Plots Project Machine Learning

Hans

March 12, 2018

Setup

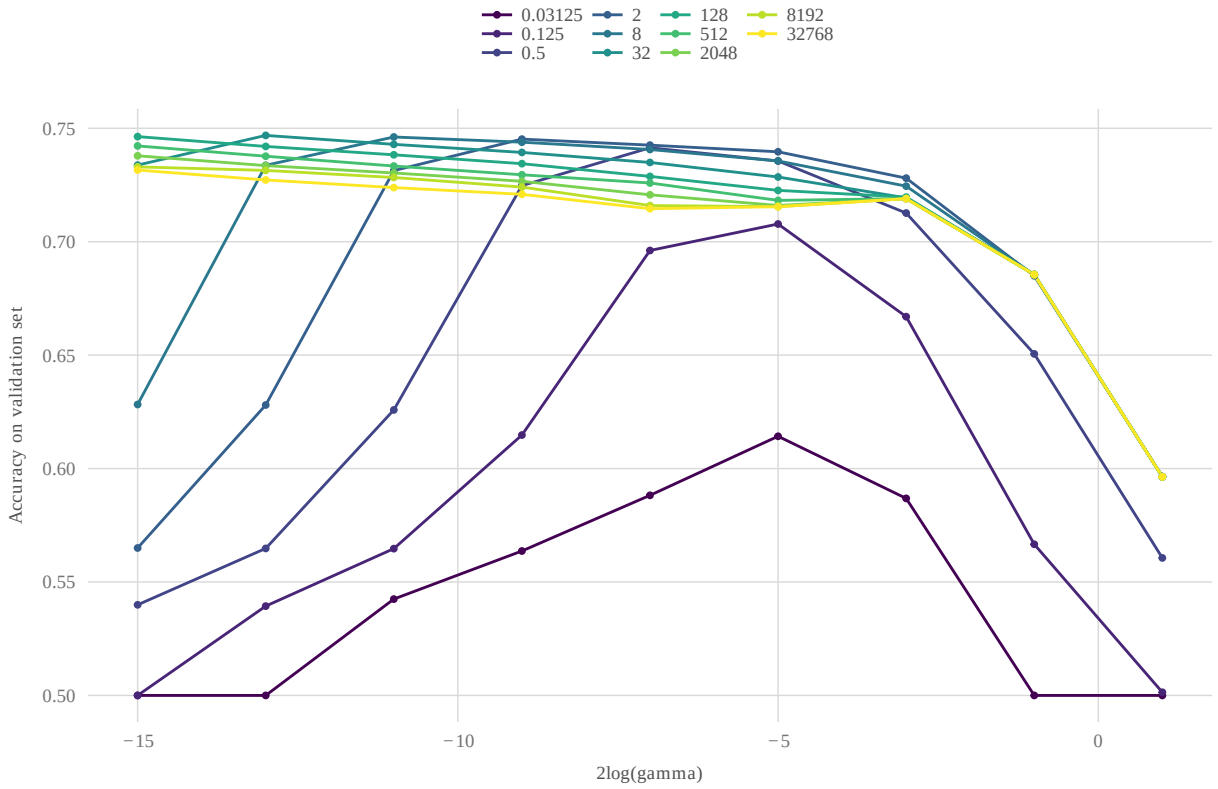
Plot for grid-search.

```
df <- read_csv('results_gridsearch_SVM_rbf') %>% mutate(C = as.factor(C))
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   X1 = col_integer(),
##   C = col_double(),
##   cache_size = col_double(),
##   gamma = col_double(),
##   kernel = col_character(),
##   score = col_double()
## )
```

```
plot <- df %>% ggplot(aes(x=log(gamma)/log(2), y=score, colour=C)) +
    geom_line(aes(y=)) +
    geom_point(size=.7) +
    fte_theme() +
    theme(legend.title = element_blank(),
        legend.position="top",
        legend.direction="horizontal",
        legend.key.width=unit(0.5, "cm"),
        legend.key.height=unit(0.25, "cm"),
        legend.spacing=unit(0,"cm"),
        plot.title = element_text(hjust = 0.5)) +
    labs(title="Results of grid-search for SVM with rbf kernel",
        x="2log(gamma)",
        y="Accuracy on validation set")+
    scale_color_viridis(discrete=TRUE) +
    scale_fill_viridis(discrete=TRUE)
plot
```




```
max(df$score)
```

```
## [1] 0.746861
```

```
save_image(plot, "results_grid_search_RBFkernel", pdf=TRUE)
```

Classification performance with majority votes (test)

```
df <- read_csv('output_SVM_RBF_classifier')

# Construct majority votes for DTM only predictions and DTM for characters and words combined with pred
df.joined <- read_csv('output_DTM_classifiers') %>% merge(df)
cols = c("native","prediction_struc","prediction_chars","prediction_words","prediction")
df.joined[cols] <- lapply(df.joined[cols], factor, levels=c("non-native","native"))
df.joined <- df.joined %>% mutate(maj_vote_dtm = ifelse(as.numeric(prediction_chars)+as.numeric(predicti
  mutate(maj_vote_dtm_sim = ifelse(as.numeric(prediction_chars)+as.numeric(prediction_words)+as.numeric

# Evaluation different classifiers.
accuracy.words = sum(df.joined$prediction_words == df.joined$native)/nrow(df.joined)
accuracy.chars = sum(df.joined$prediction_chars == df.joined$native)/nrow(df.joined)
accuracy.struc = sum(df.joined$prediction_struc == df.joined$native)/nrow(df.joined)
accuracy.sim = sum(df.joined$prediction==df.joined$native)/nrow(df.joined)
accuracy.joint_dtm = sum(df.joined$maj_vote_dtm == df.joined$native)/nrow(df.joined)
accuracy.joint_dtmsim = sum(df.joined$maj_vote_dtm_sim == df.joined$native)/nrow(df.joined)

cat("Contingency table for Char+Word+Sim (validation data)")
```

```
## Contingency table for Char+Word+Sim (validation data)
```

```
table(df.joined$maj_vote_dtm_sim,df.joined$native)
```

```
##
##              non-native native
##   native           3870  11972
##   non-native      11979   3877
```

```
cat("\n")
```

```
cat(paste0("Accuracy word  DTM:           ",100*accuracy.words,"\n"))
```

```
## Accuracy word  DTM:           71.9919237806802
```

```
cat(paste0("Accuracy char  DTM:           ",100*accuracy.chars,"\n"))
```

```
## Accuracy char  DTM:           71.5565650829705
```

```
cat(paste0("Accuracy struc DTM:           ",100*accuracy.struc,"\n"))
```

```
## Accuracy struc DTM:           60.4012871474541
```

```
cat(paste0("Accuracy sim RBF:             ",100*accuracy.sim,"\n"))
```

```
## Accuracy sim RBF:             74.6166950596252
```

```
cat(paste0("Accuracy sim+word+char DTM:   ",100*accuracy.joint_dtmsim,"\n"))
```

```
## Accuracy sim+word+char DTM:   75.5599722379961
```

```
cat(paste0("Accuracy word+struc+char DTM: ",100*accuracy.joint_dtm,"\n"))
```

```
## Accuracy word+struc+char DTM: 73.2349044103729
```

```
df <- read_csv('output_SVM_RBF_classifier_TEST')

# Construct majority votes for DTM only predictions and DTM for characters and words combined with pred
df.joined <- read_csv('output_DTM_classifiers_TEST') %>% merge(df)
cols = c("native","prediction_struc","prediction_chars","prediction_words","prediction")
df.joined[cols] <- lapply(df.joined[cols], factor, levels=c("non-native","native"))
df.joined <- df.joined %>% mutate(maj_vote_dtm = ifelse(as.numeric(prediction_chars)+as.numeric(predicti
  mutate(maj_vote_dtm_sim = ifelse(as.numeric(prediction_chars)+as.numeric(prediction_words)+as.numeric

# Evaluation different classifiers.
accuracy.words = sum(df.joined$prediction_words == df.joined$native)/nrow(df.joined)
accuracy.chars = sum(df.joined$prediction_chars == df.joined$native)/nrow(df.joined)
accuracy.struc = sum(df.joined$prediction_struc == df.joined$native)/nrow(df.joined)
accuracy.sim = sum(df.joined$prediction==df.joined$native)/nrow(df.joined)
accuracy.joint_dtm = sum(df.joined$maj_vote_dtm == df.joined$native)/nrow(df.joined)
accuracy.joint_dtmsim = sum(df.joined$maj_vote_dtm_sim == df.joined$native)/nrow(df.joined)

cat("Contingency table for Char+Word+Sim (test data)")
```

```
## Contingency table for Char+Word+Sim (test data)
```

```
table(df.joined$maj_vote_dtm_sim,df.joined$native)
```

```
##
```

```
##              non-native native
##   native           7798  23750
##   non-native      23633   7681
```

```
cat("\n")
```

```
cat(paste0("Accuracy word  DTM:           ",100*accuracy.words,"\n"))
```

```
## Accuracy word  DTM:           72.2423721803315
```

```
cat(paste0("Accuracy char  DTM:           ",100*accuracy.chars,"\n"))
```

```
## Accuracy char  DTM:           71.8208138462028
```

```
cat(paste0("Accuracy struc DTM:           ",100*accuracy.struc,"\n"))
```

```
## Accuracy struc DTM:           60.4514651140594
```

```
cat(paste0("Accuracy sim RBF:             ",100*accuracy.sim,"\n"))
```

```
## Accuracy sim RBF:             73.8808819318507
```

```
cat(paste0("Accuracy sim+word+char DTM:   ",100*accuracy.joint_dtmsim,"\n"))
```

```
## Accuracy sim+word+char DTM:   75.3762209283828
```

```
cat(paste0("Accuracy word+struc+char DTM: ",100*accuracy.joint_dtm,"\n"))
```

```
## Accuracy word+struc+char DTM: 73.1268492889186
```

Factors that may are associated with not being able to predict correctly.

```
df.joined <- df.joined[order(df.joined$num_words),] %>% mutate(FN = ifelse(maj_vote_dtm_sim!=native, if
  mutate(FP = ifelse(maj_vote_dtm_sim!=native, ifelse(maj_vote_dtm_sim=="native",TRUE,FALSE),FALSE))

df.joined <- df.joined %>%  mutate(numwordquantile = ifelse(num_words < quantile(num_words, probs=0.25)
```

See how factors are important, ceteris paribus. We can do this using glm.fit.

```
glm.fit_FP = glm(FP ~ level_english + native_lang + numwordquantile, data = df.joined, family = binomia
glm.fit_FN = glm(FN ~ numwordquantile, data = df.joined, family = binomial)

cat("Coefficient estimates logistic regression of False-Positives on self-reported English level, nativ
```

```
## Coefficient estimates logistic regression of False-Positives on self-reported English level, native
```

```
sort(glm.fit_FP$coefficients)
```

```
##    level_englishN       (Intercept) numwordquantileQ4     native_langPL
##      -18.80110782       -1.50074760       -0.66718415       -0.57786786
##     native_langTR     native_langHU     native_langRU numwordquantileQ3
##       -0.52398582       -0.36551647       -0.33558633       -0.32775659
```

```
## numwordquantileQ2     native_langNL     native_langPT     native_langIT
##       -0.14719971       -0.02470597        0.03495840        0.06422289
##     native_langDE     native_langDA     native_langES     native_langKO
##        0.06953213        0.13029772        0.13264221        0.14204860
##     native_langSV     native_langFR     native_langFI     native_langNO
##        0.15895522        0.22771521        0.25191245        0.31960669
##    level_english2    level_english3     native_langJA     native_langZH
##        0.36321194        0.37324528        0.48421924        0.50290032
##    level_english4  level_englishUNK    native_langYUE    level_english5
##        0.67029267        0.79552317        0.79670388        1.10851262
```

```
cat("\n")
```

```
cat("Coefficient estimates logistic regression of False-Negatives on quantile of number of words\n")
```

```
## Coefficient estimates logistic regression of False-Negatives on quantile of number of words
```

```
sort(glm.fit_FN$coefficients)
```

```
##       (Intercept) numwordquantileQ4 numwordquantileQ3 numwordquantileQ2
##        -1.7543438        -0.4315599        -0.3119154        -0.1607539
```

```
summary(glm.fit_FP)
```

```
##
## Call:
## glm(formula = FP ~ level_english + native_lang + numwordquantile,
##     family = binomial, data = df.joined)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.35329  -0.66822  -0.00006  -0.00005   2.36982
##
## Coefficients: (1 not defined because of singularities)
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -1.50075    0.23602  -6.358 2.04e-10 ***
## level_english2      0.36321    0.22276   1.631 0.102989
## level_english3      0.37325    0.21834   1.709 0.087365 .
## level_english4      0.67029    0.21856   3.067 0.002163 **
## level_english5      1.10851    0.22006   5.037 4.72e-07 ***
## level_englishN    -18.80111   99.56203  -0.189 0.850220
## level_englishUNK    0.79552    0.22413   3.549 0.000386 ***
## native_langDA       0.13030    0.10951   1.190 0.234108
## native_langDE       0.06953    0.09497   0.732 0.464058
## native_langEN            NA         NA      NA       NA
## native_langES       0.13264    0.09676   1.371 0.170413
## native_langFI       0.25191    0.11796   2.136 0.032710 *
## native_langFR       0.22772    0.09920   2.296 0.021701 *
## native_langHU      -0.36552    0.14185  -2.577 0.009972 **
## native_langIT       0.06422    0.12236   0.525 0.599666
## native_langJA       0.48422    0.16533   2.929 0.003403 **
## native_langKO       0.14205    0.27862   0.510 0.610176
## native_langNL      -0.02471    0.09780  -0.253 0.800557
## native_langNO       0.31961    0.11225   2.847 0.004408 **
## native_langPL      -0.57787    0.11249  -5.137 2.79e-07 ***
## native_langPT       0.03496    0.11497   0.304 0.761073
```

```
## native_langRU      -0.33559    0.10145  -3.308 0.000940 ***
## native_langSV       0.15896    0.10325   1.539 0.123683
## native_langTR      -0.52399    0.16741  -3.130 0.001748 **
## native_langYUE      0.79670    0.14264   5.586 2.33e-08 ***
## native_langZH       0.50290    0.11330   4.439 9.06e-06 ***
## numwordquantileQ2  -0.14720    0.03637  -4.047 5.19e-05 ***
## numwordquantileQ3  -0.32776    0.03665  -8.943  < 2e-16 ***
## numwordquantileQ4  -0.66718    0.03877 -17.210  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 47136  on 62861  degrees of freedom
## Residual deviance: 34155  on 62834  degrees of freedom
## AIC: 34211
##
## Number of Fisher Scoring iterations: 19
```