Chapter 5
Supervised learning methods
Part II - Classification

Simon Fraser University
ECON 483
Summer 2023

Disclaimer

I do not allow this content to be published without my consent.

Classification problems

- When the variable of interest is qualitative, we attribute a number to each **attribute**, or **class**
- Example: a Yes/No becomes a 1/0 variable
- All fields involving data are interested in classification:
  - Predicting victory/estimating victory odds given some characteristics in ports
  - Predicting a disease given some symptoms
  - Predicting default given some borrower's characteristics
  - Predicting accidents for an individual buying insurance
  - Predicting purchasing/clicking behavior given some browsing history to make suggestions (YouTube, Amazon)

Classification problems

- Estimation problems aim at estimating some parameters to look at the influence of a change in a variable on the dependent variable
- Prediction problem aim at predicting whether an observation for which we observe $\boldsymbol{X_i}$ will fall in $\textbf{class} = \mathbf{1}$ or $\textbf{class} = \mathbf{0}$
- If $\boldsymbol{Y_i}$ is a binary variable, then $\mathbb{E}[\boldsymbol{Y_i}|\boldsymbol{X_i}] = \mathbf{1} \times \mathbb{P}(\boldsymbol{Y_i} = \mathbf{1}|\boldsymbol{X_i}) + \mathbf{0} \times \mathbb{P}(\boldsymbol{Y_1} = \mathbf{0}|\boldsymbol{X_i}) = \mathbb{P}(\boldsymbol{Y_i} = \mathbf{1}|\boldsymbol{X_i})$
- In words: Estimating the conditional expectation (i.e. predicting the average $\boldsymbol{Y_i}$ given $\boldsymbol{X_i}$) is the same as estimating the conditional probability that $\boldsymbol{Y_i} = \mathbf{1}$ given $\boldsymbol{X_i}$!

Outline

- **Linear regression**
- **Logistic regression**
- **K nearest neighbours**
- **Classification trees**
- **Support Vector Machines**
- Suggested reading: Chapter 4 and 9 in **ISLR**

Linear regression

The linear probability model (LPM)

- Why not consider a linear model?

$$\boldsymbol{Y_i = \beta_0 + \beta_1 X_{i,1} + \beta_2 X_{i,2} + \ldots + \beta_K X_{i,K} + u_i}$$

  where $\boldsymbol{Y_i}$ is a binary variable. Let $\boldsymbol{p(x_i) \equiv \mathbb{P}(Y_i = 1 | X_i = x_i)}$

- The OLS estimator can deliver estimates of the coefficients of interest and

$$\boldsymbol{\hat{p}(x_i) = \hat{\beta}_0 + \hat{\beta}_1 X_{i,1} + \ldots + \hat{\beta}_K X_{i,K}}$$

- If one needs to make a clear prediction, it can be a 1 if $\boldsymbol{\hat{p}(x_i) > 0.5}$ and a 0 if $\boldsymbol{\hat{p}(x_i) < 0.5}$

The linear probability model (LPM)

- That model has useful interpretations of the marginal effects of $\boldsymbol{x_{i,k}}$ on $\boldsymbol{p(x_i)}$: When $\boldsymbol{x_{i,k}}$ increases by one unit, $\boldsymbol{\hat{p}(x_i)}$ increases by $\boldsymbol{\hat{\beta}_k}$
- But it has caveats
  - Some predictions $\boldsymbol{\hat{p}(x_i)}$ can be lower than 0, or bigger than 1, which does not make sense
  - LPM always feature **heteroskedasticity**: since $\boldsymbol{Y_i}$ is either equal to 1 or 0, it is a Bernoulli variable, for which the conditional variance is $\mathbb{V}(\boldsymbol{Y_i}|\boldsymbol{X_i}) = \boldsymbol{p(x_i)(1-p(x_i))}$ and since $\boldsymbol{u_i = Y_i - \beta_0 - \beta_1 X_{i,1} - \beta_2 X_{i,2} - \ldots - \beta_K X_{i,K}}$, then $\mathbb{V}[\boldsymbol{u_i}|\boldsymbol{X_i}]$ depends on $\boldsymbol{X_i}$!!
  - How to deal with more than two classes? Coding them 1, 2 and 3 would introduce an order to them which is not necessarily the case (e.g. red/blue/green, catholic/protestant/atheist, ...)

LPM in R

- Consider the ***heart_disease*** data set
- Each observation shows values for several covariates like cholesterol, gender, blood pressure,...
- Each observation also reports whether an individual was subject to a heart disease or not
- That is what we want to predict given an individual's characteristics
- I split the sample in 2: The first one to run all the models, the second one to check the accuracy of the different predictions

LPM in R

```
# LPM
lpm <- lm(heart_disease ~ Age + sex + asymptomatic
          + typical_angina + atypical_angina + blood_sugar
          + exercise_induced_angina + MaximumHR + BP + Cholesterol,
          data = train_heart)
lpm_predictions <- predict(lpm, newdata = test_heart, type = "response")
# Making predictions based on the estimated probabilities
lpm_predictions <- ifelse(lpm_predictions > 0.5, 1, 0)
```

LPM in R

```
##
## Call:
## lm(formula = heart_disease ~ Age + sex + asymptomatic + typical_angina +
##     atypical_angina + blood_sugar + exercise_induced_angina +
##     MaximumHR + BP + Cholesterol, data = train_heart)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.07983 -0.24077 -0.01379  0.22190  0.75412
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                0.3707230  0.4060011   0.913 0.362755
## Age                        0.0041086  0.0040294   1.020 0.309650
## sex                       -0.2716464  0.0694545  -3.911 0.000143 ***
## asymptomatic               0.1943775  0.0848244   2.292 0.023425 *
## typical_angina             0.0994127  0.1343972   0.740 0.460725
## atypical_angina           -0.1414955  0.0917434  -1.542 0.125259
## blood_sugar               -0.0633272  0.0894900  -0.708 0.480342
## exercise_induced_angina    0.2826706  0.0818441   3.454 0.000732 ***
## MaximumHR                 -0.0047720  0.0016340  -2.920 0.004076 **
## BP                         0.0031109  0.0019986   1.556 0.121848
## Cholesterol                0.0004524  0.0007039   0.643 0.521427
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3791 on 140 degrees of freedom
## Multiple R-squared:  0.4616, Adjusted R-squared:  0.4232
## F-statistic: 12.01 on 10 and 140 DF,  p-value: 8.358e-15
```

Logistic regression

Logistic regression

- Rather, one can assume a different functional form for $\mathbb{P}(\boldsymbol{Y_i = 1})$ than linear so that predictions stay within range
- A very popular one is the **logistic distribution**: $\mathbb{P}[\boldsymbol{X} \leq \boldsymbol{x}] = \frac{\boldsymbol{\exp(x)}}{\boldsymbol{1+\exp(x)}} = \frac{\boldsymbol{1}}{\boldsymbol{1+\exp(-x)}}$ so that

$$\mathbb{P}(\boldsymbol{Y_i = 1 | X_i = x_i}) = \boldsymbol{p(x_i'\beta) = F(x_i'\beta) = \frac{1}{1 + \exp(-x_i'\beta)}}$$

- Since we assume a full distribution, we are looking at a **parametric** model (the first one in the course)
- It can be used both for estimation and prediction, and is also called **logit regression**

Logistic regression

- Interpretation of the coefficients is not as straightforward as a linear regression, but nevertheless useful
- A one unit increase in $\boldsymbol{X_{i,k}}$ leads to a change in $\mathbb{P}(\boldsymbol{Y_i = 1 | X_i = x_i})$ in

$$\frac{\boldsymbol{\partial F(x_i' \hat{\beta})}}{\boldsymbol{\partial x_{i,k}}} = \boldsymbol{F'(x_i' \hat{\beta}) \hat{\beta}_k} = \frac{\boldsymbol{\exp(x_i' \hat{\beta}))}}{\boldsymbol{(1 + \exp(x_i' \hat{\beta}))^2}} \boldsymbol{\hat{\beta}_k}$$

Logistic regression and maximum likelihood

- The model can be estimated via **maximum likelihood**
- Idea: Assume a distribution, and estimate its parameters by maximizing the likelihood that we observe these data under the distributional assumption
- The probability that $\boldsymbol{Y_i}$ equals 1 can be written $\boldsymbol{p(x_i'\beta)^{y_i}(1-p(x_i'\beta))^{1-y_i}}$
- If the sample is i.i.d., then the joint density of the $\boldsymbol{Y_i}$'s given the $\boldsymbol{X_i}$'s is

$$\mathcal{L}(\beta|X) = \prod_{i=1}^{n} p(x_i'\beta)^{y_i}(1-p(x_i'\beta))^{1-y_i}$$

- For some, it will be $\boldsymbol{p(x_i'\beta)}$ (if $\boldsymbol{y_i = 1}$), for others $\boldsymbol{1 - p(x_i'\beta)}$ (if $\boldsymbol{y_i = 0}$)

Maximum likelihood estimation (MLE)

- Imagine I have a coin that I flip ten times, and got Heads 7 times, Tails 3 times
- I **am not sure** it is a fair coin, i.e. a 50% chance of Heads/Tails
- I want to estimate the probability of landing on each side given my sample
- Since the coin flips are independent of one another, the probability (or "likelihood") of observing the sample we have is

$$\begin{aligned}\mathcal{L}(p_H) &= p_H \times p_H \times p_H \times p_H \times p_H \times p_H \times p_H \\ &\quad \times (1-p_H) \times (1-p_H) \times (1-p_H) = p_H^7 (1-p_H)^3\end{aligned}$$

  where p_H is the probability of getting Heads
- We want to find $\hat{p}_H$ that maximizes the probability of observing that sample: **Maximum likelihood**
- Spoiler: $\hat{p}_H = 7/10$

Maximum likelihood

- Note that in a linear model, assuming the error terms follow a normal distribution lead to maximum likelihood estimates that are the same as the OLS estimates for the $\hat{\boldsymbol{\beta}}$'s and $\hat{\boldsymbol{\sigma}}^{\mathbf{2}}$!!
- Maximum likelihood estimators are very popular, in particular when the data strongly suggest some distributions
- They are **consistent** and **asymptotically normal**, so inference is relatively straightforward
- But they impose strong parametric assumptions, which is not always desirable
- Some alternatives exist: The pseudo-likelihood estimator uses kernels

Logistic regression in R

- The ***glm*** function from the ***MASS*** library estimates logit models among others
- Give it the formula as with ***lm***, the data set, and the "family" of models (here, it is "binomial")
- One can use ***summary*** as with ***lm***, and get the fitted values
- Here, I use the ***predict*** function to get the predictions on the second part of the data set

Logistic regression in R

```
# Logistic regression
logit_model <- glm(heart_disease ~ Age + sex + asymptomatic
                   + typical_angina + atypical_angina + blood_sugar
                   + exercise_induced_angina + MaximumHR + BP + Cholesterol,
                     data = train_heart, family = "binomial")
# Predict test data based on model
logit_predictions <- predict(logit_model, newdata = test_heart,
                             type = "response")
logit_predictions <- ifelse(logit_predictions > 0.5, 1, 0)
```

Logistic regression in R

```
##
## Call:
## glm(formula = heart_disease ~ Age + sex + asymptomatic + typical_angina +
##     atypical_angina + blood_sugar + exercise_induced_angina +
##     MaximumHR + BP + Cholesterol, family = "binomial", data = train_heart)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7443  -0.5983  -0.1928   0.5636   2.0231
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -0.558091   3.029932  -0.184 0.853862
## Age                       0.031545   0.029682   1.063 0.287877
## sex                      -2.064333   0.584363  -3.533 0.000411 ***
## asymptomatic              1.068452   0.566872   1.885 0.059454 .
## typical_angina            0.491122   0.831889   0.590 0.554943
## atypical_angina          -1.323583   0.794855  -1.665 0.095875 .
## blood_sugar              -0.458281   0.622850  -0.736 0.461864
## exercise_induced_angina   1.643220   0.608392   2.701 0.006915 **
## MaximumHR                -0.033523   0.012904  -2.598 0.009383 **
## BP                        0.017744   0.014518   1.222 0.221619
## Cholesterol               0.004947   0.005142   0.962 0.336024
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 207.84  on 150  degrees of freedom
## Residual deviance: 121.42  on 140  degrees of freedom
## AIC: 143.42
```

K nearest neighbours

K-nn for classification

- Nearest neighbours methods choose a number of close observations to the one we try to predict, and compute an average of the dependent variable $\boldsymbol{y_i}$ for these neighbours
- What about classification?
- The estimator is still the average of the $\boldsymbol{y_i}$, $\boldsymbol{i \in \mathcal{N}_0}$
- If $\boldsymbol{\sum_{i \in \mathcal{N}_0} y_i > 0.5}$, then we predict $\boldsymbol{\hat{y}_0} = 1$

K-nn for classification in R

```
# K-nn
knn_est <- gknn(heart_disease ~ Age + sex + asymptomatic
                    + typical_angina + atypical_angina + blood_sugar
                    + exercise_induced_angina + MaximumHR + BP
                    + Cholesterol,
                      data = train_heart,  k = 10 )

# Predict test data based on model
knn_predictions <- predict(knn_est, newdata = test_heart,
                           type = "prob")
knn_predictions <- ifelse(knn_predictions > 0.5, 1, 0)
```

Classification trees

Classification trees

- Similar to regression trees, but the dependent variable belongs to **classes** (Example: 1 for a "Yes", 0 for a "No" or 1 for option 1, 2 for option 2,...)
- Helps predict someone's decision based on her characteristics
- Example: predict purchasing behavior based on browsing history (Amazon...)
- Algorithm: similar to regression trees, but the function to minimize at each split is not the same
- Let $\hat{\boldsymbol{p}}_{\boldsymbol{m},\boldsymbol{k}}$ be the proportion of observations from class $\boldsymbol{k}$ in leaf $\boldsymbol{m}$
- The prediction for an individual in leaf $\boldsymbol{m}$ will be the class that is the most occurring
- Example: An observation ending up in a terminal node where "Yes" is the majority will get the prediction "Yes"

Classification trees

- Consider the following criterion called **entropy**:

$$\boldsymbol{E_m \equiv -\sum_{k=1}^{K} \hat{p}_{m,k} \log\left(\hat{p}_{m,k}\right)}$$

- Another criterion is the **Gini index**:

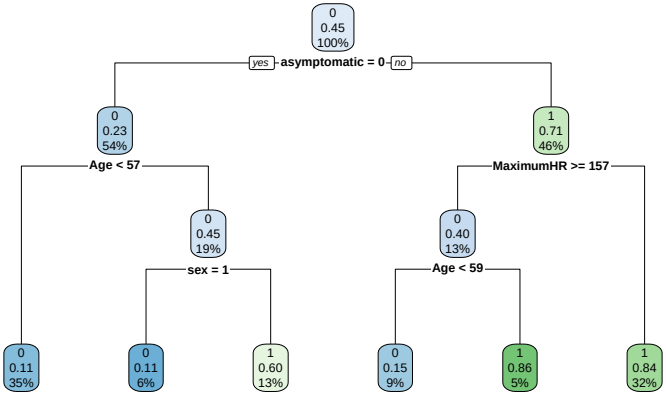
$$\boldsymbol{G_m \equiv \sum_{k=1}^{K} \hat{p}_{m,k}\left(1 - \hat{p}_{m,k}\right)}$$

- Both criteria are positive (or 0)
- When $\boldsymbol{\hat{p}_{m,k}}$ are all near 0 (or 1), both criteria will be close to 0
- The more "pure" the node is (i.e. the more it contains the same class), the lower the criteria. Classification trees minimize one of these criteria at each split

Classification trees in R

```
# Classification tree
tree_est <- rpart(heart_disease ~ Age + sex + asymptomatic
                  + typical_angina + atypical_angina + blood_sugar
                  + exercise_induced_angina + MaximumHR + BP + Cholesterol,
                  data = train_heart, cp = .02, method = "class")
# rpart.plot(tree_est)
tree_predictions <- predict(tree_est, newdata = test_heart, type = "class")
# tree_predictions <- ifelse(tree_predictions > 0.5, 1, 0)
```

Classification trees in R



Random forests for classification in R

- If we can grow trees for classification, we can grow random forests
- Each observation ends up in one leaf of each tree, where a prediction is made
- The forest then takes the decision that appears the most among all the trees as the prediction. Example: Out of 500 trees, if 300 trees predict a 1 and 200 predict a 0 for an observation, the final prediction for that observation is a 1
- Equivalently, the final prediction is 1 if $\hat{\boldsymbol{p}}(\boldsymbol{x_i}) > \mathbf{0.5}$, 0 if $\hat{\boldsymbol{p}}(\boldsymbol{x_i}) < \mathbf{0.5}$ (for ties, **R** typically randomly chooses a class)

Random forests for classification in R

```
# Random forest
rf_est <- randomForest(heart_disease ~ Age + sex + asymptomatic
                  + typical_angina + atypical_angina + blood_sugar
                  + exercise_induced_angina + MaximumHR + BP + Cholesterol,
                  data = train_heart)
rf_pred <- predict(rf_est, newdata = test_heart)
rf_predictions <- ifelse(rf_pred > 0.5, 1, 0)
```

Support vector machines (SVM)

Classification: Support vector machines (SVM)

- SVMs are used for classification problems
- Idea: cut the data with **hyperplanes**
- In a p-dimensional space, a **hyperplane** is an affine ($\simeq$ linear) subspace of dimension $p - 1$
- In a two-dimensional space (i.e. a surface), a hyperplane is a one-dimensional space (i.e. a line) that separates the surface into two halves
- In a three-dimensional space, a hyperplane is a two-dimensional space (i.e. a surface, like a flat wall) that separates the space into two halves
- Each plane contains one class, and any observation in a plane gets the corresponding class as a prediction

SVMs: Maximal margin classifier

- How to choose the best hyperplane?
- We want the hyperplane to be as far as possible from the observations
- Compute the (perpendicular) distance between the observations and the hyperplane
- The smallest of these distances is called the **margin**
- The **Maximal margin classifier** finds the hyperplane that maximizes the **margin**
- The vector representing the distance between the hyperplane and the closest point to it is called the **support vector**

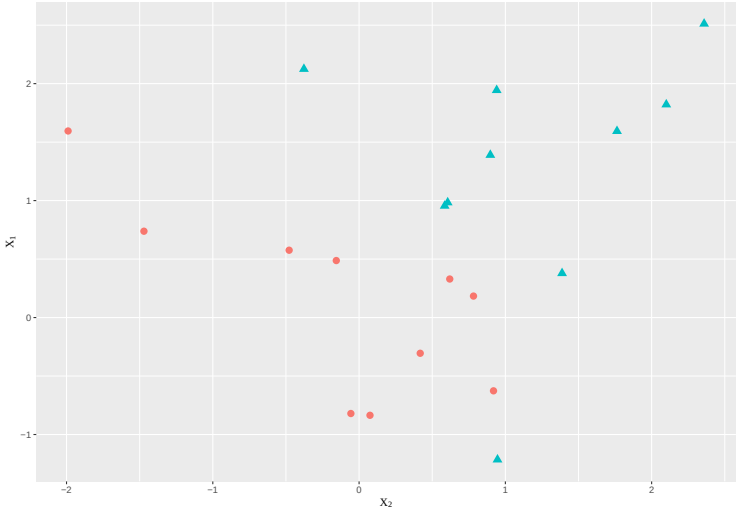
SVMs: Soft margins

- What if a **separating hyperplane** can't be found? As in, no hyperplane separates observations into two halves with one class in each?
- We can use a **soft margin**, i.e. we can allow some observations to be on the wrong side of the hyperplane or margin
- The minimization problem is modified by introducing **slack** variables, i.e. some variables that allow observations to be on the other side of the hyperplane containing their class

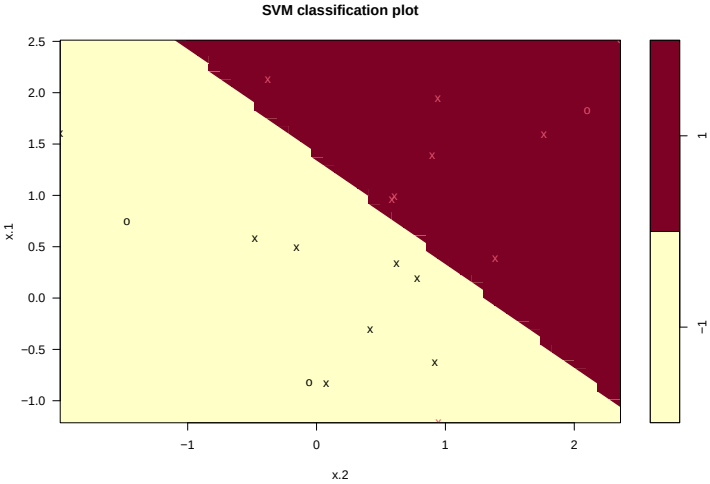
Nonlinear boundaries and the support vector machine

- What if the boundary between classes is better not be linear?
- One could include polynomials of the covariates: $\boldsymbol{X_{i,1}^2}$, $\boldsymbol{X_{i,2}^2}$ etc
- The extension of that method is what we refer to as the **support vector machine**
- The problem of maximization of the margin needs to include other measures of similarity between observations
- **kernels** are functions that quantify such similarity differently
- Using a specific kernel will produce different boundaries

SVMs: linear boundaries in R

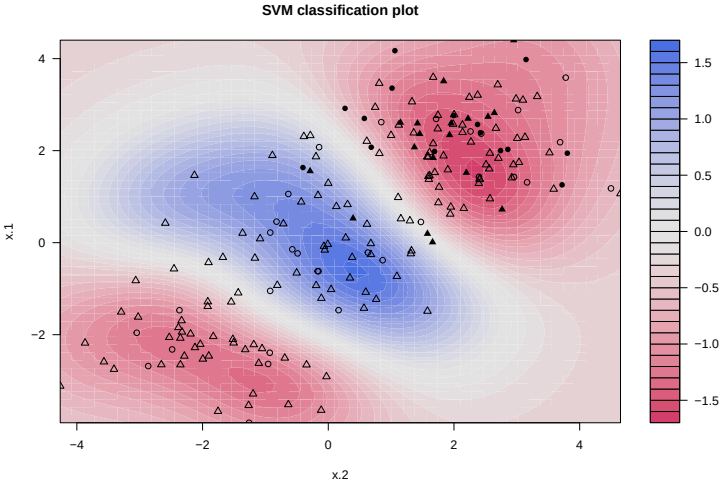


SVMs: linear boundaries in R



SVMs: Nonlinear boundaries in R

SVMs: Nonlinear boundaries in R



SVMs in R: Heart Disease data

```
# Support vector machine
svm_est <- svm(heart_disease ~ Age + sex + asymptomatic
               + typical_angina + atypical_angina
               + blood_sugar + exercise_induced_angina
               + MaximumHR + BP + Cholesterol, data = train_heart,
               kernel = 'linear', cost = 1, gamma = 1,
               decision.values = TRUE, scale = FALSE)

tune.out <- tune(svm, heart_disease ~ Age + sex + asymptomatic
                 + typical_angina + atypical_angina + blood_sugar
                 + exercise_induced_angina + MaximumHR + BP + Cholesterol,
                 data = train_heart, kernel = "radial", ranges = list (
cost = c (0.1 , 1 , 10 , 100 , 1000) ,
gamma = c (0.5 , 1 , 2 , 3 , 4)
))
# summary(tune.out)

svm_predictions <- predict(tune.out$best.model, newdata = test_heart)
svm_predictions <- ifelse(svm_predictions > 0.5, 1, 0)
```

SVMs: Multiple classes

- How do we apply SVMs when there are more than 2 classes?
- The extension is not straightforward, and the two most popular are:
  - ***One-versus-one***: Many SVMs are built, each comparing a pair of classes. We then assign a test observation to the class it was assigned the most often in each of the comparisons
  - ***One-versus-all***: We fit a SVM for each class vs all the other classes (e.g. class 1 vs all the other classes), and assign a test observation to the class where its distance from the boundary is the largest

Performance across methods

	Heart disease	Predictions	K-nn	SVM	LPM	Logit	Tree	Random forest
1	Yes	Yes	64	55	59	59	57	60
2	No	Yes	21	22	23	18	13	21
3	Yes	No	17	26	22	22	24	21
4	No	No	50	49	48	53	58	50

Assessing performance across classification methods

- Let us look at the performance measure, MCC (Matthews correlation coefficient). The higher, the better

```
MCC_table
```

```
##           Method       MCC
## 1           Tree 0.5211037
## 2          K-nn 0.4967734
## 3         Logit 0.4738862
## 4 Random forest 0.4449661
## 5           LPM 0.4048388
## 6           SVM 0.3683853
```

Conclusion

- There are many more classification methods:
  - Linear/quadratic discriminant analysis
  - Neural networks
  - Kernel based estimators: Klein Spady's, Ichimura's
  - Probit (normal distribution instead of logistic)
- Some estimators can deal with multiple classes: Multinomial logit/probit, unordered logit/probit
- And many more!