💣

Support Vector Machines report

▼ Here are what the columns represent

- credit.policy: 1 if the customer meets the credit underwriting criteria of LendingClub.com, and 0 otherwise.
- purpose: The purpose of the loan (takes values "credit_card", "debt_consolidation", "educational", "major_purchase", "small_business", and "all_other").
- int.rate: The interest rate of the loan, as a proportion (a rate of 11% would be stored as 0.11). Borrowers judged by LendingClub.com to be more risky are assigned higher interest rates.
- installment: The monthly installments owed by the borrower if the loan is funded.
- log.annual.inc: The natural log of the self-reported annual income of the borrower.
- dti: The debt-to-income ratio of the borrower (amount of debt divided by annual income).
- fico: The FICO credit score of the borrower.
- days.with.cr.line: The number of days the borrower has had a credit line.
- revol.bal: The borrower's revolving balance (amount unpaid at the end of the credit card billing cycle).
- revol.util: The borrower's revolving line utilization rate (the amount of the credit line used relative to total credit available).
- inq.last.6mths: The borrower's number of inquiries by creditors in the last 6 months.
- delinq.2yrs: The number of times the borrower had been 30+ days past due on a payment in the past 2 years.
- pub.rec: The borrower's number of derogatory public records (bankruptcy filings, tax liens, or judgments)

Get Data

→ **Open the loan_data.csv file and save it as a dataframe called loans**

```
df <- read.csv("loan_data.csv")
head(df)
```

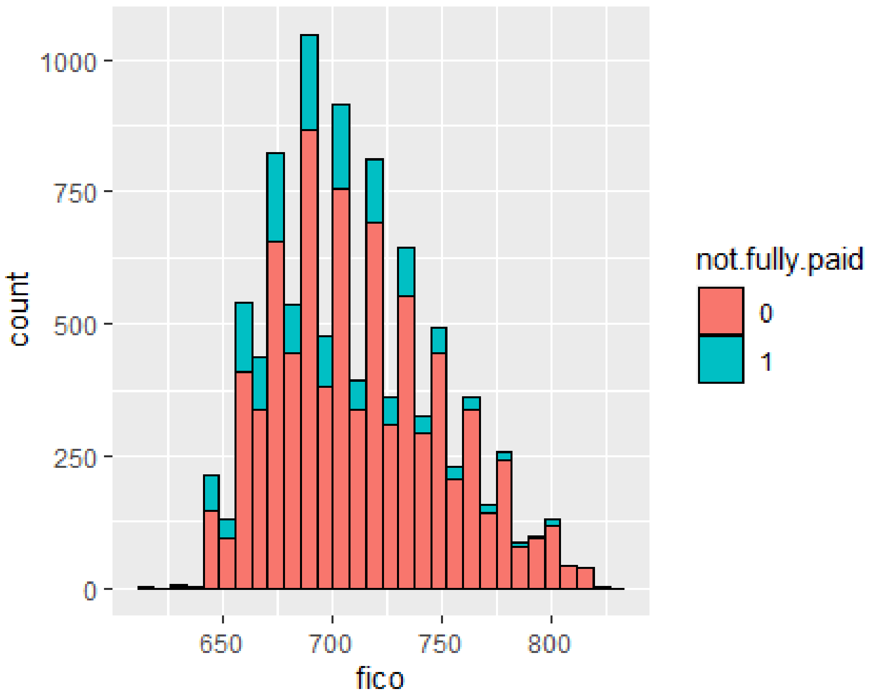
→ **Convert to categorical data using factor()**

```
# Convert class to factor
df$credit.policy <- factor(df$credit.policy)
df$inq.last.6mths <- factor(df$inq.last.6mths)
df$delinq.2yrs <- factor(df$delinq.2yrs)
df$not.fully.paid <- factor(df$not.fully.paid)
df$pub.rec <- factor(df$pub.rec)
```

EDA

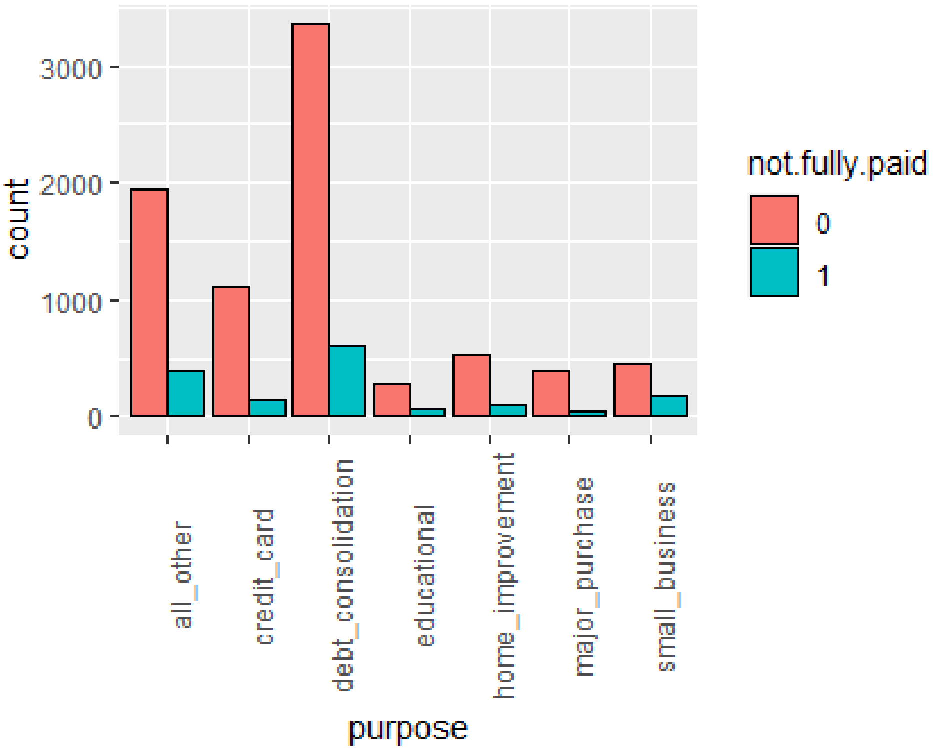
→ **Create a histogram of fico scores colored by not.fully.paid**

```
pl <- ggplot(df, aes(fico))
pl + geom_histogram(color = "black", aes(fill = not.fully.paid), position = position_stack(reverse = TRUE))
```



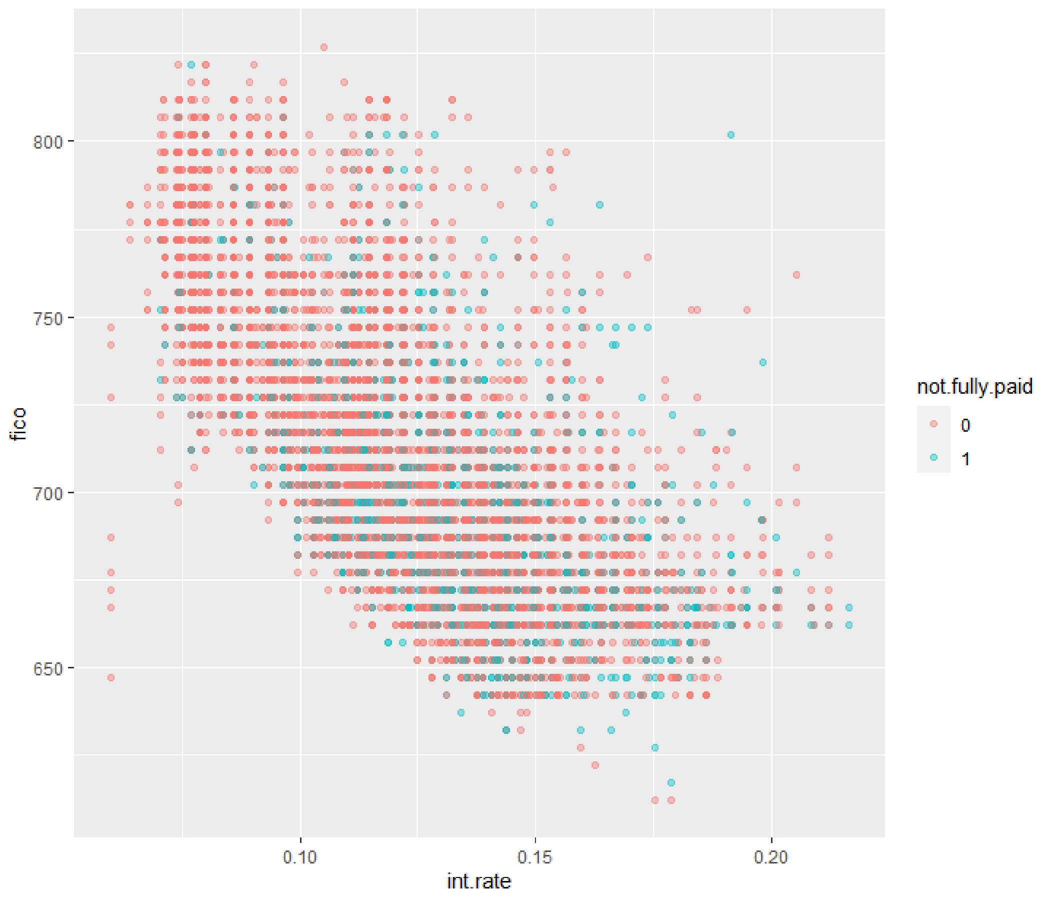
→ Create a barplot of purpose counts, colored by not.fully.paid. Use position=dodge in the geom_bar argument

```
pl2 <- ggplot(df, aes(purpose))
pl2 + geom_bar(color = "black", aes(fill = not.fully.paid), position = "dodge") + theme(axis.text.x = element_text(angle = 90))
```



→ **Create a scatterplot of fico score versus int.rate.**

```
pl3 <- ggplot(df, aes(y = fico, x = int.rate))
pl3 + geom_point(aes(color = not.fully.paid), alpha = 0.4)
```



💡 เราพบว่ายิ่ง อัตราดอกเบี้ยยิ่งต่ำ คะแนน FICO ของผู้กู้ก็จะยิ่งสูง ซึ่งสงผลต่อการจ่ายเงินให้ครบตามกำหนด ดังนั้นผู้กู้ที่มี คะแนน FICO ที่ต่ำ มีแนวโน้มที่จะค้างผู้ที่ค้างชำระเป็นส่วนมาก

Building the Model

▼ model predict → credit.policy

```
library(caTools)
library(e1071)

# Train and Test Sets
set.seed(24)
split <- sample.split(df$credit.policy, SplitRatio = 0.70)
train = subset(df, split == TRUE)
test = subset(df, split == FALSE)

# Building the Model
model <- svm(credit.policy ~ ., data=train)
summary(model)

# predict
p <- predict(model,test)
table(p,test$credit.policy)
```

Tuning the Model

Use the tune() function to test out different cost and gamma values.

```
tune.results <- tune(svm,train.x=not.fully.paid~., data=train,kernel='radial',ranges=list(cost=some.vector, gamma=some.other.vector))
```

```
# Tuning the Model
tune.results <- tune(svm,train.x=credit.policy ~.,data=train,kernel='radial',ranges=list(cost=c(1,10), gamma=c(0.1,1)))

# Best performance
b.model <- svm(credit.policy ~ .,data=train,cost=10,gamma = 0.1)
b.p <- predict(b.model,test[,-1])
table(b.p,test$credit.policy)
```

▼ model predict → not.fully.paid

```
library(caTools)
library(e1071)

# Train and Test Sets
set.seed(24)
split <- sample.split(df$not.fully.paid, SplitRatio = 0.70)
train = subset(df, split == TRUE)
test = subset(df, split == FALSE)

# Building the Model
model <- svm(not.fully.paid ~ ., data=train)
summary(model)

# predict
p <- predict(model,test[,-14])
table(p,test$not.fully.paid)


# Tuning the Model
tune.results <- tune(svm,train.x=not.fully.paid ~.,data=train,kernel='radial',ranges=list(cost=c(1,10), gamma=c(0.1,1)))

# Best performance
b.model <- svm(not.fully.paid ~ .,data=train,cost=10,gamma = 0.1)
b.p <- predict(b.model,test[,-14])
table(b.p,test$not.fully.paid)
```