

**School of InfoComm Technology**

**Machine Learning**

Diploma in Data Science (DS)

Diploma in Information Technology (IT)

October 2022 Semester

**INDIVIDUAL ASSIGNMENT 2**

(40% of Machine Learning Module)

# Deadline for Submission:

**Presentation: 29th Jan 2023 (Sunday), 2359 Hours**

**Report: 11th Feb 2023 (Saturday), 2359 Hours**

|  |  |  |
| --- | --- | --- |
| Student Name | : | Kiara Avendano |
| Student Number | : | S10219186a |

**Penalty for late submission:**

10% of the marks will be deducted every day after the deadline.

**NO** submission will be accepted after 18th Feb 2023, 23:59.

Contents

[Deadline for Submission: 1](#_Toc127046240)

[Introduction 3](#_Toc127046241)

[HR Analytics 3](#_Toc127046242)

[Approaches 3](#_Toc127046243)

[Model Building 3](#_Toc127046244)

[Model Optimization 5](#_Toc127046245)

[Summary 6](#_Toc127046246)

[Airbnb 7](#_Toc127046247)

[Approaches 7](#_Toc127046248)

[Model Building 7](#_Toc127046249)

[Model optimization 9](#_Toc127046250)

[Summary 9](#_Toc127046251)

[Conclusion 10](#_Toc127046252)

[Reflection 11](#_Toc127046253)

[Improvements 11](#_Toc127046254)

[Module Conclusion 11](#_Toc127046255)

[Discussions 12](#_Toc127046256)

[Appendix 16](#_Toc127046257)

[Appendix 1 : The parameters used for *GridSearchCV* for every algorithm. 16](#_Toc127046258)

[Appendix 2 : The values from the statsmodels’ fitted models and the features’ importance from the random forest model. 20](#_Toc127046259)

# Introduction

Many problems can be solved using supervised machine learning models (classification and regression). Classification aims to predict a **categorical** label / variable, whereas regression aims to predict a numerical and **continuous** variable. This is achieved through supplying data to train and test the statistical models.

The report will be focussing on the HR analytics dataset (classification) and the Airbnb prices dataset (regression). Multiple trial and errors will be conducted to find the optimal model for each case. Additional steps were taken to further optimize the trained models, such as feature selection, conducting AdaBoost and using a Voting Ensemble. After comparing every model trained with different hyperparameters using different algorithms, the highest performing one will be chosen and used for both problems.

# HR Analytics

For the classification issue, the model had to predict whether an employee will get promoted. A special consideration had to be made of how getting promoted is rather rare, so the target variable’s classes would be heavily unbalanced. Therefore, regardless of the algorithm used, the entire dataset first needs to be stratified, to reduce the imbalance and bias in the final model.

## Approaches

The action plan is to first stratify the dataset based on the target variable of *“is\_promoted”,* so that the classes would have the ratio of 1:1. Afterwards, the optimization should also maintain this ratio when training and conducting cross validation, so StratifiedKFold was employed for every trial, where *k* would for 5. 5 folds is understandable for a dataset that now has 9 336 rows, as compared to 54 808 rows it had prior to stratification.

For this section, the dataset created from the previous assignment was used. Unfortunately, there were some issues present that were only identified towards the end of the project. The major issue would how the cleansed mismatched, where the rows of the dependent variables and the independent variable were not joined properly by the rows’ indexes. Due to this, practically all of models trained in the following sections are trained with the wrong dataset. Luckily, the issue got resolved so the models trained for the Voting Ensemble are using the proper cleansed and transformed dataset. It is truly regrettable that during the earlier stages in building and optimizing the models, this glaring issue was not addressed then.

## Model Building

This section will showcase the models trained using the mismatched dataset. They were all evaluated and compared to each other based on their accuracy score. The following algorithms were explored and even had their ideal hyperparameters based on their performance on the dataset:

* Logistic Regression
* Random Forest
* Artificial Neural Networks / Multi-layer Perceptron
* XGBoost
* LightGBM (Light Gradient Boosting Machine)
* CatBoost

**Table 1**

*Comparison of all the tuned and trained models’ accuracy on the mismatched training and testing datasets.*

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm used | Training Dataset | Testing Dataset | Difference in Accuracy |
| Naïve Baseline | 0.50 | 0.49 | -0.01 |
| Logistic Regression | 0.72 | 0.73 | +0.01 |
| Random Forest | 0.79 | 0.77 | -0.02 |
| ANN / MLP | 0.75 | 0.76 | +0.01 |
| XGBoost | 0.79 | 0.77 | -0.02 |
| LightGBM | 0.79 | 0.78 | -0.01 |
| CatBoost | 0.76 | 0.78 | +0.02 |

In Table 1, the tuned models trained using XGBoost, LightGBM, and Random Forest performed the best. As compared to the naïve baseline, the overall accuracies increased by at least 20%. Luckily, the models trained all does not seemed to be overfitted, with the minimal differences in the accuracies. However, there is always room for improvement.

The parameters grid that was used in cross validation can be seen in Appendix 1.

**Feature Selection**

All the models in Table 1 utilizes all the features from the dataset produced in the previous assignment. Thus, this section will explore if all these features are really necessary and may be dropped by on 3 metrics. These metrics are obtained from conducting logistic regression using *statsmodel* and fitting a tuned random forest model onto the datasets. After all, random forest machines are known for its ability to select significant features based on their features importance.

The metrics are the dependent variables’ coefficients, p-value, and feature importance. The threshold set for each of these metrics are as such, where if it falls within the range the feature would be considered insignificant and therefore should be dropped.

Thresholds for coefficients, p-value, and feature importance, where *x* is the value for each feature:

* Coefficients : -0.1 < *x* < 0.1
* P-value: *x* > 0.05
* Feature Importance : *x* < 0.05

**Table 2**

*Comparison of columns deemed to be insignificant by the 3 metrics, derived from Appendix 2.*

|  |  |  |
| --- | --- | --- |
| Coefficients : -0.1 < *x* < 0.1 | P-value: *x* > 0.05 | Feature Importance : *x* < 0.05 |
| Education  Age  Length of service  Average Training Score | Education  In region 26  In Legal department | Education  In region 26  In Legal department  In Tech department  In region 4  No. of trainings  In HR department  Age  Length of service |

Out of fear that dropping all these columns end up losing some valuable information, the features selected to be dropped will be those that at least appear twice among the 3 columns in Table 2. Thus, the following features would be dropped when conducting one more round of training and testing models: *'education', 'region\_region\_26', 'department\_Legal', 'age'*, and *'length\_of\_service'*.

**Table 3**

*Comparison of all the tuned and trained models’ accuracy on the mismatched training and testing datasets after dropping insignificant features. The models also underwent the same type of cross validation as the models used for Table 1.*

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm used | Training Dataset | Testing Dataset | Difference in Accuracy |
| Random Forest | 0.77 | 0.77 | 0 |
| ANN / MLP | 0.74 | 0.71 | -0.03 |
| XGBoost | 0.78 | 0.77 | -0.01 |
| LightGBM | 0.79 | 0.78 | -0.01 |
| CatBoost | 0.77 | 0.78 | +0.01 |

Moving forward, the algorithm “Logistic Regression” will no longer be tested, due to its weak base performance. Table 2 showcases minimal improvement in any of the models. From this, it can be inferred that there may not be any need to drop any features in the first place.

However, more needs to be done in optimizing for the ideal classification model.

## Model Optimization

At this point of the project, the issue with the matching rows in the datasets was identified and rectified, albeit too late that the entire project could not be redone. In fact, the ideal hyperparameters used for the models in the following section were all derived from the cross validation done onto the mismatched datasets. After all, it would take too long to redo all of the prior optimization that took over 2 weeks to complete, with each *GridSearchCV* taking around 3-4 hours

each to complete. Regrettably, only now onwards was the dataset used to train models properly cleansed and transformed.

To further optimize the models, AdaBoost is applied on each of the tuned models and once again compared. The parameters used to conduct cross validation onto the boosted models are in Appendix 1.

**Table 4**

*Comparison of all the boosted tuned models after undergoing cross validation on their train and test accuracies.*

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm used | Training Dataset | Testing Dataset | Difference in Accuracy |
| Random Forest | 0.89 | 0.81 | -0.08 |
| XGBoost | 0.50 | 0.49 | -0.01 |
| LightGBM | 0.75 | 0.76 | +0.01 |

Shockingly, only one of the models showed significant improvements, even though it can be argued that the very model is overfitted, with a difference in training and testing accuracy of 0.08. Perhaps more can be done to obtain a better model, that do not exhibit the same amount of overfitting. Otherwise, the boosted random forest classifier may be the best model so far.

With over 14 unique models that underwent too many hours of cross validation, the last step of optimization will be to use the best performers and to place them into an ensemble. Essentially meta-modelling, a voting classifier is used alongside a diverse range of well-trained and well-tuned models.

**Table 5**

*Comparison of all the voting classifier using the lightGBM model and the AdaBoosted Random Forest model, which have been highlighted in yellow in the previous tables.*

|  |  |  |  |
| --- | --- | --- | --- |
| Type of voting | Training Dataset | Testing Dataset | Difference in Accuracy |
| Soft | 0.86 | 0.82 | -0.04 |
| Hard | 0.88 | 0.81 | -0.07 |

The soft voting classifier has shown an ideal balance in both the bias and variance present in the trained model. As seen in Table 5, the soft voting classifier has high accuracies (both above 80%) and minimal difference between them of a 0.04. Thus, this soft voting classifier is the best so far and will be used for the final model.

## Summary

4 main steps were taken in building and optimizing a classification machine learning model. They essentially were: hyperparameter tuning, feature selection & extraction, boosting, and using voters.

After stratifying the dataset, StratifiedKFold was used for every cross validation done. From the multiple trial and errors conducted, the following model seen in Figure 1 performs the best when fitted. It returns a training accuracy of 86% and a testing accuracy of 82%, which showcases a balance in both low bias and low variance present in the model.

**Figure 1**

*The final model’s performance visualized in a Confusion Matrix.*

Graphical user interface

Description automatically generated with medium confidence

This performance is achieved through:

* A soft voting classifier
  + AdaBoost of a learning rate of 0.08 and with 60 estimators
    - On a Random Forest model with 30 estimators and a max tree depth of 9
  + LightGBM classifier with 40 estimators, 0.13 learning rate, regularization values for alpha of 0.03 and for lambda of 0.06.

# Airbnb

The target variable is **the approximate price an Airbnb listing is for a night**. The model trained only takes a subset of the entire dataset, where the Airbnb prices range from *$0 - $400*.

The dataset used to build the model only included records of listings that ranged from $0 to $400 to prevent it from predicting excessive prices for just one night stay. It's important to note that not many locations are expected to be too expensive, with often the more expensive listings being done by local hotels in central or downtown areas. However, the good news is that this dataset was properly cleaned, ensuring accuracy in the results.

## Approaches

To reassemble the dataset, the same cleansing and transformation techniques used in the previous assignment will be utilized. Additionally, the same four steps taken to optimize the classification problem will be considered to solve this regression problem. The same few algorithms will be taken into consideration, along with any additional ones that are specific to regression problems. For example, other algorithms such as Ridge, LASSO, and ElasticNet was used too.

Additionally, as a regression problem, there is no 1 evaluation metric that can easily encompass the performance of a model. Thus, 2 main metrics were used to evaluate a model’s performance in reference to the amount of bias and variance it exhibits. These metrics are the Root Mean Square Error (RMSE) and the R^2 value respectively.

## Model Building

Prior to building the model, the dataset was once again cleaned, transformed, and scaled, using the same methods from the previous assignment. This is to ensure that the records do not get mixed up, in a sense where the dependent features do not actually result or are related to the pricing of the Airbnb listing. This process will also ensure that the dataset is properly prepared and optimized to produce accurate results. Adding on, all the models have had their hyperparameters tuned in a way to minimizes the RMSE value, which hopefully reduces the amount of bias the model will have.

**Table 6**

*Comparison of all the tuned and trained models’ RMSE values and R^2 values on the training and testing datasets.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm used | Train RMSE | Test RMSE | Train R^2 | Test R^2 |
| Naïve Baseline | 64.587 | 63.463 | 0.442 | 0.434 |
| Linear Regression | 65.367 | 64.331 | 0.419 | 0.429 |
| Ridge | 317.875 | 239.914 | 0.215 | 0.097 |
| LASSO | 327.353 | 241.353 | 0.167 | 0.086 |
| ElasticNet | 317.960 | 239.744 | 0.215 | 0.098 |
| Random Forest | 38.736 | 50.939 | 0.800 | 0.636 |
| ANN / MLP | 53.070 | 56.679 | 0.634 | 0.549 |
| XGBoost | 41.704 | 57.630 | 0.768 | 0.533 |
| LightGBM | 36.838 | 49.164 | 0.819 | 0.660 |
| CatBoost | 41.704 | 57.630 | 0.768 | 0.533 |

In Table 6, the same algorithms were used to train and test some models. Unfortunately, all the models showcase overfitting, with a vast difference between the metrics on the training dataset and the testing dataset. Additionally, the models trained using the linear regression algorithm and those algorithms similar to it did not perform very well, so moving forward they would not be considered in future steps. On the flip side, the LightGBM and Random Forest models seem to be performing quite well with reasonable values for their RMSE and R^2 evaluation, albeit quite overfitted.

After all, considering the range and the values that the pricing of an Airbnb listing could be, an error of around $30- $40 is still reasonable, and can even be attributed to external circumstances. For example, someone who booked an Airbnb could be easily charged extra $50 due to seasonal demand, surcharge for cleaning, or even fees for the extra amenities the listing may provide. All of this are not detailed in the dataset used to train, but they would have an undeniable impact on a listing’s price.

Thus, the problem area would mostly be tackling how overfitted is instead, as opposed to minimizing the RMSE and R^2 values.

**Feature Selection**

In hopes to minimize overfitting, only a subset of the features was trialed to be used for training some models with their hyperparameters tuned. By only utilizing significant features, the end goal is for the trained models to have a perfect balance of low bias and low variance while avoiding being overfitted.

The same metrics used for the classification problem were employed to identify insignificant features in the dataset. This was done using a random forest regressor and a model that was trained using the ordinary least squares method through *statsmodel*. These methods allowed for a consistent and thorough analysis of the features, ensuring that only the most relevant information was used to make predictions.

**Table 7**

*Comparison of columns deemed to be insignificant by the 3 metrics, derived from Appendix 2.*

|  |  |  |
| --- | --- | --- |
| Coefficients : -0.1 < *x* < 0.1 | P-value: *x* > 0.05 | Feature Importance : *x* < 0.05 |
|  | Neighborhood group  Latitude  Longitude  Minimum nights  Number of reviews  Reviews per month  Availability / 365 | Neighborhood group  Neighborhood  Number of reviews  Number of reviews per month  Days since last review |

Seen in Table 7, the following features have been identified to be quire insignificant and do not contribute much to the trained model when predicting the prices. These are*: "last\_review", "neighbourhood\_group", "number\_of\_reviews",* and "*reviews\_per\_month".*

Table 8 explores the models that were trained using the selected subset of features, that were not deemed insignificant and were dropped.

**Table 8**

*Comparison of all tuned and trained models’ RMSE values and R^2 values on the training and testing datasets after dropping insignificant features. The models also underwent the same type of cross validation as the models used for Table 7.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm used | Train RMSE | Test RMSE | Train R^2 | Test R^2 |
| Random Forest | 39.026 | 50.411 | 0.796 | 0.643 |
| ANN / MLP | 54.259 | 56.920 | 0.606 | 0.545 |
| XGBoost | 42.958 | 58.130 | 0.753 | 0.525 |
| LightGBM | 37.446 | 48.759 | 0.813 | 0.666 |
| CatBoost | 44.313 | 51.387 | 0.738 | 0.629 |

The results of the analysis were found to be unsatisfactory, as there were no major positive differences observed between retaining or dropping the features. As a result, it has been decided that no features will be dropped moving forward. However, another major issue that was identified was that the models were mostly overfitted, indicating a need for further optimization and improvement. Overall, these challenges highlight the need for additional work to be done in order to achieve more accurate results.

## Model optimization

For further optimization, AdaBoost was once again employed and applied onto some models tuned with the ideal hyperparameters, in hopes to minimize overfitting. This AdaBoost also underwent some cross validation, to find the ideal number of estimators and learning rate, which were then evaluated in Table 9. In Table 10, voting regressors were explored, utilizing the best performing models so far.

**Table 9**

*Comparison of all the boosted tuned models after undergoing cross validation on their RMSE and R^2 values, while still being trained by the datasets.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm used | Train RMSE | Test RMSE | Train R^2 | Test R^2 |
| Random Forest | 36.986 | 49.682 | 0.817 | 0.653 |
| XGBoost | 33.381 | 49.749 | 0.851 | 0.652 |
| LightGBM | 33.437 | 48.822 | 0.851 | 0.665 |

**Table 10**

*Evaluation of the Voting Regressor using the models that were highlighted in yellow in this section.*

|  |  |  |  |
| --- | --- | --- | --- |
| Train RMSE | Test RMSE | Train R^2 | Test R^2 |
| 34.415 | 48.719 | 0.842 | 0.667 |

## Summary

Adapting the 4 steps used for the classification problem, a model was trained in hopes for low bias and low variance. With the following evaluated results, the final model is trained using AdaBoost and LightGBM:

* RMSE Train & Test : 34.415 & 48.719
* R^2 Train & Test : 0.842 & 0.667

# Conclusion

The report focuses on two datasets: HR analytics dataset (classification) and Airbnb prices dataset (regression). The goal is to find the optimal model for each case by conducting multiple trials and optimizations. The algorithms used throughout *were Regression, Random Forest, Artificial Neural Networks, XGBoost, LightGBM, and CatBoost*. The author then explored dropping some features based on the metrics of coefficients, p-value, and feature importance derived from conducting analysis based on the trained regression and trained random forest models. The project also sought to further optimize the models by applying AdaBoost and creating a Voting Ensemble. The highest performing models are finally chosen after all the trials from these 4 steps, which was evaluated based on each models’ respective metric that works well for their problem’s contextual background.

The ideal model for the HR Analysis has been found to be trained as such:

* A soft voting classifier
  + AdaBoost of a learning rate of 0.08 and with 60 estimators
    - Base : Random Forest Classifier
      * Number of estimators : 30
      * Max tree depth : 9
    - Learning rate : 0.08
    - Number of estimators : 60
  + LightGBM Classifier
    - Number of estimators : 40
    - Learning rate : 0.13
    - Alpha regularization : 0.03
    - Lambda regularization : 0.06
    - Importance type : “gain”

For the Airbnb listings’ prices, the model that performs well enough with a bit of overfitting would be:

* AdaBoost Regressor
  + Base : LightGBM Regressor
    - Importance type : “gain”
    - Number of estimators : 150
    - Alpha Regularization : 0.05
  + Learning Rate : 0.02
  + Loss : “exponential”
  + Number of Estimators: 140

# Reflection

## Improvements

Undeniably, there is still some room for improvement in this entire project. This includes, but not limited to:

* Ensuring the datasets were properly cleansed and transformed at first.
* Stratifying the classes in the HR dataset to be of other ratios, e.g., 1:2, 1:3, 1:4, etc.
* Using the proper datasets for training any model.
* Having a more comprehensive parameter grid for cross validation.
* Using more folds for cross validation.
* Using other evaluation metrics for the classification problem (e.g., Matthew’s Correlation Coefficient).
* Utilizing other methods to clean the original datasets.
  + Exploring other methods for missing value imputation.
  + Trying out more methods for categorical encoding.
  + Binning the continuous variables.
  + Using other scaling methods like Min-Max.
* Limiting the data for the regression problem even further:
  + Grouped by the neighborhood group.
  + Grouped by the neighborhood.
  + Grouped by the room type.
* Creating new features through feature extraction.
* Experimenting with bagging techniques.
* Applying XGBoost the same way AdaBoost was used in this report.
* Etc.

## Module Conclusion

This module spelt out the universal workflow of machine learning to me really plainly, where it encompasses several key steps such as: problem definition, data collection and preparation, model selection and training, evaluation, and deployment. The module also went through multiple frameworks one could follow, such as CRISP-DM and SEMMA. Thanks to these structured approaches, it was very easy to systematically go though the workflow of creating a machine learning solution to a given problem statement. Additionally, compared to the previous modules, this modules went above and beyond to go in-depth of the different algorithms and concepts used in Machine Learning, with detailed explanations and theory given to me. I especially found the additional materials useful that showcases how the complex machine learning algorithms works, such as the slides detailing the steps an artificial neural network (ANN) and a Support Vector Machine (SVM). Other concepts such as how the p-value is calculated and related to a feature’s significance really piqued my interest as a mathematics lover, so I really appreciated those lessons in particular. In fact, this module provided me with more fine tuning techniques that I was not originally aware of, such as regularization, hyperparameter tuning, and meta-modelling with ensembles.

Unfortunately, all this knowledge meant I have more things to be aware of when modelling, such as whether I can perfectly balance out the bias-variance tradeoff. This in particular, and concepts regarding bagging and boosting, I find I am struggling a bit still to grasp these ideas, the proper use cases for them, and how to even implement them well as part of my workflow, as seen in this report.

## Discussions

The following figures showcases my participation in the online discussion forums, and my answers to my classmates’ queries.

**My Questions :**

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application

Description automatically generated**

**My Answers to my classmates’ queries :**

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

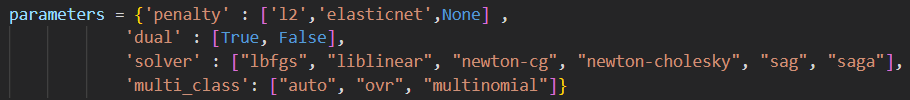
Description automatically generated

# Appendix

## Appendix 1 : The parameters used for *GridSearchCV* for every algorithm.

For HR Analytics:

Logistic Regression



Random Forest

Text

Description automatically generated

Artificial Neural Networks / Multi-layer Perceptron

A screenshot of a computer

Description automatically generated with medium confidence

XGBoost

Text

Description automatically generated

LightGBM (Light Gradient Boosting Machine)

Text

Description automatically generated

CatBoost

Text

Description automatically generated

AdaBoost

Text

Description automatically generated

For Airbnb Prices:

Linear Regression

Text

Description automatically generated

Random Forest

Graphical user interface, text

Description automatically generated

Artificial Neural Networks / Multi-layer Perceptron

A screenshot of a computer

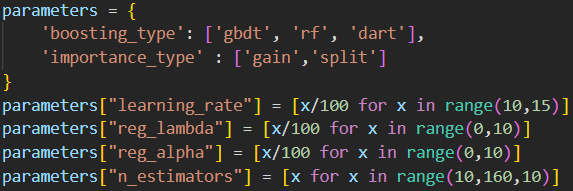
Description automatically generated with medium confidence

XGBoost

Text

Description automatically generated

LightGBM (Light Gradient Boosting Machine)



CatBoost

**Text

Description automatically generated**

## Appendix 2 : The values from the statsmodels’ fitted models and the features’ importance from the random forest model.

For Classification problem:

Graphical user interface, text

Description automatically generated

**Chart, bar chart

Description automatically generated**

**Text

Description automatically generated**

For Regression problem:

**Graphical user interface, text

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Text

Description automatically generated**