Lab 1 - Machine Learning for Social Science

To be handed in no later than October 2nd (10:00). The submission should include code, relevant output, as well as answers to questions. We recommend the use of RMarkdown to create the report.

Part 1: Bernie Sanders and Donald Trump tweets.

For the first part of the lab, you will work with a dataset containing a sample of tweets from Donald Trump and Bernie Sanders. The objective is to explore how accurately we can predict who the author of a given tweet is based on its content, and to identify which words are the most discriminative.

The tweets have been preprocessed & cleaned for you, and are stored in a so-called document-term matrix format with rows indicating tweets, and columns indicating the frequency of words in different tweets.

1. Begin by importing the file "`trumpbernie.csv`" (hint: for example by using `data.table`'s `fread()` function or the standard `read.csv()`). Report how many *rows* and *columns* there are in this dataset (hint: you may for example use `dim()`). Would you characterize this data set as being *high-dimensional* or *low-dimensional*? Based on this, do you expect that a standard logistic regression will work well for the purpose of prediction?

2. Estimate a *standard logistic regression model* on the whole data set using the `glm` function (recall to specify `family="binomial"` in `glm`). The outcome variable is `trump_tweet`, and the rest of the columns (the word frequencies) are the input variables. Note: estimating the model may take a couple of minutes. When the estimation of the model is finished, do the following:

   a. Extract the coefficients from the estimated model using the `coef()` function and inspect the coefficients that are placed 1010–1050 in the output from `coef()` (hint: to inspect the coefficients placed 1010–1050 you may use standard brackets, e.g., `coef(mymodel)[1010:1050]`). Do you notice anything special?
   b. Examine the *training* accuracy of the estimated model. What does this result suggest about the predictive capacity of the model? You may use the following code to compute the training accuracy:

```r
# Extract predictions on training data & observed values
comparison_df <- data.frame(train_predictions=mymodel$fitted.values,
                            observed=mymodel$y)
# Apply prediction threshold
comparison_df$train_predictions<-ifelse(comparison_df$train_predictions>=0.5,
                                        yes = 1,
                                        no = 0)
# Compute accuracy (scale: 0-1, 0=0%, 1=100%)
nrow(comparison_df[comparison_df$train_predictions==comparison_df$observed,]) /
  nrow(comparison_df)
```

3. Use the `caret` package to implement a 3-fold cross-validation procedure that estimates the *test* accuracy of a *standard logistic regression* (hint 1: two functions are relevant here: `trainControl` and `train` | hint 2: specify `method="glm"` and `family='binomial'` in `train` to fit a standard logistic regression). Note, before you run `train()`, make sure you have formatted the outcome variable `trump_tweet` as a `factor` variable (this is needed for the "caret"'package to recognize the problem as a classification problem). Report the accuracy. Does this result align with your expectations from #1 and #2? Do the results from #2 and #3 provide any indications of either over- or underfitting?

4. Now we shall move beyond the standard logistic regression, and more specifically, turn to ridge regression for our prediction task. This importantly entails deciding on a value for the parameter λ. Use `glmnet`'s function `cv.glmnet` to find the λ that minimizes the test error, and report the associated test accuracy. Is this a better or worse prediction model compared to the one in #2–3? Which of the two models do you believe have a higher variance? Why?

   - When specifying `cv.glmnet`'s arguments, note the following: (i) Unlike `glm` and `train`, the `cv.glmnet` function does not have a `data` argument. Instead, you have to specify `x` and `y` separately (hint: you can for example use `$` to extract and delete columns). (ii) `alpha` dictates the model type (0=ridge, 1=lasso), and `standardize` whether you want to standardize the data prior to estimation (which we *do*). (iii) Finally, set the number of folds to 5 (`nfolds=5`), `family`="binomial" (to indicate that `y` should be treated as a binary variable), and `type.measure="class"` (to retrieve a measure of *accuracy*).

5. Plot lambda against the misclassification error (hint: just `plot()` the object which you stored the output from `cv.glmnet`). Interpret the plot in terms of bias and variance.

6. Lastly, extract the coefficients associated with the lowest test error (hint: you can use `coef(myfit, s='lambda.min'` for this). Have a closer look at the coefficients with the largest *positive* and largest *negative* values. What do they reveal? Do the words you find on either side confine to your expectations?

Part 2: Social Network Ad Purchase

For the second part of the lab, you will work with a dataset that contains information about individuals' purchasing behavior under exposure to online ads. The data originates from an online shopping site, and can be downloaded from *Kaggle*. Our goal is to examine how well we can predict purchases on the basis of `Age`, `Gender` and `Salary`, and to explore the character of the associations between these variables and the outcome.

1. Begin by importing the file "`Kaggle_Social_Network_Ads.csv`". Format the outcome variable `Purchased` as a `factor` variable (this is required for the subsequent analysis using the `caret` package).

2. Use the `caret` package to implement a 5-fold cross-validation that assesses the test accuracy of a *standard logistic regression model* (hint: two functions are relevant here: `trainControl` and `train`). Report its test accuracy. Note: To ensure that the folds generated by `caret` are identical for different models, add a `set.seed(12345)` above each use of `train()`.

3. To investigate whether *GAMs* can improve the performance over the standard logistic regression, implement three separate 5-fold cross-validations; each estimating a GAM with a different degree of freedom for the natural cubic splines ($df \in \{2,3,4\}$). Create splines for the two variables *Age* and *Salary*, but not for *Gender*, which is a categorical variable (hint: use the `ns()` function from the `splines` package to create splines). Again, to ensure identical folds, add `set.seed(12345)` above each `train()`. You may also re-use the `trainControl` object from the previous task. Report the accuracies of the different models. Having do so, answer the following questions:

a. Do you observe any improvement compared to the standard logistic regression?
b. What does the difference in performance between the standard logistic regression and the GAMs suggest about the former? Is it over- or underfitted? Does it have high(er) bias or high(er) variance compared to the GAMs?
c. Which of the three GAMs do you prefer? Motivate.

4. Next, you shall examine the predictive relationships between the two continuous variables (*Age* and *Salary*) and the outcome. For this, you will use `ggeffects`'s `ggpredict()` function which computes predictions while *varying one* variable and holding the *remaining fixed* at their means/mode. To do so, first re-estimate the GAM-specification that you found to be the best, on the full data using `lm` (`ggeffects` does not accept objects from `caret`). Interpret. Do you find any non-linear relationship?
5. In this second part of the lab, we used GAMs to improve predictive performance. Would we expect to see similar improvements if we instead had used *ridge* and *lasso* regression? Why/why not?

Part 3: K-fold cross validation from scratch (bonus; not required for passing)

For the third and last part of the lab (which is a *bonus* for those who happen to finish part 1 and 2 early, i.e., it is not needed to pass the assignment), you will work with a dataset named `dt.rds` that contains two variables `X` and `Y`. Your task is to implement a k-fold cross validation procedure—from scratch—that helps you identify an appropriate polynomial degree for `X` when predicting `Y` in a standard `lm()` model. You shall create a function that takes as input two parameters, `k` (the number of folds), `d` (the polynomial degree), and `dt` (the dataset), and which returns the average MSE across the `k` folds. Here are a few hints that may help you along the way:

- You may use `lm(Y ~ poly(X,d),data=dt)` to fit a d-degree polynomial.
- You may use `stackoverflow`'s function `chunk2()` to partition observation indices into groups/folds.

Quiz-wrap up:

Provided here are two optional wrap-up questions that help recap some central ideas from the lecture, and which are taken from old exams.

1. For each of the scenarios described below (a–c), indicate whether it is a *classification* or *regression* problem, and further whether we are most interested in *inference* or *prediction*:
    a. We collect a set of data on students from 1000 randomly sampled schools in Sweden. For each school we record student-level information (average grade, gender) and school-level information (number of students in the school, whether the school is private or public). We are interested in understanding which individual-level and which school-level factors affect the average grade of students. Grades are recorded on a scale 1-100.
    b. We are interested in understanding the relation between the centrality of a twitter user (proxied by the number of followers) and the number of retweets her tweets get. We therefore collect a random sample of 10000 twitter users and their tweets. For each tweet, we record the content of the tweet (the text), the author's current number of followers, and the tweet's number of retweets.
    c. We are interested in predicting whether a student will drop out of high school or not based on the information we have on them in middle-school. The motivation for this is that we want to identify "students at risk" early on. We have therefore collected lots of data on students who either finished or dropped out of high school. For each student, we record individual- and family-level information when they attended middle school, and also whether the finished or dropped out of high school.

2. Figure 1 (see next page) displays a conceptual graph of the so-called *bias-variance trade-off*. Your task is to pair each shaded region (denoted a–c) with one of the fitted lines (denoted i–iii) found in the scatter-plots below the bias-variance plot. For example, one possible pairing-configuration could be $\{a : ii\}$, $\{b : i\}$, $\{c : iii\}$.

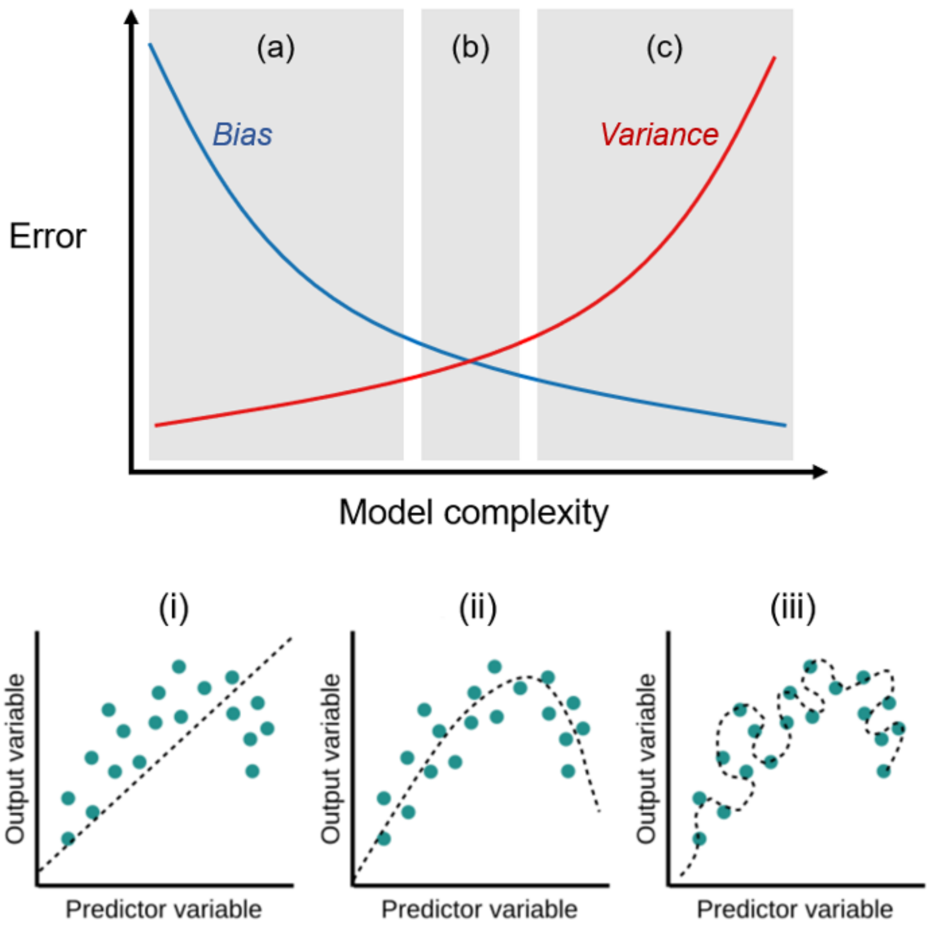

Figure 1: Bias variance trade-off