ASSQ2

April 12, 2024

1 Question2 a

[1]:
```
#Q2 reading the data into Python
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.stats.api as sms
import statsmodels.api as sm
from statsmodels.stats.stattools import durbin_watson
from scipy import stats
from sklearn.utils import resample
import scipy;
df2 = pd.read_csv("nassCDS.csv");
print(df2.head())
```

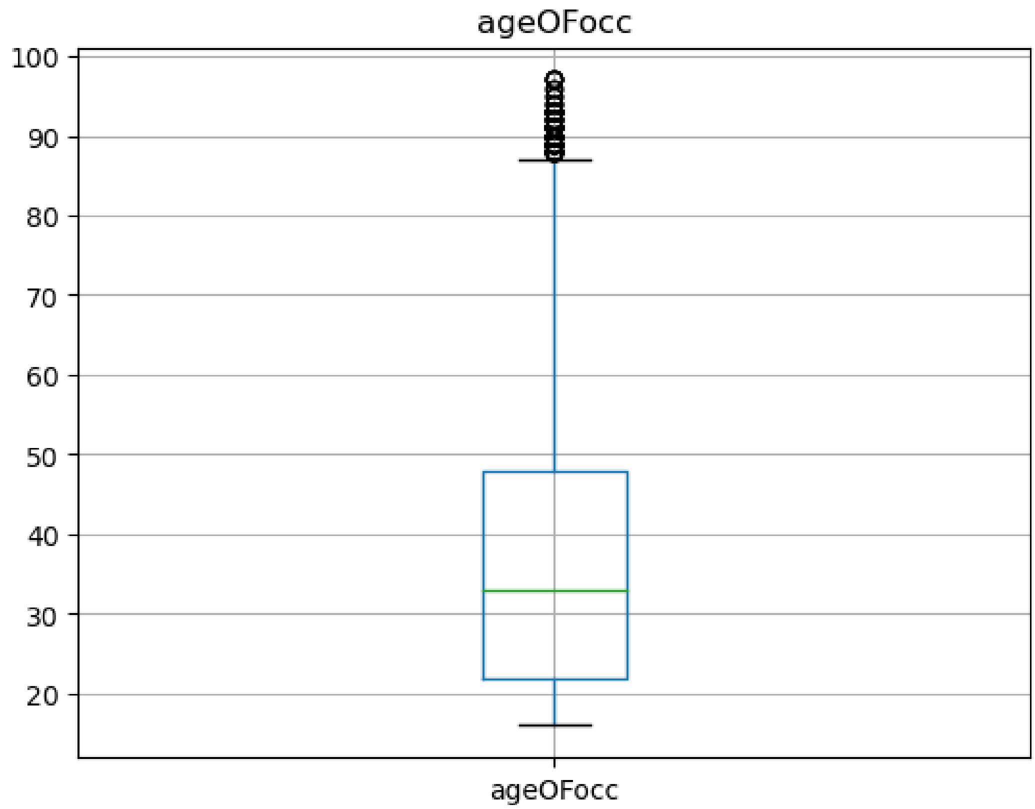
```
   rownames  dvcat   weight   dead   airbag seatbelt  frontal sex  ageOFocc  \
0         1  25-39   25.069  alive     none   belted        1   f        26
1         2  10-24   25.069  alive  airbag   belted        1   f        72
2         3  10-24   32.379  alive     none     none        1   f        69
3         4  25-39  495.444  alive  airbag   belted        1   f        53
4         5  25-39   25.069  alive     none   belted        1   f        32

   yearacc  yearVeh    abcat occRole  deploy  injSeverity  caseid
0     1997   1990.0  unavail  driver       0          3.0   2:3:1
1     1997   1995.0   deploy  driver       1          1.0   2:3:2
2     1997   1988.0  unavail  driver       0          4.0   2:5:1
3     1997   1995.0   deploy  driver       1          1.0  2:10:1
4     1997   1988.0  unavail  driver       0          3.0  2:11:1
```

[2]:
```
# Q1a Data preprocessing.
print("Number of observation: ", df2.shape[0])      # check dimension
print("Any NA value:", df2.isnull().values.any()); # Check for missing values
print("Any row duplictaes:",df2.duplicated().any());# check for dupllicates rows
```

```python
df2 = df2.dropna() # drop all the NA values
#check for date error among all the variables of interestes.
print("Number of error values in 'dead':", ((df2['dead']!= "alive")&␣
 ↪(df2['dead'] != "dead")).sum())
print("Number of error values in 'seltbelt':", ((df2['seatbelt']!= "belted")&␣
 ↪(df2['seatbelt'] != "none")).sum())
print("Number of error values in 'frontal':", ((df2['frontal']!= 0)&␣
 ↪(df2['frontal'] != 1)).sum())
print("Number of error values in 'airbag':", ((df2['airbag']!= "none")&␣
 ↪(df2['airbag'] != "airbag")).sum())
print("Number of error values in 'sex':", ((df2['sex']!= "m")& (df2['sex'] !=␣
 ↪"f")).sum())
print("Number of error values in 'sex':", ((df2['sex']!= "m")& (df2['sex'] !=␣
 ↪"f")).sum())
print("Number of error values in 'ageOFocc':", ((df2['ageOFocc']<0) |␣
 ↪(df2['ageOFocc']>100)).sum())
print("Number of error values in 'deploy':", ((df2['deploy']!= 1)&␣
 ↪(df2['deploy'] != 0)).sum())
# Check outlier for numeric variable 'ageOFocc'
df2.boxplot("ageOFocc")
plt.title('ageOFocc')
plt.tight_layout
plt.show()
# Check data types
print(df2.dtypes)
# Check for data balancing
response_count = df2.groupby("dead")["dead"].count();
print(response_count);
print("Percentage of alive:", 100*response_count[0]/np.sum(response_count));
print("Percentage of dead:", 100*response_count[1]/np.sum(response_count));
print(df2.shape)
df2.reset_index(drop=True, inplace=True)
```

```
Number of observation:  26217
Any NA value: True
Any row duplictaes: False
Number of error values in 'dead': 0
Number of error values in 'seltbelt': 0
Number of error values in 'frontal': 0
Number of error values in 'airbag': 0
Number of error values in 'sex': 0
Number of error values in 'sex': 0
Number of error values in 'ageOFocc': 0
Number of error values in 'deploy': 0
```




```
rownames         int64
dvcat           object
weight         float64
dead            object
airbag          object
seatbelt        object
frontal          int64
sex             object
ageOFocc         int64
yearacc          int64
yearVeh        float64
abcat           object
occRole         object
deploy           int64
injSeverity    float64
caseid          object
dtype: object
dead
alive    24883
dead      1180
Name: dead, dtype: int64
```

```
Percentage of alive: 95.47250892069216
Percentage of dead: 4.527491079307831
(26063, 16)
```

In this dataset, we have 26217 observations with missing values and no duplicate rows. There is no obvious data error in the dataset, as all the values are plausible. There are some outliers on the upper side in age, as indicated by the box plot. Since we work with categorical variables, there is no need to perform any standardization. However, feature selection plays a crucial role in the later part of this question, such as finding the relation of two categorical variables(Chi-square, ANOVA). More importantly, we have unbalanced data in this question, and we are going to use oversampling techniques to balance it(This is performed in later parts).Before the analysis, we drop all the NA values.

2 Question2 b

[3]:
```python
#chi-square is used to determine whether two categorical are independent or not␣
 ↪("seatbelt" and "dead")
from scipy.stats import chi2_contingency
# Converting the characters in data set into 0s and 1s for simplicity.
# Replace 'alive' with 1 and 'dead' with 0
df2['dead'].replace({'alive': 1, 'dead': 0}, inplace=True)
# Replace 'belted' with 1 and 'none' with 0
df2['seatbelt'].replace({'belted': 1, 'none': 0},inplace = True)
# Replace 'airbag' with 1 and 'none' with 0
df2['airbag'].replace({'airbag': 1, 'none': 0},inplace = True)
# Replace 'm' with 1 and 'f' with 0
df2['sex'].replace({'m': 1, 'f': 0},inplace = True)

# Now we convert 'seatbelt' and 'dead' to category type for Chi-square analysis
df_chi = df2[["seatbelt","dead"]].astype("category")
# Hypothesis:
#H0: the features are independent
#H1: the features are not independent
contingency_table = pd.crosstab(df_chi['seatbelt'], df_chi['dead'])# Generate␣
 ↪contigency table

# Perform the Chi-square test
chi2_stat, p_value, dof, expected = chi2_contingency(contingency_table)
print("Statistics:",chi2_stat)
print("p-value:", round(p_value,2))
print("Degrees of freedom:", dof)
```

```
Statistics: 483.7579238069683
p-value: 0.0
Degrees of freedom: 1
```

Since the P-value is approximately zero, we have very strong evidence against the null hypothesis. We have strong evidence that 'seatbelt' and 'dead' are not independent, which is what we expect in *real life. In conclusion, we have enough evidence to keep the variable 'seatbelt' in the analysis that aims to explain the variable 'dead'.*

3 Question2 c

[4]:
```
# ANOVA is used to analyze the mean age difference between injury severity␣
↪groups.
from scipy.stats import ttest_ind
from scipy.stats import f_oneway

df_none = df2[df2["injSeverity"]== 0]; # dataset for none injury
df_possible = df2[df2["injSeverity"]== 1]; # dataset for possible injury
df_no = df2[df2["injSeverity"]== 2];# dataset for no incapacity injury
df_incapacity = df2[df2["injSeverity"]== 3];#dataset for incapacity injury
df_killed = df2[df2["injSeverity"]== 4];#dataset for killed injury
# Apply Oneway ANOVA
#hypothesis:
#H0:There is no age mean difference.
#H1: There is age mean differnce between injury severity groups.
print(f_oneway(df_none["ageOFocc"],␣
↪df_possible["ageOFocc"],df_no["ageOFocc"],df_incapacity["ageOFocc"],
               df_killed["ageOFocc"]));
```

```
F_onewayResult(statistic=78.26858783063506, pvalue=4.1325230342567886e-66)
```

The p-value is zero. Therefore, we have strong evidence against H0. There is sufficient statistical evidence to claim that the injury severity groups have different means. Therefore, it is not appropriate to exclude the variable experiment from the analysis.

4 Question2 d

[5]:
```
response_count = df2.groupby("dead")["dead"].count();
print(response_count);
print("Percentage of 0s:", 100*response_count[0]/np.sum(response_count));
print("Percentage of 1s:", 100*response_count[1]/np.sum(response_count));
# We use overampling to balance our data.
df_minority = df2[(df2['dead']==0)];
df_majority = df2[(df2['dead']==1)];
df_minority_upsampled = resample(df_minority,
                                 replace=True,     # sample with replacement
                                 n_samples= response_count[1], # to match␣
↪majority class
                                 random_state=123);  # reproducible results
df_minority_upsampled.reset_index(drop=True, inplace=True); # reseting row␣
↪numbers
df_upsampled = pd.concat([df_minority_upsampled, df_majority]);
response_count = df_upsampled.groupby("dead")["dead"].count();
```

```
print(response_count); # Check for data balancing again and make sure they are␣
 ↪equal.


#train the model and fit
X =␣
 ↪df_upsampled[["airbag","seatbelt","frontal","sex","ageOFocc","yearVeh","deploy"]]#␣
 ↪explannatory variables
y = df_upsampled[['dead']];# response variable

# Here we define training and testing sets.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
 ↪random_state=0);

data_train = pd.concat([X_train, y_train], axis = 1)#trained dataset

#model= sm.GLM.from_formula("dead ~ C(airbag) + C(seatbelt) + C(frontal) +␣
 ↪C(sex) + ageOFocc + yearVeh + C(deploy) ", family = sm.families.Binomial(),
                              #data=data_train);
#result= model.fit();
#print(result.summary());

#Since the'yearVeh' is not significant(P-value greater the 0.05), we remove it␣
 ↪from the model.

model = sm.GLM.from_formula("dead ~ C(airbag) + C(seatbelt) + C(frontal) +␣
 ↪C(sex) + ageOFocc + C(deploy)",
                            family=sm.families.Binomial(),
                            data=data_train)
result = model.fit();
print(result.summary()); # Now all the variables are significant with p-values␣
 ↪less than 0.05.

#Check Over_dispersion
dev = result.deviance; # Residual Deviance
dof = result.df_resid; # Degree of freedoms of Residuals
pvalue = 1 - scipy.stats.chi2.cdf(dev, dof); # p-value
# H0: Logistic regression model provides an adequate fit for the data
# H1: Logistic regression model does not provide an adequate fit for the data
if pvalue < 0.05:
    print("Saturated model -- p-value: ", pvalue);
else :
    print("Logistic model is ok -- p-value=", pvalue);

# Calculation of Pearson chi2 / n - (p+1)
print("Pearson2 / Df",result.pearson_chi2 / result.df_resid);
```

```python
# This value is close to 1
# We also fit a quasi-binomial model
result_quasi = model.fit(scale="X2");
print(result_quasi.summary());

# Predictions and model evaluation(Accuracy, sensetivity and specificity)
predictions = result.predict(X_test);
predictions_nominal = [ 0 if x < 0.5 else 1 for x in predictions];
from sklearn.metrics import confusion_matrix, classification_report
cm = confusion_matrix(y_test, predictions_nominal)
print("Confusion matrix:", cm);
# The diagonal elements of the confusion matrix indicate correct predictions,
# while the off-diagonals represent incorrect predictions
print("Accuracy: ", round(np.sum(np.diagonal(cm))/np.sum(cm),3));
print("Sensitivity: ", round(cm[1,1]/np.sum(cm[1,:]),3));
print("Specificity: ", round(cm[0,0]/np.sum(cm[0,:]),3));
# We can also get those values as follows
print(classification_report(y_test, predictions_nominal,digits = 3))
```

```
dead
0     1180
1    24883
Name: dead, dtype: int64
Percentage of 0s: 4.527491079307831
Percentage of 1s: 95.47250892069216
dead
0    24883
1    24883
Name: dead, dtype: int64
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                   dead   No. Observations:                34836
Model:                            GLM   Df Residuals:                    34829
Model Family:                Binomial   Df Model:                            6
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -20487.
Date:                Fri, 12 Apr 2024   Deviance:                       40973.
Time:                        11:02:32   Pearson chi2:                 3.48e+04
No. Iterations:                     4   Pseudo R-squ. (CS):             0.1895
Covariance Type:            nonrobust
================================================================================
====
                       coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
----
Intercept           -0.4583      0.039    -11.619      0.000      -0.536
-0.381
```

```
C(airbag)[T.1]       1.0322      0.036     28.521      0.000       0.961
1.103
C(seatbelt)[T.1]     1.4126      0.025     55.962      0.000       1.363
1.462
C(frontal)[T.1]      1.0829      0.026     41.036      0.000       1.031
1.135
C(sex)[T.1]         -0.2578      0.025    -10.479      0.000      -0.306
-0.210
C(deploy)[T.1]      -0.8494      0.039    -21.967      0.000      -0.925
-0.774
ageOFocc            -0.0261      0.001    -41.231      0.000      -0.027
-0.025
================================================================================
====
Saturated model -- p-value:  0.0
Pearson2 / Df 0.9990645482163427
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                   dead   No. Observations:                34836
Model:                            GLM   Df Residuals:                    34829
Model Family:                Binomial   Df Model:                            6
Link Function:                  Logit   Scale:                         0.99906
Method:                          IRLS   Log-Likelihood:                -20487.
Date:                Fri, 12 Apr 2024   Deviance:                       40973.
Time:                        11:02:32   Pearson chi2:                 3.48e+04
No. Iterations:                     6   Pseudo R-squ. (CS):             0.1895
Covariance Type:            nonrobust
================================================================================
====
                       coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
----
Intercept           -0.4583      0.039    -11.624      0.000      -0.536
-0.381
C(airbag)[T.1]       1.0322      0.036     28.534      0.000       0.961
1.103
C(seatbelt)[T.1]     1.4126      0.025     55.988      0.000       1.363
1.462
C(frontal)[T.1]      1.0829      0.026     41.055      0.000       1.031
1.135
C(sex)[T.1]         -0.2578      0.025    -10.484      0.000      -0.306
-0.210
C(deploy)[T.1]      -0.8494      0.039    -21.977      0.000      -0.925
-0.774
ageOFocc            -0.0261      0.001    -41.250      0.000      -0.027
-0.025
================================================================================
```

```
====
Confusion matrix: [[5152 2357]
 [2430 4991]]
Accuracy:  0.679
Sensitivity:  0.673
Specificity:  0.686
              precision    recall  f1-score   support

           0      0.680     0.686     0.683      7509
           1      0.679     0.673     0.676      7421

    accuracy                          0.679     14930
   macro avg      0.679     0.679     0.679     14930
weighted avg      0.679     0.679     0.679     14930
```

The scale parameter is 0.999 from the quasi-binomial model, which is very close to 1. Hence, the logistic regression model provides an adequate fit for the data, even though this hypothesis was rejected according to the chi-square test above.

The logistic regression correctly predicted the survival statuses 67.9% of the time. The model correctly predicted 67.3% of the time those who survived car accidents. The model correctly predicted 68.6 % of the time those who died of car accidents.

5 Question2 e

ageOFocc : For every unit increase in age(one year), we expect that the odds of surviving decrease by a factor of (exp(-0.0261))= 0.974,keeping other factors constant, which means that as people get older, the odds of survival decreases.

Seatbelt: The expected odds of survival for those who have their seatbelt fastened over the odds of survival for those who do not increase by a factor of exp(1.41)=4.1, which means that people with seatbelt on would help save lives.

6 Question2 f

[6]:
```
## Q2f prediction
pred_1 = {"airbag":[0], "seatbelt":[0], "frontal":[1],"sex":[0],"deploy":
 ↪[0],"ageOFocc":[70]};
pred_1 = pd.DataFrame(data=pred_1);
pred_prob1 = result.predict(pred_1);# probability of survial for senario 1
prob_not1 = 1-pred_prob1[0] # probability of death for senario 1
odds_of_not1 = prob_not1/(1-prob_not1) # odds of not survial(death)is␣
 ↪calculated by p(not)/(1-p(not))



pred_2 = {"airbag":[1], "seatbelt":[1], "frontal":[1],"sex":[0],"deploy":
 ↪[1],"ageOFocc":[70]};
```

```
pred_2 = pd.DataFrame(data=pred_2);
pred_prob2 = result.predict(pred_2);# probability of survial for senario 2
prob_not2 = 1-pred_prob2[0]# probability of death for senario 2
odds_of_not2 = prob_not2/(1-prob_not2)# odds of not survial(death)is calculated
 ↪by p(not)/(1-p(not))
print("The odds of not surving for scenario 1 is ", odds_of_not1)
print("The odds of not surving for scenario 2 is ",odds_of_not2 )
```

```
The odds of not surving for scenario 1 is  3.3208866098275056
The odds of not surving for scenario 2 is  0.6734880020488657
```

For the first scenario, where there is no airbag, the seatbelt is not fastened, the accident is frontal, and the person is 70 years old woman with the airbag not deployed, the odds of not surviving is 3.32, meaning that the person is 3.32 more likely to not survive with above conditions than to survive.

For the second scenario, where there is an airbag, the seatbelt is fastened, the accident is frontal, the person is 70 years old woman with the airbag being deployed, the odds of not surviving is 0.67, meaning that the person is 0.67 times more likely(less likely indeed) to not survive under those conditions than to survive.

Those predictions are indeed plausible as airbags and seatbelts play important roles in saving people's lives on the road in reality.