Data Mining Exercise 3

Magdalena König

01455794

1. Labor negotiation

- Task 1: Data cleaning & exploration

```
Missing Percentage before cleaning
profession         0.000000
duration           1.833333
wage1              3.666667
wage2             21.500000
wage3             72.166667
cola              41.166667
hours              8.000000
pension           49.833333
stby_pay          73.166667
shift_diff        43.000000
educ_allow        44.000000
holidays           4.333333
vacation           7.333333
lngtrm_disabil    59.666667
dntl_ins          37.833333
breavement        48.833333
Empl.hplan        35.666667
consent            0.000000
```

- all columns that have a value of more than 40% missing values will be deleted

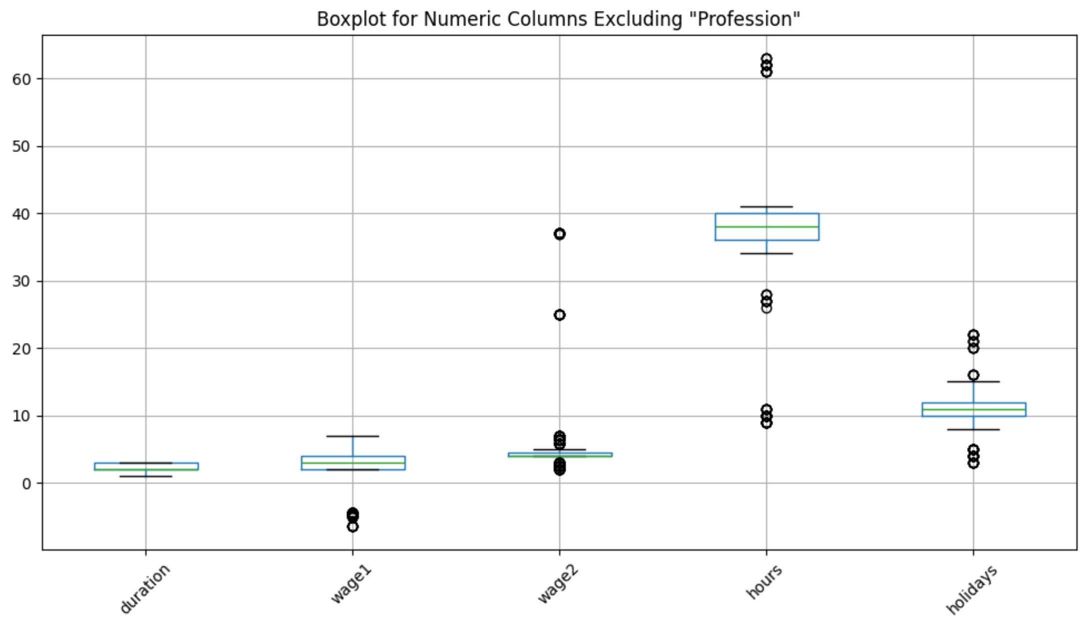
```
Missing Percentage after cleaning
profession     0.000000
duration       1.833333
wage1          3.666667
wage2         21.500000
hours          8.000000
holidays       4.333333
vacation       7.333333
consent        0.000000
```

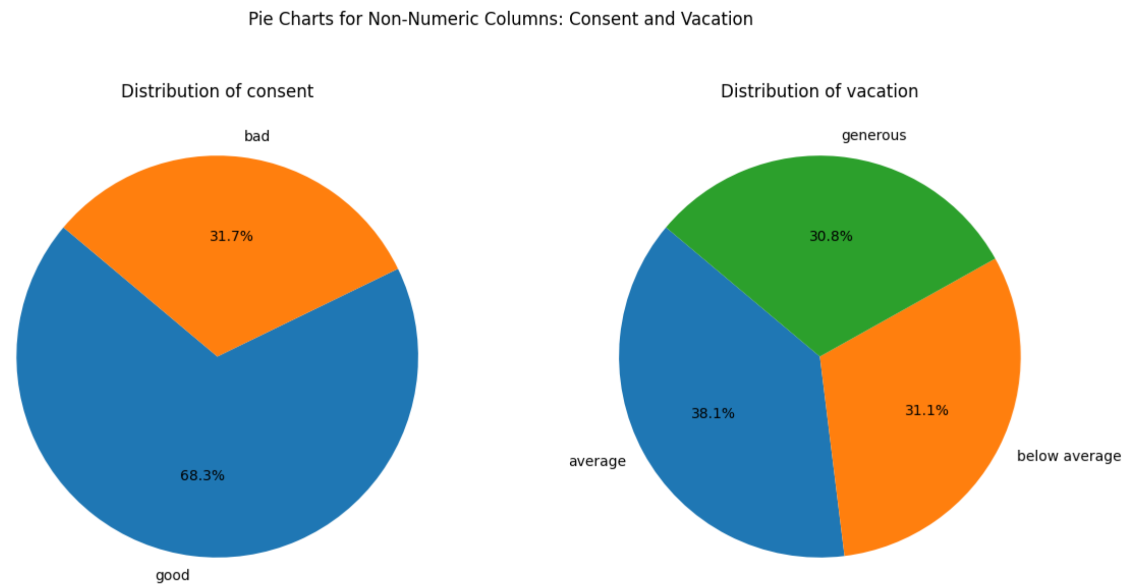
- The still missing values will be filled with the medium (numerical columns) and most frequent ones (categorical columns)

Rows with unexpected values are deleted in the categorical columns (and consent).

Also those rows are deleted that don't have a unique professions in order to identify the workers clearly:

```
Number of unique professions: 546, Total rows: 549
Non-unique professions: [501 591 339]
Number of rows after removal: 546
```





- Task 2: Classification

Logical Regression was chosen here → statistical method to determine the properbility of the occurance of a dependent variable (consents) in combination with one or more independet factors (vacation, hours, wage,...).

- The training data was unbalanced (consent 0 (bad/unaccepted) was way less frequent than 1 (good/accepted).

When unbalanced there was the following result:

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        89
           1       0.68      1.00      0.81       186

    accuracy                           0.68       275
   macro avg       0.34      0.50      0.40       275
weighted avg       0.46      0.68      0.55       275

Average Accuracy: 0.6763636363636365
Average Precision: 0.6763636363636365
Average Recall: 1.0
```

Results with balancing strategies:

- SMOTE (SMOT creates synthetic training data for the underrepresented groups)

```
With SMOTE:
              precision    recall  f1-score   support

           0       0.36      0.55      0.43        89
           1       0.71      0.53      0.60       186

    accuracy                           0.53       275
   macro avg       0.53      0.54      0.52       275
weighted avg       0.60      0.53      0.55       275

Average Accuracy: 0.5345454545454544
Average Precision: 0.710144927536232
Average Recall: 0.5268817204301076
```

- Oversampling (Oversample the bad consent rows so its more balanced)

```
With Oversampling:
              precision    recall  f1-score   support

           0       0.34      0.56      0.42        89
           1       0.69      0.47      0.56       186

    accuracy                           0.50       275
   macro avg       0.52      0.52      0.49       275
weighted avg       0.58      0.50      0.52       275

Average Accuracy: 0.5018181818181818
Average Precision: 0.6929133858267715
Average Recall: 0.4731182795698924
```

- Undersampling (undersample the overrepresented data – good consent)

```
With Undersampling:
              precision    recall  f1-score   support

           0       0.34      0.58      0.43        89
           1       0.70      0.46      0.56       186

    accuracy                           0.50       275
   macro avg       0.52      0.52      0.49       275
weighted avg       0.58      0.50      0.52       275

Average Accuracy: 0.5018181818181818
Average Precision: 0.6991869918699187
Average Recall: 0.4623655913978494
```

- To see a good effect every model was ran 10 times (the classification report was computed at the last run) – and the average accuracy, precision and recall was taken.
- The balanced models have a worse accuracy than the unbalanced one, however the model using SMOTE showed the best precision – a balanced sample would surely make sense here, even if the accuracy is lower.

2. Flight data

- Task 1: Data cleaning and preperation
    - Cancelled flights are deleted (rows)
    - The column cancelled are deleted
    - Each delay column was transferred from NaN to 0 (however I could have deleted those ones from the beginning)
    - Rows with missing values were deleted

```
Total number of rows flight_data: before and after 1348838 1318351
```

    - Later after the merge there was some more data cleaning necesarry (since the merging was a really time-consuming process it was done after merging in a second data cleaning step
    - Rows with the vlaue “M“ (for missing) were set to NaN
    - All Rows with NaN values were deleted as well as those columns deleted that had a precentage of more than 30% of missing values
    - Time and Date (date and wheels off) is converted to datetime and cleaned if necessarry + a new column for these values
    - Weather stations’s valid column is also converted to datetime
    - Unnecesarry weather column was dropped (lan and lat)
    - Wheels off times and the time from the weatherstation are rounded to 10 minutes in order to make them match better

- Task 2: Merging

```
#merge airport

flight_data = flight_data.merge(airport_data, left_on='ORIGIN', right_on='iata_code', suffixes=('', '_dep'))
flight_data = flight_data.merge(airport_data, left_on='DEST', right_on='iata_code', suffixes=('', '_arr'))
flight_data.rename(columns={'iana_tz': 'iana_tz_dep', 'windows_tz': 'windows_tz_dep', 'iata_code': 'iata_code_dep'}, inplace=True)
print("Checkpoint 1")

#Merge timezones

flight_data = flight_data.merge(timezones_data, left_on='iana_tz_dep', right_on='timezone', suffixes=('', '_dep'))
flight_data = flight_data.merge(timezones_data, left_on='iana_tz_arr', right_on='timezone', suffixes=('', '_arr'))
flight_data.rename(columns={'timezone': 'timezone_dep', 'offset': 'offset_dep', 'offset_dst': 'offset_dst_dep'}, inplace=True)
print("Checkpoint 2")

#Merge weather data

flight_data = flight_data.merge(weather_data, left_on=['ORIGIN', 'datetime_dep'], right_on=['station', 'valid'],
                                suffixes=('', '_weather_dep'))
```

The merging was incredibly time consuming and took about 2h.

~5100 rows were in the merged dataset

- Task 3 – Classification
  - First step was to devide to determine which features make sense
  - Then they were divided into numerical and categorical features

```
numerical_features = [
    'CRS_DEP_TIME', 'DEP_TIME', 'TAXI_OUT', 'DISTANCE',
    'tmpf_dep', 'dwpf_dep', 'relh_dep', 'drct_dep', 'sknt_dep',
    'p01i_dep', 'alti_dep', 'vsby_dep', 'feel_dep', 'skyl1_dep'
]

categorical_features = [
    'OP_UNIQUE_CARRIER', 'TAIL_NUM', 'OP_CARRIER_FL_NUM',
    'ORIGIN', 'DEST', 'skyc1_dep'
]
```

After that the information gain of every feature was determinded with the following result:

```
num__CRS_DEP: 0.4202
num__DEP: 0.2662
num__TAXI: 0.1194
cat__TAIL_NUM: 0.0393
cat__OP_CARRIER_FL_NUM: 0.0333
num_: 0.0175
cat__ORIGIN: 0.0161
cat__DEST: 0.0160
num__skyl1: 0.0121
num__sknt: 0.0097
num__drct: 0.0095
num__vsby: 0.0089
num__p01i: 0.0079
cat__skyc1_dep: 0.0052
cat__OP_UNIQUE_CARRIER: 0.0050
num__relh: 0.0045
num__feel: 0.0034
num__alti: 0.0032
num__dwpf: 0.0027
num__tmpf: 0.0000
```

- After that a decision tree classification was created – first with all features, no matter which information gain the have:

```
Accuracy on test data with all features: 0.9137254901960784
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.96      0.95       845
           1       0.77      0.71      0.74       175

    accuracy                           0.91      1020
   macro avg       0.86      0.83      0.84      1020
weighted avg       0.91      0.91      0.91      1020
```

- All features with a information gain with more than 0.08:

```
Accuracy on test data with the features with a information gain over 0.08: 0.9166666666666666
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.96      0.95       845
           1       0.77      0.73      0.75       175

    accuracy                           0.92      1020
   macro avg       0.86      0.84      0.85      1020
weighted avg       0.91      0.92      0.92      1020
```

- The best 5 features according to their information gain:

```
Accuracy on test data with all features that have the top 5 information gains: 0.9186274509803921
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.96      0.95       845
           1       0.79      0.71      0.75       175

    accuracy                           0.92      1020
   macro avg       0.87      0.84      0.85      1020
weighted avg       0.92      0.92      0.92      1020
```

- The worst 5 features according to their information gain:

```
Accuracy on test data with the worst 5 features: 0.7852941176470588
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.87      0.87       845
           1       0.37      0.35      0.36       175

    accuracy                           0.79      1020
   macro avg       0.62      0.61      0.62      1020
weighted avg       0.78      0.79      0.78      1020
```

When it comes to Accuracy the model with the worst features is showably lower than the other three. Accuracy- and precision-wise the model with only the top 5 features as parameters is the best, however only slightly. Between all features as parametes and only those over an information gain of 0.08 there are almost no differences. However what's also worth meantioning is that the dataset is not balanced and with a balancing strategy the result could be different.