Assignment 1 - Basic Data Mining Techniques

Ankur Anmol (2701807), Yorick Mengelers (2557326), and Qiao Ren (11828668)

University of Amsterdam / Vrije University Amsterdam

1 TASK 1: EXPLORE A SMALL DATASET (40 POINTS)

1.1 TASK 1A: EXPLORATION (20 POINTS)

We used HTML with d3.js for the visualizing the ODI data, since d3 supports more customized visualizations than matplotlib and seaborn. With the help of d3 we were able to render multiple barcharts together and explore the dataset more effectively.

There were total of 313 records with 17 features. We used barcharts for rendering all the attributes data, and below are some observations which we made;
- Majority of students enrolled in the course were from Computer Science (37.06 %) and Artificial Intelligence (27.80%) streams (. 3).
- Most of the students were aged between 23 - 26 (born 1995 -1998). Most students slept between 12 - 2 AM(2).
- Most popular answers for the questions 'What makes a good day for you?' were - Sunshine, Friends, and Food. Other popular answers were Sports, No Stress, Sleep, etc

Relationship between different attributes:

To further explore the data, we added the functionality to zoom into a particular attribute and observe how the graph plots change for different answer sets. Below are some interesting observations:

The Information Studies course had a much greater strength of girls (57.14%) as compared to other courses (32.27The Information Studies stream also showed higher levels of stress among students. AI students reported lower ratings average (44.83% as compared to 37.70%)

1.2 TASK 1B: BASIC CLASSIFICATION/REGRESSION (20 POINTS)

Language Used: Python
Libraries: pandas, sklearn

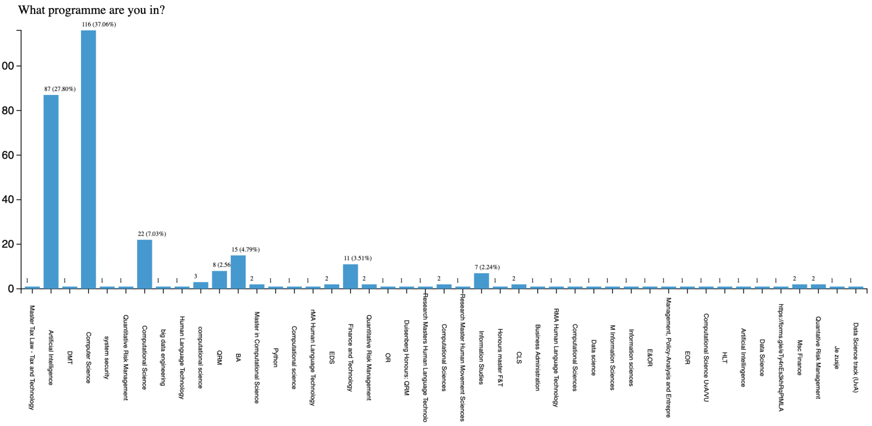


Fig. 1. Most students were from CS and AI courses

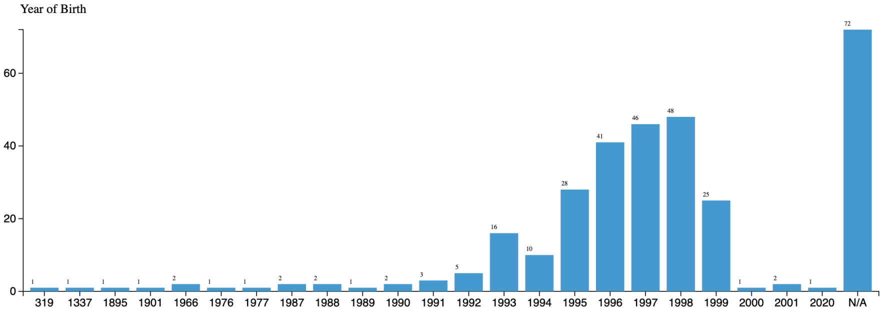


Fig. 2. Most students slept between 12-2 AM the previous night

The pandas library was used to read the ODI dataset. We performed some basic analysis of the dataset: There were 313 instances and 17 features. The task was to predict if someone has taken a machine learning course before. As predictive features the first six columns were selected for the task. Missing value and selected features:

missing data: Total Have you taken a course on information retrieval? 19 6.1%
Have you taken a course on statistics? 16 5.1%
Have you taken a course on databases? 8 2.6%
Have you taken a course on machine learning? 0 0.0%
What programme are you in? 0 0.0%

Missing instances (people) were removed from the data set. The column 'What programme are you in?' was converted into a one hot labeling scheme and the other (boolean) features were converted to numbers with the pandas

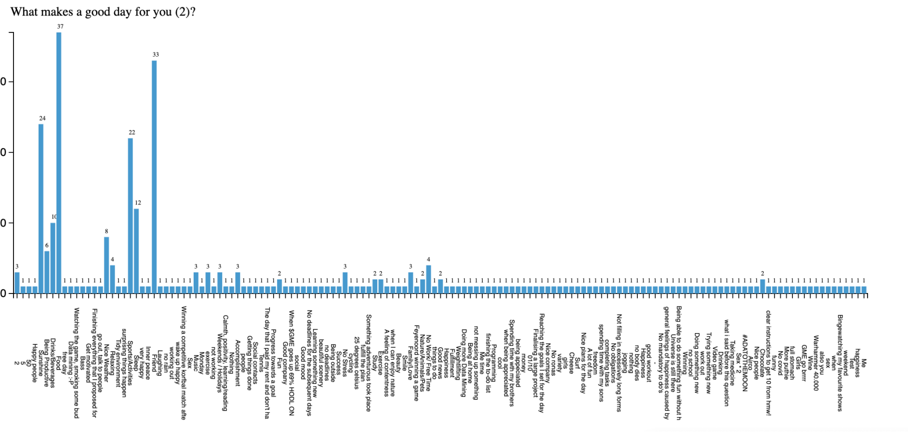


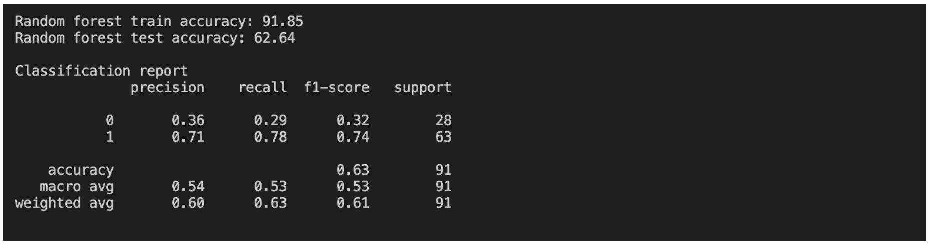
Fig. 3. Sunshine, friends, food made a good day for many students

get_dummie's. Numeric data is needed for most ML algorithms. The question if someone has taken a machine learning course before was dropped from the dataset and added as a label.

For cross validation the pandas function train_test_split was used to split the data set into a train (77%) and test (23%) set.

Random forest was used to make the classification. The used parameters were kept default (n_estimators=100, *, criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0).

The training and testing accuracies for the classifier were 91.85% and 62.64%. The big difference between test and train data is because the model overfits the train data. (Refer. 4).

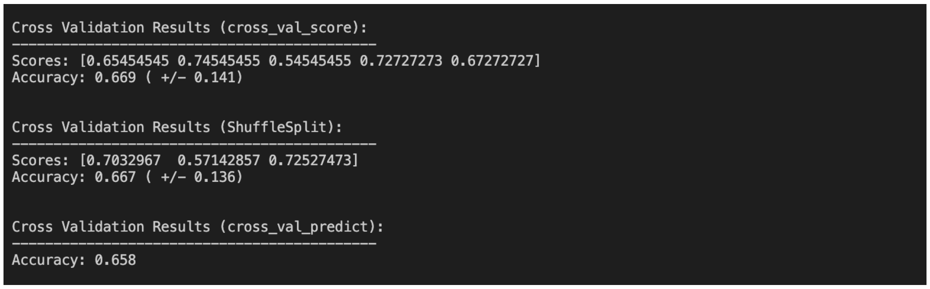


```
Random forest train accuracy: 91.85
Random forest test accuracy: 62.64

Classification report
              precision    recall  f1-score   support

           0       0.36      0.29      0.32        28
           1       0.71      0.78      0.74        63

    accuracy                           0.63        91
   macro avg       0.54      0.53      0.53        91
weighted avg       0.60      0.63      0.61        91
```

Fig. 4. Report : Random Forest Classifier

For further cross validating this classier, multiple library functions (from sklearn.model_selection) were used. These included: cross_val_score cross_val_score with ShuffleSplit cross_val_predict

The cross validations gave the same accuracy as predicted by the classifier. (refer Fig. 5).



```
Cross Validation Results (cross_val_score):
--------------------------------------------
Scores: [0.65454545 0.74545455 0.54545455 0.72727273 0.67272727]
Accuracy: 0.669 ( +/- 0.141)


Cross Validation Results (ShuffleSplit):
--------------------------------------------
Scores: [0.7032967  0.57142857 0.72527473]
Accuracy: 0.667 ( +/- 0.136)


Cross Validation Results (cross_val_predict):
--------------------------------------------
Accuracy: 0.658
```

Fig. 5. Results of the cross validations performed on the random-forest classifier

Findings The cross validations confirmed that the test accuracy predicted by the random-forest classifier was correct.
The classifier had predicted it to be 62.64%
The cross validators showed the similar values: 0.647, 0.670, 0.647

2 TASK 2: COMPETE IN A KAGGLE COMPETITION TO PREDICT TITANIC SURVIVAL (30 POINTS)

2.1 Task

The titanic dataset consists of 891 people or instances that boarded the Titanic when disaster stroke. The task is to build a classifier that can predict whether someone survived the crash given a test set with 11 features. The training set also includes the information if someone survived. The files were loaded using the pandas get csv package.

2.2 Data exploration

The titanic train data set consists of 12 features one of which is the boolean class label ‘Survived’. Two of the features are floats (age and fare), 5 are integers (passenger id, survived, Pclass, nr of siblings and parch) and 5 are objects (name, sex, ticket, cabin and embarked). 38 % of the passengers survived the crash, so there is a class imbalance but because of the binary nature of the classification, no class balancing has been done. Because the features ‘ticket’ and ‘name’ revering to the passenger name and ticket number contains almost exclusively unique and categorical data, intuitively not correlated to survival, the features were dropped. Of the remaining features, only three contained missing values (table 1). The "Cabin" column contains a lot of missing values, so dropping this column may seem appropriate. But after taking a closer look at the data, the missing values are probably people that do not have a cabin, as it is considered a luxury, it may have predictive power. The cabin feature was converted into a boolean: 1 if someone stayed in a cabin and 0 if not. In order to compare features and use them to train a classifier, the remaining categorical features were transformed into numerical features using the pandas get dummies function.

Feature	Total	%
Cabin	687	77.1
Age	177	19.9
Embarked	2	0.2

To get an impression of the data and to decide what to do with the missing values, a correlation matrix was made using the Pearson correlation (although because of the different features units and unknown distributions other correlations would possibly be more suitable). The correlation matrix (Table 1) shows that survival correlates most heavily with the fare, cabin, pclass and sex. Because the embarking position shows some correlation with survival the passengers with missing embarked data were removed. Staying in a cabin also shows to have a positive correlation with survival so the column was not dropped. Surprising is the absence of shown correlation between age and survival. There are multiple possible explanations for this absence, one of them is the distribution of different ages. To show the distribution of age, sex and survival plot 4 was made. There is no obvious correlation seen in the plots between age and survival. Because there may be underlying correlations that may have a predictive power the age column was kept. But because we do not want to lose passengers who contain valuable information and the classifiers can not handle the missing values, so the missing ages were filled with the mean age. The two passengers with missing embarked information were removed from the train set. In the test set, the missing values were all filled with the column mean.

2.3 Classification

The dataset contains a relatively low number of features compared to instances. This makes the chance to overfit the model relatively low (Altman Krzywinski, 2018). For this reason, all remaining features are used to build the models and no feature selection algorithm was applied. Most models perform best, with only continuous or categorical data. The titanic data set contains both. To tackle this problem there are multiple options: use a flexible classifier that can handle this kind of data, convert numerical feature into categories or separate the numerical features from the categorical, perform different algorithms to both and combine the results by a stacking or ensemble algorithm. To make the task, not over

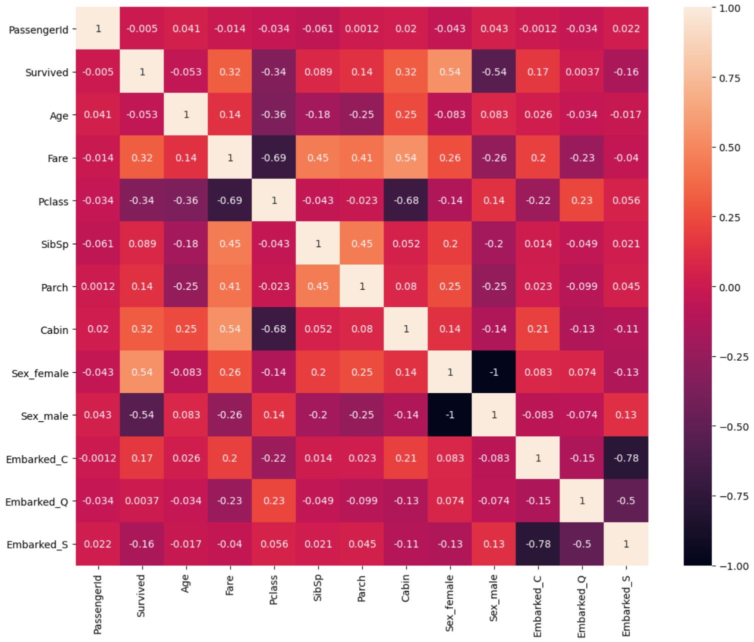

PassengerId 1 -0.005 0.041 -0.014 -0.034 -0.061 0.0012 0.02 -0.043 0.043 -0.0012 -0.034 0.022
Survived -0.005 1 -0.053 0.32 -0.34 0.089 0.14 0.32 0.54 -0.54 0.17 0.0037 -0.16
Age 0.041 -0.053 1 0.14 -0.36 -0.18 -0.25 0.25 -0.083 0.083 0.026 -0.034 -0.017
Fare -0.014 0.32 0.14 1 -0.69 0.45 0.41 0.54 0.26 -0.26 0.2 -0.23 -0.04
Pclass -0.034 -0.34 -0.36 -0.69 1 -0.043 -0.023 -0.68 -0.14 0.14 -0.22 0.23 0.056
SibSp -0.061 0.089 -0.18 0.45 -0.043 1 0.45 0.052 0.2 -0.2 0.014 -0.049 0.021
Parch 0.0012 0.14 -0.25 0.41 -0.023 0.45 1 0.08 0.25 -0.25 0.023 -0.099 0.045
Cabin 0.02 0.32 0.25 0.54 -0.68 0.052 0.08 1 0.14 -0.14 0.21 -0.13 -0.11
Sex_female -0.043 0.54 -0.083 0.26 -0.14 0.2 0.25 0.14 1 -1 0.083 0.074 -0.13
Sex_male 0.043 -0.54 0.083 -0.26 0.14 -0.2 -0.25 -0.14 -1 1 -0.083 -0.074 0.13
Embarked_C -0.0012 0.17 0.026 0.2 -0.22 0.014 0.023 0.21 0.083 -0.083 1 -0.15 -0.78
Embarked_Q -0.034 0.0037 -0.034 -0.23 0.23 -0.049 -0.099 -0.13 0.074 -0.074 -0.15 1 -0.5
Embarked_S 0.022 -0.16 -0.017 -0.04 0.056 0.021 0.045 -0.11 -0.13 0.13 -0.78 -0.5 1
PassengerId
Survived
Age
Fare
Pclass
SibSp
Parch
Cabin
Sex_female
Sex_male
Embarked_C
Embarked_Q
Embarked_S
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

Fig. 6.

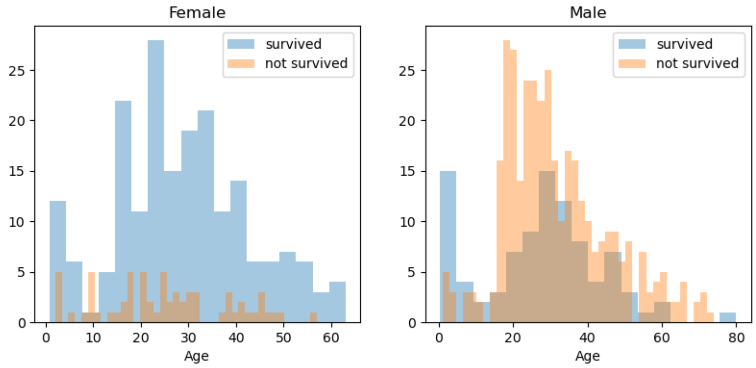

Female
survived
not survived
0
5
10
15
20
25
0
10
20
30
40
50
60
Age
Male
survived
not survived
0
5
10
15
20
25
0
20
40
60
80
Age

Fig. 7.

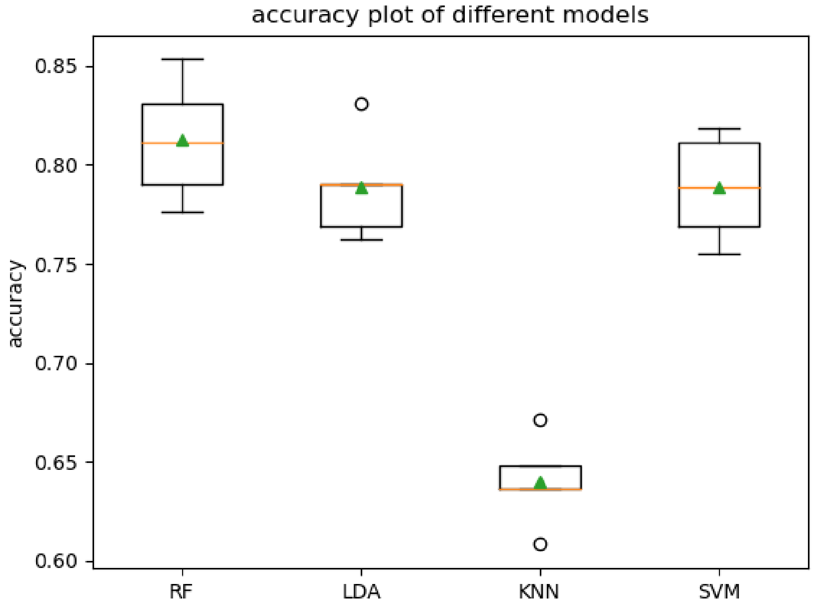
complicated the first approach was chosen. Classifiers that use a distance measure (KNN for example) to make their classification are probably not suited for the job, although it may work with a cosine distance and scaled data (Li Han, 2013) .The most suitable classifiers using categorical data are neural networks (NN), support vector machines (SVM), decision trees (DT), random forest (RF) or naive based (NB). SVM, NN and RF would probably work best because they are the most accurate and have a really flexible feature space, which can handle categorical and numerical data. If the data set was really big DT could be used to reduce computational costs, but for the titanic data set this was not the case, so RF would probably perform better or equal as it uses multiple decision trees. NN are quite complicated to build and are too challenging for the basic assignment, so SVM and RF were chosen for the task. Random forest uses multiple decision trees and combines their accuracy and correlation to make the eventual classification (Breiman, 2001). Support Vector machines divide the features space by maximizing the margin between the categories (Srivastava Bhambhu, 2010). To compare our algorithms two, in theory, unsuited classifiers k nearest neighbours (KNN) and linear discriminant analyses (LDA), were also included as benchmarks.

To prevent overfitting and get the most accurate representation of model performance double cross-validation was used. The training set was split five times (Kfold) into a test and train set and each split was used to calculate the model performance, this is the outer CV loop. In the inner CV loop, the train set 5 Fold CV was used to train the models (Filzmoser et al., 2009) . To select the best hyperparameter for each classifier a grid search was used with the input hyperparameters shown in table 2 (Bergstra Bengio, 2012). The grid search runs all classifiers with all combinations of hyperparameter and selects the best to evaluate model accuracy on the test set. The mean accuracy of the five outer CV loops was used to select the best performing model (figure 3).

As shown in figure 3. RF with a tree 'max depth': 5, 'max features': 6, 'n estimators': 10 performs best with an accuracy of 0.81. The LDA performed better than expected. This is probably due to the fact that they try to perform a dimension reduction to keep the most variation and the best separation (M. Li Yuan, 2005). So the decision is predominantly based on the fare feature which explains the most variance (the rest is 1 or 0) and separates the feature space better than age. The best performing RF model was build using the data of all passengers with the best hyperparameters. This model was used to make predictions on the unknown test set. The Kaggle score was 0.76, less than our training score. So our model was overfitting even with the double cross-validation. The score was not very good probably because of the combination of continuous and categorical data.

Table 1. Table captions should be placed above the tables.

KNN	N neighbors: 1,2,3,4,6	
Random forest	$\text{Max}_depth : 3, 5, 15, 50, 100$	$\text{N}_estimators : 10, 400$
$\text{Max}_features : Auto, 2, 4, 6, 8, 10, 12$		
SVM	C: 1, 5, 15	Gamma: [auto, 1, 0.1]
kernel : linear and rbf		
LDA	solver : svd, lsqr, eigen	



3 TASK 3: RESEARCH AND THEORY (30 POINTS)

3.1 TASK 3A: RESEARCH - STATE OF THE ART SOLUTIONS

The competition KDD Cup 2010 Educational Data Mining Challenging https://pslcdatashop.web.cmu.edu/KD was held in July, 2010. The competition task is to develop a learning model based on logs of student interaction with Intelligent Tutoring Systems and use this model to predict student performance on mathematical problems in the test sections. The goal is to learn a model from students' past behavior and then predict their future performance. There are 5 provided data sets: 3 development data sets (Algebra I 2005-2006, Algebra I 2006-2007, Bridge to Algebra 2006-2007) and 2 challenge data sets (Algebra I 2008-2009, Bridge to Algebra 2008-2009). In terms of evaluation measures, the winner is determined based on their model's performance on an unseen portion of the challenge test sets.

The winner is a team from the Department of Computer Science and Information Engineering, National Taiwan University. The team members include: Hsiang-Fu Yu, Hung-Yi Lo, ect. The techniques they used are: 1)Feature generation. 2) Feature selection: Wrapper with forward or backward selection (nested subset method). 3) Classification methods: Decision tree, stub, or Random Forest, Linear classifier (Fisher's discriminant, SVM, linear regression). 4) Regularizer: L1 and L2 regularizer. 5) ensemble method: boosting, bagging

The main idea of the winning approach (Hsiang-Fu, 2010) is to extract useful features and apply suitable classifiers (see figure below). 1) The first step is to extract two types of features: sparse feature and dense feature. Sparse feature sets are generated by binarization and discretization techniques. Condensed feature sets are extracted by using simple statistics on the data. The authors applied and compared various classifiers: random forest, adaboost, logistic regression via liblinear. 2) The second step is to ensemble the classifiers, which means to find a weight vector to linearly combine the predicted probabilities which are generated in step 1. The authors found that linear ensemble is better than nonlinear ensemble in this project. They evaluated several linear models including simple averaging, linear SVM, linear regression and logistic regression. In the end, the authors found that the L1-regularized logistic regression solver in LIBLINEAR is the best.

What makes the winning approach stand out is their success is feature diversity. Different sub-teams tried various ideas guided by their human intelligence.

3.2 TASK 3B: THEORY - MSE VERSUS MAE (10 POINTS)

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_{real} - y_{pred}|^2 \qquad MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{real} - y_{pred})^2$$

Q1 Mean Absolute Error (MAE) measures "the absolute average distance between the real data and the predicted data, but it fails to punish large errors in prediction". Mean Square Error (MSE) measures "the squared average distance between the real data and the predicted data."

Q2 MAE is less biased for higher values, while MSE is highly biased for higher values. MSE is not suitable when data is skewed. MAE doesn't necessarily penalize large errors, while MSE penalizes large errors. MSE gives high error to outliers, because it is "squared". So MSE is more sensitive to outliers.

Q3 In the following cases, using MSE or MAE would give identical results. 1) when y_{real}-y_{pred}=0, meaning that the regressed function perfectly matches with all the data points without any error. Or 2) when y_{real}-y_{pred}=1, meaning that the difference between real and predicted value for each data point is exactly 1. Or 3) when y_{real}-y_{pred}=-1 meaning that the difference between real and predicted value for each data point is exactly -1

Q4 The dataset comes from: https://www.kaggle.com/sonjabutler/linear-regression-analysis-on-szeged-weather We chose this dataset because it has been used for linear regression analysis. It has 96453 rows and 12 columns (attributes). The goal is to build up a linear regression model in order to fit the "apparent temperature", with the given 11 input attributes: humidity, wind speed, precip type, ect. The result is shown in the table below. Linear regression gives higher MSE and higher MAE than polynomial regression. This shows that polynomials are better than linear for this dataset. Because there are lots of input attributes. This dataset requires a complicated model.

Table 2. MSE and MAE of different regression models

regression model	MSE	MAE
linear regression	1.16	0.85
polynomial regression	0.37	0.44

3.3 TASK 3C: THEORY - ANALYZE A LESS OBVIOUS DATASET (10 POINTS)

Q1 The modeling techniques which are suitable to use with this type of data: Naive Bayes, SVM, Neural Network are suitable to classify the spam and ham data. Because the number of words in the data is huge. This data has a large size. Random forest or decision trees are not able to deal with this kind of data.

Q2 The data transformations we used on this dataset: 1) Tokenization: it means to remove white-spaces and punctuation. The goal is to chop a sentence or a phrase into individual words. 2) remove stop words. It means to remove the words which are extremely common but have no use in classification 3) convert terms to lower-case. This helps the algorithm count how many times a word has occurred, no matter whether it is uppercase or lowercase. 4) Convert terms to their stems. It reduces inflectional forms. For example, it converts "went", "go", "will go" to their stem "go". 5) Convert training dataset to TF-IDF (term frequency-inverse document frequency) or bag of words. It means that we build up a dictionary for the whole training set. For each word, it has a corresponding weight. The weight is computed by TFIDF.

Q3 We built a multinomial Naïve Bayes model. Here is the process: We split the SmsCollection.csv into a training set (70 % of all data) and test set (30 % of all data). We convert the messages in the training set into a TF-IDF matrix with the labels "spam" and "ham". We did the same conversion on the test set. Then we built up a multinomial Naïve Bayes model on the training set. After that, we make predictions on the test set. In the end, we evaluate the performance of Naïve Bayes model. The accuracy of classification on the test set: 96.83 %

Table 3. Confusion Matrix

True negative = 1352	False negative = 2
False negative = 48	True positive = 173

Table 4. Precision, Recall and F1 score

class	precision	recall	F1 score
Spam	0.97	1.00	0.98
Ham	0.99	0.78	0.87

To make an improvement, the suggestion is to use neural networks such as LSTM (long short term memory). Because the Naive Bayes model assumes all the words are independent. But actually there is dependency among words. Neural network learns the dependency among words. It improves the accuracy of classification.

References

Hsiang-Fu Yu et al. 2010 Feature Engineering and Classifier Ensemble for KDD Cup 2010. Department of Computer Science and Information Engineering, National Taiwan University Taipei 106, Taiwan JMLR: Workshop and Conference Proceedings 1: 1-16

Altman, N., Krzywinski, M. (2018). The curse(s) of dimensionality this-month. In Nature Methods. https://doi.org/10.1038/s41592-018-0019-x

Bergstra, J., Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research.

Breiman, L. (2001). Random forests. Machine Learning. https://doi.org/10.1023/A:1010933404324

Filzmoser, P., Liebmann, B., Varmuza, K. (2009). Repeated double cross validation. Journal of Chemometrics. https://doi.org/10.1002/cem.1225

Li, B., Han, L. (2013). Distance weighted cosine similarity measure for text classification. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). https://doi.org/10.1007/978-3-642-41278-3$_7$4