```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import KFold
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error
```

```
df = pd.read_csv("/content/winequality-red.csv")
df.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.isna().sum()
```

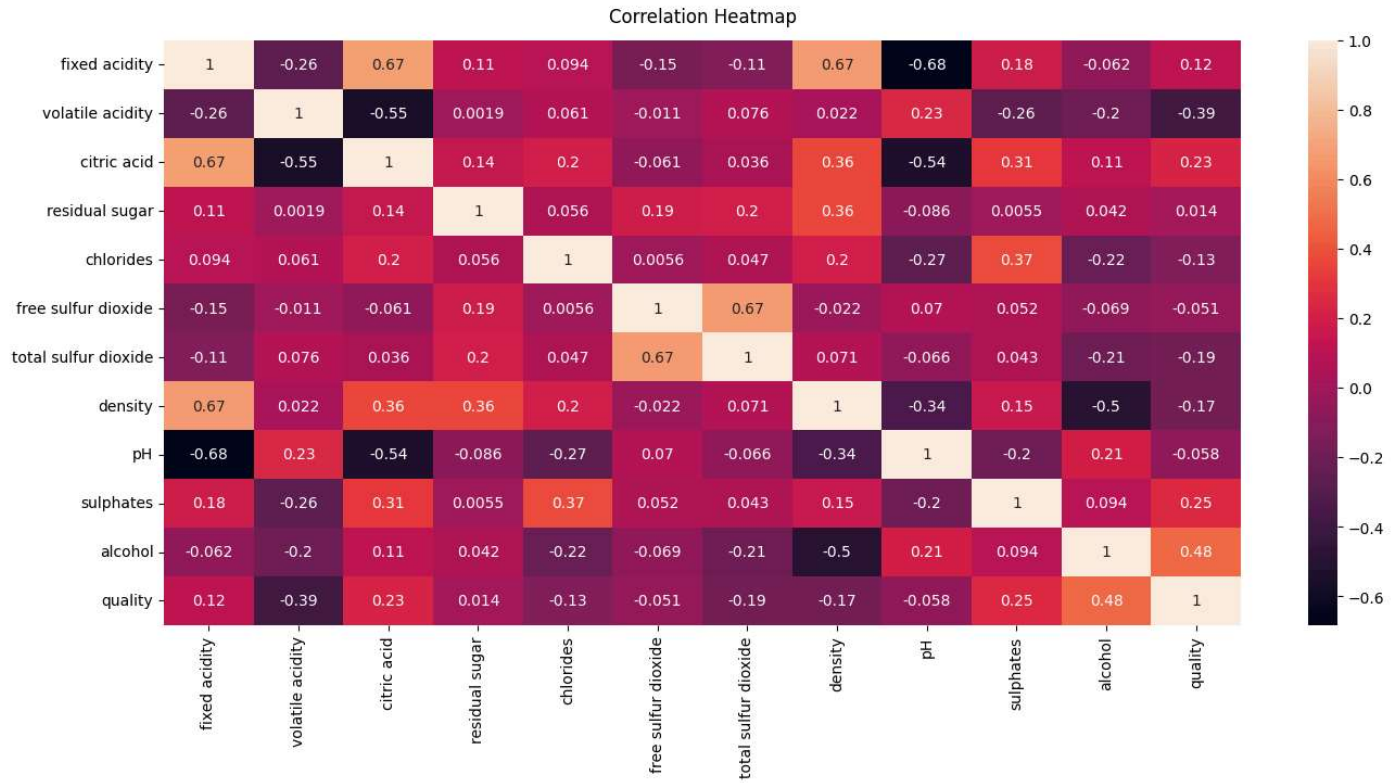
	0
fixed acidity	0
volatile acidity	0
citric acid	0
residual sugar	0
chlorides	0
free sulfur dioxide	0
total sulfur dioxide	0
density	0
pH	0
sulphates	0
alcohol	0
quality	0

```
df.shape
```

```
(1599, 12)
```

```
corr = df.corr()
```

```
plt.figure(figsize=(16, 7))
h_map = sns.heatmap(corr,annot=True)
h_map.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad=12);
plt.show()
```




```
df.drop(columns={"residual sugar","free sulfur dioxide","pH"},inplace=True)
df.head()
```

	fixed acidity	volatile acidity	citric acid	chlorides	total sulfur dioxide	density	sulphates	alcohol	quality
0	7.4	0.70	0.00	0.076	34.0	0.9978	0.56	9.4	5
1	7.8	0.88	0.00	0.098	67.0	0.9968	0.68	9.8	5
2	7.8	0.76	0.04	0.092	54.0	0.9970	0.65	9.8	5
3	11.2	0.28	0.56	0.075	60.0	0.9980	0.58	9.8	6
4	7.4	0.70	0.00	0.076	34.0	0.9978	0.56	9.4	5

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
X = df.drop(columns={"quality"})
y =df["quality"]
```

```
X.head()
```

	fixed acidity	volatile acidity	citric acid	chlorides	total sulfur dioxide	density	sulphates	alcohol
0	7.4	0.70	0.00	0.076	34.0	0.9978	0.56	9.4
1	7.8	0.88	0.00	0.098	67.0	0.9968	0.68	9.8
2	7.8	0.76	0.04	0.092	54.0	0.9970	0.65	9.8
3	11.2	0.28	0.56	0.075	60.0	0.9980	0.58	9.8
4	7.4	0.70	0.00	0.076	34.0	0.9978	0.56	9.4

Next steps: Generate code with X | View recommended plots | New interactive sheet

```
y.head()
```

	quality
0	5
1	5
2	5
3	6
4	5

```
from sklearn.model_selection import StratifiedKFold
import numpy as np
str_k_fold = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
```

```
errors = []
```

```
k_fold = KFold(n_splits=5, shuffle=True, random_state=42)
for k, (train_idx,test_idx) in enumerate(k_fold.split(X)):
  X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
  y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

  print(f"K value: {k+1}")
  print(f"Train samples : {X_train.shape[0]}")
  print(f"Test samples : {X_test.shape[0]}")

  model = LinearRegression()
  model.fit(X_train,y_train)

  y_pred = model.predict(X_test)

  mse = mean_squared_error(y_test,y_pred)

  errors.append(mse)

  mae = mean_absolute_error(y_test,y_pred)

  print(f"Mean Squared Error: {mse}")
  print(f"Mean Absolute Error: {mae}")
```

```
K value: 1
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.3925476726077163
Mean Absolute Error: 0.5062120061229034
K value: 2
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.46754015846026586
Mean Absolute Error: 0.5367669567677668
K value: 3
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.4868868537822788
Mean Absolute Error: 0.5388388981022544
K value: 4
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.4593242511768576
Mean Absolute Error: 0.5282258005261841
K value: 5
Train samples : 1280
Test samples : 319
Mean Squared Error: 0.3401149908432861
Mean Absolute Error: 0.4382362435867091
```

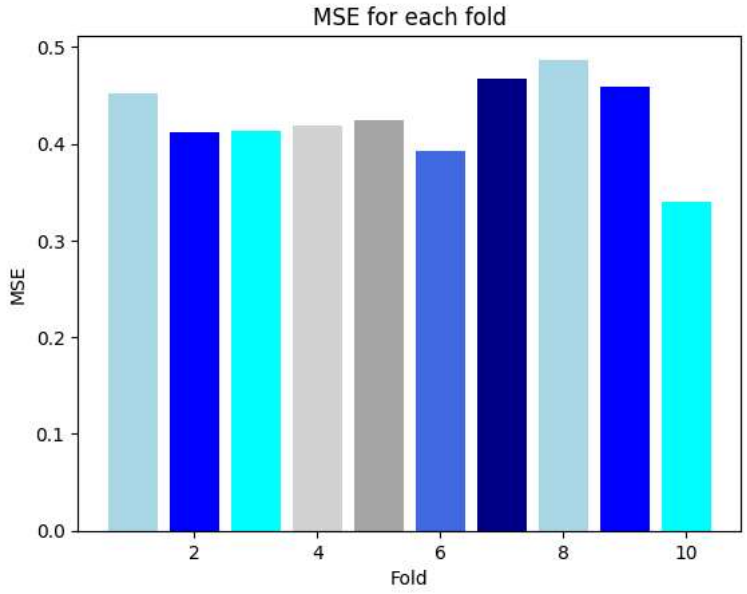
```
avg_mse = sum(errors) / len(errors)
print(f"Average mean squared error is {avg_mse}")
```

```
Average mean squared error is 0.4267026123252749
```

```
plt.figure(figsize=(8,5))
fig , ax = plt.subplots()
```

```
color = ['lightblue', 'blue', 'Cyan', 'lightgrey',"darkgrey","royalblue","darkblue"]
ax.bar(range(1,len(errors)+1),errors,color=color)
ax.set_xlabel("Fold")
ax.set_ylabel("MSE")
ax.set_title("MSE for each fold")
plt.show()
```

```
<Figure size 800x500 with 0 Axes>
```




```
# Stratified K-Fold Cross-Validation
for k, (train_idx, test_idx) in enumerate(str_k_fold.split(X, y)):
    X_train, X_test = X.iloc[train_idx], X.iloc[test_idx]
    y_train, y_test = y.iloc[train_idx], y.iloc[test_idx]

    print(f"K value: {k+1}")
    print(f"Train samples : {X_train.shape[0]}")
    print(f"Test samples : {X_test.shape[0]}")

    # Initialize Linear Regression model
    model = LinearRegression()
    model.fit(X_train, y_train)

    # Make predictions on the test set
    y_pred = model.predict(X_test)

    # Calculate Mean Squared Error and Mean Absolute Error
    mse = mean_squared_error(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)

    # Store the error values for analysis later
    errors.append(mse)

    print(f"Mean Squared Error: {mse}")
    print(f"Mean Absolute Error: {mae}")

# After the loop, you can analyze the errors list for average MSE/MAE across all folds
average_mse = np.mean(errors)
print(f"Average Mean Squared Error across all folds: {average_mse}")
```

```
K value: 1
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.45180989240051667
Mean Absolute Error: 0.5219116528931457
K value: 2
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.4119781788986695
Mean Absolute Error: 0.506153308522676
K value: 3
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.41382183810235834
```

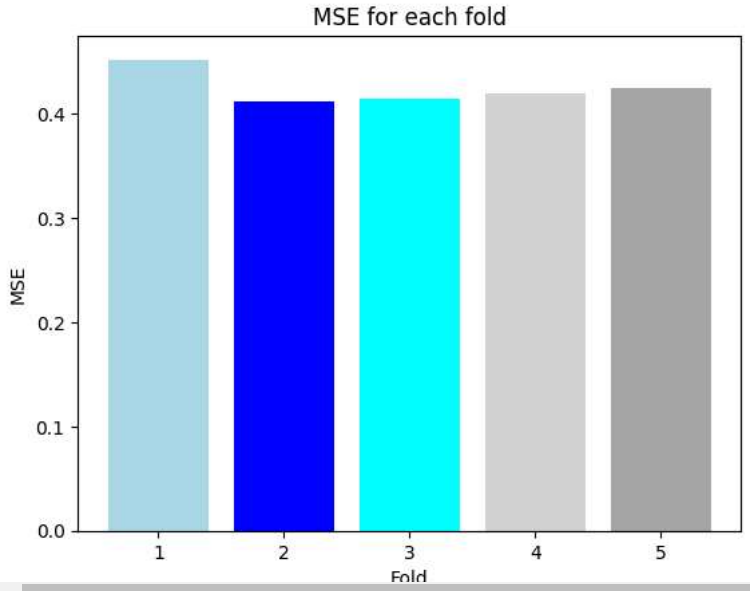
```
Mean Absolute Error: 0.49167761782267627
K value: 4
Train samples : 1279
Test samples : 320
Mean Squared Error: 0.41888327604983217
Mean Absolute Error: 0.5025939786059167
K value: 5
Train samples : 1280
Test samples : 319
Mean Squared Error: 0.42411901093096804
Mean Absolute Error: 0.5112863513168298
Average Mean Squared Error across all folds: 0.42584255464233967
```

```
avg_mse = sum(errors) / len(errors)
print(f"Average mean squared error is {avg_mse}")
```

```
Average mean squared error is 0.4258425546423396
```

```
plt.figure(figsize=(8,5))
fig , ax = plt.subplots()
color = ['lightblue', 'blue', 'Cyan', 'lightgrey',"darkgrey","royalblue","darkblue"]
ax.bar(range(1,len(errors)+1),errors,color=color)
ax.set_xlabel("Fold")
ax.set_ylabel("MSE")
ax.set_title("MSE for each fold")
plt.show()
```

```
<Figure size 800x500 with 0 Axes>
```




```
# Visualization of MSE for each fold
plt.figure(figsize=(8, 5))
fig, ax = plt.subplots()

# Define color scheme
colors = ['lightblue', 'blue', 'cyan', 'lightgrey', 'darkgrey', 'royalblue', 'darkblue']

# Create bar plot
ax.bar(range(1, len(errors)+1), errors, color=colors[:len(errors)])

# Set labels and title
ax.set_xlabel("Fold")
ax.set_ylabel("Mean Squared Error (MSE)")
ax.set_title("MSE for each fold in Stratified K-Fold")

# Show the plot
plt.show()
```

```
<Figure size 800x500 with 0 Axes>
```