**T.C.**

**DOKUZ EYLÜL ÜNİVERSİTESİ**

**İKTİSADİ VE İDARİ BİLİMLER FAKÜLTESİ**

**YÖNETİM BİLİŞİM SİSTEMLERİ BÖLÜMÜ**

**PROGRAMLAMA 2**

**PROGRAMLAMA 2 FİNAL RAPORU**

***2019469089***

***Melisa UYAR***

**2020, İZMİR**

**Hangi işletme problemini ele alıyorum?**

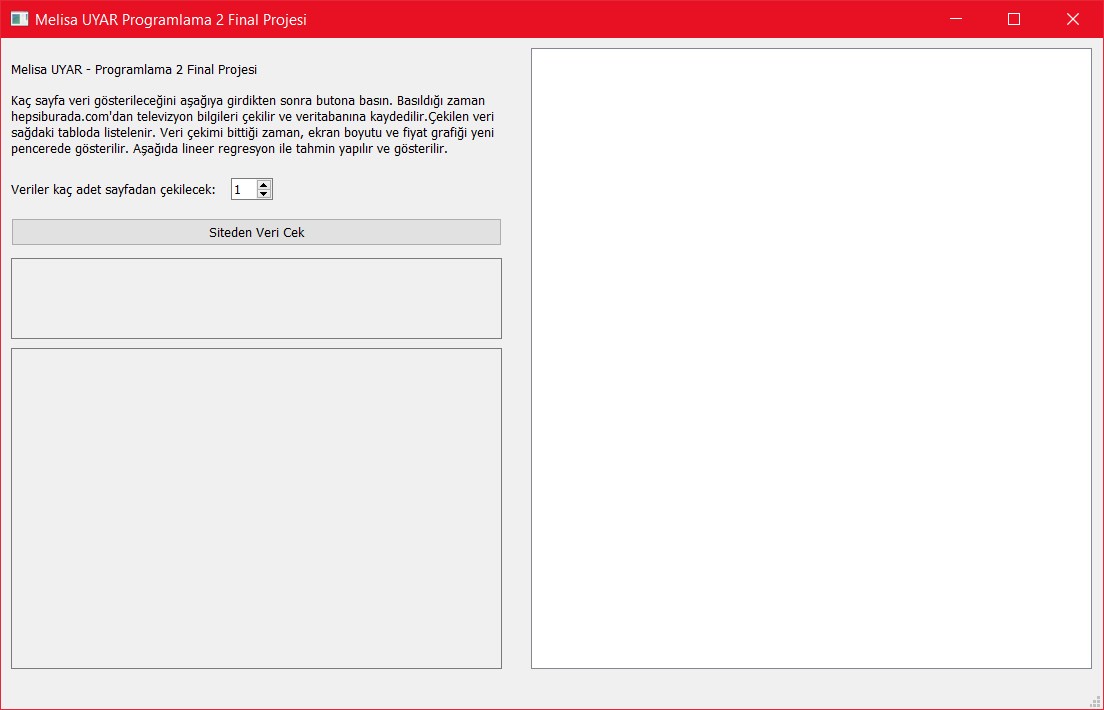
Bu projede İşletmelerin fiyatlandırma problemini ele aldım. İşletmelerin amaçlarına bakıldığında karlılık, hayatını devam ettirme ve sosyal sorumluluğunu yerine getirebilme gibi temel çalışma yaklaşımları görülebilir. İşletme bu amaçlarına ulaşabilmek için ürettiği mal ve hizmete bir değer biçmek durumundadır.

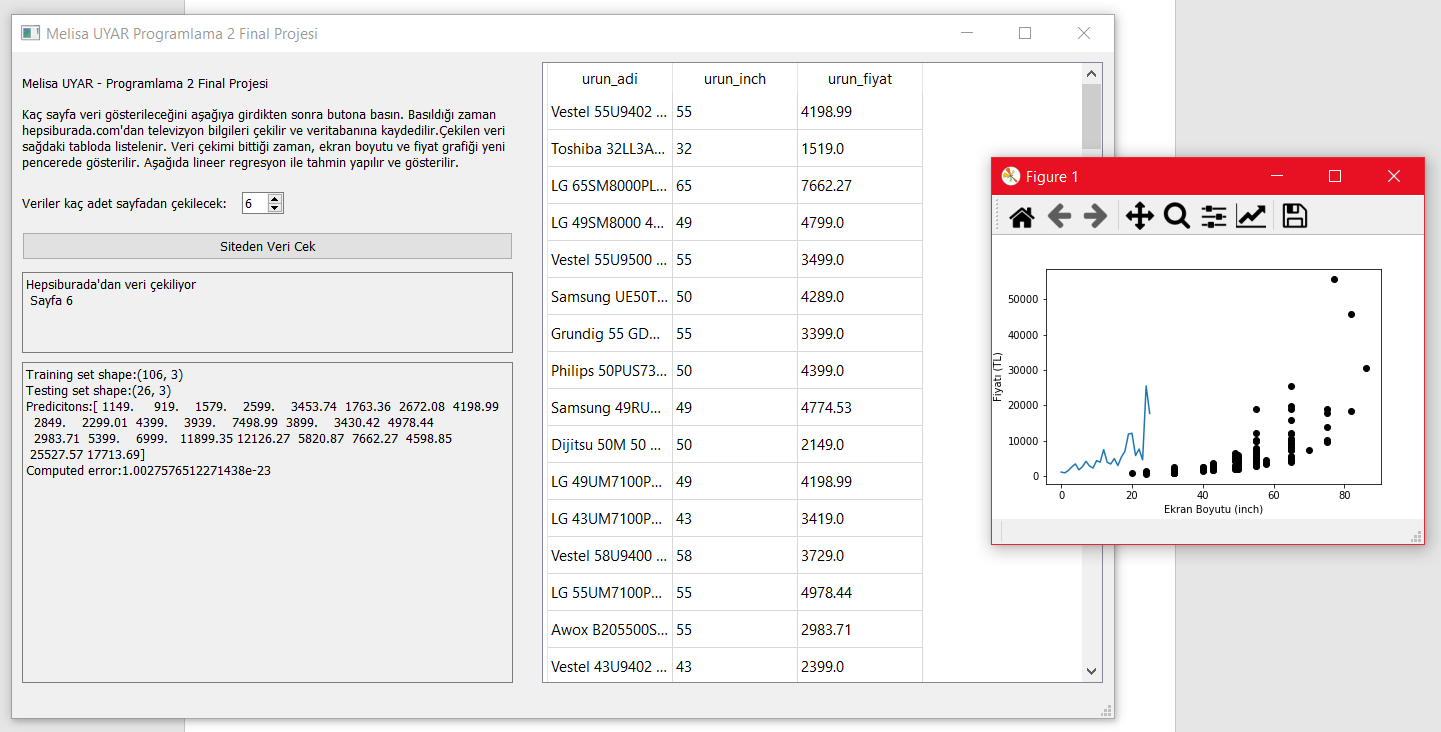
Fiyat belirleme tüketiciyi ya da hedef kitleyi o ürüne çekmede önemli rol oynar. Çünkü o ürünleri kullanacak müşteride fiyat bir algı yaratmaktadır. Alışverişe çıkacak insanların zihinlerinde önceden oluşmuş referans fiyatlar vardır. Bu referans fiyatlara yakın fiyatlara ılımlı bakıp, referans fiyatın çok altında veya üzerindeki fiyatlara ilgi göstermekten çekinirler. Böyle durumlarda işletmeler uygulayacakları fiyatları belirlerken pazardaki rekabeti, pazar talebini göz önünde bulundurmalıdır. İşletmelerin her alanı için önem taşıyan fiyat, kar ya da zararı belirleyen en güçlü etmendir.

**Amaç nedir?**

Burada amaç işletmenin fiyatlandırma problemini, ürünün internette satılan emsal fiyatlarına bakarak, ortalama fiyatlandırmasını yapıp bu problemi çözebilmek. Program, hepsiburada’dan televizyonlar hakkında verileri çekip, sitedeki bütün televizyon adı, ekran boyutu(inch) ve ürün fiyatlarını veritabanına kaydedip, masaüstü uygulama sayesinde çekilen verileri listeliyor. Listelenen verilerin kaç sayfadan çekileceğini kullanıcı girerek filtrelemesini yapabilir. Daha sonra çekilen verilerden tahmin grafiği oluşturulur. Bu grafik yeni pencerede kullanıcı önüne çıkarılır.

Ekran Görüntüleri:





**Kodları:**

# -\*- coding: utf-8 -\*-

#MELISA UYAR

import sys

import matplotlib

import matplotlib.pyplot as plt

matplotlib.use('Qt5Agg')

from PyQt5 import QtCore, QtGui, QtWidgets

from PyQt5.QtCore import Qt

from PyQt5.QtWidgets import QVBoxLayout

import bs4 as bs

import urllib.request

import mysql.connector

import pandas as pd

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

from matplotlib.backends.backend\_qt5agg import FigureCanvasQTAgg

from matplotlib.backends.backend\_qt5agg import FigureCanvasQTAgg, NavigationToolbar2QT as NavigationToolbar

from matplotlib.figure import Figure

from time import sleep

import requests

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(1102, 671)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.sitedenGelenVeriler = QtWidgets.QTableView(self.centralwidget)

self.sitedenGelenVeriler.setGeometry(QtCore.QRect(530, 10, 561, 621))

self.sitedenGelenVeriler.setObjectName("sitedenGelenVeriler")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(10, 40, 491, 91))

self.label.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.AlignVCenter)

self.label.setWordWrap(True)

self.label.setObjectName("label")

self.sitedenVeriCekmeButonu = QtWidgets.QPushButton(self.centralwidget)

self.sitedenVeriCekmeButonu.setGeometry(QtCore.QRect(10, 180, 491, 28))

self.sitedenVeriCekmeButonu.setObjectName("sitedenVeriCekmeButonu")

self.tahminLabel = QtWidgets.QLabel(self.centralwidget)

self.tahminLabel.setGeometry(QtCore.QRect(10, 310, 491, 321))

self.tahminLabel.setAutoFillBackground(False)

self.tahminLabel.setFrameShape(QtWidgets.QFrame.StyledPanel)

self.tahminLabel.setFrameShadow(QtWidgets.QFrame.Sunken)

self.tahminLabel.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.AlignTop)

self.tahminLabel.setWordWrap(True)

self.tahminLabel.setObjectName("tahminLabel")

self.spinBox = QtWidgets.QSpinBox(self.centralwidget)

self.spinBox.setGeometry(QtCore.QRect(230, 140, 42, 22))

self.spinBox.setMinimum(1)

self.spinBox.setObjectName("spinBox")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(10, 140, 211, 21))

self.label\_2.setObjectName("label\_2")

self.uyariLabel = QtWidgets.QLabel(self.centralwidget)

self.uyariLabel.setGeometry(QtCore.QRect(10, 220, 491, 81))

self.uyariLabel.setFrameShape(QtWidgets.QFrame.StyledPanel)

self.uyariLabel.setFrameShadow(QtWidgets.QFrame.Sunken)

self.uyariLabel.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignLeft|QtCore.Qt.AlignTop)

self.uyariLabel.setWordWrap(True)

self.uyariLabel.setObjectName("uyariLabel")

self.MelisaUYAR = QtWidgets.QLabel(self.centralwidget)

self.MelisaUYAR.setGeometry(QtCore.QRect(10, 10, 481, 41))

self.MelisaUYAR.setScaledContents(False)

self.MelisaUYAR.setObjectName("MelisaUYAR")

MainWindow.setCentralWidget(self.centralwidget)

self.statusbar = QtWidgets.QStatusBar(MainWindow)

self.statusbar.setObjectName("statusbar")

MainWindow.setStatusBar(self.statusbar)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "Melisa UYAR Programlama 2 Final Projesi"))

self.label.setText(\_translate("MainWindow", "Kaç sayfa veri gösterileceğini aşağıya girdikten sonra butona basın. Basıldığı zaman hepsiburada.com\'dan televizyon bilgileri çekilir ve veritabanına kaydedilir.Çekilen veri sağdaki tabloda listelenir. Veri çekimi bittiği zaman, ekran boyutu ve fiyat grafiği yeni pencerede gösterilir. Aşağıda lineer regresyon ile tahmin yapılır ve gösterilir."))

self.sitedenVeriCekmeButonu.setText(\_translate("MainWindow", "Siteden Veri Cek"))

self.tahminLabel.setText(\_translate("MainWindow", "TextLabel"))

self.label\_2.setText(\_translate("MainWindow", "Veriler kaç adet sayfadan çekilecek:"))

self.uyariLabel.setText(\_translate("MainWindow", "TextLabel"))

self.MelisaUYAR.setText(\_translate("MainWindow", "Melisa UYAR - Programlama 2 Final Projesi"))

def buttonlariBagla(self):

self.sitedenVeriCekmeButonu.clicked.connect(self.sitedenVeriCekmeDurumu)

def sitedenVeriCekmeDurumu(self):

self.SitedenVeriCek()

frame = self.KayitliVeriyiCek()

model = PandasModel(frame)

self.sitedenGelenVeriler.setModel(model)

print(frame)

frame = frame.sort\_values(by=['urun\_inch'], ascending=True)

print('sort sonrasi--------------------------------------')

print(frame)

self.grafigiGoster(frame)

def grafigiGoster(self, frame):

tahmin = self.linearTahmin(frame)

plt.scatter(frame["urun\_inch"], frame["urun\_fiyat"], color='Black')

plt.plot(tahmin)

plt.xlabel('Ekran Boyutu (inch)')

plt.ylabel('Fiyatı (TL)')

plt.show()

def linearTahmin(self,frame):

sutunlar = frame.columns.tolist()

sutunlar = [c for c in sutunlar if c not in ["urun\_adi"]]

hedef = "urun\_fiyat"

egit = frame.sample(frac = 0.8, random\_state = 0)

print('---egit---')

print(egit)

test = frame.loc[~frame.index.isin(egit.index)]

cikti = ""

cikti+=('Training set shape:' + str(egit.shape)) + '\n'

cikti+=('Testing set shape:' + str(test.shape)) + '\n'

lin\_model = LinearRegression()

lin\_model.fit(egit[sutunlar], egit[hedef])

lin\_tahmin = lin\_model.predict(test[sutunlar])

cikti+=('Predicitons:' + str(lin\_tahmin)) + '\n'

lin\_mse = mean\_squared\_error(lin\_tahmin, test[hedef])

cikti+=('Computed error:' + str(lin\_mse)) + '\n'

print('---cikti---')

print(cikti)

self.tahminLabel.setText(cikti)

return lin\_tahmin

def SitedenVeriCek(self):

self.uyariLabel.setText("Hepsiburada'dan veri çekiliyor")

sleep(0)

try:

baglanti=mysql.connector.connect(user='root',password='12345678',host='127.0.0.1',database='hepsiburada')

isaretci=baglanti.cursor()

sitelinki = "https://www.hepsiburada.com/led-tv-televizyonlar-c-163192?sayfa="

sayfano = 0

print('Veritabanini siliyoruz')

isaretci.execute('''DELETE FROM televizyonlar''')

sayfalimiti = self.spinBox.value()

sleep(0)

while(sayfano < sayfalimiti):

sayfano+=1

self.uyariLabel.setText("Hepsiburada'dan veri çekiliyor \n Sayfa " + str(sayfano))

sleep(0)

tamlink = sitelinki+ str(sayfano)

#r = requests.get(tamlink)

kaynak=urllib.request.urlopen(tamlink).read()

sayfa=bs.BeautifulSoup(kaynak,'lxml')

veriCekilmeDurumu = False

sayac = 0

#for url in sayfa.findAll('div',{'class','contain-lg-3 contain-md-3 contain-sm-3 fluid with-bottom-border'}):

for url in sayfa.findAll('div',{'class','product-detail'}):

#print(url)

#u=url.find('div',{'class':'box product'})

#u=url.findAll('span',{'class':'hb-pl-cn'})[0].string.strip()

print('----------------------------------------------------------')

print(sayac)

sayac+=1

print('----------------------------------------------------------')

try:

u=url.findAll('p')[0].findAll('span',recursive=False)[0].contents[0]

except:

continue

#u=url.strip

print(u)

cifttirnak=u.find('"')

if cifttirnak==-1:

cifttirnak=u.find('\'\'')

if cifttirnak!=-1:

if cifttirnak>2:

inch=u[cifttirnak-3]+u[cifttirnak-2]+u[cifttirnak-1]

elif cifttirnak>1:

inch=u[cifttirnak-2]+u[cifttirnak-1]

else:

continue

inch=''.join(filter(str.isdigit,inch))

f=url.findAll('div',{'class':'price-value'})

if len(f)>0:

fiyattext=f[0].text

fiyattext=fiyattext.replace(' TL','')

fiyattext=fiyattext.replace('.','')

fiyattext=fiyattext.replace(',','.')

fiyat=float(fiyattext)

#print(fiyat)

else:

continue

#print(inch + '\"')

u=u.replace('"','')

u=u.replace("'","")

isaretci.execute('''INSERT INTO televizyonlar (urun\_adi,urun\_inch,urun\_fiyat) VALUES ("%s","%s","%f")'''%(u,inch,fiyat))

baglanti.commit()

#print(u)

veriCekilmeDurumu = True

sleep(.25)

except Exception as e:

veriCekilmeDurumu = False

print('----Veri Cekme durdu. Cekilen sayfa sayisi %d----' %sayfano)

print(e)

finally:

if(baglanti.is\_connected()):

isaretci.close()

baglanti.close()

return veriCekilmeDurumu

def KayitliVeriyiCek(self):

baglanti=mysql.connector.connect(user='root',password='12345678',host='127.0.0.1',database='hepsiburada')

isaretci=baglanti.cursor()

frame = pd.read\_sql("SELECT \* FROM televizyonlar", baglanti);

pd.set\_option('display.expand\_frame\_repr', False)

isaretci.close()

baglanti.close()

return frame

class MplCanvas(FigureCanvasQTAgg):

def \_\_init\_\_(self, parent=None, width = 5, height=4, dpi = 100):

fig = Figure(figsize=(width, height),dpi=dpi)

self.axes = fig.add\_subplot(111)

super(MplCanvas, self).\_\_init\_\_(fig)

class PandasModel(QtCore.QAbstractTableModel):

"""

Class to populate a table view with a pandas dataframe

"""

def \_\_init\_\_(self, data, parent=None):

QtCore.QAbstractTableModel.\_\_init\_\_(self, parent)

self.\_data = data

def rowCount(self, parent=None):

return len(self.\_data.values)

def columnCount(self, parent=None):

return self.\_data.columns.size

def data(self, index, role=QtCore.Qt.DisplayRole):

if index.isValid():

if role == QtCore.Qt.DisplayRole:

return str(self.\_data.values[index.row()][index.column()])

return None

def headerData(self, col, orientation, role):

if orientation == QtCore.Qt.Horizontal and role == QtCore.Qt.DisplayRole:

return self.\_data.columns[col]

return None

if \_\_name\_\_ == "\_\_main\_\_":

app = QtWidgets.QApplication(sys.argv)

MainWindow = QtWidgets.QMainWindow()

ui = Ui\_MainWindow()

ui.setupUi(MainWindow)

ui.buttonlariBagla()

MainWindow.show()

ui.uyariLabel.setText("")

ui.tahminLabel.setText("")

sys.exit(app.exec\_())