基于机器学习的风险预测

吴沛豪

2023-05-16

目录

1 环境准备 **2**

2 模型数据准备 **4**

3 建立模型 **4**

3.1 逻辑回归 4
3.2 决策树 5
3.3 SVM 8
3.4 朴素贝叶斯 NaiveBayes 9
3.5 K 近邻 11
3.6 随机森林 13
3.7 XGBoost 14
3.8 深度学习 MLP 多层感知机 17

4 模型比较 **19**

1 环境准备

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Chinese (Simplified)_China.utf8
## [2] LC_CTYPE=Chinese (Simplified)_China.utf8
## [3] LC_MONETARY=Chinese (Simplified)_China.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=Chinese (Simplified)_China.utf8
##
## attached base packages:
## [1] stats4    grid      stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] ggpubr_0.6.0         runway_0.0.0.9000    pROC_1.18.0
##  [4] tensorflow_2.11.0    keras_2.11.1         party_1.3-13
##  [7] strucchange_1.5-3    sandwich_3.0-2       zoo_1.8-11
## [10] modeltools_0.2-23    mvtnorm_1.1-3        RWeka_0.4-46
## [13] rattle_5.5.1         bitops_1.0-7         xgboost_1.7.5.1
## [16] glmnet_4.1-7         Matrix_1.5-4         e1071_1.7-13
## [19] MASS_7.3-58.3        caret_6.0-94         lattice_0.20-45
## [22] randomForest_4.7-1.1 rpart.plot_3.1.1     rpart_4.1.19
## [25] mice_3.15.0          yardstick_1.1.0      workflowsets_1.0.1
## [28] workflows_1.1.3      tune_1.1.0           rsample_1.1.1
## [31] recipes_1.0.5        parsnip_1.0.4        modeldata_1.1.0
## [34] infer_1.0.4          dials_1.2.0          scales_1.2.1
## [37] broom_1.0.4          tidymodels_1.0.0     VIM_6.2.2
## [40] colorspace_2.1-0     patchwork_1.1.2      qgraph_1.9.4
## [43] reshape2_1.4.4       lubridate_1.9.2      forcats_1.0.0
## [46] stringr_1.5.0        dplyr_1.1.1          purrr_1.0.1
## [49] readr_2.1.4          tidyr_1.3.0          tibble_3.2.1
## [52] ggplot2_3.4.2        tidyverse_2.0.0      igraph_1.4.1
##
```

```
## loaded via a namespace (and not attached):
##   [1] backports_1.4.1      Hmisc_5.0-1          plyr_1.8.8
##   [4] sp_1.6-0             splines_4.2.2        listenv_0.9.0
##   [7] tfruns_1.5.1         TH.data_1.1-1        digest_0.6.31
##  [10] foreach_1.5.2        htmltools_0.5.4      fansi_1.0.4
##  [13] magrittr_2.0.3       checkmate_2.1.0      cluster_2.1.4
##  [16] tzdb_0.3.0           globals_0.16.2       gower_1.0.1
##  [19] matrixStats_0.63.0   hardhat_1.3.0        timechange_0.2.0
##  [22] jpeg_0.1-10          xfun_0.37            libcoin_1.0-9
##  [25] jsonlite_1.8.4       zeallot_0.1.0        survival_3.5-5
##  [28] iterators_1.0.14     glue_1.6.2           gtable_0.3.3
##  [31] ipred_0.9-14         car_3.1-2            shape_1.4.6
##  [34] future.apply_1.10.0  DEoptimR_1.0-12      abind_1.4-5
##  [37] rstatix_0.7.2        Rcpp_1.0.10          laeken_0.5.2
##  [40] htmlTable_2.4.1      reticulate_1.28      GPfit_1.0-8
##  [43] foreign_0.8-84       proxy_0.4-27         Formula_1.2-5
##  [46] lava_1.7.2.1         prodlim_2023.03.31   vcd_1.4-11
##  [49] htmlwidgets_1.6.2    lavaan_0.6-15        rJava_1.0-6
##  [52] pkgconfig_2.0.3      nnet_7.3-18          utf8_1.2.3
##  [55] tidyselect_1.2.0     rlang_1.1.0          DiceDesign_1.9
##  [58] munsell_0.5.0        tools_4.2.2          cli_3.6.0
##  [61] generics_0.1.3       ranger_0.15.1        fdrtool_1.2.17
##  [64] evaluate_0.20        fastmap_1.1.1        yaml_2.3.7
##  [67] rticles_0.24         ModelMetrics_1.2.2.2 knitr_1.42
##  [70] robustbase_0.95-1    coin_1.4-2           glasso_1.11
##  [73] pbapply_1.7-0        future_1.32.0        nlme_3.1-162
##  [76] whisker_0.4.1        compiler_4.2.2       rstudioapi_0.14
##  [79] png_0.1-8            ggsignif_0.6.4       lhs_1.1.6
##  [82] pbivnorm_0.6.0       stringi_1.7.12       psych_2.3.3
##  [85] RWekajars_3.9.3-2    vctrs_0.6.1          pillar_1.9.0
##  [88] lifecycle_1.0.3      furrr_0.3.1          lmtest_0.9-40
##  [91] data.table_1.14.8    corpcor_1.6.10       R6_2.5.1
##  [94] gridExtra_2.3        parallelly_1.35.0    codetools_0.2-19
##  [97] boot_1.3-28.1        gtools_3.9.4         withr_2.5.0
## [100] mnormt_2.1.1         multcomp_1.4-23      parallel_4.2.2
## [103] hms_1.1.3            quadprog_1.5-8       timeDate_4022.108
## [106] class_7.3-21         rmarkdown_2.21       carData_3.0-5
## [109] base64enc_0.1-3
```

```
if (length(tf$config$list_physical_devices("GPU"))) {
  message("TensorFlow **IS** using the GPU")
} else {
```

```
  message("TensorFlow **IS NOT** using the GPU")
}
```

2 模型数据准备

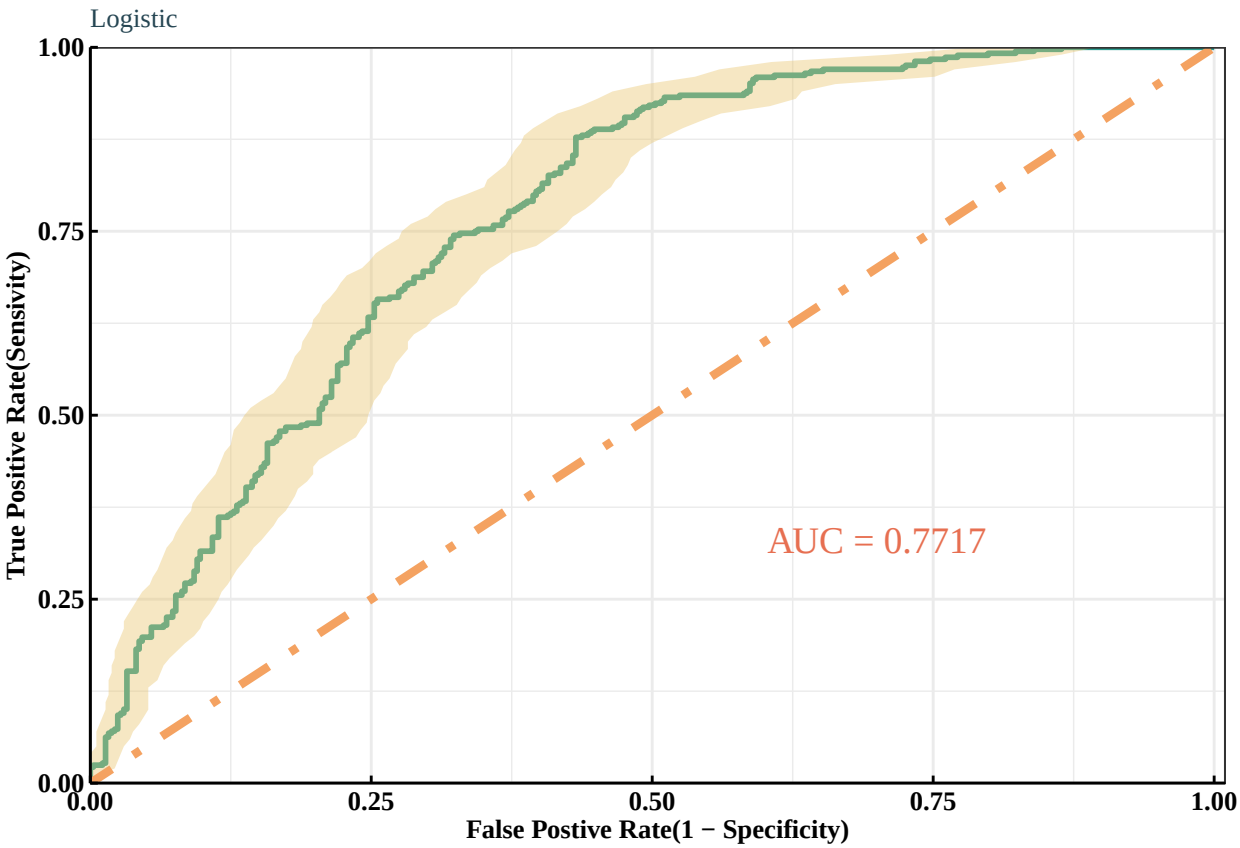
3 建立模型

3.1 逻辑回归

```
source("model_code_old/logistic_old.R")
source("extra.R")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 276 126
##          0  92 242
##
##                Accuracy : 0.7038
##                  95% CI : (0.6694, 0.7366)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.4076
##
##  Mcnemar's Test P-Value : 0.02541
##
##             Sensitivity : 0.7500
##             Specificity : 0.6576
##          Pos Pred Value : 0.6866
##          Neg Pred Value : 0.7246
##               Precision : 0.6866
##                  Recall : 0.7500
##                      F1 : 0.7169
##              Prevalence : 0.5000
##          Detection Rate : 0.3750
##    Detection Prevalence : 0.5462
##       Balanced Accuracy : 0.7038
```

```
##
##         'Positive' Class : 1
##
```



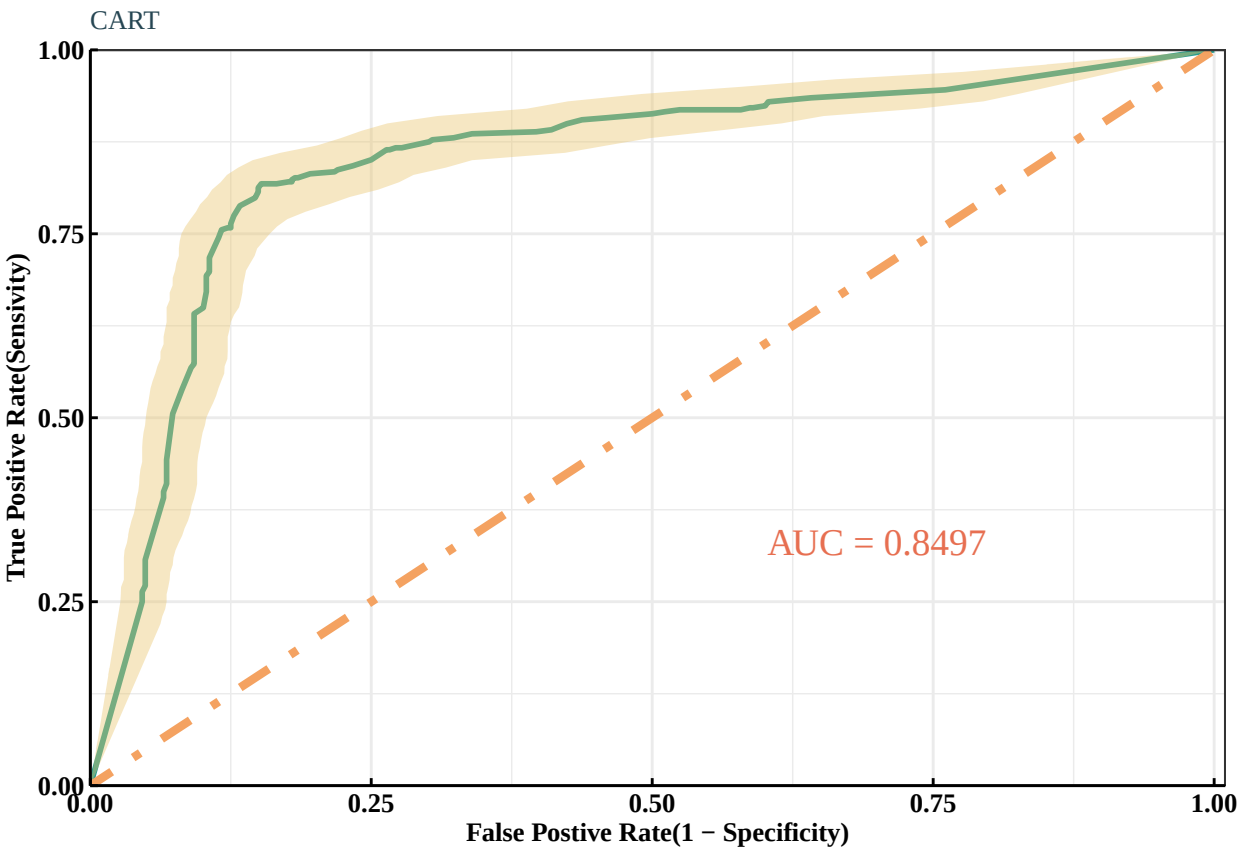
3.2 决策树

3.2.1 CART 算法

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 301  56
##          0  67 312
##
##                Accuracy : 0.8329
##                  95% CI : (0.8039, 0.8591)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6658
```

```
##
##  Mcnemar's Test P-Value : 0.3672
##
##             Sensitivity : 0.8179
##             Specificity : 0.8478
##          Pos Pred Value : 0.8431
##          Neg Pred Value : 0.8232
##               Precision : 0.8431
##                  Recall : 0.8179
##                      F1 : 0.8303
##              Prevalence : 0.5000
##          Detection Rate : 0.4090
##    Detection Prevalence : 0.4851
##       Balanced Accuracy : 0.8329
##
##        'Positive' Class : 1
##
```

```
p2 <- ROC.p(outcomes0$pred0_treeCART,title = 'CART')
p2
```



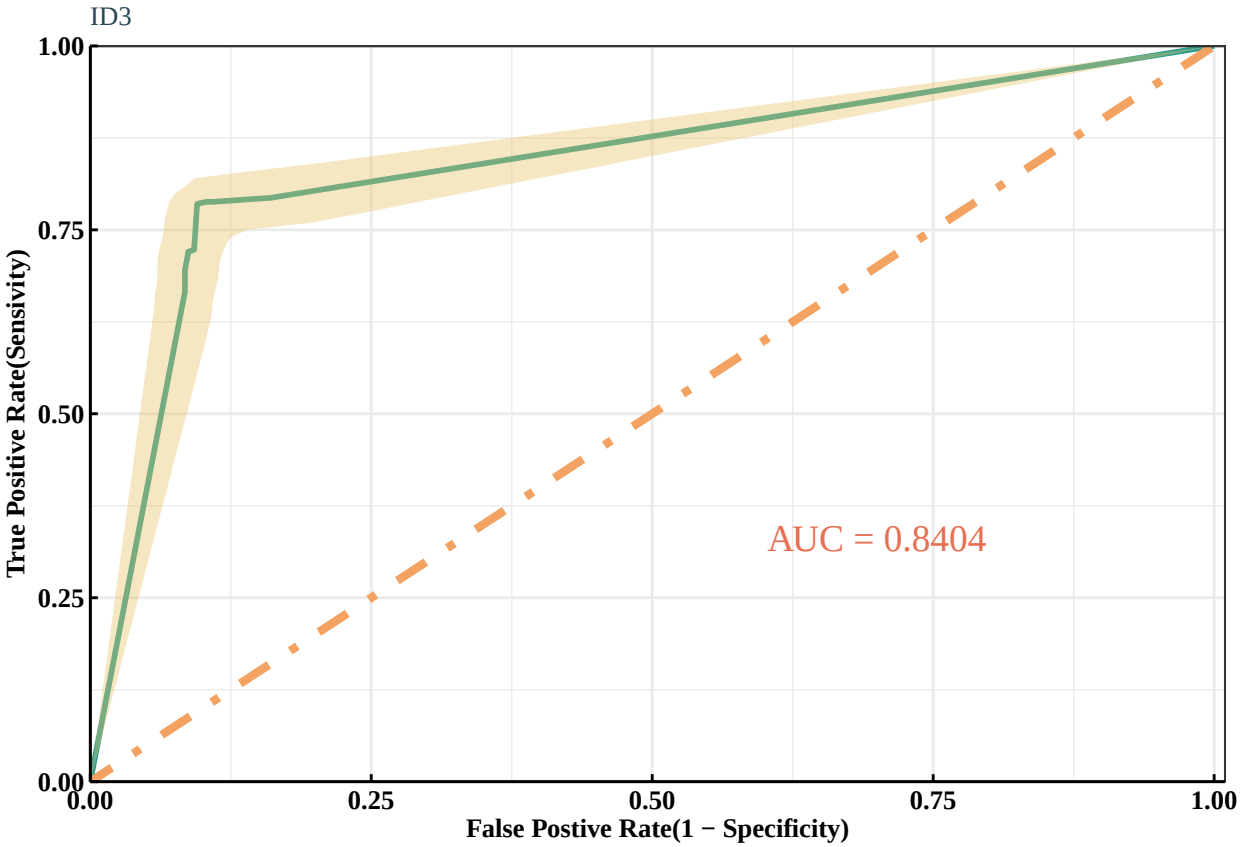
3.2.2 ID3 算法

```
source("model_code_old/id3_old.R")
```

```
CMX(outcomes$pred_treeID3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 289  35
##          0  79 333
##
##                Accuracy : 0.8451
##                  95% CI : (0.8169, 0.8705)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6902
##
##  Mcnemar's Test P-Value : 5.642e-05
##
##             Sensitivity : 0.7853
##             Specificity : 0.9049
##          Pos Pred Value : 0.8920
##          Neg Pred Value : 0.8083
##               Precision : 0.8920
##                  Recall : 0.7853
##                      F1 : 0.8353
##              Prevalence : 0.5000
##          Detection Rate : 0.3927
##    Detection Prevalence : 0.4402
##       Balanced Accuracy : 0.8451
##
##        'Positive' Class : 1
##
```

```
p3 <- ROC.p(outcomes0$pred0_treeID3,title = 'ID3')
p3
```

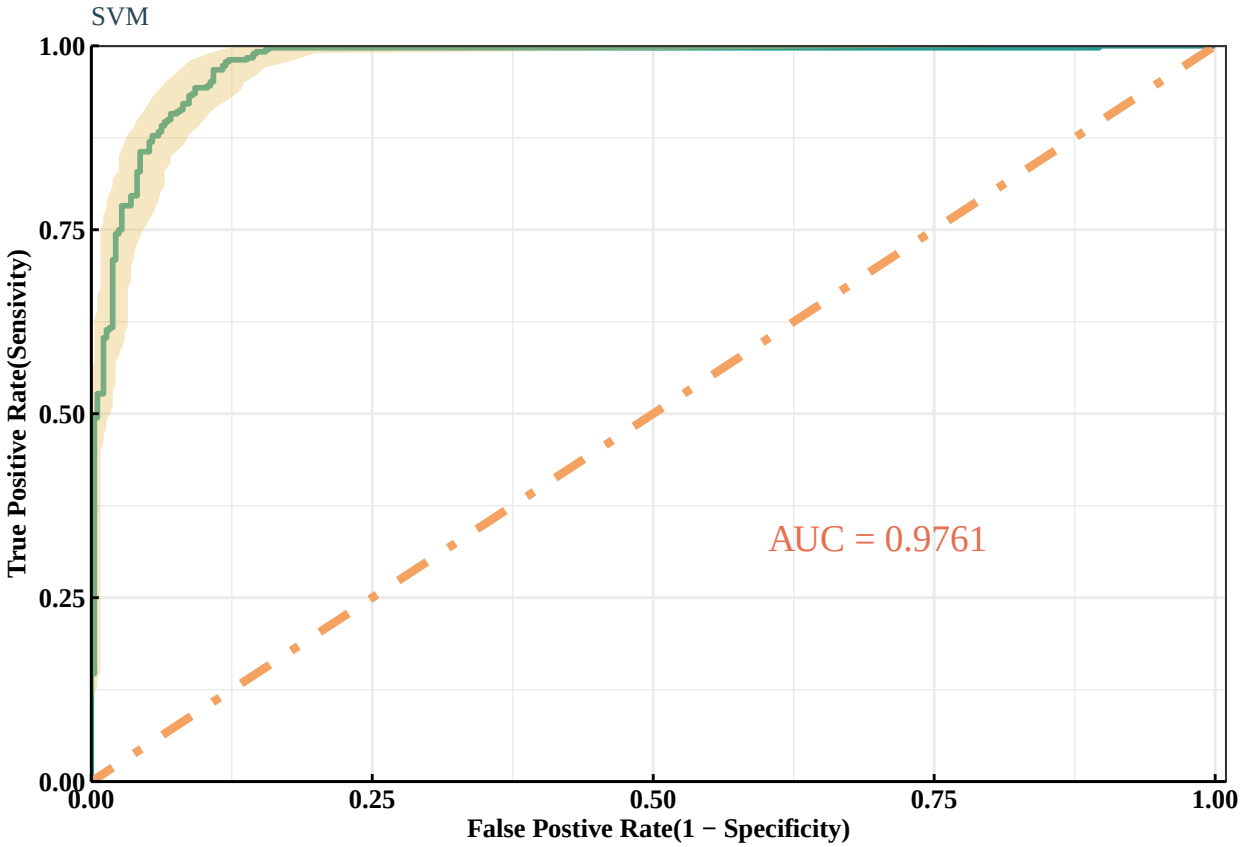


3.3 SVM

```
CMX(outcomes$pred_SVM)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 334  28
##          0  34 340
##
##                Accuracy : 0.9158
##                  95% CI : (0.8933, 0.9348)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8315
##
##  Mcnemar's Test P-Value : 0.5254
##
```

```
##             Sensitivity : 0.9076
##             Specificity : 0.9239
##          Pos Pred Value : 0.9227
##          Neg Pred Value : 0.9091
##               Precision : 0.9227
##                  Recall : 0.9076
##                      F1 : 0.9151
##              Prevalence : 0.5000
##          Detection Rate : 0.4538
##    Detection Prevalence : 0.4918
##       Balanced Accuracy : 0.9158
##
##        'Positive' Class : 1
##
```



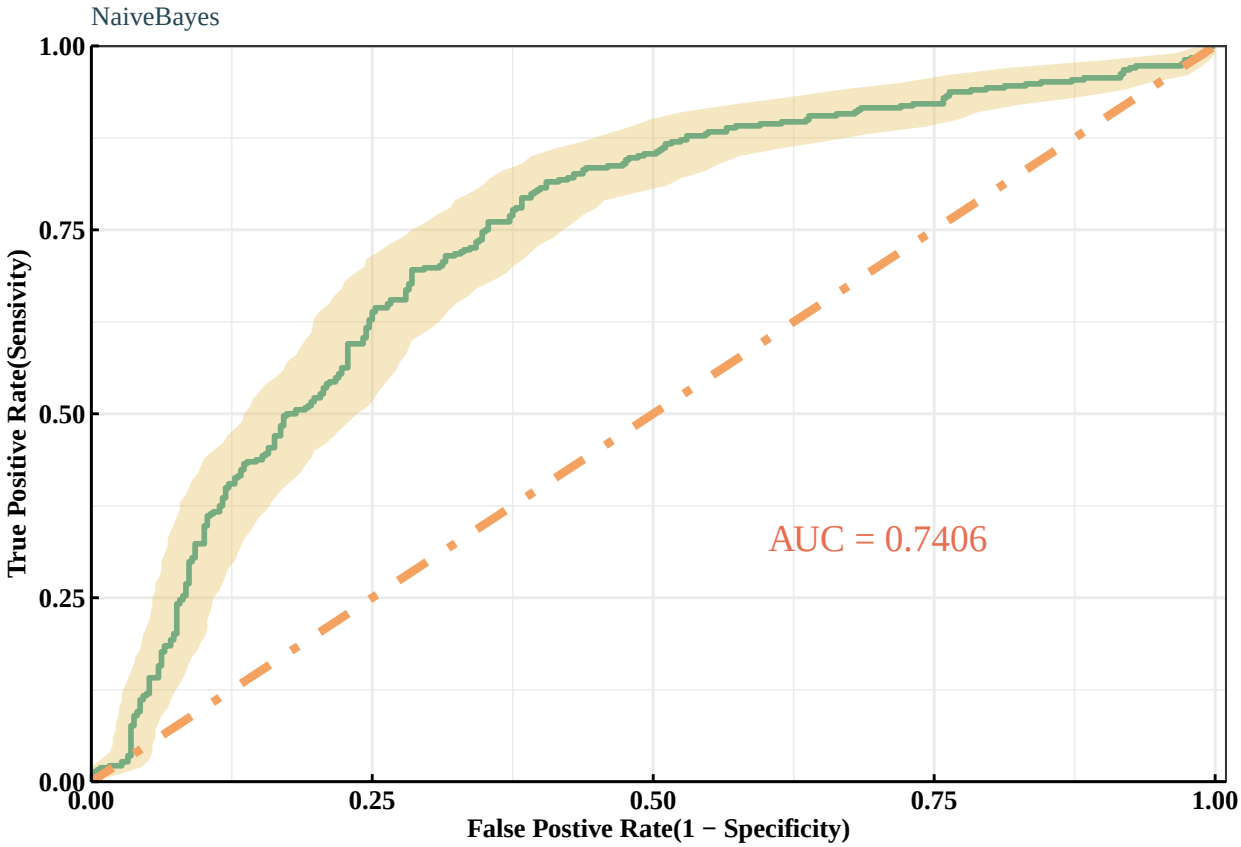
3.4 朴素贝叶斯 NaiveBayes

```
CMX(outcomes$pred_NB)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction   1   0
##          1 275 128
##          0  93 240
##
##                Accuracy : 0.6997
##                  95% CI : (0.6652, 0.7327)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.3995
##
##  Mcnemar's Test P-Value : 0.02219
##
##             Sensitivity : 0.7473
##             Specificity : 0.6522
##          Pos Pred Value : 0.6824
##          Neg Pred Value : 0.7207
##               Precision : 0.6824
##                  Recall : 0.7473
##                      F1 : 0.7134
##              Prevalence : 0.5000
##          Detection Rate : 0.3736
##    Detection Prevalence : 0.5476
##       Balanced Accuracy : 0.6997
##
##        'Positive' Class : 1
##
```

```
p5 <- ROC.p(outcomes0$pred0_NB,title = 'NaiveBayes')
p5
```



3.5 K 近邻

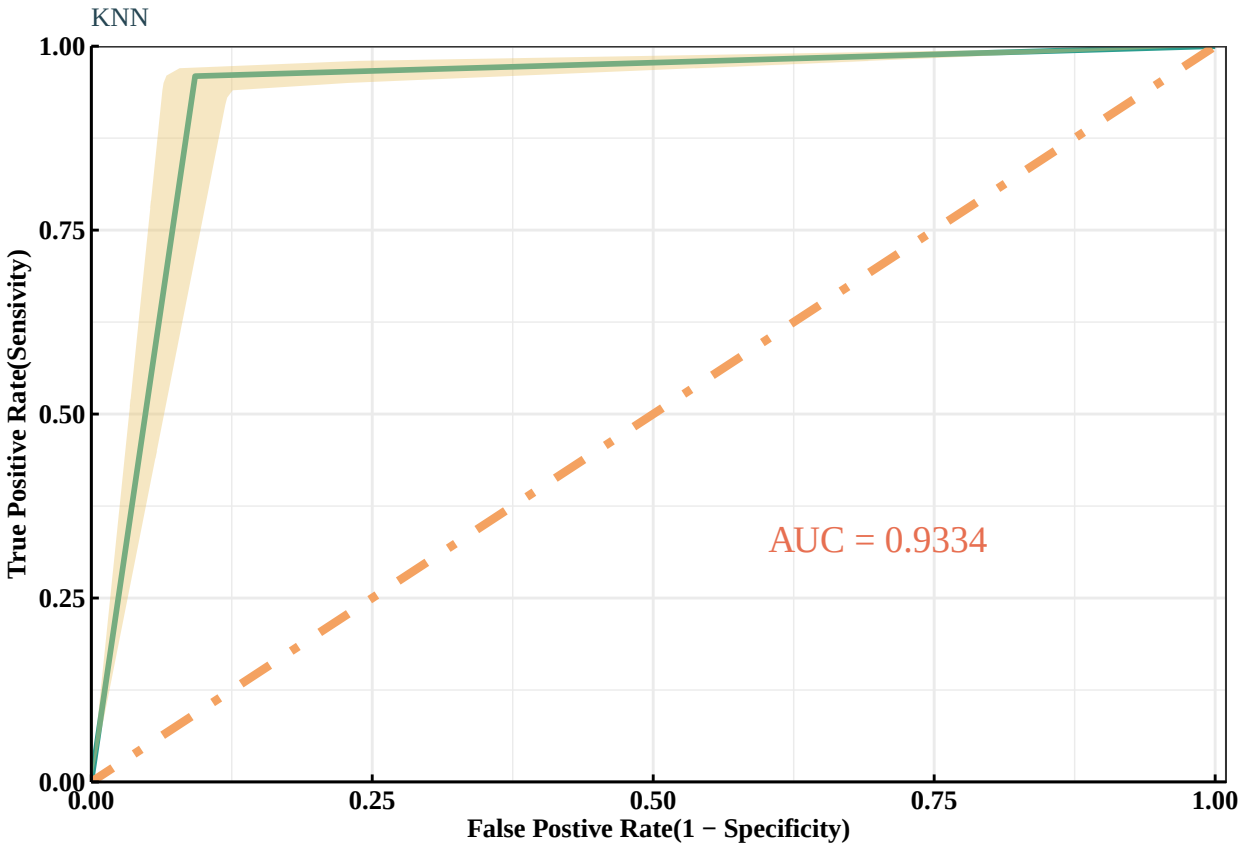
```
source("model_code_old/K_old.R")
```

```
CMX(outcomes$pred_KNN)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 353  34
##          0  15 334
##
##                Accuracy : 0.9334
##                  95% CI : (0.9129, 0.9503)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8668
##
```

```
##  Mcnemar's Test P-Value : 0.01013
##
##             Sensitivity : 0.9592
##             Specificity : 0.9076
##          Pos Pred Value : 0.9121
##          Neg Pred Value : 0.9570
##               Precision : 0.9121
##                  Recall : 0.9592
##                      F1 : 0.9351
##              Prevalence : 0.5000
##          Detection Rate : 0.4796
##    Detection Prevalence : 0.5258
##       Balanced Accuracy : 0.9334
##
##        'Positive' Class : 1
##
```

```
p6 <- ROC.p(outcomes0$pred0_KNN,title = 'KNN')
p6
```



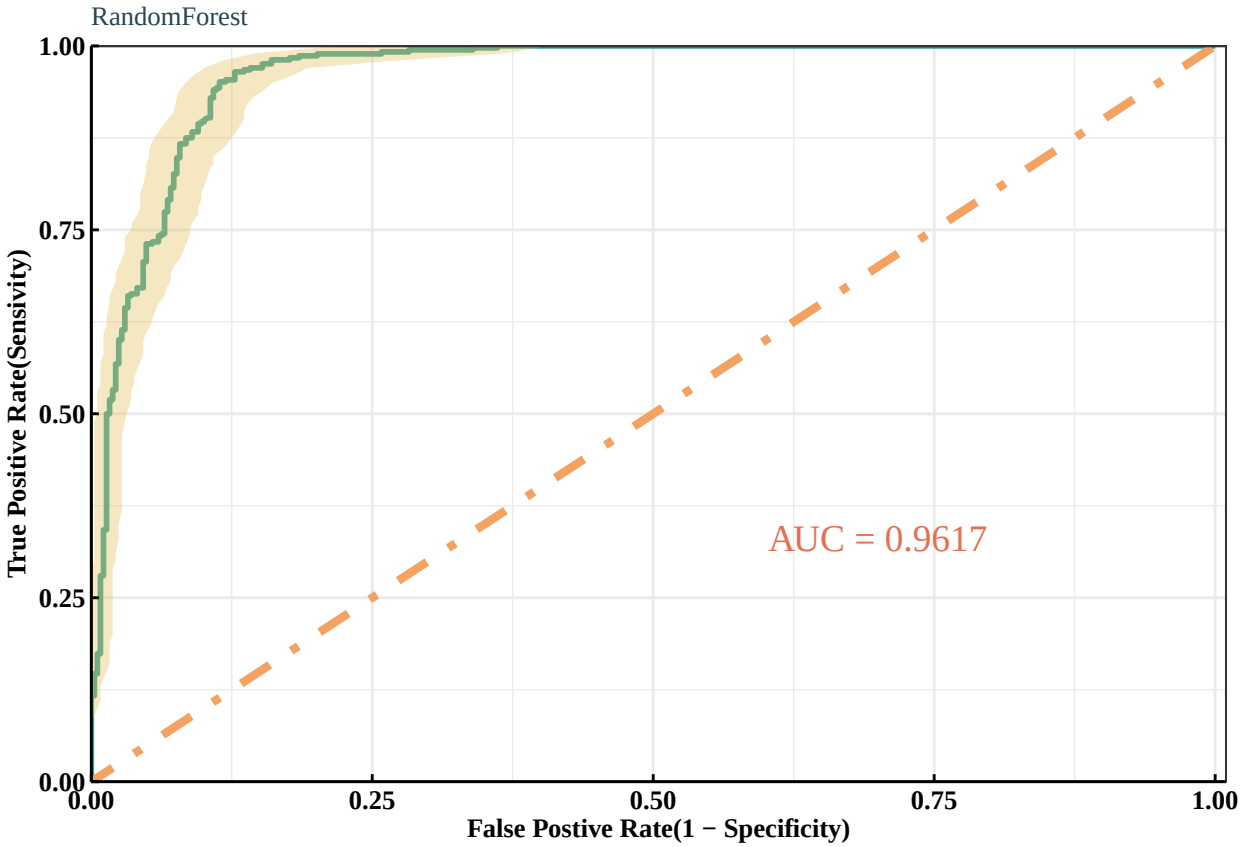
3.6 随机森林

```
source("model_code_old/rf_old.R")
```

```
CMX(outcomes$pred_RF)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 330  37
##          0  38 331
##
##                Accuracy : 0.8981
##                  95% CI : (0.8739, 0.919)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7962
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8967
##             Specificity : 0.8995
##          Pos Pred Value : 0.8992
##          Neg Pred Value : 0.8970
##               Precision : 0.8992
##                  Recall : 0.8967
##                      F1 : 0.8980
##              Prevalence : 0.5000
##          Detection Rate : 0.4484
##    Detection Prevalence : 0.4986
##       Balanced Accuracy : 0.8981
##
##        'Positive' Class : 1
##
```

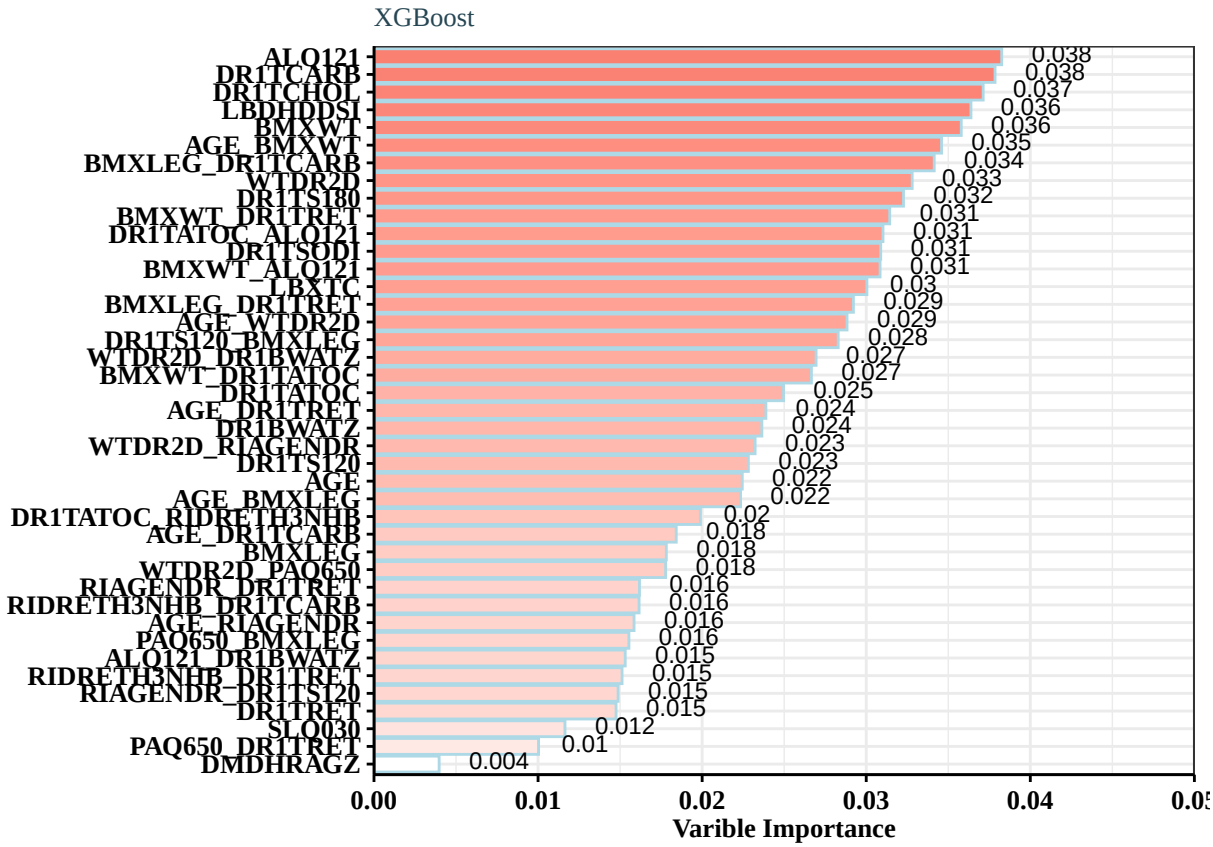
```
p7 <- ROC.p(outcomes0$pred0_RF,title = 'RandomForest')
p7
```



3.7 XGBoost

```
XGB_fe <- cbind.data.frame(Feature = importance$Feature,
                           coef = importance$Cover)
pXGB_fe <- ggplot(data = XGB_fe, aes(y = coef,
                                     x = reorder(Feature, coef))) +
  geom_col(aes(fill = coef), col = "lightblue") +
  scale_fill_gradient(low = "white", high = "#FA8072") +
  theme(axis.text.x = element_text(angle = 45,
                                   vjust = 1,
                                   size = 12,
                                   hjust = 1)) +
  coord_flip() +
  labs(x = "", y = "Varible Importance") +
  ggtitle("XGBoost") +
  scale_y_continuous(expand = c(0, 0),
                     limits = c(0, 0.05)) +
  geom_text(aes(label = round(coef, 3)),
            vjust = 0.3, hjust = if_else(XGB_fe$coef >0,-0.5,1.2),
```

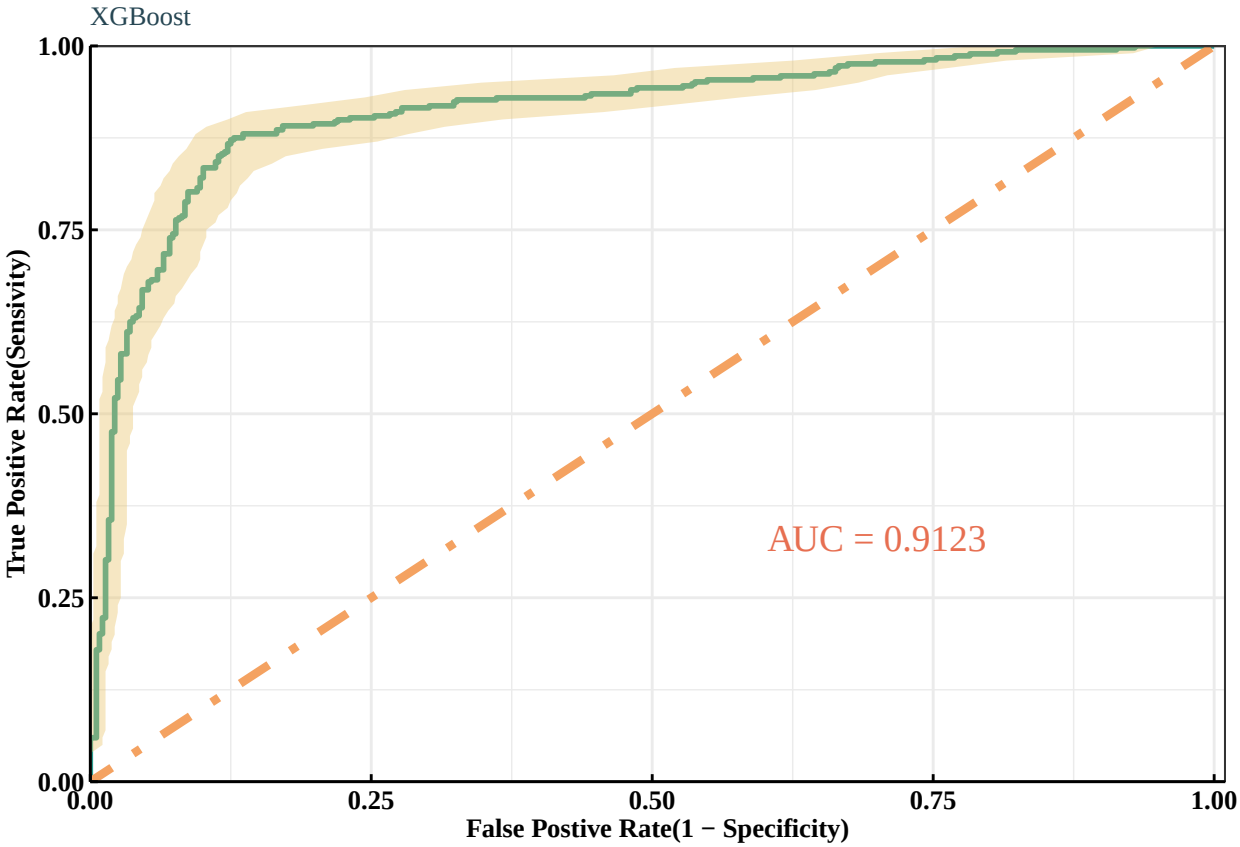
```
            size = 3) +
  theme_bw()+
  theme(panel.background = element_rect(fill = "transparent"),# 设置背景透明
          axis.ticks = element_line(color = "black"),# 设置刻度线颜色
          axis.line = element_line(size = 0.5,
                                   colour = "black"),# 设置边框线颜色
          axis.title = element_text(colour = "black",
                                    size = 10,
                                    face = "bold"),# 设置标题字体
          axis.text = element_text(colour = "black",
                                   size = 10,
                                   face = "bold"),# 设置 x,y 轴标签字体
          axis.text.x = element_text(angle = 0,hjust = 0.5,vjust = 0.5),
          text = element_text(size = 8,
                              color = "#264653",
                              family = "serif"))+# 设置文本字体
  guides(fill=FALSE)
pXGB_fe
```



```
CMX(outcomes$pred_XGB)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 295  35
##          0  73 333
##
##                Accuracy : 0.8533
##                  95% CI : (0.8256, 0.878)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7065
##
##  Mcnemar's Test P-Value : 0.0003704
##
##             Sensitivity : 0.8016
##             Specificity : 0.9049
##          Pos Pred Value : 0.8939
##          Neg Pred Value : 0.8202
##               Precision : 0.8939
##                  Recall : 0.8016
##                      F1 : 0.8453
##              Prevalence : 0.5000
##          Detection Rate : 0.4008
##    Detection Prevalence : 0.4484
##       Balanced Accuracy : 0.8533
##
##        'Positive' Class : 1
##
```

```
p8 <- ROC.p(outcomes0$pred0_XGB,title = 'XGBoost')
p8
```



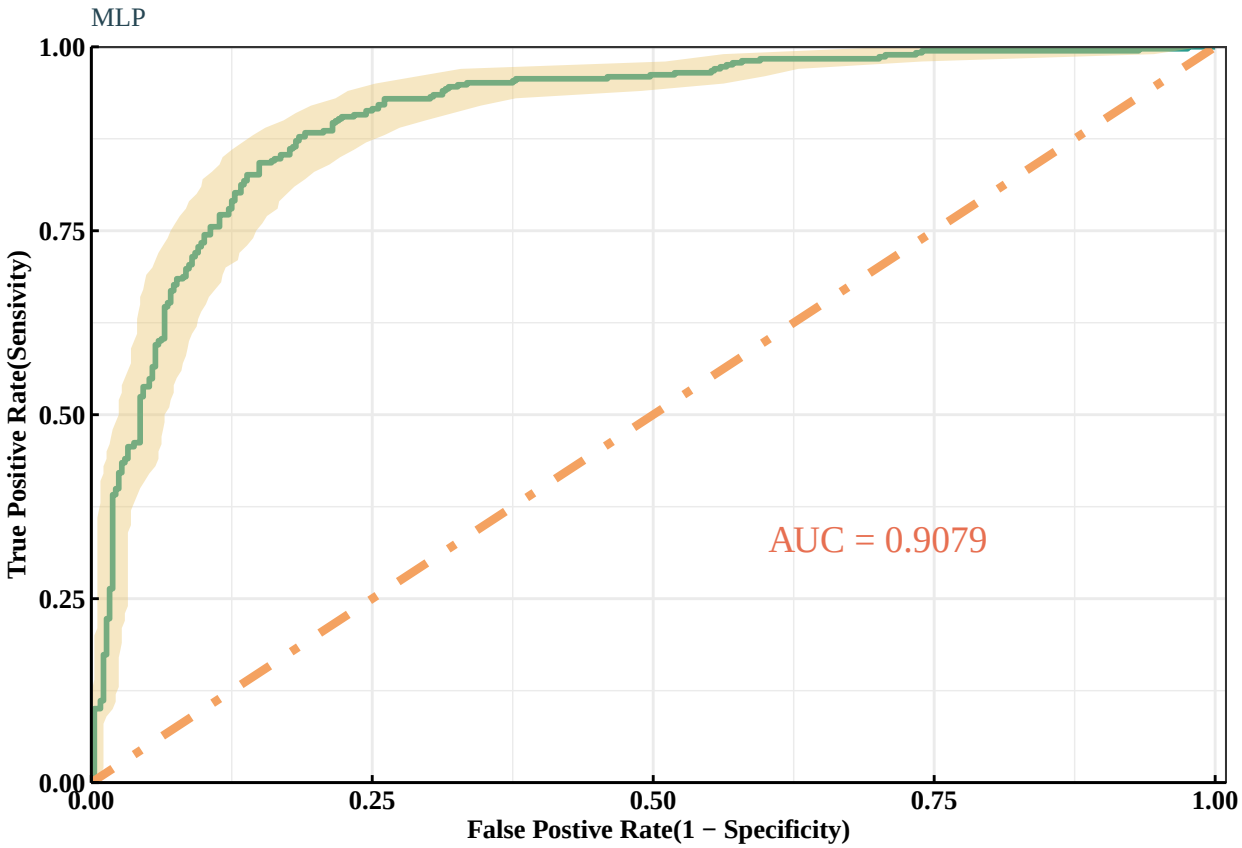
3.8 深度学习 MLP 多层感知机

```
CMX(outcomes$pred_MLP)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   0
##          1 290  46
##          0  78 322
##
##                Accuracy : 0.8315
##                  95% CI : (0.8025, 0.8579)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.663
##
##  Mcnemar's Test P-Value : 0.005371
##
```

```
##             Sensitivity : 0.7880
##             Specificity : 0.8750
##          Pos Pred Value : 0.8631
##          Neg Pred Value : 0.8050
##               Precision : 0.8631
##                  Recall : 0.7880
##                      F1 : 0.8239
##              Prevalence : 0.5000
##          Detection Rate : 0.3940
##    Detection Prevalence : 0.4565
##       Balanced Accuracy : 0.8315
##
##        'Positive' Class : 1
##
```

```
p9 <- ROC.p(outcomes0$pred0_MLP,title = 'MLP')
p9
```




```
MLP %>% summary()
```

```
## Model: "sequential_1500"
## _________________________________________________________________________
```

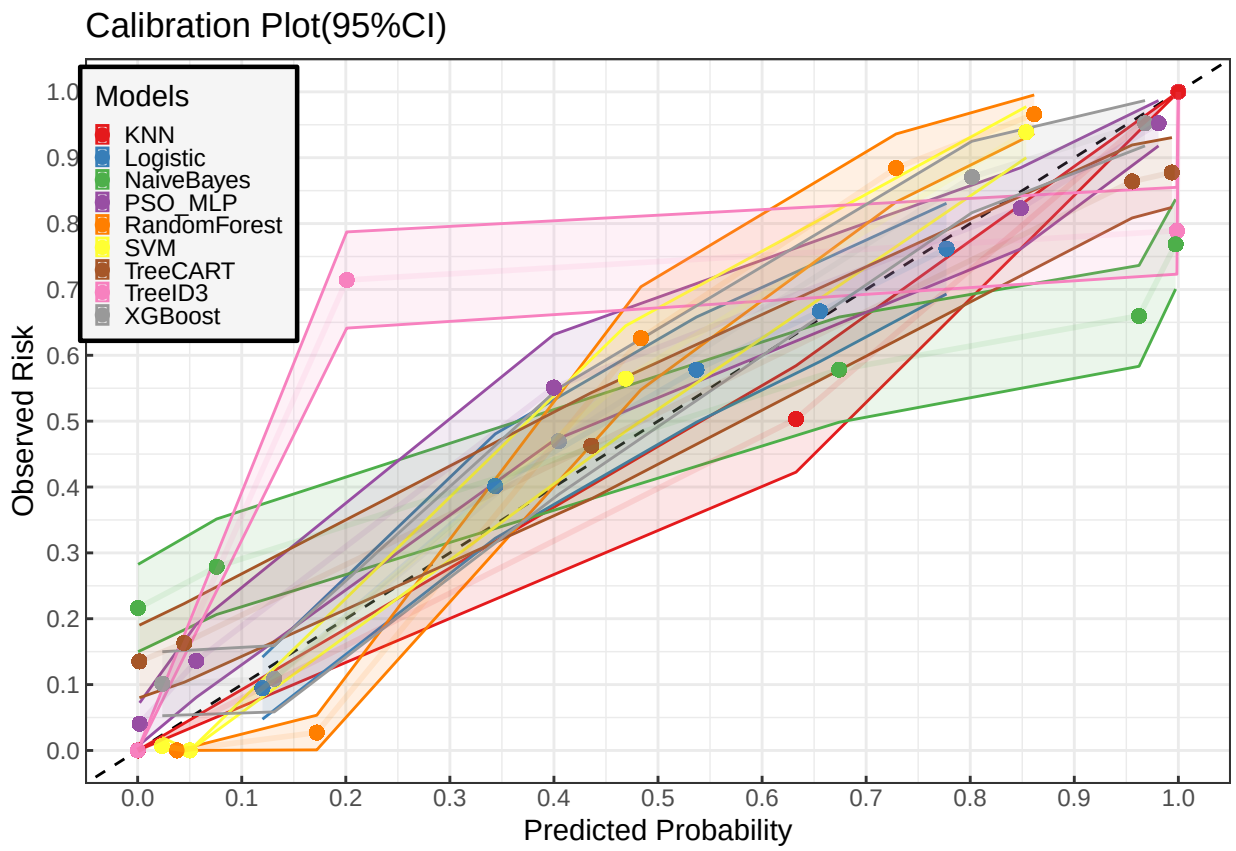
```
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  dense_7505 (Dense)                 (None, 128)                     5760
##  dropout_4502 (Dropout)             (None, 128)                     0
##  dense_7504 (Dense)                 (None, 64)                      8256
##  dropout_4501 (Dropout)             (None, 64)                      0
##  dense_7503 (Dense)                 (None, 32)                      2080
##  dropout_4500 (Dropout)             (None, 32)                      0
##  dense_7502 (Dense)                 (None, 16)                      528
##  dense_7501 (Dense)                 (None, 1)                       17
## ================================================================================
## Total params: 16,641
## Trainable params: 16,641
## Non-trainable params: 0
## --------------------------------------------------------------------------------
```

4 模型比较

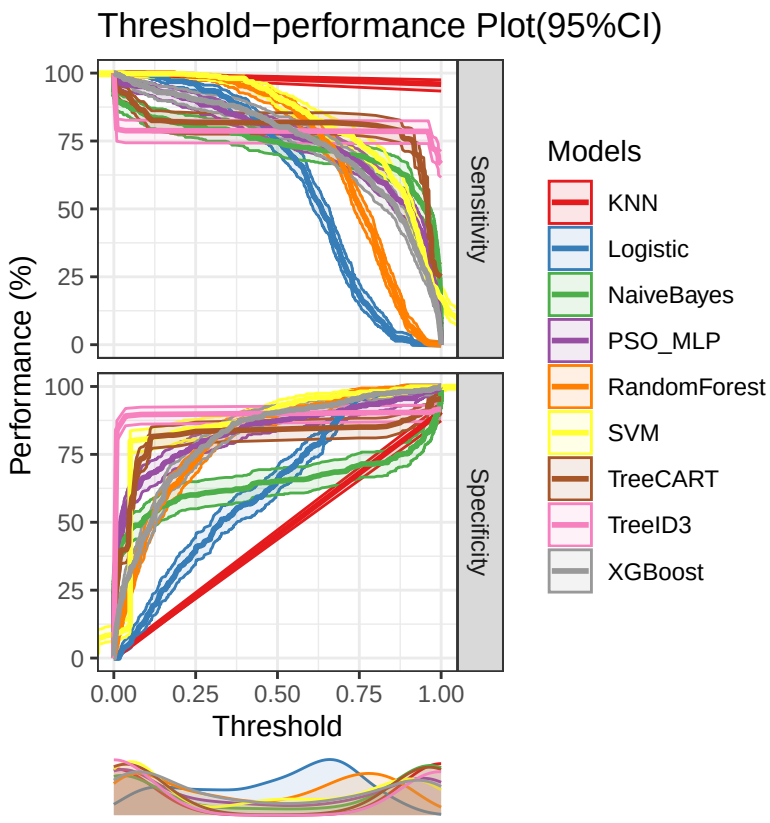
```
colnames(outcomes0) <- c(
  "TrueValue", "Logistic",
  "TreeCART", "TreeID3",
  "SVM", "NaiveBayes",
  "KNN", "RandomForest",
  "XGBoost", "PSO_MLP"
)
colnames(outcomes) <- c(
  "TrueValue", "Logistic",
  "TreeCART", "TreeID3",
  "SVM", "NaiveBayes",
  "KNN", "RandomForest",
  "XGBoost", "PSO_MLP"
)
colnames(outcomesDS) <- c(
  "TrueValue", "Logistic",
  "TreeCART", "TreeID3",
  "SVM", "NaiveBayes",
  "KNN", "RandomForest",
  "XGBoost", "PSO_MLP"
)
colnames(outcomesDS0) <- c(
```

```
  "TrueValue", "Logistic",
  "TreeCART", "TreeID3",
  "SVM", "NaiveBayes",
  "KNN", "RandomForest",
  "XGBoost", "PSO_MLP"
)
```

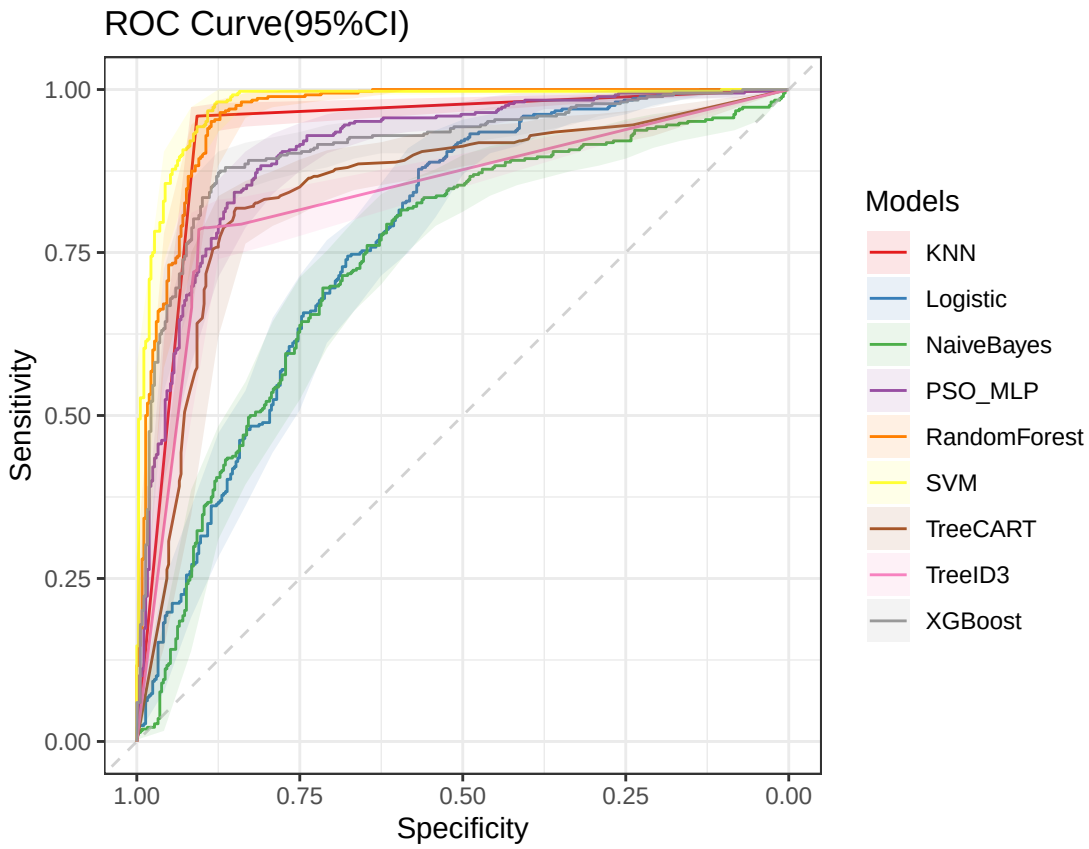
```
cp <- cal_plot_multi(outcomes2,
  outcome = "TrueValue",
  prediction = "pred",
  model = "model",
  n_bins = 5,
  plot_title = "Calibration Plot(95%CI)"
)
cp <- cp$patches$plots[[1]]
cp +
  theme_bw()+
  theme(legend.position=c(0.09,0.8),
        legend.background = element_rect(fill = '#F5F5F5',
                                         colour = 'black',
                                         size = 0.8),
        legend.key.size = unit(2,'mm'),
        legend.key.height = unit(2,'mm'),
        legend.key.width = unit(2,'mm'),
        legend.key = element_rect(colour = '#F5F5F5',
                                  fill = '#F5F5F5'))+
  theme(axis.ticks.length.y = unit(-0.1, 'cm'),
        axis.ticks.length.x = unit(-0.1, 'cm'))
```



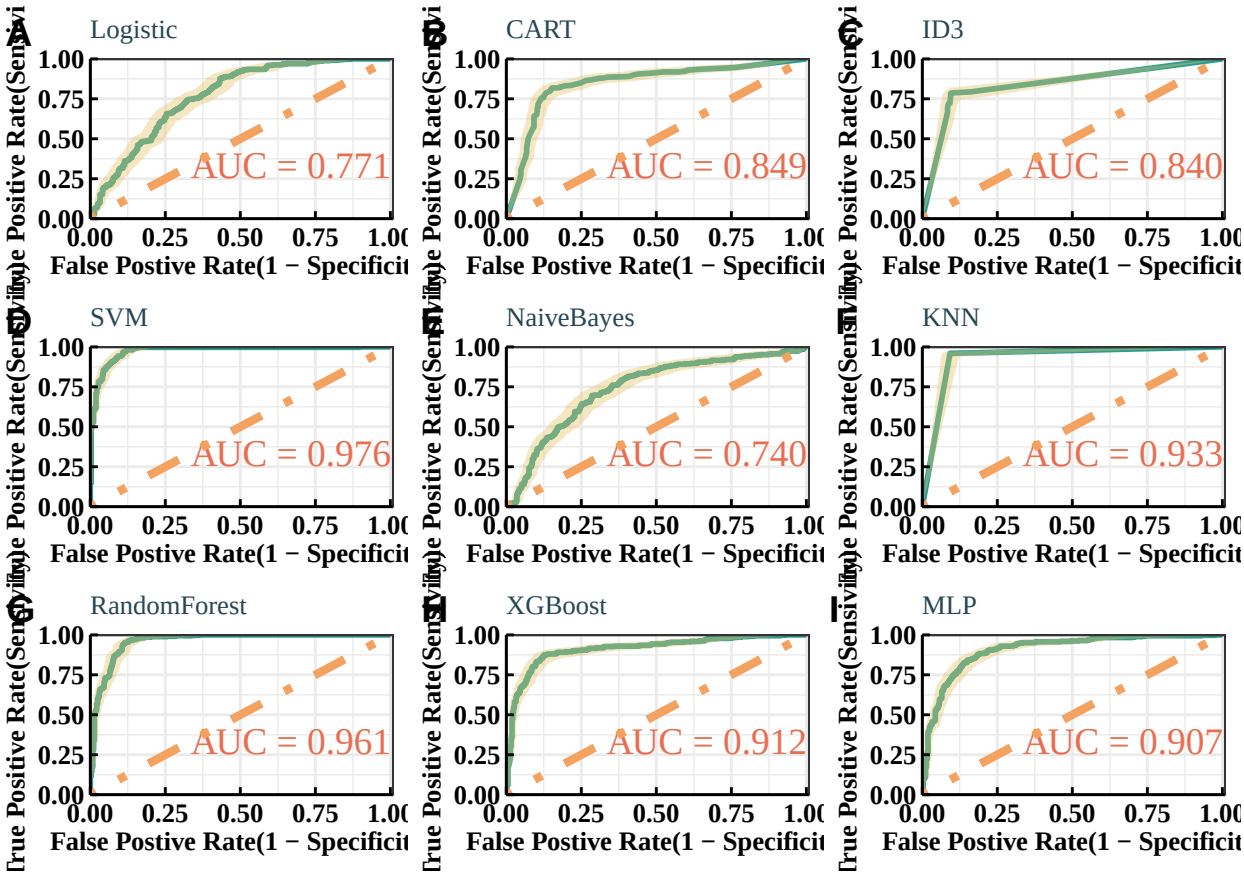

```
threshperf_plot_multi(outcomes2,
  outcome = "TrueValue",
  prediction = "pred",
  model = "model",
  plot_title = "Threshold-performance Plot(95%CI)"
)+theme_bw()
```




```
roc_plot_multi(outcomes2,
  outcome = "TrueValue",
  prediction = "pred",
  model = "model",
  ci = T,
  plot_title = "ROC Curve(95%CI)"
)+theme_bw()
```




```
ggarrange(p1, p2, p3, p4,p5,p6,p7,p8,p9, ncol = 3, nrow = 3,
          labels = c("A","B","C","D","E",'F','G','H','I'), # 添加标签
          font.label = list(size = 14, face = "bold")) # 设置标签字体样式
```




```
CMXALL <- CMX.all(outcomes)
knitr::kable(CMXALL,
  caption = " 各模型在测试集上的效果",
  align = "c"
)
```

表 1: 各模型在测试集上的效果

	准确率	Kappa 值	F1 值	灵敏度/召回率	特异度	阳性预测值/精确率	阴性预测值
Logistic	0.704(0.669,0.737)	0.408	0.717	0.750	0.658	0.687	0.725
TreeCART	0.833(0.804,0.859)	0.666	0.830	0.818	0.848	0.843	0.823
TreeID3	0.845(0.817,0.871)	0.690	0.835	0.785	0.905	0.892	0.808
SVM	0.916(0.893,0.935)	0.832	0.915	0.908	0.924	0.923	0.909
NaiveBayes	0.7(0.665,0.733)	0.399	0.713	0.747	0.652	0.682	0.721
KNN	0.933(0.913,0.95)	0.867	0.935	0.959	0.908	0.912	0.957
RandomForest	0.898(0.874,0.919)	0.796	0.898	0.897	0.899	0.899	0.897
XGBoost	0.853(0.826,0.878)	0.707	0.845	0.802	0.905	0.894	0.820
PSO_MLP	0.832(0.802,0.858)	0.663	0.824	0.788	0.875	0.863	0.805

```
CMXALL2 <- cbind.data.frame(MODEL=rownames(CMXALL),
                            CMXALL)
write_excel_csv(CMXALL2,'CMXALL_old.csv')
```

```
CMXDS <- CMX.all(outcomesDS)
knitr::kable(CMXDS,
  caption = " 各模型在负采样数据集上的效果",
  align = "c"
)
```

表 2: 各模型在负采样数据集上的效果

	准确率	Kappa 值	F1 值	灵敏度/召回率	特异度	阳性预测值/精确率	阴性预测值
Logistic	0.68(0.637,0.721)	0.356	0.711	0.761	0.593	0.668	0.698
TreeCART	0.888(0.857,0.914)	0.775	0.894	0.907	0.867	0.880	0.897
TreeID3	0.944(0.92,0.962)	0.888	0.947	0.973	0.913	0.923	0.969
SVM	0.948(0.925,0.966)	0.895	0.952	0.988	0.904	0.918	0.986
NaiveBayes	0.672(0.629,0.713)	0.341	0.699	0.734	0.606	0.667	0.679
KNN	0.944(0.92,0.962)	0.887	0.949	0.996	0.888	0.905	0.995
RandomForest	0.946(0.922,0.964)	0.892	0.950	0.988	0.900	0.914	0.986
XGBoost	0.908(0.879,0.932)	0.816	0.913	0.927	0.888	0.899	0.918
PSO_MLP	0.892(0.861,0.918)	0.784	0.897	0.903	0.880	0.890	0.895

```
CMXDS2 <- cbind.data.frame(MODEL=rownames(CMXDS),
                           CMXDS)
write_excel_csv(CMXDS2,'CMXDS_old.csv')
```