به نام خدا

پروژه- یادگیری ماشین کاربردی

استاد مربوطه: دکتر ناظرفرد

نام: زهرا اخلاقی

شماره دانشجویی: ۴۰۱۱۳۱۰۶۴

ایمیل: zahra.akhlaghi@aut.ac.ir

تابستان ۱۴۰۲

فهرست مطالب

Problem 1: Price Predictor

1- Create Dataset

برای ساختن دیتاست در این سوال از کتابخانه selenium استفاده شده است، این کتابخانه میتواند امکان تعامل با مرورگر را فراهم کند و اجازه میدهد به صورت تعاملی با صفحات وب کار کرده و داده هایی که مورد نیاز پروژه هست را از صفحات وب جمع آوری کرده و دیتاست بسازیم.

```
: from urllib.parse import urljoin

for i in range(1500):

    # scroll one screen height each time
    driver.execute_script(
        "window.scrollTo(0, {screen_height}*{i});".format(screen_height=s_height, i=i))
    time.sleep(1)


    for each_div in driver.find_elements(
        By.CSS_SELECTOR, '.post-card-item-af972.kt-col-6-bee95.kt-col-xxl-4-e9d46'):
        if each_div == None:
            continue
        url = ''

        # find a tag
        a_tag = each_div.find_element(By.TAG_NAME,'a' )

        if a_tag != None :
            url = urljoin('https://divar.ir', a_tag.get_attribute('href'))
            # find the rent urls and save in the text file
            with open(file, 'a+', newline='', encoding='utf-8') as write_file:
                write_file.writelines(url + '\n')
```

با استفاده از کد بالا، ۱۵۰۰ مرتبه صفحه حاصل از جستجو پیمایش میشود و لینک خودرو ها در فایل Url.txt ذخیره میشود.

```
]: data = []
for link in links:
    try:
        driver.get(link)
        details = {}
        description_elements = driver.find_elements(
            By.CSS_SELECTOR, '.kt-group-row-item--info-row')
        if description_elements != None:
            for element in description_elements:
                title = element.find_element(
                    By.CLASS_NAME, 'kt-group-row-item__title').text
                value = element.find_element(
                    By.CLASS_NAME, 'kt-group-row-item__value').text
                details[title] = value
            second_set_elements = driver.find_elements(
                By.CSS_SELECTOR, '.kt-unexpandable-row')
            for element in second_set_elements:
                title = element.find_element(
                    By.CLASS_NAME, 'kt-unexpandable-row__title').text
                value = element.find_element(
                    By.CLASS_NAME, 'kt-unexpandable-row__value-box').text
                details[title] = value
            data.append(details)
    except Exception:
        continue
```

با استفاده از کد بالا، اطلاعات خودرو هایی که لینک آنها در فایل ذخیره شده بود استخراج شده و در لیست data ذخیره میشود.

دیتاست به دست آمده حاصل از جستجوی بالا به صورت زیر می باشد. این دیتاست شامل ۲۰ ستون (مدل، کارکرد، رنگ، برند و تیپ، نوع سوخت، وضعیت شاسی، وضعیت موتور، وضعیت بدنه، مهلت بیمه، گیربکس، قیمت پایه، شاسی جلو، شاسی عقب، نمایشگاه، نوع آگهی، معاوضه، پیش پرداخت، اقساط و فروشنده) و ۱۶۹۱ ردیف میباشد.

	کارکرد	مدل (سال تولید)	رنگ	برند و تیپ	نوع سوخت	وضعیت موتور	وضعیت شاسی‌ها	وضعیت بدنه	مهلت بیمهٔ شخص ثالث	گیربکس	قیمت پایه	شاسی جلو	شاسی عقب	نمایشگاه	نوع آگهی	مایل به معاوضه	حداقل مبلغ پیش‌پرداخت	مبلغ هر قسط	تعداد اقساط	فروشنده
0	۳۰۰٬۰۰۰	۱۳۸۳	نقره‌ای	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	رنگ‌شدگی	۹ ماه	دنده‌ای	۱۸۶٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	۴۶٬۰۰۰	۱۴۰۰	سفید	پژو 206 تیپ ۲	بنزینی	سالم	NaN	صافکاری بی‌رنگ	۱ ماه	دنده‌ای	۳۸۵٬۰۰۰٬۰۰۰ تومان	ضربه‌خورده	سالم و پلمب	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	۱۸٬۶۰۰	۱۳۹۹	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	سالم و بی‌خط‌و خش	۱۱ ماه	دنده‌ای	۴۳۲٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	۲۵۱٬۰۰۰	۱۳۸۸	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	خط و خش جزیی	۹ ماه	دنده‌ای	۲۶۸٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	۲۰۰٬۰۰۰	۱۳۹۰	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	رنگ‌شدگی، در ۲ ناحیه	۱۰ ماه	دنده‌ای	۲۹۰٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1686	۴۰٬۰۰۰	۱۴۰۰	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	خط و خش جزیی	۶ ماه	دنده‌ای	۴۲۵٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1687	۰	۱۴۰۱	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	سالم و بی‌خط‌و خش	۱۲ ماه	دنده‌ای	۴۹۰٬۰۰۰٬۰۰۰ تومان	NaN	NaN	مهرداد خودرو	فروشی	NaN	NaN	NaN	NaN	NaN
1688	۱۲۰٬۰۰۰	۱۳۹۱	سفید	پژو 206 تیپ ۲	بنزینی	سالم	سالم و پلمب	رنگ‌شدگی	۸ ماه	دنده‌ای	۲۸۰٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	فروشی	NaN	NaN	NaN	NaN	NaN
1689	۹۷٬۰۰۰	۱۳۹۶	سفید	پژو 206 تیپ ۲	بنزینی	سالم	NaN	رنگ‌شدگی، در ۲ ناحیه	NaN	NaN	۳۳۰٬۰۰۰٬۰۰۰ تومان	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1690	۰	۱۴۰۱	سفید	پژو 206 تیپ ۲	بنزینی	NaN	NaN	سالم و بی‌خط‌و خش	NaN	NaN	توافقی	NaN	NaN	شرکت اتحاد خودرو خاورمیانه	NaN	NaN	NaN	NaN	NaN	NaN

rows × 20 columns 1691

2- Preprocessing

همه داده های جدول بالا ممکن است برای تخمین قیمت مناسب نباشد و مقادیر بسیاری از ستون ها null است و نیاز به پیش پردازش دارد. برای پیش پردازش کارهای زیر انجام شده است:

- ستون های برند و نوع سوخت به دلیل اینکه همه ی آنها دارای یک مقدار میباشند، حذف شده.
- ستون های حداقل مبلغ پیش‌پرداخت، مبلغ هر قسط، تعداد اقساط حذف شده و ستون قسطی اضافه شده که براساس این سه ستون نشان میدهد این ماشین فروش قسطی دارد یا خیر.
- ستون نوع آگهی به دلیل اینکه تنها دارای مقدار فروش میباشد و در خروجی تاثیری ندارد حذف شده است.
- ردیف های دارای مقادیر null در ستون های وضعیت بدنه و شاسی و موتور حذف شده اند.
- مقدار نال در ستون گیربکس با "دنده ای" جایگزین شده است.
- مقادیر نال در ستون مهلت بیمه با صفر جایگذاری شده و در نهایت به دلیل تاثیر کم آن بر روی قیمت حذف شد.
- ستون مایل به معاوضه به دلیل عدم تاثیر آن بر قیمت حذف شده

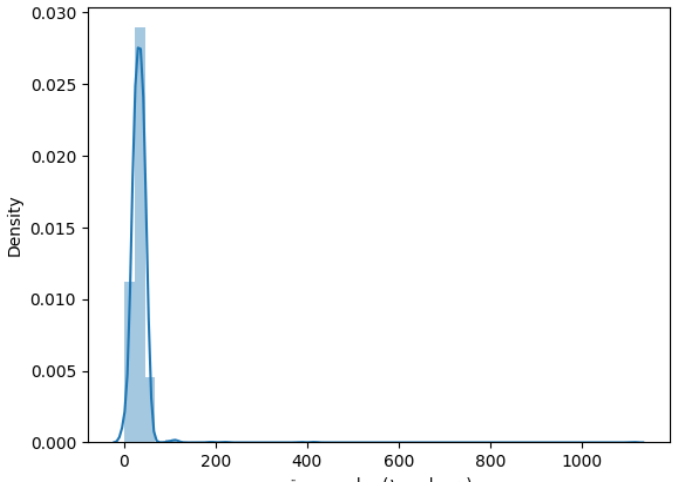
- ستون نمایشگاه دارای اسم نمایشگاه است که با دو مقدار شخصی یا نمایشگاهی مقدار دهی شده
- ستون های شاسی عقب، شاسی جلو، مبلغ هر قسط، حداقل مبلغ پیش پرداخت، تعداد اقساط، فروشنده، مایل به معاوضه و مهلت بیمه حذف شده
- ردیف هایی که دارای قیمت توافقی هستند حذف شده اند
- ردیف هایی که کارکرد آنها کوچکتر از صفر است حذف شده اند.
- قیمت پایه بر مبنای ۱۰ میلیون قرار گرفته است

دیتاست نهایی به صورت زیر میباشد:

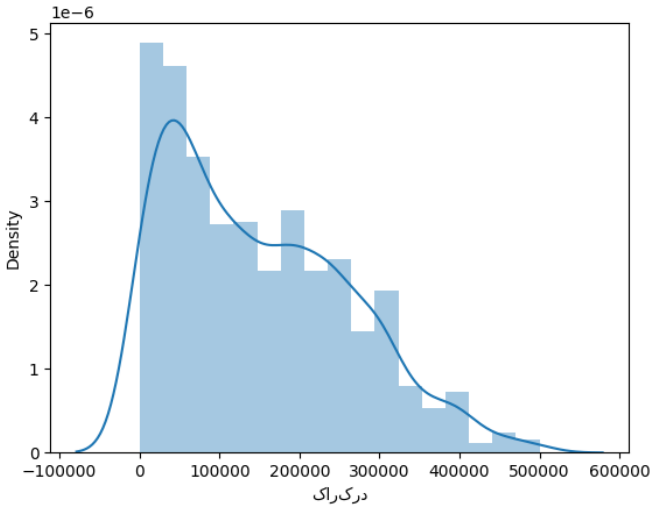
	کارکرد	مدل (سال تولید)	رنگ	وضعیت موتور	وضعیت شاسی‌ها	وضعیت بدنه	گیربکس	نمایشگاه	قسط	قیمت پایه(۱۰ میلیون)
0	300000	1383	نقره‌ای	سالم	سالم و پلمپ	رنگ‌شدگی	دنده‌ای	شخصی	True	18.6
2	18600	1399	سفید	سالم	سالم و پلمپ	سالم و بی‌خط و خش	دنده‌ای	شخصی	True	43.2
3	251000	1388	سفید	سالم	سالم و پلمپ	خط و خش جزیی	دنده‌ای	شخصی	True	26.8
4	200000	1390	سفید	سالم	سالم و پلمپ	رنگ‌شدگی، در ۲ ناحیه	دنده‌ای	شخصی	True	29.0
5	31000	1400	سفید	سالم	سالم و پلمپ	سالم و بی‌خط و خش	دنده‌ای	شخصی	True	44.5
...	...	...	...	...	...	...	...	...	...	...
1684	29000	1398	سفید	سالم	سالم و پلمپ	سالم و بی‌خط و خش	دنده‌ای	شخصی	True	40.5
1685	300000	1387	خاکستری	سالم	سالم و پلمپ	رنگ‌شدگی	دنده‌ای	شخصی	True	24.5
1686	40000	1400	سفید	سالم	سالم و پلمپ	خط و خش جزیی	دنده‌ای	شخصی	True	42.5
1687	0	1401	سفید	سالم	سالم و پلمپ	سالم و بی‌خط و خش	دنده‌ای	نمایشگاهی	True	49.0
1688	120000	1391	سفید	سالم	سالم و پلمپ	رنگ‌شدگی	دنده‌ای	شخصی	True	28.0

3- Data Visualization

شکل زیر نمودار توزیع قیمت را نشان میدهد:

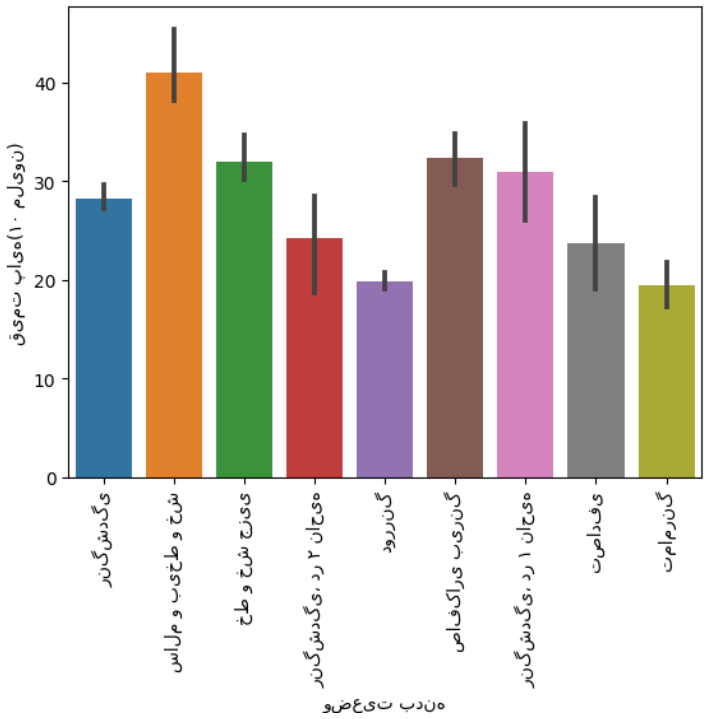


شکل زیر نمودار توزیع کارکرد را نشان میدهد

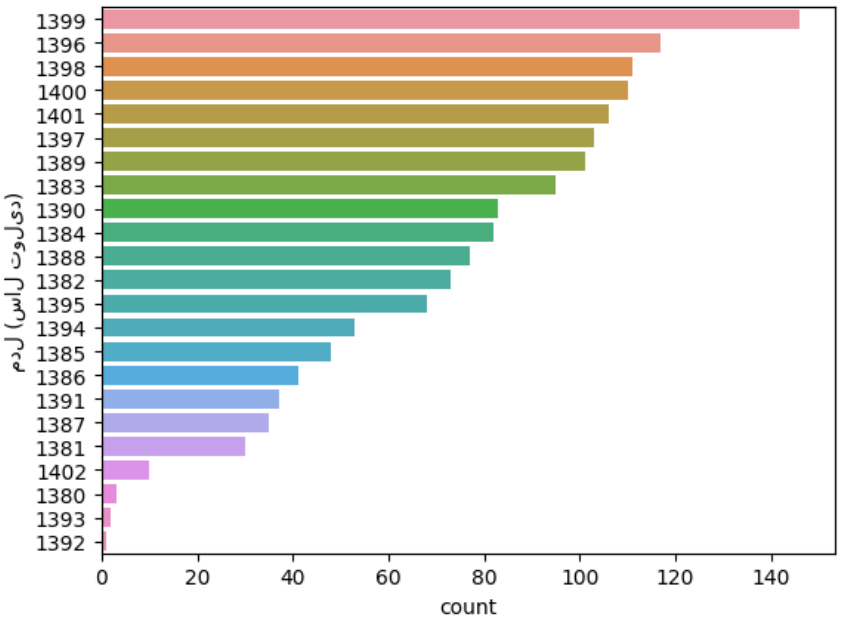


نمودار زیر نشان دهنده تاثیر وضعیت بدنه بر قیمت خودرو میباشد و جدول میانگین قیمت برای هر رنگ را نشان میدهد

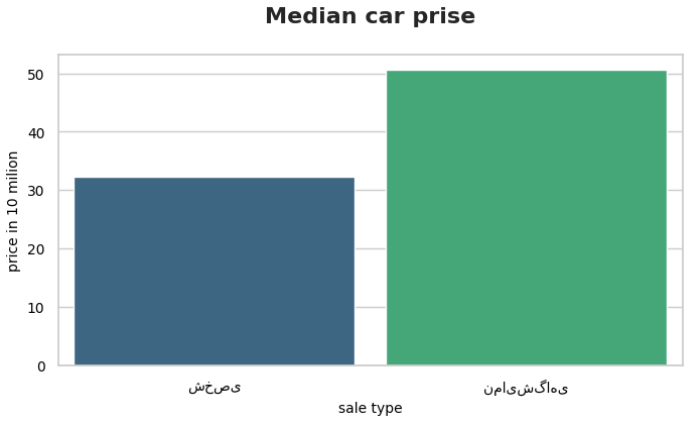
رنگ	قیمت پایه(۱۰ ملیون)
گیلاسی	42.600000
آبی	40.183333
سفید	37.976399
سفید صدفی	34.193939
قرمز	33.940000
سرمه‌ای	33.500000
بادمجانی	32.166667
خاکستری	31.499943
مشکی	28.592465
نوکمدادی	26.375956



شکل زیر نشان دهنده تعداد خودرو ها در هر سال می باشد:



شکل زیر میانگین قیمت خودرو های نمایشگاهی و شخصی را نشان میدهد:



4- Model Training

برای آموزش مدل همه ستون های categorical به مقدار عددی تبدیل شدند و پیش بینی قیمت بر اساس کارکرد، وضعیت شاسی، وضعیت بدنه، وضعیت موتور، مدل، رنگ، گیربکس، نمایشگاهی و قسطی میباشد. دیتاست نهایی به صورت زیر است:

	کارکرد	مدل (سال تولید)	رنگ	وضعیت موتور	وضعیت شاسی‌ها	وضعیت بدنه	گیربکس	نمایشگاه	قسط	قیمت پایه(۱۰ میلیون)
0	300000	1383	2	0	0	2	0	0	0	18.6
2	18600	1399	0	0	0	0	0	0	0	43.2
3	251000	1388	0	0	0	1	0	0	0	26.8
4	200000	1390	0	0	0	6	0	0	0	29.0
5	31000	1400	0	0	0	0	0	0	0	44.5
...	...	...	...	...	...	...	...	...	...	...
1684	29000	1398	0	0	0	0	0	0	0	40.5
1685	300000	1387	1	0	0	2	0	0	0	24.5
1686	40000	1400	0	0	0	1	0	0	0	42.5
1687	0	1401	0	0	0	0	0	1	0	49.0
1688	120000	1391	0	0	0	2	0	0	0	28.0

برای آموزش مدل در ابتدا مقادیر بهینه برای مدل های Ridge, Lasso, Elastic Net با تقسیم کردن داده های آموزش به آموزش و ارزیابی به دست آورده شده:

```
{'Ridge': {'alpha': 100}, 'Lasso': {'alpha': 0.1}, 'Elastic Net': {'alpha': 0.1, 'l1_ratio': 0.1}}
{'Ridge': 0.135339761871767, 'Lasso': 0.13491736806015364, 'Elastic Net': 0.13530307947549958}
```

5- Model Evaluation

در نهایت ۷ مدل آموزش دیده شده (Linear Regression, Ridge, Lasso, Bayesian, Elastic Net, Decision Tree Regressor, RandomForest Regressor) که نتایج آنها به صورت زیر است:

```
Explained Variance Score of OLS model is 0.04433804941354569
--------------------------------------------------------------------------------
Explained Variance Score of Ridge model is 0.04145760699079504
--------------------------------------------------------------------------------
Explained Variance Score of Lasso model is 0.04059413821273772
--------------------------------------------------------------------------------
Explained Variance Score of Bayesian model is 0.04047708541413131
--------------------------------------------------------------------------------
Explained Variance Score of ElasticNet is 0.04126131077235573
--------------------------------------------------------------------------------
Explained Variance Score of DecisionTreeRegressor model is -0.10061108037169464
--------------------------------------------------------------------------------
Explained Variance Score of RandomForestRegressor is 0.017616892715087595
```

```
R-Squared of OLS model is 0.04262536683180529
--------------------------------------------------------------------------
R-Squared of Ridge model is 0.03976230771765399
--------------------------------------------------------------------------
R-Squared of Lasso model is 0.03888920005673091
--------------------------------------------------------------------------
R-Squared of Bayesian model is 0.03879640444061416
--------------------------------------------------------------------------
R-Squared of ElasticNet is 0.03956685998647613
--------------------------------------------------------------------------
R-Squared of DecisionTreeRegressor model is -0.10177712993258403
--------------------------------------------------------------------------
R-Squared of RandomForestRegressor model is 0.01572342009294414
--------------------------------------------------------------------------
```

```
MSE of OLS model is 3718.0139058641757
--------------------------------------------------------------------------
MSE of Ridge model is 3729.132744018994
--------------------------------------------------------------------------
MSE of Lasso model is 3732.523502779633
--------------------------------------------------------------------------
MSE of Bayesian model is 3732.883879354458
--------------------------------------------------------------------------
MSE of ElasticNet is 3729.891775392098
--------------------------------------------------------------------------
MSE of DecisionTreeRegressor model is 4278.808470928846
--------------------------------------------------------------------------
MSE of RandomForestRegressor model is 3822.4890074645864
--------------------------------------------------------------------------
```

با مقایسه نتایج بالا Linear Regression دارای خطا کمتری است و عملکرد بهتری دارد و به عنوان مدل نهایی انتخاب و ذخیره میشود.

پژو 206 تیپ ۲، مدل ۱۳۸۹

یک ربع پیش در تهران، یافت‌آباد

زنگ خطرهای قبل از معامله

اطلاعات تماس | چت

کارکرد	مدل (سال تولید)	رنگ
۲۴۵,۰۰۰	۱۳۸۹	سفید

برند و تیپ	پژو 206 تیپ ۲
نوع سوخت	بنزینی
وضعیت موتور	سالم
وضعیت شاسی‌ها	سالم و پلمپ
وضعیت بدنه	رنگ‌شدگی
مهلت بیمهٔ شخص ثالث	۸ ماه
گیربکس	دنده‌ای
قیمت پایه	۲۵۵,۰۰۰,۰۰۰ تومان

```
245000 :کارکرد
1389        :(مدل (سال تولید
0:سفید1:خاکستری 2:نقره‌ای 3:مشکی 4:نوک‌مدادی
5:بژ 6:سفید صدفی 7:نقرآبی 8:آلبالویی
9:قرمز 10:یشمی آبی 11:دلفینی 12:
13:بادمجانی 14:گیلاسی 15:سرمه‌ای 16:طلایی
17:نارنجی 18:قهوه‌ای 19:عدسی 20:سبز 21:ذغالی
0        :شماره رنگ ماشین
0       :شماره وضعیت موتور 2:نیاز به تعمیر 1:تعویض شده 0:سالم
0:شماره وضعیت شاسی 2:رنگشده 1:ضربه‌خورده :سالم و پلمپ
4:صافکاری بورنگ 3:دوررنگ 2:رنگشدگی 1:خط و خش جزیی 0:سالم و بی‌خط و خش
8:تصادفی 7:رنگشدگی، در ۱ ناحیه 6:رنگشدگی، در ۲ ناحیه 5:تمام‌رنگ
2       :شماره وضعیت بدنه
1:اتوماتیک 0:دنده‌ای
0       :شماره نوع گیربکس
0:بله 1:خیر  :ماشین نمایشگاهی است؟
0
0:بله 1:خیر  :قادر به فروش به صورت اقساط هستید؟
0
****************************************
255000000:قیمت پیشنهادی شما
270000000  :قیمت تخمین زده شده
15000000 :اختلاف قیمت
قیمت پایین
```

پژو 206 تیپ ۲، مدل ۱۳۹۸

نیم ساعت پیش در تهران، اقدسیه

زنگ خطرهای قبل از معامله

اطلاعات تماس | چت

کارکرد	مدل (سال تولید)	رنگ
۵۳,۰۰۰	۱۳۹۸	سفید

برند و تیپ	پژو 206 تیپ ۲
نوع سوخت	بنزینی
وضعیت موتور	سالم
وضعیت شاسی‌ها	سالم و پلمپ
وضعیت بدنه	خط و خش جزیی
مهلت بیمهٔ شخص ثالث	۸ ماه
گیربکس	دنده‌ای
قیمت پایه	۴۳۰,۰۰۰,۰۰۰ تومان

این خودرو

بالا | منصفانه | پایین

```
53000 :کارکرد
1398        :(مدل (سال تولید
0:سفید1:خاکستری 2:نقره‌ای 3:مشکی 4:نوک‌مدادی
5:بژ 6:سفید صدفی 7:نقرآبی 8:آلبالویی
9:قرمز 10:یشمی آبی 11:دلفینی 12:
13:بادمجانی 14:گیلاسی 15:سرمه‌ای 16:طلایی
17:نارنجی 18:قهوه‌ای 19:عدسی 20:سبز 21:ذغالی
0        :شماره رنگ ماشین
0       :شماره وضعیت موتور 2:نیاز به تعمیر 1:تعویض شده 0:سالم
0:شماره وضعیت شاسی 2:رنگشده 1:ضربه‌خورده :لم و پلمپ
4:صافکاری بورنگ 3:دوررنگ 2:رنگشدگی 1:خط و خش جزیی
8:تصادفی 7:رنگشدگی، در ۱ ناحیه 6:رنگشدگی، در ۲ ناحیه 5:
1       :شماره وضعیت بدنه
1:اتوماتیک 0:دنده‌ای
0       :شماره نوع گیربکس
0:بله 1:خیر  :ماشین نمایشگاهی است؟
0
0:بله 1:خیر  :قادر به فروش به صورت اقساط هستید؟
0
****************************************
430000000:قیمت پیشنهادی شما
400000000  :قیمت تخمین زده شده
30000000 :اختلاف قیمت
قیمت بالا
```

Problem 2: Hamshahri Newspaper

Step 1 - Introduction to the Dataset

همشهری یکی از پرمخاطب ترین روزنامه های ایران است. دیتاست همشهری یک مجموعه آزمایشی فارسی است که شامل 345 مگابایت متن خبری این روزنامه از سال 1375 تا 1381 میباشد (حجم پیکره با برچسب ها 564 مگابایت است). این مجموعه شامل بیش از 160000 مقاله خبری در مورد موضوعات مختلف و شامل نزدیک به 417000 کلمه مختلف است.

Step 2 - Data Loading

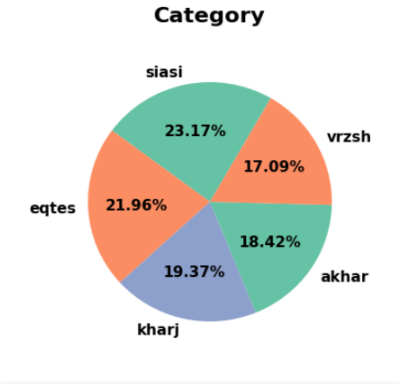
نتیجه تبدیل فایل متنی به فرمت جدول، به صورت زیر است. این دیتاست، از ۴ ستون تشکیل شده است که شامل متن خبری، دسته بندی آن، تاریخ خبر و آیدی میباشد.

	DID	Date	Cat	Text
0	1S1	75\04\02	adabh	...جاودانگي در زندگي گروهي از طريق هنر نگاهي به ن
1	2S1	75\04\02	adabh	...رويدادهاي هنري جهان نمايشگاه هنر در خدمت ديكتا
2	3S1	75\04\02	adabh	...برديوار نگارخانه ها گالري گلستان: نمايشگاه طرح
3	4S1	75\04\02	ejtem	...بازي را جدي بگيريم مطالعه اي مقدماتي پيرامون ن
4	5S1	75\04\02	elmfa	...تخته سياه و غباري كه سترده نمي شود... اشاره; ب
...	...	...	...	...
165220	60055S2	81\11\20	vrzsh	...نماينده فدراسيون جهاني واليبال از ايران هر نظر
165221	60055S3	81\11\20	vrzsh	...شكست نامداران تكواندودر پيكارهاي برتر ليگ گروه
165222	60055S4	81\11\20	vrzsh	...ورزشگاه بزرگ دانشگاه آزاد در تهران ساخته مي شو
165223	60055S5	81\11\20	vrzsh	...رئيس فدراسيون پزشكي انتخاب شد گروه ورزشي: مجمع
165224	60055S6	81\11\20	vrzsh	...نتايج هفته يازدهم وزنه برداري باشگاهها گروه ور

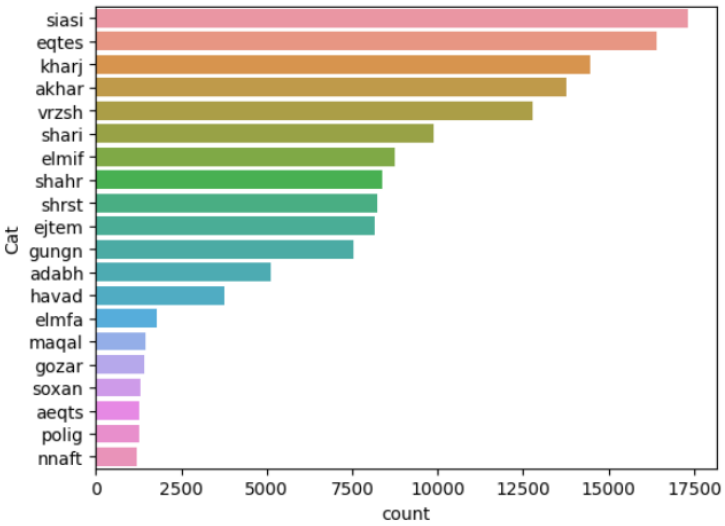
165225 rows × 4 columns

Step 3 - Data Visualization

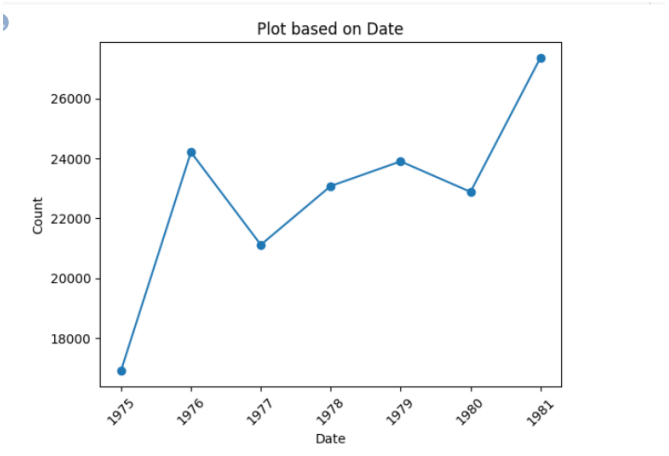
شکل زیر نمودار دایره ای می‌باشد، که تعداد داده ها در ۵ دسته بندی پرتکرار را نشان می دهد و برای مقایسه دسته بندی ها کاربرد دارد، این دسته‌ها تقریبا درصد نزدیک به یکدیگری دارند و siasi پرتکرارترین دسته بندی است.



شکل زیر تعداد متن خبری به ازای هر دسته بندی را نشان میدهد (با توجه به اینکه ۱۰۵ دسته بندی وجود دارد امکان نشان دادن همه آنها وجود ندارد) بیشترین تعداد خبر برای دسته بندی سیاسی با تعدادی حدود ۱۷ هزار میباشد.



شکل زیر تعداد متن خبری در هر سال را نشان میدهد، سال ۱۳۸۱ دارای بیشترین تعداد خبر و سال۱۳۵ دارای کمترین تعداد خبر میباشد، و از سال ۱۳۷۶ تا ۱۳۷۷ تعداد اخبار کاهش یافته است.



Step 4 – Preprocessing

پیش پردازش روی این داده ها شامل موارد زیر میباشد:

- چک کردن تعداد مقادیر null در هر ستون، در این جدول مقدار null برای حذف کردن وجود نداشت.
- با استفاده از تابع remove_tag تگ های html , url موجود در متن حذف شده.
- فایل PersianStopWords شامل حروف اضافی، کاراکتر ها، حروف اشاره و ... است. وجود این حروف در متن خبر ها بررسی شده و در صورت وجود حذف شده.
- \n و \r از متن حذف شده
- حروف عربی موجود در متن با معادل فارسی جایگذاری شده
- اسپیس های اضافی موجود در متن حذف شده
- با استفاده از کتابخانه هضم lemmatize، stemming روی متن اعمال شد
- تعداد کلمات هر متن بررسی شده و متن های خبری با تعداد کلمه کمتر از ۱۰۰ حذف شده اند
- دسته بندی ها و تعداد آنها بررسی شده و دسته بندی هایی که تعداد اخبار آنها کمتر از ۵ درصد کل داده ها است، حذف شده و در نهایت ۱۰ دسته بندی استخراج شده.

دیتاست نهایی:

	DID	Date	Cat	Text	text_len_by_words
0	4S1	75\04\02	ejtem	... بازی جدی بگیریم مطالعه ای مقدماتی پیرامون نقش	1153.0
1	11S1	75\04\02	eqtes	...رشد اقتصادی کشورهای صنعتی سال آینده سازمان همک	87.0
2	12S1	75\04\02	eqtes	...تن مدیران ارشد کشور تهران آموزش دیدند سرویس اق	150.0
3	13S1	75\04\02	eqtes	...کمیته راهنمایی سرمایه گذاران ایرانی تشکیل می ت	144.0
4	14S1	75\04\02	eqtes	...هفته صرفه جویی مصرف آب آغاز سرویس اقتصادی هفته	87.0
...	...	...	...	...	...
116631	59437S6	81\10\14	eqtes	...منطقه آزاد کیش تالار فرعی معاملات ارزی کشور تب	130.0
116632	59437S7	81\10\14	eqtes	...روغن موتور سطح کیفیت جهانی ایران تولید گروه اق	103.0
116633	59437S8	81\10\14	eqtes	...یادداشت احزاب سیاسی بودجه محمدصادق جنان صفت بر	135.0
116634	59437S9	81\10\14	eqtes	...خبرها نکته شهامت یک مدیر خبر سال پیش وسیعی شرق	109.0
116635	59437S10	81\10\14	eqtes	...یادآوری دقت معاون اقتصادی سازمان مدیریت برنامه	114.0

116636 rows × 5 columns

Step 5 - Feature Engineering

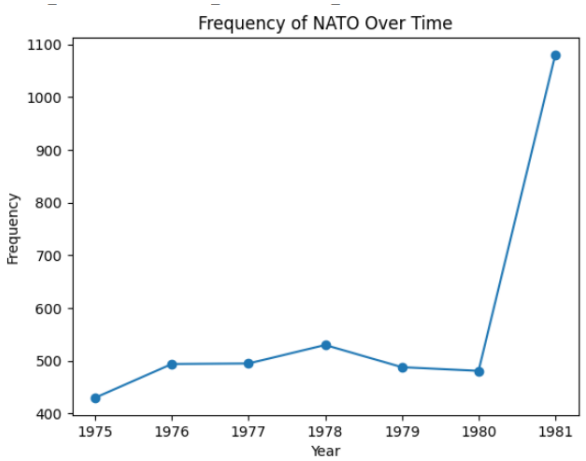
TF-IDF، یک آمار عددی، جهت نشان دادن اهمیت یک کلمه برای یک سندی که در مجموعه ای واقع شده است. عموما مقدار tf-idf, متناسب با تعداد تکرار یک کلمه در یک سند (document) افزایش می یابد و تعدادی از سندها که در بدنه خود دارای این کلمه هستند را متعادل می کند. هدف استفاده از این روش، تنظیم کردن و یکدست کردن کلماتی است که در یک متن، مدام تکرار می شوند.

از تابع TfidfVectorizer برای به دست آوردن tfidf استفاده میشود و با توجه به حجم زیاد داده ها در صورت استفاده از همه اطلاعات، مشکل crash حافظه وجود دارد، بنابراین ۵۰۰۰ بهترین ویژگی ها استخراج شده است.

```
# TF-IDF Matrix for top 5000 words
vectorizer = TfidfVectorizer(ngram_range=(1,1), max_features=5000)

# Fit and transform the 'Text' column using TF-IDF
tfidf_matrix = vectorizer.fit_transform(df['Text'])
```

شکل زیر تعداد تکرار کلمه ناتو را در سال های مختلف نشان میدهد



در سال ۱۳۸۱ که معادل با ۲۰۰۲ میلادی است، شامل بیشترین تعداد تکرار این کلمه است زیرا درنشست پراگ که در سال ۲۰۰۲ میلادی دایر گردید، رهبران ناتو اصلاحات گسترده را معرفی نمودند که به شکل دراماتیک دارایی های نظامی ناتو را دوباره شکل داد. آنها ساحات مشخصی را برای بهتر ساختن معرفی نمودند، نیروی پاسخگوی ناتو را ایجاد نمودند و ساختار قومنده نظامی را سریعتر ساختند، که این تغییرات باعث شد اخبار زیادی درباره آن وجود داشته باشد.

Step 6 - Dimensionality Reduction

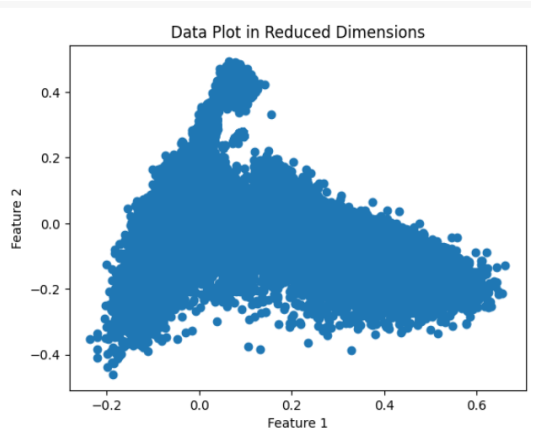
برای کاهش بعد به صورت زیر از PCA استفاده شده است.

```
pca_reducer_2 = PCA(n_components = 2)
reduced_features = pca_reducer_2.fit_transform(tfidf_array)

sum(pca_reducer_2.explained_variance_ratio_)
```

```
0.02542951748377438
```

نتیجه کاهش بعد به صورت زیر میباشد:



Step 7 - Clustering In this section

خوشه بندی:

```
from sklearn.cluster import KMeans

num_clusters = 10
kmeans = KMeans(n_clusters=num_clusters, random_state=42)

# Fit the K-means model to the reduced data
kmeans.fit(reduced_df)

# Get the cluster labels assigned by K-means
cluster_labels = kmeans.labels_
```

با توجه به اینکه داده ها دارای ۱۰ دسته بندی میباشد، تعداد خوشه انتخاب شد برابر با ۱۰ میباشد. نتیجه (شکل زیر) نشان می دهد کاهش بعد نتوانسته به درستی داده ها را مدل کند زیرا این تعداد دسته بندی برای این داده ها مناسب نیست و به درستی نتوانسته داده ها را از یکدیگر جدا کند.

Step 8 - Storage

کاهش بعد با تعداد ابعاد ۵:

```
pca_reducer_5 = PCA(n_components = 5)
reduced_features = pca_reducer_5.fit_transform(tfidf_array)

sum(pca_reducer_5.explained_variance_ratio_)
```

```
0.05106137341931134
```

خوشه بندی و ذخیره نتایج:

```
from sklearn.cluster import KMeans

num_clusters = 10
kmeans = KMeans(n_clusters=num_clusters, random_state=42)

# Fit the K-means model to the reduced data
kmeans.fit(reduced_df)

# Get the cluster labels assigned by K-means
cluster_labels = kmeans.labels_
```

نتیجه خوشه بندی با کاهش بعد ۵

```
result_df.sample(7)
```

	DID	Date	Cat	Text	Cluster Labels
50099	45129S7	78\04\14	elmif	شناسنامه اشیای تاریخی فرهنگی کشور صادر می گروه...	4
73473	49187S3	79\06\02	akhar	مناسبت سال امیرالمومنین مسابقه کتابخوانی استان...	0
34763	42503S5	77\06\24	vrzsh	تیم تکواندو نوجوانان ایران جهان هفتم گروه ورزش...	2
89145	51588S9	80\03\13	vrzsh	خبرهایی فوتبال جهان دومین پیروزی تیم ملی ژاپن ...	2
110641	57123S4	81\06\03	kharj	وزیر خارجه قطر بغداد می رود حمد بن جاسم آل ثان...	8
70647	48719S6	79\04\19	kharj	مذاکرات صلح شاخ آفریقا آمریکا پایان یافت گوی س...	8
70574	48706S2	79\04\18	kharj	عفو بین الملل عربستان حقوق مهاجران غیرقانونی م...	8

متن هایی با cat یکسان تقریبا در یک دسته بندی قرار گرفته اند.

نتایج خوشه بندی در فایل زیر ذخیره شده است:

https://drive.google.com/file/d/1-6ILwwCLmTS3mn4gsMkn_OEQt0RY2oSH/view?usp=drive_link

Step 9 - Classification (Model Building)

ساخت مدل ها:

```
from sklearn.neighbors import KNeighborsClassifier

# Create a KNN classifier with k=7
knn = KNeighborsClassifier(n_neighbors=7)
```

```
from sklearn.linear_model import LogisticRegression

# Create a Logistic Regression classifier
logreg = LogisticRegression()
```

```
from sklearn.naive_bayes import GaussianNB

# Create a Naïve Bayes classifier
nb = GaussianNB()
```

```
from sklearn.ensemble import RandomForestClassifier

# Create a Random Forest classifier with 100 trees
rf = RandomForestClassifier(n_estimators=100)
```

```
from sklearn.ensemble import VotingClassifier

# Create the ensemble model with voting
ensemble_model = VotingClassifier(estimators=[
    ('knn', knn),
    ('logreg', logreg),
    ('nb', nb),
    ('rf', rf)
], voting='hard')  # 'hard' voting for majority vote, 'soft' voting for weighted average
```

ساخت مدل ensemble با استفاده از مدل های بالا:

```
from sklearn.ensemble import VotingClassifier

# Create the ensemble model with voting
ensemble_model = VotingClassifier(estimators=[
    ('knn', knn),
    ('logreg', logreg),
    ('nb', nb),
    ('rf', rf)
], voting='hard')  # 'hard' voting for majority vote, 'soft' voting for weighted average
```

Step 10 - Preprocessing on Data

در پیش پردازش داده ها در این مرحله، کارهای زیر انجام شده:

- از LabelEncoder برای ستون Cat استفاده شده،

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
encoded_labels = label_encoder.fit_transform(df['Cat'])
```

- داده ها به دو دسته تست و آموزش تقسیم شده اند.
- روی داده های تست و آموزش از tf-idf استفاده شده است

```python
from sklearn.feature_extraction.text import TfidfVectorizer


vectorizer = TfidfVectorizer(ngram_range=(1,1), max_features=4000)

X_train = vectorizer.fit_transform(X_train).toarray()
X_test = vectorizer.fit_transform(X_test).toarray()
```

Step 11 - Model Training

در این مرحله آموزش مدل ها انجام شده است.

Step 12 - Model Evaluation

نتایج accuracy روی داده آموزش:

```
KNN Accuracy on Training Data: 0.8073155570797788
Logistic Regression Accuracy on Training Data: 0.8860440691044712
Naïve Bayes Accuracy on Training Data: 0.7203348051614009
Random Forest Accuracy on Training Data: 0.9999571312213315
Ensemble Model Accuracy on Training Data: 0.9076713679427273
```

نتایج accuracy روی داده تست:

```
KNN Accuracy on Testing Data: 0.26727537722908096
Logistic Regression Accuracy on Testing Data: 0.2808641975308642
Naïve Bayes Accuracy on Testing Data: 0.13473079561042525
Random Forest Accuracy on Testing Data: 0.2069187242798354
Ensemble Model Accuracy on Testing Data: 0.23563957475994513
```

نتایج precision روی داده تست:

```
KNN Precision: 0.26727537722908096
Logistic Regression Precision: 0.2808641975308642
Naïve Bayes Precision: 0.13473079561042525
Random Forest Precision: 0.2069187242798354
Ensemble Model Precision: 0.23563957475994513
```

Confusion Matrix روی داده های تست:

```
Logistic Regression Confusion Matrix:
[[1998   69   28   42  177   83   33   13  103   26]
 [ 861  311   33   58   64   58   37   13  163   10]
 [1079   93  287   23   33   47   14   14  143   12]
 [1658  419   13  300  142  159   61  109  321   35]
 [1182  133   12   78 1013  100   26    7  273   22]
 [ 747  116   34   65  101  257  119   59  121   31]
 [ 789  111   24   54   45  248  415   60  146   21]
 [ 645  216   20   96   68  138   80  124  212   28]
 [1901  291   36   84  212  104   69   40  822   15]
 [ 959  105    9   74   86   60   29    6  223 1025]]
```

```
KNN Confusion Matrix:
[[711 348 242 212 187  75 135  86 235 341]
 [149 431 126 171 203  82 157 101 127  61]
 [209 178 647 151 171  49  83  76  96  85]
 [279 585 212 830 370 136 165 143 340 157]
 [191 331 121 409 884 147 201  90 363 109]
 [175 189 161 247 168 142 220 109 153  86]
 [136 236 152 272 167 122 499 105 153  71]
 [231 166 164 265 117  80 149 199 180  76]
 [272 660 230 440 428 148 209 115 931 141]
 [137 367 176 258 176  80 210  39 172 961]]
```

```
Naïve Bayes Confusion Matrix:
[[ 951  351  268   59    3  314  238   74  281   33]
 [ 437  393  101   18    0  212   83   10  347    7]
 [ 531  249  211   10    0  286   94   13  344    7]
 [1198  414  174   29    2  584  205   16  578   17]
 [ 690  395  257   28    2  480  223   19  734   18]
 [ 696   96   45   35    0  522   46    4  194   12]
 [ 441  307  194   62    1  363  298   16  218   13]
 [ 407  261  146   20    0  352  173   38  216   14]
 [1245  520  206   30    1  692  178   16  675   11]
 [1037  420  117    5    2  512   90    6  363   24]]
```

```
Random Forest Confusion Matrix:
[[2325    3    9   40   23   37    6    3   10  116]
 [1180   16   50  109   19   21    5    2   35  171]
 [1439    3  134   24    9   12    4    1   23   96]
 [2576   36   13  213  170   17    6   45   35  106]
 [1819   88    1  251  515   19    3    1   11  138]
 [1302   11   25   62   32   38   12    6   14  148]
 [1396   10   20  102   25   86   84   20   21  149]
 [1238    3   20  107   49   46   10   10   30  114]
 [2369   84   52  261  161   36   16    6  349  240]
 [1278    8    5   54   76   10    0    0    2 1143]]
```

```
Ensemble Model Confusion Matrix:
[[2278   63   23   20   33   52   22    2   55   24]
 [1096  241   26   37   33   37   23    4   99   12]
 [1341   59  196   11   19   29    7    6   58   19]
 [2319  204   19  227   88   76   44   45  163   32]
 [1642  120   29   89  666   62   24    2  188   24]
 [1220   58   20   41   28  141   51   11   51   29]
 [1167   83   43   58   26  127  312   11   55   31]
 [1088  119   29   82   34   80   53   29   88   25]
 [2358  243   32   96  128   67   44    8  577   21]
 [1414  104   17   37   34   34   22    0   84  830]]
```

برای داده های آموزش Random Forest دارای بهترین عملکرد می‌باشد،در داده های تست Logistic Regression دارای بهترین عملکرد میباشد و Naive Bayes در هر دو داده تست و آزمایش بدترین عملکرد را دارد.

What is the reason for its better performance?

رگرسیون لجستیک یک مدل خطی است که از یک مرز تصمیم گیری خطی برای جداسازی کلاس های مختلف استفاده می کند , رگرسیون لجستیک یک الگوریتم نسبتا ساده و قابل تفسیر در مقایسه با مدل های پیچیده تر مانند جنگل تصادفی است. از یک الگوریتم بهینه سازی ساده برای تخمین پارامترها استفاده می کند و هایپر پارامترهای کمتری برای تنظیم دارد. این سادگی می تواند منجر به عملکرد خوب با خطر کمتری برای بیش از حد نصب شود.

رگرسیون لجستیک اهمیت هر ویژگی را با تعیین وزن به آنها تخمین می زند. در مورد ویژگی‌های TF-IDF، رگرسیون لجستیک می‌تواند به طور موثر به کلمات یا اصطلاحاتی که برای دسته‌های هدف متمایزتر هستند، وزن اختصاص دهد. این مکانیسم انتخاب ویژگی به رگرسیون لجستیک کمک می‌کند تا روی آموزنده‌ترین ویژگی‌ها تمرکز کند و به طور بالقوه منجر به عملکرد بهتر شود. مدیریت ویژگی‌های نامربوط: در وظایف طبقه‌بندی متن، داشتن تعداد زیادی ویژگی (کلمات یا اصطلاحات) که ممکن است برای طبقه‌بندی مرتبط نباشد، معمول است. رگرسیون لجستیک این توانایی را

دارد که وزن‌های کمتر یا نزدیک به صفر را به این ویژگی‌های نامربوط اختصاص دهد و تأثیر آن‌ها را در تصمیم‌گیری طبقه‌بندی کم‌اهمیت جلوه دهد. این قابلیت انتخاب ویژگی می تواند عملکرد را با کاهش نویز در داده ها بهبود بخشد.

If we used the Linear Regression model:

استفاده از رگرسیون خطی برای یک کار طبقه بندی متن یک رویکرد رایج نیست، زیرا رگرسیون خطی در درجه اول برای کارهای رگرسیونی که متغیر هدف پیوسته است استفاده می شود. با این حال، اگر از رگرسیون خطی برای طبقه‌بندی متن استفاده کنید، نتایج به احتمال زیاد رضایت‌بخش نخواهد بود.

مدل های رگرسیون خطی با هدف تخمین یک متغیر خروجی پیوسته بر اساس یک رابطه خطی با ویژگی های ورودی است. در مورد طبقه‌بندی متن، متغیر هدف نشان‌دهنده دسته‌های گسسته است.

رگرسیون خطی فرض می کند که رابطه بین ویژگی های ورودی و متغیر هدف خطی است. با این حال، در وظایف طبقه‌بندی متن، رابطه معمولاً غیرخطی و پیچیده‌تر است. رگرسیون خطی ممکن است الگوهای پیچیده و غیرخطی بودن موجود در داده های متنی را نشان ندهد که منجر به عملکرد ضعیف می شود.

رگرسیون خطی فرض می کند که باقیمانده ها (تفاوت بین مقادیر پیش بینی شده و واقعی) از توزیع نرمال با واریانس ثابت پیروی می کنند. در طبقه‌بندی متن، به دلیل ماهیت داده‌های متنی، ممکن است پرت و ناهمسانی (واریانس خطای متغیر) وجود داشته باشد. این نقض مفروضات می تواند بر عملکرد مدل و قابلیت اطمینان پیش بینی های آن تأثیر منفی بگذارد.

Problem 3: Face Recognition

1- Create Dataset

برای ایجاد کردن دیتاست از fetch_lfw_people در کتابخانه sklearn استفاده شده و از هر شخص ۱ عکس انتخاب شده و حداکثر تعداد این عکس ها ۷۰۰ در نظر گرفته شده، در نهایت صورت هر شخص در عکس ها استخراج شده و در فولدر Other ذخیره شده.

```
os.makedirs(destination_dir, exist_ok=True)

count = 0
# Iterate through the folders in the source directory
for folder_name in os.listdir(source_dir):
    folder_path = os.path.join(source_dir, folder_name)

    # Skip any non-directory items
    if not os.path.isdir(folder_path):
        continue

    # Iterate through the image files in each folder
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)

        # Skip any non-image files
        if not file_name.lower().endswith(('.jpg', '.jpeg', '.png', '.gif')):
            continue


        image = cv2.imread(file_path)
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray,scaleFactor=1.1, minNeighbors=10, minSize=(40, 40))

        if len(faces) > 0:

          for i, (x,y,w,h) in enumerate(faces):

              # To draw a rectangle in a face
              face = image[y:y+h, x:x+w]
              image_file = os.path.join(destination_dir, f"image_{count}.jpg")
              cv2.imwrite(image_file, face)
        count+=1
        break
    if count > 700:
        break
print("Images copied successfully!")
```

```
Images copied successfully!
```

پوشه ای از عکس های خودم ایجاد کردم و همه‌ی آن عکس ها را خوانده و صورت را استخراج کرده و در پوشه Zahra ذخیره کردم.

```
count = 0
os.makedirs(destination_dir, exist_ok=True)
for file_name in source_dir:
    if not file_name.lower().endswith(('.jpg', '.jpeg', '.png', '.gif')):
            continue
    image = cv2.imread(file_name)
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray,scaleFactor=1.1, minNeighbors=10, minSize=(40, 40))

    if len(faces) > 0:

       for i, (x,y,w,h) in enumerate(faces):

        # To draw a rectangle in a face
        face = image[y:y+h, x:x+w]
        image_file = os.path.join(destination_dir, f"image_{count}.jpg")
        cv2.imwrite(image_file, face)

        count += 1
print("Face images saved successfully!")
```

2- Preprocessing

صورت های استخراج شده از تصاویر از فایل Zahra , Other خوانده شده و پیش پردازش های زیر روی تصاویر انجام شده:

- اندازه هر تصویر به (128,128) تغییر میکند. تصویر به آرایه تبدیل شده و به ۱ بعد کاهش می یابد

```
import cv2
import numpy

def preprocess_img(filepath):

  try:
    img = cv2.imread(filepath)

    img = cv2.resize(img,target_size)

    img_array = np.array(img).flatten()
    #img_array = np.array(img)

    #img_array = img_array / 255.0

    return img_array

  except Exception as e:
                print(f"Error processing image {filepath}: {e}")
```

- برای تصاویر خوانده شده برای zahra برچسب ۱ و برای Other برچسب صفر در نظر گرفته میشود.

3- Split the data into training and testing

۰.۲ داده ها برای آزمایش مدل در نظر گرفته میشود

```
from sklearn.model_selection import train_test_split
import numpy as np

images = np.array(images)
labels = np.array(labels)
train_images, test_images, train_labels, test_labels = train_test_split(images, labels, train_size=0.8, random_state=
```

4- Model Training

مدل های Random Forest Classifier و SVC و Logistic Regression برای آموزش انتخاب شده و نتایج آنها مقایسه شده و در نهایت بهترین انتخاب شده است.

```
Accuracy: 0.9726027397260274
Precision: 1.0
Recall: 0.9310344827586207
F1 score: 0.9642857142857143
confusion_matrix:
 [[132   0]
 [  6  81]]
```

نتایج حاصل از Random Forest Classifier :

```
Accuracy: 0.9497716894977168
confusion_matrix:
 [[125   7]
 [  4  83]]
```

نتایج حاصل از SVC:

نتایج Logistic Regression :

```
Accuracy: 0.9726027397260274
Precision: 0.9550561797752809
Recall: 0.9770114942528736
F1 score: 0.9659090909090908
confusion_matrix:
 [[128   4]
 [  2  85]]
```

با توجه به نتایج بالا Random Forest Classifier و Logistic Regression دارای بهترین نتایج و دقت شبیه به یکدیگر می باشند، در نهایت Random Forest به عنوان مدل نهایی انتخاب و ذخیره شده است.

با استفاده از cv2.VideoCapture امکان استفاده از webcam وجود دارد، همانطور که در کد زیر مشخص است، در هنگام اجرا ۲۰ فریم دریافت شده و صورت موجود در این فریم ها ذخیره شده است.

```
try:
    while True:
       ret, frame = cap.read()
       if not ret:
           break


       gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
       faces = face_cascade.detectMultiScale(gray,scaleFactor=1.1, minNeighbors=10, minSize=(40, 40))

       if len(faces) > 0:

         for i, (x,y,w,h) in enumerate(faces):

            # To draw a rectangle in a face
            face = frame[y:y+h, x:x+w]
            frames.append(face)

       if forms_started >= max_forms:
          who = frame
          who_faces = faces
          break

       forms_started+=1

except:
    print(f'Video has ended..')
```

در نهایت بعد از ۲۰ فریم با استفاده از cap.release و cv2.destroyAllWindows وب کم خاموش میشود.

صورت های موجود در ۲۰ فریم، برای پیش بینی به مدل داده میشود و اگر ۵ فریم پیش بینی ۱ باشد، hello zahra و عکس فریم آخر نمایش داده میشود و در غیر این صورت can not login و عکس فریم آخر نمایش داده میشود.

در Prob درصد حضور و یا عدم حضور میان صورت فریم ها محاسبه میشود

```
total = 0
for img in frames:

    img = cv2.resize(img,(128, 128))
    img_array = np.array(img).flatten()


    predict =  model.predict(img_array.reshape(1, -1))
    total+=1
    if predict ==1 :
        consecutive_recognitions +=1

if consecutive_recognitions > required_recognitions :
    color = (0, 255, 0)
    prob = (consecutive_recognitions / total)*100
    label = f'Hello Zahra    {prob}%'
else:
    color = (0, 0, 255)
    prob = (1- (consecutive_recognitions / total))*100
    label = f'can not login       {prob}%'


if len(who_faces) > 0:
   for i, (x,y,w,h) in enumerate(faces):

      # To draw a rectangle in a face
      cv2.rectangle(who,(x,y),(x+w,y+h),color,2)
      cv2.putText(who, label, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, color, 2)

   cv2.imshow("image",who)
   cv2.waitKey(0)
   cv2.destroyAllWindows()
```