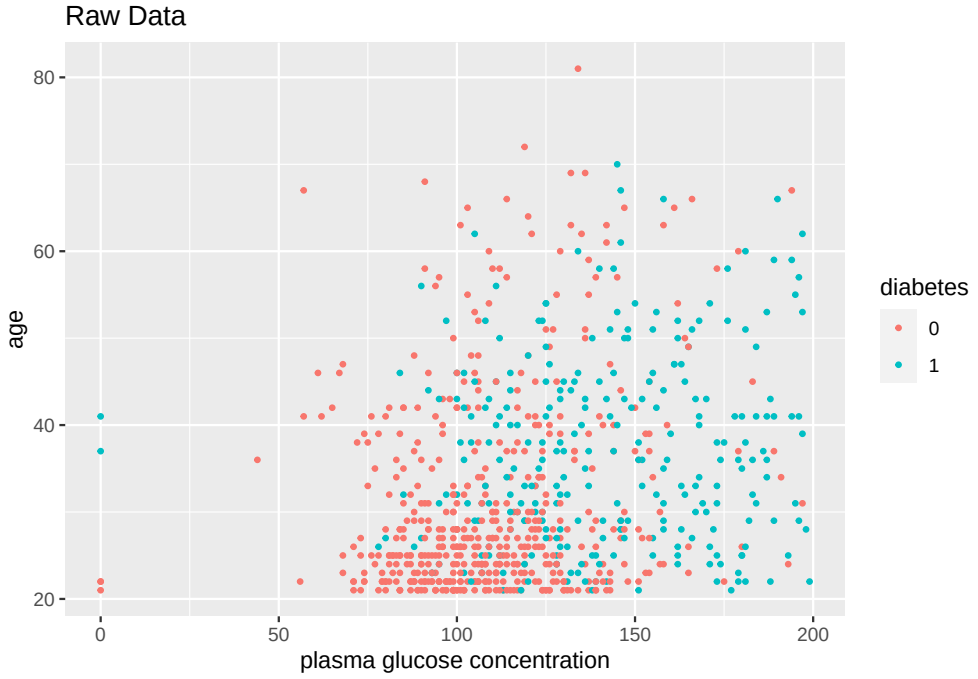
732A99 Machine Learning | Assignment 3

Group 8

18 November, 2021

Q1

Although elements of clustering exist in the bottom middle part of the plot, it is hard to draw a distinct decision boundary. The problem is that this classification problem is both *imbalanced* and *asymmetric*. *Imbalance* is caused by unequal distribution of *positive* and *negative* responses in the data frame. From the responses, it can be calculated that 65% of results from diabetes has "YES" (or 1 for positive) status while only 35% has "NO" (or 0 negative) status. In addition, this is a medical diagnosis classification problem meaning that *false negatives* are considered to be more severe than *false positives*. This makes the problem *asymetric*. Results of applied logistic regression must be analyzed from confusion matrix, recall, precision and F-scores.



Q2

Estimated parameters from the model are provided below:
Intercept, θ_0 = -5.912
Plasma Glucose Concentration, θ_1 = 0.036
Age, θ_2 = 0.025

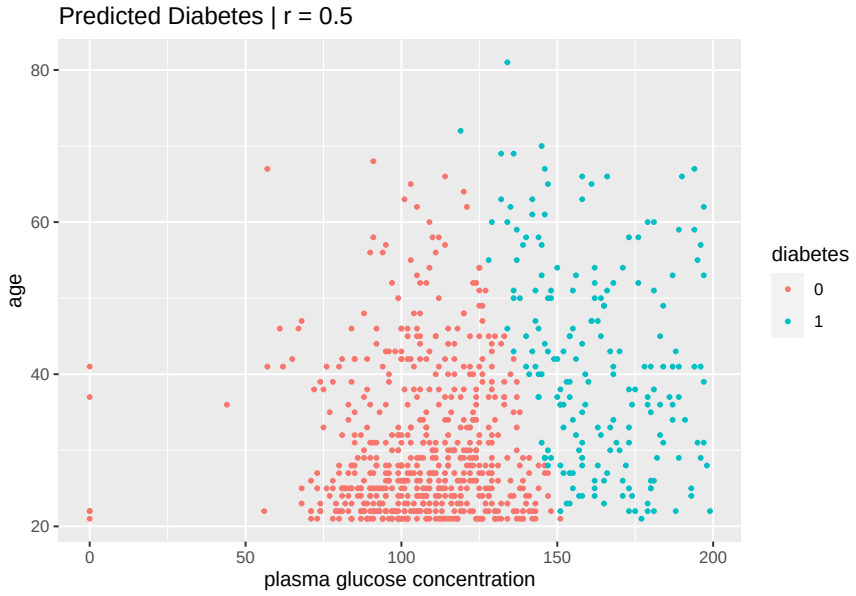
Probabilistic equation for **logistic regression** is:

$$p(y = m|x, \theta) = \frac{1}{1 + exp(-(\theta_0 + x_1\theta_1 + x_2\theta_2))}$$

p(y) - probability of y = m
m - [0,1] where 0 stands for *"no"* and 1 stands for *"yes"* status of diabete
x_1 - Plasma Glucose Concentration
x_2 - Age
θ_i - estimated parameter for $i = 0, 1, 2$

Substituting:

$$p(y) = \frac{1}{1 + exp(-(-5.912 + 0.036x_1 + 0.025x_2))}$$



Training misclassification rate = 0.263
This is a high value meaning that more than a quarter of the data has been misclassified. However, for both *imbalanced* and *asymetric* scenarios *misclassification error* is a poor indicator of the model performance. Thus, to make a better assessment of the model, it is important to calculate and analyze *confusion matrix, recall, precision and F-score* parameters for $r = 0.5$:

```
## Confusion Matrix | r = 0.5
```

```
##           real
## predicted   0   1
##         0 436 138
##         1  64 130
```

```
## misclassification_rate =  0.2630208
```

```
## F_score =  0.5134281
```

```
## recall =  0.4850746
```

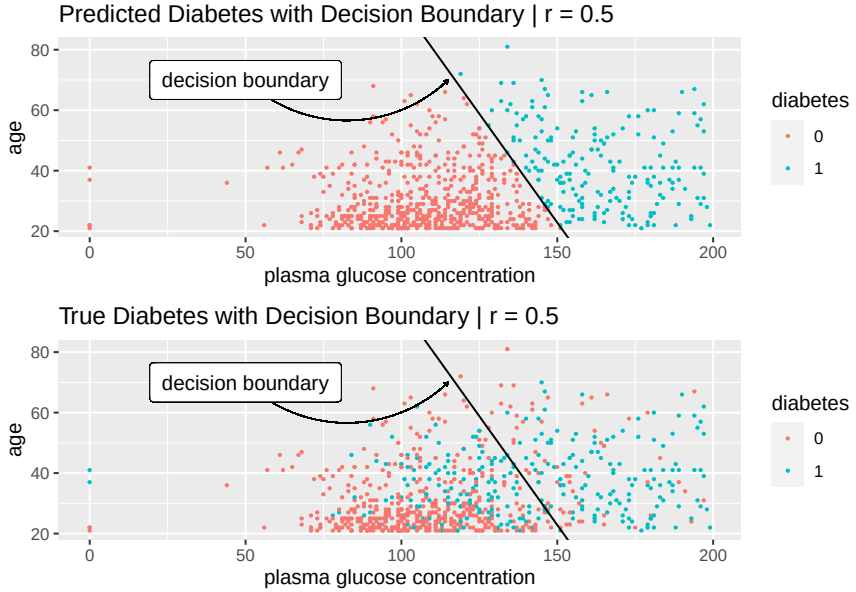
```
## precision =  0.6701031
```

Confusion matrix is a very clear example that model performs poorly on *false negatives*, in other words, only (roughly speaking) half of the people with "YES" diabete status have been successfully predicted by the model. Since the model is *asymetric*, *false negatives* deserve more importance, thus, to estimate F-score, $beta = 2$ is considered. Nevertheless, recall and F-score values are still around 0.5. It makes the model with $r = 0.5$ threshold unreliable in this situation.

Q3

To derive equation of decision boundary, consider:
$x_2 = x_1 k + b$, $k = slope$ and $b = intercept$
$p(y) = 0.5$ when $r = 0.5$
As mentioned earlier, for the logistic regression:
$p(y) = \frac{1}{1+exp(-(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2))} = 0.5$
$exp(-(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2)) = 1$ and $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 = 0$
For *intercept* b, substitute $x_2 = x_1 k + b$, $x_1 = 0$, $x_0 = 1$ into above equation to get:
$\theta_0 + \theta_2 b = 0$

Since this is an equation of line, *intercept* $b = -\frac{\theta_0}{\theta_2}$ and *slope* $k = -\frac{\theta_1}{\theta_2}$. Note that, this is only valid for two-dimensional problem (two features x_1 and x_2). As it will be seen later, the line is no more linear when more than two features are used for the logistic regression.



The decision boundary line separates data quite good. However, it behaves good for *predicted responses* not *real responses* ! As it can be seen above, for *real responses* decision boundary performs poorly.

Q4.

For $r = 0.2$ and $r = 0.8$, b *intercept* in the equation of decision boundary will change. However, k *slope* value stays the same since the problem is still two-dimensional (only two features are included into the model) and line remains linear. The slope could have changed, if the model has been run with other two features since parameters would be different in that case.
For $r = 0.2$, $b = \frac{-ln(4)-\theta_0}{\theta_2}$ and for $r = 0.8$, $b = \frac{ln4-\theta_0}{\theta_2}$

Below are a set of parameters to analyze the model performance:

```
##
##
## Confusion Matrix | r = 0.2

##           real
## predicted   0   1
##         0 238  24
##         1 262 244

## misclassification_rate =  0.3723958

## F_score =  0.7731305

## recall =  0.9104478

## precision =  0.4822134

##
##
## Confusion Matrix | r = 0.8

##           real
## predicted   0   1
##         0 490 232
##         1  10  36

## misclassification_rate =  0.3151042

## F_score =  0.1610018
```

```
## recall =  0.1343284
```

```
## precision =  0.7826087
```

It is obvious that $r = 0.8$ is the worst case scenario where the model has failed to correctly predict 87% of patients with "YES" diabete status. This is also supported by a very low F-score. It should be mentioned that in scenarios where *false negatives* are prioritized over *false positives*, recall is given a greater priority than precision. For this case, recall also has a very low value (must be close to 1).
For $r = 0.2$, situation is much better for *false negatives*, in other words, 91% of patients with "YES" diabete status has been predicted correctly. Model also has a relatively high F-score and recall. The problem with this scenario is that the model performs poorly for patients with "NO" diabete status or *true negatives*. Here, looking at the confusion matrix, it can be said that roughly half of the patients with "NO" diabete status has been predicted incorrectly. As it has been mentioned earlier, since this is a medical experiment involving patient diagnosis, one must aim for minimizing amount of *false negatives*. Thus, the model with $r = 0.2$ can be considered viable. Despite having a higher misclassification rate, the model with threshold $r = 0.2$ performs better that the model with $r = 0.5$.

Q5.

After application of basis function expansion, **training misclassification error** decreases to 0.245. As in the previous scenarios, a set of the model performance parameters must be analyzed:

```
##
##
## Confusion Matrix for BFE | r = 0.5
```

```
##           real
## predicted   0   1
##         0 433 121
##         1  67 147
```

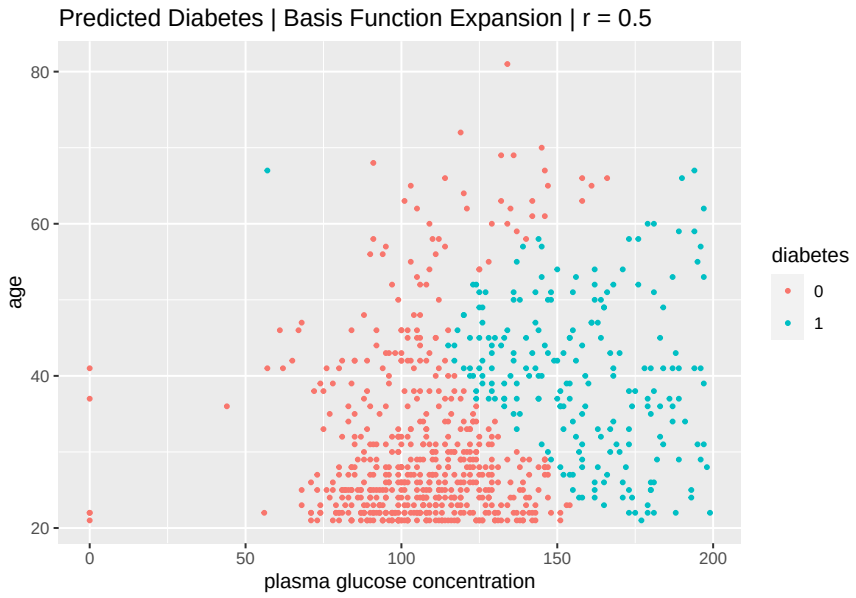
```
## misclassification_rate =  0.2447917
```

```
## F_score =  0.5715397
```

```
## recall =  0.5485075
```

```
## precision =  0.6869159
```

Here, threshold is the same $r = 0.5$, however, the number of features has been increased to 7. The model performance is very similar to the one with only two features (discussed in Q2). Slight improvements in F-score, recall and, as a result, in decreasing number of *false negatives* can be seen. However, the results are hardly feasible for the medical diagnosis experiment.



The model is no longer two-dimensional and, thus, the decision boundary is hard to define from two features. However, since x_1 and x_2 still have a larger impact in comparison with other features (much lower p-values), the decision boundary can be approximated by plotting only these two features as in the previous scenarios. If drawn, a decision boundary will have something close to the shape of an inverted hyperbola.

To sum up the analysis of all previous models, the model with $r = 0.2$ threshold appears to be the most appropriate for the given experiment. The important point is that only training data has been used for the analysis of the logistic regression model. To generalize beyond the training data, the model has to demonstrate decent performance on the test data.

Appendix

```
# Assignment 3

 library(ggplot2)
 library(gridExtra)

df<-read.csv("pima-indians-diabetes.csv", header=FALSE)

colnames(df)<-c("not_pregnant", "plasma_glucose_concentration",
                "diastolic_blood_pressure", "triceps_skinfold_thickness",
                "2h_serum_insulin", "body_mass_index",
                "diabetes_pedigree_function", "age", "diabetes")

# 1._____________________________________

df$diabetes<-as.factor(df$diabetes)

pl<-ggplot(df)+
  geom_point(aes(x=plasma_glucose_concentration, y=age, color=diabetes),size=0.8)+
  ggtitle("Raw Data")+xlab("plasma glucose concentration")+ylab("age")
```

```r
#plot(pl)

# 2._____________________________________

# run logistic regression model

logreg<-glm(formula = diabetes ~ plasma_glucose_concentration + age,
                    data = df,
                    family = "binomial")

# use your model to predict responses for the whole data frame

predicted_diabetes<-predict(logreg, newdata=df, type='response')

r<-0.5   # define your threshold to classify probabilities

r05_predicted_diabetes<-as.factor(ifelse(predicted_diabetes >=r, 1, 0))

# calculate misclassification error

misclass_error<-mean(r05_predicted_diabetes!=df$diabetes)

cat("Misclassification error = ", misclass_error,"\n")
```

```
## Misclassification error =  0.2630208
```

```r
# attach your predicted values to original data frame

new_df<-cbind(df, r05_predicted_diabetes)

# 3._____________________________________________

tetta = coefficients(logreg)
intercept = -tetta[1]/tetta[3]
slope = -tetta[2]/tetta[3]

pl1<-ggplot(data=new_df, aes(x=plasma_glucose_concentration, y=age, color = r05_predicted_diabetes))+
  geom_point(size=0.4)+
  geom_abline(aes(slope = slope, intercept=intercept), color = 'black')+
  geom_curve(x = 50, y = 70, xend =115 , yend =70,
             arrow = arrow(length = unit(0.02, "npc"), type='closed'),
             size=0.3,
             curvature = 0.4,
             color='black',
             ncp = 40)+
  geom_label(label="decision boundary",
             label.size = 0.1,
             label.padding = unit(0.45, "lines"),
             x=50,
             y=70,
             color='black')+
  ggtitle("Predicted Diabetes with Decision Boundary | r = 0.5 ")+
  labs(color = "diabetes")+xlab("plasma glucose concentration")
```

```r
#plot(pl1)

# 4.______________________________________________

# r = 0.2

r=0.2

r02_predicted_diabetes<-as.factor(ifelse(predicted_diabetes > r, 1, 0))

new_df<-cbind(new_df, r02_predicted_diabetes)

tetta = coefficients(logreg)
intercept_02 = (-log(4)-tetta[1])/tetta[3]
slope = -tetta[2]/tetta[3]

pl2<-ggplot(data=new_df, aes(x=plasma_glucose_concentration, y=age, color = r02_predicted_diabetes))+
  geom_point(size=0.4)+
  geom_abline(aes(slope = slope, intercept=intercept_02), color = 'black')+
  geom_curve(x = 30, y = 70, xend =80 , yend =60,
             arrow = arrow(length = unit(0.02, "npc"),type='closed'),
             size=0.3,
             curvature = 0.4,
             color='black',
             ncp=40)+
  geom_label(label="decision boundary",
             label.size = 0.1,
             label.padding = unit(0.45, "lines"),
             x=30,
             y=70,
             color='black')+
  ggtitle("Predicted Diabetes with Decision Boundary", subtitle = "r = 0.2")+
  labs(color = "diabetes")+xlab("plasma glucose concentration")


#plot(pl2)

# r = 0.8

r=0.8

r08_predicted_diabetes<-as.factor(ifelse(predicted_diabetes > r, 1, 0))

new_df<-cbind(new_df, r08_predicted_diabetes)

tetta = coefficients(logreg)
intercept_08 = (log(4)-tetta[1])/tetta[3]
slope = -tetta[2]/tetta[3]

pl3<-ggplot(data=new_df, aes(x=plasma_glucose_concentration, y=age, color = r08_predicted_diabetes))+
  geom_point(size=0.4)+
  geom_abline(aes(slope = slope, intercept=intercept_08), color = 'black')+
  geom_curve(x = 80, y = 70, xend =150 , yend =70,
```

```
              arrow = arrow(length = unit(0.02, "npc"),type='closed'),
              size=0.3,
              curvature = 0.4,
              color='black',
              ncp=40)+
  geom_label(label="decision boundary",
              label.size = 0.1,
              label.padding = unit(0.45, "lines"),
              x=80,
              y=70,
              color='black')+
  ggtitle(label=NULL, subtitle = "r = 0.8")+
  labs(color = "diabetes")

#plot(pl3)

# 5._____________________________________________

# Basis function expansion

z1<-(df$plasma_glucose_concentration)^4
z2<-(df$plasma_glucose_concentration)^3*(df$age)
z3<-(df$plasma_glucose_concentration)^2*(df$age)^2
z4<-(df$plasma_glucose_concentration)*(df$age)^3
z5<-(df$age)^4

z_df<-cbind(df,z1,z2,z3,z4,z5)

z_logreg<-glm(formula = diabetes ~
              plasma_glucose_concentration + age + z1 + z2 + z3 + z4 + z5,
              data = z_df,
              family = "binomial")

z_predicted_diabetes<-predict(z_logreg, newdata=z_df, type='response')

r=0.5

z_predicted_diabetes<-ifelse(z_predicted_diabetes > r, 1, 0)

z_predicted_diabetes<-as.factor(z_predicted_diabetes)
z_misclass_error<-mean(z_predicted_diabetes!=z_df$diabetes)

cat("BFE Misclassification error = ", z_misclass_error)
```

```
## BFE Misclassification error =  0.2447917
```

```
new_z_df<-cbind(z_df, z_predicted_diabetes)

pl4<-ggplot(new_z_df)+
  geom_point(aes(x=plasma_glucose_concentration, y=age, color=z_predicted_diabetes),size=0.8)+
  ggtitle("Predicted Diabetes | Basis Function Expansion | r = 0.5")+
  labs(color = "diabetes")+xlab("plasma glucose concentration")
```

```
#plot(pl4)

#grid.arrange(grobs=list(pl, pl1, pl2, pl3, pl4))


# Final Model Assessment_____________________________________

# 0 -  "NO"  | status of diabete
# 1 - "YES" | status of diabete

beta=2

# confusion matrix, recall, precision and F-score for r = 0.5

c1<-table(new_df$r05_predicted_diabetes, new_df$diabetes,dnn=c("predicted", "real"))
misclass_rate<-mean(new_df$r05_predicted_diabetes!=df$diabetes)
recall=c1[4]/(c1[4]+c1[3])
precision=c1[4]/(c1[4]+c1[2])
f_score=(1+beta^2)*precision*recall/(beta^2*precision+recall)

# confusion matrix, recall, precision and F-score for r = 0.2

c2<-table(new_df$r02_predicted_diabetes, new_df$diabetes, dnn=c("predicted", "real"))
misclass_rate<-mean(new_df$r02_predicted_diabetes!=df$diabetes)
recall=c2[4]/(c2[4]+c2[3])
precision=c2[4]/(c2[4]+c2[2])
f_score=(1+beta^2)*precision*recall/(beta^2*precision+recall)


# confusion matrix, recall, precision and F-score for r = 0.8

c3<-table(new_df$r08_predicted_diabetes, new_df$diabetes, dnn=c("predicted", "real"))
misclass_rate<-mean(new_df$r08_predicted_diabetes!=df$diabetes)
recall=c3[4]/(c3[4]+c3[3])
precision=c3[4]/(c3[4]+c3[2])
f_score=(1+beta^2)*precision*recall/(beta^2*precision+recall)


# confusion matrix, recall, precision and F-score for BFE | r = 0.5

c4<-table(new_z_df$z_predicted_diabetes, new_z_df$diabetes, dnn=c("predicted", "real"))
misclass_rate<-mean(new_z_df$z_predicted_diabetes!=new_z_df$diabetes)
recall=c4[4]/(c4[4]+c4[3])
precision=c4[4]/(c4[4]+c4[2])
f_score=(1+beta^2)*precision*recall/(beta^2*precision+recall)
```