Assignment 1 - Basic Data Mining - Group 142

Kamiel Gülpen, Dante de Lang, and Louky Schutte

University of Amsterdam, Amsterdam 1012 WX, Netherlands

1 Task 1: Explore a small data set

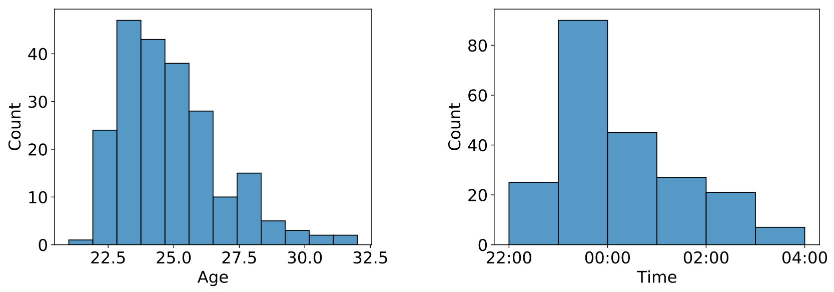
1.1 Exploration

Pre-processing The ODI-2021 dataset consists of 16 columns representing the answers on various questions, in total 313 Data Mining students between 16-3-2021 and 30-3-2021 responded. Before we could study and make sense of the data we pre-processed a large part of it. Table 1 shows an overview of the questions and alterations made to the dataset. We applied One-Hot Encoding to the multiple choice questions. For open questions outlying input values were detected and replaced, where possible values were converted to integers. For Q16/17 we used the FLAIR Natural Language Processing (NLP) library [1] in order to get a sentiment reading of the textual answers.

Table 1: ODI-2021 dataset features

Column	Alterations made	Range of values
Q1: Time stamp	Converted to days before deadline	[0, 14]
Q2: Programme	Applied One-Hot Encoding (OHE)	[0,1]
Q3-Q6: Courses	Applied OHE	[0,1]
Q7: Gender	Converted to binary	[0,1]
Q8: Chocolate	Applied OHE	[0,1]
Q9: Birthday	Converted to age	[20, 55]
Q10: Neighbours	Converted to integers, deleted outliers	[0,100]
Q11: Stand up	Applied OHE	[0,1]
Q12: Stress level	Set to numerical and bounded	[0, 100]
Q13: Money (self esteem)	Set to numerical and bounded	[0, 100]
Q14: Random	Converted to integers	[0, 10]
Q15: Bed time	Converted to integers	[0,23]
Q16/17: Good day 1 & 2	Applied NLP for sentiment analyses	[-1,1]

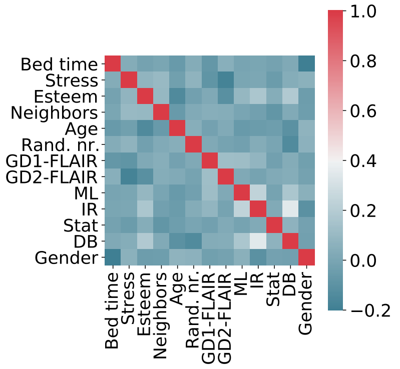
Exploring the data set A descriptive analysis is conducted in order to explore the data set. During this analysis it is found that 65.5% of the students identify themselves as male, 32.2% as female and 2.3% is unknown. Most of these students follow the master programme AI (28.7%) followed by Business analytics, Computational science and Computer science which cover 11.1%, 11.1% and 10.4% respectively. The large majority of these students followed a statistics course, namely, 86.3% while only 51.8% of the students followed a course about data bases. We furthermore found that the mean age is 18.6 with a std of 11.68. It might be more insight full to look at the median of the age, which is 23, as the mean is affected heavily by outliers. We also looked at the stress level (M=51.244, SD=29.97) and Self esteem (M=29.66, SD=36.49) of students. We additionally examined the bedtimes of students which showed that they went to bed at 11am on average. The last explored columns were the good day columns with applied sentiment analysis, here we saw that the mean of GD1(M=0.804, SD=0.514) was higher than GD2(M=0.762, SD=0.589). Another interesting point regarding to these features is that the minimum of both GD1 and GD2 had a value of -1, meaning a negative sentiment.



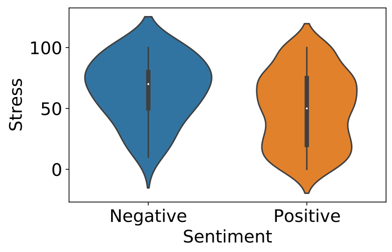
(a) (b)

Fig. 1: This figure shows the distribution of Age 1a and bed time 1b of the students

Graphical depictions are used in order to make some of the features more insightful. Figure 1 shows the distributions of the age (1a) and bed times (1b) after removing the outliers with a quantile of 0.975. Figure 1a shows that most of the students are between 22.5 and 27.5 years old. Figure1b show that the majority of the students go to bed between 22:00 and 01:00. We furthermore looked at the correlation of the features, which are presented in Figure 2a. This figure shows two relatively high negative correlations, namely between GD2-FLAIR and stress(-0.18) and between gender and bedtime (-0.20). The two highest positive correlations are between having done a course about Data Bases (DB) (0.35) and Information Retrieval (IR) and between IR and Machine Learning (ML) (0.24). Figure 2b shows the relation between the sentiment of GD2-FLAIR and stress. It shows the stress level of people who either answered with a positive or a negative sentiment. One thing that stands out is the fact that a large portion of people with a negative sentiment have a high stress level while the stress level of people with a positive sentiment are more spread out.



(a)



(b)

Fig. 2: Figure 2a shows a correlation plot of the features, 2b shows a violin-plot of the negative and positive values for sentiment (GD2-FLAIR).

1.2 Basic regression

An interesting approach could be to predict the stress level of a student. We select two simple regression algorithms: Random Forest regressor and Epsilon-Support Vector Regression, the basis of these techniques will be discussed in further detail in Section 2.2. Subsequently we create two different selections of features, one 'small dataset' with only the 5 features most correlated with Stress ('GD1-FLAIR', 'GD2-FLAIR', 'Neighbors', 'Self esteem' and 'Random number') and one 'larger dataset' where we take all features into account.

Random forest We apply functionalities from the Python package *Skicit-learn* to fit a random forest regressor to predict the values of stress of students. The main parameters in this model are the number of decision trees, which is set to 1000 and the criterion for measuring the quality of a split, for which we use Gini impurity. Other parameters are the minimum number of samples required to split an internal node and the minimum number of samples required to be at a leaf node, for these parameters we use default values 2 and 1 respectively.

Epsilon-Support Vector Regression The *Skicit-learn* package can also be applied to fit a Epsilon-Support Vector Regression. The free parameters in this model are the Regularization parameter C and epsilon, in our model both parameters are 1 (the default value). Epsilon specifies the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value.

To have a somewhat reliable output we use cross validation and summarize the metrics mean absolute error (MAE) and coefficient of determination (R^2).

We used 10-fold cross validation, which has became a standard method in practical terms [6]. The average MAE and R^2 for both the Random forest and SVM of the cross validation procedure are displayed in table 2.

Table 2: Performance metrics for Random forest and SVM with 5 and 10 fold crossvalidation

	Random Forest		SVM	
	Small dataset	Large dataset	Small dataset	Large dataset
10-fold MAE [mean, sd]	26.1, 3.74	25.3, 3.90	26.0, 4.31	25.9, 4.32
10-fold R^2 [mean,sd]	-0.204, 0.234	-0.102, 0.144	-0.132, 0.177	-0.131, 0.178

For both the Random Forest and the SVM we retrieve a MAE larger than 25. Since the stress level is modelled on a scale from 0 to 100, the MAE above 25 indicates that the models perform quite poorly. This is confirmed by the negative R^2 values, which indicate that the models perform arbitrarily worse than a constant model that always predicts the expected value of Stress. The R^2 and MAE seem to be best for the large dataset with a Random forest. A t-test is performed (the cross validation scores were normally distributed) for the R^2 of the Random forest and the SVM in order to compare both models, as the p-values are 0.473 and 0.704 for the small and large data set respectively we do not reject the null hypothesis of equal means, which means that the differences between the models are not significant.

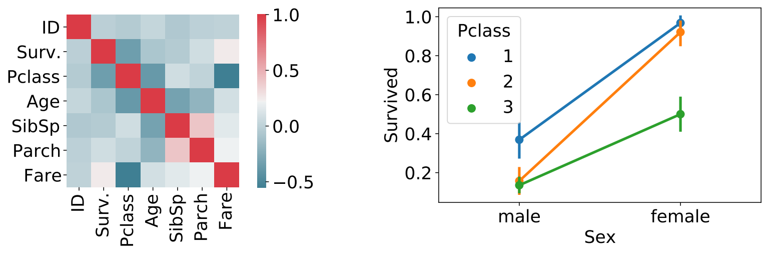
2 Task 2: Titanic Kaggle Competition

2.1 Feature analysis

The whole data set of the Titanic, test and traning data combined, contains 1309 rows and 12 columns. It is split into a training and test dataset with respectively 891 and 418 entries. The data set contains the following descriptive features, *PassengerId*, the Passenger's Id, *Age*, Age of the Passenger, *Sex*, Sex of the Passenger, *Name*, name of the Passenger, *Embarked*, whether the passenger embarked in Southampton, Cherbourg or Queenstown, *Parch*, the number of Parents/Children Aboard, *SibSp*, the number of Siblings/Spouses Aboard, *Fare*, the Passenger Fare, *Ticket*, the Ticket Number, *Cabin*, which Cabin the passenger stayed in, *Pclass*, a proxy for socio-economic status (1,2 or 3), and finally the label that we want to predict: *Survived*, which is 1 if the Passenger survived and 0 otherwise.

Since the test data set will only be used for predicting the Survival for the Kaggle competition, all further inference made will be based on the Training set containing 891 rows of passengers. Of the total of passengers 38% survived. With 74% of females and 18% of men surviving, gender seems to be quite predictive of survival.

Furthermore from the correlation plot in Figure 3a we find that Pclass has the greatest absolute correlation (-0.338) with Survival, followed by Fare (0.257) and Parch (0.0816). Additionally there seems to be high positive correlation between Parch and SibSp, and negative correlation between Pclass and Fare. Fig 3b shows that females had more chance of surviving and also the higher classes have a higher chance of surviving. Fore females the there is a bigger difference of surviving between the first two classes and the third class, for men this difference is largest between the first class and the two lower classes.



(a) (b)

Fig. 3: In Figure 3a correlation plot of some interesting features of the titanic dataset. In Figure 3b the survival rate per class and sex is plotted.

For the Pre-processing of the data both train and test data were combined in order to process the data in a convenient way. New columns were introduced after the dataset was analyzed in order to make better predictions. The first added column is the *Family size* column which adds the *Parch* column and the *SibSp* together. Secondly the *Is alone* column is added to the data frame, which checks if the passenger came alone on the ship or not. Another added column is the *Title_num* column which looks at the title of the passengers and classify those into four groups, namely, Mr, Master, Mrs and Mss, which were then given a number. The ages were categorised into 4 groups (*Age_div*), specifically, [0-20], [20-40], [40-60] and [60+]. Further more the *Deck* column is introduced, which grouped the cabin values into places on the deck. These places are: ABC, DE, FG or no cabin (M). The places are then one hot encoded into the columns *Deck_1*, *Deck_2*, *Deck_3*, *Deck_4*. Also *Sex* and *Pclass* are one hot encoded into *Sex_1, Sex_2* and *Pclass_1, Pclass_2, Pclass_3* respectively. Lastly two interaction features where added tot the data frame, which were the *Class*Sex* and the *Class*Age* feature.

The data set contained 263 rows of missing data in the age column, which is roughly 20%. In order to fill the missing values a linear regression model is used. This linear regression model took as features: *Family_size, Title_num, Fare and Pclass*, which returned a R^2 value of 0.21.

2.2 Implementing classifiers

Two classifiers are used in order to predict the the survival state of the passengers in the test data, namely, the Random Forest Classifier (RF) and Support Vector Classifier (SVC)[1]. These classifiers where chosen because they have shown to be good predictors for classification problems [7]. Without going into detail a short explanation of both classifiers will follow.

The RF classifier uses an ensemble of multiple uncorrelated decision trees in order to create a voting system for each classification. These trees are then fed with data using a method called bagging (Bootstrap Aggregation), this data is sampled with replacement from the training set.

For the SVC the data is classified by fitting a hyperplane, this is mathematically scale-able to higher dimensions with $N_{dim} = N_{feature}$. Using data points close to the hyperplane, so-called support vectors, it becomes possible to maximize the margins between the separating hyperplane and the vectors using a cost function and gradient updates.

2.3 Optimizing classifiers

In order to measure and compare the accuracy of the implemented classifiers, the cleaned training data was split in two subsets. Setting these subsets as the new training and validation data, with a ratio of 80:20, it becomes possible to rate and optimize both classifiers. Before optimizing the classifiers a selection was made within the available features. These features are chosen based on their correlation and through a feature importance search. For the feature importance search we used the RF classifier which was fed with all the numerical columns created throughout the feature analysis. Based on these methods the following features were chosen for further optimization: *Deck(1,2,3,4), Fare, Age, Title_num, Sex(1,2), Pclass(1,2,3), Family size.*

In order to optimize both classifiers a grid-search setup was created which was fed with value options for different parameters per classifier. The model was then evaluated on the train data during the parameter search using a 10-fold cross validation. For the RF classifier the optimal parameters found within our given boundaries were: *criterion = 'gini', n_estimator = 100, min_samples_leaf = 4* and *max_depth = None.* For the SVC optimal parameters were: *kernel = 'rbf', gamma = 'scale', degree = 0.1, probability = True* and *decision_function_shape = 'ovr'*. Furthermore, both classifiers used a random state set to 0.

After the parameters of the classifiers were determined the models could be validated. This was done by making predictions with the trained model on the validation data set. In order to get a reliable reading of their performance we evaluated using the 10-fold cross-validation. The 95% confidence interval (CI) of the accuracies are shown in Table 3.

In order to get some more insight in the performance of both classifiers their classifications were summarized in a confusion matrix, see Figure 4. Figure 4a

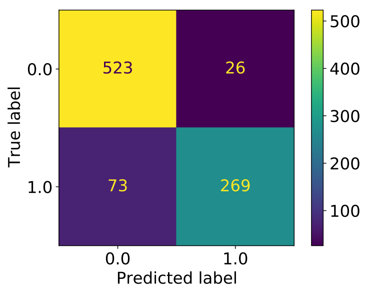
[1] Both classifiers were implemented using the Scikit-Learn library [5].

Table 3: Performance metrics for Random forest and SVC with 10 fold cross-validation on train and validation set

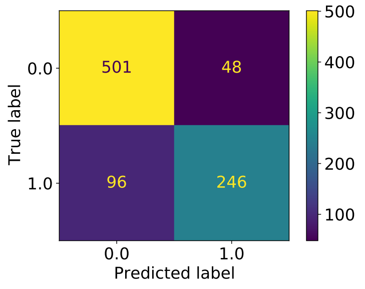
	Random Forest	SVC
10-fold mean score train	0.811	0.809
95% Confidence interval	[0.782, 0.84]	[0.786, 0.832]
10-fold mean score test	0.849	0.855
95%Confidence interval	[0.789, 0.909]	[0.797, 0.912]

presents the confusion matrix for the RF classification model whereas Figure 4b shows the confusion matrix for the SVC model. Both figures show the outcome of the optimized model on the whole training set.

When comparing both models we see that the RF model performed better on both the true negative and true positive classifications with respect to the SVC model. Furthermore, we see that the SVC model shows 1:2 proportion for the false positives and false negatives classifications. While for the RF model the total false classifications is lower, this proportion is more skew $\approx$ 1:2.8.



(a)



(b)

Fig. 4: In the above two figures the confusion matrices are shown for the the RF model 4a and the SVC model 4b.

2.4 Kaggle

A comparison is made between the RF and SVC in order to submit the best classifier to the kaggle competition. This comparison is made with Welch's t-test and is based on the 10-Fold validation values of the test set. The statisical test showed a p-value of 0.355, and is therefore not significant. For this reason both classifiers were send to the kaggle competition. The final kaggle score for the RF was 0.785, while the SVC classifier received a score of 0.778. This result is eventually not an expected outcome as the mean of both test scores found during the validation were above 0.8. Therefore a score of 0.8 or higher was expected.

A possible explanation for this expectation mismatch can be the fact that the model was over fitted on the train set, meaning that the models were too closely fit to the limited set of data. This could be the reason why the results of the model seemed to have a high score but decreased after applying the new test values.

3 Task 3

3.1 State of the art solutions

A competition that comes up in the top 10 of Kaggle competitions with highest rewards is that of the Data Science Bowl in 2017 about improving lung cancer detection [2]. With lung cancer accounting for $12 billion in health care cost in the US a meager $1 million in prizes was awarded to those who best leveraged a data-set to develop an algorithm that accurately determines cancerous lesions in the lungs. The data set, Provided by the US national Cancer Institute, consisted of over a thousand lung scans from high-risk patients. The images contain a series with plane sections of the chest cavity and is accompanied by a header that contains information about the patients and ground truth labels indicating whether the patient was diagnosed with lung cancer within one year of the date the scan was taken. The competition task was to develop a method able to determine the correct labels 'Cancer' or 'No cancer'. The evaluation metric used in this a multiple instance learning (MIL) problem was the cross-entropy loss on the test set.

Out of a total of 1972 teams Liao, Liang, Li and Hu [3] won the competition with the lowest cross-entropy loss of 0.3998 using the 3D Deep Leaky Noisy-or Network [4]. The main idea of their approach was that diagnosing the images is a two step process. First nodules (abnormal aggregations of cells) are detected by a Convolutional Neural Network, secondly the cancer probability of each nodule is calculated and combined by the leaky noisy-or model. Liao et al. realised that the relationship between nodules and cancer is a complex one. They chose the leaky noisy-or model because it assumes that different factors can independently lead to an event (cancer), additionally it accounts for a leakage probability to explain the cancer in case no other module can. This last functionality of the leaky noisy-or model is one of the things that make Liao et al. stand out from the other teams, that did not manage to find a superior way of aggregating the malignancy predictions of the nodules [2].

3.2 Performance metrics

Two widely used performance metrics are the mean absolute error (MAE) and mean squared error (MSE). These metrics measure the error between paired observations and can be used in order to compare the outcomes of forecast with their real outcomes.

[2] https://www.kaggle.com/c/data-science-bowl-2017

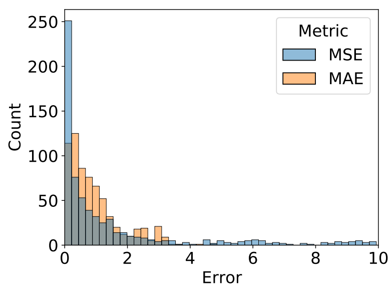
$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i| \qquad (1) \qquad\qquad MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \qquad (2)$$

The MAE can be calculated by taking the mean absolute error between two observations in a data set or between a prediction and its true value. This can also be written like Equation 1. The MAE can be best used when a dataset counains a small amount, or when one would not want to punish large outliers harder.

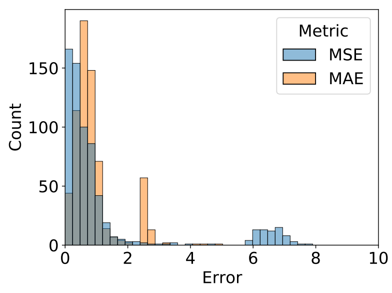
The MSE has a similar way of calculating the error, it differs from the MAE in the fact that it takes the squared error of two observations. The MSE is formulated as presented in Equation 2. The MSE is helpful for optimizing a model and when more insight on outlying values is desired.

When comparing both metrics it is clear that for most data they will give different results. However, for a binary problem both models will give identical results. Namely, when substituting $Y_i - \hat{Y}_i$ with x, in both metrics we can see that when Y_i and $\hat{Y}_i$ can only take values 0 or 1, $|x| = x^2$ is always true.

To further analyse the differences between the two error metrics, a experiment is conducted. In this experiment the insurance charges are predicted of people based on a persons bmi, age and number of children[3]. The predictions are done with two regression models, namely, a linear regression model and a random forest regressor. The experiment showed a MAE of 9026 and a MSE of 1.3e8 for the linear regression model, for the random forest regressor it returned a MAE of 9190 and a MSE of 1.5e8. The difference between the models is larger for the MSE then for the MAE. Distributions are made for the errors in order to make the difference between the MSE and MAE also graphically clear (see Figure 5). The charge price values were reduced by a factor 10000 in order to make the distributions. As visible in both Figure 5a and 5b, the MSE has a larger spread than the MAE which, looking at the theory, is a expected outcome.



(a)



(b)

Fig. 5: In the above two figures the MAE and MSE metrics are shown for a random forest model 5a and linear regression model 5b.

[3] https://www.kaggle.com/awaiskaggler/insurance-csv

3.3 Handling textual data

Till now we have only discussed handling categorical or numerical data, albeit after some transformations. However, their are situations were this conversion is not so candid. An example dataset is for instance one with text messages classified as either spam or bonafide. In order to train, fit and evaluate a classification model on this data, the data needs to be transformed in order for it to be handled correctly. One method that is commonly used to transform textual data is the Bag of Words (BoW) model. With BoW all unique words are collected from the whole dataset and set as row indices. Every column can then represent a text message with binary values for all words, where a 1 represents the presence of that word in the message.

With this transformation it becomes feasible for a classifier to do its job. Before feeding the BoW model with words, all text messages were filtered by lowering capital letters and removing punctuations and empty spaces. After these transformations the dataset, consisting of 5574 text messages, returned 7877 unique words using the BoW method. A big difference with datasets that were discussed earlier, is the feature dimensionality of the data. For example, for the Titanic data only 13 features were used for analysis, while using the BoW method on the text data creates hundreds of features (every unique word becomes a feature).

This characteristic has a direct influence on model choice, since not every method is capable of handling such dimensions. One model that is able to handle such a large feature space is the earlier mentioned Support Vector Machine model. In order to still suppress the amount of features the 500 most common words were selected as features for the modelling, with a word frequency ranging from 21 to 2528. Furthermore, the labels 'spam' and 'ham' (bonafide message) were converted to binary values.

After scaling the transformed dataset, using a StandardScaler[4], the SVC model was fed with 60% of the data for training. When doing a 10-fold cross validation on the SVC with default values, a accuracy with a mean of 0.957 and a 95% CI of [0.953, 0.962] was found. After some parameter optimization using a grid-search, returning *degree = 1, gamma = 'scale'* and *kernel = 'sigmoid'* , the accuracy increased with a mean of 0.967 and a 95% CI of [0.964, 0.97].

While these accuracies could already be considered as acceptably high, improvements are possible. Not only is it possible to implement other models and train and test using more advanced approaches, the data could also be improved. For instance, words could be grouped into sub-sentences or combined with their plural counterparts.

[4] Present in the Sklearn preprocessing library.

References

1. Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: COLING 2018, 27th International Conference on Computational Linguistics. pp. 1638–1649 (2018)
2. Hammack, D.: Forecasting lung cancer diagnoses with deep learning. In: Technical Report (2017)
3. Liao, F., Liang, M., Li, Z., Hu, X., Song, S.: Evaluate the malignancy of pulmonary nodules using the 3-d deep leaky noisy-or network. IEEE transactions on neural networks and learning systems **30**(11), 3484–3495 (2019)
4. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Elsevier (2014)
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
6. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques with java implementations. Acm Sigmod Record **31**(1), 76–77 (2002)
7. Zhang, C., Liu, C., Zhang, X., Almpanidis, G.: An up-to-date comparison of state-of-the-art classification algorithms. Expert Systems with Applications **82**, 128–150 (2017)