fetal-health-over-sampled-1

August 10, 2024

[34]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
from imblearn.over_sampling import SMOTE
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
```

[35]:
```
data=pd.read_csv('/kaggle/input/fetal-health-classification/fetal_health.csv')
data.head()
```

[35]:

	baseline value	accelerations	fetal_movement	uterine_contractions	\
0	120.0	0.000	0.0	0.000	
1	132.0	0.006	0.0	0.006	
2	133.0	0.003	0.0	0.008	
3	134.0	0.003	0.0	0.008	
4	132.0	0.007	0.0	0.008	

	light_decelerations	severe_decelerations	prolongued_decelerations	\
0	0.000	0.0	0.0	
1	0.003	0.0	0.0	
2	0.003	0.0	0.0	
3	0.003	0.0	0.0	
4	0.000	0.0	0.0	

	abnormal_short_term_variability	mean_value_of_short_term_variability	\
0	73.0	0.5	
1	17.0	2.1	
2	16.0	2.1	
3	16.0	2.4	
4	16.0	2.4	

	percentage_of_time_with_abnormal_long_term_variability	…	histogram_min	\
0	43.0	…	62.0	
1	0.0	…	68.0	
2	0.0	…	68.0	

```
3                                                 0.0       …           53.0
4                                                 0.0       …           53.0

   histogram_max  histogram_number_of_peaks  histogram_number_of_zeroes  \
0          126.0                        2.0                         0.0
1          198.0                        6.0                         1.0
2          198.0                        5.0                         1.0
3          170.0                       11.0                         0.0
4          170.0                        9.0                         0.0

   histogram_mode  histogram_mean  histogram_median  histogram_variance  \
0           120.0           137.0             121.0                73.0
1           141.0           136.0             140.0                12.0
2           141.0           135.0             138.0                13.0
3           137.0           134.0             137.0                13.0
4           137.0           136.0             138.0                11.0

   histogram_tendency  fetal_health
0                 1.0           2.0
1                 0.0           1.0
2                 0.0           1.0
3                 1.0           1.0
4                 1.0           1.0

[5 rows x 22 columns]
```

[36]:
```
print("Rows:",data.shape[0])
print("Columns:",data.shape[1])
```

```
Rows: 2126
Columns: 22
```

[37]:
```
data.describe()
```

[37]:
```
       baseline value  accelerations  fetal_movement  uterine_contractions  \
count     2126.000000    2126.000000     2126.000000           2126.000000
mean       133.303857       0.003178        0.009481              0.004366
std          9.840844       0.003866        0.046666              0.002946
min        106.000000       0.000000        0.000000              0.000000
25%        126.000000       0.000000        0.000000              0.002000
50%        133.000000       0.002000        0.000000              0.004000
75%        140.000000       0.006000        0.003000              0.007000
max        160.000000       0.019000        0.481000              0.015000

       light_decelerations  severe_decelerations  prolongued_decelerations  \
count          2126.000000           2126.000000               2126.000000
mean              0.001889              0.000003                  0.000159
```

std	0.002960	0.000057	0.000590
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000
75%	0.003000	0.000000	0.000000
max	0.015000	0.001000	0.005000

	abnormal_short_term_variability	mean_value_of_short_term_variability	\
count	2126.000000	2126.000000	
mean	46.990122	1.332785	
std	17.192814	0.883241	
min	12.000000	0.200000	
25%	32.000000	0.700000	
50%	49.000000	1.200000	
75%	61.000000	1.700000	
max	87.000000	7.000000	

	percentage_of_time_with_abnormal_long_term_variability	…	\
count	2126.00000	…	
mean	9.84666	…	
std	18.39688	…	
min	0.00000	…	
25%	0.00000	…	
50%	0.00000	…	
75%	11.00000	…	
max	91.00000	…	

	histogram_min	histogram_max	histogram_number_of_peaks	\
count	2126.000000	2126.000000	2126.000000	
mean	93.579492	164.025400	4.068203	
std	29.560212	17.944183	2.949386	
min	50.000000	122.000000	0.000000	
25%	67.000000	152.000000	2.000000	
50%	93.000000	162.000000	3.000000	
75%	120.000000	174.000000	6.000000	
max	159.000000	238.000000	18.000000	

	histogram_number_of_zeroes	histogram_mode	histogram_mean	\
count	2126.000000	2126.000000	2126.000000	
mean	0.323612	137.452023	134.610536	
std	0.706059	16.381289	15.593596	
min	0.000000	60.000000	73.000000	
25%	0.000000	129.000000	125.000000	
50%	0.000000	139.000000	136.000000	
75%	0.000000	148.000000	145.000000	
max	10.000000	187.000000	182.000000	

```
       histogram_median  histogram_variance  histogram_tendency  fetal_health
count       2126.000000         2126.000000         2126.000000   2126.000000
mean         138.090310           18.808090            0.320320      1.304327
std           14.466589           28.977636            0.610829      0.614377
min           77.000000            0.000000           -1.000000      1.000000
25%          129.000000            2.000000            0.000000      1.000000
50%          139.000000            7.000000            0.000000      1.000000
75%          148.000000           24.000000            1.000000      1.000000
max          186.000000          269.000000            1.000000      3.000000

[8 rows x 22 columns]
```

[38]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2126 entries, 0 to 2125
Data columns (total 22 columns):
 #   Column                                                Non-Null Count
Dtype
---  ------                                                --------------
-----
 0   baseline value                                        2126 non-null
float64
 1   accelerations                                         2126 non-null
float64
 2   fetal_movement                                        2126 non-null
float64
 3   uterine_contractions                                  2126 non-null
float64
 4   light_decelerations                                   2126 non-null
float64
 5   severe_decelerations                                  2126 non-null
float64
 6   prolongued_decelerations                              2126 non-null
float64
 7   abnormal_short_term_variability                       2126 non-null
float64
 8   mean_value_of_short_term_variability                  2126 non-null
float64
 9   percentage_of_time_with_abnormal_long_term_variability  2126 non-null
float64
 10  mean_value_of_long_term_variability                   2126 non-null
float64
 11  histogram_width                                       2126 non-null
float64
 12  histogram_min                                         2126 non-null
float64
```

```
 13  histogram_max                                          2126 non-null
float64
 14  histogram_number_of_peaks                              2126 non-null
float64
 15  histogram_number_of_zeroes                             2126 non-null
float64
 16  histogram_mode                                         2126 non-null
float64
 17  histogram_mean                                         2126 non-null
float64
 18  histogram_median                                       2126 non-null
float64
 19  histogram_variance                                     2126 non-null
float64
 20  histogram_tendency                                     2126 non-null
float64
 21  fetal_health                                           2126 non-null
float64
dtypes: float64(22)
memory usage: 365.5 KB
```

[39]:
```
data.duplicated()
```

[39]:
```
0       False
1       False
2       False
3       False
4       False
        …
2121    False
2122    False
2123    False
2124    False
2125    False
Length: 2126, dtype: bool
```

[40]:
```
data.notnull()
```

[40]:
```
      baseline value  accelerations  fetal_movement  uterine_contractions  \
0               True           True            True                  True
1               True           True            True                  True
2               True           True            True                  True
3               True           True            True                  True
4               True           True            True                  True
…               …              …               …                     …
2121            True           True            True                  True
2122            True           True            True                  True
```

```
2123            True            True             True                    True
2124            True            True             True                    True
2125            True            True             True                    True

      light_decelerations  severe_decelerations  prolongued_decelerations  \
0                    True                  True                      True
1                    True                  True                      True
2                    True                  True                      True
3                    True                  True                      True
4                    True                  True                      True
…                     …                     …                         …
2121                 True                  True                      True
2122                 True                  True                      True
2123                 True                  True                      True
2124                 True                  True                      True
2125                 True                  True                      True

      abnormal_short_term_variability  mean_value_of_short_term_variability  \
0                                True                                  True
1                                True                                  True
2                                True                                  True
3                                True                                  True
4                                True                                  True
…                                 …                                     …
2121                             True                                  True
2122                             True                                  True
2123                             True                                  True
2124                             True                                  True
2125                             True                                  True

      percentage_of_time_with_abnormal_long_term_variability  …  \
0                                                     True     …
1                                                     True     …
2                                                     True     …
3                                                     True     …
4                                                     True     …
…                                                      …       …
2121                                                  True     …
2122                                                  True     …
2123                                                  True     …
2124                                                  True     …
2125                                                  True     …

      histogram_min  histogram_max  histogram_number_of_peaks  \
0              True           True                       True
1              True           True                       True
2              True           True                       True
```

```
3                True                True                          True
4                True                True                          True
…                …                   …                             …
2121             True                True                          True
2122             True                True                          True
2123             True                True                          True
2124             True                True                          True
2125             True                True                          True

      histogram_number_of_zeroes  histogram_mode  histogram_mean  \
0                           True            True            True
1                           True            True            True
2                           True            True            True
3                           True            True            True
4                           True            True            True
…                           …               …               …
2121                        True            True            True
2122                        True            True            True
2123                        True            True            True
2124                        True            True            True
2125                        True            True            True

      histogram_median  histogram_variance  histogram_tendency  fetal_health
0                 True                True                True          True
1                 True                True                True          True
2                 True                True                True          True
3                 True                True                True          True
4                 True                True                True          True
…                 …                   …                   …             …
2121              True                True                True          True
2122              True                True                True          True
2123              True                True                True          True
2124              True                True                True          True
2125              True                True                True          True

[2126 rows x 22 columns]
```

[41]:
```
data.keys()
```

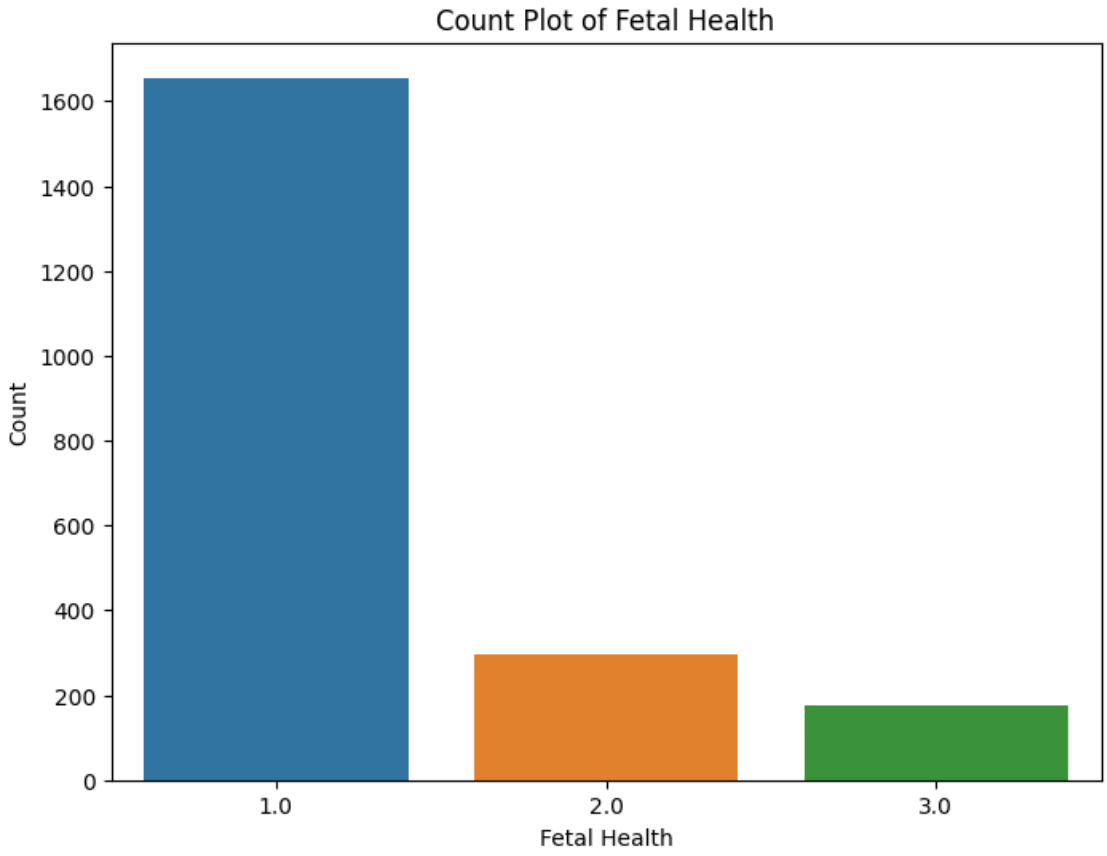
[41]:
```
Index(['baseline value', 'accelerations', 'fetal_movement',
       'uterine_contractions', 'light_decelerations', 'severe_decelerations',
       'prolongued_decelerations', 'abnormal_short_term_variability',
       'mean_value_of_short_term_variability',
       'percentage_of_time_with_abnormal_long_term_variability',
       'mean_value_of_long_term_variability', 'histogram_width',
       'histogram_min', 'histogram_max', 'histogram_number_of_peaks',
       'histogram_number_of_zeroes', 'histogram_mode', 'histogram_mean',
```

```
       'histogram_median', 'histogram_variance', 'histogram_tendency',
       'fetal_health'],
      dtype='object')
```

[42]:
```
# Analysing the target column
data['fetal_health'].unique()
```

[42]:
```
array([2., 1., 3.])
```

[43]:
```
# Plot count plot
plt.figure(figsize=(8, 6))
sns.countplot(x='fetal_health', data=data)
plt.title('Count Plot of Fetal Health')
plt.xlabel('Fetal Health')
plt.ylabel('Count')
plt.show()
```



[44]:
```
import matplotlib.pyplot as plt
import numpy as np
```

```
# Define the figure size
plt.figure(figsize=(15, 8))  # Adjust the width and height according to your␣
 ↪preference

# List of features
features = ['baseline value', 'accelerations', 'fetal_movement',
            'uterine_contractions', 'light_decelerations',␣
 ↪'severe_decelerations',
            'prolongued_decelerations', 'abnormal_short_term_variability',
            'mean_value_of_short_term_variability',
            'percentage_of_time_with_abnormal_long_term_variability',
            'mean_value_of_long_term_variability', 'histogram_width',
            'histogram_min', 'histogram_max', 'histogram_number_of_peaks',
            'histogram_number_of_zeroes', 'histogram_mode', 'histogram_mean',
            'histogram_median', 'histogram_variance', 'histogram_tendency']

# Plotting each feature against the target variable 'fetal_health'
num_features = len(features)
num_rows = num_features // 3 + (1 if num_features % 3 != 0 else 0)  # Calculate␣
 ↪the number of rows needed
fig, axes = plt.subplots(nrows=num_rows, ncols=3, figsize=(15, 4*num_rows))

for i, feature in enumerate(features):
    row = i // 3  # Calculate the row index
    col = i % 3   # Calculate the column index
    ax = axes[row, col] if num_rows > 1 else axes[col]  # Select the␣
 ↪appropriate axis
    ax.scatter(data[feature], data['fetal_health'], cmap='viridis')
    ax.set_ylabel('Fetal Health')
    ax.set_xlabel(feature)
    ax.set_title(f'Scatter Plot of {feature} against Fetal Health')
    ax.grid(True)

# Remove empty subplots
if num_features % 3 != 0:
    for j in range(num_features % 3, 3):
        fig.delaxes(axes[num_rows - 1, j])

plt.tight_layout()
plt.show()
```

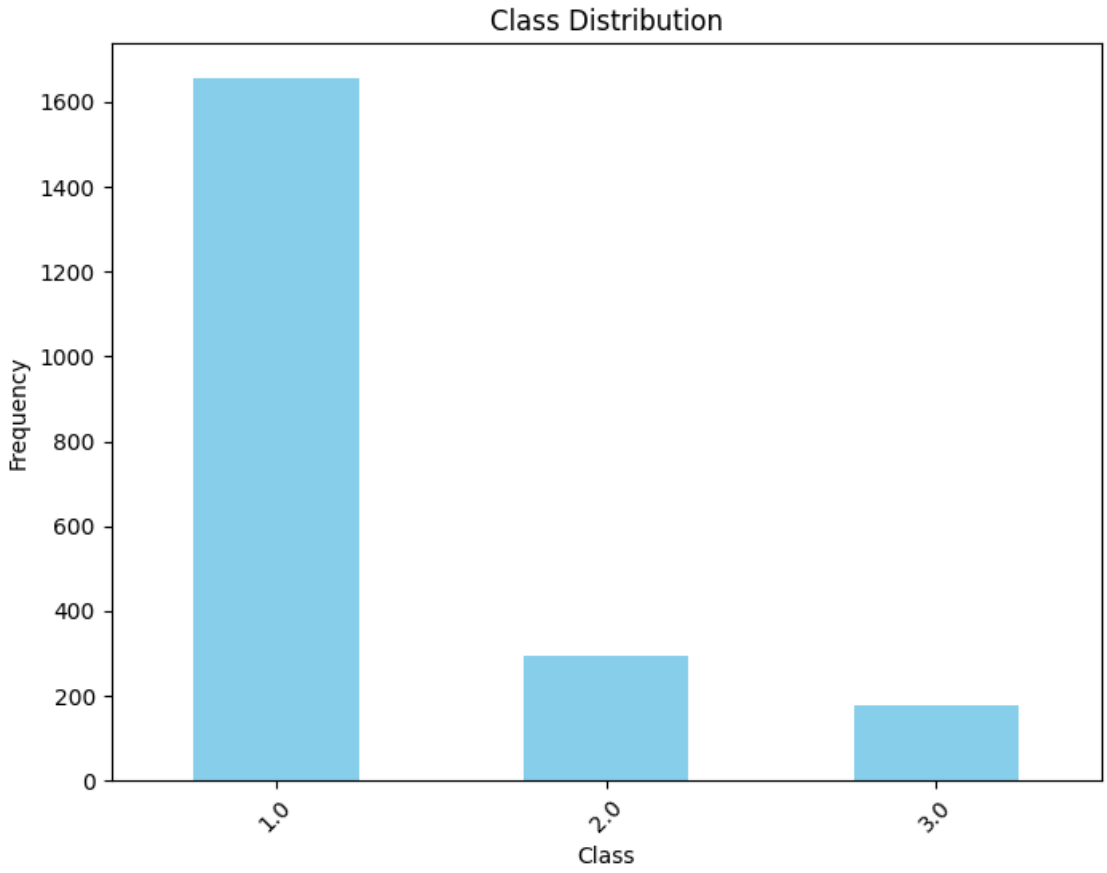
```
/tmp/ipykernel_33/2953099270.py:27: UserWarning: No data for colormapping
provided via 'c'. Parameters 'cmap' will be ignored
  ax.scatter(data[feature], data['fetal_health'], cmap='viridis')
```

```
<Figure size 1500x800 with 0 Axes>
```


Scatter Plot of baseline value against Fetal Health
baseline value
Scatter Plot of accelerations against Fetal Health
accelerations
Scatter Plot of fetal_movement against Fetal Health
fetal_movement
Scatter Plot of uterine_contractions against Fetal Health
uterine_contractions
Scatter Plot of light_decelerations against Fetal Health
light_decelerations
Scatter Plot of severe_decelerations against Fetal Health
severe_decelerations
Scatter Plot of prolongued_decelerations against Fetal Health
prolongued_decelerations
Scatter Plot of abnormal_short_term_variability against Fetal Health
abnormal_short_term_variability
Scatter Plot of mean_value_of_short_term_variability against Fetal Health
mean_value_of_short_term_variability
Scatter Plot of percentage_of_time_with_abnormal_long_term_variability against Fetal Health
percentage_of_time_with_abnormal_long_term_variability
Scatter Plot of mean_value_of_long_term_variability against Fetal Health
mean_value_of_long_term_variability
Scatter Plot of histogram_width against Fetal Health
histogram_width
Scatter Plot of histogram_min against Fetal Health
histogram_min
Scatter Plot of histogram_max against Fetal Health
histogram_max
Scatter Plot of histogram_number_of_peaks against Fetal Health
histogram_number_of_peaks
Scatter Plot of histogram_number_of_zeroes against Fetal Health
histogram_number_of_zeroes
Scatter Plot of histogram_mode against Fetal Health
histogram_mode
Scatter Plot of histogram_mean against Fetal Health
histogram_mean
Scatter Plot of histogram_median against Fetal Health
histogram_median
Scatter Plot of histogram_variance against Fetal Health
histogram_variance
Scatter Plot of histogram_tendency against Fetal Health
histogram_tendency
Fetal Health

[45]:
```
# Assuming 'df' is your DataFrame and 'target_column' is the column containing␣
↪class labels
class_distribution = data['fetal_health'].value_counts()
print("Class distribution:")
print(class_distribution)

# Plot class distribution
plt.figure(figsize=(8, 6))
class_distribution.plot(kind='bar', color='skyblue')
plt.title('Class Distribution')
plt.xlabel('Class')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()

# Calculate imbalance ratio
imbalance_ratio = class_distribution.min() / class_distribution.max()
print("Imbalance ratio:", imbalance_ratio)
```

```
Class distribution:
fetal_health
1.0    1655
2.0     295
3.0     176
Name: count, dtype: int64
```



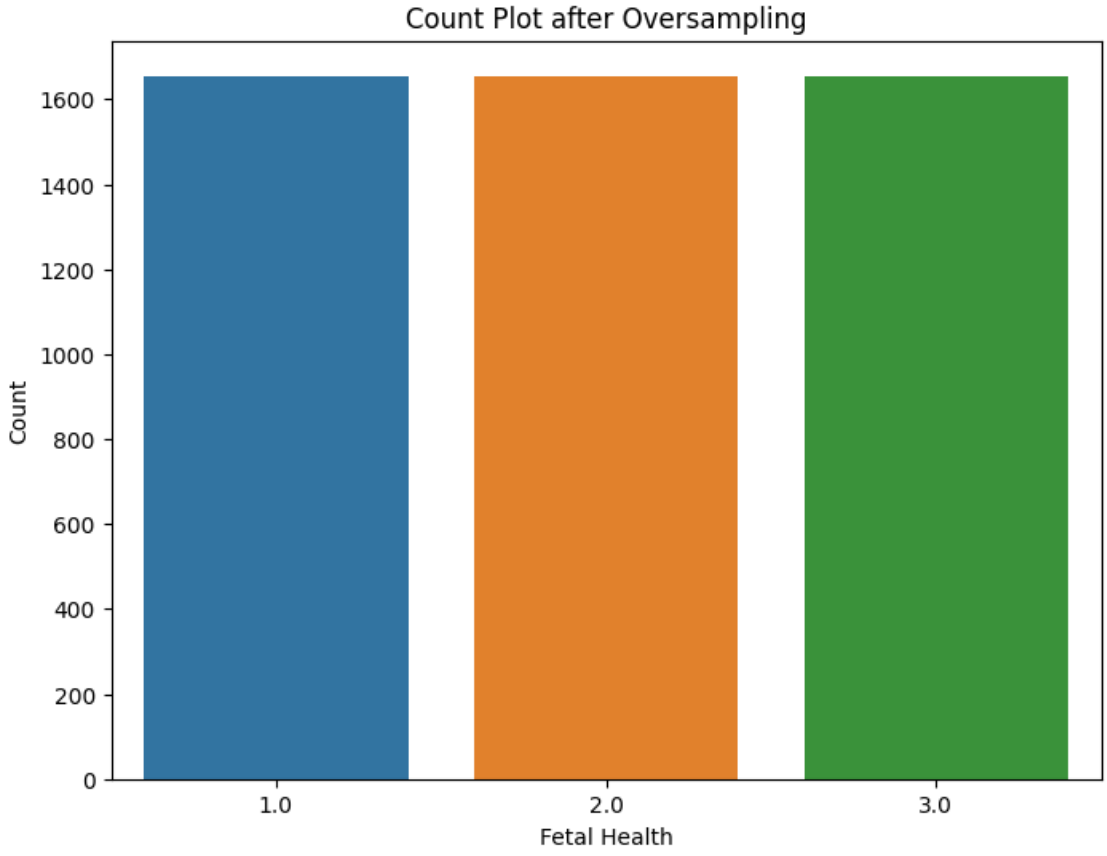

```
Imbalance ratio: 0.10634441087613293
```

[46]:
```
# Assuming 'df' is your DataFrame and 'fetal_health' is the target variable
X = data.drop(columns=['fetal_health'])  # Features
y = data['fetal_health']  # Target variable

# Initialize SMOTE
smote = SMOTE()

# Perform SMOTE oversampling
X_resampled, y_resampled = smote.fit_resample(X, y)

# Convert the resampled data to DataFrame
df_resampled = pd.concat([pd.DataFrame(X_resampled, columns=X.columns), pd.
 ↪DataFrame(y_resampled, columns=['fetal_health'])], axis=1)

# Check the class distribution after oversampling
print("Class distribution after oversampling:")
print(df_resampled['fetal_health'].value_counts())
```

```
Class distribution after oversampling:
fetal_health
2.0    1655
1.0    1655
3.0    1655
Name: count, dtype: int64
```

[47]:
```
df_resampled
x=df_resampled.drop('fetal_health',axis=1)
y=df_resampled['fetal_health']
```

[48]:
```
# Plot count plot
plt.figure(figsize=(8, 6))
sns.countplot(x='fetal_health', data=df_resampled)
plt.title('Count Plot after Oversampling')
plt.xlabel('Fetal Health')
plt.ylabel('Count')
plt.show()
```



[49]:
```
features = ['baseline value', 'accelerations', 'fetal_movement',␣
 ↪'uterine_contractions','light_decelerations',
           ␣
 ↪'severe_decelerations','prolongued_decelerations','abnormal_short_term_variability','mean_v
          ␣
 ↪'percentage_of_time_with_abnormal_long_term_variability','mean_value_of_long_term_variabili
           ␣
 ↪'histogram_min','histogram_max','histogram_number_of_peaks','histogram_number_of_zeroes','h
           'histogram_median','histogram_variance','histogram_tendency']

# Separating out the features
x = df_resampled.loc[:, features].values

# Separating out the target
y = df_resampled.loc[:,['fetal_health']].values
```

[50]:
```
x = StandardScaler().fit_transform(df_resampled)
```

[51]:
```
y.shape
```

[51]:
```
(4965, 1)
```

[52]:
```
from sklearn.decomposition import PCA

pca = PCA(n_components=5)

principalComponents = pca.fit_transform(x)

principalDf = pd.DataFrame(data = principalComponents
             , columns = ['principal component 1', 'principal component␣
 ↪2','principal component 3','principal component 4','principal component 5'])
finalDf=principalDf
```

[53]:
```
#finalDf = pd.concat([principalDf, data[['fetal_health']]], axis = 1)
```

[54]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix
from sklearn.metrics import classification_report
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
```

[55]:
```
y.shape
```

[55]:
```
(4965, 1)
```

[56]:
```
finalDf.shape
```

[56]:
```
(4965, 5)
```

[57]:
```
X_train, X_test, y_train, y_test = train_test_split(finalDf, y, test_size=0.4)
```

[58]:
```
X_train
```

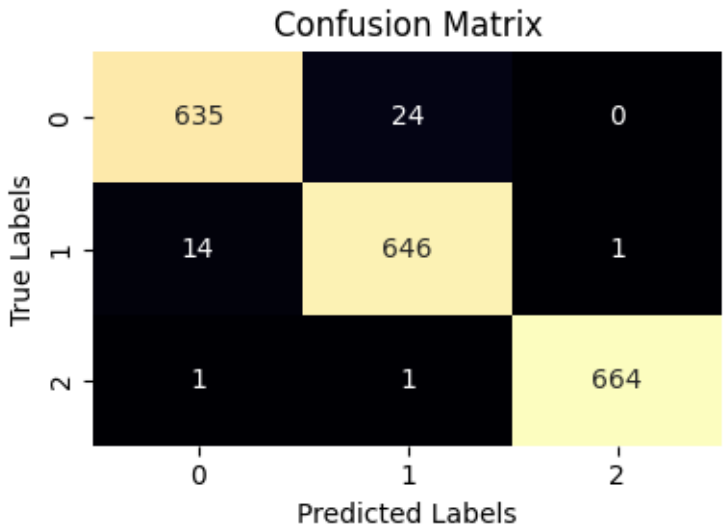
[58]:
```
      principal component 1  principal component 2  principal component 3  \
1587               0.956878               1.569415               0.588985
2343              -2.531951              -0.828686              -0.077501
2499              -1.368807               1.482100              -0.344704
4502               4.359033              -0.496193              -0.316583
1976              -1.492004               1.287881               1.256608
…                       …                      …                      …
3276              -3.047368              -1.533019              -0.419620
4902               5.815971              -1.661513               0.439983
2641              -2.387667              -0.701829               0.287404
3016              -1.488183               0.712646              -0.675620
3876               3.328543              -1.426344               1.187249

      principal component 4  principal component 5
1587               1.228358               1.344599
2343              -0.286005              -0.253239
2499              -0.359640               0.394669
4502              -1.633983               1.208086
1976              -2.469014              -1.108613
…                       …                      …
3276              -0.270883              -0.137230
4902              -1.260138              -0.765890
2641              -0.104080              -0.544272
3016               1.389087               1.081168
3876               1.155247               1.347318

[2979 rows x 5 columns]
```

[84]:
```
clf1 = RandomForestClassifier()
clf1.fit(X_train, y_train)

# Making predictions
y1_pred = clf1.predict(X_test)

# Evaluating the model
acc1 = accuracy_score(y_test, y1_pred)
print("Random forest Accuracy:", accuracy1)
print(classification_report(y_test, y1_pred))
conf_matrix = confusion_matrix(y_test, y1_pred)
```

```
plt.figure(figsize=(4, 3))
sns.heatmap(conf_matrix, annot=True, cmap='magma', fmt='g', cbar=False)

# Add labels, title, and adjust layout
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.tight_layout()

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

```
/tmp/ipykernel_33/3900602095.py:2: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  clf1.fit(X_train, y_train)
```

```
Random forest Accuracy: 0.9803625377643505
              precision    recall  f1-score   support

         1.0       0.98      0.96      0.97       659
         2.0       0.96      0.98      0.97       661
         3.0       1.00      1.00      1.00       666

    accuracy                           0.98      1986
   macro avg       0.98      0.98      0.98      1986
weighted avg       0.98      0.98      0.98      1986
```



[87]:
```
# Create Decision Tree classifer object
clf2 = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf2 = clf2.fit(X_train,y_train)

# Predict the response for test dataset
y2_pred = clf2.predict(X_test)
acc2=accuracy_score(y_test, y2_pred)
# Model Accuracy, how often is the classifier correct?
print("Decision tree Accuracy:",acc2 )
print(classification_report(y_test, y2_pred))
```

```
Decision tree Accuracy: 0.9652567975830816
              precision    recall  f1-score   support

         1.0       0.97      0.94      0.96       659
         2.0       0.94      0.96      0.95       661
         3.0       0.99      0.99      0.99       666

    accuracy                           0.97      1986
   macro avg       0.97      0.97      0.97      1986
weighted avg       0.97      0.97      0.97      1986
```

[88]:
```
# Create KNN classifier object
clf3 = KNeighborsClassifier(n_neighbors=3) # You can adjust the number of␣
 ↪neighbors (k) here

# Train KNN classifier
clf3.fit(X_train, y_train)

# Predict the response for test dataset
y3_pred = clf3.predict(X_test)
acc3=accuracy_score(y_test, y3_pred)
# Model Accuracy, how often is the classifier correct?
print("KNN Accuracy:", acc3)
print(classification_report(y_test, y3_pred))
```

```
KNN Accuracy: 0.9813695871097684
              precision    recall  f1-score   support

         1.0       1.00      0.95      0.97       659
         2.0       0.95      0.99      0.97       661
         3.0       1.00      1.00      1.00       666

    accuracy                           0.98      1986
```

```
   macro avg       0.98      0.98      0.98      1986
weighted avg       0.98      0.98      0.98      1986
```

```
/opt/conda/lib/python3.10/site-
packages/sklearn/neighbors/_classification.py:215: DataConversionWarning: A
column-vector y was passed when a 1d array was expected. Please change the shape
of y to (n_samples,), for example using ravel().
  return self._fit(X, y)
```

[89]:
```
# Create SVM classifier object
clf4 = SVC()

# Train SVM classifier
clf4.fit(X_train, y_train)

# Predict the response for test dataset
y4_pred = clf4.predict(X_test)
acc4=accuracy_score(y_test, y4_pred)
# Model Accuracy, how often is the classifier correct?
print("SVC Accuracy:",acc4 )
print(classification_report(y_test, y4_pred))
```

```
SVC Accuracy: 0.9602215508559919
              precision    recall  f1-score   support

         1.0       0.98      0.93      0.95       659
         2.0       0.92      0.96      0.94       661
         3.0       0.98      0.99      0.98       666

    accuracy                           0.96      1986
   macro avg       0.96      0.96      0.96      1986
weighted avg       0.96      0.96      0.96      1986
```

```
/opt/conda/lib/python3.10/site-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

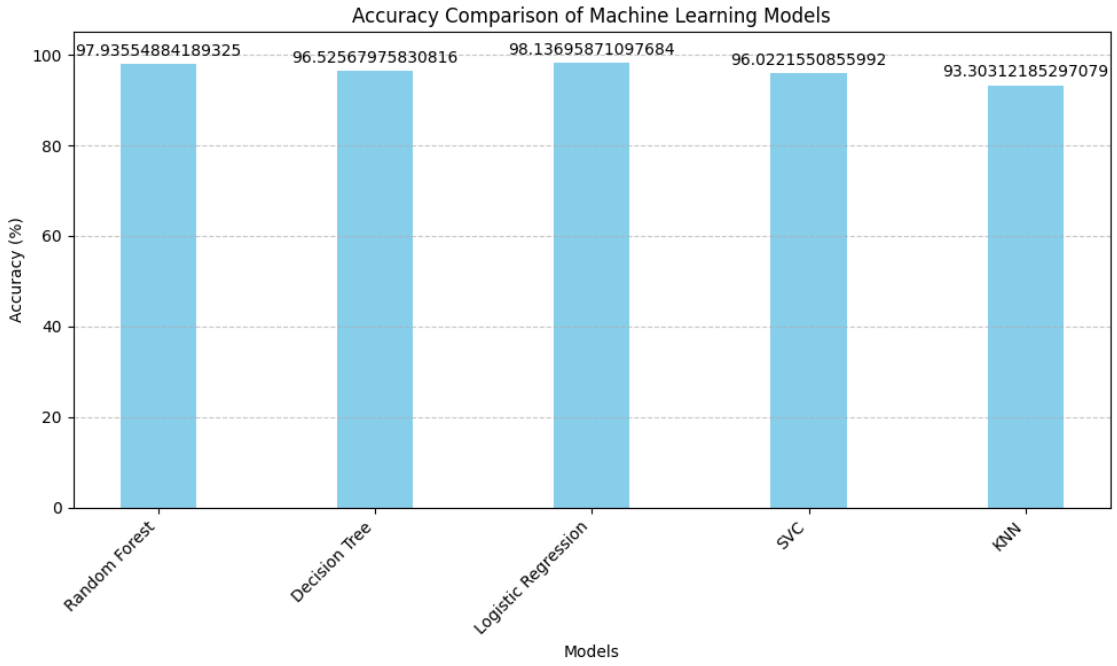
[90]:
```
from sklearn.linear_model import LogisticRegression

# Assuming X_train_pca and y_train are the PCA-transformed training data and␣
 ↪corresponding labels
# Initialize logistic regression model
lr = LogisticRegression()
```

```
# Train logistic regression on the transformed data
lr.fit(X_train, y_train)
y5_pred = lr.predict(X_test)
acc5=accuracy_score(y_test, y5_pred)
# Model Accuracy, how often is the classifier correct?
print("Logistic regression Accuracy:",acc5 )
print(classification_report(y_test, y5_pred))
```

```
Logistic regression Accuracy: 0.9330312185297079
              precision    recall  f1-score   support

         1.0       0.95      0.93      0.94       659
         2.0       0.90      0.90      0.90       661
         3.0       0.95      0.96      0.95       666

    accuracy                           0.93      1986
   macro avg       0.93      0.93      0.93      1986
weighted avg       0.93      0.93      0.93      1986


/opt/conda/lib/python3.10/site-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

[91]:
```
import matplotlib.pyplot as plt
import numpy as np

# Data
models = ['Random Forest', 'Decision Tree', 'Logistic Regression', 'SVC',␣
 ↪'KNN']  # Names of the models
accuracies = [acc1*100, acc2*100, acc3*100,acc4*100,acc5*100]  # Accuracies of␣
 ↪the models

# Create a figure and axis object
fig, ax = plt.subplots(figsize=(10, 6))

# Define width of each bar
bar_width = 0.35

# Index for the x-axis (position of each group)
index = np.arange(len(models))

# Plot the bars for accuracies
bars = ax.bar(index, accuracies, bar_width, label='Accuracy', color='skyblue')
```

```
# Add labels, title, and grid
ax.set_xlabel('Models')
ax.set_ylabel('Accuracy (%)')
ax.set_title('Accuracy Comparison of Machine Learning Models')
ax.set_xticks(index)
ax.set_xticklabels(models, rotation=45, ha='right')
ax.set_ylim(0, 105)
ax.grid(axis='y', linestyle='--', alpha=0.7)

# Add labels on top of the bars
for bar in bars:
    height = bar.get_height()
    ax.annotate('{}'.format(height),
                xy=(bar.get_x() + bar.get_width() / 2, height),
                xytext=(0, 3),  # 3 points vertical offset
                textcoords="offset points",
                ha='center', va='bottom')

plt.tight_layout()
plt.show()
```



[65]:
```
p=data
data
p_x=data.drop('fetal_health',axis=1)
p_y=data['fetal_health']
```

[66]:
```
X_t, X_te, y_t, y_te = train_test_split(p_x, p_y, test_size=0.4)
c1 = LogisticRegression()
c1.fit(X_t, y_t)

# Making predictions
y_pr = c1.predict(X_te)

# Evaluating the model
acc1 = c1.score(X_te, y_te)
print("Random forest Accuracy:", acc1)
print(classification_report(y_te, y_pr))
```

```
Random forest Accuracy: 0.8719153936545241
              precision    recall  f1-score   support

         1.0       0.90      0.97      0.93       670
         2.0       0.60      0.39      0.47       111
         3.0       0.85      0.74      0.79        70

    accuracy                           0.87       851
   macro avg       0.78      0.70      0.73       851
weighted avg       0.86      0.87      0.86       851


/opt/conda/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:458:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```