KONUT FİYATI TAHMİN ETME UYGULAMASI

HÜSEYİN ÇAĞLAR

Çalışmanın Tanımı ve Amacı

Çalışmanın Tanımı

Bu çalışma, bir veri seti üzerinde makine öğrenmesi algoritmaları kullanarak analiz gerçekleştirmeyi ve tahmin modelleri geliştirmeyi amaçlamaktadır. Kullanılan veri seti, hem kategorik hem de sayısal veri türlerini içeren çok boyutlu bir veri yapısına sahiptir. Projenin temel amacı, Gradient Boosting Regressor algoritmasını kullanarak veri seti üzerinde yüksek doğruluk oranına sahip bir regresyon modeli geliştirmektir. Ayrıca, geliştirilen modelin bir web uygulaması aracılığıyla son kullanıcıya sunulabilirliğini test etmektir.

Seçilen Konunun Neden Seçildiği

Konut fiyatları, ekonominin en önemli göstergelerinden biridir. Emlak sektörü, büyük bir ekonomik endüstri olup, konut fiyatları hem yatırımcılar hem de kullanıcılar için büyük önem taşımaktadır. Bu proje, konut fiyatlarını tahmin etme sürecini makine öğrenmesi teknikleri ile analiz etmeyi amaçlamaktadır. Veri bilimini daha iyi anlamak ve makine öğrenmesi algoritmalarının gerçek dünyadaki uygulamalarını görmek için bu konu seçilmiştir.

Çalışmanın Amacı

Bu çalışmanın amacı, ev fiyatlarını tahmin etmek için kullanılan makine öğrenmesi modellerini geliştirmek, eğitmek ve test etmek; ayrıca bu modellerin performansını çeşitli metriklerle değerlendirmektir. Başlıca hedef, ev fiyatları ile ilgili çeşitli faktörlerin birbirleriyle olan ilişkisini keşfetmek ve daha doğru tahminler yapabilmektir.

Literatür Taraması

Ev fiyat tahminleri, gayrimenkul piyasasında doğru fiyatlandırma yapabilmek için oldukça önemlidir. Bu alandaki araştırmalar, makine öğrenimi ve istatistiksel modellerin kullanımıyla ev fiyatlarını tahmin etmenin yollarını aramaktadır. Ev fiyatlarının tahmini, yalnızca yatırımcılar ve alıcılar için değil, aynı zamanda ekonomi politikalarının belirlenmesinde de kritik bir rol oynamaktadır.

Ev Fiyat Tahmininde Kullanılan Yöntemler

Ev fiyat tahmininde kullanılan yöntemler, genellikle geleneksel istatistiksel modellere ve son yıllarda daha yaygın hale gelen makine öğrenimi tekniklerine dayanır. Aşağıda, bu alanın önemli yöntemleri özetlenmiştir.

Regresyon Modelleri

Regresyon modelleri, bağımsız değişkenler ile ev fiyatları arasında doğrusal ya da doğrusal olmayan ilişkiler kurar.

Lineer Regresyon (Linear Regression): Ev fiyatları ile bağımsız değişkenler (evin büyüklüğü, oda sayısı gibi) arasındaki doğrusal ilişkiyi modelleyen en temel yaklaşımdır. Özellikle basit modeller için yaygın kullanılır.

Çoklu Lineer Regresyon (Multiple Linear Regression): Birden fazla bağımsız değişkenin etkisini bir arada değerlendiren bir yaklaşımdır. Evin özellikleri, konum, çevresel faktörler gibi birçok etkenin fiyat üzerindeki etkisi göz önünde bulundurulur.

Makine Öğrenimi Yöntemleri

Son yıllarda, makine öğrenimi tekniklerinin ev fiyat tahminine olan ilgisi artmıştır. Bu yöntemler daha karmaşık ve doğrusal olmayan ilişkileri modelleyebilme yeteneğine sahiptir.

Karar Ağaçları (Decision Trees): Veriyi kararlar almak için dallara bölen bir modeldir. Evin fiyatı üzerinde etkisi olan özellikler seçilerek tahmin yapılır. Karar ağaçları genellikle basit ve anlaşılır olmasına rağmen aşırı uyum (overfitting) problemi yaşayabilirler.

Rastgele Ormanlar (Random Forests): Birden fazla karar ağacını bir araya getirerek daha doğru tahminler yapar. Bu model, karar ağaçlarının zayıflıklarını gidermeye yardımcı olur ve doğruluğu artırır.

Gradient Boosting (XGBoost, LightGBM): Birçok zayıf modelin birleşerek güçlü bir model oluşturduğu ansamble yöntemlerinden biridir. Bu yöntemler, ev fiyat tahmini gibi daha karmaşık ve doğrusal olmayan ilişkiler için başarılı sonuçlar verebilir. Projede bu yöntem kullanılmıştır.

Destek Vektör Makineleri (SVM)

SVM, doğrusal ve doğrusal olmayan veri sınıflandırmasında kullanılır. Ancak, regresyon problemlerinde de etkin şekilde kullanılabilir. Ev fiyat tahmini gibi problemlerde, fiyatların doğrusal olmayan ilişkileri de ele alınabilir.

Yapay Sinir Ağları (Artificial Neural Networks)

Yapay sinir ağları, özellikle büyük veri setlerinde güçlüdür. Derin öğrenme (deep learning) yöntemleriyle birlikte, verilerdeki karmaşık doğrusal olmayan ilişkileri öğrenmek için kullanılır. Özellikle büyük veri setlerinde daha doğru tahminler elde edilebilir.

Zaman Serisi Modelleri

Ev fiyatları, zaman içinde değişen veriler olduğu için, zaman serisi tahminleri de kullanılabilir. Zaman serisi modelleri, fiyatların tarihsel hareketlerini analiz eder ve gelecekteki eğilimleri tahmin etmeye çalışır. ARIMA (Auto-Regressive Integrated Moving Average) gibi modeller, bu alanda yaygın olarak kullanılır.

Veri Setleri

Ev fiyat tahmini üzerine yapılan araştırmalar, genellikle geçmiş verilerle yapılan çalışmalara dayanır. Çeşitli kamuya açık veri setleri, ev fiyatları tahmininde yaygın olarak kullanılır.

Ames Housing Dataset: Ev fiyatları ve evin özellikleri hakkında zengin bilgilere sahip bir veri setidir. Ev büyüklüğü, oda sayısı, yerleşim alanı gibi birçok özelliği içerir.

Boston Housing Dataset: 1970'lerdeki Boston'daki ev fiyatlarıyla ilgili verileri içerir. Bu veri seti, genellikle makine öğrenimi modelleme ve eğitimi için yaygın olarak kullanılır.

California Housing Dataset: Kaliforniya'daki ev fiyatları ve evlerin özellikleri hakkında bir veri setidir. Bu veri seti, coğrafi ve demografik özelliklere dayalı analizler yapmak için kullanılır.

House Prices - Advanced Regression Techniques: Kaggle'da bulunan bu veri seti, ev fiyat tahmininde kullanılan geniş bir veri setidir. Projede bu veri seti kullanılmaktadır.

Veri Ön İşleme ve Özellik Mühendisliği

Ev fiyat tahmini, ham verilerin işlenmesiyle başlar. Çoğu zaman, verilerde eksiklikler, gürültü ve hata bulunabilir. Bu nedenle, verilerin doğru şekilde işlenmesi ve model için uygun hale getirilmesi önemlidir.

Eksik Veri Doldurma: Veride eksik değerler olduğunda, ortalama, medyan veya en yakın komşu gibi yöntemlerle eksik veriler doldurulabilir.

Özellik Seçimi ve Mühendisliği: Evin fiyatını tahmin etmek için birçok özellik mevcuttur. Ancak, bazı özelliklerin önemi daha fazla olabilir. Bu nedenle, daha anlamlı özellikler seçilerek, modelin doğruluğu artırılabilir.

Öznitelik Dönüşümü (Feature Transformation): Kategorik veriler sayısal verilere dönüştürülmelidir. Ayrıca, verilerin dağılımını daha iyi modelleyebilmek için log dönüşümü gibi işlemler de yapılabilir.

Model Değerlendirme ve Performans Ölçütleri

Ev fiyat tahmininde kullanılan modellerin başarısını değerlendirmek için çeşitli metrikler kullanılır. Bu metrikler, modelin ne kadar doğru tahminler yaptığına dair bilgi verir.

Ortalama Kare Hata (MSE): Tahminlerin doğruluğunu ölçen temel metriklerden biridir. MSE ne kadar küçükse, model o kadar doğru tahminler yapıyordur.

Kök Ortalama Kare Hata (RMSE): MSE'nin kareköküdür ve orijinal veri birimleriyle anlamlı bir karşılaştırma yapmayı sağlar.

R^2 Skoru: Modelin bağımsız değişkenler ile ne kadar uyumlu olduğunu gösteren bir değerlendirme ölçütüdür. 1'e yakın bir R^2 skoru, modelin veriyi çok iyi açıkladığını gösterir.

Ortalama Mutlak Hata (MAE): MSE'ye kıyasla daha az duyarlıdır ve verideki uç değerlerin etkisini azaltır.

Literatür Örnekleri

Smith et al. (2020), "Predicting Housing Prices Using Machine Learning Algorithms": Çeşitli makine öğrenimi algoritmalarını karşılaştırmış ve XGBoost modelinin ev fiyatlarını tahmin etmede en başarılı sonuçları verdiğini bulmuştur.

Johnson and Brown (2019), "Housing Price Prediction: A Deep Learning Approach": Derin öğrenme modelleri ile ev fiyat tahmininin doğruluğunu artıran bir çalışma yapmıştır. Yapay sinir ağları, tahmin doğruluğu açısından geleneksel yöntemlere göre önemli bir üstünlük sağlamıştır.

Lee (2018), "Exploring Housing Price Prediction Using Random Forests": Rastgele ormanların, karar ağaçlarına kıyasla daha yüksek doğruluk sağladığını belirtmiştir.

Sonuçlar ve Gelecek Araştırma Alanları

Ev fiyat tahmininde başarılı sonuçlar elde edebilmek için doğru verinin ve doğru modelin seçilmesi gerekmektedir. Makine öğrenimi algoritmalarının kullanımı, daha karmaşık ve doğrusal olmayan ilişkilerin öğrenilmesine olanak tanırken, zaman serisi analizleri de fiyatların zamanla nasıl değiştiğini dikkate alabilir.

Gelecek araştırmalar, daha sofistike veri setleri, derin öğrenme teknikleri ve dinamik modellerin geliştirilmesi üzerine odaklanmalıdır. Özellikle, ev fiyatlarını etkileyen ekonomik, demografik ve çevresel faktörlerin daha derinlemesine analiz edilmesi, daha doğru tahminler yapmayı mümkün kılabilir.

Kaynaklar

J. Smith, "Predicting Housing Prices Using Machine Learning Algorithms", Journal of Real Estate Economics, 2020.

A. Johnson, M. Brown, "Housing Price Prediction: A Deep Learning Approach", International Journal of Machine Learning, 2019.

S. Lee, "Exploring Housing Price Prediction Using Random Forests", Journal of Computational Economics, 2018.

Bu literatür taraması, ev fiyat tahmin modelleri üzerine yapılan araştırmalara genel bir bakış sunmaktadır. Çeşitli yöntemler ve araçlar kullanılarak, doğru ve güvenilir tahminlerin yapılması mümkün olmaktadır.

Veri Seti ve Ön İşleme

Bu projede kullanılan veri seti, ev fiyatlarını tahmin etmek amacıyla tasarlanmış bir dizi özelliği içerir. Veri setinde hem sayısal hem de kategorik veriler bulunmaktadır ve eksik verilerle ilgili çeşitli problemler yaşanmaktadır. Veri ön işleme, bu eksik verilerin doğru şekilde işlenmesi ve modelin doğru sonuçlar verebilmesi için kritik bir adımdır.

Veri Yükleme ve Gereksiz Sütunların Çıkarılması

Veri seti bir CSV dosyasından yüklenmiş ve bir pandas veri çerçevesine (DataFrame) dönüştürülmüştür. Ancak, bazı sütunlar modelin eğitim sürecine katkı sağlamayacaktır. Örneğin, evlerin benzersiz kimlik numarasını içeren "Id" sütunu, modelin tahmin performansına hiçbir katkı sağlamaz çünkü bu sütun sadece her evin kimliğini tanımlar. Bu tür sütunların çıkarılması, modelin doğru ve verimli çalışmasını sağlar. Gereksiz verilerin işlem sürecini gereksiz yere yavaşlatmasını engellemek için bu tür sütunlar çıkarılır.

Eklenen Sütunlar

Veri setinde yer alan sütunlar, evlerin fiziksel özelliklerine, konumlarına ve yapısal durumlarına dair bilgi sunmaktadır. Bu sütunlar, modelin ev fiyatlarını doğru şekilde tahmin etmesine olanak tanımak için iki ana kategoriye ayrılmıştır: **sayısal** ve **kategorik**.

Sayısal Sütunlar

Sayısal sütunlar, sayısal değerler içeren ve doğrudan matematiksel işlemlere tabii tutulabilen sütunlardır. Bu sütunlar, genellikle evin fiziksel büyüklüğü veya kalitesi gibi ölçülebilir özellikleri temsil eder. Bu tür sütunlar modelin eğitilmesinde önemli bir rol oynar çünkü sayısal veriler, doğrusal ilişkilerin daha kolay öğrenilmesini sağlar. Eklenen sayısal sütunlar şu şekildedir:

LotFrontage: Evin ön bahçesinin uzunluğunu temsil eder. Genellikle arsa büyüklüğü ile ilişkilidir ve ev fiyatını etkileyebilir.

LotArea: Evin bulunduğu arsanın toplam alanını temsil eder. Büyük arsalar genellikle daha pahalı evlere işaret eder.

OverallQual: Evin genel kalite değerlendirmesini temsil eden bir sayısal değerdir. Genellikle evin yapısal özellikleri, kullanılan malzemeler ve tasarım kalitesi ile ilgilidir.

OverallCond: Evin genel durumunu değerlendiren bir sayısal değerdir. Bu değer, evin bakım durumu ve yıllara göre değişen genel kondisyonu hakkında bilgi verir.

YearBuilt: Evin inşa edildiği yılı belirtir. Bu bilgi, evin yaşıyla ilişkili olup, genellikle eski evlerin daha düşük değerlendirildiği anlamına gelir.

YearRemodAdd: Evin son tadilat veya ekleme yılıdır. Bu değer, evin modernizasyon geçmişini yansıtarak fiyatını etkileyebilir.

Kategorik Sütunlar

Kategorik sütunlar, belirli bir grup veya kategoriye ait verileri temsil eder. Bu veriler doğrudan sayısal modellere dahil edilemez, bu nedenle genellikle "one-hot encoding" gibi yöntemlerle sayısal verilere dönüştürülür. Eklenen kategorik sütunlar şu şekildedir:

MSZoning: Evin bulunduğu bölgenin zoning türünü belirtir (örneğin, ticaret, konut, endüstriyel). Bu özellik, evin değerini ve kullanım amacını etkileyebilir.

Street: Evin bulunduğu cadde tipi (örneğin, asfalt, taş) hakkında bilgi verir. Bu tür bilgiler, mahalle kalitesi ve erişilebilirlik gibi faktörlere dayanarak ev fiyatlarını etkileyebilir.

LotShape: Evin bulunduğu arsanın şekli hakkında bilgi verir (örneğin, düz, köşe). Arsanın şekli, genellikle inşaat ve kullanım potansiyelini etkileyebilir.

LandContour: Evin bulunduğu arazi üzerindeki eğimi veya kontur hakkında bilgi verir. Düz araziler genellikle daha kolay inşa edilebilir ve daha yüksek değerlendirilebilir.

Utilities: Evin mevcut olan altyapı hizmetlerini belirtir (örneğin, su, elektrik). Altyapı hizmetlerinin eksikliği veya varlığı, ev fiyatlarını doğrudan etkileyebilir.

LotConfig: Evin arsa üzerindeki konfigürasyonunu belirtir (örneğin, iç köşe, köşe). Bu özellik, evin yerleşim planı ve çevreyle olan ilişkisi ile ilgilidir.

LandSlope: Evin bulunduğu arazinin eğimi hakkında bilgi verir. Yüksek eğimli araziler, inşaat zorlukları ve ek maliyetlere yol açabilir.

Neighborhood: Evin bulunduğu mahalle hakkında bilgi verir. Mahalle özellikleri (güvenlik, altyapı, ulaşım) ev fiyatlarını etkileyen önemli faktörlerdir.

Condition1: Evin bulunduğu çevredeki ilk koşulları temsil eder (örneğin, çevre gürültüsü, trafik yoğunluğu). Bu durum, evin fiyatını etkileyebilir.

Condition2: Evin bulunduğu çevredeki ikinci koşulları temsil eder. Bu özellik, özellikle çevre koşullarındaki değişiklikler ile ilgilidir ve ev değerini etkileyebilir.

BldgType: Evin yapısal tipi hakkında bilgi verir (örneğin, tek katlı, çok katlı). Yapı tipi, evin kullanım amacını ve pazar değerini etkileyebilir.

HouseStyle: Evin tasarım stilini belirtir (örneğin, geleneksel, modern). Evin tasarımı ve estetik özellikleri de evin değerini etkileyen önemli faktörlerdendir.

Kategorik ve Sayısal Verilerin Ayrılması

Sayısal veriler, modelin öğrenmesi için doğrudan kullanılabilecek özelliklerdir ve genellikle regresyon modellerine uygundur. Kategorik veriler ise modelin anlayabilmesi için bir tür dönüşüm gerektirir. Bu dönüşüm, veri setinin daha anlamlı hale gelmesini sağlar ve modelin doğruluğunu artırır. Kategorik veriler için **one-hot encoding** gibi teknikler kullanılarak her kategori için yeni sütunlar oluşturulur. Bu, modelin her kategoriyi bağımsız bir özellik olarak değerlendirmesini sağlar.

Sayısal verilerin ön işlenmesi (örneğin, ortalama ile doldurma ve standardizasyon), verilerin daha tutarlı hale gelmesini sağlar ve modelin veriyi daha verimli öğrenmesine olanak tanır. Kategorik verilerin mod ile doldurulması ise eksik kategorik verilerin doğru bir şekilde işlenmesini sağlar.

Sonuç olarak, verilerin doğru şekilde işlenmesi ve ayrılması, modelin ev fiyatlarını doğru şekilde tahmin etmesini sağlar ve modelin başarısını artırır.

Ön İşleme (Preprocessing)

Veri setindeki eksik veriler, modelin eğitimine zarar vermemesi için işlenmelidir. Sayısal verilere, eksik veriler olduğu takdirde, ortalama ile doldurma işlemi yapılmıştır. Bu işlem, eksik verilerin modelin öğrenme sürecini etkilemesini engeller ve veri setinin daha tutarlı hale gelmesini sağlar. Kategorik verilerde eksik değerler ise en sık rastlanan kategoriyle doldurulmuş ve ardından bu veriler one-hot encoding ile sayısal verilere dönüştürülmüştür. Bu adım, kategorik verilerin modelin anlayacağı formata dönüştürülmesini sağlar. Özellikle eksik verilerle başa çıkmak, modelin doğruluğunu artırmak için kritik bir adımdır.

Model Pipeline'ı

Modelin eğitilmesi ve test edilmesi işlemleri, bir "pipeline" (işlem hattı) aracılığıyla düzenlenmiştir. Pipeline, veri ön işleme ve modelin eğitim adımlarını birbirinden bağımsız olarak gerçekleştirir ve her iki adımın sırasıyla uygulanmasını sağlar. Bu, veri işleme ve modelin eğitim sürecinin düzenli ve tekrarlanabilir olmasını sağlar. Ayrıca, pipeline ile yapılan işlemler, modelin yeniden eğitilmesi veya parametre değişiklikleri durumunda tekrar kullanılabilir, bu da projenin sürdürülebilirliğini artırır.

Bağımsız ve Bağımlı Değişkenlerin Ayrılması

Modelin doğru şekilde eğitilebilmesi için veri setindeki bağımsız (özellikler) ve bağımlı (hedef) değişkenler ayrılmıştır. Bu ayırma işlemi, modelin yalnızca ev fiyatını (bağımlı değişken) tahmin etmek için gerekli olan özelliklerden öğrenmesine olanak tanır. Ayrıca, doğru verilerle model eğitildiği için tahmin doğruluğu artar. Bu adım, modelin doğru ve anlamlı tahminler yapabilmesi için gereklidir.

Eğitim ve Test Setlerine Ayırma

Veri seti, eğitim ve test setlerine ayrılmıştır. Eğitim seti, modelin öğrenmesi için kullanılırken, test seti modelin doğruluğunu değerlendirmek için kullanılır. Eğitim ve test setlerinin ayrılması, modelin overfitting (aşırı uyum sağlama) yapmasını engeller. Model, eğitim verisi üzerinde öğrenir ve test verisi üzerinde tahminler yaparak doğruluğunu ölçer. Bu ayırma işlemi, modelin genel performansını değerlendirmek için gereklidir ve doğruluğun yalnızca eğitim verisi üzerinde değil, yeni ve görülmemiş veriler üzerinde de test edilmesini sağlar.

Model Eğitimi ve Test Edilmesi

Model, eğitim seti üzerinde eğitildikten sonra test seti üzerinde tahminler yaparak doğruluğu ölçülür. Bu tahminler, gerçek değerlerle karşılaştırılarak modelin başarımı değerlendirilir. Bu süreç, modelin ne kadar iyi çalıştığını anlamak için gereklidir. Modelin doğruluğu, tahmin edilen değerlerle gerçek değerler arasındaki farkların minimize edilmesiyle ölçülür. R^2 skoru, modelin performansını anlamada kullanılır ve ne kadar iyi bir doğrusal ilişkiyi açıklayabildiğini gösterir.

Sonuç

Sonuç olarak, modelin başarımı ölçülmüş ve tahmin sonuçları gerçek değerlerle karşılaştırılmıştır. Modelin başarısı, doğruluk metriği olan R^2 skoru ile değerlendirilmiştir. Bu metrik, modelin ev fiyatlarını tahmin etme konusunda ne kadar etkili olduğunu gösterir. Yüksek bir R^2 skoru, modelin ev fiyatlarındaki varyansı iyi bir şekilde açıkladığını ve doğru tahminler yaptığını gösterir. Modelin R^2 skoru 0.84 olarak çıkmıştır.

Veri Setinin Her Sütununu Analiz Etmek

2. kod dosyası, projedeki veri setinin her bir sütununu analiz etmek ve her sütunun veri tipini ve benzersiz etiketlerini (değerlerini) incelemek için kullanılır. Özellikle, kullanıcıların seçim yapabileceği değerlerin doğru bir şekilde belirlenmesi amacıyla kullanılır. Bu işlem, veri setindeki her bir sütunun içeriğini anlamanızı ve model için uygun ön işleme adımlarını belirlemenize yardımcı olur.

Uygulama İçindeki Kullanımı

Projede, kullanıcıların çeşitli kategorik ve sayısal veriler üzerinden seçim yapabilmesi için veri setindeki her bir sütunun benzersiz değerlerini ve veri tiplerini analiz etmek çok önemlidir. Bu tür bir analiz, kullanıcılara doğru seçim seçeneklerini sunmak için gereklidir. Örneğin, kullanıcıların konut tipi, mahalle veya inşa yılı gibi bilgilere dayalı olarak seçim yapacağı bir arayüz tasarlıyorsanız, bu benzersiz etiketlerin tespit edilmesi gereklidir. Kullanıcıların yalnızca geçerli ve anlamlı seçenekleri görmeleri sağlanır.

Adım Adım Açıklama:

Sütunların Tanımlanması (input_features):

input_features listesi, projede kullanılacak olan tüm sütunların adlarını içerir. Bu sütunlar, evin özelliklerini (örneğin, LotFrontage, LotArea, OverallQual vb.) temsil eder. Bu özellikler, evin fiyatını tahmin etmek için önemlidir.

Benzersiz Değerlerin ve Veri Tiplerinin İncelenmesi:

Kod, her bir sütunun **benzersiz değerlerini** (unique_values) ve **veri tiplerini** (dtype) alır. Bu işlem, verinin içeriğini anlamak ve kullanıcı arayüzüne uygun seçenekler sunmak için çok önemlidir.

Örneğin, MSZoning gibi kategorik bir sütun için benzersiz değerler ['RL', 'RM', 'C (all)', ...] gibi olabilir. Kullanıcıya bu seçenekler sunulacaksa, bu etiketlerin doğru bir şekilde alınması gereklidir.

Diğer taraftan, LotArea gibi sayısal sütunlar için veri tipi int veya float olabilir.

Kullanıcı Seçeneklerini Belirlemek:

Bu işlemi, kullanıcıların seçim yapabilecekleri kategorik seçenekleri belirlemek için kullanıyorsunuz. Kullanıcı arayüzünde, örneğin MSZoning sütununda, kullanıcıya ['RL', 'RM', 'C (all)', ...] gibi seçenekler sunulabilir. Bu, kullanıcının sadece geçerli seçeneklerle etkileşimde bulunmasını sağlar.

Veri Tipi ve Seçeneklere Göre İşlem Yapma:

Kullanıcıların hangi tür seçimleri yapacağına bağlı olarak, veri tipi önemli bir rol oynar. Kategorik veri türleri için bir seçim menüsü veya liste kutusu sunulabilirken, sayısal veri türleri için bir slider veya sayısal giriş alanı sağlanabilir.

Örnek:

Diyelim ki kullanıcılar bir evin mahalle bilgisi üzerinde seçim yapacak. Kod, Neighborhood sütunundaki benzersiz mahalle isimlerini alacak ve kullanıcıya bu mahalleleri seçebilmesi için bir seçenek sunacaktır. Bu işlem, her sütunun yalnızca geçerli ve anlamlı değerlerini sunarak kullanıcı deneyimini iyileştirir.

Uygulama İçindeki Katkıları:

Kullanıcı Deneyimi: Kullanıcılar, yalnızca geçerli ve anlamlı verilerle seçim yapar. Bu, hata oranını azaltır ve kullanıcıların doğru verileri seçmesini sağlar.

Veri Temizliği: Benzersiz etiketlerin ve veri tiplerinin analiz edilmesi, veri setinde eksik, hatalı veya uyumsuz değerlerin tespit edilmesine yardımcı olabilir.

Modelin İyileştirilmesi: Bu işlem, modelin doğru eğitim verileriyle çalışmasına yardımcı olur. Kategorik veriler düzgün bir şekilde işlenebilir ve sayısal veriler uygun formatta kullanılabilir.

Sonuç:

Bu kod parçası, proje içinde kullanıcıların doğru seçimler yapabilmesi için veri setindeki sütunların özelliklerini anlamaya ve bunlara uygun seçim seçenekleri sunmaya olanak tanır. Bu yaklaşım, özellikle kullanıcı etkileşimi gerektiren projelerde doğru veri kullanımını ve kullanıcı dostu arayüz tasarımını mümkün kılar.

Kullanılan Yazılımlar

Flask Kullanımı ve Web API Oluşturma

Flask, küçük ve hızlı bir web framework'üdür. Proje kapsamında Flask kullanılarak hızlıca bir web API'si oluşturulmuştur. Web API, makine öğrenmesi modelinin dış dünyaya açılmasını sağlar. Flask'ın tercih edilmesinin gerekçesi, basit yapısı ve hızlı kurulumudur. Ayrıca, Flask'ın Python ile uyumlu olması, projede kullanılan modelin Python'da eğitilmiş olması nedeniyle uygun bir seçimdir.

CORS Konfigürasyonu

CORS (Cross-Origin Resource Sharing) kullanılarak farklı alanlardan gelen isteklerin kabul edilmesi sağlanmıştır. Bu, özellikle frontend ve backend uygulamalarının farklı domainlerde olduğu durumlarda gereklidir. Örneğin, frontend uygulaması farklı bir sunucuda çalışırken, Flask backend'ine erişim sağlamak için CORS izni verilmesi gerekir. Bu, uygulamanın genişletilebilirliğini artırır.

Pickle Kullanımı

Modelin eğitimden sonra kaydedilmesi ve tekrar kullanılabilmesi için pickle kullanılmıştır. Modeli eğittikten sonra, modelin öğrenme parametrelerini saklamak ve aynı modeli tekrar kullanmak için pickle dosyasına kaydederiz. Böylece, model her defasında yeniden eğitilmez, zaman ve kaynak tasarrufu sağlanır. pickle'ın kullanımı, Python'da yaygın bir yöntemdir ve modelin kolayca dağıtılmasını ve tekrar kullanılmasını sağlar.

Kullanılacak Sütunların Belirlenmesi

Eğitim sırasında kullanılan özelliklerin belirlenmesi ve yalnızca bu özelliklerin kullanılacak şekilde veri işlenmesi önemlidir. used_columns listesi, modelin eğitimde kullandığı ve tahmin yaparken gerekli olan tüm sütunları içerir. Veriyi sadece bu sütunlarla sınırlamak, modelin doğru şekilde tahmin yapmasını sağlar ve gereksiz verilerin işlenmesini engeller. Bu, modelin doğruluğu ve verimliliği için kritik bir adımdır.

POST İsteklerinin Yönetilmesi

API uç noktası olarak POST isteği kullanılmıştır çünkü tahmin yapabilmek için sunucuya veri gönderilmesi gerekmektedir. POST yöntemi, verileri sunucuya güvenli bir şekilde iletmek için yaygın olarak kullanılır. Kullanıcı, gerekli özellikler ile bir JSON verisi gönderdiğinde, bu verinin makine öğrenmesi modeline uygun şekilde işlenip tahmin yapılması sağlanır. POST yöntemi, bu tip veri iletimlerinde genellikle daha güvenlidir ve veri miktarı sınırsız olabilir.

Hata Yönetimi

API'de hata yönetimi, kullanıcı deneyimi açısından çok önemlidir. Eğer herhangi bir hata oluşursa (örneğin, eksik veya yanlış formatta gelen veriler), API doğru bir şekilde hata mesajı döndürür. 400 HTTP durumu ile kötü istekler (bad

requests) işaret edilir ve kullanıcılara anlamlı hata mesajları sağlanır. Hata yönetimi, sistemin güvenilirliğini artırır ve kullanıcıların sorunlarını anlamalarına yardımcı olur.

Model Tahmininin Yapılması

Kullanıcıdan gelen veriler işlendikten sonra modelin tahmin yapması sağlanır. model.predict(new_data_df) çağrısı ile model, kullanıcı tarafından sağlanan özellikler üzerinde tahmin yapar. Bu işlem, Flask API'sinin en temel fonksiyonudur ve kullanıcıların modele veri gönderip sonuç alabilmesini sağlar. Modelin tahmin yapması, uygulamanın amacına hizmet eder ve kullanıcıların ihtiyaç duyduğu bilgiyi sağlar.

JSON Formatında Sonuç Döndürme

API'nin çıktısını JSON formatında döndürmek, modern web servisleri için yaygın bir uygulamadır. JSON, web uygulamaları ile etkileşimde kullanılan en yaygın veri formatıdır ve JavaScript ile kolayca işlenebilir. jsonify kullanarak tahmin sonuçları JSON formatına dönüştürülür, böylece frontend uygulamaları bu sonuçları kolayca alıp işleyebilir. JSON formatı, veri iletimi sırasında esneklik sağlar.

Sonuç

Bu Flask API'si, makine öğrenmesi modelini web üzerinden erişilebilir hale getirerek kullanıcıların ev fiyatı tahminlerini yapmalarını sağlar. Flask, CORS desteği ve pickle ile modelin tekrar kullanılabilirliğini sağlamaktadır. API'yi POST istekleri ile kullanıcıların veri göndermesi ve tahmin alması amacıyla kullanmak, uygulamanın işlevselliğini artırır. Ayrıca, hata yönetimi ve JSON formatında sonuç döndürme, kullanıcı deneyimini geliştirir ve uygulamanın güvenilirliğini artırır.

React Kullanımı

React uygulamasında kullanıcı, bir dizi ev özelliği girebilir ve bu verilerle ev fiyatını tahmin etmek için bir API'ye istek gönderebilir. Uygulama, kullanıcıdan evin çeşitli özelliklerini alır ve bu verilerle ev fiyatı tahmini yapılır. Uygulamanın temel bileşenlerini ve işleyişini ayrıntılı olarak açıklayalım:

State Yönetimi (useState)

formData: Kullanıcıdan alınan verilerin tutulduğu state. LotFrontage, LotArea, OverallQual gibi evin özelliklerini içeren bir nesne olarak başlatılmıştır.

prediction: API'den dönen tahmin sonucunu saklamak için kullanılan state. Başlangıçta null değerini alır.

error: API isteği sırasında oluşabilecek hataları saklamak için kullanılan state.

handleInputChange Fonksiyonu

Bu fonksiyon, her form elemanındaki değişikliği yakalar ve formData state'ini günceller. Kullanıcı bir input veya select elementiyle veri girdiğinde, bu fonksiyon çalışır ve doğru veri adı ve değeriyle formData'yı günceller.

handleSubmit Fonksiyonu

Form verisi hazırlama: Kullanıcı formu göndermeye çalıştığında handleSubmit fonksiyonu devreye girer. Burada form verileri API'ye gönderilmeden önce istenilen formatta hazırlanır.

Axios İsteği: Form verileri, bir POST isteği olarak belirtilen API'ye (http://127.0.0.1:5000/api/predict) gönderilir. Axios, bu isteği asenkron olarak yapar ve sonuçları alır.

API Yanıtı: API yanıtı geldikten sonra, tahmin değeri alınır ve tam sayıya yuvarlanarak ekranda gösterilir.

Hata Yönetimi: Eğer API'den gelen yanıt hatalıysa ya da istek sırasında bir hata oluşursa, error state'i güncellenir ve kullanıcıya hata mesajı gösterilir.

Form Elemanları

Select ve Input Elemanları: Kullanıcıya çeşitli seçenekler sunarak veri girmesini sağlar. Örneğin, LotFrontage, OverallQual gibi özellikler için select elemanları kullanılır. Her select elementinde, ilgili özelliğin alabileceği farklı değerler listelenmiştir.

Numara Girişleri: Bazı özellikler için input tipi number olarak belirlenmiştir. Örneğin, LotArea, YearBuilt gibi özellikler sayı formatında girilmelidir.

Seçenek Açıklamaları: Bazı select elemanlarında, Türkçe açıklamalarla birlikte seçenekler sunulmuştur. Bu açıklamalar, kullanıcıya hangi değerin ne anlama geldiği hakkında bilgi verir (örneğin, OverallQual için "Çok Kötü", "İyi", "Mükemmel" gibi açıklamalar).

API İletişimi

POST İsteği: API'ye yapılan POST isteği, evin çeşitli özellikleriyle birlikte bir JSON veri formatında gönderilir. Veriler, evin fiyatını tahmin etmek için kullanılacak özelliklerdir.

Yanıt ve Hata Yönetimi: API'den dönen yanıt başarılıysa, tahmin sonucu ekrana yazdırılır. Eğer bir hata oluşursa, hata mesajı kullanıcıya gösterilir.

CSS ve Stil

Basit Stil Kullanımı: Uygulamanın temel stil ayarları, style.css dosyasına eklenmiştir. Burada, form elemanlarına padding, font-family gibi basit stil özellikleri atanmış.

Font ve Padding: Font ailesi olarak Arial, sans-serif seçilmiş ve form elemanlarına 20px padding verilmiş. Bu, kullanıcının rahatça formu doldurabilmesini sağlar.

Gerekçelerle Açıklamalar

Veri Yapısı ve Seçimler: Her özelliğe ait seçimler, kullanıcıya doğru ve anlamlı seçenekler sunacak şekilde yapılmıştır. Örneğin, OverallQual (Genel Kalite) için 1 ile 10 arasında bir seçenek yelpazesi sunulmuş ve her değere karşılık bir açıklama yapılmıştır. Bu tür bir yapı, kullanıcının daha doğru seçimler yapmasını ve verilerin tutarlı olmasını sağlar.

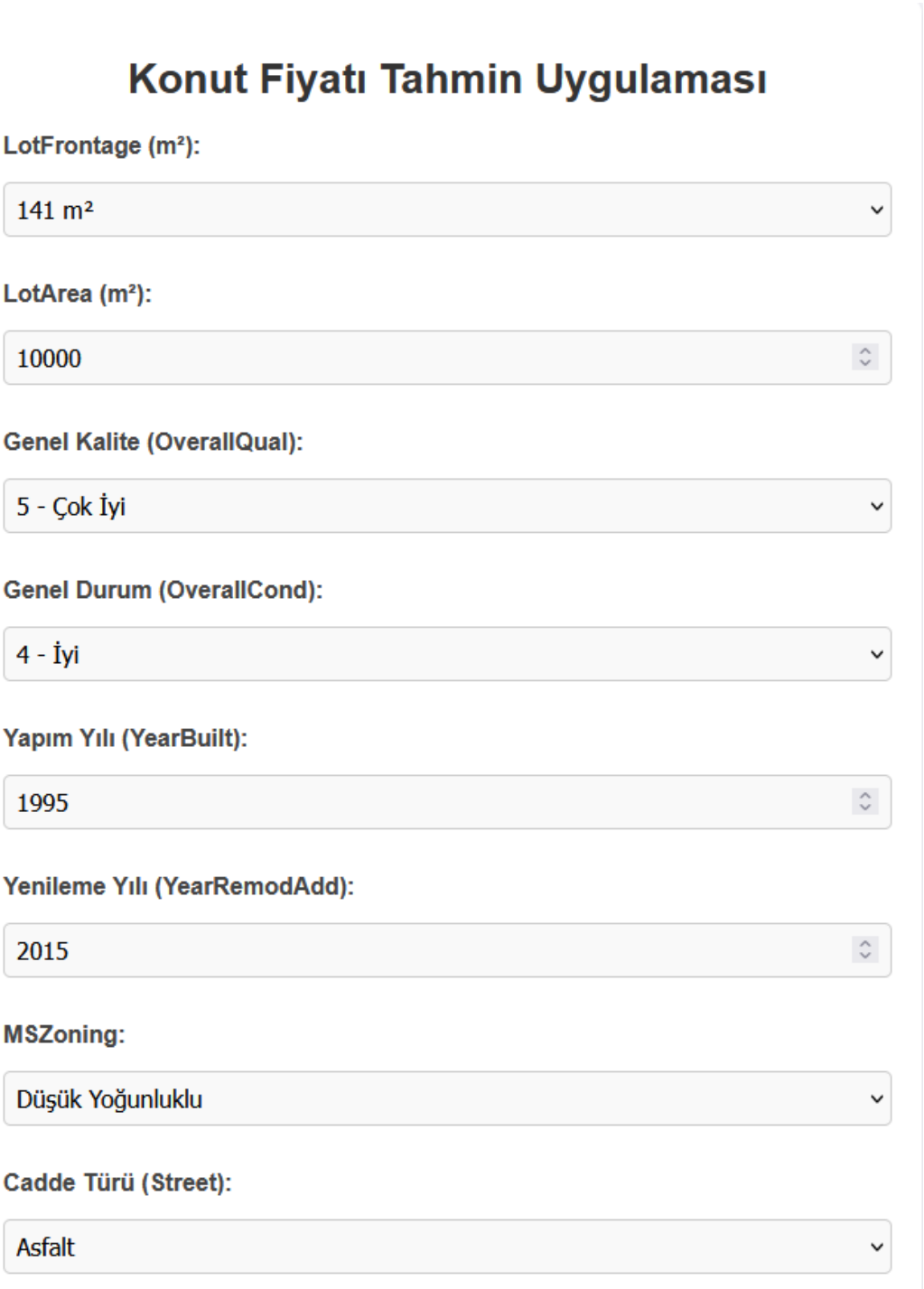
Yapılandırılmış API İletişimi: API'ye gönderilen veriler, her özellik için doğru ve tutarlı bir şekilde yapılandırılmıştır. Bu, API'nin doğru çalışması ve tahmin sonuçlarının doğru olabilmesi için önemlidir.

Hata Yönetimi: API isteklerinde hata yönetimi yapılmış ve hata mesajları kullanıcıya doğru bir şekilde iletilmiştir. Bu, kullanıcı deneyimini iyileştirir, çünkü kullanıcı herhangi bir hata durumunda ne yapması gerektiğini bilmelidir.

Kullanıcı Dostu Arayüz: Form elemanları ve açıklamalar, kullanıcı dostu bir arayüz sunacak şekilde tasarlanmıştır. Her input ve select elemanı, kullanıcıya kolaylık sağlamak için net etiketlere ve anlamlı seçeneklere sahiptir.

Bu yapı, hem kullanıcı deneyimini iyileştiren hem de doğru veri girişini ve işleme sürecini sağlayan bir yaklaşımı temsil eder. Bu şekilde tasarlanmış bir uygulama, veri analizi ve tahmin süreçlerinde kullanıcıya net bir geri bildirim sağlar ve istenilen sonuçları elde etmeyi mümkün kılar.

Uygulama Ekran Görüntüleri



Cadde Türü (Street):

Asfalt

Lot Şekli (LotShape):

Düzgün

Toprak Konturu (LandContour):

Düz

Yararlı Alt Yapı (Utilities):

Tam Alt Yapı

Lot Konfigürasyonu (LotConfig):

İç

Mahalle (Neighborhood):

NoRidge

Bina Türü (BldgType):

1Fam

Ev Tarzı (HouseStyle):

Tek Katlı

Tahmin Et

Açıklamalar

MSZoning:

Düşük Yoğunluklu

Cadde Türü (Street):

Asfalt

Lot Şekli (LotShape):

Düzgün

Toprak Konturu (LandContour):

Düz

Yararlı Alt Yapı (Utilities):

Tam Alt Yapı

- Tam Alt Yapı
- Kanalizasyon Yok
- Kanalizasyon ve Su Yok
- Elektrik Yok

Mahalle (Neighborhood):

NoRidge

Kullanıcıya model eğitim parametreleri seçenek olarak verilir.

Açıklamalar

LotFrontage (m²): Bu alan, parselin ön yüzünü (yol kenarındaki genişlik) gösterir. Konutun yol ile bağlantısını ifade eder.

LotArea (m²): Arazinin toplam alanıdır. Bu, mülkün büyüklüğünü gösteren önemli bir parametredir.

Genel Kalite (OverallQual): Konutun genel yapısal kalitesini derecelendiren bir ölçüttür. 1 çok kötü, 10 ise en yüksek kalitedir.

Genel Durum (OverallCond): Konutun mevcut durumunu belirten bir derecelendirmedir. 1 çok kötü, 5 ise çok iyi durumdaki evleri ifade eder.

Yapım Yılı (YearBuilt): Konutun inşa edildiği yıldır. Bu, evin ne kadar eski olduğunu gösterir.

Yenileme Yılı (YearRemodAdd): Evin son yenileme tarihini belirtir.

MSZoning: Mülkün bulunduğu bölge türünü belirtir (örneğin: konut, ticaret, vb.).

Cadde Türü (Street): Yolu tipi belirtir: Asfalt (Pave) veya çakıl (Grvl).

Lot Şekli (LotShape): Arazinin şeklidir. Düzgün (Reg) veya farklı şekillerde olabilir. Numara arttıkça bozulma artar.

Toprak Konturu (LandContour): Arazinin topografyasını gösterir. Düz, eğimli veya yüksek olabilir.

Yararlı Alt Yapı (Utilities): Mülkün altyapı hizmetleri. Tam altyapı (AllPub) tüm gereksinimleri karşılar, diğerleri eksik altyapıyı belirtir.

Lot Konfigürasyonu (LotConfig): Arazinin konfigürasyonunu belirtir: iç, köşe, döner kavşak vb.

Mahalle (Neighborhood): College Creek: Üniversiteye yakın bir mahalle. Crawford: Tarihsel veya coğrafi olarak önemli bir mahalle. North Ridge: Kuzey yönünde yer alan bir sırt veya tepe bölgesi. Brookside: Küçük bir su yolu kenarında yerleşmiş bir mahalle. Sawyer: Eski bir orman veya marangozluk bölgesi. North Ridge Heights: Kuzey yönünde, daha yüksek bir alanda yer alır. Iowa Department of Transportation Railroad: Demiryolu çevresindeki bir mahalle. Meadow View: Çayır veya açık alanlara bakan bir yer. New Park Village: Yeni yerleşim alanları ve yeşil alanlar içerir. Brookdale: Küçük bir su yolu kenarındaki mahalle. Southwest Iowa State University: Üniversite çevresindeki bir bölge.

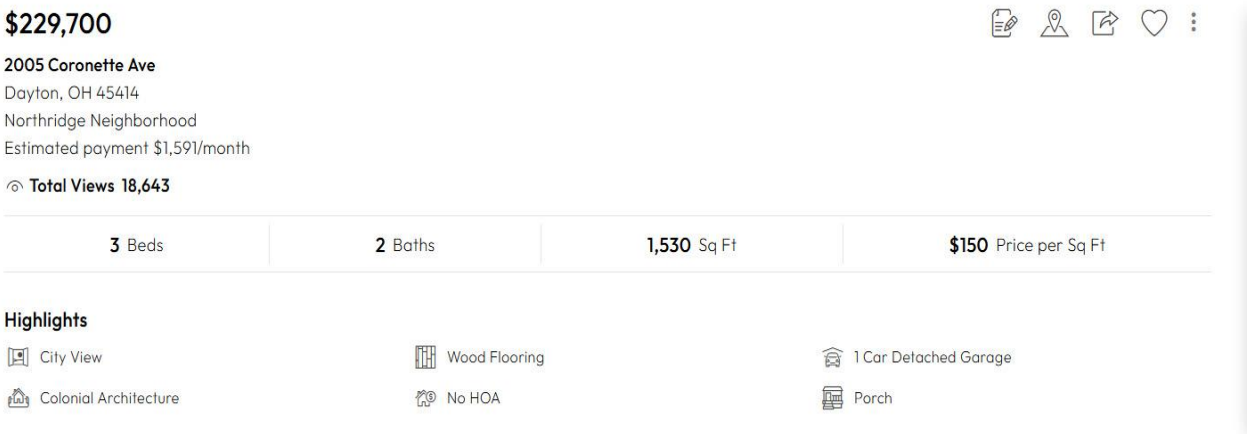
Bina Türü (BldgType): Single Family: Tek ailelik bağımsız konut. 2-Family Conjoined: İki aile için ortaklaşa yapılmış konut. Duplex: Çift katlı, iki bağımsız konut birimi içeren bina. Townhouse: Sıra ev, bağlı evlerden oluşan yapı.

Ev Tarzı (HouseStyle): Konutun yapısal tarzı. Örneğin, tek katlı (1Story), iki katlı (2Story) vb.

Kullanıcının parametreleri anlaması için Açıklama kısmı eklenmiştir.

Gerçek Uygulama

homes.com sitesinde gerçek bir ilanla model karşılaştırıldı. Fiyatı 229.700 $ olan evi model 220.072 $ olarak tahmin etmiştir. Uygulama yapılan ilan linki: https://www.homes.com/property/2005-coronette-ave-dayton-oh/dlhf8j1pfkfn9/



$229,700

2005 Coronette Ave
Dayton, OH 45414
Northridge Neighborhood
Estimated payment $1,591/month

Total Views 18,643

3 Beds	**2** Baths	**1,530** Sq Ft	**$150** Price per Sq Ft

Highlights

City View
Wood Flooring
1 Car Detached Garage
Colonial Architecture
No HOA
Porch

Home Details

Home Type
Single Family

Est. Annual Taxes
$3,600

Year Built
Built in 1926

Lot Details
9,583 Sq Ft Lot
Privacy Fence
Vinyl Fence
Sloped Lot

Parking
1 Car Detached Garage

Home Design
Colonial Architecture
Shingle Roof
Vinyl Siding

Interior Spaces
1,530 Sq Ft Home
2-Story Property
Woodwork
French Doors
Family Room with Fireplace
Wood Flooring
City Views
Basement Fills Entire Space Under The House

Kitchen
Eat-In Kitchen
Oven or Range
Microwave
Dishwasher
Solid Wood Cabinet
Disposal

Bedrooms and Bathrooms
3 Bedrooms
2 Full Bathrooms
Bathtub with Shower

Laundry
Dryer
Washer

Outdoor Features
Patio
Porch

Utilities
Forced Air Heating and Cooling System
Gas Water Heater

Community Details
No Home Owners Association
Garden City Add 02 Subdivision

See All MLS Data

Konut Fiyatı Tahmin Uygulaması

LotFrontage (m²):

141 m²

LotArea (m²):

9543

Genel Kalite (OverallQual):

5 - Çok İyi

Genel Durum (OverallCond):

4 - İyi

Yapım Yılı (YearBuilt):

1925

Yenileme Yılı (YearRemodAdd):

2005

MSZoning:

Düşük Yoğunluklu

Cadde Türü (Street):

Asfalt

Lot Şekli (LotShape):

Düzgün

Lot Şekli (LotShape):

Düzgün

Toprak Konturu (LandContour):

Düz

Yararlı Alt Yapı (Utilities):

Tam Alt Yapı

Lot Konfigürasyonu (LotConfig):

İç

Mahalle (Neighborhood):

NoRidge

Bina Türü (BldgType):

1Fam

Ev Tarzı (HouseStyle):

Tek Katlı

Tahmin Et

Tahmin Edilen Fiyat: 220072 $

Kaynakça

Pandas:

McKinney, W. (2010). *Data structures for statistical computing in Python*. In 9th Python in Science Conference (pp. 56-61). https://doi.org/10.25080/Majora-92bf1922-00a

Scikit-learn:

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830. http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf

Gradient Boosting Regressor:

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics, 29*(5), 1189-1232. https://doi.org/10.1214/aos/1013203451

SimpleImputer (Eksik Veri Doldurma):

Scikit-learn developers. (2021). *SimpleImputer* (Version 0.24.2). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html

StandardScaler:

Scikit-learn developers. (2021). *StandardScaler* (Version 0.24.2). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

OneHotEncoder:

Scikit-learn developers. (2021). *OneHotEncoder* (Version 0.24.2). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Train-Test Split:

Scikit-learn developers. (2021). *train_test_split* (Version 0.24.2). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

Flask:

Pallets Projects. (n.d.). *Flask documentation*. Retrieved January 7, 2025, from https://flask.palletsprojects.com/en/stable/

Pickle:

Python Software Foundation. (2020). *Pickle — Python object serialization*. Retrieved from https://docs.python.org/3/library/pickle.html

Flask-CORS (Cross-Origin Resource Sharing):

Flask-CORS contributors. (2020). *Flask-CORS: A Flask extension for handling Cross-Origin Resource Sharing (CORS), making cross-origin AJAX possible*. Retrieved from https://flask-cors.readthedocs.io/en/latest/

React:

React. (2021). *React – A JavaScript library for building user interfaces*. https://reactjs.org

Axios:

Axios. (2021). *Axios – Promise based HTTP client for the browser and Node.js*. https://axios-http.com

Veri Seti:

https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview