Supplementary Material 3: Random Forests

Measuring and assessing indeterminacy and variation in the morphology-syntax distinction

Sandra Auderset Adam J.R. Tallman

In this file we provide the code as well as some additional explanation and plots of the Random Forest models (cf. Section 6 of the paper).

Data Preparation

First, we load the necessary packages and set the seed to make the analysis fully reproducible (see the Rmd document for details).

```
# set seed
set.seed(200)
```

Next we load data (available at DOI), excluding columns we do not need for the analysis, i.e. we exclude the variables that are summarized in Exponence Complexity and certain meta information about the languages such as affiliation and glottocode.

```
db <- read_csv(here("SM1_Database.csv")) %>%
    select(Language, ClassAuthorSimple:EXPcomplex)
```

```
## Rows: 767 Columns: 16
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): Language, Glottocode, Family, Morphs, GlossAuthor, ClassAuthorSimp...
## dbl  (5): EXPcomplex, AMsegment, AMsupplN, EXPmultN, MSTfossilN
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(db)
```

```
## Rows: 767
## Columns: 8
## $ Language          <chr> "AsheninkaPerene", "AsheninkaPerene", "AsheninkaPere~
## $ ClassAuthorSimple <chr> "word", "word", "word", "word", "word", "word", "wor~
## $ PMfixed           <chr> "y", "y", "y", "y", "y", "y", "y", "y", "y", "y", "y~
## $ FREE              <chr> "n", "n", "n", "y", "n", "n", "y", "n", "n", "y", "n~
## $ INTERone          <chr> "y", "y", "y", "n", "n", "y", "y", "y", "y", "y", "y~
## $ PRM               <chr> "both", "both", "both", "both", "both", "both", "bot~
## $ CODelab           <chr> "y", "y", "y", "y", "y", "y", "y", "y", "n", "y", "y~
## $ EXPcomplex        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
```

Since there are too few data points for 'incorporand', we exclude this category from the analysis. We convert all variables to factors and subset the columns relevant for the analysis. Finally, we recode exponence complexity as a binary variable with values 'low' (1) and 'high' (over 1).

```
db_rf <- db %>%
    filter(ClassAuthorSimple != "incorporand") %>%
```

```
    droplevels() %>%
    mutate(EXPcomplex = if_else(EXPcomplex == 1, "low", "high")) %>%
    mutate_if(is.character, as.factor)
glimpse(db_rf)
```

```
## Rows: 754
## Columns: 8
## $ Language          <fct> AsheninkaPerene, AsheninkaPerene, AsheninkaPerene, A~
## $ ClassAuthorSimple <fct> word, word, word, word, word, word, word, word, word~
## $ PMfixed           <fct> y, y, y, y, y, y, y, y, y, y, y, y, y, y, y, y, y, y~
## $ FREE              <fct> n, n, n, y, n, n, y, n, n, y, n, n, y, y, n, y, n, y~
## $ INTERone          <fct> y, y, y, n, n, y, y, y, y, y, y, y, y, y, y, y, y, n~
## $ PRM               <fct> both, both, both, both, both, both, both, both, n, b~
## $ CODelab           <fct> y, y, y, y, y, y, y, y, n, y, y, y, n, y, n, y, y, y~
## $ EXPcomplex        <fct> low, low, low, low, low, low, low, low, low, low, lo~
```

We split the data set by language.

```
db_names <- db_rf %>%
    group_by(Language) %>%
    group_keys() %>%
    pull(1)
db_lang <- db_rf %>%
    group_split(Language) %>%
    setNames(db_names) %>%
    map(as_tibble) %>%
    map(., ~(.x %>%
        select(-Language))) %>%
    drop.levels()
glimpse(db_lang)
```

```
## List of 8
##  $ AsheninkaPerene    : tibble [98 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 3 3 3 3 3 3 3 3 3 3 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 2 1 1 2 1 1 2 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 1 1 2 2 2 2 2 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 1 1 1 1 1 1 1 1 2 1 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 1 2 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Cavineña           : tibble [56 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 1 1 1 2 1 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 1 1 1 1 2 2 2 2 2 2 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 2 2 2 2 1 1 1 1 1 1 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 1 2 2 2 1 2 2 2 2 2 ...
##  $ CentralAlaskanYupik: tibble [81 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 2 2 2 2 1 1 1 1 1 1 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 1 1 1 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 2 1 1 1 1 1 1 ...
##   ..$ PRM              : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 1 2 2 2 2 2 2 2 2 2 ...
##  $ Chacobo            : tibble [96 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 3 2 2 2 2 2 3 2 3 3 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 1 1 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 2 1 1 1 1 1 1 1 2 2 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 1 2 2 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 3 1 1 1 1 1 3 2 3 3 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 2 1 2 2 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 2 2 2 2 2 2 2 1 2 2 ...
##  $ Hup                : tibble [65 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 3 2 2 2 2 3 3 3 3 3 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 2 2 2 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 1 1 2 1 1 1 1 1 1 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 3 2 2 2 2 2 2 2 2 2 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 1 1 1 2 2 2 2 2 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Movima             : tibble [140 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 2 levels "affix","word": 2 2 1 1 1 1 1 1 1 1 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 1 1 1 1 1 1 1 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 3 3 2 1 1 1 1 1 1 1 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 2 2 1 2 2 2 2 2 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Puinave            : tibble [99 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 3 2 1 1 1 1 1 1 1 1 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 2 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 1 1 1 1 1 1 1 1 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 1 2 2 2 2 2 2 2 2 2 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 1 2 2 1 1 2 2 2 2 1 ...
##  $ Tariana            : tibble [119 x 7] (S3: tbl_df/tbl/data.frame)
##   ..$ ClassAuthorSimple: Factor w/ 3 levels "affix","clitic",..: 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ PMfixed          : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE             : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ INTERone         : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ PRM              : Factor w/ 3 levels "both","n","y": 2 2 1 1 1 1 1 1 1 1 ...
##   ..$ CODelab          : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ EXPcomplex       : Factor w/ 2 levels "high","low": 1 1 2 2 2 2 2 2 2 2 ...
```

We create a custom function for calculating baseline, accuracy and difference.

```
rf_acc <- function(x) {
    cm <- as.matrix(x$confusion[, -3])
    bl <- round(max(rowSums(cm))/sum(cm), 4)
    acc <- round(sum(diag(cm))/sum(cm), 4)
    diff <- round(acc - bl, 4)
    returnlist <- c(baseline = bl, accuracy = acc, difference = diff)
}
```

Author Classification

We apply a Random Forest model for each language separately with author classification as the dependent variable and all the wordhood criterial variables (exponence complexity, prominence projection, coding elaboration, fixedness, and interruptability) as predictors. More formally, the model is specified as:

> Author Classification ~ Exponence Complexity + Fixedness + Boundedness + Interruptability + Prominence + Coding Elaboration

This one and all the following RF models are implemented aggregating over 10'000 trees with two variables tried at each split. Each model outputs error rates for each level of the variable in a confusion matrix and an overall error rate for the model (called out-of-bag error or estimate, OOB for short). The former provides the classification accuracy for words vs. clitics vs. affixes and the latter how accurate the classification is overall. The model also provides a plot displaying Mean Decrease in Accuracy, which can be related to the relative importance of variables. Broadly speaking, certain variables are better and more consistent at classifying the dependent variable correctly than others across all the decision trees and this is reflected by MDA. Note that the actual value does not matter, since it is not comparable across models. What is important is the ranking of the independent variables and whether or not there are clear breaks between them.

To this we add a calculation of the baseline, the accuracy, and the difference between these two. The idea is to account for the skewness of the data, since the more skewed the data are, the easier it is to get a correct classification by chance. For example, imagine two languages X and Y with 100 data points each. Language X has 34 affixes, 33 clitics, and 33 words, while language Y has 15 affixes, 5 clitics, and 85 words. If we classify all elements as words in language Y, we get 85% right. But no matter what category we choose for language X, we do not achieve over 34% correct classification. The relative skewness by language needs to be taken into account when assessing whether the RF model performed well or not. The calculation of each of these measures is very simple. The baseline is exactly what we just discussed. More simply put, it represents the maximal row-sum of the confusion matrix divided by the number of data points. The accuracy represents the sum of correct predictions (i.e. the diagonal in the confusion matrix) divided by the number of data points, i.e. it reflects the proportion of correct predictions. The difference is calculated by subtracting the baseline measure from the accuracy, i.e. it reflects how much better the RF model performed over chance.

```
rf_aut <- lapply(1:length(db_lang), function(a) randomForest(ClassAuthorSimple ~
    ., data = db_lang[[a]], ntree = 10000, mtry = 2, importance = TRUE))
names(rf_aut) <- names(db_lang)
rf_aut
```

```
## $AsheninkaPerene
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 14.29%
## Confusion matrix:
## affix clitic word class.error
## affix 42 0 5 0.106383
## clitic 1 0 0 1.000000
## word 8 0 42 0.160000
##
## $Cavineña
##
## Call:
```

```
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 7.14%
## Confusion matrix:
## affix clitic word class.error
## affix 52 0 0 0
## clitic 3 0 0 1
## word 1 0 0 1
##
## $CentralAlaskanYupik
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 0%
## Confusion matrix:
## affix clitic word class.error
## affix 65 0 0 0
## clitic 0 8 0 0
## word 0 0 8 0
##
## $Chacobo
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 18.75%
## Confusion matrix:
## affix clitic word class.error
## affix 36 2 0 0.05263158
## clitic 8 22 6 0.38888889
## word 0 2 20 0.09090909
##
## $Hup
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
```

```
## OOB estimate of error rate: 10.77%
## Confusion matrix:
## affix clitic word class.error
## affix 4 1 0 0.20000000
## clitic 1 0 4 1.00000000
## word 0 1 54 0.01818182
##
## $Movima
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 0%
## Confusion matrix:
## affix word class.error
## affix 130 0 0
## word 0 10 0
##
## $Puinave
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 24.24%
## Confusion matrix:
## affix clitic word class.error
## affix 72 0 4 0.05263158
## clitic 12 0 0 1.00000000
## word 8 0 3 0.72727273
##
## $Tariana
##
## Call:
## randomForest(formula = ClassAuthorSimple ~ ., data = db_lang[[a]], ntree = 10000, mtry
= 2, importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 0.84%
## Confusion matrix:
## affix clitic word class.error
## affix 83 1 0 0.01190476
## clitic 0 18 0 0.00000000
## word 0 0 17 0.00000000
```

The OOB error rates range from 0 to over 23%, basically reflecting skewness as discussed above. CAY and Movima, which both have OOB errors of 0% also both have very skewed distributions while Chácobo, which has the most even distribution across the three categories also has a quite high error. It thus more instructive to look at the difference between baseline and accuracy. The highest performing models – in Chácobo and Ashéninka Perené – are performing over 30% better than chance. In Wãnsöjöt and Cavineña, however, the model is performing close to chance level and in Hup it is underperforming the baseline. Overall, we see that even though many models achieve an impressionistically high accuracy, the performance is not better than the baseline (i.e. chance).

```
rf_aut_measures <- lapply(1:length(rf_aut), function(m) rf_acc(rf_aut[[m]]))
names(rf_aut_measures) <- names(rf_aut)
rf_aut_measures
```

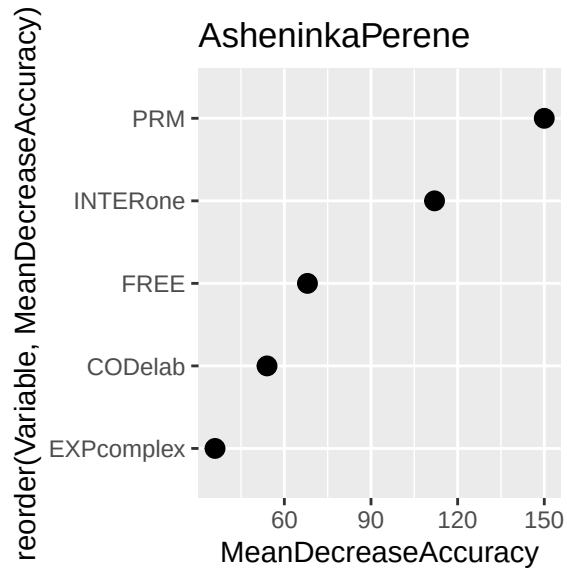
```
## $AsheninkaPerene
##   baseline   accuracy difference
##     0.8056     0.8066     0.0010
##
## $Cavineña
##   baseline   accuracy difference
##     0.8966     0.9138     0.0172
##
## $CentralAlaskanYupik
##   baseline   accuracy difference
##     0.8904     1.0000     0.1096
##
## $Chacobo
##   baseline   accuracy difference
##     0.5395     0.8236     0.2841
##
## $Hup
##   baseline   accuracy difference
##     0.6327     0.4889    -0.1438
##
## $Movima
##   baseline   accuracy difference
##     0.9286     1.0000     0.0714
##
## $Puinave
##   baseline   accuracy difference
##     0.7683     0.7755     0.0072
##
## $Tariana
##   baseline   accuracy difference
##     0.8236     0.9901     0.1665
```

```
# create data frames for variable importance plots
rf_aut_imp <- lapply(1:length(rf_aut), function(i) as_tibble(varImpPlot(rf_aut[[i]]),
    rownames = "Variable") %>%
    slice(-1) %>%
    mutate(MeanDecreaseAccuracy = round(MeanDecreaseAccuracy)))
```

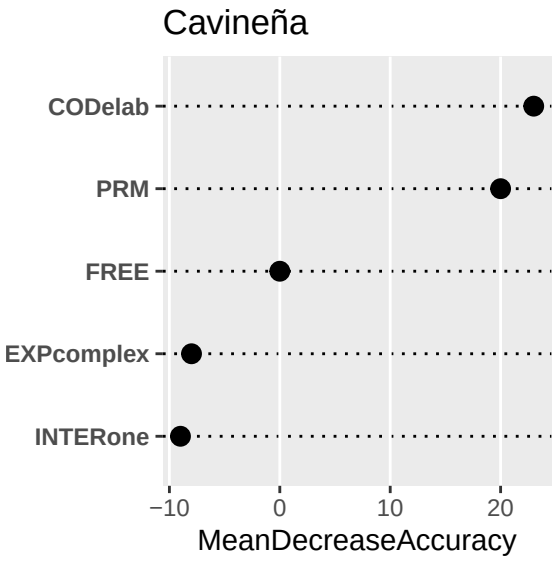
```
# display plots
rf_aut_plots <- lapply(1:length(rf_aut_imp), function(p) ggplot(rf_aut_imp[[p]],
    aes(x = MeanDecreaseAccuracy, y = reorder(Variable, MeanDecreaseAccuracy))) +
    geom_point(size = 3) + theme_bw() + theme_update(panel.grid.minor.x = element_blank(),
```

```
    panel.grid.major.y = element_line(color = "black", linetype = "dotted"),
    axis.title.y = element_blank(), axis.text.y = element_text(face = "bold")) +
    ggtitle(names(rf_aut)[[p]]))
names(rf_aut_plots) <- names(rf_aut)
rf_aut_plots
```

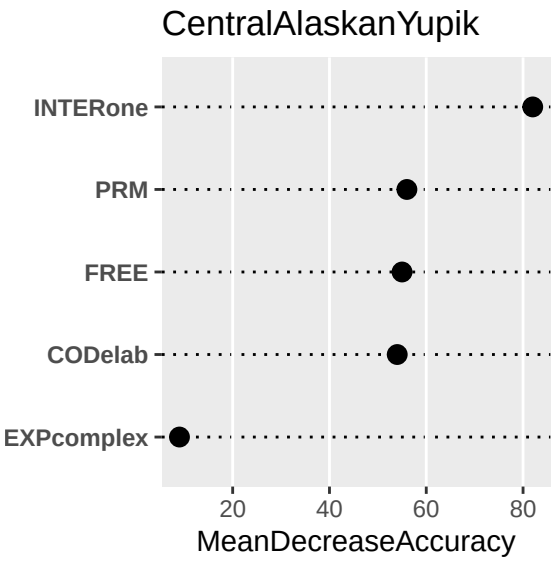
```
## $AsheninkaPerene
```



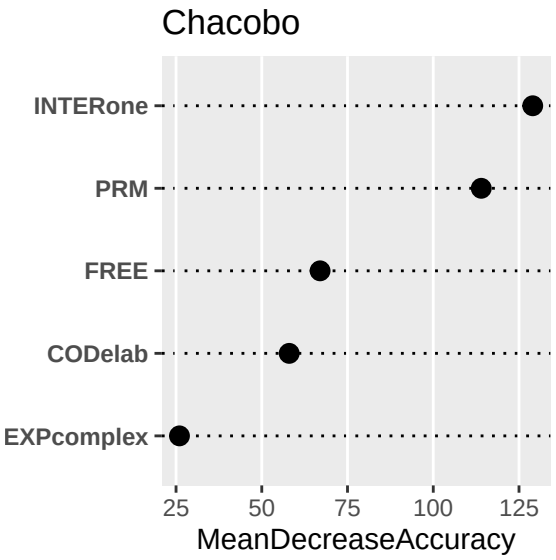

```
##
## $Cavineña
```



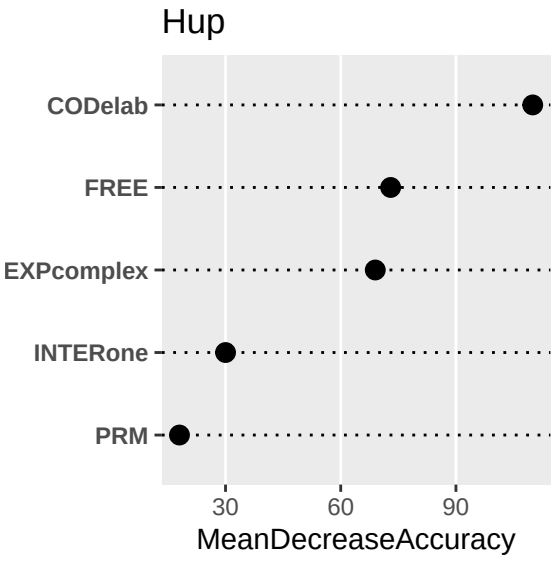

```
##
## $CentralAlaskanYupik
```


CentralAlaskanYupik
INTERone
PRM
FREE
CODelab
EXPcomplex
20
40
60
80
MeanDecreaseAccuracy

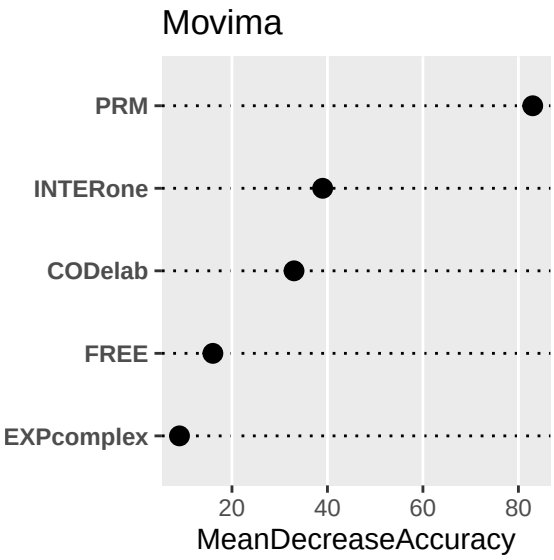
```
##
## $Chacobo
```


Chacobo
INTERone
PRM
FREE
CODelab
EXPcomplex
25
50
75
100
125
MeanDecreaseAccuracy

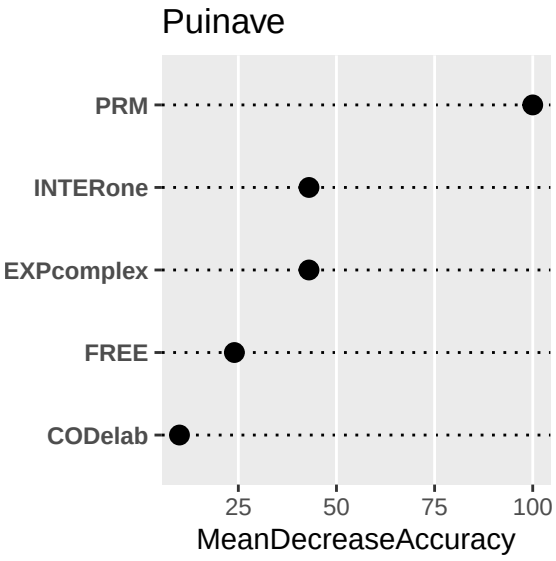
```
##
## $Hup
```



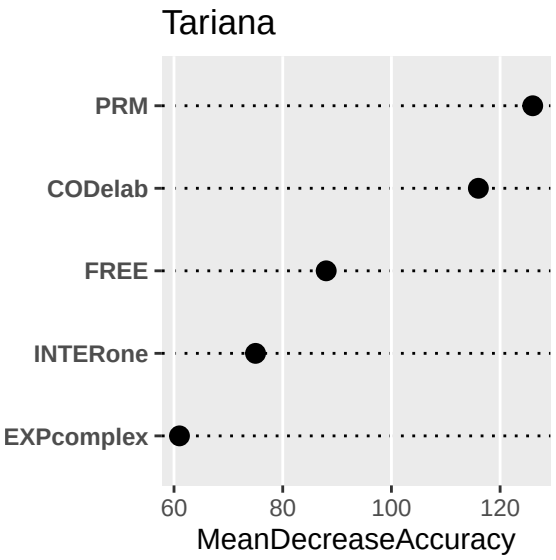

```
##
## $Movima
```




```
##
## $Puinave
```




```
##
## $Tariana
```



The MDA plots show that authors generally rely on one to three variables to classify elements into affixes, clitics, and words. This can be seen by the break that is present in each plot between the upper one to three variables to the lower ones. The plots also show that different authors rely on different variables, although prominence projection appears to be the most important. Both Wãnsöjöt and Hup show the first break already after one variable. This means the authors rely predominantly on prosodic prominence and coding elaboration, respectively. Cavineña shows a break after three variables, fixedness, coding elaboration, and prominence. However, it appears that relying on either just one or all of three variables results in the model performing at about chance level. For the languages studied here, it seems that relying on two variables results in the most consistent classifications by authors. In all of the remaining languages, prominence projection is one of these variables, but the others vary. To summarize, the results show that authors rely on different variables and different numbers of variables, but prominence often plays a role when classifying elements into words, affixes, and clitics.

Exponence Complexity

We now apply a Random Forest model for each language separately with exponence complexity as the dependent variable and the other wordhood criterial variables (prominence projection, coding elaboration, fixedness, and interruptability) as predictors. More formally, the model is specified as:

> Exponence Complexity ~ Fixedness + Boundedness + Interruptability + Prominence + Coding Elaboration

We split the data by language, this time excluding author classification.

```
db_names <- db_rf %>%
    group_by(Language) %>%
    group_keys() %>%
    pull(1)
db_lang2 <- db_rf %>%
    group_split(Language) %>%
    setNames(db_names) %>%
    map(as_tibble) %>%
    map(., ~(.x %>%
        select(-Language, -ClassAuthorSimple))) %>%
    drop.levels()
glimpse(db_lang2)
```

```
## List of 8
##  $ AsheninkaPerene    : tibble [98 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 2 1 1 2 1 1 2 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 1 1 2 2 2 2 2 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 1 1 1 1 1 1 1 1 2 1 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 1 2 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Cavineña           : tibble [56 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 1 1 1 2 1 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 1 1 1 1 2 2 2 2 2 2 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 2 2 2 2 1 1 1 1 1 1 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 1 2 2 2 1 2 2 2 2 2 ...
##  $ CentralAlaskanYupik: tibble [81 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 1 1 1 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 2 1 1 1 1 1 1 ...
##   ..$ PRM       : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 1 2 2 2 2 2 2 2 2 2 ...
##  $ Chacobo            : tibble [96 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 1 1 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 2 1 1 1 1 1 1 1 2 2 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 1 2 2 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 3 1 1 1 1 1 3 2 3 3 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 2 1 2 2 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 2 2 2 2 2 2 2 2 1 2 2 ...
##  $ Hup                : tibble [65 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 2 2 2 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 1 1 2 1 1 1 1 1 1 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 3 2 2 2 2 2 2 2 2 2 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 1 1 1 2 2 2 2 2 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Movima             : tibble [140 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 1 1 1 1 1 1 1 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 3 3 2 1 1 1 1 1 1 1 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 2 2 1 2 2 2 2 2 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 2 1 2 2 2 2 2 2 2 2 ...
##  $ Puinave            : tibble [99 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 2 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 1 1 1 1 1 1 1 1 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 1 2 2 2 2 2 2 2 2 2 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 1 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 1 2 2 1 1 2 2 2 2 1 ...
##  $ Tariana            : tibble [119 x 6] (S3: tbl_df/tbl/data.frame)
##   ..$ PMfixed   : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ FREE      : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ INTERone  : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...
##   ..$ PRM       : Factor w/ 3 levels "both","n","y": 2 2 1 1 1 1 1 1 1 1 ...
##   ..$ CODelab   : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 2 2 2 2 ...
##   ..$ EXPcomplex: Factor w/ 2 levels "high","low": 1 1 2 2 2 2 2 2 2 2 ...
```

We apply the random forest models to this new data frame as specified above.

```
rf_ex <- lapply(1:length(db_lang2), function(a) randomForest(EXPcomplex ~
    ., data = db_lang2[[a]], ntree = 10000, mtry = 2, importance = TRUE))
names(rf_ex) <- names(db_lang2)
rf_ex
```

```
## $AsheninkaPerene
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 11.22%
## Confusion matrix:
## high low class.error
## high 0 11 1
## low 0 87 0
##
## $Cavineña
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
```

```
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 19.64%
## Confusion matrix:
## high low class.error
## high 0 11 1
## low 0 45 0
##
## $CentralAlaskanYupik
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 16.05%
## Confusion matrix:
## high low class.error
## high 0 13 1
## low 0 68 0
##
## $Chacobo
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 35.42%
## Confusion matrix:
## high low class.error
## high 0 31 1.00000000
## low 3 62 0.04615385
##
## $Hup
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 13.85%
## Confusion matrix:
## high low class.error
## high 0 6 1.00000000
## low 3 56 0.05084746
##
## $Movima
```

```
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 6.43%
## Confusion matrix:
## high low class.error
## high 52 8 0.1333333
## low 1 79 0.0125000
##
## $Puinave
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 29.29%
## Confusion matrix:
## high low class.error
## high 0 27 1.00000000
## low 2 70 0.02777778
##
## $Tariana
##
## Call:
## randomForest(formula = EXPcomplex ~ ., data = db_lang2[[a]], ntree = 10000, mtry = 2,
importance = TRUE)
## Type of random forest: classification
## Number of trees: 10000
## No. of variables tried at each split: 2
##
## OOB estimate of error rate: 10.08%
## Confusion matrix:
## high low class.error
## high 0 11 1.000000000
## low 1 107 0.009259259
```

We calculate baseline, accuraccy and difference.

```
rf_ex_measures <- lapply(1:length(rf_ex), function(m) rf_acc(rf_ex[[m]]))
names(rf_ex_measures) <- names(rf_ex)
rf_ex_measures
```

```
## $AsheninkaPerene
##   baseline   accuracy difference
##     0.8878     0.8878     0.0000
##
## $Cavineña
```

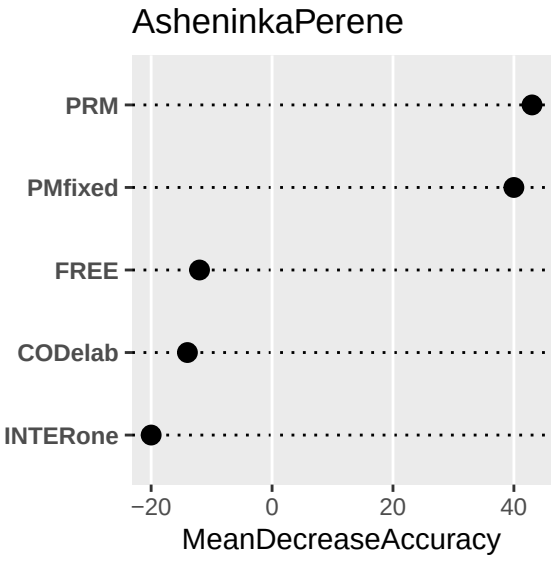
```
##   baseline   accuracy difference
##     0.8036     0.8036     0.0000
##
## $CentralAlaskanYupik
##   baseline   accuracy difference
##     0.8395     0.8395     0.0000
##
## $Chacobo
##   baseline   accuracy difference
##     0.6771     0.6458    -0.0313
##
## $Hup
##   baseline   accuracy difference
##     0.9077     0.8615    -0.0462
##
## $Movima
##   baseline   accuracy difference
##     0.5714     0.9357     0.3643
##
## $Puinave
##   baseline   accuracy difference
##     0.7273     0.7071    -0.0202
##
## $Tariana
##   baseline   accuracy difference
##     0.9076     0.8992    -0.0084
```

```
# create data frames for variable importance plots
rf_ex_imp <- lapply(1:length(rf_ex), function(i) as_tibble(varImpPlot(rf_ex[[i]]),
    rownames = "Variable") %>%
    mutate(MeanDecreaseAccuracy = round(MeanDecreaseAccuracy)))
```

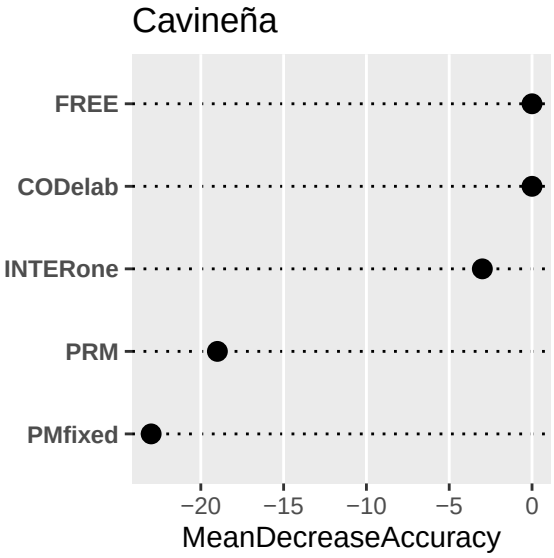
And we display the variable importance plots.

```
# display plots
rf_ex_plots <- lapply(1:length(rf_ex_imp), function(p) ggplot(rf_ex_imp[[p]],
    aes(x = MeanDecreaseAccuracy, y = reorder(Variable, MeanDecreaseAccuracy))) +
    geom_point(size = 3) + theme_bw() + theme_update(panel.grid.minor.x = element_blank(),
    panel.grid.major.y = element_line(color = "black", linetype = "dotted"),
    axis.title.y = element_blank(), axis.text.y = element_text(face = "bold")) +
    ggtitle(names(rf_ex)[[p]]))
names(rf_ex_plots) <- names(rf_ex)
rf_ex_plots
```

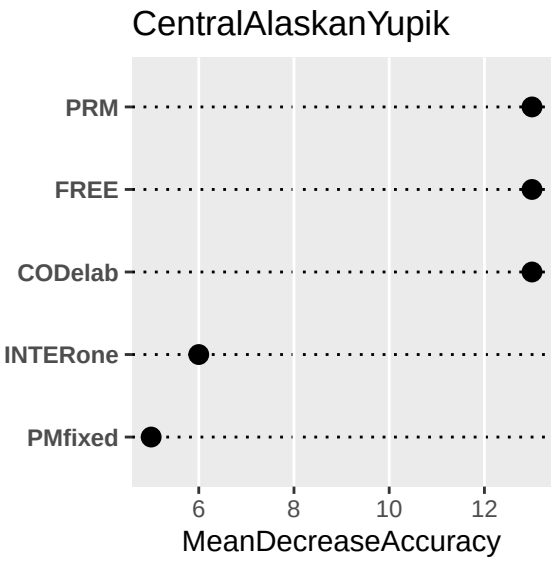
```
## $AsheninkaPerene
```



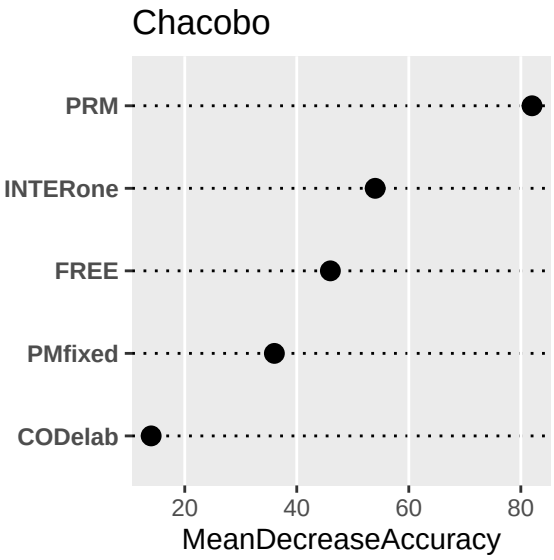

```
##
## $Cavineña
```



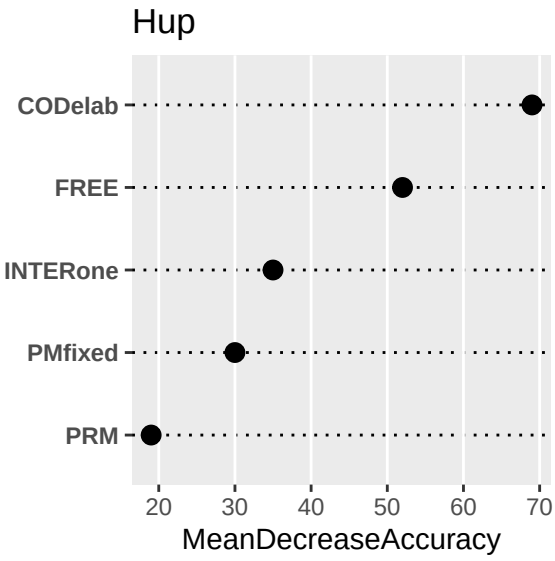

```
##
## $CentralAlaskanYupik
```


CentralAlaskanYupik
PRM
FREE
CODelab
INTERone
PMfixed
6
8
10
12
MeanDecreaseAccuracy

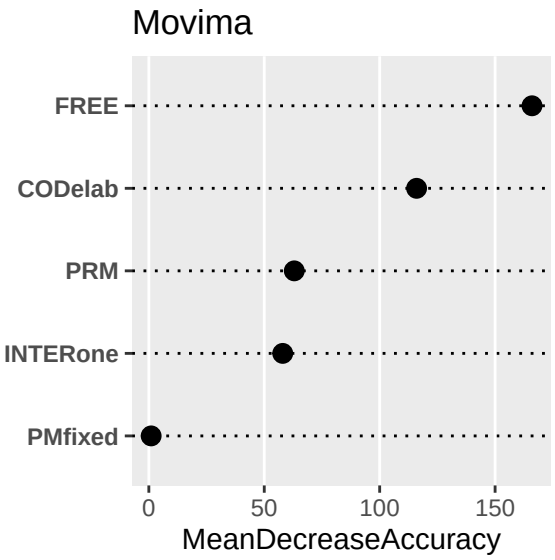
```
##
## $Chacobo
```


Chacobo
PRM
INTERone
FREE
PMfixed
CODelab
20
40
60
80
MeanDecreaseAccuracy

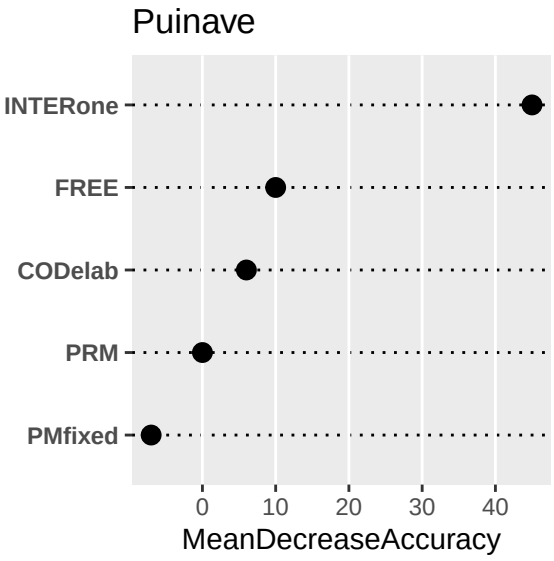
```
##
## $Hup
```



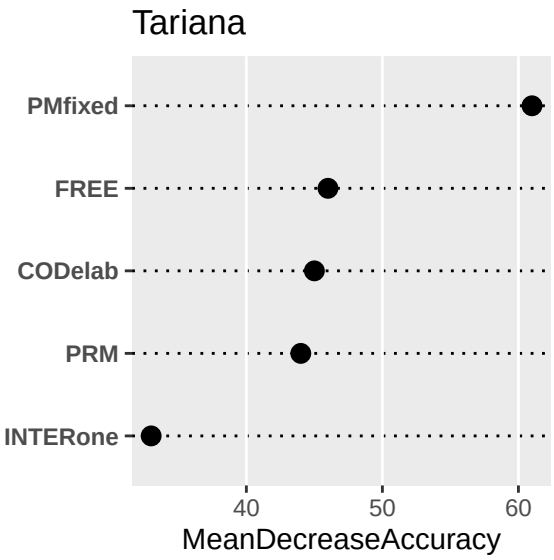

```
##
## $Movima
```




```
##
## $Puinave
```


Puinave
INTERone
FREE
CODelab
PRM
PMfixed
0
10
20
30
40
MeanDecreaseAccuracy

```
##
## $Tariana
```


Tariana
PMfixed
FREE
CODelab
PRM
INTERone
40
50
60
MeanDecreaseAccuracy