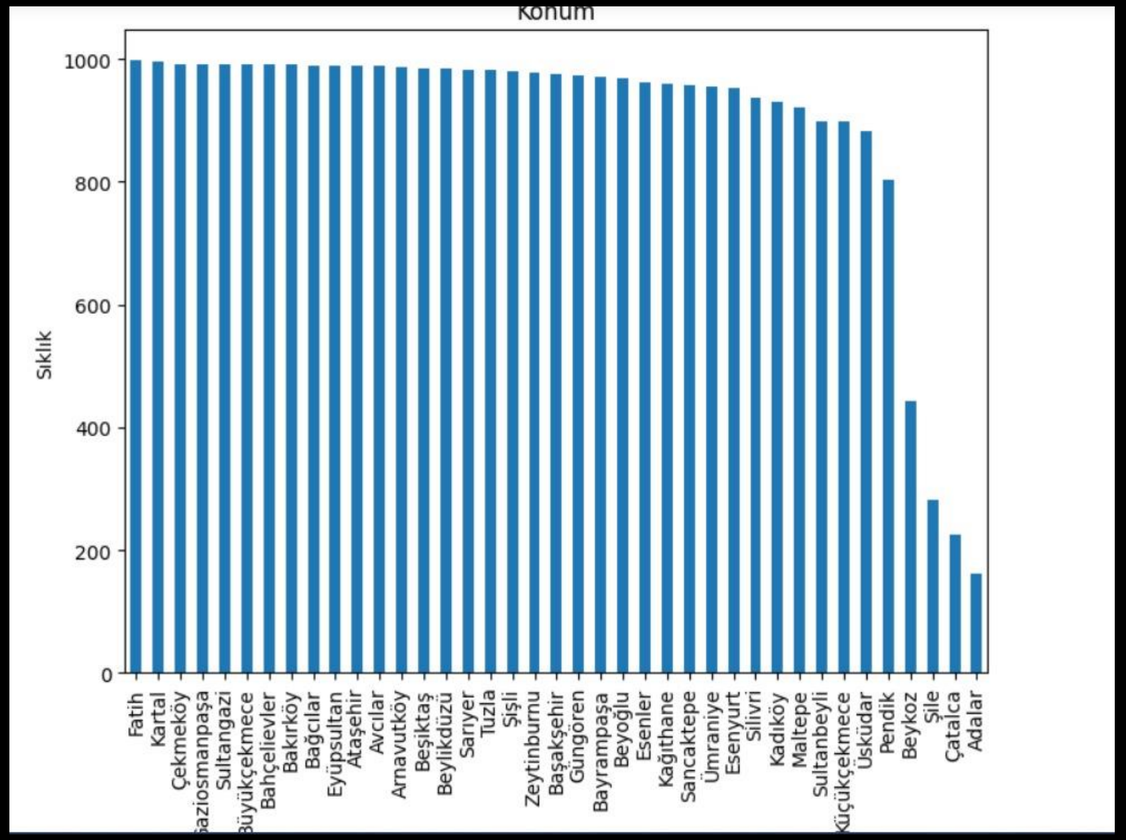
Farklı Makine Öğrenimi Algoritmaları Kullanılarak İstanbul Konut Fiyatlarının Tahmini ve Karşılaştırılması

"SVM, Random Forest, Polynomial Regression ve Yapay Sinir Ağı"

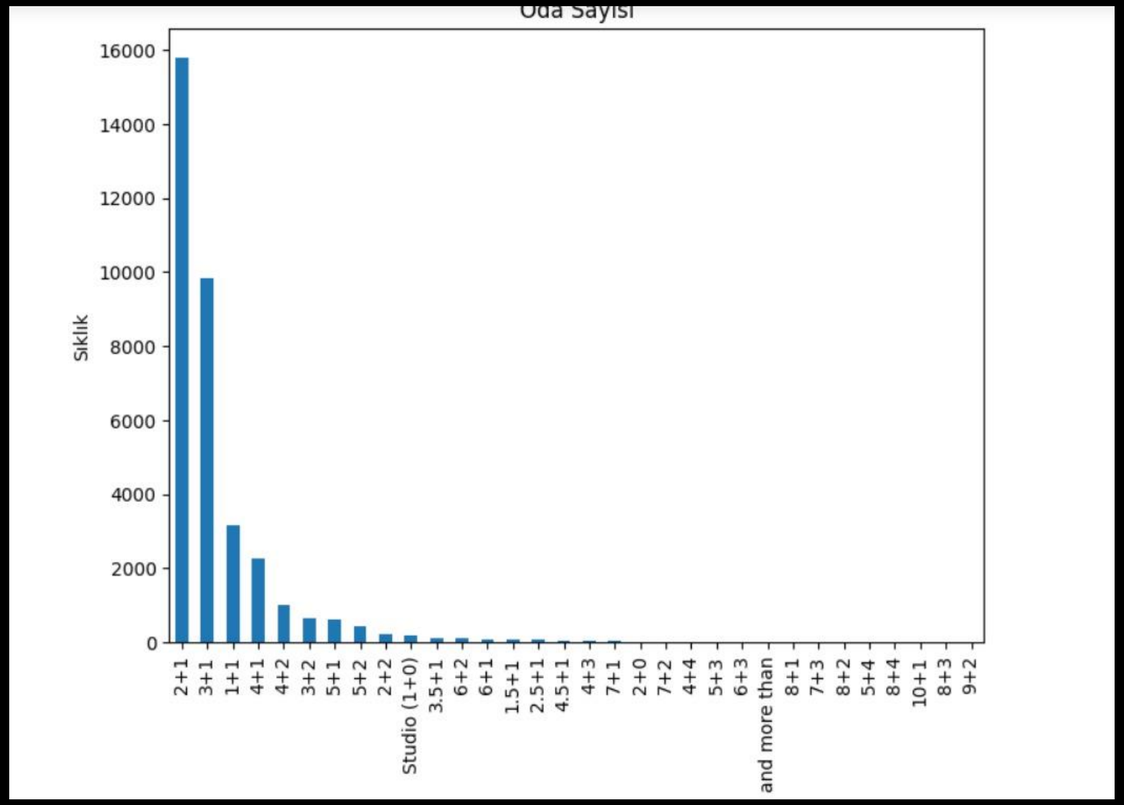
İnsanların barınma ihtiyacını karşılamak için kullanılan evler, yaşanan coğrafya, kullanılan malzeme vb. açısından çeşitli farklılıklar göstermektedir. Bu farklılıklar, yapısal, güvenlik, ısınma, donanım ve yakınlık olarak beş ayrı kategori altında toplanabilir. Her bir kategori ise kendi alt gruplarına sahiptir. Bu çalışmada İstanbul ilinin 39 ilçesindeki toplam 34844 konut incelenerek oluşturulan ve 179 kriterden oluşturulan HomeSalesData.csv dosyası kullanılmıştır. Geliştirilen model, 800 adet örnek için test edilerek destek vektör makineleri, random forest, polinomal resgesyon ve yapay sinir ağı ile analiz edilmiştir . Ayrıca bazı kriterlerin sıklığı ile ilgili analiz yapılmıştır.

Kullanılan veri seti

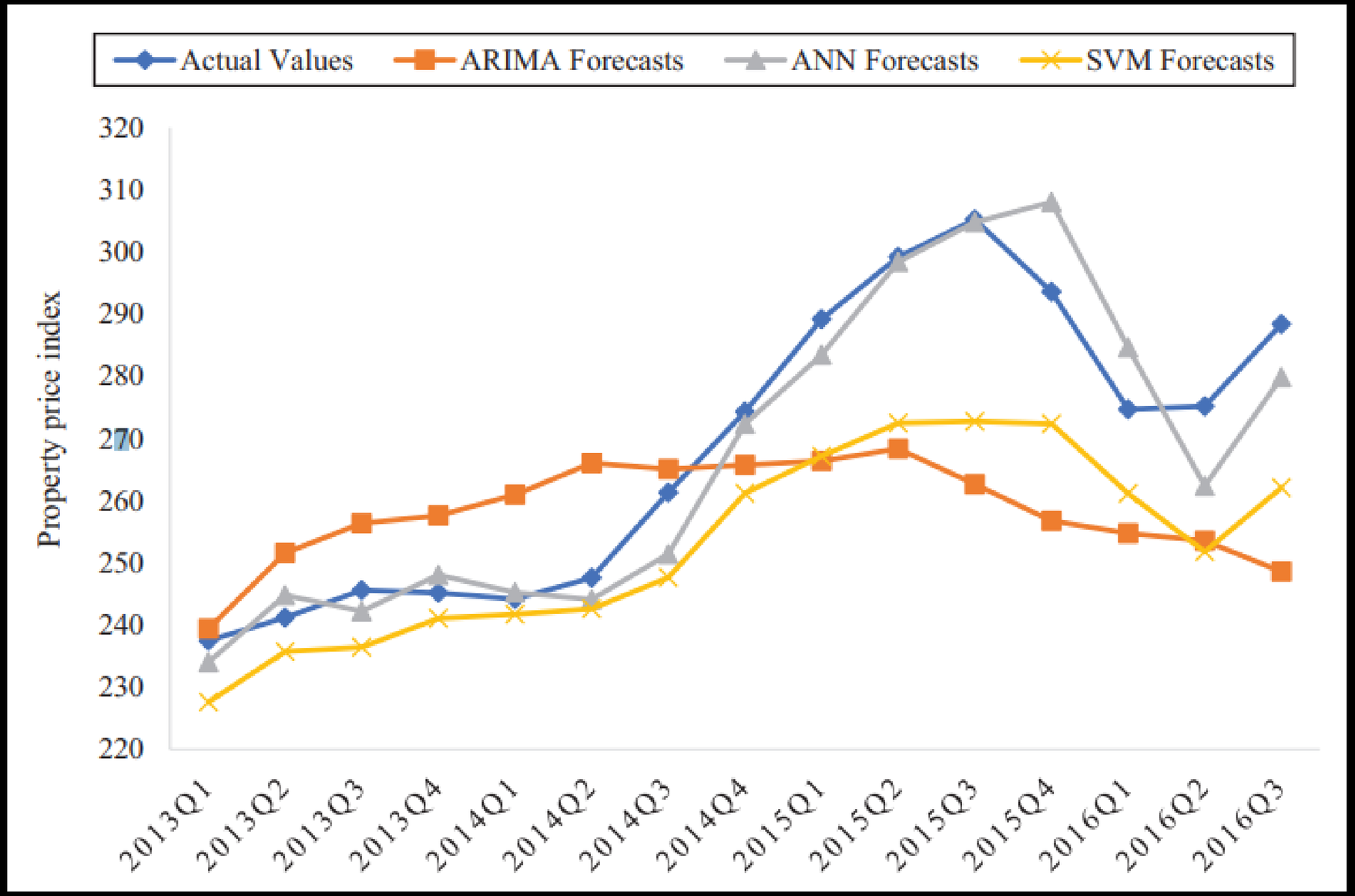
Konuma göre ev sayısı:



Oda sayısına göre ev sayısı:



- Benzer bir çalışma, 2019 yılında Hong Kong için gerçekleştirilmiştir. Bu çalışmada, veriler otoregresif entegre hareketli ortalama (ARIMA), yapay sinir ağı (ANN) ve destek vektör makinesi (SVM) modellerine tabi tutulmuştur. Ardından geliştirilen modeller, emlak fiyatlarına dair örnek dışı tahminler üretmek amacıyla kullanılmıştır.



- Bir diğer çalışma, 2021 yılında Kuzey İtalya'nın iki şehri (Brescia ve Varese) için konut fiyatlarına dair bir veri seti toplanılmış. Bu veriler, altı farklı özellik kullanılarak ev fiyatlarını tahmin etmek amacıyla en popüler üç makine öğrenimi modeli (ElasticNet, XGBoost ve Yapay Sinir Ağı) ile eğitilip test edilmek üzere kullanılmıştır.

Model	Brescia		Varese	
	MAE score	Relative difference (%)	MAE score	Relative difference (%)
ElasticNet	92,988 €	–	62,921 €	–
XGBRegressor	81,025 €	13	58,990 €	6
ANN	77,015 €	5	56,128 €	5

Note(s): The percentage shown in the "Relative difference" column is calculated concerning the model above

Table 4. MAE Score for the three prediction algorithms: the smaller is the MAE, the better is the predictor

Kullanılan Algoritmalar:

- 1.SVM:
- Bir sınıflandırma ve regresyon algoritmasıdır. İki sınıf arasındaki ayrım çizgisini (veya düzlemi) maksimize eden bir hiperdüzlem oluşturarak çalışır.

```
#verilerin olceklenmesi
from sklearn.preprocessing import StandardScaler

sc1=StandardScaler()

x_olcekli = sc1.fit_transform(X)

sc2=StandardScaler()
y_olcekli = np.ravel(sc2.fit_transform(Y.reshape(-1,1)))

#Destek Vektör makine
from sklearn.svm import SVR

svr_reg = SVR(kernel='poly') #rbf linear sigmoid
svr_reg.fit(x_olcekli,y_olcekli)

print('SVR R^2 degeri: ',r2_score(y_olcekli, svr_reg.predict(x_olcekli)))

y_pred_svr = svr_reg.predict(x_olcekli)
y_pred_svr_original_scale = sc2.inverse_transform(y_pred_svr.reshape(-1, 1))

mae_svr = mean_absolute_error(Y, y_pred_svr_original_scale)

print('SVR MAE değeri:', mae_svr)
```

```
SVR R^2 degeri:  0.6043898535992259
SVR MAE değeri: 223376.6485322009
```

Kullanılan Algoritmalar:

- 2.Random Forest:
- Birçok karar ağacının bir araya getirilerek oluşturulduğu bir enseble (birleşik) öğrenme algoritmasıdır. Her bir karar ağacı, rastgele seçilen alt örnek verileri ve rastgele seçilen özelliklerle eğitilir.

```
#Random Forest Regresyonu
from sklearn.ensemble import RandomForestRegressor
rf_reg=RandomForestRegressor(n_estimators = 8,random_state=0)
rf_reg.fit(X,Y.ravel())
print('Random Forest R2 degeri')
print(r2_score(Y, rf_reg.predict(X)))

mae = mean_absolute_error(Y, rf_reg.predict(X))
print('Random Forest MAE değeri:')
print(mae)
```

```
Random Forest R2 degeri
0.9068531897930552
Random Forest MAE değeri:
99178.30140625
```

Kullanılan Algoritmalar:

- 3.Polynomial Regression:
- Doğrusal olmayan ilişkileri modellemek için kullanılan bir regresyon yöntemidir. Bu yöntem, bağımlı değişken ile bir veya daha fazla bağımsız değişken arasındaki ilişkiyi ifade etmek üzere polinom derecesi kullanır.

```python
#polynomial regression
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree = 2)
x_poly = poly_reg.fit_transform(X)
lin_reg = LinearRegression()
lin_reg.fit(x_poly,y)

print('Polynomial R2 degeri:')
print(r2_score(Y, lin_reg.predict(poly_reg.fit_transform(X))))

mae_poly = mean_absolute_error(Y, lin_reg.predict(poly_reg.fit_transform(X)))
print('Polynomial MAE değeri:')
print(mae_poly)
```

```
Polynomial R2 degeri
0.8019083756418168
Polynomial MAE değeri:
220658.0058203125
```

Kullanılan Algoritmalar:

- 4.Yapay Sinir Ağı:
- YSA, birbirine bağlı sinir hücrelerinden oluşan katmanlar arasındaki ağırlıkları öğrenerek karmaşık ilişkileri modelleyebilir. Giriş, gizli ve çıkış katmanlarından oluşan bu yapısı sayesinde geniş bir uygulama alanına sahiptir ve özellikle görüntü işleme, doğal dil işleme ve sınıflandırma problemlerinde kullanılır.

```
# Yapay Sinir ağı
import keras
from keras.models import Sequential
from keras.layers import Dense
sc=StandardScaler()

X_train = sc.fit_transform(x_train)
X_test = sc.transform(x_test)

classifier = Sequential()

classifier.add(Dense(88, kernel_initializer = 'he_normal', activation = 'linear' , input_dim = 176))
classifier.add(Dense(88, kernel_initializer = 'he_normal', activation = 'relu'))
classifier.add(Dense(1, kernel_initializer = 'he_normal', activation = 'linear'))

classifier.compile(optimizer = 'adam', loss =  'mean_squared_error' , metrics = ['mae'] )

classifier.fit(X_train, y_train, epochs=100)

y_pred = classifier.predict(X_test)
```

```
Epoch 92/100
17/17 [==============================] - 0s 2ms/step - loss: 751368470528.0000 - mae: 282464.0000
Epoch 93/100
17/17 [==============================] - 0s 2ms/step - loss: 744802418688.0000 - mae: 282051.8438
Epoch 94/100
17/17 [==============================] - 0s 1ms/step - loss: 740023795712.0000 - mae: 282410.8438
Epoch 95/100
17/17 [==============================] - 0s 2ms/step - loss: 733084844032.0000 - mae: 282440.6875
Epoch 96/100
17/17 [==============================] - 0s 2ms/step - loss: 728347181056.0000 - mae: 283040.3438
Epoch 97/100
17/17 [==============================] - 0s 1ms/step - loss: 723763068928.0000 - mae: 283631.1875
Epoch 98/100
17/17 [==============================] - 0s 1ms/step - loss: 717746995200.0000 - mae: 284446.7500
Epoch 99/100
17/17 [==============================] - 0s 1ms/step - loss: 713928671232.0000 - mae: 285324.7500
Epoch 100/100
17/17 [==============================] - 0s 1ms/step - loss: 707955458048.0000 - mae: 285834.3125
9/9 [==============================] - 0s 1ms/step
```

45]:

```
print(y_test[1],y_pred[1])
print(y_test[50],y_pred[50])
print(y_test[160],y_pred[160])
mae_nn = mean_absolute_error(y_test, y_pred)

print('Neural Network MAE değeri:', mae_nn)
```

```
207000.0 [329982.1]
120000.0 [715401.8]
280000.0 [585787.]
Neural Network MAE değeri: 305547.14945937647
```

Sonuç:

Her algoritma için elde edilen en iyi sonuç:

Algoritmalar	R^2 değerleri
SVM “Destek Vektör Makineleri“	0.22
Random Forest	0.90
Polynomial Regression	0.80
Yapay Sinir Ağı	.

Algoritmalar	MAE değeri
SVM “Destek Vektör Makineleri“	223376.6
Random Forest	99178.3
Polynomial Regression	220658.005
Yapay Sinir Ağı	305547.149

Referanslar:

https://doi.org/10.21923/jesd.690278
https://doi.org/10.29228/TurkishStudies.43161
https://www.kaggle.com/datasets/emrahaydemr/home-sales-data-details-in-istanbul/data
https://www.emerald.com/insight/content/doi/10.1108/IJHMA-11-2018-0095/full/html
https://www.emerald.com/insight/content/doi/10.1108/JPIF-08-2021-0073/full/html

Thanks!