Air Dataset

Lakshita

2025-11-10

Contents

1 Loading Libraries

```
library(ggplot2)
library(maps)
library(ggrepel)
library(tidyverse)
library(dplyr)
library(tidyr)
library(corrplot)
library(gridExtra)
library(plotly)
library(factoextra)
library(psych)
library(GGally)
library(webshot2)
```

2 Loading data

```
air=read.csv('indian_weather_data.csv')
#head(air)

# making numerical df
num_df=air[c("lat","lon","temperature","weather_code","co","no2","o3","so2","pm2_5","pm10","wind_speed"
num_df=data.frame(num_df)

# making categorical df
cat_df=air[c("city","sunrise","sunset","moonrise","moonset","wind_dir")]
```

3 Data Handling

3.1 Checking missing value

```
colSums(is.na(air))
```

```
##         city          lat          lon  temperature weather_code      sunrise
##            0            0            0            0            0            0
##       sunset     moonrise      moonset           co          no2           o3
##           0            0            0            0            0            0
##          so2        pm2_5         pm10   wind_speed  wind_degree     wind_dir
##            0            0            0            0            0            0
##     pressure       precip     humidity   cloudcover    feelslike     uv_index
##            0            0            0            0            0            0
##   visibility
##            0
```

There are no missing value

4 PCA: Principal Component Analysis

```
# Normalising data
scaled_df<-scale(num_df)

# Applying PCA
data.pca<-princomp(scaled_df)
summary(data.pca)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3     Comp.4     Comp.5
## Standard deviation     2.5335546 1.7258433 1.5140987 1.19297081 1.02755715
## Proportion of Variance 0.3827547 0.1776081 0.1366999 0.08486323 0.06296104
## Cumulative Proportion  0.3827547 0.5603627 0.6970626 0.78192587 0.84488691
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     0.86033620 0.82063083 0.54083074 0.52940705 0.45271033
## Proportion of Variance 0.04413634 0.04015648 0.01744145 0.01671242 0.01222083
## Cumulative Proportion  0.88902325 0.92917973 0.94662118 0.96333361 0.97555444
##                            Comp.11     Comp.12     Comp.13     Comp.14
## Standard deviation     0.376134960 0.338575912 0.251403178 0.227654813
## Proportion of Variance 0.008436209 0.006835528 0.003768786 0.003090392
## Cumulative Proportion  0.983990646 0.990826174 0.994594960 0.997685352
##                           Comp.15      Comp.16      Comp.17
## Standard deviation     0.15902327 0.1152045144 1.602468e-02
## Proportion of Variance 0.00150793 0.0007914053 1.531225e-05
## Cumulative Proportion  0.99919328 0.9999846878 1.000000e+00
```

- Performing PCA requires scaling of data, princomp function was used for this purpose.
- The first five components have eigen value greater than 1. Thus, first five components retain **84% variance** of data. The first 2 components retain maximum variance 56%.
- going ahead we will check the factor loadings of first 5 components for analysis.

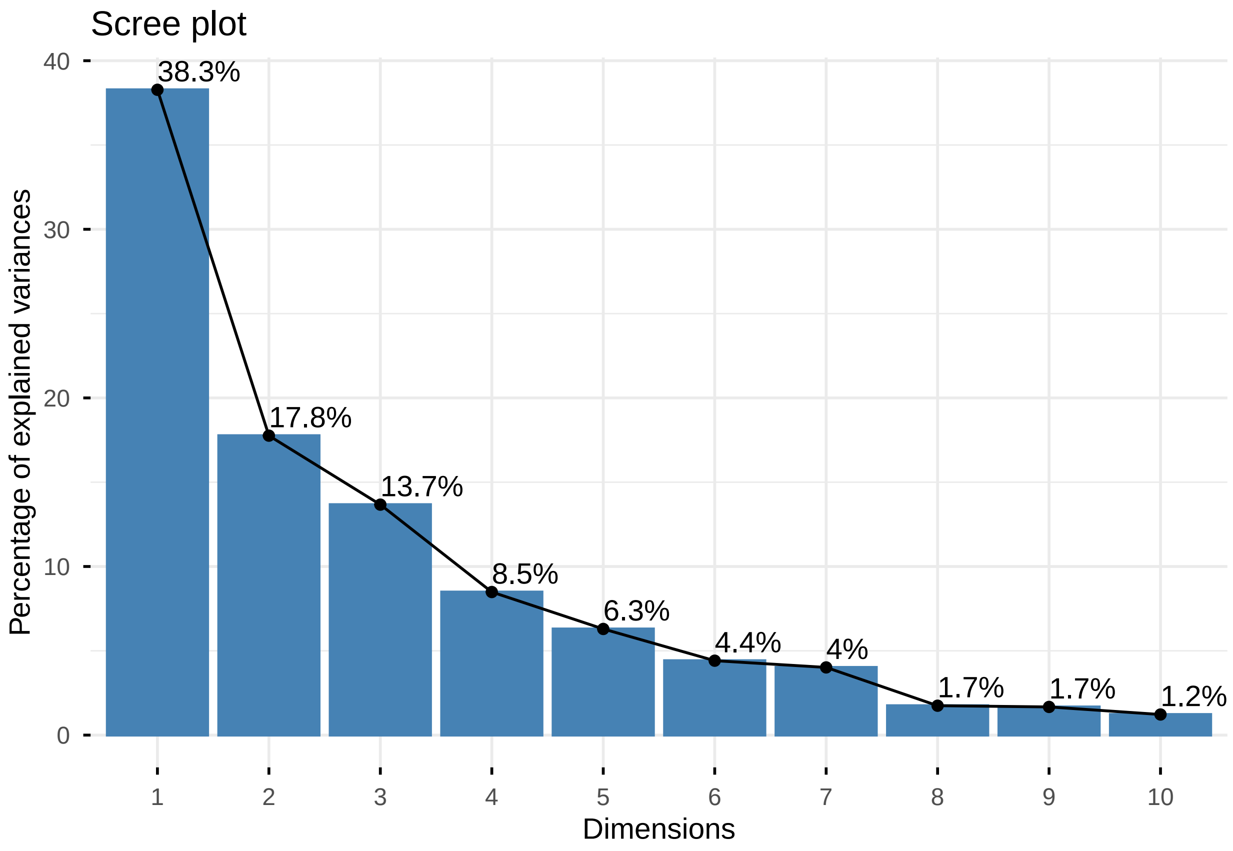
4.1 Extracting Loadings

```
# extracting the loadings
data.pca$loadings[,1:5]
```

```
##                    Comp.1        Comp.2       Comp.3       Comp.4      Comp.5
## lat           0.298693929  0.1741996935  0.264271832  0.039111452  0.06567887
## lon           0.121570340  0.3737757341 -0.288322711  0.074254450 -0.16502060
## temperature  -0.276398614 -0.3920491370 -0.016168420 -0.028792546 -0.03397166
## weather_code  0.151077988 -0.0485232318 -0.433015634 -0.267606169 -0.25553804
## co            0.348664946 -0.1977444783  0.008432181  0.029817564 -0.12049839
## no2           0.119454953 -0.1702609241 -0.073198459  0.643911842 -0.31616965
## o3            0.238100059 -0.3369639851  0.008192443 -0.078741718  0.32803458
## so2           0.311701586 -0.2264924355  0.020555419  0.207170883  0.01272117
## pm2_5         0.348722657 -0.2133589498  0.011597373 -0.006125823  0.06623659
## pm10          0.347364379 -0.2109712547  0.004900234 -0.012363201  0.06874963
## wind_speed   -0.208144702 -0.0008543208 -0.220755352  0.555330430 -0.11612031
## wind_degree   0.007514336  0.1053899469 -0.099366795  0.344485505  0.75038979
## pressure      0.262516331  0.3453643477  0.019249046  0.006053016 -0.03103226
## humidity      0.134890612  0.1227875390 -0.533724406 -0.008214219  0.08728228
## cloudcover    0.226067461 -0.0171820487  0.313975205  0.054212352 -0.28846463
## feelslike    -0.262454176 -0.4108471857 -0.053851895 -0.016649897 -0.02914395
## visibility   -0.119776506  0.1907126064  0.455927384  0.159091429 -0.03773750
```

4.2 Visualization of Principal components

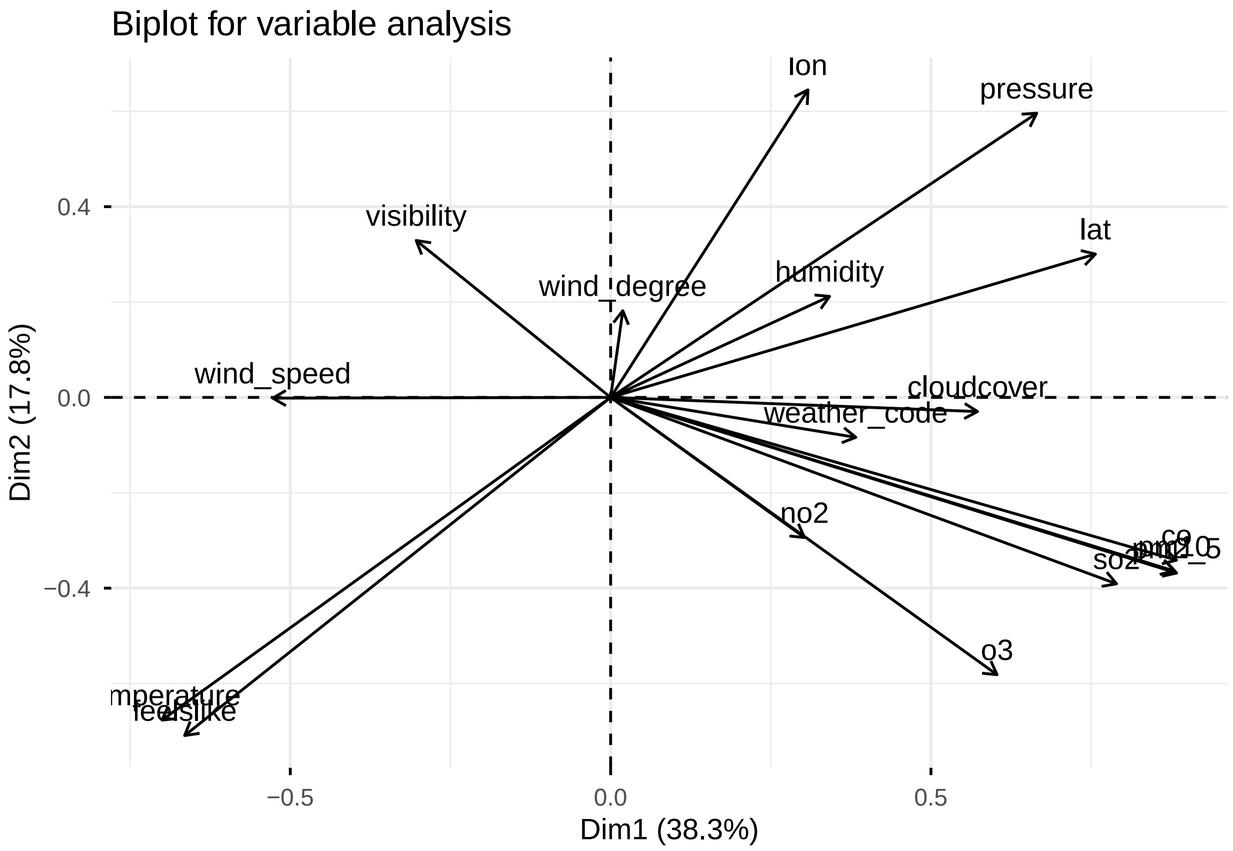
```
fviz_eig(data.pca,addlabels = TRUE)
```



4.3 Biplot

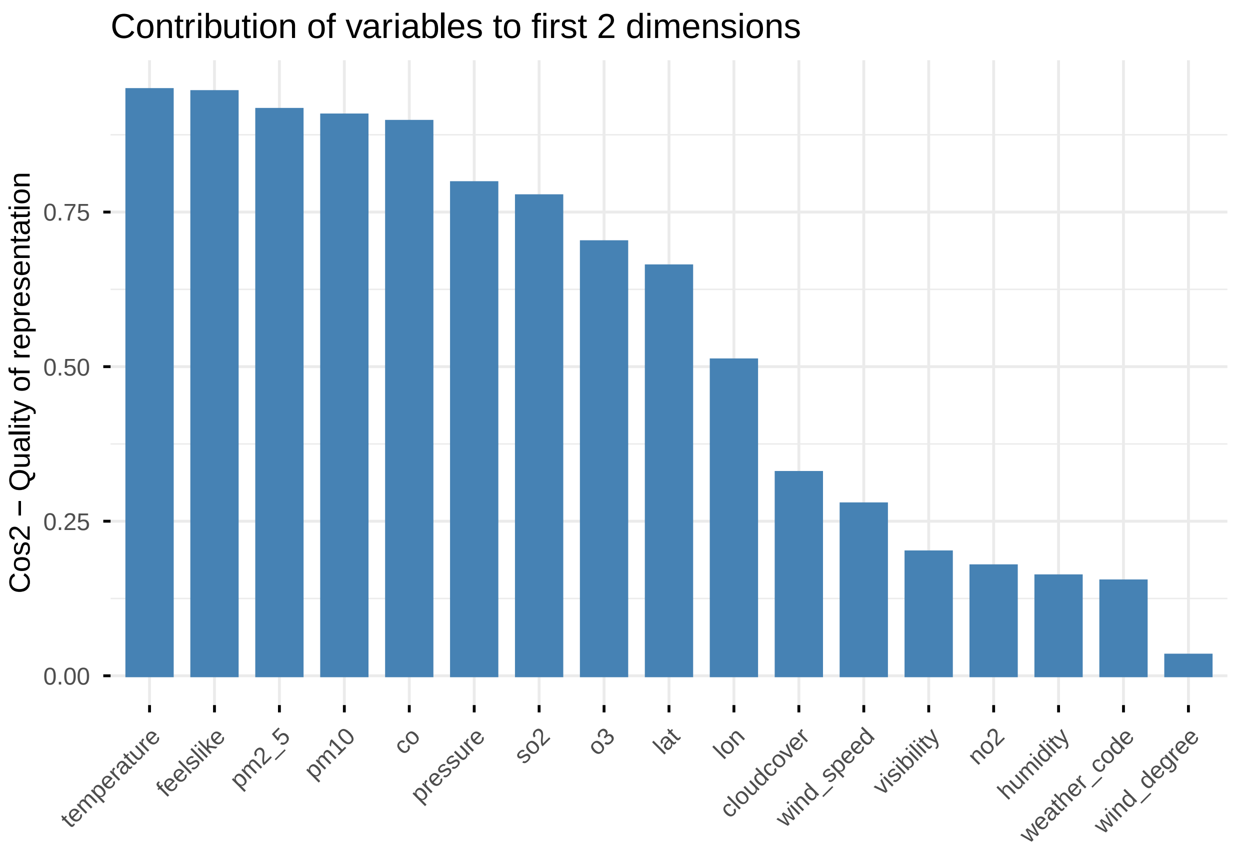
```
fviz_pca_var(data.pca,col.var="black")+
  ggtitle("Biplot for variable analysis")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## i The deprecated feature was likely used in the ggpubr package.
##   Please report the issue at <https://github.com/kassambara/ggpubr/issues>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



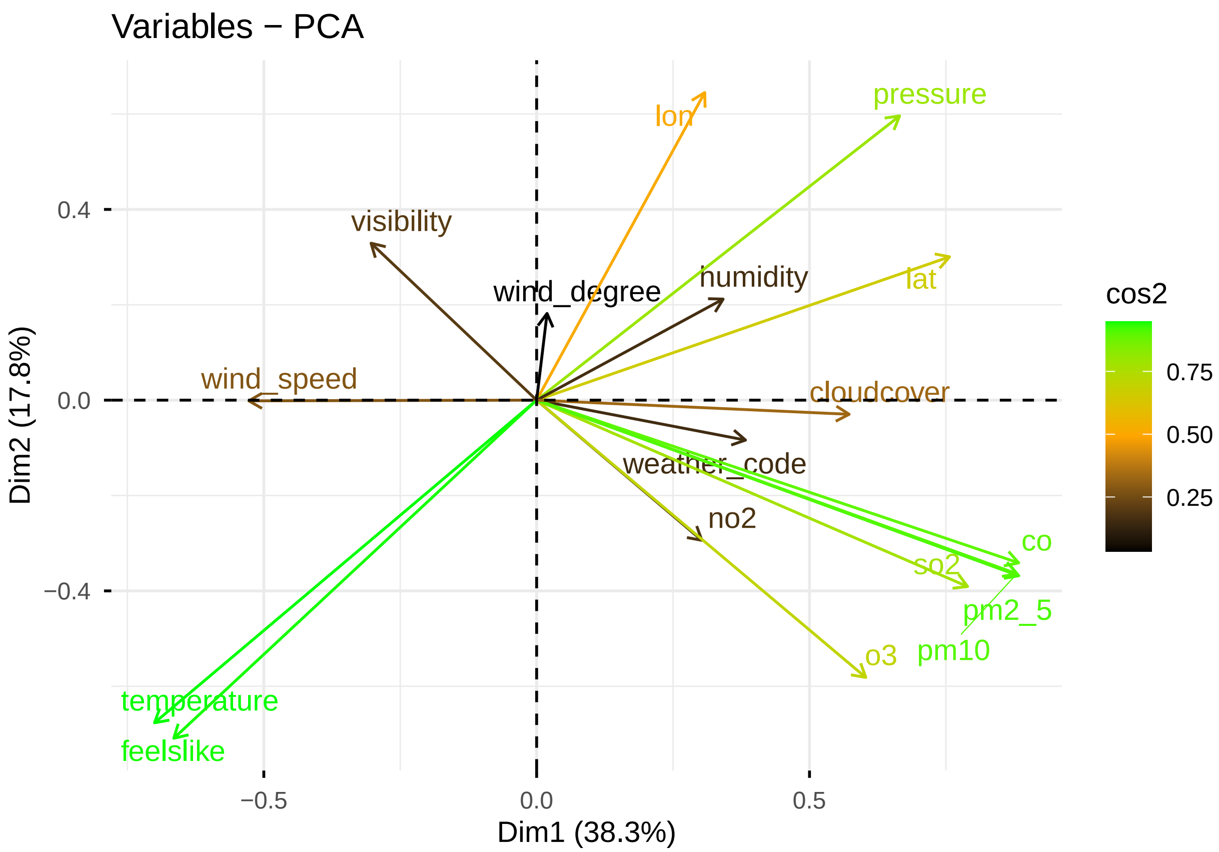
## Checking contribution of each variable

```
fviz_cos2(data.pca,choice="var",axes=1:2)+
  ggtitle("Contribution of variables to first 2 dimensions")
```



## Biplot combined with cos2

```
fviz_pca_var(data.pca,col.var="cos2",
             gradient.cols=c("black","orange","green"),
             repel=TRUE)
```



4.4 Conclusion

1. **Component 1**: The variables pollutants(co2, so2, o3, pm2_5, pm10) and weather variables such as pressure humidity and cloud cover positively contribute to first dimension. The variables like wind_speed , temperature , feelslike negatively contribute to first dimension.This indicates that pollutants dominate this dimension with higher loading.**We can observe that as wind speed decreases and temperature increases small particles settle in that area leading to higher pollutants. They have inverse relationship.**

2. **Component 2:** This is weather dominant component. pollutants negatively contribute to it while weather degree, humidity, pressure a nd visibility affect it positively. **With longitude the pressure in the area increases.Visibility also contributes to this dimension significantly. As pollutants decrease visibility increases.**

3. The first 5 principal components retain **84% variance in data**. But first 2 components constitute most variance.

4. All pollutants are clustered together in biplot. Thus are positively correlated and these variables ar well reperesented. Variables feelslike , temprature , lat, long and pressure are also well represented.

5 Factor Analysis

5.1 KMO and Bartlett's Test of Sphericity

```r
# making a correlation matrix
correlation_matrix<-cor(num_df)

#KMO Test
kmo_result <- KMO(correlation_matrix)
print(kmo_result)
```

```
## Kaiser-Meyer-Olkin factor adequacy
## Call: KMO(r = correlation_matrix)
## Overall MSA =  0.7
## MSA for each item = 
##          lat          lon  temperature weather_code           co          no2 
##         0.80         0.68         0.72         0.60         0.83         0.33 
##           o3          so2        pm2_5         pm10   wind_speed  wind_degree 
##         0.82         0.77         0.75         0.75         0.57         0.33 
##     pressure     humidity   cloudcover    feelslike   visibility 
##         0.76         0.51         0.76         0.76         0.49
```

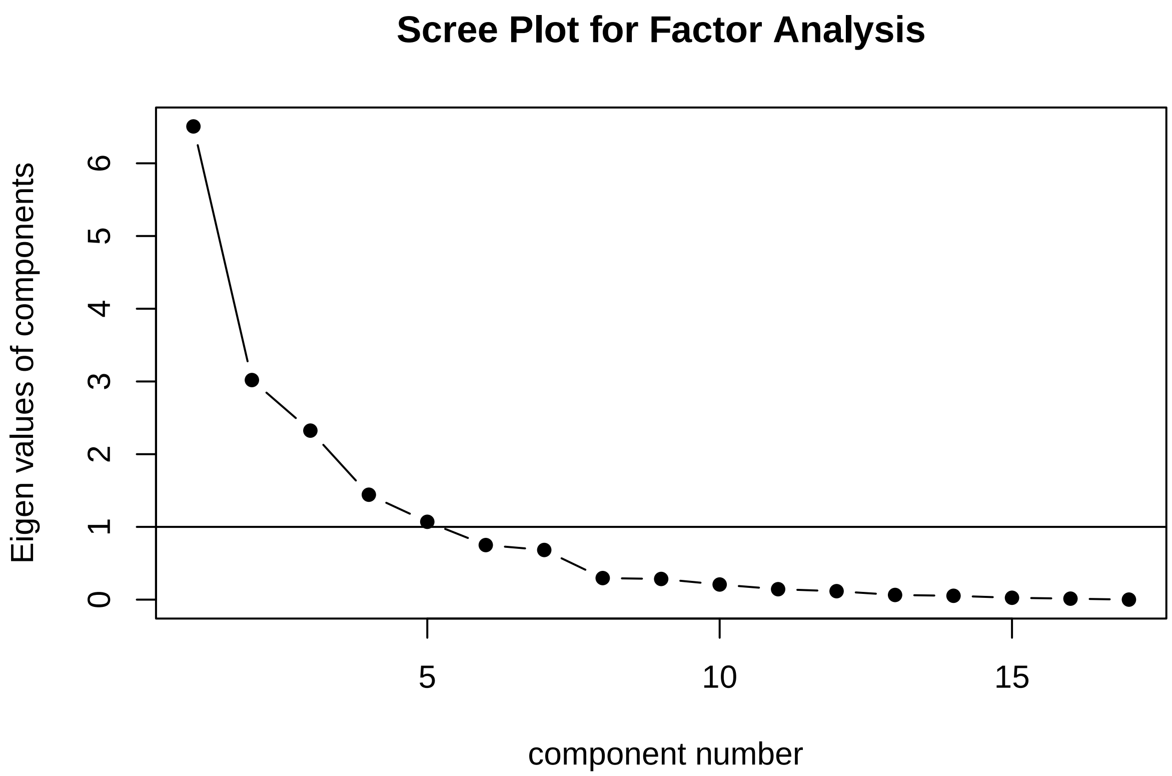
```r
# Bartlett's Test of Sphericity
cortest.bartlett(correlation_matrix, n = nrow(num_df))
```

```
## $chisq
## [1] 1760.473
## 
## $p.value
## [1] 3.011205e-280
## 
## $df
## [1] 136
```

1. **KMO Test :** we can observe the overall MSA value is greater than 0.69 indicating that correlation matrix is not identity matrix.
2. **Bartlett's Test of Sphericity :** We can Observe that the p-value is 1.449223e-293 <0.05 indicating that the data is suitable for factor analysis

5.2 Deciding number of factors

```r
scree(num_df, factors = FALSE, pc = TRUE,
      main = "Scree Plot for Factor Analysis")
```



Thus, 4 factors can be retained to explain the underlying structure as eigen value is greater than 1.

```
fa<- fa(num_df, nfactors=4,rotate="varimax",scores="regression")
```

```
## In factor.stats, I could not find the RMSEA upper bound . Sorry about that
```

```
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :
## The estimated weights for the factor scores are probably incorrect.  Try a
## different factor score estimation method.
```

```
## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An
## ultra-Heywood case was detected.  Examine the results carefully
```

```
fa
```

```
## Factor Analysis using method =  minres
## Call: fa(r = num_df, nfactors = 4, rotate = "varimax", scores = "regression")
## Standardized loadings (pattern matrix) based upon correlation matrix
##                MR1   MR2   MR3   MR4    h2     u2 com
## lat           0.51  0.69 -0.24 -0.13 0.804  0.1956 2.2
## lon          -0.15  0.62  0.39  0.20 0.593  0.4067 2.1
## temperature  -0.19 -0.98 -0.09  0.01 0.997  0.0026 1.1
## weather_code  0.20  0.09  0.68 -0.07 0.519  0.4809 1.2
## co            0.91  0.21  0.19 -0.02 0.913  0.0866 1.2
## no2           0.45 -0.03 -0.03  0.67 0.661  0.3395 1.8
## o3           0.76 -0.13  0.18 -0.17 0.644  0.3555 1.3
## so2          0.88  0.13  0.09  0.16 0.819  0.1811 1.1
## pm2_5        0.93  0.18  0.21 -0.09 0.947  0.0534 1.2
## pm10         0.92  0.18  0.22 -0.10 0.933  0.0671 1.2
## wind_speed  -0.40 -0.28  0.01  0.76 0.818  0.1825 1.8
## wind_degree -0.06  0.12  0.05  0.18 0.051  0.9485 2.2
## pressure     0.21  0.84  0.08 -0.02 0.749  0.2509 1.1
## humidity     0.01  0.30  0.84  0.19 0.833  0.1666 1.4
## cloudcover   0.52  0.31 -0.27 -0.08 0.447  0.5533 2.3
## feelslike   -0.15 -0.99 -0.02  0.04 1.003 -0.0032 1.0
## visibility  -0.27  0.13 -0.68 -0.01 0.558  0.4419 1.4
##
##                        MR1  MR2  MR3  MR4
## SS loadings           5.01 3.93 2.10 1.24
## Proportion Var        0.29 0.23 0.12 0.07
## Cumulative Var        0.29 0.53 0.65 0.72
## Proportion Explained  0.41 0.32 0.17 0.10
## Cumulative Proportion 0.41 0.73 0.90 1.00
##
## Mean item complexity =  1.5
## Test of the hypothesis that 4 factors are sufficient.
##
## df null model =  136  with the objective function =  26.47 with Chi Square =  1760.47
## df of  the model are 74  and the objective function was  627.51
##
## The root mean square of the residuals (RMSR) is  0.05
## The df corrected root mean square of the residuals is  0.07
##
## The harmonic n.obs is  74 with the empirical chi square  49.28  with prob <  0.99
## The total n.obs was  74  with Likelihood Chi Square =  40055.82  with prob <  0
##
## Tucker Lewis Index of factoring reliability =  -46.288
## RMSEA index =  2.702  and the 90 % confidence intervals are  2.698 NA
## BIC =  39737.32
## Fit based upon off diagonal values = 0.98
```

Factors are:

1. Factor 1: positive effect: co, o3, so2, pm2_5,pm10,lat negative effect: lon, temprature
2. Factor 2: positive effect: lat,lon,pressure negative effect: temperature,feelslike
3. Factor 3: positive effect: wether_code,humidity negative effect: visibility
4. Factor 4: positive effect: no2,wind_speed

- Factor Analysis is used for identifying the **underlying structure of the data**. It helps in reducing variable and these obtained factors can be effectively used for EDA.
- The First Factor can be named as**pollutants**, Second Factor as **geographic_cond**, Third Factor is **weather_cond** , Fourth can be named as **ozone** since higher wind speed decreases no2 concentration.
- All the variables have high commonality that means they are **well represented by the factors.**

6 Data Engineering

6.1 Adding FA Score / Factors for EDA

```r
# Storing FA Scores as df
fa_scores<-as.data.frame(fa$scores)

# renaming column names
colnames(fa_scores)=c("pollutants","geographic_cond","weather_cond","ozone")

# concatenating 2 dfs air and fa_scores
df<-cbind(air,fa_scores)
#head(df)
```

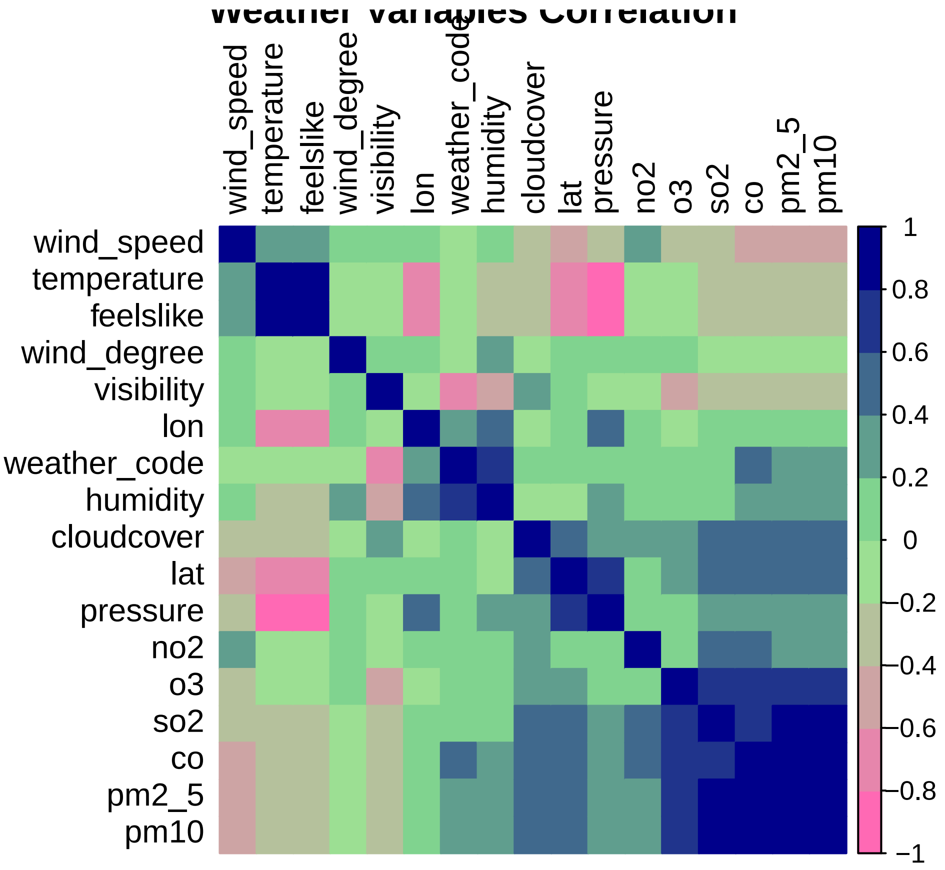
6.2 Converting Weather code and visibility to categorical variables

```r
df<-df %>%
  mutate(weather_cat=case_when(
    weather_code==113~"sunny",
    weather_code==122~"partly cloudly",
    weather_code==143~"mist",
    weather_code==116~"moderate rain",
    weather_code==119~"showers",
    weather_code==248~"fog",
    weather_code==176~"moderate rain",
  )) %>%
  mutate(visibility_cat=case_when(
    visibility<1~"very poor",
    between(visibility,1,3) ~"poor",
    between(visibility,3,5) ~"moderate",
    visibility>5 ~"good"
  )) %>%
  mutate(parts_of_India = case_when(
    between(lat, 28, 37.6) & between(lon, 68.7, 97.25) ~ "North",
    between(lat, 15, 28) & between(lon, 68, 78) ~ "West",
    between(lat, 20, 28) & between(lon, 83, 97.25) ~ "East",
    lat < 20 & between(lon, 74, 84) ~ "South",
    between(lat, 18, 26) & between(lon, 74, 85) ~ "Central",
    between(lat, 22, 28) & between(lon, 89, 97.25) ~ "Northeast",
    TRUE ~ "Other Region"
  ))
```

7 Exploratory Data Analysis

7.1 Correlation among variables

```
# Calculate correlation with pairwise complete observations
weather_cor <- cor(num_df,
                   use = "pairwise.complete.obs")

c_color<- colorRampPalette(c("hotpink", "lightgreen","darkblue"))

corrplot(weather_cor,
         method = "color",
         title = "Weather Variables Correlation",
         order="hclust",
         col=c_color(10),
         tl.col="black"
         )
```

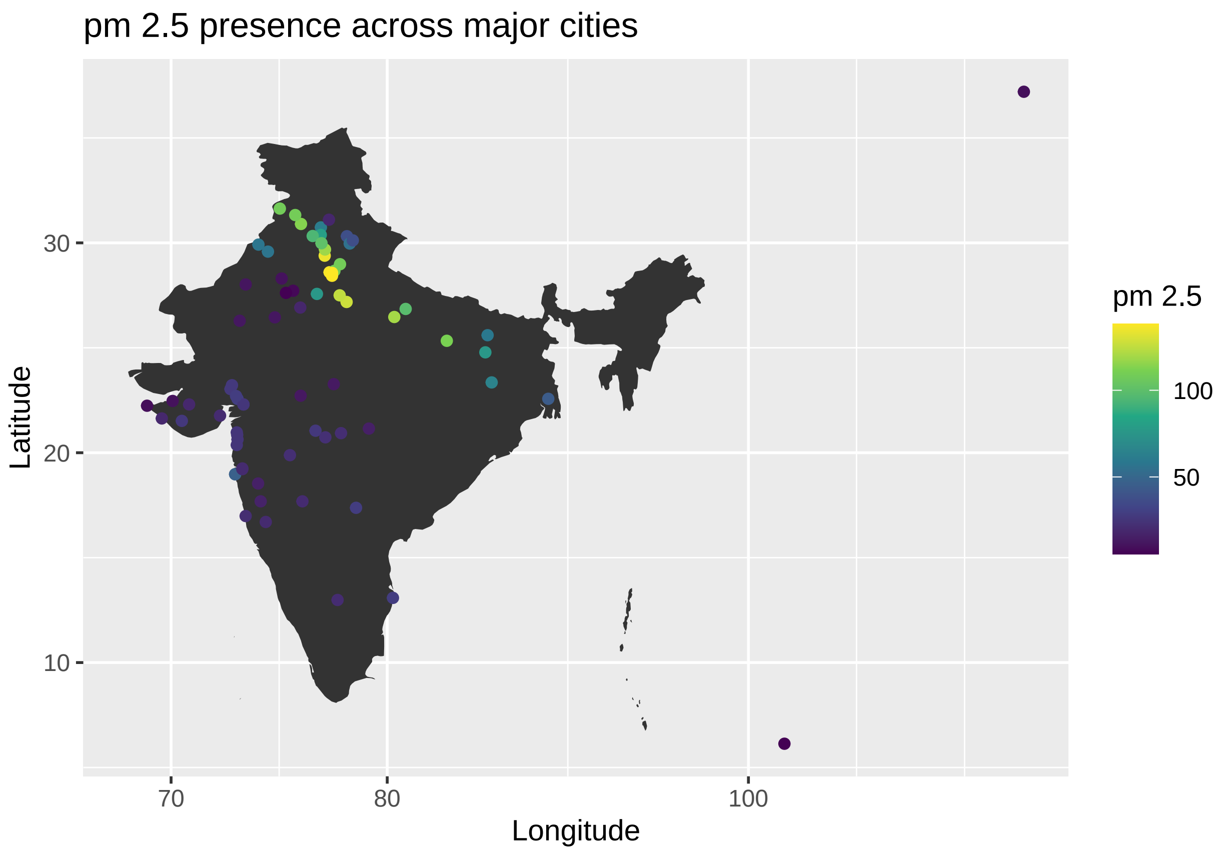


* Positive Correlation: pollutants are highly correlated such as pm2_5, pm_10, co, so2 , no2 is moderately correlated, feelslike and temperature are highly correlated.

- Moderate Correlation (positive and negative): We can see moderate correlation between co2 and weather code,longitude with temperature and feelslike,latitude with temprature and feelslike
- Negative Correlation:pressure with feelslike and temprature

7.2 pm 2.5 according to latitude and longitude values

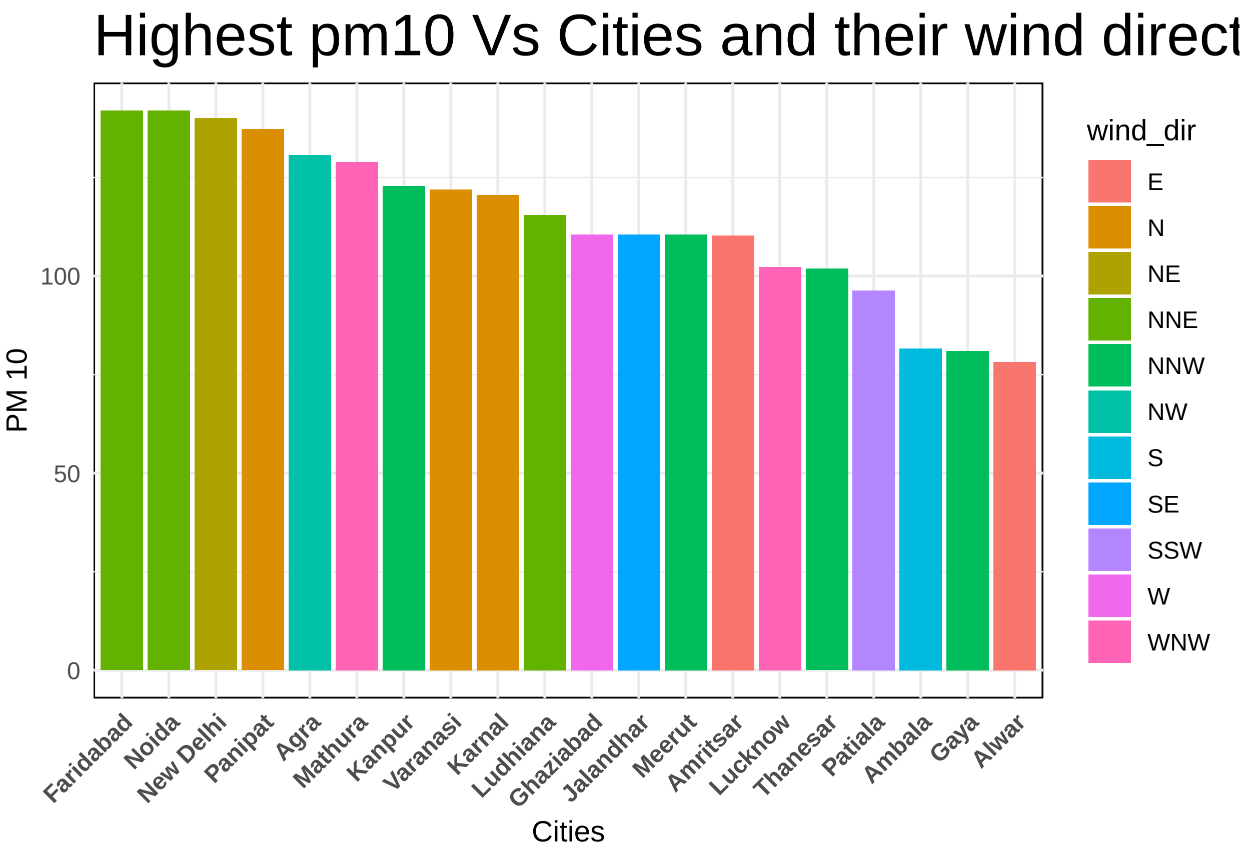
```
world<-map_data("world")

#getting the map for india
india<-subset(world,region=="India")


ggplot()+
   geom_polygon(data=india,
                aes(x=long,y=lat,group=group))+
  geom_point(data=df,
             aes(x=lon,y=lat,color=pm2_5))+
  scale_color_continuous(
    type = "viridis",
    name = "pm 2.5"
  )+
  scale_x_log10()+
  labs(
    title="pm 2.5 presence across major cities",
    x="Longitude",
    y="Latitude"
  )
```



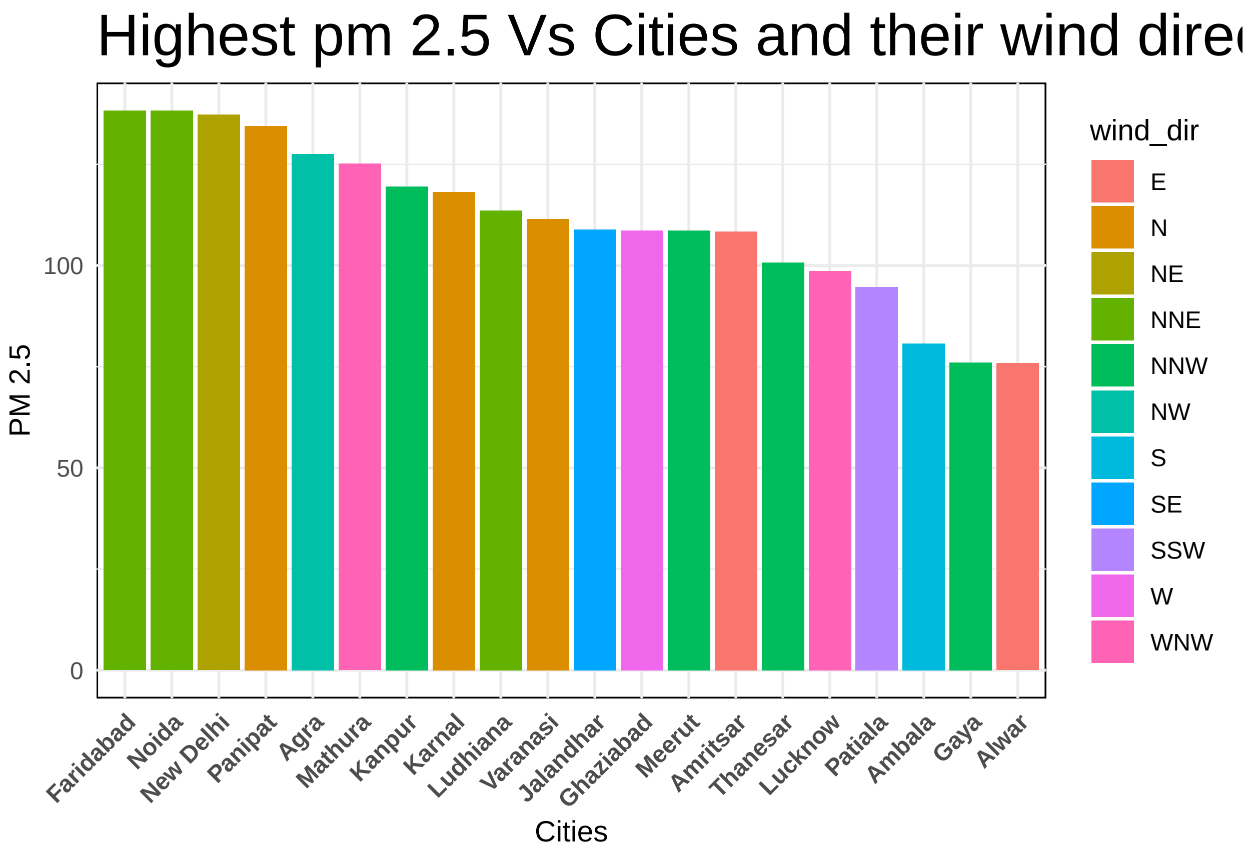
```r
# setting theme for all plots
set_theme(theme_minimal()+
            theme(
                plot.title=
                  element_text(
                    size=rel(2)),

                panel.background =
                  element_rect(color="black"),

              ))
```

7.3 Top 20 cities with worst air quality pm10 and pm2

```r
# pm10
df %>%
  arrange(desc(pm10)) %>%
  select(city,pm10,wind_dir) %>%
  slice_head(n=20) %>%
  ggplot(
    aes(x=reorder(city,-pm10),y=pm10,fill=wind_dir)
  )+
  labs(
    title="Highest pm10 Vs Cities and their wind direction",
    x="Cities",
    y="PM 10"
  ) +
  geom_bar(stat="identity")+
  theme(
      axis.text.x =
                  element_text(angle=45,
                                hjust=1,
                               face="bold")
  )
```



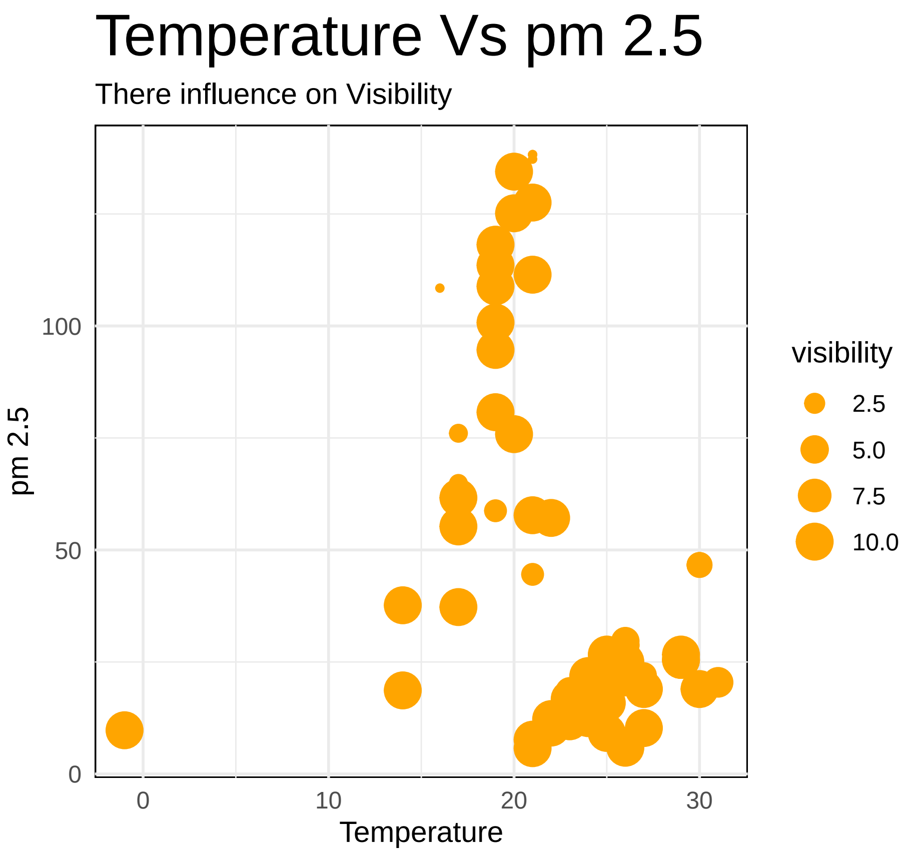
```
df %>%
  arrange(desc(pm2_5)) %>%
  select(city,pm2_5,wind_dir) %>%
  slice_head(n=20) %>%
  ggplot(
    aes(x=reorder(city,-pm2_5),y=pm2_5,fill=wind_dir)
  )+
  geom_bar(stat="identity")+
  labs(
    title="Highest pm 2.5 Vs Cities and their wind direction",
    x="Cities",
    y="PM 2.5"
  )+
  theme(
      axis.text.x =
                element_text(angle=45,
                              hjust=1,
                             face="bold")
  )
```



#One-on-one relationships between two continuous variables

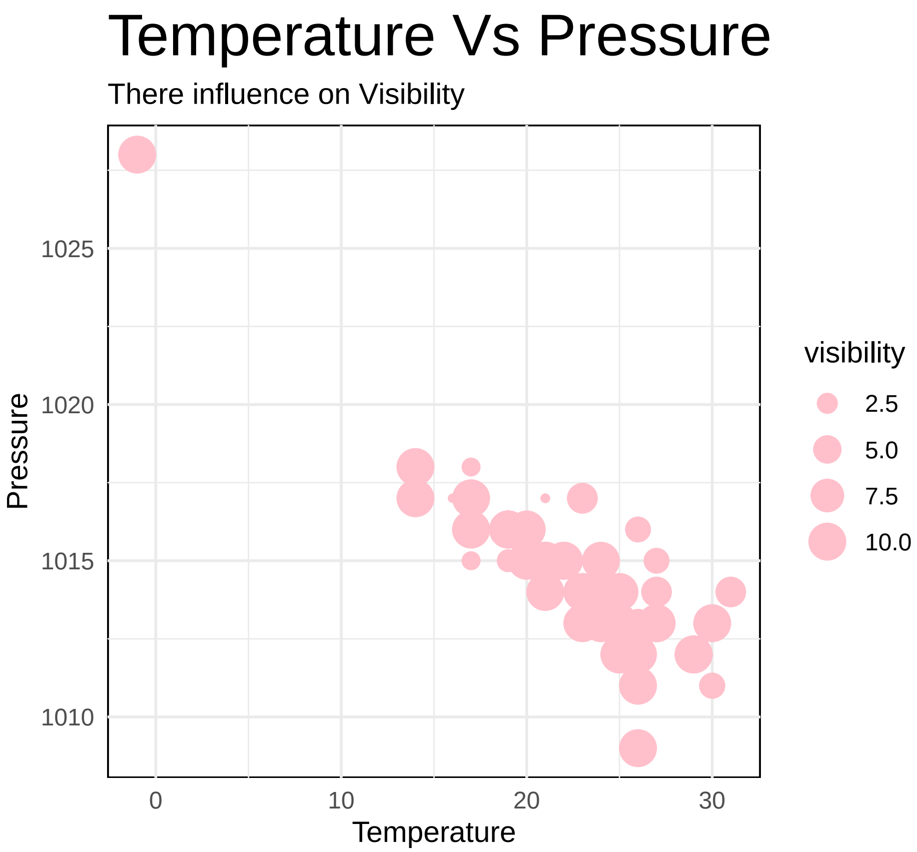
7.4 Temperature vs PM2.5 levels

```
# temperature Vs pm2.5 levels
plot1<-df %>%
  ggplot(aes(temperature,pm2_5,size=visibility))+
  geom_point(color="orange")+
  labs(
    title="Temperature Vs pm 2.5",
    subtitle="There influence on Visibility",
    x="Temperature",
    y="pm 2.5"
  )+
  theme(
    aspect.ratio = 1
  )
plot1
```



7.5 Temptrature Vs Pressure

```
# temperature Vs pm2.5 levels
plot2<-df %>%
  ggplot(aes(temperature,pressure,size=visibility))+
  geom_point(color="pink")+
  labs(
    title="Temperature Vs Pressure",
    subtitle="There influence on Visibility",
    x="Temperature",
    y="Pressure"
  )+
  theme(
    aspect.ratio = 1
  )

plot2
```



7.6 Observing 3 variables Wind speed , Temprature and pm 2.5 concentration

```
fig <-plot_ly(df, x = ~wind_speed, y = ~temperature, z = ~pm2_5, color = ~city)


fig <- fig %>% add_markers()
fig <- fig %>% layout(
                    title = 'Temperature , Wind Speed and pm 2.5 coded with city',
                    scene =
                    list(xaxis = list(title = 'Wind Speed'),
                     yaxis = list(title = 'Temprature'),
                     zaxis = list(title = 'pm 2.5')))

fig
```

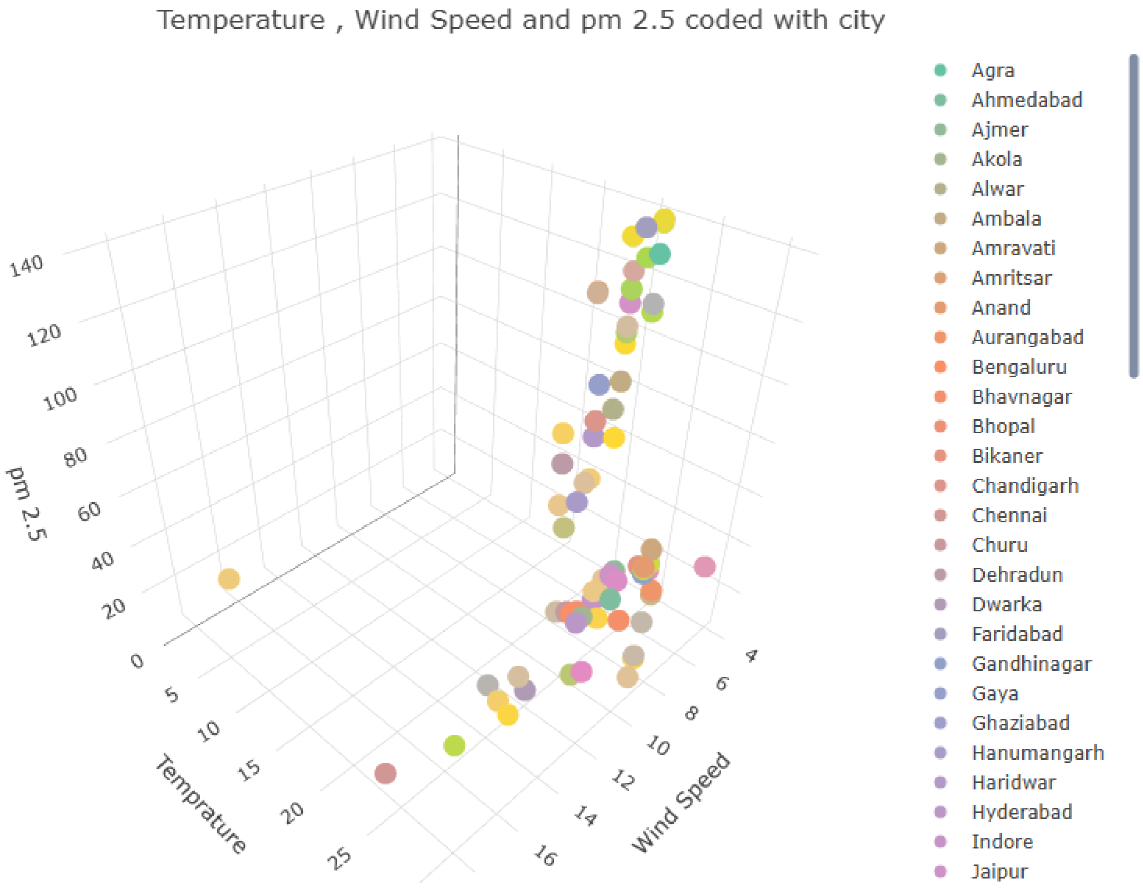
```
## Warning in RColorBrewer::brewer.pal(max(N, 3L), "Set2"): n too large, allowed maximum for palette Se
## Returning the palette you asked for with that many colors
## Warning in RColorBrewer::brewer.pal(max(N, 3L), "Set2"): n too large, allowed maximum for palette Se
## Returning the palette you asked for with that many colors
```

Temperature , Wind Speed and pm 2.5 coded with city

```
htmlwidgets::saveWidget(as_widget(fig), "temp.html")
```

```
## Warning in RColorBrewer::brewer.pal(max(N, 3L), "Set2"): n too large, allowed maximum for palette Se
## Returning the palette you asked for with that many colors
## Warning in RColorBrewer::brewer.pal(max(N, 3L), "Set2"): n too large, allowed maximum for palette Se
## Returning the palette you asked for with that many colors
```

```
webshot2::webshot("temp.html", "plot.png", vwidth = 800, vheight = 600)
```



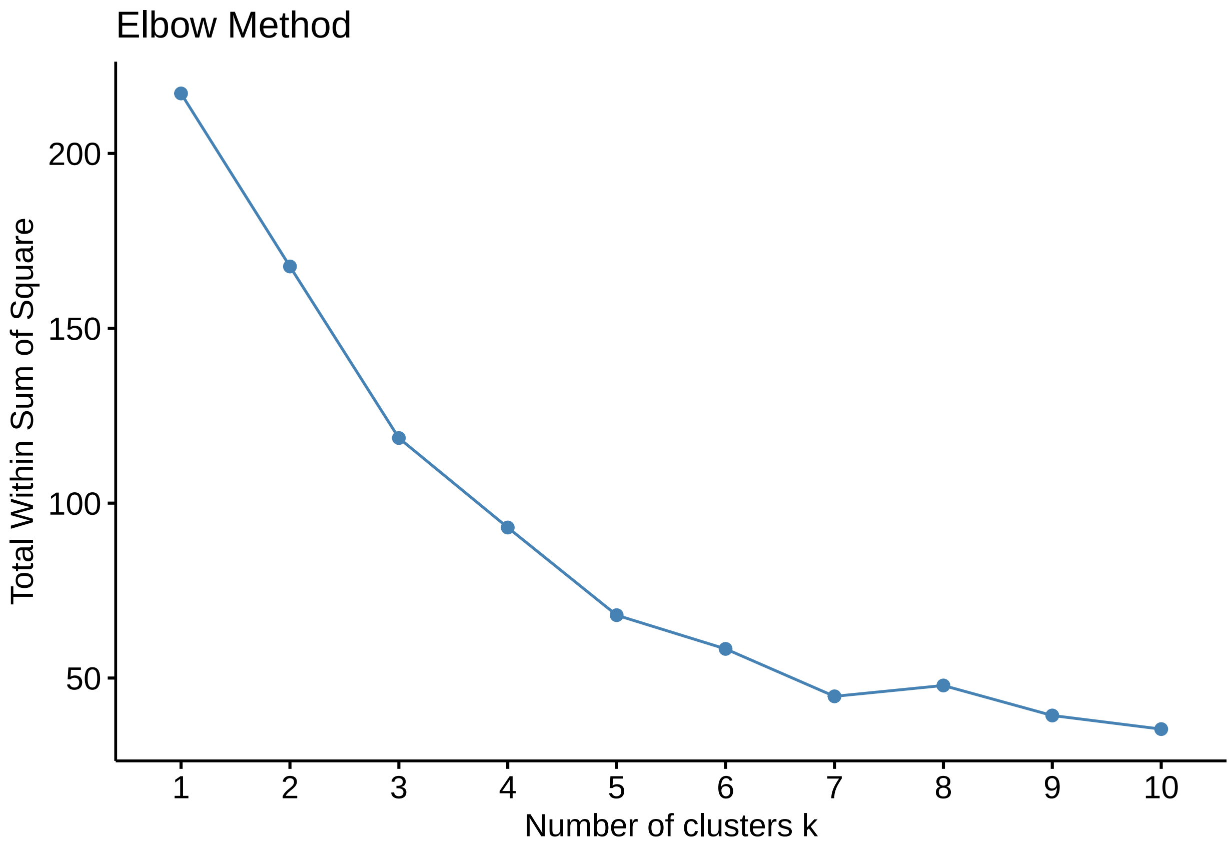
8 Clustering based on 3 Factors retained

8.1 Making dataset for clustering

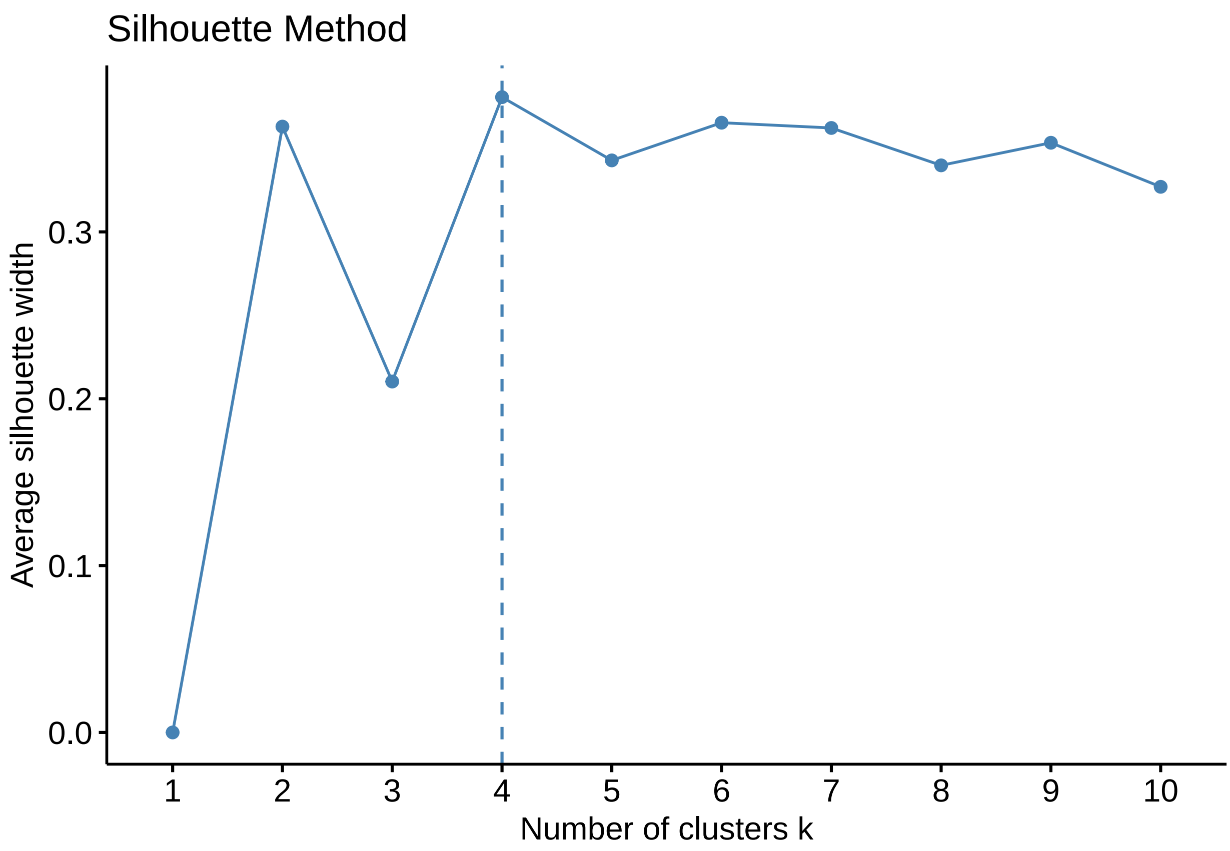
```
# Data for clustering
k_data<-df %>%
  select("city","pollutants","weather_cond","geographic_cond")
```

8.2 Selecting number of clusters

```
fviz_nbclust(k_data[,c("pollutants","weather_cond","geographic_cond")], kmeans, method = "wss") +
  ggtitle("Elbow Method")
```



```
fviz_nbclust(k_data[,c("pollutants","weather_cond","geographic_cond")], kmeans, method = "silhouette") +
  ggtitle("Silhouette Method")
```



## Performing K-means clustering

```r
# Perform k-means clustering (e.g., 4 clusters)
set.seed(123)
kmeans_result <- kmeans(k_data[,c("pollutants","weather_cond","geographic_cond")], centers = 4, nstart
print(kmeans_result)
```

```
## K-means clustering with 4 clusters of sizes 11, 12, 13, 38
##
## Cluster means:
##   pollutants weather_cond geographic_cond
## 1  1.0521595   1.52978703       0.3104511
## 2  1.3119575  -0.79206252       0.3998618
## 3 -0.6659475  -0.62980568       1.2501290
## 4 -0.4910507   0.02275176      -0.6438153
##
## Clustering vector:
##  [1] 1 4 1 4 4 4 4 4 4 3 1 1 4 4 4 1 3 1 4 2 2 1 1 1 1 2 2 2 1 2 2 2 3 3 3 3 4 3
## [39] 3 4 2 3 3 3 2 2 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 2 4 4 4 4 4 4 4 4 4 3 3 4
##
## Within cluster sum of squares by cluster:
## [1] 16.630440  5.361423 34.609670 35.293387
##  (between_SS / total_SS =  57.7 %)
##
## Available components:
##
```

```
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

8.3 Storing Clusters as factor

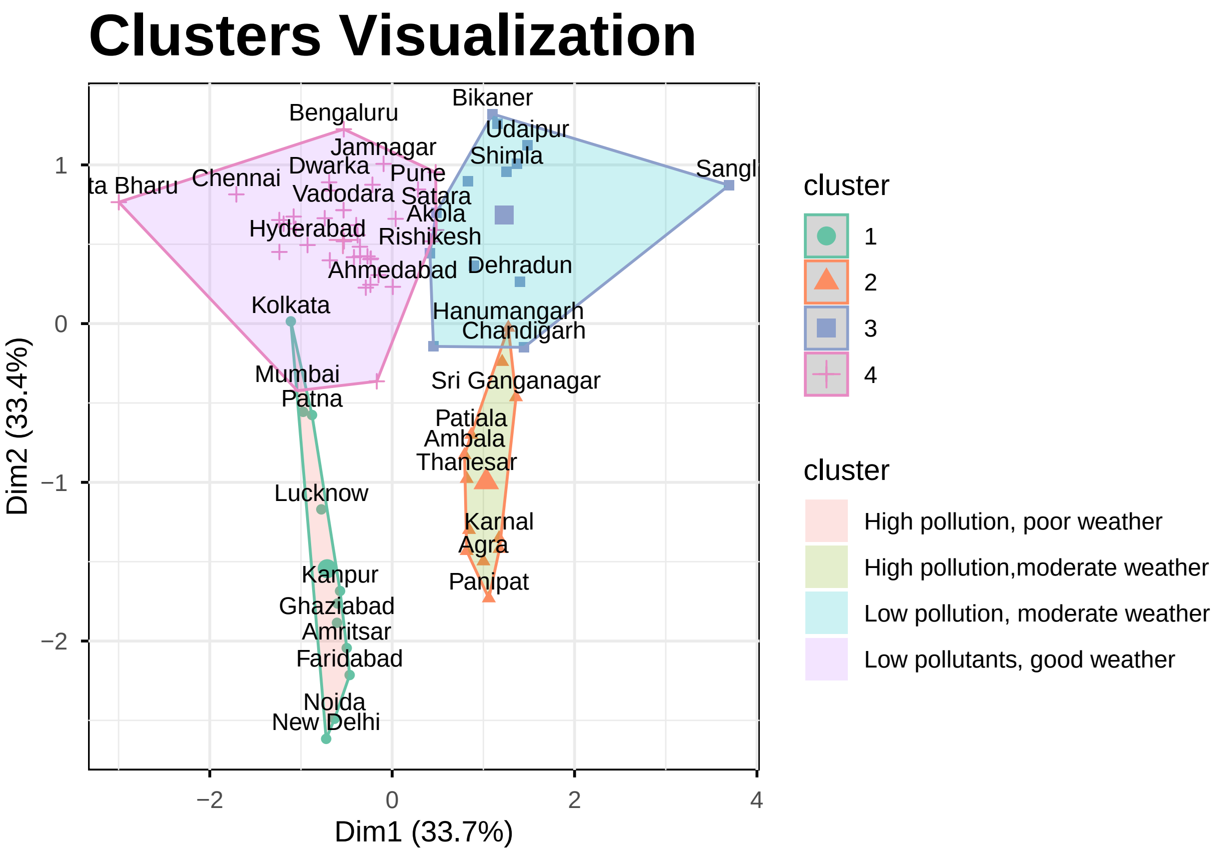
```
k_data$cluster <- as.factor(kmeans_result$cluster)
head(k_data)
```

```
##        city pollutants weather_cond geographic_cond cluster
## 1 New Delhi  2.3898941    1.2332044     -0.27910897       1
## 2    Mumbai  0.7275207   -0.1247146     -1.75305313       4
## 3   Kolkata -0.6789153    1.6993385      0.27044612       1
## 4   Chennai -1.3535364    1.5147861     -0.62542879       4
## 5 Bengaluru -1.4458486    0.3791381     -0.07514414       4
## 6 Hyderabad -0.6457790    0.5358054     -0.64707795       4
```

8.4 Visualizing clusters

```
city_names<-k_data$city
fviz_cluster(kmeans_result, data = k_data[,c("pollutants","weather_cond","geographic_cond")],
             palette = "Set2", ggtheme = theme_minimal(),
             geom = "point") +
  geom_text(aes(label = city_names),
            check_overlap = TRUE,
            size = 3,
            vjust = -0.5)+
 labels(
   title="Clusters Visualization"
 )+
  theme(
    plot.title =
      element_text(face = "bold",
                   size=rel(2)),
    panel.background =
                  element_rect(color="black")
  )+
  scale_fill_discrete(labels = c("High pollution, poor weather", "High pollution,moderate weather", "Lo
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
```


Clusters Visualization
Bikaner
Bengaluru
Udaipur
Jamnagar
Shimla
Sangl
ta Bharu
Chennai
Dwarka
Pune
Vadodara
Satara
Akola
Hyderabad
Rishikesh
Dehradun
Ahmedabad
Kolkata
Hanumangarh
Chandigarh
Mumbai
Sri Ganganagar
Patna
Patiala
Ambala
Thanesar
Lucknow
Karnal
Agra
Kanpur
Panipat
Ghaziabad
Amritsar
Faridabad
Noida
New Delhi
1
0
−1
−2
−2
0
2
4
Dim1 (33.7%)
Dim2 (33.4%)
cluster
1
2
3
4
cluster
High pollution, poor weather
High pollution,moderate weather
Low pollution, moderate weather
Low pollutants, good weather