Feature Selection using Fisher Score and Chi2 (χ2) Test

What is Fisher Score and Chi2 (χ2) Test

Fisher score is one of the most widely used supervised feature selection methods. However, it selects each feature independently according to their scores under the Fisher criterion, which leads to a suboptimal subset of features

Chi Square (χ2) Test

A chi-squared test, also written as χ2 test, is any statistical hypothesis test where the sampling distribution of the test statistic is a chi-squared distribution.

chi-square test measures dependence between stochastic variables, so using this function "weeds out" the features that are the most likely to be independent of class and therefore irrelevant for classification.

In [31]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [32]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier

from sklearn.feature_selection import chi2
from sklearn.feature_selection import SelectKBest, SelectPercentile
from sklearn.metrics import accuracy_score
```

In [33]:
```
data = pd.read_csv('0.9_5subjectslabelled_data.csv', nrows = 31437)
data.head()
```

Out[33]:

	Time Snap	AccX	AccY	AccZ	Gyro_X	Knee Angles	Gait Cycle Phase
0	120.775	-0.181472	-0.088708	-0.665352	0.087145	67.223821	5
1	120.780	-0.181443	-0.088745	-0.659870	0.103506	67.217858	5
2	120.785	-0.183826	-0.089735	-0.654215	0.117635	67.154903	5
3	120.790	-0.188545	-0.091706	-0.648504	0.129259	67.011479	5
4	120.795	-0.195535	-0.094645	-0.642857	0.138193	66.799616	5

In [34]:
```
X = data.drop('Gait Cycle Phase', axis = 1)
y = data['Gait Cycle Phase']

X.shape, y.shape
```

Out[34]:
```
((31436, 6), (31436,))
```

In [35]:
```
data.isnull().sum()
```

Out[35]:
```
Time Snap           0
AccX                0
AccY                0
AccZ                0
Gyro_X              0
Knee Angles         0
Gait Cycle Phase    0
dtype: int64
```

In [36]:
```
data.drop(labels = ['Time Snap', 'AccX', 'AccY', 'AccZ', 'Gyro_X', 'Knee Angle
```

In [37]:
```
data = data.dropna()
```

In [38]:
```
data.isnull().sum()
```

Out[38]:
```
Gait Cycle Phase    0
dtype: int64
```

In [39]:
```
data = data[['Gait Cycle Phase']].copy()
```

In [40]:
```
data.head()
```

Out[40]:

	Gait Cycle Phase
0	5
1	5
2	5
3	5
4	5

In [41]:
```
data.isnull().sum()
```

Out[41]:
```
Gait Cycle Phase    0
dtype: int64
```

Do F_Score

In [42]:
```
X = data.copy()
y = data['Gait Cycle Phase']
```

In [43]:
```
X.shape, y.shape
```

Out[43]:
```
((31436, 1), (31436,))
```

In [44]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, ran
```

In [45]:
```
f_score = chi2(X_train, y_train)
```

In [46]:
```
f_score
```

Out[46]:
```
(array([31832.62402445]), array([0.]))
```

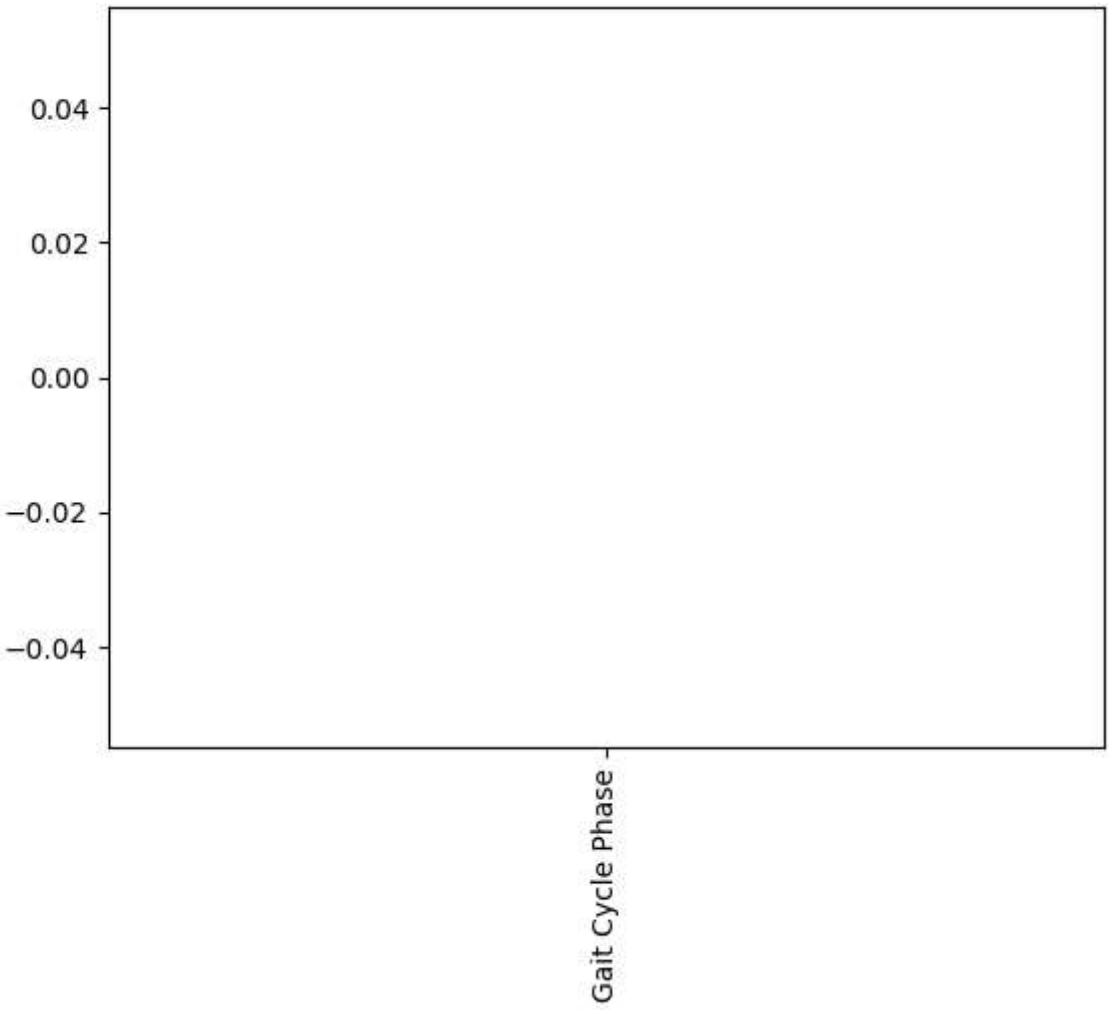
In [47]:
```
p_values = pd.Series(f_score[1], index = X_train.columns)
p_values.sort_values(ascending = True, inplace = True)
```

In [48]:
```
p_values
```

Out[48]:
```
Gait Cycle Phase    0.0
dtype: float64
```

In [49]:
```
p_values.plot.bar()
```

Out[49]: <Axes: >



In [50]:
```
X_train_2 = X_train[['Gait Cycle Phase']]
X_test_2 = X_test[['Gait Cycle Phase']]
```

In [51]:
```
def run_randomForest(X_train, X_test, y_train, y_test):
    clf = RandomForestClassifier(n_estimators=100, random_state=0, n_jobs=-1)
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    print('Accuracy: ', accuracy_score(y_test, y_pred))
```

In [52]:
```
%%time
run_randomForest(X_train_2, X_test_2, y_train, y_test)
```

```
Accuracy:  1.0
CPU times: total: 2.72 s
Wall time: 2.19 s
```

In [ ]:
```

```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: