

An online community for showcasing R & Python tutorials

- [About Us](#)
- [Archives](#)
- [Contribute](#)
- [Sign Up](#)
- [Log In](#)

- [Advanced Modeling in R](#)

Principal Component Analysis – Unsupervised Learning

- Published on October 9, 2017 at 10:00 am
- Updated on February 15, 2018 at 12:01 pm

9,284 reads

0 shares

[2](#) comments

7 min read

[Facebook 0](#) [Twitter](#) [LinkedIn](#) [Reddit 0](#) [Pinterest 0](#) [Email this](#) [Print](#)

Unsupervised learning is a machine learning technique in which the dataset has no target variable or no response value- Y . The data is unlabelled. Simply saying, there is no target value to **supervise** the learning process of a learner unlike in supervised learning where we have training examples which have both input variables X_i and target variable- Y i.e. (x_i, y_i) vectors and by looking and learning from the training examples the learner generates a mapping function (also called a **hypothesis**) $f: X_i \rightarrow Y$ which maps X_i values to Y and learns the relationship between input variables and target variable so that we could generalize it to some random unseen test examples and predict the target value.

The best example of unsupervised learning is when a small child is given some unlabelled pictures of cats and dogs, so only by looking at the structural and visual similarities and dissimilarities between the images, the child classifies one as a dog and other as cat.

Unsupervised learning is inclined towards finding groups and subgroups from data by finding the associations, similarities and relationships between the inputs X_i . It is important for understanding the variations and grouping structure of a dataset and is also used as a pre-processing tool for finding the best and most important features X_i which explain the most variance and summarize the most information in the data using techniques such as principal component analysis (PCA) for supervised learning techniques.

example-If we have a dataset with 100 predictors and we wanted to generate a model, it would be highly inefficient to use all those 100 predictors because that would increase the variance and complexity of the model and which in turn would lead to overfitting. Instead, what PCA does is find 10 most correlated variables and linearly combine them to generate principal components $-Z_m$ which could be further used as features for our model.

Principal Component Analysis

PCA introduces a lower-dimensional representation of the dataset. It finds a sequence of linear combination of the variables called the principal components- Z_1, Z_2, \dots, Z_m that explain the maximum variance and summarize the most information in the data and are mutually uncorrelated.

What we try to do is find most relevant set of variables and simply linearly combine the set of variables into a single variable- Z_m called a principal component.

- 1) The first principal component PC_1 has the highest variance across data.
- 2) The second principal component PC_2 is uncorrelated with PC_1 which also has high variance.

We have tons of correlated variables in a high dimensional dataset and what PCA tries to do is pair and combine them to a set of some important variables that summarize all information in the data.

PCA will give us new set of variables called *principal components* which could further be used as inputs in a supervised learning model. So now we have lesser and most important set of variables paired together to form a new single variable which explains most variance in data. This technique is often termed as **dimensionality reduction** which is famous technique to do feature selection/reduction and use only relevant features X_i in the model.

Details

We have a set of input vectors $x_1, x_2, x_3, \dots, x_p$ with n observations in dataset.

The 1st principal component Z_1 of a set of features is the *normalized linear combination* of the features x_1, x_2, \dots, x_p .

$$Z_1 = \sum_{i=1}^p \phi_1 x_1 + \phi_2 x_2 + \phi_3 x_3 + \dots \phi_i x_i$$

where n =no of observations, p = number of variables. It is a linear combination to find out the highest variance across data. By normalized it means $\sum_{j=1}^p \phi_j^2 = 1$.

We refer to the weights $\phi_{n \times p}$ as **Loading** matrix. The loadings make up the principal components loading vector. $\phi_1 = (\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \dots, \phi_{n1})^T$ is the loading vector for PC_1 .

We constrain the loadings so that their sum of squares could be 1, as otherwise setting these elements to be arbitrarily large in absolute value could result in an arbitrarily large variance

The first Principal component solves the below optimization problem of maximizing variance across the components-

$$\text{maximize} : \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^p (\phi_{ji} \cdot X_{ij})^2 \text{ subject to } \sum_{j=1}^p \phi_{ji}^2 = 1$$

Here each principal component has mean 0.

The above problem can be solved via Single value decomposition of matrix X , which is a standard technique in linear algebra.

Enough maths now let's start implementing PCA in R.

Implementing PCA in R

We will use USArrests data.

```
?USArrests
#dataset which contains Violent Crime Rates by US State
dim(USArrests)
dimnames(USArrests)
## [1] 50 4

## [[1]]
## [1] "Alabama" "Alaska" "Arizona" "Arkansas"
## [5] "California" "Colorado" "Connecticut" "Delaware"
## [9] "Florida" "Georgia" "Hawaii" "Idaho"
## [13] "Illinois" "Indiana" "Iowa" "Kansas"
## [17] "Kentucky" "Louisiana" "Maine" "Maryland"
## [21] "Massachusetts" "Michigan" "Minnesota" "Mississippi"
## [25] "Missouri" "Montana" "Nebraska" "Nevada"
## [29] "New Hampshire" "New Jersey" "New Mexico" "New York"
## [33] "North Carolina" "North Dakota" "Ohio" "Oklahoma"
## [37] "Oregon" "Pennsylvania" "Rhode Island" "South Carolina"
## [41] "South Dakota" "Tennessee" "Texas" "Utah"
## [45] "Vermont" "Virginia" "Washington" "West Virginia"
## [49] "Wisconsin" "Wyoming"
##
## [[2]]
## [1] "Murder" "Assault" "UrbanPop" "Rape"
```

Copy

Finding means for all variables.

```
#finding mean of all
apply(USArrests,2,mean)
## Murder Assault UrbanPop Rape
## 7.788 170.760 65.540 21.232
```

Copy

Finding variance of all variables.

```
apply(USArrests,2,var)
## Murder Assault UrbanPop Rape
## 18.97047 6945.16571 209.51878 87.72916
```

Copy

There is a lot of difference in variances of each variables. In PCA mean does not play a major role, but variance plays a major role in defining principal components so very large differences in variance value of a variable will definitely dominate the principal components. We need to **standardize** the variables so as to get mean $\mu = 0$ and variance $\sigma^2 = 1$. To standardize we use formula $x' = \frac{x - \text{mean}(x)}{\text{sd}(x)}$.

The function `prcomp()` will do the needful of standardizing the variables.

```
pca.out<-prcomp(USArrests,scale=TRUE)
pca.out
## Standard deviations (1, ..., p=4):
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
##
## Rotation (n x k) = (4 x 4):
## PC1 PC2 PC3 PC4
## Murder -0.5358995 0.4181809 -0.3412327 0.64922780
## Assault -0.5831836 0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158 0.13387773
## Rape -0.5434321 -0.1673186 0.8177779 0.08902432
```

Copy

```
#summary of the PCA
summary(pca.out)
## Importance of components$:
## PC1 PC2 PC3 PC4
## Standard deviation 1.5749 0.9949 0.59713 0.41645
## Proportion of Variance 0.6201 0.2474 0.08914 0.04336
## Cumulative Proportion 0.6201 0.8675 0.95664 1.00000
```

Copy

```
names(pca.out)
## [1] "sdev" "rotation" "center" "scale" "x"
```

Copy

Now as we can see maximum % of variance is explained by PC_1 , and all PCs are mutually uncorrelated. Around 62 % of variance is explained by PC_1 .

Let's build a biplot to understand better.

```
biplot(pca.out,scale = 0, cex=0.65)
Copy
```

Gives this plot:

Now in the above plot red colored arrows represent the variables and each direction represent the direction which explains the most variation. Example- for all the countries in the direction of **‘UrbanPop’** are countries with most urban-population and opposite to that direction are the countries with least. So this is how we interpret our Bi-plot.

Conclusion

PCA is a great pre-processing tool for picking out the most relevant linear combination of variables and use them in our predictive model. It helps us find out the variables which explain the most variation in the data and only use them. PCA plays a major role in the data analysis process before going for advanced analytics and model building.

The only drawback PCA has is that it generates the principal components in an unsupervised manner i.e without looking the target vector $(y_1, y_2, y_3 \dots y_n)$, hence the principal components which explain the most variation in dataset without looking at the target- Y variable, may or may not explain good percentage of variance for the response variable Y which could affect and degrade the performance of the predictive model.

Credits

- The R code for implementing PCA in R is adapted from the amazing online course “Statistical learning” offered by Stanford University Online. I urge readers to definitely go and try out this course to get clear with the core statistics and maths behind various statistical models. The details about the course can be found here- [Statistical Learning](#)
- Also, this book [Elements Of statistical learning](#) has helped me learn lots of amazing stuff

Hope you guys liked the article, make sure to like and share it. Happy machine learning!!

Author



[Anish Singh Walia](#)

Bachelor's in Engineering, Information Technology, VIT Vellore.

More from Author

- [Kaggle data science survey data analysis using Highcharter](#)
- [Making a Shiny dashboard using 'highcharter' – Analyzing Inflation Rates](#)
- [Text Message Classification](#)

Disclosure

- Anish Singh Walia does not work or receive funding from any company or organization that would benefit from this article. Views expressed here are personal and not supported by university or company.

Tags

[Linear Regression](#)[Principal Component Analysis](#)[Unsupervised Learning](#)

Share it

[Facebook](#) [Twitter](#) [Reddit](#) [LinkedIn](#) [Email this](#)

Related Posts


- [K-Nearest Neighbors \(KNN\) with Python](#)
- [Image Recognition with Keras: Convolutional Neural Networks](#)
- [Predicting Irish electricity consumption with an LSTM neural network](#)
- [Visualizing New York City WiFi Access with K-Means Clustering](#)
- [Kalman Filter: Modelling Time Series Shocks with KFAS in R](#)

★ Related Courses

- [Fundamentals of Bayesian Data Analysis in R](#)
- [Supervised Learning in R: Case Studies](#)
- [Survival Analysis in R](#)
- [Machine Learning Toolbox](#)
- [Machine Learning with Tree-Based Models in R](#)

Affiliated with DataCamp.com


🔔 Discussion

Join the discussion...

LOG IN WITH


OR SIGN UP WITH DISQUS ?

Name

- 


Obayed Moni • a year ago

Its very handy & resourceful for beginner. Thanks for uploading.

1 ^ | v • Reply • Share ›
- 

Passarete 10 • a year ago

How can I select all the variables inside the PC1?


^ | v • Reply • Share ›
- 

Nikhil Singh • 2 years ago

Is there a down-vote button for this article?

This post is plagiarized from Stanford Statistical Learning course. Even the code is copied badly and not at all explained.

Useless plagiarized article.

^ | v • Reply • Share ›
- 

Srikanth K S • 2 years ago

This is a ripoff from Introduction to Statistical Learning (www-bcf.usc.edu/~gareth/ISL/) from page 374-379. Same equations(with a mistake in the optimization problem equation) and same example!

@Anish Singh Walia: What is original in this article?


Datascience+ admin: Are you watching this?

^ | v • Reply • Share ›

ALSO ON DATASCIENCE+ HUB


Linear Regression with Python

1 comment • a month ago

 Aman Lodha — Really good article. It helped me a lot in understanding the linear regression model.


Machine Learning in Excel With Python

1 comment • 6 months ago

 Ben Bartling — Hello, could a sci kit learn pickle of a model ever be opened in excel? I am training a model in Python with ...


Leaf Plant Classification: Statistical Learning Model – Part 2

2 comments • 4 months ago

 giorgiog — Hi,I may suggests the ...

How to combine Multiple ggplot Plots to make Publication-ready Plots

2 comments • 4 months ago

 Thomas John Flaherty — Wow yeah - anything from Thomas Pedersen is legit. Thanks for this.

DataScience+Bridging the gap between talent and opportunity

>_ Site Links

- About Us
- Archives
- Contribute
- R Markdown

>_ Legal

- Privacy Policy
- Terms of Service
- Disclosure
- Contact Us

>_ Articles

- Introduction
- Getting Data
- Data Management
- Visualizing Data
- Basic Statistics

- [Regression Models](#)
- [Advanced Modeling](#)
- [Programming](#)



Connect with Us

© 2019 DataSciencePlus.com

