Oblig report

October 14, 2019

1 STK-IN4300 Oblig 1

1.1 Task 1

[28]:
```python
# Importing needed modules
import pandas as pd
import numpy as np
import sklearn.linear_model as skllm
import sklearn.model_selection as sklms
import sklearn.preprocessing as sklpre
import warnings
# Ignoring annoying deprecation warnings from Scikit-Learn
warnings.filterwarnings("ignore", category=DeprecationWarning)
```

[1]:
```python
# Loading R into notebook
%load_ext rpy2.ipython
```

[ ]:
```r
%%R
# Running R-code in notebook, nice way to contain R and Python in same file!
library(BiocManager)
library(ArrayExpress)
system("mkdir E-GEOD-12288")
tmp <- getAE("E-GEOD-12288", type = "processed", path = "E-GEOD-12288")
tmpProc <- getcolproc(tmp)
tmpE <- procset(tmp, tmpProc[2])
# extract information
info <- pData(tmpE)
# extract input matrix (gene expressions)
X <- t(exprs(tmpE))[info[, 40] == "case", ]
# extract response (CADi)
CADi <- info[info[ , 40] == "case", 37]
# remove temporary files
rm(info, tmp, tmpE, tmpProc)
# Saving data for later to be loaded into Python. Do not actually need to save
 ↪it here, as Python could read R variables
# because of the notebook environment I'm running, but I did not want to
 ↪download the data more than once.
```

```
write.csv(data.frame(CADi, X), file="data/data_for_python.csv")
```

[14]:
```python
# Loading the saved file using Pandas
data = pd.read_csv("data/data_for_python.csv")

# Dropping useless column
data.drop(["Unnamed: 0"], axis=1, inplace=True)

# Extracting CADi-data from dataframe
y = np.array(data["CADi"], dtype=np.float32)#.reshape(-1, 1)

# Dropping CADi data from dataframe to make creating X simpler
data.drop(["CADi"], axis=1, inplace=True)

# Creating design matrix in array-form, as arrays are more memory efficient than␣
↪dataframes.
X = np.array(data, dtype=np.float32)

# Deleting dataframe to save RAM
del data

# Used for scaling
scaler = sklpre.StandardScaler()

# Scaling features
X = scaler.fit_transform(X)

# Scaling data (Not sure if you should actually scale the data), the reshape()␣
↪and ravel() are just so that the
# fit_transform function works properly, probably a dumb workaround.
y = scaler.fit_transform(y.reshape(-1, 1)).ravel()

# Splitting into train and test
X_train, X_test, y_train, y_test = sklms.train_test_split(X, y, test_size=0.33)

# Deleting X to save RAM
del X
```

[30]:
```python
# Ridge with k=5 cross validation used to find best hyperparameter
ridge = skllm.RidgeCV(fit_intercept=False, cv=5, gcv_mode="auto").fit(X_train,␣
↪y_train)
```

[7]:
```python
# Lasso with k=5 cross validation used to find best hyperparameter
lasso = skllm.LassoCV(fit_intercept=False, cv=5, n_jobs=-1, selection="random").
↪fit(X_train, y_train)
```

[31]:
```python
# Fancy print formatting
print(f"Ridge: Train R2: {ridge.score(X_train, y_train):7.3f}. Test R2: {ridge.
↪score(X_test, y_test):7.3f}. Best lambda: {ridge.alpha_:7.3f}.")
print(f"LASSO: Train R2: {lasso.score(X_train, y_train):7.3f}. Test R2: {lasso.
↪score(X_test, y_test):7.3f}. Best lambda: {lasso.alpha_:7.3f}.")
```

```
Ridge: Train R2:   1.000. Test R2:  -0.033. Best lambda:  10.000.
LASSO: Train R2:  -0.001. Test R2:  -0.004. Best lambda:   0.532.
```

As we can see, the best models of both Ridge and LASSO are completely useless, with R^2-scores of ~ 0, meaning they are as bad as using only the mean of the data as a model. This seems to imply that gene data cannot be used to predict fat content in blood, which makes sense, as fat comes from lifestyle and not genetics! It also makes sense that trying to use ~ 22000 features with only ~ 100 data points is a bad idea.

1.2 Task 2

We want to solve

$$\text{argmin}_\omega \left[\sum_{i=1}^{n} g'(\omega_{\text{old}}^T x_i)^2 \left(\frac{y_i - g(\omega_{\text{old}}^T x_i)}{g'(\omega_{\text{old}}^T x_i)} + \omega_{\text{old}}^T x_i - \omega^T x_i \right)^2 \right]. \tag{1}$$

We recognize this as a weighted least squares problem if we set the weights $\mathbf{W} = g'(\omega_{\text{old}}^T x_i)^2$ and $\mathbf{b} = \frac{y_i - g(\omega_{\text{old}}^T x_i)}{g'(\omega_{\text{old}}^T x_i)} + \omega_{\text{old}}^T x_i$ and rewrite everything on matrix form, we get

$$\text{argmin}_\omega \left(\mathbf{W} || \mathbf{b} - \mathbf{X}\omega ||^2 \right).$$

This has the solution (differentiate it and set it to zero)

$$\omega = \left(\mathbf{X}^T \mathbf{W} \mathbf{X} \right)^{-1} \mathbf{X}^T \mathbf{W} \mathbf{b}.$$