Predicting if a Dwelling is Occupied by Owners or Renters

5/1/23

Ava Delanty

Abstract

This project investigates factors associated with variables regarding people and their housing situations, using a selection of data from Washington State which was imported by the US Census and was accessed through IPUMS USA. The study uses support vector models for classification to predict based if a dwelling is occupied by owners or renters on factors about education, age, marital status, number of rooms, and number of families. Results indicated that marital status, education, number of rooms, and number of families were the strongest predictors. The study emphasizes the importance of considering people and housing factors when studying renting and owning properties in Washington.

Introduction

Based on people and housing factors it can help predict whether dwellings are occupied by owners or renters. In this report, we will be specifically looking at factors about education, age, marital status, the number of rooms, and the number of families. We will use the method support vector machines to predict and classify. There will be three models that are linear, radial, and polynomial. The goal is to perform accurate machine learning models and predictions using support vector machines to select which response variables within the data from Washington State predict if a dwelling is occupied by owners or renters. In order to conduct concise modeling, only five variables will be used with some binary encoded. This includes marital status and age. The importance of these findings will help understand housing, renting, and ownership within Washington state.

Theoretical Background

Support Vector Classifiers:

Throughout this lab, we will be using a Support Vector Classifier which is a type of supervised learning algorithm used for classification tasks. SVC will separate the different classes by creating a hyperplane. The hyperplane is chosen that maximizes the margin which is the distance between the hyperplane and the support vectors. Support vectors are the closest points to the hyperplane. Additionally, the SVC algorithm uses a kernel function that transforms the data into a new space making it easier to classify. The kernel functions that will be used in the study are linear, radial, and polynomial. Depending on the kernel function used it can significantly affect the performance overall.

Tuning parameters in SVM:

A key aspect of support vector machines is tuning their parameters which affect the overall performance. There are the cost and gamma parameters. The cost parameter controls the trade-off between achieving a low training and testing error, determining the width of the margin of the hyperplane. Gamma controls the flexibility of the decision boundary. Additionally in the e1071 package in R, there is the tune function. This function takes training data, the type of SVM

model, and the range of values for the parameters. It evaluates the performance using cross-validation. The function overall will return the best values for the parameters which can be used to train a final model on the entire training set. It's overall efficient and convenient to use these tuning parameters and functions which will be shown in the results of this report.

ROC Curves:

Receiver Operating Characteristic (ROC) curve shows the performance of a binary classification model. The ROC curve plots the true positive against the false positive rate for the model's predicted probabilities. It is a way to visualize the sensitivity of the model. It was not done in this lab but a concept we used in class.

Methodology

Data Processing/Cleaning

Using the subset data frame from Housing.rdata that was provided on Canvas I used the household identification number to group the data and to only look at the max-age to determine the owner of each dwelling. I then pulled the variables OWNERSHP, ROOMS, EDUC, AGE, MARST, and NFAMS to use for modeling. The OWNERSHP variable indicates whether the housing unit was rented or owned by its inhabitants. ROOMS was the number of units per unit, EDUC was the highest education of homeowners (only looking at high school and college), AGE was their age (only looking at 25+), MARST was their marital status (only looking at divorced and married), and NFAMS which was the number of families within each unit. Additionally, marital status, age, and education were binary encoded into different columns grouped by married, divorced, college education, high school education, working-age adults (25-44), older working adults (45-64), and seniors(65+).

Models Created

Linear SVM

Using the kernel linear we looked at variables married, college educated, number of rooms, and number of families. A training test was created looking at 70% of the data while the test set was only 30%. Tuning was then used to find the best cost parameter, and a plot was created with the two strongest predictors.

Radial SVM

Using the kernel radial we looked at variables ages 25+ grouped into three columns binary encoded as prime working-age adults (25-44), older working adults (45-64), seniors(65+), the number of rooms, and the number of families. A training test was created looking at 70% of the data while the test set was only 30%. Tuning was then used to find the best cost and gamma parameters, and a plot was created with the two strongest predictors.

Polynomial SVM

Using the kernel radial we looked at variables married, divorced, college education, high school education, the number of rooms, and the number of families per unit. A training test was created looking at 70% of the data while the test set was only 30%. Tuning was then used to find the best cost and degree parameters, and a plot was created with the two strongest predictors.

Results

Linear Model:

Results gave a training error of .79 and a test error of 1.13 for both the SVM model and tuning model where the cost parameter was .01. Married and the number of rooms were the strongest predictors for whether the dwelling was owned or rented. A plot was created that includes the hyperplane and support vectors as shown below.

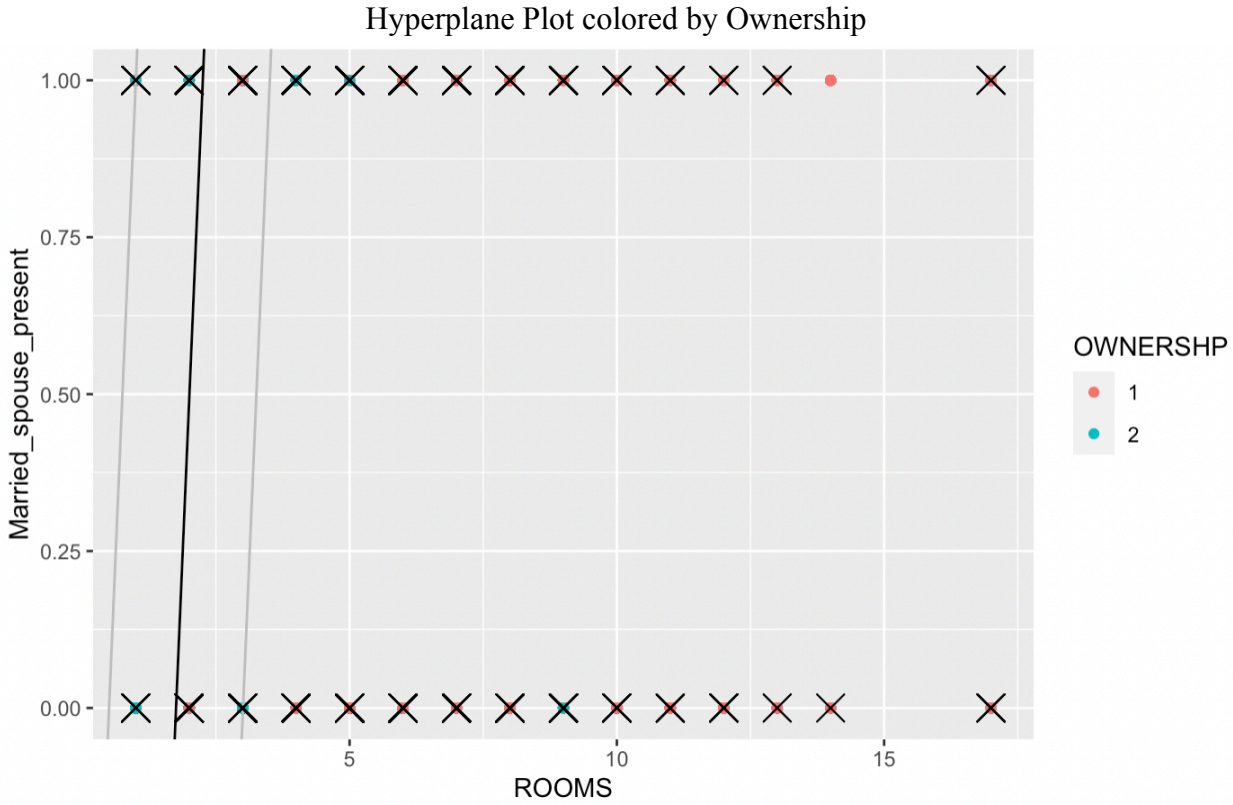

Figure 1: Hyperplane Plot with Married and the Number of Rooms per unit. Ownership key is 1: Owner and 2: Renter

Radial Model:

Results gave a training error of .80 and a test error of 1.14 for both the SVM model and tuning model where the cost parameter was .01 and gamma was 1 . Ages 25 to 44 and the

number of rooms was the strongest predictors for whether the dwelling was owned or rented. A plot was created showing the SVM classification plot.

Polynomial Model:

Results gave a training error of .79 and an accuracy of .55 and a test error accuracy of 1.13 for both the SVM model and tuning model where the cost parameter was .1 and degree was 1 . The number of rooms and the number of families per unit were the strongest predictors for whether the dwelling was owned or rented. A plot was created showing the SVM classification plot as shown below.

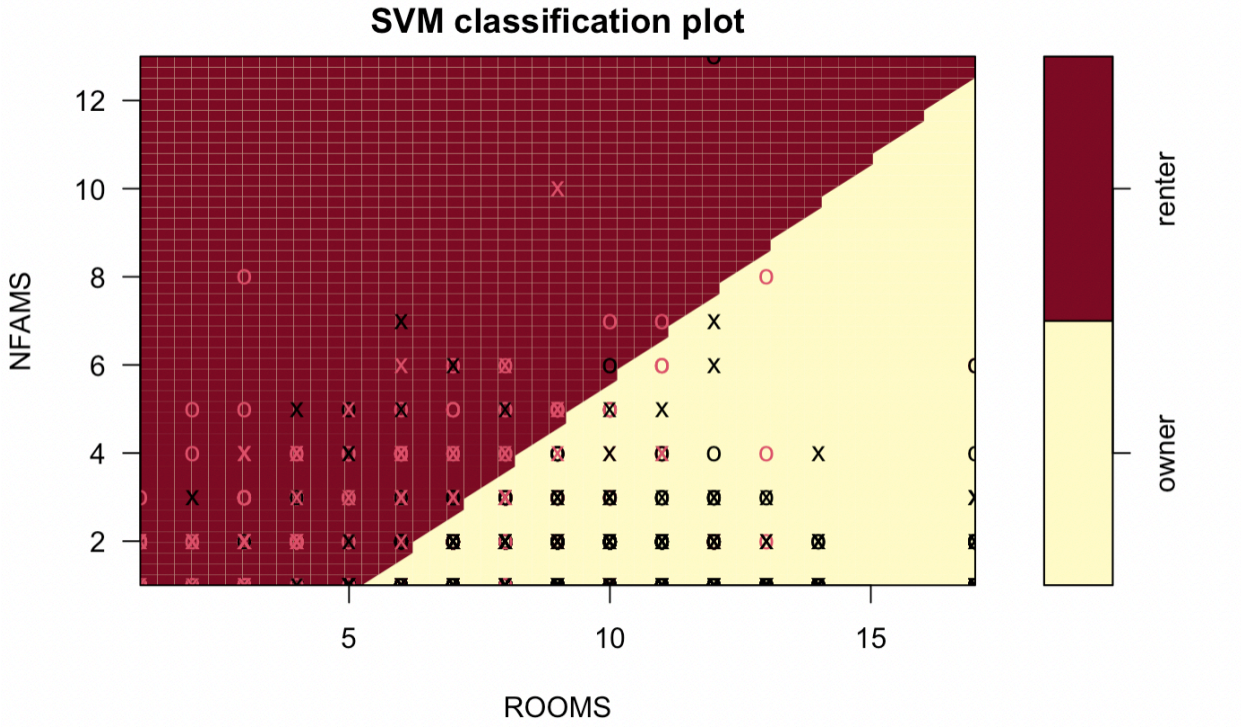

Figure 2: SVM Classification plot with variables number of families and number of rooms per unit

Discussion

From Figure 2 we can see that the SVM classification plot is trained on a training set on two separate data points into two classes: "owner" and "renter". The plot shows the decision boundary that the SVM learned to separate the two classes based on the input features "number of rooms per unit" (x-axis) and "number of families per unit" (y-axis). The hyperplane separates the two classes and is split diagonally. This means that the points represent a unit with a certain number of rooms and families living in it. By looking at the plot we can see that there is a nice split and that more rooms are classified into owner whereas more number of families are

classified into the renter class. This could indicate that more families mean more of a chance it is rented. It should also be noted that in the radial and linear models, the best predictors were also the number of rooms which could indicate based on this set of variables used that rooms play a significant factor in whether a dwelling is owned or rented.

Conclusion

The results from this study indicate that marital status, education, number of rooms, and number of families were the strongest predictors for determining if a dwelling is occupied by owners or renters. The three models used in this study indicated that the number of rooms per unit was significant as shown in the plots provided. Since more rooms are occupied by owners, it's most likely due to financial reasons and wanting to rent out those rooms. Going forward looking at a study on income could provide more interesting results. Based on this information and due to the challenges around homeownership in the state of Washington, creating new policies by providing low-income housing with more rooms based on the factors analyzed would be beneficial.

References

Steven Ruggles, Sarah Flood, Matthew Sobek, Danika Brockman, Grace Cooper, Stephanie Richards, and Megan Schouweiler. IPUMS USA: Version 13.0 [dataset]. Minneapolis, MN: IPUMS, 2023. https://doi.org/10.18128/D010.V13.0

Housing.rdata from Canvas

Appendix

NOTE: Due to how long my tuning models were taking I did not have time to knit a HTML file. I have organized the code as best as I could.

Libraries Used:

```
library(dplyr)
library(ggplot2)
library(haven)
library(e1071)
```

Data Cleaning Process:

Loading in the dataset and aggregating:

```
load("~/DATA 5322/Housing.rdata")
```

 # Create a new data frame with only the relevant columns

```
data <- data %>% select(SERIAL, OWNERSHP, ROOMS, EDUC,AGE, MARST,NFAMS)
```

 # Group the data by household and select the oldest member in each

```
data <- data %>%
  group_by(SERIAL) %>%
  slice(which.max(AGE)) %>%
  ungroup()
```

Removing OWNERSHP labels:

```
library(haven)
data$OWNERSHP<- zap_labels(data$OWNERSHP)
unique(data$OWNERSHP)
table(data$OWNERSHP)
```

Binary encoding:

```
# Create binary indicator variables for MARST
data$Married_spouse_present <- ifelse(data$MARST == 1, 1, 0)
data$Married_spouse_absent <- ifelse(data$MARST == 2, 1, 0)
data$Separated <- ifelse(data$MARST == 3, 1, 0)
```

```
data$Divorced <- ifelse(data$MARST == 4, 1, 0)
data$Widowed <- ifelse(data$MARST == 5, 1, 0)
data$Never_married_single <- ifelse(data$MARST == 6, 1, 0)

#age binary indicator variables for AGE:
data$children_teenagers <- as.numeric(data$AGE >= 0 & data$AGE <= 17,1,0)
data$young_adults <- as.numeric(data$AGE >= 18 & data$AGE <= 24,1,0)
data$prime_working_age_adults <- as.numeric(data$AGE >= 25 & data$AGE <= 44,1,0)
data$older_working_age_adults <- as.numeric(data$AGE >= 45 & data$AGE <= 64,1,0)
data$seniors <- as.numeric(data$AGE >= 65,1,0)

#create binary indicator variables for NFAMS
data$vacant_unit <- ifelse(data$NFAMS == 0,1,0)
data$one_family <- ifelse(data$NFAMS == 1,1,0)
data$two_family <- ifelse(data$NFAMS == 2,1,0)

#create binary indicator variables for EDUC
data$edu_noschooling <- ifelse(data$EDUC == 0, 1, 0)
data$edu_primary <- ifelse(data$EDUC == 1 | data$EDUC == 2, 1, 0)
data$edu_highschool <- ifelse(data$EDUC >= 3 & data$EDUC <= 6, 1, 0)
data$edu_college <- ifelse(data$EDUC >= 7, 1, 0)
```

#Changing 1 and 2 to owner and renter

```
data$OWNERSHP <- ifelse(data$OWNERSHP == 1, "owner", ifelse(data$OWNERSHP == 2,
"renter", data$OWNERSHP))
data$OWNERSHP <- as.factor(data$OWNERSHP)
data
```

Modeling:

Linear Model:

```
linear_data2 <-data %>%
select(OWNERSHP,Married_spouse_present,edu_college,NFAMS,ROOMS)
linear_data2$OWNERSHP <- as.factor(linear_data2$OWNERSHP)
```

Training and testing data:

```
set.seed(1)
```

#use 70% of dataset as training set and 30% as test set

```
train <- sample(c(TRUE, FALSE), nrow(linear_data2), replace=TRUE, prob=c(0.7,0.3))
L.train  <- linear_data2[train, ]
L.test   <- linear_data2[-train, ]
```

SVM Model:

```
model <- svm(OWNERSHP ~ .-OWNERSHP, data = L.train, kernel = "linear",type =
"C-classification",cost = .1, scale = FALSE)
summary(model)
```

Training error and testing error:

```
#training error rates with matrix table
ypred <- predict(model, L.train)
table1 <- table(predict = ypred, truth = L.train$OWNERSHP)
table1
#test error rates with matrix table
ypred <- predict(model, L.test)
table2 <- table(predict = ypred, truth = L.test$OWNERSHP)
table2
(13947+3276)/21561.4
(19856+4661)/21561.4
accuracy_Test <- sum(diag(table1)) / sum(table1)
print(paste('Accuracy for test', accuracy_Test))
accuracy_Test2 <- sum(diag(table2)) / sum(table2)
print(paste('Accuracy for test', accuracy_Test2))
```

Tuning:

```
tune.out <- tune(svm,OWNERSHP ~ .-OWNERSHP, data = L.train, kernel = "linear",scale =
FALSE,
   ranges = list(cost = c(0.01, .1)))
```

```
summary(tune.out)
bestmod = tune.out$best.model
#training error
ypred <- predict(bestmod,L.train)
table3<- table(predict = ypred, truth = L.train$OWNERSHP)
#test error
ypred <- predict(bestmod, L.test)
table4 <- table(predict = ypred, truth = L.test$OWNERSHP)
accuracy_Test <- sum(diag(table3)) / sum(table3)
print(paste('Accuracy for test', accuracy_Test))
accuracy_Test2 <- sum(diag(table4)) / sum(table4)
print(paste('Accuracy for test', accuracy_Test2))
```

Plotting:

```
plot(bestmod,L.train,Married_spouse_present~ROOMS)
w <- t(bestmod$coefs) %*% bestmod$SV
w
```

Figure 1 Plot:

```
# pick the x and y coefficients
bx = w[1] # married
by = w[3] #rooms
b0 = bestmod$rho # equivalent to beta_0 in e1071 implementation
base = ggplot(data=L.train, mapping= aes(x = ROOMS, y =  Married_spouse_present))
base + geom_point(aes(color=OWNERSHP)) +
  geom_abline(slope=-bx/by, intercept=b0/by, color='black') +
  geom_abline(slope=-bx/by, intercept=(b0+1)/by, color='gray') +
  geom_abline(slope=-bx/by, intercept=(b0-1)/by, color='gray') +
  geom_point(data=data.frame(bestmod$SV), mapping = aes(x = ROOMS, y =
Married_spouse_present), shape=4, size=5)
```

Radial Model:

```
radial_data <-data %>%
select(OWNERSHP,prime_working_age_adults,older_working_age_adults,seniors,ROOMS,NFAMS)
```

Testing and Training data:

```
set.seed(1)
#use 70% of dataset as training set and 30% as test set
train <- sample(c(TRUE, FALSE), nrow(radial_data), replace=TRUE, prob=c(0.7,0.3))
R.train  <- radial_data[train, ]
R.test   <- radial_data[-train, ]
```

SVM Model:

```
model2 <- svm(OWNERSHP ~ .-OWNERSHP, data = R.train, kernel = "radial",type = "C-classification",cost = .01,gamma = 1, scale = FALSE)
summary(model2)
```

#training error rates with matrix table

```
ypred <- predict(model2, R.train)
table1 <- table(predict = ypred, truth = R.train$OWNERSHP)
table1
#test error rates with matrix table
ypred <- predict(model2, R.test)
table2 <- table(predict = ypred, truth = R.test$OWNERSHP)
table2
(14270 + 3027)/21561.4
(20351+4302)/21561.4
accuracy_Test <- sum(diag(table1)) / sum(table1)
print(paste('Accuracy for test', accuracy_Test))
accuracy_Test2 <- sum(diag(table2)) / sum(table2)
print(paste('Accuracy for test', accuracy_Test2))
```

Tuning:

```
set.seed(1)
tune.out <- tune(svm,OWNERSHP ~ .-OWNERSHP, data = R.train, kernel = "radial",
   ranges = list(cost = c(0.01, 0.1),
```

```
            gamma = c(1, 2, 3)))
summary(tune.out)
```

Plot:

```
plot(model2,R.train,ROOMS~prime_working_age_adults)
```

Polynomial Model:

```
poly_data <-data %>%
select(OWNERSHP,ROOMS,NFAMS,Married_spouse_present,Divorced,edu_college,edu_highschool)
```

Training and testing:

```
set.seed(1)
#use 70% of dataset as training set and 30% as test set
train <- sample(c(TRUE, FALSE), nrow(poly_data), replace=TRUE, prob=c(0.7,0.3))
P.train  <- poly_data[train, ]
P.test   <- poly_data[-train, ]
```

SVM model:

```
svmpoly <- svm(OWNERSHP ~ .-OWNERSHP, data =P.train , type='C-classification',
            kernel = "polynomial", degree=1, coef0 = 1, cost = 1)
summary(svmpoly)
```

Tuning:

```
set.seed(1)
tune.out <- tune(svm, OWNERSHP ~ .-OWNERSHP, data = P.train,
   kernel = "polynomial", coef0 = 1,
   ranges = list(
     degree =c(1,2,3),
     cost = c(.01, .1)))
summary(tune.out)
bestmod2 = tune.out$best.model
```

Training and testing error:

```
#training error
ypred <- predict(bestmod2,P.train)
```

```
table5<- table(predict = ypred, truth = P.train$OWNERSHP)
table5
#test error
ypred <- predict(bestmod2, P.test)
table6 <- table(predict = ypred, truth = P.test$OWNERSHP)
table6
(14221+2983)/21561.4
(20286+4256)/21561.4
accuracy_Test <- sum(diag(table5)) / sum(table6)
print(paste('Accuracy for test', accuracy_Test))
accuracy_Test2 <- sum(diag(table5)) / sum(table6)
print(paste('Accuracy for test', accuracy_Test2))
```

Figure 2 Plot:

```
plot(bestmod2,poly_data,NFAMS~ROOMS)
```