Interpretation with LIME of classification in high-dimensional tabular data

2020

Introduction

TODO

Data Description

The dataset was uploaded to Kaggle by/thanks to UCI Machine Learning Repository:
https://www.kaggle.com/murats/gene-expression-cancer-rnaseq

Source: Samuele Fiorini, samuele.fiorini@dibris.unige.it, University of Genoa, redistributed under Creative Commons license.

The data consists of 801 patients.

The 20532 independent variables are all RNA sequencing gene expression levels measured by a sequencing platform (Illumina HiSeq).

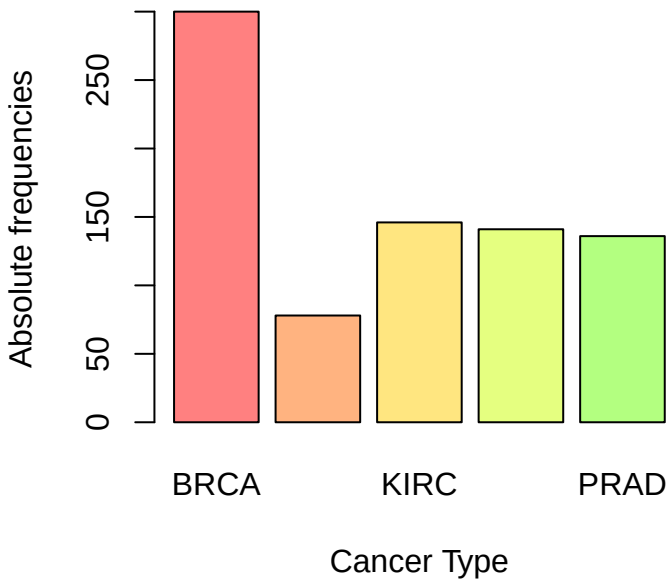
The dependent categorical variable represent primary tumors occurring in different parts of the body, covering 5 tumor types including:

- lung adenocarcinoma (LUAD)
- breast carcinoma (BRCA)
- kidney renal clear-cell carcinoma (KIRC)
- colon adenocarcinoma (COAD)
- prostate adenocarcinoma (PRAD)

A machine learning model will be trained to predict the type of tumor for new patients.
TODO: buscar un dataset similar pero que contenga tambien pacientes sanos.

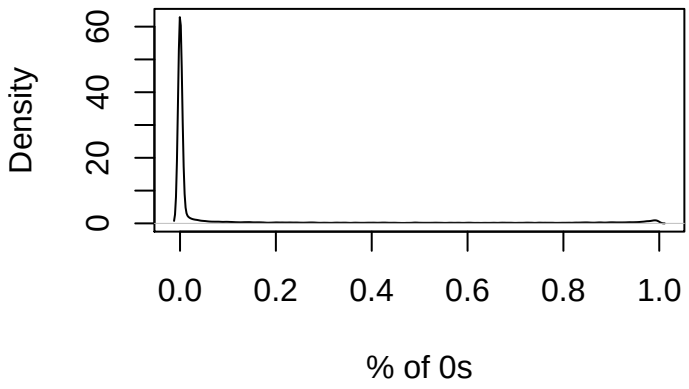
Data Analysis and Preprocessing

Absolute frequencies of the response:

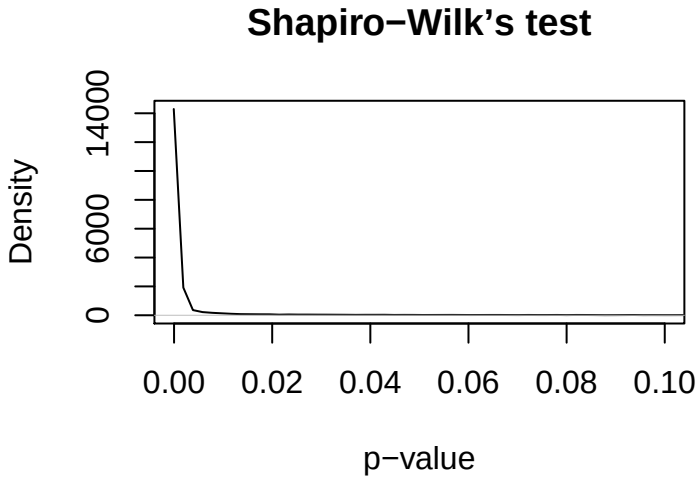


268 gene expression variables contain only 0s are removed.

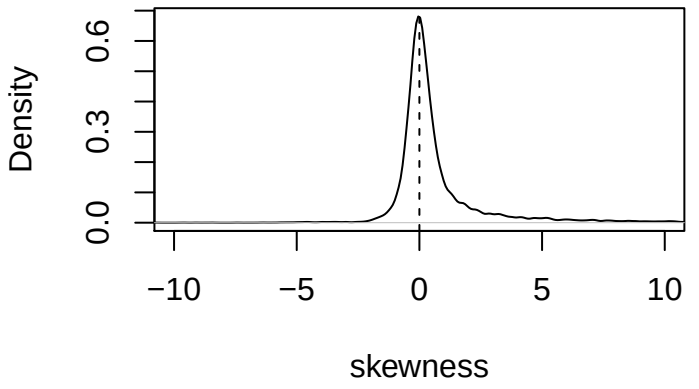
From the remaining 20264 gene expressions, 670 have at least 95% of 0s:

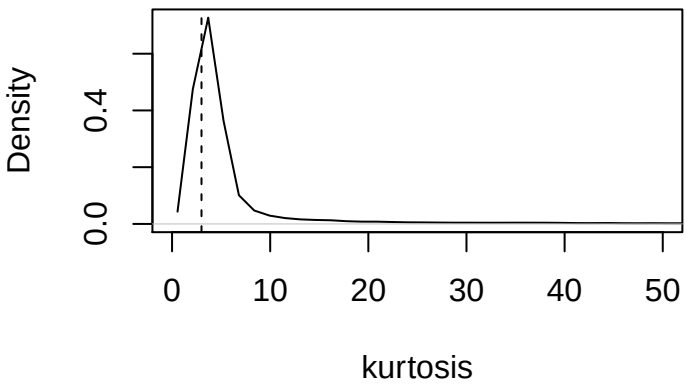


The predictors don't follow distributions close to normal. Nornality cannot be assumed:

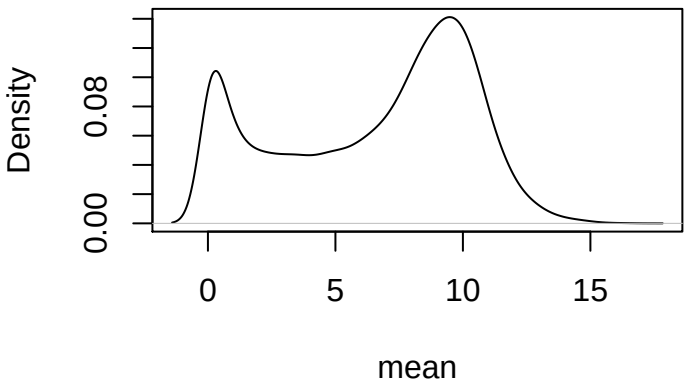


Symmetry and kurtosis:





The distribution of means suggests there are two categories of gene expression:

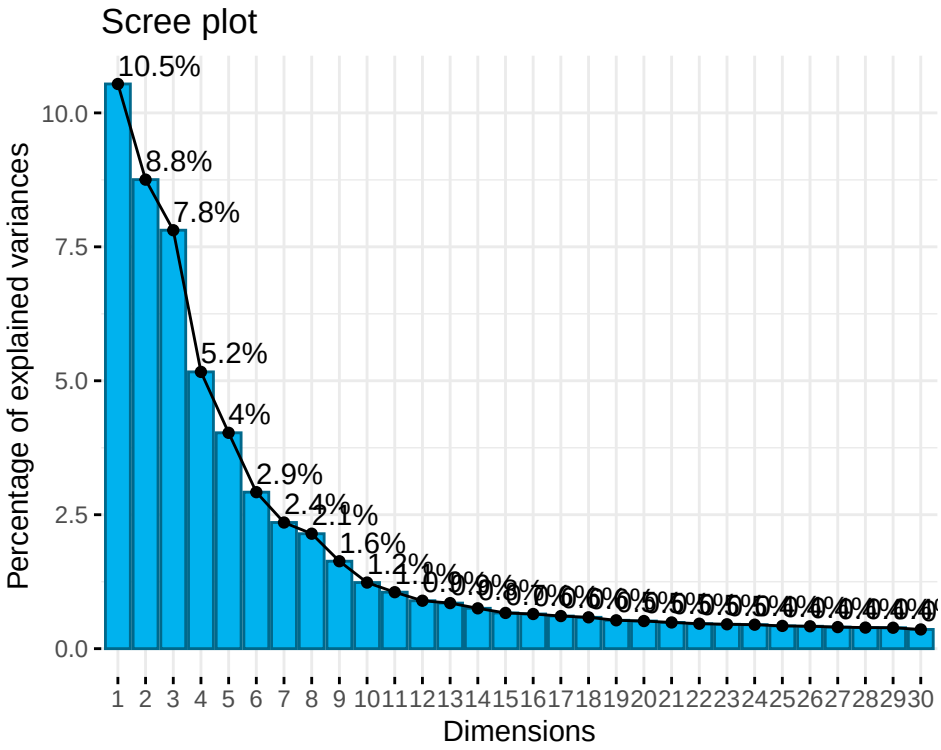


Outliers:

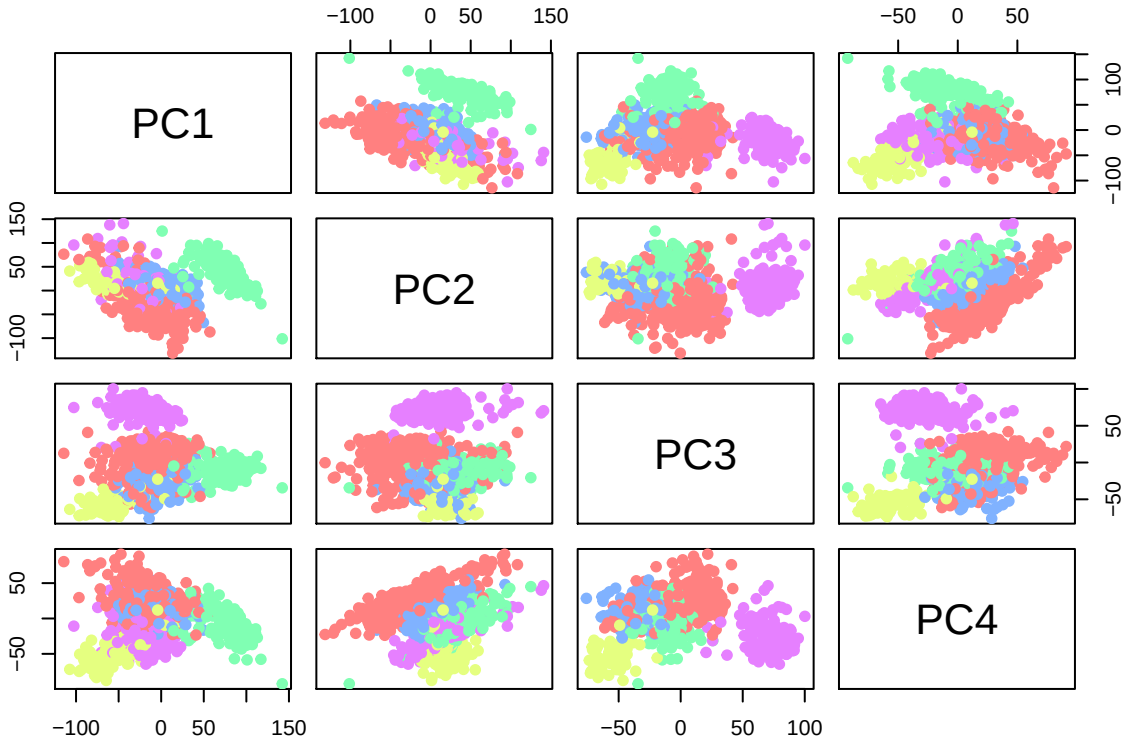
TODO

The data is scaled to help with the training of the prediction model.

At least 45% of the variability of the data is explained by the first 10 PCA components:

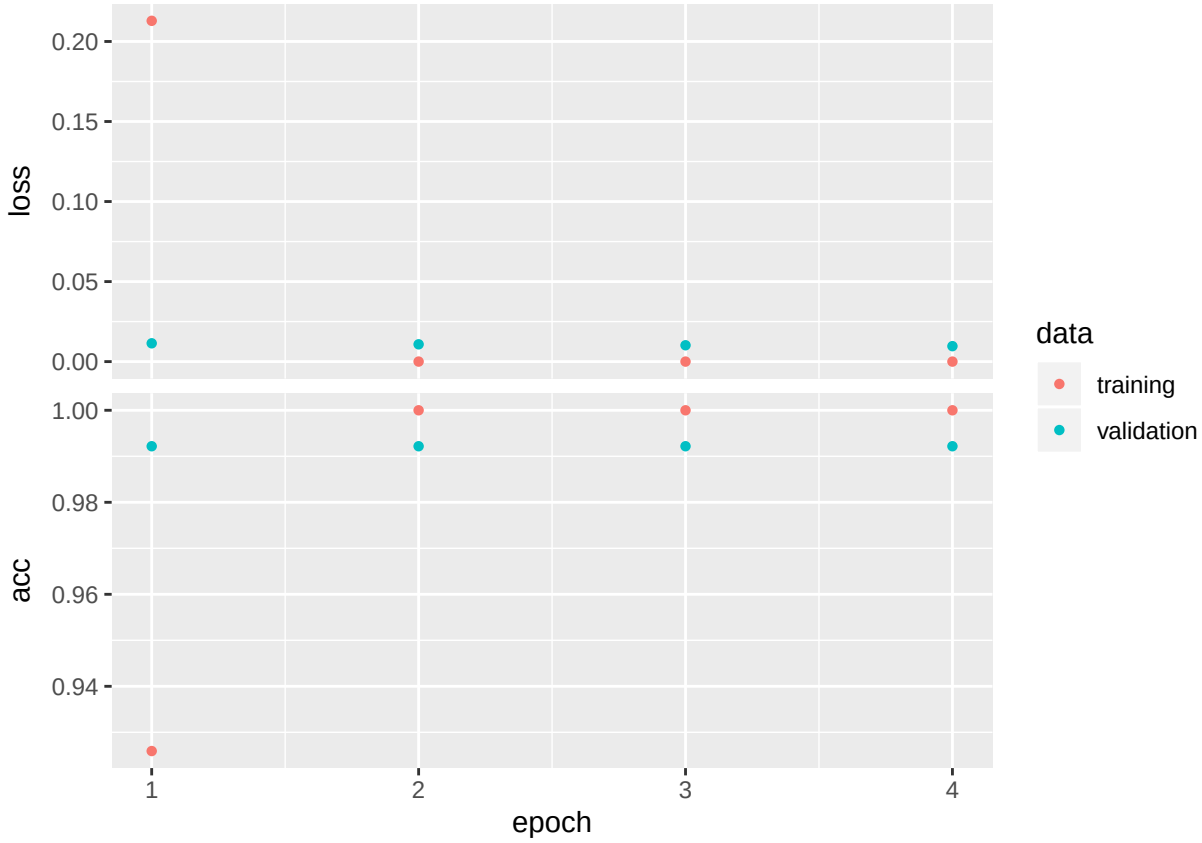


First 4 PCs look enough to classify the types of cancer despite only accounting for 32.27% of the variability of the data, which suggests high collinearity:


PC1
PC2
PC3
PC4
−100
0
50
150
−50
0
50
100
−100
0
50
150
−50
0
50
100
−100
0
50
150
−50
0
50
100

Predictions with a fully connected network

A fully connected network is trained with 80% of the data:



The accuracy of the network is close to 1 (sometimes 1 depending on the random initialization of the network):

```
## $loss
## [1] 8.835016e-06
##
## $acc
## [1] 1
```

Global interpretation with surrogate model

Neural networks use to be powerful prediction tools but they come with a cost in terms of interpretability of the predictions. They contain a huge number of parameters making difficult the identification of features that are influential in the response of the model.

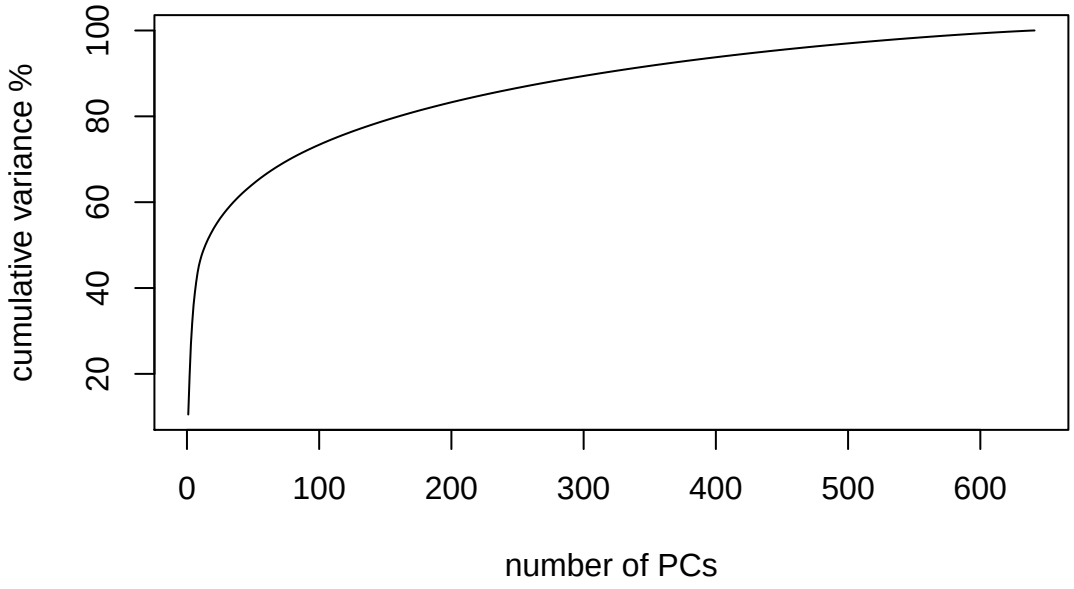
To tackle the interpretability issue of the prediction model (which we'll call *black box* from now on), a surrogate interpretable model will be fitted in parallel with the training data that was used to fit the black box. This surrogate model will help us to undestand the relationship between the features and the response at a global level (i.e. for no particular prediction).

For the interpretable surrogate model we'll use lasso regularization, as it performs feature selection efficiently for high-dimensional data and it's very easy to understand.

The data representation for the interpretations can be different from the data representation used in the predictions. We could use whatever is more convenient for the interpretation, as long as we keep a mapping between both data representations.

To get a more interpretable data representation, we do dimension reduction with PCA. This also will help with the high variance issue. The expert in the domain has to advise if the representation would be useful for interpretation. The expert might be able to understand the meaning of the main components.

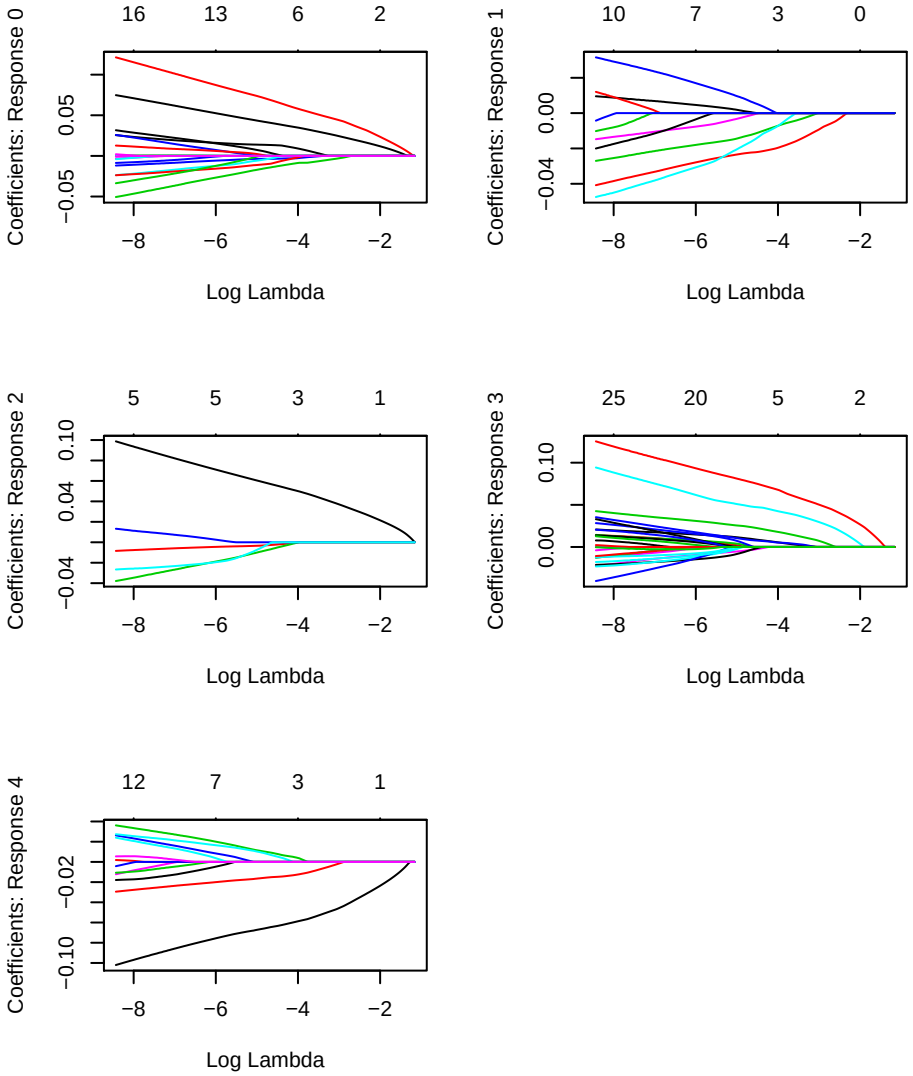
The number of components in the interpretable data representation could be reduced to a percentage of the explained variance, since probably only a few dozens of them will be really influencial. We'll start with the first 100 PCs for now.



Because the purpose of this model is interpretation, we'll add a parameter on top of lasso to select the number of features we want to interpret the black box model. We'll fit several models with different lambdas and pick the lambda with the desired number of selected features (or close to that number, it will depend on the given lambda range). This is done for each category. With a fixed number of features we'll be able to fairly

compare the surrogate model with other models with the same number of features as we'll see later with LIME.

The 10 more influential features will be selected.



The genes selected by the lasso give a global sense of the more influential features in driving the model. The higher the absolute value, the higher the influence (globally) in the output. However a potential issue with high-dimensional data is the high variance in the interpretations, depending on the selected observation the influential genes will be completely different even for the same category, specially in regions of the data space that are complex (non-linear).

Because the number of features is fixed, the selected value for lambda is not optimal for the fitting. Not a problem since the accuracy of the Lasso model is close to 1 with the test data:

```
# Each category has a different lambda. Here we use the smallest one from all the categories.
min_lambda = min(as.numeric(surrogate_lasso_coeffs$lambda))
surrogate_test = predict(surrogate_lasso_pca, newx = x_train_pca$x[,1:pcs_n], s = min_lambda, type="resp
surrogate_pred = max.col(as.data.frame(surrogate_test))-1
```

```r
mean(cancer[training_index] == surrogate_pred)
```

```
## [1] 1
```

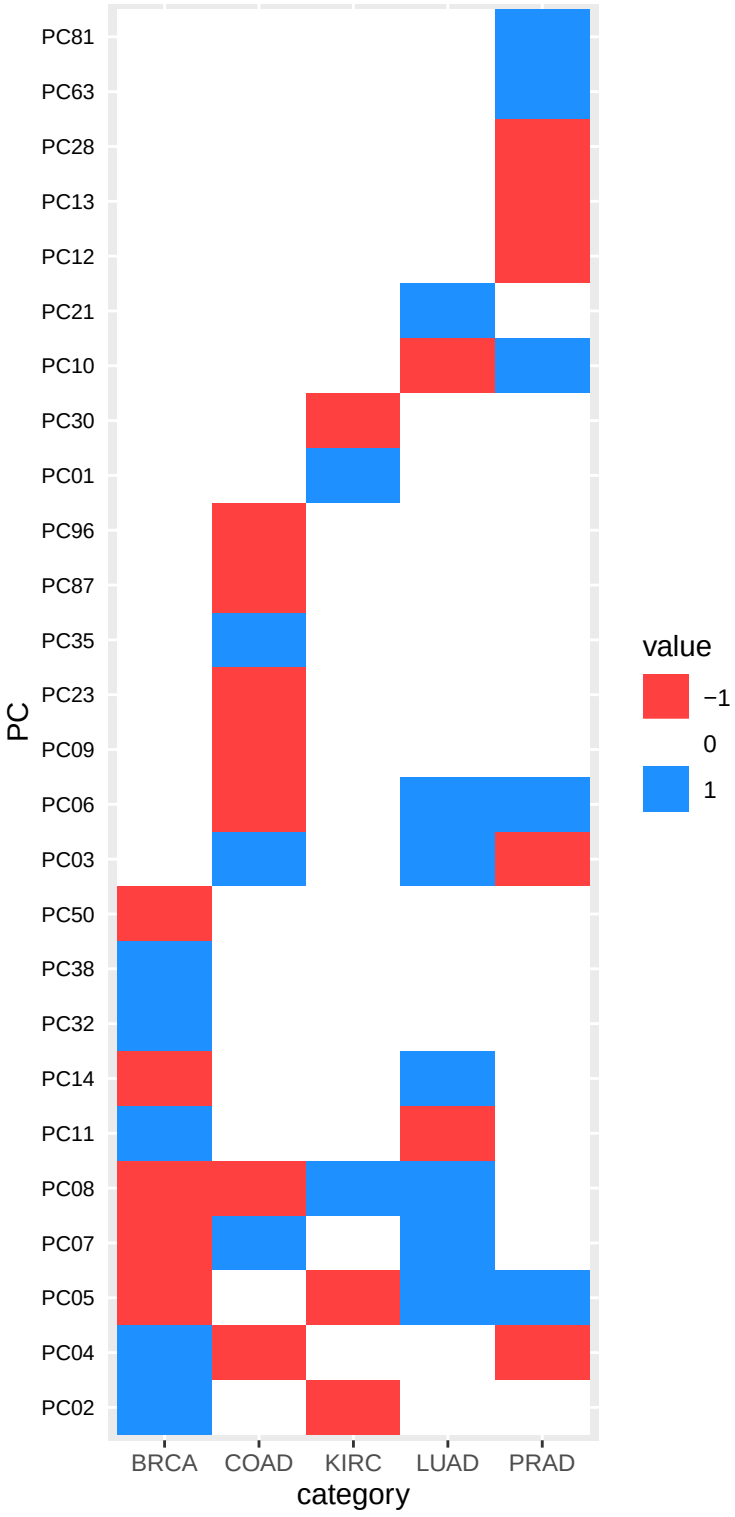
Loss:

```r
label_output_prob = apply(surrogate_test[,,1],1,max) * as.integer(as.integer(cancer[training_index]) ==
1-mean(label_output_prob)
```

```
## [1] 0.001452267
```

TODO: que pasa si el accuracy es mucho peor que el accuracy del black box model?

Heatmap showing the more influential components:

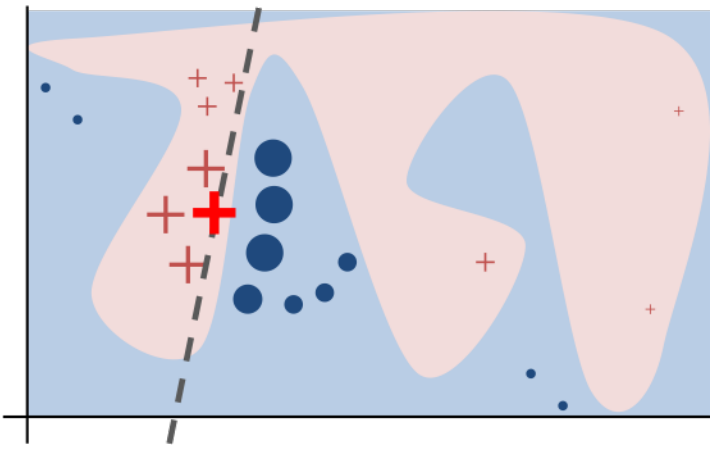


LIME

Intuition

TODO

(image taken from https://github.com/marcotcr/lime)



Formulation

TODO

Algorithm

- In order to fit a local linear model around an observation which classification we want to explain, we need enough data in the surroundings of that observation. To achieve that, kernel densities estimations are computed for each variable and sampled to "simulate" new data, reducing the sparsity of the data and therefore the variance in the local fit for the explanation.

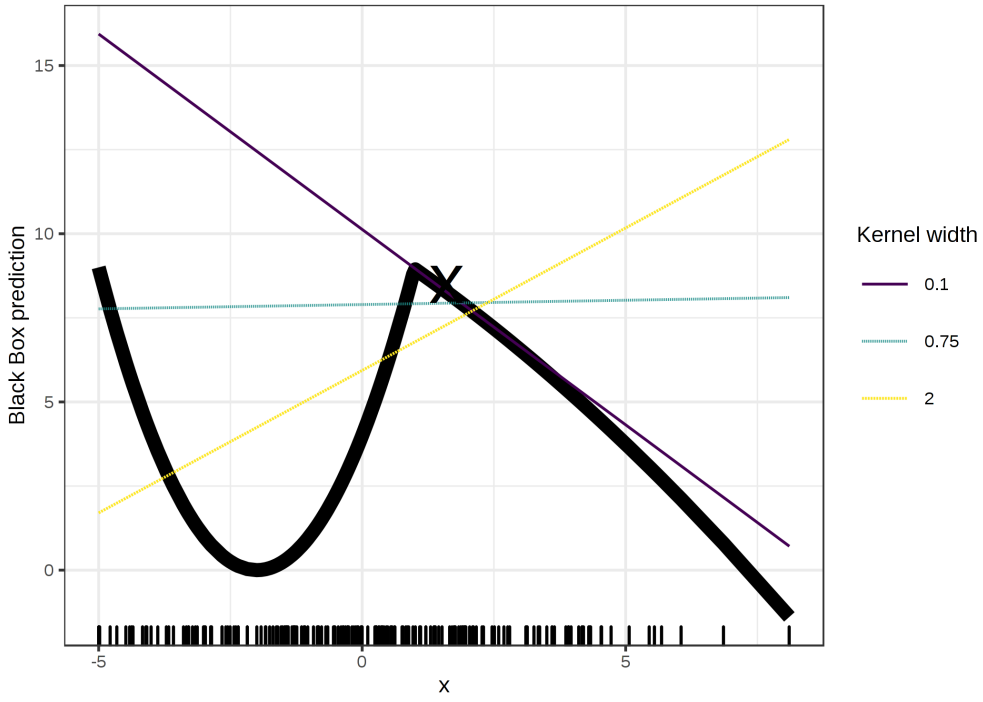
The parameter to tune in this step is the number of "simulated" data points (**n_permutations**).
The risk if the parameter is too low is the high variance of the simulated data each time the prediction for a same observation is interpreted. If each time the observation is explained the simulated data is too different, the variance in the interpretation will also be high, making the interpretations inconsistent and untrustworthy.

TODO: no veo inconveniente en incrementar este parametro al maximo hasta que las limitaciones en CPU/memoria lo permitan.

TODO: Los estimated kernel densities de cada variable no tienen en cuenta las relaciones entre las variables lo cual puede generar data points "irreales". Idea -> usar multivariate kernel density estimation para generar un data set mas real (aplicar dimensionality reduction si es demasiado costoso).

- Of data points simulated by sampling from the features kdes, we are specially interested in those in the surroundings to the observation of interest, since we are fitting a local model. So we need to give more weight to the data close to the instance of interest and this is done with a smoothing exponential kernel. The width of the kernel is a parameter of the LIME model (**kernel_width**) and probably the more tricky one as shown in the following example of a 1-dimensional dataset:

(image taken from https://christophm.github.io/interpretable-ml-book/lime.html)



In this example changes in the kernel width lead to drastic changes in the local linear fit for the instance of interest (the cross in the plot). Adding more dimensions would increase the sensitivity of the width parameter even more.

In the *lime* package, the default value is $0.75\sqrt{p}$ - the more number of dimensions the more the data space is sparse and therefore the kernel width is increased depending on p.
The appropriate value seems to depend on the surroundings of the data point being explained so there isn't a clear rule of thumb to follow for this parameter.

Too small values could lead to insufficient data to fit the local model and too much variance in different interpretations for the same observation to explain.
Too high values could lead to loosing "locality" and hence incorrect interpretations.

TODO: Buscar algun criterio, quizas basado en la complejidad (clasificaciones con menos o mas certidumbre) para seleccionar el kernel width.

Another parameter is the distance function that measures the proximity between the instance of interest and the simulated data points. The default option is Gower's distance but others like Euclidean or Manhattan (see *?dist()* for details) can also be used.

TODO: Elegir la funcion mas adecuada teniendo en cuenta:
- la alta dimensionalidad
- la alta colinearidad
- variables no-normales (pero tampoco exponenciales)
- las variables han sido standardizadas

- The next step is to feed the black box with the simulated data to get the classifications required to fit the local model.

- With the simulated data and the corresponding predictions, a linear model is fit. Several local models are available (**feature_select**). We'll choose lasso for two reasons: the high-dimensionality of the

data and to get a better comparison with the global surrogate lasso model. Lasso will use the weights from the smoothing kernel to give more influence to the neighborhood to the original observation to be explained.

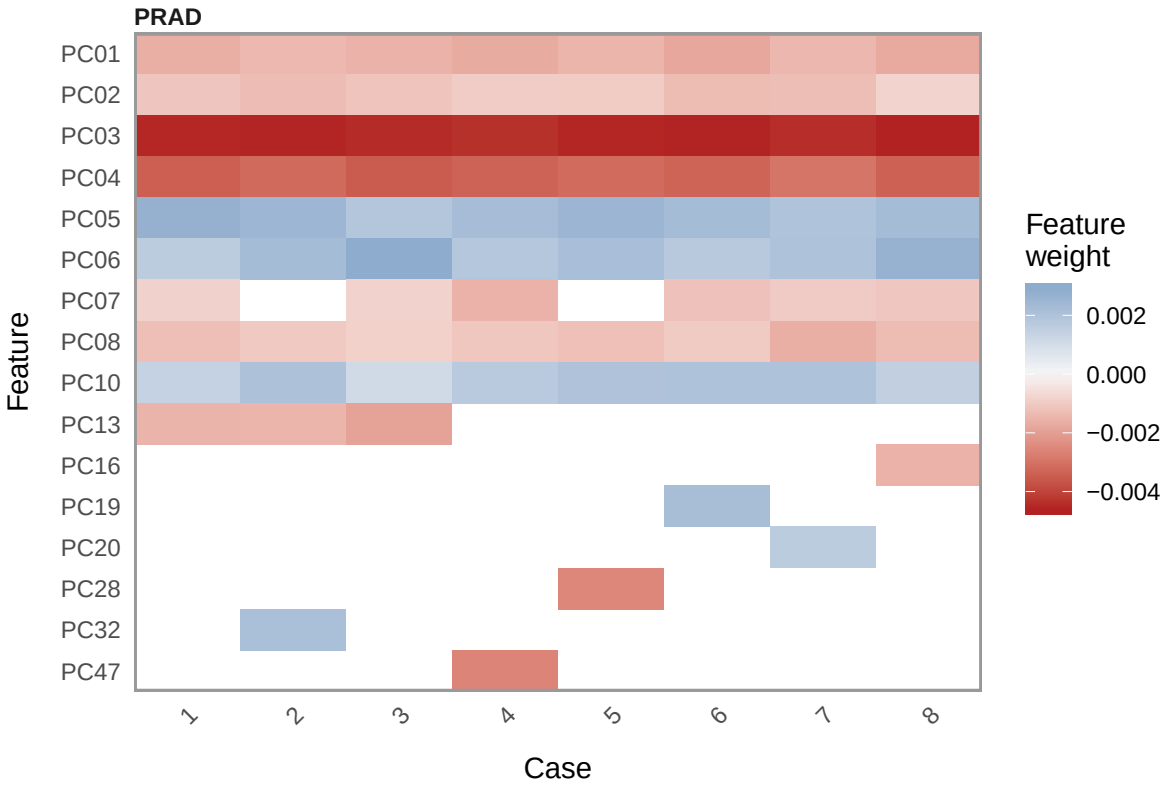
- Finally the coefficients with higher absolute values are selected (**n_features**) to explain the output (**n_labels**, *1* if we are just interested in the selected category).

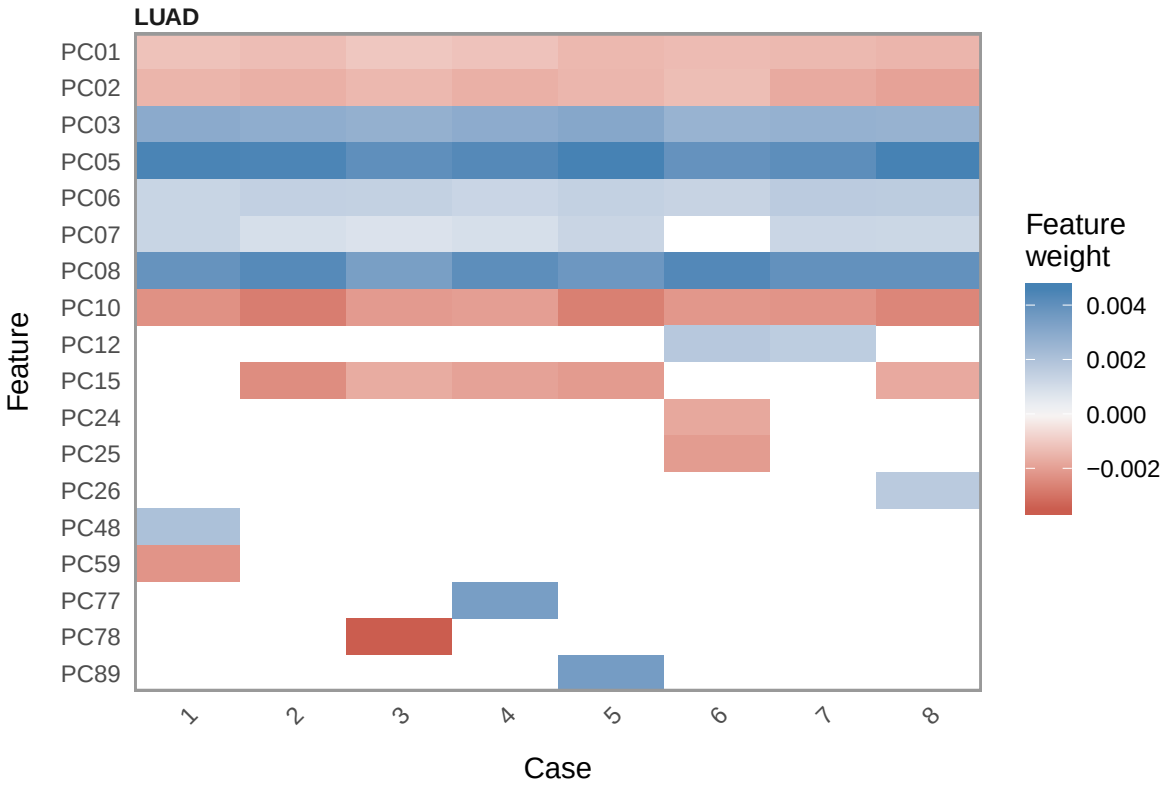
Interpretation with PCA

Let's use LIME to make interpretations of the predictions of individual observations in the training data with PCA reduction.

```
get_PCA_explanation <- function(datapoints_index, n_permutations = 2000)
{
  preprocessing <- function(x){

    # reversing PCA (with the remaining info) to feed the black box which only understands the original
    a = as.matrix(x) %*% t(x_train_pca$rotation[,1:pcs_n])
    b = t(a) + x_train_pca$center
    t(b)
  }

  explainer_PCA <- lime(
    x = as.data.frame(x_train_pca$x)[,1:pcs_n],
    model = black_box,
    use_density = TRUE,
    preprocess = preprocessing,
    bin_continuous = FALSE
  )

  lime::explain(
    # converting train data coordinates to the PCA space:
    x = as.data.frame(x_train[datapoints_index,] %*% x_train_pca$rotation[,1:pcs_n]),
    explainer = explainer_PCA,
    n_permutations = n_permutations,
    feature_select = "lasso_path",
    n_features = 10,
    n_labels = 1
  )
}
```

We analyse the consistency in the interpretations by plotting the more influential components eight times for the same observation. The test is done with two categories: PRAD (this category barely overlaps with other categories) and LUAD (this category overlaps with other categories as seen in the PCA plots - it's harder to predict).


PRAD
PC01
PC02
PC03
PC04
PC05
PC06
PC07
PC08
PC10
PC13
PC16
PC19
PC20
PC28
PC32
PC47
Feature
1
2
3
4
5
6
7
8
Case
Feature weight
0.002
0.000
−0.002
−0.004



Some observations:

- The plots show that the interpretations are consistent.
- There is more variance in the selection of the components for PRAD (easier to predict), probably because the influence is concentrated in just one component (PC3) compared to LUAD.
- R^2 is high in both categories, specially with category PRAD:

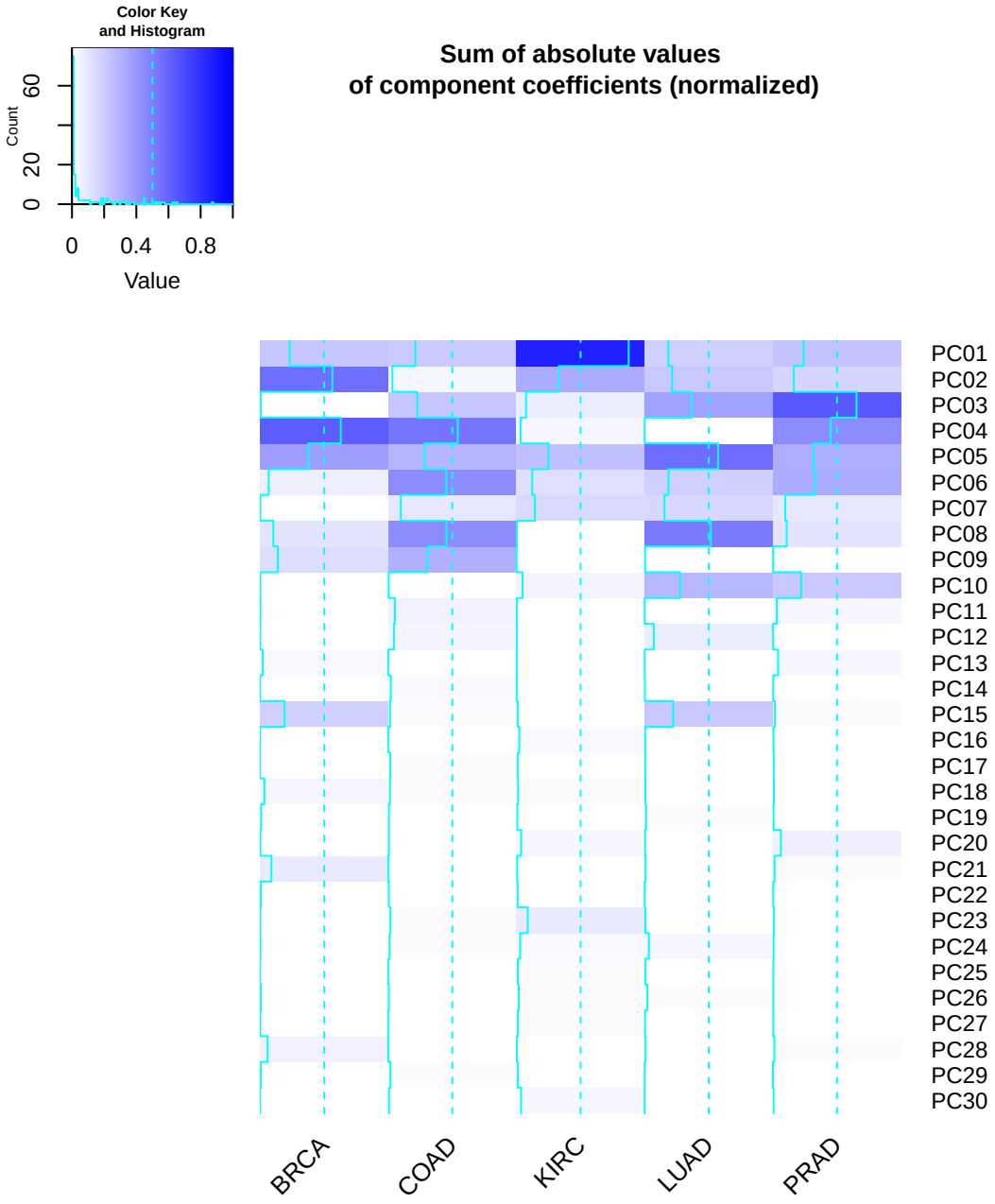
```
##
## R^2 mean for PRAD:
## [1] 0.4434
##
## R^2 mean LUAD:
## [1] 0.3851
```

Comparison of lime interpretations with surrogate lasso model

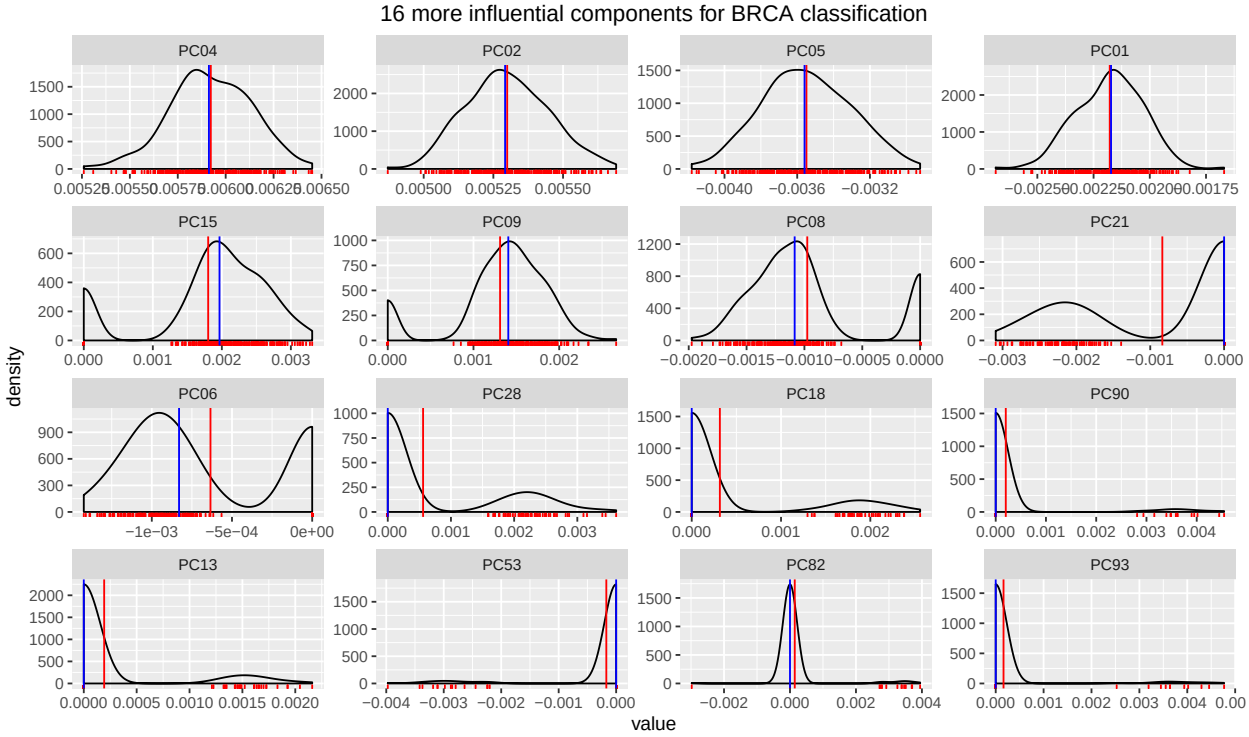
We now check the correlation between the global surrogate model and the local lime models of all the observations in all the categories, using the training data and 2000 permutations for each observation.

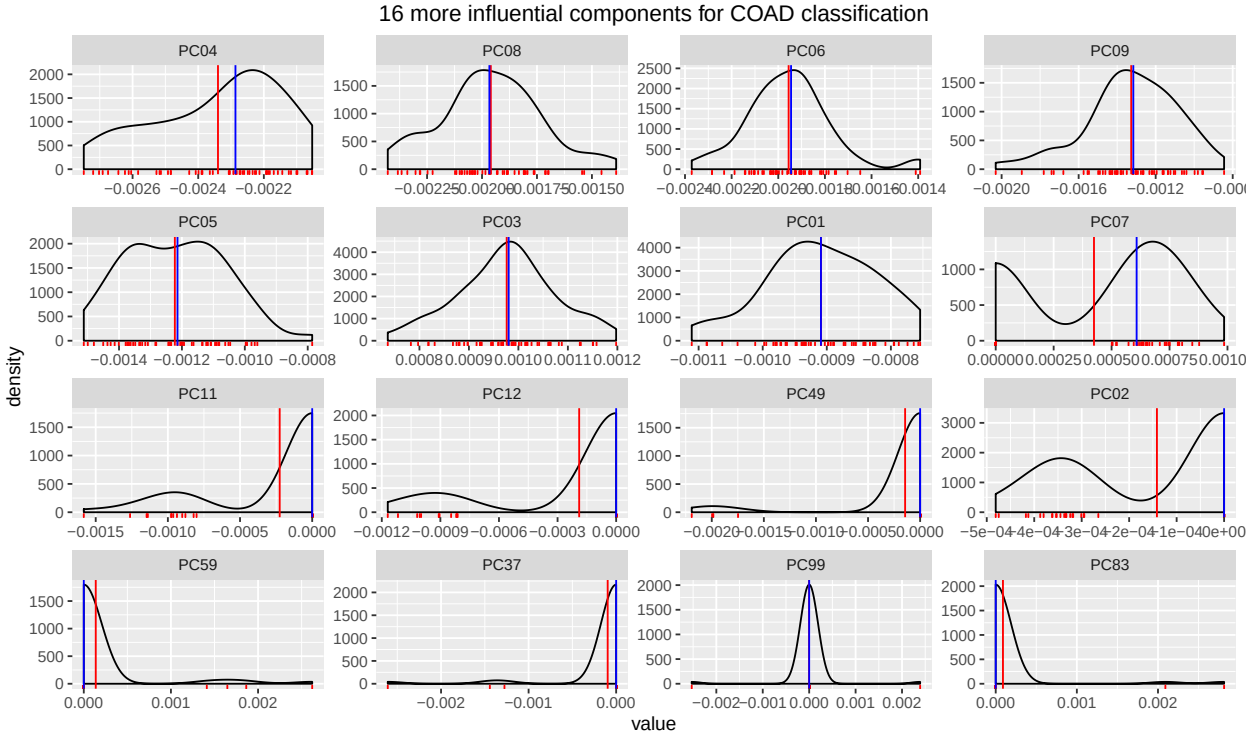
The coefficients of the selected features in the interpretations of all the observations for the same category are summarized in a single list of component coefficients. Each summarized component coefficient will be computed with a statistic describing the central tendency of the coefficient across the observations.

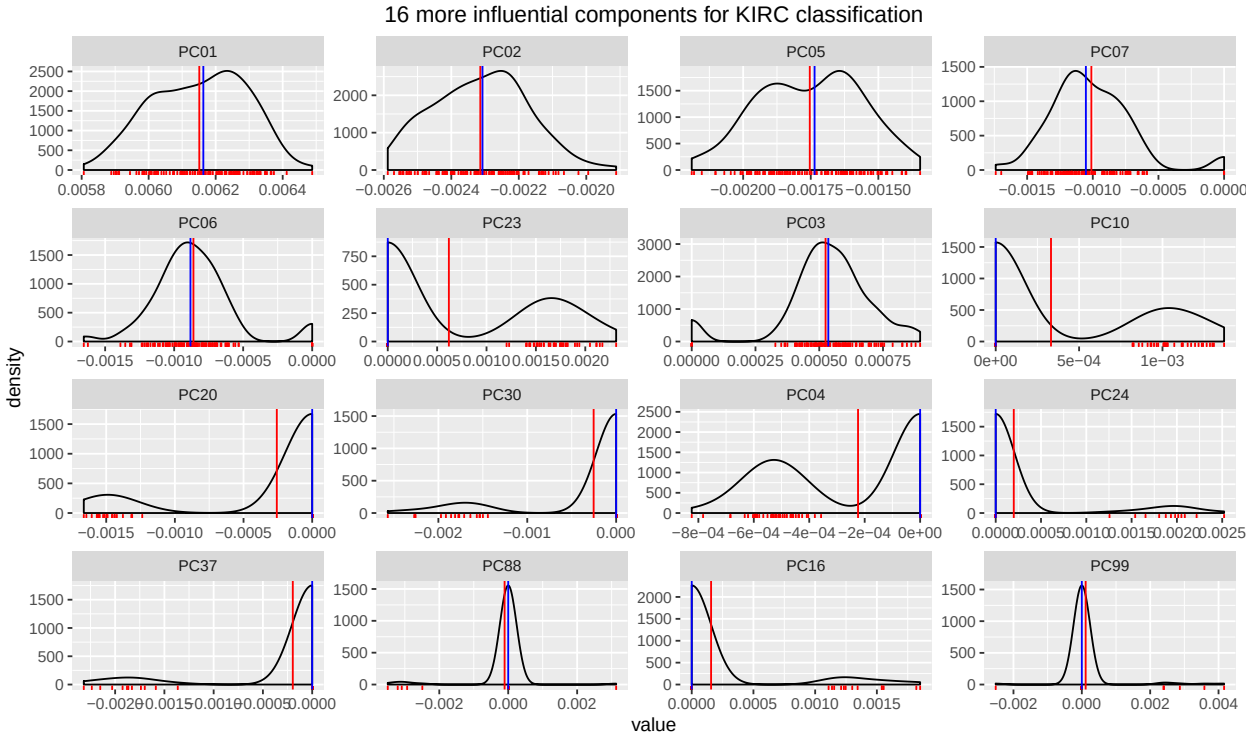
To decide which statistic to use we analyse the more influential features for each category. To measure the influence we use the sum of the absolute values of the coefficients of all the observations for each component.

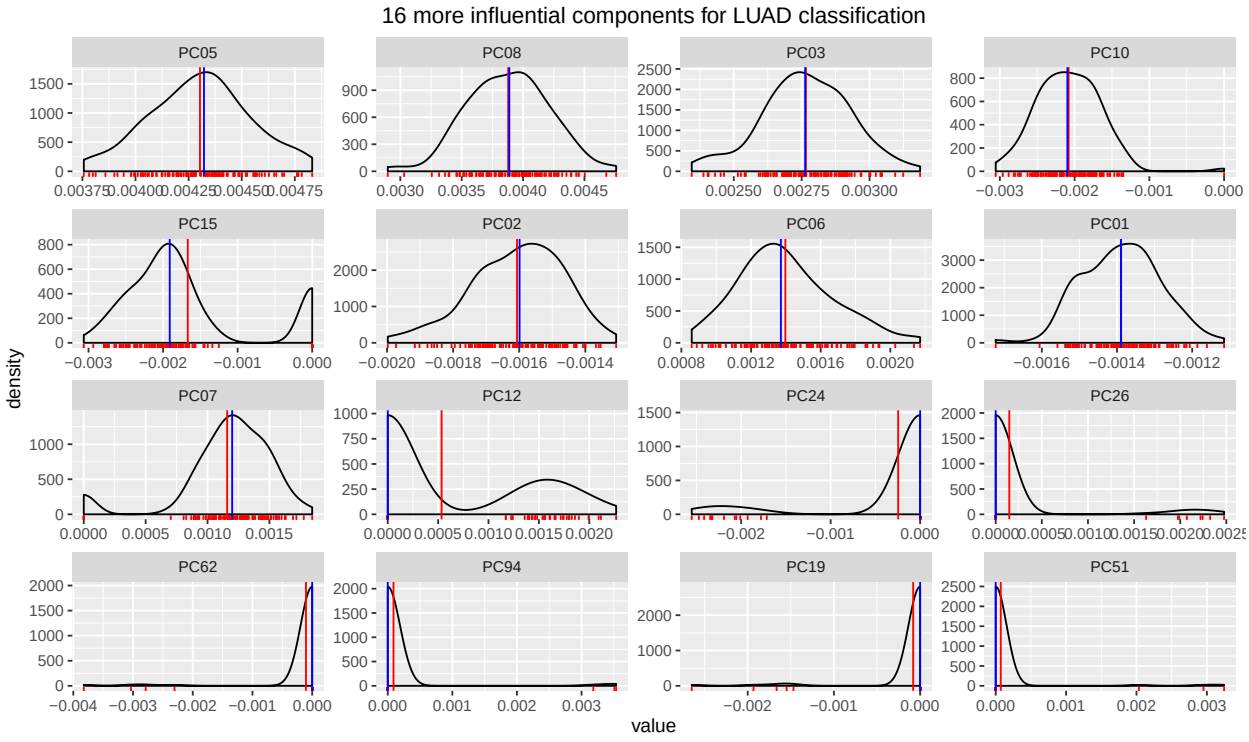


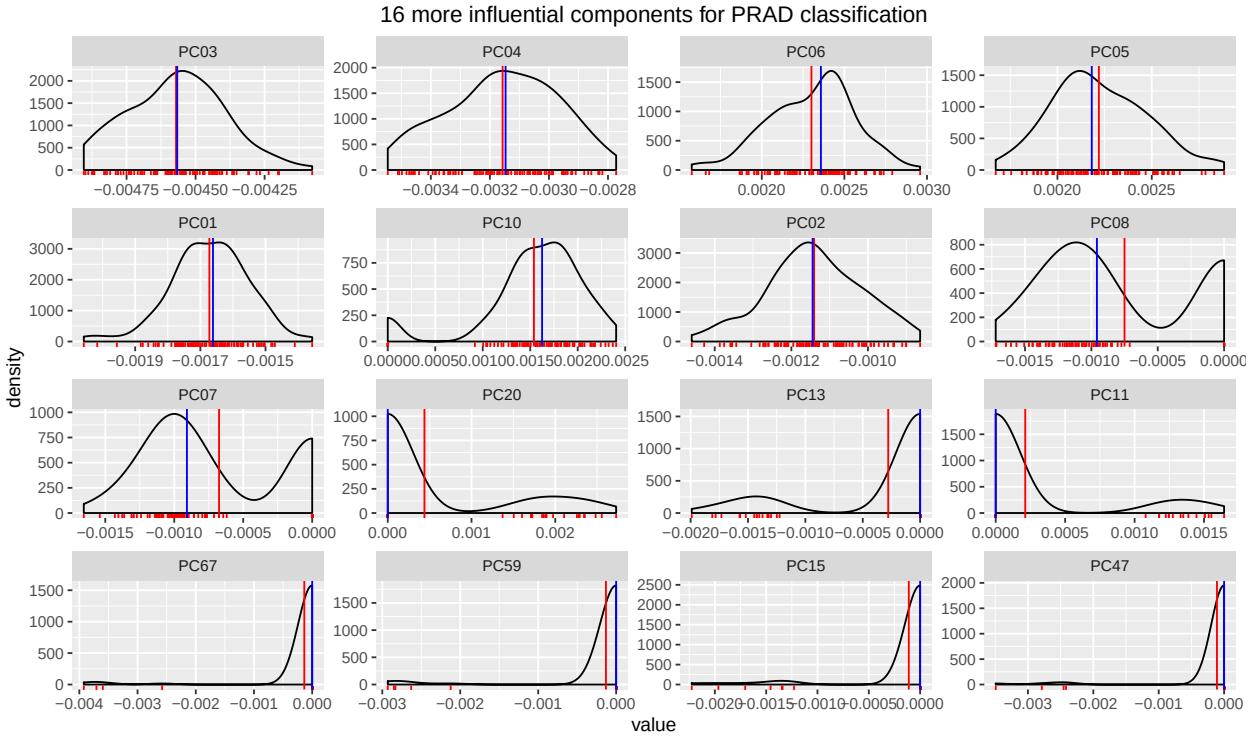
Below are displayed the distributions of the 16 more influential components for all the categories. The vertical red line represents the mean, the blue one represents the median.






16 more influential components for KIRC classification
PC01
PC02
PC05
PC07
PC06
PC23
PC03
PC10
PC20
PC30
PC04
PC24
PC37
PC88
PC16
PC99
density
value

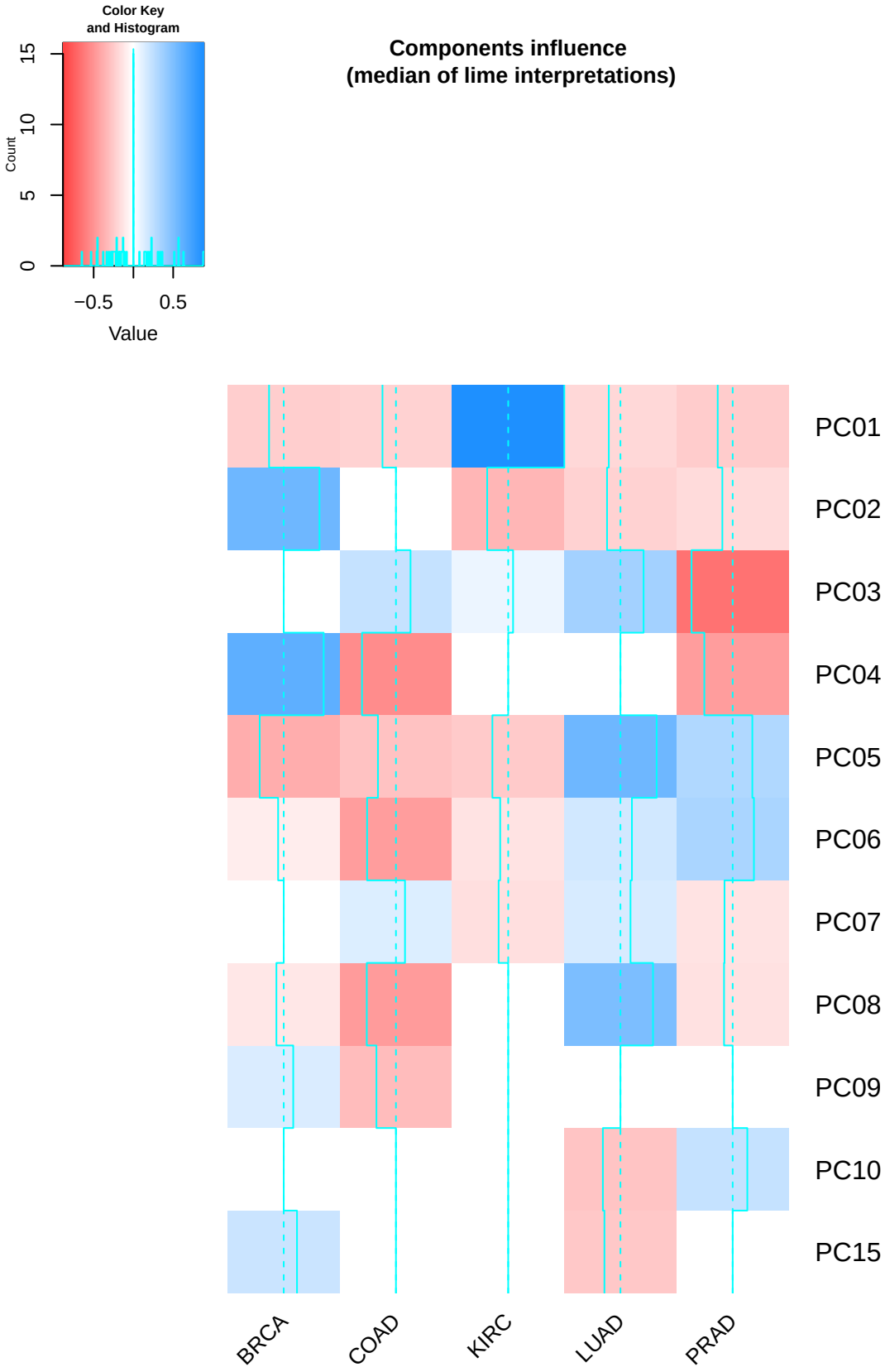

16 more influential components for LUAD classification
PC05
PC08
PC03
PC10
PC15
PC02
PC06
PC01
PC07
PC12
PC24
PC26
PC62
PC94
PC19
PC51
density
value

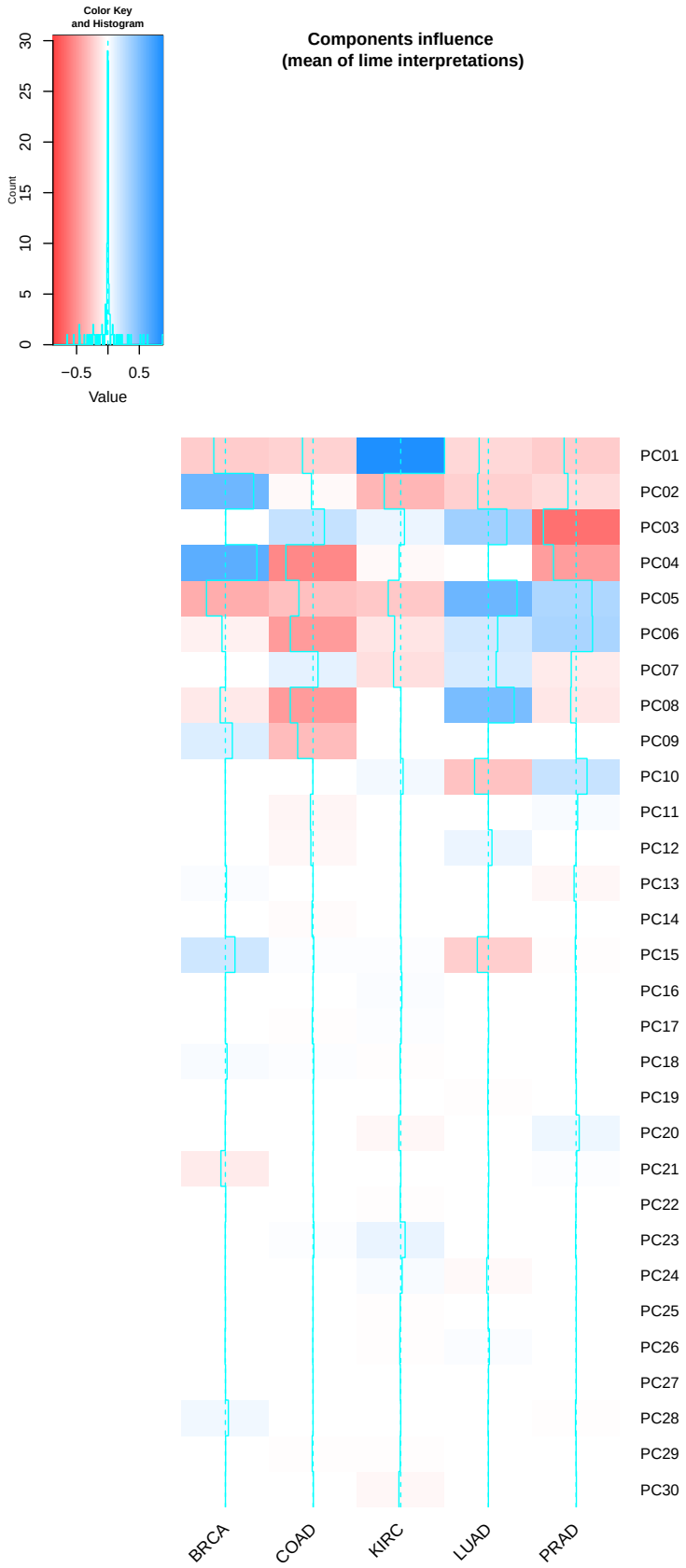


We can see that the more influential components are more or less symmetric, whereas the less influentual ones are bimodal, one of the modes lying on value 0 which represents the absence of influence for a subset of observations.

To avoid the components that only appear as influential for a few observations (and when they are included in the list of 10 more important components their value is very small) we use the median to compute the representing value of the component coefficient for the whole category. The median will ignore components that come up rarely and have small values by setting them to 0 in the summarize component coefficient.

We compute a single set of coefficients for each category using the median of the components coefficients from all the observations for both categories, and get a histogram to compare them with the surrogate lasso model coefficients:


Color Key
and Histogram
Count
0
5
10
15
−0.5
0.5
Value
Components influence
(median of lime interpretations)
PC01
PC02
PC03
PC04
PC05
PC06
PC07
PC08
PC09
PC10
PC15
BRCA
COAD
KIRC
LUAD
PRAD


Color Key
and Histogram
Count
0
5
10
15
20
25
30
−0.5
0.5
Value
Components influence
(mean of lime interpretations)
PC01
PC02
PC03
PC04
PC05
PC06
PC07
PC08
PC09
PC10
PC11
PC12
PC13
PC14
PC15
PC16
PC17
PC18
PC19
PC20
PC21
PC22
PC23
PC24
PC25
PC26
PC27
PC28
PC29
PC30
BRCA
COAD
KIRC
LUAD
PRAD

Heatmap showing the more influential components (lasso VS median of lime interpretations):