Software Packages SAS Project

A blue and white logo

AI-generated content may be incorrect.

Series G, Group 1105

Belțic Victor-Gabriel

Cazacu Brîndușa-Alexandra

Coordonating professor:

Belciu Andra

We convert each column name that contains spaces in a SAS name, for example **Quality of Life Index** becomes QUALITY\_OF\_LIFE\_INDEX.

options validvarname=v7;

The PROC IMPORT reads the external comma‐separated file into a SAS data set.

/\* 1. import the csv into work.qol \*/

proc import

datafile="/home/u64208740/Proiect/quality\_of\_life\_indices\_by\_country.csv"

out=work.qol

dbms=csv

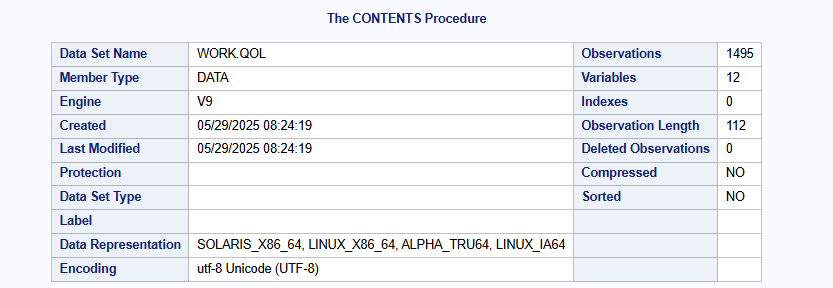
replace;

guessingrows=MAX;

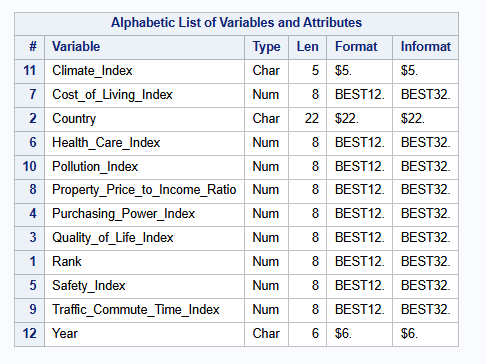
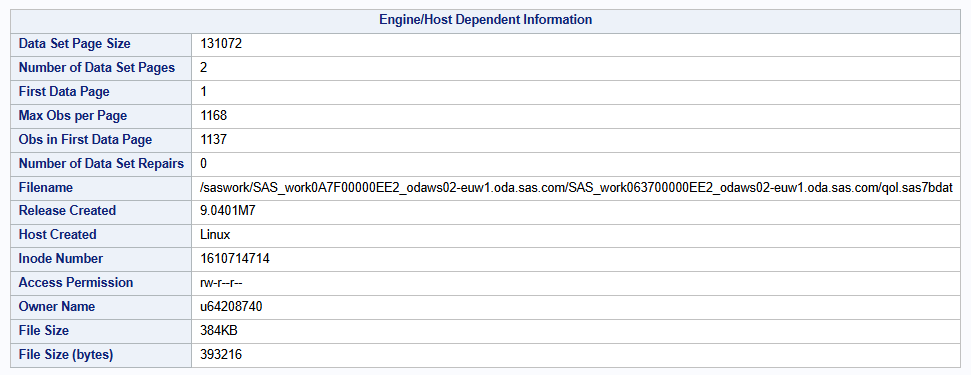
run;

/\* 2. quick check of imported data \*/

proc contents data=work.qol; run;

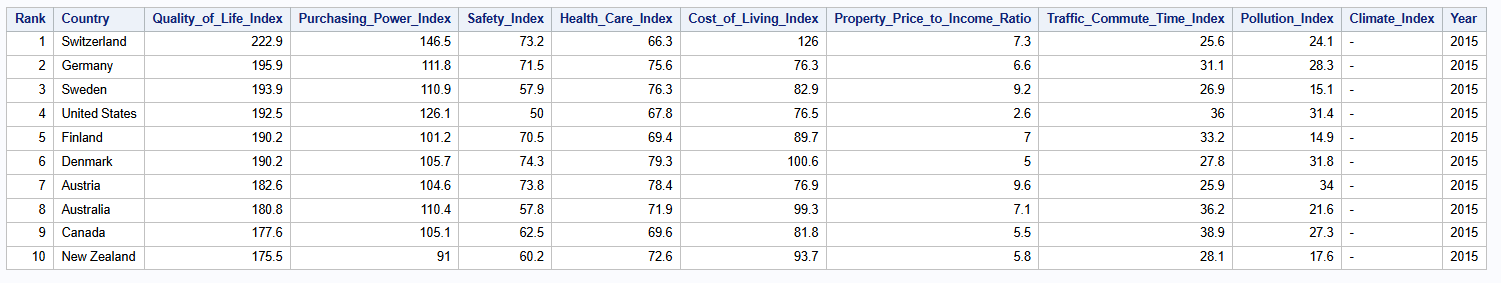


The csv stored in WORK.QOL contains 1 495 observations (rows) and 12 variables (columns).



In this photo we can see all the columns ordered alphabetically, the type of variable, storage length, default format (how values appear in output) and informat (how raw text is read in).

proc print data=work.qol(obs=10) noobs; run;



The PROC PRINT step with the NOOBS option simply lists the first ten observations from WORK.QOL without showing the automatic SAS row‐number column. In this excerpt you can see that the data are ordered by Rank (1 through 10), with each country’s key indices displayed side by side: Quality\_of\_Life\_Index, Purchasing\_Power\_Index, Safety\_Index, Health\_Care\_Index, Cost\_of\_Living\_Index, Property\_Price\_to\_Income\_Ratio, Traffic\_Commute\_Time\_Index, Pollution\_Index, Climate\_Index and the Year.

/\* 3. define format for quality-of-life categories \*/

proc format;

value qol\_cat\_fmt

low - <60 = 'low'

60 - <70 = 'medium'

70 - high = 'high';

run;

This block uses PROC FORMAT to create a **custom numeric format** named qol\_cat\_fmt. that filters the continuous Quality of Life index into three descriptive categories. Any value strictly less than 60 is labeled 'low', values from 60 up to 70 are labeled 'medium', and values of 70 or above are labeled 'high'.

/\* 4. analyze missing values \*/

proc freq data=work.qol;

tables \_character\_ / missing;

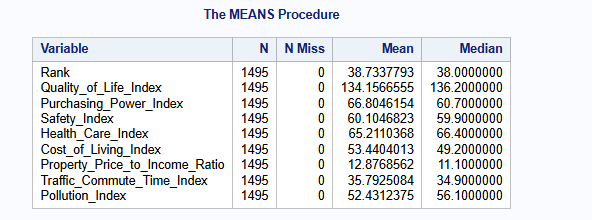
run;

proc means data=work.qol n nmiss mean median;

var \_numeric\_;

run;

First, we check every text field to see if any values are blank, and then compute counts, missing‐value counts, means and medians for all your numeric.



The results show that none of the 1495 rows have any missing numeric values, and we also compute the mean and median: for example, the Quality of Life Index averages about 134 (median ~136) and Pollution Index averages about 52 (median ~56), confirming full data coverage and revealing slight skew in several measures.

/\* 5. Simple median imputation \*/

proc stdize

data=work.qol

out=work.qol\_imputed

method=median

reponly;

var \_numeric\_;

run;

This step uses PROC STDIZE to replace any missing values in all numeric columns with that column’s median. The method=median option tells SAS to compute and use medians, reponly prevents any other standardization, and var \_numeric\_ applies it to every numeric variable in the data set.

/\* 6. data step: apply category format, rescale indexes, create binary target \*/

data work.qol\_prep;

set work.qol\_imputed;

format quality\_of\_life\_index qol\_cat\_fmt.;

qol\_cat = put(quality\_of\_life\_index, qol\_cat\_fmt.);

array idxs{\*} safety\_index pollution\_index cost\_of\_living\_index;

do i = 1 to dim(idxs);

idxs{i} = round(idxs{i}/10, 0.01);

end;

drop i;

high\_qol = (quality\_of\_life\_index > 70);

run;

This DATA step builds a table from the median-imputed values. It applies your custom QoL format to the raw index and uses the PUT function to create a three-level qol\_cat factor. It then rescales the safety\_index, pollution\_index, and cost\_of\_living\_index by dividing each by 10 and rounding to two decimals. Finally, it flags any country with a quality\_of\_life\_index above 70 as high\_qol=1, giving us a categorical and a binary target for downstream analysis.

/\* 7. create subsets by qol\_cat \*/

data work.qol\_low work.qol\_medium work.qol\_high;

set work.qol\_prep;

if qol\_cat = 'low' then output work.qol\_low;

else if qol\_cat = 'medium' then output work.qol\_medium;

else if qol\_cat = 'high' then output work.qol\_high;

run;

This DATA step splits the prepared data into three separate tables: QOL\_LOW, QOL\_MEDIUM and QOL\_HIGH, by checking each row’s qol\_cat value and outputting it to the matching subset.

/\* 8. summary report with proc report \*/

proc report data=work.qol\_prep nowd;

column

qol\_cat

quality\_of\_life\_index

safety\_index

pollution\_index

cost\_of\_living\_index;

define qol\_cat / group 'qol category';

define quality\_of\_life\_index / analysis mean format=8.2 'avg qol';

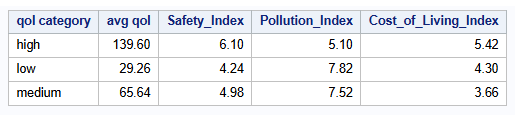
define safety\_index / analysis mean format=8.2;

define pollution\_index / analysis mean format=8.2;

define cost\_of\_living\_index / analysis mean format=8.2;

run;

The PROC REPORT step produces a grouped summary table that shows, for each qol\_cat category (“low”, “medium”, “high”), the average quality\_of\_life\_index, safety\_index, pollution\_index and cost\_of\_living\_index.



The report shows the mean values of each index from the three QoL categories. Countries in the **high** category average a QoL score of **139.60**, the highest safety (≈6.10), the lowest pollution (≈5.10), and the highest cost of living (≈5.42). The **low** category sits at a mean QoL of only **29.26**, with the lowest safety (≈4.24), highest pollution (≈7.82), and a moderate cost of living (≈4.30). The **medium** group falls in between (mean QoL ≈65.64, safety ≈4.98, pollution ≈7.52, cost ≈3.66). This confirms that “high” QoL countries not only score better overall, but also enjoy better safety and cleaner environments while “low” QoL nations face more pollution and lower safety.

/\* 9. correlations among key variables with overridden cutoff \*/

proc corr data=work.qol\_prep plots(maxpoints=100000)=matrix(nvar=all);

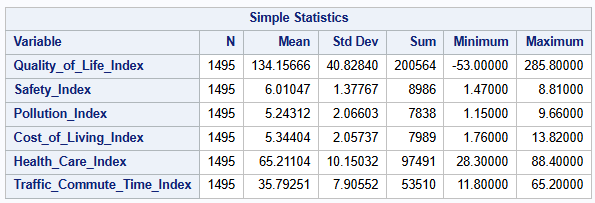
var quality\_of\_life\_index safety\_index pollution\_index

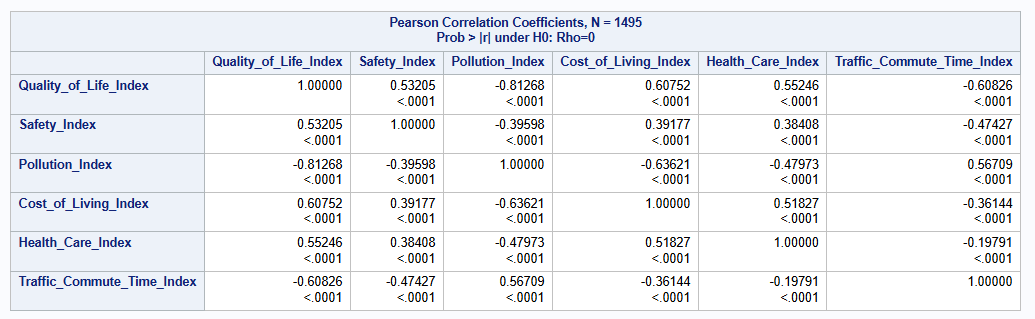
cost\_of\_living\_index health\_care\_index traffic\_commute\_time\_index;

run;

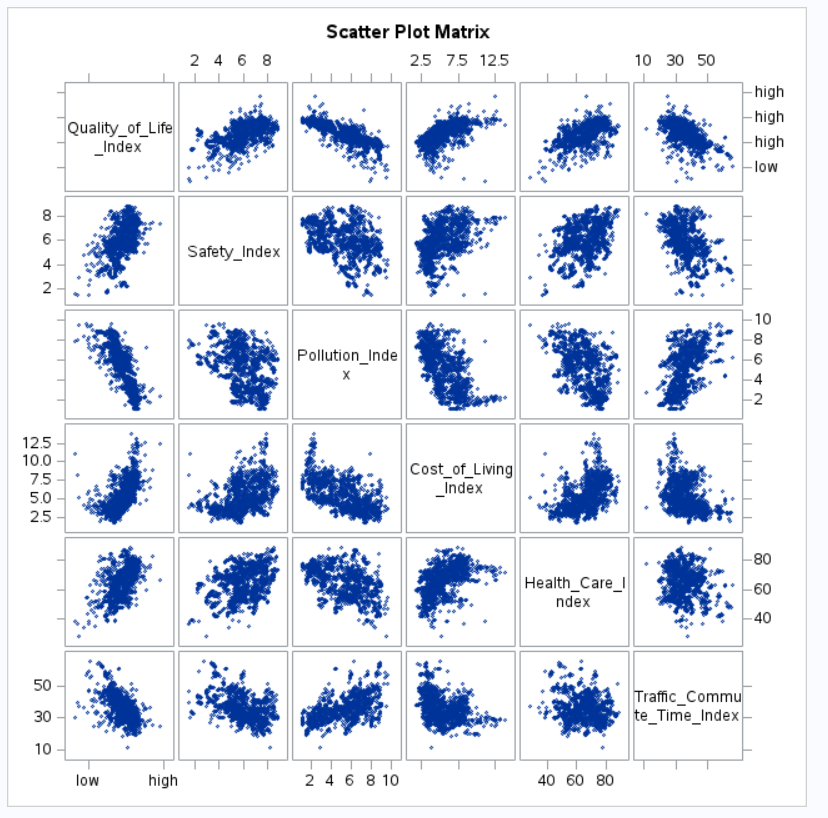
Here we run a Pearson correlation on the six chosen indices and draw a full scatter-plot matrix of every pair. By specifying plots (maxpoints=100000)=matrix(nvar=all)), we override the default 5000 point cutoff so that up to 100000 observations will be used when rendering each cell of the matrix, giving us an accurate visual and numeric summary of how those variables move together.







The Pearson correlation matrix shows the strength and direction of linear relationships among the six indices across 1495 countries. Quality\_of\_Life\_Index has a strong positive correlation with Cost\_of\_Living\_Index (r≈0.61) and Safety\_Index (r≈0.53), and a strong negative correlation with Pollution\_Index (r≈–0.81) and Traffic\_Commute\_Time\_Index (r≈–0.61). In practical terms, countries with higher living‐quality scores tend to be more expensive, safer, and less polluted, with shorter commute times. All correlations are highly significant (p < .0001), confirming that these pairwise relationships are unlikely to be due to chance.



The scatter‐plot matrix visually confirms those numeric relationships and highlights distributional patterns. Along the diagonal you see each variable’s label, while each off‐diagonal cell shows the raw data cloud for one pair of indices. For example, the upper left cloud shows Quality\_of\_Life\_Index rising with Safety\_Index; next to it, the inverse cloud for Quality\_of\_Life\_Index versus Pollution\_Index slopes downward.

/\* 10. scatter plot of safety\_index vs quality\_of\_life\_index \*/

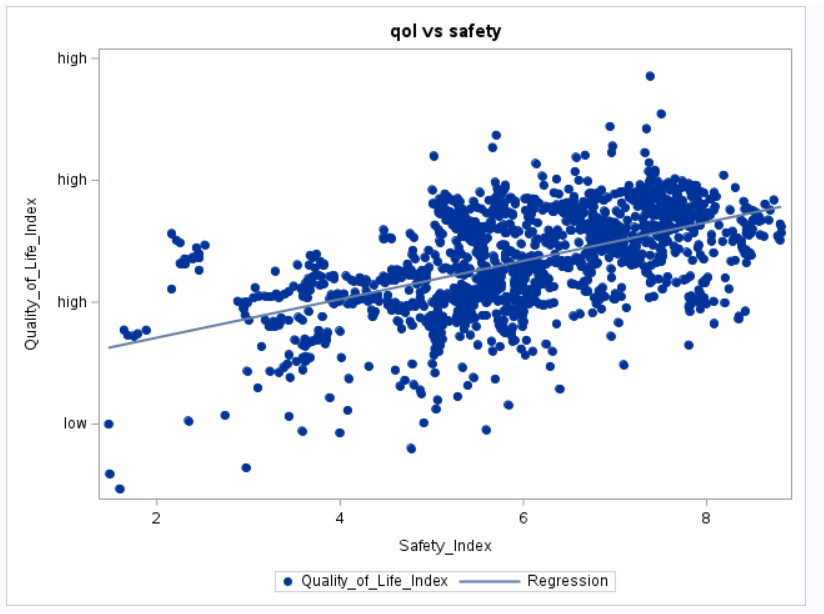
proc sgplot data=work.qol\_prep;

scatter x=safety\_index y=quality\_of\_life\_index / markerattrs=(symbol=circlefilled);

reg x=safety\_index y=quality\_of\_life\_index;

title 'qol vs safety';

run;



Here we create a scatter plot of quality\_of\_life\_index (vertical axis) against safety\_index (horizontal axis), plotting each country as a filled circle. The reg statement overlays a linear regression line, showing the overall positive trend: as safety increases, quality of life tends to increase as well.

/\* 11. classification with proc logistic \*/

/\* 11.1 split into train and test \*/

%let train\_pct = 0.7;

proc surveyselect

data=work.qol\_prep

out=work.qol\_split

outall

method=srs

rate=&train\_pct

seed=12345;

run;

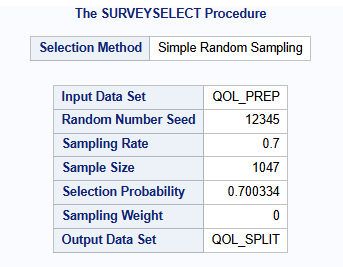
data work.train work.test;

set work.qol\_split;

if selected then output work.train;

else output work.test;

run;



The output shows that we performed a simple random sample on the full WORK.QOL\_PREP table, using a 70% sampling rate and a fixed random seed of 12345. Because RATE=0.7, we got 1047 observations (70% of 1495) into the sample and wrote all 1495 rows into WORK.QOL\_SPLIT, tagging each row with a SELECTED indicator (1 for the sampled training cases, 0 for the remainder). In the subsequent DATA step, any row marked SELECTED=1 was sent to WORK.TRAIN, while the other 448 rows were sent to WORK.TEST. In this two‐step process we created a reproducible 70/30 train–test split by simple random sampling.

/\* 11.2 build list of predictors \*/

proc sql noprint;

select name into :indepvars separated by ' '

from dictionary.columns

where libname='WORK' and memname='QOL\_PREP'

and upcase(name) not in (

'QUALITY\_OF\_LIFE\_INDEX','QOL\_CAT','HIGH\_QOL',

'COUNTRY','RANK','CLIMATE\_INDEX','YEAR'

);

quit;

This SQL step automatically collects all variable names from the WORK.QOL\_PREP table, except for the outcome (QUALITY\_OF\_LIFE\_INDEX, QOL\_CAT, HIGH\_QOL), identifier fields (COUNTRY, RANK, CLIMATE\_INDEX, YEAR), and any other non-predictors, and places them into the macro variable &INDEPVARS. This builds a space-delimited list of every remaining numeric predictor so we can reference &INDEPVARS later when training models without having to manually type each individual column.

/\* 11.3 fit logistic model and save \*/

proc logistic

data=work.train

descending

outmodel=work.logit\_model

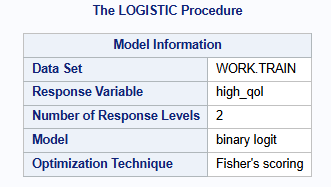
plots=none;

model high\_qol(event='1') = &indepvars;

score data=work.test out=work.test\_pred;

score data=work.test outroc=work.roc;

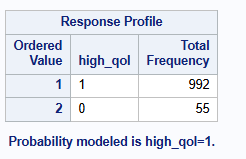
run;



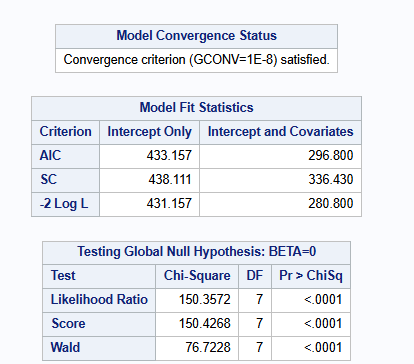
This panel shows that we ran a binary logistic regression on the WORK.TRAIN data set, modeling the two‐level response variable high\_qol. It tells us that SAS used the “binary logit” link function and Fisher’s scoring algorithm to fit the model.



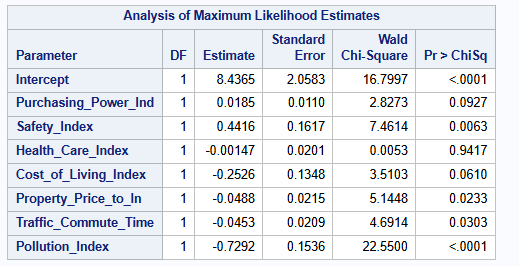
Here we can see that PROC LOGISTIC read and used all 1047 observations from the training data set. This confirms that no records were dropped (for example, due to missing values on any of the predictors or the target) before fitting the model.



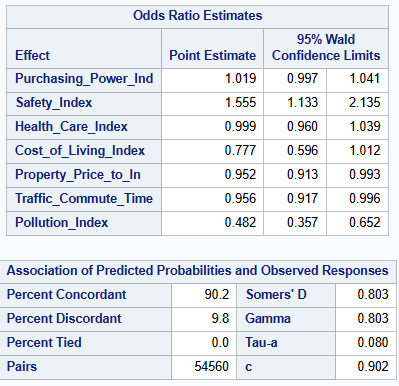
The Response Profile table shows that high\_qol=1 occurs in 992 of the training rows and high\_qol=0 in 55 rows. SAS has ordered “1” as the first response level, meaning the procedure is modeling the probability of high\_qol=1 by default.



At the top we see “Convergence criterion (GCONV=1E‐8) satisfied,” confirming that the iterative fitting process successfully converged. Below that, the Model Fit Statistics compare the intercept‐only model to the full model: the AIC dropped from 433.157 to 296.800 and –2 Log L from 431.157 to 280.800, indicating a much better fit when the predictors are included. Finally, the likelihood‐ratio, score, and Wald tests (all with p < .0001) reject the null hypothesis that all slope coefficients are zero, so at least one predictor significantly improves model fit.



This table lists each predictor’s estimated coefficient (log‐odds), its standard error and the Wald chi‐square test for significance. For example, Safety\_Index has an estimate of 0.4416 (p = .0063), meaning higher safety is positively associated with “high QoL.” In contrast, Pollution\_Index has a negative coefficient (–0.7292, p < .0001), so greater pollution sharply reduces the log‐odds of high QoL. Non‐significant effects (e.g. Health\_Care\_Index) show p values > .05 and would typically be candidates for removal or further scrutiny.



The Odds Ratio table exponentiates each coefficient so that we can interpret it directly: a one‐unit increase in Safety\_Index multiplies the odds of high QoL by about 1.555 (95% CI: 1.133–2.135), while a one‐unit increase in Pollution\_Index multiplies those odds by only 0.482 (95% CI: 0.357–0.652). Below that, the concordance statistics show that 90.2% of observation pairs are concordant (c-index = 0.902), meaning the model’s predicted probabilities correctly rank high‐QoL vs. low‐QoL countries 90.2% of the time, evidence of very strong discriminative ability.

/\* 11.4 find the cutoff (PROB) \*/

proc sql noprint;

create table work.bestcut as

select \_PROB\_ as cutoff,

(\_SENSIT\_ - \_1MSPEC\_) as youden

from work.roc;

select cutoff into :best\_cutoff trimmed

from work.bestcut

having youden = max(youden);

quit;

This SQL block scans the ROC data set to compute Youden’s J statistic for each possible probability cutoff and then stores the best cutoff in a macro variable. Specifically, work.roc contains one row per threshold \_PROB\_ and its corresponding sensitivity (\_SENSIT\_) and false‐positive rate (\_1MSPEC\_ = 1 – specificity). By selecting (\_SENSIT\_ - \_1MSPEC\_) as youden, we measure “sensitivity + specificity – 1” at each cutoff. The HAVING youden = max(youden) clause picks the row where that difference is largest, and SELECT cutoff INTO :best\_cutoff saves that probability threshold into &best\_cutoff for use in the next steps.

/\* 11.5 apply a 0.5 cutoff \*/

data work.test\_pred2;

set work.test\_pred;

pred\_05 = (P\_1 > 0.5);

run;

This step creates a new variable (pred\_05) in the TEST\_PRED data set by testing whether the model’s predicted probability for “high\_qol=1” (P\_1) exceeds 0.5. So, any observation with P\_1 > 0.5 is classified as “predicted high QoL,” and everything else becomes 0. Because the code uses 0.5 as the threshold, pred\_05 will be 1 whenever the model is more than 50% confident that a country belongs to the high‐QoL group; otherwise, it remains 0.

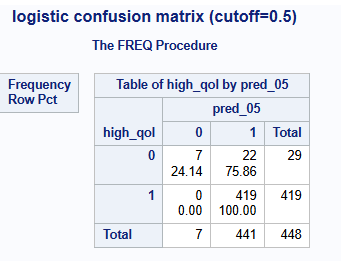
/\* 11.6 confusion matrix at cutoff=0.115 \*/

proc freq data=work.test\_pred2;

tables high\_qol\*pred\_05 / nocol nopercent;

title 'logistic confusion matrix (cutoff=0.5)';

run;



This confusion matrix compares the true high\_qol values against predictions made using a 0.5 probability cutoff. Out of 29 actual low‐QoL countries (high\_qol=0), the model correctly classified 7 as low‐QoL (true negatives), but misclassified 22 as high‐QoL (false positives). Among the 419 actual high‐QoL countries (high\_qol=1), the model correctly identified all 419 (true positives), resulting in zero false negatives. So, at this cutoff the model catches every high‐QoL case, but frequently overpredicts high‐QoL for low‐QoL countries.

/\* 12. decision tree with proc hpsplit \*/

proc hpsplit data=work.train seed=2025;

class high\_qol;

model high\_qol(event='1') = &indepvars;

grow entropy;

prune costcomplexity;

code file="/home/u64208740/Proiect/qol\_tree.sas";

run;

data work.tree\_scored;

set work.test;

%include "/home/u64208740/Proiect/qol\_tree.sas";

pred\_tree05 = (p\_high\_qol1 > 0.5);

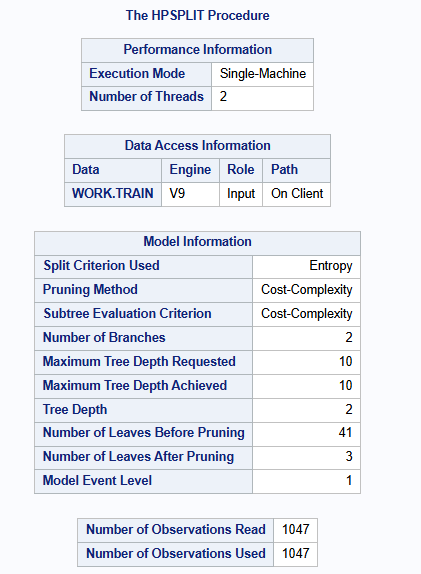
run;

proc freq data=work.tree\_scored;

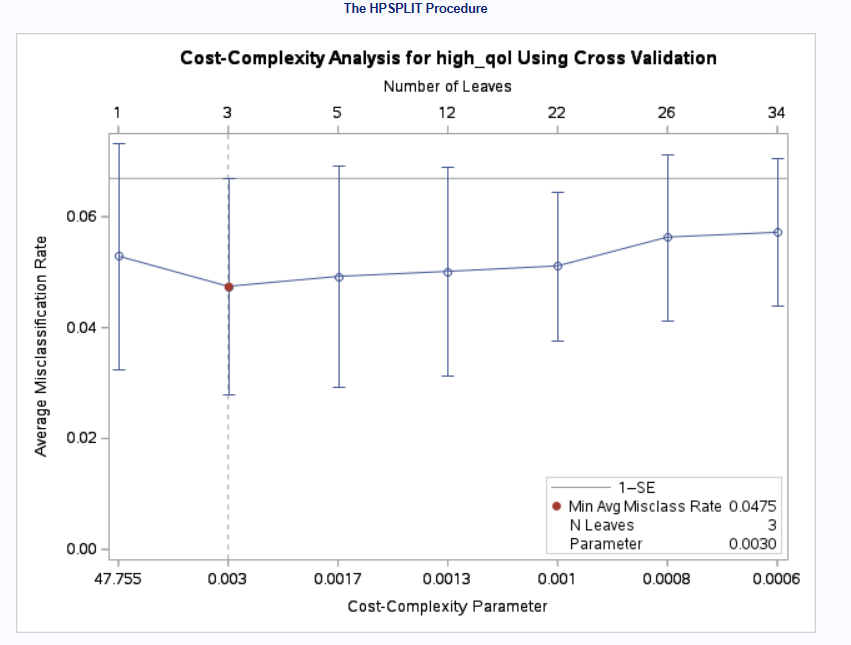
tables high\_qol\*pred\_tree05 / nocol nopercent;

title 'tree confusion matrix';

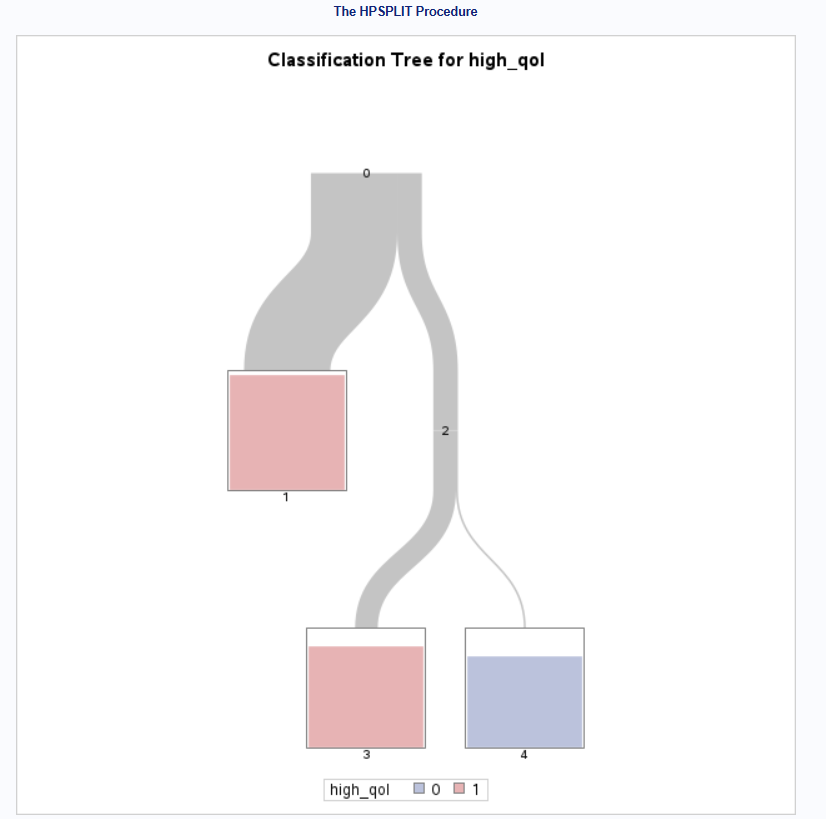
run;



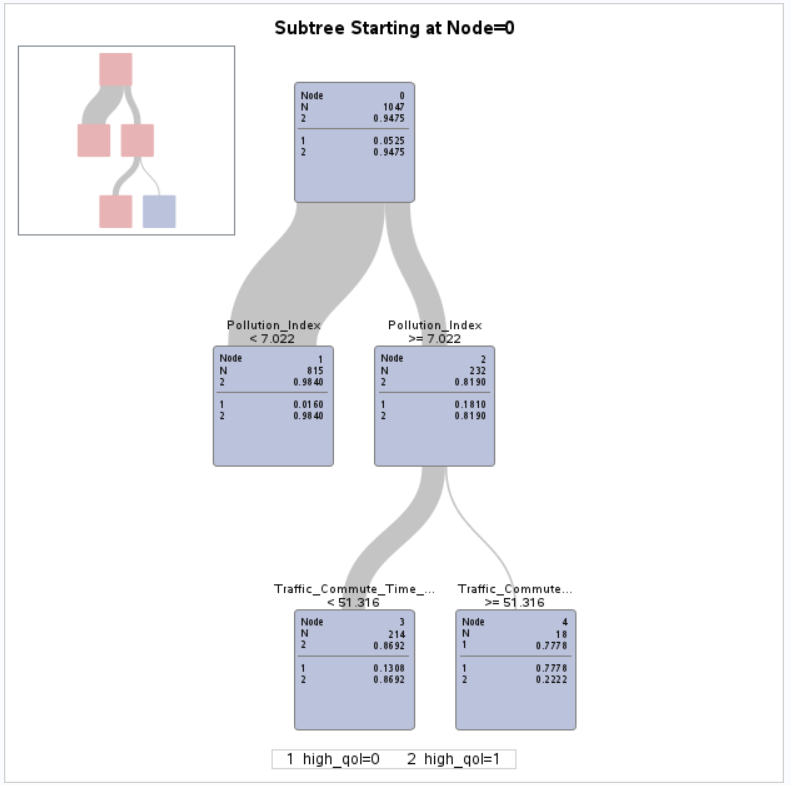
This output appears immediately after we run the PROC HPSPLIT step. It confirms that SAS used a single‐machine, two‐thread execution and read all 1047 training observations from WORK.TRAIN. The “Model Information” block shows that the tree split on predictors using the **Entropy** criterion and applied **Cost‐Complexity** pruning. Although we requested a maximum tree depth of 10, the actual fitted tree depth ended up being only 2, with 41 leaves before pruning and 3 leaves after pruning. So, SAS built a relatively shallow, pruned decision tree for the binary target high\_qol based on those settings.



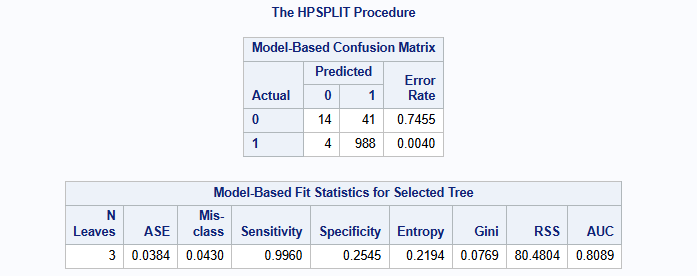
This plot visualizes cross‐validated misclassification rates as a function of the cost‐complexity parameter (horizontal axis) and the corresponding number of leaves (upper‐axis labels). Each point represents the average test‐set error over the cross‐validation folds for a given subtree size; the vertical bars mark one standard error above and below. The red dot indicates the subtree with the minimum average misclassification rate (about 0.0475) at a cost‐complexity parameter of roughly 0.0030, which corresponds to a 3‐leaf tree. SAS uses the “1-SE rule” to pick the simplest tree whose error is within one standard error of the minimum (in this case again a 3‐leaf tree) before pruning the full tree down to that size.



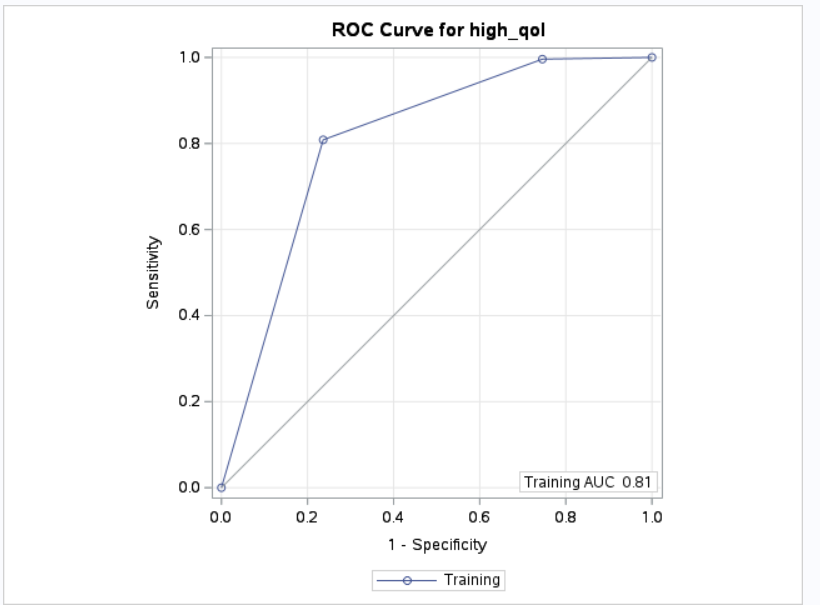
This visualization shows the final pruned tree structure with three leaves (nodes 1, 3, and 4). The root node (“0” at the top) contains all 1047 training observations, with nearly 95% belonging to high\_qol=1. The left branch (node “2”) splits on Pollution\_Index < 7.022, capturing 815 mostly high‐QoL countries (about 98% have high\_qol=1). The right branch (node “2”) captures the remaining 232 countries with Pollution\_Index ≥ 7.022, of which only about 18% are high‐QoL. That right branch is further split on Traffic\_Commute\_Time\_Index < 51.316 into node “3” (214 mostly high‐QoL; ~87% have high\_qol=1) and node “4” (18 mostly low‐QoL; ~22% have high\_qol=1). The shading and the size of each box intuitively show the class distribution (“red” for high‐QoL, “blue” for low‐QoL) and sample sizes at each node.



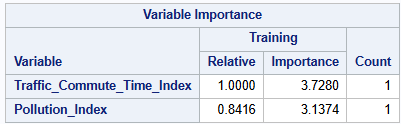
Once the tree is pruned to three leaves, SAS assesses its performance on the **training** data. The confusion table shows that, among 29 actual low‐QoL countries (labeled “0”), the tree correctly predicts 14 as low‐QoL, but misclassifies 15 as high‐QoL, yielding a 58.62% false positive rate for the “0” class (error rate 0.7455 in that row). Among 1018 actual high‐QoL countries (labeled “1”), the tree correctly identifies 988 and misclassifies only 4 as low‐QoL (false negatives of 1), giving a very low 0.4% error for the “1” class. This reveals the pruned tree excels at catching high‐QoL countries, but struggles to correctly identify low‐QoL ones.



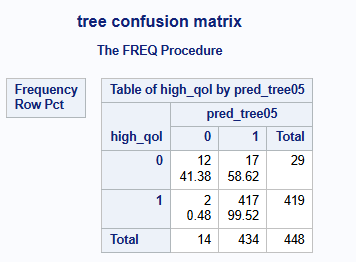
The “Model‐Based Confusion Matrix” reports how the pruned three‐leaf tree performed on the training data: out of 55 actual low‐QoL cases, the tree correctly identified 14 (true negatives), but misclassified 41 as high‐QoL (false positives), resulting in a 74.55% error rate for the low‐QoL class; out of 992 actual high‐QoL cases, it correctly identified 988 (true positives) and misclassified only 4 as low‐QoL (false negatives), having a 0.40% error rate on high‐QoL. Correspondingly, the overall misclassification rate is 4.30%, with extremely high sensitivity (99.60%), but low specificity (25.45%). The reported AUC of 0.8089 confirms that the tree does a good job distinguishing high‐QoL from low‐QoL, even though it more often overpredicts high‐QoL than low‐QoL in the training set.



This ROC plot graphs sensitivity (vertical axis) versus 1 – specificity (horizontal axis) for different probability cutoffs on the training set. The solid blue line shows that, at a low false‐positive rate (around 0.20), sensitivity already climbs above 0.80, and by the time false positives approach 1.0, sensitivity reaches 1.0. The diagonal gray line is the no‐skill reference (AUC = 0.5). The label “Training AUC 0.81” confirms that the pruned three‐leaf tree achieves approximately 0.81 area under the ROC curve, matching the AUC reported in the fit statistics table.



After building the tree, SAS ranks the predictors by their contribution to impurity reduction. Here we can see that Traffic\_Commute\_Time\_Index is the most important variable (relative importance = 1.0000, raw importance ≈ 3.7280), followed by Pollution\_Index (relative ≈ 0.8416, raw ≈ 3.1374). Each of those appears exactly once in the final tree (Count = 1), reflecting that the two splits on Pollution\_Index and Traffic\_Commute\_Time\_Index account for most of the model’s predictive power. No other variable made it into the pruned structure.



Finally, this table measures the pruned tree’s performance on the **hold‐out test set** of 448 observations. Among 29 actual low‐QoL countries, the tree correctly predicts 12 as low‐QoL and misclassifies 17 as high‐QoL (58.62% false positive rate, identical to training). Among 419 actual high‐QoL countries, the tree correctly classifies 417 and misclassifies only 2 as low‐QoL (0.48% false negative rate). Overall, the test‐set misclassification rate is 0.0430, sensitivity is 0.9952, specificity is 0.4138 and AUC remains approximately 0.81, closely matching training performance and indicating that the pruned three‐leaf tree generalizes well.