

## Outputs inspection half CIFAR10

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
##
```

```
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
## filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
## intersect, setdiff, setequal, union
```

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
library("ggpubr")
```

```
## Warning: package 'ggpubr' was built under R version 4.0.5
```

```
library(LDATS)
```

```
## Warning: package 'LDATS' was built under R version 4.0.5
```

```
library(stringr)
```

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.0.3
```

```
##
```

```
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
```

```
##
```

```
## smiths
```

```
library(reticulate)
```

```
## Warning: package 'reticulate' was built under R version 4.0.5
```

```
np <- import("numpy")
```

```
source("utils.R")
```

```
## Warning: package 'hash' was built under R version 4.0.5
```

```
## hash-2.2.6.1 provided by Decision Patterns
```

```
## Warning: package 'berryFunctions' was built under R version 4.0.5
```

```
##
```

```
## Attaching package: 'berryFunctions'
```

```
## The following object is masked from 'package:dplyr':
```

```
##
```

```
##      between
```

```
## Warning: package 'purrr' was built under R version 4.0.3
```

Visualization on CIFAR10. We are using data of three neural networks trained on reduced CIFAR10 training set. Half of the CIFAR10 training set was extracted as a validation set. We then divided both the reduced training set and validation set into 50 disjoint subsets and trained an ensemble on each of them. In this visualization, we are trying to inspect the outputs deeper, mainly to make sense of strange behavior of nll metric for ensemble outputs.

```
base_dir <- "../data/data_train_val_half_c10"
```

```
repls <- 0:0
```

```
folds <- 0:49
```

```
classes <- 10
```

```
nets_outputs <- load_network_outputs(base_dir, repls)
```

```
ens_outputs <- load_ensemble_outputs(base_dir, repls, folds)
```

```
net_results <- read.csv(file.path(base_dir, "net accuracies.csv"))
```

```
ens_results <- read.csv(file.path(base_dir, "ensemble accuracies.csv"))
```

```
preds <- nets_outputs$test_outputs
```

```
for (ri in repls + 1)
```

```
{
```

```
  for (net_i in seq_along(nets_outputs[["networks"]]))
```

```
  {
```

```
    preds[ri, net_i, ] <- softmax(preds[ri, net_i, , ])
```

```
  }
```

```
}
```

```
nets_test_cor_probs <- gather(preds, 1 + nets_outputs$test_labels[1, ], 3, 4)
```

```
nets_test_cor_probs <- melt(nets_test_cor_probs)
```

```
nets_test_cor_probs <- nets_test_cor_probs[, c(-3, -4)]
```

```
names(nets_test_cor_probs) <- c("replication", "network", "prediction")
```

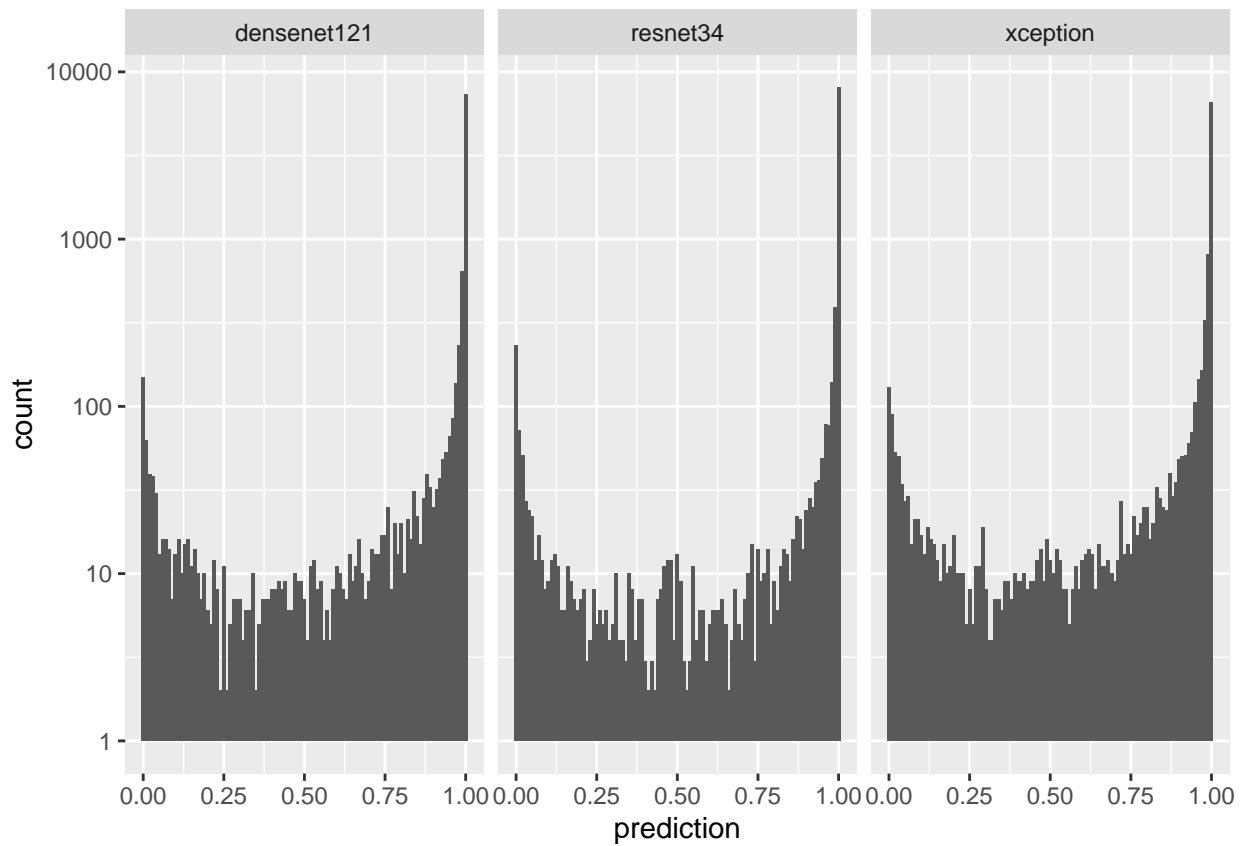
```
nets_test_cor_probs$network <- as.factor(nets_test_cor_probs$network)
```

```
levels(nets_test_cor_probs$network) <- nets_outputs$networks
```

```

nets_cor_preds_histo <- ggplot(data=nets_test_cor_probs) + geom_histogram(mapping=aes(x=prediction), bin
nets_cor_preds_histo

```



```

val_ens_cor_probs <- gather(ens_outputs$val_training, 1 + nets_outputs$test_labels[1, ], 4, 5)
val_ens_cor_probs <- melt(val_ens_cor_probs)
val_ens_cor_probs <- val_ens_cor_probs[, c(-4, -5)]
names(val_ens_cor_probs) <- c("replication", "method", "fold", "prediction")
val_ens_cor_probs$method <- as.factor(val_ens_cor_probs$method)
levels(val_ens_cor_probs$method) <- ens_outputs$methods

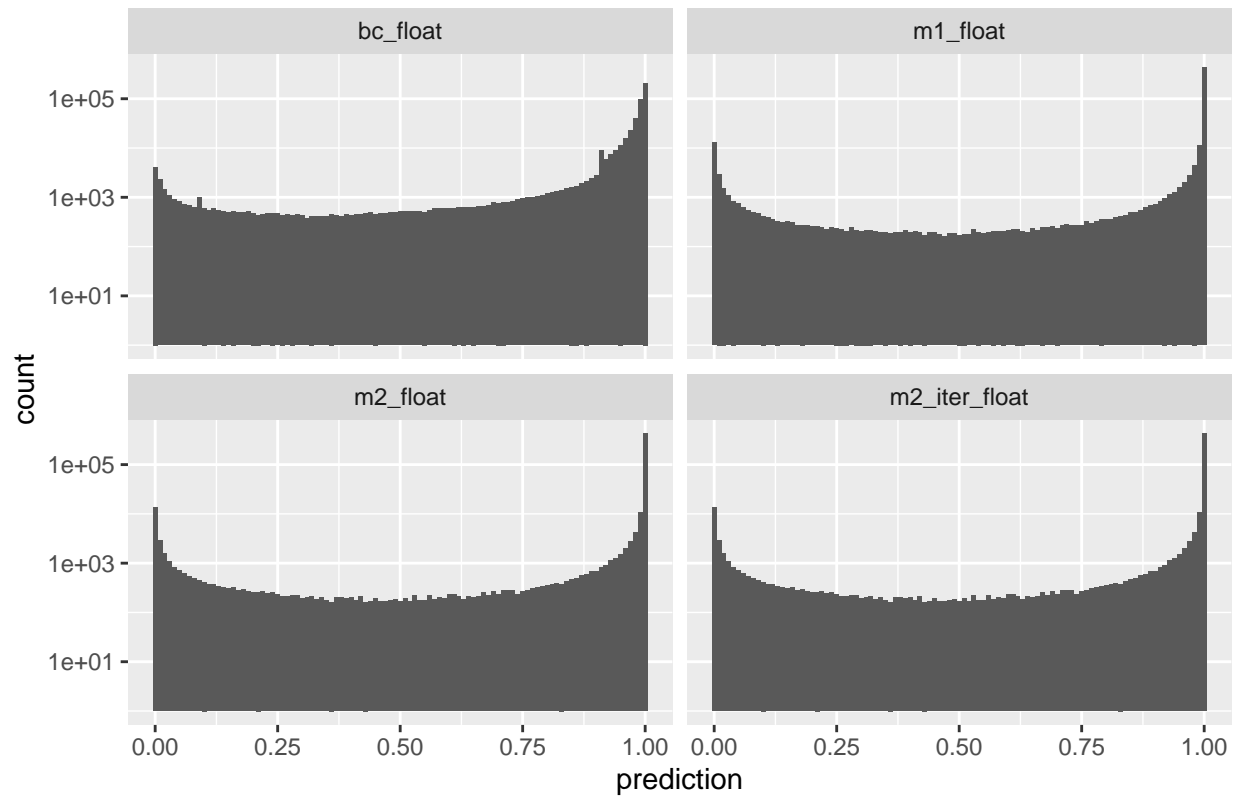
```

```

val_ens_cor_preds_histo <- ggplot(data=val_ens_cor_probs) + geom_histogram(mapping=aes(x=prediction), bin
val_ens_cor_preds_histo

```

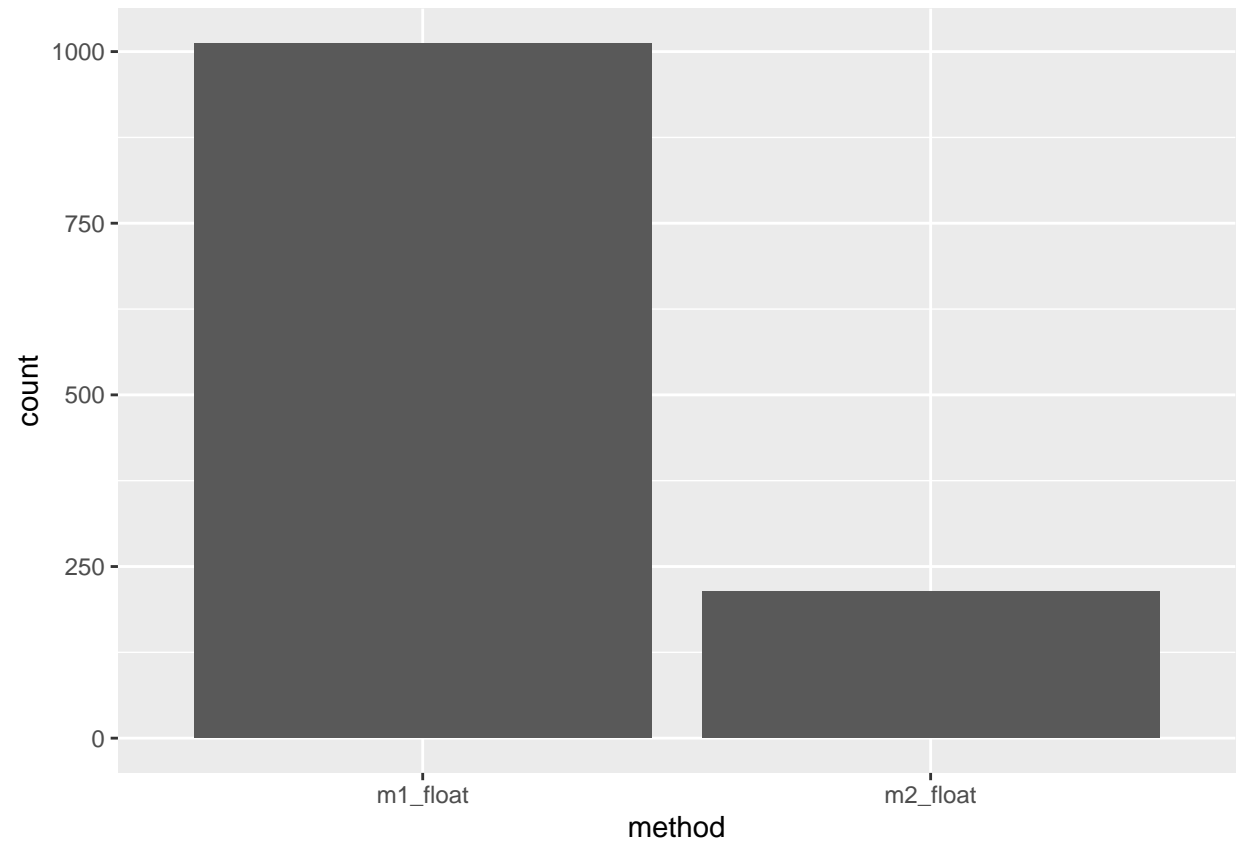
Probabilities predicted for the correct class – ens trained on val



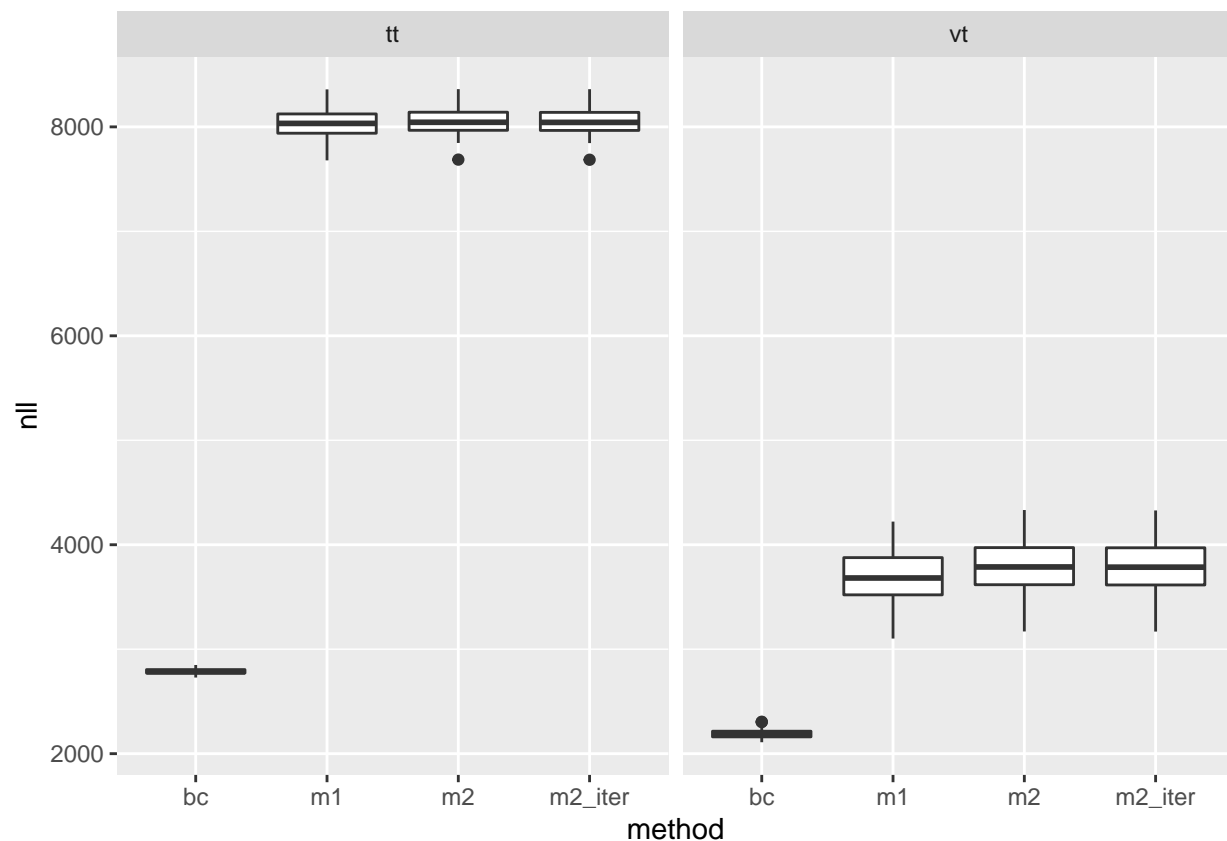
```
val_ens_zero_counts <- ggplot(data=val_ens_cor_probs[val_ens_cor_probs$prediction <= 0, ]) + geom_histogram
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

```
val_ens_zero_counts
```



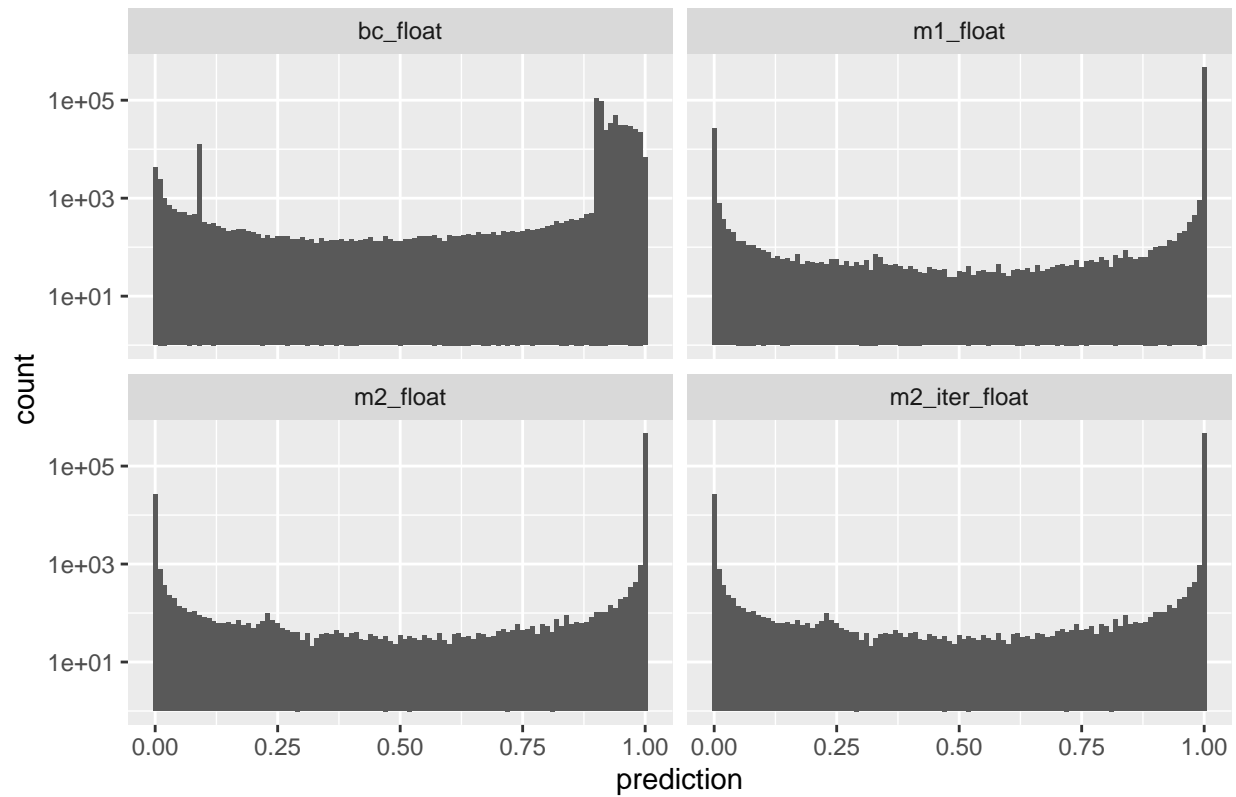
```
val_ens_nll <- ggplot(data=ens_results) + geom_boxplot(mapping=aes(x=method, y=nll)) + facet_wrap(~train)
val_ens_nll
```



```
train_ens_cor_probs <- gather(ens_outputs$train_training, 1 + nets_outputs$test_labels[1, ], 4, 5)
train_ens_cor_probs <- melt(train_ens_cor_probs)
train_ens_cor_probs <- train_ens_cor_probs[, c(-4, -5)]
names(train_ens_cor_probs) <- c("replication", "method", "fold", "prediction")
train_ens_cor_probs$method <- as.factor(train_ens_cor_probs$method)
levels(train_ens_cor_probs$method) <- ens_outputs$methods
```

```
train_ens_cor_preds_histo <- ggplot(data=train_ens_cor_probs) + geom_histogram(mapping=aes(x=prediction))
train_ens_cor_preds_histo
```

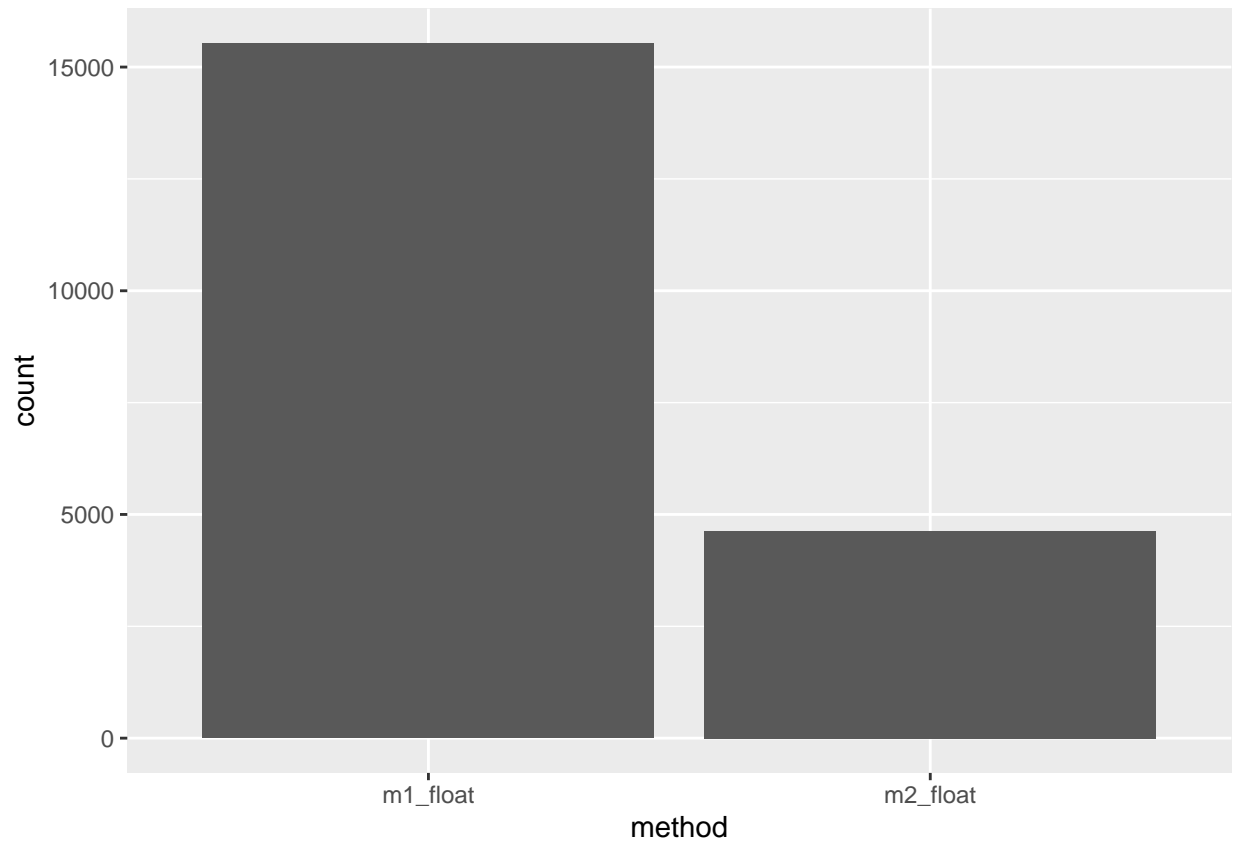
## Probabilities predicted for the correct class – ens trained on train



```
train_ens_zero_counts <- ggplot(data=train_ens_cor_probs[train_ens_cor_probs$prediction <= 0, ]) + geom
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

```
train_ens_zero_counts
```

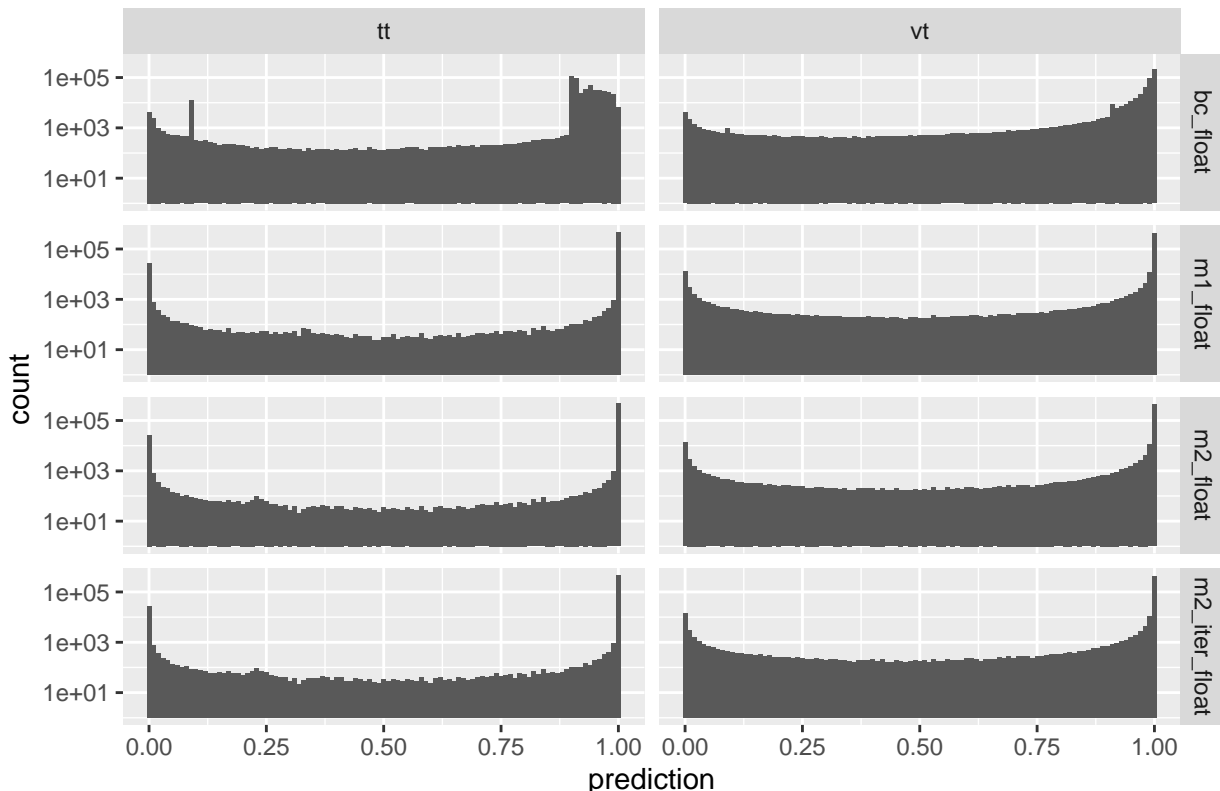


```
val_ens_cor_probs$train_type <- "vt"  
train_ens_cor_probs$train_type <- "tt"  
ens_cor_probs <- rbind(val_ens_cor_probs, train_ens_cor_probs)
```

```
ens_cor_preds_histo <- ggplot(data=ens_cor_probs) + geom_histogram(mapping=aes(x=prediction), binwidth=  
ens_cor_preds_histo
```



## Probabilities predicted for the correct class



Here we observe similar results to those in visualizations ensemble outputs CIF10.

```
aggreg_Rs <- load_class_averaged_R_matrices(base_dir, nets_outputs$test_labels[1, ], repls, folds=folds)
```

[illegible]

[illegible]

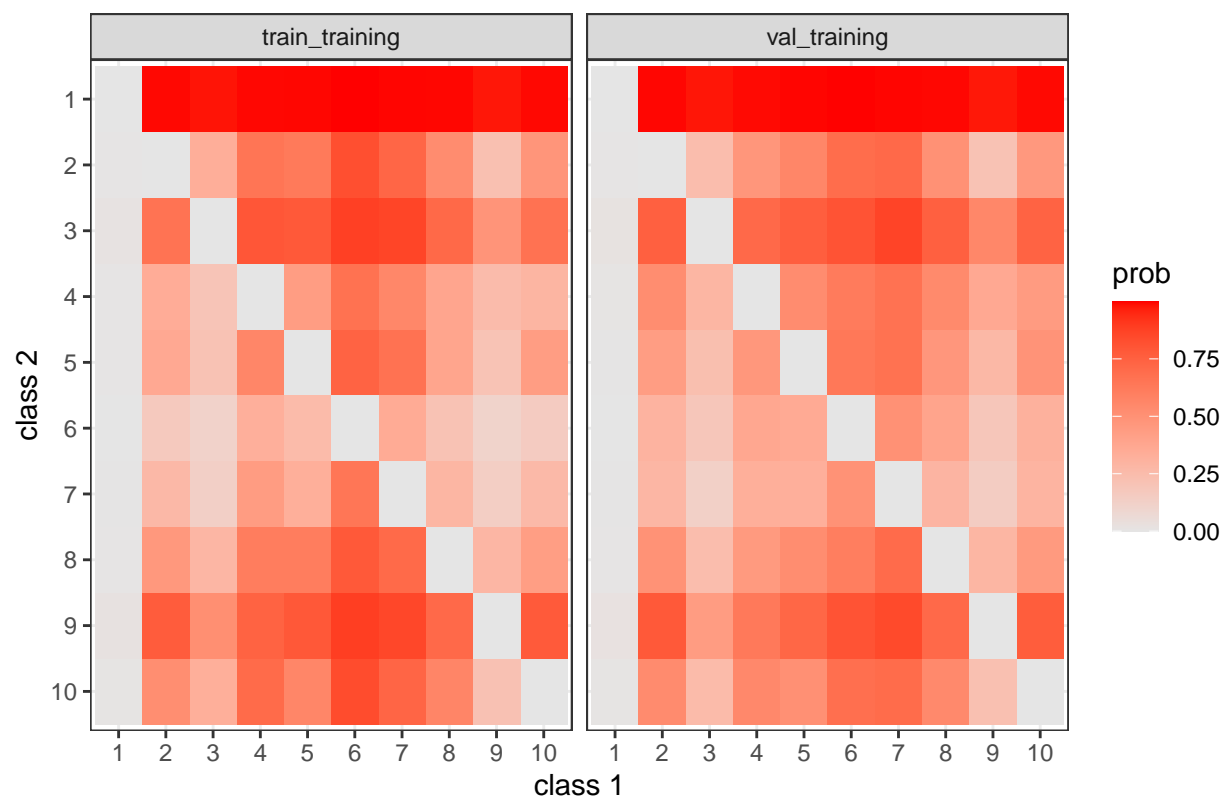
[illegible]

```
## 'summarise()' has grouped output by 'precision', 'train_type', 'class1', 'class2'. You can override
```

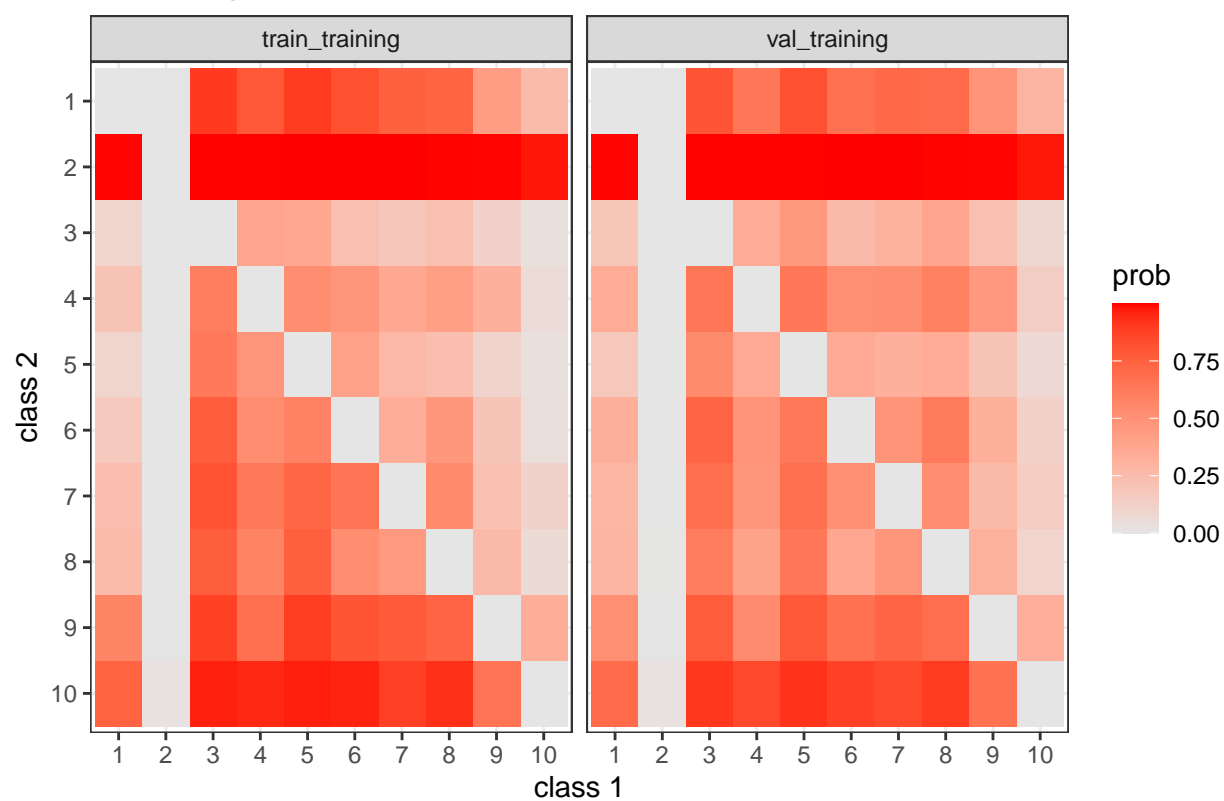
```
for (cls in 1:classes)
{
  cur_class_Rs <- aggreg_Rs %>% filter(class == cls)
  plot_cls <- ggplot(cur_class_Rs, aes(x = class2, y = class1)) +
    geom_raster(aes(fill=prob)) +
    facet_wrap(~train_type) +
    scale_fill_gradient(low="grey90", high="red") +
    scale_y_discrete(limits=rev) +
    labs(x="class 1", y="class 2", title=paste("Pairwise probabilities - class ", cls)) +
    theme_bw()

  print(plot_cls)
}
```

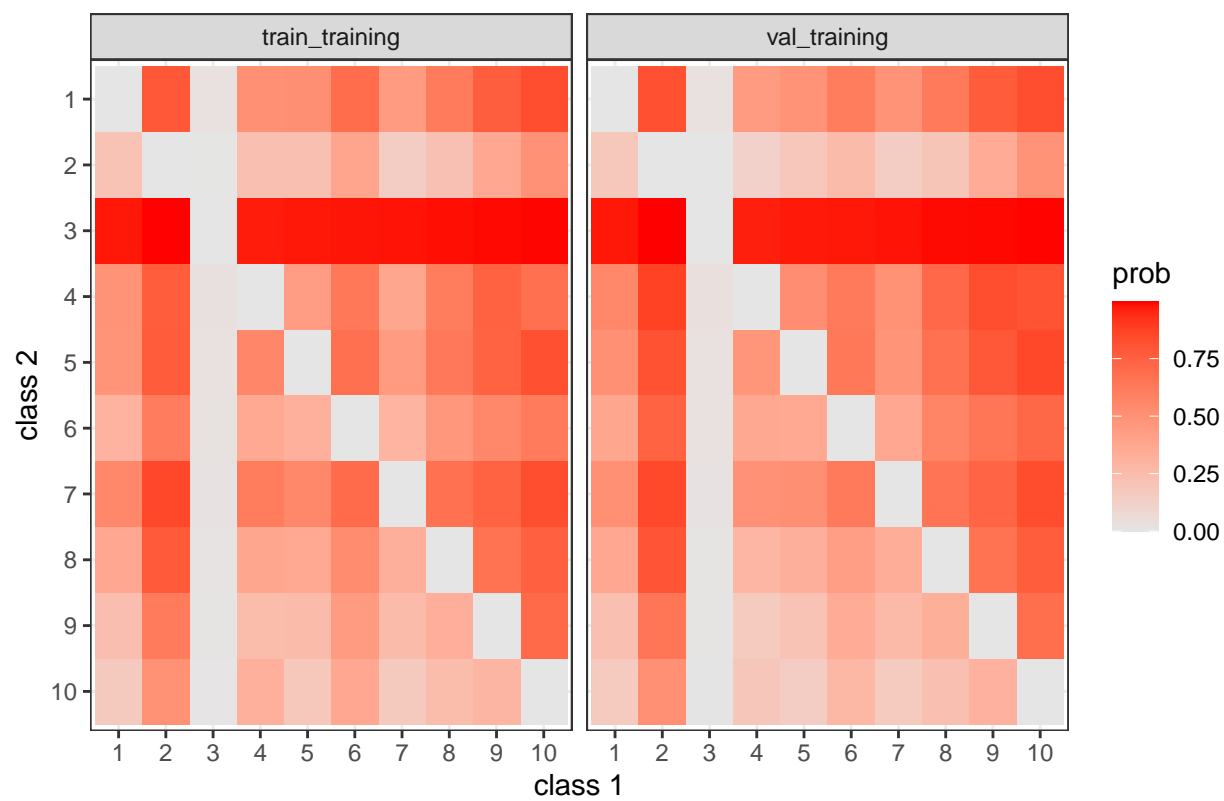
Pairwise probabilities – class 1



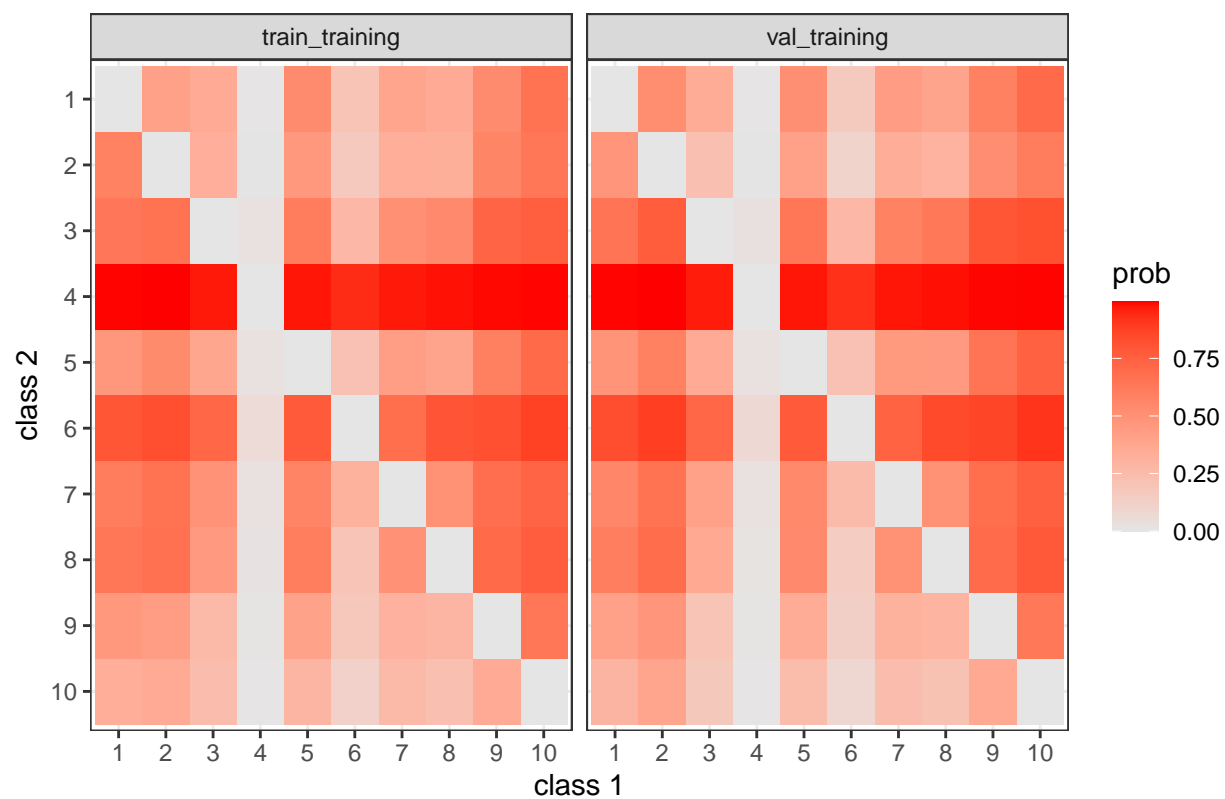
Pairwise probabilities – class 2



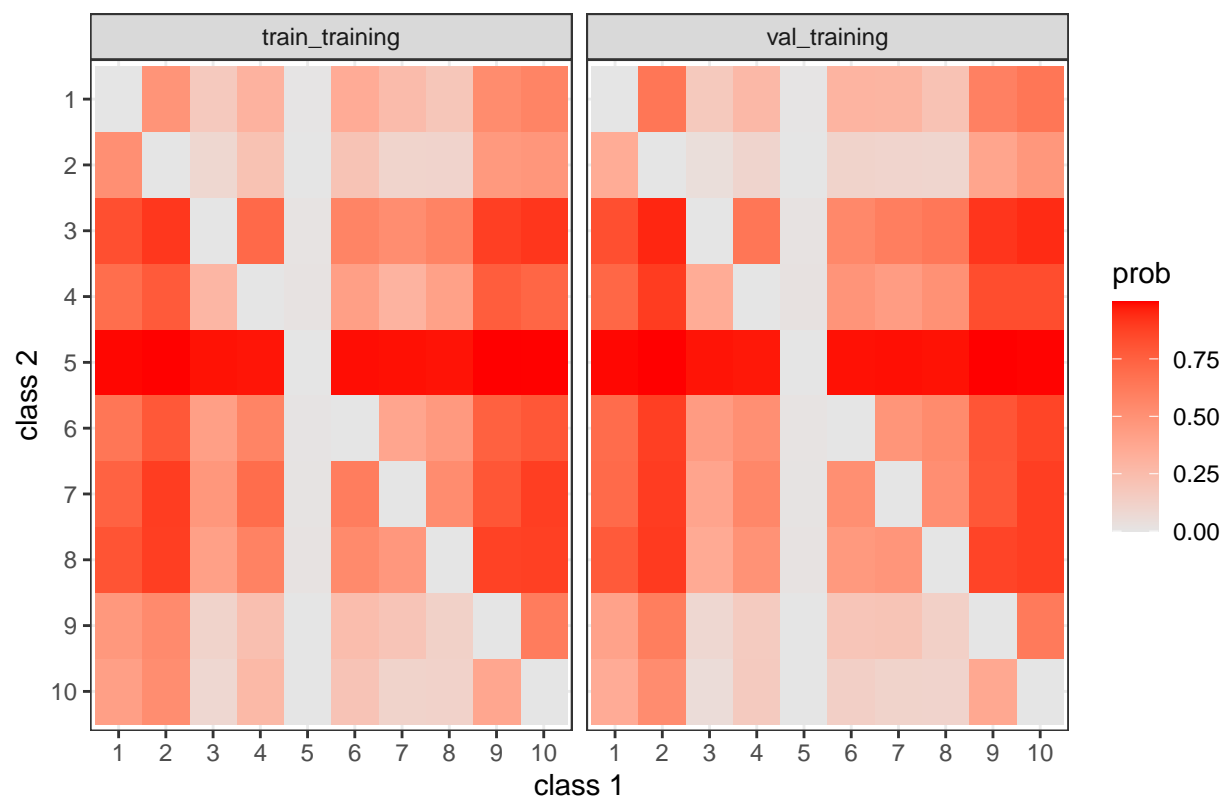
Pairwise probabilities – class 3



Pairwise probabilities – class 4

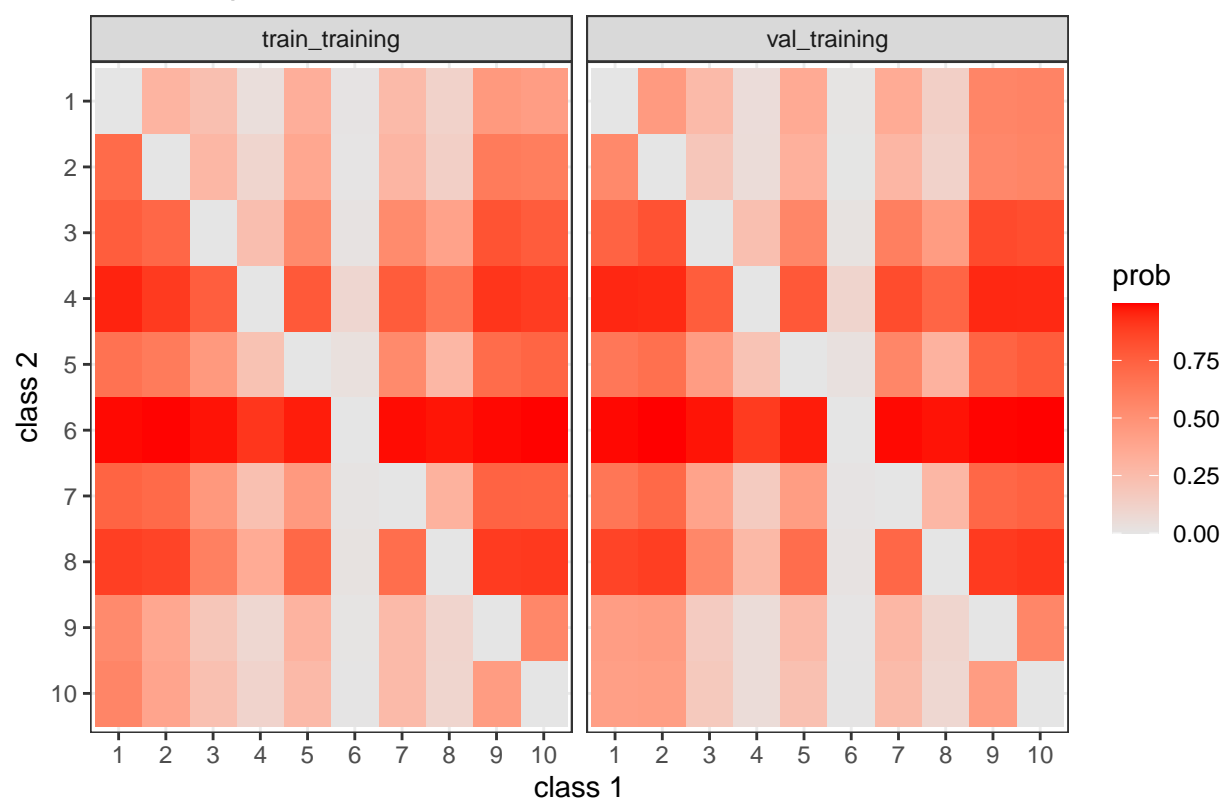


Pairwise probabilities – class 5

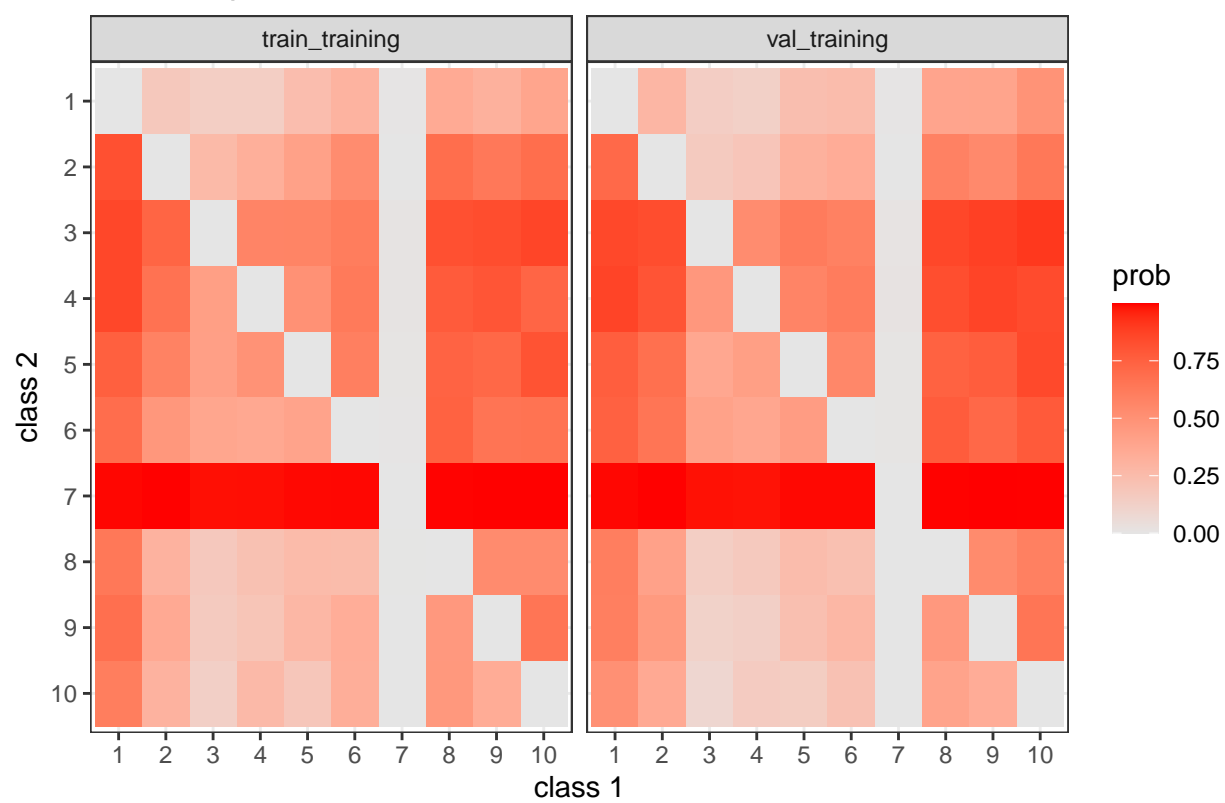




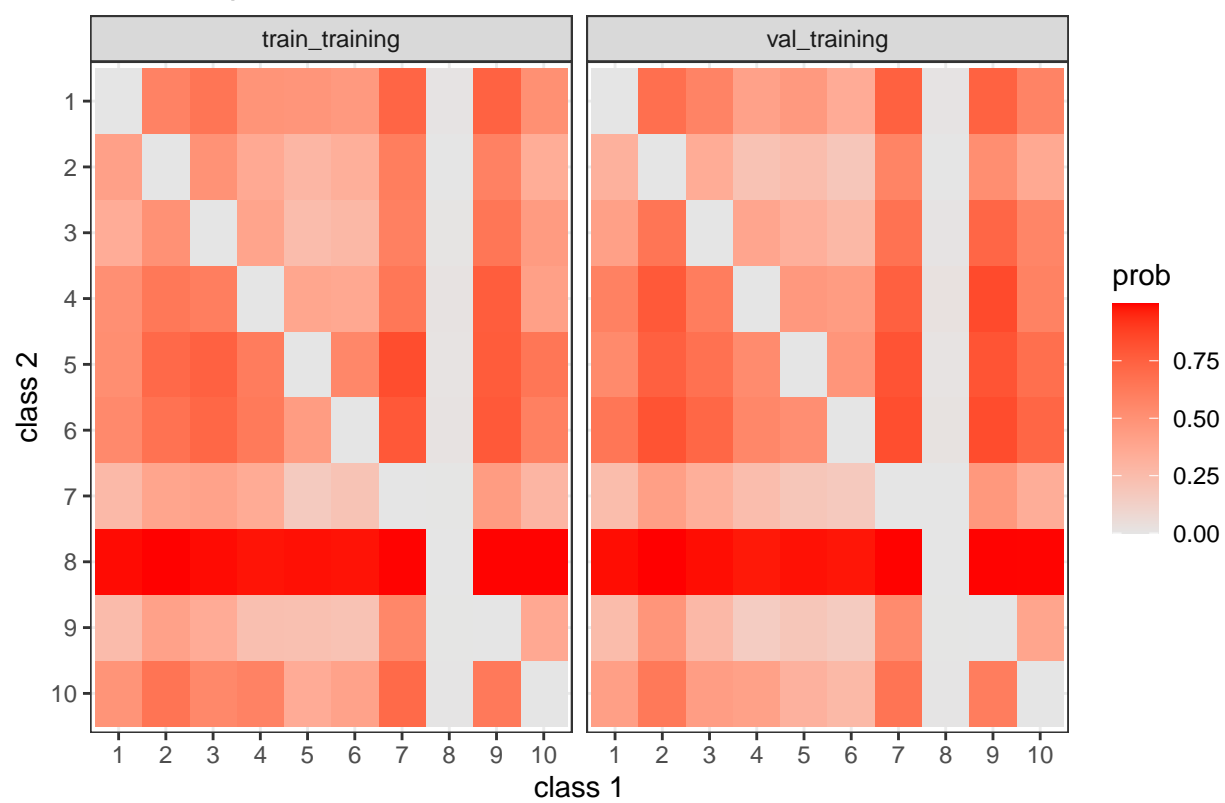
Pairwise probabilities – class 6



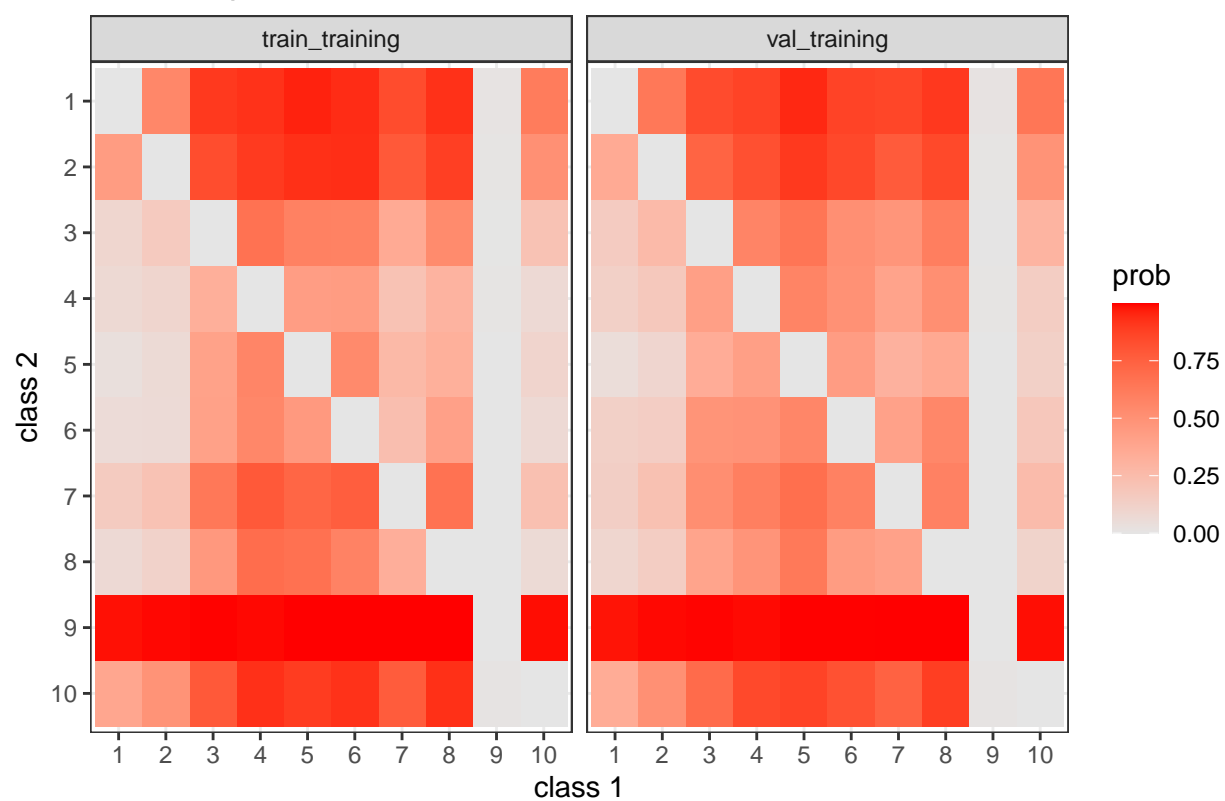
Pairwise probabilities – class 7



Pairwise probabilities – class 8



Pairwise probabilities – class 9



Pairwise probabilities – class 10

