Mart / 2022

Veri Bilimi Yüksek Gelişim Programı Final Ödevi

Tuncay YAYLALI/Çevre Y. Mühendisi

Tuncay YAYLALI

Kasım 1970

Almanya' da doğdu

1977-1987

İlk, orta ve lise öğrenimini Manisa' da tamamladı

1988-1993

İ.T.Ü. Çevre Mühendisliği' nden mezun oldu

1995-1997

Topçu ve Füze Okulu' nu dönem 6 ncısı olarak tamamladıktan sonra Aksaz Deniz Üs Komutanlığı' nda askerlik görevini tamamladı

1998-2004

Çevre Bakanlığı' nda Çevre Mühendisi olarak göreve başladı

2004-...

Çevre sektöründe mühendislik ve müşavirlik hizmeti veriyor

2017-2021

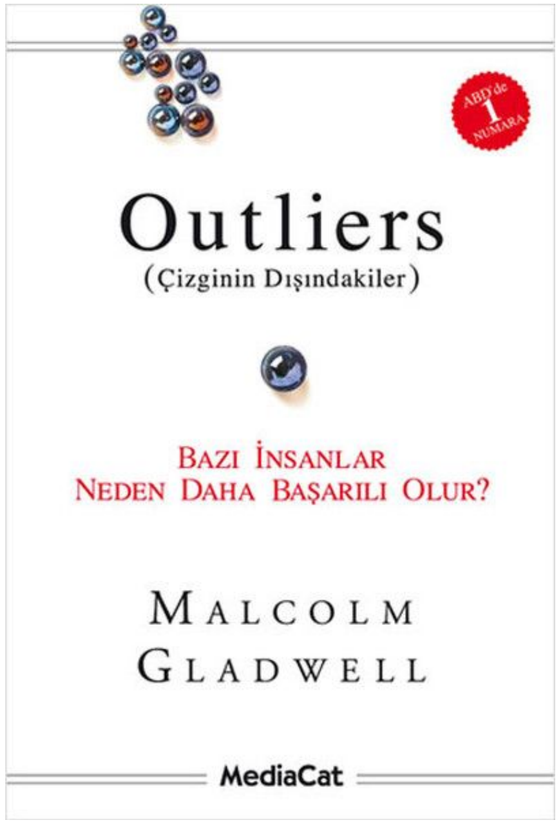
Anadolu Üniversitesi (İşletme), E.S.T.Ü. (Çevre Yönetimi Y.L.) ve Marmara Üniversitesi (İş Güvenliği Y.L.)

2021-...

Veri Bilimi Yüksek Gelişim Programı' na devam ediyor

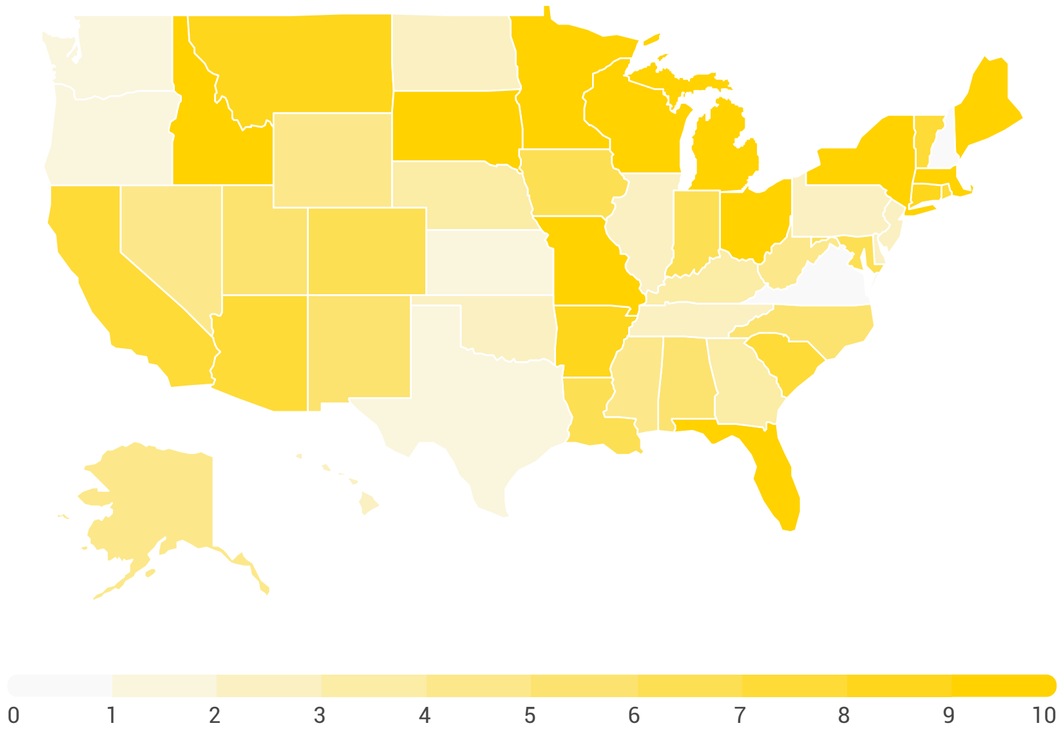
Outliers, GLADWELL, M.

Malcolm GLADWELL' in Outliers adlı kitabı, elde ettikleri başarılar ve hayat hikayetleriyle diğer insanlardan farklılaşan kişiler üzerine yapılmış bir çalışmadır. Çalışma kapsamında başarılı kişilerin, kişisel özelliklerinin yanı sıra ekonomik, demografik, sosyolojik, kültürel, fiziksel vb. dış çevre özeliklerinin de dikkate alınması gerektiği vurgulanarak daha geniş bir perspektiften konu ele alınmıştır



Veri Seti

University of California Center of Machine Learning and Intelligent Systems' in internet sitesinde yer alan "Adult" adlı veri seti kullanılmıştır.



Data Set Characteristics:	Multivariate
Attribute Characteristics:	Categorical, Integer
Associated Tasks:	Classification
Number of Instances:	48842
Number of Attributes:	14
Missing Values?	Yes
Area:	Social
Date Donated	1996-05-01
Number of Web Hits:	2398348

[UCI Machine Learning Repository: Adult Data Set]

Veri seti, 1994 Nüfus Sayım Sonuçları veri tabanından Barry Becker tarafından yapılmıştır. Bazı koşullar kullanılarak bir dizi kayıt veri setindençıkarılmıştır.

Veri Seti Açıklaması

- **age**: Yaş
- **workclass**: Çalışma Şekli (Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked)
- **fnlwgt**: Benzer sosyo-ekonomik özelliklere sahip her bir personanın toplam eleman sayısıdır.
- **education**: Eğitim Durumu (Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool)
- **education Num**: Eğitim Süresi
- **marital_status**: Medeni Durumu (Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse)
- **occupation**: Meslek (Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces)
- **relationship**: İlişki Durumu (Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried)
- **race**: Irk (White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black)
- **sex**: Cinsiyet (Female, Male)
- **capital_gain**: Sermaye Birikimi
- **capital_loss**: Borç Durumu
- **hours_per_week**: Haftalık Çalışma Süresi
- **native-country**: Oirijin Ülke (United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands)
- **income**: Gelir (>50K veya <=50K)

Keşifçi Veri Analizi

```python
# Veri Setine Genel Bakış


def check_df(dataframe, head=5):
    print(f"\nVERİ SETİNİN BOYUTU")
    print(dataframe.shape)
    print(f"\nDEĞİŞKEN TİPLERİ")
    print(dataframe.dtypes)
    print(f"\nİLK 5 DEĞER")
    print(dataframe.head(head))
    print(f"\nSON 5 DEĞER")
    print(dataframe.tail(head))
    print(f"\nEKSİK DEĞERLER")
    print(dataframe.isnull().sum())
    print(f"\nBETİMLEYİCİ İSTATİSTİK")
    print(dataframe.quantile([0, 0.05, 0.50, 0.95, 0.99, 1]).T)


check_df(df)
```

DEĞİŞKEN	TİP
age	int64
workclass	object
fnlwgt	int64
education	object
education_num	int64
marital_status	object
occupation	object
relationship	object
race	object
sex	object
capital_gain	int64
capital_loss	int64
hours_per_week	int64
native_country	object
income	object
first_situation	object

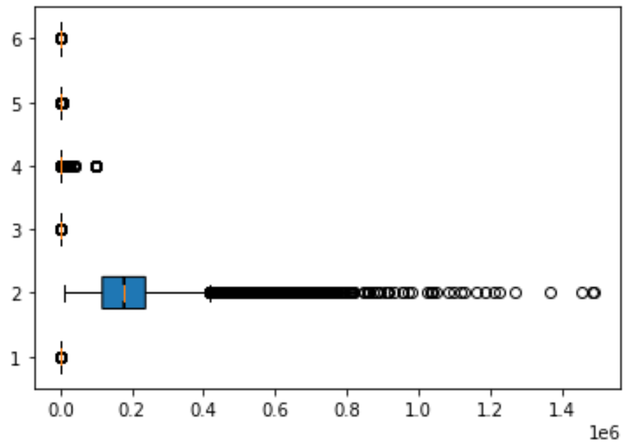
DEĞER	DEĞİŞKEN
48842	16

İLK 5 DEĞER	age	workclass	fnlwgt	...	native_country	income	first_situation
0	39	State-gov	77516	...	United-States	<=50K	train
1	50	Self-emp-not-inc	83311	...	United-States	<=50K	train
2	38	Private	215646	...	United-States	<=50K	train
3	53	Private	234721	...	United-States	<=50K	train
4	28	Private	338409	...	Cuba	<=50K	train

Keşifçi Veri Analizi

DEĞİŞKENLER	EKSİK DEĞERLER
age	0
workclass	0
fnlwgt	0
education	0
education_num	0
marital_status	0
occupation	0
relationship	0
race	0
sex	0
capital_gain	0
capital_loss	0
hours_per_week	0
native_country	0
income	0
first_situation	0

BETİMLEYİCİ İSTATİSTİK						
	0.00	0.05	0.50	0.95	0.99	1.00
age	17.0	19.00	37.0	63.00	74.00	90.0
fnlwgt	12285.0	39615.40	178144.5	379481.65	509484.42	1490400.0
education_num	1.0	5.00	10.0	14.00	16.00	16.0
capital_gain	0.0	0.00	0.0	5013.00	15024.00	99999.0
capital_loss	0.0	0.00	0.0	0.00	2001.00	4356.0
hours_per_week	1.0	17.05	40.0	60.00	80.00	99.0



Keşifçi Veri Analizi

education	frequency	Ratio
HS-grad	15784	32.316449
Some-college	10878	22.271815
Bachelors	8025	16.430531
Masters	2657	5.439990
Assoc-voc	2061	4.219729
11th	1812	3.709922
Assoc-acdm	1601	3.277917
10th	1389	2.843864
7th-8th	955	1.955284
Prof-school	834	1.707547
9th	756	1.547848
12th	657	1.345154
Doctorate	594	1.216166
5th-6th	509	1.042136
1st-4th	247	0.505712
Preschool	83	0.169936

education_num	frequency	Ratio
9	15784	32.316449
10	10878	22.271815
13	8025	16.430531
14	2657	5.439990
11	2061	4.219729
7	1812	3.709922
12	1601	3.277917
6	1389	2.843864
4	955	1.955284
15	834	1.707547
5	756	1.547848
8	657	1.345154
16	594	1.216166
3	509	1.042136
2	247	0.505712
1	83	0.169936

occupation	frequency	Ratio
Prof-specialty	6172	12.636665
Craft-repair	6112	12.513820
Exec-managerial	6086	12.460587
Adm-clerical	5611	11.488064
Sales	5504	11.268990
Other-service	4923	10.079440
Machine-op-inspct	3022	6.187298
?	2809	5.751198
Transport-moving	2355	4.821670
Handlers-cleaners	2072	4.242251
Farming-fishing	1490	3.050653
Tech-support	1446	2.960567
Protective-serv	983	2.012612
Priv-house-serv	242	0.495475
Armed-Forces	15	0.030711

Keşifçi Veri Analizi

workclass	frequency	Ratio
Private	33906	69.419762
Self-emp-not-inc	3862	7.907129
Local-gov	3136	6.420703
?	2799	5.730724
State-gov	1981	4.055935
Self-emp-inc	1695	3.470374
Federal-gov	1432	2.931903
Without-pay	21	0.042996
Never-worked	10	0.020474

marital_status	frequency	Ratio
Married-civ-spouse	22379	2.759920
Never-married	16117	2.737398
Divorced	6633	2.733303
Separated	1530	2.721019
Widowed	1518	2.712829
Married-spouse-absent	628	0.012285
Married-AF-spouse	37	0.010237

fnlwgt	frequency	Ratio
203488	21	0.042996
120277	19	0.038901
190290	19	0.038901
125892	18	0.036854
126569	18	0.036854
286983	1	0.002047
185942	1	0.002047
234220	1	0.002047
214706	1	0.002047
350977	1	0.002047

relationship	frequency	Ratio
Husband	19716	40.366897
Not-in-family	12583	25.762663
Own-child	7581	15.521477
Unmarried	5125	10.493018
Wife	2331	4.772532
Other-relative	1506	3.083412
Married-AF-spouse	37	0.010237

age	frequency	Ratio
36	1348	2.759920
35	1337	2.737398
33	1335	2.733303
23	1329	2.721019
31	1325	2.712829
88	6	0.012285
85	5	0.010237
87	3	0.006142
89	2	0.004095
86	1	0.002047

race	frequency	Ratio
White	41762	85.504279
Black	4685	9.592154
Asian-Pac-Islander	1519	3.110028
Amer-Indian-Eskimo	470	0.962287
Other	406	0.831252

sex	frequency	Ratio
male	32650	66.848204
female	16192	33.151796

Keşifçi Veri Analizi

capital_gain	freuency	Ratio
0	44807	91.738668
15024	513	1.050326
7688	410	0.839441
7298	364	0.745260
99999	244	0.499570
...	...	...
22040	1	0.002047
2387	1	0.002047
1639	1	0.002047
1111	1	0.002047
6612	1	0.002047

capital_loss	frequency	Ratio
0	46560	95.327792
1902	304	0.622415
1977	253	0.517997
1887	233	0.477048
2415	72	0.147414
...	...	...
1539	1	0.002047
1870	1	0.002047
1911	1	0.002047
2465	1	0.002047
1421	1	0.002047

hours_per_week	frequency	Ratio
40	22803	46.687277
50	4246	8.693338
45	2717	5.562835
60	2177	4.457229
35	1937	3.965849
..	...	...
79	1	0.002047
94	1	0.002047
82	1	0.002047
87	1	0.002047
69	1	0.002047

native_country	frequency	Ratio
United-States	43832	89.742435
Mexico	951	1.947095
?	857	1.754637
Philippines	295	0.603988
Germany	206	0.421768
Puerto-Rico	184	0.376725
Canada	182	0.372630
El-Salvador	155	0.317350
India	151	0.309160
Cuba	138	0.282544
England	127	0.260022

income	frequency	Ratio
<=50K	24720	50.612178
<=50K.	12435	25.459645
>50K	7841	16.053806
>50K.	3846	7.874370

Özellik Mühendisliği

```
# age Değişkeni Üzerinden Gruplandırma Yapılarak Yeni Değişken Üretilmesi
df.loc[(df['age'] < 18), 'new_age_cat'] = 'young'
df.loc[((df['age'] >= 18) & (df['age'] < 56)), 'new_age_cat'] = 'mature'
df.loc[(df['age'] >= 56), 'new_age_cat'] = 'senior'
```

```
# relationship Değişkeni Üzerinden Gruplandırma Yapılarak Yeni Değişken Üretilmesi
df["new_is_alone"] = df["relationship"].apply(lambda x: "Yes" if x.strip() == "Not-in-family" else "No")
```

```
# native_country Değişkeni Üzerinden Gruplandırma Yapılarak Yeni Değişken Üretilmesi
north_america_countries = ["Jamaica", "Mexico", "Cuba", "Canada", "United-States", "Puerto-Rico", "Haiti", "Dominican-Republic", \
                           "El-Salvador", "Nicaragua", "Guatemala", "Honduras", "Trinadad&Tobago", "Outlying-US(Guam-USVI-etc)"]
south_america_countries = ["Ecuador", "Colombia", "Peru"]
europe_countries = ["England", "France", "Germany", "Greece", "Holand-Netherlands", "Hungary", "Ireland", "Italy", "Poland", "Portugal", "Scotland", "Yugoslavia"]
asia_countries = ["Iran", "Japan", "India", "Philippines", "Taiwan", "Thailand", "Vietnam", "China", "South", "Laos", "Cambodia", "Hong"]

for i in north_america_countries:
  df.loc[df["native_country"].str.contains(i), "new_continent"] = "North America"

for i in south_america_countries:
  df.loc[df["native_country"].str.contains(i), "new_continent"] = "South America"

for i in europe_countries:
  df.loc[df["native_country"].str.contains(i), "new_continent"] = "Europe"

for i in asia_countries:
  df.loc[df["native_country"].str.contains(i), "new_continent"] = "Asia"

df["new_continent"].fillna(value="Unknown", axis=0, inplace=True)
```

Özellik Mühendisliği

```
[ ]
# workclass Değişkeni Üzerinden Gruplandırma Yapılarak Yeni Değişken Üretilmesi
df.loc[df["workclass"].str.contains("Private"), "new_work_class"] = "Serbest"
df.loc[df["workclass"].str.contains("Self-emp-not-inc"), "new_work_class"] = "Serbest"
df.loc[df["workclass"].str.contains("Self-emp-inc"), "new_work_class"] = "Serbest"
df.loc[df["workclass"].str.contains("Local-gov"), "new_work_class"] = "Kamu"
df.loc[df["workclass"].str.contains("State-gov"), "new_work_class"] = "Kamu"
df.loc[df["workclass"].str.contains("Federal-gov"), "new_work_class"] = "Kamu"
df.loc[df["workclass"].str.contains("Without-pay"), "new_work_class"] = "Ucretsiz"
df.loc[df["workclass"].str.contains("Never-worked"), "new_work_class"] = "Ucretsiz"
df["new_work_class"].fillna(value="Unknown", axis=0, inplace=True)
```

```
[ ]
# Değişkenlerde Yer Alan Soru İşareti Değerlerinin Silinmesi
for col in ["workclass", "occupation", "native_country"]:
    df[col] = df[col].str.strip().map(lambda x: np.nan if x=="?" else x)
```

```
[ ]
# education Değişkeninin Silinmesi
df.drop("education", axis=1, inplace=True)
```

```
[ ]
# income Değişkeninin Standartlaştırılması
df["income"] = df["income"].apply(lambda x: 1 if ">50K" in x else 0)
```

Değişken Analizi

Observations: 48842

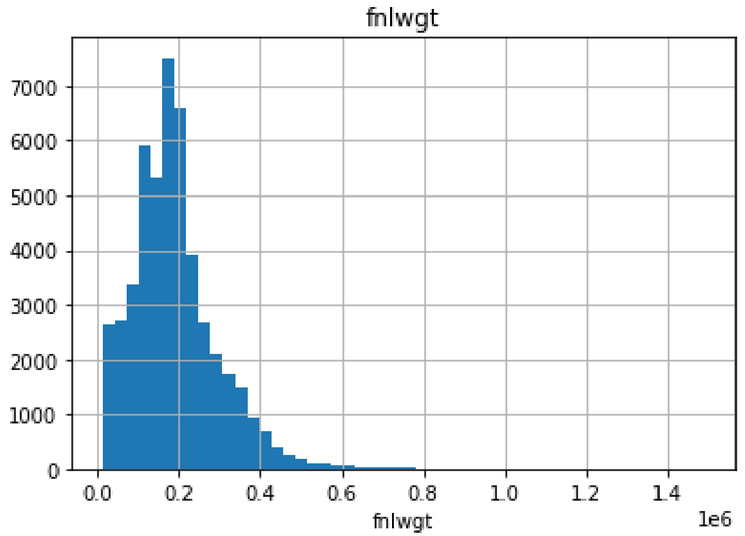
Variables: 19

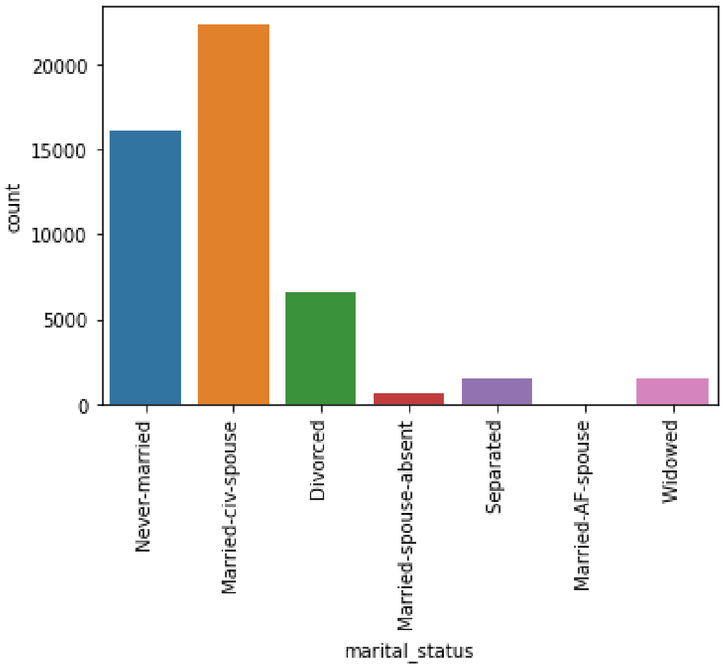
cat_cols: 13 ['workclass', 'marital_status', 'occupation', 'relationship', 'race', 'sex', 'native_country', 'first_situation', 'new_age_cat', 'new_is_alone', 'new_continent', 'new_work_class', 'income']

num_cols: 6 ['age', 'fnlwgt', 'education_num', 'capital_gain', 'capital_loss', 'hours_per_week']

cat_but_car: 0 []

num_but_cat: 1 ['income']





Hedef Değişken Analizi

workclass değişkeninde Never-worked sınıfındaki personanın, benzer şekilde new_age_cat değişkeninde young sınıfındaki personanın hedef değişkene hiç bir katkısının olmadığı görülmektedir. Bu sebeple new_age_cat değişkenindeki young sınıfı ve workclass değişkenindeki never_worked sınıfına ait değerler veri setinden silinmiştir.

Geliri 50K' nın üzerinde olanların oranı şirket sahibi serbest meslek erbabında, doktora yapmış olanlarda, eşi sivil evlilerde, yönetici pozisyonunda olanlarda, sarı ırk mensuplarında, erkeklerde, 56 yaş üzerinde, yalnız yaşamayanlarda, Avrupa' lılarda, kamuda çalışanlarda ve Fransız kökenlilerde daha yüksektir.

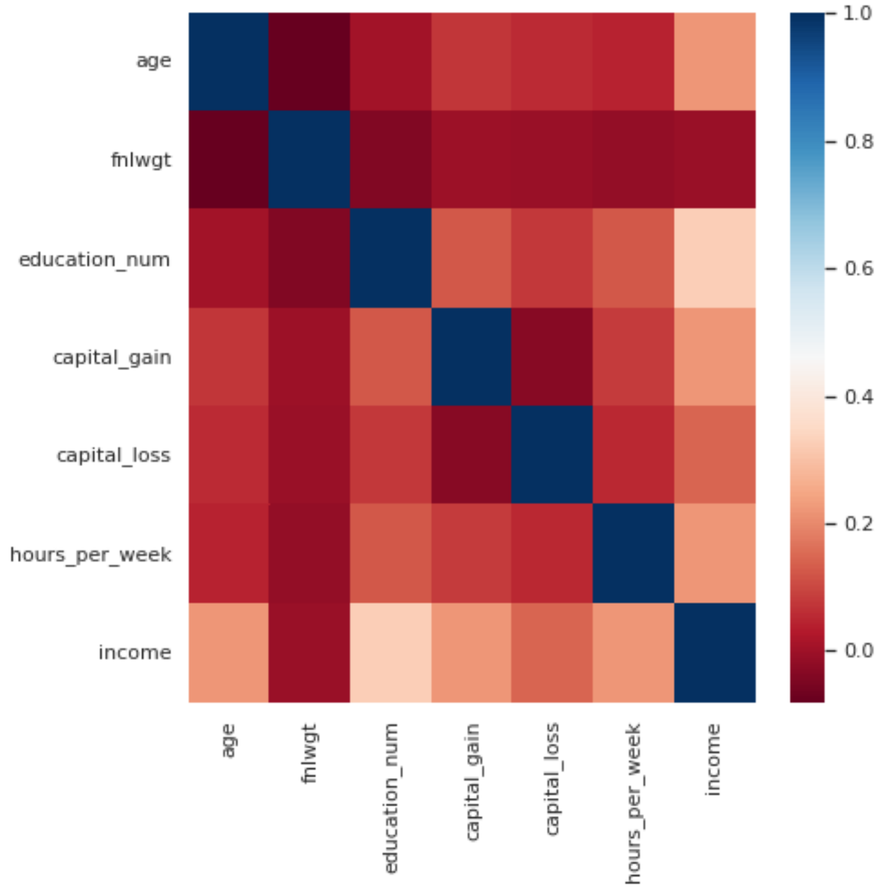
Geliri 50K' nın üzerinde olanların frekansı özel sektör çalışanlarında, lisans mezunlarında, eşi sivil evlilerde, yönetici pozisyonunda olanlarda, beyaz ırk mensuplarında, erkeklerde, 18 yaş üzeri yetişkinlerde, yalnız yaşamayanlarda, Kuzey Amerika' lılarda, serbest çalışanlarda ve Amerika Birleşik Devletleri kökenlilerde daha yüksektir.

workclass	TARGET_MEAN	TARGET_FREQUENCY
Federal-gov	0.391760	561
Local-gov	0.295599	927
Never-worked	0.000000	0
Private	0.217867	7387
Self-emp-inc	0.553392	938
Self-emp-not-inc	0.278871	1077
State-gov	0.267542	530
Without-pay	0.095238	2

new_age_cat	TARGET_MEAN	TARGET_FREQUENCY
mature	0.236529	9934
senior	0.280570	1753
young	0.000000	0

Korelasyon Analizi

Korelasyon analizi neticesinde sayısal değişkenler arasında pozitif veya negatif korelasyona sahip herhangi bir değişken tespit edilmemiştir.



Aykırı Değer Analizi

fnlwgt Amerika Birleşik Devletleri' nde benzer sosyo-ekonomik özelliklere sahip her bir personanın toplam eleman sayısını vermekte olup veri setinden daha büyük bir anakitlenin özelliklerini barındırdığı için aykırı değer baskılama işlemi tercih edilmemiştir. capital_gain ve capital_loss değişkenindeki aykırı değerlerin baskılanması tercih edilmiştir.

```python
# Aykırı Değer Analizi


def outlier_thresholds(dataframe, variable, low_quantile=0.01, up_quantile=0.99):
    quantile_one = dataframe[variable].quantile(low_quantile)
    quantile_three = dataframe[variable].quantile(up_quantile)
    interquantile_range = quantile_three - quantile_one
    up_limit = quantile_three + 1.5 * interquantile_range
    low_limit = quantile_one - 1.5 * interquantile_range
    return low_limit, up_limit


def check_outlier(dataframe, col_name):
    low_limit, up_limit = outlier_thresholds(dataframe, col_name)
    if dataframe[(dataframe[col_name] > up_limit) | (dataframe[col_name] < low_limit)].any(axis=None):
        return True
    else:
        return False


for col in num_cols:
    if col != "income":
      print(col, check_outlier(df, col))
```

```
age False
fnlwgt True
education_num False
capital_gain True
capital_loss False
hours_per_week False
```

```python
# Aykırı Değerlerin Baskılanması


def replace_with_thresholds(dataframe, variable):
    low_limit, up_limit = outlier_thresholds(dataframe, variable)
    dataframe.loc[(dataframe[variable] < low_limit), variable] = low_limit
    dataframe.loc[(dataframe[variable] > up_limit), variable] = up_limit


replace_with_thresholds(df, "capital_gain")
replace_with_thresholds(df, "capital_loss")

for col in num_cols:
    if col != "income":
      print(col, check_outlier(df, col))
```

```
age False
fnlwgt True
education_num False
capital_gain False
capital_loss False
hours_per_week False
```

Eksik Değer Analizi

Eksik değerlerin hedef değişkeni ile analizi neticesinde workclass ve occupation değişkeninde eksik değere sahip olan personaların gelirinin 50K' dan büyük olma olasılığı % 6 civarında olduğu, benzer şekilde native_country değişkeninde eksik değere sahip olan personaların gelirinin 50K' dan büyük olma olasılığının ise % 2 civarında olduğu tespit edilmiştir. Bu sebeple söz konusu değişkenlerde eksik değere sahip olup olmama üzerinden yeni değişkenler türetilmiştir.

```python
# Eksik Değer Analizi


def missing_values_table(dataframe, na_name=False):
    na_columns = [col for col in dataframe.columns if dataframe[col].isnull().sum() > 0]
    n_miss = dataframe[na_columns].isnull().sum().sort_values(ascending=False)
    ratio = (dataframe[na_columns].isnull().sum() / dataframe.shape[0] * 100).sort_values(ascending=False)
    missing_df = pd.concat([n_miss, np.round(ratio, 2)], axis=1, keys=['n_miss', 'ratio'])
    print(missing_df, end="\n")

    if na_name:
        return na_columns


missing_values_table(df, True)
```

```
               n_miss  ratio
workclass        2702   5.60
occupation       2702   5.60
native_country    854   1.77
['workclass', 'occupation', 'native_country']
```

Rare Analizi

Rare analizi neticesinde frekansı, oranı ve hedef değişken açısından ortalaması dikkate alınarak gruplandırılabilecek herhangi bir değer tespit edilmemiştir.

```python
# Rare Analizi


def rare_analyser(dataframe, target, cat_cols):
    for col in cat_cols:
        print(col, ":", len(dataframe[col].value_counts()))
        print(pd.DataFrame({"COUNT": dataframe[col].value_counts(),
                            "RATIO": dataframe[col].value_counts() / len(dataframe),
                            "TARGET_MEAN": dataframe.groupby(col)[target].mean()}), end="\n\n\n")


rare_analyser(df, "income", cat_cols)
```

```
workclass : 7
                 COUNT     RATIO  TARGET_MEAN
Federal-gov       1430  0.029645     0.392308
Local-gov         3115  0.064576     0.297592
Private          33451  0.693457     0.220830
Self-emp-inc      1687  0.034972     0.556017
Self-emp-not-inc  3853  0.079875     0.279522
State-gov         1979  0.041026     0.267812
Without-pay         21  0.000435     0.095238


marital_status : 7
                       COUNT     RATIO  TARGET_MEAN
 Divorced               6632  0.137485     0.101176
 Married-AF-spouse        37  0.000767     0.378378
 Married-civ-spouse    22376  0.463867     0.446192
 Married-spouse-absent   626  0.012977     0.092652
 Never-married         15520  0.321738     0.047229
 Separated              1530  0.031718     0.064706
 Widowed                1517  0.031448     0.084377
```

Encoding

binary değişkenler için label encoding kategorik değişkenler için label encoding işlemi gerçekleştirilmiştir.

```python
# Label Encoding


def label_encoder(dataframe, binary_col):
    labelencoder = LabelEncoder()
    dataframe[binary_col] = labelencoder.fit_transform(dataframe[binary_col])
    return dataframe


binary_cols = [col for col in df.columns if df[col].dtype not in [int, float] and df[col].nunique() == 2]

for col in binary_cols:
    df = label_encoder(df, col)
```

```python
# One Hot Encoding


def one_hot_encoder(dataframe, categorical_cols, drop_first=True):
    dataframe = pd.get_dummies(dataframe, columns=categorical_cols, drop_first=drop_first)
    return dataframe


df = one_hot_encoder(df, cat_cols)
```

Kullanışsız Değişkenler

Hedef değişken üzerinde etkisi % 0,1' in altında olan değişkenler veri setinden silinmiştir. ['workclass_Without-pay', 'marital_status_ Married-AF-spouse', 'occupation_Armed-Forces', 'native_country_Ecuador', 'native_country_France', 'native_country_Honduras', 'native_country_Hong', 'native_country_Hungary', 'native_country_Ireland', 'native_country_Laos', 'native_country_Outlying-US(Guam-USVI-etc)', 'native_country_Peru', 'native_country_Scotland', 'native_country_Thailand', 'native_country_Trinadad&Tobago', 'native_country_Yugoslavia', 'new_work_class_Ucretsiz']

```
# Kullanışsız Değişkenler
useless_cols = [col for col in df.columns if df[col].nunique() == 2 and (df[col].value_counts() / len(df) < 0.001).any(axis=None)]
print(useless_cols)
df.drop(useless_cols, axis=1, inplace=True)
```

Scaling ve Veri Setinin Ayrılması

```
# Standard Scaling
scaler = StandardScaler()
df[num_cols] = scaler.fit_transform(df[num_cols])
```

```
# Veri Setinin Train ve Test Olarak Bölünmesi
df_train = df.loc[df["first_situation_1"]==1]
df_test = df.loc[df["first_situation_1"]==0]
train_y = df_train["income_1"]
train_X = df_train.drop(["income_1"], axis=1)
test_y = df_test["income_1"]
test_X = df_test.drop(["income_1"], axis=1)
```

Model

```python
# Model


classifiers = [('LR', LogisticRegression()),
               ('KNN', KNeighborsClassifier()),
               ("SVC", SVC()),
               ("CART", DecisionTreeClassifier()),
               ("RF", RandomForestClassifier()),
               ('GBM', GradientBoostingClassifier()),
               ('XGBoost', XGBClassifier(use_label_encoder=False, eval_metric='logloss')),
               ('LightGBM', LGBMClassifier()),
               ('CatBoost', CatBoostClassifier(verbose=False))
               ]

for name, classifier in classifiers:
    cv_results = cross_validate(classifier, train_X, train_y, cv=3, scoring=["roc_auc","accuracy"])
    print("roc_auc : ", f" {round(cv_results['test_roc_auc'].mean(), 4)} ({name})")
    print("acc     : ", f" {round(cv_results['test_accuracy'].mean(), 4)} ({name})")
    print("***********")
```

```
roc_auc :   0.9067 (LR)
acc     :   0.8514 (LR)
***********
roc_auc :   0.8648 (KNN)
acc     :   0.8375 (KNN)
***********
roc_auc :   0.9009 (SVC)
acc     :   0.8552 (SVC)
***********
roc_auc :   0.7473 (CART)
acc     :   0.81 (CART)
***********
roc_auc :   0.9045 (RF)
acc     :   0.8548 (RF)
***********
roc_auc :   0.9207 (GBM)
acc     :   0.8648 (GBM)
***********
roc_auc :   0.9191 (XGBoost)
acc     :   0.8626 (XGBoost)
***********
roc_auc :   0.9256 (LightGBM)
acc     :   0.869 (LightGBM)
***********
roc_auc :   0.9272 (CatBoost)
acc     :   0.8692 (CatBoost)
***********
```

Hiperparametre Optimizasyonu

```
[ ]  # Hiperparametre Optimizasyonu
     catboost_model = CatBoostClassifier(verbose=False)
     catboost_params = {"iterations": [200, 500],
                        "learning_rate": [0.01, 0.1],
                        "depth": [3, 6]}
     catboost_best_grid = GridSearchCV(catboost_model, catboost_params, cv=5, n_jobs=-1, verbose=True).fit(train_X, train_y)
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
```

```
[ ]  # Hiperparametre Optimizasyonu Sonucu Train Veri Setinde Model Başarısı
     catboost_final = catboost_model.set_params(**catboost_best_grid.best_params_).fit(train_X, train_y)
     cv_results = cross_validate(catboost_final, train_X, train_y, cv=5, scoring=["accuracy", "f1", "roc_auc"])
     print(cv_results['test_roc_auc'].mean())
     print(cv_results['test_accuracy'].mean())
```

```
0.9270093249169478
0.8711714518683212
```

```
[ ]  # Hiperparametre Optimizasyonu Sonucu Test Veri Setinde Model Başarısı
     cv_results = cross_validate(catboost_final, test_X, test_y, cv=5, scoring=["accuracy", "f1", "roc_auc"])
     print(cv_results['test_roc_auc'].mean())
     print(cv_results['test_accuracy'].mean())
```

```
0.9191278084054346
0.8652901124238837
```

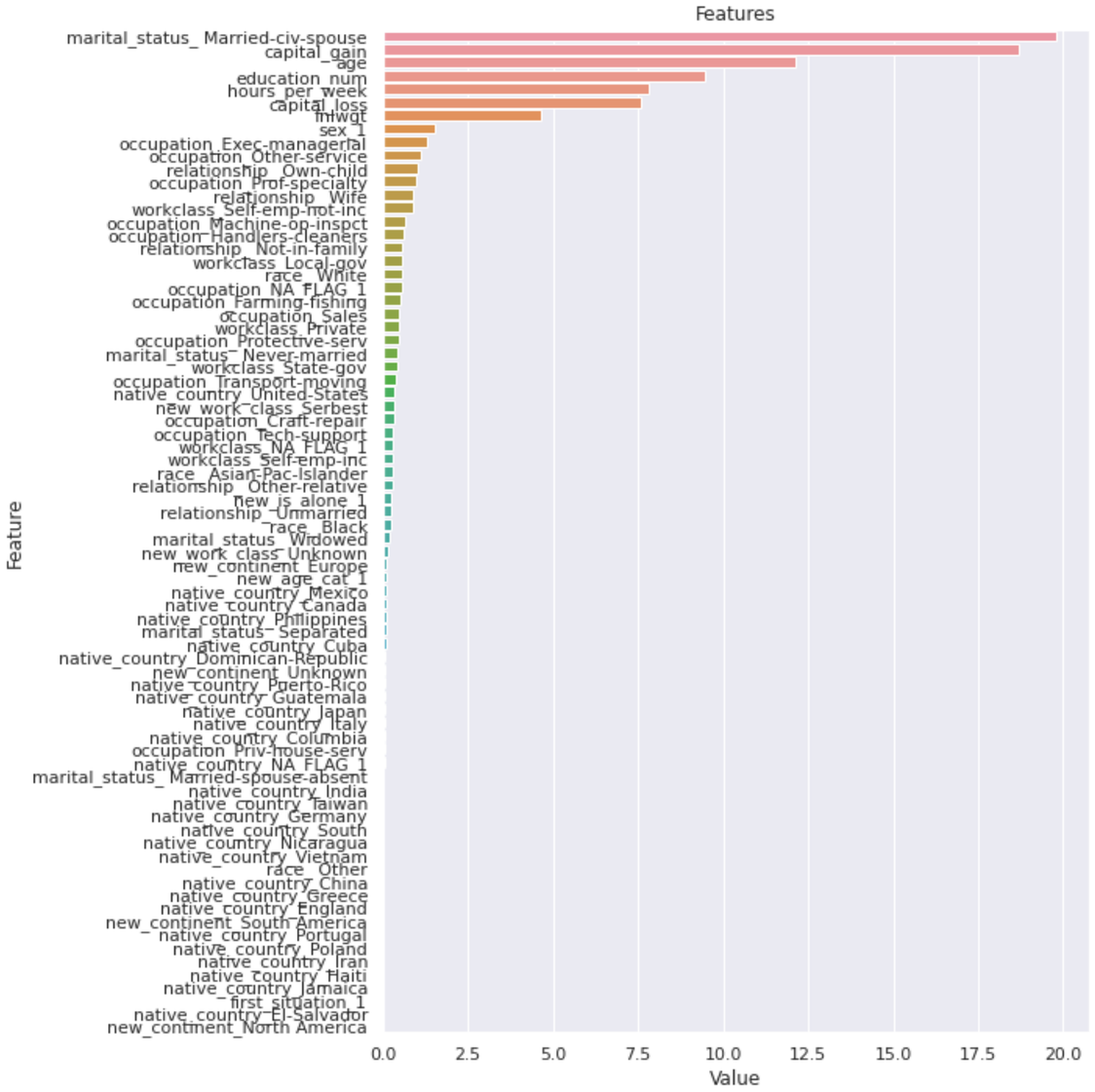
Feature Importance

```
# Feature Importance


def plot_importance(model, features, num=len(test_X), save=False):
    feature_imp = pd.DataFrame({'Value': model.feature_importances_, 'Feature': features.columns})
    plt.figure(figsize=(10, 10))
    sns.set(font_scale=1)
    sns.barplot(x="Value", y="Feature", data=feature_imp.sort_values(by="Value",
                                                     ascending=False)[0:num])
    plt.title('Features')
    plt.tight_layout()
    plt.show()
    if save:
        plt.savefig('importances.png')


plot_importance(catboost_final, test_X)
```

Sonuç ve Değerlendirme



Amerika Birleşik Devletleri' nin 1994 Nüfus Sayım Sonuçları' na ait veri seti üzerinden gerçekleştirilen gelir tahmini çalışmasında % 92' lik başarıyla demografik ve sosyo-ekonomik bilgileri verilen bir personanın gelir düzeyinin 50K' dan büyük olup olmadığı tahmin edilmiştir.

Model çalışmasında türetilen özelliklerinin modele etkisinin kayda değer olmadığı görülmüş olup gelir düzeyinin 50K' dan büyük olmasına etki eden en önemli değişkenlerin sivil bir eşle evli olma, yaş, eğitim durumu, haftalık çalışma süresi ve erkek olma olduğu görülmüştür.

Sabrınız ve dinlediğiniz için

Teşekkürler!