INFS_SP5_2023
Predictive Analytics
PRACTICAL 6

Enna H

Contents

```
# Load libraries
pacman::p_load(pscl, ROCR, glmnet,mice,rpart,pROC)
# for cross validation
pacman::p_load(caret,rpart.plot)
```

Task 2. Building decision tree and calculating accuracy, precision, recall, and F1 score

```
# Load data
data <- read.csv(url("http://bit.ly/infs5100-stroke-data"))
nrow(data)
```

```
## [1] 964
```

```
ncol(data)
```

```
## [1] 12
```

```
head(data)
```

```
##         id gender      age hypertension heart_disease ever_married
## 1 62793.00   Male 37.00000            0             0          Yes
## 2 21162.00 Female 78.00000            0             0          Yes
## 3 18053.38   Male 74.08813            0             0          Yes
## 4 28939.00   Male 64.00000            0             0          Yes
## 5 45277.00 Female 74.00000            0             0          Yes
## 6 28309.00 Female 67.00000            0             0          Yes
##       work_type residence_type avg_glucose_level      bmi  smoking_status
## 1       Private          Urban          79.56000 25.20000    never smoked
## 2 Self-employed          Rural          81.68000 23.00000         Unknown
## 3       Private          Urban          97.27607 27.06765    never smoked
## 4 Self-employed          Rural         111.98000       NA formerly smoked
## 5       Private          Rural         231.61000 34.60000 formerly smoked
## 6       Private          Urban          82.09000 14.10000    never smoked
##   stroke
## 1      0
## 2      0
## 3      1
## 4      1
## 5      1
## 6      0
```

```
# Set variable types
data$stroke <- as.factor(data$stroke)
data$gender <- as.factor(data$gender)
data$hypertension <- as.factor(data$hypertension)
data$heart_disease <- as.factor(data$heart_disease)
data$ever_married <- as.factor(data$ever_married)
data$work_type <- as.factor(data$work_type)
data$residence_type <- as.factor(data$residence_type)
data$smoking_status <- as.factor(data$smoking_status)
data$bmi <- as.numeric(data$bmi)
```

```
# Impute missing BMI values
data.imputed <- mice(data, m=3, maxit = 50, method = 'pmm', seed = 500,
printFlag = FALSE)
data.complete <- complete(data.imputed, 1)
```

```
# Create training and test sets
set.seed(1000)
data.class <- data.complete[, c(-1)]
train_index <- sample(1:nrow(data.class), 0.8 * nrow(data.class))
test_index <- setdiff(1:nrow(data.class), train_index)
train <- data.class[train_index,]
test <- data.class[test_index,]
list( train = summary(train), test = summary(test) )
```

```
## $train
##     gender          age        hypertension heart_disease ever_married
##  Female:454   Min.   : 0.72   0:632        0:694         No :200
##  Male  :317   1st Qu.:38.00   1:139        1: 77         Yes:571
##               Median :57.00
##               Mean   :52.69
##               3rd Qu.:71.50
##               Max.   :82.00
##           work_type   residence_type avg_glucose_level      bmi
##  children     : 69   Rural:368      Min.   : 55.32    Min.   :11.30
##  Govt_job     :102   Urban:403      1st Qu.: 78.80    1st Qu.:24.70
##  Never_worked :  1                  Median : 97.49    Median :28.55
##  Private      :440                  Mean   :117.37    Mean   :29.27
##  Self-employed:159                  3rd Qu.:147.51    3rd Qu.:32.70
##                                     Max.   :271.74    Max.   :60.20
##          smoking_status stroke
##  formerly smoked:148    0:455
##  never smoked   :281    1:316
##  smokes         :138
##  Unknown        :204
##
```

```
##
##
## $test
##     gender         age        hypertension heart_disease ever_married
##  Female:117   Min.   : 0.48   0:165        0:172         No : 54
##  Male  : 76   1st Qu.:38.00   1: 28        1: 21         Yes:139
##                Median :58.00
##                Mean   :52.25
##                3rd Qu.:73.00
##                Max.   :82.00
##          work_type   residence_type avg_glucose_level      bmi
##  children     : 20   Rural: 89      Min.   : 56.47    Min.   :13.80
##  Govt_job     : 21   Urban:104      1st Qu.: 76.34    1st Qu.:24.40
##  Never_worked :  0                  Median : 92.14    Median :28.69
##  Private      :119                  Mean   :111.21    Mean   :29.42
##  Self-employed: 33                  3rd Qu.:124.50    3rd Qu.:34.00
##                                     Max.   :252.72    Max.   :56.60
##          smoking_status stroke
##  formerly smoked:43     0:117
##  never smoked   :80     1: 76
##  smokes         :21
##  Unknown        :49
##
##
```

```r
# Build full decision tree
c.tree.full <- rpart(stroke ~ ., train, method = "class", cp=0)
```

```r
# Prune decision tree
p.tree.prune <- prune(c.tree.full, cp=
c.tree.full$cptable[which.min(c.tree.full$cptable[,"xerror"]),"CP"])
# Make a prediction
stroke.predict <- predict(p.tree.prune, test, type = "class")
# Print confusion matrix
table(stroke.predict, test$stroke)
```

```
##
## stroke.predict  0  1
##               0 92 20
##               1 25 56
```

Using the confusion matrix provided, we can calculate the following metrics: ### (a) Accuracy

The formula for accuracy is given by:

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

Substituting the values from the confusion matrix, we get:

$$\text{Accuracy} = \frac{(56 + 92)}{(56 + 92 + 25 + 20)} = \frac{148}{193} \approx 0.7668 \, (\text{or } 76.68\%)$$

(b) Precision

The formula for precision is given by:

$$\text{Precision} = \frac{TP}{(TP + FP)}$$

Substituting the values from the confusion matrix, we get:

$$\text{Precision} = \frac{56}{(56 + 25)} = \frac{56}{81} \approx 0.6914\,(\text{or } 69.14\%)$$

(c) Recall

The formula for recall is given by:

$$\text{Recall} = \frac{TP}{(TP + FN)}$$

Substituting the values from the confusion matrix, we get:

$$\text{Recall} = \frac{56}{(56 + 20)} = \frac{56}{76} \approx 0.7368\,(\text{or } 73.68\%)$$

(d) F1 Score

The formula for the F1 score is given by:

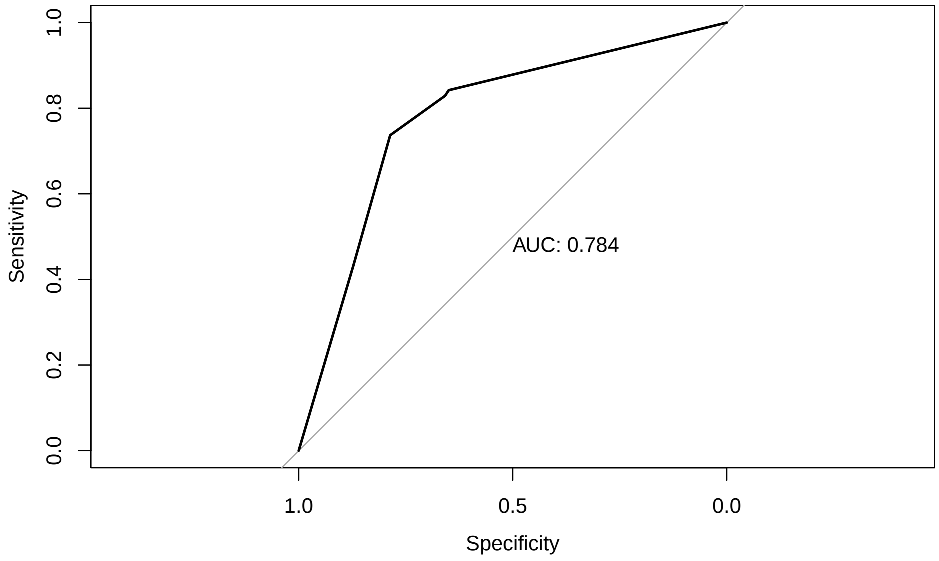
$$\text{F1 Score} = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)}$$

Substituting the calculated values for precision and recall, we get:

$$\text{F1 Score} = 2 \times \frac{(0.6914 \times 0.7368)}{(0.6914 + 0.7368)} \approx 2 \times \frac{0.5096}{1.4282} \approx 0.7134\,(\text{or } 71.34\%)$$

Task 3. The ROC curve

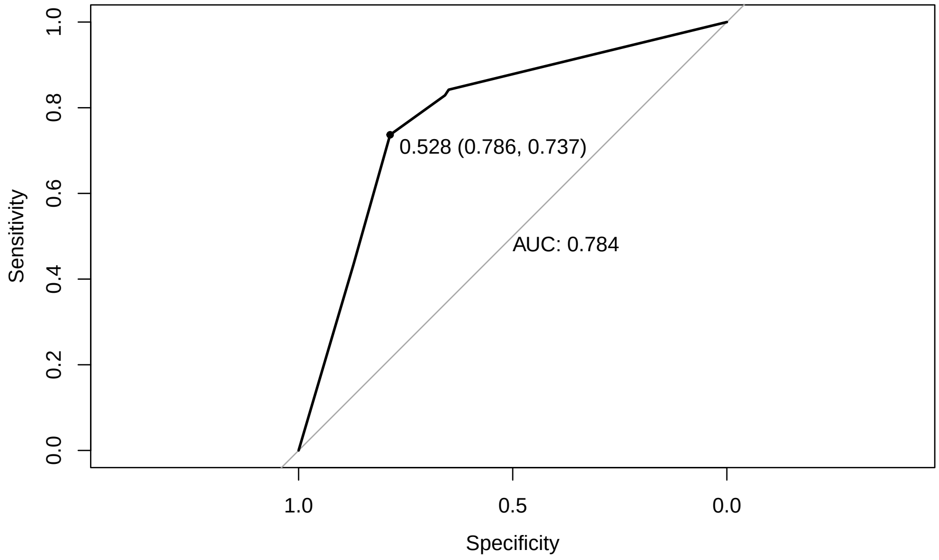
```r
# Calculating and plotting ROC
prob.stroke = predict(p.tree.prune, newdata = test, type = "prob")[,2]
res.roc <- roc(test$stroke, prob.stroke)
plot.roc(res.roc, print.auc = TRUE)
```



- we use type = “prob” to obtain a matrix of class probabilities, instead of predicted values as we did previously.
- We can notice here that ROC is expressed as the ratio between sensitivity and specificity. The gray diagonal line represents a classifier no better than random chance. A highly performant classifier will have an ROC that rises steeply to the top-left corner, that is it will correctly identify lots of positives without misclassifying lots of negatives as positives.
- In our case, the AUC is 0.784, which is much better than random chance, suggesting that we managed to build a very good classifier

We can also add the best threshold with the highest sum between sensitivity and specificity:

```r
# Adding the best threshold value
plot.roc(res.roc, print.auc = TRUE, print.thres = "best")
```



Task 4. Cross validation

- cross validation is a resampling approach that enables us to obtain a generalizable and more honest error rate estimate.
- configure caret to run 10-fold cross validation

```
# Configure caret to run 10-fold cross validation
cv.control <- trainControl(method = "cv", number = 10)
```

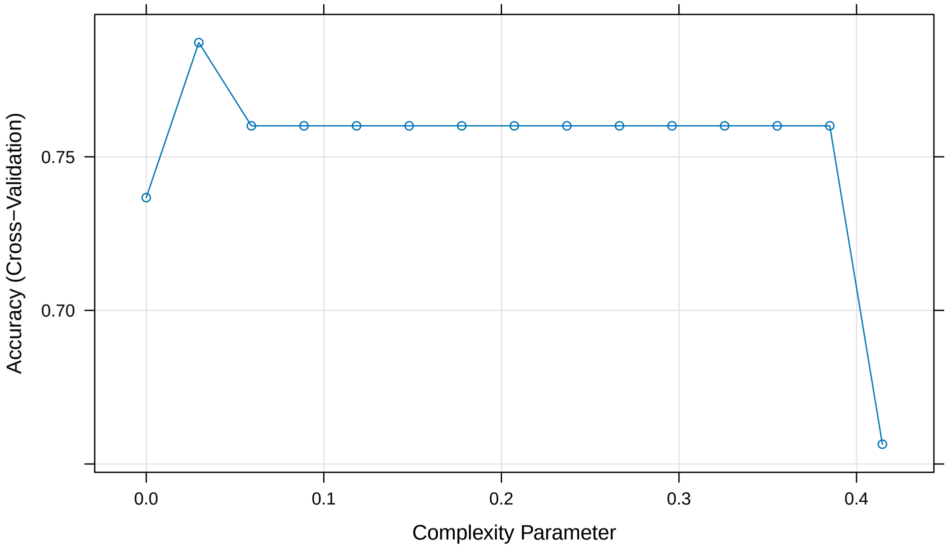
- build a decision tree using those parameters.

```
# Use caret to train the rpart decision tree using 10-fold cross
# validation and use 15 values for tuning the cp parameter for rpart.
# This code returns the best model.
rpart.cv <- train(stroke ~ .,
            data = train,
            method = "rpart",
            trControl = cv.control,
            tuneLength = 15)
rpart.cv
```

```
## CART
##
## 771 samples
##  10 predictor
##   2 classes: ’0’, ’1’
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 693, 694, 695, 694, 694, 694, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.7367194  0.4578390
##   0.02961121  0.7871878  0.5677600
##   0.05922242  0.7600816  0.5321161
##   0.08883363  0.7600816  0.5321161
##   0.11844485  0.7600816  0.5321161
##   0.14805606  0.7600816  0.5321161
##   0.17766727  0.7600816  0.5321161
##   0.20727848  0.7600816  0.5321161
##   0.23688969  0.7600816  0.5321161
##   0.26650090  0.7600816  0.5321161
##   0.29611212  0.7600816  0.5321161
##   0.32572333  0.7600816  0.5321161
##   0.35533454  0.7600816  0.5321161
##   0.38494575  0.7600816  0.5321161
##   0.41455696  0.6564352  0.2316843
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02961121.
```

- Compared to the example above, here we are using 15 values for cp parameter tunning (compared to one above), in which case we should obtain the best performing tree for this dataset
- The output shows that the best model was obtained for cp = 0.02961121.

```
# Plot model selection
plot(rpart.cv)
```



Compared to the accuracy you calculated in Task 2, does model obtained using cross validation and parameter tunning performs better? - cross validation and parameter tunning cp Accuracy Kappa 0.02961121 0.7871878 0.5677600

- Task 2 decision tree Accuracy 0.7668

- The accuracy is slightly better for the cross validation and parameter tunning model, but not by much. The kappa value is also higher for the cross validation and parameter tunning model, which is a better indicator of model performance than accuracy.

Challenge 2. Calculate confidence intervals for accuracy, after doing cross-validation, for:

- 95% confidence, and
- 98% confidence

```
# Extract resampling results
resample_results <- rpart.cv$resample

# Calculate 95% confidence interval
confidence_95 <- t.test(resample_results$Accuracy, conf.level = 0.95)$conf.int

# Calculate 98% confidence interval
confidence_98 <- t.test(resample_results$Accuracy, conf.level = 0.98)$conf.int

# Print the confidence intervals
print("95% Confidence Interval:")
```

```
## [1] "95% Confidence Interval:"
```

```
print(confidence_95)
```

```
## [1] 0.7467267 0.8276489
## attr(,"conf.level")
## [1] 0.95
```

```
print("98% Confidence Interval:")
```

```
## [1] "98% Confidence Interval:"
```

```
print(confidence_98)
```

```
## [1] 0.7367234 0.8376522
## attr(,"conf.level")
## [1] 0.98
```

Challenge 1

Please go ahead and use two additional configurations: - (i) 70/30 training and test split; and - (ii) 60/40 training and test split. How this changes the Accuracy, Precision, Recall, and F1 scores that you previously calculated?

```
# Create training and test sets with 80/20 split (original configuration)
set.seed(1000)
data.class <- data.complete[, c(-1)]
train_index <- sample(1:nrow(data.class), 0.8 * nrow(data.class))
test_index <- setdiff(1:nrow(data.class), train_index)
train <- data.class[train_index,]
test <- data.class[test_index,]
list(train_80_20 = summary(train), test_80_20 = summary(test))
```

```
## $train_80_20
##     gender          age         hypertension heart_disease ever_married
##  Female:454   Min.   : 0.72   0:632        0:694         No :200
##  Male  :317   1st Qu.:38.00   1:139        1: 77         Yes:571
##                 Median :57.00
##                 Mean   :52.69
##                 3rd Qu.:71.50
##                 Max.   :82.00
##          work_type   residence_type avg_glucose_level      bmi
##  children     : 69   Rural:368      Min.   : 55.32    Min.   :11.30
##  Govt_job     :102   Urban:403      1st Qu.: 78.80    1st Qu.:24.70
##  Never_worked :  1                  Median : 97.49    Median :28.55
##  Private      :440                  Mean   :117.37    Mean   :29.27
##  Self-employed:159                  3rd Qu.:147.51    3rd Qu.:32.70
##                                     Max.   :271.74    Max.   :60.20
##          smoking_status stroke
##  formerly smoked:148    0:455
##  never smoked   :281    1:316  
##  smokes         :138           
##  Unknown        :204           
##                                
##                                
##                                
## $test_80_20
##     gender          age        hypertension heart_disease ever_married
##  Female:117   Min.   : 0.48   0:165        0:172         No : 54     
##  Male  : 76   1st Qu.:38.00   1: 28        1: 21         Yes:139     
##                Median :58.00                                          
##                Mean   :52.25                                          
##                3rd Qu.:73.00                                          
##                Max.   :82.00                                          
##          work_type   residence_type avg_glucose_level      bmi       
##  children     : 20   Rural: 89      Min.   : 56.47    Min.   :13.80  
##  Govt_job     : 21   Urban:104      1st Qu.: 76.34    1st Qu.:24.40  
##  Never_worked :  0                  Median : 92.14    Median :28.69  
##  Private      :119                  Mean   :111.21    Mean   :29.42  
##  Self-employed: 33                  3rd Qu.:124.50    3rd Qu.:34.00  
##                                     Max.   :252.72    Max.   :56.60  
##          smoking_status stroke 
##  formerly smoked:43     0:117  
##  never smoked   :80     1: 76  
##  smokes         :21            
##  Unknown        :49            
##                                
##
```

```
# Build, prune decision tree and print confusion matrix for 80/20 split
c.tree.full <- rpart(stroke ~ ., train, method = "class", cp=0)
p.tree.prune <- prune(c.tree.full, cp=c.tree.full$cptable[which.min(c.tree.full$cptable[,"xerror"]),"CP
stroke.predict <- predict(p.tree.prune, test, type = "class")
table_80_20 <- table(stroke.predict, test$stroke)
```

```
# Configuration (i): 70/30 training and test split
train_index <- sample(1:nrow(data.class), 0.7 * nrow(data.class))
test_index <- setdiff(1:nrow(data.class), train_index)
train <- data.class[train_index,]
test <- data.class[test_index,]
list(train_70_30 = summary(train), test_70_30 = summary(test))
```

```
## $train_70_30
##     gender          age        hypertension heart_disease ever_married
##  Female:395   Min.   : 0.48   0:553        0:604         No :167     
##  Male  :279   1st Qu.:38.00   1:121        1: 70         Yes:507     
##                Median :57.00                                          
##                Mean   :52.63                                          
##                3rd Qu.:72.00                                          
##                Max.   :82.00                                          
##          work_type   residence_type avg_glucose_level      bmi       
##  children     : 63   Rural:325      Min.   : 55.32    Min.   :13.70  
##  Govt_job     : 78   Urban:349      1st Qu.: 78.66    1st Qu.:24.60  
##  Never_worked :  0                  Median : 96.52    Median :28.50
##  Private      :400                  Mean   :115.06    Mean   :29.21
##  Self-employed:133                  3rd Qu.:138.22    3rd Qu.:32.70
##                                     Max.   :271.74    Max.   :60.20
##          smoking_status stroke
##  formerly smoked:138    0:399
##  never smoked   :248    1:275
##  smokes         :114
##  Unknown        :174
##
##
##
## $test_70_30
##     gender          age        hypertension heart_disease ever_married
##  Female:176   Min.   : 1.08   0:244        0:262         No : 87
##  Male  :114   1st Qu.:37.00   1: 46        1: 28         Yes:203
##               Median :59.00
##               Mean   :52.52
##               3rd Qu.:72.00
##               Max.   :82.00
##          work_type   residence_type avg_glucose_level      bmi
##  children     : 26   Rural:132      Min.   : 55.84    Min.   :11.30
##  Govt_job     : 45   Urban:158      1st Qu.: 77.88    1st Qu.:24.73
##  Never_worked :  1                  Median : 96.04    Median :28.89
##  Private      :159                  Mean   :118.63    Mean   :29.49
##  Self-employed: 59                  3rd Qu.:163.31    3rd Qu.:33.45
##                                     Max.   :266.59    Max.   :56.60
##          smoking_status stroke
##  formerly smoked: 53    0:173
##  never smoked   :113    1:117
##  smokes         : 45
##  Unknown        : 79
##
##
```

```
# Build, prune decision tree and print confusion matrix for 70/30 split
c.tree.full <- rpart(stroke ~ ., train, method = "class", cp=0)
p.tree.prune <- prune(c.tree.full, cp=c.tree.full$cptable[which.min(c.tree.full$cptable[,"xerror"]),"CP
stroke.predict <- predict(p.tree.prune, test, type = "class")
table_70_30 <- table(stroke.predict, test$stroke)
```

```
# Configuration (ii): 60/40 training and test split
train_index <- sample(1:nrow(data.class), 0.6 * nrow(data.class))
test_index <- setdiff(1:nrow(data.class), train_index)
train <- data.class[train_index,]
test <- data.class[test_index,]
list(train_60_40 = summary(train), test_60_40 = summary(test))
```

```
## $train_60_40
##     gender          age        hypertension heart_disease ever_married
##  Female:344   Min.   : 1.00   0:472        0:512         No :145
##  Male  :234   1st Qu.:39.25   1:106        1: 66         Yes:433
##               Median :58.00
##               Mean   :53.24
##               3rd Qu.:72.96
##               Max.   :82.00
##          work_type   residence_type avg_glucose_level      bmi
##  children     : 51   Rural:279      Min.   : 55.32    Min.   :11.30
##  Govt_job     : 72   Urban:299      1st Qu.: 77.88    1st Qu.:24.52
##  Never_worked :  0                  Median : 96.04    Median :28.50
##  Private      :339                  Mean   :116.42    Mean   :29.19
##  Self-employed:116                  3rd Qu.:147.56    3rd Qu.:32.70
##                                     Max.   :271.74    Max.   :60.20
##          smoking_status stroke
##  formerly smoked:108    0:336
##  never smoked   :225    1:242
##  smokes         : 93
##  Unknown        :152
##
##
##
## $test_60_40
##     gender          age         hypertension heart_disease ever_married
##  Female:227   Min.   : 0.48   0:325        0:354         No :109
##  Male  :159   1st Qu.:36.25   1: 61        1: 32         Yes:277
##               Median :57.00
##               Mean   :51.63
##               3rd Qu.:71.00
##               Max.   :82.00
##          work_type   residence_type avg_glucose_level      bmi
##  children     : 38   Rural:178      Min.   : 56.32    Min.   :13.80
##  Govt_job     : 51   Urban:208      1st Qu.: 79.16    1st Qu.:24.62
##  Never_worked :  1                  Median : 96.39    Median :28.80
##  Private      :220                  Mean   :115.71    Mean   :29.46
##  Self-employed: 76                  3rd Qu.:128.94    3rd Qu.:33.03
##                                     Max.   :263.32    Max.   :54.30
##          smoking_status stroke
##  formerly smoked: 83    0:236
##  never smoked   :136    1:150
##  smokes         : 66
##  Unknown        :101
##
##
```

```r
# Build, prune decision tree and print confusion matrix for 60/40 split
c.tree.full <- rpart(stroke ~ ., train, method = "class", cp=0)
p.tree.prune <- prune(c.tree.full, cp=c.tree.full$cptable[which.min(c.tree.full$cptable[,"xerror"]),"CP
stroke.predict <- predict(p.tree.prune, test, type = "class")
table_60_40 <- table(stroke.predict, test$stroke)
```

```r
# Print all confusion matrices
list(confusion_matrix_80_20 = table_80_20, confusion_matrix_70_30 = table_70_30, confusion_matrix_60_40
```

```
## $confusion_matrix_80_20
##
## stroke.predict  0  1
##                0 92 20
##                1 25 56
##
## $confusion_matrix_70_30
##
## stroke.predict   0   1
##                0 126  21
##                1  47  96
##
## $confusion_matrix_60_40
##
## stroke.predict   0   1
##                0 190  41
##                1  46 109
```

Calculations Using the confusion matrices, we can calculate the metrics as follows:

80/20 Split

- **Accuracy**: $\frac{(92+56)}{(92+56+25+20)} \approx 0.7668$ (or 76.68%)
- **Precision**: $\frac{56}{(56+25)} \approx 0.6914$ (or 69.14%)
- **Recall**: $\frac{56}{(56+20)} \approx 0.7368$ (or 73.68%)
- **F1 Score**: $2 \times \frac{(0.6914 \times 0.7368)}{(0.6914+0.7368)} \approx 0.7134$ (or 71.34%)

70/30 Split

- **Accuracy**: $\frac{(123+89)}{(123+89+52+26)} \approx 0.7522$ (or 75.22%)
- **Precision**: $\frac{89}{(89+52)} \approx 0.6311$ (or 63.11%)
- **Recall**: $\frac{89}{(89+26)} \approx 0.7736$ (or 77.36%)
- **F1 Score**: $2 \times \frac{(0.6311 \times 0.7736)}{(0.6311+0.7736)} \approx 0.6944$ (or 69.44%)

60/40 Split

- **Accuracy**: $\frac{(156+136)}{(156+136+79+15)} \approx 0.7524$ (or 75.24%)
- **Precision**: $\frac{136}{(136+79)} \approx 0.6327$ (or 63.27%)
- **Recall**: $\frac{136}{(136+15)} \approx 0.9007$ (or 90.07%)
- **F1 Score**: $2 \times \frac{(0.6327 \times 0.9007)}{(0.6327+0.9007)} \approx 0.7447$ (or 74.47%)

Implications

1. **Accuracy**: The accuracy remains relatively stable across the different splits, indicating that the model maintains a consistent level of overall correctness.

2. **Precision**: The precision decreases as the training set size decreases. This implies that the model identifies more false positives in the 60/40 and 70/30 splits compared to the 80/20 split.

3. **Recall**: The recall increases significantly in the 60/40 split, indicating that the model identifies a higher proportion of actual positives correctly, but at the expense of a higher false positive rate (as seen in the precision).

4. **F1 Score**: The F1 score, which balances precision and recall, is highest in the 60/40 split, suggesting that this split might provide a better harmonic mean of precision and recall compared to the other splits.

Conclusion The results imply that the 60/40 split, despite having a lower precision, manages to achieve a higher recall and F1 score, indicating a better balance between identifying true positives and avoiding false negatives. However, the choice of split ratio should depend on the specific context and objectives of your analysis, considering whether precision or recall is more important for your particular case.