Problem Statement:

Take any Dataset of your choice, perform EDA(Exploratory Data Analysis) and apply a suitable Classifier, Regressor or Clusterer and calculate the accuracy of the model.

APPROACH:

We have chosen "mtcars dataset" to work with this project.

First we imported all the libraries required to perform EDA and Regressions on the dataset.
Later we used linear regression for mileage prediction and multiple linear regression for the further analysis.

PROGRAM AND APPROPRIATE SCREENSHOTS:

```
import numpy as np
import scipy as sp
import matplotlib as mpl
import matplotlib.cm as cm
import matplotlib.pyplot as plt
import pandas as pd
import seaborn.apionly as sns
df=pd.read_csv("mtcars.csv")
df=df.rename(columns={"Unnamed: 0":"name"})
df.head()
```

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

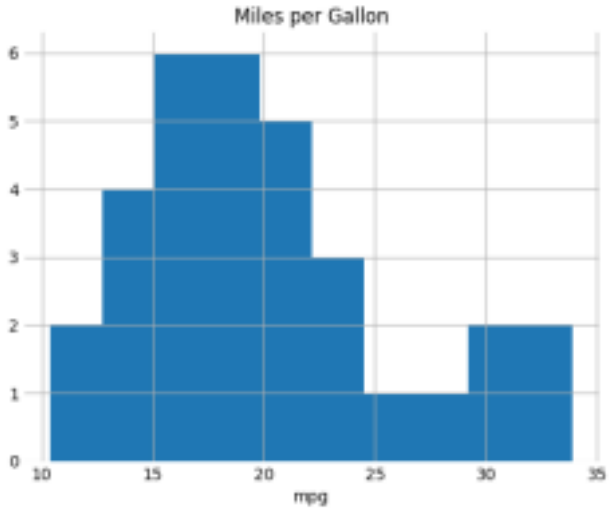
```
df.shape
```

```
(32, 12)
```

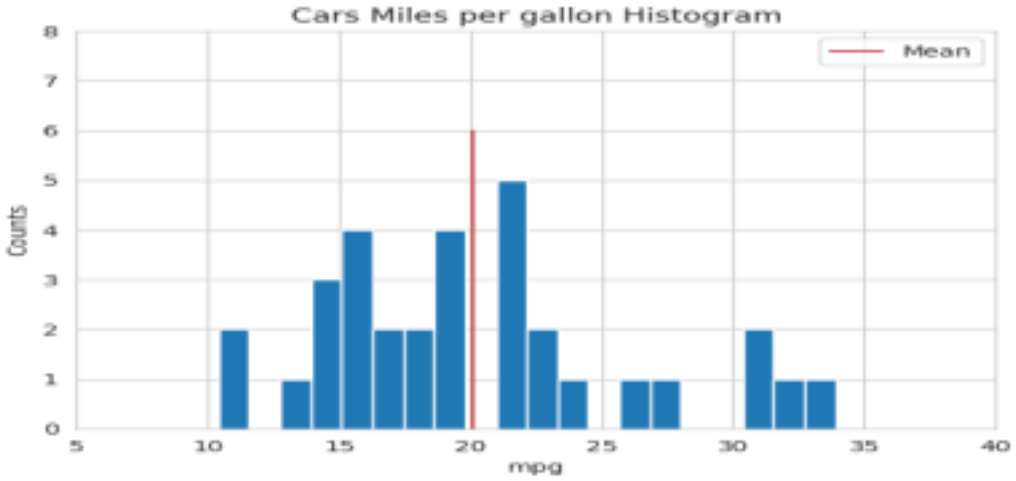
```
df.describe()
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
count	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000	32.000000	32.0000
mean	20.090625	6.187500	230.721875	146.687500	3.596563	3.217250	17.848750	0.437500	0.406250	3.687500	2.8125
std	6.026948	1.785922	123.938694	68.562868	0.534679	0.978457	1.786943	0.504016	0.498991	0.737804	1.6152
min	10.400000	4.000000	71.100000	52.000000	2.760000	1.513000	14.500000	0.000000	0.000000	3.000000	1.0000
25%	15.425000	4.000000	120.825000	96.500000	3.080000	2.581250	16.892500	0.000000	0.000000	3.000000	2.0000
50%	19.200000	6.000000	196.300000	123.000000	3.695000	3.325000	17.710000	0.000000	0.000000	4.000000	2.0000
75%	22.800000	8.000000	326.000000	180.000000	3.920000	3.610000	18.900000	1.000000	1.000000	4.000000	4.0000
max	33.900000	8.000000	472.000000	335.000000	4.930000	5.424000	22.900000	1.000000	1.000000	5.000000	8.0000

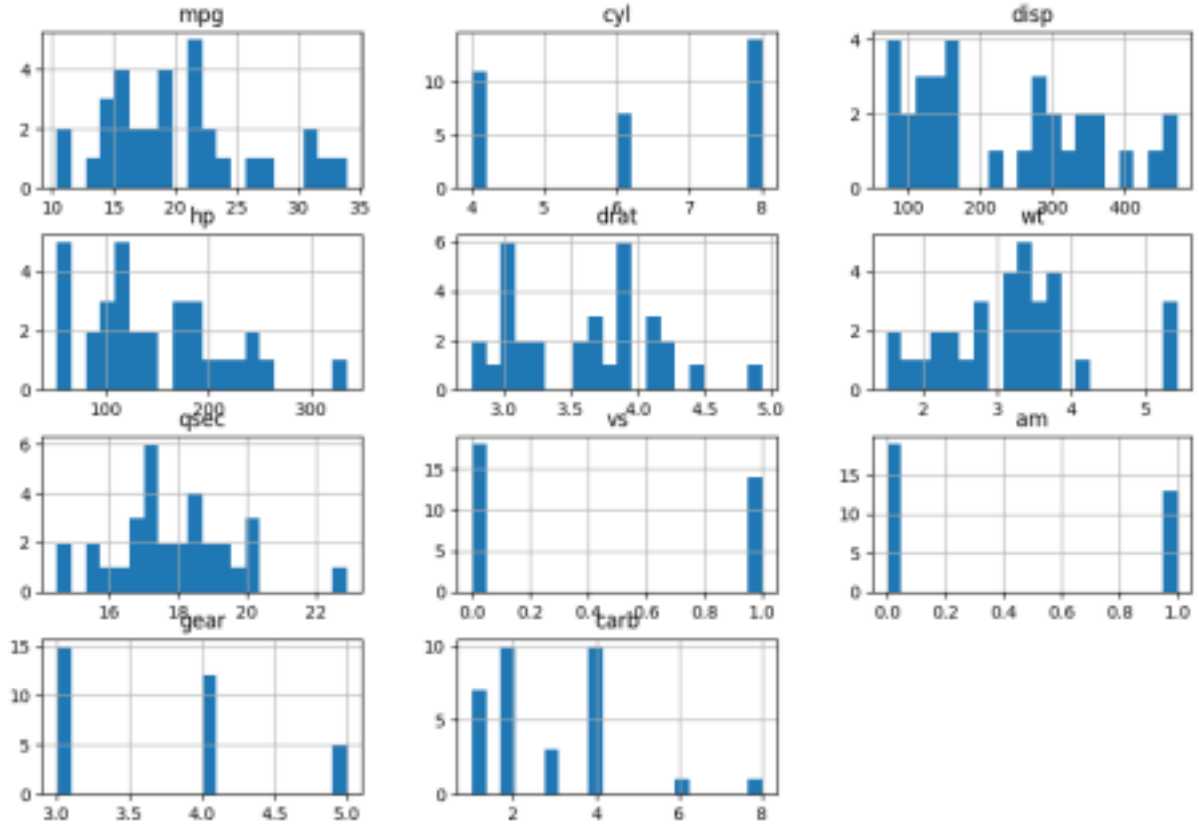
```
sns.reset_defaults()
ax = plt.gca()
df.mpg.hist()
plt.xlabel("mpg");
plt.title("Miles per Gallon")
ax.tick_params(axis='both', which='both',length=0)
sns.despine(bottom=True, left=True)
```



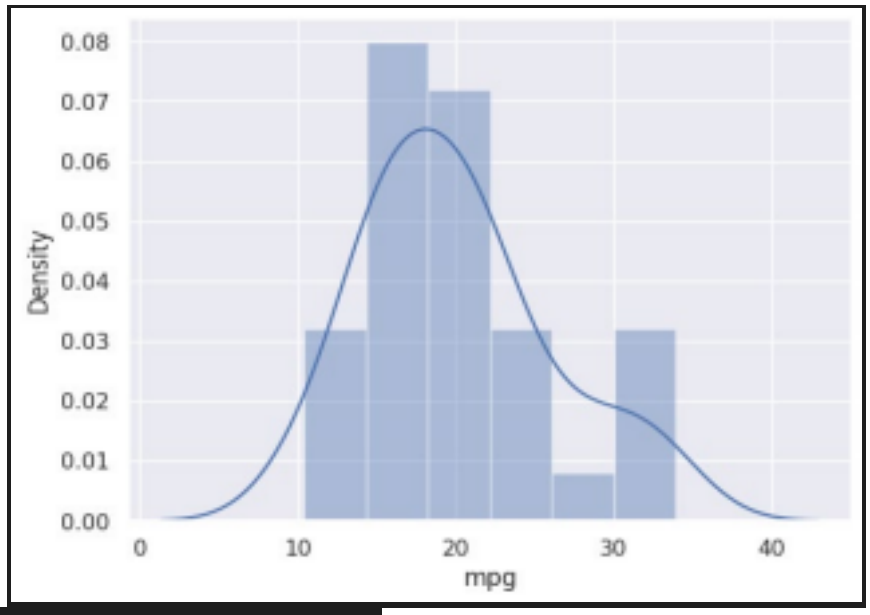

```
with sns.axes_style("whitegrid"):
 plt.hist(df.mpg.values, bins=20)
 plt.xlim(5, 40)
 plt.ylim([0, 8])
 plt.axvline(df.mpg.mean(), 0, 0.75, color='r', label='Mean')
plt.xlabel("mpg")
 plt.ylabel("Counts")
 plt.title("Cars Miles per gallon Histogram")
 plt.legend()
```



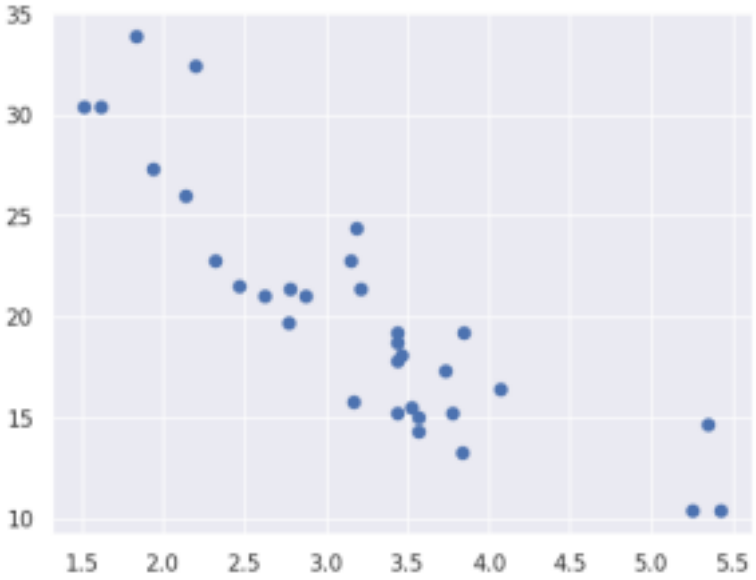

```
df.hist(figsize=(12,8),bins=20)
plt.show()
```




```
sns.set()
sns.distplot(df.mpg);
```



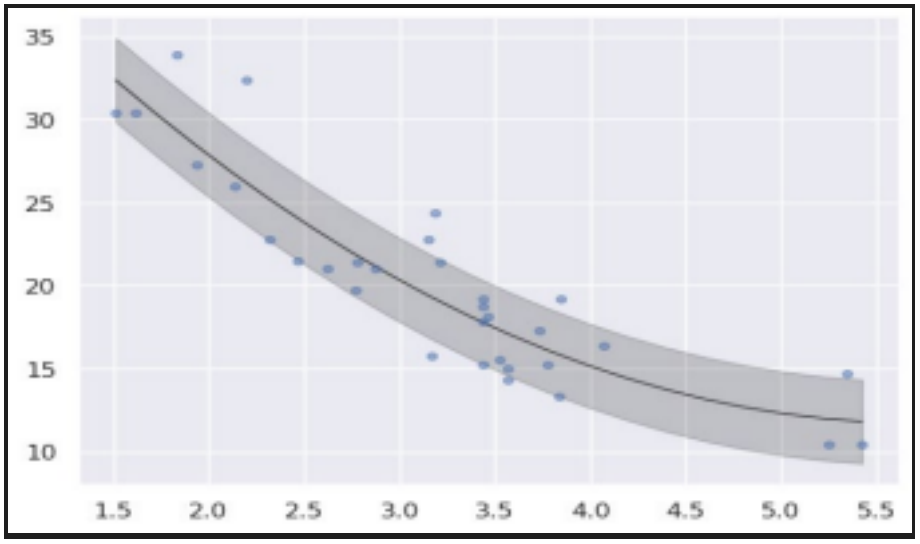

```
plt.plot(df.wt, df.mpg, 'o')
plt.show()
```



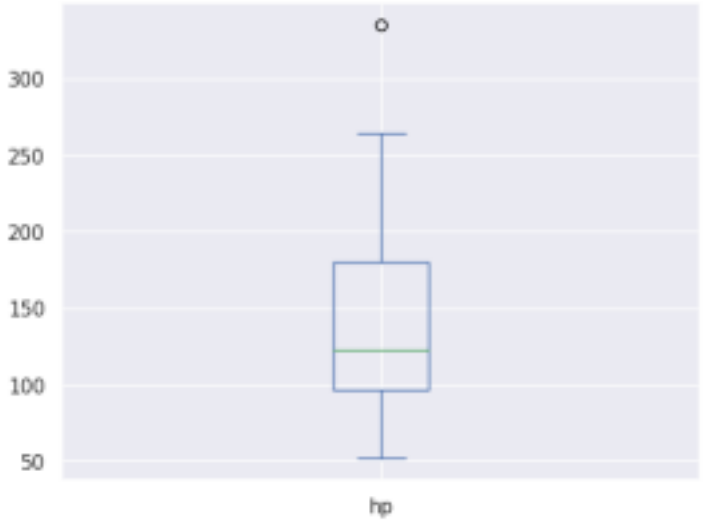
Correlation between data is identified using Regression. A quadratic fit for the current dataset with a 1 standard deviation bar is suitable.

```
x = df.wt
y = df.mpg
params = np.polyfit(x, y, 2)
xp = np.linspace(x.min(), x.max(), 20)
yp = np.polyval(params, xp)
plt.plot(xp, yp, 'k', alpha=0.8, linewidth=1)
plt.plot(df.wt, df.mpg, 'o', markersize=4, alpha=0.5)
sig = np.std(y - np.polyval(params, x))
```

```
plt.fill_between(xp, yp - sig, yp + sig, color='k',
alpha=0.2)
```




```
df.hp.plot(kind='box');
```



Linear regression for mileage prediction in cars

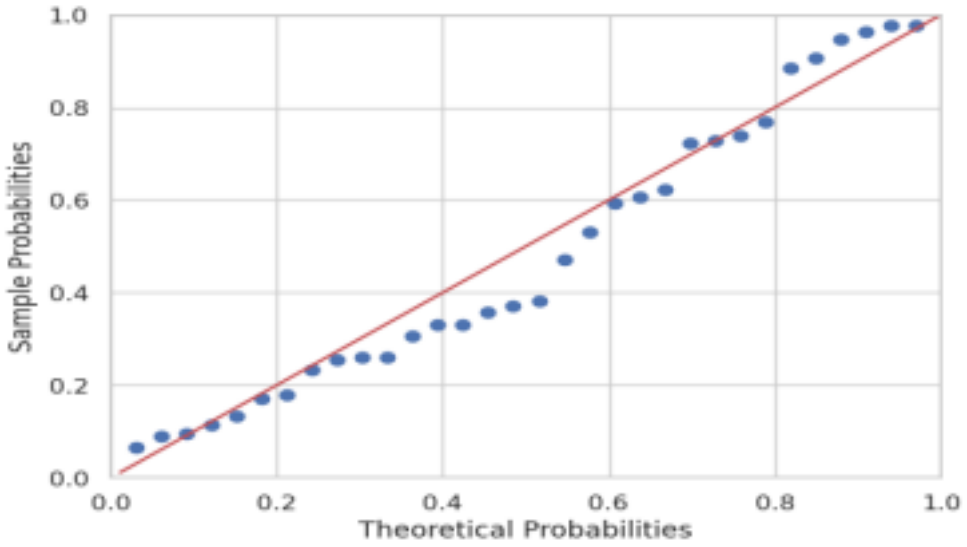
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
data_without_outlier = df.drop(index=30)
X = data_without_outlier.loc[:,['hp', 'wt']]
y = data_without_outlier.mpg
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0 .15, random_state=14)
```

```
model = LinearRegression()
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
normaliz e=False)
model.fit(X_train, y_train)
y_predict = model.predict(X_test)
r2_score(y_test, y_predict)*100
#Accuracy is : 95.8%
```

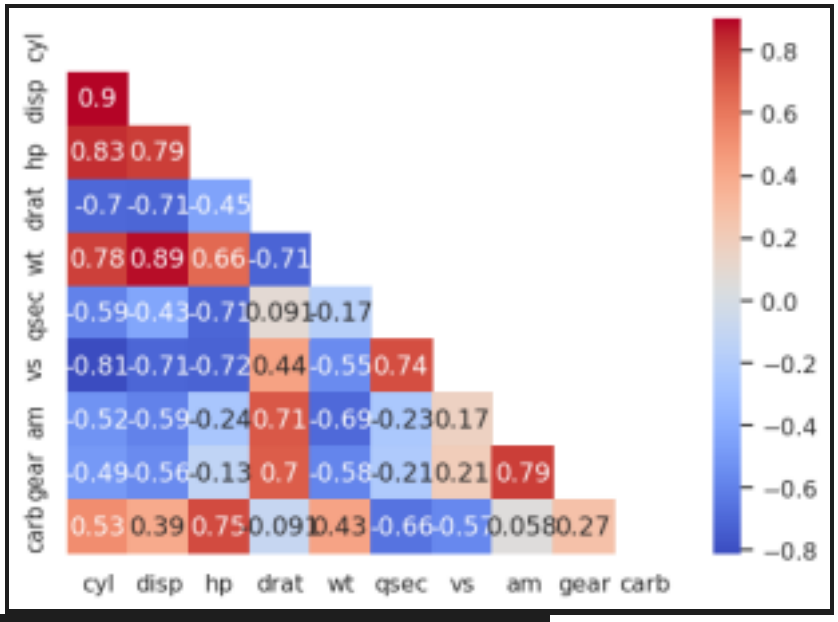
Mileage prediction sample:

```
model.predict([[110, 3.2]])
```

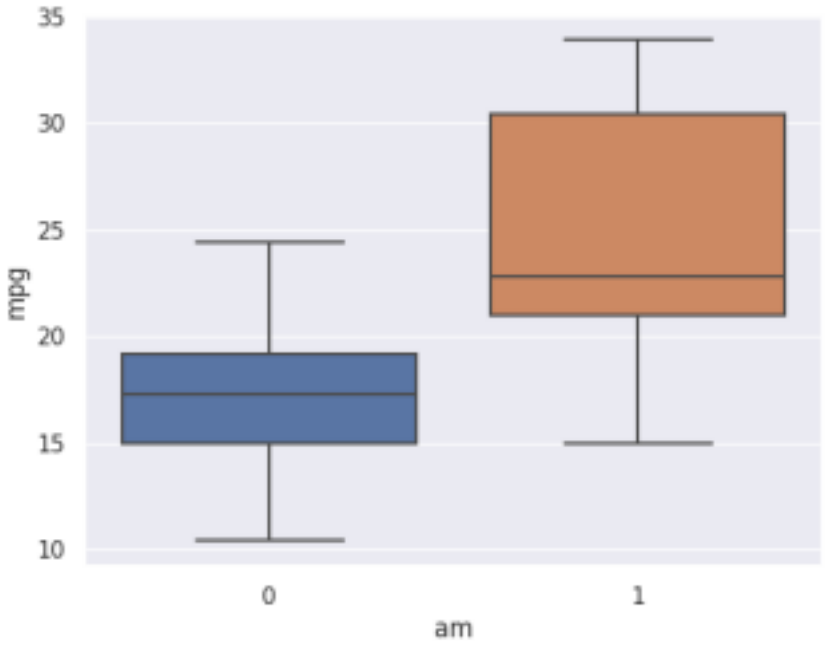
```
>>> array([21.50505271])
import statsmodels.api as sm
Y=df.mpg
X2 = sm.add_constant(X)
X3 = X2[['const','wt','qsec','am']]
est2 = sm.OLS(Y, X3).fit()
probplot = sm.ProbPlot(est2.resid)
plt.figure()
probplot.ppplot(line='45')
plt.show()
```



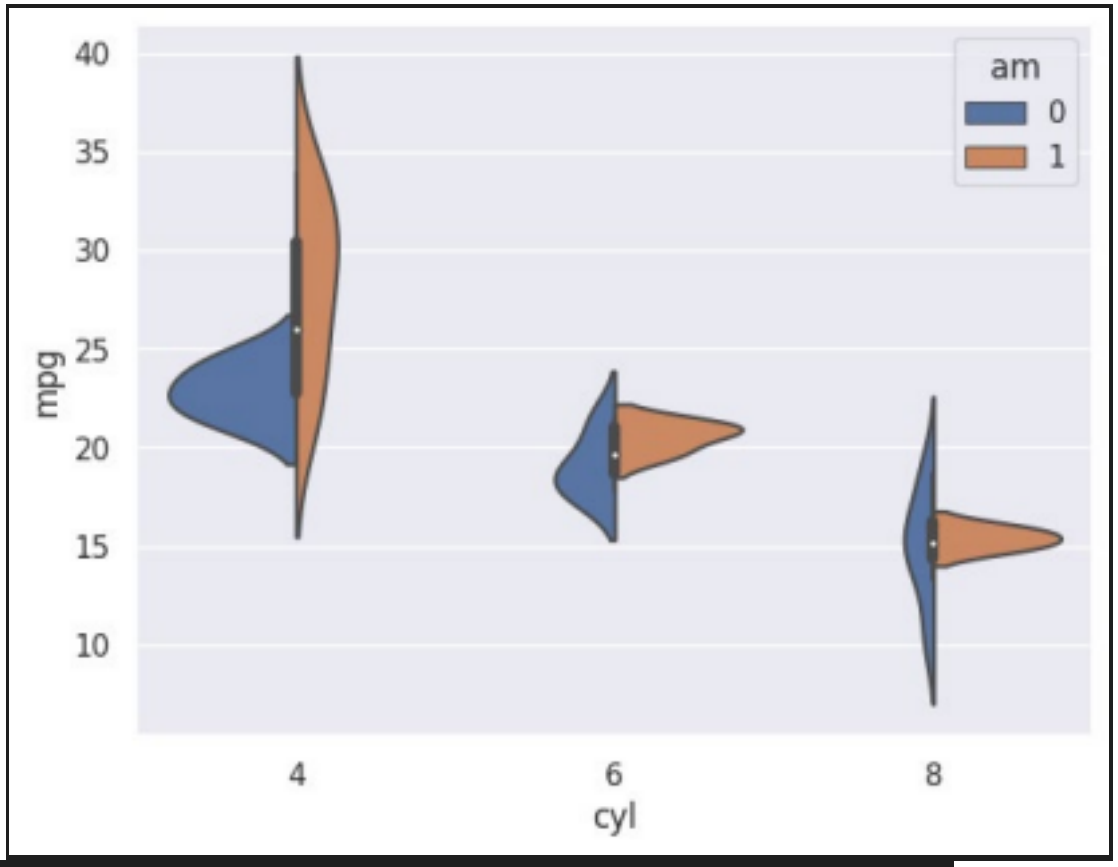

```
X=df.iloc[:,2:]
sns.set_style("whitegrid")
mask = np.triu(np.ones_like(X.corr()))
corr = sns.heatmap(X.corr(), annot=True, cmap='coolwarm', mask=mask)
```



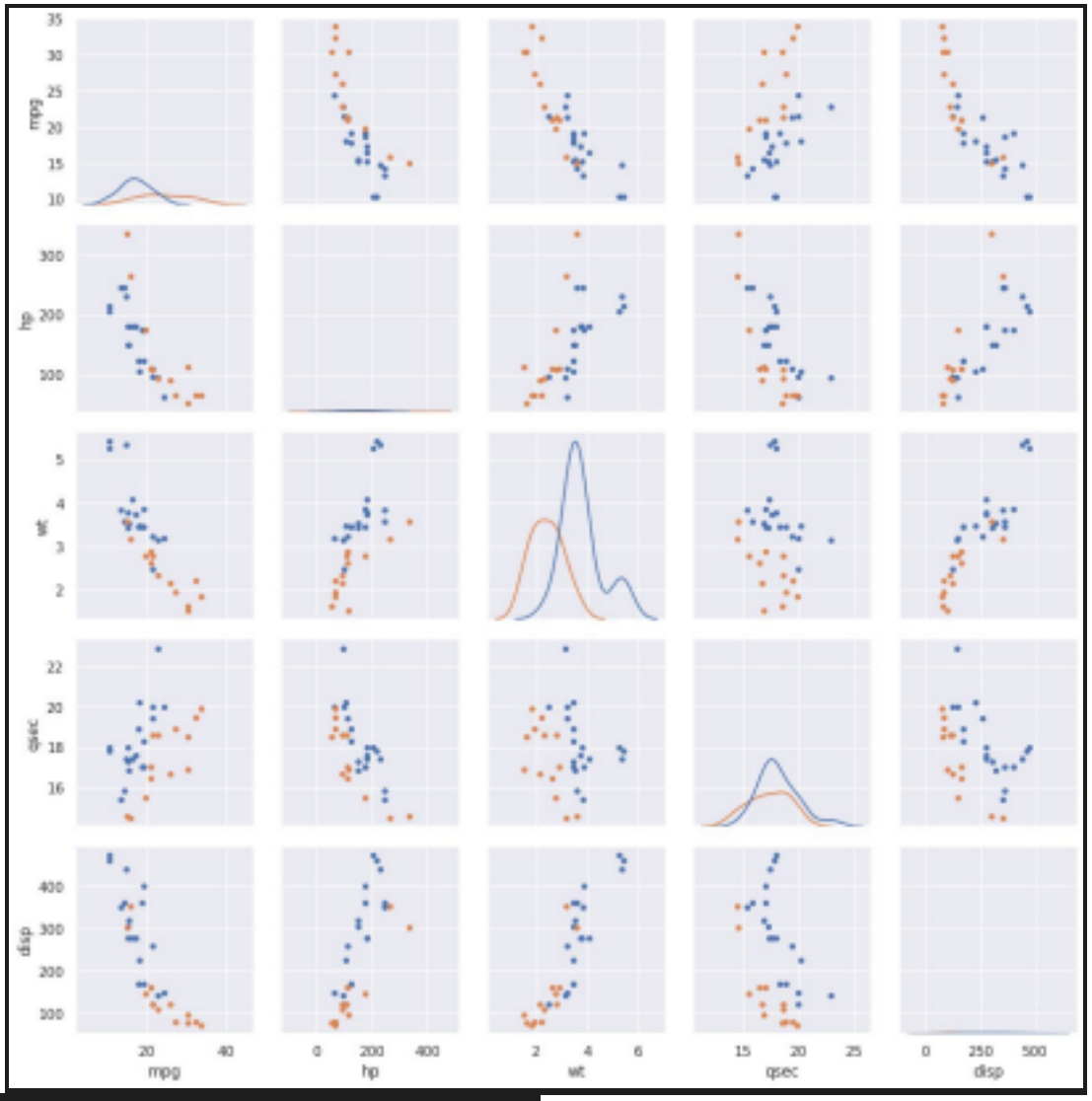

```
sns.boxplot(x='am', y='mpg', data=df);
```




```
sns.violinplot(x='cyl', y='mpg', hue='am', order=[4, 6, 8], data=df, split=True);
```



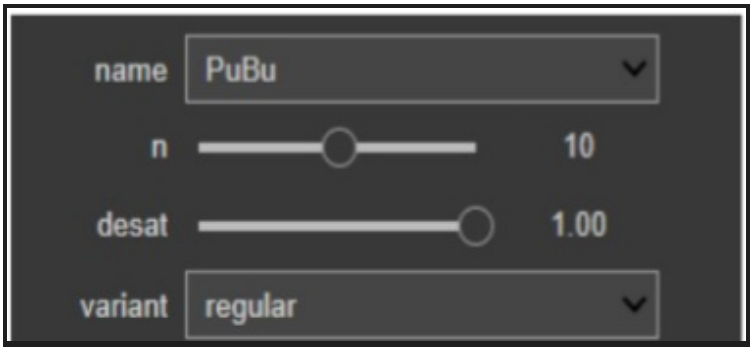

```
g = sns.PairGrid(df, vars=['mpg', 'hp', 'wt', 'qsec', 'disp'],
hue="am") g.map_diag(sns.kdeplot)
g.map_offdiag(plt.scatter, s=15)
```



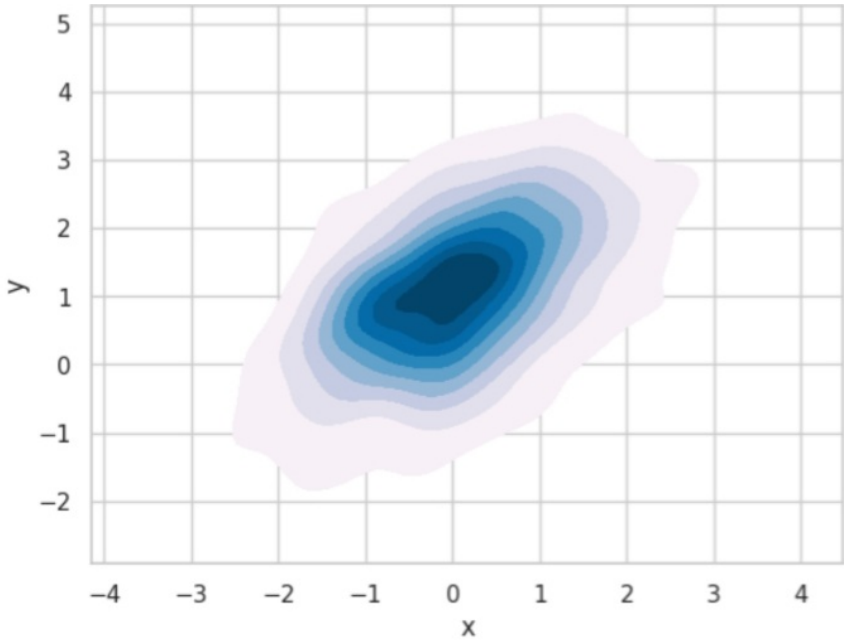

```
mean, cov = [0, 1], [(1, .5), (.5, 1)]
data = np.random.multivariate_normal(mean, cov, 1000)
df = pd.DataFrame(data, columns=["x", "y"])
df.head()
```

	x	y
0	-0.131869	1.402894
1	-1.408933	2.307948
2	-1.119510	0.558828
3	0.432207	2.842776
4	0.270141	0.109031

```
seqpal = sns.choose_colorbrewer_palette("sequential", as_cmap=True)
```



```
sns.kdeplot(df.x, df.y, cmap=seqpal, shade=True);
```



CONCLUSION:

The accuracy obtained for the "mtcars dataset" model after using multiple linear regressions is 95.83680916832566

Therefore the accuracy obtained is 95.83%(approx.).