Assignment Declaration

MIS41120 Statistical Learning

Module Coordinator: Dr. Seán McGarraghy

Submission date: 23/07/2018
Submission PDF: stl_Chen_Daly_Wang_report.pdf

We confirm that:

1. **We understand what plagiarism means and accept the plagiarism policy as set out in the Student Handbook.**
2. **This assignment has been prepared entirely by the members of the group and steps have been taken to ensure that nobody else has been able to copy our work in any form.**
3. **This assignment does not contain any material taken from unacknowledged sources and that all material has been referenced.**
4. **We are the original authors of all the work presented in this assignment.**

The assignment has been prepared for assessment in this course and has not been presented as course work in any other course.

Group Members' Details

Group Member's Name	**Student Number**	**Student Signature**
Li Chen	17200395	*Li Chen*
Niamh Daly	16202504	*Niamh Daly*
Suohuijia Wang	17200170	*Suohuijia Wang*

1. Introduction

Statistical learning is a field plays a major role in many areas of science, finance and technology. It aims to learn patterns from data to be able to analyse the data and to make predictions for future similar scenarios (Freidman et al., 2001).

As technology has grown over the past number of years, there have been many statistical modelling techniques developed for use. However, when it comes to choosing a model, there is no "one size fits all". A method may be chosen based on the functionality of the method, the interpretability of the technique, the properties of the dataset, as well as what we aim to do with our final model.

We investigated three different statistical learning techniques: Multiple Linear Regression (MLR), Support Vector Machine (SVM) and Multilayer Perceptron (MLP). For each method, different regularisation methods are used, including ℓ1 penalty (lasso regression), ℓ2 penalty (ridge regression) and elastic net regression. All the methods are implemented across two datasets.

The first data set is the Boston dataset, which can be loaded from the 'MASS' package in R. It is a classical test dataset which we used to predict the crime rate relative to the other variables in the dataset.

The second dataset is a real dataset called "Appliances energy prediction" (Candenado, 2017) which gives the energy usage in Watts per hour for a number of appliances, along with a number of other variables such as temperature and humidity in a number of rooms in the house, as well as current weather conditions. The data set includes 19,735 rows and has more than 20 predictors, making it a relatively large dataset. We used this data set to test our learning methods by predicting the energy use of appliances as a function of all the relevant variables.

For each dataset, we built 15 models according to the three algorithms and three regularisation methods, shown in table 2 below. Some of the models were also applied with cross validation to see if we could make improvements in the model performance. For both datasets, 10-fold cross validation was used for MLR.

Initially, all three algorithms were implemented in R. However, whilst the MLR algorithm is quick to run, we did not experience the same speed for SVM and MLP, which took too much time and tended to crash the PC, particularly for the larger dataset. Because of this, we decided to implement them in Python instead.

For our loss function, we used the Mean Square Error (MSE) between the values in the test datasets and the model's predicted values to determine the models' accuracies. After running 10 times for each model, we got ten MSE values and then we averaged the MSEs to get the final results. We also discussed interpretability and efficiency of the algorithms in the experiments, so that we can evaluate the models in all aspects. Standard deviation is also given, but only used as a reference.

For each variable in each model, we discuss whether or not we can reject the null hypothesis, i.e. that the variable has no effect and is not relevant for predicting the output.

2. Test Dataset – Boston Dataset

2.1 Dataset Description

The first dataset we used is the Boston dataset, which is included in 'MASS' package in R. The dataset contains information concerning housing in the area of Boston, Massachusetts, and there are 506 rows, each of which has 14 attributes. The attributes are shown in Table 1.

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 - nitric 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 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
B - 1000(Bk - 0.63) ^2 where Bk is the proportion of blacks by town
LSTAT - % lower status of the population
MEDV - Median value of owner-occupied homes in $1000's

Table 1: The Attributes in Boston dataset

We chose to predict CRIM in terms of other 13 variables. The algorithms we used are: multiple linear regression (MLR), support vector machine (SVM) and multilayer perceptron (MLP). Among them, multiple linear regression and MLP are applied with ridge regression (ℓ2), lasso (ℓ1) and elastic net while SVM is only applied with ridge regression and lasso regression.

The MLR is implemented in R and the other two algorithms – SVM and MLP are implemented in python.

2.2 Data pre-processing

(1) Data loading

In R, we loaded the Boston dataset from the package 'MASS' and saved it to a .csv file. In python, we loaded the dataset from the saved .csv file. There are 506 rows and 14 columns in this dataset.

(2) Data splitting and normalization

The dataset was randomly split into two datasets, training dataset and test dataset. The proportion of the training dataset and test dataset is 70% and 40%, so there are 354 rows in the training dataset and 152 rows in the test dataset.

For SVM and MLP, we normalised the data to a 0-1 scale. Normalisation is recommended for these methods due to the nature of the technique; if we neglected to do this, our algorithms would detect changes in variables distributed over a large range and fail to detect similar percentage changes in variables distributed over a much smaller range of values. Thus, the larger values would have much higher weightings simply for being large. Normalising the data removes this risk.

(3) Modelling

Using the Boston dataset, we built 15 models with 3 algorithms. The algorithms and methods are shown below in Table 2:

MLR	MLR: Multiple Linear Regression

	Ridge: Ridge Regression
	Ridge.CV: Ridge Regression with Cross Validation
	Lasso: Lasso Regression
	Lasso.CV: Lasso Regression with Cross Validation
	ElasticNet: Elastic net Regression
	ElasticNet.CV: Elastic net Regression with Cross Validation
SVM	SVM: Support Vector Machine (C=0.5)
	SVM: Support Vector Machine (C=1)
	SVM.ℓ1: Support Vector Machine with ℓ1(Lasso) penalty
	SVM.ℓ2: Support Vector Machine with ℓ2(Ridge) penalty
MLP	MLP: Multilayer Perceptron
	MLP.ℓ1: Multilayer Perceptron with ℓ1(Lasso) penalty
	MLP.ℓ2: Multilayer Perceptron with ℓ2(Ridge) penalty
	MLP.Elastic: Multilayer Perceptron with elastic net

Table 2. Different algorithms and methods in modelling

In linear regression, we used 3 types of regularization as penalty in the simple model and for each model, we applied 10-fold cross validation to enhance the accuracy. In SVM, we were not able to find a way to apply elastic net regularisation, so we only applied ℓ1 and ℓ2 penalty to the model. For MLP, we applied 3 regularisations to the original model.

(4) Analytical methods

We ran each model 10 times after shuffling the dataset by setting different seeds in R and then calculated the MSE of the test dataset for each model. We then averaged the 10 results for each model to get the mean MSE for each model, which we used as our main criterion for acceptance or rejection of the model in question. The standard deviation was also calculated for each model to show the fluctuations of the data.

2.3 Modelling and Results

(1) Multiple Linear Regression (MLR)

In multiple linear regression, 7 methods were used: MLR, Ridge, Ridge.CV, Lasso, Lasso.CV, ElasticNet and ElasticNet.CV. For each method, we calculated the averaged MSE and standard deviation of the test dataset for the 10 runs in R. The results are shown below in Table 3:

MSE of Boston Test Dataset							
	MLR	**Ridge**	**Ridge.CV**	**Lasso**	**Lasso.CV**	**ElasticNet**	**ElasticNet.CV**
1	12.84	12.85	12.12	24.57	**12.05**	15.06	12.50
2	**37.14**	38.74	37.57	57.43	37.57	38.75	37.22
3	34.87	35.10	**34.26**	55.76	34.49	41.23	34.69
4	43.34	44.06	42.58	60.71	44.20	45.29	**42.58**

5	19.60	**15.08**	17.43	21.84	18.77	21.84	17.54
6	39.10	41.80	39.28	58.80	39.10	42.63	**39.03**
7	43.17	**42.28**	42.36	53.24	43.34	43.81	42.39
8	33.74	33.67	**33.07**	49.17	33.27	35.88	33.08
9	18.08	19.19	**17.68**	31.63	17.79	20.09	17.88
10	20.65	19.34	**19.18**	33.17	20.06	21.86	19.24
mean	30.25	30.21	**29.55**	44.63	30.06	32.64	29.62
std	**10.74**	11.62	11.08	14.37	11.17	10.98	10.95

Table 3. MSE of Boston Test Dataset with MLR

From Table 3, we can see that the majority of the lowest MSE are from the Ridge.CV method, and the mean of the 10 MSEs with Ridge.CV is also the lowest. Therefore, we can conclude that Ridge regression with cross validation performed the best in Boston dataset. Besides, the differences of MSE with methods using cross validation are not very large, which shows us that the multiple linear regression with different penalties performed nearly the same in predicting the crime rate with Boston dataset.

Secondly, the MSE with methods using cross validation are lower than those with methods not using cross validation in all the algorithms, which means that the methods with cross validation performs better in prediction than methods without cross validation.

A different summary of the model was generated for each of the ten runs. For the 5% of the confidence level, the null hypothesis can be rejected in all the ten runs for the predictors: 'rad', 'medv', 'dis', 'zn'. For some of the ten runs (but not all), we can reject the null hypothesis for the predictors 'black', 'nox', 'indus'.

For LASSO, the zero-coefficient of variables changed in terms of the randomly chosen training datasets. In some dataset, 'tax' is the zero-coefficient variable, and in some other dataset, 'nox', 'age', 'tax' and 'ptratio' are the zero-coefficient variables. For the variables with zero-coefficient, we cannot reject the null hypothesis while for the variables with non-zero coefficients we can reject the null hypothesis. The changes in the variables with zero-coefficient may because of the small size of the dataset. If we use a huge dataset, we might get the stable results.

For elastic net regularisation, for some runs, the coefficient of variables 'age' and 'tax' are 0. This means that 'age' and 'tax' are not very important predictors, and could be removed from the model.

(2) Support Vector Machines (SVM)

For Support Vector Machines, we ran three different models, all using a linear kernel. One version had no regularisation, and the other two used ℓ1 and ℓ2 penalties to train the model from the data. The MSE was once again used as our cost function. The models were built in Python and the results are shown below in Table 4:

MSE of Boston Test Dataset				
SVM				
	no penalty(C=1)	no penalty(C=0.5)	ℓ1	ℓ2
1	181.63	119.99	52.94	**48.78**
2	93.18	126.99	**6.10**	13.02

3	96.50	168.21	**43.86**	83.60
4	133.49	118.88	**43.31**	43.69
5	165.33	108.87	**32.15**	73.99
6	93.83	138.42	55.28	**54.85**
7	133.75	152.87	**42.77**	75.86
8	132.38	159.75	32.17	**24.01**
9	145.82	176.66	**39.26**	54.05
10	118.69	127.00	91.44	**62.30**
mean	129.46	139.76	**43.93**	53.42
std	28.47	22.03	**20.49**	21.29

Table 4. MSE of Boston Test Dataset with SVM

It is obvious that MSEs of SVM with penalties are much lower than those without any sort of penalty, which means that the both the ℓ1 penalty and the ℓ2 penalty on the coefficients play an important role in training a linear SVM model. The penalty greatly enhances the accuracy of the prediction.

Comparing the performance of the two methods with penalties, we can see that the MSE of SVM with ℓ1 penalty and the standard deviation are the lowest of the SVM models. Therefore, we can make the conclusion that SVM with ℓ1 penalty performed the best in the prediction of Boston crime rate and the result is the most stable according to the standard deviation of the MSE.

(3) Multilayer Perceptron (MLP)

In Multilayer Perceptron, we used normalised data and then built MLP models with linear activation layer functions, first with no penalisation, and then with ℓ1, ℓ2 and elastic net penalties. The activation of the output layer is linear. Similar to both MLR and SVM, the cost function used was also MSEs of the test dataset. The models were built in Python and the results are shown in Table 5.

MSE of Boston Test Dataset				
MLP				
	no penalty	ℓ1	ℓ2	Elastic net
1	55.30	144.20	69.83	**31.09**
2	99.69	112.04	**23.45**	103.16
3	**99.09**	117.39	102.01	114.80
4	148.24	171.37	**101.18**	128.03
5	22.92	92.33	34.70	**21.75**
6	143.76	77.82	156.61	**65.70**
7	**38.50**	106.05	54.15	108.82
8	117.04	89.01	**46.56**	86.65
9	**36.52**	38.65	221.37	47.25
10	105.06	**36.00**	50.68	118.87
mean	86.61	98.49	86.05	**82.61**
std	43.03	40.05	58.67	**36.65**

Table 5. MSE of Boston Test Dataset with MLP

Across the ten runs, there is no method which consistently achieves the best result; the performance of each method seems entirely dependent on the data split. The mean and the standard deviation of MSE for the ten runs in MLP with elastic net are the lowest; however, due to the large variation across runs and the fact that no algorithm stands out across the three probably means that we need a large number of runs to be sure that elastic net MLP is superior.

MLP with ℓ2 penalty has the highest standard deviation, which means that the prediction of this method is much more unstable than the other 3 methods.

2.4 Conclusion of Test Dataset

With the Boston dataset, we built 15 models to predict the crime rate in Boston area and for each model, we got the averaged MSE of test dataset for 10 runs. The cost function used for each technique is the MSE and the model with the lowest MSE on the test dataset is considered the most superior model in predicting the crime rate with Boston dataset.

Table 6 below shows all the MSEs achieved across each different model. Among these results, MLR ridge regression using cross validation is the lowest with a MSE of 29.55. Therefore, we can conclude that ridge regression with cross validation outperformed all the other methods when predicting the crime rate with Boston dataset.

MSE of Boston Test Dataset				
MLR	**MLR**	**Ridge**	**Lasso**	**ElasticNet**
	30.25	30.21	44.63	32.64
		Ridge.CV	**Lasso.CV**	**ElasticNet.CV**
		29.55	30.06	29.62
SVM	**no penalty(C=1)**	**no penalty(C=0.5)**	**ℓ1**	**ℓ2**
	129.46	139.76	**43.93**	53.42
MLP	**no penalty**	**ℓ1**	**ℓ2**	**ElasticNet**
	86.61	98.49	86.05	**82.61**

Table 6. MSE of Boston Test Dataset with All Algorithms

A second conclusion is that methods with cross validation tend to perform much better than methods without cross validation, which we can see from the experiments of MLR. In general, k-fold cross validation can enhance the prediction accuracy by calculating the mean of results from the k training datasets and our result of the experiment is consistent with it.

Lastly, it is clear that applying a penalty term to the method reduces the model's capacity to overfit the data and allows us to make better predictions on the test dataset.

3. Real Dataset

3.1 Dataset Description

Data source

We chose a real dataset online, "Appliances energy prediction" (Candenado, 2017) to analyse model performance based across a number of different methods as we have done for the test data. This dataset has just short of 20,000 rows and consists of 29 variables. We chose to use the appliances' energy use in Watts per hour as our target variable, and treat all other variables as predictors.

Dataset Information

Each line in the dataset records an appliance's energy use at ten-minute intervals over a 4.5 month period. The house temperature and humidity conditions were monitored using a wireless sensor network, which transmitted temperature and humidity conditions every few minutes, and which were then averaged for 10-minute periods. Weather data was obtained from publicly available weather data which had been logged at the nearest airport weather station. Interestingly, and usefully for our own purposes, two columns containing completely random variables have been included in order to test regression models and to allow for the exclusion of non-predictive attributes where required (Candenado, 2017). Table 7 features descriptions of all of the variables contained in the dataset.

Attribute Information (predictors)	
date time year-month-day hour: minute: second	T7, Temperature in ironing room, in Celsius
lights, energy use of light fixtures in the house in Watts per hour	RH_7, Humidity in ironing room, in %
T1, Temperature in kitchen area, in Celsius	T8, Temperature in teenager room 2, in Celsius
RH_1, Humidity in kitchen area, in %	RH_8, Humidity in teenager room 2, in %
T2, Temperature in living room area, in Celsius	T9, Temperature in parents room, in Celsius
RH_2, Humidity in living room area, in %	RH_9, Humidity in parents room, in %
T3, Temperature in laundry room area	To, Temperature outside (from Chievres weather station), in Celsius
RH_3, Humidity in laundry room area, in %	Pressure (from Chievres weather station), in mm Hg
T4, Temperature in office room, in Celsius	RH_out, Humidity outside (from Chievres weather station), in %
RH_4, Humidity in office room, in %	Wind speed (from Chievres weather station), in m/s
T5, Temperature in bathroom, in Celsius	Visibility (from Chievres weather station), in km
RH_5, Humidity in bathroom, in %	Tdewpoint (from Chievres weather station), Â°C
T6, Temperature outside the building (north side), in Celsius	rv1, Random variable 1, non-dimensional
RH_6, Humidity outside the building (north side), in %	rv2, Random variable 2, non-dimensional
Attribute Information (target):	Appliances, energy use in Watts per hour

Table 7: Attributes Information for Predictors and Target. (Cabenado, 2017)

We chose to predict the variable "Appliances" as a function of the other 28 variables. The algorithms we used are: multiple linear regression (MLR), support vector machine (SVM) and multilayer perceptron (MLP). We performed ridge regression (ℓ2), lasso (ℓ1) and elastic net regularisation on MLR and MLP, whilst SVM is only trialled with ridge regression and lasso regression penalisation.

The MLR methods are implemented in R and the other two algorithms, SVM and MLP, are implemented in Python.

3.2 Data Pre-processing

(1) Data loading

We downloaded the dataset "energydata_complete.csv" and read the data into our scripts. There are 19,735 rows and 28 columns in this dataset.

(2) Data normalisation and splitting

The first few lines of the code deal with some pre-processing. This involves removing null or incomplete values from the data, and converting non-numeric values to numerical values – in this case, the date and time is converted to a UTC values. As above, we then normalised the data to 0-1 scale so that the predictions of the models can be much more precise. Then, we split the data into two parts: training data and test data according the proportion is 70%/30%. This means that we have 13814 rows in the training dataset and 5921 rows in the test dataset.

(3) Modelling

We created the same models for the real data as we did for the test data (see Table 2 above); we created three models: MLR, SVM and MLP. We also used the same three regularisation approaches, lasso, ridge, and elastic net penalisation. We then built these three models and included different regularisations to observe the performance. We used the MSE of the test data set to evaluate which modelling technique was preferable.

(4) Analytical methods

We used the same analytical methods as what we have done in Boston dataset. For each model in MLR, we ran ten iterations of the technique after splitting the dataset by setting different seeds in R. For each model in SVM and MLP, we ran ten iterations times after splitting the dataset randomly in Python, and then calculated the MSE of the test dataset for each model. We then took the average of the ten test MSE results for each model to get the mean MSE per model. The standard deviation was also calculated for each model to show the fluctuations of the data. The main criterion is the average MSE and the standard deviation is for reference only.

3.3 Modelling and Results

(1) Multiple Linear Regression (MLR)

In MLR, 7 methods were used: MLR, Ridge, Ridge.CV, Lasso, Lasso.CV, ElasticNet and ElasticNet.CV. For each method, we calculated the averaged MSE and standard deviation of the test dataset for the 10 runs in R. The results are shown below in Table 8.

From Table 8, we have 7 columns. MLR means the original multiple linear regression with no penalisation. Ridge refers to ℓ2 regression, and lasso ℓ1 regression. ElasticNet refers to elastic net regression. The other three models, Ridge.CV, Lasso.CV and ElasticNet.CV refer to the above three methods with cross-validation.

We can see that the smallest mean MSE belongs to the last column, which is combining MLR with elastic net regularisation and doing the cross-validation. Looking at individual runs, we can see that the elastic net technique achieves the best result in six out of the ten runs. Based on the results, we see that ElasticNet.CV performs better than MLR with no penalisation, and both lasso and ridge regularisation, since it combines their advantages together to achieve an optimal result.

In addition, we can see that, for each regularisation, model performance is improved when we include the cross-validation. As expected, methods with cross-validation performs better in prediction than methods without cross-validation.

Summaries of the models were generated for each of the ten runs. At the 5% confidence level, the null hypothesis can be rejected in all ten runs for the predictors: ‘lights’, ‘RH-1’, ‘T2’, ‘RH_2’, ‘T3’, ‘RH_3’, ‘T4’, ‘T6’, ‘RH_7’, ‘T8’, ‘RH_8’, ‘T9’, ‘RH_9’, ‘T_out’, ‘Windspeed’. And some of the predictors, such as ‘Visibility, ‘RH_5’, ’Tdewpoint’, can only be used to reject the null hypothesis in some of the ten runs.

For lasso regularisation, as what we have done in Real dataset, the zero-coefficient of variables changed in terms of the randomly chosen training datasets. Some are ‘T5’ and ‘T7’, some are ‘rv2’, and some are ‘RH_out’. We noticed that the confidence levels for each variable changed during the ten runs, which is due to the fact that we are using different training dataset and test datasets for each run which have marginally different properties.

When we consider our results for elastic net regularisation, all of the coefficients of variables are non-zero, which would indicate that these predictors are all important in prediction. This is surprising since two of the columns are completely random variables and in theory should have absolutely no relevance to our final output variable!

Multiple Linear Regression							
	MLR	**Ridge**	**Ridge.CV**	**Lasso**	**Lasso.CV**	**ElasticNet**	**ElasticNet.CV**
1	8620.60	8748.43	8693.03	9482.58	**8620.49**	8774.85	**8620.49**
2	9673.82	9800.57	9747.58	10420.43	9673.79	9819.14	**9673.46**
3	8421.65	8544.51	8497.20	9161.74	**8418.12**	8562.12	8418.19
4	9013.00	9052.29	9021.94	9582.73	9008.76	9196.37	**9006.06**
5	8590.09	8693.23	8645.75	9296.82	8591.38	8721.44	**8589.73**
6	**8372.87**	8469.75	8423.75	9038.66	8375.60	8488.39	8373.13
7	**9424.12**	9531.91	9485.30	10148.00	9427.01	9532.89	9427.01
8	9221.92	9336.64	9278.51	10031.65	9216.47	9439.72	**9215.66**
9	8669.32	8779.87	8726.80	9410.86	8668.31	8795.56	**8668.23**

10	8489.82	8580.08	8528.80	9207.26	8483.98	8669.05	**8482.90**
mean	8849.72	8953.73	8904.87	9578.07	8848.39	8999.95	**8847.48**
std	**455.45**	458.18	457.43	466.44	455.73	461.88	455.96

Table 8. MSE of Real Test Dataset with MLR

(2) Support Vector Machine (SVM)

For our Support Vector Machine section, we used SVM with a linear kernel, with different coefficients C and linear SVM with ℓ1, ℓ2 penalty to make predictions. The cost function is also the MSEs of the test dataset. The models were built in Python and the results are shown in Table 9.

We defined four cases for our experiment. We wrote two methods using no penalty with different C values, which are based on the SVM from the *sklearn.svm* package. We also built one model using ℓ1 regularisation, and one other using ℓ2 regularisation, which are based on the *LinearSVR* package.

As we can see, the model SVM with ℓ2 has the best performance over our dataset, with the mean MSE resulting at 8748.56. Moreover, performance can be improved significantly when we include the ℓ1 and ℓ2 regularisation.

When we look at the methods that aren't implementing regularisation, the model with C=0.5 performs a little better than the model with C=1 based on our results. It is possible that when C gets smaller, the margin for SVM gets larger, and the model can tolerate more mistakes and prevent overfitting. However, the difference is comparatively tiny so it would be difficult and inaccurate to draw strong conclusions from this finding.

SVM				
	no penalty(C=1)	no penalty(C=0.5)	ℓ1	ℓ2
1	30381.36	30641.09	9627.07	**8751.27**
2	30431.64	30440.48	9728.88	**8997.94**
3	30821.96	30506.50	10267.86	**9285.55**
4	30403.80	30285.87	9019.69	**8653.61**
5	30745.40	30092.84	10180.00	**9090.80**
6	30631.81	31114.44	10087.30	**8510.11**
7	29786.03	30635.74	10309.98	**8268.16**
8	30690.16	30370.21	10170.78	**9186.37**
9	30551.37	30649.69	10598.82	**7954.18**
10	30463.11	30131.39	10614.39	**8787.59**
mean	30490.66	30486.82	10060.48	**8748.56**
std	**275.37**	283.63	459.36	398.57

Table 9. MSE of Real Test Dataset with SVM

(3) Multilayer Perceptron (MLP)

Using Multilayer Perceptron, we created four models: one with no penalty, and one each with ℓ1, ℓ2 and elastic net regularisation to build models on our normalised training data and make predictions on the test data. The input shape is 28, since we have 28 predictors. The activation for the hidden layer is 'sigmoid', and the output layer is 'linear' since our problem is a regression problem. In the hidden layer, we add the regularisations for ℓ1, ℓ2 and elastic net penalties. As for the weights updating process, we use SGD with learning rate (0.001). The cost function used is the MSEs of the test dataset against our predictions. The models were built in Python and the results are shown in Table 10.

Based on our results, we can see that MLP with elastic net regularisation has the best performance, with mean MSE is 11655.02. When we look closer at the individual runs, we see that the elastic net penalty method has the best result for five out of the ten runs. The standard deviation is also the smallest for this method, which means that it model performance is the most stable.

MLP				
	no penalty	ℓ1	ℓ2	ElasticNet
1	11986.38	10974.25	11816.85	**10167.91**
2	**11012.70**	12174.12	11703.05	11955.14
3	13030.97	13327.79	**11876.08**	12776.62
4	11804.42	13327.79	11898.28	**11707.81**
5	12908.75	**11077.05**	12725.19	12401.55
6	**9962.59**	10459.38	13657.69	12019.97
7	11197.13	12557.12	11659.15	**10914.67**
8	13080.07	12222.35	13614.82	**11471.63**
9	12151.27	12065.99	12856.02	**11340.41**
10	12029.24	11485.13	**10683.39**	11794.48
mean	11916.35	11967.10	12249.05	**11655.02**
std	939.04	919.37	893.93	**703.78**

Table 10. MSE of Real Test Dataset with MLP

3.4 Conclusion of Real Dataset

Comparison of the three learners: MLR, SVM and MLP

In this part, we compare the three learners together and observe the performance. With the Real dataset, we built 15 models to predict the energy use of the appliances in Watts per hour and for each model, we got the averaged MSE of test dataset for 10 runs. The cost function used to evaluate each model is the MSE of the test dataset, and we consider the best model to be the model with the lowest test MSE.

As we can see that MLR with elastic net regularisation and cross-validation (ElasticNet.CV) performs the best among 15 models. Within MLP also, elastic net regularisation outperforms no regularisation, lasso regularisation and ridge regularisation.

Unfortunately, we didn't get the model performance of elastic net regularisation for the SVM. We can see however, that SVM with regularisation performs far better than SVM with no penalty. Based on this, and after observing the performance of elastic net regularisation on the other two methods MLR and MLP, we would be very interested to learn how elastic net would perform on SVM.

We can also see that cross-validated methods outperform methods without cross-validation, which is consistent with what we'd expect, and with what we observed in the Boston models.

MSE of Real Test Dataset				
MLR	**MLR**	**Ridge**	**Lasso**	**ElasticNet**
	8849.72	8953.73	9578.07	8999.95
		Ridge.CV	**Lasso.CV**	**ElasticNet.CV**
		8904.87	8848.39	**8847.48**
SVM	**no penalty(C=1)**	**no penalty(C=0.5)**	**ℓ1**	**ℓ2**
	30490.66456	30486.82489	10060.47764	**8748.557631**
MLP	**no penalty**	**ℓ1**	**ℓ2**	**ElasticNet**
	11916.35127	11967.09571	12249.05234	**11655.0179**

Table 11. MSE of Real Test Dataset with All Algorithms

4. Conclusion

In conclusion, we investigated the performance of three statistical learning methods, Multiple Linear Regression (MLR), Support Vector Machines (SVM) and Multi-Layer Perceptron (MLP). For each of these, we incorporated three regularisation approaches, ℓ2, ℓ1 and elastic net regularisation where possible to see how this would affect our model's performance. For all MLR models, we included cross-validation for all models. Thus, we produced 15 models for each dataset.

Based on our results, MLR performs the best across both models. This makes sense, since both of our datasets are suitable for the regression problems, MLR is a better choice than other two learners. SVM and MLP are normally used in classification problems, so if we want to predict the classes for some datasets, we may choose SVM and MLP rather than MLR.

As for the cross validation, we used 10-fold cross-validation to prevent model overfitting. From our results, we can see that our models with cross-validation outperforms the models without cross-validation.

The elastic net penalty by design has the ability to combine the advantages of ℓ2 and ℓ1. This is reflected in our results, as most of our models are at their best point when using elastic net regularisation.

In terms of model interpretability, MLR is better than the other two algorithms, as the summary provides easy analysis of the effect of each of the variables and can be visualised easily where required. This makes it easy to understand the relationship between the response and predictors by viewing the coefficient values and their statistical relevance.

For speed, MLR appeared to be the most efficient. Both SVM and MLP took so long to run in R for the real dataset that we had to switch to Python. This is obviously not ideal if our dataset is large and R is our language of choice.

For both datasets, the MLR method appeared to come out as superior when predicting unknown cases. However, this is probably specific to the properties of each dataset, which are straightforward and appear to suit regression problems. Both datasets have very few subtleties that might suit an SVM or MLP method and probably consist of predominantly linear relationships between response and predictor variables. Whilst it is clear which method performs most effectively on each of our own two datasets, it would be incorrect to make any sort of conclusion about which methods would be best for a separate dataset based on this experiment alone. Further consideration must be given to the properties of a future dataset before selecting a method that would be able to model the unknown dataset most accurately.

5. References

[1] Candanedo, L., 2017. *Appliances energy prediction Data Set*, electronic dataset, University of Carolina, Irvine Institutional Repository, doi:

https://archive.ics.uci.edu/ml/datasets/Appliances+energy+prediction#.

[2] Friedman, J., Hastie, T. and Tibshirani, R., 2001. *The elements of statistical learning* (Vol. 1, No. 10). New York, NY, USA:: Springer series in statistics.