statistical-analysis

September 5, 2024

1 Exercises for Applying Statistical Methods

1.1 Loading the Diabetes Dataset

[3]:
```
import pandas as pd
from sklearn.datasets import load_diabetes

# Load the dataset
diabetes = load_diabetes()
df = pd.DataFrame(data=diabetes.data, columns=diabetes.feature_names)
df['target'] = diabetes.target

# Display the first few rows
print(df.head())
```

```
        age       sex       bmi        bp        s1        s2        s3  \
0  0.038076  0.050680  0.061696  0.021872 -0.044223 -0.034821 -0.043401
1 -0.001882 -0.044642 -0.051474 -0.026328 -0.008449 -0.019163  0.074412
2  0.085299  0.050680  0.044451 -0.005670 -0.045599 -0.034194 -0.032356
3 -0.089063 -0.044642 -0.011595 -0.036656  0.012191  0.024991 -0.036038
4  0.005383 -0.044642 -0.036385  0.021872  0.003935  0.015596  0.008142

         s4        s5        s6  target
0 -0.002592  0.019907 -0.017646   151.0
1 -0.039493 -0.068332 -0.092204    75.0
2 -0.002592  0.002861 -0.025930   141.0
3  0.034309  0.022688 -0.009362   206.0
4 -0.002592 -0.031988 -0.046641   135.0
```

1.2 Performing Descriptive Statistics

[7]:
```
# Calculate basic descriptive statistics
print("Mean:\n", df.mean())
print("\nMedian:\n", df.median())
print("\nMode:\n", df.mode().iloc[0])
print("\nStandard Deviation:\n", df.std())
print("\nVariance:\n", df.var())
```

```python
# Additional descriptive statistics
print("\nRange:\n", df.max() - df.min())
print("\nSkewness:\n", df.skew())
print("\nKurtosis:\n", df.kurt())
```

```
Mean:
 age      -1.444295e-18
sex       2.543215e-18
bmi      -2.255925e-16
bp       -4.854086e-17
s1       -1.428596e-17
s2        3.898811e-17
s3       -6.028360e-18
s4       -1.788100e-17
s5        9.243486e-17
s6        1.351770e-17
target    1.521335e+02
dtype: float64

Median:
 age         0.005383
sex        -0.044642
bmi        -0.007284
bp         -0.005670
s1         -0.004321
s2         -0.003819
s3         -0.006584
s4         -0.002592
s5         -0.001947
s6         -0.001078
target    140.500000
dtype: float64

Mode:
 age        0.016281
sex       -0.044642
bmi       -0.030996
bp        -0.040099
s1        -0.037344
s2        -0.001001
s3        -0.013948
s4        -0.039493
s5        -0.018114
s6         0.003064
target    72.000000
Name: 0, dtype: float64
```

```
Standard Deviation:
 age        0.047619
sex        0.047619
bmi        0.047619
bp         0.047619
s1         0.047619
s2         0.047619
s3         0.047619
s4         0.047619
s5         0.047619
s6         0.047619
target    77.093005
dtype: float64

Variance:
 age          0.002268
sex          0.002268
bmi          0.002268
bp           0.002268
s1           0.002268
s2           0.002268
s3           0.002268
s4           0.002268
s5           0.002268
s6           0.002268
target    5943.331348
dtype: float64

Range:
 age         0.217952
sex         0.095322
bmi         0.260831
bp          0.244442
s1          0.280694
s2          0.314401
s3          0.283486
s4          0.261629
s5          0.259694
s6          0.273379
target    321.000000
dtype: float64

Skewness:
 age      -0.231382
sex       0.127385
bmi       0.598148
bp        0.290658
s1        0.378108
```

```
s2        0.436592
s3        0.799255
s4        0.735374
s5        0.291754
s6        0.207917
target    0.440563
dtype: float64

Kurtosis:
 age      -0.671224
sex      -1.992811
bmi       0.095094
bp       -0.532797
s1        0.232948
s2        0.601381
s3        0.981507
s4        0.444402
s5       -0.134367
s6        0.236917
target   -0.883057
dtype: float64
```

1.3 Performing Inferential Statistics

[10]:
```python
from scipy import stats

# Example data: BMI values
bmi_values = df['bmi']

# Hypothetical population mean for BMI
population_mean = 0.05

# Perform one-sample t-test
t_stat, p_value = stats.ttest_1samp(bmi_values, population_mean)

print(f"T-Statistic: {t_stat}")
print(f"P-Value: {p_value}")
```

```
T-Statistic: -22.074985843710174
P-Value: 2.7634312235044638e-73
```

1.4 Confidence Intervals

[13]:
```python
import numpy as np
from scipy import stats

# Sample mean and standard error for BMI
sample_mean = np.mean(bmi_values)
```

```
standard_error = stats.sem(bmi_values)

# Compute 95% confidence interval for BMI
confidence_interval = stats.norm.interval(0.95, loc=sample_mean,␣
↪scale=standard_error)

print(f"95% Confidence Interval for BMI: {confidence_interval}")
```

```
95% Confidence Interval for BMI: (-0.004439332370169141, 0.004439332370168915)
```

1.5 Regression Analysis

[16]:

```
import statsmodels.api as sm

# Define independent variable (add constant for intercept)
X = sm.add_constant(df['bmi'])

# Define dependent variable
y = df['target']

# Fit linear regression model
model = sm.OLS(y, X).fit()

# Print model summary
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 target   R-squared:                       0.344
Model:                            OLS   Adj. R-squared:                  0.342
Method:                 Least Squares   F-statistic:                     230.7
Date:                Thu, 05 Sep 2024   Prob (F-statistic):           3.47e-42
Time:                        23:12:36   Log-Likelihood:                -2454.0
No. Observations:                 442   AIC:                             4912.
Df Residuals:                     440   BIC:                             4920.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        152.1335      2.974     51.162      0.000     146.289     157.978
bmi          949.4353     62.515     15.187      0.000     826.570    1072.301
==============================================================================
Omnibus:                       11.674   Durbin-Watson:                   1.848
Prob(Omnibus):                  0.003   Jarque-Bera (JB):                7.310
Skew:                           0.156   Prob(JB):                       0.0259
Kurtosis:                       2.453   Cond. No.                         21.0
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

1.6 Exercise 1: Analyzing a Health-Related Dataset

1.6.1 Loading dataset(breast cancer dataset)

Create visualizations to illustrate the relationships between variables and the regression line.

[25]:
```python
import pandas as pd
from sklearn.datasets import load_breast_cancer
# load the dataset
cancer=load_breast_cancer()
df=pd.DataFrame(data=cancer.data, columns=cancer.feature_names)
df['target'] = cancer.target

# Display the first few rows
print(df.head())
```

```
   mean radius  mean texture  mean perimeter  mean area  mean smoothness  \
0        17.99         10.38          122.80     1001.0          0.11840
1        20.57         17.77          132.90     1326.0          0.08474
2        19.69         21.25          130.00     1203.0          0.10960
3        11.42         20.38           77.58      386.1          0.14250
4        20.29         14.34          135.10     1297.0          0.10030

   mean compactness  mean concavity  mean concave points  mean symmetry  \
0           0.27760          0.3001              0.14710         0.2419
1           0.07864          0.0869              0.07017         0.1812
2           0.15990          0.1974              0.12790         0.2069
3           0.28390          0.2414              0.10520         0.2597
4           0.13280          0.1980              0.10430         0.1809

   mean fractal dimension  …  worst texture  worst perimeter  worst area  \
0                 0.07871  …          17.33           184.60      2019.0
1                 0.05667  …          23.41           158.80      1956.0
2                 0.05999  …          25.53           152.50      1709.0
3                 0.09744  …          26.50            98.87       567.7
4                 0.05883  …          16.67           152.20      1575.0

   worst smoothness  worst compactness  worst concavity  worst concave points  \
0            0.1622             0.6656           0.7119                0.2654
1            0.1238             0.1866           0.2416                0.1860
2            0.1444             0.4245           0.4504                0.2430
3            0.2098             0.8663           0.6869                0.2575
```

```
4           0.1374             0.2050           0.4000                 0.1625

   worst symmetry  worst fractal dimension  target
0          0.4601                  0.11890       0
1          0.2750                  0.08902       0
2          0.3613                  0.08758       0
3          0.6638                  0.17300       0
4          0.2364                  0.07678       0

[5 rows x 31 columns]
```

1.6.2 Performing Descriptive Statistics

Calculate the mean, median, mode, standard deviation, and variance for all the relevant features.

[28]:
```python
# Calculate basic descriptive statistics
print("Mean:\n", df.mean())
print("\nMedian:\n", df.median())
print("\nMode:\n", df.mode().iloc[0])
print("\nStandard Deviation:\n", df.std())
print("\nVariance:\n", df.var())

# Additional descriptive statistics
print("\nRange:\n", df.max() - df.min())
print("\nSkewness:\n", df.skew())
print("\nKurtosis:\n", df.kurt())
```

```
Mean:
 mean radius                  14.127292
mean texture                 19.289649
mean perimeter               91.969033
mean area                   654.889104
mean smoothness               0.096360
mean compactness              0.104341
mean concavity                0.088799
mean concave points           0.048919
mean symmetry                 0.181162
mean fractal dimension        0.062798
radius error                  0.405172
texture error                 1.216853
perimeter error               2.866059
area error                   40.337079
smoothness error              0.007041
compactness error             0.025478
concavity error               0.031894
concave points error          0.011796
symmetry error                0.020542
fractal dimension error       0.003795
```

```
worst radius                16.269190
worst texture               25.677223
worst perimeter            107.261213
worst area                 880.583128
worst smoothness             0.132369
worst compactness            0.254265
worst concavity              0.272188
worst concave points         0.114606
worst symmetry               0.290076
worst fractal dimension      0.083946
target                       0.627417
dtype: float64

Median:
 mean radius                 13.370000
mean texture                18.840000
mean perimeter              86.240000
mean area                  551.100000
mean smoothness              0.095870
mean compactness             0.092630
mean concavity               0.061540
mean concave points          0.033500
mean symmetry                0.179200
mean fractal dimension       0.061540
radius error                 0.324200
texture error                1.108000
perimeter error              2.287000
area error                  24.530000
smoothness error             0.006380
compactness error            0.020450
concavity error              0.025890
concave points error         0.010930
symmetry error               0.018730
fractal dimension error      0.003187
worst radius                14.970000
worst texture               25.410000
worst perimeter             97.660000
worst area                 686.500000
worst smoothness             0.131300
worst compactness            0.211900
worst concavity              0.226700
worst concave points         0.099930
worst symmetry               0.282200
worst fractal dimension      0.080040
target                       1.000000
dtype: float64

Mode:
```

```
 mean radius                 12.340000
mean texture               14.930000
mean perimeter             82.610000
mean area                 512.200000
mean smoothness             0.100700
mean compactness            0.114700
mean concavity              0.000000
mean concave points         0.000000
mean symmetry               0.160100
mean fractal dimension      0.056670
radius error                0.220400
texture error               0.856100
perimeter error             1.778000
area error                 16.640000
smoothness error            0.005080
compactness error           0.011040
concavity error             0.000000
concave points error        0.000000
symmetry error              0.013440
fractal dimension error     0.001784
worst radius               12.360000
worst texture              17.700000
worst perimeter           101.700000
worst area                284.400000
worst smoothness            0.121600
worst compactness           0.148600
worst concavity             0.000000
worst concave points        0.000000
worst symmetry              0.222600
worst fractal dimension     0.074270
target                      1.000000
Name: 0, dtype: float64

Standard Deviation:
 mean radius                  3.524049
mean texture                4.301036
mean perimeter             24.298981
mean area                 351.914129
mean smoothness             0.014064
mean compactness            0.052813
mean concavity              0.079720
mean concave points         0.038803
mean symmetry               0.027414
mean fractal dimension      0.007060
radius error                0.277313
texture error               0.551648
perimeter error             2.021855
area error                 45.491006
```

```
smoothness error             0.003003
compactness error            0.017908
concavity error              0.030186
concave points error         0.006170
symmetry error               0.008266
fractal dimension error      0.002646
worst radius                 4.833242
worst texture                6.146258
worst perimeter             33.602542
worst area                 569.356993
worst smoothness             0.022832
worst compactness            0.157336
worst concavity              0.208624
worst concave points         0.065732
worst symmetry               0.061867
worst fractal dimension      0.018061
target                       0.483918
dtype: float64

Variance:
 mean radius                    12.418920
mean texture                   18.498909
mean perimeter                590.440480
mean area                  123843.554318
mean smoothness                 0.000198
mean compactness                0.002789
mean concavity                  0.006355
mean concave points             0.001506
mean symmetry                   0.000752
mean fractal dimension          0.000050
radius error                    0.076902
texture error                   0.304316
perimeter error                 4.087896
area error                   2069.431583
smoothness error                0.000009
compactness error               0.000321
concavity error                 0.000911
concave points error            0.000038
symmetry error                  0.000068
fractal dimension error         0.000007
worst radius                   23.360224
worst texture                  37.776483
worst perimeter              1129.130847
worst area                 324167.385102
worst smoothness                0.000521
worst compactness               0.024755
worst concavity                 0.043524
worst concave points            0.004321
```

```
worst symmetry                0.003828
worst fractal dimension       0.000326
target                        0.234177
dtype: float64

Range:
 mean radius                21.129000
mean texture               29.570000
mean perimeter            144.710000
mean area                2357.500000
mean smoothness             0.110770
mean compactness            0.326020
mean concavity              0.426800
mean concave points         0.201200
mean symmetry               0.198000
mean fractal dimension      0.047480
radius error                2.761500
texture error               4.524800
perimeter error            21.223000
area error                535.398000
smoothness error            0.029417
compactness error           0.133148
concavity error             0.396000
concave points error        0.052790
symmetry error              0.071068
fractal dimension error     0.028945
worst radius               28.110000
worst texture              37.520000
worst perimeter           200.790000
worst area               4068.800000
worst smoothness            0.151430
worst compactness           1.030710
worst concavity             1.252000
worst concave points        0.291000
worst symmetry              0.507300
worst fractal dimension     0.152460
target                      1.000000
dtype: float64

Skewness:
 mean radius              0.942380
mean texture             0.650450
mean perimeter           0.990650
mean area                1.645732
mean smoothness          0.456324
mean compactness         1.190123
mean concavity           1.401180
mean concave points      1.171180
```

```
mean symmetry              0.725609
mean fractal dimension     1.304489
radius error               3.088612
texture error              1.646444
perimeter error            3.443615
area error                 5.447186
smoothness error           2.314450
compactness error          1.902221
concavity error            5.110463
concave points error       1.444678
symmetry error             2.195133
fractal dimension error    3.923969
worst radius               1.103115
worst texture              0.498321
worst perimeter            1.128164
worst area                 1.859373
worst smoothness           0.415426
worst compactness          1.473555
worst concavity            1.150237
worst concave points       0.492616
worst symmetry             1.433928
worst fractal dimension    1.662579
target                    -0.528461
dtype: float64

Kurtosis:
 mean radius                 0.845522
mean texture                0.758319
mean perimeter              0.972214
mean area                   3.652303
mean smoothness             0.855975
mean compactness            1.650130
mean concavity              1.998638
mean concave points         1.066556
mean symmetry               1.287933
mean fractal dimension      3.005892
radius error               17.686726
texture error               5.349169
perimeter error            21.401905
area error                 49.209077
smoothness error           10.469840
compactness error           5.106252
concavity error            48.861395
concave points error        5.126302
symmetry error              7.896130
fractal dimension error    26.280847
worst radius                0.944090
worst texture               0.224302
```

```
worst perimeter            1.070150
worst area                 4.396395
worst smoothness           0.517825
worst compactness          3.039288
worst concavity            1.615253
worst concave points      -0.535535
worst symmetry             4.444560
worst fractal dimension    5.244611
target                    -1.726811
dtype: float64
```

1.6.3 Performing Inferential Statistics

[ ]:
```
###
```

[41]:
```
from scipy import stats

# Select a specific feature for inferential statistics - let's choose 'mean␣
↪radius'
mean_radius = df['mean radius']


mean_val = mean_radius.mean()
std_val = mean_radius.std()
n = len(mean_radius)


# 95% confidence interval for the mean of 'mean radius'
conf_interval = stats.norm.interval(0.95, loc=mean_val, scale=std_val/np.
↪sqrt(n))

# Hypothesis test: Null Hypothesis H0: The average mean radius is 14
chosen_value = 14
t_statistic, p_value = stats.ttest_1samp(mean_radius, chosen_value)

print(f"T-Statistic: {t_statistic}")
print(f"P-Value: {p_value}")

print(f"95% Confidence Interval for mean_radius: {conf_interval}")
```

```
T-Statistic: 0.8616173566232037
P-Value: 0.3892617071079777
95% Confidence Interval for mean_radius: (13.837734868964587,
14.416848610824518)
```

1.7 Exploring Regression Analysis on a New Dataset

a linear regression analysis to determine the relationship between two or more variables

```
[53]: from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score

# Select two variables for regression analysis:
# Let's predict 'mean texture' (dependent variable) using 'mean radius'␣
↪(independent variable)
X = df[['mean radius']]  # Independent variable
y = df['mean texture']   # Dependent variable

# Split the dataset into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)

# Create a Linear Regression model
model = LinearRegression()

# Fit the model to the training data
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Model evaluation: calculate R^2 and Mean Squared Error
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)

# Output the coefficients and model performance metrics
coef = model.coef_
intercept = model.intercept_

coef, intercept, r2, mse
```

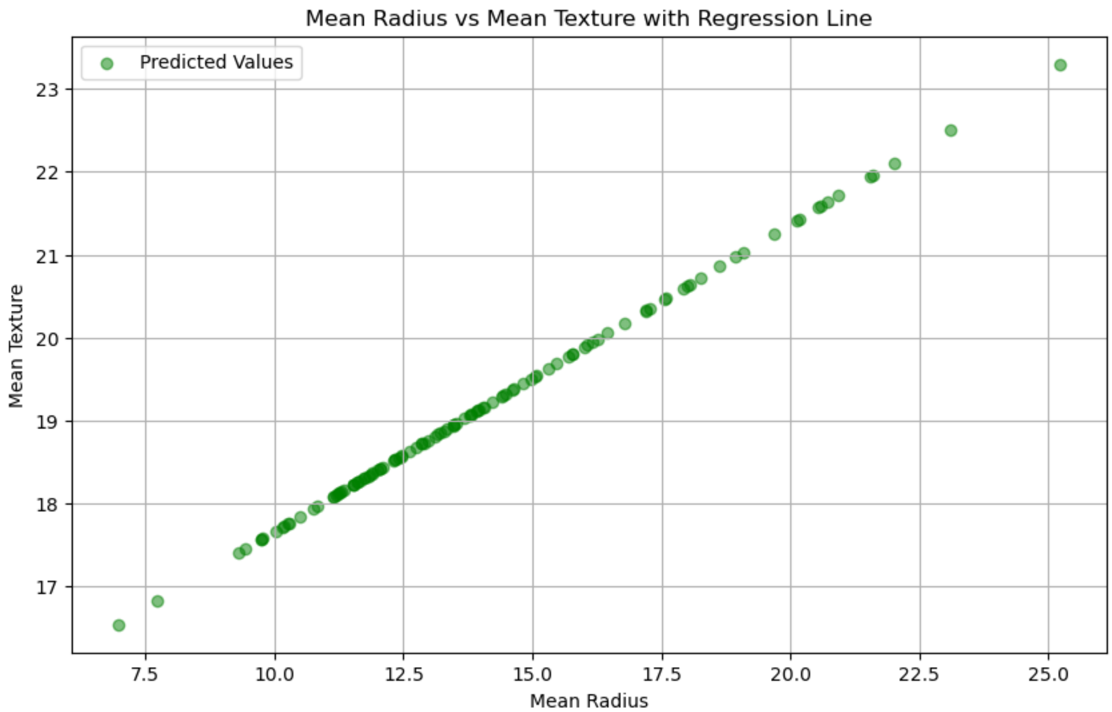
```
[53]: (array([0.37025527]),
 13.95790420860676,
 0.12989038563227218,
 16.946319739410345)
```

Create visualizations to illustrate the relationships between variables and the regression line.

```
[55]: import matplotlib.pyplot as plt

# Scatter plot with regression line for 'mean radius' vs 'mean texture'
plt.figure(figsize=(10,6))
plt.title('Mean Radius vs Mean Texture with Regression Line')
plt.xlabel('Mean Radius')
```

```
plt.ylabel('Mean Texture')
plt.legend()
plt.grid(True)
plt.show()
```




```
[ ]:
```