Classify2TeX

Outline

- Target group and Classify2TeX specification
- What is under the hood?
    - Data preprocessing
    - Model Optimization
    - Data postprocessing
- Conclusion

Target group and Classify2TeX specification

Who is Classify2TeX for?

People

- ▲ both **beginners and experts**
- ▲ need a clear **LaTeX report(tex, pdf)** with insights
- ▲ want to get an **explainable** model
- ▲ want to perform **fast or prolonged** model selection

Data

- ▲ **tabular data**
- ▲ **unstructured text features are not particularly important**
- ▲ **binary classification**, target may be unbalanced

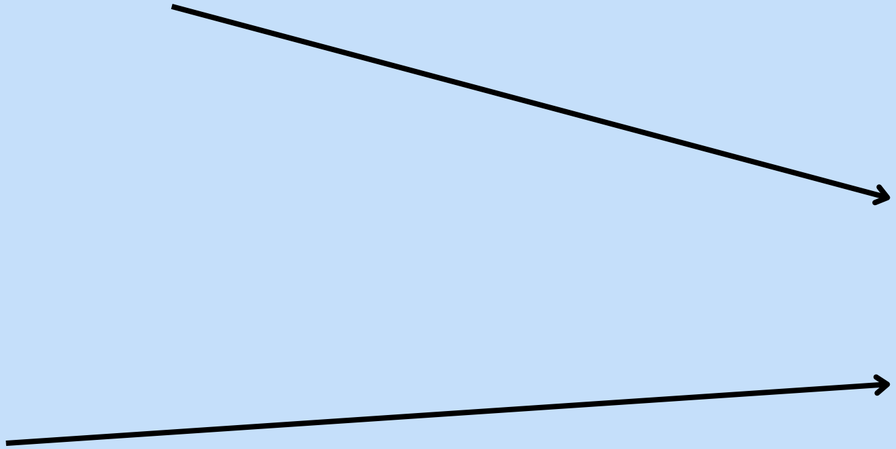
What is under the hood?

Data Preprocessing

- Extracting date-related components

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Date            3000 non-null   object
```

	Date
0	2013-12-31
1	2015-11-16
2	2014-01-22
3	2009-04-04



m	Date_year	Date_month	Date_day
.5	2013	12	31
.7	2009	4	4
.5	2009	1	25
.2	2009	11	18
.0	2009	1	22
...	...	...	...
.6	2013	10	2
.9	2010	5	22
.4	2012	1	23
.7	2016	3	14
.7	2013	11	30

Data Preprocessing

- Removing **index** and features with high percentage of missing values
- Handling **missing values** (median and most frequent)
- Handling **outliers** (Z-method)
- **Encoding categorical** features (label and one-hot-encoding)
- Removing **highly correlated** features (threshold=0.9)
- Handling class **imbalance** (SMOTE, undersampling/oversampling)

Data Preprocessing

IMPORTANT NOTE:

WE DON'T DO SCALING OF NUMERICAL DATA

REASONS:

We want to achieve maximal Explainabilty of models

We use tree-based models

Model selection

Classify2TeX supports building three types of models:

- **Random Forest**
- **Decision Tree**
- **XGBoost**

```
from Classify2TeX.classify_2_tex import Classify2TeX # Import the class

automl = Classify2TeX(dataframe = df, target_column_name='status') # Initialize the class with only the required arguments
automl.perform_model_selection() # preprocess the data and perform the model selection
```

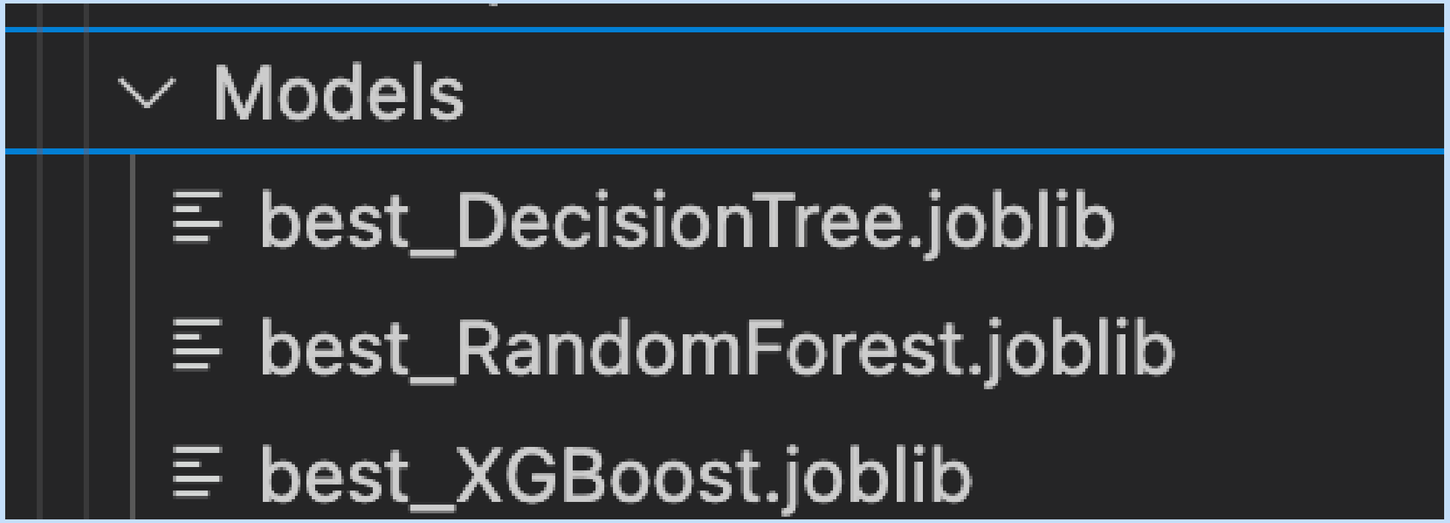
Process:

- Checks **default models** first of all
- Performs **random search** (n_iter = [0,0,0] or provided by user)

Model selection

User gets 3 best models in joblib extension.

“Best” means achieved highest metric (ROC AUC / F1 / ACCURACY)



User stays informed all the time

```
---------------Preprocessing the dataset------------------
---------------Extracting Day, Month and Year--------------
---------------Deleting redundant features-----------------
Feature: Date_hour has only one unique value. It will be removed.
Feature: Date_minute has only one unique value. It will be removed.
Feature: Date_second has only one unique value. It will be removed.
---------------Handling missing values---------------------
The missing values in the feature "MinTemp" are filled with the median value.
The missing values in the feature "MaxTemp" are filled with the median value.
The missing values in the feature "Rainfall" are filled with the median value.
The missing values in the feature "Evaporation" are filled with the median value.
The missing values in the feature "Sunshine" are filled with the median value.
The missing values in the feature "WindGustDir" are filled with the most frequent value.
The missing values in the feature "WindGustSpeed" are filled with the median value.
The missing values in the feature "WindDir9am" are filled with the most frequent value.
The missing values in the feature "WindDir3pm" are filled with the most frequent value.
The missing values in the feature "WindSpeed9am" are filled with the median value.
The missing values in the feature "WindSpeed3pm" are filled with the median value.
The missing values in the feature "Humidity9am" are filled with the median value.
---------------Handling outliers---------------------------
5 outliers in the feature MaxTemp were replaced with the median value.
12 outliers in the feature Humidity9am were replaced with the median value.
14 outliers in the feature Pressure3pm were replaced with the median value.
5 outliers in the feature Temp3pm were replaced with the median value.
--------------- Encoding categorical features --------------
----------Transforming boolean features to int-------------
--------------- Encode the target--------------------------
Target column was encoded as follows:
{'No': 0, 'Yes': 1}
------------Removing highly correlated columns ------------
Due to high correlation with other columns, the columns: ['Pressure3pm', 'Temp3pm', 'RainToday_Yes'] have been removed.
--------------- Handling imbalanced classes----------------
Moderate imbalance detected. Applying oversampling.
--------------- Dataset preprocessing is done--------------
```

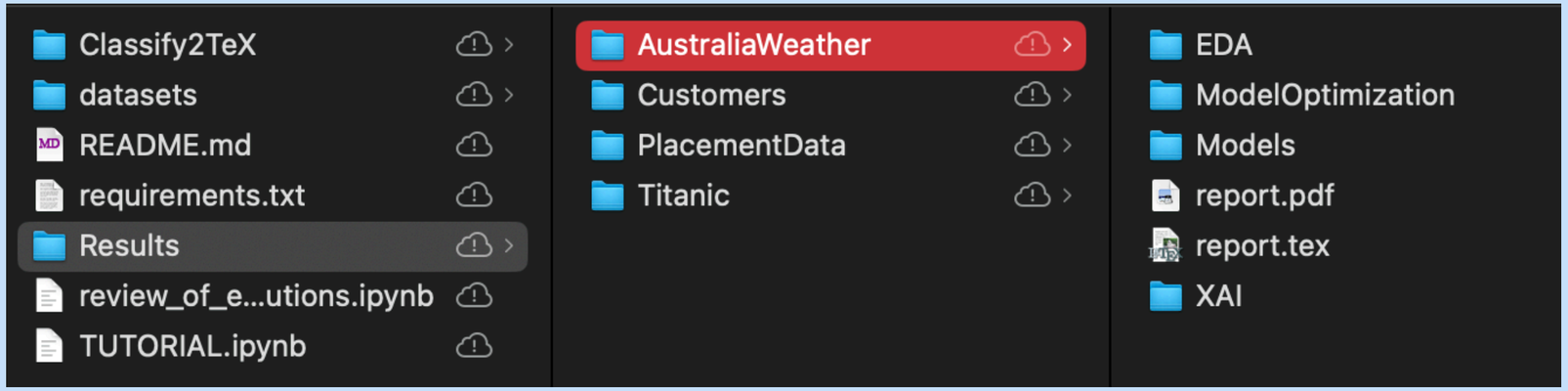
```
---Performing hyperparameter tuning for DecisionTreeClassifier...
Default model results: {'f1': 0.9136003521126761, 'accuracy': 0.913907
Checked another model, results using cross-validation: {'f1': 0.870311
Checked another model, results using cross-validation: {'f1': 0.871746
Checked another model, results using cross-validation: {'f1': 0.764410
Checked another model, results using cross-validation: {'f1': 0.684164
Checked another model, results using cross-validation: {'f1': 0.684164
Checked another model, results using cross-validation: {'f1': 0.794319
Checked another model, results using cross-validation: {'f1': 0.684164
---Performing hyperparameter tuning for RandomForestClassifier...
Default model results: {'f1': 0.9525131186806357, 'accuracy': 0.952538
Checked another model, results using cross-validation: {'f1': 0.775713
Checked another model, results using cross-validation: {'f1': 0.929510
Checked another model, results using cross-validation: {'f1': 0.825377
Checked another model, results using cross-validation: {'f1': 0.911152
Checked another model, results using cross-validation: {'f1': 0.920255
Checked another model, results using cross-validation: {'f1': 0.780788
Checked another model, results using cross-validation: {'f1': 0.833466
---Performing hyperparameter tuning for XGBoostClassifier...
Default model results: {'f1': 0.9436508759199752, 'accuracy': 0.943708
Checked another model, results using cross-validation: {'f1': 0.934776
Checked another model, results using cross-validation: {'f1': 0.889776
Checked another model, results using cross-validation: {'f1': 0.920615
Checked another model, results using cross-validation: {'f1': 0.882896
Checked another model, results using cross-validation: {'f1': 0.935223
Checked another model, results using cross-validation: {'f1': 0.928573
Checked another model, results using cross-validation: {'f1': 0.936110
```

Data postprocessing

```
automl.generate_report(dataset_name="AustraliaWeather") # Generate the report and Results folder
```

```
Report generated successfully.
```

Results folder and report

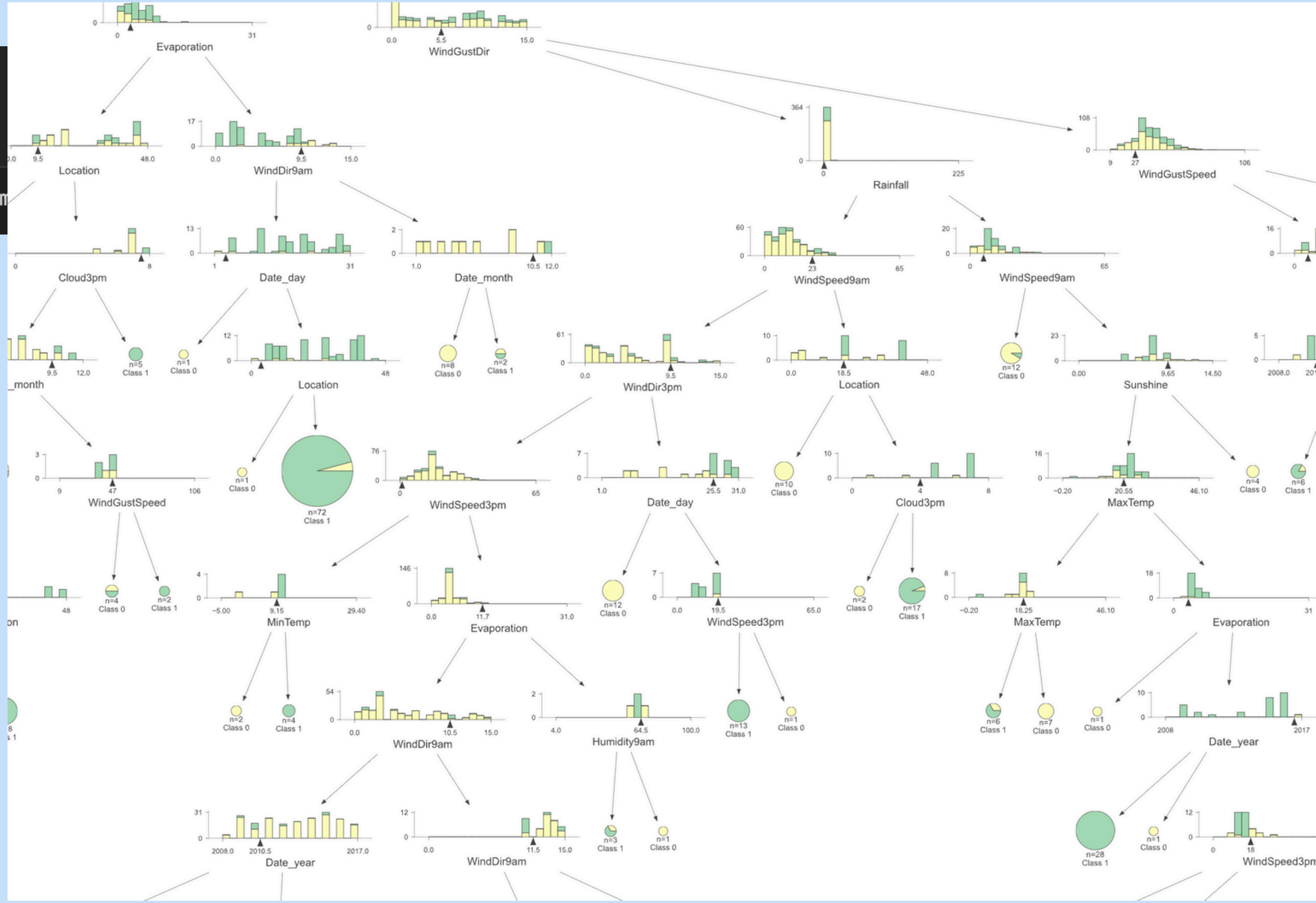


we will show it to you in a minute

Data postprocessing (best decision tree)

```
tree = automl.build_best_decision_tree()
tree.view()
```

Below you can see how the best Decision Tree m



Thank you for your attention