DS-6030 Homework Assignments

Peter Gedeck

2026-01-26

Table of contents

Introduction

Homework assignments for DS-6030 Machine Learning II

1 DS-6030 Homework Module 1

Module 1

> **Note**
>
> Assignments (1) and (2) cover theoretical aspects of the course. Assignment (3) requires you to explore the `ISLR2::Boston` dataset using graphs.
> Use *Tidyverse* packages for assignment (3).
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-1.qmd) and use it as a basis for your solution.

1. Classification vs Regression - Inference or Prediction? (3 points)

Explain whether each scenario is a classification or regression problem, and indicate whether we are most interested in inference or prediction. Finally, provide the number of data points, n, and the number of predictors, p.

(1.1) We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry and the CEO salary. We are interested in understanding which factors affect CEO salary.

(1.2) We are considering launching a new product and wish to know whether it will be a success or a failure. We collect data on 20 similar products that were previously launched. For each product we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables.

(1.3) We are interested in predicting the % change in the USD/Euro exchange rate in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of 2012. For each week we record the % change in the USD/Euro, the % change in the US market, the % change in the British market, and the % change in the German market.

2. Describe the differences between a parametric and a non-parametric statistical learning approach (2 points)

What are the advantages of a parametric approach to regression or classification (as opposed to a non-parametric approach)? What are its disadvantages?

(2.1) Advantages

(2.2) Disadvantages

3. Explore the dataset `ISLR2::Boston` (11 points)

This `Boston` dataset is often used as an example for regression problems. It contains the following variables:

- `crim`: per capita crime rate by town.
- `zn`: proportion of residential land zoned for lots over 25,000 sq.ft.
- `indus`: proportion of non-retail business acres per town.
- `chas`: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- `nox`: nitrogen oxides concentration (parts per 10 million).
- `rm`: average number of rooms per dwelling.

- `age`: proportion of owner-occupied units built prior to 1940.
- `dis`: weighted mean of distances to five Boston employment centres.
- `rad`: index of accessibility to radial highways.
- `tax`: full-value property-tax rate per $10,000.
- `ptratio`: pupil-teacher ratio by town.
- `lstat`: lower status of the population (percent).
- `medv`: median value of owner-occupied homes in $1000s.

(A) Create histograms and/or densityplots of each feature using `ggplot2`.

(3.1) Don't create individual graphs. Use `patchwork` to combine multiple graphs into a figure. (2 points - coding)

(3.2) Look for interesting patterns in the distributions. For example, are distributions highly skewed? Do you notice any outliers? Document your findings. (2 point - discussion)

(3.3) Are there any variables that should be transformed? (2 point - discussion)

(B) With the processed dataset, create three or more plots using `ggplot2` to explore the relationship between the variables. Document your findings.

(3.4) Create three or more scatterplots (2 points - coding)

(3.5) Do you see strong correlation between some of the variables? What type of correlation do you see? (2 points - discussion)

(3.6) What would be the consequence of these if you want to train a regression model? (1 points - discussion)

2 DS-6030 Homework Module 2

Module 2

> **Note**
>
> In this assignment you will build ordinary linear regression models.
> Use *Tidyverse* and *Tidymodels* packages for the assignments.
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-2.Rmd) and use it as a basis for your solution.

1. Flexible vs Inflexible Methods (2 points)

For each of parts (a) through (d), indicate whether we would generally expect the performance of a flexible statistical learning method to be better or worse than an inflexible method. Justify your answer.

(1.1) The sample size n is extremely large, and the number of predictors p is small.

(1.2) The number of predictors p is extremely large, and the number of observations n is small.

(1.3) The relationship between the predictors and response is highly non-linear.

(1.4) The variance of the error terms, i.e. $\sigma^2 = Var(\epsilon)$, is extremely high.

2. Predicting Airfare on New Routes

The following problem takes place in the United States in the late 1990s, when many major US cities were facing issues with airport congestion, partly as a result of the 1978 deregulation of airlines. Both fares and routes were freed from regulation, and low-fare carriers such as Southwest (SW) began competing on existing routes and starting nonstop service on routes that previously lacked it. Building completely new airports is generally not feasible, but sometimes decommissioned military bases or smaller municipal airports can be reconfigured as regional or larger commercial airports. There are numerous players and interests involved in the issue (airlines, city, state and federal authorities, civic groups, the military, airport operators), and an aviation consulting firm is seeking advisory contracts with these players. The firm needs predictive models to support its consulting service. One thing the firm might want to be able to predict is fares, in the event a new airport is brought into service. The firm starts with the dataset *Airfares.csv.gz*, which contains real data that were collected between Q3-1996 and Q2-1997. The variables in these data are listed in the following Table, and are believed to be important in predicting FARE. Some airport-to-airport data are available, but most data are at the city-to-city level. One question that will be of interest in the analysis is the effect that the presence or absence of Southwest has on FARE.

Variable	Description
S_CODE	Starting airport's code
S_CITY	Starting city
E_CODE	Ending airport's code
E_CITY	Ending city
COUPON	Average number of coupons (a one-coupon flight is a nonstop flight,a two-coupon flight is a one-stop flight, etc.) for that route
NEW	Number of new carriers entering that route between Q3-96 and Q2-97

Variable	Description
VACATION	Whether (Yes) or not (No) a vacation route
SW	Whether (Yes) or not (No) Southwest Airlines serves that route
HI	Herfindahl index: measure of market concentration
S_INCOME	Starting city's average personal income
E_INCOME	Ending city's average personal income
S_POP	Starting city's population
E_POP	Ending city's population
SLOT	Whether or not either endpoint airport is slot-controlled (this is a measure of airport congestion)
GATE	Whether or not either endpoint airport has gate constraints (this is another measure of airport congestion)
DISTANCE	Distance between two endpoint airports in miles
PAX	Number of passengers on that route during period of data collection
FARE	Average fare on that route

(A) Data Exploration:

(2.1) Load the data from `https://gedeck.github.io/DS-6030/datasets/homework/Airfares.csv.gz` and preprocess the data; convert categorical variables to factors. (1 point - coding)

(2.2) Explore the numerical (continuous) predictors and response (FARE) by creating a correlation table and examining some scatterplots between FARE and those predictors. What seems to be the best single predictor of FARE? (2 points - coding/discussion)

(2.3) Explore the categorical predictors (excluding the first four) by creating individual graphs comparing the distribution of average fare for each category (e.g. box plots). Which categorical predictor seems best for predicting FARE? (2 points - coding/discussion)

(B) Find a model for predicting the average fare on a new route:

(2.4) Partition the data into training and holdout sets. The model will be fit to the training data and evaluated on the holdout set. (see DS-6030: Creating an initial split of the data into training and holdout set) (1 point - coding)

(2.5) Train a linear regression model with *tidymodels* using all predictors. You can ignore the first four predictors (S_CODE, S_CITY, E_CODE, E_CITY). Examine the model coefficients and interpret them. Which predictors are significant? (see DS-6030: Linear regression models)
Determine the model performance using `r^2`, `RMSE` and `MAE` on the training and test set. How does the model perform on the test set? Is the model overfitting? How can you tell? (see DS-6030: Measuring performance of regression models) (2 points - coding/discussion)

(2.6) Taking the results from **(2.2)**, **(2.3)**, and **(2.5)** into account, build a model that includes only the most important predictors. Determine the model performance and compare with the full model from **(2.5)**. (2 points - coding/discussion)

(2.7) Using the models from **(2.5)** and **(2.6)**, predict the average fare on a route with the following characteristics: COUPON = 1.202, NEW = 3, VACATION = No, SW = No, HI = 4442.141, S_INCOME = $28,760, E_INCOME = $27,664, S_POP = 4,557,004, E_POP = 3,195,503, SLOT = Free, GATE = Free, PAX = 12,782, DISTANCE = 1976 miles. *Hint*: make sure that you treat the categorical variables in the same way as in the training data. (1 point - coding)

(2.8) Using the smaller model from **(2.6)**, predict the reduction in average fare on the route in **(2.7)** if Southwest decides to cover this route. (1 point - coding/discussion)

(C) Predictors

(2.9) In reality, which of the factors will not be available for predicting the average fare from a new airport (i.e., before flights start operating on those routes)? Which ones can be estimated? How? (1 point - discussion)

(2.10) Train a model that includes only factors that are available before flights begin to operate on the new route. (1 point - coding)

(2.11) Compare the predictive accuracy of this model with models from **(2.5)** and **(2.6)**. Is this model good enough, or is it worthwhile reevaluating the model once flights begin on the new route? (1 point - discussion)

3 DS-6030 Homework Module 3

Module 3

> **Note**
>
> In this module, we learned about classification models. Assignment (1) tests your understanding of the differences between LDA and QDA. In assignments (2), you will build several classification models for the NASA Asteroid dataset and estimate their predictive performance using a holdout/test set. Assignment (3) prepares you for the final project by researching approaches to handling class imbalance in classification problems.
> Use *Tidyverse* and *Tidymodels* packages for the assignments.
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-3.Rmd) and use it as a basis for your solution.

```
library(tidyverse)
library(tidymodels)
```

1. Differences between LDA and QDA. (4 points)

(1.1) If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?

(1.2) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?

(1.3) In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?

(1.4) True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.

2. NASA: Asteroid classification

The dataset `nasa.csv` contains information about asteroids and if they are considered to be hazardous or not.

(A) Data loading and preprocessing

(2.1) Load the data from https://gedeck.github.io/DS-6030/datasets/nasa.csv and preprocess the data. You can reuse the preprocessing pipeline we developed in class. (1 point - coding)

```
filename <- "https://gedeck.github.io/DS-6030/datasets/nasa.csv"
remove_columns <- c("Name", "Est Dia in M(min)",
  "Semi Major Axis", "Jupiter Tisserand Invariant",
  "Epoch Osculation", "Mean Motion", "Aphelion Dist",
  "Equinox", "Orbiting Body", "Orbit Determination Date",
  "Close Approach Date", "Epoch Date Close Approach",
  "Miss Dist.(Astronomical)", "Miles per hour")
asteroids <- read_csv(filename, show_col_types = FALSE) %>%
```

```
  select(-all_of(remove_columns)) %>%
  select(-contains("Relative Velocity")) %>%
  select(-contains("Est Dia in KM")) %>%
  select(-contains("Est Dia in Feet")) %>%
  select(-contains("Est Dia in Miles")) %>%
  select(-contains("Miss Dist.(lunar)")) %>%
  select(-contains("Miss Dist.(kilometers)")) %>%
  select(-contains("Miss Dist.(miles)")) %>%
  distinct() %>%
  mutate(Hazardous = as.factor(Hazardous))
dim(asteroids)
```

```
[1] 3692   15
```

(2.2) Split the dataset into a training and test set. Use 80% of the data for training and 20% for testing. Use stratified sampling to ensure that the training and test set have the same proportion of hazardous asteroids. (2 points - coding)

(B) Model training Build a classification model with *tidymodels* using four different methods: Null model, Logistic regression, LDA, and QDA.

(2.3) Use the training set to fit the four models. (2 point - coding)

(2.4) For each model, determine and plot the ROC curves for both the training and test set. What do you observe? Use `patchwork` to arrange the graphs for the four models in a single figure (1 point - coding/discussion)

(2.5) Create a single plot that overlays the ROC curves of the four models for the test set. Which model separates the two classes best? (1 point - coding/discussion)

(2.6) For each model, determine the threshold that maximizes the F-measure (`yardstick::f_meas`) using the training set. Why is the F-measure a better metric than accuracy in this case? Create plots that show the dependence of the F-measure from the threshold. (2 points - coding/discussion)

(2.7) Determine the accuracy, sensitivity, specificity, and F-measure for each model at the determined thresholds. Which model performs best? How does this compare to the result from the ROC curves? (1 point - coding/discussion)

3. Handling class imbalance in classification problems (4 points)

Write a short essay (about 1/2 page) on the topic of handling class imbalance in classification problems. Here are a few questions to guide your research:

- What is class imbalance, and why is it a problem in classification problems?
- What are common strategies to handle class imbalance?
- What are appropriate evaluation metrics for imbalanced classification problems?

Consult 2 or more sources for your research. You can use the following resource to get started:

- https://machinelearningmastery.com/what-is-imbalanced-classification/

Don't forget to reference your sources including the use of large language models like Chat-GPT.

4 DS-6030 Homework Module 4

Module 4

> **Note**
>
> This module introduced cross-validation and bootstrapping as methods to estimate model performance. The homework assignment will give you the opportunity to apply these methods to two datasets. In assignment (1), you will use cross-validation to compare logistic regression, LDA, and QDA classification models. Assignment (2) uses bootstrap to estimate confidence intervals for the mean absolute error and root mean squared error of a linear regression model.
> Use *Tidyverse* and *Tidymodels* packages for the assignments.
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-4.Rmd) and use it as a basis for your solution.
> As you will find out, the knitting times for this assignment will be longer than in previous homework. To speed up the knitting process, use caching and parallel processing. You can find more information about caching and about parallel processing in the course material.

1. Diabetes dataset

The Diabetes Health Indicators Dataset contains healthcare statistics and lifestyle survey information about people in general along with their diagnosis of diabetes. The 35 features consist of some demographics, lab test results, and answers to survey questions for each patient. The target variable for classification is whether a patient has diabetes, is pre-diabetic, or healthy. For this study, the target variable was converted to a binary variable with 1 for diabetes or pre-diabetic and 0 for healthy. Information about the dataset can be found here.

For this exercise use caching and parallel processing to speed up your computations.

(A) Data loading and preprocessing

(1.1) Load the diabetes dataset from https://gedeck.github.io/DS-6030/datasets/diabetes/diabetes_binary_5050split_health_indicators_BRFSS2015.csv.gz. Convert the `Diabetes_binary` to a factor with labels *Healthy* and *Diabetes*. Convert all other variables that only contain values of 0 and 1 to factors. (1 point - coding)

(1.2) Split the data into a training and test set using a 50/50 split. Use the `set.seed()` function to ensure reproducibility. (1 point - coding)

(B) Model training

(1.3) Build a logistic regression model to predict `Diabetes_binary` using all other variables as predictors. Use the training set to fit the model using 10-fold cross-validation. Report the cross-validation accuracy and ROC-AUC of the model. (see DS-6030: Model validation using cross-validation) (2 points - coding)

(1.4) Use the approach from **(1.3)** to build LDA and QDA models. Report the cross-validation accuracy and ROC-AUC of each model. (4 points - coding)

(C) Cross-validation and test set performance

(1.5) Create a plot that compares the ROC curves of the three models from **(1.3)** and (1.4). The ROC curve should be based on the predictions from cross-validation. (1 point - coding/discussion)

How do the models compare? Which model would you choose for prediction?

(1.6) After fitting the three models using the full training set, estimate the performance metrics on the test set. Report the accuracy and ROC-AUC of each model. How do the models compare? Do you see a difference to the cross-validation results? (1 point - coding/discussion)

2. Estimate model performance using bootstrap

(2.1) Use the `mtcars` dataset from DS-6030: The mtcars dataset to estimate the mean absolute error (MAE) and root mean squared error (RMSE) of the linear regression model for the prediction of `mpg` using bootstrap. Use 1000 bootstrap samples. Report the mean and standard deviation of the two metrics. (see DS-6030: Model validation using bootstrapping) (3 points - coding)

(2.2) Create a plot of the distribution of the performance metrics. Comment on the shape of the distribution. (1 point - coding/discussion)

(2.3) Use the performance metrics calculated for the bootstrap samples to estimate the 95% confidence interval for the mean absolute error (MAE) and root mean squared error (RMSE). Report the confidence intervals. (2 points - coding/discussion)

5 DS-6030 Homework Module 5

Module 5

> **Note**
>
> In module 5, we learned about tuning hyperparameters of models. In this homework, we are using L1/L2 penalties and dimensionality reduction using PCA and PLS to control the flexibility of our models and reduce the risk of overfitting. We will apply these techniques to build classification and regression models. Both assignments 1 and 2 use datasets from previous assignments. You can reuse the preprocessing steps from these assignments.
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-5.Rmd) and use it to answer the following questions.
> If not otherwise stated, use *Tidyverse* and *Tidymodels* for the assignments.
> As you will find out, the knitting times for the assignment will get longer as you add more code. To speed up the knitting process, use caching and parallel processing. You can find more information about caching here and about parallel processing here.

1. Build elasticnet model for predicting airfare prices (L1/L2 regularization)

The `Airfares.csv.gz` dataset was already used in problem 1 of module 2. In that assignment, we built a model to predict the price of an airline ticket `FARE` using a linear regression model. In this assignment, we will build a model to predict the price of an airline ticket `FARE` using a linear regression model with both L1 and L2 regularization (tuning parameters `mixture` and `penalty`).

Load the data from `https://gedeck.github.io/DS-6030/datasets/homework/Airfares.csv.gz`

(1.1) Load and preprocess the data. Reuse the preprocessing steps you developed in module 2. (1 point - coding)

(1.2) Split the data into a training (75%) and test set (25%). Prepare the resamples for 10-fold cross-validation using the training set. (1 point - coding)

(1.3) Define workflow and tuneable parameters. In the recipe, include a step to convert the categorical / nominal variables to dummy variables (`step_dummy(all_nominal_predictors())`) (1 point - coding)

(1.4) Tune the model with 10-fold cross-validation using Bayesian hyperparameter optimization. Make sure that your search space covers a suitable range of values. (see DS-6030: Bayesian Hyperparameter optimization) (2 point - coding)

(1.5) Train a final model using the best parameter set. (1 point - coding)

(1.6) Predict the `FARE` for the test set and calculate the performance metrics on the test set. (1 point - coding)

2. NASA: Asteroid classification - classification with dimensionality reduction

The dataset `nasa.csv` contains information about asteroids and if they are considered to be hazardous or not.

(A) Data loading and preprocessing

(2.1) Load the data from https://gedeck.github.io/DS-6030/datasets/nasa.csv and preprocess the data. You can find the necessary preprocessing steps in module 3. (1 point - coding)

(2.2) Split the dataset into a training and test set. Use 80% of the data for training and 20% for testing. Use stratified sampling to ensure that the training and test set have the same proportion of hazardous asteroids. (1 point - coding)

(B) Model building

(2.3) Build a logistic regression classification model using principal components (`step_pca`) as predictors. Use cross-validation to determine the optimal number of components. (see DS-6030: Specifying tunable parameters) (3 points - coding)

- Use `step_normalize` and `step_pca` to preprocess the data in a recipe.
- Use the `tune` function to find the best number of components (`num_comp`) in the range 1 to 14 using AUC as the selection criterium. Check **all** possible numbers of components from 1 to 14.
- Use the `autoplot` function on the cross-validation results to visualize the results. Describe your observations.
- Report the best number of components and the associated regression metrics.
- Using the best parameter set, train a final model using the full training set and determine the performance metrics on the test set.

(2.4) Repeat **(2.3)** using PLS (`step_pls`) in the preprocessing steps of the predictors. The model is still classification using logistic regression (see DS-6030: Partial least squares regression on how to install the required packages) (3 points - coding)

- Use `step_normalize` and `step_pls` to preprocess the data in a recipe.
- Use the `tune` function to find the best number of components (`num_comp`) in the range 1 to 14 using AUC as the selection criterium. Check **all** possible numbers of components from 1 to 14.
- Use the `autoplot` function on the cross-validation results to visualize the results. Describe your observations.
- Report the best number of components and the associated classification metrics.
- Using the best parameter set, train a final model using the full training set and determine the performance metrics on the test set.

(2.5) Compare the tuning results in **(2.3)** and **(2.4)** and comment on the differences. Do you see different behavior in the `autoplot` graphs? What do you think is going on? Could you reduce the number of components further from what is suggested by CV? (2 points - discussion)

6 DS-6030 Homework Module 6

Module 6

> **Note**
>
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-6.Rmd) and use it to answer the following questions.
> If not otherwise stated, use *Tidyverse* and *Tidymodels* for the assignments.

1. Predict out of state tuition (feature selection)

The data `College.csv` contains a number of variables for 777 different universities and colleges in the US. In this exercise, we will try to predict the `Outstate` tuition fee using the other variables in the data set.

(A) Data loading and preprocssing

(1.1) Load the data from `ISLR2::College` and split into training and holdout sets using a 80/20 split. (1 point - coding)

(B) Train linear regression and Lasso models

Use *tidymodels* to define a workflow and build a linear regression model to predict `Outstate` from all the other variables using L1 regularization (Lasso).

(1.2) For preprocessing, normalize all the numerical variables (`step_normalize(all_numeric_predictors())` and convert the categorical / nominal variables to dummy variables (`step_dummy(all_nominal_predictors())`). (1 point - coding)

(1.3) Train a normal linear regression model using the `lm` engine with the training set. Look at the p-values of the individual features. Which features are significant (use `extract_fit_engine` and `summary`). (1 point - coding/discussion)

(1.4) Use `glmnet` and tune the L1 penalty parameter using 10-fold cross-validation. Tune the penalty over the range `penalty(c(-1, 2.5))` and check that the range is appropriate using autoplot. (2 points - coding)

(1.5) Determine the best penalty parameters using the lowest $RMSE$ and the penalty obtained from the one-standard-error rule (`select_by_one_std_err`, see online material). For both penalties, finalize the the workflow and train models using the full training set. Report the coefficients of each model. Which variables are selected in each case? (2 points - coding/discussion)

(1.6) Use the model from (1.3) and the two models from (1.5) to predict the `Outstate` variable on the training and test set. Report the RMSE and R^2 of each model on the training and test set. (1 point - coding/discussion)

2. Predict out of state tuition (GAM model)

(C) GAM model

Using the significant features from (1.3), build a generalized additive model (GAM) to predict `Outstate`. (see Generalized additive models (GAM) for how to build GAM models in `tidymodels`)

(2.1) Define a model formula setting all numerical variables as splines and all categorical variables as factors (`Outstate ~ Private + s(Apps) + ...`) (1 point - coding)

(2.2) Define the `gen_additive_mod` model using `mgcv` as the engine and fit the model using the training data (2 points - coding)

(2.3) Report the RMSE and R^2 of the model on the training and test set and compare with the result from **(1.6)** (1 point - discussion)

(2.4) Use the `plot` function with the fitted model (use `extract_fit_engine` to get the actual `mgcv` model for plotting; set `scale=0` to get individual y-scales). Describe your observations. (1 point - coding/discussion)

(2.5) Use the `summary` function to get information about the model. Based on the reported significance levels, could you simplify the model further? (1 point - coding/discussion)

(2.6) Simplify the model by removing the non-significant variables and re-fit the model. Report the RMSE and R^2 of the model on the training and test set. (1 point - coding/discussion)

(D) Comparison

(2.7) Compare the results from the three models from **(1.3 and 1.5)**, the full GAM model from **(2.2)** and the reduced model from **(2.6)**? (2 points - discussion)

7 DS-6030 Homework Module 7

Module 7

> **Note**
>
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-7.Rmd) and use it to answer the following questions.
> If not otherwise stated, use *Tidyverse* and *Tidymodels* for the assignments.

1. Predicting Prices of Used Cars (Regression Trees)

The dataset contains the data on used cars (Toyota Corolla) on sale during late summer of 2004 in the Netherlands. It has 1436 records containing details on 38 variables, including Price, Age, Kilometers, HP, and other specifications. The goal is to predict the price of a used Toyota Corolla based on its specifications.gzc

Load the data from `https://gedeck.github.io/DS-6030/datasets/homework/ToyotaCorolla.csv.gz`.

(A) Load and preprocess the data

(1.1) Load and preprocess the data. Convert all relevant variables to factors. (2 points - coding)

(1.2) Split the data into training (60%), and test (40%) datasets. (1 point - coding)

(B) Large tree:

Define a workflow for a model to predict the outcome variable `Price` using the following predictors: `Age_08_04`, `KM`, `Fuel_Type`, `HP`, `Automatic`, `Doors`, `Quarterly_Tax`, `Mfr_Guarantee`, `Guarantee_Period`, `Airco`, `Automatic_airco`, `CD_Player`, `Powered_Windows`, `Sport_Model`, and `Tow_Bar`. Use the following settings depending on the used model engine:

- `rpart` engine: Keep the minimum number of records in a terminal node to 2 (`min_n = 2`), maximum number of tree levels to 30 (`tree_depth`), and *cost_complexity* = 0.0005 (`cost_complexity`) to make the run least restrictive resulting in a large tree.
- `partykit` engine: Keep the minimum number of records in a terminal node to 2 (`min_n = 2`) and maximum number of tree levels to 30 (`tree_depth`) to make the run least restrictive resulting in a large tree.

(1.3) Fit a model using the full training dataset. Inspect the resulting tree. Which appear to be the three or four most important car specifications for predicting the car's price? (1 point - coding/discussion)

(1.4) Determine the prediction errors of the training and test sets by examining their RMS error. How does the predictive performance of the test set compare to the training set? Why does this occur? (1 point - coding/discussion)

(1.5) How might we achieve better test predictive performance at the expense of training performance? (1 point - discussion)

(C) Smaller tree:

(1.6) Create a smaller tree. Compared to the deeper tree, what is the predictive performance on the test set? (3 points - coding/discussion)

- `rpart` engine: `min_n=2`, `tree_depth=3`, `cost_complexity=0.001`
- `partykit` engine: `min_n=2`, `tree_depth=3`

(D) Tuned tree:

(1.7) Now define a workflow that tunes the decision tree. Define a suitable range for the tuning parameter and use a tuning strategy of your choice. Make sure that the resulting best parameter is within the given range. (2 points - coding)

- `rpart` engine: tune `cost_complexity`
- `partykit` engine: tune `tree_depth`

(1.8) What is the best value for of your tuning parameter? What is the predictive performance of the resulting model on the test set? (1 point - discussion)

(1.9) How does the predictive performance of the tuned model compare to the models from **(1.3)** and **(1.6)**? What do you observe? (1 point - discussion)

(1.10) Train a final model for the optimal tuning parameters and visualize the resulting tree. (1 point - coding/discussion)

(E) Predicting the price of a car:

(1.11) Given the various models, what is the predicted price for a car with the following characteristics (make sure to handle the categorical variables correctly): (1 point - coding/discussion)

> `Age_08_04=77`, `KM=117000`, `Fuel_Type=Petrol`, `HP=110`, `Automatic=No`, `Doors=5`, `Quarterly_Tax=100`, `Mfr_Guarantee=No`, `Guarantee_Period=3`, `Airco=Yes`, `Automatic_airco=No`, `CD_Player=No`, `Powered_Windows=No`, `Sport_Model=No`, `Tow_Bar=Yes`

8 DS-6030 Homework Module 8

Module 8

> **Note**
>
> Module 8 introduced ensemble models. In this homework, we will build various regression model to predict fish toxicity of chemical compounds.
> You can download the R Markdown file and use it to answer the following questions. If not otherwise stated, use *Tidyverse* and *Tidymodels* for the assignments.
> The knitting times for the assignment can be very long without caching and parallel processing. For example, the calculations for problem 1 will take more than 16 minutes without parallel excution. Parallel processing reduces it to 5 minutes. With all results being cached, knitting a document will take only a few seconds after changing the text. You can find more information about caching here and about parallel processing here.

1. Predicting fish toxicity of chemicals (regression)

The dataset `qsar_fish_toxicity.csv` contains information about toxicity of 908 chemicals to fish. The data was downloaded from the UCI Machine Learning Repository. The dataset contains 7 features and the toxicity of the chemicals. The features are a variety of molecular structure descriptors. The toxicity is measured as the negative logarithm of the concentration that kills 50% of the fish after 96 hours of exposure.

- CIC0: Information indices
- SM1_Dz(Z): 2D matrix-based descriptors
- GATS1i: 2D autocorrelations
- NdsCH: Atom-type counts
- NdssC: Atom-type counts
- MLOGP: Molecular properties
- LC50: Toxicity towards fish (log value)

(A) Data loading and preparation

(1.1) Load the data from https://gedeck.github.io/DS-6030/datasets/homework/qsar_fish_toxicity.csv. *Hint*: the dataset has **no** column headers and is fields are separated by a semicolon; read the documentation for `read_delim`. (1 point - coding)

(1.2) Split the data into training (80%) and test (20%) sets using stratified sampling on `LC50`. Prepare the folds for 10-fold cross validation of all models. (1 point - coding)

(B) Model training

(1.3) Build the following models using the training set and evaluate them using 10-fold cross validation. For tuned models, use the `autoplot` function to inspect the tuning results and carry out cross-validation with the optimal parameters. (6 points - coding)

- Linear regression
- Random forest (tune `min_n` and `mtry`)
- Boosting model (tune `min_n` and `mtry`)
- k-nearest neighbors (tune `neighbors`)

(1.4) Report the cross-validation metrics (RMSE and r^2). What do you observe? Which model would you pick based on these results (2 points - discussion)

(1.5) Following cross-validation and tuning, fit final models with the optimal parameters to the training set. (1 point - coding)

(C) Model evaluation

(1.6) Evaluate the models on the test set and report their performance metrics RMSE and MAE. (1 point - coding/discussion)

(1.7) Create a visualization that compares the RMSE values for training and test sets of the four models. Do you see a difference between the models? Is there an indication of overfitting for any of the models? (2 point - coding/discussion)

(1.8) Create residual plots from the cross-validated predictions for the four models (add `geom_smooth` line). Combine the four plots in a single figure using the `patchwork` package. What do you observe? Are there differences in the residuals of the four models. (1 point - coding/discussion)

(1.9) For the boosting model, report the variable importance. You can use the `vip` package to create a variable importance plot. (1 point - coding)

9 DS-6030 Homework Module 9

Module 9

> **Note**
>
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-9.Rmd) and use it to answer the following questions.
> This assignment is only a first glimpse into handling text data. For a detailed introduction to text analytics in *tidymodels* see Hvitfeldt and Silge (2022, Supervised Machine Learning for Text Analysis in R).
> If not otherwise stated, use *Tidyverse* and *Tidymodels* for the assignments.

1. Sentiment analysis using SVM

In this assignment, we will build a model to predict the sentiment expressed in Amazon reviews. In order to build a model, we need to convert the text review into a numeric representation. We will use the `textrecipes` package to process the text data.

The data are taken from https://archive.ics.uci.edu/dataset/331/sentiment+labelled+sentences. You can load the data from https://gedeck.github.io/DS-6030/datasets/homework/sentiment_labelled_sentences/amazon_cells_labelled.txt

You will need to install the packages `textrecipes` and `stopwords` to complete this assignment.

(A) Setup

(1.1) Load the data. The data has no column headers. Each line contains a review sentence separated by a tab character (`\t`) from the sentiment label (0 or 1). Create a tibble with the column names `sentence` and `sentiment`. Use the *tidyverse* function `read_delim` to load the data. The dataset has **1000 rows** (the `read.csv` function fails to load the data correctly). Don't forget to convert sentiment to a factor. (2 point - coding)

(1.2) Split the dataset into training (80%) and test sets (20%). Prepare resamples from the training set for 10-fold cross validation. (1 point - coding)

(1.3) Create a recipe to process the text data. The formula is `sentiment ~ sentence`. Add the following steps to the recipe: (1 point - coding)

- `step_tokenize(sentence)` to tokenize the text (split into words).
- `step_tokenfilter(sentence, max_tokens=1000)` to remove infrequent tokens keepling only the 1000 most frequent tokens. This will give you a term frequency matrix (for each token, how often a token occurs in the sentence)
- `step_tfidf(sentence)` applies function to create a term frequency-inverse document frequency matrix.
- Use the `step_normalize()` function to normalize the data.
- Use the `step_pca()` function to reduce the dimensionality of the data. Tune the number of components, `num_comp`, in a range of 200 to 700.

(B) Model training Create workflows with the recipe from (c) and tune the following models:

(1.4) logistic regression with L1 regularization (`glmnet` engine tuning `penalty`) (2 point coding)

(1.5) SVM with linear kernel (`kernlab` engine tuning `cost`) (2 point coding)

(1.6) SVM with polynomial kernel (`kernlab` engine tuning `cost` and `degree`; use e.g. `degree = degree_int(range=c(2, 5))`) (2 point coding)

(1.7) SVM with radial basis function kernel (`kernlab` engine tuning `cost` and `rbf_sigma`; use `rbf_sigma(range=c(-4, 0), trans=log10_trans())`) (2 point coding)

Keep the default tuning ranges and only udpate `rbf_sigma` as mentioned above. For the PCA preprocessing step, tune `num_comp` in a range of 200 to 700. Use Bayesian hyperparameter optimization to tune the models. What are the tuned hyperparameters for each model?

(C) Model performance

Once you have tuned the models, fit finalized models and assess their performance.

(1.8) Compare the cross-validation performance of the models using ROC curves (combine in one graph) and performance metrics (AUC and accuracy). Which model performs best? (2 points - discussion)

(1.9) Compare the performance of the finalized models on the test set. Which model performs best? (2 points - discussion)

10 DS-6030 Homework Module 10

Module 10

> **Note**
>
> You can download the R Markdown file (https://gedeck.github.io/DS-6030/homework/Module-10.Rmd) and use it to answer the following questions.
> If not otherwise stated, use *Tidyverse*, *Tidymodels*, and *Tidyclust* for the assignments.

1. Analyzing the ANES 2022 Pilot Study - PCA

The ANES 2022 Pilot Study is a cross-sectional survey conducted to test new questions under consideration for potential inclusion in the ANES 2024 Time Series Study and to provide data about voting and public opinion after the 2022 midterm elections in the United States. Information about this study is available at https://electionstudies.org/data-center/2022-pilot-study/.

(A) Setup

Load the data from https://gedeck.github.io/DS-6030/datasets/anes_pilot_2022_csv_20221214/anes_pilot_2022_csv_20221214.csv

The dataset contains information about the respondents profile (e.g. birthyr, gender, race, educ, marstat, …) and answers to 235 questions from different categories.

Load and preprocess the data. The data contain answers to 235 questions (see PDF questionnaire). The dataset also contains a number of variables that are not questions, but rather contain information about how the survey was conducted (see user's guide and codebook).

(1.1) Identify the feeling thermometer questions. These questions ask respondents to rate their feelings toward a number of groups on a scale from 0 to 100. The questions are listed in variables starting with `ft...`. Identify the names of all feeling thermometer questions ignoring the `ftblack` and `ftwhite` questions as these were only asked based on race of the respondent and therefore contain a large number of missing values. Make sure that you exclude timing (e.g. `ftjourn_page_timing`) and order variables from your analysis. You should have 16 columns left. (1 point - coding)

(1.2) If a respondent did not answer a feeling thermometer question, the value is coded as a negative number. Replace the negative values with `NA` and remove all rows that have NA values for *any of the selected feeling thermometer questions*. (see `drop_na` function). You should have about 1560 data points left (1 point - coding)

(B) Principal Component Analysis (PCA)

You should now have a data frame that is suitable for a principal component analysis of the feeling thermometer responses.

Perform a principal component analysis of the feeling thermometer responses using `step_pca`.

(1.3) Create a scree plot of the eigen values. How many components should be considered? (1 point - coding/discussion)

(1.4) Create a biplot using the first two components. You will need to multiply the loadings with a factor to get an improved visualization. (1 point - coding)

(1.5) Interpret the first two components. What do they represent? Check the questionnaire for the questions that were asked. (1 point - discussion)

(C) Explore dataset

(1.6) The ANES 2022 Pilot Study is a rich data set. We can map the respondents profile and responses to other questions onto the principal component scatterplot. We start with the respondents profile. (2 points - coding/discussion)

Select the following profile data:

- gender
- educ (education level)
- marstat (marital status)

Add steps to convert the columns into factors in your data processing pipeline. See the questionnaire for the meaning of the different factor levels.

Combine the data set with the transformed PCA values.

Create scatterplots of the first two components, add a `geom_density2d` layer, and use `facet_wrap` to create a separate plot for each factor level.

Interpret the results. Can you see patterns?

(1.7) As an extension of **(1.6)**, we now focus on the answers to the actual questions. Select one of the question categories, formulate an hypothesis and see if you find a correlation with the PCA analysis. The categories are:

- 2022 Turnout and choice (6-20)
- Retrospective turnout and choice 2020 (21-24)
- Prospective turnout (25)
- Participation (26-33)
- Global emotion battery (34-40)
- Presidential approval (41-43)
- Party identification (44-50)
- Ideology (51)
- Economic performance (52-54)
- Inflation (55-64)
- Issue importance (65-79)
- Issue ownership (80-92)
- Climate change (93-94)
- Trust in experts (95-99)
- Political disagreement (100-101)
- Abortion (102-114)
- Abortion emotions (102-114)
- Transgender attitudes (123-126)
- Guns and crime (127-129)
- Imigrant emotions (130-136)
- Democratic attitudes / misinformation (137-146)
- Electoral integrity (147-159)
- Political efficacy (160)
- Feeling thermometers (161-179)
- Racism (180)
- Feminist attitudes (181-185)
- Racial resentment (186-190)
- Political tolerance (191-193)
- Racial stereotypes (194-206)
- Identities (207-208)
- Identity importance (210-220)
- Role of schools (221-226)
- Great replacement (227)
- Racial privilege (228-235)

Use a similar approach to **(1.6)** for the analysis (2 points - coding/discussion)

2. Analyzing the ANES 2022 Pilot Study - Clustering

We continue with the analysis of the ANES 2022 Pilot study and cluster the respondents based on their answers to the feeling thermometer questions.

(A) Hierarchical clustering

(2.1) Create a hierarchical clustering using the feeling thermometer data with the `tidyclust` package. Explore a variety of clustering methods (hier_clust). How many clusters should be considered? (2 points - coding)

(B) k-means clustering

(2.2) Use k-means clustering to cluster the respondents based on their answers to the feeling thermometer questions. Use the `tidyclust` package. (2 points)

- Create a k-means clustering with 5 clusters.
- Combine the dataset, the results from the PCA, and the k-means clustering in a tibble. [*Hint*: add to result from **(1.6)**]
- Create a scatterplot of the first two principal components and color the points by the cluster assignment. Describe your observations.
- Applying the `tidy` command to the fitted k-means model extracts the cluster centroids. Visualize the cluster centers in a parallel coordinate plot and interpret the different clusters. It can be helpful to order the variables for the visualization (use `scale_x_discrete(limits=c("fttrans", "ftfem", ...))` where the order is defined by the `limits` argument).

(C) Explore dataset

Characterize the different clusters.

(2.3) Use the profile data from **(1.6)** to characterize the different clusters. You can for example visualize the distributions of the different factor levels in a stacked 100%-bar plot (`geom_bar(position="fill")`). How does the distribution of the different factor levels differ between the clusters? Are your observations in agreement with the analysis from **(1.6)**? (2 points)

(2.4) Now use the questions from **(1.7)** to characterize the different clusters. How does the distribution of the different factor levels differ between the clusters? Are your observations in agreement with the analysis from **(1.7)**? (2 points)