Predictive analysis of client subscribed for term deposit

Fox325

6/11/2021

Basic data exploration.

Data summary

Name	df
Number of rows	4521
Number of columns	17

Variable type: character

skim_variable	n_missing	n_unique
job	0	12
marital	0	3
education	0	4
default	0	2
housing	0	2
loan	0	2
contact	0	3
month	0	12
poutcome	0	4
y	0	2

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	min	median	max
age	0	1	41.17	10.58	19	39	87
balance	0	1	1422.66	3009.64	-3313	444	71188
day	0	1	15.92	8.25	1	16	31
duration	0	1	263.96	259.86	4	185	3025
campaign	0	1	2.79	3.11	1	2	50
pdays	0	1	39.77	100.12	-1	-1	871
previous	0	1	0.54	1.69	0	0	25

This dataset consists of 17 variables with 4521 observations. Among this 17 variables; 7 are numeric and 10 are character variables. The source of the data set I Kaggle.

https://www.kaggle.com/janiobachmann/bank-marketing-campaign-opening-a-term-deposit/data

Short discription about the variables

- age (numeric)
- job : type of job (categorical)
- marital : marital status (categorical)
- education (categorical)
- default: has credit in default? (binary)
- balance: average yearly balance, in euros (numeric)
- housing: has housing loan? (binary)
- loan: has personal loan? (binary)
- contact: contact communication type (categorical)
- day: last contact day of the month (numeric)
- month: last contact month of year (categorical)
- duration: last contact duration, in seconds (numeric)
- campaign: number of contacts performed during this campaign and for this client (numeric)
- pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric, -1 means client was not previously contacted)
- previous: number of contacts performed before this campaign and for this client (numeric)
- poutcome: outcome of the previous marketing campaign (categorical)

Output variable (desired target): - y - has the client subscribed a term deposit? (binary: “yes”,“no”)

First 6 rows of the dataset.

age	job	marital	education	default	balance	housing	loan	contact	day	month	duration
30	unemployed	married	primary	no	1787	no	no	cellular	19	oct	79
33	services	married	secondary	no	4789	yes	yes	cellular	11	may	220
35	management	single	tertiary	no	1350	yes	no	cellular	16	apr	185
30	management	married	tertiary	no	1476	yes	yes	unknown	3	jun	199
59	blue-collar	married	secondary	no	0	yes	no	unknown	5	may	226
35	management	single	tertiary	no	747	no	no	cellular	23	feb	141

campaign	pdays	previous	poutcome	y
1	-1	0	unknown	no
1	339	4	failure	no
1	330	1	failure	no

4	-1	0	unknown	no
1	-1	0	unknown	no
2	176	3	failure	no

Data cleaning and feature engineering.

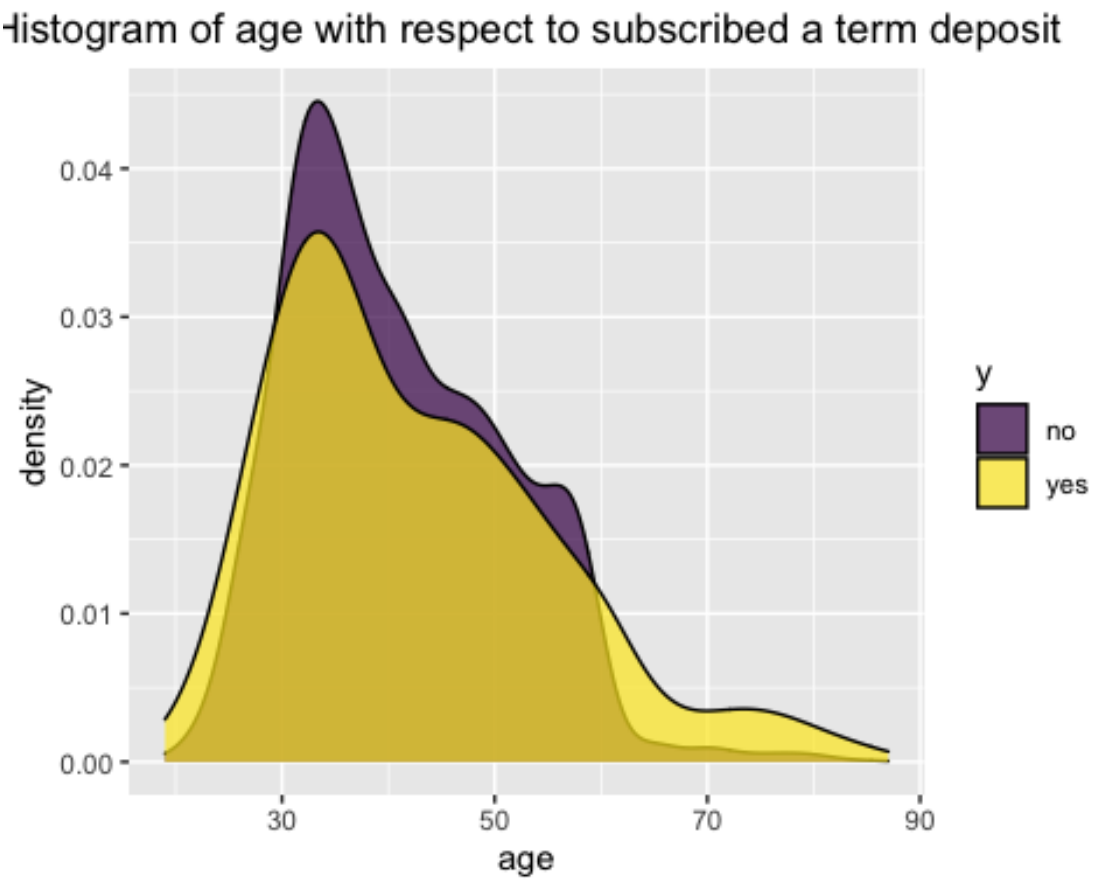
In this tase we unite the day and month variable together and then find the distance between days from decamber 31 and replace the value of -1 for variable pdays by 900. After the cleaning stuffs the data set become. Then change all the character variable type to the factor type.

Table continues below

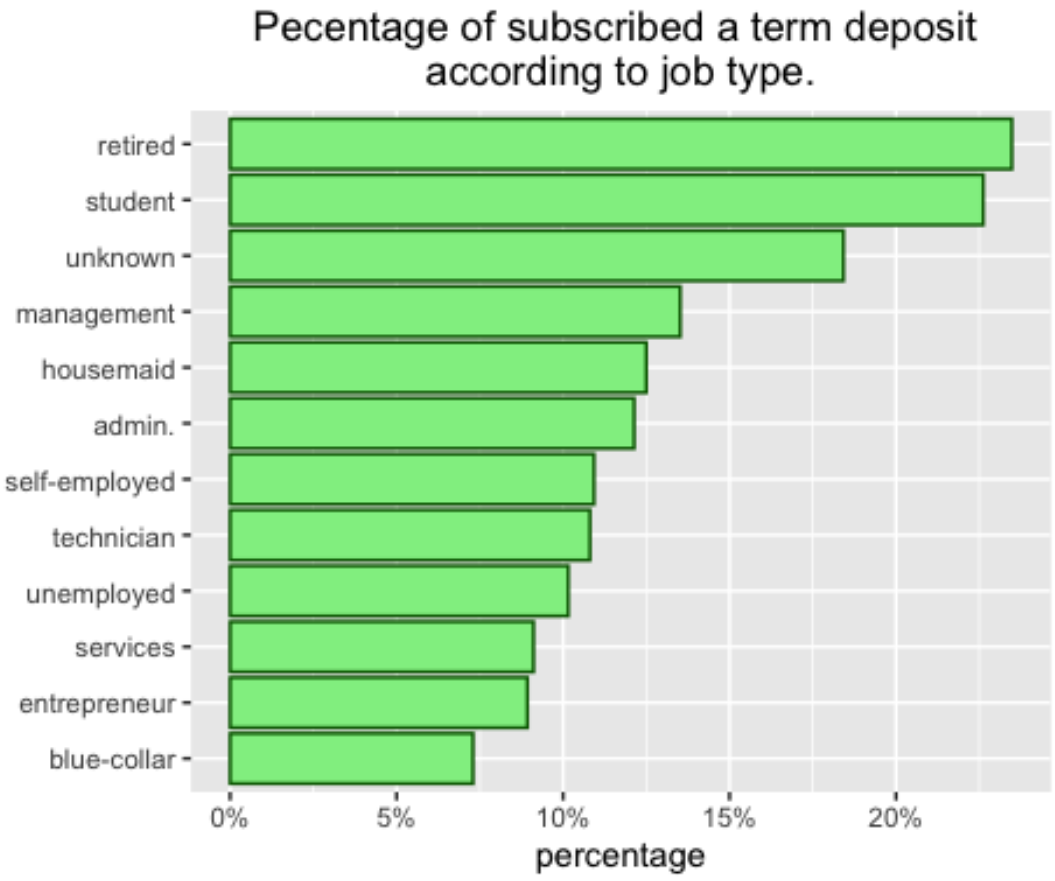
age	job	marital	education	default	balance	housing	loan	contact	date	duration
30	unemployed	married	primary	no	1787	no	no	cellular	73	79
33	services	married	secondary	no	4789	yes	yes	cellular	234	220
35	management	single	tertiary	no	1350	yes	no	cellular	259	185
30	management	married	tertiary	no	1476	yes	yes	unknown	211	199
59	blue-collar	married	secondary	no	0	yes	no	unknown	240	226
35	management	single	tertiary	no	747	no	no	cellular	312	141

campaign	pdays	previous	poutcome	y
1	900	0	unknown	no
1	339	4	failure	no
1	330	1	failure	no
4	900	0	unknown	no
1	900	0	unknown	no
2	176	3	failure	no

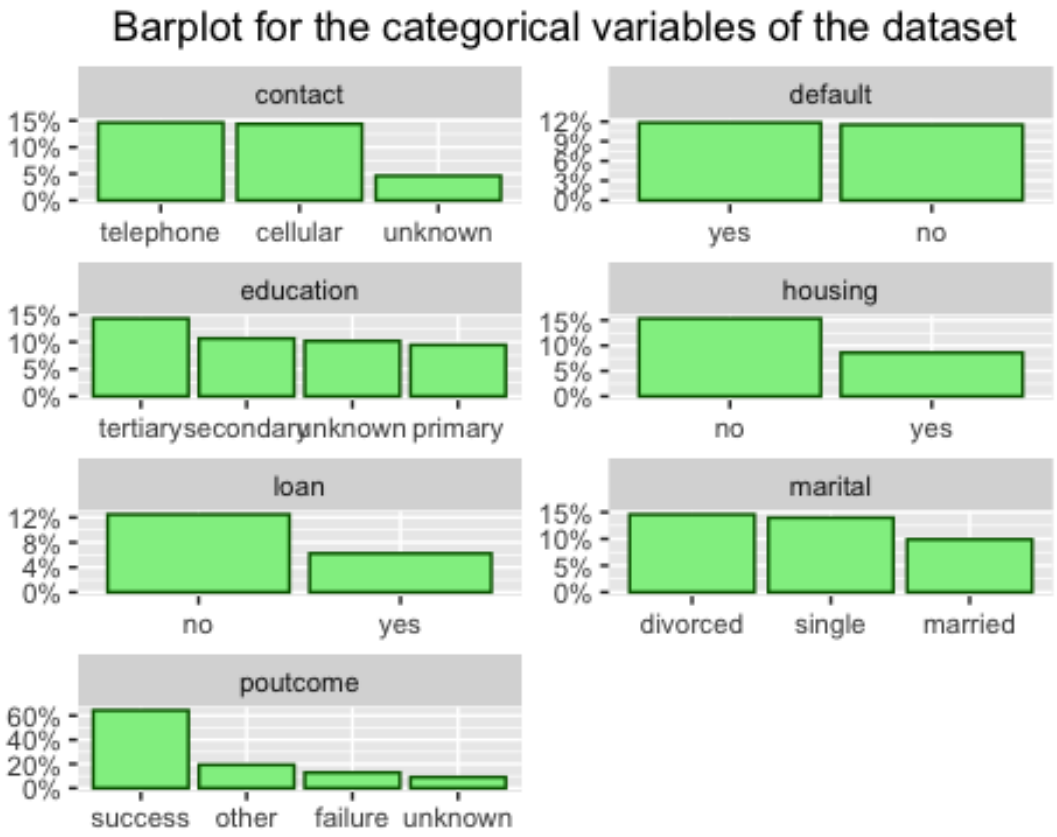
Explanatory data analysis



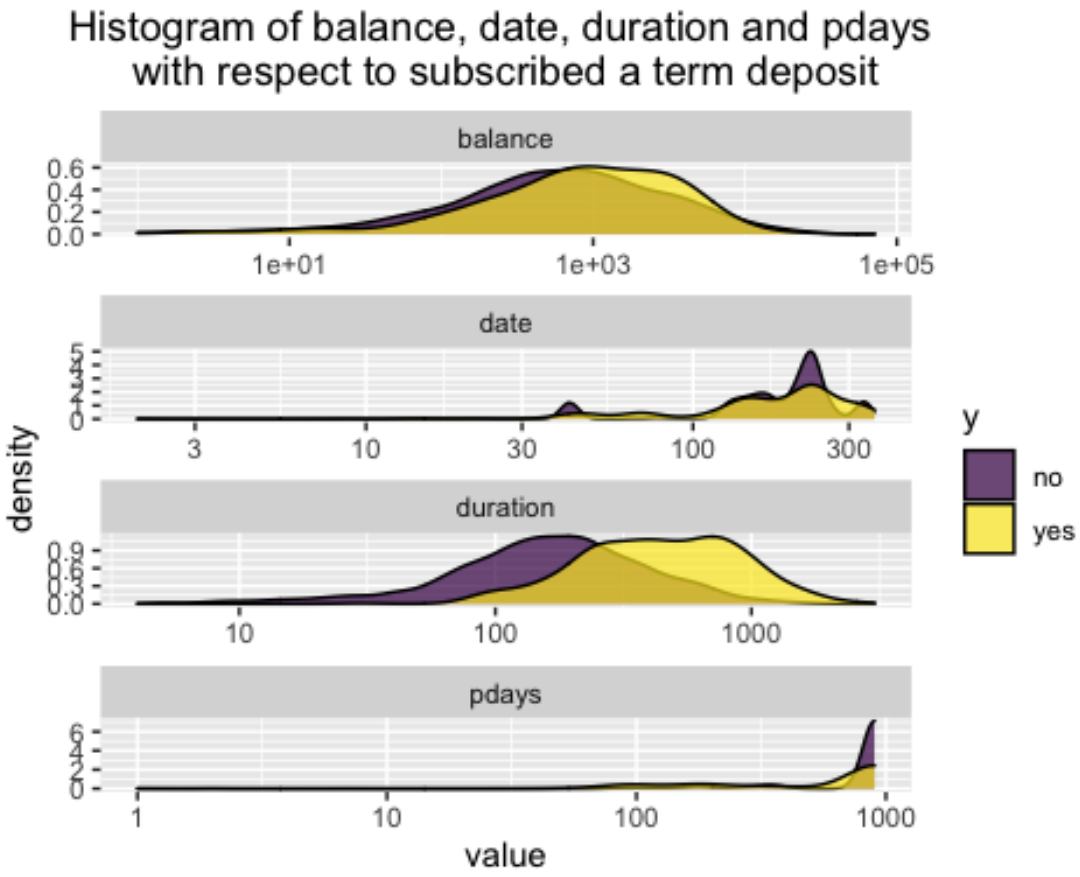
So for the higher age group the tendency of subscribed a term deposit is higher.



So student and retired people use to subscribed a term deposit more in percentage than other people.



So from this graph we can see that the percentage of subscribed of term deposit increase for the contact variable telephone and cellular group; education variable tertiary group; housing variable no group; loan variable no group; marital variable divorced and single group; poutcome variable success group.



From this plot we can see that lower values for balance, date and duration variables has less odds to subscribed a term deposit. On the other hand lower value of variable pdays has higer oddas to subscribe a term deposit.

Models

We will fit some models like - Logistic Regression - Elastic net - Decision Tree - Random Forest - Xgboost

Class imbalance

y	n
no	4000
yes	521

This data set has a class imbalance problem. We will down sample the no class to reduce the effect.

y	n
no	625
yes	521

After this down sample our sample size become 1146 in which no group has 625 observations and yes group has 521 observations.

Spliting data into train and test set

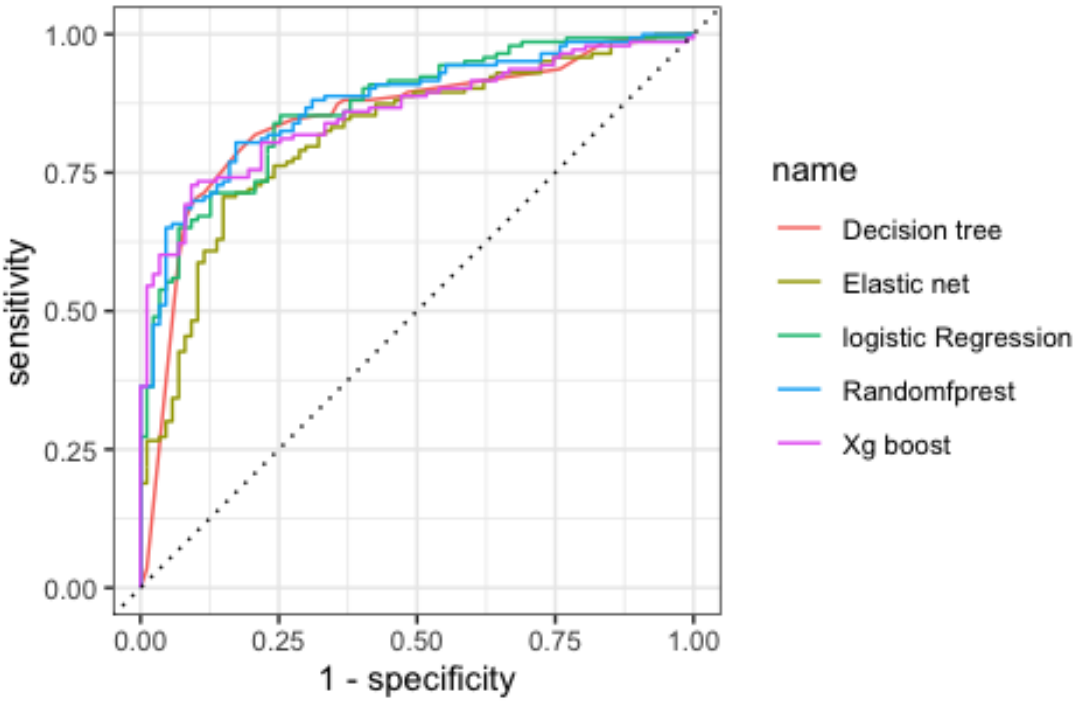
We have 1146 observations in our data set we will split the data in 80% train and 20% test set we get 916 observation in the train set and 230 observations in the test set.

Preprocessing of the data

Before fitting those data to the model we need to pre-processing. For logistic regression, elasticnet regression we need to scale the numeric data and make the categorical variable to dummy variable. For the tree based model we dont need any type of pre-processing.

Hyper-parameters

It is necessary to find out the optimal value of the hyper-parameter to ensure the maximum performance by the model. So for this task we use 10 fold cross validation and we randomly choose some hyper-parameter values in the parameter space and then find out that which set is performing the best. Then we select the values of the hyper-parameter set and fit our model on the full training dataset on this setting.



From this plot we can see that all model performs equally. no we will check for the numerical evidence.

name	.metric	.estimator	.estimate
Randomfprest	roc_auc	binary	0.8752
logistic Regression	roc_auc	binary	0.868
Xg boost	roc_auc	binary	0.8582
Decision tree	roc_auc	binary	0.8478
Elastic net	roc_auc	binary	0.8182

From this we can say that the area under curve for the random forest model is maximum. hence our selected model is random forest.

Performance metric for random forest

.metric	.estimator	.estimate
accuracy	binary	0.8
precision	binary	0.8702
recall	binary	0.7972

So from this table we can see that the accuracy of the random forest model is .8 that is it can identify 80% of the cases correctly. The precision is .87 that is 87% of the cases classified as “yes” by the model is correct. The recall is .797 that is about 80% of all true “yes” class can be classified correctly by the model.

Appendix

```
##Basic data exploration.
skimr::skim(df)

##Data cleaning
df <-df %>%
   unite("date",day:month,sep = "-") %>%
   mutate(date = paste0(date,"-2020"),
          date = dmy(date),
          date = dmy("31-12-2020")-date,
          date = as.numeric(date),
          pdays = ifelse(pdays == -1,900,pdays),
          y = factor(y,c("no","yes"))) %>%
   mutate_if(is.character,as.factor)

##Explonatory data analysis
df %>%
   ggplot(aes(age,fill = y)) +
   geom_density()+
   labs(title = "Histogram of age with respect to subscribed a term deposit") +
   scale_fill_viridis_d(alpha = .7)

df %>%
   group_by(job) %>%
   count(y) %>%
   mutate(prop= n/sum(n)) %>%
   filter(y == "yes") %>%
   ungroup() %>%
   mutate(job = reorder(job,prop)) %>%
   ggplot(aes(y=job,prop)) +
   geom_col(col="darkgreen",fill="lightgreen") +
   scale_x_continuous(labels = scales::percent_format(accuracy = 1)) +
   labs(title = "Pecentage of subscribed a term deposit \naccording to job type.",x="percentage",y="")

df %>%
   select(marital:contact,poutcome,y,-balance) %>%
   pivot_longer(-y) %>%
   group_by(name,value) %>%
   count(y) %>%
   mutate(prop = n/sum(n)) %>%
   filter(y == "yes") %>%
   ungroup() %>%
   mutate(value = tidytext::reorder_within(value,by = -prop,within = name)) %>%
   ggplot(aes(value,prop)) +
   geom_col(col = "darkgreen",fill = "lightgreen")+
   tidytext::scale_x_reordered()+
   facet_wrap(~name,scales = "free",ncol = 2) +
   scale_y_continuous +
  labs(title = "Barplot for the categorical variables of the dataset")

df %>%
   select(balance,date, duration, pdays,y) %>%
   pivot_longer(-y) %>%
   ggplot(aes(value,fill = y)) +
   geom_density()+
   facet_wrap(~name,ncol = 1,scales = "free")+
   scale_x_log10()+
   scale_fill_viridis_d(alpha = .7) +
   labs(title = "Histogram of balance, date, duration and pdays \nwith respect to subscribed a term deposit")

##Class imbalance
df %>%
   count(y) %>%
   pander()

##Model specifications
lr_specs <-
   logistic_reg(mode = "classification") %>%
```

```
  set_engine("glm")

enet_specs <-
  logistic_reg(mode = "classification",penalty = tune(),mixture = tune()) %>%
  set_engine("glmnet")

tree_specs <-
  decision_tree(mode = "classification",cost_complexity = tune(),tree_depth = tune(),min_n = tune()) %>%
  set_engine("rpart")

rf_specs <-
  rand_forest(mode = "classification",trees = 1000,min_n = tune()) %>%
  set_engine("ranger")

xgb_specs <-
  boost_tree(mode = "classification",trees = 1000,min_n = tune(),learn_rate = tune(),tree_depth = tune()) %>%
  set_engine("xgboost")

##Cross validation
set.seed(10)
df_cv <- vfold_cv(df_train)

## recipe
lr_rec <-
  df_train %>%
  recipe(y~ .) %>%
  step_normalize(all_numeric()) %>%
  step_dummy(all_nominal(),-all_outcomes()) %>%
  prep()

tree_rec <-
  df_train %>%
  recipe(y~.) %>%
  prep()

##Function for model fitting
model_fitting <-
  function(name = "lr",model = enet_specs, recipe = lr_rec) {

    if (name != "logistic Regression") {
      set.seed(1234)
      parms_grid <-
        model %>%
        parameters() %>%
        grid_latin_hypercube(size = 10 * nrow(.))

      wf <-
        workflow() %>%
        add_model(model) %>%
        add_recipe(recipe)

      resample_results <-
        tune_grid(
          object = wf,
          grid = parms_grid,
          resamples = df_cv,
          metrics = metric_set(accuracy)
        )

      fit <-
        wf %>%
        finalize_workflow(select_best(resample_results)) %>%
        fit(df_train)
    }

    else
      fit <-
        workflow() %>%
        add_model(model) %>%
```

```
            add_formula(y ~ .) %>%
            fit(df_train)

        print(name)
        return(fit)
    }

##Fitting the model
fit <-
    tibble(id = 1:5) %>%
    mutate(
        name = c(
            "logistic Regression",
            "Elastic net",
            "Decision tree",
            "Randomfprest",
            "Xg boost"
        ),
        model = list(lr_specs, enet_specs, tree_specs, rf_specs, xgb_specs),
        recipe = list(lr_rec, lr_rec, tree_rec, tree_rec, lr_rec),
        fit = pmap(
            list(name, model, recipe),
            ~ model_fitting(
                name = ..1,
                model = ..2,
                recipe = ..3
            )
        )
    )

fit <-
    fit %>%
    transmute(
        name,
        fit,
        split = list(df_split),
        last_fit = map2(
            fit,
            split,
            ~ last_fit(.x, .y))
    )

##Plotting roc-auc
fit %>%
    select(-c(fit,split)) %>%
    unnest(last_fit) %>%
    unnest(.predictions) %>%
    select(name,.pred_no,y) %>%
    group_by(name) %>%
    roc_curve(truth = y, .pred_no) %>%
    autoplot()

##Roc-auc table
fit %>%
  select(-c(fit, split)) %>%
  unnest(last_fit) %>%
  unnest(.predictions) %>%
  select(name, .pred_no, y) %>%
  group_by(name) %>%
  roc_auc(truth = y, .pred_no) %>%
  arrange(-.estimate) %>%
  pander()

##Random forest performance
fit$fit[[4]] %>%
  augment(testing(df_split)) %>%
  accuracy(truth = y,.pred_class) %>%
  bind_rows(fit$fit[[4]] %>%
  augment(testing(df_split)) %>%
```

```
precision(truth = y,.pred_class)) %>%
bind_rows(fit$fit[[4]] %>%
augment(testing(df_split)) %>%
recall(truth = y,.pred_class)) %>%
pander()
```