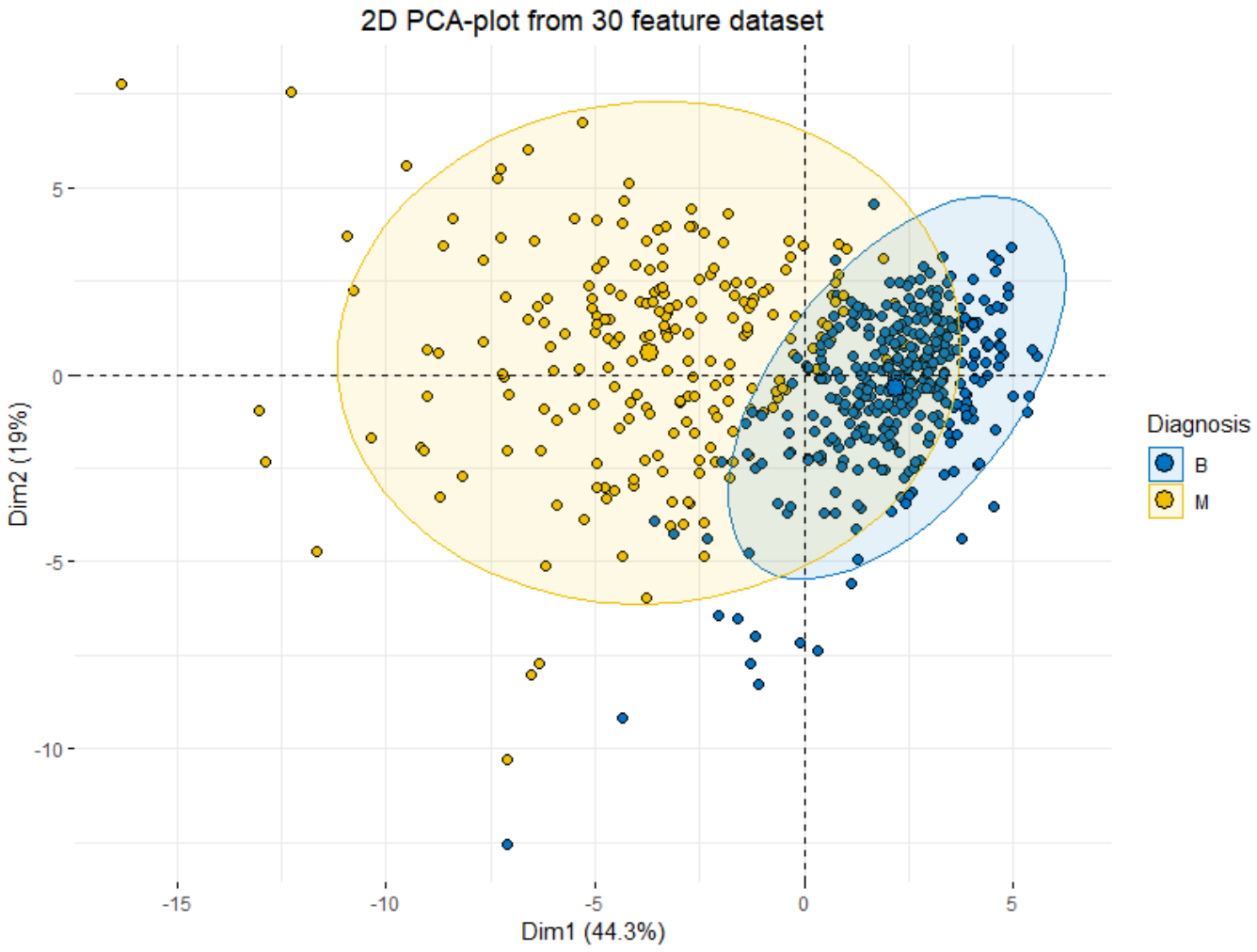
Principal Component Analysis (PCA) 101, using R

Peter Nistrup Follow
Jan 29, 2019 · 8 min read ★

Improving predictability and classification one dimension at a time! “Visualize” 30 dimensions using a 2D-plot!



Basic 2D PCA-plot showing clustering of ”Benign” and ”Malignant” tumors across 30 features.

Make sure to **follow my profile** if you enjoy this article and want to see more!

. . .

Setup

For this article we'll be using the Breast Cancer Wisconsin data set from the *UCI Machine learning repo* as our data. Go ahead and load it for yourself if you want to follow along:

```
wdbc <- read.csv("wdbc.csv", header = F)

features <- c("radius", "texture", "perimeter", "area", "smoothness",
"compactness", "concavity", "concave_points", "symmetry",
"fractal_dimension")

names(wdbc) <- c("id", "diagnosis", paste0(features,"_mean"),
paste0(features,"_se"), paste0(features,"_worst"))
```

The code above will simply load the data and name all 32 variables. The **ID**, **diagnosis** and ten distinct (30) features. From UCI:

*"The **mean, standard error**, and "**worst**" or largest (mean of the three largest values) of these features were computed for each image, resulting in **30 features**. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius."*

. . .

Why PCA?

Right, so now we've loaded our data and find ourselves with 30 variables (thus excluding our response "diagnosis" and the irrelevant ID-variable).

Now some of you might be saying "30 variable is a lot" and some might say "Pfft.. Only 30? I've worked with THOUSANDS!!" but rest assured that this is equally applicable in either scenario..!

There's a few pretty good reasons to use PCA. The plot at the very beginning af the article is a great example of how one would plot multi-dimensional data by using PCA, we actually capture **63.3%** (Dim1 44.3% + Dim2 19%) of variance in the entire dataset by just using those **two principal components**, pretty good when taking into consideration that the original data consisted of **30 features** which would be impossible to plot in any meaningful way.

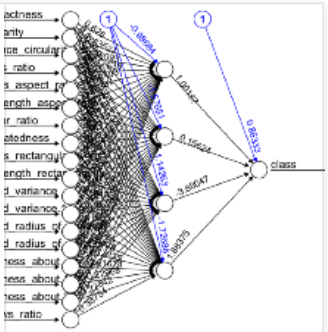
A very powerful consideration is to acknowledge that **we never specified a response variable** or anything else in our PCA-plot indicating whether a tumor was "*benign*" or "*malignant*". It simply turns out that when we try to describe variance in the data using the linear combinations of the PCA we find some pretty obvious clustering and separation between the "*benign*" and "*malignant*" tumors! This makes a great case for developing a classification model based on our features!

Another major "feature" (no pun intended) of PCA is that it can actually directly improve performance of your models, please take a look at this great article to read more:

Dimensionality Reduction — Does PCA really improve classification outcome?

Introduction

towardsdatascience.com



• • •

What is PCA and how does it work?

Lets get something out the way immediately, PCAs primary purpose is **NOT** as a ways of feature removal! PCA can reduce dimensionality but **it wont reduce the number of features / variables in your data.** What this means is that you might discover that you can explain 99% of variance in your 1000 feature dataset by just using 3 principal components but you still need those 1000 features to construct those 3 principal components, this also means that in the case of predicting on future data you still need those same 1000 features on your new observations to construct the corresponding principal components.

Right, right enough of that, how does it work?

Since this is purely introductory I'll skip the math and give you a quick rundown of the workings of PCA:

- **Standardize the data** (Center and scale).
- **Calculate the Eigenvectors and Eigenvalues from the covariance matrix or correlation matrix** (One could also use Singular Vector Decomposition).
- **Sort the Eigenvalues in descending order and choose the *K* largest Eigenvectors** (Where *K* is the desired number of dimensions of the new feature subspace $k \leq d$).
- **Construct the projection matrix W from the selected *K* Eigenvectors.**
- **Transform the original dataset X via W to obtain a *K*-dimensional feature subspace Y.**

This might sound a bit complicated if you haven't had a few courses in algebra, but the gist of it is to transform our data from it's initial state **X** to a subspace **Y** with ***K*** dimensions where ***K*** is — *more often than not* — less than the original dimensions of **X.** Thankfully this is easily done using R!

. . .

PCA on our tumor data

So now we understand a bit about how PCA works and that should be enough for now. Lets actually try it out:

```
wdbc.pr <- prcomp(wdbc[c(3:32)], center = TRUE, scale = TRUE)
summary(wdbc.pr)
```

This is pretty self-explanatory, the '*prcomp*' function runs PCA on the data we supply it, in our case that's '*wdbc[c(3:32)]*' which is our data excluding the ID and diagnosis variables, then we tell R to center and scale our data (thus **standardizing** the data). Finally we call for a summary:

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7     PC8    PC9    PC10
Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172 0.69037 0.6457 0.59219
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251 0.01589 0.0139 0.01169
Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 0.91010 0.92598 0.9399 0.95157
```

The values of the first 10 principal components

Recall that a property of PCA is that our components are sorted from largest to smallest with regard to their standard deviation (**Eigenvalues**). So let's make sense of these:

- ***Standard deviation:*** This is simply the **eigenvalues** in our case since the data has been centered and scaled (**standardized**)
- ***Proportion of Variance***: This is the amount of variance the component accounts for in the data, ie. **PC1** accounts for **>44% of total variance** in the data alone!
- ***Cumulative Proportion***: This is simply the accumulated amount of explained variance, ie. if we used **the first 10 components** we would be able to account for **>95% of total variance** in the data.

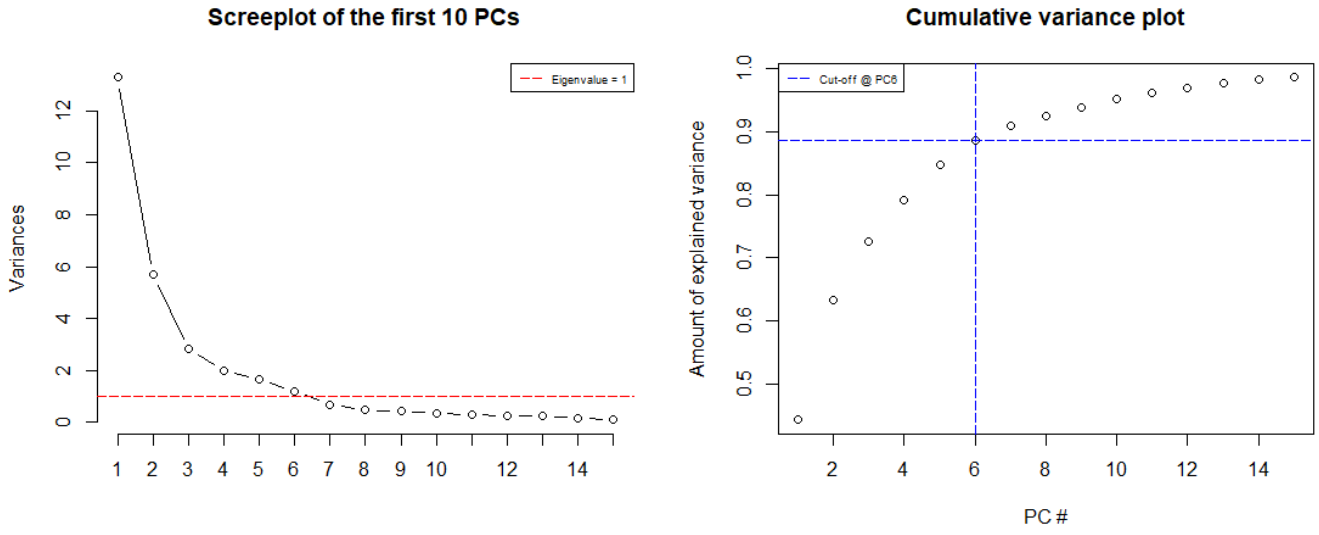
Right, so how many components do we want? We obviously want to be able to explain as much variance as possible but to do that we would need all 30 components, at the same time we want to reduce the number of dimensions so we definitely want less than 30!

Since we **standardized** our data and we now have the corresponding eigenvalues of each PC we can actually use these to draw a boundary for us. Since an **eigenvalues <1** would mean that the component actually explains less than a single explanatory variable

we would like to discard those. If our data is well suited for **PCA** we should be able to discard these components while retaining at least **70–80% of cumulative variance**. Lets plot and see:

```
screeplot(wdbc.pr, type = "l", npcs = 15, main = "Screeplot of the
first 10 PCs")
abline(h = 1, col="red", lty=5)
legend("topright", legend=c("Eigenvalue = 1"),
       col=c("red"), lty=5, cex=0.6)

cumpro <- cumsum(wdbc.pr$sdev^2 / sum(wdbc.pr$sdev^2))
plot(cumpro[0:15], xlab = "PC #", ylab = "Amount of explained
variance", main = "Cumulative variance plot")
abline(v = 6, col="blue", lty=5)
abline(h = 0.88759, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC6"),
       col=c("blue"), lty=5, cex=0.6)
```

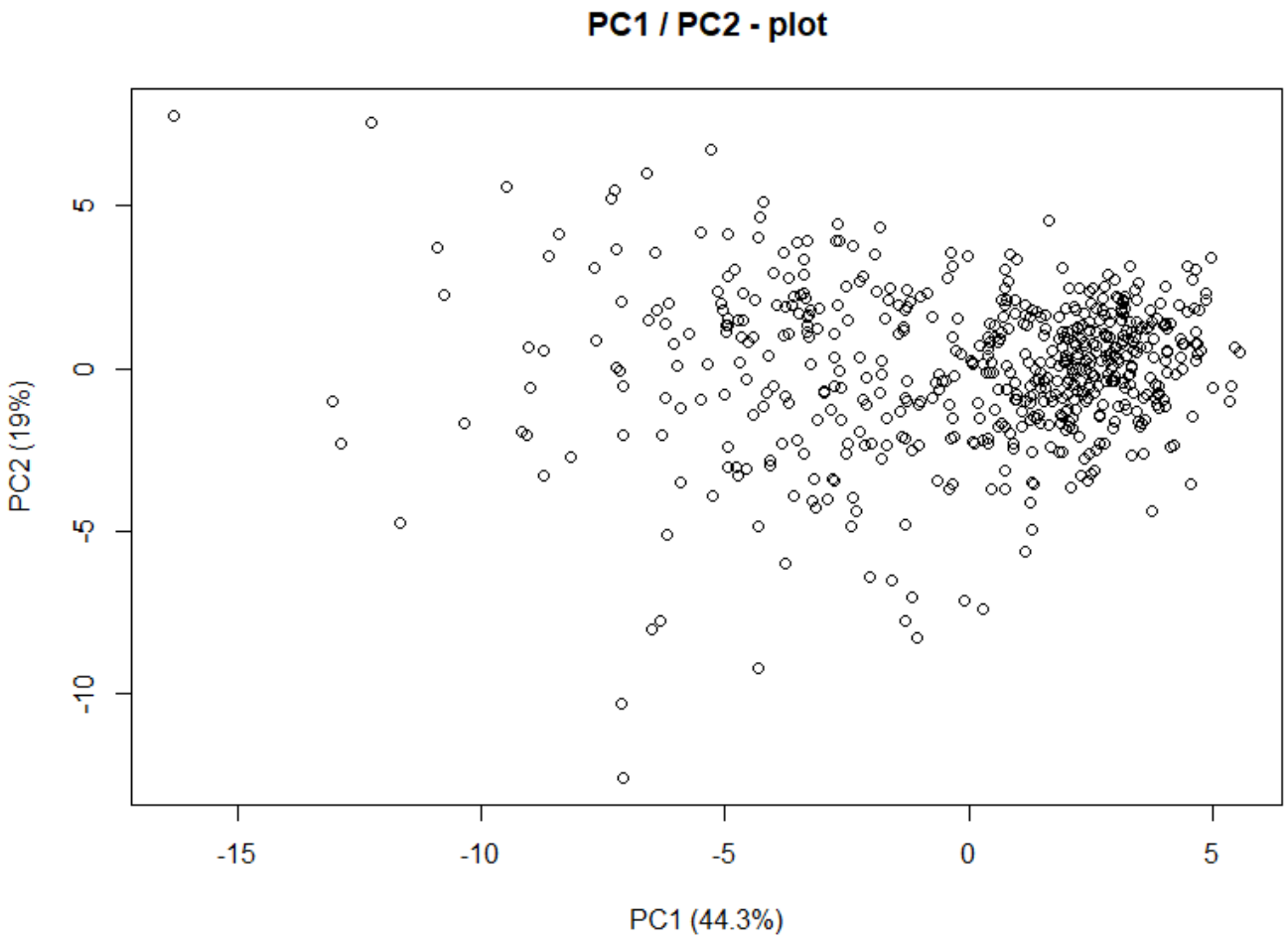


Screeplot of the Eigenvalues of the first 15 PCs (left) & **Cumulative variance plot** (right)

We notice is that **the first 6 components** has an **Eigenvalue >1** and explains almost **90% of variance**, this is great! We can effectively **reduce dimensionality from 30 to 6** while only "loosing" about 10% of variance!

We also notice that we can actually explain more than 60% of variance with just the first two components. Let's try plotting these:

```
plot(wdbc.pr$x[,1],wdbc.pr$x[,2], xlab="PC1 (44.3%)", ylab = "PC2
(19%)", main = "PC1 / PC2 - plot")
```



Alright, this isn't really too telling but consider for a moment that this is representing **60%+ of variance in a 30 dimensional dataset.** But what do we see from this? There's some **clustering** going on in the **upper/middle-right.** Lets also consider for a moment what the goal of this analysis actually is. We want to explain difference between **malignant** and **benign** tumors. Let's actually add the **response variable** (*diagnosis*) to the plot and see if we can make better sense of it:

```
library("factoextra")
fviz_pca_ind(wdbc.pr, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = wdbc$diagnosis,
             col.ind = "black",
             palette = "jco",
```

```
             addEllipses = TRUE,
             label = "var",
             col.var = "black",
             repel = TRUE,
             legend.title = "Diagnosis") +
  ggtitle("2D PCA-plot from 30 feature dataset") +
  theme(plot.title = element_text(hjust = 0.5))
```

This is essentially the exact same plot with some fancy ellipses and colors corresponding to the diagnosis of the subject and now we see **the beauty of PCA**. With just the first two components we can clearly see some separation between the **benign** and **malignant** tumors. This is a clear indication that the data is well-suited for some kind of **classification model** (like **discriminant analysis**).

. . .

What's next?

Our next immediate goal is to construct some kind of model using the first 6 principal components to predict whether a tumor is benign or malignant and then compare it to a model using the original 30 variables.

We'll take a look at this in the next article:

Linear Discriminant Analysis (LDA) 101, using R

Decision boundaries, separations, classification and more. Let's dive into LDA!

towardsdatascience.com

. . .

If you want to see and learn more, be sure to **follow my profile** and visit my ***splashpage*** to get in contact!

Peter Nistrup — Medium

Read writing from Peter Nistrup on Medium. https://nistrup.github.io/ — DATA SCIENCE, STATISTICS & AI ... Stay up to...

medium.com

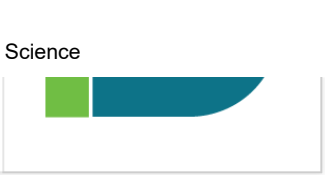
. . .

Additional resources:

Making sense of principal component analysis, eigenvectors & eigenvalues

begingroup$ Imagine a big family dinner, where everybody starts asking you about PCA. First you explain it to your...

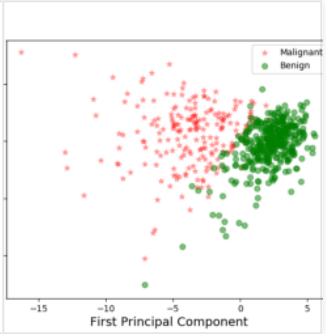
stats.stackexchange.com



Understanding PCA (Principal Component Analysis) with Python

Getting stuck in the sea of variables to analyze your data ? Feeling lost in deciding which features to choose so that...

towardsdatascience.com



A One-Stop Shop for Principal Component Analysis

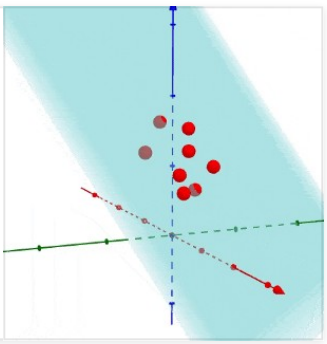
At the beginning of the textbook I used for my graduate stat theory class, the authors (George Casella and Roger...

towardsdatascience.com

Dimensionality Reduction For Dummies — Part 1: Intuition

Dimensionality Reduction with PCA and SVD. Explained in a simple, visual, and intuitive way. From the big picture to...

towardsdatascience.com



Predicting breast cancer using PCA + LDA in R | Kaggle

Edit description

www.kaggle.com

How to project a new vector onto PCA space?

Thanks for contributing an answer to Cross Validated! Please be sure to answer the question. Provide details and share...

stats.stackexchange.com

PCA - Principal Component Analysis Essentials

Statistical tools for data analysis and visualization

www.sthda.com

Data Science | Statistics | Machine Learning | Analytics