Отчёт по проекту по дисциплине "Основы машинного обучения"

Улучшение проекта, сделанного в предыдущем семестре

Добрецова Елизавета, группа 5030102/00101

Апрель 2024

Проект прошлого семестра

Отчёт по проекту, сделанному в прошлом семестре, можно посмотреть здесь.

Что можно улучшить и доделать?

- В реализации задачи предсказания доли переработанного пластика с учётом значений признаков сделать ввод этих значений не вручную, а путём чтения их из конфигурационного файла;
- Добавить Unit-тесты для функций предсказания;
- Исследовать, насколько влияет каждый из признаков на линейную регрессию;
- Нарисовать пайплайн не от руки, а средствами электронного редактора (наппример, MS Powerpoint);
- Подписать точки как в графиках PCA, так и в трёхмерных графиках;
- Сделать Leave-one-out оценку модели, обучив её на разных тренировочных данных.

Что нужно исправить:

- В файле .py надо работать с нормализованными значениями признаков, раньше было не так и это была ошибка.
- Также в этом файле по ошибке не были сдвинуты значения года на 2000, поэтому программа выдавала нереалистичные значения искомой доли.

Подробнее об ошибках реализации и их исправлении

Исправить ошибку с забытым сдвигом года (который вводился в основном для того, чтобы графики были визуально менее нагружено) было легко, но именно благодаря этому исправлению ответы программы стали реалистичными. Ранее они были в районе 300%.

Подробнее об ошибках реализации и их исправлении

Несколько сложнее оказалось исправить ошибку, связанную с работой с нормализованными данными в одном месте программы и работой с ненормализованными в другой. Дело в том, что перед применением PCA при обработке данных в Jupyter Notebook значения признаков прошли нормализацию, то есть из каждого из них вычли среднее и поделили результат на стандартное отклонение. А при подсчёте итогового результата в файле .py ошибочно использовались исходные, ненормализованные данные, что приводило к неверному ответу. Проблема была решена, для этого пришлось сохранять в локальную память средние значения и стандартные отклонения по каждому из признаков.

Улучшения: ввод из конфигурационного файла

Ранее при предсказании с учётом признаков значения признаков вводились с консоли вручную по очереди, что создавало пользователю большие неудобства. Теперь же в консоль надо ввести только путь к файлу .json, в котором хранятся значения в виде словаря. Также реализована «защита от дурака», то есть программа проверяет, что путь к файлу валидный и в словаре хранятся значения для всех ключей.

```
/Users/paks/PycharmProjects/pythonProject/venv/bin/python /Users/paks/Polytech/semester7/ML/machine_learning/src/prediction_from_previous_years.py
Input path to json file with coefficients:
/Users/paks/Polytech/semester7/ML/machine_learning/data/example_features.json
/Users/paks/Polytech/semester7/ML/machine_learning/src/prediction_from_previous_years.py:65: FutureWarning: Series.__getitem__ treating keys as posi
  tan += feature * linear_regression_coefficients.loc["Tangent"][i]
/Users/paks/Polytech/semester7/ML/machine_learning/src/prediction_from_previous_years.py:66: FutureWarning: Series.__getitem__ treating keys as posi
  bias += feature * linear_regression_coefficients.loc["Bias"][i]
tan =  0.15501097531805078 , bias =  3.8307201839791922
According to prediction with features, in 2030 the share of plastic recycled will be 8.481049443520716 %

Process finished with exit code 0
```

Улучшения: Unit-тесты

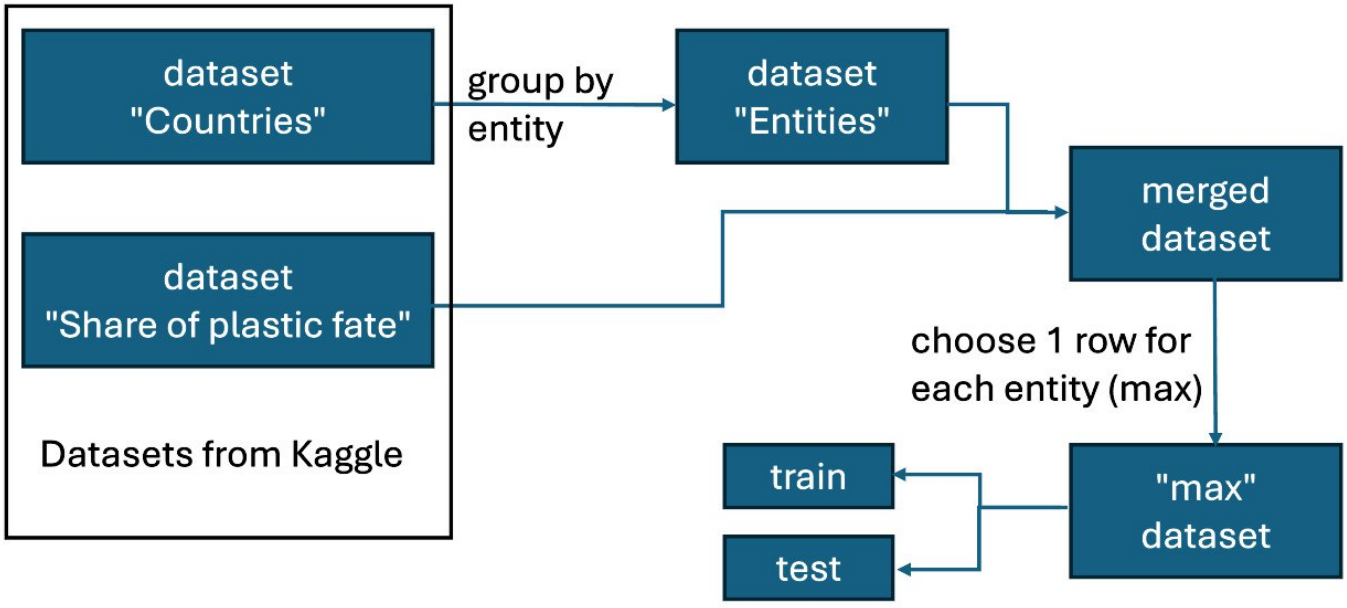
В проект был добавлен файл с тестовыми функциями, которые проверяют на различные условия значения, получаемые функциями условного и безусловного предсказания. Проверялось, что полученная доля в процентах больше 0, меньше 100, что программа падает при тех введённых значениях, когда она должна падать.

```python
def test_predict_share():
    assert np.isclose(predict_share_from_coefficients(2000, 0.3, 1.5), 1.5)
    assert 0 <= predict_share_from_coefficients(2030, 4, 2.0) <= 100
    assert predict_share_from_coefficients(2024, 0, 0) >= 0
    assert predict_share_from_coefficients(2024, -1, -1) >= 0


def test_predict_unconditionally():
    with pytest.raises(Exception):
        predict_unconditionally(2000, "Russia")  # Should fail: Russia is not in list of entities
    assert predict_unconditionally(2100, "United States") <= 100
    assert predict_unconditionally(1900, "Oceania") >= 0


def test_predict_with_features():
    void_df = pd.DataFrame()
    with pytest.raises(Exception):
        predict_with_features(2000, [], void_df)
```

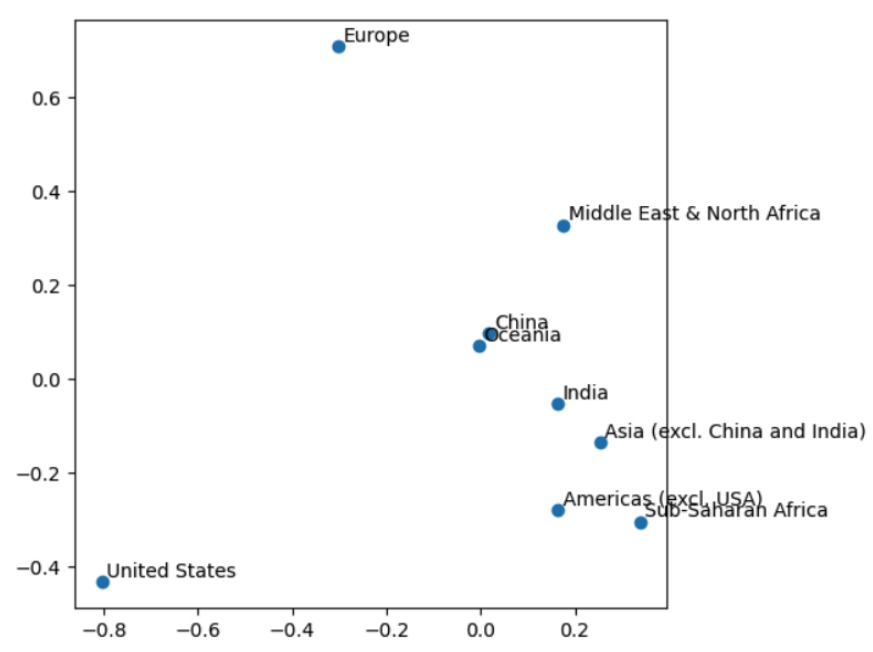
Наблюдение за влиянием каждого из признаков на линейную регрессию

Было интересно исследовать, в какой мере влияет каждый из рассматриваемых признаков (ВВП, расходы на образование, расходы на науку, экспорт, население и так далее) на итоговый ответ, получаемый линейной регрессией. Оказалось, что они все влияют примерно одинаково мало, основное влияние оказывает свободный член, или сдвиг.

Пайплайн


dataset
"Countries"
group by
entity
dataset
"Entities"
dataset
"Share of plastic fate"
merged
dataset
choose 1 row for
each entity (max)
Datasets from Kaggle
train
test
"max"
dataset

Графики PCA


Europe
Middle East & North Africa
China
Oceania
India
Asia (excl. China and India)
Americas (excl. USA)
Sub-Saharan Africa
United States
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.8
−0.6
−0.4
−0.2
0.0
0.2

Leave-one-out оценка (1)

Ранее выборка из 9 строк делилась на тренировочную и тестовую. 6 строчек из 9 вошли в тренировочную, в тестовую - все остальные. Тем самым осуществлялась некоторая подгонка под данные.

[18]:
```
#train_df = merged_df_2.loc[list(range(6))]
train_df = merged_df_2.loc[[0, 1, 3, 5, 6, 8]]
train_df
```

[18]:

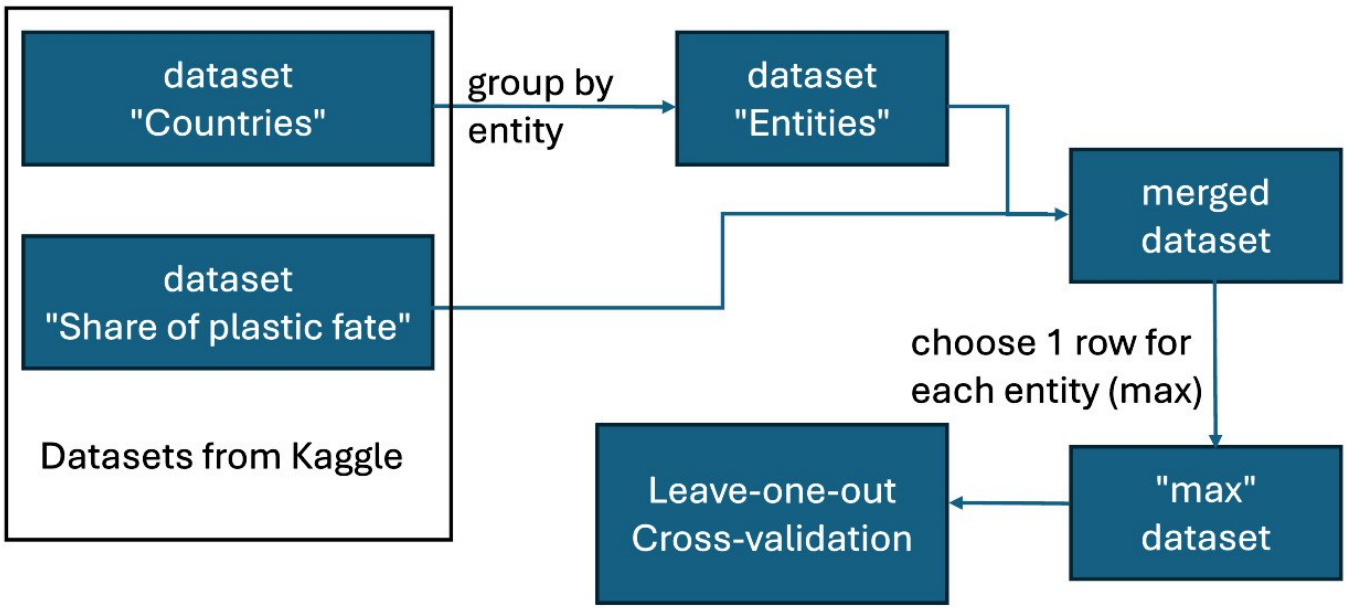
	Entity	Tan	Bias	GDP	R&D	Population	Land	Export	Education Expenditure	Health Expenditure	Net Trade
0	Americas (excl. USA)	0.254503	4.718604	8.246311e+12	7.699779e+10	6.830172e+08	3.072047e+07	1.963236e+12	1.274479e+12	6.504999e+11	1.179535e+11
1	Asia (excl. China and India)	0.237067	3.876733	1.410295e+13	2.901652e+11	1.679359e+09	3.155283e+07	5.097880e+12	1.774069e+12	1.012535e+12	4.201649e+11
3	Europe	0.424986	4.454272	2.065720e+13	4.315568e+11	6.011706e+08	6.323004e+06	9.576989e+12	2.332803e+12	2.015527e+12	6.484722e+11
5	Middle East & North Africa	0.136690	2.852235	3.838420e+12	4.079392e+10	4.061546e+08	1.190942e+07	1.778679e+12	5.328720e+11	1.860261e+11	5.018867e+11
6	Oceania	0.159232	3.810024	1.820014e+12	3.656481e+10	4.323473e+07	8.560171e+06	4.208563e+11	2.561926e+11	1.646632e+11	3.087079e+10
8	United States	0.111727	2.343558	2.138098e+13	6.769409e+11	3.283300e+08	9.831510e+06	2.538450e+12	2.860892e+12	3.565593e+12	-3.767470e+11

[19]:
```
#test_df = merged_df_2.loc[list(range(6, 9))]
test_df = merged_df_2.loc[[2, 4, 7]]
test_df
```

[19]:

	Entity	Tan	Bias	GDP	R&D	Population	Land	Export	Education Expenditure	Health Expenditure	Net Trade
2	China	0.426067	4.797952	1.427997e+13	3.205325e+11	1.407745e+09	9.600013e+06	2.655609e+12	1.603640e+12	7.640180e+11	3.588364e+11
4	India	0.441104	5.096093	2.835606e+12	2.119359e+10	1.383112e+09	3.287260e+06	5.386352e+11	4.743586e+11	8.346679e+10	-4.233284e+09
7	Sub-Saharan Africa	0.143035	3.021784	2.130520e+12	7.834763e+09	1.185010e+09	2.347501e+07	5.760199e+11	2.947490e+11	1.016391e+11	4.081170e+10

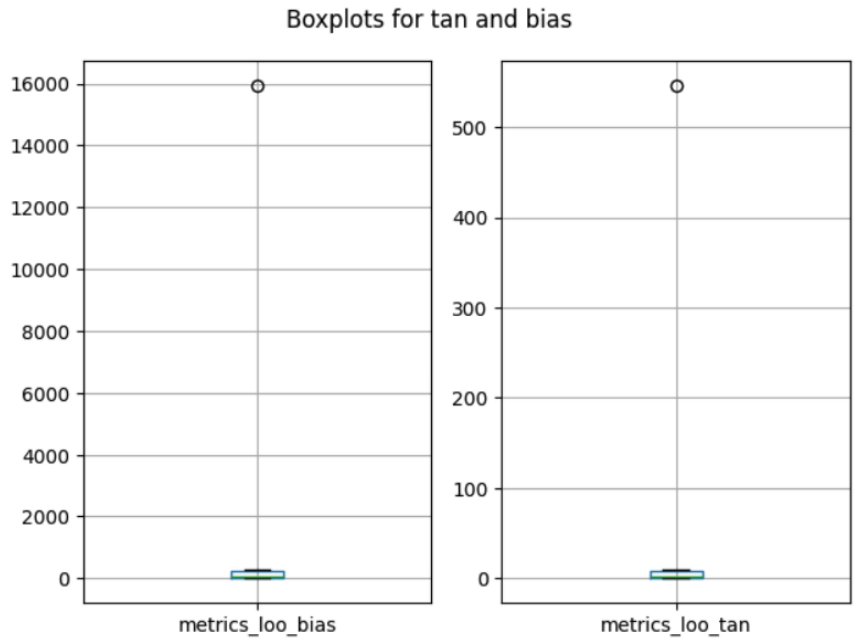
Leave-one-out оценка (Пайплайн)



Leave-one-out оценка (2)

Теперь же был применён один из семейства алгоритмов кросс-валидации, а именно Leave-one-out, заключающийся в том, что каждый элемент модели по очереди выбирался в качестве тестового, а обучение проводилось на всей остальной выборке.

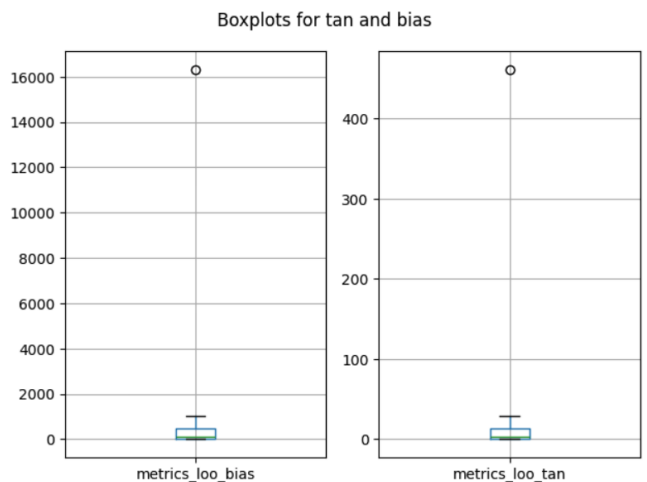
По итогам алгоритма кросс-валидации были построены боксплоты («ящики с усами») для тангенса и свободного члена линейной регрессии по регионам:



Leave-one-out оценка (3)

Видно, что США «портят» график, делая его не особенно наглядным. Предполагаю, что в какой-то мере это можно объяснить тем, что обычно чем более развитая страна (т.е. чем больше значения фичей), тем больше она заботится об экологии. В случае США это, к сожалению, не совсем так. Попробуем переобучить модель на данных без Америки и по этим значениям перестроить боксплоты:

```
[42]: merged_df_no_US = merged_df_2.copy()
      merged_df_no_US = merged_df_no_US.drop(merged_df_no_US[merged_df_no_US["Entity"] == "United States"].index)
      df_loo_res_no_US = leave_one_out(merged_df_no_US)

[41]: build_boxplot(df_loo_res_no_US)
```



Leave-one-out оценка (4)

Можно видеть, что это не помогло - в отстутствие США модель ведёт себя не лучшим образом, из общей статистики начинают выбиваться другие регионы.
Исходный код программы можно посмотреть здесь: https://github.com/paxwritescode/machine_learning

Что ещё можно улучшить?

- Добавить интеграционные тесты
- Добавить данные
- Что-то ещё предсказать
- Выбирать не максимальные значения в каждом регионе, а, например, средние, и посмотреть на поведение модели в этом случае
- Использовать регуляризацию в регрессии
- *Использовать SVM для регрессии вместо линейной регрессии