

Exercise 2 - Multiple Linear Regression

Introduction

In this article we aim to make an accurate linear model prediction of Ozone levels in the atmosphere based on the given dataset. At the beginning we try to gain intuition of the variables by plotting them in pairs and calculate Pearson's correlation coefficient. Then, we construct a simple linear model and study the Adjusted-R-squared and R-squared coefficients. We make residuals analysis and Q-Q plots to draw conclusions of the model. Finally, we follow the best regression selection method by using *regsubsets* command in order to find the best multiple linear model by examine 4 main metrics, RSS, Adjusted-R-squared, Cp and BIC.

Load Dataset

We first load ozone dataset and store it into `df` variable. The dataset consists of 14 columns. First column is the *date* of the observation but won't be used in our analysis.

```
df <- read.table("ozone.txt", header = TRUE, sep=" ")
df <- df[, 2:ncol(df)]
attach(df)
head(df)
```

##	maxO3	T9	T12	T15	Ne9	Ne12	Ne15	Wx9	Wx12	Wx15	maxO3y	wind	rain
## 1	87	15.6	18.5	18.4	4	4	8	0.6946	-1.7101	-0.6946	84	North	Dry
## 2	82	17.0	18.4	17.7	5	5	7	-4.3301	-4.0000	-3.0000	87	North	Dry
## 3	92	15.3	17.6	19.5	2	5	4	2.9544	1.8794	0.5209	82	East	Dry
## 4	114	16.2	19.7	22.5	1	1	0	0.9848	0.3473	-0.1736	92	North	Dry
## 5	94	17.4	20.5	20.4	8	8	7	-0.5000	-2.9544	-4.3301	114	West	Dry
## 6	80	17.7	19.8	18.3	6	6	7	-5.6382	-5.0000	-6.0000	94	West	Rainy

Second column labeled as *maxO3* is the depended variable which we would like to predict by constructing a model from the remaining 12 variables, *Temperature(T)*, *Neon(Ne)*, *Wx* and *maxO3y*, which had been collected at three different times during the day, and two nominal variables *Wind* and *Rain*.

First we make a simple check for potential missing values in our dataset.

```
head(is.na.data.frame(df))
```

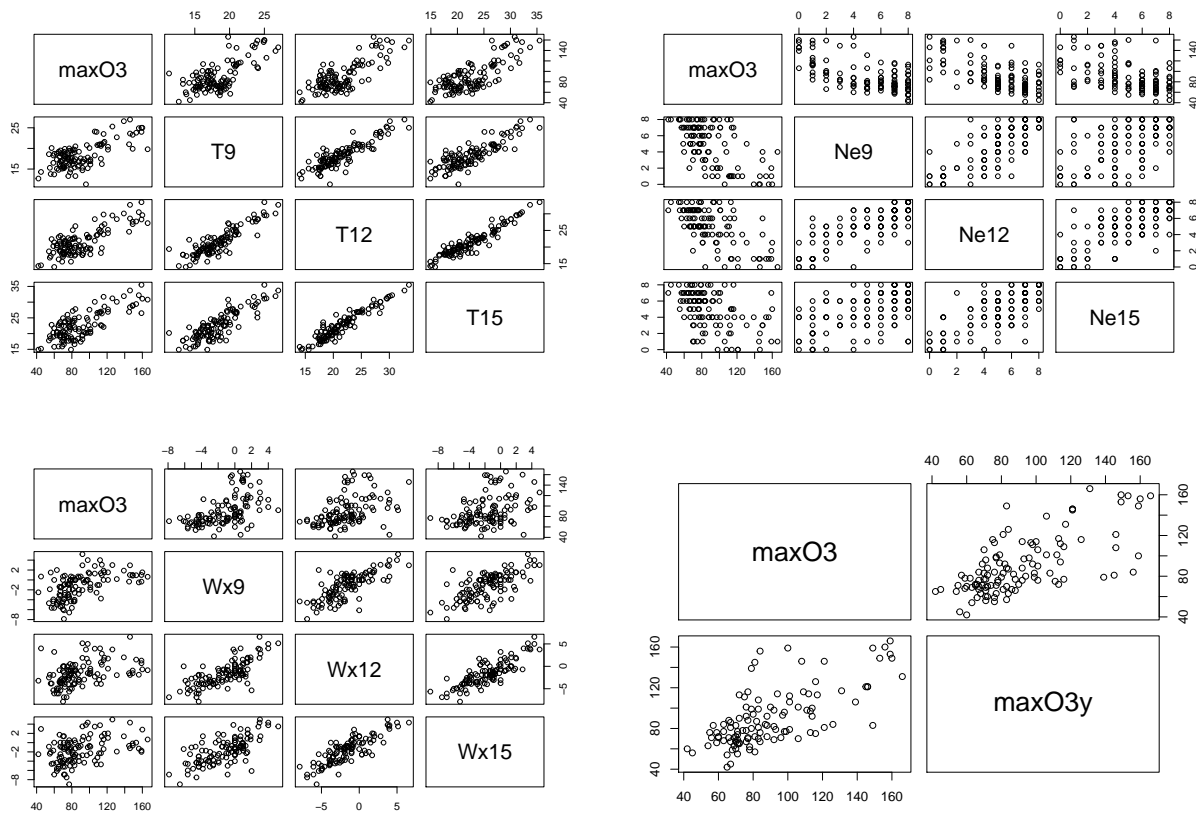
##	maxO3	T9	T12	T15	Ne9	Ne12	Ne15	Wx9	Wx12	Wx15	maxO3y	wind	rain
## [1,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
## [2,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
## [3,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
## [4,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
## [5,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
## [6,]	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
##	rain												
## [1,]	FALSE												

```
## [2,] FALSE
## [3,] FALSE
## [4,] FALSE
## [5,] FALSE
## [6,] FALSE
```

Understanding the data

Once we have validated the integrity of the dataset, we plot the variables in pairs. We use the built-in command *pairs* which is provided by RStudio. In order to keep our plots simple and readable, we will plot *max03* with the 3 main variables, *Temperature*, *Ne*, *Wx* and *max03y* separately.

```
pairs(subset(df, select = c(1,2,3,4)))
pairs(subset(df, select = c(1,5,6,7)))
pairs(subset(df, select = c(1,8,9,10)))
pairs(subset(df, select = c(1,11)))
```



The main diagonal is the correlation of each variable with itself. Intuitively, we could say that *max03* levels are more correlated with temperature compared to *Wx9* and *Ne*. A statistical method to capture any linear correlation which will be used, is Pearson's coefficient.

Pearson coefficient

We've seen before that temperature has a strong correlation with *max03*. In order to measure that relationship we will calculate the Pearson correlation coefficient *R*. Pearson coefficient measures the strength and

direction of a linear relationship between two variables. The value of R is always between +1 and -1. Closer to +1 values means there is a very strong positive correlation between variables while closer to -1 a very strong negative correlation. 0 indicates that there is no linear correlation.

In order to compute coefficient R we use built-in method *cor* and we explicitly ask for pearson method.

```
cor_9 = cor(df$T9, df$max03, method = c("pearson"))
cor_12 = cor(df$T12, df$max03, method = c("pearson"))
cor_15 = cor(df$T15, df$max03, method = c("pearson"))
```

	T9	T12	T15
r-coeff(max03)	0.6993865	0.7842623	0.77457

We observe that all values (*0.6993865*, *0.7842623*, *0.77457*) are positive and close to 1 which indicates a strong positive linear relationship. It's worth mentioning that when adding multiple predictors to the model, we care about features with different Pearson coefficients, even in the case with weak correlation because it will increase the performance by learning the data better.

Linear Regression

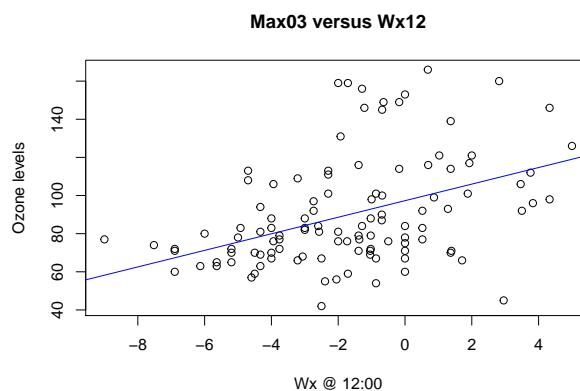
Simple Linear Regression

Previous analysis hinted that *Temperature* variable has the strongest correlation with *max03*. For the purposes of the exercise, we will use *Wx12* as a regressor to construct a simple linear model. A simple linear model is expressed mathematically as shown below.

$$\hat{y} = \beta_0 + \beta_1 x + \epsilon \quad (1)$$

The objective is to fit a straight line to the data such that the sum of squared errors are minimized. In RStudio, we simply use the command *lm* (linear model) to fit a linear model to observations and *summary* to get basics statistics of the fit.

```
simple.model <- lm(df$max03 ~ df$Wx12)
plot(df$Wx15, df$max03, main = "Max03 versus Wx12", xlab = "Wx @ 12:00", ylab = "Ozone levels")
abline(simple.model, col="blue")
```



```
summary(simple.model)
```

```
##
## Call:
## lm(formula = df$max03 ~ df$Wx12)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -69.67 -14.61  -6.49   10.22   72.46
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   97.3009     2.7898  34.877  < 2e-16 ***
## df$Wx12        4.3435     0.8675   5.007 2.12e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25.55 on 110 degrees of freedom
## Multiple R-squared:  0.1856, Adjusted R-squared:  0.1782
## F-statistic: 25.07 on 1 and 110 DF,  p-value: 2.123e-06
```

The first metric we are interested in, is R-squared or coefficient of determination. R-squared represents the proportion of variance in dependent variable, in this case *max03* that has been explained by the independent variables, in this case *Wx12* in the model. It is defined as:

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total variance}} \quad (2)$$

In this *simple.model* R-squared is *0.1856* and interprets as: ~18% of the increase in *max03* levels is due to increase in *Wx12* level. It gives a measure of how well unseen samples are likely to be predicted by the model based on the proportion of explained variance. Larger R-squared values indicate better linear model that fits the observations. Visually, it means that the observed data points are closer to the regression line.

Limitation of R-squared coefficient is that it does not provide any information whether our model is biased to the data. R-squared can be misleading when you assess the goodness-of-fit for linear regression analysis. A good model could have a low R-squared value which we will deal with it later by performing a residuals plots analysis.

Residual Plots

In linear regression, the difference between the observed value of the dependent variable and the predicted value, is called *residual*. For each value/point the mathematic expression with one independent variable is shown in 3 which applies for models with multiple regressors as well.

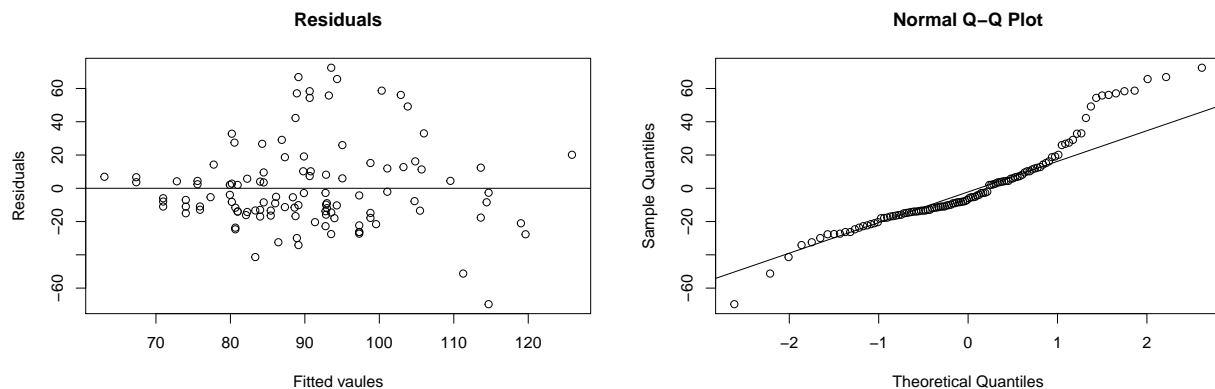
$$res = \hat{y} - (\beta_0 + \beta x) \quad (3)$$

Both the sum and the mean of the residuals are equal to zero. A random scatter of residuals mean that they do not contradict the linear assumption, while distinct curved patterns suggests that a linear model is not a good fit. The following RStudio commands allow us to create the residuals and Q-Q plot.

```

simple.model.residuals = resid(simple.model)
simple.model.fitted = fitted.values(simple.model)
plot(simple.model.fitted, simple.model.residuals,
     ylab = "Residuals",
     xlab = "Fitted vaules",
     main = "Residuals")
abline(0,0)
#create Q-Q plot for residuals
qqnorm(simple.model.residuals)
#add a straight diagonal line to the plot
qqline(simple.model.residuals)

```



The x-axis on left figure displays the fitted values and the y-axis displays the residuals. From the plot we can see that the spread of the residuals tends to be higher for higher fitted values. Additionally, we can use Q-Q plot to validate the assumption that the residuals follow a normal distribution. Closer to straight line validates this. Though, it is clear that the upper tail tends to stray away from the line.

Multiple Linear Regression

Previously, we've seen how to assess a linear model with on predictor. In this section we will use multiple regressors to predict *max03* by taking into consideration the 3 variables *T12*, *Ne12* and *Wx12*. Multiple linear regression model is as expressed similarly to (1) but with number of predictors $p > 1$.

$$\hat{y} = \beta_0 + \beta_1 x_9 + \beta_2 x_{12} + \beta_3 x_{15} \quad (4)$$

Matrix notation:

$$Y = X\beta + \epsilon \quad (5)$$

Similarly to simple model, we use the *lm* command with the addition of two extra variables *T12* and *Ne12*.

```

multi.model <- lm(df$max03 ~ df$T12 + df$Ne12 + df$Wx12)
summary(multi.model)

```

```

##
## Call:
## lm(formula = df$max03 ~ df$T12 + df$Ne12 + df$Wx12)

```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -37.462 -11.448  -0.722   8.908  46.331
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.8958    14.8243   0.263   0.7932
## df$T12         4.5132     0.5203   8.674 4.71e-14 ***
## df$Ne12        -1.6189     1.0181  -1.590   0.1147
## df$Wx12         1.6290     0.6571   2.479   0.0147 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.63 on 108 degrees of freedom
## Multiple R-squared:  0.6612, Adjusted R-squared:  0.6518
## F-statistic: 70.25 on 3 and 108 DF,  p-value: < 2.2e-16
```

As expected, R-squared value is higher than the *simple.model* as it never decreases when new predictors are added. R-squared is encouraging us to make more complex model for the prediction of *max03*. Though, that would result to *overfitting* and lack of generalization ability.

Instead, we focus on another metric called Adjusted R-squared coefficient which is defined as shown below.

$$AdjustedR^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1} \quad (6)$$

p - number of predictors

n - sample size.

For every predictor added in the model there is a penalty factor. As the denominator decreases the fraction increases, thus adjusted- R^2 gets smaller. In case R^2 is significantly larger with the addition of new regressors then adding new variables to the model was worth it. Next section we will discuss how select the best model for the dataset.

Model Selection

In this section, we will present a methodology that helps assessing the quality of complex linear models as well as quantitative comparison among different models. We apply the best subset selection approach to the train data. First we create a new dataset containing only numerical features. Before continuing with the implementation it is important to define model comparison metrics.

```
df_num <- df[, purrr::map_lgl(df, is.numeric)]
head(df_num)
```

```
##      max03    T9   T12   T15  Ne9  Ne12  Ne15      Wx9      Wx12      Wx15  max03y
## 1      87 15.6 18.5 18.4    4    4    8  0.6946 -1.7101 -0.6946    84
## 2      82 17.0 18.4 17.7    5    5    7 -4.3301 -4.0000 -3.0000    87
## 3      92 15.3 17.6 19.5    2    5    4  2.9544  1.8794  0.5209    82
## 4     114 16.2 19.7 22.5    1    1    0  0.9848  0.3473 -0.1736    92
## 5      94 17.4 20.5 20.4    8    8    7 -0.5000 -2.9544 -4.3301   114
## 6      80 17.7 19.8 18.3    6    6    7 -5.6382 -5.0000 -6.0000    94
```

The best subsets regression, *regsubsets* is a model selection approach that consists of testing all possible combinations of the regressor variables and then selecting the best model according to statistical metrics. Particularly, we are interested in, *Adjusted-R2*, *RSS*, *Cp* and *BIC* which are the most commonly used metrics for measuring regression model quality and models comparison. As mentioned above, *Adjusted-R2* shows the percentage of variance in the outcome that can be explained by predictors variance. *Cp* and *BIC*, address the issue of overfitting, as inevitably more variables added to the model will results to smaller errors. Mathematically are expressed in equations 7 and 8:

$$Cp = \frac{RSS_p}{S^2} - n + 2(p + 1) \quad (7)$$

RSS - Residual sum of squares.

p - number of predictors.

n - sample size.

Bayesian information criterion (BIC) introduces a penalty term for the number of predictors in the model.

$$BIC = k \ln(n) - 2 \ln(\hat{L}) \quad (8)$$

n - sample size. L - the maximized value of likelihood function. k - number of predictors estimated by the model.

Residual sum of Squares (RSS) are the deviations predicted from actual empirical values of the data. *RSS* is defined such:

$$RSS = \sum_{i=1}^n \epsilon_i = \sum_{i=1}^n y_i - (\beta_0 + \beta x_i)^2 \quad (9)$$

The objective is to find a model such that it maximizes the *Adjusted-R2* value while keeping *Cp*, *BIC* and *RSS* values low. In RStudio we call the following method to get all possible models. We set *nvmax* to 10 to include all numeric variables of the dataset.

```
#install.packages("leaps")
library(leaps)
models <- regsubsets(max03~., data=df_num, nvmax = 10)
models.summary <- summary(models)
models.summary
```

```
## Subset selection object
## Call: regsubsets.formula(max03 ~ ., data = df_num, nvmax = 10)
## 10 Variables (and intercept)
##           Forced in Forced out
## T9           FALSE      FALSE
## T12          FALSE      FALSE
## T15          FALSE      FALSE
## Ne9          FALSE      FALSE
## Ne12         FALSE      FALSE
## Ne15         FALSE      FALSE
## Wx9          FALSE      FALSE
## Wx12         FALSE      FALSE
## Wx15         FALSE      FALSE
## max03y       FALSE      FALSE
## 1 subsets of each size up to 10
```

```
## Selection Algorithm: exhaustive
##           T9  T12 T15 Ne9 Ne12 Ne15 Wx9 Wx12 Wx15 max03y
## 1  ( 1 )  " " "*" " " " " " " " " " " " " " "
## 2  ( 1 )  " " "*" " " " " " " " " " " " " "*"
## 3  ( 1 )  " " "*" " " "*" " " " " " " " " " " "*"
## 4  ( 1 )  " " "*" " " "*" " " " " "*" " " " " " "*"
## 5  ( 1 )  " " "*" " " "*" " " " " "*" " " "*" "*"
## 6  ( 1 )  " " "*" "*" "*" " " " " "*" " " "*" "*"
## 7  ( 1 )  " " "*" "*" "*" "*" " " "*" " " "*" "*"
## 8  ( 1 )  " " "*" "*" "*" "*" "*" "*" " " "*" "*"
## 9  ( 1 )  " " "*" "*" "*" "*" "*" "*" "*" "*" "*"
## 10 ( 1 )  "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "
```

The below command give us the R-squared value for all possible models with up to 10 predictors. As expected, every time a new variable is included to the model the value gets higher. The model tends to learn the train set really well and it fails to generalize on new unseen data.

```
models.summary$rsq
```

```
## [1] 0.6150674 0.7012408 0.7519764 0.7622198 0.7630603 0.7635768 0.7637610
## [8] 0.7638390 0.7638407 0.7638413
```

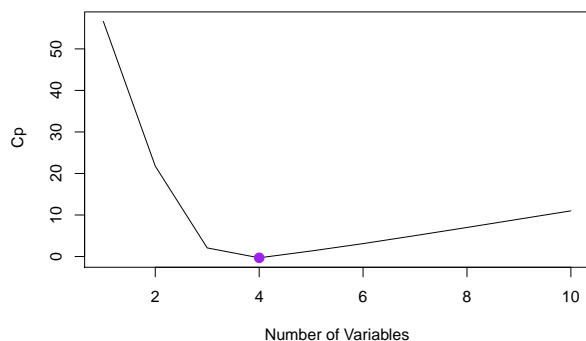
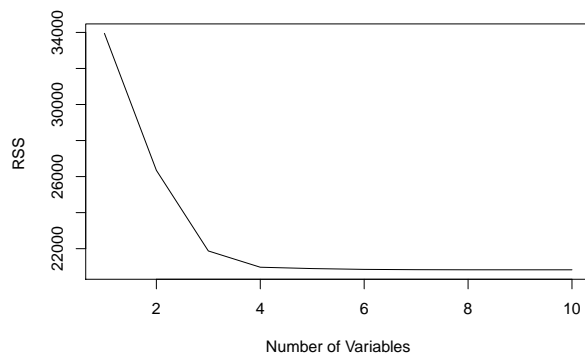
Furthermore, we plot the important metrics we defined previously.

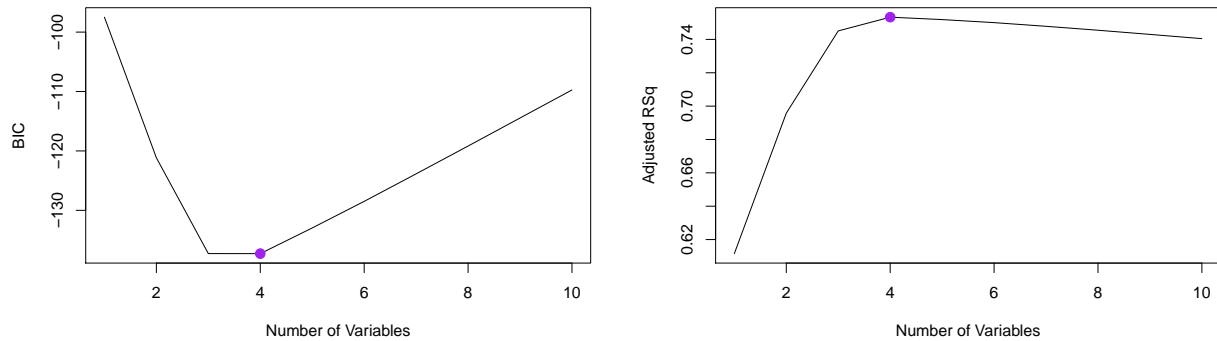
```
plot(models.summary$rss , xlab ="Number of Variables", ylab ="RSS ",type ="l")

plot(models.summary$cp ,xlab =" Number of Variables ", ylab =" Cp",type="l")
cp.min <- which.min(models.summary$cp)
points (cp.min, models.summary$cp[cp.min] , col ="purple ", cex =2, pch =20)

plot(models.summary$bic , xlab =" Number of Variables ", ylab =" BIC ",type="l")
bic.min <- which.min (models.summary$bic )
points (bic.min, models.summary$bic[bic.min] , col ="purple ", cex =2, pch =20)

plot(models.summary$adjr2 ,xlab ="Number of Variables",ylab ="Adjusted RSq", type ="l")
adjr2.max <- which.max (models.summary$adjr2)
points (adjr2.max, models.summary$adjr2[adjr2.max] , col ="purple ", cex =2, pch =20)
```





We see, based on given dataset the more suitable model should consist of 4 predictors. From the *models.summary* we find that *T12*, *Ne9*, *Wx9* and *maxO3y* are the variables that explain the data in the best possible way while the generalization aspect of the model is strong.

```
best.model <- lm(maxO3 ~ T12 + Ne9 + Wx9 + maxO3y, data=df)
```

Finally, we test our model with *maxO3=70*, *T12=19*, *Ne9=8*, *Wx9=2.05*.

```
new.data = data.frame(T12=19, Ne9=8, Wx9=2.05, maxO3y=70)
predict(best.model, new.data, interval='confidence')
```

```
##          fit      lwr      upr
## 1 72.51437 65.16834 79.8604
```

According to our model the predicted maximum ozone level is 72.5 with 95% confidence that the value lies between 65.1 and 79.8.