**Wasil Engel**

**12231558**

**PSet 1**

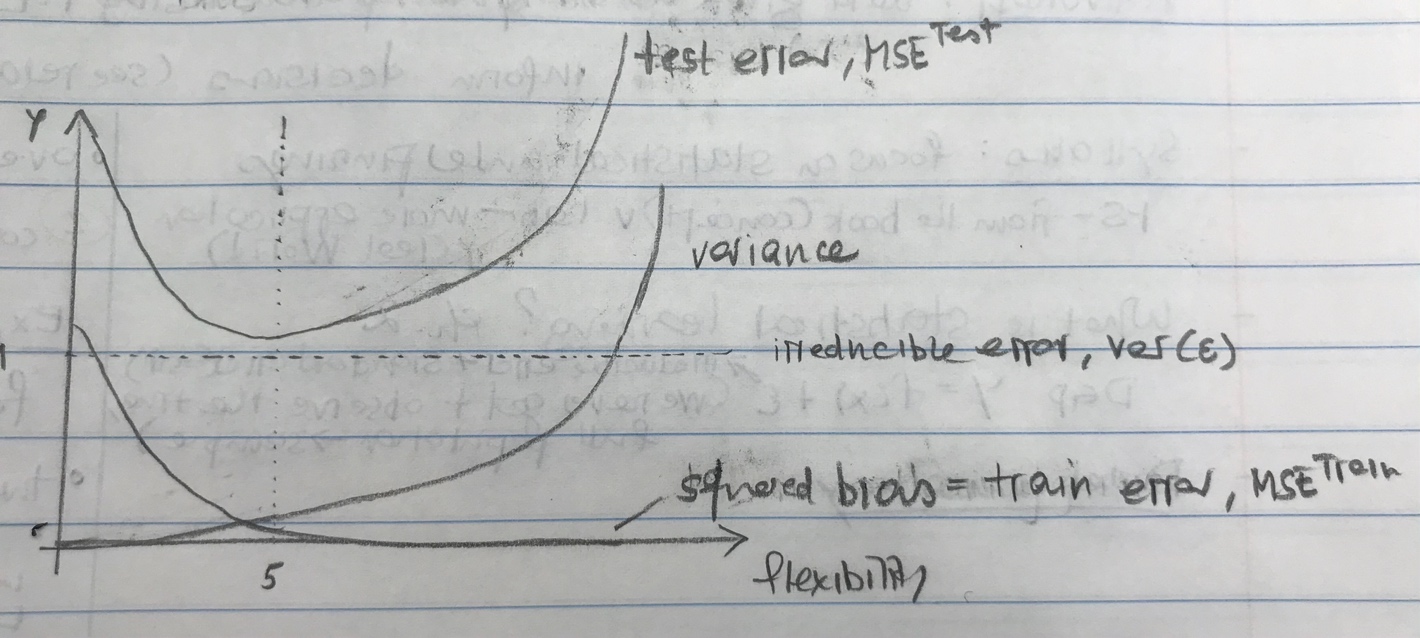
*I partnered up with Ricardo Saucedo (12245077).*

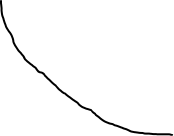
*To see more than the screenshots of the plots I created for the coding part of the assignment, please see my Jupyter notebook on Github here:*

**Ch 2**

**# 3**

a. bias-variance trade-off:





In grey // MSETrain in black

1

0

b. We want to minimize the test error, MSETest, so in order to do that, we need to achieve both low variance and low bias, both of which are. In the graph I plotted above, the bias initially decreases rapidly resulting in an initial sharp decrease of MSETest too. However, as I apply more flexible methods towards an even better fit, the bias and the training data, MSETrain, further decrease, and with the increase in flexibility, the variance increases (note that once MSETrain is below the irreducible error, our model is overfit). It is the relative rate of change between the variance and the bias that determines whether the test error will increase or decrease. If the bias decreases faster than the variance increases, the test error declines (see left half of the plot up until dotted, vertical line). However, if we reach a certain threshold at which increasing flexibility has only little impact on the bias and the variance starts to significantly increase, the test error increases too (convex shape). The irreducible error defines the lower limit that MSETest is bounded by.

**#5**

As I have shown above, methods that are extremely flexible can eliminate bias in the training data giving it “the right fit”. However, that comes at the risk of overfitting the model and not be useful/ applicable to data other than the training data so that the test error and variance will sharply increase. For example, a linear, less flexible model might work fine in situations in which we have a limited number of observations allowing us to simply uncover a trend in the data, e.g. downward/ upward slope. However, should we have a large number of training data, we could “afford” using a more flexible approach – yet, it will generally require us to estimate a greater number of parameters (consequently: increasing variance explained) since they are what gives the quadratic function its shape. This comes at the expense of interpretability of the results. Hence, a less flexible approach is more desirable if we care more about easily interpreting our findings.

**# 10 Coding p.56**

# a

# To begin, load in the Boston data set.

path = '/Users/wasilengel/Desktop/School/Harris/Machine Learning/Boston/Boston.csv'

df = pd.read\_csv(path)

df

# How many rows are in this data set? 506 (505 observations, excluding headline with variable names)

# How many columns? 14 (index automatically excluded)

# What do the rows and columns represent? It shows Boston housing data and as such lists the various factors affecting the quality of housing per town/ suburb.

# b

# Make some pairwise scatterplots of the predictors (columns) in this data set. Describe your findings.

sns.pairplot(df, vars = ["CRIM", "B", "LSTAT", "MDEV", "PTRATIO"])



# Based on the predictors that I chose, the pairwise scatterplot yields some interesting results. For example, there is a strong negative correlation between the percentage of the lower status of the population and the median value of owner-occupied home. That means that the larger the percentage share of the lower population status is, the lower is the median value of the homes in these areas (that's pretty intuitive). Furthermore, there is a positive trend between the percentage of lower status of the population and the pupil-teacher ratio, which is sad but true: the poorer a suburb is, the higher the pupil-teacher ratio is. In a similar vein, a (slight) positive trend can also be observed with regards to the crime rate: the poorer the suburb, the higher the crime rate is. This observation will lead me to answering the next question.

# c

# Are any of the predictors associated with per capita crime rate? If so, explain the relationship.

for col in df.iloc[:,1:14].columns:

sns.scatterplot(df[col],df['CRIM'])

plt.title(col)

plt.xlabel(col)

plt.ylabel('CRIM')

plt.show()



# Plotting crime against the other variables, I can observe that the crime rate is associstated with (to name a few):

# NOX, nitric oxides concentration (parts per 10 million): higher crime rates where nitric oxides concentrates around 60-70%

# RM, average number of rooms per dwelling: higher crime rates where the average number of rooms per dwelling is lower

# AGE, proportion of owner-occupied units built prior to 1940: higher crimes rates where there are older homes (interesting!)

# DIS, weighted distances to five Boston employment centres: higher crime rates where the distance to employment centres is closer

# MDEV, Median value of owner-occupied homes in $1000's: higher crime rates where the median value of homes is lower

# d

# Do any of the suburbs of Boston appear to have particularly high crime rates?

# Yes, suburbs 380 (89%), 418 (74%), and 405 (68%) are the three suburbs with the highest crime rates.

df.sort\_values(by=['CRIM'], ascending=False).head(3)

# Tax rates:

tax\_list = df["TAX"]

tax\_list.min()

tax\_list.max()

# With a full-value property-tax rate per $10,000 of $666, these suburbs are particularly highly taxed (max tax value at $711, min tax value at $187).

# Pupil-teacher ratios?

ptratio\_list = df["PTRATIO"]

ptratio\_list.min()

ptratio\_list.max()

# With a pupil-teacher ratio by suburb at 20.2, these pupil-teacher ratio is particularly high (max ratio at 22, min ratio at 12.6).

# e

# How many of the suburbs in this data set bound the Charles river?

# Note that it's a dummy variable = 1 if tract bounds river, 0 otherwise

len(df.loc[df['CHAS'] == 1.0])

# The answer is 35 suburbs.

# f

# What is the median pupil-teacher ratio among the towns in this data set?

df["PTRATIO"].median()

# The median pupil-teacher ratio among the towns/ suburbs is 19.05.

# g

# Which suburb of Boston has lowest median value of owneroccupied homes?

df.sort\_values(by=['MDEV']).head(3)

# The two suburbs 398 and 405 have the lowest median value of owneroccupied homes.

# What are the values of the other predictors for that suburb, and how do those values compare to the overall ranges for those predictors? Comment on your findings.

# Compared to the two ranges that I had already identified above, PTRATIO between 12.6 and 22, and TAX between $187 and $711, I can see that these two suburbs are also particularly high taxed and are characterized by a very high pupil-teacher ratio too. With regards to other predictors, for example LSTAT and AGE (see ranges below), I see that the proportion of owner-occupied units built prior to 1940 is at the max end in both cases (that means all houses are old: 100%) and the percentage of the lower status of the population is also relatively high, 30.59% and 22.98%, respectively.

df["LSTAT"].max() # 1.73

df["LSTAT"].max() # 37.97

df["AGE"].min() # 2.9

df["AGE"].max() # 100

# h

# In this data set, how many of the suburbs average more than seven rooms per dwelling? More than eight rooms per dwelling?

len(df.loc[df['RM'] > 7])

# 64 suburbs have more than seven rooms per dwelling.

len(df.loc[df['RM'] > 8])

# 13 suburbs have more than eight rooms per dwelling.

# Comment on the suburbs that average more than eight rooms per dwelling.

eightplus = df.loc[df['RM'] > 8]

eightplus

# Given the knowledge about the range values I obtained in the previous taxes, I can see that dwellings with 8 or more rooms:

# - are higher than average in terms of AGE, the proportion of owner-occupied units built prior to 1940 -- range in df between 2.9% and 100% with a mean of 68.6%, here: 71.5%

df["AGE"].mean() # 68.57490118577078

eightplus["AGE"].mean() # 71.53846153846153

# - are lower than average in terms of LSTAT, the percentage of the lower status of the population -- range between 1.73% and 37.97% with a mean of 12.7%, here: 4.3%

df["LSTAT"].mean() # 12.653063241106723

eightplus["LSTAT"].mean() # 4.31

# - are lower than average in terms of PTRATIO, the pupil-teacher ratio by town -- range between 12.6 and 22 with a mean of 18.5, here: 16.4

df["PTRATIO"].mean() # 18.455533596837967

eightplus["PTRATIO"].mean() # 16.361538461538462

# - are lower than average in terms of TAX, the full-value property-tax rate per $10,000 -- range between $187 and $711 with a mean of $408.2, here: $325.1

df["TAX"].mean() # 408.2371541501976

eightplus["TAX"].mean() # 325.0769230769231

# Overall, these dwellings seem to be located in one of the better Boston suburbs!

**Ch 3**

**# 3**

a. iii is the correct answer because when we fix IQ at 100 and GPA at a moderate level, say 3, we see that for male with gender=0, Y = 50 + 20(3) + 0.07(100) + 35(0) + 0.01(3 \* 100) - 10 (3 \* 0) = **120**, while for female with gender=1, Y = 50 + 20(3) + 0.07(100) + 35(1) + 0.01(3 \* 100) - 10 (3 \* 1) = **125**. However, once we set the GPA at a better level, say 4, we see that for male, Y = 50 + 20(4) + 0.07(100) + 35(0) + 0.01(4 \* 100) - 10 (4 \* 0) = **141** while for female, Y = 50 + 20(4) + 0.07(100) + 35(1) + 0.01(4 \* 100) - 10 (4 \* 1) = **136.** I assume the numbers are given in thousand dollars/ year.

b. Y = 50 + 20(4) + 0.07(110) + 35(1) + 0.01(4 \* 110) - 10 (4 \* 1) = **137.1**

c. False. The coefficient estimate doesn’t provide the statistical evidence needed to conclude that there’s an interaction effect, or not. Instead, in order to come to such a conclusion, we must conduct appropriate hypothesis testing and examine the respective p-value, or calculate the confidence intervals.

**# 15 Coding p.126**

﻿# We will now try to predict per capita crime rate using the other variables in this data set. In other words, per capita crime rate is the response, and the other variables are the predictors.

# a

# For each predictor, fit a simple linear regression model to predict the response.

import statsmodels.formula.api as smf

for col in df.iloc[:,1:14].columns:

result = smf.ols('CRIM ~ df[col]', data = df).fit()

print("The next predictor is")

print(col)

print(result.summary())

print('' + '\n' + '\n' + '\n')

list = [0.1071, -0.0355, -1.8715, -1.5428, 0.5068, 0.5444, -0.3606, 30.9753, 1.1446, 0.6141, -2.6910, 0.0296, 4.4292]

# Describe your results. In which of the models is there a statistically significant association between the predictor and the response? Create some plots to back up your assertions.

# There is a statistically significant association at alpha = 0.05 between crime (dependent variable) and all the predictors except CHAS (which is binary).

# Please note that I already created an extensive series of plots above in ch. 2, no. 10.

# However, to back my findings, I will now map the following two plots using regplot this time, two showing statistically significant relationships, e.g. TAX and MDEV, and one that is non-significant for the data I was given, here CHAS.

# A 1%-increase in full-value property-tax is associated with a 2.96% increase in crime rate per town.

plt.figure(figsize=(12,8))

sns.regplot(x="TAX", y="CRIM", data=df)

plt.xlabel("full-value property-tax rate per $10,000", fontsize=15)

plt.ylabel("per capita crime rate by town", fontsize=15)

plt.title('CRIM vs TAX', fontsize=20)

# A one-unit increase in the median value of owner-occupied homes, here in $1000, is associated with a decrease in crime by 36%.

plt.figure(figsize=(12,8))

sns.regplot(x="MDEV", y="CRIM", data=df)

plt.xlabel("Median value of owner-occupied homes in $1000's", fontsize=15)

plt.ylabel("per capita crime rate by town", fontsize=15)

plt.title('CRIM vs MEDV', fontsize=20)

# The straight line implies that the relationship between the two variables is not statistically meaningful.

plt.figure(figsize=(12,8))

sns.regplot(x="CHAS", y="CRIM", data=df)

plt.xlabel("Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)", fontsize=15)

plt.ylabel("per capita crime rate by town", fontsize=15)

plt.title('CRIM vs CHAS', fontsize=20)



# b

# Fit a multiple regression model to predict the response using all of the predictors.

predictors = ' + '.join(df.columns.difference([14,'CRIM']))

results = smf.ols('CRIM ~ {}'.format(predictors),data = df).fit()

print(result.summary())

# Describe your results. For which predictors can we reject the null hypothesis H0 : βj = 0?

# Here, we can reject the null at alpha = 0.05 for ZN, RAD, NOX (note how it's at the margin yet t>2 and p<5.0), MDEV, and DIS. For all other predictors, we cannot reject the null, hence, we cannot conclude a statistically significant relationship. Given that the null for an F-test (which is what we are performining here, it's our default for multivariate regression analysis) is much more pessimistic by design (we assume that no regressor affects our dependent variable, CRIM), it doesn't surprise me that less predictors are now statistically significant compared with a. To describe the results, I can say that, e.g.

# - A 1%-increase in full-value property-tax is associated with a -0.37% decrease in crime rate per town, holding all other factors constant. This is interesting because the relationship is suddenly reverse (unlike a). However, given that our R-squared is so much higher now - it's at 44.8% and before, in a, for TAX, it was only at 33.6% - this model explains more variation but is it the preferred choice? Perhaps, but I keep in mind that R-square does never increase and either stays constant or increases as we add more predictors. That's why I look at the adjusted R-squared, which is much higher here too, it's at 43.4%, so, I'd conclude that the tax relationship is indeed negative.

# - A one-unit increase in the median value of owner-occupied homes, here in $1000, is associated with a decrease in crime by 19.92%. Similarly to how I argued above with regards to the R-squared value, the multivariate model explains more variation, hence, I'd pay heed to the approx. 20% decrease here. However, unlike before, at least the direction of the relationship, here negative, remains the same.

# c

# How do your results from (a) compare to your results from (b)? For a comparison regarding the null and the R-squared value, see above in b.

# Create a plot displaying the univariate regression coefficients from (a) on the x-axis, and the multiple regression coefficients from (b) on the y-axis. That is, each predictor is displayed as a single point in the plot. Its coefficient in a simple linear regression model is shown on the x-axis, and its coefficient estimate in the multiple linear regression model is shown on the y-axis.

result.params # multivariate

multi = results.params[1:14] # removing the intercept -> 13 variables

uni = list # 13 variables, see above in a

multi\_df = multi.to\_frame().reset\_index()

uni\_dictionary = {"uni":[0.1071,

-0.0355,

-1.8715,

-1.5428,

0.5068,

0.5444,

-0.3606,

30.9753,

1.1446,

0.6141,

-2.691,

0.0296,

4.4292]}

uni\_df = pd.DataFrame(data = uni\_dictionary)

joined\_df = multi\_df.join(uni\_df)

joined\_df.columns = ["index", "multi", "uni"]

joined\_df

plt.figure(figsize=(12,8))

sns.regplot(x="uni", y="multi", data=joined\_df)

plt.xlabel("univariate regression coefficients", fontsize=15)

plt.ylabel("multivariate regression coefficients", fontsize=15)

plt.title('Comparison univariate v. multivariate regression coefficients', fontsize=20)



# d

# Is there evidence of non-linear association between any of the predictors and the response? To answer this question,

# for each predictor X, fit a model of the form Y = β0 + β1X + β2X^2 + β3X^3 + E.

## We haven't covered this yet, neither in class nor in lab ...