SOFTENG 755 Assignment 1

Junjie Zeng
Univeristy of Auckland
jzen188@aucklanduni.ac.nz

***Abstract*—This document is a report for SOFTENG 755, Bayesian Machine Learning, Assignment 1. Four dataset from different problem domain will be explored (*World Cup Final 2018, SH1 Traffic Volume, Occupancy Sensor Data and Landsat Satellite Data*). Machine learning methodologies will be applied to create best possible models in solving these problems.**

Keywords—Machine Learning, Regression, Classification

I. Introduction

In this report, we are going to conduct several data analysis tasks on four different data sets with the corresponding machine learning models, including regression and classification. A reproducible Python project will be delivered separately. The following is the description of four data sets and related problems from them:

A. World Cup Final 2018

This data set contains basic information about a match held for World Cup Final 2018 in Russia, including locations, dates, time, statistics of the match, and the scores. One characteristic of this data sets is that it is relatively small. We are interested in predicting the match score. This can be classified as regression problem. A learner, built by supervised learning, is construct to predict score for each match; and, we are using the predicted score and the actual score to measure model performance. Nevertheless, since the statistics information is obtained after each match, we can't just directly use the original features to construct the final predictive models. Feature selection/extraction is required, in advanced, to obtain/summarise historical match statistics.

B. SH1 Traffic Volume

Traffic congestion is an open problem because the underlying, complex spatiotemporal dependencies are difficult to model. Spatiotemporal dependencies raise when data are collected across time as well as spatial. In this data set, historical data of traffic volume measurements along 45 segments (different spatial) of SH1 over a period of time (different time). Basic feature extraction by the sliding window technique has been performed. The window size is 10 (including time stamps t-9, … , t). Thus, each data instance contains all 45 segments' traffic volume across 10 periods. We are interested in predicting the volume of Segment 23 at time stamp t+1. This is another regression problem. In this challenge, a learner, built by supervised learning, is construct to predict future traffic volume, conditions by relatively high dimensions of features of spatiotemporal dependencies. Based on exploratory data analysis we have done, most of them are highly positive correlated. This is reasonable in such domain and expect to be a reasonable outcome. For instance, at any given point of time i in t (t-9 <= i <= t), segment k is correlated with segment k+1 or segment k-1. If segment k suffered high traffic volume, its closed segments are most likely suffering the same issue. Meanwhile, at any given point of segment k, traffic volume in time i is similar to traffic volume in time i+1 or time i-1 since a high traffic volume will be resolved in a short period).

C. Occupancy Sensor

In smart home application, one of the frequent problem is whether there are occupancy detected. The Smart Home control system can use the occupancy information to switch on or off a series of appliances. Data set is provided with data collected time (particular date per minute), and the related light, temperature, humidity and CO2 measurements in an office room. The focus is to use this information to inference whether occupancy detected. Human labelled occupancy label is provided (0, as no occupancy; 1, as occupancy observed) in assisting the model training. This is a typical binary classification problem. One key highlight is the data is large imbalance. Based on our exploratory data analysis, 79% records are classified as no occupancy (Y=0). A dumb model in imbalance class case may result to be high accuracy but always predict the major class. In such case, simple measurement like accuracy will not be appropriate in measuring model performance.

D. Landsat Satellite

This data set consists of the multi-spectral values of pixels in 3x3 neighbourhoods in a satellite image, and the label information associated with the central pixel in each neighbourhood. This data set was generated from Landsat Multi-Spectral Scanner image data. Basic pre-processing has been done to ensure all features using the same 8-bit code (0 indicates black while 255 indicates white). Each pixel is combined by 4 layers of spectral bans (green, red, near infrared 1, near infrared 2). Each layer is one feature and we are using 4 features (the above 4 bands) to represent 1 pixel. In total, there are 9 pixels for each satellite image. So, there will be 36 features. We are particularly interested in the central pixel, which can be represented by linear combination of 4 features (16, 17, 18, 19). By using this information, we need to solve a multi-class classification problem to indicate what objects shown in the central pixel. Since the object could have some boundary captured by the nearest pixel, we believe by considering other pixel features besides central pixel (features No 16-19), we may train a better model to predict object class.

II. Methodologies

In this report, we are limited our choices of supervised learning models within the below methods. Since they are well-known traditional methods, introduction of these methods will only include these aspects:

1. Problem of this method can be handled
2. Advantage and disadvantage of this method
3. What to maximize/minimize to coverage model
4. Parameters space we used for parameter tuning

A. *Ordinary regression*

Ordinary regression[5] is also called linear regression. It is originally designed for continuously target variable that has Gaussian distribution. Generalized linear regression can handle classification problem and further different kinds of target variable's distribution in regression problem. If assumption is satisfied, ordinary regression can produce quite accurate estimate of the coefficient of exploratory features. However, linear assumption is quite hard to satisfy in reality; also, multi-collinearity will violate the assumption and result as incorrect estimation, though this problem is not serious when machine learning goal is for predictive purpose. Ordinary regression is trying to minimalize the squared error function as below[5]:

$$\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 \tag{1}$$

In our case, we only tune whether regression includes intercept.

B. *Ridge regression*

Ridge regression[7] is an extend version of ordinary regression, which solves the least-squares problem while using regularization to further constrain the resulting solution. It can support generalization (handling both classification and non-Gaussian continuous target variable). Though it is not proper to use its for estimating population effect from exploratory feature, ridge regression uses regularization to give more weight to variables that will improve model performance. Therefore, it can be more or less release some of issues from multi-collinearity. Ridge regression is trying to minimalize the original formula (1) with extra penalized term[7]:

$$\sum_{i=1}^{n}(y_i - z_i^T\beta)^2 + \lambda \sum_{j=1}^{p}\beta_j^2 \tag{2}$$

In our case, we tune whether regression includes intercept and alpha from [-5, 0] log10 space. Alpha represents regularization strength. Feature scaling could be helpful in model fitting and will be introduced in future chapter.

C. *Perceptron*

Perceptron[11] is an algorithm for supervised learning of binary classifiers (functions that can decide whether an input, represented by a vector of numbers, belongs to some specific class or not). It is a type of linear classifier. It is similar to logistics regression. It tries to solve this equation, called perceptron criterion[11]

$$\frac{1}{N}\sum_{i=1}^{N}\max(0, -y_i w \cdot x_i) \tag{3}$$

In our case, we tune whether regression includes intercept and alpha from [-8, 0] log10 space. Alpha represents regularization strength. Feature scaling could be helpful as well. For multiple class case (LandSat data), multiple binary perceptron classifier will be trained for each class and choose the maximal likelihood one for the predicted class. This strategy is known as one vs all. Other methods we introduced later will use the same strategy for multiple class case if there is not specified.

D. *Support Vector Machine (SVM)*

Support vector machine[6] can handle classification (SVC) and regression (SVR) problem. In geometry, a maximum-margin hyperplane is a hyperplane which separates two 'clouds' of points and is at equal distance from the two. The margin between the hyperplane and the clouds is maximal. Support vector classifier is using the same technique to discriminative different classes in hyperspace by maximizing the below formula[6]:

$$\max(0, 1 - y_i(\vec{w} \cdot \vec{x_i} - b)) \tag{4}$$

Kernel trick techniques are used widely in support vector machine in dealing with non-linear classification problem. Common kernel includes *linear* (SVM converts as logistics regression), *sigmoid* (SVM converts as 1-layer neural network), *polynomial* and *radial basis function* (RBF). In our case, we only considered linear and RBF kernel. The former handles linearity while the latter captures non-linearity quite well in hyperspace. Both kernel is seeking optimal parameter, C, in range [-3, 3] log10 space. Feature scaling could be helpful as well.

E. *Decision trees*

Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items. Common algorithm choices include ID3, C4.5, CART (handle both classification and regression), etc. In this report, we are using C4.5 tree generation algorithm[9]. Features to select for splitting are determined by calculating Entropy to maximize information gain. Entropy formula is below[2]:

$$H(X) = -\sum_{i=1}^{n} p(x_i)\log p(x_i) \tag{5}$$

Though Decision tree is greedy search algorithm, it has some distinct advantage compared to other methods we used in this report: simple to understand and interpret, robust against multi-collinearity, no normalization/standardization pre-processing requirement, robust of missing value and outlier, etc. Tree depth is commonly used to control the tree size and model complexity to get a balancing performance between train set and cross validation set. Therefore, we are using (1, 3, 5, 7, 9, 11) as our choice of depth for parameter space.

F. *K-Nearest Neighbour*

k-nearest neighbours algorithm (k-NN) is a non-parametric method used for classification and regression[3]. In contrast to other methods choosing variables in a column-wise way, K-NN find the optimal k closest training instances in a row-wise way. It is therefore a type of instance-based learning, where the function (see below) is only approximately locally to find the most frequent class. In this practice, we are using KD-trees algorithm to optimal k value within [2, 30] instances. Notice that k-NN is also sensitive to normalization and has weaker performance in high dimensional case. The latter has been referred as curse of dimensionality. Principal component analysis could be helpful in improving k-NN method's performance.

G. *Naïve Bayes*

The Naive Bayes classification method are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong but naive independence assumptions between the features. It is essentially to solve the below formula based on maximum a posterior (MAP)[14]:

$$\underset{k \in \{1, \dots, K\}}{argmax}\, p(C_K) \prod_{i=1}^{n} p(x_i|C_k) \qquad (6)$$

Since, in our classification data (LandSat and Occupancy) most of the exploratory features are continuous-based, we will use Gaussian naïve Bayes for the above probability function. We are choosing two strategy for prior choices: proportion of target variable from observed the whole training data (without K-fold cross validation) and uniformed distribution (class is uniform distributed).

III. PRE-PROCESSING

From chapter I Introduction part, we understand target variables and potential challenge in fitting model for four data sets provided. Before going through parameter tuning to find best model, we need to do some pre-processing to ensure we adopt the right methods to find the optimal model.

A. *Train Test Split*

Firstly, all data sets provided need to adopt *train-test split*. Test set or hold out set, as guided, need to be 10% of the original data set. One key rule for test set is that never train on test data. We will hold test set out without using it to train or involve into any parameter tuning step to ensure models can be perform on test set to find generalized error. In particular, two of the data sets we had are classification problem, to ensure the test set is good represented of the original data, we use stratify strategy to ensure train set and test set have the same proportion of target variable (both binary and multiple class adopted).

B. *K-fold Cross Validation*

In modelling fitting stage, we need to grid search (test and trial every parameters combination given) to find the best parameter combination for any given methods. We introduced K-fold cross validation[8] to help us find the optimal parameters set.

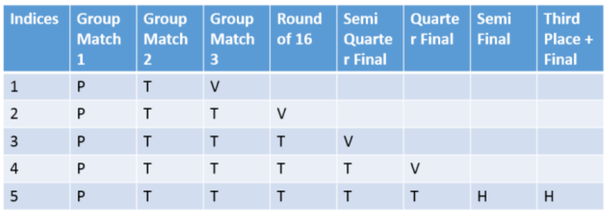
In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k − 1 subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data. The k results can then be averaged to produce a single estimation. The advantage of this method over other cross validation techniques (Bootstrapping, leave-1-out cross validation, etc.)[8] is that all observations are used for both training and validation, and each observation is used for validation exactly once. We are using 5-fold cross validation here based on rule of thumb.

In particular, two of the data sets we had are classification problem, to ensure the validation set is good represented of the train data, we use stratify strategy to ensure kth validation set and (k-1) train sets have the same proportion of target variable (both binary and multiple class adopted).

A. *Time Series Split*

There is requiring special treatment to WorldCup 2018 data. The fast and powerful methods that we rely on in machine learning, such as using train-test splits and k-fold cross validation, do not work in the case of time series data. This is because they ignore the temporal components inherent in the problem. This is because they assume that there is no relationship between the observations, that each observation is independent. This is not true of time series data, where the time dimension of observations means that we cannot randomly split them into groups. Instead, we must split data up and respect the temporal order in which values were observed.

Time series split repeats the process of splitting the time series into train and test/validation sets multiple times. The below figure illustrate the whole test and validation process.



Indices	Group Match 1	Group Match 2	Group Match 3	Round of 16	Semi Quarter Final	Quarter Final	Semi Final	Third Place + Final
1	P	T	V					
2	P	T	T	V				
3	P	T	T	T	V			
4	P	T	T	T	T	V		
5	P	T	T	T	T	T	H	H

Fig. 1. Table to indicate Train(T), Validation (V) and Hold Out (H) set

The original data has provided 64 matches. Details as below:

1. Group Match 1 Day: 32 parties have 16 matches
2. Group Match 2 Day: 32 parties have 16 matches
3. Group Match 3 Day: 32 parties have 16 matches
4. Rounds of 16: 16 parties have 8 matches
5. Semi Quarter Final: 8 parties have 4 matches
6. Quarter Final: 4 parties have 2 matches
7. Semi Final
8. Third Place + Final: 4 parties have 2 matches

We need to derive features that is statistic from the past history of matches. We preserve (P) the 1st match since we don't try to predict 1st match as we don't have statistics before 1st match! From 2nd match will use each party 1st match statistics as training data (T) to predict their result in 2nd match to fit model. After model fitted, we combine Group Match Day 1 and Group Match Day 2 data to apply onto the existing model to predict 3rd match result as validation (V). This is the 1st indices of cross validation in time series split. In the 2nd indices of cross validation in time series split, we combine Group Match Day 1~2 statistics to build model predict Group Match Day 3 result, then apply the same model on Group Match Day 1~3 statistics to predict Rounds of 16 matches result. There are 16 matches preserved in Group Match Day 1. We have only 48 matches to do train, test, validation to find the optimal model. As required, we need 10% of training data to hold out. Therefore, last 4 matches (Semi Final, Third Place and Final) are held out. For further details of how we "combine" statistics from multiple instances, please see future chapter – Feature Engineering, Rolling Windows. For details of how to prepare train, test, validation data, please refer to the coding project attached.

C. Standardization and Normalization

Feature transformation is a method used to transform the range of independent variables or features of data. Commonly, it includes standardization and normalization.

1) Scaling: Normally, scaling applied on column wise. The below two methods are widely applied as the preprocessing step for many machine learning models. For instance, Ridge regression will assign further weight on irrelevant features if there are no scaling pre-performed.

a) Min-Max: it transforms the data set from one range to another. Commonly, the re-mapping range is [0, 1]. This can be done by the following function:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{7}$$

a) Standardization: In machine learning, we can handle various types of data, e.g. audio signals and pixel values for image data, and this data can include multiple dimensions. Feature standardization makes the values of each feature in the data have zero-mean (when subtracting the mean in the numerator) and unit-variance. Function is applied to features*:*

$$x' = \frac{x - \mu}{\sigma} \tag{8}$$

2) Normalization: normalization applied mostly on row wise data. In our case, both landsat and traffic data have instance having the same unit, though they have different variance. Standardization may not be helpful for such cases compares features that havee different unit and big different variance. In Landsat data, instances are collection of pixels from 3D satellite image (3 X 3 X 4). Image has been flatten to 1D array. All features have the same range [0, 255]. In other example, Traffic congestion, each instance represents, at certain time range, all road segments' traffic volume across 10 windows period.

b) MinMax: in image processing task, histogram equalization is a method in image processing of contrast adjustment. This method usually increases the global contrast of many images, especially when the usable data of the image is represented by close contrast values. To perform histogram equliazation, it is applying min-max scaling formula on row wise. In Landsat data, we want to test whether histogram equalization can help predicter better identified features to distinguish different objects to improve accuracy.

c) L1 Norm: Another option that is widely used in machine-learning is to scale the components of a feature vector such that the complete vector has length one. In Traffic Congestion case, supposed in collecting two instances in different time range, even a same amount of traffic volume could be largely different. For instance, instance A and B represent road segment 1 at T-1 has 100 traffic volume. However, instance A have other features indicates that 100 traffic volume is quite small while instance B would give a different conclusion. Normalizing two instance can help us estimate the actual matitude of the traffic volumes by considering other closed time and closed road segment. This may potentially help us better estimating the traffic congestion in future. L1 Norm usually divides each features by the Euclidean length of the vector:

$$x' = \frac{x}{||x||} \tag{9}$$

D. Feature Engineering

1) Dimension Reduction: dimension reduction is the process of reducing the number of random variables under consideration (Roweis, S. T.; Saul, L. K. (2000)) by obtaining a set of principal variables. It can be divided into feature selection and feature extraction (Pudil, P.; Novovičová, J. (1998)). Principal component analysis (PCA) is a common technique in dimension reduction. It uses an orthogonal transformation to covert correlated variables into a set of values of linearly independent variables called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components.

One of the great benefit of PCA is to reduce time and complexity in high dimensional data training. The first couple of principal components (PCs) could keep as much variation as possible. Since the components are independent, we are also get correct estimation in regression setting hassle free from multi-collinearity issue. Especially, in image data task, PCA can reduce noise (bottom level PCs) and preserve signal (high level PCs) only[1]. Due to this benefit, we will apply principal component pre-processing in Landsat data and Traffic Congestion data to check whether selecting only high relevant (high variance) will improve model performance. The previous discussed feature transformation is also a must-have for PCA. We can also check how different feature transformation (standardization/normalization) affect PCA and final model performance.

2) K Best Selection: K-best selection is a technique to select features according to the k highest scores. It is univariate test, which means each feature perform an univariate test with target variable to collect test score. Scores can be calculated by different test methods, including correlation, CHI2, Mutual Information, etc. In our case, we are considering to use F-test in both regression and classification case since all attributes are continue range.

a) Classification: for classification case, target variable is category. This would be used as the factor to perform Analysis of variance (ANOVA) F test to each exploratory attributes one by one and collect F-value. The larger the F-value the greater dispersion, which suggests an exploratory variable will have more significant different means across different level of target variable. This is normally indicating a variable having a predictive power. The drawback is the test is univariate test. The predictive power may be caused by the other higher correlated variable. K-best strategy is introduce to pick up the highest relevant variables by ranking, though it still doesn't guarantee the top K variables will result in the optimal model performance. Other factors need to be considered as well (transformation, method, parameters, etc).

d) Regression: for regression case, it is similar to the above classification case. Instead, linear regression between target variable and each exploratory variable is fitted one by one. F score is collected and ranked. It has the same drawback as above.

3) Rolling Windows: previous two feature engineering techniques focus on feature selection and reduction. For WorldCup and Occupancy Sensor data, limited feature we have. We need to extract more effective features out of the existing data. Since these data contains time series feature, we can use rolling windows to create new features that contain multiple time spot informaiton in an aggregation way.

Suppose that you have data for all periods [1, T] in the sample. In total, we will have T instances. For each instance, we also extract data from previous instance, given a window size is m, we can aggregated these m instance into 1 instance by applying max(), sum(), average(), etc. functions.

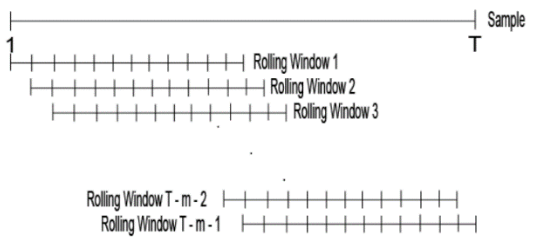


Fig. 2. Example of rolling window methods

In WorldCup 2018 data, the original data has 64 matches results. All features are generated after the match. Our interest is to predict each match's total scores. We need to derive features that is statistic from the past history. In previous chapter Time Series Split, we understand that there will be 64 matches. A team could have [3, 6] rounds. We preserve the first match as the training data (we don't try to predict 1st match since we don't have statistics before 1st match!) 2nd match will use each party 1st match statistics to predict their result in 2nd match. Rolling window applied since 3rd match. From 3rd match, we roll 1st and 2nd match statistics by average to calculate each party performance to create our training data. This iteration continue. This means, in the final match, two party will roll up their group (3 matches), round of 16, quarter final, semi final statistics.

In Occupancy sensor case, we are provided mainly temperature data at a given minute. We assume, there will be a big chance to the temperature when people occupy the room. Rolling window applied to calculate difference of a current instance to the previous k instance's temperature. For instances do not have any previous instances, we said there are no difference.

Further details of these pre-processing please refer to the actual coding project.

IV. PERFORMANCE METRICS

In this chapter, we defined performance metrics. There are two validation steps we need to perform. Internal cross validation is focusing on, given a method/algorithm and parameters' space, we find the optimal parameters across K-fold cross validation. External Hold out is comparing each optimal model from different methods on hold out set to choose the best model. Different requirement is therefore set up to suit two different needs.

A. *Internal Cross Validation*

a) Mean Absolute Error (MAE): MAE is often called as median absolute deviation (MAD). Formula is shown below. It is one of the traditional robust estimation of regression validation error in model selection[13]. The purpose of internal cross validation stage is to tune optimal parmeters for each methods. We are using MAE to avoid outlier affecting our parameter selection.

b) F1: F1 score (also F-score or F-measure) is one of the traditional robust estimate to check model performance in calssification problem. It can solve multiple class, imbalance class issues. F1 score is essentially the harmonic mean of precision and recall. Precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances, while recall (also known as sensitivity) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Variation of F1 score includes F1-micro, F1-macro and F1-weighted. In our case, we considered F1-micro, which calculates the true positive and false positive globally.

B. *External HoldOut*

a) R2: Root Mean Square Error (RMSE) is a common metrics to measure regression error, though it may be affected by outlier. Therefore, we didn't use it for parameters tuning. In external hold out case, we hope to use a widely usage score for benchmark comparison. In our case, we choose to use R-squared, which is proportional RMSE and easier to interpred.

b) Accuracy: Accuracy is a common metrics to measure classification error, though it may be affected by imbalance class. In external hold out case, we hope to use a widely usage score for benchmark comparison. Accuracy is used since we have already used F1 score to choose model that is not going to bias to certain classes in multiple class and imbalance class situation. Accuracy can give us a more inituitive view how good our model is.

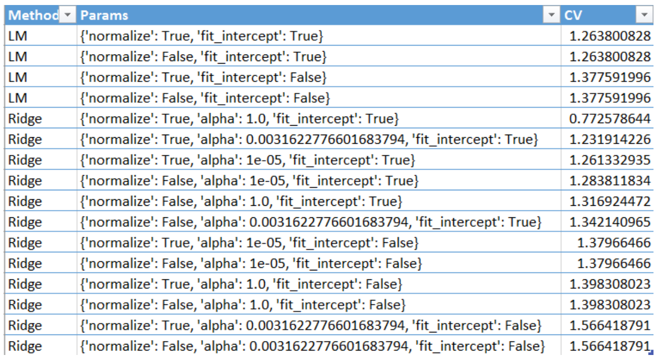
V. RESULT

The implementation is using Python 2.7 and machine learning framework Sklearn. I am running on my laptop which has . Each data set will run through below process and we treated methods of transformation as one of tuning parameter as well. For method with different transformation (row/column) setting will use K-fold cross validation (in World Cup 2018 data, Time Series Split cross validation used instead) to find optimal parameter in the given parameter space. Then each optimal model with optimal parameter by certain transformation setting is applied onto hold out test set to get best model.

A. *World Cup Final 2018*

Through cross validation, we identified, firstly, fitting intercept is important. If the other parameters as the same setting, fitting intercept is always bringing better cross validation (CV) result (MAE used, the less the better).

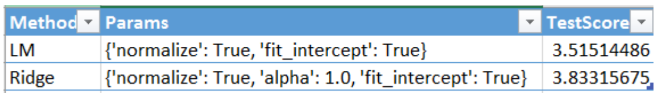
Normalization made small difference to ordinary regression (LM below) while it is important in Ridge regression (Ridge below). Given the other parameter same setting, normalized data in ridge regression has better CV MAE. Decreasing alpha is tend to weaken CV result (larger MAE).



Method	Params	CV
LM	{'normalize': True, 'fit_intercept': True}	1.263800828
LM	{'normalize': False, 'fit_intercept': True}	1.263800828
LM	{'normalize': True, 'fit_intercept': False}	1.377591996
LM	{'normalize': False, 'fit_intercept': False}	1.377591996
Ridge	{'normalize': True, 'alpha': 1.0, 'fit_intercept': True}	0.772578644
Ridge	{'normalize': True, 'alpha': 0.0031622776601683794, 'fit_intercept': True}	1.231914226
Ridge	{'normalize': True, 'alpha': 1e-05, 'fit_intercept': True}	1.261332935
Ridge	{'normalize': False, 'alpha': 1e-05, 'fit_intercept': True}	1.283811834
Ridge	{'normalize': False, 'alpha': 1.0, 'fit_intercept': True}	1.316924472
Ridge	{'normalize': False, 'alpha': 0.0031622776601683794, 'fit_intercept': True}	1.342140965
Ridge	{'normalize': True, 'alpha': 1e-05, 'fit_intercept': False}	1.37966466
Ridge	{'normalize': False, 'alpha': 1e-05, 'fit_intercept': False}	1.37966466
Ridge	{'normalize': True, 'alpha': 1.0, 'fit_intercept': False}	1.398308023
Ridge	{'normalize': False, 'alpha': 1.0, 'fit_intercept': False}	1.398308023
Ridge	{'normalize': True, 'alpha': 0.0031622776601683794, 'fit_intercept': False}	1.566418791
Ridge	{'normalize': False, 'alpha': 0.0031622776601683794, 'fit_intercept': False}	1.566418791

Fig. 3. Cross Validation set result of World Cup Final 2018

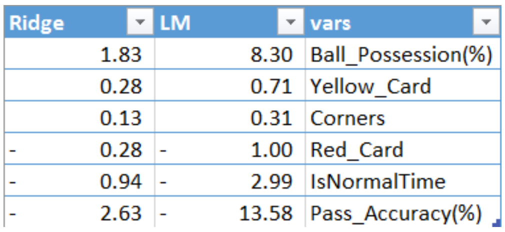
We choose each method's best parameter model to test on hold out set. Though optimal model in Ridge method has less CV MAE than LM method, in test case, LM method won slightly. The difference resulted since we have only small number of test case, introducing lots of variantion.



Method	Params	TestScore
LM	{'normalize': True, 'fit_intercept': True}	3.51514486
Ridge	{'normalize': True, 'alpha': 1.0, 'fit_intercept': True}	3.83315675

Fig. 4. Hold-Out set result of World Cup Final 2018

Looking at the coefficient of two final models' comparison (top 3 and bottom 3 coefficient shown only), we get similar conclusion: the larger the average *ball_possession(%)* between two teams, the higher chance the score is large; the larger difference between average *pass_accuracy(%)* between two teams, the higher chance the score is small. The former (*ball possession*) is representing control ability between teams while the latter (*pass_accuracy*) is reflecting defence level. For further details of performance of each parameters by methods, please check Append A.

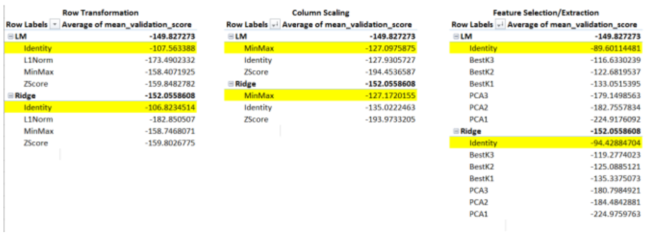


Ridge	LM	vars
1.83	8.30	Ball_Possession(%)
0.28	0.71	Yellow_Card
0.13	0.31	Corners
- 0.28	- 1.00	Red_Card
- 0.94	- 2.99	IsNormalTime
- 2.63	- 13.58	Pass_Accuracy(%)

Fig. 5. Comparison of Ridge regression and Ordinary (LM) regression coefficient's difference. Notice, since ridge applied regularization penalty, coefficient estimation is biased but without large variance that affected by multi-collinearity. LM result has larger dispension instead.

B. SH1 Traffic Volume

Through cross validation, we test whether difference choices of row transformation, column scaling, feature extraction increases method's performance. Average Negative MAE is reported. The larger the better CV result. We identified, on average, only min-max scaling returned better outcome.



Row Transformation

Row Labels	Average of mean_validation_score
LM	-149.827273
Identity	-107.563388
L1Norm	-173.4902332
MinMax	-158.4071925
ZScore	-159.8482782
Ridge	-152.0558608
Identity	-106.8234514
L1Norm	-182.850507
MinMax	-158.7468071
ZScore	-159.8026775

Column Scaling

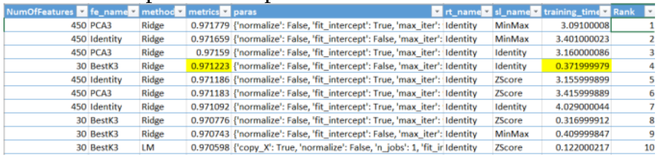
Row Labels	Average of mean_validation_score
LM	-149.827273
MinMax	-127.0975875
Identity	-127.9305727
ZScore	-194.4536587
Ridge	-152.0558608
MinMax	-127.1720155
Identity	-135.0222463
ZScore	-193.9733205

Feature Selection/Extraction

Row Labels	Average of mean_validation_score
LM	-149.827273
Identity	-89.60114481
BestK3	-116.6330239
BestK2	-122.6819537
BestK1	-133.0515395
PCA3	-179.1498563
PCA2	-182.7557834
PCA1	-224.9176092
Ridge	-152.0558608
Identity	-94.42884704
BestK3	-119.2774023
BestK2	-125.0885121
BestK1	-135.3375073
PCA3	-180.7984921
PCA2	-184.4842881
PCA1	-224.9759763

Fig. 6. Average cross-validation to test whether row transformation, column scaling or feature selection/extraction have effect on model performance.

For each optimal model, we applied test set to get hold out error rate. Top 10 result provied.



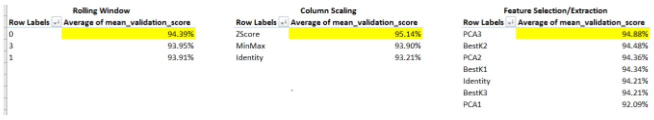
NumOfFeatures	fe_name	method	metrics	paras	rt_name	sl_name	training_time	Rank
450	PCA3	Ridge	0.971779	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	MinMax	3.09100008	1
450	Identity	Ridge	0.971659	{'normalize': False, 'fit_intercept': False, 'max_iter':	Identity	MinMax	3.401000023	2
450	PCA3	Ridge	0.97159	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	Identity	3.160000086	3
30	BestK3	Ridge	0.971223	{'normalize': False, 'fit_intercept': False, 'max_iter':	Identity	Identity	0.371999979	4
450	Identity	Ridge	0.971186	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	ZScore	3.155999899	5
450	PCA3	Ridge	0.971183	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	ZScore	3.415999889	6
450	Identity	Ridge	0.971092	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	Identity	4.029000044	7
30	BestK3	Ridge	0.970776	{'normalize': False, 'fit_intercept': True, 'max_iter':	Identity	ZScore	0.316999912	8
30	BestK3	Ridge	0.970743	{'normalize': False, 'fit_intercept': False, 'max_iter':	Identity	MinMax	0.409999847	9
30	BestK3	LM	0.970598	{'copy_X': True, 'normalize': False, 'n_jobs': 1, 'fit_ir	Identity	ZScore	0.122000217	10

Fig. 7. Hold-Out set result of Traffic Congestion

The best model comes from ridge regression with no row normalization, min-max scaling and full PCA transformation. Though PCA didn't reduce any column, it reduces multi-collinearity and win slightly of the 2nd model which has the same setting but no PCA applied). We also notice that only picking top 30 relevant features, we get a similar performance model (4th) but only need 1/10 training time. For further details of performance of each parameters by methods, please check Append B.

C. Occupancy Sensor

In smart home application, looking through the CV result, we found, on average:



Rolling Window

Row Labels	Average of mean_validation_score
0	94.39%
3	93.95%
1	93.91%

Column Scaling

Row Labels	Average of mean_validation_score
ZScore	95.14%
MinMax	93.90%
Identity	93.21%

Feature Selection/Extraction

Row Labels	Average of mean_validation_score
PCA3	94.88%
BestK2	94.48%
PCA2	94.36%
BestK1	94.34%
Identity	94.21%
BestK3	94.21%
PCA1	92.09%

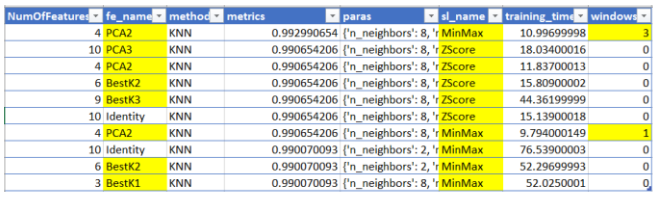
Fig. 8. Average cross-validation to test whether row transformation, column scaling or feature selection/extraction have effect on model performance.

1. **Rolling Windows (0/1/3)**: We are treating original data as rolling windows as 0 special case. Result doesn't suggest looking previous statistics and calculate difference to the current data will result a model improvement.

2. **Column Scaling**: Standardization and Mini-Max scaler both achieves better result than non-scaling.

3. **Features Selection/Extraction**: after PCA transformation, we expected signal and noisy extracted. Multiple metrics (CO2, temperature, humidity, etc.) will therefore combine by linear function to achieve better result. The result admitted our hypothesis. Notice, PCA2 is using top 95% variance, which is just using 4 columns only and achieving similar performance as PCA3 full transformed column kept. Best K selection at low number setting (3, 6 features respectively) is working better than no feature selection/extraction applied.

For each optimal model, we applied test set to get hold out error rate. Top 10 result provied.



NumOfFeatures	fe_name	method	metrics	paras	sl_name	training_time	windows
4	PCA2	KNN	0.992990654	{'n_neighbors': 8, 'r	MinMax	10.99699998	3
10	PCA3	KNN	0.990654206	{'n_neighbors': 8, 'r	ZScore	18.03400016	0
4	PCA2	KNN	0.990654206	{'n_neighbors': 8, 'r	ZScore	11.83700013	0
6	BestK2	KNN	0.990654206	{'n_neighbors': 8, 'r	ZScore	15.80900002	0
9	BestK3	KNN	0.990654206	{'n_neighbors': 8, 'r	ZScore	44.36199999	0
10	Identity	KNN	0.990654206	{'n_neighbors': 8, 'r	ZScore	15.13900018	0
4	PCA2	KNN	0.990654206	{'n_neighbors': 8, 'r	MinMax	9.794000149	1
10	Identity	KNN	0.990070093	{'n_neighbors': 2, 'r	MinMax	76.53900003	0
6	BestK2	KNN	0.990070093	{'n_neighbors': 2, 'r	MinMax	52.29699993	0
3	BestK1	KNN	0.990070093	{'n_neighbors': 8, 'r	MinMax	52.0250001	0

Fig. 9. Hold-Out set result of Traffic Congestion

The best model comes from K-NN model with n_neighhours as 8 instances, with rolling windows previous 3 instances, min-max scaling and PCA first 95% variance (4 out of 10). As highlighted in figure 9, the proposed transformation has been shown in top rank in hold-out test set result. Scaling is working particularly good in this data set; unlike other previous case, different variance/unit variables used as predicting feature. Feature scaling play an important roles in giving fair comparison for model selection. Since Support Vector Machine is slow to fit, and is complexity-wise approximately: O(n_samples^2 * n_features). We didn't run the same process to find optimal parameter set for SVM. We are using the best model above to tune SVM. The result is only 98.75% accuracy in hold-out set. For further details of performance of each parameters by methods, please check Append C.

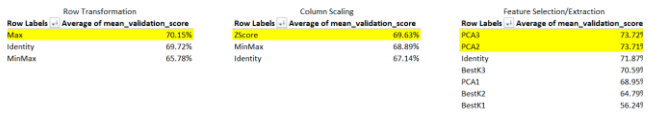
D. *Landsat Satellite*

In LandSat classification, looking through the CV result, we found, on average across all methods:

1. **Row Transformation**: Dividing maximal value (255) per row result a model performance improvement; though histogram equalization is famous in image processing task, it is not achieving better result than non-transformation.

2. **Column Scaling**: Standardization and Mini-Max scaler both achieves better result than non-scaling.

3. **Features Selection/Extraction**: after PCA transformation, we expected signal and noisy extracted. Multiple bands will therefore combine by linear function to achieve better result. The result admitted our hypothesis. Notice, PCA2 is using top 95% variance, which is just using 6 columns only and achieving similar performance as PCA3 full transformed column kept. Best K selection is working well in Traffic congestion case but it is working bad in this case.



Row Transformation

Row Labels	Average of mean_validation_score
Max	70.15%
Identity	69.72%
MinMax	65.78%

Column Scaling

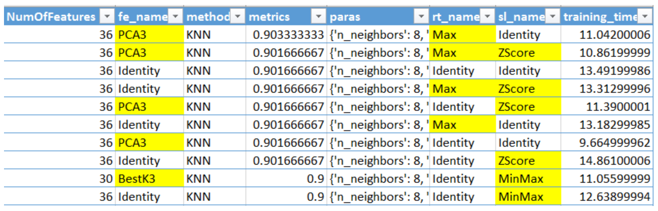
Row Labels	Average of mean_validation_score
ZScore	69.63%
MinMax	68.89%
Identity	67.14%

Feature Selection/Extraction

Row Labels	Average of mean_validation_score
PCA3	73.72
PCA2	73.71
Identity	71.87
BestK3	70.59
PCA1	68.95
BestK2	64.79
BestK1	56.24

Fig. 10. Average cross-validation to test whether row transformation, column scaling or feature selection/extraction have effect on model performance.

For each optimal model, we applied test set to get hold out error rate. Top 10 result provied.



NumOfFeatures	fe_name	method	metrics	paras	rt_name	sl_name	training_time
36	PCA3	KNN	0.903333333	{'n_neighbors': 8, '	Max	Identity	11.04200006
36	PCA3	KNN	0.901666667	{'n_neighbors': 8, '	Max	ZScore	10.86199999
36	Identity	KNN	0.901666667	{'n_neighbors': 8, '	Identity	Identity	13.49199986
36	Identity	KNN	0.901666667	{'n_neighbors': 8, '	Max	ZScore	13.31299996
36	PCA3	KNN	0.901666667	{'n_neighbors': 8, '	Identity	ZScore	11.3900001
36	Identity	KNN	0.901666667	{'n_neighbors': 8, '	Max	Identity	13.18299985
36	PCA3	KNN	0.901666667	{'n_neighbors': 8, '	Identity	Identity	9.664999962
36	Identity	KNN	0.901666667	{'n_neighbors': 8, '	Identity	ZScore	14.86100006
30	BestK3	KNN	0.9	{'n_neighbors': 8, '	Identity	MinMax	11.05599999
36	Identity	KNN	0.9	{'n_neighbors': 8, '	Identity	MinMax	12.63899994

Fig. 11. Hold-Out set result of LandSat Multiple Classification

The best model comes from K-NN model with n_neighhours as 8 instances, with row max transformation, standardization scaling and PCA full variance columns (36). As highlighted in figure 11, the proposed transformation has been shown in top rank in hold-out test set result. We didn't run the same process to find optimal parameter set for SVM. We are using the best model above to tune SVM. The result is only 88.3% accuracy in hold-out set. For further details of performance of each parameters by methods, please check Append D.

VI. CONCLUSION

In this report, 4 data sets with different characteristics have been explored. We proposed a pipeline of process to find best model in test set. We admitted some limitation existed and we would like to give more explanation in this chapter.

A. *Performance Definition*

We adopt some of the good practice in machine learning literature for our performance definition (MAE to avoid outlier in regression, F1 score to avoid imbalance issue in classification). However, in practice, subject matter knowledge is more important to help us define what is a good model.

In Occupancy case, we consider higher F1 score is better than accuracy since it is considering both precision and recall. In reality, algorithm need to have some more weighting on precision or recall-to be sensitive to convert more scenario or to be more precise to ensure no weird closing off or tuning on of the smart home application. This is applied to Landsat identification as well. One may prefer higher accuracy in detection certain objects/classes rather than an ideal learner that predict every class well. Cost weight matrix can be used to help us better define metrics to satisfy the practical need. Also, it may be more making sense, in occupancy sensor case, of testing continuous time instead of treating each minute independently like our case, which may require a whole different methodology instead of generic machine learning algorithm.

B. *Optimal Parameter Set*

Parameters for each method is covert large enough parameter space to illustrate how our process to find optimal model. We are suffering two issues this time that explain why we are not pursuing higher performance model:

1. Searching optimal parameter combination in hyper-parameter space is complicated and time consuming. One may result to get a local optimal parameters set. Evolution technique like genetic algorithm can be used to help us locate global optimal parameter set but it is out-of-scope and required more time and computational power.

2. Better estimation of error to reach generalized error. To save our time in this practice, we are using K-fold cross validation. It is fast to get reasonable good estimate of

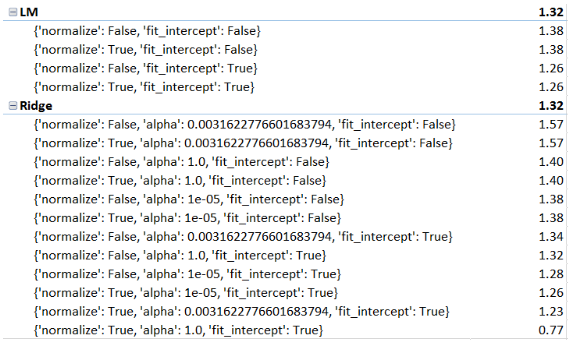
validation error but it is also high variance. Repeated K-fold cross validation, repeating > 30, can reduce this issue. Bootstrapping, particular its variation, .632+ Bootstrapping [4], is more preferred if we want to pursue best model. Bootstrapping can help us better estimate distribution of error. It is more time consuming, though.

ACKNOWLEDGMENT

Thanks for reading my long report. I have attached my coding project, which can allow marker re-produce the output. Not all information is provided; only most relevant information has been covered. For more details, please refer to my output. Thanks again for your reading.

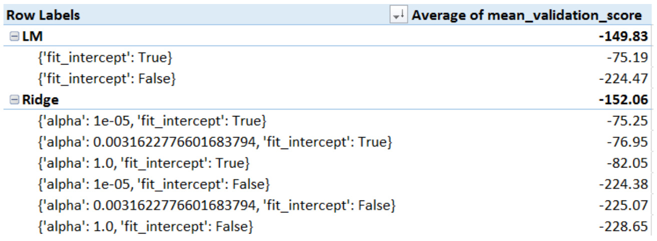
APPENDIX

A. *World Cup Final 2018 CV (MAE)*



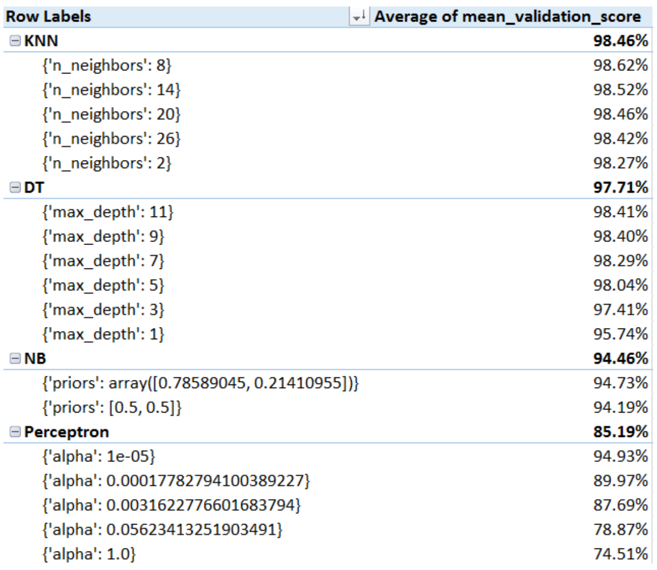
LM	**1.32**
{'normalize': False, 'fit_intercept': False}	1.38
{'normalize': True, 'fit_intercept': False}	1.38
{'normalize': False, 'fit_intercept': True}	1.26
{'normalize': True, 'fit_intercept': True}	1.26
Ridge	**1.32**
{'normalize': False, 'alpha': 0.0031622776601683794, 'fit_intercept': False}	1.57
{'normalize': True, 'alpha': 0.0031622776601683794, 'fit_intercept': False}	1.57
{'normalize': False, 'alpha': 1.0, 'fit_intercept': False}	1.40
{'normalize': True, 'alpha': 1.0, 'fit_intercept': False}	1.40
{'normalize': False, 'alpha': 1e-05, 'fit_intercept': False}	1.38
{'normalize': True, 'alpha': 1e-05, 'fit_intercept': False}	1.38
{'normalize': False, 'alpha': 0.0031622776601683794, 'fit_intercept': True}	1.34
{'normalize': False, 'alpha': 1.0, 'fit_intercept': True}	1.32
{'normalize': False, 'alpha': 1e-05, 'fit_intercept': True}	1.28
{'normalize': True, 'alpha': 1e-05, 'fit_intercept': True}	1.26
{'normalize': True, 'alpha': 0.0031622776601683794, 'fit_intercept': True}	1.23
{'normalize': True, 'alpha': 1.0, 'fit_intercept': True}	0.77

B. *SH1 Traffic Volume CV (Negative MAE)*



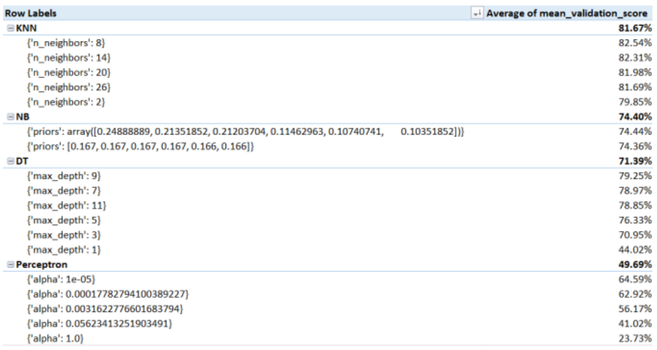
Row Labels	Average of mean_validation_score
LM	**-149.83**
{'fit_intercept': True}	-75.19
{'fit_intercept': False}	-224.47
Ridge	**-152.06**
{'alpha': 1e-05, 'fit_intercept': True}	-75.25
{'alpha': 0.0031622776601683794, 'fit_intercept': True}	-76.95
{'alpha': 1.0, 'fit_intercept': True}	-82.05
{'alpha': 1e-05, 'fit_intercept': False}	-224.38
{'alpha': 0.0031622776601683794, 'fit_intercept': False}	-225.07
{'alpha': 1.0, 'fit_intercept': False}	-228.65

C. *Occupancy Sensor CV (F1 Score)*



Row Labels	Average of mean_validation_score
KNN	**98.46%**
{'n_neighbors': 8}	98.62%
{'n_neighbors': 14}	98.52%
{'n_neighbors': 20}	98.46%
{'n_neighbors': 26}	98.42%
{'n_neighbors': 2}	98.27%
DT	**97.71%**
{'max_depth': 11}	98.41%
{'max_depth': 9}	98.40%
{'max_depth': 7}	98.29%
{'max_depth': 5}	98.04%
{'max_depth': 3}	97.41%
{'max_depth': 1}	95.74%
NB	**94.46%**
{'priors': array([0.78589045, 0.21410955])}	94.73%
{'priors': [0.5, 0.5]}	94.19%
Perceptron	**85.19%**
{'alpha': 1e-05}	94.93%
{'alpha': 0.00017782794100389227}	89.97%
{'alpha': 0.0031622776601683794}	87.69%
{'alpha': 0.05623413251903491}	78.87%
{'alpha': 1.0}	74.51%

D. *Landsat Satellite CV(F1-score)*



Row Labels	Average of mean_validation_score
KNN	**81.67%**
{'n_neighbors': 8}	82.54%
{'n_neighbors': 14}	82.31%
{'n_neighbors': 20}	81.98%
{'n_neighbors': 26}	81.69%
{'n_neighbors': 2}	79.85%
NB	**74.40%**
{'priors': array([0.24888889, 0.21351852, 0.21203704, 0.11462963, 0.10740741, 0.10351852])}	74.44%
{'priors': [0.167, 0.167, 0.167, 0.167, 0.166, 0.166]}	74.36%
DT	**71.39%**
{'max_depth': 9}	79.25%
{'max_depth': 7}	78.97%
{'max_depth': 11}	78.85%
{'max_depth': 5}	76.33%
{'max_depth': 3}	70.95%
{'max_depth': 1}	44.02%
Perceptron	**49.69%**
{'alpha': 1e-05}	64.59%
{'alpha': 0.00017782794100389227}	62.92%
{'alpha': 0.0031622776601683794}	56.17%
{'alpha': 0.05623413251903491}	41.02%
{'alpha': 1.0}	23.73%

REFERENCES

[1] Boyat, A., & Joshi, B. K. (2013, November). Image denoising using wavelet transform and median filtering. In Engineering (NUiCONE), 2013 Nirma University International Conference on (pp. 1-6). IEEE.

[2] Cover, T. M., & Thomas, J. A. (2012). Elements of information theory. John Wiley & Sons.

[3] Cunningham, P., & Delany, S. J. (2007). k-Nearest neighbour classifiers. Multiple Classifier Systems, 34(8), 1-17.

[4] Efron, B., & Tibshirani, R. (1997). Improvements on cross-validation: the 632+ bootstrap method. Journal of the American Statistical Association, 92(438), 548-560.

[5] Fletcher, D., MacKenzie, D., & Villouta, E. (2005). Modelling skewed data with many zeros: a simple approach combining ordinary and logistic regression. Environmental and ecological statistics, 12(1), 45-54.

[6] Hearst, Marti A., et al. "Support vector machines." IEEE Intelligent Systems and their applications 13.4 (1998): 18-28.

[7] Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12(1), 55-67.

[8] Kohavi, Ron (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. San Mateo, CA: Morgan Kaufmann. 2 (12): 1137–1143.

[9] Quinlan, J. R. (1986). Induction of decision trees. Machine learning, 1(1), 81-106.

[10] Pudil, P.; Novovičová, J. (1998). "Novel Methods for Feature Subset Selection with Respect to Problem Knowledge". In Liu, Huan; Motoda, Hiroshi. Feature Extraction, Construction and Selection. p. 101.

[11] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6), 386.

[12] Roweis, S. T.; Saul, L. K. (2000). "Nonlinear Dimensionality Reduction by Locally Linear Embedding". Science. 290 (5500): 2323–2326.

[13] Schabenberger, O., & Gotway, C. A. (2017). Statistical methods for spatial data analysis. CRC press.

[14] Zhang, H. (2004). The optimality of naive Bayes. AA, 1(2), 3.