Machine Learning Worksheet

Azka Javaid, Caleb Ki & Muling Si

March 20, 2017

Data Preparation

Reading census data to predict income

```
census <- read.csv(
  "http://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data",
  header = FALSE
)
names(census) <- c("age", "workclass", "fnlwgt",
                   "education", "education.num",
                   "marital.status", "occupation",
                   "relationship", "race", "sex",
                   "capital.gain", "capital.loss",
                   "hours.per.week", "native.country",
                   "income")
```

```
glimpse(census)
```

```
## Observations: 32,561
## Variables: 15
## $ age            <int> 39, 50, 38, 53, 28, 37, 49, 52, 31, 42, 37, 30,...
## $ workclass      <fctr>  State-gov,  Self-emp-not-inc,  Private,  Priv...
## $ fnlwgt         <int> 77516, 83311, 215646, 234721, 338409, 284582, 1...
## $ education      <fctr>  Bachelors,  Bachelors,  HS-grad,  11th,  Bach...
## $ education.num  <int> 13, 13, 9, 7, 13, 14, 5, 9, 14, 13, 10, 13, 13,...
## $ marital.status <fctr>  Never-married,  Married-civ-spouse,  Divorced...
## $ occupation     <fctr>  Adm-clerical,  Exec-managerial,  Handlers-cle...
## $ relationship   <fctr>  Not-in-family,  Husband,  Not-in-family,  Hus...
## $ race           <fctr>  White,  White,  White,  Black,  Black,  White...
## $ sex            <fctr>  Male,  Male,  Male,  Male,  Female,  Female, ...
## $ capital.gain   <int> 2174, 0, 0, 0, 0, 0, 0, 0, 14084, 5178, 0, 0, 0...
## $ capital.loss   <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ hours.per.week <int> 40, 13, 40, 40, 40, 40, 16, 45, 50, 40, 80, 40,...
## $ native.country <fctr>  United-States,  United-States,  United-States...
## $ income         <fctr>  <=50K,  <=50K,  <=50K,  <=50K,  <=50K,  <=50K...
```

Partitioning data in train and test sets

```
set.seed(164)
n <- nrow(census)
test <- sample.int(n, size = round(0.2 * n))
train <- census[-test, ]
test <- census[test, ]
tally(~income, data = train, format = "percent")
```

```
## income
##  <=50K   >50K
##   75.8   24.2
```

Logistic regression to model income

```
logmod <- glm(income ~ capital.gain + age + workclass + education +
                marital.status + occupation +
                relationship + race + sex +
                capital.loss + hours.per.week,
                data = train, family=binomial(link='logit'))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Variable importance plot of logistic regression

```
head(varImp(logmod), 10)
```

```
##                              Overall
## capital.gain                 27.6232
## age                          13.9956
## workclass Federal-gov         5.8029
## workclass Local-gov           2.6110
## workclass Never-worked        0.0367
## workclass Private             4.0323
## workclass Self-emp-inc        4.3961
## workclass Self-emp-not-inc    0.4456
## workclass State-gov           1.0499
## workclass Without-pay         0.0437
```

Calculating accuracy

```
pred = predict(logmod, newdata=test)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```
accuracy <- table(pred, test[,"income"])
sum(diag(accuracy))/sum(accuracy)
```

```
## [1] 0.000154
```

K-nearest neighbor

```
trainX <- train %>%
  select(age, education.num,capital.gain, capital.loss, hours.per.week)
trainY <- train$income
incomeknn <- knn(trainX, test=trainX, cl=trainY, k = 10)
head(incomeknn)
```

```
## [1]  <=50K  <=50K  <=50K  >50K   >50K   <=50K
## Levels:  <=50K  >50K
```

Calculating confusion matrix and accuracy

```r
confusion <- tally(incomeknn~trainY, format="count")
confusion
```

```
##          trainY
## incomeknn  <=50K  >50K
##     <=50K  18845  3005
##     >50K     897  3302
```

```r
sum(diag(confusion))/nrow(train)
```

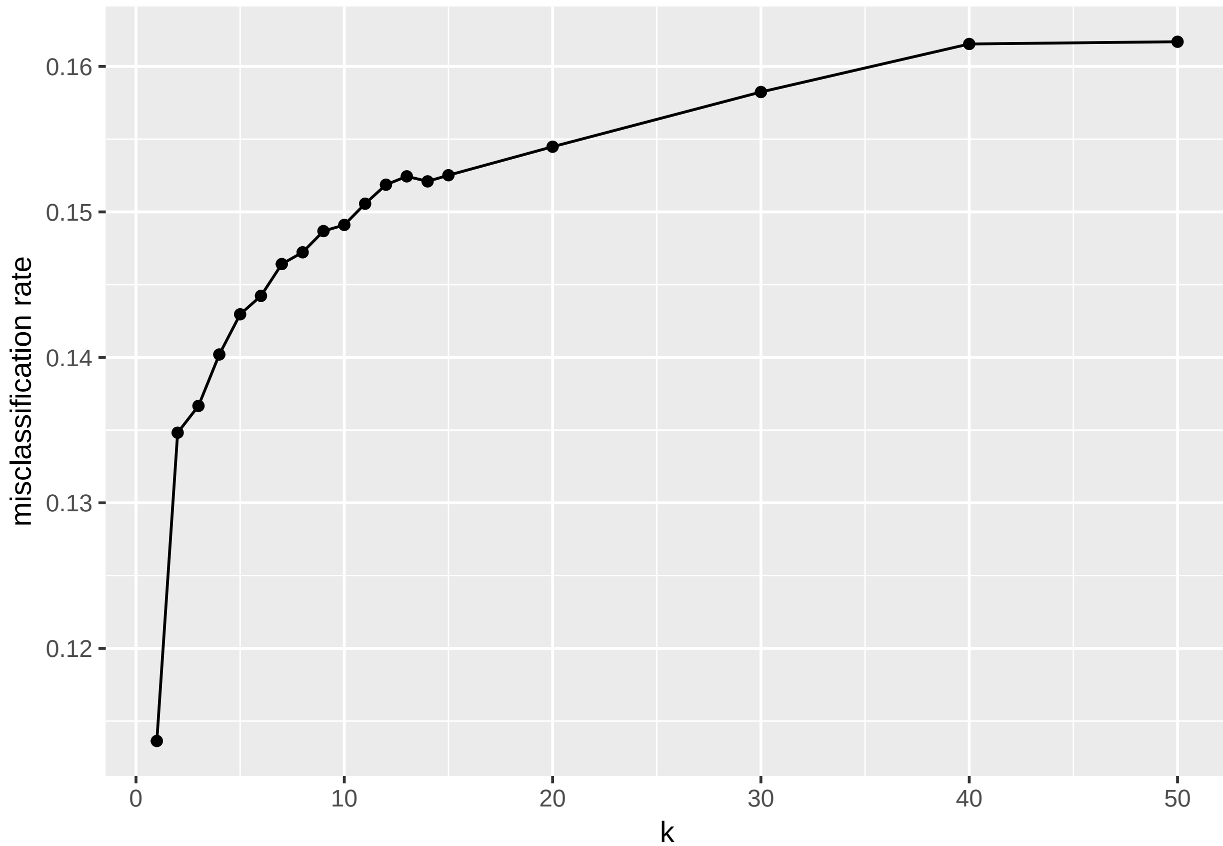
```
## [1] 0.85
```

Observing changes with different k

```r
knn_error_rate <- function(x,y, numNeighbors, z=x){
  y_hat <- knn(train =x, test=z, cl=y, k=numNeighbors)
  return(sum(y_hat!=y)/nrow(x))
}
ks <- c(1:15,20,30,40,50)
train_rate <- sapply(ks, FUN=knn_error_rate, x=trainX, y=trainY)
knn_error_rates <- data.frame(k=ks, train_rates=train_rate)
```

Plotting results

```r
ggplot(data=knn_error_rates, aes(x=k,y=train_rate)) +
  geom_point() + geom_line() + ylab("misclassification rate")
```



Decision tree

Decision tree using capital gain

```
mod_treeCap <- rpart(income ~ capital.gain, data = train)
mod_treeCap
```

```
## n= 26049
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 26049 6310  <=50K (0.7579 0.2421)
##   2) capital.gain< 5.12e+03 24774 5100  <=50K (0.7941 0.2059) *
##   3) capital.gain>=5.12e+03 1275   70  >50K (0.0549 0.9451) *
```

Decision tree using all predictors

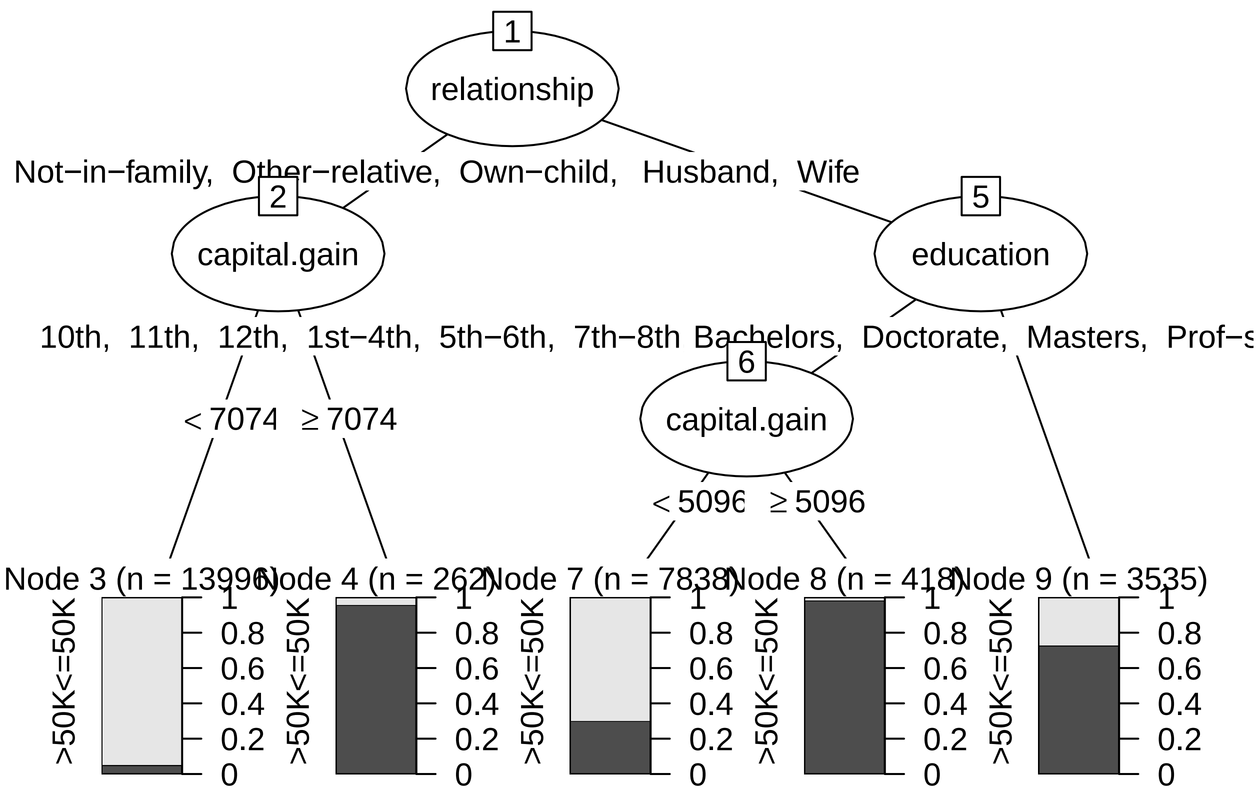
```
form <- as.formula("income ~ age + workclass +
                   education + marital.status +
                   occupation + relationship +
                   race + sex + capital.gain +
                   capital.loss + hours.per.week")
```

```
mod_tree <- rpart(form, data = train)
mod_tree
```

```
## n= 26049
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 26049 6310  <=50K (0.7579 0.2421)
##    2) relationship= Not-in-family, Other-relative, Own-child, Unmarried 14258  957  <=50K (0.9329 0.
##      4) capital.gain< 7.07e+03 13996  706  <=50K (0.9496 0.0504) *
##      5) capital.gain>=7.07e+03 262   11  >50K (0.0420 0.9580) *
##    3) relationship= Husband, Wife 11791 5350  <=50K (0.5463 0.4537)
##      6) education= 10th, 11th, 12th, 1st-4th, 5th-6th, 7th-8th, 9th, Assoc-acdm, Assoc-voc, HS-grad,
##       12) capital.gain< 5.1e+03 7838 2370  <=50K (0.6980 0.3020) *
##       13) capital.gain>=5.1e+03 418    7  >50K (0.0167 0.9833) *
##      7) education= Bachelors, Doctorate, Masters, Prof-school 3535  963  >50K (0.2724 0.7276) *
```

Plotting decision tree

```
plot(as.party(mod_tree))
```



Variable importance plot of decision tree

```
varImp(mod_tree)
```

```
##                 Overall
## age               338.2
## capital.gain     2719.3
## capital.loss      307.2
## education        2073.9
## hours.per.week     95.3
## marital.status   1901.0
## occupation       1879.6
## relationship     1929.3
## workclass           0.0
## race                0.0
## sex                 0.0
```

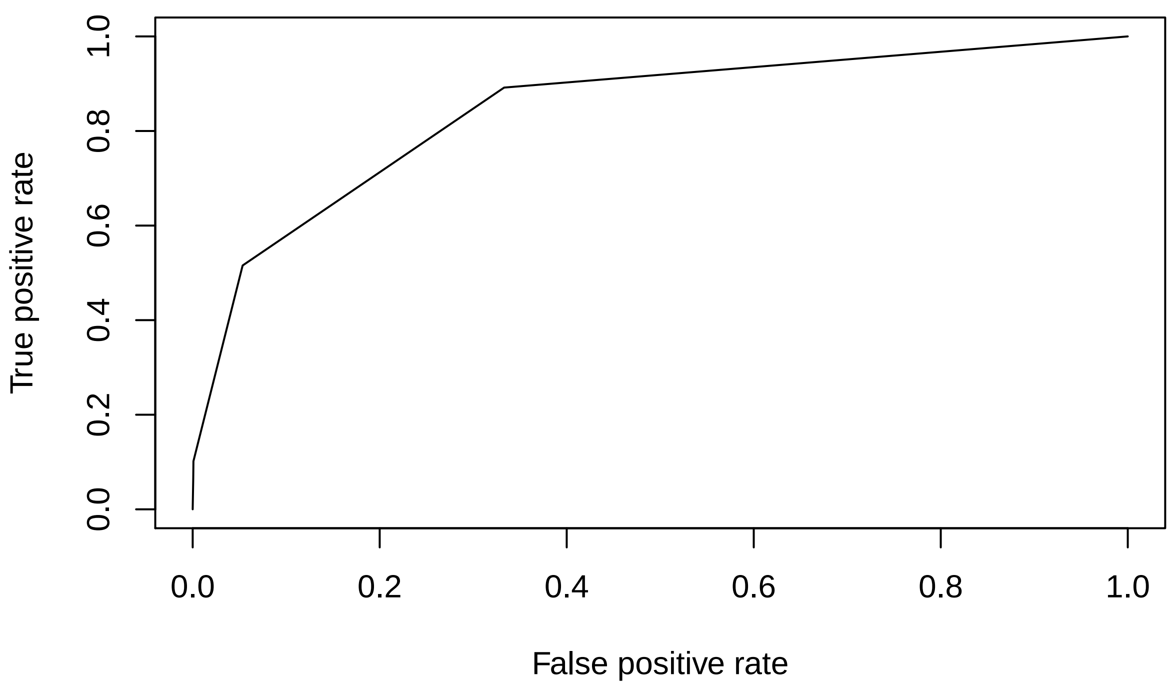
Calculating accuracy of decision tree

```r
pred <- predict(mod_tree, test, type = "class")
conf <- table(test$income, pred)
sum(diag(conf))/sum(conf) #accuracy
```

```
## [1] 0.845
```

Plotting ROC curve

```r
income_prob <- predict(mod_tree, newdata=test, type="prob")
perf <- prediction(income_prob[, 2], test$income)
perf <- performance(perf, measure = "tpr",
                    x.measure = "fpr")
plot(perf)
```



Random Forest

```
mod_forest <- randomForest(formula = form, data = train,
                           ntrain = 201, mtry = 3)
```

Calculating model accuracy

```
sum(diag(mod_forest$confusion))/nrow(train)
```

```
## [1] 0.865
```

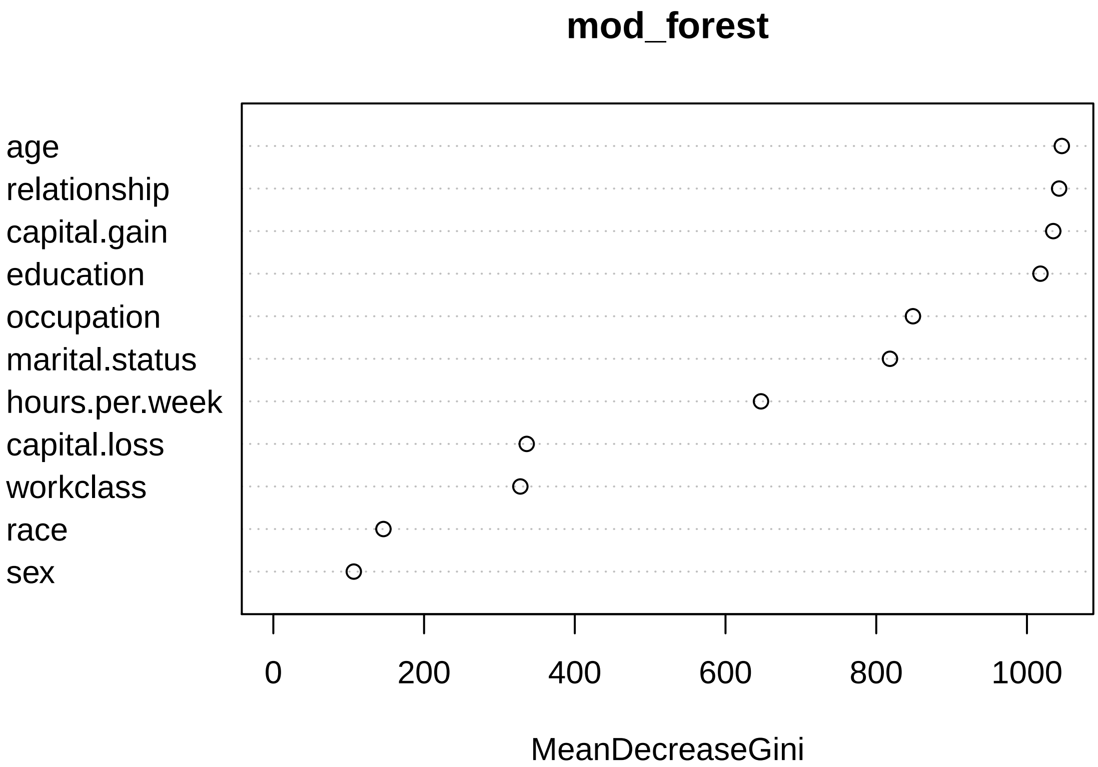
Calculating variable importance

```
importance(mod_forest) %>%
  as.data.frame() %>%
  tibble::rownames_to_column() %>%
  arrange(desc(MeanDecreaseGini))
```

```
##           rowname MeanDecreaseGini
## 1             age             1046
## 2    relationship             1043
## 3    capital.gain             1035
## 4       education             1018
## 5      occupation              849
## 6  marital.status              818
## 7  hours.per.week              647
## 8    capital.loss              336
## 9       workclass              328
## 10           race              146
## 11            sex              107
```

Variable importance plot

```
varImpPlot(mod_forest, type = 2)
```



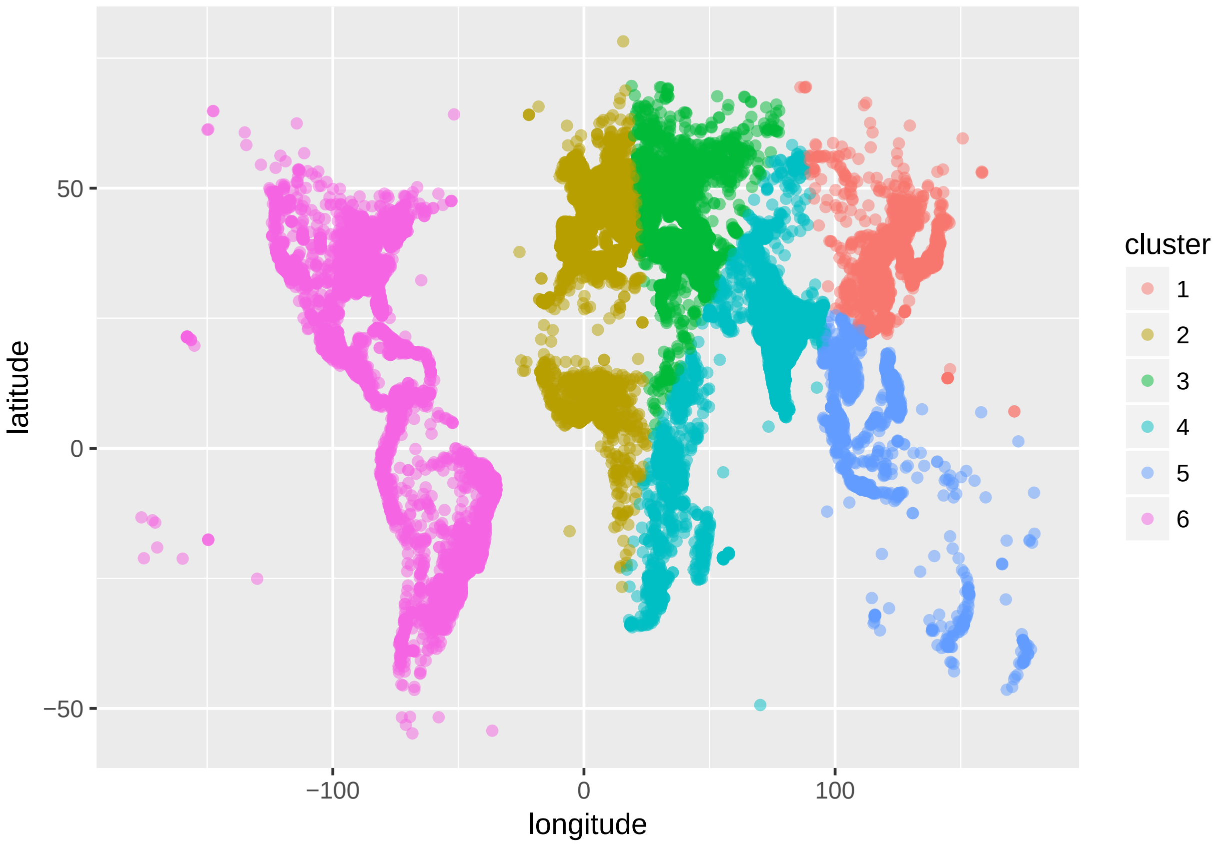
Clustering

```
WorldCities <- WorldCities %>%
  arrange(desc(population)) %>%
  select(longitude, latitude)

city_clusts <- WorldCities %>%
  kmeans(centers = 6) %>%
  fitted("classes") %>%
  as.character()

WorldCities <- WorldCities %>% mutate(cluster = city_clusts)

WorldCities %>% ggplot(aes(x = longitude, y = latitude)) +
  geom_point(aes(color = cluster), alpha = 0.5)
```


50
0
−50
−100
0
100
latitude
longitude
cluster
1
2
3
4
5
6