Incident management process enriched event log

Shuyu CHEN

1 Introduction

This event log was extracted from data gathered from the audit system of an instance of the ServiceNow platform used by an IT company and enriched with data loaded from a relational database.

It contains the entire process of incident management, including key points in time, Id of responsible persons, etc.

The dataset contains 141712 events of 24918 incidents.

2 Type of variables

36 variables in total:

5 time variables

15 different categories of ids

4 Boolean variables

3 int variables

9 categorical variables

3 Target

- Change all the time variable to timestamp
- Target variable is generated by subtracting “sys_updated_at” variable from “closed_at” variable

4 Label encoder

- Regression would be used, so the data type should be numeric, so I used label encoder to change category and bool type data to numeric.

```
from sklearn.preprocessing import LabelEncoder
```

```
def convert(data,column):
    number = LabelEncoder()
    data[column] = number.fit_transform(data[column])
    data=data.fillna(-999)
    return data
```

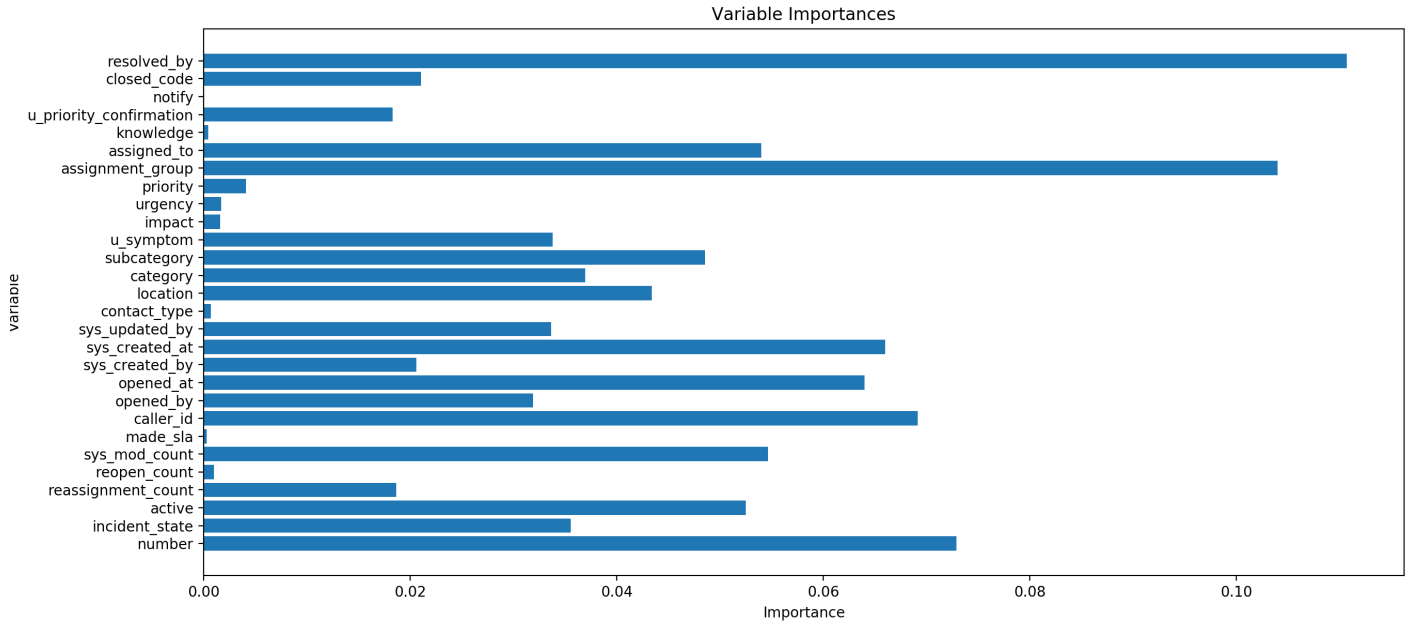
4 Methods

- Decision tree regression
- Random forest regression
- Grid Search

5 feature engineering

- First, I used the dataset with “resolved_at” column. The score is about 0.96 but the feature importance showed that “resolved_at” variable is the most important.
- So, I deleted “resolved_at” column, and build a new random forest model. But the score was about 0.78.
- Then I deleted all columns whose importance equal 0, and built a new model, then the score increased to 0.96.

6 feature importance



Final chosen variables:

- number
- active
- sys_mod_count
- opened_by
- sys_created_by
- sys_updated_by
- category
- u_symptom
- assigned_to
- closed_code
- resolved_at

7 result

Result without feature engineering

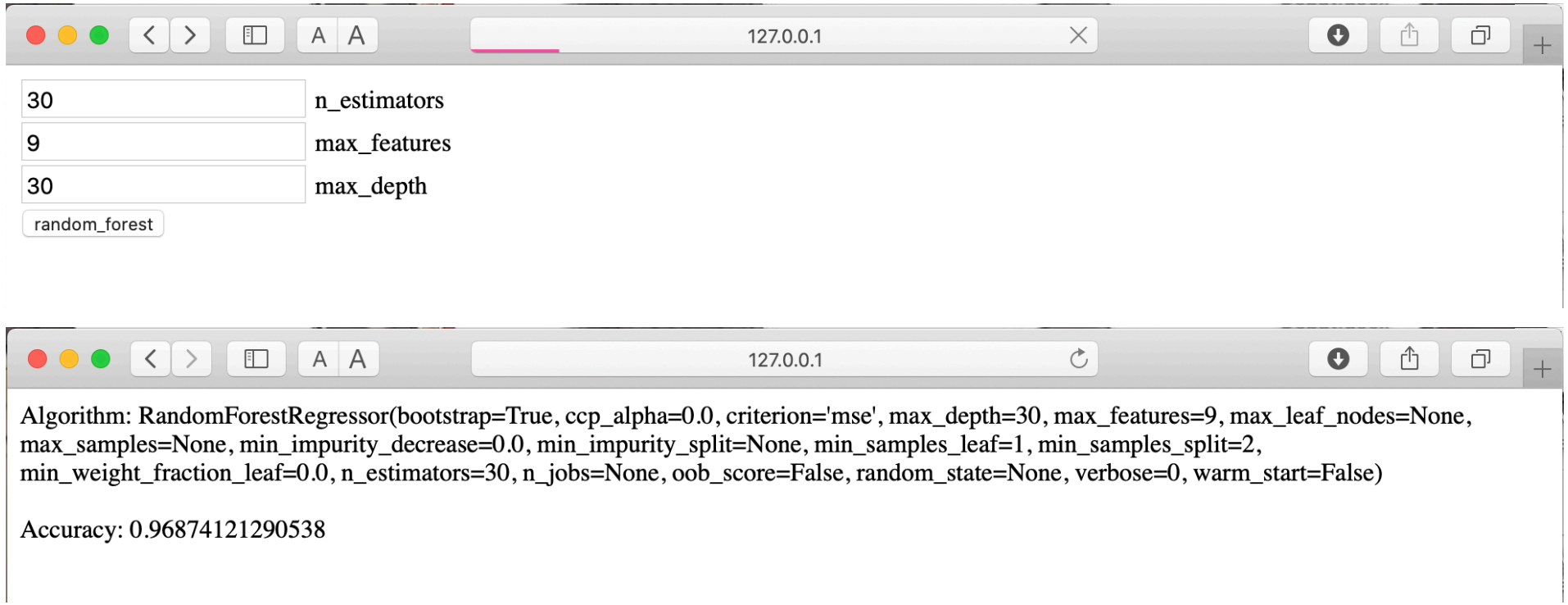
```
(0.7711257408991313,
 RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=50,
                       max_features=28, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=50,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False))
```

Result with feature engineering

```
(0.9690632635369949,
 RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=50,
                       max_features=20, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=30,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False))
```

8 Flask API

Choose the hyperestimators



Thank you for your evaluating