Nonparametric Project of Agricultural Productivity in the U.S.

Conformal prediction

Sofia Moroni*

2023-06-27

Contents

1 Load libraries and data

```
library(mgcv)
library(conformalInference)
library(rgl)
```

```
## Warning: il pacchetto 'rgl' è stato creato con R versione 4.1.3
```

```
library(dbscan)
```

```
## Warning: il pacchetto 'dbscan' è stato creato con R versione 4.1.3
```

```
library(pbapply)
```

```
## Warning: il pacchetto 'pbapply' è stato creato con R versione 4.1.3
```

```
library(FNN)
```

```
## Warning: il pacchetto 'FNN' è stato creato con R versione 4.1.3
```

*sofia.moroni@mail.polimi.it

```
data_path = file.path('data')
output_path = file.path('results')
data <-
    read.table(
        file.path(data_path, 'agricultural_indices.csv'),
        header = T,
        sep = ';'
    )

# Sostituzione delle virgole con punti
data<- data.frame(lapply(data, function(x) gsub(",", ".", x)))
data <- as.data.frame(lapply(data, as.numeric))

y = data$Total.agricultural.output
n_b = n = length(y)
```

2 Conformal prediction

```
grid_factor = 1.25
n_grid = 200
alpha = 0.05
```

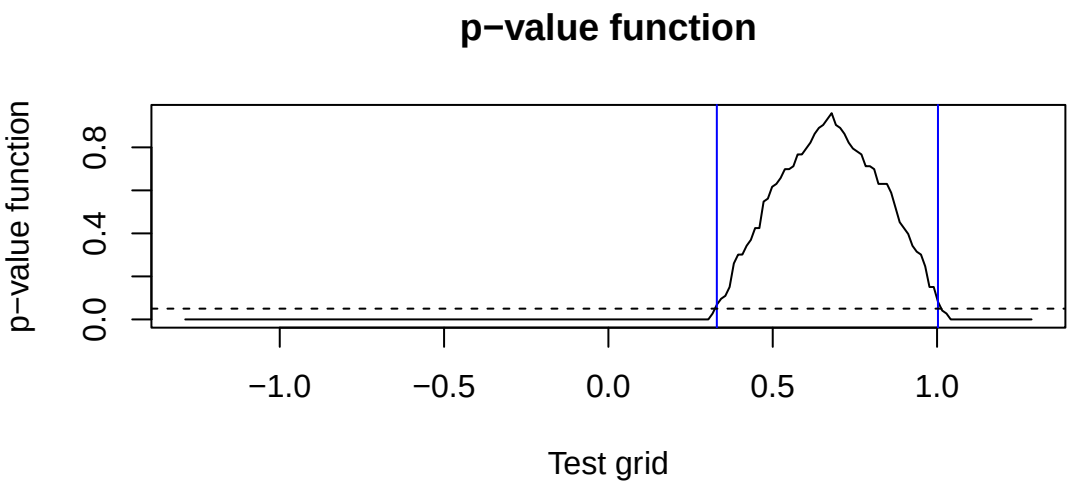
2.1 Using T Prediction Intervals

```
wrapper_full = function(grid_point) {
    aug_y = c(grid_point, y)
    mu = mean(aug_y)
    ncm = abs(mu - aug_y)
    sum((ncm[-1] >= ncm[1])) / (n + 1)
}

test_grid = seq(-grid_factor * max(abs(y)), +grid_factor * max(abs(y)),
                length.out = n_grid)

pval_fun = sapply(test_grid, wrapper_full)
index_in = pval_fun > alpha
pred_t_interval = range(test_grid[index_in])
```

Plot p-value function

```
plot_pval = function(test_grid, pval_fun, pred, alpha) {
    plot(
        test_grid,
        pval_fun,
        type = 'l',
        main = "p-value function",
        xlab = "Test grid",
        ylab = "p-value function"
```

```
    )
    abline(v = pred, col = 'blue')
    abline(h = alpha, lty = 2)
}

plot_pval(test_grid, pval_fun, pred_t_interval, alpha)
```

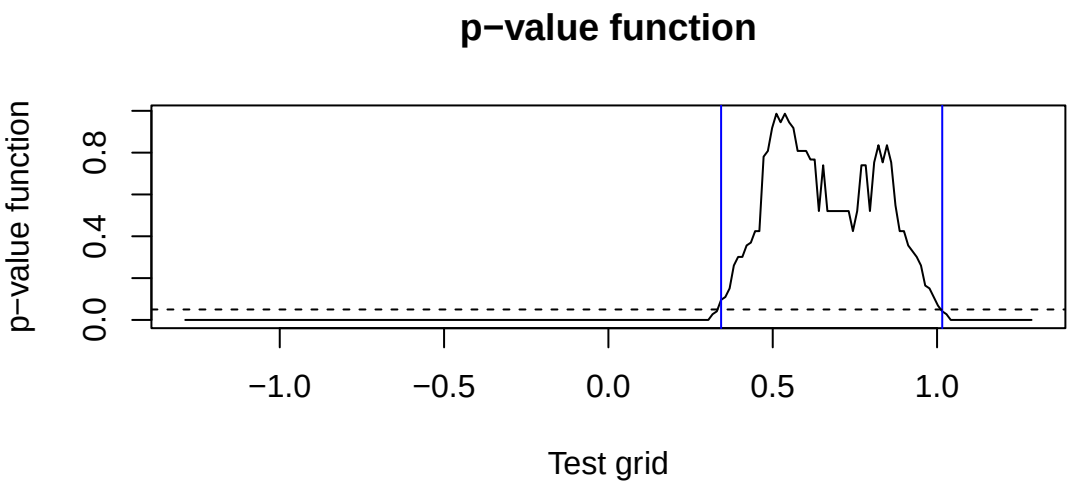


2.2 Using KNN distance

```
pval_fun = numeric(n_grid)
k_s = 0.5
wrapper_knn = function(grid_point) {
    aug_y = c(grid_point, y)
    ncm = kNNdist(matrix(aug_y), k_s * n)
    sum((ncm[-1] >= ncm[1])) / (n_b + 1)
}

pval_fun = sapply(test_grid, wrapper_knn)
index_in = pval_fun > alpha
pred_knn = test_grid[as.logical(c(0, abs(diff(index_in))))]
```

Plot p-value function

```
plot_pval(test_grid, pval_fun, pred_knn, alpha)
```

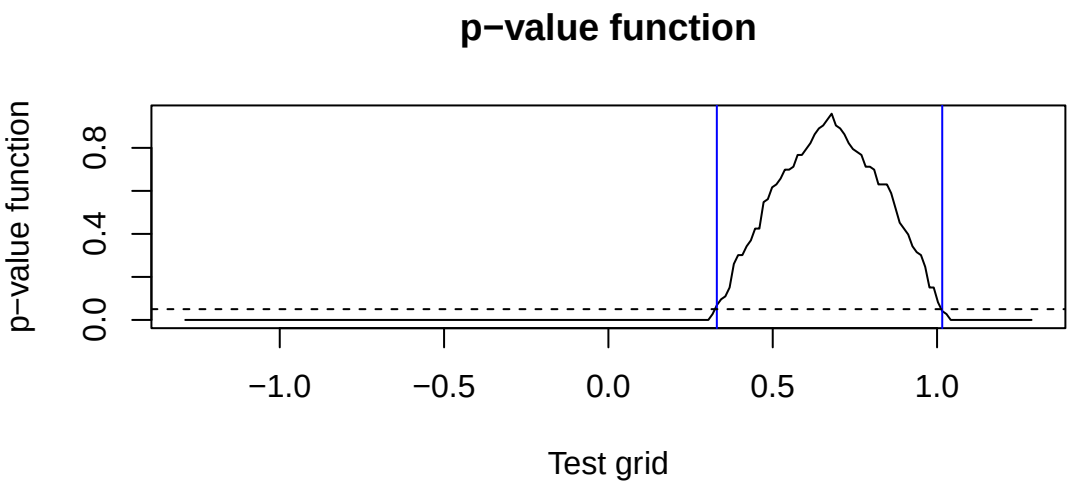


2.3 Using Mahalanobis distance

```
pval_fun = numeric(n_grid)
wrapper_mal = function(grid_point) {
    aug_y = c(grid_point, y)
    ncm = mahalanobis(matrix(aug_y), colMeans(matrix(aug_y)), cov(matrix(aug_y)))
    sum((ncm[-1] >= ncm[1])) / (n_b + 1)
}

pval_fun = sapply(test_grid, wrapper_mal)
index_in = pval_fun > alpha
pred_mahalanobis = test_grid[as.logical(c(0, abs(diff(index_in))))]
```

Plot p-value function

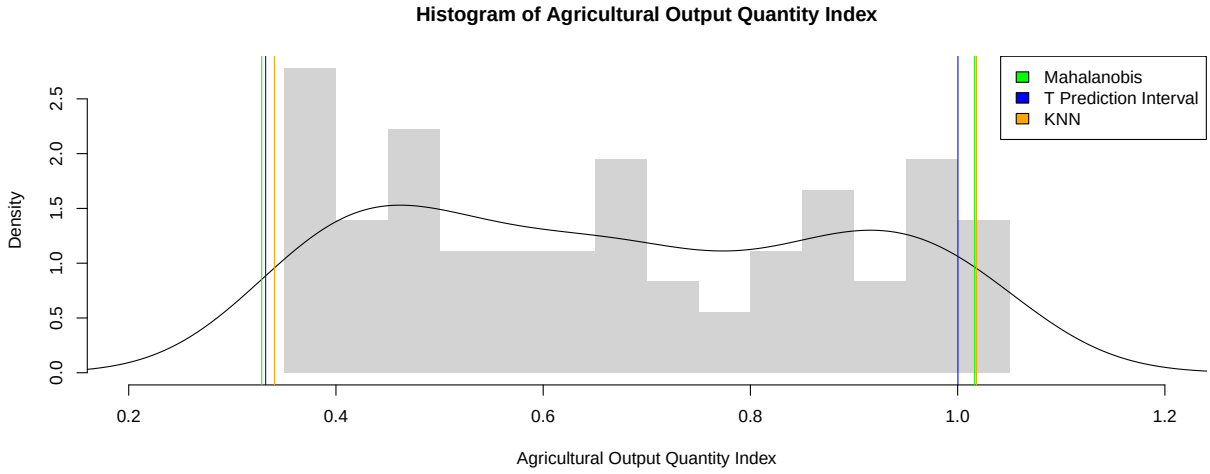
```
plot_pval(test_grid, pval_fun, pred_mahalanobis, alpha)
```



3 Show result

Plot histogram of target variable

```
hist(y,
    breaks = 10,
    freq = FALSE,
    main = 'Histogram of Agricultural Output Quantity Index',
    xlab = 'Agricultural Output Quantity Index',
    xlim = c(0.2,1.2),
    border = NA
)
lines(density(y))

abline(v = jitter(pred_mahalanobis, amount=0.003), col = 'green', lwd = 1)
abline(v = jitter(pred_t_interval, amount=0.003), col = 'blue', lwd = 1)
abline(v = jitter(pred_knn, amount=0.003), col = 'orange', lwd = 1)


legend("topright",
       legend = c("Mahalanobis","T Prediction Interval", "KNN"),
       fill = c("green","blue", "orange"))
```




```
result = data.frame(
    rbind(
        "Mahalanobis"=pred_mahalanobis,
        "T Prediction Interval"=pred_t_interval,
        "KNN"=pred_knn
    )
)
names(result) = c("LOWER", "UPPER")

#knitr::kable(result, format = "latex")
knitr::kable(result)
```

	LOWER	UPPER
Mahalanobis	0.3299623	1.015766
T Prediction Interval	0.3299623	1.002827
KNN	0.3429020	1.015766