Machine Learning for Economics and Finance

01_Auto_data_2

Ole Wilms

July 29, 2024

Setup for Tasks 2

The following steps will outline the initial setup of the Auto dataset, including the creation of separate training (train data) and test (test data) sets.

Get and Set working directory

[2]:
```
import os            # Package to access system related information
print(os.getcwd())   # Prints the current working directory
path = os.getcwd()
os.chdir(path)       # Set the working directory
```

```
/mnt/ds/home/UHH_MLSJ_2024/Code/Python/01_SupLearn_Regression
```

Loading the package and data

[3]:
```
from ISLP import load_data  # Package which contains the data
Auto = load_data('Auto')    # Loading the data
Auto.head()                 # Showing the first 5 Lines of Data.
```

[3]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
0	18.0	8	307.0	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	ford torino

[4]:
```
n  = int(len(Auto)) # Number of observations in the dataset
nT = int(n/2)       # training sample size
nV = int(n/2)       # validation sample size
```

Define training and test set

[5]:
```
import pandas as pd
import numpy as np

np.random.seed(2)  # set seed

# Define training and test sets
train_sample = np.random.choice(n, nT, replace=False)  # indices for training␣
 ↪data
train_data = Auto.iloc[train_sample]                   # training dataset
test_data = Auto.drop(train_sample)                    # test dataset
```

We start with the following univariate linear regression:

$$\text{mpg} = \beta_0 + \beta_1 \text{horsepower} + \varepsilon$$

Fit model on training data and calculate training MSE

[6]:
```
import statsmodels.formula.api as smf

# fit model on training data and calculate training MSE
fit_lm = smf.ols(formula='mpg ~ horsepower', data = train_data).fit()
```

[7]:
```
print(fit_lm.summary().tables[1])
```

```
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     39.0131      0.994     39.245      0.000      37.053      40.974
horsepower    -0.1510      0.009    -17.040      0.000      -0.168      -0.134
==============================================================================
```

[10]:
```
# Predictions for training data
y_head_train = fit_lm.predict(train_data)
```

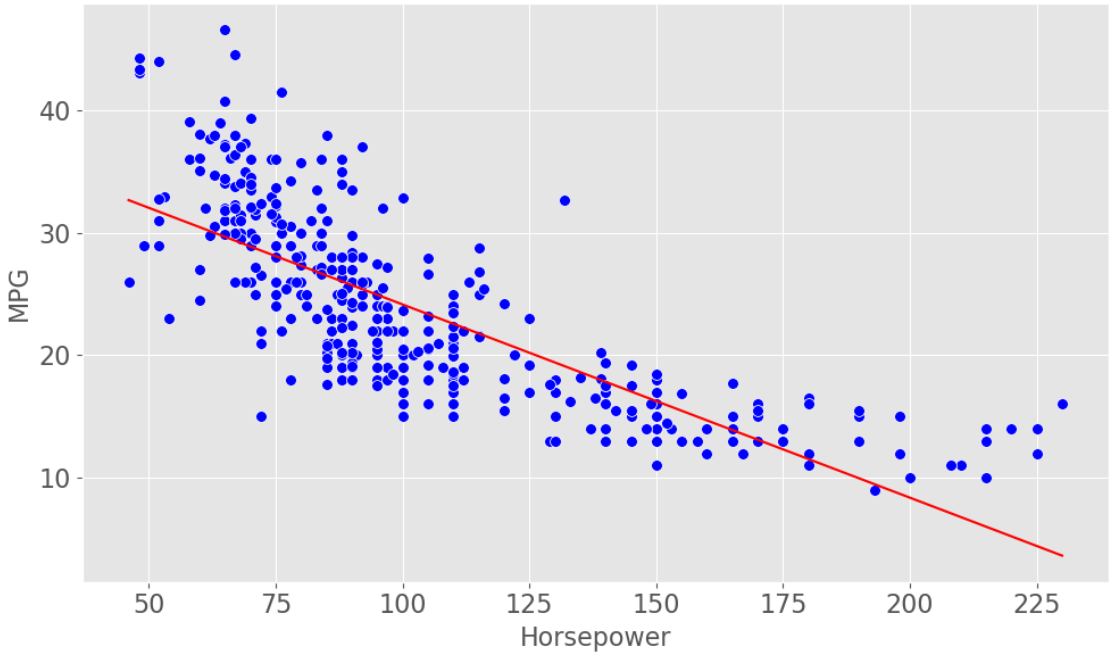
[9]:
```
# Function to compute the mean squared error (MSE)
# Takes realized values y and corresponding predictions y_head
# as inputs and returns MSE as output
def MSE(y, y_head):
  return((y - y_head)**2).mean()

# Compute the mean squared error
MSE_train = MSE(train_data['mpg'], y_head_train)
print(f"Mean Squared Error:  {MSE_train:.3f}")
```

```
Mean Squared Error:  23.002
```

Extra visualisation

[21]:
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression

# Plotting
plt.figure(figsize=(10, 6), tight_layout=True)
plt.style.use('ggplot')

# Scatter plot of the Auto data
sns.scatterplot(x=Auto['horsepower'], y=Auto['mpg'], color='blue', s=50)

X = Auto[['horsepower']]  # Make sure X is a 2D array
y = Auto['mpg']
lm = LinearRegression().fit(X,y)

# Generate predictions across the range of horsepower values
x_range = np.linspace(Auto['horsepower'].min(), Auto['horsepower'].max(), 100)
x_range_reshaped = x_range.reshape(-1, 1)  # Reshape x_range to a 2D array
y_pred = lm.predict(x_range_reshaped)

# Plot the predicted line together with the data
plt.plot(x_range, y_pred, color='red', label='Fitted line')

# Adjust the text size
plt.xticks(fontsize=16)
plt.yticks(fontsize=16)
plt.xlabel('Horsepower', fontsize=16)
plt.ylabel('MPG', fontsize=16)
plt.title('', fontsize=18)

# Show plot
plt.show()
```

```
/usr/lib/python3.12/site-packages/sklearn/base.py:493: UserWarning: X does not
have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```



Task 2

1. Compute the test MSE for the univariate linear model and note the training and test MSE.

Hint: First copy line 84 and generate predictions $\hat{y}$ for the test data. Then copy line 98 and change it correspondingly to compute the test MSE.

[ ]:

2. Compute and note the training and test MSE for a quadratic model:

$$\text{mpg} = \beta_0 + \beta_1 \text{horsepower} + \beta_2 \text{horsepower}^2 + \varepsilon$$

[ ]:

3. Redo Step 2 but add the term horsepower^3 to the regression. Then add horsepower^4 and so on. For each model note the training and test MSE. How do the two MSE change with the flexibility of the method?

[ ]: