Apical4 Datathon

Wenqi Zeng, Xiangxuan Yu, Jiayi Pan, Frank Sun

2023-03-24

Loading required library

```
#install.packages('prophet')
library(tidyverse)
library(lubridate)
library(prophet)
library(dplyr)
library(ggplot2)
library(ISLR2)
library(survival)
```

Access data

```
#reading the dataset
training_data <- read_csv("/Users/zengwenqi/Desktop/training_data.csv")
forecast_starting_data <- read_csv("/Users/zengwenqi/Desktop/forecast_starting_data.csv")
```

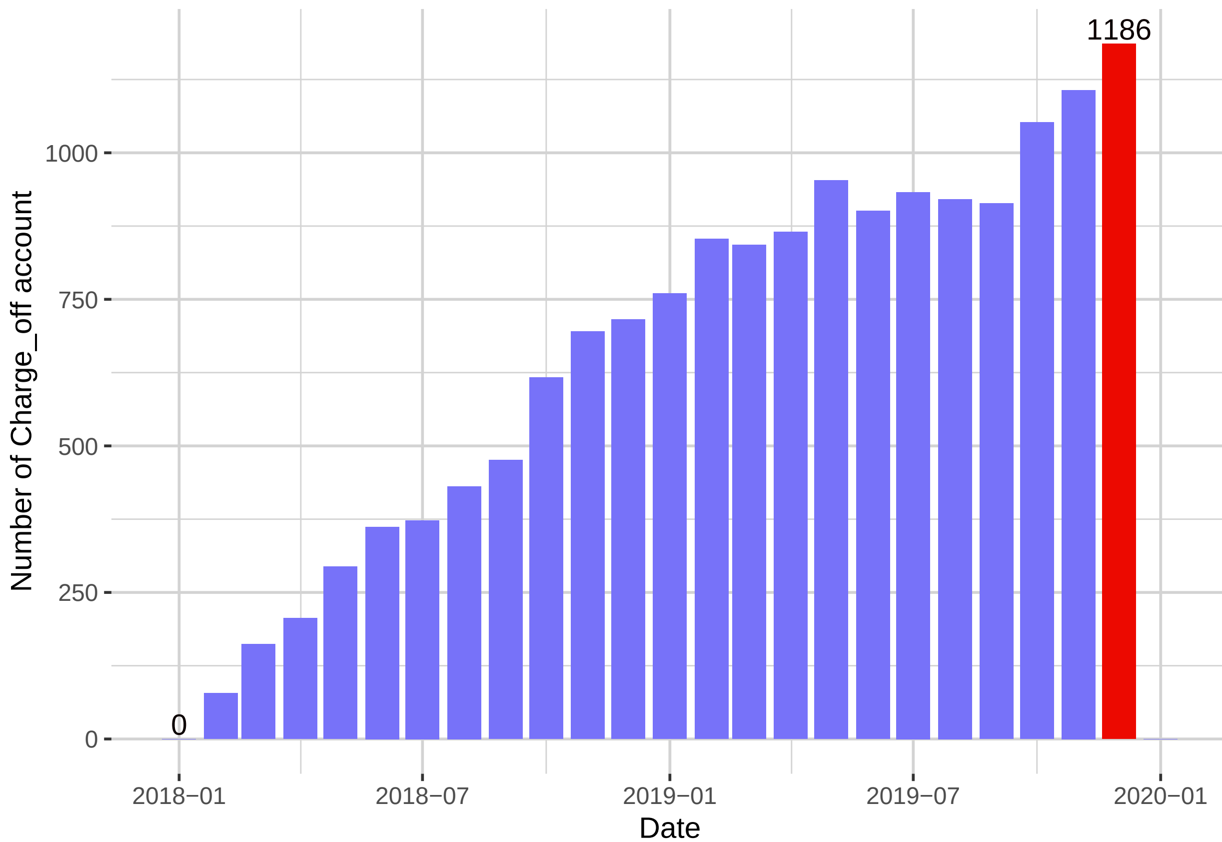
Manipulate and exploring data

```
#including forecast starting dataset in 2020-01
full_df = rbind(training_data,forecast_starting_data)
```

```
#changing some data type to date format
df2=full_df
df2$snapshot<-lubridate::ym(df2$snapshot)
df2$mth_code<-lubridate::ym(df2$mth_code)
df2 <- df2 %>%
  mutate(time_diff = as.numeric(interval(snapshot, mth_code) / months(1)))
```

```
# count charge_off account in each month which is ending observation
coun = df2 %>%
  group_by(mth_code) %>%
  summarise(total=sum(charge_off))
```

```
library(ggplot2)

ggplot(coun, aes(x = mth_code, y = total)) +
  geom_bar(stat = "identity", fill = ifelse(1:nrow(coun) == which.max(coun$total),"#ec0900", "#7772f9"))
  labs(x = "Date", y ="Number of Charge_off account" ) +
  geom_text(aes(label = ifelse(1:nrow(coun) == which.max(coun$total), max(coun$total), "")),
           vjust = -0.2, color = "#0c0000") +
  geom_text(aes(label = ifelse(1:nrow(coun) == which.min(coun$total), min(coun$total), "")),
           vjust = -0.2, color = "#0c0000")+
  theme(
  panel.background = element_rect(fill = "white",
                               size = 2, linetype = "solid"),
  panel.grid.major = element_line(size = 0.5, linetype = 'solid',
                               colour = "lightgrey"),
  panel.grid.minor = element_line(size = 0.25, linetype = 'solid',
                               colour = "lightgrey")
  )
```



The number of charge_off increases along the time, with the number of 1186 in 2019/12.

Survival anaylsis to select variables to do the grouping

We decided to use Cox Proportional Hazards Model as a reference to help us decide what variables in the training dataset should be picked up. At the end, we want to select related variables to group data at the aggregated level.

```r
#We only have data from 2018-2019, we want to first drop continuous variables because we have no inform
first_removing=df2%>%
  mutate(time_diff = as.numeric(interval(snapshot, mth_code) / months(1)))%>%
  select(-contains("due_balance"))%>%
  select(-c('snapshot','mth_code','account_status_code','bank_fico_buckets_20','charge_off_reason_code'
  drop_na()
```

```r
# first, to
fit.all <- coxph(Surv(time_diff,charge_off) ~., data=first_removing)
summary(fit.all)
```

```
## Call:
## coxph(formula = Surv(time_diff, charge_off) ~ ., data = first_removing)
##
##   n= 5778085, number of events= 15698
##
##                                  coef  exp(coef)   se(coef)       z Pr(>|z|)
## financial_active            9.553e-02  1.100e+00  4.149e-02   2.302 0.021312
## promotion_flag             -2.267e-02  9.776e-01  1.601e-02  -1.416 0.156725
## variable_rate_index        -5.846e-02  9.432e-01  1.973e-02  -2.962 0.003056
## active_12_mths              1.015e-01  1.107e+00  6.341e-02   1.602 0.109251
## open_closed_flag           -1.187e+00  3.053e-01  1.609e-02 -73.744  < 2e-16
## ever_delinquent_flg         7.207e-02  1.075e+00  2.156e-02   3.343 0.000828
## purchase_active            -6.695e-02  9.352e-01  1.865e-01  -0.359 0.719597
## closed                      3.146e-02  1.032e+00  4.719e+42   0.000 1.000000
## active                     -1.237e+02  1.828e-54  1.739e+30   0.000 1.000000
## charge_off_aged             2.535e-01  1.289e+00  2.695e-02   9.408  < 2e-16
## charge_off_bk               1.965e+00  7.137e+00  1.599e-02 122.889  < 2e-16
## writeoff_type_bko          -5.187e-01  5.953e-01  3.026e-02 -17.143  < 2e-16
## writeoff_type_fraud_kiting -1.513e+01  2.699e-07  6.585e+02  -0.023 0.981674
## writeoff_type_fraud_synthetic  0.000e+00  1.000e+00  0.000e+00     NaN      NaN
## writeoff_type_deceased      1.400e+00  4.055e+00  2.245e-02  62.373  < 2e-16
## writeoff_type_other         0.000e+00  1.000e+00  0.000e+00     NaN      NaN
## writeoff_type_aged          1.775e+01  5.109e+07  1.108e+02   0.160 0.872785
## writeoff_type_settlement    2.706e-01  1.311e+00  4.038e-02   6.700 2.08e-11
## writeoff_type_fraud_other  -6.370e-03  9.936e-01  6.937e+41   0.000 1.000000
## writeoff_type_repo          0.000e+00  1.000e+00  0.000e+00     NaN      NaN
## writeoff_type_null         -5.739e+01  1.192e-25  3.563e+11   0.000 1.000000
## due_account_2              -2.801e-01  7.557e-01  8.961e-02  -3.125 0.001776
## due_account_3              -2.197e-01  8.028e-01  6.879e-02  -3.193 0.001406
## due_account_4              -1.518e-01  8.592e-01  8.721e-02  -1.740 0.081841
## due_account_5              -2.381e-01  7.881e-01  8.938e-02  -2.664 0.007716
## due_account_6              -6.635e-02  9.358e-01  1.063e-01  -0.624 0.532451
## due_account_7              -4.005e-02  9.607e-01  8.288e-02  -0.483 0.628907
## due_account_8               6.377e-02  1.066e+00  2.657e+43   0.000 1.000000
## industryB                   1.153e-01  1.122e+00  1.598e-02   7.217 5.32e-13
## industryC                  -4.825e-03  9.952e-01  2.254e-02  -0.214 0.830548
##
## financial_active           *
## promotion_flag
## variable_rate_index        **
## active_12_mths
## open_closed_flag           ***
```

```
## ever_delinquent_flg           ***
## purchase_active
## closed
## active
## charge_off_aged               ***
## charge_off_bk                 ***
## writeoff_type_bko             ***
## writeoff_type_fraud_kiting
## writeoff_type_fraud_synthetic
## writeoff_type_deceased        ***
## writeoff_type_other
## writeoff_type_aged
## writeoff_type_settlement      ***
## writeoff_type_fraud_other
## writeoff_type_repo
## writeoff_type_null
## due_account_2                 **
## due_account_3                 **
## due_account_4                 .
## due_account_5                 **
## due_account_6
## due_account_7
## due_account_8
## industryB                     ***
## industryC
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                               exp(coef) exp(-coef) lower .95  upper .95
## financial_active              1.100e+00  9.089e-01 1.014e+00  1.193e+00
## promotion_flag                9.776e-01  1.023e+00 9.474e-01  1.009e+00
## variable_rate_index           9.432e-01  1.060e+00 9.074e-01  9.804e-01
## active_12_mths                1.107e+00  9.034e-01 9.775e-01  1.253e+00
## open_closed_flag              3.053e-01  3.276e+00 2.958e-01  3.150e-01
## ever_delinquent_flg           1.075e+00  9.305e-01 1.030e+00  1.121e+00
## purchase_active               9.352e-01  1.069e+00 6.489e-01  1.348e+00
## closed                        1.032e+00  9.690e-01 0.000e+00        Inf
## active                        1.828e-54  5.470e+53 0.000e+00        Inf
## charge_off_aged               1.289e+00  7.760e-01 1.222e+00  1.358e+00
## charge_off_bk                 7.137e+00  1.401e-01 6.917e+00  7.364e+00
## writeoff_type_bko             5.953e-01  1.680e+00 5.610e-01  6.317e-01
## writeoff_type_fraud_kiting    2.699e-07  3.705e+06 0.000e+00        Inf
## writeoff_type_fraud_synthetic 1.000e+00  1.000e+00 1.000e+00  1.000e+00
## writeoff_type_deceased        4.055e+00  2.466e-01 3.881e+00  4.238e+00
## writeoff_type_other           1.000e+00  1.000e+00 1.000e+00  1.000e+00
## writeoff_type_aged            5.109e+07  1.957e-08 2.264e-87 1.153e+102
## writeoff_type_settlement      1.311e+00  7.630e-01 1.211e+00  1.419e+00
## writeoff_type_fraud_other     9.936e-01  1.006e+00 0.000e+00        Inf
## writeoff_type_repo            1.000e+00  1.000e+00 1.000e+00  1.000e+00
## writeoff_type_null            1.192e-25  8.388e+24 0.000e+00        Inf
## due_account_2                 7.557e-01  1.323e+00 6.340e-01  9.008e-01
## due_account_3                 8.028e-01  1.246e+00 7.015e-01  9.186e-01
## due_account_4                 8.592e-01  1.164e+00 7.242e-01  1.019e+00
## due_account_5                 7.881e-01  1.269e+00 6.615e-01  9.390e-01
```

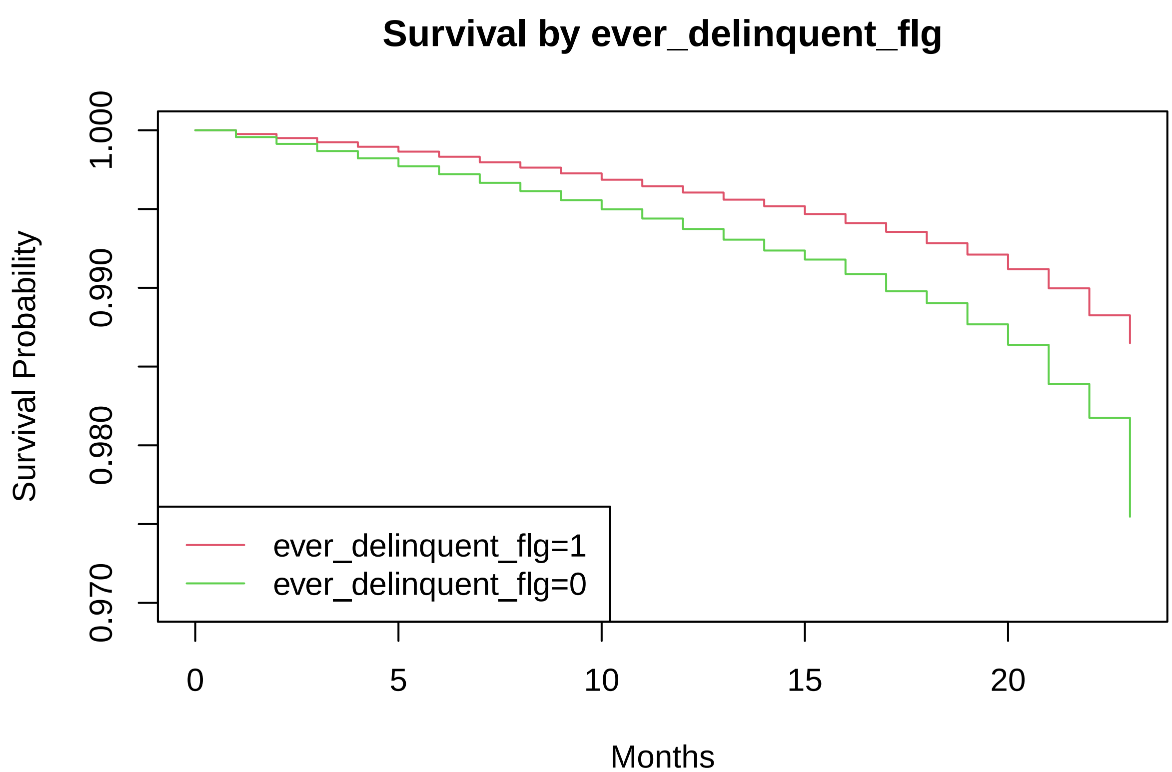
```
## due_account_6                  9.358e-01  1.069e+00 7.598e-01  1.153e+00
## due_account_7                  9.607e-01  1.041e+00 8.167e-01  1.130e+00
## due_account_8                  1.066e+00  9.382e-01 0.000e+00        Inf
## industryB                      1.122e+00  8.911e-01 1.088e+00  1.158e+00
## industryC                      9.952e-01  1.005e+00 9.522e-01  1.040e+00
##
## Concordance= 0.999  (se = 0 )
## Likelihood ratio test= 60426  on 30 df,   p=<2e-16
## Wald test            = 0  on 30 df,   p=1
## Score (logrank) test = 6020183  on 30 df,   p=<2e-16
```

```
#We drop categorigal variables that have very large p-value or p-value equals to NA
second_removing=first_removing%>%
  select(-c('closed','active','writeoff_type_other','writeoff_type_fraud_synthetic','writeoff_type_repo
fit.1 <- coxph(Surv(time_diff,charge_off) ~., data=second_removing)
summary(fit.1)
```

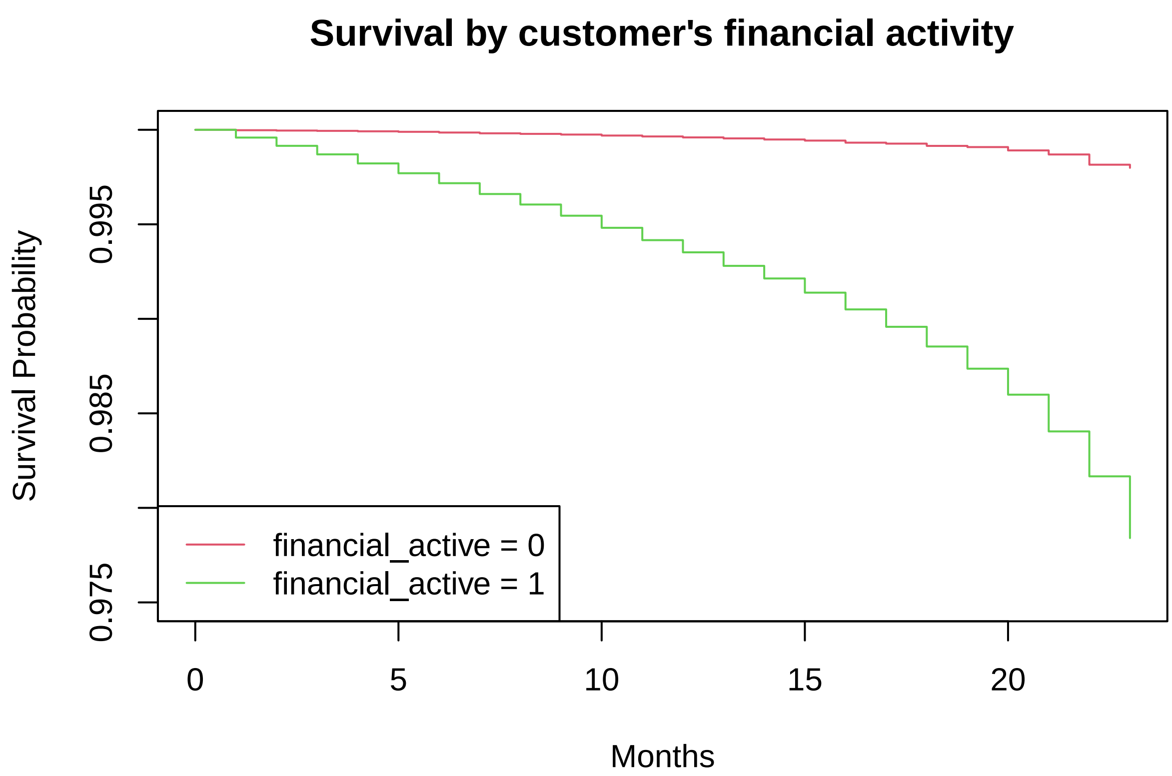
```
## Call:
## coxph(formula = Surv(time_diff, charge_off) ~ ., data = second_removing)
##
##   n= 5778085, number of events= 15698
##
##                                coef exp(coef)  se(coef)       z Pr(>|z|)
## financial_active            0.96101   2.61434   0.04457  21.563  < 2e-16 ***
## promotion_flag              0.06326   1.06530   0.01614   3.920 8.86e-05 ***
## variable_rate_index        -0.04376   0.95718   0.01928  -2.270 0.023217 *
## active_12_mths              0.68668   1.98710   0.06128  11.206  < 2e-16 ***
## open_closed_flag           -0.38554   0.68008   0.01778 -21.678  < 2e-16 ***
## ever_delinquent_flg         0.05868   1.06043   0.02147   2.733 0.006277 **
## purchase_active            -2.35974   0.09445   0.15403 -15.320  < 2e-16 ***
## charge_off_aged             6.70936 820.04196   0.03845 174.485  < 2e-16 ***
## charge_off_bk               0.17316   1.18906   0.02527   6.851 7.32e-12 ***
## writeoff_type_bko           0.12887   1.13754   0.02526   5.102 3.36e-07 ***
## writeoff_type_fraud_kiting -0.22522   0.79834   0.37832  -0.595 0.551636
## writeoff_type_deceased      0.14723   1.15862   0.03734   3.943 8.04e-05 ***
## writeoff_type_aged          0.86382   2.37220   0.02887  29.918  < 2e-16 ***
## writeoff_type_settlement    0.14115   1.15159   0.04338   3.254 0.001138 **
## due_account_2              -0.23230   0.79271   0.07866  -2.953 0.003145 **
## due_account_3               0.06761   1.06995   0.05988   1.129 0.258813
## due_account_4               0.11318   1.11984   0.07068   1.601 0.109289
## due_account_5              -0.07298   0.92962   0.08693  -0.840 0.401137
## due_account_6               0.15550   1.16824   0.10360   1.501 0.133367
## due_account_7               0.36192   1.43608   0.08474   4.271 1.95e-05 ***
## industryB                  -0.04671   0.95436   0.01769  -2.641 0.008262 **
## industryC                  -0.08416   0.91928   0.02440  -3.449 0.000562 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                            exp(coef) exp(-coef) lower .95 upper .95
## financial_active             2.61434   0.382505   2.39567    2.8530
## promotion_flag               1.06530   0.938699   1.03214    1.0995
## variable_rate_index          0.95718   1.044734   0.92169    0.9940
## active_12_mths               1.98710   0.503245   1.76222    2.2407
## open_closed_flag             0.68008   1.470410   0.65678    0.7042
```

```
## ever_delinquent_flg          1.06043   0.943013  1.01673    1.1060
## purchase_active              0.09445  10.588150  0.06983    0.1277
## charge_off_aged            820.04196   0.001219 760.51071  884.2332
## charge_off_bk                1.18906   0.841003  1.13159    1.2494
## writeoff_type_bko            1.13754   0.879090  1.08260    1.1953
## writeoff_type_fraud_kiting   0.79834   1.252596  0.38033    1.6758
## writeoff_type_deceased       1.15862   0.863095  1.07686    1.2466
## writeoff_type_aged           2.37220   0.421550  2.24168    2.5103
## writeoff_type_settlement     1.15159   0.868362  1.05774    1.2538
## due_account_2                0.79271   1.261492  0.67946    0.9248
## due_account_3                1.06995   0.934624  0.95147    1.2032
## due_account_4                1.11984   0.892988  0.97498    1.2862
## due_account_5                0.92962   1.075711  0.78400    1.1023
## due_account_6                1.16824   0.855989  0.95356    1.4312
## due_account_7                1.43608   0.696341  1.21632    1.6955
## industryB                    0.95436   1.047819  0.92185    0.9880
## industryC                    0.91928   1.087807  0.87635    0.9643
##
## Concordance= 0.996  (se = 0 )
## Likelihood ratio test= 160925  on 22 df,   p=<2e-16
## Wald test            = 103294  on 22 df,   p=<2e-16
## Score (logrank) test = 5590467  on 22 df,   p=<2e-16
```

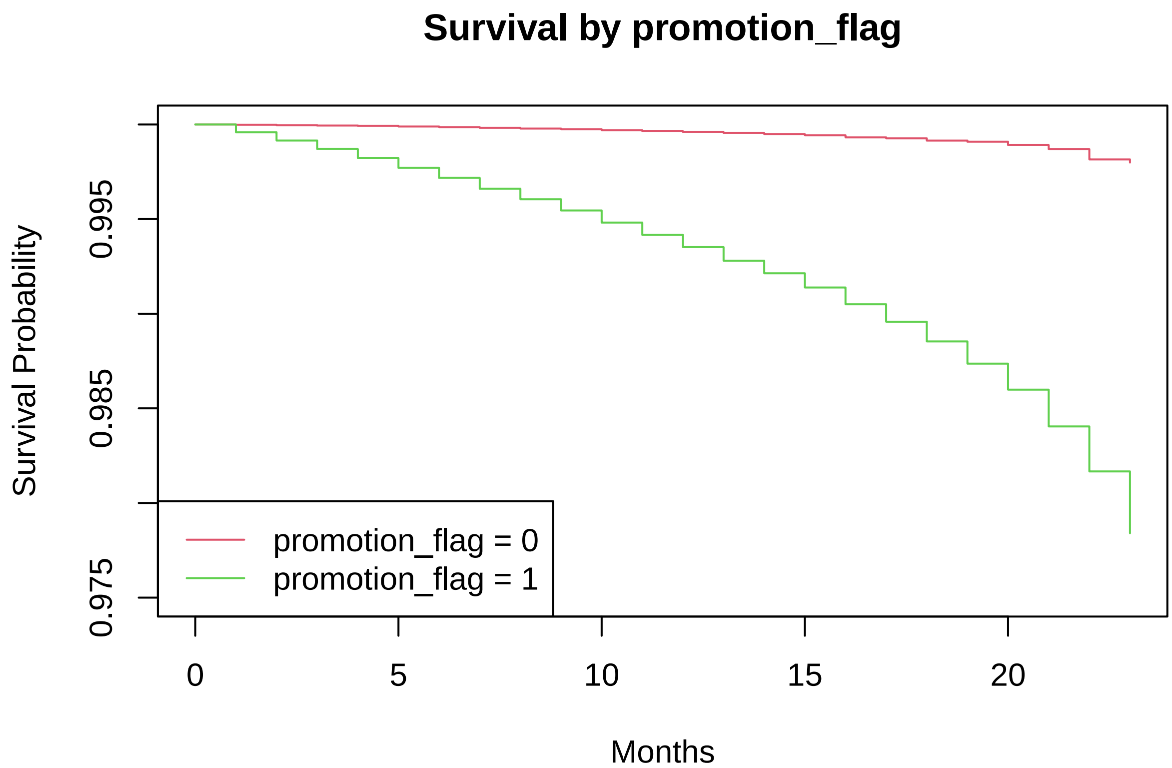
```
delip<- survfit(Surv(time_diff,charge_off) ~ ever_delinquent_flg, data=second_removing)
plot(delip, xlab="Months",ylab="Survival Probability",col=2:3, main="Survival by ever_delinquent_flg",yl
legend("bottomleft", c("ever_delinquent_flg=1 ","ever_delinquent_flg=0"), col = 2:3, lty = 1)
```



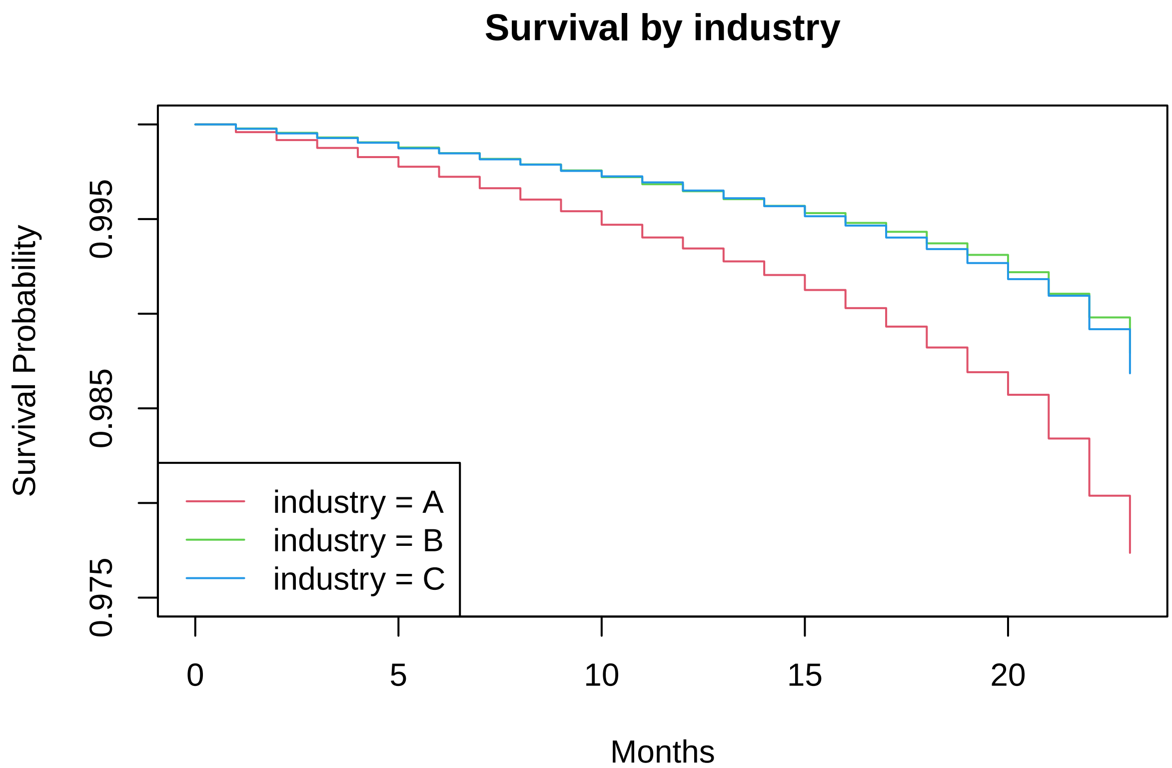
```
financial<- survfit(Surv(time_diff,charge_off) ~ financial_active , data=second_removing)
plot(financial, xlab="Months",ylab="Survival Probability",col=2:3, main="Survival by customer's financi
legend("bottomleft", c("financial_active = 0","financial_active = 1"), col = 2:3, lty = 1)
```




```
promotion_flag<- survfit(Surv(time_diff,charge_off) ~ promotion_flag , data=second_removing)
plot(financial, xlab="Months",ylab="Survival Probability",col=2:3, main="Survival by promotion_flag",yl
legend("bottomleft", c("promotion_flag = 0","promotion_flag = 1"), col = 2:3, lty = 1)
```




```
ind<- survfit(Surv(time_diff,charge_off) ~ industry , data=second_removing)
plot(ind, xlab="Months",ylab="Survival Probability",col=2:4, main="Survival by industry",ylim=c(0.975,1
legend("bottomleft", c("industry = A","industry = B", "industry = C"), col = 2:4, lty = 1)
```



With the fit.1 model, we detect 4 variables that have both statistical meanings and economic meanings in this context, which is financial_active, industry, ever_delinquent_fl, and promotion_flag. When looking at the Kaplan Meier (KM) survival curves, we noticed that industry and financial_active have very obvious different probabilities between each levels and may have significant effects on charge_off activities. Therefore, we decided to pick up these two variables to group and stratify the data at the end in order to make prediction more accurate.

Model forecasting

predict in a original data without grouping data by the variables we detected above

```
#import macro dataset
macro <- read_csv("/Users/zengwenqi/Desktop/macro_data.csv",
    col_names = FALSE, col_types = cols(X1 = col_date(format = "%m/%d/%Y")))
colnames(macro)[colnames(macro) == "X1"] <- "mth_code"
macro$mth_code <- as.Date(sub("\\d{2}$", "01", macro$mth_code))
macro=macro[9:440,]
macro
```

```
## # A tibble: 432 x 97
##    mth_code   X2     X3    X4    X5    X6    X7    X8    X9    X10   X11   X12
##    <date>     <chr>  <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
##  1 2000-01-01 136.3~ 5715~ 2921~ 5     6542~ 2974~ 9914~ 7925~ 5114~ 101.~ 1228~
```

```
##  2 2000-02-01 136.1~ 5754~ 2965~ 4.5   6625~ 2994~ 9989~ 7978~ 5128~ 102.~ 1344~
##  3 2000-03-01 137.2~ 5777~ 2997~ 4.3   6686~ 3013~ 1010~ 8030~ 5142~ 103.~ 1205~
##  4 2000-04-01 138.1~ 5787~ 2949~ 4.8   6679~ 3032~ 1017~ 8064~ 5155~ 104.~ 1081~
##  5 2000-05-01 138.7~ 5772~ 2954~ 4.8   6709~ 3050~ 1026~ 8096~ 5167~ 104.~ 1286~
##  6 2000-06-01 139.7~ 5815~ 2977~ 4.8   6746~ 3067~ 1030~ 8143~ 5179~ 105.~ 1300~
##  7 2000-07-01 140.4~ 5885~ 2966~ 5.1   6768~ 3083~ 1028~ 8202~ 5189~ 106.~ 1241~
##  8 2000-08-01 140.9~ 5899~ 2970~ 5.2   6802~ 3097~ 1028~ 8240~ 5200~ 107.~ 1433~
##  9 2000-09-01 141.8~ 5927~ 3022~ 4.5   6888~ 3109~ 1038~ 8275~ 5210~ 107.~ 1267~
## 10 2000-10-01 142.6~ 5942~ 3014~ 4.8   6893~ 3119~ 1040~ 8299~ 5220~ 108.~ 1346~
## # ... with 422 more rows, and 85 more variables: X13 <chr>, X14 <chr>,
## #   X15 <chr>, X16 <chr>, X17 <chr>, X18 <chr>, X19 <chr>, X20 <chr>,
## #   X21 <chr>, X22 <chr>, X23 <chr>, X24 <chr>, X25 <chr>, X26 <chr>,
## #   X27 <chr>, X28 <chr>, X29 <chr>, X30 <chr>, X31 <chr>, X32 <chr>,
## #   X33 <chr>, X34 <chr>, X35 <chr>, X36 <chr>, X37 <chr>, X38 <chr>,
## #   X39 <chr>, X40 <chr>, X41 <chr>, X42 <chr>, X43 <chr>, X44 <chr>,
## #   X45 <chr>, X46 <chr>, X47 <chr>, X48 <chr>, X49 <chr>, X50 <chr>, ...
```

```r
#macro data during predicting period
prediction_macro=macro[macro$mth_code>='2020-02-01' & macro$mth_code<='2021-01-01',]
colnames(prediction_macro)=c('ds',paste('macro',1:96,sep=''))
prediction_macro
```

```
## # A tibble: 12 x 97
##    ds         macro1     macro2 macro3 macro4 macro5 macro6 macro7 macro8 macro9
##    <date>     <chr>      <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
##  1 2020-02-01 279.61505~ 11922~ 5494.~ 9.3    14785~ 58491~ 21937~ 15128~ 82715~
##  2 2020-03-01 280.37714~ 11613~ 5188.~ 13.8   13805~ 58800~ 20739~ 14865~ 82911~
##  3 2020-04-01 281.16714~ 10746~ 4511.~ 33.8   12082~ 59133~ 18746~ 16638~ 83103~
##  4 2020-05-01 280.46397~ 11070~ 5304.~ 24.9   13129~ 59487~ 19641~ 15995~ 83298~
##  5 2020-06-01 283.55849~ 11343~ 5666.~ 20.1   13937~ 59866~ 20522~ 15956~ 83499~
##  6 2020-07-01 287.22478~ 11452~ 5774.~ 19.2   14230~ 60270~ 21028~ 16102~ 83696~
##  7 2020-08-01 291.51485~ 11573~ 5788.~ 15.5   14352~ 60704~ 21364~ 15602~ 83874~
##  8 2020-09-01 296.18066~ 11663~ 5912.4 14.6   14583~ 61146~ 21694~ 15693~ 84023~
##  9 2020-10-01 300.76860~ 11866~ 5889.~ 14     14626~ 61597~ 21749~ 15663~ 84186~
## 10 2020-11-01 304.18276~ 11988~ 5841.~ 13.3   14560~ 62048~ 21672~ 15507~ 84411~
## 11 2020-12-01 308.12946~ 12062~ 5866.~ 13.8   14571~ 62493~ 21692~ 15597~ 84729~
## 12 2021-01-01 311.66217~ 12065~ 6159.~ 20     14932~ 62930~ 22084~ 17081~ 85112~
## # ... with 87 more variables: macro10 <chr>, macro11 <chr>, macro12 <chr>,
## #   macro13 <chr>, macro14 <chr>, macro15 <chr>, macro16 <chr>, macro17 <chr>,
## #   macro18 <chr>, macro19 <chr>, macro20 <chr>, macro21 <chr>, macro22 <chr>,
## #   macro23 <chr>, macro24 <chr>, macro25 <chr>, macro26 <chr>, macro27 <chr>,
## #   macro28 <chr>, macro29 <chr>, macro30 <chr>, macro31 <chr>, macro32 <chr>,
## #   macro33 <chr>, macro34 <chr>, macro35 <chr>, macro36 <chr>, macro37 <chr>,
## #   macro38 <chr>, macro39 <chr>, macro40 <chr>, macro41 <chr>, ...
```

```r
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]

training_data2=df2%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
```

```r
#changing col names
colnames(training_data2)=c('ds','y',paste('macro',1:96,sep=''))
training_data2
```

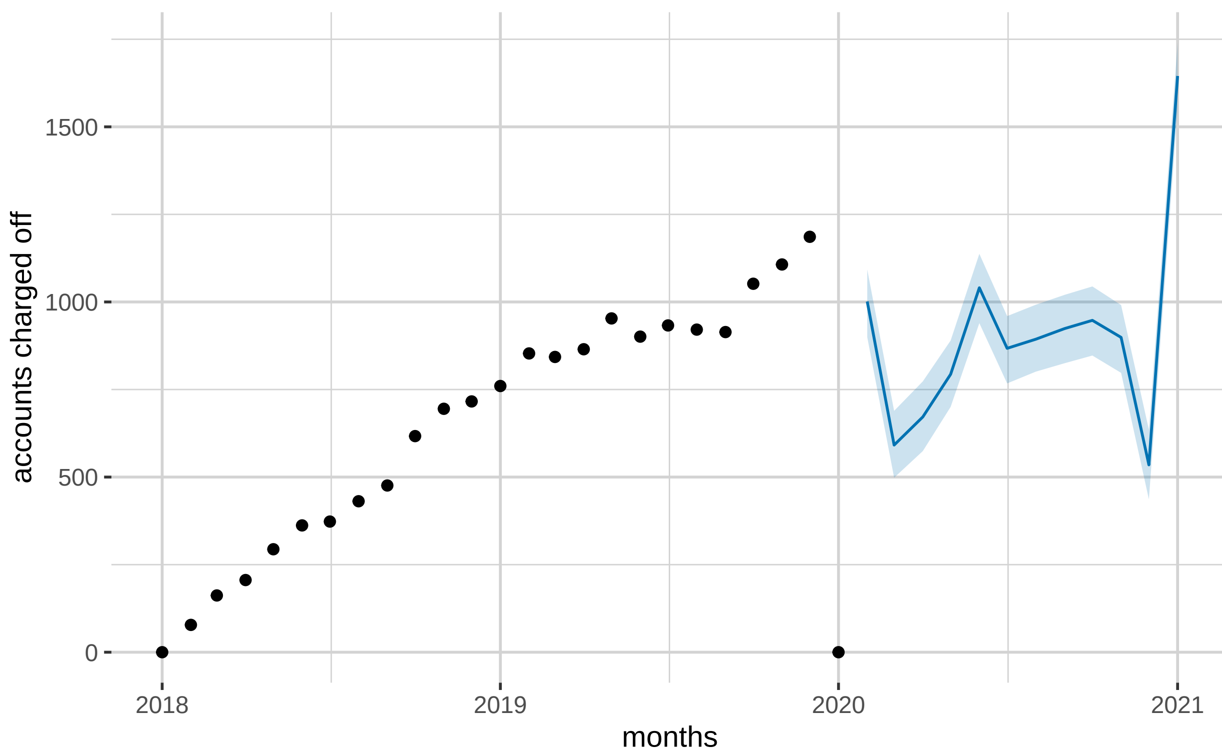
```
## # A tibble: 25 x 98
##    ds             y macro1      macro2 macro3 macro4 macro5 macro6 macro7 macro8
##    <date>     <dbl> <chr>       <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
##  1 2018-01-01     0 248.873161  10761~ 5181.~ 7.1    13628~ 53049~ 20073~ 13817~
##  2 2018-02-01    78 250.8678133 10780~ 5215.~ 7.2    13668~ 53233~ 20141~ 13857~
##  3 2018-03-01   162 251.6692577 10808~ 5198.~ 7.2    13735~ 53417~ 20251~ 13901~
##  4 2018-04-01   206 252.3962837 10850~ 5220.~ 7.2    13792~ 53607~ 20361~ 13941~
##  5 2018-05-01   294 253.7272947 10884~ 5277.~ 7.2    13860~ 53798~ 20482~ 13995~
##  6 2018-06-01   362 254.8264638 10938~ 5250.~ 7.4    13900~ 53991~ 20566~ 14054~
##  7 2018-07-01   373 255.8970963 11000~ 5285.~ 7.5    13952~ 54185~ 20646~ 14126~
##  8 2018-08-01   431 257.1583704 11056~ 5278.~ 7.6    14001~ 54384~ 20705~ 14181~
##  9 2018-09-01   476 257.5543485 11071~ 5272.~ 7.7    14013~ 54582~ 20709~ 14196~
## 10 2018-10-01   617 258.4575107 11075~ 5329.~ 7.6    14096~ 54782~ 20786~ 14240~
## # ... with 15 more rows, and 88 more variables: macro9 <chr>, macro10 <chr>,
## #   macro11 <chr>, macro12 <chr>, macro13 <chr>, macro14 <chr>, macro15 <chr>,
## #   macro16 <chr>, macro17 <chr>, macro18 <chr>, macro19 <chr>, macro20 <chr>,
## #   macro21 <chr>, macro22 <chr>, macro23 <chr>, macro24 <chr>, macro25 <chr>,
## #   macro26 <chr>, macro27 <chr>, macro28 <chr>, macro29 <chr>, macro30 <chr>,
## #   macro31 <chr>, macro32 <chr>, macro33 <chr>, macro34 <chr>, macro35 <chr>,
## #   macro36 <chr>, macro37 <chr>, macro38 <chr>, macro39 <chr>, ...
```

```r
#using fb prophet forecasting procedure to perform a Time Series forecasting

#idenity all regressors
regressors <- training_data2 %>% select(-ds, -y)

#fitting all regressors
for (col in names(regressors)) {
  model <- prophet() %>% add_regressor(col, mode = "additive")
}

#fitting the model
model <- fit.prophet(model, training_data2)

#tail
tail(prediction_macro)
```

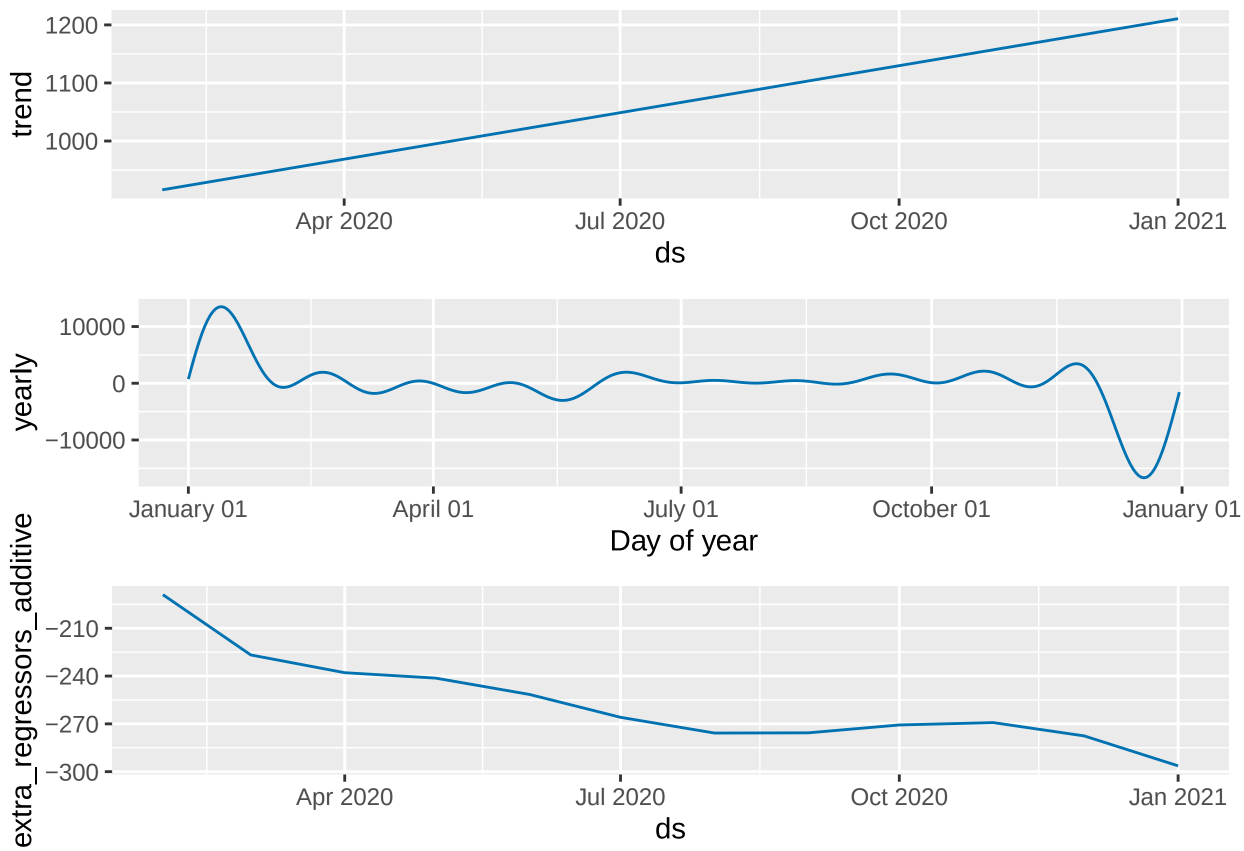
```
## # A tibble: 6 x 97
##   ds         macro1      macro2 macro3 macro4 macro5 macro6 macro7 macro8 macro9
##   <date>     <chr>       <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>  <chr>
## 1 2020-08-01 291.5148534 11573~ 5788.~ 15.5   14352~ 60704~ 21364~ 15602~ 83874~
## 2 2020-09-01 296.1806646 11663~ 5912.4 14.6   14583~ 61146~ 21694~ 15693~ 84023~
## 3 2020-10-01 300.7686026 11866~ 5889.~ 14     14626~ 61597~ 21749~ 15663~ 84186~
## 4 2020-11-01 304.1827623 11988~ 5841.~ 13.3   14560~ 62048~ 21672~ 15507~ 84411~
## 5 2020-12-01 308.1294672 12062~ 5866.~ 13.8   14571~ 62493~ 21692~ 15597~ 84729~
## 6 2021-01-01 311.6621741 12065~ 6159.~ 20     14932~ 62930~ 22084~ 17081~ 85112~
## # ... with 87 more variables: macro10 <chr>, macro11 <chr>, macro12 <chr>,
## #   macro13 <chr>, macro14 <chr>, macro15 <chr>, macro16 <chr>, macro17 <chr>,
## #   macro18 <chr>, macro19 <chr>, macro20 <chr>, macro21 <chr>, macro22 <chr>,
```

```
## #   macro23 <chr>, macro24 <chr>, macro25 <chr>, macro26 <chr>, macro27 <chr>,
## #   macro28 <chr>, macro29 <chr>, macro30 <chr>, macro31 <chr>, macro32 <chr>,
## #   macro33 <chr>, macro34 <chr>, macro35 <chr>, macro36 <chr>, macro37 <chr>,
## #   macro38 <chr>, macro39 <chr>, macro40 <chr>, macro41 <chr>, ...
```

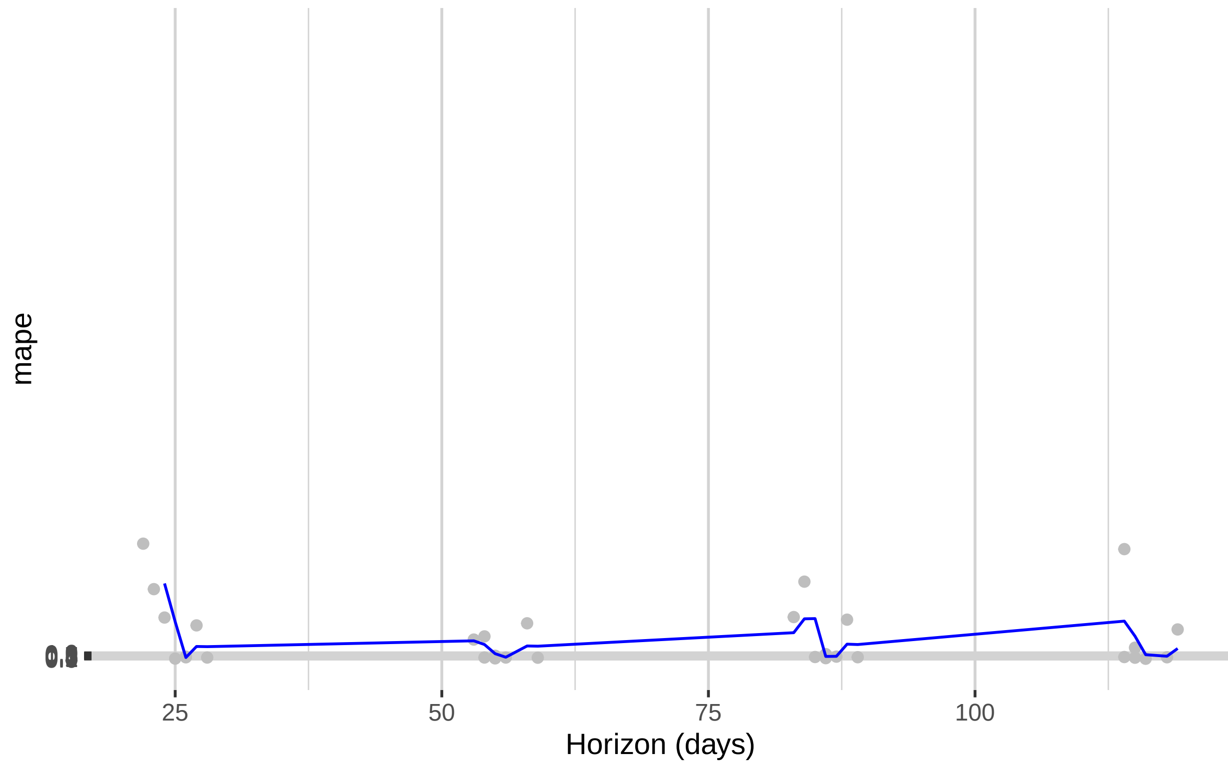
```
#predict the future
forecast <- predict(model, prediction_macro)
par(bg="white")

plot(model, forecast,panels = NULL,xlab='months',ylab='accounts charged off')+
  theme(
  panel.background = element_rect(fill = "white",
                              size = 2, linetype = "solid"),
  panel.grid.major = element_line(size = 0.5, linetype = 'solid',
                              colour = "lightgrey"),
  panel.grid.minor = element_line(size = 0.25, linetype = 'solid',
                              colour = "lightgrey")
  )
```




```
prophet_plot_components(model,forecast)
```




```r
#cross-validation to test model's accuracy
df.cv <- cross_validation(model, initial=180, period=60, horizon=120, units='days')
df.cv <- slice(df.cv, 1:(nrow(df.cv)-1))

#plotting to visualize the accuracy
plot_cross_validation_metric(df.cv, metric = 'mape') +
  scale_y_continuous(breaks = seq(0, 1, by = 0.1), limits = c(0, 100))+
  theme(
  panel.background = element_rect(fill = "white",
                              size = 2, linetype = "solid"),
  panel.grid.major = element_line(size = 0.5, linetype = 'solid',
                              colour = "lightgrey"),
  panel.grid.minor = element_line(size = 0.25, linetype = 'solid',
                              colour = "lightgrey")
  )
```



The final prediction for the period of 2020/02 - 2021/01

```
forecasting=forecast
forecasting$ds= format(forecasting$ds, "%Y-%m")
forecasting=forecasting%>%select(ds,yhat)
colnames(forecasting)=c('month','accounts_charged_off')
forecasting
```

```
## # A tibble: 12 x 2
##    month   accounts_charged_off
##    <chr>                  <dbl>
##  1 2020-02                1001.
##  2 2020-03                 591.
##  3 2020-04                 672.
##  4 2020-05                 794.
##  5 2020-06                1040.
##  6 2020-07                 868.
##  7 2020-08                 894.
##  8 2020-09                 924.
##  9 2020-10                 947.
## 10 2020-11                 899.
## 11 2020-12                 535.
## 12 2021-01                1645.
```

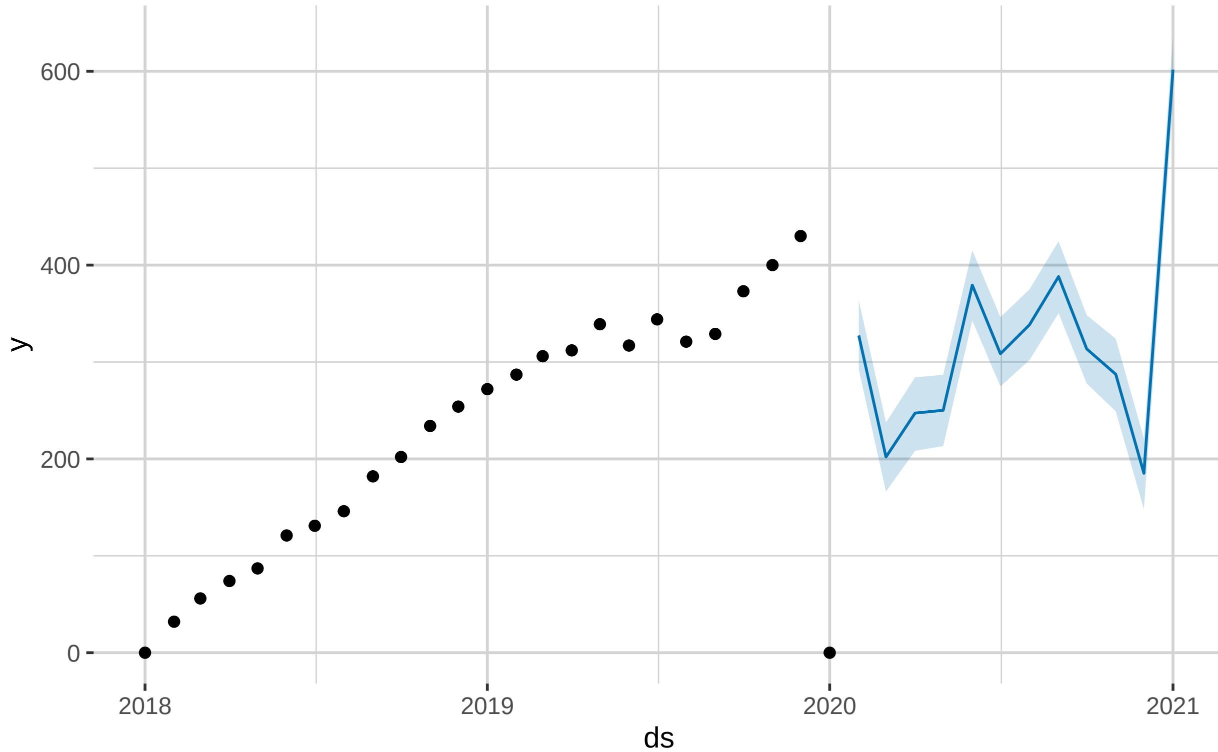
Make prediction based on industries and financial_active

function for prediction

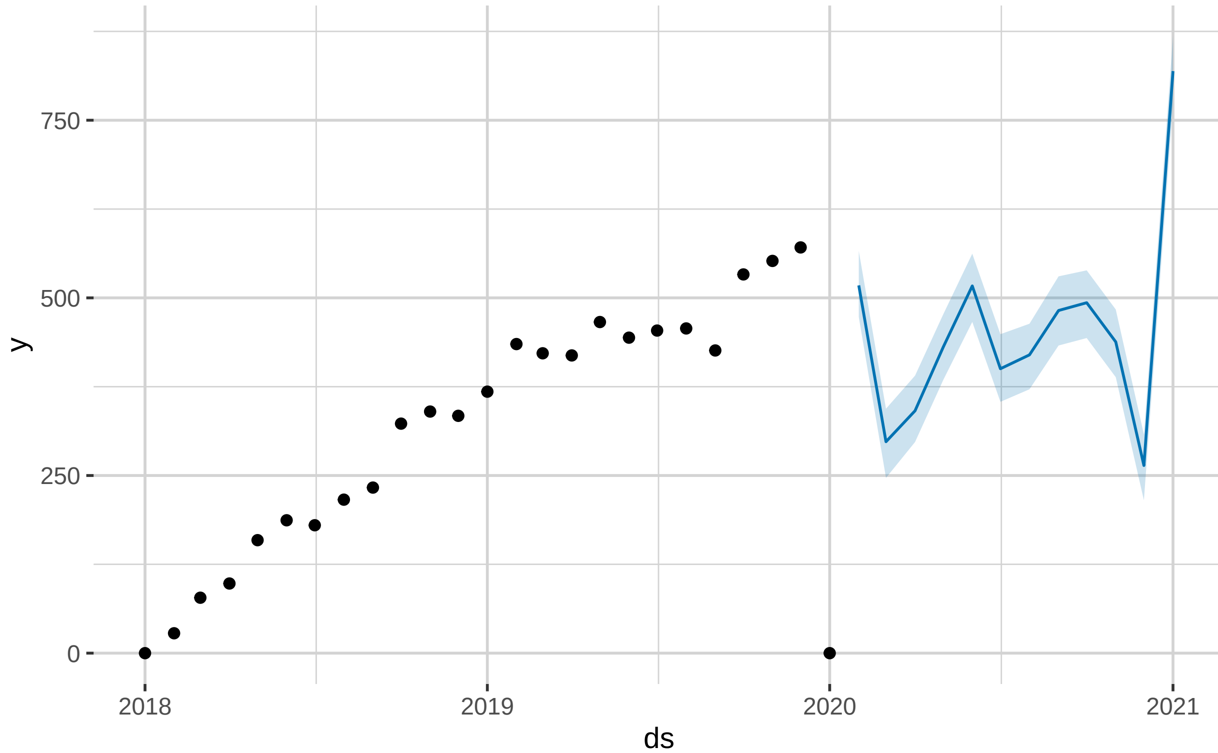
```r
#using fb prophet forecasting procedure to perform a Time Series forecasting
fb_prophet=function(training,prediction_macro){
  #idenity all regressors
  regressors <- training %>% select(-ds, -y)
  #fitting all regressors
  for (col in names(regressors)) {
    model <- prophet() %>% add_regressor(col, mode = "additive")
  }
  #fitting the model
  model <- fit.prophet(model, training)

  #tail
  tail(prediction_macro)

  #predict the future
  forecast <- predict(model, prediction_macro)
  plot(model, forecast,bg='white')+  theme(
  panel.background = element_rect(fill = "white",
                              size = 2, linetype = "solid"),
  panel.grid.major = element_line(size = 0.5, linetype = 'solid',
                              colour = "lightgrey"),
  panel.grid.minor = element_line(size = 0.25, linetype = 'solid',
                              colour = "lightgrey")
  )

}
```

Industry level

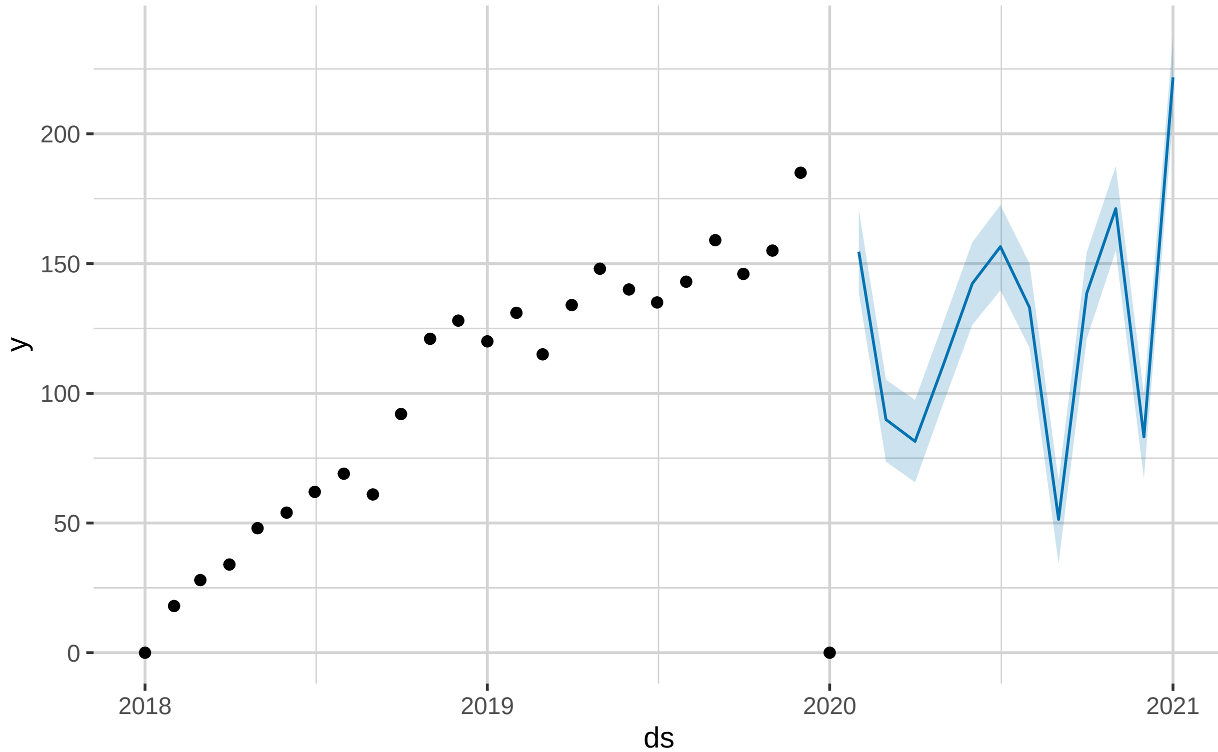
```r
training_data_indusA=df2[df2$industry=='A',]
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]
training_data_indusA=training_data_indusA%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
colnames(training_data_indusA)=c('ds','y',paste('macro',1:96,sep=''))
fb_prophet(training_data_indusA,prediction_macro=prediction_macro)
```




```
training_data_indusB=df2[df2$industry=='B',]
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]
training_data_indusB=training_data_indusB%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
colnames(training_data_indusB)=c('ds','y',paste('macro',1:96,sep=''))
fb_prophet(training_data_indusB,prediction_macro=prediction_macro)
```

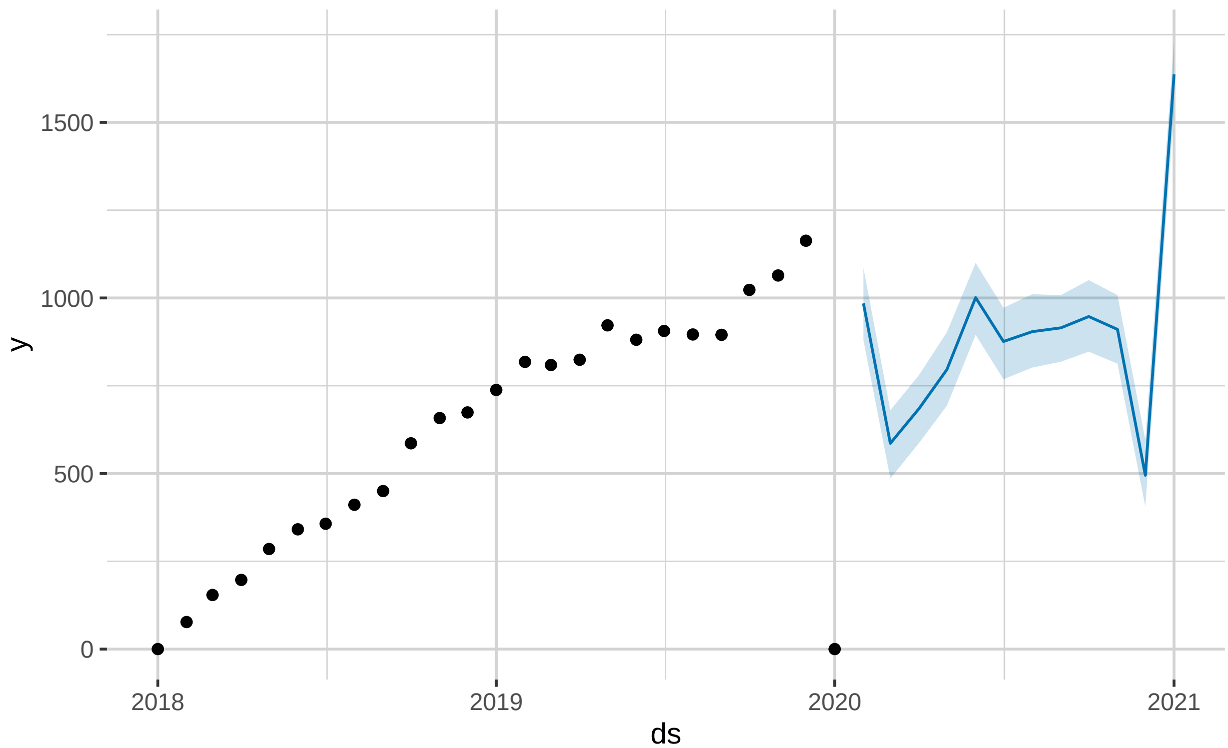



```
training_data_indusC=df2[df2$industry=='C',]
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]
training_data_indusC=training_data_indusC%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
colnames(training_data_indusC)=c('ds','y',paste('macro',1:96,sep=''))
fb_prophet(training_data_indusC,prediction_macro=prediction_macro)
```

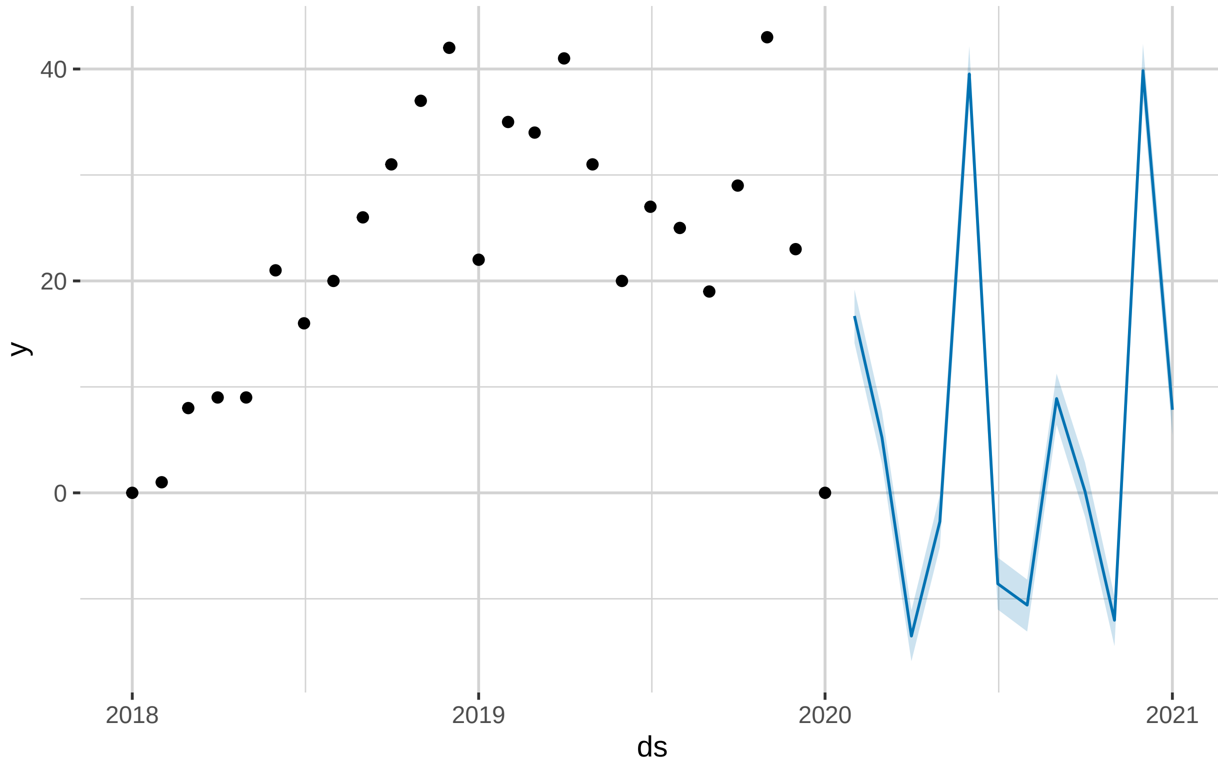


Financial active level

```
training_data_active=df2[df2$financial_active==1,]
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]
training_data_active=training_data_active%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
colnames(training_data_active)=c('ds','y',paste('macro',1:96,sep=''))
fb_prophet(training_data_active,prediction_macro=prediction_macro)
```




```
training_data_inactive=df2[df2$financial_active==0,]
filtered_macro_train=macro[macro$mth_code>='2018-01-01' & macro$mth_code<='2020-01-01', ]
training_data_inactive=training_data_inactive%>%
  group_by(mth_code)%>%
  summarize(sum_chargeoff=sum(charge_off))%>%
  ungroup()%>%
  left_join(filtered_macro_train,by='mth_code')
colnames(training_data_inactive)=c('ds','y',paste('macro',1:96,sep=''))
fb_prophet(training_data_inactive,prediction_macro=prediction_macro)
```


40
20
0
y
2018
2019
2020
2021
ds