```
test_data = testing(split)
```

#recipe

```
#data preprocessing:
#1.remove low variance features
#2.remove highly correlated features (threshold is 0.9)
#3.scale the features
dat_recipe = recipe(class~.,data=train_data) %>% step_nzv(all_predictors()) %>% step_corr(all_numeric()
```

#create cross validation folds, we are going to use 5fold

```
set.seed(123)
my_folds = bake(dat_recipe,new_data = train_data) %>% vfold_cv(v=5)
```

#linear discriminant model has no parameters to tune

```
lda_mod = discrim_linear(mode = "classification",
  engine = "MASS")
```

#auc plot and confusion matrix

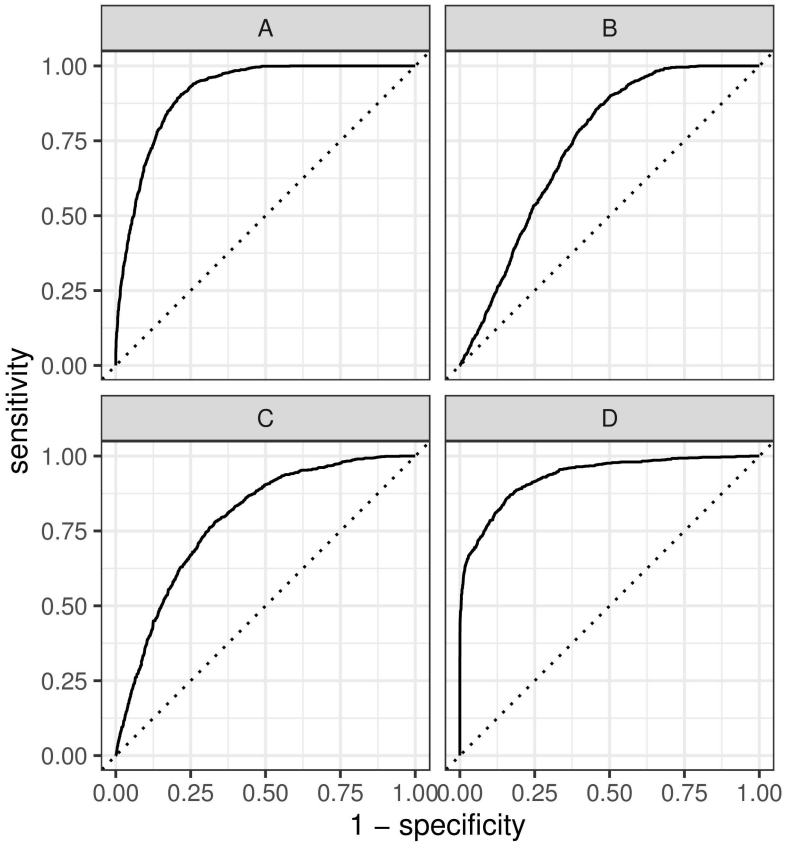
```
lda_mod = workflow() %>% add_recipe(dat_recipe) %>% add_model(lda_mod) %>% last_fit(split)

lda_mod %>% show_best(metric="accuracy")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric  .estimator  mean     n std_err .config
##   <list>     <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> accuracy multiclass 0.595     1      NA Preprocessor1_Model1
```

```
lda_mod %>% show_best(metric="roc_auc")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric .estimator  mean     n std_err .config
##   <list>     <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> roc_auc hand_till  0.841     1      NA Preprocessor1_Model1
```

```
lda_mod %>% collect_predictions() %>% roc_curve(class,.pred_A:.pred_D) %>% autoplot()
```




```
(lda_preds =lda_mod %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_class))
```

```
##            Truth
## Prediction   A   B   C   D
##          A 715 231  88  11
##          B 290 432 246  77
##          C  20 284 524 200
##          D   0  19 160 721
```

#random forest model

```
rf_mod = rand_forest(
  mtry = tune(),
  min_n = tune(),
  trees= 500) %>% set_mode("classification") %>% set_engine("ranger")
```

#rf work flow

```
rf_wf = workflow() %>% add_recipe(dat_recipe)%>% add_model(rf_mod)
```

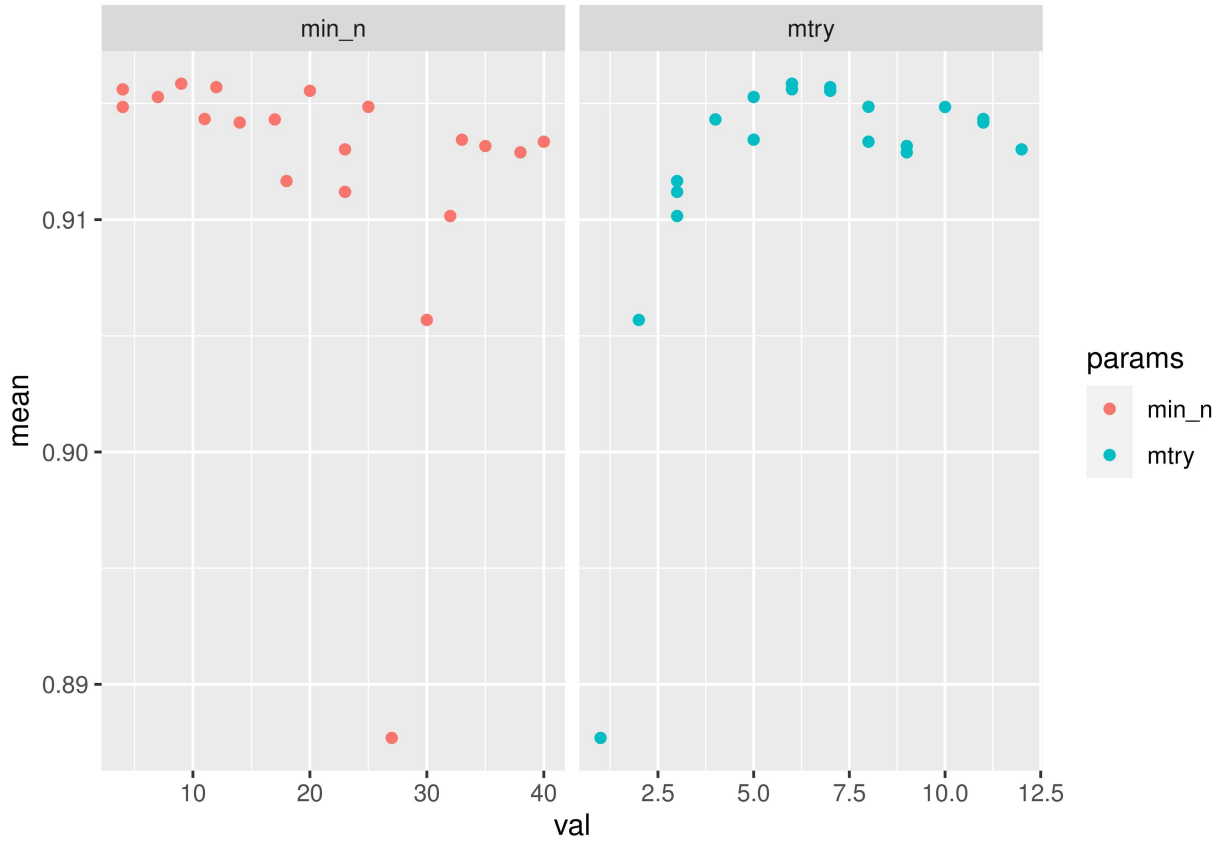
#rf results

```
set.seed(123)
rf_results = rf_wf %>% tune_grid(
                                  resamples = my_folds,
                                  metrics = metric_set(roc_auc),
                                  grid=20
                                  )
```

```
## i Creating pre-processing data to finalize unknown parameter: mtry
```

#visualization for tuning

```
rf_results %>% collect_metrics() %>%
  filter(.metric == "roc_auc") %>%
  pivot_longer(min_n:mtry,values_to = "val",names_to = "params") %>%
  ggplot(aes(val,mean,color = params)) + geom_point() + facet_wrap(~params,scales="free_x")
```




```
#looking at this plot we can see that we want lower values of min_n and values between ~4 and ~8 for mt
```

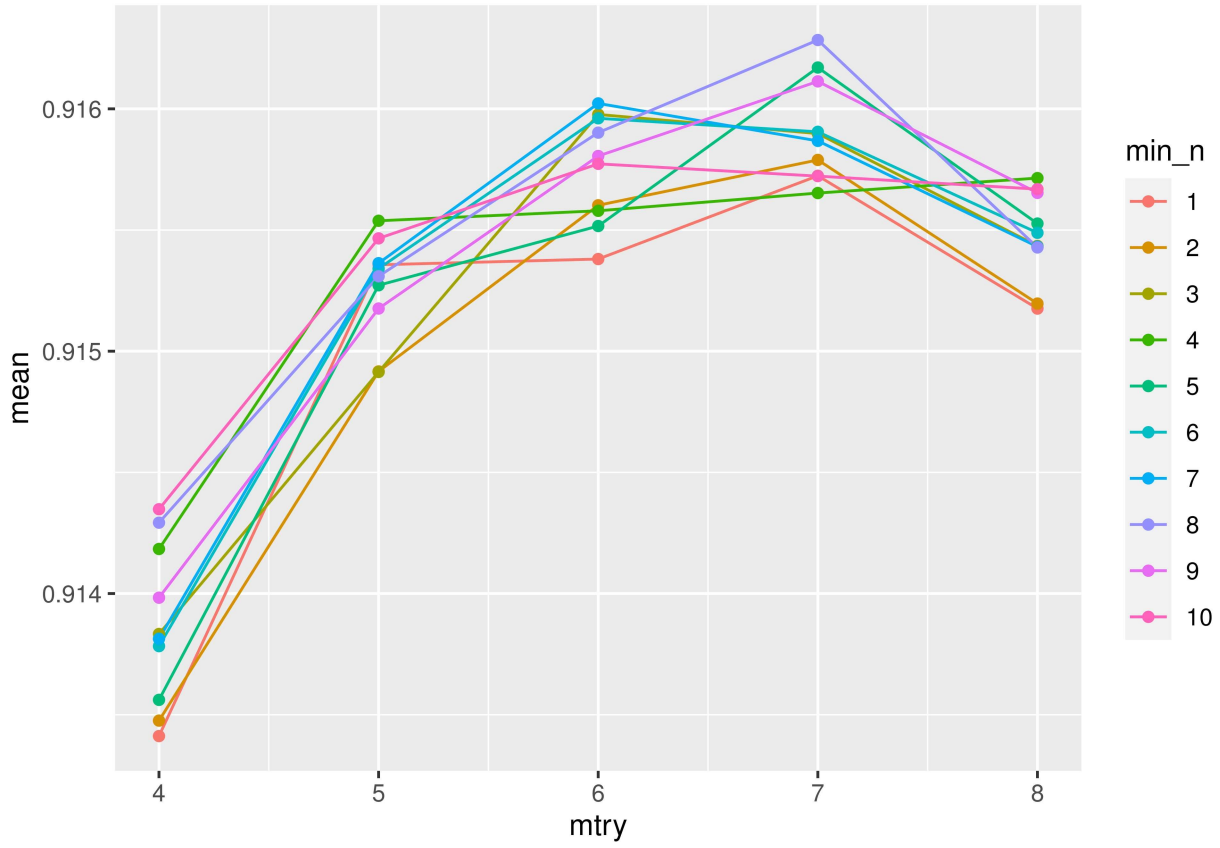
#narrowing the best fit

```
rf_grid = expand.grid(mtry = seq(4,8),min_n = seq(1,10))

set.seed(123)
rf_tune_res = rf_wf %>% tune_grid(resamples = my_folds,
```

```
                                control = control_grid(save_pred=TRUE,save_workflow = TRUE),
                                grid = rf_grid
                                )
```

#plotting the best combination of parameters

```
#plot a visualization
rf_tune_res %>% collect_metrics() %>% filter(.metric == "roc_auc") %>%
  mutate(min_n = factor(min_n)) %>% ggplot(aes(mtry,mean,color=min_n)) + geom_line() +geom_point()
```




```
#looks like the best combination is mtry =7, and min_n =8

best_rf_params = rf_tune_res %>% select_best(metric= "roc_auc")
```

#accuracy roc_auc / auc plot and confusion matrix

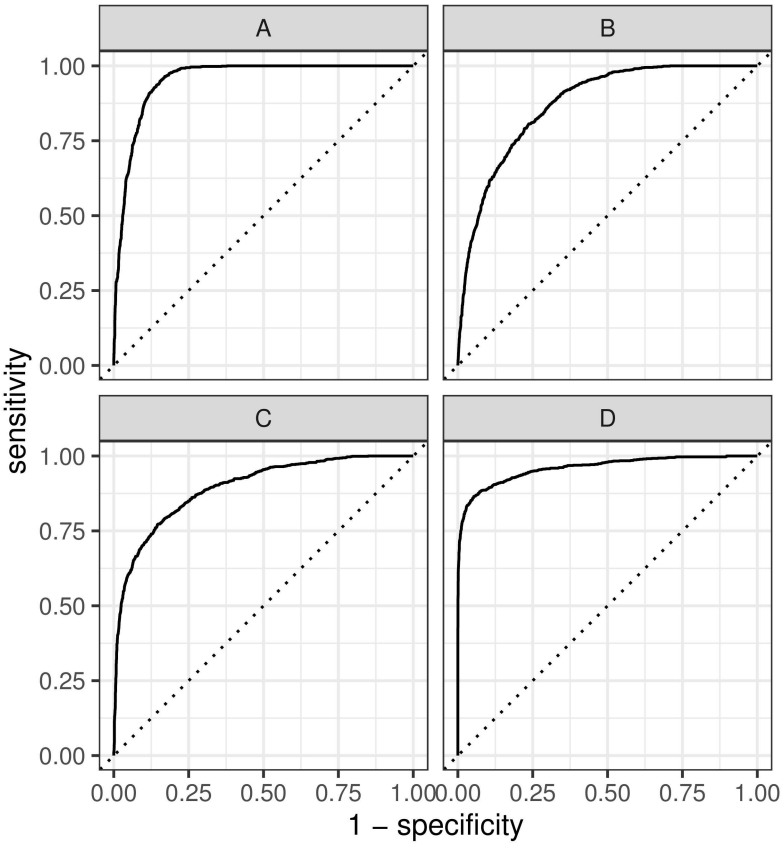
```
final_rf_mod = rf_wf %>% finalize_workflow(best_rf_params)

#using the model on the test set
final_rf_fit = final_rf_mod %>% last_fit(split)

#roc_auc
final_rf_fit %>% show_best(metric="roc_auc")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric .estimator  mean     n std_err .config
##   <list>     <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> roc_auc hand_till  0.919     1      NA Preprocessor1_Model1
```

```
#accuracy
final_rf_fit %>% show_best(metric="accuracy")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric  .estimator  mean     n std_err .config
##   <list>     <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> accuracy multiclass 0.750     1      NA Preprocessor1_Model1
```

```
#auc plot of the test set
final_rf_fit %>% collect_predictions() %>% roc_curve(class,.pred_A:.pred_D) %>% mutate(model = "Random
```




```
#get confusion matrix
(test_preds = final_rf_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_class
```

```
##           Truth
## Prediction   A   B   C   D
##          A 870 197  89  13
##          B 146 616 194  64
##          C   8 112 712 118
##          D   1  41  23 814
```

#Logistic Multinomial

```
lr_mod = multinom_reg(
  mode = "classification",
  engine = "nnet",
  penalty = NULL,
  mixture = NULL
  )
```

#workflow

```
lr_wf = workflow() %>% add_recipe(dat_recipe)%>% add_model(lr_mod)
```

#accuracy roc_auc / auc plot and confusion matrix

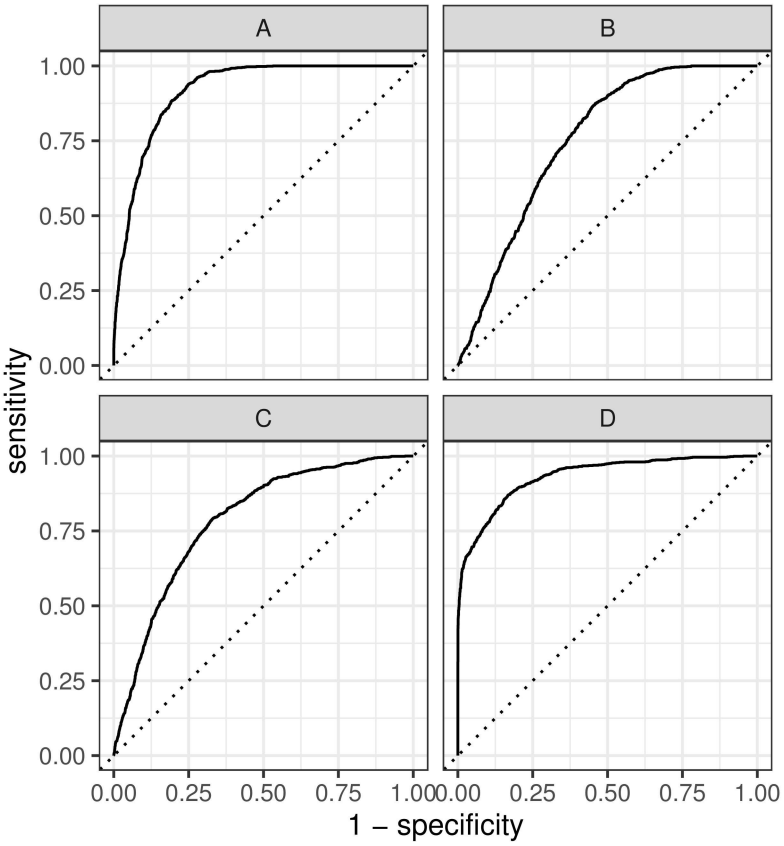
```
lr_fit = lr_wf %>% last_fit(split)


#roc_auc
lr_fit %>% show_best(metric="roc_auc")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric .estimator  mean     n std_err .config
##   <list>     <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> roc_auc hand_till  0.846     1      NA Preprocessor1_Model1
```

```
#accuracy
lr_fit %>% show_best(metric="accuracy")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric  .estimator  mean     n std_err .config
##   <list>     <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> accuracy multiclass 0.603     1      NA Preprocessor1_Model1
```

```
#auc plot of the test set
lr_fit %>% collect_predictions() %>% roc_curve(class,.pred_A:.pred_D) %>% mutate(model = "multinomial")
```




```
#get confusion matrix
(lr_test_preds = lr_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_class))
```

```
##           Truth
## Prediction   A   B   C   D
##          A 733 218  90  14
##          B 277 434 225  60
##          C  15 281 492 173
##          D   0  33 211 762
```

#KNN

```
knn_mod=nearest_neighbor(
  mode = "classification",
  engine = "kknn",
  neighbors = tune(),
  weight_func = tune(),
  dist_power = NULL
)
```

#knn workflow

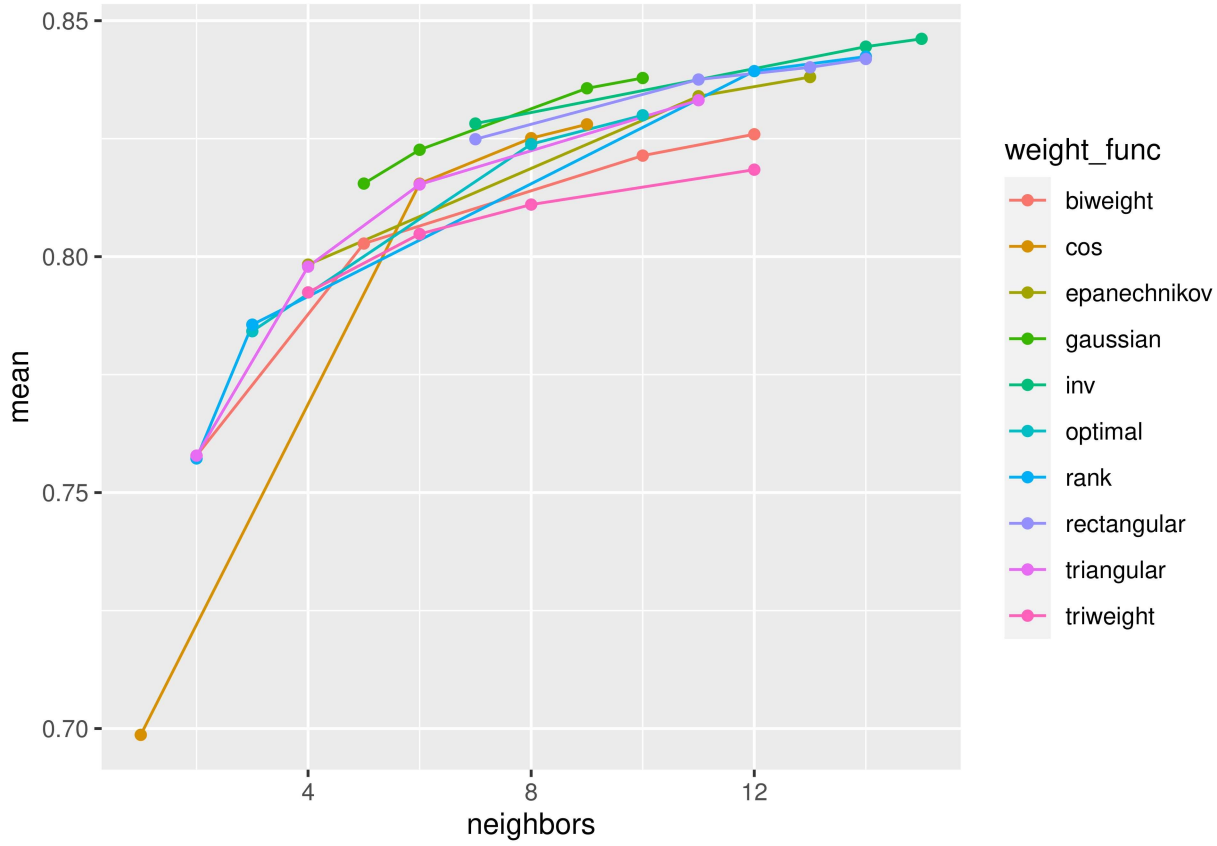
```
knn_wf = workflow() %>% add_recipe(dat_recipe) %>% add_model(knn_mod)
```

#knn tune

```
set.seed(123)
knn_results = knn_wf %>% tune_grid(
                                   resamples = my_folds,
                                   metrics = metric_set(roc_auc),
                                   grid=40
                                   )
```

#Visualization of n neighbors

```
#plot a visualization
knn_results %>% collect_metrics() %>% filter(.metric == "roc_auc") %>% mutate("weight_func" = factor(we
ggplot(aes(neighbors,mean, color = weight_func)) + geom_line() +geom_point()
```

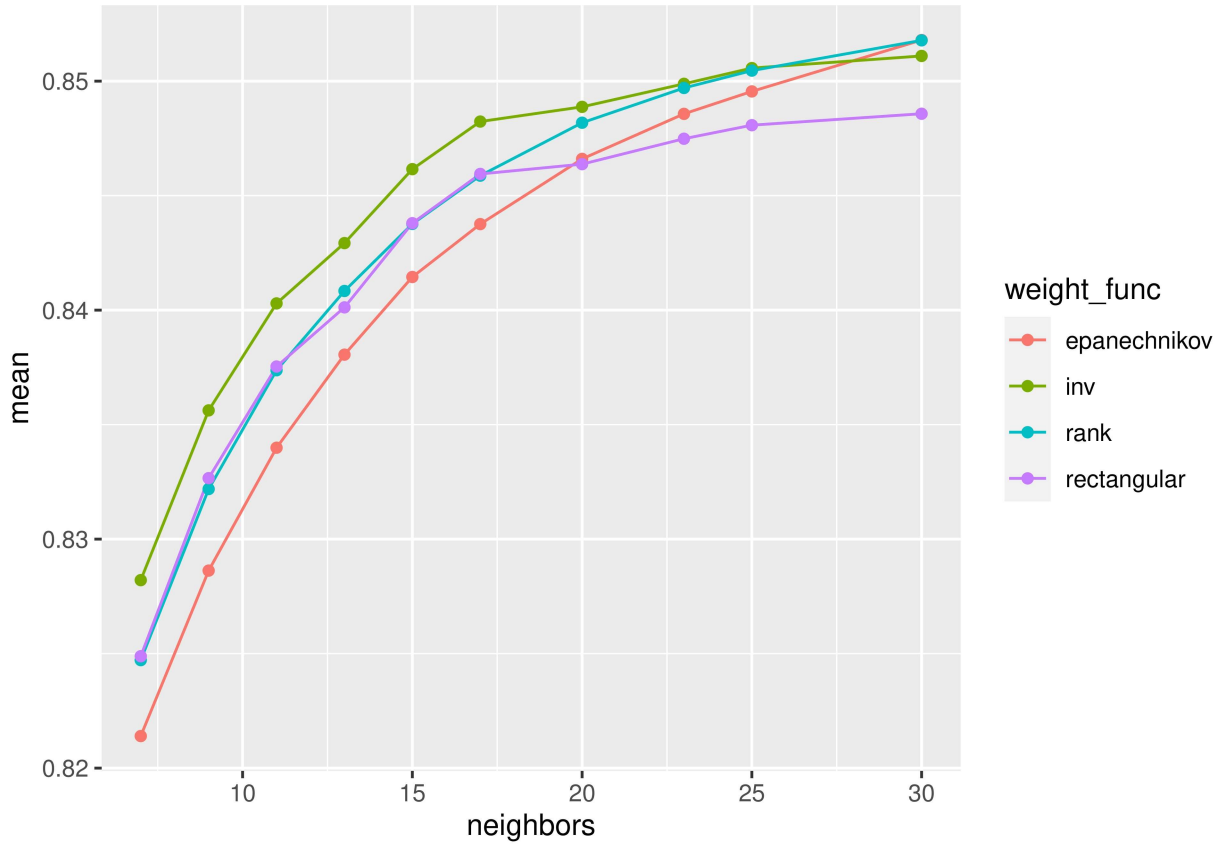



```
#looks like more neighbors with weight functions: epanechnikov, inv, rank, and rectangular produces hig
```

#knn tune

```
knn_grid = expand.grid(weight_func = c("epanechnikov","inv","rank","rectangular"), neighbors = c(7,9,11

set.seed(123)
knn_tune_res = knn_wf %>% tune_grid(resamples = my_folds,
                                   control = control_grid(save_pred=TRUE,save_workflow = TRUE),
                                   grid = knn_grid
                                   )
```

```
#plot a visualization
knn_tune_res %>% collect_metrics() %>% filter(.metric == "roc_auc") %>% mutate("weight_func" = factor(w
ggplot(aes(neighbors,mean, color = weight_func)) + geom_line() +geom_point()
```




```
(best_knn_params = knn_tune_res %>% select_best(metric= "roc_auc"))
```

```
## # A tibble: 1 x 3
##   neighbors weight_func  .config
##       <dbl> <chr>        <chr>
## 1        30 epanechnikov Preprocessor1_Model10
```

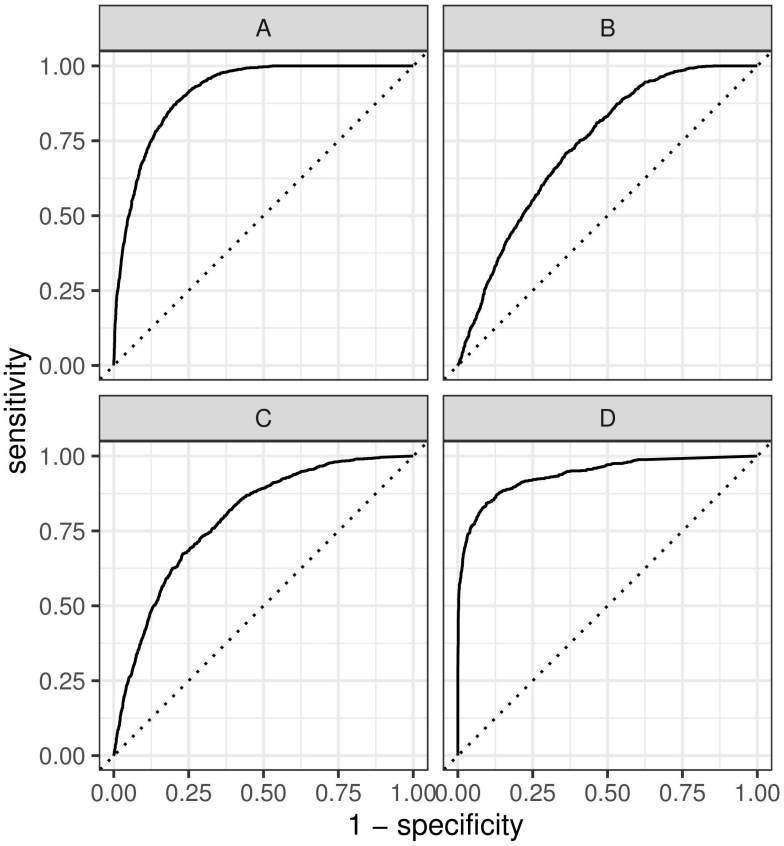
```
final_knn_mod = knn_wf %>% finalize_workflow(best_knn_params)

#using the model on the test set
final_knn_fit = final_knn_mod %>% last_fit(split)

#roc_auc
final_knn_fit %>% show_best(metric="roc_auc")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric .estimator  mean     n std_err .config
##   <list>     <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> roc_auc hand_till  0.845     1      NA Preprocessor1_Model1
```

```
#accuracy
final_knn_fit %>% show_best(metric="accuracy")
```

```
## # A tibble: 1 x 7
##   .workflow  .metric  .estimator  mean     n std_err .config
##   <list>     <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 <workflow> accuracy multiclass 0.613     1      NA Preprocessor1_Model1
```

```
#auc plot of the test set
final_knn_fit %>% collect_predictions() %>% roc_curve(class,.pred_A:.pred_D) %>% autoplot()
```




```
#get confusion matrix
(knn_test_preds = final_knn_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_
```

```
##            Truth
## Prediction   A   B   C   D
##          A 799 314 132  18
##          B 206 465 314  75
##          C  20 162 526 244
##          D   0  25  46 672
```

#XGBoost

```
XGBoost_mod = boost_tree( mode = "classification",
                      engine = "xgboost",
                      mtry = tune(),
                      trees = 800,
                      tree_depth = tune(),
                      min_n = tune(),
                      loss_reduction = tune(),
                      sample_size = tune(),
                      learn_rate = tune()
                      )


XGB_grid = grid_max_entropy(finalize(mtry(),train_data),
                            learn_rate(),
                            sample_size = sample_prop(),
                            min_n(),
                            tree_depth(),
                            loss_reduction(),
                            size=25)
```

#xgb wf

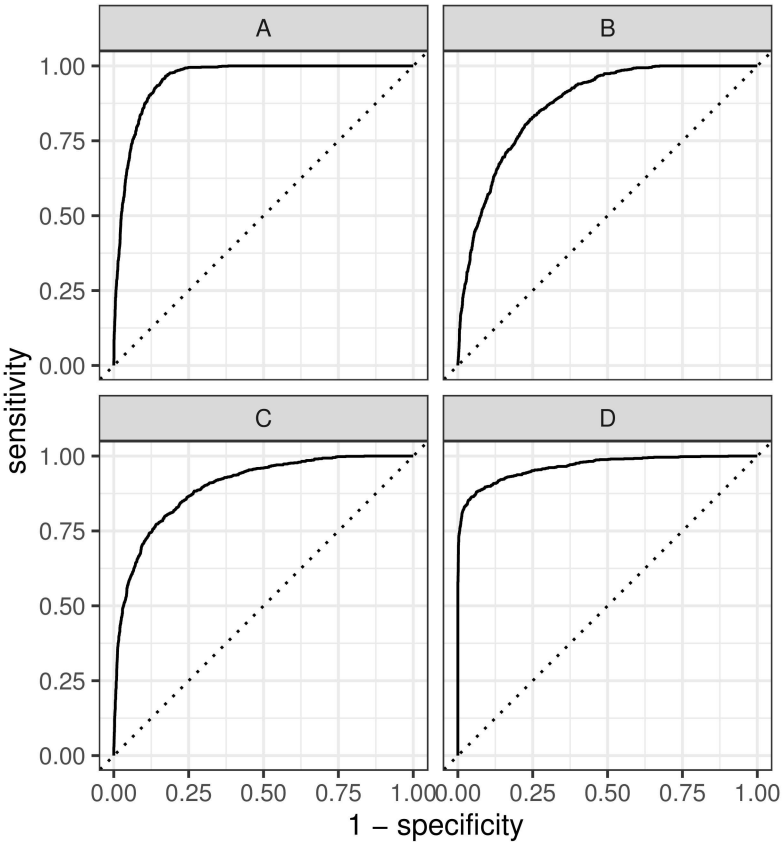
```
XGB_wf = workflow() %>% add_recipe(dat_recipe) %>% add_model(XGBoost_mod)
```

```
set.seed(123)
XGB_tune_res = XGB_wf %>% tune_grid(resamples = my_folds,
                                    control = control_grid(save_pred=TRUE,save_workflow = TRUE),
                                    grid = XGB_grid
                                    )
```

```
best_XGB_params = XGB_tune_res %>% select_best(metric= "roc_auc")


final_XGB_mod = XGB_wf %>% finalize_workflow(best_XGB_params)


#using the model on the test set
final_XGB_fit  = final_XGB_mod %>% last_fit(split)

#roc_auc and accuracy
final_XGB_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy multiclass     0.752 Preprocessor1_Model1
## 2 roc_auc  hand_till      0.922 Preprocessor1_Model1
```

```
#auc plot of the test set
final_XGB_fit %>% collect_predictions(parameters = best_XGB_params) %>% roc_curve(class,.pred_A:.pred_D
```




```
#get confusion matrix
(XGB_test_preds = final_XGB_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_
```

```
##            Truth
## Prediction   A   B   C   D
##          A 896 212  93  11
##          B 121 593 192  59
##          C   7 133 704 112
##          D   1  28  29 827
```

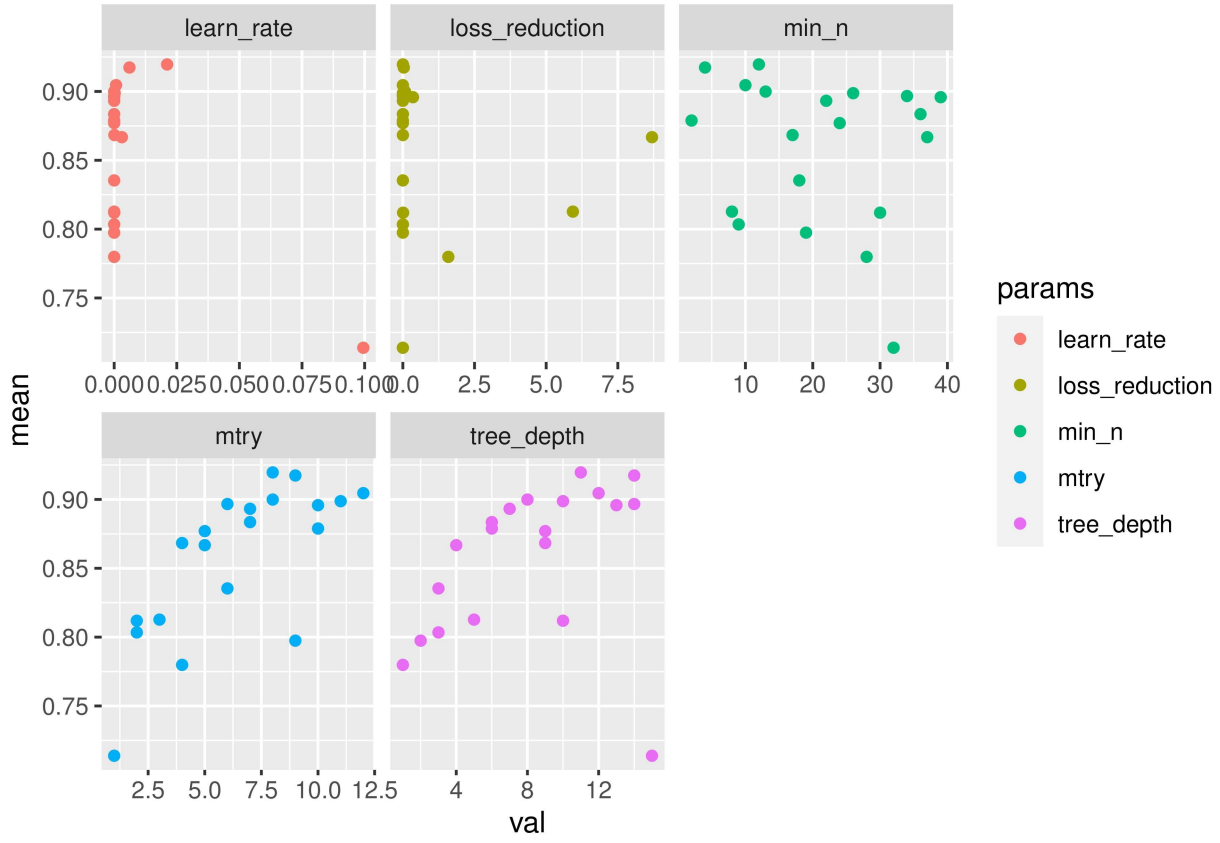
#LightGBM

```
set_dependency("boost_tree", eng = "lightgbm", "lightgbm")
set_dependency("boost_tree", eng = "lightgbm", "treesnip")


LightGBM_mod = boost_tree(mode = "classification",
                          engine = "lightgbm",
                          mtry = tune(),
                          trees = 1000,
                          min_n = tune(),
                          tree_depth = tune(),
                          loss_reduction = tune(),
                          learn_rate = tune(),
                          sample_size = tune())
```

```
LightGBM_wf = workflow() %>% add_recipe(dat_recipe) %>% add_model(LightGBM_mod)



set.seed(123)

LightGBM_tune_res = LightGBM_wf %>% tune_grid(resamples = my_folds,
                                    control = control_grid(save_pred=TRUE,save_workflow = TRUE),
                                    grid = 20
                                    )
```

```
## i Creating pre-processing data to finalize unknown parameter: mtry
```

```
LightGBM_tune_res %>% collect_metrics() %>% filter(.metric == "roc_auc") %>%
  pivot_longer(mtry:loss_reduction,values_to = "val",names_to = "params") %>%
  ggplot(aes(val,mean,color = params)) + geom_point() + facet_wrap(~params,scales="free_x")
```



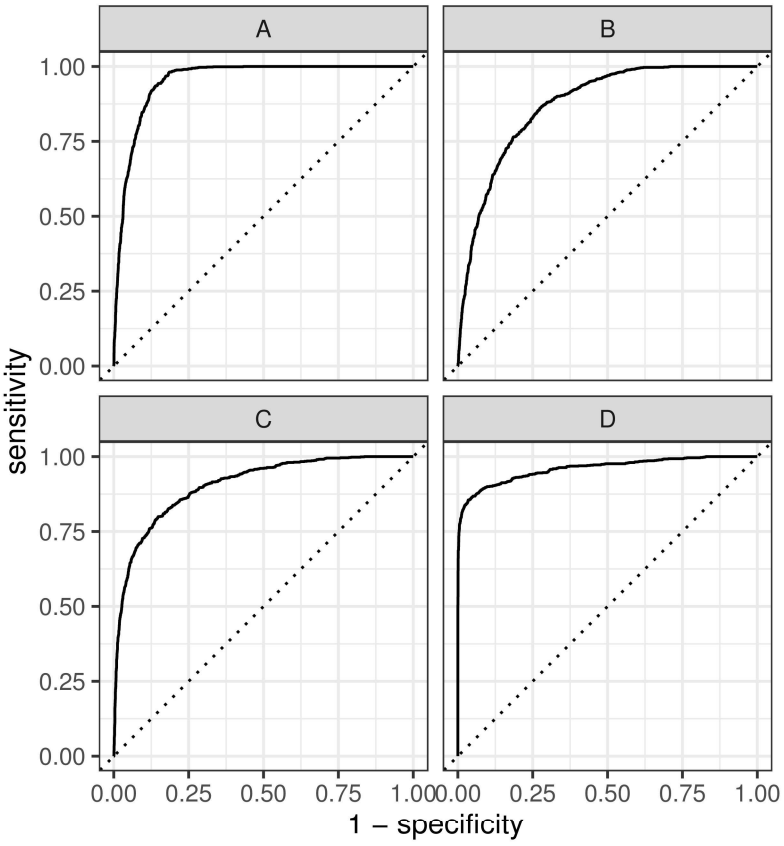

```
best_LightGBM_params = LightGBM_tune_res %>% show_best(metric= "roc_auc")
```

#finalizing LightGBM model

```
final_LightGBM_mod = LightGBM_wf %>% finalize_workflow(best_LightGBM_params[1,])


#using the model on the test set
final_LightGBM_fit  = final_LightGBM_mod %>% last_fit(split)

#roc_auc and accuracy
final_LightGBM_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy multiclass     0.757 Preprocessor1_Model1
## 2 roc_auc  hand_till      0.922 Preprocessor1_Model1
```

```
#auc plot of the test set
final_LightGBM_fit %>% collect_predictions(parameters = best_LightGBM_params) %>% roc_curve(class,.pred
```




```
#get confusion matrix
(LightGBM_test_preds = final_LightGBM_fit %>% collect_predictions() %>% conf_mat(truth = class, estimat
```

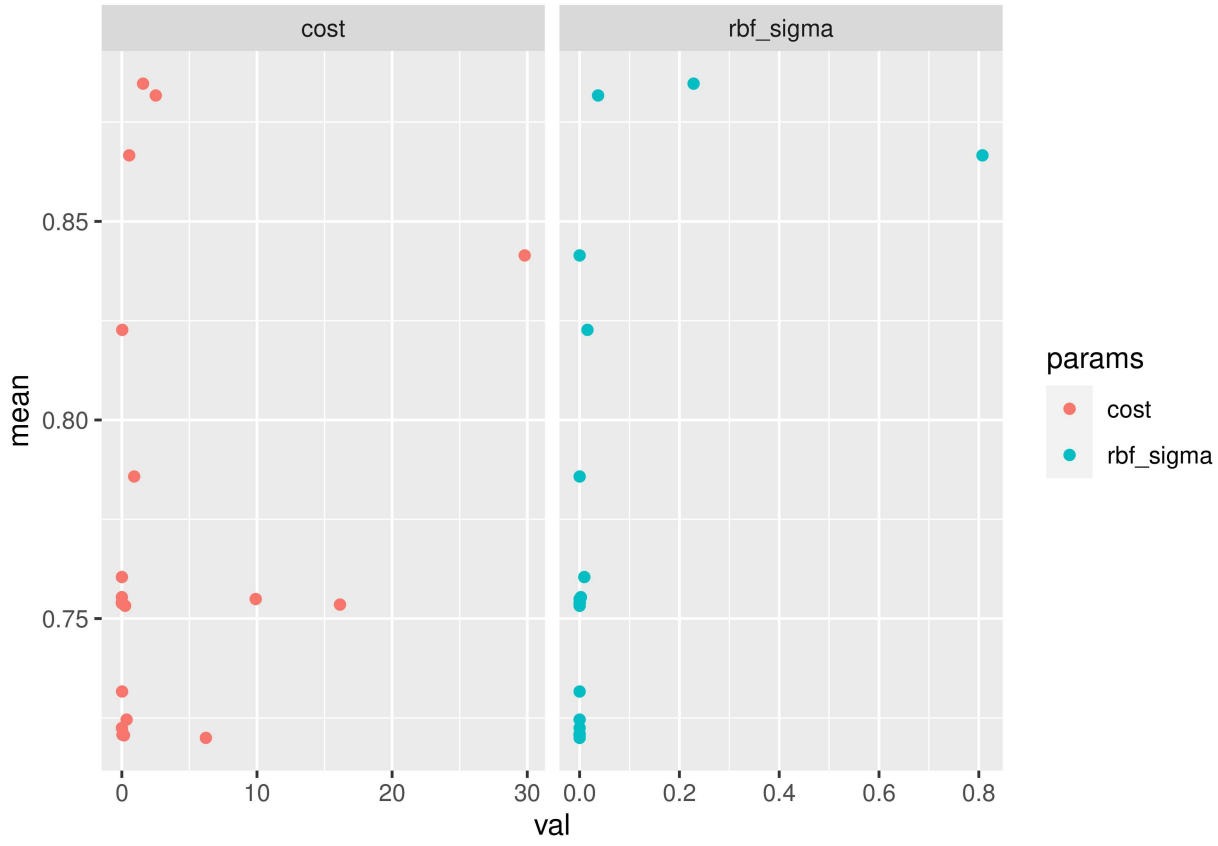
```
##            Truth
## Prediction   A   B   C   D
##          A 860 186  84  12
```

```
##          B 145 640 195  59
##          C  20 113 715 111
##          D   0  27  24 827
```

#SVM rbf model

```
svmRBF_mod =svm_rbf(mode = "classification",
                    engine="kernlab",
                    cost = tune(),
                    rbf_sigma = tune())



svmRBF_wf = workflow() %>% add_recipe(dat_recipe) %>% add_model(svmRBF_mod)




set.seed(123)
svmRBF_tune_res = svmRBF_wf %>% tune_grid(resamples = my_folds,
                                  control = control_grid(save_pred=TRUE),
                                  grid = 20
                                  )
```

```
svmRBF_tune_res %>% collect_metrics() %>% filter(.metric == "roc_auc") %>%
  pivot_longer(cost:rbf_sigma, values_to = "val",names_to = "params") %>%
  ggplot(aes(val,mean,color = params)) + geom_point() + facet_wrap(~params,scales="free_x")
```




```
#we can see that lower cost values are preferred. (can't say for rbf_sigma so we can try a range of num
```

```
svmRBF_grid = expand.grid(cost = c(0.5,1,2,3,4),rbf_sigma = c(0,0.1,0.2,0.4,0.5,0.7))

svmRBF_tune_res = svmRBF_wf %>% tune_grid(resamples = my_folds,
                                  control = control_grid(save_pred=TRUE,save_workflow = TRUE),
                                  grid = svmRBF_grid
                                  )
```

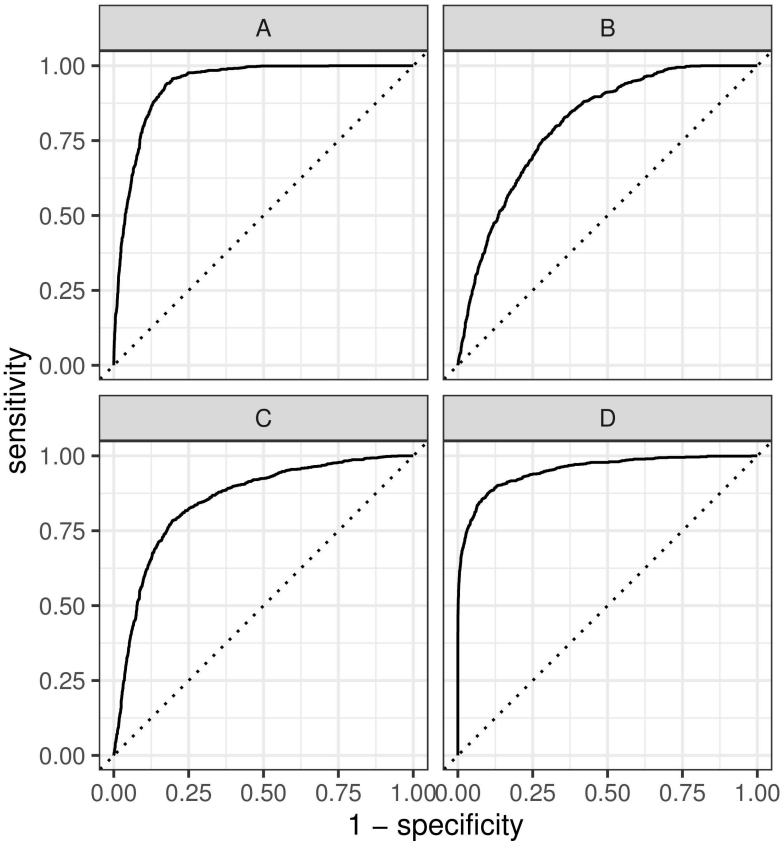
#finalizing SVM_RBF model

```
best_svmRBF_params = svmRBF_tune_res %>% show_best(metric= "roc_auc")

final_svmRBF_mod = svmRBF_wf %>% finalize_workflow(best_svmRBF_params[1,])


#using the model on the test set
final_svmRBF_fit  = final_svmRBF_mod %>% last_fit(split)

#roc_auc and accuracy
final_svmRBF_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
```

```
## 1 accuracy multiclass     0.704 Preprocessor1_Model1
## 2 roc_auc  hand_till      0.886 Preprocessor1_Model1
```

```
#auc plot of the test set
final_svmRBF_fit %>% collect_predictions(parameters = best_svmRBF_params) %>% roc_curve(class,.pred_A:.
```




```
#get confusion matrix
(svmRBFtest_preds = final_svmRBF_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .
```

```
##            Truth
## Prediction   A   B   C   D
##          A 845 226  79  11
##          B 172 545 231  60
##          C   6 172 637 138
##          D   2  23  71 800
```

#multi layer perceptron

```
mlp_mod = mlp(mode="classification",
              engine = "nnet",
              hidden_units = tune(),
              penalty = tune(),
              epochs = tune())
```

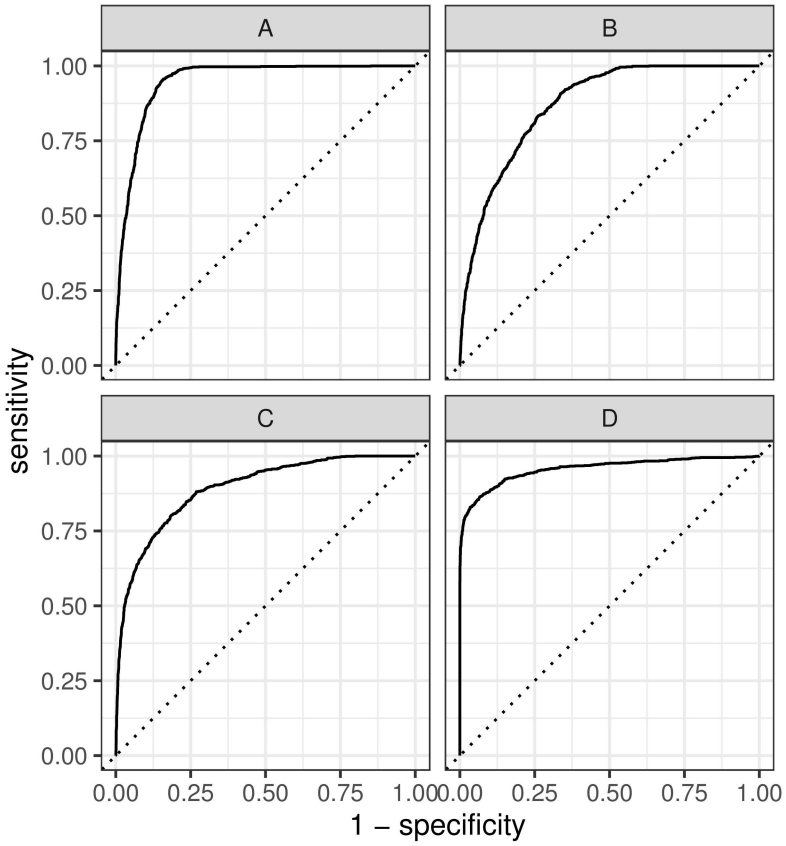
```
mlp_wf = workflow() %>% add_recipe(dat_recipe) %>% add_model(mlp_mod)




set.seed(123)
mlp_tune_res = mlp_wf %>%
  tune_grid(resamples = my_folds,
            control=control_grid(save_pred=TRUE,save_workflow = TRUE),
            grid = 20
                                  )
best_mlp_params = mlp_tune_res %>% show_best(metric= "roc_auc")
```

#finalizing mlp model

```
final_mlp_mod = mlp_wf %>% finalize_workflow(best_mlp_params[1,])


#using the model on the test set
final_mlp_fit  = final_mlp_mod %>% last_fit(split)

#roc_auc and accuracy
final_mlp_fit %>% collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy multiclass     0.738 Preprocessor1_Model1
## 2 roc_auc  hand_till      0.916 Preprocessor1_Model1
```

```
#auc plot of the test set
final_mlp_fit %>% collect_predictions(parameters = best_mlp_params) %>% roc_curve(class,.pred_A:.pred_D
```




```
#get confusion matrix
(mlp_preds = final_mlp_fit %>% collect_predictions() %>% conf_mat(truth = class, estimate = .pred_class
```

```
##            Truth
## Prediction   A   B   C   D
##          A 904 230  96  14
##          B 115 567 193  50
##          C   5 145 682 132
##          D   1  24  47 813
```

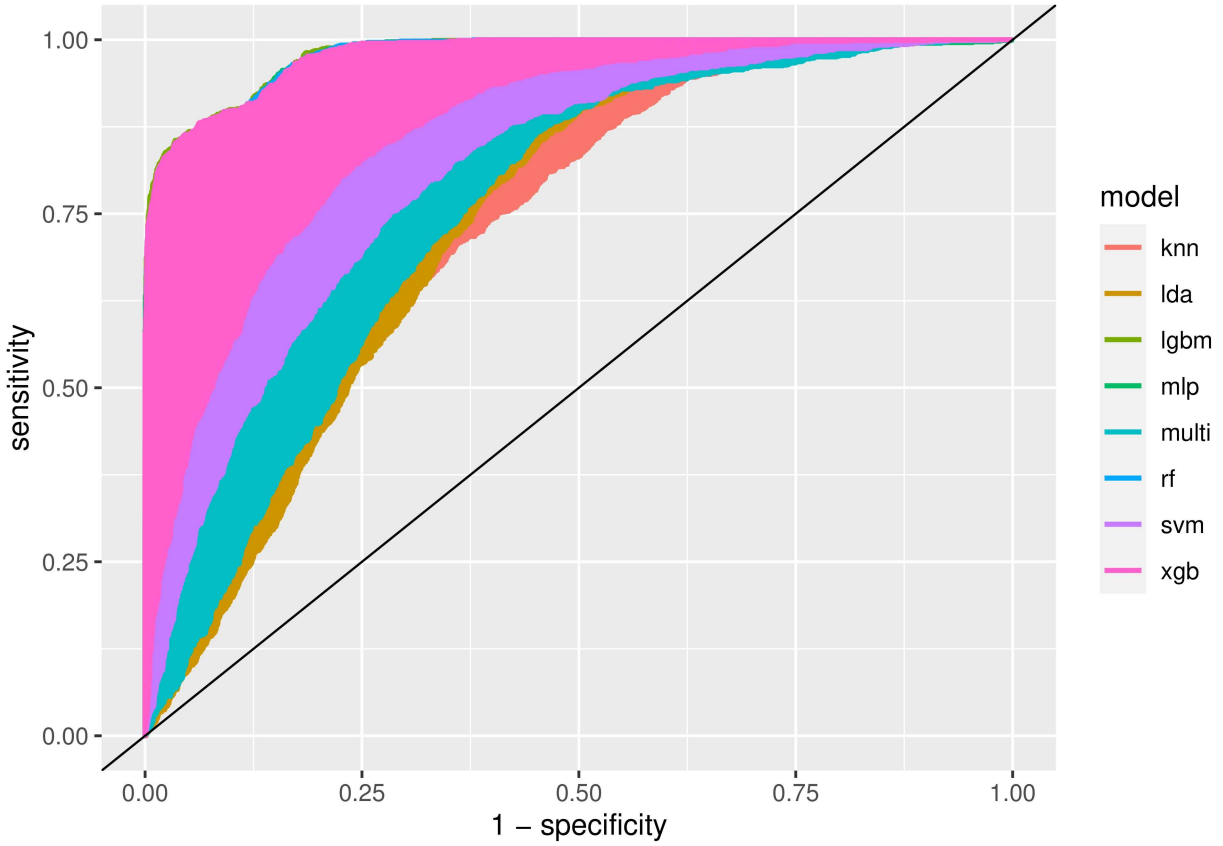
#aggregating the data to make a roc_auc plot

```
#lda
lda_collection = lda_mod %>%
collect_predictions() %>% mutate(.config = "lda")

#logistic multinomial
lr_collection = lr_fit %>% collect_predictions() %>% mutate(.config = "multi")

#svm_rbf
svm_collection = final_svmRBF_fit %>% collect_predictions() %>% mutate(.config = "svm")

#knn
knn_collection = final_knn_fit %>%
collect_predictions(parameters = best_knn_params)%>% mutate(.config = "knn")
```

```
#rf
rf_collection = final_rf_fit %>%
collect_predictions(parameters = best_rf_params)%>% mutate(.config = "rf")

#mlp
mlp_collection = final_mlp_fit %>%
collect_predictions(parameters = best_mlp_params)%>% mutate(.config = "mlp")

#xgb
xgb_collection = final_XGB_fit %>%
collect_predictions(parameters = best_XGB_params)%>% mutate(.config = "xgb")

#lgbm
lgbm_collection = as.data.frame(final_LightGBM_fit %>%
collect_predictions(parameters = best_LightGBM_params[1,])%>% mutate(.config = "lgbm"))



all_sets = rbind(lgbm_collection,rf_collection,xgb_collection,svm_collection,lr_collection,mlp_collecti
```

#auc plot

```
all_sets %>% mutate(model = .config) %>%
  group_by(model) %>% #group by model name
  roc_curve(class,.pred_A:.pred_D) %>%
  ggplot(
    aes(
      x = 1 - specificity,
      y = sensitivity,
      color = model
    )
  ) +
  geom_line(size = 0.9) +
  geom_abline(slope = 1, intercept = 0, size = 0.4)
```



#auc scores

```
all_sets %>%
  group_by(.config) %>%
  roc_auc(class,.pred_A:.pred_D) %>% arrange(desc(.estimate))
```

```
## # A tibble: 8 x 4
##   .config .metric .estimator .estimate
##   <chr>   <chr>   <chr>          <dbl>
## 1 lgbm    roc_auc hand_till      0.922
## 2 xgb     roc_auc hand_till      0.922
## 3 rf      roc_auc hand_till      0.919
## 4 mlp     roc_auc hand_till      0.916
## 5 svm     roc_auc hand_till      0.886
## 6 multi   roc_auc hand_till      0.846
## 7 knn     roc_auc hand_till      0.845
## 8 lda     roc_auc hand_till      0.841
```

#accuracy score (proportion of data that are predicted correctly)

```
all_sets %>%
  group_by(.config) %>%
  accuracy(class,.pred_class) %>% arrange(desc(.estimate))
```

```
## # A tibble: 8 x 4
```

```
##   .config .metric  .estimator .estimate
##   <chr>   <chr>    <chr>          <dbl>
## 1 lgbm    accuracy multiclass     0.757
## 2 xgb     accuracy multiclass     0.752
## 3 rf      accuracy multiclass     0.750
## 4 mlp     accuracy multiclass     0.738
## 5 svm     accuracy multiclass     0.704
## 6 knn     accuracy multiclass     0.613
## 7 multi   accuracy multiclass     0.603
## 8 lda     accuracy multiclass     0.595
```