HW 9

Marion Geary

3/17/2022

```r
library(mlbench)
library(readxl)
library(janitor)
library(neuralnet)
library(scales)
library(tidymodels)
tidymodels_prefer()

mycores <- parallel::detectCores(logical = FALSE)
library(doMC)
registerDoMC(cores = mycores)

#setwd("/Users/Marion/Desktop/math386/hw/hw-9/")
load("HW_8.Rdata")
load("HW_9.Rdata")
```

Create the Table

```r
nn_final_metrics <- my_metrics(
    nn_final_test,
    truth = reading,
    estimate = .pred
  )

elastic_final_metrics <- my_metrics(
    elastic_net_final,
    truth = reading,
    estimate = .pred
  )

table <- tibble(
    Method = c("Elastic Net", "Elastic Net", "Neural Net", "Neural Net"),
    Metric = c("RMSE", "$R^{2}$", "RMSE", "$R^{2}$"),
    Value = c(
      elastic_final_metrics$.estimate[1],
      elastic_final_metrics$.estimate[2],
      nn_final_metrics$.estimate[1],
      nn_final_metrics$.estimate[2]
    )
  )
```

Support Vector Machine: Linear

```r
set.seed(24)
# use same splits and recipe :)
# tune C and \epsilon
svm_lin_mod <- svm_linear(cost = tune(), margin = tune()) %>%
  set_engine("kernlab") %>%
  set_mode("regression")

svm_lin_wkflow <- workflow() %>%
  add_recipe(ozone_recipe) %>%
  add_model(svm_lin_mod)

svm_lin_tuning_grid <- grid_max_entropy(parameters(svm_lin_mod), size = 100)

svm_lin_tuned <- svm_lin_wkflow %>%
  tune_grid(resamples = ozone_folds, grid = svm_lin_tuning_grid)

svm_lin_final_fit <- finalize_workflow(
    svm_lin_wkflow,
    select_best(svm_lin_tuned, "rmse")
  ) %>% fit(ozone_train)

svm_lin_final_test <- augment(
    svm_lin_final_fit,
    new_data = ozone_test
  )
svm_lin_metrics <- my_metrics(
    svm_lin_final_test,
    truth = reading,
    estimate = .pred
  )
```

Support Vector Machine: Polynomial

```r
set.seed(24)

svm_poly_mod <- svm_poly(
    cost = tune(),
    margin = tune(),
    scale_factor = tune(),
    degree = tune()
  ) %>%
  set_engine("kernlab") %>%
  set_mode("regression")

svm_poly_wkflow <- workflow() %>%
  add_recipe(ozone_recipe) %>%
  add_model(svm_poly_mod)

svm_poly_tuning_grid <- grid_max_entropy(parameters(svm_poly_mod), size = 100)
```

```r
svm_poly_tuned <- svm_poly_wkflow %>%
  tune_grid(grid = svm_poly_tuning_grid, resamples = ozone_folds)

svm_poly_final_fit <- finalize_workflow(
    svm_poly_wkflow,
    select_best(svm_poly_tuned, "rmse")
  ) %>% fit(data = ozone_train)

svm_poly_final_res <- augment(svm_poly_final_fit, new_data = ozone_test)

svm_poly_metrics <- my_metrics(
    svm_poly_final_res,
    truth = reading,
    estimate = .pred
  )
```

Support Vector Machine: Radial

```r
set.seed(24)
svm_rad_mod <- svm_rbf(cost = tune(), margin = tune(), rbf_sigma = tune()) %>%
  set_engine("kernlab") %>%
  set_mode("regression")

svm_rad_wkflow <- workflow() %>%
  add_recipe(ozone_recipe) %>%
  add_model(svm_rad_mod)

svm_rad_tuning_grid <- grid_max_entropy(parameters(svm_rad_mod), size = 100)

svm_rad_tuned <- svm_rad_wkflow %>%
  tune_grid(grid = svm_rad_tuning_grid, resamples = ozone_folds)

svm_rad_final_fit <- finalize_workflow(
    svm_rad_wkflow,
    select_best(svm_rad_tuned, "rmse")
  ) %>% fit(data = ozone_train)

svm_rad_final_res <- augment(svm_rad_final_fit, new_data = ozone_test)

svm_rad_metrics <- my_metrics(
    svm_rad_final_res,
    truth = reading,
    estimate = .pred
  )
```

Support Vector Machine Final Data Table

```r
table <- table %>%
  add_row(
    Method = "Linear SVM",
    Metric = "RMSE",
    Value = svm_lin_metrics$.estimate[1]
  ) %>%
  add_row(
    Method = "Linear SVM",
    Metric = "$R^{2}$",
    Value = svm_lin_metrics$.estimate[2]
  ) %>%
  add_row(
    Method = "Polynomial SVM",
    Metric = "RMSE",
    Value = svm_poly_metrics$.estimate[1]
  ) %>%
  add_row(
    Method = "Polynomial SVM",
    Metric = "$R^{2}$",
    Value = svm_poly_metrics$.estimate[2]
  ) %>%
  add_row(
    Method = "Radial SVM",
    Metric = "RMSE",
    Value = svm_rad_metrics$.estimate[1]
  ) %>%
  add_row(
    Method = "Radial SVM",
    Metric = "$R^{2}$",
    Value = svm_rad_metrics$.estimate[2]
  )
```

K Nearest Neighbors Regression

```
set.seed(24)
knn_reg_mod <- nearest_neighbor(
    neighbors = tune(),
    dist_power = tune(),
    weight_func = tune()
  ) %>%
  set_engine("kknn") %>%
  set_mode("regression")

knn_reg_wkflow <- workflow() %>%
  add_recipe(ozone_recipe) %>%
  add_model(knn_reg_mod)

knn_reg_tuning_grid <- grid_max_entropy(parameters(knn_reg_mod), size = 100)

knn_reg_tuned <- knn_reg_wkflow %>%
  tune_grid(grid = knn_reg_tuning_grid, resamples = ozone_folds)

knn_reg_final_fit <- finalize_workflow(
    knn_reg_wkflow,
```

```r
    select_best(knn_reg_tuned, metric = 'rmse')
  ) %>% fit(data = ozone_train)

knn_reg_final_res <- augment(knn_reg_final_fit, new_data = ozone_test)

knn_reg_metrics <- my_metrics(
    knn_reg_final_res,
    truth = reading,
    estimate = .pred
  )
```

Final Results

```r
table <- table %>%
  add_row(
    Method = "KNN Regression",
    Metric = "RMSE",
    Value = knn_reg_metrics$.estimate[1]
  ) %>%
  add_row(
    Method = "KNN Regression",
    Metric = "$R^{2}$",
    Value = knn_reg_metrics$.estimate[2]
  )

if(knitr::is_html_output(knitr::kable(table, digits = 3, escape = TRUE)))
{ table } else {
  knitr::kable(table, digits = 3, escape = TRUE)
}
```

Method	Metric	Value
Elastic Net	RMSE	4.620
Elastic Net	R^2	0.568
Neural Net	RMSE	4.558
Neural Net	R^2	0.567
Linear SVM	RMSE	4.439
Linear SVM	R^2	0.575
Polynomial SVM	RMSE	4.142
Polynomial SVM	R^2	0.629
Radial SVM	RMSE	4.276
Radial SVM	R^2	0.626
KNN Regression	RMSE	5.327
KNN Regression	R^2	0.506

After using all 6 models to predict the `Daily maximum one-hour-average ozone reading` for the `Ozone` data, the best performer, both by RMSE and by R^2, is the polynomial support vector machine. It is followed closely by the radial support vector machine model, which has only slightly lower metrics. I believe that one reason that the polynomial support vector machine performed so well could be because the data is nonlinear. This would explain why the nonlinear support vector machines performed better than the elastic net and linear support vector machine models which are both linear models. I am not surprised that the K-nearest neighbors model performed the worst. While it is not an ineffective model, it cannot compete with the neural net, elastic net, and support vector machines for most data sets because these models are much more advanced.

Another factor in the performance of the models are the tuning parameters. The polynomial support vector machine has the most tuning parameters, so this allows the model to be fit very particularly to the data set.

All of the models have been tuned for the best RMSE. I suspect that if I had tuned for the best R^2, the radial support vector machine model might have been the best performer.