بسمه تعالی

دانشگاه تهران

دانشکده مهندسی مکانیک

تمرین سری دوم درس هوش مصنوعی

نام و نام خانوادگی:

محمدمهدی تویسرکانی

شماره دانشجویی:

۸۱۰۶۰۳۰۰۵

تاریخ تحویل:

۱۴۰۴/۰۱/۲۸

نیم‌سال دوم سال تحصیلی ۱۴۰۳-۱۴۰۴

فهرست مطالب

عنوان **صفحه**

هوش مصنوعی و یادگیری ماشین (رگرسیون)

❖ اطلاعات مسئله:

هدف این تمرین آشنایی با روش‌های پیش‌پردازش داده‌ها و ارزیابی کارآیی مدل‌های مختلف رگرسیون است.
دادگانی که برای این کار در نظر گرفته شده است (فایل Housing.csv) شامل ۲۹۳۰ داده مربوط به فروش خانه‌های مسکونی در یکی از شهرهای آمریکا در فاصلهٔ سال‌های ۲۰۰۶ و ۲۰۱۰ است که قیمت فروش خانه‌ها را بر پایهٔ ۸۰ ویژگی آن‌ها از جمله مساحت زمین، تعداد اتاق‌ها، سال ساخت، محله، نوع سقف و کیفیت ساخت (ضعیف تا عالی) نشان می‌دهد. هدف ما یافتن بهترین مدلی است که می‌تواند با استفاده از این ویژگی‌ها قیمت خانه را پیش‌بینی کند.

۱- دادگان را به صورت dataframe خوانده و با استفاده از روش info اطلاعات کلی آن (شامل تعداد مقادیر موجود برای هر یک از ویژگی‌ها) را نمایش دهید.
۲- در صورت وجود داده‌های پرت (outlier) آن‌ها را حذف کنید و مقادیر ناموجود (missing value) را (با ذکر روش بکار گرفته شده) با مقادیر مناسب جایگزین کنید.
۳- اطلاعات آماری دادگان را بررسی کنید (مقادیر کمینه، بیشینه و انحراف از معیار را برای دادگان بدست آورید.)
۴- با استفاده از ماتریس همبستگی، ویژگی‌هایی را که بیشترین تاثیر را بر قیمت خانه دارند مشخص کنید (بخش توضیحات را ببینید).
۵- برای مشخص‌تر کردن ویژگی‌هایی (از میان ویژگی‌های انتخاب شده در بند ۴) که بیشترین تاثیر را بر قیمت خانه دارند، با استفاده از کتابخانهٔ seaborn و دستور jointplot، jointplot مربوط به این ویژگی‌ها را رسم کنید.
۶- با استفاده از دستور SelectKBest در کتابخانهٔ scikit-learn، تعداد ویژگی‌ها را به گونه ای انتخاب کنید که مدل‌های رگرسون بیشترین دقت را داشته باشند (از آنجا که هدف این تمرین ارزیابی کارآیی مدل‌های رگرسیون است باید از آزمون f_regression استفاده کنید.)
۷- دادگان را به دو بخش آموزش (training) و آزمون (test) تقسیم کنید. (random_state = 42, test_size = 0.25)
(انتخاب عدد ۴۲ دلیل خاصی ندارد و صرفا کمک می‌کند که در همهٔ اجراها عدد تصادفی یکسانی تولید شود تا نتایج این اجراها قابل مقایسه باشند.)
۸- به کمک داده‌های آموزش و با استفاده از کتابخانهٔ scikit-learn مدل‌های Linear Regression، Lasso Regression، Ridge Regression و Polynomial Regression را آموزش دهید.
۹- توضیح دهید که خطای RMS و معیار R^2 score چگونه محاسبه می‌شوند. سپس مقدار آن‌ها را برای هر یک از مدل‌های بالا روی داده‌های آزمون محاسبه و گزارش کنید.
۱۰-توضیح دهید که bias-variance trade-off چیست و چگونه بر عملکرد مدل‌های یادگیری ماشین تأثیر می‌گذارد. سپس با ارایهٔ یک مثال نشان دهید که افزایش پیچیدگی مدل چگونه بر خطاهای بایاس و واریانس تأثیر می‌گذارد.

توضیحات:

ماتریس همبستگی (Correlation Matrix)

این ماتریس که درایه‌های آن ضرایب همبستگی بین متغیرها (در اینجا ویژگی‌های) مختلف است میزان و جهت همبستگی خطی بین این متغیرها را مشخص می‌کند. همبستگی مثبت به این معنی است که با افزایش یک متغیر، دیگری هم افزایش می‌یابد و همبستگی منفی به این معنی است که با افزایش یک متغیر، دیگری کاهش می‌یابد. میزان همبستگی دو متغیر با عددی در بازهٔ ۱- تا ۱+ نشان داده می‌شود. مقدار ۱+ نشان دهنده همبستگی مثبت کامل، مقدار ۱- نشان‌دهنده همبستگی

منفی کامل و مقدار صفر نشان‌دهنده عدم همبستگی دو متغیر است. همبستگی بین ویژگی‌ها از رابطهٔ زیر بدست می‌آید که در آن $\bar{x}$ و $\bar{y}$ میانگین مقادیر دو ویژگی است. x_i و y_i نیز نشان‌دهنده مقادیر دو ویژگی در داده‌های مختلف است.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

رگرسیون‌هاي Ridge و Lasso

در کاربردهایی با تعداد ویژگی زیاد می‌توان از رگرسیون‌هاي ریج و لاسو براي ایجاد مدل‌هاي آماري ساده‌تر و قابل اعتمادتر استفاده کرد. رگرسیون Ridge با کنترل تأثیر هر ویژگی بر خروجی مدل و پیشگیري از تاثیر بیش از حد برخی ویژگی‌ها بر خروجی، به افزایش دقت و تعمیم‌پذیري مدل کمک می‌کند ولی همهٔ متغیرها را در مدل نگه می‌دارد

رگرسیون Lasso علاوه بر این کنترل، ویژگی‌هاي کم‌تاثیر بر خروجی را نیز حذف می‌کند و به مدل کمک می‌کند که بر مهم‌ترین ویژگی‌ها تمرکز کند. هر دو روش شامل یک جملهٔ عادي‌ساز (Regularizer) هستند که میزان ساده‌سازي مدل را کنترل می‌کند. این جمله در رگرسیون ریج، نرم ۲ ضرایب وزنی هر ویژگی و در رگرسیون لاسو، نرم ۱ این ضرایب را به تابع هزینه می‌افزاید. براي آشنایی بیشتر با این دو روش می‌توانید به لینک زیر مراجعه کنید.

https://medium.com/@devsachin0879/ridge-regression-and-lasso-regression-a-beginners-guide-b3e33c77678

لینک‌های مفید:

- راهنمایی دستور SelectKBest:

https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html

- راهنمایی پیاده‌سازی Linear Regression، Ridge Regression، Lasso Regression، Polynomial Regression

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.PolynomialFeatures.html

چند تذکر:

- تحویل گزارش این تمرین ضروری است و به تمرین بدون گزارش نمره‌ای تعلق نمی‌گیرد. حجم گزارش معیاری برای ارزیابی نخواهد بود و لزومی به توضیح جزئیات کد نیست؛ اما از آنجا که برای این تمرین از کتابخانه‌های موجود استفاده می‌کنید لطفا تمامی پارامترهای تنظیم‌شده در هر قسمت از کد را گزارش کرده و فرض‌هایی را که برای پیاده‌سازی‌ها و محاسبات خود به کار برده‌اید ذکر کنید. از ارائهٔ توضیحات کلیشه‌ای و همانند بَرداری از منابع موجود بپرهیزید.

❖ حل مسئله:

این مسئله یک تحلیل و رویکرد مدل‌سازی پیش‌بینی‌کننده برای پیش‌بینی قیمت مسکن با استفاده از روش‌های رگرسیون ارائه می‌کند. مجموعه داده شامل ۲۹۳۰ ورودی با ۸۲ ویژگی است که ویژگی‌های مسکن مختلف را پوشش می‌دهد. هدف اولیه توسعه و ارزیابی مدل‌های رگرسیون برای تخمین قیمت‌های فروش است.

برای حل این مسئله، کد زیر در نرم افزار Python نوشته شده و روش‌های پیش‌پردازش داده‌ها و ارزیابی کارایی مدل‌های مختلف رگرسیون مورد بررسی قرار می‌گیرد.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.feature_selection import SelectKBest, f_regression
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.metrics import mean_squared_error, r2_score
from scipy.stats import zscore
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("Housing.csv")

df.info()

df.fillna(df.median(numeric_only=True), inplace=True)

df = df[(np.abs(zscore(df.select_dtypes(include=[np.number]))) < 3).all(axis=1)]

print(df.describe())

label_encoders = {}
for col in df.select_dtypes(include=["object"]).columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col].astype(str))
    label_encoders[col] = le

plt.figure(figsize=(12, 8))
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=False, cmap="coolwarm")
```

```
plt.title("Correlation Matrix")
plt.show()

top_corr_features = corr_matrix['SalePrice'].abs().sort_values(ascending=False)[1:6].index
print("Top correlated features:", top_corr_features)

for feature in top_corr_features:
    sns.jointplot(data=df, x=feature, y='SalePrice', kind='reg')
    plt.show()

X = df.drop(columns=['SalePrice'])
y = df['SalePrice']
selector = SelectKBest(score_func=f_regression, k=10)
X_selected = selector.fit_transform(X.select_dtypes(include=[np.number]), y)
selected_features = X.select_dtypes(include=[np.number]).columns[selector.get_support()]
print("Selected Features:", selected_features)

X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=0.25,
random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

models = {
    "Linear Regression": LinearRegression(),
    "Lasso Regression": Lasso(alpha=0.1),
    "Ridge Regression": Ridge(alpha=1.0),
    "Polynomial Regression (degree=2)": PolynomialFeatures(degree=2)
}

for name, model in models.items():
    if name == "Polynomial Regression (degree=2)":
        poly = PolynomialFeatures(degree=2)
        X_train_poly = poly.fit_transform(X_train)
        X_test_poly = poly.transform(X_test)
        poly_model = LinearRegression()
        poly_model.fit(X_train_poly, y_train)
        y_pred = poly_model.predict(X_test_poly)
    else:
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)

    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
```

```
    r2 = r2_score(y_test, y_pred)
    print(f"{name} - RMSE: {rmse}, R²: {r2}")
```

در ادامه، جزئیات و مفهوم هر بخش از کد فوق به همراه پاسخ به سوالات توضیح داده می‌شود:

✓ بخش اول کد: معرفی کتابخانه‌های استفاده شده

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from sklearn.feature_selection import SelectKBest, f_regression
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.metrics import mean_squared_error, r2_score
from scipy.stats import zscore
from sklearn.preprocessing import LabelEncoder
```

pandas (pd): کار با داده و تحلیل داده‌ها را انجام می‌دهد.

numpy (np): از محاسبات عددی پشتیبانی می‌کند.

seaborn (sns): قابلیت نمایش نمودارها را فراهم می‌کند.

matplotlib.pyplot (plt): برای ایجاد نمودار استفاده می‌شود.

sklearn.model_selection.train_test_split: مجموعه داده را به مجموعه‌های آموزشی و آزمایشی تقسیم می‌کند.

sklearn.preprocessing.StandardScaler: ویژگی‌ها را نرمال می‌کند (آنها را به صورت میانگین ۰ و واریانس ۱ مقیاس می‌کند).

sklearn.preprocessing.PolynomialFeatures: ویژگی‌های چند جمله‌ای را برای رگرسیون ایجاد می‌کند.

sklearn.feature_selection.SelectKBest: بهترین ویژگی‌ها را بر اساس امتیازات آماری انتخاب می‌کند.

sklearn.feature_selection.f_regression: از همبستگی (F-statistics) برای انتخاب ویژگی استفاده می‌کند.

sklearn.linear_model.LinearRegression: رگرسیون خطی را پیاده سازی می‌کند.

sklearn.linear_model.Lasso: رگرسیون لاسو (قانونی سازی L1) را پیاده سازی می‌کند.

sklearn.linear_model.Rridge: رگرسیون Ridge (قانونی سازی L2) را پیاده سازی می‌کند.

sklearn.metrics.mean_squared_error: RMSE را محاسبه می‌کند.

sklearn.metrics.r2_score: R^2 را محاسبه می‌کند.

scipy.stats.zscore: امتیاز Z را برای تشخیص مقادیر پرت محاسبه می‌کند.

sklearn.preprocessing.LabelEncoder: متغیرهای طبقه بندی شده را در مقادیر عددی رمزگذاری می‌کند.

✓ بخش دوم کد: پاسخ سوال ۱

```
df = pd.read_csv("Housing.csv")
```

مجموعه داده را از یک فایل CSV در یک DataFrame (df) می‌خواند.

```
df.info()
```

اطلاعات کلی درمورد مجموعه داده‌ها را چاپ می‌کند، از جمله: تعداد مقادیر غیرتهی و انواع داده‌های هر ستون که در شکل زیر نشان داده شده است.

```
df.info()
```
Python
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2930 entries, 0 to 2929
Data columns (total 82 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Order            2930 non-null   int64
 1   PID              2930 non-null   int64
 2   MS SubClass      2930 non-null   int64
 3   MS Zoning        2930 non-null   object
 4   Lot Frontage     2440 non-null   float64
 5   Lot Area         2930 non-null   int64
 6   Street           2930 non-null   object
 7   Alley            198 non-null    object
 8   Lot Shape        2930 non-null   object
 9   Land Contour     2930 non-null   object
 10  Utilities        2930 non-null   object
 11  Lot Config       2930 non-null   object
 12  Land Slope       2930 non-null   object
 13  Neighborhood     2930 non-null   object
 14  Condition 1      2930 non-null   object
 15  Condition 2      2930 non-null   object
 16  Bldg Type        2930 non-null   object
 17  House Style      2930 non-null   object
 18  Overall Qual     2930 non-null   int64
 19  Overall Cond     2930 non-null   int64
...
 80  Sale Condition   2930 non-null   object
 81  SalePrice        2930 non-null   int64
dtypes: float64(11), int64(28), object(43)
memory usage: 1.8+ MB
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

شکل ۱: نمایش اطلاعات کلی مجموعه دادگان

✓ بخش سوم کد: پاسخ سوال ۲

➢ پیش پردازش داده‌ها

- **مدیریت مقادیر عددی ناموجود (Missing Values)**

مقادیر عددی ناموجود با میانه ستون‌های مربوطه پر شده است. از میانه (df.median(numeric_only=True استفاده می‌شود، زیرا نسبت به نقاط پرت مقاوم است.

متغیرهای دسته‌بندی دارای مقادیر ناموجود با برچسب، کدگذاری شده و برای مراجعات بعدی ذخیره شدند.

- **حذف نقاط پرت**

روش Z-score برای حذف نقاط پرت دارای بیش از ۳ انحراف استاندارد، استفاده شده است.

Threshold: |Z-score| < 3.

- **توضیحات مربوط به کد این بخش:**

```
df.fillna(df.median(numeric_only=True), inplace=True)
```

مقادیر ناموجود در ستون‌های عددی با مقادیر میانه آنها جایگزین می‌شود.

```
df = df[(np.abs(zscore(df.select_dtypes(include=[np.number]))) < 3).all(axis=1)]
```

مقدار Z-score برای ستون‌های عددی محاسبه می‌شود. سطرهایی که در آنها هر ویژگی عددی دارای امتیاز Z بیشتر از ۳ باشد، حذف می‌شود.

✓ بخش چهارم کد: پاسخ سوال ۳

```
print(df.describe())
```

نمایش اطلاعات آماری دادگان مانند میانگین، انحراف معیار، حداقل و حداکثر مقادیر برای ستون‌های عددی.

در شکل زیر اطلاعات آماری دادگان شامل تعداد، میانگین، انحراف معیار، حداقل، صدک ۲۵، میانه (صدک ۵۰)، صدک ۷۵، و حداکثر آورده شده است.

```
print(df.describe())
```

```
[6]                                                                    Python

...              Order           PID  MS SubClass  Lot Frontage      Lot Area  \
count  2051.000000  2.051000e+03  2051.000000   2051.000000   2051.000000
mean   1464.613359  7.090646e+08    54.934178     66.586543   9083.000975
std     848.677739  1.886374e+08    40.337395     18.493234   3633.678767
min       2.000000  5.263020e+08    20.000000     21.000000   1300.000000
25%     736.500000  5.284310e+08    20.000000     60.000000   7200.000000
50%    1469.000000  5.354041e+08    50.000000     68.000000   9037.000000
75%    2190.500000  9.071870e+08    60.000000     75.000000  10965.000000
max    2930.000000  9.241510e+08   180.000000    130.000000  33120.000000

       Overall Qual  Overall Cond   Year Built  Year Remod/Add  Mas Vnr Area  \
count   2051.000000   2051.000000  2051.000000     2051.000000   2051.000000
mean       6.109703      5.526085  1974.820088     1985.681131     84.010239
std        1.345329      0.976143    29.514136       20.934083    134.251135
min        2.000000      3.000000  1885.000000     1950.000000      0.000000
25%        5.000000      5.000000  1955.000000     1967.000000      0.000000
50%        6.000000      5.000000  1978.000000     1995.000000      0.000000
75%        7.000000      6.000000  2003.000000     2004.000000    148.000000
max       10.000000      8.000000  2010.000000     2010.000000    632.000000

       ...  Wood Deck SF  Open Porch SF  Enclosed Porch   3Ssn Porch  \
count  ...   2051.000000    2051.000000     2051.000000  2051.000000
mean   ...     87.035105      43.425158       15.346173     0.011214
std    ...    106.716130      54.047853       43.726769     0.507861
min    ...      0.000000       0.000000        0.000000     0.000000
...
75%    209250.000000
max    418000.000000

[8 rows x 39 columns]
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

شکل۲: نمایش اطلاعات آماری دادگان

✓ بخش پنجم کد: شناسایی ستون‌های طبقه‌بندی شده و تبدیل به مقادیر عددی

➢ رمزگذاری ویژگی

رمزگذاری برچسب برای متغیرهای طبقه‌بندی اعمال شده است. رمزگذاری برچسب LabelEncoder() برای متغیرهای طبقه‌ای که روابط ترتیبی ندارند، مناسب است.

```
label_encoders = {}
for col in df.select_dtypes(include=["object"]).columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col].astype(str))
    label_encoders[col] = le
```

ستون‌های طبقه‌بندی شده شناسایی می‌شود و سپس با استفاده از LabelEncoder به مقادیر عددی تبدیل می‌شود. پاسخ کد مربوط به این بخش در شکل زیر آورده شده است:

جدول زیر آمار مجموعه داده را خلاصه می‌کند و تعداد، میانگین، انحراف معیار، حداقل، صدک ۲۵، میانه (صدک ۵۰)، صدک ۷۵، و حداکثر برای هر ویژگی عددی را نشان می‌دهد.

جدول۱: اطلاعات آماری دادگان شامل تعداد، میانگین، انحراف معیار، حداقل، صدک ۲۵، میانه (صدک ۵۰)، صدک ۷۵، و حداکثر

Max	75%	50% (Median)	25%	Min	Std Dev	Mean	Count	Column
2930	2190.5	1469	736.5	2	848.68	1464.61	2051	**Order**
9.24e+08	9.07e+08	5.35e+08	5.28e+08	5.26e+08	1.89e+08	7.09e+08	2051	**PID**
180	60	50	20	20	40.34	54.93	2051	**MS SubClass**
130	75	68	60	21	18.49	66.59	2051	**Lot Frontage**
33120	10965	9037	7200	1300	3633.68	9083.00	2051	**Lot Area**
10	7	6	5	2	1.35	6.11	2051	**Overall Qual**
8	6	5	5	3	0.98	5.53	2051	**Overall Cond**
2010	2003	1978	1955	1885	29.51	1974.82	2051	**Year Built**
2010	2004	1995	1967	1950	20.93	1985.68	2051	**Year Remod/Add**
632	148	0	0	0	134.25	84.01	2051	**Mas Vnr Area**
468	168	0	0	0	106.72	87.04	2051	**Wood Deck SF**
247	66	29	0	0	54.05	43.43	2051	**Open Porch SF**
214	0	0	0	0	43.73	15.35	2051	**Enclosed Porch**
23	0	0	0	0	0.51	0.01	2051	**3Ssn Porch**
184	0	0	0	0	31.61	7.27	2051	**Screen Porch**
0	0	0	0	0	0.00	0.00	2051	**Pool Area**
1500	0	0	0	0	102.75	13.20	2051	**Misc Val**
12	8	6	4	1	2.68	6.14	2051	**Mo Sold**
2010	2009	2008	2007	2006	1.32	2007.78	2051	**Yr Sold**
418000	209250	160000	128700	35000	66022.62	174511.24	2051	**SalePrice**

✓ بخش ششم کد: پاسخ سوال ۴

➢ انتخاب ویژگی

انتخاب مبتنی بر همبستگی sort_values(ascending=False)[1:6] و فقط ویژگی‌های عددی در نظر گرفته می‌شود.

پنج ویژگی برتر مرتبط با SalePrice عبارتند از:

۱- Overall Qual

۲- Gr Liv Area

۳- Garage Cars

۴- Garage Area

۵- Exter Qual

✓ بخش هفتم کد: پاسخ سوال ۵

```
plt.figure(figsize=(12, 8))
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=False, cmap="coolwarm")
plt.title("Correlation Matrix")
plt.show()
  end
```

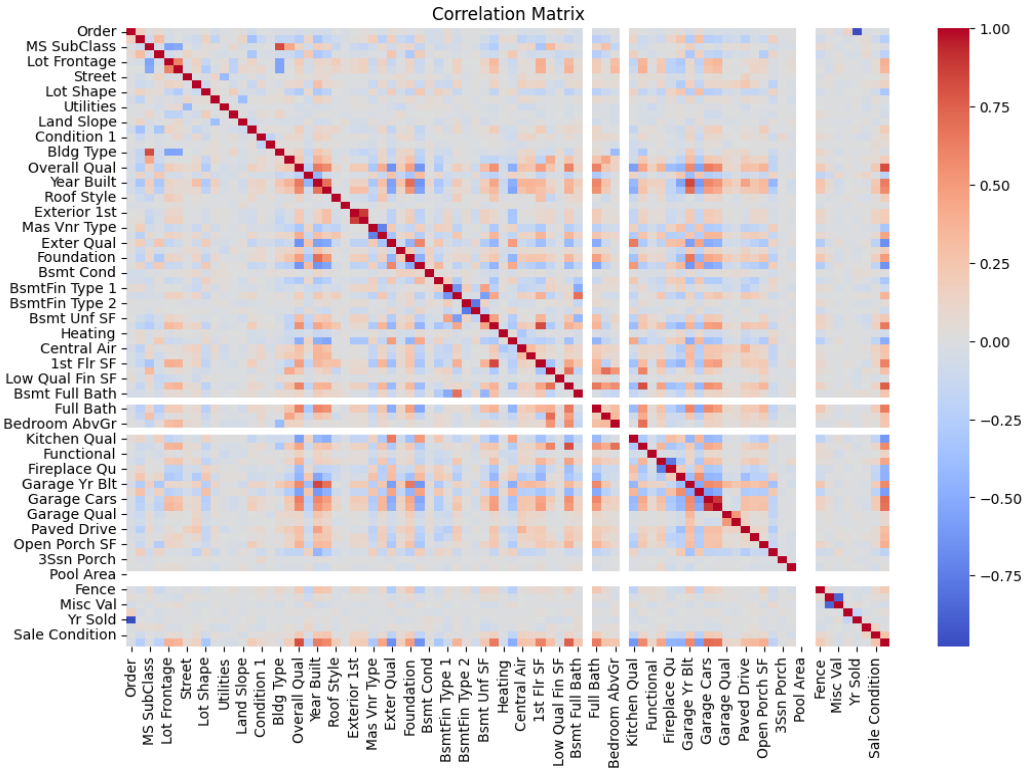
ماتریس همبستگی محاسبه می‌شود. از seaborn برای نمایش همبستگیِ بین ویژگی‌ها در یک نقشه حرارتی استفاده می‌شود.

```
top_corr_features = corr_matrix['SalePrice'].abs().sort_values(ascending=False)[1:6].index
print("Top correlated features:", top_corr_features)
```

ویژگی‌ها بر اساس همبستگی مطلق آنها با قیمت فروش، مرتب شده و ۵ ویژگی برتر انتخاب می‌شود (به استثنای خود SalePrice).

```
for feature in top_corr_features:
  sns.jointplot(data=df, x=feature, y='SalePrice', kind='reg')
  plt.show()
```

نمودار joinplot با خطوط رگرسیون برای هر ویژگی انتخاب شده رسم می‌شود. در شکل ۱ نمودار ماتریس همبستگی رسم شده است.



شکل۳: نمودار ماتریس همبستگی

✓ بخش هشتم کد: پاسخ سوال ۶

روش KBest انتخاب شده است و ۱۰ ویژگی برتر دارای بیشترین توانایی پیش‌بینی هستند. ۱۰ ویژگی عددی انتخاب شده با استفاده از f_regression عبارتند از:

۱- Overall Qual

۲- Year Built

۳- Exter Qual

۴- Bsmt Qual

۵- Total Bsmt S

۶- st Flr SF1

۷- Gr Liv Area

۸- Full Bath

۹- Garage Cars

۱۰- Garage Area

```
X = df.drop(columns=['SalePrice'])
y = df['SalePrice']
selector = SelectKBest(score_func=f_regression, k=10)
X_selected = selector.fit_transform(X.select_dtypes(include=[np.number]), y)
selected_features = X.select_dtypes(include=[np.number]).columns[selector.get_support()]

print("Selected Features:", selected_features)
```

X: ویژگی‌ها (به استثنای SalePrice).

Y: متغیر هدف (SalePrice).

SelectKBest: ۱۰ ویژگی عددی مرتبط با استفاده از f_regression (همبستگی آماری با SalePrice) انتخاب می‌شود.

✓ بخش نهم کد: پاسخ سوال ۷

➢ تقسیم و مقیاس بندی داده‌ها

مجموعه داده‌ها به مجموعه‌های آموزشی (۷۵٪) و آزمایشی (۲۵٪) تقسیم شدند. حالت تصادفی: ۴۲ (برای تکرارپذیری).

استانداردسازی با استفاده از StandardScaler برای نرمال‌سازی مقادیر ویژگی اعمال شده است. استانداردسازی (میانگین = ۰، واریانس = ۱) عملکرد مدل را بهبود می‌بخشد.

- **توضیحات مربوط به کد این بخش:**

```
X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=0.25, random_state=42)
```

مجموعه داده را به موارد زیر تقسیم می کند:

X_train, y_train: داده‌های آموزش (۷۵٪).

X_test، y_test: داده‌های آزمایش (۲۵٪).

random_state=42: تکرارپذیری انجام می‌شود.

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

StandardScaler: ویژگی‌ها را باتوجه به میانگین ۰ و واریانس ۱ مقیاس می‌کند.

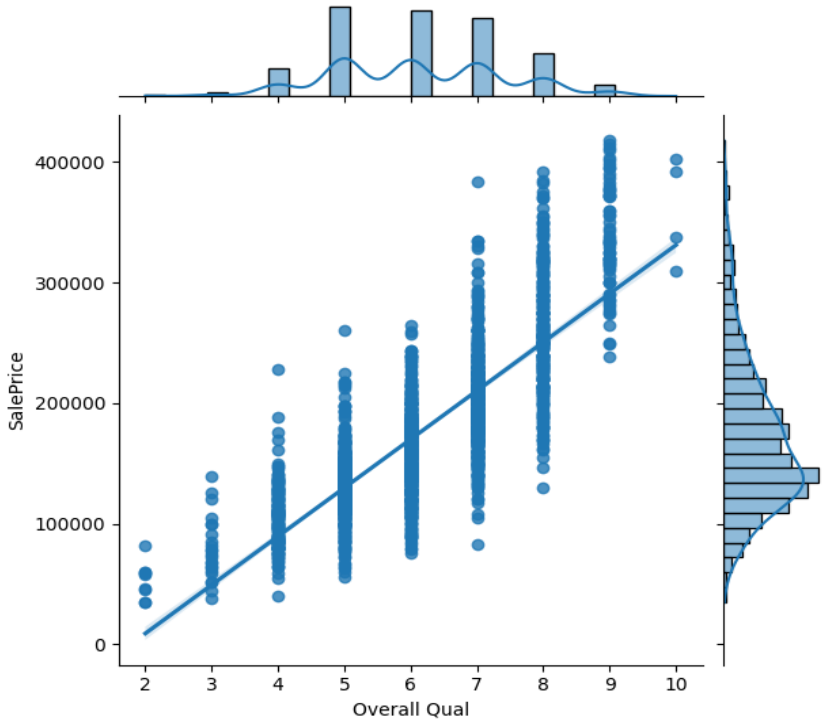
✓ بخش دهم کد: پاسخ سوال ۸

```
models = {
    "Linear Regression": LinearRegression(),
    "Lasso Regression": Lasso(alpha=0.1),
    "Ridge Regression": Ridge(alpha=1.0),
    "Polynomial Regression (degree=2)": PolynomialFeatures(degree=2)
}
```

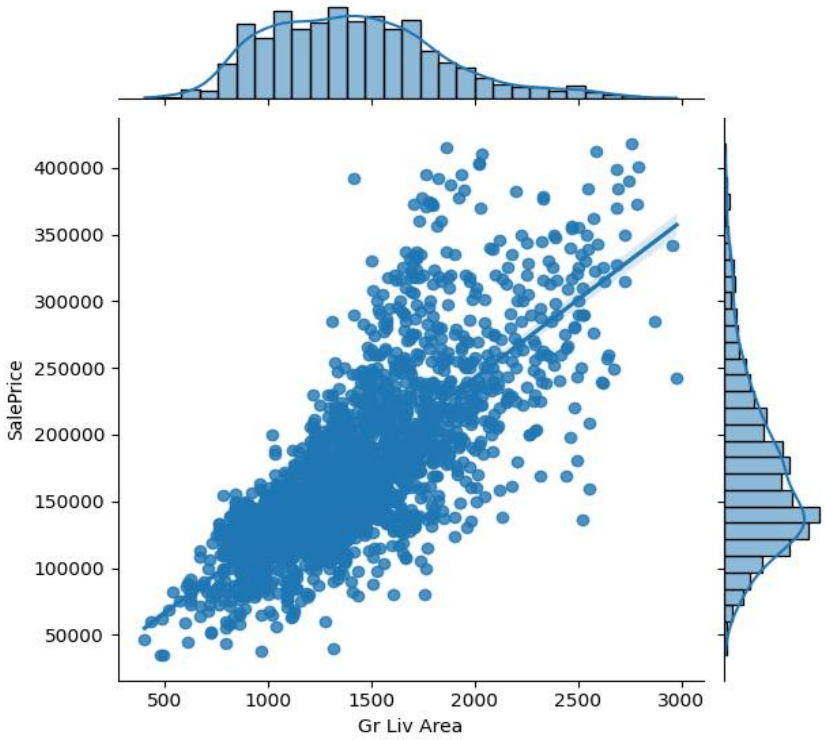
چهار مدل رگرسیون تعریف می‌شود:

- رگرسیون خطی (مدل پایه).
- رگرسیون لاسو (Lasso(alpha=0.1)) ویژگی‌های با اهمیت کمتر را به صفر می‌رساند (قانونی سازی L1).
- رگرسیون ریج (ریج (آلفا=۱.۰)) ضرایب را برای جلوگیری از overfitting (قانونی سازی L2) کوچک می‌کند.
- رگرسیون چند جمله‌ای (PolynomialFeatures (درجه=۲) ویژگی‌های مربعی ایجاد می‌کند.

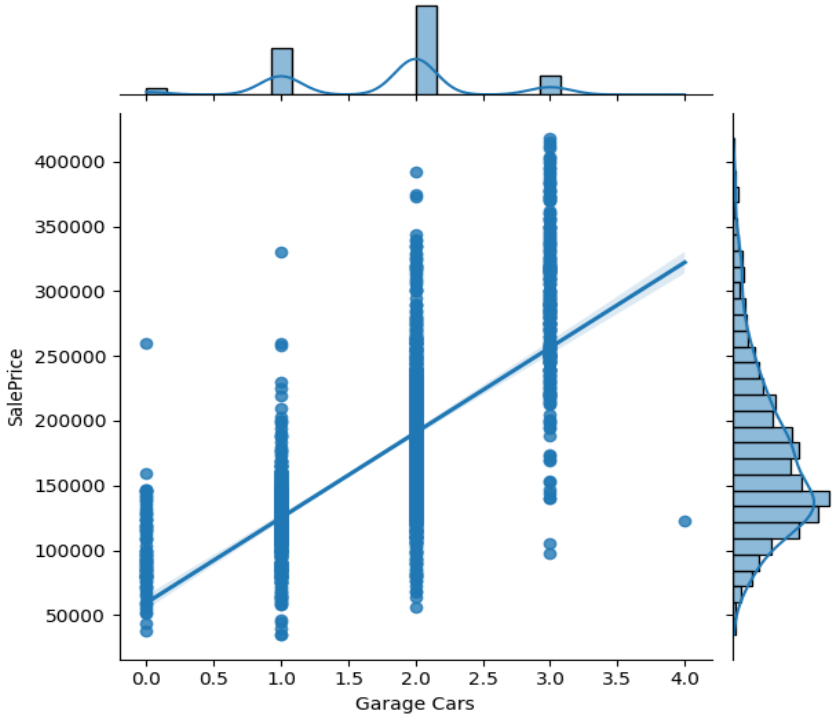
نمودارهای زیر پراکندگی با یک خط رگرسیون خطی را نشان می‌دهد و همچنین شامل توزیع هر ویژگی روی محورها می‌شود.



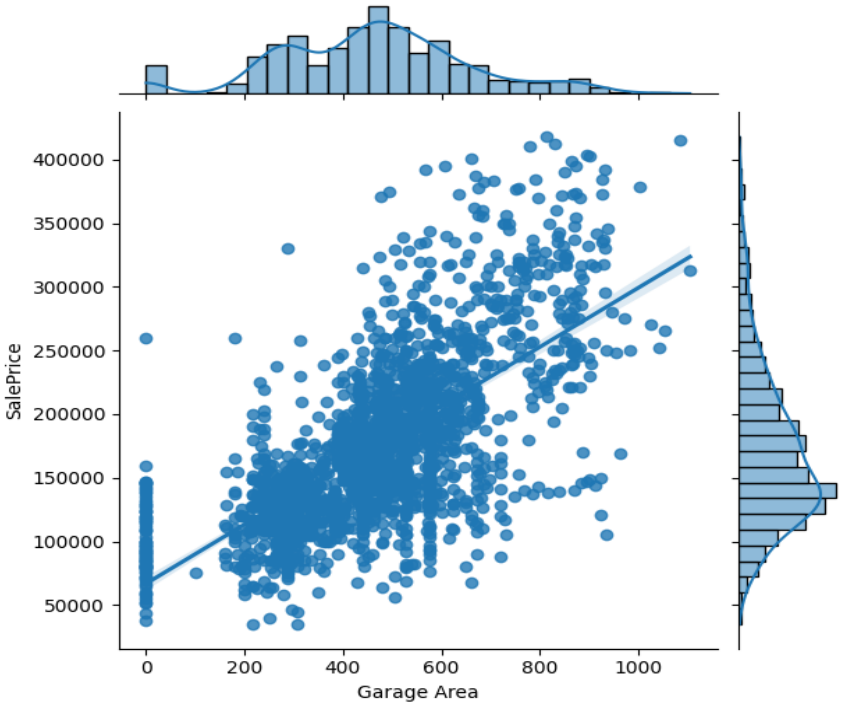
شکل۴: نمودار پراکندگی با خط رگرسیون خطی به همراه توزیع ویژگی Overall Qual نسبت به SalePrice



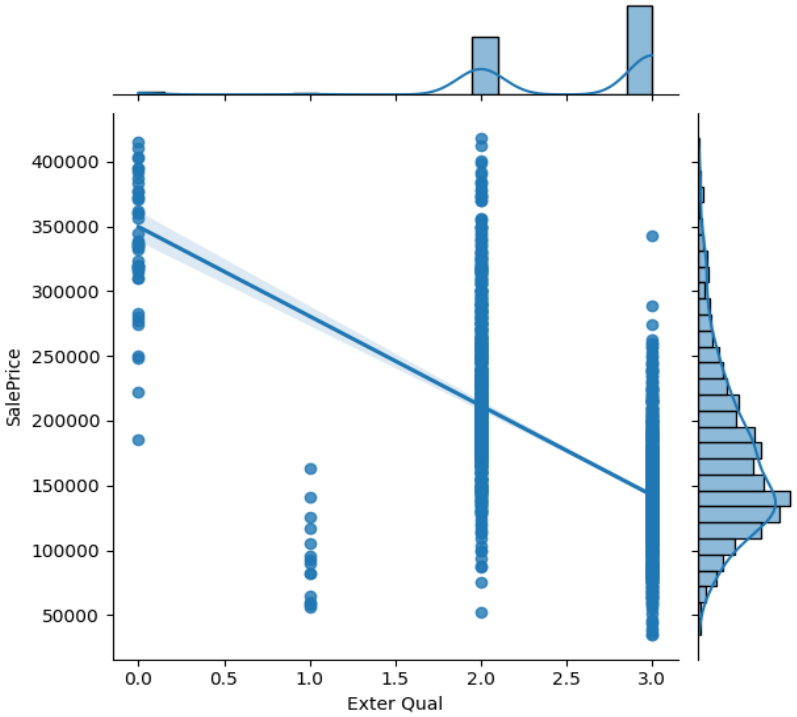
شکل۵: نمودار پراکندگی با خط رگرسیون خطی به همراه توزیع ویژگی Gr Liv Area نسبت به SalePrice



شکل۶: نمودار پراکندگی با خط رگرسیون خطی به همراه توزیع ویژگی Garage Cars نسبت به SalePrice



شکل۷: نمودار پراکندگی با خط رگرسیون خطی به همراه توزیع ویژگی Garage Area نسبت به SalePrice



شکل۸: نمودار پراکندگی با خط رگرسیون خطی به همراه توزیع ویژگی Exter Qual نسبت به SalePrice

- Scatter Plot: رابطه بین ویژگی انتخاب شده از (top_corr_features) و SalePrice را نشان می‌دهد. این به نمایش اینکه آیا همبستگی خطی یا غیرخطی بین دو متغیر وجود دارد کمک می‌کند.
- خط رگرسیون خطی: این خط روند رابطه بین ویژگی و قیمت فروش را نشان می‌دهد. یک شیب مثبت یک همبستگی مثبت را نشان می‌دهد (با افزایش ویژگی، قیمت فروش نیز افزایش می‌یابد)، و یک شیب منفی نشان دهنده یک همبستگی منفی است.
- توزیع‌ها بر روی محورها: هیستوگرام‌های بالا و سمت راست نمودار توزیع هر ویژگی را ارائه می‌دهند. این توزیع‌ها به درک گسترش و گرایش مرکزی ویژگی و SalePrice به صورت جداگانه کمک می‌کنند.

به طور کلی، این نمودارها به شناسایی قدرت، جهت و شکل رابطه بین ویژگی انتخاب شده و متغیر هدف (SalePrice) کمک می‌کنند.

✓ بخش یازدهم کد: پاسخ سوال ۹

```
for name, model in models.items():
    if name == "Polynomial Regression (degree=2)":
        poly = PolynomialFeatures(degree=2)
        X_train_poly = poly.fit_transform(X_train)
        X_test_poly = poly.transform(X_test)
        poly_model = LinearRegression()
        poly_model.fit(X_train_poly, y_train)
        y_pred = poly_model.predict(X_test_poly)
    else:
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)

    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    r2 = r2_score(y_test, y_pred)
    print(f"{name} - RMSE: {rmse}, R²: {r2}")
```

- **محاسبه RMSE و امتیاز R^2**

۱. RMSE

RMSE میانگین بزرگی خطاهای پیش‌بینی را در یک مدل رگرسیونی اندازه‌گیری می‌کند و نشان می‌دهد که مقادیر پیش بینی شده چقدر از مقادیر واقعی انحراف دارند.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

n = تعداد کل مشاهدات

y_i = مقدار (واقعی) متغیر هدف

$\hat{y}_i$ = مقدار پیش بینی شده

$(y_i - \hat{y}_i)^2$ = مربع خطا برای هر مشاهده

مراحل محاسبه RMSE:

- تفاوت بین مقادیر واقعی و پیش بینی شده را محاسبه می‌کند.
- هر تفاوت را Square می کند تا خطاهای بزرگتر جریمه شود.
- میانگین این اختلافات مجذور را پیدا می کند.

- جذر میانگین را برای بدست آوردن RMSE گرفته می شود.

مقادیر کمتر RMSE نشان دهنده عملکرد بهتر مدل است.

۲. امتیاز R^2

امتیاز R^2 اندازه‌گیری می‌کند که مدل رگرسیون چقدر متغیر وابسته را توضیح می‌دهد. عملکرد مدل را با یک مدل پایه ساده مقایسه می‌کند که همیشه میانگین را پیش‌بینی می‌کند.

$$R^2 = 1 - \frac{SSres}{SStot}$$

Ssres: مجذور کل اختلاف بین مقادیر واقعی و پیش‌بینی شده.

Sstot: مجذور کل اختلاف بین مقادیر واقعی و میانگین آنها.

مراحل محاسبه R^2:

- واریانس کل مقادیر واقعی (SStot) را محاسبه می‌شود.
- مجموع مجذور باقیمانده ها محاسبه می‌شود (SSres).
- نسبت آنها محاسبه می‌شود.
- نسبت از ۱ کم می‌شود تا نسبت واریانس توضیح داده شده به‌دست آید.

$R^2 = 1$: مدل کامل (همه نقاط روی خط رگرسیون قرار دارند).

$R^2 = 0$: مدل هیچ واریانسی را در y توضیح نمی‌دهد.

$R^2 < 0$: عملکرد مدل بدتر از پیش‌بینی میانگین است.

پاسخ کد مربوط به این بخش در شکل زیر نشان داده شده است:

```
Linear Regression - RMSE: 24693.815246691534, R²: 0.8720629110932117
Lasso Regression - RMSE: 24693.819430292413, R²: 0.8720628677432669
Ridge Regression - RMSE: 24694.070122346384, R²: 0.8720602700904123
Polynomial Regression (degree=2) - RMSE: 21188.391250410517, R²: 0.9058075609303413
```

شکل۹: پاسخ کد مربوط به محاسبه RMSE و R^2

- **پیاده سازی مدل**

مدل های زیر اجرا شده است:

۱- Linear Regression

- ✓ RMSE: 24693.81
- ✓ امتیاز R^2:0.8721

۲- Lasso Regression

- ✓ پارامتر منظم سازی (آلفا): ۰.۱
- ✓ RMSE:24,693.82
- ✓ امتیاز R^2:0.8721

۳- Ridge Regression

- ✓ پارامتر منظم سازی (آلفا): ۱.۰
- ✓ RMSE:24,694.07
- ✓ امتیاز R^2:0.8721

۴- Polynomial Regression (Degree = 2)

- ✓ بسط ویژگی چند جمله‌ای با درجه ۲ اعمال شد.
- ✓ RMSE:21,188.39
- ✓ امتیاز R^2:0.9058

سطح اطمینان ۹۵٪ برای حذف موارد پرت در نظر گرفته شد.

همه متغیرهای طبقه‌بندی پس از رمزگذاری برچسب، روابط ترتیبی معنی‌داری دارند.

رگرسیون چند جمله‌ای (درجه = ۲) از مدل‌های دیگر با کمترین RMSE و بالاترین امتیاز R^2 بهتر عمل کرد.

❖ پاسخ سوال ۱۰

Bias-Variance Trade-Off in Machine Learning

Bias-Variance Trade-Off یک مفهوم اساسی است که رابطه بین پیچیدگی مدل و خطاهای پیش‌بینی را توضیح می‌دهد. این به ما کمک می‌کند تا هنگام طراحی مدل‌های یادگیری ماشینی، trade-off بین عدم تناسب و overfitting را درک کنیم.

۱. Bias (Underfitting)

- Bias به خطای معرفی شده با تقریب یک مسئله پیچیده دنیای واقعی با یک مدل ساده تر اشاره دارد.
- مدل‌های با Bias بالا مفروضات قوی درباره داده‌ها ایجاد می‌کنند که منجر به تعمیم ضعیف می‌شود.
- نمونه‌هایی از مدل های با Bias بالا: رگرسیون خطی، درخت‌های تصمیم‌گیری ساده.

۲. واریانس (Overfitting)

- واریانس حساسیت مدل را به نوسانات کوچک در داده های آموزشی اندازه‌گیری می‌کند.
- مدل‌های با واریانس بالا بیش از حد انعطاف‌پذیر هستند و به جای روندهای عمومی، نویز را ضبط می‌کنند.
- نمونه‌هایی از مدل‌های با واریانس بالا: شبکه‌های عصبی عمیق، درختان تصمیم گیری بسیار عمیق.
- تطبیق بیش از حد - مدل الگوهایی را می‌آموزد که فقط در مجموعه آموزشی وجود دارند که منجر به تعمیم ضعیف در داده‌های جدید می‌شود.

- چگونه Bias و واریانس بر عملکرد مدل تأثیر می‌گذارد؟

خطای کل یک مدل شامل موارد زیر است:

$$\text{Total Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

balance بهینه پیدا می‌شود که در آن bias و واریانس به حداقل برسد تا به کمترین خطای ممکن دست یابد.

مثال: تاثیر پیچیدگی مدل بر bias و واریانس

رگرسیون چند جمله‌ای را در نظر می‌گیریم.

مثال: پیش بینی قیمت خانه

ما سه مدل رگرسیون چند جمله‌ای را آموزش می‌دهیم:

- ✓ **درجه = ۱ (رگرسیون خطی – عدم تناسب)**
  - bias زیاد، واریانس کم.
  - مدل خیلی ساده است و روابط را به خوبی نشان نمی‌دهد.
  - پیش بینی‌ها به طور مداوم نادرست هستند.

- ✓ **درجه = ۳ (تناسب بهینه)**
  - bias و واریانس متعادل.
  - الگوهای ضروری را بدون overfitting ثبت می‌کند.
  - بهترین تعمیم را ارائه می‌دهد.

- ✓ **درجه = ۱۰ (Overfitting)**
  - bias کم، واریانس بالا.
  - نویز و همچنین الگوها را می‌گیرد.

- در داده‌های آموزشی عملکرد خوبی دارد، اما در داده‌های دیده نشده ضعیف است.

جدول۲: نتایج کلی سه مدل استفاده شده

پیچیدگی مدل	Bias	واریانس	کل خطا	عملکرد
مدل ساده (رگرسیون خطی)	بالا	کم	بالا (Underfitting)	ضعیف
پیچیدگی متوسط (چند جمله ای درجه ۳)	متوسط	متوسط	کم (بهینه)	بهترین
مدل Complex (چند جمله ای درجه ۱۰)	کم	بالا	بالا (overfitting)	ضعیف

بنابراین می‌توان نتیجه‌گیری کرد:

- افزایش پیچیدگی، bias را کاهش می‌دهد؛ اما واریانس را افزایش می‌دهد.
- کاهش پیچیدگی، bias را افزایش می‌دهد؛ اما واریانس را کاهش می‌دهد.
- عملکرد بهینه با متعادل کردن bias و واریانس به دست می‌آید.

❖ گیت هاب

کد مربوط به این تمرین در گیت هاب به آدرس

https://github.com/MM-Touiserkani/AI_HW2_Regression/blob/main/AI_HW2_Touiserkani.ipynb

قرار داده شده است.