187292e-15],
       [-1.26755335e+03,  9.15083027e+00,  6.45685318e-01, ...,
        -5.53068462e-04, -6.76568413e-13, -3.00082148e-14],
       [-1.26658732e+03, -5.85542849e+00, -3.26296832e-01, ...,
         1.09596031e-03, -1.71666647e-12,  8.55564897e-14],
       ...,
       [ 1.30841059e+03, -9.70016782e+00, -5.99273940e-02, ...,
         2.59761008e-04, -4.84678970e-14,  1.35702407e-16],
       [ 1.30841740e+03, -6.70034037e+00, -4.51232531e-02, ...,
         2.08085059e-04, -4.89946773e-14,  1.60838454e-16],
       [ 1.30841512e+03, -7.69891138e+00, -5.97573898e-02, ...,
         6.38007889e-04,  2.76855261e-14, -6.31915859e-17]])
```

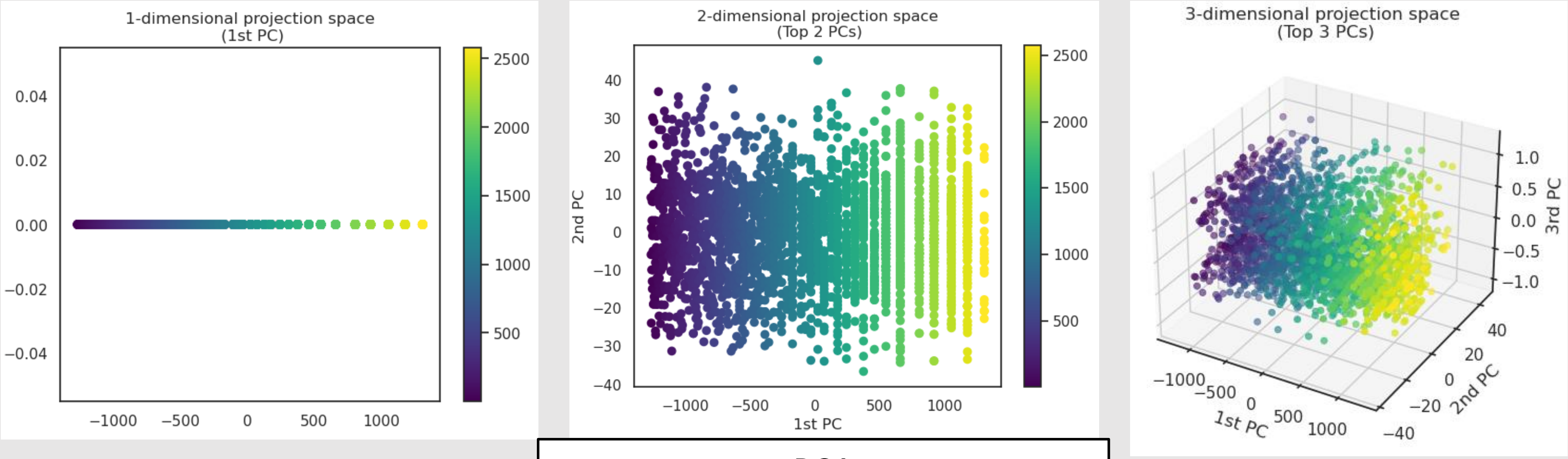
PCA

```
kernel_pca = KernelPCA(kernel="rbf" ,fit_inverse_transform=True, alpha=0.1)
kernel_score=kernel_pca.fit_transform(new_data)
kernel_score
```

```
array([[-3.94548027e-03, -1.36187246e-02, -1.12854093e-03, ...,
        -7.74713991e-14,  1.09351403e-13,  1.09461082e-14],
       [-3.99953835e-03, -1.38121977e-02, -1.14594935e-03, ...,
        -1.25805551e-14,  3.84769762e-14,  5.49070056e-14],
       [-4.01033364e-03, -1.38512661e-02, -1.14955766e-03, ...,
        -3.33272479e-14,  6.11371718e-14,  4.08517499e-14],
       ...,
       [-4.57890943e-03, -1.58931839e-02, -1.33482432e-03, ...,
        -3.56146092e-14,  6.36353282e-14,  3.93021976e-14],
       [-4.61216337e-03, -1.60127204e-02, -1.34569474e-03, ...,
         7.01019473e-14, -5.18301116e-14,  1.10921725e-13],
       [-4.60475783e-03, -1.59861261e-02, -1.34328190e-03, ...,
         1.77731170e-14,  5.32434152e-15,  7.54706448e-14]])
```

KernelPCA

Model: PCA

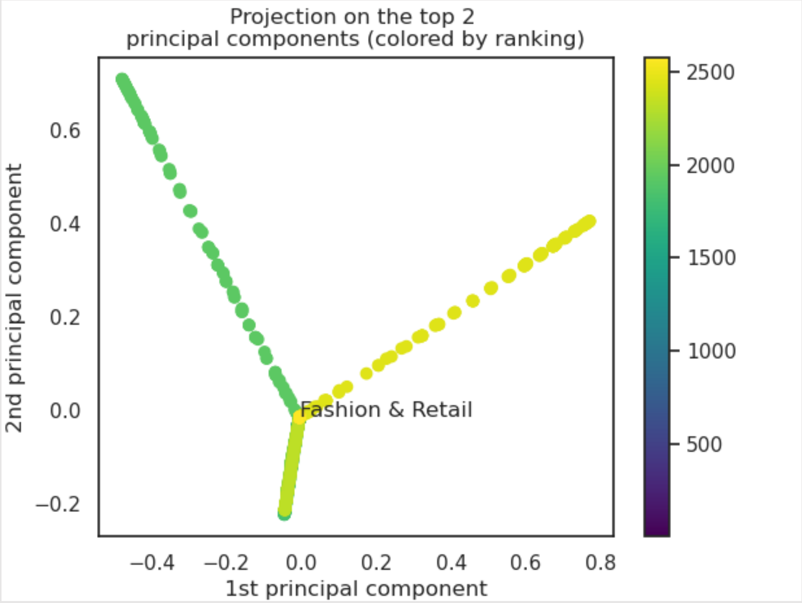


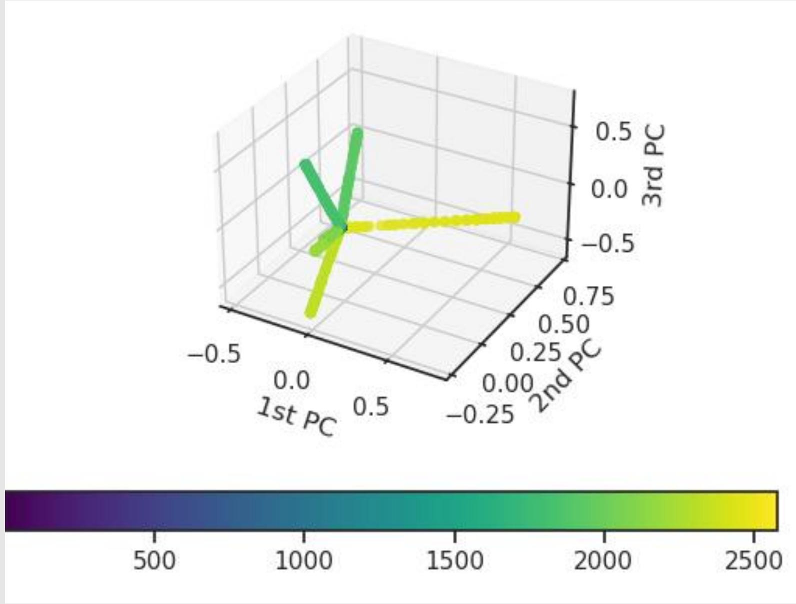
PCA

Apply PCA and plot the projection space in 1, 2, and 3 dimensions respectively.

In the 2-dimensional and 3-dimensional projection space, we see a similar trend that rankings change across the x-axis, meaning that most of its variation occurs in the projection on the first principal component.

Model: KernelPCA

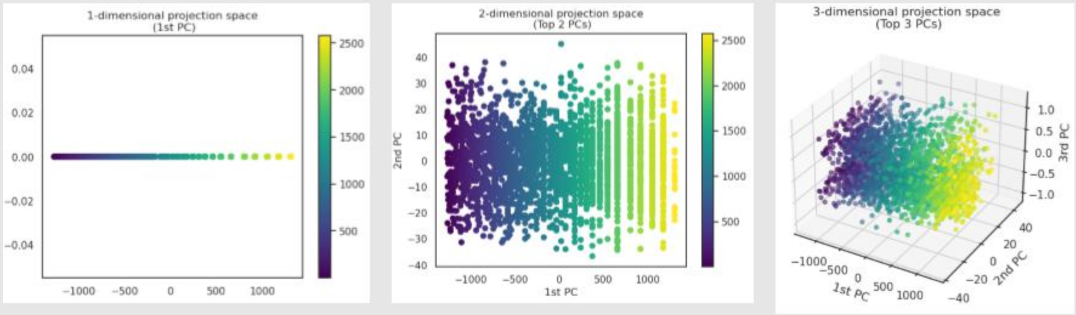




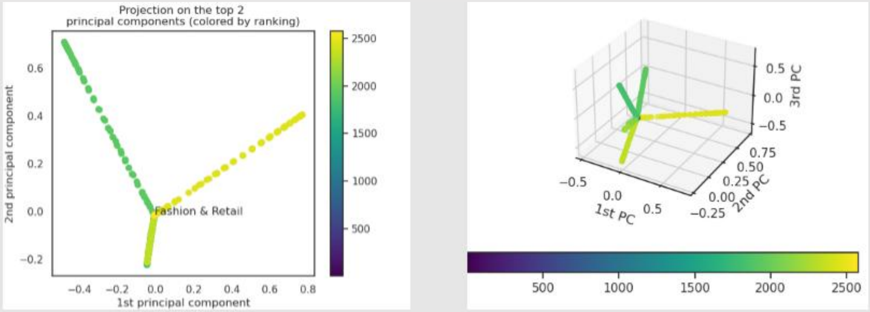
KernelPCA

A Bifurcation of the data dependent on the *rank*.
The diverging branches suggest that difference in ranking can be associated with certain patterns in the data.
This becomes more apparent in the 3-dimensional projection space.
Each line corresponds to a different range of ranking (represented by the varying colors).

Model: PCA vs KernelPCA



PCA



KernelPCA

From this example, we can observe that surely, some combination of the features *country*, *industry* and *age* leads to consistent change in rankings.

It is important to note that due to the nature of PCA in general, unable to decrypt what that combination is.

Difference in ranking can be associated with certain patterns in the data.

KernelPCA will improve Visualization, compare to PCA.

Model: PCA vs KernelPCA

Add Ridge.

```
from sklearn.linear_model import Ridge
```

```
X_train, X_test, y_train, y_test = train_test_split(kernel_score, y, test_size=0.4, random_state=0)
lr = Ridge(alpha=0).fit(X_train, y_train)
print(str.format("Test set R^2 score for Kernel PCA: {}", lr.score(X_test, y_test)))
```

```
Test set R^2 score for Kernel PCA: 0.9885118791781703
```

```
X_train, X_test, y_train, y_test = train_test_split(score_pca, y, test_size=0.40, random_state=0)
lr= Ridge(alpha=0).fit(X_train, y_train)
print(str.format("Test set R^2 score for PCA: {}", lr.score(X_test, y_test)))
```

```
Test set R^2 score for PCA: 0.6785924170486036
```

KernelPCA

R^2: 0.989

PCA

R^2: 0.679

Kernel PCA generally performs better given a higher R^2 (coefficient of determination) score on the test set, compare to PCA.

KernelPCA will improve Prediction, compare to PCA.

CONCLUSION

In conclusion, **KernelPCA** seems to be the **best method** to determine if any patterns exist among the richest people in the world.

Improve Visualization

Improve Prediction

POSSIBLE FLAW IN MODEL

Computational Cost

- PCA generally has lower memory and runtime requirements than KernelPCA, and can be scaled to massive datasets

Inverse Mapping

- Unlike PCA, KernelPCA may not give perfect reconstruction
- KernelPCA spans a subspace of the original data, so applying inverse transformation on the data after KernelPCA will not return the original data.