education

March 16, 2024

Import Data

[1]:
```
from ucimlrepo import fetch_ucirepo
import numpy as np
import matplotlib.pyplot as plt
pi = np.pi
import pandas as pd

from ucimlrepo import fetch_ucirepo


import seaborn as sns
%matplotlib inline
```

1 Step 1: Pick (and clean) the data

This is some data about how different social factors impact secondary school performance.

[2]:
```
# fetch dataset
student_performance = fetch_ucirepo(id=320)

# data (as pandas dataframes)
X = student_performance.data.features
y = student_performance.data.targets

# metadata
print(student_performance.metadata)

# variable information
print(student_performance.variables)
```


```
{'uci_id': 320, 'name': 'Student Performance', 'repository_url':
'https://archive.ics.uci.edu/dataset/320/student+performance', 'data_url':
'https://archive.ics.uci.edu/static/public/320/data.csv', 'abstract': 'Predict
student performance in secondary education (high school). ', 'area': 'Social
Science', 'tasks': ['Classification', 'Regression'], 'characteristics':
['Multivariate'], 'num_instances': 649, 'num_features': 30, 'feature_types':
['Integer'], 'demographics': ['Sex', 'Age', 'Other', 'Education Level',
```

'Occupation'], 'target_col': ['G1', 'G2', 'G3'], 'index_col': None,
'has_missing_values': 'no', 'missing_values_symbol': None,
'year_of_dataset_creation': 2008, 'last_updated': 'Fri Jan 05 2024',
'dataset_doi': '10.24432/C5TG7T', 'creators': ['Paulo Cortez'], 'intro_paper':
{'title': 'Using data mining to predict secondary school student performance',
'authors': 'P. Cortez, A. M. G. Silva', 'published_in': 'Proceedings of 5th
Annual Future Business Technology Conference', 'year': 2008, 'url': 'https://www
.semanticscholar.org/paper/61d468d5254730bbecf822c6b60d7d6595d9889c', 'doi':
None}, 'additional_info': {'summary': 'This data approach student achievement in
secondary education of two Portuguese schools. The data attributes include
student grades, demographic, social and school related features) and it was
collected by using school reports and questionnaires. Two datasets are provided
regarding the performance in two distinct subjects: Mathematics (mat) and
Portuguese language (por). In [Cortez and Silva, 2008], the two datasets were
modeled under binary/five-level classification and regression tasks. Important
note: the target attribute G3 has a strong correlation with attributes G2 and
G1. This occurs because G3 is the final year grade (issued at the 3rd period),
while G1 and G2 correspond to the 1st and 2nd period grades. It is more
difficult to predict G3 without G2 and G1, but such prediction is much more
useful (see paper source for more details).', 'purpose': None, 'funded_by':
None, 'instances_represent': None, 'recommended_data_splits': None,
'sensitive_data': None, 'preprocessing_description': None, 'variable_info': "#
Attributes for both student-mat.csv (Math course) and student-por.csv
(Portuguese language course) datasets:\r\n1 school - student's school (binary:
'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)\r\n2 sex - student's sex
(binary: 'F' - female or 'M' - male)\r\n3 age - student's age (numeric: from 15
to 22)\r\n4 address - student's home address type (binary: 'U' - urban or 'R' -
rural)\r\n5 famsize - family size (binary: 'LE3' - less or equal to 3 or 'GT3' -
greater than 3)\r\n6 Pstatus - parent's cohabitation status (binary: 'T' -
living together or 'A' - apart)\r\n7 Medu - mother's education (numeric: 0 -
none, 1 - primary education (4th grade), 2 â€" 5th to 9th grade, 3 â€"
secondary education or 4 â€" higher education)\r\n8 Fedu - father's education
(numeric: 0 - none, 1 - primary education (4th grade), 2 â€" 5th to 9th grade,
3 â€" secondary education or 4 â€" higher education)\r\n9 Mjob - mother's job
(nominal: 'teacher', 'health' care related, civil 'services' (e.g.
administrative or police), 'at_home' or 'other')\r\n10 Fjob - father's job
(nominal: 'teacher', 'health' care related, civil 'services' (e.g.
administrative or police), 'at_home' or 'other')\r\n11 reason - reason to choose
this school (nominal: close to 'home', school 'reputation', 'course' preference
or 'other')\r\n12 guardian - student's guardian (nominal: 'mother', 'father' or
'other')\r\n13 traveltime - home to school travel time (numeric: 1 - <15 min., 2
- 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)\r\n14 studytime - weekly
study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 -
>10 hours)\r\n15 failures - number of past class failures (numeric: n if 1<=n<3,
else 4)\r\n16 schoolsup - extra educational support (binary: yes or no)\r\n17
famsup - family educational support (binary: yes or no)\r\n18 paid - extra paid
classes within the course subject (Math or Portuguese) (binary: yes or no)\r\n19
activities - extra-curricular activities (binary: yes or no)\r\n20 nursery -

```
attended nursery school (binary: yes or no)\r\n21 higher - wants to take higher
education (binary: yes or no)\r\n22 internet - Internet access at home (binary:
yes or no)\r\n23 romantic - with a romantic relationship (binary: yes or
no)\r\n24 famrel - quality of family relationships (numeric: from 1 - very bad
to 5 - excellent)\r\n25 freetime - free time after school (numeric: from 1 -
very low to 5 - very high)\r\n26 goout - going out with friends (numeric: from 1
- very low to 5 - very high)\r\n27 Dalc - workday alcohol consumption (numeric:
from 1 - very low to 5 - very high)\r\n28 Walc - weekend alcohol consumption
(numeric: from 1 - very low to 5 - very high)\r\n29 health - current health
status (numeric: from 1 - very bad to 5 - very good)\r\n30 absences - number of
school absences (numeric: from 0 to 93)\r\n\r\n# these grades are related with
the course subject, Math or Portuguese:\r\n31 G1 - first period grade (numeric:
from 0 to 20)\r\n31 G2 - second period grade (numeric: from 0 to 20)\r\n32 G3 -
final grade (numeric: from 0 to 20, output target)", 'citation': None}}
          name     role         type      demographic  \
0       school  Feature  Categorical             None
1          sex  Feature       Binary              Sex
2          age  Feature      Integer              Age
3      address  Feature  Categorical             None
4      famsize  Feature  Categorical            Other
5      Pstatus  Feature  Categorical            Other
6         Medu  Feature      Integer  Education Level
7         Fedu  Feature      Integer  Education Level
8         Mjob  Feature  Categorical       Occupation
9         Fjob  Feature  Categorical       Occupation
10      reason  Feature  Categorical             None
11    guardian  Feature  Categorical             None
12  traveltime  Feature      Integer             None
13   studytime  Feature      Integer             None
14    failures  Feature      Integer             None
15   schoolsup  Feature       Binary             None
16      famsup  Feature       Binary             None
17        paid  Feature       Binary             None
18  activities  Feature       Binary             None
19     nursery  Feature       Binary             None
20      higher  Feature       Binary             None
21    internet  Feature       Binary             None
22    romantic  Feature       Binary             None
23      famrel  Feature      Integer             None
24    freetime  Feature      Integer             None
25       goout  Feature      Integer             None
26        Dalc  Feature      Integer             None
27        Walc  Feature      Integer             None
28      health  Feature      Integer             None
29    absences  Feature      Integer             None
30          G1   Target  Categorical             None
31          G2   Target  Categorical             None
32          G3   Target      Integer             None
```

```
                                          description units missing_values
0   student's school (binary: 'GP' - Gabriel Perei…  None             no
1   student's sex (binary: 'F' - female or 'M' - m…  None             no
2             student's age (numeric: from 15 to 22)  None             no
3   student's home address type (binary: 'U' - urb…  None             no
4   family size (binary: 'LE3' - less or equal to …  None             no
5   parent's cohabitation status (binary: 'T' - li…  None             no
6   mother's education (numeric: 0 - none,  1 - pr…  None             no
7   father's education (numeric: 0 - none,  1 - pr…  None             no
8   mother's job (nominal: 'teacher', 'health' car…  None             no
9   father's job (nominal: 'teacher', 'health' car…  None             no
10  reason to choose this school (nominal: close t…  None             no
11  student's guardian (nominal: 'mother', 'father…  None             no
12  home to school travel time (numeric: 1 - <15 m…  None             no
13  weekly study time (numeric: 1 - <2 hours, 2 - …  None             no
14  number of past class failures (numeric: n if 1…  None             no
15      extra educational support (binary: yes or no)  None             no
16     family educational support (binary: yes or no)  None             no
17  extra paid classes within the course subject (…  None             no
18    extra-curricular activities (binary: yes or no)  None             no
19        attended nursery school (binary: yes or no)  None             no
20  wants to take higher education (binary: yes or…  None             no
21        Internet access at home (binary: yes or no)  None             no
22   with a romantic relationship (binary: yes or no)  None             no
23  quality of family relationships (numeric: from…  None             no
24  free time after school (numeric: from 1 - very…  None             no
25  going out with friends (numeric: from 1 - very…  None             no
26  workday alcohol consumption (numeric: from 1 -…  None             no
27  weekend alcohol consumption (numeric: from 1 -…  None             no
28  current health status (numeric: from 1 - very …  None             no
29  number of school absences (numeric: from 0 to 93)  None             no
30         first period grade (numeric: from 0 to 20)  None             no
31        second period grade (numeric: from 0 to 20)  None             no
32  final grade (numeric: from 0 to 20, output tar…  None             no
```

Our data points are different students, and our features for each data point are different characteristics of their family, education, and general parts of their life.

[3]:
```
X
```

[3]:
```
    school sex  age address famsize Pstatus  Medu  Fedu      Mjob      Fjob  \
0       GP   F   18       U     GT3       A     4     4   at_home   teacher
1       GP   F   17       U     GT3       T     1     1   at_home     other
2       GP   F   15       U     LE3       T     1     1   at_home     other
3       GP   F   15       U     GT3       T     4     2    health  services
4       GP   F   16       U     GT3       T     3     3     other     other
..     …  ..  …     …     …     …   …    …       …        …
```

```
644      MS   F   19        R     GT3        T     2     3  services     other
645      MS   F   18        U     LE3        T     3     1   teacher  services
646      MS   F   18        U     GT3        T     1     1     other     other
647      MS   M   17        U     LE3        T     3     1  services  services
648      MS   M   18        R     LE3        T     3     2  services     other

     … higher internet  romantic  famrel  freetime goout Dalc Walc health  \
0    …    yes       no        no       4         3     4    1    1      3
1    …    yes      yes        no       5         3     3    1    1      3
2    …    yes      yes        no       4         3     2    2    3      3
3    …    yes      yes       yes       3         2     2    1    1      5
4    …    yes       no        no       4         3     2    1    2      5
..   …      …      …        …      …       …      …   …   …    …
644  …    yes      yes        no       5         4     2    1    2      5
645  …    yes      yes        no       4         3     4    1    1      1
646  …    yes       no        no       1         1     1    1    1      5
647  …    yes      yes        no       2         4     5    3    4      2
648  …    yes      yes        no       4         4     1    3    4      5

    absences
0          4
1          2
2          6
3          0
4          0
..       …
644        4
645        4
646        6
647        6
648        4

[649 rows x 30 columns]
```

G3 is each individual's final grade in the course, it will be our target data.

[4]:
```
y
```

[4]:
```
     G1  G2  G3
0     0  11  11
1     9  11  11
2    12  13  12
3    14  14  14
4    11  13  13
..   ..  ..  ..
644  10  11  10
645  15  15  16
```

```
646  11  12   9
647  10  10  10
648  10  11  11

[649 rows x 3 columns]
```

Want to turn these categorical features into numbers.

```
[ ]: ### Convert objects to categories
obj_columns = X.select_dtypes(['object']).columns

for col in obj_columns:
    X[col] = X[col].astype('category')



### Converty categories to int8s
cat_columns = X.select_dtypes(['category']).columns



X[cat_columns] = X[cat_columns].apply(lambda x: x.cat.codes)

# running this cell makes pandas upset at how I'm doing this, so I've hidden
 ↪the output
```

I have turned the categorical features, into numerical points in order to perform calculations on them

This could introduce some errors in our conclusions, as, for example, jobs do not necessarily have the same inequality relationship that their numerical represenations may encode

(eg. a "veteranarian" may not be greater than or less than a "dentist", but since they will be transfered to unique integers, one will be encoded with a value greater than or less than the other, additionally, there will be an equally arbitrary linear heirarchy that will arise)

```
[6]: X
```

```
[6]:      school  sex  age  address  famsize  Pstatus  Medu  Fedu  Mjob  Fjob  …  \
0         0    0   18        1        0        0     4     4     0     4  …
1         0    0   17        1        0        1     1     1     0     2  …
2         0    0   15        1        1        1     1     1     0     2  …
3         0    0   15        1        0        1     4     2     1     3  …
4         0    0   16        1        0        1     3     3     2     2  …
..      …   …   …     …      …      …    …   …   …   …  …
644       1    0   19        0        0        1     2     3     3     2  …
645       1    0   18        1        1        1     3     1     4     3  …
646       1    0   18        1        0        1     1     1     2     2  …
647       1    1   17        1        1        1     3     1     3     3  …
648       1    1   18        0        1        1     3     2     3     2  …
```

```
     higher  internet  romantic  famrel  freetime  goout  Dalc  Walc  health  \
0         1         0         0       4         3      4     1     1       3
1         1         1         0       5         3      3     1     1       3
2         1         1         0       4         3      2     2     3       3
3         1         1         1       3         2      2     1     1       5
4         1         0         0       4         3      2     1     2       5
..      …       …       …     …       …    …   …   …     …
644       1         1         0       5         4      2     1     2       5
645       1         1         0       4         3      4     1     1       1
646       1         0         0       1         1      1     1     1       5
647       1         1         0       2         4      5     3     4       2
648       1         1         0       4         4      1     3     4       5

     absences
0           4
1           2
2           6
3           0
4           0
..        …
644         4
645         4
646         6
647         6
648         4

[649 rows x 30 columns]
```

[7]:
```
X.dtypes
```

[7]:
```
school         int8
sex            int8
age           int64
address        int8
famsize        int8
Pstatus        int8
Medu          int64
Fedu          int64
Mjob           int8
Fjob           int8
reason         int8
guardian       int8
traveltime    int64
studytime     int64
failures      int64
schoolsup      int8
famsup         int8
```

```
paid           int8
activities     int8
nursery        int8
higher         int8
internet       int8
romantic       int8
famrel        int64
freetime      int64
goout         int64
Dalc          int64
Walc          int64
health        int64
absences      int64
dtype: object
```

Now convert to numpy arrays so we can do math on the data.

[8]:
```
Xnp = X.to_numpy()
print(Xnp)
```

```
[[ 0  0 18 …  1  3  4]
 [ 0  0 17 …  1  3  2]
 [ 0  0 15 …  3  3  6]
 …
 [ 1  0 18 …  1  5  6]
 [ 1  1 17 …  4  2  6]
 [ 1  1 18 …  4  5  4]]
```

[9]:
```
ynp = y.to_numpy()
print(ynp)
```

```
[[ 0 11 11]
 [ 9 11 11]
 [12 13 12]
 …
 [11 12  9]
 [10 10 10]
 [10 11 11]]
```

Since Pandas are formatted with different labels/features for each column, and a different sample of data in each row, we transpose the matrix to get it in a form we like better.

[10]:
```
Xnp = Xnp.T
ynp = ynp.T
print(Xnp.shape)
```

```
(30, 649)
```

2 Part 2: Calculate eigenvalues, of C, histogram them, and fit best M-P distribution

Center the data

[11]:
```
Xmean = np.mean(Xnp, axis=1) # the mean of each row/feature across all data␣
↪points
# changing sape for formatting purposes
Xmean.shape = (Xmean.size, 1)

X_circ = Xnp - Xmean
#print(Xmean.shape)
print("means of features after centering (should all be approx 0): ", np.
↪mean(X_circ, axis=1))
```

```
means of features after centering (should all be approx 0):  [-4.37930808e-17
3.28448106e-17 -1.75172323e-16  4.37930808e-17
 -3.83189457e-17  1.47117381e-17 -1.64224053e-16 -1.53275783e-16
  1.09482702e-16  3.28448106e-17 -3.28448106e-17 -1.16325371e-17
 -6.56896212e-17  1.31379242e-16  4.65301483e-17 -5.47413510e-18
 -5.47413510e-18  1.74488056e-17  3.01077430e-17 -1.23168040e-17
 -6.84266887e-17  5.47413510e-17 -2.46336079e-17 -1.57381384e-16
  1.72435256e-16  8.75861616e-17 -1.25905107e-16  2.73706755e-18
 -1.86120593e-16  8.75861616e-17]
```

Calculate Sample Cov Matrix

[12]:
```
m, N = X_circ.shape
gamma = m/N

print('number of rows/features, m is:', m)
print('number of cols/data, N is:', N)
print('gamma is:', gamma)
print('1/gamma is:', 1/gamma)

C = X_circ@X_circ.T/N # Sample Covariance Matrix
lamb_minus = (1 - np.sqrt(gamma))**2
lamb_plus = (1 + np.sqrt(gamma))**2

eig_vals, eig_vecs = np.linalg.eigh(C)

#print(eig_vals)
print("The number of eigenvalues is", len(eig_vals))
```

```
number of rows/features, m is: 30
number of cols/data, N is: 649
gamma is: 0.046224961479198766
1/gamma is: 21.633333333333333
The number of eigenvalues is 30
```

[13]:

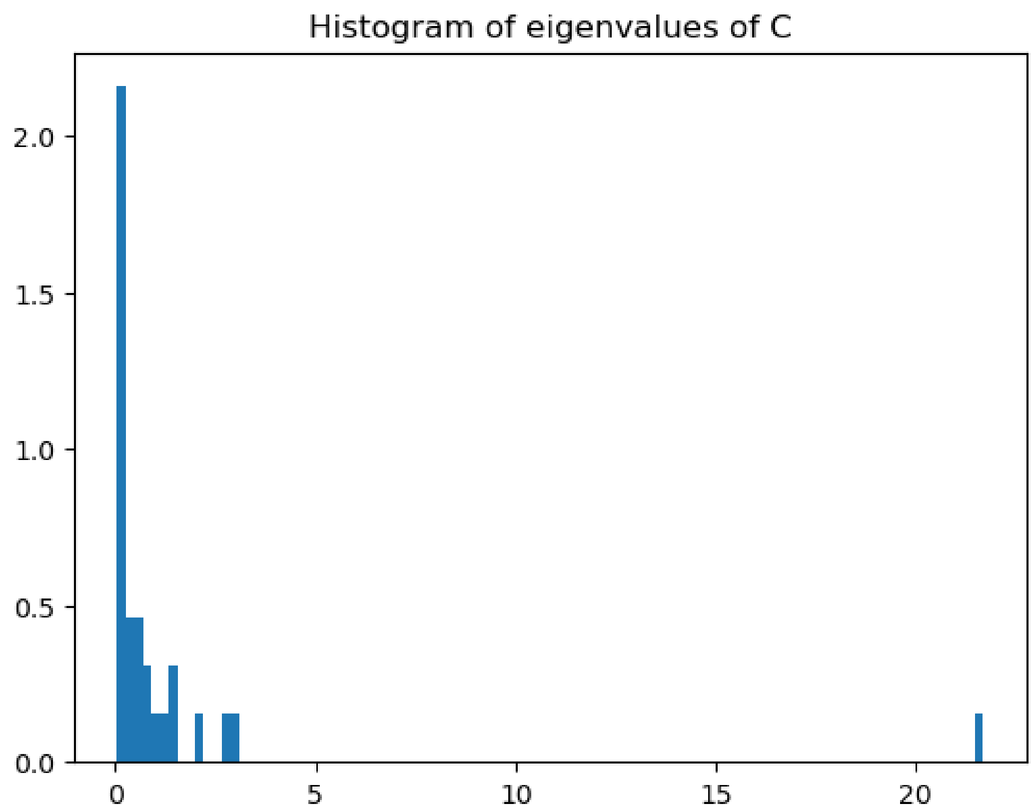
```
print(eig_vals)
```

```
[ 0.05152604  0.0735033   0.08177821  0.08407377  0.12530595  0.13974355
  0.14383113  0.16219291  0.20013945  0.20525621  0.22103834  0.24355781
  0.24670771  0.25531718  0.29479909  0.3718137   0.41579068  0.53536536
  0.61547308  0.67299266  0.76064932  0.84194604  1.05917787  1.33861433
  1.37540904  1.44033446  2.09326526  2.70659398  2.91516739 21.67649624]
```

Histogram of the eigenvalues of C

[14]:

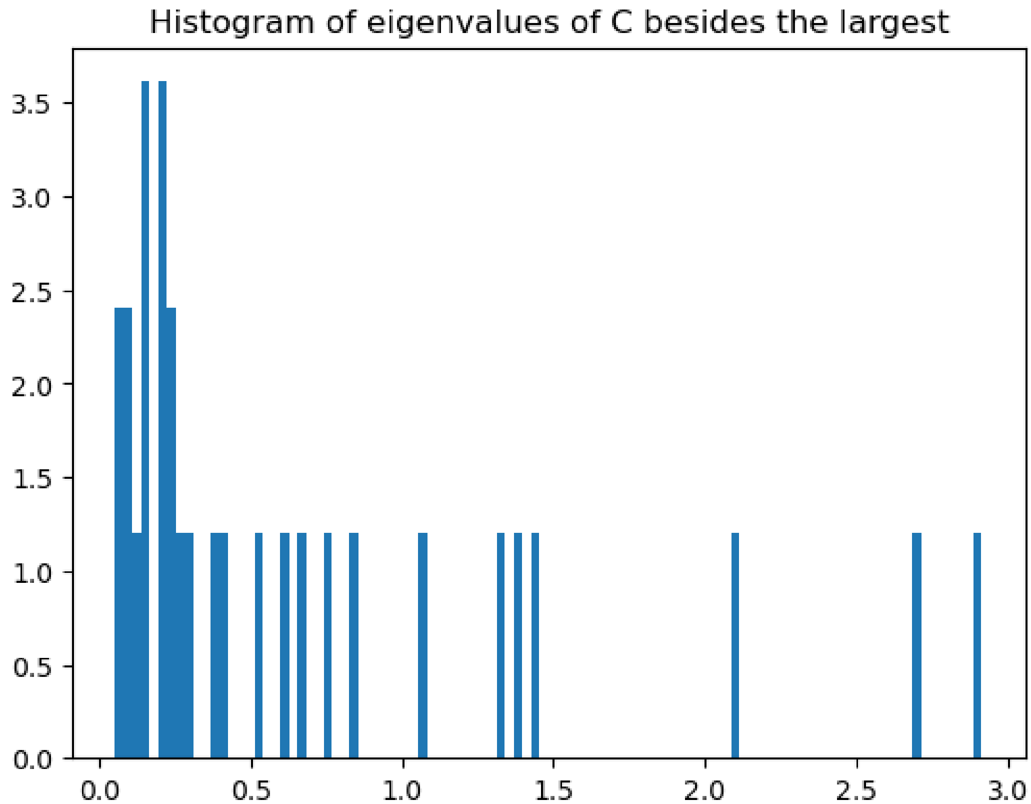
```
plt.title("Histogram of eigenvalues of C")
plt.hist(eig_vals,density=True,bins=100, label='Empirical eigenvalues');
```



Histogram of the eigenvalues of C without the big outlier

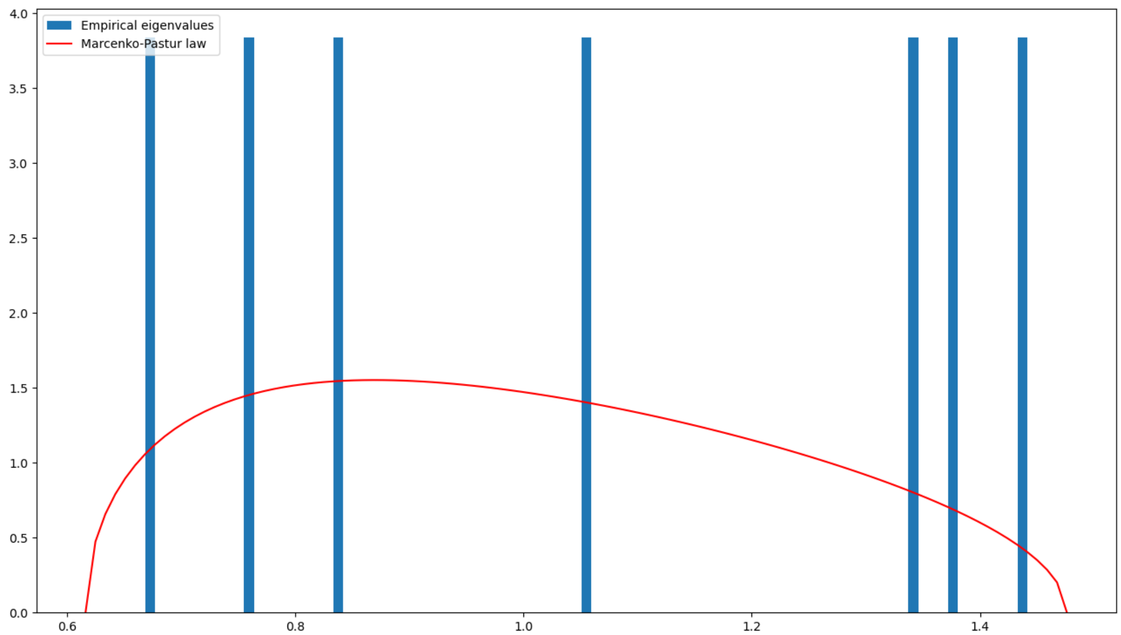
[15]:

```
plt.title("Histogram of eigenvalues of C besides the largest")
plt.hist(eig_vals[:-1],density=True,bins=100, label='Empirical eigenvalues');
```



[16]:

```
# Limiting measure
edges = np.linspace(lamb_minus,lamb_plus,100);
mu = np.sqrt((edges-lamb_minus)*(lamb_plus-edges))/(2*pi*gamma)

# Empirical histogram
plt.figure(figsize=(16,9))
plt.hist(eig_vals,bins=edges, weights=1/(m*(edges[1]-edges[0])*np.ones(m)),␣
 ↪label='Empirical eigenvalues')
plt.plot(edges,mu/edges,'r',label='Marcenko-Pastur law')
_ = plt.legend()
```



We can conclude that the data doesn't fit the distribution very well. Very few of the eigenvalues fall within the lambda +/- range, which is what we expect from Marcenko-Pastur. So likely, our data is not modeled by the "random noise + noisy signal" that we expect, and is mostly the noisy signal part.

3 Part 3: Determining which eigenvalues are outliers

Finding outliers

[17]:
```
outliers = [e for e in eig_vals if e < lamb_minus or e > lamb_plus]
print(outliers)
print(len(outliers))
```

```
[0.05152604360203202, 0.07350329636072442, 0.08177820891741958,
0.0840737726825042, 0.12530594637800255, 0.13974355046280207,
0.14383112874409573, 0.16219290931858033, 0.2001394499474628,
0.20525620924943086, 0.2210383426984562, 0.2435578050099582,
0.24670770894905159, 0.25531717628862155, 0.2947990893395774,
0.3718137049165279, 0.4157906798110284, 0.5353653614938254, 0.615473075767388,
2.0932652565320846, 2.706593983342054, 2.9151673949211365, 21.676496239762976]
23
```

Finding outliers that correspond to C_r data from X = I_m + C_r

[30]:
```
outliers2 = [e for e in outliers if e > np.sqrt(gamma)]
print(outliers2)
print(len(outliers2))
```

```
[0.2210383426984562, 0.2435578050099582, 0.24670770894905159,
0.25531717628862155, 0.2947990893395774, 0.3718137049165279, 0.4157906798110284,
0.5353653614938254, 0.615473075767388, 2.0932652565320846, 2.706593983342054,
2.9151673949211365, 21.676496239762976]
13
```

Can see that 10 of the outlier eigenvalues could have still converged to lambda if 1. we had more data points and 2. the data truly fit the model (we have already concluded that 2. likely already fails, so this is a lttle pointless)

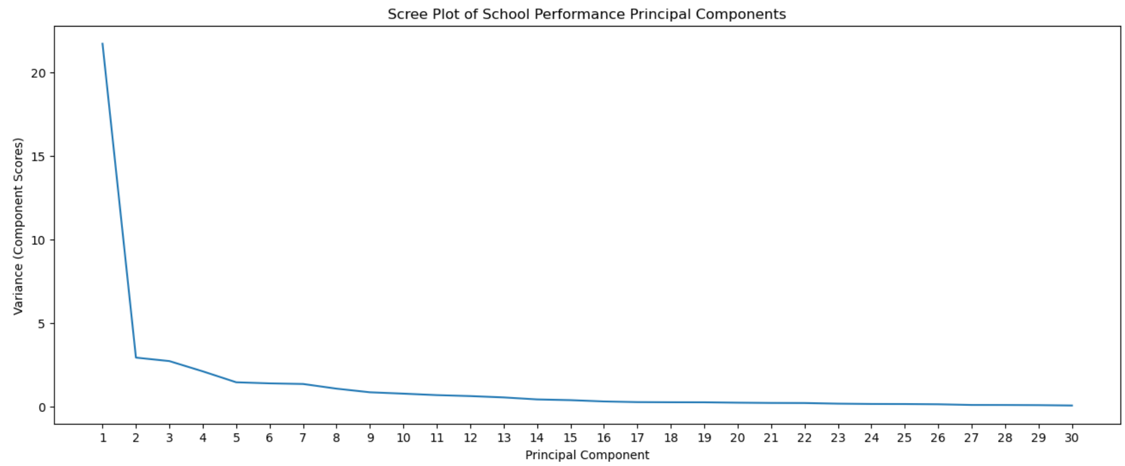
This means that, according to our model of Noise + Noisy Signal, the top 13 eigenvalues are truly outliers.

4 Part 4: PCA

[18]:
```
U, S, Vt = np.linalg.svd(X_circ, full_matrices=False)

print(U.shape)
print(S)
print(Vt.shape)
```

```
(30, 30)
[118.60879419  43.49647847  41.91156756  36.85823044  30.57412411
  29.87708931  29.47474686  26.21843693  23.37569201  22.21849245
  20.89909656  19.98604579  18.64006759  16.42705546  15.53406239
  13.83201392  12.87248412  12.65358855  12.57255008  11.97722357
  11.5417191   11.39695148  10.25978548   9.6615942    9.52331687
   9.0179576    7.38673666   7.28519441   6.90678213   5.7827677 ]
(30, 649)
```

[19]:
```
plt.figure(figsize=(16, 6))
plt.xticks([i for i in range(1,31)])
plt.xlabel("Principal Component")
plt.ylabel("Variance (Component Scores)")
plt.title("Scree Plot of School Performance Principal Components")
plt.plot([i for i in range(1,31)], np.square(S)/(N-1));
```



We can see visually that there are big(-ish) dropoffs of variance around 2, 5, and 9 PCs, while our choice of outliers says that 13 of them are from the signal.

[32]:
```
total_variance = np.sum(np.square(S))/(N-1)

print("total_variance = {:.3f} should approximately equal the sum of feature␣
↪variances: {:.3f}"
      .format(total_variance, np.sum(np.var(Xnp, axis=1))))


two_dim_variance = np.sum(np.square(S[:2]))/(N-1)
three_dim_variance = np.sum(np.square(S[:3]))/(N-1)
five_dim_variance = np.sum(np.square(S[:5]))/(N-1)
nine_dim_variance = np.sum(np.square(S[:9]))/(N-1)
thirteen_dim_variance = np.sum(np.square(S[:13]))/(N-1)

print("The variance of first two components is ", two_dim_variance)
ratio = two_dim_variance / total_variance
print("The ratio two_dim_variance / total_variance = ", ratio)

print("--------------------------------------------------------------")

print("The variance of first three components is ", three_dim_variance)
ratio = three_dim_variance / total_variance
print("The ratio three_dim_variance / total_variance = ", ratio)

print("--------------------------------------------------------------")

print("The variance of first five components is ", five_dim_variance)
ratio = five_dim_variance / total_variance
print("The ratio five_dim_variance / total_variance = ", ratio)
```

```
print("------------------------------------------------------------")

print("The variance of first nine components is ", nine_dim_variance)
ratio = nine_dim_variance / total_variance
print("The ratio nine_dim_variance / total_variance = ", ratio)

print("------------------------------------------------------------")

print("The variance of first thirteen components is ", thirteen_dim_variance)
ratio = thirteen_dim_variance / total_variance
print("The ratio thirteen_dim_variance / total_variance = ", ratio)
```

```
total_variance = 41.412 should approximately equal the sum of feature variances:
41.348
The variance of first two components is  24.62961373288578
The ratio two_dim_variance / total_variance =  0.5947505773272356
------------------------------------------------------------
The variance of first three components is  27.340384558794725
The ratio three_dim_variance / total_variance =  0.6602096840430871
------------------------------------------------------------
The variance of first five components is  30.879437362561468
The ratio five_dim_variance / total_variance =  0.7456699645436057
------------------------------------------------------------
The variance of first nine components is  35.501706776808426
The ratio nine_dim_variance / total_variance =  0.857287525115202
------------------------------------------------------------
The variance of first thirteen components is  38.090175588449796
The ratio thirteen_dim_variance / total_variance =  0.9197933092827263
```

As we can see, the first thirteen components account for about 92% of the total variance

[34]:
```
fourteen_dim_variance = np.sum(np.square(S[:14]))/(N-1)
fifteen_dim_variance = np.sum(np.square(S[:15]))/(N-1)
sixteen_dim_variance = np.sum(np.square(S[:16]))/(N-1)

print("------------------------------------------------------------")

print("The variance of first 13 components is ", thirteen_dim_variance)
ratio = thirteen_dim_variance / total_variance
print("The ratio thirteen_dim_variance / total_variance = ", ratio)

print("------------------------------------------------------------")

print("The variance of first 14 components is ", fourteen_dim_variance)
ratio = fourteen_dim_variance / total_variance
print("The ratio fourteen_dim_variance / total_variance = ", ratio)
```

```
print("---------------------------------------------------------------")

print("The variance of first 15 components is ", fifteen_dim_variance)
ratio = fifteen_dim_variance / total_variance
print("The ratio fifteen_dim_variance / total_variance = ", ratio)

print("---------------------------------------------------------------")

print("The variance of first 16 components is ", sixteen_dim_variance)
ratio = sixteen_dim_variance / total_variance
print("The ratio fifteen_dim_variance / total_variance = ", ratio)
```

```
---------------------------------------------------------------
The variance of first 13 components is  38.090175588449796
The ratio thirteen_dim_variance / total_variance =  0.9197933092827263
---------------------------------------------------------------
The variance of first 14 components is  38.50660792054448
The ratio fourteen_dim_variance / total_variance =  0.9298492270334925
---------------------------------------------------------------
The variance of first 15 components is  38.87899541204267
The ratio fifteen_dim_variance / total_variance =  0.9388415595142194
---------------------------------------------------------------
The variance of first 16 components is  39.1742494382485
The ratio fifteen_dim_variance / total_variance =  0.9459712897831779
```

We can see that the data somewhat supports the conclusion of the first 13 eigenvalues being outliers, since including more PCs (more dimensions) doesn't account for much more of the total variance, only less than 1% each time compared to the much larger increases at previous points (which we continue to not see at later dimensions)

We're going to plot the first 3 Principal Components as a visual aid for the PCA. Trying to observe all 13 dimensions would be unruly to plot.

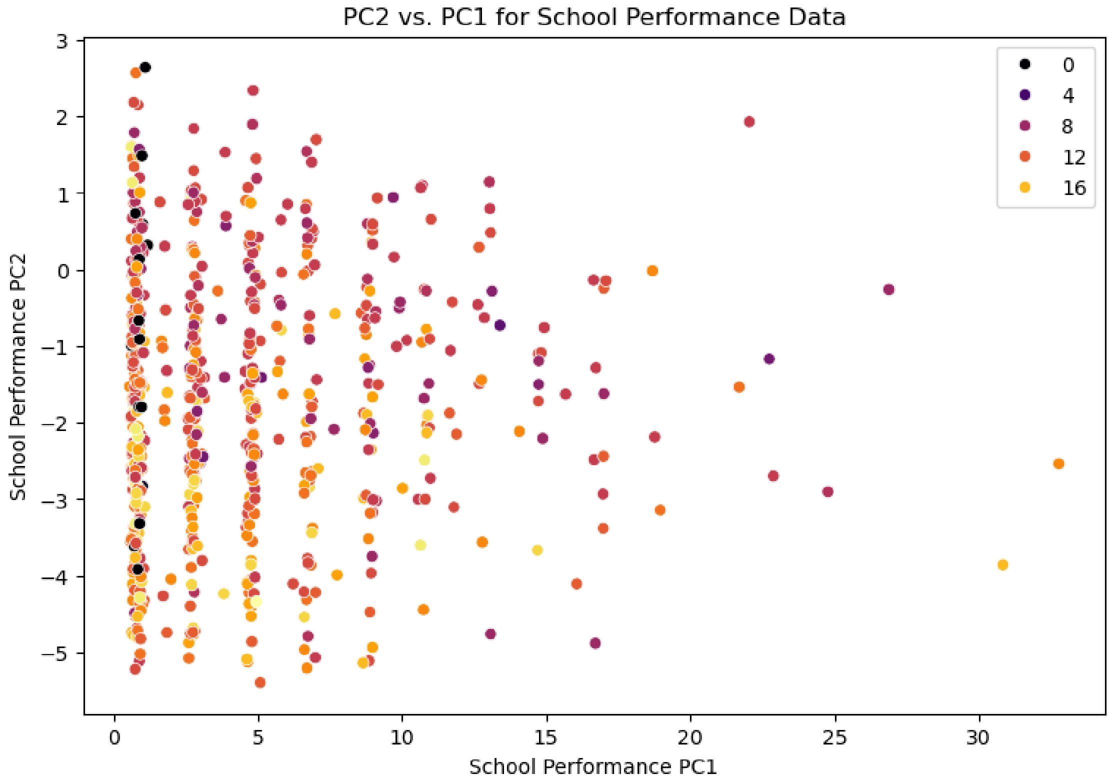
[41]:
```
mu = Xmean
Q = U[:,[0, 1, 2, 3, 4]]
print('Q.shape = ', Q.shape)
print('X_circ.shape = ', X_circ.shape)

PCs_centered = Q.T @ X_circ

print('PCs_centered.shape = ', PCs_centered.shape)

PCs_mean = Q.T @ Xmean
print('PCs_mean.shape = ', PCs_mean.shape)

PCs = PCs_mean + PCs_centered
print('PCs.shape = ', PCs.shape)
```

```
Q.shape =  (30, 5)
```

```
X_circ.shape =  (30, 649)
PCs_centered.shape =  (5, 649)
PCs_mean.shape =  (5, 1)
PCs.shape =  (5, 649)
```

5 Part 5: Project the Data

[22]:
```
plt.figure(figsize=(9, 6))
plt.title("PC2 vs. PC1 for School Performance Data")
plt.xlabel("School Performance PC1")
plt.ylabel("School Performance PC2")
sns.scatterplot(x = PCs[0, :], y = PCs[1, :], hue= y['G3'].to_numpy(),
  palette='inferno');
```



Can see from the above graph, two PCs don't really cut it.

[ ]:
```
plt.figure(figsize=(9, 6))
plt.title("PC3 vs. PC1 for School Performance Data")
plt.xlabel("School Performance PC1")
plt.ylabel("School Performance PC3")
sns.scatterplot(x = PCs[0, :], y = PCs[2, :], hue= y['G3'].to_numpy(),
  palette='inferno');
```

```
[ ]: plt.figure(figsize=(9, 6))
     plt.title("PC3 vs. PC2 for School Performance Data")
     plt.xlabel("School Performance PC2")
     plt.ylabel("School Performance PC3")
     sns.scatterplot(x = PCs[1, :], y = PCs[2, :], hue= y['G3'].to_numpy(),␣
      ↪palette='inferno');
```

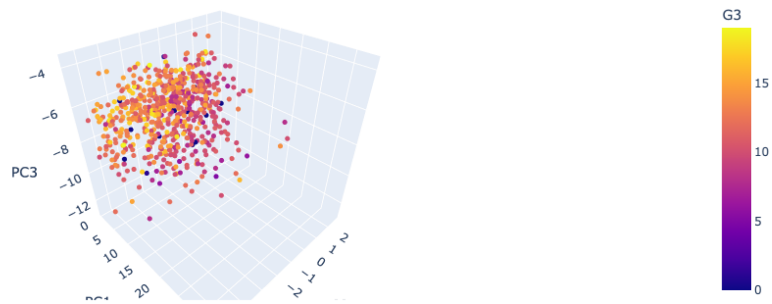
```
[25]: PCdf = pd.DataFrame(data=PCs.T, columns=['PC1', 'PC2', 'PC3', 'PC4', 'PC5'])
      PCdfplusy = pd.concat([PCdf, y], axis=1)
      PCdfplusy
```

```
[25]:         PC1       PC2       PC3       PC4        PC5  G1  G2  G3
      0    4.770820 -1.941235 -6.759534  1.095342 -16.863318   0  11  11
      1    2.688708  1.005930 -6.492180  0.206895 -15.476193   9  11  11
      2    6.729952  0.312722 -6.876292  0.858326 -13.084200  12  13  12
      3    0.611806 -2.582423 -5.979268 -1.103219 -14.643675  14  14  14
      4    0.698886 -2.674170 -7.190306 -0.957796 -15.126322  11  13  13
      ..        …         …         …         …          …  ..  ..  ..
      644  4.785204 -1.829842 -7.938944 -1.188398 -18.217544  10  11  10
      645  4.803436 -1.798681 -6.208951  2.492926 -17.499192  15  15  16
      646  6.729090  0.412956 -5.819307 -1.920517 -16.827689  11  12   9
      647  7.041484 -1.440050 -9.809495  3.073765 -15.445400  10  10  10
      648  4.932543 -1.821104 -9.607965 -0.649286 -16.494678  10  11  11

      [649 rows x 8 columns]
```

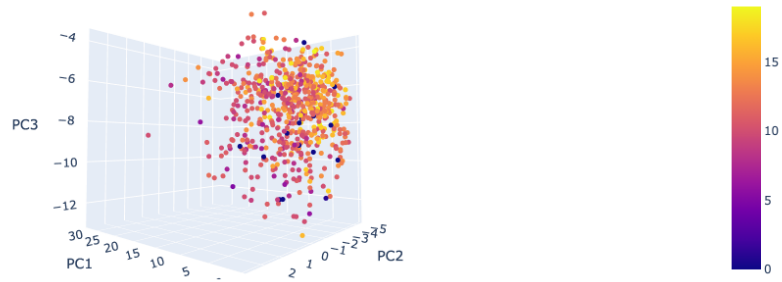
Here is a 3D scatterplot of the first 3 Principal Components

Even though these 3 PCs account for about 66% of the total variance, which is only about 7% more than the first two, I believe that the visual clarity and split of the data is much more clear.

```
[26]: import plotly.express as px
      import plotly.io as pio
      pio.renderers.default = "plotly_mimetype+notebook"

      fig = px.scatter_3d(PCdfplusy, x='PC1', y='PC2', z='PC3', color='G3')
      fig.update_traces(marker_size = 3)
      fig.show()
      # this isnt showing up when I try to export it to PDF, but I have images of it␣
       ↪working below
```



Example Angle:



Another Example Angle:

We can see some clusters of the lower G3 values pulling away from the higher G3 values, however the lowest and highest G3 valued data points still seem to be fairly randomly scattered.

The fifth dimensional data set is hard to visualize, but below is an attempt to view more components that could classify the data further.

[27]:
```
num = 5
plt.figure(figsize=(27, 18))
plt.suptitle("Scatter Matrix of PCs")
plt.subplots_adjust(wspace=0.2, hspace=0.3)
for i in range(0, num):
    for j in range(i):
        plt.subplot(num, num, i+num*j)
        sns.scatterplot(y = PCs[i,: ], x = PCs[j, :], hue= y['G3'].to_numpy(), palette='inferno')
        plt.ylabel("PC" + str(i+1))
        plt.xlabel("PC" + str(j+1))
```

Scatter Matrix of PCs