هوش مصنوعی

تمرین دوم

مائده اسماعیل‌زاده

۸۱۰۶۰۲۱۶۱

استاد: دکتر شریعت‌پناهی

دانشکده مهندسی مکانیک

پردیس دانشکده‌های فنی دانشگاه تهران

فهرست مطالب

سوال اول

در این تمرین کد مربوط به سوالات در google colab نوشته شده‌اند. برای خواسته این سوال کد زیر نوشته شده است. نتیجه مربوط به این کد نیز نشان می‌دهد فایل به خوبی در محیط آپلود شده است.

```
from google.colab import files
Housing = files.upload()
```

```
Choose Files Housing.csv
• Housing.csv(text/csv) - 963738 bytes, last modified: 3/23/2025 - 100% done
Saving Housing.csv to Housing (1).csv
```

در مرحله بعد برای خواندن اطلاعات موجود در فایل مربوطه، کد زیر نوشته شده است.

```
import pandas as pd

Housing1 = pd.read_csv('Housing.csv')
Housing1.info()
```

نتیجه این کد در زیر قابل مشاهده است.

```
<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
 20  Year Built       2930 non-null   int64
 21  Year Remod/Add   2930 non-null   int64
 22  Roof Style       2930 non-null   object
 23  Roof Matl        2930 non-null   object
 24  Exterior 1st     2930 non-null   object
 25  Exterior 2nd     2930 non-null   object
 26  Mas Vnr Type     1155 non-null   object
 27  Mas Vnr Area     2907 non-null   float64
 28  Exter Qual       2930 non-null   object
 29  Exter Cond       2930 non-null   object
 30  Foundation       2930 non-null   object
 31  Bsmt Qual        2850 non-null   object
 32  Bsmt Cond        2850 non-null   object
 33  Bsmt Exposure    2847 non-null   object
 34  BsmtFin Type 1   2850 non-null   object
 35  BsmtFin SF 1     2929 non-null   float64
 36  BsmtFin Type 2   2849 non-null   object
 37  BsmtFin SF 2     2929 non-null   float64
 38  Bsmt Unf SF      2929 non-null   float64
 39  Total Bsmt SF    2929 non-null   float64
 40  Heating          2930 non-null   object
 41  Heating QC       2930 non-null   object
 42  Central Air      2930 non-null   object
 43  Electrical       2929 non-null   object
 44  1st Flr SF       2930 non-null   int64
 45  2nd Flr SF       2930 non-null   int64
 46  Low Qual Fin SF  2930 non-null   int64
 47  Gr Liv Area      2930 non-null   int64
 48  Bsmt Full Bath   2928 non-null   float64
 49  Bsmt Half Bath   2928 non-null   float64
 50  Full Bath        2930 non-null   int64
```

```
51  Half Bath        2930 non-null   int64
52  Bedroom AbvGr    2930 non-null   int64
53  Kitchen AbvGr    2930 non-null   int64
54  Kitchen Qual     2930 non-null   object
55  TotRms AbvGrd    2930 non-null   int64
56  Functional       2930 non-null   object
57  Fireplaces       2930 non-null   int64
58  Fireplace Qu     1508 non-null   object
59  Garage Type      2773 non-null   object
60  Garage Yr Blt    2771 non-null   float64
61  Garage Finish    2771 non-null   object
62  Garage Cars      2929 non-null   float64
63  Garage Area      2929 non-null   float64
64  Garage Qual      2771 non-null   object
65  Garage Cond      2771 non-null   object
66  Paved Drive      2930 non-null   object
67  Wood Deck SF     2930 non-null   int64
68  Open Porch SF    2930 non-null   int64
69  Enclosed Porch   2930 non-null   int64
70  3Ssn Porch       2930 non-null   int64
71  Screen Porch     2930 non-null   int64
72  Pool Area        2930 non-null   int64
73  Pool QC          13 non-null     object
74  Fence            572 non-null    object
75  Misc Feature     106 non-null    object
76  Misc Val         2930 non-null   int64
77  Mo Sold          2930 non-null   int64
78  Yr Sold          2930 non-null   int64
 79  Sale Type        2930 non-null   object
 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
```

همانطور که از نتایج قابل مشاهده است، ۸۲ ستون(ویژگی) و ۲۹۳۰ ردیف وجود دارد. همچنین قابل مشاهده است که تعدادی از ستون‌ها مقادیر گمشده دارند (مثلاً Alley، Fireplace Qu، Pool QC، Fence و ...). همچنین مشخص است هر ستون شامل اعداد می‌باشد یا محتویات غیرعددی دارند. ستون هدف برای پیش‌بینی، SalePrice می‌باشد که کامل بوده و نوعش int64 می‌باشد.

سوال دوم

در این بخش از کد زیر استفاده شده است. در ابتدا تابعی برای حذف داده‌های پرت بر اساس IQR تعریف شده است. سپس سه ستون اصلی انتخاب شده و داده‌های پرت حذف شده‌اند. این انتخاب باعث میشود داده‌ها خیلی زیاد حذف نشده ولی در عین حال بخش بزرگی از داده‌های پرت حذف شوند. لزومی به پاکسازی تمام ستون‌ها نمی‌باشد؛ زیرا برخی ستون‌ها (مثلاً سال ساخت یا تعداد حمام) مقدارهای محدودی دارند در نتیجه پرت بودن داده‌ها تأثیر کمی دارد. همچنین اگر برای همه ستون‌ها داده‌های پرت حذف شوند، ممکن است حجم زیادی از داده حتی برخی داده‌های مفید پاکسازی شده، نمونه‌ها بیش‌ازحد محدود شده و در بخش‌های بعد مدل‌ها overfit شوند. همچنین یک ستون انتخاب نشده زیرا انتخاب فقط یک ستون برای پاکسازی از داده‌های پرت می‌تواند منطقی باشد، اما حذف داده‌های پرت در یک ستون خاص ممکن است باعث از دست دادن داده‌های مهم از سایر ویژگی‌ها نیز شود. به عبارت دیگر، داده‌های پرت ممکن است در ستون‌های مختلف وجود داشته باشند، و حذف فقط یک ستون به تنهایی ممکن است تصویری کامل از داده‌های پرت ندهد. علت انتخاب سه پارامتری که در کد زیر آمده است به شرح زیر می‌باشد: لازم به ذکر می‌باشد این پارامترها بصورت تحلیلی انتخاب شده‌اند.

- SalePrice: زیرا این متغیر هدف ما در این تمرین می‌باشد. داده‌های پرت در این ستون می‌توانند هم باعث ایجاد مشکل در مدل شده و هم در تحلیل‌های آماری مثل همبستگی و رگرسیون باعث انحراف شوند.
- Gr Liv Area: متغیر مساحت فضای زندگی یکی از مهم‌ترین متغیرها در پیش‌بینی قیمت خانه می‌باشد. در نتیجه حذف داده‌های پرت مهم می‌باشد.
- Overall Qual: متغیر کیفیت کلی خانه نیز در تحلیل قیمت تأثیرگذار می‌باشد. در نتیجه حذف داده‌های پرت مهم می‌باشد.

در نتیجه بطور کلی می‌توان گفت این سه پارامتر تأثیر مستقیم روی قیمت خانه داشته، پتانسیل بالایی برای داشتن داده‌های پرت داشته و بهتر است حتما پاکسازی شوند. به عبارت دیگر می‌توان گفت این سه پارامتر بصورت تحلیلی انتخاب شده‌اند.

```python
import numpy as np
import pandas as pd

def hazf_outliers(Housing, Sutun):
    Q1 = Housing[Sutun].quantile(0.25)
    Q3 = Housing[Sutun].quantile(0.75)
    IQR = Q3 - Q1
    bound_L = Q1 - 1.5 * IQR
    bound_H = Q3 + 1.5 * IQR
    return (Housing[Sutun] >= bound_L) & (Housing[Sutun] <= bound_H)


Imp_columns = ['SalePrice', 'Gr Liv Area', 'Overall Qual']

# Ba in mask har 3 sutun ba ham barresi mishavand
mask = np.ones(len(Housing1), dtype=bool)

for col in Imp_columns:
    mask &= hazf_outliers(Housing1, col)

Housing1 = Housing1[mask]

print("Tedade dadeha pas az hazfe outliers", Housing1.shape[0])
```

```
Tedade dadeha pas az hazfe outliers 2753
```

همانطور که از نتیجه مشخص می‌باشد، تعداد داده‌ها پس از حذف ۲۷۵۳ می‌باشد که عدد معقولی است.

در مرحله بعد در ابتدا تعداد مقادیر ناموجود در هر ستون شمارش شده است. سپس برای داده‌های عددی این مقادیر با میانگین جایگزین شده است. برای داده‌های غیر عددی نیز با مد جایگزین شده‌اند. در انتها تعداد مقادیر ناموجود دوباره محاسبه شده‌اند که مشاهده می‌شود هیچ مقدار ناموجودی وجود ندارد.

```python
# Shomareshe missing values ghabl az jaygozini
missing_values = Housing1.isnull().sum()
print("Tedad maghadire gomshode:")
print(missing_values[missing_values > 0])

# Jaygozinie missing values baraye sutunhaye adadi ba miangin
numeric_cols = Housing1.select_dtypes(include=['float64', 'int64']).columns
for col in numeric_cols:
    Housing1[col] = Housing1[col].fillna(Housing1[col].mean())

# Jaygozinie missing values baraye sutunhaye gheyre adadi ba mode
non_numeric_cols = Housing1.select_dtypes(include=['object']).columns
for col in non_numeric_cols:
    Housing1[col] = Housing1[col].fillna(Housing1[col].mode()[0])

# Dadeha bad az jaygozinie missing values
print("Tedad maghadire gomshode bad az jaygozini:")
Housing3 = Housing1.isnull().sum()
print(Housing3[Housing3 > 0])
```

```
Tedad maghadire gomshode:
Lot Frontage       471
Alley             2559
Mas Vnr Type      1735
Mas Vnr Area        21
Bsmt Qual           76
Bsmt Cond           76
Bsmt Exposure       79
BsmtFin Type 1      76
BsmtFin SF 1         1
BsmtFin Type 2      77
BsmtFin SF 2         1
Bsmt Unf SF          1
Total Bsmt SF        1
Electrical           1
Bsmt Full Bath       1
Bsmt Half Bath       1
Fireplace Qu      1413
Garage Type        153
Garage Yr Blt      155
Garage Finish      155
Garage Cars          1
Garage Area          1
Garage Qual        155
Garage Cond        155
Pool QC           2746
Fence             2194
Misc Feature      2650
dtype: int64
Tedad maghadire gomshode bad az jaygozini:
Series([], dtype: int64)
```

سوال سوم

در این سوال ابتدا ستون‌های عددی استخراج شده‌اند. سپس مقادیر کمینه، بیشینه و انحراف از معیار محاسبه شده‌اند. در نهایت پس از اعمال چند کار کوتاه برای بهتر نمایش دادن نتایج، جدول بصورت زیر بدست می‌آید.

```
# Sutunhaye adadi
numeric_Housing = Housing1.select_dtypes(include=['int64', 'float64'])

# Estekhraje min, max, enheraf az meyar
Vizhegi_amari = numeric_Housing.describe().loc[['min', 'max', 'std']]

Vizhegi_amari = Vizhegi_amari.T
Vizhegi_amari.columns = ['Min', 'Max', 'Enheraf az meyar']

print(Vizhegi_amari)
```

	Min	Max	Enheraf az meyar
Order	1.0	2.930000e+03	8.495737e+02
PID	526301100.0	1.007100e+09	1.886639e+08
MS SubClass	20.0	1.900000e+02	4.322874e+01
Lot Frontage	21.0	3.130000e+02	2.188408e+01
Lot Area	1300.0	1.646600e+05	6.660146e+03
Overall Qual	2.0	1.000000e+01	1.276573e+00
Overall Cond	1.0	9.000000e+00	1.114003e+00
Year Built	1872.0	2.010000e+03	2.999233e+01
Year Remod/Add	1950.0	2.010000e+03	2.089510e+01
Mas Vnr Area	0.0	1.600000e+03	1.516743e+02
BsmtFin SF 1	0.0	1.880000e+03	4.015760e+02
BsmtFin SF 2	0.0	1.526000e+03	1.650453e+02
Bsmt Unf SF	0.0	2.042000e+03	4.284791e+02
Total Bsmt SF	0.0	3.206000e+03	3.818745e+02
1st Flr SF	372.0	2.524000e+03	3.345000e+02
2nd Flr SF	0.0	1.788000e+03	4.006178e+02
Low Qual Fin SF	0.0	1.064000e+03	4.332604e+01
Gr Liv Area	407.0	2.654000e+03	4.151733e+02
Bsmt Full Bath	0.0	3.000000e+00	5.158379e-01
Bsmt Half Bath	0.0	2.000000e+00	2.473962e-01
Full Bath	0.0	4.000000e+00	5.309793e-01
Half Bath	0.0	2.000000e+00	4.950978e-01
Bedroom AbvGr	0.0	6.000000e+00	7.979964e-01
Kitchen AbvGr	0.0	3.000000e+00	2.175388e-01
TotRms AbvGrd	3.0	1.300000e+01	1.437679e+00
Fireplaces	0.0	4.000000e+00	6.296366e-01
Garage Yr Blt	1895.0	2.207000e+03	2.548672e+01
Garage Cars	0.0	5.000000e+00	7.248086e-01
Garage Area	0.0	1.488000e+03	1.994899e+02
Wood Deck SF	0.0	1.424000e+03	1.222507e+02
Open Porch SF	0.0	7.420000e+02	6.387875e+01
Enclosed Porch	0.0	1.012000e+03	6.425790e+01
3Ssn Porch	0.0	5.080000e+02	2.512134e+01
Screen Porch	0.0	5.760000e+02	5.347092e+01
Pool Area	0.0	8.000000e+02	3.060316e+01
Misc Val	0.0	1.550000e+04	4.847543e+02
Mo Sold	1.0	1.200000e+01	2.702382e+00
Yr Sold	2006.0	2.010000e+03	1.321793e+00
SalePrice	12789.0	3.389310e+05	5.790344e+04

با توجه به نتایج بعنوان مثال می‌توان گفت ستون SalePrice که مربوط به قیمت فروش خانه‌ها می‌باشد، کمینه‌ای برابر با ۱۲۷۸۹ و بیشینه‌ای برابر با ۳۳۸۹۳۱ دارد، که نشان‌دهنده تفاوت بزرگ در قیمت‌ها است. می‌توان گفت قیمت فروش خانه‌ها گستردگی زیادی دارد و انحراف از معیار بالا نشان‌دهنده تفاوت زیاد در قیمت‌ها است. همچنین ستون‌هایی مانند Pool Area، Fireplaces، Half Bath که اغلب مقادیر صفر دارند (که احتمالاً به این معنی است که بسیاری از خانه‌ها ویژگی‌های خاصی مثل استخر یا شومینه ندارند) نیز در بین ویژگی‌هایی هستند که پراکندگی کمی دارند. همچنین قابل مشاهده می‌باشد ویژگی کیفیت کلی خانه

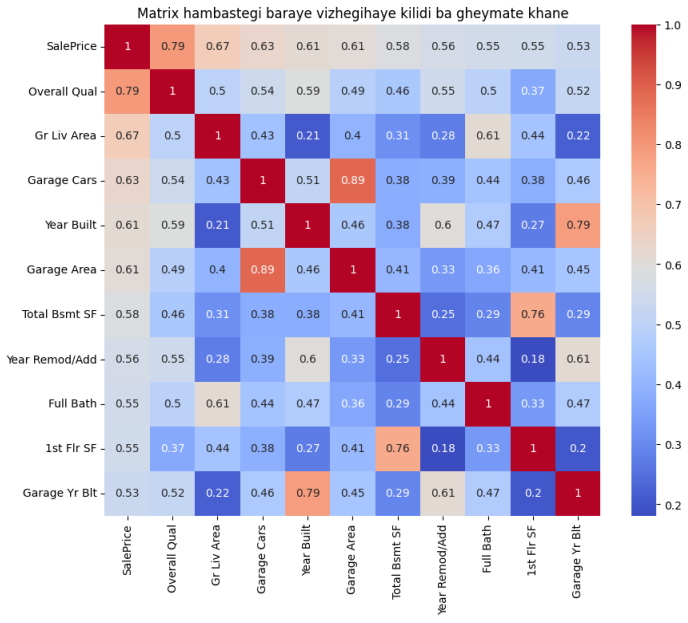
(Overall Qual) انحراف از معیار نسبتا پایینی دارد؛ نشان‌دهنده این است که بیشتر خانه‌ها در سطح کیفیت مشابهی قرار دارند. بنابراین بطور کلی می‌توان گفت ویژگی‌هایی مانند SalePrice, Lot Area و Garage Area دارای انحراف از معیار بالایی هستند که این نشان‌دهنده تنوع بالا در داده‌ها و تفاوت‌های زیاد در ویژگی‌ها است. همچنین با مشاهده مقادیر گمشده یا صفر برای ویژگی‌هایی مانند BsmtFin SF 1, Garage Area, Pool Area می‌توان گفت این ویژگی‌ها باید با دقت بیشتری برای مدل‌سازی پیش‌بینی شوند.

سوال چهارم

در این بخش از کد زیر استفاده شده است. نتایج نیز قابل مشاهده می‌باشند. در این بخش ماتریس همبستگی با قیمت خانه فقط برای ۹ ویژگی رسم شده است. همانطور که از نتایج قابل مشاهده می‌باشد سه ویژگی با همبستگی بالا شامل *Overall Qual, Gr liv area, Garage cars* می‌باشند.

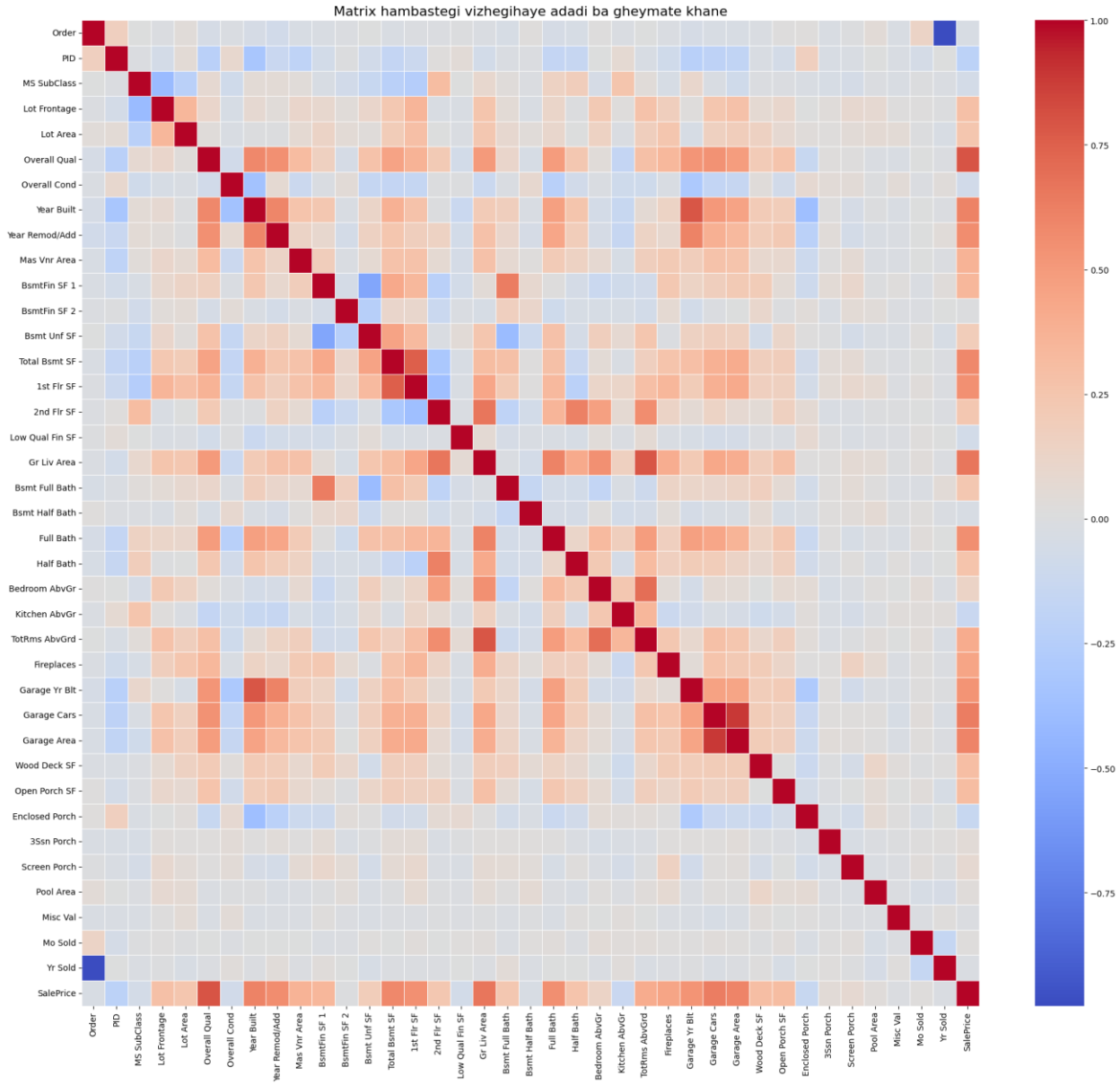
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Matrix hambastegi baraye sutunhaye adadi
Matrix_hambastegi = Housing1.corr(numeric_only=True)

# Hambastegi ba gheymat
Hambastegi_ba_gheymat = Matrix_hambastegi['SalePrice'].sort_values(ascending=False)

Hambastegi_bala = Hambastegi_ba_gheymat[(Hambastegi_ba_gheymat > 0.5) | (Hambastegi_ba_gheymat < -0.5)]

print("Vizhegihaye ba hambastegie bishtar ba gheymate khane")
print(Hambastegi_bala)

# Nemudar baraye 10 vizhegi
vizhegihaye_kilidi = Hambastegi_bala.index
plt.figure(figsize=(10, 8))
sns.heatmap(Housing1[vizhegihaye_kilidi].corr(), annot=True, cmap='coolwarm')
plt.title('Matrix hambastegi baraye vizhegihaye kilidi ba gheymate khane')
plt.show()
```

```
Vizhegihaye ba hambastegie bishtar ba gheymate khane
SalePrice         1.000000
Overall Qual      0.792924
Gr Liv Area       0.665150
Garage Cars       0.631674
Year Built        0.610261
Garage Area       0.605120
Total Bsmt SF     0.583202
Year Remod/Add    0.564671
Full Bath         0.554965
1st Flr SF        0.549797
Garage Yr Blt     0.533003
Name: SalePrice, dtype: float64
```



در ادامه ماتریس همبستگی برای تمام داده‌های عددی با استفاده از کد زیر رسم شده است.

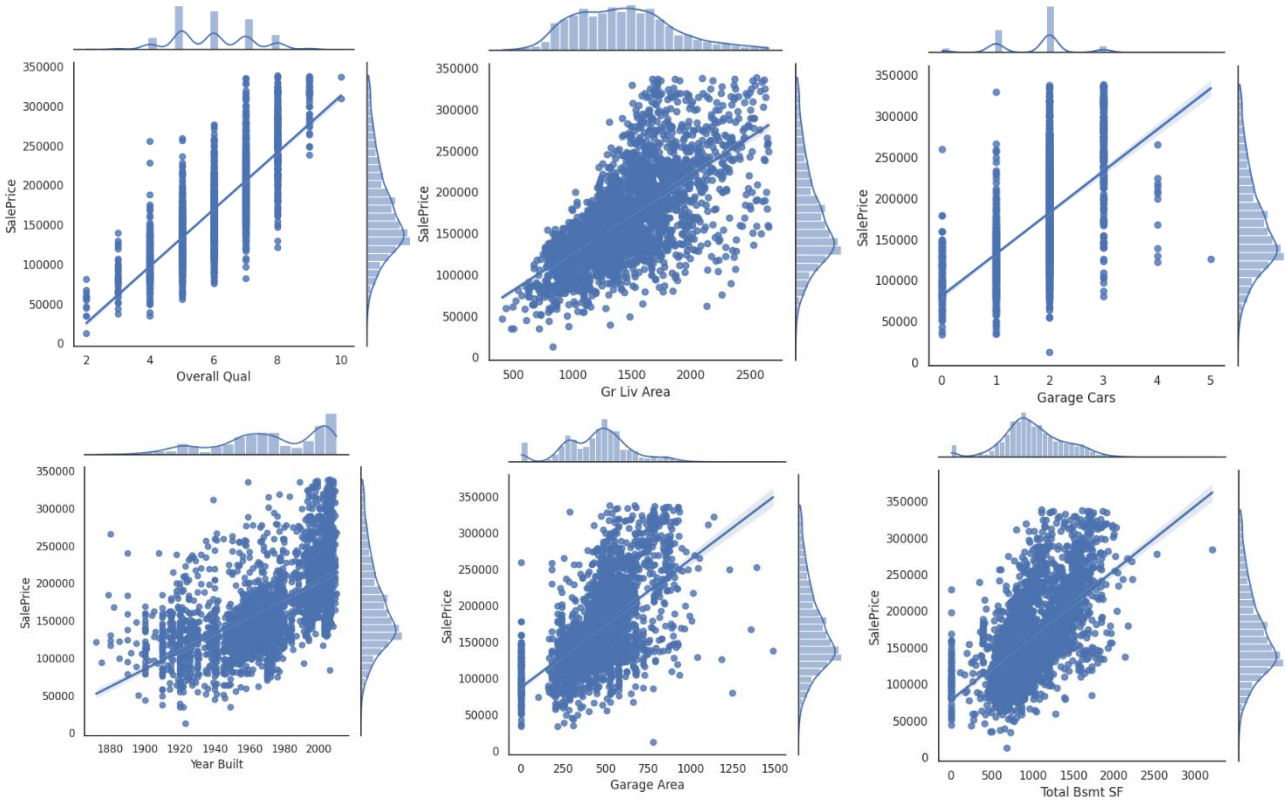
```
[13] import seaborn as sns
     import matplotlib.pyplot as plt

     # Matrix hambastegi baraye dadehaye adadi
     Matrix_hambastegi2 = Housing1.corr(numeric_only=True)

     plt.figure(figsize=(20, 18))
     sns.heatmap(Matrix_hambastegi2, annot=False, cmap='coolwarm', linewidths=0.5)
     plt.title('Matrix hambastegi vizhegihaye adadi ba gheymate khane', fontsize=16)
     plt.xticks(rotation=90)
     plt.yticks(rotation=0)
     plt.tight_layout()
     plt.show()
```



سوال پنجم

در این بخش از کد زیر استفاده شده است. نمودار خواسته شده برای ۹ ویژگی رسم شده است. ۶ نمودار در زیر آمده‌اند.

```
sns.set(style="white", color_codes=True)

Vizhegi_hambastegi_bala = ['Overall Qual', 'Year Built', 'Gr Liv Area',
                           'Garage Cars','Year Remod/Add','Full Bath',
                           'Garage Area','Garage Yr Blt','Total Bsmt SF']

for vizhegi in Vizhegi_hambastegi_bala:
    sns.jointplot(data=Housing1, x=vizhegi, y='SalePrice', kind='reg', height=6)
    plt.suptitle(f"Jointplot: {vizhegi} vs SalePrice", y=1.02)
    plt.tight_layout()
    plt.show()
```



در نمودار مربوط به کیفیت کلی خانه مشاهده می‌شود یک رابطه‌ی خطی و مستقیم قوی بین کیفیت کلی (Overall Qual) و قیمت فروش (SalePrice) دیده می‌شود. مشاهده می‌شود هرچه کیفیت کلی بالاتر باشد، قیمت خانه هم به طور قابل توجهی افزایش پیدا می‌کند. همچنین می‌توان گفت نقاط اطراف خط رگرسیون نسبتاً نزدیک‌اند، که نشانه‌ی همبستگی بالا بین این دو متغیر است. بنابراین می‌توان گفت Overall Qualیکی از تاثیرگذارترین ویژگی‌ها بر SalePrice است.

در نمودار دوم، مشاهده می‌شود بین مساحت زیربنای قابل استفاده (Gr Liv Area) و قیمت خانه هم رابطه‌ی مستقیم و قوی وجود دارد. نقاط زیادی روی خط رگرسیون یا در نزدیکی آن قرار دارند، که نشان‌دهنده‌ی همبستگی بالا است. همچنین با افزایش متراژ، قیمت به شکل پیوسته بالا می‌رود. این ویژگی هم یکی از مهم‌ترین فاکتورها برای پیش‌بینی قیمت خانه است.

در نمودار چهارم رابطه بین سال ساخت و قیمت خانه هم مثبت است ولی نسبت به کیفیت کلی، پراکندگی بیشتری در داده‌ها دیده می‌شود. هرچه خانه جدیدتر باشد (سال ساخت بالاتر)، تمایل به قیمت بیشتر دارد، اما با نوسانات زیاد. نقاط داده‌ها به طور یکنواخت در اطراف خط رگرسیون پراکنده نیستند؛ این موضوع نشان می‌دهد که Year Built تاثیر دارد، اما نه به اندازه‌ی Overall Qual.

سوال ششم

در این سوال از کد زیر استفاده شده است.

```
[11] from sklearn.feature_selection import SelectKBest, f_regression
     from sklearn.linear_model import LinearRegression
     from sklearn.model_selection import cross_val_score

     # Joda kardane vizhegiha va hadaf
     vizhegiha = Housing1.drop(columns=['SalePrice'])
     vizhegi_adadi = vizhegiha.select_dtypes(include=[np.number])
     hadaf = Housing1['SalePrice']

     Behtarin = -np.inf
     Behtarin_k = 0
     scores = []

     for k in range(1, vizhegi_adadi.shape[1] + 1):
         selector = SelectKBest(score_func=f_regression, k=k)
         vizhegi_entekhabi = selector.fit_transform(vizhegi_adadi, hadaf)

         # Arzyabie deghate model(cross validation)
         model = LinearRegression()
         score = cross_val_score(model, vizhegi_entekhabi, hadaf, cv=5, scoring='r2').mean()
         scores.append(score)
         if score > Behtarin:
             Behtarin = score
             best_k = k

     # Behtarin tedade vizhegiha = k
     # Bishtarin deghat (R^2)
     print(f"k: {best_k}")
     print(f"R^2: {Behtarin:.4f}")
```

```
k: 30
R^2: 0.8734
```

نتیجه بدست آمده نشان می‌دهد ۳۰ ویژگی با بیشترین تأثیر بر پیش‌بینی قیمت خانه انتخاب شده‌اند و مدل رگرسیون تعریفی پس از این انتخاب توانسته است تا ۸۷.۳۴٪ از واریانس داده‌ها را توضیح دهد که دقت خوبی می‌باشد. این انتخاب ویژگی‌ها و ارزیابی به روش درست انجام شده است و مدل تعریفی توانسته است پیش‌بینی‌های دقیقی برای قیمت خانه‌ها ارائه دهد. در ادامه از کد زیر استفاده شده است تا ۳۰ ویژگی انتخاب شده نشان داده شوند.

```
# Amuzesh ba behtarin k az bakhshe ghabl
selector = SelectKBest(score_func=f_regression, k=best_k)
selector.fit(vizhegi_adadi, hadaf)

# Vizhegihaye entekhab shode
vizhegiha_entekhabi = vizhegi_adadi.columns[selector.get_support()]
print("Vizhegihaye entekhab shode:")
print(vizhegiha_entekhabi)
```

```
Vizhegihaye entekhab shode:
Index(['PID', 'MS SubClass', 'Lot Frontage', 'Lot Area', 'Overall Qual',
       'Overall Cond', 'Year Built', 'Year Remod/Add', 'Mas Vnr Area',
       'BsmtFin SF 1', 'Bsmt Unf SF', 'Total Bsmt SF', '1st Flr SF',
       '2nd Flr SF', 'Low Qual Fin SF', 'Gr Liv Area', 'Bsmt Full Bath',
       'Full Bath', 'Half Bath', 'Bedroom AbvGr', 'Kitchen AbvGr',
       'TotRms AbvGrd', 'Fireplaces', 'Garage Yr Blt', 'Garage Cars',
       'Garage Area', 'Wood Deck SF', 'Open Porch SF', 'Enclosed Porch',
       'Screen Porch'],
      dtype='object')
```

سوال هفتم

در این بخش از کد زیر استفاده شده است. همانطور که مشخص است تعداد داده ها پس از پاکسازی ۲۷۵۳ بود و از نتیجه این بخش واضح است داده‌ها به خوبی تقسیم شده‌اند.

```
[13] from sklearn.model_selection import train_test_split

     # Joda kardane vizhegiha va hadaf bad az taeine vizhegihaye entekhabi
     X = vizhegi_adadi[vizhegiha_entekhabi]
     y = hadaf

     # Taghsime dadeha be test va train
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
                                                         random_state=42)

     # Andazeha
     print(f"Tedad nemunehaye amuzesh: {X_train.shape[0]}")
     print(f"Tedad nemunehaye azmun: {X_test.shape[0]}")
```

```
Tedad nemunehaye amuzesh: 2064
Tedad nemunehaye azmun: 689
```

سوال هشتم و نهم

کد و نتایج مربوط به سوال ۸ و ۹ بصورت زیر می‌باشد. لازم به ذکر می‌باشد در ابتدا از مقدارهای پیش‌فرض یا ساده برای تست اولیه برای آلفاهای (۱)Ridge regression و Lasso regression(۰.۱) و درجه (دوم) Polynomial regression استفاده شد. نتایج استفاده از این کد بصورت زیر بدست آمد.

```
[36] from sklearn.linear_model import LinearRegression, Ridge, Lasso
     from sklearn.preprocessing import PolynomialFeatures
     from sklearn.pipeline import make_pipeline
     from sklearn.metrics import mean_squared_error, r2_score

     # Tarife modelha
     models = {'Linear Regression': LinearRegression(),
         'Ridge Regression': Ridge(alpha=1.0),
         'Lasso Regression': Lasso(alpha=0.1, max_iter=10000),
         'Polynomial Regression (degree=2)': make_pipeline(PolynomialFeatures(degree=2),
                                                           LinearRegression())}

     for name, model in models.items():
         model.fit(X_train, y_train) # Amuzeshe model
         y2 = model.predict(X_test) # Pishbini ruye dadeye test
         R2 = r2_score(y_test, y2)
         RMSE = np.sqrt(mean_squared_error(y_test, y2))
         print(f"{name}")
         print(f"R²: {R2:.4f}")
         print(f"RMSE: {RMSE:.2f}\n")
```

```
Linear Regression
R²: 0.8836
RMSE: 20733.90

Ridge Regression
R²: 0.8836
RMSE: 20731.56

Lasso Regression
R²: 0.8836
RMSE: 20733.85

Polynomial Regression (degree=2)
R²: 0.8783
RMSE: 21199.49
```

حال برای یافتن آلفاها و درجه بهینه از روش‌هایی مثل Cross Validation (در اینجا GridSearchCV) استفاده شده است. کد این بخش بصورت زیر نوشته شده است.

```
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error, r2_score

#Tarife modelha
models = {'Linear Regression': LinearRegression()}

# Entekhabe alpha monaseb baraye Ridge regression
alpha_R = {'alpha': [0.01, 0.1, 1, 10, 100]}
alpha_R1 = GridSearchCV(Ridge(), alpha_R, cv=5, scoring='r2')
alpha_R1.fit(X_train, y_train)
models['Ridge Regression'] = alpha_R1.best_estimator_

# Entekhabe alpha monaseb baraye Ridge regression
alpha_L = {'alpha': [0.0001, 0.001, 0.01, 0.1, 1]}
alpha_L1 = GridSearchCV(Lasso(max_iter=10000), alpha_L, cv=5, scoring='r2')
alpha_L1.fit(X_train, y_train)
models['Lasso Regression'] = alpha_L1.best_estimator_

# Entekhabe daraje monaseb baraye Polynomial regression
degree = {'polynomialfeatures__degree': [2, 3]}
poly_model = make_pipeline(PolynomialFeatures(), LinearRegression())
degree_poly = GridSearchCV(poly_model, degree, cv=5, scoring='r2')
degree_poly.fit(X_train, y_train)
```

```python
models['Polynomial Regression'] = degree_poly.best_estimator_


for name, model in models.items():
    model.fit(X_train, y_train)  # Amuzeshe model
    y_pred = model.predict(X_test) # Pishbini ruye dadeye test
    R2 = r2_score(y_test, y_pred)
    RMSE = np.sqrt(mean_squared_error(y_test, y_pred))
    print(f"{name}")
    print(f"R²: {R2:.4f}")
    print(f"RMSE: {RMSE:.2f}\n")

print("Best alpha for Ridge:", alpha_R1.best_params_['alpha'])
print("Best alpha for Lasso:", alpha_L1.best_params_['alpha'])
print("Best degree for Polynomial:", degree_poly.best_params_['polynomialfeatures__degree'])
```

```
Linear Regression
R²: 0.8836
RMSE: 20733.90

Ridge Regression
R²: 0.8838
RMSE: 20716.00

Lasso Regression
R²: 0.8836
RMSE: 20733.38

Polynomial Regression
R²: 0.8783
RMSE: 21199.49

Best alpha for Ridge: 10
Best alpha for Lasso: 1
Best degree for Polynomial: 2
```

همانطور که مشاهده می‌شود مدل Ridge regression با آلفای ۱۰ بهتر عمل کرده و دقت بالاتری دارد. بقیه مدل‌ها مانند قبل عمل کرده‌اند زیرا همه چیز ثابت مانده است. در کل آلفای بهینه برای مدل Ridge regression برابر ۱۰ و آلفای بهینه برای مدل Lasso regression برابر ۱ و همچنین درجه ۲ برای Polynomial regression مناسب می‌باشند. بطور کلی می‌توان گفت مدل Linear regression دارای دقت بالا ولی حساس به وابستگی ویژگی‌ها می‌باشد. مدل Ridge regression بهترین عملکرد را داشته، مدل Lasso regression عملکرد مشابه رگرسیون خطی، ولی با قابلیت انتخاب ویژگی‌ها داشته و در نهایت Polynomial regression کمی ضعیف‌تر عمل کرده است؛ که ممکن است بعلت overfitting باشد.

معیار R^2 نشان می‌دهد چه درصدی از تغییرات متغیر هدف (قیمت خانه) توسط مدل توضیح داده می‌شود. اگر R^2 نزدیک به ۱ باشد مدل خوب است. اگر R^2 نزدیک به ۰ باشد مدل خوب نیست. در پایتون برای محاسبه این معیار از دستوری استفاده می‌شود که در بالا آمده است. فرمول این معیار بصورت زیر محاسبه می‌شود:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

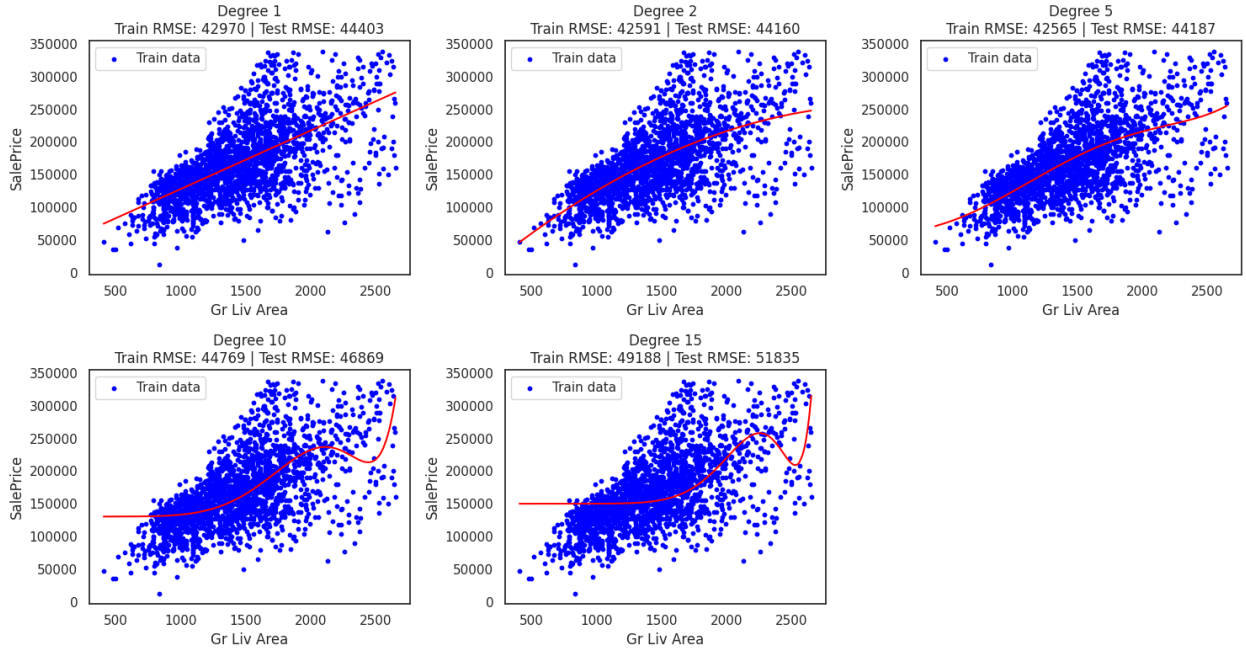
خطای RMS معیاری برای اندازه‌گیری میانگین فاصله پیش‌بینی‌ها از مقادیر واقعی می‌باشد. هرچه RMSE کمتر باشد، مدل دقیق‌تر است. فرمول این خطا بصورت زیر محاسبه می‌شود:

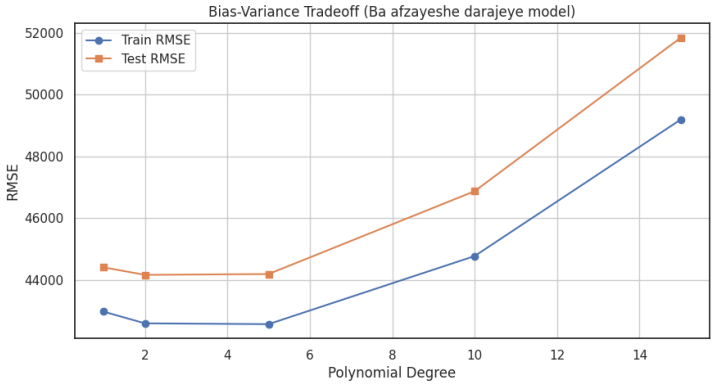
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

سوال دهم

برای این سوال از کد زیر استفاده شده است. لازم به ذکر می‌باشد وقتی یک مدل پیچیده‌تر باشد بایاس کاهش پیدا کرده (یادگیری بهتر الگوها توسط مدل) ولی واریانس افزایش پیدا می‌کند (مدل به داده‌های آموزشی حساس‌تر میشود)، به همین ترتیب مدل ساده‌تر دارای واریانس پایین‌تر ولی بایاس بالا می‌باشد. مدلی مناسب می‌باشد که تعادل خوبی بین این دو مشخصه برقرار کند.

```python
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.model_selection import train_test_split

# Yek vizhegi va hadaf
X1 = Housing1[['Gr Liv Area']].values
y1 = Housing1['SalePrice'].values

X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1, test_size=0.2, random_state=42)

# 5 darajeye mokhtalef
degrees = [1, 2, 5, 10, 15]
train_errors = []
test_errors = []

plt.figure(figsize=(15, 8))

for idx, degree in enumerate(degrees, start=1):

    model = make_pipeline(
        PolynomialFeatures(degree=degree, include_bias=False),
        LinearRegression())
    model.fit(X1_train, y1_train)
    y2_train = model.predict(X1_train)
    y2_test = model.predict(X1_test)

    train_rmse = np.sqrt(mean_squared_error(y1_train, y2_train))
    test_rmse = np.sqrt(mean_squared_error(y1_test, y2_test))
    train_errors.append(train_rmse)
    test_errors.append(test_rmse)

# Rasme nemudar
    plt.subplot(2, 3, idx)
    plt.scatter(X1_train, y1_train, color='blue', s=10, label='Train data')
    plt.plot(np.sort(X1_train, axis=0), model.predict(np.sort(X1_train, axis=0)), color='red')
    plt.title(f'Degree {degree}\nTrain RMSE: {train_rmse:.0f} | Test RMSE: {test_rmse:.0f}')
    plt.xlabel('Gr Liv Area')
    plt.ylabel('SalePrice')
    plt.legend()

plt.tight_layout()
plt.show()

plt.figure(figsize=(10, 5))
plt.plot(degrees, train_errors, label='Train RMSE', marker='o')
plt.plot(degrees, test_errors, label='Test RMSE', marker='s')
plt.xlabel('Polynomial Degree')
plt.ylabel('RMSE')
plt.title('Bias-Variance Tradeoff (Ba afzayeshe darajeye model)')
plt.legend()
plt.grid(True)
plt.show()
```





همانطور که مشخص است خط قرمز در هر نمودار، منحنی تابع پیش‌بینی‌شده توسط مدل پلی‌نومیال آموزش‌دیده است. یعنی پس از این که برای یک درجه مشخص مدل را روی داده‌های آموزشی فیت کردیم، خروجی مدل روی ورودی‌های مرتب‌شده را با یک خط پیوسته رسم شده تا رابطه یادگرفته‌شده بین «Gr Liv Area» بدست آید. با توجه به نتایج قابل مشاهده می‌باشد برای حالت درجه ۱ یک خط صاف قرمز روی نمودار می‌باشد. با توجه به خطای RMS داده‌های تست وآموزش و نمودار، بایاس نسبتاً بالاست (مدل ساده است) و واریانس پایینی دارد (خط صاف). برای حالت درجه ۲ منحنی قرمز کم‌عمق و صافی رسم شده است و خطای RMS داده‌های تست وآموزش در نمودار قابل مشاهده است. مشاهده می‌شود هم بایاس کاهش یافته (مدل کمی پیچیده‌تر شده) و هم واریانس هنوز خیلی بالا نرفته. برای درجه‌های بالاتر مشاهده می‌شود هر چه درجه بیشتر شود منحنی قرمز ، انحنای بیشتری پیدا می‌کند.

بطور کلی می‌توان گفت مدل ساده (بایاس بالا و واریانس پایین)خط صاف، توانایی یادگیری الگوهای پیچیده را ندارد در نتیجه خطای روی هر دو مجموعه نسبتاً بالا و مشابه است. در مدل خیلی پیچیده (بایاس پایین و واریانس بالا) منحنی ناپایدار که روی داده‌ی

آموزشی تقریباً خطای صفر دارد، ولی روی نمونه‌های جدید خطا زیاد می‌شود. نقطه بهینه معمولاً در درجات متوسط است، جایی که ترکیب بایاس و واریانس کمینه می‌شود. این اتفاقات دقیقاً همان چیزی است که Bias–Variance Trade-Off پیش‌بینی می‌کند.

****لازم به ذکر است گزارش و کد در گیت هاب نیز ارائه شده است. لینک آن در ایلرن آمده است****