# algorithms

December 12, 2021

[ ]:
```
#importing core libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

#importing essential libraries
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn import metrics
#importing Machine learning libraries
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import SGDRegressor
```

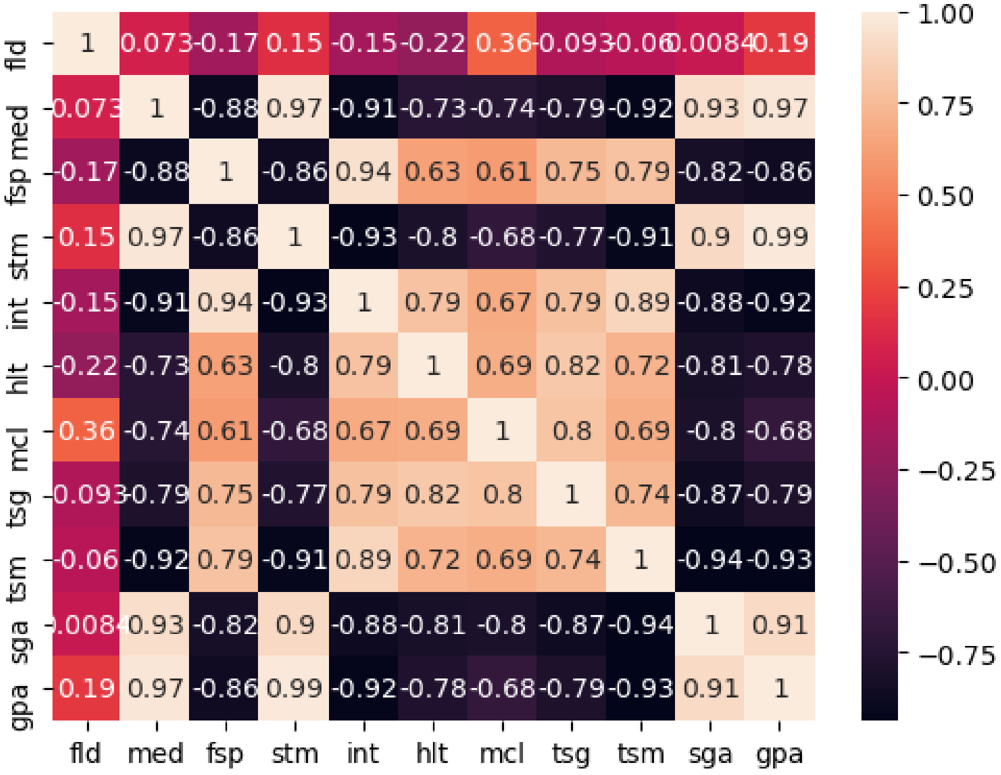
[ ]:
```
#importing and reading dataset
data = pd.read_csv('dataset.csv')
data
```

[ ]:
```
    fld  med  fsp  stm  int  hlt  mcl  tsg  tsm  sga   gpa
0     2    1    5    1    3    3    1    2    3    2  1.43
1     6    3    1    3    1    2    1    1    1    4  3.98
2     7    4    2    4    1    1    1    1    1    4  4.65
3     3    1    6    1    3    4    2    3    2    2  1.66
4     9    2    3    2    2    2    2    2    2    3  2.55
..  …  …  …  …  …  …  …  …  …  …   …
85    6    2    5    2    3    3    2    2    2    3  2.77
86    7    1    4    1    3    5    3    3    3    1  1.65
87    4    4    1    4    1    2    1    2    1    4  4.44
88    7    0    6    1    3    3    3    3    3    1  1.43
89    3    2    3    2    2    3    2    2    2    3  2.33

[90 rows x 11 columns]
```

[ ]:
```
#Determination of dataset correlation
cor_max = data.corr()
sns.heatmap(cor_max, annot=True)
```

```
plt.show()
```



| | fld | med | fsp | stm | int | hlt | mcl | tsg | tsm | sga | gpa |
|---|---|---|---|---|---|---|---|---|---|---|---|
| fld | 1 | 0.073 | -0.17 | 0.15 | -0.15 | -0.22 | 0.36 | -0.093 | -0.06 | 0.0084 | 0.19 |
| med | 0.073 | 1 | -0.88 | 0.97 | -0.91 | -0.73 | -0.74 | -0.79 | -0.92 | 0.93 | 0.97 |
| fsp | -0.17 | -0.88 | 1 | -0.86 | 0.94 | 0.63 | 0.61 | 0.75 | 0.79 | -0.82 | -0.86 |
| stm | 0.15 | 0.97 | -0.86 | 1 | -0.93 | -0.8 | -0.68 | -0.77 | -0.91 | 0.9 | 0.99 |
| int | -0.15 | -0.91 | 0.94 | -0.93 | 1 | 0.79 | 0.67 | 0.79 | 0.89 | -0.88 | -0.92 |
| hlt | -0.22 | -0.73 | 0.63 | -0.8 | 0.79 | 1 | 0.69 | 0.82 | 0.72 | -0.81 | -0.78 |
| mcl | 0.36 | -0.74 | 0.61 | -0.68 | 0.67 | 0.69 | 1 | 0.8 | 0.69 | -0.8 | -0.68 |
| tsg | -0.093 | -0.79 | 0.75 | -0.77 | 0.79 | 0.82 | 0.8 | 1 | 0.74 | -0.87 | -0.79 |
| tsm | -0.06 | -0.92 | 0.79 | -0.91 | 0.89 | 0.72 | 0.69 | 0.74 | 1 | -0.94 | -0.93 |
| sga | 0.0084 | 0.93 | -0.82 | 0.9 | -0.88 | -0.81 | -0.8 | -0.87 | -0.94 | 1 | 0.91 |
| gpa | 0.19 | 0.97 | -0.86 | 0.99 | -0.92 | -0.78 | -0.68 | -0.79 | -0.93 | 0.91 | 1 |

[ ]:
```
#Declaration of dependent variable
X=data[['fld','med','fsp','stm','int','hlt','mcl','tsg','tsm','sga' ]]

#Declaration of independent variable
y=data['gpa']
```

[ ]:
```
#spliting data
X_train, X_test, y_train, y_test = train_test_split(X,y, train_size=0.8,␣
↪test_size=0.2, random_state=42)
```

[ ]:
```
#training the model using linear regression as classifier
regressor = LinearRegression()
svmachine = SVR()
sdregressor = SGDRegressor()

regressor.fit(X_train, y_train)
svmachine.fit(X_train, y_train)
sdregressor.fit(X_train, y_train)
```

[ ]:
```
SGDRegressor()
```

[ ]:

```
#prediction score
r_score = regressor.score(X_test,y_test)
sv_score = svmachine.score(X_test,y_test)
sd_score = sdregressor.score(X_test,y_test)

#printing output
print("Regression = ",r_score*100)
print("SVmachine = ",sv_score*100)
print("SGDRegressor = ",sd_score*100)
```

```
Regression =  100.0
SVmachine =  99.11693791122808
SGDRegressor =  96.34637269474959
```

[ ]:

```
#performance evaluation using metrics of MAE, MSE and RMSE
x_pred = regressor.predict(X_test)
y_pred = svmachine.predict(X_test)
z_pred = sdregressor.predict(X_test)

#performance on Mean Absolute Error
print('Mean Absolute Error')
print('Regression :',metrics.mean_absolute_error(y_test,x_pred))
print('SVmachine :',metrics.mean_absolute_error(y_test,y_pred))
print('SGDRegressor :',metrics.mean_absolute_error(y_test,z_pred))
```

```
Mean Absolute Error
Regression : 8.635067969306773e-16
SVmachine : 0.100018994966541
SGDRegressor : 0.1466979512437259
```

[ ]:

```
#performance on Mean Squared Error
print('Mean Squared Error')
print('Regression :',metrics.mean_squared_error(y_test,x_pred))
print('SVmachine :',metrics.mean_squared_error(y_test,y_pred))
print('SGDRegressor :',metrics.mean_squared_error(y_test,z_pred))
```

```
Mean Squared Error
Regression : 1.139465751985906e-30
SVmachine : 0.01000383155915905
SGDRegressor : 0.04139037628996145
```

[ ]:

```
#performance evaluation using nearest prediction accuracy
x_pred = regressor.predict(X_test)
y_pred = svmachine.predict(X_test)
z_pred = sdregressor.predict(X_test)

df = pd.DataFrame({'Actual':y_test,'linearR_Prediction':
 ↪x_pred,'supportVM_Prediction': y_pred, 'stochasticGD_Prediction': z_pred})
```

```
df
```

```
[ ]:    Actual  linearR_Prediction  supportVM_Prediction  stochasticGD_Prediction
     40    1.43                1.43              1.530273                 1.566876
     22    4.65                4.65              4.549909                 4.638486
     55    2.77                2.77              2.670055                 2.875014
     70    1.43                1.43              1.530273                 1.566876
     0     1.43                1.43              1.530273                 1.566876
     26    1.65                1.65              1.749776                 1.674573
     39    2.33                2.33              2.429759                 2.632138
     65    2.77                2.77              2.670055                 2.875014
     10    1.43                1.43              1.530273                 1.566876
     44    2.55                2.55              2.650128                 3.047113
     81    3.98                3.98              4.080073                 3.848912
     35    2.77                2.77              2.670055                 2.875014
     56    1.65                1.65              1.749776                 1.674573
     86    1.65                1.65              1.749776                 1.674573
     12    4.65                4.65              4.549909                 4.638486
     4     2.55                2.55              2.650128                 3.047113
     18    1.43                1.43              1.529909                 1.303093
     28    1.43                1.43              1.529909                 1.303093
```